

**RTC4AO 6**

# **MICADO SCAO RTC: BUILDING THE ELT FIRST LIGHT RTC**

---

Florian Ferreira

On behalf of the MICADO SCAO RTC Team



## SUMMARY

- MICADO SCAO RTC Design overview
- Prototyping activities
  - H-RTC development status
  - RTC Tk integration
  - On-bench activities

## MICADO

- Status after FDR #4 (from FDR board report):

« The review of the final design can be considered complete for the majority of the MICADO sub-systems, with no showstoppers over the design but a few actions, most of them being considered “normal work” and a few being critical ones »

- Agreement from FDR board and ESO to start procurement and manufacturing

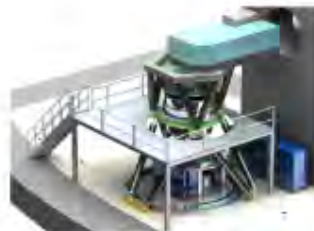
# MICADO SCHEDULE



2010



2015



2016



2018

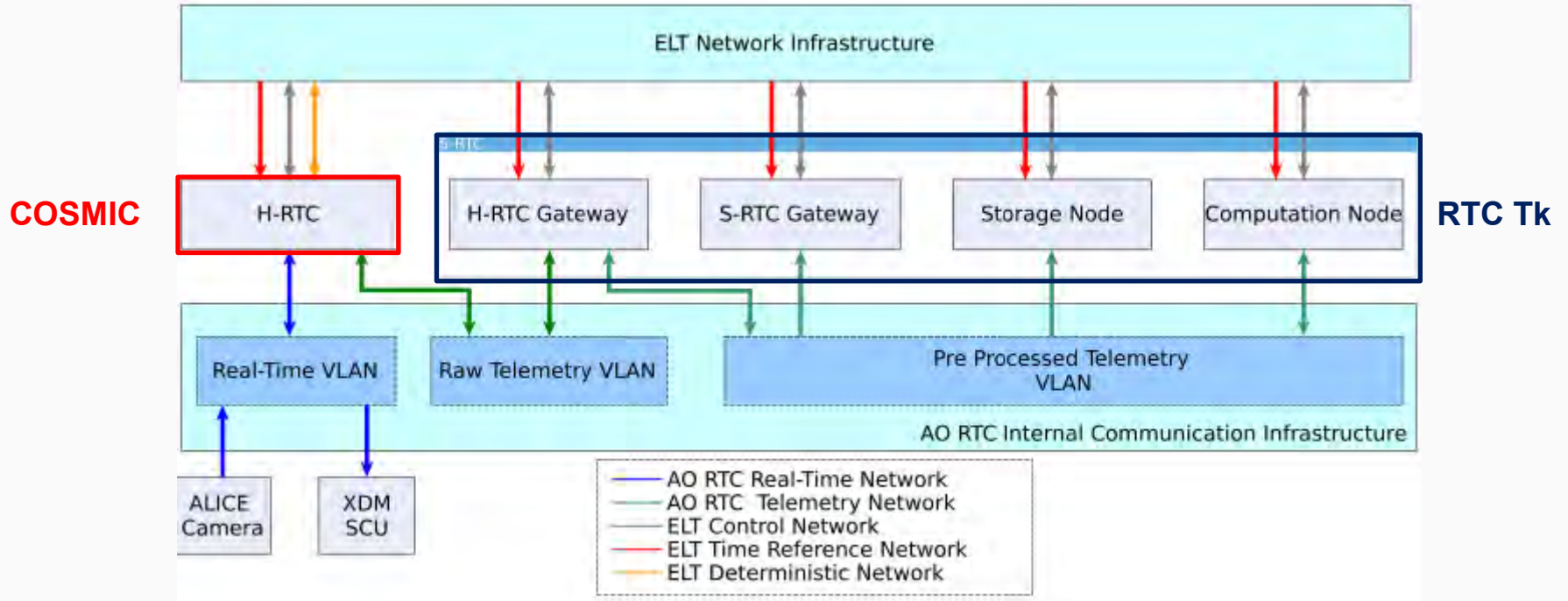


2018



2022

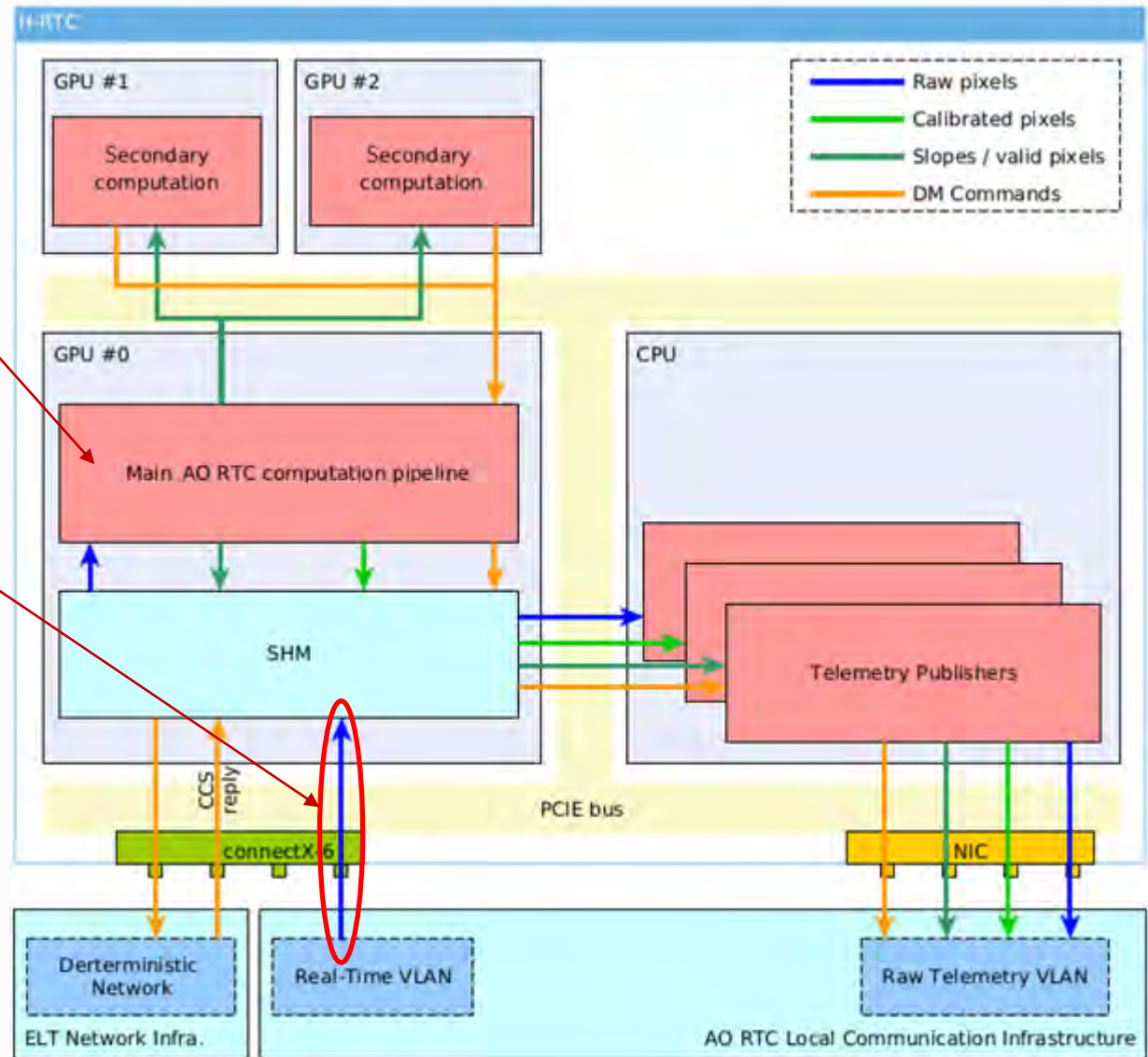
# RTC DESIGN OVERVIEW



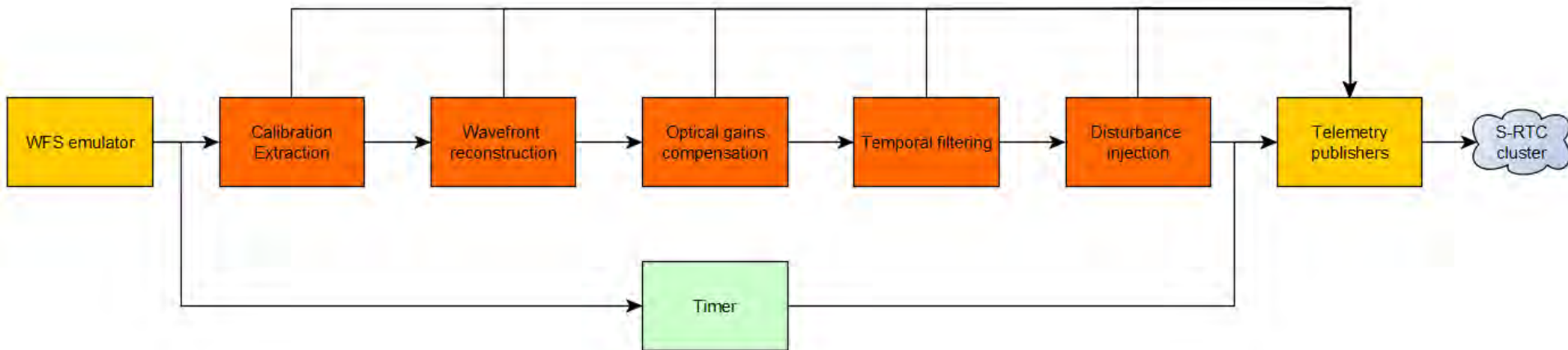
- ELT Standard implementation
- COSMIC-based H-RTC
- ESO RTC Toolkit based S-RTC

# H-RTC OVERVIEW

- COSMIC with GPU-based AO pipeline
- GPUDirect RTMS acquisition
- MUDPI Telemetry publishers



# H-RTC PERFORMANCE



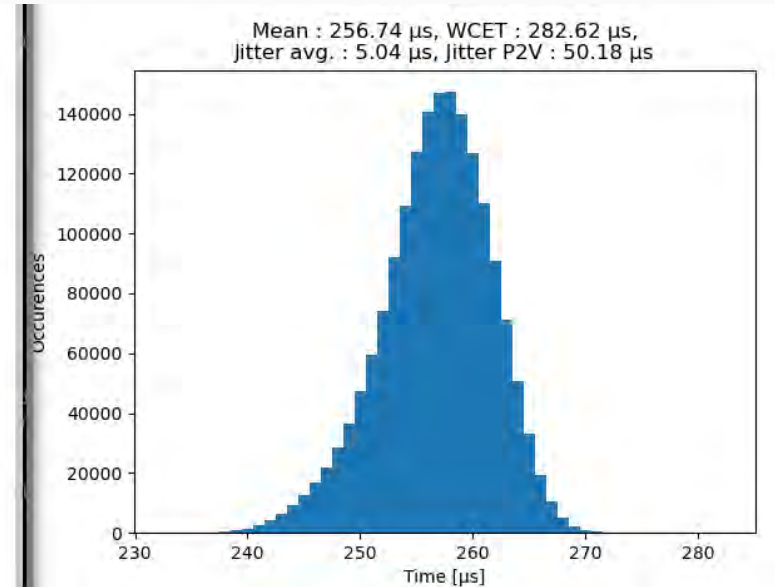
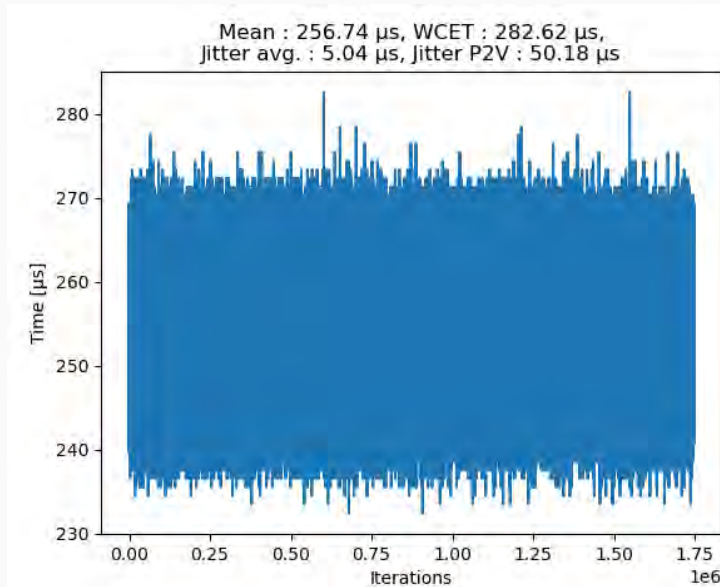
- Emulator mode @ 500 Hz (H-RTC producing ramp images)
- 4,868 modes and 24,416 pixels (PWFS full-pixels algorithm), full pipeline
- Measurement of the H-RTC computation time from image availability to commands computed
- H-RTC server under ELT devEnv 4 (Fedora 34) with 2x NVIDIA A100

