

Lab11

Hyunki Lee

Problem1

- a. al= 0019FF6A CF= 0 OF= 1
- b. al= 0019FFEA CF= 0 OF= 0
- c. al= 0019FFFD CF= 0 OF= 0
- d. al= 0019FFA9 CF= 1 OF= 0
- e. al= 0019FF9A CF= 1 OF= 0

Problem2

Output al= 0019FF90

The screenshot shows the Microsoft Visual Studio IDE in debug mode. The assembly code for `AddTwo.asm` is displayed in the main window, and the `Registers` window on the right shows the current state of the CPU registers.

Assembly Code (AddTwo.asm):

```
1 ; AddTwo.asm - adds two 32-bit integers.
2 ; Chapter 3 example
3
4 .386
5 .model flat,stdcall
6 .stack 4096
7 ExitProcess proto,dwExitCode:dword
8 .data
9
10 .code
11 main proc
12     mov al,9
13     shl al,4
14     invoke ExitProcess,0
15 main endp
16 end main
```

Registers:

EAX = 0019FF90	EBX = 00300000
ECX = 00401005	EDX = 00401005
ESI = 00401005	
EDI = 00401005	EIP = 00401015
ESP = 0019FF74	
EBP = 0019FF80	EFL = 00000286

Flags:

OV = 0	UP = 0	EI = 1	PL = 1
ZR = 0	AC = 0	PE = 1	CY = 0

Problem3

Var4: 00190080

The screenshot displays the Microsoft Visual Studio IDE in a debugging state. The main window shows the assembly file `AddTwo.asm` with the following code:

```
4 .386
5 .model flat,stdcall
6 .stack 4096
7 ExitProcess proto,dwExitCode:dword
8 .data
9 var1 word 40
10 var2 byte 13
11 var4 word ?
12 .code
13 main proc
14     mov ax,var1
15     mov bl,5
16     div bl
17     mov cl,var2
18     add cl,3
19     mov bl,0
20     mov bl,cl
21     mul bl
22     mov var4,ax
23     invoke ExitProcess,0 ≤ 1ms elapsed
24 main endp
25 end main
```

The `Registers` window on the right shows the following values:

- EAX = 00190080
- EBX = 003CC010
- ECX = 00401010
- EDX = 00401005
- ESI = 00401005
- EDI = 00401005
- EIP = 0040102F
- ESP = 0019FF74
- EBP = 0019FF80
- EFL = 00000282

Below the registers, the status of various flags is shown:

- OV = 0, UP = 0, EI = 1, PL = 1
- ZR = 0, AC = 0, PE = 0, CY = 0

The `Memory` window at the bottom shows the memory address `0x00404003` with the following data:

Address	Value
0x00404003	80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00404017	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0040402B	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0040403F	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00404053	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00404067	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0040407B	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

The `Memory` window also includes tabs for `Call Stack`, `Breakpoints`, `Exception Settings`, `Command Window`, `Immediate Window`, `Output`, and `Error List`.

Problem4

Var4= 00190046

I figure out that idiv is also using ax register, thus I used ax as Dividend and Multiplicand.

The screenshot shows the Microsoft Visual Studio IDE in Debug mode. The main window displays the assembly code for 'AddTwo.asm'. The code is as follows:

```
8 .data
9 var1 sword -2
10 var2 sword 20
11 var3 sword 2
12 var4 sdword ?
13 .code
14 main proc
15     mov cx,var1
16     sub cx,5
17
18     mov ax,var2
19     neg ax
20     cwd
21     mov bx,var3
22     idiv bx
23
24     mov bl,0
25     mov bx,cx
26     imul bx
27     mov var4,eax
28     invoke ExitProcess,0
29 main endp
30 end main
```

The 'Registers' window on the right shows the following values:

Register	Value
EAX	00190046
EBX	0034FFF9
ECX	0040FFF9
EDX	00400000
ESI	00401005
EDI	00401005
EIP	0040103D
ESP	0019FF74
EBP	0019FF80
EFL	0000202

The 'Memory' window at the bottom shows the memory dump starting at address 0x00404006. The first line of memory contains the value 46 00 19 00, which corresponds to the hexadecimal value 00190046 shown in the registers window.