EC413 Lab 6 – Single-Cycle MIPS CPU Design and Enhancement

Prof. Martin Herbordt

1 Overview

The objective of this lab is to deepen your understanding of CPU operations by working with a single-cycle MIPS CPU. Starting with a basic CPU implementation, you will incrementally add, debug, and test several new instructions, enhancing both the control logic and datapath as needed.

2 Lab Tasks

Complete the following tasks. Since modifying parts of the CPU may affect its overall functionality, it's important to test your CPU after each modification.

1. **Simulate the Basic CPU:**
   - Simulate the provided single-cycle CPU design using a predefined instruction sequence in `tb_cpu.v`.
   - Generate outputs and analyze the CPU's behavior.
2. **Add the `SLT` Instruction (Set on Less Than):**
   - Implement the `SLT` instruction.
   - You only need to modify the ALU and ALU control logic to support this instruction. No additional hardware should be necessary beyond these modifications.
3. **Add the `ADDI` Instruction (Add Immediate):**
   - Implement the `ADDI` instruction.
   - Modify the control logic to handle this instruction without adding new hardware.
   - General support for I-type ALU instructions is not required.
4. **Add the `J` Instruction (Jump):**
   - Implement the `J` instruction as described in the textbook.

- Update the control logic and datapath accordingly.
- *Reference:* See Chapter 4, Section 4.4 of *Computer Organization and Design* by Patterson and Hennessy.

5. **Add the `BNE` Instruction (Branch Not Equal):**
   - Implement the `BNE` instruction.
   - Modify the control logic and datapath to support this new branch instruction.
   - Additional hardware components may be required.
6. **Add the `LUI` Instruction (Load Upper Immediate):**
   - Implement the `LUI` instruction.
   - Update the datapath and control logic as necessary.
   - Ensure thorough testing with relevant test cases.
7. **Extra Credit - Implement I-Type ALU Instructions:**
   - Extend your CPU to support I-type versions of ALU instructions (e.g., `ANDI`, `ORI`).
   - Adjust the control logic and datapath to handle these instructions.
   - Provide comprehensive testing to verify functionality.

3 Report Requirements

For each task, provide the following in your lab report:

- **Description:** A detailed explanation of the modifications made to the CPU design, including any changes to the control logic and datapath.
- **Testbenches:** Include the testbenches you created for each instruction, along with simulation results.
- **Diagrams:** If you added hardware components (e.g., for `BNE`, `LUI`), include a modified datapath diagram highlighting the changes. (see Figure 1)

4 Datapath Diagram

This diagram should be a good starting point for the lab. You can modify it to show the hardware you added to support new instructions for the report.

