1.7 INTERRUPTS

Interrupt is a signal send by an external device to the processor, to the processor to perform a particular task or work. Mainly in the microprocessor based system the interrupts are used for data transfer between the peripheral and the microprocessor.

When a peripheral is ready for data transfer, it interrupts the processor by sending an appropriate signal to the interrupt pin of the processor. If the processor accepts the interrupt then the processor suspends its current activity and executes an interrupt service subroutine to complete the data transfer between the peripheral and processor. After executing the interrupt service routine the processor resumes its current activity. This type of data transfer scheme is called interrupt driven data transfer scheme.

Types of Interrupts

The interrupts are classified into software interrupts and hardware interrupts.

- The software interrupts are program instructions. These instructions are inserted at desired locations in a program. While running a program, lf a software interrupt instruction is encountered, then the processor executes an interrupt service routine (ISR).
- The hardware interrupts are initiated by an external device by placing an appropriate signal at the interrupt pin of the processor. If the interrupt is accepted, then the processor executes an interrupt service routine (ISR).

Software Interrupts of 8085

The software interrupts are program instructions. When the instruction is executed, the processor executes an interrupt service routine stored in the vector address of the software interrupt instruction. The software interrupts of 8085 are RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6 and RST 7.

Table 1.7.1 Interrupt vector address

Interrupt	Vector address
RST 0 RST 1	0000_H 0008_H
RST 2 RST 3	0010_H 0018_H
RST 4 RST 5	0020_H 0028_H
RST 6 RST 7	0030_H 0038_H

Interrupt	Vector address
RST 7.5	$003C_H$
RST 6.5	0034_H
RST 5.5	$002C_H$
TRAP	0024_H

The software interrupt instructions are included at the appropriate (or required) place in the main program. When the processor encounters the software instruction, it pushes the content of PC (Program Counter) to stack. Then loads the Vector address in PC and starts executing the Interrupt Service Routine (ISR) stored in this vector address. At the end of ISR, a return instruction - RET will be placed. When the RET instruction is executed, the processor POP the content of stack to PC. Hence the processor control returns to the main program after servicing the interrupt. Execution of ISR is referred to as servicing of interrupt. All software interrupts of 8085 are vectored interrupts. The software interrupts cannot be masked and they cannot be disabled.

Hardware Interrupts of 8085

An external device, initiates the hardware interrupts of 8O85 by placing an appropriate signal at the interrupt pin of the processor. The processor keeps on checking the interrupt pins at the second T -state of 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 INTA signal to the interrupted device.

The processor saves the content of PC (program Counter) in stack and then loads the vector address of the interrupt in PC. (If the interrupt is non-vectored, then the interrupting device has to supply the address of ISR when it receives INTA signal). It starts executing ISR in this address. At the end of ISR, a return instruction, RET will be placed. When the processor executes the RET instruction, it POP the content of top of stack to PC. Thus the processor control returns to main program after servicing interrupt. The hardware interrupts of 8085 are TRAP, RST 7.5, RST 6.5, RST 5.5 and INTR. Further the interrupts may be classified into vectored and non-vectored interrupts.

Vectored Interrupt

In vectored interrupts, the processor automatically branches to the specific address in response to an interrupt.

Non-Vectored Interrupt

But in non-vectored interrupts the interrupted device should give the address of the interrupt service routine (ISR).

In vectored interrupts, the manufacturer fixes the address of the ISR to which the program control is to be transferred.

The type of signal that has to be placed on the interrupt pin of hardware interrupts of 8085 are defined by INTEL.

- The TRAP interrupt is edge and level sensitive. Hence, to initiate TRAP, the interrupt signal has to make a low to high transition and then it has to remain high until the interrupt is recognized.
- The RST 7.5 interrupt is edge sensitive (positive edge). To initiate the RST 7.5, the interrupt signal has to make a low to high transition and it need not remain high until it is recognized.
- The RST 6.5, RST 5.5 and INTR are level sensitive interrupts. Hence for these interrupts the interrupting signal should remain high, until it is recognized.

Maskable & Non-Maskable Inetrrupts:

The hardware vectored interrupts are classified into maskable and non-maskable interrupts.

- TRAP is non-maskable interrupt
- RST 7.5, RST 6.5 and RST 5.5 are maskable interrupt.

Masking is preventing the interrupt from disturbing the main program. When an interrupt is masked the processor will not accept the interrupt signal. The interrupts can be masked by moving an appropriate data (or code) to accumulator and then executing SIM instruction. (SIM - Set Interrupt Mask). The status of maskable interrupts can be read into accumulator by executing RIM instruction (RIM - Read Interrupt Mask).

