8085 INTERRUPTS

Introduction

- **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.
- **Interrupts can be classified into two types:**
  - Maskable (can be delayed)
  - Non-Maskable (can not 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)

Introduction

- **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 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 Microprocessor

8085 Interrupt

- 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.

8085 Interrupt

- **The 8085 has 5 interrupt inputs.**
  - The INTR input.
  - The INTR input is the only non-vectored interrupt.
  - INTR is maskable using the EI/DI instruction pair.
  - 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.
  - TRAP is the only non-maskable interrupt in the 8085
  - TRAP is also automatically vectored

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

Interrupt Vector

- 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”

Non-Vectored Interrupt

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 flopand 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.

Non-Vectored Interrupt

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 Non-Vectored Interrupt Process

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

Restart Instruction	Equivalent to
RST0	CALL 0000H
RST1	CALL 0008H
RST2	CALL 0010H
RST3	CALL 0018H
RST4	CALL 0020H
RST5	CALL 0028H
RST6	CALL 0030H
RST7	CALL 0038H

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.

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:

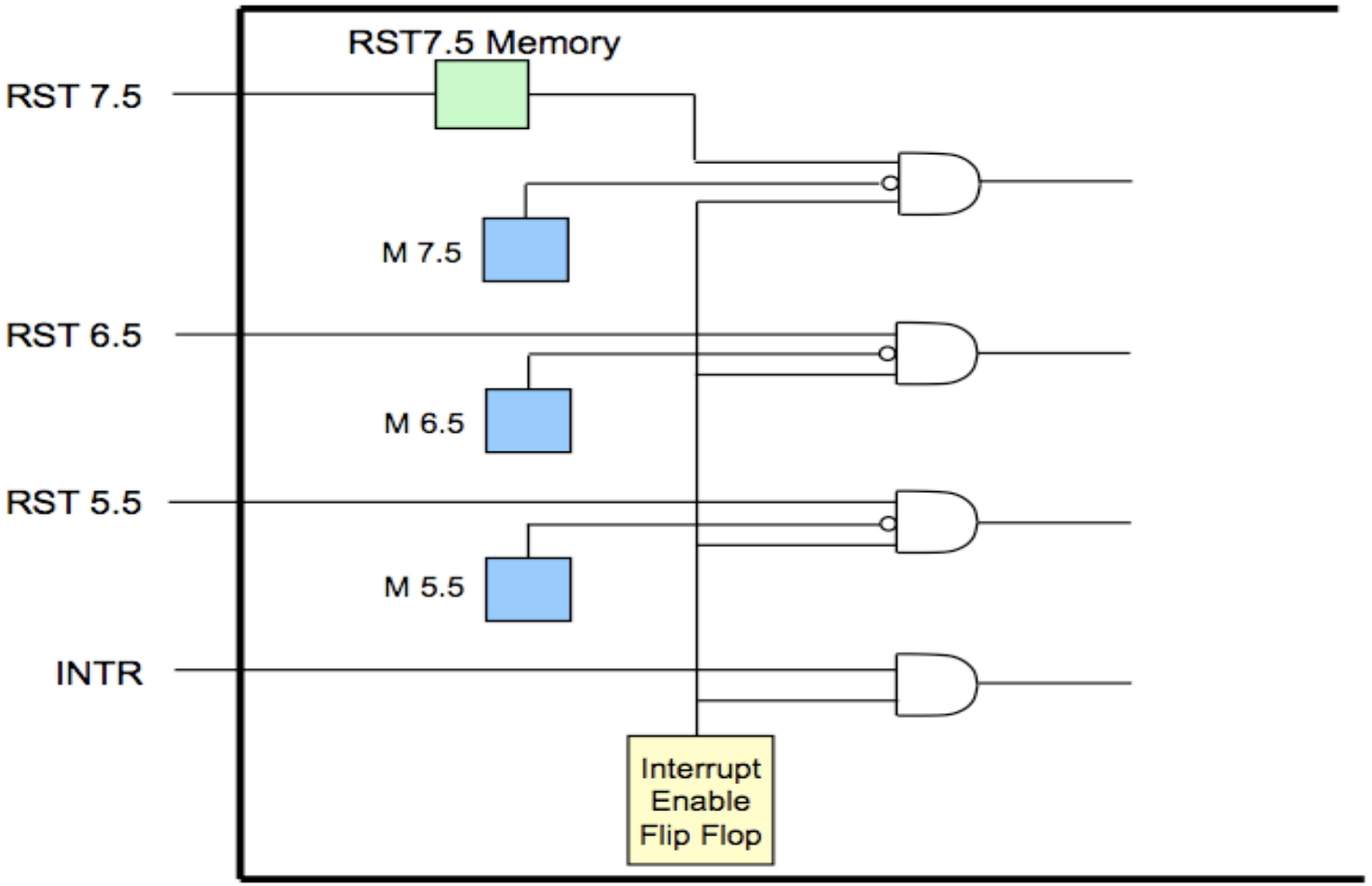
Interrupt	Vector
RST 5.5	002CH
RST 6.5	0034H
RST 7.5	003CH

      - 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

- **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.

