Microprocessor and Interfacing Lab Experiment 1

Name: Om Ashish Mishra
Registration No.: 16BCE0789
Slot: B2
Date: 11/12/2017

Write an ALP

1. To perform basic arithmetic operations using 8 bit and 16 bit number (Add, SUB, MUL, and DIV).

16-Bit

Addition

```
mov ax, 5000h
mov ds, ax
mov ax, 23h
mov bx, 10h
add ax, bx
hlt
```

Subtraction

```
mov ax, 5000h
mov ds, ax
mov ax, 23h
mov bx, 10h
sub ax, bx
hlt
```

Multiplication

```
mov ax, 5000h
mov ds, ax
mov ax, 200h
mov bx, 4
mul bx
ret
```

Division

```
mov ax, 5000h
mov ds, ax
mov ax, 0020h
mov bx, 4
div bx
```

ret

8-Bit

Addition

```
mov al, 5000h
mov ah, al
mov al, 23h
mov bl, 10h
add ah, bl
hlt
```

Subtraction

```
mov al, 5000h
mov ah, al
mov al, 23h
mov bl, 10h
sub ah, bl
hlt
```

Multiplication

```
mov al, 5000h
mov ah, al
mov al, 200h
mov bl, 4
mul bl
ret
```

Division

```
mov al, 5000h
mov ah, al
mov al, 0020h
mov bl, 4
div bl
ret
```

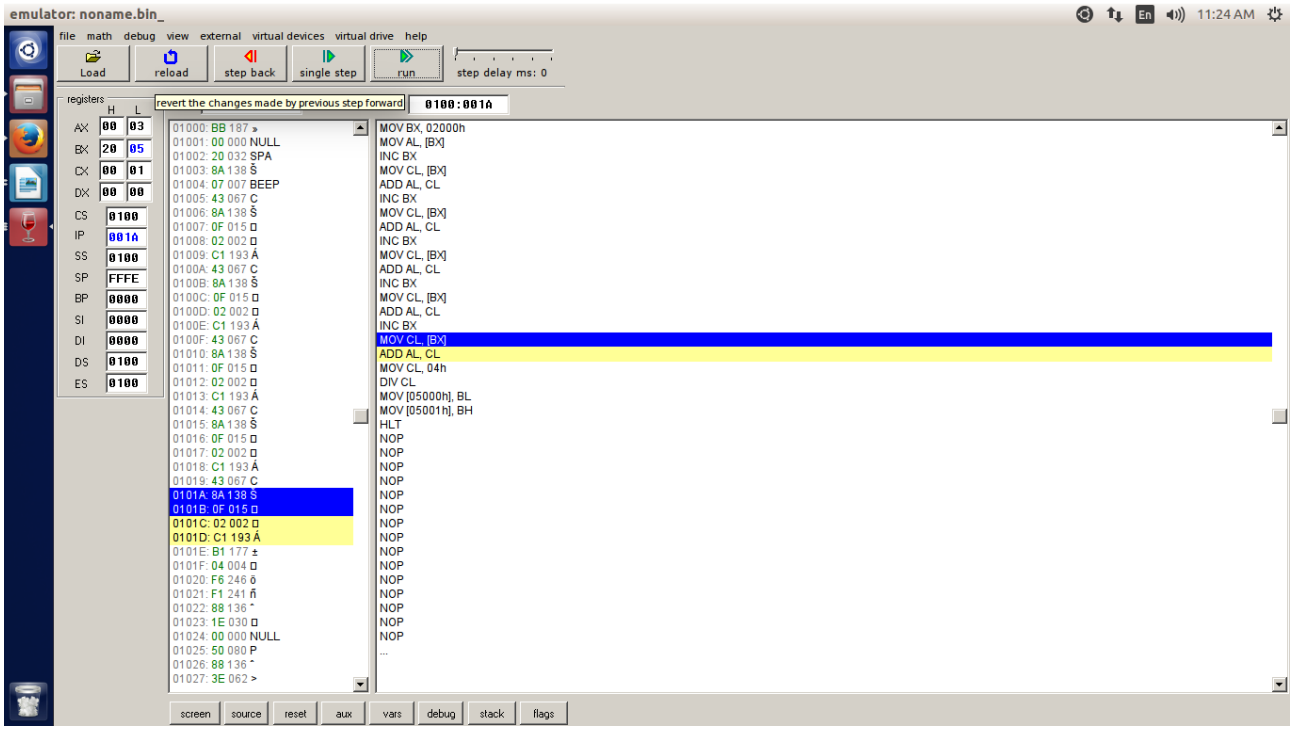
2. Write an ALP to perform the sum and average of 5 numbers stored in the address location 2000h onwards and put the result in memory locations starting from 5000h onwards.

