



Name: Abdul Ghani Khan

Roll No: 22P-9037

Section: BCS-3D

Course: Computer Organization & Assembly Language

Assignment: 01

Question No. 1:

Write a program in assembly language for each of the below separately that sets

the following flags.

(Write four programs i.e. One for each part)

A) Zero Flag

Code

```
1 [org 0x0100]
2
3 mov ax, 1
4 sub ax, 1
5
6 mov ax, 0x4c00
7 int 0x21
8
```

Result

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

Register	Value	Segment	Value	Register	Value	Segment	Value
AX	0000	SI	0000	CS	19F5	IP	0106
BX	0000	DI	0000	DS	19F5		
CX	000B	BP	0000	ES	19F5	HS	19F5
DX	0000	SP	FFFE	SS	19F5	FS	19F5

Stack: +0 0000, +2 20CD, +4 9FFF, +6 EA00

Flags: 7244 (ZF=1, AF=0, PF=0, CF=0)

CMD >

Address	Instruction	Disassembly	Hex
0103	2D0100	SUB AX,0001	CD 20 FF 9F 00 EA F0 FE
0106	B8004C	MOV AX,4C00	AD DE 1B 05 C5 06 00 00
0109	CD21	INT 21	18 01 10 01 18 01 92 01
010B	01B9C3B9	ADD [B9C3+BX+DI],CX	01 01 01 00 02 FF FF FF
010F	D0B9DAEB	ROR B/[EBDA+BX+DI],1	FF FF FF FF FF FF FF FF
0113	0431	ADD AL,31	FF FF FF FF EB 19 C0 11
0115	D2	DB D2	A2 01 14 00 18 00 F5 19
0116	31C0	XOR AX,AX	FF FF FF FF 00 00 00 00
0118	8956E4	MOV [BP-1C],DX	05 00 00 00 00 00 00 00

Memory dump (DS:0000):

Address	Hex	ASCII
0000	CD 20 FF 9F 00 EA F0 FE	= f.Ω= i .+....
0010	18 01 10 01 18 01 92 01f.
0020	FF FF FF FF FF FF FF FFδ.L.
0030	A2 01 14 00 18 00 F5 19	6.....J.
0040	05 00 00 00 00 00 00 00

Step 1 ProcStep 2 Retrieve 3 Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri

B) Carry Flag

Code

```
1 [org 0x0100]
2
3 mov ax, 0xFFFF
4 add ax, 1
5
6 mov ax, 0x4c00
7 int 0x21
```

Result

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 0000 SI 0000 CS 19F5 IP 0106 Stack +0 0000 Flags 7255

BX 0000 DI 0000 DS 19F5 +2 20CD

CX 000B BP 0000 ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF

DX 0000 SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 1 1 1

CMD >

0103 050100 ADD AX,0001

0106 B8004C MOV AX,4C00

0109 CD21 INT 21

010B 0189C389 ADD [89C3+BX+DI],CX

010F D089DAEB ROR B/[EBDA+BX+DI],1

0113 0431 ADD AL,31

0115 D2 DB D2

0116 31C0 XOR AX,AX

0118 8956E4 MOV [BP-1C],DX

DS:0000 CD 20 FF 9F 00 EA F0 FE

DS:0008 AD DE 1B 05 C5 06 00 00

DS:0010 18 01 10 01 18 01 92 01

DS:0018 01 01 01 00 02 FF FF FF

DS:0020 FF FF FF FF FF FF FF FF

DS:0028 FF FF FF FF EB 19 C0 11

DS:0030 A2 01 14 00 18 00 F5 19

DS:0038 FF FF FF FF 00 00 00 00

DS:0040 05 00 00 00 00 00 00 00

DS:0048 00 00 00 00 00 00 00 00

2 0 1 2 3 4 5 6 7 8 9 A B C D E F

DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00 = f.0= i[...+...

DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FFf.

DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11δ.L.

DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 6.....J.

DS:0040 05 00 00 00 00 00 00 00 00 00 00 00 00 00 00

1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri

C) Parity Flag

Code

```
1 [org 0x0100]
2
3 mov ax, 10
4 add ax, 2
5
6 mov ax, 0x4c00
7 int 0x21
8
```

Result

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip: 0, Program: AFD
AX 000C SI 0000 CS 19F5 IP 0106 Stack +0 0000 Flags 7204
BX 0000 DI 0000 DS 19F5 +2 20CD
CX 000B BP 0000 ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF
DX 0000 SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 0 0 1 0

CMD >
0103 050200 ADD AX,0002
0105 B8004C MOV AX,4C00
0109 CD21 INT 21
010B 0189C389 ADD EB9C3+BX+DI,1,CX
010F D089DAEB ROR B/[EBDA+BX+DI],1
0113 0431 ADD AL,31
0115 D2 DB D2
0116 31C0 XOR AX,AX
0118 B956E4 MOV EB9C3+BX+DI,DX

DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
DS:0048 00 00 00 00 00 00 00 00

2 0 1 2 3 4 5 6 7 8 9 A B C D E F
DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00 = f.Ω≡ i|.+.
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF .....f. ....
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11 .....δ.L.
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 ó.....J. ....
DS:0040 05 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

