Karlsruhe Institute of Technology

The UFO DAQ-Framework for High-performance Data Acquisition in Synchrotron Applications

A. Kopmann, M. Balzer, M. Caselle, S. Chilingaryan, T. Dritschler, T. Farago,
A. Herth, L. Rota, U.Stevanovic, M. Vogelgesang, M. Weber

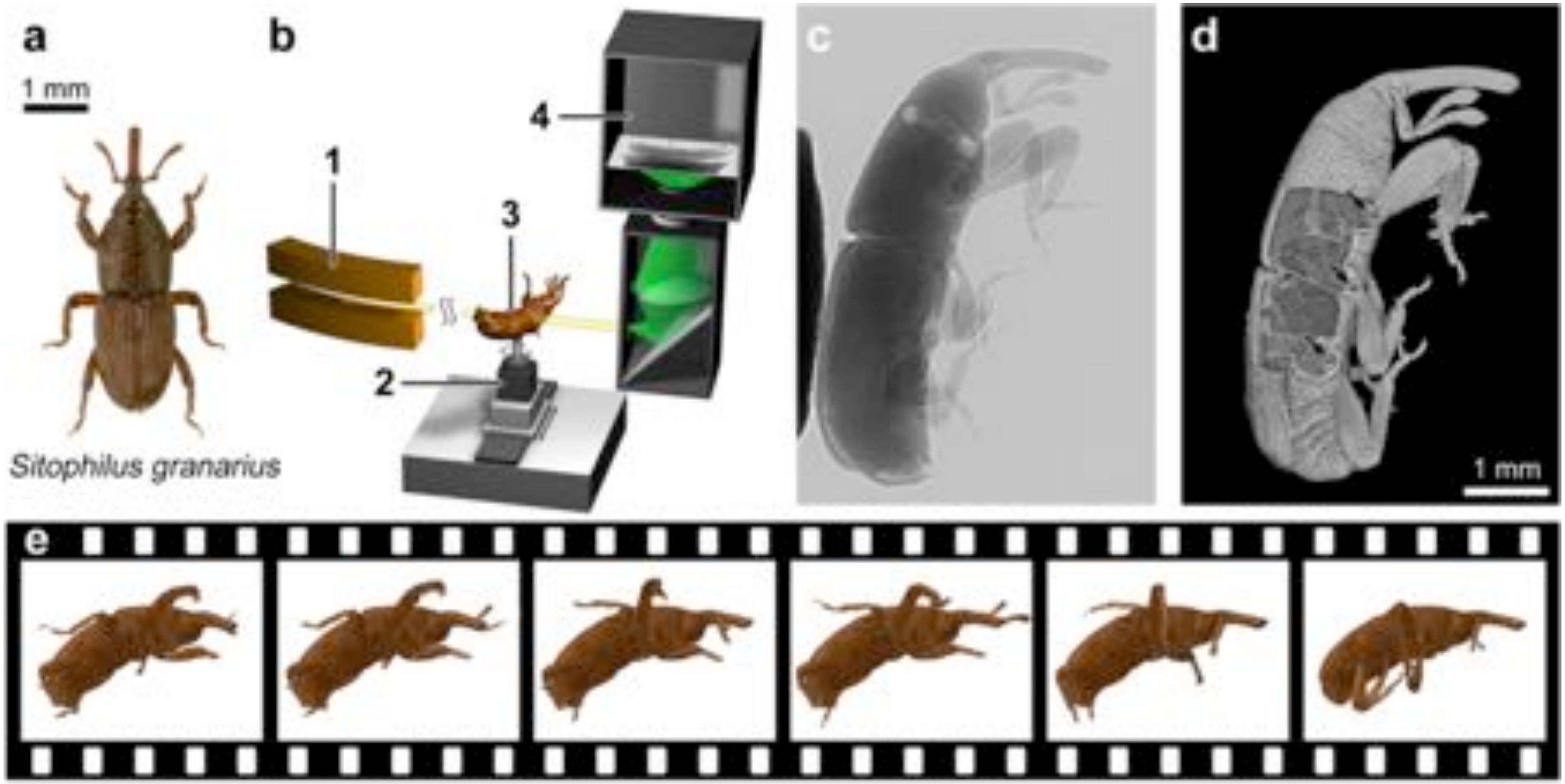
Institute for Data Processing and Electronics - IPE

Goals

- Micro X-ray imaging offers a broad field of applications:
  - Medical science
  - Non-destructive testing
  - Material and life science research etc…

- Are time-resolved measurements possible?

Application: 4D X-ray Cine-Tomography



dos Santos Rolo T et al. PNAS 2014;111:3921-3926

Goals

- Micro X-ray imaging offers a broad field of usage:
  - Medical science
  - Non-destructive testing
  - Material and life science research etc…
- Are time-resolved measurements possible?
  - 2D: radiography with > 1000 fps
  - 3D: tomography and laminography > 10 fps
    - 2D projections > 5000 fps
- DAQ Requirements:
  - Large (continuous) data rates
  - On-line data analysis + monitoring
  - Trigger + automatic control

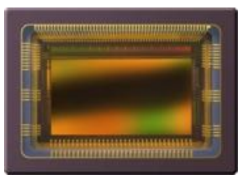
Motivation: Smart scientific cameras

- **Detectors for high speed imaging**
  - What is commercially available?

Data rates limit camera performance

Scientific sensor developments?

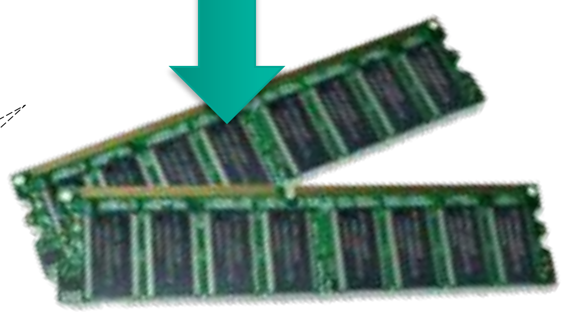
CMOS sensor



pco.dimax 4MP@1300fps
(7500MB/s)

Readout
Camera Link (250 / 850 MB/s)
USB3 (new: 500 MB/s)
GigE Vision (125 MB/s)

Internal Memory
e.g.: 36GB → 5s



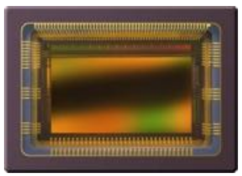
No online data analysis

Custom embedded logic is missing

Motivation: Smart scientific cameras

- **Goal: Modular DAQ Framework**

1. Modular sensor interface



e.g. for Helmholtz Detector and Systems Platform (DTS)

2. Embedded logic & Control
e.g. Smart Phase Contrast (w HZG)

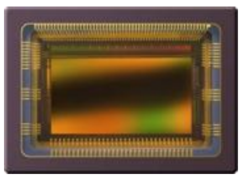
3. Full streaming readout
e.g. using PCIe, IB

