Karlsruhe Institute of Technology

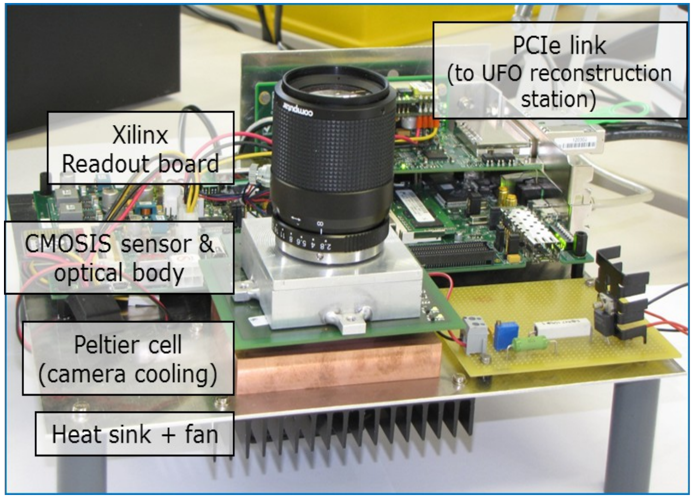
High-performance computing hardware for high data rates

Agenda

Parallel Computing:
    Possibilities & Challenges
Handling Data I/O at High Rates
Accelerating Synchrotron Tomography
Scaling to Cluster

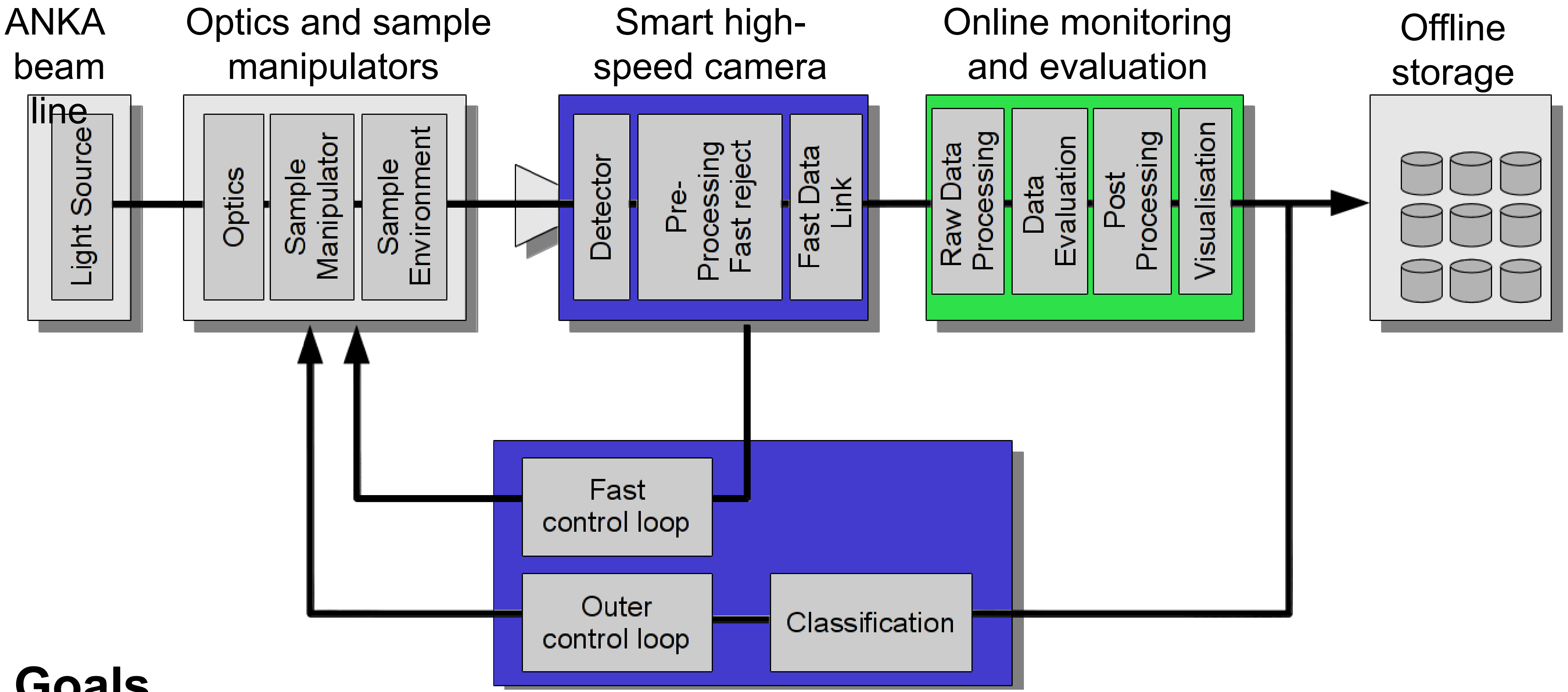
Authors

Suren A. Chilingaryan, KIT
Michele Caselle, KIT
Thomas van de Kamp, KIT
Andreas Kopmann, KIT
Alessandro Mirone, ESRF
Uros Stevanovic, KIT
Tomy dos Santos Rolo, KIT
Matthias Vogelgesang, KIT



UFO *Ultra Fast X-ray Imaging of Scientific Processes with On-Line Assessment and Data-Driven Process Control*

Karlsruhe Institute of Technology



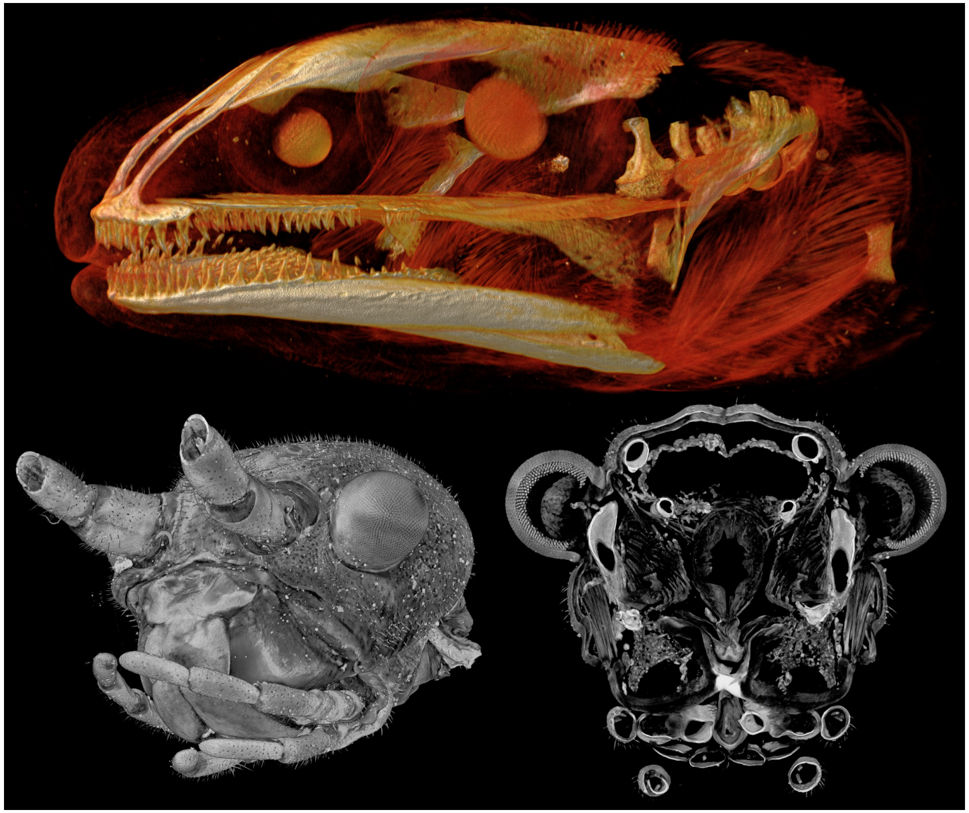
Goals

- High speed tomography
- Increase sample throughput
- Tomography of temporal processes
- Allow interactive quality assessment
- Enable data driven control
  - Auto-tunning optical system
  - Tracking dynamic processes
  - Finding area of interest

Reconstruction Problem

PCO.edge

Resolution: 2560 x 2160
Dynamic Range: 16 bit
Frame Rate: 100 fps



Heads of a newt larva showing bone formation and muscle insertions (top) and a stick insect (bottom), acquisition time 2s.

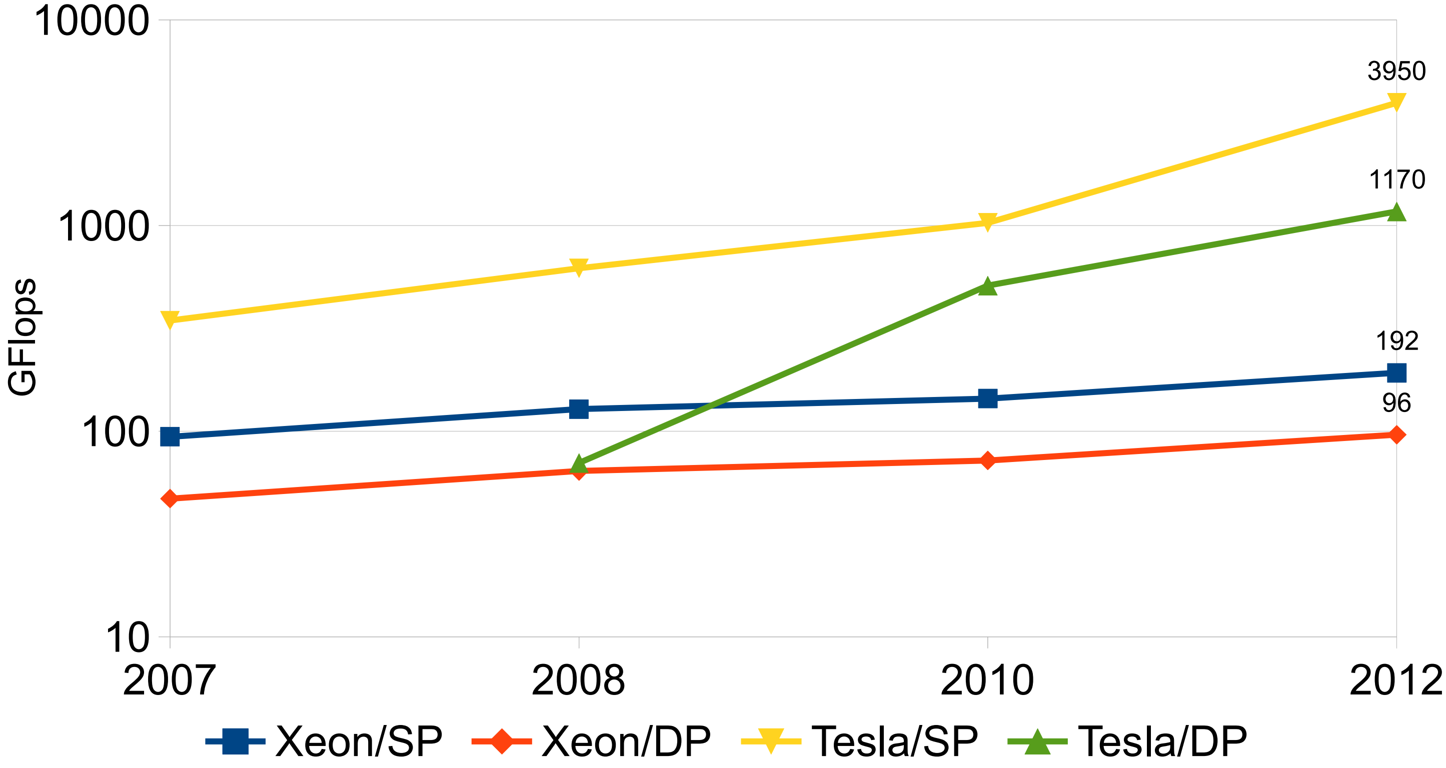
Tomographic Reconstruction
3D image: 2000^3
Projections: 2000
Acquisition time: 20 seconds

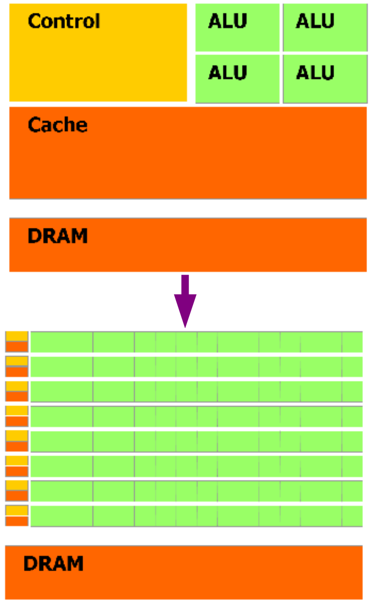
FBP Complexity: 144 Tflops
Xeon Performance: ~ 100 Gflops
Minimum time: ~ 15 minute on DP
Actually: ~ 1 hour

