Max-Planck-Institut für Plasmaphysik

GPU enabled EUTERPE

Nils Moschüring[1] Matthias Borchardt[1]
[1]Max-Planck-Institute for Plasma Physics

This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No 101052200 — EUROfusion). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

Introduction

- HLST project MULTIKIN in 2020
- enabling GPU for EUTERPE
- different path
  - language based (Kokkos, CUDA, OpenCL, Julia)
  - directive based (OpenMP 4.5, OpenACC)
- Nils selected Kokkos for investigation
- implementing a toy problem in Fortran (one dimensional PIC solver)
- Kokkos C++ library leads to a complex Fortran layer
- not efficient without porting EUTERPE to C++
- decision last year to use OpenACC

GPU EUTERPE

initial approach

- calculate electrons parallel to ions on GPU

Parts on GPU

- particle pushing
- charge assignment
- particle communication
- cache sorting

Difficulties

- fractured code (many interdependent modules)
- widely used global variables (declaration and updating)
- inlineable functions and subroutines for parallel regions

Changes

Types of parallelization

- using MPI for domain decomposition
- using OpenMP for CPU compiled parts in order to use multiple cores of the CPU
- using CPU/GPU for different types of particles

- equilibrium data accessible on GPU
- introducing species specific parts
- avoid code duplication
- particle dependent subroutines
- particle communication after pushing
- time integrators need particle dependent copying and calculating of particle properties
- disable cache optimization due to doubled memory requirements

Further developments

- pushing and charge assignment of ions and electrons asynchronously
- after first tests it was better to put ions to GPU instead of electrons
- Nils implemented a compile dependent distribution of particles to GPU or CPU
- since September code is maintained in Greifswald
- introduce real version of the code
- all test cases runable with debug options
- code now gives same results for all actual test cases

Outlook

- merge with current master
- production runs to check stability
- rewrite cache optimization
- remove global variables from modules
- implement more strict programming style
- employing OpenMP on more places
- execute more parts asynchronously
- data structures
- solver (PETSc) to GPU?