Efficient Scientific Computing School – 13th Edition

Exploiting Heterogeneous Architectures: Applications and Lessons Learned

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

www.casus.science

```
mirror object to mirror
irror_mod.mirror_object
peration == "MIRROR_X":
irror_mod.use_x = True
irror_mod.use_y = False
irror_mod.use_z = False
 operation == "MIRROR_Y"
irror_mod.use_x = False
irror_mod.use_y = True
irror_mod.use_z = False
 operation == "MIRROR_Z"
irror_mod.use_x
```

HZDR HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

TECHNISCHE UNIVERSITÄT DRESDEN

CBG Max Planck Institute of Molecular Cell Biology and Genetics

UFZ HELMHOLTZ Centre for Environmental Research

Uniwersytet Wrocławski

SPONSORED BY THE

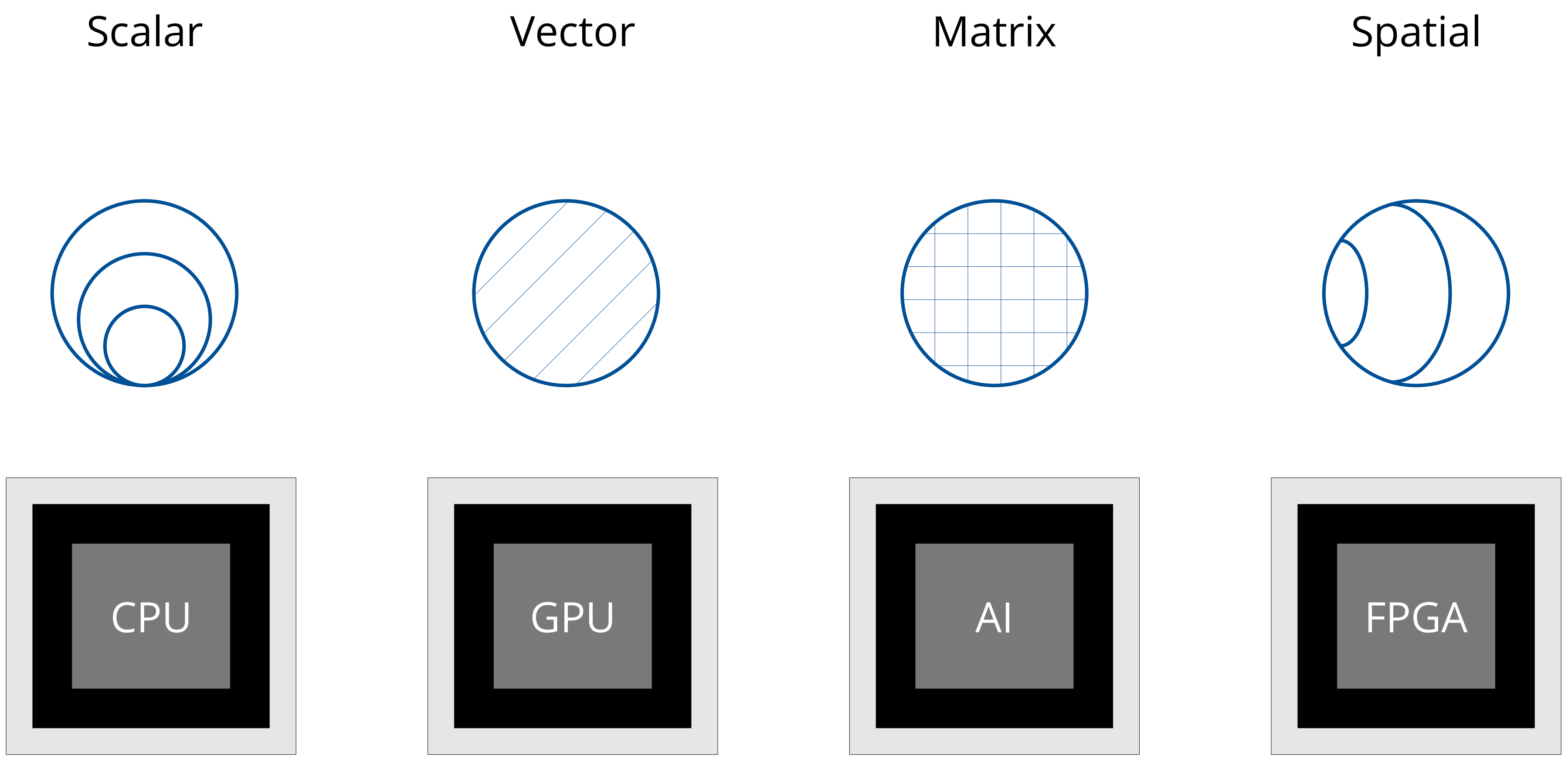
Federal Ministry of Education and Research

