National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Computer Architecture	Course Code:	EE204
Program:	BS (Computer Science)	Semester:	Fall 2020
Duration:	90 Minutes	Total Marks:	30
Paper Date:	25-11-2020	Weight	20
Exam Type:	Midterm II	Page(s):	4

Student : Name:_______________________________ Roll No.__________________ Section: ______

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

Question 1a [4]

1. A ***software scheme*** in which ***variable number of instructions*** can be launched **in a single cycle** is called **__VLIW____**.
2. In **___dynamic___** branch prediction, we need to evaluate previous branch results to predict current branch decision.
3. In case of an exception in ***execution stage*** of a pipelined processor, instructions at Mem and WB are flushed.
   a. True
   b. False
4. When compiler attempts to reschedule instructions to avoid hazards, this approach is called dynamic scheduling.
   a. True
   b. False

Question 1b [3]

If we speedup 20% of a program by 5 times. What would be the overall speedup? Explain your answer.

Speedup = $1/\ 0.8 + 0.2/5$

$= 1.19$ that means 19%

20ps * 5 = orig

20/5ps*1 + 20*4 = modified

Question 1c [3]

Specifications of two processors are as follows. Which one will provide higher performance, provide details.

Processor A: CPI = 2, clock = 5 GHz

Processor B: CPI = 1, clock = 3 GHz

$Ex_A = I^* \ 2/5 \times 10^9$

$Ex_B = I^* \ 1/3 \times 10^9$

$Ex_A / Ex_B = 0.4/ 0.33 = 1.2$

That means Performance of B is better than A by 20%.

Question 2 [12]

Consider the following set of instructions

Lw R1, 20(R2)
Add R2, R1, R3
Sub R4, R2, R3
Add R5, R1, R2

Assume that data memory is all zeros and that the processor's registers have the following values before executing the above code

R1	R2	R3	R4	R5
10	10	5	5	10

Note: You have to run the above code statements directly on the processor for the following questions. You are not supposed to rearrange code. Neither you can add any stalls yourself.

1. What will be the register values after executing the above code on a pipelined processor with NO forwarding and data hazard detection unit? [2.5 marks] 0.5 each register

R1	R2	R3	R4	R5
0	15	5	5	10

Mention the values read by instruction in the ID stage [1.5 marks] 0.5 for each of the last 3 rows

Instruction	Register Values Read
Lw R1, 20(R2)	R2 = 10

Add R2, R1, R3	R1 =10
	R3 = 5

Sub R4, R2, R3	R2 = 10 R3 = 5

Add R5, R1, R2	R1 = 0
	R2 = 10

2. What will be the register values after executing the above code on a pipelined processor with forwarding unit? The processor does not have the hazard detection unit. [4 marks]

R1	R2	R3	R4	R5
0	35	5	30	35

Mention the values read by each instruction in the ID stage [2 marks]

Instruction	Register Values Read
Lw R1, 20(R2)	R2 = 10

Add R2, R1, R3	R1 =10
	R3 = 5

Sub R4, R2, R3	R2 = 10 R3 = 5
Add R5, R1, R2	R1 = 0

	R2 = 10

3. What will be the register values after executing the above code on a pipelined processor with forwarding unit and hazard detection unit installed? [4 marks]

R1	R2	R3	R4	R5
0	5	5	0	5

Mention the values read by each instruction in the ID stage [2 marks]

Instruction	Register Values Read
Lw R1, 20(R2)	R2 = 10

Add R2, R1, R3	R1 =10
	R3 = 5

Sub R4, R2, R3	R2 = 10 R3 = 5

Add R5, R1, R2	R1 = 0
	R2 = 10

Question 3 [2+6]

Consider the following instruction sequence:

```
Loop:Add R1, R2, R3
     ld R3, R1, 100
     ld R4, R3, 200
     Sub R3, R4, R3
     St R3, R1, 20
     Add R1, R1, 4
     Beq R1, R5, Loop
```

Assume that the following code is being run on 5-stage mips processor with each stage consuming one clock cycle and full forwarding is implemented.

a. Insert stalls/Nops in the above code where required for its correct execution.

```
Loop:Add R1, R2, R3
     ld R3, R1, 100
     NOP
     ld R4, R3, 200
     NOP
     Sub R3, R4, R3
     St R3, R1, 20
     Add R1, R1, 4
     NOP[Assuming branch hardware moved to ID stage]
     Beq R1, R5, Loop
     NOP
```

b. Consider the above code with stalls and now reschedule the code to remove all stalls (Use unrolling if required to remove all stalls)
Reschedule – 3 marks
Unrolling try – unrolling gives no benefit – 3 marks

```
Loop: Add R1, R2, R3
        ld R3, R1, 100
        NOP
        ld R4, R3, 200
        Add R1, R1, 4
        Sub R3, R4, R3
        St R3, R1, 16
        Beq R1, R5, Loop

Loop:Add R1, R2, R3
        ld R3, R1, 100
        NOP
        ld R4, R3, 200
        Add R6, R1, 4
        Sub R3, R4, R3
        Beq R6, R5, Loop
        St R3, R1, 20
```

One stall remains

Unrolling will not be helpful in this case as R3 cannot be renamed due to the read operation performed in first instruction that needs the value produced in previous iteration. Due to this reason multiple iterations are not independent and instructions cannot be mixed during rescheduling.