HITACHI®

HD64180 8-BIT MPU HARDWARE MANUAL

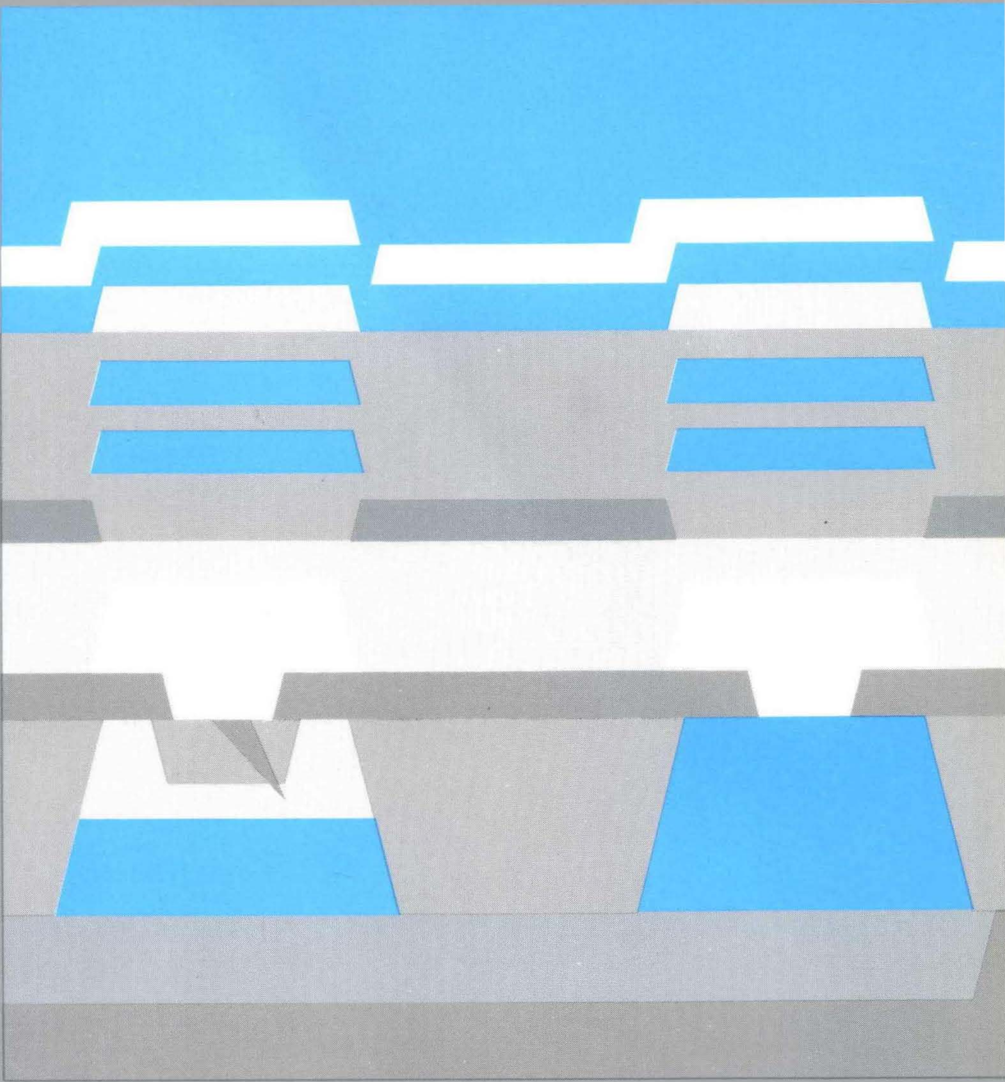
#U77

1/88

January, 1988

HITACHI®

HD64180
8-BIT MICROPROCESSOR
HARDWARE MANUAL



#U77

High-Integration 180 Family

HD64180
8-BIT MICROPROCESSOR
HARDWARE MANUAL

HD64180R
HD64180Z

#U77

HITACHI®

ADE-602-002B

MEDICAL APPLICATIONS

Hitachi's products are not authorized for use in MEDICAL APPLICATIONS, including, but not limited to, use in life support devices without the written consent of the appropriate officer of Hitachi's sales company. Buyers of Hitachi's products are requested to notify Hitachi's sales offices when planning to use the products in MEDICAL APPLICATIONS.

When using this manual, the reader should keep the following in mind:

1. This manual may, wholly or partially, be subject to change without notice.
2. All rights reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this manual without Hitachi's permission.
3. Hitachi will not be responsible for any damage to the user that may result from accidents or any other reasons during operation of his unit according to this manual.
4. This manual neither ensures the enforcement of any industrial properties or other rights, nor sanctions the enforcement right thereof.
5. Circuitry and other examples described herein are meant merely to indicate characteristics and performance of Hitachi semiconductor-applied products. Hitachi assumes no responsibility for any patent infringements or other problems resulting from applications based on the examples described herein.
6. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi, Ltd.

February 1988

Printed in U.S.A.

Preface

Based on a microcoded execution unit and advanced CMOS manufacturing technology, the HD64180 is an 8-bit MPU which provides the benefits of high performance, reduced system cost and low power operation while maintaining compatibility with the large base of industry standard 8-bit software.

Performance is improved by virtue of high operating frequency, pipelining, enhanced instruction set and an integrated Memory Management Unit (MMU) with 1M or 512k bytes memory physical address space.

System cost is reduced by incorporating key system functions on-chip including the MMU, two channel Direct Memory Access Controller (DMAC), wait state generator, dynamic RAM refresh, two channel Asynchronous Serial Communication Interface (ASCI), Clocked Serial I/O Port (CSI/O), two channel 16-bit Programmable Reload Timer (PRT), Versatile 12 source interrupt controller and a 'dual' (68× ×, 80× ×) bus interface.

This manual describes the HD64180R0, HD64180R1, HD64180Z, hardware architecture and is combined with the programming manual for the entire HD64180 series.

NOTES:

1. HD64180R0 mask has been superceded by the HD64180R1 mask level. See Page 3 for R1 part no. ordering codes.
2. 4 MHz versions are currently not available in the U.S.

TABLE OF CONTENTS

APPENDIX

Figures

HD64180
HIGH INTEGRATION CMOS MPU

Based on a microcoded execution unit and advanced CMOS manufacturing technology, the HD64180 is an 8-bit MPU which provides the benefits of high performance, reduced system cost and low power operation while maintaining compatibility with the large base of industry standard 8-bit software.

Performance is improved by virtue of high operating frequency, pipelining, enhanced instruction set and an integrated Memory Management Unit (MMU) with 1M or 512k bytes memory physical address space.

System cost is reduced by incorporating key system functions on-chip including the MMU, two channel Direct Memory Access Controller (DMAC), wait state generator, dynamic RAM refresh, two channel Asynchronous Serial Communication Interface (ASCI), Clocked Serial I/O Port (CSI/O), two channel 16-bit Programmable Reload Timer (PRT), Versatile 12 source interrupt controller and a 'dual' (68× ×, 80× ×) bus interface.

Low power consumption during normal CPU operation is supplemented by two specific software controlled low power operation modes.

The HD64180, when combined with CMOS VLSI memories and peripherals, is useful in system applications requiring high performance, battery power operation and standard software compatibility.

HD64180 is fully compatible with Z64180 which is marketed by Zilog Inc.

High Performance, High Integration CPU.

- Operating Frequency to 8 MHz (R1 and Z mask) and 6 MHz (R0 mask).
- On-Chip MMU Supports 1M or 512k Bytes Memory and 64k Bytes I/O Address Space.
- Two Channel DMAC With Memory to/from Memory, Memory to/from I/O and Memory to/from Memory Mapped I/O Transfer Capability.
- $\overline{\text{WAIT}}$ Input and Wait State Generator for Slow Memory and I/O Device Interface.
- Programmable Dynamic RAM Refresh Addressing and Timing.
- Two Channel, Full Duplex Asynchronous Serial Communication Interface (ASCI) with Programmable Baud Rate Generator and Modem Control Handshake Signals.
- Clocked Serial I/O Port (CSI/O) with High Speed Operation (200k Bits/Second at 4 MHz).
- Two Channel 16-bit Programmable Reload Timer (PRT) for Counting, Timing and Output Waveform Generation.
- Versatile Interrupt Controller Manages Four External and Eight Internal Interrupt Sources.
- 'Dual Bus' Interface Compatible With All Standard Memory and Peripheral LSI.
- On-chip Clock Generator.

Enhanced Standard 8-bit Software Architecture.

- Fully Compatible with CP/M-80, CP/M Plus** and Existing System and Application Software.
- Seven new Instructions including Multiply.
- On-chip I/O Address Relocation Register for Board Level Compatibility with Existing Systems and Software.
- SLEEP mode and SYSTEM STOP mode for Low Power Operation.

VLSI CMOS Process Technology.

- Low Power Operation — 100 mW at 8 MHz Operation.
  25 mW SYSTEM STOP mode at 8 MHz operation.
- V_{CC} = 5V ± 10% — Fully TTL Compatible.

** CP/M-80 and CP/M plus are registered trademarks of Digital Research, Inc.

Type of Products

HD64180R0

Part No.	Clock Frequency (MHz)	Package Type	Address Space
HD64A180R0P	4	DP-64S	512k Byte
HD64B180R0P	6		
HD64A180R0F	4	FP-80	512k Byte
HD64B180R0F	6		
HD64A180R0CP	4	CP-68	512k Byte
HD64B180R0CP	6		

HD64180R1

Part No.	Clock Frequency (MHz)	Package Type	Address Space
HD64180RP-6	6	DP-64S	512k Byte
HD64180RP-8	8		
HD64180RF-6X	6	FP-80	1M Byte
HD64180RF-8X	8		
HD64180RCP-6X	6	CP-68	1M Byte
HD64180RCP-8X	8		

NOTES:

1. HD64180R0 mask has been superceded by the HD64180R1 mask level. See above for R1 part no. ordering codes.
2. 4 MHz versions are currently not available in the U.S.

Part No.	Clock Frequency (MHz)	Package Type	Address Space
HD64180ZP-6	6	DP-64S	512k Byte
HD64180ZP-8	8		
HD64180ZF-6X	6	FP-80	1M Byte
HD64180ZF-8X	8		
HD64180ZCP-6X	6	CP-68	1M Byte
HD64180ZCP-8X	8		

1. HD64180 OVERVIEW

1.1 Block Diagram

The HD64180 combines a high performance CPU core with many of the systems and I/O resources required by a broad range of applications.

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)

Block Diagram

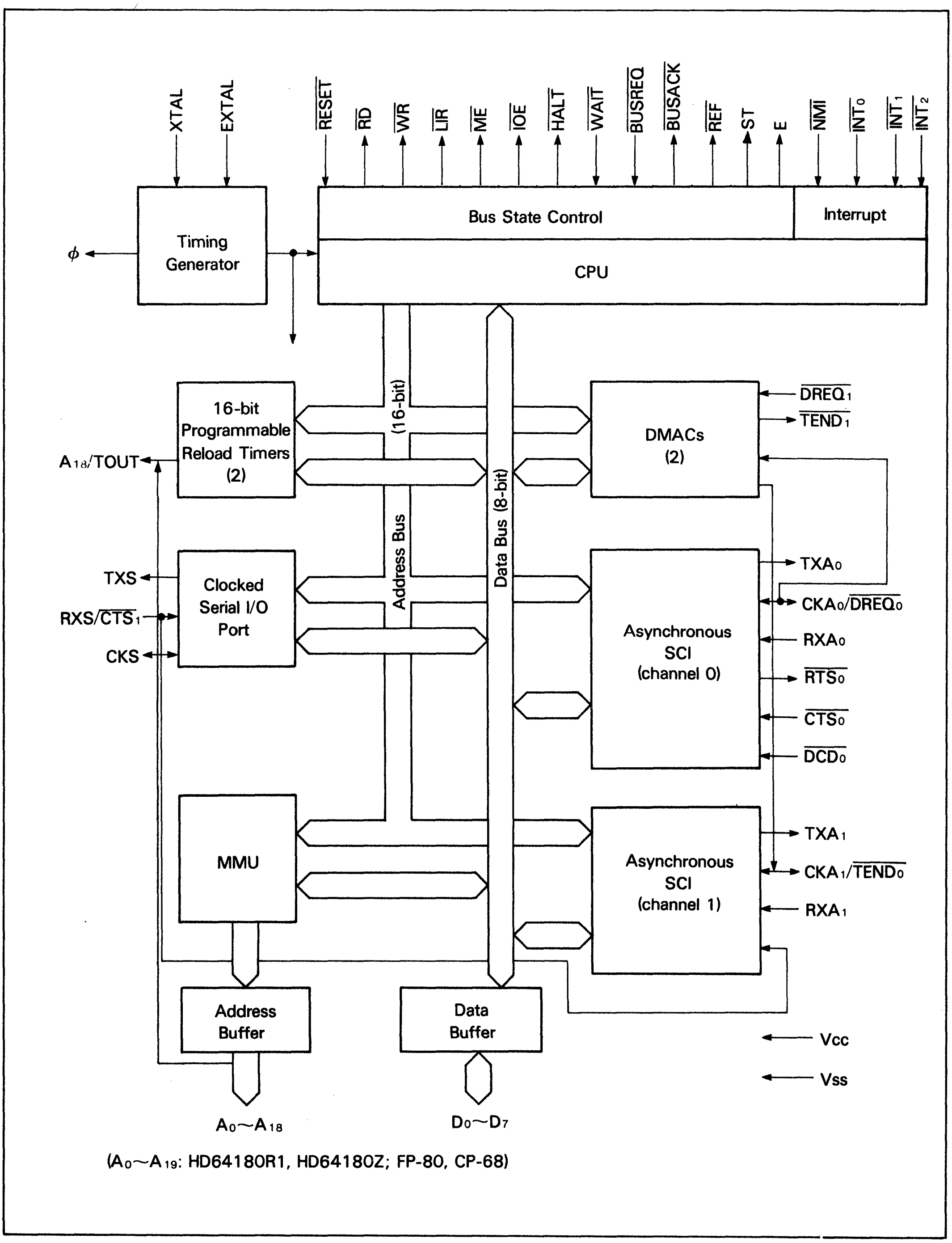
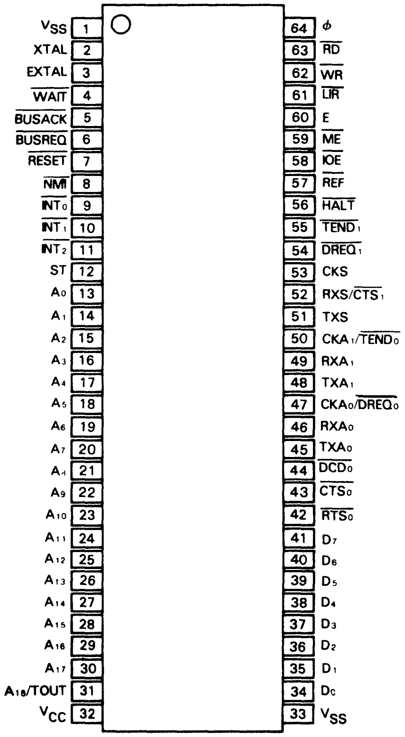


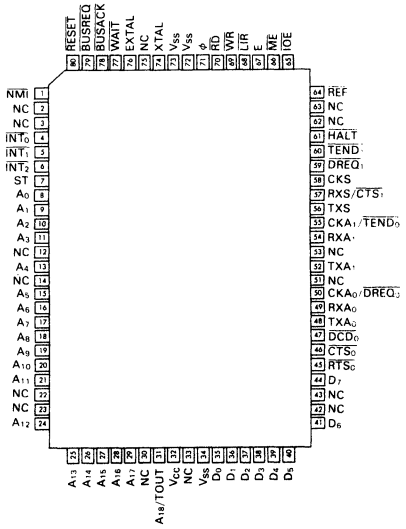
Figure 1.1.1 Block Diagram

1.2 Pin Assignment (Top View)

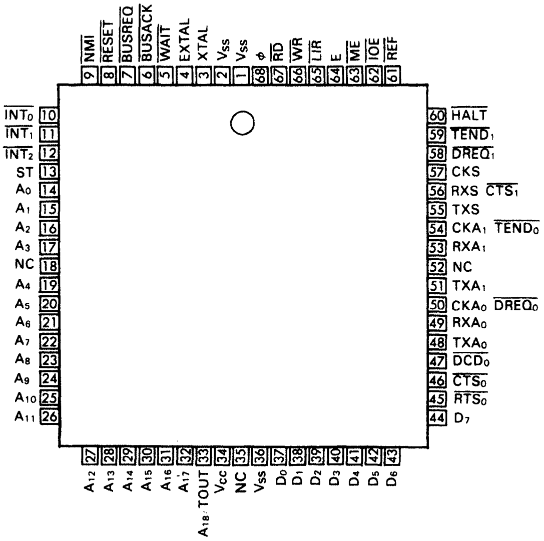
HD64180R0



(DP-64S)

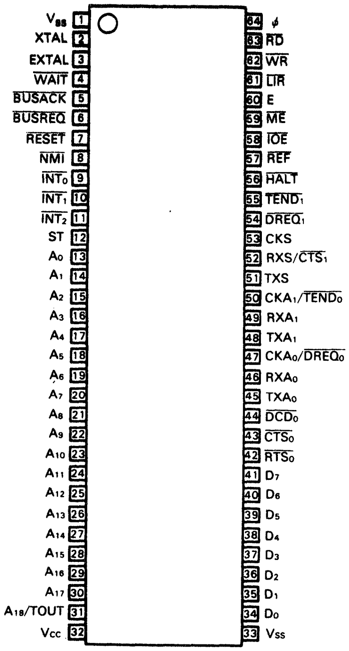


(FP-80)

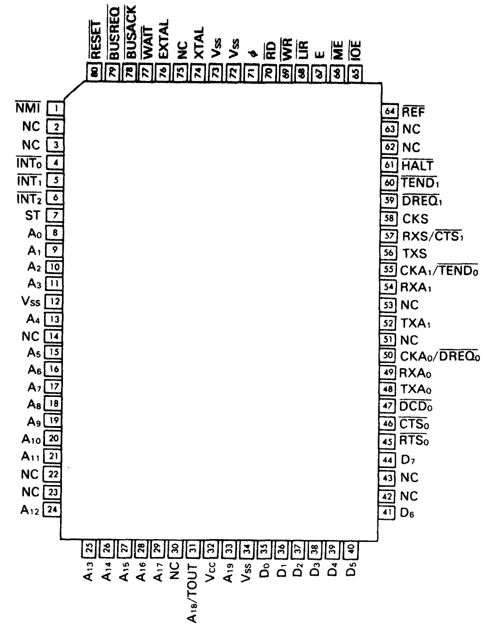


(CP-68)

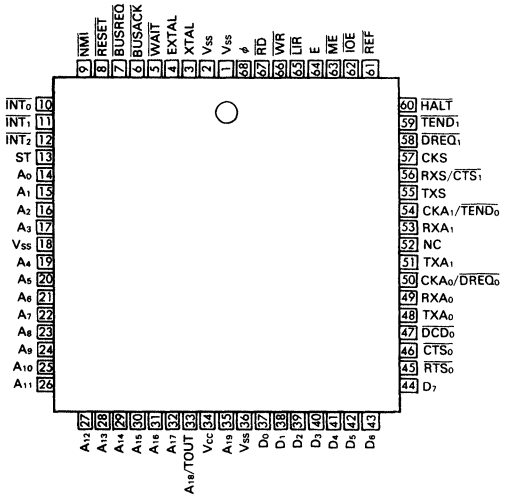
NC: Not connected.
Please leave the
NC pins open.



(DP-64S)

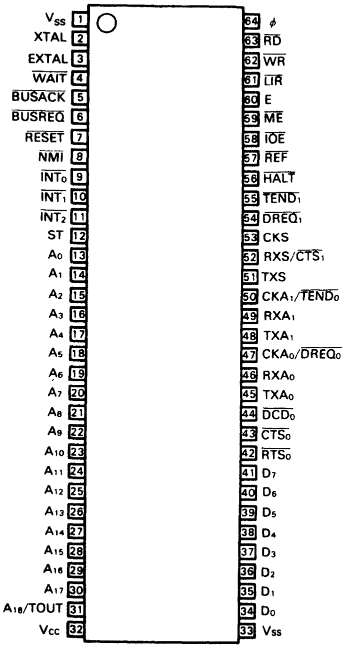


(FP-80)

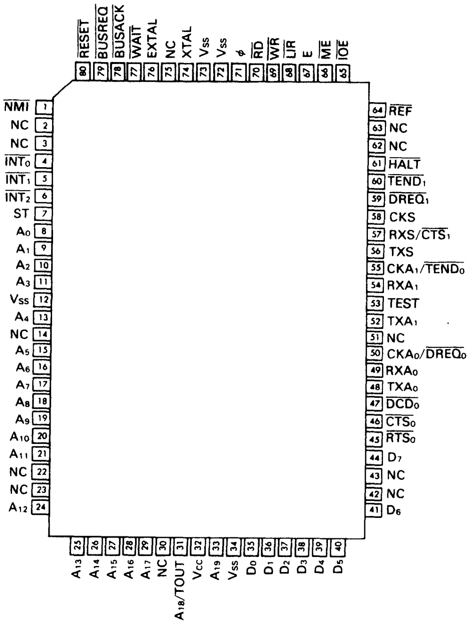


NC: Not connected.
Please leave the
NC pins open.

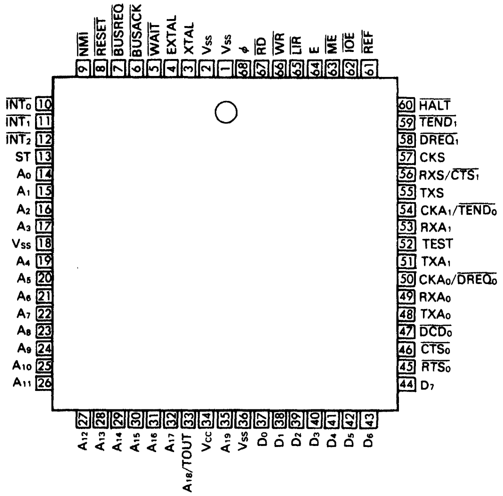
(CP-68)



(DP-64S)



(FP-80)



Please leave the TEST pin open.

(CP-68)

1.3 CPU Architecture

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

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.

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.

Interrupt Controller

Monitors and prioritizes the four external and eight internal interrupt sources. A variety of interrupt response modes are programmable.

Memory Management Unit (MMU)

Maps the CPU 64k bytes logical memory address space into a 1M or 512k bytes 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 (64k bytes I/O address space) bypass the MMU.

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 I/O Resources

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 1M or 512k bytes physical memory address space (the MMU is bypassed during DMA) and transfers (up to 64k bytes in length) can cross 64k bytes boundaries. See Fig. 2.9.1 for further details.

Asynchronous Serial Communication Interface (ASCI)

The ASCI provides two separate full duplex UARTs and includes 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 Fig. 2.10.1 for further details.

Clocked Serial I/O Port (CSI/O)

The CSI/O provides a half duplex clocked serial transmitter and receiver. This can be used for simple, high speed connection to another microprocessor or microcomputer. See Fig. 2.11.1 for further details.

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 divided by 20 (fixed) from the system clock and PRT channel 1 has an optional output allowing waveform generation. See Fig. 2.12.1 for further details.

2. HD64180 HARDWARE ARCHITECTURE

2.1 Signal Description

XTAL (IN)

Crystal oscillator connection. Should be left open if an external TTL clock is used. It is noted this input is not a TTL level input. See Table D.C. characteristics.

EXTAL (IN)

Crystal oscillator connection. An external TTL clock can be input on this line. This input is schmitt triggered.

ϕ (OUT)

System Clock. The frequency is equal to one-half of crystal oscillator.

$\overline{\text{RESET}}$ – CPU Reset (IN)

When LOW, initializes the HD64180 CPU. All output signals are held inactive during RESET.

A_0-A_{19} – Address Bus (OUT, 3-STATE)
A_{18}/TOUT

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. A_{18} is multiplexed with the TOUT output from PRT channel 1. During RESET, the address function is selected. TOUT function can be selected under software control.

D_0-D_7 – Data Bus (IN/OUT, 3-STATE)

Bidirectional 8-bit data bus. The 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.

$\overline{\text{RD}}$ – Read (OUT, 3-STATE)

Used during a CPU read cycle to enable transfer from the external memory or I/O device to the CPU data bus.

$\overline{\text{WR}}$ – Write (OUT, 3-STATE)

Used during a CPU write cycle to enable transfer from the CPU data bus to the external memory or I/O device.

$\overline{\text{ME}}$ – Memory Enable (OUT, 3-STATE)

Indicates memory read or write operation. The HD64180 asserts $\overline{\text{ME}}$ LOW in the following cases.

(a) When fetching instructions and operands.
(b) When reading or writing memory data.
(c) During memory access cycles of DMA.
(d) During dynamic RAM refresh cycles.

$\overline{\mathrm{IOE}}$ — I/O Enable (OUT, 3-STATE)

Indicates I/O read or write operation. The HD64180 asserts $\overline{\mathrm{IOE}}$ LOW in the following cases.

(a) When reading or writing I/O data.
(b) During I/O access cycles of DMA.
(c) During $\overline{\mathrm{INT}_0}$ acknowledge cycle

$\overline{\mathrm{WAIT}}$ — Bus Cycle Wait (IN)

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 $\overline{\mathrm{WAIT}}$ input is sampled HIGH at the falling edge of Tw, at which time the bus cycle will proceed to completion.

E — Enable (OUT)

Synchronous clock for connection to HD63×× series and other 6800/6500 series compatible peripheral LSIs.

$\overline{\mathrm{BUSREQ}}$ — Bus Request (IN)

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

$\overline{\mathrm{BUSACK}}$ — Bus Acknowledge (OUT)

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

$\overline{\mathrm{HALT}}$ — Halt/Sleep Status (OUT)

Asserted LOW after execution of the HALT or SLP instructions. Used with $\overline{\mathrm{LIR}}$ and ST output pins to encode CPU status.

$\overline{\mathrm{LIR}}$ — Load Instruction Register (OUT)

Asserted LOW when the current cycle is an op-code fetch cycle. Used with $\overline{\mathrm{HALT}}$ and ST output pins to encode CPU status.

ST — Status (OUT) [12]

Used with the $\overline{\mathrm{HALT}}$ and $\overline{\mathrm{LIR}}$ output pins to encode CPU status.

Table 2.1.1 Status Summary

ST	$\overline{\text{HALT}}$	$\overline{\text{LIR}}$	Operation
0	1	0	CPU operation (1st op-code fetch)
1	1	0	CPU operation (2nd op-code and 3rd op-code fetch)
1	1	1	CPU operation (MC except for op-code 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

$\overline{\text{REF}}$ – Refresh (OUT)

When LOW, indicates the CPU is in the dynamic RAM refresh cycle and the low-order 8 bits (A_0-A_7) of the address bus contain the refresh address.

$\overline{\text{NMI}}$ – Non-Maskable Interrupt (IN)

When edge transition from HIGH to LOW is detected, 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.

$\overline{\text{INT}_0}$ – Maskable Interrupt Level 0 (IN)

When LOW, 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.

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 vector table address fetched from data bus.

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

$\overline{\text{INT}_1}$, $\overline{\text{INT}_2}$ – Maskable Interrupt Level 1, 2 (IN)

When LOW, requests a CPU interrupt (unless masked) and saves 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 $\overline{\text{INT}_0}$.

$\overline{\mathrm{DREQ}_0}$ – DMA Request – Channel 0 (IN)

When LOW (programmable edge or level sense), requests DMA transfer service from channel 0 of the HD64180 DMAC. $\overline{\mathrm{DREQ}_0}$ is used for Channel 0 memory to/from I/O and memory to/from memory mapped I/O transfers. $\overline{\mathrm{DREQ}_0}$ is not used for memory to/from memory transfers. This pin is multiplexed with CKA_0.

$\overline{\mathrm{TEND}_0}$ – Transfer End – Channel 0 (OUT)

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.

$\overline{\mathrm{DREQ}_1}$ – DMA Request – Channel 1 (IN)

When LOW (programmable edge or level sense), requests DMA transfer service from channel 1 of the HD64180 DMAC. Channel 1 supports Memory to/from I/O transfers.

$\overline{\mathrm{TEND}_1}$ – Transfer End – Channel 1 (OUT)

Asserted LOW synchronous with the last write cycle of channel 1 DMA transfer to indicate DMA completion to an external device.

TXA_0 – Asynchronous Transmit Data – Channel 0 (OUT)

Asynchronous transmit data from channel 0 of the Asynchronous Serial Communication Interface (ASCI).

RXA_0 – Asynchronous Receive Data – Channel 0 (IN)

Asynchronous receive data to channel 0 of the ASCI.

CKA_0 – Asynchronous Clock – Channel 0 (IN/OUT)

Clock input/output for channel 0 of the ASCI. This pin is multiplexed (software selectable) with $\overline{\mathrm{DREQ}_0}$.

$\overline{\mathrm{RTS}_0}$ – Request to Send – Channel 0 (OUT)

Programmable modem control output signal for channel 0 of the ASCI.

$\overline{\mathrm{CTS}_0}$ – Clear to Send – Channel 0 (IN)

Modem control input signal for channel 0 of the ASCI.

$\overline{\mathrm{DCD}_0}$ – Data Carrier Detect – Channel 0 (IN)

Modem control input signal for channel 0 of the ASCI.

TXA_1 – Asynchronous Transmit Data – Channel 1 (OUT)

Asynchronous transmit data from channel 1 of the ASCI.

RXA_1 – Asynchronous Receive Data – Channel 1 (IN)

Asynchronous receive data to channel 1 of the ASCI.

$\mathrm{CKA_1}$ — Asynchronous Clock — Channel 1 (IN/OUT)

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

$\overline{\mathrm{CTS_1}}$ — Clear to Send — Channel 1 (IN)

Modem control input signal for channel 1 of the ASCI. This pin is multiplexed (software selectable) with RXS.

TXS — Clocked Serial Transmit Data (OUT)

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

RXS — Clocked Serial Receive Data (IN)

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

CKS — Serial Clock (IN/OUT)

Input or output clock for the CSI/O.

TOUT — Timer Output (OUT)

Pulse output from Programmable Reload Timer channel 1. This pin is multiplexed (software selectable) with $\mathrm{A_{18}}$ (Address 18).

Vcc — Power Supply

Vss — Ground

Multiplexed pin descriptions

$\mathrm{A_{18}}$/TOUT — During RESET, this pin is initialized as $\mathrm{A_{18}}$ pin. If either TOC1 or TOC0 bit in Timer Control Register (TCR) is set to 1, TOUT function is selected.
If TOC1 and TOC0 bits are cleared to 0, $\mathrm{A_{18}}$ function is selected.

$\mathrm{CKA_0}$/$\overline{\mathrm{DREQ_0}}$ — During RESET, this pin is initialized as $\mathrm{CKA_0}$ pin. If either DM1 or SM1 in DMA Mode Register (DMODE) is set to 1, $\overline{\mathrm{DREQ_0}}$ function is always selected.

$\mathrm{CKA_1}$/$\overline{\mathrm{TEND_0}}$ — During RESET, this pin is initialized as $\mathrm{CKA_1}$ pin. If $\overline{\mathrm{CKA1D}}$ bit in ASCI control register ch 1 (CNTLA1) is set to 1, $\overline{\mathrm{TEND_0}}$ function is selected. If $\overline{\mathrm{CKA1D}}$ bit is set to 0, $\mathrm{CKA_1}$ function is selected.

RXS/$\overline{\mathrm{CTS_1}}$ — During RESET, this pin is initialized as RXS pin. If CTS1E bit in ASCI status register ch1 (STAT1) is set to 1, $\overline{\mathrm{CTS_1}}$ function is selected.
If CTS1E bit is set to 0, RXS function is selected.

Pin Function Differences in the HD64180 Series

Package type	Pin No.	HD64180R0	HD64180R1	HD64180Z
CP-68	18	NC	V_{SS}	V_{SS}
	35	NC	A_{19}	A_{19}
	52	NC	NC	TEST
FP-80	12	NC	V_{SS}	V_{SS}
	33	NC	A_{19}	A_{19}
	53	NC	NC	TEST

2.2 CPU Bus Timing

This section explains the HD64180 CPU timing for the following operations.

(1) Instruction (op-code) 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) $\overline{\text{BUSREQ}}$/$\overline{\text{BUSACK}}$ 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 while accessing memory or I/O, or it consists of one system clock, Ti while the CPU internal operation. The system clock (ϕ) is half frequency of crystal oscillation (Ex. 8 MHz crystal ⟶ ϕ of 4 MHz, 250 nsec). For interfacing to slow memory or peripherals, optional wait states (Tw) may be inserted between T_2 and T_3.

2.2.1 Instruction (op-code) fetch timing

Fig. 2.2.1 shows the instruction (op-code) fetch timing with no wait states.

An op-code fetch cycle is externally indicated when the $\overline{\text{LIR}}$ (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 HD64180 on-chip MMU.

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 op-code on the data bus is latched at the rising edge of T_3 and the bus cycle terminates at the end of T_3.

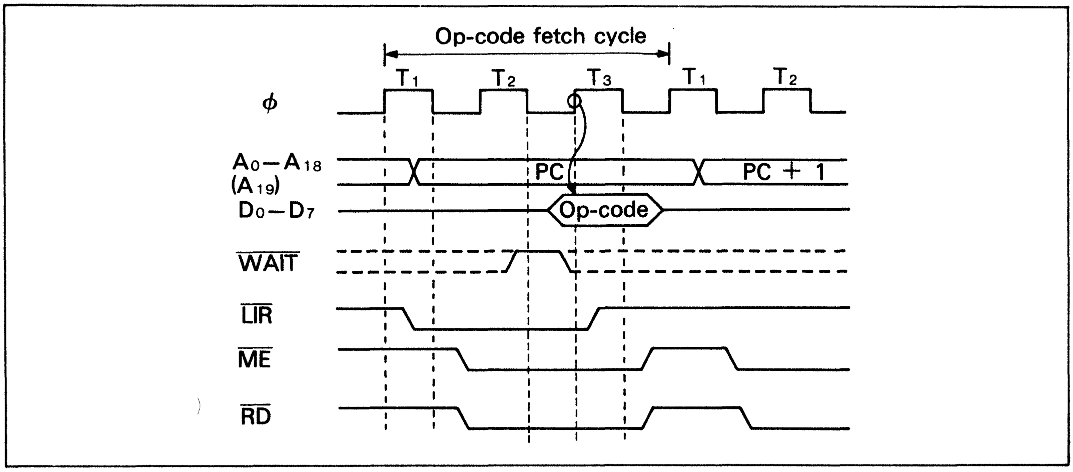


Figure 2.2.1 Op-Code Fetch Timing

Fig. 2.2.2 illustrates the insertion of wait states (Tw) into the op-code 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.

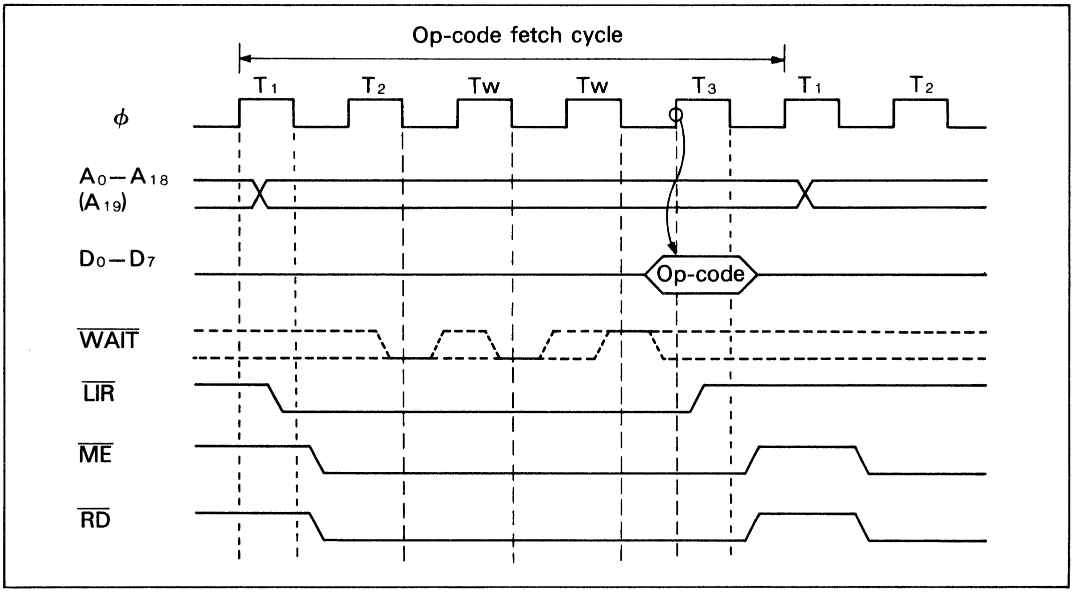


Figure 2.2.2 Op-Code Fetch Timing (with wait state)

2.2.2 Operand and data read/write timing

The instruction operand and data read/write timing differs from op-code fetch timing in two ways. First, the $\overline{\text{LIR}}$ 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 $\overline{\text{ME}}$ 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 $\overline{\text{WR}}$ signal is asserted LOW enabling the memory. $\overline{\text{ME}}$ and $\overline{\text{WR}}$ 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 op-code fetch cycles.

Fig. 2.2.3 illustrates the read/write timing without wait states (Tw), while Fig. 2.2.4 illustrates read/write timing with wait states (Tw).

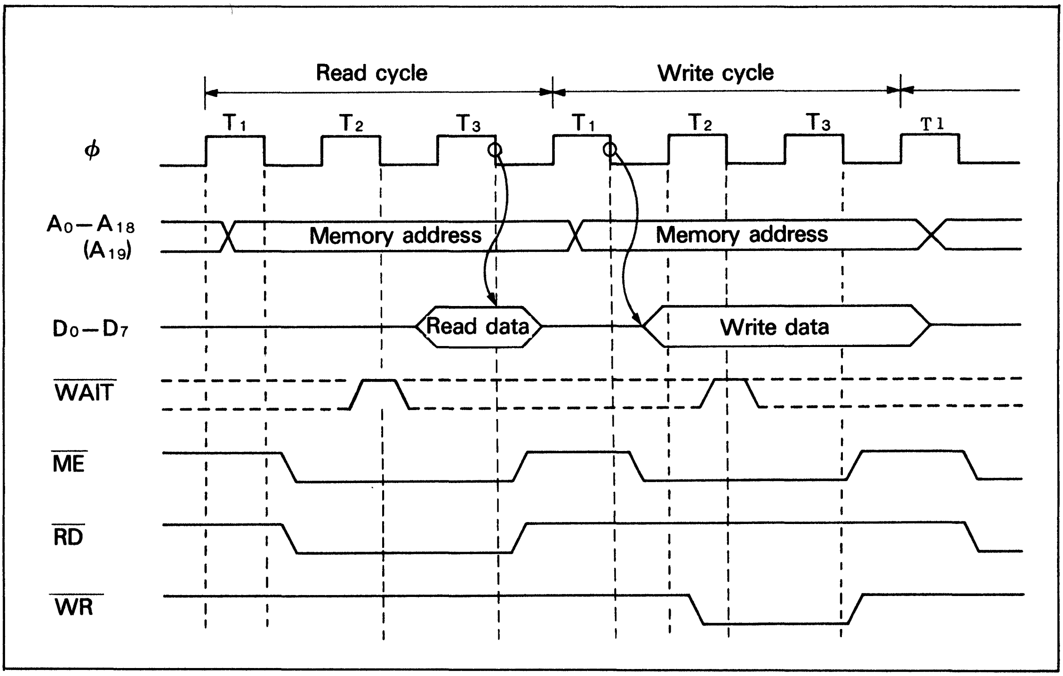


Figure 2.2.3 Memory Read/Write Timing (without wait state)

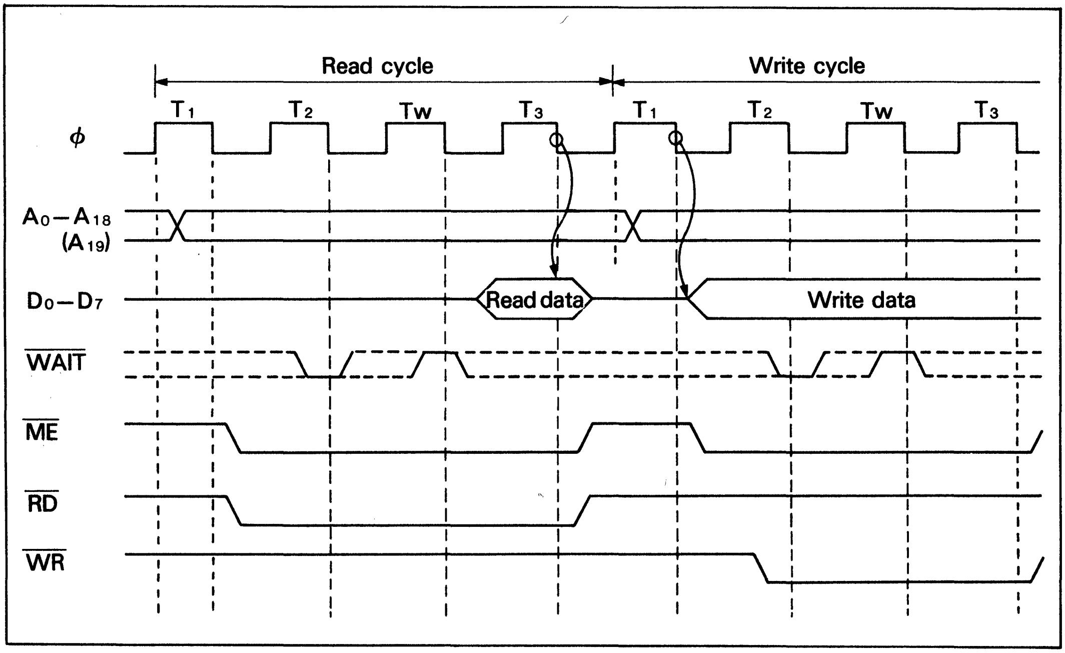
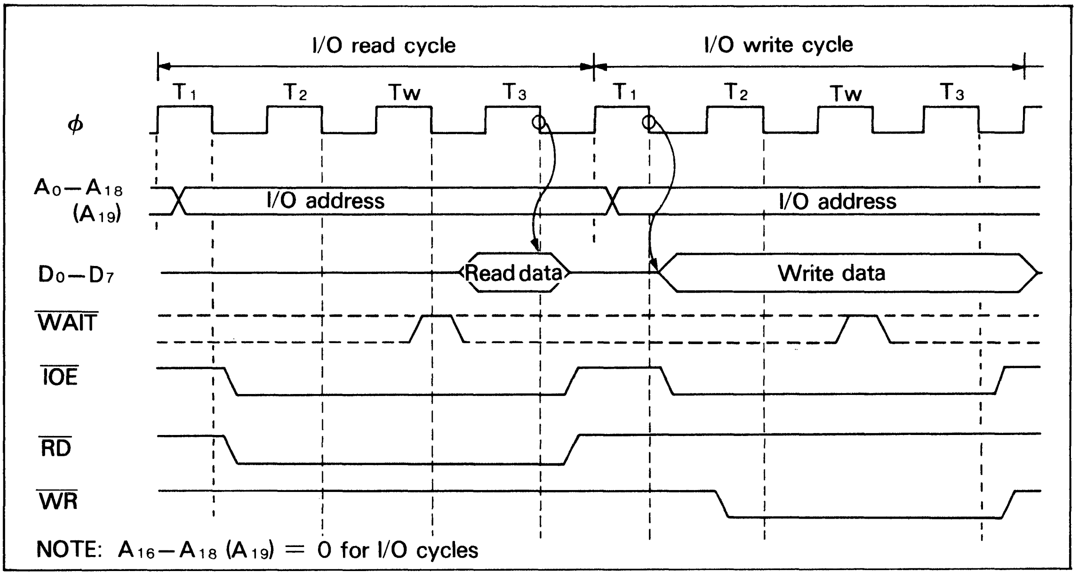


Figure 2.2.4 Memory Read/Write Timing (with wait state)

2.2.3 I/O read/write timing

I/O instructions cause data read/write transfer which differs from memory data transfer 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_{18} (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).

Fig. 2.2.5 shows I/O read/write timing with the automatically inserted wait state (Tw).



NOTE: $A_{16}-A_{18}$ (A_{19}) = 0 for I/O cycles

Figure 2.2.5 I/O Read/Write Timing

2.2.4 Basic instruction timing

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

The example in Fig. 2.2.6 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 op-code as indicated by $\overline{LIR}$ LOW. Next, the instruction operand (d) is fetched.

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

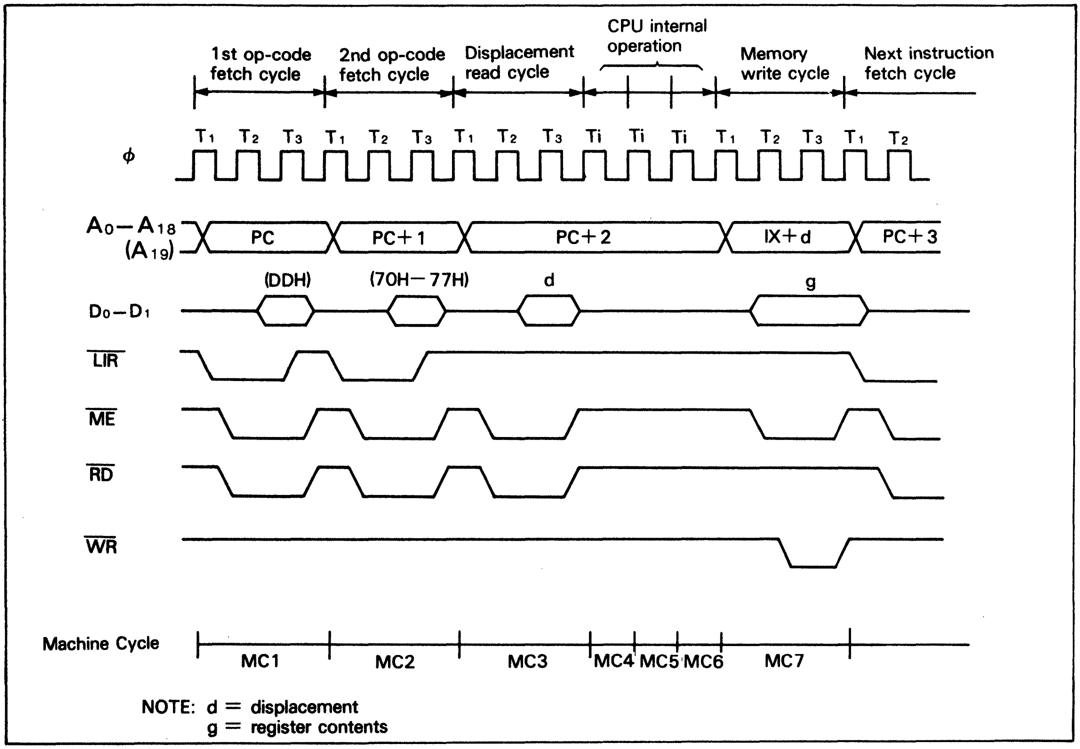


Figure 2.2.6 LD (IX+d), g Instruction Timing

2.2.5 RESET timing

Fig. 2.2.7 shows the HD64180 hardware RESET timing. If the RESET pin is LOW for six or more than six clock cycles, processing is terminated and the HD64180 restarts execution from (logical and physical) address 00000H.

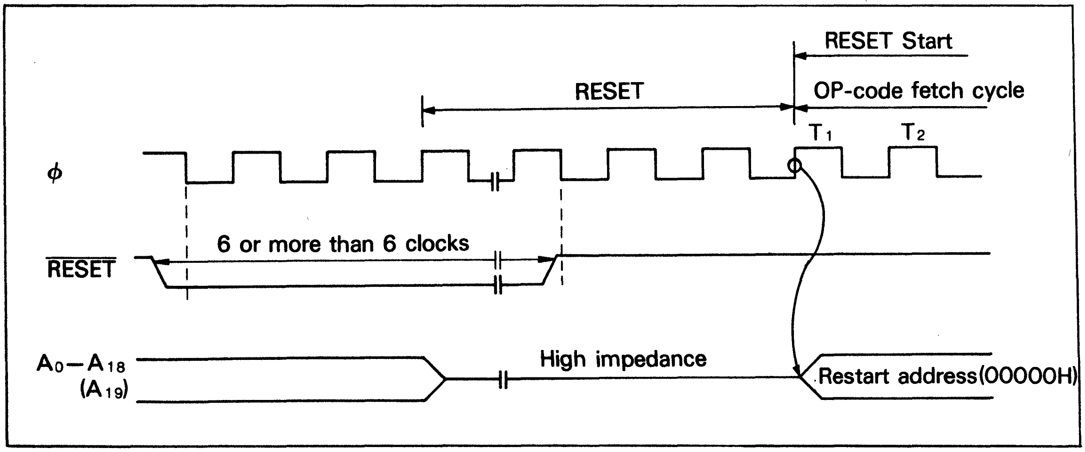


Figure 2.2.7 RESET Timing

2.2.6 $\overline{\text{BUSREQ}}/\overline{\text{BUSACK}}$ bus exchange timing

The HD64180 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 HD64180 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 HD64180 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 op-code 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_{18} (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 HD64180 has released the bus. The alternate bus master must provide dynamic memory refreshing if the bus is released for long periods of time.

Fig. 2.2.8 illustrates $\overline{\text{BUSREQ}}/\overline{\text{BUSACK}}$ bus exchange during a memory read cycle. Fig. 2.2.9 illustrates bus exchange when the bus release is requested during an HD64180 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.

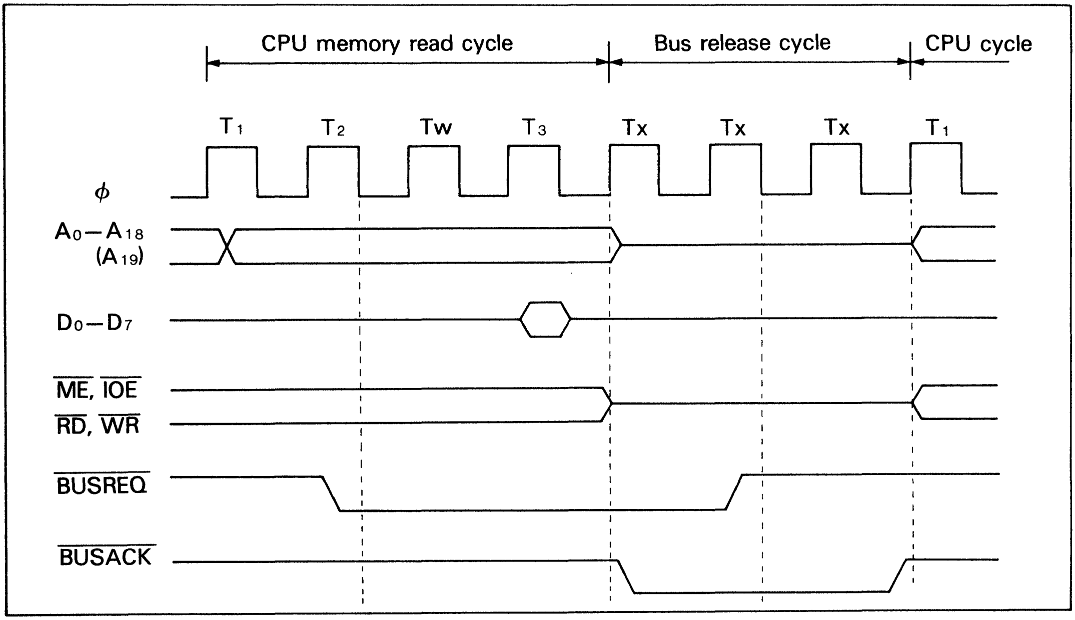


Figure 2.2.8 Bus Exchange Timing (1)

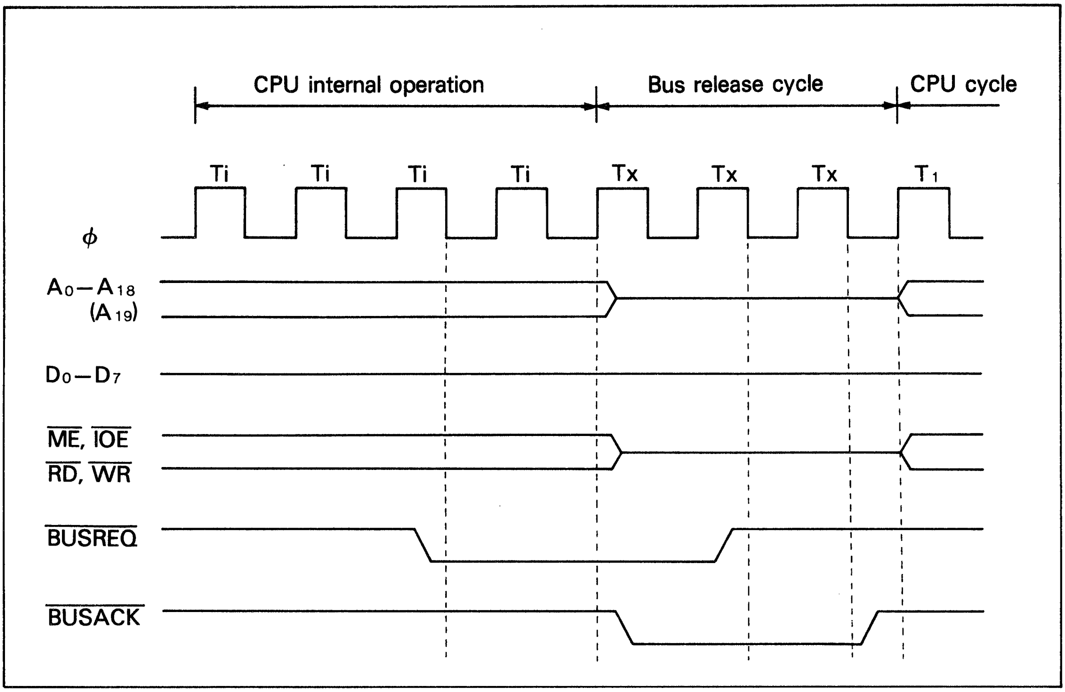

CPU internal operation
Bus release cycle
CPU cycle
Ti
Ti
Ti
Ti
Tx
Tx
Tx
T_1
ϕ
A_0-A_{18}
(A_{19})
D_0-D_7
$\overline{\text{ME}}$, $\overline{\text{IOE}}$
$\overline{\text{RD}}$, $\overline{\text{WR}}$
$\overline{\text{BUSREQ}}$
$\overline{\text{BUSACK}}$

Figure 2.2.9 Bus Exchange Timing (2)

2.2.7 Z80-Type Bus Interface

Users can enable or disable the $\overline{\text{LIR}}$ output and control the timing of the $\overline{\text{IOE}}$ and $\overline{\text{RD}}$ signals by software. The operation of the RETI instruction of the Z Mask is different from that of the R1 Mask. The details are described below. These features enable HD64180 to connect with directly Z80 peripherals.

