Interrupts

Interrupts

- Interrupt is a process where an external device can get the attention of the microprocessor.
  - The process starts from the I/O device
  - The process is asynchronous, means can occur at any time during execution of program.
- In order to communicate with µP & I/O devices either **Polling** or **Interrupt method is used.**
- An interrupt is considered to be an emergency signal. The Microprocessor should respond to it as soon as possible.

1. Polling Method

- ✓ In polling, µP polls i.e. ask each device in sequence whether it is ready for communication (datatransfer).
- ✓ If device is ready, then data transfer takes place between device & µP.
- ✓ If device is not ready or completed its data transfer, then µP asks the next device in chain.
- ✓ Main disadvantage of this method is that most of the time µP remains busy in polling. So some useful tasks get less time to execute.
- ✓ This method is useful only if µP has contains few I/O devices.

2. Interrupt Method

- ✓ Interrupt is signal send by an external device to the microprocessor to request the processor to perform a particular task or work.
- ✓ It is a simple routine program that keeps a check for the occurrence of the interrupt.
- ✓ Mainly in the microprocessor based system the interrupts are used for data transfer between the peripheral (I/O) and the microprocessor.
- ✓ If the µP accept the interrupt and send the INTA (active low) signal to the peripheral.
- ✓ When interrupt is received, µP suspends its current activity and upon completion, it resumes the suspended activity.
- ✓ The processor executes an interrupt service routine (ISR) addressed in program counter.
- ✓ It returned to main program by RET instruction.
- ✓ Advantage is that µP need not waste time in polling the devices.

Interrupt Process

1. When the MPU is executing a program it checks all the interrupt lines during the execution of each instruction.
2. If any Interrupt line is enables, the processor completes the current going instruction execution.
3. If more than one lines are enabled simultaneously then the processor pick up the request which have the highest priority and all other are discarded.
4. After completion of the current instruction execution, processor checks for the respective conditions for the activated interrupt or selected interrupt in case of more than one.
5. If condition are not favorable then request is discarded or stored or if the condition are favorable then the processor generates an external INTA or internal acknowledges signal to insert a RST(restart) instruction or the vector location respectively.
6. Now the processor save the address of the next instruction (program counter value) on to stack and switch to the related RST location or vector location.
7. Service routine written on the location is completed which have RET as its last instruction which returns the program control to the main program by retrieving the return address from the stack.

