CS224 – Fall 2020 – Lab #5

Implementing the MIPS Processor with Pipelined Microarchitecture

Dates:
Section 1: Wednesday, 16 December, 13:30-16:20
Section 2: Friday, 18 December, 13:30-16:20
Section 3: Sunday, 20 December, 8:30-11:20
Section 4: Thusday, 17 December, 13:30-16:20

Submission Date: December 21, Monday, 23:59, for all lab days.
(No preliminary work and lab work distinction.
See the submission instructions below)
Physical Lab Administration: No physical lab: All labs are online. Your TAs will be available during your lab hours. Other times you may reach them by email.

TAs:
Section 1 (Wed): Ege Berkay Gülcan, Zülal Bingöl
Section 2 (Fri): Pouya Ghahramanian, Yusuf Dalva
Section 3 (Sun): Yusuf Dalva, Zülal Bingöl
Section 4 (Thu): Berkay Gülcan, Eren Çalık

TA email Addresses:
Berkay Gülcan: berkay.gulcan@bilkent.edu.tr
Eren Çalık: eren.calik@bilkent.edu.tr
Pouya Ghahramanian: ghahramanian@bilkent.edu.tr
Yusuf Dalva: yusuf.dalva@bilkent.edu.tr
Zülal Bingöl: zulal.bingol@bilkent.edu.tr

Purpose: In this lab you will use an online digital design tool, EDA Playground (https://www.edaplayground.com/), to implement and test the Pipelined MIPS Processor and implement a hazard unit to handle possible hazards. You will be provided a skeleton SystemVerilog code for the Pipelined MIPS processor and fill the necessary parts to make it work. To do this, you must first implement the pipeline registers, then implement a datapath to make the necessary connections between the pipelines, control signals and other hardware elements, e.g, forwarding muxes, adders, etc., and finally implement the hazard unit using the logic equations discussed in class. Implementing the Pipelined MIPS Processor will require you to fill-in some SystemVerilog modules in the HDL model of the processor which is provided in the same folder as this one in Moodle. To test and prove correctness, you will simulate the microarchitecture on EDAPlayground.

Summary

Part 1 (50 points): Design Report: Pipeline hazards evaluation and preparing test programs in MIPS.

Part 2 (50 Points): Implementation and simulation of the MIPS-lite pipelined processor.

You will **upload your lab work** as **one PDF** and **one TXT** file as a two file submissions to the Moodle Assignment. Code submissions will be used for similarity testing by MOSS. Please see the related section below for further instructions on MOSS submission.

If we suspect that there is cheating, we will send the work with the names of the students to the university disciplinary committee.

Part 1. Design Report (50 points)

In this part you will prepare a report in PDF format named as:

StudentID_StudentFirstName_StudentLastName_SecNo_LabNo_REPORT.pdf

At the end of this lab, you will have implemented the pipelined MIPS architecture that can be seen in the file that is provided as ***PipelineDatapath.PNG*** (Notice that there is no early branch prediction in this pipeline. Hence, the branch resolution is done in the Decode stage.). Note also that there is no jump instruction implemented as well. Your Design Report should contain the following items:

a) Cover page, with university name, department name, and course name and number at the top, "Design Report", Lab # (e.g. 5), Section #, and your name and ID# in the middle, and the date of your lab at the bottom.

b) **[10 points]** The list of all hazards that can occur in this pipeline. For each hazard, give its type (data or control), its specific name ("compute-use" "load-use", "load-store", "branch" etc.), the pipeline stages that are affected.

c) **[10 points]** For each hazard, give the solution (forwarding, stalling, flushing, combination of these), and explanation of what, when, how.

d) **[10 points]** Give the logic equations for each signal output by the hazard unit, as a function of the input signals that come to the hazard unit. This hazard unit should handle all the data and control hazards that can occur in your pipeline (listed in b) so that your pipelined processor computes correctly.

e) **[20 points]** Write small test programs, in MIPS assembly, that will show whether the pipelined processor is working or not. Each of your test programs should be designed to catch problems, if there are any, in the execution of MIPS instructions in your pipelined machine. Write:

- A test program with no hazards (to verify that there are no problems with the connections in your pipeline etc.)
- A test program that has one type of hazard, and another, and another...

In the end, have at least 4 test programs (testing at least 3 hazards) with their machine code (in hex) and MIPS code (written in your report).

You can use the student-written assembler tool available online to help you quickly implement your test programs[1]*. Remember that the goal of testing is to verify that all the instructions are fully working, and that all the instructions are working even in the presence of hazards.*

Part 2: Implementation and Simulation (50 Points)

a) **[35 points]** You are given a skeleton SystemVerilog code for your pipelined MIPS processor in the file ***PipelinedMIPSProcessorToFill.txt***. The places in the code that needs to be modified are shown with comment blocks above them. Fill them to implement the Pipelined Processor. You don't need to follow the skeleton code point by point. If you think your design is better, you are welcome to try it in your code, as long as your version of the code works, too.

b) **[15 points]** Now make a SystemVerilog testbench file and using EDAPlayground, simulate your Pipelined Processor by executing the test programs you wrote at Part 1-e). Implement a new top module in order to see memwriteM, regwriteW, writedataM, pc, instruction and resultM signals. Print these signals to the console using $display and/or $monitor functions.

Part 3. Submit your code for MOSS similarity testing

Combine all the new and modified Verilog codes into a file called **StudentID_FirstName_LastName_SecNo_LabNo_LAB.txt**. You will then upload this file to the Moodle > CS224 > CS224 Lab #5 for your section along with the report you have prepared in Part 1. If you don't submit your code, your grade for the lab will be 0. Your codes will be compared against all the other codes in the class, by the MOSS program, to determine how similar it is (as an indication of plagiarism). So be sure that you only submit code that you actually wrote yourself!

[1] https://github.com/bilkentCraps/mips