National University of Computer & Emerging Sciences, Karachi

EL-213: Computer Organization & Assembly Language Lab

Lab 3: *Operators, Instructions & Flags*	**Session:** Fall 2019
Instructor(s): Sumaiyah Zahid, Fahim Ahmed	

MOV Instruction

It is used to move data from source operand to destination operand

• Both operands must be the same size.
• Both operands cannot be memory operands.
• CS, EIP, and IP cannot be destination operands.
• An immediate value cannot be moved to a segment register.

Syntax:
MOV *destination, source*

Example:
MOV *bx, 2*
MOV *ax, cx*

Example:
'A' has ASCII code 65D (01000001B, 41H)

The following MOV instructions stores it in register BX:

MOV bx, 65d
MOV bx, 41h
MOV bx, 01000001b
MOV bx, 'A'

All of the above are equivalent.

Examples:
The following examples demonstrate compatibility between operands used with MOV instruction:

MOV ax, 2	✓
MOV 2, ax	✗
MOV ax, *var*	✓
MOV *var*, ax	✓
MOV *var1*, *var2*	✗
MOV 5, *var*	✗

INC Instruction

The INC instruction takes an operand and adds 1 to it.

Example:
MOV *ax, 8*
INC *ax*   ; *ax now contains 9*

DEC Instruction

The DEC instruction takes an operand and subtracts 1 from it.

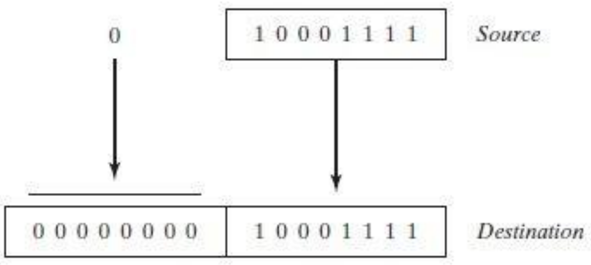
Example:
MOV *ax, 5*
DEC *ax* *; ax now contains 4*

MOVZX Instruction

The MOVZX (MOV with zero-extend) instruction moves the contents and zero-extends the value to 16 or 32 bits. This instruction is only used with unsigned integers.

Syntax:
MOVZX *reg32,reg/mem8*
MOVZX *reg32,reg/mem16*
MOVZX *reg16,reg/mem8*

Example:



The following examples use registers for all operands, showing all the size variations:

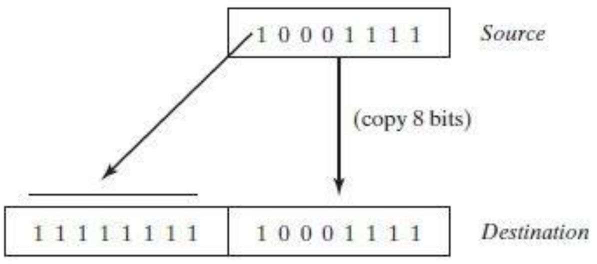
```
mov    bx,0A69Bh
movzx  eax,bx                  ; EAX = 0000A69Bh
movzx  edx,bl                  ; EDX = 0000009Bh
movzx  cx,bl                   ; CX  = 009Bh
```

MOVSX Instruction

The MOVSX (MOV with sign extend) instruction moves the contents and sign-extends the value to 16 or 32 bits. This instruction is only used with signed integers.

Example:

Using MOVSX to copy a byte into a 16-bit destination.



FLAGS Register

Status flags are updated to indicate certain properties of the result. Once a flag is set, it remains in that state until another instruction that affects the flags is executed.

Not all instructions affect all status flags:
• ADD and SUB affect all six flags
• INC and DEC affect all but the carry flag
• MOV, PUSH, and POP do not affect any flags

Z- Zero Flag:
This flag is set, if the result of the computation or comparison performed by the previous instruction is zero.

C- Carry Flag:
This flag is set, when there is a carry out of MSB in case of addition and borrow in case of subtraction.
Ranges of 8, 16, and 32 bit unsigned numbers are:
- 8 bits 0 to 255 (2^8 - 1)
- 16 bits 0 to 65,535 (2^16 - 1)
- 32 bits 0 to 4,294,967,295 (2^32-1)

S-Sign Flag:
This flag indicates the sign of the result of an operation. A 0 for positive number and 1 for a negative number.

AC-Auxilary Carry Flag:
This flag is set, if there is a carry from the lowest nibble, i.e., bit three during addition, or borrow for the lowest nibble, i.e. bit three, during subtraction.

P- Parity Flag:
This flag is set to 1, if the lower byte of the result contains even number of 1's

O- Over flow Flag:
This flag is set, if an overflow occurs, i.e., if the result of a signed operation is too large to fit into a destination register. Range of 8-, 16-, and 32-bit signed numbers:

- 8 bits (- 128 to +127)
- 16 bits (- 32,768 to +32,767 215)
- 32 bits (-2,147,483,648 to +2,147,483,647 231)

```
INCLUDE Irvine32.inc
.data
Rval SDWORD ?
Xval SDWORD 26
Yval SDWORD 30
Zval SDWORD 40
.code
main PROC
                ; INC and DEC
mov ax,1000h
inc ax          ; 1001h
dec ax          ; 1000h
                ; Expression: Rval = -Xval + (Yval - Zval)
mov eax,Xval
neg eax         ; -26
mov ebx,Yval
sub ebx,Zval    ; -10
```

```
add eax,ebx
mov Rval,eax    ; -36

                ; Zero flag example:
mov cx,1
sub cx,1        ; ZF = 1
mov ax,0FFFFh
inc ax          ; ZF = 1

                ; Sign flag example:
mov cx,0
sub cx,1        ; SF = 1
mov ax,7FFFh
add ax,2        ; SF = 1

                ; Carry flag example:
mov al,0FFh
add al,1        ; CF = 1, AL = 00

                ; Overflow flag example:
mov al,+127
add al,1        ; OF = 1
mov al,-128
sub al,1        ; OF = 1 exit
main ENDP
 END main
```

Exercises:

1. Convert the following high-level instruction into Assembly Language:
x = (x+1) – (y-1) + y

2. Write a program in assembly language that implements following expression:
eax = -val2 + 7 – val3 +val1

Use these data definitions:

```
val1  word 8
val2  word 15
val3  word 20
```

3. Write a program to find perimeter of a rectangle. Declare necessary variables length & width for the program (assign arbitrary values to the variables).

4. Use this code for the following questions:

```
.data
val1 BYTE 10h
val2 WORD 8000h
val3 DWORD 0FFFFh
val4 WORD 7FFFh
```

i. Write an instruction that increments val2.
ii. Write an instruction that subtracts val3 from EAX.
iii. Write instructions that subtract val4 from val2.
iv. If val2 is incremented by 1 using the ADD instruction, note down the values of Carry and Sign flags?
v. If val4 is incremented by 1 using the ADD instruction, note down the values of Overflow and Sign flag.