Karlsruhe Institute of Technology

Introduction and status report

Matthias Vogelgesang

matthias.vogelgesang@kit.edu

Institute for Data Processing and Electronics

Introduction

Welcome

- Thanks in advance for showing interest and attending this workshop 👍
- This workshop is affiliated with the general meeting of the Helmholtz program *Matter and Technologies*

Introduction

Welcome

- Thanks in advance for showing interest and attending this workshop 👍
- This workshop is affiliated with the general meeting of the Helmholtz program *Matter and Technologies*

Scope of this workshop

- Get to know each other within Helmholtz and beyond
- Be a platform for users and developers alike
- Consists of a series of talks covering different aspects of many-core technologies

Parallel computing technologies

13:00	Erik Zenker – *Core Concepts – Zero Overhead Abstractions for Scalable Many-core Data Analysis*
	Fabian Jung – *A scalable Experiment to HPC Cluster Solution for Data Analysis*
14:00	Tobias Stockmanns, Hannes Mohr – *GPUs for Track Trigger in High Energy Physics*
	Coffee break

Applications

15:00	Felix Beckmann – *Computational Challenges for Microtomography using Synchrotron Radiation at PETRA III*
	André Bieberle – *Analysis of Hydrodynamic Effects using Ultrafast X-ray Tomography with GPU-accelerated Data Acquisition*
16:00	Malte Zacharias – *Just-in-time Dosimetry using Positron Emission Tomography*
	Alexander Matthes – *Live, Steerable in-situ Visualization for High Performance Computing*

Visualization & optimization

16:00	Malte Zacharias – *Just-in-time Dosimetry using Positron Emission Tomography*
	Alexander Matthes – *Live, Steerable in-situ Visualization for High Performance Computing*
17:00	Nicholas Tan Jerome – *3D Web-Visualization Technologies*
	Suren Chilingaryan – *Optimizing Algorithms for Parallel Architectures*

Remaining days

Tomorrow

- We will offer two tutorials related to GPU programming
- NVIDIA GPGPU within a Docker environment (Andrei Shkarin)
- Set up, usage and extension of the UFO framework (myself)

General meeting

If you attend the “Matter and Technologies” meeting, please *keep* your badges.

Lunch

Lunch tickets will be offered in the Tulla lecture hall, ask me or Andreas for further details.

STATUS

UFO core

- Mostly cleanup and various fixes
- ZMQ and MPI are entirely optional now
- Environment variable `UFO_DEVICES` generalizes `CUDA_VISIBLE_DEVICES`
- `ufo-launch` learned to `--dump` and `--trace` and comes with bash completion
- Support for plugin packages (Buldygin)

UFO plugins

Improved plugins

- `backproject` fixed and optimized
- `read` can load raw files and override type detection
- `fft`, `ifft` and `phase-retrieve` use a common code base

New plugins

- `ir` for iterative reconstruction (Buldygin, Shkarin)
- `stdin` and `stdout` for better UNIX pipeline integration
- `merge`, `loop` and `monitor` to manipulate and inspect the stream
- `flip` and `clip` for basic image operations
- `dgma` for RDMA between FPGA and GPU

UFO plugins

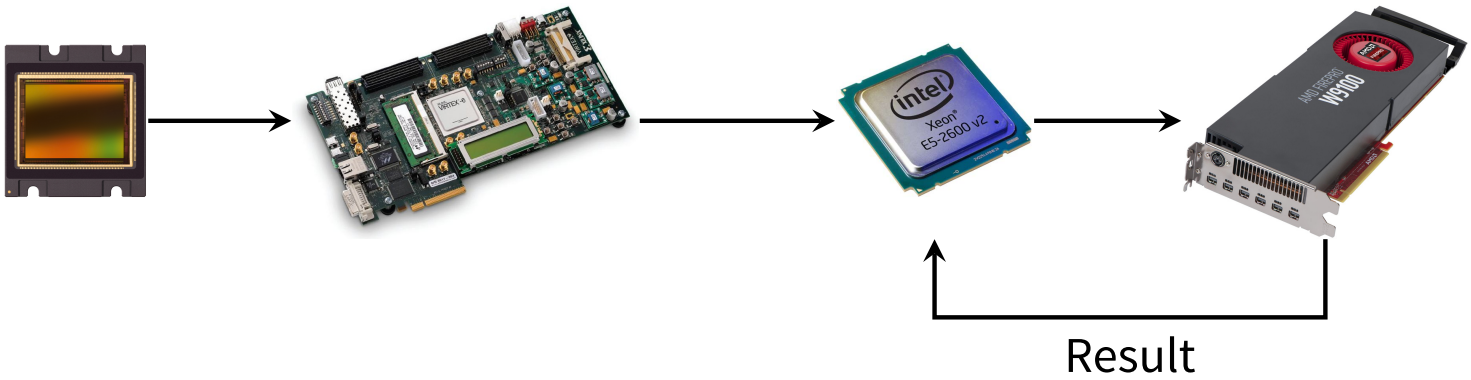
Improved plugins

- `backproject` fixed and optimized
- `read` can load raw files and override type detection
- `fft`, `ifft` and `phase-retrieve` use a common code base

