MICROGATE

FPGAs for flexible interfacing and distributed computational tasks

AO4RTC 2023

Roberto Biasi

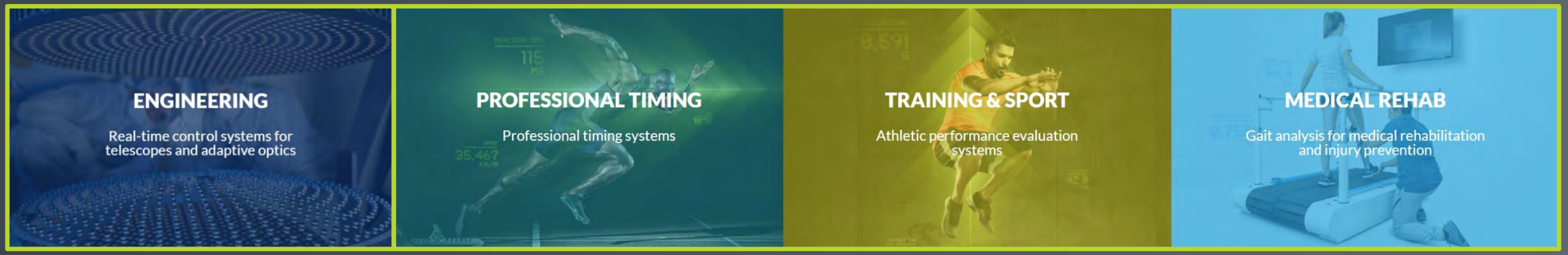
Outline

MICROGATE

- Very brief company presentation
- FPGA-based flexible sensors and actuators interface platform
- FPGA for distributed computational tasks
- FPGA: lessons learned, pros and cons

Main Business Units

MICROGATE



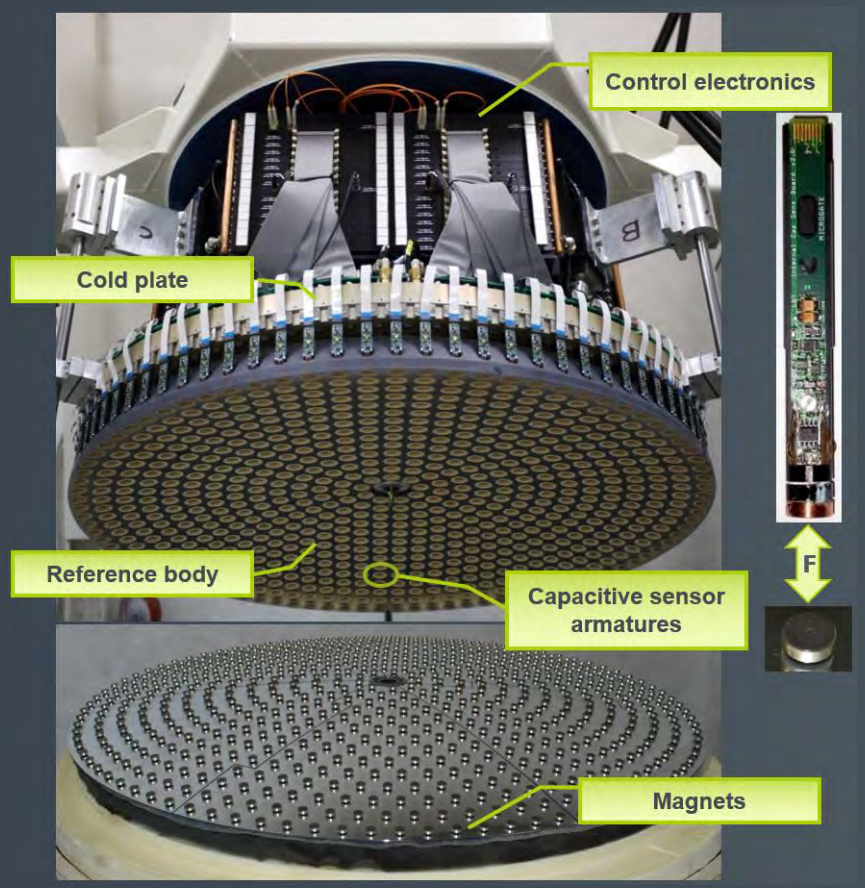
MPD
MICRO PHOTON DEVICES

Where we are, what we do

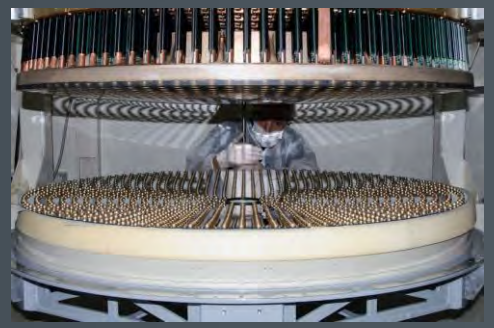
- New premises located in the industrial area of Bolzano-south, South Tyrol
  - 6,000 m^2 (including MPD)
  - Electronics labs
  - Mechanical workshop
  - Thermal and EMC tests
  - Optical test areas
  - Large clean integration room: 400 m^2, 20 t overhead crane, large climatic test pit

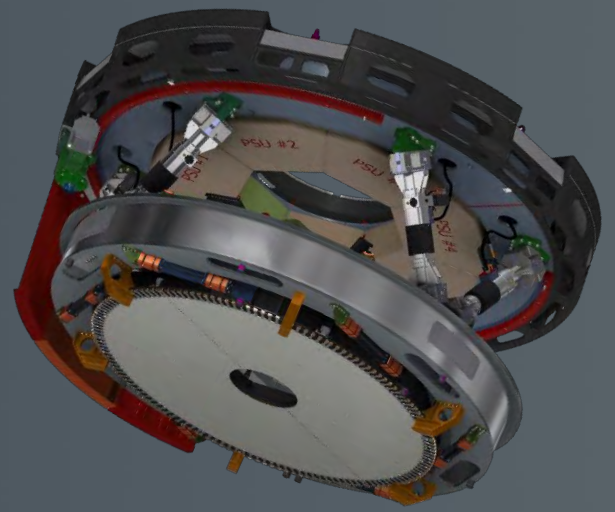
- The internal capabilities cover the entire process of electronic systems design and manufacturing
  - Hardware design (digital, analog)
  - Firmware (FPGA, microcontrollers)
  - Software
  - Control system design and multiphysics simulation
  - Prototyping
  - Integration of complex mechatronics systems
  - Testing
  - Production

