The Heterogeneous Exascale Particle-In-Cell (HEXAPIC)

Ezhilmathi Krishnasamy [1,2] Shan-Chang Lin [1] Ayushi Agrawal [1] Pascal Bouvry [1] Stefan Costea [2] Jernej Kovačič [2] Leon Kos [2]

[1]University of Luxembourg, Luxembourg
[2]University of Ljubljana, Slovenia

ezhilmathi.krishnasamy@uni.lu

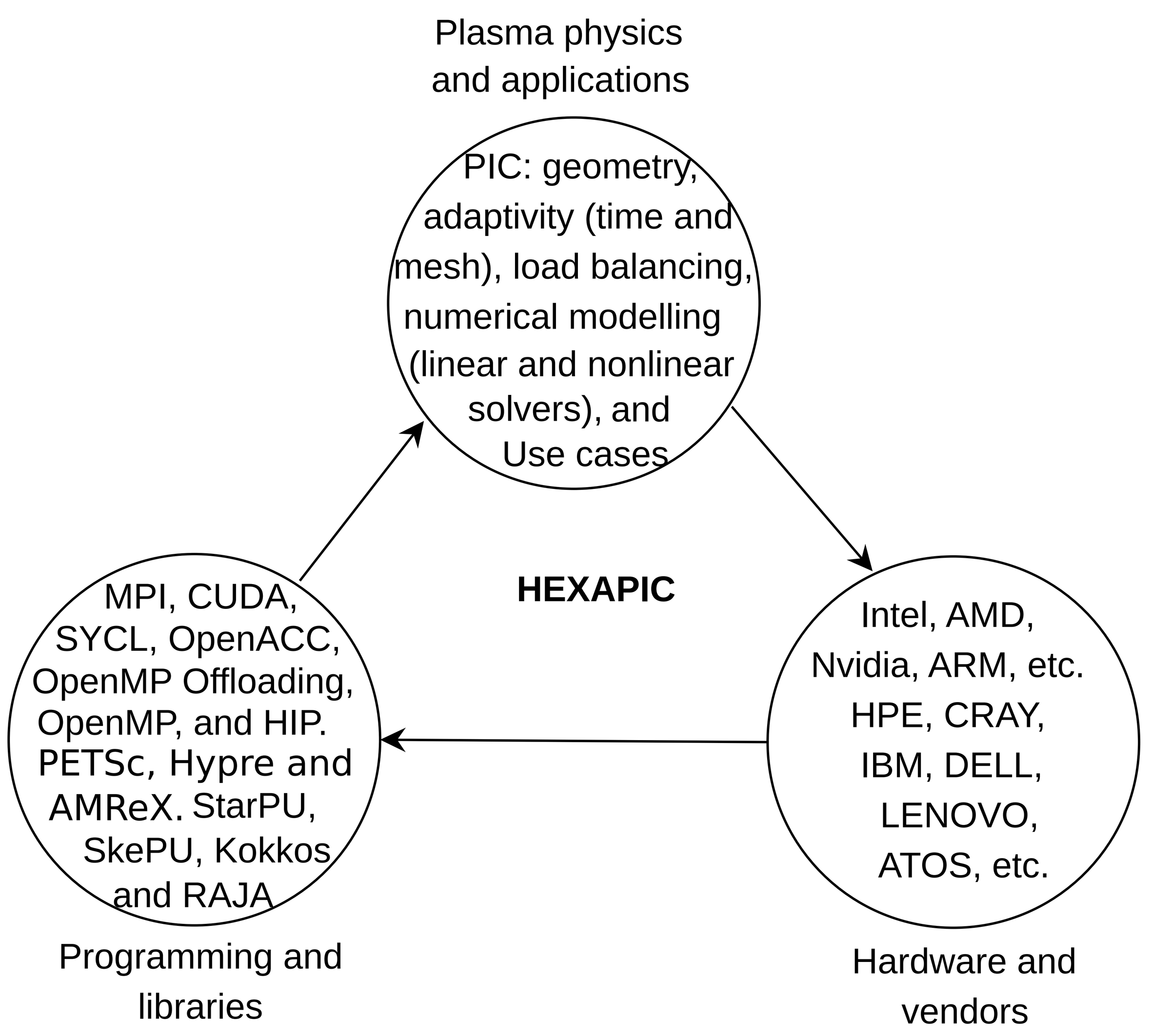
CONNECTING THE DOTS
ISC High Performance
JUNE 22–26, 2026 | HAMBURG, GERMANY