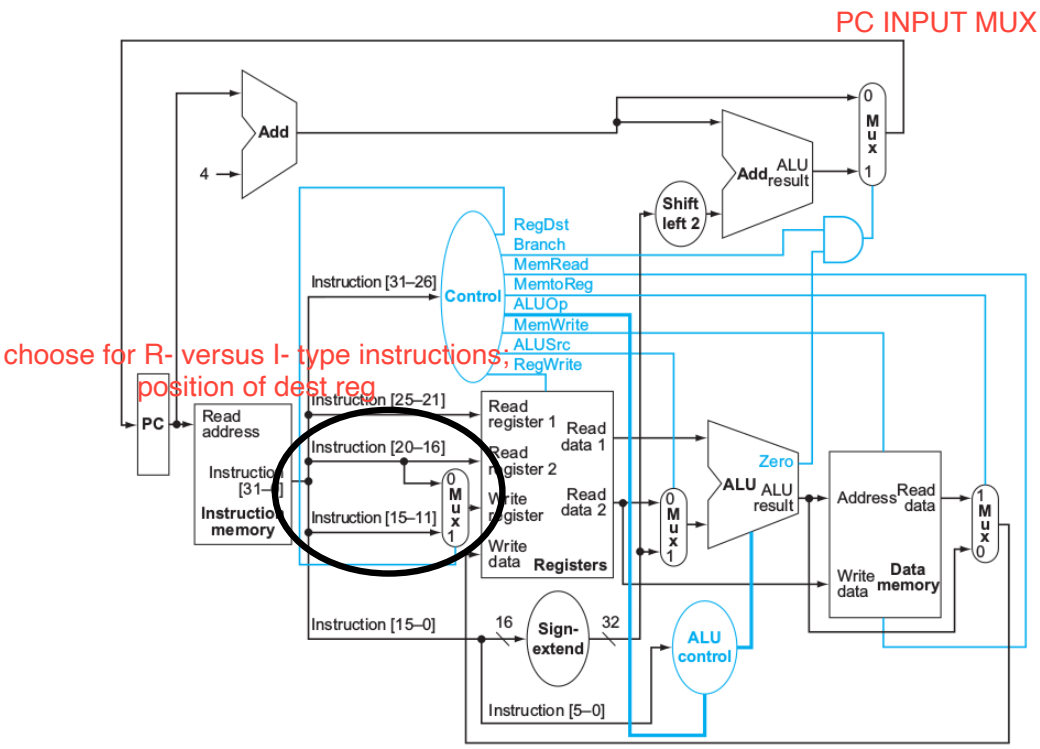

Figure 1: Single-Cycle MIPS CPU Datapath Diagram

5 Grading Rubric

The lab will be graded out of 100 points, distributed as follows:

1. **Task Completion (60 points total)**
   (a) **Task 1: Simulate the Basic CPU (10 points)**
   - Simulation results included and correct: **5 points**
   - Analysis of CPU behavior: **5 points**

   (b) **Task 2: Add `SLT` Instruction (10 points)**
   - Correct modification of ALU and ALU control logic: **5 points**
   - Testbench and simulation results demonstrating `SLT`: **5 points**

   (c) **Task 3: Add `ADDI` Instruction (10 points)**
   - Correct modification of control logic without new hardware: **5 points**
   - Testbench and simulation results demonstrating `ADDI`: **5 points**

   (d) **Task 4: Add `J` Instruction (10 points)**
   - Correct implementation of `J` instruction: **5 points**

- Testbench and simulation results demonstrating `J`: **5 points**

(e) **Task 5: Add `BNE` Instruction (10 points)**

- Correct implementation including necessary hardware changes: **5 points**
- Testbench and simulation results demonstrating `BNE`: **3 points**
- Modified datapath diagram included: **2 points**

(f) **Task 6: Add `LUI` Instruction (10 points)**

- Correct implementation including necessary hardware changes: **5 points**
- Testbench and simulation results demonstrating `LUI`: **3 points**
- Modified datapath diagram included: **2 points**

2. **Report Quality (20 points total)**

- Clear and detailed descriptions of modifications for each task: **20 points**

3. **Demo (20 Points total, Mandatory for Credit)**

- Successful demonstration of CPU functionality for the implemented instructions: **Required**
- Answer two questions asked by the TA to show your understanding of the lab:
  - **0 correct answers:** Demo failed; re-demo required.
  - **1 correct answer:** 0 points for this section.
  - **2 correct answers:** 20 points.
- ***Note:*** *Failure to demo will result in zero credit for the lab.*

4. **Extra Credit (Up to 20 points)**

- Implementation of I-type ALU instructions (e.g., `ANDI`, `ORI`): Points awarded based on correctness and thorough testing and demo.

5. **Early Submission Bonus (Up to 8 points)**

- Demo on Wednesday or earlier: **+8 points**
- Demo on Thursday: **+5 points**

6. **Late Submission Penalty**

- **10 points deducted** for each business day late

6 Additional Instructions

- **Code and Testbenches:** The Verilog code for the CPU and the testbenches will be provided separately.
- **Assumptions:** Clearly state any assumptions made during your implementation.
- **AI Usage: Do not use AI to write code for this lab.** We will be very upset if you use them and may take further step.
- **Submission:** Submit your lab report as a PDF, along with any modified Verilog files. Zip them properly.

7 References

- D. A. Patterson and J. L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, 5th ed. Morgan Kaufmann, 2013.