Department of Computer Science

Date: June 29, 2020

Spring Semester 2020
Microprocessor & Assembly Language

Name _____ **Mohammad Basir**

ID _____ **13142**

INSTRUCTOR Name _____ **Muhammad Amin**

Semester _____ **8th**

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

ANSWER :

Direct-Offset Operands

You can add a displacement to the name of a variable, creating a direct-offset operand. This lets you access memory locations that may not have explicit labels.

arrayB BYTE 10h,20h,30h,40h,50h

If we use MOV with arrayB as the source operand, we automatically move the first byte in the array:

mov al,arrayB ; AL = 10h

We can access the second byte in the array by adding 1 to the offset of arrayB:

mov al,[arrayB+1] ; AL = 20h

The third byte is accessed by adding 2:

mov al,[arrayB+2] ; AL = 30h

The brackets are not required by MASM, so the following statements are equivalent:

mov al,[arrayB+1]

mov al,arrayB+1

Word and Doubleword Arrays:

In an array of 16-bit words, the offset of each array element is 2 bytes beyond the previous one. That is why we add 2 to ArrayW in the next example to reach the second element:

.data

arrayW WORD 100h,200h,300h

.code

mov ax,arrayW ; AX = 100h

mov ax,[arrayW+2] ; AX = 200h

movzx cx, bl cx=009bh

movzx sx, bl cx=009bh

xchg val2 ,ax val2 =1000h

Similarly, the second element in a doubleword array is 4 bytes beyond the first one:

.data

arrayD DWORD 10000h,20000h

.code

mov eax,arrayD ; EAX = 10000h

mov eax,[arrayD+4] ; EAX = 20000h

Note: The MOV instruction never affects the flags

Q.2

Write down the values of destination operands and flags after the execution of each instruction …….

ANSWER:

Mov cx ,1

Sub cx,1; (a) cx = 0zf = 11

Mov cx,0

Sub,cx,1; (b) cx = -1sf =1

Mov al ,0ffh

Add al, 1; (c) al = 00 cf =1

Mov al ,0

Sub al , 1; (d) al =ff0f=1

Mov al ,7fh

Add al , 1;(e) al =80 of =1

Mov al, -128

Neg al; (f)cf=1 of=1

Q.3

What will be the value of EAX after each of the following instruction execute?

ANSWER:

.data

myBytes BYTE 10h,20h,30h,40h

myWords WORD 3 DUP(?),2000h

myString BYTE "ABCDE"

mov eax,TYPE myBytes ; a. 1

mov eax,LENGTHOF myBytes ; b. 4

mov eax,SIZEOF myBytes ; c. 4

mov eax,TYPE myWords ; d. 2

mov eax,LENGTHOF myWords ; e. 4

mov eax,SIZEOF myWords ; f. 8

mov eax,SIZEOF myString ; g. 5

Q_

ANSWER :

.DATA

Val 32 LABEL DWORD

VAR B   BYTE   78H ,56H,34H,,12H

VAL8 LABEL BYTE

VAR D DWORD 12345678H

.CODE

MOV BL ,BYTE PTR VAR D: (a) BL =78h

Mov eax , DOWRD PTR VAR B ; (B)

Eax= 7856….h

Mov al , val8; ?(c) AL =78H

MOV EAX , VAL32 ; (D) EAX =12345678h

Q.5

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

Answer :

myBytes BYTE 10h,20h,30h,40h

myWords WORD 8Ah,3Bh,72h,44h,66h

myDoubles DWORD 1,2,3,4,5

.code

myPointer DWORD myDoubles

mov esi,OFFSET myBytes

```
mov al,[esi]                    ; a. AL = 10h
mov al,[esi+3]                  ; b. AL = 40h
mov esi,OFFSET myWords + 2
mov ax,[esi]                    ; c. AX = 003Bh
mov edi,8
mov edx,[myDoubles + edi]       ; d. EDX = 3
mov edx,myDoubles[edi]          ; e. EDX = 3
mov ebx,myPointer
mov eax,[ebx+4]                 ; f. EAX = 2
```

Q.6

Write assembly language code for each of the following:

(a) Convert the character in AL to upper case.

