

Addressing Modes:

Immediate operand addressing mode

Register operand addressing mode

Memory operand addressing mode

Direct addressing mode

Register indirect addressing mode

Based addressing mode

Indexed addressing mode

Based-indexed addressing mode

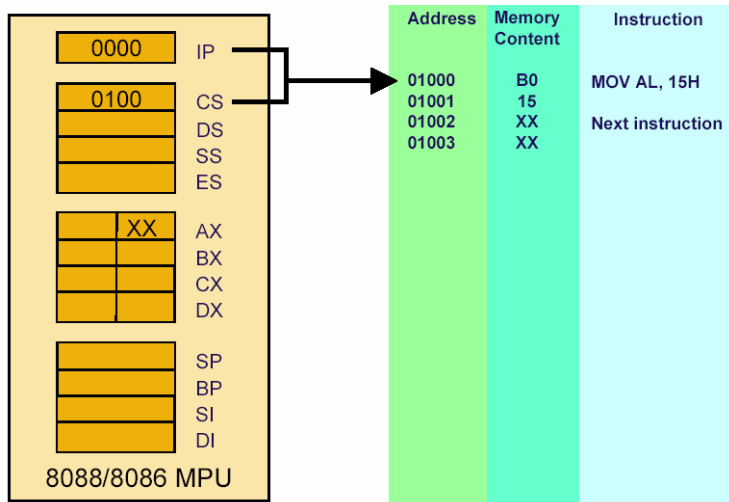
$$PA = \begin{matrix} \boxed{\text{CS}} \\ \boxed{\text{SS}} \\ \boxed{\text{DS}} \\ \boxed{\text{ES}} \end{matrix} : \begin{matrix} \boxed{\text{BX}} \\ \boxed{\text{BP}} \end{matrix} + \begin{matrix} \boxed{\text{SI}} \\ \boxed{\text{DI}} \end{matrix} + \begin{matrix} \boxed{\text{8 bit displacement}} \\ \boxed{\text{16 bit displacement}} \end{matrix}$$

The default segment register is DS

9

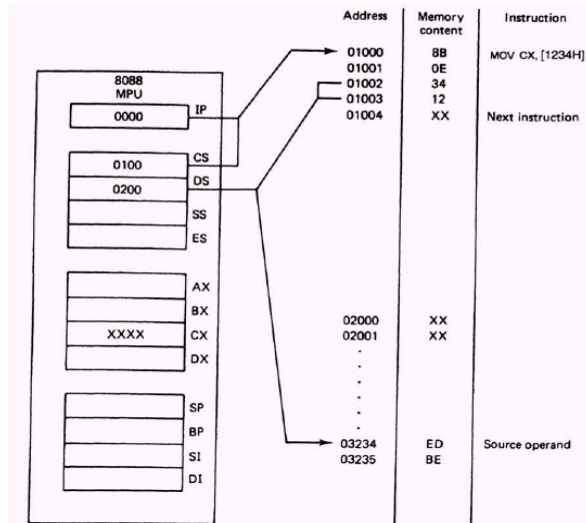
Immediate Addressing

- 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



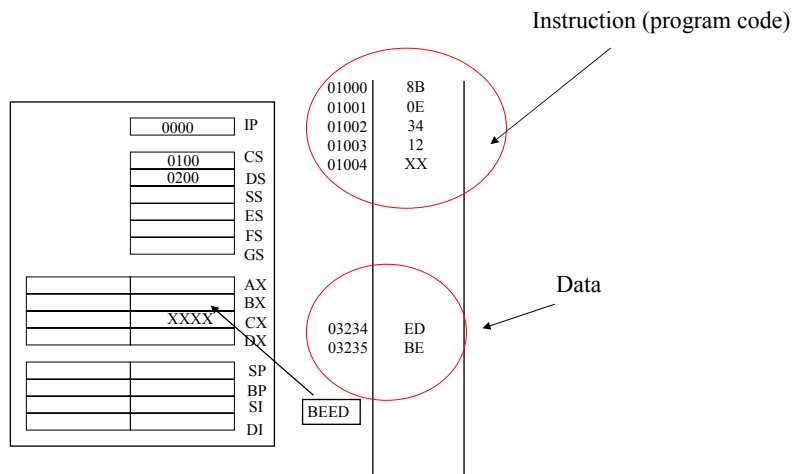
Direct Addressing Mode

MOV CX, [1234h]



