The alpaka SYCL back-end

Patatrack Students Meeting // 18 October 2022

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

(Short) Introduction to alpaka

Introduction to alpaka

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

alpaka – Abstraction Library for Parallel Kernel Acceleration

alpaka is…

- A parallel programming library: Accelerate your code by exploiting your hardware‘s parallelism!
- An abstraction library: Create portable code that runs on CPUs and GPUs!
- Free & open-source software

Introduction to alpaka

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Programming with alpaka

- C++ only!
- Header-only library: No additional runtime dependency introduced
- Modern library: alpaka is written entirely in C++17
- Supports a wide range of modern C++ compilers (g++, clang++, Apple LLVM, MS Visual Studio)
- Portable across operating systems: Linux, macOS, Windows are supported

Introduction to alpaka

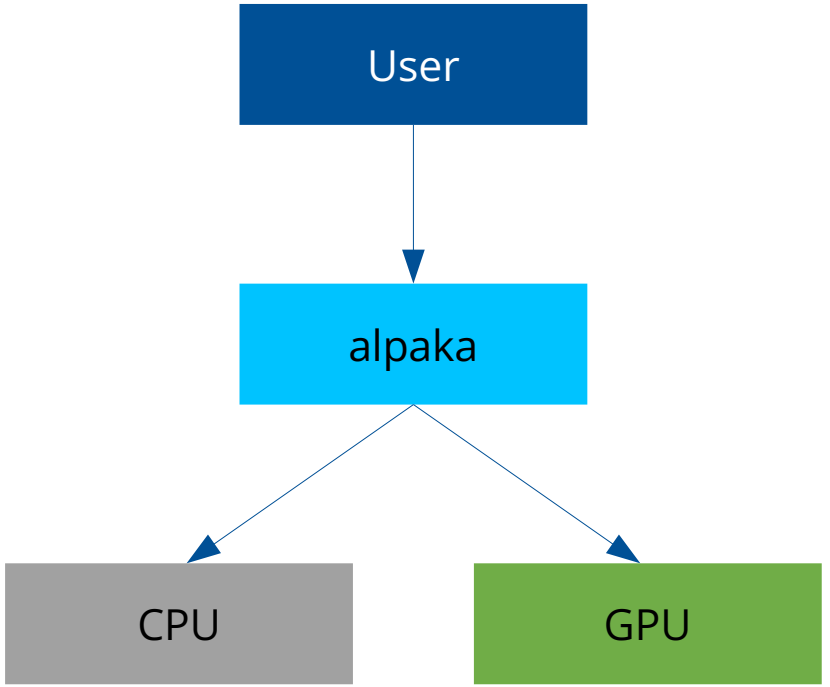
alpaka's purpose

Without alpaka

- Multiple hardware types commonly used (CPUs, GPUs, ...)
- Increasingly heterogeneous hardware configurations available
- Platforms not inter-operable → parallel programs not easily portable

alpaka: one API to rule them all

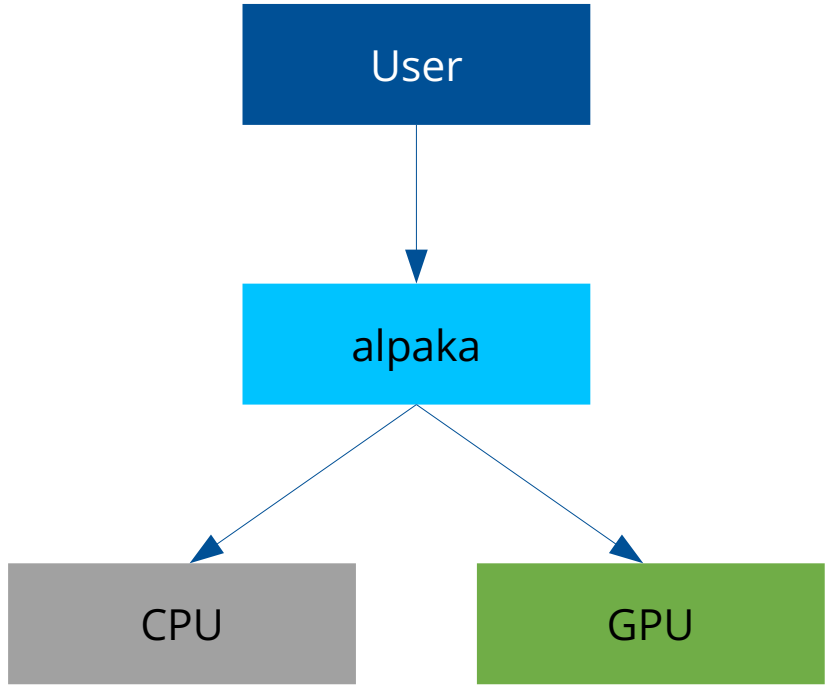
- Abstraction (not hiding!) of the underlying hardware & software platforms
- Code needs only minor adjustments to support different accelerators