STAATSMINISTERIUM FÜR WISSENSCHAFT KULTUR UND TOURISMUS

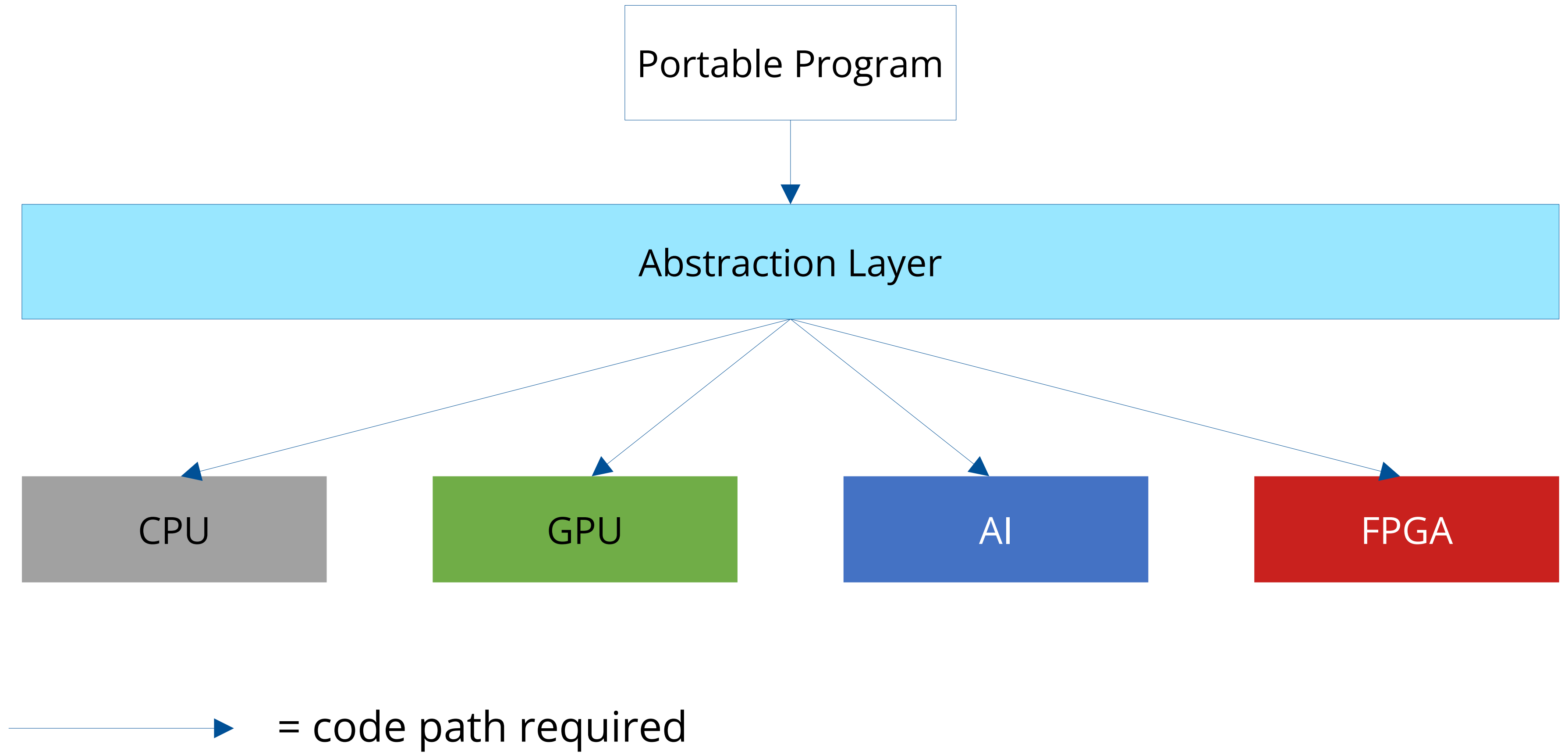
Freistaat SACHSEN

Recap

Workload Patterns



General Idea



Available Libraries

kokkos Developed by Sandia National Laboratories (USA)

RAJA Developed by Lawrence Livermore National Laboratory (USA)

