Heaven's light is our guide

Rajshahi University of Engineering & Technology

Department of Computer Science & Engineering

Course Title: Peripherals & Interfacings

Topic : 8259 Programmable Interrupt Controller

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 completes 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.

Interrupt

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

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

❖ **Need for Interrupt**:

- External devices are comparatively slower than CPU. So, if there is no interrupt, CPU would waste a lot of time waiting for external devices to match its speed with that of CPU. This decreases the efficiency of CPU. Hence, interrupt is required to eliminate these limitations.

Interrupt

❖**Response of processor due to interrupt:**

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).

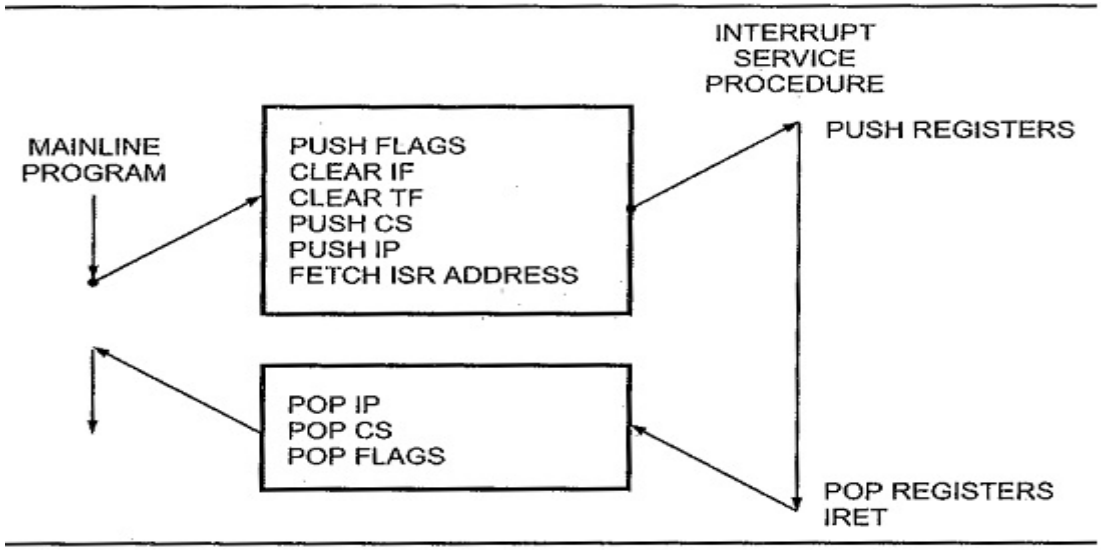


Fig. 9.1 8086 interrupt response

Types of Interrupt

❖**Types of interrupt: 1. Hardware interrupt 2. Software interrupt**

❖**Hardware Interrupts:** Hardware interrupt is caused by any peripheral device by sending a signal through a specified pin to the microprocessor. 8086 has two 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.

•**INTA:** This pin indicates that the interrupt is received by the microprocessor or not.

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

- Completes the current instruction that is in progress.
- 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.

❖**These actions are taken by the microprocessor:**

- First completes the current instruction.
- Activates INTA output and receives the interrupt type, say X.
- Interrupt flag and trap flag is reset to 0.
- Flag register value, CS value of the return address and IP value of the return address are pushed on to the stack.
- IP value is loaded from the contents of word location X × 4.
- CS is loaded from the contents of the next word location.

Software Interrupt

❖**Software Interrupts:** Some instructions are inserted at the desired position into the program to create interrupts.

- **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.

❖**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.

Priority of interrupt:

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.

