The Compute and Control for Adaptive Optics (cacao) real-time control software package

Arnaud Sevin,
Julien Bernard,
Damien Gradatour

Observatoire de Paris
(France)

Software engineering & development
Integration with COMPASS simulation environment, RT hardware (GPU, FPGA) and RT processes management/monitoring

SCExAO Subaru Coronagraphic Extreme Adaptive Optics

Olivier Guyon, Julien Lozi,
Nour Skaf

Subaru Telescope / SCExAO
(US, Japan)

cacao development and On-sky testing for ExAO applications

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Hatem Ltaief & Dalal Sukkari
KAUST (Saudi Arabia)

HPC expertise
Develop and provide linear algebra libraries for Machine Learning

Users / co-developers

Keck Observatory
Sylvain Cetre

W. M. KECK OBSERVATORY
Maunakea, Island of Hawai'i

Facility-class AO (NGS & LGS)

Kernel project/OCA
Frantz Martinache

KERNEL

Focal plane WFS/C

MagAO-X
Jared Males

MagAO-X

Extreme-AO

Subaru Telescope
Christophe Clergeon

SUBARU TELESCOPE
National Astronomical Observatory of Japan

Facility-class AO (NGS & LGS)

THE UNIVERSITY OF SYDNEY

Alison Wong & Barnaby Norris

Machine learning / AI
Neural Networks

cacao's goals

Support today & tomorrow's off-the-shelf powerful computing hardware
→ manycore systems (CPU, GPU) and FPGAs
→ high performance computing engine to requirements of large scale AO systems

Support and enable advanced AO systems and algorithms
→ flexible modular software architecture
→ scalable solution
→ built-in (and growing) machine learning support for predictive control, sensor fusion
→ facilitates asynchronous links between sensors and loops
→ full speed telemetry to disk for post-processing
Foreseen applications : Extreme-AO, MCAO, Tomographic AO ...

Community effort, fully open-source
→ no proprietary / closed source roadblocks
→ enables easy/quick implementation of new algorithms
→ adopts standard data stream format

Easy to adopt and use: short path from ideas to real-time implementation
→ abstract away / hide HPC complexity
→ manage challenging real-time timing constraints for the user
→ provide high-level GPU/CPU configuration

Current Status

Provides low-latency to run control loops

→ Use mixed CPU & GPU resources, configured to RTC computer system

On SCExAO, control matrix is 14,000 x 2000. Matrix-vector computed in ~100us using 15% of RTC resources @ 3kHz

Portable, open source, modular, COTS hardware

→ No closed-source driver

→ std Linux install (no need for real-time OS)

→ using NVIDIA GPUs, also working on FPGA use

→ All code on github: **https://github.com/cacao-org/cacao**

Easy for collaborators to improve/add processes

→ Hooks to data streams in Python or C

→ Template code, easy to adapt and implement new algorithms

→ Provide abstraction of link between loops

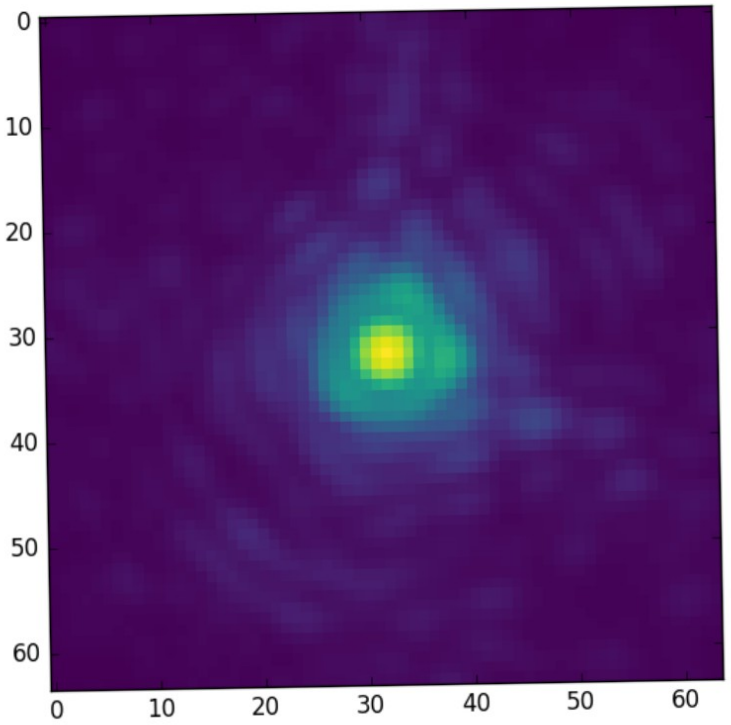
→ Toolkit includes viewers, data logger, low-latency TCP transfer of streams

cacao RTC: current status & on-sky performance for ExAO

Supports high frame rate conventional ExAO operation

SCExAO: 1200 modes corrected at 3.5 kHz (input: 120x120 Pyramid image, output: 2000 actuator DM)

On-sky visible image (750nm), log scale (VAMPIRES)



Supports advanced operation modes for enhanced science performance … Why ?

