Tiffany Yang
CS380L - Rossbach
March 28, 2019

Final Project Proposal

For this project, I intend to explore how different Verilog modeling approaches affect the performance of several smaller tasks on an FPGA. To be more concrete, I plan to rewrite the following CHStone benchmarks in Verilog: double-precision floating point addition, a simplified MIPS processor, an adaptive differential pulse code modulation decoder and encoder, and a secure hash algorithm. My goal was to choose tasks that were relatively heterogeneous in order to see how their structure affects the process of both encoding solutions in Verilog and how the synthesized code performs on the FPGA under different models. A higher-level goal of this experiment is to develop a taxonomy of sorts that relates certain tasks to the best Verilog model to handle them.

Each of these tasks and the 3 related source files (written under a dataflow model, a behavioral model, and a gate-level model) represents a milestone. Ideally, double precision floating point addition will be completed by April 13, the simplified MIPS processor will be completed by April 18, the pulse code modulation codec will be completed by April 22, and the secure hashing algorithm will be completed by April 25. At this point, each of these solutions can be standardized (so that dataflow models match each other more closely, etc.) and by April 28. At this point, I would like to start running experiments on the FPGA.

For this project, I will need an FPGA and a computer to run Quartus, both of which I already have access to.

For experiments, I plan to time the performance of each of the tasks by timing the number of cycles it takes to complete them. It would also be interesting to look at the amount of logic required to implement each of the different designs. Both of these values can be graphed on a bar graph.

I will know if I have succeeded if I see statistical differences in the performance under different models.

Given that certain problems lend themselves more naturally to certain models, being able to translate a dataflow solution into a gate-level model may be difficult. Still, having Cascade available will make the process of iterating over different solutions more feasible in the meantime. Final solutions will be translated by hand into Verilog rather than relying on the Cascade abstractions.