Brown University
School of Engineering
EN164 Design of Computing Systems
Professor Sherief Reda
HW 04. 65 points. Due Date: Tuesday, April 14th in class.

---

1. [13 points] The importance of having a good branch predictor depends on how often conditional branches are executed. Together with branch predictor accuracy, this will determine how much time is spent stalling due to mispredicted branches. In this exercise, assume that the breakdown of dynamic instructions into various instruction categories is as follows:

R-Type: 40% BEQ: 25% JMP: 5% LW: 25% SW: 5%

Also assume the following branch predictor accuracies:

Always-Taken: 45% Always-Not-taken: 55% 2-Bit: 85%

a. [4 points] Stall cycles due to mispredicted branches increase the CPI. What is the extra CPI due to mispredicted branches with the always-taken predictor? Assume that branch outcomes are determined in the EX stage, that there are no data hazards, and that the no delay slot are used.
b. [4 points] Repeat (a) for the "Always-not-taken" predictor.
c. [5 points] Repeat (a) for the 2-bit predictor.

---

2. [20 points] Assume the following repeating pattern (e.g., in a loop) of branch outcomes:

T, NT, T, T, NT

a. [5 points] What is the accuracy of always-take and always-not-taken predictors for this sequence of branch outcomes?
b. [5 points] What is the accuracy of the two-bit predictor for the first four branches in this pattern, assuming that the predictor starts off in (predict not taken) state (bottom-left dark-grey state in the lecture slides)?
c. [5 points] What is the accuracy of the two-bit predictor if this pattern is repeated forever?
d. [5 points] Design a predictor that would achieve a perfect accuracy if this pattern is repeated forever. Your predictor should be a sequential circuit with one output that provides a prediction (1 for taken, 0 for not taken) and no inputs other than the clock and the control signal that indicates that the instruction is a conditional branch.

---

3. [12 points] The following core is written in C, where elements within the same row are stored contiguously

```
for (J = 0; J < 8000; J++)
   for (I = 0; I < 8; I++)
      A[I][J] = B[I][0] + A[J][I];
```

a. [1 points] How many 32-bit integers can be stored in a 16-byte cache line?

b. [1 points] References to which variables exhibit temporal locality?

c. [1 points] References to which variables exhibit spatial locality?

Locality is affected by both the reference order and data layout. The same computation can also be written below in Matlab, which differs from C by contiguously storing matrix elements within the same column.

```
for J = 1:8000
  for I = 1:8
    A(I, J) = B(I, 0) + A(J, I);
  end
end
```

d. [1 points] How many 16-byte cache lines are needed to store all 32-bit matrix elements being referenced?

e. [1 points] References to which variables exhibit temporal locality?

f. [1 points] References to which variables exhibit spatial locality?

---

4. [20 points] Assume the following address stream is for a **word-addressable memory** (e.g., 253 reference to word 253 in the memory):

3, 180, 43, 2, 191, 88, 190, 14, 181, 44, 186, 253

a. [5 points] Show the final cache contents for a three-way set associate cache with two-word blocks and a total size of 24 words. Use the LRU replacement. For each reference identify the index bits, the tag bits, the block offset bits, and if it is a hit or a miss.

b. [5 points] Show the final cache contents for a fully associate cache with one-word blocks and a total size of 8 words. Use LRU replacement. For each reference, identify the index bits, the tag bits, and if it is a hit or a miss.

c. [10 points] What is the miss rate for a fully associate cache with two-word blocks and a total size of 8 words, using LRU replacement? What is the miss rate using MRU (most recently used) replacement? Finally what is the best possible miss rate for this cache, given any replacement policy?