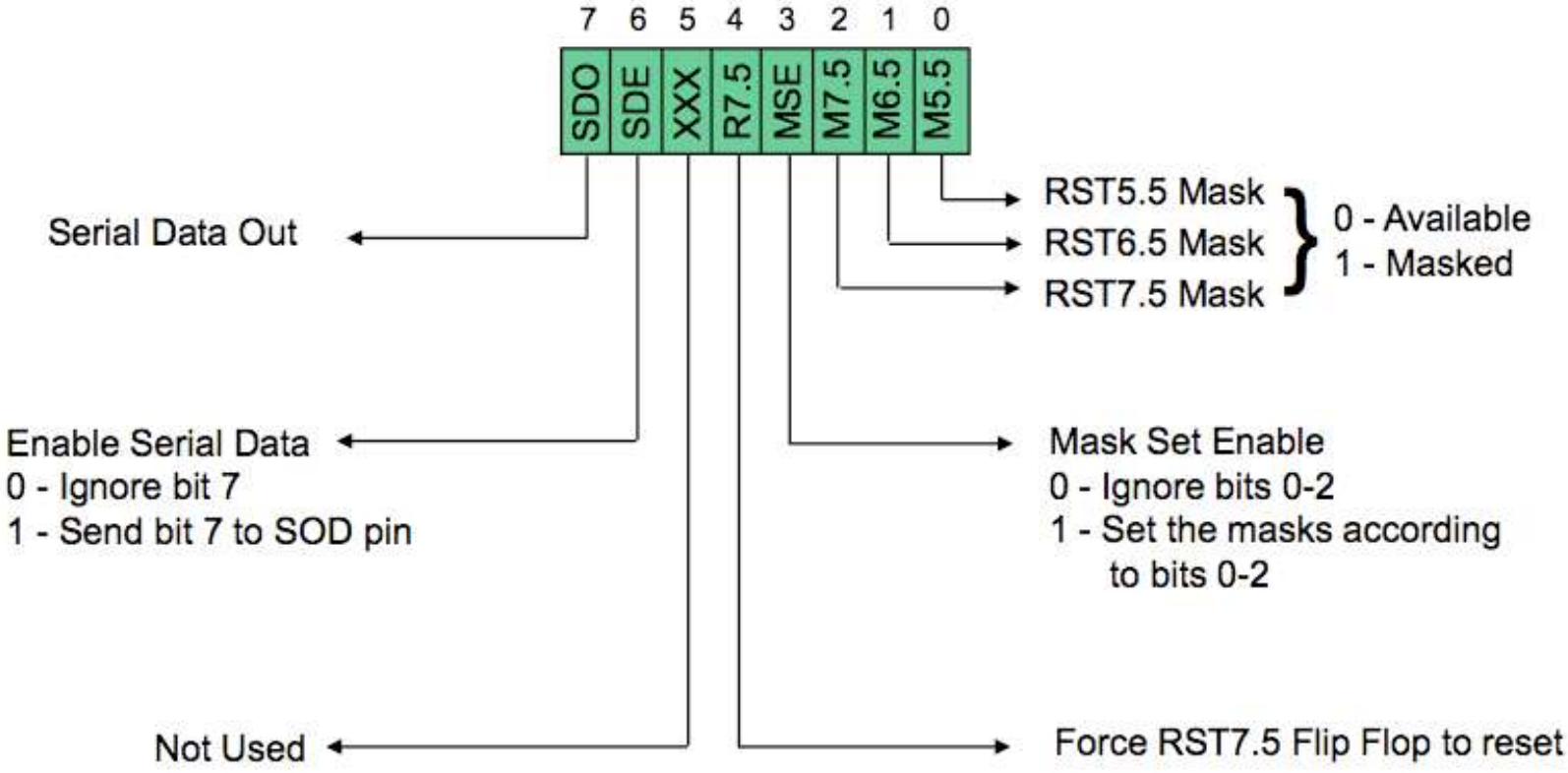
Hardware of EI



Manipulating the Mask

- **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


7 6 5 4 3 2 1 0
SDO
SDE
XXX
R7.5
MSE
M7.5
M6.5
M5.5
Serial Data Out
RST5.5 Mask
RST6.5 Mask
RST7.5 Mask
0 - Available
1 - Masked
Enable Serial Data
0 - Ignore bit 7
1 - Send bit 7 to SOD pin
Mask Set Enable
0 - Ignore bits 0-2
1 - Set the masks according to bits 0-2
Not Used
Force RST7.5 Flip Flop to reset

SIM and 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

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 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