Abstract

Plasma physics relies heavily on particle-in-cell (PIC) codes, which this research aims to enhance through modern **heterogeneous** computer architectures, focusing on two key hypotheses. First, advanced algorithms for complex physical mechanisms can improve PIC codes by optimizing workload distribution, efficiently using accelerators, and enhancing communication and memory management. Second, fully-kinetic PIC codes are essential to accurately model plasma technological processes. Our aim is to optimize the PIC code with state-of-the-art techniques that integrate realistic geometries and comprehensive plasma dynamics, achieving unprecedented precision in plasma simulations.

Objectives

- Develop a novel Particle-In-Cell (PIC) code architecture that fully exploits the heterogeneous capabilities of modern HPC systems, ensuring scalability to **exascale**.
- Establish a formalized PIC code engineered for exascale performance, enabling realistic plasma simulations that accurately represent complex geometries and plasma processes.



Schematic overview of the workflow components in the HEXAPIC project.

Contributions

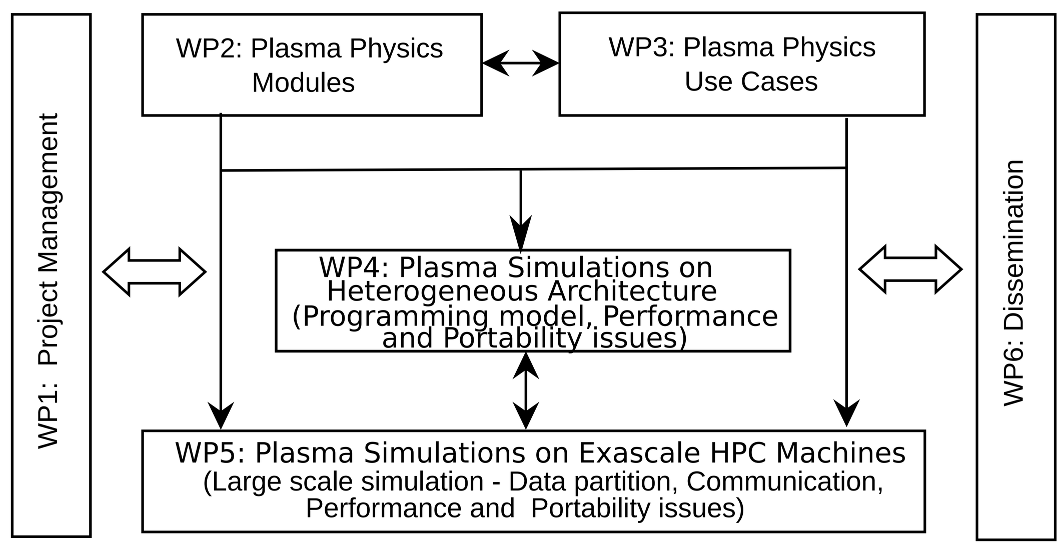
- A novel PIC code optimized for **exascale** performance, built on an adaptable heterogeneous architecture with advanced memory management and effective workload distribution.
- Practical use cases demonstrating realistic plasma simulations that account for intricate geometries and various collisional and transport processes, yielding new insights into physical phenomena.