– Direct Addressing Mode

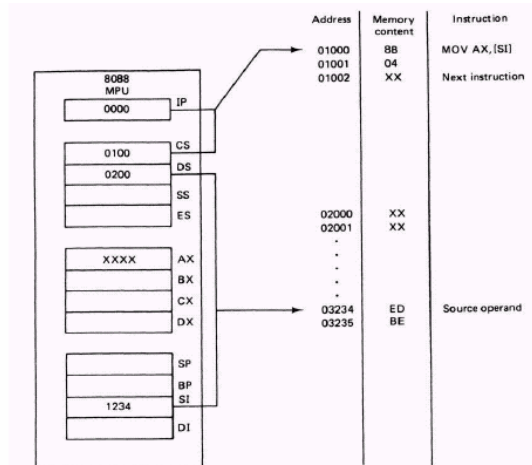
- PA = Segment Base : Direct Address
- MOV CX, [1234H]



Register Indirect Addressing Mode

MOV AX, [SI]

- Effective address is found in either the BX, BP, SI, DI register
- SUB DX, [BX]



Examples

ASSUMPTION: ASSUME all specified offsets are in the DATA Segment:

BX contains 0400h	Offset 0400h contains ABCDh
SI contains 0500h	Offset 0500h contains 1000h
DI contains 0600h	Offset 0600h contains 0045h

INSTRUCTIONS:

RESULTS:

