ADDRESSING MODES OF 8086 (ADDRESSING MODES OF IA 32 ARCHITECTURE)

Addressing modes is the manner in which operands are given in an instruction.
Most processors have various addressing modes, used in their instruction set.
The addressing modes of 8086 are as follows.

I IMMEDIATE ADDRESSING MODE

In this mode the **operand** is specified in the **instruction** itself.
Instructions are **longer** but the **operands** are **easily identified**.

```
Eg:   MOV CL, 12H        ; Moves 12 immediately into CL register
      MOV BX, 1234H      ; Moves 1234 immediately into BX register
```

II REGISTER ADDRESSING MODE

In this mode **operands** are specified using **registers**.
Instructions are **shorter** but **operands cant** be **identified** by looking at the instruction.

```
Eg:   MOV CL, DL         ; Moves data of DL register into CL register
      MOV AX, BX         ; Moves data of BX register into AX register
```

III DIRECT ADDRESSING MODE

In this mode **address** of the operand is directly specified **in the instruction**.

```
Eg:   MOV CL, [2000H]    ; CL Register gets data from memory location 2000H
                         ; CL 🡨 [2000H]
Eg:   MOV [3000H], DL    ; Memory location 3000H gets data from DL Register
                         ; [3000H] 🡨 DL
```

IV INDIRECT ADDRESSING MODE

In Indirect Addressing modes, **address is given by a register.**
The register can be incremented in a **loop** to access a **series of locations.**
There are various sub-types of Indirect addressing mode.

REGISTER INDIRECT ADDRESSING MODE

This is the most basic form of indirect addressing mode.
Here address is simply **given by a register**.

```
Eg:   MOV CL, [BX]       ; CL gets data from a memory location pointed by BX
                         ; CL 🡨 [BX]. If BX = 2000H, CL 🡨 [2000H]
Eg:   MOV [BX], CL       ; CL is stored at a memory location pointed by BX
                         ; [BX] 🡨 CL. If BX = 2000H, [2000H] 🡨 CL.
```

REGISTER RELATIVE ADDRESSING MODE

Here address is given by a **register plus a numeric displacement**.

Eg: MOV CL, [BX + 03H] *; CL gets data from a location BX + 03H*
; CL 🡐 [BX+03H]. If BX = 2000H, then CL 🡐 [2003H]

Eg: MOV [BX + 03H], CL *; CL is stored at location BX + 03H*
; [BX+03H] 🡐 CL. If BX = 2000H, then [2003H] 🡐 CL.

BASE INDEXED ADDRESSING MODE

Here address is given by **a sum of two registers**.
This is typically useful in accessing an array or a look up table.
One register acts as the base of the array holding its starting address and the **other acts as an index** indicating the element to be accessed.

Eg: MOV CL, [BX + SI] *; CL gets data from a location BX + SI*
; CL 🡐 [BX+SI].
; If BX = 2000H, SI = 1000H, then CL 🡐 [3000H]

Eg: MOV [BX + SI], CL *; CL is stored at location BX + SI*
; [BX+SI] 🡐 CL.
; If BX = 2000H, SI = 1000H, then [3000H] 🡐 CL.

BASE RELATIVE PLUS INDEX ADDRESSING MODE

Here address is given by a sum of base register plus index register plus a numeric displacement.

Eg: MOV CL, [BX+SI+03H] *; CL gets data from a location BX + SI + 03H*
; CL 🡐 [BX+SI+03H].
; If BX = 2000H, SI = 1000H, then CL 🡐 [3003H]

Eg: MOV [BX+SI+03H], CL *; CL is stored at location BX + SI + 03H*
; [BX+SI+03H] 🡐 CL.
; If BX = 2000H, SI = 1000H, then [3003H] 🡐 CL.

V IMPLIED ADDRESSING MODE

In this addressing mode, the operand is not specified at all, as it is an implied operand. Some instructions operate only on a particular register. In such cases, specifying the register becomes unnecessary as it becomes implied.

Eg: STC *; Sets the Carry flag.*
; This instruction can only operate on the Carry Flag.

Eg: CMC *; Complements the Carry flag.*
; This instruction can only operate on the Carry Flag.

ADDRESSING MODES OF 8085

Addressing modes is the manner in which operands are given in an instruction.
Most processors have various addressing modes, used in their instruction set.
The addressing modes of 8085 are as follows.

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 BC register pair. 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**.

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:
Instructions are of **one byte** so only one cycle is required to fetch them.

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

DIRECT ADDRESSING MODE

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

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

INDIRECT ADDRESSING MODE

In this mode, the **Address** of the operand is specified **in Registers.**
Hence, the instruction indirectly points to the operands.

E.g.:: **LDAX B** *; Loads "A" register with the contents of the memory Location pointed by BC Pair*
; So if BC pair = 4000 i.e. [BC] = 4000 then A ← [4000].
#Please refer Bharat Sir's Lecture Notes for this ...

STAX B *; Stores the contents of the Accumulator at the memory location pointed by BC pair.*
; So if contents of BC pair = 4000 i.e. [BC] = 4000 then [4000] ← A.
#Please refer Bharat Sir's Lecture Notes for this ...

Advantage:
Address of the operand is **not fixed** and hence can be used in a **loop**.

Disadvantage:
Requires initialization of the register pair hence more **complex**.

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:
Rigid, as it works only on a fixed operand.