# CS 2340 Assignment 5 (4%)

Due 11:59pm, April 30, 2024

**QUESTION 1** (20%)

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];
```

a. How many 32-bit integers can be stored in a 16-byte cache block?
b. References to which variables exhibit temporal locality?
c. 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 storing matrix elements within the same column contiguously in memory.

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

d. How many 16-byte cache blocks are needed to store all 32-bit matrix elements being referenced?
e. References to which variables exhibit temporal locality?
f. References to which variables exhibit spatial locality?

**QUESTION 2** (20%)

Caches are important to providing a high-performance memory hierarchy to processors. Below is a list of 32-bit memory address references, ***given as word addresses***.

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

a. For each of these references, identify the binary address, the tag, and the index given a direct-mapped cache with 16 one-word blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty.
b. For each of these references, identify the binary address, the tag, and the index given a direct-mapped cache with two-word blocks and a total size of 8 blocks. Also list if each reference is a hit or a miss, assuming the cache is initially empty

**QUESTION 3** (20%)

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 |

a. What is the cache block size (in words)?
b. How many entries does the cache have?
c. What is the ratio between total bits required for such a cache implementation over the data storage bits?

Beginning from power on, the following byte-addressed cache references are recorded.

| Address | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Hex | 00 | 04 | 10 | 84 | E8 | A0 | 400 | 1E | 8C | C1C | B4 | 884 |
| Dec | 0 | 4 | 16 | 132 | 232 | 160 | 1024 | 30 | 140 | 3100 | 180 | 2180 |

d. For each reference, list (1) its tag, index, and offset, (2) whether it is a hit or a miss, and (3) which bytes were replaced (if any)
e. What is the hit ratio?
f. List the final state of the cache, with each valid entry represented as a record of <index, tag, data>.

**QUESTION 4** (20%)

Below is a list of 32-bit memory address references, given as word addresses.

2, 3, 11, 16, 21, 13, 64, 48, 19, 11, 3, 22, 4, 27, 6, and 11

a. Show the hits and misses and final cache contents for a two-way set-associative cache with one-word blocks and a total size of 16 words. Assume LRU replacement.
b. Show the hits and misses and final cache contents for a fully associative cache with one-word blocks and a total size of 16 words. Assume LRU replacement.

**QUESTION 5** (20%)

Virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This excise shows how this table must be updated as addresses are accessed. The following data constitutes a stream of virtual addresses as seen on a system. Assume 4 KiB pages, a 4-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number.

| **Virtual Address** | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal | 4669 | 2227 | 13916 | 34587 | 48870 | 12608 | 49225 |
| Hex | 0x123d | 0x08b3 | 0x365c | 0x871b | 0xbee6 | 0x3140 | 0xc049 |

**TLB**

| Valid | Tag | Physical Page Number | Time Since Last Access |
|---|---|---|---|
| 1 | 11 | 12 | 4 |
| 1 | 7 | 4 | 1 |
| 1 | 3 | 6 | 3 |
| 0 | 4 | 9 | 7 |

**Page table**

| Index | Valid | Physical Page or in Disk |
|---|---|---|
| 0 | 1 | 5 |
| 1 | 0 | Disk |
| 2 | 0 | Disk |
| 3 | 1 | 6 |
| 4 | 1 | 9 |
| 5 | 1 | 11 |
| 6 | 0 | Disk |
| 7 | 1 | 4 |
| 8 | 0 | Disk |
| 9 | 0 | Disk |
| a | 1 | 3 |
| b | 1 | 12 |
| c | 0 | Disk |

For each access shown in the address table, list

a. Whether the access is a hit or miss in the TLB
b. Whether the access is a hit or miss in the page table
c. Whether the access is a page fault
d. The updated state of the TLB

**BONUS** (20%)

Continuing QUESTION 4, show the hits and misses and final cache contents for a fully associative cache with four-word blocks and a total size of 16 words. Assume LRU replacement.

**SUBMISSION**:

1. Clearly write your answers to the corresponding questions in a WORD or plain text file.
2. Submit your answers, clearly marked with your name, through eLearning by the due date.
3. Plagiarizing homework answers obtained from the internet or AI chatbots is not permitted.
4. **No late homework or assignment will be accepted**!

---

Yi Zhao