

EEM332 Lecture3: Adressing modes in 8086

Addressing Modes:

The CPU can access **operands** (data) in various ways, called addressing modes. In 80x86 there are 7 addressing modes.

The number of addressing modes is determined when
the microprocessor is designed & cannot be changed

The x86 provides seven distinct addressing

- 1 - Register
- 2 - Immediate
- 3 - Direct
- 4 - Register indirect
- 5 - Based relative
- 6 - Indexed relative
- 7 - Based indexed relative

Register addressing mode involves use of registers to hold the data to be manipulated. Memory is not accessed, so it is relatively fast.

```
MOV  BX,DX ;copy the contents of DX into BX
MOV  ES,AX ;copy the contents of AX into ES
ADD  AL,BH ;add the contents of BH to contents of AL
```

Memory is not accessed, so it is relatively fast.

In *immediate addressing mode*, as the name implies, when the instruction is assembled, the operand comes immediately after the opcode.

The source operand is a constant.

2. Immediate addressing mode:

- source operand is a constant
- possible in all registers except segment and flag registers.

Ex:

```

MOV  BX,1234H      ; move 1234H into BX
MOV  CX,223        ; load the decimal value 625 into CX
ADD  AL,40H        ; AL=AL+33H

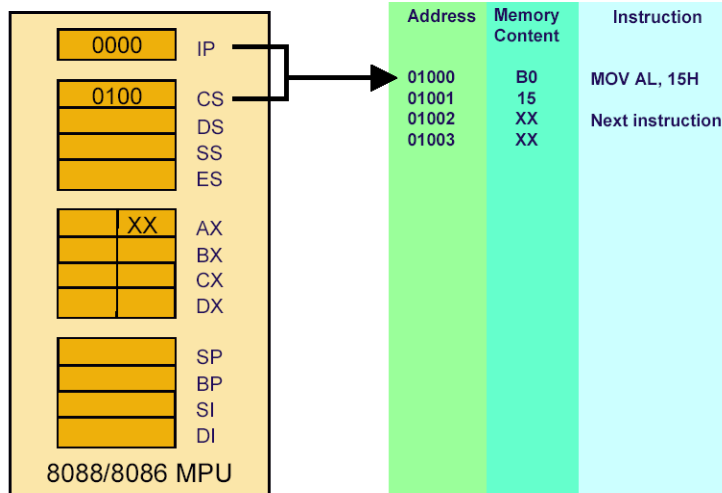
MOV  DS,1234H      ;illegal          (ask what to do)
```

```

MOV  BX,DX  ;copy the contents of DX into BX
MOV  ES,AX  ;copy the contents of AX into ES
ADD  AL,BH  ;add the contents of BH to contents of AL
```

Immediate Addressing

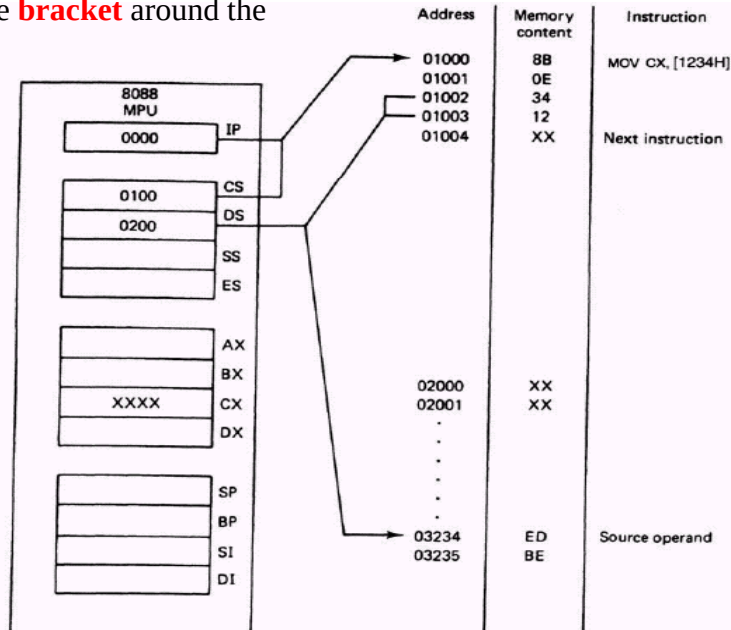
- In immediate addressing, source operand is a constant
- Used to load information to any one of the registers except for the segment registers and the flag register
 - MOV AX,2550h
 - MOV CX, 625 ; decimal 625
 - MOV BL,40h



Direct Addressing Mode

MOV CX, [1234h]

Note the **bracket** around the address



In register indirect addressing mode,

the address of the memory location where the operand resides is held by a register.

```
MOV AL,[ BX] ;moves into AL the contents of the memory
              ;location pointed to by DS:BX.
```

Notice that **BX** is in **brackets**.