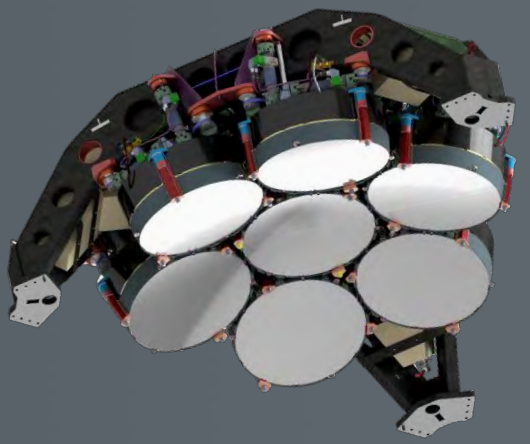
Microgate Engineering



- Microgate (with ADS International, INAF and Politecnico di Milano) has developed the large, contactless, VCM-driven adaptive mirror technology over the past > 25 years
  - Deployed on MMT, LBT, Magellan, VLT
  - In construction: Subaru, ESO-ELT, GMT







Microgate Engineering

- Other fields:
  - Motion control systems
  - Metrology

  } ALMA
  - Optical communication: ALASCA
  - **AO RTCs**: Keck, Padova AO Lab RTC (ANU, SUT)

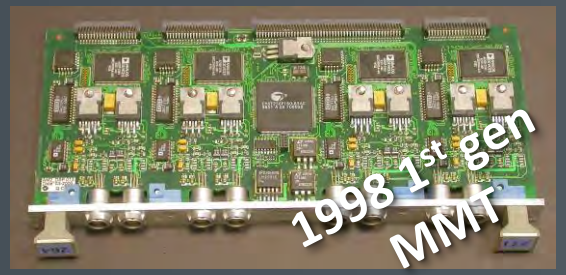
ALASCA (ESA)

Advanced Laser guide star Adaptive optics for Satellite Communication Assessments

FPGA background at Microgate

DM control: Typical hard-real-time application

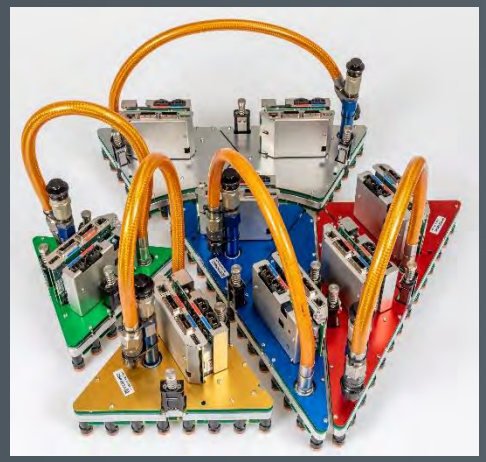
- Very fast control loop cycle, ~ 80 kHz
- Very few µs latency
- Large number of on-board devices (up to 36 ch, ADCs, DACs)
- Strong parallelism

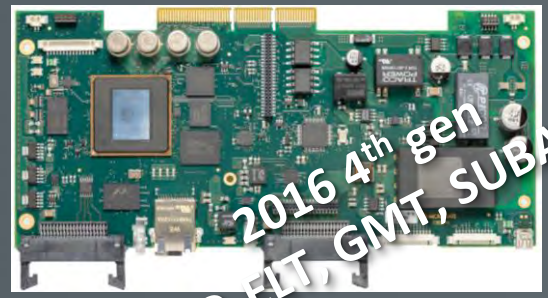


FPGA: real-time communication, on-board device management
Real-time control on DSPs

Ideal playground for FPGAs

- ‘Modern’ approach, DSP not supported any more
- System on chip, one device to cope with all needs
- Lower power consumption
- Less space





