Particle-in-Cell Simulations on Modern Computing Platforms, V. K. Decyk and T. V. Singh, UCLA.*

With the introduction of many more processors, High Performance Computing (HPC) is undergoing a revolution in hardware and unrest in software. To develop strategies on how scientific developers can adapt to this rapidly changing environment, we have developed a number of compact 2D Particle-in-Cell (PIC) applications. This allows us to evaluate different hardware and software environments, as well as to develop new approaches which can work in these varied environments. We started with NVIDIA Graphical Processing Units (GPUs) and the Cuda C language. This effort resulted in an adaptable algorithm based on fine-grained partitions (a few cells), and obtained speedups of up to a factor of 60 [1]. OpenCL is intended as a portable language for such devices. With the OpenCL version of our PIC code, we evaluated 4 different OpenCL implementations on 4 different hardware platforms. Finally, we also ported our code to Cuda Fortran, provided by the Portland Group (PGI). In general, Cuda C gave the best performance, with Cuda Fortran a close second. Cuda Fortran was somewhat simpler to program, however, due to the use of Fortran90 and Fortran2003 features.

At the same time, we have also not abandoned more traditional approaches based on MPI and shared memory languages such as OpenMP. Although OpenMP has usually given worse performance than MPI, we have implemented an OpenMP PIC code which gives performance very close to that of MPI. There are actually two different methodologies in using OpenMP, a traditional style using parallel loops, as well as an alternative SPMD style. Finally, we have implemented a hybrid MPI/OpenMP code which uses nested domain decompositions which allow for very fine-grained domains with reduced communications requirements. In the hybrid approach, each shared memory node has its own domain decomposition, with an MPI domain decomposition “to rule them all...” Nested domains three layers deep may be useful with clustered GPUs.

All these varying approaches have been valuable: lessons learned in one environment have been useful in another. There is hope that a common approach with adaptable algorithms will work in this new world.

*Work supported by USDOE and UCLA IDRE.

[1] V. K. Decyk and T. V. Singh, “Adaptable Particle-in-Cell algorithms for graphical processing units,” Computer Phys. Comm. 182, 641 (2011).