CS 203 (2019 Fall) Assignment #2

Student ID #:

Name:

1. Consider the following matrix transpose code

```
int i, j ,k;
double *A, *B, *C;
A = (double *)malloc(sizeof(double)*N*N);
B = (double *)malloc(sizeof(double)*N*N);
init_data(A, N*N);
for(i = 0; i < N; i++)
    for(j = 0; j < N; j++)
            B[i*N+j] = A[j*N+i];
// assume load A[j*N+i] and then store B[i*N+j]
output_data(B, N*N);
```

Assume that the starting address of array A is 0x20000 and array B is 0x40000. Assume N = 128. Please answer the following questions:

(1) Assuming that you have an AMD Bulldozer microarchitecture that has a 16K, 4-way, 64-byte blocked L1 data blocked L1 data cache, please estimate the cache miss rate.

(2) Continued from the previous question, how many of the misses are compulsory misses? How many of them are conflict misses?

2. Smith and Goodman found that for a given small size, a direct-mapped instruction cache consistently outperformed a fully associative instruction cache using LRU replacement.
   (1) Explain how this would be possible. (Hint: You can't explain this using the three C's (compulsory, capacity, conflict) model of cache misses because it "ignores" the cache policy.)

   (2) Explain where replacement policy fits into the three C's model, and explain why this means that misses caused by replacement policy are "ignored"--or, more precisely, cannot in general be definitively classified--by the three C's model.

(3) Are there any replacement policies for the fully-associative cache that would outperform the direct-mapped cache? Ignore the policy of "do what a direct mapped cache would do."

3. Way prediction allows an associative cache to provide the hit time of a direct-mapped cache. The MIPS R10K processor uses way prediction to achieve a different goal: reduce the cost of a chip package. The R10K hardware includes an on-chip L1 cache, an on-chip L2 tag comparison circuitry, and an on-chip L2 way prediction table. L2 tag information is brought on chip to detect an L2 hit or miss. The way prediction tables contains 8K 1-bit entries, each corresponding to two L2 cache blocks. L2 cache storage is built externally to the processor package, must be two-way associative, and may have one of several block sizes.

   (1) How can way predictoin reduce the number of pins needed on the R10K package to read L2 tags and data, and what is the impact on performance compared to a package with a full complement of pins to interface to the L2 cache?

   (2) What is the performance drawback of just using the same smaller number of pins but not including way prediction?

(3) If a 512 KB L2 cache has 64-byte blocks, how many way prediction entries are needed? How would the R10K support this need?

(4) For a 4 MB L2 cache with 128-byte blocks, how is the usefulness of the R10K way prediction table analogous to that of a branch history table?

4. You are building a system around a single-issue in-order processor running at 2 GHz and the processor has a base CPI of 1 if all memory accesses are hits. The only instructions that read or write data from memory are loads (20% of all instructions) and stores (5% of all instructions). The memory system for this computer is composed of a split L1 cache that imposes no penalty on hits. Both the I-cache and D-cache are direct mapped and hold 32KB each. You may assume the caches use write-allocate and write-back policies. The L1 I-cache has a 2% miss rate and the L1 D-cache has a 5% miss rate. Also, 50% of all blocks replaced from L1 D-cache are dirty. The 512KB write-back, unified L2 cache has an access time of 12ns. Of all memory references sent to the L2 cache in this system, 80% are satisfied without going to main memory. Also 25% of all blocks replaced are dirty. The main memory has an access latency of 60ns. What is the overall CPI, including memory accesses?