ITSC 3181 Introduction to Computer Architecture, Spring 2023

Homework #5, Due 04/25/2023

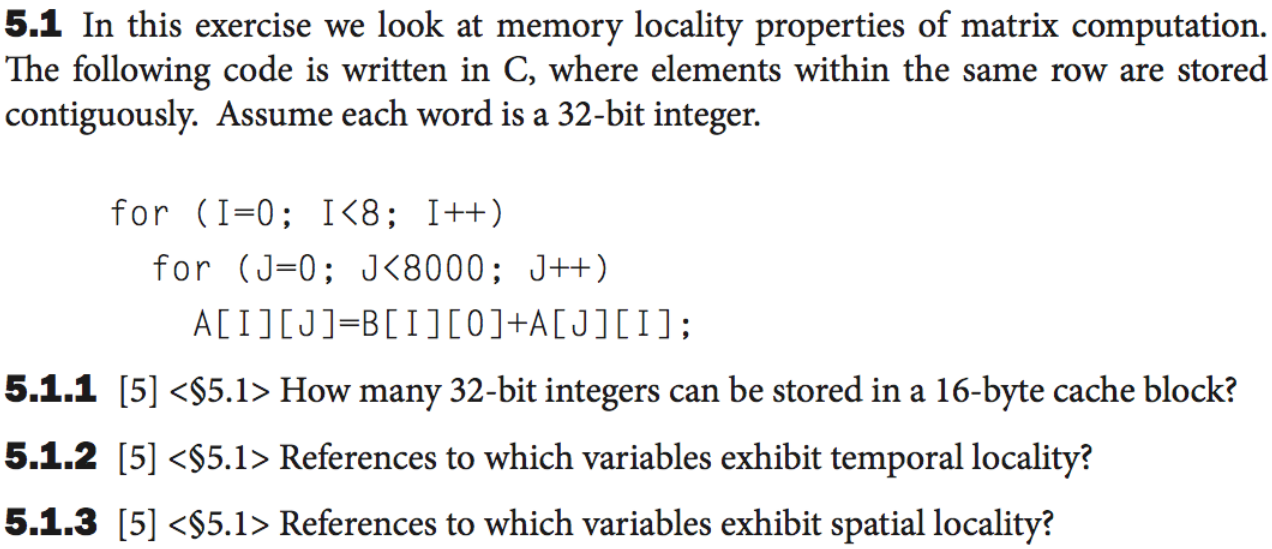
Covered topics: Chapter 5 Memory Hierarchy, Cache and Virtual Memory

5% of the grade of this homework will be added to your final grade.

Question	5.1	5.2	5.3	5.6	5.12	Total points
Total Points	25	20	15	30	10	100
Grade						

Submission:

1. **Only electronic submissions on Canvas are accepted.**
2. **All your solutions should be included in a SINGLE PDF file. Include your full name in the PDF file.**
3. **Number your solutions in the same way and in the same order as the questions are numbered in this document.**
4. **You must show the necessary steps when you solve each question. 0 point will be given to the question if the answer only includes the final answer with no necessary steps.**
5. **Scanned copy of handwritten work will be accepted and graded only if it is written clearly and readable.**



5.1 In this exercise we look at memory locality properties of matrix computation. The following code is written in C, where elements within the same row are stored contiguously. Assume each word is a 32-bit integer.

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

5.1.1 [5] <§5.1> How many 32-bit integers can be stored in a 16-byte cache block?

5.1.2 [5] <§5.1> References to which variables exhibit temporal locality?

5.1.3 [5] <§5.1> References to which variables exhibit spatial locality?

5.2 Caches are important to providing a high-performance memory hierarchy to processors. Below is a list of 64-bit memory address references, given as word addresses.

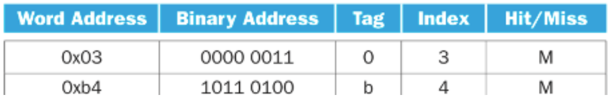
0x03, 0xb4, 0x2b, 0x02, 0xbf, 0x58, 0xbe, 0x0e, 0xb5, 0x2c, 0xba, 0xfd

5.2.1 [10] <§5.3> For each of these references, identify the binary word address, the tag, and the index given a direct-mapped cache with 16 one-word blocks. Also list whether each reference is a hit or a miss, assuming the cache is initially empty.

5.2.2 [10] <§5.3> For each of these references, identify the binary word address, the tag, the index, and the offset given a direct-mapped cache with two-word blocks and a total size of eight blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty.

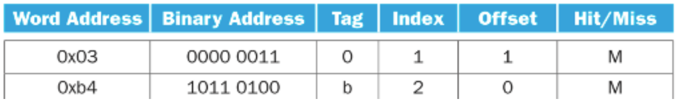
Use the following table structure to work on the question:

5.2.1:



Word Address	Binary Address	Tag	Index	Hit/Miss
0x03	0000 0011	0	3	M
0xb4	1011 0100	b	4	M

5.2.2:



Word Address	Binary Address	Tag	Index	Offset	Hit/Miss
0x03	0000 0011	0	1	1	M
0xb4	1011 0100	b	2	0	M

5.3 For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache.

Tag	Index	Offset
31–10	9–5	4–0

5.3.1 [5] <§5.3> What is the cache block size (in words)?

5.3.2 [5] <§5.3> How many entries does the cache have?

5.3.3 [5] <§5.3> What is the ratio between total bits required for such a cache implementation over the data storage bits?

5.6 In this exercise, we will look at the different ways capacity affects overall performance. In general, cache access time is proportional to capacity. Assume that main memory accesses take 70 ns and that memory accesses are 36% of all instructions. 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	2 KiB	8.0%	0.66 ns
P2	4 KiB	6.0%	0.90 ns

5.6.1 [5] <§5.4> Assuming that the L1 hit time determines the cycle times for P1 and P2, what are their respective clock rates?

5.6.2 [5] <§5.4> What is the Average Memory Access Time for P1 and P2?

5.6.3 [5] <§5.4> Assuming a base CPI of 1.0 without any memory stalls, what is the total CPI for P1 and P2? Which processor is faster?

5.12 In this exercise, we will examine space/time optimizations for page tables. The following list provides parameters of a virtual memory system.

Virtual Address (bits)	Physical DRAM Installed	Page Size	PTE Size (byte)
43	16 GiB	4 KiB	4

5.12.1 [10] <§5.7> For a single-level page table, how many page table entries (PTEs) are needed? How much physical memory is needed for storing the page table?