7.6 SHORT QUESTIONS AND ANSWERS

7.1 ***What is an interrupt ?***

An interrupt is a signal sent by an external device to the processor so as to request the processor to perform a particular task or work.

7.2 ***How are interrupts classified ?***

There are three methods of classifying interrupts:

Method I : The interrupts are classified into hardware and software interrupts.

Method II : The interrupts are classified into vectored and non-vectored interrupts.

Method III : The interrupts are classified into maskable and non-maskable interrupts.

7.3 ***Explain how a microprocessor services an interrupt request ?***

When the processor recognizes an interrupt, it saves the processor status in the stack. Then it calls and executes an Interrupt Service Routine (ISR). At the end of the ISR, it restores the processor status and the program control is transferred to the main program.

7.4 ***What is the role of the interrupt services routine?***

For each interrupt, the processor has to perform a specific job. An interrupt service routine has been developed in order to perform the operations required for a device that is interrupting the processor.

7.5 ***How are interrupts affected by system reset in an 8085 ?***

Whenever the 8085 processor is reset, all the interrupts except TRAP are disabled. In order to enable the interrupts, EI instruction has to be executed after a reset.

7.6 ***What are software interrupts?***

Software interrupts are program instructions. These instructions are inserted at desired locations in a program. While running a program, if a software interrupt instruction is encountered then the processor executes an interrupt service routine.

7.7 ***What is hardware interrupt?***

If an interrupt is initiated in a processor by applying an appropriate signal to the interrupt pin, then the interrupt is called a hardware interrupt.

7.8 ***What is the difference between a hardware and software interrupt?***

The software interrupt is initiated by the main program, but the hardware interrupt is initiated by an external device.

In an 8085, the software interrupts cannot be disabled or masked but the hardware interrupts except TRAP can be disabled or masked.

In an 8086, the software interrupts cannot be disabled or masked but the hardware interrupts except NMI can be disabled or masked.

7.9 ***What are vectored and non-vectored interrupts?***

When an interrupt is accepted, if the processor control branches to a specific address defined by the manufacturer, then the interrupt is called a vectored interrupt.

In a non-vectored interrupt there is no specific address for storing the interrupt service routine. Hence the interrupting device should give the address of the interrupt service routine.

7.10 ***What is masking and why is it required?***

Masking is preventing the interrupt from disturbing the current program execution. When the processor is performing an important job (process) and, if the process should not be interrupted then all the interrupts should be masked or disabled.

In a processor with multiple interrupts, the lower priority interrupt can be masked so as to prevent it from interrupting the execution of the interrupt service routine of a higher priority interrupt.

7.11 ***What is vectoring?***

Vectoring is the process of generating the address of an interrupt service routine to be loaded in the program counter.

7.12 ***List the software and hardware interrupts of 8085?***

Software interrupts : RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6 and RST 7.

Hardware interrupts : TRAP, RST 7.5, RST 6.5, RST 5.5 and INTR.

7.13 ***What is TRAP?***

TRAP is a non-maskable interrupt of 8085. It is not disabled by processor reset or after recognition of the interrupt.

7.14 ***State whether HOLD has a higher priority than TRAP ?***

The interrupts including TRAP are recognized only if the HOLD is not valid, hence TRAP has lower priority than HOLD.

7.15 ***When does an 8085 processor accept a hardware interrupt?***

The processor keeps on checking the interrupt pins at the second T-state of the last machine cycle of every instruction. If the processor finds a valid interrupt signal and if the interrupt is unmasked and enabled, then the processor accepts the interrupt. The acceptance of the interrupt is acknowledged by sending an $\overline{\text{INTA}}$ signal to the interrupting device.

7.16 ***List the type of signals that has to be applied to initiate a hardware interrupt in an 8085.***

The TRAP is level and edge-sensitive and so the interrupt signal has to take a **low** to **high** transition and then remain **high,** until it is recognized. The RST 7.5 is edge-sensitive and so the interrupt signal has to take a **low** to **high** transition and need not remain **high,** until it is recognized. The RST 6.5, RST 5.5 and INTR are level-senstive and so the interrupt signal should be **high** until the interrupt is recognized.