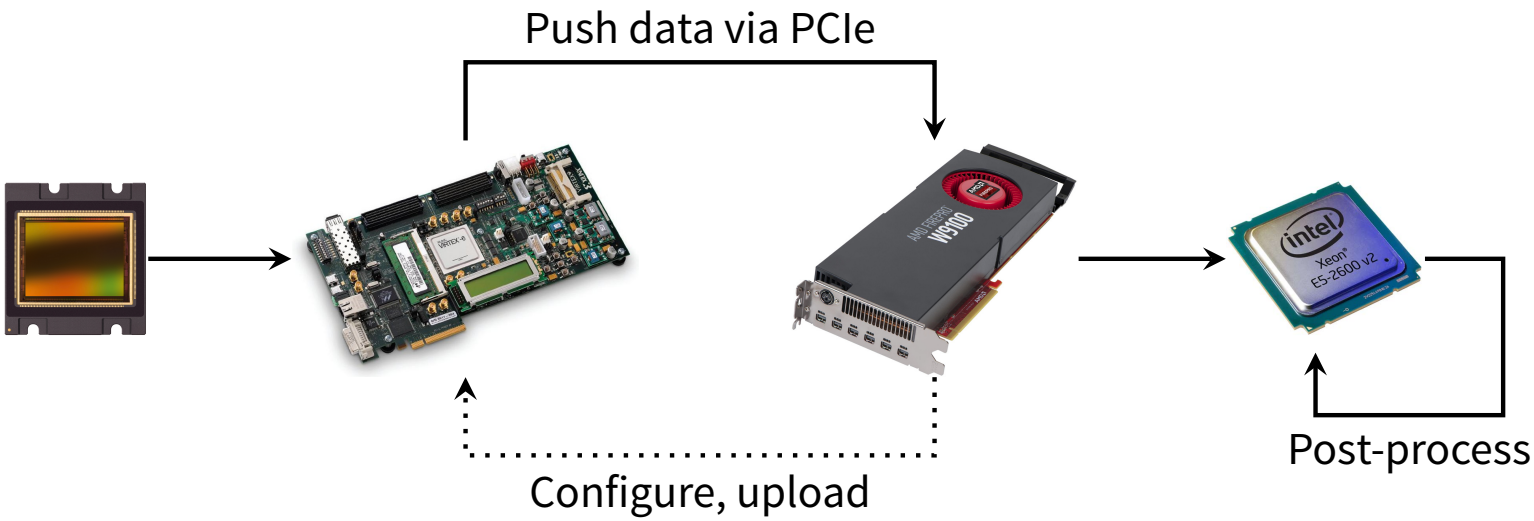
New plugins

- `ir` for iterative reconstruction (Buldygin, Shkarin)
- `stdin` and `stdout` for better UNIX pipeline integration
- `merge`, `loop` and `monitor` to manipulate and inspect the stream
- `flip` and `clip` for basic image operations
- `dgma` for RDMA between FPGA and GPU

