Computer Architecture LAB2

B09901089 黃柏穎

1. IDEX

   It's a pipeline register, compared to LAB1, I added the following control signals: pc, branchpc, pc is the address of the current instruction, and branchpc is pc+ offset from branch instruction. Both of them are for correction when misprediction occurred.

2. ALU

   Compared to lab1, I added the Zero signal at the output. For branch instruction to determine the result.

3. ALU_Control

   I added Beq's aluop to aluctrl mapping to support the branch instruction in ALU.

4. Branch predictor

   The update_i is branch signal in ex-stage, and result is branch & zero. Because signal arrived at different times in a single cycle, the next state would change multiple times in a cycle. Thus, when update or result signal arrived, I only allow it to modify a dummy state, instead of change the current state directly. The current state can be updated to dummy state only when next clock edge arrives. Thus we can correct the next state multiple times in a clock cycle.

Difficulties:

The main difficulty is choosing the event for always block in branch_prediction module. I tried to use posedge of clk and posedge of update, because I want it to trigger only one time per clock, but if I use posedge of clk, the signal would be 1 cycle late, and if I used posedge of update, the result might change after the edge but not updating the result of branch. So in the end I designed the method described in the branch predictor part above.

Mac OS 12.5