ADDRESSING MODES OF 8086

ADDRESSING MODES

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing

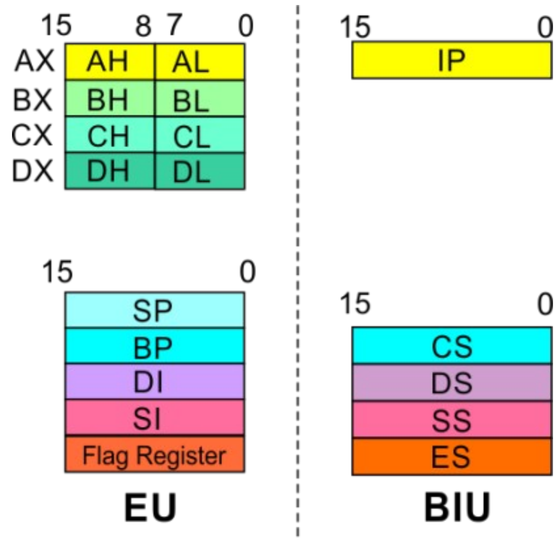
The instruction will specify the name of the register which holds the data to be operated by the instruction.

Example:

MOV CL, DH

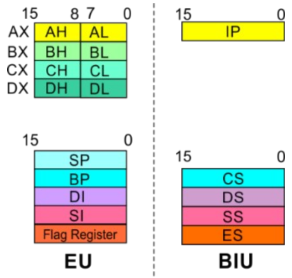
The content of 8-bit register DH is moved to another 8-bit register CL

(CL) ← (DH)



ADDRESSING MODES

1. Register Addressing
2. **Immediate Addressing**
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing



In immediate addressing mode, an 8-bit or 16-bit data is specified as part of the instruction

Example:

MOV DL, 08H

The 8-bit data (08_H) given in the instruction is moved to DL

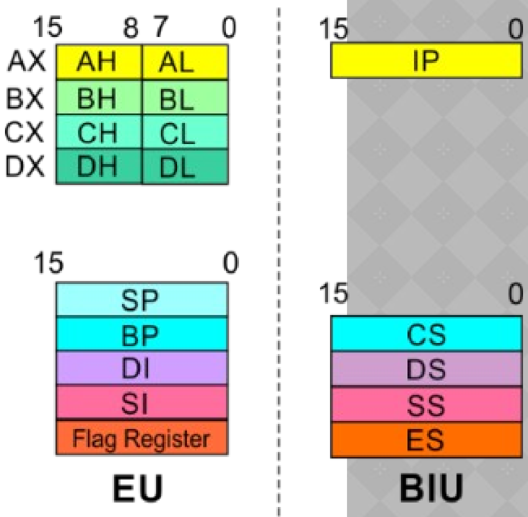
(DL) ← 08_H

MOV AX, 0A9FH

The 16-bit data ($0A9F_H$) given in the instruction is moved to AX register

(AX) ← $0A9F_H$

ADDRESSING MODES : MEMORY ACCESS



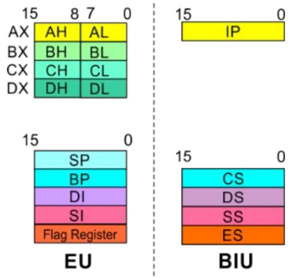
- **20 Address lines ⇒ 8086 can address up to 2^{20} = 1M bytes of memory**
- **However, the largest register is only 16 bits**
- **Physical Address will have to be calculated**
  Physical Address : Actual address of a byte in memory. i.e. the value which goes out onto the address bus.
- **Memory Address represented in the form – Seg : Offset (Eg - 89AB:F012)**
- **Each time the processor wants to access memory, it takes the contents of a segment register, shifts it one hexadecimal place to the left (same as multiplying by 16_{10}), then add the required offset to form the 20- bit address**

16 bytes of contiguous memory

89AB : F012 → 89AB → 89AB0 (Paragraph to byte → 89AB x 10 = 89AB0)
F012 → 0F012 (Offset is already in byte unit)
+ -------
98AC2 (The absolute address)

ADDRESSING MODES

1. Register Addressing
2. Immediate Addressing
3. **Direct Addressing**
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing



Here, the effective address of the memory location at which the data operand is stored is given in the instruction.

