ELE 3230
Microprocessors and Computer Systems

Chapter 13
Interrupts

(PIC materials, D. Hall Ch8, Brey Ch11)

Interrupts Revisited

▌ Three basic categories of interrupts:

1. Software interrupts: generated by INT *interrupt-vector*

2. Hardware interrupts: generated by asserting the NMI or INTR pin

3. Automatic interrupts: generated by certain error conditions (eg. divide by 0), or at the end of every instruction if the TRAP flag is set in the FLAGS register.

▌ Refer to earlier notes on software interrupts using the INT instruction to review the interrupt process (push FLAGS, clear IF, TF, push CS, push IP, load interrupt vector into CS:IP). All interrupts are processed in the same way.

Interrupts Revisited (cont.)

- Hardware interrupts are either maskable (ie. INTR only recognized if the interrupt flag in the FLAGS register is set) or non- maskable (ie. NMI is processed irrespective of the value of the interrupt flag).

- Hardware interrupts allow peripherals to be serviced by the microprocessor <u>only when they need attention</u> - the microprocessor can perform other tasks when the peripheral does not need the dedicated attention of microprocessor.

- (Read chapter 11 again about Interrupt).

Interrupts Instruction

Mnemonic	Meaning	Format	Operation	Flags Affected
CLI	Clear interrupt flag	CLI	$0 \rightarrow (IF)$	IF
STI	Set interrupt flag	STI	$1 \rightarrow (IF)$	IF
INT n	Type n software interrupt	INT n	$(Flags) \rightarrow ((SP-2)$ $0 \rightarrow TF, IF$ $(CS) \rightarrow ((SP)-4)$ $(2+4 \cdot n) \rightarrow (CS)$ $(IP) \rightarrow ((SP)-6)$ $(4 \cdot n) \rightarrow (IP)$	TF, IF
IRET	Interrupt return	IRET	$((SP)) \rightarrow (IP)$ $((SP)+2) \rightarrow (CS)$ $((SP)+4) \rightarrow (Flags)$ $(SP)+6 \rightarrow (SP)$	All
INTO	Interrupt on overflow	INTO	INT 4 steps	TF, IF
HLT	Halt	HLT	Wait for an external Interrupt or reset to occur	None
WAIT	Wait	WAIT	Wait for $\overline{TEST}$ input to go active	None

Usually connected to $\overline{BUSY}$ of 8087

Some Special Types of Interrupts in the Intel Series

- Type 0 : Divide error interrupt

  Interrupt number 0 (interrupt vector is located at memory locations 00000 to 0003) is generated when an attempt is made to divide by zero. The interrupt is generated automatically.

- Type 1: Single-Step Interrupts

  When bit 8 of the FLAGS register (trap flag) is set to 1, an interrupt number 1 is generated after the execution of every instruction. When the microprocessor enters the interrupt service routine, the T flag is automatically cleared (otherwise it would not be able to go beyond the first instruction of the interrupt service routine!).

Some Special Types of Interrupts in the Intel Series (cont.)
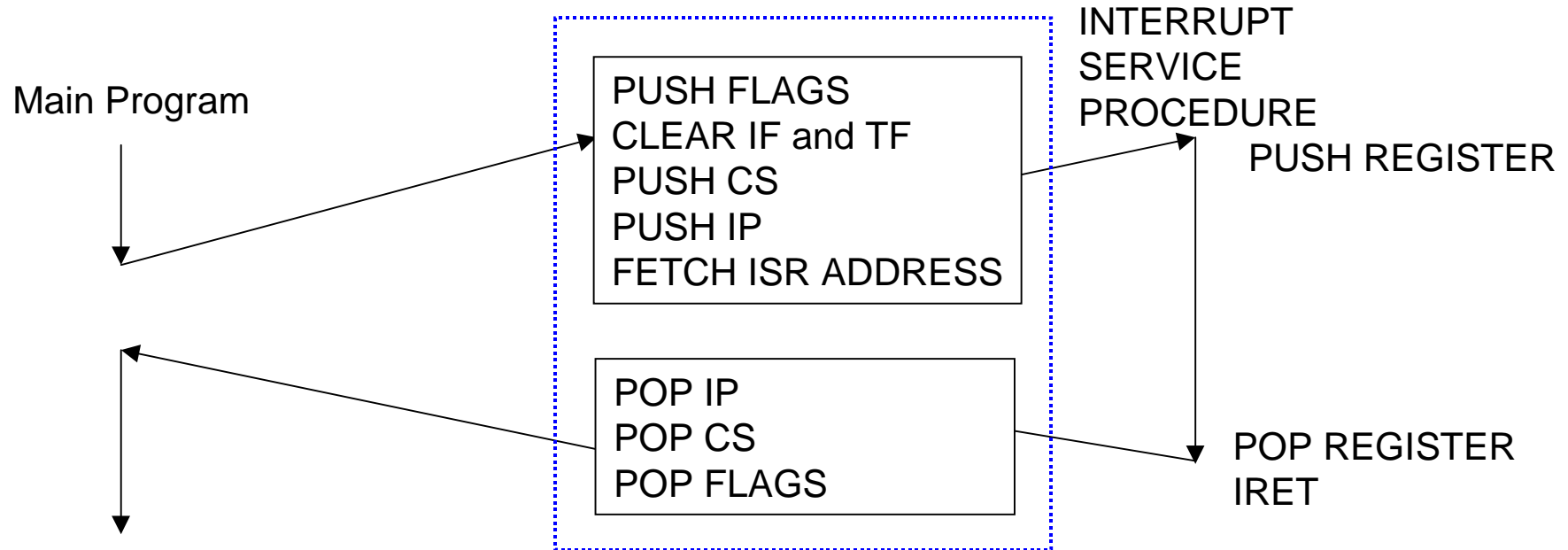
■ Type 3: Breakpoint (One-byte) Interrupt

Interrupt number 3 has a compact single byte opcode (CCh) which may be inserted anywhere in a program to generate a breakpoint in a program. Clearing the breakpoint in a program is accomplished by simply copying back the original instruction

■ Type 6: Undefined Opcode Interrupt

When the microprocessor encounters an opcode which it does not recognize, it generates interrupt number 6 (generated only in 80386 and later microprocessors).

Interrupt Response Sequence

1. PUSH Flag Register

2. Clear IF and TF

3. PUSH CS

4. PUSH IP

5. Fetch Interrupt vector contents and place into both IP and CS to start the Interrupt Service Procedure (ISP)

Main Program

INTERRUPT
SERVICE
PROCEDURE

PUSH FLAGS
CLEAR IF and TF
PUSH CS
PUSH IP
FETCH ISR ADDRESS

PUSH REGISTER

POP IP
POP CS
POP FLAGS

POP REGISTER
IRET

8088 Hardware Interrupts

■ Hardware interrupts are very efficient in handling peripheral devices, particularly I/O operations, since the processor can perform other tasks and only services the peripheral when required.

■ The NMI is usually reserved for the most urgent type of interrupt (eg. imminent power failure of a peripheral), whereas the INTR is used for "normal" interrupts (where the peripheral can wait if necessary until the interrupt flag is set).

NMI ←——— ⎫
INTR ←——— ⎬ Interrupt input

INTA ———→ Interrupt output
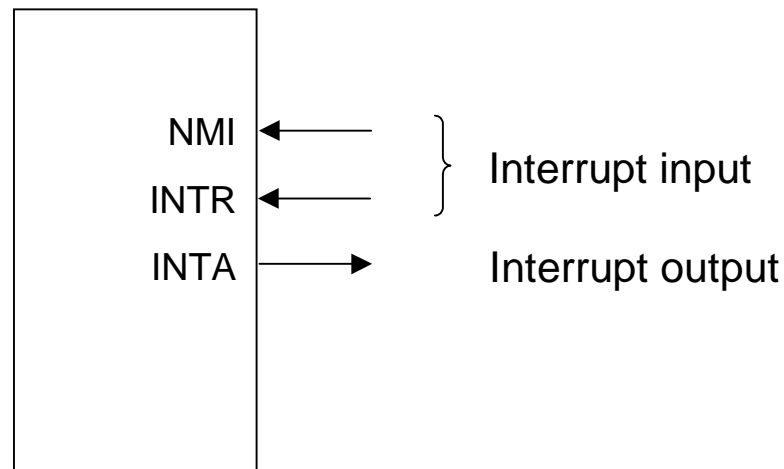
8088 Hardware Interrupts (cont.)

▌ Asserting the NMI pin generates interrupt number 2 (the interrupt vector is located at memory location 0008, new offset address IP is stored in 00008 and 00009; new segment address CS is stored in 0000Ah and 0000Bh). NMI is edge triggered (the interrupt is generated only at the 0-to-1 transition).

▌ When more than one peripheral are connected to the NMI pin, the interrupt service routine may need to check which peripheral generated the NMI (eg. by *polling* the possible sources of the NMI).

▌ Interrupt request pin INTR is level sensitive - it must be held at logic 1 until it is acknowledged by INTA. INTR is automatically disabled when the microprocessor is already servicing an INTR, and INTR is re-enabled at the end of the interrupt service routine.

▌ The interrupt number generated by an INTR is read from the data bus.

Interrupt Priority

■ When different type of interrupt (ie. software, NMI, INTR or automatic interrupts) occur at the same time, the one with the highest priority is handled first. Intel microprocessors use the following order of priority:

Interrupt Type	Priority
divide error interrupt, INT n, INT0 NMI INTR TRAP flag (single step)	highest ↕ lowest

(See Reference Note on Interrupt)

Interrupt Priority (cont.)

▮ If several INTR are generated from different peripherals simultaneously, it is necessary to decide their priority and send the INTA signal only to the <u>highest</u> priority peripheral. (If more than one peripheral receive an INTA at the same time they will both try to put the interrupt number onto the data bus simultaneously).

