High Speed Tomography at KIT

Agenda

PyHST improvements at KIT

Evaluation of GPU platforms

+ OpenCL Performance

UFO Project

+ Handling the I/O

Goal: Process up to 30GB of image data in a minute (*1 Tflop/s required*)

In collaboration with ESRF:

European Synchrotron Radiation Facility

Polygone Scientifique Louis Néel, 6 rue Jules Horowitz, 38000 GRENOBLE

S. Chilingaryan, T. Farago, A. Kopmann, M. Vogelgesang

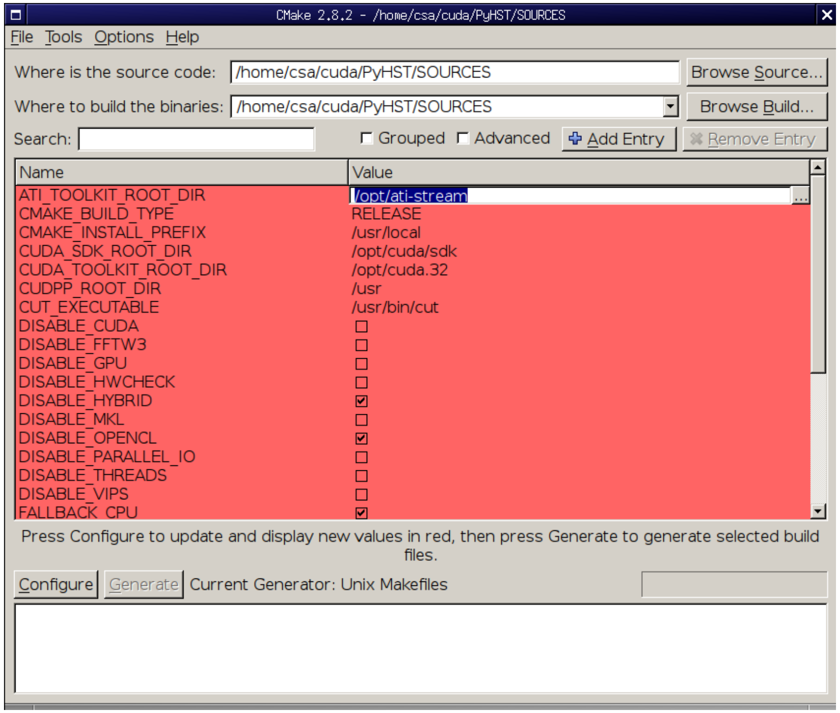
PyHST at KIT

- **Architecture**
  - CMake based build system
  - New modular architecture
  - Consistent logging with Python-logging
- **Performance**
  - Sinogram filtering on GPU
  - Faster data transfers between GPU and main memory
  - Scheduler supporting multiple CPU/GPU cores
  - I/O Optimizations: Faster EDF, Image preloading
- **New Features**
  - Support TIFF and other image formats
  - Reconstruction using OpenCL
  - Simple Java-based GUI

PyHST: New Build System

PyHST/KIT have significantly more required and optional dependencies compared to ESRF version.

- **Detect Dependencies**
  - **Required: Glib, Python with Numpy, Imaging, Logging modules**
  - **Math: FFTW and Intel MKL**
  - **Image Support: VIPS**
- **Find parallel processing SDKs**
  - **NVIDIA CUDA + CUDA SDK**
  - **ATI Stream**
  - **OpenCL Support + oclfft library**
  - **Check Hardware**
- **Set Execution Modes**
  - **GPU/CPU/Hybrid/Fallback**
  - **Threading modes**
  - **Benchmarking Modes**



Where is the source code: /home/csa/cuda/PyHST/SOURCES

Where to build the binaries: /home/csa/cuda/PyHST/SOURCES

Search:

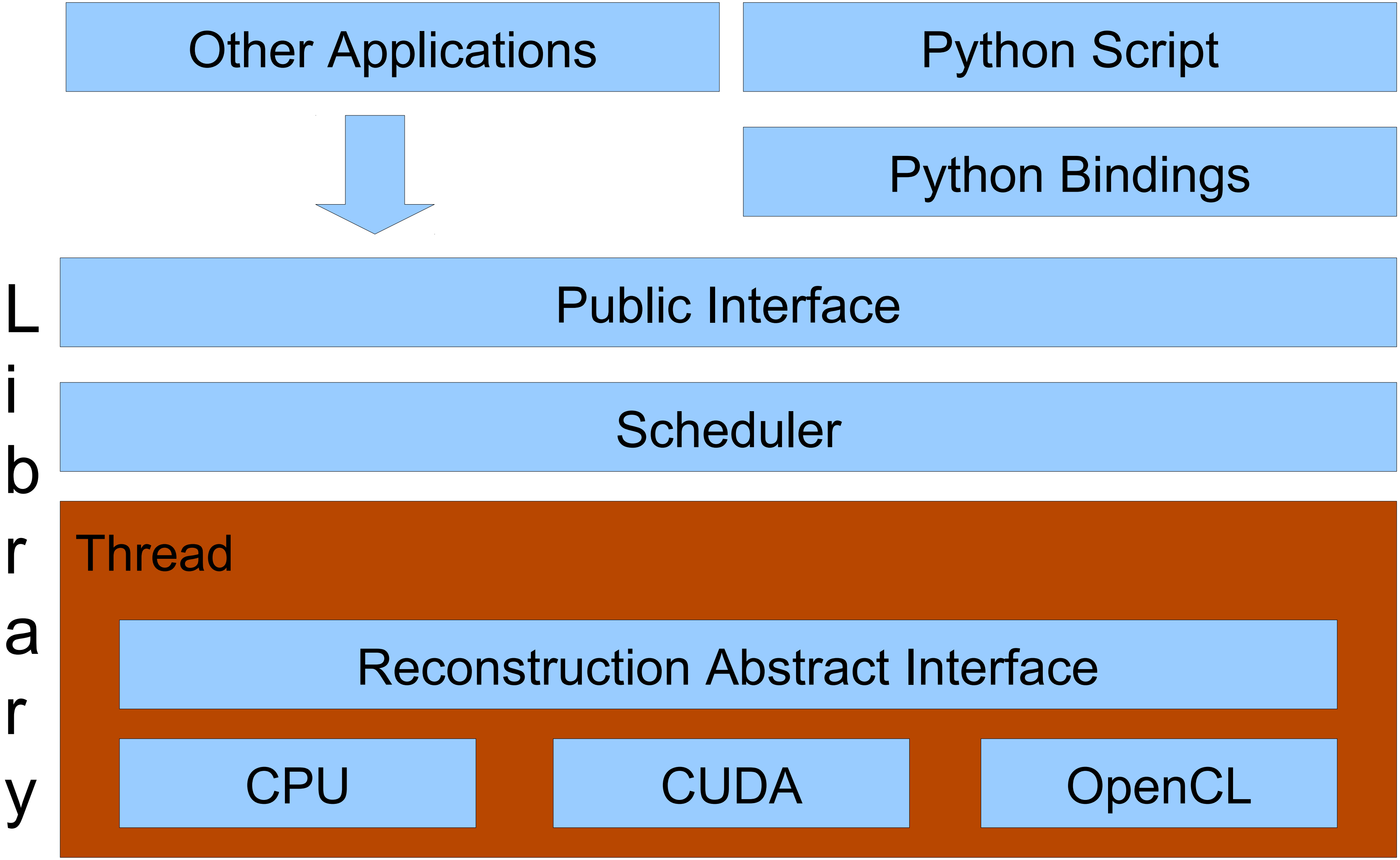
□ Grouped □ Advanced

Name	Value
ATI_TOOLKIT_ROOT_DIR	/opt/ati-stream
CMAKE_BUILD_TYPE	RELEASE
CMAKE_INSTALL_PREFIX	/usr/local
CUDA_SDK_ROOT_DIR	/opt/cuda/sdk
CUDA_TOOLKIT_ROOT_DIR	/opt/cuda.32
CUDPP_ROOT_DIR	/usr
CUT_EXECUTABLE	/usr/bin/cut
DISABLE_CUDA	□
DISABLE_FFTW3	□
DISABLE_GPU	□
DISABLE_HWCHECK	□
DISABLE_HYBRID	☑
DISABLE_MKL	□
DISABLE_OPENCL	☑
DISABLE_PARALLEL_IO	□
DISABLE_THREADS	□
DISABLE_VIPS	□
FALLBACK_CPU	☑

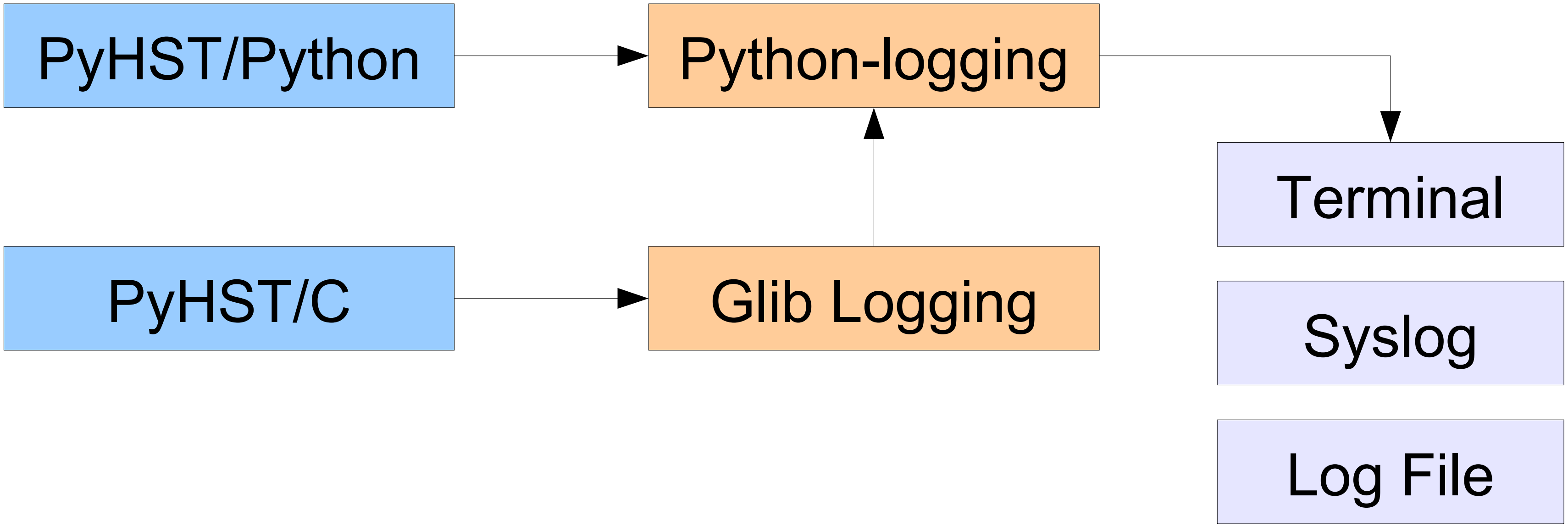
Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Unix Makefiles

PyHST: New Architecture



PyHST: Logging and Benchmarking



PyHST/Python

Python-logging

PyHST/C

Glib Logging

Terminal

Syslog

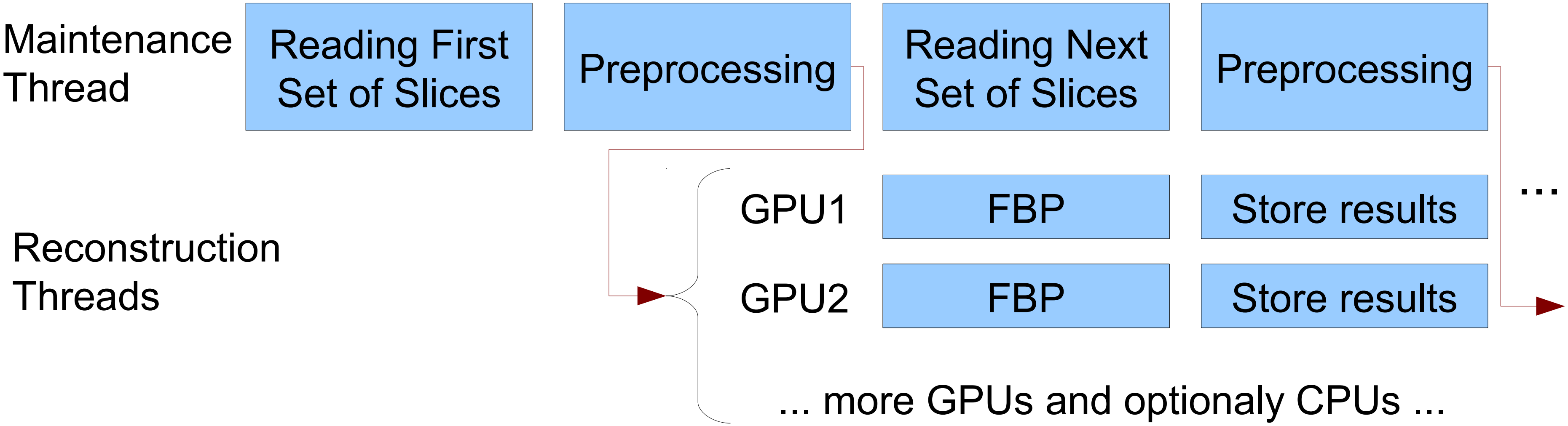
Log File

<u>Builds for Performance Evaluation</u>

- **Measure Timings**
- **I/O Benchmarking Mode**
- **Reconstruction Benchmarking Mode**

I/O Time
Preprocessing
Sinogram Generation
PCIe transfer
Fourier Filtering
Back Projection

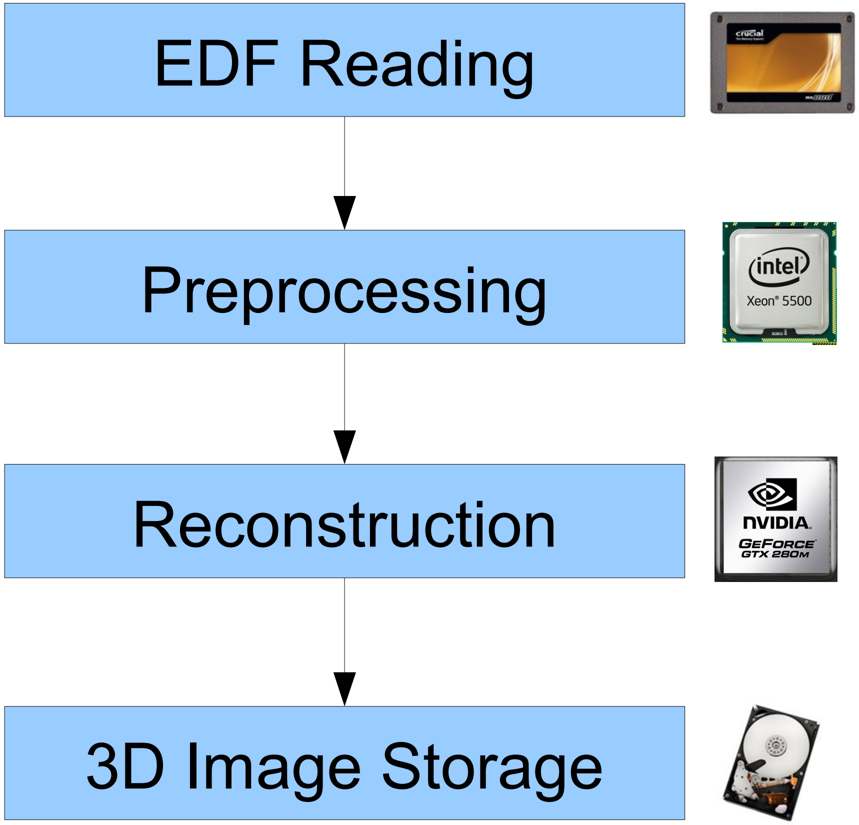
PyHST: Threading Model



Features

- Use all CPU/GPU cores available in the system
- Use CPUs and GPUs for reconstruction in parallel
- Fallback to CPU-reconstruction if GPU is not available
- Computations and I/O is performed in parallel