The effective address is just a 16-bit number written directly in the instruction.

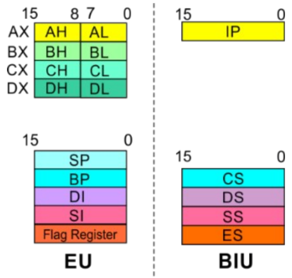
Example:

```
MOV   BX, [1354H]
MOV   BL,  [0400H]
```

The square brackets around the 1354_H denotes the contents of the memory location. When executed, this instruction will copy the contents of the memory location into BX register.

This addressing mode is called direct because the displacement of the operand from the segment base is specified directly in the instruction.

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. **Register Indirect Addressing**
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing



In Register indirect addressing, name of the register which holds the effective address (EA) will be specified in the instruction.

Registers used to hold EA are any of the following registers:

BX, BP, DI and SI.

Content of the DS register is used for base address calculation.

Example:

MOV CX, [BX]

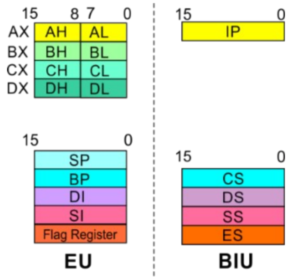
Note : Register/ memory enclosed in brackets refer to content of register/ memory

Operations:

EA = (BX)
BA = (DS) x 16_{10}
MA = BA + EA

(CX) ← (MA) or,

(CL) ← (MA)
(CH) ← (MA +1)



In Based Addressing, BX or BP is used to hold the base value for effective address and a signed 8-bit or unsigned 16-bit displacement will be specified in the instruction.

In case of 8-bit displacement, it is sign extended to 16-bit before adding to the base value.

When BX holds the base value of EA, 20-bit physical address is calculated from BX and DS.

When BP holds the base value of EA, BP and SS is used.

Example:

MOV AX, [BX + 08H]

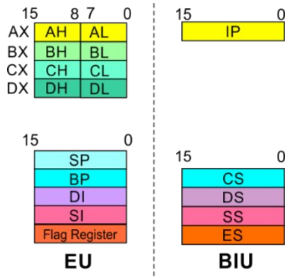
Operations:

$0008_H \leftarrow 08_H$ (Sign extended)
$EA = (BX) + 0008_H$
$BA = (DS) \times 16_{10}$
$MA = BA + EA$

$(AX) \leftarrow (MA)$ or,

$(AL) \leftarrow (MA)$
$(AH) \leftarrow (MA + 1)$

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. **Indexed Addressing**
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing



SI or DI register is used to hold an index value for memory data and a signed 8-bit or unsigned 16-bit displacement will be specified in the instruction.

Displacement is added to the index value in SI or DI register to obtain the EA.

In case of 8-bit displacement, it is sign extended to 16-bit before adding to the base value.

Example:

MOV CX, [SI + 0A2H]

Operations:

$FFA2_H \leftarrow A2_H$ (Sign extended)

$EA = (SI) + FFA2_H$

$BA = (DS) \times 16_{10}$

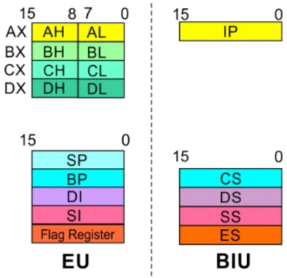
$MA = BA + EA$

$(CX) \leftarrow (MA)$ or,

$(CL) \leftarrow (MA)$

$(CH) \leftarrow (MA + 1)$

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing



In Based Index Addressing, the effective address is computed from the sum of a base register (BX or BP), an index register (SI or DI) and a displacement.

