

**Microprocessors and Microcontrollers**  
**Prof. Santanu Chattopadhyay**  
**Department of E and EC Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 18**  
**8085 Microprocessors (Contd.)**

The other instruction that we have in this RST category is that trap instruction which is a not Maskable interrupt, so that is that is a non Maskable interrupt. So, you cannot mask it off.

(Refer Slide Time: 00:28)

**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:

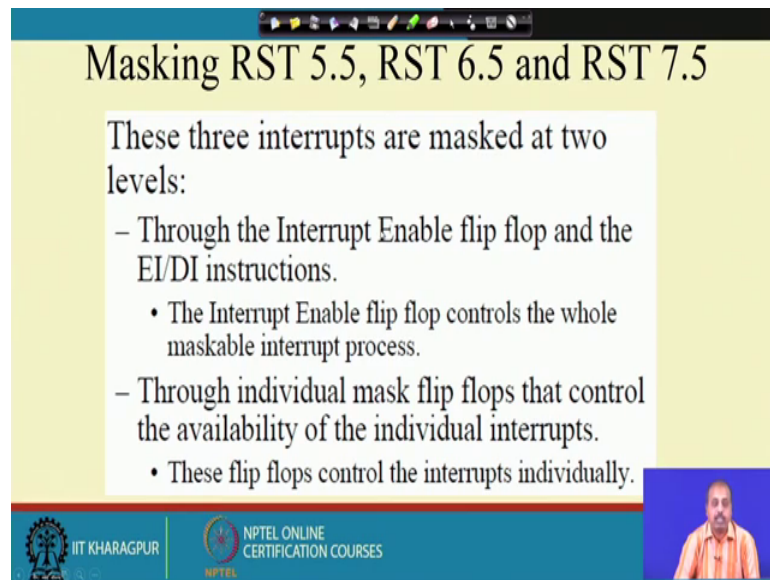
Interrupt	Vector
RST 5.5	002CH
RST 6.5	0034H
RST 7.5	003CH

- 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).

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 148

But these instructions these interrupts RST 5.5, 6.5, 7.5, so these are hardware interrupt. So, hardware interrupts also they are Maskable so this we can mask off this interrupts we will see how they can be done and they also vectored interrupt because the addresses are fixed. So, they are automatically vectored as per this table is shown.

(Refer Slide Time: 00:48)



**Masking RST 5.5, RST 6.5 and RST 7.5**

These three interrupts are masked at two levels:

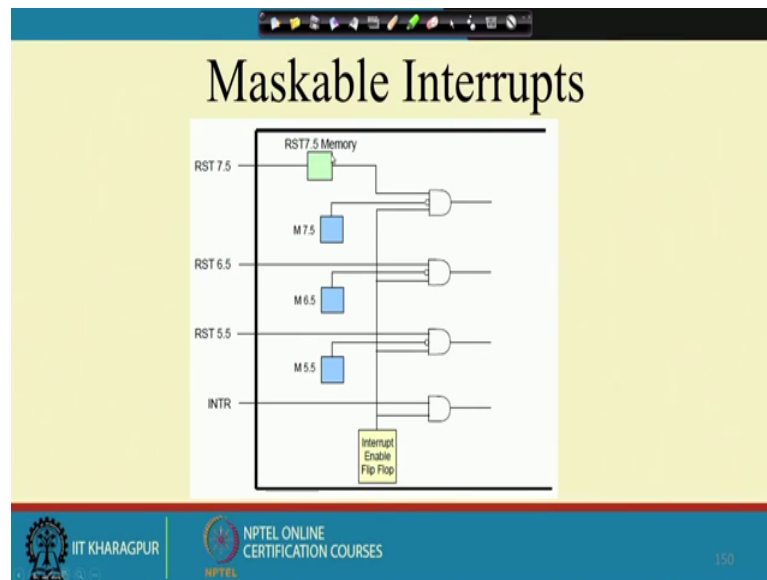
- Through the Interrupt Enable flip flop and the EI/DI instructions.
  - The Interrupt Enable flip flop controls the whole maskable interrupt process.
- Through individual mask flip flops that control the availability of the individual interrupts.
  - These flip flops control the interrupts individually.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, they are mask there this masking of this interrupt 5.5, 6.5, 7.5 it can be done at two levels one level is via the interrupt in double flip flop and the E I, D I instructions so that is the one possibility, so you can just set the E I, D I instruction you can use the E I, D I instruction to change the interrupt enable flip flop and all the Maskable interrupts will get marks so if you put the D I instruction.

So, this interrupt enable flip flop so it will control the whole Maskable interrupt process and also you can individually mask out the interrupt like; so there are individual controls individual mask flip flop are available and you can control individual interrupts that way. So, E I or D I instruction will mask off all the interrupt all the Maskable interrupt, but you may want that only say 6.5 be must so 5.5 and 7.5 should remain. So, how to do this thing? For that purpose so we have to follow some other technique we have to set reset the interrupt corresponding to individual lines ok.

(Refer Slide Time: 01:59)



So, this is the diagram that shows the situation. So, we have got these flip flops mask 7.5 or M 7.5, M 6.5, M 5.5 if these bits are made 0 so you see that this is inverted here, so this bit is so if say 6.5 this bit is 0 so will get a 1 at the end input of this and get and then this interrupt enable flip flop is there suppose this bit is 1.