4. Real-time data processing
e.g. with GPUs

Motivation: Smart scientific cameras

- **Components:**

1. Modular sensor interface

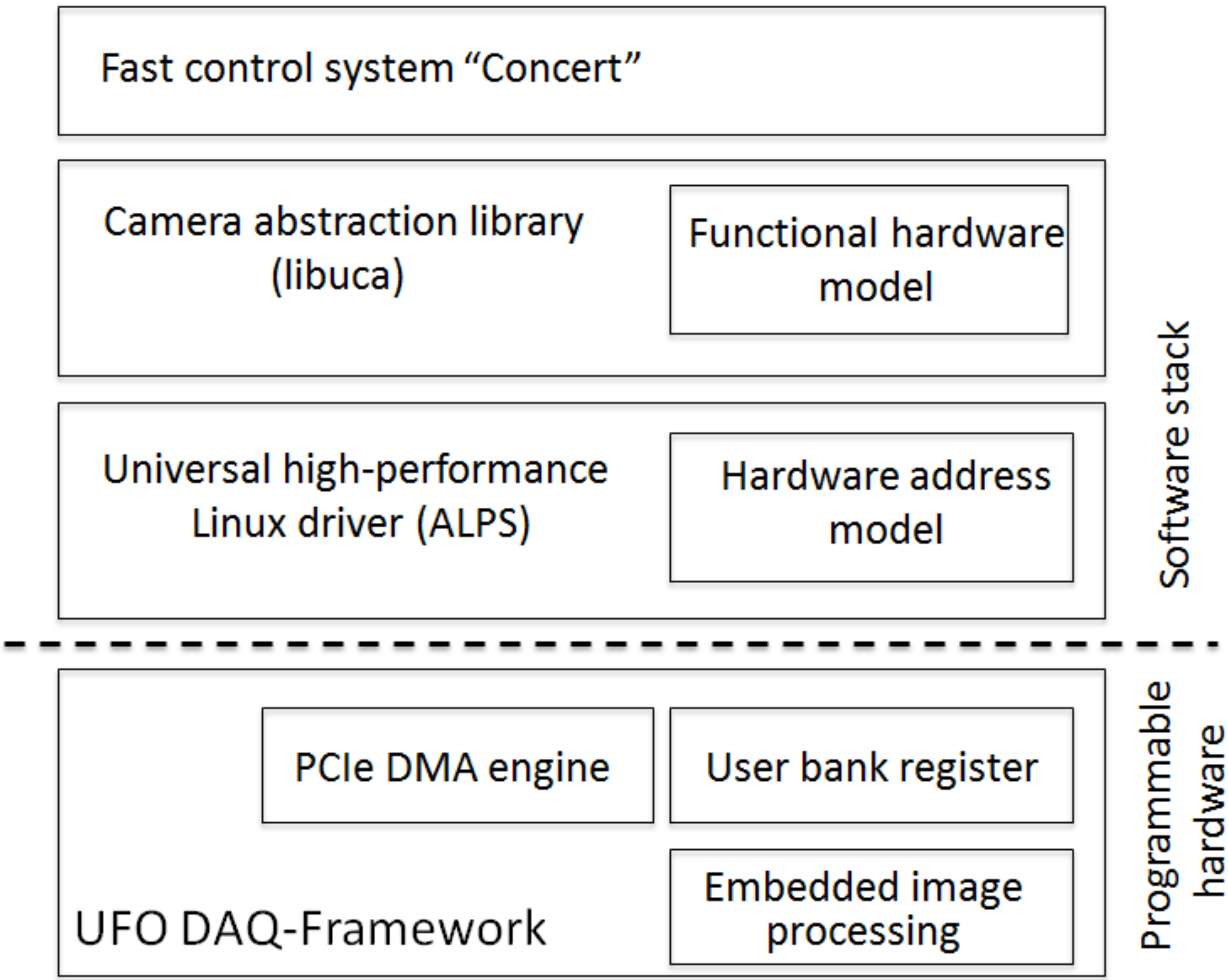


2. Embedded data processing
- Fast control system

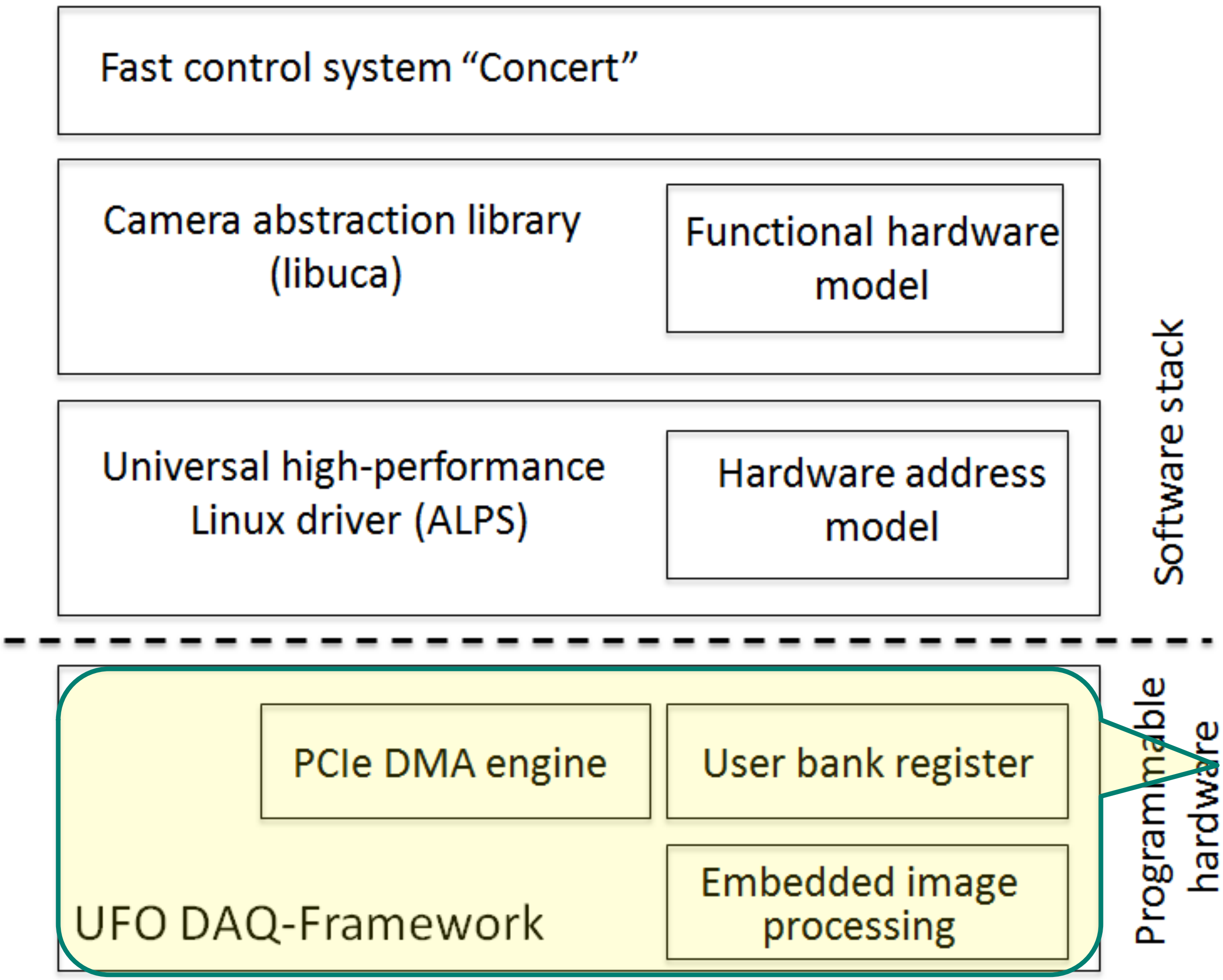
3. Readout
- DMA engine
- Linux drivers

4. GPU Computing
- Camera abstraction
- Programming environment
- Optimized algorithms

DAQ framework

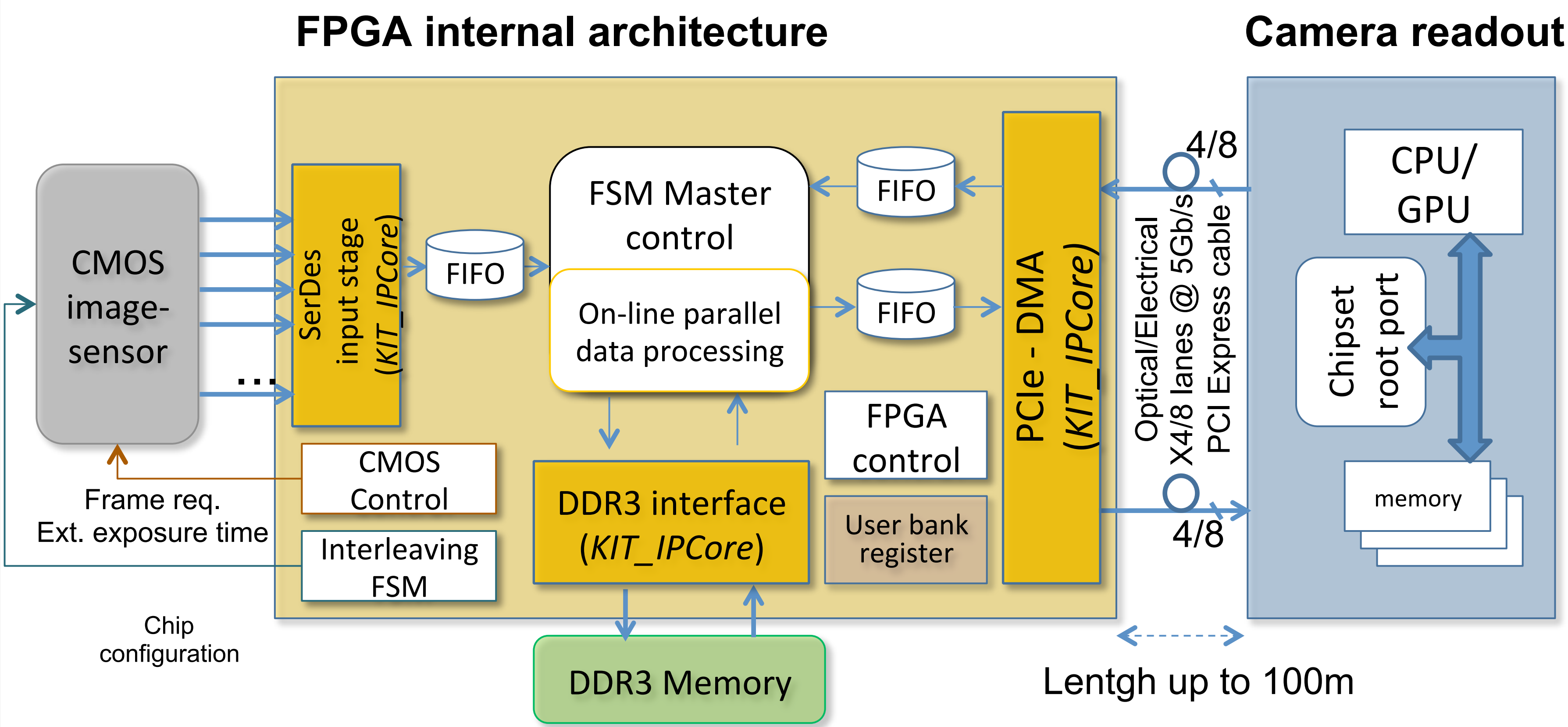


DAQ framework



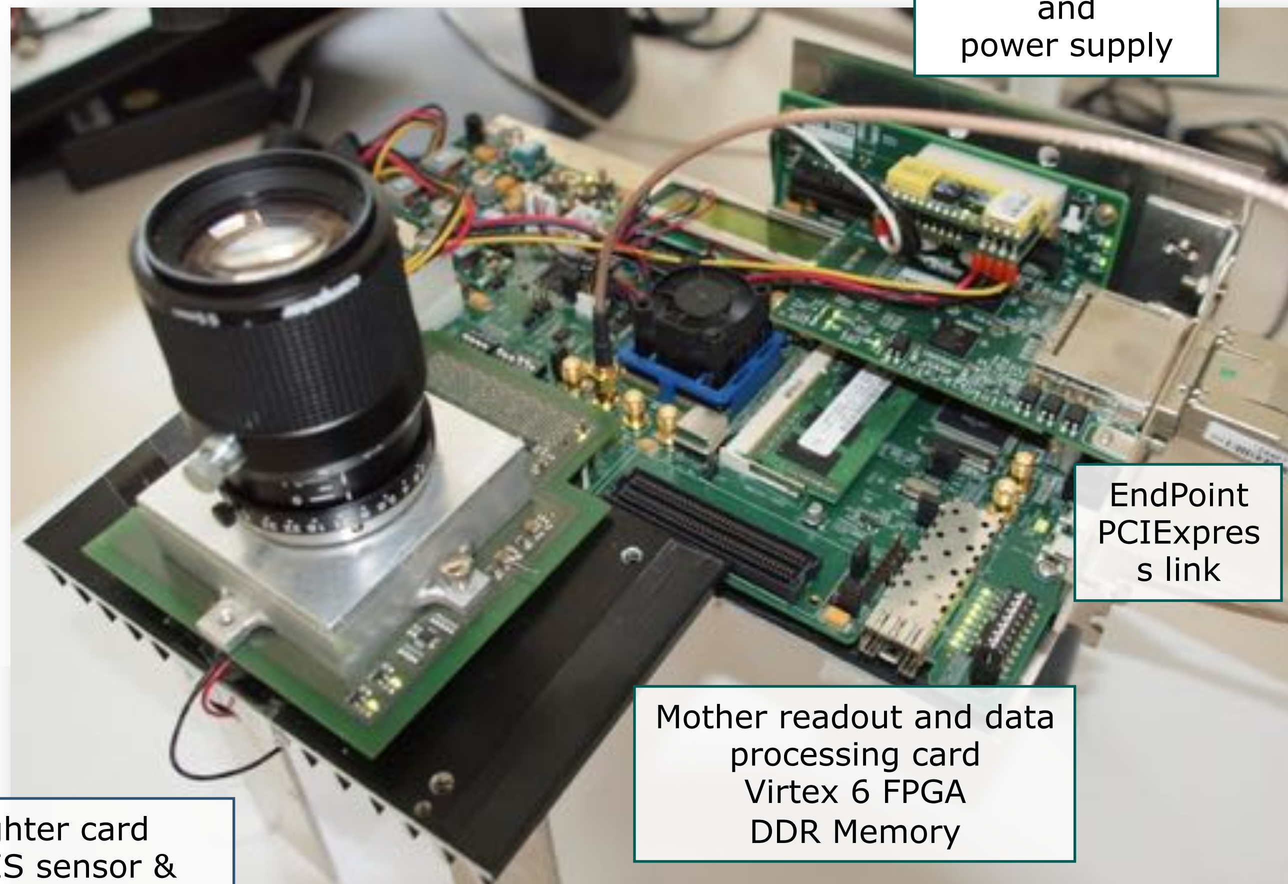
- Smart camera platform
- Image based trigger
- PCIe DMA engine for fast data streaming

Smart camera platform – architecture

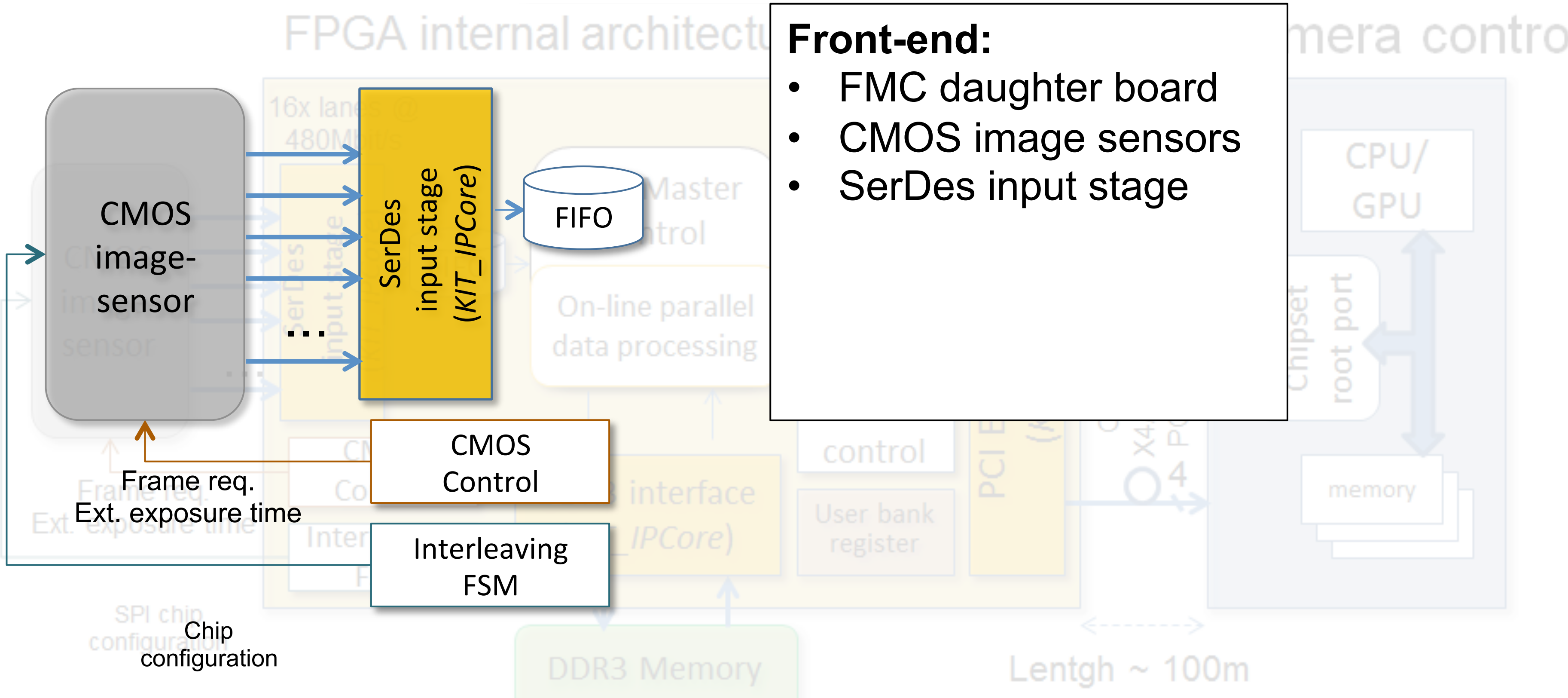


First high smart camera with CMOS sensor

- Low noise daughter PCB design
- Cooling system (silicon sensor) controlled by FPGA.
- Remote camera control (i.e. external exp. time, frame req, enable readout, busy, etc.)