Project Overview

Runtime: 2024 September - 2027 August
Funding source: Programme / Grant No. C23/IS/18105668/HEXAPIC

Participating partners:

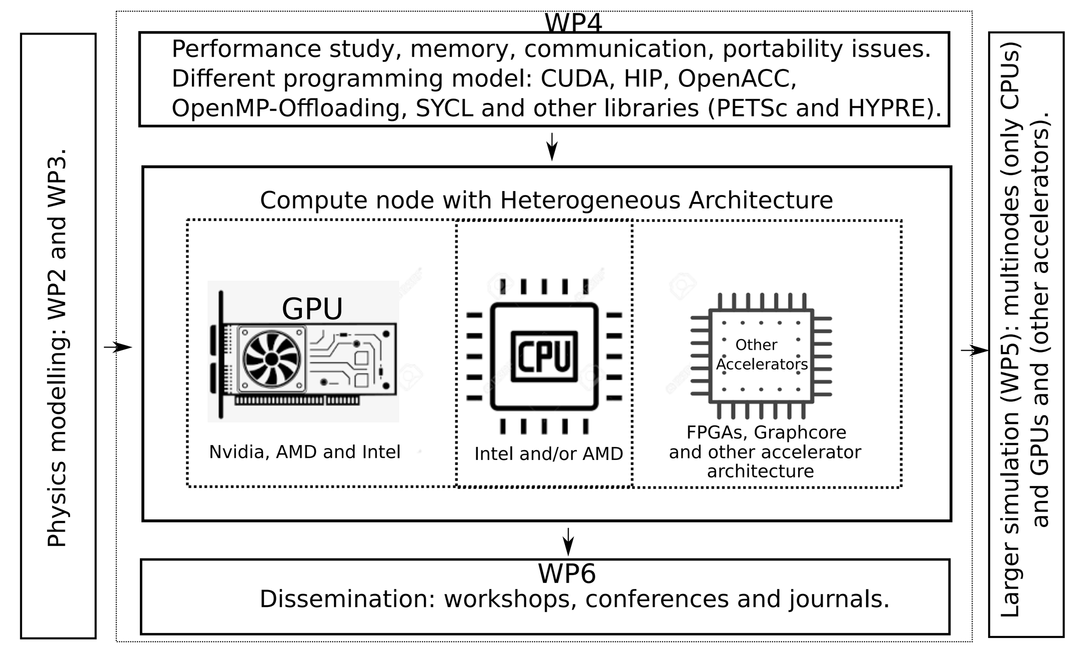
- University of Luxembourg (Lead)
- University of Ljubljana (Partner)
- University of Oslo (In-kind partner)
- Simula Research Laboratory (In-kind partner)
- Universidade de Santiago de Compostela (In-kind partner)
- University of Groningen (In-kind partner)
- FNR / ARIS (funding agencies)



Complete work plan of the HEXAPIC project, outlining key phases, tasks, and milestones.