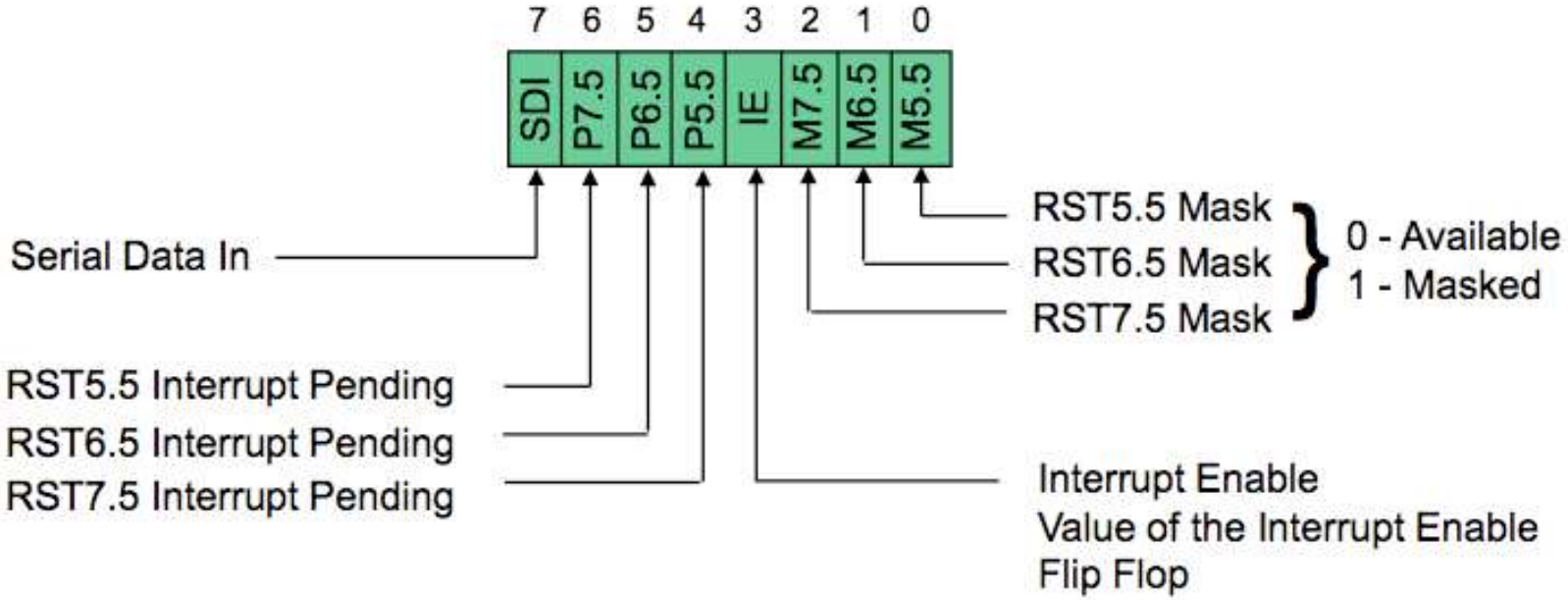
Contents of accumulator are: 0AH

```
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 Level

- **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.**

RIM


7 6 5 4 3 2 1 0
SDI
P7.5
P6.5
P5.5
IE
M7.5
M6.5
M5.5
Serial Data In
RST5.5 Interrupt Pending
RST6.5 Interrupt Pending
RST7.5 Interrupt Pending
RST5.5 Mask
RST6.5 Mask
RST7.5 Mask
0 - Available
1 - Masked
Interrupt Enable
Value of the Interrupt Enable Flip Flop

RIM instruction

- **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.

RIM instruction

- 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.

Using RIM and SIM to set Individual Masks

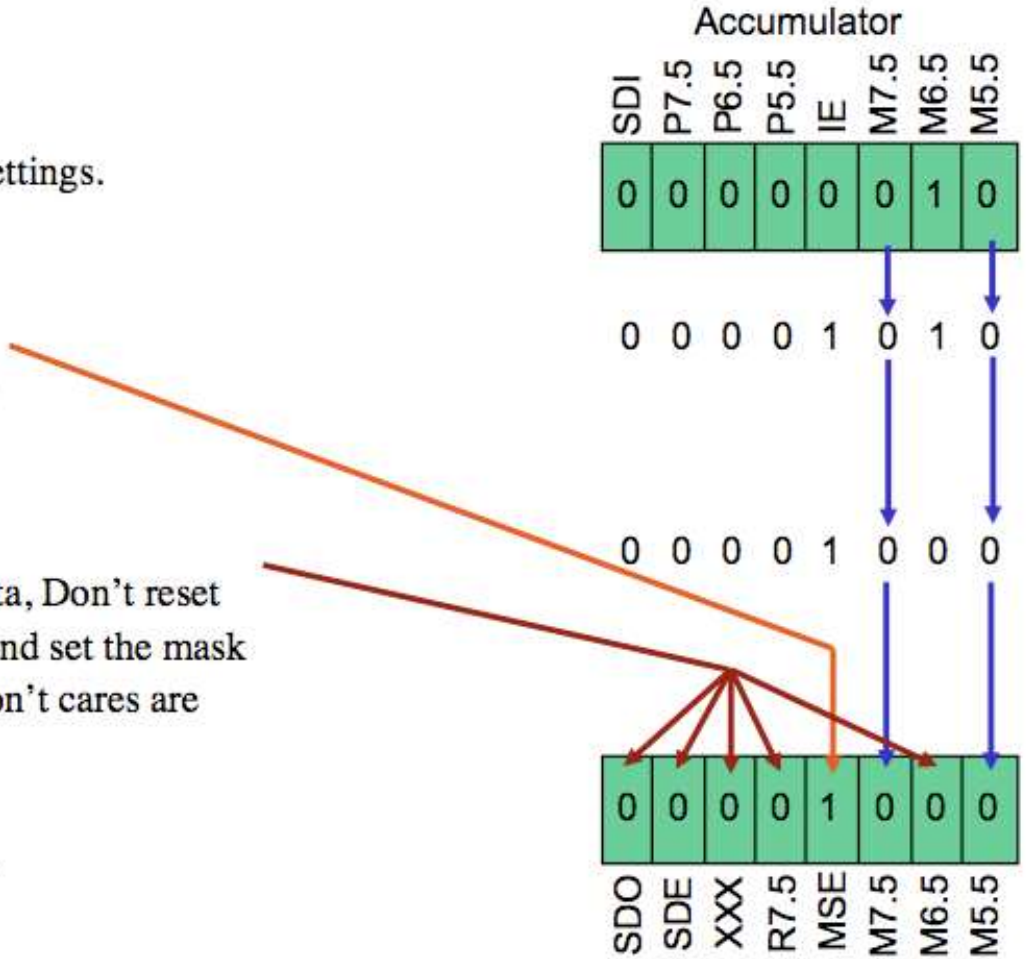
– Assume the RST5.5 and RST7.5 are enabled and the interrupt process is disabled.

