Imperial College London

EPSRC Engineering and Physical Sciences Research Council

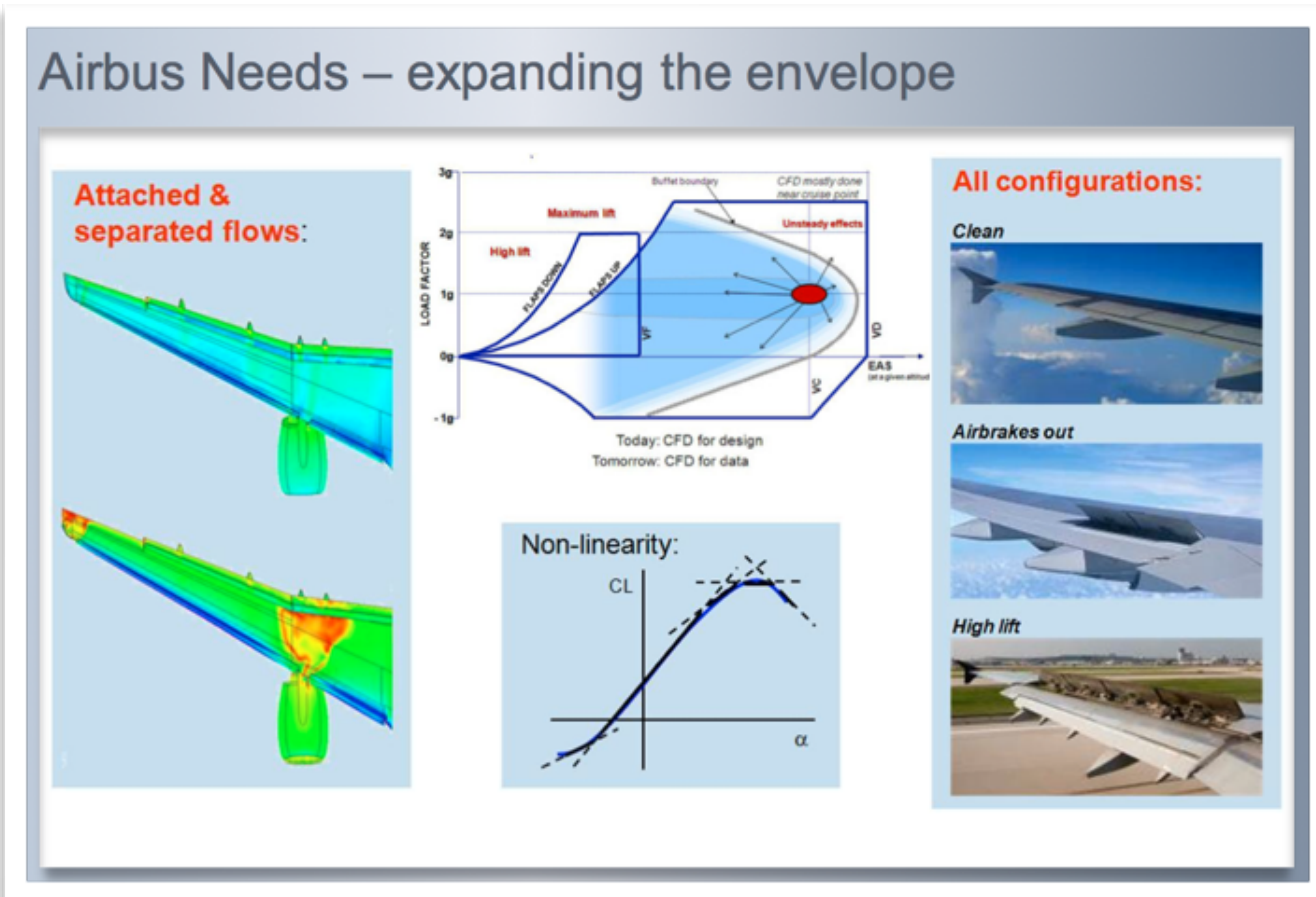
PyFR: Heterogeneous Computing on Mixed Unstructured Grids with Python

F.D. Witherden, M. Klemm, P.E. Vincent

Overview

- Motivation.
- Accelerators and Modern Hardware
- Python and PyFR.
- Summary.