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

Operation Mode Control Register

The $\overline{\text{LIR}}$, $\overline{\text{IOE}}$, and $\overline{\text{RD}}$ signals are controlled through the Operation Mode Control Register. The register is newly added in the Z Mask.

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

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

○ bit 7: LIRE (LIR Enable)

LIRE controls the $\overline{\text{LIR}}$ output and is set to 1 during RESET.

(a) LIRE =1:

The $\overline{\text{LIR}}$ output is the same as that of R1 version and is asserted low in the following cases:

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

(b) 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 op-code fetch cycle of RETI (Please see (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.

○ bit 6: $\overline{\text{LIRTE}}$ ($\overline{\text{LIR}}$ Temporary Enable)

$\overline{\text{LIRTE}}$ activates the 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.

(a) $\overline{\text{LIRTE}}$ set to 1:

There is no effect and the $\overline{\text{LIR}}$ output is subject to the LIRE bit.

(b) $\overline{\text{LIRTE}}$ 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 op-code fetch cycle just after 0 is written to $\overline{\text{LIRTE}}$ bit. The timing is shown in figure 2.2.10.

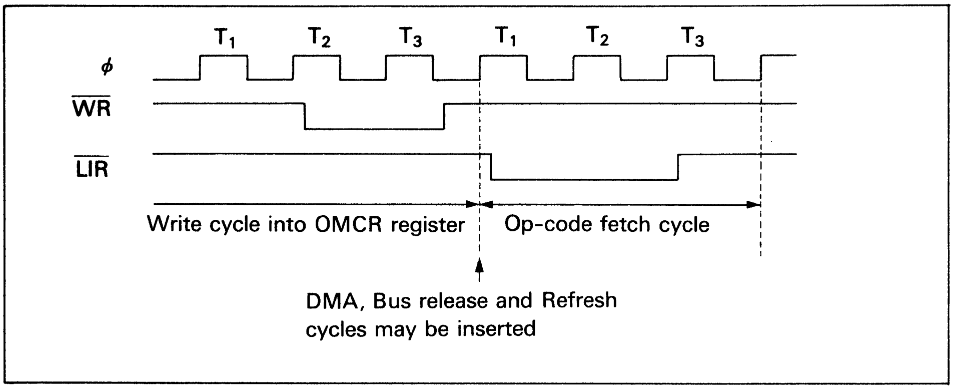


Figure 2.2.10 Writing 0 to $\overline{\text{LIRTE}}$ When LIRE=0

○ bit 5: $\overline{\mathrm{IOC}}$ (I/O Compatibility)

$\overline{\mathrm{IOC}}$ controls the $\overline{\mathrm{IOE}}$ and $\overline{\mathrm{RD}}$ output and is set to 1 during RESET.

(a) $\overline{\mathrm{IOC}}$ =1 (The $\overline{\mathrm{IOE}}$ and $\overline{\mathrm{RD}}$ outputs are the same as those of the R1 Mask.)

In an I/O read cycle, $\overline{\mathrm{IOE}}$ and $\overline{\mathrm{RD}}$ signals go to low at a falling edge of T_1. The timing is shown in figure 2.2.11.

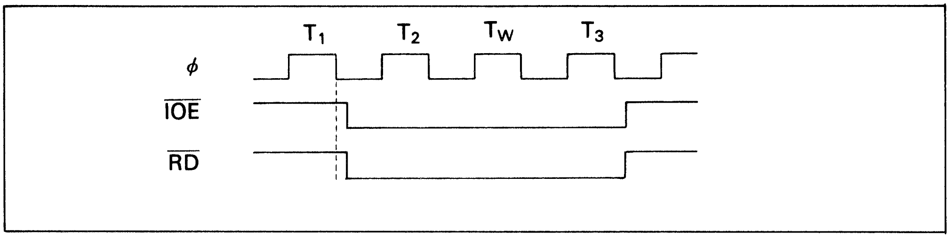


Figure 2.2.11 I/O Read Cycle When $\overline{\mathrm{IOC}}$=1

In an I/O write cycle, $\overline{\mathrm{IOE}}$ signal goes to low at a falling edge of T_1. The timing is shown in figure 2.2.12.

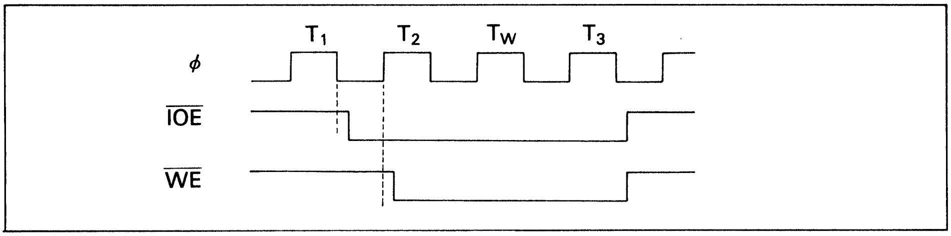


Figure 2.2.12 I/O Write Cycle When $\overline{\mathrm{IOC}}$=1

(b) $\overline{\mathrm{IOC}}$ =0 (The $\overline{\mathrm{IOE}}$ and $\overline{\mathrm{RD}}$ outputs are compatible with the Z80's peripheral LSIs.)

In an I/O read cycle, $\overline{\mathrm{IOE}}$ and $\overline{\mathrm{RD}}$ signals go to low at a rising edge of T_2. The timing is shown in figure 2.2.13.

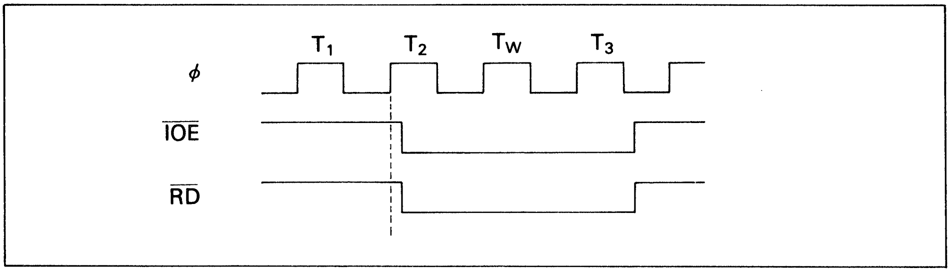


Figure 2.2.13 I/O Read Cycle When $\overline{\mathrm{IOC}}$=0

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 2.2.14.

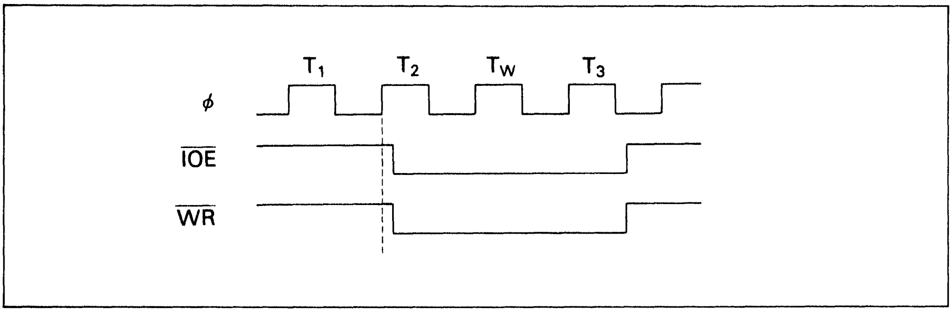


Figure 2.2.14 I/O Write Cycle When $\overline{IOC}$=0

RETI Instruction

It should be noted that the operation of RETI instruction of the Z Mask is different from that of the R1 Mask. In the Z Mask, the CPU reads the op-code, EDH and 4DH, twice as shown in figure 2.2.15.

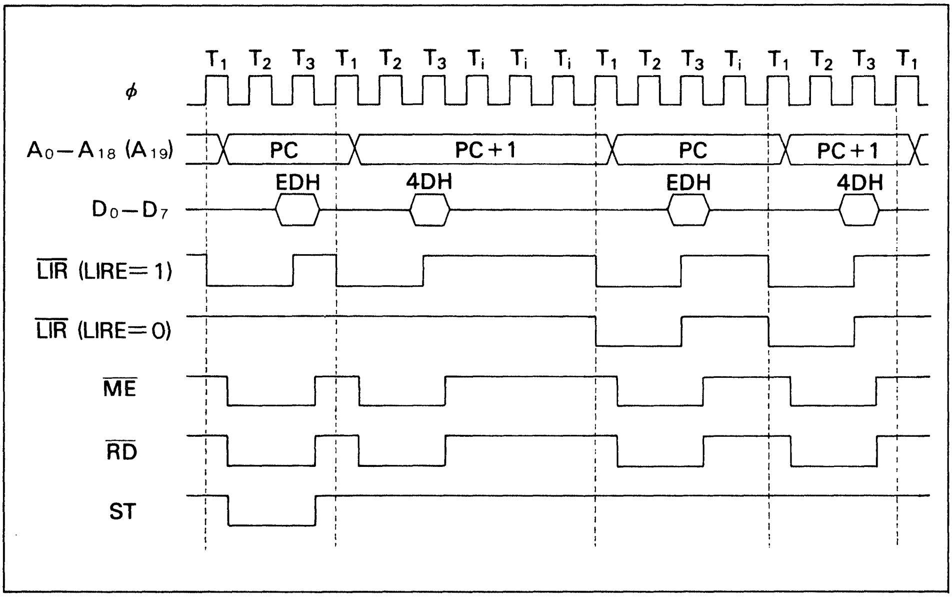


Figure 2.2.15 Operation of RETI Instruction

The number of states and machine cycles is shown in Table 2.2.1.

Table 2.2.1 The number of states and machine cycles

Version	Number of states	Number of machine cycles
R1	12	4
Z	22	10

Note 1: Interrupt request during RETI Instruction
The CPU can't be interrupted between the first and the second read of the op-code. The CPU can be interrupted after it completes the unstack operation.

How to Set the Bits in Operation Mode Control Register

Please set the bits in OMCR according to table 2.2.2.

Table 2.2.2 How to Set the Bits in Operation Mode Control Register

Usage of Z80's peripheral			Bits in OMCR		
Daisy chain	CTC	PIO	LIRE	$\overline{\text{LIRTE}}$	$\overline{\text{IOC}}$
YES	YES	YES	0	writing 0	0
		NO	0	*	0
	NO	YES	0	writing 0	0/1
		NO	0	*	0/1
NO	YES		1	*	0
	NO		1	*	0/1

* No operation to $\overline{\text{LIRTE}}$

2.3 WAIT State Generator

2.3.1 Wait state timing

To ease interfacing with slow memory and I/O devices, the HD64180 uses wait states (Tw) to extend bus cycle timing. A wait state(s) 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.

2.3.2 $\overline{\text{WAIT}}$ input

When the external $\overline{\text{WAIT}}$ input is asserted LOW, wait state (Tw) are 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.

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 2.3.1 shows $\overline{\text{WAIT}}$ timing.

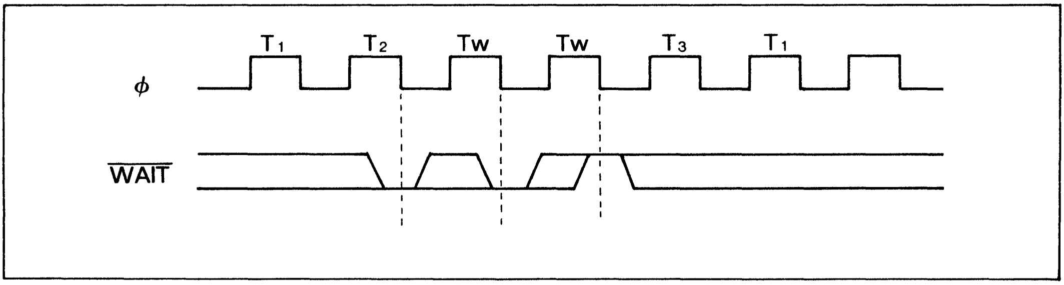


Figure 2.3.1 $\overline{\text{WAIT}}$ Timing

2.3.3 Programmable wait state insertion

In addition to the $\overline{\text{WAIT}}$ input, wait states (Tw) can also be programmably inserted using the HD64180 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.

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

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.

○ **Bit 7,6 : MWI1, MWI0 (Memory Wait Insertion)**

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.

MWI1	MWI0	The number of wait states
0	0	0
0	1	1
1	0	2
1	1	3

○ Bit 5, 4: IWI1, IWI0 (I/O Wait Insertion)

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.

		the number of wait states				
IWI1	IWI0	For external I/O registers accesses	For internal I/O registers accesses	For $\overline{\text{INT}_0}$ interrupt acknowledge cycles when $\overline{\text{LIR}}$ is LOW	For $\overline{\text{INT}_1}$, $\overline{\text{INT}_2}$ and internal interrupts acknowledge cycles (Note (2))	For $\overline{\text{NMI}}$ interrupt acknowledge cycles when $\overline{\text{LIR}}$ is LOW (Note (2))
0	0	1	0 (Note (1))	2	2	0
0	1	2		4		
1	0	3		5		
1	1	4		6		

Note:

(1) For HD64180 internal I/O register access (I/O addresses 0000H-003FH), IWI1 and IWI0 do not determine wait state (Tw) timing. For ASCI, CSI/O and PRT Data Register accesses, 0 to 4 wait states (Tw) will be generated. The number of wait states inserted during access to these registers is a function of internal synchronization requirements and CPU state.
All other on-chip I/O register accesses (i.e. MMU, DMAC, ASCI Control Registers, etc.) have 0 wait states inserted and thus require only three clock cycles.

(2) For interrupt acknowledge cycles in which $\overline{\text{LIR}}$ is HIGH, such as interrupt vector table read and PC stacking cycle, memory access timing applies.

2.3.4 $\overline{\text{WAIT}}$ input and RESET

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

2.4 HALT and Low Power Operation Modes

The HD64180 can operate in 4 different modes. HALT mode, IOSTOP 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.

2.4.1 HALT mode

HALT mode is entered by execution of the HALT instruction (op-code = 76H) and has the following characteristics.

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

Essentially, the HD64180 operates normally in HALT mode, except that instruction execution is stopped.

HALT mode can be exited in the following two ways.

RESET Exit from HALT mode

If the $\overline{\text{RESET}}$ input is asserted LOW for at least six clock cycles, HALT mode is exited and the normal RESET sequence (restart at address 00000H) is initiated.

Interrupt Exit from HALT mode

When an internal or external interrupt is generated, HALT mode is exited 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 HD64180 remains in HALT mode. However, $\overline{\text{NMI}}$ interrupt will initiate the normal $\overline{\text{NMI}}$ interrupt response sequence independent of the state of IEF$_1$.

HALT timing is shown in Fig. 2.4.1.

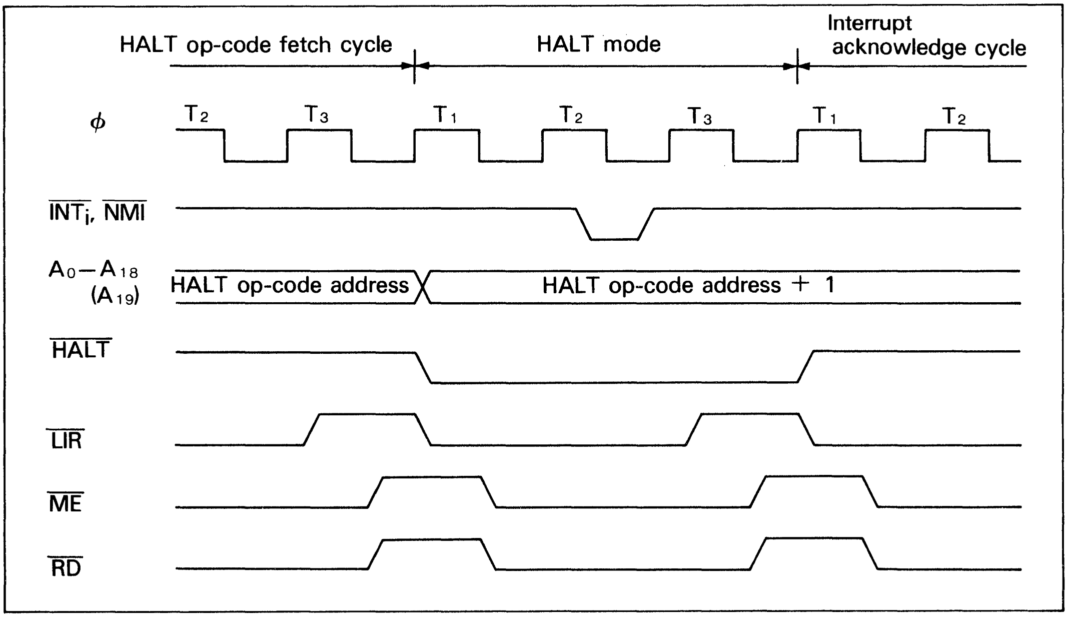


Figure 2.4.1 HALT Timing

2.4.2 SLEEP mode

SLEEP mode is entered by execution of the 2 byte SLP instruction. SLEEP mode has the following characteristics.

(1) The internal CPU clock stops, reducing power consumption.
(2) The internal crystal oscillator does not stop.
(3) Internal and external interrupt inputs can be received.
(4) DRAM refresh cycles stop.
(5) I/O operations using on-chip peripherals continue.
(6) The internal DMAC stop.
(7) $\overline{\text{BUSREQ}}$ can be received and acknowledged.
(8) Address outputs go HIGH and all other control signal output become inactive HIGH.
(9) Data Bus, 3-state.

SLEEP mode is exited in one of two ways as shown below.

RESET Exit from SLEEP mode

If the $\overline{\text{RESET}}$ input is held LOW for at least six clock cycles, the HD64180 will exit SLEEP mode and begin the normal RESET sequence with execution starting at address (logical and physical) 00000H.

Interrupt Exit from SLEEP Mode

The SLEEP mode is exited by detection of an external ($\overline{\text{NMI}}$, $\overline{\text{INT}_0}$, $\overline{\text{INT}_1}$, $\overline{\text{INT}_2}$) or internal (ASCI, CSI/O, PRT) interrupt.

In the case of $\overline{\text{NMI}}$, SLEEP Mode is exited and the CPU begins the normal $\overline{\text{NMI}}$ 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, occurrence of that interrupt is ignored and the CPU remains in the SLEEP state.

Assuming 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, SLEEP mode is exited and the appropriate normal interrupt response sequence is executed.

If interrupts are globally disabled ($IEF_1=0$) and an individually enabled interrupt occurs, SLEEP mode is exited 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.

Fig. 2.4.2 shows SLEEP timing.

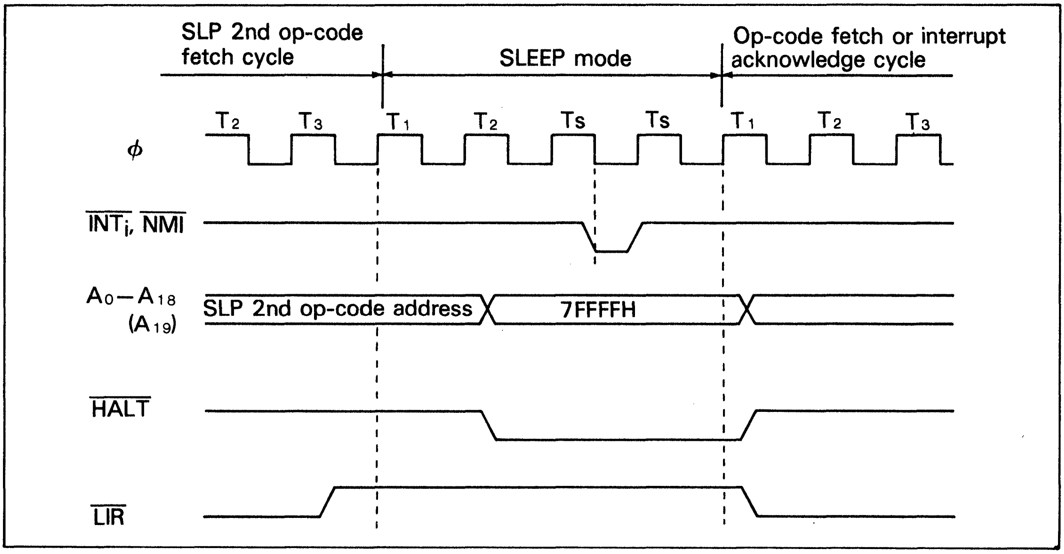


Figure 2.4.2 SLEEP Timing

In the case that interrupt requests occur during the CPU fetches SLP instruction, $\overline{\text{HALT}}$ output goes low level only by 1 state in SLEEP MODE of the R1 and Z Mask as shown in Fig. 2.4.3. In the case of R0 Mask, $\overline{\text{HALT}}$ output remains high level as shown in Fig. 2.4.4.

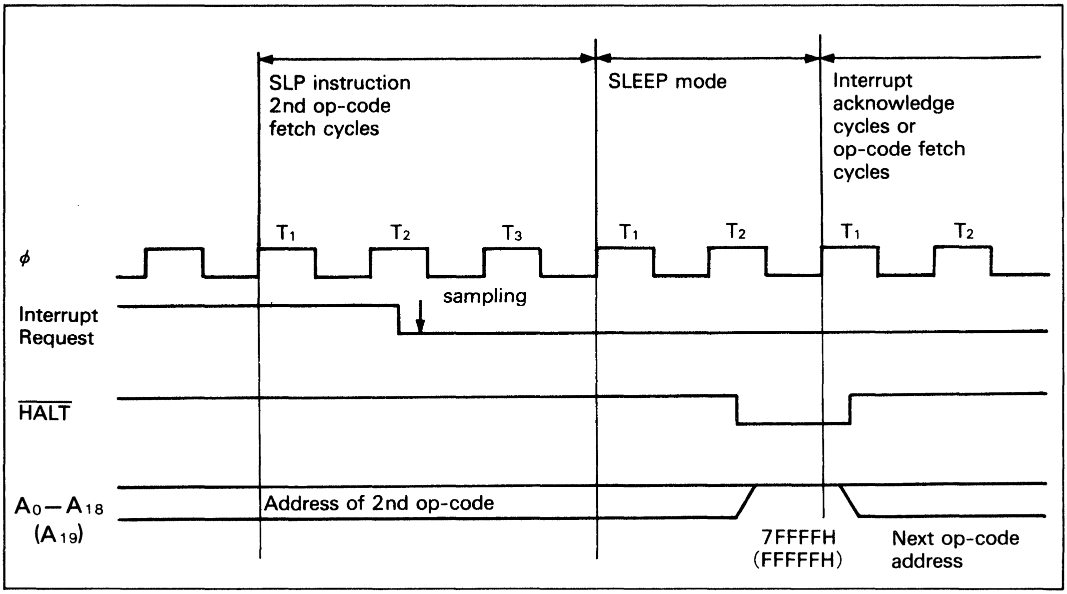


Figure 2.4.3 $\overline{\text{HALT}}$ Output of R1 and Z Mask

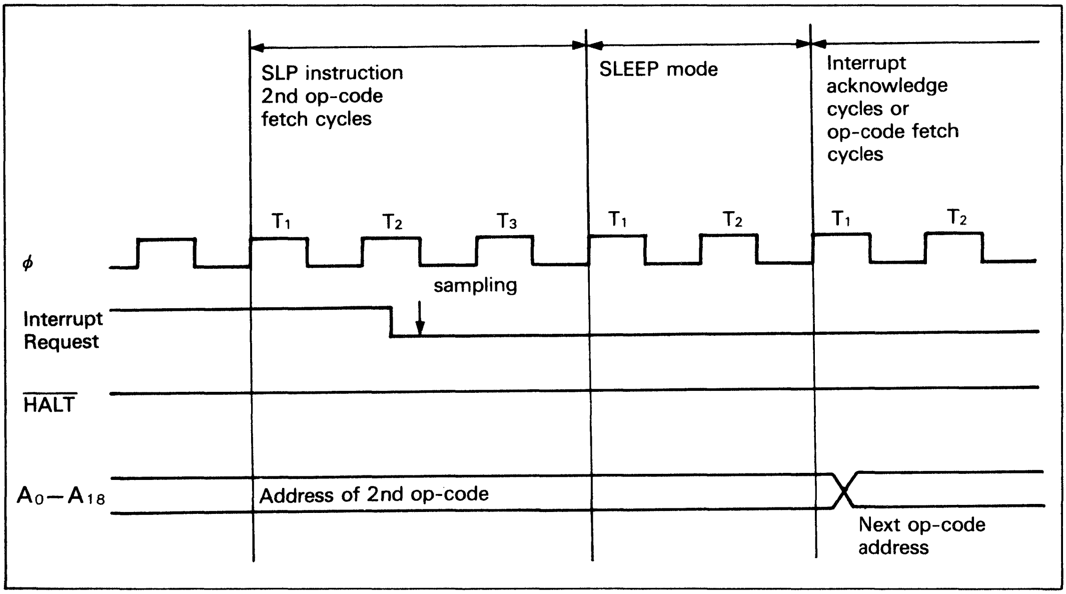


Figure 2.4.4 $\overline{\text{HALT}}$ Output of R0 Mask

2.4.3 IOSTOP mode

IOSTOP mode is entered 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 IOSTOP mode is by resetting the IOSTP bit in ICR to 0.

2.4.4 SYSTEM STOP mode

SYSTEM STOP mode is the combination of SLEEP and IOSTOP modes. SYSTEM STOP mode is entered by setting the IOSTP bit in ICR to 1 followed by execution of 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 IOSTOP) cannot generate a recovery interrupt.

2.5 Internal I/O Registers

The HD64180 internal I/O Registers occupy 64 I/O addresses (including reserved addresses). These registers access the internal I/O modules (ASCI, CSI/O, PRT) and control functions (DMAC, DRAM refresh, interrupts, wait state generator, MMU and I/O relocation).

To avoid address conflicts with external I/O, the HD64180 internal I/O addresses can be relocated on 64 bytes boundaries within the bottom 256 bytes of the 64k bytes I/O address space.

I/O Control Register (ICR)

ICR allows relocating of the internal I/O addresses. ICR also controls enabling/disabling of the IOSTOP mode.

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

bit	7	6	5	4	3	2	1	0
	IOA7	IOA6	IOSTP	–	–	–	–	–
	R/W	R/W	R/W					

○ IOA7,6: I/O Address Relocation (bits 7-6)

IOA7 and IOA6 relocate internal I/O as shown in Fig. 2.5.1. Note that the high-order 8 bits of 16-bit internal I/O addresses are always 0. IOA7 and IOA6 are cleared to 0 during RESET.

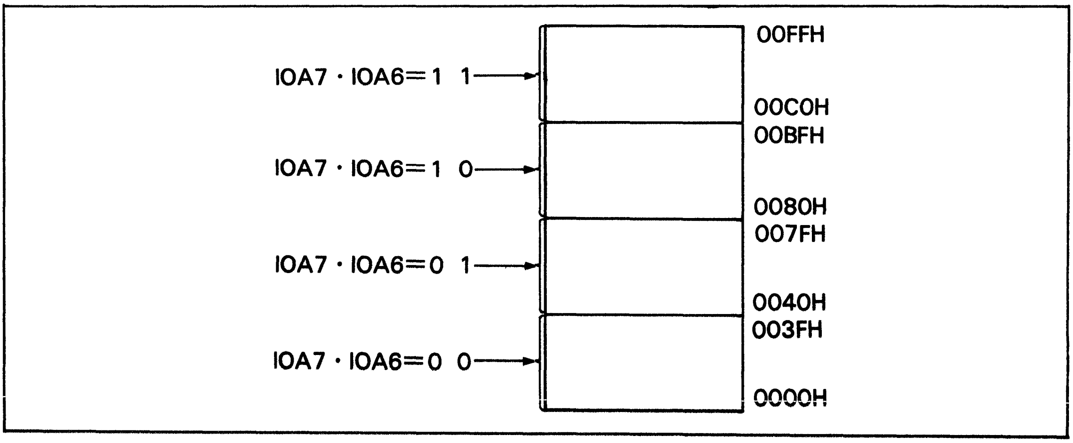


Figure 2.5.1 Internal I/O Address Relocation

○ IOSTP: IOSTOP Mode (bit 5)

IOSTOP 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.

Internal I/O Registers Address Map

The internal I/O register addresses are shown in Table 2.5.1. These addresses are relative to the 64 bytes boundary base address specified in ICR.

Table 2.5.1 Internal I/O Register Address Map (1)

	Register	Mnemonic	Address	
			Binary	Hexadecimal
ASCI	ASCI Control Register A Ch 0	CNTLA0	XX000000	00H
	ASCI Control Register A Ch 1	CNTLA1	XX000001	01H
	ASCI Control Register B Ch 0	CNTLB0	XX000010	02H
	ASCI Control Register B Ch 1	CNTLB1	XX000011	03H
	ASCI Status Register Ch 0	STAT0	XX000100	04H
	ASCI Status Register Ch 1	STAT1	XX000101	05H
	ASCI Transmit Data Register Ch 0	TDR0	XX000110	06H
	ASCI Transmit Data Register Ch 1	TDR1	XX000111	07H
	ASCI Receive Data Register Ch 0	RDR0	XX001000	08H
	ASCI Receive Data Register Ch 1	RDR1	XX001001	09H
CSI/O	CSI/O Control Register	CNTR	XX001010	0AH
	CSI/O Transmit/Receive Data Register	TRDR	XX001011	0BH
Timer	Timer Data Register Ch 0L	TMDR0L	XX001100	0CH
	Timer Data Register Ch 0H	TMDR0H	XX001101	0DH
	Reload Register Ch 0L	RLDR0L	XX001110	0EH
	Reload Register Ch 0H	RLDR0H	XX001111	0FH
	Timer Control Register	TCR	XX010000	10H
	Reserved		XX010001	11H
			≀	≀
			XX010011	13H
	Timer Data Register Ch 1L	TMDR1L	XX010100	14H
	Timer Data Register Ch 1H	TMDR1H	XX010101	15H
	Reload Register Ch 1L	RLDR1L	XX010110	16H
	Reload Register Ch 1H	RLDR1H	XX010111	17H
Others	Free Running Counter	FRC	XX011000	18H
	Reserved		XX011001	19H
			≀	≀
			XX011111	1FH

Table 2.5.1 Internal I/O Register Address Map (2)

	Register	Mnemonic	Address	
			Binary	Hexadecimal
DMA	DMA Source Address Register Ch 0L	SAR0L	XX100000	20H
	DMA Source Address Register Ch 0H	SAR0H	XX100001	21H
	DMA Source Address Register Ch 0B	SAR0B	XX100010	22H
	DMA Destination Address Register Ch 0L	DAR0L	XX100011	23H
	DMA Destination Address Register Ch 0H	DAR0H	XX100100	24H
	DMA Destination Address Register Ch 0B	DAR0B	XX100101	25H
	DMA Byte Count Register Ch 0L	BCR0L	XX100110	26H
	DMA Byte Count Register Ch 0H	BCR0H	XX100111	27H
	DMA Memory Address Register Ch 1L	MAR1L	XX101000	28H
	DMA Memory Address Register Ch 1H	MAR1H	XX101001	29H
	DMA Memory Address Register Ch 1B	MAR1B	XX101010	2AH
	DMA I/O Address Register Ch 1L	IAR1L	XX101011	2BH
	DMA I/O Address Register Ch 1H	IAR1H	XX101100	2CH
	Reserved		XX101101	2DH
	DMA Byte Count Register Ch 1L	BCR1L	XX101110	2EH
	DMA Byte Count Register Ch 1H	BCR1H	XX101111	2FH
	DMA Status Register	DSTAT	XX110000	30H
	DMA Mode Register	DMODE	XX110001	31H
	DMA/WAIT Control Register	DCNTL	XX110010	32H
INT	IL Register (Interrupt Vector Low Register)	IL	XX110011	33H
	INT/TRAP Control Register	ITC	XX110100	34H
	Reserved		XX110101	35H

Table 2.5.1 Internal I/O Register Address Map (3)

	Register	Mnemonic	Address	
			Binary	Hexadecimal
Refresh	Refresh Control Register	RCR	XX110110	36H
	Reserved		XX110111	37H
MMU	MMU Common Base Register	CBR	XX111000	38H
	MMU Bank Base Register	BBR	XX111001	39H
	MMU Common/Bank Area Register	CBAR	XX111010	3AH
I/O	Reserved		XX111011 ≀ XX111101	3BH ≀ 3DH
	Operation Mode Control Register	OMCR	XX111110	3EH
	I/O Control Register	ICR	XX111111	3FH

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 3.1 Instruction Set).

Note that when writing to an internal I/O register, 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 HD64180.

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

2.6 Memory Management Unit (MMU)

The HD64180 contains an on-chip MMU which performs the translation of the CPU 64k bytes (16-bit addresses- 0000H to FFFFH) logical memory address space into a 512k bytes (19-bit addresses- 00000H to 7FFFFH) or 1M bytes (20-bit addresses- 00000H to FFFFFH) physical memory address space. Address translation occurs internally in parallel with other CPU operation.

2.6.1 Logical address spaces

The 64k bytes 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 Fig. 2.6.1 a variety of logical memory configurations are possible. The boundaries between the Common and Bank Areas can be programmed with 4k bytes resolution.

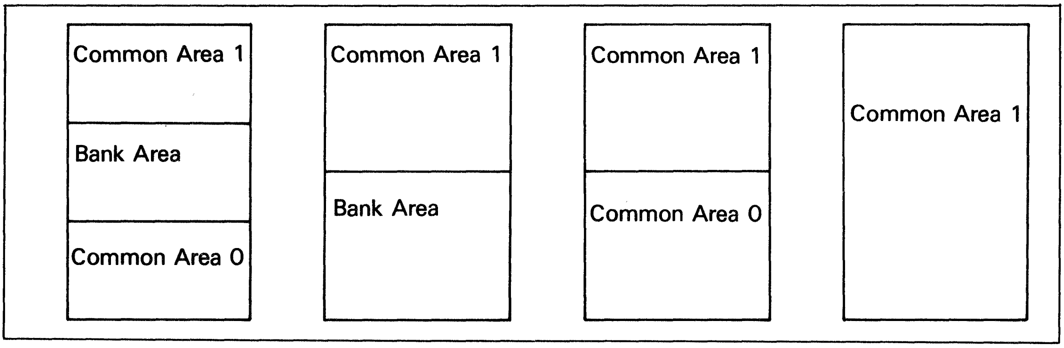


Figure 2.6.1 Logical Address Mapping Examples

2.6.2 Logical to physical address translation

Fig. 2.6.2 shows an example in which the three logical address space portions are mapped into a 512k (1M) bytes 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 4k bytes physical address boundaries). Common Area 0 (if it exists) is always based at physical address 00000H.

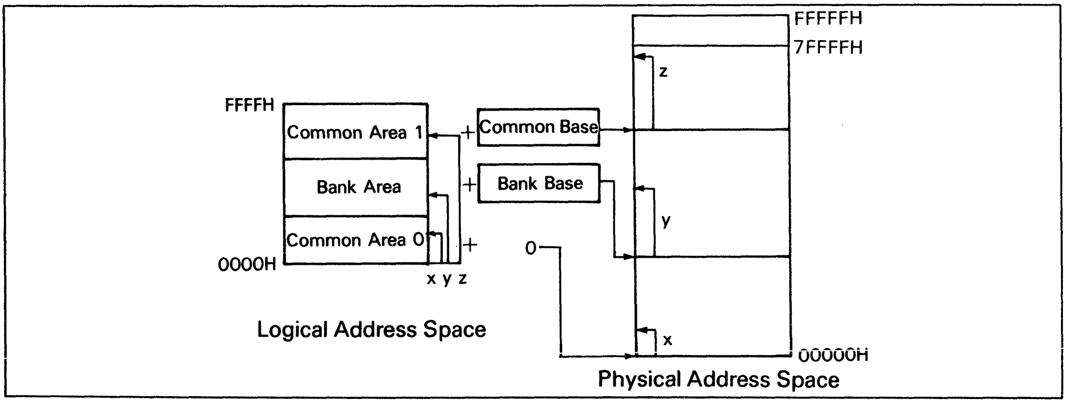


Figure 2.6.2 Logical to Physical Memory Mapping Example

2.6.3 MMU block diagram

The MMU block diagram is shown in Fig. 2.6.3. The MMU translates internal 16-bit logical addresses to external 19-bit physical addresses.

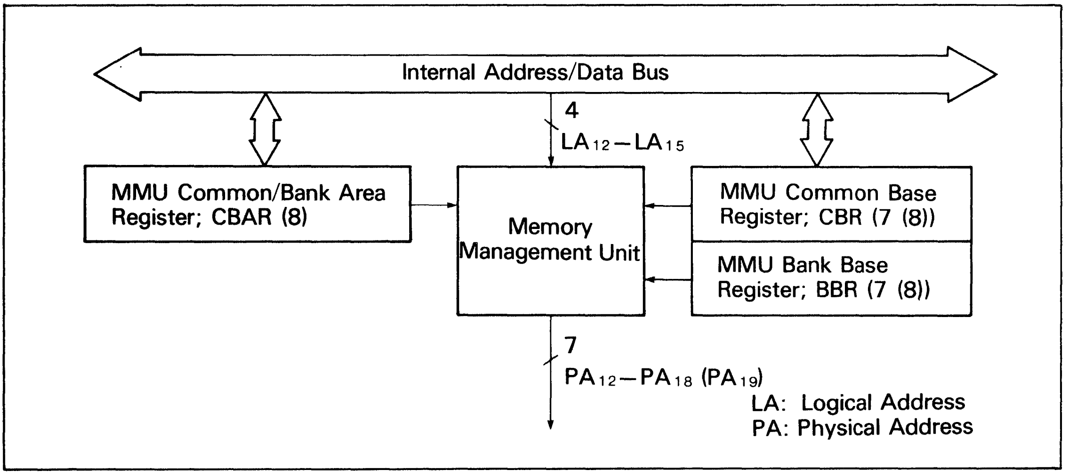

Internal Address/Data Bus
4
LA12—LA15
MMU Common/Bank Area Register; CBAR (8)
Memory Management Unit
MMU Common Base Register; CBR (7 (8))
MMU Bank Base Register; BBR (7 (8))
7
PA12—PA18 (PA19)
LA: Logical Address
PA: Physical Address

Figure 2.6.3 MMU Block Diagram

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

(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.

(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 three high order bits (A_{16}-A_{18} (A_{19})) of the physical address are always 0 during I/O cycles.

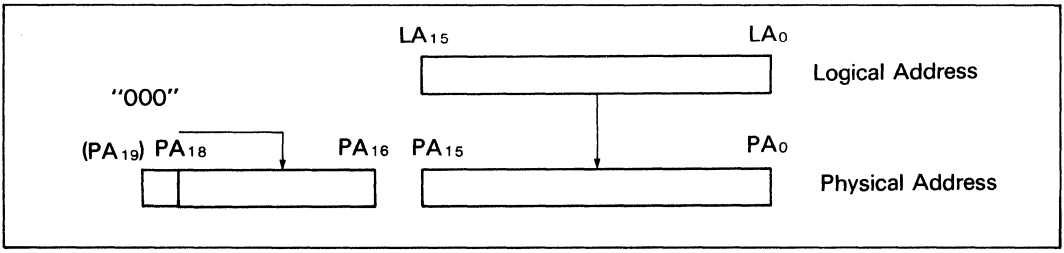


Figure 2.6.4 I/O Address Translation

(3) DMA Cycles

When the HD64180 on-chip DMAC is using the external bus, the MMU is physically bypassed. The 19-bit source and destination registers in the DMAC are directly output on the physical address bus (A_0-A_{18} (A_{19})).

2.6.4 MMU registers

Three MMU registers are used to 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 is used to define the logical memory organization, while CBR and BBR are used to relocate logical areas within the 512k (1M) bytes physical address space. The resolution for both setting boundaries within the logical space and relocation within the physical space is 4k bytes.

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. Fig. 2.6.5 and Fig. 2.6.6 shows example of logical memory organizations associated with different values of CA and BA.

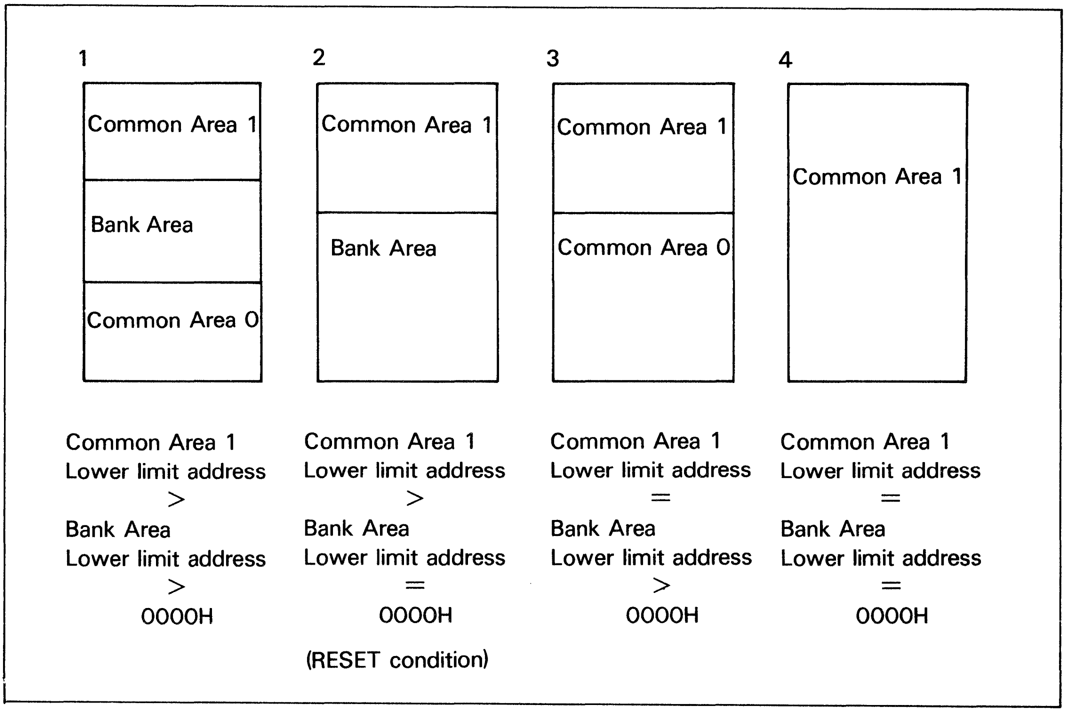

1
Common Area 1
Bank Area
Common Area 0
Common Area 1 Lower limit address
>
Bank Area Lower limit address
>
0000H
2
Common Area 1
Bank Area
Common Area 1 Lower limit address
>
Bank Area Lower limit address
=
0000H
(RESET condition)
3
Common Area 1
Common Area 0
Common Area 1 Lower limit address
=
Bank Area Lower limit address
>
0000H
4
Common Area 1
Common Area 1 Lower limit address
=
Bank Area Lower limit address
=
0000H

Figure 2.6.5 Logical Memory Organization

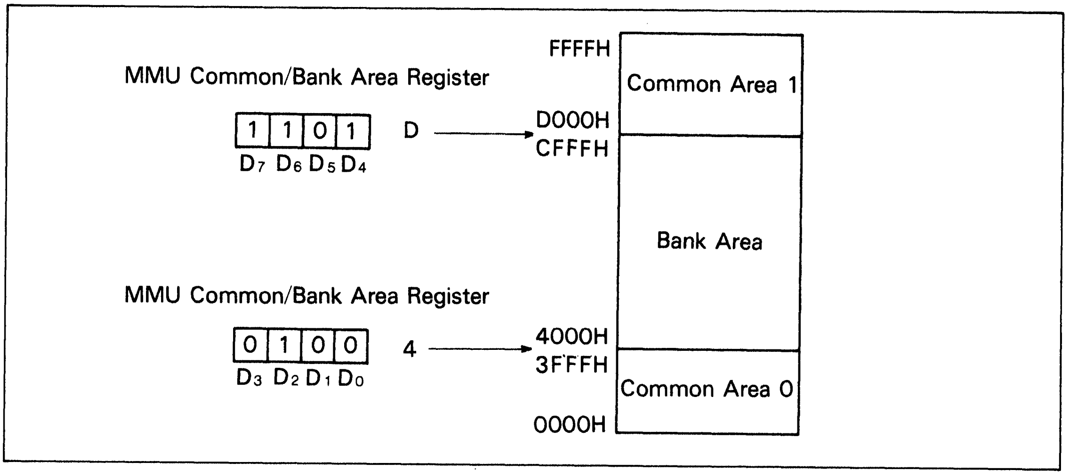


Figure 2.6.6 Logical Space Configuration (Example)

MMU REGISTER DESCRIPTION

MMU Common/Bank Area Register (CBAR)

CBAR specifies boundaries within the HD64180 64k bytes 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

○ CA3-CA0: CA (bits 7-4)

CA specifies the start (low) address (on 4k bytes boundaries) for the 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: BA (bits 3-0)

BA specifies the start (low) address (on 4k bytes boundaries) for the Bank Area. This also determines the last address of the Common Area 0. All bits of BA are reset to 0 during RESET.

MMU Common Base Register (CBR)

CBR specifies the base address (on 4k bytes boundaries) used to generate a 19-bit or 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

MMU Bank Base Register (BBR)

BBR specifies the base address (on 4k bytes boundaries) used to generate a 19-bit or 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

2.6.5 Physical address translation

Fig. 2.6.7 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 7-bit base address is added to the high-order 4 bits of the logical address, yielding a 19-bit or 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.

* In case of R1 and Z Mask
CBR and BBR are expanded from 7 bits to 8 bits in the package version of CP-68 and FP-80.
There is no change in DP-64S.

2.6.6 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 64k bytes address space corresponds directly with the first 64k bytes (0000H to FFFFH) of the 512k bytes (00000H to 7FFFFH) physical address space. Thus, after RESET, the HD64180 will begin execution at logical and physical address 0.

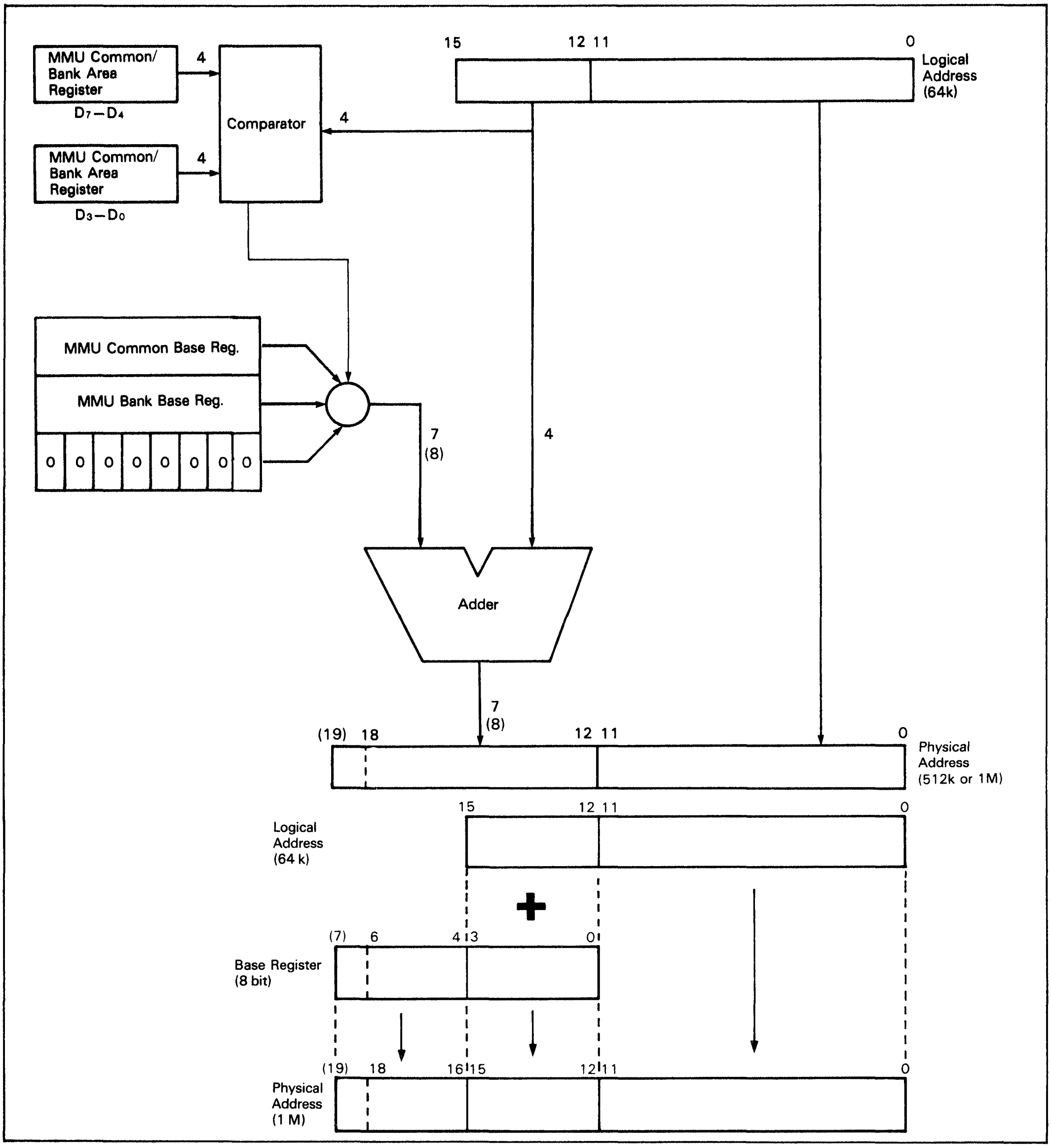


Figure 2.6.7 Physical Address Generation

2.6.7 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 op-code fetch from the newly translated address. One simple technique is to localize all MMU programming routines in a Common Area that is always enabled.

2.7 Interrupts

The HD64180 CPU has twelve interrupt sources, four external and eight internal, with fixed priority.

Higher	(1)	TRAP (Undefined Op-code Trap)	Internal Interrupt
Priority	(2)	$\overline{NMI}$ (Non Maskable Interrupt)	External Interrupt
↑	(3)	$\overline{INT_0}$ (Maskable Interrupt Level 0)	
	(4)	$\overline{INT_1}$ (Maskable Interrupt Level 1)	
	(5)	$\overline{INT_2}$ (Maskable Interrupt Level 2)	
	(6)	Timer 0	Internal Interrupt
	(7)	Timer 1	
	(8)	DMA channel 0	
	(9)	DMA channel 1	
↓	(10)	Clocked Serial I/O Port	
Lower	(11)	Asynchronous SCI channel 0	
Priority	(12)	Asynchronous SCI channel 1	

Figure 2.7.1 Interrupt Sources

This section explains the CPU registers associated with interrupt processing, the TRAP interrupt, interrupt response modes and the external interrupts. The detailed discussion of internal interrupt generation (except TRAP) is presented in the appropriate hardware section (i.e. PRT, DMAC, ASCI and CSI/O).

2.7.1 Interrupt control registers and flags

The HD64180 contains three registers and two flags which are associated with interrupt processing.

Function	Name	Access Method
(1) Interrupt Vector High	I	LD A, I and LD I, A instructions
(2) Interrupt Vector Low	IL	I/O instruction (addr=33H)
(3) Interrupt/Trap Control	ITC	I/O instruction (addr=34H)
(4) Interrupt Enable Flag 1,2	IEF_1, IEF_2	EI and DI LD A, I LD A, R instructions

Interrupt Vector Register (I)

Mode 2 for $\overline{INT_0}$ external interrupt, $\overline{INT_1}$ 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 64k bytes 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.

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 Code				

This register 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 bytes boundaries.

IL is initialized to 00H during RESET.

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

ITC 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 op-code is fetched. TRAP can be reset under program control by writing it with 0, however it cannot be written with 1 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 allow determination of the starting address of the undefined instruction. This is necessary since the TRAP may occur on either the second or third byte of the op-code. UFO allows the stacked PC value (stacked in response to TRAP) to be correctly adjusted. If UFO = 0, the first op-code should be interpreted as the stacked PC−1. If UFO = 1, the first op-code address is 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 $\overline{\text{INT}_2}$, $\overline{\text{INT}_1}$ and $\overline{\text{INT}_0}$ respectively. If reset to 0, the interrupt is masked. During RESET, ITE0 is initialized to 1 while ITE1 and ITE2 are initialized to 0.

Interrupt Enable Flag 1,2 (IEF$_1$, IEF$_2$)

IEF$_1$ controls the overall enabling and disabling of all internal and external maskable interrupts (i.e. all interrupts except $\overline{\text{NMI}}$ 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 $\overline{\text{NMI}}$. During $\overline{\text{NMI}}$, 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 $\overline{\text{NMI}}$ interrupt service routine, execution of the RETN (Return from Nonmaskable Interrupt) will automatically restore the interrupt receiving state (by copying IEF$_2$ to IEF$_1$) prior to the occurrence of $\overline{\text{NMI}}$.

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 2.7.1 shows the state of IEF$_1$ and IEF$_2$.

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

In case of R0 Mask

$\overline{\text{NMI}}$ requests are received during execution of LD A, I or LD A, R and $\overline{\text{NMI}}$ acknowledge cycle begins just after completion of the instruction. At this time, the value of IEF$_2$ transfered to P/V Flag is the value of IEF$_1$ which changes from "1" to "0" during the $\overline{\text{NMI}}$ acknowledge cycle.