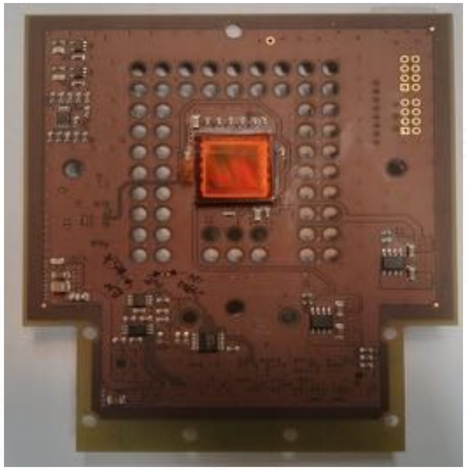
Smart camera platform – front-end



Front-end:

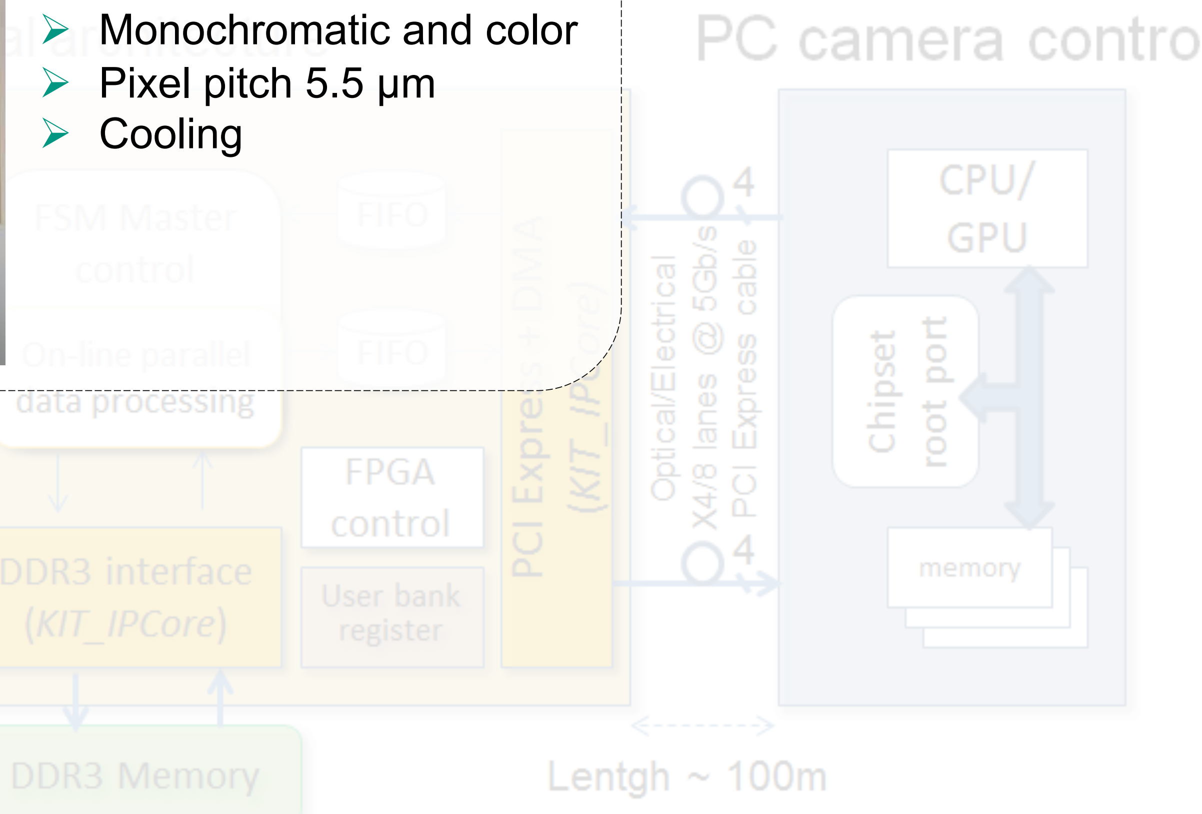
- FMC daughter board
- CMOS image sensors
- SerDes input stage

Smart camera platform – sensors

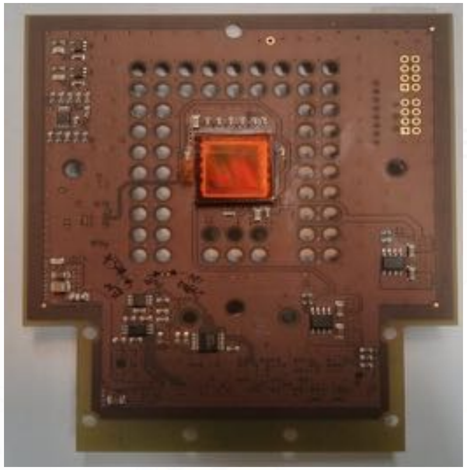


CMOS sensors:

- 4 MPixels and 2.2 MPixels
- Monochromatic and color
- Pixel pitch 5.5 µm
- Cooling

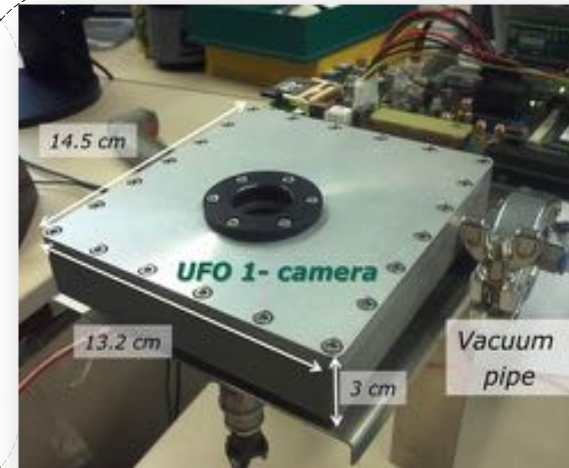


Smart camera platform – sensors

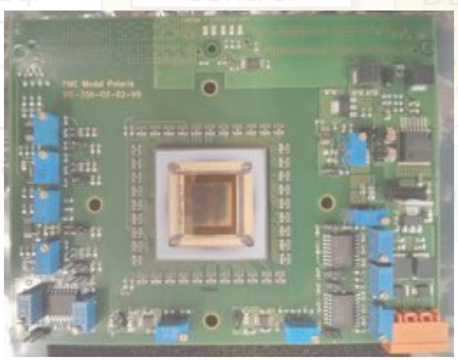


CMOS sensors:

- 4 MPixels and 2.2 MPixels
- Monochromatic and color
- Pixel pitch 5.5 µm
- Cooling



- Vacuum environment
- Peltier cell: -5 °C with 30 % of the power

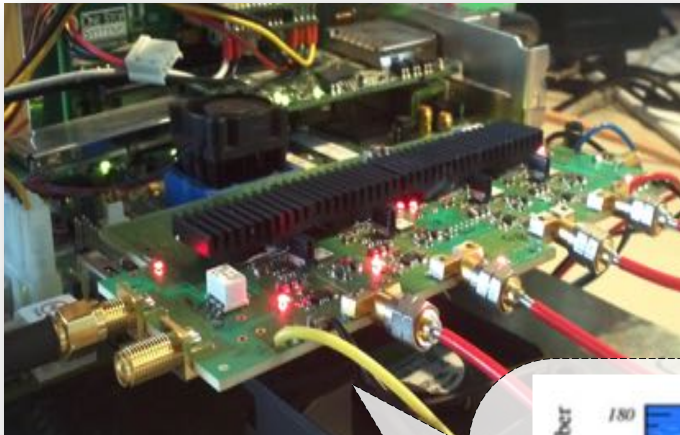


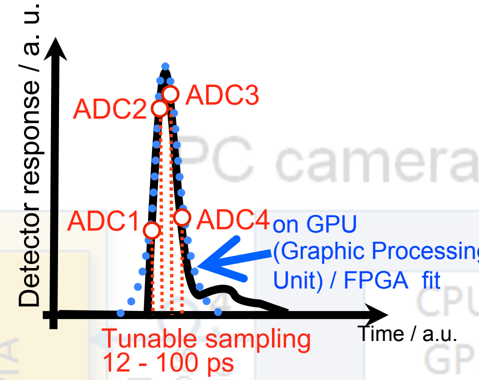
In progress:
High frame rate CMOS sensor

