View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by UPCommons. Portal del coneixement obert de la UPC

Plasma Physics Code Contribution to the Mont-Blanc Project

Xavier Sáez
Barcelona Supercomputing Center (BSC-CNS), Spain
xavier.saez@bsc.es

Alejandro Soba
Centro de Simulación Computacional para Aplicaciones Tecnológicas (CSC-CONICET), Argentina
soba@cnea.gov.ar

Mervi Mantsinen
Barcelona Supercomputing Center (BSC-CNS), Spain
Institució Catalana de Recerca i Estudis Avançats (ICREA), Spain
mervi.mantsinen@bsc.es

Abstract*-*This work develops strategies for adapting a particle-in-cell code to heterogeneous computer architectures and, in particular, to an ARM-based prototype of the Mont-Blanc project using OmpSs programming model and the OpenMP and OpenCL languages.

I. Introduction

The scientific grand challenges, such as fusion reactors, have been the driving force for the evolution of High-Performance Computing (HPC).

During the last two decades, the supercomputers have grown rapidly in performance to provide scientists the required computing power, at the cost of a similar growth in power consumption.

However, nowadays the computer's performance is limited by the power consumption and power density, so new developed platforms to construct a future sustainable exaflop supercomputer will have to be based on the power efficiency. The Mont-Blanc project appeared with the aim to design computer architectures capable of delivering an exascale performance using 15 to 30 time less energy than present architectures [1].

Particle-in-cell (PIC) is one of the most used methods in plasma physics simulations [2]. The quality of results achieved by this method relies on tracking a very large number of particles. Therefore, PIC codes (such as EUTERPE) are good candidates to be adapted to new HPC platforms since they require intensive computation.

II. Particle-in-Cell Code

A. *Particle-in-cell Methods*

PIC methods are used to model physical systems whose behavior varies over different ranges of spatial scales. The individual particles are tracked in a continuous phase space, whereas densities and the current are computed concurrently on stationary mesh points.

A PIC algorithm can be summarized in three steps (Fig.1) repeated at each time step [3]:

- **pull**: the particle properties are interpolated to neighboring points in the computational mesh.
- **solve**: the moment equations are solved on the mesh.
- **push**: the momentum of each particle is calculated by interpolation on the mesh. The particle properties are updated.

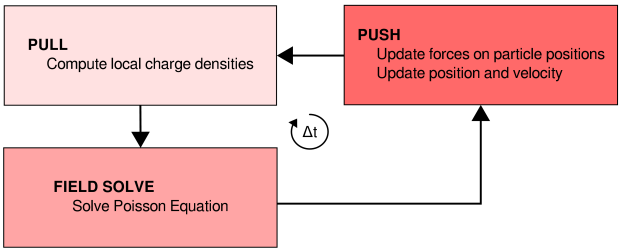

Fig. 1. The steps of a PIC algorithm.

B. *EUTERPE Code*

EUTERPE is a gyrokinetic PIC code for global linear and non-linear simulations of fusion plasma instabilities in three-dimensional geometries, in particular in tokamaks and stellarators [4, 5].

It has been written to target traditional HPC clusters using MPI and a domain cloning technique to increase the number of processors without boosting the interprocessor communications to prohibitive levels [6].

III. Computer Prototype

Mont-Blanc is a European exascale computing approach to develop a full energy-efficient HPC prototype. The project is coordinated by Barcelona Supercomputing Center (BSC) since October 2011.

The aim is reducing energy consumption using low-power commercially available processors that were designed for mobile and embedded systems. In this way, we exploit their cheapness due to the large volume of these platforms and their high accessibility in the commodity market.

The prototype used in this work is based on a system-on-chip (SoC) Samsung Exynos 5 which contains an ARM Cortex-A15 dual core and an ARM Mali T604 GPU [7].

This embedded SoC with integrated GPU accelerator is the first that could be used for HPC, since it supports 64-bit floating point arithmetic and provides support for parallel programming languages (such as OpenCL 1.1).

IV. Achievements

We have developed three versions of the EUTERPE code for this prototype:

A. *Hybrid version*

EUTERPE was only parallelized at task level using MPI. For that reason, we developed a hybrid version of the code introducing OpenMP to take advantage of all the levels of parallelism that a multi-core architecture can offer [8].

B. *OpenCL + Hybrid version*

In order to obtain the best possible results on the prototype, all the available resources were used by the application: the dual core CPU (using previous hybrid version) and the GPU (using a new OpenCL version).

We only wrote the most time-consuming routines to OpenCL with the aim to minimize the necessary changes. In the *push* part, one work-item was assigned to each particle. The *pull* part was more challenging to implement, since different particles could contribute to the charge density on the same mesh point. To avoid these memory conflicts, a mesh copy was created per work-group, so the lock contention was far minor.

C. *OmpSs version*

When we port a code to a new platform not only is important the possibility to reach the maximum performance of the platform, but also the ease of programming it. The main drawback of OpenCL is its low programmability because of it is a low-level programming language.

To address this shortcoming, we tested OmpSs, a task-based programming model developed by Barcelona Supercomputing Center (BSC) [9]. It provides an abstraction to the user thereby reducing programmer effort and unifies the SMP, heterogeneous and cluster programming in one model.

Although OmpSs version is a bit slower than the hybrid version, it is a simpler version and its productivity has improved considerably (see Table I).

TABLE I
Comparison Between the Different Versions

Routine	Performance - Best time (s)		Programmability / Productivity	
	Hybrid version + OpenCL	OmpSs version	OpenCL API calls	OmpSs directives
Push	6.02	6.92	161	12
Pull	10.31	10.83	167	18

V. Conclusions

This work confirmed that is possible to port this kind of plasma physics codes to an ARM-based platform and we can say that OmpSs simplifies the porting of codes to this new platform.

References

[1] Mont-Blanc project: European approach towards energy efficient high performance. *http://www.montblanc-project.eu*

[2] C.K. Birdsall and A.B. Langdon. *Plasma Physics via Computer Simulation*. Institution of Physics, Bristol, 1991.

[3] E. Akarsu et al. "Particle-in-cell Simulation Codes in High Performance Fortran", *Proceedings of the 1996 ACM/IEEE Conf. on Supercomputing (IEEE Computer Society)*, 38, 1996.

[4] V. Kornilov, R. Kleiber, R. Hatzky, et al., Gyrokinetic global three-dimensional simulations of linear ion-temperature-gradient modes in Wendelstein 7-X", *Physics of Plasmas*, 11, 2004.

[5] E. Sánchez, R. Kleiber, R. Hatzky, et al., Linear and non-linear simulations using the EUTERPE gyrokinetic code, IEEE Transaction on Plasma Science 38, 9, 2010.

[6] R. Hatzky, "Domain Cloning for a Particle-in-Cell (PIC) Code on a Cluster of Symmetric-Multiprocessor (SMP) Computers". *Parallel Computing, 32, 2006.*

[7] ARM: The architecture for the Digital World. *http://www.arm.com*

[8] X. Sáez, A. Soba, E. Sánchez et al. "Particle-In-Cell algorithms for Plasma simulations on heterogeneous architectures", *Proc. of the 19th Euromicro Conf. on Parallel, Distributed and Network-based Processing (IEEE)*, 385-389, 2011.

[9] A. Duran, E. Ayguadé, R.M. Badia, J. Labarta et al. "OmpSs: A proposal for programming heterogeneous multi-core architectures". *Parallel Processing Letters*, 21, 2, 173-193, 2011.