Lesson Nine

8085 Interrupts

Lesson Objectives:

- **To define Interrupt, introduce its types and procedure.**
- **To write a program to manipulate the maskable interrupts**

Pre Test:

- **Define Interrupt and its classification.**
- **What is the function of “EI” and “DI” instructions?**
- **Explain the 8085 Non-Vectored Interrupt Process.**
- **Explain the function of SIM and RIM to manipulated individual masks for RST 5.5, RST 6.5 and RST 7.5.**

Interrupts

❖ Interrupt is a process where an external device can get the attention of the microprocessor. The process starts from the I/O device.

❖ Interrupts can be classified into two types:

Maskable (can be delayed)

Non-Maskable (cannot be delayed)

❖ Interrupts can also be classified into

Vectored (the address of the service routine is hard-wired)

Non-vectored (the address of the service routine needs to be supplied externally)

❖ An interrupt is considered to be an emergency signal. The microprocessor should respond to it as soon as possible.

❖ When the Microprocessor receives an interrupt signal, it suspends the currently executing program and jumps to an Interrupt Service Routine (ISR) to respond to the incoming interrupt. Each interrupt will most probably have its own ISR.

Responding to Interrupts

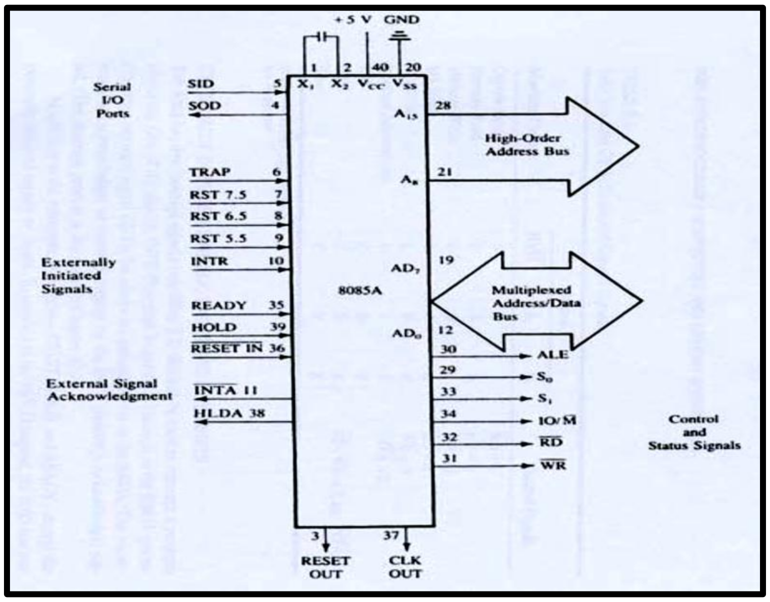
Responding to an interrupt may be immediate or delayed depending on whether the interrupt is maskable or non-maskable and whether interrupts are being masked or not.

There are two ways of redirecting the execution to the ISR depending on whether the interrupt is vectored or non-vectored. The vector is already known to the Microprocessor, while in the non-vectored type, the device will have to supply the vector to the Microprocessor

The 8085 Interrupts

The maskable interrupt process in the 8085 is controlled by a single flip flop inside the microprocessor. This Interrupt Enable flip flop is controlled using the two instructions “EI” and “DI”.

The 8085 has a single Non-Maskable interrupt. The non-maskable interrupt is not affected by the value of the Interrupt Enable flip flop.

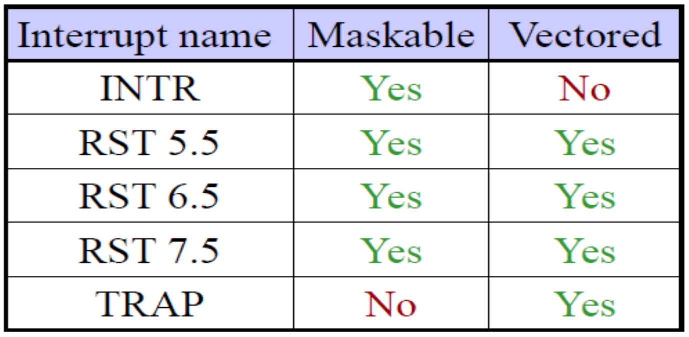


The 8085 has 5 interrupt inputs:

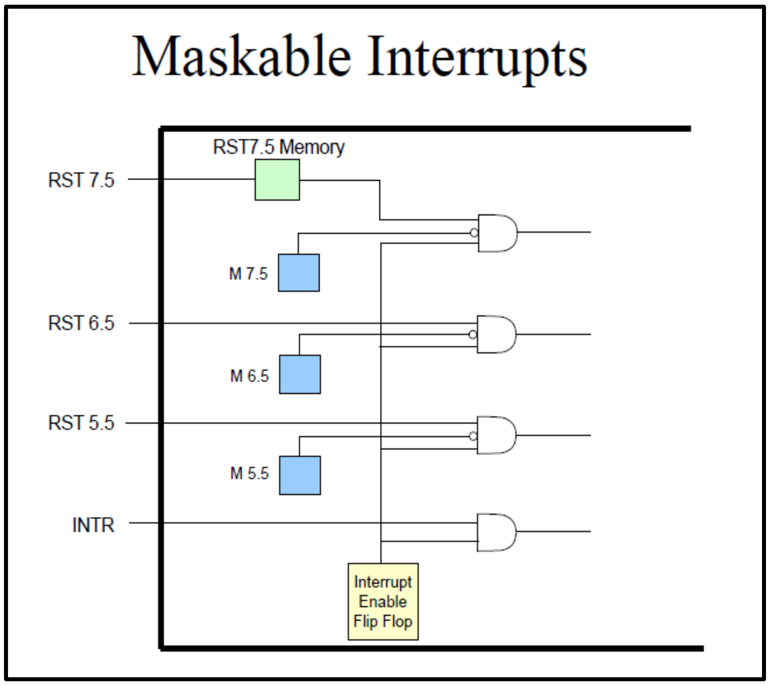
1)The INTR input. The INTR input is the only non-vectored interrupt. INTR is maskable using the EI/DI instruction pair.