7.17 ***What are maskable and non-maskable interrupts of an 8085?***

The TRAP is an non-maskable interrupt. The RST 7.5, RST 6.5 and RST 5.5 are maskable interrupts. The INTR of 8085 can also be disabled by DI instruction.

7.18 ***When does an 8085 processor disable the interrupt system ?***

The interrupts of an 8085 except TRAP are disabled after any one of the following operations:

- Executing the EI instruction.
- System or processor reset.
- After recognition (acceptance) of an interrupt.

7.19 ***What is the function performed by DI instruction?***

The function of DI instruction is to disable the entire interrupt system.

7.20 ***What is the function performed by EI instruction?***

The EI instruction can be used to enable all the interrupts after disabling.

7.21 ***How can the interrupt INTR of 8085 be expanded?***

The interrupt INTR of an 8085 can be expanded of eight interrupts using an 8-to-3 priority encoder. It can also be expanded to eight interrupts using one number of 8259 (Programmable interrupt controller) or upto 64 interrupts using 8259's in cascaded mode.

7.22 ***How can the hardware interrupt of an 8085 can be masked or unmasked?***

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 the accumulator and then executing the SIM instruction. The format of the 8-bit data is shown in Fig. Q7.22.

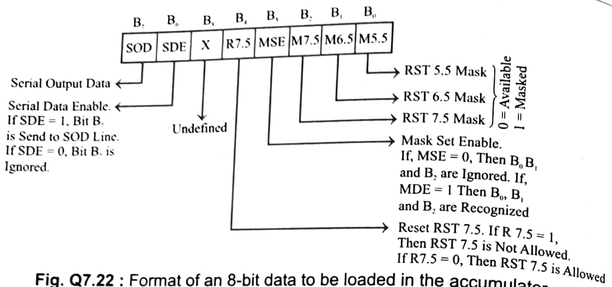

B7 B6 B5 B4 B3 B2 B1 B0
SOD
SDE
X
R7.5
MSE
M7.5
M6.5
M5.5
Serial Output Data
Serial Data Enable. If SDE = 1, Bit B7 is Send to SOD Line. If SDE = 0, Bit B7 is Ignored.
Undefined
RST 5.5 Mask
RST 6.5 Mask
RST 7.5 Mask
0 = Available
1 = Masked
Mask Set Enable. If, MSE = 0, Then B0 B1 and B2 are Ignored. If, MDE = 1 Then B0, B1 and B2 are Recognized
Reset RST 7.5. If R 7.5 = 1, Then RST 7.5 is Not Allowed. If R7.5 = 0, Then RST 7.5 is Allowed

Fig. Q7.22 : Format of an 8-bit data to be loaded in the accumulator before executing a SIM instruction.

7.23 ***How can the status of maskable interrupts be read in an 8085 processor?***

The status of hardware interrupts like interrupt request pending or not, interrupts enabled or not, and masked or unmasked can read from the accumulator after executing the RIM instruction. When the RIM instruction is executed, an 8-bit data is loaded in the accumulator which can be interpreted as shown in Fig. Q 7.23.

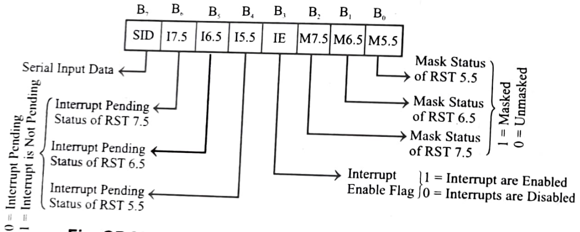


Fig. Q7.23 : Interpretation of the content of accumulator after executing RIM instruction.

7.24 ***How to check whether an 8085 interrupt is masked or not?***

The masking status of an 8085 interrupt can be obtained by executing the RIM instruction. When the RIM instruction is execute, an 8-bit data is loaded in the accumulator. The bits B_0, B_1 and B_2 will give the masking status of RST 5.5, RST 6.5 and RST 7.5 respectively. If this bit is 1, then the corresponding interrupt is masked, otherwise it is unmasked.

7.25 ***How to check the interrupt request pending status of 8085 interrupt?***

The pending status of an 8085 interrupt can be obtained by executing the RIM instruction. When the RIM instruction is executed an 8-bit data is loaded in the accumulator. The bits B_4, B_5 and B_6 will give the pending status of RST 5.5, RST 6.5 and RST 7.5 respectively. If this bit is 1 then the interrupt is pending, otherwise it is not pending.