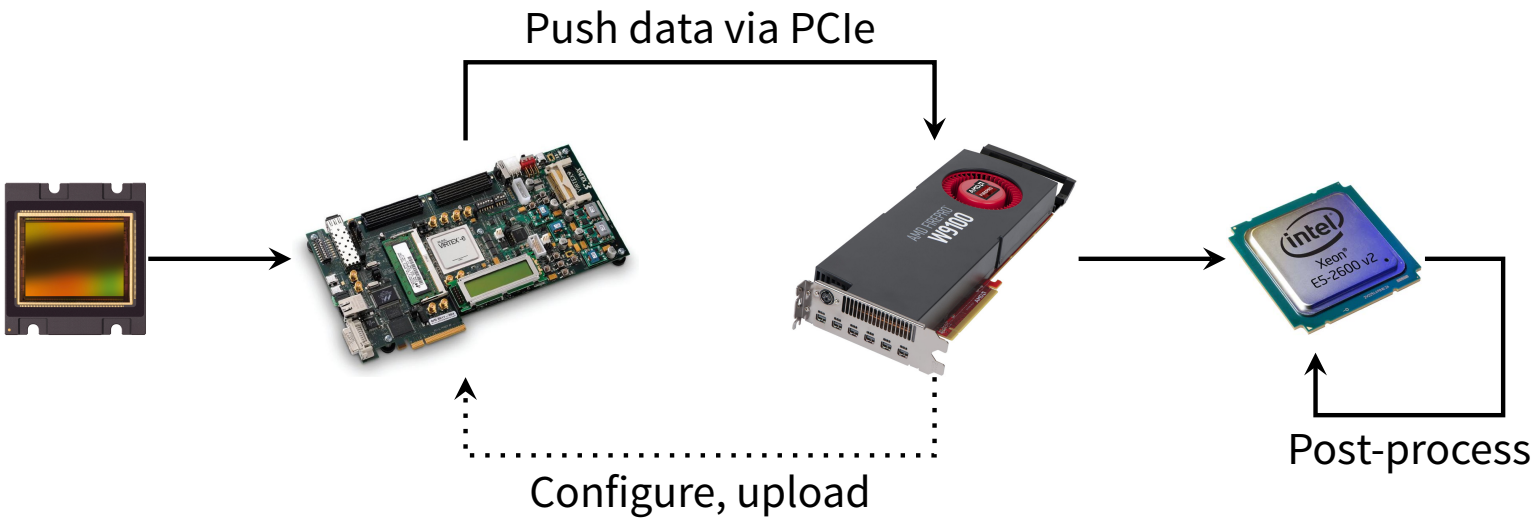
FPGA-GPU data acquisition



FPGA-GPU data acquisition



FPGA-GPU data acquisition



Benefits: lower latency and fewer memory transfers

Involved hardware

FPGA

- Xilinx VC709 (PCIe 3.0) on the frontend side (Caselle, Rota)
- On-board DDR memory management added recently (Ardila Perez)
- Linux driver used to access the FPGA via PCIe (Chilingaryan)

GPU

- AMD FirePro W9100 on the backend
- AMDs *DirectGMA* RDMA OpenCL extension for buffer management
- Writable buffers are limited in size ($\approx$ 100 MB)

Writing across devices

Writing to GPU

1. Create OpenCL buffer with `CL_MEM_BUS_ADDRESSABLE_AMD`
2. Determine address by passing buffer to `clEnqueueMakeBuffersResidentAMD()`
3. Set address and paging information in FPGA control registers
4. Busy-wait until FPGA flag signals completion

Writing to FPGA

1. Create OpenCL buffer with `CL_MEM_EXTERNAL_EXTERNAL_PHYSICAL_AMD`
2. Determine physical address of FPGA's "memory space"
3. Pass buffer and address to `clEnqueueMakeBuffersResidentAMD()`
4. Writes from the GPU are proxied transparently to the FPGA

Integration into UFO

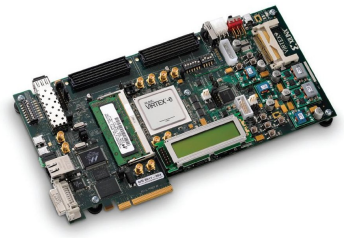
New dgma plugin

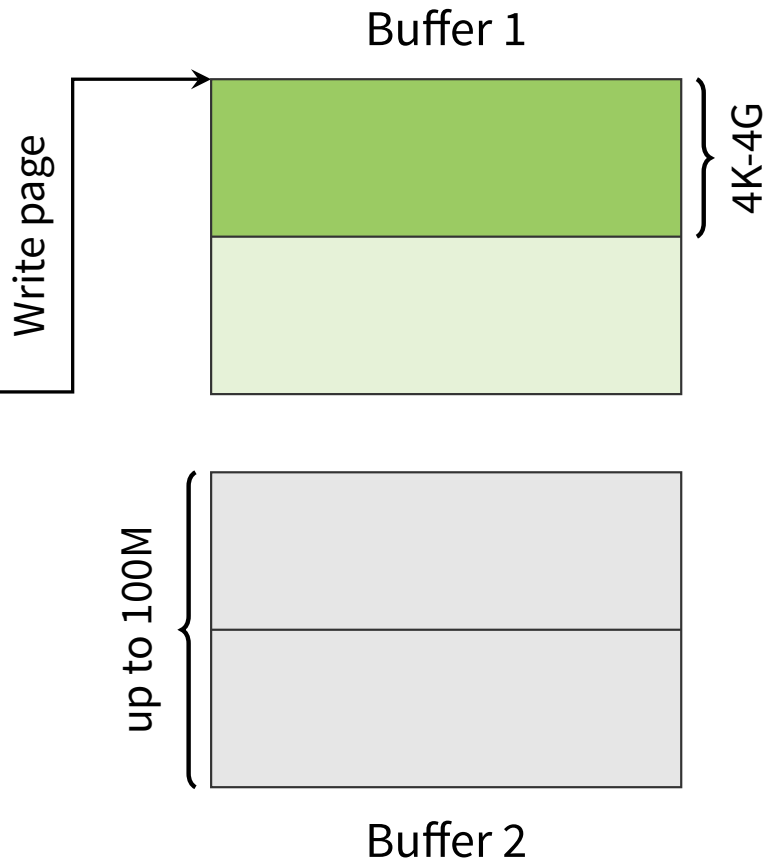
- Encapsulates FPGA communication protocol
- Schedules writes in asynchronous, double-buffered fashion for high throughput
- Optional data conversion to re-interpret raw data