All FPGA-based – on-chip DSP blocks

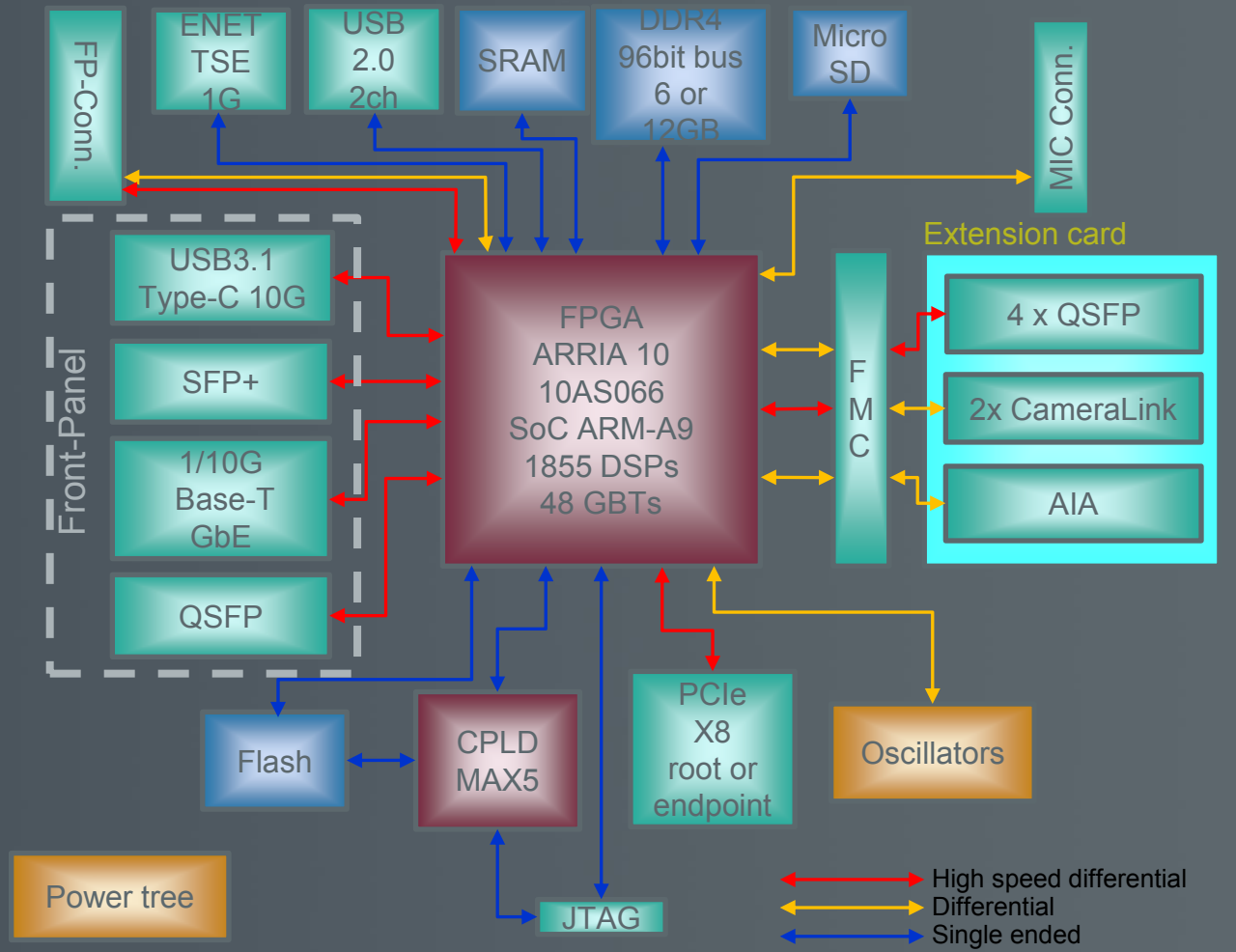
µXLink, PCIe FPGA board by Microgate

MICROGATE

Main motivations to develop a general-purpose PCIe FPGA board

- The **GreenFlash** H2020 EU founded project, led by D.Gratadour, aiming to compare RTC technologies for the ELTs, gave us the opportunity to develop a new cutting-edge FPGA based interface board (**µXLink**)
- Developing our own FPGA board instead of using a COTS allows us to be **vendor-independent** in terms of drivers and software
- Since we have all **knowledge in our hands** (hardware and firmware), we can better guarantee **long term support** of our electronics, including proper obsolescence management.
- Optimal selection of **interfaces** during the design phase, so to allow connecting of a large **variety of sensors and mirrors**
- Optimal usage of all available hardware resources to route the real-time data path so to achieve **minimum latency** and **maximum time determinism**.

µXLink architecture



Board facts:

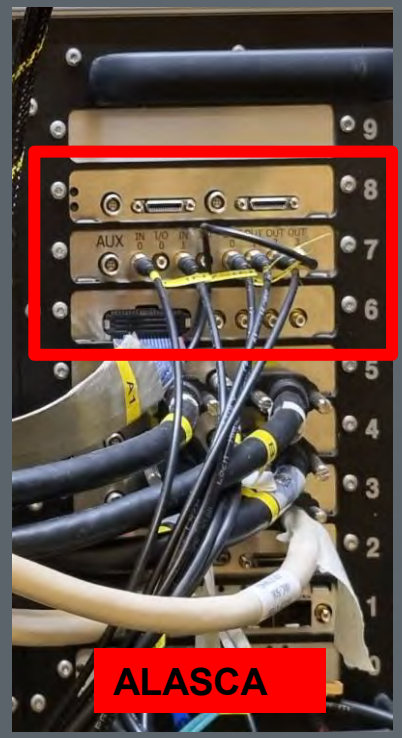
- SoC FPGA with ARM-A9 dual core + 1855 FP DSP blocks
- Board size compliant with PCIe standard full height and >= ¾ length
- # Layers: 18 (9 signal, 9 power-ground)
- # Components: 1752
- # Tracks: 7155 (300 LVDS pairs)
- # Vias: 10207

µXLink applications: Interface Module

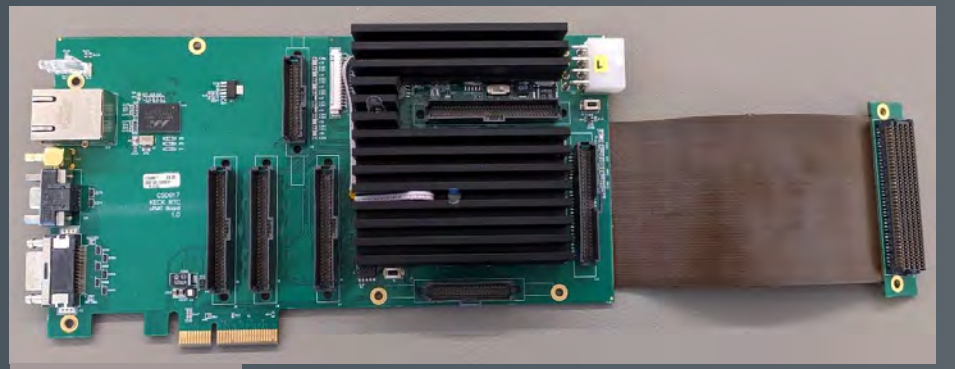
MICROGATE

- Interfaces to a large **variety of sensor and actuators** (mirrors)
- **Flexibility** in adding/changing interfaces by exchanging simple passive front-end boards
- **Rugged system** that can be in telescope enclosure environment, close to the devices
- Flexible and accurate **synchronization** of devices operation
- **Low power consumption**
- **Hardware abstraction** – one single interface and communication protocol to/from the computational units