7.26 ***How are the vector addresses generated for hardware interrupts of an 8085?***

For the hardware interrupts TRAP, RST 7.5, RST 6.5 and RST 5.5 the vector addresses are generated by the processor itself. These addresses are fixed by the manufacturer.

7.27 ***How is the vector address generated for the INTR interrupt of an 8085?***

For the interrupt INTR, the interrupting device has to place either the RST opcode or the CALL opcode followed by the 16-bit address. If the RST opcode is placed, then the corresponding vector address is generated by the processor. In case of CALL opcode, the given 16-bit address will be the vector address.

7.28 ***How are the vector addresses generated for software interrupts of an 8085?***

For the software interrupts RST 0 to RST 7, the vector addresses are generated internal to the processor. These vector addresses are fixed by the manufacturer.

7.29 ***What are the sources of an 8086 interrupt?***

There are three sources for interrupts in an 8086.

1. One source is from an external signal applied to the INTR or NMI pin of the processor.
2. The second source of an interrupt is execution of the interrupt instruction "INT n".
3. The third source of an interrupt is from some condition produced in the 8086 by the execution of an instruction.

7.30 ***What is exception? Give an example.***

Exception is an interrupt generated due to exceptional condition (i.e., impossible situation) which occurs while executing an instruction. An example of exception is divide by zero interrupt in 8086. While executing the division instruction if the divisor is zero, then the 8086 will generate a divide by zero (type-0) interrupt.

7.31 ***How many interrupts are available in an 8086? How are they classified?***

The 8086 has 256 types of interrupts. INTEL has given a type number to the interrupts in the range of 0 to 255_{10}. Type-0 to type-4 are defined by INTEL and they are called INTEL predefined interrupts. Type-5 to type-31 are reserved by INTEL for use in future processors. Type-32 to type-255 are available for the user as hardware or software interrupts.

7.32 ***How can the interrupts be initiated in an 8086?***

The 8086 processor has dual facility of initiating all the 256 interrupts. The interrupts can be initiated either by executing the "INT n" instruction where n is the type number or the interrupt can be initiated by sending an appropriate signal to the INTR pin of the processor.

7.33 ***List the INTEL predefined interrupts.***

The INTEL predefined interrupts are:

i) Divide by zero (Type-0 interrupt)
ii) Single step (Type-1 interrupt)
iii) Non-maskable interrupt, NMI (Type-2 interrupt)
iv) Breakpoint interrupt (Type-3 interrupt)
v) Interrupt on overflow (Type-4 interrupt)

7.34 ***What are software and hardware interrupts of an 8086?***

In 8086 the interrupts initiated by executing "INT n" instruction are called software interrupts.

The interrupts initiated by applying appropriate signals to the INTR and NMI pins of the 8086 are called hardware interrupts.

7.35 ***What are maskable and nonmaskable interrupts of an 8086?***

The hardware interrupts initiated by applying an appropriate signal to the INTR pin of an 8086 are maskable interrupts.

The software interrupts and the hardware interrupt NMI are non-maskable.

7.36 ***How can the interrupts be masked/unmasked in an 8086?***

The maskable interrupts of 8086 can be masked by clearing the interrupt flag to zero and they can be unmasked/allowed by setting the interrupt flag to one.

7.37 ***What is a vector table? Where is it located?***

The memory block consisting of vector addresses of all the 256 types of interrupts of an 8086 is called a vector table. The vector table is stored in the first 1 kb of physical memory space.

7.38 *How is the interrupt address generated in 8086?*

The 8086 will multiply the type number by four and sign extend to 20-bit to get a memory address of the vector table. The vector address for an interrupt will be available in four consecutive memory location starting from this 20-bit address. The first word in the table is the offset address of ISR (Interrupt Service Routine) and the next word is the segment base address of the ISR.

7.39 *What is the need for an interrupt controller?*

The interrupt controller is employed to expand the interrupt input. It can handle the interrupt request from various devices and allow them one by one to the processor.

