EE/CSCI 451
Fall 2020
Homework 7

Assigned: October 9, 2020
Due: October 16, 2020 AOE
Total Points: 100

1 [10 points]

Explain the following terms:

- Streaming Multi-Processor in a GPU
- SIMT
- Warp
- CUDA Core
- Kernel

2 [10 points]

If a kernel is launched with only one Warp (32 threads) in each thread block, the $\_\_syncthreads()$ instruction can be left out. Do you think this is correct? Explain.

3 [20 points]

A CUDA program has $2K$ threads. 3 GB of data need to be transferred from CPU to GPU and 1 GB of result data need to be transferred back from GPU to CPU. Data transfer is through PCIe whose bandwidth is 16 GB/s. The GPU has $1K$ CUDA cores and can run at most $1K$ threads in parallel. Each CUDA core runs at 1 GHz and is able to perform 1 multiply-add operation in each clock cycle. Each access to global memory takes 100 cycles. Assume each thread needs 10 memory accesses to the global memory and 100 multiply-add operations. What is the execution time of the CUDA program in the best case? Clearly state your assumptions.

4 [30 points]

Matrix-vector multiplication using CUDA

- Design a CUDA program to perform matrix-vector multiplication $c = A \times b$. The size of matrix A is $1K \times 1K$. The size of vectors b and c is $1K \times 1$. Your program should use $1K$ threads in total. Assume the shared memory is large enough to hold the entire vector b. The input matrix and the vector are initially stored in the host memory. Write a pseudo code for the host function and kernel function. Note that your kernel function must use shared memory to store vector b.

- Assume each element of A, b, c is 4 bytes; data transfer between CPU and GPU is through PCIe whose bandwidth is 16 GB/s in each direction; the clock rate of GPU is 1 GHz; the access latency to global memory and shared memory is 100 clock cycles and 10 clock cycles, respectively; multiply-add operations are overlapped with memory access operations. What is the execution time of your CUDA program in the best case?

5 [30 points]

Design a CUDA program to perform matrix multiplication $C = A \times B$. The size of each matrix is $1K \times 1K$. Each element is 1 byte. The matrices are initially stored in the global memory of GPU. The GPU has one streaming multiprocessor (SM) with $1K$ CUDA cores. Each CUDA core runs at 1GHz and can perform one floating point operation in each clock cycle. The peak bandwidth between the GPU and the global memory is 100 GB/s and the cache of GPU has been disabled.

- Design a simple CUDA program using straightforward vector inner product approach.
  - Write the pseudo code for your CUDA program including both kernel function and host function.
  - Derive a lower bound on the execution time of your kernel function? Explain.
- Assuming the GPU has an on-chip buffer (shared memory) which can store $3 \times 32 \times 32$ elements and has a peak access bandwidth of 1 TB/s.
  - Write the pseudo code for an optimized CUDA program using block matrix multiplication approach including both kernel function and host function.
  - Derive a lower bound on the computation time, the local data access time in SM, and the global data access time of your optimized kernel function. Explain.