Progress Report

- Responsibilities
  - (some tasks overlapped)
  - Jincheng Liu: control_rom, connecting IF/ID_stage module, arbiter and forwarding design.
  - Daniel Marks: Pipeline registers, connecting components in datapath, setting up testbench, debugging
  - Shenjiang Liu: Brought in previous module, help with debugging, they wrote everything so fast, so I just help with them to review little mistakes in the code.

- Functionality
  - Basic CPU with 5 pipelined stages, no protection against data and control hazards

- Test strategy
  - Test bench using source program mp4_cp1.s
  - Loading the memory.lst and tracing the program execution in Modelsim
    - Tracked PC as instructions were executed to validate the functionality of each operation
    - Verified register values at the end of the program
    - Checked the final PC register value during halt loop

Road map

- Work assignment
  - (some tasks overlapped)
  - Jincheng Liu: arbiter, forwarding unit
  - Shenjiang Liu: Icache Dcache, maybe the arbiter with Jincheng
  - Danie Marks: Data hazard protection, branch prediction

- Feature and functionality
  - Protection against data/control hazards
    - Dependency calculation
    - Pipeline stalling
  - Pipeline optimizations
    - Forwarding
    - Transparent Regfile
    - Specialized instruction and data caching system
    - Branch translation buffer (BTB)
    - Case where writeback data need to connect to decode stage
  - Icache Dcache design
    - Single state possible?

Progress Report

- Responsibilities
    - Daniel: Setting up the monitor struct and wiring in top.sv, debugging
    - Jincheng: arbiter, forwarding unit, connecting forwarding unit in the datapath.
    - Shenjiang Liu: ICache, Dcache, debugging
- Functionality
    - Pipelined CPU with protection against data and control hazards
    - Forwarding for data hazards
    - Monitor setup to help with debugging functionality

- Test strategy
    - Running mp4-cp2.s test code using the monitor
    - Tracing execution through modelsim and debugging based on the PC values and the expected control signals based on shadow CPU execution

Road map

- Work assignment
    - Jason: 4 way; pipelined L1 Caches(ask Ta for design)
    - Jincheng: Victim cache; L2 cache;
    - Daniel: Tournament branch predictor
- Feature and functionality
    - Cache organization and design options
        - L2 cache
        - 4-way set associative cache, possibly more way
    - Advanced Cache Options
        - Victim cache
        - Pipelined L1 caches
    - Branch prediction options
        - Tournament branch predictor
            - Local Branch History Table
            - Global Branch History Table
            - 2-bit predictor

Progress Report CP3

- Responsibilities
    - Daniel: tournament branch predictor, pipelined cache, debugging pipeline cache
    - Jincheng: L2, 4 way cache, victim cache
    - Shenjiang Liu: 4 way pipelined cache, connecting victim cache,debugging
- Functionality
    - Pipelined cpu with 4 way pipelined cache
    - L2 cache and 4 way L2 cache
    - Victim cache connected between L2 and physical memory

- Test strategy
    - Running mp4-cp3.s test code using the monitor
    - Tracing execution through modelsim and debugging based on the PC values and the expected control signals based on shadow CPU execution
    - Using performance counters to optimize parametrized modules

Road map

- Work assignment
    - Jason: changing variables to maximize performance, maybe write a multiplier
    - Jincheng: changing variables to maximize performance, maybe write a multiplier
    - Daniel: Optimizing tournament predictor performance, pipelining branch predictors
- Feature and functionality
    - Multiplier:
        - Handle multiply loop
        - Pipeline L2 cache