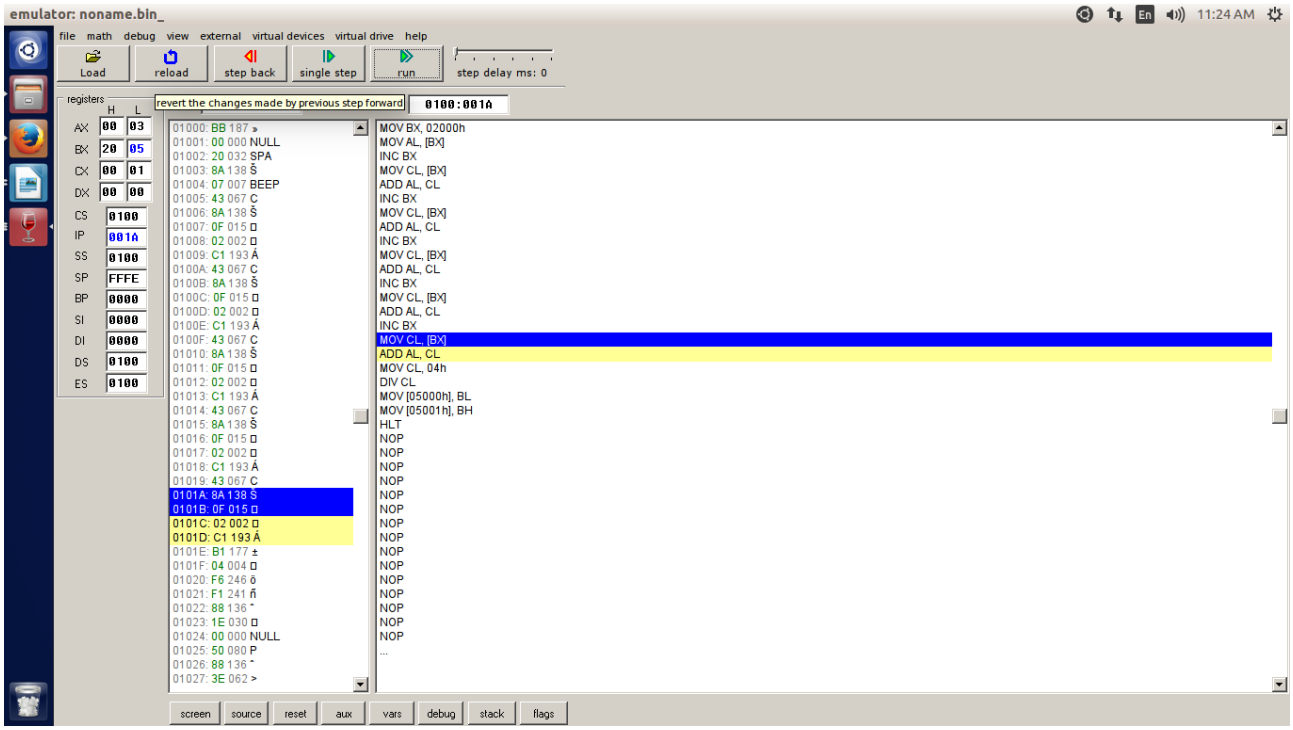
```
mov bx, 02000h
mov al,[bx]
inc bx
mov cl,[bx]
add al,cl
```

```
inc bx
mov dl,[bx]
add al,cl
inc bx
mov dl,[bx]
add al,dl
mov cl,04h
div cl
mov [5000h],al
mov [5001h],ah
hlt
```

