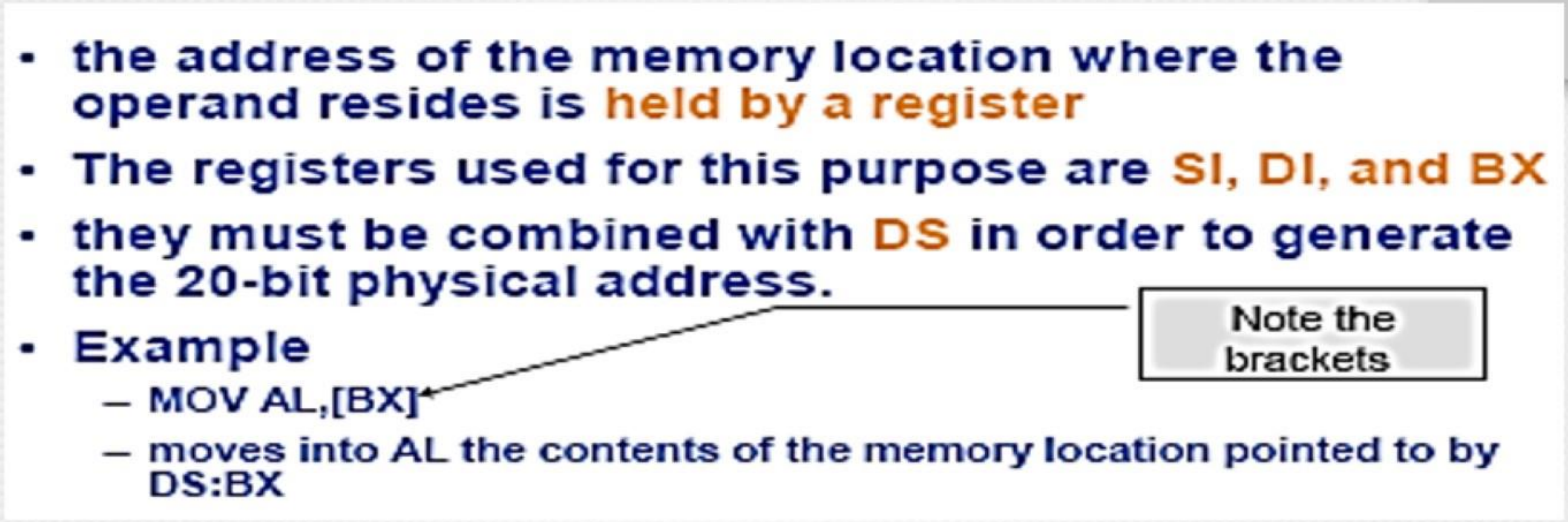
4. Register indirect addressing mode



- the address of the memory location where the operand resides is **held by a register**
- The registers used for this purpose are **SI, DI, and BX**
- they must be combined with **DS** in order to generate the 20-bit physical address.
- Example
  - MOV AL,[BX]
  - moves into AL the contents of the memory location pointed to by DS:BX

Note the brackets

a. Based addressing Mode

PA=Segment Base: Indirect address(Base register)

$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} BX \\ BP \end{Bmatrix}$$

Another segment register can be referenced by using a segment override prefix.