## USEFUL REAL-TIME METRICS

- **Mean latency:** mean of the measured execution time (as defined just before)
- **Worst-Case Execution Time (WCET):** self-explanatory...
- **Best-Case Execution Time (BCET):** again, you got it...
- **Mean jitter:** mean of the execution time deviation (wrt to the mean latency)  
→ i.e. standard deviation of the measured execution time
- **Maximum jitter:** maximum deviation from the mean execution time  
→ i.e. WCET – mean latency
- **Peak-to-Peak jitter:** difference between WCET and BCET



# H-RTC PERFORMANCE



- Specs:
  - lat. < 305  $\mu$ s
  - Max. jitter < 10% mean lat.
- Mean latency  $\sim$ 257  $\mu$ s
- WCET  $\sim$ 283  $\mu$ s
- Average jitter  $\sim$ 5  $\mu$ s
- Max. jitter  $\sim$ 25  $\mu$ s ( $\sim$ 10% lat.)

**$\sim$ 2 TB/s of sustained memory bandwidth**

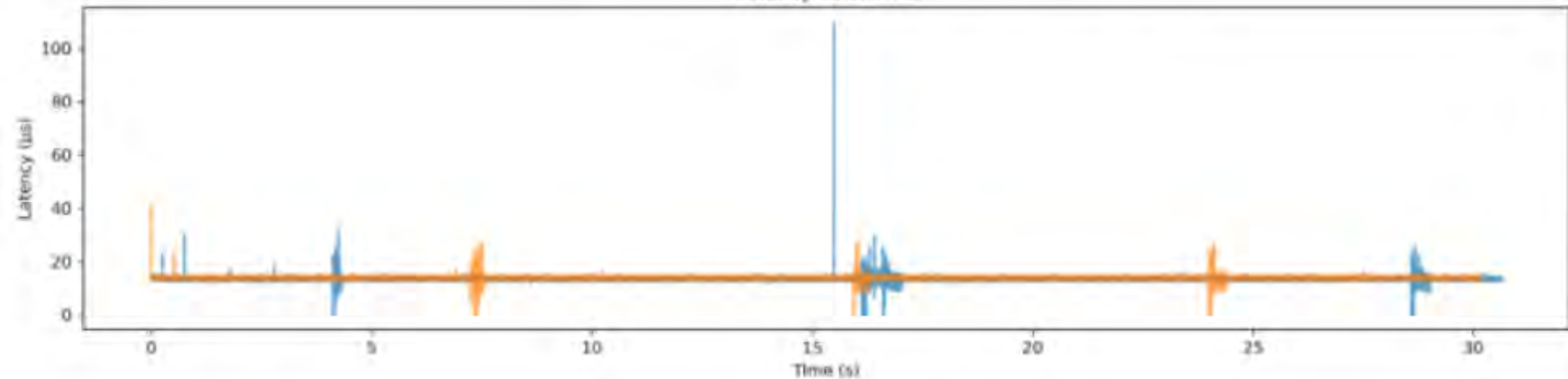
**60% of max. measured memory bandwidth (1.6 TB/s per GPU)**

**Trade-off toward maintainability:** standard implementation only, no custom CUDA kernels

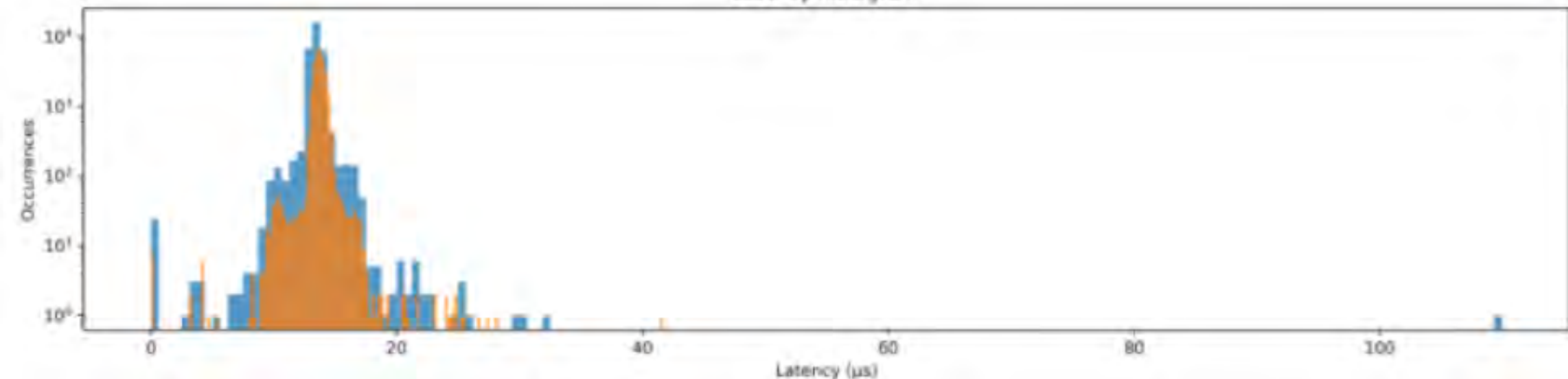
# H-RTC PERFORMANCE

1 WFS | 240x240 16-bit frames @ 500Hz per WFS

Latency as of time



Latency histogram



cpu\_only:  
-  $\mu = 13.42 \mu\text{s}$   
-  $\sigma = 0.99 \mu\text{s}$   
-  $\text{min} = 0.00 \mu\text{s}$   
-  $\text{max} = 109.92 \mu\text{s}$   
-  $\text{ptp} = 109.92 \mu\text{s}$   
goudirect:  
-  $\mu = 13.71 \mu\text{s}$   
-  $\sigma = 0.73 \mu\text{s}$   
-  $\text{min} = 0.00 \mu\text{s}$   
-  $\text{max} = 41.64 \mu\text{s}$   
-  $\text{ptp} = 41.64 \mu\text{s}$

- GPUDirect acquisition mean latency between sent of first packet and receival of the last packets:  $\sim 14 \mu\text{s}$ , max.  $42 \mu\text{s}$

