CSE 560 – Practice Problem Set 3

Three of these problems come from Hennesy & Patterson's *Computer Architecture: A Quantitative Approach*, 3rd edition.

1. The decode pipeline stage is a poorly chosen name. Considering that it does more than decoding, what else does it do? What would you propose as an alternative name?

2. This exercise asks how well hardware can find and exploit instruction-level parallelism (i.e., pipelining). Consider the following four RISC machine code fragments, each containing two instructions:

   i. addi r1 ← r1, #4
      load r2, 7(r1)

   ii. add r3 ← r1, r2
       store r2, 7(r1)

   iii. breq r1, place
        store r1, 7(r1)

   iv. store r3, 17(r10)
       load r2, 12(r8)

   (a) For each code fragment (i) to (iv) identify each dependence that exists or that may exist (a fragment may have no dependencies).

   (b) For each code fragment, indicate whether data forwarding is sufficient to resolve the dependence or if stall cycles are required. Indicate the number of stall cycles.

3. Consider the following RISC assembly code.

   (1) load r1,45(r2)
   (2) add r7 ← r1, r5
   (3) sub r8 ← r1, r6
   (4) or r9 ← r5, r1
   (5) brneq r7, target
   (6) add r10 ← r8, r5
   (7) xor r2 ← r3, r4

   (a) Identify each dependence (both data and control); list the two instructions involved; identify which instruction is dependent; and, if there is one, name the storage location involved (register or memory) (e.g., register r1 or memory address 4(r1)).

(b) Using the 5-stage pipeline from class, which of the data dependences that you found in part (a) become hazards and which do not?

(c) Draw a pipeline diagram for the sequence, including stalls needed to rectify the hazards. Assume the branch is predicted NT and was taken.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
(1)																
(2)																
(3)																
(4)																
(5)																
(6)																
(7)																

4. Increasing the size of a branch prediction buffer means that it is less likely that two branches in a program will share the same predictor. A single predictor predicting a single branch instruction is generally more accurate than is that same predictor serving more than one branch instruction.
   (a) List a sequence of branch taken and not taken actions to show a simple example of 1-bit predictor sharing that reduces misprediction rate.
   (b) List a sequence of branch taken and not taken actions that show a simple example of how sharing a 1-bit predictor increases misprediction.
   (c) Discuss why the sharing of branch predictors can be expected to increase mispredictions for the long instruction sequences of actual programs.