▮ Two different methods can be used to establish the priority of interrupt requests from different peripherals

1. Polling (and daisy chaining)
2. Interrupt priority management hardware

Priority Allocation by Polling and Daisy-Chaining

▌ Polling involves asking each peripheral, in a predetermined order, whether it needs attention from the microprocessor. The first peripheral which responds "yes" is served by the appropriate routine.

▌ Daisy-chaining is a method of implementing the polling scheme by hardware. The INTA signal passes along a series of peripherals from one peripheral to the next only if the peripheral is not requesting an interrupt. Hence the first peripheral in the daisy-chain has the highest priority and the last peripheral has the lowest priority.

Priority Allocation by Daisy-Chaining
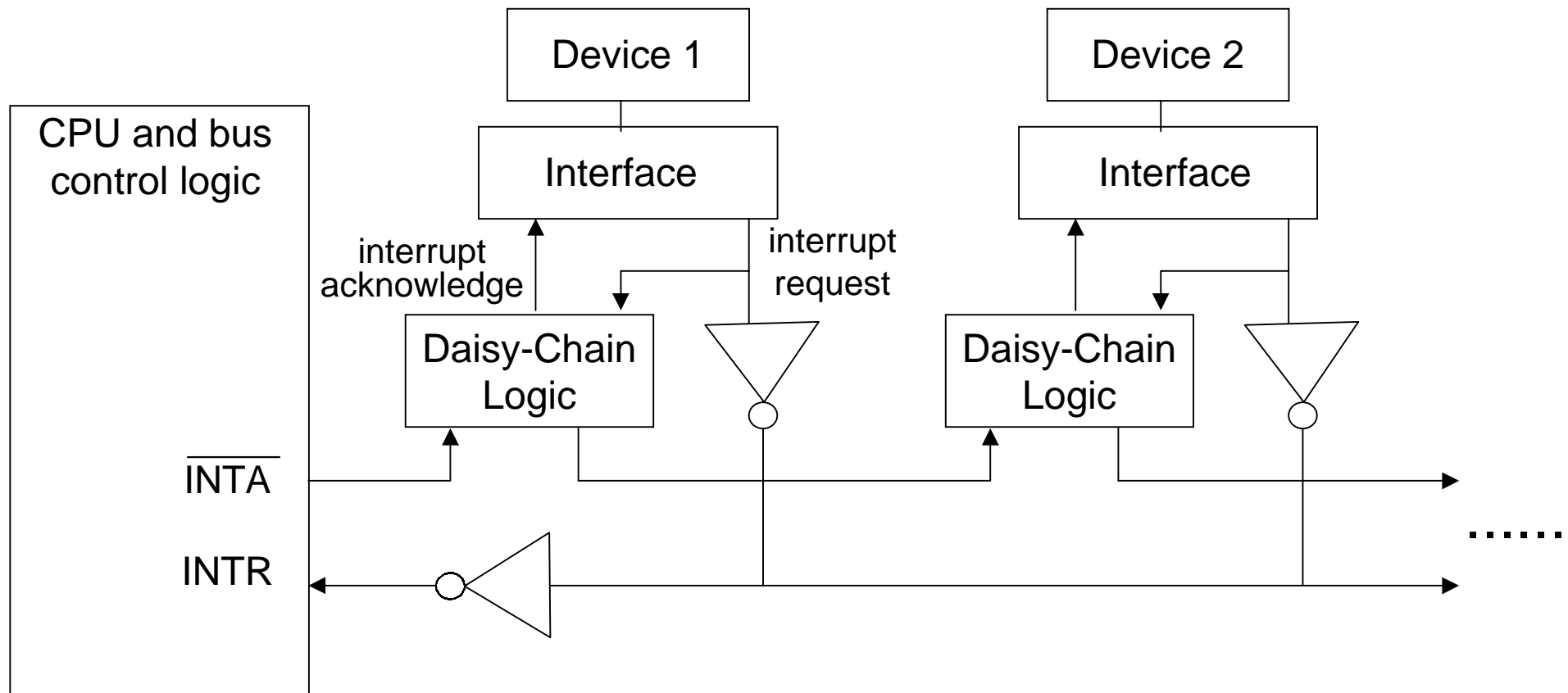
- Daisy chaining may be combined with software polling to determine which routine is needed by the peripheral

Interrupt Priority Management

▌ The daisy chain method suffers the disadvantage of being limited in the number of devices which could be chained together because of the delays in passing the $\overline{INTA}$ through the chain. The microprocessor expects the interrupt number to be placed on the data bus within a certain time after the $\overline{INTA}$ is sent out. Too long a delay will lead to uncertain errors.

▌ Interrupt requests from different peripherals may be sent in parallel to a PRIORITY ENCODER such as the 74LS148.

Interrupt Priority Management (cont.)

- The priority encoder generates a binary coded output which is sent to a PRIORITY DECODER (eg. 74LS138) that determines which request is acknowledged. The binary number generated by the priority encoder may also be used to put an interrupt number onto the data bus.

- A Programmable interrupt controller (8259A) is usually used in practical systems to determine the priority of interrupts.

- The 8259A programmable interrupt controller assigns the priority of up to 8 vectored interrupt sources to be connected to the INTR pin. The 8259A may be cascaded (one master 8259A and eight slave 8259A) to provide 64 interrupt lines).

The 8259A Programmable Interrupt Controller (PIC)

- 8259A is a 28-pin integrated circuit which was designed specifically for the 8088/8086 microprocessors.

11	D0	18 IR0
10	D1	19 IR1
9	D2	20 IR2
8	D3	21 IR3
7	D4	22 IR4
6	D5	23 IR5
5	D6	24 IR6
4	D7	25 IR7
27	A0	
1	$\overline{CS}$	
3	$\overline{RD}$	
2	$\overline{WR}$	
16	SP/$\overline{EN}$	
17	INT	12 CAS0
26	$\overline{INTA}$	13 CAS1
		15 CAS2

The 8259A Programmable Interrupt Controller (cont.)

- The 28 pins of the 8259A include:

  1. Data pins D0-D7 - connected to the data bus to allow programming
  2. IR0-IR7 - 8 interrupt inputs
  3. CAS0-CAS2 - cascade lines (used in multiple 8259A systems)
  4. SP/$\overline{\text{EN}}$ - functions as data buffer enable output (when buffered mode) or as an input to program the 8259A as a master or slave
  5. A0 - input which selects different command words in the 8259A

- Usually IR0 has the highest priority and IR7 the least priority.

- Fully nested interrupts are supported (higher priority interrupts may interrupt the interrupt service routine of a lower priority interrupt) if bit 4 of ICW is set. (Refer to data sheets for further details.)

8259A Interrupt Controllers In Cascade
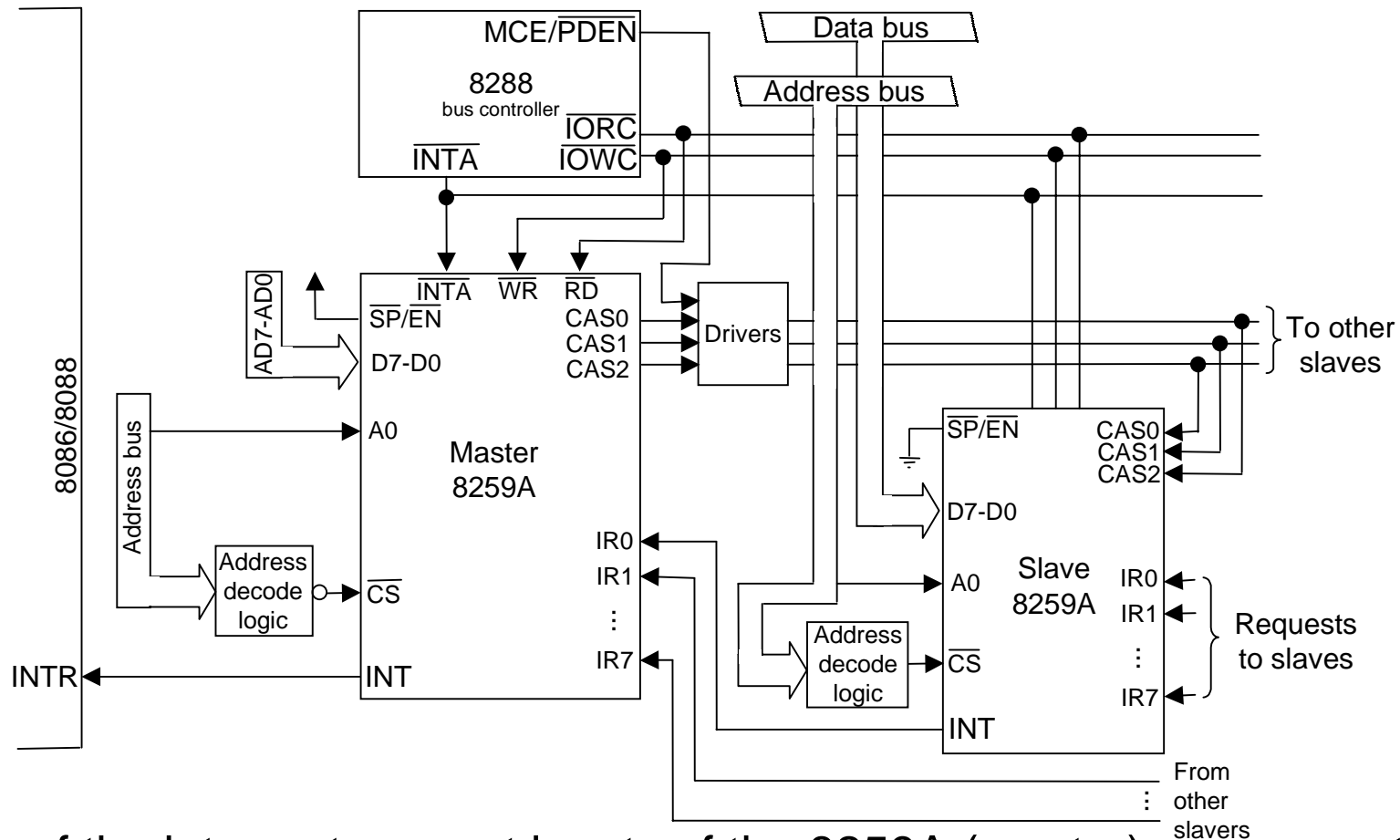
Each of the interrupt request inputs of the 8259A (master) may accept the INT output from another (slave) 8259A device, thus increasing the number of interrupt request lines beyond 8.