With that plugin reading data is transparent to the end user, e.g.

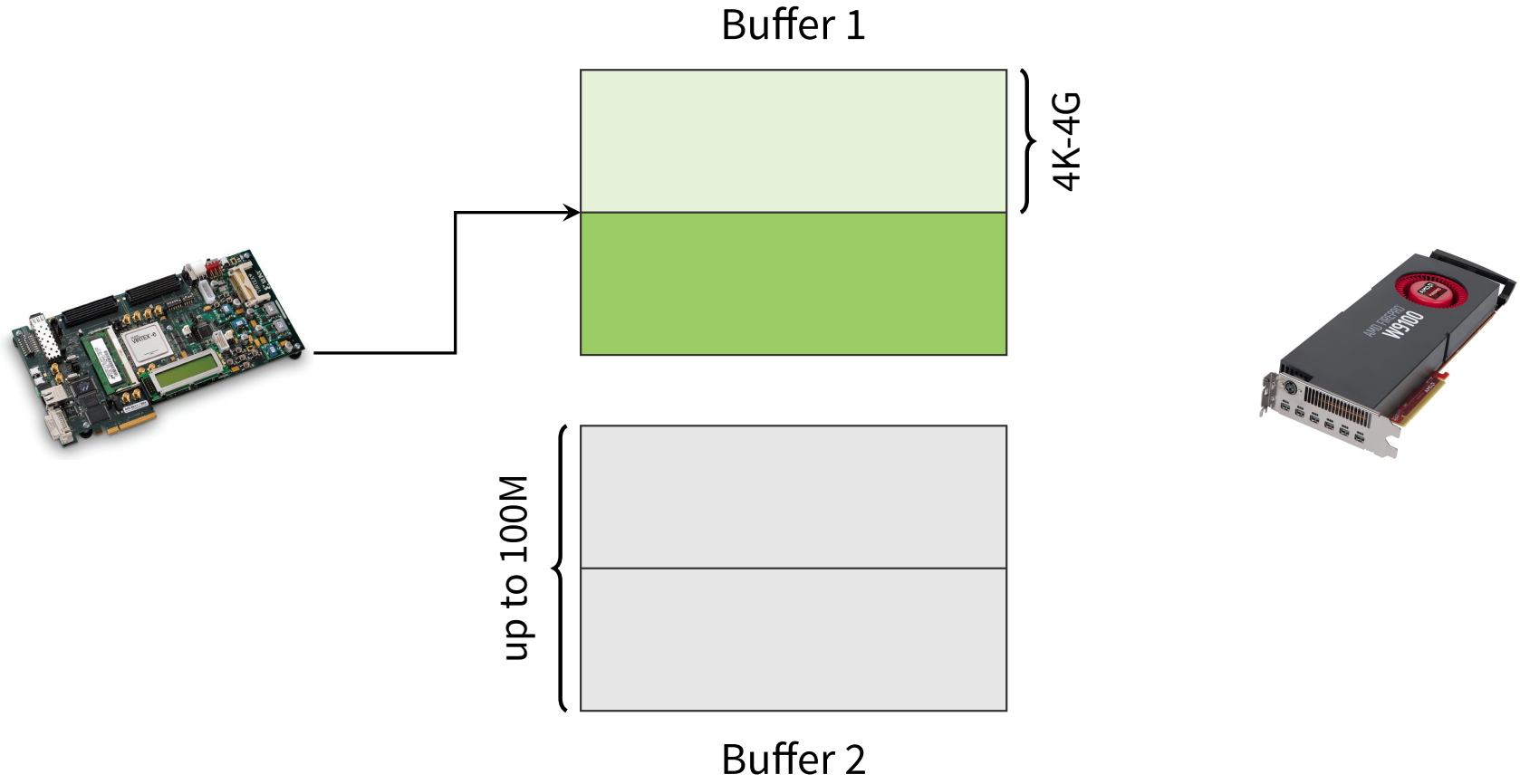
```
$ ufo-launch dgma number=1 num-pages=32
                  width=512 height=512 !
             stdout | gzip -c - > data.raw.gz
```