Pipeline in next version of PyHST


EDF Reading

Preprocessing

Reconstruction

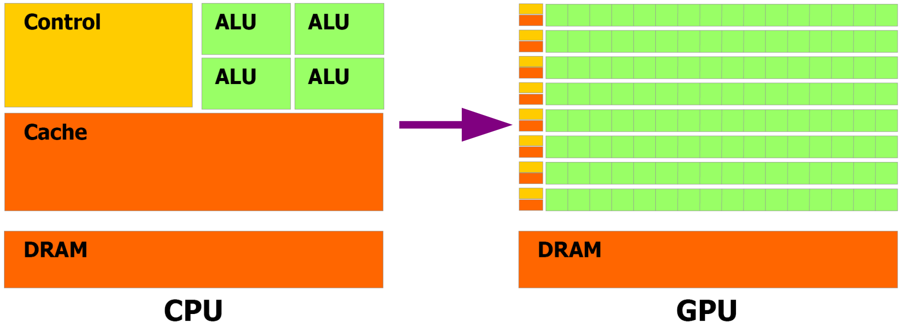
3D Image Storage

4 stage pipeline

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

Optimal programming of GPU devices

- **Memory (GPU lacks cache!)**
  - Allocate all device memory during initialization
  - Pad data to multiples of block dimensions
  - Optimally access memory (**increase of speed up to 10 times**)



- **Data Transfer (Slow interface between GPU and memory ~ 2 - 4 GB/s)**
  - Reduce amount of data transfers between GPU and host (**extreme**)
  - Use pinned memory for transfers if possible (**up to 2 times**)
  - Interleave data transfers with computations
- **Arithmetics (GPU is optimized only for a few specific instructions)**
  - Avoid integer arithmetics (it is only fast on Fermi)
  - MADD – two operation at cost of one
  - Use texture engine for interpolation and caching
- **Filtering (cuFFT fast for 2^n cases and always do complex transforms)**
  - Pad data to a size equal to the closest power of 2
  - Use batched calls
  - Compute two real convolutions using a single complex

PyHST: FBP Implementation

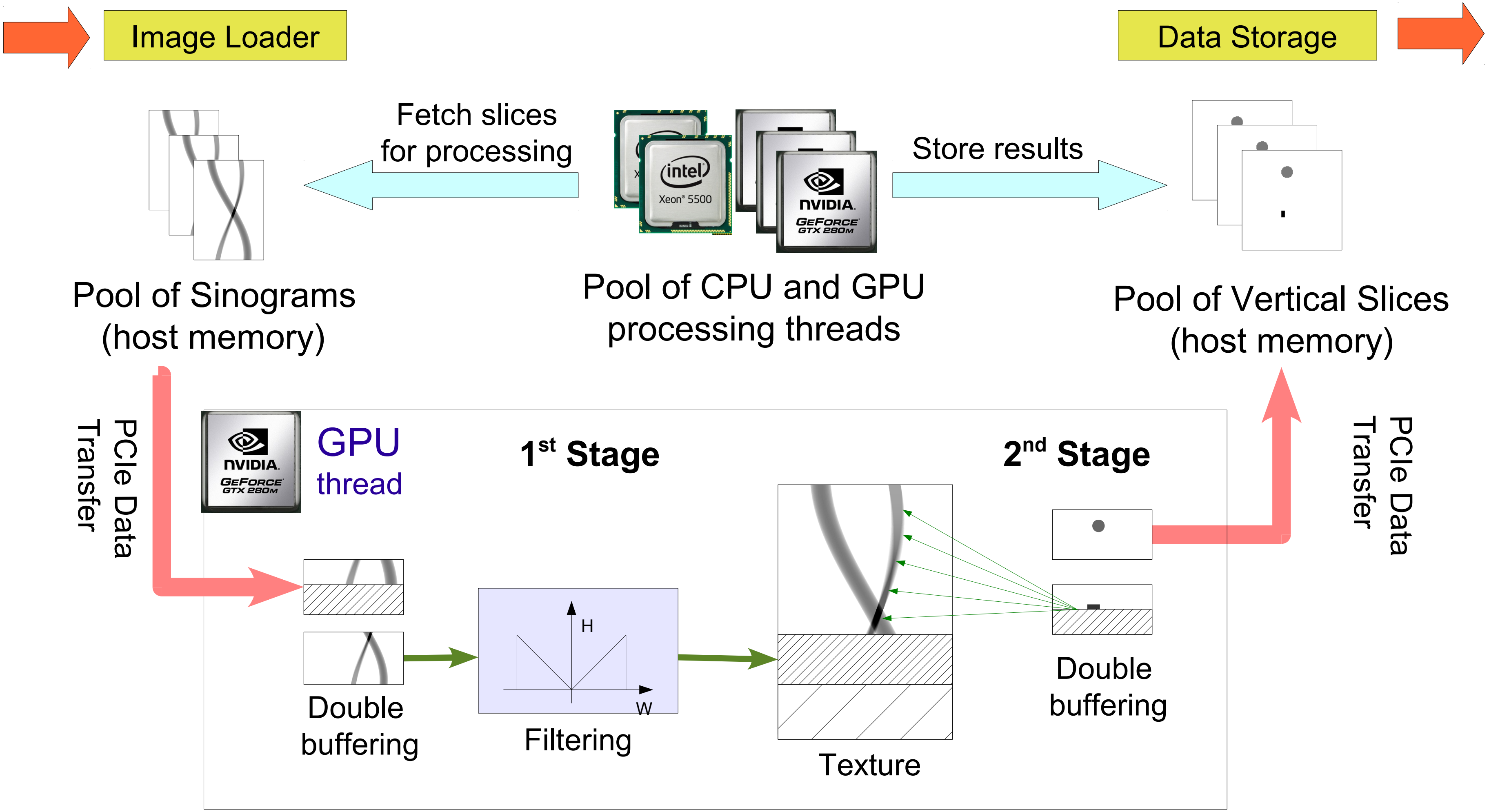


Image Loader

Data Storage

Fetch slices for processing

Store results

Pool of Sinograms (host memory)

Pool of CPU and GPU processing threads

Pool of Vertical Slices (host memory)

PCIe Data Transfer

GPU thread

1st Stage

2nd Stage

Double buffering

Filtering

Texture

Sample Data-Set

Porose polyethylene grains in a conical plastic holder

Sample Data Set:

Number of Projections:	2000
Resolution of Projections:	1776x1707
Total Size:	24GB

Resulting 3D Image:

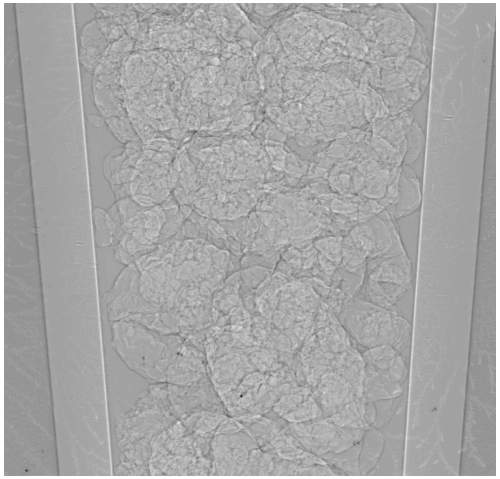
Resolution:	1691x1331x1311
Total Size:	11GB

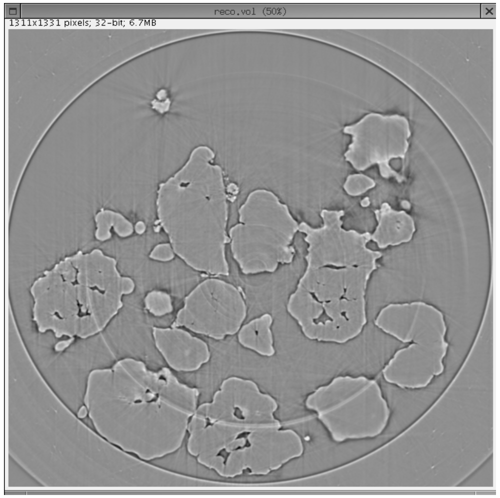
Required Operations:

Data I/O:	35GB
Filtering:	~ 0.6 TFlop
Back Projection:	~ 53.0 TFlop

Maximum Performance Estimation:

Opteron 6176 (~ 110Gflops)	~ 10 minutes
HDD I/O (~ 50 MB/s)	~ 15 minutes

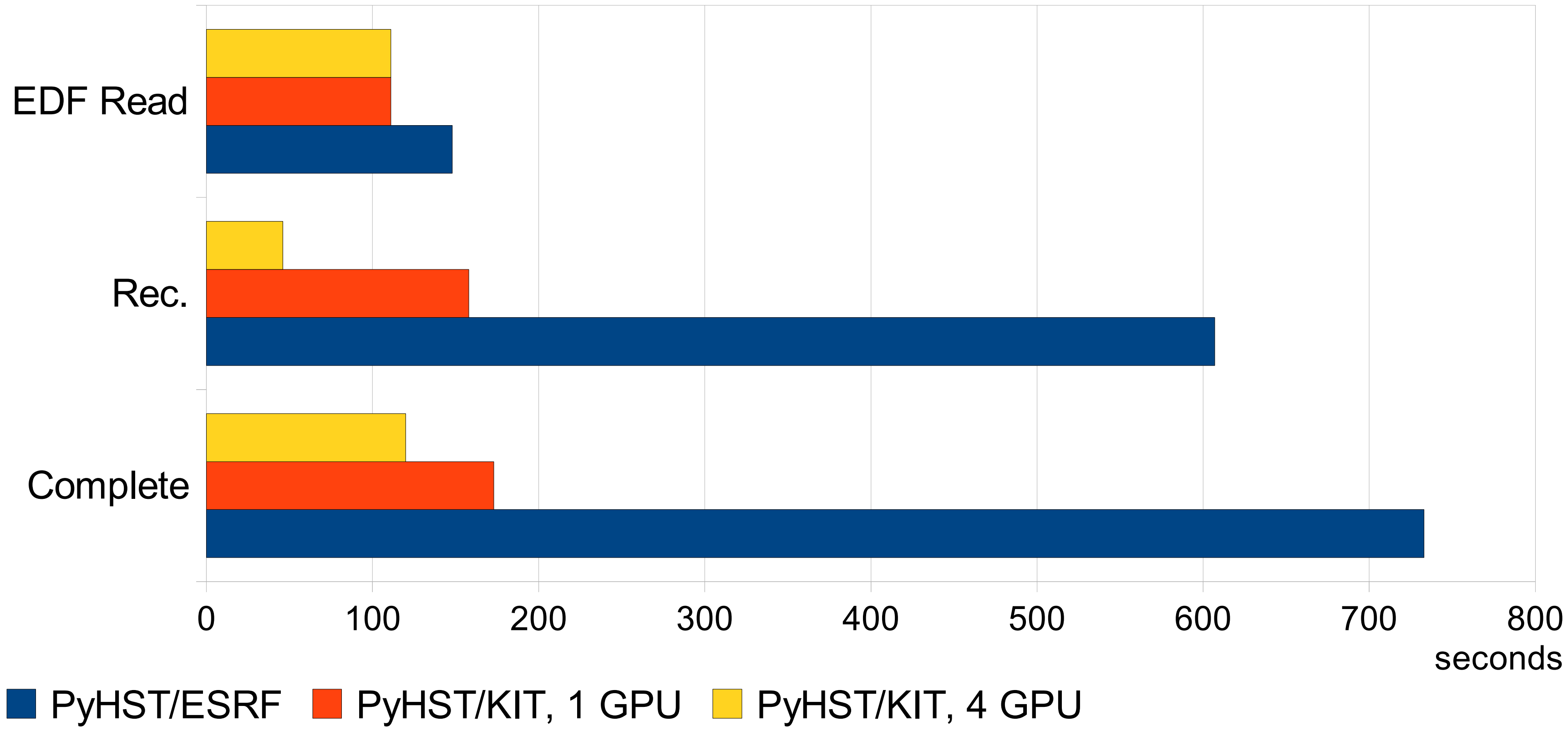




reco.vol (50%)

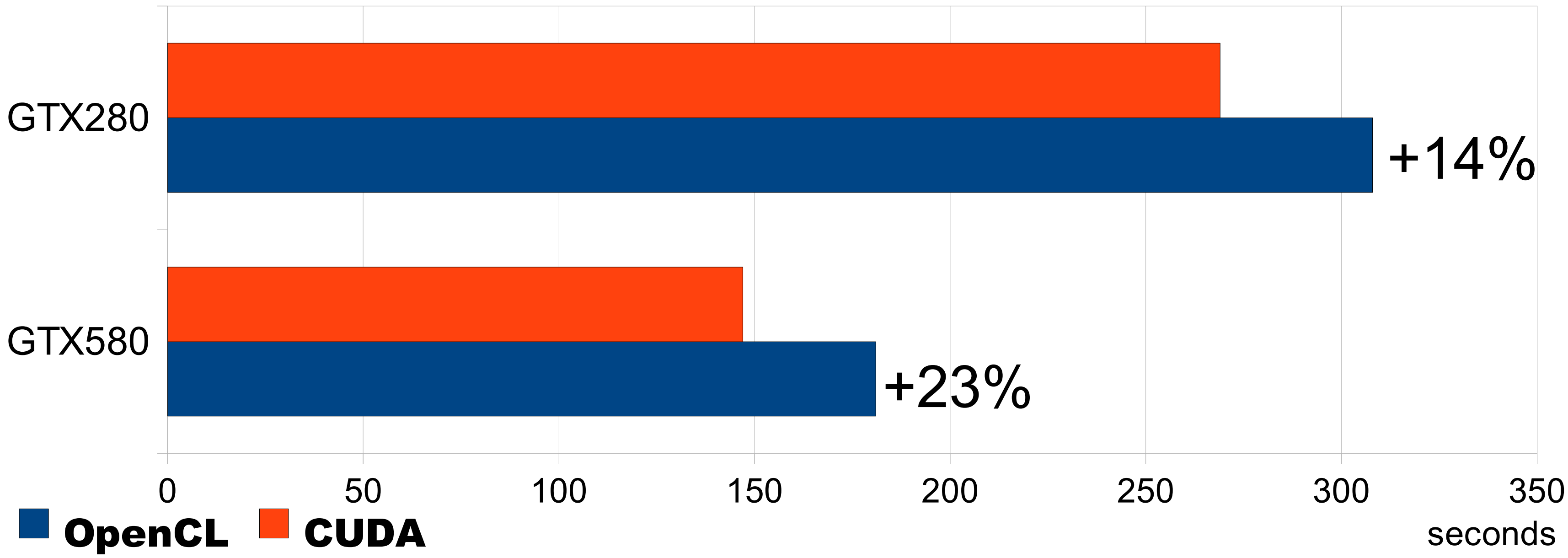
1311x1331 pixels; 32-bit; 6.7MB

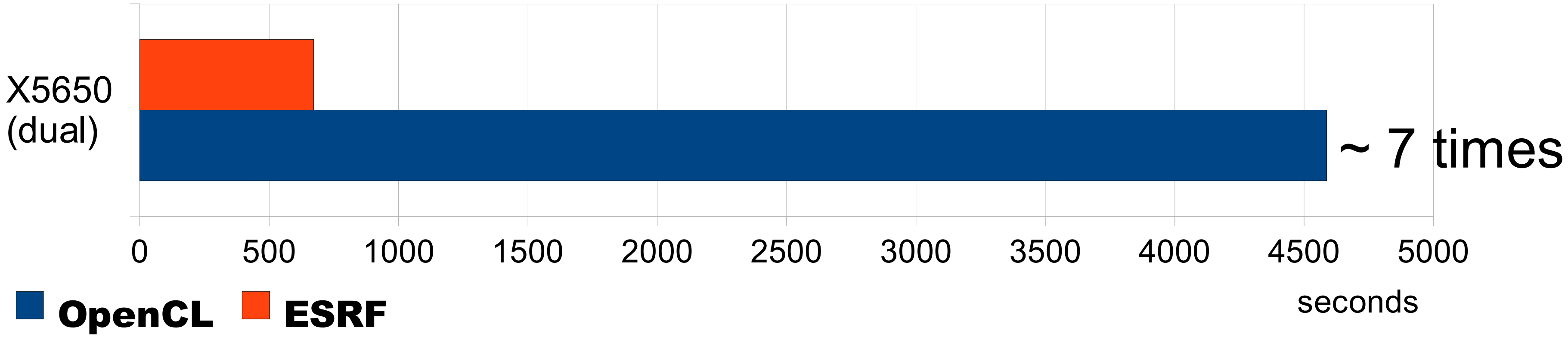
PyHST: Performance Evaluation



CPU	Intel Core i7 950 (4 cores at 3.07 Ghz)
GPU	4 x NVidia GTX 580 (by ASUS)
Memory	12GB DDR3 (Tripple channel DDR3 PC1333)
HDD	2 x Crucial RealSSD C300