The physical address is calculated by shifting DS left one hex position and adding BX to it

Example 1-16

Assume that DS = 1120, SI = 2498, and AX = 17FE. Show the contents of memory locations after the execution of "MOV [SI] ,AX".

Solution:

The contents of AX are moved into memory locations with logical address DS:SI and DS:SI + 1; therefore, the physical address starts at DS (shifted left) + SI = 13698. According to the little endian convention, low address 13698H contains FE, the low byte, and high address 13699H will contain 17, the high byte.

Example 1-16

Assume that DS = 1120, SI = 2498, and AX = 17FE. Show the contents of memory locations after the execution of "MOV [SI], AX".

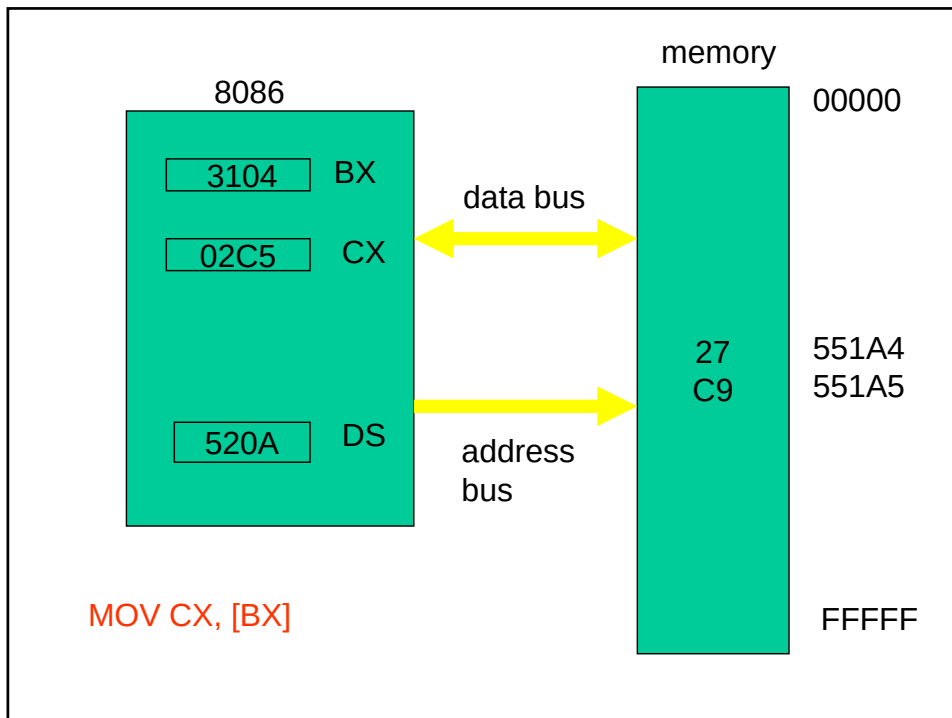
Solution:

