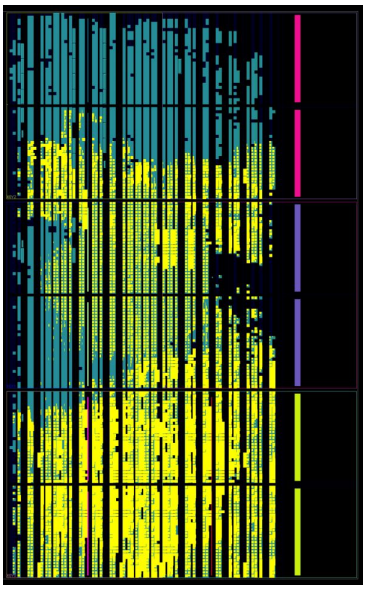
Project Status Report #11
David Gronlund
Group A3
5/4/19

This week I focused on finishing the floating point pipeline, augmenting it to work with the vector operations, and then working on getting the design implemented in the FPGA for testing. This required mostly writing the logic in the floating point decode module, since the math logic had previously already been pipelined.

The floating point logic had a couple of issues that I found in testing, including that adding any number to zero would result in zero, but I managed to run down these issues fairly quickly, and soon the core was running the floating point instructions that I was generating from GCC. I made sure that the compiler never generated instructions from double-promotion, since the core did not support that extension, but also there was some potential GCC issue where the generated double-emulation code would always generate zero.

Testing the vector functionality of the core required writing some assembly as well as using the vector assembler that Alex had wrote. After running down some toolchain issues with Alex I was able to generate some vector assembly. Alex had already written the necessary intrinsic definitions for using the instructions, but the issue I ran into was how to efficiently load the floating point register file. I eventually settled on using the slideup and slidedown instructions in between floating point loads and stores directly from memory, which when interleaved properly should not cause any instruction stalls. This ultimately let me use the vector unit to perform operations like addition, subtraction, and any of the other math operations our normally FPU supported, and correctly.

The final implementation of the vector unit had some hiccups as usual with getting Vivado to recognize the logic, but our final implementation of the core with an integrated FPU and vector unit finally implemented as well.



FPGA Implementation Results