CprE 381 Homework 8

[Note: This assignment is targeted to help solidify your knowledge of exam 2 topics as well as pipeline control hazard. Such understanding is useful for both part 2 of your term project and to understand transient execution attacks such as Spectre.]

1. Exam 2 Rework
   Rework and submit your worst graded problem (i.e., 2, 3, ..., 12) from Exam 2. I leave it up to you to select an appropriate problem where you had a genuine misconception. You must rework the entire problem and submit it as a legible, one- or two-page pdf. **Include a description of your misconception.** This problem's grade will be the percent score from your best revised problem. Since you have time and the ability to access resources, partial credit will be less available. In particular, if I made comments on your question and took off few to no points, there may still be mild errors in your solution! Please ask questions in OHs, before class, and on the Homework Help channel to make sure you understand the problem.

2. Control Hazards
   The following problem was sourced from the fifth edition of our textbook, problem 4.14.

   This exercise is intended to help you understand the relationship between delay slots, control hazards, and branch execution in a pipelined processor. In this exercise, we assume that the following MIPS code is executed on a pipelined processor with a 5-stage pipeline, full forwarding, and a predict-taken branch predictor:

```
        lw $2,0($1)
label1: beq $2,$0,label2 # not taken once, then taken
        lw $3,0($2)
        beq $3,$0,label1 # taken
        add $1,$3,$1
label2: sw $1,0($2)
```

   a. 4.14.1: Draw the pipeline execution diagram for this code, assuming there are no delay slots and that branches execute in the EX stage.
   b. 4.14.2: Repeat 4.14.1, but assume that delay slots are used. *[Note, up until this point we have not used delay slots and our version of MARS has them turned off by default.]* In the given code, the instruction that follows the branch is now the delay slot instruction for that branch.
   c. Repeat 4.14.1 but assume no prediction and not hardware detection or prediction of control hazards (data hazards are still detected and forwarded). Insert the appropriate NOPs and draw the corresponding pipeline diagram.

   Section (COD 4.9) describes how the severity of control hazards can be reduced by moving branch execution into the ID stage. This approach involves a dedicated comparator in the ID stage, as shown in COD Figure 4.62. However, this approach potentially adds to the latency of the ID stage, and requires additional forwarding logic and hazard detection.

d. 4.14.4: Using the first branch instruction in the given code as an example, describe the hazard detection logic needed to support branch execution in the ID stage as in COD Figure 4.62. Which type of hazard is this new logic supposed to detect?
e. 4.14.5: For the given code, what is the speedup achieved by moving branch execution in the ID stage? Explain your answer. In your speedup calculation, assume that the additional comparison in the ID stage does not affect clock cycle time.
f. 4.14.6: Using the first branch instruction in the given code as an example, describe the forwarding support that must be added to support branch execution in the ID stage. Compare the complexity of this new forwarding unit to the complexity of the existing forwarding unit in COD Figure 4.62.

3. Computer Architecture in the Media (***Extra Credit*** to be applied globally to the course)

   Find an online article from the past two years that has some relation to computer architecture. Make sure you can relate it to at least three of the following terms: CPI, frequency/clock cycle, speculative execution, pipeline/pipelining, execution time, instruction, and power. Summarize the article in one paragraph (4-6 complete sentences) and, in a second paragraph, describe how the article relates to the above terms and what you have been learning in class.