Mixed CPU/GPU solution provides ample computing power and also supports advanced operation modes:

- **Model-free predictive control** using machine learning approach → **deeper contrast, fainter stars**

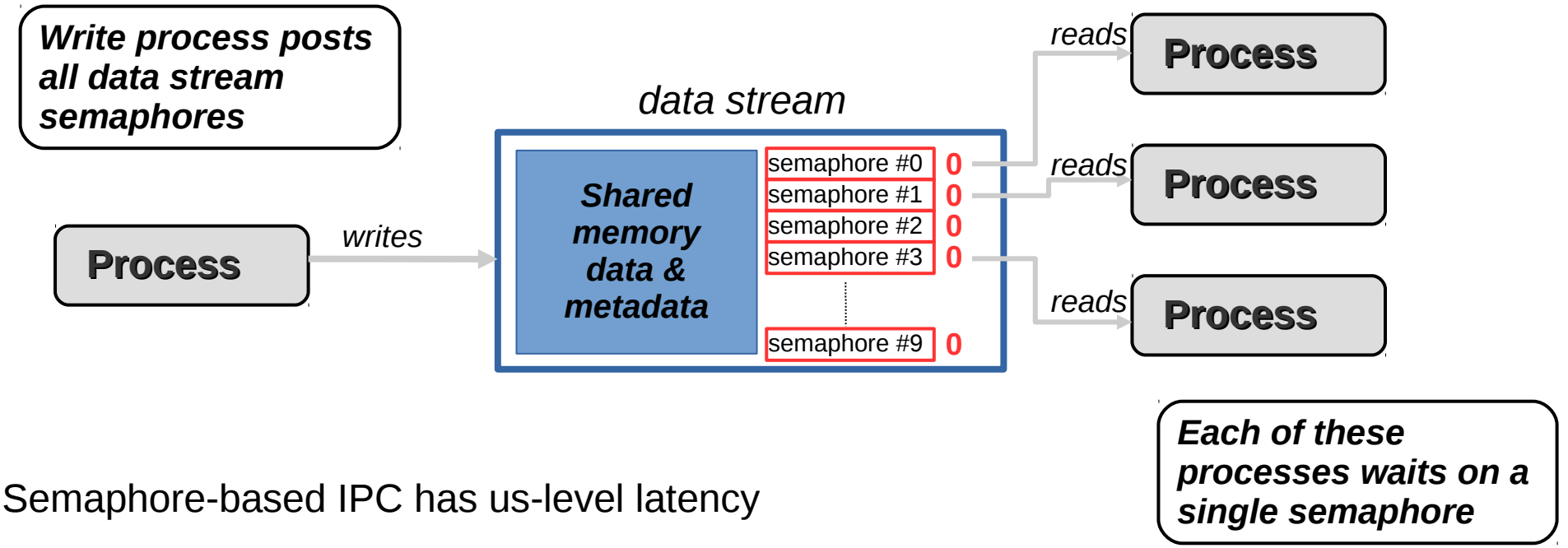
- **On-sky response matrix acquisition** while loop is running (<2 sec to acquire 2000-mode response matrix) → improved calibration → **higher performance control**

- **Real-time links between control loops, sensor fusion** → **speckle control, low-order modes correction**

Main Guiding Principle: data owns IPC
Processes sign up to semaphores

Build AO control loop as a finite set of CPU-mananged processes.
Processes can manage computations on GPGPU(s)

Interprocess communication (IPC) is contained in data: cacao streams are held in shared memory and contain semaphores
→ Complexities of IPC are handled by compiler and Kernel (semaphores, shared memory)



Semaphore-based IPC has us-level latency

A simple example

Frame Grabber camera acquisition

writes

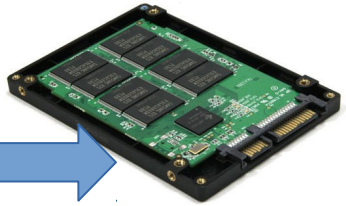
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

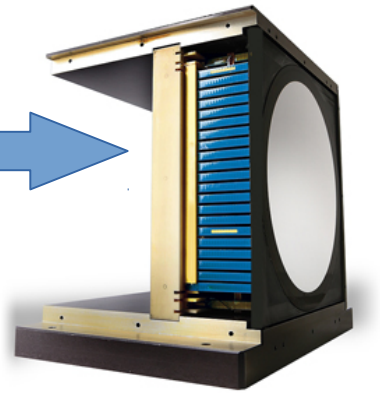
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Cam read

Frame Grabber camera acquisition

writes

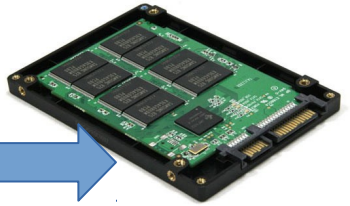
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

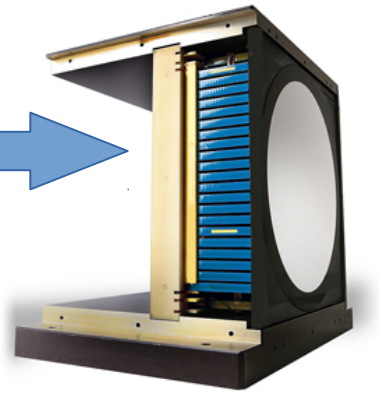
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Process posts semaphores

Frame Grabber camera acquisition

writes

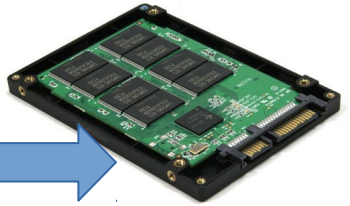
WFS image

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	1
semaphore #3	1
semaphore #9	1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

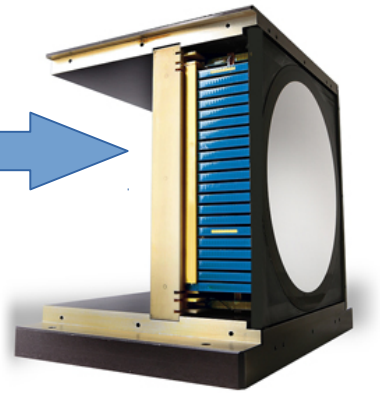
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