4. Register indirect addressing mode

b. Index addressing Mode

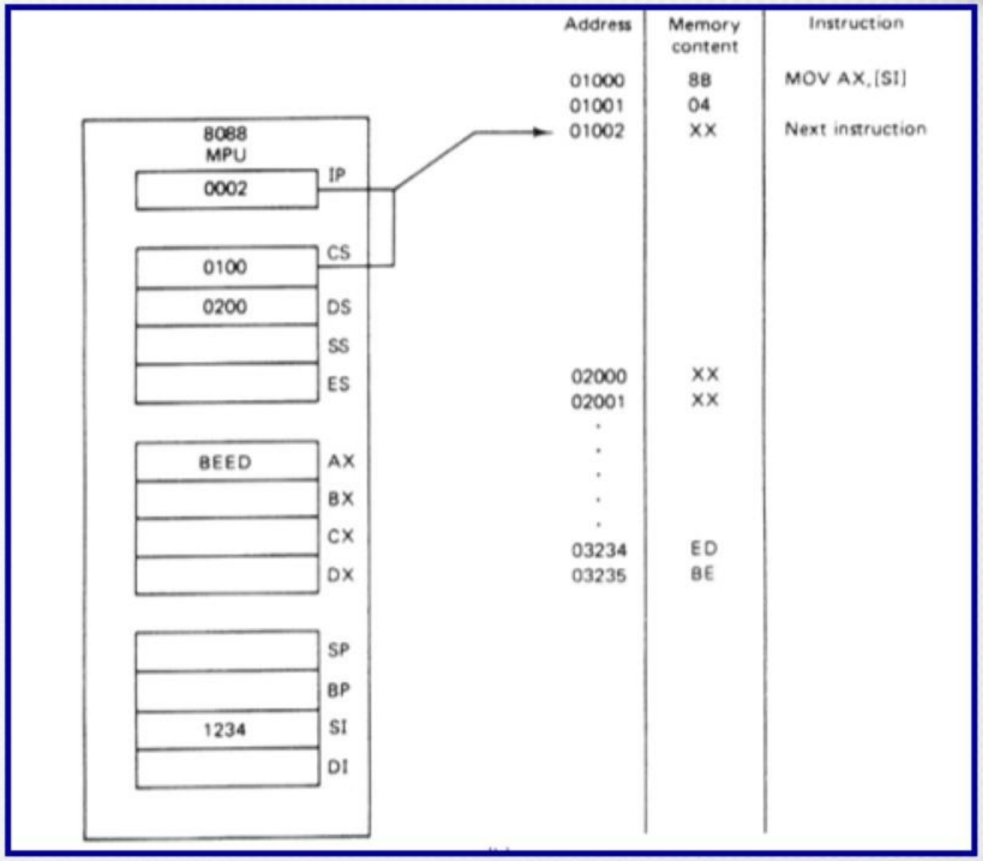
PA=Segment Base: Indirect address(Index register)

$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} SI \\ DI \end{Bmatrix}$$

4. Register indirect addressing mode

Example

MOV AX,[SI] ; Indirect index addressing mode.



4. Register indirect addressing mode

c. Base-Index addressing mod

$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} BX \\ BP \end{Bmatrix} + \begin{Bmatrix} SI \\ DI \end{Bmatrix}$$

- combining based and indexed addressing modes
- one base register and one index register are used.
- Examples:

4. Register indirect addressing mode

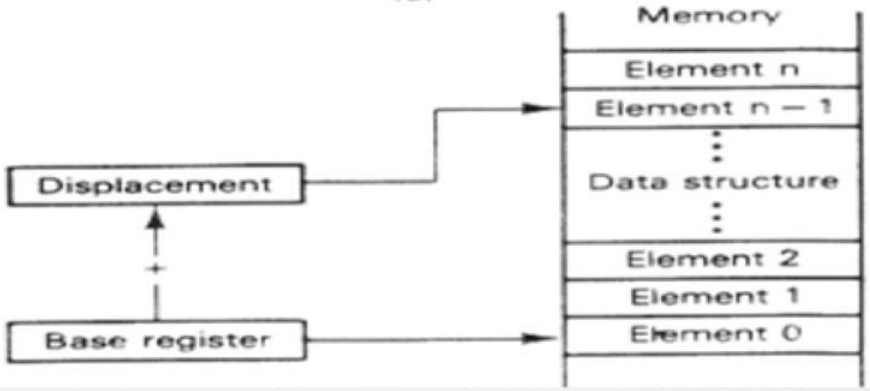
d. Base with Displacement Addressing Mode

In the ***based with Displacement*** *addressing mode*, the effective address of the operand is obtained by adding a direct or indirect displacement to the contents of either base register BX or base pointer register BP. The physical address calculation is shown in the next figure.