# S-RTC OVERVIEW

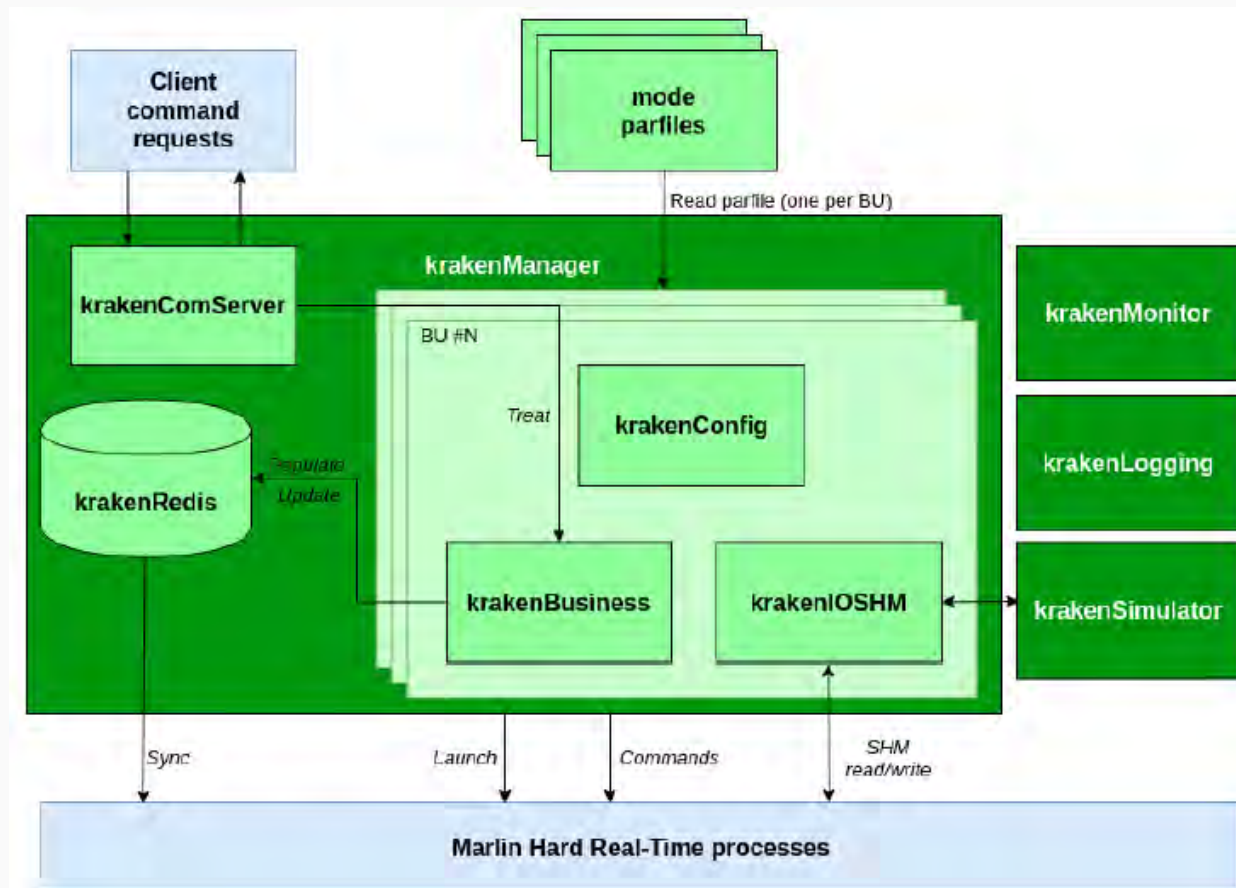
- Based on ESO RTC Toolkit
- ...but leveraging in-house S-RTC python-based software
- 4 physical nodes, aligned with ESO RTC Tk reference design:
  - **H-RTC Gateway:** hosting Telemetry Republishers (MUDPI → DDS) and H-RTC Supervisors
  - **S-RTC Gateway:** hosting the RTC supervisor and main connection point with AOCS
  - **Storage node:** hosting Telemetry Recorders
  - **Compute node:** hosting Data Tasks
- Design includes :
  - 12 telemetry topics
  - 33 Data Tasks
  - 12 H-RTC supervisors
- 2 setups for development: full-scale simulation & bench

# DATA TASK

- Baseline is to use Python code + pybind11 interpreter
  - Allows easy re-use of existing in-house S-RTC algorithm
  - Allows continuous development & integration along in-house S-RTC software
- AO team easily develops and tests algorithms...
- ...which are easily integrated into a Data Task

