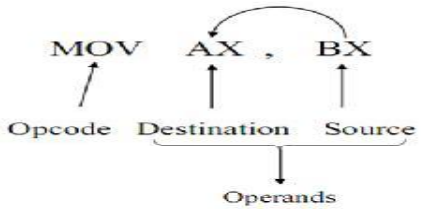
8086 Addressing Mode

Introduction

- Program is a sequence of commands used to tell a microcomputer what to do.
- Each command in a program is an instruction
- Programs must always be coded in machine language before they can be executed by the microprocessor.
- A program written in machine language is often referred to as *machine code*.
- Machine code is encoded using **0**s and **1**s
- A single machine language instruction can take up one or more bytes of code
- In assembly language, each instruction is described with alphanumeric symbols instead of with **0**s and **1**s
- Instruction can be divided into two parts : its *opcode* and *operands*
- Op-code identify the operation that is to be performed.
- Each opcode is assigned a unique letter combination called a *mnemonic*.
- Operands describe the data that are to be processed as the microprocessor carried out, the operation specified by the opcode.
- For example, the move instruction is one of the instructions in the data transfer group of the 8086 instruction set.
- Execution of this instruction transfers a byte or a word of data from a source location to a destination location.



Addressing Mode of 8086

An addressing mode is a method of specifying an operand. The 8086 addressing modes categorized into three types:

1. Register Addressing
2. Immediate Addressing
3. Memory Addressing

Register addressing mode

In this addressing mode, the operands may be:

- reg16: 16-bit general registers: AX, BX, CX, DX, SI, DI, SP or BP.
- reg8 : 8-bit general registers: AH, BH, CH, DH, AL, BL, CL, or DL.

- Sreg : segment registers: CS, DS, ES, or SS. There is an exception: CS cannot be a destination.

For register addressing modes, there is no need to compute the effective address. The operand is in a register and to get the operand there is no memory access involved.

Example: Register Operands		
MOV AX, BX	;	mov reg16, reg16
ADD AX, SI	;	add reg16, reg16
MOV DS, AX	;	mov Sreg, reg16

Some rules in register addressing modes:

1. You may not specify CS as the destination operand.

   Example: MOV CS, 02H –> wrong

2. Only one of the operands can be a segment register. You cannot move data from one segment register to another with a single MOV instruction. To copy the value of CS to DS, you would have to use some sequence like:

   MOV DS,CS -> wrong

   MOV AX, CS

   MOV DS, AX -> the way we do it

3. You should never use the segment registers as data registers to hold arbitrary values. They should only contain segment addresses.

Immediate Addressing Mode

In this addressing mode, the operand is stored as part of the instruction. The immediate operand, which is stored along with the instruction, resides in the code segment -- not in the data segment. This addressing mode is also faster to execute an instruction because the operand is read with the instruction from memory. Here are some examples:

Example: Immediate Operands	
MOV AL, 20	; Copies a 20 decimal into register AL
MOV BX,55H	; Copies a 0055H into register BX
MOV SI,0	; Copies a 0000H into register SI
MOV DX, ‘Ahmed’	; Copies an ASCII Ahmed into register DX
MOV CL, 10101001B	; Copies a 10101001 binary into register CL

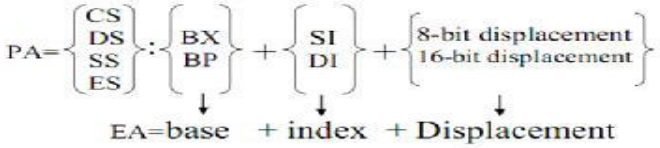
Memory Addressing Modes

To reference an operand in memory, the 8086 must calculate the physical address (PA) of the operand and then initiate a read or write operation of this storage location. The 8086 MPU is provided with a group of addressing modes known as the memory operand addressing modes

for this purpose. Physical address can computed from a segment base address (SBA) and an effective address (EA). SBA identifies the starting location of the segment in memory, and EA represents the offset of the operand from the beginning of this segment of memory.

PA=SBA (segment): EA (offset)

PA=segment base: base + index + Displacement

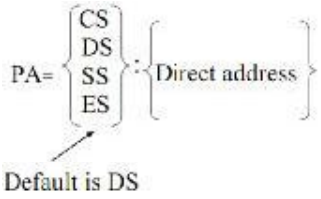


There are different forms of memory addressing modes

1. Direct Addressing
2. Register indirect addressing
3. Based addressing
4. Indexed addressing
5. Based indexed addressing
6. Based indexed with displacement

1. Direct Addressing Mode

Direct addressing mode is similar to immediate addressing in that information is encoded directly into the instruction. However, in this case, the instruction opcode is followed by an effective address, instead of the data. As shown below:



For example the instruction

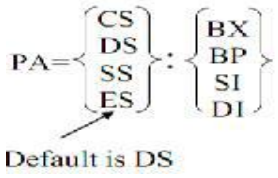
Example: Immediate Operands	
MOV AL, DS:[2000H] or MOV AL, [2000H]	; move the contents of the memory location with offset 2000 into register AL
MOV AL,DS:[8088H] or MOV AL,[8088H]	; move the contents of the memory location with offset 8088 into register AL
MOV DS:[1234H],DL or MOV [1234H],DL	; stores the value in the DL register to memory location with offset 1234H

By default, all displacement-only values provide offsets into the data segment. If you want to provide an offset into a different segment, you must use a segment override prefix before your address. For example, to access location 1234H in the extra segment (ES) you would use an instruction of the form MOV AX,ES:[1234H]. Likewise, to access this location in the code segment you would use the instruction MOV AX, CS:[1234H].