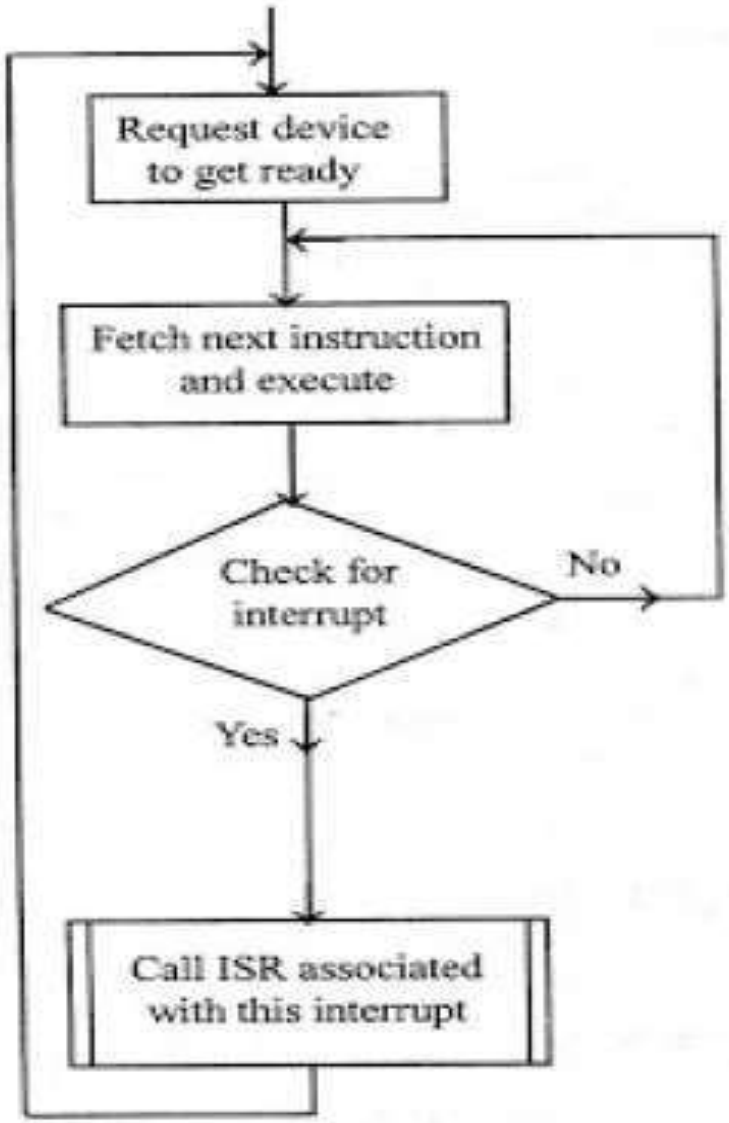


Fig (a) : Main program execution sequence

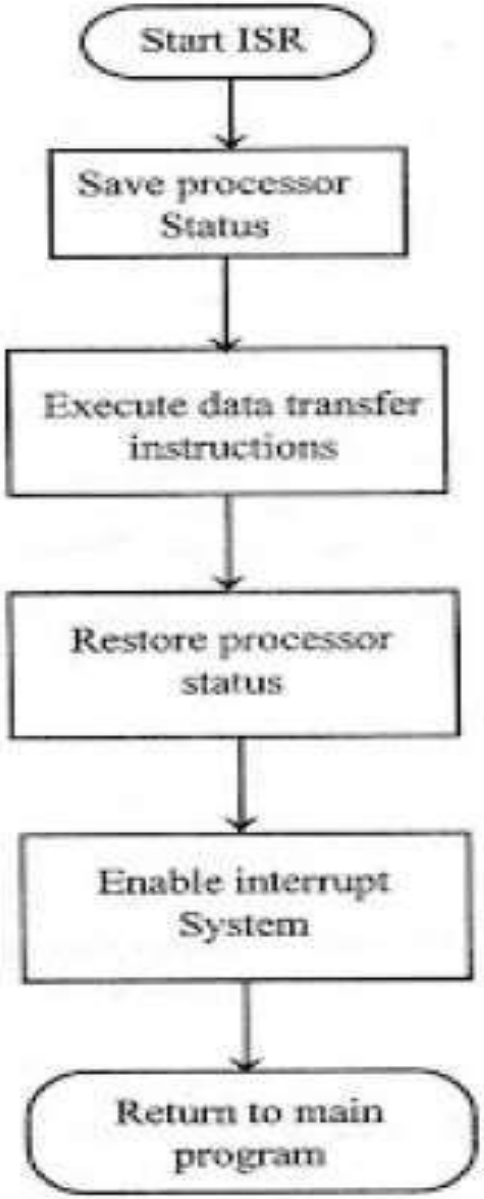


Fig (b) : ISR execution sequence

Types of Interrupt

- **Software Interrupt [RST-restart]**

- **Hardware Interrupt [TRAP, RST7.5, RST6.5, RST5.5,INTR]**

1. Software Interrupt:

- ✓ It is a instruction based Interrupt which is completely control by software.
- ✓ That means programmer can use this instruction to execute interrupt in main program.
- ✓ There are eight software interrupt available in μP that are **RST0 to RST7.** The vector address for these interrupts can be calculate as

  Interrupt number * 8 = vector address

  For RST 5 5*8 = 40(in decimal) =28H (in Hexa)

  Vector address for interrupt RST5 is 0028H. This vector address is stored in Program Counter(PC).
- ✓ These instruction allow transfer of program control from the main program to predefined service routine is also referred to as ISR(Interrupt Service Routine).

Software Interrupt...cont..

See the example with their hex code and vector address.

Instruction	Corresponding HEX code	Vector addresses
RST 0	C7	0000H
RST 1	CF	0008H
RST 2	D7	0010H
RST 3	DF	0018H
RST 4	E7	0020H
RST 5	EF	0028H
RST 6	F7	0030H
RST 7	FF	0038H

RST n Interrupt vectoraddress

2. Hardware Interrupt:

- ✓ This interrupt is caused by sending a signal on one of the interrupt pins of the microprocessor.
- ✓ An external device initiates the hardware interrupts and placing an appropriate signal at the interrupt pin of the processor.
- ✓ If the interrupt is accepted then the process or executes an interruptservice routine (ISR).
- ✓ Hardware interrupt is Asynchronous(it can occur at any time).
- ✓ The 8085 has five hardware interrupts

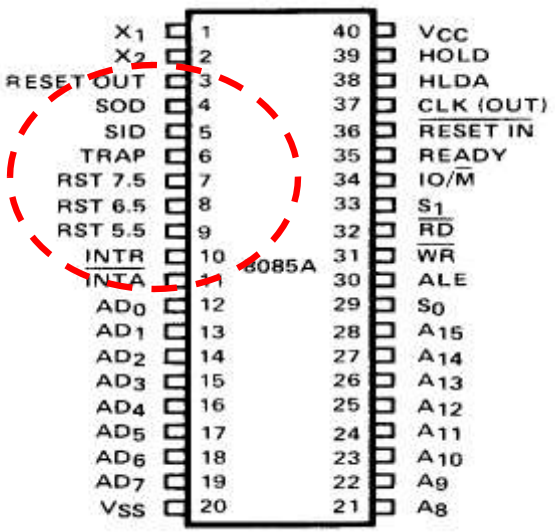
  (1) TRAP

  (2) RST7.5

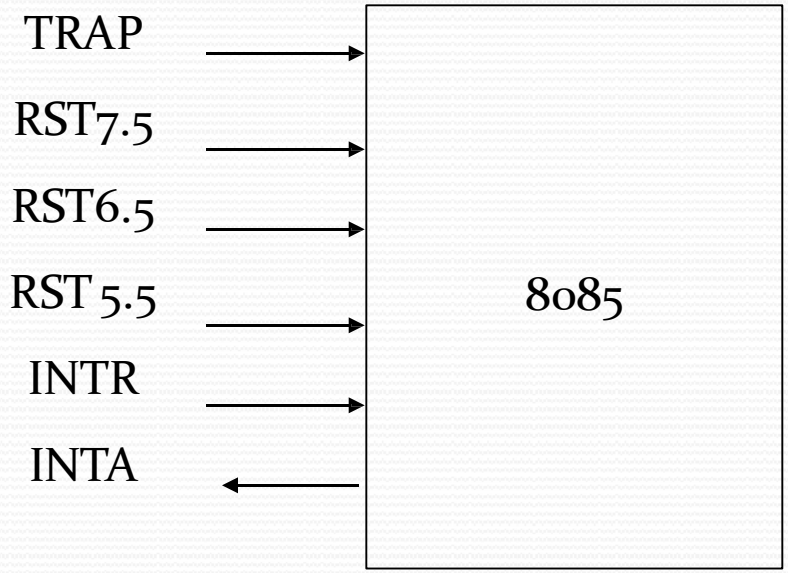
  (3) RST6.5

  (4) RST5.5

  (5) INTR(address is suppliedexternally)



8085 Interrupts



Hardware Interrupt.. cont..

