


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Computer Architecture	Course Code:	EE204
	Program:	BS(Computer Science)	Semester:	Fall 2018
			Total Marks:	70
	Due Date:	06-11-2018	Weight	~3.3
	Evaluation Type:	Assignment 2	Page(s):	6

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

Question 1 [15]

Consider the following MIPS assembly code to be executed on a pipelined CPU

Loop: ld R2, 20 (R6)

addi R2, R2, 4

add R4, R1, R3

addi R5, R6, 8

sw R2, 30 (R6)

addi R6, R6, 4

bne R6, R7, Loop

Consider the clock cycle time of the CPU is 300 ps. All calculations related to branch are done in decode stage. For each part, stalls should be considered for proper execution of the code.

- What is the total execution time of the above code sequence when pipeline without forwarding is used?
- What is the total execution time of the above code sequence when pipeline with ALU-ALU and MEM-ALU forwarding is used?
- What is the total execution time of the above code sequence when pipeline with full forwarding (ALU-ALU, MEM-ALU and ALU-Decode is used?
- Reschedule the code (for **part c** including stalls) to remove as many stalls as possible

Question 2 [10]

Processor X has a clock speed of 3 GHz, and takes 2 cycle for integer operations, 3 cycles for memory operations, and 5 cycles for floating point operations. Empirical data shows that programs run on Processor X typically are composed of 30% floating point operations, 40% memory operations, and 30% integer operations. You are designing Processor Y, an improvement on Processor X which will run the same programs and you have 2 options to improve the performance:

- Increase the clock speed to 3.2 GHz, but memory operations take 4 cycles
- Decrease the clock speed to 2.5 GHz, but floating point operations only take 4 cycles

Compute the speedup for both options and decide the option Processor Y should take.

Question 3 [15]

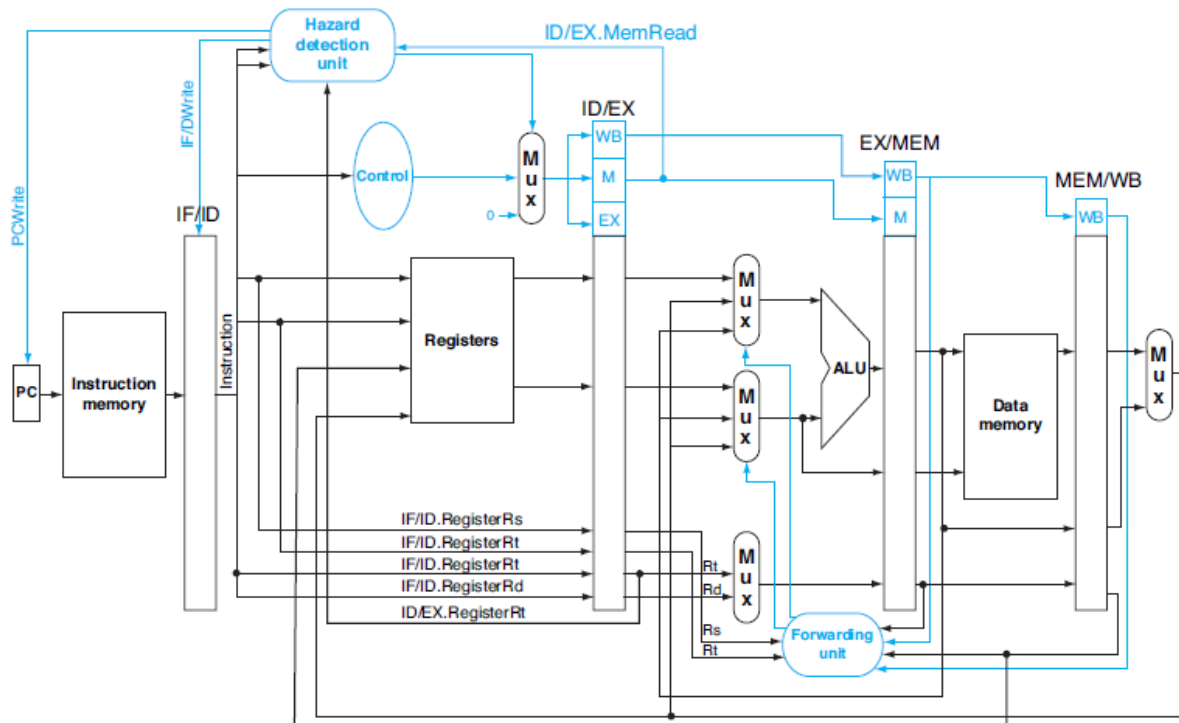
We have studied multi-cycle MIPS as shown below with datapath and control unit. Forwarding unit and Data-hazard detection unit is also implemented.