(b) Convert a binary decimal byte into its equivalent ASCII decimal digit.

(c) Jump to label L1 if bits 0, 1, and 3 in AL are all set

ANSWER :

a) :

```
b) READ MACRO MSG
c)     MOV AH,0AH
d)     LEA DX,MSG
```

e) DATA SEGMENT
MSG1 DB 10,13,'ENTER ANY STRING :- $'
MSG2 DB 10,13,'CONVERTED STRING IS : $'

f) P1 LABEL BYTE
M1 DB 0FFH
L1 DB ?
P11 DB 0FFH DUP ('$')
DATA ENDS

g) DISPLAY MACRO MSG
MOV AH,9
LEA DX,MSG
INT 21H
ENDM

h) CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
MOV AX,DATA
MOV DS,AX

i) DISPLAY MSG1

j) LEA DX,P1
MOV AH,0AH
INT 21H

k) DISPLAY MSG2

l) LEA SI,P11

m) MOV CL,L1
MOV CH,0
CHECK:
CMP [SI],61H
JB DONE

n) CMP [SI],5BH

o) UPR: SUB [SI],20H

p) DONE: INC SI
LOOP CHECK

q) DISPLAY P11

r) MOV AH,4CH
INT 21H
CODE ENDS

s) END START

.....

b) :

- Solution: Use the OR instruction to set bits 4 and 5.

```
mov al,6              ; AL = 00000110b
or  al,00110000b      ; AL = 00110110b
```

The ASCII digit '6' = 00110110b

……

C) :

• Solution: Clear all bits except bits 0, 1,and 3. Then

compare the result with 00001011 binary

```
and al,00001011b          ;clear unwanted bits
cmp al,00001011b          ;check remaining bits
je L1                     ;all set? jump to L1
```

Q_7

Answer:

A)

```
Mov eax , var1

Cmp eax , var 2

Jle  L1

MOV VAR3 ,110

MOV VAR4 , 90

JMP L2

L1: MOV VAR 3 , 128

…….
```

B)

```
CMP VAL1 ,ECX

Jna L1 CMP ECX , EDX

Jna L1

Mov x ,30

JMP NEXT

L1: MOV X, 40

NEXT :

……
```

C)

```
Top : cmp eax ,ebx : check loop condition

Jae next                    ; false ? exit loop

Jnc eax                     ; body of loop

Jmp top                     ;repeat the loop
```

Next :

Q.8

(a) Write a sequence of statements that use only PUSH and POP instructions to exchange the values in the EAX and EBX registers.

(b) Write a program with a loop and indirect addressing that copies a string from source to target, reversing the character order in the process. Use the following variables: source BYTE " This is the source string ", 0 target BYTE SIZEOF source DUP ('#')

(c) Write a program that displays a string in all possible combinations of foreground and background colors (16 x 16 =256). The colors are numbered from 0 to 15, so you can use a nested loop to generate all possible combinations. Also use a delay of 1s in each foreground color change

ANSWER :

A) :

```
push ebx
push eax
pop ebx
pop eax
```

……

B) :

```
.386
.model flat,stdcall
.stack 4096
ExitProcess PROTO, dwExitCode:DWORD

   .data
   source BYTE "This is the source string",0
   target BYTE SIZEOF source DUP('#')

   .code
   main PROC
      mov esi,0
      mov edi,LENGTHOF source - 1
      mov ecx,SIZEOF source

   L1:
      mov eax, 0
```

```
        mov al,source[esi]
        mov target[edi],al
        inc esi
        dec edi
        loop L1

    INVOKE ExitProcess,0
    main ENDP
    END main
```

…….

C) :

```
INCLUDE Irvine32.inc

.data
count DWORD ?

.code
main PROC

   mov eax, 0 + (0 * 16)
   mov ecx, 16
   L1:
      mov count, ecx
      push eax
      mov ecx, 16
      L2:
         call SetTextColor
         push eax
         mov al,'H'
         call WriteChar
         pop eax
         inc eax
      LOOP L2
      call crlf
      pop eax
      add eax, 16
      mov ecx, count
   LOOP L1

   call crlf
   call WaitMsg
   exit
main ENDP
```

END main

The end