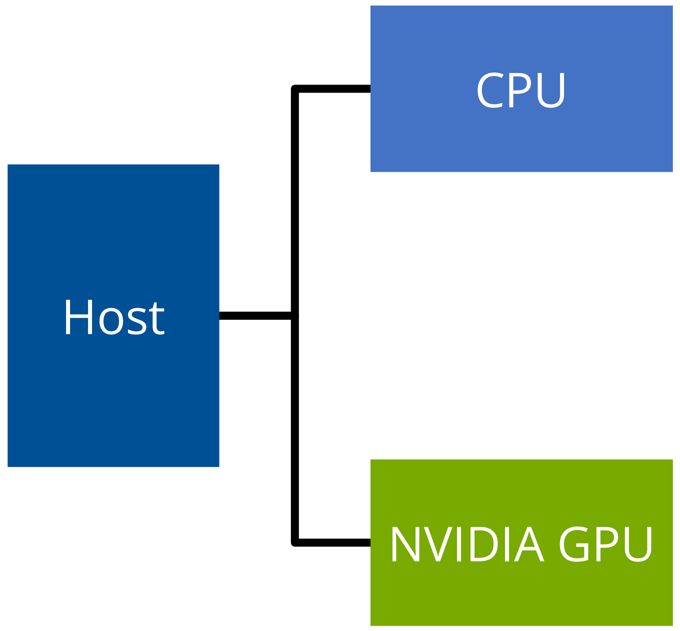
SYCL™ Designed by the Khronos industry consortium (USA)
Implemented by hardware vendors

alpaka Developed by Helmholtz-Zentrum Dresden-Rossendorf (Germany)

Programming Heterogeneous Systems

Heterogeneous Systems

- Real-world scenario: Use all available compute power
- Also real-world scenario: Multiple different hardware types available
- Requirement: Usage of one back-end per hardware platform
- Requirement: Back-ends need to be interoperable



Mastering Heterogeneous Systems

Why Do We Need This?

Can’t we just decide on one accelerator?

- In one word: Yes.

Why Do We Need This?

Can’t we just decide on one accelerator?

- In one word: Yes.
- In two words: It depends.

Why Do We Need This?

Can’t we just decide on one accelerator?

- In one word: Yes.
- In two words: It depends.
- Questions to ask yourself:

Why Do We Need This?

Can’t we just decide on one accelerator?

- In one word: Yes.
- In two words: It depends.
- Questions to ask yourself:
  - What is the expected lifetime of the application?

Why Do We Need This?

Can’t we just decide on one accelerator?

- In one word: Yes.
- In two words: It depends.
- Questions to ask yourself:
  - What is the expected lifetime of the application?
  - Is the accelerator suitable for all required algorithms?

Why Do We Need This?

Can’t we just decide on one accelerator?

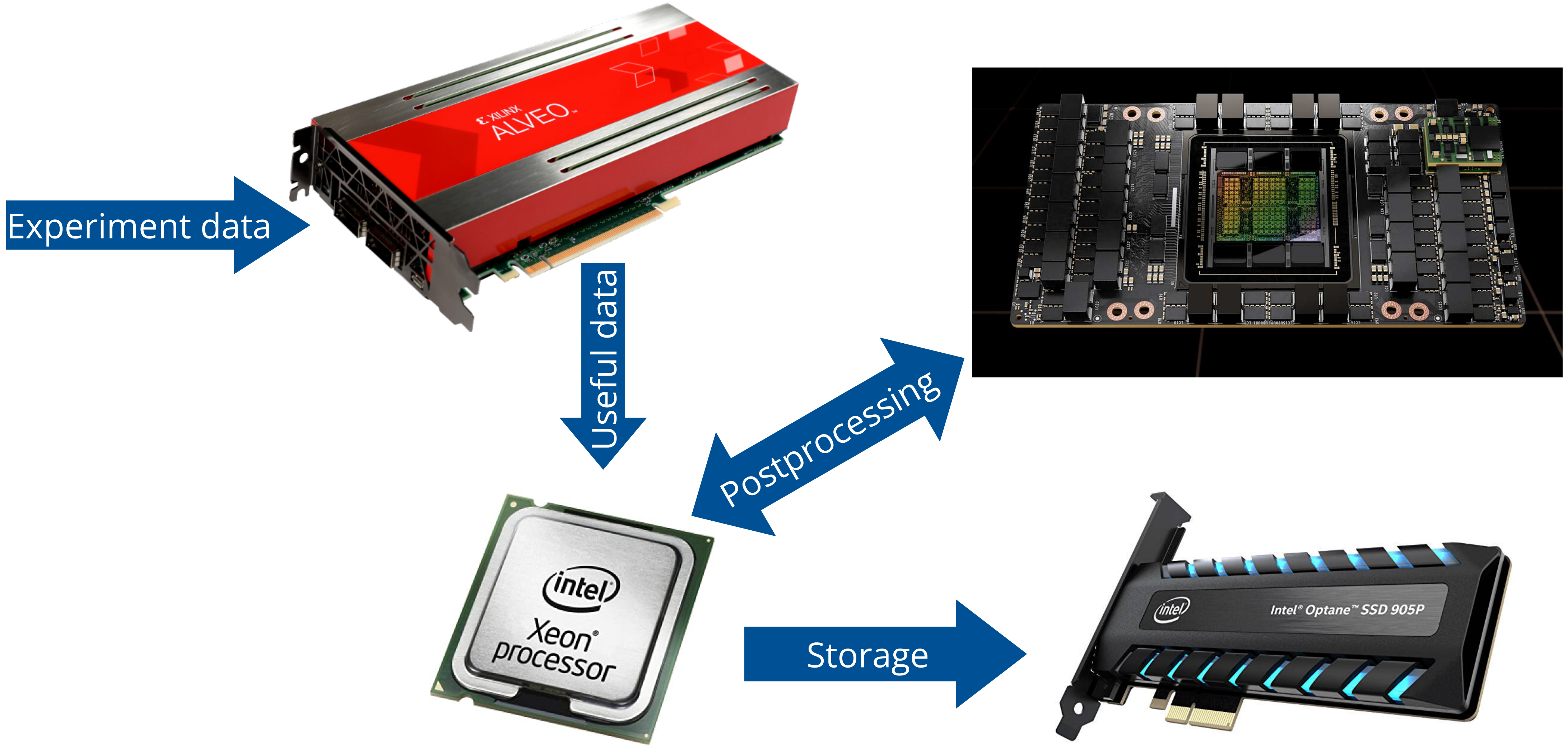
- In one word: Yes.
- In two words: It depends.
- Questions to ask yourself:
  - What is the expected lifetime of the application?
  - Is the accelerator suitable for all required algorithms?
  - If not, what is the penalty for using a non-ideal accelerator?