Double-buffered, paged readout

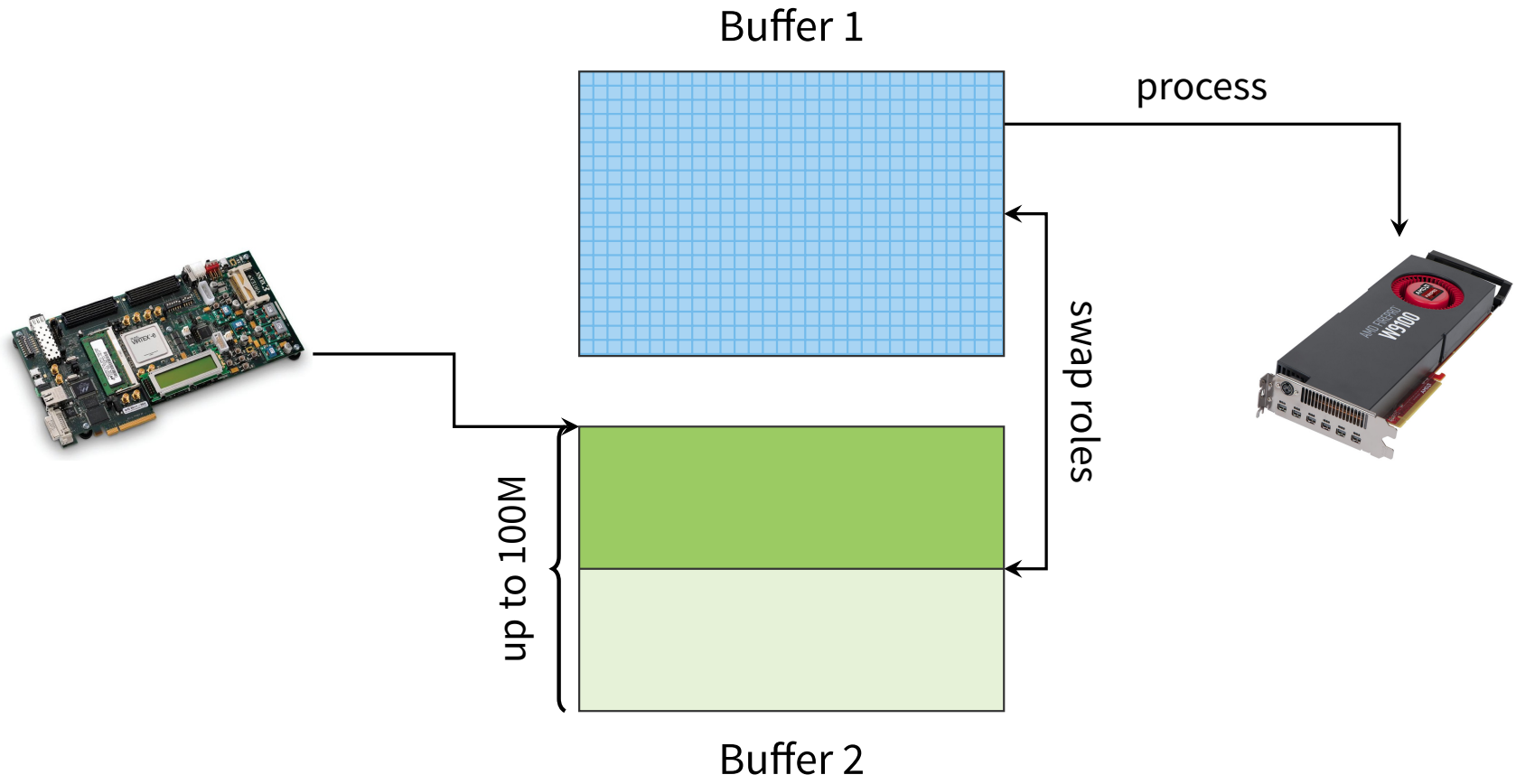




Double-buffered, paged readout



Double-buffered, paged readout



Larger target buffers