✓ The hardware interrupts are classified Two types–

(1) Maskable Interrupts (Can be delayed or Rejected) :

- An interrupt which can be disabled by software that means we can disable the interrupt by sending appropriate instruction, is called a maskable interrupt.
- RST 7.5, RST 6, RST 5.5 , INT R are the example of Maskable Interrupt.

(2) Non-Maskable Interrupts (Can not be delayed or Rejected):

- Cannot disable the interrupt by sending any instruction is called Non Maskable Interrupt.
- TRAP interrupt is the non-maskable interrupt for 8085. It means that if an interrupt comes via TRAP, 8085 will have to recognize the interrupt we cannot mask it.

Hardware Interrupt... cont..

Interrupts can also beclassified into:

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

- In vectored interrupts, the processor automatically branches to the specific address in response to an interrupt.
- In vectored interrupts, the manufacturer fixes the address of the ISR to which the program control is to be transferred.
- The TRAP, RST 7.5, RST 6.5 and RST 5.5 are vectored interrupts.

(1) Non-Vectored (the address of the service routine needs to be supplied externally by the device):

- In non-vectored interrupts the interrupted device should give the address of the interrupt service routine (ISR).
- The INTR is a non-vectored interrupt. Hence when a device interrupts through INTR, it has to supply the address of ISR after receiving interrupt acknowledge signal.

Interrupts

What happens when MP is interrupted?

- 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.

Interrupt.. cont..

What happenswhen MP is responded to interrupt?

- 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
  - The device will have to supply the vector to the Microprocessor.
- 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.

Interrupt.. cont..

- When a device interrupts, it actually wants the MP to give a service which is equivalent to asking the MP to call a subroutine. This subroutine is called ISR (Interrupt Service Routine).
- This interrupts can be enable and disable by using EI (enable interrupt) & DI (disable interrupt) instructions.
- The 'EI' instruction is a one byte instruction and is used to Enable the non-maskable interrupts.
- The 'DI' instruction is a one byte instruction and is used to Disable the non-maskable interrupts.

Enable Interrupt(EI)

- ✓ The interrupt process is enable by using EI instruction in the main program.
- ✓ It is 1-byte instruction.
- ✓ It enables the interrupt process.
- ✓ Enabling will save the current status and jumps to an interrupt service routine (ISR). After completion it will return back to the main program again.

Disable Interrupt(DI)

- ✓ This DI instruction is used to disable theinterrupt.
- ✓ It is 1-byte instruction.
- ✓ This instruction reset the interrupt enable and disables the interrupt.
  - ➢ Both EI & DI are used to enable and disable the interrupts. If the interrupt is masked (disabled), they will not be recognized by microprocessor.
  - ➢ To enable It again they must be unmasked (enabled) by using EI.

8085 Interrupt

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

TRAP

- This interrupt is a Non-Maskable interrupt. It is unaffected by any mask or interrupt enable.
- TRAP is the highest priority and vectored interrupt (as vector address is fixed i.e. memory location where to transfercontrol).
- TRAP interrupt is edge and level triggered. This means that the TRAP must go high and remain high until it isacknowledged.
- In sudden power failure, it executes a ISR and send the data from main memory to back up memory.
- The signal, which over rides the TRAP, is HOLD signal. (i.e., If the process or receives HOLD and TRAP at the same time then HOLD is recognized first and then TRAP is recognized).However, TRAP has lower priority than the HLD signal used for DMA.
- There are two ways to clear TRAP interrupt.
    1. By resetting microprocessor (External signal)
    2. By giving a high TRAP ACKNOWLEDGE (Internal signal)

RST 7.5

- The RST7.5 interrupt is a Maskable interrupt.
- It has the second highest priority.
- It is edge sensitive .i.e. Input goes to high and no need to maintain high state until it recognized.
- Maskable interrupt. It is disabled by,
    1. DI instruction
    2. System or process or reset.
    3. After reorganization of interrupt.

RST 6.5 and 5.5

- The RST 6.5 AND 5.5 interrupt is a Maskable interrupt.
- It RST 6.5 has the third and RST 5.5 has forth highest priority.
- It is level triggered. i.e. Input goes to high stay high state until it recognized.
- Enable by EI instruction.
- Maskable interrupt. It is disabled by,
    1. DI, SIM instruction
    2. System or process orreset.
    3. After reorganization of interrupt.

