High-Performance Stream Computing for Particle Beam Transport Simulations

R. Appleby, D. Bailey, M. Salt

School of Physics and Astronomy, University of Manchester, Oxford Road, Manchester, M13 9PL, UK

Robert.Appleby@manchester.ac.uk, David.Bailey-2@manchester.ac.uk

Abstract. Understanding modern particle accelerators requires simulating charged particle transport through the machine elements. These simulations can be very time consuming due to the large number of particles and the need to consider many turns of a circular machine. Stream computing offers an attractive way to dramatically improve the performance of such simulations by calculating the simultaneous transport of many particles using dedicated hardware. Modern Graphics Processing Units (GPUs) are powerful and affordable stream computing devices. The results of simulations of particle transport through the booster-to-storage-ring transfer line of the DIAMOND synchrotron light source using an NVidia GeForce 7900 GPU are compared to the standard transport code MAD. It is found that particle transport calculations are suitable for stream processing and large performance increases are possible. The accuracy and potential speed gains are compared and the prospects for future work in the area are discussed.

Introduction

The dynamics of a charged particle within a particle accelerator may be modeled as a succession of matrix – vector operations, plus second order perturbations. Both the position and momentum of the particles are given by a vector in six-dimensional phase space. This is the method employed by industry standard code, 'Methodical Accelerator Design' (MAD). The magnetic elements of the accelerator act upon the vector as given in the Taylor series.

$$z_j = \Delta x_j + \sum_{k=1}^{6} R_{jk} x_k + \sum_{k=1}^{6}\sum_{l=1}^{6} T_{ijk} x_k x_l$$

In understanding the function of the accelerator, it is beneficial to model the motion of many millions of particles. This is a highly parallel task, and therefore well suited to stream processing techniques. Generally, the distribution of the vector values forms a Gaussian distribution. The Booster to Storage ring on the DIAMOND accelerator in Oxford was used as a test case for this example. This is used to contrast performance and precision compared to the current methods employed.

Implementation

The GPU implementation of the program was written using Brook, which originates from Stanford Graphics Laboratory. Brook uses the shader language, and the concept of graphical 'streams' to permit parallel computation. Being derived from C, much of the syntax is familiar, and it is able to interface with standard C++, which makes up the bulk of the supporting code. Codes written in this language are compiled using the Brook Runtime C Compiler, which produces standard C code. The matrix-vector operations take place in the Brook 'kernel', which is called from the main code to act upon the data. Transfer of the particle arrays between main memory and the kernel are implemented using the 'streamRead' and 'streamWrite' commands.

For particle tracking, a beam-line is read from a file that is in the standard XSIF parser format. The parameters of this are translated into floating point numbers in a 6 x 6 matrix, which are passed into the kernel for use in computation. In this particular example, the particles were evolved through all 94 elements of the DIAMOND BTS beamline, and compared with the equivalent result from MAD.

Results – single particle tracking

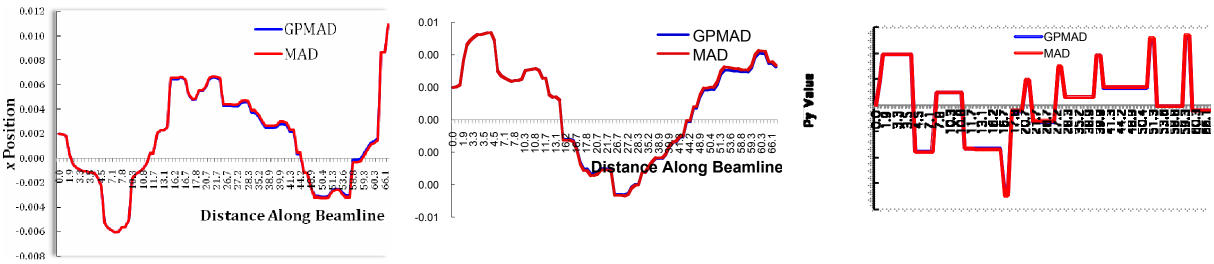

Figure 1, Single particle tracking of x position, y position and momentum in x

Throughout the beamline, the position of the particle in MAD and in the GPU-based code incurs very little deviation, thus suggesting that there is no precision problem.

Results – high statistic tracking

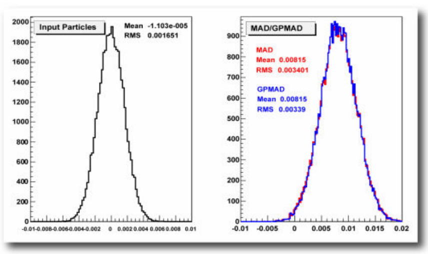

Figure 2, High statistic tracking, distribution of x at input and comparison between MAD / GPMAD at output

At input, the x-position of the particles is of the form of a Gaussian distribution. By tracking through the whole lattice and looking at the distribution at the end, it is possible to confirm the suitability of the GPU for high-statistic tracking.

Results – Performance gains

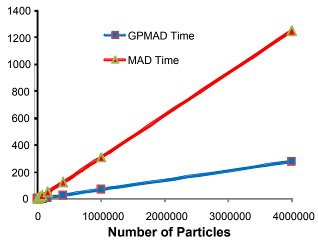

Figure 3, Performance scaling as number of particles increases

At high particle numbers, such as a few million, the GPU-based program completes in only a fraction of the time MAD took for the same task.

Conclusions and Prospects

The GPU may be used as a powerful tool for particle accelerator simulation, particularly with very high statistics. The code is currently being re-written to leverage the NVidia CUDA development environment on a modern GForce 8800 GPU. First results from the CUDA implementation will be presented, and strategies for taking the work further will be discussed.