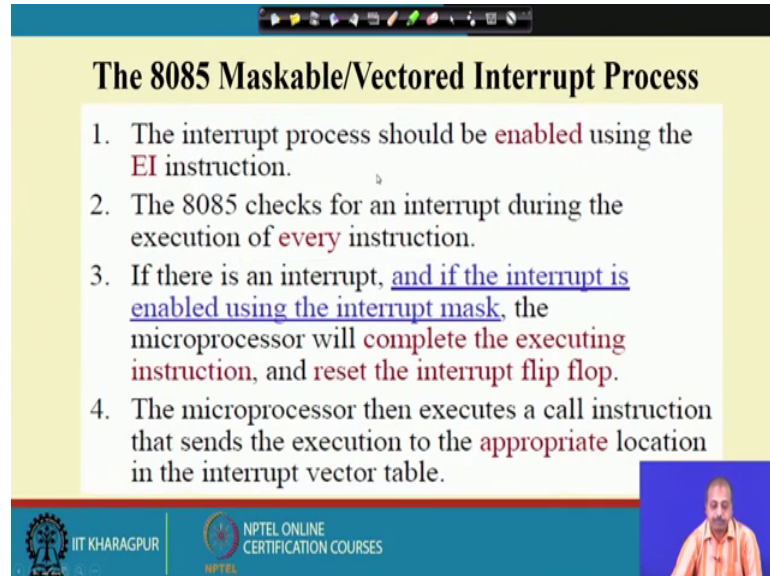
So, this one now if now if that 6.5 interrupt comes then that interrupt is allowed to reach the processor, so this is actually going to the processor, now if this bit is set to 1 if M 6.5 flip flop is set to 1 so that is the 6.5 will be masked of because this and get we will get an input of 0, so whatever will be the value of RST 6.5 so it will not reach the output. So, we have got this mask flip flops M 7.5, M 6.5 and M 5.5 for masking out this interrupt individually, but for INTR we do not have this individual masking so INTR can be mask of by this interrupt enable flip flop only so through the E I, D I instructions only so they cannot be ma it cannot be masked of otherwise.

This RST 7.5 Flip Flop, so there is another RST 7.5 interrupt it has got another flip flop that will see later so this is basically keeping the memory like; see this 6.5, 5.5 etcetera, so these interrupts so you see that if these interrupt has to be sensed then they the interrupt line must be active for that 17.5 T cycle T status, for as for 7.5 so since this a flip flop. So, if this interrupt occurs then the value will get last here.

So, when the processor checks at the end that 17.5 clock cycle then also it will find that flip flop value to be equal to 1, so that interrupt will be sensed.

So, gives as some special facility with the RST 7.5 flip flop, so that is it can be of much shorter duration than this 6.5, 5.5 and INTR. So, we will come to that later.

(Refer Slide Time: 04:22)



**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.

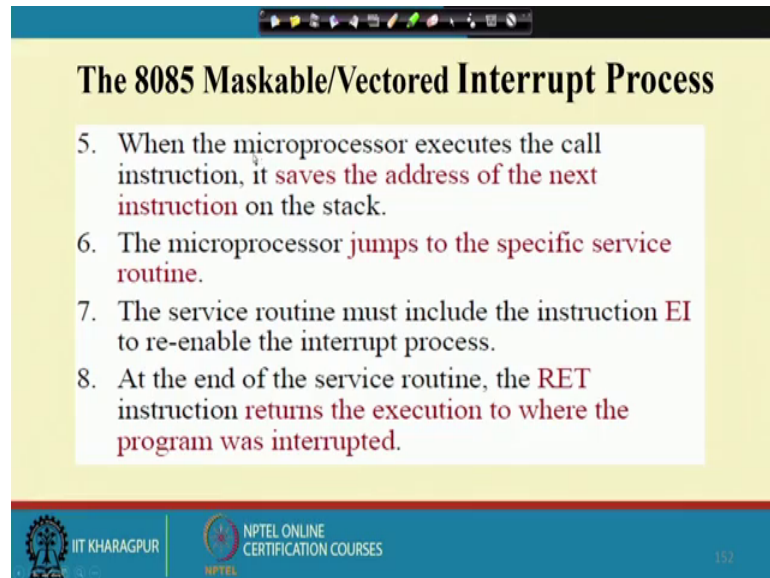
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, the interrupt process first then as per as this entire process is concerned, first of all the process has to be enabled through the E I instructions, if E I is not executed then that that interrupt enable flag maybe flip flop maybe 0 as a result none of the interruption be reaching the processor and 8085 just like that the non vectored inter non vectored interrupt that is like that INTR so here also 885 will check the interrupt during the execution of every instruction and if there is an interrupt and the interrupt is enabled by the interrupt mask, so this is the additional thing that we have here it will check that; M 7.5, M 6.5, M 5.5 flip flop and then only it will say that the interrupt is received by the processor.

So, it will the processor if that flip flop is 0 if the flip flop value is 0, then the processor will sense that interrupt and it will complete executing the current instruction and reset the interrupt flip flop, the flip flop will be reset and the processor will execute a call instruction and this call instruction is a hard call so now for an like that INTR where the device at to provide the vector address through some special arrangement of buffers and all that so here we do not need all those things, here the processor it is fixed the addresses are fixed so processor will directly go to the corresponding address and that address is also obtained in the similar fashion like; 5.5 if it is RST 5.5 so 5.5 multiplied

by 8 so it will be 42 44 so it will go to the location 44 and start executive from there so that way it continues. So, when the processor execute the call instruction it saves the address of the next instruction on the stack the processor jumps to the specific service routine and the service routine must include the instruction E I to re enable the interrupt process.