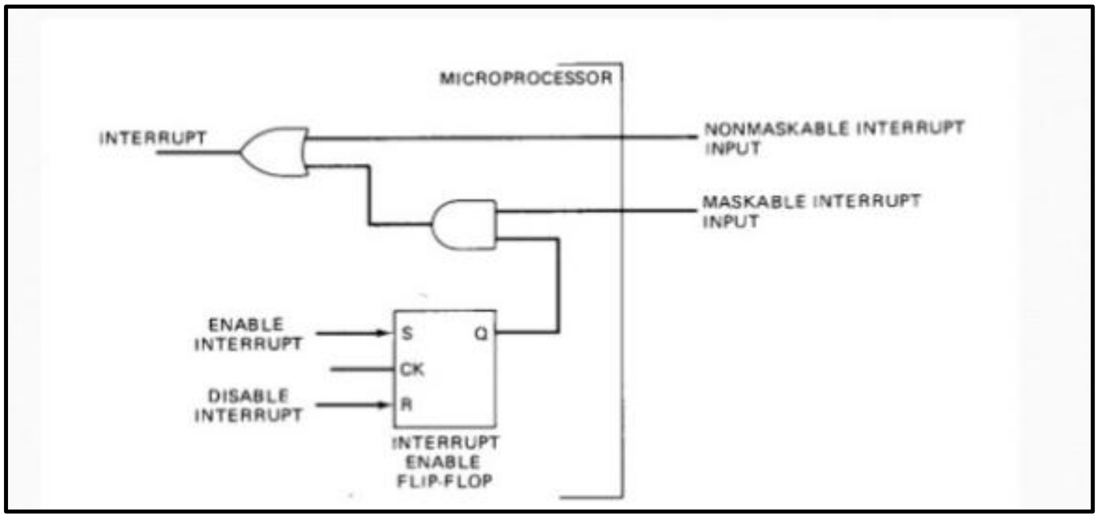
2) RST 5.5, RST 6.5, RST 7.5 are all automatically vectored. RST 5.5, RST 6.5, and RST 7.5 are all maskable.

3)TRAP is the only non-maskable interrupt in the 8085TRAP is also automatically vectored.



Interrupt name	Maskable	Vectored
INTR	Yes	No
RST 5.5	Yes	Yes
RST 6.5	Yes	Yes
RST 7.5	Yes	Yes
TRAP	No	Yes





An interrupt vector is a pointer to where the ISR is stored in memory.

All interrupts (vectored or otherwise) are mapped onto a memory area called the Interrupt Vector Table (IVT). The IVT is usually located in memory page 00(0000H -00FFH).

The purpose of the IVT is to hold the vectors that redirect the microprocessor to the right place when an interrupt arrives. The IVT is divided into several blocks. Each block is used by one of the interrupts to hold its "vector"

The 8085 Non-Vectored Interrupt Procedure

1. The interrupt process should be enabled using the EI instruction.

2. The 8085 checks for an interrupt during the execution of every instruction.

3. If there is an interrupt, the microprocessor will complete the executing instruction, and start a RESTART sequence.

4. The RESTART sequence resets the interrupt flip flop and activates the interrupt acknowledge signal (INTA).

5. Upon receiving the INTA signal, the interrupting device is expected to return the op-code of one of the 8 RST instructions.

6. When the microprocessor executes the RST instruction received from the device, it saves the address of the next instruction on the stack and jumps to the appropriate entry in the IVT.

7. The IVT entry must redirect the microprocessor to the actual service routine.

8. The service routine must include the instruction EI to re-enable the interrupt process.

9. At the end of the service routine, the RET instruction returns the execution to where the program was interrupted.

❖ The 8085 recognizes 8 RESTART instructions: RST0 -RST7.each of these would send the execution to a predetermined hard-wired memory location:

Restart instruction	OP code	Binary code D_7 D_6 $\mathbf{D_5}$ $\mathbf{D_4}$ $\mathbf{D_3}$ D_2 D_1 D_0	Memory location
RST0	C7	1 1 **0 0 0** 1 1 1	0000
RST1	CF	1 1 **0 0 1** 1 1 1	0008
RST2	D7	1 1 **0 1 0** 1 1 1	0010
RST3	DF	1 1 **0 1 1** 1 1 1	0018
RST4	E7	1 1 **1 0 0** 1 1 1	0020
RST5	EF	1 1 **1 0 1** 1 1 1	0028
RST6	F7	1 1 **1 0 1** 1 1 1	0030
RST7	FF	1 1 **1 1 1** 1 1 1	0038

Restart Sequence

The restart sequence is made up of three machine cycles.

In the 1st machine cycle: The microprocessor sends the INTA signal.

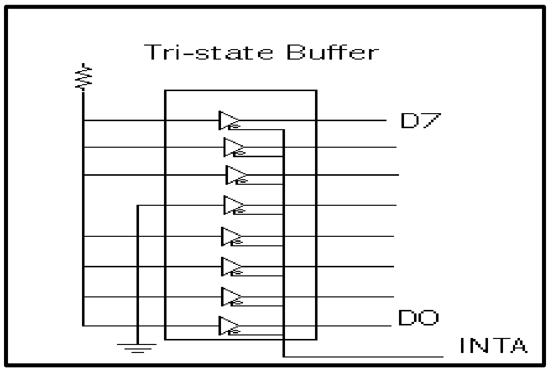
While INTA is active the microprocessor reads the data lines expecting to receive, from the interrupting device, the opcode for the specific RST instruction.

In the 2nd and 3rd machine cycles: the 16-bit address of the next instruction is saved on the stack. Then the microprocessor jumps to the address associated with the specified RST instruction.