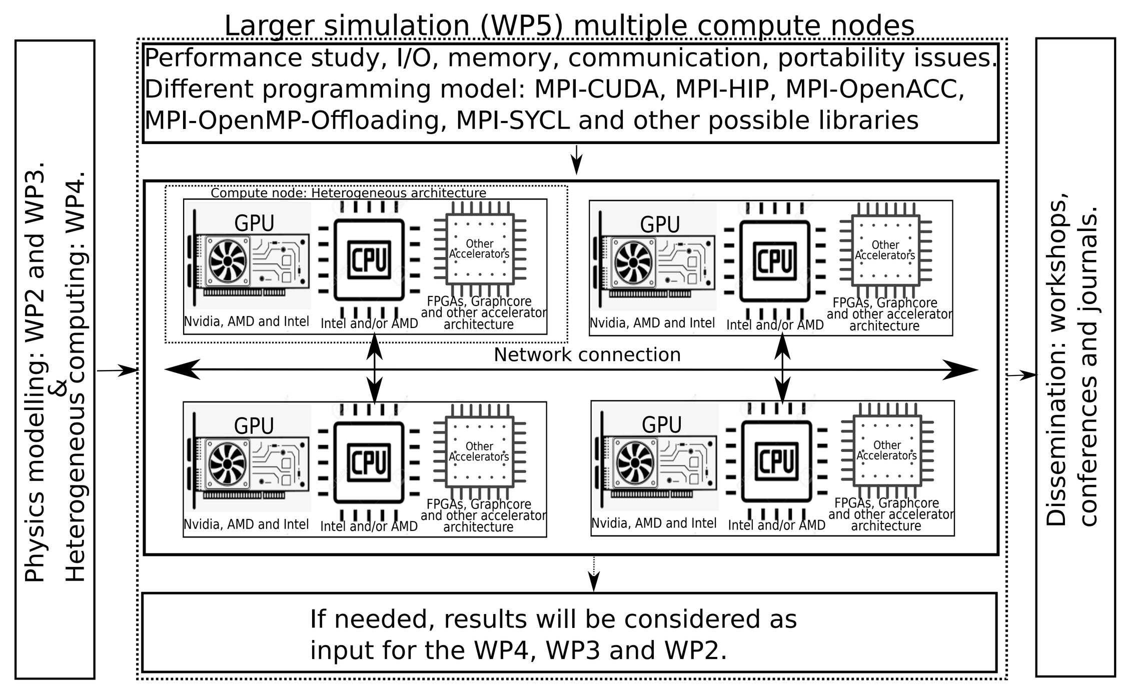
Approach & Methodology

Stable sequential version: A stable version of the software captures all physical phenomena along with material properties, boundary conditions, and initial conditions, while introducing an MPI version with support for scientific libraries such as **PETSc** and **HYPRE**.
Intra-node host–device distribution: Within each compute node, work is partitioned between the **host (CPU)** and **device (GPU)** to balance computation against data movement, which is central to exploiting heterogeneous nodes efficiently.



Scientific workflow outlining the proposed host–device optimization strategy for a single compute node.

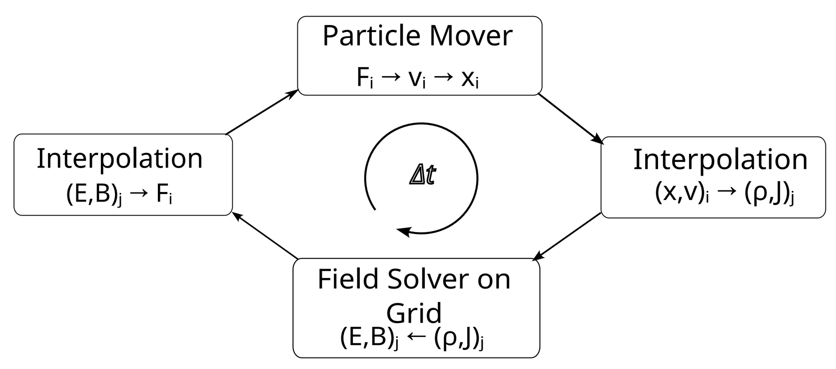
Multi-node execution: In a multi-node setup, it is important to evaluate the efficiency of multigrid solvers with HYPRE for GPU applications, where applicable. Additionally, implementing custom solvers should be considered using suitable programming models or libraries that target large heterogeneous HPC systems.



Scientific workflow outlining the proposed optimization strategy for multiple compute nodes in HEXAPIC.

Current Status

The 2D MPI code of the PIC model has been released with **HYPRE** support for single-node and multi-node **CPU and GPU** environments, available at **github.com/LeCAD-PEG/hexapic**. Various parallel programming models have also been studied for GPUs across single- and multi-node setups.



Explicit PIC algorithm: i and j indicate particles and grids; v: velocity, x: position, F: force, ρ: charge density, J: current density, E: electric field, B: magnetic flux, Δt: time step.