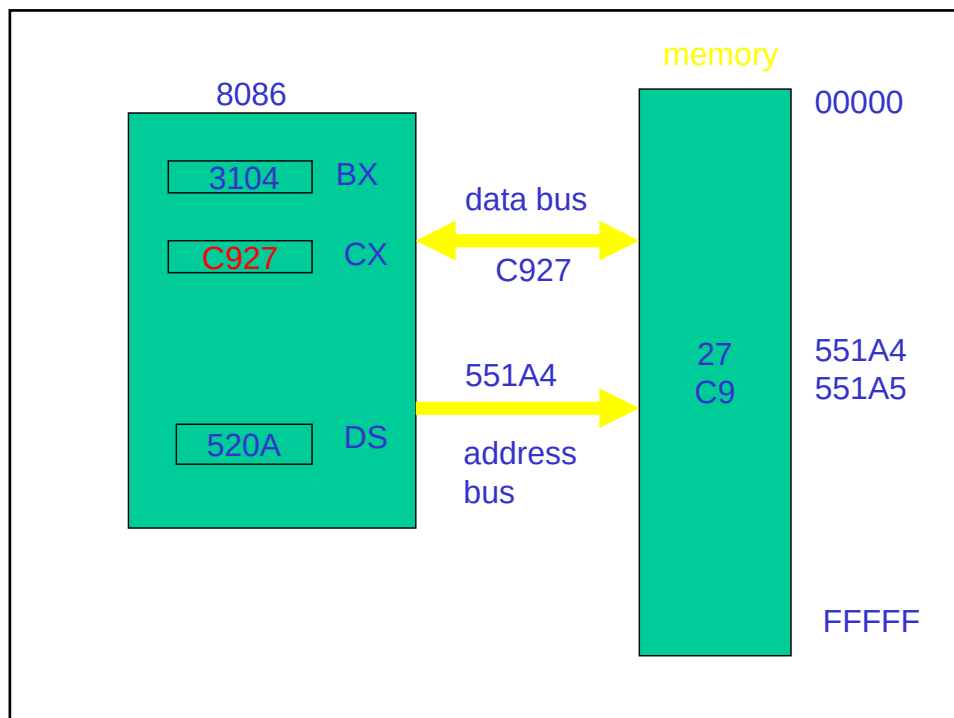
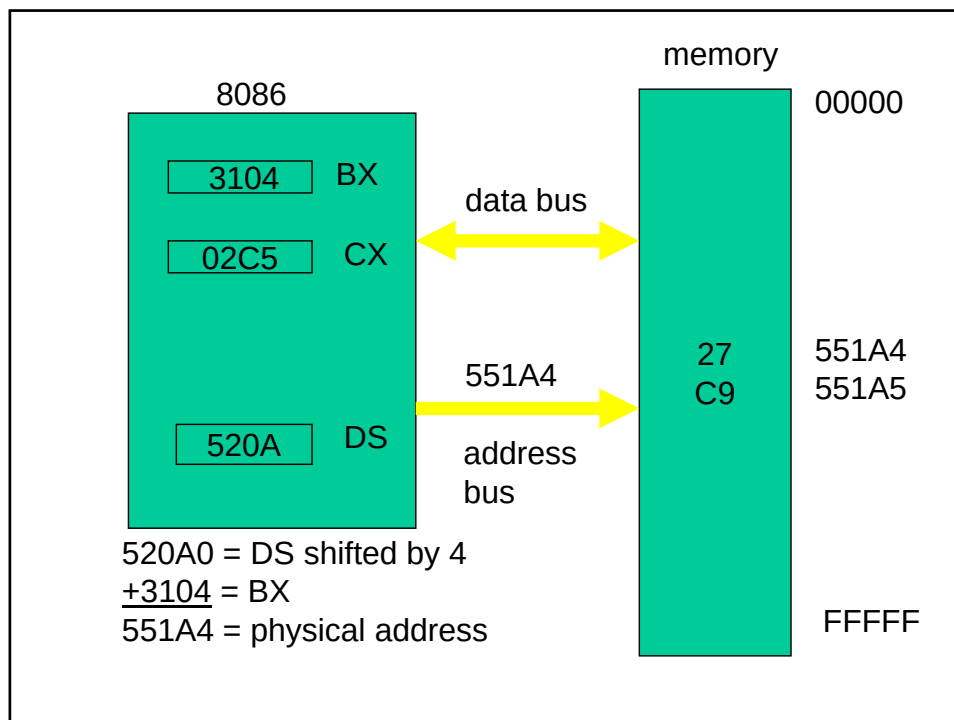
The contents of AX are moved into memory locations with logical address DS:SI and DS:SI + 1; therefore, the physical address starts at DS (shifted left) + SI = 13698. According to the little endian convention, low address 13698H contains FE, the low byte, and high address 13699H will contain 17, the high byte.

