Anna Davis and Toby Hollis

Homework 5 Report

## Summary

1. **How much of the CPU specification did you complete (given your group size)? Is there anything you didn’t get to do?**

We completed everything except for the conditional functions, which we had to do because our group size is 2.

1. **What cases do your tests cover? What cases might your tests not have covered?**

The first tests we added were to cover most of the basic commands of the ALU like subtract, and, or, etc. We made sure to add tests that covered registers and immediates. Then, we also added a load and store test to make sure the memory instructions worked as well. Then, we made sure to also have shift tests and other branch tests to check branching forward and backwards. We didn’t test more conditionals and compares. In addition, we didn’t cover if any of the ALU flags were set and what that would cause the processor to do.

1. **Briefly describe the design of your control unit.**

For the most part, we followed the diagram in the textbook and the diagram of an ARM processor we made in class. However, to specifically implement it, we used truth tables. We made truth tables that took in the functs, ops, and conds. We used those inputs to calculate their results for all of our multiplexor select bits like RegWrite, ALUSrc, ALUControl, MemToReg, etc. We made something very similar to the truth table we did in class for the control unit and then added to it to include other instructions and operations. Thus, our control would take in the op, funct, and cond and split them by bits then based on those values would output certain values to tell the processor to run certain parts/commands.

## Process

1. **Tell us about the specific steps you took to complete the project. How did you start? What was the sequence of specific improvements you made?**

We started by getting the incrementation of the PC to work and then split up the instruction into all of the different pieces we would need. Our next step was to get the add instruction working. We did that by attaching things to the RegFile and using an adder. However, we were able to pass a few tests but weren’t passing all of the add tests due to not having parts of our control set up. Then, we implemented parts of our control. Once we got the mov and both add tests working, we moved on to adding the full ALU. When we first added that, it caused all our tests to fail. We worked to get those basic tests (mov, add reg, and add imm) to pass again by modifying the CPU and the Control. After we got the tests we had previously passed to pass again, we moved on to adding more tests and checking new commands. We followed this format throughout the project where we’d complete one task and then move on to the next task and have to reevaluate to pass the original test and the new tests.

1. **If working on a team**: **how did you manage subcircuits, circuit files, and test code so that you could work both independently and collaboratively? Give specific examples. If you changed your strategy, talk about that, too.**

We both had our own circuits and sub-circuits on our own devices, so we could test our own ideas and do work when we weren’t meeting. However, we would meet up multiple times whether during lab sessions or in our own time and make sure we were on the same page. We’d also sit together for a few hours and work together to make improvements on both of our ideas. So, we had time to work independently and also time to come together and share what we found and how that could be incorporated into what the other person found.

## Debugging story

1. **Describe a time when your project had an undesired behavior. Give the specific behavior you *expected* as well as the specific behavior you *actually observed*.**

When we were working on the project, we were able to pass the move immediate and add immediate tests. However, we still failed the add register tests. Normally our answer would show that register one would be zero whereas the real answer should’ve been another value like two.

1. **Describe the specific steps you took to investigate the cause of the problem. It should be detailed enough that we can follow your thought process.**

We looked first at the test cases that Professor Myers provided. We started by looking at the test cases that our project passed and checked the functionality of those ones. Then, we moved the clock to the test we were failing to see what we were doing wrong. We realized the error was occurring with the value stored in a register as it was storing the wrong value. We looked at each piece of it and compared it to what we expected the numbers to be and how it had worked on the previous test. Then, we reread the textbook on this part after looking more at what we thought the error would be

1. **Describe the specific fix you attempted as a result of that investigation. Why did you think this fix would work? What was the new behavior you observed after the fix?**

We realized that the issue was likely because we had wires attached in the wrong place. Some of the values were continuing to be stored past their instruction which would mess up the next mov instruction. We thought adding a comparator to make sure the ALU was checking the correct instruction and sending that correct instruction to the right place would be the best fix. This new comparison would also be added to multiplexor that would help with producing the correct output. We thought this would work because it would use a conditional statement to make sure that the circuit was using the correct value in the mov instruction which would then also give the registers the correct values. When we tried this in our project, it gave the desired result, and we were able to pass the add register test as well as other basic ALU tests that used registers.