In case of R1 and Z Mask

All of interrupt requests including $\overline{\text{NMI}}$ can't be sampled during execution of LD A, I or LD A, R instruction 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 2.7.1 State of IEF$_1$ and IEF$_2$

CPU Operation	IEF$_1$	IEF$_2$	REMARKS
RESET	0	0	Inhibits the interrupt except NMI and TRAP.
$\overline{\text{NMI}}$	0	IEF$_1$	Copies the contents of IEF$_1$ to IEF$_2$.
RETN	IEF$_2$	not affected	Returns from the NMI service routine.
Interrupt except $\overline{\text{NMI}}$ and TRAP	0	0	Inhibits the interrupt except $\overline{\text{NMI}}$ 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. In the R1 and Z Mask, interrupts are not sampled.
LD A, R	not affected	not affected	

2.7.2 TRAP interrupt

The HD64180 generates a non-maskable (not affected by the state of IEF$_1$) TRAP interrupt when an undefined op-code fetch occurs. This feature can be used to increase software reliability, implement an 'extended' instruction set, or both. TRAP may occur during op-code fetch cycles and also if an undefined op-code is fetched during the interrupt acknowledge cycle for $\overline{\text{INT}_0}$ when Mode 0 is used.

When a TRAP interrupt occurs the HD64180 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 op-code, is saved on the stack.

(3) The HD64180 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 op-code 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. Fig. 2.7.2 shows TRAP Timing.

Note that Bus Release cycle, Refresh cycle, DMA cycle and WAIT cycle can't be inserted just after T_{TP} state which is inserted for TRAP interrupt sequence.

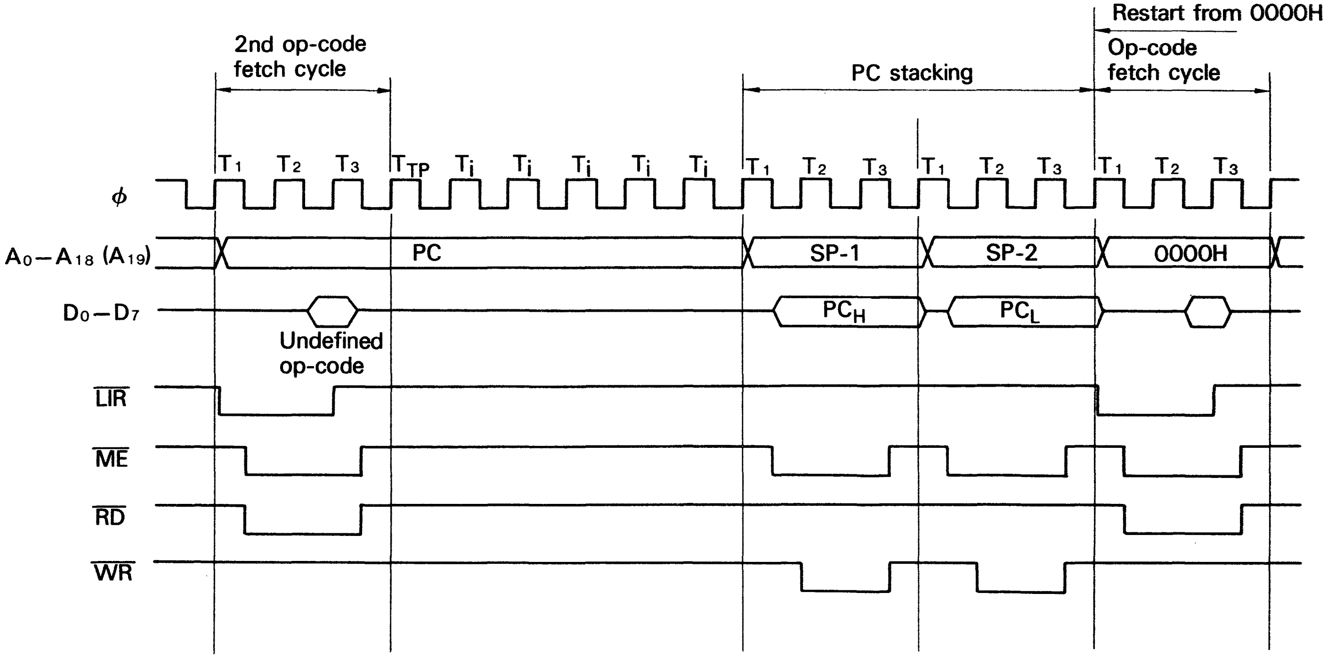


Figure 2.7.2(a) TRAP Timing — 2nd Op-code Undefined

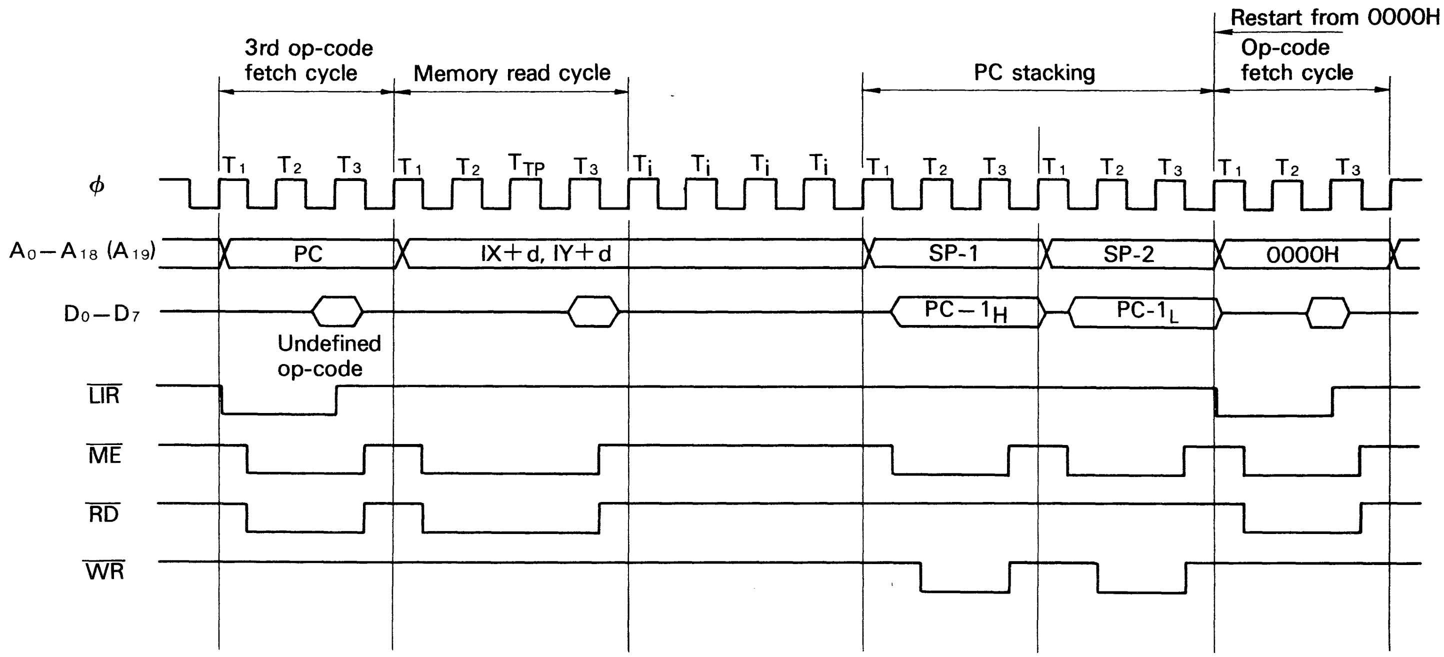


Figure 2.7.2(b) TRAP Timing – 3rd Op-code Undefined

2.7.3 External interrupts

The HD64180 has four external hardware interrupt inputs.

(1) $\overline{\text{NMI}}$ – Non-maskable Interrupt
(2) $\overline{\text{INT}_0}$ – Maskable Interrupt Level 0
(3) $\overline{\text{INT}_1}$ – Maskable Interrupt Level 1
(4) $\overline{\text{INT}_2}$ – Maskable Interrupt Level 2

$\overline{\text{NMI}}$, $\overline{\text{INT}_1}$ and $\overline{\text{INT}_2}$ have fixed interrupt response modes. $\overline{\text{INT}_0}$ has three different software programmable interrupt response modes – Mode 0, Mode 1 and Mode 2.

2.7.4 $\overline{\text{NMI}}$ – Non-Maskable Interrupt

The $\overline{\text{NMI}}$ interrupt input is edge sensitive and cannot be masked by software. When $\overline{\text{NMI}}$ is detected, the HD64180 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 $\overline{\text{NMI}}$.

(4) IEF_1 is cleared to 0. This disables all external and internal maskable interrupts (i.e. all interrupts except $\overline{\text{NMI}}$ and TRAP).

(5) Execution commences at logical address 0066H.

The last instruction of an $\overline{\text{NMI}}$ 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 $\overline{\text{NMI}}$.

Note that $\overline{\text{NMI}}$, since it can be accepted during HD64180 on-chip DMAC operation, can be used to externally interrupt DMA transfer. The $\overline{\text{NMI}}$ service routine can reactivate or abort the DMAC operation as required by the application.

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.

Fig. 2.7.3 shows the use of $\overline{\text{NMI}}$ and RETN while Fig. 2.7.4 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

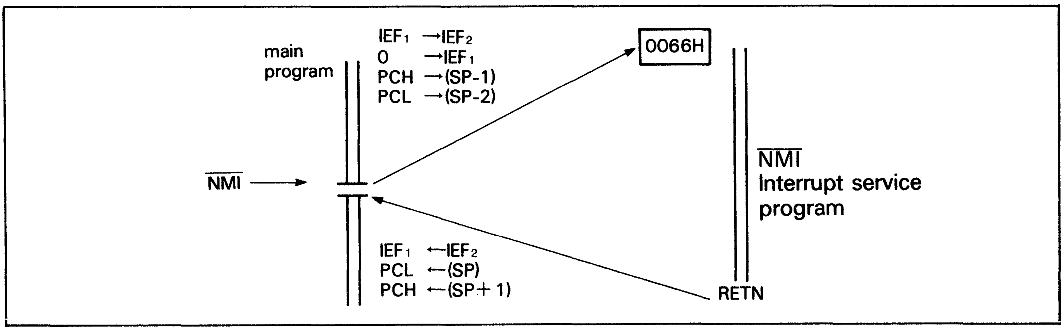


Figure 2.7.3 $\overline{\text{NMI}}$ Sequence

of 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 $\overline{\text{NMI}}$ acknowledge cycle begins at the end of the current machine cycle.

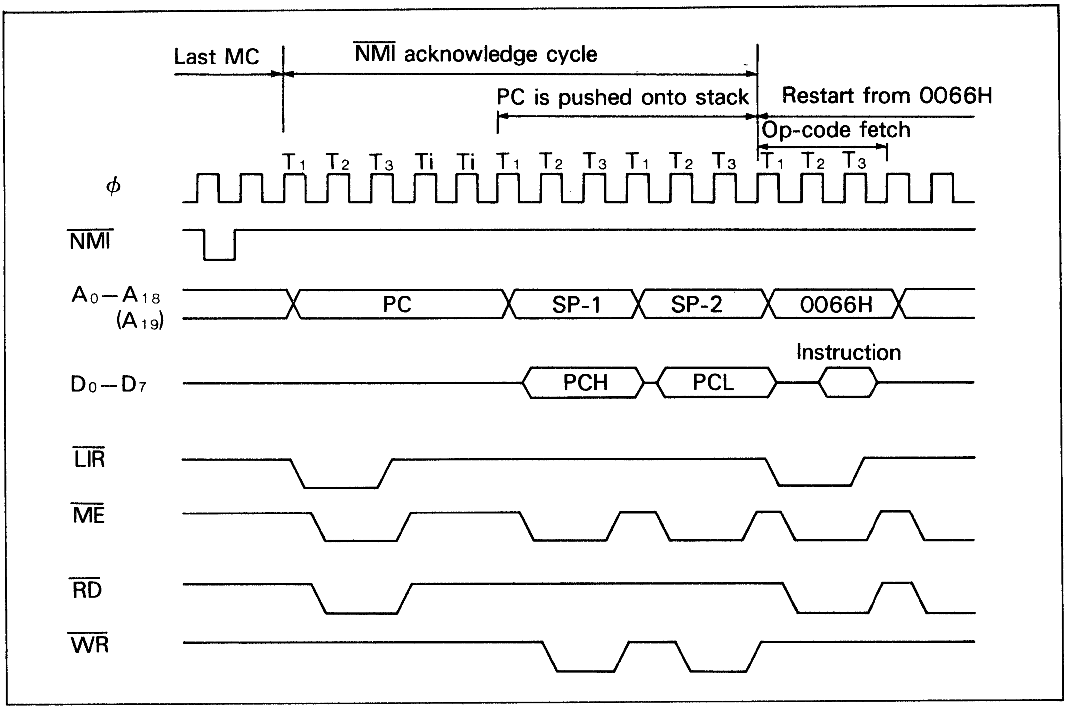


Figure 2.7.4 $\overline{\text{NMI}}$ Timing

2.7.5 $\overline{\text{INT}_0}$ – Maskable Interrupt Level 0

The next highest priority external interrupt after $\overline{\text{NMI}}$ is $\overline{\text{INT}_0}$. $\overline{\text{INT}_0}$ is sampled at the falling edge of the clock state prior to T_3 or Ti in the last machine cycle. If $\overline{\text{INT}_0}$ is asserted LOW at the falling edge of the clock state prior to T_3 or Ti in the last machine cycle, $\overline{\text{INT}_0}$ 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.

(1) IEF_1 is 0, so $\overline{\text{INT}_0}$ is masked.
(2) ITE0 is 1, so $\overline{\text{INT}_0}$ is enabled by execution of the EI (Enable Interrupts) instruction.

The $\overline{\text{INT}_0}$ 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. During RESET, the HD64180 is initialized to use Mode 0 for $\overline{\text{INT}_0}$.

The three interrupt response modes for $\overline{\text{INT}_0}$ 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.

○ $\overline{\text{INT}_0}$ 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 TRAP interrupt will occur if an invalid instruction is fetched during Mode 0 interrupt acknowledge.

Fig. 2.7.5 shows $\overline{\text{INT}_0}$ Mode 0 Timing.

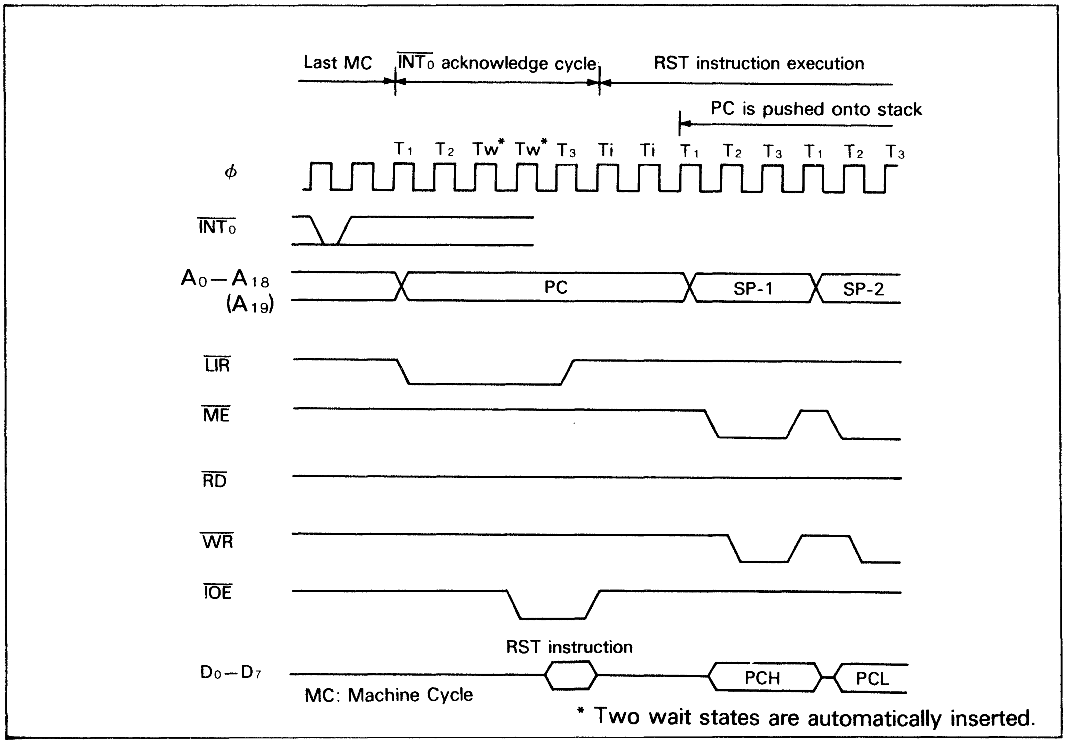


Figure 2.7.5 $\overline{\text{INT}_0}$ Mode 0 Timing (RST Instruction on the Data Bus)

○ $\overline{\text{INT}_0}$ Mode 1

When $\overline{\text{INT}_0}$ 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. Fig. 2.7.6 shows the use of $\overline{\text{INT}_0}$ (Mode 1) and RETI.

Fig. 2.7.7 shows $\overline{\text{INT}_0}$ Mode 1 timing.

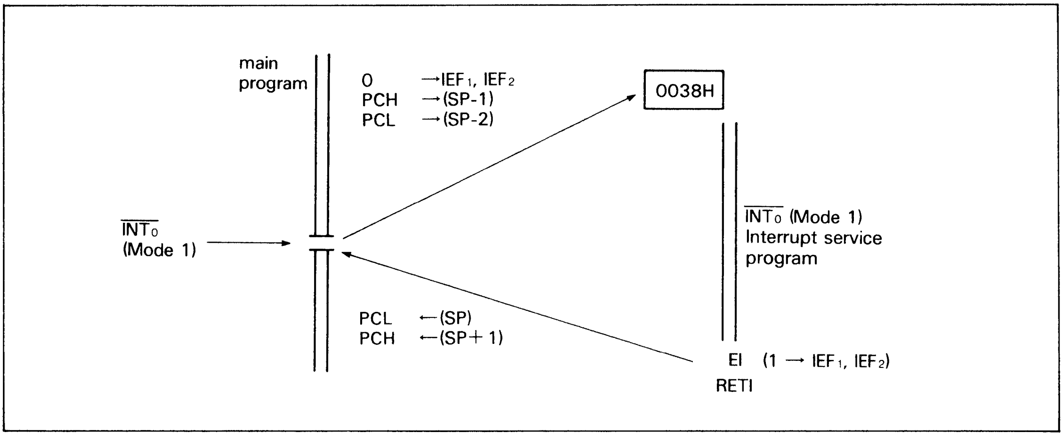
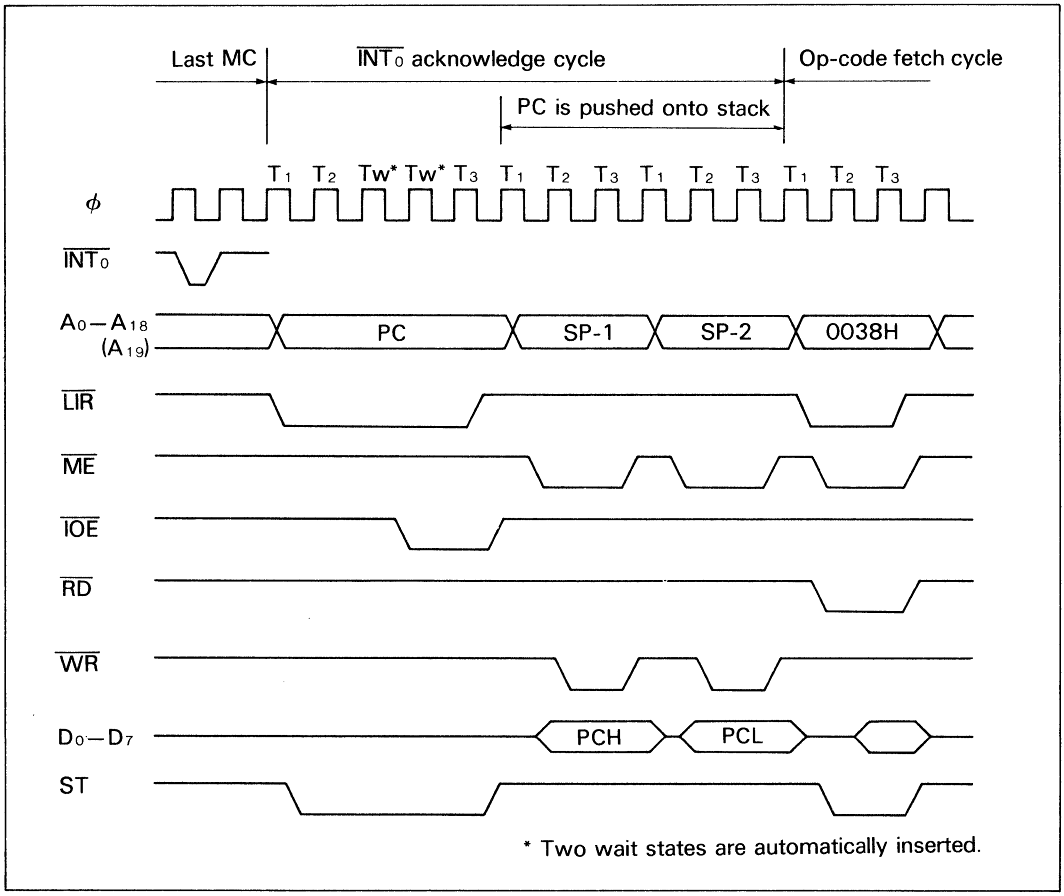


Figure 2.7.6 $\overline{\text{INT}_0}$ Mode 1 Interrupt Sequence



* Two wait states are automatically inserted.

Figure 2.7.7 $\overline{\text{INT}_0}$ Mode 1 Timing

○ $\overline{\text{INT}}_0$ Mode 2

This method 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 64k bytes logical address space as programmed in the 8-bit Interrupt Vector Register (I). Fig. 2.7.8 shows the $\overline{\text{INT}}_0$ Mode 2 Vector acquisition.

During $\overline{\text{INT}}_0$ 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 $\overline{\text{LIR}}$ and $\overline{\text{IOE}}$ 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 $\overline{\text{INT}}_0$ interrupt. Fig. 2.7.9 shows $\overline{\text{INT}}_0$ interrupt Mode 2 Timing.

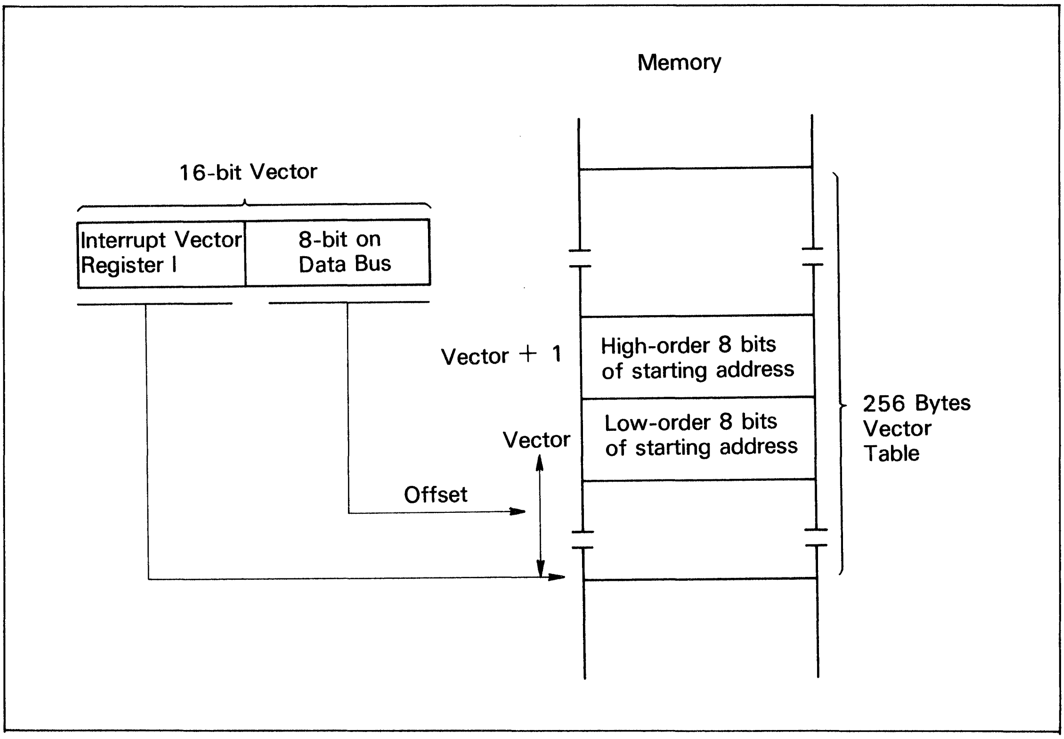

Memory
16-bit Vector
Interrupt Vector Register I
8-bit on Data Bus
Vector + 1
High-order 8 bits of starting address
Vector
Low-order 8 bits of starting address
Offset
256 Bytes Vector Table

Figure 2.7.8 $\overline{\text{INT}}_0$ Mode 2 Vector Acquisition

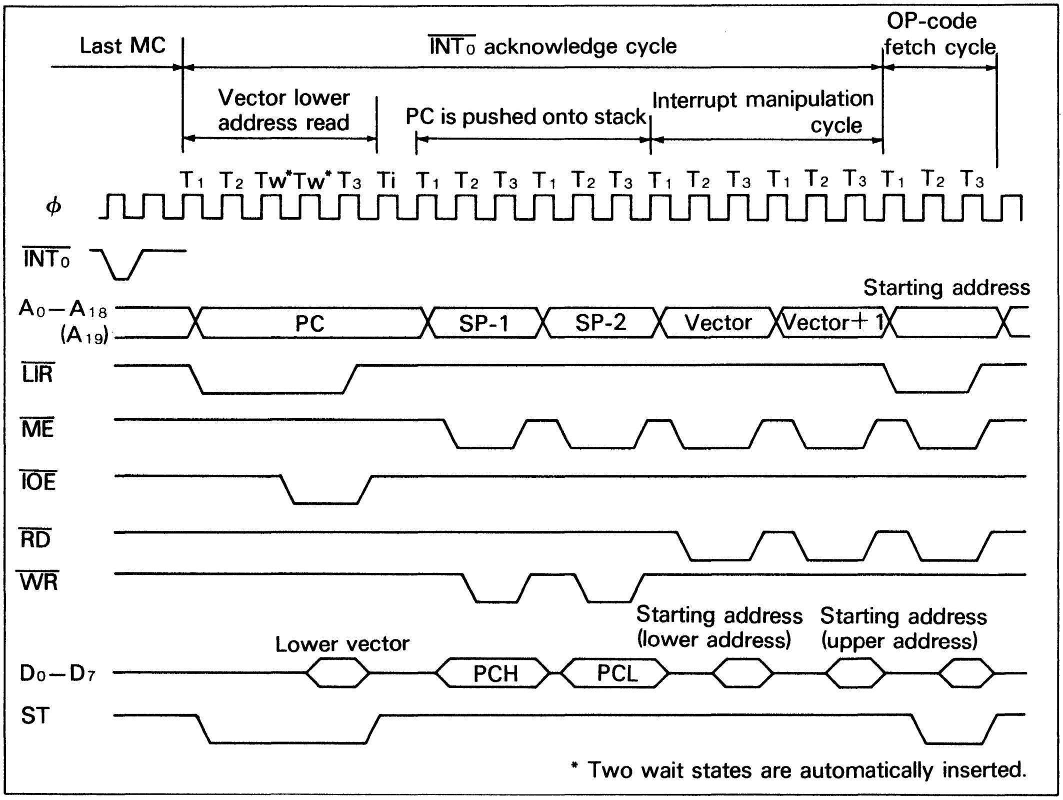

Last MC
$\overline{\mathrm{INT}_0}$ acknowledge cycle
OP-code fetch cycle
Vector lower address read
PC is pushed onto stack
Interrupt manipulation cycle
T1 T2 Tw*Tw* T3 Ti T1 T2 T3 T1 T2 T3 T1 T2 T3 T1 T2 T3 T1 T2 T3
φ
$\overline{\mathrm{INT}_0}$
Starting address
A0—A18 (A19)
PC
SP-1
SP-2
Vector
Vector+1
$\overline{\mathrm{LIR}}$
$\overline{\mathrm{ME}}$
$\overline{\mathrm{IOE}}$
$\overline{\mathrm{RD}}$
$\overline{\mathrm{WR}}$
Lower vector
Starting address (lower address)
Starting address (upper address)
D0—D7
PCH
PCL
ST
* Two wait states are automatically inserted.

Figure 2.7.9 $\overline{\mathrm{INT}_0}$ Mode 2 Timing

2.7.6 $\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$

The operation of external interrupts $\overline{\mathrm{INT}_1}$ and $\overline{\mathrm{INT}_2}$ is a vector mode similar to $\overline{\mathrm{INT}_0}$ Mode 2. The difference is that $\overline{\mathrm{INT}_1}$ and $\overline{\mathrm{INT}_2}$ 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 Fig. 2.7.10 the low-order byte of vector table address is comprised 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 ($\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$ and internal) source.

$\overline{\mathrm{INT}_1}$ and $\overline{\mathrm{INT}_2}$ 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.

2.7.7 Internal interrupts

Internal interrupts (except TRAP) use the same vectored response mode as $\overline{\mathrm{INT}_1}$ and $\overline{\mathrm{INT}_2}$ (Fig. 2.7.10). 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 $\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$

and internal interrupt are summarized in Table 2.7.2.

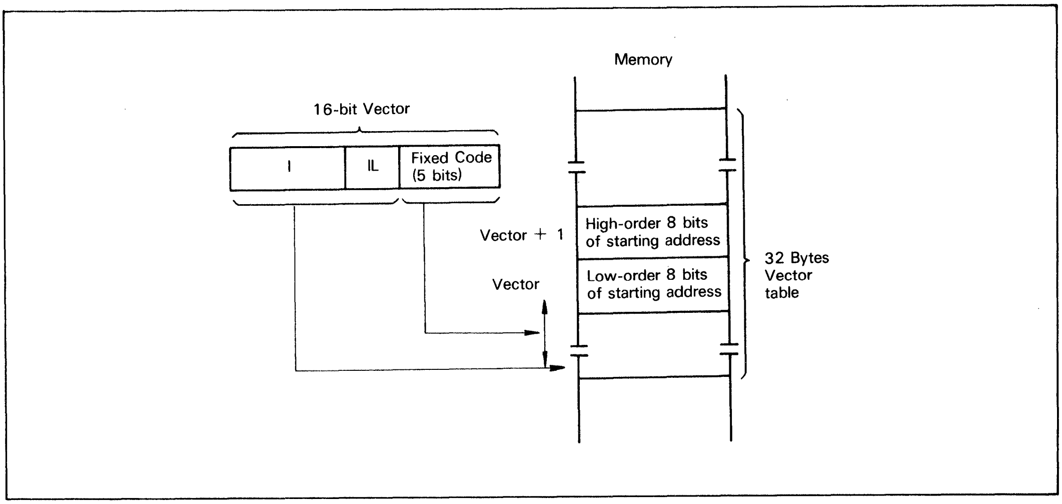


Figure 2.7.10 $\overline{\text{INT}}_1$, $\overline{\text{INT}}_2$ and Internal Interrupts Vector Acquisition

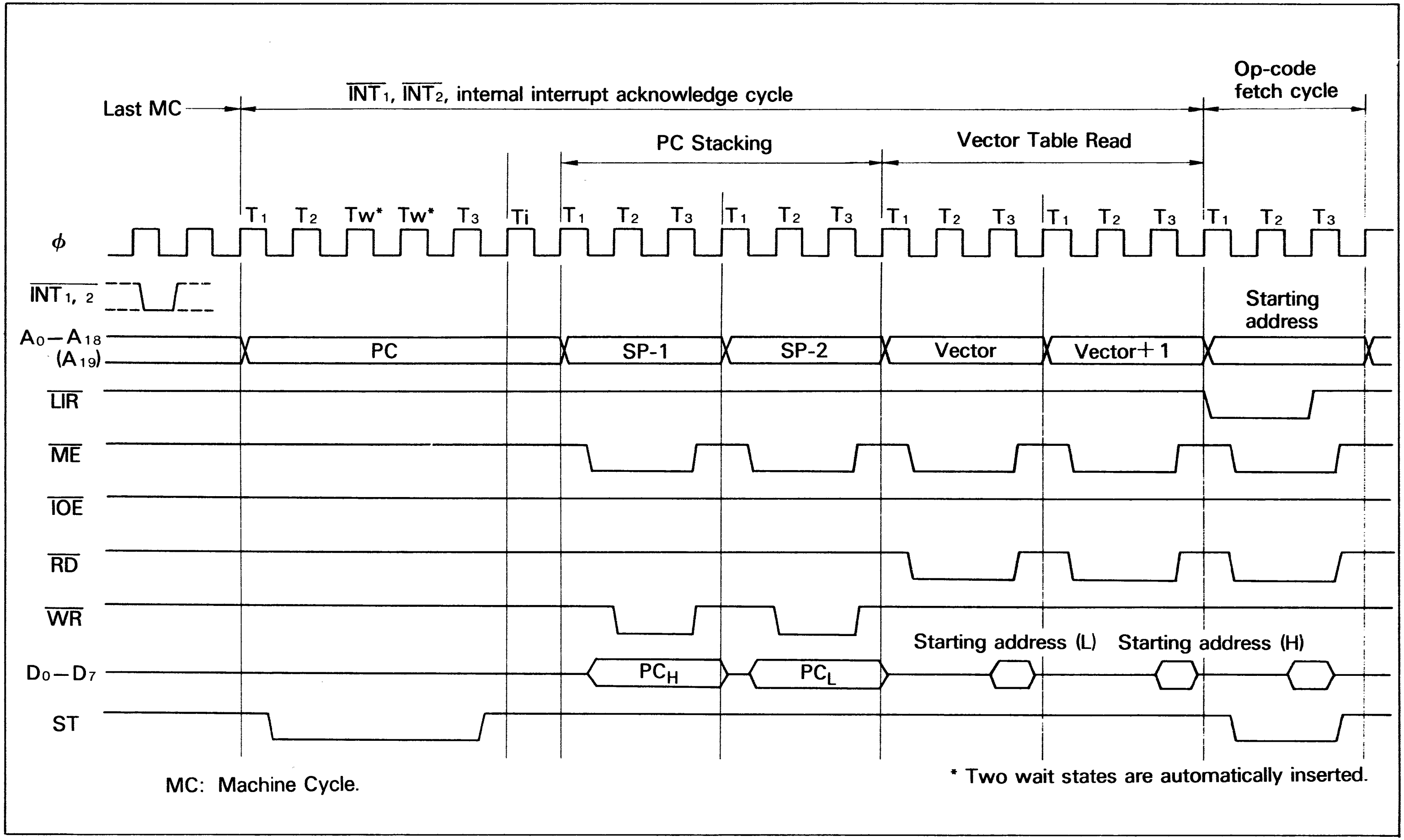
Table 2.7.2 Interrupt Source and Lower Vector

Interrupt Source	Priority	IL			Fixed Code				
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
$\overline{\text{INT}}_1$	Highest	*	*	*	0	0	0	0	0
$\overline{\text{INT}}_2$	↑	*	*	*	0	0	0	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

* Programmable

○ Interrupt Acknowledge Cycle Timing

Fig. 2.7.11 shows interrupt acknowledge cycle timing for internal interrupts, $\overline{\text{INT}}_1$ and $\overline{\text{INT}}_2$. $\overline{\text{INT}}_1$ and $\overline{\text{INT}}_2$ are sampled at the falling edge of clock state prior to T_3 or Ti in the last machine cycle. If $\overline{\text{INT}}_1$ or $\overline{\text{INT}}_2$ 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.



MC: Machine Cycle.

* Two wait states are automatically inserted.

Figure 2.7.11 $\overline{INT_1}$, $\overline{INT_2}$ and Internal Interrupts Timing

2.7.8 Interrupt sources and reset

Interrupt Vector Register (I)

All bits reset to 0.

Since I = 0 locates the vector tables starting at logical address 0000H, vectored interrupts ($\overline{\mathrm{INT}_0}$ Mode 2, $\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$ and internal interrupts) will overlap with fixed restart interrupts like RESET (0), $\overline{\mathrm{NMI}}$ (0066H), $\overline{\mathrm{INT}_0}$ 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.

IL Register

Bits 7 – 5 are reset to 0.

The IL Register can be programmed to locate the vector table for $\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$ and internal interrupts on 32 bytes sub-boundaries within the 256 bytes area specified by I.

IEF_1, IEF_2 Flags

Reset to 0.

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

ITC Register

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

$\overline{\mathrm{INT}_0}$ can be enabled by the EI instruction, which sets IEF_1 = 1. To enable $\overline{\mathrm{INT}_1}$ and $\overline{\mathrm{INT}_2}$ also requires that the ITE1 and ITE2 bits be respectively set = 1 by writing to ITC.

I/O Control Registers

Interrupt enable bits reset to 0.

All HD64180 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 interrupt enable bit.

2.7.9 Difference between $\overline{\mathrm{INT}_0}$ interrupt and the other interrupts

As shown in Fig. 2.7.5, Fig. 2.7.7, Fig. 2.7.9 and Fig. 2.7.11, the interrupt acknowledge cycle of $\overline{\mathrm{INT}_0}$ is different from those of the other interrupts, that is, $\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$ and internal interrupts concerning the state of control signals. The state of the control signals in each interrupt acknowledge cycle are shown below.

$\overline{\mathrm{INT}_0}$ interrupt acknowledge cycle: $\overline{\mathrm{LIR}}$ = 0, $\overline{\mathrm{IOE}}$ = 0, ST = 0

$\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$ and internal interrupt acknowledge cycle: $\overline{\mathrm{LIR}}$ = 1, $\overline{\mathrm{IOE}}$ = 1, ST = 0

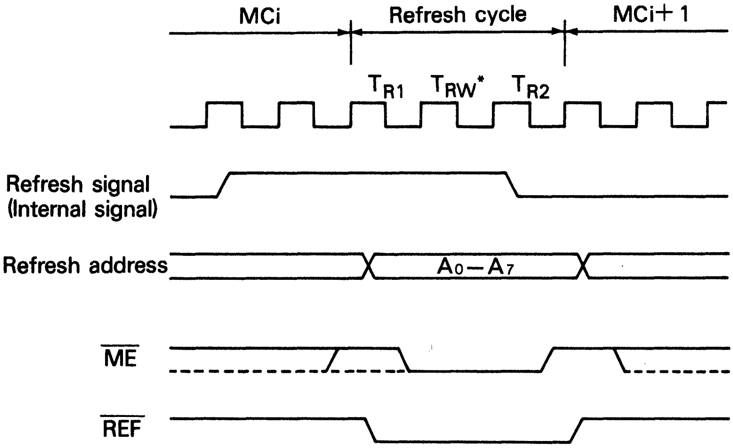
2.8 Dynamic RAM Refresh Control

The HD64180 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 programmable 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 $\overline{\text{REF}}$ 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 $\overline{\text{WAIT}}$ input and the internal wait state generator are not effective during refresh.

Fig. 2.8.1 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 2.8.1 Refresh Timing

Refresh Control Register (RCR)

RCR 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

○ **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.

○ **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.

○ **CYC1, 0: Cycle Interval (bit 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 indicate the best refresh interval depending on CPU clock frequency. CYC0 and CYC1 are cleared to 0 during RESET.

Table 2.8.1 Refresh Interval

CYC1	CYC0	Insertion interval	Time 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	8.0 µs
1	0	40 states	5.0 µs	6.6 µs	10.0 µs	16.0 µs
1	1	80 states	10.0 µs	13.3 µs	20.0 µs	32.0 µs

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.

DYNAMIC RAM REFRESH OPERATION NOTES

(1) Refresh cycle insertion is stopped when the CPU is in the following states.

(a) During RESET
(b) When the bus is released in response to $\overline{\text{BUSREQ}}$
(c) During SLEEP mode

(d) During WAIT states

(2) 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 HD64180 re-acquires the bus depends on the refresh timer, and has no timing relationship with the bus exchange.

(3) 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 depending on the refresh time, and has no timing relationship with the exit from SLEEP mode.

(4) Regarding (2) and (3), the refresh address is incremented by 1 for each successful refresh cycle, not for each refresh request. Thus, independent of the number of 'missed' refresh requests, each refresh bus cycle will use a refresh address incremented by 1 from that of the previous refresh bus cycles.

(5) When the internal refresh requests are asserted during Bus Release mode, all of the requests are ignored in the R0 Mask. Please see Fig. 2.8.2. In the case of R1 Mask, 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 Fig. 2.8.3.

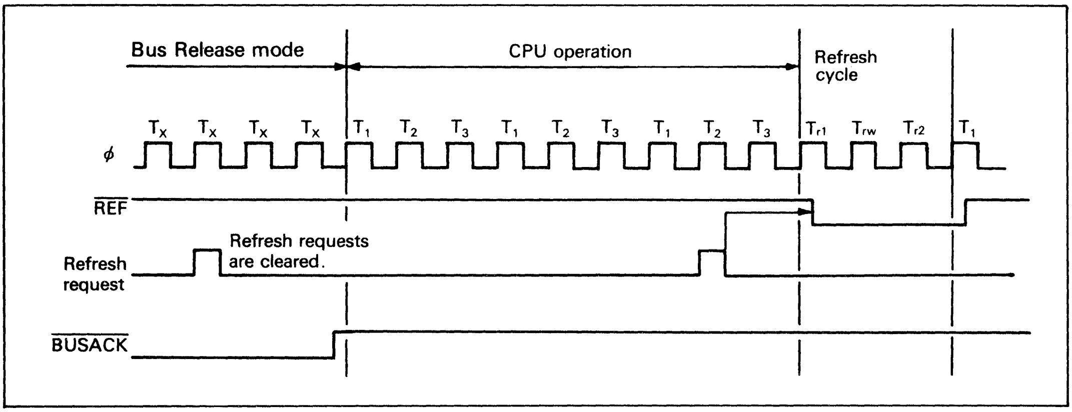


Figure 2.8.2 Refresh Requests during Bus Release Mode (R0 Mask)

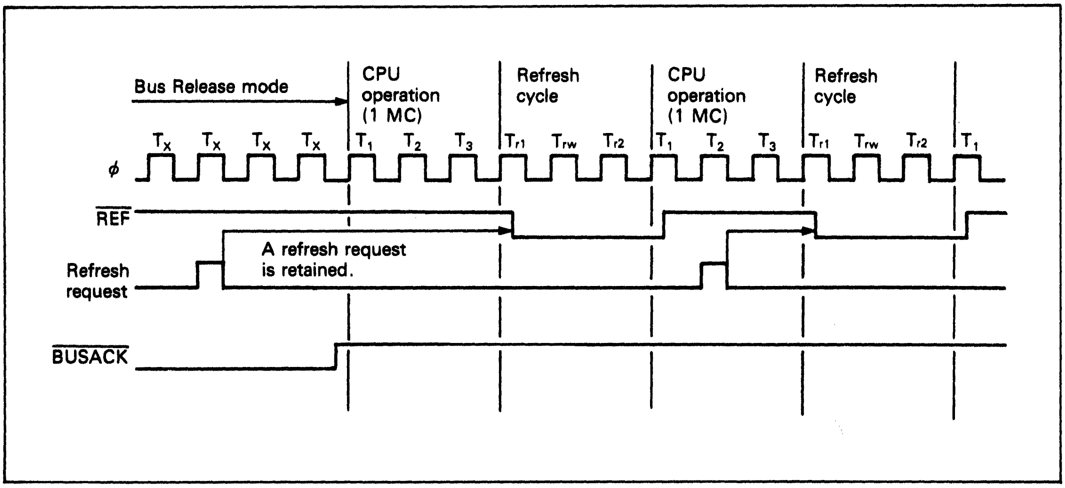


Figure 2.8.3 Refresh Requests during Bus Release Mode (R1 and Z Mask)

2.9 DMA Controller (DMAC)

The HD64180 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 512k or 1M bytes physical address space using 19-bit or 20-bit source and destination memory addresses. In addition, memory transfers can arbitrarily cross 64k bytes physical address boundaries without CPU intervention.

I/O Address Space

I/O source and destination addresses can be directly specified anywhere within the 64k bytes I/O address space (16-bit source and destination I/O addresses).

Transfer Length

Up to 64k bytes can be transferred based on a 16-bit byte count register.

$\overline{\text{DREQ}}$ Input

Level and edge sense $\overline{\text{DREQ}}$ input detection are selectable.

$\overline{\text{TEND}}$ Output

Used to indicate DMA completion to external devices.

Transfer Rate

Each 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).

Additional feature disc for DMA interrupt request by DMA END.

Each channel has the following additional specific capabilities.

Channel 0

○ 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

○ 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 in common.

DSTAT – DMA Status Register
DMODE – DMA Mode Register
DCNTL – DMA Control Register

2.9.1 DMAC block diagram

Fig. 2.9.1 shows the HD64180 DMAC Block Diagram.

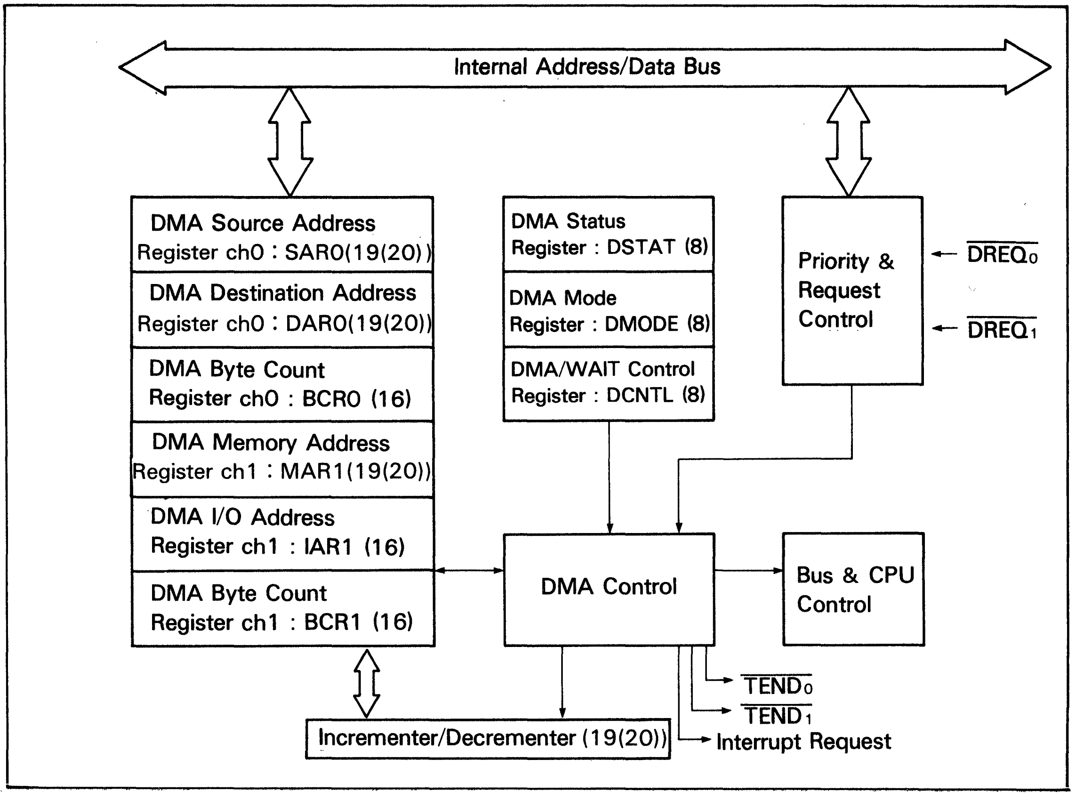

Internal Address/Data Bus
DMA Source Address Register ch0 : SAR0(19(20))
DMA Destination Address Register ch0 : DAR0(19(20))
DMA Byte Count Register ch0 : BCR0 (16)
DMA Memory Address Register ch1 : MAR1(19(20))
DMA I/O Address Register ch1 : IAR1 (16)
DMA Byte Count Register ch1 : BCR1 (16)
DMA Status Register : DSTAT (8)
DMA Mode Register : DMODE (8)
DMA/WAIT Control Register : DCNTL (8)
Priority & Request Control
$\overline{DREQ_0}$
$\overline{DREQ_1}$
DMA Control
Bus & CPU Control
$\overline{TEND_0}$
$\overline{TEND_1}$
Interrupt Request
Incrementer/Decrementer (19(20))

Figure 2.9.1 DMAC Block Diagram

2.9.2 DMAC register description

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

Specifies the physical source address for channel 0 transfers. The register contains 19 bits or 20 bits and may specify up to 512k or 1M bytes memory addresses or up to 64k bytes I/O addresses. Channel 0 source can be memory, I/O or memory mapped I/O.

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

Specifies the physical destination address for channel 0 transfers. The register contains 19 bits or 20 bits and may specify up to 512k or 1M bytes memory addresses or up to 64k bytes I/O addresses. Channel 0 destination can be memory, I/O or memory mapped I/O.

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

Specifies the number of bytes to be transferred. This register contains 16 bits and may specify up to 64k bytes 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.

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

Specifies the physical memory address for channel 1 transfers. This may be destination or source memory address.

This register contains 19 bits or 20 bits and may specify up to 512k or 1M bytes memory addresses.

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

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 64k bytes I/O addresses.

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

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

DMA Status Register (DSTAT)

DSTAT is used to enable and disable DMA transfer and DMA termination interrupts. DSTAT also allows determining the status of a DMA transfer i.e. 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

○ **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, $\overline{\text{DWE1}}$ should be written with 0 during the same register write access. Writing DE1 to 0 disables channel 1 DMA, but DMA is restartable. Writing DE1 to 1 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, $\overline{\text{DWE0}}$ should be written with 0 during the same register write access. Writing DE0 to 0 disables channel 0 DMA. Writing DE0 to 1 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, $\overline{\text{DWE1}}$ should be written with 0 during the same access. $\overline{\text{DWE1}}$ write value of 0 is not held and $\overline{\text{DWE1}}$ is always read as 1.

○ **$\overline{\text{DWE0}}$: DE0 Bit Write Enable (bit 4)**

When performing any software write to DE0, $\overline{\text{DWE0}}$ should be written with 0 during the same access. $\overline{\text{DWE0}}$ write value of 0 is not held and $\overline{\text{DWE0}}$ 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 channel 0 of 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 NMI interrupt service routine. To restart DMA, DE0 and/or DE1 should be written with 1 (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. 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.

DMA Mode Register (DMODE)

DMODE is used to set 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	

○ DM1, DM0: Destination Mode Channel 0 (bits 5, 4)

Specifies whether the destination for channel 0 transfers is memory, I/O or memory mapped I/O and the corresponding address modifier. DM1 and DM0 are cleared to 0 during RESET.

Table 2.9.1 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)

Specifies whether the source for channel 0 transfers is memory, I/O or memory mapped I/O and the corresponding address modifier. SM1 and SM0 are cleared to 0 during RESET.

Table 2.9.2 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 2.9.3 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 2.9.3 Combination of Transfer Mode

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	

* : 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.

DMA/WAIT Control Register (DCNTL)

DCNTL controls the insertion of wait states into DMAC (and CPU) accesses of memory or I/O. Also, the DMA request mode for each $\overline{\text{DREQ}}$ ($\overline{\text{DREQ}_0}$ and $\overline{\text{DREQ}_1}$) 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

○ **MWI1, MWI0: Memory Wait Insertion (bits 7-6)**

Specifies the number of wait states introduced into CPU or DMAC memory access cycles. MWI1 and MWI0 are set to 1 during RESET. See section of Wait State Control for details.

○ **IWI1, IWI0: I/O Wait Insertion (bits 5-4)**

Specifies 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 of Wait State Control for details.

○ **DMS1, DMS0: DMA Request Sense (bits 3-2)**

DMS1 and DMS0 specify the DMA request sense for channel 0 ($\overline{\text{DREQ}_0}$) and channel 1 ($\overline{\text{DREQ}_1}$) respectively. When reset to 0, the input is level sense. When set to 1, the input is edge sense. DMS1 and DMS0 are cleared to 0 during RESET.

○ **DIM1, DIM0: DMA Channel 1 I/O and Memory Mode (bits 1-0)**

Specifies the source/destination and address modifier for channel 1 memory to/from I/O transfer modes. IM1 and IM0 are cleared to 0 during RESET.

Table 2.9.4 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

2.9.3 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) as well as Channel 1 DMA are described.

Memory to/from Memory — Channel 0

For memory to/from memory transfers, the external $\overline{\text{DREQ}_0}$ 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, the DMA and CPU operation are alternated 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. Fig. 2.9.2 shows cycle steal mode DMA timing.

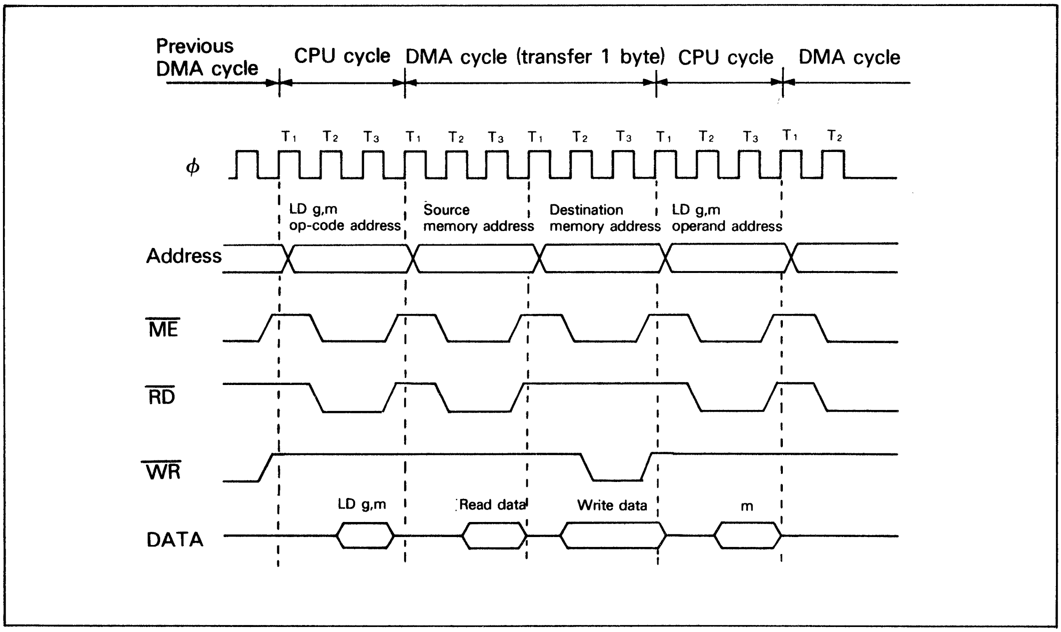


Figure 2.9.2 Cycle Steal Mode DMA Timing

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 $\overline{\mathrm{DWE0}}$ = 0 in the same access) in DSTAT and the DMA operation will start 1 machine cycle later. If interrupt occurs at the same time, the DIE0 bit should be set to 1.

