

**Submission Instructions:**

1. Prepare a PDF document with your solutions to the questions below. You may use the L<sup>A</sup>T<sub>E</sub>Xtemplate that is uploaded to Canvas, or any other tool. Please do not submit hand-written solutions since it will be hard to grade.
2. Ensure that each answer is clearly marked, and starting on a new page. This includes subparts of a question (e.g., Q2.1 on P3 and Q2.2 on P4, or Q1 on P1-2, Q2.1 on P3-4 and so on). Your answer may use multiple pages, but ensure that no single page has multiple answers.
3. Include the following acknowledgement on the first page of your submission, and sign it with your full name and date:

*"I affirm that the work I am submitting is my own. I have not copied or used any portion of another student's work. I have not used external sources without proper acknowledgement. If I have used AI tools, I will acknowledge the version used and explain how I used it."*

**Full Name:**

**Date:**

4. Submit the PDF on Gradescope, matching each question to the corresponding page.

## 1 L1 Misses [8]

Consider a computer system with a first-level data cache with the following characteristics:

- Size: 16 KBytes
- Associativity: Direct-Mapped
- Line Size: 64 Bytes
- Addressing: Physical
- Write Policy: Write-Allocate, Write-Back

The system has a separate instruction cache and you can ignore instruction misses in this problem. This system is used to run the following code:

```
for ( i = 0; i < 4096; i++) {
    X[ i ] = X[ i ] * Y[ i ] + C;
}
```

Assume that both X and Y have 4096 elements, each consisting of 4 Bytes (single-precision floating-point). These arrays are allocated consecutively in physical memory. The assembly code generated by a naive compiler is the following:

```
loop:    lw      f2 , 0(r1)      # load X[ i ]
            lw      f4 , 0(r2)      # load Y[ i ]
            multd f2 , f2 , f4      # multiply
            addd  f2 , f2 , f0      # add C (in f0)
            sw      0(r1) , f2       # store new value of X[ i ]
            addi   r1 , r1 , 4        # update address of X
            addi   r2 , r2 , 4        # update address of Y
            addi   r3 , r3 , 1        # increment loop counter
            bne   r3 , 4096 , loop  # branch back if not done
```

- (a) How many data cache misses will this code generate? Breakdown your answer into the three [2] types of misses. What is the data cache miss rate?
- (b) Provide a software solution that significantly reduces the number of data cache misses. How [3] many data cache misses will your code generate? Breakdown the cache misses into the three types of misses. What is the data cache miss rate?
- (c) Provide a hardware solution that significantly reduces the number of data cache misses. You [3] are free to alter the cache organization and/or the processor. How many data cache misses will your code generate? Breakdown the cache misses into three types of misses. What is the data cache miss rate?

## 2 Caches and Virtual Memory [8]

Consider an L1 cache which is 2-way set associative, has a capacity of 16 KB, and a block size of 64 bytes. Assume that both virtual addresses and physical addresses have 32 bits. Also, assume that the system has a page size of 4 KB.

1. Compute the size of the index and the size of the tag in bits. [2]
  2. Illustrate, with examples, the problems that can arise if the above L1 cache were to be implemented with a VI-PT (virtually indexed physically tagged) organisation? [4]
  3. Describe two ways to fix the above problems? [2]
-