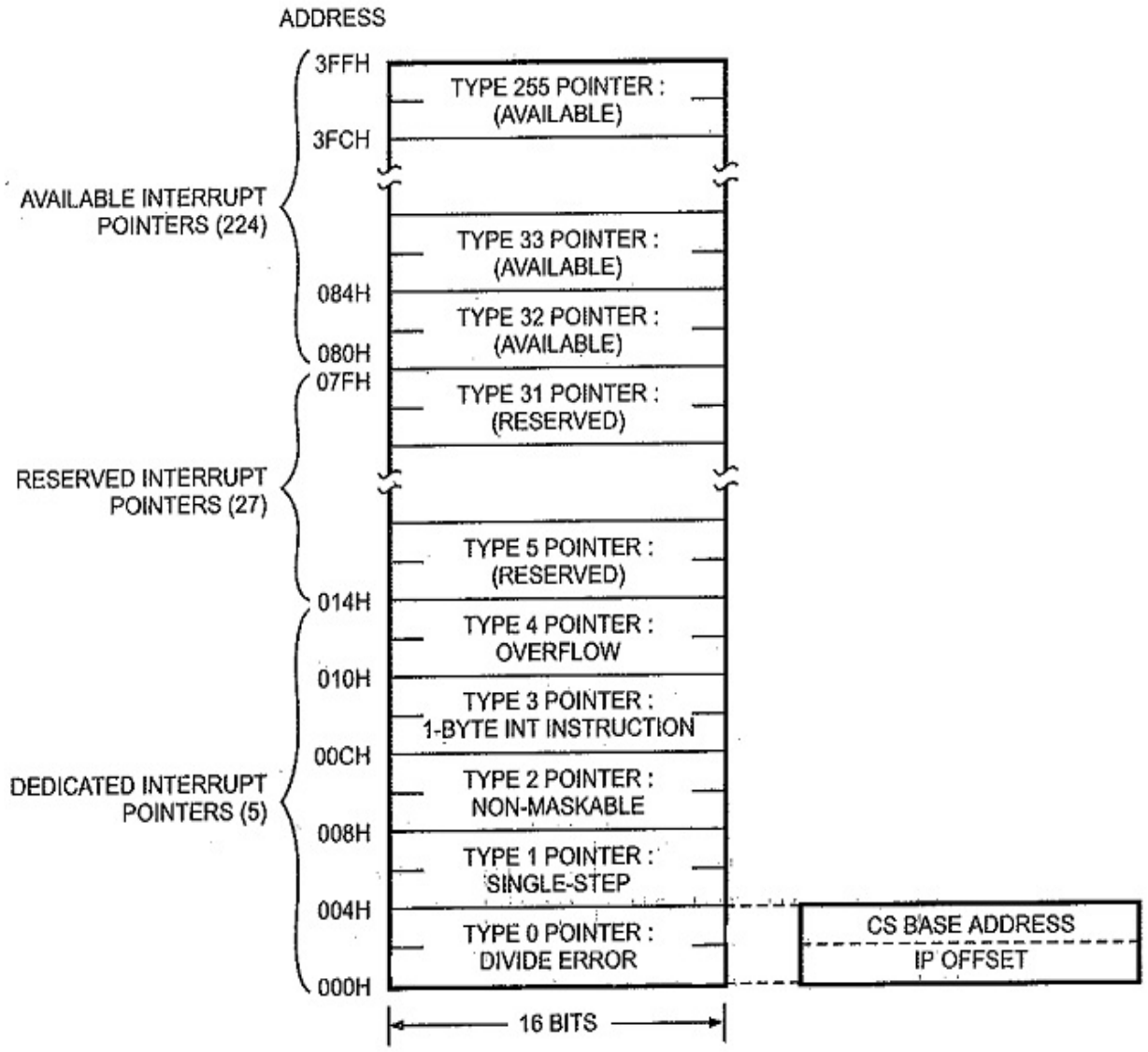


Fig. 9.2 8086 interrupt vector table

8259 Programmable Interrupt Controller

❖**Needed of 8259A PIC:**

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.

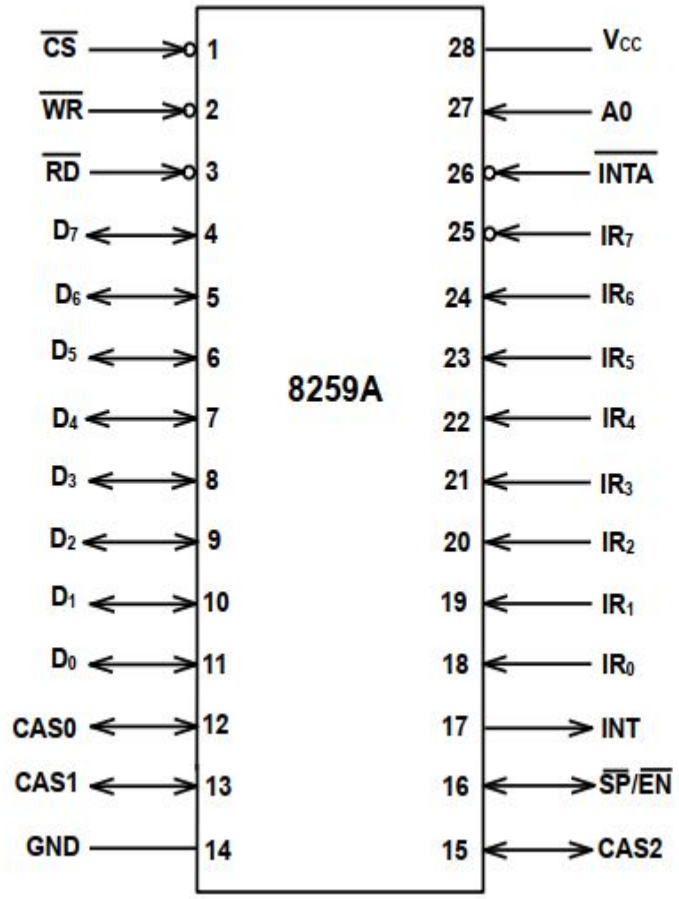
8259 Programmable Interrupt Controller

❖**Features of 8259A PIC:**

- 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 8259As.
- The interrupts can be masked or unmasked individually.
- Available in 28-pin DIP.
- It requires a single +5v supply.

8259 Programmable Interrupt Controller

❖Pin Description of 8259A 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.

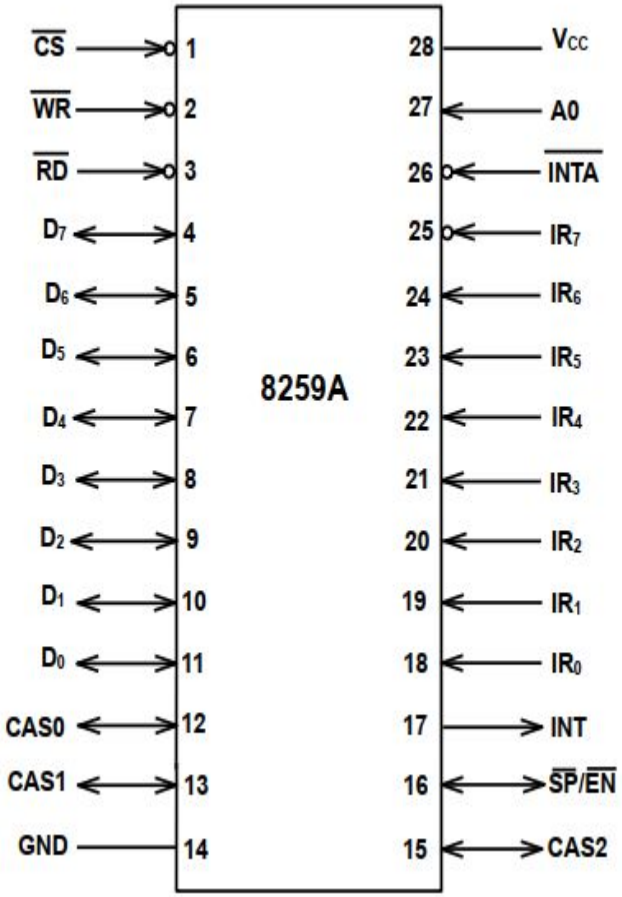
RD**:** A LOW on this input enables the 8259A to send the status of the Interrupt Request Register (IRR), In Service Register (ISR), the Interrupt Mask Register (IMR), or the Interrupt level onto the Data Bus.