EX: RST5

During the interrupt acknowledge machine cycle, (the 1st machine cycle of the RST operation): The Microprocessor activates the INTA signal. This signal will enable the Tri-state buffers, which will place the value EFH on the data bus. Therefore, sending the Microprocessor the RST 5 instruction.

The RST 5 instruction is exactly equivalent to CALL 0028H



Issues in Implementing INTR Interrupts

1) How long must INTR remain high?

The microprocessor checks the INTR line one clock cycle before the last T-state of each instruction. The interrupt process is Asynchronous.

The INTR must remain active long enough to allow for the longest instruction. The longest instruction for the 8085 is the conditional CALL instruction which requires 18 T-states.

Therefore, the INTR must remain active for 17.5 T-states.

2) How long can the INTR remain high?

The INTR line must be deactivated before the EI is executed. Otherwise, the microprocessor will be interrupted again.

The worst case situation is when EI is the first instruction in the ISR.

Once the microprocessor starts to respond to an INTR interrupt, INTA becomes active (=0).

Therefore, INTR should be turned off as soon as the INTA signal is received

3) Can the microprocessor be interrupted again before the completion of the ISR?

As soon as the 1st interrupt arrives, all maskable interrupts are disabled.

They will only be enabled after the execution of the EI instruction.

Therefore, the answer is: “only if you allow it to”.

If the EI instruction is placed early in the ISR, other interrupt may occur before the ISR is done.

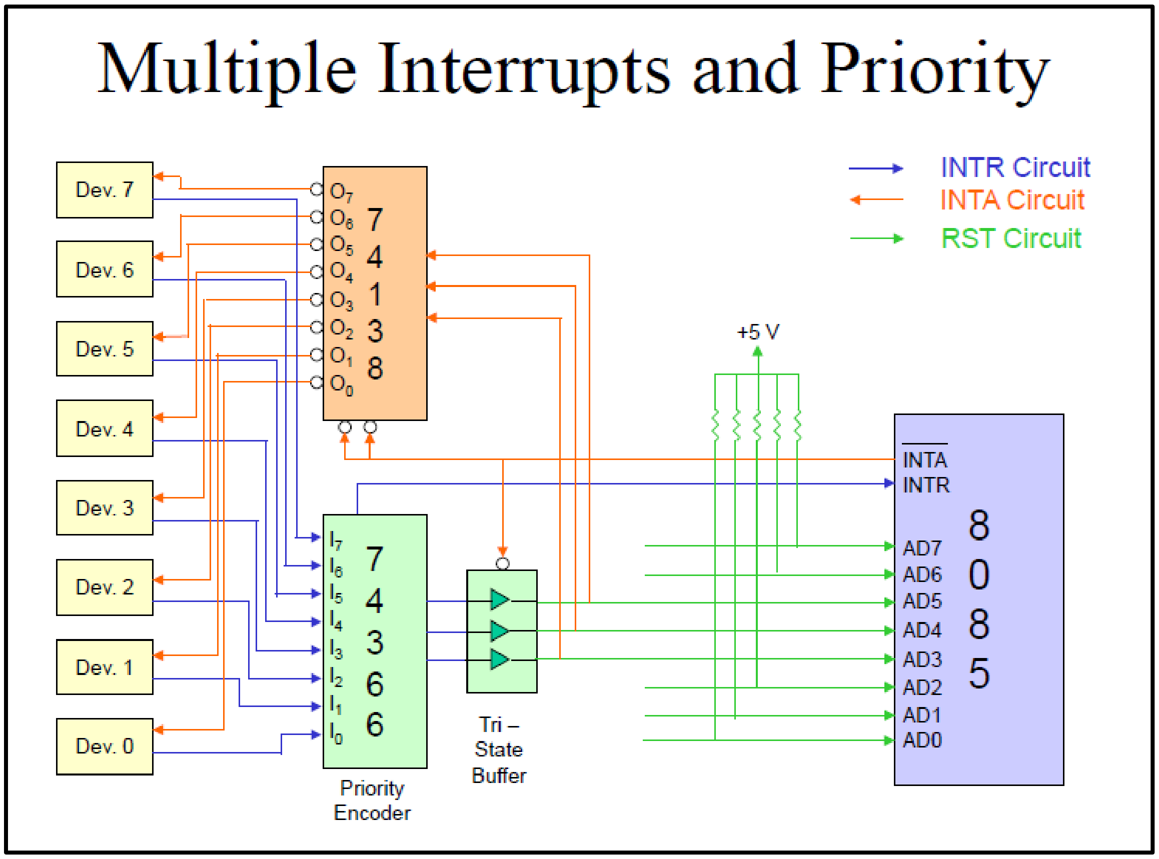
Multiple Interrupts & Priorities

How do we allow multiple devices to interrupt using the INTR line?

