Project Guidelines

EE 371L/CS 330L/CE 321L Computer Architecture Lab

Maximum Members per Group	3
Weightage	28 % of the lab grade
Submission Deadline	Due in Week 15
Evaluation	Viva-based

Objective:

To **build a 5-stage pipelined processor capable of executing any one array sorting algorithm.**

Description:

Basically, you will be converting your single cycle processor to a pipelined one. Normally the instructions you have already implemented should enable you to execute a sorting algorithm program with small additions i.e., you might need to implement the **bgt or blt** instruction, or something similar, so that you know when you'd need to swap two values. This would require small modifications to the circuit.

Here's the recommended course of action:

Part 1. Choose a suitable sorting algorithm. Convert its pseudocode to assembly using RISC-V instruction set and test its working on the assembly code simulator. Now, modify the existing lab 11 single-cycle processor architecture to run the sorting algorithm code on it. Test and verify that it is doing the sorting correctly.

Part 2. Modify this processor to make it a pipelined one (5 stages). Test and run each instruction separately to verify that the pipelined processor version can at least execute one instruction correctly in isolation. Test different types of instructions. Test cases matter!

Part 3. Introduce circuitry to detect hazards (data, control, and structural if needed) and try to handle them in hardware i.e., by forwarding, stalling and flushing the pipeline. If this is done correctly, then your array sorting code should be able to function in its entirety

Part 4. Compare the performance of running the array sorting program on Single Cycle Processor with RISC-V Processor in terms of execution time.

The book Chapter 4 till end of section 4.8, covered in the theory classes, can be referred for implementing the above step by step.

Submission and Evaluation:

1. You will submit all your codes (the Verilog files for modules as well as test benches) task wise
2. You will submit a PDF report containing explanations of how you implemented your processor, test cases and results, performance comparison (part 4), any difficulties you had and how you

overcame them, and any deficiencies in your projects if there are any. Also, add all the codes in appendices.

3. Any marks are conditional on your being able to explain each line of your code to the examiner during a viva. Each member of the group will be evaluated through the viva.

Part-wise Weightage:

Part		Weightage
1	Sorting Algorithm on Single Cycle Processor	25 %
2	Pipelined Processor	30 %
3	Hazard Detection & Sorting Algorithm on Pipelined Processor	30 %
4	Performance comparison	5%
5	Viva	10%
	Total	100 %

Implementation Help for the Pipelined Processor and its Forwarding Unit

We have studied pipeline implementation of a RISC-V processor with data forwarding techniques to overcome data hazards. Implement the pipeline version of RISC-V processor shown in Figure 1. Initialize all the pipeline registers to an appropriate size. The control values for the forwarding multiplexers are shown in Table 1. **For each of the pipeline registers, you can create a separate module.**

Table 1: The control values for forwarding multiplexers

Mux control	Source	Explanation
ForwardA = 00	ID/EX	The first ALU operand comes from the register file.
ForwardA = 10	EX/MEM	The first ALU operand is forwarded from the prior ALU result.
ForwardA = 01	MEM/WB	The first ALU operand is forwarded from data memory or an earlier ALU result.
ForwardB = 00	ID/EX	The second ALU operand comes from the register file.
ForwardB = 10	EX/MEM	The second ALU operand is forwarded from the prior ALU result.
ForwardB = 01	MEM/WB	The second ALU operand is forwarded from data memory or an earlier ALU result.

Refer to the lecture slides in order to understand the behavior of forwarding unit. Verify the functionality of forwarding by introducing data dependencies in R-format instructions. Do not check the dependency of a load instruction result on the next instruction, as the architecture shown in Figure 1 does not support stalling to overcome certain type of data hazard.

For Task 3, you will be modifying this to include stalling and pipeline flushing. After these modifications, you will be able to test the sorting algorithm on pipelined processor.