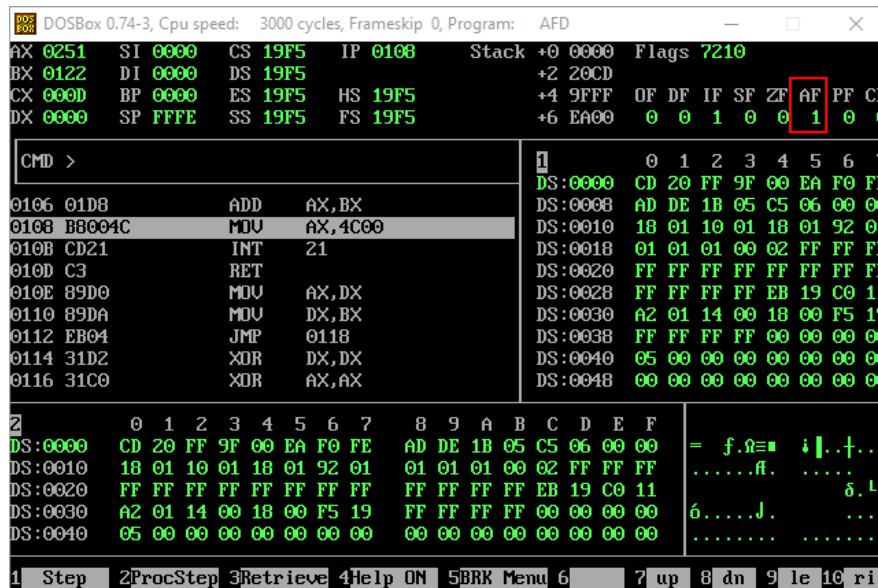
1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri
```

D) Auxiliary Flag

Code

```
1 [org 0x0100]
2
3 mov bx, 0x0122
4 mov ax, 0x012F
5
6 add ax, bx
7
8 mov ax, 0x4c00
9 int 0x21
10
```

Result



The screenshot shows the DOSBox 0.74-3 interface. At the top, it displays 'Cpu speed: 3000 cycles, Frameskip 0, Program: AFD'. Below this, the register state is shown: AX 0251, SI 0000, CS 19F5, IP 0108, Stack +0 0000, Flags 7210; BX 0122, DI 0000, DS 19F5; CX 0000, BP 0000, ES 19F5, HS 19F5; DX 0000, SP FFFE, SS 19F5, FS 19F5. The command line shows 'CMD >'. The assembly code is listed on the left, and the memory dump is on the right. The memory dump shows the current instruction at address 0108: CD 20 FF 9F 00 EA F0 FE, which corresponds to the instruction 'MOV AX, 4C00' in the code. The status bar at the bottom shows '1 Step 2 ProcStep 3 Retrieve 4 Help ON 5 BRK Menu 6 7 up 8 dn 9 le 10 ri'.

Question No. 2:

What will be the size of the following assembly language program in bytes?

Explain your answer using “.lst” file of this code.

```
1                                     [org 0x0100]
2 00000000 B80500                    mov ax, 5
3 00000003 BB0A00                    mov bx, 10
4 00000006 01D8                      add ax, bx
5 00000008 BB0F00                    mov bx, 15
6 0000000B 01D8                      add ax, bx
7
8 0000000D B8004C                    mov ax, 0x4c00
9 00000010 CD21                      int 0x21
```

Explanation: -

Let's Break down one by one as we know that the mov instruction takes 3 bytes and its used four times so $3 * 4$ is 12. Also, we know that the add instruction takes 2 bytes and it is used twice so $2 * 2$ is 4 so $12 + 4$ is 16. Now the last instruction (int 0x21) takes 2 bytes Now adding them $16 + 2 = 18$ so this program takes 18 bytes

Line By Line If We Add: $3 + 3 + 2 + 3 + 2 + 3 + 2 = 18$

Also looking at the .lst file our answer matches by counting the number of bytes each line of the instructions is taking.

Question No. 3:

Calculate the physical memory address generated by the following segment-offset pairs:

A. 1DDD:0436

Physical Address: **1E206**

B. 1234:7920

Physical Address: **19C60**

C. 74F0:2123

Physical Address: **77023**

D. 0000:6727

Physical Address: **06727**

E. FFFF:4336

Physical Address: **04326**

F. 1080:0100

Physical Address: **10900**

Rough Work

The image shows handwritten calculations for six segment-offset pairs, numbered 1 through 6. Each calculation involves adding a segment value (shifted left by 4 hex digits) to an offset value.

① 1DDD : 0436.

$$\begin{array}{r} 1\text{DDD}0 \\ + 00436 \\ \hline 1\text{E}206 \end{array}$$

② 1234 : 7920

$$\begin{array}{r} 12340 \\ + 07920 \\ \hline 19\text{C}60 \end{array}$$

③ 74F0 : 2123

$$\begin{array}{r} 74\text{F}00 \\ + 02123 \\ \hline 77023 \end{array}$$

④ 0000 : 6727

$$\begin{array}{r} 00000 \\ + 06727 \\ \hline 06727 \end{array}$$

⑤ FFFF : 4336.

$$\begin{array}{r} \text{FFFF}0 \\ + 04336 \\ \hline 04326 \end{array}$$
This will be dropped because it is a wraparound case. It will go into the carry.

⑥ 1080 : 0100

$$\begin{array}{r} 10800 \\ + 00100 \\ \hline 10900 \end{array}$$