```
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.
```

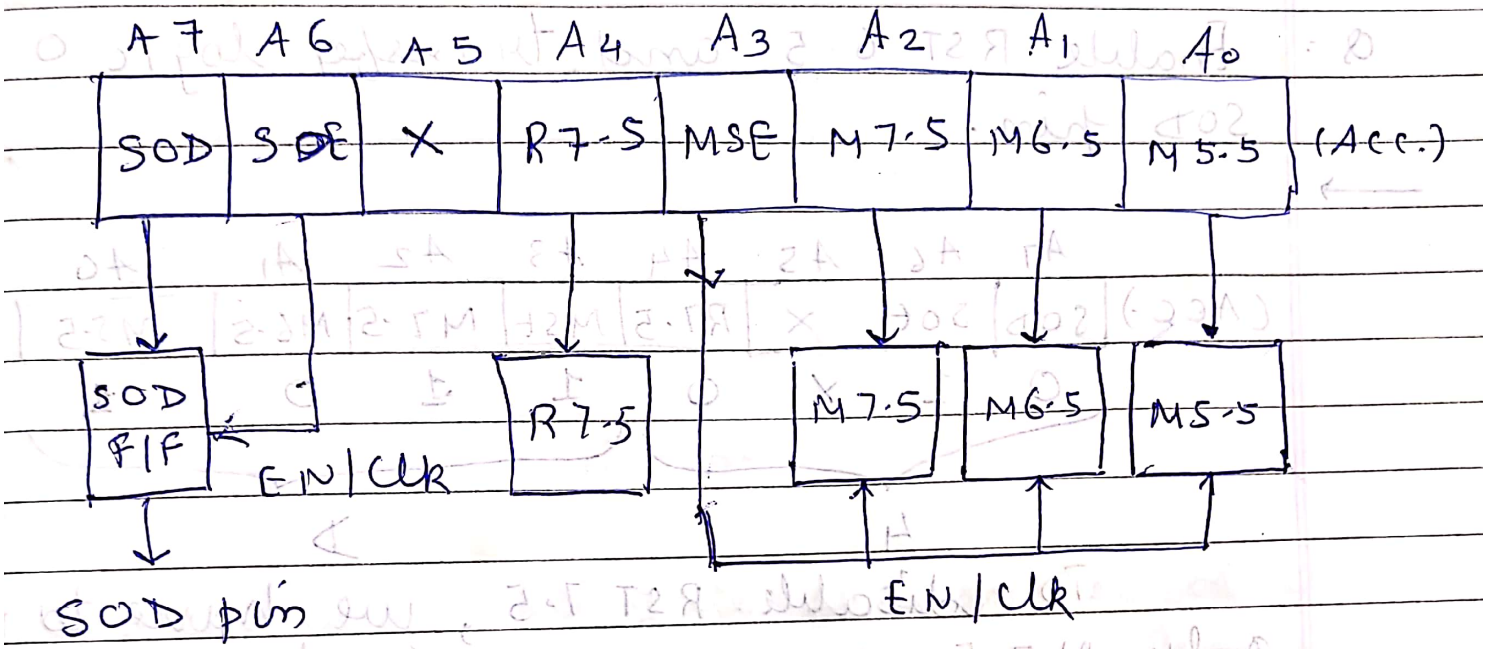


Trap

- **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.

Machine control instns cont:

SIM; [Set Interrupt Mask]
(D/5 Pg. 7)



#D/5#- **SIM : [No addressing Mode] [Single Byte Instruction]**
[No Flags are changed] [machine cycle = 1 , T- states = 4]

When µp executes **SIM instruction, then 8-bit data of accumulator is transferred to different** flipflops as shown in fig.

If A3=MSE (Masking Set Enable)=1 then binary bit present in A2A1A0 filpflop of accumulator are **transferred to masking flipflops M7.5, M6.5, M5.5 respectively.**

If A3=MSE =0 then binary bit present in A2A1A0 flipflops of accumulator are not transferred to masking flipflops, so A2A1A0=XXX (don't care).

The binary bit present in A4 flipflop is always transferred into R7.5 flipflop.

If A6=SOE (Serial Output Enable)=1 then the binary bit present in A7 flipflop of accumulator is stored in SOD flipflop and this bit is continuously available on SOD (Serial Output data) pin.

If A6=SOE=0 then A7 bit is not stored into SOD flipflop so A7=X (don't care).

Q:- Write a program to enable RST 5.5 and RST 7.5 without changing data of SOD pin.

→ To enable RST 5.5, we have to make SIM = 1 & MSS = 0

To enable RST 7.5, we have to make SIM = 1, M7.5 = 0, R7.5 = 0.

8-bit data of accumulator for SIM is

	A7	A6	A5	A4	A3	A2	A1	A0
(Acc.)	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5
	X	0	X	0	1	0	1	0
			0				A	