DM electronics driver



Semaphores posted

Frame Grabber camera acquisition

writes

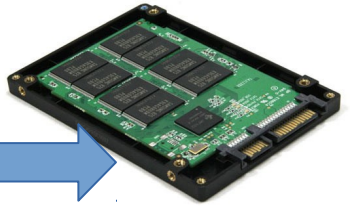
WFS image

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	1
semaphore #3	1
semaphore #9	1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

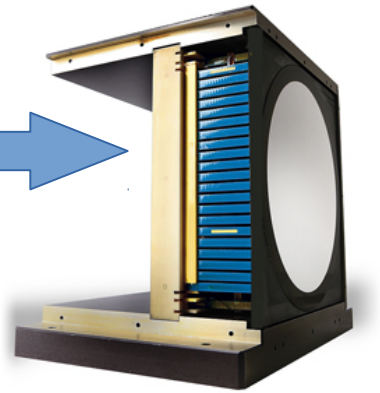
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

DM electronics driver



sem_wait launches next processes

Frame Grabber camera acquisition

writes

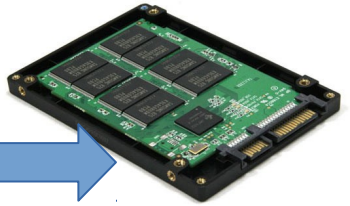
WFS image

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	1
semaphore #3	0
semaphore #9	1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

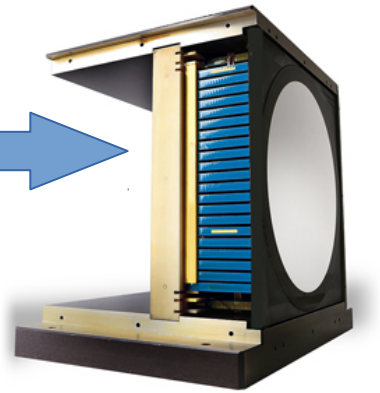
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

DM electronics driver



processes run...

Frame Grabber camera acquisition

writes

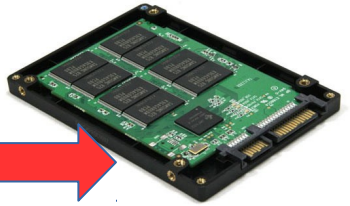
WFS image

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	1
semaphore #3	0
semaphore #9	1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

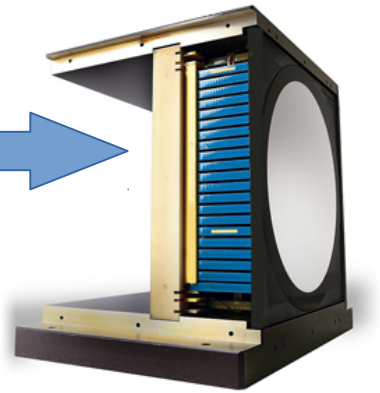
x gain and add

Total DM displacement

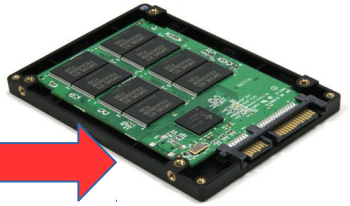
Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

DM electronics driver



… asynchronously



reads

Data logger

WFS image

Frame Grabber camera acquisition

writes

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	1
semaphore #3	0
semaphore #9	1

reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

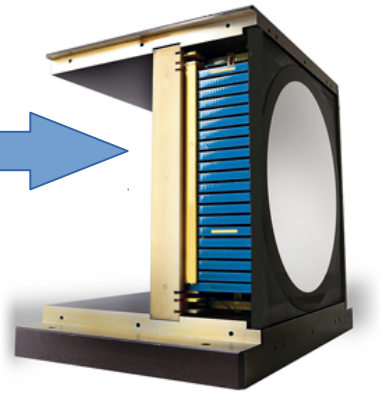
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

DM electronics driver



Frame Grabber camera acquisition

writes

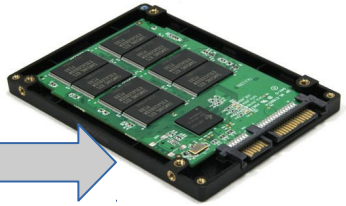
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

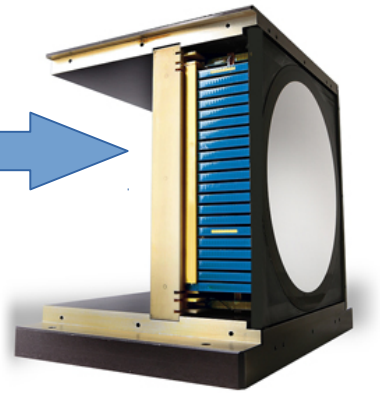
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Frame Grabber camera acquisition

writes

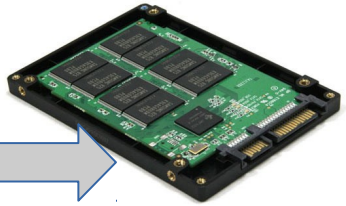
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

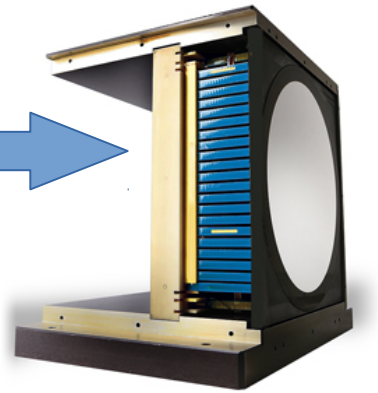
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Frame Grabber camera acquisition

writes

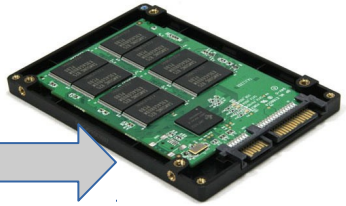
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 1
semaphore #3 1
semaphore #9 1

