Study Guide

Week 12 November 9th – November 15th, 2015

Author B. Wilkinson Modification date November 8th, 2015

Study Materials on Moodle

- *PowerPoint Slides*
    - CUDA synchronization
    - GPU memory structures
    - Performance Analysis
    - Memory coalescing
    - Using Shared Memory
- *Videos*
    - Lecture 21 video: 75-minute video of Lecture 21 in Fall 2014 on *CUDA synchronization* and *GPU memory structures*. (The beginning of the lecture is missing, talking about Assignment 7.)
    - Lecture 22 video: 75-minute video of Lecture 22 in Fall 2014 on *Performance Analysis*, *Memory Coalescing*, and *Shared Memory*.
- *Sample Quiz Questions*
    - CUDA quiz questions (continued)

Tasks

- **Mini-Quiz:** Answer the short posted quiz before 11:55 pm Sunday November 15th, 2015.
- **Complete Assignment 6** Seeds Framework
    - Assignment 6 Due: *Sunday November 15th, 2015 (Week 12)*
- **Assignment 7** CUDA
    - Assignment 7 Instructions
    - Assignment 7 Set: *Sunday November 15th*,, 2015 (Week 12)
    - Assignment 7 Due: *Friday December 4th, 2015 (Week 15)*

Moodle Saba meeting – Friday November 13th, 2015, 6 pm

CUDA (Thread) synchronization describes how to synchronize threads. There are two principal ways:

1. Using the **cudaThread Synchronize()** routine in the host that waits until all the threads in the device kernel have finished (technically all streams have completed), and
2. Using **__synchthread()** routine in the kernel, which synchronizes only those threads in each block, i.e. threads wait for all the threads in the same block to complete before continuing not for threads in other blocks.

There are other ways to synchronize in later versions of CUDA. The name, **cudaThread Synchronize()**, is deprecated, and the routine is now called **cudaDeviceSynchronize()**, but only to make it clearer what it does.

GPU memory structures outlines the use of global device memory and shared memory, and in particular the concept of memory coalescing, which is continued in *Memory Coalescing*. The idea is to combine multiple memory accesses into one transaction. You are expected to appreciate and be able explain this concept.

Performance Analysis describe various low level tools to access the performance of parallel programs. These notes are not specific to GPUs and CUDA and describe mostly MPI tools and general profilers. There are CUDA-specific tools offered by NVIDIA, added to Eclipse and Visual Studio. We do not actually use these tools in the CUDA assignment, except instrumenting the code to measure execution time, which was covered in Lecture 20. Generally we would use CUDA events to measure time, not as described in these slides.

Memory Coalescing describes some experiments done to see the effect of memory coalescing, with actual results on a UNC-C GPU server (**coit-grid06.uncc.edu**, which 448-core NVIDIA C2050 GPU but a very old CPU).

Using Shared Memory describes some experiments done to see the effect of shared memory with actual results on a UNC-C GPU server **(coit-grid06.uncc.edu**)

Assignment 7 Instructions: This assignment does not use the course VM and has to be done on one of the the GPU servers within the **cci-gridgw** cluster unless you have an NVIDIA GPU card and software installed on your computer. So we can expect issues with the servers. You have almost three weeks to do this assignment (including over the Thanksgiving break) **BUT PLEASE DO NOT WAIT TO THE LAST DAY**. Part 1 is trivial and asks you to compile and execute a sample program. For Part 2, the CUDA program for matrix multiplication can actually be found in the slides in Week 11 ("11.4 CUDA_MultiDimBlocks") and can be used. Note the use of dynamic allocated memory on the GPU and flattened 1-D indices to access 2-D arrays. You then have to add code for keyboard input. The code for Part 3 is not given, but a data parallel algorithm and pseudo code are given in the slides in Week 11 ("11.1_DataParallelPattern"). Read the footnote on slide #9. It turns out the algorithm is not efficient but is acceptable. *Do not use code you might find on the NVIDIA site as this is much more complicated than is required and does not satisfy the assignment. Anyway you are not to copy code from the Internet and you will get zero for the assignment at the very least if you do so. You are expected to develop the code yourself.*

There is an extra part for everyone (+20%!)

This brings us to the end of GPUs and CUDA, and the course materials. Next week (Week 13), we will have a review and the second class quiz (40-question Moodle quiz).