20 seconds acquisition
1 hour reconstruction

Parallel Architectures

Rise of GPU performance as compared to Xeons



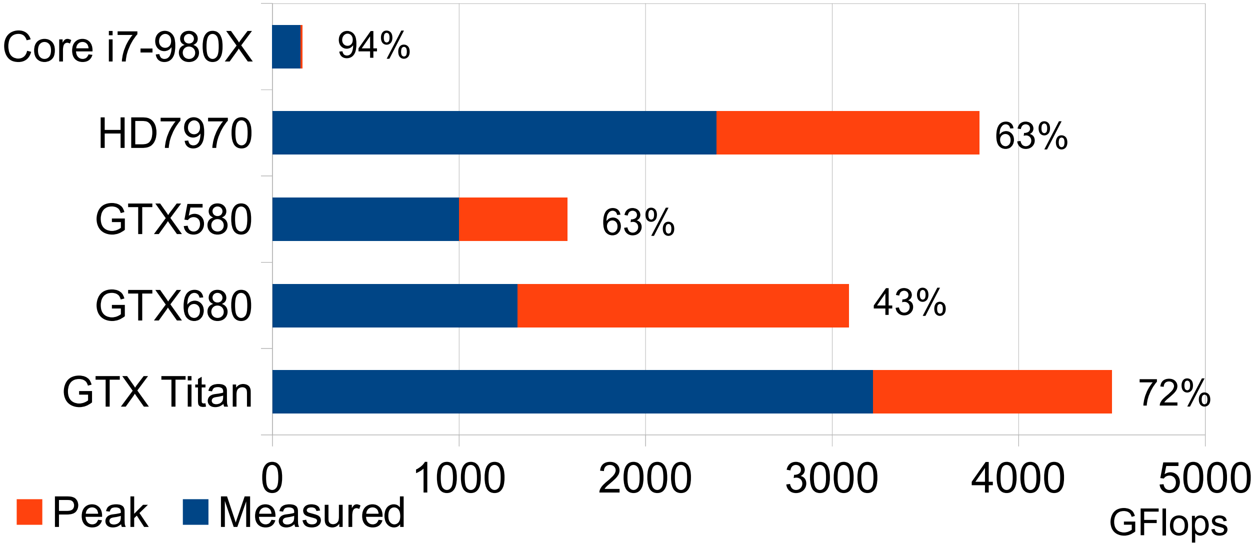


Parallel Architecture

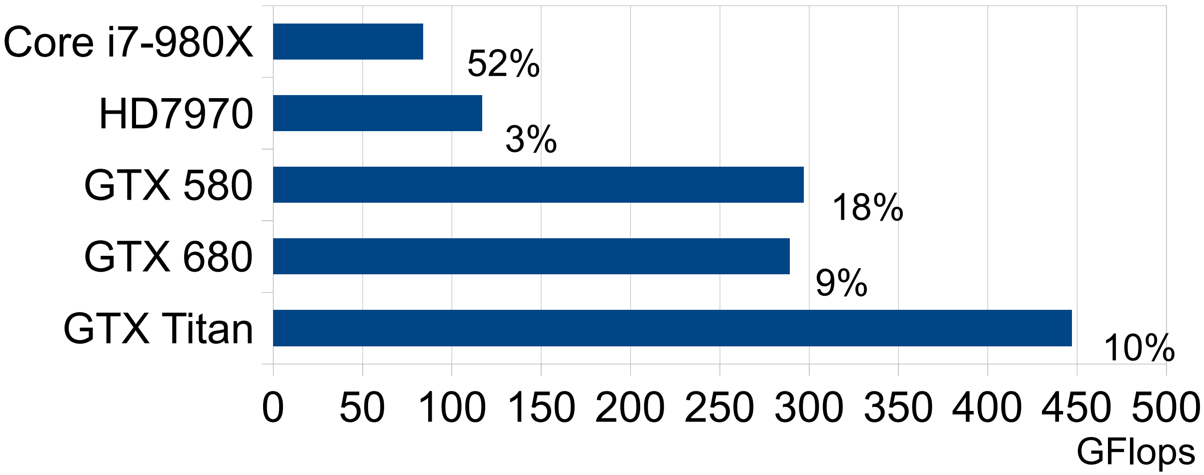
	E7-8870	Xeon/Phi	Tesla K20	GeForce Titan	AMD HD7970	Power7+
SP	192	2020	3950	**4500**	3790	265
DP	96	1010	1170	**1300**	950	132
Mem	34.11	**320**	250	288	264	68
Max dev.	8	?	8	8	>=4	32
Price	$4,800.00	$2,800.00	$3,200.00	$1,000.00	**$400.00**	$$$$$$$

Efficiency

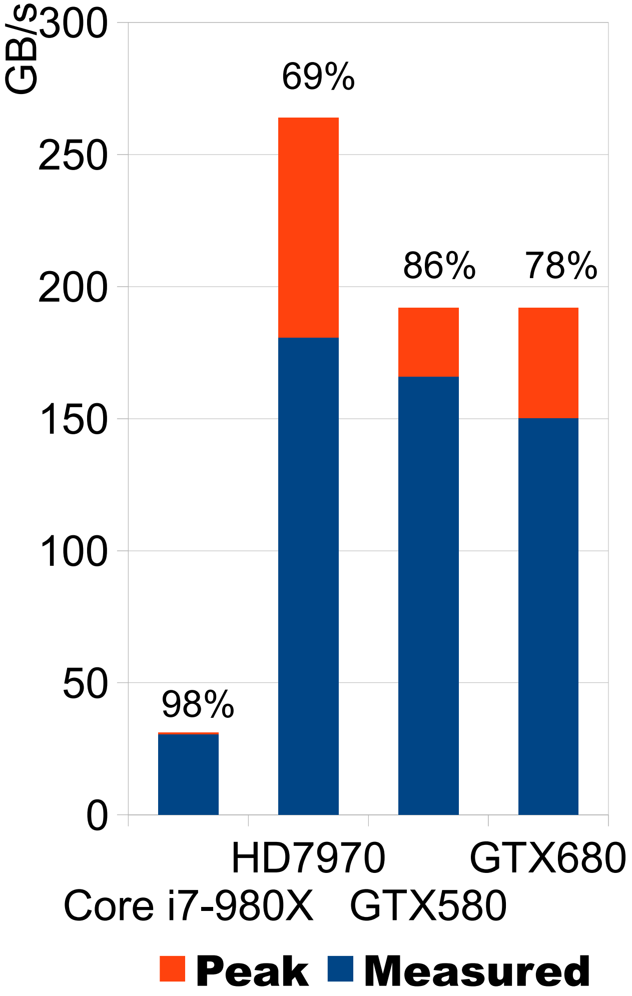
Matrix Multiplication



1D Fast Fourier Transform



Memory Bandwidth



GTX Titan performance is taken from Anandtech

GPU-programing considerations

- **Special programming tools and techniques are required**
  - Multiple different and ever-changing architectures
  - Branching, non-fp operations are very expensive
  - Optimized mathematical libraries are some times missing
- **Limited amount of memory and expensive data transfers**
  - x16 PCIe gen2 (8 GB/s), gen3 (16 GB/s)
  - Specially allocated (pinned) memory required for a full performance and to overlap computations and data transfers
- **Reduced caches, low memory to computation ratio, strict access patterns**
  - 177 GB/s per Teraflop for Xeon, 60 - 70 GB/s per Teraflop for GPUs
  - Varying cache hierarchies on different architectures
  - Special access patterns are required for better performance. For instance, bandwidth of matrix transpose (GTX280 with 142 GB/s memory bandwidth)
    - **2 GB/s –** for naive approach
    - **17 GB/s –** if shared memory is used
    - **80 GB/s –** if care taken for shared memory banks and global memory partitions
- **I/O problem**
  - 100 MB/s sequential write while camera produces ~ 1 GB/s
  - Handling big data sets not fitting in the memory (up to 500 GB)
- **Various problems with growing number of GPUs connected to a system**

Parallel Programming Environments

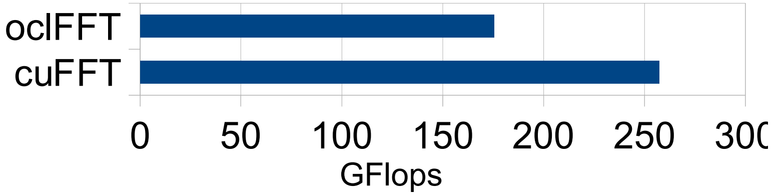
- **CUDA –** The oldest GPU programming technology from NVIDIA
- **OpenCL –** Open standard technology close to CUDA, but working with wide range of hardware platforms including CPUs and GPUs
- **OpenAC –** Declarative technology similar to OpenMP
- MATLAB and other mathematical packages with integrated GPU support. Only some operations are parallelized and necessity to transfer over slow PCIe bus to execute non-parallelized operations kill the performance.

CUDA

- **Supports latest NVIDIA technologies**
  - GPUDirect – direct transfers between GPU and IB, etc. Integration with MPI frameworks
  - Dynamic parallelism – GPUs are ablee to spawn new jobs
- **NVIDIA provides a set of highly optimized libraries (BLAS, FFT, Lapack, Reduction, etc.)**
- **Only NVIDIA GPUs are supported**

Programming Environments

OpenCL



- Syntax is very similar to CUDA (easy porting)
- Well written code is as fast as CUDA
- Works with CPUs and GPUs from multiple vendors (Intel, AMD, IBM, NVIDIA)
- Still no way to run code simultaneously on NVIDIA GPU and CPU, but possible with AMD cards
- Many libraries existing, but generally slightly slower than CUDA counterparts. Some libraries are only available commercially
- No GPUDirect, significantly limited options to use pinned memory (i.e. slower data transfers)

OpenACC

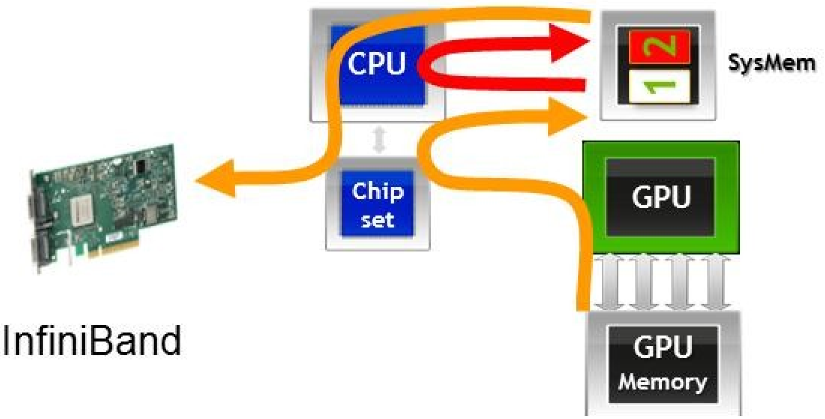
- Existing applications may be easily parallelized. Also developing new code is easy compared to OpenCL/CUDA
- No free compilers are existing at the moment. Though there is a similar technology OmpSS developed at Barcelona Supercomuter Center.
- At current level, technology does not support shared memory and some other technologies available with direct programing (i.e. it is slower)

GPUDirect and Frame Grabbing

Without GPUDirect

Same data copied three times:

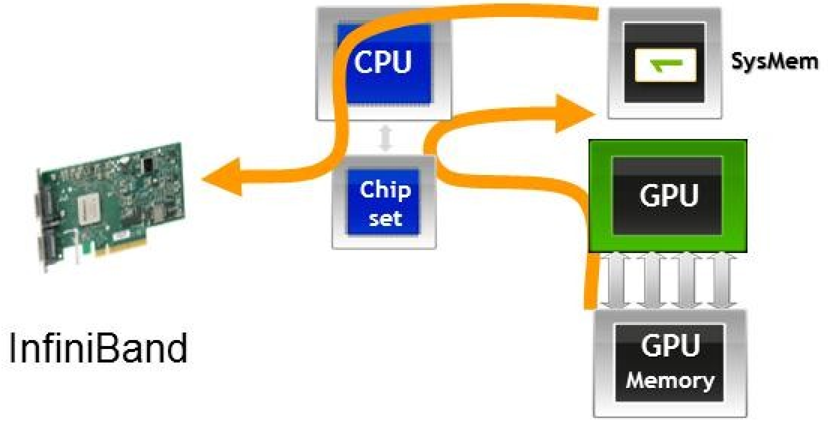
1. GPU writes to pinned sysmem1
2. CPU copies from sysmem1 to sysmem2
3. InfiniBand driver copies from sysmem2