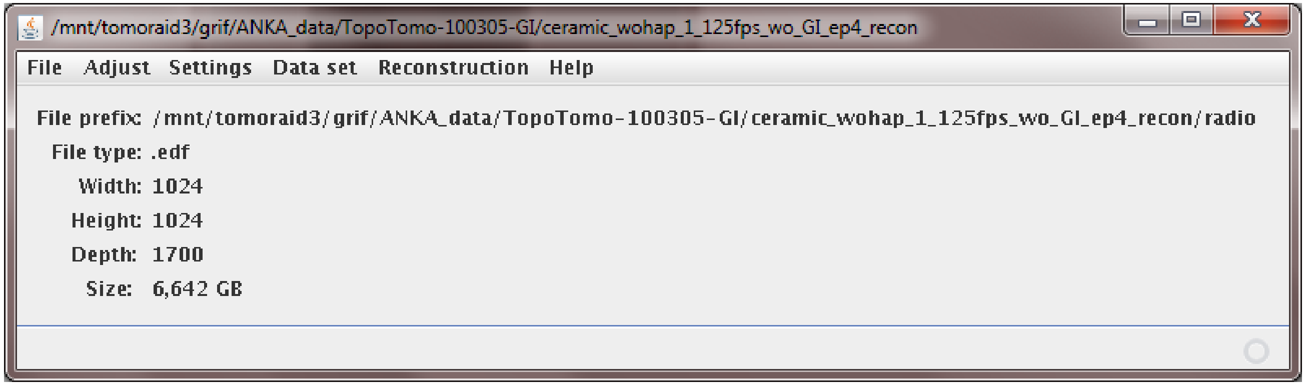
PyHST: Performance of OpenCL





* This is preliminary version, better performance is possible

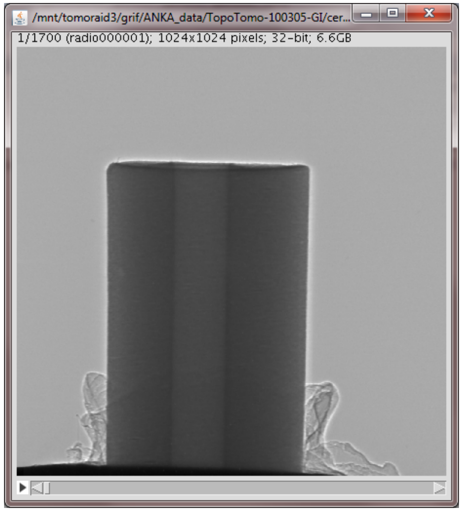
PyHST: GUI


/mnt/tomoraid3/grif/ANKA_data/TopoTomo-100305-GI/ceramic_wohap_1_125fps_wo_GI_ep4_recon
File
Adjust
Settings
Data set
Reconstruction
Help

File prefix: /mnt/tomoraid3/grif/ANKA_data/TopoTomo-100305-GI/ceramic_wohap_1_125fps_wo_GI_ep4_recon/radio
File type: .edf
Width: 1024
Height: 1024
Depth: 1700
Size: 6,642 GB

- Based on Java/ImageJ
- Preview of Radiograms and reconstructed volume (2D)
- Automatic detection of Center of Rotation
- Automatic generation of PyHST configuration
- Reconstruction settings Dialog
- Overlapping of Radiograms

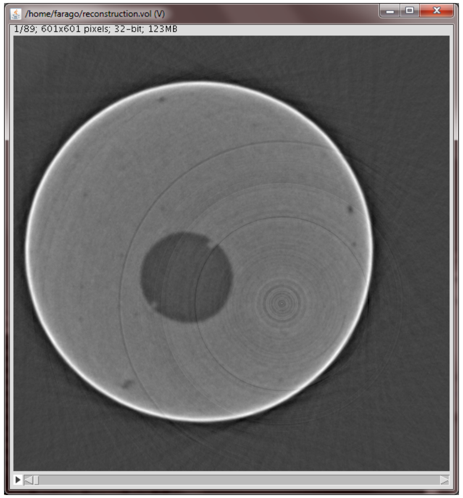
PyHST: GUI



/mnt/tomoraid3/grif/ANKA_data/TopoTomo-100305-GI/cer...

1/1700 (radio000001); 1024x1024 pixels; 32-bit; 6.6GB

Radiograms



/home/farago/reconstruction.vol (V)

1/89; 601x601 pixels; 32-bit; 123MB

Reconstructed Volume

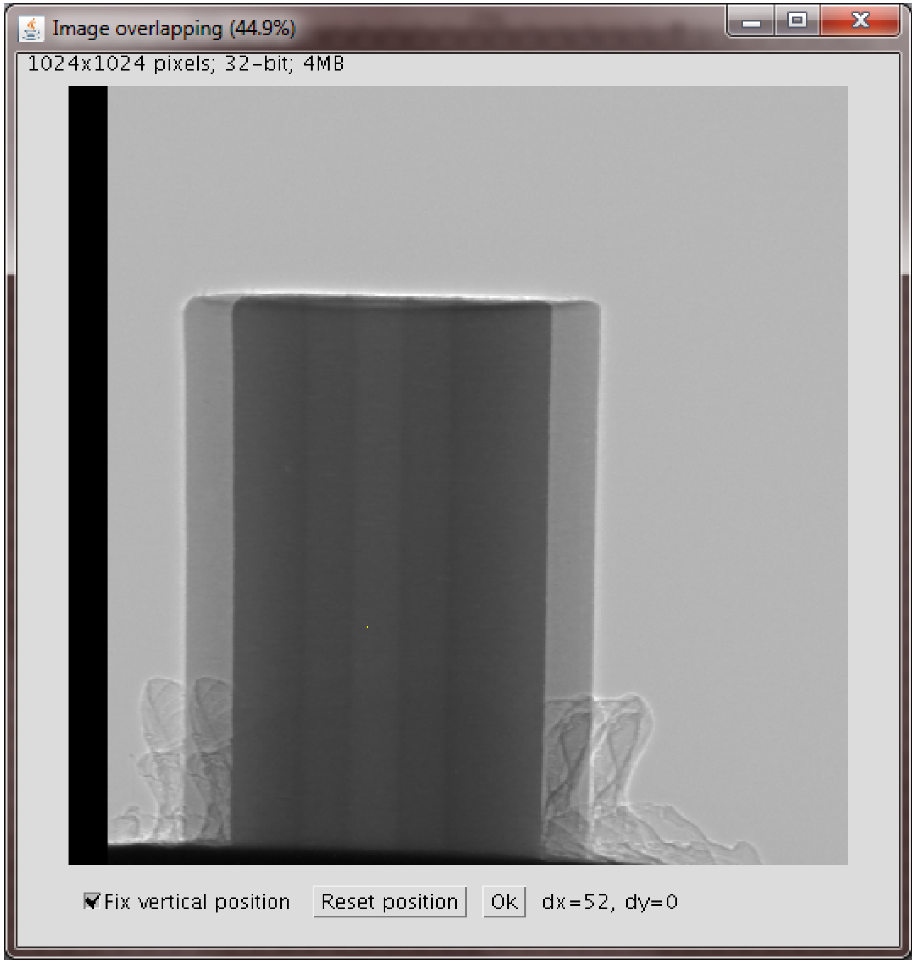


Image overlapping (44.9%)

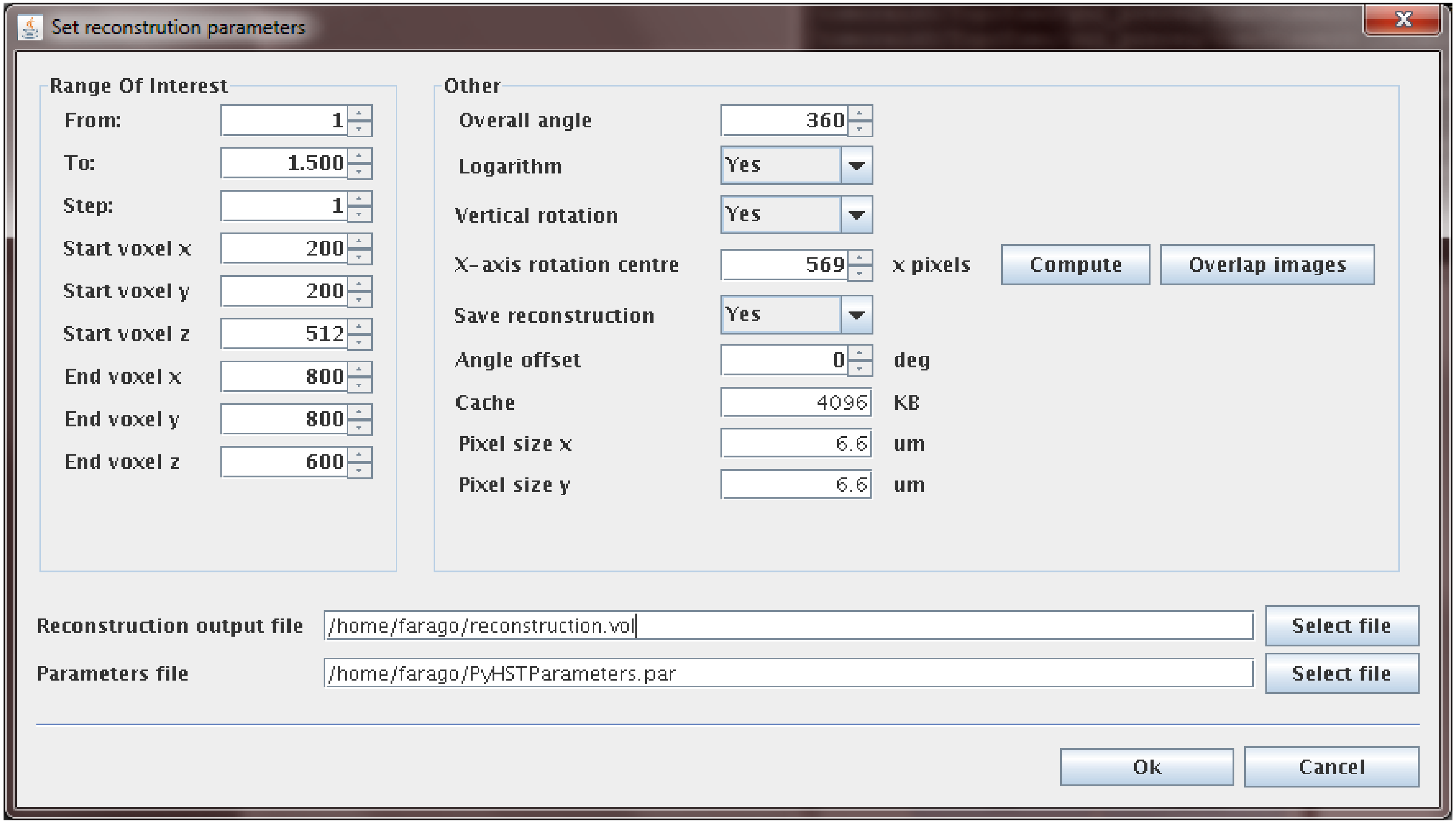
1024x1024 pixels; 32-bit; 4MB

dx=52, dy=0

Overlapping Dialog

- **Fixable vertical position**
- **Position with mouse or keyboard**

PyHST: GUI Settings Dialog



Set reconstrution parameters

Range Of Interest

From: 1
To: 1.500
Step: 1
Start voxel x 200
Start voxel y 200
Start voxel z 512
End voxel x 800
End voxel y 800
End voxel z 600

Other

Overall angle 360
Logarithm Yes
Vertical rotation Yes
X-axis rotation centre 569 x pixels
Save reconstruction Yes
Angle offset 0 deg
Cache 4096 KB
Pixel size x 6.6 um
Pixel size y 6.6 um

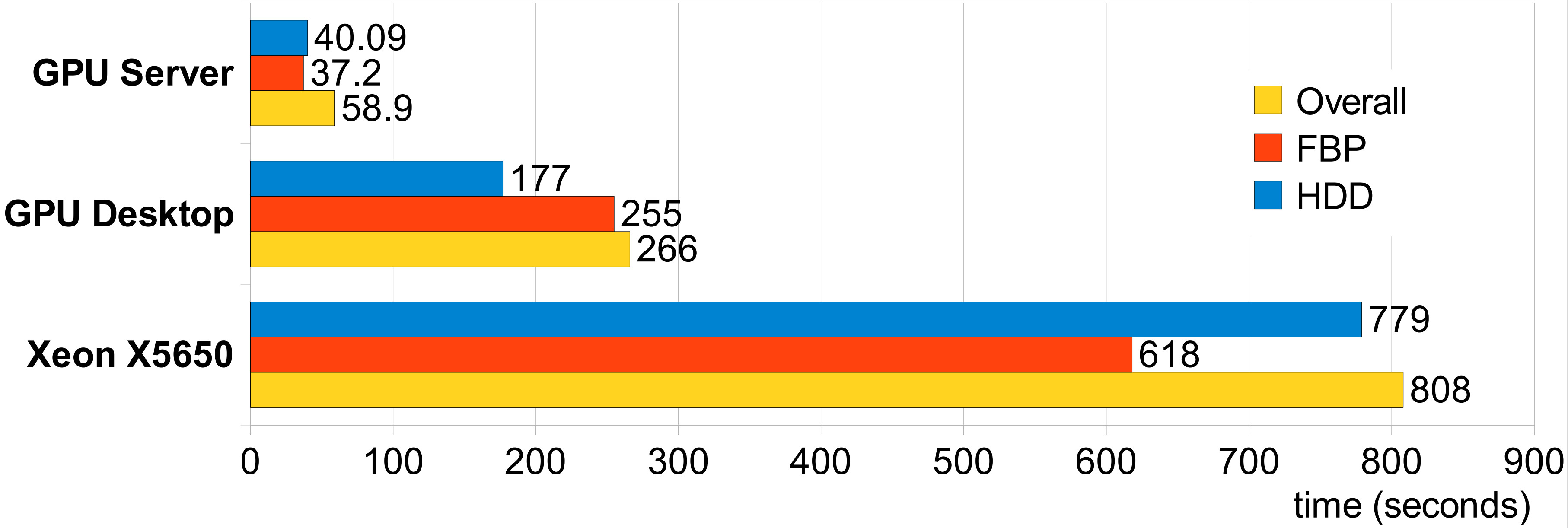
Reconstruction output file /home/farago/reconstruction.vol
Parameters file /home/farago/PyHSTParameters.par

Hardware Evaluation

- GPU vs. CPU: Performance Evaluation
- NVIDIA and AMD products for consumer market
- Evaluation of NVIDIA professional cards
- GPU computing platforms at KIT
- External GPU box
- Evaluation of storage configurations

Performance: GPU vs. CPU

	Xeon Server	GPU Desktop	GPU Server
Type of Computation	CPU / Xeon X5650 12 cores, 2.66 GHz	**GeForce GTX 280** **1 core**	**2 x GTX295 + 2 x GTX580** **6 cores**
CPU	2 x Xeon E5650	Core2 E6300	2 x Xeon E5540
Memory	16GB DDR3	4GB DDR2	96GB DDR3
HDD/SSD	Hitachi A7K2000	**2 x Intel X25-E**	**4 x Crucial RealSSD C300**
Price	5500$ (2000$ CPUs)	1500$ (400$ GPU)	9000$ (2000$ GPU, 1200$ SSD)
Software	SuSe 11.3, CUDA 3.2, MKL 10.2.1, gcc4.5 -O3 -march=nocona -mfpmath=sse		