INTR

- The INTR interrupt is a Maskable interrupt. It is disabled by,
  1. DI, SIM instruction
  2. System or process orreset.
  3. After reorganization of interrupt.
- Enable by EI instruction. Has lowest Priority.
- Non-Vectored interrupt After receiving INTA(active low) Signal, It has to supply the address of ISR. It is a level sensitive interrupts .i.e. Input goes to high and it is necessary to maintain high state until it recognized.
- The following sequence of events occurs when INTR signal goes high.
  1. The 8085 checks the status of INTR signal during execution of each instruction.
  2. If INTR signal is high , then 8085 complete its current instruction and sends active low interrupt acknowledge signal, if the interrupt is enabled
  3. In response to the acknowledge signal, external logic places an instruction OPCODE on the data bus. In the case of multi byte instruction, additional interrupt acknowledge machine cycles are generated by the 8085 to transfer the additional bytes in to the microprocessor.
  4. On receiving the instruction, the 8085 save the address of next instruction on stack and execute received instruction.

Interrupt Vectors & the Vector Table

- 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 (0000H - 00FFH).

Vector Address = Interrupt number * 8

Interrupt Name	Calculation	Vector Address
INTR	--	--
TRAP (RST 4.5)	4.5x8=36(Decimal))	0024H(Hexa)
RST 5.5	5.5x8=44	002CH
RST 6.5	6.5x8=52	0034H
RST 7.5	7.5x8=60	003CH

The 8085 Interrupts

Interrupt Name	Maskable	Masking Method	Vectored	Priority	ISR address	Triggering Method
TRAP	No	None	Yes	1st (Highest)	0024H	Level & Edge Sensitive
RST 7.5	Yes	DI / EI SIM	Yes	2nd	003CH	Positive Edge Sensitive
RST 6.5	Yes	DI / EI SIM	Yes	3rd	0034H	Level Sensitive
RST 5.5	Yes	DI / EI SIM	Yes	4th	002CH	Level Sensitive
INTR	Yes	DI / EI	No	5th (lowest)	No specific location	Level Sensitive

Level Triggered:- The signal at these pins must be maintained until the interrupt is acknowledged. External interrupt request flip-flops arerequired.

Edge Triggered: - Only a pulse is required to set the interrupt request → this request is remembered until the 8085A responds to the interrupt or until the request is reset by the **SIM** instruction or a /RESET IN signal. The interrupt request flip-flops for RST7.5 is internal to the microprocessor.

The 8085 Non-Vectored Interrupt Process

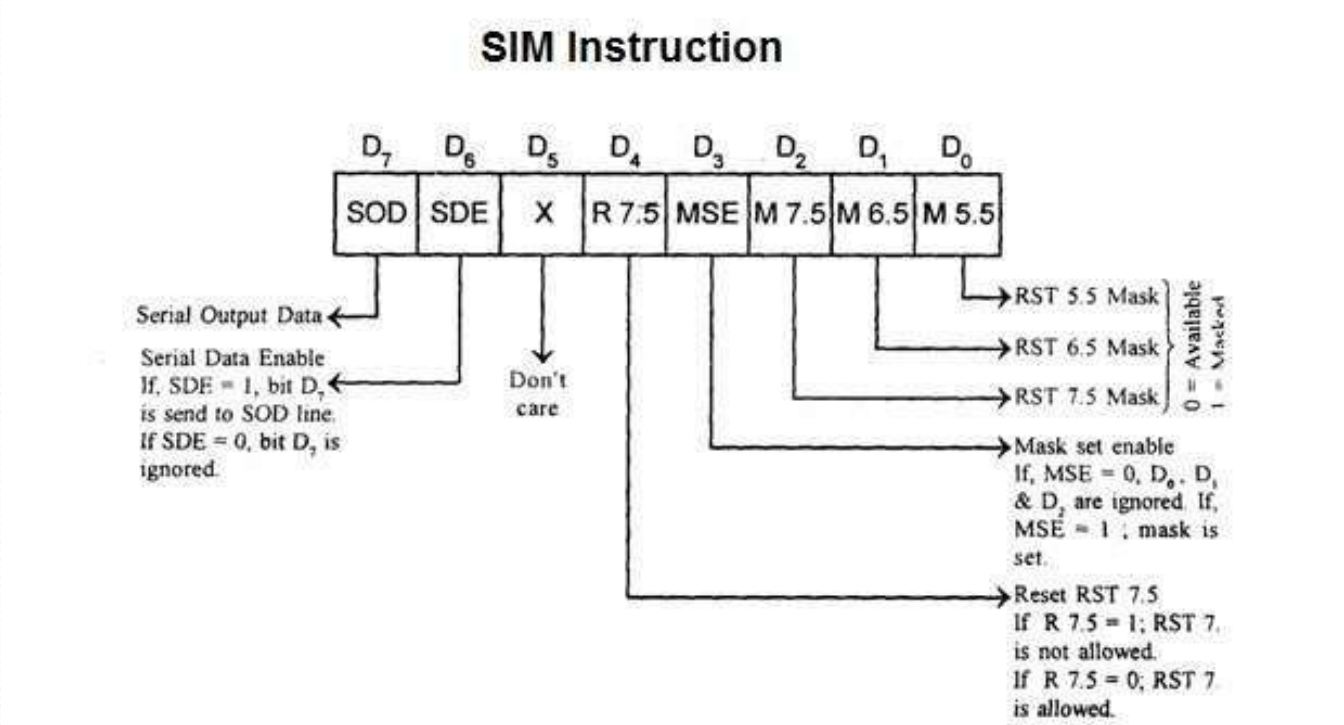
1. The interrupt process should be enabled using the EI instruction.
2. The 8085 checks for an interrupt during the execution of everyinstruction.
3. If INTR is high, MP completes current instruction, disables the interrupt and sends INTA (Interrupt acknowledge) signal to the device thatinterrupted
4. INTA allows the I/O device to send a RST instruction through data bus.
5. Upon receiving the INTA signal, MP saves the memory location of the next instruction on the stack and the program is transferred to 'call' location (ISR Call) specified by the RST instruction.
6. Microprocessor Performs the ISR.
7. ISR must include the 'EI' instruction to enable the further interrupt within the program.
8. RET instruction at the end of the ISR allows the MP to retrieve the return address from the stack and the program is transferred back to where the program was interrupted.