Motivation

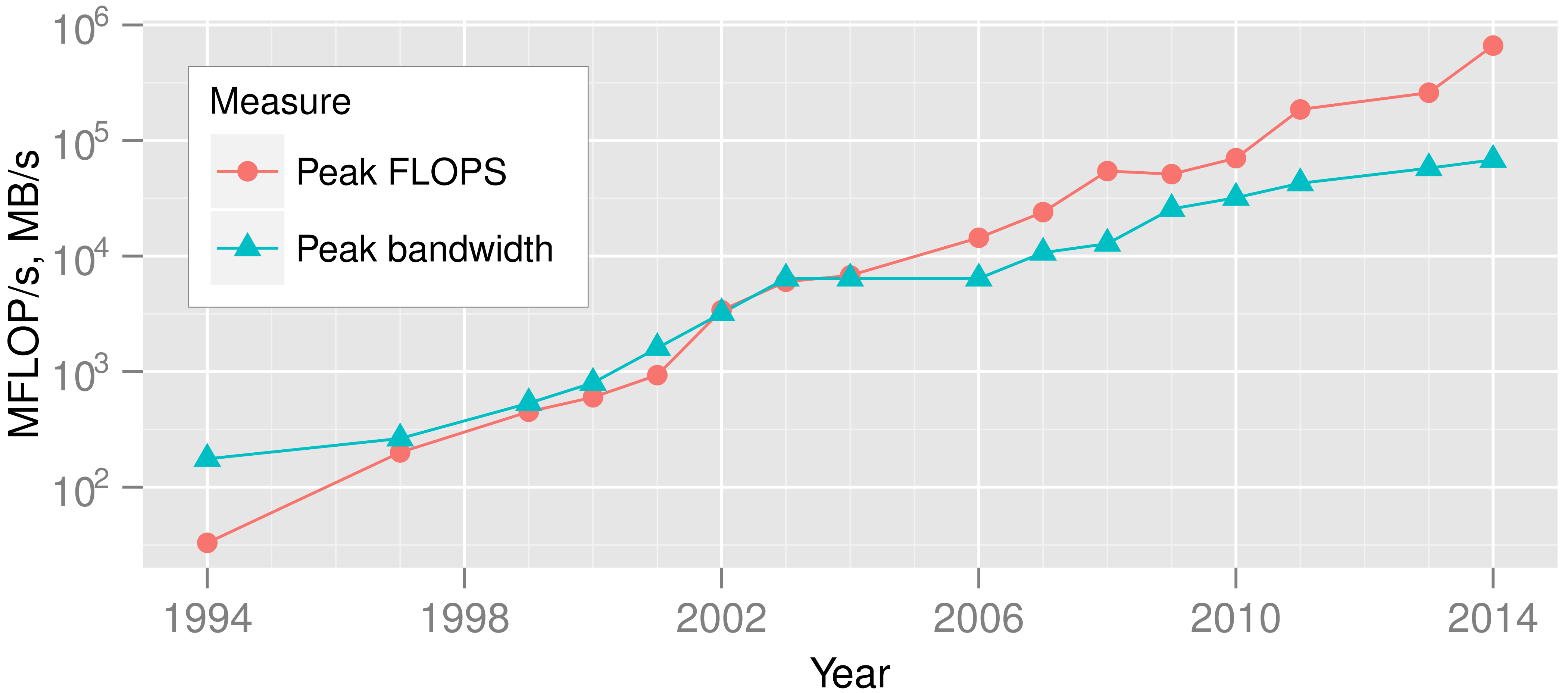


[1] Murray Cross, Airbus, Technology Product Leader - Future Simulations (2012)

Motivation

- Objective is to advance **industrial CFD** capabilities from their current **RANS plateau**.
- Achieved by intelligently leveraging benefits of high-order **Flux Reconstruction** (FR) methods for unstructured grids and massively-parallel **modern hardware** platforms.

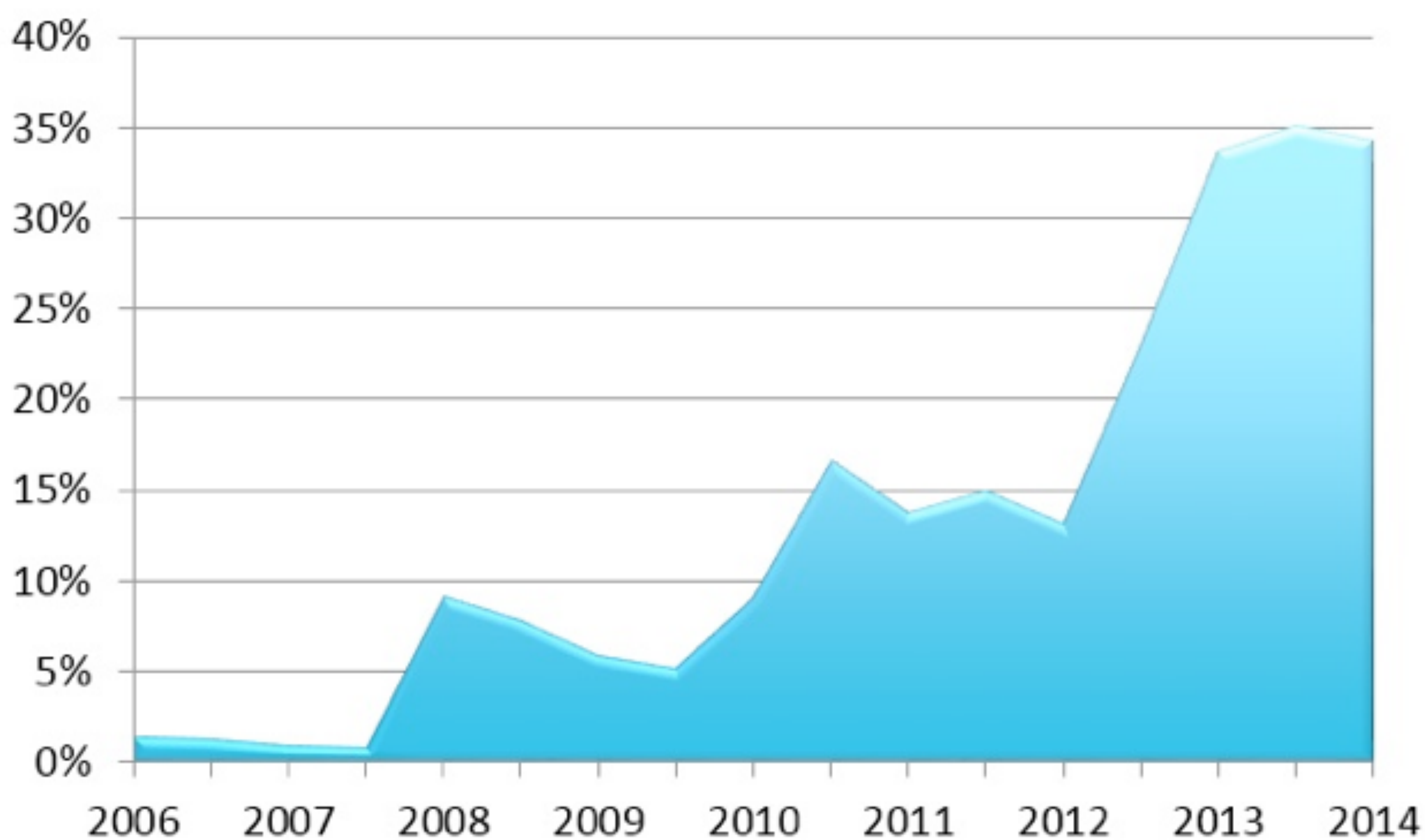
Modern Hardware



- Intel Xeon CPUs from 1994-2014.

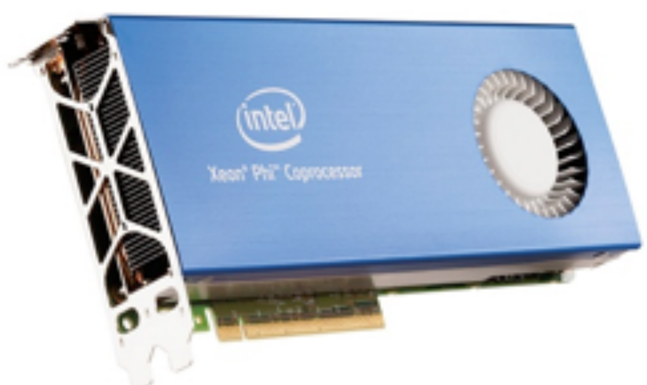
Accelerator Adoption

- FLOPS contributed by accelerators to the **TOP500**. [HPCwire]



Accelerator Adoption

- Within the **top ten**:



Intel Xeon Phi
2 of 10

NVIDIA Tesla
2 of 10

Increasing Heterogeneity

- Consider **Stampede** at **TACC**.
- Currently **#7** on the TOP500 list.