(Refer Slide Time: 06:14)

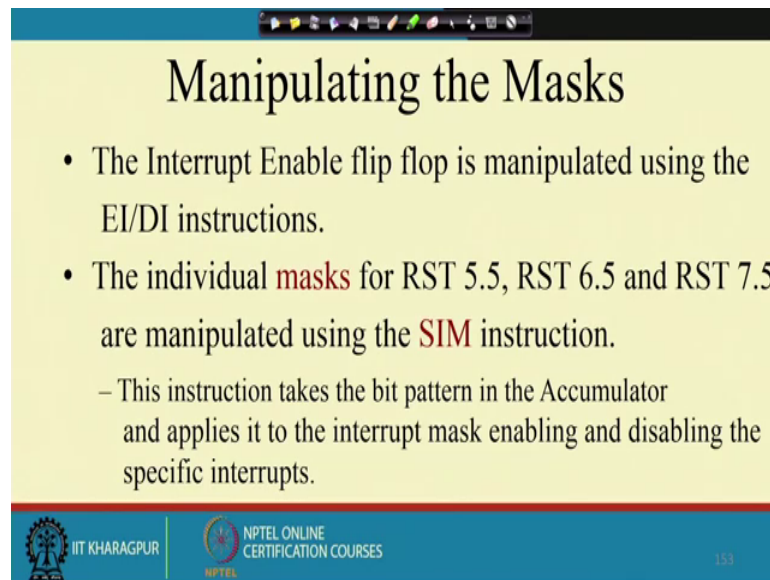


The slide is titled "The 8085 Maskable/Vectored Interrupt Process" and contains a list of four numbered points. The text is presented in a white box on a yellow background. At the bottom of the slide, there are logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, along with the number 152.

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**.

So, this is these are same as that INTR inter process once the only the sensing is sensing part is different and then there should be the rate instruction at the end of the service routine so that the execution will return from the ISR.

(Refer Slide Time: 06:44)



The slide is titled "Manipulating the Masks" and contains the following text:

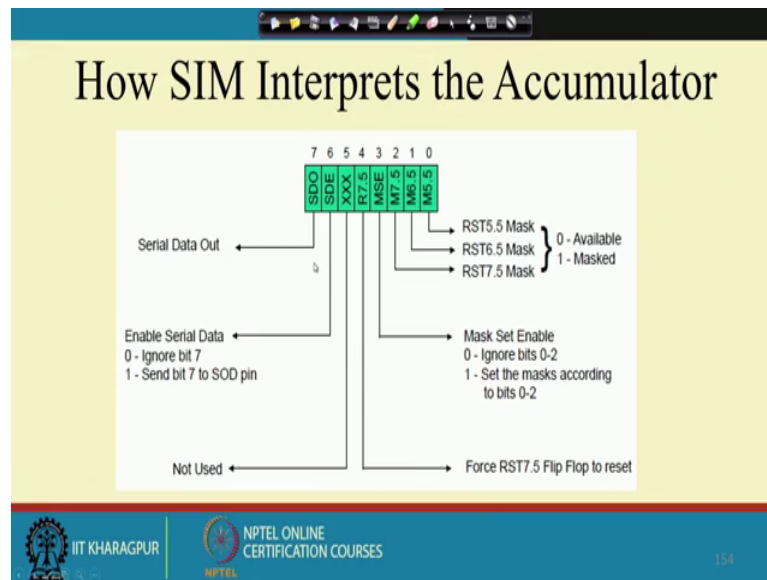
- The Interrupt Enable flip flop is manipulated using the EI/DI instructions.
- The individual **masks** for RST 5.5, RST 6.5 and RST 7.5 are manipulated using the **SIM** instruction.
  - This instruction takes the bit pattern in the Accumulator and applies it to the interrupt mask enabling and disabling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

The slide footer includes the IIT Kharagpur logo, the text "IIT KHARAGPUR", the NPTEL logo, and the text "NPTEL ONLINE CERTIFICATION COURSES". The number "153" is also present in the bottom right corner.

Now, how since the mask are there so you must know a technique by which this mask can be manipulated. So, one way to manipulate that interrupt enable flip flop is through the E I, D I instruction, that interrupt enable flip flop that particular flip flop will be modified by E I or D I but this individual mask for 5.5, 6.5 and 7.5 they are manipulated using the SIM instruction.

So, there is a special instruction SIM which stands for; set, interrupt, mask using this SIM instruction we can manipulate the setting of this mask flip flop. So, this instruction it will take the bit pattern in the accumulator and apply it to the interrupt mask enabling and disabling to the interrupt mask that will result in enabling and disabling of specific interrupt, so it will so for using that instruction so the first of all we have to say the accumulated to a particular bit pattern and after that when the SIM is executed so individual bits of the accumulated they will have some special meaning as we shall see then this accordingly those flip flop will be set to different values and this marks pattern will be set this masking of the flip flop will take place. So, this is the line this is the diagram that shows how this in individual bits of the accumulator will be interpreted by the SIM instruction.

(Refer Slide Time: 08:05)



Now there is one thing that this SIM instruction also helps in another operation which is known as the serial data transmission so we will see it after some sometime. So, the first two bits SDO and SDE so there are for serial data input output type of operation, so for our discussion run now so this two bits are of no use so if this SDE bit is set to 0, so it will ignore this SDO line so for our discussion for the time being this SDE line serial data enabled so this line will be 0 so this bit will be 0.

So, that this SDO bit setting does not have any meaning, then this bit number 5 is not used by the by in this process this is it can be any arbitrary value do not care value, then remaining 5 bits so they are used to control the flip flops so this M 5.5 so this is the reset mask reset 5.5 mask, this is RST 6.5 mask, this is RST 7.5 mask, so if this mask bit is 0; that means, we want to set the interrupt to be available and if this mask bit is 1; then we want that the interrupt should not be available, the interrupt should not reach the processor and whether we want to set this mask bit or not so that is set done by this MSE bits so mask set enable so if this bit is 0; so it will ignore bit number 0 to 2 and if this bit is 1; so it will set the mask according to the bits 0 to 2.