Why Do We Need This?

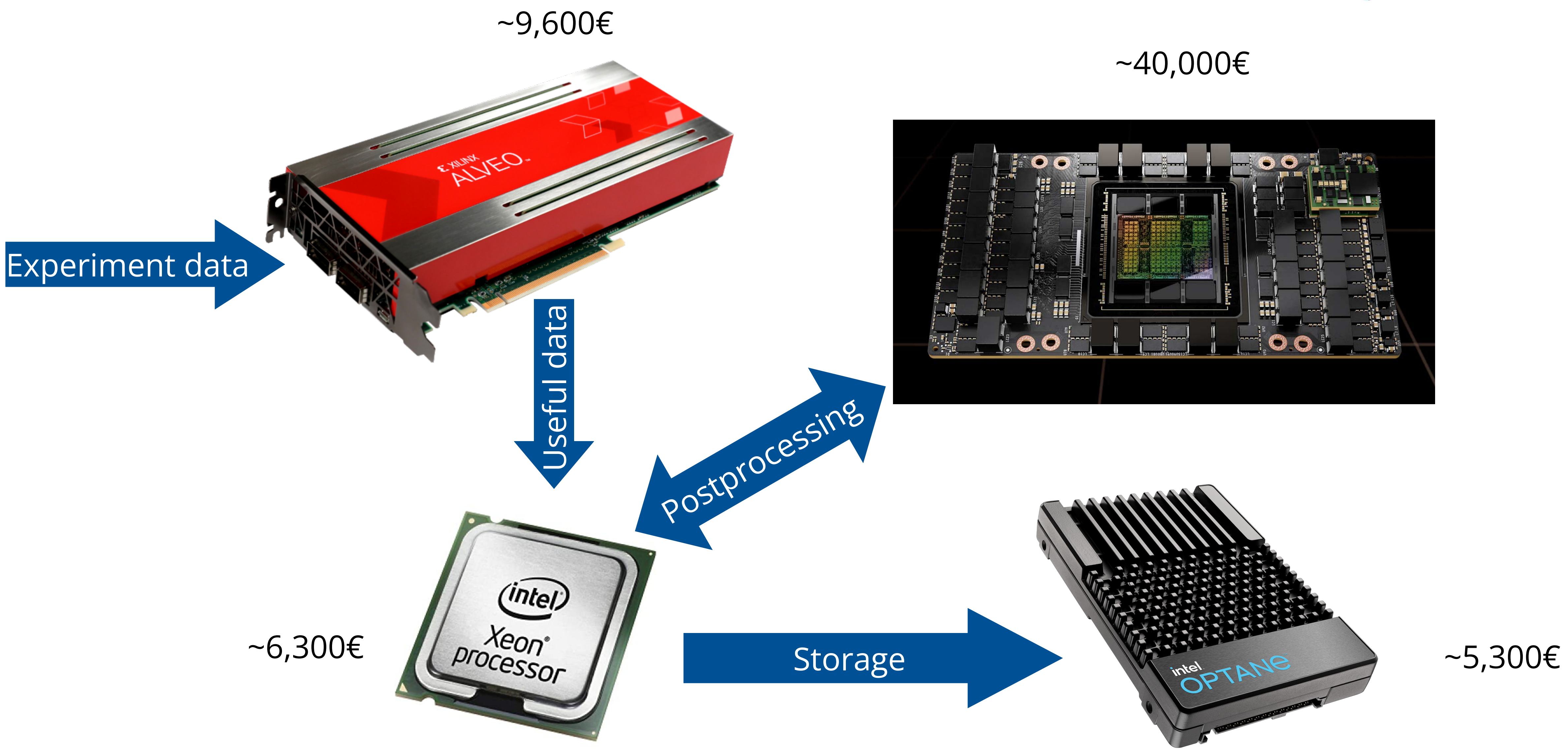
Can’t we just decide on one accelerator?

