University of Central Punjab
(Incorporated by Ordinance No. XXIV of 2002 promulgated by Government of the Punjab)
FACULTY OF INFORMATION TECHNOLOGY

Computer Organization and Assembly Language

Lab 2	
Topic	1. Mov instruction 2. Viewing memory of dosbox 3. Declare variables

Types of Registers:-

The registers are grouped into three categories:-

1. **General Purpose registers**
   - 1.1. *Data registers*
     - 1.1.1. ***AX*** is the primary accumulator.
     - 1.1.2. ***BX*** is known as the base register.
     - 1.1.3. ***CX*** is known as the count register.
     - 1.1.4. ***DX*** is known as the data register.
   - 1.2. *Pointer registers*
     - 1.2.1. Instruction Pointer ***IP***
     - 1.2.2. Stack Pointer ***SP***
     - 1.2.3. Base Pointer ***BP***
   - 1.3. *Index registers*
     - 1.3.1. Source Index ***SI***
     - 1.3.2. Destination Index ***DI***
2. **Control registers**
   - 2.1. Instruction Pointer and Flag register
3. **Segment registers**
   - 3.1. Code Segment ***CS***
   - 3.2. Data Segment ***DS***
   - 3.3. Stack Segment ***SS***
   - 3.4. Extra Segment ***ES***

Types of variables

Type	No. of bits	Example declaration:
Byte	8	Num1: db 43
Word=> 2 bytes	16	Num2: dw 0xABFF
double word=> 2 words	32	Num3: dd 0xABCDEF56

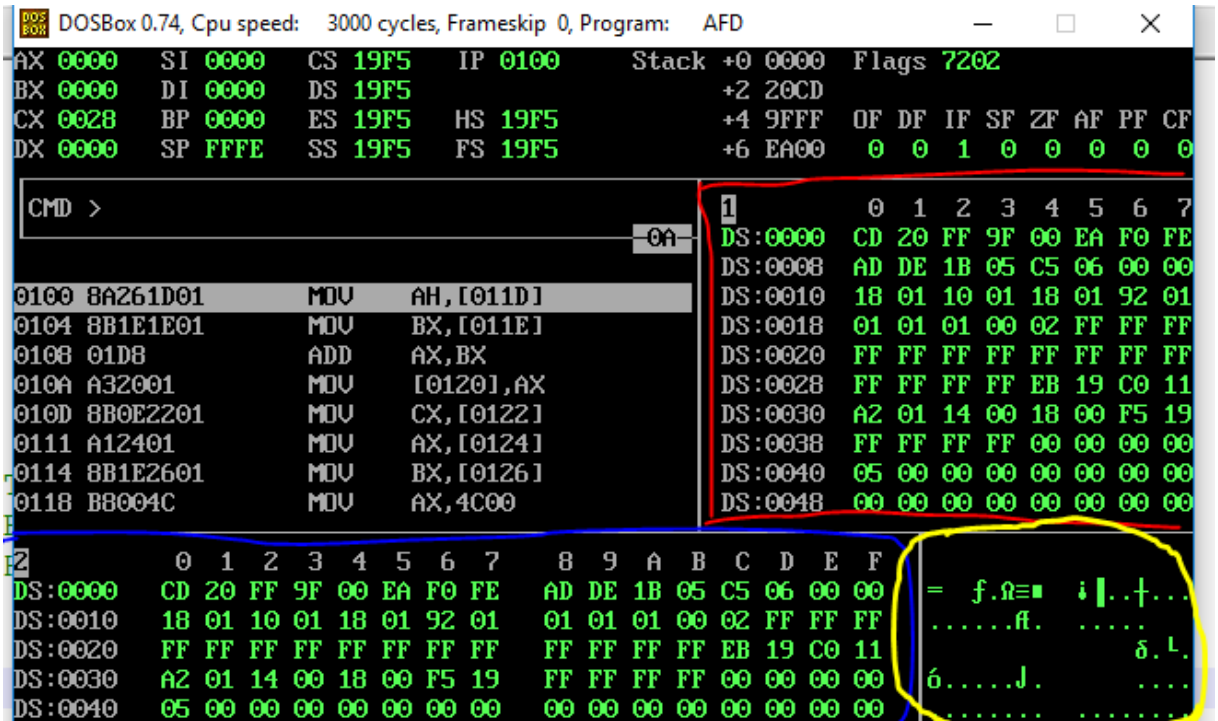
Note: size of both operands must be same for any type of instruction.

For example:
Mov ax,dh ;is wrong because destination is 2 bytes and source is 1 byte.

Viewing memory in DOSBOX

Areas highlighted in red(memory 1) "m1" and blue (memory 2) "m2" are showing the memory contents. *Note:* Two copies of the same memory is displayed in the given windows.

Area highlighted with yellow is showing the ascii values of the contents displayed in the memory m2.



Viewing sample variable in memory.

