

CS 623 Semester 2025-26-I: Mid-Semester Exam

2–4 PM, September 19th 2026

Suggestions

- There are three pages. Answer questions exactly as asked, and be concise in your explanations.
- Your answers should be legible. Show logical steps and rough work in your computation.
- Explicitly mention ANY ASSUMPTIONS that you made for coming up with your solutions.

Questions

1. Consider the following C program segment.

```
1  for (i=0; i<32; i++) {  
2      if (x[i] < 0) x[i]++;  
3      else if (x[i] < 10) x[i] += 10;  
4      else x[i]--;  
5  }
```

Assume that x is an array of 32 integers initialized with random numbers. Suppose that each iteration takes time t . Assume that the general GPU architecture resembles that of an Nvidia GPU. Ignore pipeline stalls due to control hazards.

- (a) Assuming no predication, calculate the maximum and minimum achievable speedup (i.e., sequential time over parallel time) when the loop is parallelized among the threads in a warp of a GPU with each thread executing one iteration. **[4 points]**
- (b) Assuming predication, calculate the maximum and minimum achievable speedup (i.e., sequential time over parallel time) when the loop is parallelized among the threads in a warp of a GPU with each thread executing one iteration. Assume that all instructions of the loop can be predicated and that predication does not change the time to execute one iteration. **[6 points]**
- (c) Suggest one architectural feature that you would incorporate in the GPU without altering the instruction set architecture so that the speedup is always 32 for this loop. Explain if your proposed feature introduces any overhead. **[7+3 points]**
2. Suppose that the following C program segment is executed by thread i in a warp of 32 threads, where x is an integer array having 32 elements with a random subset of elements initialized to 0 and the other subset initialized to 1. Each subset is non-empty. The variable k is shared among the threads and initialized to 0.

```
1  while (x[i] == 1);  
2  k++;  
3  x[i] = 1;  
4  x[(i+1)%32] = 0;
```

What are the possible final values of k with and without independent thread scheduling? Assume that the general GPU architecture resembles that of an Nvidia GPU. **[10 points]**

3. Consider a GPU with a single streaming multiprocessor that can schedule and run one warp instruction every cycle. The streaming multiprocessor has an L1 data cache of size 96 KB, associativity six, and block size 128 bytes. The set index of the cache for a block address a is the remainder when a is divided by the number of cache sets. The L1 cache exercises the LRU replacement policy.

Each of the following loops is parallelized among 8192 threads in a single thread block by assigning iteration i to thread i . The translation of the loop bodies to an intermediate representation is also shown where each statement of the intermediate representation can be assumed to be an instruction and $r1, r2, r3, r4$ are registers.

	Intermediate representation
for (i=0; i<8192; i++) {	
a[i] = 1;	a[i] = 1;
b[i] = 2;	b[i] = 2;
c[i] = 0;	c[i] = 0;
}	
for (i=8191; i>=0; i--) {	
c[i] = c[i] - a[i]*b[i];	r1 = a[i];
	r2 = b[i];
	r3 = r1*r2;
	r4 = c[i];
	r4 = r4 - r3;
	c[i] = r4;
}	

Each array element is of size eight bytes and initially no portion of any of the arrays is resident in the cache. The arrays a , b , and c start at byte addresses 0, 0x10000, and 0x20000, respectively. The streaming multiprocessor has enough resources to keep all threads in flight. Each warp has 32 threads.

- Calculate the number of L1 cache misses for the aforementioned code snippet when using the loose round-robin and GTO warp scheduling algorithms. [10+10 points]
 - Determine the maximum warp group size (in terms of the number of warps) in two-level loose round-robin algorithm such that the number of L1 cache misses experienced by it would match that of the GTO algorithm. [10 points]
 - Based on these calculations, mention one drawback of the two-level loose round-robin scheduling algorithm compared to GTO. [5 points]
4. Assume that a kernel is launched with 10×10 blocks and $16 \times 16 \times 4$ threads per block. How many copies of a local variable will be created throughout the lifetime of the execution of the kernel? How many copies of a shared memory variable will be created throughout the lifetime of the kernel? [1+1 points]
5. A kernel performs 40 floating-point operations and 10 32-bit word global memory accesses per thread. Explain whether the kernel is memory- or compute-bound when run on a GPU with a peak performance of 200 GFLOPS and memory bandwidth of 100 GB/s. Ignore the effect of GPU caches. [2 points]

6. Consider the following CUDA kernel to perform a reduction. Assume variable `d_X` represents the array in global memory.

```
1 __shared__ float partialSum[SIZE];
2 partialSum[threadIdx.x] = d_X[blockIdx.x * blockDim.x + threadIdx.x];
3 int t = threadIdx.x ;
4 for (int stride = 1; stride < blockDim.x ; stride *= 2) {
5     __syncthreads();
6     if (t % (2 * stride) == 0) {
7         partialSum[t] += partialSum[t + stride];
8     }
9 }
```

Suppose the block size is 2048, warp size is 32, and the size of the array pointed to by `d_X` is a multiple of the block size. How many warps in a block will have divergence during the 5th iteration? [2 points]

7. A GPU device has a frequency of 1640 MHz and memory bandwidth of 328 GB/s. Assume the latency of memory access is 800 cycles. Consider the following kernel, which shows only the relevant code.

```
1 __global__ void kernel(double *d_A , double *d_B , int N) {
2     int tid = blockIdx.x * blockDim.x + threadIdx.x ;
3     if (tid < N) {
4         ...
5         d_B[tid] = 10 * d_A[tid];
6         ...
7     }
8 }
```

How many concurrent warps would you need for complete memory bandwidth utilization? Assume all the units are in powers of 10, i.e., $1K = 10^3$, $1M = 10^6$, and $1G = 10^9$. Ignore the effect of GPU caches. [4 points]