Keck RTC

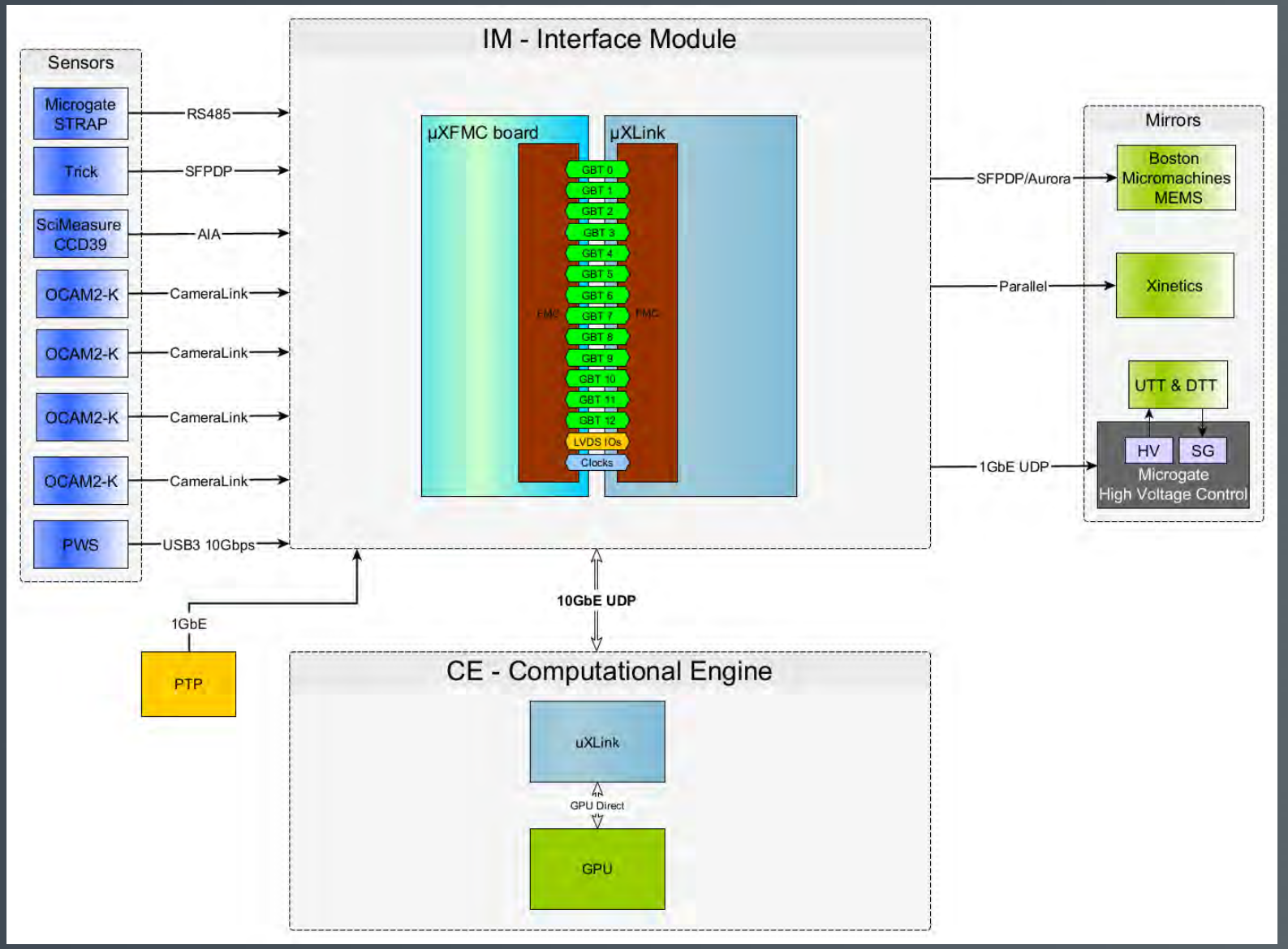


ALASCA



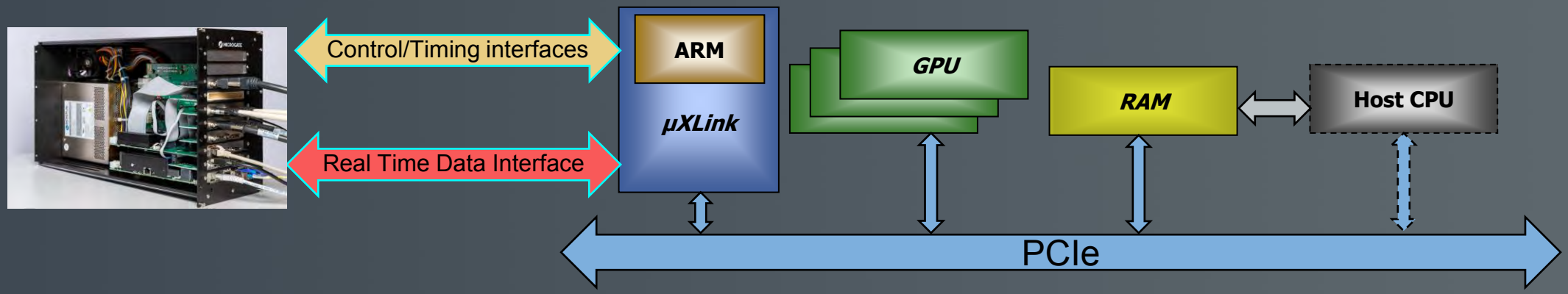
uXFMC

IM example: Keck RTC architecture



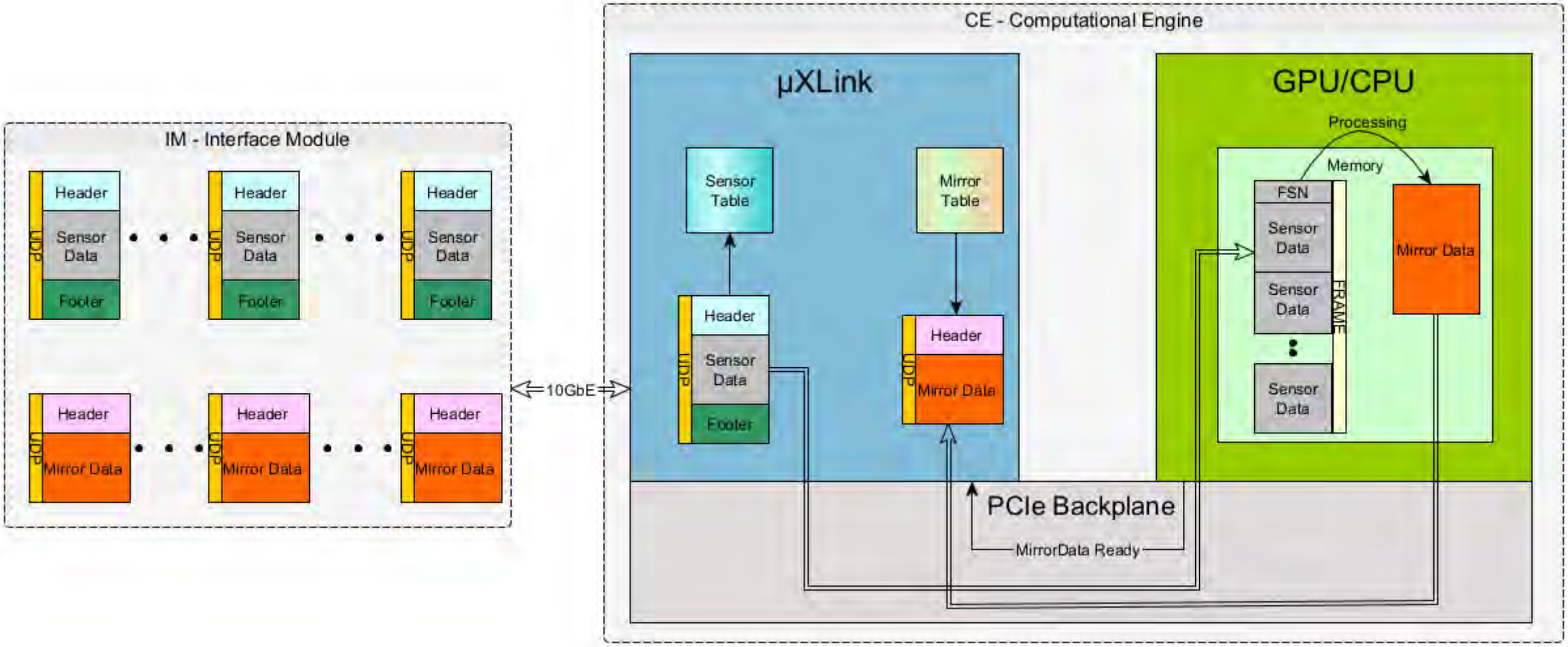
Flexible FPGA-based interface module

- GPUDirect and CPU DMA transfer
  - Direct data transfer from µXLink to the **GPU** (NVIDIA) via **GPUDirect** without interacting with Host CPU
  - Direct data transfer from µXLink to RAM memory over PCIe for CPU based reconstructor without Host CPU interaction
- SW initialization
  - Host software configures the µXLink PCIe engine to perform directly CPU/GPU RAM memory data transfers
  - The computational software acts only on data available in memory. In this way the computational software is totally **abstracted** from the **hardware interfaces**.



