# Comp 3350: Computer Organization & Assembly Language

# HW # 4: Theme: Debugging, Flags, Data Declarations

*All main questions carry equal weight.*

*(Credit awarded to only those answers for which work has been shown. All questions must be worked by hand)*

1. A. What will be the value of EAX and sign flag after the following lines execute?

Mov eax, 6

Sub eax, 5

* Eax = 00000001
* SF = 0

B. Is it possible to set the overflow flag if you add a negative integer to a negative integer? Explain with examples where applicable.

Yes because if you add two negatively signed intergers past the max bits the overflow flag will be set and the result will be “positive”. i.e.

mov al -127

add al -32

OF =1

The overflow flag will be set as -127+-32 is clearly greater than -128 which is the limit for the al register.

C. Is it possible for both the sign and zero flags to be set at that same time? State reasons.

No as the sign flag is the MSB 0 will have a 0 for its MSB thus it will be the clear on the sign flag when the zero flag is set.

D. In the following code, the value in AL is intended to be a signed byte. Does the overflow flag help to determine whether the final value in AL is within a valid signed range? Explain.

mov al, -1

add al, 130

By rules, a negative quantity and a positive quantity when added together can never set OV = 1. This rule is the driver for the OV/OF value. Thus, regardless of the result being out of the range -128..+127 for any signed 8-bit value.   
 E. What will be the value of the parity flag after the following lines execute?

Mov al, 2

Add al, 5

The parity flag will be set to 0 since 7 is and odd number.

1. Given the following data declarations:

.data

Alpha BYTE 1Ah, 2Bh, 3Ch, 4CH

Beta DWORD 11223344h

Delta DWORD 44332211h

Iota DWORD 434h

Zeta WORD 124h

1. Write instructions that move *Delta* into *ECX* and then adds *Beta* to the same register
   1. mov ECX, Detla
   2. add ECX, Beta
2. Write a set of instructions that adds all the elements of the array *Alpha* into AL
   1. mov al, 0
   2. add al, [Alpha+1]
   3. add al, [Alpha+2]
   4. add al, [Alpha+3]
3. Write a set of instructions that moves *Beta* into *EAX,* adds the value stored in *Zeta* to the same register

mov eax, Beta

add eax, DWORD PTR Zeta

1. Write an instruction that moves the last two bytes in *ALPHA* into *AX*.
   1. mov ax, WORD PTR [ALPHA+2]
2. What are the contents of AX subsequent to part D of this question?
   1. 4C3C
3. Fill in the requested register values after executions of the instructions:

Show the memory map.

.data

myBytes BYTE 11h, 22h, 33h, 44h

myWords WORD 1234h, 5678h, ABCDh, EF01h, 2345h

myDoubles DWORD AB23h, BC34h, CD54h, 8967h, 6F6Ah

myPointer DWORD myDoubles

.code

mov esi, OFFSET myBytes

mov ax, WORD PTR [esi+2] ; A. AX = 4433h

mov eax, DWORD PTR myWords ; B. EAX = 56781234h

mov esi, myPointer

mov ax, WORD PTR [esi+8] ; C. AX = CD54h

mov ax, WORD PTR [esi+1] ; D. AX = 00ABh

mov ax, WORD PTR [esi-6] ; E. AX = ABCDh

Memory Map

|  |  |
| --- | --- |
| BYTE | OFFSET |
| 11 | myBytes |
| 22 | +1 |
| 33 | +2 |
| 44 | +3 |
| 34 | myWords |
| 12 | +1 |
| 78 | +2 |
| 56 | +3 |
| CD | +4 |
| AB | +5 |
| 01 | +6 |
| EF | +7 |
| 45 | +8 |
| 23 | +9 |
| 23 | myDoubles  myPointer |
| AB | +1 |
| 00 | +2 |
| 00 | +3 |
| 34 | +4 |
| BC | +5 |
| 00 | +6 |
| 00 | +7 |
| 54 | +8 |
| CD | +9 |
| 00 | +10 |
| 00 | +11 |
| 67 | +12 |
| 89 | +13 |
| 00 | +14 |
| 00 | +15 |
| 6A | +16 |
| 6F | +17 |
| 00 | +18 |
| 00 | +19 |

1. What is the value of ax after the follow:

.data

myArray DWORD 7 DUP (2), 21, 4, 65, 0CDE

.code

mov ax, TYPE myarray ax = 4

mov ax, lengthof myarray ax = B

mov ax, sizeof myarray ax = 2C

1. Fill in the requested register values after executions of the instructions (Do not let your eyes deceive you. There are some mov**S**x instructions and some mov**Z**x instructions.):

.code

mov bx, 0B24Ch

movzx eax, bx ; A. EAX = 0000B24C

movzx edx, bl ; B. EDX = 0000004C

movzx cx, bh ; C. CX = 00B2

mov bx, 0D3C2h

movsx eax, bx ; D. EAX = FFFFD3C2

movsx edx, bl ; E. EDX = FFFFFFC2

movsx cx, bh ; F. CX = FFD3

1. What will be the value of the destination operand after each of the following instructions execute?

.data

var1 SBYTE -4, -2, 3, 1

var2 WORD 1000h, 2000h, 3000h, 4000h

var3 SWORD -16, -42

var4 DWORD 1, 2, 3, 4

.code

mov ax, var2 ;ax = 1000

mov ax, [var2+4] ;ax = 3000

mov ax, var3 ;ax = FFF0

mov ax, [var3-2] ;ax = 4000