CS-UY 2214 — Recitation 11

Introduction

Complete the following exercises. Unless otherwise specified, put your answers in a plain text file named `recitation11.txt`. Number your solution to each question. When you finish, submit your file on Gradescope. Then, in order to receive credit, you must ask your TA to check your work. Your work should be completed and checked during the recitation session.

Problems

1. Explain the difference between *write-through* and *write-back* policies.
2. Explain the difference between *write-around* and *write-allocate* policies.
3. We know that a memory address can be divided into three components for the purposes of cache access:

Tag	Index	Offset

   For example, on a system with 32-bit addresses and one-byte cells, and a direct-mapped cache with 128-byte blocks and 32 rows:

   - the block offset would occupy 7 bits ($2^7 = 128$),
   - the index would occupy 5 bits ($2^5 = 32$),
   - and the tag would occupy the remaining 20 bits ($32 - 5 - 7 = 20$).

   For what kind of cache would it be true that the tag is equal to the address? In other words, what kind of cache will have both the offset and the index occupy zero bits?
4. Your E20 computer has a 4-row 2-way set associative cache with a blocksize of 32 bits (i.e. two memory cells). As you know, E20 memory addresses are 13 bits. The cache is initially empty. Instruction reads are not cached (in other words, only `lw` and `sw` affect the cache).

   Write an E20 assembly language program that will generate a cache eviction on this computer. Your program should be as short as possible.
5. Using the E20 cache system described in the previous question:
   (a) calculate the data capacity of the cache (i.e. excluding metadata) in bits.
   (b) calculate the total size of the cache (i.e. including metadata) in bits. When calculating metadata, include the tag and valid bit.
6. Now consider an E20 computer that has two caches:

   - the L1 cache is an 8-row direct-mapped cache with a blocksize of 32 bits.
   - the L2 cache is a 4-row 2-way set-associative cache with a blocksize of 64 bits.

E20 memory addresses are 13 bits. Memory cells are 16 bits. The caches are initially empty. Instruction reads are not cached (in other words, only `lw` and `sw` affect the cache).

Write an E20 assembly language program that will generate a memory reference that causes a miss on L1, but a hit on L2. Your program should be as short as possible.

7. Assume a system with 32-bit pointers and 512-byte blocks. Its cache is 2 MB, excluding metadata. Each memory cell is one byte.

   (a) How many blocks can be stored in the cache?

   (b) How many bits is the block offset?

   (c) Assuming a fully-associative cache, how many bits is the tag?

   (d) Assuming a two-way set-associative cache, how many sets (i.e. rows) are there?

   (e) Assuming a two-way set-associative cache, how many bits is the index (i.e. the component of the address that indicates the cache row)?

   (f) Assuming a two-way set-associative cache, how many bits is the tag?

   (g) Assuming an eight-way set-associative cache, how many sets are there?

   (h) Assuming an eight-way set-associative cache, how many bits is the index?

   (i) Assuming an eight-way set-associative cache, how many bits is the tag?