ECE411 CP3 Progress Report

Progress

For checkpoint 3, we implemented support for memory instructions and all control instructions, thus completing our baseline out of order processor. We tested the functionality of our processor by debugging coremark, running the provided cp3 benches, as well as running additional tests for edge-cases.

Design Decisions

To issue memory instructions, we implemented a load-store queue to act as a reservation station for memory instructions. The memory instructions get popped in the order they're inserted as soon as the head's instruction gets committed at the head of the ROB if it's a store. Then, the popped instruction gets fed into the memory unit where the address is calculated and fed into the data cache with the operands from the PRF.

For the control instructions, we added a branch functional unit to handle target pc calculations. To handle mispredicts, whenever a taken branch gets committed at the head of the ROB, we do the following:

- Fetch
    - Flush inst queue
    - Update PC to branch target
- Rat
    - Restore rat on mispredict (copy RRF)
- Freelist
    - Rollback pointer on a mispredict
        - Keep free list head pointer in ROB, increment every time a committed inst uses rd, restore FL head to that pointer on mispredict
- Valid array
    - Restore valid phys regs from RRF copy
- Res stations
    - Flush on mispredict
- Execute
    - Flush on mispredict
- CDB
    - Flush on mispredict
- Rob
    - Keep free list head pointer in ROB, increment every time a committed inst uses rd, restore FL head to that pointer on mispredict
    - On mispredict, output committed instruction's br_result and br_target to top level, flush ROB

Work Split

Ahmed worked on implementing and verifying the memory units and LSQ.
Eddie worked on implementing and verifying the control instructions.
Mahir started working on a branch predictor.