We know that if branch instruction is dependent on previous instruction then we need to add a stall even though we have EXE to EXE forwarding. Suppose a branch instruction is immediately after an add instruction as below:

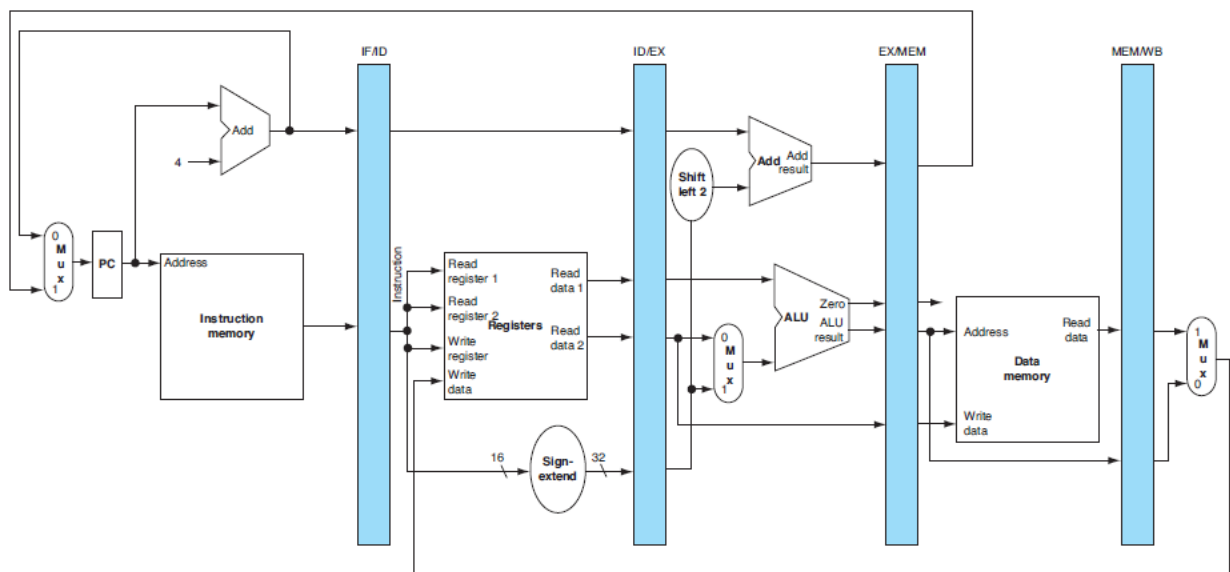
add R1, R2, R5

beq R1, R3, 10

Analyze the situation and modify Hazard detection unit to detect this hazard and then how stall will be inserted between these two instructions. Moreover we need to add another forwarding unit that will forward from Execution to Decode stage. Add this new unit and write code to perform forwarding.



Question 4 [10]



Suppose we have the following **generic architecture with maximum 5 stages**. We want to customize our architecture **where we may not need all stages or all components**.

Latencies of the major elements used in processor design are as follows:

Instruction Memory	Adder	Mux	ALU	Register File	Data Memory	Sign-Extension unit	Shift-Left Unit
250ps	50ps	20ps	120ps	150ps	250ps	20ps	15ps

1. What is the clock cycle time if the only type of instructions we need to support are ALU instructions in a (Arithmetic and logic) Single Cycle processor (Un-pipelined)?
Clock Cycle Time: _____
2. What is the clock cycle time if the only type of instructions we need to support are ALU instructions (Arithmetic and logic) in a Pipelined processor?
Clock Cycle Time: _____
3. What is the clock cycle time if the only type of instructions we need to support are Memory and Branch instructions in a Single Cycle processor (Un-pipelined)?
Clock Cycle Time: _____
4. What is the clock cycle time if the only type of instructions we need to support are Memory and Branch instructions in a Pipelined processor?
Clock Cycle Time: _____
5. What is the total latency of the load instruction (Load word instruction) in pipelined processor?
Clock Cycle Time: _____

Question 5[20]

Consider the execution of the following code snippet on MIPS 5-stage pipelined processor with hazard detection and **forwarding fully implemented**. Static Branch prediction is applied which always assumes branch to be **not taken**.

Instruction 1	Loop: ld R2, 20 (R6)
Instruction 2	addi R2, R2, 4
Instruction 3	add R4, R1, R3
Instruction 4	addi R5, R6, 8
Instruction 5	sw R2, 30 (R6)
Instruction 6	addi R6, R6, 4
Instruction 7	bne R6, R26, Loop

Initial values for all registers is equal to their register number and value stored at each memory location is equal to its memory address.

- a. Consider the execution of the above snippet in the 8th Clock Cycle. Which instructions are executing at each of the pipeline stages? Fill in the following table to answer the question. Write instruction number and if a stall is at some stage then simply write **stall** for that stage.