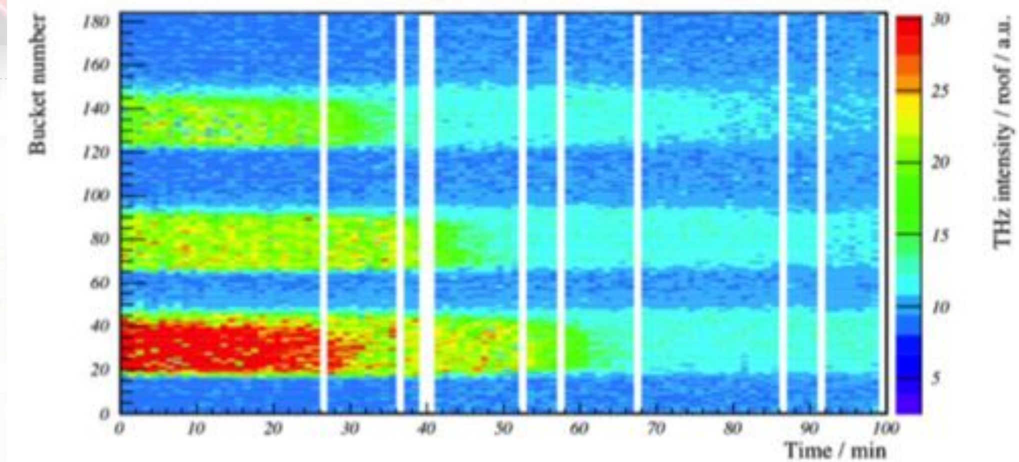
➡ 5k fps @ 1MP

Karlsruhe Institute of Technology

A smart non-camera application





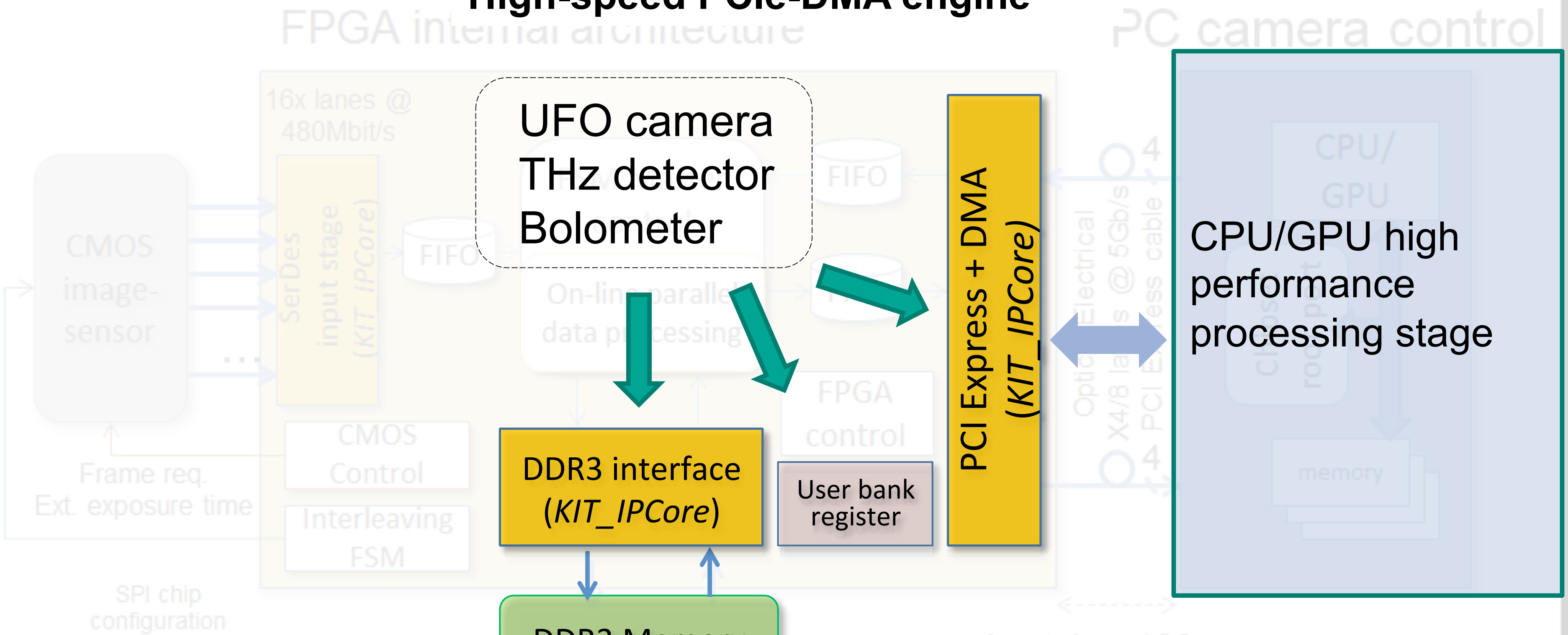


Ultrafast digitizer for beam monitoring:

- Programmable picosecond sampling
- Data rate 4x500MS/s → 3 GB/s
- Pulse reconstruction

Smart camera platform – streaming readout

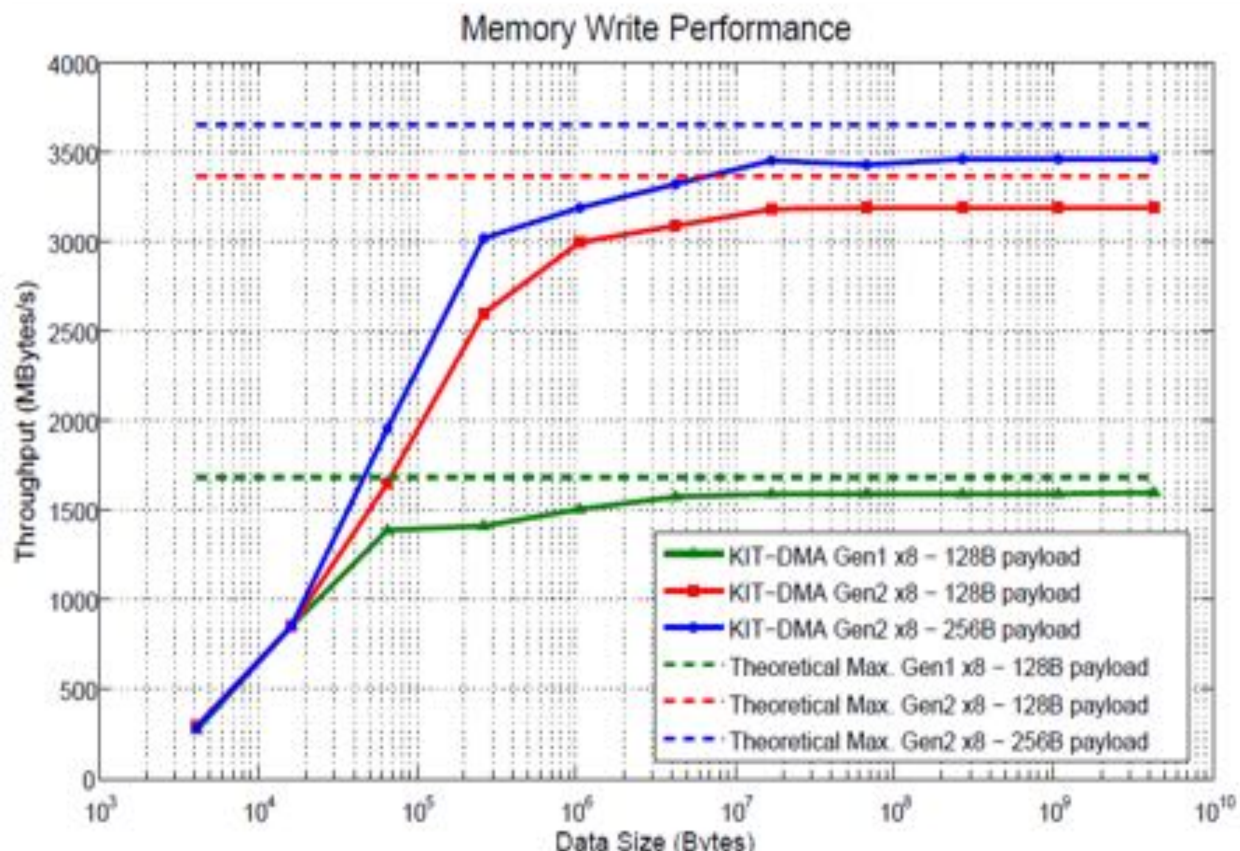
High-speed PCIe-DMA engine



Gen2 x8 – 3.4 GB streaming

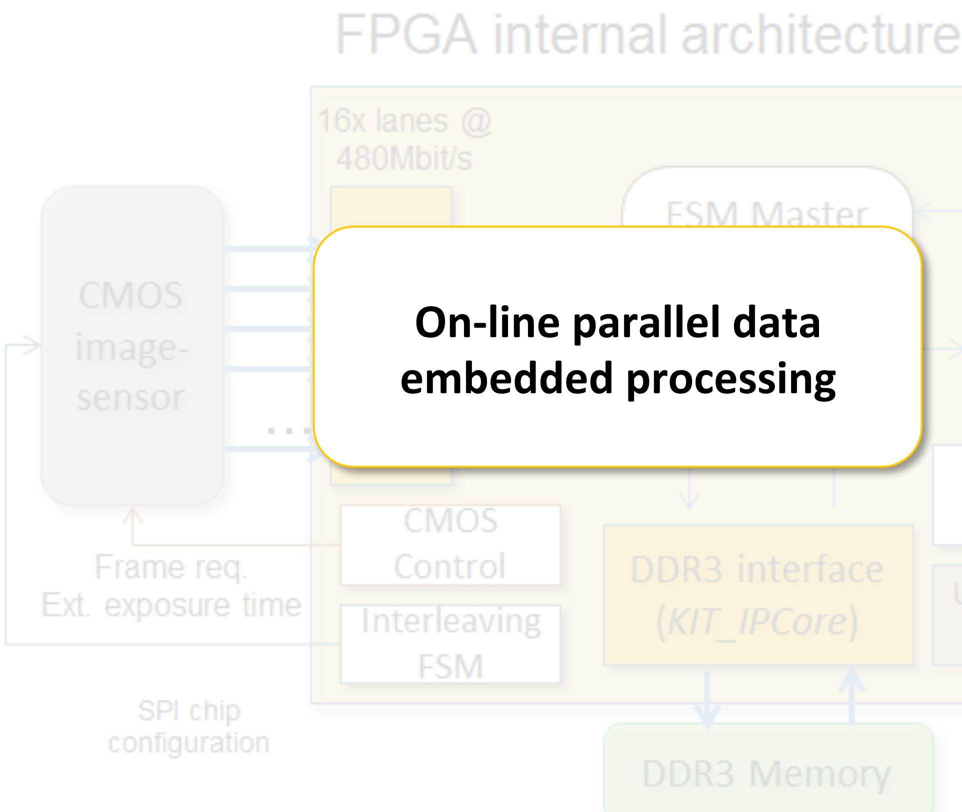
Smart camera platform – streaming readout

High-speed PCIe-DMA engine



CPU/GPU high performance processing stage

Smart camera platform – embedded processing



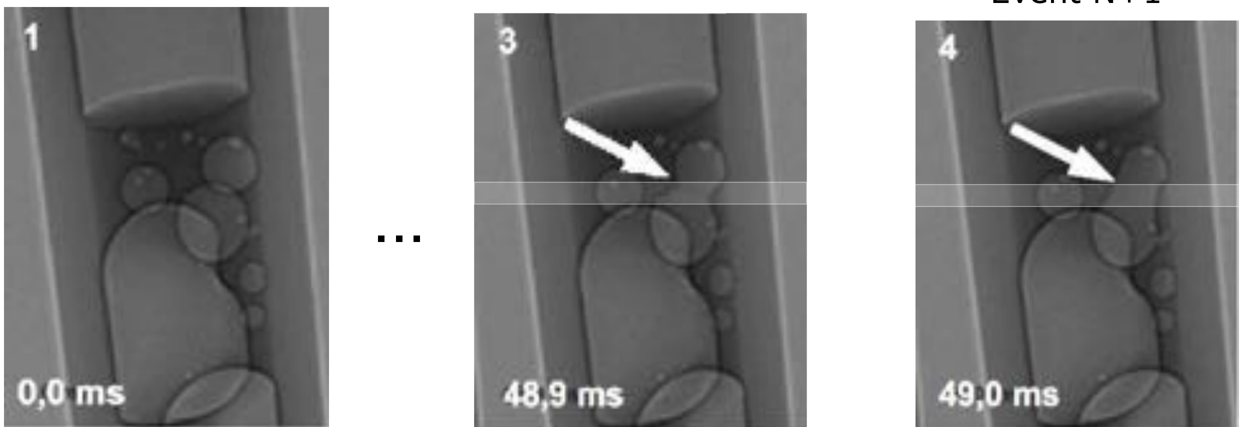
Possible applications:

- (Simple) image processing
- Camera internal control:
  - Trigger
  - Selective readout
  - Exposure/gain control
- Control of sample environment
  - Automation of complex procedures
  - Feedback control