```
MVI A, 0AH
SIM
EI ; SIM = 1
HLT/RST 1
```

Q:- Enable RST 6.5 and transfer logic 0 on SOD pin.

→

	A7	A6	A5	A4	A3	A2	A1	A0
(ACC.)	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5
	0	1	X	0	1	1	0	1
			4				D	

To disable RST 7.5, we have to make only M7.5 = 1 and R7.5 is kept 0. R7.5 is made 1 only to reset RST 7.5 flip flop (2)

```
MVI A, 4DH
SIM
EI
HLT/RST 1
```

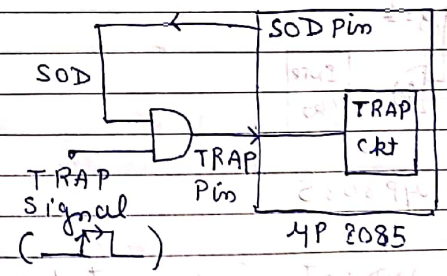
Q:- Set SOD pin without changing data of masking f/f.

	A7	A6	A5	A4	A3	A2	A1	A0
	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5
	1	1	X	0	0	0	0	0
			C				0	

```
MVI A, C0H
SIM ; SOD = 1
HLT/RST 1
```

Q:- Draw the hardware and WAP to disable TRAP.

→



If External AND gate is connected as shown in fig. then TRAP becomes maskable interrupt. If SOD = 1, then TRAP is enabled. and if SOD = 0, then TRAP is disabled.

To disable TRAP, we have to make SOD pin = 0.

8-bit data of acc. for SIM to make SOD=0 is

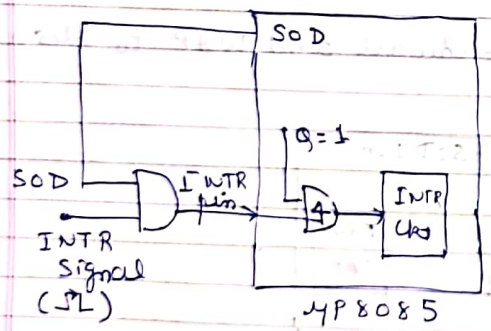
	A7	-	-	-	-	-	-	-	A1	A0
(ACC)	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5		
	0	1	X	X	0	X	X	X		

4 0

```
MVI A, 40H
SIM ; SOD = 0
HLT / RST 1
```

Q: Draw the hardware & WAP to enable RST 5.5 and disable INTR.

→



If external AND gate is connected as shown, then INTR signal can be controlled by two methods. To enable INTR, we have to make SOD = 1 and Q = 1 (INTE f/f) - F

To enable RST 5.5, we have to make Q = 1 & M5.5 = 0. At the same time, we have to disable INTR, so with Q = 1, INTR can be disabled by

Strive for ...

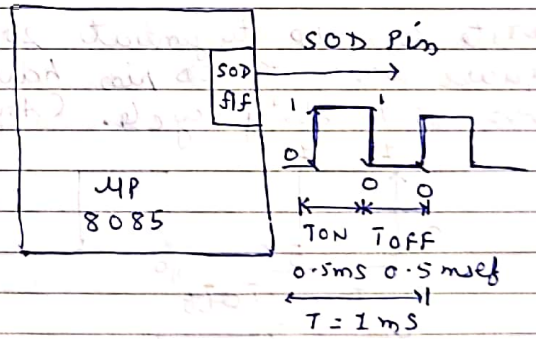
	A7	A6	-	-	-	-	-	-	A1	A0
(ACC.)	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5		
	0	1	X	0	1	1	1	0		

4 E