Problem

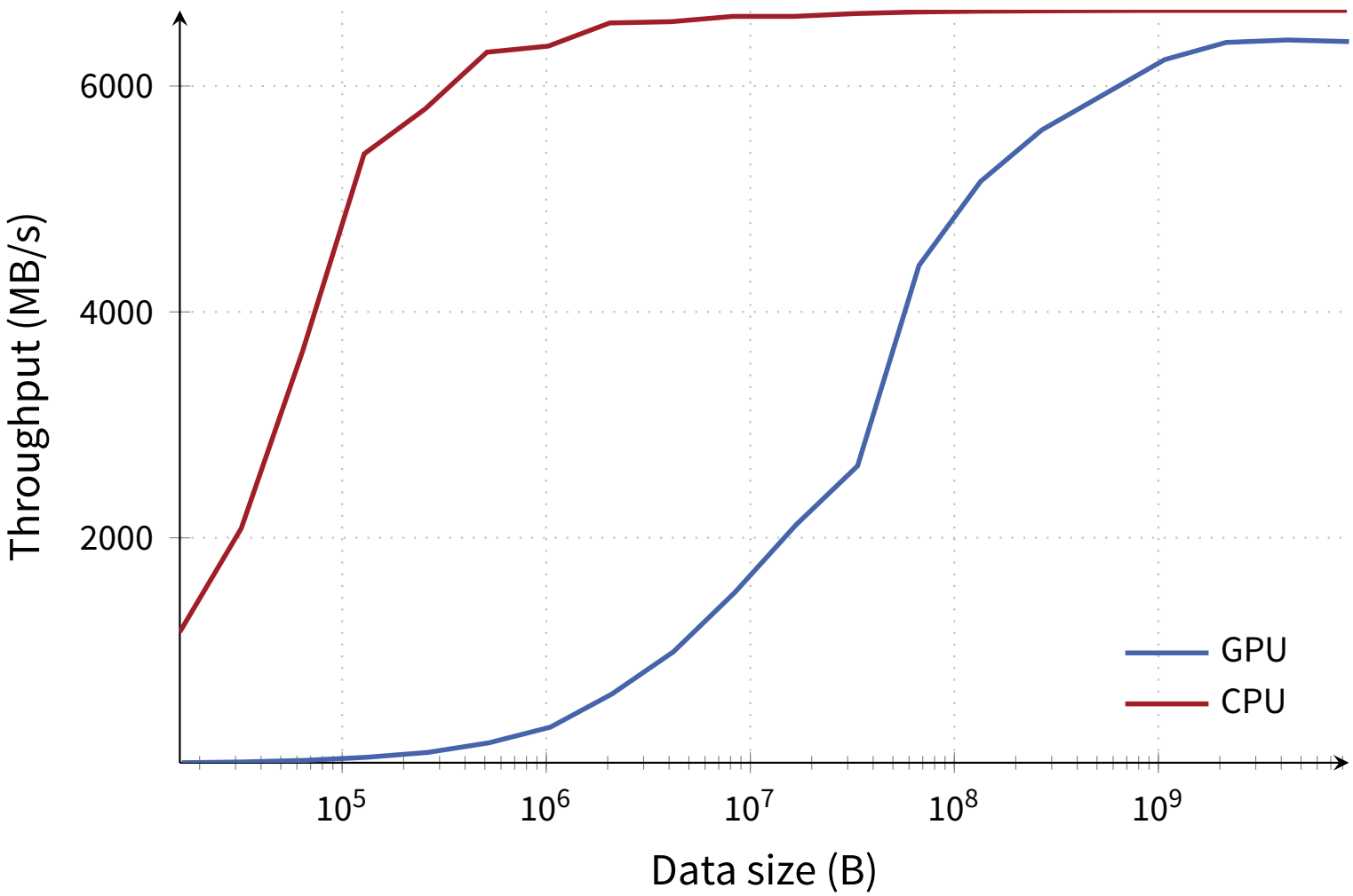
- Resident buffers can only be some hundred MBs large
- FPGA can write to a 4 GB address space …