PA=segment base : Base + displacement

$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} BX \\ BP \end{Bmatrix} + \begin{Bmatrix} \text{8-bit displacement} \\ \text{16-bit displacement} \end{Bmatrix}$$

(a)

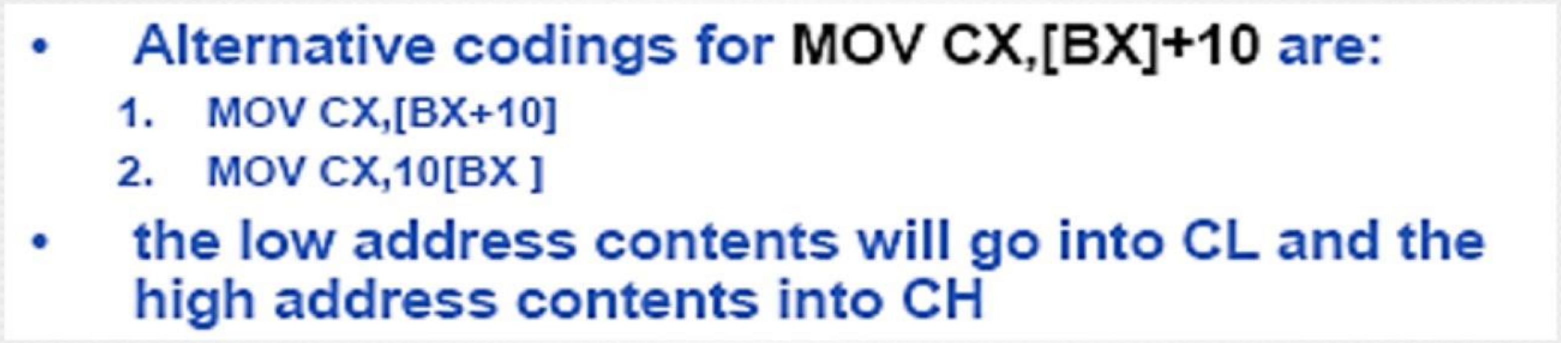


Addressing Mode

Example:

MOV [BX] +1234H, AL

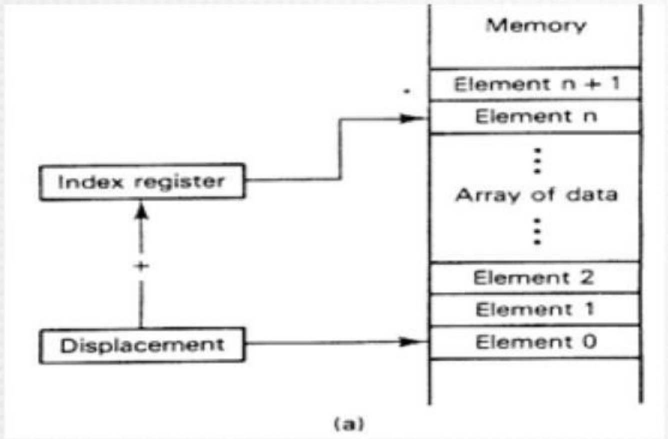
If BP is used instead of BX, the calculation of the physical address is performed using the contents of the stack segment (SS) instead DS. This permits access of data in the stack segment of memory.



- Alternative codings for **MOV CX,[BX]+10** are:
  1. MOV CX,[BX+10]
  2. MOV CX,10[BX]
- the low address contents will go into CL and the high address contents into CH

e. Indexed with displacement Addressing Mode

- works the same as the based relative addressing mode, except that registers DI and SI hold the offset address
- Examples:
  1. MOV DX, [SI]+5 ;PA = DS (shifted left) + SI + 5
  2. MOV CL, [DI]+20 ;PA = DS (shifted left) + DI + 20



(a)

PA = Segment base: Index + Displacement

$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} SI \\ DI \end{Bmatrix} + \begin{Bmatrix} \text{8-bit displacement} \\ \text{16-bit displacement} \end{Bmatrix}$$

(b)

Example:

- Assume that DS = 4500, SS = 2000, BX = 2100, SI = 1486, DI = 8500, BP= 7814, and AX = 2512. Show the exact physical memory location where AX is stored in each of the following. All values are in hex.