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.

8259 Programmable Interrupt Controller

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. 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: 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.

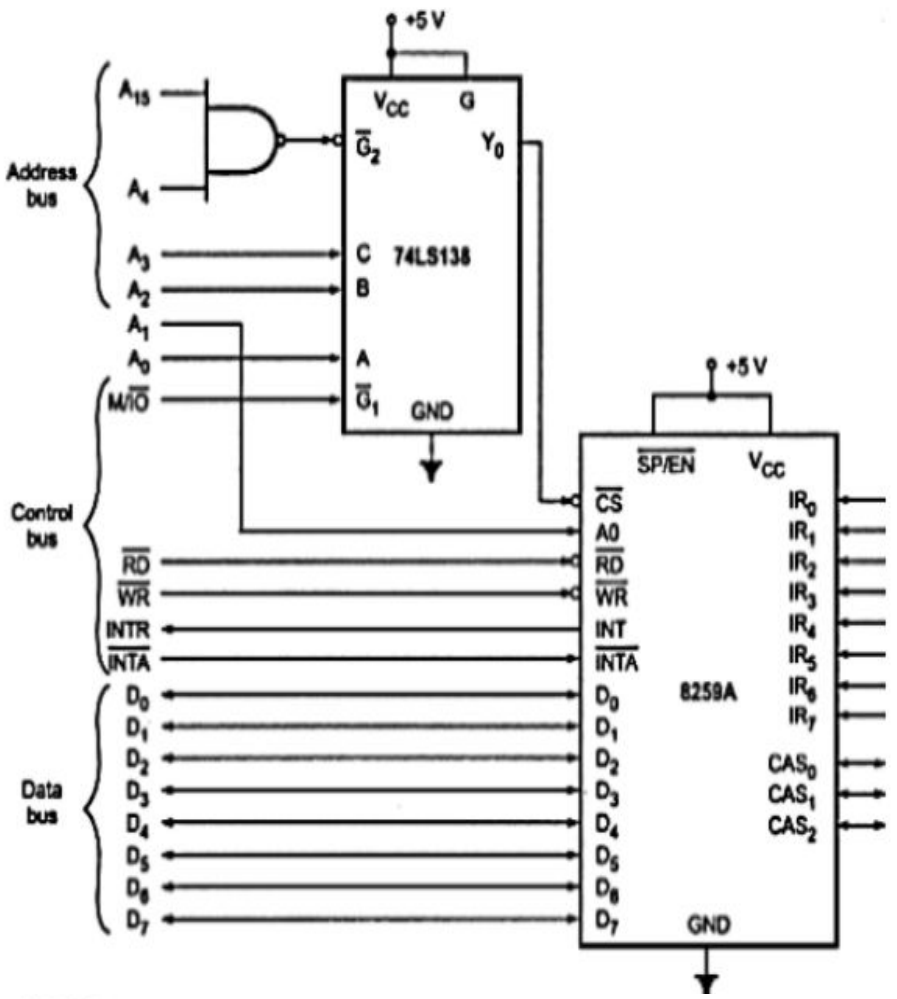
IR7-IR0: Asynchronous inputs. An interrupt request is executed by raising an IR input (low to high), and holding it high until it is acknowledged.

8259 Programmable Interrupt Controller

❖ **Interfacing of 8259 to 8086 microprocessor:**

- When 8086 interrupt flag is set, and the INTR input receives a high signal, the 8086 will-

1. Send out two interrupts acknowledge pulses on its INTA pin to the INTA pin of an 8259A PIC. The INTA pulses tell the 8259A to send the desired interrupt type to the 8086 on the data bus.
2. Multiply the interrupt type it receives from the 8259A by 4 to produce the address in the interrupt vector table.
3. Push the flags on the stack.
4. Clear TF and IF.
5. Push the return addresses on the stack.
6. Get the starting address for the interrupt procedure from the interrupt vector table and load that address in CS and IP.
7. Execute the interrupt service procedure.