8259A Pin Description

Symbol	Pin No.	Type	Name and Function
V_{CC}	28	I	SUPPLY: +5V Supply
GND	14	I	GROUND
$\overline{CS}$	1	I	CHIP SELECT: A LOW on this pin enables $\overline{RD}$ and $\overline{WR}$ communication between the CPU and the 8259A. INTA functions are independent of $\overline{CS}$.
$\overline{WR}$	2	I	WRITE: A LOW on this pin when $\overline{CS}$ is low enables the 8259A to accept command words from the CPU.
$\overline{RD}$	3	I	READ: A LOW on this pin when $\overline{CS}$ is low enables the 8259A to release data onto the data bus for the CPU.
D_7-D_0	4 -11	I/O	BIDIRECTIONAL DATA BUS: Control, status and interrupt-vector information is transferred via this bus.
CAS_0-CAS_2	12,13,15	I/O	CASCADE LINES: The CAS lines form a private 8259A bus to control a multiple 8259A structure. These pins are outputs for a master 8259A and inputs for a slave 8259A.

8259A Pin Description (cont.)

SP/$\overline{EN}$	16	I/O	SLAVE PROGRAM/ENABLE BUFFER: This is a dual function pin. When in the Buffered Mode it can be used as an output to control buffer transceivers (EN). When not in the buffered mode it is used as an input to designate a master (SP - 1) or slave (SP - 0).
INT	17	O	INTERRUPT: This pin goes high whenever a valid interrupt request is asserted. It is used to interrupt the CPU, thus it is connected to the CPU's interrupt pin.
IR_0-IR_7	18-25	I	INTERRUPT REQUESTS: Asynchronous inputs. An interrupt request is executed by raising an IR input (low to high), and holding it high until it is acknowledged (*Edge Triggered Mode*), or just by a high level on an IR input (*Level Triggered Mode*).
$\overline{INTA}$	26	I	INTERRUPT ACKNOWLEDGE: This pin is used to enable 8259A interrupt-vector data onto the data bus by a sequence of interrupt acknowledge pulses issued by the CPU
A_0	27	I	A0 ADDRESS LINE: This pin acts in conjunction with the $\overline{CS}$, $\overline{WR}$, and $\overline{RD}$ pins. It is used by the 8259A to decipher various Command Words the CPU writes and status the CPU wishes to read.

Internal Architecture of 8259A
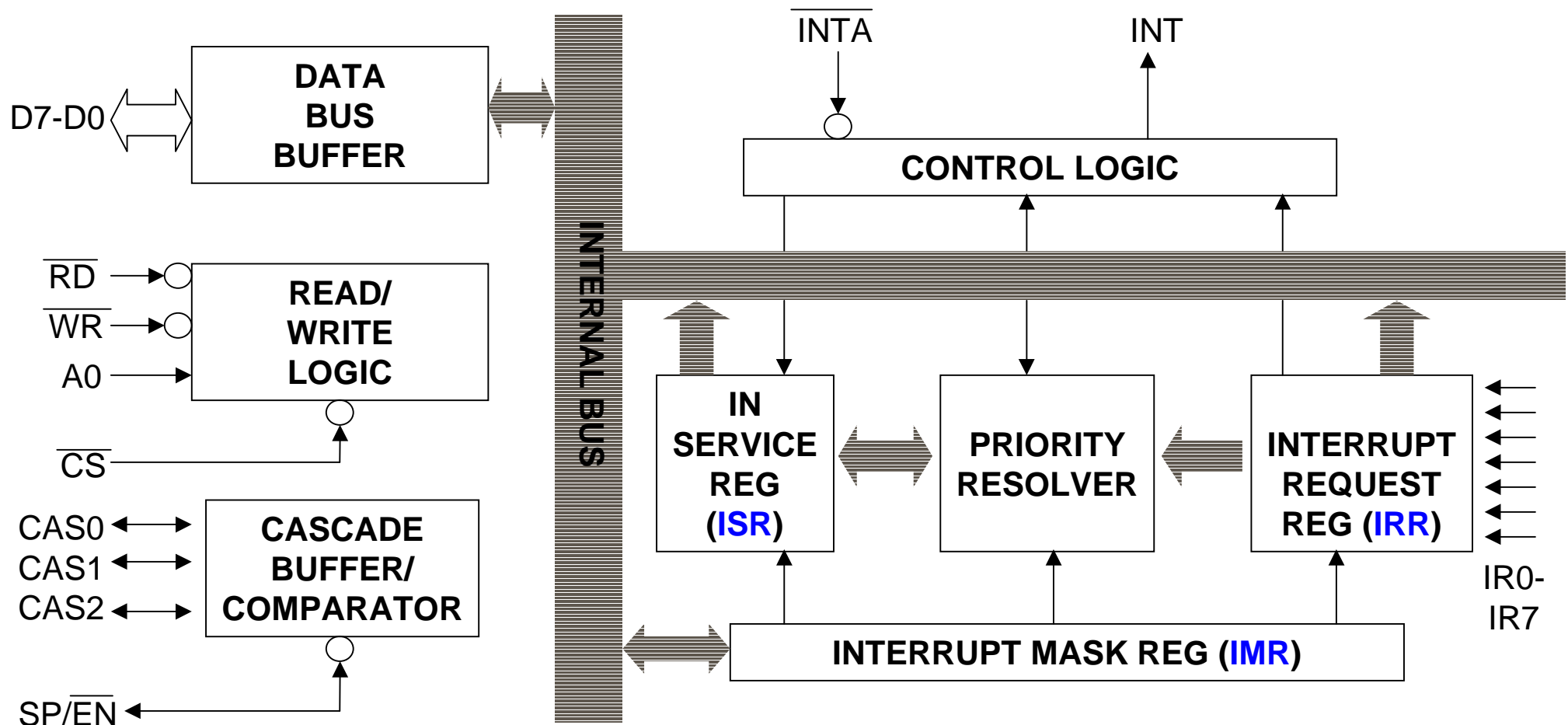
Internal Block Diagram

(D Hall Fig.8-27 or

W Triebel Fig. 9.11)

Internal Architecture of 8259A

- **Data bus buffer**: bidirectional three state buffer through which MPU can access 8259A's internal registers.

- **R/W Logic**: accept input $\overline{RD}$,$\overline{WR}$,A0, and $\overline{CS}$ to control the direction, timing, and source/destination of data transfer.

- **IMR**: used to mask out individual interrupt request input (IR0-IR7).

- **IRR**: stores the current status of the interrupt request input.

- **Priority resolver**: determine the interrupt priority of the active interrupt inputs.

- **ISR**: stores the interrupt level that is presently being served.

- **Cascade buffer/ comparator**: provides the interface between master and slave 8259A. Each slave has an ID code stored here.

8259A and 8088 Program Flow For IR4 Followed by IR2

MAINLINE
INITIALIZE 8259A
UNMASK IR2, IR4
STI

IR4
PROCEDURE
STI

IR2
PROCEDURE
STI

EOI
COMMAND
IRET

EOI
COMMAND
IRET
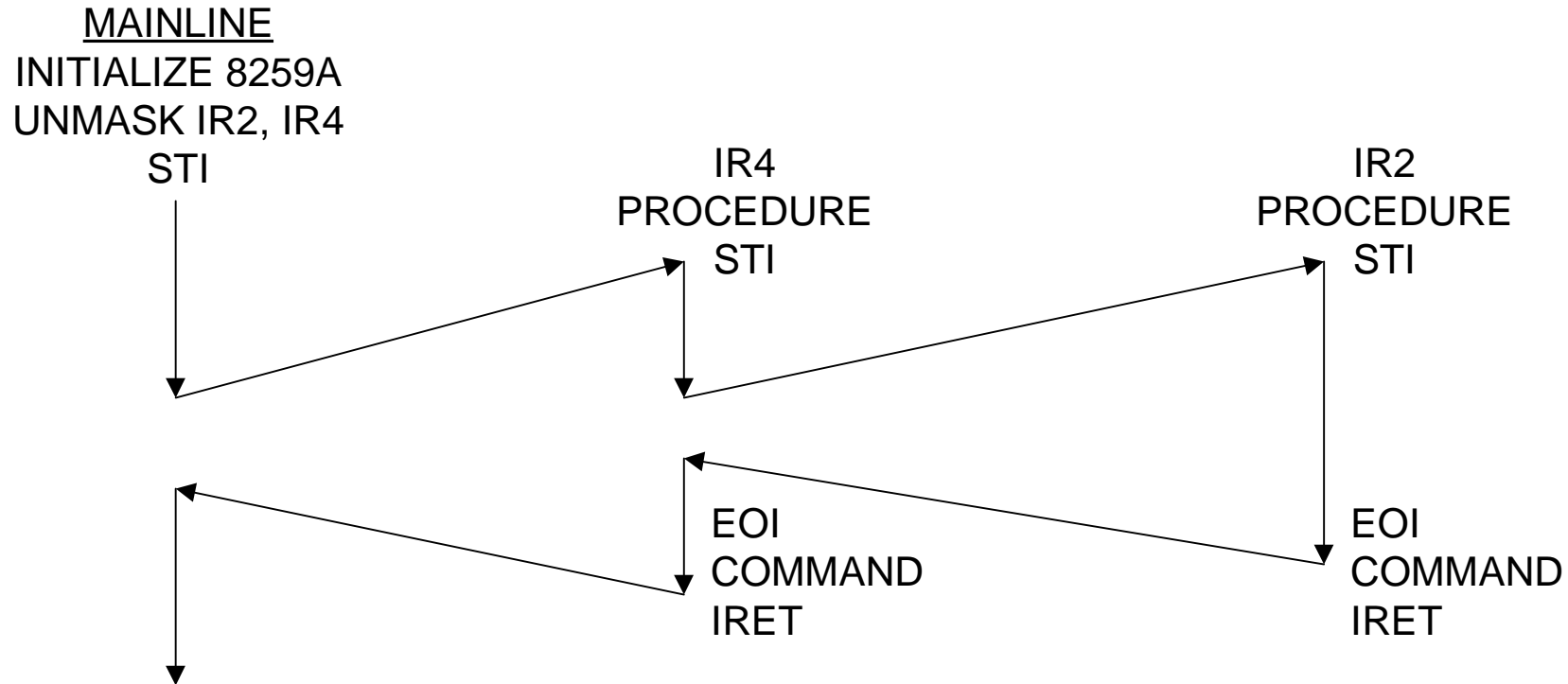
(a) Response with INTR enabled in IR4 procedure (IF=1)

8259A and 8088 Program Flow For IR4 Followed by IR2 (cont.)

