**Pledge Honor:** I pledge my honor I have abided by the Stevens Honor System

**Final Exam 100 Points NAME: Alex Gaskins**

**Answer all questions in this document and upload in Canvas in 120 minutes.**

1. (7 Points) True or False, Justify: Developing multiprocessor architecture has proven to be easier than developing software to use them.

**True. The architecture for multiprocessor development is largely based on the number of threads with respect to the processors. Developing multiprocessor architecture involves designing hardware that can handle multiple tasks simultaneously by distributing the workload across multiple processors or cores. On the other hand, developing software that can effectively utilize multiprocessor architecture can be more challenging, which requires understanding the principles of parallel computing and optimizing the software to take advantage of the available processing power.**

2. (7 Points) True or false, justify: One advantage of the GPUs is that they do not require substantial memory bandwidth to achieve high performance.

**False. GPUs are highly parallel computing devices that are designed to perform large numbers of floating-point calculations in parallel. To achieve high performance, GPUs require a high memory bandwidth to keep up with the demands of the processing units. This is because the processing units in a GPU require a large amount of data to be transferred to and from memory in order to perform their calculations.In order to ensure that there is enough memory bandwidth available for the processing units, GPUs are typically designed with high-speed memory, and use specialized memory controllers that are optimized for high throughput.**

3. (7 Points) What is the major difference between fine-grain and coarse-grain multi-threading?

**Coarse-grain multi-threading involves allowing a thread to execute on a processor until a long-latency event, such as waiting for memory, occurs. At this point, the processor switches to another thread. On the other hand, fine-grain multi-threading switches between threads at a much finer level, such as between individual instructions.**

4. (7 Points) The performance of the shared memory multiprocessors is heavily dominated by two factors. What are they?

**Shared memory multiprocessors are a type of parallel computing architecture where multiple processors share a common memory space. In order for the processors to operate effectively and efficiently, they must maintain coherence with respect to the shared memory. This means that all processors must have consistent and up-to-date copies of the data in memory.**

**Cache coherence is a major factor that affects the performance of shared memory multiprocessors. Each processor in a shared memory system typically has its own cache, and maintaining coherence between these caches can be challenging. Cache coherence protocols are used to manage this process, but they can be complex and can have a significant impact on performance.**

**The architecture of the shared memory multiprocessor is also a key factor that affects performance. The interconnect between processors and memory can have a significant impact on performance, as can the design of the memory hierarchy and the caching and coherence protocols used.**

5. (7 Points) Why MIMD are less energy efficient than SIMD?

**SIMD only needs to fetch a single instruction for every data operation. For this reason, it is often used with mobile devices. It is designed in a way that allows devs to continue to think sequentially, as opposed to MIMD, and ultimately achieve parallel speedups.**

**MIMD (Multiple Instruction Multiple Data) is a parallel computing architecture that supports multiple instruction streams and data streams. It can be implemented as tightly-coupled MIMD with thread-level parallelism or loosely-coupled MIMD with request-level parallelism. MIMDs can be designed to function as single-user machines that prioritize high performance for one application, multiprogrammed machines that run multiple tasks simultaneously, or a combination of these functions.**

**Because of these structural differences, it is no surprise that SIMD are much more energy efficient than MIMD.**

6. (7 Points) What is Arithmetic Intensity of a program and what is it used for?

**Arithmetic intensity is a metric used to measure the efficiency of a program, which is calculated by dividing the number of floating-point operations required to run the program by the number of bytes accessed in main memory. Some computational problems, such as dense matrix calculations, have an arithmetic intensity that increases with problem size. However, there are also many other computational problems that have a fixed arithmetic intensity that is independent of problem size.**

7. (10 Points) Use Compiler Optimization Loop Interchange technique to re-write the following code in order to reduce cache miss rate:

```
for (j=0; j < 100; j = j+1)
    for (i = 0; i < 10000; i = i + 1)
    A[i][j] = 5 * A[i][j];
```

**In this case, the program accesses elements in column major order, which can result in poor performance due to a large number of cache misses. Since the array elements are stored in row major order, it is important to optimize the code so that elements are accessed in this order to improve locality of reference. The loop interchange technique can be used to achieve this optimization as follows:**

```
// Loop interchange
for (i=0; i<10000; i++)
      for (j=0; j<100; j++)
            A[i][j]=5*A[i][j]
```

8. (12 Points) A four-core i7 has a 8 MB L3 cache 16-way set associative of block size 64 bytes. It uses 30-bit block address (36-bit physical address, 6-bit block offset). What is the addressing organization of L3?

$$Number\ of\ sets = \frac{cache\ size}{(block\ size)(associativity)} = \frac{8MB}{(64)(16)} = 8192\ sets$$

$$Number\ of\ index\ bits = log_2\ (number\ of\ sets) = log_2\ (8192) = 13$$

$$Number\ of\ tag\ bits = Address\ bits - Number\ of\ index\ bits - Offset\ bits$$

$$= 36 - 13 - 6 = 17$$

←——---------------------------------------36 bits---------------------------------------——→

| Tag | Set Index | Offset |
|---|---|---|
| 17 bits | 13 bits | 6 bits |

9. (12 Points) Let’s assume a computer with L2 cache that:

    - Has a 128-byte cache block.
    - It takes 5 ns to transfer the first 8 bytes to L1, and then 1 ns per 8 bytes to transfer the rest of the block.

   Compare the miss penalty times with and without Early Re-start.

**Without Early Re-start:**

$5(1) + (16 - 1)(1) = 20\,ns$

**With Early Re-start:**

$\frac{(5+20)}{2} = 12.5\,ns$

10. (12 Points) Consider a dual core (A and B) caches with

    a. Snooping Protocol
    b. Write Invalidate
    c. Write through
    d. Write Allocate
    e. Variable x in memory with content 13

    In the table below, show the activity and cache and memory content as:

    - Core A reads x
    - Core B writes 26 to x
    - Core A reads x
    - Core A writes 3 to x
    - Core A writes 4 to x
    - Core B reads x

| Processor Request | Activity | A’s Cache & Status | B’s Cache & Status | Memory x |
|---|---|---|---|---|
| | | | | **13** |
| Core A reads x | **Cache read miss** | **13(E)** | | **13** |
| Core B writes 26 to x | **Cache write miss; Invalidate for x** | **13(I)** | **26(E)** | **26** |
| Core A read x | **Cache read miss** | **26(S)** | **26(S)** | **26** |
| Core A writes 3 to x | **Cache write hit; invalidate for x** | **3(E)** | **26(I)** | **3** |
| Core A writes 4 to x | **Cache write hit; invalidate for x** | **4(E)** | **26(I)** | **4** |
| Core B reads x | **Cache read miss** | **4(S)** | **4(S)** | **4** |

11. (12 Points) Assume a GPU with the following characteristics:

- Clock rate 3.2 GHZ
- Contains 16 SIMD processors, each containing 16 single-precision floating points units.

(a) What is the peak single-precision floating-point throughput for this GPU in GFLOP/second?

$Peak\ Throughput = 3.2(16)(16) = 819.2\ GFLOPS/s$

(b) What is the memory bandwidth required to support this peak GFLOP/second, assuming 40% of operations are addition (C=A+B) and 60% of operations are reciprocal (B=1/A).

**The operations would feature 2 operands for 4 byte inputs and 1 4 byte result = 12 bytes per FLOP.**

$Memory\ bandwidth = 12(819.2) = 9.8\ TB/s$