Memory to/from I/O (Memory Mapped I/O) – Channel 0

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

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

When level sense is programmed, the DMA operation begins when $\overline{\mathrm{DREQ}_0}$ is sampled LOW. If $\overline{\mathrm{DREQ}_0}$ is sampled HIGH, after the next DMA byte transfer, control is relinquished to the HD64180 CPU. As shown in Fig. 2.9.3. $\overline{\mathrm{DREQ}_0}$ is sampled at the rising edge of the clock cycle prior to T_3 i.e. either T_2 or Tw.

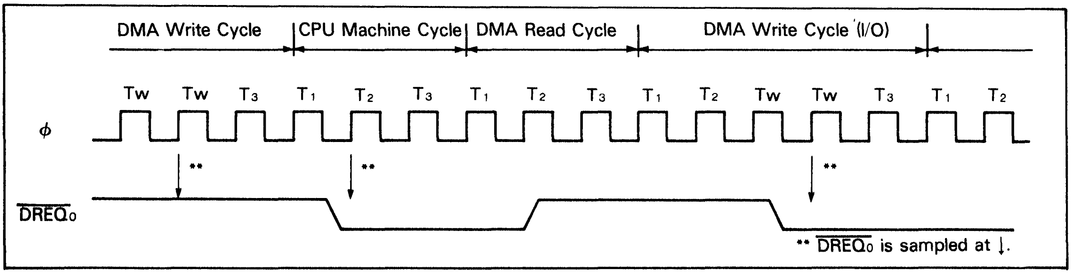


Figure 2.9.3 CPU Operation and DMA Operation ($\overline{\mathrm{DREQ}_0}$ is programmed for level sense)

When edge sense is programmed, DMA operation begins at the falling edge of $\overline{\mathrm{DREQ}_0}$. If another falling edge is detected before the rising edge of the clock prior to T_3 during DMA write cycle (i.e. 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 $\overline{\mathrm{DREQ}_0}$ falling edge is detected before the rising edge of the clock prior to T_3 at which time the DMA operation will (re)start. Fig. 2.9.4 shows the edge sense DMA timing.

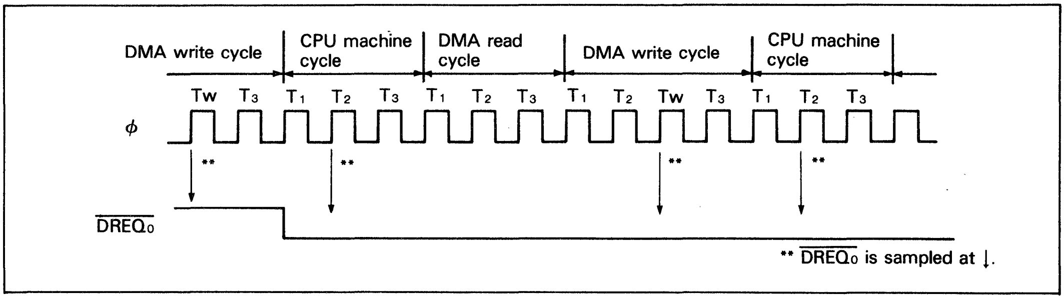


Figure 2.9.4 CPU Operation and DMA Operation (DREQ0 is programmed for edge sense)

During the transfers for channel 0, the TEND0 output will go LOW synchronous with the write cycle of the last (BCR0 = 00H) DMA transfer as shown in Fig. 2.9.5.

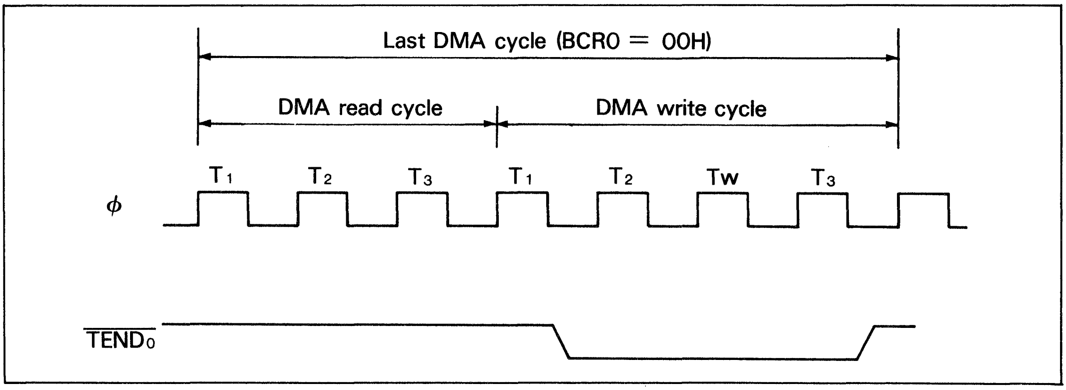


Figure 2.9.5 TEND0 Output Timing

The DREQ0 and TEND0 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/DREQ0 pin automatically functions as input pin even if it has been programmed as output pin for CKA0. And the CKA1/TEND0 pin functions as output pin for TEND0 by setting CKA1D to 1 in CNTLA1.

Figure 2.9.6 shows memory to/from memory mapped I/O transfer timing and Figure 2.9.7 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.

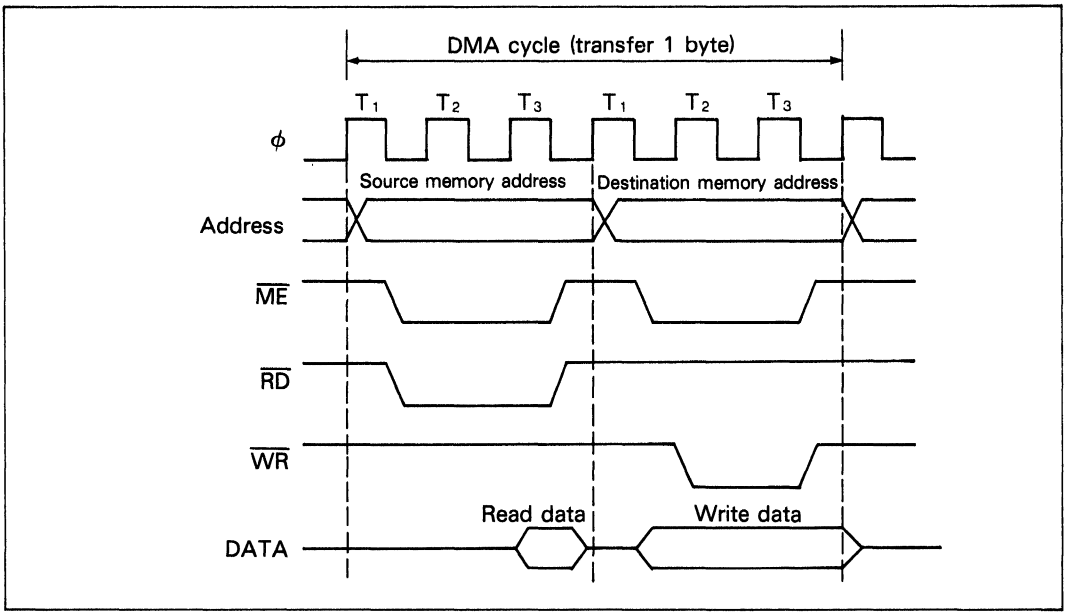


Figure 2.9.6 DMA Cycle (Memory to/from Memory Mapped I/O (Memory))

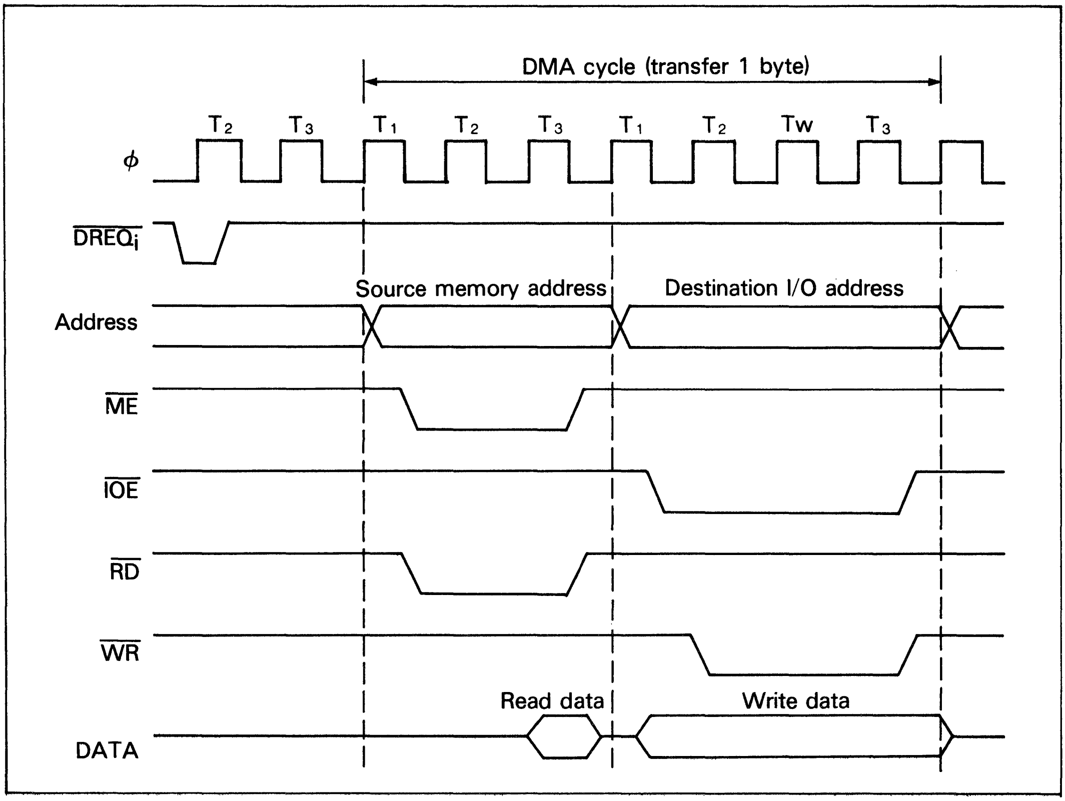


Figure 2.9.7 DMA Cycle (Memory to I/O)

In the case of R1 and Z Mask

(1) Load the memory and I/O or memory mapped I/O source and destination addresses (A_9-A_{18} (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.

In the case of R0 Mask

(1) Load the memory and I/O or memory mapped I/O source and destination addresses into SAR0 and DAR0. Note that I/O addresses (not memory mapped I/O) are limited to 16 bits (A_0-A_{15}). Make sure that bits A_{16}, and A_{17} are 0 (A_{18} is a don't care) to correctly enable the external $\overline{DREQ_0}$ input.
(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.

Memory to/from ASCI — Channel 0

Channel 0 has 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 are used to 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 the following table to enable use of the appropriate ASCI status bit as an internal DMA request.

Table 2.9.5 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

(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.

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.

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 (19 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 $\overline{\text{DWE1}}$ = 0 in the same access) in DSTAT and the DMA operation with the external I/O device will begin using the external $\overline{\text{DREQ}_1}$ input and $\overline{\text{TEND}_1}$ output.

2.9.4 DMA bus timing

When memory (and memory mapped I/O) is specified as a source or destination, $\overline{\text{ME}}$ goes LOW during the memory access. When I/O is specified as a source or destination, $\overline{\text{IOE}}$ 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 $\overline{\text{DREQ}}$ input and the $\overline{\text{TEND}}$ 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, 1 wait state is automatically inserted. Additional wait states can be inserted by programming the on-chip wait state generator or using the external $\overline{\text{WAIT}}$ 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 $\overline{\text{DREQ}_0}$ 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 64k bytes boundaries), the minimum bus cycle is extended to four clocks by automatic insertion of one internal Ti state.

2.9.5 DMAC channel priority

For simultaneous $\overline{\text{DREQ}_0}$ and $\overline{\text{DREQ}_1}$ 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.

2.9.6 DMAC and $\overline{\text{BUSREQ}}$, $\overline{\text{BUSACK}}$

The $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ inputs allow another bus master to take control of the HD64180 bus. $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ 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 HD64180 bus ($\overline{\text{BUSREQ}}$ HIGH), the on-chip DMAC will correctly continue the suspended DMA operation.

2.9.7 DMAC internal interrupts

Fig. 2.9.8 illustrates the internal DMA interrupt request generation circuit.

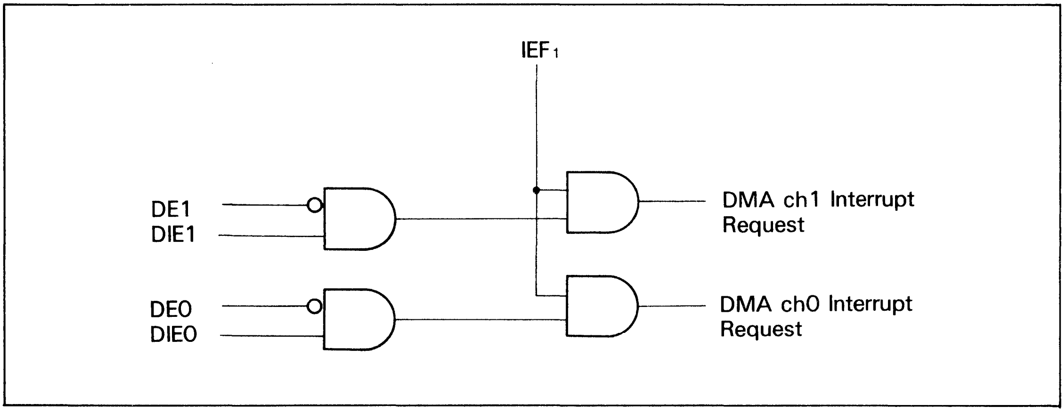


Figure 2.9.8 DMAC Interrupt Request Circuit Diagram

DE0 and DE1 are automatically cleared to 0 by the HD64180 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 (i.e. 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 resume by programming the channel DE bit = 1.

2.9.8 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 channels DE bit to 1, that channels operation can be restarted, and DMA will correctly resume from the point at which it was suspended by $\overline{\text{NMI}}$. See Fig. 2.9.9 for details.

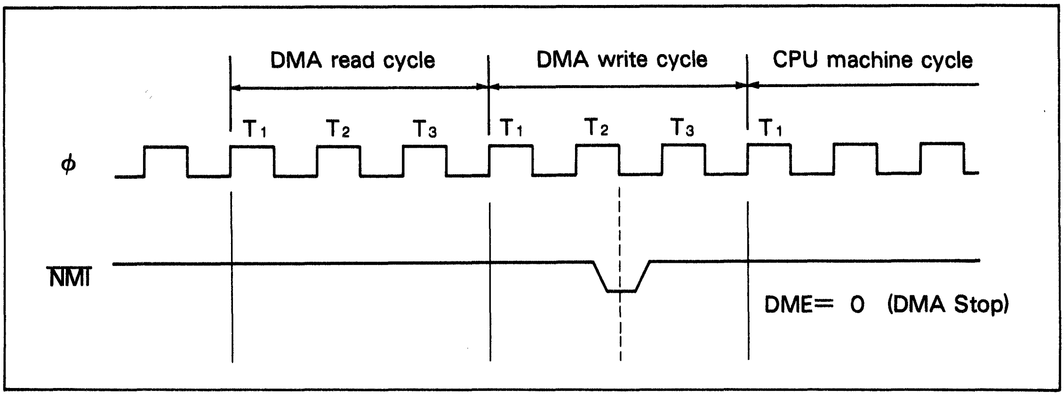


Figure 2.9.9 $\overline{\text{NMI}}$ and DMA Operation

2.9.9 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.

2.10 Asynchronous Serial Communication Interface (ASCI)

The HD64180 on-chip ASCI has two independent full duplex channels. Based on full programmability of the following functions, 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 38.4k bits per second (CPU f_C = 6.144 MHz)
○ Modem control signals – Channel 0 has $\overline{DCD_0}$, $\overline{CTS_0}$ and $\overline{RTS_0}$ Channel 1 has $\overline{CTS_1}$
○ Programmable interrupt condition enable and disable
○ Operation with on-chip DMAC

2.10.1 ASCI block diagram

Fig. 2.10.1 shows the ASCI Block Diagram.

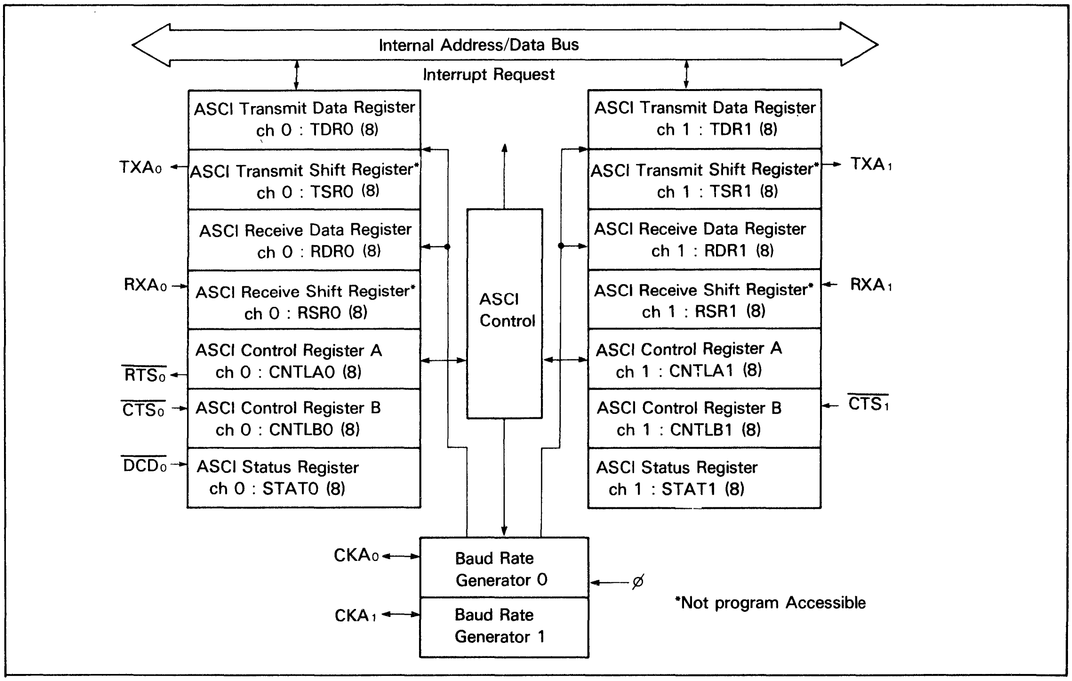

Internal Address/Data Bus
Interrupt Request
ASCI Transmit Data Register ch 0 : TDR0 (8)
ASCI Transmit Shift Register* ch 0 : TSR0 (8)
ASCI Receive Data Register ch 0 : RDR0 (8)
ASCI Receive Shift Register* ch 0 : RSR0 (8)
ASCI Control Register A ch 0 : CNTLA0 (8)
ASCI Control Register B ch 0 : CNTLB0 (8)
ASCI Status Register ch 0 : STAT0 (8)
ASCI Transmit Data Register ch 1 : TDR1 (8)
ASCI Transmit Shift Register* ch 1 : TSR1 (8)
ASCI Receive Data Register ch 1 : RDR1 (8)
ASCI Receive Shift Register* ch 1 : RSR1 (8)
ASCI Control Register A ch 1 : CNTLA1 (8)
ASCI Control Register B ch 1 : CNTLB1 (8)
ASCI Status Register ch 1 : STAT1 (8)
ASCI Control
TXA0
RXA0
RTS0
CTS0
DCD0
TXA1
RXA1
CTS1
CKA0
CKA1
Baud Rate Generator 0
Baud Rate Generator 1
ϕ
*Not program Accessible

Figure 2.10.1 ASCI Block Diagram

2.10.2 ASCI register description

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

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.

ASCI Transmit Data Register 0, 1 (TDR0, 1: 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 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 by this read operation.

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

This 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.

ASCI Receive Data Register 0, 1 (RDR0, 1: 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 read-only-register.

However, if RDRF = 0, data can be written into the ASCI Receive Data Register, and the data can be read.

ASCI Status Register 0, 1 (STAT0, 1)

Each channel status register 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

○ **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 IOSTOP 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 IOSTOP mode and during RESET.

○ **PE: Parity Error (bit 5)**

PE is set to 1 when a parity error is detected on 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 the EFR bit (Error Flag Reset) of CNTLA is written to 0, when $\overline{DCD_0}$ is HIGH, in IOSTOP mode and during RESET.

○ **FE: Framing Error (bit 4)**

If a receive data byte frame is delimited by an invalid stop bit (i.e. 0, should be 1), FE is set to 1. FE is cleared to 0 when the EFR bit (Error Flag Reset) of CNTLA is written to 0, when $\overline{DCD_0}$ is HIGH, in IOSTOP mode and during RESET.

○ **RIE: Receive Interrupt Enable (bit 3)**

RIE should be set to 1 to enable ASCI receive interrupt requests. When RIE 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 $\overline{\text{CTS}_1}$ 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 IOSTOP mode and during RESET. When the external $\overline{\text{CTS}}$ 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.

ASCI Control Register A0, 1 (CNTLA0, 1)

Each ASCI channel Control Register A 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	$\overline{\text{RTS}_0}$	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

○ **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 IOSTOP 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 IOSTOP mode and during RESET.

○ **$\overline{\text{RTS}_0}$ – Request to Send Channel 0 (bit 4 in CNTLA0)**

When $\overline{\text{RTS}_0}$ is reset to 0, the $\overline{\text{RTS}_0}$ output pin will go LOW. When $\overline{\text{RTS}_0}$ is set to 1, the $\overline{\text{RTS}_0}$ output immediately goes HIGH. $\overline{\text{RTS}_0}$ is set to 1 during RESET.

○ **CKA1D: CKA1 Clock Disable (bit 4 in CNTLA1)**

When CKA1D is set to 1, the multiplexed CKA1/$\overline{\text{TEND}_0}$ pin is used for the $\overline{\text{TEND}_0}$ 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 written to 0, the EFR function is selected to reset all error flags (OVRN, FE and PE) to 0. MPBR/EFR is undefined during RESET.

○ **MOD2, 1, 0: ASCI Data Format Mode 2, 1, 0 (bits 2-0)**

These bits 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 2.10.1.

Table 2.10.1 Combination of Data Format

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

ASCI Control Register B0, 1 (CNTLB0, 1)

Each ASCI channel control register B 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

○ **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 (i.e. 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 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, 1, 0: Source/Speed Select 2, 1, 0 (bits 2-0)**

Specify the data clock source (internal or external) and baud rate prescale factor. SS2, SS1, SS0 are all set to 1 during RESET. Table 2.10.2 shows the divide ratio corresponding to SS2, SS1 and SS0.

Table 2.10.2 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$_0$/$\overline{\text{DREQ}_0}$ and CKA$_1$/$\overline{\text{TEND}_0}$). 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$_0$/$\overline{\text{DREQ}_0}$ 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 the CKA$_1$/$\overline{\text{TEND}_0}$ reverts to the DMA Control output function regardless of SS2, SS1 and SS0 programming.

Final data clock rates are based on $\overline{\text{CTS}}$/PS (prescale), DR, SS2, SS1, SS0 and the HD64180 system clock (ϕ) frequency as shown in Table 2.10.3.

Table 2.10.3 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	ϕ÷10	0	16	0	0	0	÷1	ϕ÷160	38400		19200	O	ϕ÷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	ϕ÷640	9600		4800	O	ϕ÷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	ϕ÷30	0	16	0	0	0	÷1	ϕ÷480		9600		O	ϕ÷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	ϕ÷1920		2400		O	ϕ÷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

2.10.3 MODEM control signals

ASCI channel 0 has $\overline{\text{CTS}_0}$, $\overline{\text{DCD}_0}$ and $\overline{\text{RTS}_0}$ external modem control signals. ASCI channel 1 has a $\overline{\text{CTS}_1}$ modem control signal which is multiplexed with RXS pin (Clocked Serial Receive Data).

$\overline{\text{CTS}_0}$: Clear to Send 0 (input)

The $\overline{\text{CTS}_0}$ input allows external control (start/stop) of ASCI channel 0 transmit operations. When $\overline{\text{CTS}_0}$ is HIGH, channel 0 TDRE bit is held at 0 regardless of whether the TDR0 (Transmit Data Register) is full or empty. When $\overline{\text{CTS}_0}$ is LOW, TDRE will reflect the state of TDR0. Note that the actual transmit operation is not disabled by $\overline{\text{CTS}_0}$ HIGH, only TDRE is inhibited.

$\overline{\text{DCD}_0}$: Data Carrier Detect 0 (input)

The $\overline{\text{DCD}_0}$ input allows external control (start/stop) of ASCI channel 0 receive operations. When $\overline{\text{DCD}_0}$ 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 $\overline{\text{DCD}_0}$ 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 $\overline{\text{DCD}_0}$ input is HIGH ($\overline{\text{DCD0}}$ bit = 1) even though it has gone LOW. Thus, the STAT0 register should be read twice to insure the $\overline{\text{DCD0}}$ bit is reset to 0.

$\overline{\text{RTS}_0}$: Request to Send 0 (output)

$\overline{\text{RTS}_0}$ allows the ASCI to control (start/stop) another communication devices transmission (for example, by connection to that devices $\overline{\text{CTS}}$ input). $\overline{\text{RTS}_0}$ is essentially a 1 bit output port, having no side effects on other ASCI registers or flags.

$\overline{\text{CTS}_1}$: Clear to Send 1 (input)

Channel 1 $\overline{\text{CTS}_1}$ input is multiplexed with the RXS pin (Clocked Serial Receive Data). The $\overline{\text{CTS}_1}$ function is selected when the CTS1E bit in STAT1 is set to 1. When enabled, the $\overline{\text{CTS}_1}$ operation is equivalent to $\overline{\text{CTS}_0}$.

Modem control signal timing is shown in Fig. 2.10.2 (a) and Fig. 2.10.2 (b).

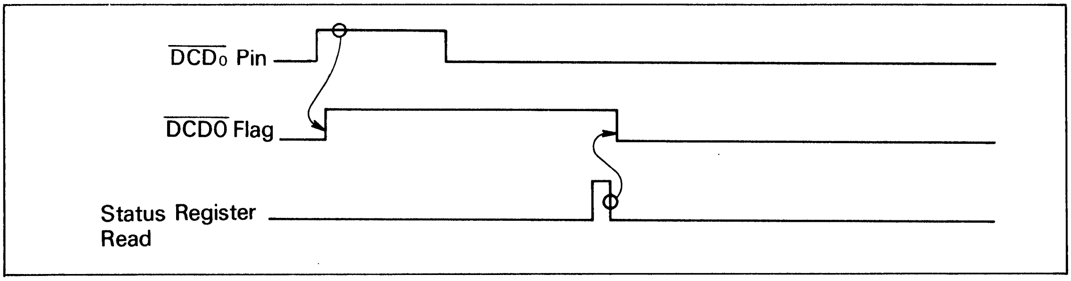


Figure 2.10.2 (a) $\overline{\text{DCD}_0}$ Timing

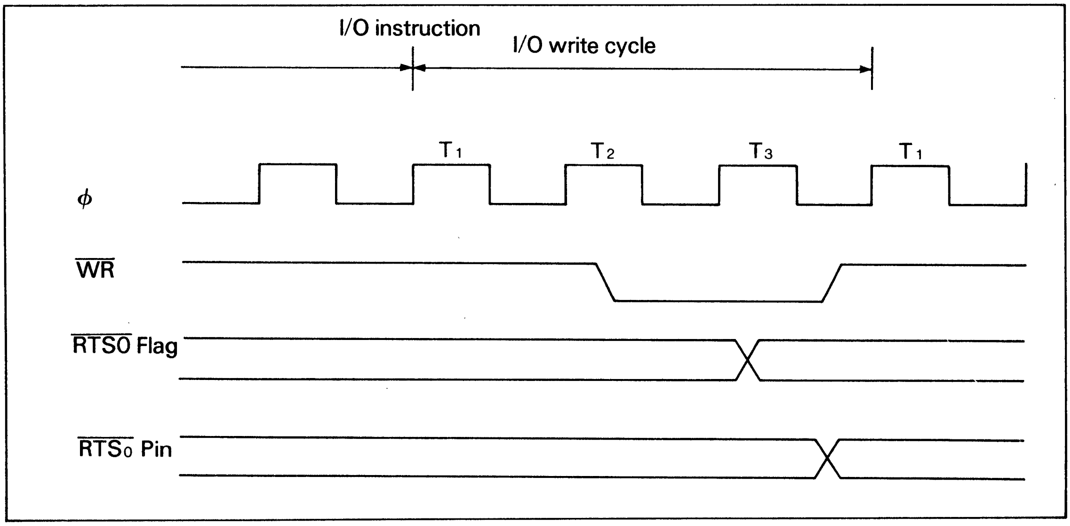


Figure 2.10.2 (b) $\overline{\text{RTS}_0}$ Timing

2.10.4 ASCI interrupts

Fig. 2.10.3 shows the ASCI interrupt request generation circuit.

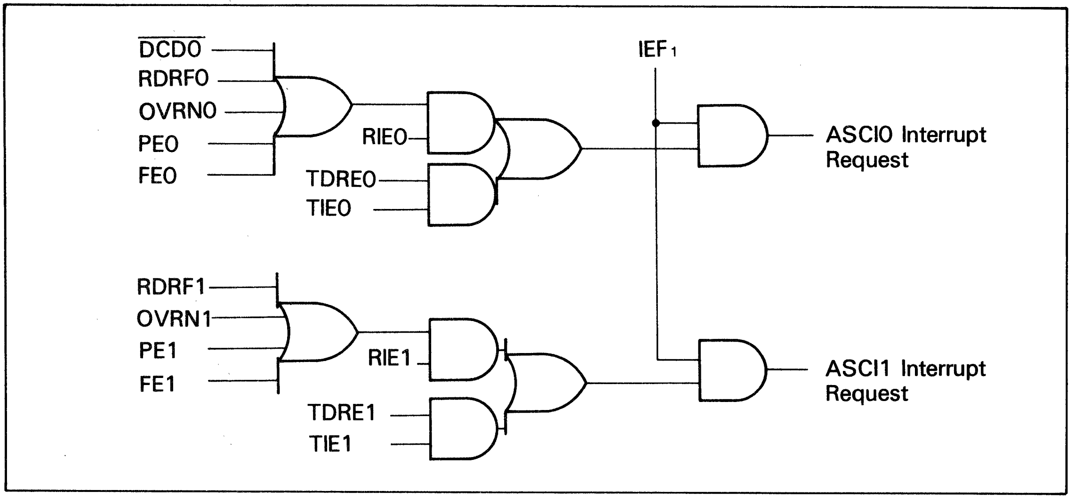


Figure 2.10.3 ASCI Interrupt Request Circuit Diagram

2.10.5 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.

2.10.6 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.

2.10.7 ASCI clock

In external clock input mode, the external clock is directly input to the sampling rate (÷16/÷64) as shown in Fig. 2.10.4.

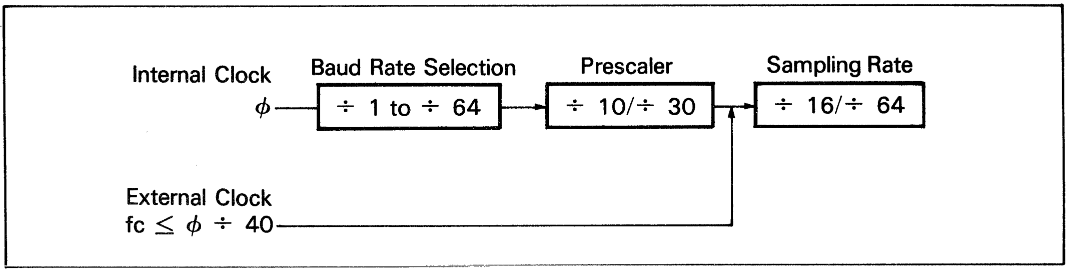


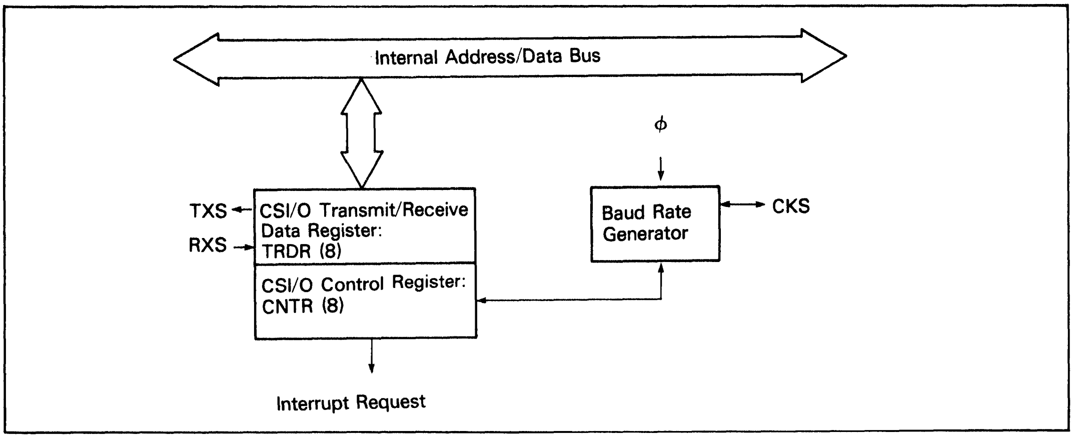
Figure 2.10.4 ASCI Clock Block Diagram

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

The HD64180 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 200k bits/second at $f_C = 4$ MHz) is provided. The CSI/O is ideal for implementing a multiprocessor communication link between the HD64180 and the HMCS400 series (4-bit) and the HD6301 series (8-bit) single chip controllers as well as additional HD64180 CPUs. These secondary devices may typically perform a portion of the system I/O processing such as keyboard scan/decode, LDC interface etc.

2.11.1 CSI/O block diagram

The CSI/O block diagram is shown in Fig. 2.11.1. The CSI/O consists of two registers — the Transmit/Receive Data Register (TRDR) and Control Register (CNTR).



Internal Address/Data Bus
ϕ
TXS
RXS
CSI/O Transmit/Receive Data Register: TRDR (8)
CSI/O Control Register: CNTR (8)
Baud Rate Generator
CKS
Interrupt Request

Figure 2.11.1 CSI/O Block Diagram

2.11.2 CSI/O register description

CSI/O Transmit/Receive Data Register (TRDR: I/O Address = OBH)

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 operation can't occur simultaneously). For example, if a CSI/O transmission is attempted at the same time that the CSI/O is receiving data, a 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.

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

CNTR is used to monitor CSI/O status, enable and disable the CSI/O, enable and disable interrupt generation and select 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

○ 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 IOSTOP 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, EF is set to 1 and an interrupt (if enabled by EIE = 1) will be generated. Note that RE and TE should never both be set to 1 at the same time. RE is cleared to 0 during RESET and IOSTOP mode.

Note that the RXS pin is multiplexed with $\overline{CTS_1}$ 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, EF is set to 1 and an interrupt (if enabled by EIE = 1) will be generated. Note that TE and RE should never both be set to 1 at the same time. TE is cleared to 0 during RESET and IOSTOP mode.

○ **SS2, 1, 0: 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 2.11.1 shows CSI/O Baud Rate Selection.

Table 2.11.1 CSI/O Baud Rate Selection

SS2	SS1	SS0	Divide Ratio	Baud Rate
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)	

( ) shows the baud rate (BPS) at 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.

2.11.3 CSI/O interrupts

The CSI/O interrupt request circuit is shown in Fig. 2.11.2.

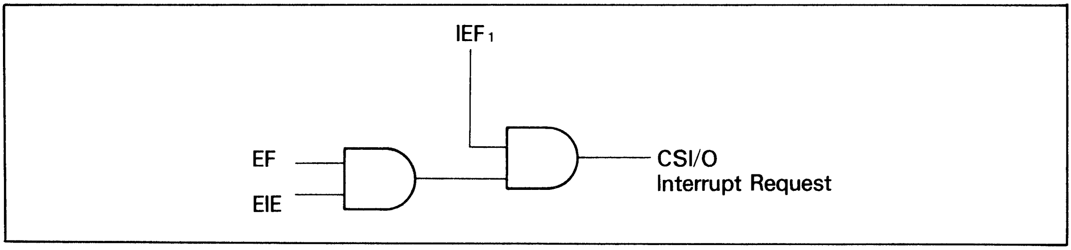


Figure 2.11.2 CSI/O Interrupt Circuit Diagram

2.11.4 CSI/O operation

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

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.

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.

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.

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.

2.11.5 CSI/O operation timing notes

(1) Note that transmitter clocking and receiver sampling timings are different from internal and external clocking modes. Fig. 2.11.3 to Fig. 2.11.6 shows CSI/O Transmit/Receive Timing.

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

2.11.6 CSI/O operation notes

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

(2) When RE or TE is cleared to 0 by software, a corresponding receive or transmit operation is immediately terminated. Normally, TE or RE should only be cleared to 0 when EF = 1.

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

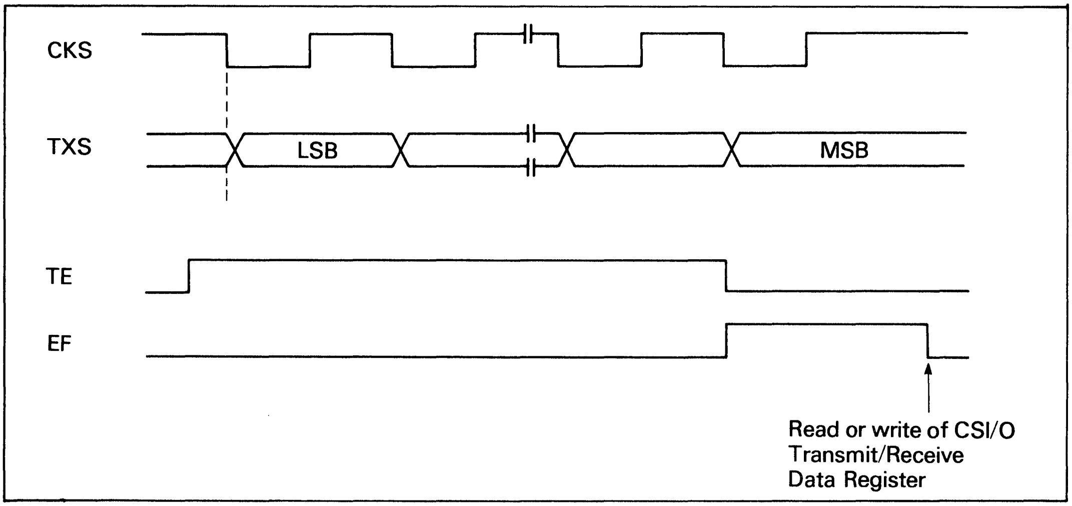


Figure 2.11.3 Transmit Timing — Internal Clock

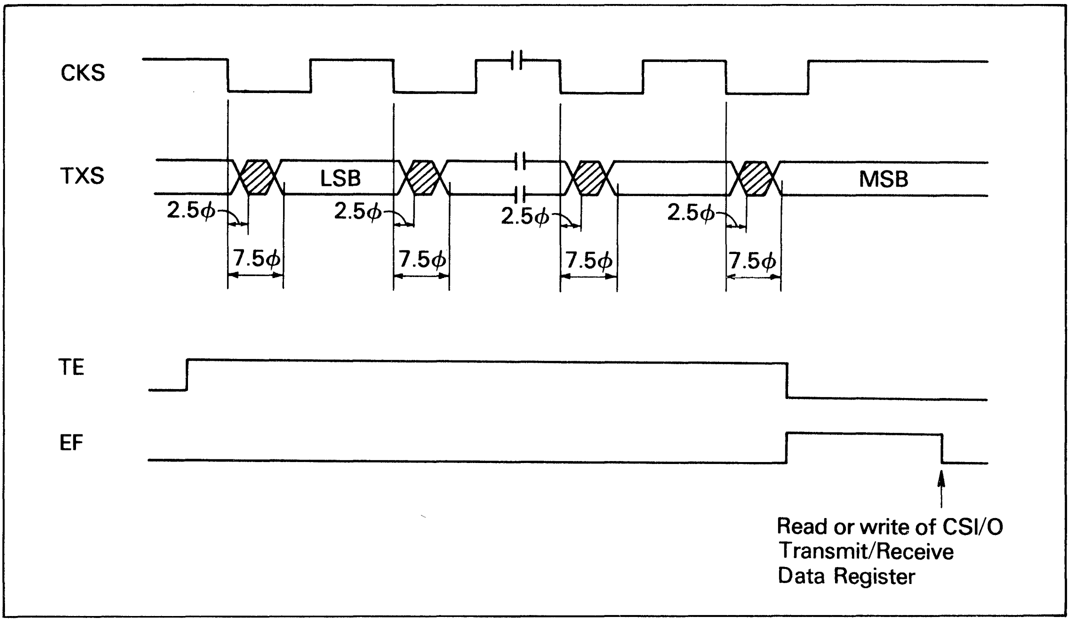


Figure 2.11.4 Transmit Timing — External Clock

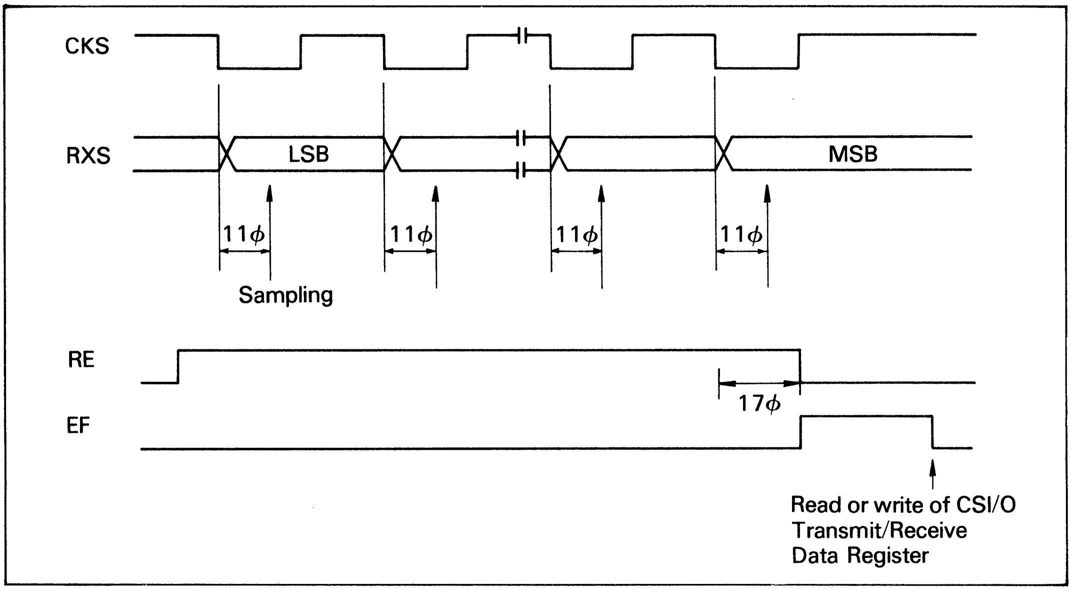


Figure 2.11.5 Receive Timing — Internal Clock

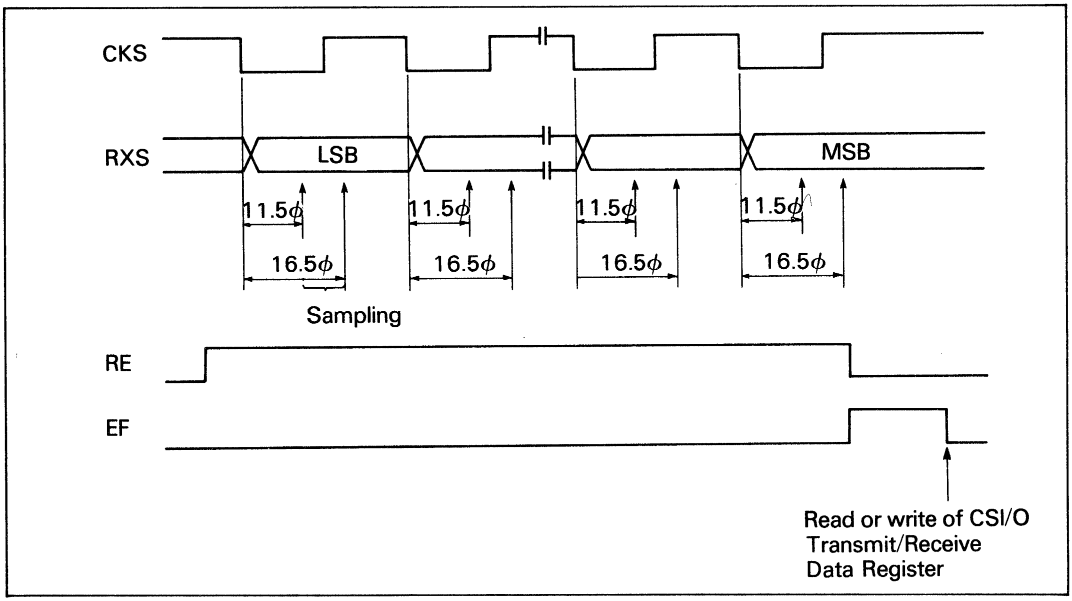


Figure 2.11.6 Receive Timing — External Clock

2.11.7 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 $\overline{\text{RESET}}$. However, the contents of TRDR are not changed.

2.12 Programmable Reload Timer (PRT)

The HD64180 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 (multiplexed with A_{18}) which can be set HIGH, LOW or toggled. Thus PRT1 can perform programmable output waveform generation.

2.12.1 PRT block diagram

The PRT block diagram is shown in Fig. 2.12.1. 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.

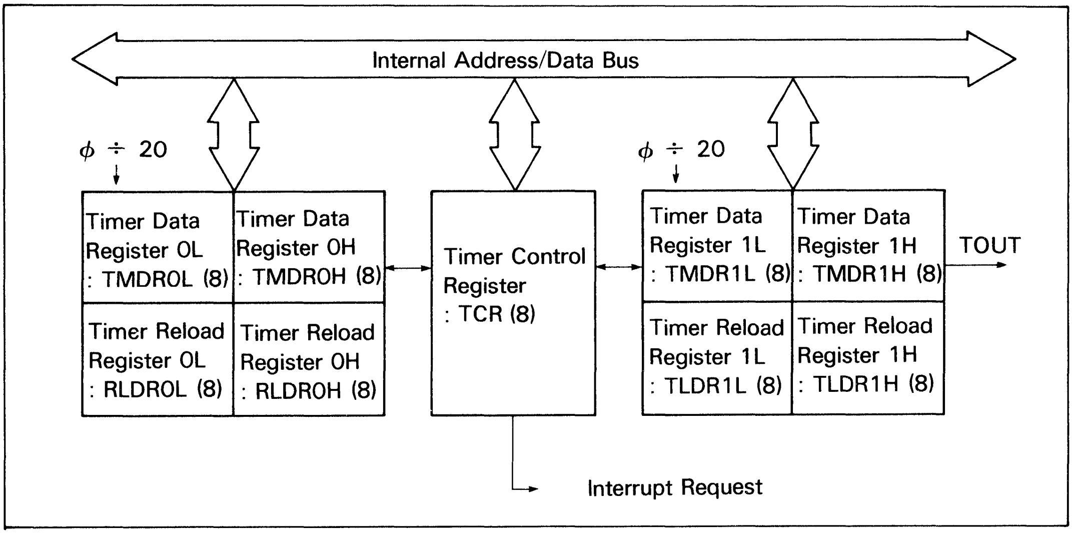

Internal Address/Data Bus
φ ÷ 20
Timer Data Register OL : TMDROL (8)
Timer Data Register OH : TMDROH (8)
Timer Reload Register OL : RLDROL (8)
Timer Reload Register OH : RLDROH (8)
Timer Control Register : TCR (8)
φ ÷ 20
Timer Data Register 1L : TMDR1L (8)
Timer Data Register 1H : TMDR1H (8)
Timer Reload Register 1L : TLDR1L (8)
Timer Reload Register 1H : TLDR1H (8)
TOUT
Interrupt Request

Figure 2.12.1 PRT Block Diagram

2.12.2 PRT register description

Timer Data Register (TMDR: I/O Address = CHO: ODH, OCH 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 potential 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, the TMDR down counting must be inhibited using the TDE (Timer Down Count Enable) bits in the TCR (Timer Control Register), following which any or both higher and lower bytes of TMDR can be freely written (and read) in any order.

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.

Timer Control Register (TCR)

TCR monitors both channels (PRT0, PRT1) TMDR status and controls enabling and disabling of down counting and interrupts as well as controlling the output pin (A_{18}/TOUT) 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

○ 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 are 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 to 1, TIF0 = 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, 0: Timer Output Control (bits 3, 2)

TOC1 and TOC0 control the output of PRT1 using the multiplexed A_{18}/TOUT pin as shown below. During RESET, TOC1 and TOC0 are cleared to 0. This selects the address function for A_{18}/TOUT. By programming TOC1 and TOC0, the A_{18}/TOUT pin can be forced HIGH, LOW or toggled when TMDR1 decrements to 0.

TOC1	TOC0	OUTPUT	
0	0	Inhibited	(A_{18}/TOUT pin is selected as an address output function.)
0	1	toggled*	(A_{18}/TOUT pin is selected as a PRT1 output function.)
1	0	0	
1	1	1	

* When TMDR1 decrements to 0, TOUT level is reversed. This leads to production of a square wave with 50% duty to the external without any software support.

○ TDE1, 0: 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, down counting is executed for TMDRn. 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.

Fig. 2.12.2 shows timer initialization, count down and reload timing. Fig. 2.12.3 shows timer output (A_{18}/TOUT) timing.

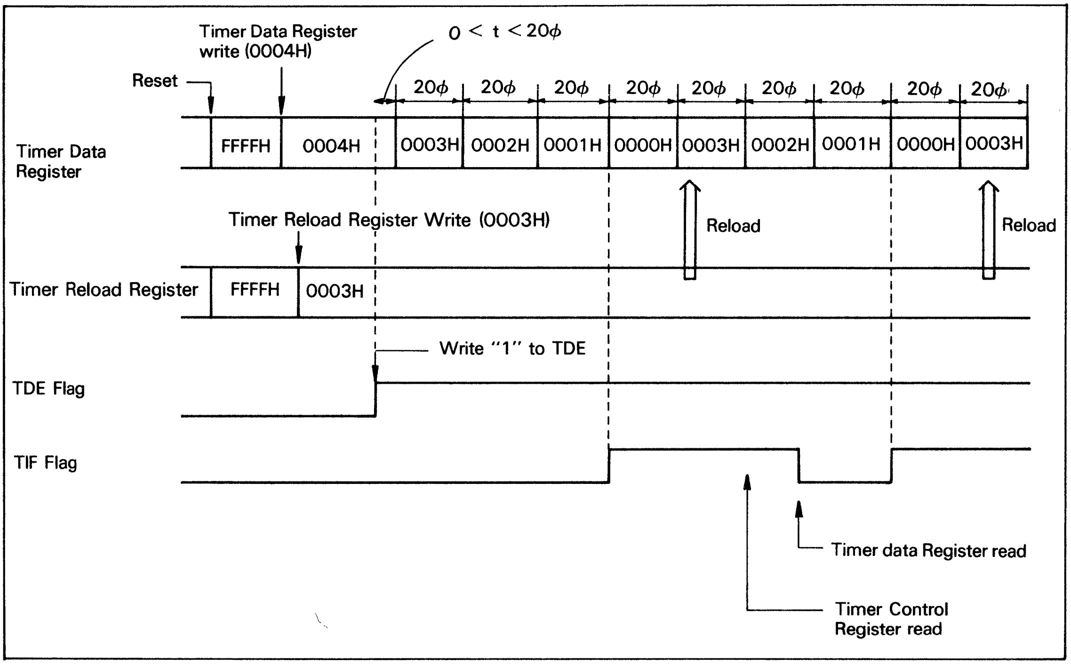


Figure 2.12.2 PRT Operation Timing

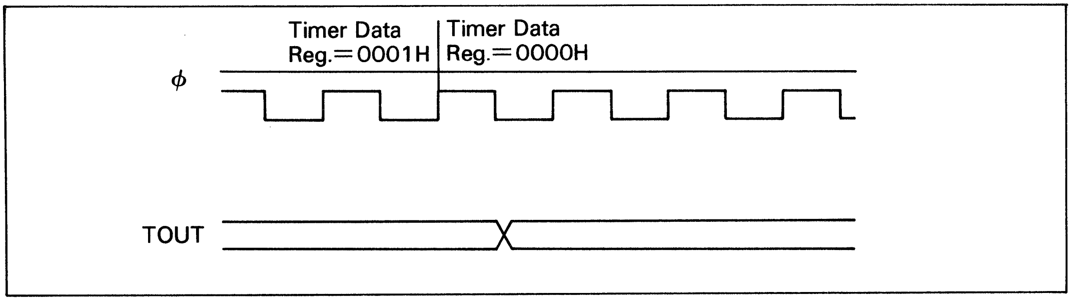


Figure 2.12.3 PRT Output Timing

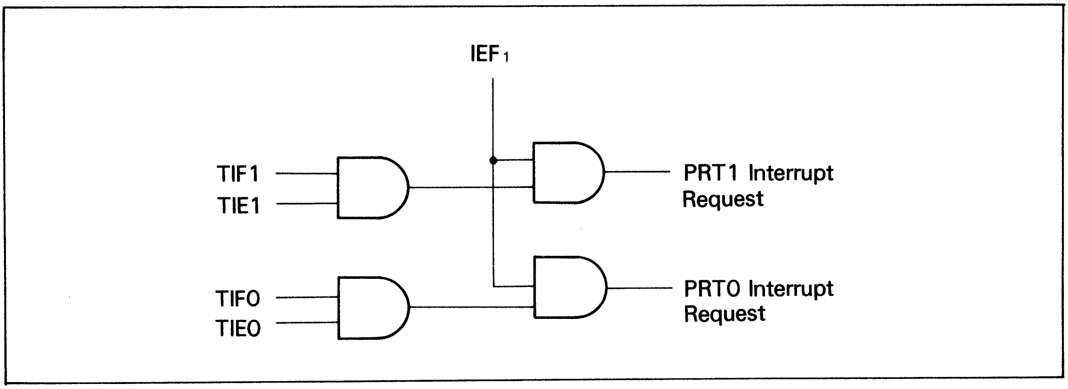


Figure 2.12.4 PRT Interrupt Request Circuit Diagram

2.12.3 PRT interrupts

The PRT interrupt request circuit is shown in Fig. 2.12.4.