Intel Xeon CPUs

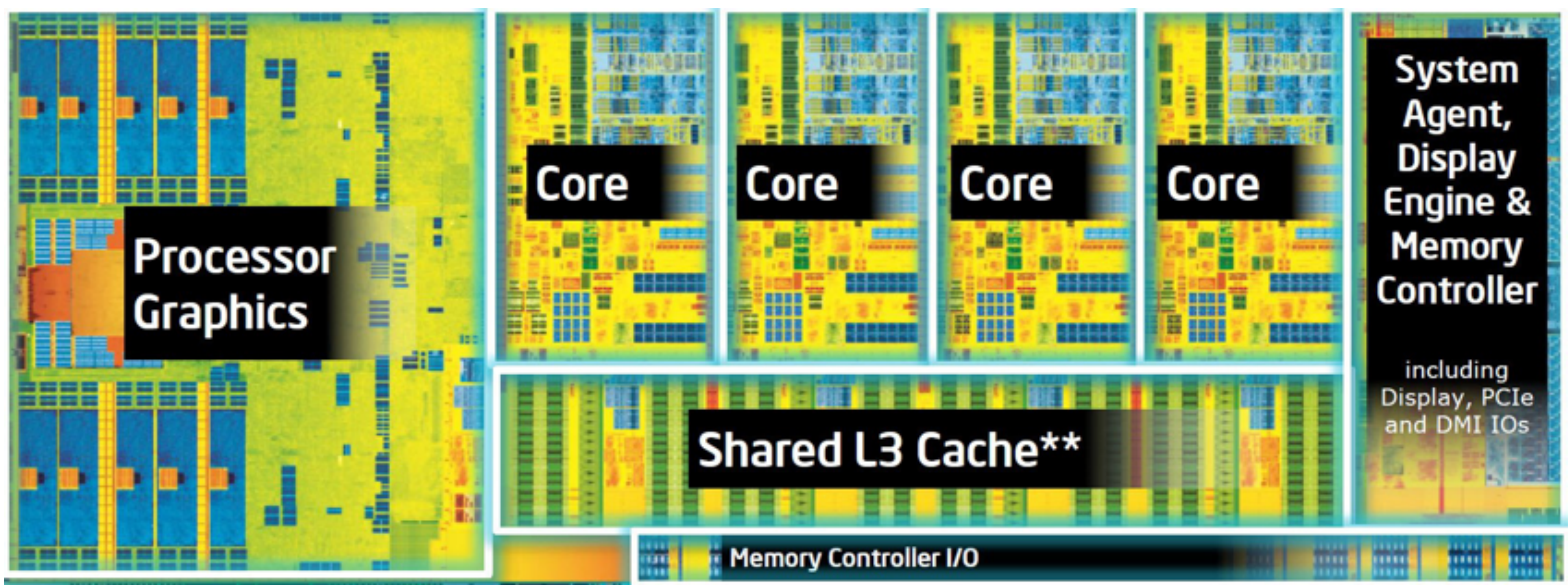
2.2 PFLOP/S

Intel Xeon Phis

7.4 PFLOP/S

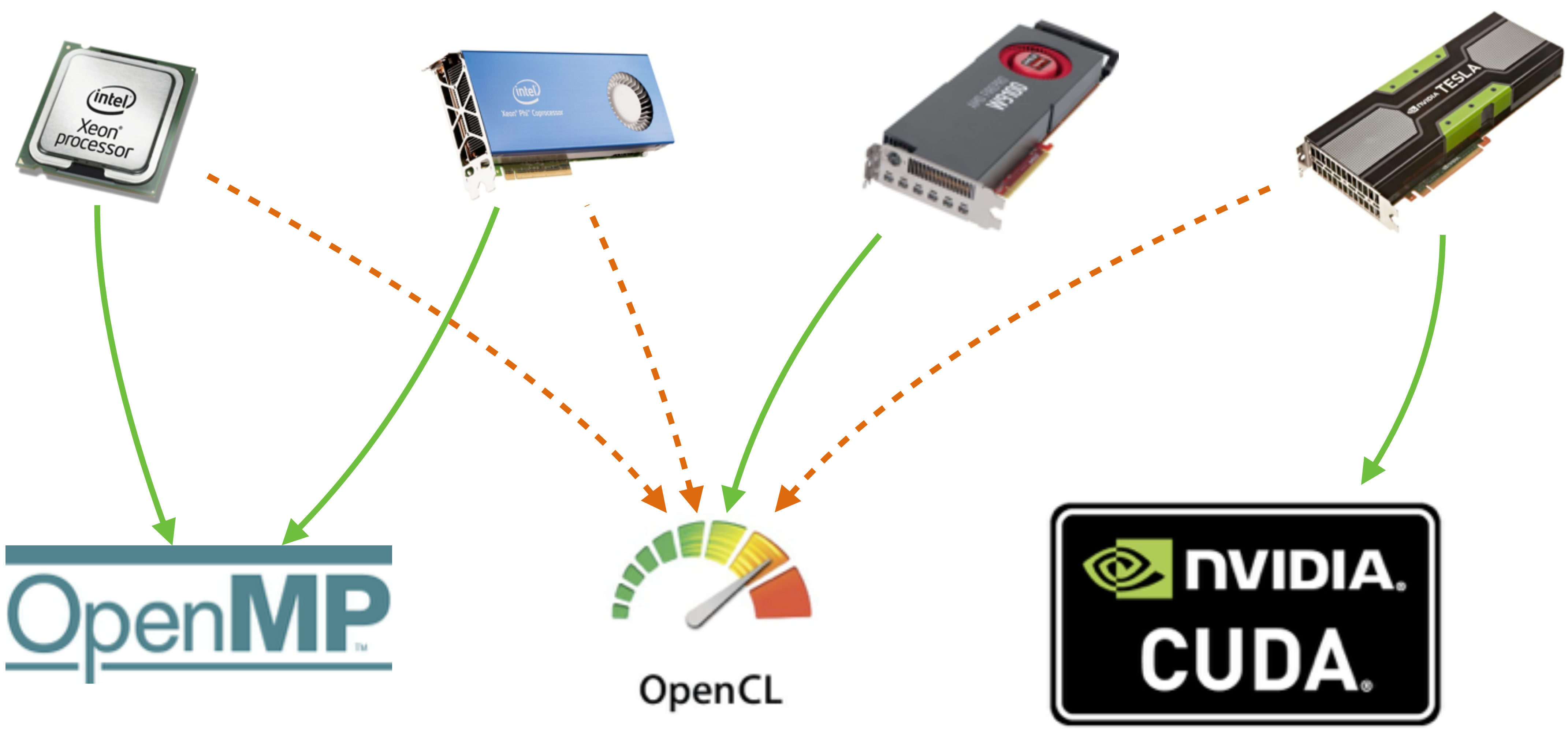
Increasing Heterogeneity

- …and it is not just the high end.