2. Register Indirect Addressing Mode

This mode is similar to the direct address except that the effective address held in any of the following register: BP, BX, SI, and DI. As shown below:

MOV AL, [BX]
MOV AL, [BP]
MOV AL, [SI]
MOV AL, [DI]



The [BX], [SI], and [DI] modes use the DS segment by default. The [BP] addressing mode uses the stack segment (SS) by default. You can use the segment override prefix symbols if you wish to access data in different segments. The following instructions demonstrate the use of these overrides:

MOV AL, CS:[BX]
MOV AL, DS:[BP]
MOV AL, SS:[SI]
MOV AL, ES:[DI]

For example:

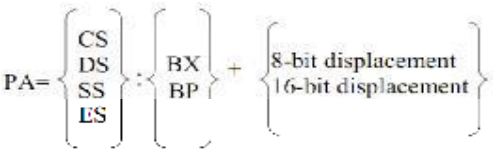
MOV SI, 1234H
MOV AL, [SI]

If SI contains 1234H and DS contains 0200H the result produced by executing the instruction is that the contents of the memory location at address:

PA = 02000H + 1234H
= 03234 are moved to the AX register.

3. Based Addressing Mode

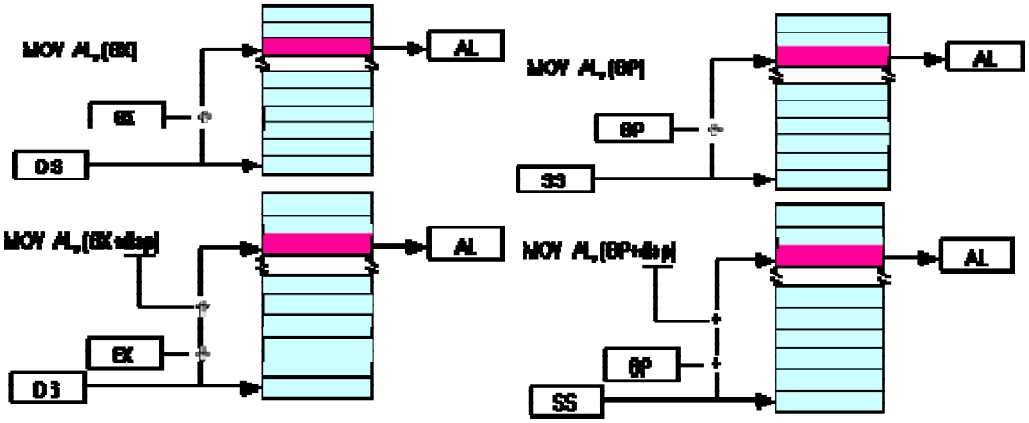
In the based addressing mode, the effective address of the operand is obtained by adding a direct or indirect displacement to the contents of either base register BX or base pointer register BP. The physical addresses calculate as shown:



For example if BX=1000, DS=0200, and AL=EDH, for the following instruction:
MOV [BX] + 1234H, AL

EA=BX+1234H = 1000H+1234H = 2234H
PH=DS*10+EA =0200H*10+2234H = 4234H

So it writes the contents of source operand AL (EDH) into the memory location 04234H. If BP is used instead of BX, the calculation of the physical address is performed using the contents of the stack segment (SS) register instead of DS. This permits access to data in the stack segment of memory.



4. Indexed Addressing Modes

In the Indexed addressing mode, the effective address of the operand is obtained by adding a direct or indirect displacement to the contents of either SI or DI register. The physical addresses calculate as shown below:

The indexed addressing modes use the following syntax:
MOV AL, [SI]
MOV AL, [DI]
MOV AL, [SI+DISP]
MOV AL, [DI+DISP]

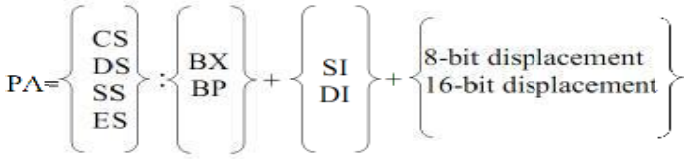
$$PA = \begin{Bmatrix} CS \\ DS \\ SS \\ ES \end{Bmatrix} : \begin{Bmatrix} SI \\ DI \end{Bmatrix} + \begin{Bmatrix} \text{8-bit displacement} \\ \text{16-bit displacement} \end{Bmatrix}$$

5. Based Indexed Addressing Modes

Combining the based addressing mode and the indexed addressing mode results in a new, more powerful mode known as based-indexed addressing mode. This addressing mode can be used to access complex data structures such as two-dimensional arrays. As shown below this mode can be used to access elements in an m X n array of data. Notice that the displacement, which is a fixed value, locates the array in memory. The base register specifies the m coordinate

of the array, and the index register identifies the n coordinate. Simply changing the values in the base and index registers permits access to any element in the array.