2.12.4 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. The A_{18}/TOUT pin reverts to the address output function.

2.12.5 PRT operation notes

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

(2) Care should be taken to insure that a timer reload does not occur during or between lower (RLDRnL*) and higher (RLDRnH*) byte writes. This may be guaranteed by system design/timing or by stopping down counting (with TMDR containing a non-zero value) during the RLDR updating.
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, time base inaccuracy will occur.

NOTE: * n = 0, 1

(3) During RESET, the multiplexed A_{18}/TOUT pin reverts to the address output.
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 TOUT pin after TOC1 and TOC0 are programmed to select the PRT channel 1 timer output function.

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

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

		TOUT State After	TOUT State After
TOC1	TOC0	Programming TOC1/TOC0	Next Timeout
0	1	HIGH (1)	LOW (0)
1	0	HIGH (1)	LOW (0)
1	1	HIGH (1)	HIGH (1)

(ii) 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 TOUT depends on the number of time outs (even or odd) that have occurred.

Numbers of Timeouts	TOUT State After
(even or odd)	Programming TOC1/TOC0
Even (2, 4, 6 ...)	HIGH (1)
Odd (1, 3, 5 ...)	LOW (0)

2.13 6800 Type Bus Interface

2.13.1 E clock output timing

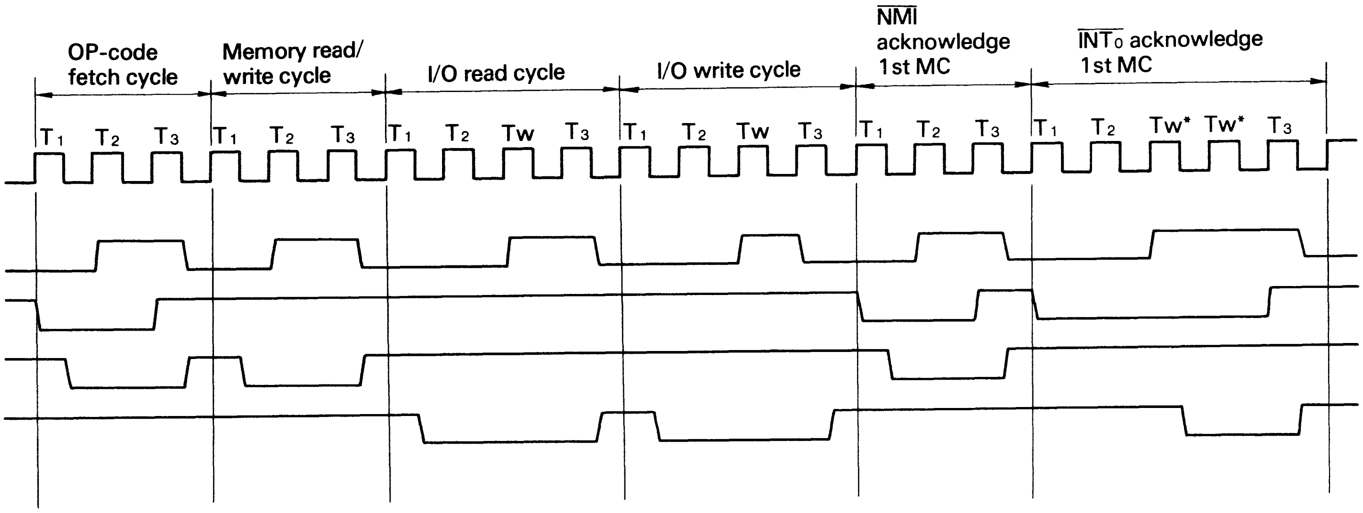
A large selection of 6800 type peripheral devices can be connected to the HD64180, including the Hitachi 6300 CMOS series (6321 PIA, 6350 ACIA, etc.) as well as 6800 family devices.

These devices require connection with the HD64180 synchronous E clock output. The speed (access time) required for the peripheral device are determined by the HD64180 clock rate. Table 2.13.1, Fig. 2.13.1 and Fig. 2.13.2 define E clock output timing.

Table 2.13.1 E Clock Timing in Each Condition

Condition	Duration of E Clock Output "High"	
Op-code Fetch Cycle Memory Read/Write Cycle	$T_2\uparrow - T_3\downarrow$	$(1.5\phi + n_W \cdot \phi)$
I/O read Cycle	1st $Tw\uparrow - T_3\downarrow$	$(0.5\phi + n_W \cdot \phi)$
I/O Write Cycle	1st $Tw\uparrow - T_3\uparrow$	$(n_W \cdot \phi)$
$\overline{\text{NMI}}$ Acknowledge 1st MC	$T_2\uparrow - T_3\downarrow$	(1.5ϕ)
$\overline{\text{INT}_0}$, $\overline{\text{INT}_1}$, $\overline{\text{INT}_2}$ and Internal Interrupt Acknowledge 1st MC	1st $Tw\uparrow - T_3\downarrow$	$(0.5\phi + n_W \cdot \phi)$
BUS RELEASE mode SLEEP mode SYSTEM STOP mode	$\phi\downarrow - \phi\downarrow$	$(2\phi$ or $1\phi)$

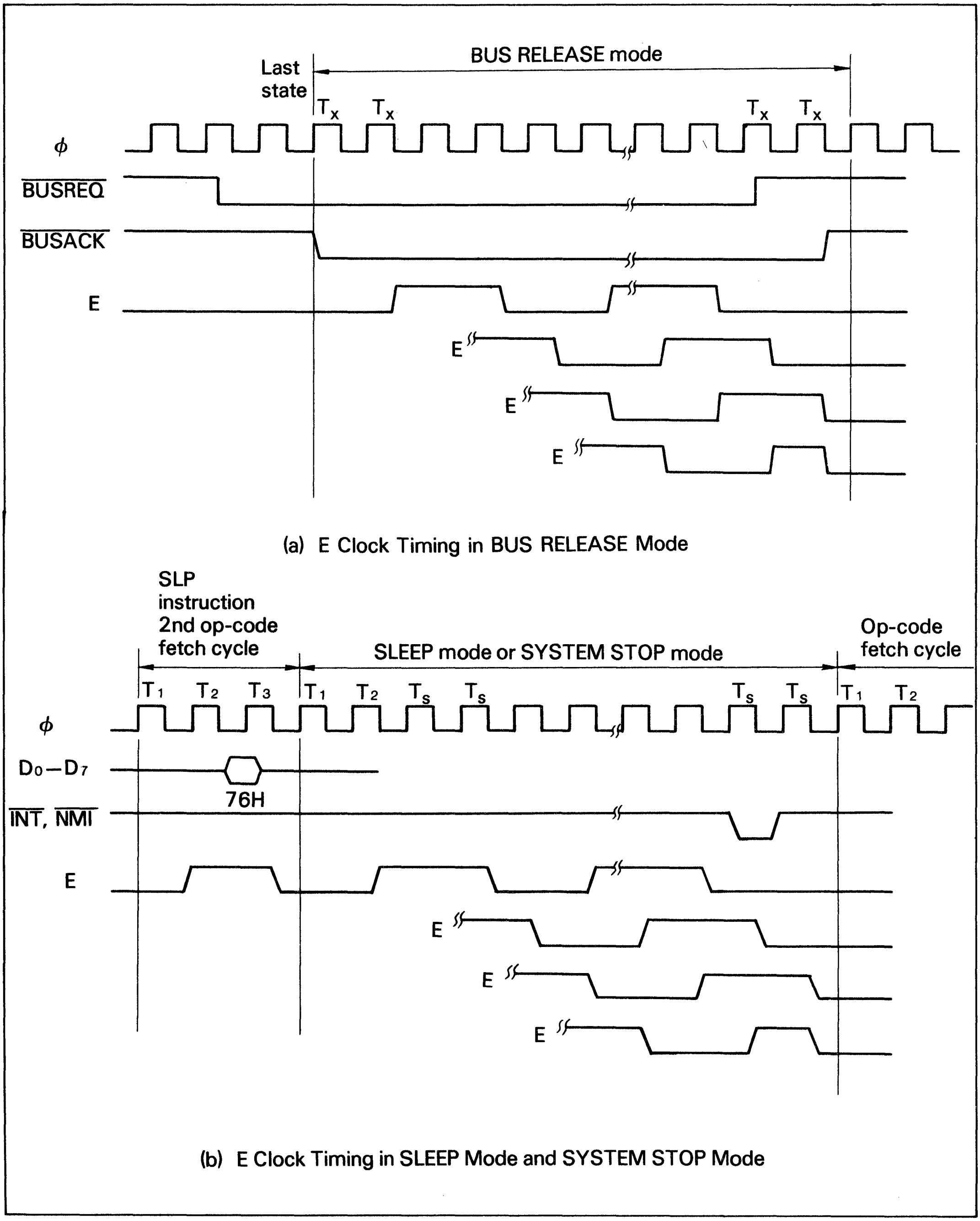
NOTE) n_W : the number of wait states
MC : Machine Cycle



NOTE) MC: Machine Cycle

* Two wait states are automatically inserted.

Figure 2.13.1 E Clock Timing (During Read/Write Cycle and Interrupt Acknowledge Cycle)



(a) E Clock Timing in BUS RELEASE Mode

(b) E Clock Timing in SLEEP Mode and SYSTEM STOP Mode

Figure 2.13.2 E Clock Timing (in BUS RELEASE mode, SLEEP mode, SYSTEM STOP mode)

Wait states inserted in op-code 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 as follows.

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.

2.13.2 6800 type bus interfacing note

When the HD64180 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.

2.14 On-chip Clock Generator

The HD64180 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.

The following table shows the AT cut crystal characteristics (Co, Rs) and the load capacitance (CL1, CL2) required for various frequencies of HD64180 operation.

Table 2.14.1 Crystal Characteristics

Clock Frequency / Item	4MHz	4MHz < f ≦ 12MHz	12MHz < f ≦ 16MHz
Co	<7 pF	<7 pF	<7 pF
Rs	<60Ω	<60Ω	<35Ω
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 $V_{CC}-0.6V$. The external clock input is connected to the EXTAL pin, while the XTAL pin is left open. Fig. 2.14.1 shows external clock interface.

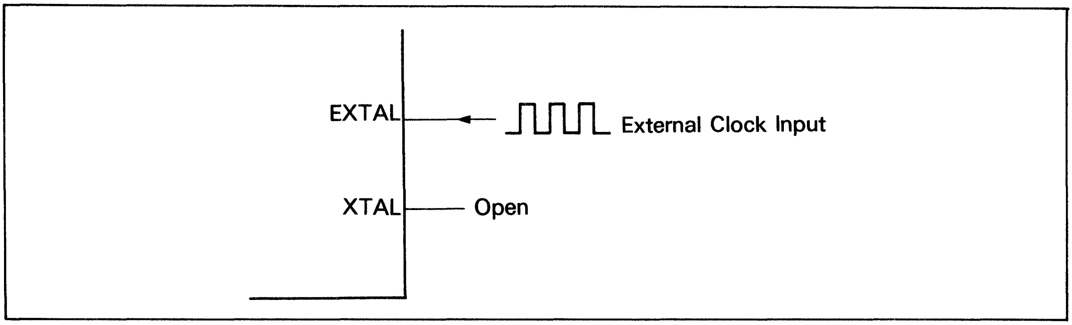


Figure 2.14.1 External Clock Interface

Fig. 2.14.2 shows the HD64180 clock generator circuit while Fig. 2.14.3 and Fig. 2.14.4 specify circuit board design rules.

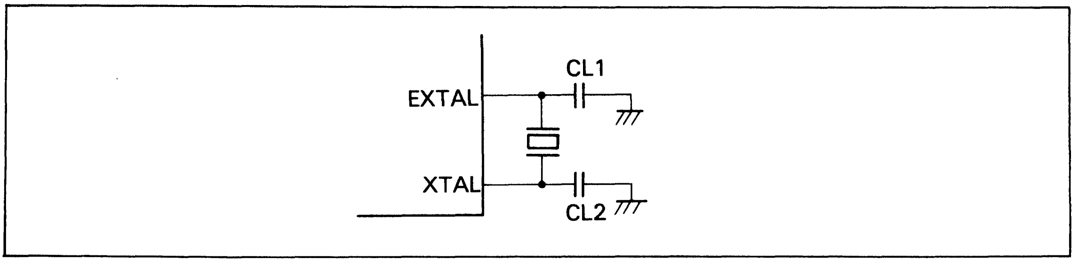


Figure 2.14.2 Crystal Interface

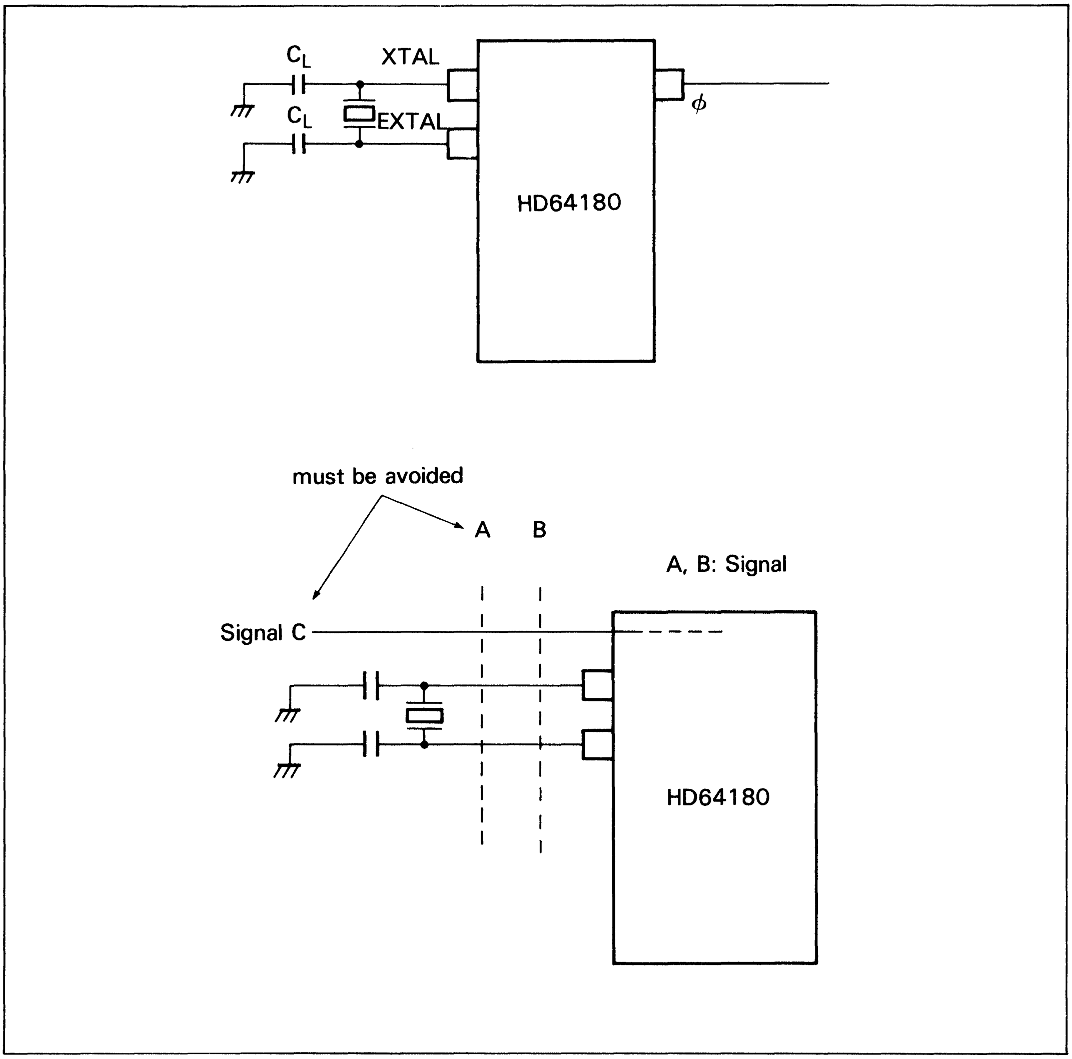


Figure 2.14.3 Note for Board Design of the Oscillation Circuit

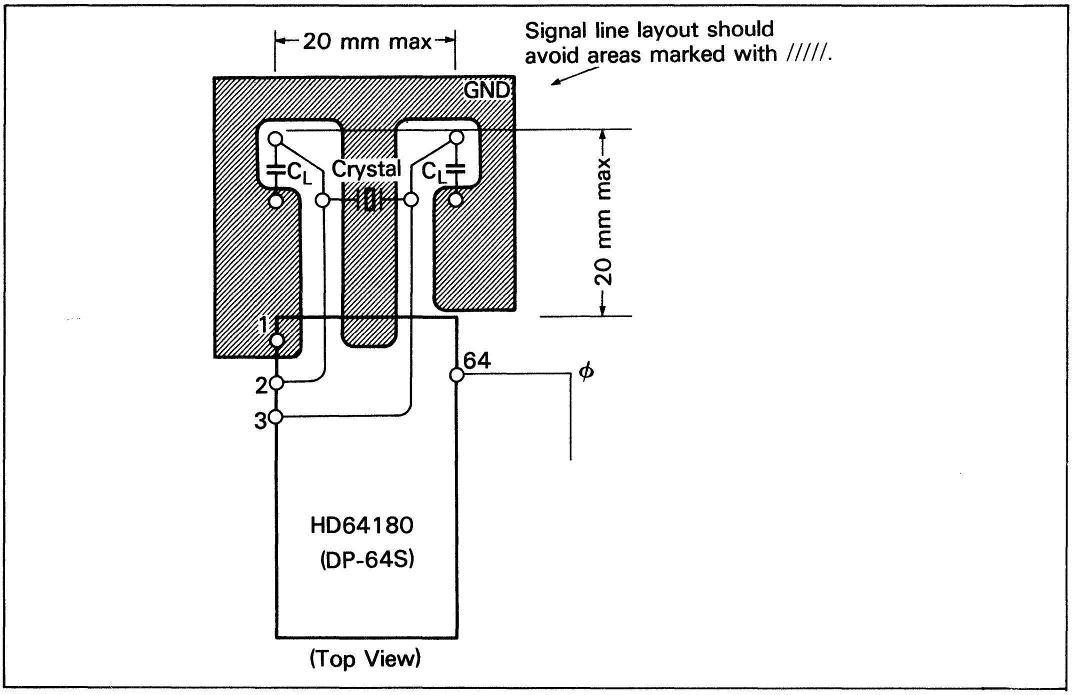


Figure 2.14.4 Example of Board Design

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) Similar to (2), 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 areas marked with /////.

2.15 Miscellaneous

Free Running Counter (I/O Address = 18H)

Read only 8-bit free running counter without control registers and status registers. The contents of the 8-bit free running counter is counted down by 1 with an interval of 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 baud rates of ASCI and CSI/O.

In IOSTOP mode, the free running counter continues counting down. It is initialized to FFH during RESET.

2.16 Operation Notes

2.16.1 Noise spike on TOUT (R0 MASK)

When A_{18}/TOUT pin functions as TOUT and outputs LOW, noise spike up to 2V may appear on the TOUT output at address outputs (A_0-A_{17}) changing from HIGH to LOW. This noise spike will become the largest, when all of the A_0-A_{17} address outputs change from HIGH to LOW simultaneously.

To avoid the noise spike on TOUT, a resistor and a capacitor should be connected to TOUT as shown in Fig. 2.16.2.

In HD64180R1 and HD64180Z, the noise spike is held to less than V_{OL}.

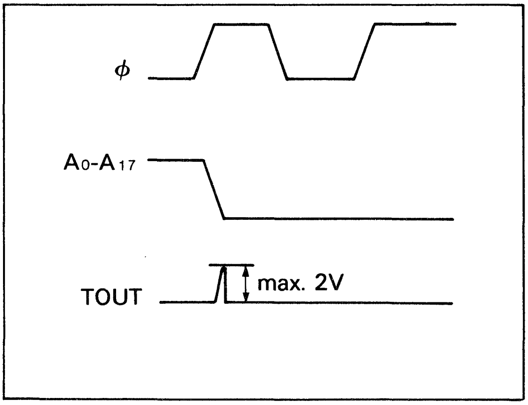


Figure 2.16.1 Noise Spike on TOUT

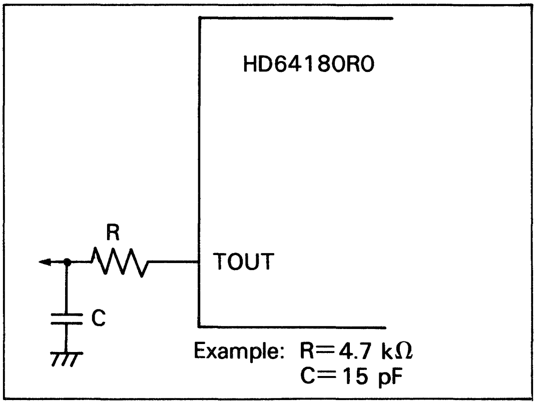


Figure 2.16.2 Resistor and Capacitor Connection

2.16.2 CKA$_0$/$\overline{\text{DREQ}_0}$ action in SLEEP mode (R0 MASK)

When CPU enters into SLEEP mode, CKA$_0$/$\overline{\text{DREQ}_0}$ functions as CKA$_0$. Therefore, if CKA$_0$/$\overline{\text{DREQ}_0}$ is used as input of DMA request signal, conflicts between DMA request signal and CKA$_0$ output may occur in SLEEP mode and cause overcurrent.

To avoid the overcurrent, a current limit resistor should be connected to CKA$_0$/$\overline{\text{DREQ}_0}$ as shown in Fig. 2.16.3.

In HD64180R1 and HD64180Z, the overcurrent problem has been solved.

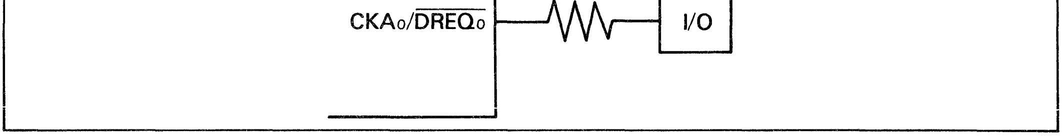


Figure 2.16.3 A Current Limit Resistor Connection

2.16.3 Precaution on interfacing the Z80 family peripheral LSIs to the HD64180 (R0, R1 MASK)

(1) Problem

In daisy chain, the Z80 family peripheral LSI (PIO, DMA, CTC, SIO, or DART) resets interrupt circuit (i.e. IEO changes from LOW to HIGH) by fetching the RETI op-code on the data bus concurrently during the CPU fetches the RETI. Therefore, the followings should be noted for the RETI op-code (EDH, 4DH) fetch timing in the Z80 peripheral LSI.

When the peripheral LSI fetches the first op-code of RETI (EDH), $\overline{\text{LIR}}$ should be asserted HIGH at the rising edge of system clock ϕ as shown in Fig. 2.16.4, A. (This isn't referred in the manuals for the Z80 peripheral LSI.) So, $\overline{\text{LIR}}$ hold time ($\overline{\text{LIR}}$ = HIGH) should be required as shown in Fig. 2.16.4.

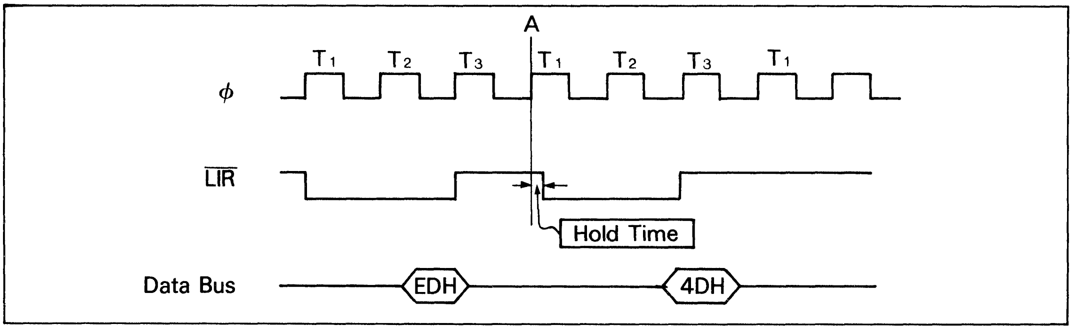


Figure 2.16.4 $\overline{\text{LIR}}$ Hold Time

Because $\overline{\text{LIR}}$ changes synchronously with the rising edge of system clock ϕ, $\overline{\text{LIR}}$ delay time is equal to $\overline{\text{LIR}}$ hold time of the Z80 peripheral LSI. However, this $\overline{\text{LIR}}$ hold time may not be sufficient for the Z80 peripheral LSI in some case and IEO line may not be reset.

(2) An example of countermeasure

Fig. 2.16.5 shows an example of circuit, while Fig. 2.16.6 shows the $\overline{\text{LIR}}$ and $\overline{\text{LIR}}$' timing in the circuit.

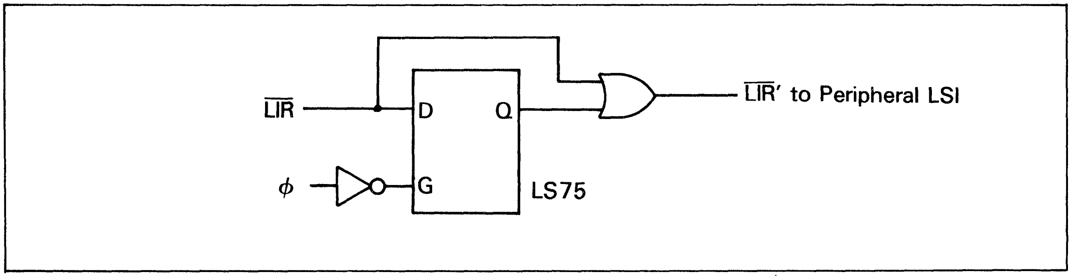


Figure 2.16.5 Circuit Example

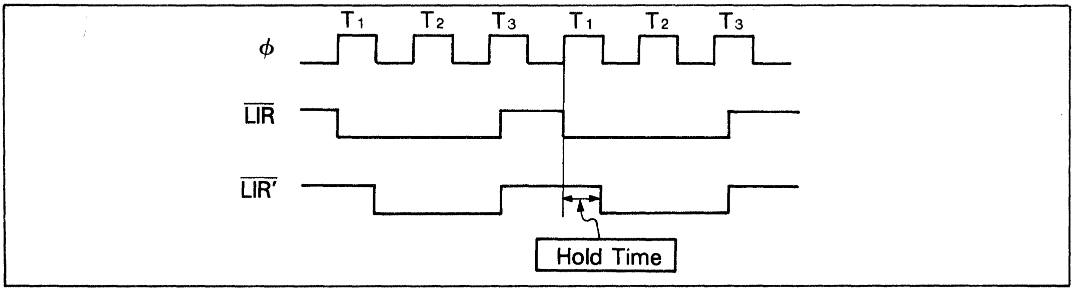


Figure 2.16.6 $\overline{\text{LIR}}$ and $\overline{\text{LIR}}'$ Timing in the Circuit

$\overline{\text{LIR}}$', which is synchronized with the falling edge of system clock ϕ, is provided to the peripheral LSI. In this case, one-half clock cycle duration is confirmed as the hold time.

Please carefully examine the circuit before you use it on your application.

2.16.4 Precautions on t_{AD} and t_{AS} (R0 MASK)

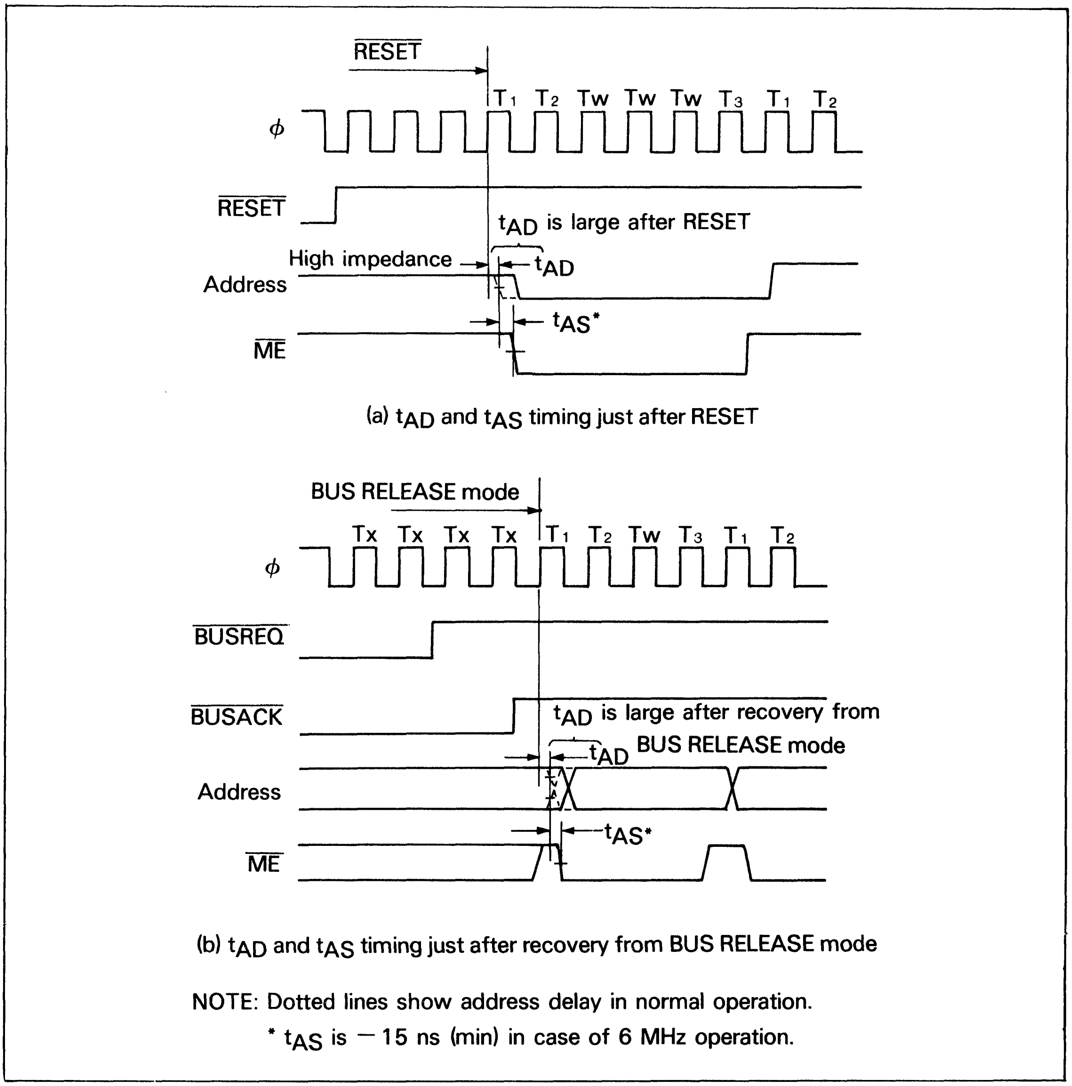
1. Specification of t_{AD} and t_{AS}

The specification of t_{AD} (Address delay time) and t_{AS} (Address set-up time) is shown in Table 2.16.1.

Table 2.16.1 Spec. of t_{AD} and t_{AS}

Symbol	Item		Spec.		Remarks
			HD64A180R0	HD64B180R0	
t_{AD}	Address Delay Time	Just after RESET, or after recovery from BUS RELEASE mode, or at the beginning of SLEEP mode and SYSTEM STOP mode	130 (ns) max	125 (ns) max	Refer to 2.(1)
		Normal operation	110 (ns) max	105 (ns) max	
t_{AS}	Address Set-up Time	Just after RESET, or after recovery from BUS RELEASE mode	30 (ns) min	− 15 (ns) min	Refer to 2.(2) (a)
		Normal operation	45 (ns) min	10 (ns) min	Refer to 2.(2) (b)

Fig. 2.16.7 show t_{AD} and t_{AS} timings just after RESET and just after recovery from BUS RELEASE mode.



(a) t_{AD} and t_{AS} timing just after RESET

(b) t_{AD} and t_{AS} timing just after recovery from BUS RELEASE mode

NOTE: Dotted lines show address delay in normal operation.

* t_{AS} is − 15 ns (min) in case of 6 MHz operation.

Figure 2.16.7 t_{AD} and t_{AS} Timings

2. Problems and countermeasures on t_{AD} and t_{AS}

Due to the above specification of t_{AD} and t_{AS}, the following problems may occur.

(1) t_{AD}

Just after recovery from BUS RELEASE mode, t_{AD} is larger than usual by 20 ns. Therefore, if Memory or I/O LSI access timing is designed based on the old t_{AD}, an access time may not be enough just after recovery from BUS RELEASE mode.

So, one of the following two ways should be taken to design memory or I/O LSI access timing.

(i) All memory or I/O LSI access timing should be designed based on t_{AD} just after recovery from BUS RELEASE mode.

(ii) Wait state (Tw) should be inserted just after recovery from BUS RELEASE mode. (Refer to Example 3.)

(2) t_{AS}

(a) In case of 6 MHz operation, t_{AS} is −15 ns (min) just after RESET or just after recovery from BUS RELEASE mode. Therefore, if it is necessary to assure address set-up time of 0 ns or more than 0 ns (min) for a falling edge of $\overline{ME}$ or $\overline{IOE}$, $\overline{ME}$ or $\overline{IOE}$ can't be directry used. Generally speaking, the problem may occur in the following cases.

(i) DRAM access using $\overline{ME}$ as $\overline{RAS}$ signal

(ii) Pseudo SRAM access using $\overline{ME}$ as $\overline{CE}$ signal

DRAM or Pseudo SRAM requires 0 ns (min) of address set-up for a falling edge of $\overline{RAS}$ or $\overline{CE}$. (Refer to Fig. 2.16.8 and Fig. 2.16.9.)

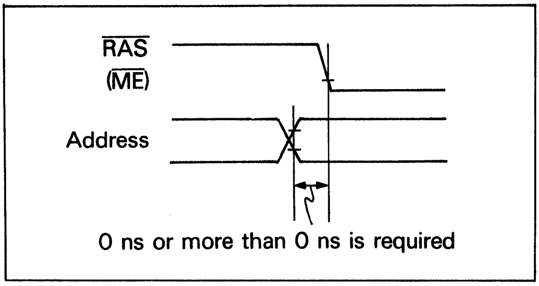


Figure 2.16.8 t_{AS} Timing for DRAM

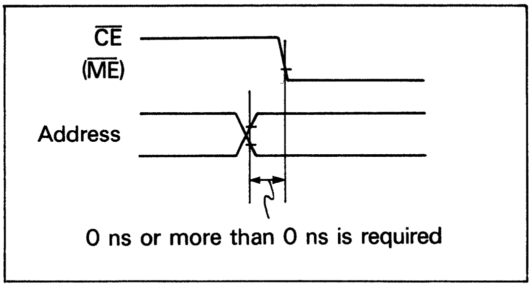


Figure 2.16.9 t_{AS} Timing for Pseudo SRAM

Therefore, if DRAM or Pseudo SRAM is accessed just after RESET or after recovery from BUS RELEASE mode, $\overline{ME}$ should be delayed by half cycle of system clock ϕ. (Refer to Example 2.)

In the case of PROM, fully static RAM or a peripheral LSI which doesn't require set-up time, $\overline{CS}$ signal may be unstable at the beginning of access when $\overline{CS}$ is synchronized with $\overline{ME}$ or $\overline{IOE}$. However, there is no problem by the following reason;

Read — If access time is assured enough, data is read correctly.
Write — As an address set-up time for a falling edge of $\overline{\text{WR}}$ is assured, unintentional writing into a memory or a peripheral LSI can not occur.

Please carefully examine the spec. of a memory or a peripheral LSI for your application.

(b) In case of 6 MHz operation, address set-up time of normal operation is 10 ns (min). If this address set-up time is not enough, $\overline{\text{ME}}$ should be delayed to assure address set-up time. (Refer to Example 2.)

Please confirm whether the problems mentioned above occur or not, refering to the flowchart in Fig. 2.16.10.

If you have some problem, please take some countermeasure as shown in Examples 1 to 3.

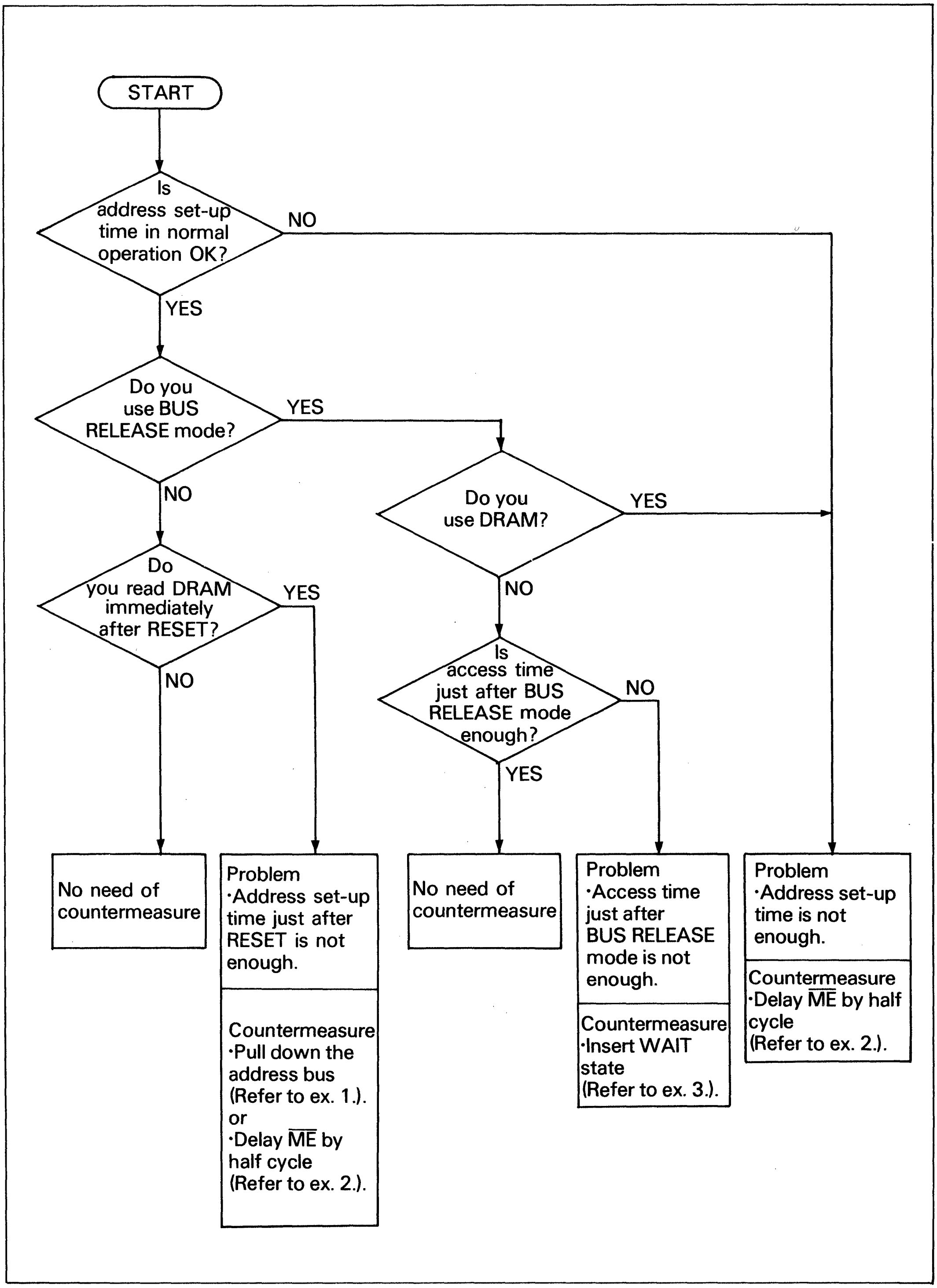

START
Is address set-up time in normal operation OK?
NO
YES
Do you use BUS RELEASE mode?
YES
NO
Do you use DRAM?
YES
NO
Do you read DRAM immediately after RESET?
YES
NO
Is access time just after BUS RELEASE mode enough?
NO
YES
No need of countermeasure
Problem
·Address set-up time just after RESET is not enough.
Countermeasure
·Pull down the address bus (Refer to ex. 1.).
or
·Delay $\overline{\text{ME}}$ by half cycle (Refer to ex. 2.).
No need of countermeasure
Problem
·Access time just after BUS RELEASE mode is not enough.
Countermeasure
·Insert WAIT state (Refer to ex. 3.).
Problem
·Address set-up time is not enough.
Countermeasure
·Delay $\overline{\text{ME}}$ by half cycle (Refer to ex. 2.).

Figure 2.16.10 Check Flow

The followings show countermeasures to assure t_{AS} just after RESET or after recovery from BUS RELEASE mode.

Example 1: Countermeasure for t_{AS} just after RESET

As Restart address is 00000H, t_{AS} can be assured by pulling down all address lines to "0" during RESET. Fig. 2.16.11 shows the circuit for countermeasure.

Address lines become high impedance during RESET. However, if pull-down resistor R are connected as shown in Fig. 2.16.11, address lines go "0". In this case, length of RESET cycle T should be much longer than C·R. (T ≫ C·R)

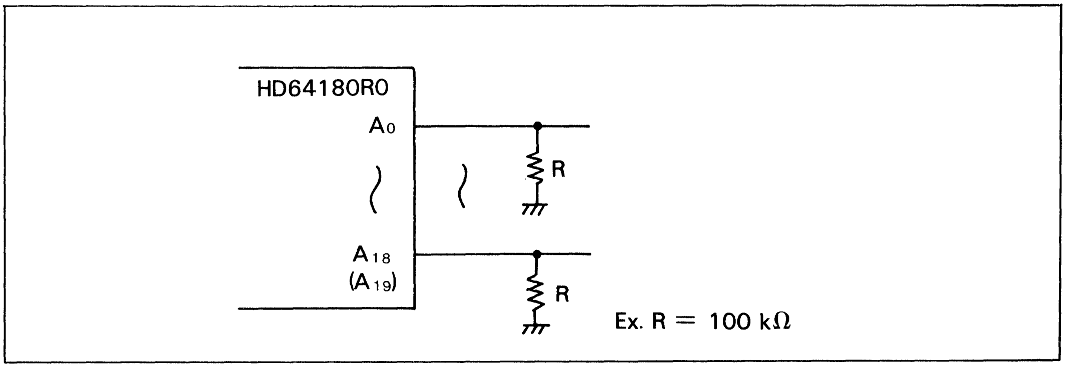
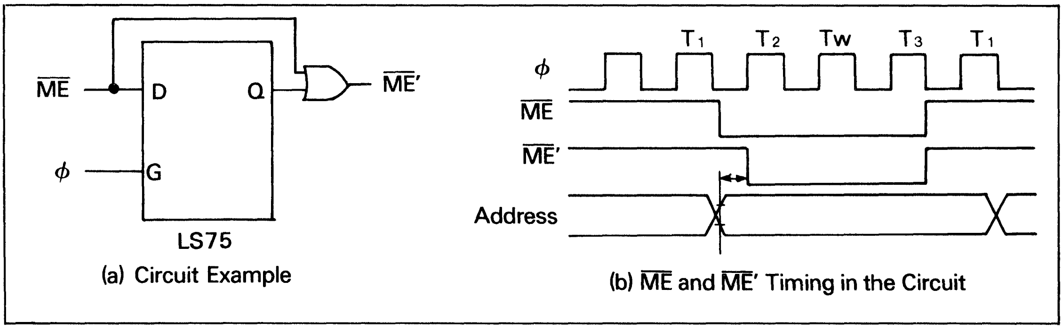


Figure 2.16.11 Countermeasure for t_{AS} Just After RESET

Example 2: Countermeasure for t_{AS}

To assure t_{AS}, $\overline{ME}$ should be delayed by half cycle of system clock ϕ by connecting external circuit as shown in Fig. 2.16.12, (a). $\overline{ME}$ and $\overline{ME}$' timing in the circuit is shown in Fig. 2.16.12, (b).



(a) Circuit Example

(b) $\overline{ME}$ and $\overline{ME}$' Timing in the Circuit

Figure 2.16.12 Countermeasure for t_{AS} by Delaying $\overline{ME}$

Example 3: Countermeasure for t_{AD} just after recovering from BUS RELEASE mode

t_{AD} just after recovery from BUS RELEASE mode can be assured by inserting wait state (Tw). To insert wait state (Tw), an external circuit should be connected as shown in Fig. 2.16.13, (a). Timing in the circuit is shown in Fig. 2.16.13, (b).

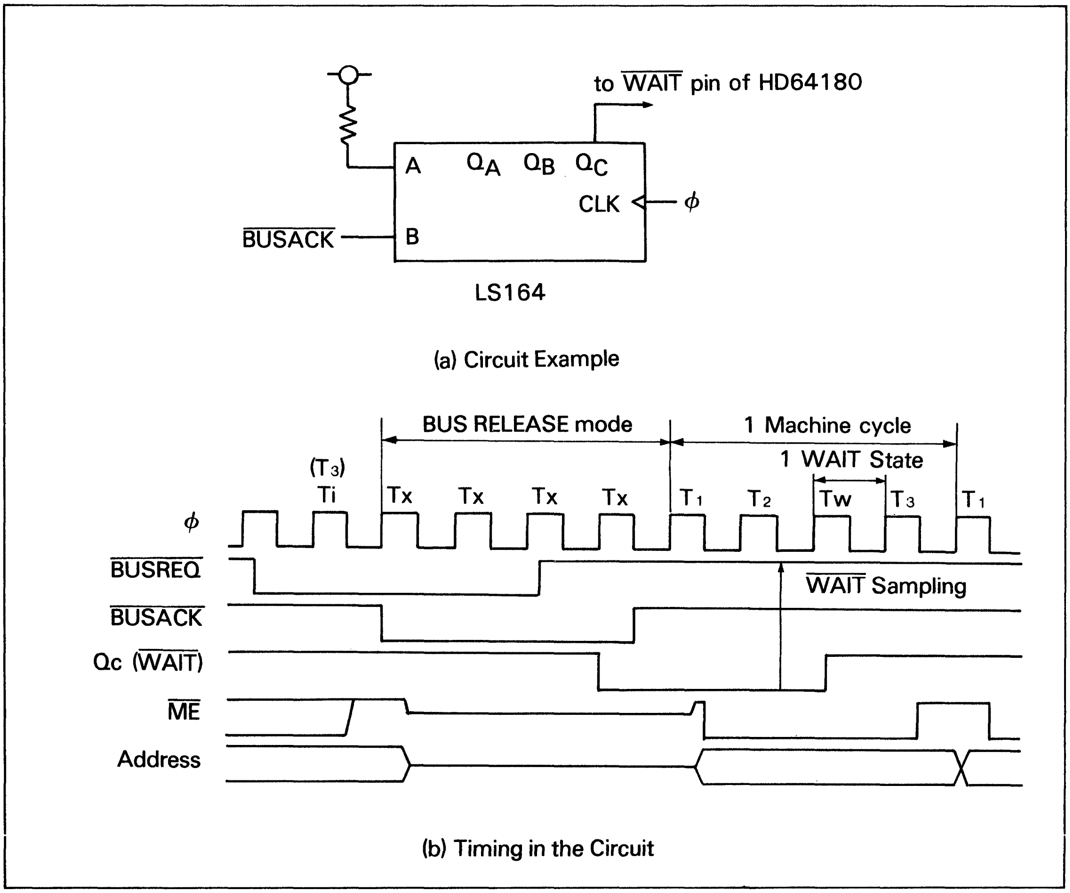


Figure 2.16.13 Countermeasure for t_{AD} Just After Recovery from BUS RELEASE Mode

2.16.5 Precaution on interfacing HD64180 with Z80 CTC (R0, R1 Mask)

1. Problem

The following problem may happen when interfacing HD64180 with Z80 CTC (Z8430). Therefore, countermeasure shown in section 2 should be taken. Fig. 2.16.14 illustrates Z80 CTC write timing specified in Z80 CTC Data Sheet. While Fig. 2.16.15 and Fig. 2.16.16 show Z80 I/O write timing and HD64180 I/O write timing respectively.

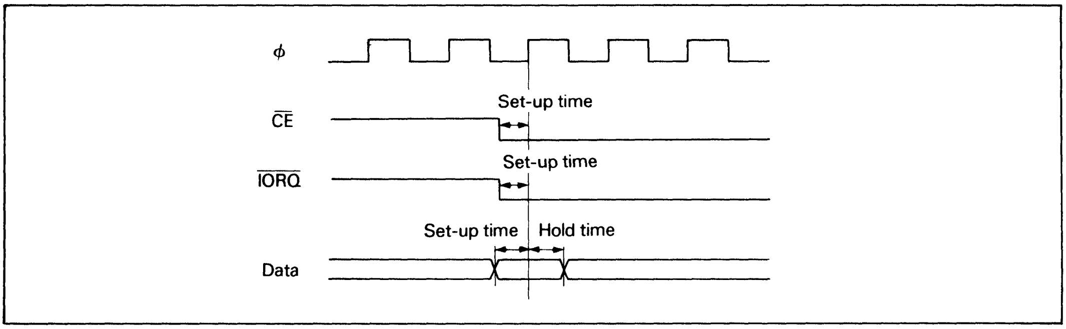


Figure 2.16.14 Z80 CTC Write Timing*

* copied from Z80 CTC Data Sheet (April, 1985)

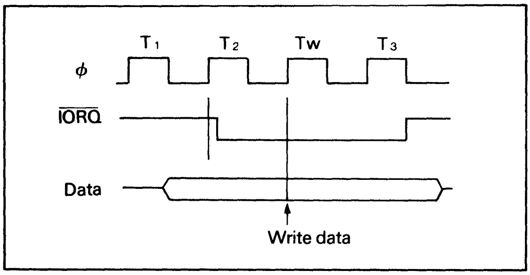


Figure 2.16.15 Z80 I/O Write Timing

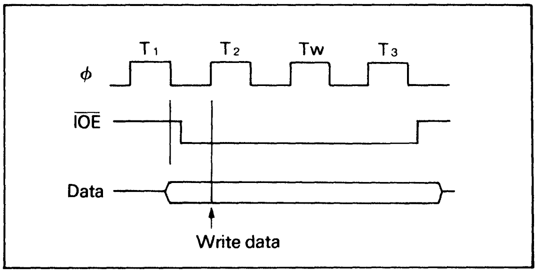


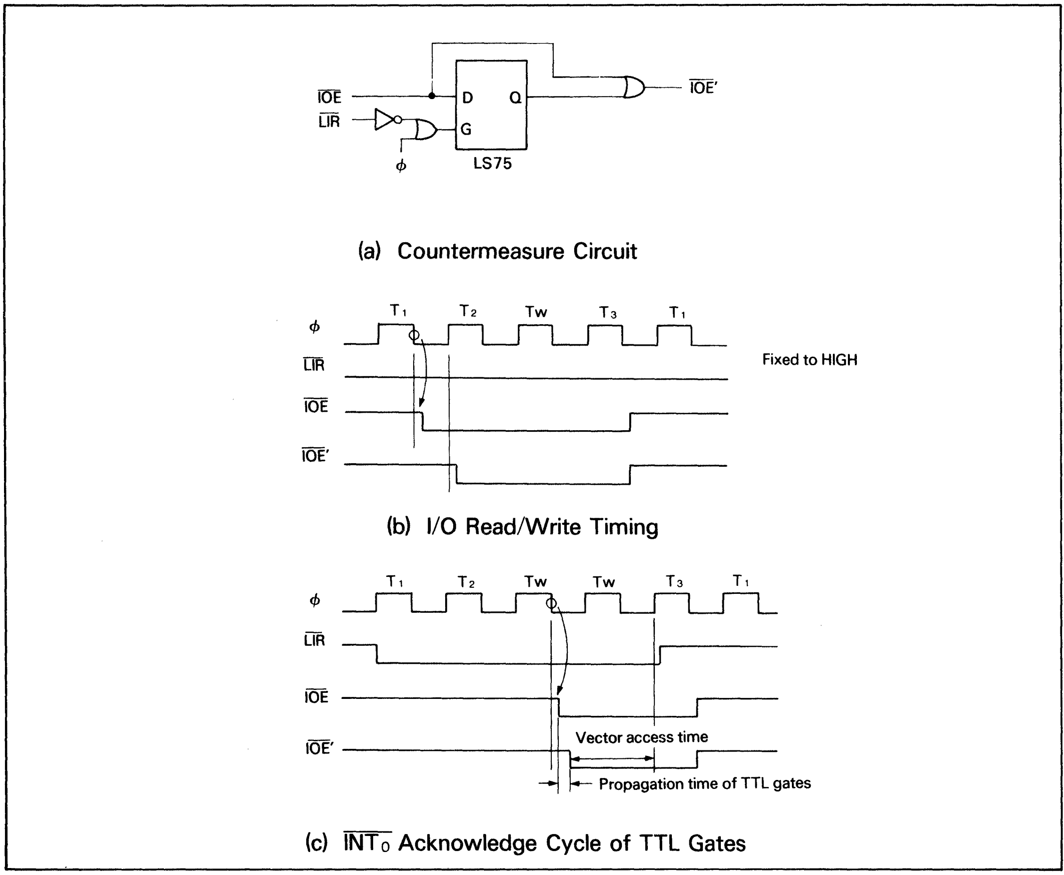
Figure 2.16.16 HD64180 I/O Write Timing

As shown above, $\overline{IOE}$ in HD64180 goes LOW by a half ϕ clock cycle faster than $\overline{IORQ}$ in Z80. When interfacing Z80 with Z80 CTC, data is written into Z80 CTC at the rising edge of Tw. While, when interfacing HD64180 with Z80 CTC, data is written into Z80 CTC at the rising edge of T2. In the latter case, data may not be written into Z80 CTC if $\overline{IOE}$ set-up time for the rising of T2 is less than the set-up time specified in Z80 CTC.

