Heaven's Light is Our Guide

Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology, Rajshahi-6204, Bangladesh.

Lecture 2 ~ 8086 Interrupts and 8259 Priority Interrupt Controller

S. M. Mahedy Hasan

Assistant Professor, Dept. of CSE

RUET, Rajshahi-6204, Bangladesh

Interrupt

❖ Interrupt means to break the sequence of operation. This halt allows peripheral devices to access the microprocessor.

❖ **What happens when microprocessor get any interrupt?**

✓Whenever an interrupt occurs the processor stops the execution of the current instruction and starts the execution of an Interrupt Service Routine (ISR) or Interrupt Handler.

✓ISR is a program that tells the processor what to do when the interrupt occurs.

✓After the execution of ISR, control returns back to the main routine where it was interrupted.

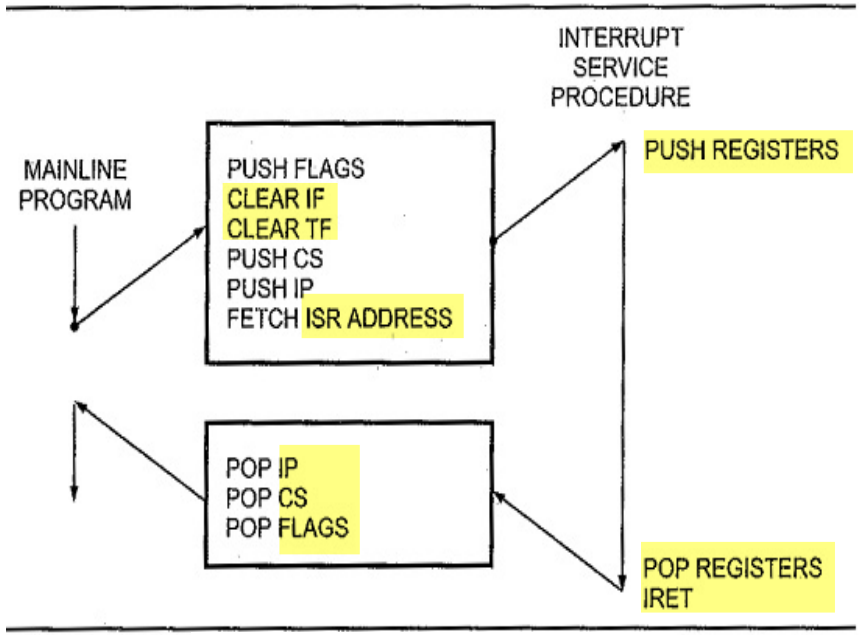
How many ways processor can be interrupted?

❖**Processor can be interrupted in three different ways:**

1. By an external signal generated by the peripheral devices.
2. By an internal signal generated by a special instruction in the program.
3. By an internal signal generated due to an exceptional condition which occurs while executing an instruction.

Responses of processor on interrupts

1. It decrements stack pointer by 2 and pushes the flag register on the stack.
2. It disables the INTR interrupt input by clearing the interrupt flag in the flag.
3. It resets the trap flag in the flag register.
4. It decrements stack pointer by 2 and pushes the current code the segment register contents on e stack.
5. It decrements stack pointer by 2 and pushes the current instruction pointer contents on the stack.
6. It does an indirect far jump at the start of the procedure by loading the CS and IP values for the start of the interrupt service routine (ISR).



Types of interrupt

❖**There are mainly two types of interrupt:**

1. **Hardware interrupt:** Hardware interrupt is caused by any peripheral device by sending a signal through a specified pin to the microprocessor. 8086 has two hardware interrupts.
2. **Software interrupt:** Some instructions are inserted at the desired position into the program to create interrupts.

Hardware interrupt

❖**Two types of Hardware interrupts:**

1. **NMI (Non-maskable interrupt):** It can't avoid this interrupt. It has highest priority. It is also called type-2 interrupt.
2. **INTR (Interrupt Request):** Maskable interrupt. It has lowest priority.

❖**Actions are taken when microprocessor encounters an NMI Interrupt:**

✓Pushes the Flag register values on to the stack.

✓Interrupt flag and trap flag are reset to 0.

✓Pushes the CS (code segment) value and IP (instruction pointer) value of the return address on to the stack.

✓IP is loaded from the contents of the word location 00008H.

✓CS is loaded from the contents of the next word location 0000AH.

Hardware interrupt

❖**INTR:** The INTR is a maskable interrupt because the microprocessor will be interrupted only if interrupts are enabled using set interrupt flag instruction. The INTR interrupt is activated by an I/O port.

- If the interrupt is enabled and NMI is disabled, then the microprocessor first completes the current execution and sends '0' on INTA pin twice.
- The first '0' means INTA informs the external device to get ready and during the second '0' the microprocessor receives the 8 bits, say X, from the programmable interrupt controller.

Software interrupt

✓**INT- Interrupt instruction with type number**

✓It is 2-byte instruction. First byte provides the op-code and the second byte provides the interrupt type number. There are 256 interrupt types under this group, like **INT 0, INT 1, INT 2, INT 3 ……..INT 255**.

❖**Its execution includes the following steps:**

✓Flag register value is pushed on to the stack.

✓Interrupt Flag and Trap Flag are reset to 0.

✓CS value of the return address and IP value of the return address are pushed on to the stack.

✓IP is loaded from the contents of the word location 'type number' × 4.

✓CS is loaded from the contents of the next word location.

Software interrupt

❖ The starting address for type 0 interrupt is 000000H, for type1 interrupt is 00004H similarly for type 2 is 00008H and ……so on. The first five pointers are dedicated interrupt pointers. i.e. −

1. **TYPE 0** interrupt represents division by zero situation. The 8086 will automatically do a type 0 interrupt if the result of a DIV operation is too large to fit in the destination register or memory location.

2. **TYPE 1** interrupt represents single-step execution during the debugging of a program. It will execute one instruction and stop (wait for further instruction by the user).

3. **TYPE 2** interrupt represents NMI interrupt.

4. **TYPE 3** interrupt represents break-point interrupt. These instructions are inserted into the program so that when the processor reaches there, then it stops the normal execution of program and follows the break-point procedure.

5. **TYPE 4** interrupt represents overflow interrupt. If the result of an arithmetic operation on two signed numbers is too large to be represented in the destination register or memory location.