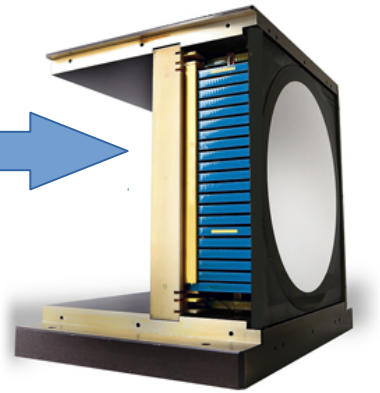
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Frame Grabber camera acquisition

writes

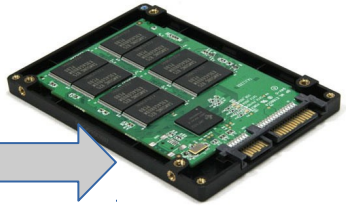
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 1
semaphore #3 1
semaphore #9 1

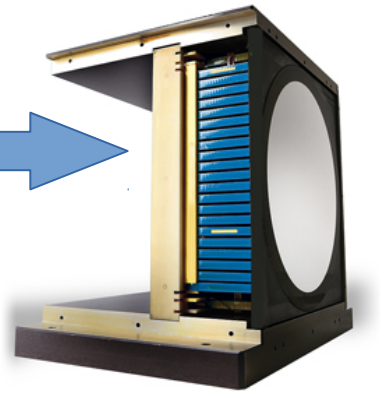
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Frame Grabber camera acquisition

writes

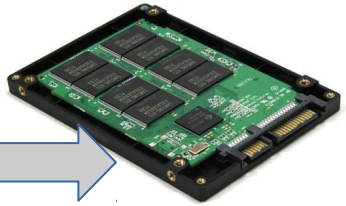
WFS image

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	1
semaphore #3	0
semaphore #9	1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	0
semaphore #3	1
semaphore #9	1

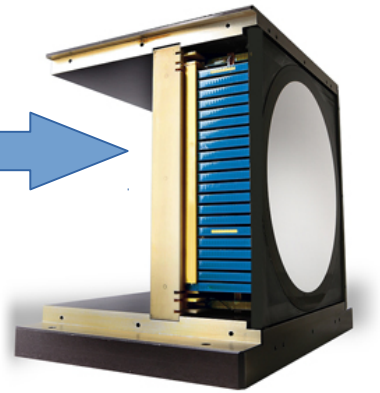
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	0
semaphore #3	0
semaphore #9	0

DM electronics driver



Frame Grabber camera acquisition

writes

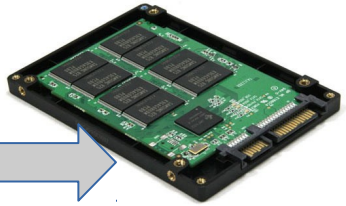
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

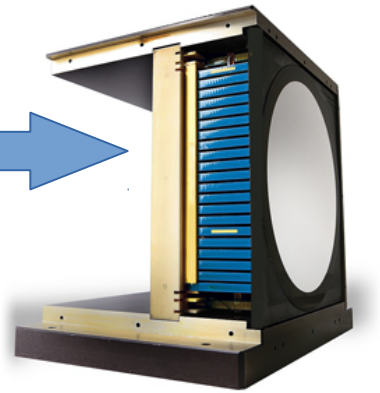
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Frame Grabber camera acquisition

writes

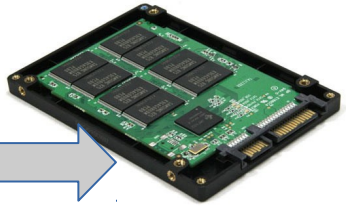
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

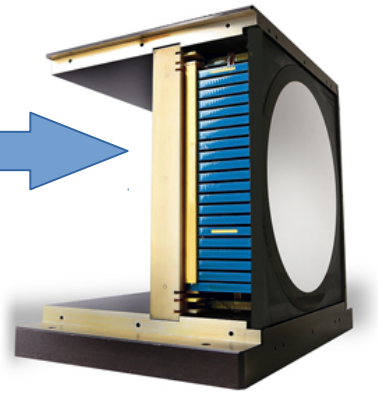
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 0
semaphore #3 0
semaphore #9 0

DM electronics driver



Frame Grabber camera acquisition

writes

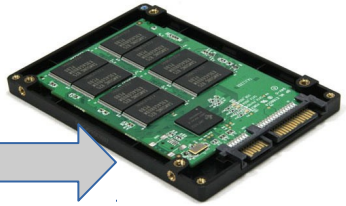
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

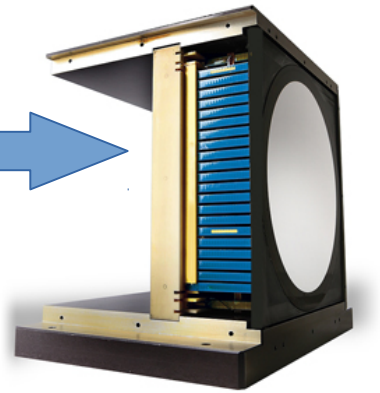
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 1
semaphore #3 1
semaphore #9 1