Preliminary Results (excerpt)

Single-node GPU:

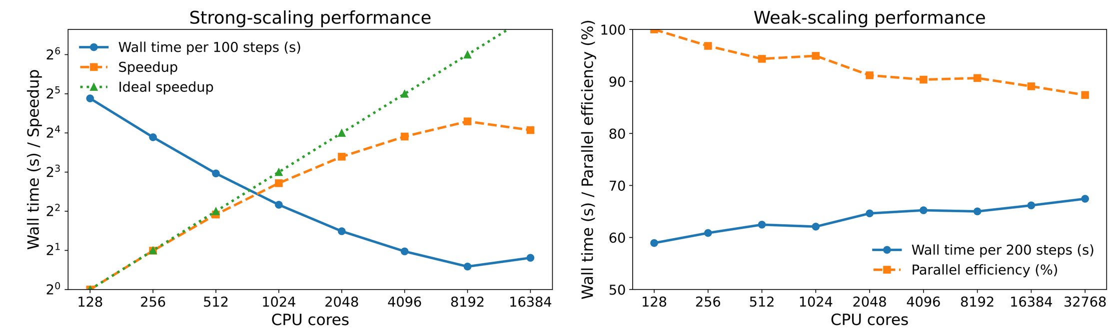
- Single-GPU optimization studies across various programming models have been completed [1, 2, 3, 4].

Multi-node GPU:

- Programming models were studied with **MPI+X**, where X refers to CUDA, HIP, SYCL, OpenACC, and OpenMP offloading for multiple GPUs [5].

HEXAPIC code on CPU and GPU:

- We tested a 2D PIC code on multi-node CPUs using **PETSc** and **HYPRE** (**PFMG**) solvers.



Strong and weak scaling of HEXAPIC on the VEGA HPC system using multi-node CPUs (baseline; GPU acceleration is the route to larger scale).

- **HYPRE** support has been introduced to accelerate computations on **NVIDIA** and **AMD** GPUs.
- Early experiments demonstrate improved **scalability** and **efficiency** over baseline approaches, confirming the design's **performance portability** across CPU and GPU architectures.

Roadmap / Next Steps

HPC perspective: Enhance performance by incorporating HYPRE/PETSc GPU libraries and developing custom solvers using OpenACC and OpenMP offloading, as well as low-level models such as CUDA, HIP, and SYCL. Well-tested libraries such as StarPU and Alpaka will be used to optimize performance on heterogeneous HPC systems, with a performance analysis comparing custom solvers against GPU-library-supported implementations.

Mathematical modeling: Propose a computationally stable alternative to the existing solver to address performance discrepancies. Introducing machine learning methods to solve all or part of the PIC algorithm could further optimize the computational process, improving both efficiency and accuracy within the PIC framework.

Use cases / Physics modeling: Test diverse use cases on realistic datasets, and introduce and validate additional complex plasma simulations with intricate geometries and physics parameters within the HEXAPIC code to further improve the accuracy of capturing physical behavior.

References

1. E. Krishnasamy and P. Bouvry, "Reproducible performance study of OpenACC vs. CUDA," 2024.
2. E. Krishnasamy and P. Bouvry, "OpenACC vs. CUDA for linear algebra computations," EECT, 2025.
3. E. Krisshnasamy and P. Bouvry, "CUDA vs. OpenMP offloading for BLAS operations on GPUs," Int. Conf. on ICT for Intelligent Systems, 2025.
4. E. Krisshnasamy and P. Bouvry, "OpenMP offloading on AMD and NVIDIA GPUs," Int. Conf. on High Performance Compilation, Computing and Communications, 2025.
5. E. Krisshnasamy, J. Trotter, X. Cai, D. Pleiter, L. Kos, L. Saavedra, and P. Bouvry, "MPI+X offloading performance for dense matrix–vector multiplication," HPCAsia Workshops (HPCAsiaWS), 2026.

This work was supported by the Fonds National de la Recherche (FNR), Luxembourg, under the **FNR CORE** project grant number **C23/IS/18105668/HEXAPIC**, and the Slovenian Research and Innovation Agency **(ARIS)**, Slovenia.

Luxembourg National Research Fund
aris Slovenian Research and Innovation Agency
UNIVERZA V LJUBLJANI
project website