The 8085 Maskable/Vectored Interrupt Process

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, and if the interrupt is enabled using the interrupt mask, the microprocessor will complete the executing instruction, and reset the interrupt flip flop.
4. The microprocessor then executes a call instruction that sends the execution to the appropriate location in the interrupt vector table.
5. When the microprocessor executes the call instruction, it saves the address of the next instruction on the stack.
6. The microprocessor jumps to the specific service routine.
7. The service routine must include the instruction EI to re-enable the interrupt process.
8. At the end of the service routine, the RET instruction returns the execution to where the program was interrupted.

SIM for interrupt

- ✓ 8085 provide the additional masking facility for RST 7.5, RST 6.5 and RST 5.5 using SIM instruction.
- ✓ The status of these interrupts can be read by executing RIM instruction.
- ✓ The masking or unmasking of RST 7.5, RST 6.5 and RST 5.5 interrupts can be performed by moving an 8-bit data to accumulator and thenexecuting SIM instruction.
- ✓ The format of the 8 bit data isshown below

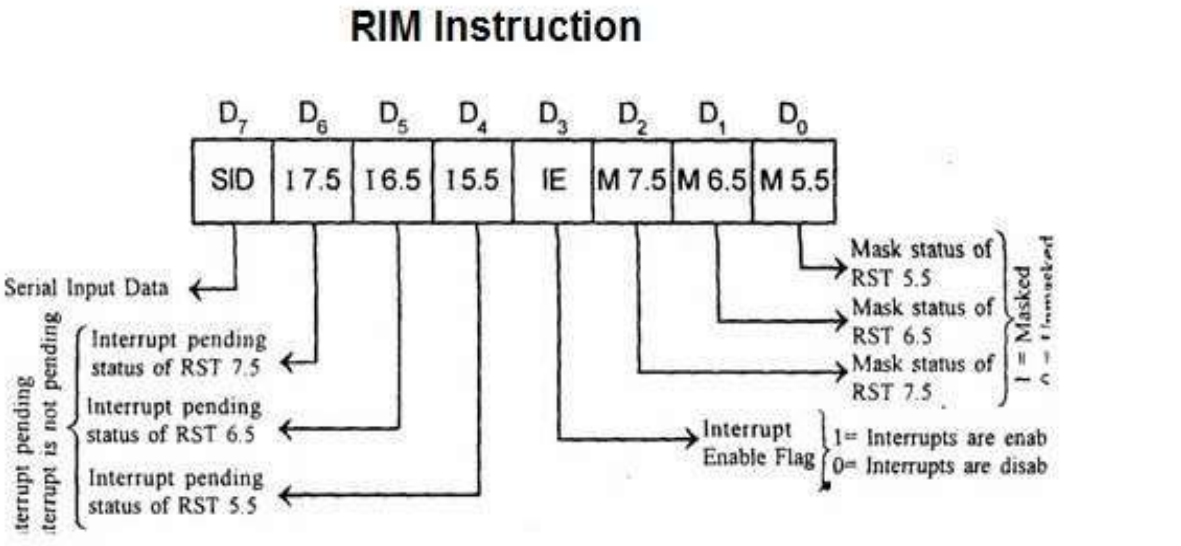


RIM for interrupt

- ✓ The status of pending interrupts can be read from accumulator after executing RIM instruction.
- ✓ Actually RIM does the following three tasks:

  1 Read the interrupt mask (bit 2, 1, 0). 2 Identify pending interrupts (bit 6, 5, 4).

  3 Receive serial input data bit (bit7).
- ✓ When RIM instruction is executed an 8-bit data is loaded in accumulator, which can be interpreted as shown in fig.



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.
    - They are automatically vectored according to the following table:

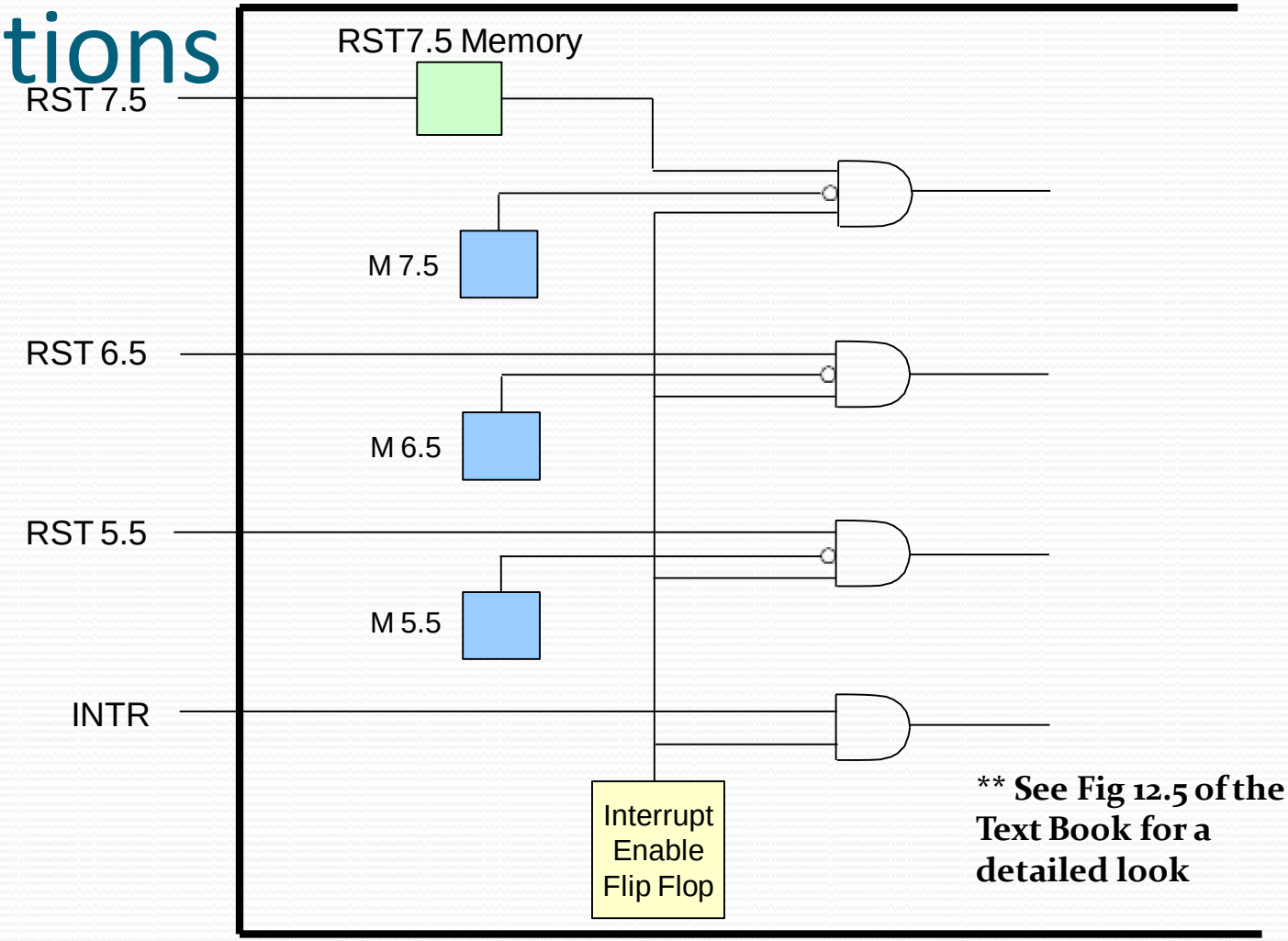
Interrupt	Vector
RST 5.5	002CH
RST 6.5	0034H
RST 7.5	003CH

      - 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).

Masking RST 5.5, RST 6.5 and RST 7.5

- These three interrupts are masked at two levels:
  - Through the Interrupt Enable flip flop and the EI/DI instructions.
    - The Interrupt Enable flip flop controls the whole maskable interrupt process.
  - Through individual mask flip flops that control the availability of the individual interrupts.
    - These flip flops control the interrupts individually.

Maskable Interrupts and vector locations



The 8085 Maskable/Vectored Interrupt Process

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, and if the interrupt is enabled using the interrupt mask, the microprocessor will complete the executing instruction, and reset the interrupt flip flop.
4. The microprocessor then executes a call instruction that sends the execution to the appropriate location in the interrupt vector table.