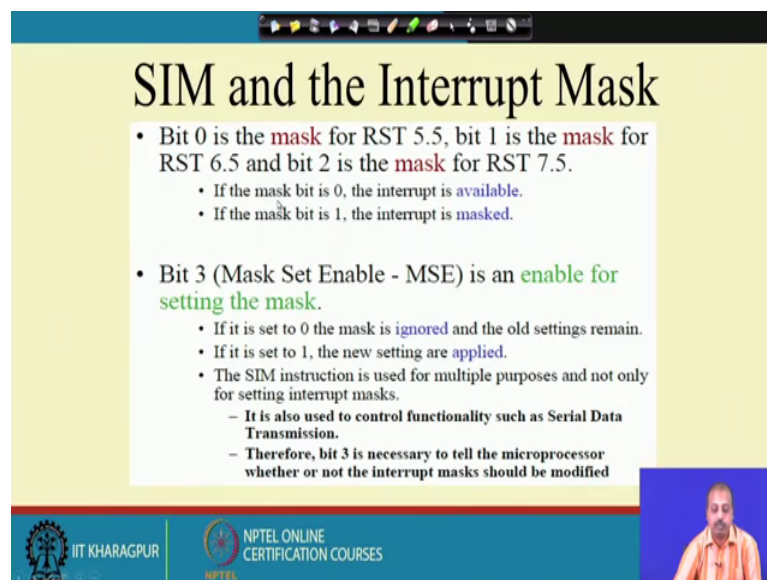
Now, you see that there is a so since the same instructions it is used for both serial data transmission and this interrupt masking. So, that is why we have got two grads actually. So, SDE is one grads so that is useful when you are doing only serial data transmission, so then that bit will be set to 1 and if you are not if you are not doing that then this bit



will be set to 0, similarly if you are using this SIM instruction for serial data transmission and not for mask setting then this MSE bit should be set to 0, so that if this MSE bit is 0 then this mask setting does not have any meaning, so then this then we can use it for serial data transmission by setting this SDE is the line to 1 and if you really want that this mask bit will be set mask bit will be set then we should make this MSE bit 1 and then this pattern will be set to the mask flip flops of the interrupt.

And this R 7.5 so this is the this setting of this RST 7.5 flip flop, so in this diagram that we have seen so this RST 7.5 so this flip flop can be set by getting that RST by setting this R 7.5 bit. So, the accumulator has to be set a value as per our requirement and then this are then the SIM instruction has to be executed.

(Refer Slide Time: 11:32)



**SIM and the Interrupt Mask**

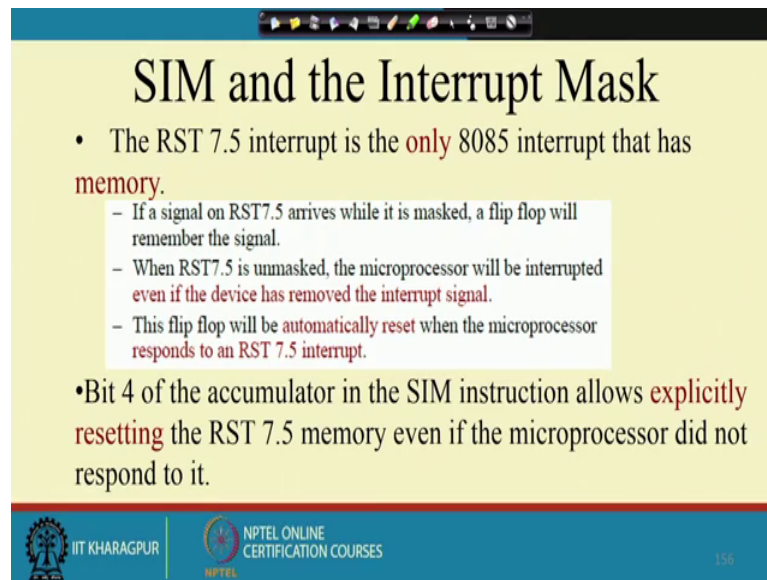
- Bit 0 is the **mask** for RST 5.5, bit 1 is the **mask** for RST 6.5 and bit 2 is the **mask** for RST 7.5.
  - If the mask bit is 0, the interrupt is **available**.
  - If the mask bit is 1, the interrupt is **masked**.
- Bit 3 (Mask Set Enable - MSE) is an **enable for setting the mask**.
  - If 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

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this is the details that bit 0 is the mask for RST 5.5, bit 1 is the mask for 6.5, bit 2 is the mask for 7.5, the mask bit is 0 then interrupt is available, the mask bit is 1 then interrupt is not available so it is masked, bit number 3 it is used for enabling it is enable for setting the mask if the bit is 0 then the mask setting will be ignored and the previous settings will continue and if the bit is one then the new settings will be applied and this SIM instruction it is multiple purposes as I have said, so the serial data transmission is also used here so bit number 3 is necessary to tell the microprocessor whether to use the interrupt set the interrupt mask or not and similarly that SDE line so that is used to say whether we are going for serial data transmission or not.