Example:

CX = destination — register
[BX] = source — memory location

1. Generate physical address for memory location from contents of BX and DS (shifted by 4)
— put on address bus
2. Retrieve contents of word in memory — put on data bus
3. Write data into CX





4. Register indirect addressing mode: Examples

- The address of the memory location where the operand resides is held by a register.
- SI, DI and BX registers are used as the pointers to hold the offset addresses.
- They must be combined with DS to generate the 20-bit physical address

MOV AL,[BX] ; moves into AL the contents of the memory location pointed to by DS:BX

MOV CL,[SI] ; move contents of DS:SI into CL

MOV [DI],AH ; move the contents of AH into DS:DI

BX contains 0400h

SI contains 0500h

DI contains 0600h

Offset 0400h contains ABCDh

Offset 0500h contains 1000h

Offset 0600h contains 0045h

INSTRUCTIONS:

1. **MOV AX,BX**

2. **MOV AX,[BX]**

3. **MOV AX,SI**

4. **MOV AX,[SI]**

RESULTS:

AX: 0400h

AX: ABCDh

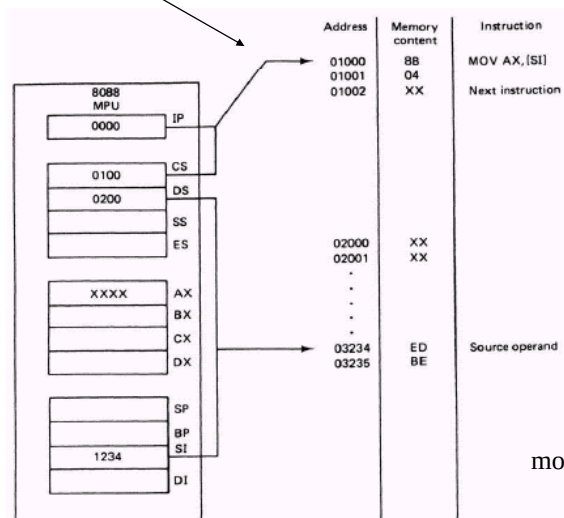
AX: 0500h

AX: 1000h

Direct versus indirect addressing mode:

address of the data in memory comes immediately after the instruction operand is a constant The address is the offset address. The offset address is put in a rectangular bracket **Ex: MOV AX,[SI]** or **SUB DX, [BX]**

indirect addressing



Direct addressing mode:

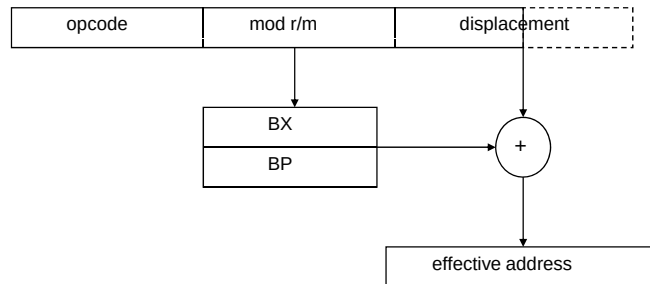
EX: MOV DL,[2400]

move contents of DS:2400H into DL

5. Based relative addressing mode:

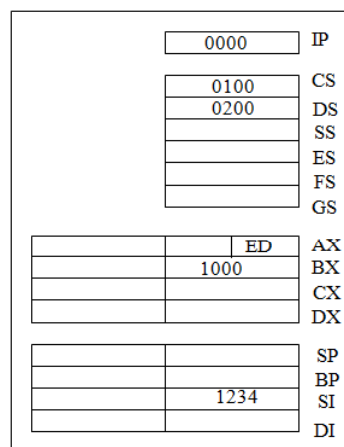
BX and BP are known as the base registers. In this mode base registers as well as a displacement value are used to calculate the **effective address**.

The default segments used for the calculation of **Physical address (PA)** are DS for BX, and SS for BP.



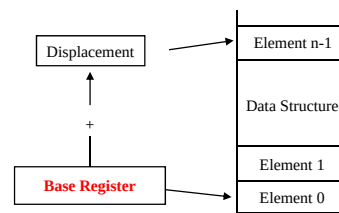
- **MOV CX,[BX]+10**
 - Physical Address : DS (shifted left) + BX + 10
- **MOV AL,[BP]+5**
 - Physical Address: SS (shifted left) + BP + 5

Example: MOV [BX]+1234H, AL



01000	88
01001	07
01002	34
01003	12
01004	XX

04234
04235

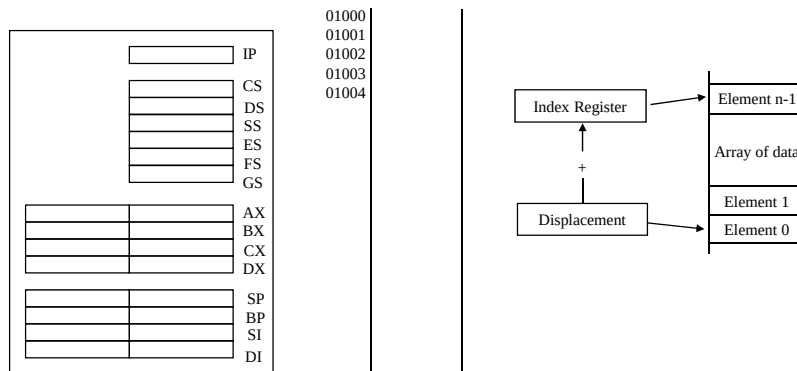


6. Indexed relative addressing mode:

Indexed relative addressing mode works the same as the based relative addressing mode. Except the registers DI and SI holds the offset address.