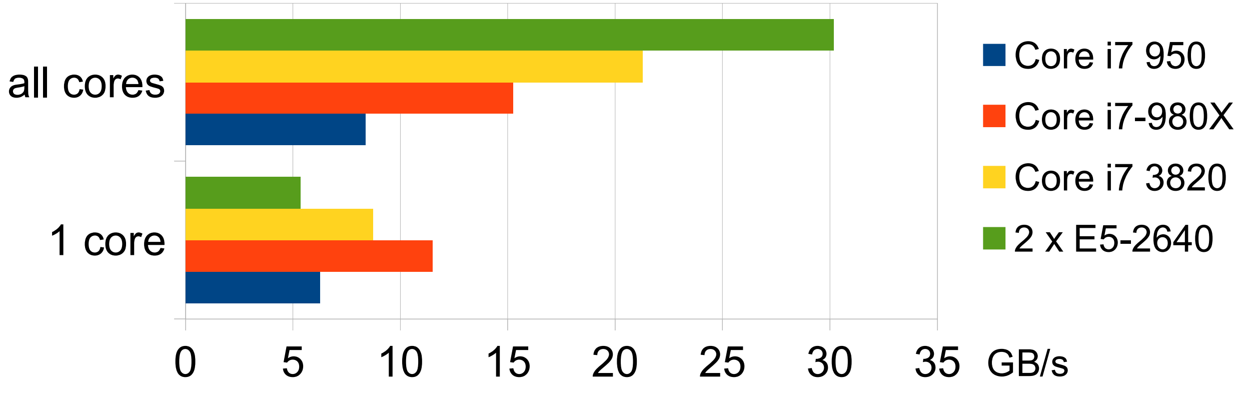


With GPUDirect

Data only copied twice

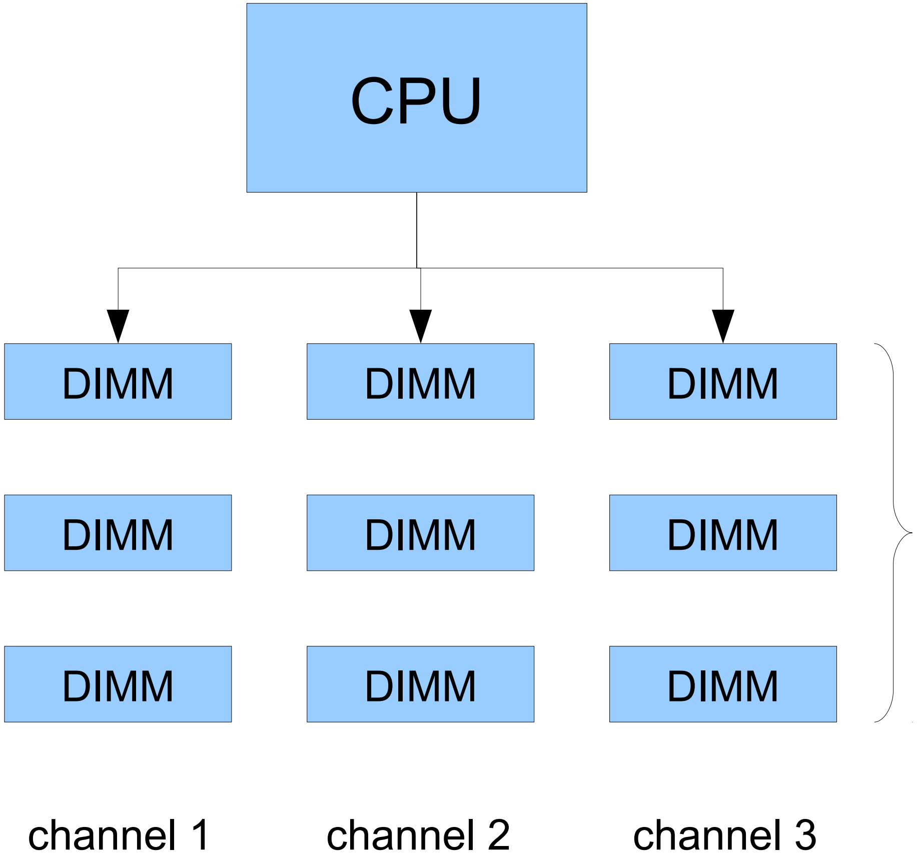
Sharing pinned system memory makes sysmem-to-sysmem copy unnecessary





And we get ~ 1 GB/s from camera. With 3 memcpy it is already on the border.

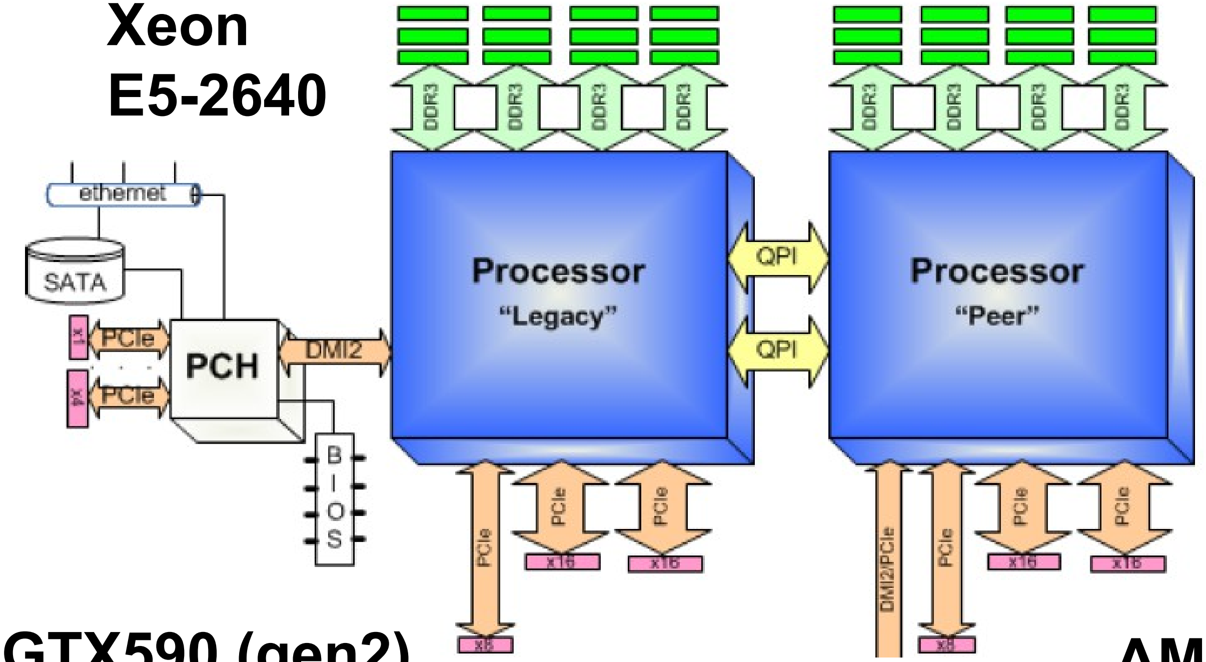
Memory: Space vs. Speed



DPC DIMM per Channel	**Xeon X5500**	**Xeon E5-2600**
1 DIMM	10.6 GB/s	12.8 GB/s
2 DIMMs	8.5 GB/s	12.8 GB/s
3 DIMMs	6.4 GB/s	8.5 GB/s

NUMA Architecture and Data Transfers

Xeon E5-2640



Bandwidth

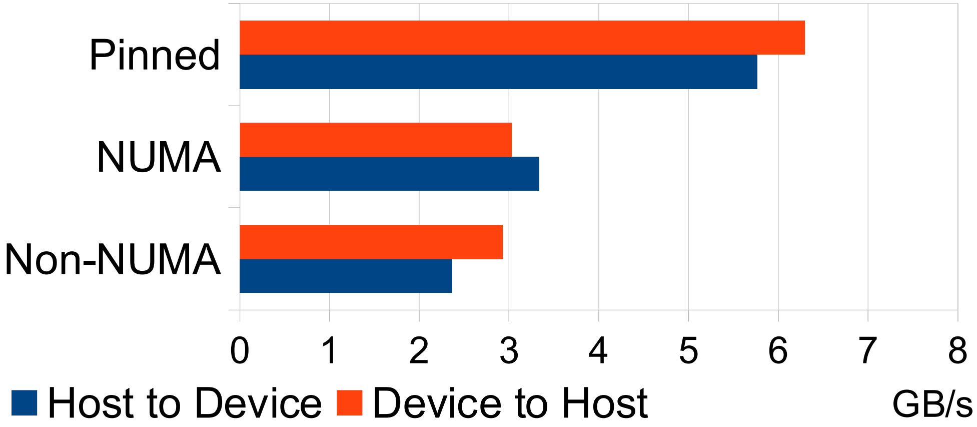
QPI: 14.4 GB/s

DDR3: 10.6 GB/s (PC1333)

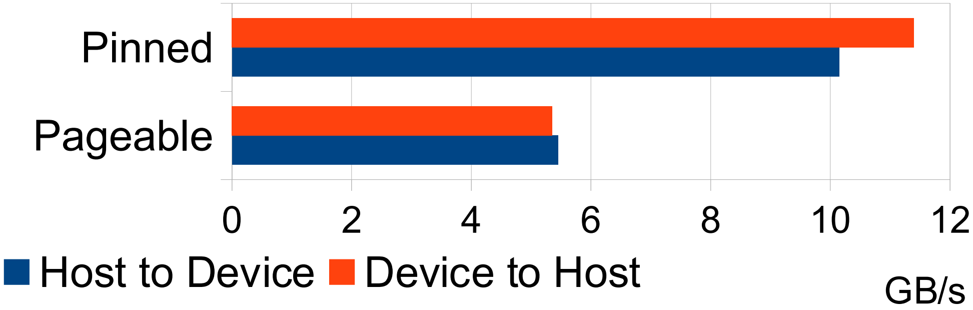
PCIe: 16 GB/s (gen3 x16)

QPI bus is even not enough to feed both GPU cards

GTX590 (gen2)



AMD HD7970 (gen3)



Pinned memory support is limited with OpenCL

NVIDIA does not support gen3 mode on X79 boards (workaround exits for Win, but not Linux)

Filtered Back Projection on GPU

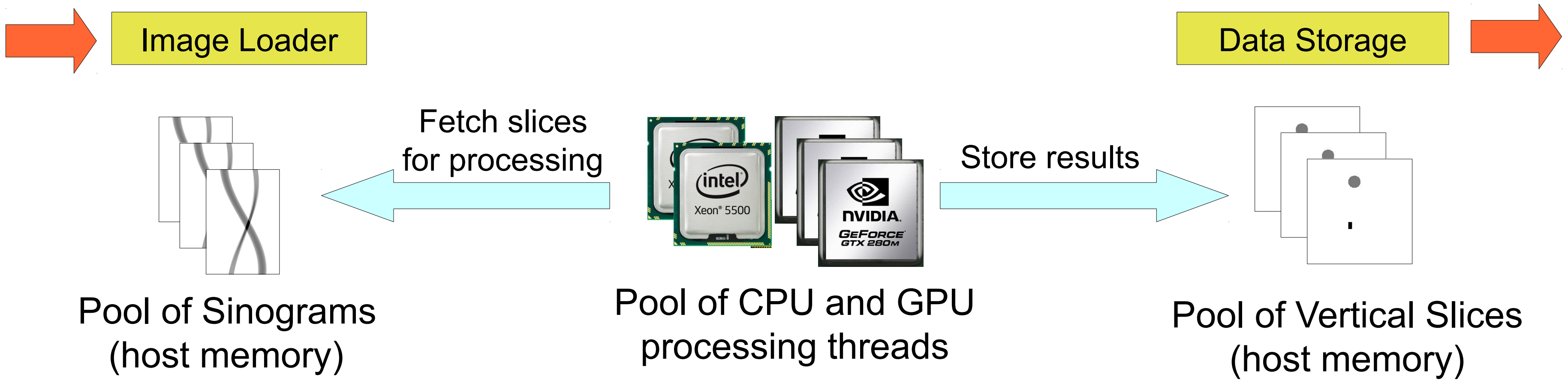
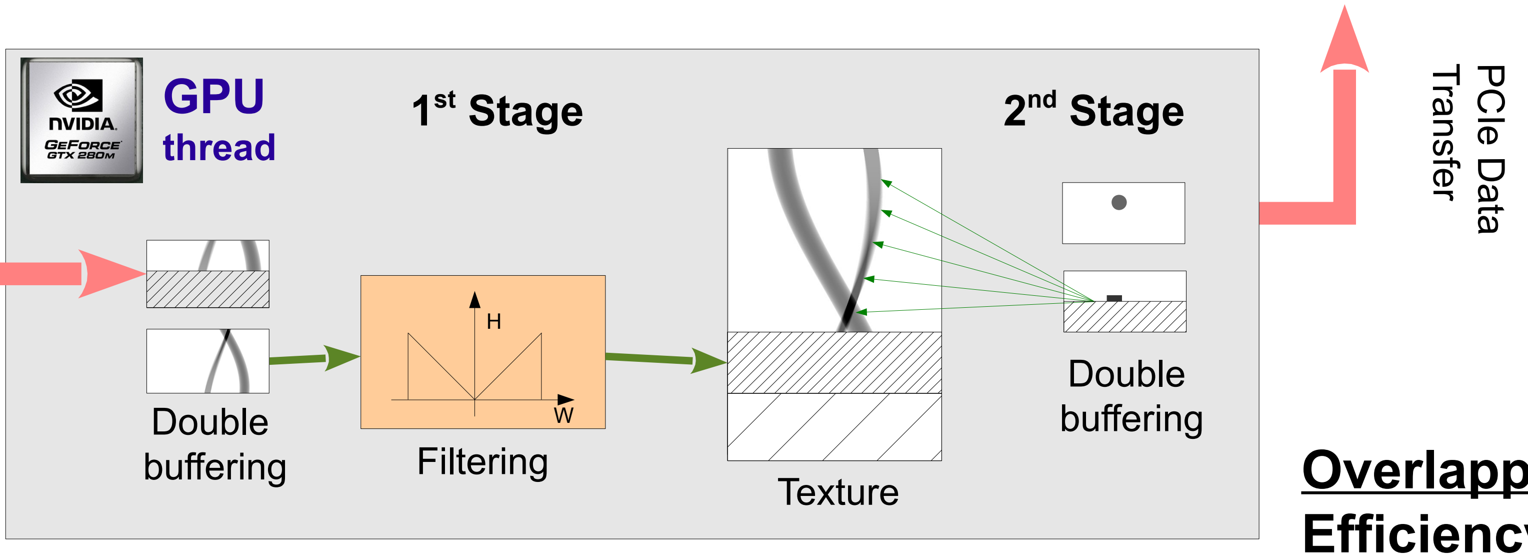
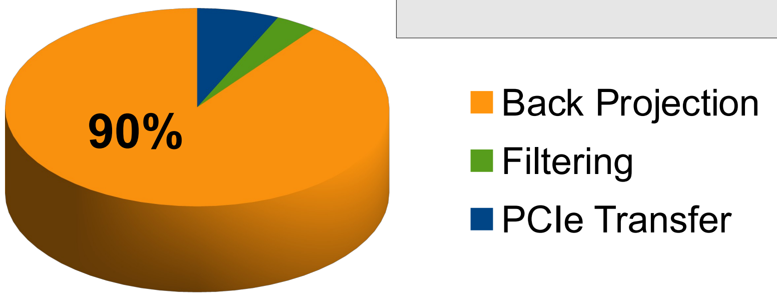


