National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Computer Organization and Assembly Language	Course Code:	EE213
Program:	BS(Computer Science)	Semester:	Fall 2018
Duration:	180 Minutes	Total Marks:	100
Paper Date:	31st December, 2018	Weight	45%
Section:	ALL	Page(s):	10
Exam Type:	FINAL		

Student : Name:_____ Roll No._____ Section:_____

Instruction/Notes:

1. Exam is Open book, Open notes.
2. Properly comment your code.
3. Syntax error will result in **negative** marking.
4. Write your answer in the space provided. You **can take extra sheets BUT they WONT BE ATTACHED WITH THE QUESTION PAPER OR MARKED.**

Question1. Answer the following multiple choice questions. **NO cutting or Over Writing. [2x10 Marks]**
For this question, please put your answers in the following table. No answer will be marked other than this table.

S.No	Answer	S.No	Answer
i.	C	**vi.**	C
ii.	A	**vii.**	C
iii.	A	**viii.**	A
iv.	D	**ix.**	A
v.	C	**x.**	C

i. Assume the initial values for registers/flags are: DF=0, SI=0x100, DI=0x200. And the following are the initial memory values at given memory address:

Memory Address	0	1	2	
0x0100	0x10	0x15	0x22	…
0x0200	0x20	0x25	0x21	…

After executing the movsb instruction, the updated values of registers and relevant memory address will be:

a. 0x100 contains 0x20, 0x200 contains 0x20, DI=0x201, and SI=0x101
b. 0x100 contains 0x20, 0x200 contains 0x20, DI=0x202, and SI=0x102
c. 0x100 contains 0x10, 0x200 contains 0x10, DI=0x201, and SI=0x101
d. 0x100 contains 0x10, 0x200 contains 0x10, DI=0x1FF, and SI=0x1FF

ii. The operation of "push ax" is equivalent to:

a. SP ← SP -2
   [SP] ← AX
b. SP ← SP -2
   SP ← AX
c. [SP] ← AX
   SP ← SP -2
d. SP ← SP +2
   [SP] ← AX

iii. The video display memory is divided into 25 rows and 80 columns. Assuming that the rows are numbered from 0 to 24 and columns are numbered from 0 to 79 and es has 0xb800, the instruction "mov word [es:1700], 0x0735" puts the number 5 at which of the following location:

a. Row=10, Column=50
b. Row=9, Column=50
c. Row=10, Column=49
d. Row=9, Column=49

iv. Imagine a sub-routine being called from the main program. The main program passes one argument (parameter) to the sub-routine. How exactly will the subroutine access this parameter?

a. [bp-2]
b. [bp]
c. [bp+2]
d. [bp+4]

v. The instruction that is used as prefix to a string instruction to execute it repeatedly until the CX register becomes zero is:

a. SCAS
b. CMPS
c. REP
d. REPN

vi. Suppose AL=11001011, CL=2 and CF=1. Give the new contents of AL after executing "sar al, cl".

a. AL=11100101, CF=1
b. AL=10010010, CF=1
c. AL=11110010, CF=1
d. AL=00110010, CF=1

vii. Suppose AL=11001011, CL=3 and CF=1. Give the new contents of AL after executing "rcl al, cl".

a. AL=10010111, CF=1
b. AL=00101111, CF=1
c. AL=01011111, CF=0
d. AL=01011111, CF=1

viii. Imagine a sub-routine being called from the main program. The sub-routine declares a local variable to store and process some data. How exactly will the subroutine access this local variable?

a. [bp-2]
b. [bp]
c. [bp+2]
d. [bp+4]

ix. Suppose SP=0x0200, and top of stack = 0x012A. What are the contents of IP and SP after "ret 4" where ret appears in a near procedure.

a. IP = 0x012A, SP=0x206
b. IP = 0x012A, SP=0x1FA
c. IP = 0x0200, SP=0x204
d. IP = 0x012A, SP=0x202

x. Imagine that the DL register contains a number between 0 and 9. The instruction "OR DL,0x30" results in DL to contain:

a. The original number
b. The original number with sign bit reversed
c. Corresponding character between'0' and '9'
d. The original number multiplied with 3

Question2. Write the output of the following short programs. **[4x7 Marks]**
For this question, please put your answers in the following table. No answer will be marked other than this table.