Addressing of 8259A :

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Address
1	1	1	1	1	1	1	1	1	1	1	1	0	0	X	0	FFF0H
F				F				F				0/2				FFF2H

8259 Programmable Interrupt Controller

❖ **Block Diagram and functional description of 8259:**

•Data Bus Buffer:

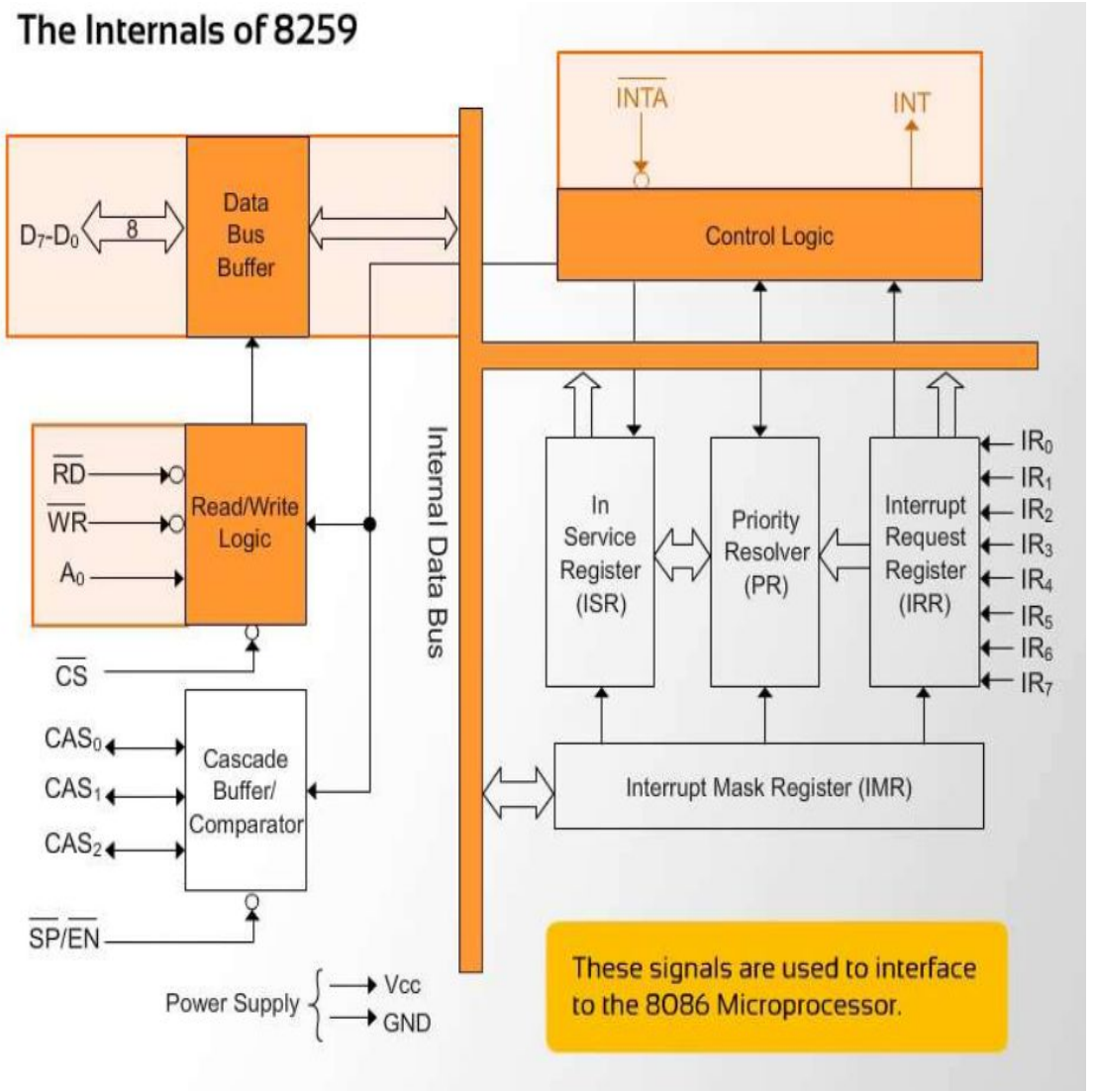
1. This bidirectional 8-bit buffer is used to interface the 8259A to the system Data Bus.
2. Control words and status information are transferred through the Data Bus Buffer.

•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.

