ECEC 413: Introduction to Parallel Computer Architecture
CUDA Lab 2: Data Parallel Reduction

Prof. Naga Kandasamy, ECE Department, Drexel University

February 22, 2010

The Lab is due March 1, 2010. You may work on the assignment in teams of up to two people.

Important: Before starting the lab assignment, please read the tutorial on getting the CUDA SDK working on the cluster. You will find the tutorial (written by Tony Shackleford) on webCT under the "Labs" folder.

Your lab assignment comprises of two parts as described below:

- **Part 1:** Edit the source files `vector_reduction.cu` and `vector_reduction_kernel.cu` to complete the functionality of the parallel addition reduction on the device using the technique shown in Fig. 6.2 in the textbook. The size of the array is guaranteed to be equal to 512 elements for this assignment. The array must be loaded from GPU global memory. The use of shared memory is not required for this assignment.
- **Part 2:** Repeat Part 1, except in this case, use the kernel design shown in Fig. 6.4.

Your program should accept no arguments. The application will create a randomly initialized array to process. After the GPU kernel is invoked, it will compute the correct solution value using the CPU, and compare that solution with the GPU-computed solution. If the solutions match (within a certain tolerance), it will print out "Test PASSED" to the screen before exiting.

You must e-mail me all of the files needed to compile and run your code as separate zip files—one zip file for Part 1 and one zip file for Part 2.

Also, once you have completed the assignment, answer the following questions in a separate lab report.

- Compare the execution times achieved by the two different reduction kernels and comment on why you think one kernel performs better than the other.
- Answer this question for Parts 1 and 2: How many times does your thread block synchronize to reduce the array of 512 elements to a single value?
- Answer this question for Parts 1 and 2: What is the minimum, maximum, and average number of useful operations that a thread will perform? Useful operations are those that directly contribute to the final reduction value.

Your assignment will be graded on the following parameters:

- Correctness: 25%.
- Functionality: 30%. Your kernels uses an $O(n)$ operation data-parallel reduction algorithm.
- Report: 45%. Answer to question 1: 25%, answer to question 2: 10%, answer to question 3: 10%