MAINLINE
INITIALIZE 8259A
UNMASK IR2, IR4
STI

IR4 PROCEDURE

EOI COMMAND
IRET

IR2 PROCEDURE

EOI COMMAND
IRET
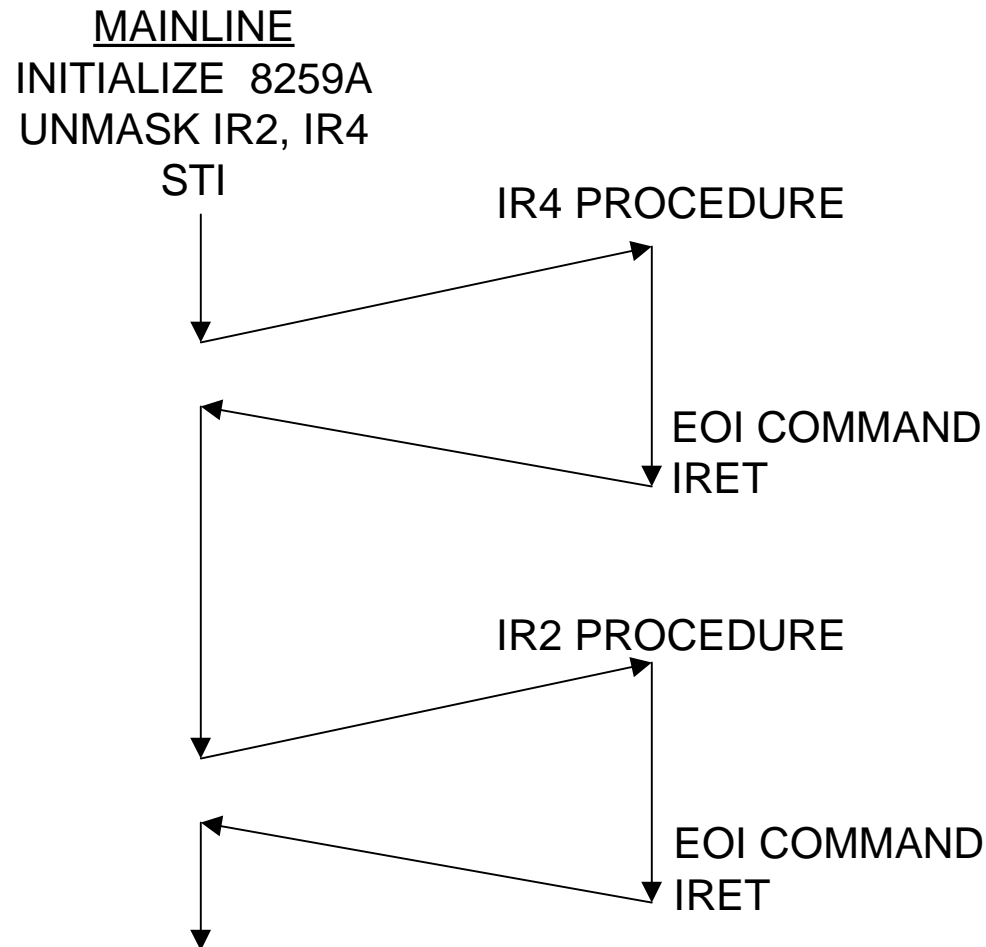
(b) Response with INTR **not** enabled in IR4 procedure (IF=0)

Interrupt Acknowledge Bus Cycle

■ Assume IF is set to accept INTR.

	first cycle				second cycle			
	T1	T2	T3	T4	T1	T2	T3	T4

CLK

ALE

$\overline{\text{LOCK}}$

$\overline{\text{INTA}}$

$AD_7\text{-}AD_0$

VECTOR TYPE
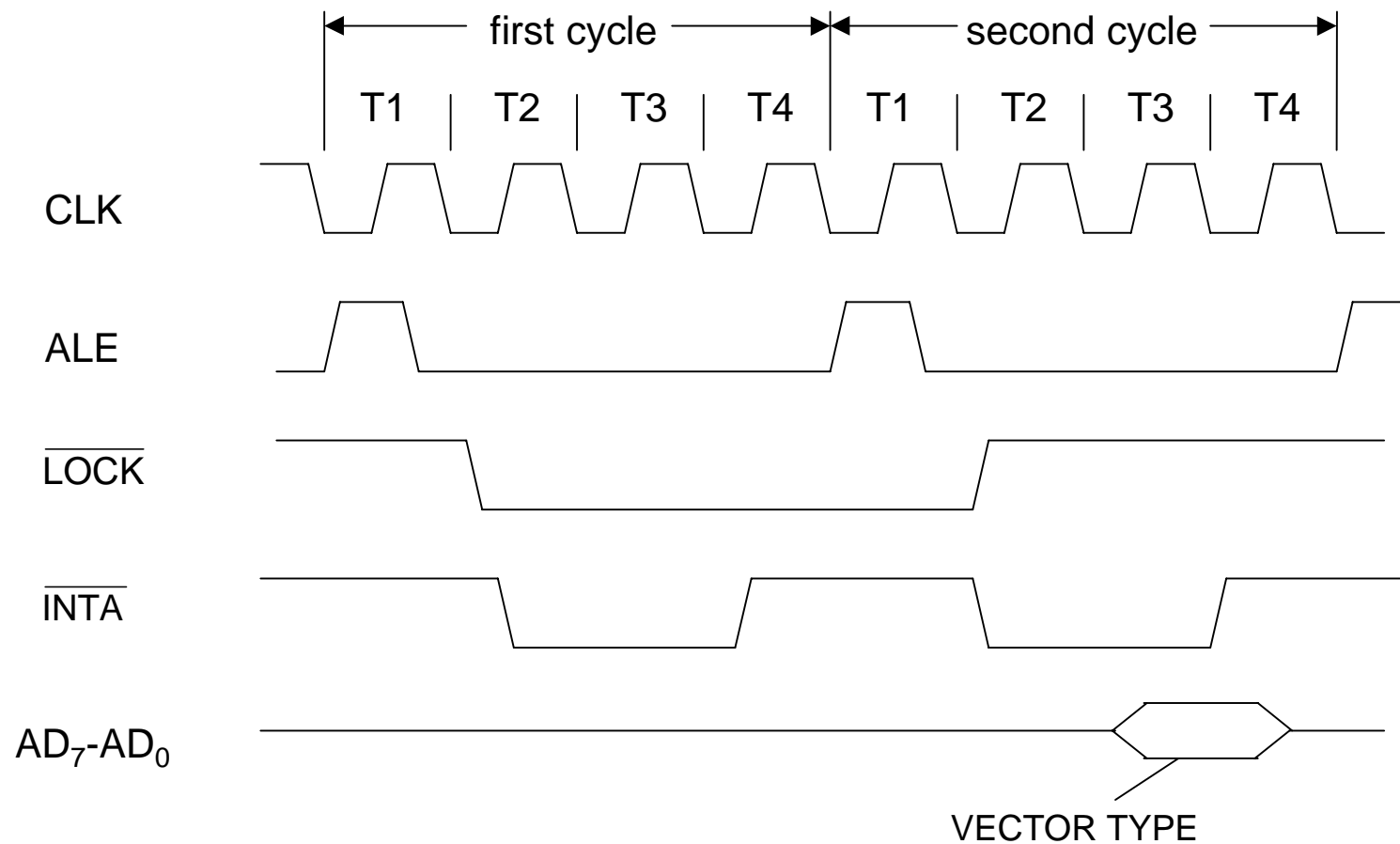
Programming the 8259A
(Brey's)

▮ 8259A may be programmed by writing appropriate bytes into its 7 internal registers (4 "initialization command words" ICW registers and 3 "operational command words" OCW).

▮ ICW commands are used to load the internal control registers of 8259A.

▮ OCW commands permit 8088 to initiate variations in the basic operating modes defined by ICW command.

▮ The ICW and OCW commands can be issued to 8259A using OUT (for I/O mapped) or MOV instruction (for memory mapped).

▮ The initialization sequence of ICW and OCW is shown next.