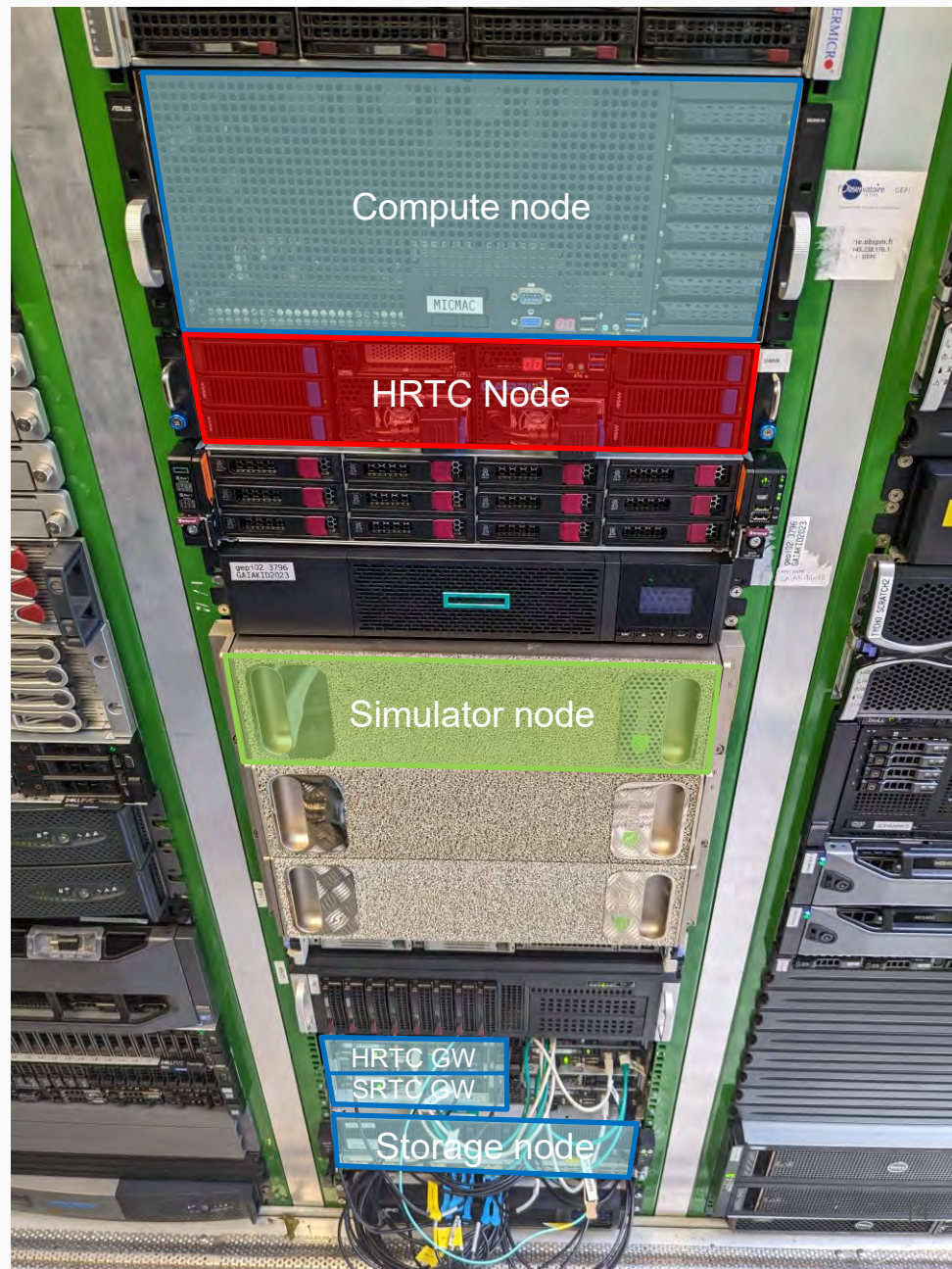
## H-RTC/S-RTC COMMAND INTERFACE

- H-RTC Supervisors will rely on COSMIC's Tides software
- Based on ZMQ request/reply + data serialization

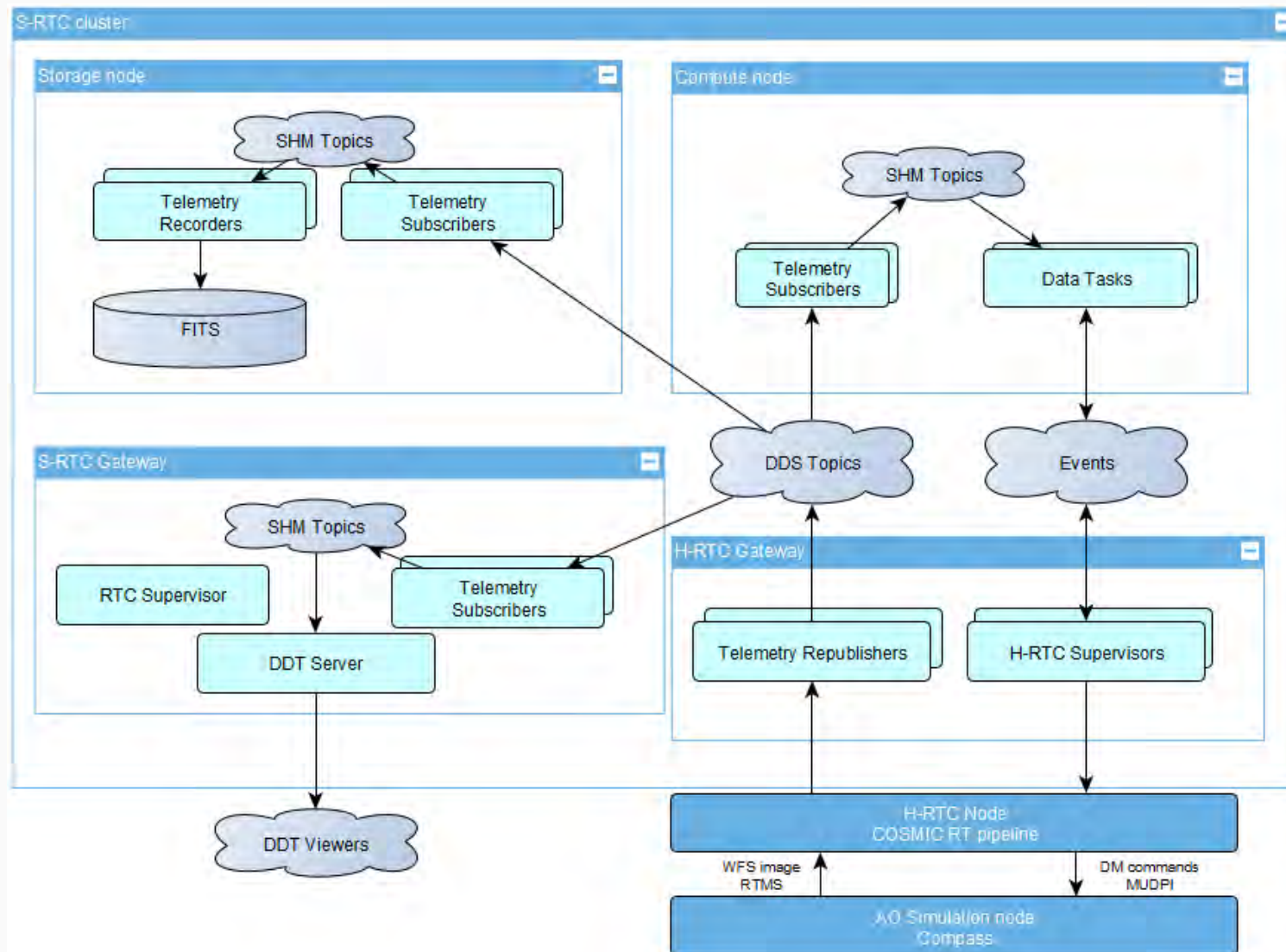


# RTC TOOLKIT INTEGRATION

- Prototyping activities full scale setup:
  - 1x WFS simulator (ESO for full-speed, or COMPASS for AO performance)
  - 1x H-RTC
  - 4x S-RTC Nodes
  - 10 Gbe switch for interconnect
- On-bench setup:
  - 1x H-RTC
  - 1x S-RTC Node
  - 1x Workstation
- Allow parallel development wrt to on-bench activities