S.No	Answer
i.	Updated Array values after code execution: 10 | 20 | 30 | 40 | 50 | 60
ii.	Value of [num1]: ___45 or 0x2D_____
iii.	Final Value of DX: ___0x0002_____
iv.	Updated Value of Str2: ___neve ro ddo reveN_____
v.	AX= ___0x9ABC_____ BX=__0x9ABC_____ CX=____0x5678_____ SP = __0x00FE_____
vi.	Updated value of DL Register: _____00111011_____
vii.	Value of Seconds: ___Default value___ Reason: __: The value will remain 1 because there is no end of interrupt signal in timer so processor will take it as timer interrupt never ended.

i. Write the updated contents of 'array' (in decimal only) after execution of the following code.

```
[org 0x0100]
        push ds
        pop es
        std
        mov si, array+8
        mov di, array+10
        mov cx, 3
        rep movsw
        mov word [di], 30
        mov ax, 0x4c00
        int 0x21
array:                  dw 10, 20, 40, 50, 60, 0
```

ii. What's the final value of the num1 memory location? Answer in hex or decimal.

```
[org 0x0100]
        mov dx, 0x0730
        mov cx, 10
        xor bx, bx
loop1:  dec cx
        mov ax, cx
        add dx, ax
        and dl, 0x0F
        add bl, dl
        mov dx, 0x0730
        or cx, cx
        jnz loop1
        mov word [num1], bx
        mov ax, 0x4c00
        int 0x21
num1:   dw 0
```

iii. What is the final value of DX register (in hex) after executing the following code?

```
[org 0x0100]
        mov si, str1
        mov di, str2
        push ds
        pop es
        cld
        mov     cx, [len]
        xor dx, dx
again:  jcxz done
        repne  cmpsb
        jne     done
        inc     dx
        jmp again
done:   mov ax, 0x4c00
        int 0x21

len:    dw 8
str1:   db 'Ukraine.'
str2:   db 'Romania.'
```

iv. Write the contents of "str2" after executing the following program.

```
[org 0x0100]
        mov si, str1
        mov di, str2
        mov cx, [len]
        add di, cx
        dec di
        push ds
        pop es
        cld
next:   movsb
        sub di, 2
        loop next

        mov ax, 0x4c00
        int 0x21

len:    dw 17
str1:   db 'Never odd or even'
str2:   db 'Final Examination'
```

v. Suppose that AX = 0x1234, BX = 0x5678, and CX = 0x9ABC, and SP=0x0100. Give the contents of AX, BX, CX, and SP after executing the following instructions:

```
push ax
push bx
xchg ax, cx
pop cx
push ax
pop bx
```

vi. How does the following code snippet update the value of DL register? Assume that the initial value of DL register is 11011100.

```
        mov cx, 8
tag1:   shl dl,1
        rcr bl,1
        loop tag1
        mov dl,bl
```

vii. Given the following piece of code, what will be the value of 'seconds' if we keep space bar pressed for 7 seconds? Assume that keyboard and timer ISR are correctly hooked in the 'main'. Give reason for your answer in maximum two lines.

```
kbisr:      push ax
            in al, 0x60             ; read char from keyboard port
            cmp al, 0x4B            ; assume that 0x5B is scancode of 'space bar'
            jne nextcmp
            cmp word[cs:timerflag], 1
            je exit
            mov word[cs:timerflag], 1
            jmp exit
nextcmp:    cmp al, 0xBB
            jne nomatch
            mov word[cs:timerflag],0
exit:       mov al, 0x20
            out 0x20, al
            pop ax
            iret
; timer ISR
timer:      push ax
            cmp word[cs:timerflag], 1
            jne skipall
            inc word[cs:seconds]
            push word[cs:seconds]
            call printnum           ; print tick count
skipall:

            pop ax
            iret
```

Question3. Write an isr for software interrupt 0x17, which calculates the run time of any subroutine that the isr invokes/executes in behalf of the user. The label of the subroutine is passed in bx register to the isr. Also ntoe that the time resolution for calculating run time is 1 ms (i.e. the run time will be reported in multiples of 1 ms, which means 1 ms, 2 ms, 3 ms and so on.) You are also required to provide the main /start subroutine. **[25 Marks]**

