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 03* | |
|---|---|
| **Topic** | 1. Direct Addressing Mode with variations. |

# PART 1

**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.***

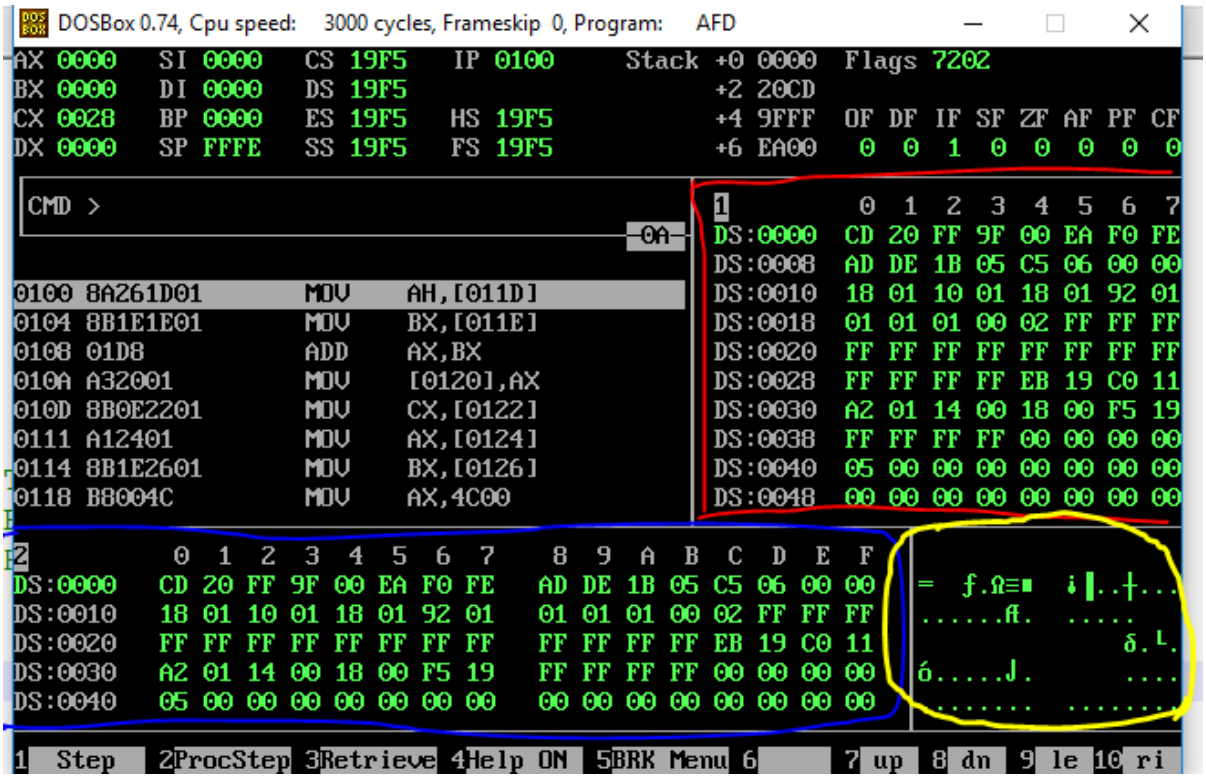
For example:

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.***

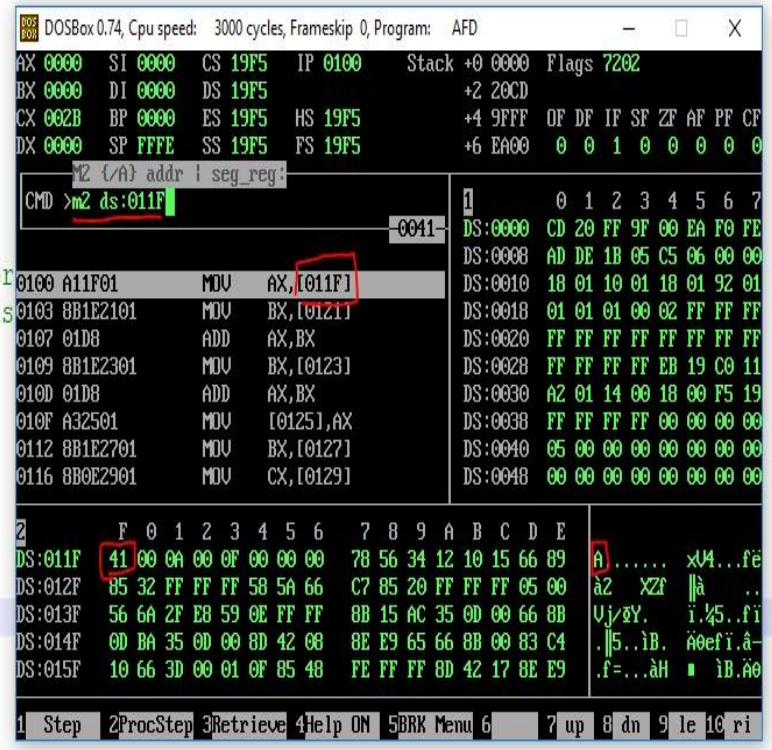
- To view memory from window m2 run the command "m2 ds:***Addressofvariable***" example: m2 ds:011F
- A variable with name "num1" is initialized at memory location 11F with value 65 decimal.
  41 hex = 65 decimal is the ascii of "A".

```
[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
```



# Direct Addressing Mode

| | |
|---|---|
| **Direct**<br>A fixed offset is given in brackets and the memory at that offset is accessed. For example "mov [1234], ax" stores the contents of the AX registers in two bytes starting at address 1234 in the current data segment. The instruction "mov [1234], al" stores the contents of the AL register in the byte at offset 1234. | • Mov ax,[num1] ;reading<br>• Mov [num2],ax ;writing |

Execute every part of Question 1 in ***Nasm with Dosbox*** and observe the memory variables and register values.

## Example 1.

```
1  ; a program to add three numbers using memory variables by direct mode.
2  [org 0x0100]
3  mov  ax, [num1]         ; load first number in ax
4  mov  bx, [num2]         ; load second number in bx
5  add  ax, bx             ; accumulate sum in ax
6  mov  bx, [num3]         ; load third number in bx
7  add  ax, bx             ; accumulate sum in ax
8  mov  [num4], ax         ; store sum in num4
9
10 mov  ax, 0x4c00         ; terminate program
11 int  0x21
12
13 num1:dw 5
14 num2:dw 10
15 num3:dw 15
16 num4:dw 0
```

Example 2

```
1  ; a program to add three numbers accessed using a single label
2  [org 0x0100]
3  mov  ax, [num1]          ; load first number in ax
4  mov  bx, [num1+2]        ; load second number in bx
5  add  ax, bx              ; accumulate sum in ax
6  mov  bx, [num1+4]        ; load third number in bx
7  add  ax, bx              ; accumulate sum in ax
8  mov  [num1+6], ax        ; store sum at num1+6
9  mov  ax, 0x4c00          ; terminate program
10 int  0x21
11
12 num1:dw   5
13      dw   10
14      dw   15
15      dw   0
```

Example 3

```
1
2  ; a program to add three numbers using byte variables
3  [org 0x0100]
4  mov  al, [num1]          ; load first number in al
5  mov  bl, [num1+1]        ; load second number in bl
6  add  al, bl              ; accumulate sum in al
7  mov  bl, [num1+2]        ; load third number in bl
8  add  al, bl              ; accumulate sum in al
9  mov  [num1+3], al        ; store sum at num1+3
10
11 mov  ax, 0x4c00          ; terminate program
12 int  0x21
13
14 num1:          db   5, 10, 15, 0
```

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

# Practice Tasks

Write a program to solve the following:

***Use Direct addressing mode to access memory variables and the objective of this lab is when we declare the variable of different types how they reserve space in memory:***

***Note: when we declare variables of same types space reserve in memory side by side therefore we can excess all variables using first one.***

***Question 1:***

Let

Var1=10
Var2=20
Var3=2
Var4=50
Var5=90

Save the sum of these (using **Direct** addressing mode) Five variables (Var1+ Var2+ Var3+ Var4+Var5) in ax and try this question using different data types of variable i.e db,dw,dd.

***Question 2:***

***Declare then variables v1, v2... v10 and add the even no variables v2, v4..v10 using direct***

***Addressing mode using dw type.***

***Question 3:***

***Calculate the table of two and save final result in ax.***