8259A Initialization Command Word Sending Order

```
┌─────────┐
│  ICW1   │
└─────────┘
     │
     ▼
┌─────────┐
│  ICW2   │
└─────────┘
     │
     ▼
    IN              N
  CASCADE        (SNGL=1)
   MODE?    ───────────────────────────►    IS ICW4      N (IC4=0)
     │                                       NEEDED?
  Y (SNGL=0)                                    │
     │                                       Y (IC4=1)
     ▼                                          ▼
┌─────────┐                              ┌─────────┐
│  ICW3   │──────────────────────────►   │  ICW4   │
└─────────┘                              └─────────┘
                                              │
                                              ▼
                                        ┌──────────────┐
                                        │  READY TO    │
                                        │  ACCEPT      │
                                        │  INTERRUPT   │
                                        │  REQUESTS    │
                                        └──────────────┘
```
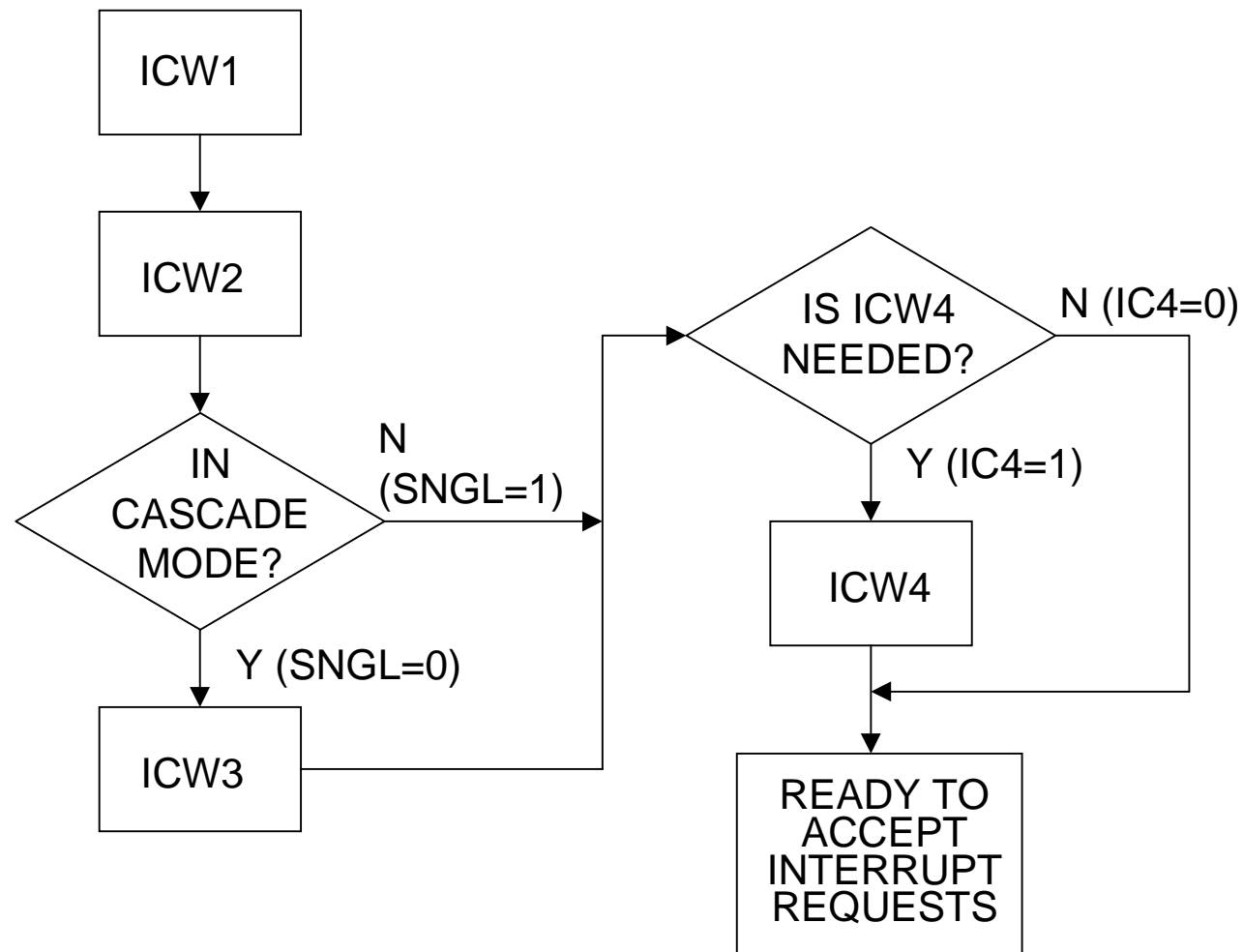
8259A Initialization Command Word Formats ICW1

ICW1: used to program the basic operation of 8259a.

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	A7	A6	A5	1	LTIM	ADI	SNGL	IC4

1=ICW4 NEEDED
0=NO ICW4 NEEDED

1=SINGLE
0=CASCADE MODE

CALL ADDRESS INTERVAL
1=INTERVAL OF 4
0=INTERVAL OF 8
(MCS-80/85 MODE ONLY)

1=LEVEL TRIGGERED MODE
0=EDGE TRIGGERED MODE

A7-A5 OF INTERRUPT
VECTOR ADDRESS
(MCS-80/85 MODE ONLY)
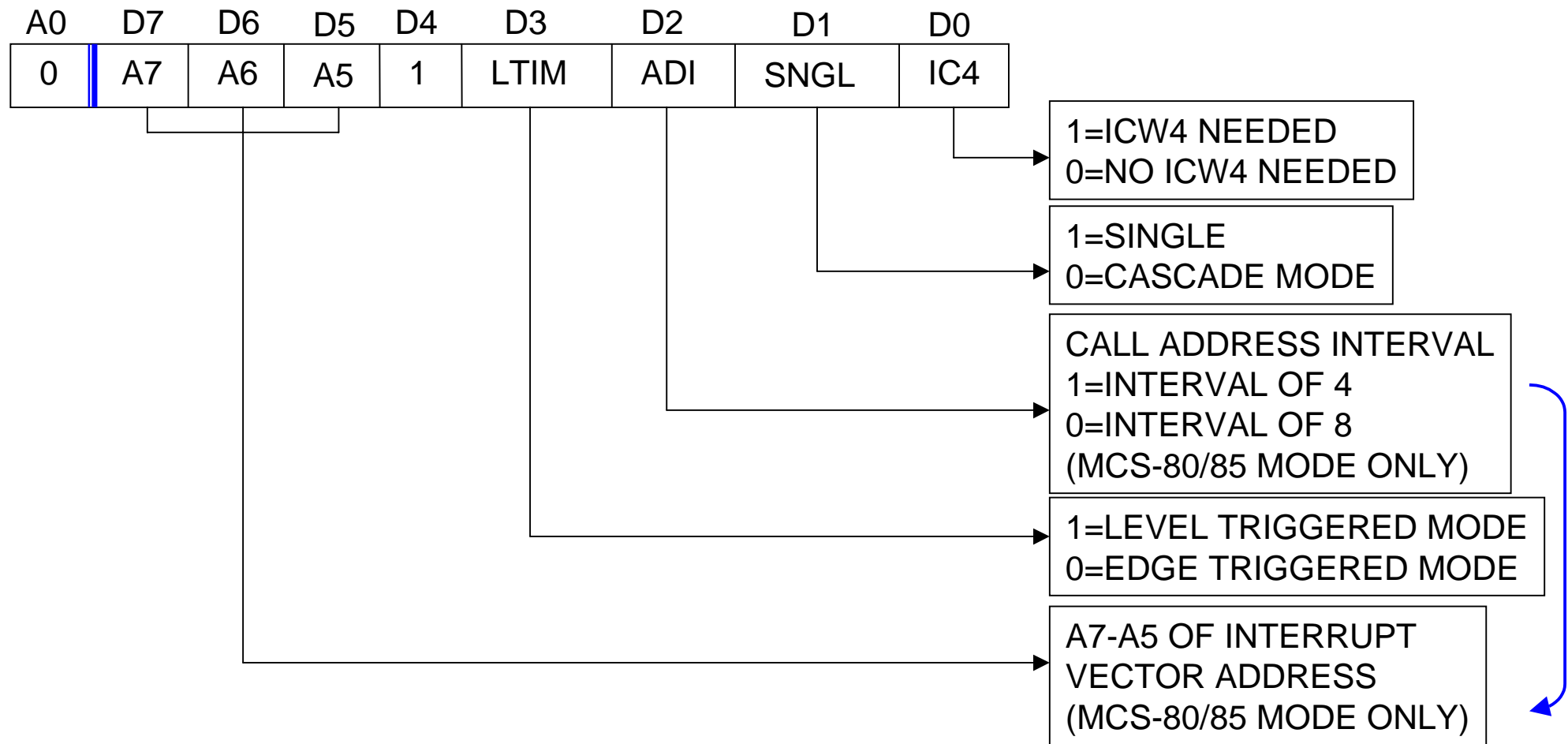
8259A Initialization Command Word Formats ICW2 & ICW3(master)

ICW2: used to select vector number used with the interrupt request inputs.

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	$A15/T7$	$A14/T6$	$A13/T5$	$A12/T4$	$A11/T3$	A10	A9	A8

ICW2

A15-A18 OF INTERRUPT VECTOR ADDRESS (MCS80/85 MODE)
T_7-T_3 OF INTERRUPT VECTOR ADDRESS (8086/8088 MODE)
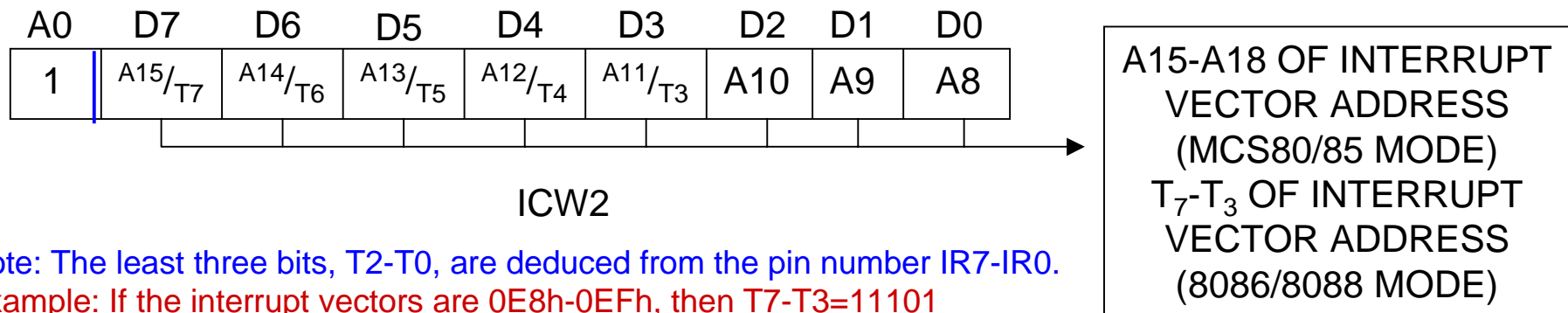
Note: The least three bits, T2-T0, are deduced from the pin number IR7-IR0.
Example: If the interrupt vectors are 0E8h-0EFh, then T7-T3=11101

ICW3: used only when ICW1 specifies that the system is operated in cascade mode; used to indicate where the slave is connected to the master.