1. MOV [BX]+20, AX
2. MOV [SI]+10, AX
3. MOV [DI]+4, AX
4. MOV [BP]+12, AX

- Solution: **In each case**

  PA = segment reg. (shifted left) + (offset reg.) + displacement

1. DS:BX+20 location 47120 = (12) and 47121 = (25)
2. DS:SI+10 location 46496 = (12) and 46497 = (25)
3. DS:DI+4 location 4D504 = (12) and 4D505 = (25)
4. DS:BP+12 location 27826 = (12) and 27827 = (25)

f. Base-Indexed with displacement Addressing Mode

$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} BX \\ BP \end{Bmatrix} + \begin{Bmatrix} SI \\ DI \end{Bmatrix} + \begin{Bmatrix} 8\_bitdisplacement \\ 16\_bitdisplacement \end{Bmatrix}$$

- Examples:
- MOV CL, [BX][DI] + 8
  - PA = DS (shifted left) + BX + DI + 8
- MOV CH, [BX][SI]+20
  - PA = DS (shifted left) + BX + SI + 20
- MOV AH, [BP][DI]+12
  - PA = SS (shifted left) + BP + DI + 12
- MOV AH, [BP][SI]+29
  - PA = SS (shifted left) + BP + SI + 29

5. Stack addressing mode

PUSH ,POP , PUSHF and POPF instructions.

Example:

```
PUSH    BX
POP     [SI]
```

6. String addressing mode

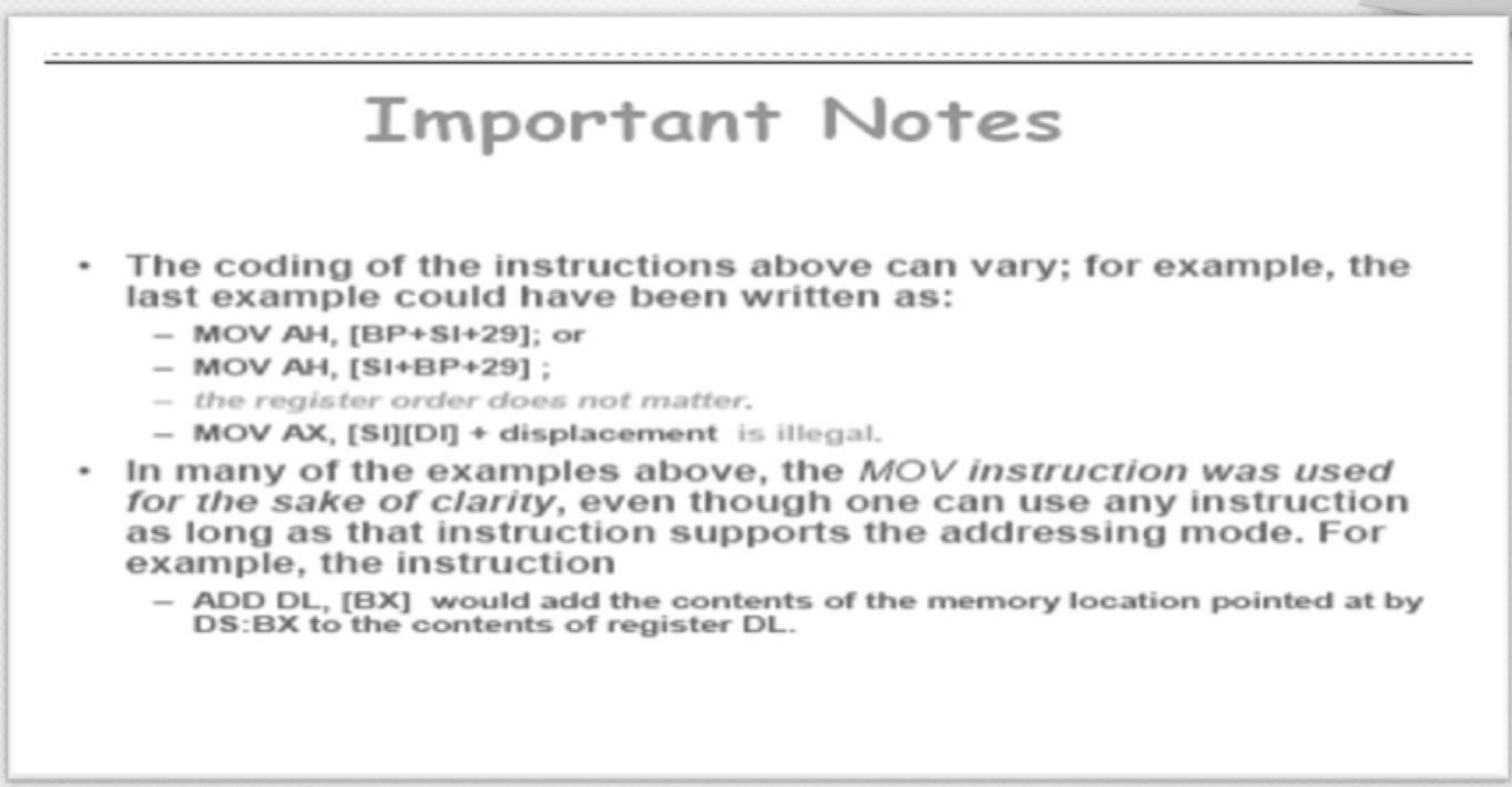
MOVS,LODS,STOS,CMPS and SCAS instructions.