# RTC TOOLKIT INTEGRATION: DEPLOYMENT VIEW



**COMPASS Simulation (Simulator node)**

**HRTC (HRTC node)**

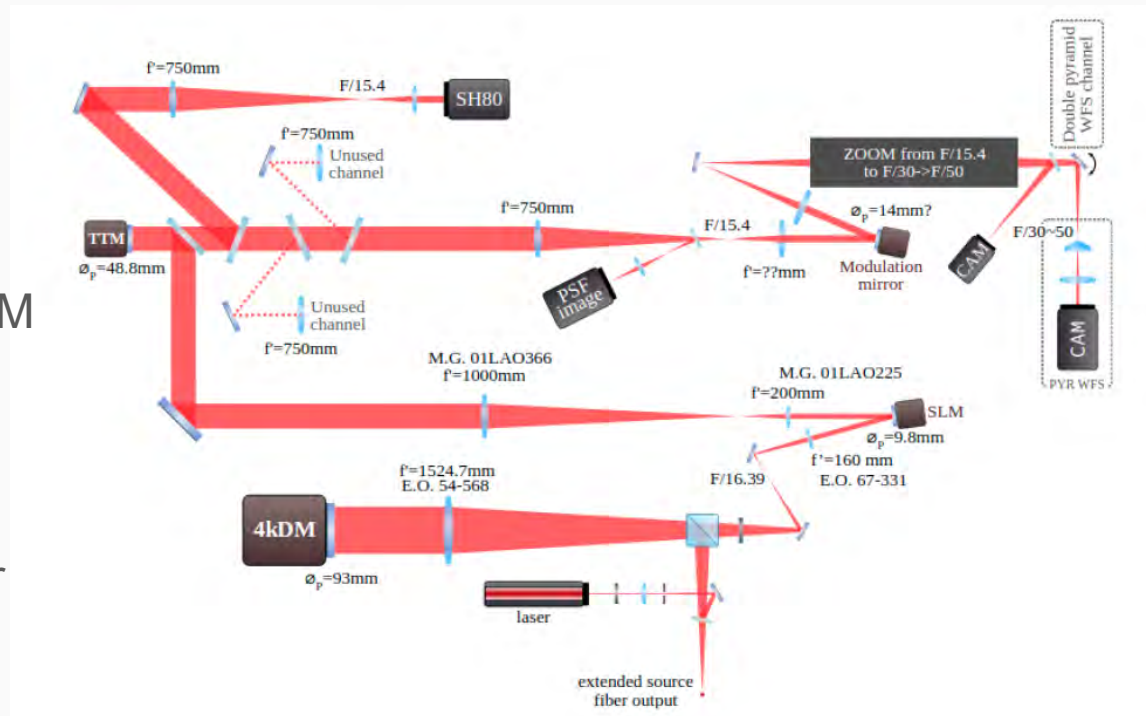
**RTC Tk CtrlMon GUI (src1 node)**

**DDT Viewer (src2 node)**



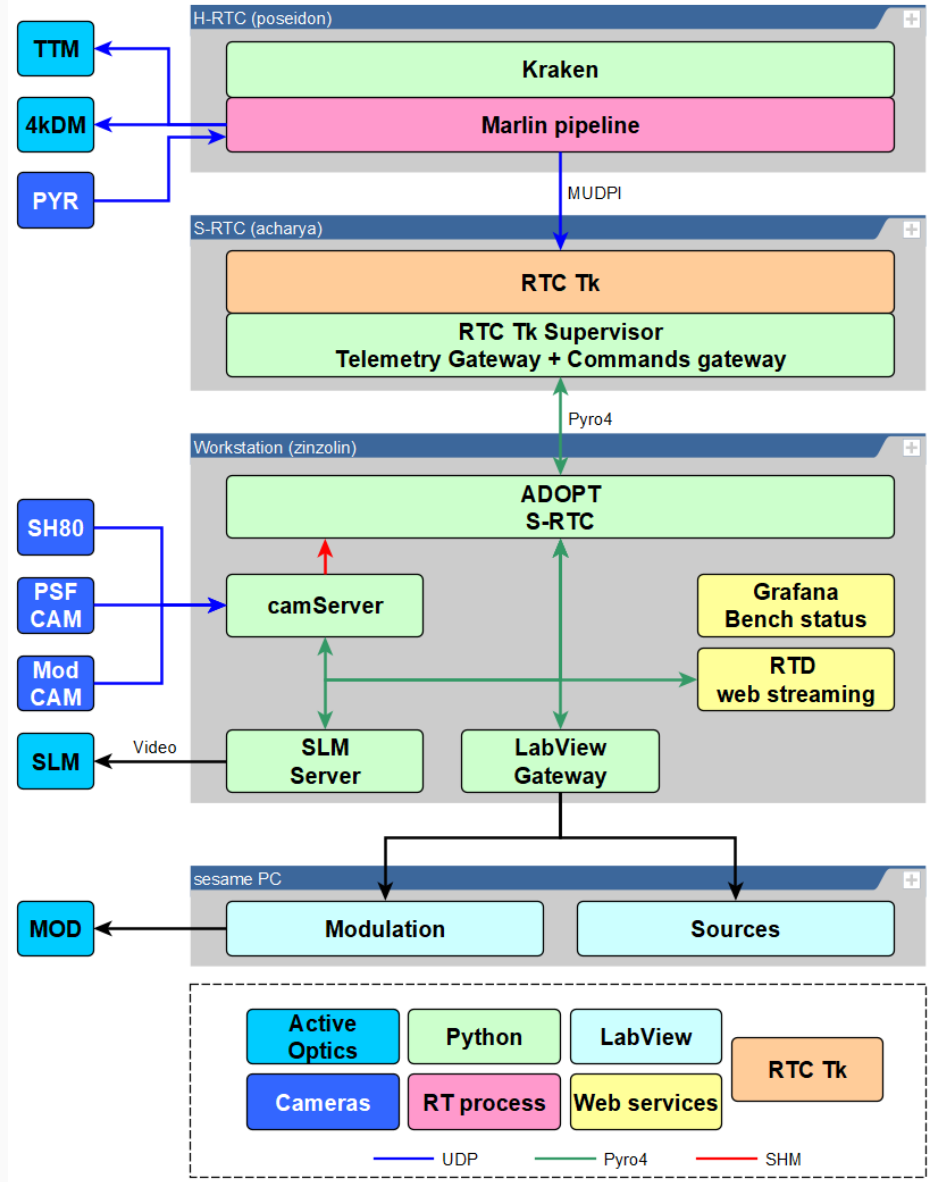