MOV AL, [BX+SI]
MOV AL, [BX+DI]
MOV AL, [BP+SI]
MOV AL, [BP+DI]



Suppose that BX contains 1000H and si contains 880H. Then the instruction

MOV AL,[BX][SI]

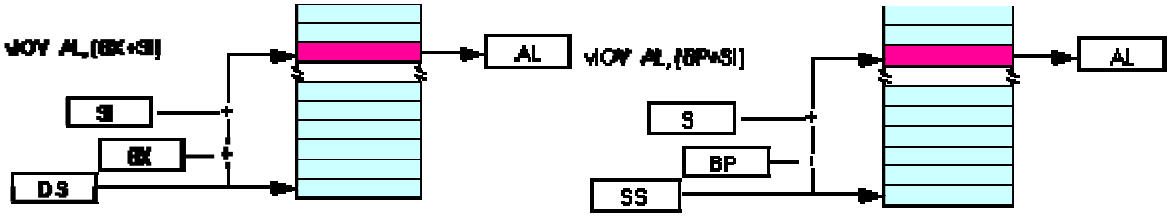
would load AL from location DS:1880h.

Likewise, if BP contains 1598h and DI contains 1004,

MOV AX,[BP+DI]

will load the 16 bits in AX from locations SS:259C and SS:259D.

The addressing modes that do not involve BP use the data segment by default. Those that have BP as an operand use the stack segment by default.

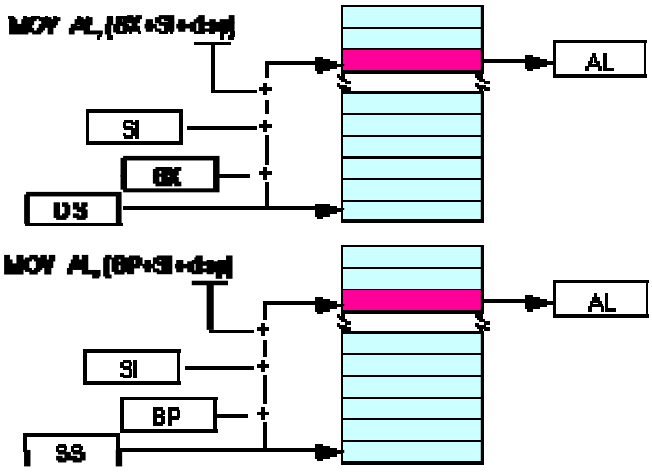


6. Based Indexed Plus Displacement Addressing Mode

These addressing modes are a slight modification of the base/indexed addressing modes with the addition of an eight bit or sixteen bit constant. The following are some examples of these addressing modes

MOV AL, DISP[BX][SI]
MOV AL, DISP[BX+DI]
MOV AL, [BP+SI+DISP]
MOV AL, [BP][DI][DISP]

You may substitute DI in the figure above to produce the [BX+DI+disp] addressing mode.



Q: Compute the physical address for the specified operand in each of the following instructions. The register contents and variable are as follows: (CS)=0A00H, (DS)=0B00H, (SS)=0D00H, (SI)=0FF0H, (DI)=00B0H, (BP)=00EAH and (IP)=0000H, LIST=00F0H, AX=4020H, BX=2500H.

1)	Destination operand of the instruction	MOV LIST [BP+DI] , AX
2)	Source operand of the instruction	MOV CL , [BX+200H]
3)	Destination operand of the instruction	MOV [DI+6400H] , DX
4)	Source operand of the instruction	MOV AL, [BP+SI-400H]
5)	Destination operand of the instruction	MOV [DI+SP] , AX
6)	Source operand of the instruction	MOV CL , [SP+200H]
7)	Destination operand of the instruction	MOV [BX+DI+6400H] , CX
8)	Source operand of the instruction	MOV AL , [BP- 0200H]
9)	Destination operand of the instruction	MOV [SI] , AX
10)	Destination operand of the instruction	MOV [BX][DI]+0400H,AL
11)	Source operand of the instruction	MOV AX, [BP+200H]
12)	Source operand of the instruction	MOV AL, [SI-0100H]
13)	Destination operand of the instruction	MOV DI,[SI]
14)	Destination operand of the instruction	MOV [DI]+CF00H,AH
15)	Source operand of the instruction	MOV CL, LIST[BX+200H]