ECE411 Progress

PCB

Prateek Tenkale, Chris Shannon, Brendan Callas

Checkpoint 1

Who Worked on What

Chris: worked on implementing the control ROM and datapath for checkpoint 1, helped with forwarding/hazard design and cache arbiter

Brendan: Worked with forwarding/hazard design, arbiter design, connected new modules back to main datapath

Prateek: Worked on forwarding design, testing/verifying datapath, and design for Checkpoint 2

Functionalities

For Checkpoint 1, we implemented the basic RISC-V RV32I instruction set in a 5-stage pipelined cpu. We implemented our design from the previous checkpoint in SystemVerilog, which is able to run code which has no hazards. We also used the Magic Memory module for this checkpoint, so it relies on a 1-cycle memory response, and cannot yet handle stalls.

We also designed the functionality for the next checkpoint, including implementing L1 Caches, memory, a cache arbiter, data forwarding, and hazard detection with stalls.

Testing Strategy

We used the provided mp4-cp1.s code to test the functionality of our design, using ModelSim to verify that the pipeline stages acted as expected, and provided the correct output. ModelSim was used for debugging in a similar way as well, by analyzing each stage to find errors. Since we do not yet have the RVFI monitor or Shadow Memory connected, our design had to be manually verified (although Magic Memory still provided some error reporting).

Features for Next Checkpoint

For the next checkpoint, we must implement the functionalities that we designed in this checkpoint. So we must implement the cache arbiter, connect our L1 Caches (using a slightly modified version of the MP3 Cache), and connect those to memory. We must also add the forwarding unit and hazard detection, which deals with data hazards and stalling if necessary. Additionally, we must implement a statically-not-taken branch predictor, which although the branch predictor itself is virtually nothing, means we must implement the proper functionality to handle incorrect branch predictions by squashing instructions when the predictor is wrong.

At the end of the checkpoint, we will also propose advanced features for the next checkpoint.

Work Plan For Next Checkpoint

Chris: Implementing Forwarding/Hazard Detection Module

Brendan: Implement cache arbiter and 1-cycle L1 Caches

Prateek: Branch predictor, also working with forwarding/hazard module