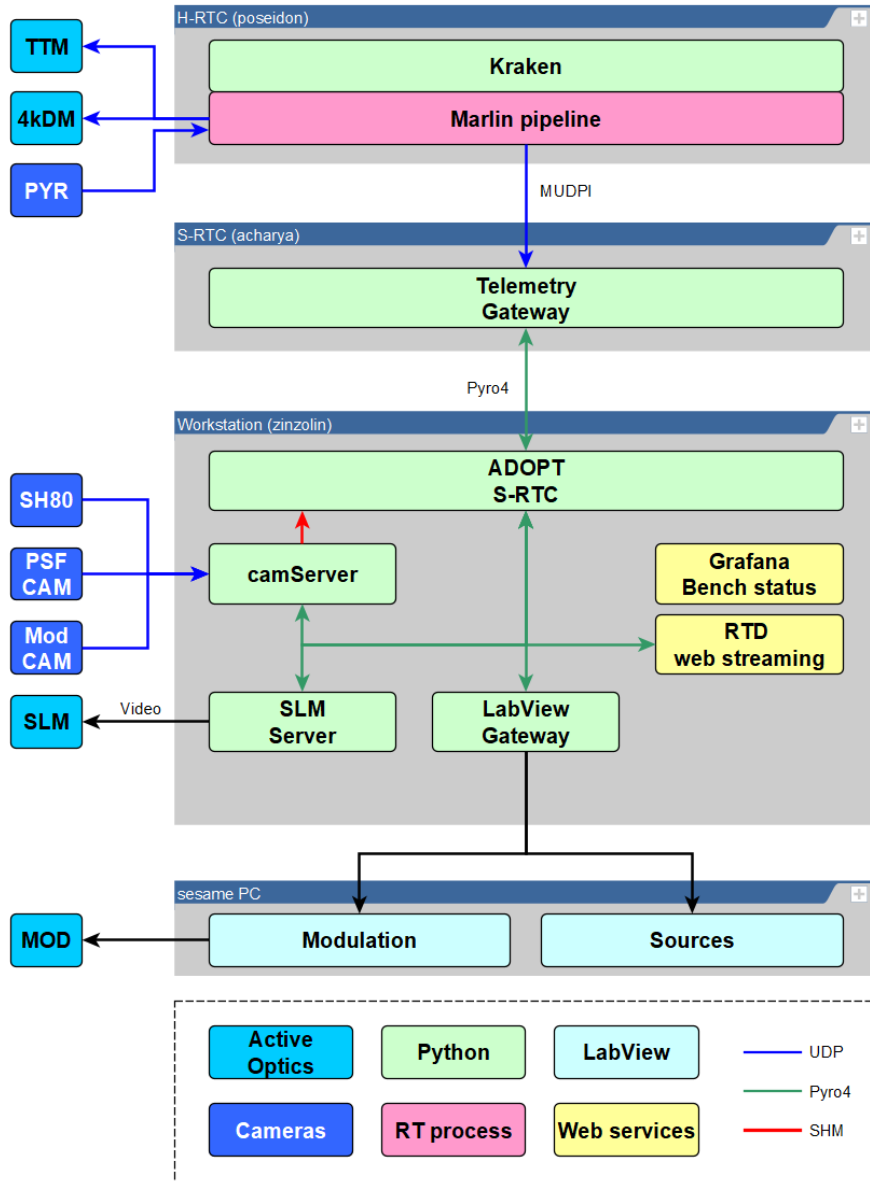
# SESAME BENCH

- 633 nm laser point source
- 102x102 pyramid WFS
- 28,320 valid pixels
- 3,228 actuators ALPAO DM
- Tip-tilt mirror
- Pyramid modulation mirror
- PSF camera