Test Cases : Input Value and the result:

Input Value:



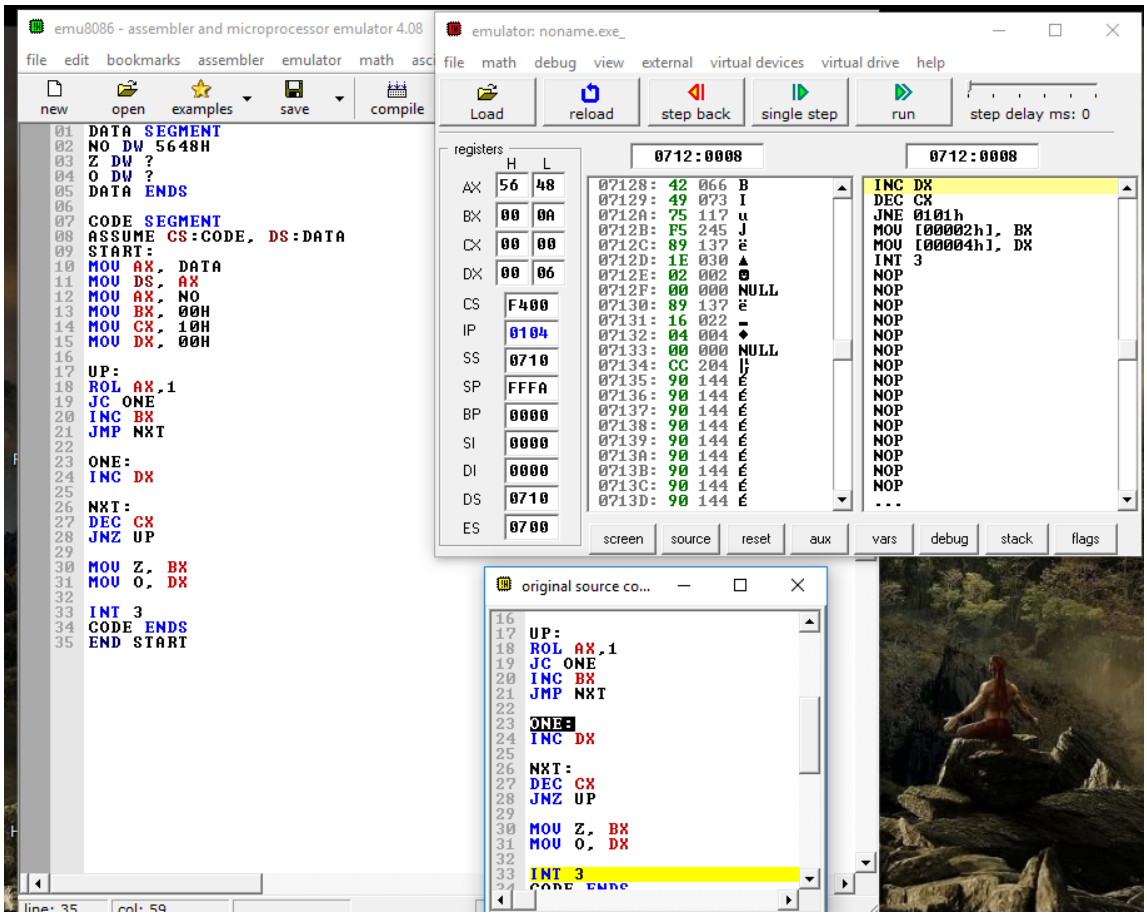


3. Write a program to count the number of 1's in a given byte located in AL register..(Hint Use Rotate instruction : Eg : Input(AL) : 0A7 h Output : 5))

```
DATA SEGMENT
NO DW 5648H
Z DW ?
O DW ?
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
MOV AX, NO
MOV BX, 00H
MOV CX, 10H
MOV DX, 00H

UP:
ROL AX,1
JC ONE
INC BX
JMP NXT

ONE:
INC DX

NXT:
DEC CX
JNZ UP
```