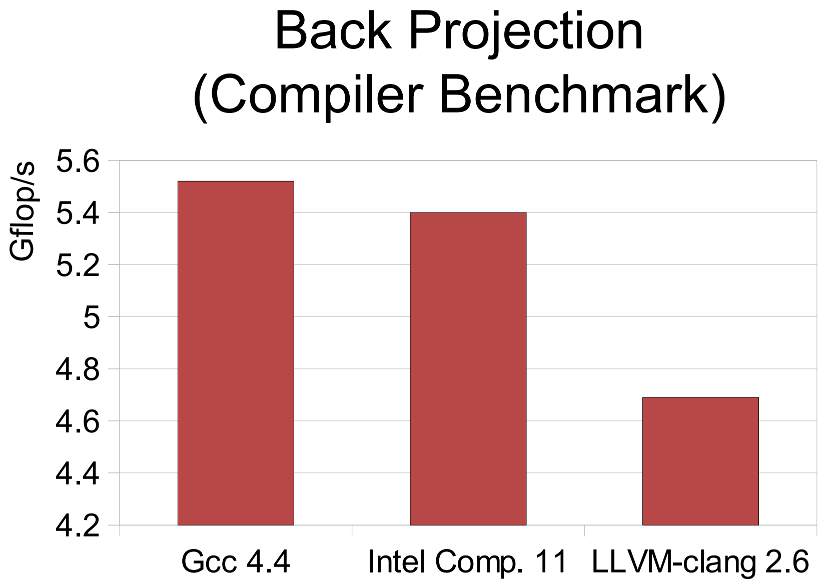


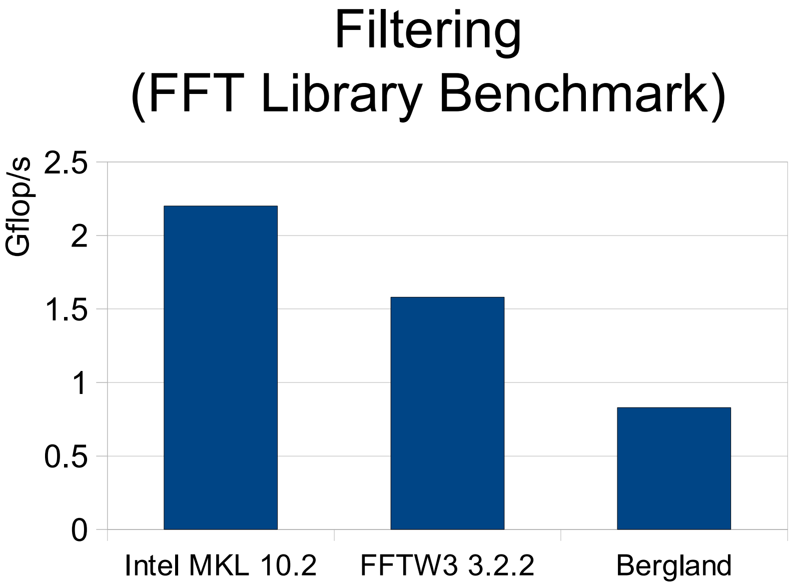
Performance: Compiler Benchmark

Compiler flags

Intel: -O3 -xS (with SSE vectorization)

Gcc/LLVM: -O3 -march=nocona (with SSE vectorization)

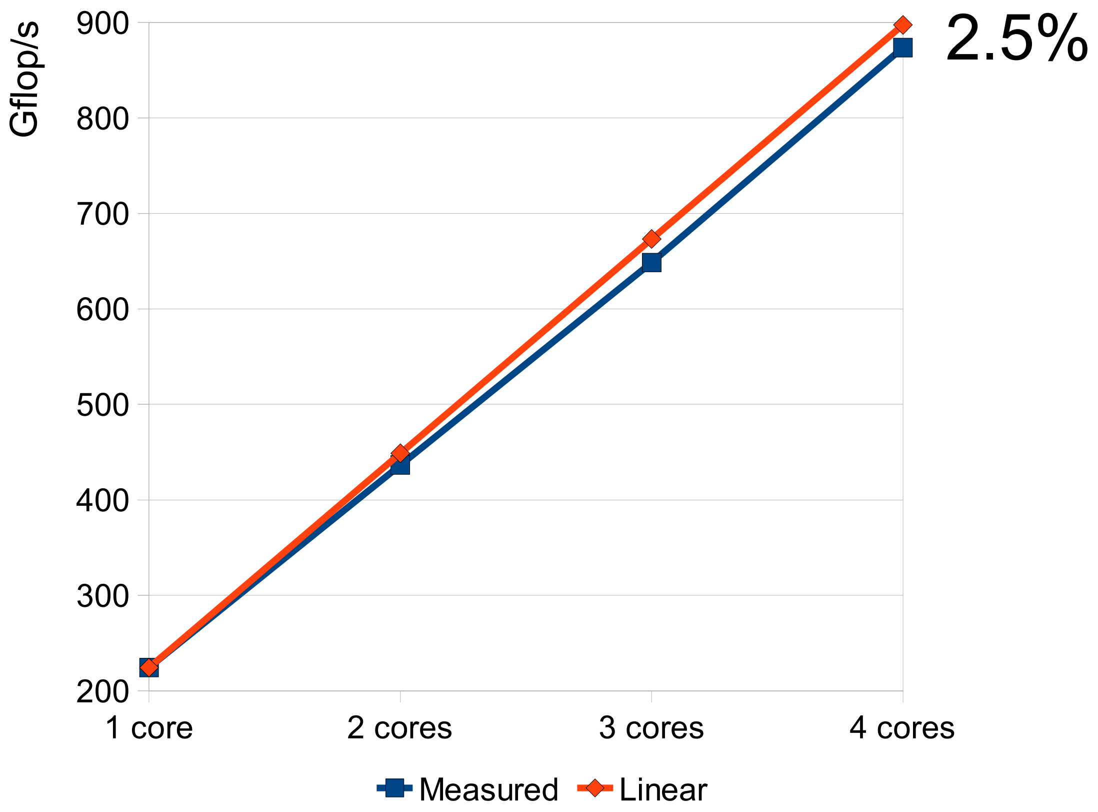




Decision: Using '**gcc**' and '**Intel MKL**' in all benchmarks

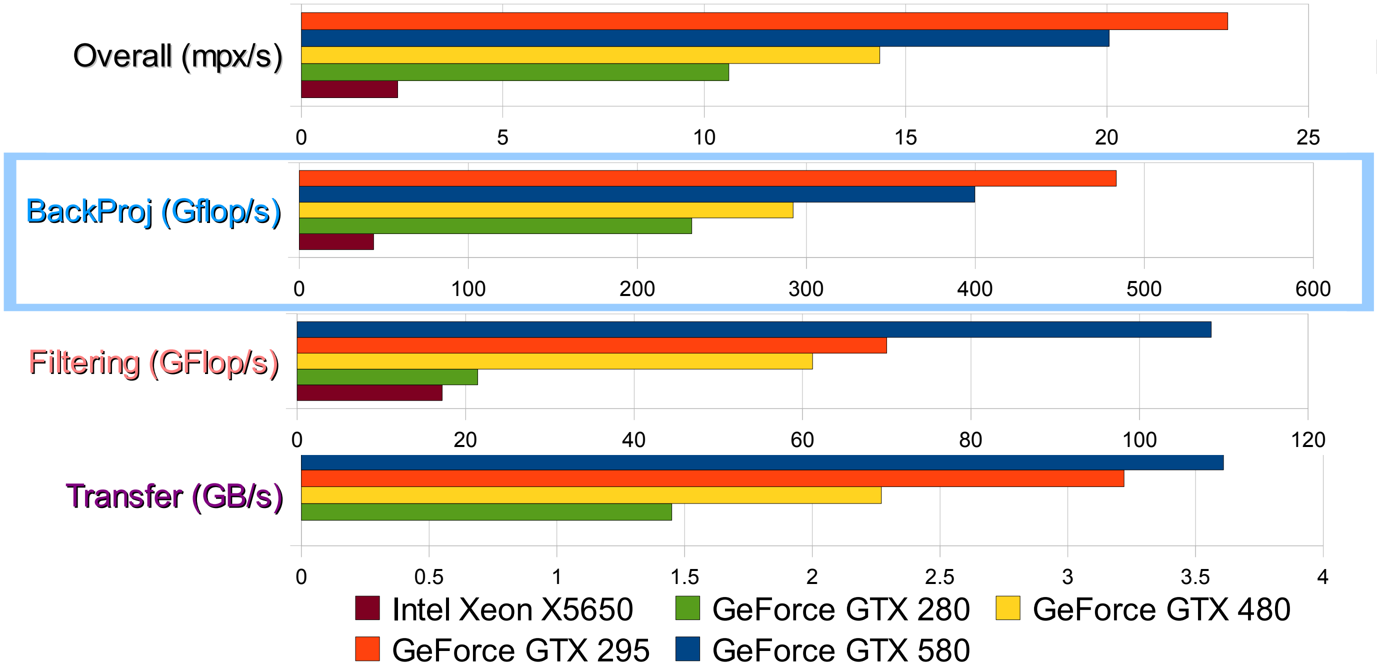
Performance: Scalability

Karlsruhe Institute of Technology

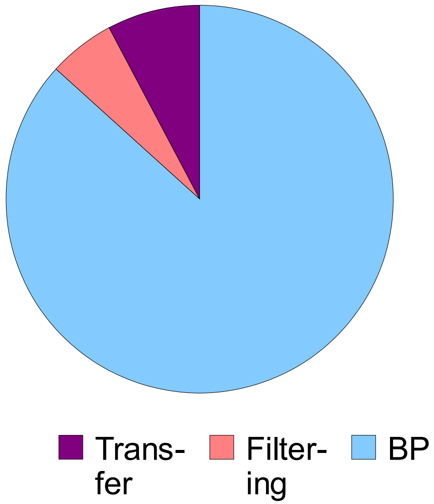


Back Projection using Tesla S1070

Performance: NVIDIA Desktop Products

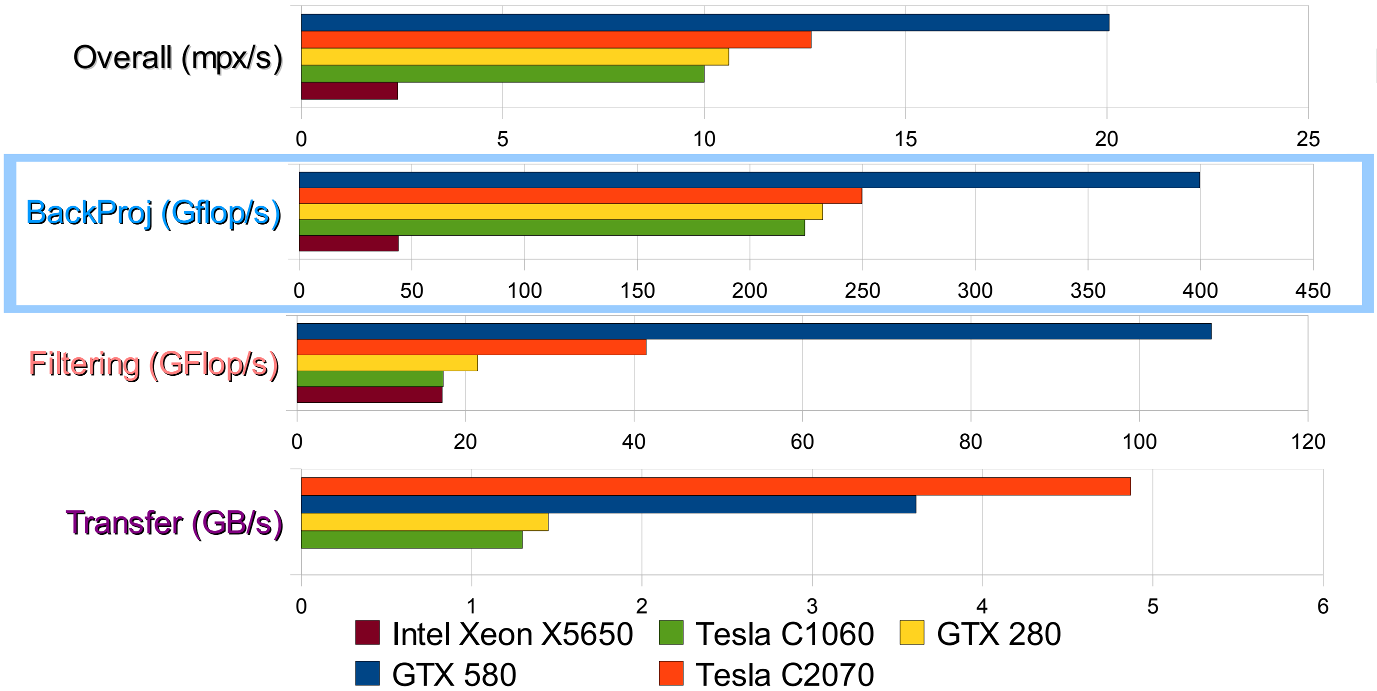


Ratio of Operations

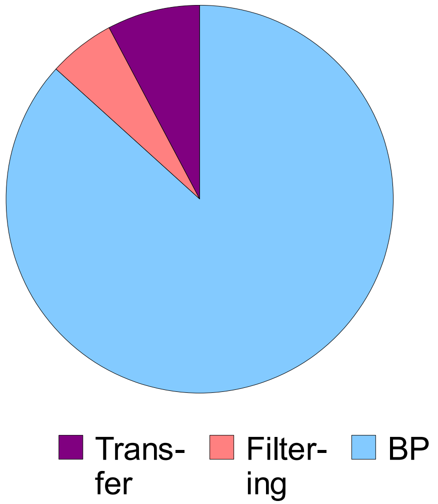


	GTX280	GTX295	GTX480	GTX580
Architecture	GT200	GT200	Fermi	Fermi
Processors	1 x 240 @ 1.3 GHz	**2 x** 240 @ 1.3 GHz	1 x 480 @ 1.4 GHz	1 x 512 @ 1.54 GHz
Memory	1 GB @ 142 GB/s	900 MB @ 112 GB/s	1.5 GB @ 177 GB/s	1.5 GB @ 192 GB/s
Texture	1 x 48 GT/s	2 x 46 GT/s	1 x **42 GT/s**	1 x 49.4 GT/s
Power Cons.	236 W	289 W	250 W	250 W
Price			$400	$500

Performance: NVIDIA Server Products

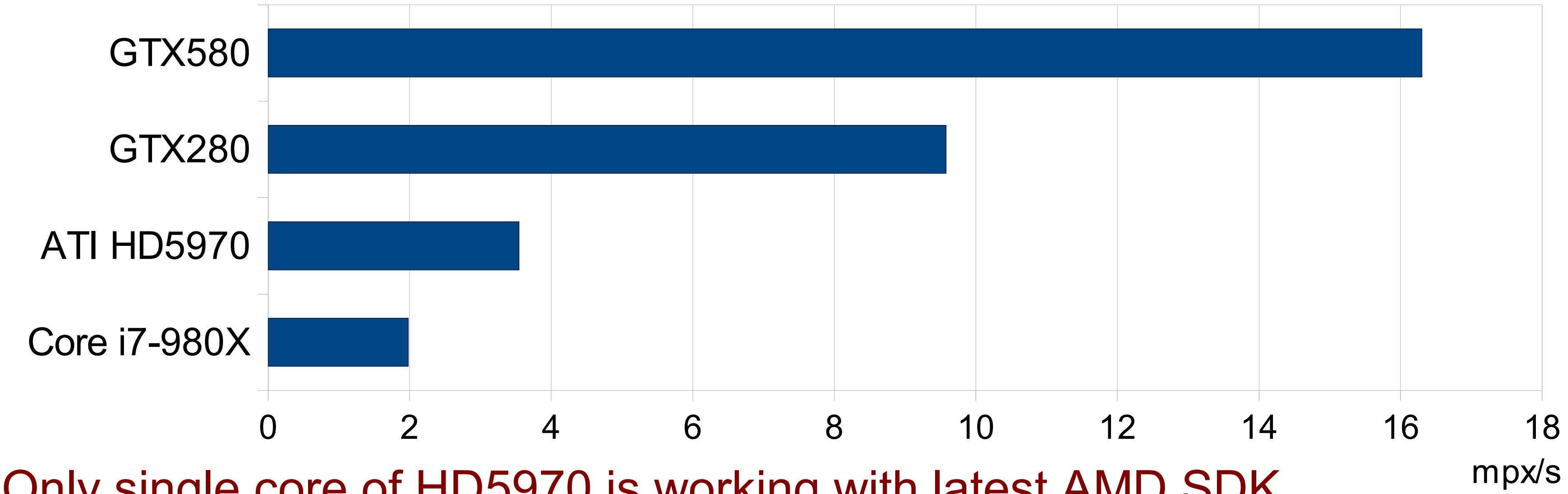


Ratio of Operations



	GTX280	Tesla C1060	GTX580	Tesla C2070
Architecture	GT200	GT200	Fermi	Fermi
Processors	1 x 240 @ 1.3 GHz	1 x 240 @ 1.25GHz	1 x 512 @ 1.54 GHz	1 x 448 @ 1.15 GHz
Memory	1 GB @ 142 GB/s	4 GB @ 102 GB/s	1.5 GB @ 192 GB/s	6 GB @ 144 GB/s
Texture	48 GT/s	48 GT/s	49.4 GT/s	42 GT/s
Power Cons.	236 W	187.8 W	250 W	238 W
Price		$2000	$500	$3500

