ID NO 14306

SEMESTER 5th

SUBJECT ASSEMBLY LANGUE

ASSIGNMENT NO 6

SUBMITTED TO SIR AMIN

ASSIGNMENT NO (6)

Q1 write a Single instruction 16-bit operands that Clears the heigh 8bits of AX and does not Change the low 8 bits.

Ans AND AX, 00FFh

Q2 write a Single instruction 16-bit operands that Set the heigh 8bits of AX and does not Change the low 8 bits.

Ans OR ax, 0FF00h

Q3 write a Single instruction (other than NOT) that reverses all the bits in EAX.

Ans XOR eax, 0FFFFFFFFh

Q4 write instruction that Set the ZF if the 32-bit value in EAX is even and Clear the

ZF if EAX is odd.

Ans Test eax,1 ; (low bit set if eax is odd.

Q5 write a single instruction that coverts an uppercase character in AL to lowercase but does not modify AL if it already contains a lowercase letter.

Ans OR al, 00100000b

Q6 which jump instruction follow unsigned integer comparsion?

Ans ~~BX = 006Bh~~ JA ; JNBE, JAE, JNB, JB, JNAE, JBE, JNA.

Q7 which jump instruction follow signed integer comparsion?

Ans JG, JNLE, JGE, JNL, JL, JNGE JLE, JNG

Q8 will the following code jump to the label named Target?

```
mov ax, 8109h
Cmp ax, 26h
Jg Target
```

Ans NO will the following code jump to the label named Target or NO. because the jg is used with Signed Value and 8109h is negative, and 26h is positive).

Q9 Implement the following Pseudocode in assembly language:

a. if ebx > ecx
   X = 1

b. if edx <= ecx
   X = 1
   else
   X = 2

Page ④

a) if ebx > ecx
X = 1

```
Cmp ebx, ecx
jna next
mov X,1
next:
```

b) if edx <= ecx
X = 1
else
X = 2

```
Cmp edx, ecx
jnbe : else
mov X,1
jump next
else : move X,2
next:
```

Q10 what will be the value of BX after the following instruction execute:

```
mov bx, 0FFFFh
and bx, 6Bh
```

Ans BX = 006Bh

Q11 wat will be the value of BX after the following instruction execte?

```
mov bx, 91BAh
and bx, 92h
```

Ans BX = 0092h

Q12 what will be the value of BX after the following instruction execute?

```
mov bx, 0649Bh
or bx, 3Ah
```

Ans BX = 064BBh

Q13 what will be the value of BX after the following instruction execute?

```
mov bx, 029D6h
xor bx, 8181h
```

Ans BX = A857h

Q14 what will be the value of EBX after the following instructions execute?

```
mov ebx, 0AFAF649Bh
or  ebx, 3A219604h
```

Ans EBX = BFAFF69Fh

Q15 What will be the value of RBX after the following instruction execute?

```
mov rbx, 0AFAF649Bh
xor rbx, 0FFFFFFFFh
```

Ans RBX = 00000000505096B64h

Q16 In the following in the following instruction sequence. Show the resulting value of

AL where indicated, in binary:

```
• mov al, 0110111b
• and al, 00101101b ; a.
• mov al, 6Dh
• and al, 4Ah       ; b.
• mov al, 00001111b
• or  al, 61h       ; C.
• mov al, 94h
• Xor al, 37h       ; D.
```

Ans a=00101101 b=01001000
C=01101111 d=10100011

Q17 In the following instruction Sequence. Show the resulting value of AL where indicated, in hexadecimal:

```
mov al, 7Ah
not al,      ; a.
mov al, 3Dh
and al, 74h  ; b.
mov al, 9Bh
```

```
or  al, 35h    ; c.
mov al, 72h
X or al, 0DCh ; d.
```

Ans Al = 85h, 34h, BFh, AEh

Q18 In the following instruction sequence, Show the value of the Carry, Zero, and Sign flags where indicated.

```
mov al, 00001111b
test al, 00000010b; a. CF=
ZF =  SF =
mov al, 00000110b
Cmp al, 00000101b; b. CF=
ZF= SF=
mov al, 00000101b
Cmp al, 00000111b; c. CF=
ZF= SF=
```

Ans a. CF = 0 ZF = 0 SF = 0
b. CF = 0 ZF = 0 SF = 0
c. CF = 1 ZF = 0 SF = 1

Q19 which Conditional jump instruction executes a branch based on the contents of ECX?

Ans JECX

Q20 How are JA and JNBE affected by the Zero and Carry flags?