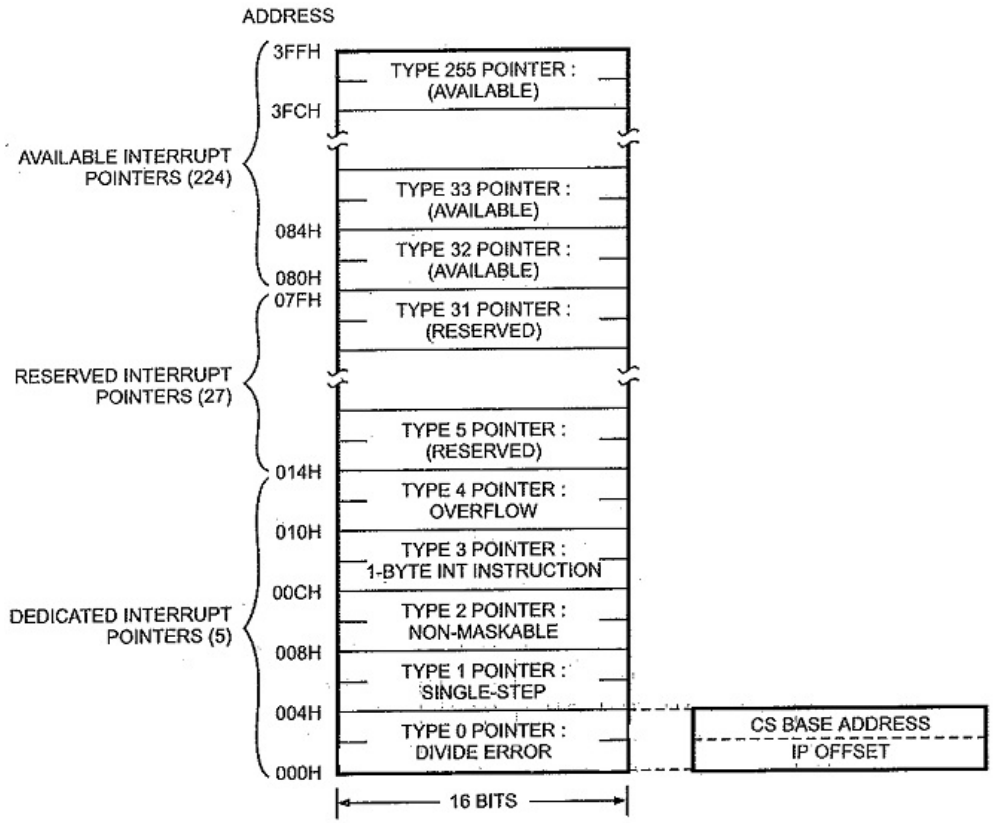
Priority of interrupt

❖The interrupts from Type 5 to Type 31 are reserved for other advanced microprocessors, and interrupts from 32 to Type 255 are available for hardware and software interrupts.

Interrupt	Priority
Type 0, INT(N), Type 4	Highest
NMI	
INTR	
TYPE 1	
	Lowest

Interrupt vector table

- **Now the question is "How to get the values of CS and IP register**?" The 8086 gets the new values of CS and IP register from four memory addresses. When it responds to an interrupt, the 8686 goes to memory locations to get the CS and IP values for the start of the interrupt service routine. In an 8086 Interrupt system the first 1 Kbyte of memory from 00000H to 003FFH is reserved for storing the starting addresses of interrupt service routines. This block of memory is often called the **interrupt vector table** or the **interrupt pointer table**. Since 4 bytes are required to store the CS and IP values for each interrupt service procedure, the table can hold the starting addresses for 256 interrupt service routines. Each interrupt type is given a number between 0 to 255 and the address of each interrupt is found by multiplying the type by 4 e.g. for type 11, interrupt address is 11 x 4 = 44_{10}= 0002CH.



Needs of 8259 programmable interrupt controller

1. 8086 has only two interrupt inputs, NMI and INTR.
2. If we save NMI for a power failure interrupt, this leaves only one interrupt for all the other applications.
3. For applications, we have interrupts from multiple sources, that's why we use an external device called a ***priority interrupt controller*** **(PIC)** to the interrupt signals into a single interrupt input on the processor.
4. It accepts requests from the peripheral devices, determines which of the incoming requests is of the highest importance (priority), ascertains whether the incoming request has a higher priority value than the level currently being serviced, and issues an interrupt to the microprocessor based on this determination.

Features of 8259 programmable interrupt controller

- It is a tool for managing the interrupt requests.
- Compatible with 8-bit as well as 16-bit microprocessors.
- It provides 8 priority interrupt request levels.
- Be expanded to 64 priority levels by cascading additional 8259's.
- The interrupts can be masked or unmasked individually.
- Available in 28-pin DIP.
- It requires a single +5v supply.

Pin description of 8259 PIC

✓***Vcc:*** +5V Supply.

✓***GND:*** Ground.

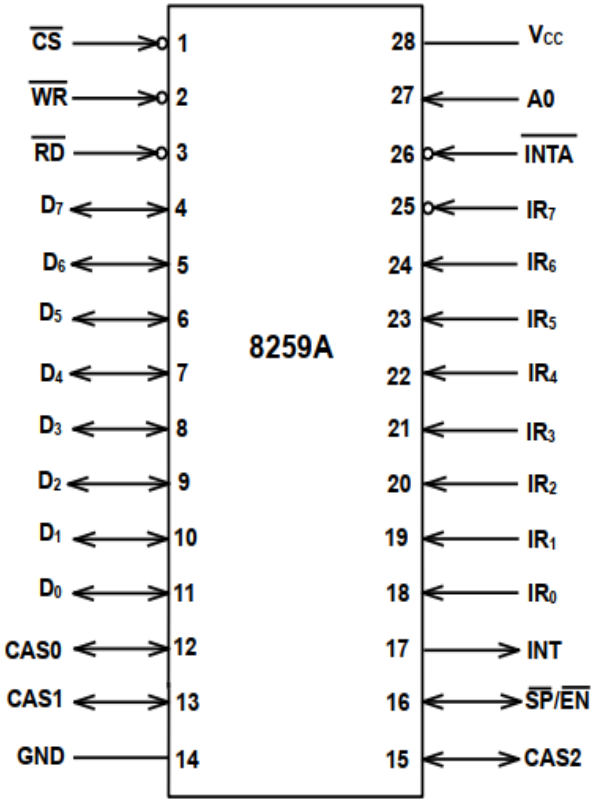
✓***CS*****:** A LOW on this input enables the 8259A. No reading or writing of the chip will occur unless the device is selected.

✓***WR*****:** A LOW on this input enables the microprocessor to write control words (ICWs and OCWs) to the 8259A.

✓***RD*****:** A LOW on this input enables the 8259A to send the status of the Interrupt Request Register (IRR), In Service Register (ISR), the Interrupt Mask Register (IMR), or the Interrupt level onto the Data Bus.

✓***A0 (A0 address line:*** Two addresses are assigned/ reserved in the I/O address space for each 8259A in the system - one with A0 =0 is called even address and other with A0 = 1 is called odd address.

✓***D7-D0:*** Control, status and interrupt-vector information is transferred via this bus.



Pin description of 8259 PIC

✓***CAS0-CAS2:*** 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.

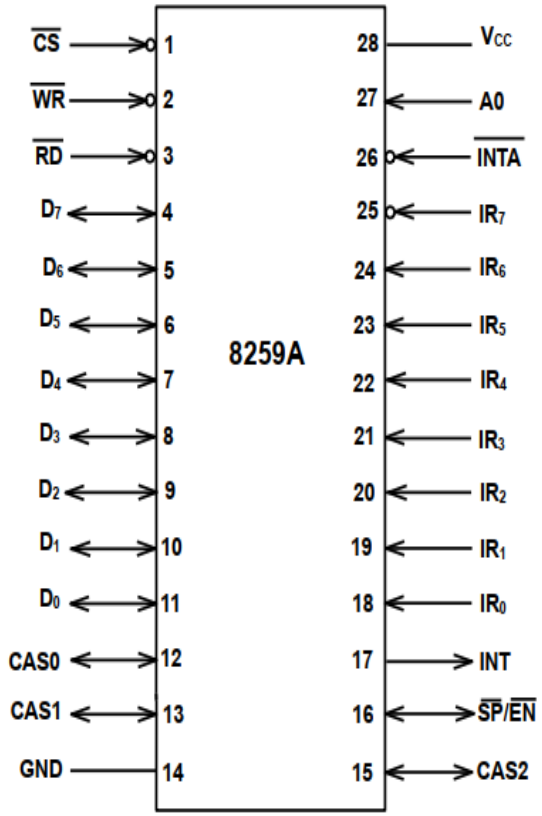
✓***SP/EN*:** This is a dual function pin and stands for slave program/enable buffer. When in the Buffered Mode it can be used as an output pin. When not in the buffered mode it is used as an input pin for master slave operation. For, master (SP=1) or slave (SP=0).

