8086 instructions

Instructions are classified on the basis of functions they perform. They are categorized into the following main types:

8086 instructions are divided into following types:

1. Data copy / Transfer instructions
2. Arithmetic instructions
3. Logical instructions
4. Branch instructions
5. Flag manipulation and Processor Control Instructions
6. Shift instructions
7. Rotate instructions
8. String instruction

1. Data copy/Transfer instructions

All the instructions which perform data movement come under this category. The source data may be a register, memory location, port etc. the destination may be a register, memory location or port. The following instructions come under this category:

Instruction	Description
MOV	Moves data from register to register, register to memory, memory to register, memory to accumulator, accumulator to memory, etc.
LDS	Loads a word from the specified memory locations into specified register. It also loads a word from the next two memory locations into DS register.
LES	Loads a word from the specified memory locations into the specified register. It also loads a word from next two memory locations into ES register.
LEA	Loads offset address into the specified register.
LAHF	Loads low order 8-bits of the flag register into AH register.
SAHF	Stores the content of AH register into low order bits of the flags register.
XLAT/XLATB	Reads a byte from the lookup table.
XCHG	Exchanges the contents of the 16-bit or 8-bit specified register with the contents of AX register, specified register or memory locations.
PUSH	Pushes (sends, writes or moves) the content of a specified register or memory location(s) onto the top of the stack.

POP	Pops (reads) two bytes from the top of the stack and keeps them in a specified register, or memory location(s).
POPF	Pops (reads) two bytes from the top of the stack and keeps them in the flag register.
IN	Transfers data from a port to the accumulator or AX, DX or AL register.
OUT	Transfers data from accumulator or AL or AX register to an I/O port identified by the second byte of the instruction.

MOV instruction

- copies the **second operand** (source) to the **first operand** (destination).
- the source operand can be an immediate value, register or memory location.
- the destination operand can be register or memory location.
- both operands must be the same size, which can be a byte or a word.

These types of operands are supported:

First operand , ***second operand***
destination , Source

MOV REG, memory
MOV memory, REG

MOV REG, REG

MOV memory, immediate
MOV REG, immediate

MOV SREG, memory
MOV memory, SREG

MOV REG, SREG
MOV SREG, REG

REG 8-bit : AH, AL, BL, BH, CH, CL, DH, DL.

REG 16-bit :AX, BX, CX, DX, SI, DI, BP, SP, and only IP is never used as destination.

SREG: DS, ES, SS, and only CS is never used as destination.

memory: [BX], [BX+SI+7], variable, etc...

immediate: 5, -24, 3Fh, 10001101b, etc...

Note: The MOV instruction <u>do not</u> affect the processor Flag register.

Note: The following must be observed in 8086 instructions:

1. **Never mix an 8-bit register with 16-bit, it is not allowed in microprocessor.**
2. **Code segment register (CS) and Instruction Pointer (IP) are never used as destination.**
3. **Segment with segment is not allowed.**
4. **Memory with memory is not allowed.**
5. **When used immediate with memory you must used (byteptr) of 8-bit or (wordptr) of 16-bit for the instructions have two operand. Also used for the instructions have one operand as memory.**

EX. Mov instruction:

MOV AL, BL ; Copies 8-bit content of BL into AL .

MOV AX, CX ; Copies 16-bit content of CX into AX

<u>The following Mov instructions are not allowed:</u>

1. MOV ES, DS ; Not allowed (segment to segment)

   The correct:
   MOV AX, DS
   MOV ES, AX

2. MOV BL, DX ; Not allowed (mixed size 8-bit with 16-bit)

   The correct:
   MOV BX , DX

3. MOV CS, AX ; Not allowed (Code segment register is never used as destination).

   The correct:
   MOV CX,AX

4. MOV [SI],[F900 h] : Not allowed (memory with memory)

   The correct:
   MOV BX, [F900 h]
   MOV [SI], BX

5. MOV [BX+03], VAR : Not allowed (memory with memory (because the variable VAR is defined in memory)).

   The correct:
   MOV AX, VAR
   MOV [BX+03], AX

6. MOV [F800 h], 44 h : Not allowed (you must specified byteptr or wordptr)

   The correct :

   MOV byteptr[F800 h], 44 h OR MOV wordptr[F800 h], 44 h

Example: What is the content of memory locations , AL ,AH, AX , BX and SI after the execution of the following instructions as a program :

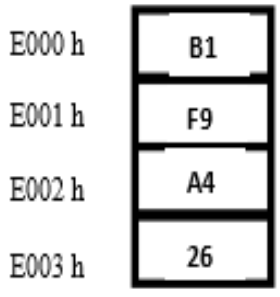
```
MOV  SI , E000 h
MOV byteptr[SI] , B1  h
MOV byteptr[SI+1] , F9  h
MOV wordptr[SI+2] , 26A4  h
MOV   AH,[SI]
MOV   AX,[SI]
MOV   AX,[SI+2]
MOV   BX,E002  h
MOV  AL,[BX-2]
MOV  AX,[BX-2]
MOV   [BX+2] , AX
```

Solution:

1. MOV SI , E000 h
   SI = E000 h

2. MOV byteptr[SI] , B1 h
   [SI] = [E000] = B1 h

3. MOV byteptr[SI+1] , F9 h
   [SI+1] = [E001] = F9 h



4. MOV wordptr[SI+2] , 26A4 h
   [SI+2] = [E002] = A4 h (low byte) and [E003] = 26 h (high byte)

5. MOV AH,[SI]
   AH = B1 h

6. MOV AX,[SI]
   AX = F9B1 h

7. MOV AX,[SI+2]
   AX = 26A4 h

8. MOV BX,E002 h
   BX = E002 h

9. MOV AL,[BX-2]
   AL = B1 h

10. MOV AX,[BX-2]
    AX = F9B1 h

11. MOV [BX+2] , AX
    [BX+2] = [E004] = B1 h and [E005] = F9 h