Portable Heterogeneous Parallel Programming

alpaka enables portability!

- Idea: Write algorithms once…
  - … independently of target architecture
  - … independently of available programming models
- Decision on target platform made during compilation
  - Choosing another platform just requires another compilation pass
- alpaka defines an abstract programming model
- alpaka utilizes C++17 to support many architectures
  - CUDA, HIP, OpenMP, TBB, …



CASUS
CENTER FOR ADVANCED
SYSTEMS UNDERSTANDING

The SYCL back-end

Initial Goals

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

- Started as student project (diploma thesis) in late 2019
- Main interest: alpaka-based programs for FPGAs
- Available vendor at Helmholtz-Zentrum Dresden-Rossendorf: Xilinx

Initial Goals

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

- Started as student project (diploma thesis) in late 2019
- Main interest: alpaka-based programs for FPGAs
- Available vendor at Helmholtz-Zentrum Dresden-Rossendorf: Xilinx

- **Implement a prototype for a SYCL back-end**
- **Support the Xilinx SYCL implementation**

Initial Goals

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

- Started as student project (diploma thesis) in late 2019
- Main interest: alpaka-based programs for FPGAs
- Available vendor at Helmholtz-Zentrum Dresden-Rossendorf: Xilinx

- **Implement a prototype for a SYCL back-end**
- **Support the Xilinx SYCL implementation**

- **Experimental** SYCL back-end merged in February 2022
- Supports Intel oneAPI and AMD/Xilinx SYCL implementations

Current State

Missing Features

- No pointer support in kernels

Current State

Missing Features

- No pointer support in kernels
  - Users must use `alpaka::experimental::accessor` instead

Current State

Missing Features

- No pointer support in kernels
  - Users must use `alpaka::experimental::accessor` instead
- No RNG library support

Current State

Missing Features

- No pointer support in kernels
  - Users must use `alpaka::experimental::accessor` instead
- No RNG library support
- Not CI tested

Current State

Missing Features

- No pointer support in kernels
  - Users must use `alpaka::experimental::accessor` instead
- No RNG library support
- Not CI tested
- No special codepaths for FPGAs

Current State

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Missing Features

- No pointer support in kernels
  - Users must use `alpaka::experimental::accessor` instead
- No RNG library support
- Not CI tested
- No special codepaths for FPGAs
- FPGA images are not being reused

Current State

Missing Features

- No pointer support in kernels
  - Users must use `alpaka::experimental::accessor` instead
- No RNG library support
- Not CI tested
- No special codepaths for FPGAs
- FPGA images are not being reused
- Some smaller functionality is missing
  - Lack of functionality in SYCL

Current State

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Active Work

- Add pointer support for Intel oneAPI targets
- Require `alpaka::experimetal::accessor`s only for AMD/Xilinx targets

Short-term Goal

- Add CI support for Intel oneAPI

Long-term Goals

- Improve FPGA support / performance
- Make use of vendor-specific SYCL extensions

Help Wanted

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

Contributors welcome!

- Find us on **https://github.com/alpaka-group/alpaka**
- Go to the issue tracker and filter for the “Backend:SYCL” flair

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