IIT JODHPUR

Minor Examination: CSL7520 GPU Programming

Guidelines (Total time: 120 minutes, Maximum Marks: 25):

- Please read the question paper very carefully. **NO clarification is required in any question.** In case of any doubt, assume whatever you wish to and state that in your answer.
- Usage of LLM/AI tool is NOT allowed.
- Hints are provided in few questions. Please follow them.

---

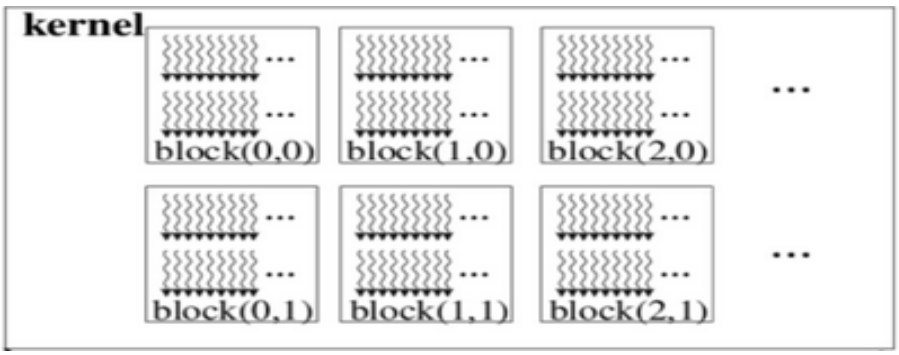
1. You are working as an application developer for a software company *PicoSoft Technologies*. You have developed a special application that takes as input an image and then provides a caption and describes the image contents in three/four sentences. You are ideally targeting a GPU for running this application, however, you wish to explore first if CPU (referred as processor) can also do the job. You are provided 2 options for executing your innovative Image-to-Text software/application:

   - Processor A has 4 cores with a clock speed of 3 GHz and a CPI of 2.
   - Processor B has 8 cores with a clock speed of 2.5 GHz and a CPI of 2.5.

   Your program consists of 1.5 billion instructions (in serial version)). How much of the program you must parallelize for Processor A to match the performance of Processor B? [**Hint**: Execution time on any processor (per core) is given by the product of number of instructions, CPI and clock period. Also, clock period is the reciprocal of the clock frequency. CPI refers to the cycles incurred per instruction.] [3.5]

2. Assume that you are leading a team in *PicoSoft Technologies* that is responsible for a stock market price optimization software package development. When developed in parallel manner, this application ultimately translates to the following number of parallel tasks/operations: load operations=5K, store operations=4.5K, FP additions=2.5 K and FP multiplications=3K. If we aim to complete execution of this program in 3 seconds total, how many **warps** are required when we are running this program on any GPU given that each task/operation takes 5 clock cycles (average). [2.5]

3. While writing parallel program for a video game, you decide to launch 64 thread-blocks with 32 threads in each thread-block (totaling 2048 threads, numbered for 0 to 2047). Therefore, each thread (as shown below) can be identified by an index-pair (index of thread-block, index of thread with in particular thread-block). Find the index pair of the following thread numbers(ids): 252, 501, 1013, 1515? [4]



4. Considering your role as technical lead at a investment consulting firm, you are aiming at release of a new software for stock price prediction. A few interns suggest to you that parallel programming should be considered for the implementation of one of the core algorithms for stock price prediction. This algorithm involves matrix multiplication heavily. Considering the matrix sizes as 512x500 and 500x256, what would be your rough estimate regarding the speed-up in parallel implementation (running on a GPU) v/s the serial implementation (running on a CPU). For reference, you can assume that clock to both the CPU

and the GPU is running at a frequency of 1.2 GHz. On the CPU, each multiplication takes 3 cycles and addition takes 1 cycle. On the GPU, you can launch multiple kernels of 32 thread-blocks with each thread-block having 128 threads in it. You can roughly assume that each thread finishes in 2 cycles on the GPU (regardless of the involved operation). As you are just looking for a rough estimate, you are willing to ignore the latency of memory accesses both in case of GPU and CPU. [4]

5. Fused Multiply-Add (FMA) operations are an integral part of deep learning algorithms. Therefore, optimization of FMA operations is an crucial factor for performance enhancement of these models during the inference stage. The figure below shows the implementation of FMA operation in the naive manner (shown in a) and the tiled re-arrangement of the loops (shown in b). #pragma PIPELINE refers to a compiler directive that is typically used in high-level synthesis of hardware designs and hence can be IGNORED in this question. Actually, this refers to a command for pielining the loop iterations.

```
1 for (int n = 0; n < N; ++n)
2   for (int m = 0; m < M; ++m) {
3     double acc = C[n][m];
4     #pragma PIPELINE
5     for (int k = 0; k < K; ++k)
6       acc += A[n][k] * B[k][m];
7     C[n][m] = acc; }
```

(a) Naive implementation of general matrix multiplication

```
1 for (int n = 0; n < N; ++n)
2   for (int m = 0; m < M/T; ++m) {
3     double acc[T]; // Tiles of size T
4     for (int k = 0; k < K; ++k)
5       double a = A[n][k]; // M/T reads
6       #pragma PIPELINE
7       for (int t = 0; t < T; ++t) {
8         double prev = (k == 0) ?
9             C[n][m*T+t] : acc[t];
10        acc[t] = prev + a * B[k][m*T+t]; }
11    for (int t = 0; t < T; ++t) // Write
12      C[n][m*T+t] = acc[t]; }   // out
```

(b) Tiled iteration (looping) space, same location written every T cycles

Given the above code representation for FMA implementation, prepare a plan for the CUDA-style parallel implementation of both the versions in (a) and (b) so that they can be executed on GPU. You DO NOT need to provide the exact CUDA program, you just need to show the **skelton CUDA code** for the parallel implementation of (a) and (b). Also, provide the speed up that the parallel implementation of (b) is able to achieve in comparison to the parallel implementation of (a)? [2 + 3 + 2]

6. The Fermi and Kepler GPU architectures from the Nvidia company are compared in the below table.

GPU MODEL	INSTRUCTION LATENCY (CYCLES)	BANDWIDTH (GB/SEC)	BANDWIDTH (B/CYCLE)	PARALLELISM (KB)
Fermi	800	144	92	74
Kepler	800	250	96	77

Consider the clock frequency of Fermi architecture is 1.5 GHz and for Kepler architecture, it is 6GHz. Assuming that each thread carries 1 floating-point data with single precision (4 bytes) from Global Memory to SM for computation, find the **number of warps** required for both these architectures to hide the memory latency (i.e., delay due to transfer of data)? You are given that 1 warp=32 threads. [4]

Hint: i) The parallelism (last column) required for memory operations can be obtained by multiplying instruction latency with the data transferred per cycle from memory. ii) Hiding memory latency means ensuring that the computational blocks are NOT waiting for memory at any instant of time. iii) Hz is unit to measure frequency. 1 Hz is defined as 1 cycle per second. So, 1 GHz is 10^9 cycles per second.
