COSE222 Computer Architecture
Final Project 5th Milestone
(20% credit towards final project)

No late turn-in accepted

Add control hazard detection and handling logic on the pipelined CPU you have designed for the 4th milestone. Your pipelined version of CPU should be able to run the following code. It will display 1, 2, 3, and 4 on HEX3 ~ HEX0 if CPU is implemented correctly.

http://esca.korea.ac.kr/teaching/cose222_CA/milestones/final-project_milestone5.zip

You should satisfy the following design requirements;

1. Branch (`beq, bne`) outcome and destination are calculated in the EX stage of the pipeline
2. Jump (`j, jal, jr`) destination are calculated in the EX stage of the pipeline
3. Do not implement the delayed branch: The real MIPS implements the delayed branch though. You can check it out by looking at the dump file after compilation. See how the compiler rearranges the assembly code (the milestone #5 assembly code) to fill up the delay slot!

Note that when you change and/or modify the Verilog code, add the comment lines in the source. Following is an example.

```
// ######  Taeweon Suh: Start  #######

      assign branch_DE = branch_D & branch_E;

// ######  Taeweon Suh: End #######
```

You should probably follow the steps below;

1. Compile the code under Eclipse and generate the MIPS binary.
2. Test-run it on your single-cycle CPU to make sure that it displays the numbers.
3. Implement the pipelining with control hazard detection and handling.
4. **Simulate with ModelSim** to debug and validate your design.
5. Synthesize the pipelined CPU with the `mif` file using Quartus-II.
6. Download the bitstream to the DE0 board.

What and How to submit:

- Create a (up to) 2-min video clip (with your smartphone or any other convenient means), showing
  - Your smiling face to camera
  - 7 segment output on DE0 board

  AND **verbally** explaining the followings:
  - **What instructions** you added to the CPU, and **how you figured out those instructions** to be added to the CPU
  - **CPU design change** (please elaborate this!)
  - **ModelSim simulation output**

- Upload **both** the video clip **and** zipped Verilog source to Blackboard

Note: This is an individual project. You are welcome to discuss, but DO NOT COPY solutions. If you are found to copy solutions from others or slightly modify the solutions from others, both of you will be given 0 credits.