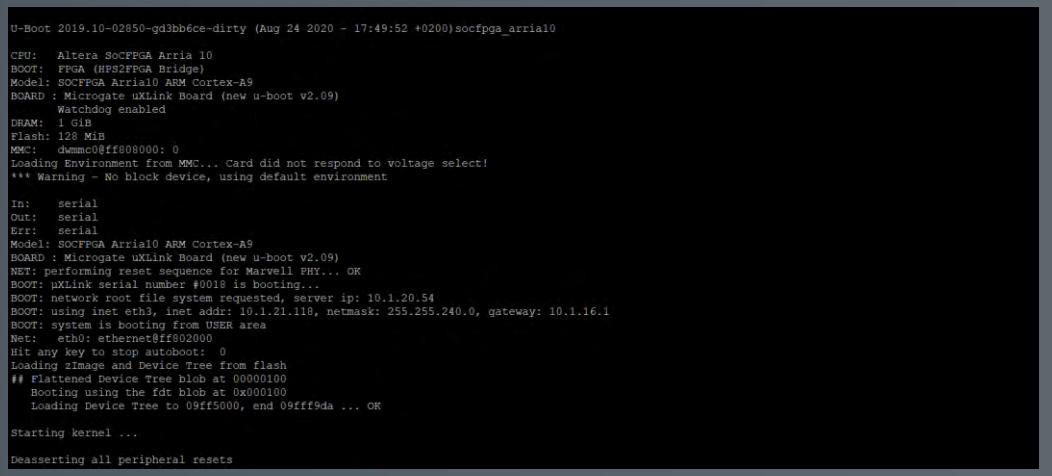
Real-time pipeline transfer processed in HW

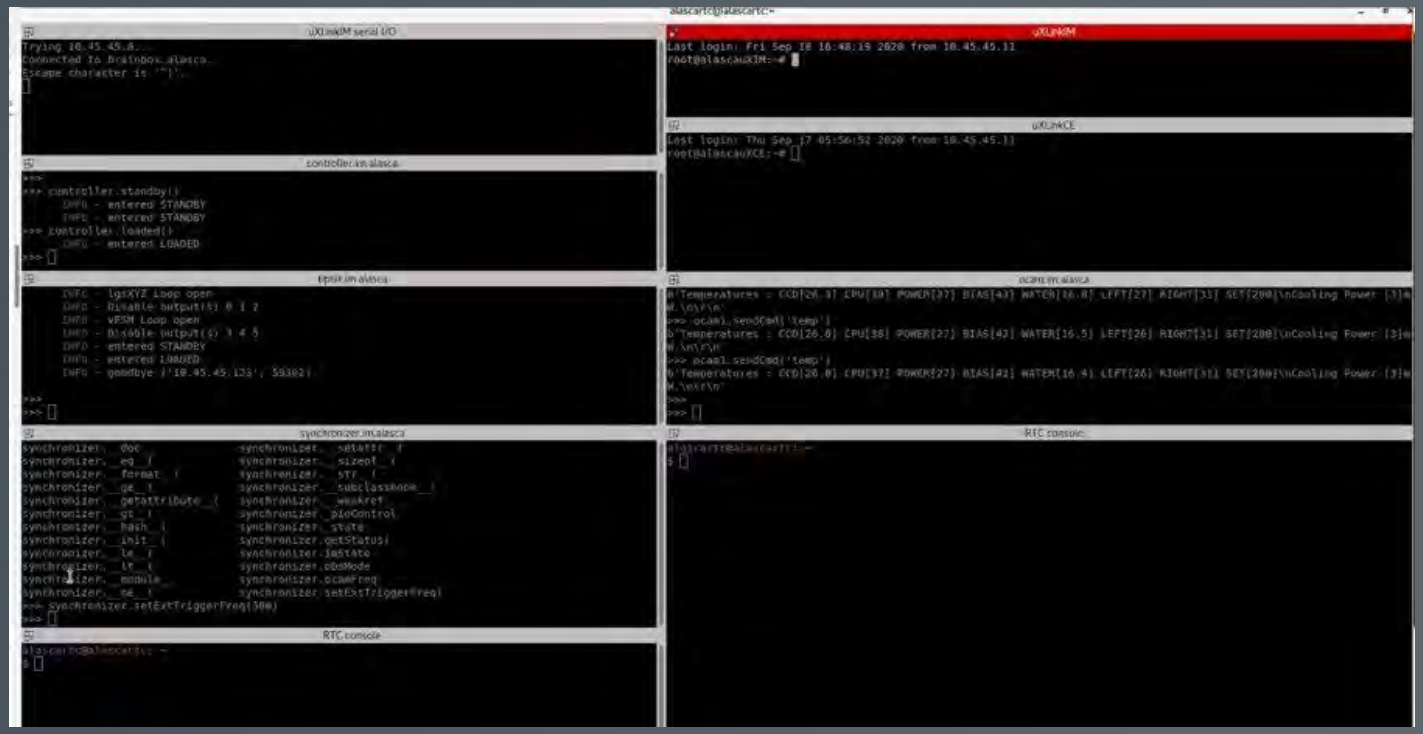
MICROGATE



Interface module SW ecosystem

- ARM Dual-Core CPU in the ARRIA 10 FPGA (SoC)
- Full software stack in our hands
  - Allows full software support to customer
- Optimized Compact Linux OS
  - based on Yocto
  - we provide dedicated drivers
- SSH interface to µXLink board
- Direct command interface to different cameras
- High level software for:
  - Configuration
  - Housekeeping
  - Maintenance