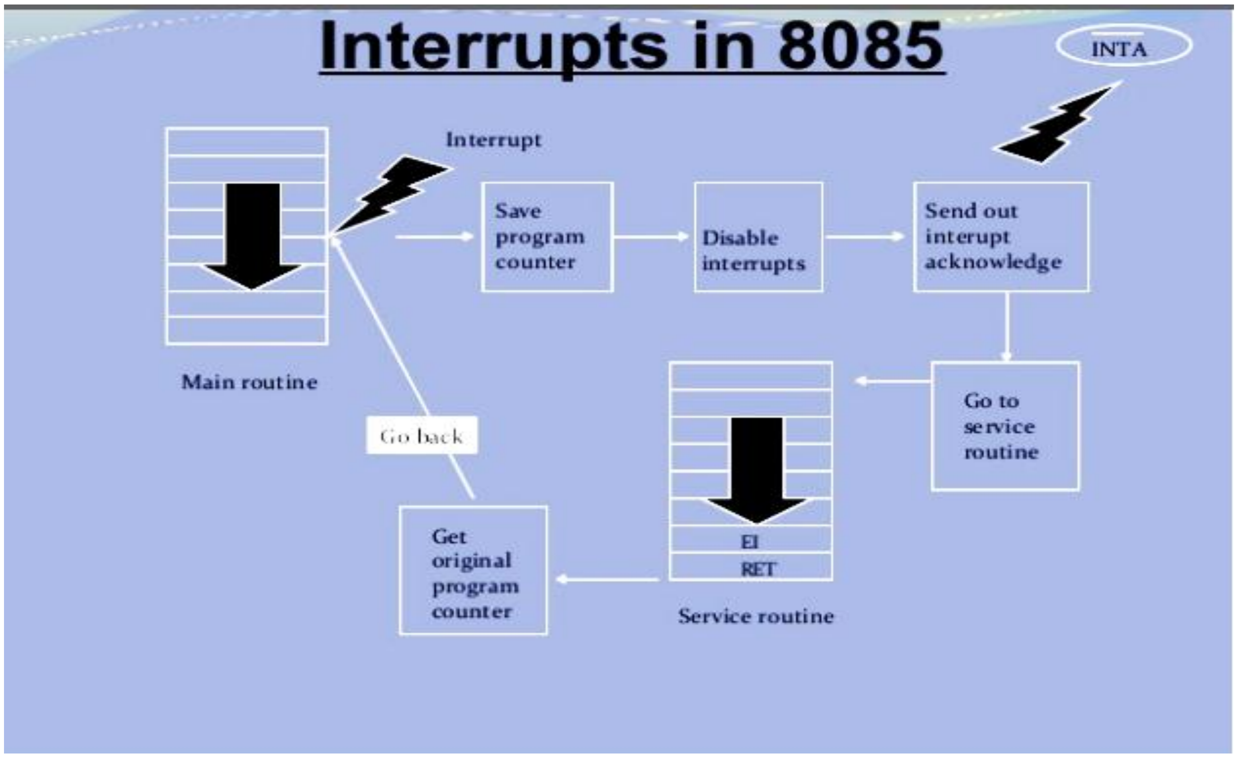
The microprocessor can only respond to one signal on INTR at a time. Therefore, we must allow the signal from only one of the devices to reach the microprocessor. We must assign some priority to the different devices and allow their signals to reach the microprocessor according to the priority.

Note that the opcodes for the different RST instructions follow a set pattern. Bit D5, D4 and D3 of the opcodes change in a binary sequence from RST 7 down to RST 0.The other bits are always 1.

This allows the code generated by the 74366 to be used directly to choose the appropriate RST instruction. The one drawback to this scheme is that the only way to change the priority of the devices connected to the 74366 is to reconnect the hardware.

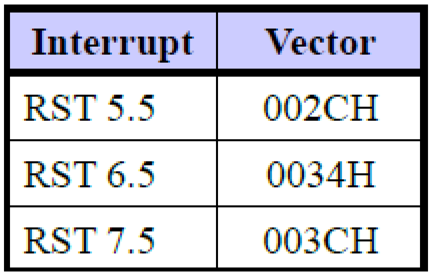


.


Interrupts in 8085
INTA
Interrupt
Save program counter
Disable interrupts
Send out interupt acknowledge
Go to service routine
Main routine
Go back
Get original program counter
EI
RET
Service routine

The 8085 Maskable/Vectored Interrupts

The 8085 has 4 Masked /Vectored interrupt inputs. RST 5.5, RST 6.5, RST 7.5. They are all maskable and are vectored automatically to the following table: The vectors for these interrupt fall in between the vectors for the RST instructions. That's why they have names like RST 5.5 (RST 5 and a half)



Interrupt	Vector
RST 5.5	002CH
RST 6.5	0034H
RST 7.5	003CH

Masking RST 5.5, RST 6.5 and RST 7.5

These three interrupts are masked at two levels:

1) Through the Interrupt Enable flip flop and the EI/DI instructions. The Interrupt Enable flip flop controls the whole maskable interrupt process.

2) Through individual mask flip flop that control the availability of the individual interrupts. These flip flops control the interrupts individually.