Performance Portability

- It is a challenging environment…



PyFR

- Our solution **PyFR**.
- Written in **Python**.
- Uses flux reconstruction to solve the **Navier-Stokes** equations on mixed unstructured grids in **2D/3D**.
- **Performance portable** across a variety of hardware platforms.

PyFR

- Python outer layer.

Python Outer Layer
(**Hardware Independent**)

- Setup
- Distributed memory parallelism
- Outer ‘for’ loop and calls to **Hardware Specific Kernels**

PyFR

- Need to generate **hardware specific kernels**.

Python Outer Layer
(**Hardware Independent**)

- Setup
- Distributed memory parallelism
- Outer 'for' loop and calls to **Hardware Specific Kernels**

PyFR

- In FR **two types** of kernel are required.

Python Outer Layer
(**Hardware Independent**)

- Setup
- Distributed memory parallelism
- Outer ‘for’ loop and calls to **Hardware Specific Kernels**

Matrix Multiply Kernels

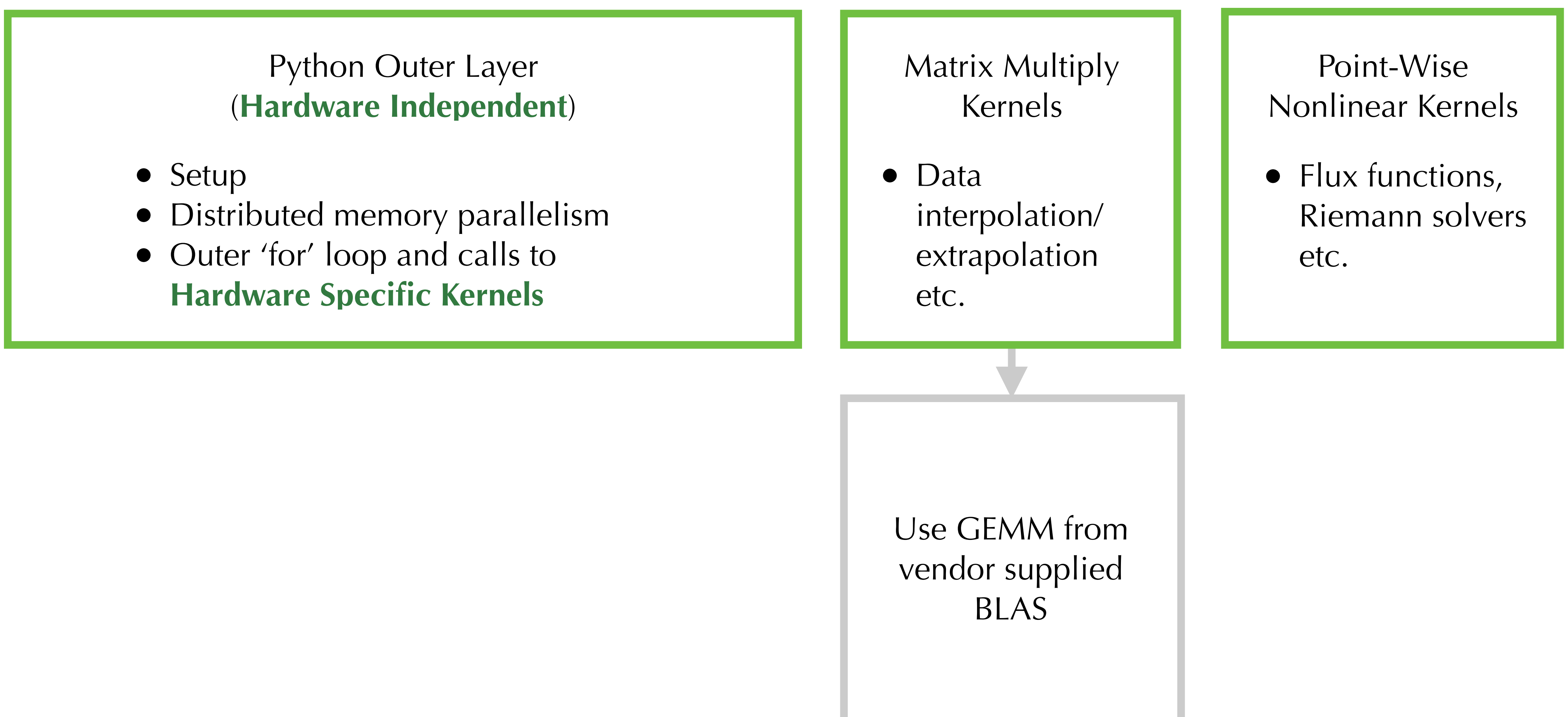
- Data interpolation/ extrapolation etc.

Point-Wise Nonlinear Kernels

- Flux functions, Riemann solvers etc.