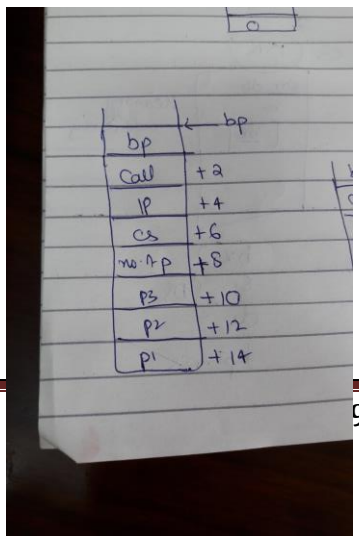
```
[org 0x100]

jmp start
tickcount: dw 0


ISR17:
pusha

mov word[tickcount],0
sti
far call [cs:bx]
mov ax, [tickcount]     ; retrieve in ax register  // or do cli
popa
iret
```

```
timer:
                    push ax
                    inc word[tickcount]

                    mov al, 0x20
                    out 0x20, al            ; end of interrupt
                    pop ax
iret
start:
                    xor ax, ax
                    mov es, ax              ; point es to IVT base

                    mov word [es:17*4], ISR17
                    mov [es:17*4+2], cs     ; hook timer interrupt
;change timer speed here to make milisecond
                    mov ax, 1100
                    out 0x40, al
                    mov al, ah
                    out 0x40, al


                    cli
                    mov word [es:8*4], timer
                    mov [es:8*4+2], cs      ; hook timer interrupt
                    sti

                    int 17

mov ax, 0x4c00
int 21h
```

Question4.

i. Consider the elaborate multitasking example 10.2. In this example, each new task created takes one parameter on its stack. Suppose we want to change the intipcb such that, it generalizes the creation of mytask. So if mytask takes one parameter, the stack created for it has one parameter, if it takes zero parameter the stack has zero parameter, if it takes two parameters, the stack created has two parameters and so on. Suggest all changes that you will make to initipcb code (line 88 to 117) and to main (line 204 to 210). **[8 Marks]**

```
102.    mov ax, [bp+6]
103.    mov [pcb+bx+18], ax     ;reading cs
104.    mov ax, [bp+4]
105.    mov [pcb+bx+16], ax     ;reading ip
106.
107.    mov [pcb+bx+16], ds
108.    mov si, [nextpcb] ; read this pcb index
109.    mov cl, 9
110.    shl si, cl ; multiply by 512
111.    add si, 256*2+stack
```



```
mov di, 8
mov cx, [bp+8]    ;reading total number of parameters
l1:
cmp cx, 0
je out
add di, 2
mov ax, [bp+di]
sub si, 2
mov [si], ax
sub cx, 1
jmp l1
out:
--------------------------------------------------------------------------
(In main, from line 205 to 209)
push p1
push p2
push p3
push 3     ;e.g. number of parameters
push cs
push IP
call initpcb
```

ii. Which of the following is correct implementation if pcb size is increased by two bytes? **[2 Marks]**

a.
```
mov cl, 6
shl bx, cl
sub bx, 2
```

b.
```
mov cl, 6
shl bx, cl
sub bx, 4
```

c.
```
mov cl, 6
shl bx, cl
add bx, 2
```

d. none of these

iii. Suppose that in the multitasking problem of 10.2, each pcb keeps a track of the number of times it goes into execution in one second. What changes will you make in the code to keep a track of this? **[9 Marks]**

```
134.  timer:  push ds
135.          push bx
136.
137.          push cs
138.          pop ds

        inc byte[tickcount]            ;add this code after line 138
        cmp byte[tickcount], 18        ;if 1 sec has elapsed, reinitialize to zero
        jne cont
        mov byte[tickcount], 0
        push cx
        mov cx, 1
        mov bx, 32

        l1:
        mov [pcb+bx+30], 0
```

```
        add bx, 32
        shl bx, 5
        cmp cx, [nextpcb]
        jne l1
        pop cx

139.    cont:
.       mov bx, [current]
.
.
178

        add [pcb+bx+30], 1      ;adding 1 to the dummy of new pcb coming in execution
```

Question 5. [3+2 Marks]

i. How many times int 0x01 will come in the following code?

```
pushf                   ; setting TF=1
pop ax
or ax, 0x100
push ax
popf

mov cx, 5
rep stosb
mov ax, 0x4c00
int 21h
```

Solution:

7 times

ii. Int 0x03 is responsible for:

(a) Inserting break point
(b) Removing breakpoint
(c) Removing opcode
(d) Restoring opcode

Best of luck ☺