LAB 4

DUE DATE: Fri Oct 20 5pm (upload to polylearn) Individual Project

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Write a program in CUDA to perform Matrix Addition.

In class we did vector addition using the Nsight Studio IDE, your job is to transform the code to perform Matrix addition (2D) .

To lunch a kernel with threads o the x and y direction

// Launch the Vector Add CUDA Kernel

**int** threadSize=16;

**int** gridSizeX=((hight-1)/threadSize)+1;

**int** gridSizeY=((width-1)/threadSize)+1;

**const** dim3 blockSize(threadSize, threadSize, 1); //**TODO**

**const** dim3 gridSize(gridSizeY, gridSizeX, 1); //**TODO**

**vectorAdd**<<<gridSize, blockSize>>>(d\_A, d\_B, d\_C, width,hight);

Note :In the class slides (2.1) you have some information about how to do it

Note II: Instructions about how to remote login to the lab and run profiler, debugger etc from the command line are uploaded to polylearn instructions folder

NOTEIII: The maximum number of threads you can run per block is 1024. so remember that when you run a block of (256,256) you are running 65536 threads per block!!!!!!! and your ernel will complain

Write a report with your name :

1. Use the Profiler to report the times for

|  |  |
| --- | --- |
| Matrix A and B Size | Threadsperblock |
|  | 256 512 1024 |
| A.512\*512 |  |
| B. 1024\*1024 |  |
| C. 8192\*8192 |  |

2. Screen Capture the program running for C. showing that you pass the test

3. Appendix with your Kernel code, clean and properly commented (I may ask for a demo of your code working on the computer labs if I don’t understand how your code is able to run).

4. Answer the following questions:

a. If we need to use each thread to calculate one output element of a vector addition, what would be the expression for mapping the thread/block indices to data index

1. blockIdx.x+threadIdx.x

2. blockIdx.x\*blockDim.x+threadIdx.x

3. blockDim.x\*threadIdx.y+threadIdx.y

4. blockIdx.x+lockDim.x

b. We want to use each thread to calculate two (adjacent) elements of a vector addition. Assume that variable I should be the index for the first element to be processed by a thread. What would be the expression for mapping the thread/block indices to data index?

1. blockIdx.x\*blockDim.x+threadIdx.x +2

2. (blockIdx.x\*blockDim.x+threadIdx.x)\*2

3. blockDim.x+threadIdx.x

4. blockIdx.x\*2\*blockDim.x+threadIdx.x

c. For a vector addition, assume that the vector length is 2000, each thread calculates one output element, and the thread block size is 512 threads. How many threads will be in the grid

1. 2000

2. 2024

3. 2048

4. 2096