DM electronics driver



Frame Grabber camera acquisition

writes

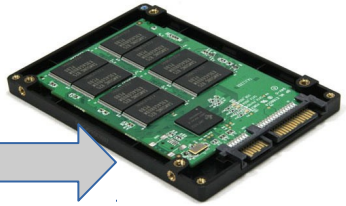
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

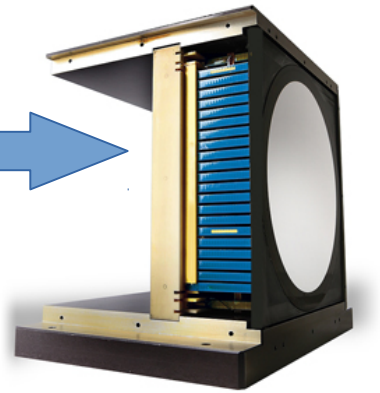
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 1
semaphore #3 1
semaphore #9 1

DM electronics driver



Frame Grabber camera acquisition

writes

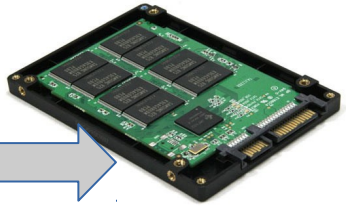
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

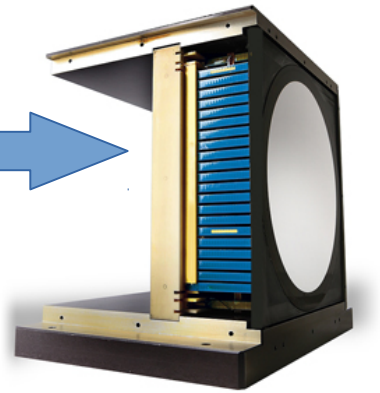
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

DM electronics driver



Frame Grabber camera acquisition

writes

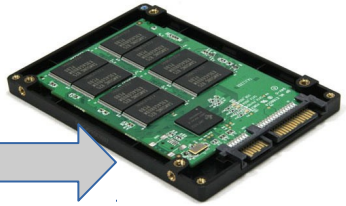
WFS image

Shared memory data & metadata

semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

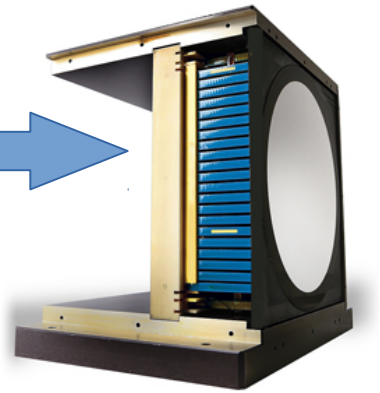
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

DM electronics driver



processes execution can overlap

Frame Grabber camera acquisition

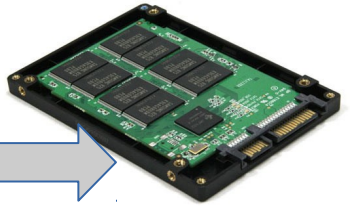
writes

WFS image

Shared memory data & metadata

semaphore #0	0
semaphore #1	0
semaphore #2	1
semaphore #3	0
semaphore #9	1

reads **Data logger**



reads **Send to other computer**

reads **Multiply by Control Matrix**

incremental DM displacement

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	0
semaphore #3	1
semaphore #9	1