✓***INT:*** This pin goes high whenever a valid interrupt request is asserted. It is used to interrupt the

microprocessor, thus it is connected to the microprocessor's INTR input pin.

✓***INTA*:** This pin is used to enable 8259A interrupt-vector data onto the data bus by a sequence of interrupt acknowledge pulses issued by the microprocessor.

✓***IR7-IR0:*** Asynchronous inputs. An interrupt request is executed by raising an IR input (low to high), and holding it high until it is acknowledged.



Architecture of 8259 programmable interrupt controller

Data Bus Buffer:

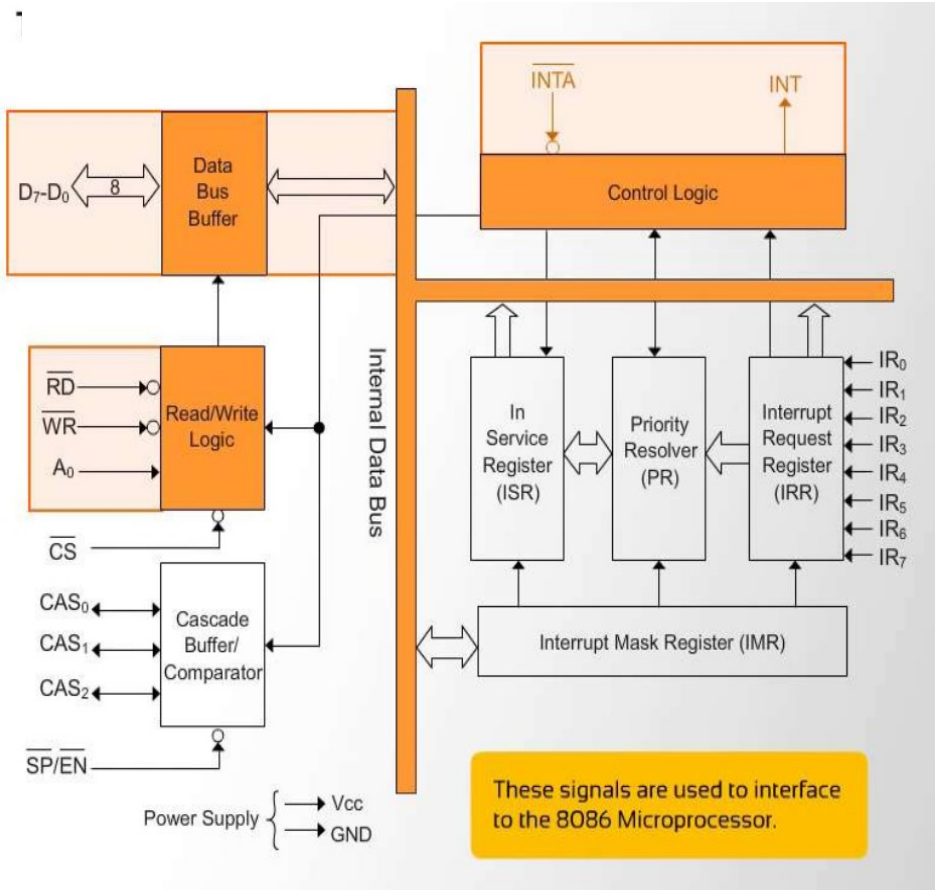
1. This bidirectional 8-bit buffer is used to interface the 8259A to the system Data Bus.
2. Control words and status information are transferred through the Data Bus Buffer.

Read/Write Control Logic:

1. The function of this block is to accept output commands from the microprocessor.
2. This function block also allows the status of the 8259A to be transferred onto the Data Bus.

Control logic: Control logic has two signals.

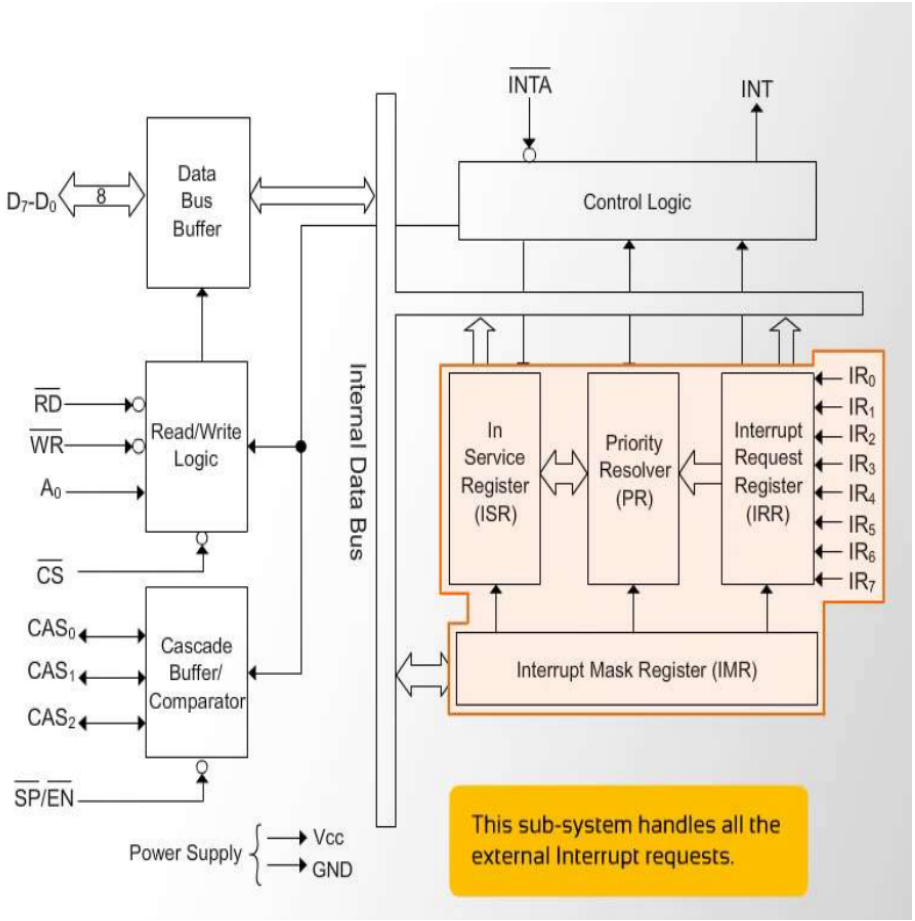
1. **INT (Interrupt):** This output goes directly to the microprocessor interrupt input.
2. ***INTA* (Interrupt Acknowledge):** INTA pulses will cause the 8259A to release vectoring information onto the data bus.