Ans JA / JNBE (Jump if above / jump if not below or equal)

Condition for jump:-

CF = 0 and ZF = 0

The JA / JNBE Conditional jump instruction will cause Program execution to transfer to another location in a rang from +127 bytes to -128 bytes from the instruction following the jump instruction if CF=0 and if ZF=0 (both must be 0). if this condition is not true no jump occurs. when

Used after CMP. this instruction is referring to the unsigned values of the operands used by the CMP instruction. DEBUG Notes:
Regardless of which mnemonic is used during assembly, DEBUG always disassembles this OP code as JA.
[Flag affected - none].

Q2: what will be the final value in EDX after this code executes?

```
mov  edx,1
mov  eax,7FFFh
Cmp  eax,8000h
jl   L1
mov  edx,0
L1:
```

Ans: The final value of EdX=

10000000, 00000000, 01111111
11111111 00000000 00000000
10000000 00000000 = 256 M.

Q22 What will be the final value in EDX after this code executes?

```
mov edx,1
mov eax,7FFFh
Cmp eax,8000h
jb L1
mov edx,0
L1:
```

Ans Operations on eax do not directly affect the value of edx but since it has been initialized to 1 and the zeroing depend on the outcome of an operation on eax. it is affected indirectly.

jb is an unsigned operation and does what you said. Note

that 7FFFh is below 8000h
So the jump will be taken,
there by skipping the mov edx
with 0 Thus, the final value
in edx will be (1).

Q23 What will be the final value in EDX after this code executes?

```
mov edx,1
mov eax, 7EFF8000h
Cmp eax, 0FFFF8000h
jl L2
mov edx,0
L2:
```

Ans The final value of EDX =
00000000 00000000 01111111
11111111 1111 1111 1111 1111
1 0000000 00000000

Eax = 016.

Q24 will the following code jump to the label named Target?

Ans Yes: will the following code jump to the label named Target?

```
mov ax, -30
Cmp ax, -50
jg Target
```

Q25 will the following code jump to the label named Target?

Ans Yes. will the following code jump to the label named Target?

```
mov ax, -42
Cmp ax, 26
jg Target
```

Q26 write a single instruction that converts an ASCII digit in AL to its corresponding binary value. If AL already contains

a binary value (00h to 09h), leave it unchanged.

Ans And al, 00001111b

Q27 write instructions that calculate the parity of a 32-bit memory operand.

Ans Use the formula presented earlier:

B0 XOR B1 XOR B2 XOR B3.

```
.data
memval DWORD ?
.code
mov al, BYTE PTR memval
Xor al, BYTE PTR memval+1
Xor al, BYTE PTR memval+2
xor al, BYTE PTR memval+3
```

Q28 write instruction that jump to label L1 when the

unsigned integer in DX is less than or equal to the integer in CX.

Ans
```
Cmp dX, CX
jbe L1
```

Q29 write instruction that jump to label L2 when the Signed integer in AX is greater that the integer in CX.

Ans
```
Cmp ax, CX
jg L2.
```

Q30 write instruction that first Clear bits 0 and 1 in AL. Then, if the destination operand is equal to zero, the Code Should jump to label L3. Otherwise, it Should jump to label L4.

Ans And al, 11111100b
jz L3
jump L4

Q31 Implement the following pseudocode in assembly language. Use Short-Circuit evaluation.

a) if (val1 > ecx) AND (ecx > edx)
X = 1
else
X = 2;

```
Cmp val1, ecx
jna else
Cmp ecx, edx
jna else
mov X, 1
jmp next
else: mov X, 2
next:
```

Page (17)

b) if (ebx > ecx) OR (ebx > val1)
X = 1
else
X = 2

```
Cmp  ebx , ecx
ja   L1
Cmp  ebx , val1
ja   L1
mov  X, 2
jump next
L1 : mov X, 1
next:
```

C) if (ebx > ecx AND ebx > edx) OR

(edx > eax)
X = 1
else
X = 2

```
Cmp ebx,ecx ; ebx > ecx ?
jna L1      ; no: try Condition after
              OR
Cmp ebx,edx ; yes: is ebx>
              edx?
jna L1.     ; no: try Condition
              after OR
jmp L2.     ; Yes: Set X to 1
;-----------OR (edx > eax)--------

L1: Cmp edx,eax; edx >eax?
jna else    ; no: Set X to 2
L2: mov X,1 ; Yes: Set X to 1
jump next   ; and quit
else: mov x,2 ; Set X to 2
next:       ;
```

Q32 Implement the following Pseudocede in assembly language. Use Short-circuit evalution and assume that A, B, and N are

32-bit Signed integers.

```
While N>0
if N!=3 AND (N<A OR N>B)
 N = N-2
 else
 N = N-1
 end while
```

