Progress Report:

For checkpoint 1, we coded the basic RV32I pipelined datapath according to our paper design, excluding the memory components. Using the magic memory given, our design can;

- Pass five instructions into the 5-stage pipeline.
- Store the signals needed from the previous stages in stage registers in between each stage.

We also made paper designs for data forwarding, hazard detection, and the arbiter.

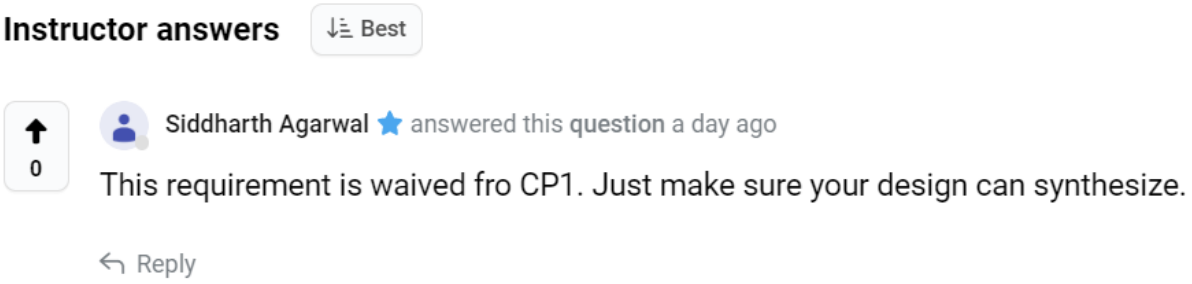
Below is the work distribution:

- Kelsey: Fetch Stage, Decode Stage, Stage Registers, Paper Designs
- Murat: Execute Stage, Memory Stage, Write Back Stage
- Xinpei: Decode Stage, Control Word Generation (Decode)

Testing Strategy:
For this checkpoint, we are given comprehensive tests by the instructors. These tests include NOPs after branches, jumps, and between dependencies because our design does not have hazard detection. These tests also include the magic memory. We will run these tests to verify our design.

The screenshot of the waveform with the regfile data and pc is in our GitLab repository (named cp1_correct.png).

Time and Energy:



Roadmap:

Checkpoint 2 tasks are

- Integration of L1 caches
- Arbiter
- Hazard detection & forwarding
- Static branch predictor

Since the caches and the arbiter are directly connected and are part of the memory hierarchy, we assigned those tasks to the same person. Here are the projected CP2 assignments:

- Kelsey: Hazard detection & forwarding
- Murat: Static branch predictor
- Xinpei: Integration of L1 caches, Arbiter

We will discuss and work on the advanced feature proposals and design together.