7.40 *List some of the features of INTEL 8259 (Programmable Interrupt Controller).*

- It manage eight interrupt requests
- The interrupt vector addresses are programmable
- The priorities of interrupts are programmable
- The interrupt can be masked or unmasked individually

7.41 *Write the various functional blocks of INTEL 8259 ?*

The various functional blocks of 8259 are Control logic, Read/ Write logic, Data bus buffer, Interrupt Request Register (IRR), Interrupt Mask Register (IMR) and In-Service Register (ISR), Priority Resolver (PR) and Cascade buffer.

7.42 *What is master and slave 8259 ?*

When 8259s are connected in cascade, one 8259 will be directly interrupting the processor and it is called master 8259. To each interrupt request input of master 8259, one slave 8259 can be connected. The 8259's interrupting the master 8259 are called slave 8259.

7.43 *How is 8259 programmed?*

The 8259 is programmed by sending Initialization Command Words (ICWs) and Operational Command Words (OCWs).

7.44 *What are the features of 8259 that are programmed using ICWs?*

The ICWs are used to program the following features of an 8259:

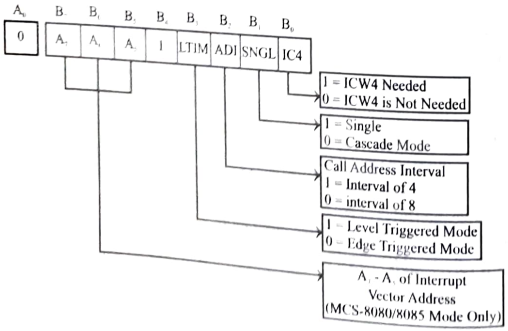
- Call address interval (in case of 8085)
- Cascade mode or single
- Level or Edge triggered
- Vector address (in case of 8085)
  or Type number (in case of 8086)
- 8085 or 8086 modes
- Auto or Normal end of interrupt
- Special fully nested mode

7.45 *What are features of 8259 that can be programmed using OCWs?*

The OCWs are used to program the following features of an 8259:

- Masking of individual interrupts.
- Specific or Non-specific end of interrupt.
- Priority modes.

7.46 *Write the format of ICW1?*



7.47 *What is the difference in programming master 8259 and slave 8259 ?*

The ICW 3 will be different for master 8259 and slave 8259. For master, the ICW3 will inform the IR input that are having slaves. For slave, the ICW3 will inform its slave ID number.

7.48 *When is ICW4 send to 8259 ?*

The ICW4 is send to 8259 to perform any one of the following features:

- 8085 or 8086 mode
- Special fully nested mode
- Auto or Normal end of interrupt
- Buffered or Non-buffered mode.

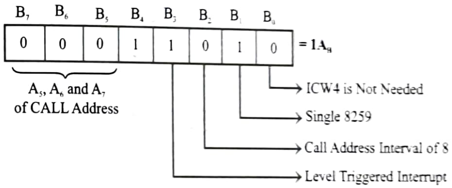
7.49 *Write a program segment to initialize a single 8259 connected to an 8085 processor.*

Let us assume that 8259 is IO-mapped in the system. The 8259 can be initialized by sending ICW1, ICW2 and OCW1. Let the 8-bit address when $A_0 = 0$ be 00_H and when $A_0 = 1$ be 01_H.

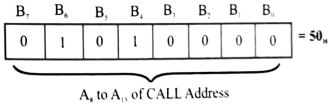
```
MVI A,ICW1  ; Move ICW1 to A-register.
OUT 00H     ; Send ICW1 to 8259.
MVI A,ICW2  ; Move ICW2 to A-register.
OUT 01H     ; Send ICW2 to 8259.
MVI A,OCW1  ; Move OCW1 to A-register.
OUT 01H     ; Send OCW1 to 8259.
HLT         ; Halt program execution.
```

7.50 *Frame the Command words ICW1, ICW2 and OCW1 for initializing a single 8259 interfaced to 8085 with the call address interval of 8 and for level triggered interrupt. Also unmask all interrupt inputs. The desired vector address is 5000_H.*