- In one word: Yes.
- In two words: It depends.
- Questions to ask yourself:
  - What is the expected lifetime of the application?
  - Is the accelerator suitable for all required algorithms?
  - If not, what is the penalty for using a non-ideal accelerator?
  - Do you have enough money to buy another accelerator?

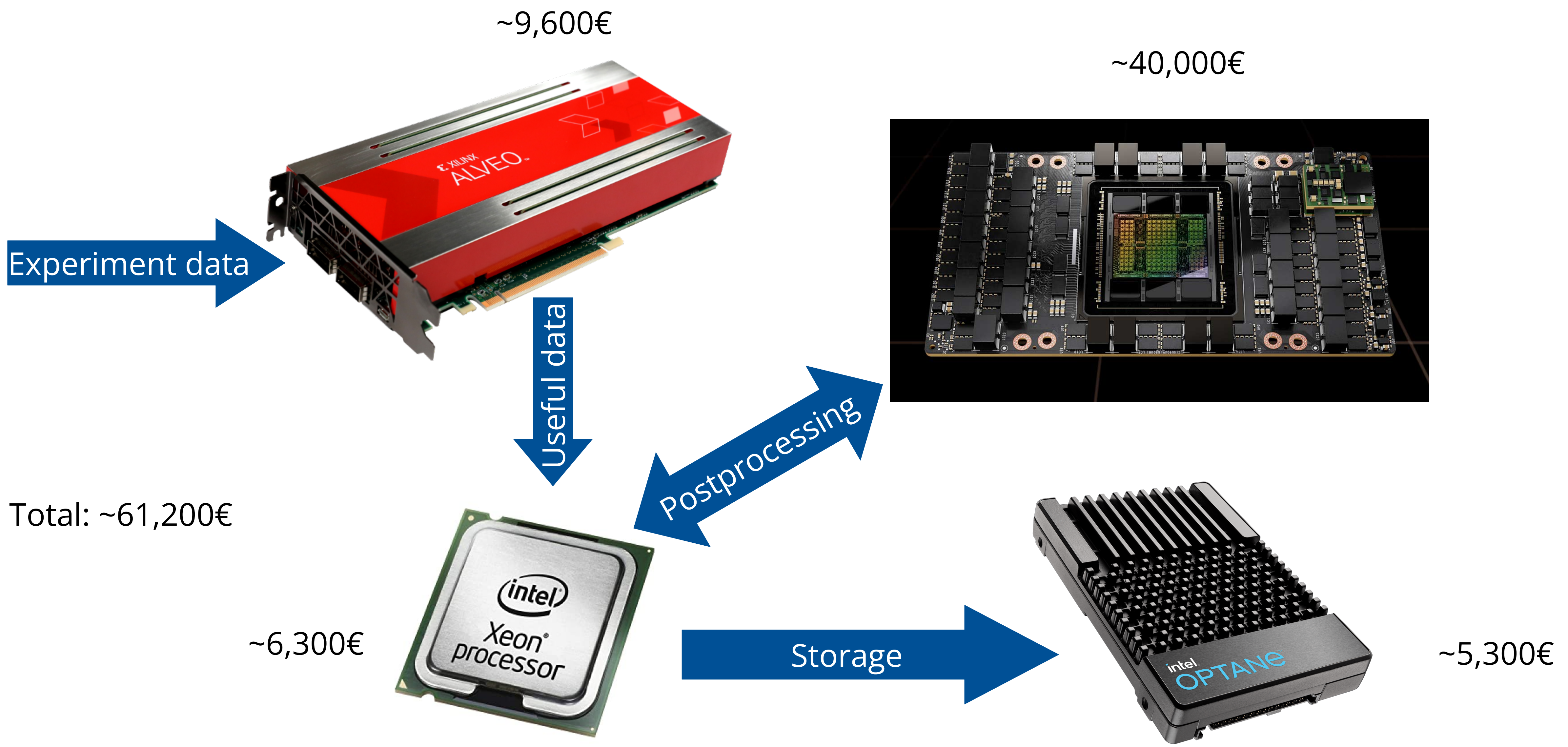
Example Setup



Example Setup



Example Setup



How Do We Control the Setup?