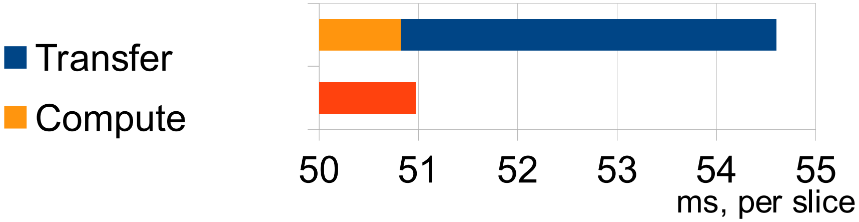
Image Loader
Data Storage
Fetch slices for processing
Store results
intel
Xeon® 5500
NVIDIA
GEFORCE GTX 280M
Pool of Sinograms (host memory)
Pool of CPU and GPU processing threads
Pool of Vertical Slices (host memory)



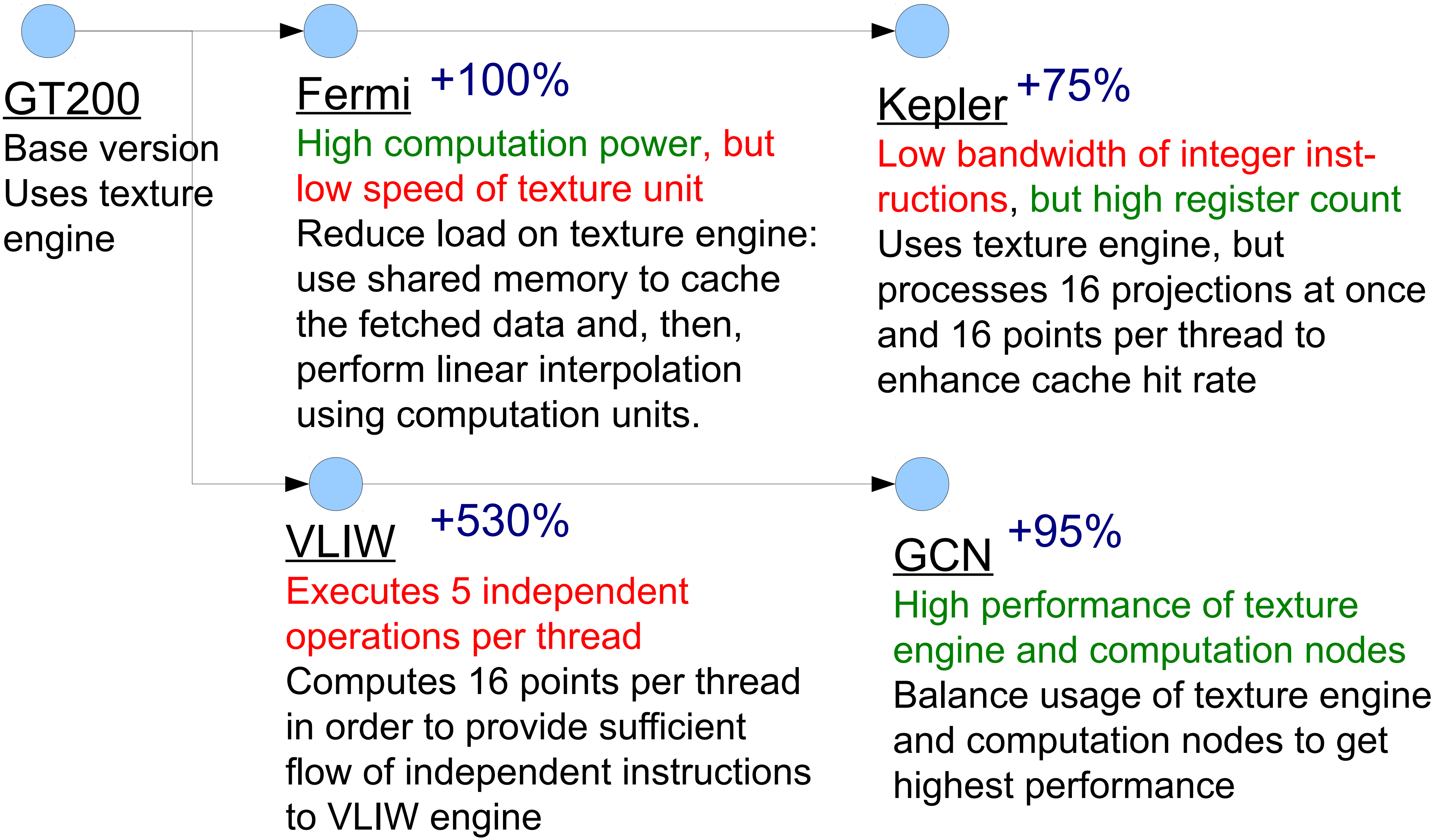
Ratio of operations



Overlapping Efficiency



Tuning for hardware architectures



GT200

Base version
Uses texture
engine

Fermi +100%

High computation power, but
low speed of texture unit
Reduce load on texture engine:
use shared memory to cache
the fetched data and, then,
perform linear interpolation
using computation units.

Kepler +75%

Low bandwidth of integer instructions, but high register count
Uses texture engine, but
processes 16 projections at once
and 16 points per thread to
enhance cache hit rate

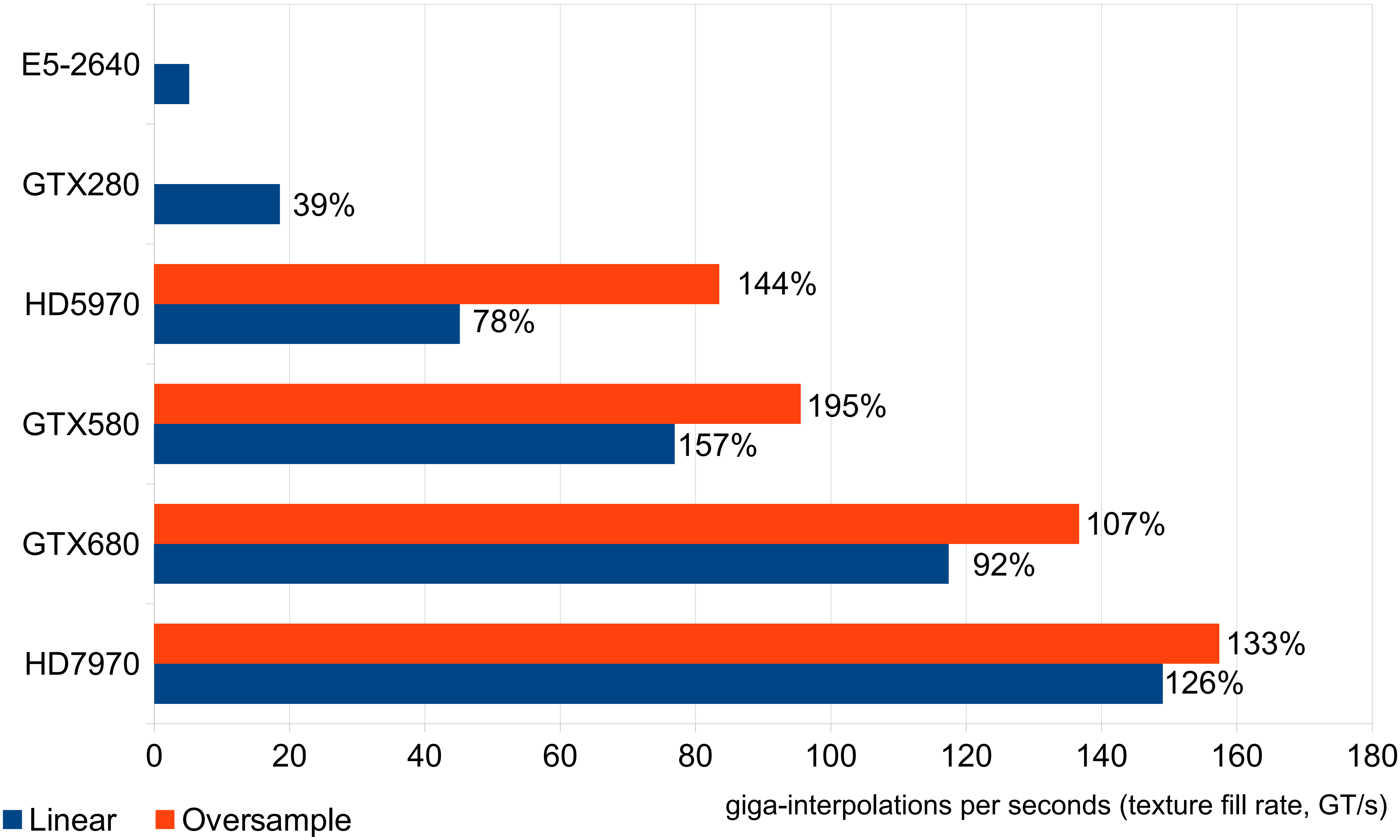
VLIW +530%

Executes 5 independent
operations per thread
Computes 16 points per thread
in order to provide sufficient
flow of independent instructions
to VLIW engine

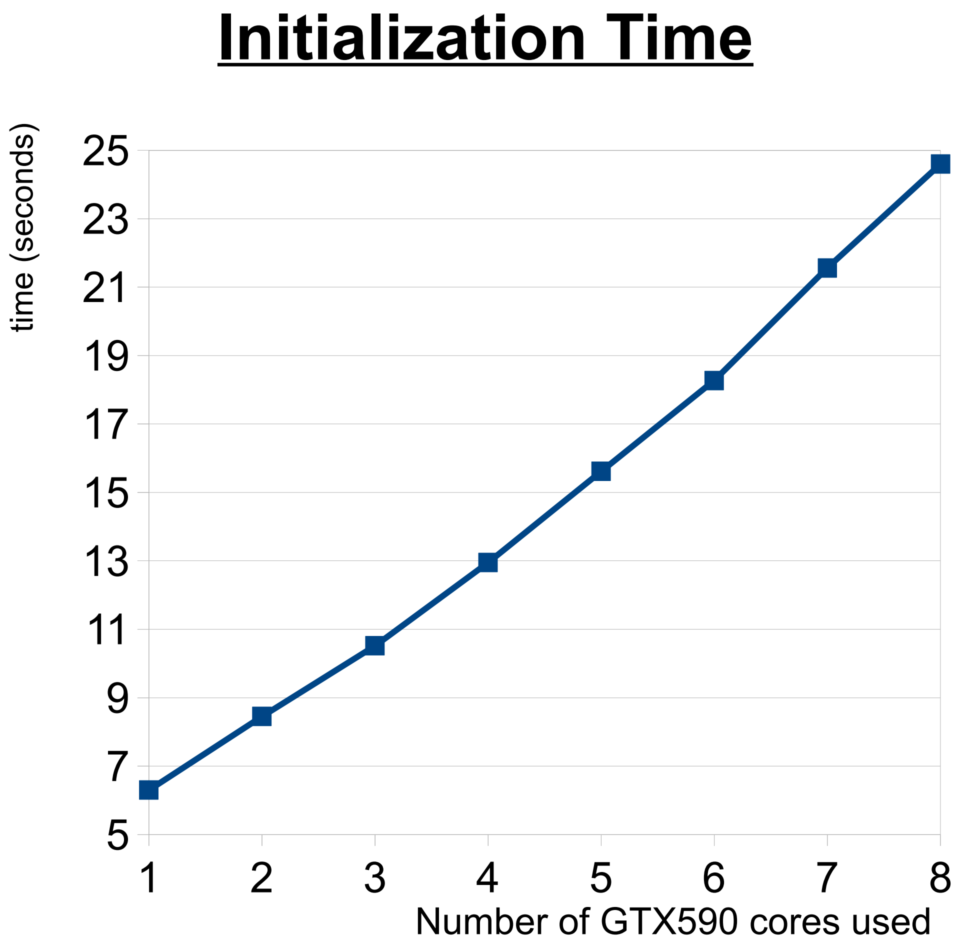
GCN +95%

High performance of texture
engine and computation nodes
Balance usage of texture engine
and computation nodes to get
highest performance

