GFCUDA

Introduction/Teaser

General Purpose Graphical Processing Units (GPGPUs) are developing into a powerful and efficient computing platform. They are especially suited for data-processing applications, many of which can be modeled as matrix operations. In some cases, it can be advantageous to perform operations using groups, rings, or fields other than the integers or reals. Solving this problem requires implementing matrix operations over a Galois field on CUDA-enabled GPGPUs.

Background

GPGPUs are coming into their own as powerful, cost-effective, and energy efficient solutions to data-intensive computing problems. They work by employing a large number of stream processors, which on their own are somewhat limited, to leverage massive parallelism. There are a number of programming systems for them, but for this problem we highly recommend using the CUDA (Compute Unified Device Architecture from NVIDIA) extension to C/C++.

Matrix multiplications can be found everywhere in scientific computing, with Matrix-Matrix Multiply being one of the more fundamental operations. For large matrices, these algorithms lend themselves to fine-grain parallelism found in multi-core systems. For this reason, Matrix-Matrix Multiply is one of the more highly developed algorithms on GPGPUs.

For some numerical problems, it becomes advantageous work on a Galois field, or finite field. Galois fields allow algorithms to control the size of their primitive chunk of data while still being able the more powerful mathematical constructs in algebra. For instance, if one wanted to work with a message of alpha-numeric characters, one could store those as 6-bit integers (64 possible values) to save space while maintaining the ability to work the message numerically. This will be explained in greater depth in the solution.

Problem

You task is to implement Matrix-Matrix multiply over a 256-element Galois field (GF256) for a GPGPU system. Your objectives are 1) correctness and 2) speed. Your score will be based on those two criteria, the sophistication/elegance of your algorithm, and your programming style. Attached is a driver and serial version of the code. Read the comments for computational details. Because you are getting someone else's code, you will almost certainly have questions and are encouraged to ask them.