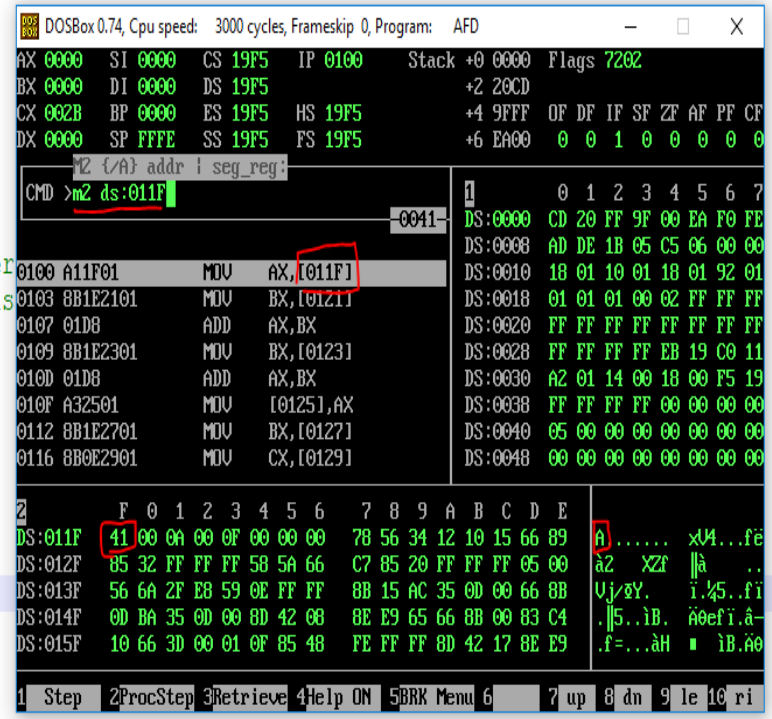
- To view memory from window m2 run the command "m2 ds:***Addressofvariable***" example: m2 ds:011F
- A variable with name "num1" is initialized at memory location 11F with value 65 decimal.
  41 hex = 65 decimal is the ascii of "A".

```
[org 0x0100]

mov ax, [num1] ; load first number in ax
mov bx, [num2] ; load second number in bx
add ax, bx ; accumulate sum in ax
mov bx, [num3] ; load third number in bx
add ax, bx ; accumulate sum in ax
mov [num4], ax ; store sum in num4
mov bx,[num5];load lower 2 bytes of num5 in bx register
mov cx,[num5+2];load higher 2 bytes of num5 in cx regis

mov ax, 0x4c00 ; terminate program
int 0x21

num1: dw 65
num2: dw 10
num3: dw 15
num4: dw 0
num5: dd 0x12345678
```



Example 1

```
;Example 1 Mov instruction variation
[org 0x0100]
mov  ax, 5     ; Load constant number in ax Register using mov instruction
mov  bx, 10    ; Load constant number in bx Register using mov instruction
add  ax, bx    ; Accumulate sum in ax
mov  bx, ax    ; Transfer value of Registers using mov instruction

mov  ax, 0x4c00  ; terminate program
int  0x21
```

Example 2

```
;Example 2  Mov Instruction for Memory based variables

mov  ax, [num1] ; Read Memory content from variable num1 in ax using mov instruction
mov  bx, [num2] ; Read Memory content from variable num2 in bx using mov instruction
add  ax,bx      ; Accumulate sum in ax
mov bx,[num3]   ; Read Memory content from variable num3 in bx using mov instruction
add ax,bx       ; Accumulate sum in ax


num1: db 5      ;num1 variable is of one byte having value 5
num2: db 3      ;num2 variable is of one byte having value 3
num3: db 9      ;num3 variable is of one byte having value 9
mov  ax, 0x4c00  ; terminate program
int  0x21
```

Example 3

```
; Example 3  Mov Instruction for Register Parts
[org 0x100]
mov ax,65   ;Load constant number in ax Register using mov instruction
mov bx,525  ;Load constant number in bx Register using mov instruction
add al,bh   ; Accumulate sum of lower part of ax,and higher part of bx
mov cl,200  ;Load constant number in cl Register using mov instruction
mov dl,56   ;Load constant number in dl of dx Register using mov instruction
add cl,dl   ;we are getting zero after addition Because cl is of 1 byte, and sum is more than one byte.
mov ax,0x4c00
int 21h
```

Add Comment in code:

Use ; to start comment

```
[org 0x100]
mov al,[num1]
mov bh,[Num1]
add ax,bx

mov cl,[num2]
mov dx,[mynum] ;when using dw variables use a 16-bit register.

add cx,dx

mov ax,0x4c00
int 21h

num1: db 01100001b ;b is for binary
Num1: db 97 ;decimal by default, case sensitive names of variables
num2: db 0x61 ; 0x treats it as a hexadecimal number
mynum: dw 6100h; h at the end treats it as a hexadecimal number
temp: dw 0xABCD ;when using characters as a hex values, use 0x
```

Practice Tasks

Task 1

Write a program in assembly language that calculates the square of six by adding six to the accumulator six times.

E.g $6^2 = 6 + 6 + 6 + 6 + 6 + 6 = 36$

Task 2

You have to take 5 variables of byte type. You have to take the sum of first two numbers and subtract the 3rd and 4th ones. After that, sum the outcome of first two number and 3rd and 4th ones and save it in 5th variable.

sum = num1 + num2

diff = num3 - num4

numb5 = sum + diff