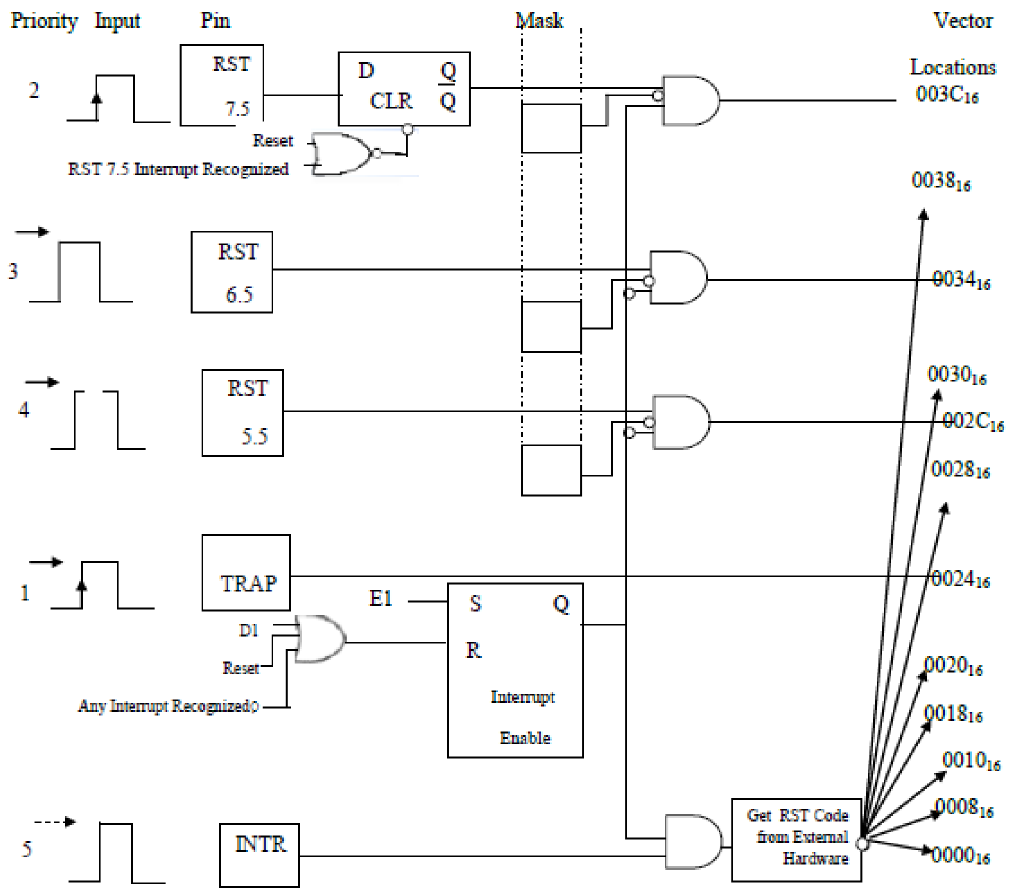


Figure (12.1) The 8085 Interrupts and Vector Locations.

The 8085 Maskable/Vectored Interrupt Process

The interrupt process should be enabled using the EI instruction.

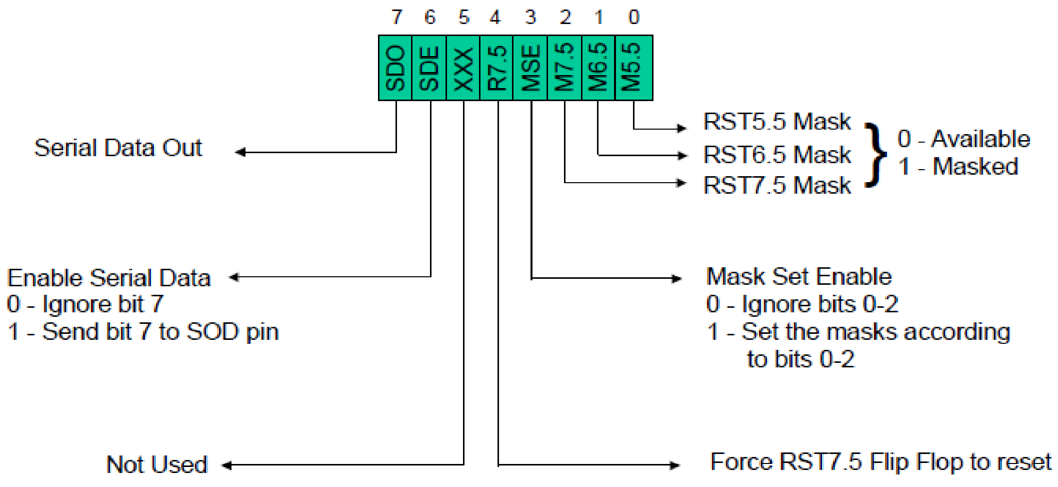
The 8085 checks for an interrupt during the execution of every instruction. If there is an interrupt, and if the interrupt is enabled using the interrupt mask, the microprocessor will complete the executing instruction, and reset the interrupt flip flop. The microprocessor then executes a call instruction that sends the execution to the appropriate location in the interrupt vector table.

When the microprocessor executes the call instruction, it saves the address of the next instruction on the stack. The microprocessor jumps to the specific service routine. The service routine must include the instruction EI to re-enable the interrupt process. At the end of the service routine, the RET instruction returns the execution to where the program was interrupted.

Manipulating the Masks

- The Interrupt Enable flip flop is manipulated using the EI/DI instructions.
- The individual masks for RST 5.5, RST 6.5 and RST 7.5 are manipulated using the SIM instruction. This instruction takes the bit pattern in the Accumulator and applies it to the interrupt mask enabling and disabling the specific interrupts.

SIM Instruction



The RST 7.5 interrupt is the only 8085 interrupt that has memory. If a signal on RST7.5 arrives while it is masked, a flip flop will remember the signal. When RST7.5 is unmasked, the microprocessor will be interrupted even if the device has removed the interrupt signal.

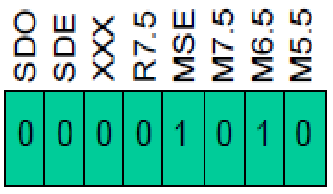
This flip flop will be automatically reset when the microprocessor responds to an RST 7.5 interrupt. Bit 4 of the accumulator in the SIM instruction allows explicitly resetting the RST 7.5 memory even if the microprocessor did not respond to it.

The SIM instruction can also be used to perform serial data transmission out of the 8085's SOD pin. One bit at a time can be sent out serially over the SOD pin. Bit 6 is used to tell the microprocessor whether or not to perform serial data transmission If 0, then do not perform serial data transmission If 1, then do. The value to be sent out on SOD has to be placed in bit 7 of the accumulator. Bit 5 is not used by the SIM instruction

Example: Set the interrupt masks so that RST5.5 is enabled, RST6.5 is masked, and RST7.5 is enabled.