Performance: ATI Radeon HD5970

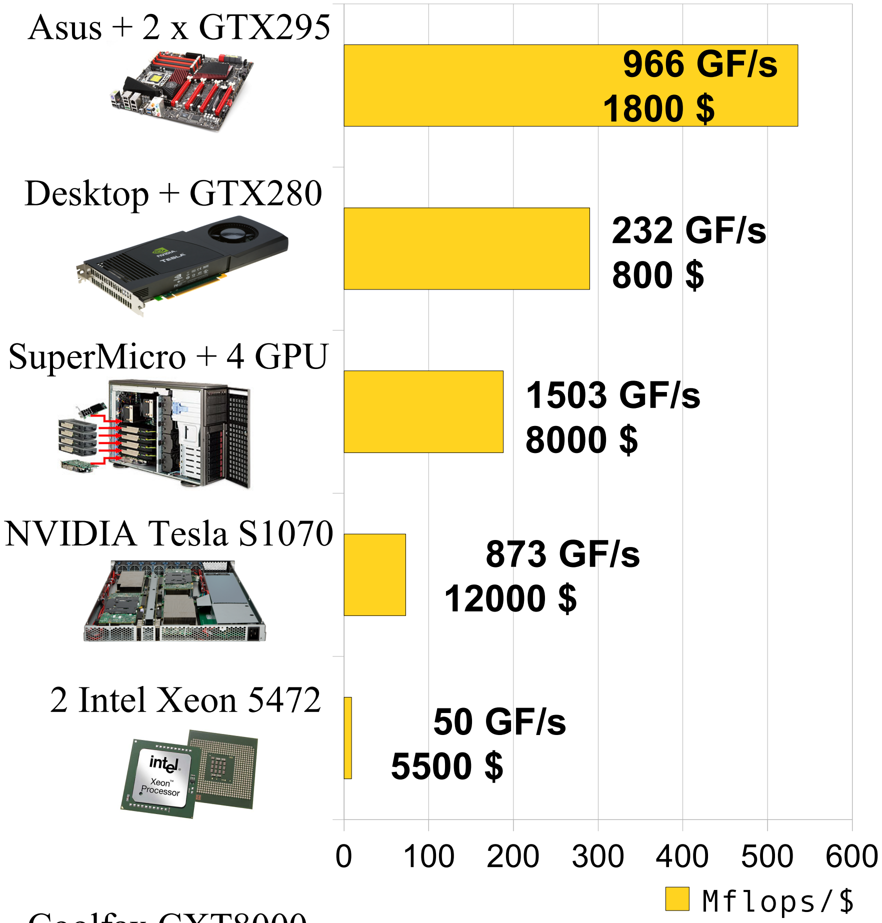


* Only single core of HD5970 is working with latest AMD SDK

** OpenCL support is experimental

	GTX280	GTX580	HD5970
Architecture	GT200	Fermi	Cypress
Processors	1 x 240 @ 1.3 GHz	1 x 512 @ 1.54 GHz	2 x 1600 @ 725 MHz
Memory	1 GB @ 142 GB/s	1.5 GB @ 192 GB/s	1.5 GB @ 153.6 GB/s
Texture	48 GT/s	49.4 GT/s	2 x 68 GT/s
Power Cons.	236 W	250 W	294 W
Price		$400	$650

Performance: GPU Platforms



Asus Rampage III Extreme (1800$ = Core i5 + 2 x GTX295 + 8 GB)
Chipset: x58, 36 PCIe 2.0 lanes; 6 DDR3 slots (48 GB max)
PCIe 2.0 x16: 4 (x8 if all 4 are used)
Max Peak Performance (ATI): 18.56 Tflops / 3.7 Tflops
Max Peak Performance (NVidia): 7.15 Tflops / 595 GFlops

Standard Desktop (800$ = Core2 + GTX285 + 2 GB)

SuperMicro 7046GT-TRF (~8000$ = 2 Xeon + 4 GPU + 96 GB)
Chipset: Dual Intel 5520, 72 PCIe 2.0 lanes, 12 DDR3 slots (192GB max)
PCIe 2.0 x16: 4 (full speed), x4: 2 (in x16 slots); PCIe 1.0 x4: (in x8 slot)
Max Peak Performance (ATI): 18.56 TFlops / 3.7 Tflops
Max Peak Performance (NVidia): 7.15 TFlops / 2.5 Tflops

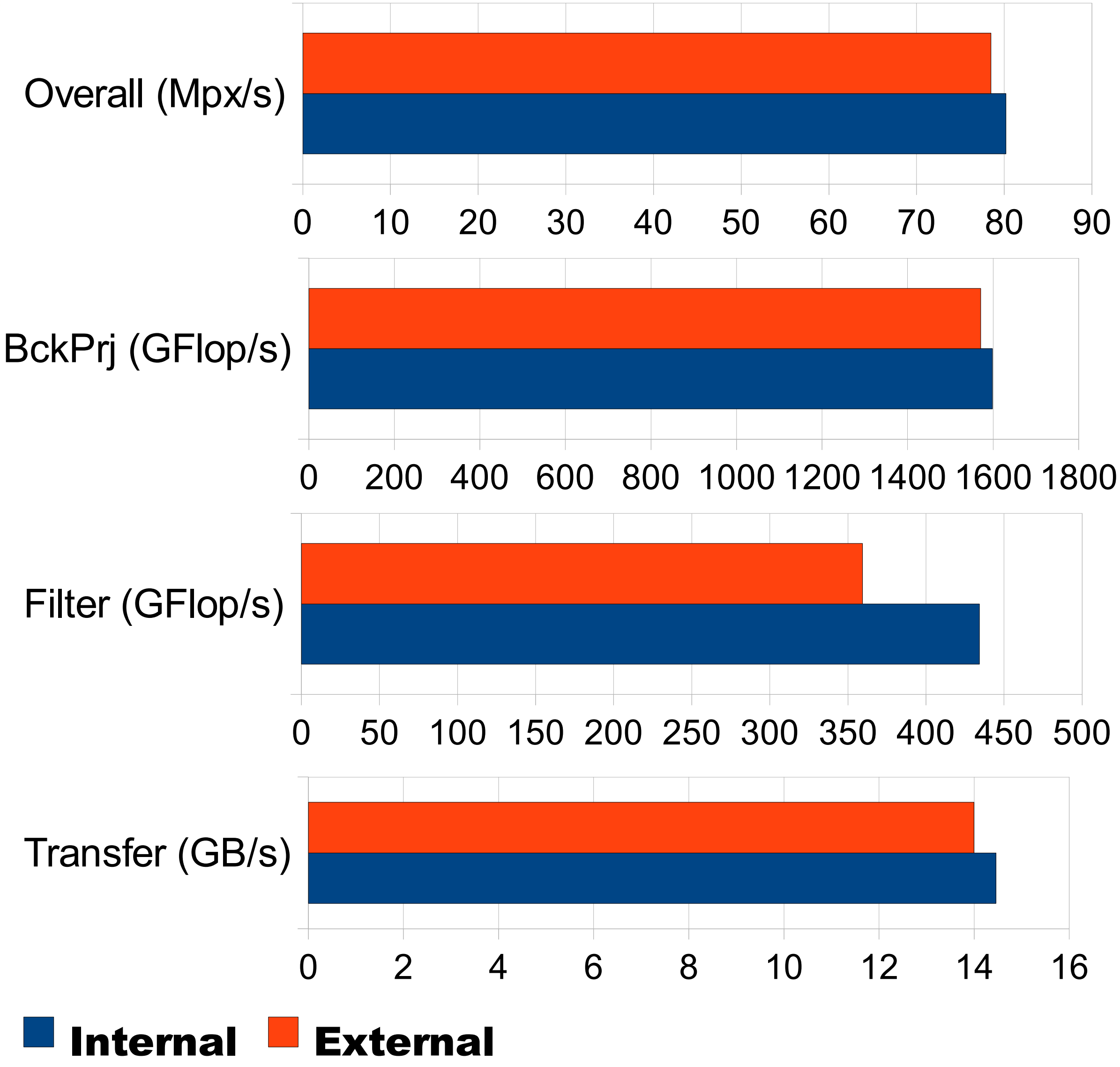
NVIDIA Tesla S1070 (~8000$ + 4000$ host)
System: Requires separate host server
GPU Devices: 4 x Tesla C1060 (960 parallel processors at 1.44 GHz)
Peak performance: 4.14 Tflops / 345 Gflops
GPU Memory Size: 16 GB

Dual Xeon 5472 Server (5500$ = 16 GB)
Max Peak Performance: 96 Gflops / 48 GFlops

Coolfax CXT8000

Coolfax CXT8000 (36000$ = 2 Xeon + 8 Tesla C2050 + 144GB)
Chipset: Dual Intel 5520, 72 PCIe 2.0 lanes
PCIe Switch: PLX PEX8647; PCIe 2.0 x16: 8 (full speed)
18 DDR3 Memory Slots: 288 GB max
Max Peak Performance: 10 Tflops / 5 Tflops
Back Projection Performance: 2336 Gflops (Estimated)

Performance: External GPU boxes



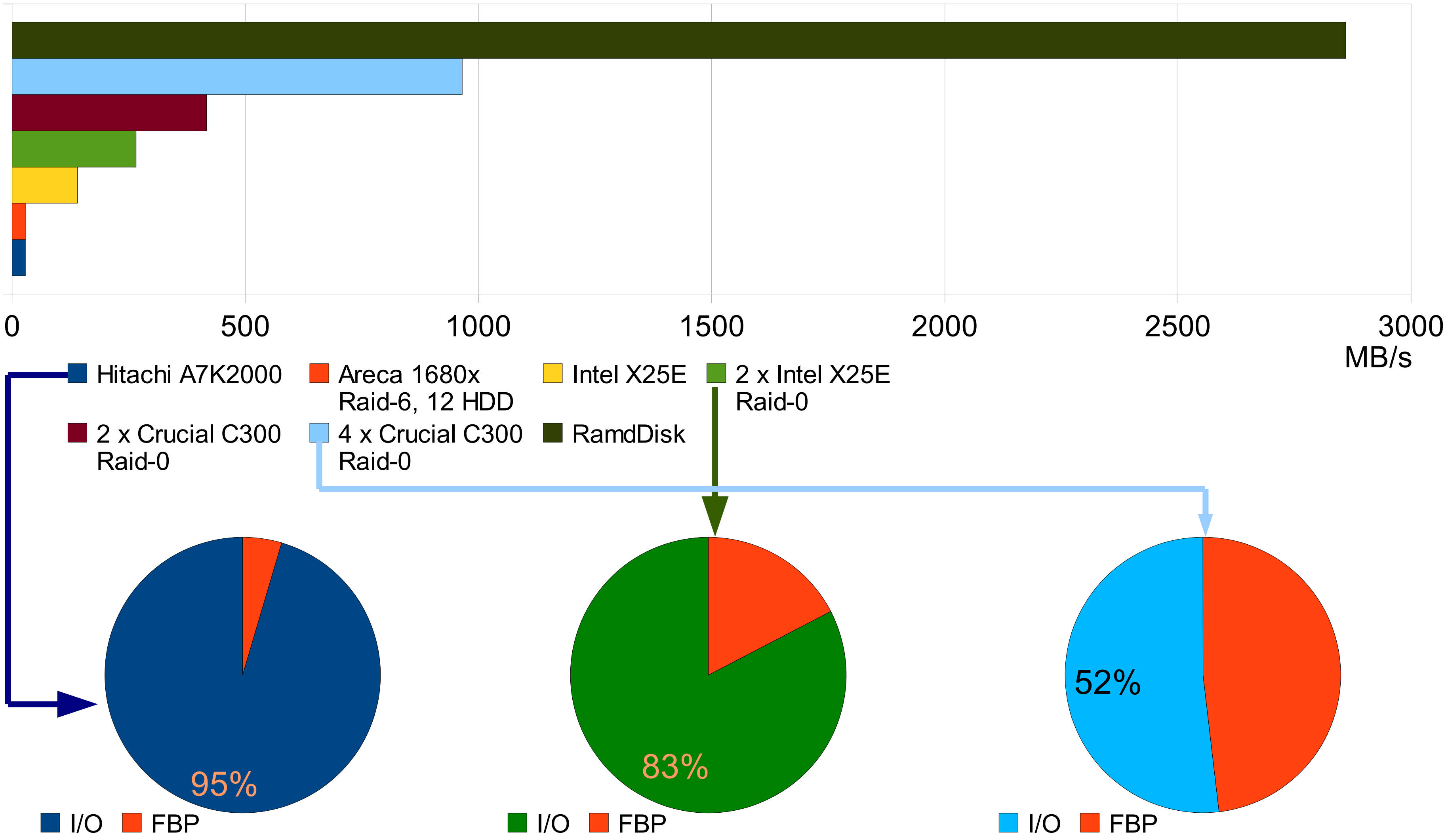
External GPU Box

PCIe Interface Card (16x)