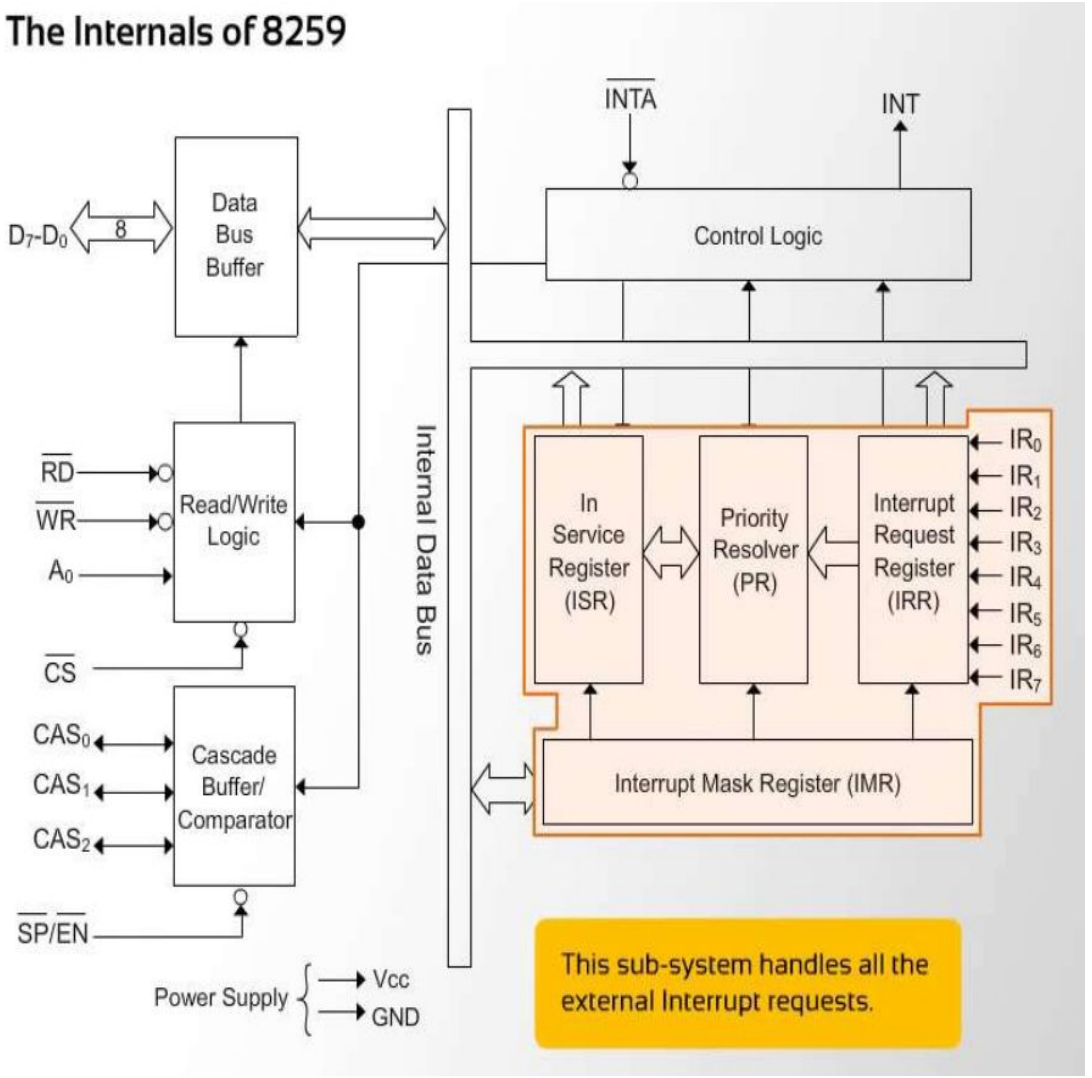


8259 Programmable Interrupt Controller

❖ **Block Diagram and functional description of 8259:**

•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.

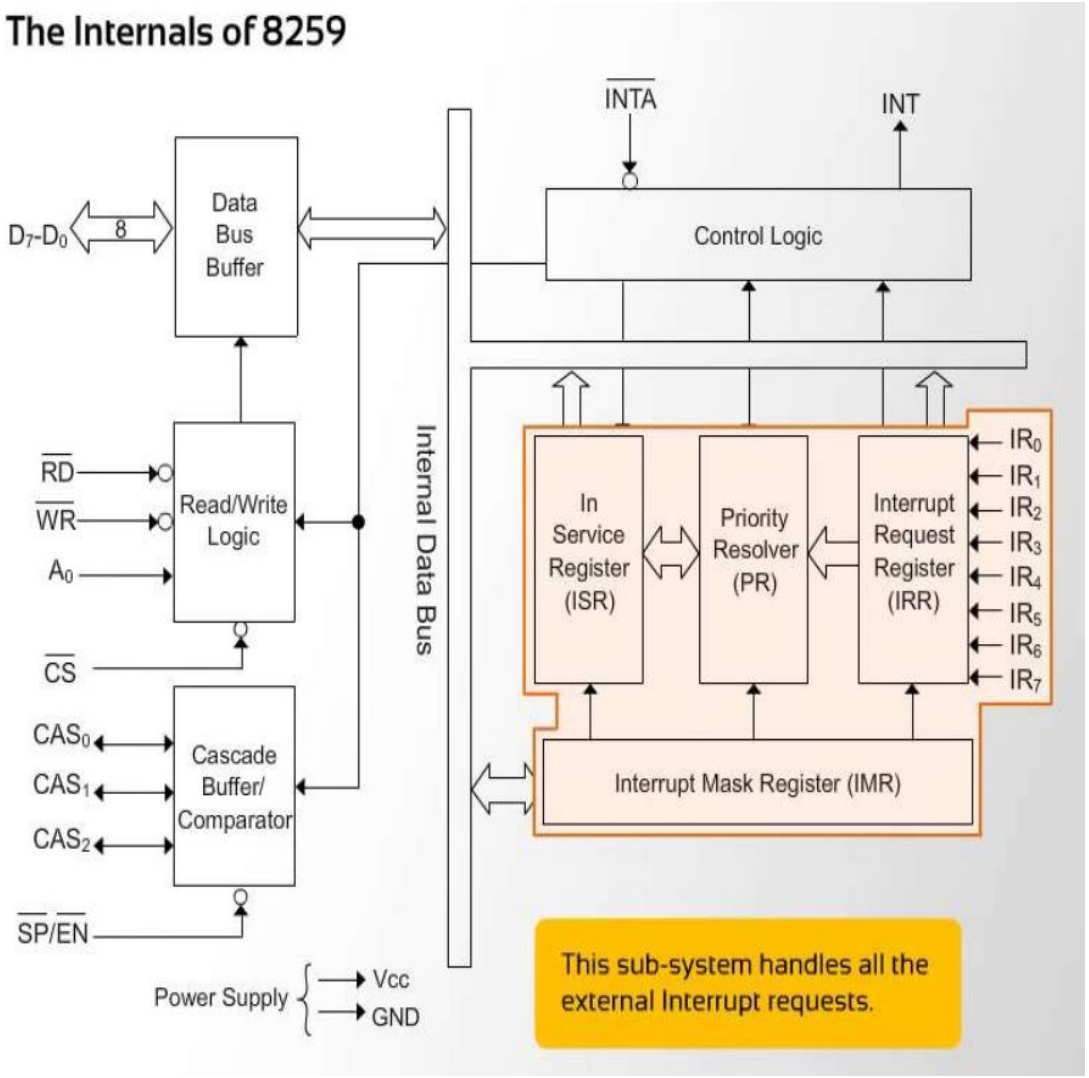


8259 Programmable Interrupt Controller

❖ **Block Diagram and functional description of 8259:**

•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.

