National University of Computer and Emerging Sciences, Lahore Campus

Course:	**COAL**	**Course Code:**	**EE2003**
Program:	**BS(CS)**	**Semester:**	**Fall 2022**
Duration:	**3 Hours**	**Total Marks:**	**90**
Paper Date:	**13-01-2023**	**Page(s):**	**11**
Section:	**All**	**Roll No. Your Section:**	
Exam:	**Final (Retake)**		

Instruction/Notes: **This is an open notes/book exam. Sharing notes and calculators is NOT ALLOWED.** All the answers should be written in provided space on this paper. Rough sheets can be used but will not be collected and checked. In case of any ambiguity, make reasonable assumptions. Questions during exams are not allowed.

Question 1 [Pipelining] [CLO 6] [10 Marks]:

Let us consider the following decomposition of the instruction processing
Fetch Instruction (FI): Read the next expected instruction into a buffer.
Decode Instruction (DI): Determine the opcode and the operand specifiers.
Fetch Operands (FO): Calculate the effective address of each source operand and fetch each operand from the memory. Operand in registers need not to be fetched.
Execute Instruction (EI): Perform the indicated operation and store the result if any, in the specified destination operand location.
Write Operand (WO): Store the result in memory.
Following is a set of instructions. Their implementation through pipelining has some data hazards. You have to solve those hazards by using stalling method.

Set of instructions
I1: mov bx, 0
I2: mov word [n1], ax
I3: add word [n1], bx
I4: add word [n1], 01
I5: mov cx, 0
I6: mov word [n2], cx
I7: add bx, [n1]
I8: add bx, word[n2]

Do it with Data Forwarding

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

Question 2 [Cache] [CLO 6] [5+5 = 10 Marks]:

Consider a sequence of memory address references given below. In the sequence, each word address is provided in both the decimal and binary formats. Below each address, the relative time at which these references occur is also listed. Memory contents and addresses are shown in the second table.

Memory		
Decimal Address	**Binary Address**	**Data**
0	00 00 00 00	15
120	01 11 10 00	100
248	11 11 10 00	87
170	10 10 10 10	510
187	10 11 10 11	52
51	00 11 00 11	80
15	00 00 11 11	41
174	10 10 11 10	32
150	10 01 01 10	707
9	00 00 10 01	5
4	00 00 01 00	98
253	11 11 11 01	24
1	00 00 00 01	23
7	00 00 01 11	65
6	00 00 0 1 10	776
2	00 00 00 10	505

Memory Access Sequence		
Time	**Address Decimal**	**Address Binary**
1	6	00 00 0 1 10
2	0	00 00 00 00
3	15	00 00 11 11
4	120	01 11 10 00
5	253	11 11 11 01
6	1	00 00 00 01
7	248	11 11 10 00
8	9	00 00 10 01
9	4	00 00 01 00
10	51	00 11 00 11
11	2	00 00 00 10
12	1	00 00 00 01

Now consider two different 8-word caches shown below. Assume that each of the caches was used independently to facilitate memory access for the sequence above. For each cache type, assume that the cache is initially empty. Assume that the least-recently used (LRU) scheme is used where appropriate. Also, when inserting an element into the cache, if there are multiple empty slots for one index, you should insert the new element into the left-most slot (first available slot).

Part (A) [5 Marks]: Use the **direct-mapped cache** to facilitate memory access for the memory sequence above. You should fill in the binary form of the Tag values. **Show the final contents of the cache in the table below.**
Note: V means Valid OR Value Bit.

Index	TAG	DATA
0		
1		
2		
3		
4		
5		
6		
7		

Hit Rate: _____

Miss Rate: _____

Part (B) [5 Marks]: Use the **2-way set associative cache** to facilitate memory access for the memory sequence above. You should fill in the binary form of the Tag values. **Show the final contents of the cache in the table below.**

	Tag	Data	Tag	Data
0				
1				
2				
3				

Hit Rate: _____

Miss Rate: _____

Question 3 [Performance] [CLO 6] [10 Marks]: It takes 14µs to complete one instruction in a non-pipelined processor. We were able to convert the circuit to a 6 stage pipeline processor. Stage 1 to 6 take 2µs, 1.5µs, 3µs, 3µs, 1.5µs, 3µs resp. No time will be taken to move from one pipe stage to another.

Calculate the following values for pipeline and non-pipelined processor (Write the answer in the given table)

Value	Non-Pipeline	Pipeline
Clock Cycle		
Clock Speed		
Latency		
Throughput for 46 instructions		
Throughput for 1 instruction		
Speedup of pipeline processor for 1 instruction		
Speedup of pipeline processor for 75 instructions		

Question 4 [Short Questions] [CLO 1,2,3,4,5] [5x6 = 30 Marks]:

a. **What will be the value of ax in hexadecimal after the execution of the following piece of code? Show your working to get full marks.**

```
 [org 0x100]
mov al, [num1]
mov ah, [num2+1]
add al, ah
num1: dw 0x12F4
num2: dw 0x00FF
```

```
; Write AX value here
```

b. Write code to call interrupt 15 h **without using int** instruction.

c. The following program is trying to add the first four numbers in the array num1 and store the sum in sum label. Identify mistakes in program if any.

```
; a program toadd four numbers

[org 0x0100]
mov bx, num1
mov si, 0
mov cx, 0
mov di, 0
l1:add word[sum], [bx+si]
add si, 1
sub cx, 1
jnz l1
mov ax, 0x4c00
int 0x21
sum: dw 0
num1: dw 0, 10, 15, 5
```

; Write only mistakes if any. Don't re-write full code.

d. If AX=9FFFh and BX=2FFFh and "cmp ax, bx" is executed, which of the following jumps will be taken? Each part is independent of others. Also give the value of zero flag, sign flag, and carry flag.

a. jg greater

b. jl smaller

c. ja above

d. jb below

ZF: _____ , SF:_____ , CF: _____

a. _____ (taken/not taken)

b. _____ (taken/not taken)

c. _____ (taken/not taken)

d. _____ (taken/not taken)

e. Formula to convert Celsius to Fahrenheit is °F = 1.8°C + 32°. Write a subroutine that it takes Celsius value as parameter, calculate Fahrenheit value in local parameter and return Fahrenheit temperature value in Ax Register. You have to take care of stack structure properly.

Question 5 [CLO 1,2,3,4,5] [30 Marks]: You are required to implement Application according to the functionality as described below:

1- Notepad will have two partitions in the display memory. Upper half (1st 12 rows) will be shape display window while 2nd half (last 12 rows) will be shape blinking window. 13th row will be boundary line like "********". Both the windows will be space with blue background initially. **[4 marks]**
2- Two red colored shapes square (5x5) and rectangle (8*5) blinks in center of shape display window. **[6 marks]**
3- After every 3 second shape changes. For example for first 3 seconds square blinks on the screen and then rectangle blinks on the screen for next 3 seconds. That's how the pattern works. **[6 marks]**
4- When user press enter whatever shape is in the screen should be copied to display window in stable for (non-blinking). **[10 marks]**
5- When user press esc application should end. **[4 marks]**

Important Instructions:

- Credit will be given on code efficiency, so use string instructions where required.
- You may use the functions given in book examples. Give proper reference and use them wisely. Function calls should exactly support the required functionality.

; Write your code here ; Data Declarations (if required) ; and Start/Main Functionality here	;Write Timer Code here (if required)

;Write KBISR here (if required)	;Write KBISR here (if required)

;Other Functions (if required)	;Other Functions (if required)