```
MVI A, 4EH
SIM ; M5.5 = 0,
      SOD = 0
EI  ; Q = 1
HLT / RST 1
```

*Q: Write an ALP to generate 1 KHz square wave on SOD pin of µP.

→



Freq. F = 1 KHz

Time $T = \frac{1}{F} = 1$ msec

Square wave logic 1 and logic 0 times are equal ie

$$T_{ON} = T_{OFF} = \frac{T}{2} = 0.5 \text{ msec}$$

8-bit data of acc. for SIM to make SOD = 0/1 is

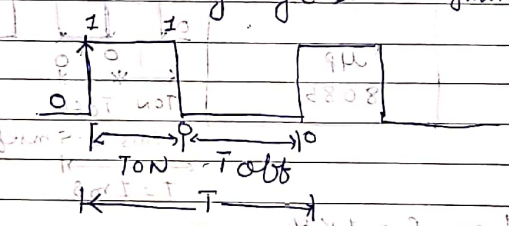
A7 A6 - - - - - - - A0

(ACC:)	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5
	1	1	X	0	0	X	X	X = C0H (SOD=1)
	0	1	X	0	0	X	X	X = 40H (SOD=0)

```
      LXI SP, stack ; Stack initialized for CALL instn.
L1:   MVI A, 0C0H ; initial value of SOD = 0
      SIM ; SOD = 1
      CALL Delay ; To give gap of 0.5msec
      MVI A, 40H
      SIM ; SOD = 0
      CALL Delay ; 0.5msec gap
      JMP L1
```

Q: Write an ALP to generate 200 cycles of a square wave on SOD pin having freq 100 Hz and 40% duty cycle (Asymmetric duty cycle)

→



Freq. F = 100 Hz

Time period $T = \frac{1}{F} = \frac{1}{100\ Hz} = 0.01\ sec = 10\ msec$

% duty cycle $= \frac{T_{on}}{T} \times 100$

$40 = \frac{T_{ON}}{10\ msec} \times 100$

$T_{ON} = 4\ msec$

$T_{off} = 6\ msec$

