CA Lab3 Report

資工三 B10705005 陳思如

1. Module Explanation

Since most of the modules are the same as the last lab. I will just talk about the ones that I have modified or added.

- Branch Predictor

  This implemented the 2-bit branch predictor which is demonstrated in the spec. It takes in 2 inputs including last cycle branch bit and the last cycle real branch result. Then outputs the predicted result for the current round.

  In this module, it will done three operations based on the its own predictor register state. It will update or reset the register state in every cycle or whenever it received any reset signal.

  1.1

  It first checks the last cycle branch bit. If it is on then it needs to go on the predict result checking and the state update. If not, then it skips the next step.

  1.2

  It checks with the last cycle real calculated result to see if the last prediction is correct or wrong. Then the state register value will change as how the figure in the spec has shown.

  If the last prediction is correct, then the not strongly taken/not taken will goes back to the strongly taken/not taken. Others will just stay at its original state.

  If the last prediction is wrong, then the strongly taken/not taken will move to the not strongly ones. The not strongly ones will move to the another prediction side.

  1.3

  Since the register state has been updated by the last cycle result. Now we will used the updated state to predict the branch for the current cycle. Then we output the predict taken/not taken to the CPU.

- CPU

  This will have all the wired connected to all the module. The jobs added to the CPU module is that it need to handle the branch prediction and the PC selection.

  - PC selection

    It does the pc_add and pc_branch selection as usual at first. Then it does another selection with the not chosen pc_value in the EX stage instruction with the select signal(Flush) to get the real pc number for this cycle.

  - Branch prediction

    It takes in the branch signal and flush signal in EX stage to update the state. Then it outputs the predict result.

    However, the predict result takes effect only when the branch signal from the control module is on.

    After this two operations, the branch predict result then will be sent to the PC selection part. The not selected pc value will be passed to the next stage as a safety check for the flush.

    - Real branch result

      This happens in the EX stage. The ALU calculates the result and we checks if the result matches our prediction or not.

      If yes, then we goes onto the next cycle.

      If not, then we set the Flush signal on and flushes all the data in IFID and IDEX register. Then continues the operation at the not chosen pc value.

- IFID register

  I have added to pass the pc_add(pc + 4) value to the EX stage. So that, it can use the branch predictor in this stage to know which is the next pc value then send it back to the PCMUX.

- IDEX register

  Since we will have the real branch result calculated until the EX stage, we have to send the current branch signal and the prediction to the next stage to check if the branch prediction is correct or not.

  Also, we need to pass the not chosen branch's pc value in case that the prediction is wrong and we will need a flush.

  Moreover, we need a flush input in this register because the flush is decided at the EX stage. If the flush is needed, then the IFID and IDEX will all be flushed.

2. Difficulties

- pc value needs to be passed down to each stage.

  In the beginning I thought that the pc value only takes effect in the PCMUX. However, since we will have the flush signal that happens so long after, we can't no longer holds the not chosen one in the MUX module. So I figure out that the pc value must be passed down to maintain the one pc value that we wanted to roll back.

3. Environment

Windows/ iverilog

I used the second option for compilation.