Bad solution

We can program each component individually using its native API!

- Requires detailed API knowledge for each platform
- Vendor lock-in for each component

How Do We Control the Setup?

Good solution

We use an abstraction layer for the entire application.

- Single API available for all underlying platforms
- Individual components can be easily exchanged

Real World Example: Porting Efforts

My Boss Walked Into My Office…

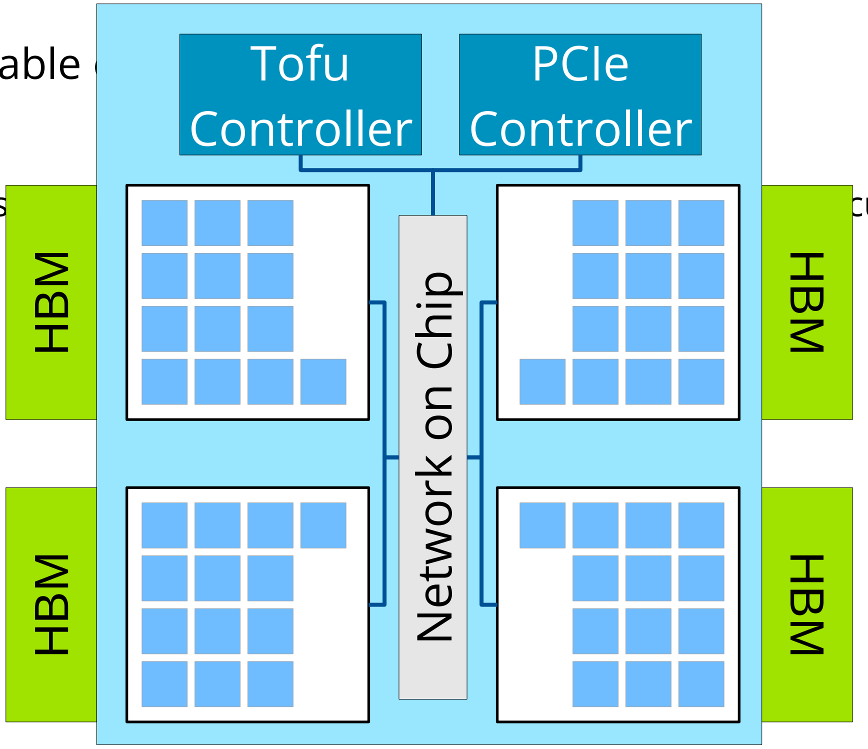
“Are you interested in testing alpaka on an experimental cluster?”

- New ARM-based CPU hardware available on cluster
- Node type #1: Fujitsu A64FX
  - Featured in Fugaku supercomputer (Fastest supercomputer from June 2020 – November 2021, currently #2)

My Boss Walked Into My Office…

“Are you interested in testing alpaka on an experimental cluster?”