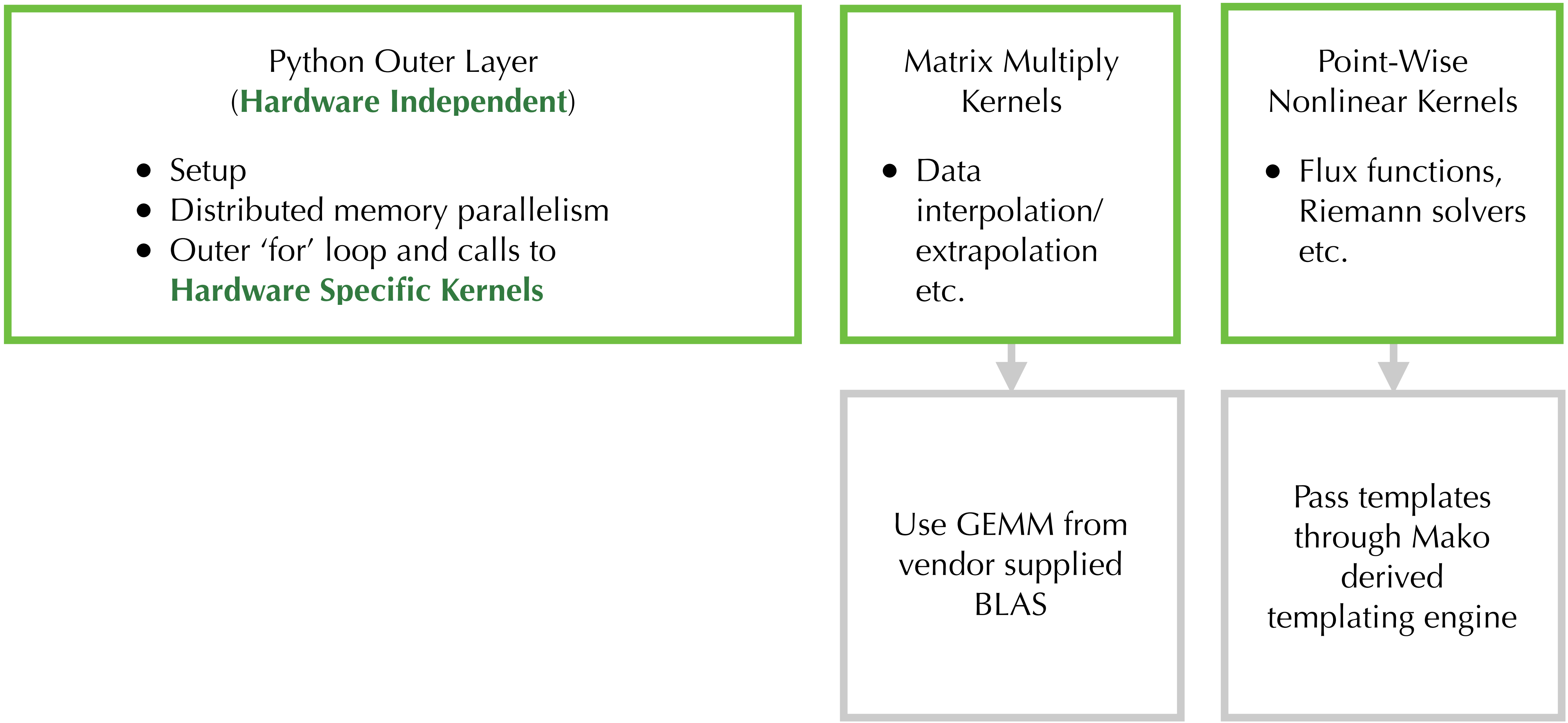
PyFR

- Matrix multiplications are quite simple.



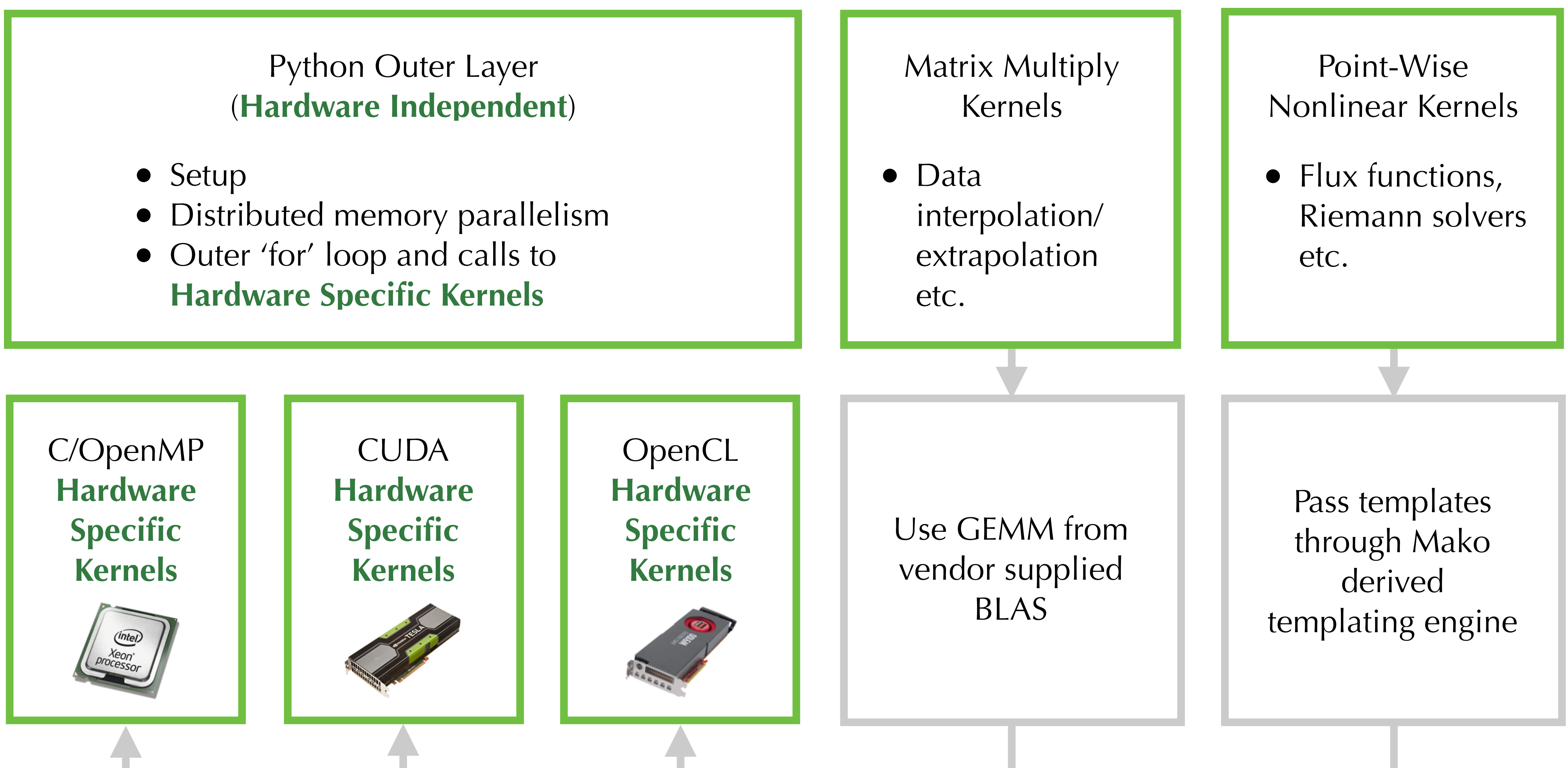
PyFR

- Harder for point-wise nonlinear kernels.