1. MOV AX, BX	AX: 0400h
2. MOV AX, [BX]	AX: ABCDh
3. MOV AX, SI	AX: 0500h
4. MOV AX, [SI]	AX: 1000h

```

mov  al,    [bp]    ;al gets 8 bits at SS:BP

mov  ah,    [bx]    ;ah gets 8 bits at DS:BX

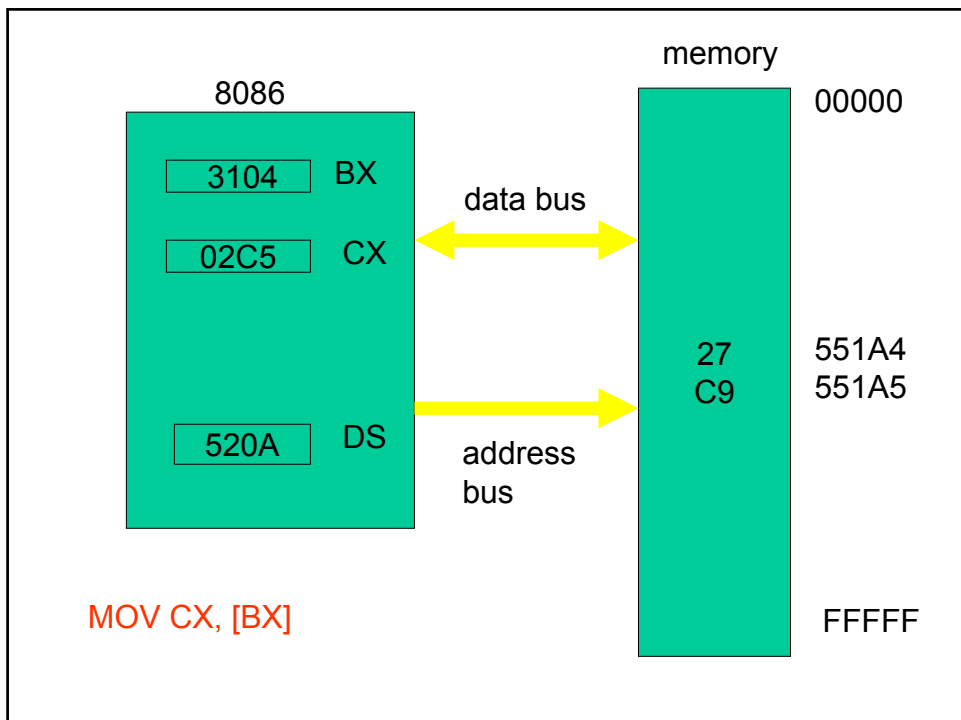
mov  ax,    [di]    ;ax gets 16 bits at DS:SI

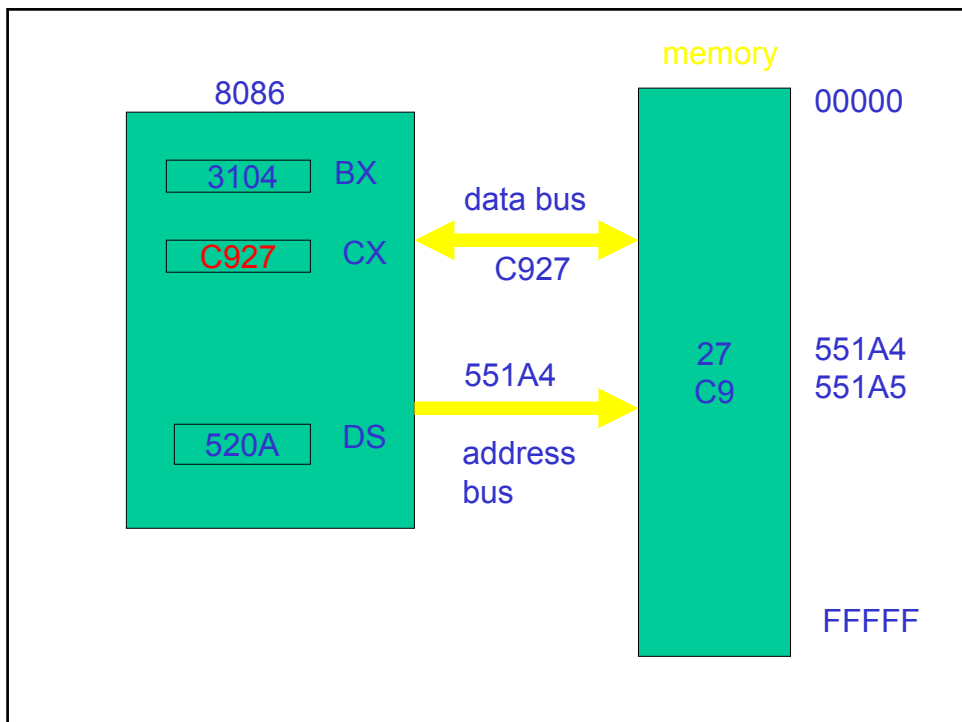
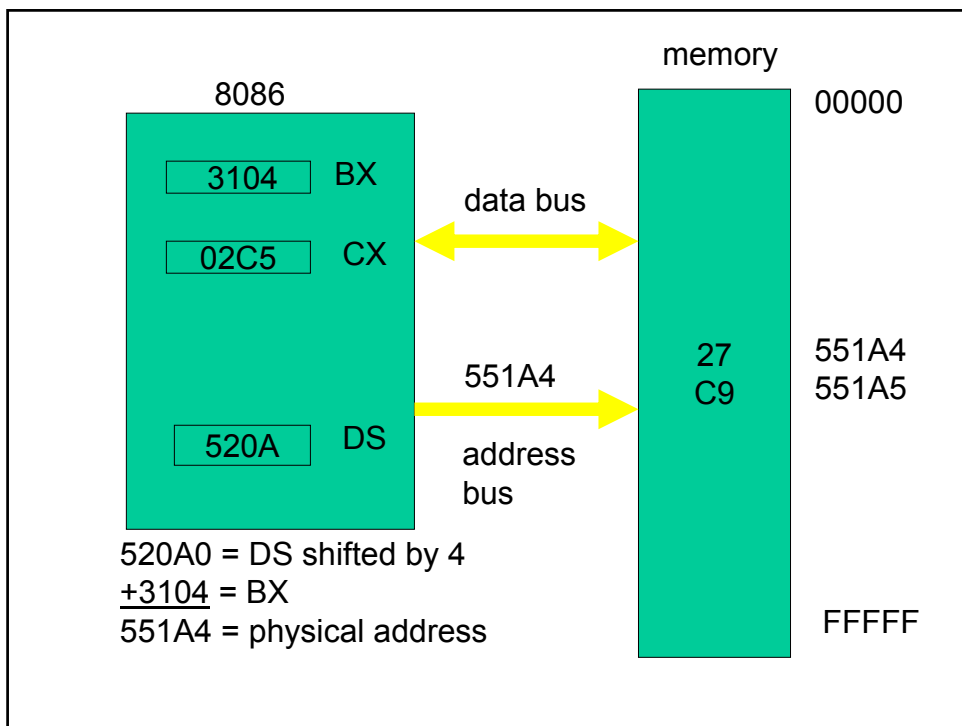
mov  eax,   [si]    ;eax gets 32 bits at DS:SI
    
```

MOV CX, [BX]