`MOV DX,[SI]+5` ;PA=DS(shifted left)+SI+5
`MOV CL,[DI]+20` ;PA=DS(shifted left)+DI+20

Example: `MOV AL, [SI]+2000H`



Example

Copy the contents of a block of memory (16 bytes) starting at location 20100h to another block of memory starting at 20120h

```
MOV AX,2000h
MOV DS,AX
MOV SI, 100h
MOV DI, 120h
MOV CX, 10h
NXTPT: MOV, AH, [SI]
        MOV [DI], AH
        INC SI
        INC DI
        DEC CX
        JNZ NXTPT
```

Adresing modes

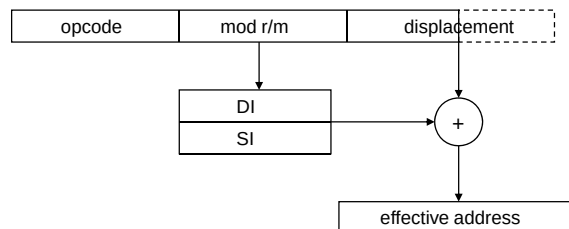
7. **Based Indexed addressing mode:**

The combination of the based and indexed addressing modes.
One base register and one index register are used.

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 AL,[BP][SI]+29 ;PA=SS(shifted left)+BP+SI+29

Alternative coding

MOV CL,[BX+DI+8]
MOV CL,[DI+BX+8]



Examples: Based and Index Addressing Modes

INSTRUCTION

SUB CX,[DI+4]

MOV AH,[4+DI]

CMP AX,[BP]+4

CMP AX,4+[BP]

ADD CX,NUMBERS[SI]

MEANING

Subtract from CX the word located in the data segment at an offset designated by the contents of DI + 4

Move to AH the byte located in the data segment at an offset designated by the contents of 4 + DI

Compare the contents of AX with the word on the STACK that is located at SS + the contents of BP + 4

Compare the contents of AX with the word on the STACK that is located at SS + 4 + the contents of BP

Add to CX the word in memory located in the array named NUMBERS that is offset from the beginning of the array by the value in SI

Accessing the STACK

- When **BP** is used in register indirect mode, it is understood that the segment address is in **SS**
- Thus **BP** can be used to access the **STACK** without changing the **STACK** or the Stack Pointer (**SP**)
- Example:

```
MOV    BP,SP           ; Set BP to the value in the stack pointer
MOV    AX,[BP]          ; move the top word on the stack to AX
MOV    BX,[BP+2]        ; move the second word from the top to BX
MOV    CX,[BP+4]        ; move the third word from the top to CX
```

Moves the 1st (top three) values in the stack to AX, BX, and CX respectively without changing **SP**

Example 1-17

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

- (a) `MOV[ BX] +20, AX` (b) `MOV[ SI] +10, AX`
(c) `MOV[ DI] +4, AX` (d) `MOV[ BP] +12, AX`

Solution:

In each case PA = segment register (shifted left) + offset register + displacement.

- (a) DS:BX+20 location 47120 = (12) and 47121 = (25)
(b) DS:SI+10 location 46496 = (12) and 46497 = (25)
(c) DS:DI+4 location 4D504 = (12) and 4D505 = (25)
(d) SS:BP+12 location 27826 = (12) and 27827 = (25)

Summary of the addressing modes

Addressing Mode	Operand	Default Segment
Register	Reg	None
Immediate	Data	None
Direct	[offset]	DS
Register Indirect	[BX] [SI] [DI]	DS DS DS
Based Relative	[BX]+disp [BP]+disp	DS SS
Indexed Relative	[DI]+disp [SI]+disp	DS DS
Based Indexed Relative	[BX][SI or DI]+disp [BP][SI or DI]+disp	DS SS