Sol.: First, determine the contents of the accumulator

- Enable 5.5	bit 0 = 0
- Disable 6.5	bit 1 = 1
- Enable 7.5	bit 2 = 0
- Allow setting the masks	bit 3 = 1
- Don't reset the flip flop	bit 4 = 0
- Bit 5 is not used	bit 5 = 0
- Don't use serial data	bit 6 = 0
- Serial data is ignored	bit 7 = 0



Contents of accumulator are: 0AH

```
EI          ; Enable interrupts including INTR
MVI A, 0A   ; Prepare the mask to enable RST 7.5, and 5.5, disable 6.5
SIM         ; Apply the settings RST masks
```

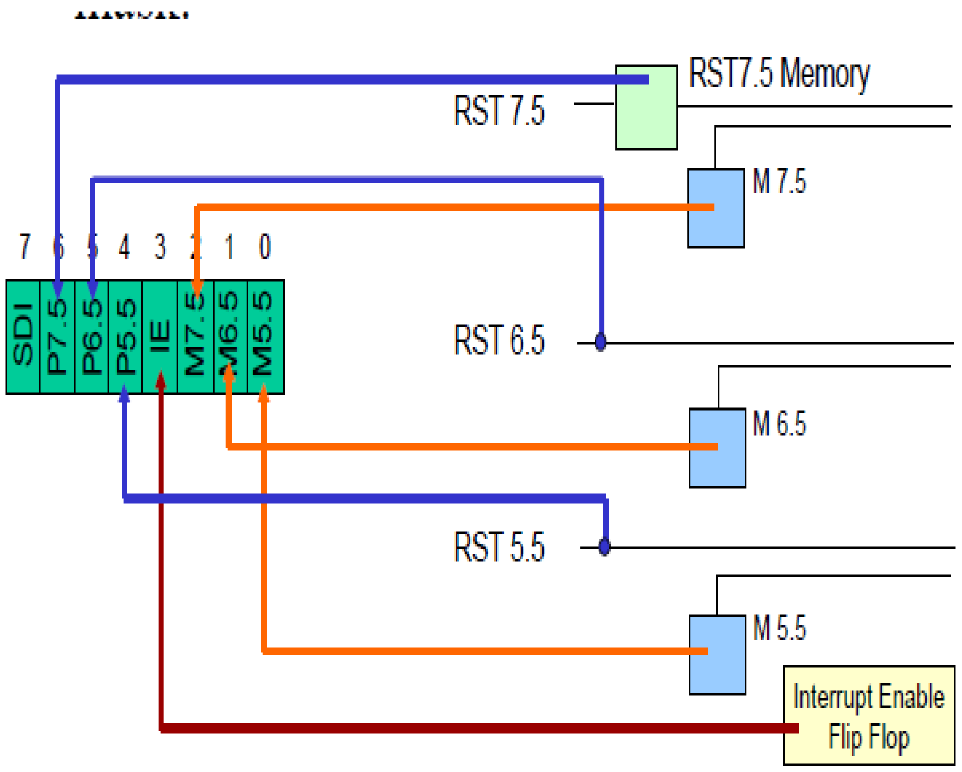
Triggering Levels

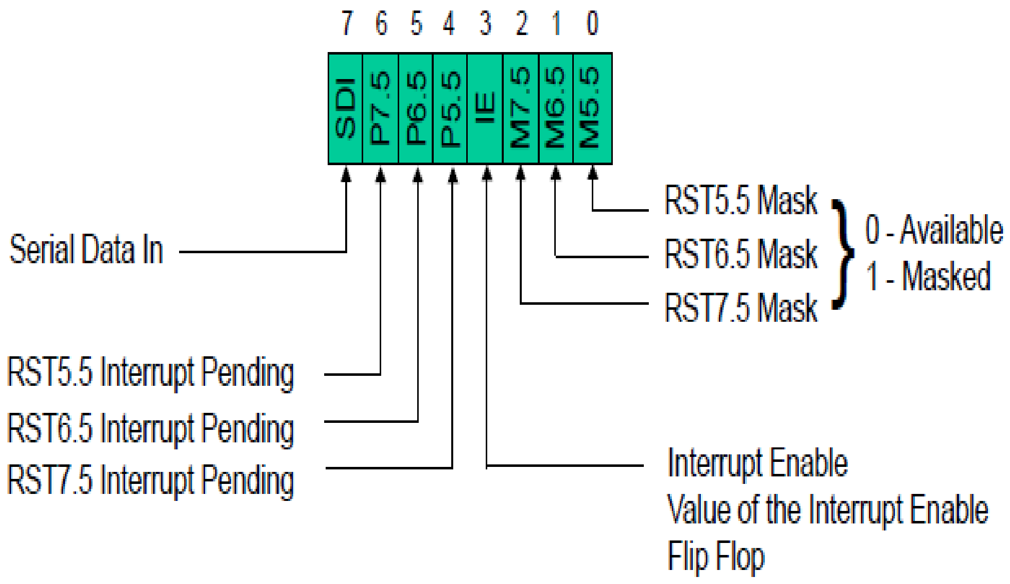
RST 7.5 is positive edge sensitive. When a positive edge appears on the RST7.5 line, logic 1 is stored in the flip-flop as a "pending" interrupt. Since the value has been stored in the flip flop, the line does not have to be high when the microprocessor checks for the interrupt to be recognized. The line must go to zero and back to one before a new interrupt is recognized.

RST 6.5 and RST 5.5 are level sensitive. The interrupting signal must remain present until the microprocessor checks for interrupts.

Determining the Current Mask Settings

RIM instruction: Read Interrupt Mask Load the accumulator with an 8-bit pattern showing the status of each interrupt pin and mask.