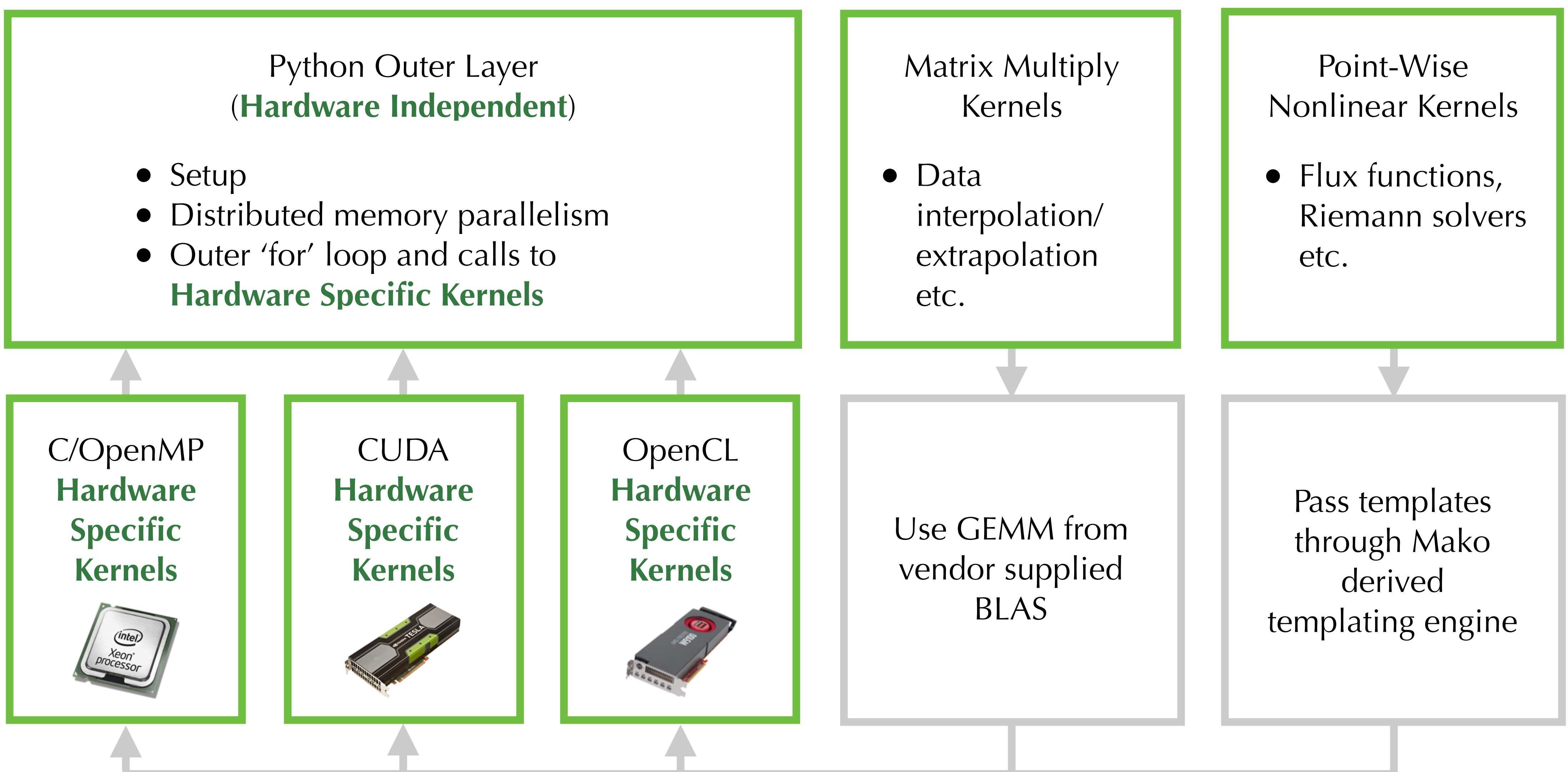
PyFR

- These can now be called.



PyFR

- These can now be called.



Mako Template

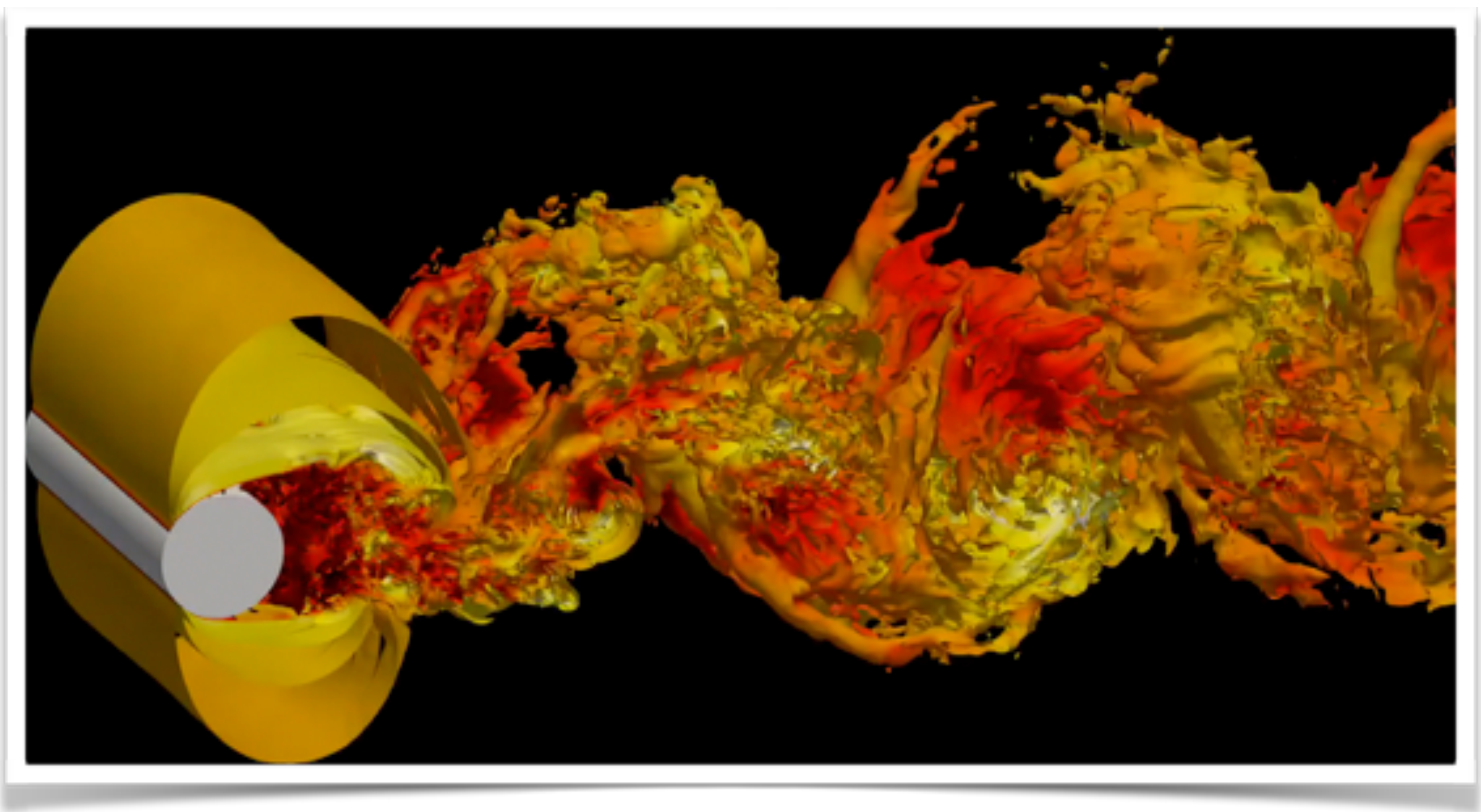
```
<%pyfr:kernel    name='negdivconf' ndim='2'
                 t='scalar fpdtype_t'
                 tdivtconf='inout fpdtype_t[${str(nvars)}]'
                 ploc='in fpdtype_t[${str(ndims)}]'
                 rcpdjac='in fpdtype_t'>
% for i, ex in enumerate(srcex):
    tdivtconf[${i}] = -rcpdjac*tdivtconf[${i}] + ${ex};
% endfor
</%pyfr:kernel>
```