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

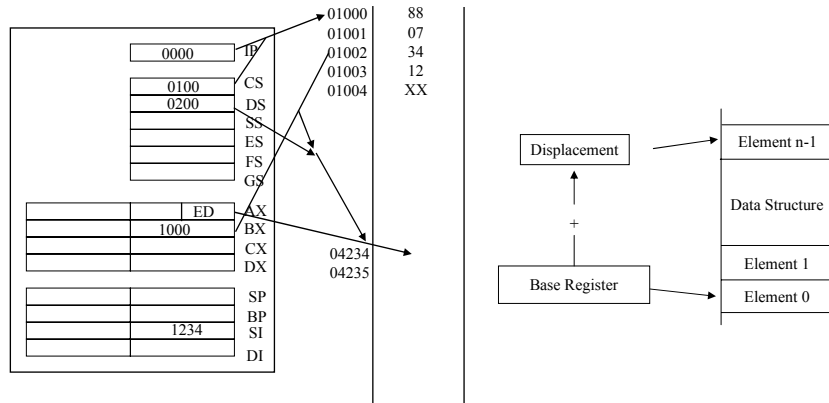
1. Generate physical address for memory location from contents of BX and DS (shifted by 4)
— put on address bus
2. Retrieve contents of word in memory — put on data bus
3. Write data into CX





– Based Addressing Mode

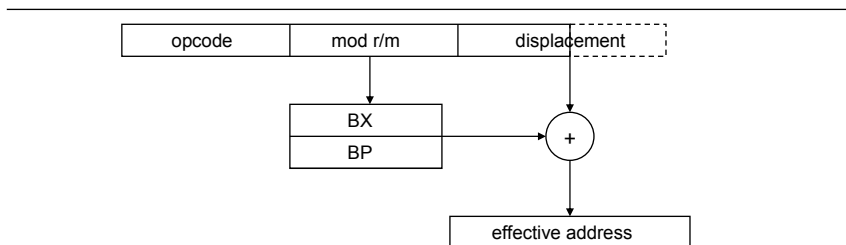
- $PA = \text{Segment Base} : \{BX \text{ or } BP\} + \{8\text{-bit or } 16\text{-bit displacement}\}$
- Base register : the beginning of a data structure
- See Fig 3.16 (b) in page 74
- Example: `MOV [BX]+1234H, AL`



Based Addressing

```

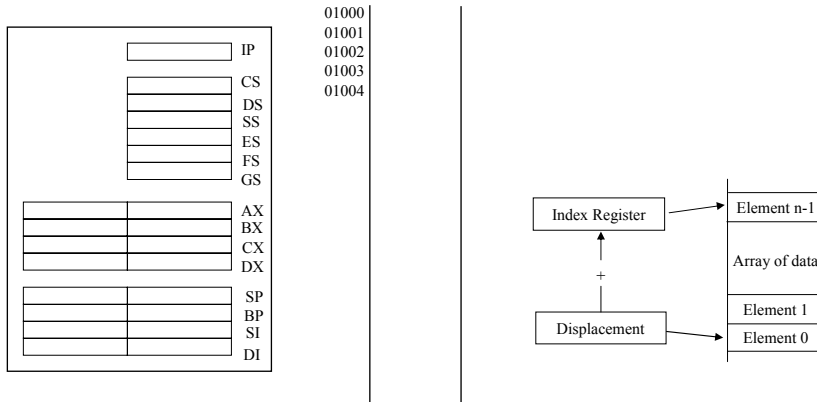
mov  al,  [bp+2]      ;al gets 8 bits at SS:BP+2
mov  ah,  [bx-4]      ;ah gets 8 bits at DS:BX-4
    
```



- `MOV CX,[BX]+10`
 - Physical Address : $DS \text{ (shifted left)} + BX + 10$
- `MOV AL,[BP]+5`
 - Physical Address: $SS \text{ (shifted left)} + BP + 5$

– Indexed Addressing Mode

- PA = Segment Base : {SI, DI} + {8-bit or 16-bit displacement}
- Displacement : the starting address of an array;
Index: selects the specific element in the array
- 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

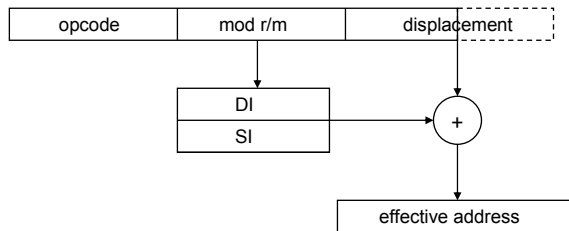
```

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

```

Indexed Indirect Addressing

```
mov    ax,    [si+1000h]    ;ax gets 16 bits at DS:SI+1000h
mov    eax,    [si+300h]    ;eax gets 32 bits at DS:SI+300h
Mov     [di+100h],    al    ;DS:DI+100h gets 8 bits in al
```



– Based-Indexed Addressing Mode

- PA= Seg Base: {BX, BP}+{SI,DI}+{8-bit or 16-bit displacement}
- to access complex data structures
- See fig 3.20 in page 80
- Example: **MOV AH, [BX][SI]+1234H**
opcode : 8A 44 34 12