(Refer Slide Time: 12:28)



**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.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 156

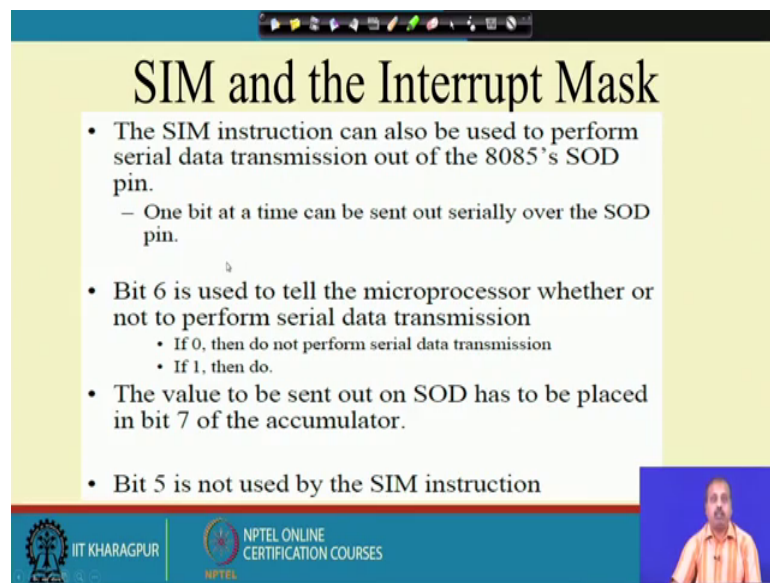
Now this RST 7.5 is the only interrupt that has memory, so as I was telling so that it has got a flip flop R 7.5 so that is the for latching that occurrence of that interrupt, so if a signal on RST 7.5 arrives where while it is masked a flip flop will remain remember the signals, so it might be that say RST 7.5 is currently masked off, but still the if the interrupt occurs that flip flop value will be set to 1 and when this RST 7.5 will be unmasked; then the processor will be interrupted even if the device is removed the interrupt signal, because that flag is set that flip flop is set so when this if after sometime if the interrupts are enable then this processor will get that interrupt.

So, this flip flop is reset when the processor respond to 7.5 interrupt, so this is basically the purpose like; if we are doing something and we were busy the processor was busy doing something at which this interrupt has occurred 7.5 so it will remain last in the system and later on the processor may like to respond to that interrupt, so this facility is provided and it is not provided with other flip flop other interrupt like; 6.5 and 5.5 only with 7.5 it is there. So, if you want to reset this bit if you if you want that the processor should not get the interrupt then you can reset this 7.5 memory by using this bit 4, so bit 4 of the accumulator the SIM instruction allows explicitly resetting the RST 7.5 memory even if microprocessor did not respond to it.

So, in this slideshow if you so this is reset 7.5, so if you if you if you keep this bit as 1 and after that executive SIM instruction then the RST 7.5 flip flop will be clear, so even

if I was when the processor was busy the inter the interrupt has occurred and the flip flop is set so instead of going into the interrupt service routine, so you may just like to reset that flip reset that flip flop so that we do not into respond to the interrupt. So, for that purpose so we can keep this RST 7.5 line 1 and execute the SIM instruction.

(Refer Slide Time: 14:59)



**SIM and the Interrupt Mask**

- 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

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, it can also be used for data transmission as I said that serial data SIDN, SODP inside there for 8085 for serial data transmission, now this bit 6 will be used to tell the microprocessor whether or not to perform serial data transmission if it is 0 then do not transfer serial data, if it is 1 then when the SIM is executed then this is SOD line has to be the value to be sent on the SOD has to be placed on the bit 7 of the accumulator. So, whatever will be the value at bit 7 so that will go to the serial data out line and bit 5 is not used with the SIM line.

(Refer Slide Time: 15:42)

**Using SIM Instruction to Modify 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 5.5	bit 0 = 0	SDO	0
- Disable 6.5	bit 1 = 1	SDE	0
- Enable 7.5	bit 2 = 0	XXX	0
- Allow setting the masks	bit 3 = 1	R7.5	0
- Don't reset the flip flop	bit 4 = 0	MSE	1
- Bit 5 is not used	bit 5 = 0	INT.5	0
- Don't use serial data	bit 6 = 0	M6.5	1
- Serial data is ignored	bit 7 = 0	M5.5	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
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

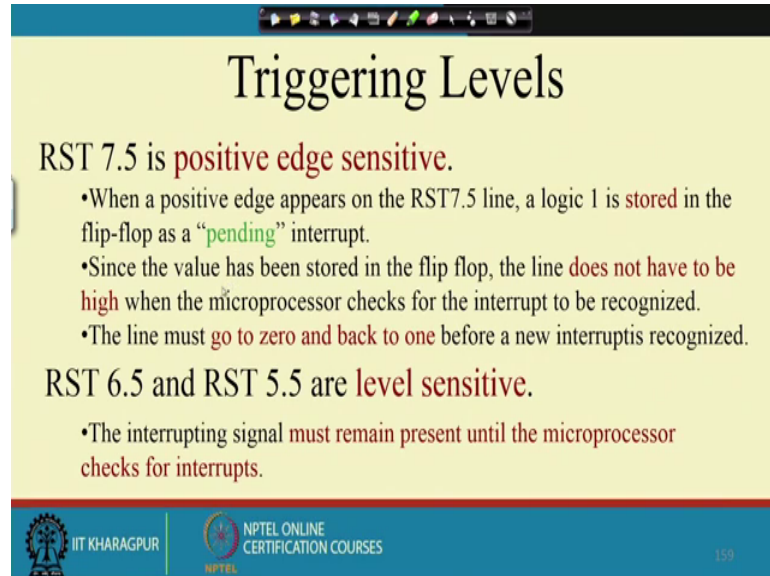
So, we take an example; suppose we want to set the interrupt marks of so that this 5.5 is enabled, 6.5 masked off and 7.5 is also enable. So, for that purpose we have to first identify the content of the accumulator that it should be there, so we want to enable 5.5 and 7.5 so we want to enable 5.5 so bit 0 should be equal to 0, we want to disable 6.5 so bit 1 should equal to 1, we want to enable RST 7.5 so bit 2 will be equal to 0.