PyFR Core Tenants

- Exploit the rich Python ecosystem:
    - numpy, mpi4py, PyCUDA, PyOpenCL, pyMIC, h5py, mako
- Offload all computation:
    - overhead from the interpreter **< 1%**.

PyFR Results I

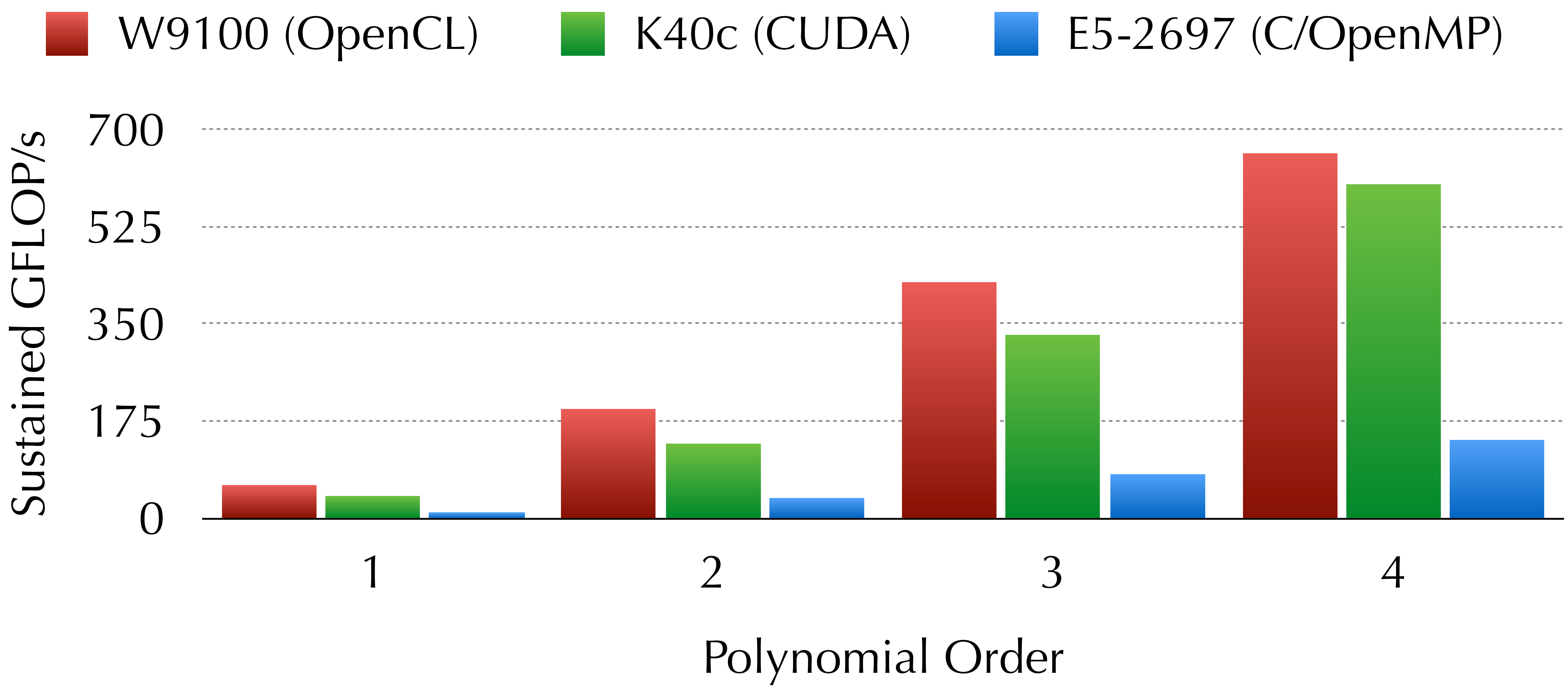
- Benchmark problem: flow over a **cylinder**:



- Isosurfaces of density **Ma = 0.2**, **Re = 3900**.