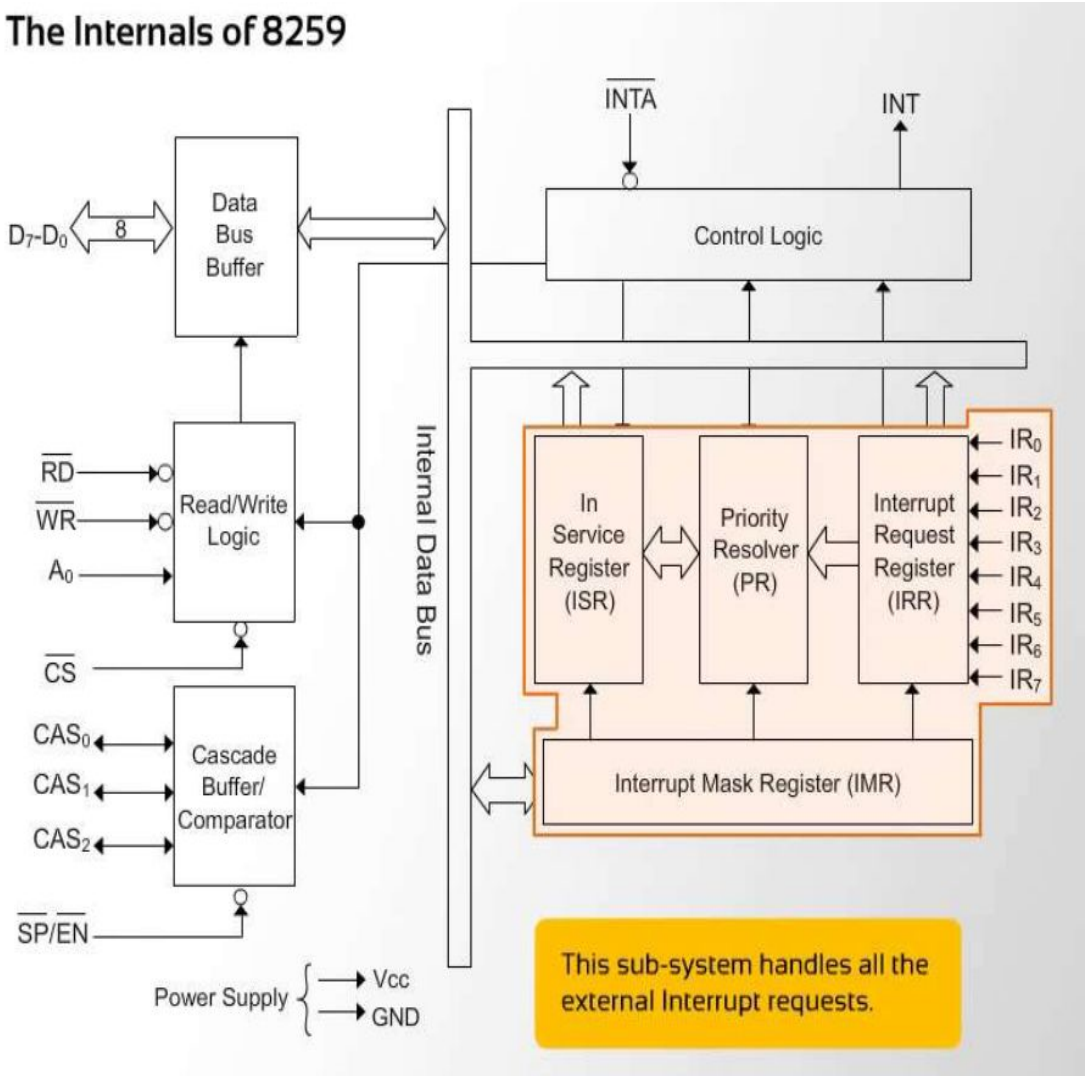


8259 Programmable Interrupt Controller

❖ **Block Diagram and functional description of 8259:**

•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.