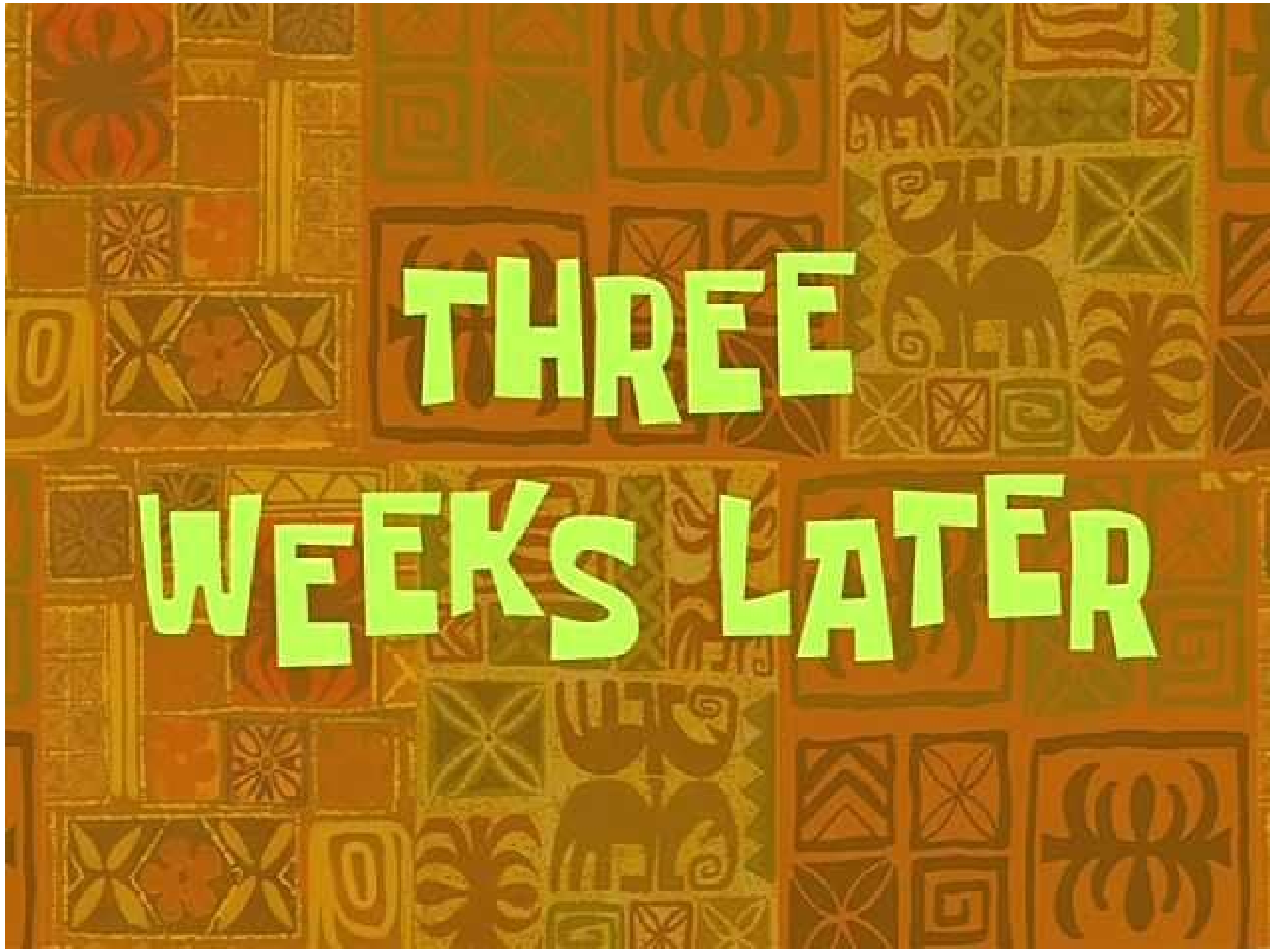
- New ARM-based CPU hardware available
- Node type #1: Fujitsu A64FX
  - Featured in Fugaku supercomputer (Fastes currently #2)



My Boss Walked Into My Office…

“Are you interested in testing alpaka on an experimental cluster?”

- New ARM-based CPU hardware available on cluster
- Node type #1: Fujitsu A64FX
  - Featured in Fugaku supercomputer (Fastest supercomputer from June 2020 – November 2021, currently #2)
- Node type #2: Ampere Altra CPUs + NVIDIA A100 GPUs
  - Envisioned for enterprise-grade server applications
  - 80 cores @ 3.30 GHz (max)
  - No SIMD registers