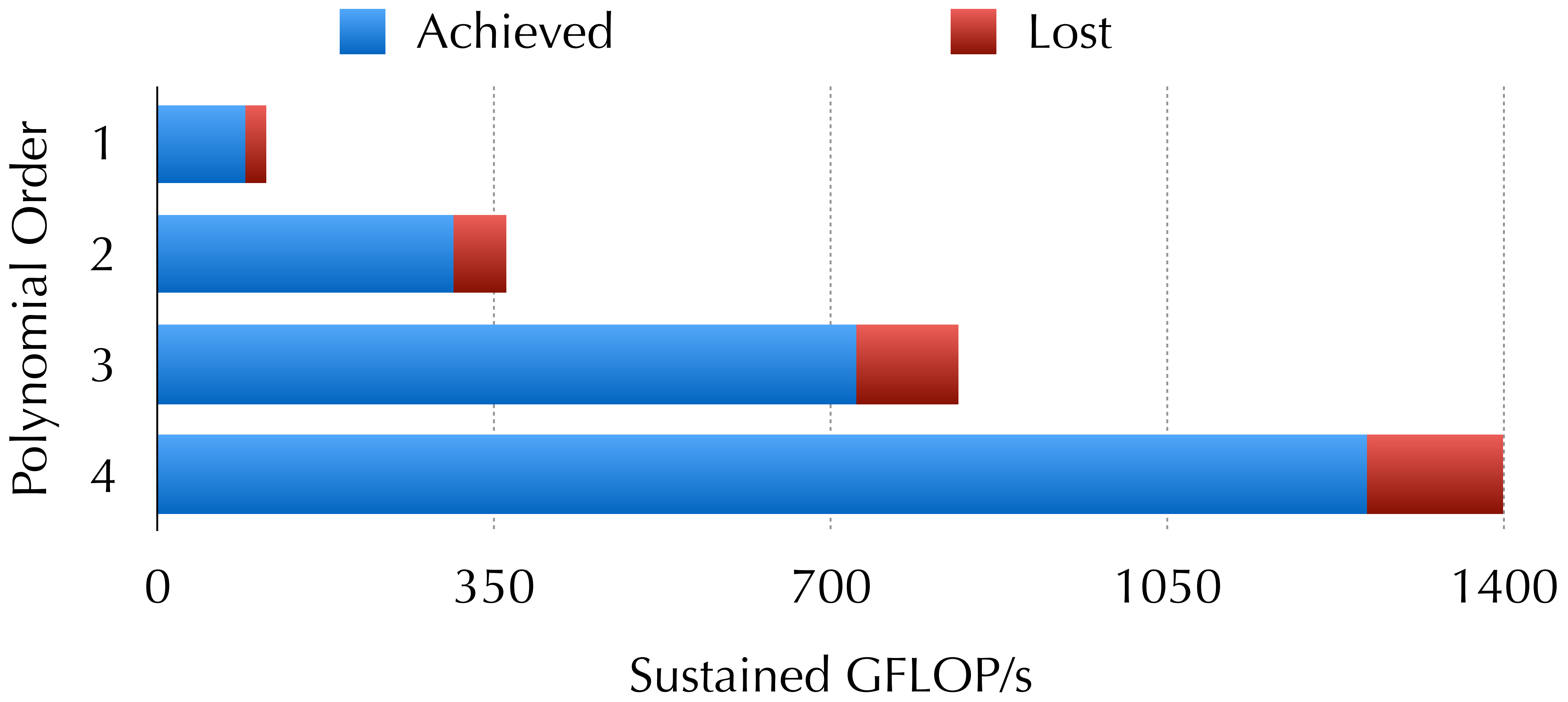
PyFR Results II

- **Single node** performance.



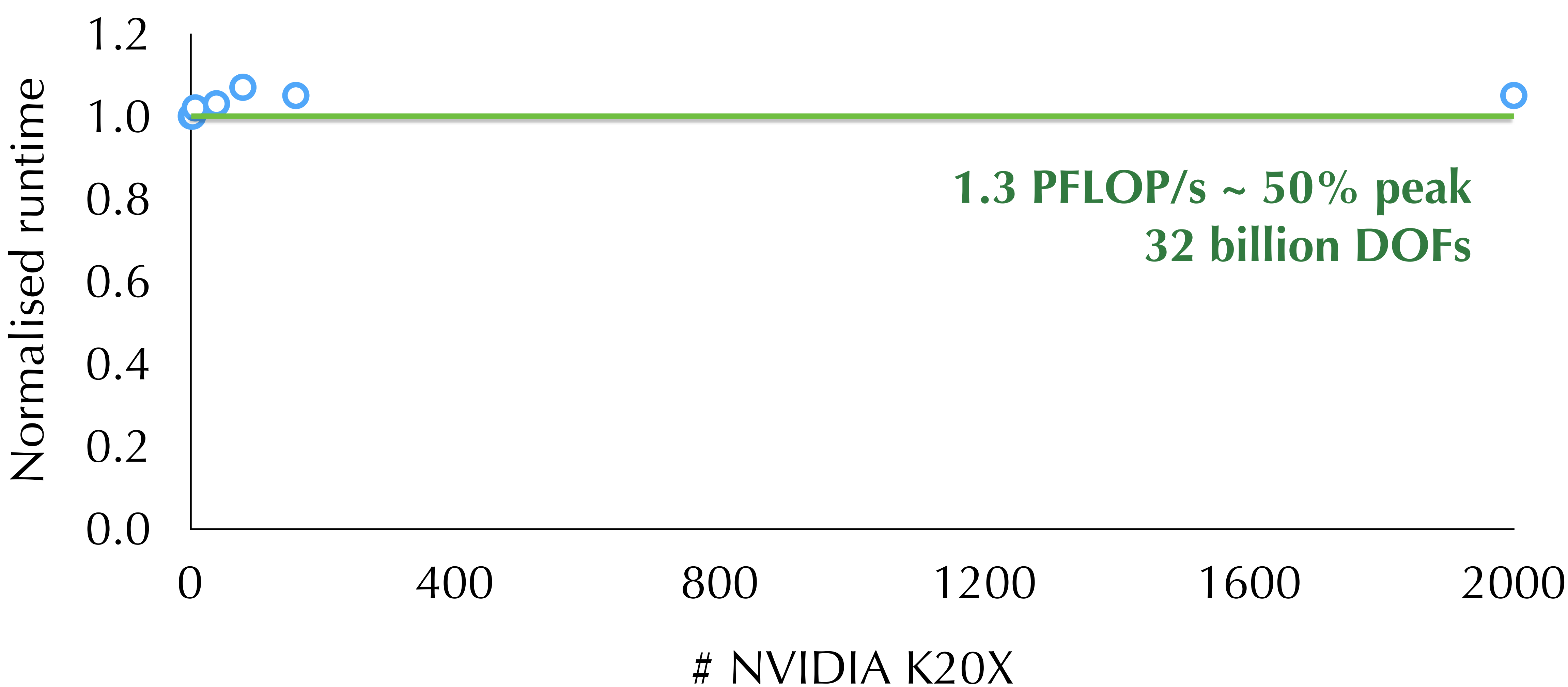
PyFR Results III

- **Multi-node heterogeneous** performance on the same mesh.



PyFR Results IV

- **Weak scaling** on Piz Daint at CSCS for a NACA 0021.



Summary

- Funded and supported by
- Any questions?
- E-mail: **freddie.witherden08@imperial.ac.uk**