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0

ICW3 (MASTER DEVICE)

1=IR INPUT HAS A SLAVE
0=IR INPUT DOES NOT HAVE A SLAVE
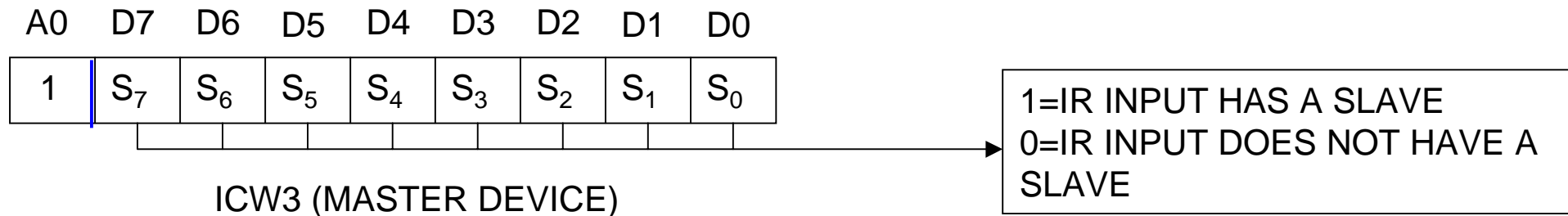
Q: Assume slave is at IR2, what is the ICW3 for master?   (Ans: 04H)

8259A Initialization Command Word Formats ICW3 (Slave)

ICW3 (SLAVE DEVICE)

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	ID2	ID1	ID0

$SLAVE_{ID}(1)$

0	1	2	3	4	5	6	7
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	0	0	0	1	1	1	1
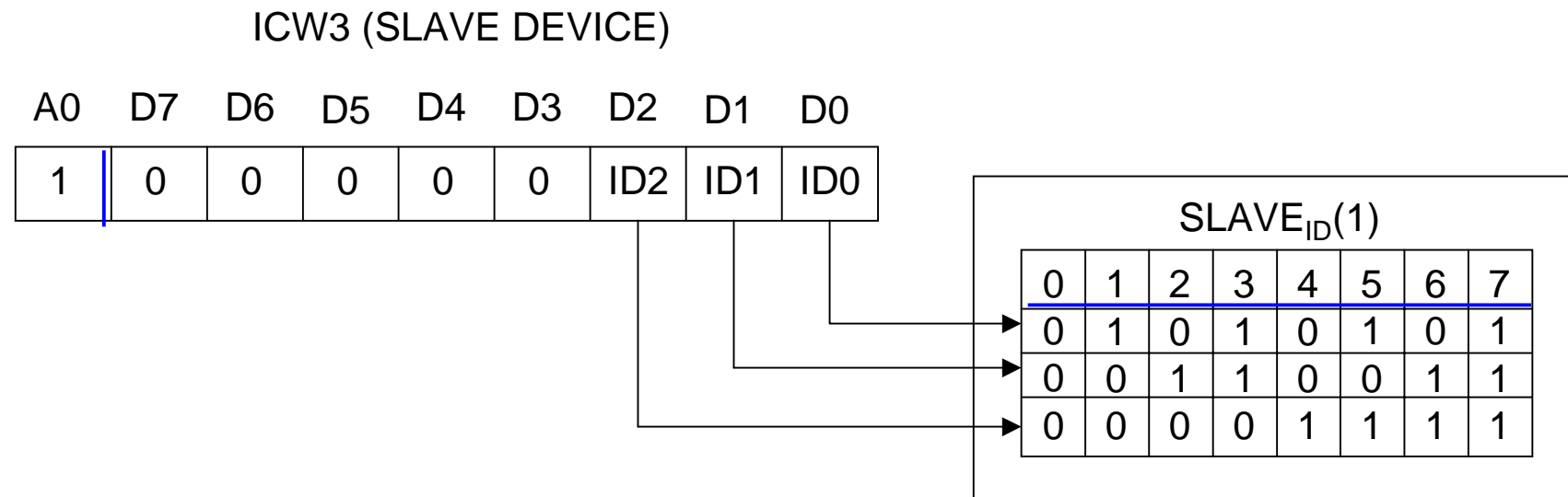
NOTE 1: SLAVE ID IS EQUAL
TO THE CORRESPONDING
MASTER IR INPUT

8259A Initialization Command Word Formats ICW4

ICW4

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	SFNM	BUF	M/S	AEOI	µPM

1=8086/8088 MODE
0=MCS-80/85 MODE

1= AUTO EOI
0=NORMAL EOI

0	X	NON BUFFERED MODE
1	0	BUFFERED MODE/SLAVE
1	1	BUFFERED MODE/MASTER

1=SPECIAL FULLY NESTED MODE
0=NOT SPECIAL FULLY NESTED MODE
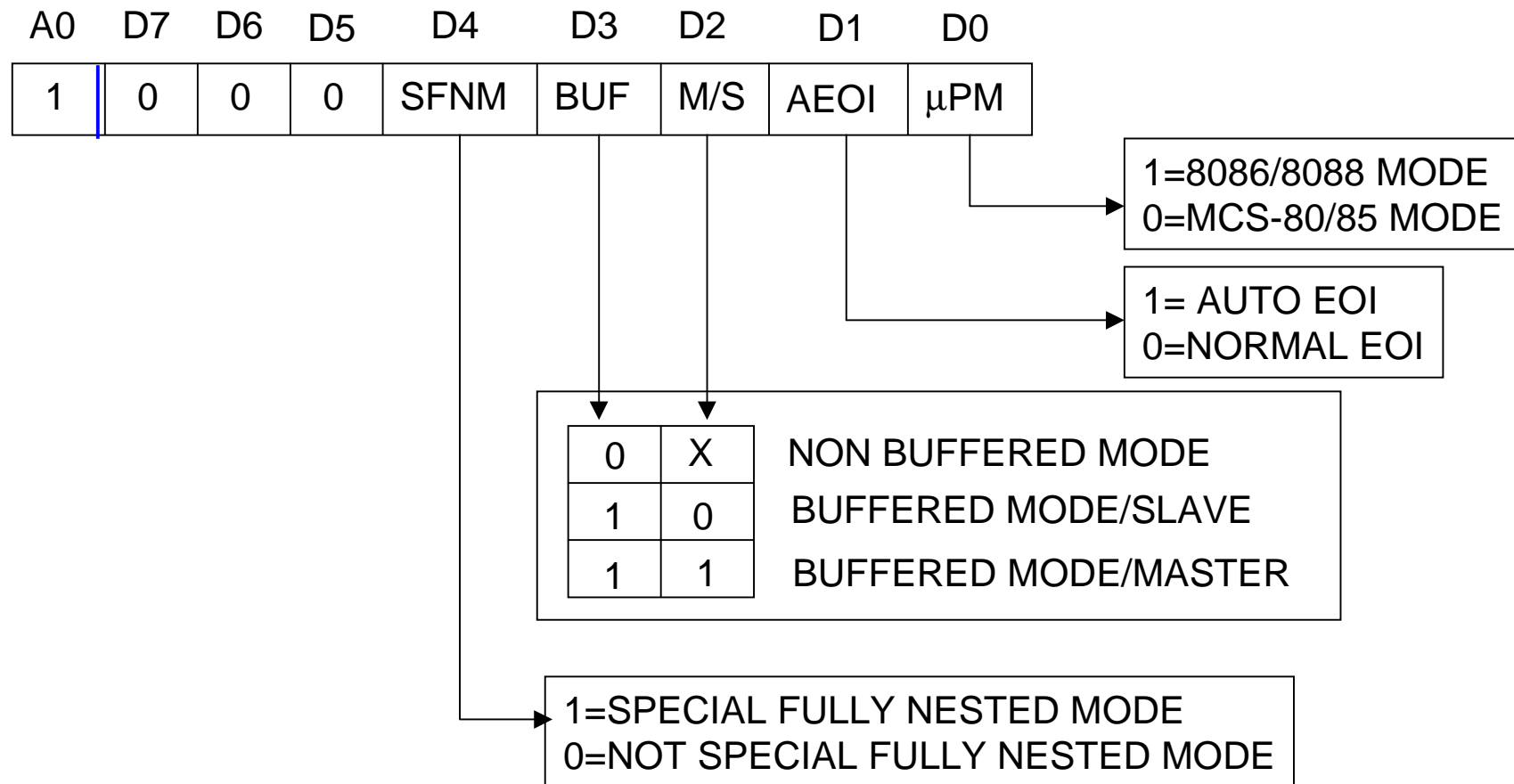
Special Fully Nested Mode: allows master PIC to accept request on a (master) IR input that is already in service

Example of initializing ICW

Example: What values should be written into ICW1 in order to configure the 8259A such that ICW4 is needed in the initialization sequence, the system is going to use multiple 8259A, and its inputs are to be level sensitive? Assume that all unused bits are to be logic 0. Give the results in both binary and hexadecimal form.