MOV BX, 0600h

(Based relative addressing mode)

MOV SI, 0010h ;4 x 4 = 16

(Indexed relative addressing mode)

MOV AL, [BX + SI + 3]

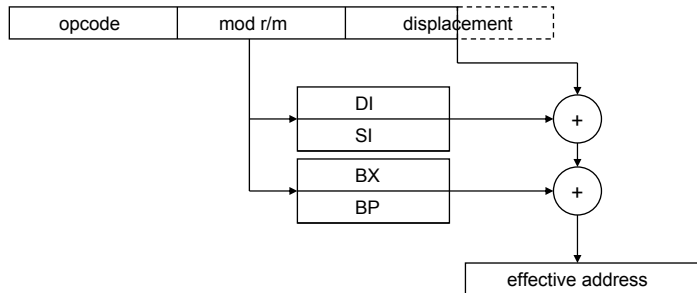
Based-indexed relative addressing mode)

Based Indexed Indirect Addressing

```

mov    ax,    [bp+di]      ;ax gets 16 bits at SS:BP+DI
mov    ax,    [di+bp]      ;ax gets 16 bits at DS:BP+DI
mov    eax,   [bx+si+10h]   ;eax gets 32 bits at DS:BX+SI+10h
mov    cx,    [bp+si-7]    ;cx gets 16 bits at SS:BP+SI-7

```



Addressing Mode Examples

mov al, bl	;8-bit register addressing	Register
mov di, bp	;16-bit register addressing	
mov eax, eax	;32-bit register addressing	
mov al, 12	;8-bit immediate, al<-0ch	Immediate
mov cx, faceh	;16-bit immediate, cx<-64,206	
mov ebx, 2h	;32-bit immediate, ebx<-00000002h	
mov al, LIST	;al<-8 bits stored at label LIST	Direct
mov ch, DATA	;ch<-8 bits stored at label DATA	
mov ds, DATA2	;ds<-16 bits stored at label DATA2	
mov al, [bp]	;al<-8 bits stored at SS:BP	Based
mov ah, [bx]	;ah<-8 bits stored at DS:BX	
mov ax, [bp]	;ax<-16 bits stored at SS:BP	
mov eax, [bx]	;eax<-32 bits stored at DS:BX	
mov al, [bp+2]	;al<-8 bits stored at SS: (BP+2)	
mov ax, [bx-4]	;ax<-16 bits stored at DS: (BX-4)	
mov al, LIST[bp]	;al<-8 bits stored at SS: (BP+LIST)	
mov bx, LIST[bx]	;bx<-16 bits stored at DS: (BX+LIST)	
mov al, LIST[bp+2]	;al<-8 bits stored at SS: (BP+2+LIST)	
mov ax, LIST[bx-12h]	;ax<-16 bits stored at DS: (BX-12+LIST)	

More Addressing Mode Examples

mov al, [si]	;al<-8 bits stored at DS:SI	
mov ah, [di]	;ah<-8 bits stored at DS:DI	
mov ax, [si]	;ax<-16 bits stored at DS:SI	
mov eax, [di]	;eax<-32 bits stored at DS:DI	
mov ax, es:[di]	;ax<-16 bits stored at ES:DI	Indexed
mov al, [si+2]	;al<-8 bits stored at DS:(SI+2)	
mov ax, [di-4]	;ax<-16 bits stored at DS:(DI-4)	
mov al, LIST[si]	;al<-8 bits stored at DS:(SI+LIST)	
mov bx, LIST[di]	;bx<-16 bits stored at DS:(DI+LIST)	
mov al, LIST[si+2]	;al<-8 bits stored at DS:(SI+2+LIST)	
mov ax, LIST[di-12h]	;ax<-16 bits stored at DS:(DI-12+LIST)	
mov al, [bp+di]	;al<-8 bits from SS:(BP+DI)	
mov ah, ds:[bp+si]	;ah<-8 bits from DS:(BP+SI)	
mov ax, [bx+si]	;ax<-16 bits from DS:(BX+SI)	
mov eax, es:[bx+di]	;eax<-32 bits from ES:(BX+DI)	Based Indexed
mov al, LIST[bp+di]	;al<-8 bits from SS:(BP+DI+LIST)	
mov ax, LIST[bx+si]	;ax<-16 bits from DS:(BX+SI+LIST)	
mov al, LIST[bp+di-10h]	;al<-8 bits from SS:(BP+DI-16+LIST)	
mov ax, LIST[bx+si+1AFH]	;ax<-16 bits from DS:(BX+SI+431+LIST)	

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**