Example:

MOV DX, [BX + SI + 0AH]

Operations:

$000A_H \leftarrow 0A_H$ (Sign extended)

$EA = (BX) + (SI) + 000A_H$

$BA = (DS) \times 16_{10}$

$MA = BA + EA$

$(DX) \leftarrow (MA)$ or,

$(DL) \leftarrow (MA)$

$(DH) \leftarrow (MA + 1)$

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. **String Addressing**
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing

Employed in string operations to operate on string data.

The effective address (EA) of source data is stored in SI register and the EA of destination is stored in DI register.

Segment register for calculating base address of source data is DS and that of the destination data is ES

Example: MOVS BYTE

Operations:

Calculation of source memory location:

$EA = (SI)$ $BA = (DS) \times 16_{10}$ $MA = BA + EA$

Calculation of destination memory location:

$EA_E = (DI)$ $BA_E = (ES) \times 16_{10}$ $MA_E = BA_E + EA_E$

Note : Effective address of the Extra segment register

$(MAE) \leftarrow (MA)$

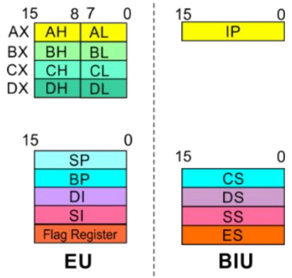
If $DF = 1$, then $(SI) \leftarrow (SI) - 1$ and $(DI) = (DI) - 1$

If $DF = 0$, then $(SI) \leftarrow (SI) + 1$ and $(DI) = (DI) + 1$

ADDRESSING MODES

Group III : Addressing modes for I/O ports

1. **Register Addressing**
2. **Immediate Addressing**
3. **Direct Addressing**
4. **Register Indirect Addressing**
5. **Based Addressing**
6. **Indexed Addressing**
7. **Based Index Addressing**
8. **String Addressing**
9. **Direct I/O port Addressing**
10. **Indirect I/O port Addressing**
11. **Relative Addressing**
12. **Implied Addressing**



These addressing modes are used to access data from standard I/O mapped devices or ports.

In direct port addressing mode, an 8-bit port address is directly specified in the instruction.

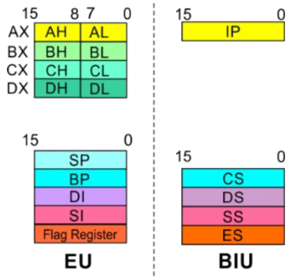
Example: IN AL, [09H]

Operations: $PORT_{addr} = 09_H$

$(AL) \leftarrow (PORT)$

Content of port with address 09_H is moved to AL register

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. **Relative Addressing**
12. Implied Addressing



In this addressing mode, the effective address of a program instruction is specified relative to Instruction Pointer (IP) by an 8-bit signed displacement.

Example: **JZ 0AH**

Operations:

$000A_H \leftarrow 0A_H$ (sign extend)

If ZF = 1, then

$EA = (IP) + 000A_H$

$BA = (CS) \times 16_{10}$

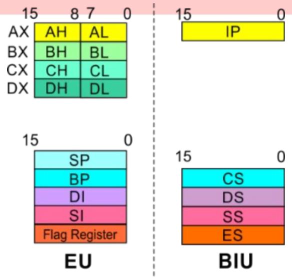
$MA = BA + EA$

If ZF = 1, then the program control jumps to new address calculated above.

If ZF = 0, then next instruction of the program is executed.

ADDRESSING MODES

1. Register Addressing
2. Immediate Addressing
3. Direct Addressing
4. Register Indirect Addressing
5. Based Addressing
6. Indexed Addressing
7. Based Index Addressing
8. String Addressing
9. Direct I/O port Addressing
10. Indirect I/O port Addressing
11. Relative Addressing
12. Implied Addressing



Instructions using this mode have no operands. The instruction itself will specify the data to be operated by the instruction.

Example: CLC

This clears the carry flag to zero.

THANK YOU