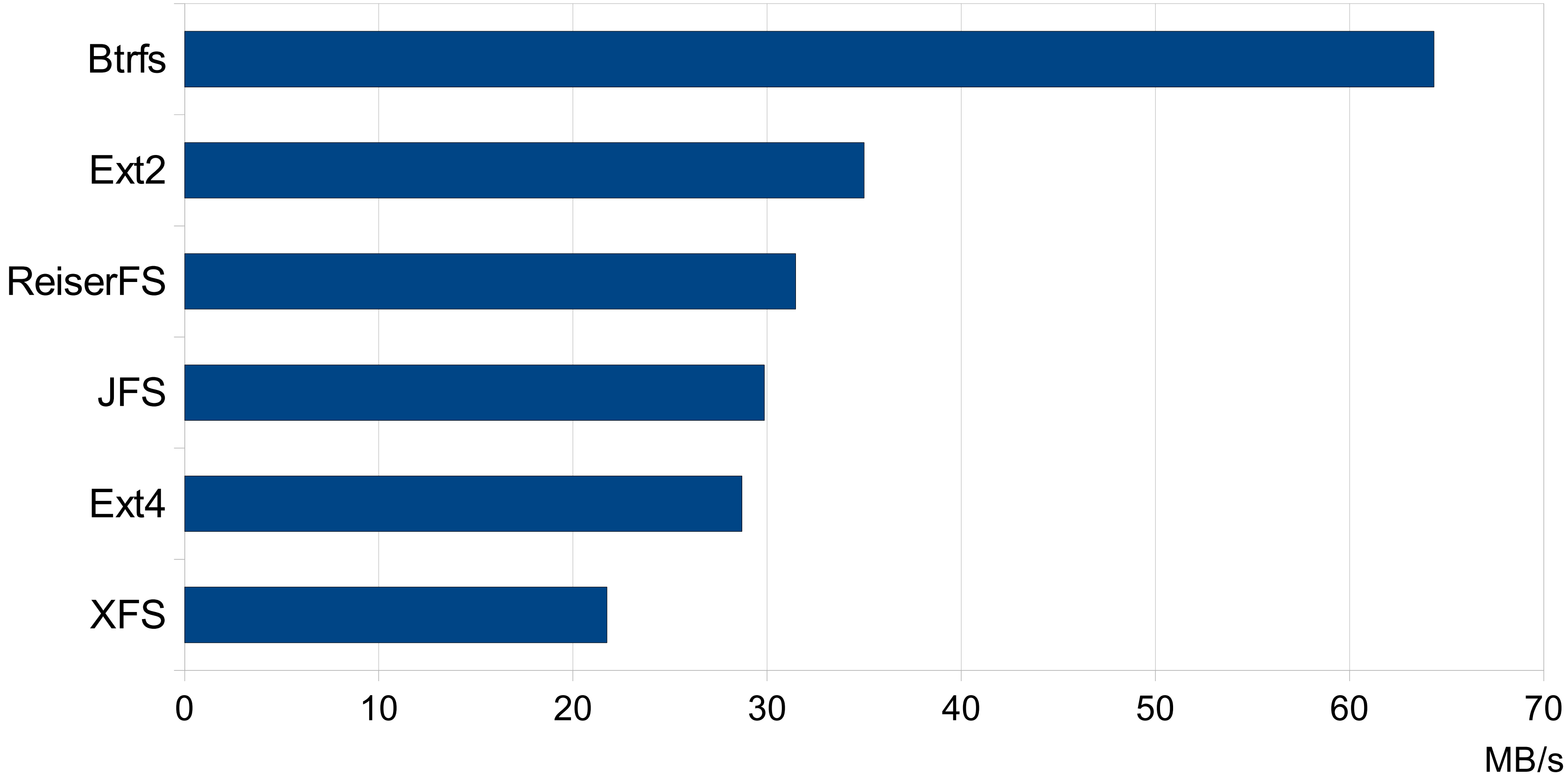
4 External GPUs

4600 EUR

Performance: Storage Systems



Performance: File Systems

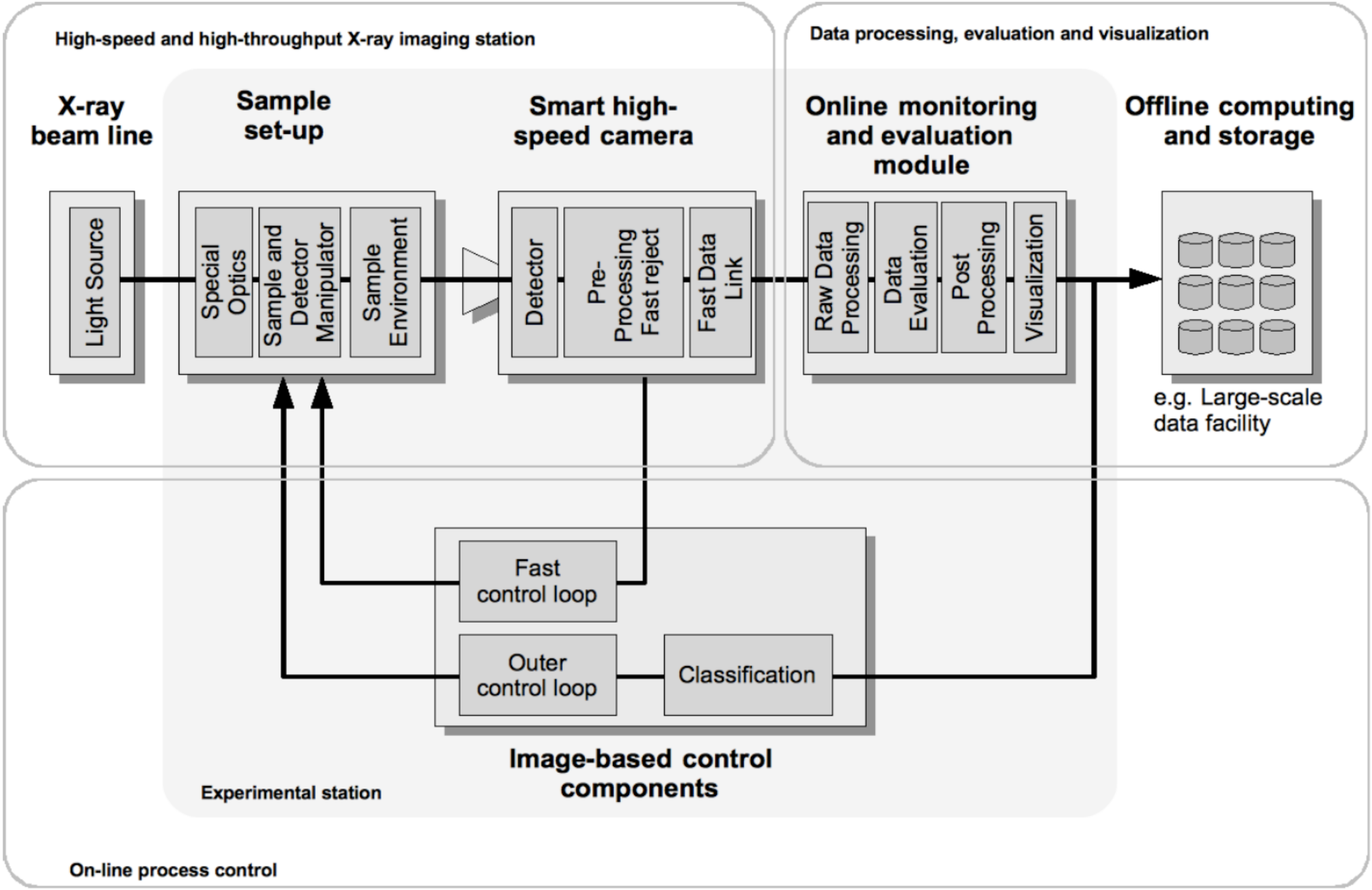


* Performance measured with Hitachi A7K2000 hard drive

UFO and Future Development

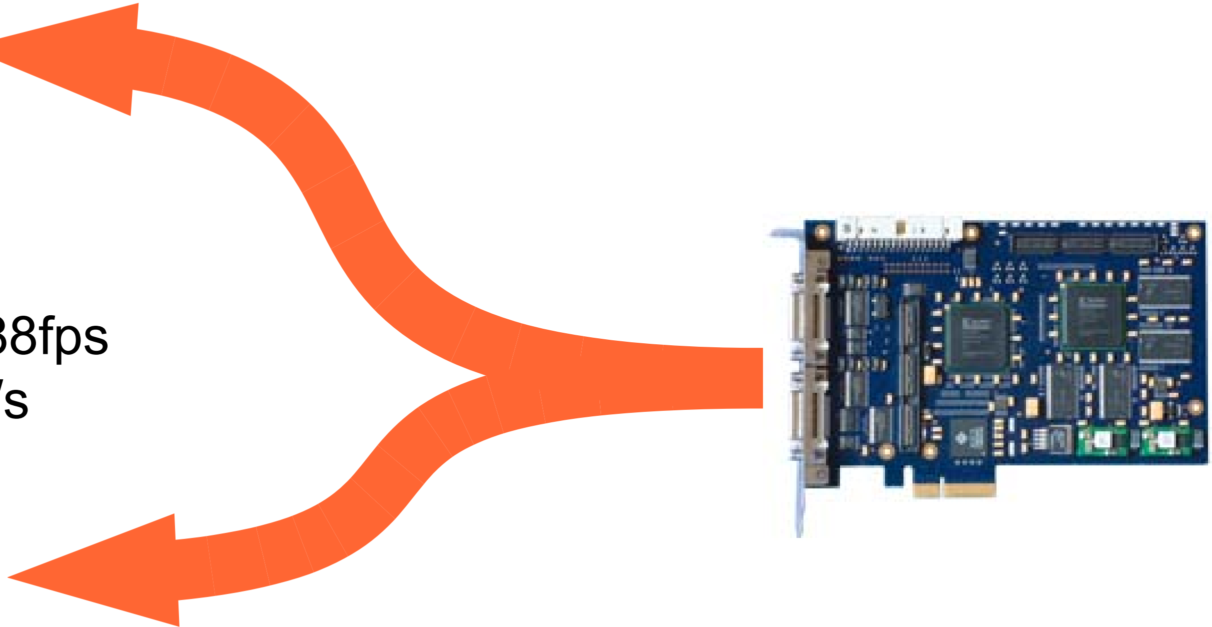
- UFO Project - Ultra Fast X-ray imaging of scientific processes with On-line assessment and data-driven process control
- UFO Hardware and Test Setup
- UFO Software Stack
- Managing data at high bandwidths

UFO – Ultra Fast X-ray imaging of scientific processes with On-line assessment and data-driven process control


High-speed and high-throughput X-ray imaging station
Data processing, evaluation and visualization
X-ray beam line
Sample set-up
Smart high-speed camera
Online monitoring and evaluation module
Offline computing and storage
Light Source
Special Optics
Sample and Detector Manipulator
Sample Environment
Detector
Pre-Processing Fast reject
Fast Data Link
Raw Data Processing
Data Evaluation
Post Processing
Visualization
e.g. Large-scale data facility
Fast control loop
Outer control loop
Classification
Image-based control components
Experimental station
On-line process control

UFO Hardware: Commercial Cameras

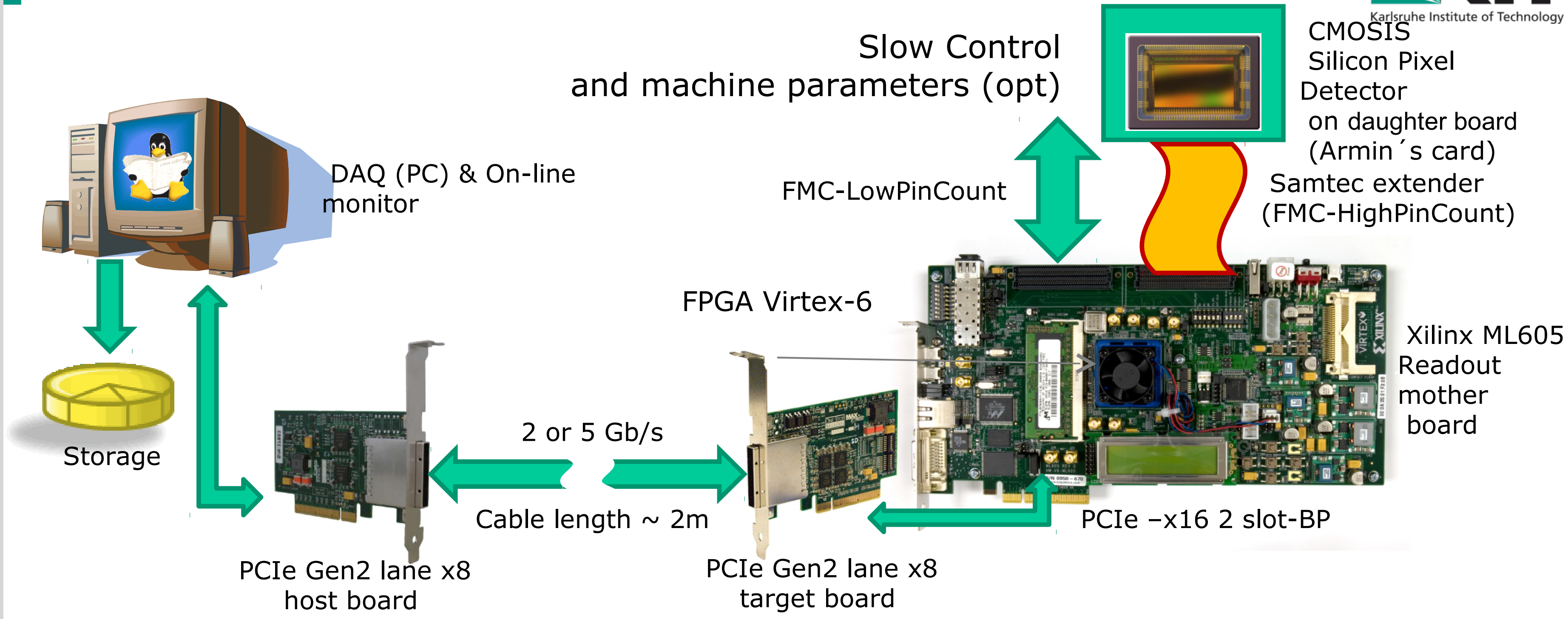
<u>Photon Focus MV2</u>
1280x1024@8 at 488fps
Bandwidth: 610 MB/s



Camera Link Interface

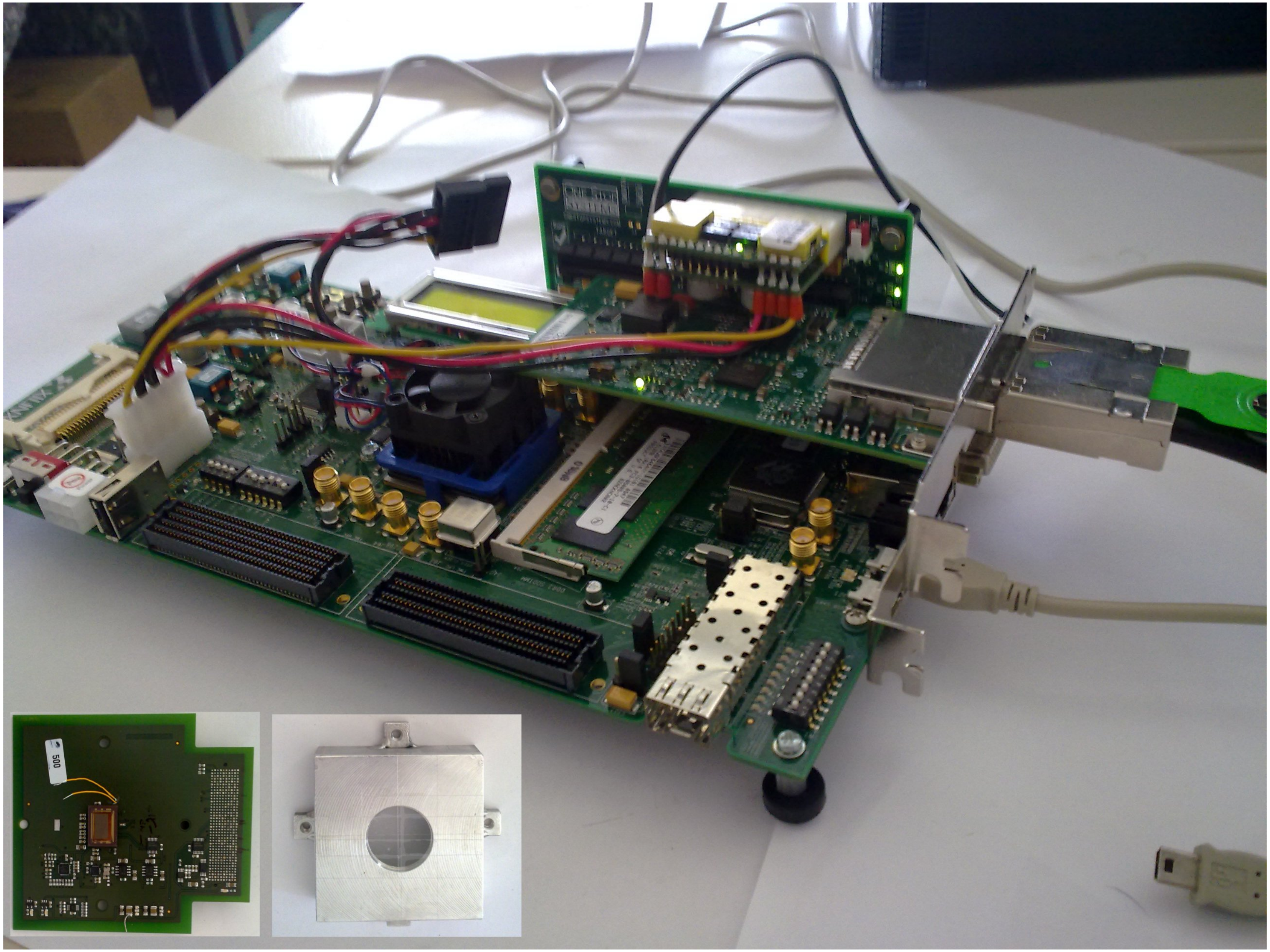
<u>PCO.Edge</u>
Slow: 2560x2160@16 at 33fps
Fast: 2560x2160@12 at 100fps
Bandwidth: 790 MB/s