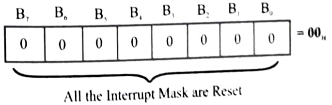
ICW1



ICW2

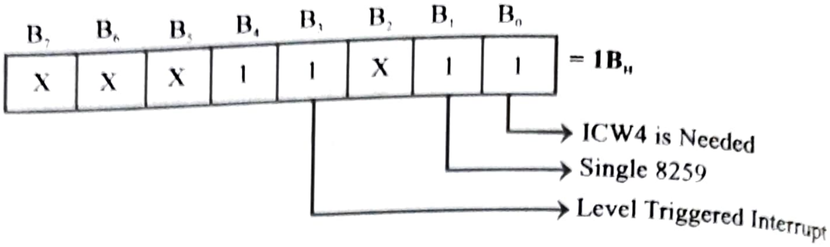


OCW1



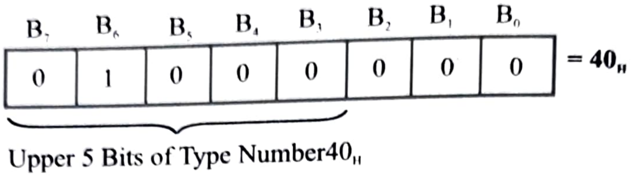
7.51 ***Frame the command words ICW1, ICW2, ICW3 and OCW1 for initializing a single 8259 to initiate INT 40H to INT 47H in an 8086-based system. The desired features are level triggered interrupt and automatic end of interrupt.***

ICW1



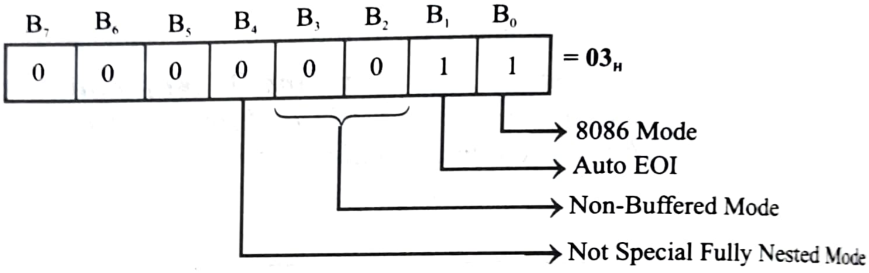
B$_7$	B$_6$	B$_5$	B$_4$	B$_3$	B$_2$	B$_1$	B$_0$	
X	X	X	1	1	X	1	1	= 1B$_H$

ICW2



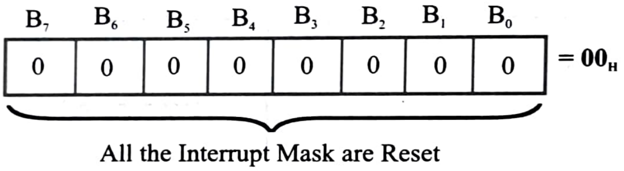
B$_7$	B$_6$	B$_5$	B$_4$	B$_3$	B$_2$	B$_1$	B$_0$	
0	1	0	0	0	0	0	0	= 40$_H$

ICW3



B$_7$	B$_6$	B$_5$	B$_4$	B$_3$	B$_2$	B$_1$	B$_0$	
0	0	0	0	0	0	1	1	= 03$_H$

OCW1



B$_7$	B$_6$	B$_5$	B$_4$	B$_3$	B$_2$	B$_1$	B$_0$	
0	0	0	0	0	0	0	0	= 00$_H$

7.52 ***Write a program segment to initialize a single 8259 connected to an 8086 processor.***

Let us assume that 8259 is IO-mapped in the system with an even address. The 8259 can be initialized by sending ICW1, ICW2, ICW4 and OCW1. Let the 8-bit address with A$_0$ = 0 be 00$_H$ and when A$_0$ = 1 be 02$_H$.

```
MOVAL,ICW1    ;Move ICW1 to AL-register
OUT [00H]    ;Send ICW1 to 8259
MOV AL,ICW2 ;Move ICW2 to AL-register
OUT [02H]    ;Send ICW2 to 8259
MOV AL,ICW4 ;Move ICW4 to AL-register
OUT [02H]    ;Send ICW4 to 8259
MOV AL,OCW1 ;Move OCW1 to AL-register
OUT [02H]    ;Send OCW1 to 8259
HLT          ;Stop
```