


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Computer Organization and Assembly Language	Course Code:	EE213
	Program:	BS(Computer Science)	Semester:	Spring 2017
	Duration:	180 Minutes	Total Marks:	60
	Paper Date:	25-05-2017	Weight	45%
	Section:	ALL	Page(s):	3
	Exam Type:	Final		

Student : Name: _____ Roll No. Section: _____

- Instruction/Notes:
- Exam is Open book, Open notes.
 - Properly comment your code.

Question 1: Short Questions. [5+5 Marks] (No partial marking)

- a) What will be the maximum value of AX in the following code? Why?

```

XOR  AX, AX
L1:
INC  AX
AND  AX, 0x0F0F
JMP  L1

```

- b) The last 3 bits of an attribute byte gives us the color (RGB value) of the character printed on a given screen location so a total of $2^3 = 8$ colors are possible. Write a complete code that will print a string of length 7 on the start of the screen such that each character is displayed in a different color on black background. Black color character on black background is not allowed as it cannot be seen.

Question 2: [10 Marks]

A computer maintains a log of the I/O devices attached to it. After every two hours the log is updated, and it is verified, which devices attached to processor are still in use and which devices are not being used. Devices which are not used consecutively for 4 hours, (i.e. two logs) are shut down to save power.

Following is a sample data of 16 devices, where 1 means the device was in use and 0 means device was not in use.

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
First log	1	0	0	1	1	0	0	1	1	1	0	1	1	0	1	0
Second Log	0	1	0	1	0	0	1	0	1	0	1	0	0	1	0	0

Write an assembly program, that determines which of these devices should be shutdown to save power supply. For the above data, define labels in your code as follows:

```

firstLog: db 0x99, 0xDA
secondLog: db 0x52, 0xA4
deviceSD: db

```


(Put the output device numbers in the label: deviceSD, where each device number should be represented in a byte.)

Question 3: [20 Marks]

You are required to implement a music equalizer displayed on the text screen. A buffer of 10 bytes length is defined in the data segment. This buffer holds amplitudes of 10 frequencies where amplitudes range between 0 and 255. This buffer is continuously modified by another program which you need to know nothing about. The equalizer simply displays the values on the bottom of screen as 10 bars where the height of each represent the amplitude value as shown in Figure 1. The maximum amplitude is shown as a bar of 10 characters height. To make it really fun, hook this to the timer interrupt with chaining so that it keeps working even if you are doing something else on the screen. Note: The ASCII Code for a solid block character is 0xDB. You have to write the COMPLETE CODE.

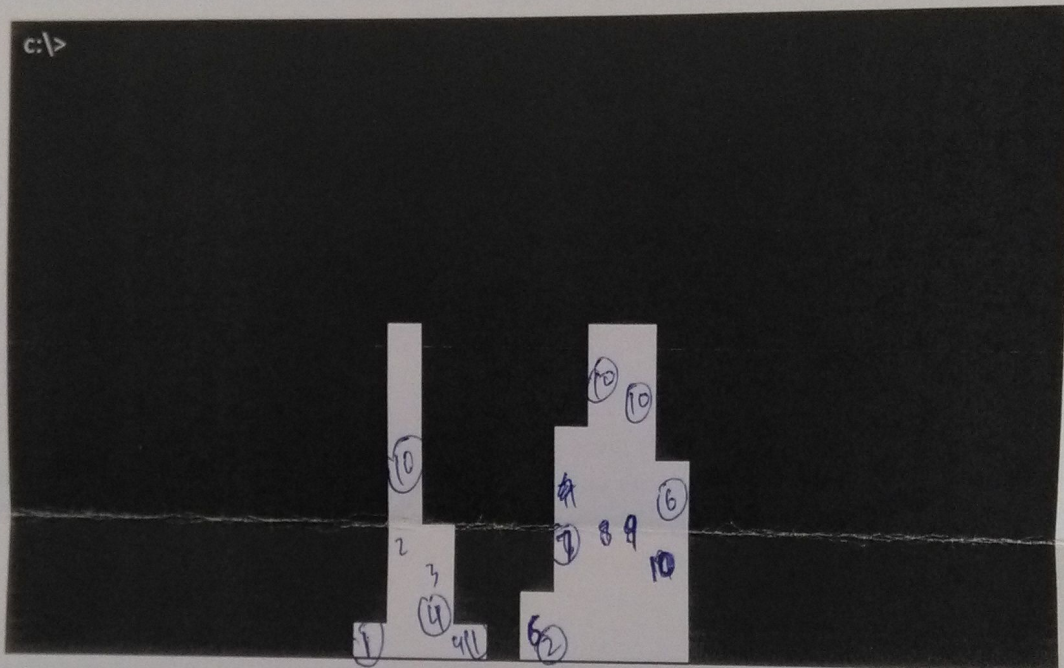


Figure 1 – Illustration of the graphical equalizer displayed on screen bottom centre.

Question 4: [20 Marks]

The following question is about multi-tasking. Read very carefully.

In the example 10.2, multiple instances (threads) of a single subroutine *mytask* are running in a multitasking environment. Now suppose that there are 4 different subroutines named *mytask1*, *mytask2*, *mytask3* and *mytask4* and we want to run multiple threads of these subroutines. The user will press number between 1 and 4 on the keyboard and the corresponding task will be initialized. For example, if user presses 1, *mytask1* pcb should be initialized, if user presses 2, *mytask2* pcb should be initialized and so on. The number of threads will not exceed 32 as in the example. The tasks should run in the sequence initialized 0,1,2,3,4,5,6, till 31 unlike in the example.

- Make all reasonable changes in example 10.2 so the above is achieved. Do not copy the whole code. This will result in zero marks.

- Hook keyboard interrupt to take input of numbers. Make reasonable assumptions for scan codes. No need of chaining. You are not allowed to use int 16h service 0 as in the example.
- Write a subroutine *printProcNames* that prints the names of the subroutines, which are running in the multitasking environment on the screen vertically starting from row 4 column 5, using string instructions **ONLY** like this:

```

mytask1
mytask2
mytask3
mytask4

```

Note that it is not important that all the 4 subroutines are being multitasked together at one time. If none of the subroutines are running, then the subroutine should print nothing on the screen. You can take help from the code of string printing from "string instructions chapter" But you have to write your own (and complete) routine here. You can define an array of subroutines names like this:

Names: db ^{↓ 0} 'mytask1', ¹ 'mytask2', ² 'mytask3', ³ 'mytask4'

si = 0
 GOOD LUCK

l1 db:

mov ~~esi~~ ^{esi = 6800}
 dbi, [Names + si]
 si, 1