CO332 – Heterogeneous Parallel Computing

A1 – CUDA Introduction

Reference: Hwu, Programming Massively Parallel Processors, 2e. Module 2.
Deadline: 9AM, August 04, 2017. Send your assignment report to co332.nitk@gmail.com.

Q1. Write the device query code, compile and run it on your system. Query enough information to know all the details of your device. Example queries: GPU card's name, GPU computation capabilities, Maximum number of block dimensions, Maximum number of grid dimensions, Maximum size of GPU memory, Amount of constant and share memory, Warp size, etc. Answer the following questions in your report.

1. What is the architecture and compute capability of your GPU?
2. What are the maximum block dimensions for your GPU?
3. Suppose you are launching a one dimensional grid and block. If the hardware's maximum grid dimension is 65535 and the maximum block dimension is 512, what is the maximum number threads can be launched on the GPU?
4. Under what conditions might a programmer choose not want to launch the maximum number of threads?
5. What can limit a program from launching the maximum number of threads on a GPU?
6. What is shared memory? How much shared memory is on your GPU?
7. What is global memory? How much global memory is on your GPU?
8. What is constant memory? How much constant memory is on your GPU?
9. What does warp size signify on a GPU? What is your GPU’s warp size?
10. Is double precision supported on your GPU?

Q2. Write a CUDA program to calculate the sum of the elements in an array. The array contains single precision floating point numbers. Generate your input array. For this question, do the following.

1. Allocate device memory
2. Copy host memory to device
3. Initialize thread block and kernel grid dimensions
4. Invoke CUDA kernel
5. Copy results from device to host
6. Free device memory
7. Write the CUDA kernel that computes the sum

Q3. Write a CUDA program to calculate the element-wise sum of two large (16K elements or greater) arrays. The arrays contain single precision floating point numbers. Generate your input arrays.

Q4. Perform Matrix Multiplication of two large integer arrays in CUDA. Answer the following questions.

1. How many floating operations are being performed in your matrix multiply kernel?
2. How many global memory reads are being performed by your kernel?
3. How many global memory writes are being performed by your kernel?
4. Describe what possible optimizations can be implemented to your kernel to achieve a performance speedup.