Ans

```
INCLUDE 32.inc
.data
        N  DWORD  10
        A  DWORD  9
        B  DWORD  8
.Code
  main PRDC

  mov eax, N
  mov ebx, A
  mov ecx, B
Top
```

```
    Cmp  eax,0
    jbe  Next
    Cmp  eax,3
    jne  L1
    jump L4

L1:
    Cmp  eax,bx
    jb   L3
    ja   L2

L2:
    Cmp  eax,ecx
    ja   L3
    jb   L4

L3:
    Sub  eax,2
    jump Top
;
;
```

```
L4:
    Sub eax, 1
    jump top

Next
    INVOKE ExitProcess, 0
main endp
end main
```

Q33 Create a procedure that fills an array of doublewords with N random integers, making sure the values fall within the range j....K, inclusive. when calling the procedure, pass a pointer to the array that will hold the data, pass N, and pass the values of j and K. Preserve all register value between calls to the procedure. write a test program that calls the procedure twice, using

different values for j and K. verify your results using a debugger.

Ans. Solution:-

```
INCLUDE Irvine32.inc
N = 10
.data
array DWORD N DUP(?)
j DWORD ?
IC DWORD ?

.Code
main PROC
Call ClrScr
mov j, -10
mov K, 10
mov ESI, OFFSET array
mov ECX, N
Call Filling An Array
```

```
mov j, 100
mov K, 1000
mov ESI, OFFSET array
mov ECX, N
Call Filling An Array

Call waiting MSG
exit
main ENDP

Filling An Array PROC
push ecx
push esi

h:
mov eax, j
mov ebx, K
dec ebx
Sub ebx, eax: Create range from
              0 to N
Xchg ebx, eax ; random work
                with eax
```

```
Call RandomRange; generate random
                  int within rang 0
                  to N.
neg ebx           ; Change sign of
                    ebx
Sub eax, ebx      ; Sub from eax
                    to define range
Call CrlF
Call Write Int
mov [esi], eax
add esi, 4

Loop h
Pop esi
Pop ecx
ret
filling An Array ENDP

END main
```

Q34 Create a procedure that returns the sum of all array elements falling within the range j....K (inclusive). write a test program that calls the procedure twice, passing a pointer to a signed doubleword array, the size of the array, and the values of j and K. Return the sum in the EAX register and preserve all other register values between calls to the procedure.

Program:-

```
INCLUDE Irvine32.inc
N = 10
.data
array SDWORD N DUP
(-10, -8, -6, -4, -2, -1, 1, 3, 5, 7)

jDWORD ?
KDWORD ?
```

- Code

```
main PROC
Call Clrscr
mov j,0
mov K,10
mov ESI, OFFSET array
mov ECX, N
Call Summing Array Elements
In Range
Call write Int
Call Crlf

mov j, -10
mov K, 0
mov ESI, OFFSET array
mov ECX, N
Call Summing Array Element
in Range
Call Crlf

Call waitMsg
exit
```

```
main ENDP
Summing Array Elements In Range PROC
Push ecx
Push esi
mov eax,0
h:
   mov ebx, [esi]
   Cmp ebx, j
   jge tove 1
   jp  next
tove 1:
      Cmp ebx, K
      jle tove2
      jmp next
tove 2:
     add eax, ebx
next:
      add esi, 4
loop h
     Pop esi
     Pop ecx
```

Yet
Summing Array Element in Range EINDP

End main.

Q35 Data transmission Systems and file subsystems often use a form of error detection that relies on calculating the Parity (even or odd) of blocks of data. Your task is to create a procedure that returns True in the EAX register if the bytes in an array contain even Parity, or false if the Parity is odd. In other words. If you count all the bits in the entire array, their count will be even or odd. Preserve all other registers values .....

Create two arrays containing at least 10 bytes.

Ans Programming :-

1 . 10

```
INCLUDE Irvine 32.inc
.data
byte 1 BYTE 1111110b,
1101110b, 1000110b, 1110100b,
1100100b, 1100101ob , 1100101ob.
1100101ob.
byte 2 BYTE 1111110b.
1101110b, 1000110b, 1110100b.
1100100b, 1100101ob. 1100101ob
1100101ob.
.Code
main PROC
    mov esi ,OFFSET byte 1
    mov ecx, SIZEOF byte 1
    Call PF Check
    Call write int
    exit

main ENDP
```

```
PF Check PROC

; eax PF = 1 True PF = 0 FAISE
; esi , ecx
        push esi
        push ecx
        Sub ecx, 1
        mov al, 0
        Xor al, 0
        mov al, [esi]

L1:
        inc esi
        Xor al, [esi]
        mov bl, [esi]
        loop L1

        jp LPF 1
        mov eax, 0
        jump LEND
```

```
LPF 1:
mov eax, 1
LEND:
pop esi

ret

PF Check ENDP

END main.
```

★ Completed ★