CUDA Introduction!

Class: CS315 – Distributed Scalable Computing
Whitworth University, Instructor: Scott Griffith

Last Modified: 10/25/2018

Part 1: Device Properties (20 pts)

Before we starting coding any kernels, we should probably survey our system and look into the GPU device (or possibly devices) that we have access to.

Using both cudaGetDeviceCount() and cudaGetDeviceProperties() query the system for relevant information regarding the GPUs attached.

Your code should output at a minimum the following properties in a well formatted way: name of device, global memory capacity, shared memory maximum, warp size, maximum threads per block, maximum dimensions for thread blocks, maximum dimensions for grid of blocks and compute capability.

(5 pts) Is there a correlation between any of the above? I might suggest thinking about the maximum threads per block, the warp size and the specifications for your GPU (you are going to have to look those up externally).

Part 2: SAXPY Performance (80 pts)

You are going to implement a Single Precision A*X + Y (SAXPY) kernel. This is a basic linear algebra operation. X and Y are both vectors, A is a constant. So: z[i] = a*x[i]+y[i] for all elements in X/Y/Z.

First: get your **basic implementation** down and working. This will require you to generate test data, and verify your output data. You should be operating on sizes of **N at least 2^16**.

Then implement some **performance gathering infrastructure**. You are going to do this using CUDA Events. We will eventually talk more about these, but I would suggest looking at https://devblogs.nvidia.com/how-implement-performance-metrics-cuda-cc/ as a reference, specifically the Timing using CUDA Events section.

A couple of tips: Make sure you are fully testing the functionality of your code. Make sure your verification is accurate and that you are performing the operation on all elements!

Make sure that your timing event is directly before and directly after your CUDA kernel call. We don't want to record the time of your verification.

(20pts) perform a number of performance tests on your code and report what happens with various sizes and configurations of threads and blocks. Did you find a correlation at all? Remember the architecture of the GPU is based on warps!