```
      LXI SP, stack ;
      MVI B, 0C8H ; counter for 200 cycles
L1:   MVI A, 0C0H
      CALL Delay 1
      MVI A, 40H
      SIM
      CALL delay 2
      DCR B
      JNZ L1
      HLT/RST 1
```

Write an ALP to generate asymmetrical square wave on SOD pin of 4p. The pulse ON time & off time are controlled by two diff. programs. Generate 1000 cycles.

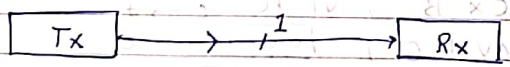
count = (1000)D = (03E8)H

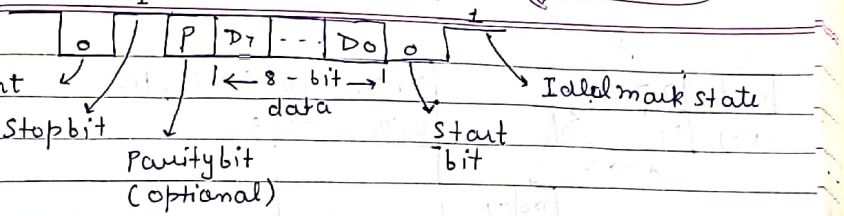
```
      LXI SP, stack
      LXI B, 03E8H
L1:   MVI A, 0C0H
      SIM
      CALL Delay 1
      MVI A, 40H
      SIM
      CALL Delay 2
      DCX B ; NFAC  C → 4
      MOV A, C ; 00000000
      ORA B ; 00000000 & B
      JNZ L1 ; 0000 0000 → 1
      HLT
```

Difference b/t parallel data transfer and serial data transfer:

Parallel data transfer	Serial data transfer
1. If 8 bits are transferred together from source to destinat^n, then it is called parallel data transfer.	1. If only one bit is transfered at a time from source to destination, then it is called serial data transfer
2. This data transfer is fast.	2. This data transfer is slow
3. 8 transmission lines are reqd. to transfer 8 bit data together.	3. only one x-mission line is reqd. to transfer 1-bit data at a time.
4. As the cost of transmission line will be more, so it is costly.	4. As only one x-mission line is reqd, so it is less costly.
5. Parallel data transfer is used for short distance.	5. It is used for long distance data transfer

Universal standard format for Asynchronous serial data x-fer:





Each 8-bit data is transferred in standard format as shown in fig. above:

(a) Under ideal cond^n, transmitter will transfer logic 1 (mark state).

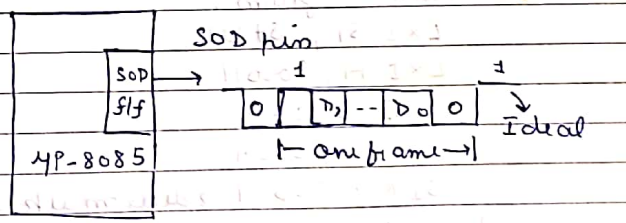
(b) Before each data, start bit = 0 is transfered.

(c) Now, 8-bit data is transferred bit by bit starting from LSB D_0.

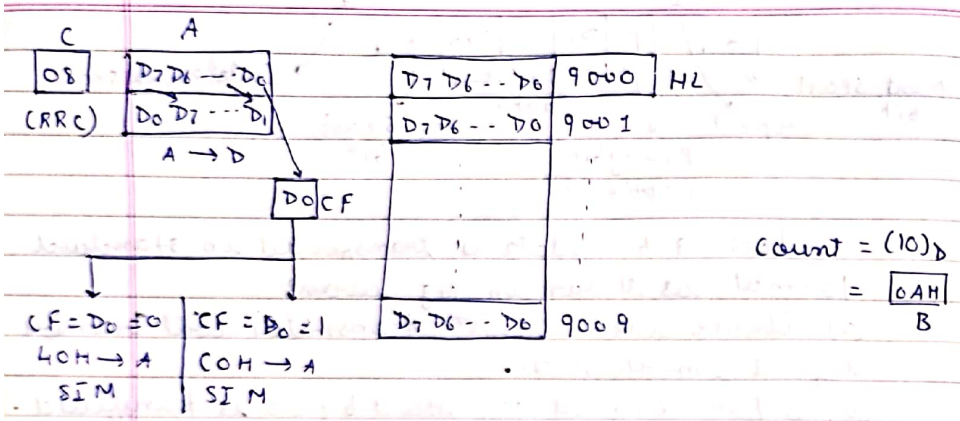
(d) After data parity bit is transferred (optional) Parity bit is used to detect error in serial data transfer. But if only one bit out of 8-bit data changes during data transfer, then it can be detected using parity bit.

(e) In the last of each data, 1 stop bit = 1 is transfed

Q: WALP to transfer 10 bytes on SOD pin in asynchronous format at 200 Band (bits/sec). This 10 bytes are present in mem. from addr. 9000H.



A subprogram Sout is used to transfer only one 8-bit data bit by bit on SOD pin. To transfer 10 bytes on SOD pin, we have to call the S/P 10 bytes.



Speed of serial data transfer is called as Baud rate.

Baud rate = 200 bits/sec

∴ Time reqd to Xfer 1 bit

$$T = \frac{1}{\text{Baud}} = \frac{1}{200} \text{ sec/bit}$$

$$T = \frac{1}{200} \times 1000 = 5 \text{ ms/bit}$$

So, after transfer of each bit, we have to give delay of 5 ms sec.

Main Program:

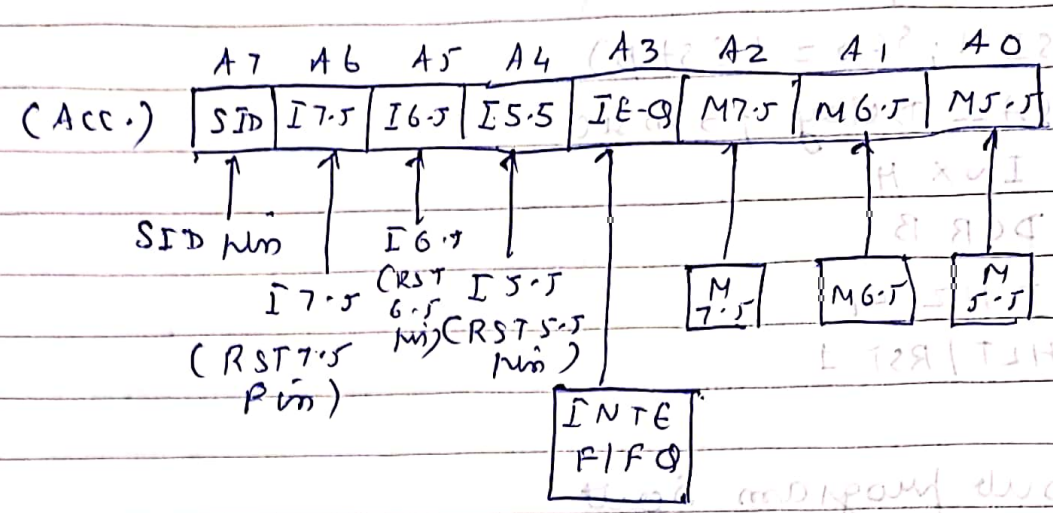
```
LXI SP, Stack
LXI H, 9000H
MVI B, 0AH
MVI A, 0C0H
SIM ; SOD = 1 ; ideal mark state
L1 : MVI A, 40H
SIM ; SOD = 0 = start bit
CALL Delay ; 5 m sec
CALL Sout ; 8 bit data transfer on SOD bit by bit
MVI A, 0C0H
SIM ; SOD = 1 (stop)
CALL Delay ; 5 m sec
INX H
DCR B
JNZ L1
HLT / RST 1
```

Sub program Sout

This sub program is used to transfer 8-bit data on SOD pin bit by bit. The address of mem. loc^n is already present in reg. pair HL.