Architecture of 8259 programmable interrupt controller

Interrupt Request Register (IRR):

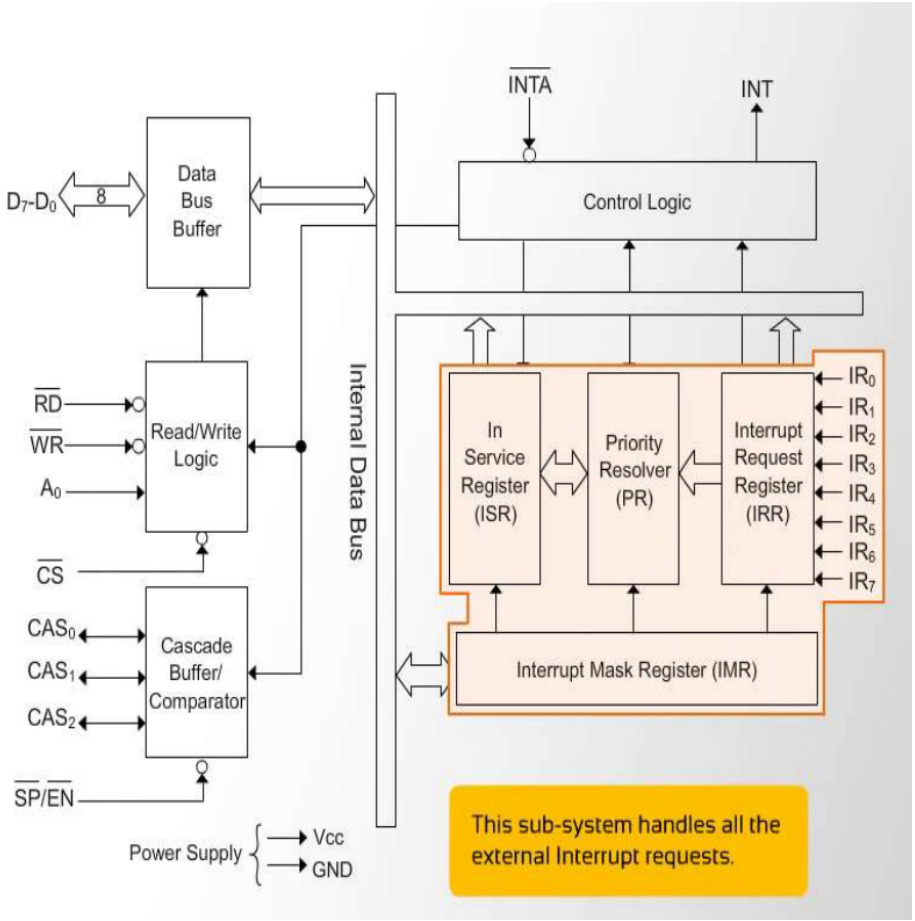
1. Interrupt request register (IRR) stores all the interrupt inputs that are requesting service.
2. It is an 8-bit register – one bit for each interrupt request. Basically, it keeps track of which interrupt inputs are asking for service.
3. If an interrupt input has an interrupt signal on it, then the corresponding bit in the interrupt request register will be set.



Architecture of 8259 programmable interrupt controller

Interrupt Mask Register (IMR):

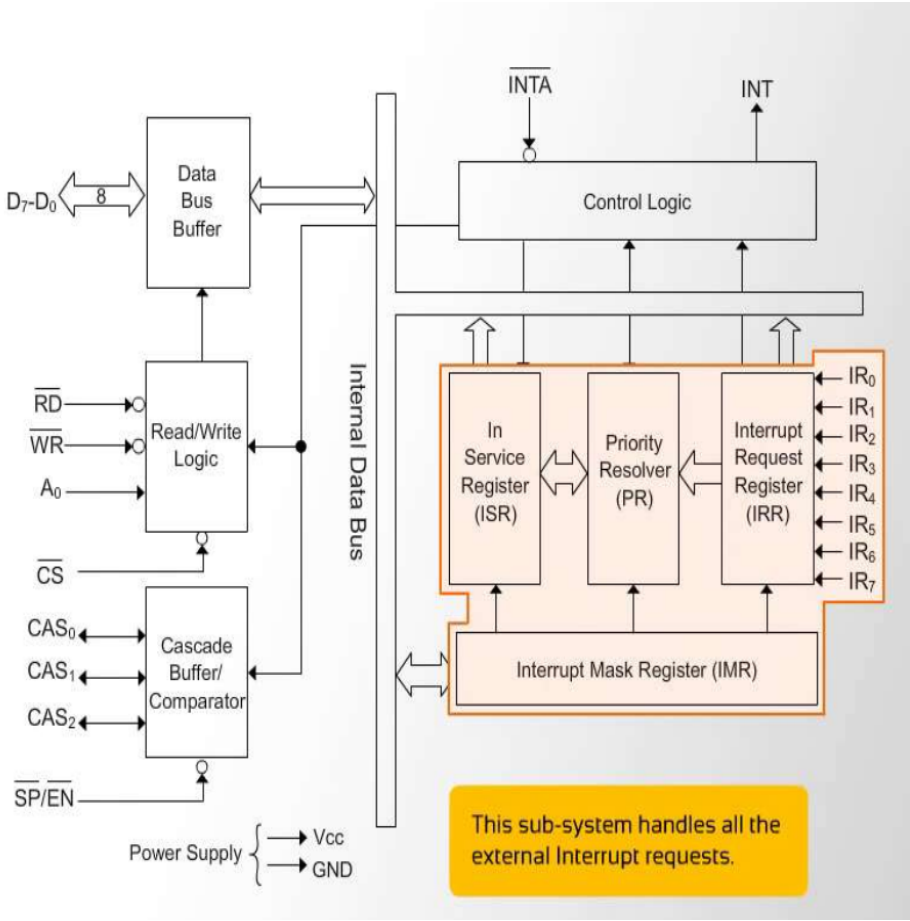
1. The IMR is used to disable (Mask) or enable (Unmask) individual interrupt request inputs.
2. This is also an 8-bit register.
3. Each bit in this register corresponds to the interrupt input with the same number. The IMR operates on the IRR.
4. You unmask an interrupt input by sending a command word with 0 in the bit position that corresponds to that input.



Architecture of 8259 programmable interrupt controller

In-service Register (ISR):

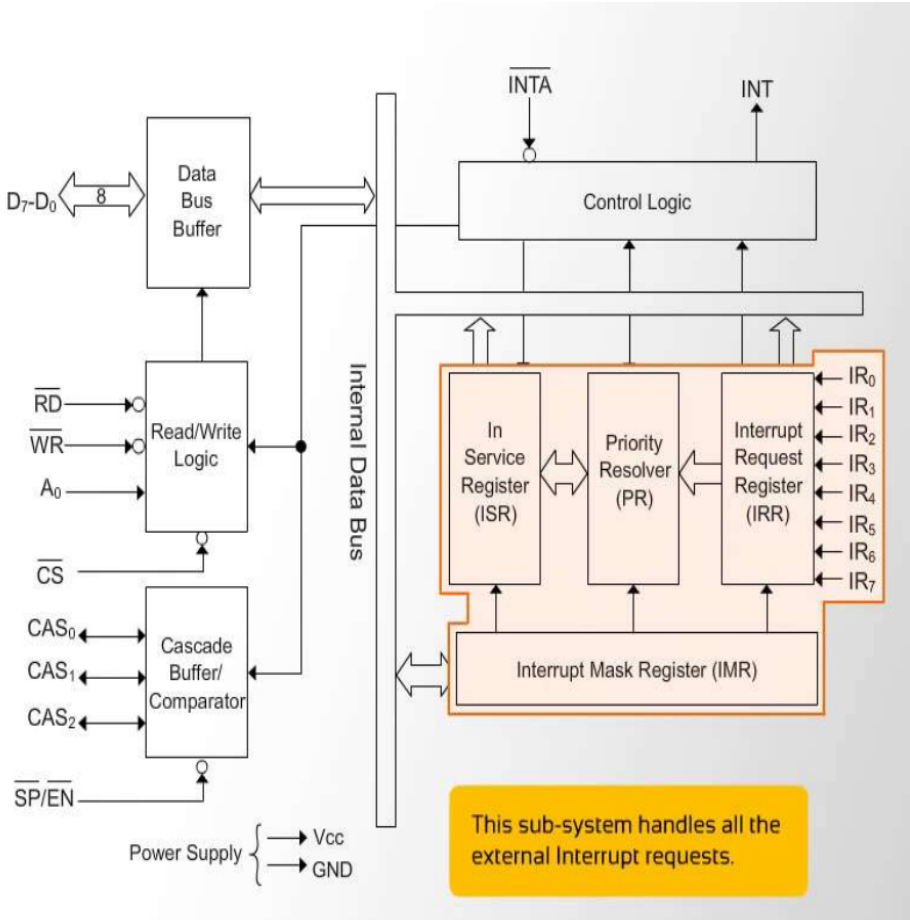
1. The in-service register keeps track of which interrupt inputs are currently being serviced.
2. For each input that is currently being serviced the corresponding bit of in-service register (ISR) will be set.
3. In 8259A, during the service of an interrupt request, if another higher priority interrupt becomes active, it will be acknowledged and the control will be transferred from lower priority interrupt service subroutine (ISS) to higher priority ISS.