Example: Image based self-trigger

Application: Bubbles merging in gelatinous agar

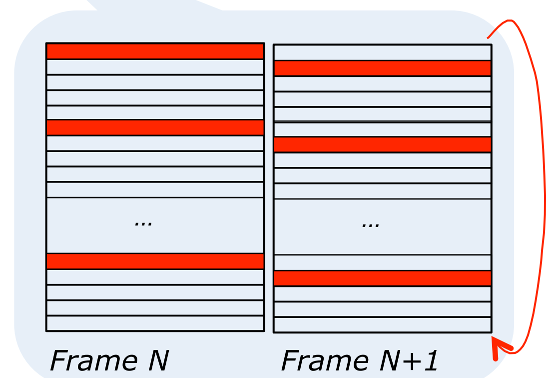
Reference frame … Event N … Event N+1



Most of the time
no changes

Rare fast event
@ 10 KHz

Only 2% of rows
have changed



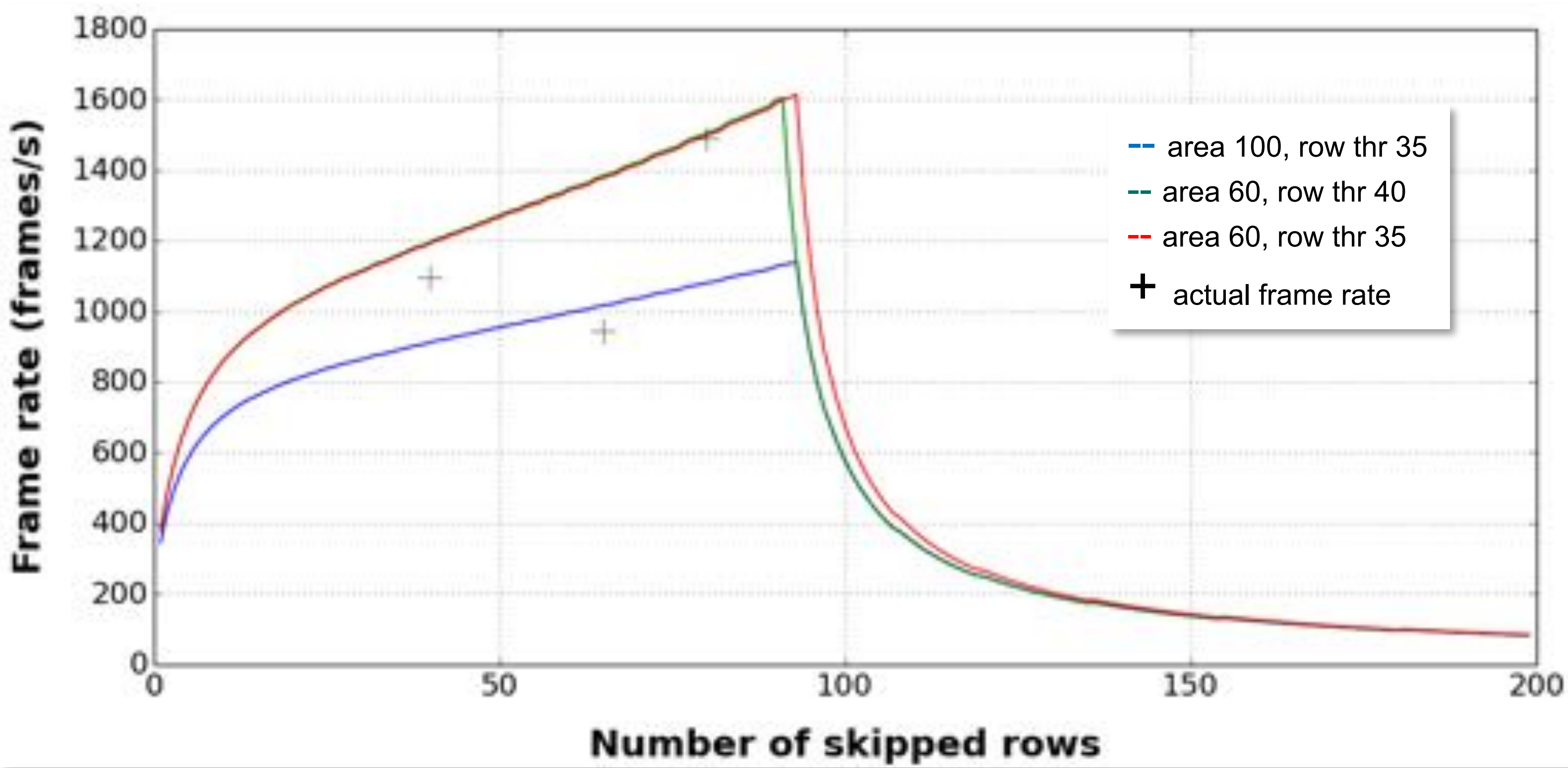
Only small and a-priori unknown changes

→ Need for adaptive frame rate

Solution:

Interleaving readout
and event-trigger

Fast reject efficiency



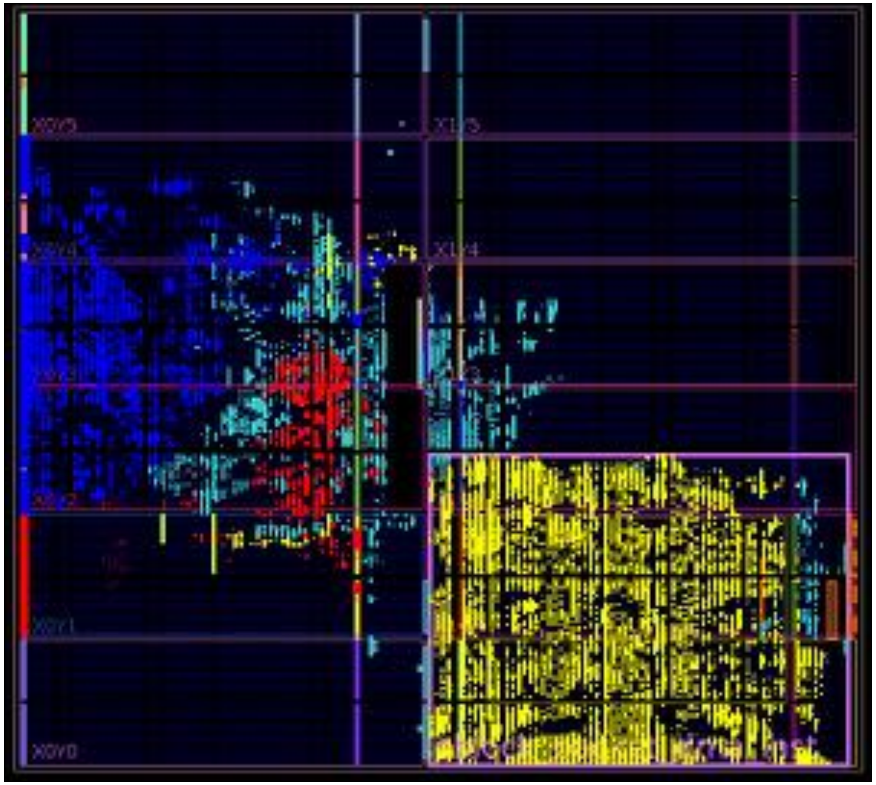
- Frame rate depends on the used parameters and the event size
  - Effective frame rate can be more than five times higher than nominal
- Concert can automatically set initial parameter values with the desired maximum speed, object size detection, and minimum noise influence

Programmable hardware

- The DAQ framework is fully programmable
  - Based on FPGAs
  - Three IP cores
  - Custom “smart” logic

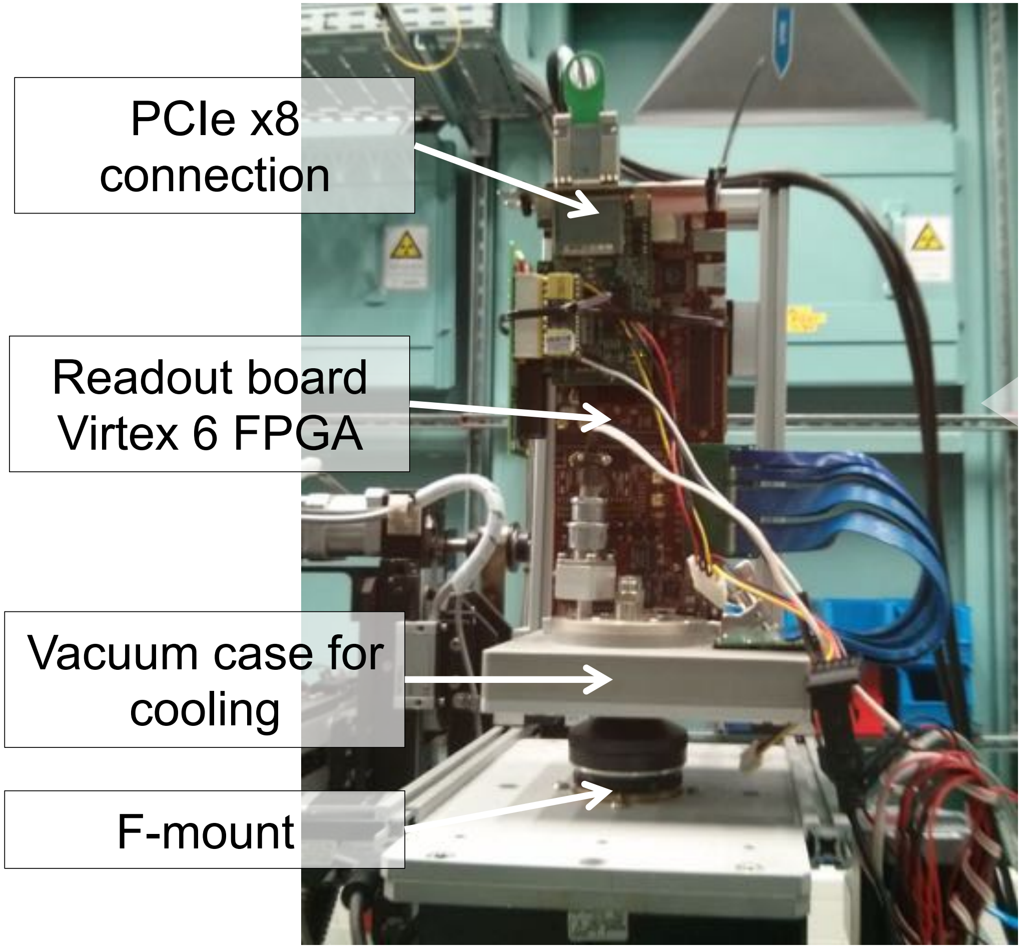
UFO camera FPGA utilization:

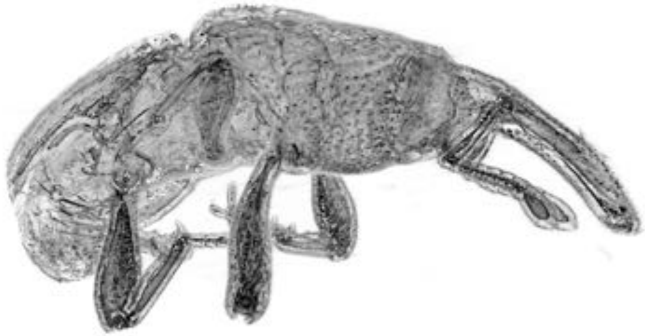
- Yellow: PCIe DMA
- Dark Blue: DDR3 interface IP
- Red: Input stage
- Bright blue: Control and processing logic