UFO Hardware: IPE Camera

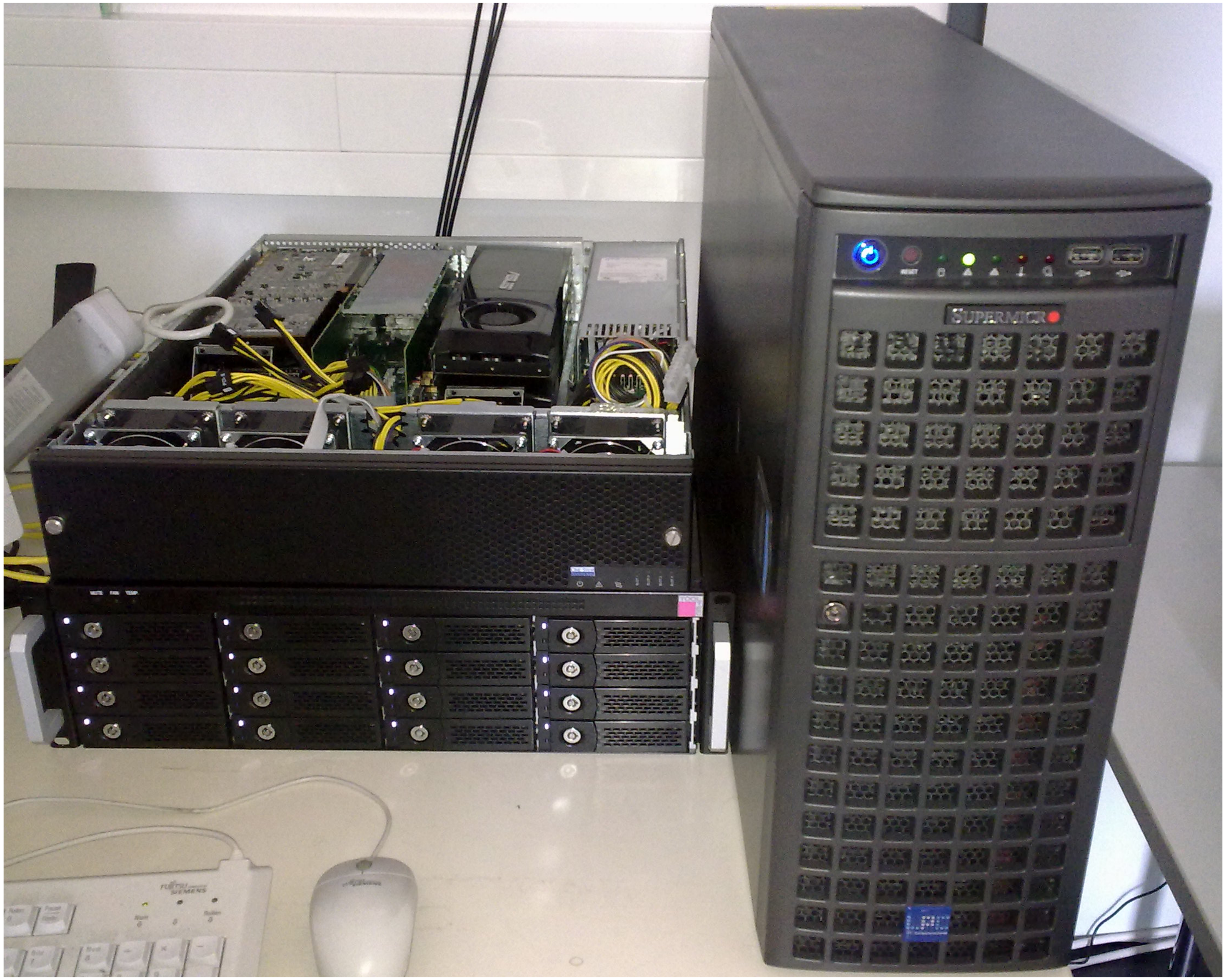


- PCIe x8 (gen2) interface (up to 4 GB/s)
- CMOSIS Pixel Detector CMV2000
  - Maximum Resolution: 2048 x 1088 @ 12 bits
  - Highest Bandwidth: 900 MB/s (2048 x 1088 @ 10 bits at 340 FPS
- FPGA-based frame reject and compression

UFO Hardware: Prototype of IPE Camera

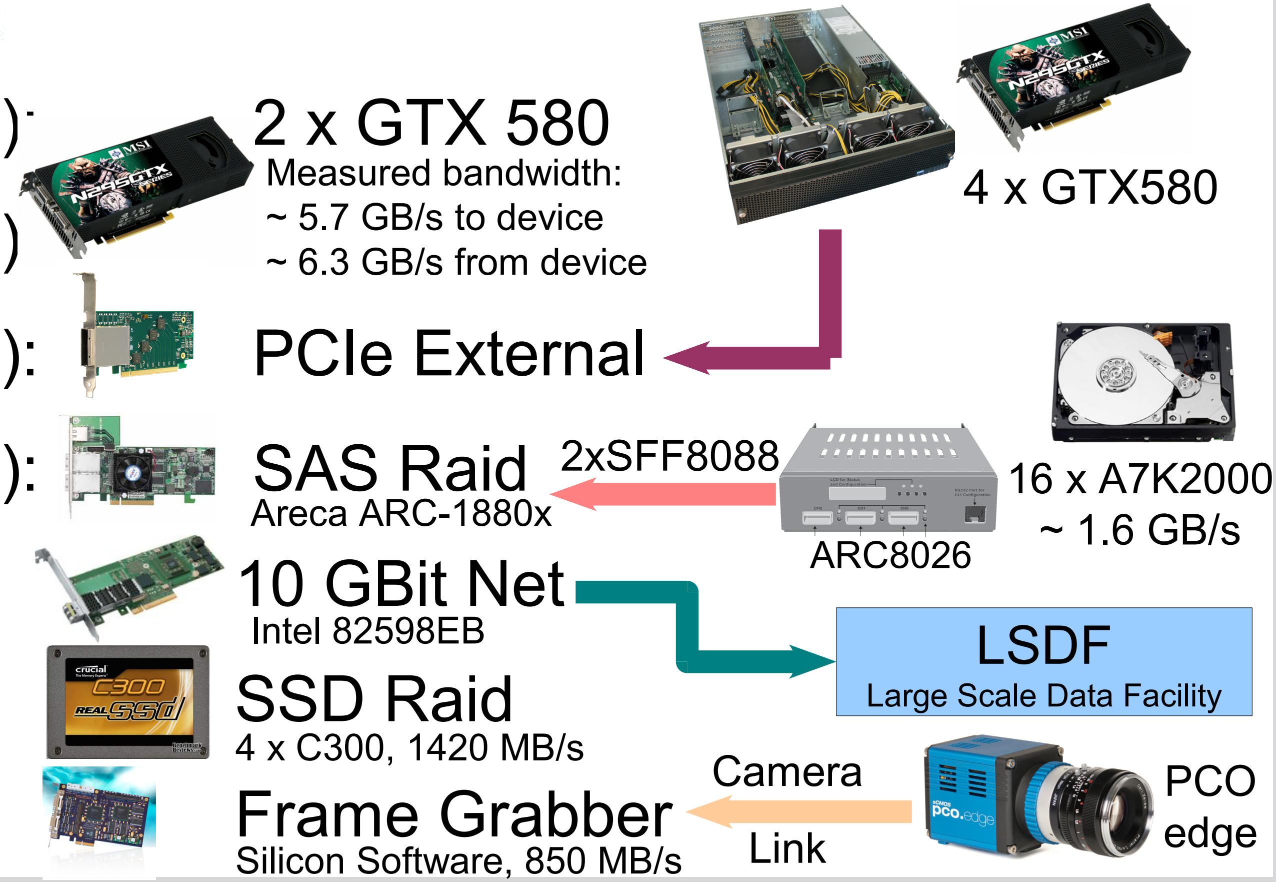


UFO Hardware: Reconstruction Station



UFO Hardware: Reconstruction Station

SuperMicro 7046GT-TRF (Dual Intel 5520 Chipset)
CPU: 2 x Xeon X5650 (total 12 cores at 2.66 Ghz)
GPUs: 2 x GTX 580 + 4 x GTX580 External
Memory: 96 GB / 12 DDR3 slots (192GB max)



PCIe 2 x16 (8 GB/s)

PCIe 2 x16 (8 GB/s)

2 x GTX 580
Measured bandwidth:
~ 5.7 GB/s to device
~ 6.3 GB/s from device

4 x GTX580

PCIe 2 x16 (8 GB/s): PCIe External

PCIe 2 x16 (8 GB/s): SAS Raid
Areca ARC-1880x

2xSFF8088

ARC8026

16 x A7K2000
~ 1.6 GB/s

PCIe 2 x4 (2 GB/s): 10 GBit Net
Intel 82598EB

LSDF
Large Scale Data Facility

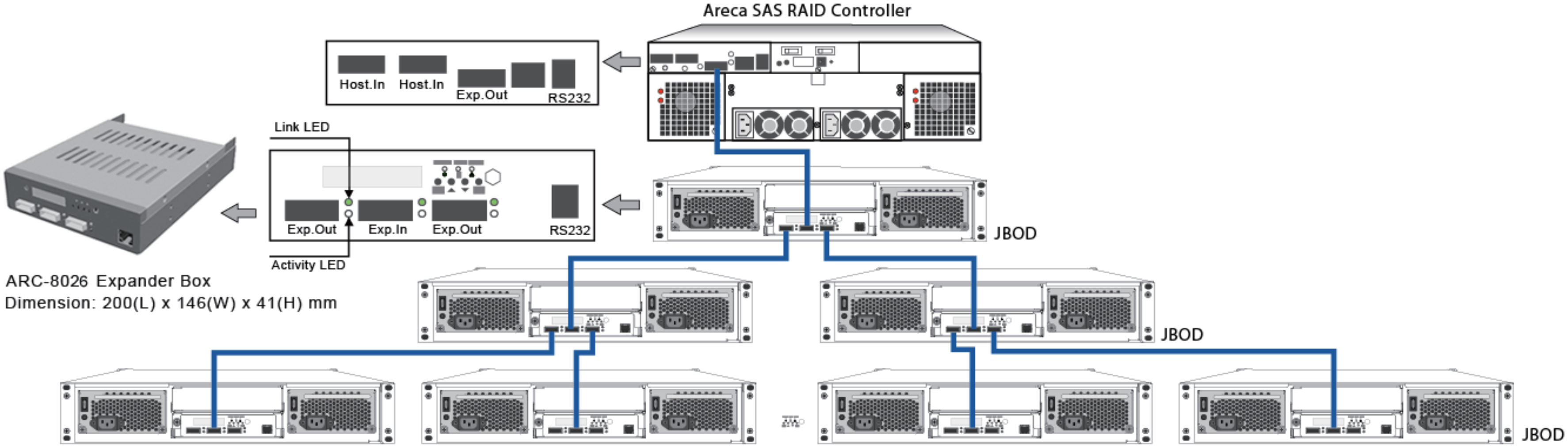
PCIe 2 x4 (2 GB/s): SSD Raid
4 x C300, 1420 MB/s

PCIe 1 x4 (1 GB/s): Frame Grabber
Silicon Software, 850 MB/s

Camera Link

PCO edge

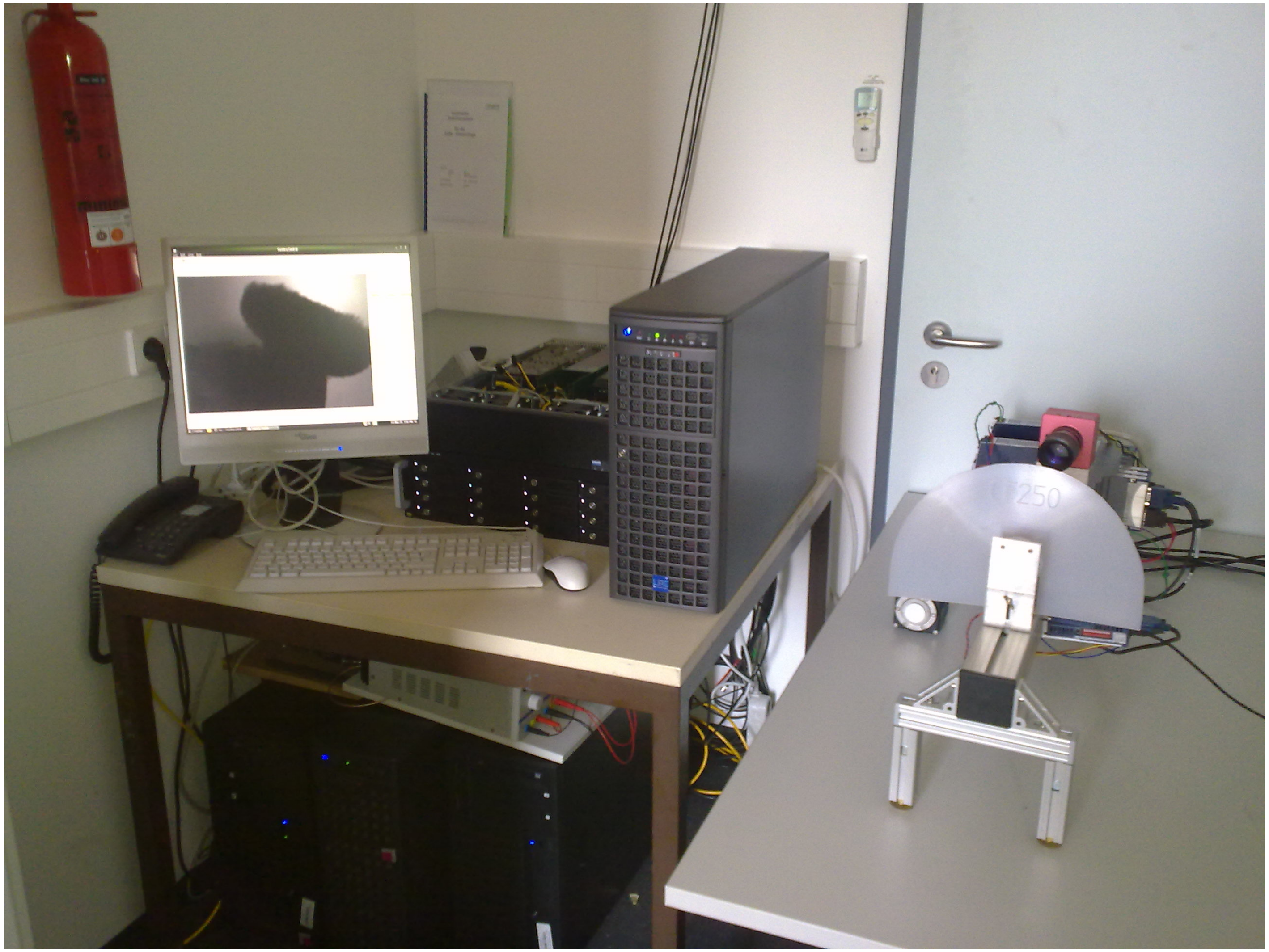
UFO Hardware: Scalable Storage



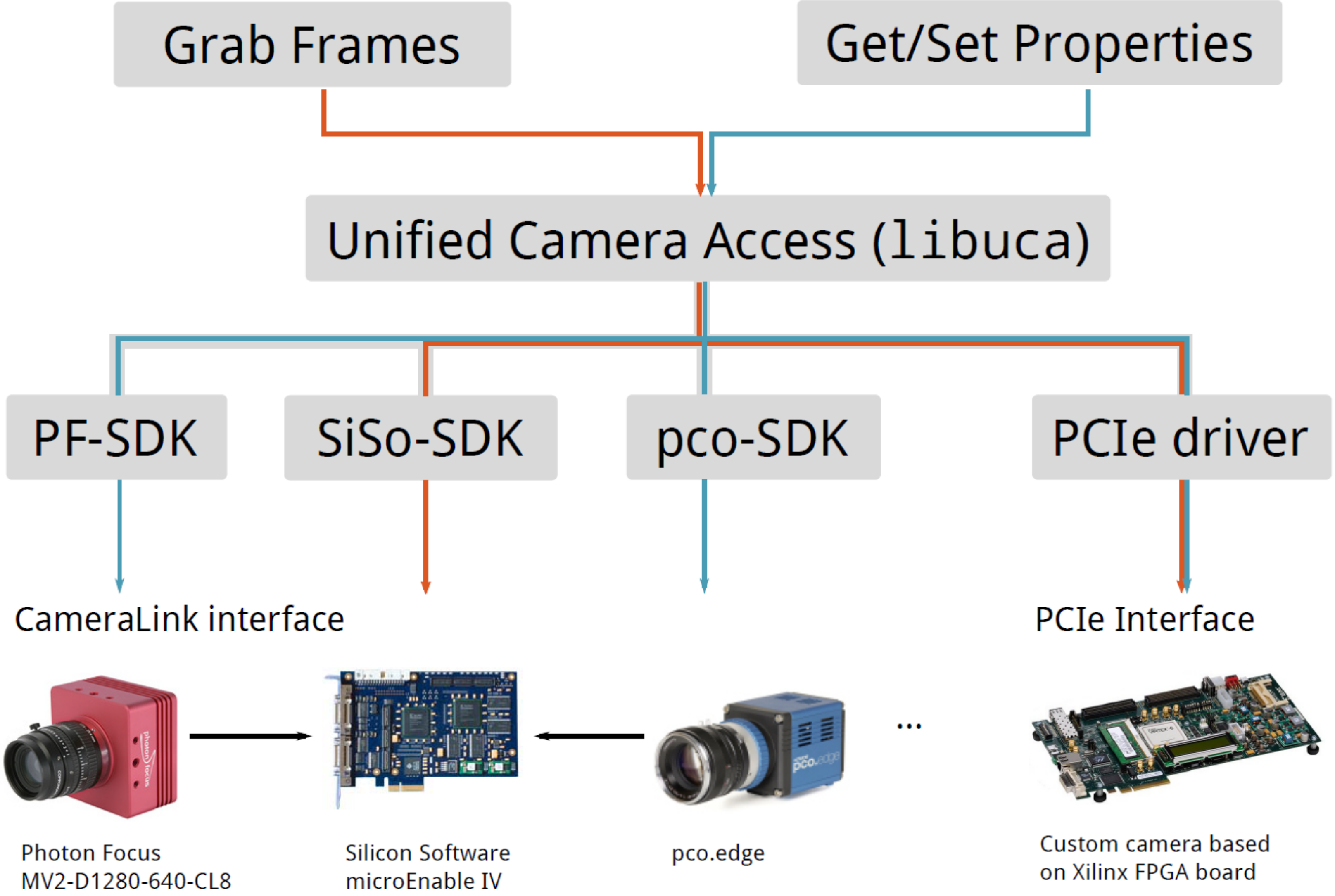
ARC 8026

- 12 – 24 SAS/SATA Hard Drives (per box)
- Up to 128 drives total
- Raid levels 0, 1, 3, 5, 6, 10, 30, 50, 60, JBOD
- Web based Raid Configuration

UFO Hardware: Test Setup



UFO: Unified Camera Access



Grab Frames

Get/Set Properties

Unified Camera Access (libuca)

PF-SDK

SiSo-SDK

pco-SDK

PCIe driver

CameraLink interface

PCIe Interface

...