```
Sout : MOV A, M
       MVI C, 08H ; counter for 8-bits i.e 1 byte
       RRC ; [D0] CF
       MOV D, A
       JNZ L2
       MVI A, 0C0H
L4 :   SIM ; D0 bit is transferred on SOD pin
       CALL Delay ; 5 m sec
       MOV A, D
       DCR C
       JNZ L3
       RET
L2 :   MVI A, 40H
       JMP L4
```

RIM :- (D16 Pg:8)



If RST 5.5 pin = 1, then I 5.5 = 1 and it is said 5.5 is pending.

If RST 6.5 pin = 1, then I 6.5 = 1 and RST 6.5 is pending.

If o/p of RST 7.5 f/f = 1, then I 7.5 = 1 and RST 7.5 is pending.

If any interrupt is pending, then to execute this interrupt, we have to enable the corresponding interrupt. Hence, up will execute this higher priority pending interrupt by hardware.

higher priority

#D/6#-- RIM [No addressing mode] [Single Byte Instruction]
[No flags are changed] [machine cycle = 1 , T- states = 4]

When µp executes RIM instruction then 8-bit data is transferred from different flip-flops to accumulator as shown in the format

→ The binary bits present in masking flipflop M7.5, M6.5, M5.5 are transferred to A2, A1, A0 flipflops of accumulator.

The binary bits present in interrupt enable flipflop-1 i.e. Q is transferred to A3 flipflop of accumulator.

These four LSBs of accumulator A3 to A0 will indicate the status of four maskable interrupt of 8085.

I7.5, I6.5, I5.5 are pending status flipflop of RST7.5, RST6.5, RST5.5

Note : ① If µP is executing subprogram of higher priority interrupt RST 7.5 and lower priority interrupt RST 5.5 is pending. Then, we have to keep RST 5.5 as pending ie we have to keep Q = 0. But, after completing subprogram of RST 7.5 interrupt, we have to make Q = 1 by giving inst^n EI before RET inst^n.

② If µP is executing subprogram of lower priority interrupt like RST 5.5 and higher priority interrupt like RST 7.5 is pending, then we have to execute higher priority pending interrupt RST 7.5. So, we have to enable RST 7.5 by making Q = 1 (EI) in the middle of lower priority interrupt RST 5.5 interrupt sub program. Hence, µP will branch over lower priority RST 5.5 S/P to higher priority RST 7.5 S/P and the priority of interrupts is not disturbed.

* (a) The common control is O/P of INTE F/F ①
Q = 0, so all the four maskable interrupts are disturbed.
(b) I 5.5 = 1, so RST 5.5 is pending.
(c) I 6.5 = 0, so RST 6.5 is not pending.
(d) I 7.5 = 1 so, RST 7.5 is pending.
(e) The binary bit is obtained from SID pin = 1.

Ex: Enable RST 7.5 without changing data of M 6.5 & M 5.5.

→ To enable RST 7.5, we have to make Q = 1, M 7.5 = 0, R 7.5 = 0.
8 - bit data of Acc. reqd. for SIM is

Note: M 6.5 and M5.5 bit should not change. To satisfy this condn, we have to read the old bits of M 6.5 & M5.5 f/fs (RIM) and this same old bits are stored into M6.5 & M5.5 f/fs using SIM instn.

	A7	A6	A5	A4	A3	A2	A1	A0	
SIM	SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5	(ACC)
	=X	=0		=0	=1	=0	old	old	
	0	0	0	0	1	0	M6.5 old	M5.5 old	

→ RIM ;

	A7	A6	A5	A4	A3	A2	A1	A0
	SID	I7.5	I6.5	I5.5	Q	M7.5 old	M6.5 old	M5.5 old
ANI 03H ;	0	0	0	0	0	0	1	1

ACC	0	0	0	0	0	0	M6.5 old	M5.5 old
ORI 08H ;	0	0	0	0	1	0	0	0
ACC	0	0	0	0	1	0	M6.5 old	M5.5 old

SIM ;
EI ; Q = 1
HLT / RST 1

Ex: Enable RST 6.5 without changing status of RST 7.5 & RST 5.5.

SOLN

SIM

SOD	SOE	X	R7.5	MSE	M7.5	M6.5	M5.5
(0)	0	(0)	0	1	M7.5 old	0	old

A7 - - - - - - - A0

RIM ;

SID	I7.5	I6.5	I5.5	IE=0	M7.5 old	M6.5 old	M5.5 old

ANI 05H ; 0 0 0 0 0 1 0 1

ORI 08H ;

0	0	0	0	0	M7.5 old	0	M5.5 old

0 0 0 0 1 0 0 0

SIM

0	0	0	0	1	M7.5 old	0	M5.5 old

EI

HLT | RST 1