Xilinx XC6VLX365TFF1759-2

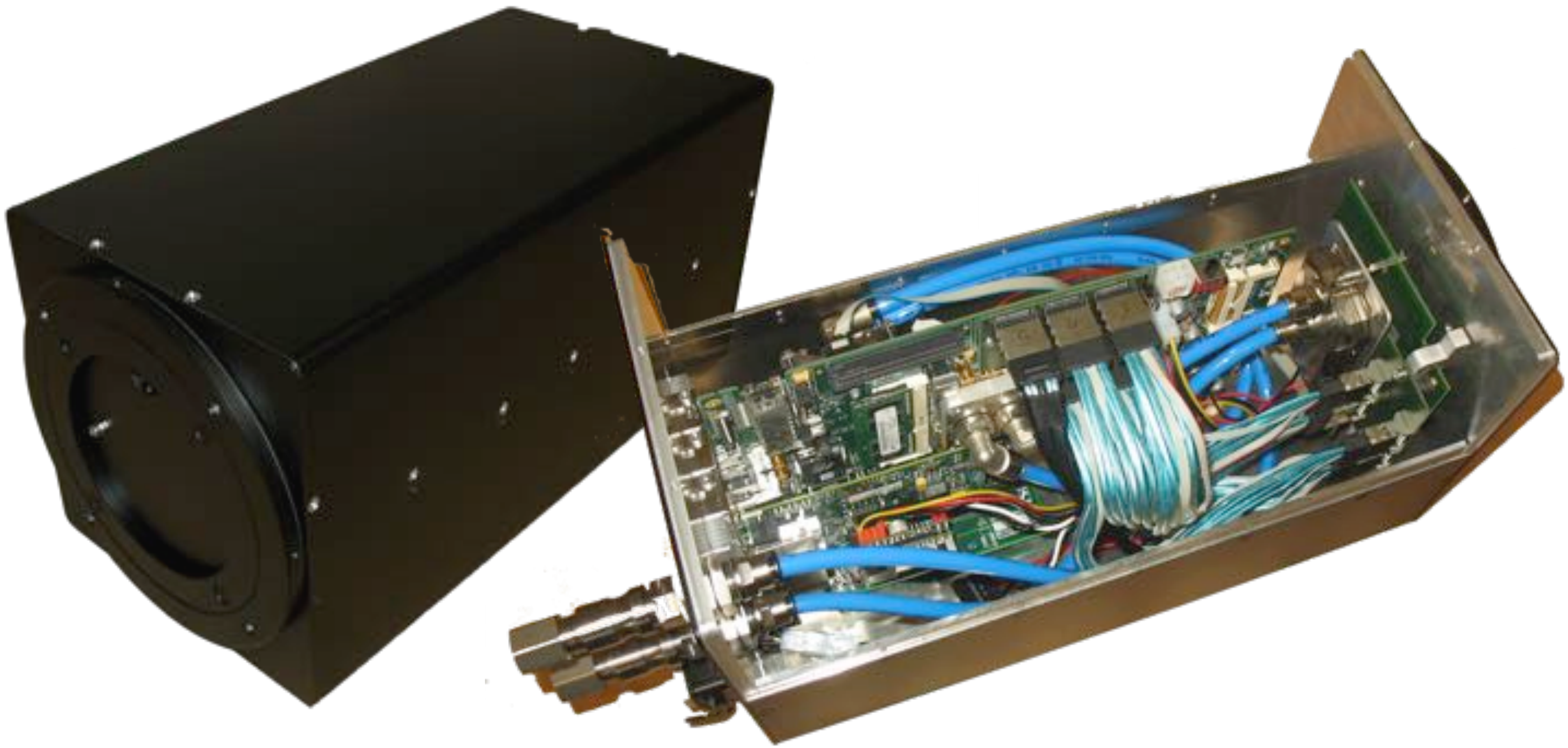
UFO camera at the IMAGE beamline



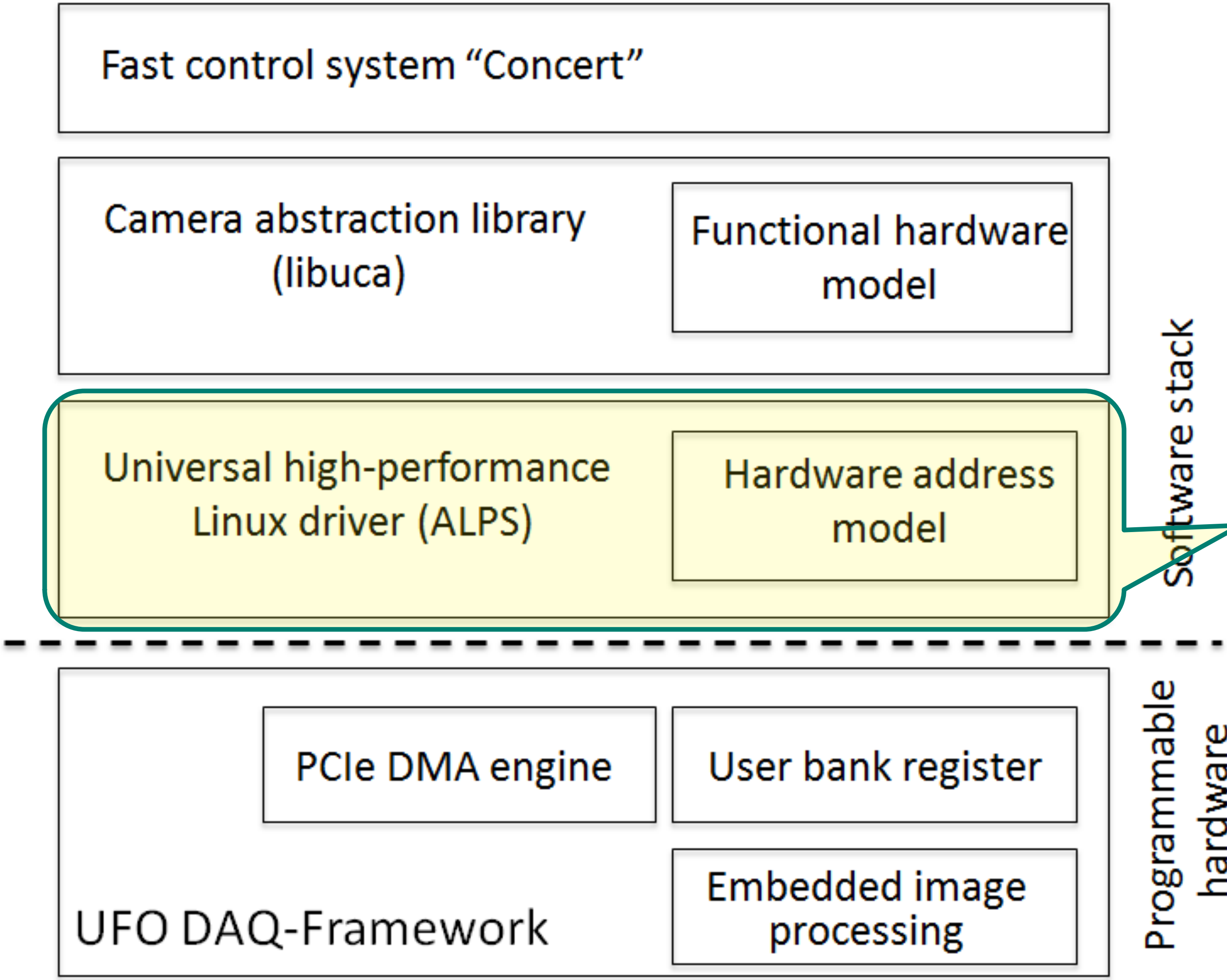


Smart phase contrast camera

- Goals:
  - Automatic grating control
  - Reconstruction (with GPU)



DAQ framework



- Optimized for high throughput
- Configurable register model
- Comfortable usage

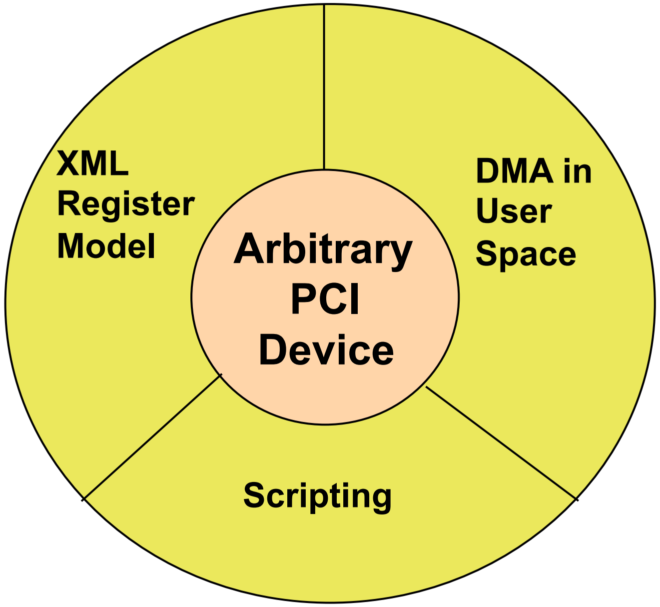
ALPS

Advanced Linux PCI Services

A reusable components for custom PCI electronics

Motivation

- Synchronization Software and Hardware development
- Easy hardware debugging
- Keeping drivers up to date with latest Linux kernels
- Multiple common components



Components

- PCI driver
- Register Model
- DMA Engine
- Custom Event Plugins
- Web API (planned)

PCI + PCIe interface in user-space

Scripting
Bash, Perl, Python

LabVIEW
Control System Integration

pcitool
Command-line tool

GUI
User Interface in Python/GTK

Web Service
Remote Programming Interface

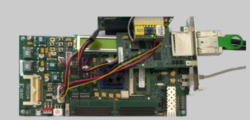
PCILIB
User-space Layer

VFIO + UIO

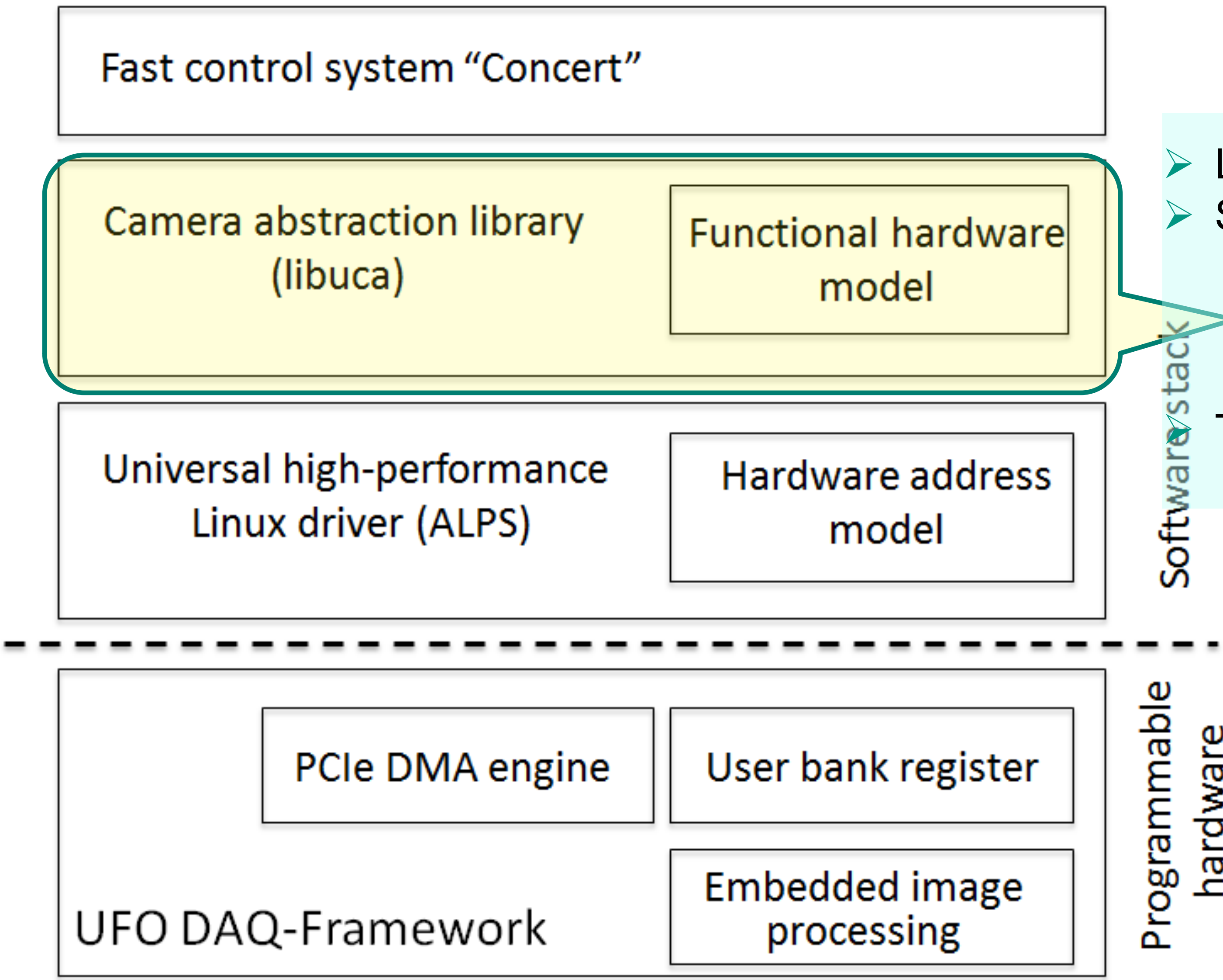
- PCI Bar mapping
- DMA Memory Management
- IRQ Handling
- Interlocking

PCI / PCI Express Board

(Variety of FPGA Boards: IPE Camera, etc)

