alpaka Parallel Programming – Online Tutorial

Lecture 10 – The alpaka Programming Model

Lesson 13: Handling Parallelism

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

Lesson 13: Handling Parallelism

CASUS
CENTER FOR ADVANCED SYSTEMS UNDERSTANDING

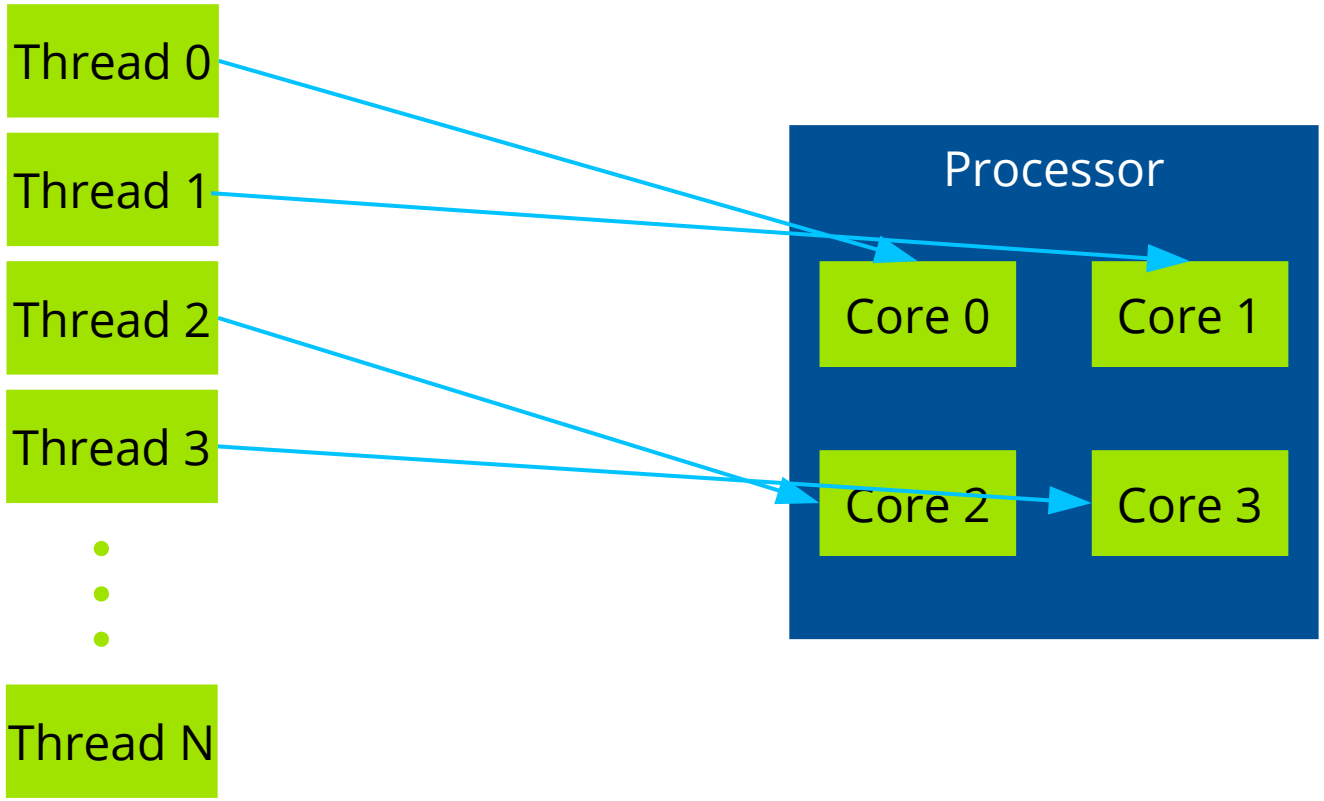
Threads and cores

- alpaka Threads are different from pthreads, `std::threads`, OpenMP threads, CUDA threads, etc.
- alpaka Thread: execution of command sequence
- Command sequence: algorithm performed on single data element (Kernel)
- Cores are physical execution units
- Cores are capable of executing alpaka Threads
- Example: AMD Threadripper 3990X with 64 CPU cores
- Example: NVIDIA Tesla V100 with 5,120 CUDA cores

Lesson 13: Handling Parallelism

Mapping Threads to cores

- alpaka Threads are mapped to hardware cores
- While running, one Thread is executed by exactly one core
- Threads may run on other cores after rescheduling
- Usually many more Threads than cores (*oversubscription*)
- Waiting Threads make room for ready Threads



