HighRISCHighReward

CP1 Progress Report

Work Distribution

For checkpoint 1, we designed the datapath all together and contributed equally on the design. We thought it is important to do this initial part together so we are all on the same page. As for the implementation part, Ayush created all the necessary modules in addition to MP2 provided ones, Calvin made the interconnection between these modules in the top level CPU.sv and MP4.sv, and Mandy is mainly in charge of verification of the implementation. At the end, all three of us checked by following the datapath we designed and made sure the routing between the pipeline stages and functional units was correct. Since we have an established datapath diagram at this point, we will be able to split up the workload more in future checkpoints.

Implementation

We implemented the five pipeline stages, and the functional units in each stage, the 4 pipeline registers in between the stages, the control word, and the control ROM.
Some design choices we implemented were the 'bypass' logic in our register file to support simultaneous reads and writes, and also resolving branches in the decode state, by moving our comparator unit from the execution stage, where we had planned to place it, into our decode state, as well as adding an adder in our decode state to calculate effective addresses for branches, in order to prevent more than one cycle penalty when branch misprediction happens in later checkpoints. Resolving the branch in the ID stage will double the number of forwarding paths, but this is a design choice we made.

Testing and Verification

To verify if the pipeline stage works, we first used the MP2 CPU to run the CP1 test code provided, in order to view the correct execution flow and to obtain the correct system state (e.g register state) at the conclusion of the CP1 test program. We then ran the same test code on our CP1 pipeline processor and verified that the final register state matched the final register state for

the test program on the MP2 processor, through Verdi. We also used Verdi to ensure that the PC was incremented and the instruction was being fetched and executed properly.

Fmax & Energy report from Design Compiler

Combinational Delay per Stage: 3.78 ns
Latch Delay per Stage: 0.14 ns
Total Delay per Stage: 3.92 ns
Fmax = 1/(3.92 ns) = 255 MHz