Project Status Report #7
David Gronlund
Group A3
4/6/19

This week I focused on improving the performance of the core as well as starting to integrate in the floating point unit, which require some modifications to the memory architecture of the core. For the core performance my focus was on improving the branch prediction, not necessarily in how the predictor works as it was doing fine already, but in allowing the core to execute instructions from a prediction without waiting for the speculative counter to run down to zero. This required having two speculative counters which I indexed with the jump flag that I was already using to indicate that the program counter had been successfully updated. The speculative counter table is as large as the jump flag can index, which allows the core to be parameterized in the future rather easily.

The floating point architecture we had laid out was mostly correct, but we realized that RISC-V does have specific instructions for loading and storing to the floating point register file directly from memory, which required us to take one of two approaches. The first was to micro-code the instruction in the decode logic, which was going to be very messy and undo a lot of the progress we made on the integer pipeline. The second was to provide the floating point unit with its own interface to memory, which we decided to go for since it required less modification to our integer core and was going to be made way easier by the memory interconnect that we already needed to write. On top of this the integer-to-floating point conversion was made another functional unit, since the old design would have required more than one write port to the floating point register file.

The memory interconnect was something I also worked on, with both an address-mapped crossbar as well as a quick packet router written and tested, which allows the core to talk to peripherals if necessary, but also to have the local scratchpad memory, which is two ports, accessed at the same time by four separate masters. Those masters are the instruction requests, the data memory requests, the floating point memory requests, and the supervisor, which can read and write from any of the scratchpad as well.