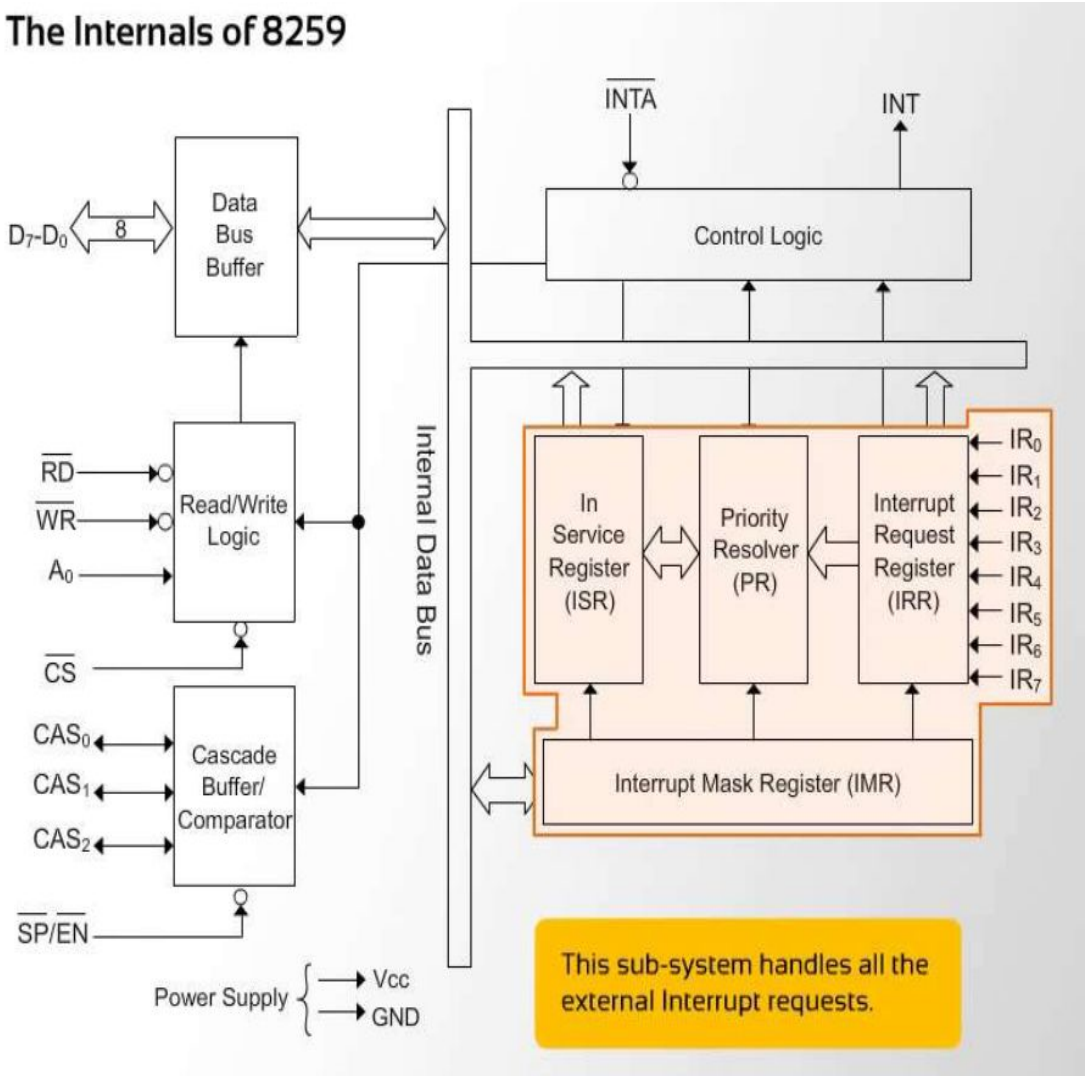


8259 Programmable Interrupt Controller

❖ **Block Diagram and functional description of 8259:**

•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.



8259 Programmable Interrupt Controller

❖**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.

8259 Programmable Interrupt Controller

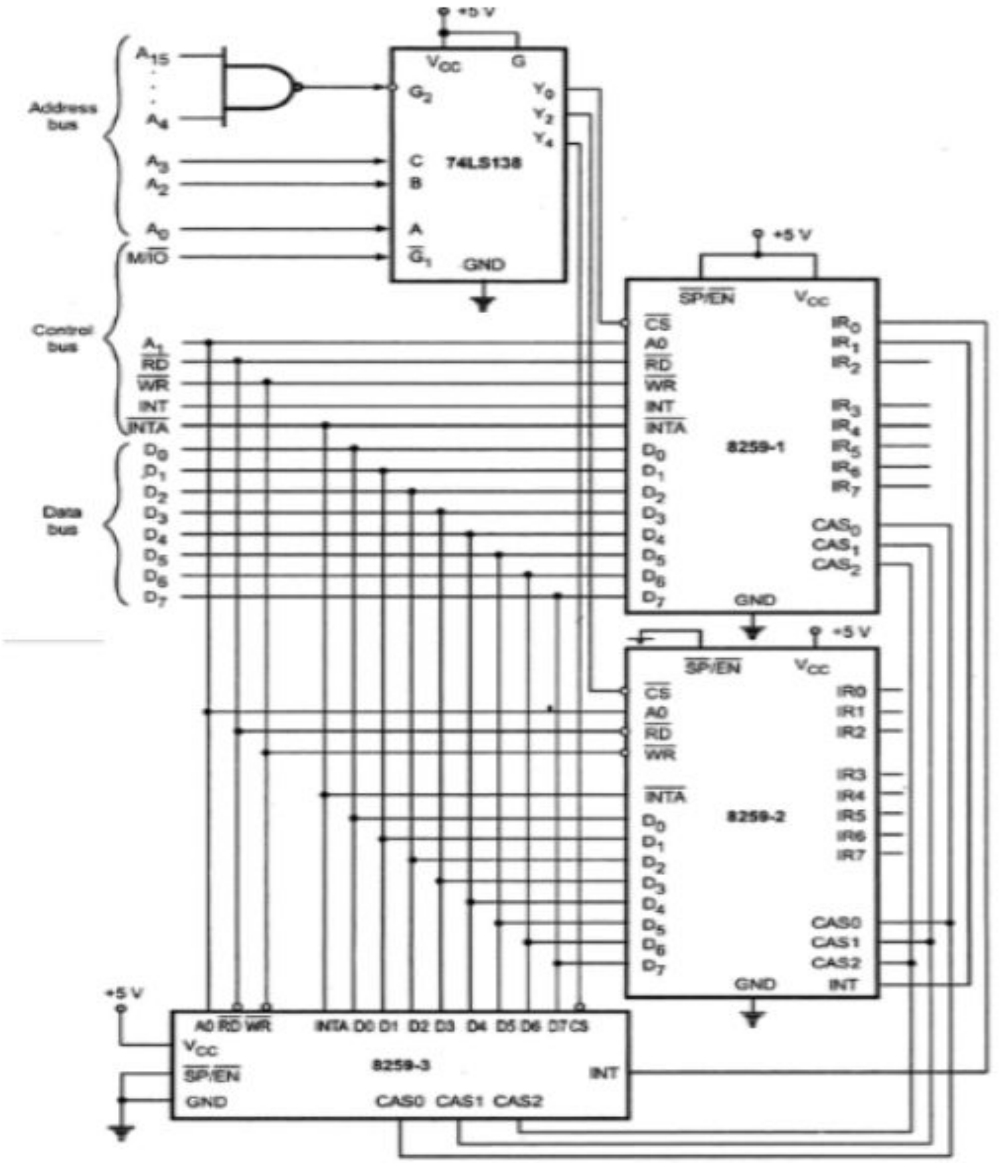
- **CASE 3: Suppose that while the 8086 is executing the IR4 service procedure, an interrupt signal arrives at the IR2 input of the 8259A.**

