CA lab 3

R11922211 葉小漓

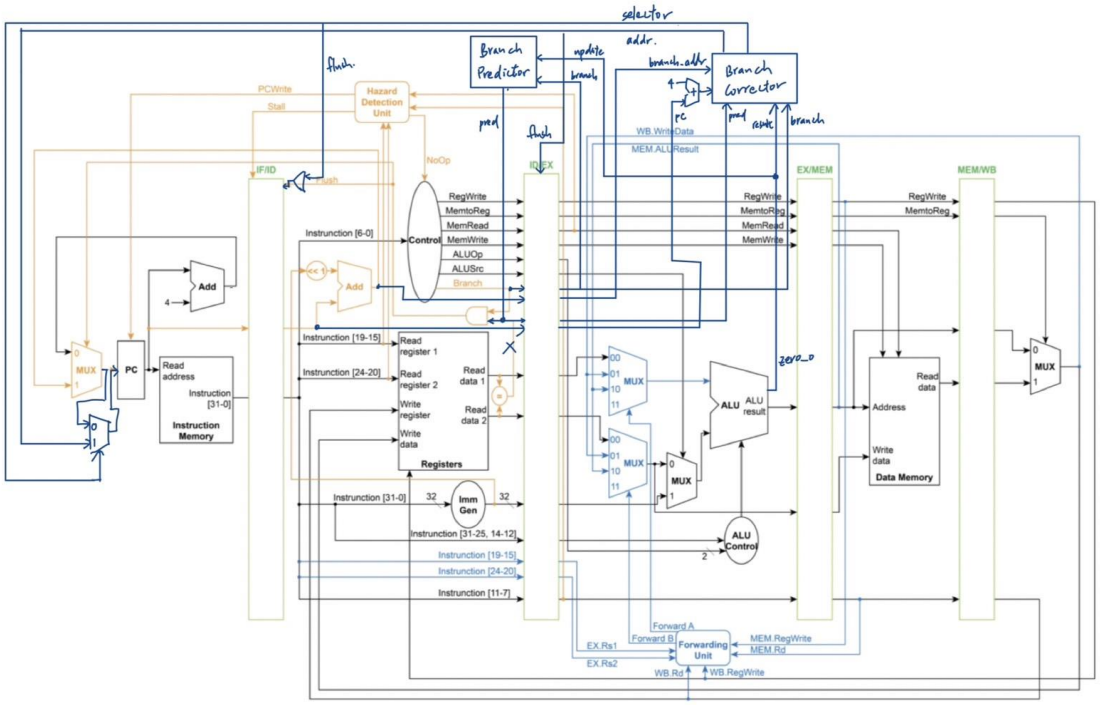
1. Modules Explanation

- **Branch Predictor**
  The Branch Predictor module predicts branch outcomes using a two-bit state register and a prediction output. At each clock edge, if the update signal is active, the module adjusts the state based on the result signal: incrementing for a taken branch and decrementing for a not taken branch, within the limits of 2'b11 and 2'b00, respectively. The prediction output is determined by the state, with 1'b0 indicating a not taken branch for states 2'b00 and 2'b01, and 1'b1 indicating a taken branch for other states.
- **ALU_Control & ALU**
  Set the ALU control so that the beq instruction outputs a subtraction ALUOP. The ALU is the same as lab2, except if the ALU result is zero, signal Zero_o is one.
- **Additional Flushes**
  If the selector output for branch corrector is 1, meaning that a wrong prediction was made, then we need to flush the IF/ID and ID/EX pipelines.
- **Testbench**
- Initialize the output ports of IF_ID, ID_EX, EX_MEM, MEM_WB registers to 0 in the initial block of testbench.v.
- **Others**
  - **Branch Corrector**
    The Branch Corrector module determines the resolved address for a branch instruction. It compares the actual branch result with the predicted outcome. If they match or the branch signal is inactive, it sets the selector to 0, indicating to not change the current instrcution sequence. Otherwise, if there is a misprediction and the branch signal is active, it selects the branch address if the result is true (taken branch) or the PC+4 value if the result is false (not taken branch).
  - **PC_mux2**
    This mux selects which value goes to the PC. It takes selector, correct address on misprediction, and the mux output for pc plus four and branch prediction. If selector is 1, it selects the correct address on misprediction, else, is selects the mux output for pc plus four and branch prediction.
  - **CPU**
    Add branch corrector, branch predictor, PC_mux2 into the CPU module. Connect the flush lines to IF/ID and ID/EX.

2. Difficulties Encountered and Solutions in this Lab

I had difficulty thinking of the design structure of the branch predictor at first. I wasn't sure how to correct the PC after a branch misprediction. But after thinking for some time, I came up with the branch corrector design, which checks whether the prediction is correct, and if mispredicted, outputs the corrected address, and also flushing the wrong instruction stages.

The final datapath design is as follows:



3. Development Environment

- OS: Ubuntu 22.04
- Compiler: iverilog