2. Countermeasure

To avoid the problem, $\overline{IOE}$ in HD64180 should be asserted LOW at the rising edge of T2 to assure the set-up time specified in Z80 CTC. Fig. 2.16.17 (a) shows a circuit for delaying $\overline{IOE}$ by a half ϕ clock cycle.

If this circuit is externally connected between HD64180 and Z80 CTC, $\overline{IOE'}$ will be pulled LOW at the rising edge of T2 only in I/O read/write cycle as shown in Fig. 2.16.17 (b). While in $\overline{INT_0}$ acknowledge cycle, $\overline{IOE}$ and $\overline{IOE'}$ are asserted LOW at the timing shown in Fig. 2.16.17 (c). In $\overline{INT_0}$ acknowledge cycle, $\overline{IOE'}$ delays because of propagation time of TTL gates of the countermeasure circuit and the vector access time is shortened. If vector access time for HD64180 is not assured during $\overline{\text{INT}_0}$ acknowledge cycle, wait states should be inserted by programming IWI0 and IWI1 bits of DMA/WAIT Control Register. However, note that wait states insertion by software should be inhibited during Z80 CTC read/write cycles, because more than one wait state can not be allowed in the case of Z80 CTC. (Please see Z80 CTC Data Sheet. One wait state is automatically inserted during the cycles.) Refer to Fig. 2.16.18 "Z80 CTC Access Flow" for details.

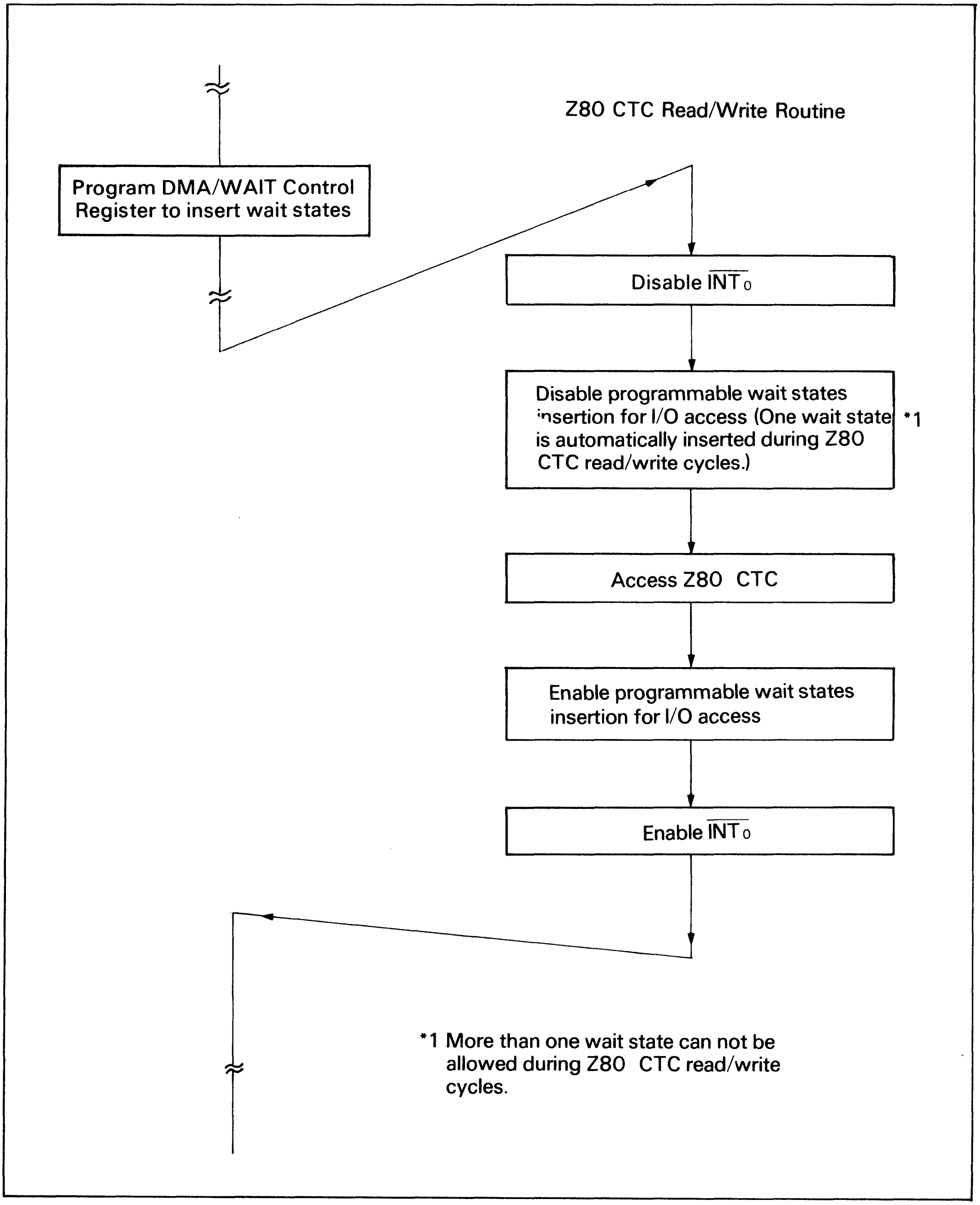


(a) Countermeasure Circuit

(b) I/O Read/Write Timing

(c) $\overline{\text{INT}_0}$ Acknowledge Cycle of TTL Gates

Figure 2.16.17 Countermeasure Circuit and Timings in the Circuit


Z80 CTC Read/Write Routine
Program DMA/WAIT Control Register to insert wait states
Disable $\overline{\text{INT}}_0$
Disable programmable wait states ⁱnsertion for I/O access (One wait state is automatically inserted during Z80 CTC read/write cycles.) *1
Access Z80 CTC
Enable programmable wait states insertion for I/O access
Enable $\overline{\text{INT}}_0$

*1 More than one wait state can not be allowed during Z80 CTC read/write cycles.

Figure 2.16.18 Z80 CTC Access Flow

2.16.6 Notes on HD64180 $\overline{\text{INT}_0}$ Mode 0 (All Masks)

1. Problem

In $\overline{\text{INT}_0}$ Mode 0, the CPU executes an instruction which is placed on the data bus during the interrupt acknowledge cycle. Usually, 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. In the case of RST instruction, the correct return address is pushed onto the stack. However, in the case of CALL instruction, the pushed return address is equal to the correct return address + 2.

2. Explanation of operation

During the 1st op-code fetch cycle in the interrupt acknowledge cycle, the CPU stops incrementing the PC. At this time, the PC contains the return address. After the 1st op-code is fetched, the CPU restarts incrementing the PC. Therefore, if 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 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 1st op-code 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.

Fig. 2.16.19 shows the CALL execution timing in $\overline{\text{INT}_0}$ Mode 0.

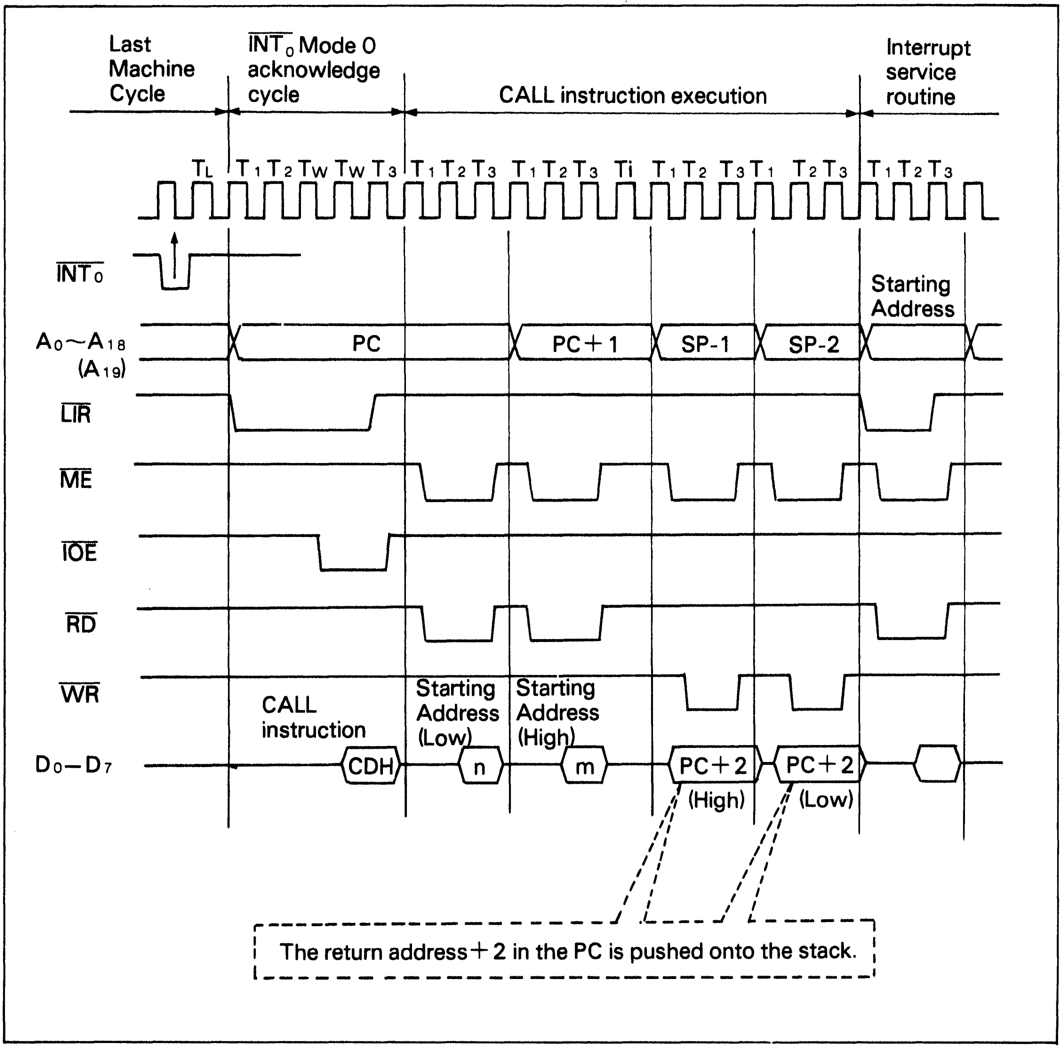


Figure 2.16.19 The CALL Execution Timing in $\overline{INT_0}$ Mode 0

3. Countermeasure

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

(1) RST

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

(2) CALL

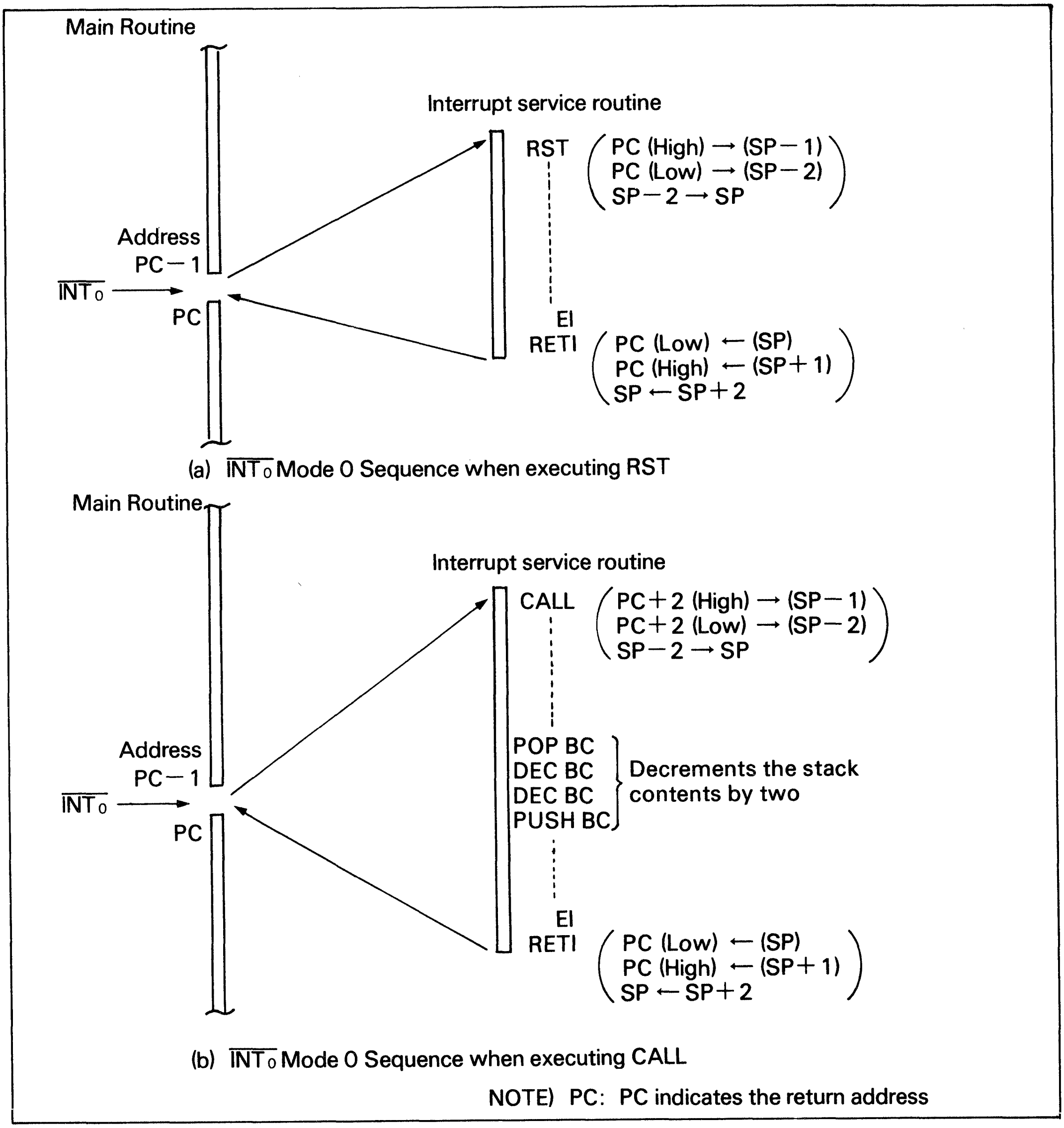
When CALL is executed, the stack contents must be decremented by two in the interrupt service routine to return from the interrupt correctly.

Table 2.16.2 summarizes how to adjust the stack contents depending on the instruction to be executed.

Table 2.16.2 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 $\overline{INT_0}$ Mode 0 sequences when executing RST and CALL are shown in Fig. 2.16.20.



(a) $\overline{INT_0}$ Mode 0 Sequence when executing RST

(b) $\overline{INT_0}$ Mode 0 Sequence when executing CALL

NOTE) PC: PC indicates the return address

Figure 2.16.20 $\overline{INT_0}$ Mode 0 Sequence

2.16.7 Note on reset operation when the CPU is in a WAIT state (R0 Mask)

1. Problem

When the $\overline{\text{WAIT}}$ signal is asserted low and the CPU is in a WAIT state, the CPU can't be reset and can't restart even if the $\overline{\text{RESET}}$ signal is asserted low.

(1) Incorrect operation

In the case that $\overline{\text{RESET}}$ signal goes to low level and then goes back to high level while the $\overline{\text{WAIT}}$ signal is asserted low, the CPU can't restart. (Please see Fig. 2.16.21)

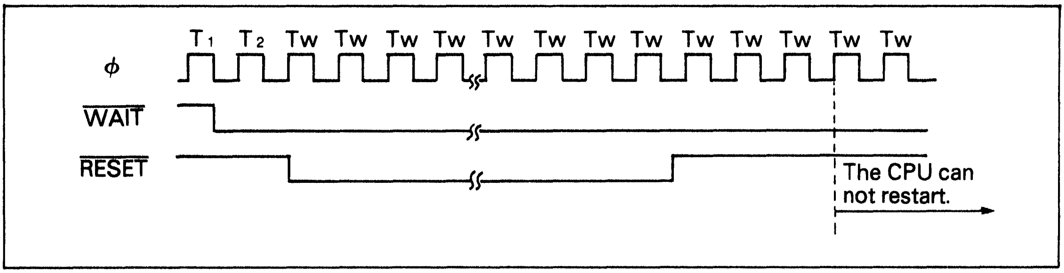


Figure 2.16.21 Incorrect Operation

(2) Correct operation

In the following cases, the CPU can restart correctly.

(a) In the case that the $\overline{\text{RESET}}$ signal goes to low level by 5 clocks earlier than the $\overline{\text{WAIT}}$ goes to low level.

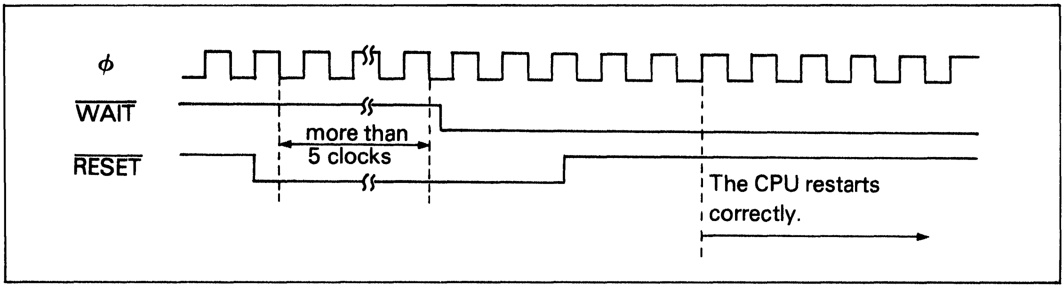


Figure 2.16.22(a) Correct Operation

(b) In the case that the $\overline{\text{WAIT}}$ signal goes to high level by 3 clocks earlier than the $\overline{\text{RESET}}$ signal goes to high.

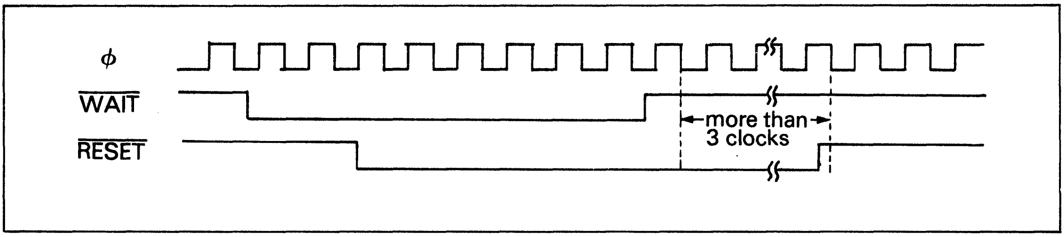


Figure 2.16.22(b) Correct Operation

(c) In the case that the $\overline{\text{RESET}}$ goes to low while the CPU is in a WAIT state caused by the internal programmable WAIT state generator and while the $\overline{\text{WAIT}}$ signal is kept in high level.

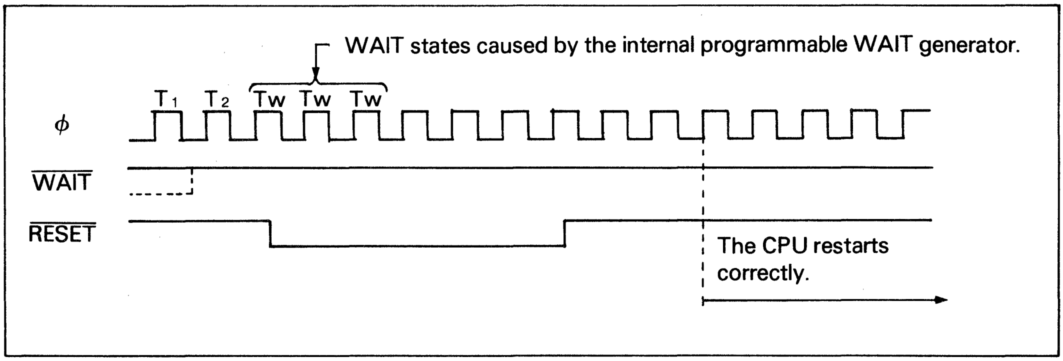

WAIT states caused by the internal programmable WAIT generator.
T1
T2
Tw
Tw
Tw
φ
$\overline{\text{WAIT}}$
$\overline{\text{RESET}}$
The CPU restarts correctly.

Figure 2.16.22(c) Correct Operation

2. Countermeasure

Please force the WAIT signal to change to high level by 3 clocks before the $\overline{\text{RESET}}$ signal goes to high level. Please see Fig. 2.16.23 and 2.16.24.

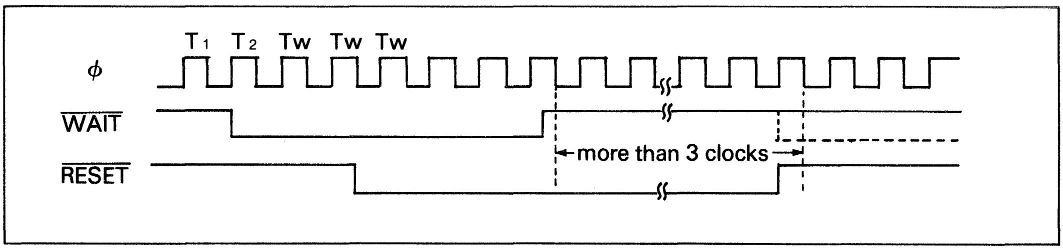


Figure 2.16.23 Timing

(1) An example of countermeasure circuits

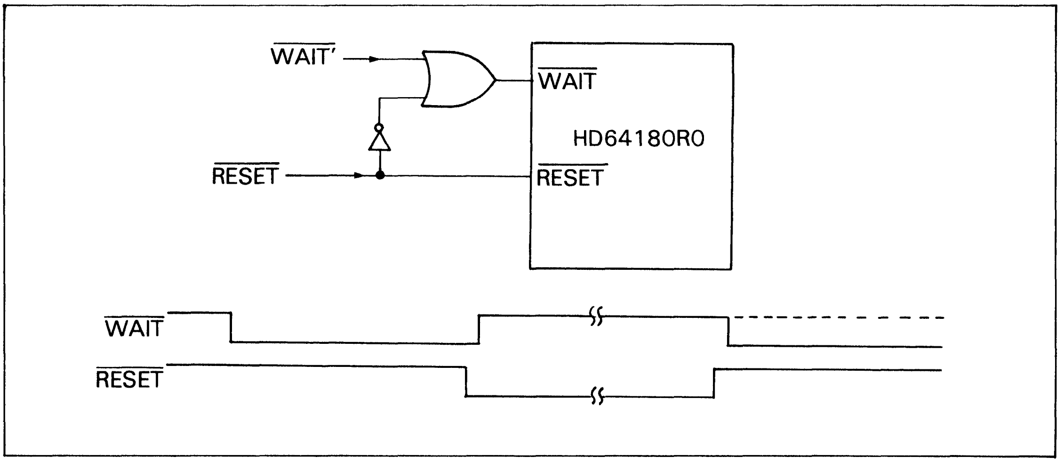


Figure 2.16.24 An Example of Circuits and the Timing

2.16.8 Note on conflict of $\overline{\text{NMI}}$ and TRAP interrupts (R0 Mask))

1. Problem

If the CPU reads an undefined op-code normally, the TRAP interrupt occurs.

If the CPU receives a $\overline{\text{NMI}}$ request during fetching an undefined op-code, the CPU jumps to 0066H, that is $\overline{\text{NMI}}$'s jump address, not to the normal TRAP address (0000H). Therefore, in this case, TRAP interrupt routine is not executed.

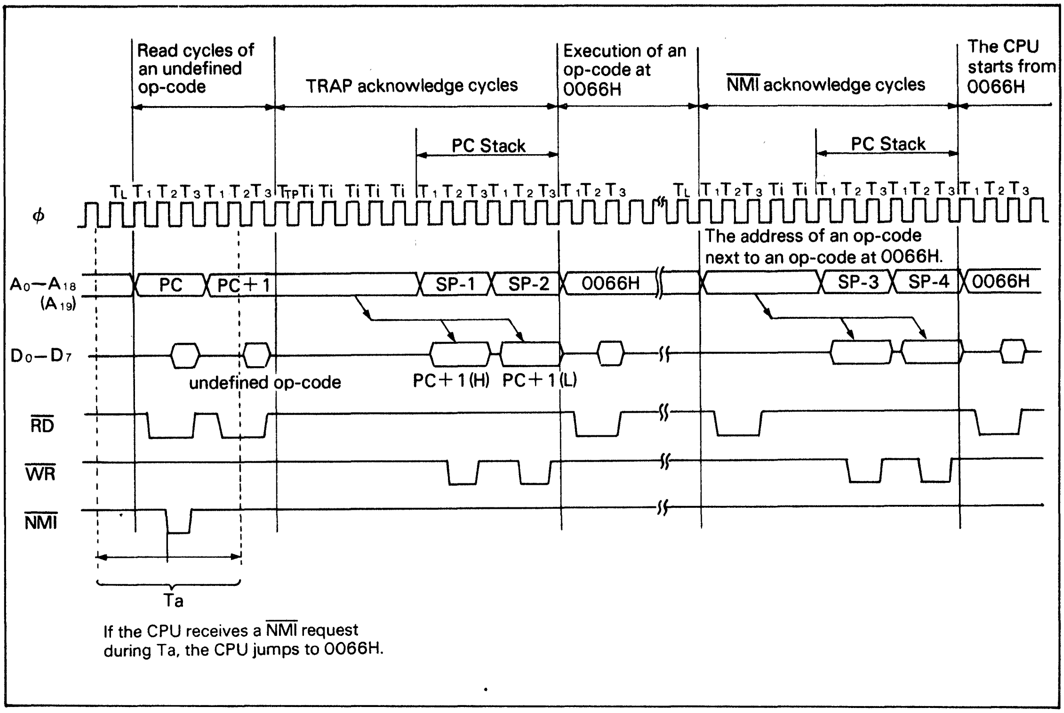
2. Explanation

Fig. 2.16.25 shows an operation timing when the CPU receives a $\overline{\text{NMI}}$ interrupt during fetching an undefined op-code. In this example, the second op-code is assumed to be undefined one. When a falling transition of $\overline{\text{NMI}}$ signal happens during the period of Ta which is described in Fig. 2.16.25, the TRAP interrupt occurs internally and the CPU starts TRAP interrupt acknowledge cycles. In the acknowledge cycles, the CPU stacks the address of an undefined op-code, and then the CPU jumps to logical address 0066H, not to logical address 0000H. As the request of $\overline{\text{NMI}}$ interrupt is still suspended at the time, the CPU starts the $\overline{\text{NMI}}$ acknowledge cycles after executing an op-code at logical address 0066H.

In the $\overline{\text{NMI}}$ acknowledge cycles, the CPU jumps to logical address 0066H again after stacking the address of an op-code next to the op-code at logical address 0066H. Then, the CPU executes normally.

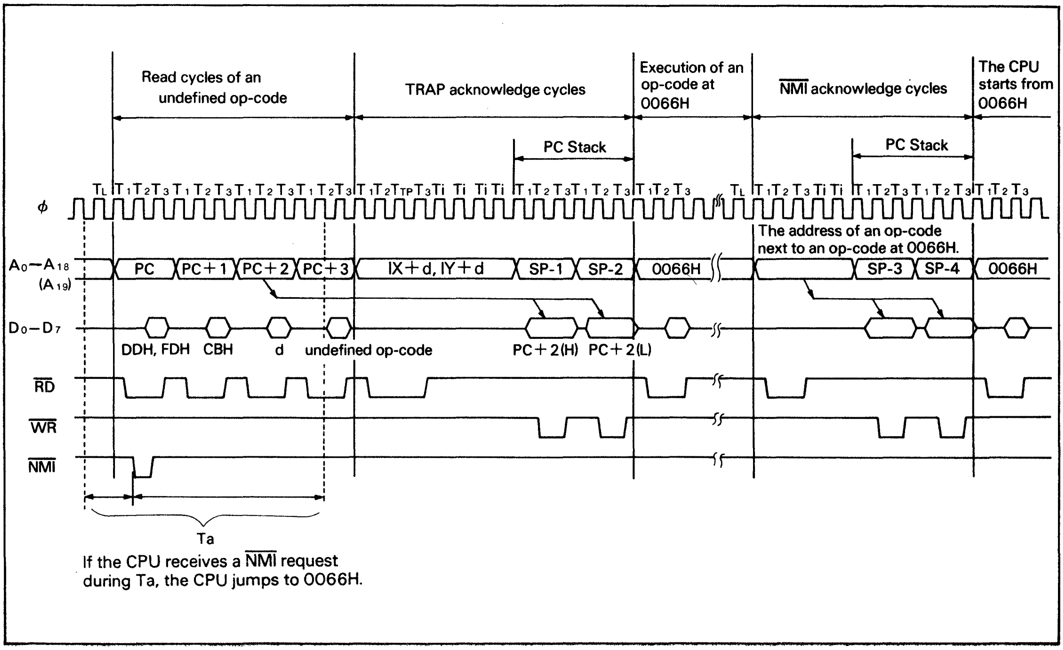
Fig. 2.16.26 shows an operation timing in the case that the third op-code is undefined. The CPU also acts similarly.

As mentioned above, if $\overline{\text{NMI}}$ interrupt is requested while the CPU reads an undefined op-code, the CPU can not start from logical address 0000H because the start address changes to logical address 0066H.


Read cycles of an undefined op-code
TRAP acknowledge cycles
Execution of an op-code at 0066H
NMI acknowledge cycles
The CPU starts from 0066H
PC Stack
PC Stack
TL T1 T2 T3 T1 T2 T3 TTP Ti Ti Ti Ti Ti T1 T2 T3 T1 T2 T3 T1 T2 T3 TL T1 T2 T3 Ti Ti T1 T2 T3 T1 T2 T3 T1 T2 T3
φ
The address of an op-code next to an op-code at 0066H.
A0—A18 (A19)
PC
PC+1
SP-1
SP-2
0066H
SP-3
SP-4
0066H
D0—D7
undefined op-code
PC+1(H) PC+1(L)
RD
WR
NMI
Ta

If the CPU receives a NMI request during Ta, the CPU jumps to 0066H.

Figure 2.16.25 Operation Timing in the Case that the Second Op-code is Undefined.


Read cycles of an undefined op-code
TRAP acknowledge cycles
Execution of an op-code at 0066H
NMI acknowledge cycles
The CPU starts from 0066H
PC Stack
PC Stack
TL T1 T2 T3 T1 T2 T3 T1 T2 T3 T1 T2 T3 T1 T2 TTP T3 Ti Ti Ti Ti T1 T2 T3 T1 T2 T3 T1 T2 T3 TL T1 T2 T3 Ti Ti T1 T2 T3 T1 T2 T3 T1 T2 T3
φ
The address of an op-code next to an op-code at 0066H.
A0—A18 (A19)
PC
PC+1
PC+2
PC+3
IX+d, IY+d
SP-1
SP-2
0066H
SP-3
SP-4
0066H
D0—D7
DDH, FDH
CBH
d
undefined op-code
PC+2(H) PC+2(L)
RD
WR
NMI
Ta

If the CPU receives a NMI request during Ta, the CPU jumps to 0066H.

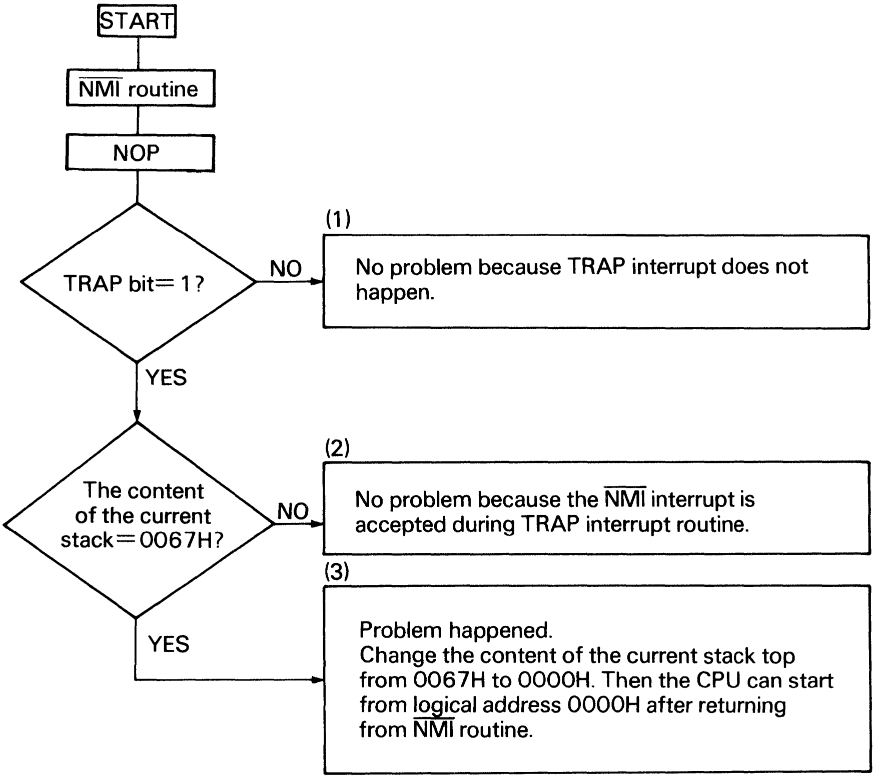
Figure 2.16.26 Operation Timing in the Case that the Third Op-code is Undefined.

3. Countermeasure

Please take countermeasure by software.

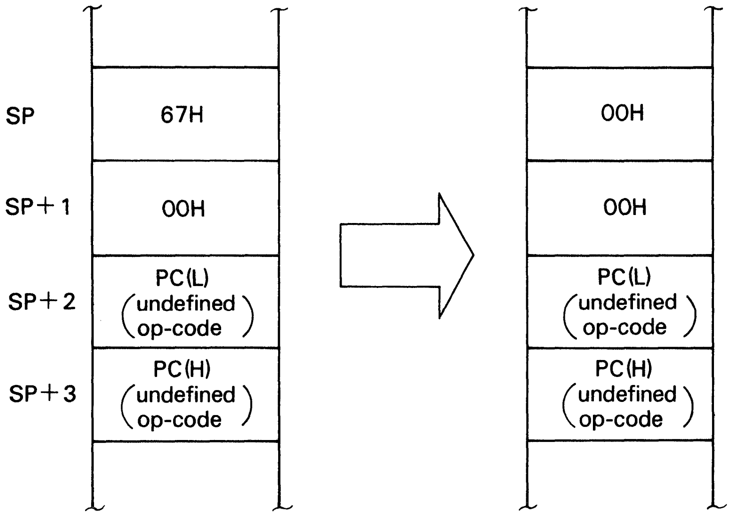
(1) Change an op-code at logical address 0066H to NOP.

(2) Execute the following at the beginning of $\overline{\text{NMI}}$ interrupt routine.


START
NMI routine
NOP
TRAP bit= 1 ?
NO
(1)
No problem because TRAP interrupt does not happen.
YES
The content of the current stack=0067H?
NO
(2)
No problem because the NMI interrupt is accepted during TRAP interrupt routine.
YES
(3)
Problem happened.
Change the content of the current stack top from 0067H to 0000H. Then the CPU can start from logical address 0000H after returning from NMI routine.

Note: In case that the next $\overline{\text{NMI}}$ requests can occur during the $\overline{\text{NMI}}$ interrupt routine, it is necessary to forbid the request until the CPU execute an op-code at logical address 0067H.

After completion of the flow (3), the content of the current stack top changes as follows:


SP
SP+1
SP+2
SP+3
67H
00H
PC(L) (undefined op-code)
PC(H) (undefined op-code)
00H
00H
PC(L) (undefined op-code)
PC(H) (undefined op-code)

Note: The PC (undefined op-code) means an address which is stacked during TRAP acknowledge cycles.

3. HD64180 SOFTWARE ARCHITECTURE

3.1 Instruction Set

The HD64180 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.

New Instructions	Operation
SLP	Enter SLEEP mode
MLT	8-bit multiply with 16-bit result
IN0 g, (m)	Input contents of immediate I/O address into register
OUT0 (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

SLP – Sleep

The SLP instruction causes the HD64180 to enter SLEEP low power consumption mode. See section 2.4 for a complete description of the SLEEP state.

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.

IN0 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.

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.

OTIM, OTIMR, OTDM, OTDMR – Block I/O

The contents of memory pointed to by HL is 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 respectively. 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 HD64180 on-chip I/O) initialization. When I/O is accessed, 00H is output in high-order bits of address automatically.

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.

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).

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).

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).

3.2. CPU Registers

The HD64180 CPU registers consist of Register Set GR, Register Set GR' and Special Registers.

The Register Set GR consists of 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 alternate register set of Register Set GR and also contains 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 8-bit Interrupt Vector Register (I), 8-bit R Counter (R), two 16-bit Index Registers (IX and IY), 16-bit Stack Pointer (SP), and 16-bit Program Counter (PC).

Fig. 3.2.1 shows CPU registers configuration.

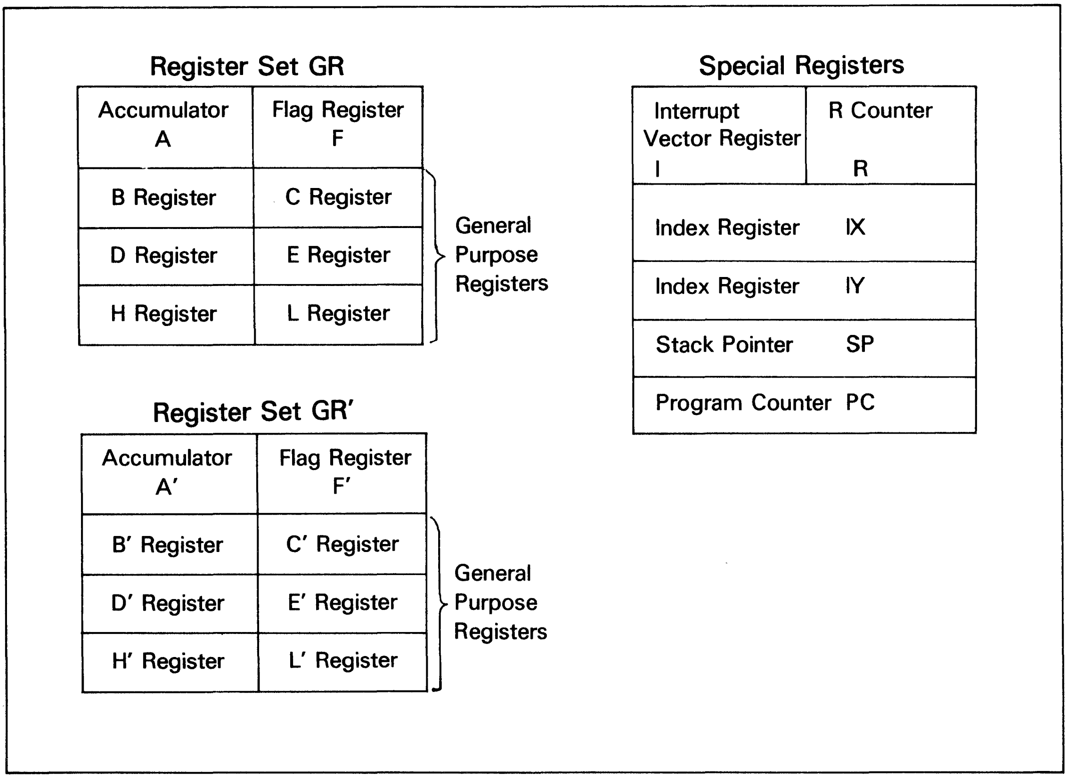


Figure 3.2.1 CPU Registers

3.2.1 Register description

Accumulator (A, A')

The Accumulator (A) serves as the primary register used for many arithmetic, logical and I/O instructions.

Flag Registers (F, F')

The flag register stores various status bits (described in the next section) which reflect the results of instruction execution.

General Purpose Registers (BC, BC', DE, DE', HL, HL')

The General Purpose Registers are used for both address and data operation. Depending on instruction, each half (8 bits) of these registers (B, C, D, E, H, and L) may also be used.

Interrupt Vector Register (I)

For interrupts which require a vector table address to be calculated ($\overline{\mathrm{INT}_0}$ Mode 2, $\overline{\mathrm{INT}_1}$, $\overline{\mathrm{INT}_2}$ 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.

R Counter (R)

The least significant seven bits of the R Counter (R) serve to count the number of instructions executed by the HD64180. R is incremented for each CPU op-code fetch cycles (each LIR cycles). R is cleared to 00H during RESET.

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.

Stack Pointer (SP)

The Stack Pointer (SP) contains the memory address based LIFO stack. SP is cleared to 0000H during RESET.

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.

3.2.2 Flag Register (F)

The Flag Register stores the logical state reflecting 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)

○ **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 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 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, +32767 to −32768 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.

3.3 Addressing Modes

The HD64180 instruction set includes eight addressing modes.

Implied Register
Register Direct
Register Indirect
Indexed
Extended
Immediate
Relative
IO

Implied Register (IMP)

Certain op-codes automatically imply register usage, such as the arithmetic operations which inherently reference the Accumulator, Index Registers, Stack Pointer and General Purpose Registers.

Register Direct (REG)

Many op-codes contain bit fields specifying registers to be used for the operation. The exact bit field definition vary depending on instruction as follows.

8-bit Register

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

ww field	Register
0 0	B C
0 1	D E
1 0	H L
1 1	S P

xx field	Register
0 0	B C
0 1	D E
1 0	I X
1 1	S P

16-bit Register

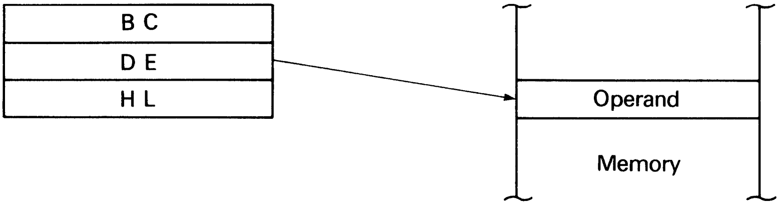
zz field	Register
0 0	B C
0 1	D E
1 0	H L
1 1	A F

yy field	Register
0 0	B C
0 1	D E
1 0	I Y
1 1	S P

Suffixed H and L to ww,xx,yy,zz (ex. wwH,IXL) indicate upper and lower 8-bit of the 16-bit register respectively.

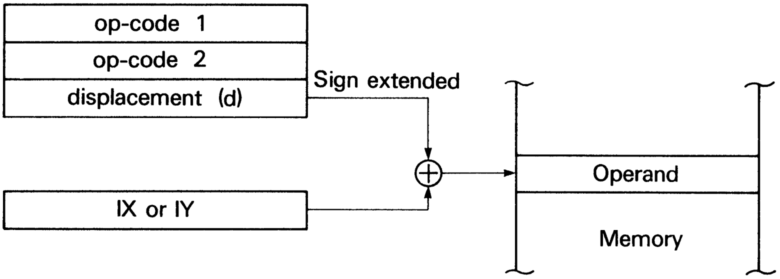
Register Indirect (REG)

The memory operand address is contained in one of the 16-bit General Purpose Registers (BC, DE and HL).



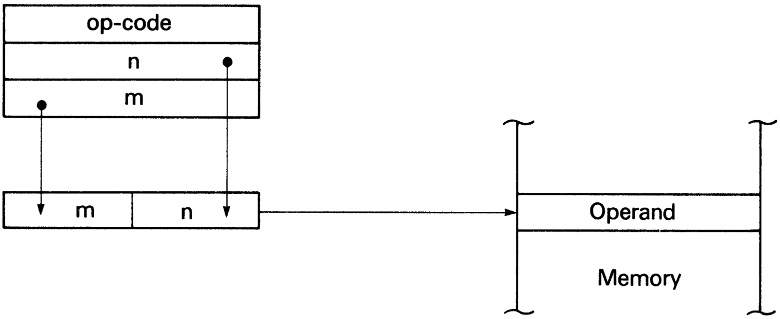
Indexed (INDX)

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.



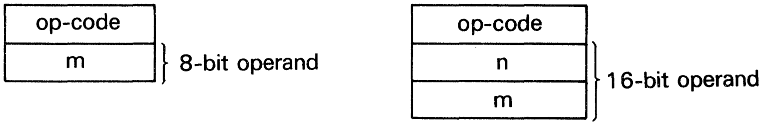
Extended (EXT)

The memory operand address is specified by two bytes contained in the instruction.



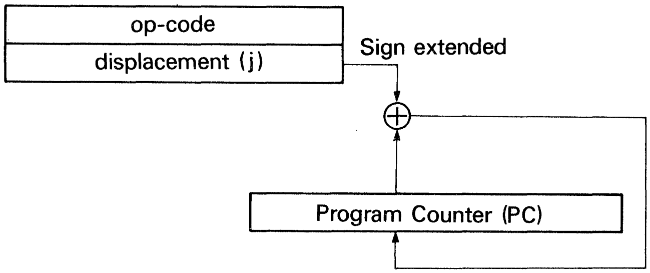
Immediate (IMMED)

The memory operands are contained within one or two bytes of the instruction.



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.



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 Accumulator is output to A_8-A_{15}.

(2) The Contents of Register B is output to A_0-A_7. The Contents of Register C is 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 is output to A_0-A_7. 00H is output to A_8-A_{15}.
(useful for internal I/O register access)

4. ELECTRICAL CHARACTERISTICS

4.1 HD64180R0 ELECTRICAL CHARACTERISTICS

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 ~ +7.0	V
Input Voltage	V_{in}	−0.3 ~ V_{CC}+0.3	V
Operating Temperature	T_{opr}	−20 ~ +75	°C
Storage Temperature	T_{stg}	−55 ~ +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.

■ DC CHARACTERISTICS (V_{CC} = 5V ± 10%, V_{SS} = 0V, Ta = −20 ~ +75°C, unless otherwise noted.)

Symbol	Item	Condition	min	typ	max	Unit
V_{IH1}	Input "H" Voltage $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		V_{CC}−0.6	–	V_{CC}+0.3	V
V_{IH2}	Input "H" Voltage Except $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		2.0	–	V_{CC}+0.3	V
V_{IL1}	Input "L" Voltage $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		−0.3	–	0.6	V
V_{IL2}	Input "L" Voltage Except $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		−0.3	–	0.8	V
V_{OH}	Output "H" Voltage All Outputs	I_{OH} = −200μA	2.4	–	–	V
		I_{OH} = −20μA	V_{CC}−1.2	–	–	
V_{OL}	Output "L" Voltage All Outputs	I_{OL} = 1.6 mA	–	–	0.45	V
I_{IL}	Input Leakage Current All Inputs Except XTAL, EXTAL	Vin=0.5 ~ V_{CC}−0.5	–	–	1.0	μA
I_{TL}	Three State Leakage Current	Vin=0.5 ~ V_{CC}−0.5	–	–	1.0	μA
I_{CC}*	Power Dissipation (Normal Operation)	f=4 MHz	–	10	20	mA
		f=6 MHz	–	15	30	
	Power Dissipation (SYSTEM STOP mode)	f=4 MHz	–	2.5	5.0	mA
		f=6 MHz	–	3.8	7.5	
Cp	Pin Capacitance	Vin=0V, f=1 MHz Ta=25°C	–	–	12	pF

* V_{IHmin} = V_{CC}− 1.0V, V_{ILmax} = 0.8V (all output terminals are at no load.)

■ AC CHARACTERISTICS (V_{CC} = 5V ± 10%, V_{SS} = 0V, Ta = −20 ~ +75°C, unless otherwise noted.)

Symbol	Item	HD64A180R0			HD64B180R0			Unit
		min	typ	max	min	typ	max	
t_{cyc}	Clock Cycle Time	250	–	2000	162	–	2000	ns
t_{CHW}	Clock "H" Pulse Width	110	–	–	57	–	–	ns
t_{CLW}	Clock "L" Pulse Width	100	–	–	57	–	–	ns
t_{cf}	Clock Fall Time	–	–	25	–	–	25	ns
t_{cr}	Clock Rise Time	–	–	20	–	–	20	ns
t_{AD}	Address Delay Time	–	–	110 130*	–	–	105 125*	ns
t_{AS}	Address Set-up Time ($\overline{ME}$ or $\overline{IOE}$ ↓)	45 30**	–	–	10 −15**	–	–	ns
t_{MED1}	$\overline{ME}$ Delay Time 1	–	–	85	–	–	75	ns
t_{RDD1}	$\overline{RD}$ Delay Time 1	–	–	85	–	–	75	ns
t_{LD1}	$\overline{LIR}$ Delay Time 1	–	–	105 120***	–	–	100 115***	ns
t_{AH}	Address Hold Time ($\overline{ME}$, $\overline{IOE}$, $\overline{RD}$ or $\overline{WR}$ ↑)	80	–	–	35	–	–	ns
t_{MED2}	$\overline{ME}$ Delay Time 2	–	–	85	–	–	75	ns
t_{RDD2}	$\overline{RD}$ Delay Time 2	–	–	85	–	–	75	ns
t_{LD2}	$\overline{LIR}$ Delay Time 2	–	–	105	–	–	100	ns
t_{DRS}	Data Read Set-up Time	50	–	–	45	–	–	ns
t_{DRH}	Data Read Hold Time	0	–	–	0	–	–	ns
t_{STD1}	ST Delay Time 1	–	–	110	–	–	100	ns
t_{STD2}	ST Delay Time 2	–	–	110	–	–	100	ns
t_{WS}	$\overline{WAIT}$ Set-up Time	80	–	–	40	–	–	ns
t_{WH}	$\overline{WAIT}$ Hold Time	70	–	–	40	–	–	ns

(to be continued)

NOTE) Each symbols shows the value at the following conditions.

*1. Just after RESET (Restart address = 00000H)
2. At the beginning of SLEEP mode or SYSTEM STOP mode
(Starting address = 7FFFFH)
3. After BUS RELEASE mode

**1. Just after RESET (Restart address = 00000H)
2. After BUS RELEASE mode

***1. Just after RESET (Restart address = 00000H)

Symbol	Item	HD64A180R0			HD64B180R0			Unit
		min	typ	max	min	typ	max	
t_{WDZ}	Write Data Floating Delay Time	–	–	100	–	–	95	ns
t_{WRD1}	$\overline{WR}$ Delay Time 1	–	–	90	–	–	80	ns
t_{WDD}	Write Data Delay Time	–	–	110	–	–	90	ns
t_{WDS}	Write Data Set-up Time ($\overline{WR}$ ↓)	60	–	–	40	–	–	ns
t_{WRD2}	$\overline{WR}$ Delay Time 2	–	–	90	–	–	80	ns
t_{WRP}	$\overline{WR}$ Pulse Width	220	–	–	135	–	–	ns
t_{WDH}	Write Data Hold Time ($\overline{WR}$ ↑)	60	–	–	40	–	–	ns
t_{IOD1}	$\overline{IOE}$ Delay Time 1	–	–	85	–	–	75	ns
t_{IOD2}	$\overline{IOE}$ Delay Time 2	–	–	85	–	–	75	ns
t_{IOD3}	$\overline{IOE}$ Delay Time 3 ($\overline{LIR}$ ↓)	540	–	–	340	–	–	ns
t_{INTS}	$\overline{INT}$ Set-up Time (ϕ ↓)	80	–	–	70	–	–	ns
t_{INTH}	$\overline{INT}$ Hold Time (ϕ ↓)	70	–	–	60	–	–	ns
t_{NMIW}	$\overline{NMI}$ Pulse Width	120	–	–	120	–	–	ns
t_{BRS}	$\overline{BUSREQ}$ Set-up Time (ϕ ↓)	80	–	–	70	–	–	ns
t_{BRH}	$\overline{BUSREQ}$ Hold Time (ϕ ↓)	70	–	–	60	–	–	ns
t_{BAD1}	$\overline{BUSACK}$ Delay Time 1	–	–	100	–	–	95	ns
t_{BAD2}	$\overline{BUSACK}$ Delay Time 2	–	–	100	–	–	95	ns
t_{BZD}	Bus Floating Delay Time	–	–	130	–	–	125	ns
t_{MEWH}	$\overline{ME}$ Pulse Width (HIGH)	200	–	–	110	–	–	ns
t_{MEWL}	$\overline{ME}$ Pulse Width (LOW)	210	–	–	125	–	–	ns

(to be continued)

Symbol	Item	HD64A180R0			HD64B180R0			Unit
		min	typ	max	min	typ	max	
t_{RFD1}	$\overline{\text{REF}}$ Delay Time 1	–	–	110	–	–	100	ns
t_{RFD2}	$\overline{\text{REF}}$ Delay Time 2	–	–	110	–	–	100	ns
t_{HAD1}	$\overline{\text{HALT}}$ Delay Time 1	–	–	110	–	–	100	ns
t_{HAD2}	$\overline{\text{HALT}}$ Delay Time 2	–	–	110	–	–	100	ns
t_{DRQS}	$\overline{\text{DREQi}}$ Set-up Time	80	–	–	70	–	–	ns
t_{DRQH}	$\overline{\text{DREQi}}$ Hold Time	70	–	–	60	–	–	ns
t_{TED1}	$\overline{\text{TENDi}}$ Delay Time 1	–	–	85	–	–	70	ns
t_{TED2}	$\overline{\text{TENDi}}$ Delay Time 2	–	–	85	–	–	70	ns
t_{ED1}	Enable Delay Time 1	–	–	100	–	–	95	ns
t_{ED2}	Enable Delay Time 2	–	–	100	–	–	95	ns
t_{TOD}	Timer Output Delay Time	–	–	300	–	–	300	ns
t_{STDI}	CSI/O Transmit Data Delay Time (Internal Clock Operation)	–	–	200	–	–	200	ns
t_{STDE}	CSI/O Transmit Data Delay Time (External Clock Operation)	–	–	7.5 tcyc +300	–	–	7.5 tcyc +300	ns
t_{SRSI}	CSI/O Receive Data Set-up time (Internal Clock Operation)	1	–	–	1	–	–	tcyc
t_{SRHI}	CSI/O Receive Data Hold Time (Internal Clock Operation)	1	–	–	1	–	–	tcyc
t_{SRSE}	CSI/O Receive Data Set-up Time (External Clock Operation)	1	–	–	1	–	–	tcyc
t_{SRHE}	CSI/O Receive Data Hold Time (External Clock Operation)	1	–	–	1	–	–	tcyc
t_{RES}	$\overline{\text{RESET}}$ Set-up Time	120	–	–	120	–	–	ns
t_{REH}	$\overline{\text{RESET}}$ Hold Time	80	–	–	80	–	–	ns
t_{OSC}	Oscillator Stabilization Time	–	–	20	–	–	20	ms
t_{EXr}	External Clock Rise Time (EXTAL)	–	–	25	–	–	25	ns
t_{EXf}	External Clock Fall Time (EXTAL)	–	–	25	–	–	25	ns
t_{Rr}	$\overline{\text{RESET}}$ Rise Time	–	–	50	–	–	50	ms
t_{Rf}	$\overline{\text{RESET}}$ Fall Time	–	–	50	–	–	50	ms
t_{Ir}	Input Rise Time (except EXTAL, $\overline{\text{RESET}}$)	–	–	100	–	–	100	ns
t_{If}	Input Fall Time (except EXTAL, $\overline{\text{RESET}}$)	–	–	100	–	–	100	ns

4.2 HD64180R1 AND HD64180Z ELECTRICAL CHARACTERISTICS

■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	$-0.3 \sim +7.0$	V
Input Voltage	V_{in}	$-0.3 \sim V_{CC}+0.3$	V
Operating Temperature	T_{opr}	$-20 \sim +75$	°C
Storage Temperature	T_{stg}	$-55 \sim +150$	°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.