7. input output addressing mode

IN , OUT instructions.

8. Implied addressing mode

XLAT , AAA , DAA , AAD , AAM and DAS instructions.

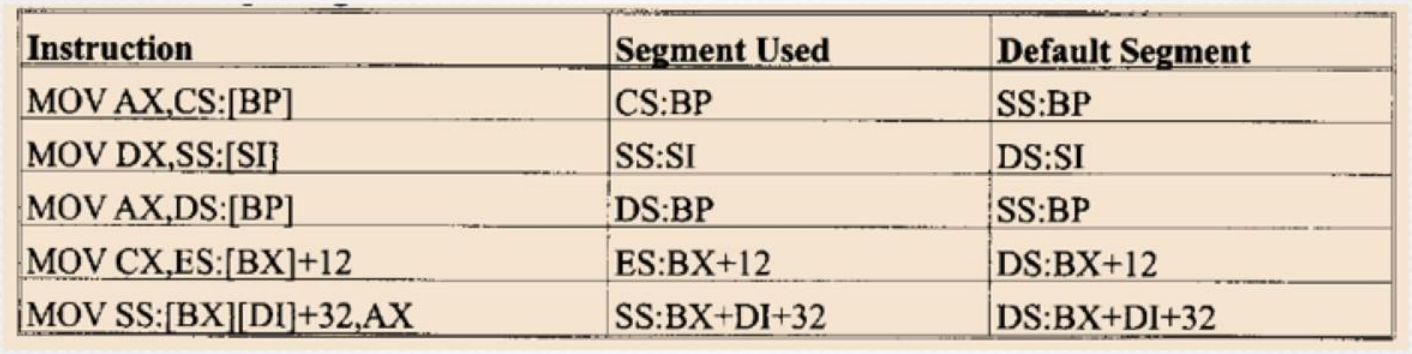


Important Notes

- The coding of the instructions above can vary; for example, the last example could have been written as:
  - MOV AH, [BP+SI+29]; or
  - MOV AH, [SI+BP+29] ;
  - *the register order does not matter.*
  - MOV AX, [SI][DI] + displacement is illegal.
- In many of the examples above, the *MOV instruction was used for the sake of clarity*, even though one can use any instruction as long as that instruction supports the addressing mode. For example, the instruction
  - ADD DL, [BX] would add the contents of the memory location pointed at by DS:BX to the contents of register DL.

Segment Overrides

- 80X86 allows the program to override the default segment registers
  - Specify the segment register in the code



Instruction	Segment Used	Default Segment
MOV AX,CS:[BP]	CS:BP	SS:BP
MOV DX,SS:[SI]	SS:SI	DS:SI
MOV AX,DS:[BP]	DS:BP	SS:BP
MOV CX,ES:[BX]+12	ES:BX+12	DS:BX+12
MOV SS:[BX][DI]+32,AX	SS:BX+DI+32	DS:BX+DI+32