Solution:

ICW1 :   D0=1

                                     D1=0

                                     D3=1

                                     D2=D5=D6=D7=0 for don't care

                                     D4=1

➔   ICW1=00011001 B = 19 H

8259A Operational Command Words OCW1

OCW1: is used to set and read the interrupt mask register

(Must be programmed after programming ICW)

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0

OCW1

INTERRUPT MASK
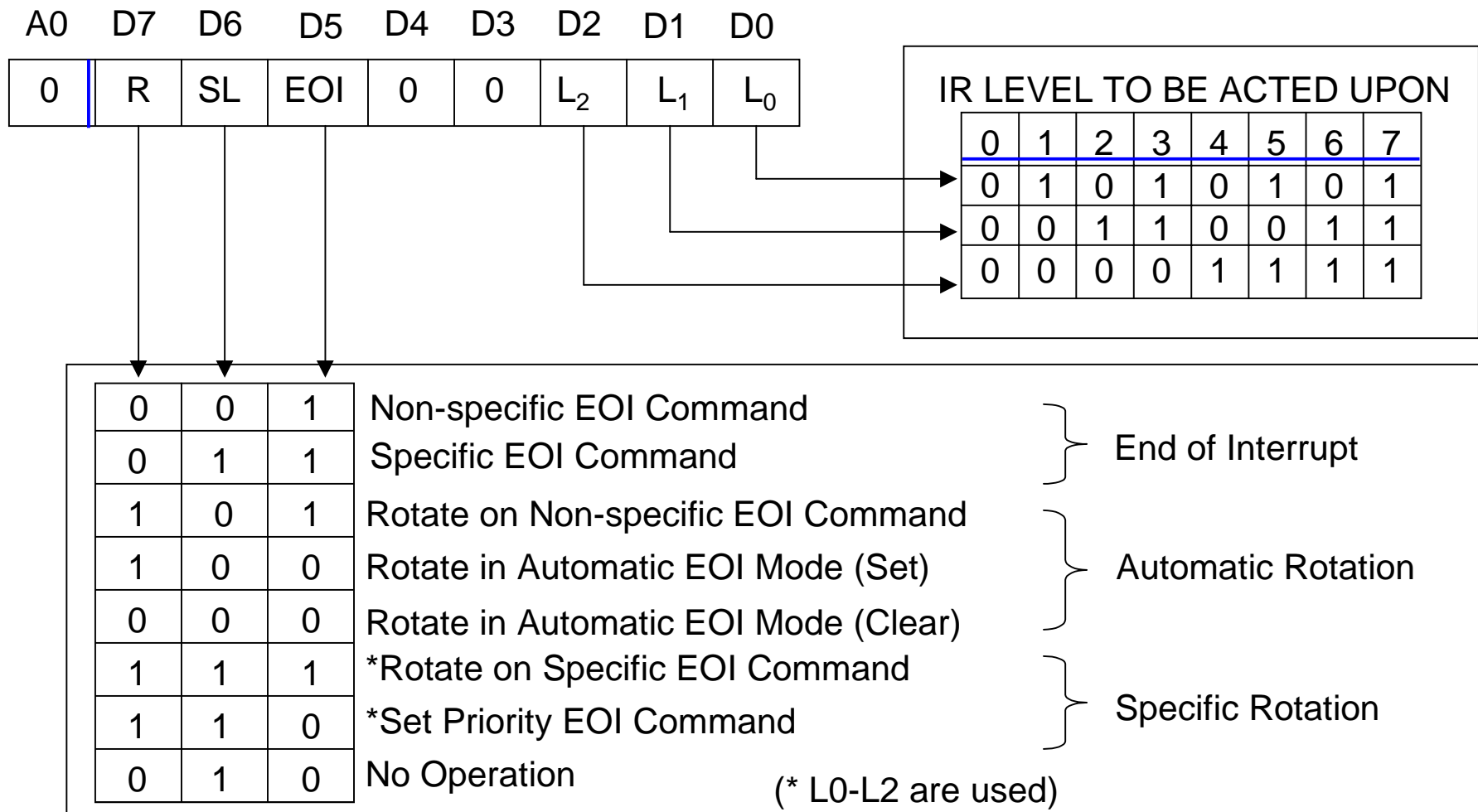1=MASK SET
0=MASK RESET
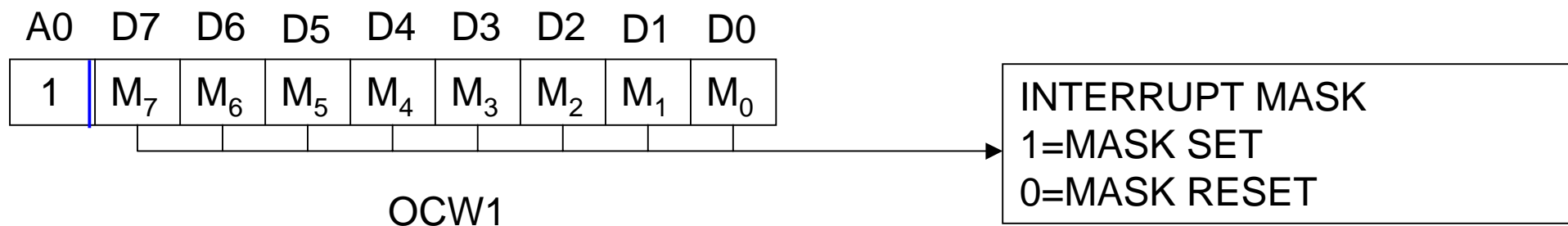
For example, if IR0 and IR2 will be used
$\rightarrow$ OCW1= 11111010

8259A Operational Command Words OCW2

OCW2: to select how the 8259A responds to an interrupt.

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	R	SL	EOI	0	0	L_2	L_1	L_0

IR LEVEL TO BE ACTED UPON

0	1	2	3	4	5	6	7
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	0	0	0	1	1	1	1

R	SL	EOI		
0	0	1	Non-specific EOI Command	End of Interrupt
0	1	1	Specific EOI Command	
1	0	1	Rotate on Non-specific EOI Command	Automatic Rotation
1	0	0	Rotate in Automatic EOI Mode (Set)	
0	0	0	Rotate in Automatic EOI Mode (Clear)	
1	1	1	*Rotate on Specific EOI Command	Specific Rotation
1	1	0	*Set Priority EOI Command	
0	1	0	No Operation	

(* L0-L2 are used)

Nonspecific Priority Rotation

Before Rotation

(Assume IR4 has the highest priority)

(ISR)

IS status

IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
0	1	0	1	0	0	0	0

Priority status

7	6	5	4	3	2	1	0

lowest   highest

After Rotation

IS status

IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
0	1	0	0	0	0	0	0

Priority status

2	1	0	7	6	5	4	3

highest   lowest

8259A Operational Command Words OCW3

OCW3: to select the register to be read, the operation of the special mask register, and the poll command.

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	ESMM	SMM	0	1	P	RR	RIS

0	1	0	1
0	0	1	1
NO ACTION		READ IR REG ON NEXT $\overline{RD}$ PULSE	READ IS REG ON NEXT $\overline{RD}$ PULSE

1 = POLL COMMAND   0 = NO POLL COMMAND

| 0	1	0	1
0	0	1	1
NO ACTION		RESET SPECIAL MASK	SET SPECIAL MASK
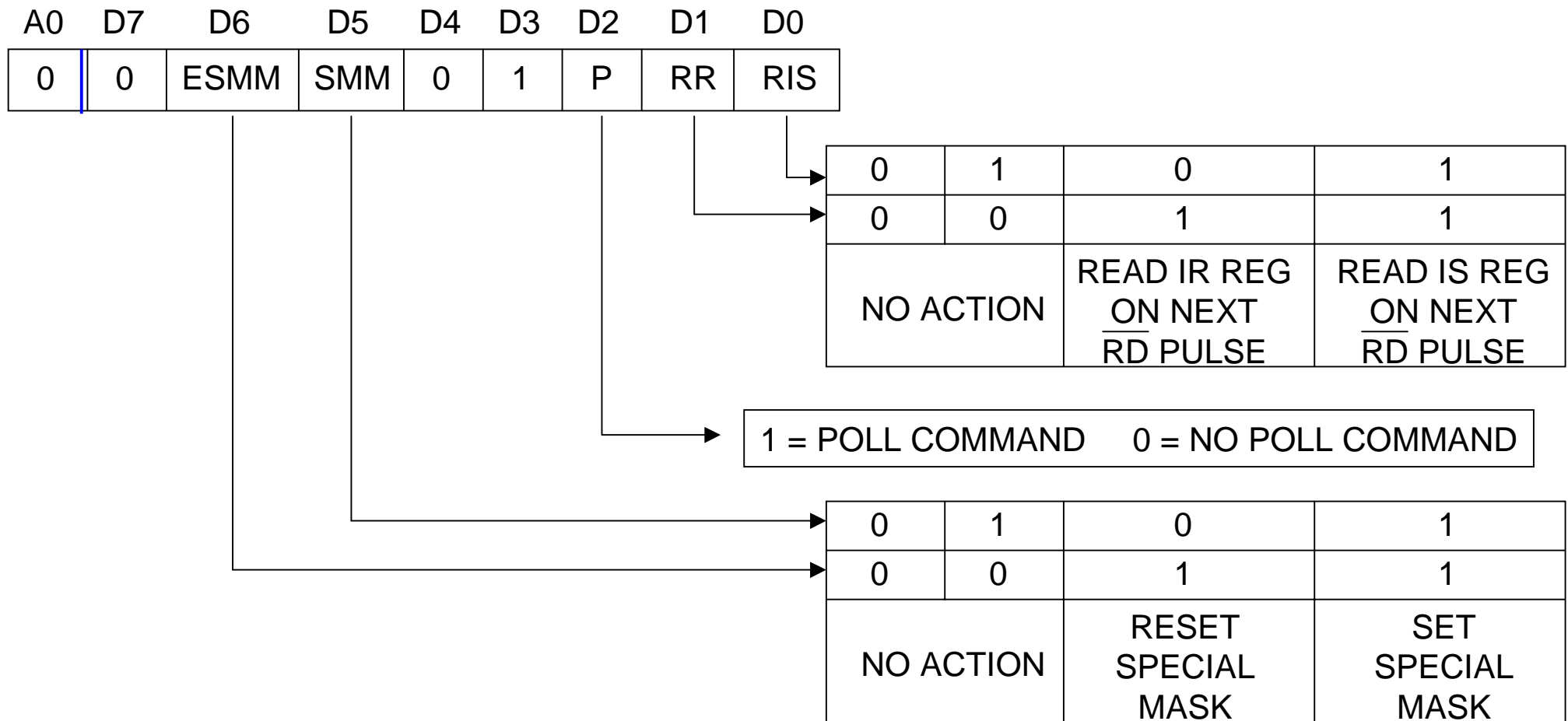
Example of Programming OCW

Example: What should be the OCW1 code if interrupt inputs IR0 through IR3 are to be disabled and IR4 through IR7 enable?