■ DC CHARACTERISTICS (V$_{CC}$ = 5V ± 10%, V$_{SS}$ = 0V, Ta = −20 ~ +75°C, unless otherwise noted.)

Symbol	Item	Condition	min	typ	max	Unit
V_{IH1}	Input "H" Voltage $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		$V_{CC}-0.6$	–	$V_{CC}+0.3$	V
V_{IH2}	Input "H" Voltage Except $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		2.0	–	$V_{CC}+0.3$	V
V_{IL1}	Input "L" Voltage $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		-0.3	–	0.6	V
V_{IL2}	Input "L" Voltage Except $\overline{\text{RESET}}$, EXTAL, $\overline{\text{NMI}}$		-0.3	–	0.8	V
V_{OH}	Output "H" Voltage All outputs	$I_{OH} = -200\mu A$	2.4	–	–	V
		$I_{OH} = -20\mu A$	$V_{CC}-1.2$	–	–	
V_{OL}	Output "L" Voltage All Outputs	$I_{OL} = 2.2$ mA	–	–	0.45	V
I_{IL}	Input Leakage Current All Inputs Except XTAL, EXTAL	Vin=0.5 ~ $V_{CC}-0.5$	–	–	1.0	μA
I_{TL}	Three State Leakage Current	Vin=0.5 ~ $V_{CC}-0.5$	–	–	1.0	μA
I_{CC}*	Power Dissipation* (Normal Operation)	f=4 MHz	–	10	20	mA
		f=6 MHz	–	15	30	
		f=8 MHz	–	20	40	
	Power Dissipation* (SYSTEM STOP mode)	f=4 MHz	–	2.5	5.0	
		f=6 MHz	–	3.8	7.5	
		f=8 MHz	–	5.0	10.0	
Cp	Pin Capacitance	Vin=0V, f=1 MHz Ta=25°C	–	–	12	pF

* $V_{IHmin} = V_{CC}-1.0V$, $V_{ILmax} = 0.8V$ (all output terminals are at no load.)

■ HD64180R1 AC CHARACTERISTICS (V_{CC} = 5V ± 10%, V_{SS} = 0V, Ta = −20 ~ +75°C, unless otherwise noted.)

Symbol	Item	HD64180R1-4			HD64180R1-6			HD64180R1-8			Unit
		min	typ	max	min	typ	max	min	typ	max	
t_{cyc}	Clock Cycle Time	250	–	2000	162	–	2000	125	–	2000	ns
t_{CHW}	Clock "H" Pulse Width	110	–	–	65	–	–	50	–	–	ns
t_{CLW}	Clock "L" 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	–	–	85	–	–	60	–	–	50	ns
t_{LD1}	$\overline{LIR}$ Delay Time 1	–	–	100	–	–	80	–	–	70*	ns
t_{AH}	Address Hold Time ($\overline{ME}$, $\overline{IOE}$, $\overline{RD}$ 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*	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

(to be continued)

* 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.

Symbol	Item	HD64180R1-4			HD64180R1-6			HD64180R1-8			Unit
		min	typ	max	min	typ	max	min	typ	max	
t_{WDZ}	Write Data Floating Delay Time	—	—	100	—	—	95	—	—	70	ns
t_{WRD1}	$\overline{\text{WR}}$ Delay Time 1	—	—	90	—	—	65	—	—	60	ns
t_{WDD}	Write Data Delay Time	—	—	110	—	—	90	—	—	80	ns
t_{WDS}	Write Data Set-up Time ($\overline{\text{WR}}$ ↓)	60	—	—	40	—	—	20	—	—	ns
t_{WRD2}	$\overline{\text{WR}}$ Delay Time 2	—	—	90	—	—	80	—	—	60	ns
t_{WRP}	$\overline{\text{WR}}$ Pulse Width	280	—	—	170	—	—	130	—	—	ns
t_{WDH}	Write Data Hold Time ($\overline{\text{WR}}$ ↑)	60	—	—	40	—	—	15	—	—	ns
t_{IOD1}	$\overline{\text{IOE}}$ Delay Time 1	—	—	85	—	—	60	—	—	50	ns
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

(to be continued)

(V_{CC} = 5V ±10%, V_{SS} = 0V, Ta = −20 ~ +75°C, unless otherwise noted.)

Symbol	Item	HD64180R1-4			HD64180R1-6			HD64180R1-8			Unit
		min	typ	max	min	typ	max	min	typ	max	
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}	$\overline{\text{DREQ}}$i Set-up Time	80	–	–	40	–	–	40	–	–	ns
t_{DRQH}	$\overline{\text{DREQ}}$i Hold Time	70	–	–	40	–	–	40	–	–	ns
t_{TED1}	$\overline{\text{TEND}}$i Delay Time 1	–	–	85	–	–	70	–	–	60	ns
t_{TED2}	$\overline{\text{TEND}}$i 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
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)	–	–	7.5 tcyc +300	–	–	7.5 tcyc +300	–	–	7.5 tcyc +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 Tirne	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, RESET)	–	–	100	–	–	100	–	–	100	ns
t_{If}	Input Fall Time (except EXTAL, RESET)	–	–	100	–	–	100	–	–	100	ns

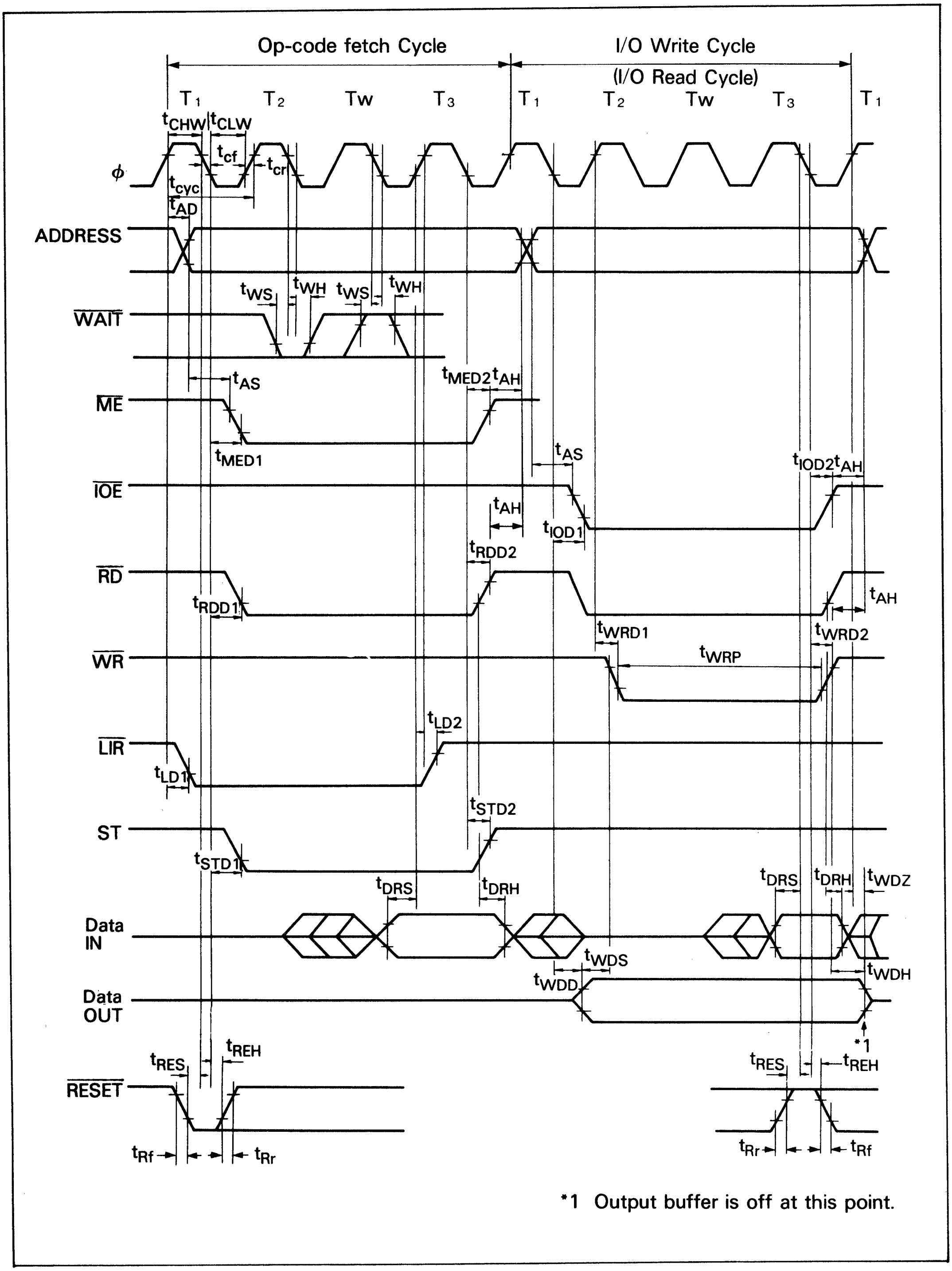
■HD64180Z AC Characteristics (V_{CC} = 5V ± 10%, V_{SS} = 0V, Ta = −20 to +75°C, unless otherwise noted.)

Symbol	Item		HD64180Z-4		HD64180Z-6		HD64180Z-8		Unit
			min	max	min	max	min	max	
t_{cyc}	Clock Cycle Time		250	2000	162	2000	125	2000	ns
t_{CHW}	Clock "H" Pulse Width		110	—	65	—	50	—	ns
t_{CLW}	Clock "L" 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 ($\overline{\text{ME}}$ or $\overline{\text{IOE}}$ ↓)		50	—	30	—	20	—	ns
t_{MED1}	$\overline{\text{ME}}$ Delay Time 1		—	85	—	60	—	50	ns
t_{RDD1}	$\overline{\text{RD}}$ Delay Time 1	$\overline{\text{IOC}}$ = 1	—	85	—	60	—	50	ns
		$\overline{\text{IOC}}$ = 0	—	85	—	65	—	60	
t_{LD1}	$\overline{\text{LIR}}$ Delay Time 1		—	100	—	80	—	70*	ns
t_{AH}	Address Hold Time 1 ($\overline{\text{ME}}$, $\overline{\text{IOE}}$, $\overline{\text{RD}}$ or $\overline{\text{WR}}$ ↑)		80	—	35	—	20	—	ns
t_{MED2}	$\overline{\text{ME}}$ Delay Time 2		—	85	—	60	—	50	ns
t_{RDD2}	$\overline{\text{RD}}$ Delay Time 2		—	85	—	60	—	50	ns
t_{LD2}	$\overline{\text{LIR}}$ Delay Time 2		—	100	—	80	—	70*	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}	$\overline{\text{WAIT}}$ Set-up Time		80	—	40	—	40	—	ns
t_{WH}	$\overline{\text{WAIT}}$ Hold Time		70	—	40	—	40	—	ns
t_{WDZ}	Write Data Floating Delay Time		—	100	—	95	—	70	ns
t_{WRD1}	$\overline{\text{WR}}$ Delay Time 1		—	90	—	65	—	60	ns
t_{WDD}	Write Data Delay Time		—	110	—	90	—	80	ns
t_{WDS}	Write Data Set-up Time ($\overline{\text{WR}}$ ↓)		60	—	40	—	20	—	ns
t_{WRD2}	$\overline{\text{WR}}$ Delay Time 2		—	90	—	80	—	60	ns
t_{WRP}	$\overline{\text{WR}}$ Pulse Width		280	—	170	—	130	—	ns

* 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.

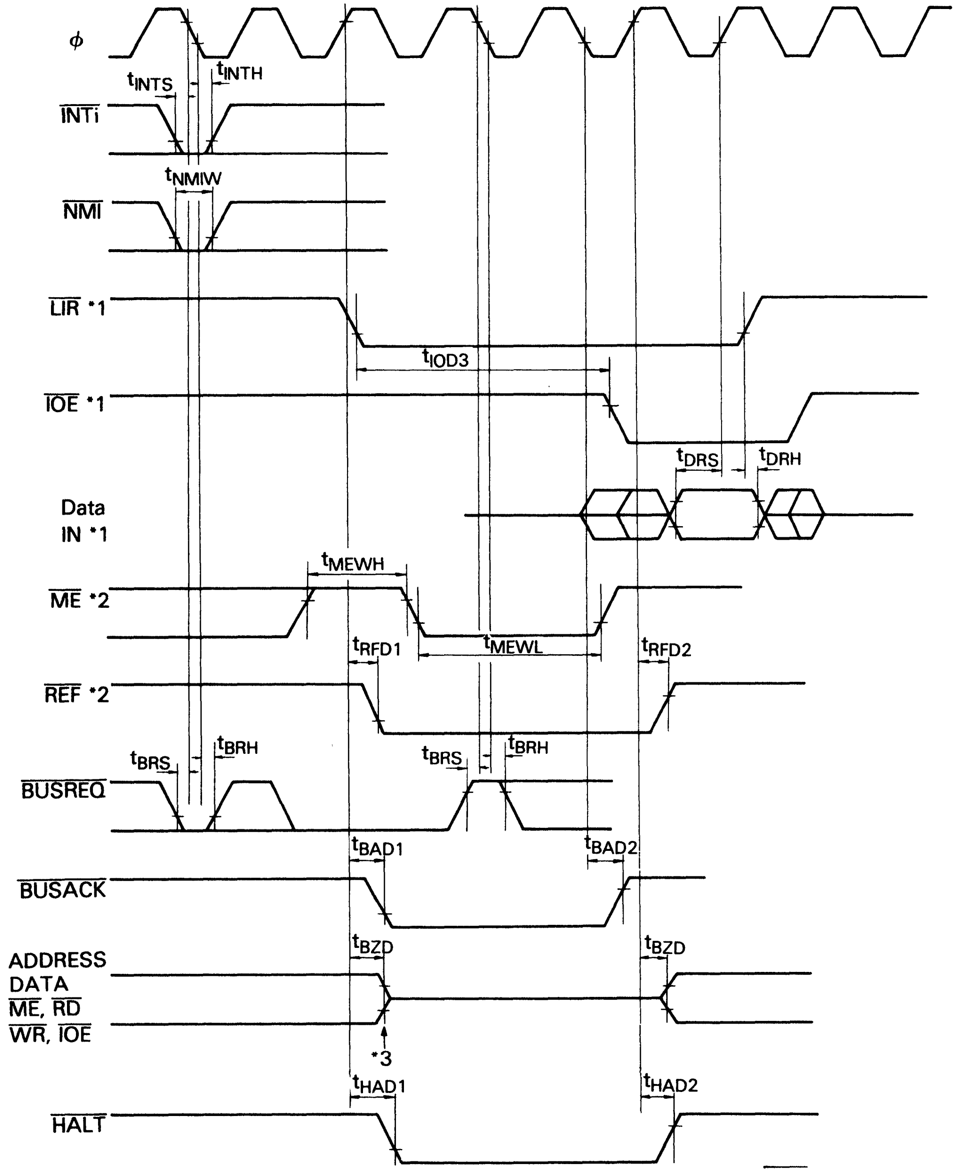
Symbol	Item		HD64180Z-4		HD64180Z-6		HD64180Z-8		Unit
			min	max	min	max	min	max	
t_{WDH}	Write Data Hold Time ($\overline{WR}$ ↑)		60	—	40	—	15	—	ns
t_{IOD1}	$\overline{IOE}$ Delay Time 1	$\overline{IOC}$= 1	—	85	—	60	—	50	ns
		$\overline{IOC}$= 0	—	85	—	65	—	60	
t_{IOD2}	$\overline{IOE}$ Delay Time 2		—	85	—	60	—	50	ns
t_{IOD3}	$\overline{IOE}$ Delay Time 3 ($\overline{LIR}$ ↓)		540	—	340	—	250	—	ns
t_{INTS}	$\overline{INT}$ Set-up Time (ϕ ↓)		80	—	40	—	40	—	ns
t_{INTH}	$\overline{INT}$ Hold Time (ϕ ↓)		70	—	40	—	40	—	ns
t_{NMIW}	$\overline{NMI}$ Pulse Width		120	—	120	—	100	—	ns
t_{BRS}	$\overline{BUSREQ}$ Set-up Time (ϕ ↓)		80	—	40	—	40	—	ns
t_{BRH}	$\overline{BUSREQ}$ Hold Time (ϕ ↓)		70	—	40	—	40	—	ns
t_{BAD1}	$\overline{BUSACK}$ Delay Time 1		—	100	—	95	—	70	ns
t_{BAD2}	$\overline{BUSACK}$ Delay Time 2		—	100	—	95	—	70	ns
t_{BZD}	Bus Floating Delay Time		—	130	—	125	—	90	ns
t_{MEWH}	$\overline{ME}$ Pulse Width (HIGH)		200	—	110	—	90	—	ns
t_{MEWL}	$\overline{ME}$ Pulse Width (LOW)		210	—	125	—	100	—	ns
t_{RFD1}	$\overline{REF}$ Delay Time 1		—	110	—	90	—	80	ns
t_{RFD2}	$\overline{REF}$ Delay Time 2		—	110	—	90	—	80	ns
t_{HAD1}	$\overline{HALT}$ Delay Time 1		—	110	—	90	—	80	ns
t_{HAD2}	$\overline{HALT}$ Delay Time 2		—	110	—	90	—	80	ns
t_{DRQS}	$\overline{DREQi}$ Set-up Time		80	—	40	—	40	—	ns
t_{DRQH}	$\overline{DREQi}$ Hold Time		70	—	40	—	40	—	ns
t_{TED1}	$\overline{TENDi}$ Delay Time 1		—	85	—	70	—	60	ns
t_{TED2}	$\overline{TENDi}$ 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	HD64180Z-4		HD64180Z-6		HD64180Z-8		Unit
		min	max	min	max	min	max	
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



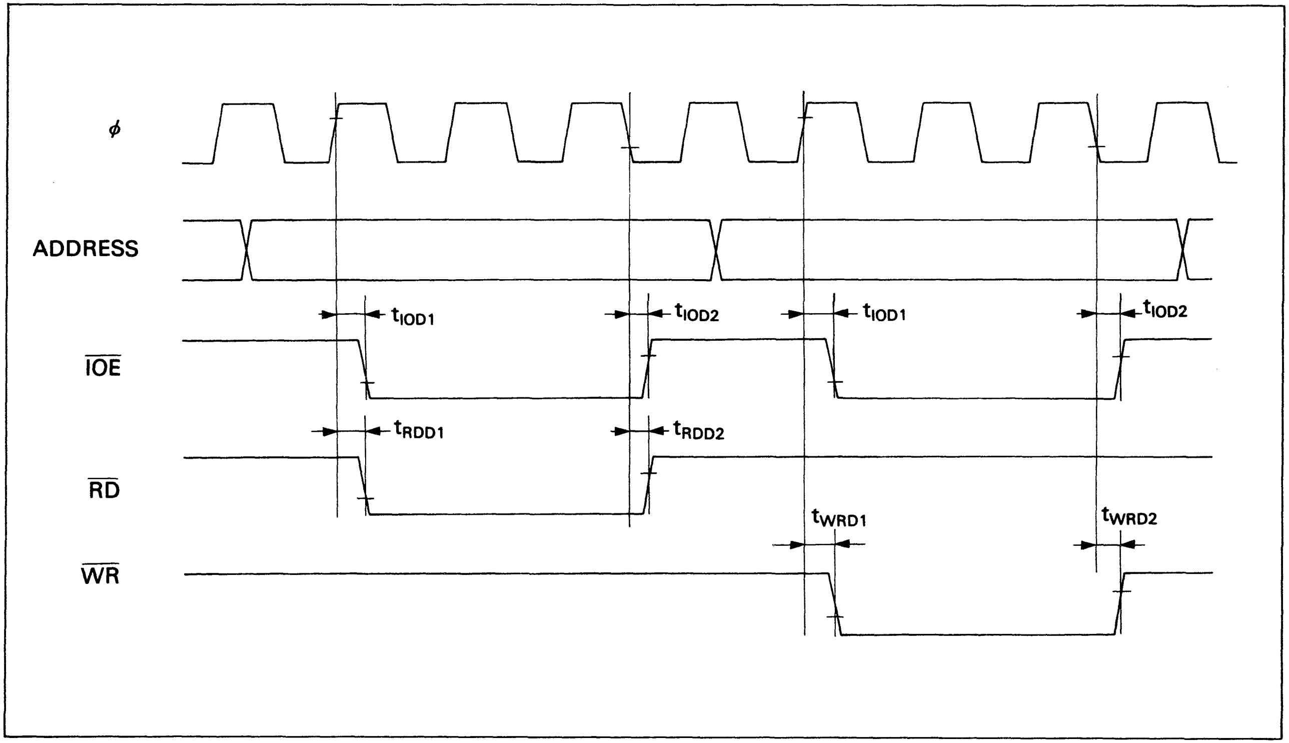
*1 Output buffer is off at this point.

CPU Timing (Op-code fetch Cycle
I/O Write Cycle
I/O Read Cycle)

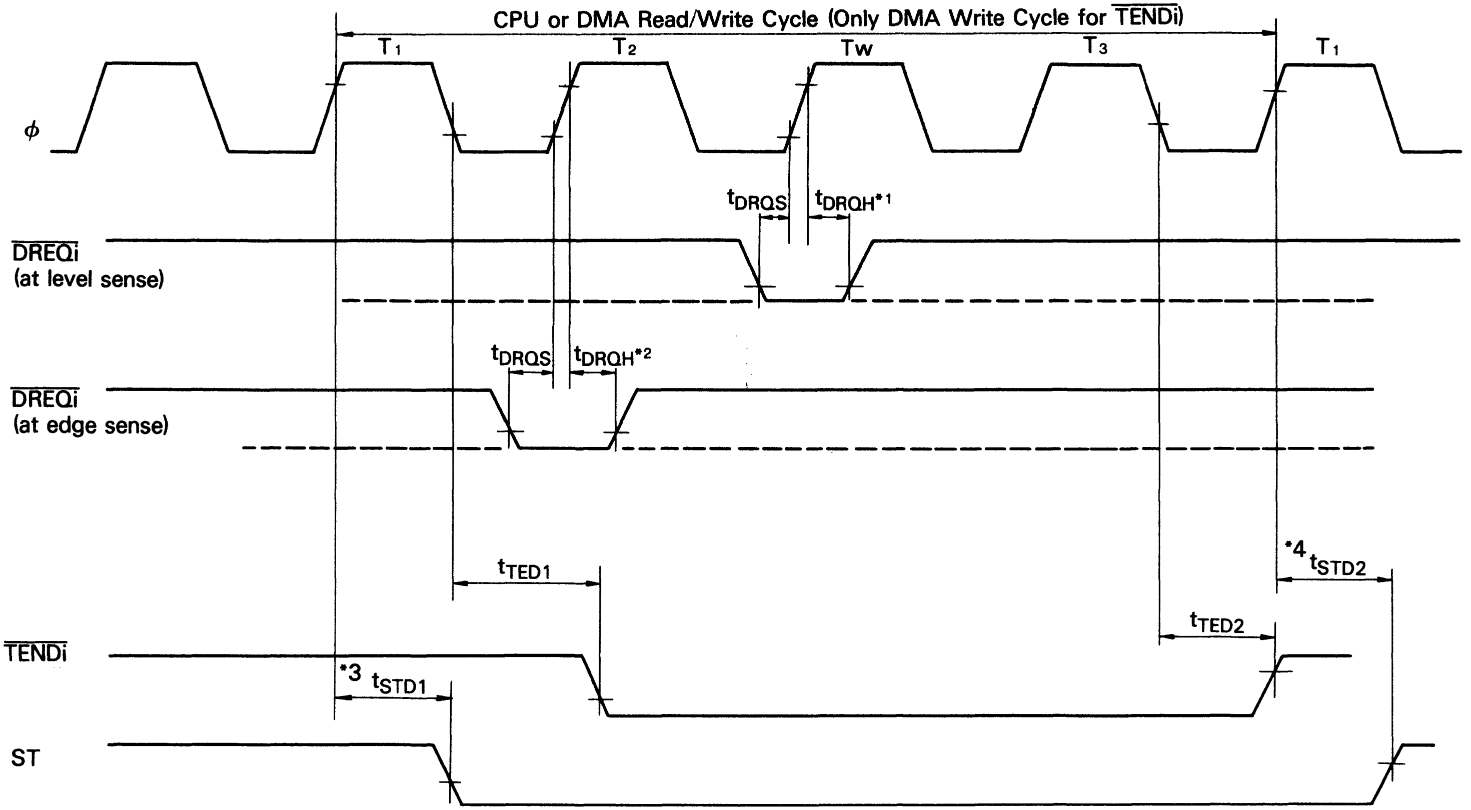


*1 during $\overline{\text{INT}}_0$ acknowledge cycle
*2 during refresh cycle
*3 Output buffer is off at this point.

CPU Timing { **$\overline{\text{INT}}_0$ Acknowledge cycle**
Refresh Cycle
BUS RELEASE Mode
HALT Mode
SLEEP Mode
SYSTEM STOP Mode }

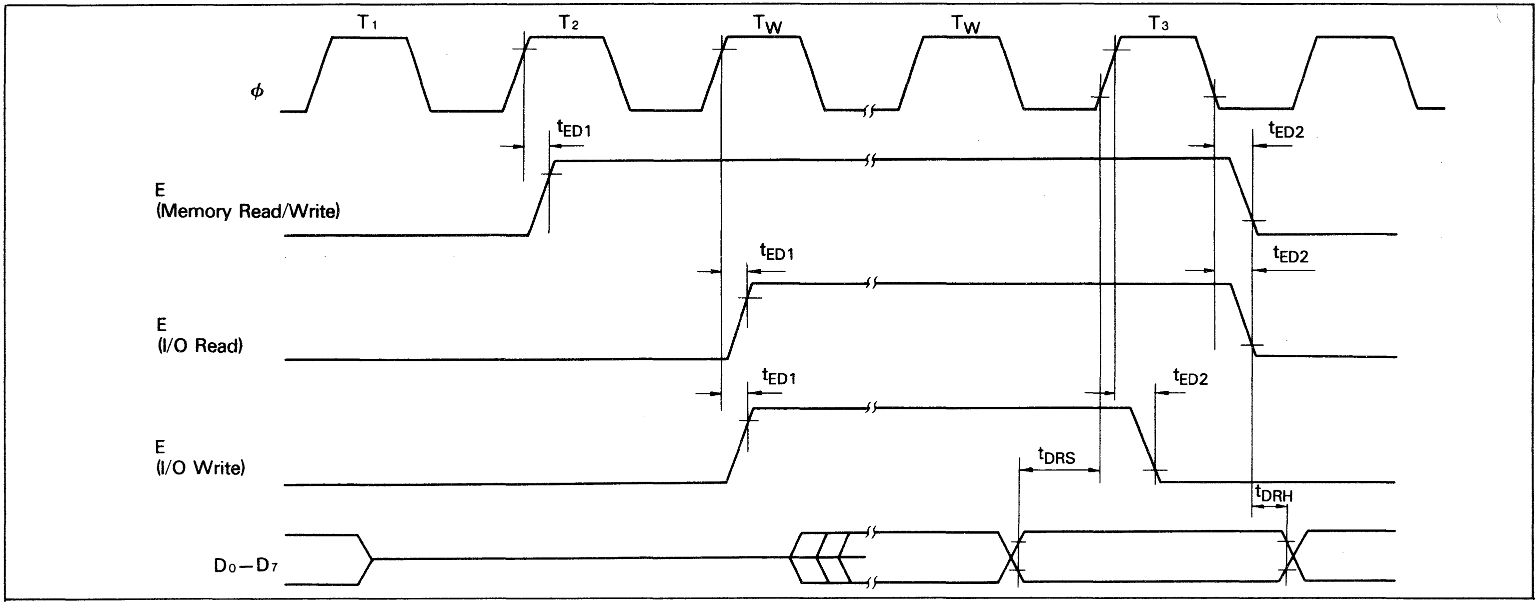


CPU Timing ($\overline{IOC}$=0)

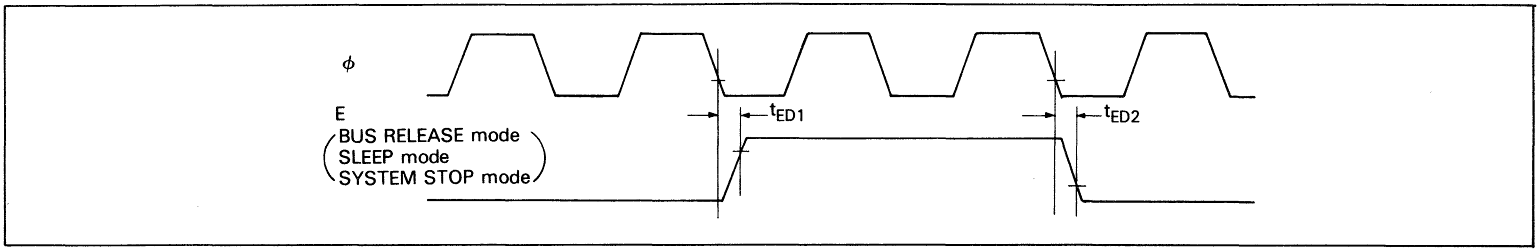

CPU or DMA Read/Write Cycle (Only DMA Write Cycle for $\overline{\text{TENDi}}$)
T_1
T_2
Tw
T_3
T_1
ϕ
t_{DRQS}
t_{DRQH}*1
$\overline{\text{DREQi}}$
(at level sense)
t_{DRQS}
t_{DRQH}*2
$\overline{\text{DREQi}}$
(at edge sense)
t_{TED1}
*4 t_{STD2}
t_{TED2}
$\overline{\text{TENDi}}$
*3 t_{STD1}
ST

*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.

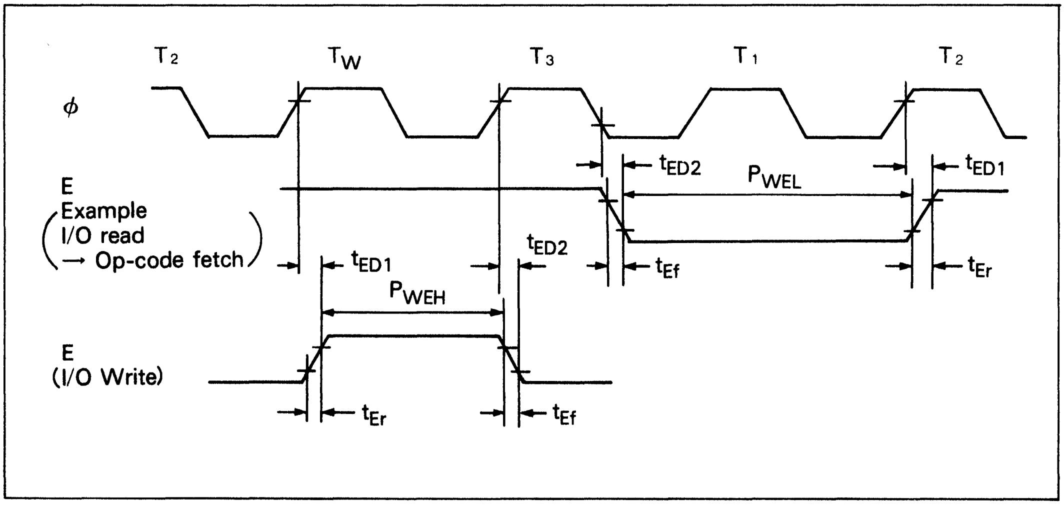
DMA Control Signals



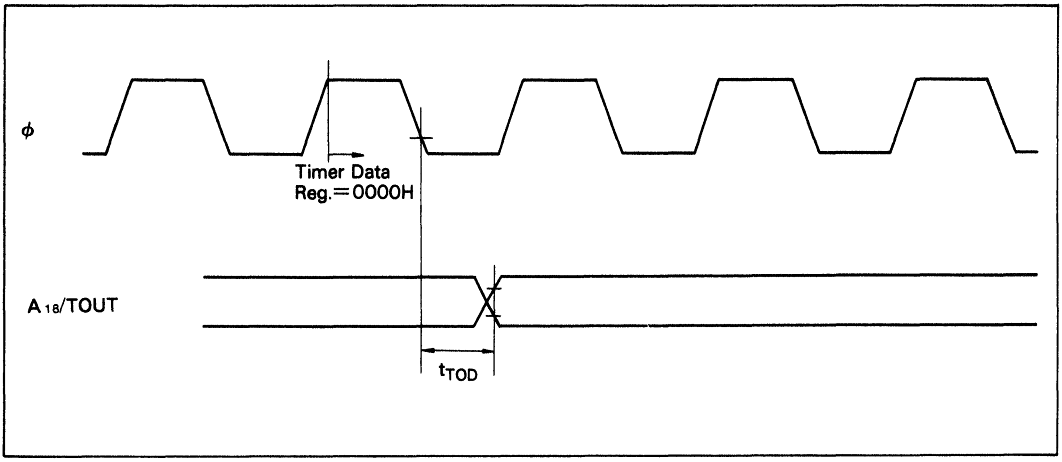
E Clock Timing (Memory Read/Write Cycle / I/O Read/Write Cycle)



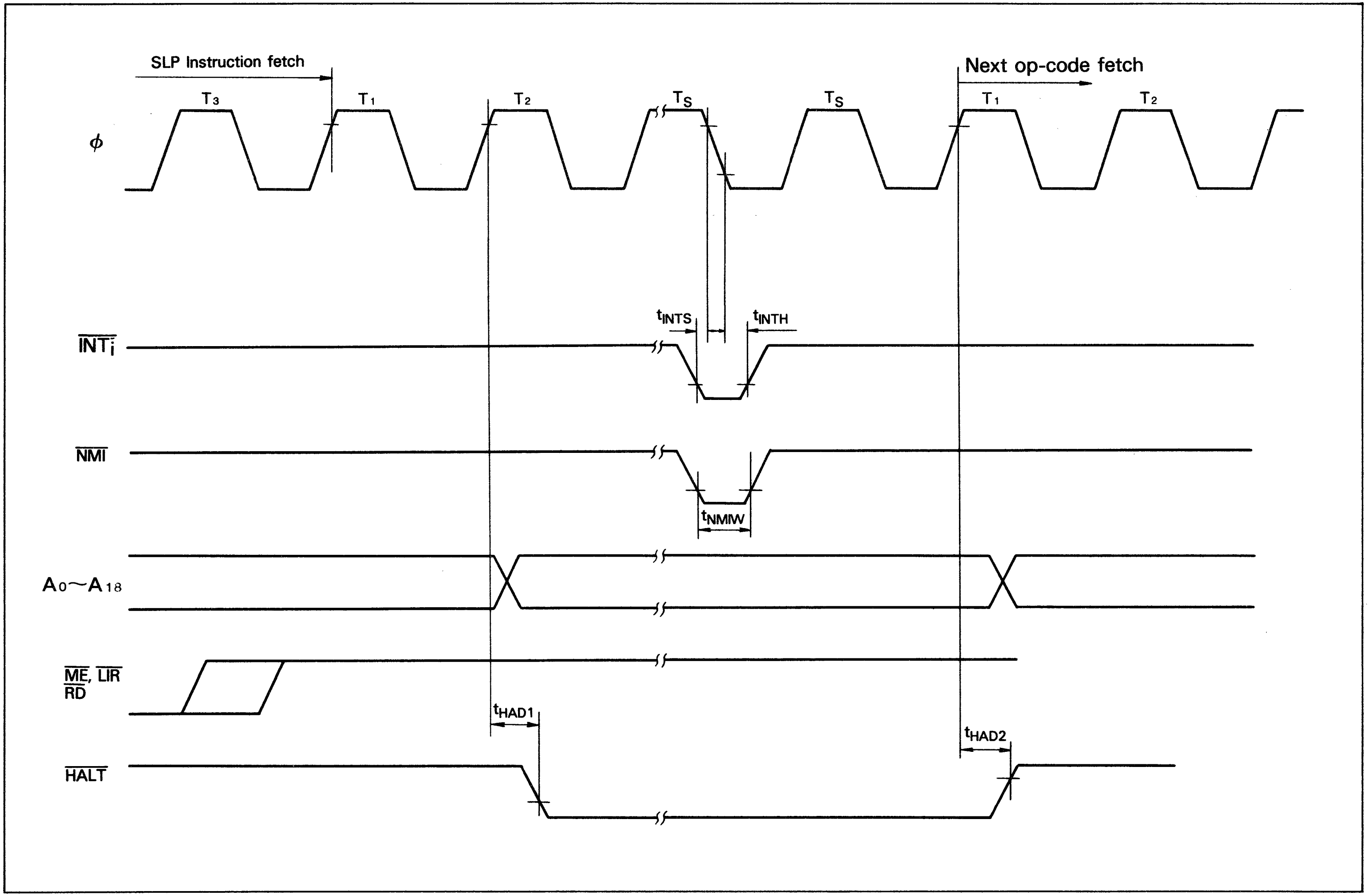
E Clock Timing (BUS RELEASE Mode / SLEEP Mode / SYSTEM STOP Mode)



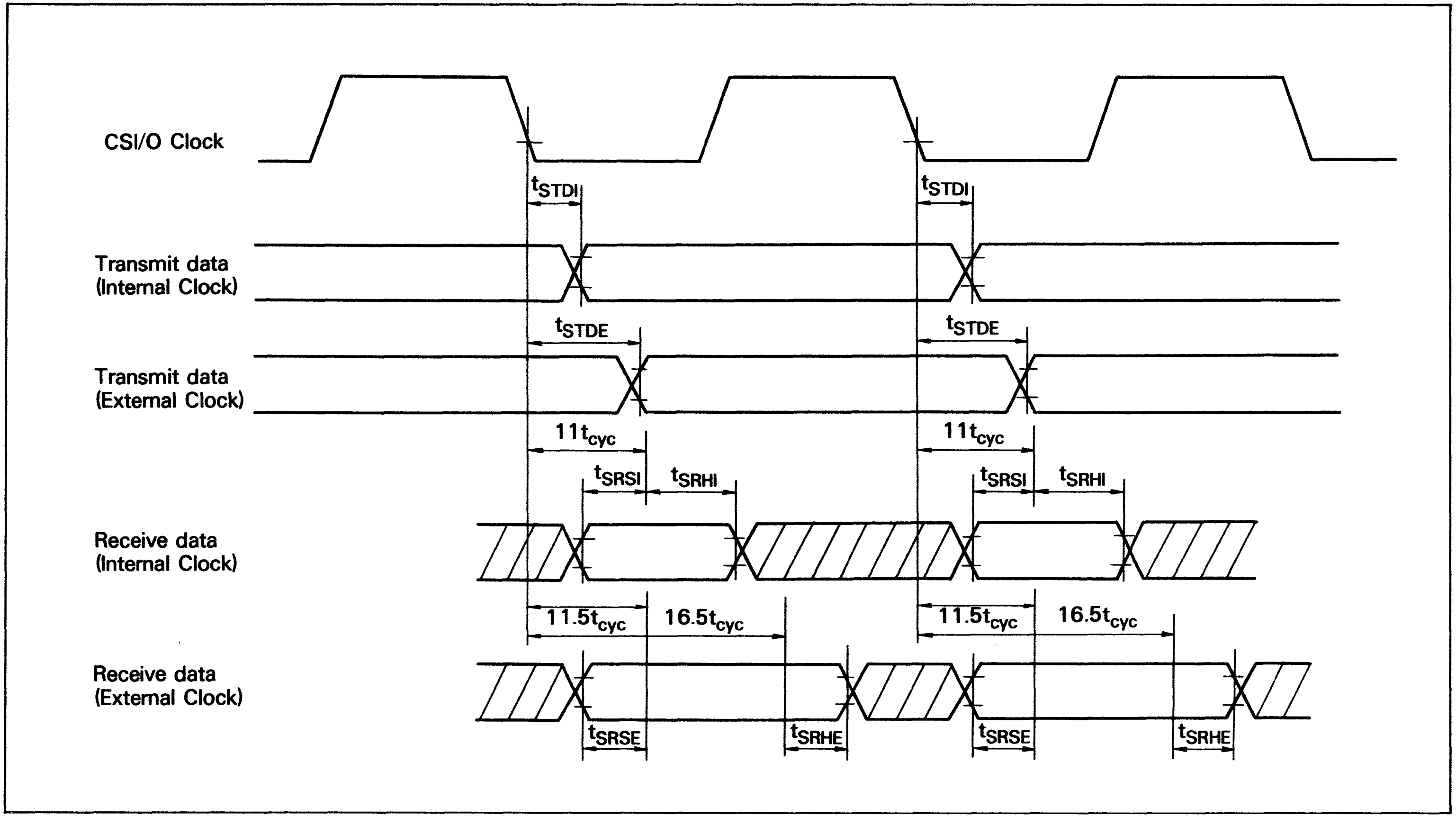
E Clock Timing (**Minimum timing example of** P_{WEL} **and** P_{WEH})



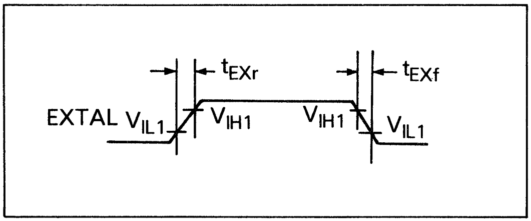
Timer Output Timing



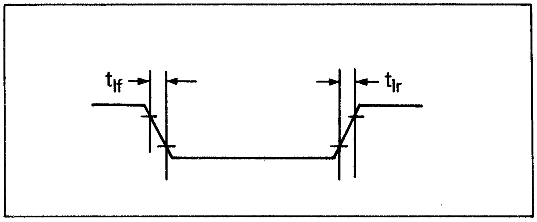
SLP Execution Cycle



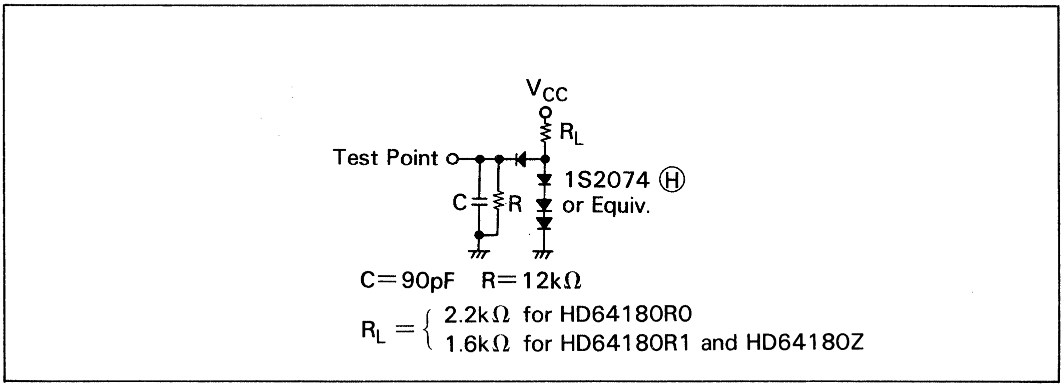
CSI/O Receive/Transmit Timing



External Clock Rise Time and Fall Time

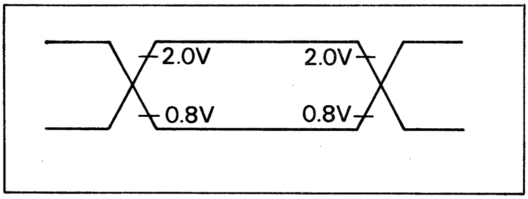


Input Rise Time and Fall Time (Except EXTAL, $\overline{\text{RESET}}$)

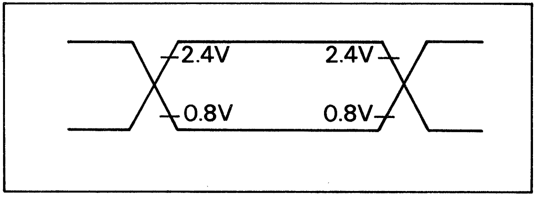


V_{CC}
R_L
Test Point
1S2074 Ⓗ
or Equiv.
C
R
C=90pF R=12kΩ
$R_L = \begin{cases} 2.2k\Omega \text{ for HD64180R0} \\ 1.6k\Omega \text{ for HD64180R1 and HD64180Z} \end{cases}$

Bus Timing Test Load (TTL Load)



Reference Level (Input)

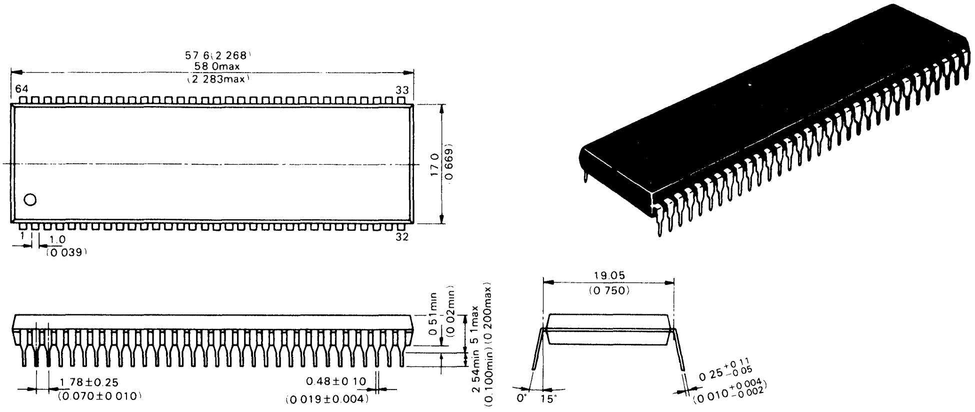


Reference Level (Output)

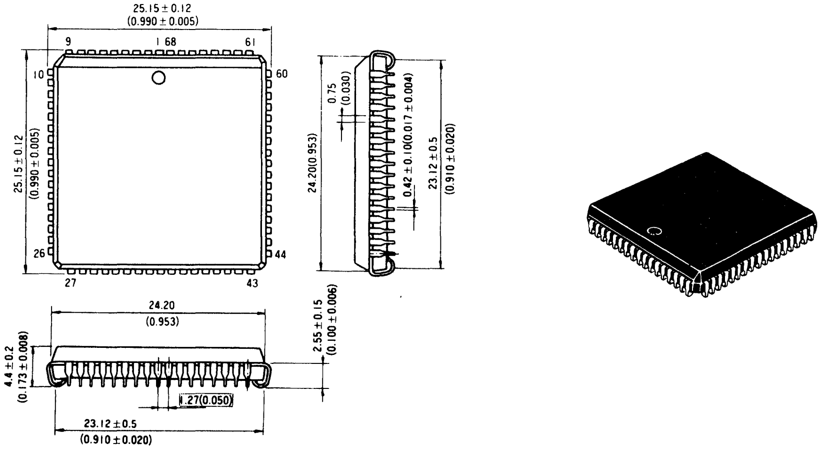
5. HD64180 PACKAGE DIMENSIONS

Unit: mm (inch)

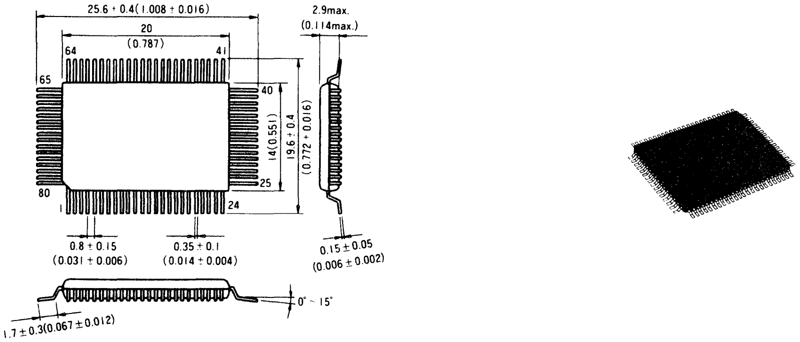
DP-64S



CP-68



FP-80



APPENDIX

A. Instruction Set

The following explains the symbols in instruction set.

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 pair of 16-bit registers. The following tables show the correspondence between symbols and registers.

g,g’	Reg.
000	B
001	C
010	D
011	E
100	H
101	L
111	A

ww	Reg.
00	BC
01	DE
10	HL
11	SP

xx	Reg.
00	BC
01	DE
10	IX
11	SP

yy	Reg.
00	BC
01	DE
10	IY
11	SP

zz	Reg.
00	BC
01	DE
10	HL
11	AF

NOTE: Suffixed H and L to ww,xx,yy,zz (ex.wwH,IXL) indicate upper and lower 8-bit of the 16-bit register respectively.

2. Bit

b specifies a bit to be manipulated in the bit manipulation instruction. The following table shows the correspondence between b and bits.

b	Bit
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

3. Condition

f specifies the condition in program control instructions. The following shows the correspondence between f and conditions.

