Benjamin Newcomer

March 11, 2015

CS 3220 Project 3

I worked on this project with Casey Evanish. Our objective was to design a pipelined processor that implements the CS 3220 ISA and completes a Sorter program in less than sixty seconds. To begin, we split the processor's work into three stages: Fetch/Decode/Register Read, Execute, and Memory/Write back. The first stage handles fetching instructions and reading registers. The second stage handles ALU operations and calculating branch targets. The final stage handles writing to or reading from memory and writing back to registers.

I created the initial pipelined design from the provided template. I created modules for each component (PC, ALU, Register File, etc.) to abstract away some of the complexity of the processor. Next I created buffers between each stage. Casey updated our assembler to be able to assemble mif files with Noops inserted between each instruction. This allowed us to test the processor without having to handle any sort of data hazards. Together we worked through bugs like passing arguments into the ALU in the wrong order for branch instructions, asserting signals with incorrect logic, and incorrectly saving the PC on jump instructions.

To test the processor we used Signal Tap and viewed many of the buffer registers. When debugging the initial design, I used Signal Tap to debug the processor running the provided Test2.a32 which was padded with Noops.

Once we got the processor working without handling data hazards, Casey implemented stalls to pause the PC when conditions for read after write or control hazards are met. Once he tested and debugged that design, we simply bumped up the clock to the maximum working speed (around 87 MHz) to achieve Sorter2 program completion in under 60 seconds (around 50 seconds). We worked independently on different design and debugging portions, but had four meetings over a week and a half to discuss design issues and work through tough bugs.