Back Projection: Evolution of GPU architectures



Adding more GPU devices



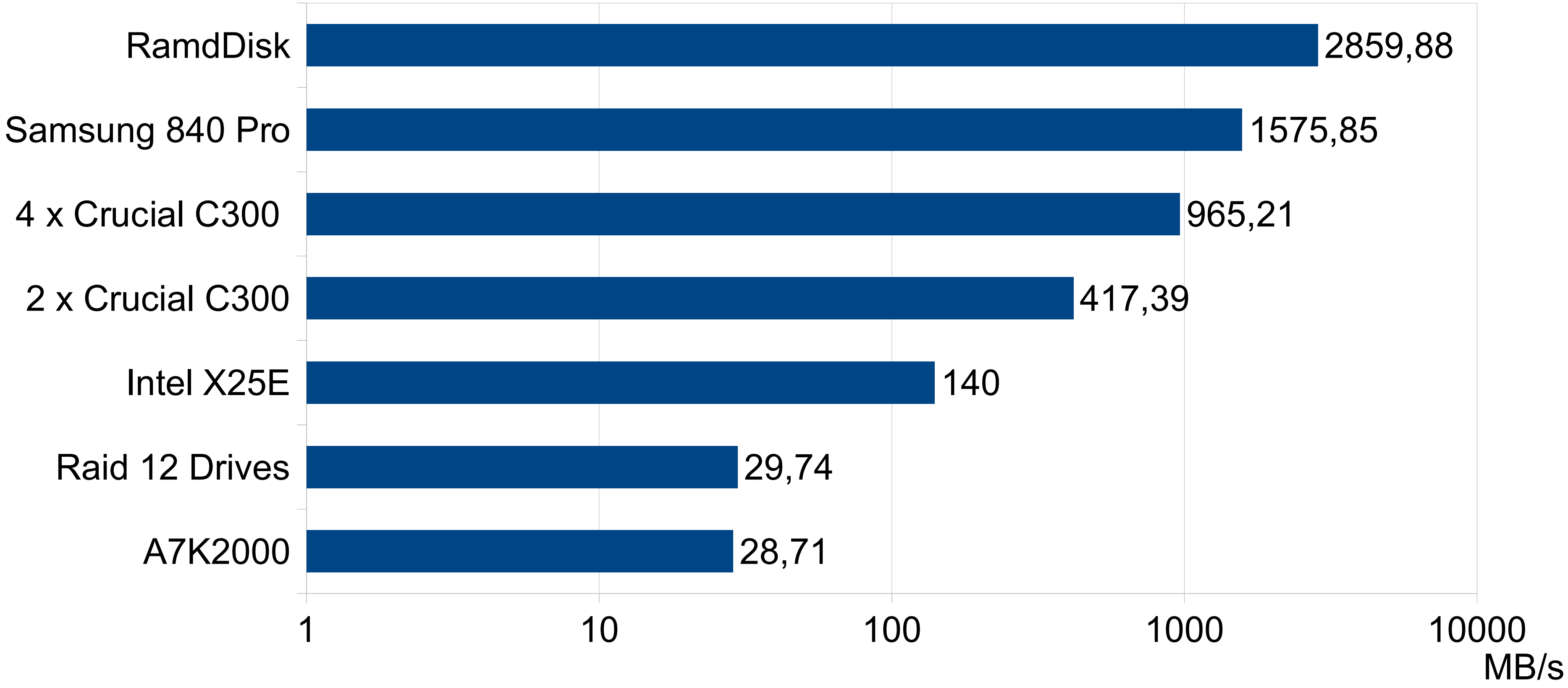
NVIDIA

- Maximum 8-9 GPU cores (not cards) per system. System will not turn on otherwise
- Lan Option ROM have to be turned off in the BIOS
- The PCIe slots, where storage adapters inserted, have to be disabled
- ASTRA Lab reported to run 13 GPU cores with modified BIOS
- To run more than 5 GPUs, NVIDIA driver have to be force to use MSI interrupts. Crashes will occur otherwise

AMD

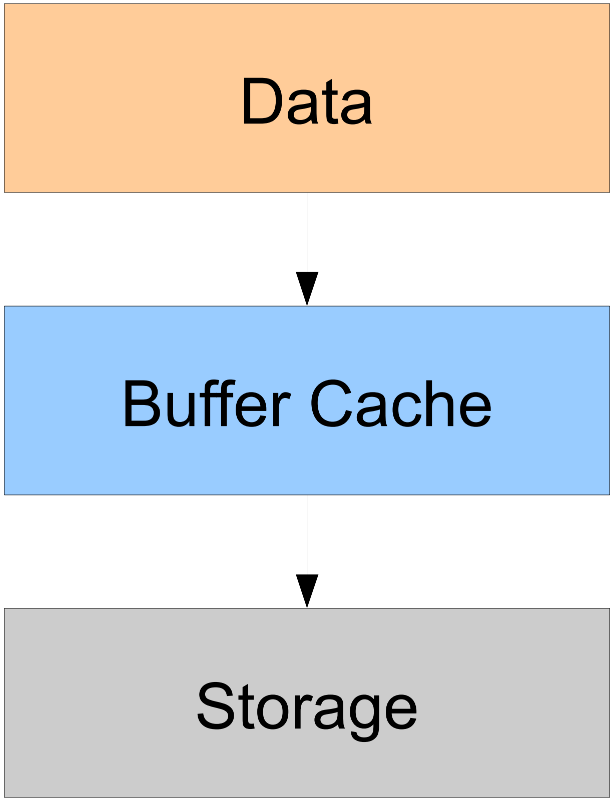
- 4 GPU cards (single core) working fine, no configuration modifications required
- Dual-core card are working in a single-core mode only

Handling large data sets

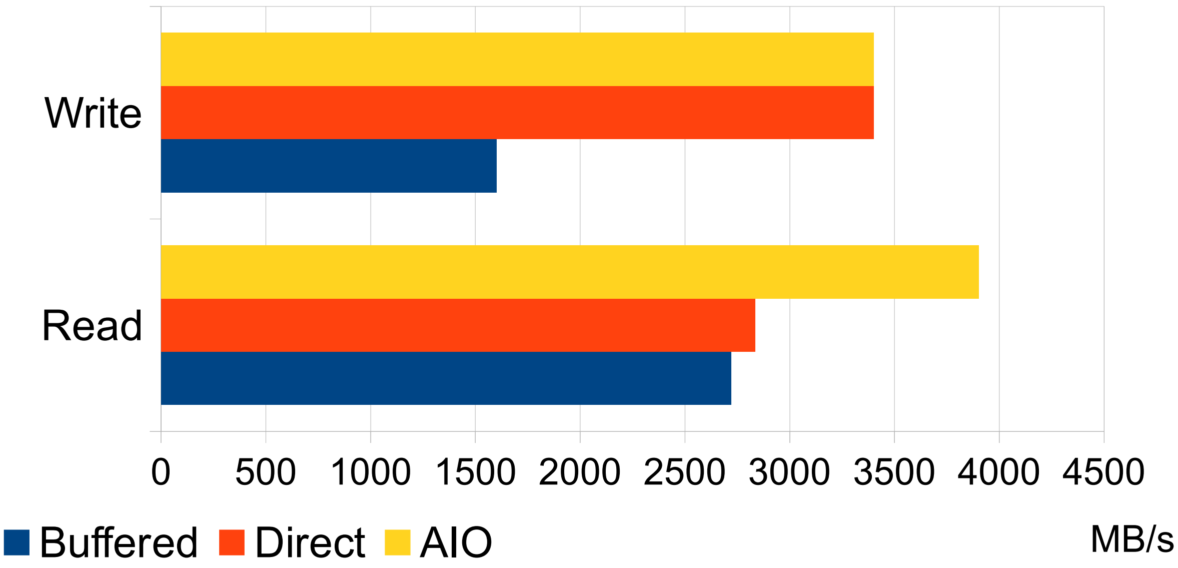


Using SSD drives may significantly increase random access performance to the data sets which are not fitting in memory completely. The big arrays of magnetic hard drives will not help unless multiple readers involved.

Streaming data: file system caches



Default data flow in Linux

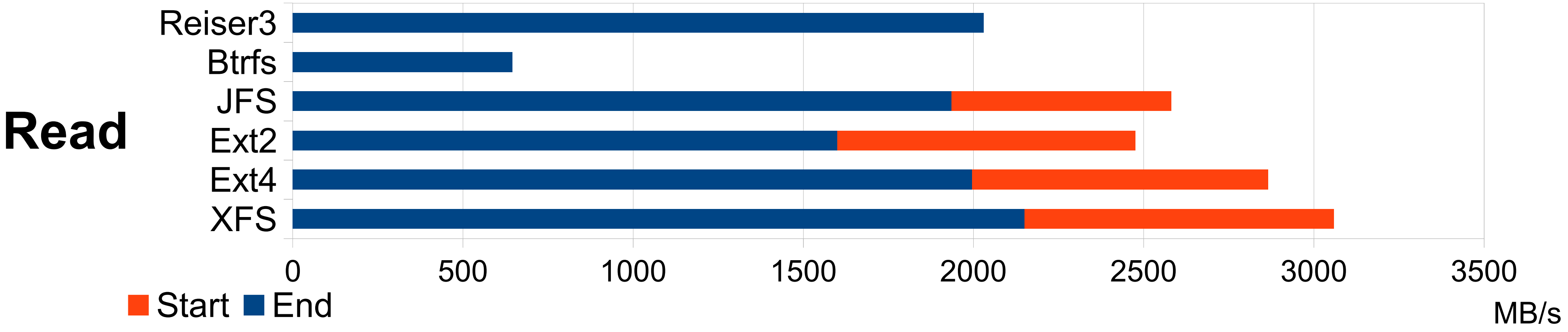


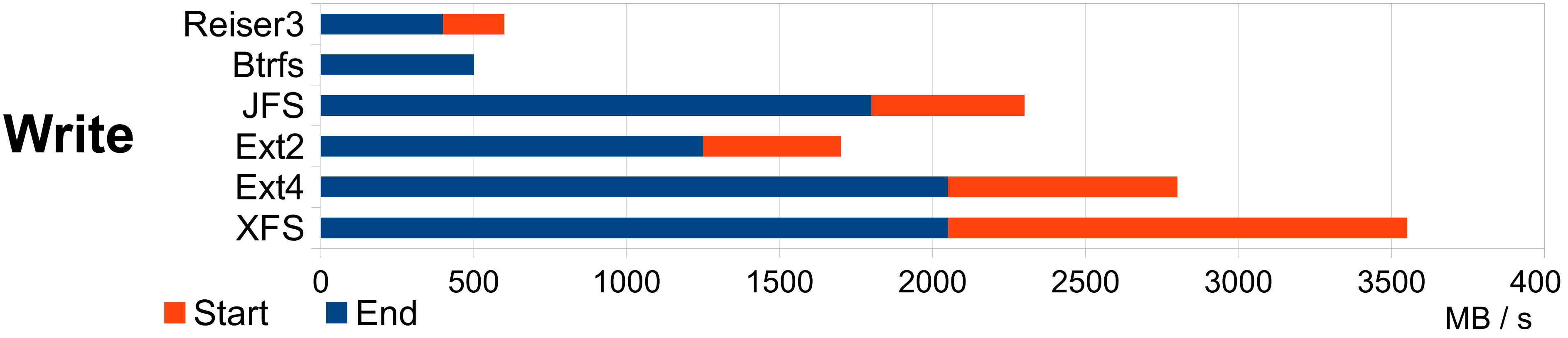
- Buffer cache significantly limits maximal write performance
- Kernel AIO may be used to program IO scheduler to issue read requests without delays

Optimizing I/O for maximum streaming performance using a single data source/receiver