# SESAME BENCH: DEPLOYMENT VIEW



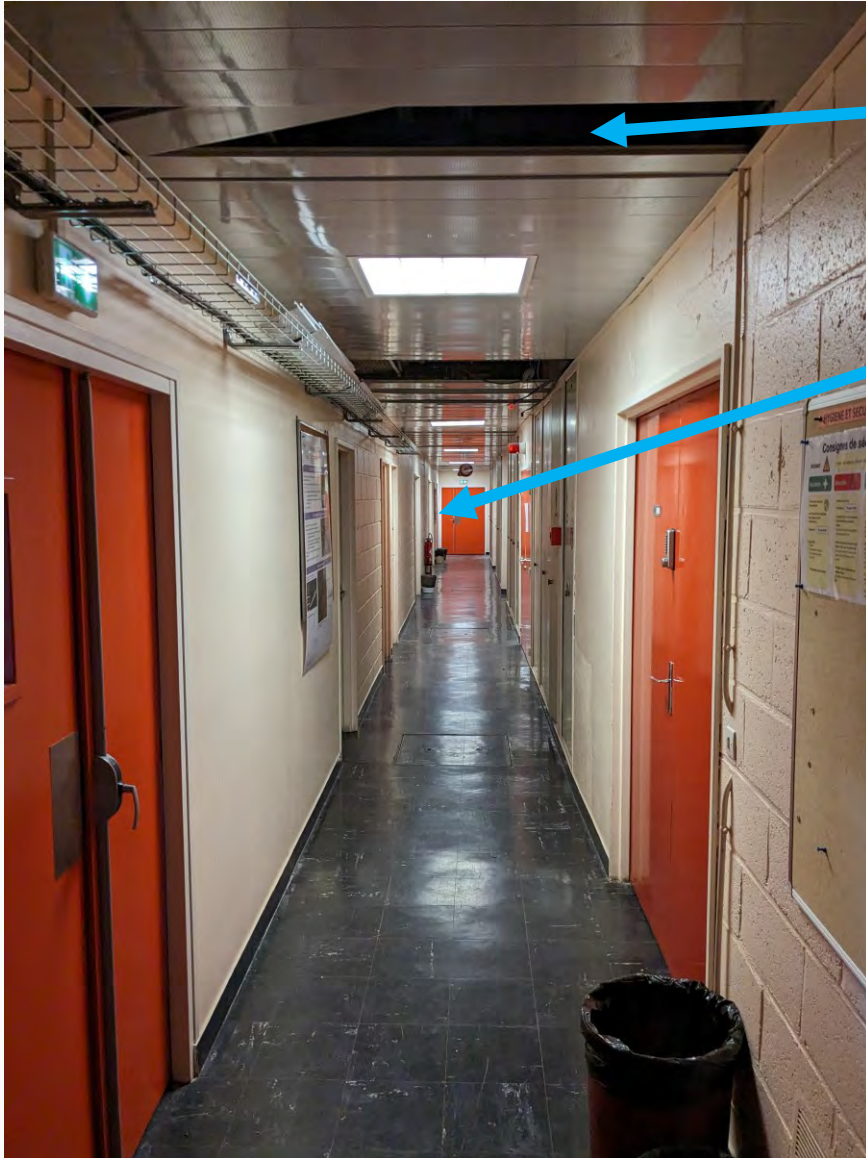
## SESAME BENCH RTC



- 2x 10-cores Intel Xeon E5-2630
- 2x NVIDIA Quadro RTX 8000 + NVlink bridge
- 1x Dual port ConnectX-6 Dx
- And a lot of noise...
- The AO team:



## SESAME BENCH RTC



We messed up a bit the ceiling while passing the fibers...

So it ended up there

The AO Team after the rearrangement



# SESAME BENCH: LET'S CLOSE THE LOOP!

The screenshot displays the Sesame bench monitoring interface, which is divided into several sections:

- General / Sesame**: Shows the current display configuration (PSF + plotDM + shapeDM + calPyr) and the source (zinzolin.obspm.fr).
- Bench monitoring**: A central dashboard with status indicators for Zinzolin Status, ALPAO Status, Acharya Status, Poseidon Status, and Bench Status. Key metrics include:
  - Zinzolin CPU & RAM**: Graph showing CPU (15.7%) and RAM (43.4%) usage.
  - ALPAO power**: Values of 2.73, 2.61, 4.15, 3.72.
  - Framerate**: 500.0 Hz.
  - RTC Pupil model**: dodecagonELT.
  - Slopes #**: 28320.
  - Actuators #**: 3230.
  - Exposure**: 2.0 ms.
  - Loop Gain**: Values of 0.0200 for Tip/Tilt/HOmin/HOmed/HOmax.
  - Loop**: OPEN.
  - ConnectDM**: ON.
- RTD**: Real-time data plots for calPyr, PSF, plotDM, and shapeDM.
- Figure 2@zinzolin.obspm.fr**: A plot showing a value of 1.00 over time (0 to 1750).
- IPython**: A terminal window showing the command `In [309]:` repeated five times.

# SESAME BENCH: LET'S CLOSE THE LOOP AGAIN!

RTC Control and Monitoring Tool

File Edit View Tools Help

Deployment Overview

Deployment Set: sesame Deploy Undeploy

ddt_server	NotReady				
dt_cmat_computation	NotReady				
dt_imat_computation	NotReady				
hrtc_sup_comp	NotReady				
hrtc_sup_hofilt	NotReady				
hrtc_sup_rec	NotReady				
repub_calibrated_pixels	NotReady				
repub_loop_data	NotReady				
repub_raw_pixels	NotReady				
rtc_sup	Initialising				
sub_calibrated_pixels	NotReady				
sub_loop_data	NotReady				
sub_raw_pixels	NotReady				

Sesame - Grafana

localhost:3000/d/ARrhmuLMz/sesame?orgId=1&refresh=5s&var-ho...

General / Sesame

Display PSF + calPyr from localhost

Bench monitoring

Zinzolin Status				ALPAO Status					
OthersCameras	OthersHardware	SLMSupervisor	TdF1	pyroServer	ALPAO#1	ALPAO#2	ALPAO#3	ALPAO#4	Dispatcher
Acharya Status		Poseidon Status		Bench Status					
RTCTkDeployment	pyPyroTkosmic	krakenManager		Cameras	Laser	PCsesame	SLM		
Zinzolin CPU & RAM			ALPAO power	Framerate	RTC Pupil m...	Slopes #	Actuators #	Useful links	
			3.36 3.11 5.06 4.61	500.0 Hz	dodecagonELT	28320	3230	TdF 2.0 Link LogBook GoogleDoc	
			Exposure	Loop Gain [Tip/Tilt/HOmin/HOme...]	Loop	ConnectDM			
			2.0 ms	0 0 0 0 0	OPEN	ON			
			#1 #2 #3 #4						

> RTD (1 panel)

> ADOPT Log (2 panels)

```
IPython: home/sesame x eltdev@acharya:~ x + - □ x
In [2]: # Initialise and enable RTC components
...: gateway.init("rtc_sup")
...: gateway.enable("rtc_sup")
...:
```

## WHAT IF YOU FORGOT TO SHUTDOWN THE BENCH ?

	STATUS	PID	pname	state	C#	tstart	tmux	sess	loopcnt	Descr
47	ACTIVE	0897887	MarlinContainer_telemetry	RUN	C0	09:03:31.172		krakenContainer_	0173088972	
46	ACTIVE	0897829	MarlinConfigDaemon_teleme	INIT	C1	09:03:29.530		krakenContainer_	0000000000	
45	ACTIVE	0897187	MarlinContainer_reconstru	RUN	C0	09:03:27.466		krakenContainer_	0173088972	
44	ACTIVE	0897128	MarlinConfigDaemon_recons	INIT	C1	09:03:25.813		krakenContainer_	0000000000	
43	ACTIVE	0896443	MarlinBU_kacou	RUN	C0	09:03:23.975		krakenBusiness_k	0173088972	
42	ACTIVE	0896365	MarlinConfigDaemon_kacou	INIT	C1	09:03:22.308		krakenBusiness_k	0000000000	
41	ACTIVE	0895717	MarlinBU_acquisitionReord	RUN	C1	09:03:20.461		krakenBusiness_a	0173088987	
40	ACTIVE	0895648	MarlinConfigDaemon_acquis	INIT	C1	09:03:18.837		krakenBusiness_a	0000000000	

- 173M frames @ 500 Hz → 4 days run ! (Yes, it was a long week-end...)
- 15 frames lost → 0.0000087 % loss
- Hopefully, AO Team said it was barely acceptable...



## CONCLUSION & FUTURE WORKS

- H-RTC development is well advanced: already on bench
  - Performance on specs
  - Reliable
  - Maintainable
- Current activities mainly focused on S-RTC: RTC Tk integration & testing
- Next steps:
  - Logging strategy for H-RTC: CII integration ?
  - Continue developing RTC Tk components
  - Development on simulation setup & Testing on bench