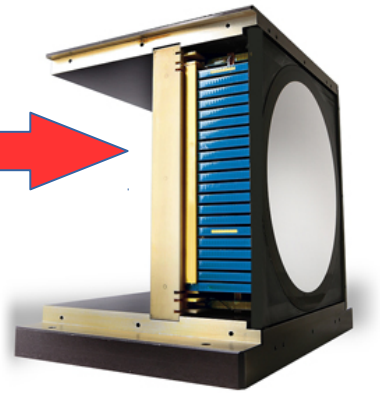
x gain and add

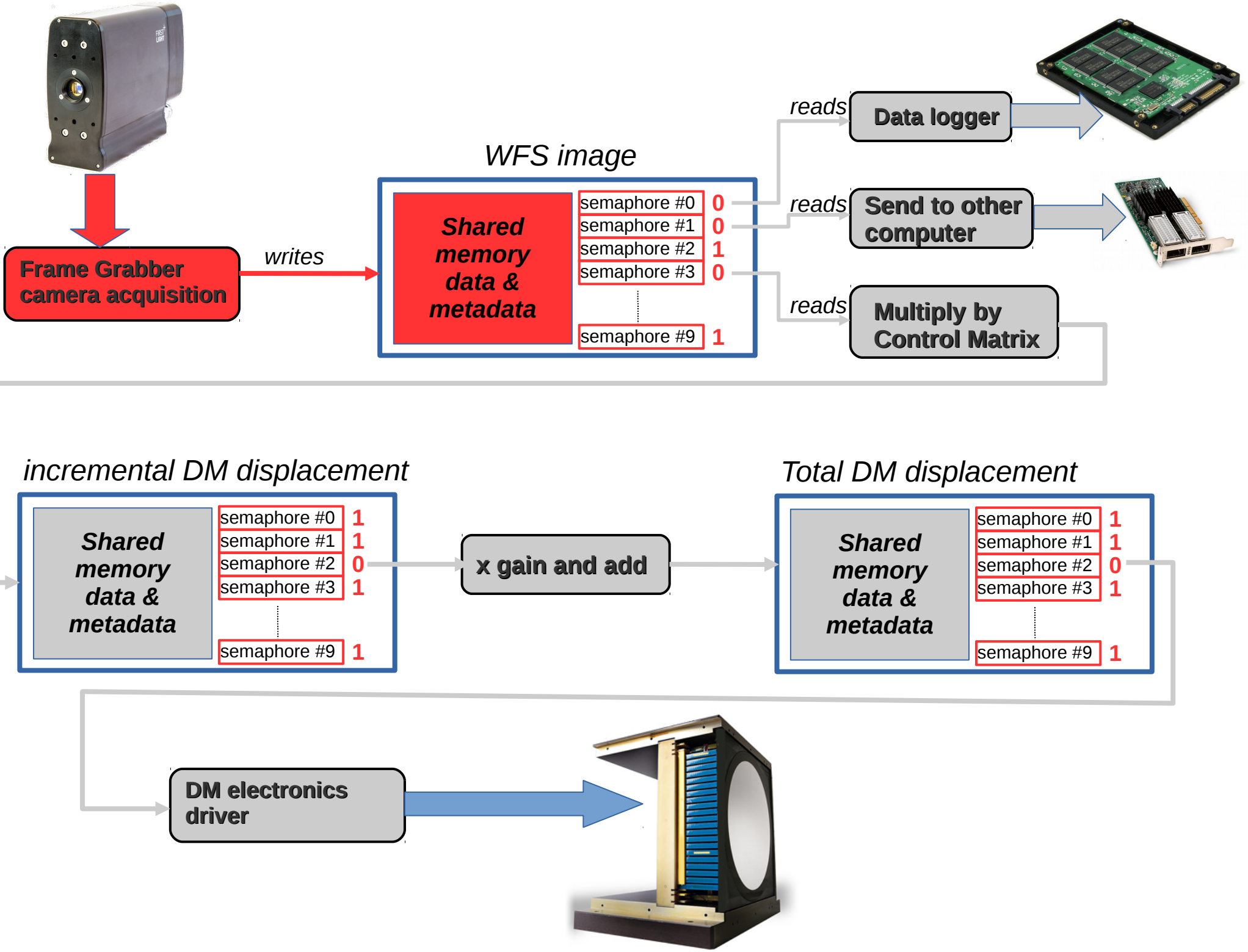
Total DM displacement

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	0
semaphore #3	1
semaphore #9	1

DM electronics driver




WFS image
Frame Grabber camera acquisition
writes
Shared memory data & metadata
semaphore #0 0
semaphore #1 0
semaphore #2 1
semaphore #3 0
semaphore #9 1
reads
Data logger
reads
Send to other computer
reads
Multiply by Control Matrix
incremental DM displacement
Shared memory data & metadata
semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1
x gain and add
Total DM displacement
Shared memory data & metadata
semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1
DM electronics driver

Unused semaphores go >1

Frame Grabber camera acquisition

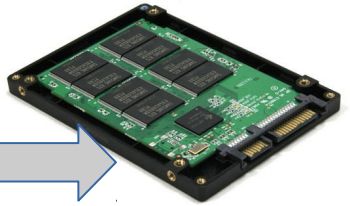
writes

WFS image

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	2
semaphore #3	1
semaphore #9	2

reads **Data logger**



reads **Send to other computer**

reads **Multiply by Control Matrix**

incremental DM displacement

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	0
semaphore #3	1
semaphore #9	1

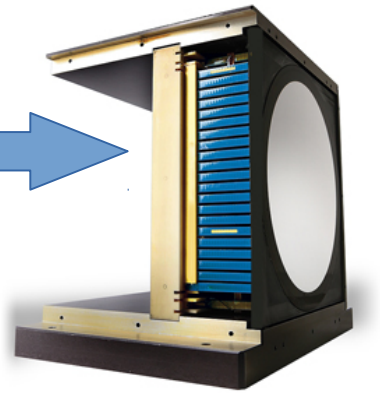
x gain and add

Total DM displacement

Shared memory data & metadata

semaphore #0	1
semaphore #1	1
semaphore #2	0
semaphore #3	1
semaphore #9	1

DM electronics driver



Frame Grabber camera acquisition

writes

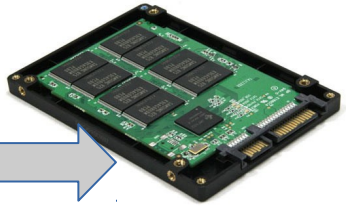
WFS image

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 2
semaphore #3 1
semaphore #9 2