Data Streaming

OpenSuSE 12.1 / Kernel 3.3.1
32 disks per 2 raid controllers, raid 60



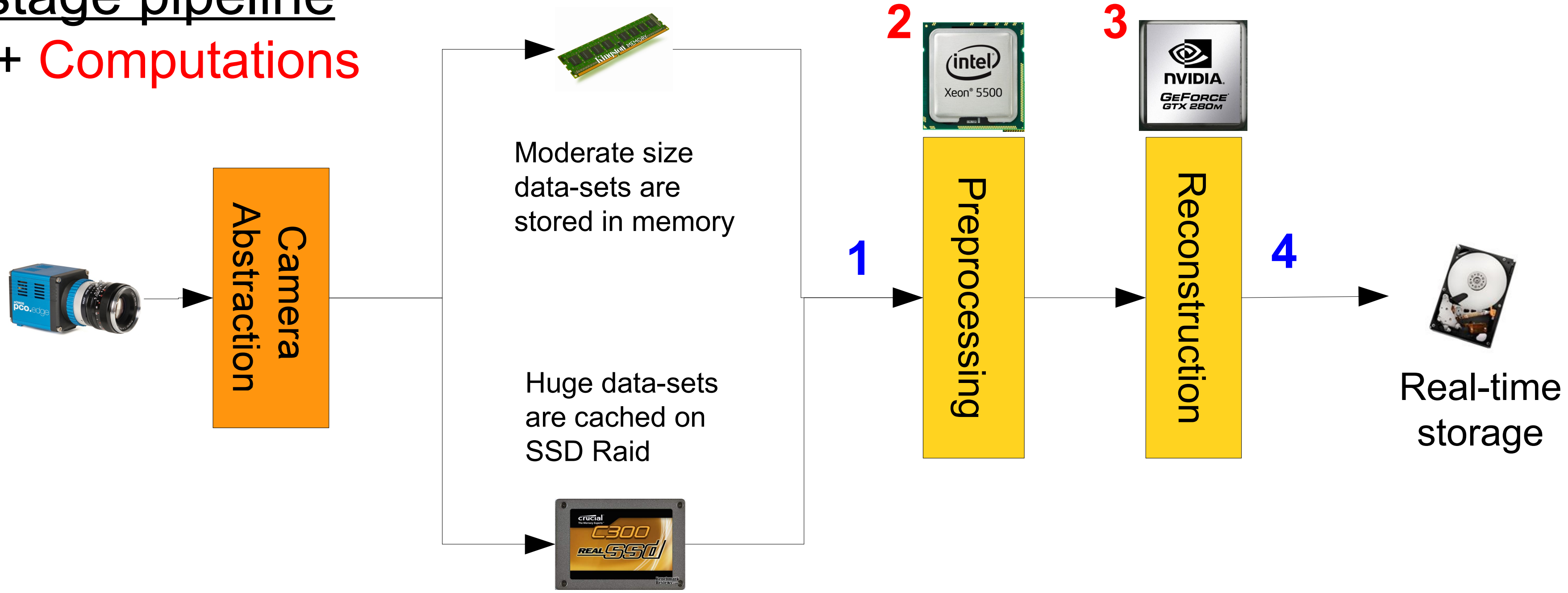


- Used file system matter. And it should be adapted to raid configuration (strip, read-ahead)
- Unless really big number of disks used, the start of partition will be faster than the end)
- fallocate may significantly improve performance (allocation unit may be increased during FS creation/mount, XFS supports allocation sizes up to 1GB)
- Ext4 does not support partitions more than 16TB yet
- Real-time feature of XFS is unstable, data is loss is likely

Processing Pipeline

4 stage pipeline

I/O + Computations


Moderate size data-sets are stored in memory
Huge data-sets are cached on SSD Raid
Camera Abstraction
1
2
Preprocessing
3
Reconstruction
4
Real-time storage

1. Reading data from fast SSD Raid-0 (random reads are effective)
2. Scheduling and preprocessing using SIMD instructions of x86 CPUs
3. Reconstructing on GPUs
4. Storing to Raid on magnetic disks (sequential writes are effective)

Building a server

- **Too much external hardware is required**
  - High speed network
  - Storage system (and SSD cache separately preferably)
  - High speed Frame Grabber for Camera
  - Normally 4-6 high speed PCIe slots per server
  - Space for 1-2 GPUs only
- **System cooling is complicated**
  - both GPUs, HDDs, and SSDs produce a lot of heat
- **Extensibility**
  - There is no space to add more storage / computing power

UFO Computing Infrastructure

Karlsruhe Institute of Technology

Camera

PCO.edge
PCO.dimax
PCO.4000

CameraLink
850MB/s

Ethernet
10 Gb/s

Storage

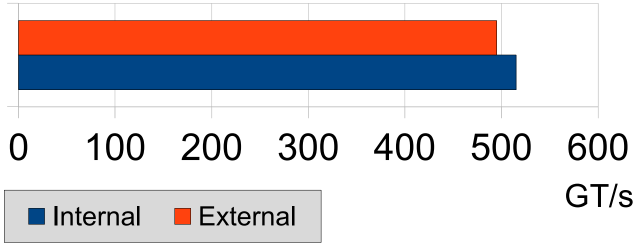
LSDF
Large Scale Data Facility

External PCIe x16 (8 GB/s)

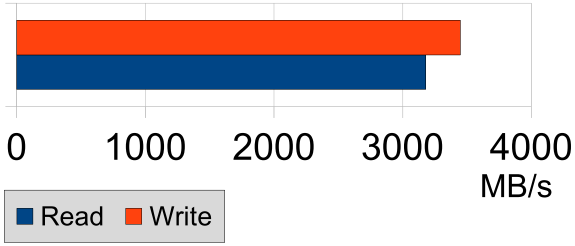
SFF8088 (2.4 GB/s)

<u>**SuperMicro 7046GT-TRF**</u> (Dual Intel 5520 Chipset)
CPU: 2 x Xeon X5650 (total 12 cores at 2.66 Ghz)
GPUs: 4 x GTX590 External
Memory: 96 GB / 12 DDR3 slots (192GB max)
Network: Intel 82598EB (10 Gb/s)
Camera Link Frame Grabber (850 MB/s)
Storage: Areca ARC-1880-ix-12 SAS Raid
16 x Hitachi A7K200 (Raid6)
8 x Samsung 840 Pro 510 (Raid0)

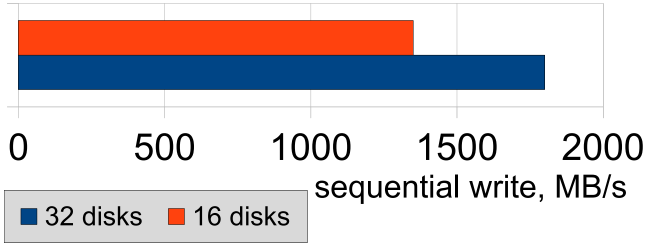
External GPU Box



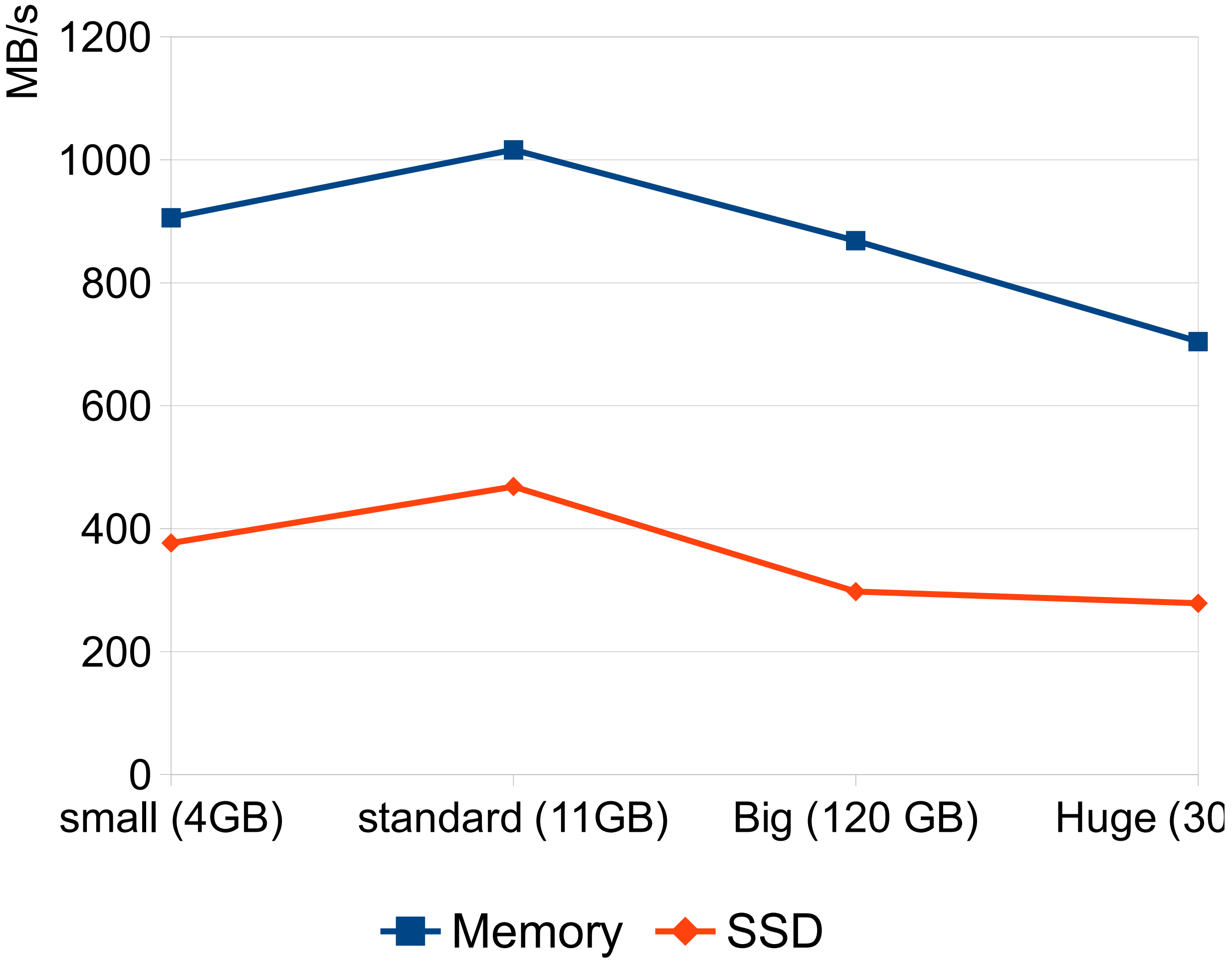
SSD Raid

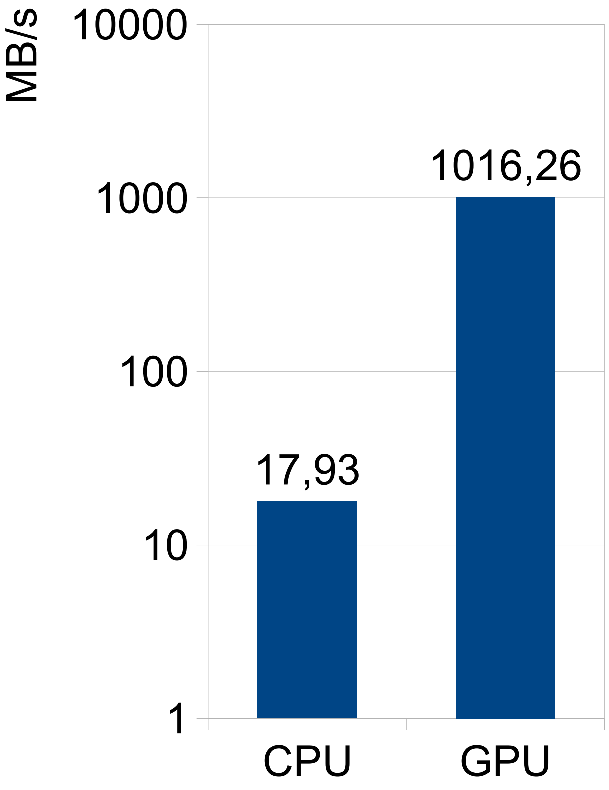


SAS Attached Storage



Filtered Back Projection Performance



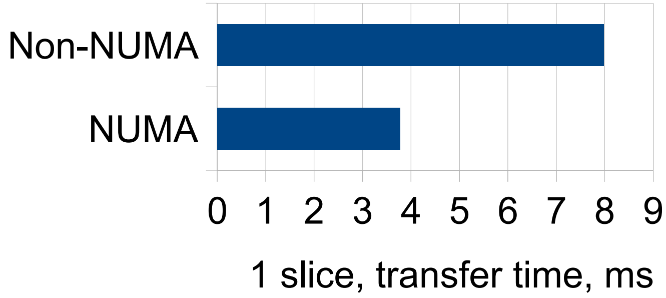


11 GB data set

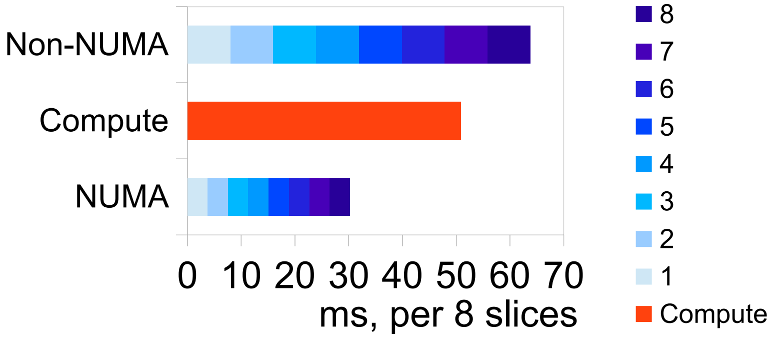
PCIe Extension Box

1 x PCIe x16 2.0
4 x GTX590
8 GPU cores

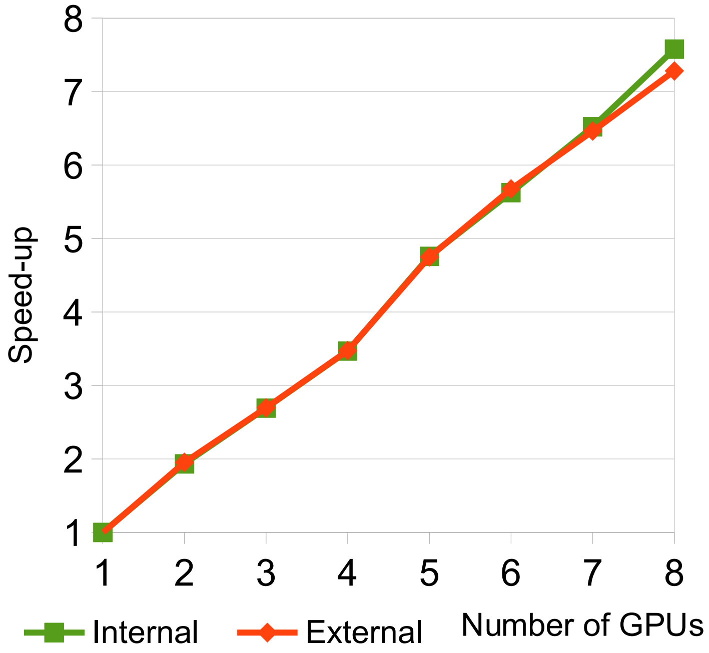
External GPU Enclosure
by One Stop Systems