Architecture of 8259 programmable interrupt controller

Priority Resolver:

1. This logic block determines the priorities of the interrupts set in the IRR.
2. It takes the information from IRR, IMR and ISR to determine whether the new interrupt request is having highest priority or not.
3. If the new interrupt request is having the highest priority, it is selected and processed.
4. The corresponding bit of ISR will be set during interrupt acknowledge machine cycle.



Interrupt sequence operation

1. One or more of the INTERRUPT REQUEST lines (IR7–0) are raised high, setting the corresponding IRR bit(s).
2. The 8259A evaluates these requests, and sends an INT to the CPU, if appropriate.
3. The CPU acknowledges the INT and responds with an INTA pulse.
4. Upon receiving an INTA from the CPU group, the highest priority ISR bit is set and the corresponding IRR bit is reset. The 8259A does not drive the Data Bus during this cycle.
5. The 8086 will initiate a second INTA pulse. During this pulse, the 8259A releases an 8-bit pointer onto the Data Bus where it is read by the CPU.
6. This completes the interrupt cycle. In the AEOI mode the ISR bit is reset at the end of the second INTA pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

Few scenarios of 8259 PIC

❖**Case 1: What happens if interrupt signals appear at IR2 and IR4 at same time?**

1. In this mode, IR0 has the highest priority, the IR1 input the next highest, and so on down to IR7, which has the lowest priority.
2. If two interrupt signals occur at the same time, 8259A will service the one with the highest priority first, assuming that both inputs are unmasked (enabled) in the 8259A.

❖ **Case 2: Suppose that IR2 and IR4 both are unmasked and that an interrupt comes in on the IR4 input.**

1. The interrupt request on the IR4 input will set bit 4 in the **Interrupt Request Register.**
2. The **priority Resolver** will detect that this bit is set and check the bits in the **In-Service Register** to see if a higher priority input is being serviced or not.
3. If yes, then **priority resolver** will take no action.
4. If no higher priority interrupt is being serviced, then the **priority resolver** will activate the circuitry which sends an interrupt signal to the 8086. When the 8086 responds with INTA pulses, the 8259A will send the interrupt type that was specified for the IR4 input when the 8259A was initialized.
5. As we said before, the 8086 will use the type number it receives from the 8259A to find and execute the interrupt-service procedure written for the IR4 interrupt.

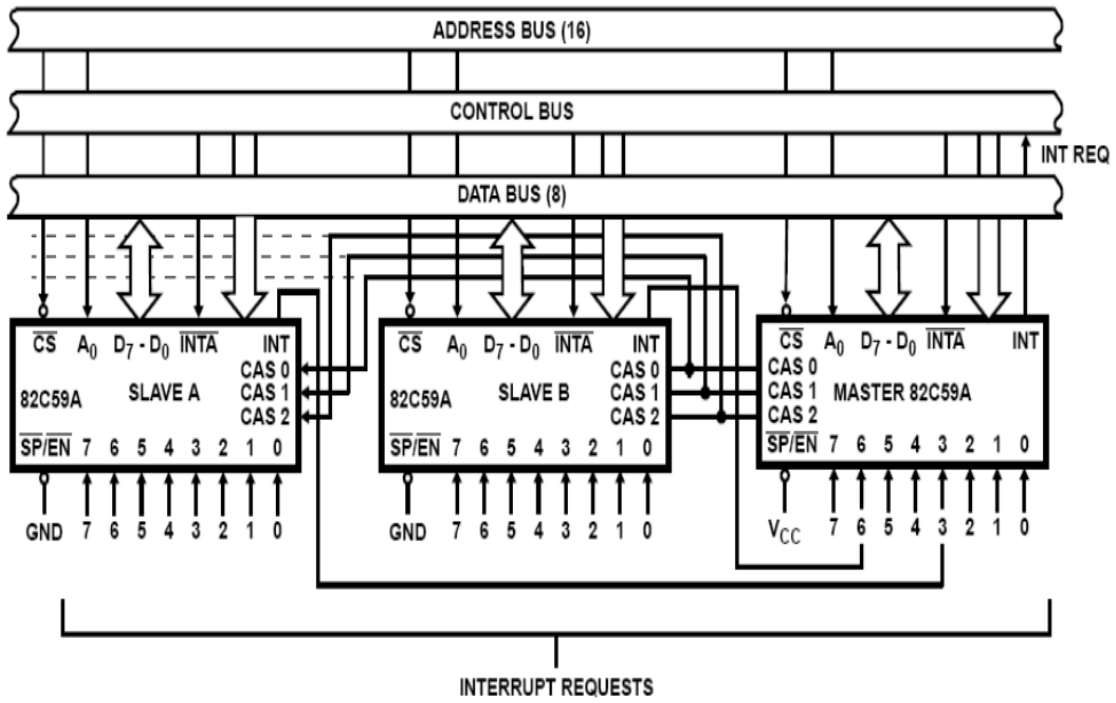
Few scenarios of 8259 PIC

❖ **CASE 3: Suppose that while the 8086 is executing the IR4 service procedure, an interrupt signal arrives at the IR2 input of the 8259A.**

1. This will set bit 2 of the **Interrupt Request Register**.
2. The **Priority Resolver** will detect that this bit is set in the **IRR** and make a decision whether to send another interrupt signal to the 8086.
3. To take decision, the **Priority Resolver** looks at the **In-Service Register**.
4. If a higher priority bit is set in the **ISR,** then the **priority resolver** will wait until the higher priority bit in the **ISR** is reset, before sending an interrupt signal to the 8086 for the new interrupt input.
5. If the Priority Resolver finds that the new interrupt has a higher priority, then the highest priority interrupt currently being serviced, it will set the appropriate bit in the **ISR,** and activate the circuitry which sends a new INT signal to the 8086.
6. In this example, IR2 has higher priority than the IR4, so the **Priority Resolver** will set bit 2 of the **ISR** and activate the circuitry which sends a new INT signal to the 8086. If the 8086 INTR input was reenabled with an STI instruction then this new INT signal will interrupt the 8086 again.
7. When the 8086 sends out a second INTA pulse in response to this interrupt, the 8259A will send it the type number of the IR2 procedure.
8. The 8086 will receive the type number to find and execute the IR2 service procedure.
9. At the end of the IR2 procedure, 8259A will reset the bit 2 in the ISR, so that lower priority interrupts can be serviced.
10. After that, execution get back to the interrupted IR4 procedure.
11. At the end of IR4 procedure, the bit 4 is reset in the ISR and control of the execution go back to the mainline program.