reads

Data logger



reads

Send to other computer

reads

Multiply by Control Matrix

incremental DM displacement

Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

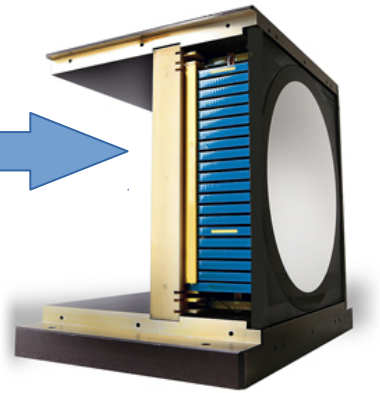
x gain and add

Total DM displacement

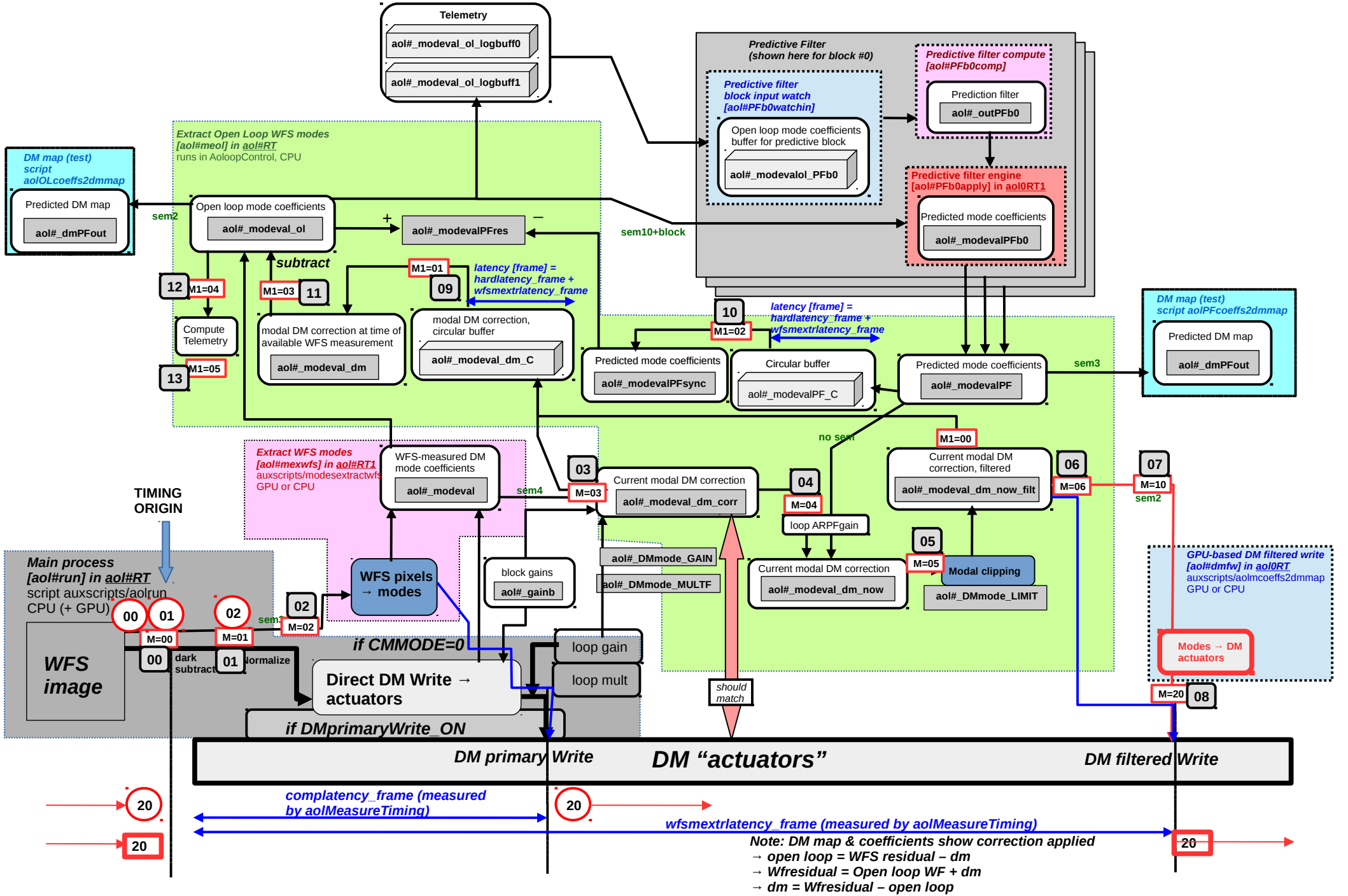