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

4. Restart Address

v specifies a restart address. The following table shows the correspondence between v and restart addresses.

v	Address
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

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

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
$b\cdot(\quad)_M$	:	a content of bit b in the memory address
b·gr	:	a content of bit b in the register gr
d or j	:	8-bit signed displacement
S	:	source addressing mode
D	:	destination addressing mode
$\cdot$	:	AND operation
$+$	:	OR operation
$\oplus$	:	EXCLUSIVE OR operation
**	:	added new instructions to Z80

1 . Data Manipulation Instructions

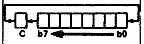
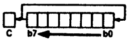
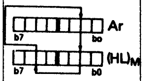
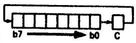
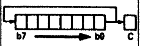
(1) Arithmetic and Logical Instructions (8-bit)

Operation name	MNEMONICS	OP code	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
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	S					D		2	6	Ar+m→Ar	↕	↕	↕	V	R	↕
		〈 m 〉																
	ADD A,(IX+d)	11 011 101			S			D		3	14	Ar+$(IX+d)_M$→Ar	↕	↕	↕	V	R	↕
		10 000 110																
		〈 d 〉																
	ADD A,(IY+d)	11 111 101			S			D		3	14	Ar+$(IY+d)_M$→Ar	↕	↕	↕	V	R	↕
		10 000 110																
		〈 d 〉																
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	S					D		2	6	Ar+m+c→Ar	↕	↕	↕	V	R	↕
		〈 m 〉																
	ADC A,(IX+d)	11 011 101			S			D		3	14	Ar+$(IX+d)_M$+c→Ar	↕	↕	↕	V	R	↕
		10 001 110																
		〈 d 〉																
	ADC A,(IY+d)	11 111 101			S			D		3	14	Ar+$(IY+d)_M$+c→Ar	↕	↕	↕	V	R	↕
		10 001 110																
		〈 d 〉																
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	S					D		2	6	Ar·m→Ar	↕	↕	S	P	R	R
		〈 m 〉																
	AND (IX+d)	11 011 101			S			D		3	14	Ar·$(IX+d)_M$→Ar	↕	↕	S	P	R	R
		10 100 110																
		〈 d 〉																
	AND (IY+d)	11 111 101			S			D		3	14	Ar·$(IY+d)_M$→Ar	↕	↕	S	P	R	R
		10 100 110																
		〈 d 〉																
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	S					D		2	6	Ar−m	↕	↕	↕	V	S	↕
		〈 m 〉																
	CP (IX+d)	11 011 101			S			D		3	14	Ar−$(IX+d)_M$	↕	↕	↕	V	S	↕
		10 111 110																
		〈 d 〉																
	CP (IY+d)	11 111 101			S			D		3	14	Ar−$(IY+d)_M$	↕	↕	↕	V	S	↕
		10 111 110																
		〈 d 〉																
COMPLE-MENT	CPL	00 101 111						S/D		1	3	$\overline{Ar}$→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			S/D					3	18	$(IX+d)_M$−1→	↕	↕	↕	V	S	·
		00 110 101										$(IX+d)_M$						
		〈 d 〉																
	DEC (IY+d)	11 111 101			S/D					3	18	$(IY+d)_M$−1→	↕	↕	↕	V	S	·
		00 110 101										$(IY+d)_M$						
		〈 d 〉																
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			S/D					3	18	$(IX+d)_M$+1→	↕	↕	↕	V	R	·
		00 110 100										$(IX+d)_M$						
		〈 d 〉																
	INC (IY+d)	11 111 101			S/D					3	18	$(IY+d)_M$+1→	↕	↕	↕	V	R	·
		00 110 100										$(IY+d)_M$						
		〈 d 〉																

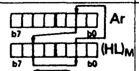
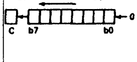
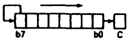
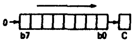
(to be continued)

Operation name	MNEMONICS	OP code	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
MULT	MLT ww **	11 101 101 01 ww1 100				S/D				2	17	wwHr×wwLr→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

(2) Rotate and Shift Instructions

Operation name	MNEMONICS	OP code	Addressing							Bytes	States	Operation	Flag					
			IMMED	EXT	IND	REG	REGI	IMP	REL				7	6	4	2	1	0
													S	Z	H	P/V	N	C
Rotate and Shift Data	RLA	00 010 111						S/D		1	3		·	·	R	·	R	↕
	RL g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
		00 010 g																
	RL (HL)	11 001 011					S/D			2	13		↕	↕	R	P	R	↕
		00 010 110																
	RL (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 010 110																
	RL (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 010 110																
	RLCA	00 000 111						S/D		1	3		·	·	R	·	R	↕
	RLC g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
		00 000 g																
	RLC (HL)	11 001 011					S/D			2	13		↕	↕	R	P	R	↕
		00 000 110																
	RLC (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 000 110																
	RLC (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 000 110																
	RLD	11 101 101						S/D		2	16		↕	↕	R	P	R	·
		01 101 111																
	RRA	00 011 111						S/D		1	3		·	·	R	·	R	↕
	RR g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
		00 011 g																
	RR (HL)	11 001 011					S/D			2	13		↕	↕	R	P	R	↕
		00 011 110																
	RR (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 011 110																
	RR (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 011 110																
	RRCA	00 001 111						S/D		1	3		·	·	R	·	R	↕
	RRC g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
		00 001 g																
	RRC (HL)	11 001 011					S/D			2	13		↕	↕	R	P	R	↕
		00 001 110																
	RRC (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 001 110																
	RRC (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 001 110																

(to be continued)

Operation name	MNEMONICS	OP code	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	RRD	11 101 101						S/D		2	16		↕	↕	R	P	R	·
and		01 100 111																
Shift	SLA g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
Data		00 100 g																
	SLA (HL)	11 001 011					S/D			2	13		↕	↕	R	P	R	↕
		00 100 110																
	SLA (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 100 110																
	SLA (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 100 110																
	SRA g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
		00 101 g																
	SRA (HL)	11 001 011					S/D			2	13		↕	↕	R	P	R	↕
		00 101 110																
	SRA (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 101 110																
	SRA (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 101 110																
	SRL g	11 001 011				S/D				2	7		↕	↕	R	P	R	↕
		00 111 g																
	SRL (HL)	11 001 011					S/D			2	3		↕	↕	R	P	R	↕
		00 111 110																
	SRL (IX+d)	11 011 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 111 110																
	SRL (IY+d)	11 111 101			S/D					4	19		↕	↕	R	P	R	↕
		11 001 011																
		〈 d 〉																
		00 111 110																

(3) Bit Manipulation Instructions

Operation name	MNEMONICS	OP code	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	·

(4) Arithmetic Instructions (16-bit)

Operation name	MNEMONICS	OP code	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
ADD	ADD HL,ww	00 ww1 001				S		D		1	7	$HL_R+ww_R \to HL_R$	·	·	X	·	R	↕
	ADD IX,xx	11 011 101 00 xx1 001				S		D		2	10	$IX_R+xx_R \to IX_R$	·	·	X	·	R	↕
	ADD IY,yy	11 111 101 00 yy1 001				S		D		2	10	$IY_R+yy_R \to IY_R$	·	·	X	·	R	↕
ADC	ADC HL,ww	11 101 101 01 ww1 010				S		D		2	10	$HL_R+ww_R+c \to HL_R$	↕	↕	X	V	R	↕
DEC	DEC ww	00 ww1 011				S/D				1	4	$ww_R-1 \to ww_R$	·	·	·	·	·	·
	DEC IX	11 011 101 00 101 011						S/D		2	7	$IX_R-1 \to IX_R$	·	·	·	·	·	·
	DEC IY	11 111 101 00 101 011						S/D		2	7	$IY_R-1 \to IY_R$	·	·	·	·	·	·
INC	INC ww	00 ww0 011				S/D				1	4	$ww_R+1 \to ww_R$	·	·	·	·	·	·
	INC IX	11 011 101 00 100 011						S/D		2	7	$IX_R+1 \to IX_R$	·	·	·	·	·	·
	INC IY	11 111 101 00 100 011						S/D		2	7	$IY_R+1 \to IY_R$	·	·	·	·	·	·
SBC	SBC HL,ww	11 101 101 01 ww0 010				S		D		2	10	$HL_R-ww_R-c \to HL_R$	↕	↕	X	V	S	↕

2 . Data Transfer Instructions

(1) 8-Bit Load

Operation name	MNEMONICS	OP code	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	↕	↕	R	IEF$_2$	R	·
	LD A,R	11 101 101 01 011 111						S/D		2	6	Rr→Ar	↕	↕	R	IEF$_2$	R	·
	LD A,(BC)	00 001 010					S	D		1	6	(BC)$_M$→Ar ①	·	·	·	·	·	·
	LD A,(DE)	00 011 010					S	D		1	6	(DE)$_M$→Ar	·	·	·	·	·	·
	LD A,(mn)	00 111 010 〈 n 〉 〈 m 〉		S				D		3	12	(mn)$_M$→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→(BC)$_M$	·	·	·	·	·	·
	LD (DE),A	00 010 010					D	S		1	7	Ar→(DE)$_M$	·	·	·	·	·	·
	LD (mn),A	00 110 010 〈 n 〉 〈 m 〉		D				S		3	13	Ar→(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)$_M$→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)$_M$→gr	·	·	·	·	·	·
	LD g,(IY+d)	11 111 101 01 g 110 〈 d 〉			S	D				3	14	(IY+d)$_M$→gr	·	·	·	·	·	·
	LD (HL),m	00 110 110 〈 m 〉	S				D			2	9	m→(HL)$_M$	·	·	·	·	·	·
	LD (IX+d),m	11 011 101 00 110 110 〈 d 〉 〈 m 〉	S		D					4	15	m→(IX+d)$_M$	·	·	·	·	·	·
	LD (IY+d),m	11 111 101 00 110 110 〈 d 〉 〈 m 〉	S		D					4	15	m→(IY+d)$_M$	·	·	·	·	·	·
	LD (HL),g	01 110 g				S	D			1	7	gr→(HL)$_M$	·	·	·	·	·	·
	LD (IX+d),g	11 011 101 01 110 g 〈 d 〉			D	S				3	15	gr→(IX+d)$_M$	·	·	·	·	·	·
	LD (IY+d),g	11 111 101 01 110 g 〈 d 〉			D	S				3	15	gr→(IY+d)$_M$	·	·	·	·	·	·

① In the case of R1 and Z Mask, interrupts are not sampled at the end of LD A, I or LD A, R.

(2) 16-Bit Load

Operation name	MNEMONICS	OP code	Addressing							Bytes	States	Operation	Flag					
			IMMED	EXT	IND	REG	REGI	IMP	REL				7	6	4	2	1	0
													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	$IYHr \rightarrow (mn+1)_M$ $IYLr \rightarrow (mn)_M$	·	·	·	·	·	·

(3) Block Transfer

Operation name	MNEMONICS	OP code	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
Block														③		②		
Transfer	CPD	11 101 101					S	S		2	12	$Ar-(HL)_M$	↕	↕	↕	↕	S	·
Search		10 101 001										$BC_R-1 \to BC_R$						
Data												$HL_R-1 \to HL_R$		③		②		
	CPDR	11 101 101					S	S		2	14	$BC_R \neq 0$ $Ar \neq (HL)_M$	↕	↕	↕	↕	S	·
		10 111 001									12	$BC_R=0$ or $Ar=(HL)_M$						
												$Ar-(HL\ )_M$						
												Q $BC_R-1 \to BC_R$						
												$HL_R-1 \to HL_R$						
												Repeat Q until						
												$Ar=(HL)_M$ or $BC_R=0$		③		②		
	CPI	11 101 101					S	S		2	12	$Ar-(HL)_M$	↕	↕	↕	↕	S	·
		10 100 001										$BC_R-1 \to BC_R$						
												$HL_R+1 \to HL_R$		③		②		
	CPIR	11 101 101					S	S		2	14	$BC_R \neq 0$ $Ar \neq (HL)_M$	↕	↕	↕	↕	S	·
		10 110 001									12	$BC_R=0$ or $Ar=(HL)_M$						
												$Ar-(HL\ )_M$						
												Q $BC_R-1 \to BC_R$						
												$HL_R+1 \to HL_R$						
												Repeat Q until						
												$Ar=(HL)_M$ or $BC_R=0$				②		
	LDD	11 101 101					S/D			2	12	$(HL)_M \to (DE)_M$	·	·	R	↕	R	·
		10 101 000										$BC_R-1 \to BC_R$						
												$DE_R-1 \to DE_R$						
												$HL_R-1 \to HL_R$						
	LDDR	11 101 101					S/D			2	14 ($BC_R \neq 0$)	$(HL)_M \to (DE)_M$	·	·	R	R	R	·
		10 111 000									12 ($BC_R=0$)	Q $BC_R-1 \to BC_R$						
												$DE_R-1 \to DE_R$						
												$HL_R-1 \to HL_R$						
												Repeat Q until						
												$BC_R=0$				②		
	LDI	11 101 101					S/D			2	12	$(HL)_M \to (DE)_M$	·	·	R	↕	R	·
		10 100 000										$BC_R-1 \to BC_R$						
												$DE_R+1 \to DE_R$						
												$HL_R+1 \to HL_R$						
	LDIR	11 101 101					S/D			2	14 ($BC_R \neq 0$)	$(HL)_M \to (DE)_M$	·	·	R	R	R	·
		10 110 000									12 ($BC_R=0$)	Q $BC_R-1 \to BC_R$						
												$DE_R+1 \to DE_R$						
												$HL_R+1 \to HL_R$						
												Repeat Q until						
												$BC_R=0$						

② $P/V=0$: $BC_R-1=0$
$P/V=1$: $BC_R-1 \neq 0$
③ $Z=1$: $Ar=(HL\ )_M$
$Z=0$: $Ar \neq (HL\ )_M$

(4) Stack and Exchange

Operation name	MNEMONICS	OP code	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	$zzLr \rightarrow (SP-2)_M$ $zzHr \rightarrow (SP-1)_M$ $SP_R-2 \rightarrow SP_R$	·	·	·	·	·	·
	PUSH IX	11 011 101 11 100 101						S/D		2	14	$IXLr \rightarrow (SP-2)_M$ $IXHr \rightarrow (SP-1)_M$ $SP_R-2 \rightarrow SP_R$	·	·	·	·	·	·
	PUSH IY	11 111 101 11 100 101						S/D		2	14	$IYLr \rightarrow (SP-2)_M$ $IYHr \rightarrow (SP-1)_M$ $SP_R-2 \rightarrow SP_R$	·	·	·	·	·	·
POP	POP zz	11 zz0 001				D		S		1	9	$(SP+1)_M \rightarrow zzHr$ ④ $(SP)_M \rightarrow zzLr$ $SP_R+2 \rightarrow SP_R$	·	·	·	·	·	·
	POP IX	11 011 101 11 100 001						S/D		2	12	$(SP+1)_M \rightarrow IXHr$ $(SP)_M \rightarrow IXLr$ $SP_R+2 \rightarrow SP_R$	·	·	·	·	·	·
	POP IY	11 111 101 11 100 001						S/D		2	12	$(SP+1)_M \rightarrow IYHr$ $(SP)_M \rightarrow IYLr$ $SP_R+2 \rightarrow SP_R$	·	·	·	·	·	·
Exchange	EX AF,AF'	00 001 000						S/D		1	4	$AF_R \leftrightarrow AF_R'$	·	·	·	·	·	·
	EX DE,HL	11 101 011						S/D		1	3	$DE_R \leftrightarrow HL_R$	·	·	·	·	·	·
	EXX	11 011 001						S/D		1	3	$BC_R \leftrightarrow BC_R'$ $DE_R \leftrightarrow DE_R'$ $HL_R \leftrightarrow HL_R'$	·	·	·	·	·	·
	EX (SP),HL	11 100 011						S/D		1	16	$Hr \leftrightarrow (SP+1)_M$ $Lr \leftrightarrow (SP)_M$	·	·	·	·	·	·
	EX (SP),IX	11 011 101 11 100 011						S/D		2	19	$IXHr \leftrightarrow (SP+1)_M$ $IXLr \leftrightarrow (SP)_M$	·	·	·	·	·	·
	EX (SP),IY	11 111 101 11 100 011						S/D		2	19	$IYHr \leftrightarrow (SP+1)_M$ $IYLr \leftrightarrow (SP)_M$	·	·	·	·	·	·

④ In the case of POP AF, Flag is written a current contents of the stack.

3 . Program Control Instructions

Operation name	MNEMONICS	OP code	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→(SP−1)$_M$	·	·	·	·	·	·
		〈 n 〉										PCLr→(SP−2)$_M$						
		〈 m 〉										mn→PC$_R$						
												SP$_R$−2→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≠0)	Br−1→Br	·	·	·	·	·	·
		〈j-2〉								2	7 (Br=0)	continue : Br=0						
												PC$_R$+j→PC$_R$: Br≠0						
	JP f,mn	11 f 010		D						3	6 (f : false)	mn→PC$_R$: f is true	·	·	·	·	·	·
		〈 n 〉								3	9 (f : true)	continue : f is false						
		〈 m 〉																
	JP mn	11 000 011		D						3	9	mn→PC$_R$	·	·	·	·	·	·
		〈 n 〉																
		〈 m 〉																
	JP (HL)	11 101 001					D			1	3	HL$_R$→PC$_R$	·	·	·	·	·	·
	JP (IX)	11 011 101					D			2	6	IX$_R$→PC$_R$	·	·	·	·	·	·
		11 101 001																
	JP (IY)	11 111 101					D			2	6	IY$_R$→PC$_R$	·	·	·	·	·	·
		11 101 001																
	JR j	00 011 000							D	2	8	PC$_R$+j→PC$_R$	·	·	·	·	·	·
		〈j-2〉																
	JR C,j	00 111 000							D	2	6	continue : C=0	·	·	·	·	·	·
		〈j-2〉								2	8	PC$_R$+j→PC$_R$: C=1						
	JR NC,j	00 110 000							D	2	6	continue : C=1	·	·	·	·	·	·
		〈j-2〉								2	8	PC$_R$+j→PC$_R$: C=0						
	JR Z,j	00 101 000							D	2	6	continue : Z=0	·	·	·	·	·	·
		〈j-2〉								2	8	PC$_R$+j→PC$_R$: Z=1						
	JR NZ,j	00 100 000							D	2	6	continue : Z=1	·	·	·	·	·	·
		〈j-2〉								2	8	PC$_R$+j→PC$_R$: Z=0						
Return	RET	11 001 001						D		1	9	(SP)$_M$→PCLr	·	·	·	·	·	·
												(SP+1)$_M$→PCHr						
												SP$_R$+2→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	12 (R0, R1)	(SP)$_M$→PCLr	·	·	·	·	·	·
		01 001 101									22 (Z)	(SP+1)$_M$→PCHr						
												SP$_R$+2→SP$_R$						
	RETN	11 101 101						D		2	12	(SP)$_M$→PCLr	·	·	·	·	·	·
		01 000 101										(SP+1)$_M$→PCHr						
												SP$_R$+2→SP$_R$						
												IEF$_2$→IEF$_1$						
Restart	RST v	11 v 111						D		1	11	PCHr→(SP−1)$_M$	·	·	·	·	·	·
												PCLr→(SP−2)$_M$						
												0→PCHr						
												v→PCLr						
												SP$_R$−2→SP$_R$						

4 . I/O Instructions

Operation name	MNEMONICS	OP code	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
INPUT	IN A,(m)	11 011 011 〈 m 〉						D	S	2	9	$(Am)_I \rightarrow Ar$ $m \rightarrow A_0 \sim A_7$ $Ar \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	IN g,(C)	11 101 101 01 g 000				D			S	2	9	$(BC)_I \rightarrow gr$ g=110 : Only the flags will change. $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	↕	↕	R	P	R	·
	IN0 g,(m) **	11 101 101 00 g 000 〈 m 〉				D			S	3	12	$(00m)_X \rightarrow gr$ g=110 : Only the flags will change. $m \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	↕	R	P	R	·
	IND	11 101 101 10 101 010					D		S	2	12	$(BC)_I \rightarrow (HL)_M$ $HL_R - 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	⑤ ↕	X	X	⑥ ↕	X
	INDR	11 101 101 10 111 010					D		S	2	14(Br≠0) 12(Br=0)	$Q\left\{\begin{array}{l}(BC)_I \rightarrow (HL)_M\\ HL_R - 1 \rightarrow HL_R\\ Br - 1 \rightarrow Br\end{array}\right.$ 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
	INI	11 101 101 10 100 010					D		S	2	12	$(BC)_I \rightarrow (HL)_M$ $HL_R + 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	⑤ ↕	X	X	⑥ ↕	X
	INIR	11 101 101 10 110 010					D		S	2	14(Br≠0) 12(Br=0)	$Q\left\{\begin{array}{l}(BC)_I \rightarrow (HL)_M\\ HL_R + 1 \rightarrow HL_R\\ Br - 1 \rightarrow Br\end{array}\right.$ 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

⑤ Z=1 : Br−1=0
Z=0 : Br−1≠0

⑥ N=1 : MSB of Data=1
N=0 : MSB of Data=0

(to be continued)

Operation name	MNEMONICS	OP code	Addressing							Bytes	States	Operation	Flag					
			IMMED	EXT	IND	REG	REGI	IMP	REL				7	6	4	2	1	0
													S	Z	H	P/V	N	C
OUTPUT	OUT (m),A	11 010 011 〈 m 〉						S	D	2	10	$Ar \rightarrow (Am)_I$ $m \rightarrow A_0 \sim A_7$ $Ar \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	OUT (C),g	11 101 101 01 g 001				S			D	2	10	$gr \rightarrow (BC)_I$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	OUT0 (m),g **	11 101 101 00 g 001 〈 m 〉				S			D	3	13	$gr \rightarrow (00m)_I$ $m \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	OTDM **	11 101 101 10 001 011					S		D	2	14	$(HL)_M \rightarrow (00C)_I$ $HL_R - 1 \rightarrow HL_R$ $Cr - 1 \rightarrow Cr$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	⑤ ↕	↕	P	⑥ ↕	↕
	OTDMR **	11 101 101 10 011 011					S		D	2	16(Br≠0) 14(Br=0)	$Q\left\{\begin{array}{l}(HL)_M \rightarrow (00C)_I \\ HL_R - 1 \rightarrow HL_R \\ Cr - 1 \rightarrow Cr \\ Br - 1 \rightarrow Br\end{array}\right.$ 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)	$Q\left\{\begin{array}{l}(HL)_M \rightarrow (BC)_I \\ HL_R - 1 \rightarrow HL_R \\ Br - 1 \rightarrow Br\end{array}\right.$ 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	$(HL)_M \rightarrow (BC)_I$ $HL_R + 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	⑤ ↕	X	X	⑥ ↕	X
	OTIR	11 101 101 10 110 011					S		D	2	14(Br≠0) 12(Br=0)	$Q\left\{\begin{array}{l}(HL)_M \rightarrow (BC)_I \\ HL_R + 1 \rightarrow HL_R \\ Br - 1 \rightarrow Br\end{array}\right.$ 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	$(00C)_I \cdot m$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	↕	S	P	R	R
	OTIM **	11 101 101 10 000 011					S		D	2	14	$(HL)_M \rightarrow (00C)_I$ $HL_R + 1 \rightarrow HL_R$ $Cr + 1 \rightarrow Cr$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	⑤ ↕	↕	P	⑥ ↕	↕
	OTIMR **	11 101 101 10 010 011					S		D	2	16(Br≠0) 14(Br=0)	$Q\left\{\begin{array}{l}(HL)_M \rightarrow (00C)_I \\ HL_R + 1 \rightarrow HL_R \\ Cr + 1 \rightarrow Cr \\ Br - 1 \rightarrow Br\end{array}\right.$ 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	$(HL)_M \rightarrow (BC)_I$ $HL_R - 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	⑤ ↕	X	X	⑥ ↕	X

⑤ Z=1 : Br−1=0
Z=0 : Br−1≠0

⑥ N=1 : MSB of Data=1
N=0 : MSB of Data=0

5 . Special Control Instructions

Operation name	MNEMONICS	OP code	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
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	·	·	·	·	·	·

⑦ Interrupts are not sampled at the end of DI or EI.

B. Instruction Summary in Alphabetical Order

** : Added new instructions to Z80

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)

(to be 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

(to be 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)

(to be 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)

(to be 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

(to be 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)

(to be 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	4 (R0, R1)	12 (R0, R1)
		10 (Z)	22 (Z)
RETN	2	4	12

(to be 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

(to be 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

(to be 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

C. Op-code Map

Table 1 1st op-code map
Instruction format : XX

				ww (LO=ALL)												LO=0~7				
				BC	DE	HL	SP									BC	DE	HL	AF	zz
				g (LO=0~7)												NZ	NC	PO	P	f
				B	D	H	(HL)	B	D	H	(HL)					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	LD g, s			NOTE1)	ADD A , s	SUB s	AND s	OR s	RET f				0
	C	0001	1	LD ww, mn												POP zz				1
	D	0010	2	LD (ww), A		LD (mn) , HL	LD (mn) , A									JP f, mn				2
	E	0011	3	INC ww												JP mn	OUT (m) , A	EX (SP) , HL	DI	3
	H	0100	4	INC g			NOTE1)									CALL f, mn				4
	L	0101	5	DEC g			NOTE1)									PUSH zz				5
	(HL)	0110	6	LD g, m			NOTE1)	NOTE2)			HALT	NOTE2)	NOTE2)	NOTE2)	NOTE2)	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	LD g, s				ADC A , s	SBC A , s	XOR s	CP s	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												Table2	IN A, (m)	EXDE, HL	EI	B
	H	1100	C	INC g												CALL f, mn				C
	L	1101	D	DEC g												CALL mn	NOTE3)	Table3	NOTE3)	D
	(HL)	1110	E	LD g, m				NOTE2)				NOTE2)	NOTE2)	NOTE2)	NOTE2)	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	
				C	E	L	A	C	E	L	A					Z	C	PE	M	f
				g (LO=8~F)												08H	18H	28H	38H	v
																LO=8~F				

NOTE1) (HL) replaces g.

2) (HL) replaces s.

3) If DDH is supplemented as 1st op-code 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 supplemented as 1st op-code 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 exception and note the followings.

If DDH is supplemented as 1st op-code for JP (HL), (IX) replaces (HL) as operand and JP (IX) is executed.

If FDH is supplemented as 1st op-code for JP (HL), (IY) replaces (HL) as operand and JP (IY) is executed.

Even if DDH or FDH is supplemented as 1st op-code for EX DE, HL, HL is not replaced and the instruction is regarded as illegal instruction.

Table 2 2nd op-code map
Instruction format : CB XX

								b (LO=0~7)												
								0	2	4	6	0	2	4	6	0	2	4	6	
			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	
g (HI=ALL)	B	0000	0	RLC g	RL g	SLA g		BIT b,g				RES b,g				SET b,g				0
	C	0001	1																	1
	D	0010	2																	2
	E	0011	3																	3
	H	0100	4																	4
	L	0101	5																	5
	(HL)	0110	6	NOTE1)	NOTE1)	NOTE1)		NOTE 1)				NOTE 1)				NOTE 1)				6
	A	0111	7																	7
	B	1000	8	RRC g	RR g	SRA g	SRL g	BIT b,g				RES b,g				SET b,g				8
	C	1001	9																	9
	D	1010	A																	A
	E	1011	B																	B
	H	1100	C																	C
	L	1101	D																	D
	(HL)	1110	E	NOTE1)	NOTE1)	NOTE1)	NOTE1)	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	
								1	3	5	7	1	3	5	7	1	3	5	7	
								b (LO=8~F)												

NOTE 1) If DDH is supplemented as 1st op-code for the instructions which have (HL) as operand in Table 2, the instructions are executed replacing (HL) with (IX+d).
If FDH is supplemented as 1st op-code for the instructions which have (HL) as operand in Table 2, the instructions are executed replacing (HL) with (IY+d).

Table 3 2nd op-code map
Instruction format : ED XX

						ww (LO=ALL)												
						BC	DE	HL	SP									
		g (LO=0~7)																
		B	D	H		B	D	H										
	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	
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	TSTI0 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	
		C	E	L	A	C	E	L	A									
		g (LO=8~F)																

D. Bus and Control Signal Condition in each Machine Cycle

*	(ADDRESS)	: invalid
Z	(DATA)	: high impedance.
**		: added new instructions to Z80

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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 SBC A,g AND g OR g XOR g CP g	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	Ti	*	Z	1	1	1	1	1	1	1
ADD A,m ADC A,m SUB m SBC A,m AND m OR m XOR m CP m	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	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) SBC A, (HL) AND (HL) OR (HL) XOR (HL) CP (HL)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	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) ADC A, (IY+d) SUB (IX+d) SUB (IY+d) SBC A, (IX+d)	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{\text{RD}}$	$\overline{\text{WR}}$	$\overline{\text{ME}}$	$\overline{\text{IOE}}$	$\overline{\text{LIR}}$	$\overline{\text{HALT}}$	ST
SBC A, (IY+d) AND (IX+d) AND (IY+d) OR (IX+d) OR (IY+d) XOR (IX+d) XOR (IY+d) CP (IX+d) CP (IY+d)	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4 $\sim MC_5$	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
BIT b,g	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
BIT b, (HL)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	3rd op-code	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 op-code Address	1st op-code	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 is false)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1

(to be continued)

Instruction	Machine 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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
CPI CPD	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4 $\sim MC_6$	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4 $\sim MC_8$	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4 $\sim MC_6$	TiTiTi	•	Z	1	1	1	1	1	1	1
CPL	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
DAA	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	Ti	•	Z	1	1	1	1	1	1	1
DI *1	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

*1 Interrupt request is not sampled.

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
DJNZ j (If Br≠0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	Ti *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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	Ti *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 *3	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
EX DE, HL EXX	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
EX AF, AF'	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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

(to be continued)

*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 op-code Address	1st op-code	0	1	0	1	0	1	0
	——	———	Next op-code Address	Next op-code	0	1	0	1	0	0	0
IM 0 IM 1 IM 2	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
INC g DEC g	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	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 op-code Address	1st op-code	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) DEC (IX+d) DEC (IY+d)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	Ti	•	Z	1	1	1	1	1	1	1
INC IX INC IY DEC IX DEC IY	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-dode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	Ti	•	Z	1	1	1	1	1	1	1

(to be 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 op-code Address	1st op-code	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 $A_0 \sim A_7$ A to $A_8 \sim A_{15}$	DATA	0	1	1	0	1	1	1
IN g,(C)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	DATA	0	1	1	0	1	1	1
INO g,(m)**	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 $A_0 \sim A_7$ 00H to $A_8 \sim A_{15}$	DATA	0	1	1	0	1	1	1
INI IND	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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≠0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 $\sim MC_6$	TiTi	*	Z	1	1	1	1	1	1	1
INIR INDR (If Br=0)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{\text{RD}}$	$\overline{\text{WR}}$	$\overline{\text{ME}}$	$\overline{\text{IOE}}$	$\overline{\text{LIR}}$	$\overline{\text{HALT}}$	ST
JP mn	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
JP (IX) JP (IY)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
JR j	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1

(to be 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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{\text{RD}}$	$\overline{\text{WR}}$	$\overline{\text{ME}}$	$\overline{\text{IOE}}$	$\overline{\text{LIR}}$	$\overline{\text{HALT}}$	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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 } *4 LD A,R LD I,A LD R,A	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
LD ww, mn	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1

*4 In the case of R1 and Z MASK, interrupt request is not sampled.

(to be continued)

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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	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

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{\text{RD}}$	$\overline{\text{WR}}$	$\overline{\text{ME}}$	$\overline{\text{IOE}}$	$\overline{\text{LIR}}$	$\overline{\text{HALT}}$	ST
LD (mn),ww	MC$_1$	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
LDIR LDDR (If BC$_R$≠0)	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	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$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	DE	DATA	1	0	0	1	1	1	1
MILT ww**	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	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$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
NOP	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
OUT (m),A	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	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$_1$T$_2$T$_3$	m to A$_0$~A$_7$ A to A$_8$~A$_{15}$	A	1	0	1	0	1	1	1

(to be 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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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
OUTO (m),g**	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 $A_0 \sim A_7$ 00H to $A_8 \sim A_{15}$	g	1	0	1	0	1	1	1
OTIM** OTDM**	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 $A_0 \sim A_7$ 00H to $A_8 \sim A_{15}$	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 $A_0 \sim A_7$ 00H to $A_8 \sim A_{15}$	DATA	1	0	1	0	1	1	1
	MC_6 $\sim MC_8$	TiTiTi	*	Z	1	1	1	1	1	1	1

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
OTIMR** OTDMR** (If Br=0)	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	Ti	*	Z	1	1	1	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1
	MC$_5$	T$_1$T$_2$T$_3$	C to $A_0 \sim A_7$ 00H to $A_8 \sim A_{15}$	DATA	1	0	1	0	1	1	1
	MC$_6$	Ti	*	Z	1	1	1	1	1	1	1
OUTI OUTD	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	BC	DATA	1	0	1	0	1	1	1
OTIR OTDR (If Br≠0)	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	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$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	BC	DATA	1	0	1	0	1	1	1
POP zz	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	SP	DATA	0	1	0	1	1	1	1
	MC$_3$	T$_1$T$_2$T$_3$	SP+1	DATA	0	1	0	1	1	1	1
POP IX POP IY	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

(to be 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 op-code Address	2nd op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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 op-code Address	1st op-code	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
RETI (R0, R1) RETN	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{\text{RD}}$	$\overline{\text{WR}}$	$\overline{\text{ME}}$	$\overline{\text{IOE}}$	$\overline{\text{LIR}}$	$\overline{\text{HALT}}$	ST
RETI (Z)	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0 *5 1	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	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 op-code Address	2nd op-code	0	1	0	1	0 *5 0	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 op-code Address	1st op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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

(to be continued)

*5 The upper and lower data show the state of $\overline{\text{LIR}}$ when $\overline{\text{IOC}}$=1 and $\overline{\text{IOC}}$=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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	3rd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	DATA	0	1	0	1	1	1	1
	MC_4 $\sim MC_7$	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2 $\sim MC_3$	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 op-code Address	1st op-code	0	1	0	1	0	1	0
SET b,g RES b,g	MC_1	$T_1T_2T_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd op-code Address	2nd op-code	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 op-code Address	1st op-code	0	1	0	1	0	1	0
	MC_2	T_1T_2 $_3$	2nd op-code Address	2nd op-code	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

(to be 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$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0

	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	3rd op-code Address	3rd op-code	0	1	0	1	0	1	1
	MC$_5$	T$_1$T$_2$T$_3$	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$_1$T$_2$T$_3$	IX+d IY+d	DATA	1	0	0	1	1	1	1
SLP**	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	–	–	7FFFFH	Z	1	1	1	1	1	0	1
TSTIO m**	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	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$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	Ti	*	Z	1	1	1	1	1	1	1
TST m**	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	T$_1$T$_2$T$_3$	1st operand Address	m	0	1	0	1	1	1	1
TST (HL)**	MC$_1$	T$_1$T$_2$T$_3$	1st op-code Address	1st op-code	0	1	0	1	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	2nd op-code Address	2nd op-code	0	1	0	1	0	1	1
	MC$_3$	Ti	*	Z	1	1	1	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	HL	DATA	0	1	0	1	1	1	1

INTERRUPT

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
$\overline{NMI}$	MC$_1$	T$_1$T$_2$T$_3$	Next op-code Address (PC)		0	1	0	1	0	1	0
	MC$_2$ $\sim$MC$_3$	TiTi	*	Z	1	1	1	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC$_5$	T$_1$T$_2$T$_3$	SP−2	PCL	1	0	0	1	1	1	1
$\overline{INT_0}$ MODE 0 (RST INSERTED)	MC$_1$	T$_1$T$_2$T$_W$ T$_W$T$_3$	Next op-code Address (PC)	1st op-code	1	1	1	0	0	1	0
	MC$_2$ $\sim$MC$_3$	TiTi	*	Z	1	1	1	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC$_5$	T$_1$T$_2$T$_3$	SP−2	PCL	1	0	0	1	1	1	1
$\overline{INT_0}$ MODE 0 (CALL INSERTED)	MC$_1$	T$_1$T$_2$T$_W$ T$_W$T$_3$	Next op-code Address (PC)	1st op-code	1	1	1	0	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	PC	n	0	1	0	1	1	1	1
	MC$_3$	T$_1$T$_2$T$_3$	PC+1	m	0	1	0	1	1	1	1
	MC$_4$	Ti	*	Z	1	1	1	1	1	1	1
	MC$_5$	T$_1$T$_2$T$_3$	SP−1	PC+2(H)	1	0	0	1	1	1	1
	MC$_6$	T$_1$T$_2$T$_3$	SP−2	PC+2(L)	1	0	0	1	1	1	1
$\overline{INT_0}$ MODE 1	MC$_1$	T$_1$T$_2$T$_W$ T$_W$T$_3$	Next op-code Address (PC)		1	1	1	0	0	1	0
	MC$_2$	T$_1$T$_2$T$_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC$_3$	T$_1$T$_2$T$_3$	SP−2	PCL	1	0	0	1	1	1	1
$\overline{INT_0}$ MODE 2	MC$_1$	T$_1$T$_2$T$_W$ T$_W$T$_3$	Next op-code Address (PC)	Vector	1	1	1	0	0	1	0
	MC$_2$	Ti	*	Z	1	1	1	1	1	1	1
	MC$_3$	T$_1$T$_2$T$_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC$_4$	T$_1$T$_2$T$_3$	SP−2	PCL	1	0	0	1	1	1	1
	MC$_5$	T$_1$T$_2$T$_3$	I, Vector	DATA	0	1	0	1	1	1	1
	MC$_6$	T$_1$T$_2$T$_3$	I, Vector+1	DATA	0	1	0	1	1	1	1

(to be continued)

Instruction	Machine Cycle	States	ADDRESS	DATA	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
$\overline{INT_1}$ $\overline{INT_2}$ Internal Interrupts	MC_1	$T_1T_2T_W$ T_WT_3	Next op-code 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

E-1. Request Acceptances in Each Operating Mode

Current Status Request		Normal Operation (CPU mode) (IOSTOP mode)	WAIT State	Refresh Cycle	Interrupt Acknowledge Cycle	DMA Cycle	BUS RELEASE mode	SLEEP mode	SYSTEM STOP mode
$\overline{WAIT}$		Acceptable	Acceptable	Not acceptable	Acceptable	Acceptable	Not acceptable	Not acceptable	Not acceptable
Refresh Request (Request of Refresh by the on-chip Refresh Controller)		Refresh cycle begins at the end of MC.	Not acceptable	Not acceptable	Refresh cycle begins at the end of MC.	Refresh cycle begins at the end of MC.	Not acceptable	Not acceptable	Not acceptable
$\overline{DREQ_0}$ $\overline{DREQ_1}$		DMA cycle begins at the end of MC.	DMA cycle begins at the end of MC.	Acceptable * Refresh cycle precedes. DMA cycle begins at the end of one MC.	Acceptable DMA cycle begins at the end of MC.	Acceptable Refer to "2.9 DMA Controller" for details.	Acceptable * After BUS RELEASE cycle, DMA cycle begins at the end of one MC.	Not acceptable	Not acceptable
$\overline{BUSREQ}$		Bus is released at the end of MC.	Not acceptable	Not acceptable	Bus is released at the end of MC.	Bus is released at the end of MC.	Continue BUS RELEASE mode.	Acceptable	Acceptable
Interrupt	$\overline{INT_0}$, $\overline{INT_1}$, $\overline{INT_2}$	Accepted after executing the current instruction.	Accepted after executing the current instruction	Not acceptable	Not acceptable	Not acceptable	Not acceptable	Acceptable Return from SLEEP mode to normal operation.	Acceptable Return from SYSTEM STOP mode to normal operation.
	Internal I/O Interrupt	↑	↑	↑	↑	↑	↑	↑	Not acceptable
	$\overline{NMI}$	↑	↑	↑	Not acceptable Interrupt acknowledge cycle precedes. $\overline{NMI}$ is accepted after executing the next instruction.	Acceptable DMA cycle stops.	↑	↑	Acceptable Return from SYSTEM STOP mode to normal operation.

NOTE) * : not acceptable when DMA Request is in level sense.
↑ : same as the above
MC : Machine Cycle

E-2. Request Priority

The HD64180 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.

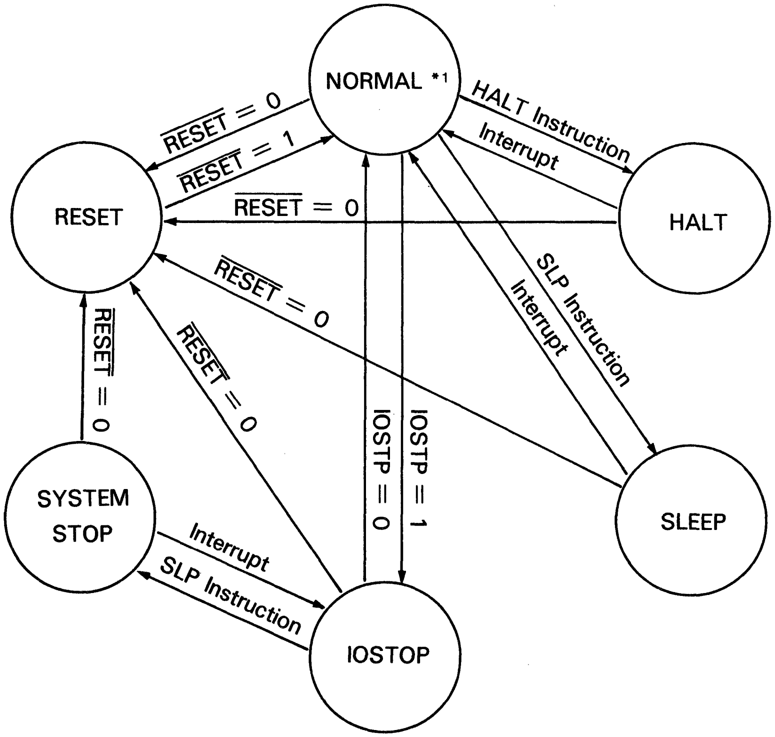
Type 3.

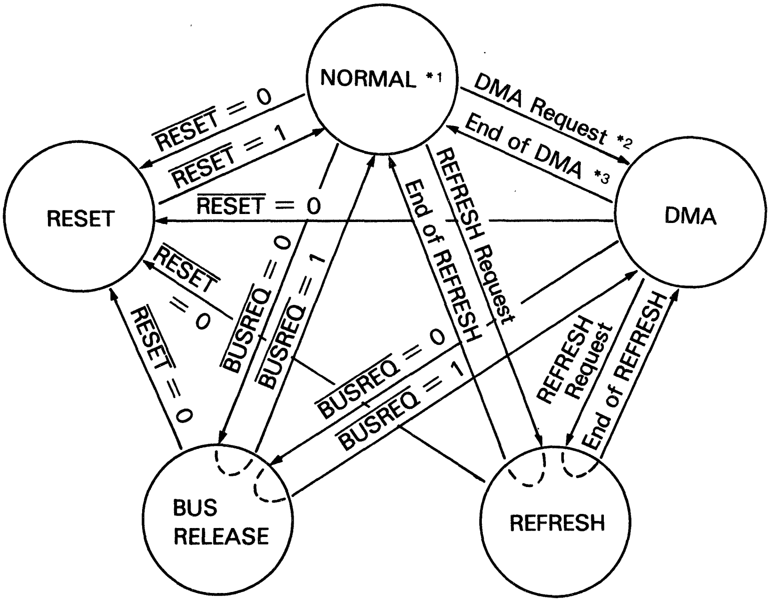
To be accepted in each instruction......... Interrupt Req.

Type 1, Type 2, and Type 3 requests priority is shown as follows.
highest priority Type 1 > Type 2 > Type 3 lowest priority
Each request priority in Type 2 is shown 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 "2.7 Interrupts" for each request priority in Type 3.

E-3. Operation Mode Transition


NORMAL *1
$\overline{\text{RESET}}$ = 0
$\overline{\text{RESET}}$ = 1
HALT Instruction
Interrupt
RESET
$\overline{\text{RESET}}$ = 0
HALT
$\overline{\text{RESET}}$ = 0
SLP Instruction
Interrupt
$\overline{\text{RESET}}$ = 0
$\overline{\text{RESET}}$ = 0
IOSTP = 0
IOSTP = 1
SYSTEM STOP
SLEEP
Interrupt
SLP Instruction
IOSTOP


NORMAL *1
DMA Request *2
End of DMA *3
$\overline{\text{RESET}}$ = 0
$\overline{\text{RESET}}$ = 1
RESET
$\overline{\text{RESET}}$ = 0
DMA
$\overline{\text{RESET}}$ = 0
$\overline{\text{RESET}}$ = 0
$\overline{\text{BUSREQ}}$ = 0
$\overline{\text{BUSREQ}}$ = 1
End of REFRESH
REFRESH Request
$\overline{\text{BUSREQ}}$ = 0
$\overline{\text{BUSREQ}}$ = 1
REFRESH Request
End of REFRESH
BUS RELEASE
REFRESH

NOTE) *1 NORMAL: CPU executes instructions normally in NORMAL mode.

*2 DMA request: DMA is requested in the following cases.

(1) $\overline{\text{DREQ}_0}$, $\overline{\text{DREQ}_1}$ = 0 (memory to/from (memory mapped) I/O DMA transfer)

(2) DEO = 1 (memory to/from memory DMA transfer)

*3 DMA end: DMA ends in the following cases.

(1) $\overline{\text{DREQ}_0}$, $\overline{\text{DREQ}_1}$ = 1 (memory to/from (memory mapped) I/O DMA transfer)

(2) BCRO, BCR1 = 0000H (all DMA transfers)

(3) $\overline{\text{NMI}}$ = 0 (all DMA transfers)

Other operation mode transitions

The following operation mode transitions are also possible.

1. $$\text{HALT} \rightleftarrows \begin{Bmatrix}\text{DMA}\\ \text{REFRESH}\\ \text{BUS RELEASE}\end{Bmatrix}$$

$$\text{IOSTOP} \rightleftarrows \begin{Bmatrix}\text{DMA}\\ \text{REFRESH}\\ \text{BUS RELEASE}\end{Bmatrix}$$

2. SLEEP ⇄ BUS RELEASE

SYSTEM STOP ⇄ BUS RELEASE

F-1. Status Signals

The following table shows pin outputs in each operating mode.

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	Op-code Fetch (1st op-code)	0	0	1	0	1	1	1	1	0	A	IN
	Op-code Fetch (except 1st op-code)	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 Acknowledge 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

F-2. Pin Status during RESET and Low Power Operation Modes

Symbol	Pin function	Pin status in each operation mode			
		RESET	SLEEP	IOSTOP	SYSTEM STOP
$\overline{WAIT}$	–	IN (N)	IN (N)	IN (A)	IN (N)
$\overline{BUSACK}$	–	1	OUT	OUT	OUT
$\overline{BUSREQ}$	–	IN (N)	IN (A)	IN (A)	IN (A)
$\overline{RESET}$	–	0	IN (A)	IN (A)	IN (A)
$\overline{NMI}$	–	IN (N)	IN (A)	IN (A)	IN (A)
$\overline{INT_0}$	–	IN (N)	IN (A)	IN (A)	IN (A)
$\overline{INT_1}$	–	IN (N)	IN (A)	IN (A)	IN (A)
$\overline{INT_2}$	–	IN (N)	IN (A)	IN (A)	IN (A)
ST	–	1	1	OUT	1
$A_0 \sim A_{17}$, A_{19}	–	Z	1	A	1
A_{18}/TOUT	A_{18}	Z	1	A	1
	TOUT	Z	OUT	H	H
$D_0 \sim D_7$	–	Z	Z	A	Z
$\overline{RTS_0}$	–	1	H	OUT	H
$\overline{CTS_0}$	–	IN (N)	IN (A)	IN (N)	IN (N)
$\overline{DCD_0}$	–	IN (N)	IN (A)	IN (N)	IN (N)
TXA_0	–	1	OUT	H	H
RXA_0	–	IN (N)	IN (A)	IN (N)	IN (N)
$CKA_0/\overline{DREQ_0}$	CKA_0 (internal clock mode)	Z	OUT	Z	Z
	CKA_0 (external clock mode)	Z	IN (A)	IN (N)	IN (N)
	$\overline{DREQ_0}$	Z	IN (N)	IN (A)	IN (N)
TXA_1	–	1	OUT	H	H
RXA_1	–	IN (N)	IN (A)	IN (N)	IN (N)
$CKA_1/\overline{TEND_0}$	CKA_1 (internal clock mode)	Z	OUT	Z	Z
	CKA_1 (external clock mode)	Z	IN (A)	IN (N)	IN (N)
	$\overline{TEND_0}$	Z	1	OUT	1
TXS	–	1	OUT	H	H
$RXS/\overline{CTS_1}$	RXS	IN (N)	IN (A)	IN (N)	IN (N)
	$\overline{CTS_1}$	IN (N)	IN (A)	IN (N)	IN (N)
CKS	CKS (internal clock mode)	Z	OUT	1	1
	CKS (external clock mode)	Z	IN (A)	Z	Z
$\overline{DREQ_1}$	–	IN (N)	IN (N)	IN (A)	IN (N)
$\overline{TEND_1}$	–	1	1	OUT	1
$\overline{HALT}$	–	1	0	OUT	0
$\overline{REF}$	–	1	1	OUT	1
$\overline{IOE}$	–	1	1	OUT	1
$\overline{ME}$	–	1	1	OUT	1
E	–	0	E clock output	←	←
$\overline{LIR}$	–	1	1	OUT	1
$\overline{WR}$	–	1	1	OUT	1
$\overline{RD}$	–	1	1	OUT	1
ϕ	–	ϕ clock output	←	←	←

1: HIGH 0: LOW A: Programmable Z: High Impedance
IN (A): Input (Active) IN (N): Input (Not active) OUT: Output
H: Holds the previous state
←: same as the left

G. Internal I/O Registers

By programming IOA7 and IOA6 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	MNEMONICS	ADDRESS	REMARKS
ASCI Control Register A Channel 0	: CNTLA0	0 0	

bit	MPE	RE	TE	$\overline{\text{RTS0}}$	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
- $\overline{\text{RTS0}}$ — 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	$\overline{\text{CTS}}$/ 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
- $\overline{\text{CTS}}$/PS — Clear To Send/Prescale
- MP — Multi Processor
- MPBT — Multi Processor Bit Transmit

* $\overline{\text{CTS}}$: Depending on the condition of $\overline{\text{CTS}}$ Pin.
PS : Cleared to 0.

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS
ASCI Control Register B Channel 1 : CNTLB1		0 3	

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

- Clock Source and Speed Select
- Divide Ratio
- Parity Even or Odd
- Clear To Send/Prescale
- Multi Processor
- 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)
000	ϕ÷ 160	ϕ÷ 640	ϕ÷ 480	ϕ÷ 1920
001	÷ 320	÷ 1280	÷ 960	÷ 3840
010	÷ 640	÷ 2560	÷ 1920	÷ 7680
011	÷ 1280	÷ 5120	÷ 3840	÷ 15360
100	÷ 2560	÷10240	÷ 7680	÷ 30720
101	÷ 5120	÷20480	÷15360	÷ 61440
110	÷10240	÷40960	÷30720	÷122880
111	External clock (frequency < ϕ ÷40)			

REGISTER	MNEMONICS	ADDRESS	REMARKS
ASCI Status Register Channel 0 : STAT0		0 4	

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

- Transmit Interrupt Enable
- Transmit Data Register Empty
- Data Carrier Detect
- Receive Interrupt Enable
- Framing Error
- Parity Error
- Over Run Error
- Receive Data Register Full

**

$\overline{CTS_0}$ Pin	TDRE
L	1
H	0

* $\overline{DCD_0}$: Depending on the condition of $\overline{DCD_0}$ Pin.

REGISTER	MNEMONICS	ADDRESS	REMARKS
ASCI Status Register Channel 1 : STAT1		0 5	

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

- Transmit Interrupt Enable
- Transmit Data Register Empty
- $\overline{CTS1}$ Enable
- Receive Interrupt Enable
- Framing Error
- Parity Error
- Over Run Error
- Receive Data Register Full

(to be continued)

REGISTER	MNEMONICS	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

Speed Select

Transmit Enable

Receive Enable

End Interrupt Enable

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 < ÷ 20)

REGISTER	MNEMONICS	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

Timer Down Count Enable 1,0

Timer Output Control 1,0

Timer Interrupt Enable 1,0

Timer Interrupt Flag 1,0

TOC1,0	A_{18}/TOUT
00	Inhibited
01	Toggle
10	0
11	1

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS
Timer Data Register Channel 1L	: TMDR1L	1 4	
Timer Data Register Channel 1H	: TMDR1H	1 5	
Timer Reload Register Channel 1L	: RLDR1L	1 6	
Timer Reload Register Channel 1H	: RLDR1H	1 7	
Free Running Counter	: FRC	1 8	read only
DMA Source Address Register Channel 0L	: SAR0L	2 0	
DMA Source Address Register Channel 0H	: SAR0H	2 1	
DMA Source Address Register Channel 0B	: SAR0B	2 2	Bits 0-2 (3) are used for SAR0B. A_{19}*, A_{18}, A_{17}, A_{16} | DMA Transfer Request X X 0 0 | $\overline{\text{DREQ}_0}$ (external) X X 0 1 | RDR0 (ASCI0) X X 1 0 | RDR1 (ASCI1) X X 1 1 | Not Used
DMA Destination Address Register Channel 0L	: DAR0L	2 3	
DMA Destination Address Register Channel 0H	: DAR0H	2 4	
DMA Destination Address Register Channel 0B	: DAR0B	2 5	Bits 0-2 (3) are used for DAR0B. A_{19}*, A_{18}, A_{17}, A_{16} | DMA Transfer Request X X 0 0 | $\overline{\text{DREQ}_0}$ (external) X X 0 1 | TDR0 (ASCI0) X X 1 0 | TDR1 (ASCI1) X X 1 1 | Not Used
DMA Byte Count Register Channel 0L	: BCR0L	2 6	
DMA Byte Count Register Channel 0H	: BCR0H	2 7	
DMA Memory Address Register Channel 1L	: MAR1L	2 8	
DMA Memory Address Register Channel 1H	: MAR1H	2 9	
DMA Memory Address Register Channel 1B	: MAR1B	2 A	Bits 0-2 (3) are used for MAR1B.
DMA I/O Address Register Channel 1L	: IAR1L	2 B	
DMA I/O Address Register Channel 1H	: IAR1H	2 C	

(to be continued)

* In the R1 and Z Mask, these DMAC registers are expanded from 4 bits to 3 bits in the package version of CP-68 and FP-80.

REGISTER	MNEMONICS	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	

bit	DE1	DE0	$\overline{\text{DWE1}}$	$\overline{\text{DWE0}}$	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

DME: DMA Master Enable

DIE1, DIE0: DMA Interrupt Enable 1,0

$\overline{\text{DWE1}}$, $\overline{\text{DWE0}}$: DMA Enable Bit Write Enable 1,0

DE1, DE0: DMA Enable ch 1,0

REGISTER	MNEMONICS	ADDRESS	REMARKS
DMA Mode Register	: DMODE	3 1	

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	

MMOD: Memory MODE Select

SM1, SM0: Ch 0 Source Mode 1,0

DM1, DM0: Ch 0 Destination Mode 1, 0

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

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS
DMA/WAIT Control Register	: DCNTL	32	

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

- DIM1,0: DMA Ch 1 I/O Memory Mode Select
- DMS1,0: $\overline{DREQi}$ Select, i = 1,0
- IWI1,0: I/O Wait Insertion
- MWI1,0: Memory Wait Insertion

MWI1,0	The number of wait states
00	0
01	1
10	2
11	3

IWI1,0	The number of wait states
00	0
01	2
10	3
11	4

DMSi	Sense
1	Edge sense
0	Level sense

DIM1,0	Transfer Mode	Address Increment/Decrement	
00	M→I/O	MAR1+1	IAR1 fixed
01	M→I/O	MAR1−1	IAR1 fixed
10	I/O→M	IAR1 fixed	MAR1+1
11	I/O→M	IAR1 fixed	MAR1−1

REGISTER	MNEMONICS	ADDRESS	REMARKS
Interrupt Vector Low Register	: IL	33	

bit	IL7	IL6	IL5	–	–	–	–	–
during RESET	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W					

- IL7–IL5: Interrupt Vector Low

REGISTER	MNEMONICS	ADDRESS	REMARKS
INT/TRAP Control Register	: ITC	34	

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

- ITE2,1,0: $\overline{INT}$ Enable 2,1,0
- UFO: Undefined Fetch Object
- TRAP: TRAP

REGISTER	MNEMONICS	ADDRESS	REMARKS
Refresh Control Register	: RCR	36	

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

- CYC1,0: Cycle Select
- REFW: Refresh Wait State
- REFE: Refresh Enable

CYC1,0	Interval of Refresh Cycle
00	10 States
01	20
10	40
11	80

(to be continued)

<table>
<tr><th>REGISTER</th><th>MNEMONICS</th><th>ADDRESS</th><th>REMARKS</th></tr>
<tr><td>MMU Common Base Register</td><td>: CBR</td><td>3 8</td><td>
<table>
<tr><td>bit</td><td>CB7*</td><td>CB6</td><td>CB5</td><td>CB4</td><td>CB3</td><td>CB2</td><td>CB1</td><td>CB0</td></tr>
<tr><td>during RESET</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
</table>
MMU Common Base Register</td></tr>
<tr><td>MMU Bank Base Register</td><td>: BBR</td><td>3 9</td><td>
<table>
<tr><td>bit</td><td>BB7*</td><td>BB6</td><td>BB5</td><td>BB4</td><td>BB3</td><td>BB2</td><td>BB1</td><td>BB0</td></tr>
<tr><td>during RESET</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
</table>
MMU Bank Base Register</td></tr>
<tr><td>MMU Common/Bank Area Register</td><td>: CBAR</td><td>3 A</td><td>
<table>
<tr><td>bit</td><td>CA3</td><td>CA2</td><td>CA1</td><td>CA0</td><td>BA3</td><td>BA2</td><td>BA1</td><td>BA0</td></tr>
<tr><td>during RESET</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
</table>
MMU Bank Area Register
MMU Common Area Register</td></tr>
<tr><td>Operation Mode Control Register (HD64180Z Mask only)</td><td>: OMCR</td><td>3 E</td><td>
<table>
<tr><td>bit</td><td>LIRE</td><td>$\overline{\text{LIRTE}}$</td><td>$\overline{\text{IOC}}$</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>during RESET</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>R/W</td><td>R/W</td><td>W</td><td>R/W</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
I/O Compatibility
$\overline{\text{LIR}}$ Temporary Enable
$\overline{\text{LIR}}$ Enable</td></tr>
<tr><td>I/O Control Register</td><td>: ICR</td><td>3 F</td><td>
<table>
<tr><td>bit</td><td>IOA7</td><td>IOA6</td><td>IOSTP</td><td>–</td><td>–</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>during RESET</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
I/O Stop
I/O Address</td></tr>
</table>

* In the R1 and Z Mask, these MMU registers are expanded from 7 bits to 8 bits in the package version of CP-68, and FP-80.

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.