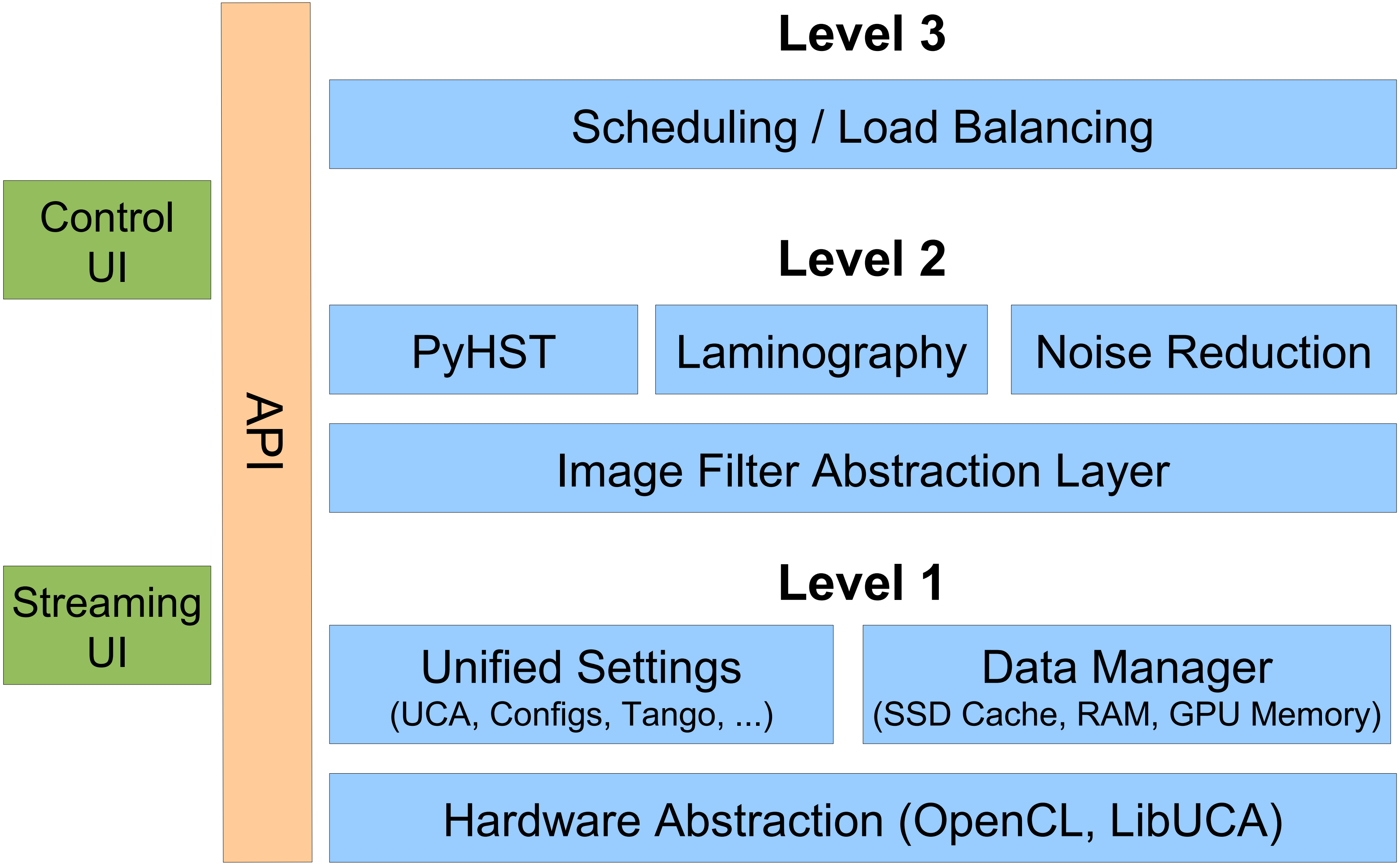
Photon Focus
MV2-D1280-640-CL8

Silicon Software
microEnable IV

pco.edge

Custom camera based
on Xilinx FPGA board

UFO Framework



Level 3

Scheduling / Load Balancing

Level 2

PyHST

Laminography

Noise Reduction

Image Filter Abstraction Layer

Level 1

Unified Settings
(UCA, Configs, Tango, ...)

Data Manager
(SSD Cache, RAM, GPU Memory)

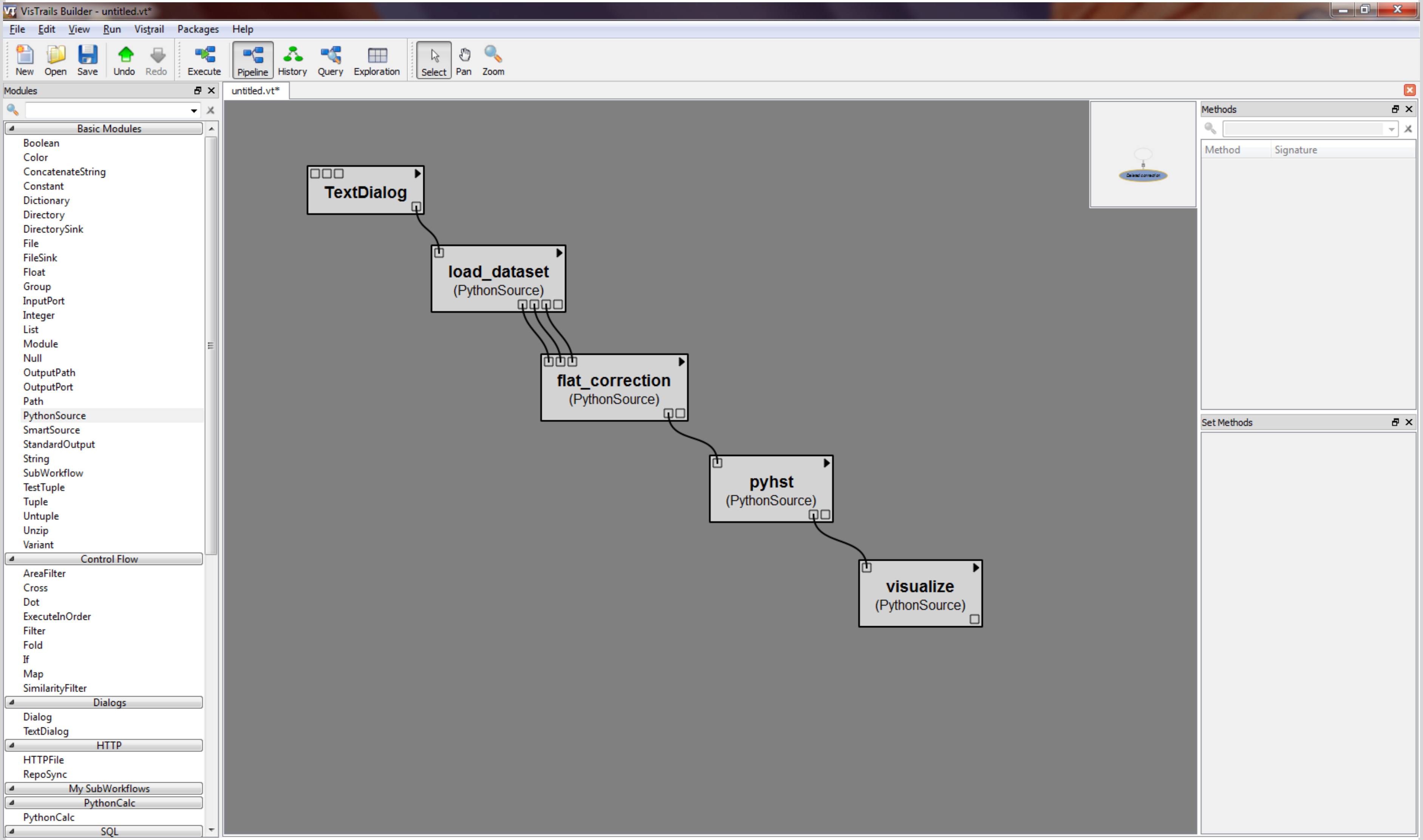
Hardware Abstraction (OpenCL, LibUCA)

API

Control UI

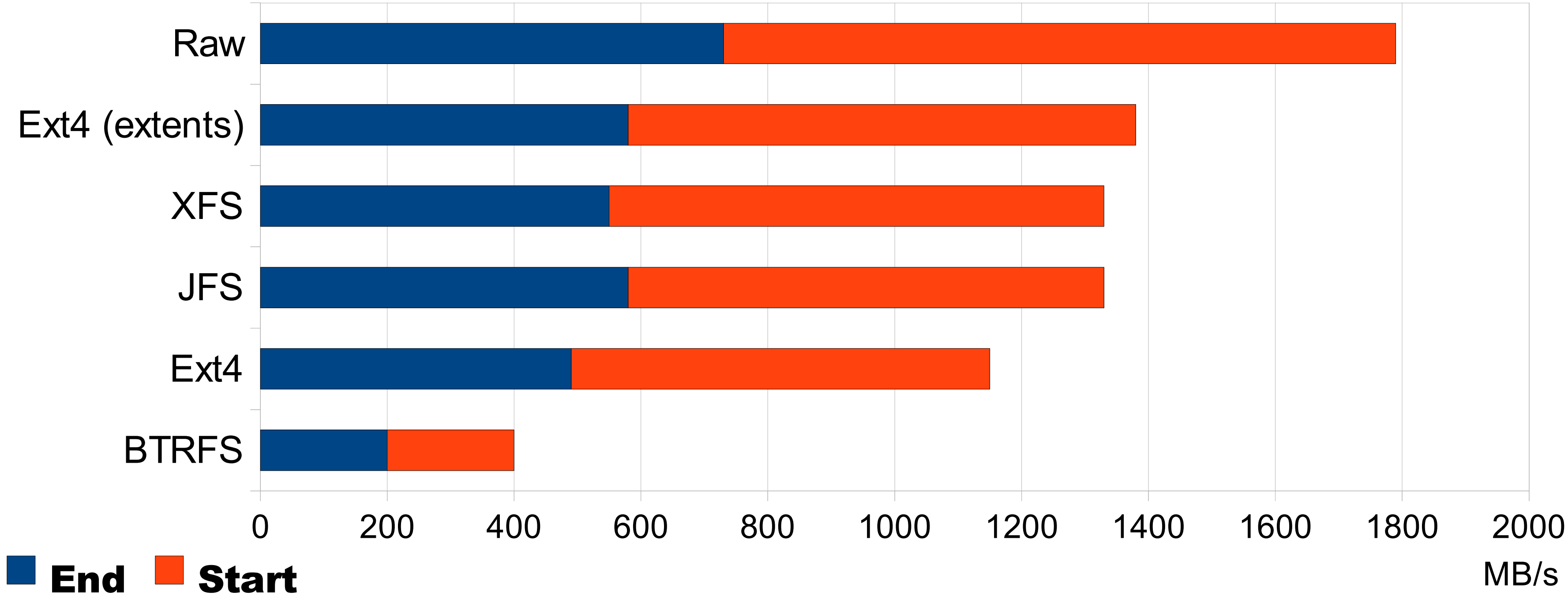
Streaming UI

UFO Framework: GUI concept



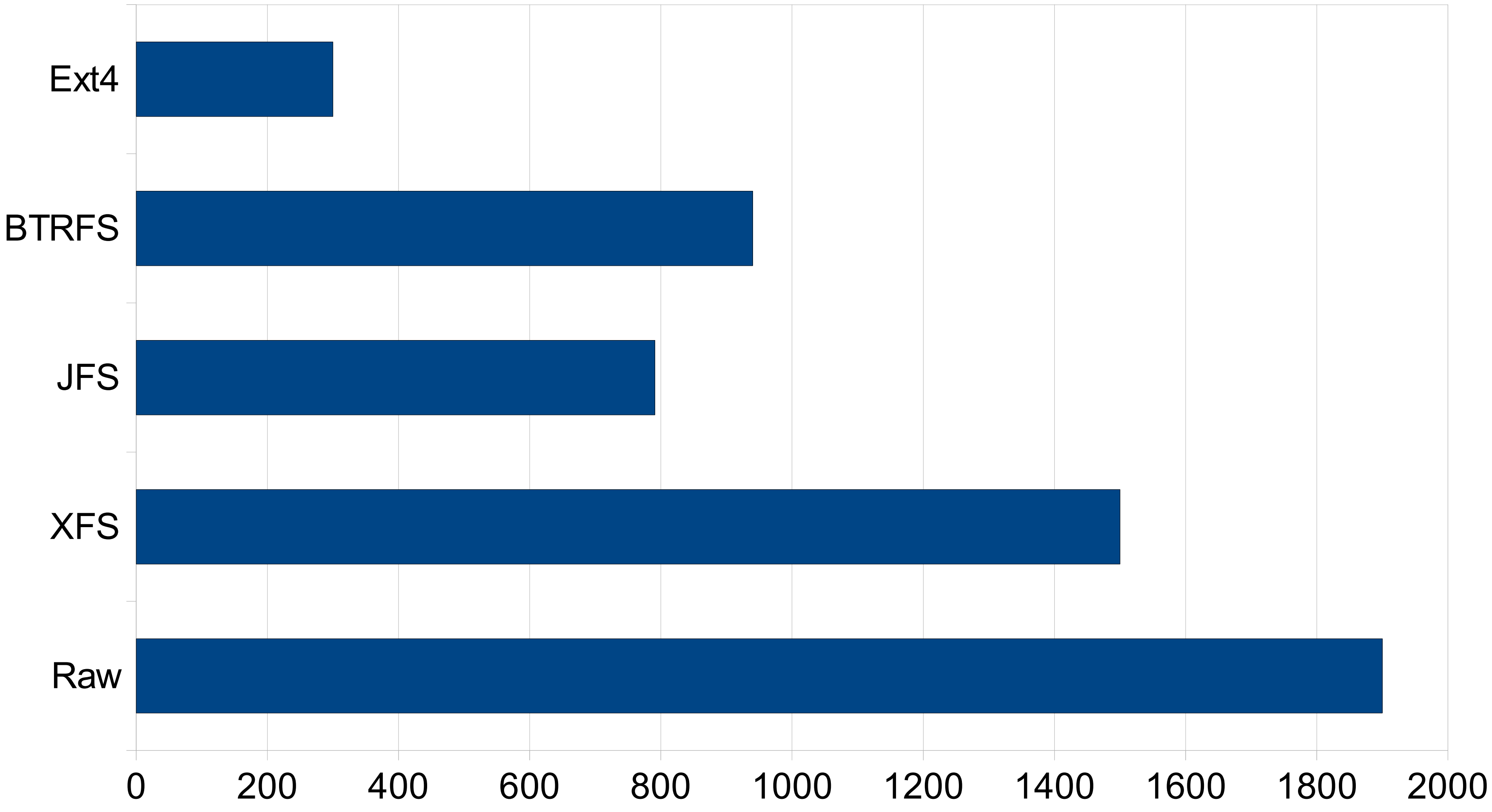
UFO: Handling Dense Data Streams

12 Hitachi A7K200 in Raid6, OpenSuSe 11.3 Kernel 2.6.34



- Ext4 performance drops significantly if free space comes to the end. XFS on other hand have spurious reductions of speed on empty disk.
- Fragmentation reduces performance

UFO: 16 A7K2000 in Raid-0



* Behavior of file system on high performance storage is unpredictable. We should use Raw disk as Ring Buffer.

Thanks

Thanks