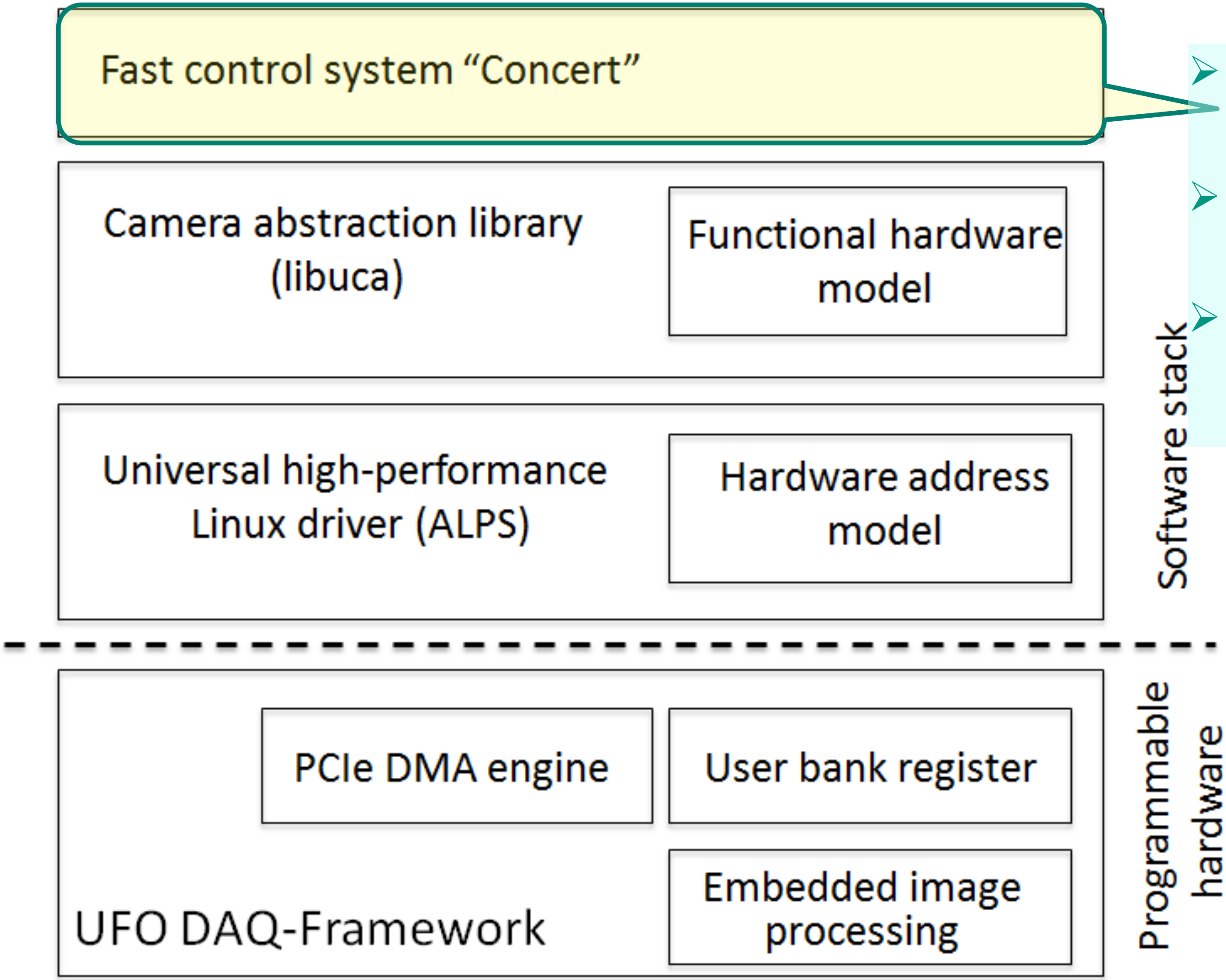


DAQ framework



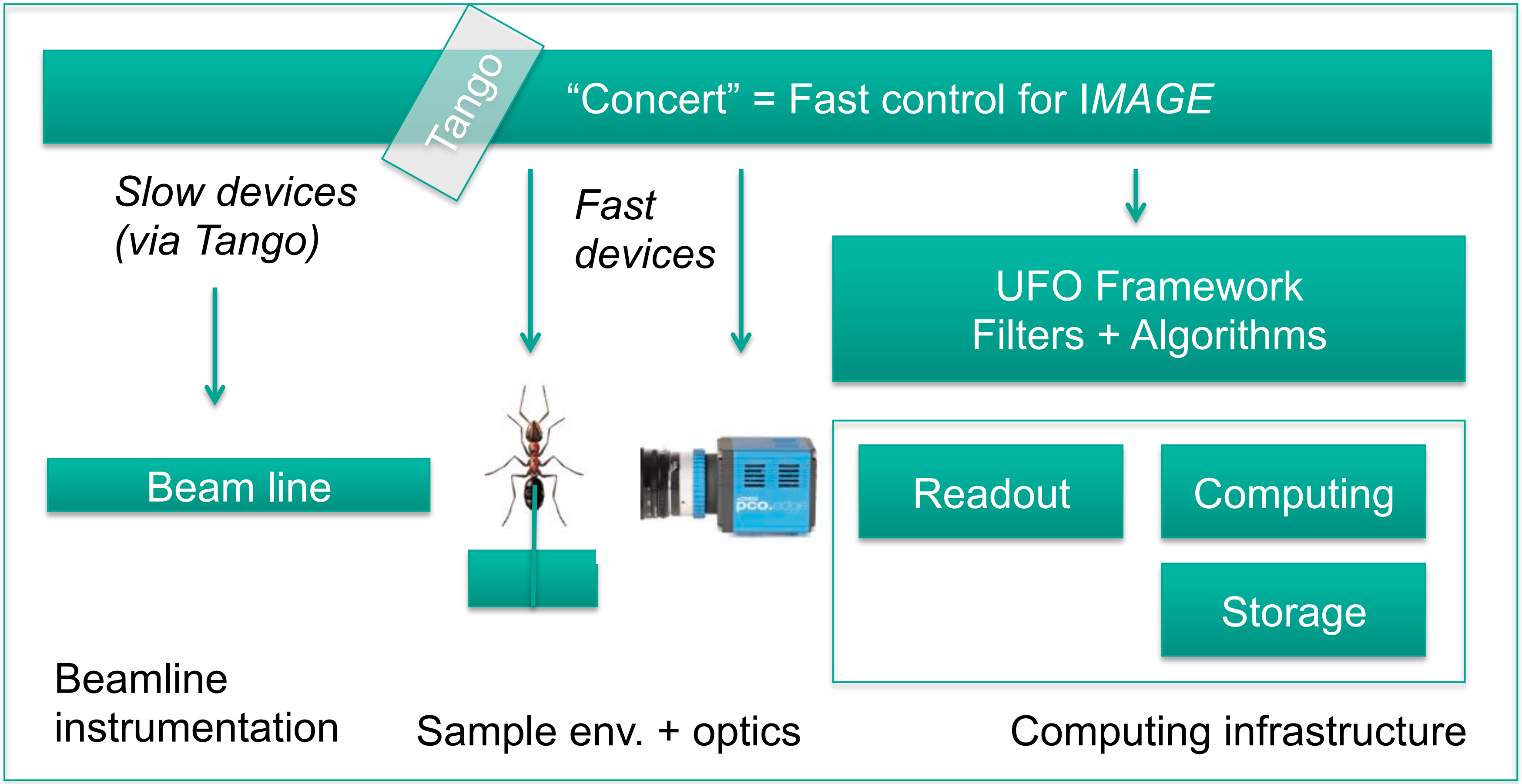
- Linux (64bit)
- Supported cameras:
  - pco
  - photon focus
  - UFO camera
- Tango driver

DAQ framework



- Seamless device integration
- High performance computing
- Image-based feedback control

“*Concert*” = Fast control system for IMAGE



Concert details

- Python based
- Underlying hardware-specific details are abstracted:
  - e.g. *Camera* class provides *grab* method to acquire a frame:

```
frame = cam.grab()
```

  - e.g. setting the exposure time:

```
cam.exposure_time= 2*q.milliseconds   # Req. units
```

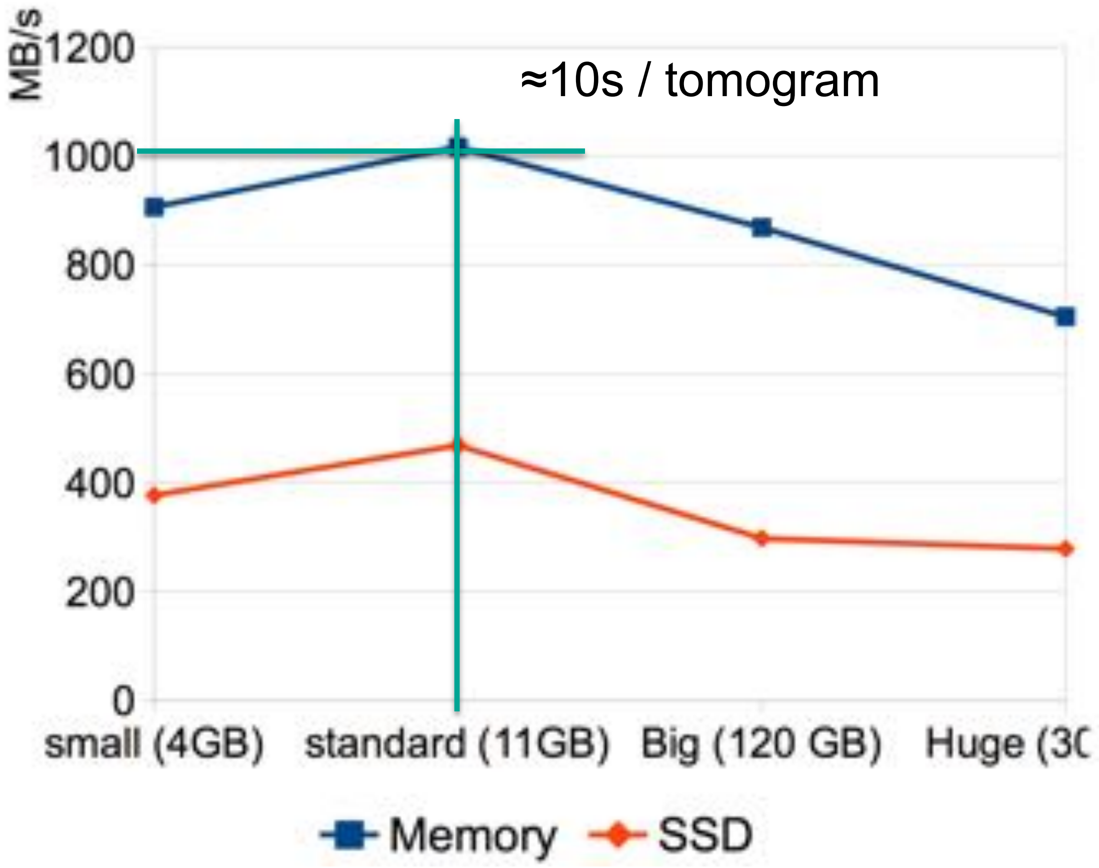
- Asynchronous device access:
  - Parameters access and device methods wrapped in *future* objects
  - Methods to query state, final result, callback

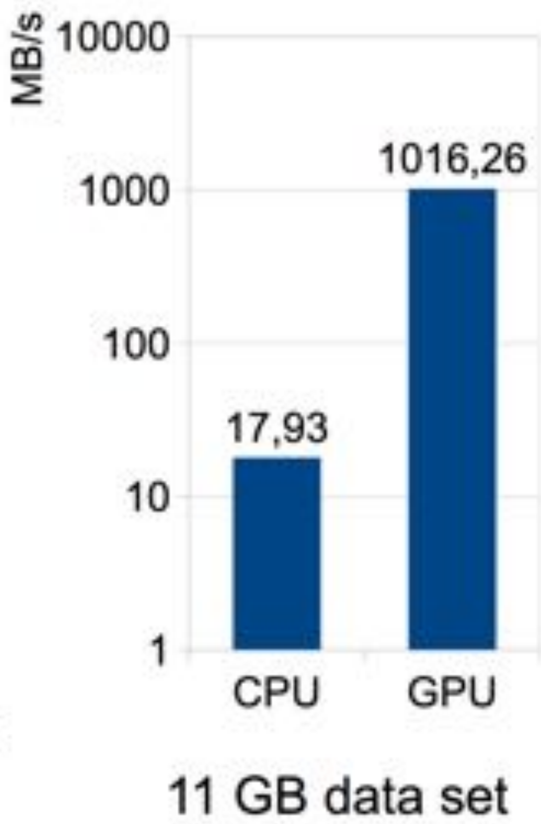
```
# Synchronous access
motor.position = 1 * q.mm

# Asynchronous access returns a future
future = motor.set_position(1 * q.mm)
future.wait()
```

```
class Motor(Device):
    @async
    def move(self, delta):
        self.position += delta

motor.move(-2 * q.mm).wait()
```

Filtered back projection performance

Karlsruhe Institute of Technology





GPU: 4 x GTX590 , 8 cores
CPU: 2 x Xeon X5650, 12 cores
(both from 2011)

FPD Computing throughput
== Readout rate ✔

UFO parallel computing framework

How to support code development for GPUs?