THREE
WEEKS LATER

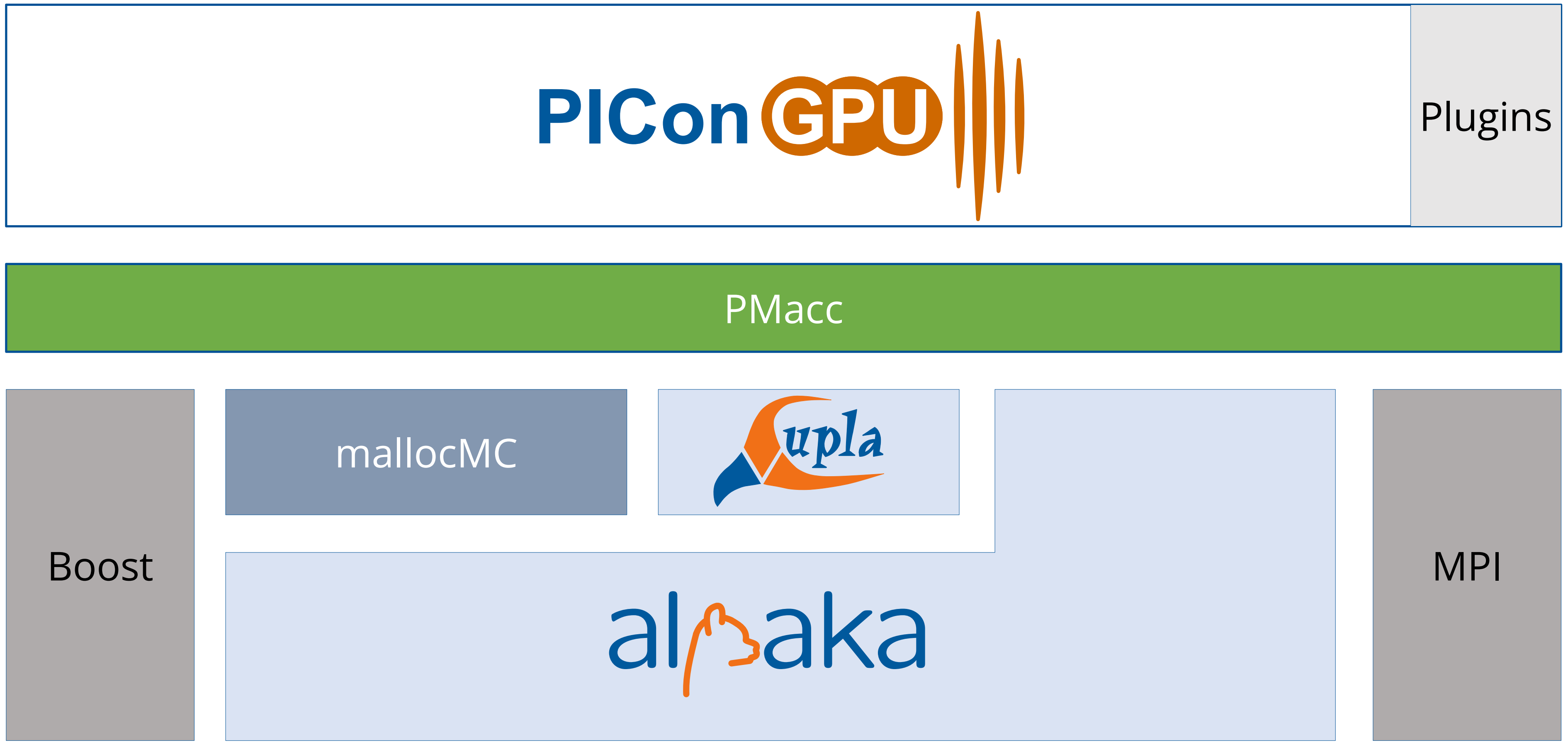
Three Weeks Later…

“We want you to port PIConGPU to the experimental hardware.”

- Fully relativistic, 3D3V particle-in-cell code
- Implements various numerical schemes to solve the PIC cycle
- Full restart and output capabilities
- 2D and 3D live view and diagnostic tools
- Many plugins available
- Utilization of several thousand GPUs possible
- 2013: Finalist of Gordon Bell prize (for scalability to > 18,000 GPUs)

https://www.github.com/ComputationalRadiationPhysics/picongpu

PIConGPU Software Stack



Porting Effort

Required efforts

- Some dependencies were not available on the cluster → self-compiled
  - Boost 1.78.0
  - libpng /pngwriter
  - openPMD
- PIConGPU's build system required support for `armclang++` compiler

Worked out-of-the-box

- Could compile for CPU architecture and utilize all CPU cores
- Could immediately utilize NVIDIA GPUs

→ **no C++ code changes required!**

Benefits of Using alpaka

Using alpaka saved several hours of porting efforts!

- alpaka's core is written in **standard** C++17
- x86-specific parts were made optional in 2021

  → Improvements in alpaka directly improve existing user code
- Utilization of ARM CPUs through alpaka's OpenMP back-end
- Utilization of NVIDIA GPUs through alpaka's CUDA back-end

Example: Maintenance & Future-Proofing

Example: CERN CMS Patatrack

- CMSSW reconstruction software currently (Run-3) runs on ~30,000 x86 CPUs
- Run-4 planned for 2027+ will see several hardware / detector upgrades for experiment
- CPUs will not be performant enough to process (much) larger data sizes
- Accelerators are required

Example: CERN CMS Patatrack

- CMSSW reconstruction software currently (Run-3) runs on ~30,000 x86 CPUs
- Run-4 planned for 2027+ will see several hardware / detector upgrades for experiment
- CPUs will not be performant enough to process (much) larger data sizes
- Accelerators are required

- **How to plan for the future?**

Available Libraries

kokkos Developed by Sandia National Laboratories (USA)

RAJA Developed by Lawrence Livermore National Laboratory (USA)

SYCL™ Designed by the Khronos industry consortium (USA)
Implemented by hardware vendors

alpaka Developed by Helmholtz-Zentrum Dresden-Rossendorf (Germany)

Benefits of Abstraction Layers

- Unified API for users

  → Vendor and hardware type (almost) do not matter
- Support for future hardware is not the user’s problem

  → Implementers of abstraction layers will deal with this
- Easy comparison of hardware types
- **Less technical debt**

CASUS

CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

www.casus.science

HZDR HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

TECHNISCHE UNIVERSITÄT DRESDEN

CBG Max Planck Institute of Molecular Cell Biology and Genetics

UFZ HELMHOLTZ Centre for Environmental Research

Uniwersytet Wrocławski

SPONSORED BY THE

Federal Ministry of Education and Research

STAATSMINISTERIUM FÜR WISSENSCHAFT KULTUR UND TOURISMUS

Freistaat SACHSEN