8085 Addressing Modes & Interrupts

Now let us discuss the addressing modes in 8085 Microprocessor.

Addressing Modes in 8085

These are the instructions used to transfer the data from one register to another register, from the memory to the register, and from the register to the memory without any alteration in the content. Addressing modes in 8085 is classified into 5 groups −

Immediate addressing mode

In this mode, the 8/16-bit data is specified in the instruction itself as one of its operand. **For example:** MVI K, 20F: means 20F is copied into register K.

Register addressing mode

In this mode, the data is copied from one register to another. **For example:** MOV K, B: means data in register B is copied to register K.

Direct addressing mode

In this mode, the data is directly copied from the given address to the register. **For example:** LDB 5000K: means the data at address 5000K is copied to register B.

Indirect addressing mode

In this mode, the data is transferred from one register to another by using the address pointed by the register. **For example:** MOV K, B: means data is transferred from the memory address pointed by the register to the register K.

Implied addressing mode

This mode doesn't require any operand; the data is specified by the opcode itself. **For example:** CMP.

Interrupts in 8085

Interrupts are the signals generated by the external devices to request the microprocessor to perform a task. There are 5 interrupt signals, i.e. TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR.

Interrupt are classified into following groups based on their parameter −

- **Vector interrupt** − In this type of interrupt, the interrupt address is known to the processor. **For example:** RST7.5, RST6.5, RST5.5, TRAP.
- **Non-Vector interrupt** − In this type of interrupt, the interrupt address is not known to the processor so, the interrupt address needs to be sent externally by the device to perform interrupts. **For example:** INTR.

- **Maskable interrupt** − In this type of interrupt, we can disable the interrupt by writing some instructions into the program. **For example:** RST7.5, RST6.5, RST5.5.
- **Non-Maskable interrupt** − In this type of interrupt, we cannot disable the interrupt by writing some instructions into the program. **For example:** TRAP.
- **Software interrupt** − In this type of interrupt, the programmer has to add the instructions into the program to execute the interrupt. There are 8 software interrupts in 8085, i.e. RST0, RST1, RST2, RST3, RST4, RST5, RST6, and RST7.
- **Hardware interrupt** − There are 5 interrupt pins in 8085 used as hardware interrupts, i.e. TRAP, RST7.5, RST6.5, RST5.5, INTA.

Note − NTA is not an interrupt, it is used by the microprocessor for sending acknowledgement. TRAP has the highest priority, then RST7.5 and so on.

Interrupt Service Routine (ISR)

A small program or a routine that when executed, services the corresponding interrupting source is called an ISR.

TRAP

It is a non-maskable interrupt, having the highest priority among all interrupts. Bydefault, it is enabled until it gets acknowledged. In case of failure, it executes as ISR and sends the data to backup memory. This interrupt transfers the control to the location 0024H.

RST7.5

It is a maskable interrupt, having the second highest priority among all interrupts. When this interrupt is executed, the processor saves the content of the PC register into the stack and branches to 003CH address.

RST 6.5

It is a maskable interrupt, having the third highest priority among all interrupts. When this interrupt is executed, the processor saves the content of the PC register into the stack and branches to 0034H address.

RST 5.5

It is a maskable interrupt. When this interrupt is executed, the processor saves the content of the PC register into the stack and branches to 002CH address.

INTR

It is a maskable interrupt, having the lowest priority among all interrupts. It can be disabled by resetting the microprocessor.

When **INTR signal goes high**, the following events can occur −

- The microprocessor checks the status of INTR signal during the execution of each instruction.
- When the INTR signal is high, then the microprocessor completes its current instruction and sends active low interrupt acknowledge signal.
- When instructions are received, then the microprocessor saves the address of the next instruction on stack and executes the received instruction.

Microprocessor - 8085 Instruction Sets

Let us take a look at the programming of 8085 Microprocessor.

Instruction sets are instruction codes to perform some task. It is classified into five categories.

S.No.	Instruction & Description
1	Control Instructions Following is the table showing the list of Control instructions with their meanings.
2	Logical Instructions Following is the table showing the list of Logical instructions with their meanings.
3	Branching Instructions Following is the table showing the list of Branching instructions with their meanings.
4	Arithmetic Instructions Following is the table showing the list of Arithmetic instructions with their meanings.
5	Data Transfer Instructions Following is the table showing the list of Data-transfer instructions with their meanings.

8085 – Demo Programs

Now, let us take a look at some program demonstrations using the above instructions −

Adding Two 8-bit Numbers

Write a program to add data at 3005H & 3006H memory location and store the result at 3007H memory location.

Problem demo −

```
(3005H) = 14H
   (3006H) = 89H
```

Result −

14H + 89H = 9DH

The program code can be written like this −

```
LXI H 3005H    : "HL points 3005H"
MOV A, M       : "Getting first operand"
INX H          : "HL points 3006H"
ADD M          : "Add second operand"
INX H          : "HL points 3007H"
MOV M, A       : "Store result at 3007H"
HLT            : "Exit program"
```

Exchanging the Memory Locations

Write a program to exchange the data at 5000M& 6000M memory location.

```
LDA 5000M    : "Getting the contents at5000M location into
accumulator"
MOV B, A     : "Save the contents into B register"
LDA 6000M    : "Getting the contents at 6000M location into
accumulator"
STA 5000M    : "Store the contents of accumulator at address
5000M"
MOV A, B     : "Get the saved contents back into A register"
STA 6000M    : "Store the contents of accumulator at address
6000M"
```

Arrange Numbers in an Ascending Order

Write a program to arrange first 10 numbers from memory address 3000H in an ascending order.

```
MVI B, 09          :"Initialize counter"
START              :"LXI H, 3000H: Initialize memory pointer"
MVI C, 09H         :"Initialize counter 2"
BACK: MOV A, M     :"Get the number"
INX H              :"Increment memory pointer"
CMP M              :"Compare number with next number"
JC SKIP            :"If less, don't interchange"
JZ SKIP            :"If equal, don't interchange"
MOV D, M
MOV M, A
DCX H
MOV M, D
INX H              :"Interchange two numbers"
SKIP:DCR C         :"Decrement counter 2"
```

```
JNZ BACK            :"If not zero, repeat"
DCR B               :"Decrement counter 1"
JNZ START
HLT                 :"Terminate program execution"
```