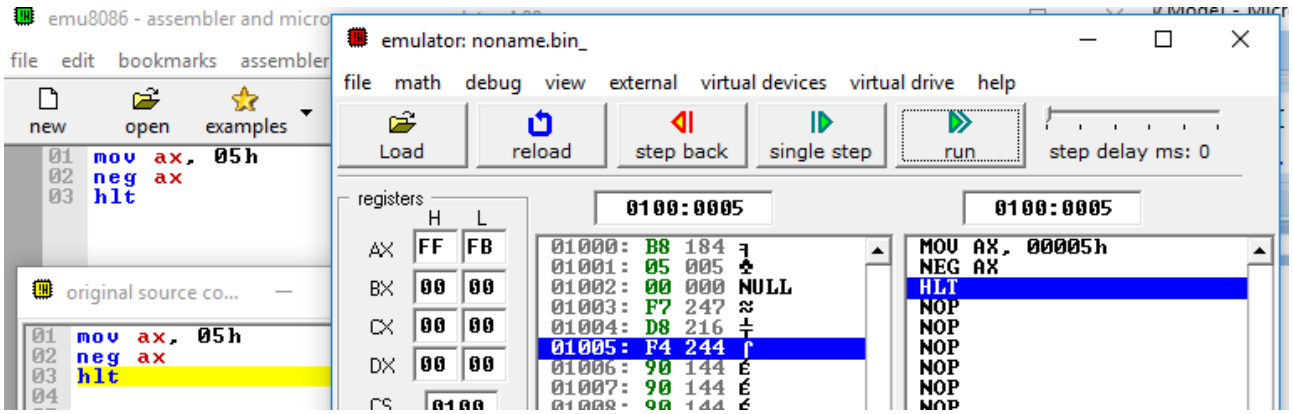
MOV Z, BX
MOV O, DX

INT 3
CODE ENDS
END START

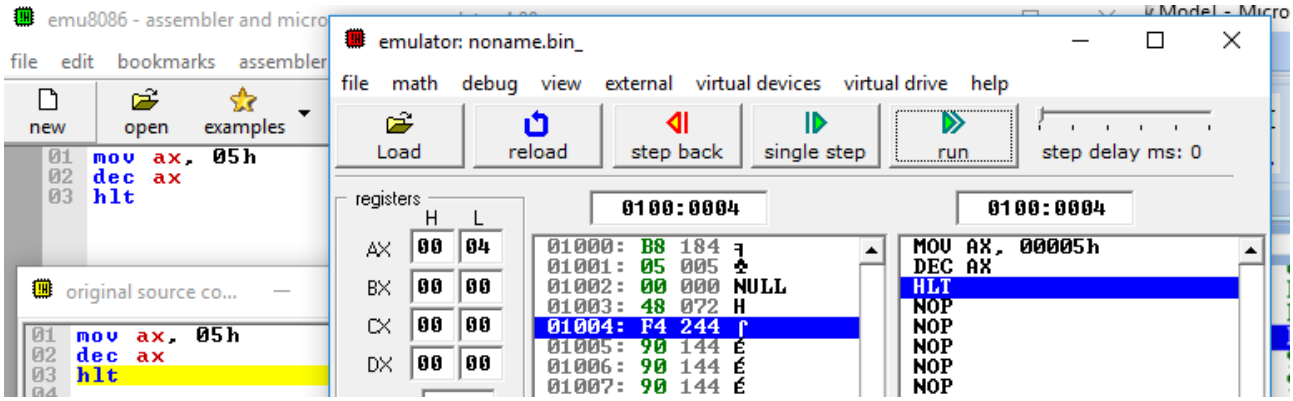


4. Analyse Any five instructions – NEG,DEC, AAM ,IMUL SAR,SL,SR(take any input number and show the output)

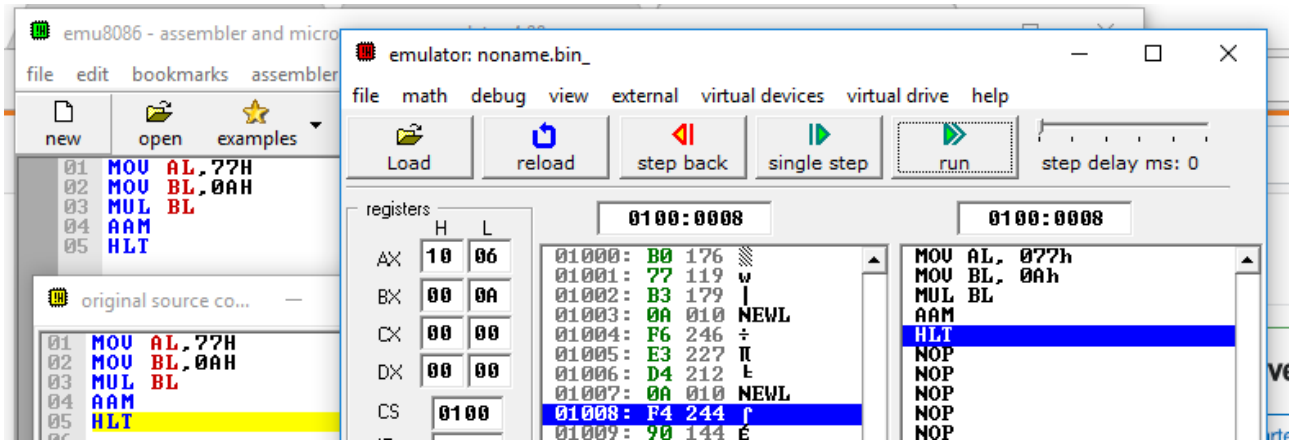
NEG: Creates 2's complement of original value (i.e., forms the negative of a given value).



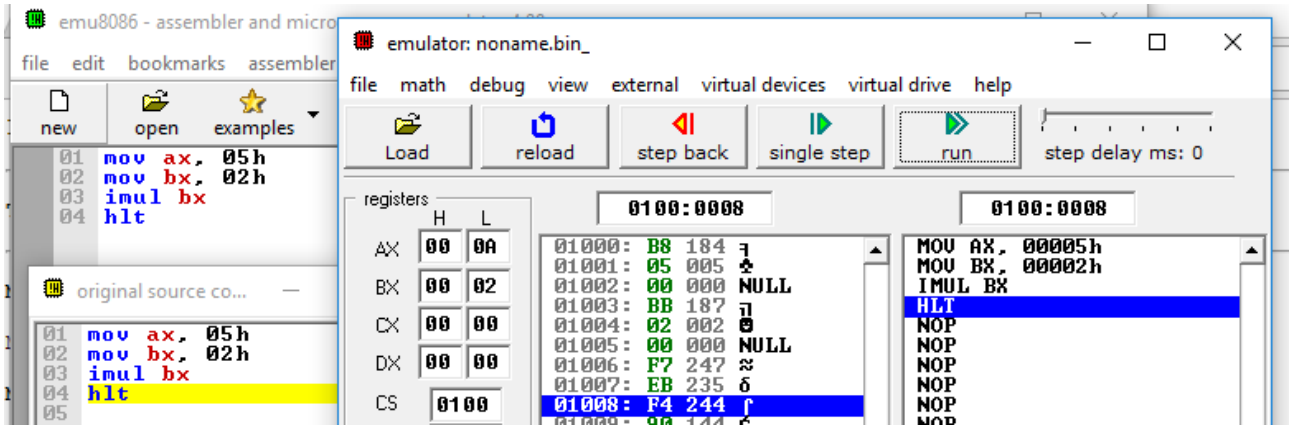
DEC: It is used to decrease the value by one.



AAM: Adjusts the result of the multiplication of two unpacked BCD values to create a pair of unpacked (base 10) BCD values. The AX register is the implied source and destination operand for this instruction. The AAM instruction is only useful when it follows an MUL instruction that multiplies (binary multiplication) two unpacked BCD values and stores a word result in the AX register. The AAM instruction then adjusts the contents of the AX register to contain the correct 2-digit unpacked (base 10) BCD result.



IMUL: This is used for signed multiplication



SAR: shifts to the right and preserves the sign bit for positive or negative numbers. I.e., you should use SAR instead of SHR when you are using signed numbers in a program.