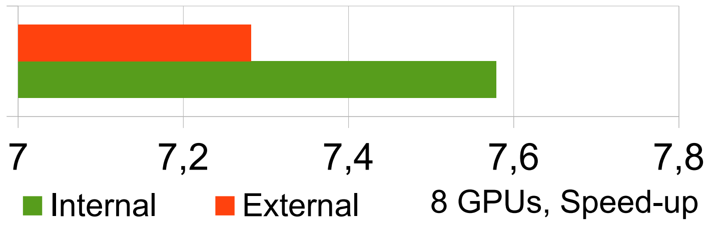


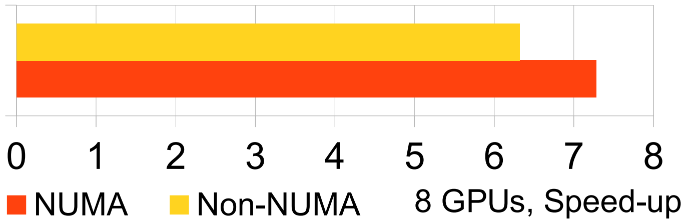
With external box configuration



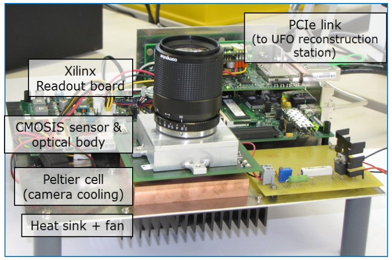
Scalability



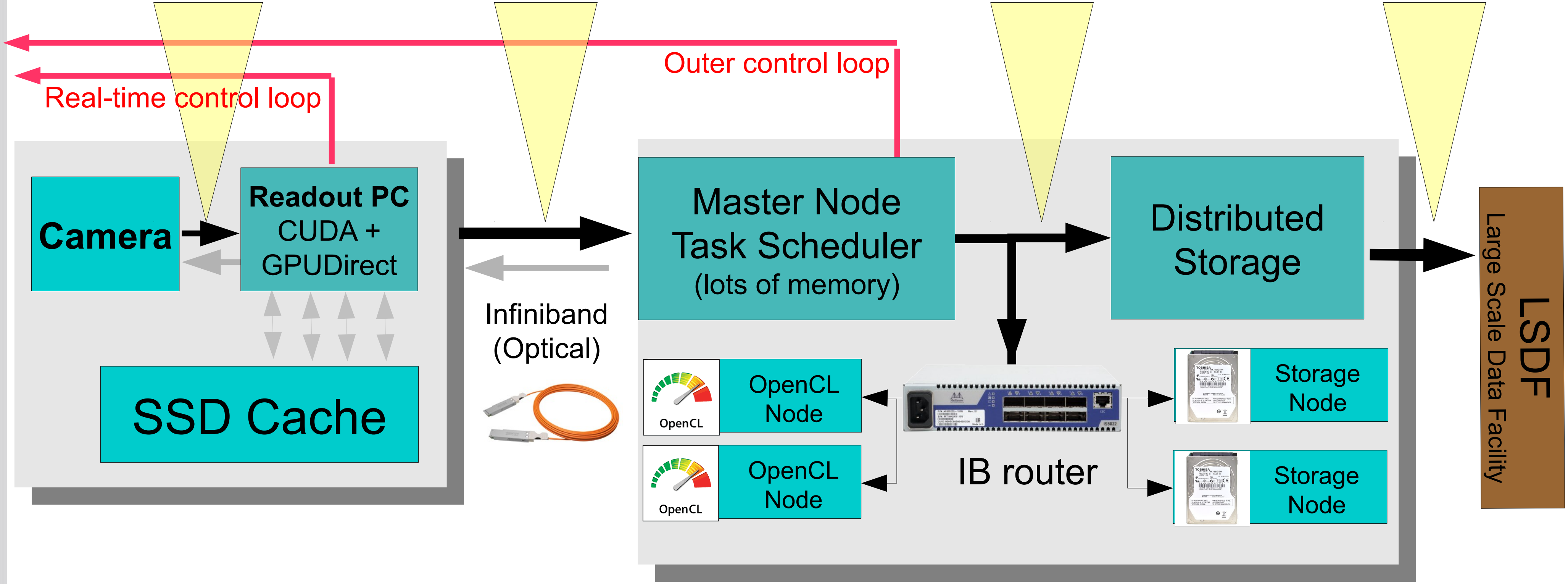




Scaling up to Cluster

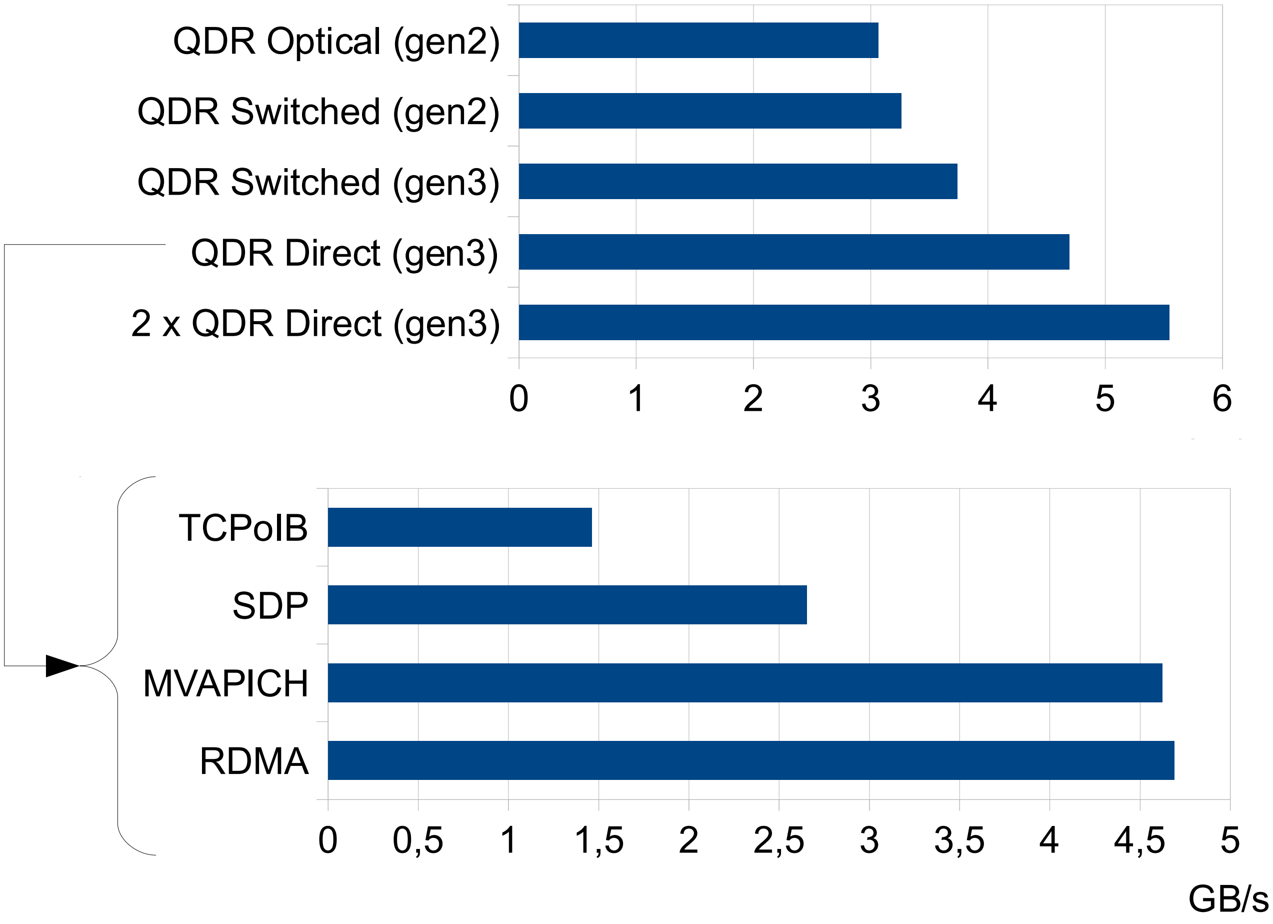


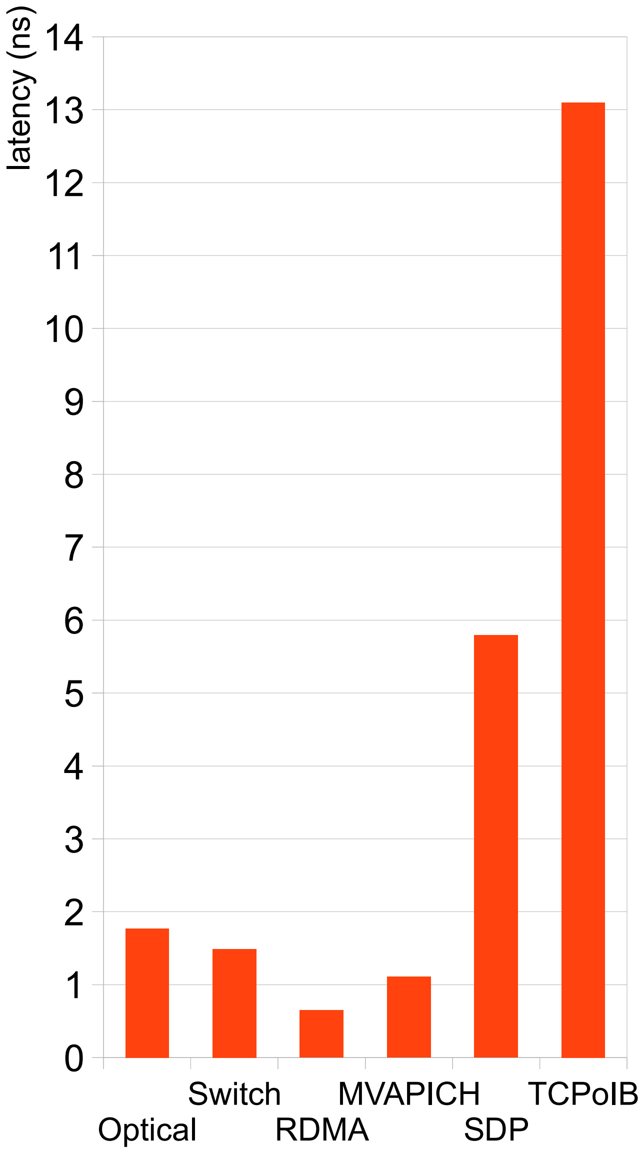
Transfer rates up to 8 GB/s 4GB/s <2 GB/s 0,25 GB/s



Infiniband: Connection and Protocols

Mellanox ConnectX 3 VPI





SDP is obsolete by OpenFabric alliance,
but we have patches for latest kernels.