Cascading of 8259 PIC

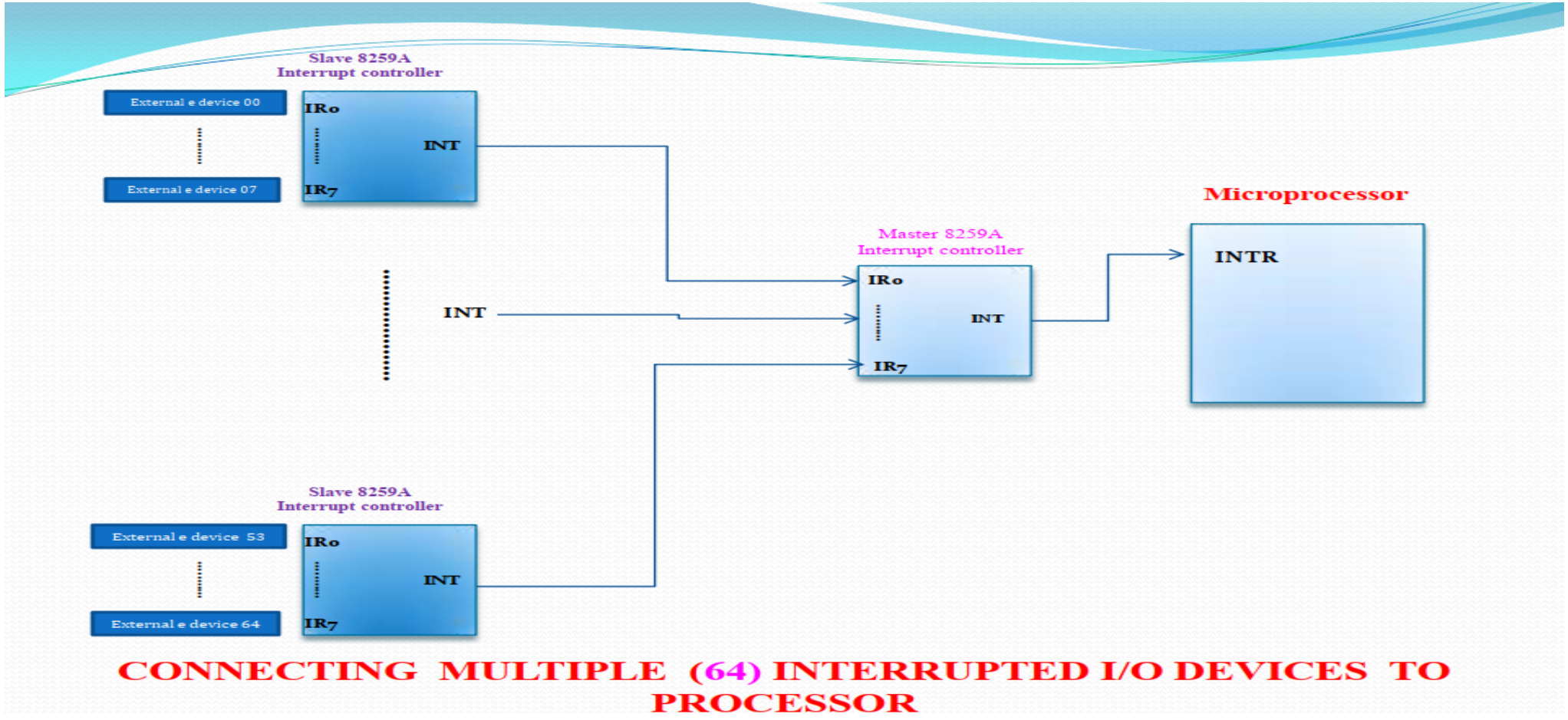
- The cascade pins CAS0 to CAS2 are connected from the master to the corresponding pins of the slave. For the master these pins work as outputs, and for the slave these pins work as inputs. The SP/EN signal is tied high for the master. However, it is grounded for the slave.



Cascading of 8259 PIC

1. When the slave receives an interrupt signal on one of its IR inputs, it checks mask condition and priority of the interrupt request.
2. If the interrupt is unmasked and its priority is higher than any other interrupt level being serviced in the slave, then the slave will send an INT signal to the IR input of a master.
3. If that IR input of the master is unmasked and if that input is a higher priority than any other IR inputs currently being serviced, then the master will send an INT signal to the 8086 INTR input.
4. If the INTR interrupt is enabled, the 8086 will go through its INTR interrupt procedure and sends out two INTA pulses to both the master and the slave.
5. The slave ignores the first interrupt acknowledge pulse but the master outputs a 3-bit slave identification number on the CAS0-CAS2 lines.
6. Sending the 3-bit ID number enables the slave.
7. When the slave receives the second INTA pulse from the 8086, the slave will send the desired type number to the 8086 on the eight data lines.
8. If an interrupt signal is applied directly to one of the IR inputs of the master, the master will send the desired interrupt type to the 8086 when it receives the second INTA pulse from the 8086.

Cascading of 8259 PIC



Command word of 8259 programmable interrupt controller

Command word of 8259 is divided into two parts :

1. Initialization command word (ICW)
2. Operational command word (OCW)

- **Initialization command words(ICW):** There are four ICW's such as ICW1, ICW2, ICW3 and ICW4**.**

i. ICW is given during the initialization of 8259 i.e. before its start functioning.

ii. ICW_1 and ICW_2 commands are compulsory for initialization.

iii. ICW_3 command is given during a cascaded configuration.

iv. If ICW_4 is needed, then it is specified in ICW_1.

v. The sequence order of giving ICW commands is fixed i.e. ICW_1 is given first and then ICW_2 and then ICW_3.

vi. Any of the ICW commands can not be repeated, but the entire initialization process can be repeated if required.

Command word of 8259 programmable interrupt controller

- **Operating command words(OCW):** There are three OCW's such as OCW1, OCW2, OCW3 for operating in various interrupt modes.

i. OCW is given during the operation of 8259 i.e. microprocessor starts using 8259.

ii. OCW commands are not compulsory for 8259.

iii. The sequence order of giving OCW commands is not fixed.

iv. The OCW commands can be repeated.

Command word of 8259 programmable interrupt controller

Value of A0 for different ICW's:

CS	A0	ICW
0	0	ICW 1
0	1	ICW 2
0	1	ICW 3
0	1	ICW 4

Command word of 8259 programmable interrupt controller

Initialization Command Word 1 (ICW 1): Here, A0=0

D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	1	LTM	X	SNGL	IC4

- ✓ **D0:** This bit indicates whether ICW4 is needed or not. If this D0=1, ICW4 is required otherwise, ICW4 is not required.
- ✓ **D1:** This bit indicates whether the 8259 PIC is cascaded or not. If D1=1 then single 8259 PIC is used, otherwise cascaded 8259 PIC is used.
- ✓ **D2:** Don't care bit for 8086 microprocessor. This is used for CALL address interval.
- ✓ **D3:** Level triggered or edge triggered mode. If D3=1 (level triggered mode), D3=0(edge triggered mode) for the interrupt line IR0 to IR7.
- ✓ **D4:** It is always 1.
- ✓ **D5, D6 & D7:** This bits are don't care and generally 0 for 8086 microprocessor.

Command word of 8259 programmable interrupt controller

Initialization Command Word 2 (ICW 2): Here, A0=1

D7	D6	D5	D4	D3	D2	D1	D0
T7	T6	T5	T4	T3	X	X	X

- Here, D0, D1, D2 defines the interrupt number IR0 to IR7 by varying the bits from 000 to 111.
- D3-D7 bits along with D0-D2 decides the interrupt type number based on the vector address provided by the 8086 microprocessor through the ICW 2.

Command word of 8259 programmable interrupt controller

Initialization Command Word 3 (ICW 3): Here, A0=1.