Now we have to allow this mask setting whatever mask we are setting this 0, 1, 0 should be allowed so accordingly this MSE bit should be equal to 1 so this bit 3 is equal to 1, RST 7.5 is enabled and we do not want it the flip flop to be reset that is if the some interrupt has occurred so this flip flop that interrupt should be there so reset the flip flop, so these R 7.5 bit is set to 0, bit 5 that is do not care that is not used and we are not going to use for the serial data transmission so bit 6 and bit 7 there are made 0, so this is the bit pattern that we have to set in the accumulator so that is done here, so this pattern is 0A so this most significant bit is all 0 and the next bit is 1 0 1 0 so that is A.

So, first we enable interrupt so it will enable all including INTR then we move the pattern 0A to the accumulator so that is done by the MVI instruction MVIA comma 0A, so it will set the accumulator to the proper pattern that we want and then execute the SIM instruction, so when the SIM instruction is executed this particular bit pattern so this will be setting the this will be setting the appropriate flip flows and masks and the desired

functionality will be achieved; that is, 5.5 and 7.5 will be enabled and 6.5 will be masked off.

(Refer Slide Time: 17:47)



**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.

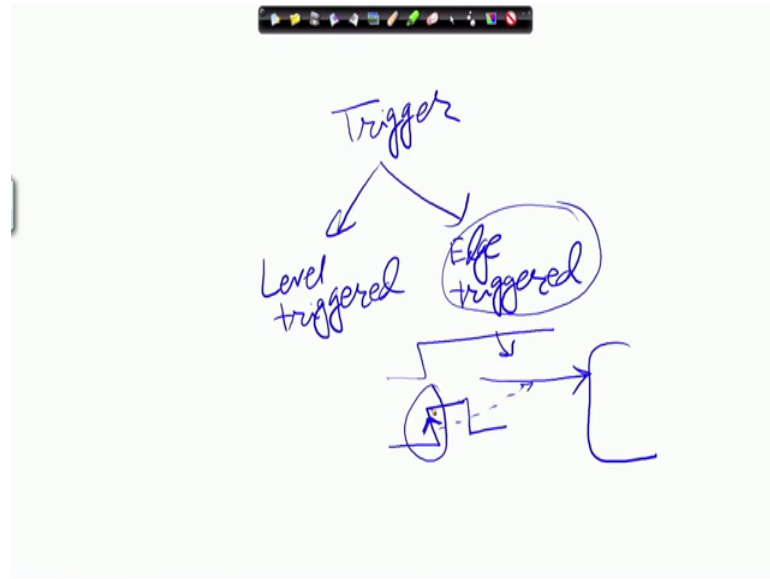
RST 6.5 and RST 5.5 are **level sensitive**.

- The interrupting signal **must remain present until the microprocessor checks for interrupts**.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | NPTEL | 159

Next important thing that we have is about the triggering level, now you see that if you to look into classification of interrupt then one classification that we have is that of triggering level, so trigger level of interrupts so they can be on that basis we can classified into two categories one is level triggered; one is level triggered and another is edge triggered, just like flip flops we have level triggering and edge triggered, so here also we have got level triggered and edge triggered.

(Refer Slide Time: 18:09)



So, level triggered means if this is the processor and this is the line the interrupting line that is there, so if it is a level triggered interrupt then for the line to be sensed the level of the interrupt should be high, so it is like this so the high value will be sensed by the processor. On the edge triggered means the edge will be sensed, so this rising edge will be sensed as the interrupting point.

So, you see that for this happen for this rising edge sense to be sensing to be are to be happy to happen so we should be able to latch the occurrence of this particular event, now so far whatever we have said that; 5.5, 6.5 they do not have any separate latch or memory, but 7.5 has got a latch associated with it so this 7.5 is basically an edge triggered flip flop edge triggered interrupt because that interrupt occurred then the value will be put into the that latch will be set accordingly even if that line goes low after sometimes after sometime it goes low so this rising edge so this will be remembered by the flip flop and this triggering will take place.

So, this when a positive edge appears on the RST 7.5 line a logic 1 is stored in the flip flop as a pending interrupt and now this value stored in the flip flop so does not the line is no more needed to be kept high. So, the processor can check the interrupt that is that is to be recognized so in the so it checks whether there are any interrupt has occurred and accordingly it can do that and the line must go to 0 and back to 1 before new interrupt has occurred.

So, this is another good thing like otherwise for level triggered interrupt what happens is? That if the processor as soon as enable line E I line E I bit is executed than if the value of that line is still 1 the interrupt line is still 1 that will be sense as another interrupt, but in the edge triggering so it is based on the edge so if another edge has not come that is in between the line has not turn down to 0 and then again it has risen to 1 if that has not happened that will not be taken as a new interrupt. So, this way edge triggering may be helpful then this RST 6.5 and 5.5 there are level sensitive and the interrupting signal must remain present until the processor checks for the interrupts.

So, this way this triggering levels so we have got various combinations like; INTR is also a level triggered interrupt, so we have got all these combinations available in the micro in the 8085 processor.

So, depending upon the type of device that you are going to connect, so can have this the level triggering and edge triggering accordingly we can choose the type of interrupt to which it should be connected.