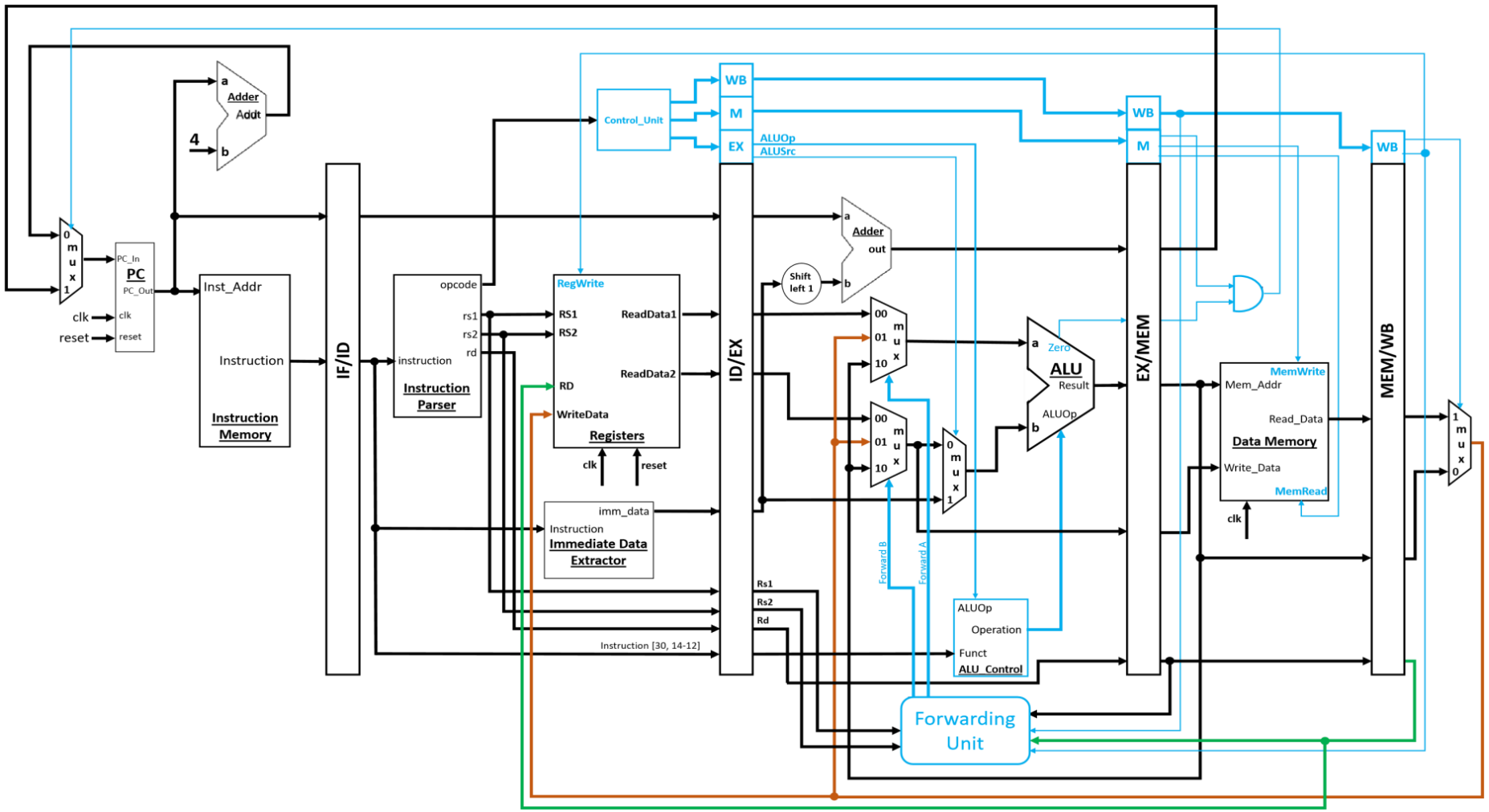

Figure 1: A subset of Pipeline RISC-V processor with forwarding functionality only. Some wires are shown in color just to help a user to distinguish wires easily (specifically between IF/ID and ID/EX stages).