The 8085 Maskable/Vectored Interrupt Process

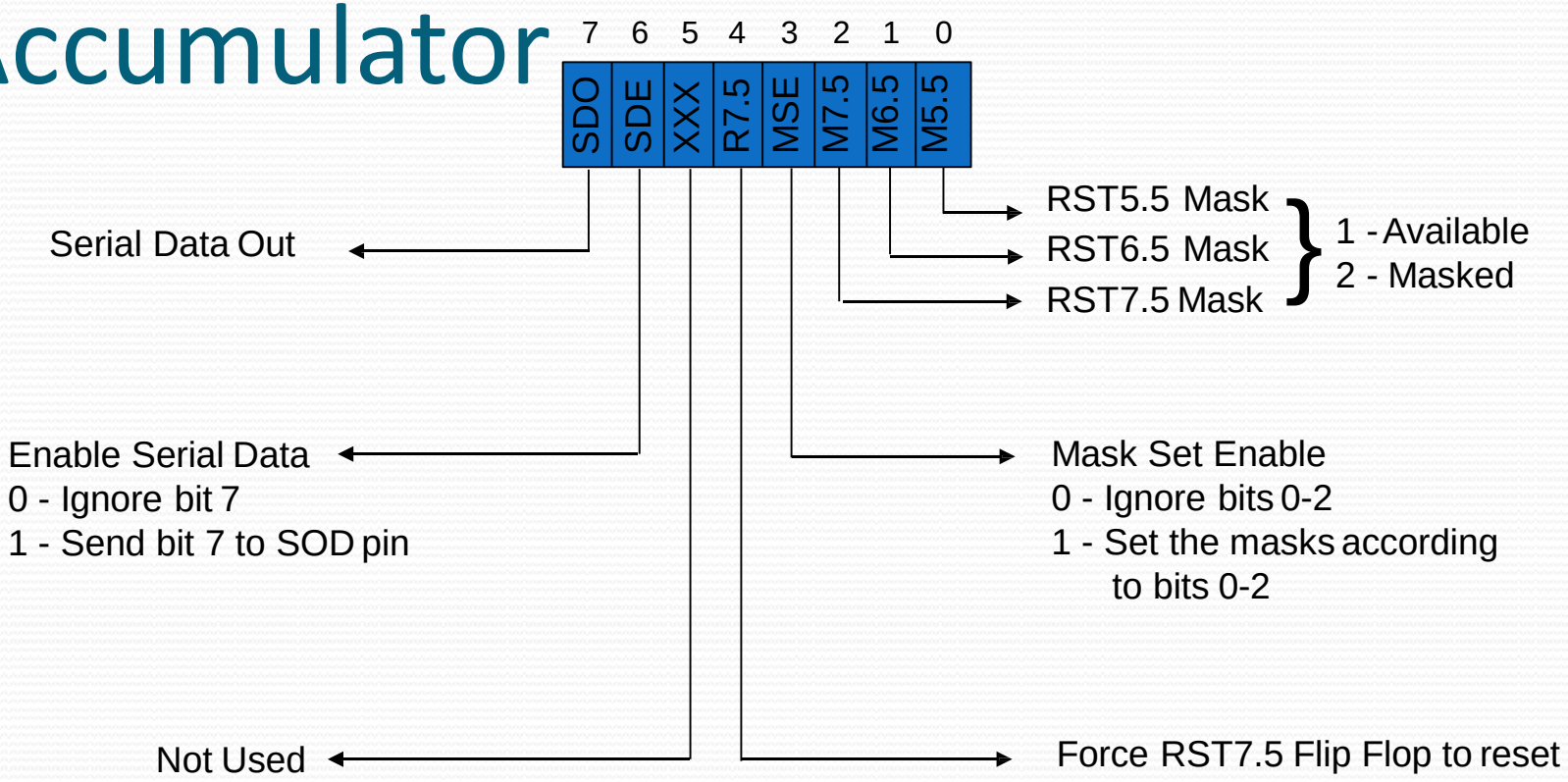
5. When the microprocessor executes the call instruction, it saves the address of the next instruction on the stack.
6. The microprocessor jumps to the specific service routine.
7. The service routine must include the instruction EI to re-enable the interrupt process.
8. At the end of the service routine, 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.

How SIM Interprets the Accumulator



SIM and the Interrupt Mask

- Bit 0 is the mask for RST 5.5, bit 1 is the mask for RST 6.5 and bit 2 is the mask for RST 7.5.
  - If the mask bit is 0, the interrupt is available.
  - If the mask bit is 1, the interrupt is masked.

- Bit 3 (Mask Set Enable - MSE) is an enable for setting the mask.
  - If it is set to 0 the mask is ignored and the old settings remain.
  - If it is set to 1, the new setting are applied.
  - The SIM instruction is used for multiple purposes and not only for setting interrupt masks.
    - It is also used to control functionality such as Serial Data Transmission.
    - Therefore, bit 3 is necessary to tell the microprocessor whether or not the interrupt masks should be modified

SIM and the Interrupt Mask

- 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.
- Bit 5 is not used by the SIM instruction

Using the SIM Instruction to Modify the Interrupt Masks

- Example: Set the interrupt masks so that RST5.5 is enabled, RST6.5 is masked, and RST7.5 is enabled.
  - First, determine the contents of the accumulator.