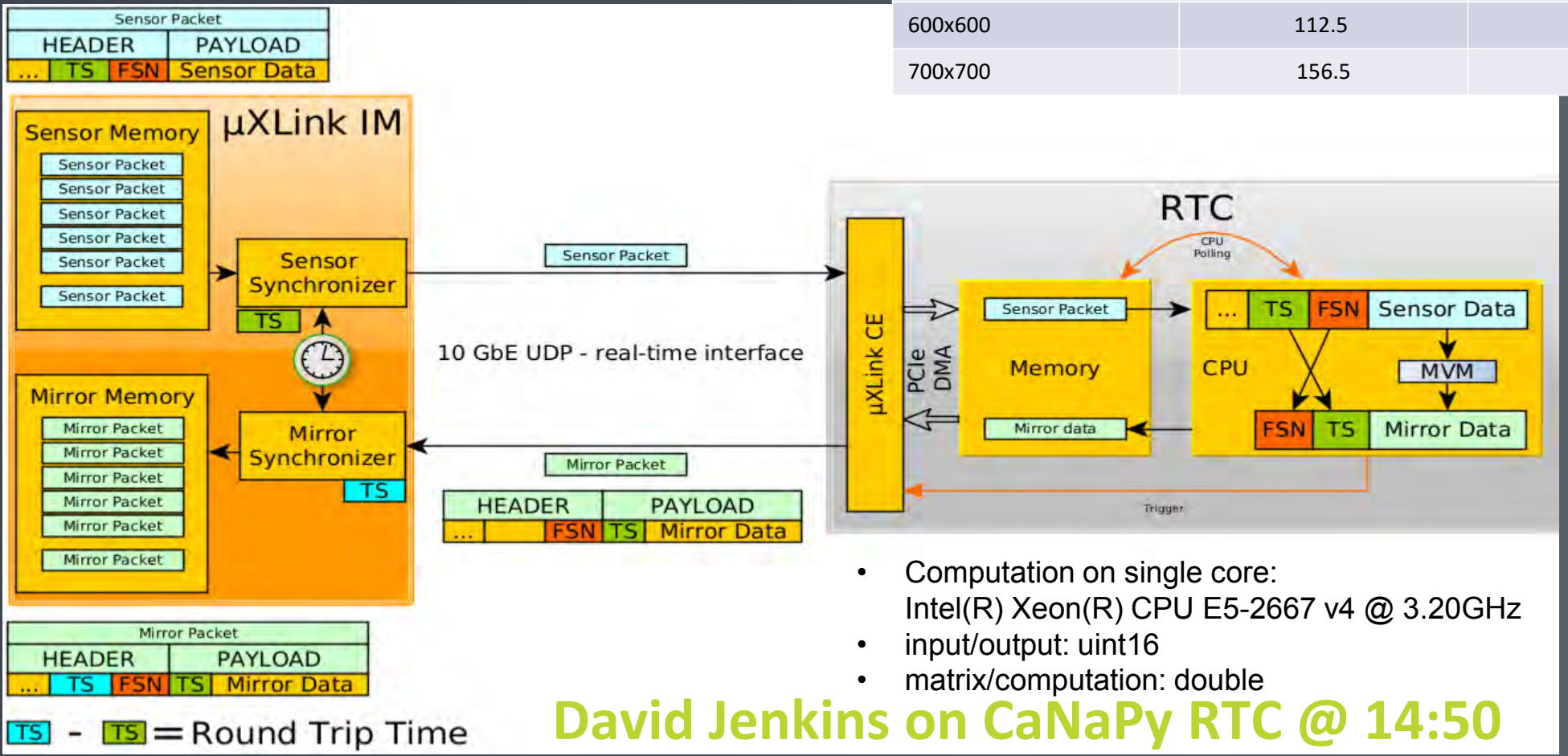
Interface Module and Computational Engine

MICROGATE

Real performance facts: ALASCA

- No computation in FPGA
- CPU based CE

Dimension (MVM)	Min [µs]	Mean [µs]	Max [µs]
0x0 (Round trip)	**15.0**	**17.6**	**21.5**
100x100	18.0	20.6	24.0
200x200	28.3	30.9	42.8
400X400	60.5	63.3	68.3
600x600	112.5	114.5	135.5
700x700	156.5	159.4	173.0

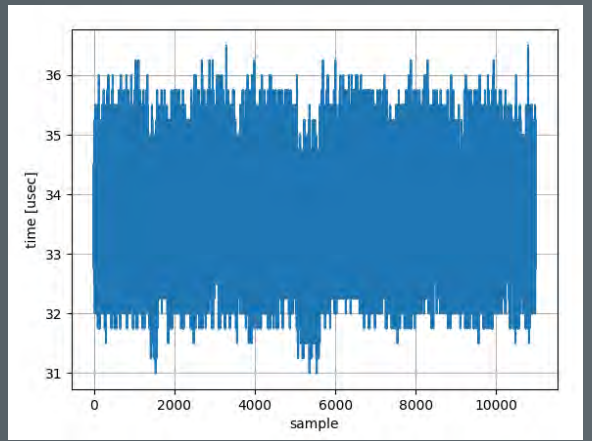


ALASCA specific case:

MVM (224x224)

T_{min} = 31.0 µs
T_{mean} = 33.7 µs
T_{max} = 36.5 µs

- Computation on single core:
  Intel(R) Xeon(R) CPU E5-2667 v4 @ 3.20GHz
- input/output: uint16
- matrix/computation: double



David Jenkins on CaNaPy RTC @ 14:50

Interface Module and Computational Engine

MICROGATE

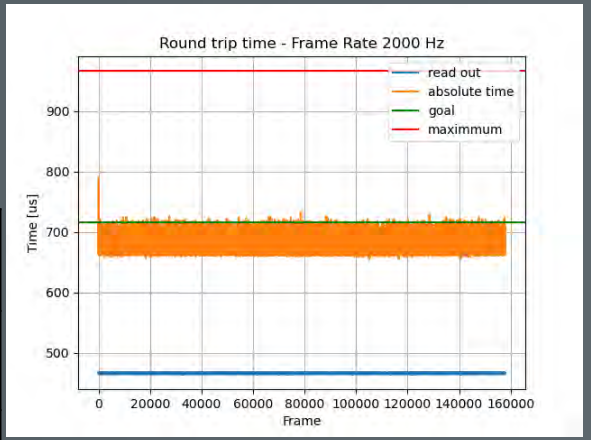
Real performance facts: Keck RTC

- No computation in FPGA
- GPU based CE,
  implementing the COSMIC ecosystem

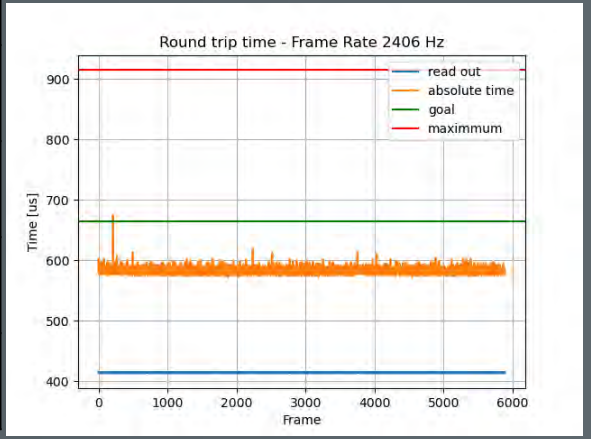
a) Image calibration
b) Centroid computation
c) MVM (352x608)
d) DM Control Filter