Master Device

D7	D6	D5	D4	D3	D2	D1	D0
IR7	IR6	IR5	IR4	IR3	IR2	IR1	IR0
SL7	SL6	SL5	SL4	SL3	SL3	SL1	SL0

Slave Device

D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	x	ID2	ID1	ID0

Slave Identification number

ID	0	1	2	3	4	5	6	7
D0	0	0	0	0	1	1	1	1
D1	0	0	1	1	0	0	1	1
D2	0	1	0	1	0	1	0	1

Command word of 8259 programmable interrupt controller

Initialization Command Word 3 (ICW 3) for Master:

- Only used when ICW1 indicates that the system is operated in cascade mode.
- ICW3 (for master device) indicates where the slave is connected to the master. Suppose, we have two slaves connected to a master using IR0 and IR1.
- The master is programmed with an ICW3 of 03H.

Initialization Command Word 3 (ICW 3) for Slave:

- ICW3 (for slave device) indicates where the slave is connected to the master.
- Suppose, we have two slaves connected to a master using IR0 and IR1.One slave is programmed with an ICW3 of 01H and other with an ICW3 of 02H.

Command word of 8259 programmable interrupt controller

Initialization Command Word 4 (ICW 4): Here, A0=1

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	SFNM	BUF	M/S	AEOI	MP

- Bit D4 indicates specially fully nested mode. If D4 =1 then it is in specially fully nested mode and if D4 =0 it is in fully nested mode. This bit is used in cascaded mode only.
- Bit D3 and D2 specifies Buffered mode and Master/Slave Mode.

D3	D2	
1	x	Non-Buffered Mode
1	0	Buffered Slave Mode
1	1	Buffered Master Mode

- If D1=1 then it is operated in auto end of interrupt mode and if D1=0 it is in normal EOI mode.
- If D0=1, it sets 8259 to operate with 8086/8088 and if D0=0, it sets 8259 to operate with 8085 system.

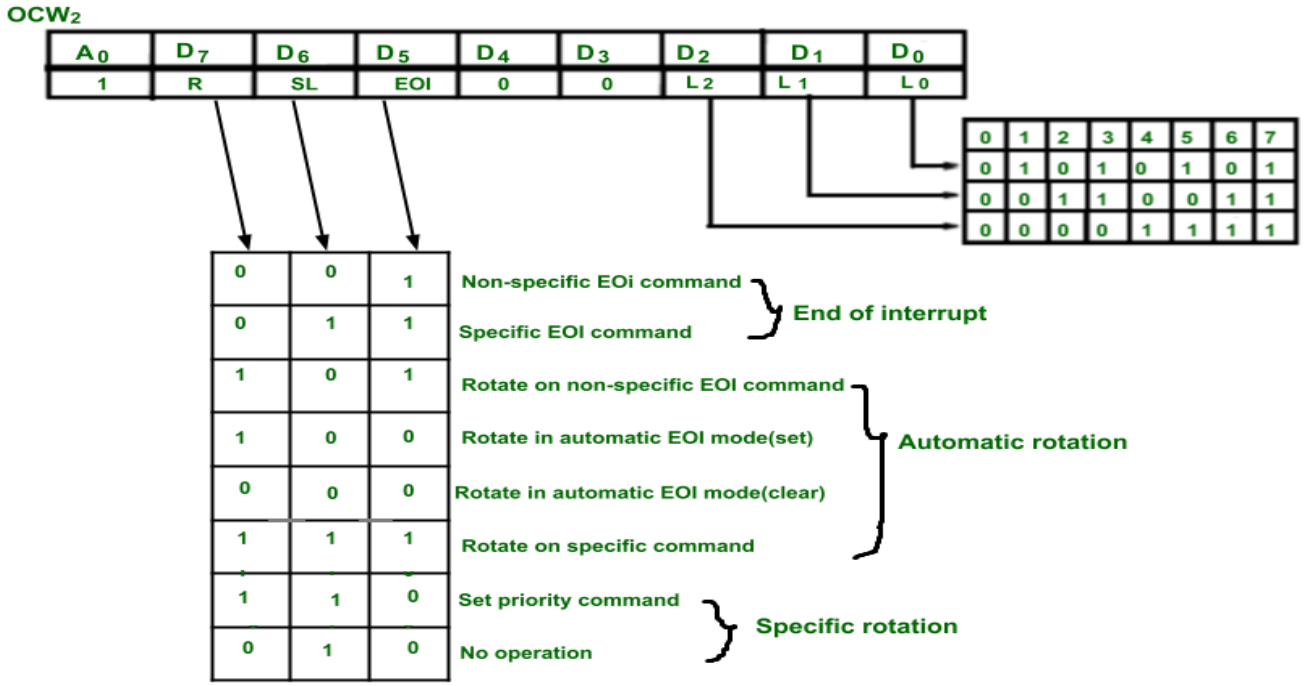
Command word of 8259 programmable interrupt controller

- **Operational Command Word(OCW):** After initialization, the 8259 Programmable Interrupt Controller is ready to process interrupt requests. However, during operation, it might be necessary to change the mode of processing the interrupts. Operation Command Words (OCWs) are used for this purpose.
- **Operational Command Word 1(OCW1):** Here A0=1, It is used to set and reset the mask bits in IMR(interrupt mask register). M7 – M0 describes 8 mask bits.

D7	D6	D5	D4	D3	D2	D1	D0
M7	M6	M5	M4	M3	M2	M1	M0

Command word of 8259 programmable interrupt controller

- **Operational Command Word 2(OCW2): Here A0=0,** It is used for selecting the mode of operation of 8259. Here L2 to L0 are used to describe interrupt level on which action need to be performed.



Command word of 8259 programmable interrupt controller

- **Operational Command Word 3(OCW3): Here A0=0,** This OCW is used to read the status of the registers; and to set or reset the Special Mask and Polled modes.

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

P=0, No polled Command
P=1, polled command

ESMM	SMM	
0	X	Disable Special Mask Mode
1	0	Special Mask mode is reset(Normal Mask Mode)
1	1	Special Mask Mode is set

RR	RIS	
0	X	No Operation
1	0	Read IRR register
1	1	Read ISR register

Operating modes of 8259 programmable interrupt controller

1. **Fully nested mode**
2. **Special fully nested mode**
3. **Priority modes**
4. **Special mask mode**
5. **Polled mode**

Operating modes of 8259 programmable interrupt controller

Fully Nested Mode:

- It is the default mode of operation of 8259.
- Here, IR_0 has the highest priority and IR_7 has the lowest priority. When any interrupt requests occurs then the highest priority interrupt request is serviced first and its vector address is placed on data bus and its corresponding bit in ISR register is set until the processor executes the EOI command before returning the interrupt service routine or AEOI(Automatic end of interrupt bit is set) until the falling of the last INTA'.
- When the ISR bit is set for an interrupt, then all the equal and lower priority interrupts are masked, but a higher level interrupt request can occur and which will be acknowledged only if the microprocessor interrupt enables flag IF= 1.
- It is suitable for a single 8259 configuration.

Operating modes of 8259 programmable interrupt controller

Special Fully Nested Mode:

- This mode will be used in the case of a big system where cascading is used, and the priority has to be conserved within each slave. In this case the fully nested mode will be programmed to the master (using ICW4).
- This mode is similar to the normal nested mode with the following exceptions:

1. When an interrupt request from a certain slave is in service this slave is not locked out from the master's priority logic and further interrupt requests from higher priority IR's within the slave will be recognized by the master and will initiate interrupts to the processor. (In the normal nested mode a slave is masked out when its request is in service and no higher requests from the same slave can be serviced.)
2. When exiting the Interrupt Service routine the software has to check whether the interrupt serviced was the only one from that slave. This is done by sending a non-specific End of Interrupt (EOI) command to the slave and then reading its In-Service register and checking for zero. If it is empty, a non-specific EOI can be sent to the master too. If not, no EOI should be sent.