- Enable RST5.5 — 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

SDO	SDE	XXX	R7.5	MSE	M7.5	M6.5	M5.5
0	0	0	0	1	0	1	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
```

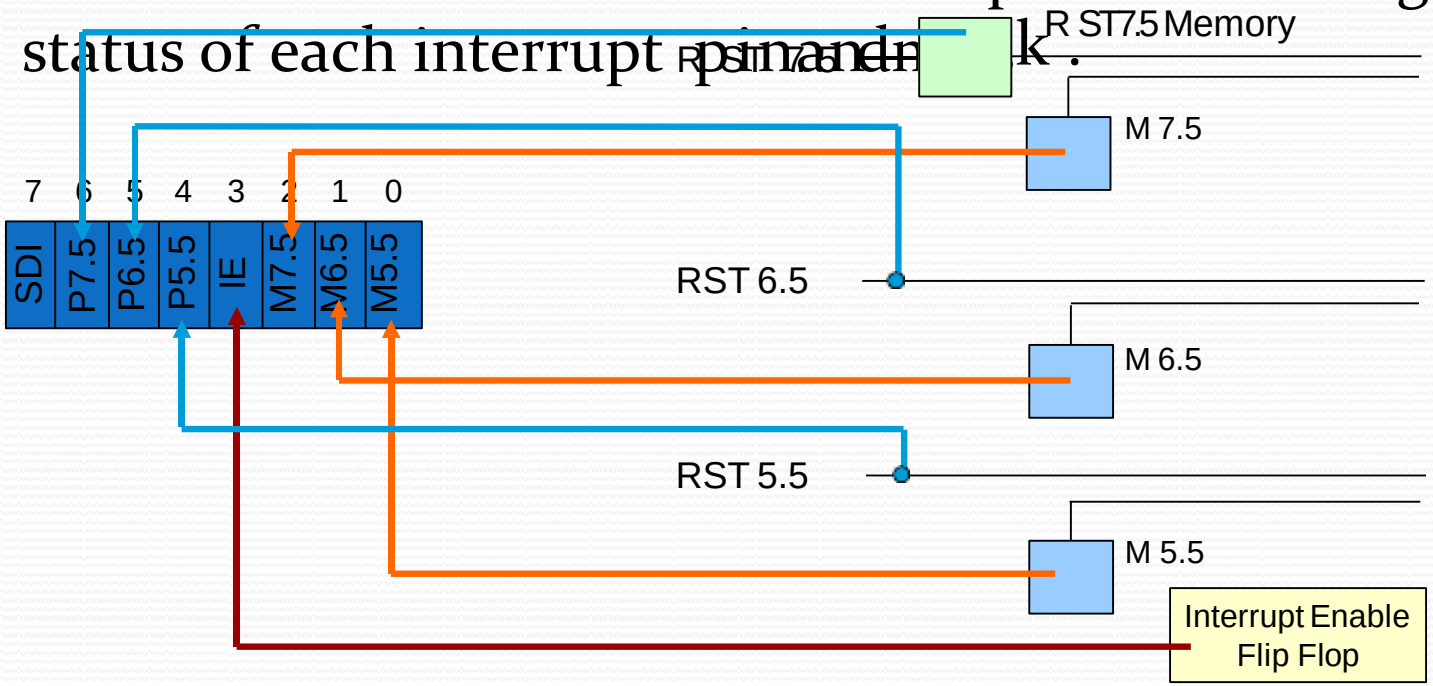
Triggering Levels

- RST 7.5 is positive edge sensitive.
  - When a positive edge appears on the RST7.5 line, a 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.

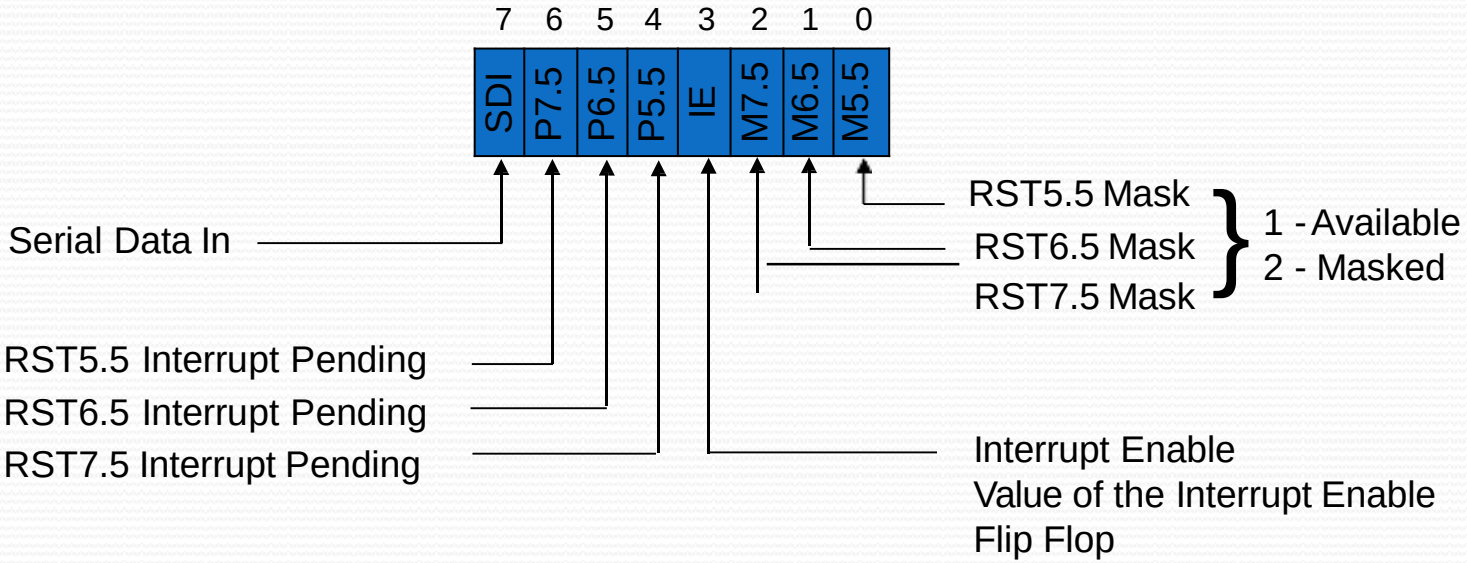
- RST 6.5 and RST 5.5 are level sensitive.
  - The interrupting signal must remain present until the microprocessor checks for interrupts.

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 the mask.



How RIM sets the Accumulator's different bits



The RIM Instruction and the Masks

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

The RIM Instruction and the Masks

- Bits 4-6 show whether or not there are pending interrupts on RST 7.5, RST 6.5, and RST5.5
  - Bits 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.

Pending Interrupts

- Since the 8085 has five interrupt lines, interrupts may occur during an ISR and remain pending.
  - Using the RIM instruction, it is possible to can read the status of the interrupt lines and find if there are any pending interrupts.
  - See the example of the class