<u>The RIM Instruction and the Masks</u>

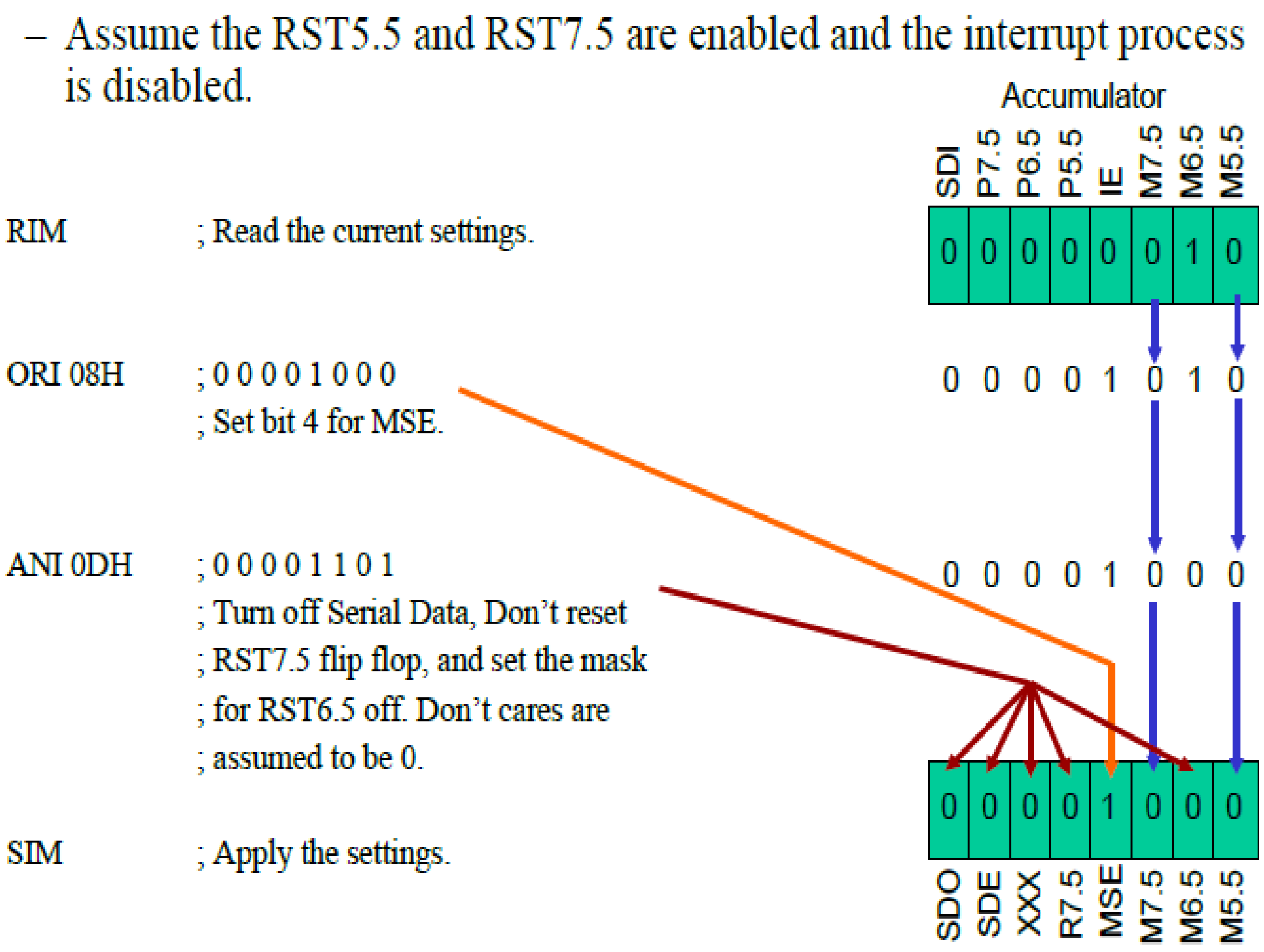
Bits 0-2 show the current setting of the mask for each of RST 7.5, RST 6.5 and RST 5.5They return the contents of the three mask flip flops.

They can be used by a program to read the mask settings in order to modify only the right mask.

Bit 3 shows whether the maskable interrupt process is enabled or not. It returns the contents of the Interrupt Enable Flip Flop. It can be used by a program to determine whether or not interrupts are enabled.

Bits 4-6 show whether or not there are pending interrupts on RST 7.5, RST 6.5, and RST 5.5Bits 4 and 5 return the current value of the RST5.5 and RST6.5 pins. Bit 6 returns the current value of the RST7.5 memory flip flop.

Bit 7 is used for Serial Data Input. The RIM instruction reads the value of the SID pin on the microprocessor and returns it in this bit.

