## National University of Computer and Emerging Sciences, Lahore Campus

STIBHAL UNIVERS
Status Sell
IS IN THE
Seam Tille
SHIDWAY & WITH

Course Name:	Advanced Computer Architecture	Course Code:	EE502
Program:	MS (Computer Science)	Semester:	Fall2020
Duration:	90 Minutes	Total Marks:	30
Paper Date:	24-11-2020	Weight	15
Exam Type:	Midterm II Solution	Page(s):	4

Student: Name: Roll No.

Instruction/Notes:

- 1. Attempt all question in the provided space. You can use rough sheets but it should not be attached.
- 2. If you think some information is missing, write your assumption and solve accordingly.

## Question 1 [3+3+4]

a) What is the effect of pipelining on latency of a single instruction and overall program throughput? Explain for each term, either it increases or decreases, provide reasoning for your answer!

b) Why we need loop unrolling? Suppose a loop has 20 iterations and we want to unroll it by degree 3. In this scenario how many iterations of new loop will be required and how the count of 20 will be completed? Explain your answer with details.

• We need loop unrolling because it reduce our clock-cycles becz in Loop-2 extra instructions are executed in each iteration 3.50, when we do loop unrolling we write these instructions after every 2,3, ...... iterations due to which our clock cycles reduce & our program take less time for completing their execution.

whilen we do loop unrolling of degree 3 & want to execute given loop for 20 iterations then we execute the unrolled loop for 6 times & after going out of this loop we make loop for single iteration so, the remaining 2 iterations are completed by that loop & execute program almost 3-times forst

c) Why Tomasulo is better than scoreboarding? How it provides the extra functionality? What is the purpose of reorder buffers in Tomasulo approach? Explain with details!

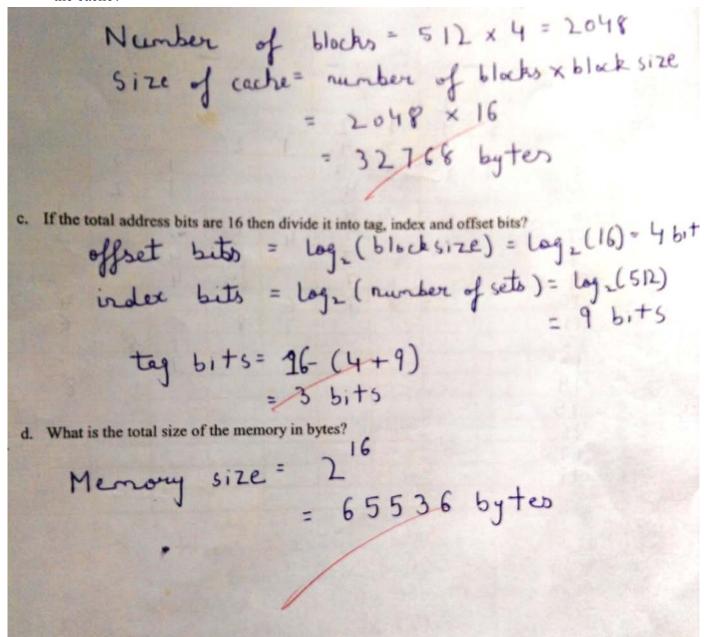
Tomasilo is better than receboarding because it implements the concept of register renaming to remove the false dependencies (WAR, WAW). On the other hand scoreboarding does not provide solution for false dependencies. To mabulo uses Reservation Stations and Buffers to store the values and immediately provide to the Reservation stations that freed the value through a common bus. tomosulo does not store the values of registers but tomasulo does. Purpose of Reorder Buffers: pipelining does not course an issue when precise interupts or exceptions occur. But there is an issue in case of imprecise. Because system has to second what causes exception when. For examples Instruction 2: IF ID EX M WB
Instruction 2: Exception. Here Instruction I should raise exception first but instruct 2 will raise exception first. So, to order the instructions in Tomasulo, we use reader suffers. Instructions commits in RO. when they and preceeding instruction have committed. So, seconde Buff with the help of Header, Identifies the instruction sequence and **Department of Computer Science** if exception occurs, it will flush the instructions of ter Page 2 neader pointer.

## Question 2 [4+2+2+2]

a. While writing in memory, contents can be brought to cache or not depending on the policy being used. Suppose we use **Allocate on write** policy, if following addresses are accessed in the given sequence then for each address tell either it would be hit or miss. Assume cache is initially empty and block size is 16 bytes.

Address	Hit/ miss?
Load 0x1200	miss
Store 0x1000	miss
Load 0x1208	Hit
Load 0x1012	Hit
Load 0x1000	Hit
Store 0x1216	miss

b. If the cache is 4 way set associative with 512 sets and 16 bytes block size, then what is the size of the cache?



## Question 3[10]

Code	Instructions
FOR: ld R1,100(R6) ld R2,200(R6)	<ul> <li>ALU instructions take 1 cycle in each stage except Mem. No cycle is consumed in memory.</li> </ul>
add R5,R2,R1 ld R4,0(R5)	<ul> <li>Beq takes one cycle in each stage and calculation are performed in decode stage.</li> </ul>
addi R6,R6,4 and R4,R5,R4	<ul> <li>Memory type instructions take 2 cycles in memory and one in all other stages.</li> </ul>
st R4,0(R5)	o In case of false dependencies, write back should be in order.
beq R6,R7, FOR	<ul> <li>Instructions are fetched in order however decode, execution, memory and writeback can be out of order if no dependency.</li> </ul>
	<ul> <li>Forwarding is implemented so values can be forwarded as soon as they are produced.</li> </ul>

Show the execution of the above code in the following table having two slots in each stage (2-way superscalar). A slot can be filled in fetch and decode as soon as it is empty.