(Refer Slide Time: 21:39)

**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.

The diagram illustrates the internal structure of the interrupt mask register. It shows an 8-bit register with bits labeled 7, 4, 3, 1, 0. Bit 7 is labeled 'SI' (Slave Interrupt), bit 4 is 'PI 7.5', bit 3 is 'PI 6.5', bit 1 is 'MI 7.5', and bit 0 is 'MI 6.5'. The register is connected to external signals: RST 7.5 (Slave Interrupt), RST 6.5 (Master Interrupt), and RST 5.5 (Slave Interrupt). The register also controls internal components: RST 7.5 Memory, M 7.5 (Mask 7.5), M 6.5 (Mask 6.5), and M 5.5 (Mask 5.5). An 'Interrupt Enable Flip Flop' is also shown, which is controlled by the RST 5.5 signal.

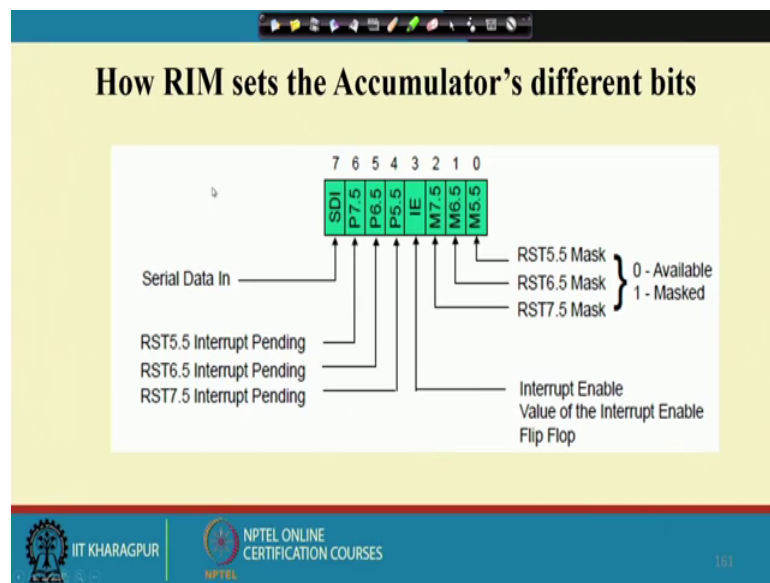
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 160

So, next part is sometimes we would like to see what is the current mask setting because maybe we are we do not want to disturb the current mask setting only change it a bit so you do not want to put a completely new setting, but just want to change the bit. So, for that purpose so there is another instruction called rim which read as read interrupt mask RIM read interrupt mask. So, what it will do? It will load there accumulator with an 8 bit

pattern showing the status of each interrupt pin and mask. So, what happens is that? RST 7.5 memory so that R 7.5 so that comes the bit number 6, then this 6.5 pins status so that will come p 6.5 this 5.5 pin will come here, then this interrupt enable flag so that will come as bit number 3 and this mask flip flop content M 7.5, 6.5 and 5.5 they will come to the bits 2 1 and 0, then this bit number 7.

So this is reserve for serial input data or serial data input, so this RIM instruction is also used for serial data input so the SIM was used for serial data output so RIM used for serial data input, so if we are not using that then that bit has to be dis discarded, but otherwise rest of the bits so they will give us the current interrupt mask setting in the processor and this will be use full like; if you want to change the mask settings slightly without disturbing others, so you can use this particular situation.

(Refer Slide Time: 23:23)

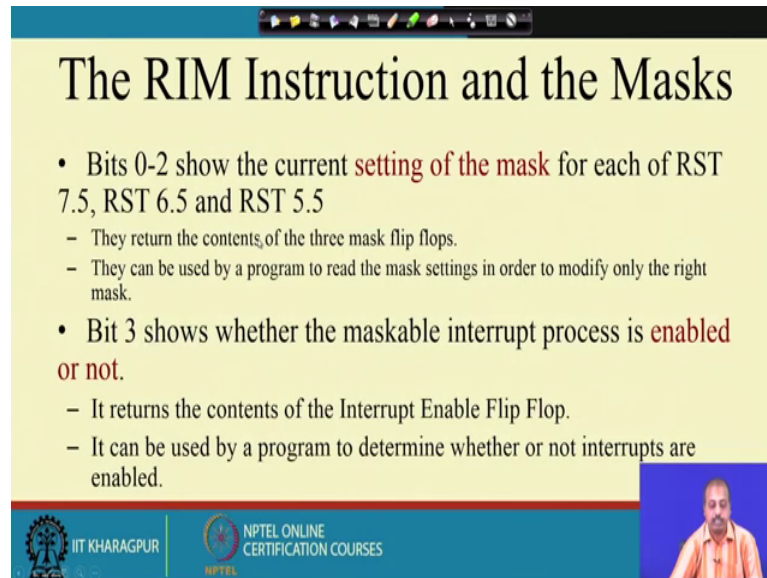


So, the thing that this RIM setting so this will be affecting the thing previous diagram so it is show like this. So, the serial data line so whatever is available in the a SID pin so that will be available in the SDI bit here, then if some RST 5.5 interrupt is pending then this p 7.5 bit is 1, if 6.5 is pending the p 6.5 is 1 and this is 7.5 is pending then I am sorry so this is this diagram is slightly. So, this p 7.5, 6.5 and 5.5 so 7.5 is a bit number 6; so 7.5 should be at yeah 7.5 is a bit number 6 so this should be 7.5 not 5.5 so this would be 7.5 this should be 5.5, so 5.5, 6.5 and 7.5 so they are storing the corresponding pending bits then this I E bits so this will store whether the interrupt enable flip flop contain so



that will be stored here and this will store the masked bits ok so that way I can through the rim instruction I can get this complete status.