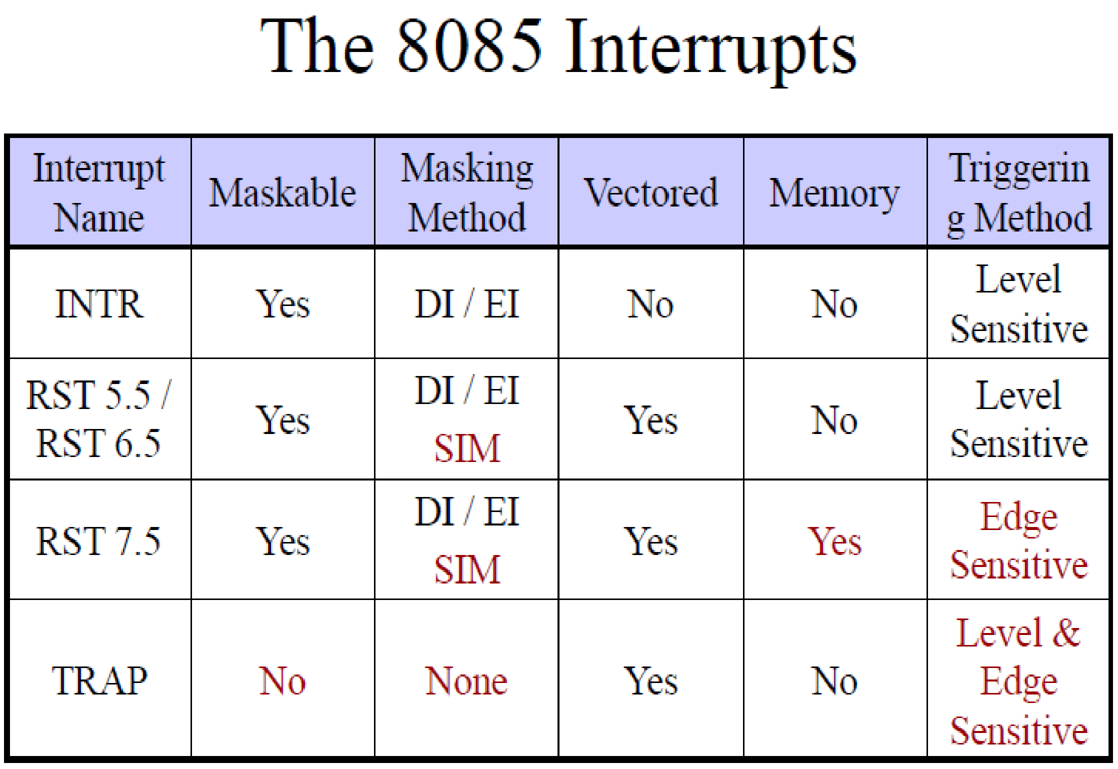


– Assume the RST5.5 and RST7.5 are enabled and the interrupt process is disabled.

```
RIM        ; Read the current settings.

ORI 08H    ; 0 0 0 0 1 0 0 0
           ; Set bit 4 for MSE.

ANI 0DH    ; 0 0 0 0 1 1 0 1
           ; Turn off Serial Data, Don't reset
           ; RST7.5 flip flop, and set the mask
           ; for RST6.5 off. Don't cares are
           ; assumed to be 0.

SIM        ; Apply the settings.
```

TRAP

TRAP is the only non-maskable interrupt. It does not need to be enabled because it cannot be disabled. It has the highest priority amongst interrupts. It is edge and level sensitive. It needs to be high and stay high to be recognized. Once it is recognized, it won't be recognized again until it goes low, then high again. TRAP is usually used for power failure and emergency shutoff.



The 8085 Interrupts

Interrupt Name	Maskable	Masking Method	Vectored	Memory	Triggering Method
INTR	Yes	DI / EI	No	No	Level Sensitive
RST 5.5 / RST 6.5	Yes	DI / EI SIM	Yes	No	Level Sensitive
RST 7.5	Yes	DI / EI SIM	Yes	Yes	Edge Sensitive
TRAP	No	None	Yes	No	Level & Edge Sensitive

Internally, the 8085 implements an interrupt priority scheme. The interrupts are ordered as follows:

TRAP
RST 7.5
RST 6.5
RST 5.5
INTR

However, TRAP has lower priority than the HLD signal used for DMA.

Programmable Interrupt Controller 8259 A

A programmable interrupt managing device. It manages 8 interrupt requests. It can vector an interrupt any wherein memory without additional H/W. It can support 8 levels of interrupt priorities.

The priority scheme can be extended to 64 levels using a hierarchy 0f 8259 device.

The 8085 INTR interrupt scheme presented earlier has a few limitations:

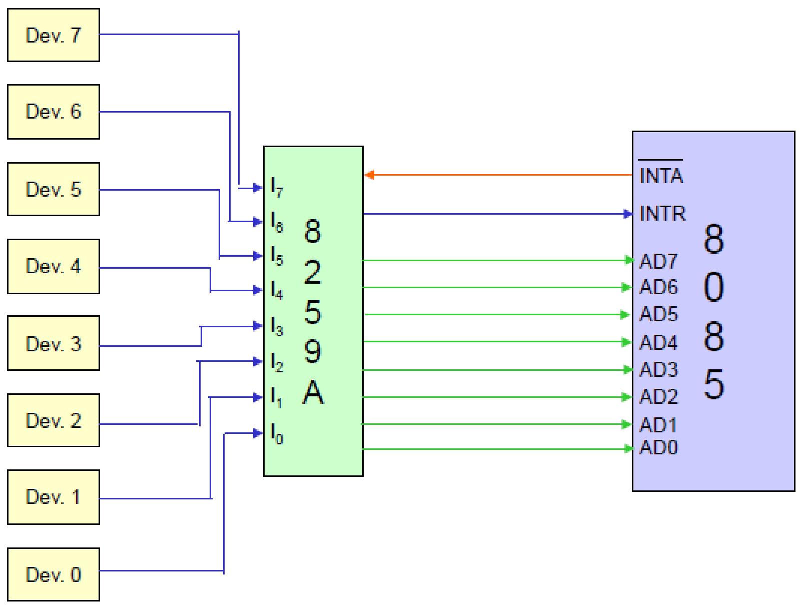
1) The RST instructions are all vectored to memory page 00H, which is usually used for ROM.

2) It requires additional hardware to produce the RST instruction opcodes.

3) Priorities are set by hardware.

Therefore, we need a device like the 8259A to expand the priority scheme and allow mapping to pages other than 00H.

Interfacing the 8259A to the 8085



Operating of the 8259A

1) When one or more interrupts come in. The 8259A resolves the interrupt priorities based on its internal settings.

2) The 8259A sends an INTR signal to the microprocessor. The microprocessor responds with an INTA signal and turns off the interrupt enable flip flop.

3) The 8259A responds by placing the op-code for the CALL instruction (CDH) on the data bus.

4) When the microprocessor receives the op-code for CALL instead of RST, it recognizes that the device will be sending 16 more bits for the address.

5) The microprocessor sends a second INTA signal. The 8259A sends the high order byte of the ISR's address. The microprocessor sends a third INTA signal. The 8259A sends the low order byte of the ISR's address.

6) The microprocessor executes the CALL instruction and jumps to the ISR.

Direct Memory Access

This is a process where data is transferred between two peripherals directly without the involvement of the microprocessor. This process employs the HOLD pin on the microprocessor. The external DMA controller sends a signal on the HOLD pin to the microprocessor.

The microprocessor completes the current operation and sends a signal on HLDA and stops using the buses.

Once the DMA controller is done, it turns off the HOLD signal and the microprocessor takes back control of the buses.

H.W. Write a program to Set the interrupt masks so that RST6.5 is enabled, RST5.5 is masked, and RST7.5 is masked.

H.W. **How can multiple devices to interrupt using the INTR line?**
H.W. **Define DMA, Vectored interrupt, TRAP.**

H.W. **Illustrate the procedure of the 8085 Non-Vectored Interrupt.**

Reference:

8085 µp architecture and programming_Gonkar