Operating modes of 8259 programmable interrupt controller

Priority Mode:

- There are two rotating priority modes –

1. Automatic rotation mode

- It is used when various interrupt sources are of the same priority. In this mode, after a device is serviced, it gets the lowest priority. All other priorities rotate according to it.
- **Example**: If IR_4 has just been serviced, it will get the lowest priority.

2. Specific Rotation Mode

- Here, the programmer can alter priorities by programming the lowest priority and thus fixing all other priorities.
- **For example**: If IR_6 is programmed as the lowest priority, then IR_7 will have the highest priority.

Operating modes of 8259 programmable interrupt controller

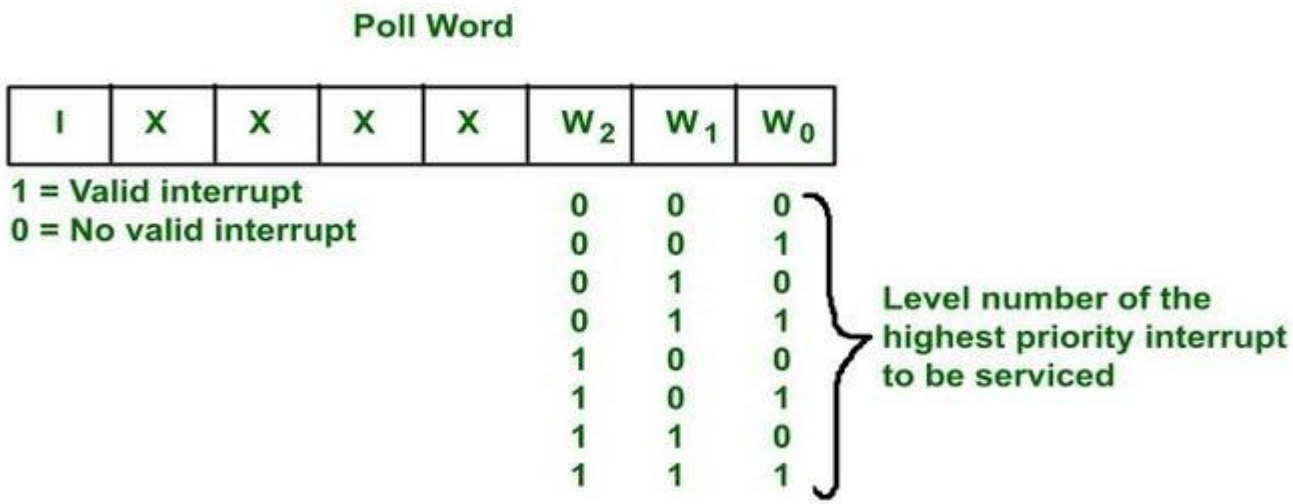
Special Mask Mode:

- Some applications may require an interrupt service routine to dynamically alter the system priority structure during its execution under software control.
- For example, the routine may wish to inhibit lower priority requests for a portion of its execution but enable some of them for another portion.
- The difficulty here is that if an Interrupt Request is acknowledged and an End of Interrupt command did not reset its IS bit (i.e., while executing a service routine), the 8259A would have inhibited all lower priority requests with no easy way for the routine to enable them.
- That is where the Special Mask Mode comes in. In the special Mask Mode, when a mask bit is set in OCW1, it inhibits further interrupts at that level and enables interrupts from all other levels (lower as well as higher) that are not masked.

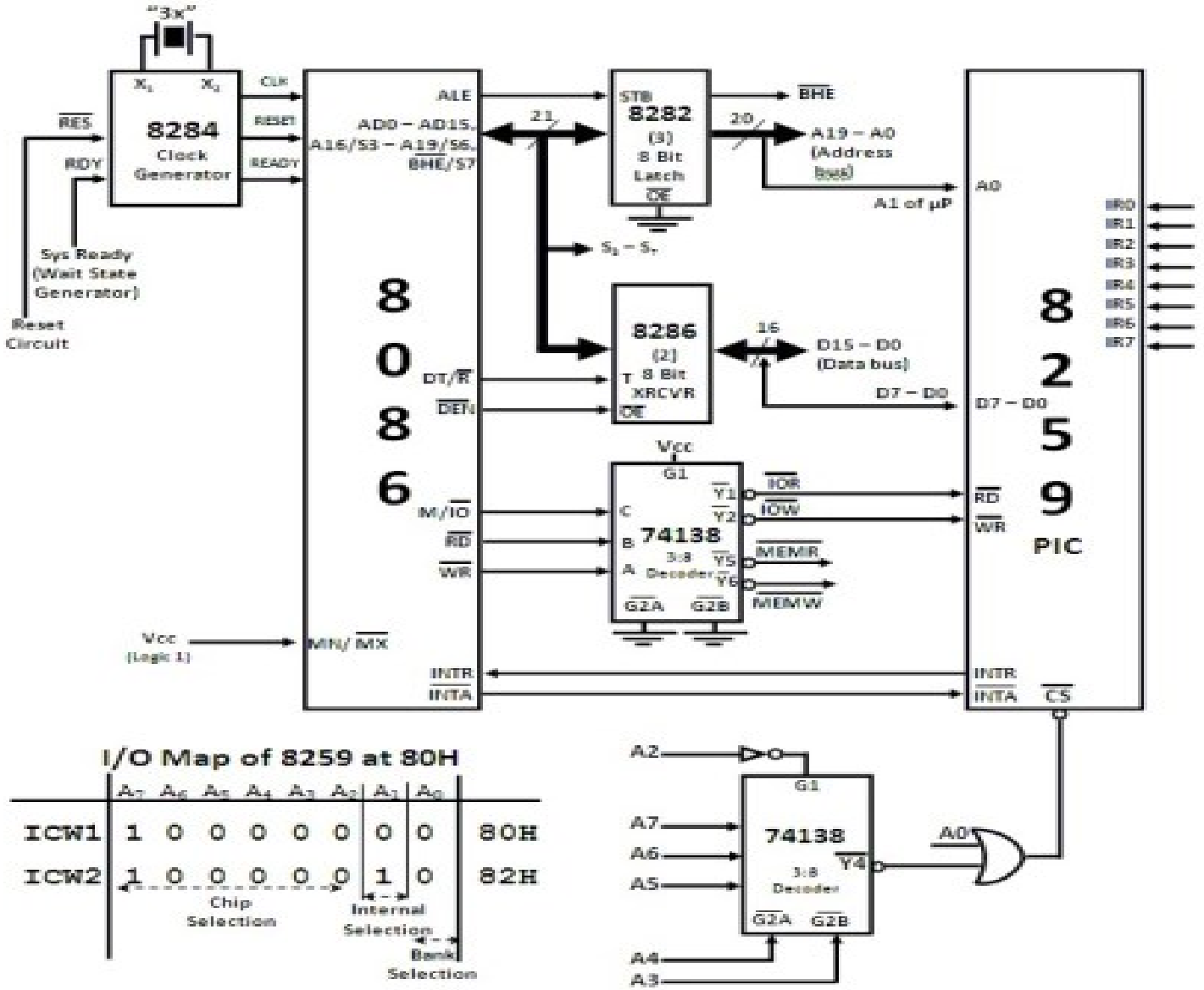
Operating modes of 8259 programmable interrupt controller

Polled Mode:

- Here the INT pin of 8259 is not used, so, 8259 cannot interrupt the µp. Instead, the µP will provide a poll command to 8259 using OCW_3. In response, 8259 provides a poll word to the µP. The poll word indicates the highest priority interrupt which needs service from µP. Thereafter, the µP services the interrupt.



Interfacing 8259 with 8086



Thank You ☺