All the hardware interrupts, except TRAP are disabled, when the processor is resetted. They can also be disabled by executing Dl instruction. (Dl-Disable Interrupt).

- When an interrupt is disabled, it will not be accepted by the processor. (i.e., INTR, RST 5.5, RST 6.5 and RST 7.5 are disabled by DI instruction and upon hardware reset).
- To enable (to allow) the disabled interrupt, the processor has to execute El instruction (El-Enable Interrupt).

Interrupt Driven Data Transfer Scheme

The interrupt driven data transfer scheme is the best method of data transfer for effectively utilizing the processor time. In this scheme, the processor first initiates the I/O device for data transfer. After initiating the device, the processor will continue the execution of instructions in the program. Also at the end of an instruction the processor will check for a valid interrupt signal. If there is no interrupt, then the processor will continue the execution.

When the I/O device is ready, it will interrupt the processor. On receiving an interrupt signal, the processor will complete the current instruction execution and saves the processor status in stack. Then the processor calls an interrupt service routine (ISR) to service the interrupted device. At the end of ISR the processor status is retrieved from stack and the processor starts executing its main program. The sequence of operations for an interrupt driven data transfer scheme is shown in figure below.

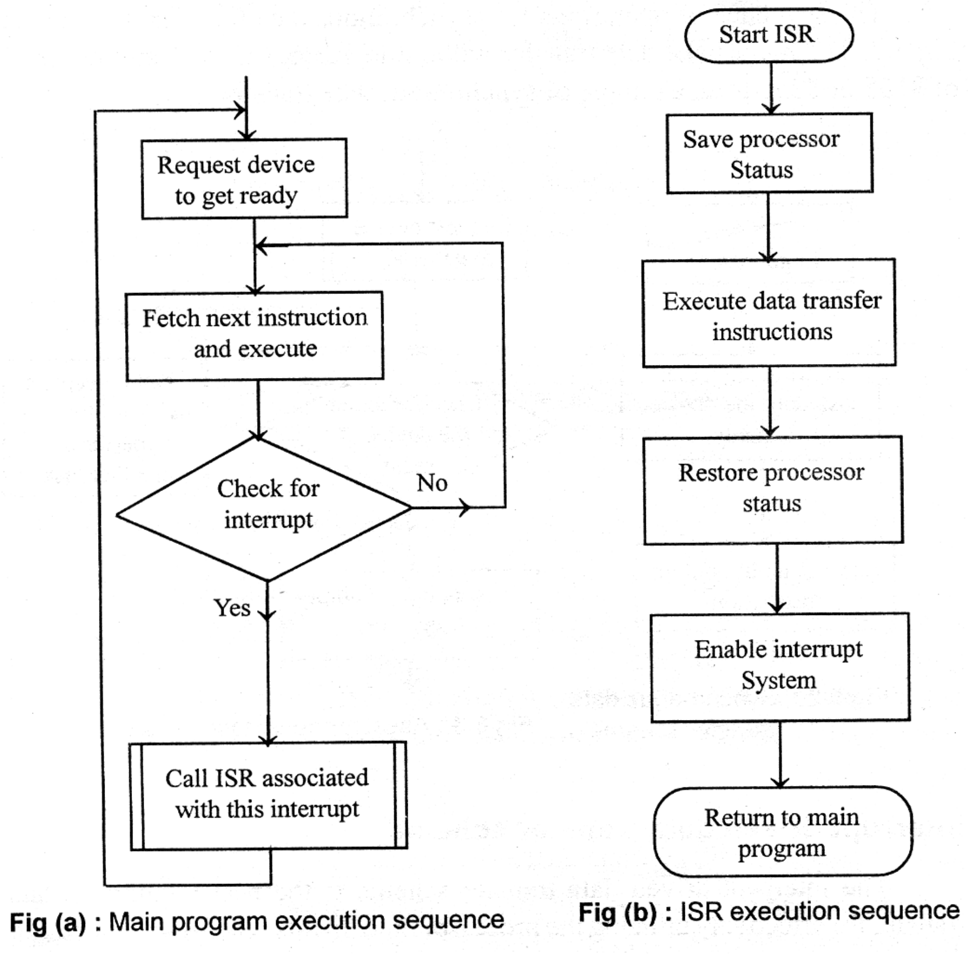
Request device to get ready
Fetch next instruction and execute
Check for interrupt
No
Yes
Call ISR associated with this interrupt
Fig (a) : Main program execution sequence
Start ISR
Save processor Status
Execute data transfer instructions
Restore processor status
Enable interrupt System
Return to main program
Fig (b) : ISR execution sequence

Figure 1.7.1 sequence of operations for an interrupt

[Source: "Microprocessor Architecture Programming and Application" by R.S. Gaonkar, page-295]