Computer Architecture
Spring 2023
Lab Project Evaluation Rubric

Criteria	CEP	Ratings					Pts
Task 1 (Assembly Code Initialized in Instruction Memory	WP4(EA5)	10 pts Included in report (Code is tested on simulator)	7.5 pts Correct code correctly initialized	5 pts correct code but incorrect initialization	2.5 pts Incorrect Code	0 pts No marks	10 pts
Task 1(Changes in Architecture) Changes in Verilog modules to include the instructions that were not executable		10 pts Required changes explained in report	7.5 pts All required changes are made	5 pts All required changes are not made but most of them are included	2.5 pts Very few changes are accommodated	0 pts No Marks	10 pts
Task 1(Results) Array Sort Testing		5 pts Results /working explained in report. Results are verified	3.75 pts The code works (try to swap element), results are not 100 % correct	2.5 pts Results of sort are tested but are incorrect	1.25 pts Sorting is not Tested but changes are verified by test cases	0 pts No Marks	5 pts
Task 2 (Pipelined Processor] Pipelined Registers are included		15 pts Included in report all connections and modules are correct	11.25 pts All pipeline reg modules are correct but complete connections are not made	7.5 pts Pipelined registers are Mostly correct (few errors)	3.75 pts Incorrect addition of pipelined reg modules	0 pts No Marks	10 pts
Task 2 (Forwarding Unit)		10 pts Explained in report Connections with Forwarding Unit and Forwarding Unit itself is correct	7.5 pts Forwarding Unit is correct (complete logic)	5 pts Forwarding Unit logic and conditions are correct but all conditions are not included	2.5 pts Incorrect Logic	0 pts No Marks	10 pts
Task 2 (Test Cases and Results)		10 pts Explained in report all test cases (valid) with correct results	7.5 pts A few test cases with correct results	5 pts Test case (instruction with forwarding, results are incorrect)	2.5 pts Test case (Few instructions without forwarding)	0 pts No Marks	10 pts
Task 3 (Hazard	WP7	15 pts	11.25 pts	7.5 pt Hazard	3.7 Hazard	0 pts No	10 pts

Detection Unit and Stall)		Included in report Correct Logic and correct connections for Hazard Detection Unit and Pipeline Rush	Hazard Detection Unit is included with correct connections but pipeline flush logic is not accommodated	Detection Unit is created with correct logic	Detection Unit / Flush with Incorrect Logic5 pts	Marks	
Task 3 (Testing with Hazard Detection Unit and Stall)		10 pts Included in report Strong Test cases with correct results	7.5 pts Test Case is Strong but Results are not all correct	5 pts Test Case Checks a few conditions (Stall / Flush)	2.5 pts Test Case is Weak	0 pts No Marks	10 pts
Task 3 (Testing with Sorting Algorithm)		10 pts Sorting works, explained and included in report	7.5 pts Tried to swap a few elements	5 pts No swap at all	2.5 pts No idea how to check results	0 pts No Marks	10 pts
Task 4 Performance Comparison	**WP1**	5 pts Comprehensive analysis with complete justification of the difference in performances	3.75 pts The analysis is done with partially justified analysis	2.5 pts The analysis is done without any justification	1.25 pts The analysis is incomplete and there is not justification	0 pts No Marks	5 pts
Viva		10 pts Level 4	8 pts Level 3	6 pts Level 2	2 pts Level 1	0 pts No marks	10 pts
							Total Points:100

Complex Engineering Problem

This project is a CEP and includes the following Washington Accord (WA) attributes:

Depth of Knowledge Required	WP1	Cannot be resolved without in-depth engineering knowledge at the level of one or more of WK3, WK4, WK5, WK6 or WK8 which allows a fundamentals-based, first principles of analytical approach
Familiarity of issues	WP4	Involve infrequently encountered issues or novel problems
Interdependence	WP7	Address high level problems with many components or sub-problems that may require a systems approach

Rubric for assessment of Complex Engineering Problem

Domain	WPs		**Unsatisfactory, 0**	**Satisfactory, 1**	**Good, 2**	**Very Good, 3**	**Comprehensive, 4**
Preamble	WP1	1. Problem Definition	Demonstrates an inability in construction of a problem statement that requires complex engineering activities	Demonstrates a limited ability in identifying a problem statement that requires complex engineering activities	Demonstrate the ability to involve some complex engineering activities, but problem statement does not correctly capture the definition	Demonstrates the ability to construct a problem statement with evidence of complex engineering activities, and problem statement is adequately detailed	Demonstrates the ability to construct a clear and insightful problem statement with detailed outlining of complex engineering activities
WP7	WP7	5. Design / detailing of multiple sub-components	Design / Detailing not done	Design / detailing done with sketches, but does not involve multiple sub-components	Design / detailing involves multiple sub-components, and detailing adequately done using unclear sketches / drawings	Design / detailing involves multiple sub-components, and detailing is adequately explained with clear sketches / drawings	Design / detailing involves multiple sub-components, and detailing is perfectly explained with clear sketches / drawings
EA5	WP4	6. Familiarity with approaches	Demonstrates no extension of previous experiences in applying principles-based approaches	Demonstrates limited extension of previous experiences in applying principles-based approaches	Demonstrate adequate extension of previous experiences in applying principles-based approaches	Demonstrate reasonable extension of previous experiences in applying principles-based approaches	Demonstrate extension of previous experiences in applying principles-based approaches