Storage Protocols

Network FS

NFS
Samba
SSHFS

Slow

Cluster FS

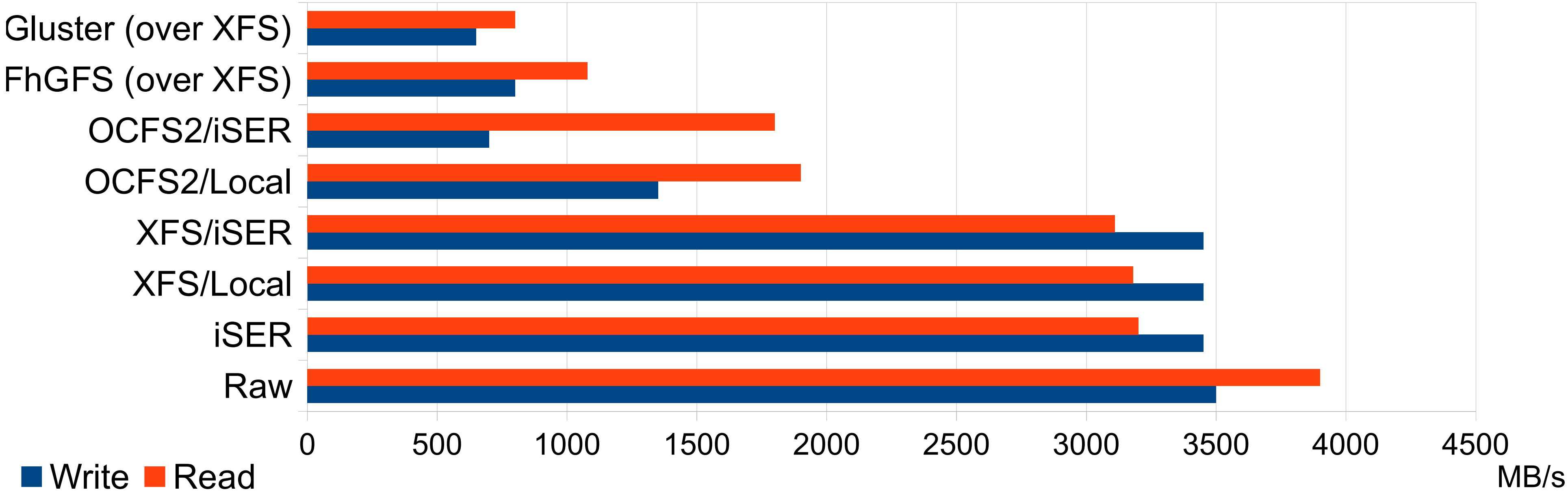
Lustre (patched kernel)
Gluster
FhGFS (close-sourced)

Network Devices

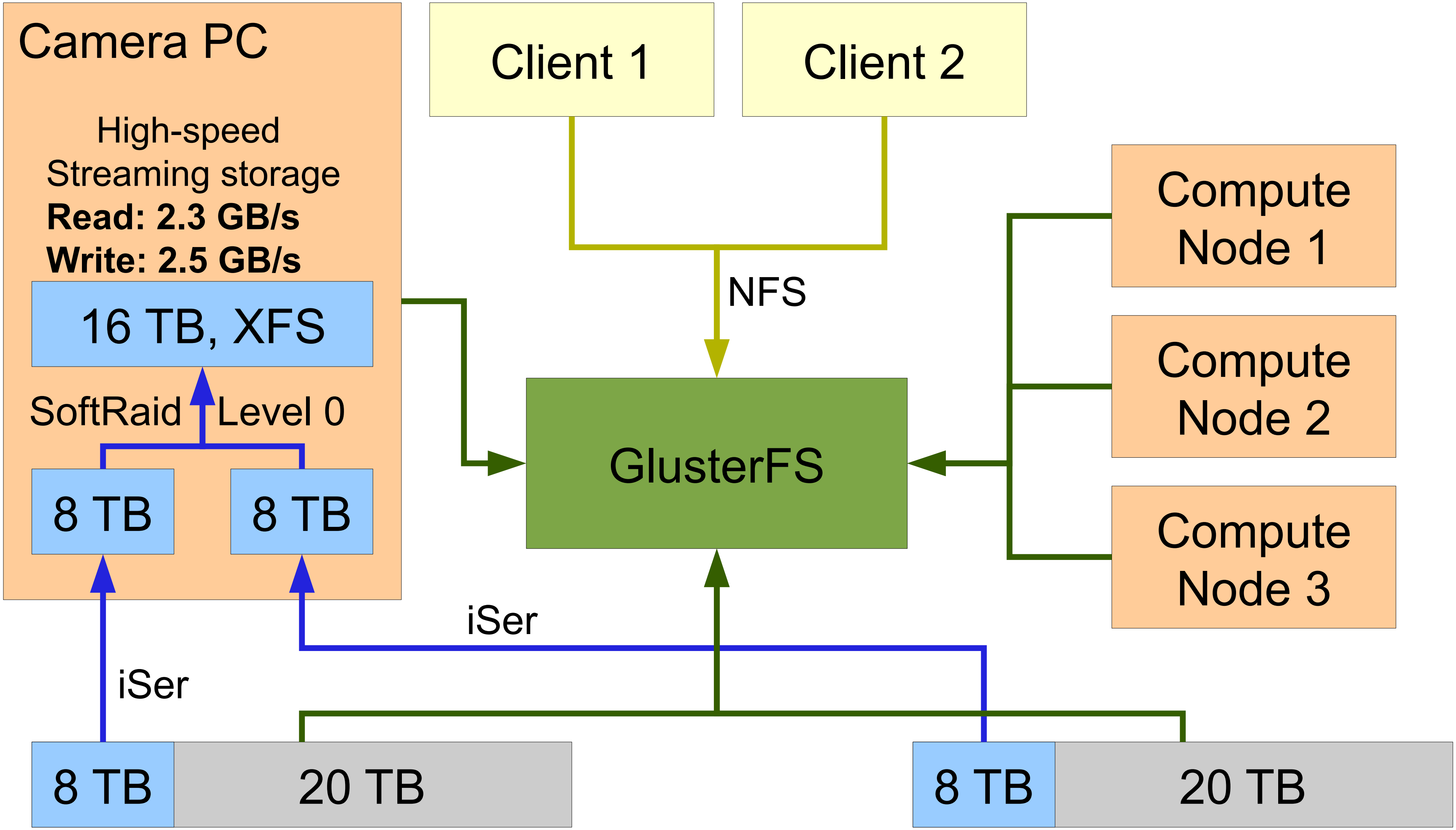
ISCSI (slow)
iSER

Slow if few nodes

OCFS2



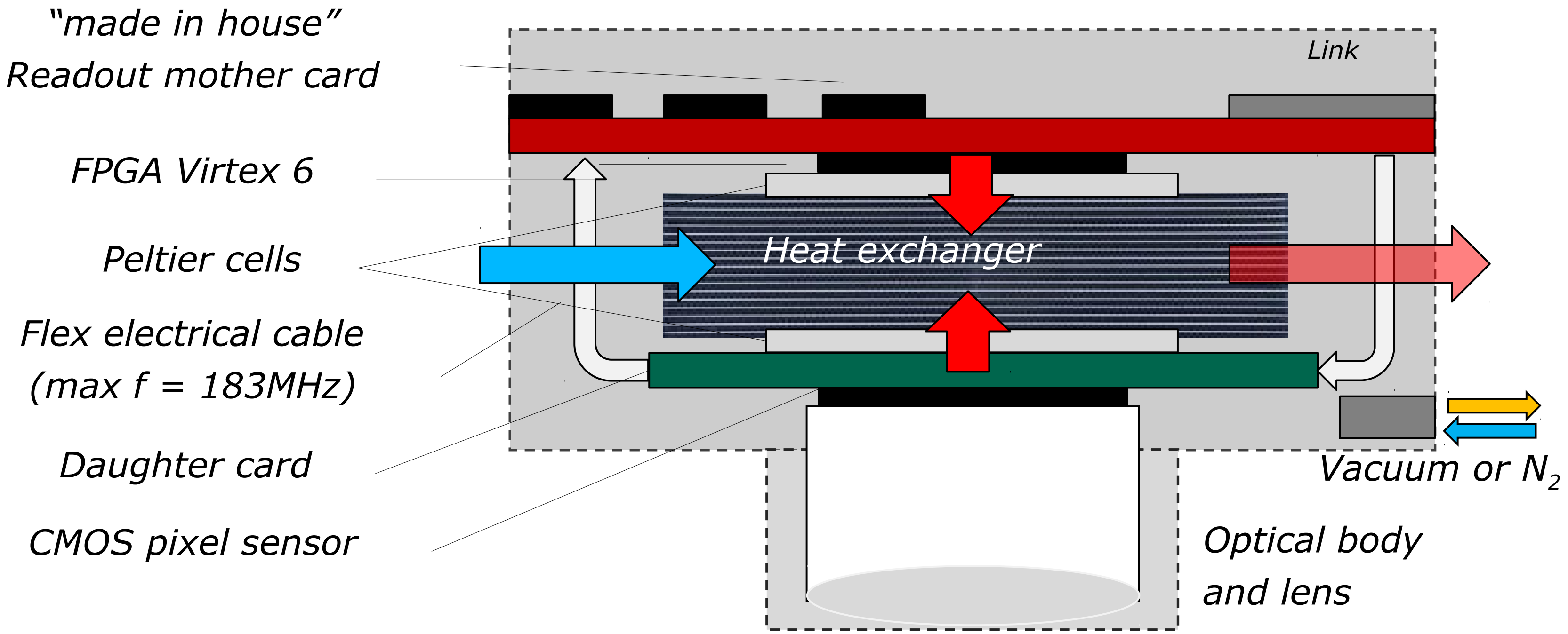
UFO Storage Subsystem



Storage Node 1
Raid6: 16 Hitachi 7K300, 28TB

Storage Node 2
Raid6: 16 Hitachi 7K300, 28TB

High-speed Programmable Camera



"made in house"
Readout mother card

FPGA Virtex 6

Peltier cells

Flex electrical cable
(max f = 183MHz)

Daughter card

CMOS pixel sensor

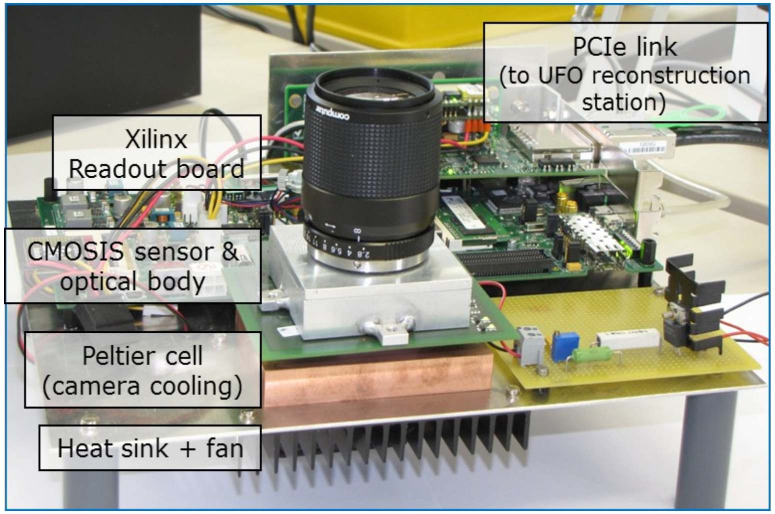
Link

Heat exchanger

Vacuum or N_2

Optical body
and lens

- ***High speed CMOS sensor***
- ***1Mpix, 5000 fps, 10 bits***
- ***Self-trigger & Data compression***
- ***On-line elaborations and control***
- ***Full Programmability***
- ***Direct connection to Infiniband-cluster***



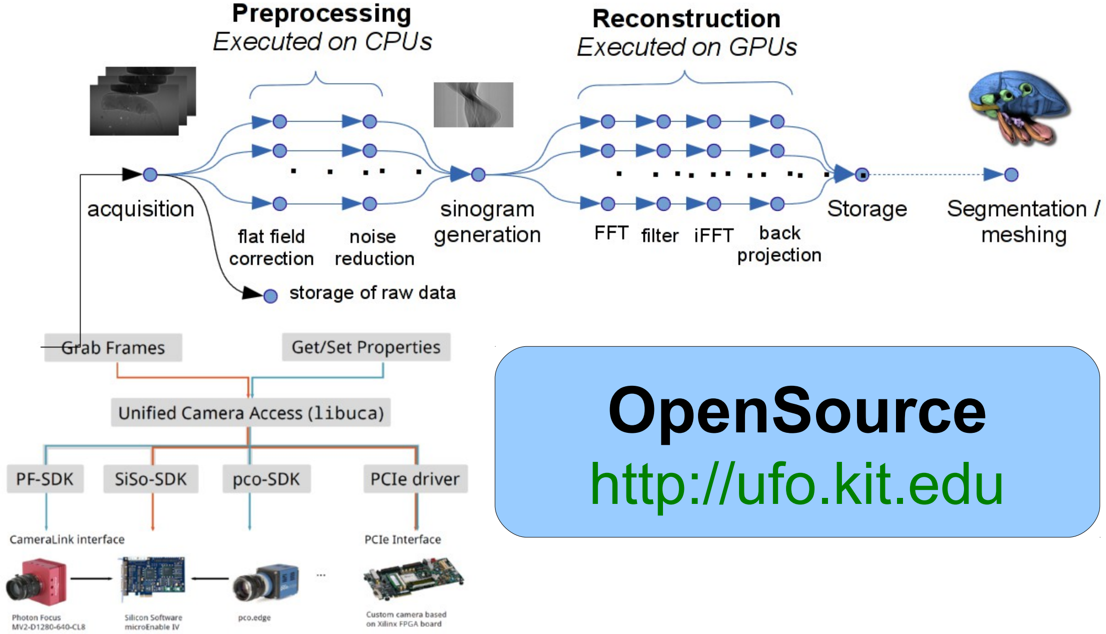
First Prototype

Summary

- We are in the age of parallel architectures
- Getting good performance is rather easy, getting ultimate-performance managing multi-gigabyte streams of data is complicated and needs care on multiple levels. The hardware should be carefully selected according to the planned tasks and data rates. The software should be tunned to the selected hardware.
- Streams about 500 MB/s may be processed with a single reconstruction station, cluster is required to handle more data in near real-time.
- Hybrid CUDA/OpenCL system is probably the best approach.
- UFO Parallel Processing Framework is provided to help you to come along some of these difficulties and will be presented in next talk.

Features

- Easy Algorithm Exchange
- Camera Abstraction
- Pipelined Processing
- Glib/GObject, scripting language support with introspection
- OpenCL + automated management of OpenCL buffers



OpenSource

http://ufo.kit.edu