Sensor	Frequency				Roundtrip Time				Camera Readout time			
CCD39	min	mean	max	jitter	min	mean	max	jitter	min	mean	max	jitter
	41.35	41.36	41.36	0.00	**159.26**	**173.09**	**261.07**	**9.52**	24150.00	24152.00	24152.00	0.44
	1031.30	1032.08	1033.33	0.49	**159.03**	**169.38**	**265.84**	**5.35**	946.52	947.65	948.19	0.45
	2402.23	2405.54	2411.90	2.73	**157.36**	**168.02**	**261.78**	**5.27**	413.41	414.32	415.32	0.54
OCAM	49.99	49.99	50.00	0.00	**185.73**	**206.51**	**315.90**	**11.50**	465.39	466.31	467.53	0.53
	998.40	999.91	1001.26	0.45	**189.30**	**205.39**	**313.76**	**6.97**	464.91	466.31	467.77	0.43
	1994.43	1999.82	2005.88	1.72	**188.11**	**205.55**	**324.01**	**6.70**	464.91	466.31	467.77	0.43

OCAM 2kHz (240x240 pixel)



CCD39 2.4kHz (80x80 pixel)



D.Gratadour, J.Bernard, N.Doucet on COSMIC

Computation with FPGA

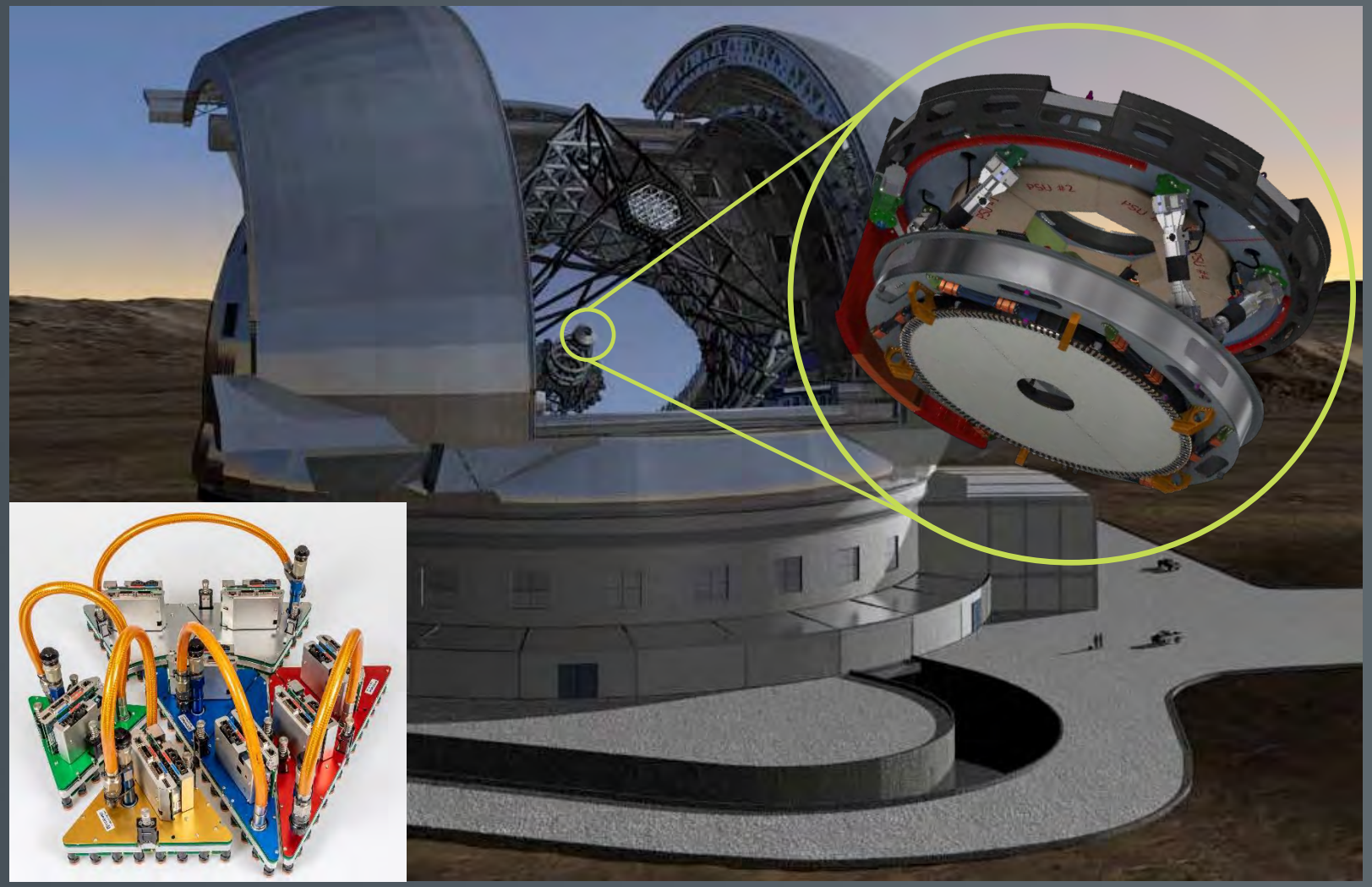
MICROGATE

- Besides the fast (~ 80 kHz) local control, our DMs require some global computational task:
  - Transformation of **modal** commands from the RTC into **zonal** commands for the actuators
  - **Modal clipping**, to avoid force saturation of any actuator:
    - Computation of the force pattern applied by actuators to achieve the position (shape) setpoint
    - Seeking for the largest number of commanded modes (usually all of them…) that can be applied to the mirror without any force saturation
    - Return of the 'applied setpoint' to the RTC
- Very **tight timing constraints**
- Substantially it's a large MVM, could be implemented directly in the RTC, but execution at DM level is preferred (safety involved, clear interface, avoid additional load to the RTC, …)

