Homework 5

Out: 11.10.21
Due: 11.22.21

1. [Memory Hierarchy]
   The following code snippet is written in C, where elements within the same row are stored contiguously. Assume each word is a 64-bit integer.

   ```
   for (i=0; i<10; i++)
     for (j=0; j<1000; j++)
       A[i][j] = B[i][0] + A[j][i];
   ```

   a) How many 64-bit integers can be stored in a 16-byte cache block?
   b) Which variable references exhibit temporal locality?
   c) Which variable references exhibit spatial locality?

2. [Memory Hierarchy]
   The following is a list of 64-bit memory address references, given as <u>word</u> addresses:
   0x03, 0xB4, 0x2B, 0x02, 0xBF, 0x58, 0xBE, 0x0E, 0xB5, 0x2C, 0xBA, 0xFD

   a) For each of the references, provide the binary word address, the tag, and the index, for a direct-mapped cache with 16 one-word blocks. Also list for each reference whether it is a hit or a miss, assuming that the cache is initially empty.
   b) For each of the references, provide the binary word address, the tag, the index, and the offset, for a direct-mapped cache with two-word blocks and a total size of eight blocks. Also list for each reference whether it is a hit or a miss, assuming that the cache is initially empty.
   c) Which of the following eight-word direct-mapped cache designs provides the best performance for the above reference? Explain.
      C1: 1-word blocks
      C2: 2-word blocks
      C3: 4-word blocks

3. [Memory Hierarchy]
   In a direct-mapped cache and 64-bit addresses, the following bits of the address are utilized for cache access: Tag – bits 63-10, Index – bits 9-5, Offset – bits 4-0. Assume byte-addressing, and you may also assume that words are 8-bytes.
   a) What is the cache block size, in words?
   b) How many blocks does the cache have?
   c) What is the ratio between total bits required for such a cache and the data storage bits?
   d) Starting from an empty cache, the following byte-addressed cache references are recorded:
      0x00, 0x04, 0x10, 0x84, 0xE8, 0xA0, 0x400, 0x1E, 0x8C, 0xC1C, 0xB4, 0x884

For each reference, list its tag, index, and offset, whether it is a hit or a miss, and which bytes were replaced (if any).

e) What is the hit ratio of the above reference pattern?

f) For the above reference pattern, list the final state of the cache, with each valid entry represented as a record of <index, tag, data>. For example, <0, 3, Mem[0xCC0]-Mem[0xC1F]>

4. [Memory Hierarchy]
Cache access time is proportional to capacity. Assume that main memory accesses take 70ns and that 36% of all instructions access data cache/memory. The following table shows data for L1 caches attached to each of two processors, P1 and P2.

	L1 Size	L1 Miss Rate	L1 Hit Time
P1	2KB	8.0%	0.66ns
P2	4KB	6.0%	0.90ns

a) Assuming that the L1 hit time determines the cycle times for P1 and P2, what are their respective clock rates?

b) What is the Average Memory Access Time for P1 and P2 for instructions only (not data), in cycles?

c) Assuming a base CPI of 1.0 without any memory stalls, what is the total CPI for P1 and P2? Which processor is faster? (When we say a "base CPI of 1.0", we mean that instructions complete in one cycle, unless either the instruction access or the data access causes a cache miss.)

5. [Memory Hierarchy]
The following is a list of 64-bit memory address references, given as word addresses:
0x03, 0xB4, 0x2B, 0x02, 0xBE, 0x58, 0xBF, 0x0E, 0x1F, 0xB5, 0xBF, 0xBA, 0x2E, 0xCE

a) Describe the organization of a 3-way set associative cache with two-word blocks and a total size of 48 words. Include the width of the tag and data fields.

b) Trace the behavior of the cache, assuming a true LRU replacement policy. For each reference, specify the binary word address, the tag, the index, the offset, whether the reference is a hit or a miss, and which tags are in each way of the cache after the reference has been handled.

c) Now assume a fully associative cache with one-word blocks and a total size of eight blocks. Trace the behavior of the cache, assuming a true LRU replacement policy. For each reference, specify the binary word address, the tag, the index, the offset, whether the reference is a hit or a miss, and which tags are in the cache after the reference has been handled.

6. [Virtual Memory]
A virtual memory system consists of a virtual address size of 32 bits, a page size of 8KB, and a page table entry size of 4 bytes. What is the maximum possible page table

size for a system running five processes?

7. [Virtual Memory]
A virtual memory system consists of 4KB pages, a four-entry fully-associative TLB, and true LRU replacement. If pages need to be brought in from disk, the page that is one larger than the largest occupied physical page number is assigned.
Following is a stream of virtual byte addresses:
0x123D, 0x08B3, 0x365C, 0x871B, 0xBEE6, 0x3140, 0xC049

The TLB content is as follows:

Valid	Tag	Physical Page Number	Time Since Last Access
1	0xB	12	4
1	0x7	4	1
1	0x3	6	3
0	0x4	9	7

The page table content is as follows (assume all other entries are invalid):

Index	Valid	Physical Page Number
0	1	5
1	0	4
2	0	7
3	1	6
4	1	9
5	1	11
6	0	9
7	1	4
8	0	2
9	0	5
A	1	3
B	1	12

For each of the above accesses, list whether the access is: a TLB hit or miss, a page table hit or a page fault. Also list the final state of the TLB.