Solution: intermediate resident buffers

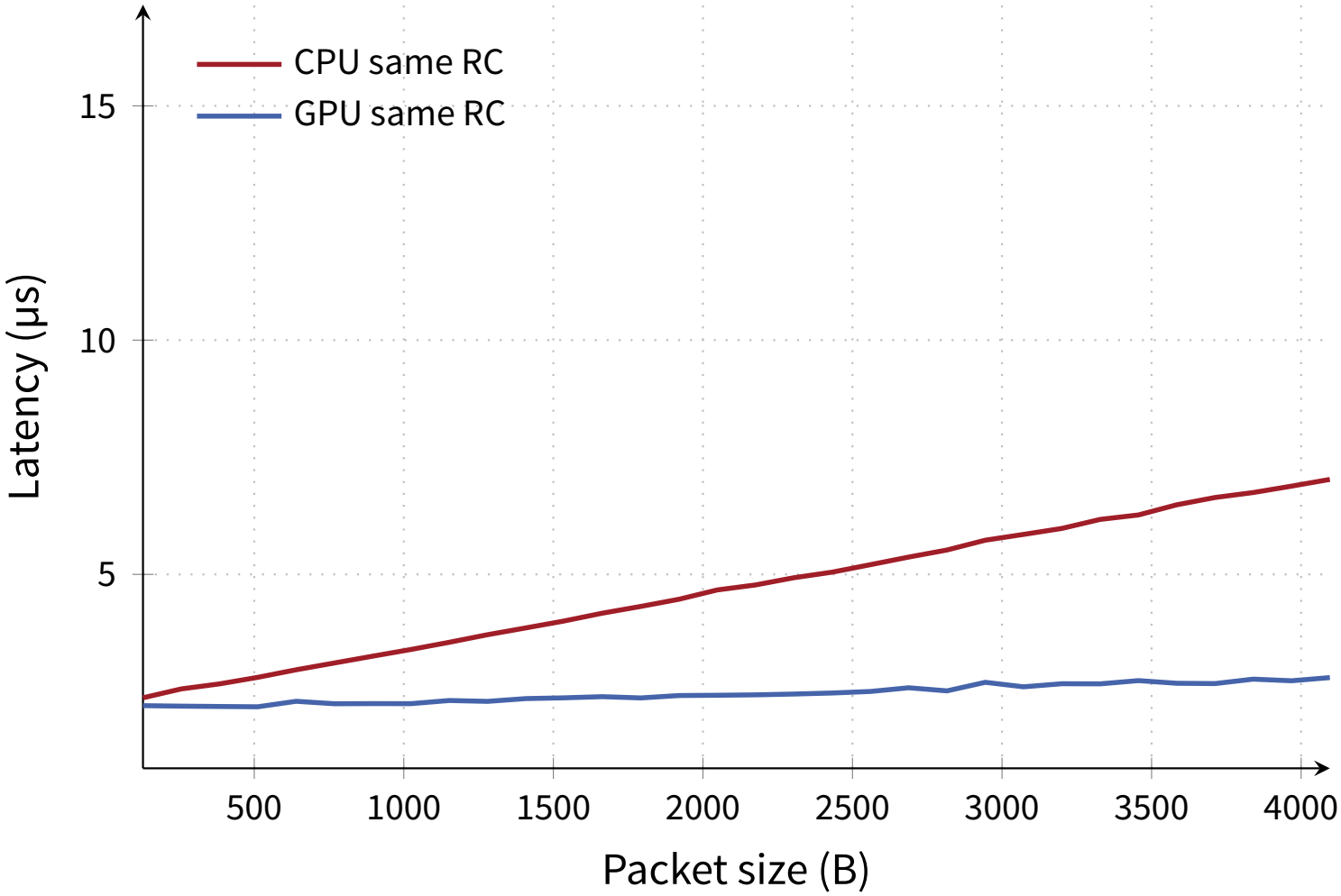
1. Allocate large target buffer
2. Write chunks of 32 MB ($\approx$ half the resident buffer size)
3. `clEnqueueCopyBuffer()` the chunk to target buffer[1]
4. Immediately start transferring next chunk

[1] No problem with 40 GB/s intra-GPU bandwidth at 5 MB blocks.