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.

8259 Programmable Interrupt Controller

❖**Cascading of 8259A:**

•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.

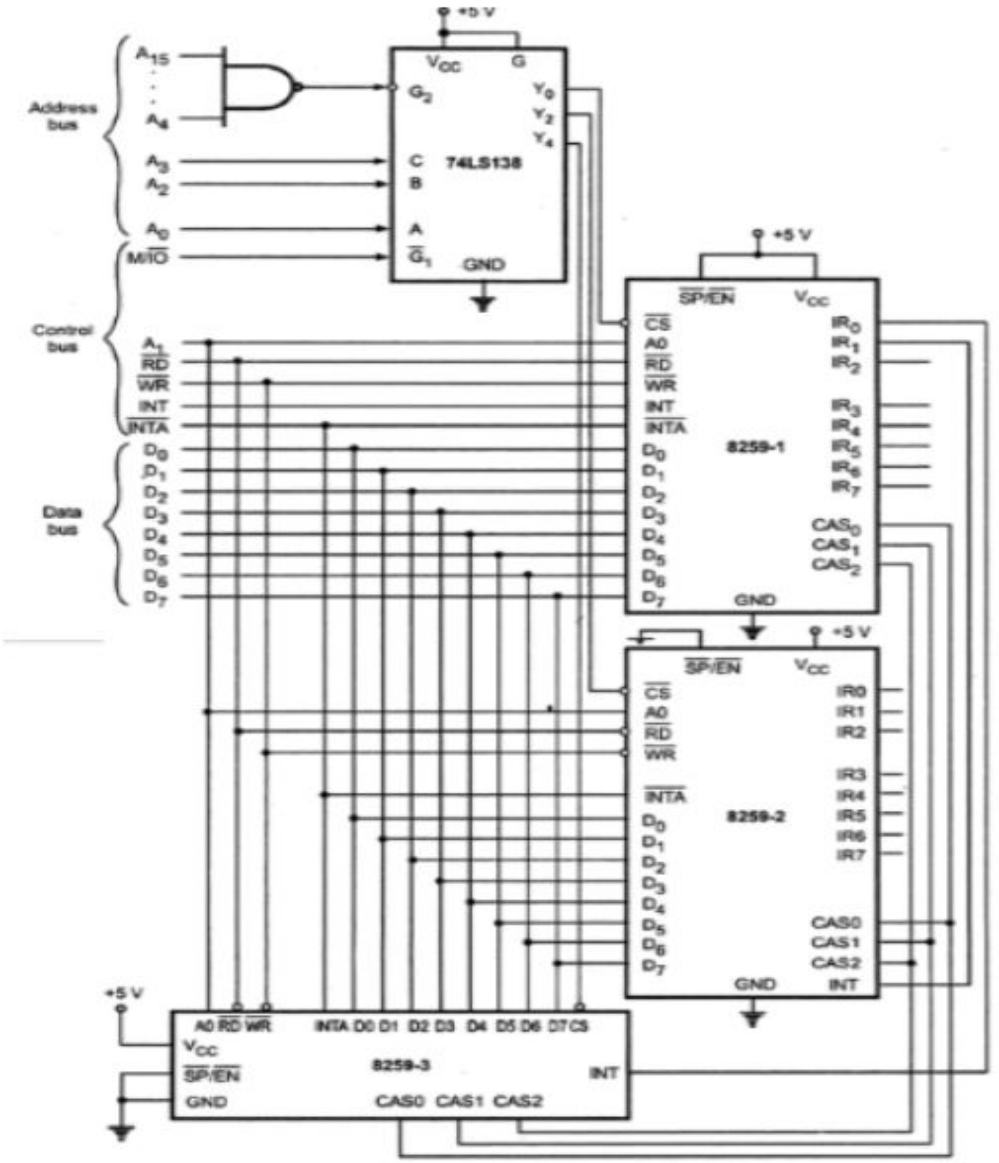


Three 8259 in the cascade mode

8259 Programmable Interrupt Controller

❖**Master and slave operation:**

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.



Three 8259 in the cascade mode

8259 Programmable Interrupt Controller

Address for 8259s

No	A_{15} A_{14} A_{13} A_{12} A_{11} A_{10} A_9 A_8	A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0	Address
8259A-1	1 1 1 1 1 1 1 1 F F	1 1 1 1 0 0 X 0 F 0/2	FFF0H FFF2H
8259A-2	1 1 1 1 1 1 1 1 F F	1 1 1 1 0 1 X 0 F 4/6	FFF4H FFF6H
8259A-3	1 1 1 1 1 1 1 1 F F	1 1 1 1 1 0 X 0 F 8/A	FFF8H FFFAH

8259 Programmable Interrupt Controller