Stage	Executing Instruction
IF	
ID	
EXE	
MEM	
WB	

- b. In the **start** of the 8th Clock Cycle, what will be the contents of each of the pipeline separation registers? Fill in the following tables to answer the question. You need to explicitly state names and values of **ALL** the variables in each register. For values that are not known to you, you can write them in terms of English phrases. For example, First entry of the IF/ID register has been filled as a guideline. You don't know the exact value of the Program Counter (PC) yet you can state that it is the value of the PC that points to instruction x plus 4 (Here you should replace "x" by the correct instruction number). All other variables, whose values can be determined, should state exact values. For example, if EXE/MEM register contains the result of the addition of instruction 5, you should state the Variable Name as "Result of ALU" and Value as "30"

IF/ID Register Contents

<u>Variable Name</u>	<u>Value</u>
PC	<u>(Program counter value of the instruction x) + 4</u>

EXE/MEM Register Contents

<u>Variable Name</u>	<u>Value</u>

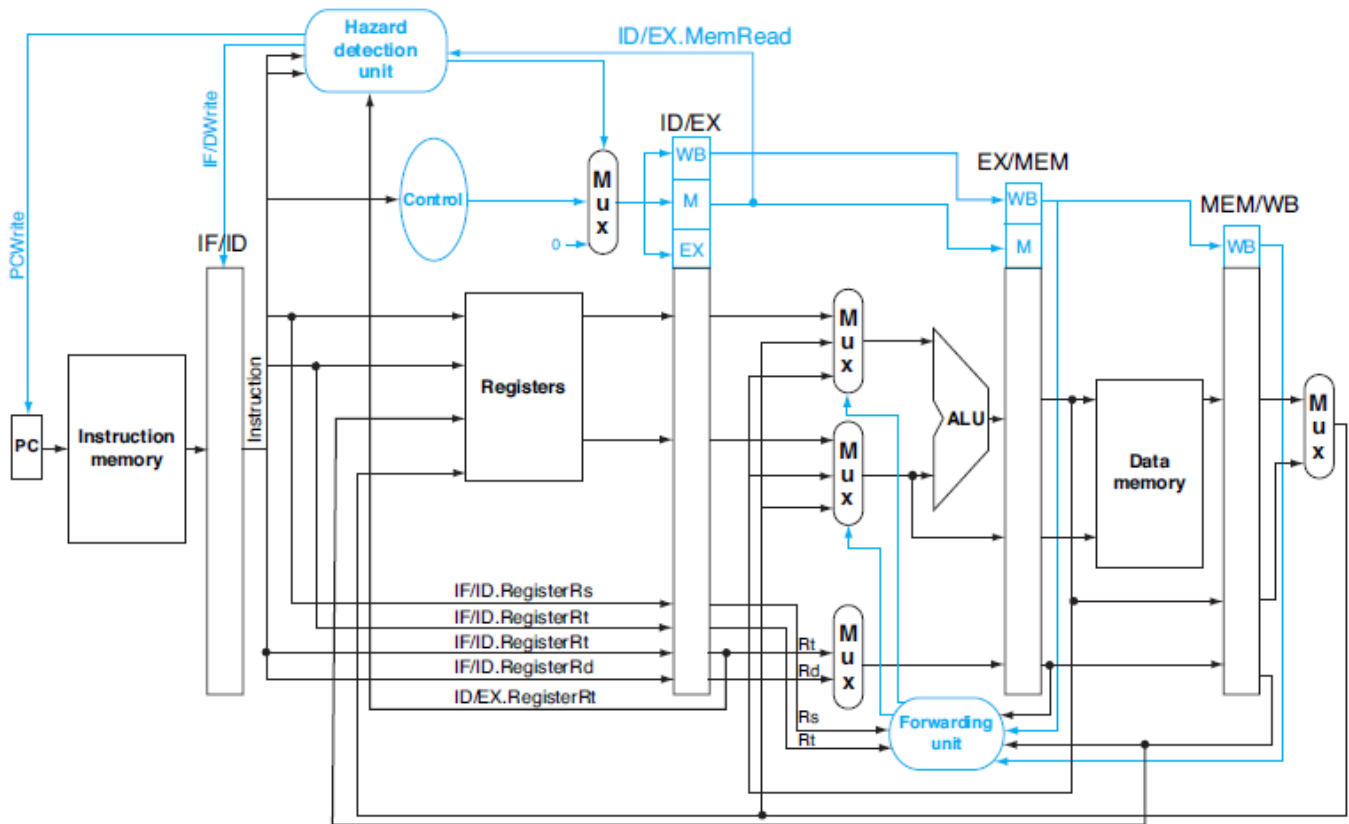
ID/EXE Register Contents

<u>Variable Name</u>	<u>Value</u>

MEM/WB Register Contents

<u>Variable Name</u>	<u>Value</u>

Pipelined Data Path with Forwarding and Hazard Detection Units



Information of control signals