Solution: OCW1=00001111B = 0FH

Example: What OCW2 must be issued to 8259A if the priority scheme rotate on nonspecific EOI command is to be selected?

Solution: OCW2=10100000B=A0H

Adding 8254 and 8259A(s) to an SDK-86 board
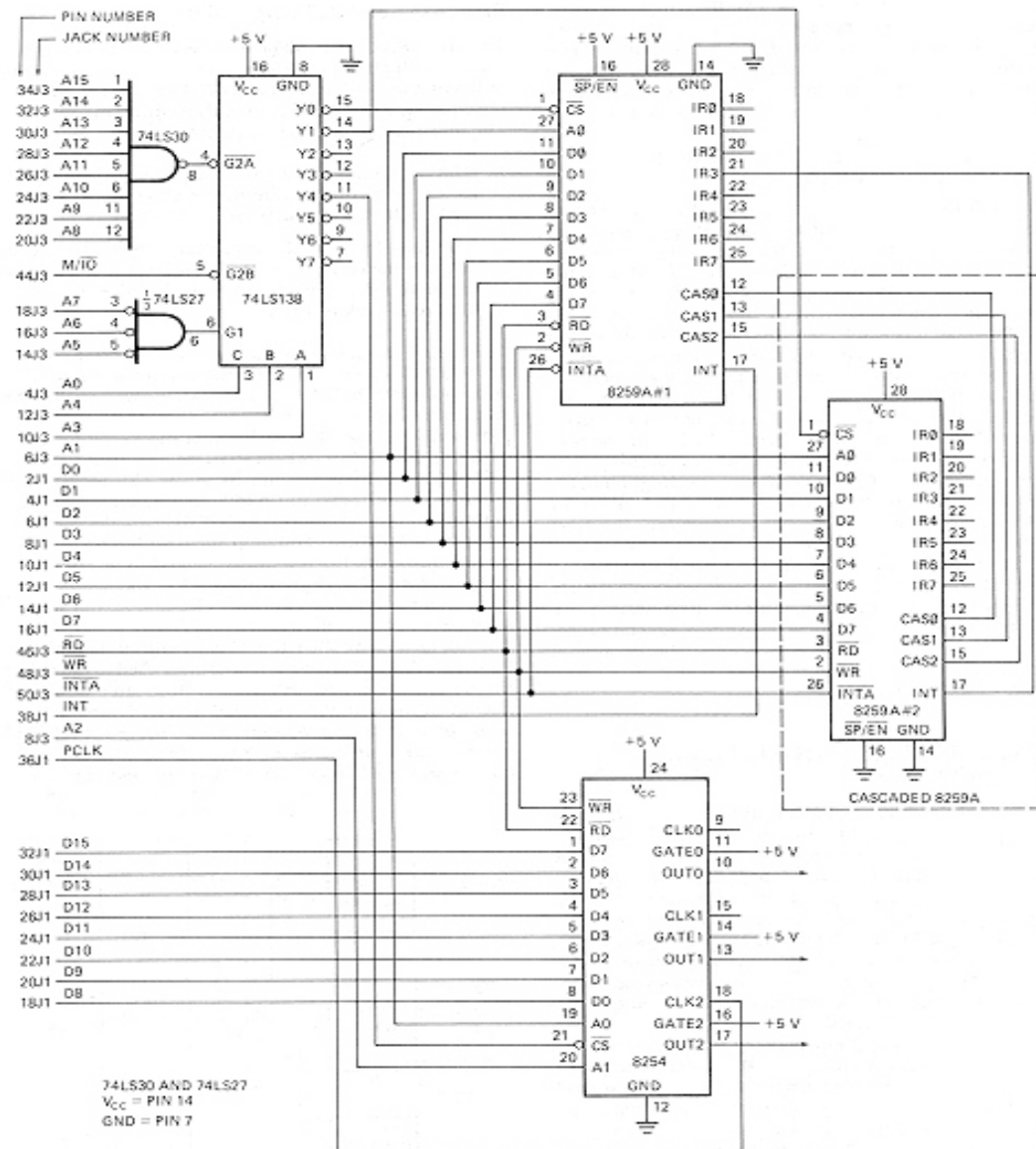
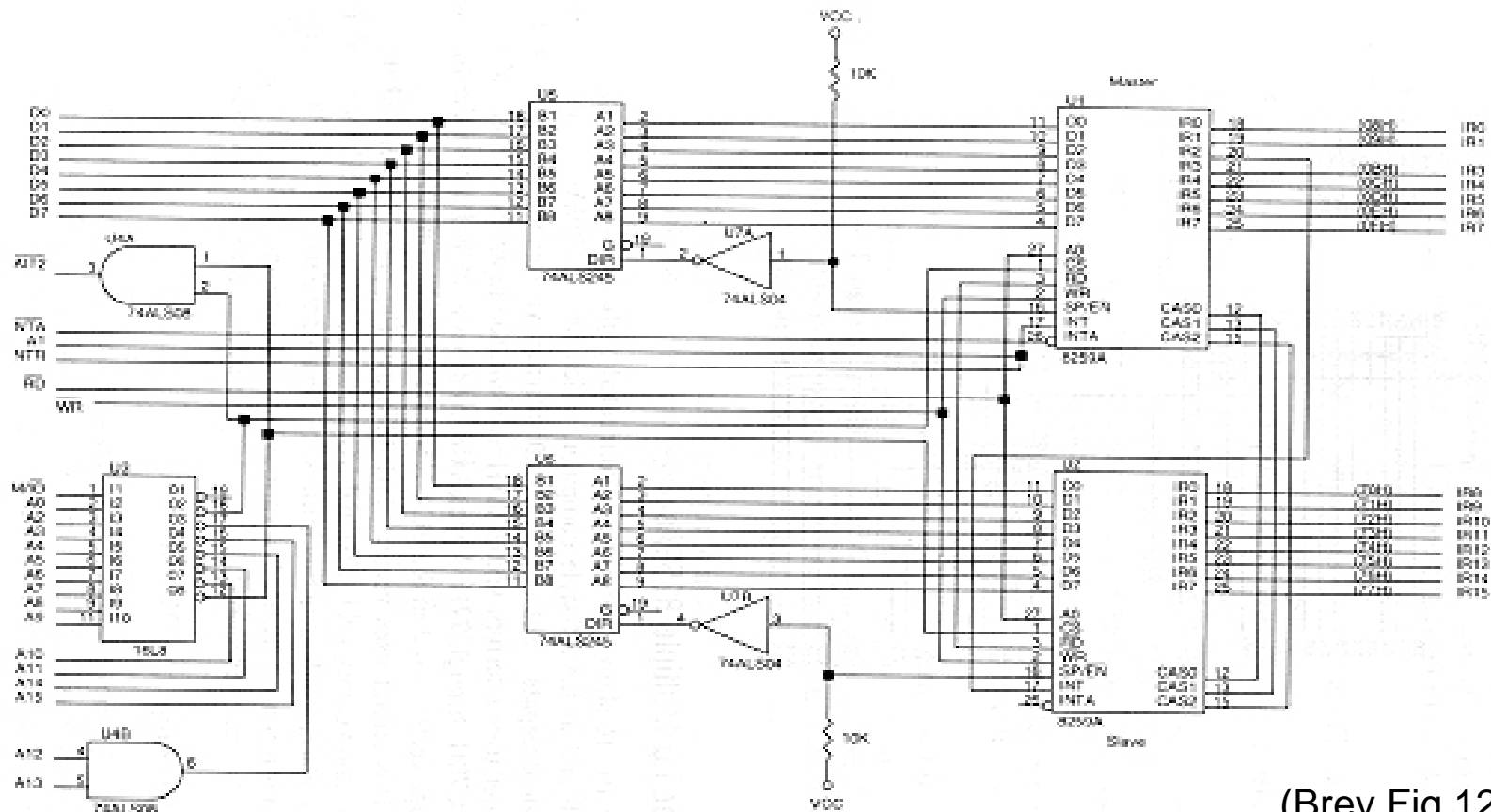

(D Hall Fig 8-14)

(Fig 8-15)

8259#1 :0FF00h

8259#2 :0FF08h

8254   :0FF01h

Multiple 8259As Interfacing

(Brey Fig 12-17)

Two 8259As interfaced to the 8259A at I/O ports 0300H and
0302H for the master and 0304H and 0306H for the slave

System address of ICW and OCW

▌ The system address for ICW and OCW commands word depends on the hardware wiring. ICW and OCW are sent to 8259 base address or base address+n, where n depends on which pin that A0 pin of 8259 connects to 8088.

  Eg. If A0 is connected to A1 of 8088, then $n=2$.

▌ ICW1, OCW2, and OCW3 (A0=0): base-address

▌ ICW2,ICW3,ICW4, and OCW1 (A0=1): base-address+n

▌ ICW2-ICW4 need to be sent sequentially to differentiate them

▌ OCW2 and OCW3 are differentiated by their D4,D3 pins.

▌ ICW need to be initialized before OCW.

(See Brey's pp235-237 and example on pp239.)

Q&A

- Q: What is the system address that store the interrupt subroutine's IP and CS for the hardware interrupt INTR?

- A: After the PIC receives INTA from CPU, it will put up the interrupt vector type on the data bus. The new IP and CS are located by (vector type x 4), derived the same way as the INT n.

- Q: What is the function for IMR?

- A: It is used to disable (mask) the interrupt request from some of the IR pins. If no hardware is connected a specific IR pin, it is not necessary to disable the pin.

- Q: What is SP/EN pin in PIC ?

- A: See the additional note for PIC in buffered mode (pp 454)