Requirements:

- Processes data streams (usually 1 to 4 dimensional floating point data)
- Detect and use all hardware resources

Developer:

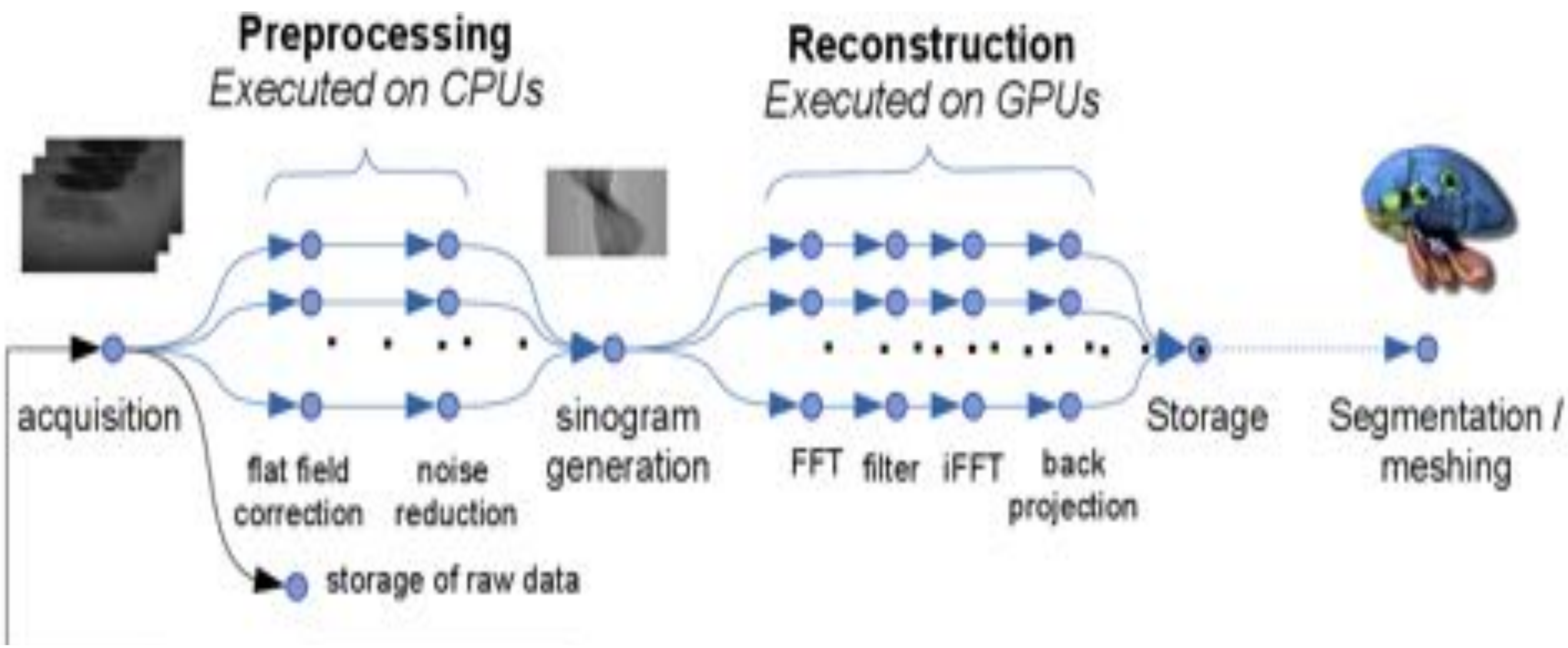
- Hides parallelization and concurrency details
- Management of memory transfers
- Multiple implementations (e.g. for CPU + GPU)
- Automatic scheduling

User:

- Simple end-user interface
  - GUI + Scripting
- Modular algorithm design

Realization

- Implemented in OpenCL
- Define algorithms as self-contained tasks
- Specify data flow as edges in a graph

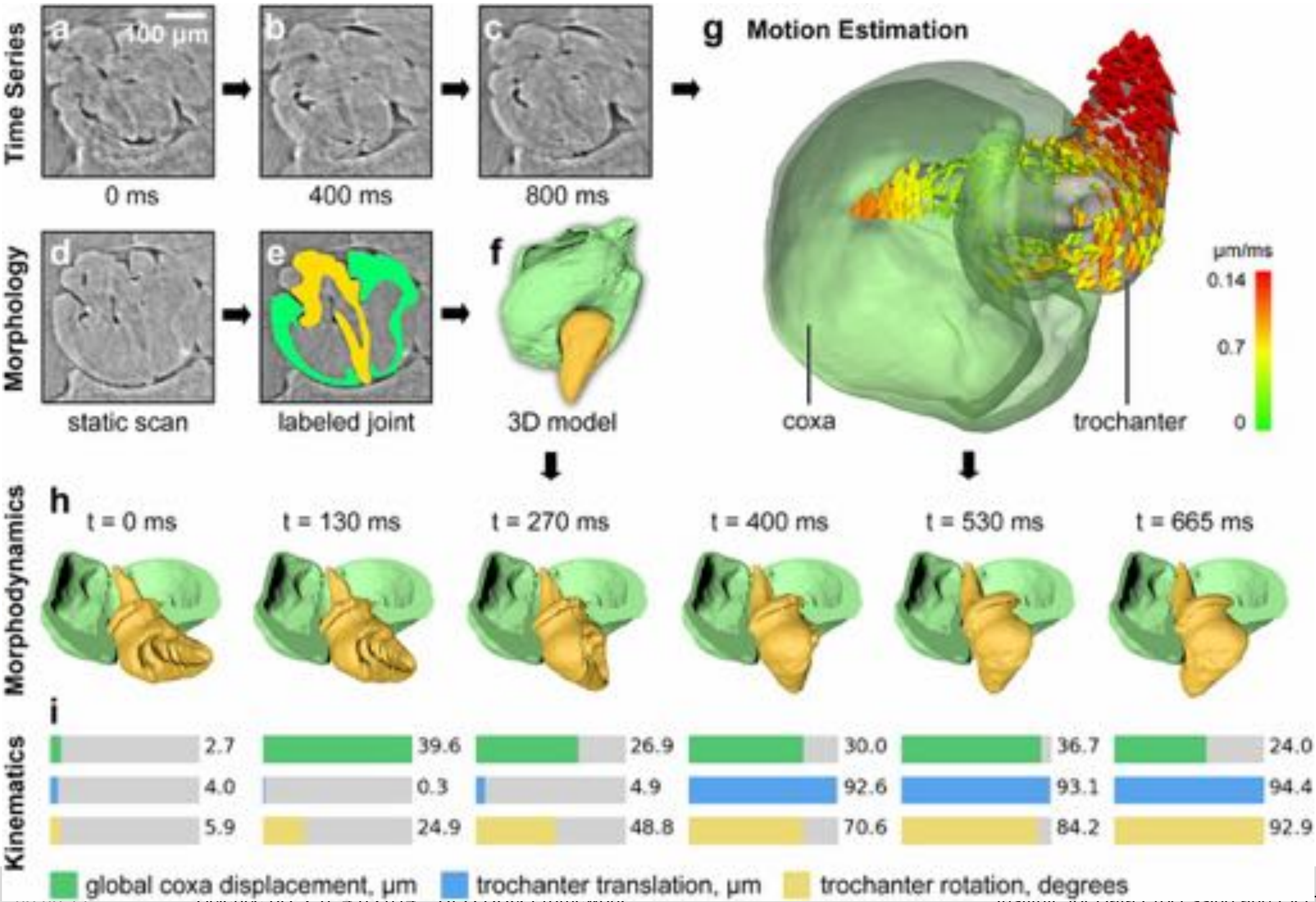


→ Matthias Vogelgesang gives an introduction tomorrow

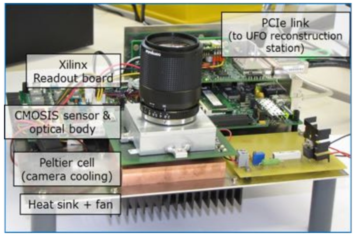
Summary

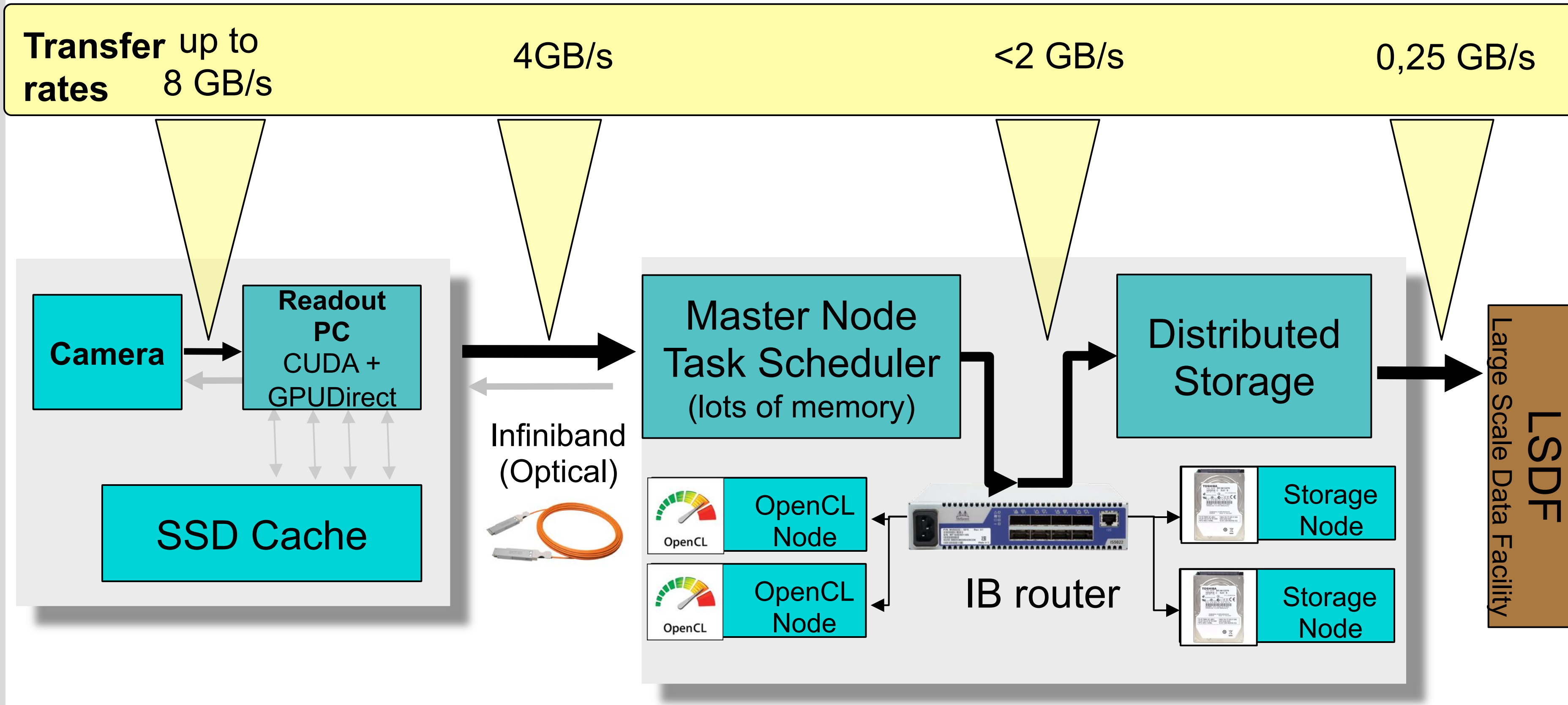
- The UFO DAQ framework is modular
  - Front-end / embedded processing / fast readout / GPU computing
  - Various sensors exist
  - Applications:
    - high-throughput camera,
    - intelligent trigger,
    - phase contrast
  - Non camera use: beam monitoring with ps resolution
- UFO parallel computing framework supports
  - Development and management of optimized code
  - Supported architectures GPU, CPU, Xeon Phi, <*every OpenCl device*>
  - FBP is now faster than DAQ
    - throughput ~1 GB/s

Morphological dynamics and kinematics analysis of a moving screw joint



What’s next? Scaling up to clusters





http://ufo.kit.edu

https://github.com/ufo-kit/concert

Thank you!