8085 ADDRESSING MODES

Addressing Modes are the different ways by which the µP address (specifies) the operands in an instruction. 8085 supports the following Addressing Modes:

1) Immediate Addressing Mode

In this mode, the **Data** is specified **in** the **Instruction** itself.

```
Eg:   MVI A, 35H      ; Move immediately the value 35 into the Accumulator.
                      ; i.e. A ← 35H
      LXI B, 4000H    ; Move immediately the value 4000 into the register pair BC.
                      ; i.e. BC ← 4000H
```

Advantage:
Programmer can easily **identify** the **operands**.

Disadvantage:
Always more than one byte hence requires **more space**.
The µP requires **two or three machine cycles** to fetch the instruction hence **slow**.

Special Notes:
The "I" in the instruction indicates "Immediate Addressing Mode"
Hence the number in the instruction must be DATA.
Hereafter, when you see a number in any instruction look for an "I".
If "I" is present then the number is DATA else its an address.
The "X" in the instruction (LXI) indicates "Register Pair".

2) Register Addressing Mode

In this mode, the **Data** is specified **in Registers**.

```
Eg:   MOV B, C        ; Move the Contents of C-Register into B-Register.
                      ; i.e. B ← C
      INR B           ; Increments the contents of B-Register.
                      ; i.e. B ← B + 1
```

Advantage:
The µP requires **only one machine cycle** to Fetch the instruction.

Disadvantage:
Operands **cannot** be easily **identified**.

3) Direct Addressing Mode

In this mode, the **Address** of the operand is specified **in** the **Instruction** itself.

Eg: **LDA 2000H** ; Loads the Accumulator with the Contents of Location 2000.
; i.e. A ← [2000]
STA 2000H ; Stores the Contents of the Accumulator at the Location 2000.
; i.e. [2000] ← A

Advantage:
The programmer **can identify** the address of the operand.

Disadvantage:
These are **three byte instructions** hence require three fetch cycles.

4) Indirect Addressing Mode

In this mode, the **Address** of the operand is specified **in Registers.**
Hence, the instruction indirectly points to the operands.
Even the Memory Pointer "M" can be used as it is pointed by the HL register pair.

Eg: **STAX B** ; Stores the contents of the Accumulator at the location
; pointed by the contents of BC pair.
; i.e. [[BC]] ← A.
; So if contents of BC pair = 4000 i.e. [BC] = 4000 then
; [4000] ← A. *#Please refer Bharat Sir's Lecture Notes for this ...*
INR M ; Increments the contents of the location pointed by HL pair
; (i.e. M) i.e. [[HL]] ← [[HL]] + 1

Advantage:
Address of the operand is **not fixed** and hence can be used in a **loop**.
Size of the instruction is **small** as compared to direct addressing mode.

Disadvantage:
Requires initialization of the register pair hence requires atleast one more instruction.

Special Notes:
Remember, during programming when you want to access only 1 or 2 locations, use Direct addressing mode as it is simpler.
But when you want to access a series of locations, use Indirect addressing mode.
Initialize the first address in a register pair.
Thereafter increment/decrement that pair in a loop to access a series of locations.

5) Implied Addressing Mode

In this mode, the **Operand** is **implied** in the instruction.
This instruction will work only on that implied operand, and not on any other operand.

Eg: **STC** ; Sets the Carry Flag in the Flag register.
; Cy ← 1.
CMC ; Complements the Carry Flag in the Flag register.

Advantage:
Instructions are generally **only one byte**.

Disadvantage:
Programmer **cannot** easily **identify** the value of the operand.