Classwork 3

We will solve these questions in class. This is done in the context of turning a homework into a classwork.

Question 1:
In this question, we will assume that moving the logic for the evaluation of the branch condition and target address from the ALU stage of the 5-stage pipeline to the ID stage causes an increase in the clock cycle time by 5%. We will also assume that 20% of the instructions executing on the pipeline are branch instructions and that 60% of the branches are taken.

(a) Assume that the CPI = 2.12 when the branch condition and target are evaluated in the ALU stage. Will resolving the branch condition/target in the ID stage result in a more efficient execution?

(b) In addition to increasing the cycle time, this change creates a new type of data hazard that cannot be resolved with the forwarding paths discussed so far. For example, the following sequence of instructions will cause a hazard

```
add $6, $2, $2
beq $5, $6, L
```

Explain why will the above sequence of instructions cause a hazard and describe the forwarding path(s) that is (are) needed to resolve that hazard.

Questions 2: Consider a five stage pipelined processor in which the branch target address is determined in the ID stage (second stage) and the branch condition is determined in the MEM stage (fourth stage). Assume that the instruction mix executing on the processor contains 20% branch instructions, 70% of which are taken. Assume also that the processor’s CPI is 1.9 without accounting for the overhead of dealing with control hazards (but accounting for other hazards). Compute the CPI when each of the following techniques is used to avoid control hazards.

(a) Always predict “branch not taken” and take a corrective action if the prediction is wrong

(b) Always predict “branch taken” and take a corrective action if the prediction is wrong

(c) Assume a branch predictor with an 80% accuracy. That is, the predictor makes the right prediction 80% of the time and the wrong prediction 20% of the time. For this part, assume that a predictor always makes a prediction and ignore the cases where the predictor cannot make a prediction because of a BTB collision or invalid entry (branch seen for the first time).