(Refer Slide Time: 24:39)



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.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this bit 0 to 2 they show the current setting of the mask for the RST 7.5, 6.5, 5.5 so they will give you the content of the mask flip flop, they can be used by a program to read the mask setting. Then bit 3 so this is used to show whether the Maskable interrupt processing enable or not it will return the content of the interrupt enable flip flop and it can be used to used by program determine whether interrupts are enabled or not if you want to check whether the interrupts enabled or not so you can execute rim instruction and then check bit number 3 of it to see whether the interrupts are enabled.

(Refer Slide Time: 25:19)

## 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 RST 5.5
  - 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.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Bits 4 to 6 so they are actually for show the pending interrupt as we said for 5.5, 7.5, 6.5 and 5.5, 4 and 5 they returned the status of RST 5.5, 6.5; bit 6 will return that the RST 7.5 memory flip flop so an this 7 bits 7 is for serial data input, then the RIM instruction is for the serial input data whatever SID pin whatever data is coming so it can be it will come to bit number 7.

(Refer Slide Time: 25:53)

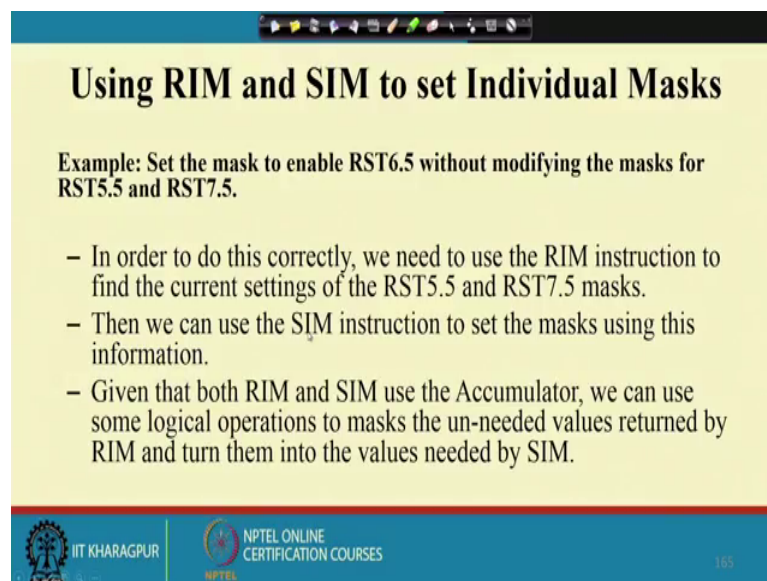
## Pending Interrupts

- Since the 8085 has five interrupt lines, interrupts may occur during an ISR and remain pending.
  - Using the **RIM** instruction, the programmer can read the status of the interrupt lines and find if there are any pending interrupts.
- The advantage is being able to find about interrupts on RST 7.5, RST 6.5, and RST 5.5 without having to enable low level interrupts like INTR.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, so we can since there so many interrupt line the 5 interrupt line so interrupt may occur during ISR and they can remain pending. So, using the RIM instruction we can check whether any interrupts are pending and if you do not want to respond to those interrupt accordingly we can clear those flip flop, so this is the advantage of being able to find out the interrupt of RST say; 7.5, 6.5, 5.5 without hurry to enable the low level interrupt like this INTR so we do not want to reset enable INTR, so we want to just access other interrupts so we can do that.

(Refer Slide Time: 26:39)



**Using RIM and SIM to set Individual Masks**

**Example: Set the mask to enable RST6.5 without modifying the masks for RST5.5 and RST7.5.**

- In order to do this correctly, we need to use the RIM instruction to find the current settings of the RST5.5 and RST7.5 masks.
- Then we can use the SIM instruction to set the masks using this information.
- Given that both RIM and SIM use the Accumulator, we can use some logical operations to mask the un-needed values returned by RIM and turn them into the values needed by SIM.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 165

So, this is an example to show how to set mask to enable RST 6.5 without modifying the mask for 5.5 and 7.5. So, 5.5 and 7.5 whatever setting is done that should continue we just want to enable the RST 6.5 bit. So, how to do this? So, we have to first execute a RIM instruction to know current setting of 5.5 and 7.5 and then we have to use SIM instruction after modifying the accumulator.

(Refer Slide Time: 27:13)

### Using RIM and SIM to set Individual Masks

— 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.
```

SIO	SDE	R7.5	MSE	M7.5	M6.5	Accumulator
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0

So, we can we can do this thing. So, suppose the current setting of RST 5.5 and 7.5 there are enable and interrupt processor is disabled so this is the current setting. So, if you if you executive a RIM instruction you will get the bit pattern like this, because this 5.5 and 7.5 so there enabled 6.5 is disabled now and interrupt enable is disabled I is disabled.

Now so after getting this patter, so we have to now we want to set a new mask so this bit 4 MSE bit will be set to 1 so that so we or this content with a 0 8 X. So, this content is odd with 0 8 X so this bit becomes (Refer Time: 29:58) and then we do an ANI immediate with 0DH. So, when you do this; so this will turned off the serial data that is this SDE bit will be turned off and we want to but you do not reset that 7.5 we do not want to talk there is a 7.5.

So, that way the 7.5 will not be touched, this M 7.5 so whatever will be the value so it is ended with 1 so the value will be just coming here then, this then 6.5 is turned off. So, 6.5 is this 1, so this M 6.5 so this is 0 this is 1 so this 1 is for this coming for this MSC then this 7.5 so it continues. So, 6.5 so this bit is made 0 so this will be ended with that and accordingly we will we will get a 0 here.

So, this process will this M 6.5 so this interrupt was enabled, so this interrupt was disabled and then after this so this interrupt becomes enabled, where as the status of this

7.5 and 5.5 they remain unchanged, 6.5 it was disabled previously now it becomes enabled. So, without touching other interrupt so you can do safe reset in the particular interrupt.