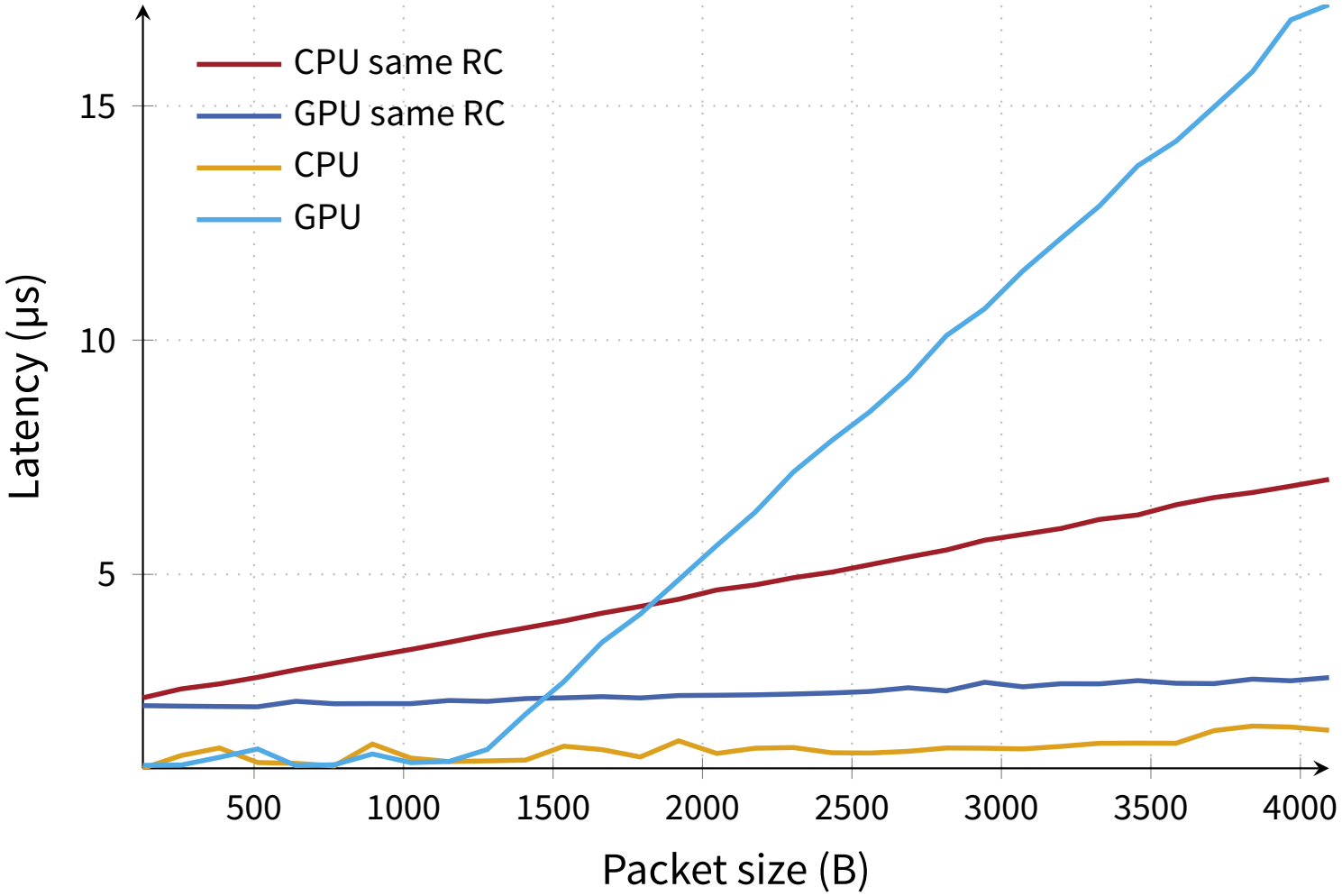
Throughput (old)



Roundtrip latency (old)



Roundtrip latency (old)



Results

Performance

- We are able to hit peak PCIe bandwidth to the GPU
- Data transfer latencies are promising for timing-sensitive applications
- Lowest latencies require use of special boards

Caveats

- Limited DGMA buffer size requires partitioning
- AMDs OpenCL implementation causes an initial $\approx$ 150 µs dead time
- Synchronization facility does not work as specified

Next steps

- Clarify implementation details with AMD
- Finish investigations with NVIDIA GPUs
- Improve performance and evaluate further applications

Further activities at IPE and ANKA

GPU-related

- Micro Particle Image Velocimetry (Miao with TVT)
- Improved tomo- and laminographic filtered backprojection (Farago at ANKA/IPS)
- Algebraic laminographic reconstruction (Reinke with ANKA/IPS)
- Pre-processing of beam monitoring data acquired with KALYPSO (Vogelgesang)

…and beyond

- Multimodal visualization of 2- and 3-dimensional data (Tan Jerome)
- Infrastructure for virtualized processing environments (Shkarin)
- High throughput data communication (Dritschler)
- High performance and custom image data acquisition (Sasidahr, Vogelgesang)

Karlsruhe Institute of Technology

ANY QUESTIONS?