Bill Jason P. Tomas, James Michael Wooten

ELEC 5200, Spring Project Part V

Submission Date: 04/18/2011

Final Report

* What did you learn from this project?

From this project, I learned the development of a RISC CPU, starting from the early developmental stages involving the instruction set, all the way to implementing the CPU onto an FPGA board. The project allowed the user to customize the CPU along with the hardware/software implementation (how instructions can be implemented in different datapath’s) options presented during the lectures, giving the user first-hand experience in CPU development trade-offs and problems that one can encounter during design. I also learned different architectures, control units, datapath’s, and instruction sets, and how they are implemented with one another to create a CPU. Also, the flexibility in creating the instruction set made me realize how important certain instructions are in executing C programs (through pseudo-instructions) and how they play a key role in how fast the CPU can operate. One of the biggest lessons I learned was testing and debugging the VHDL implementation of the CPU in ModelSim. I had to ensure each register transfer was correct for a certain instruction, and that the appropriate control signals were sent to the datapath. The difficulty arose from keeping track of how each instruction operated in the datapath (in my case, how each instruction behaved during each clock cycle), and how to develop an effective test program for the CPU.

* What would you do differently next time?

One thing I would like to do differently if I had the opportunity to repeat this project would be to try a different datapath. I began the VHDL coding for the multicycle datapath before we began the lecture on pipelining, and was unable to go back a re-create the VHDL for my datapath. The pipeline datapath seems to have many different aspects (hazard detection) that are interesting, and I would like to see how they operate during simulation. Also, I would re-design some of my VHDL code to reduce the number of latches, and also attempt to speed up the hardware by reducing the critical path delay through the datapath. I also want to design my control unit differently, maybe try implementing a micro-programmed control unit to improve performance by having a higher degree of parallelism in the datapath. This is a big interest for me since I would like to execute multiple instructions during a single clock cycle. One last thing I would do differently if redesigning my datapath would be to implement a non-linear pipeline for the use of jumps.

* What is your advice to someone who is going to work on a similar project?

Simulation and understanding how each instruction operates on the RTL (register-transfer level) is vital to catching errors in your datapath and debugging the CPU. You need to have total control over everything going on, and ensure that appropriate signals are begin sent, along with the correct register transfers during each clock cycle. Think about how each instruction will be handled during hardware implementation when creating your ISA, and use only instructions that are necessary to the specifications. Make sure to ensure correctness of the CPU before altering any piece of the datapath, because a big aspect of the project is getting the CPU to operate correctly not fast.