Lesson 13: Handling Parallelism

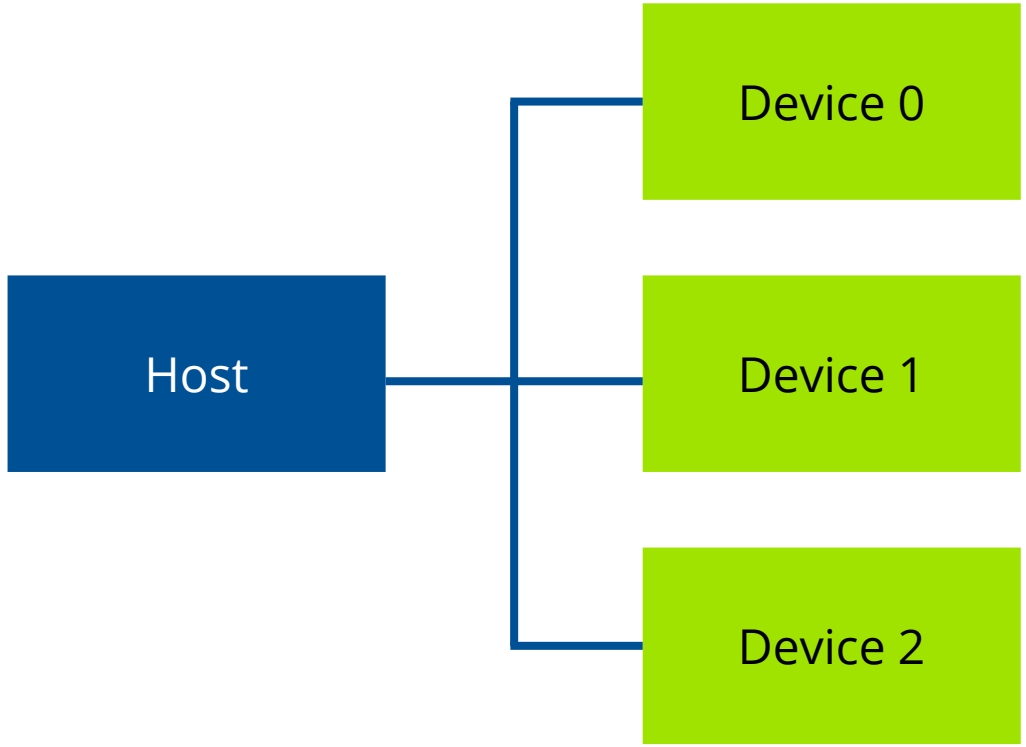
alpaka Devices

- A set of cores is called a Device
- A single core can only belong to exactly one Device (N:1 mapping)
- All cores on the Device have access to global memory
- alpaka Devices correspond to physical devices
- Example: AMD Threadripper 3990X with 64 CPU cores is a Device with 128 cores (simultaneous multithreading!)
- Example: NVIDIA Tesla V100 with 5,120 CUDA cores is a Device with 5,120 cores

Lesson 13: Handling Parallelism

Host and Device

- An alpaka Host controls the overall program flow
- An alpaka Device executes Kernels
- All Devices are attached to a single Host
- It is impossible to have more than one Host



Lesson 13: Handling Parallelism

Modifying Hello World

- From your `build` directory, switch to the `src` directory and open `helloWorld.cpp` with your favourite editor:

```
cd ../src
vim helloWorld.cpp
```

- Go to line 73 and change the number of threads (we will explain the Block-Grid terminology later):

```
Idx blocksPerGrid = 16;
```

- Go back to `build`, rebuild (`cmake --build . --config Release`) and execute again

Lesson 13: Handling Parallelism

Results

- Example output:

```
Hello, World from alpaka thread 2!
Hello, World from alpaka thread 3!
Hello, World from alpaka thread 14!
Hello, World from alpaka thread 5!
Hello, World from alpaka thread 11!
Hello, World from alpaka thread 13!
Hello, World from alpaka thread 8!
Hello, World from alpaka thread 6!
Hello, World from alpaka thread 7!
Hello, World from alpaka thread 4!
Hello, World from alpaka thread 0!
Hello, World from alpaka thread 1!
…
```

- You have successfully increased the number of alpaka threads!

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