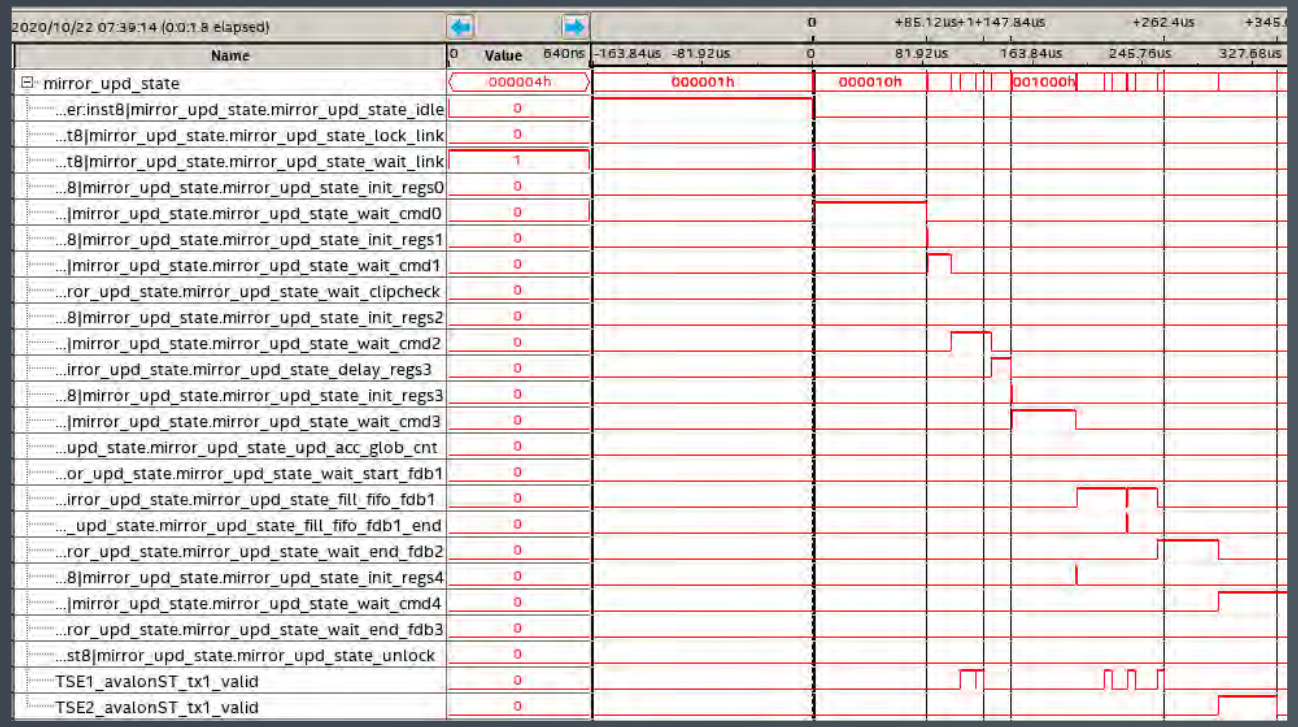
Computation with FPGA: ESO ELT-M4 case

MICROGATE

- 5316 actuators
- Fast (~80 kHz) and quite complex local control loop realized by 180 FPGA-based control *bricks* with FP DSP blocks
- Very efficient data interconnection
- Why not implementing the *global* computation (clipping) in the same FPGAs?
  - Efficient HW exploitation
  - Power efficient
  - Ultra low latency



Computation with FPGA: ESO ELT-M4 case



- Main task: execute **two FP32 MVMs**, **[6480x5352]*{5352}** each
- Computation **parallelized over the 180 FPGAs** available on the DM control *bricks* [36x5352]*{5352x1} each
- The computation is **pipelined** with the modal command data distribution to all *bricks* (3.125 Gbit/s proprietary redundant link), no additional latency introduced by the data transfer
- Data distribution and **arbiter** functions performed by a single **µXLink board**
- **85 µs computational time**, 125 µs to return the applied setpoint to the RTC (ESO req: 150 µs), jitter in sub-µs level

General considerations 1/2

MICROGATE

- Pros and cons of in-house approach (no COTS)
  - + Master all **hardware and low-level firmware** (drivers)
  - + Continuity in **support to customer**
  - + Increase **internal know how**, easy **migration to next generation**, **obsolescence** management
  - - Limited resources do not allow to follow consistently the technological updates. Really so negative? Stability, consolidation of solutions, lifetime of typical AO projects…
- Proven by several examples in the field

General considerations 2/2

MICROGATE

- Lessons learned from FPGA in real applications
  - Development time is a real issue; stable once done, but reaching final deployment is not easy and can e very time consuming
  - FPGA use shall be justified by the context – and this is often the case!
    - Need of flexible hardware interface, on- and off-board
    - Very hard-real-time constraints, can’t do by SW
    - Size, power
  - Development tools improvement not as fast as HW growth
  - Can high level programming assure a bright future to FPGAs for computational tasks – HW accelerators?
    - Our experience: still not optimal in HW utilization and performance
    - Not solving the compile-fitting time issue
    - But… strong investments by the main players + research activities (e.g. RisingSTAR project, aiming to a single SW description on heterogeneus platfoms CPU/FPGA/GPU)

Wenn eine Idee nicht zuerst absurd erscheint,
taugt sie nichts.

A.Einstein

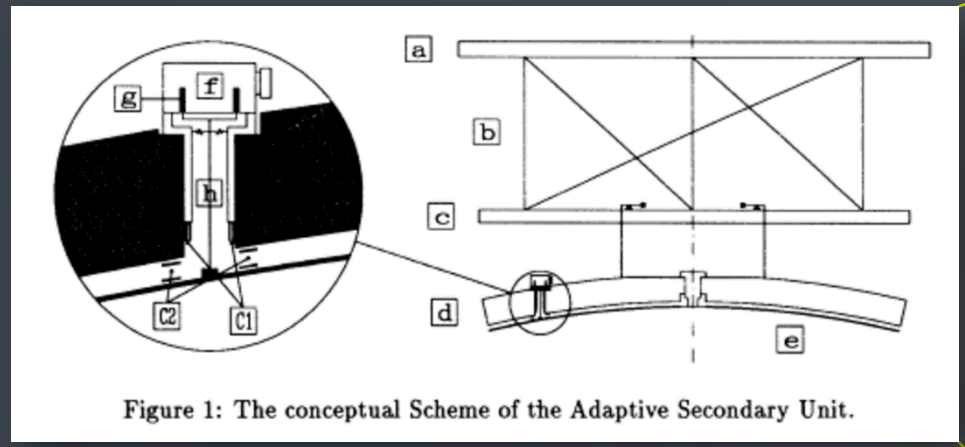


Figure 1: The conceptual Scheme of the Adaptive Secondary Unit.

microgate srl
Via Waltraud Gebert Deeg, 3e
39100 Bolzano
Italy

microgate.it

Thank you for your attention!