Shared memory data & metadata

semaphore #0 1
semaphore #1 1
semaphore #2 0
semaphore #3 1
semaphore #9 1

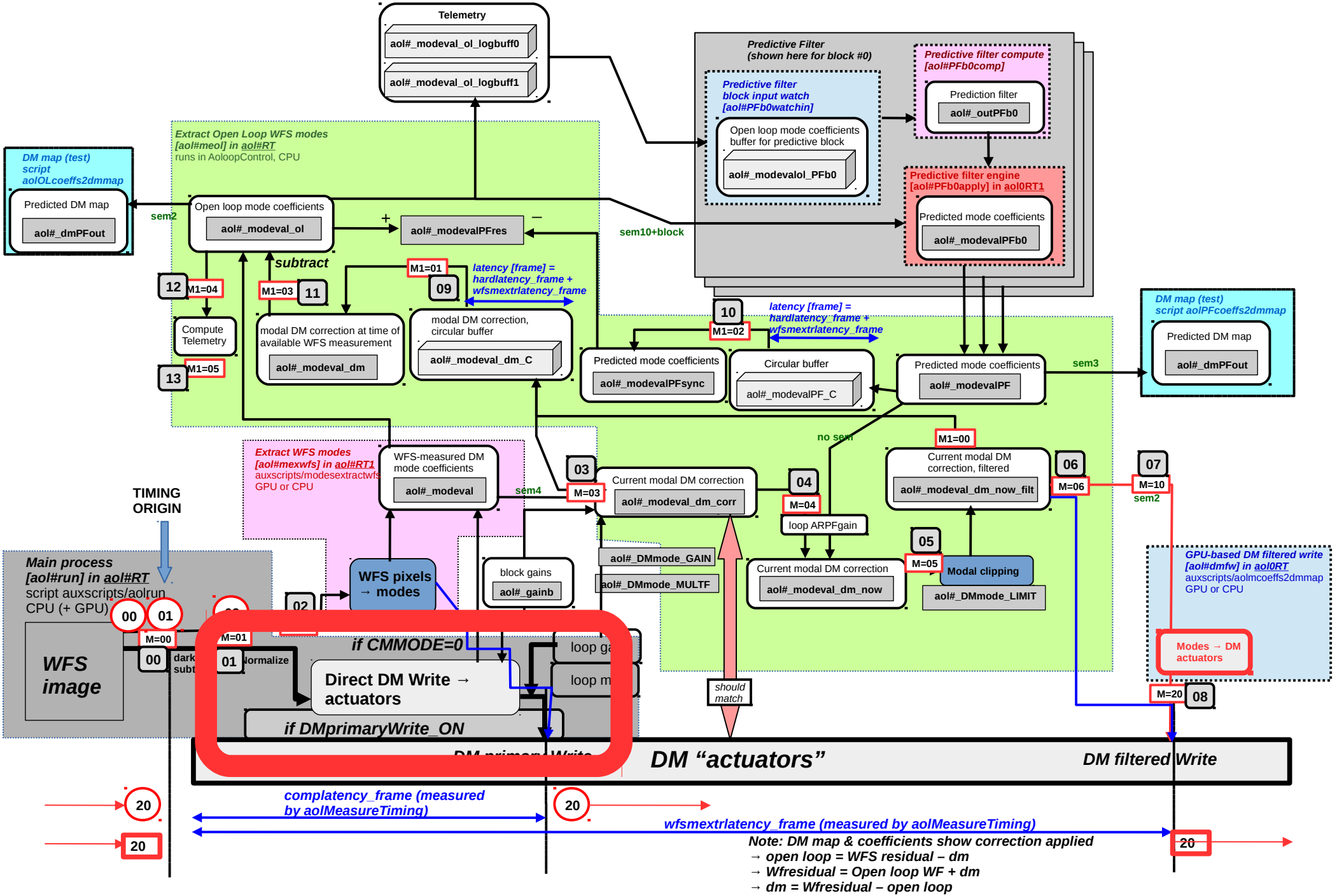
DM electronics driver



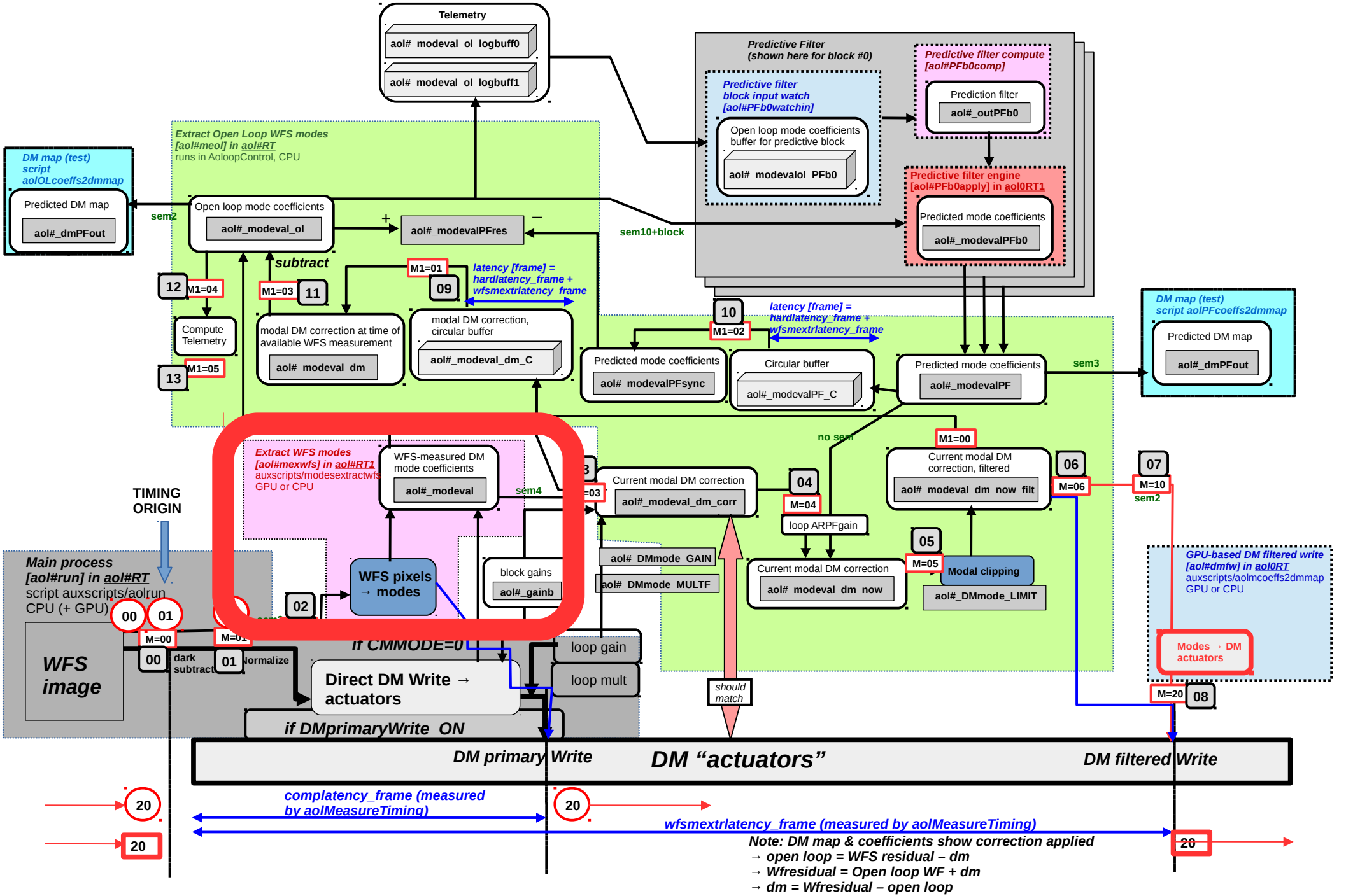
A more powerful example (SCExAO control loop)



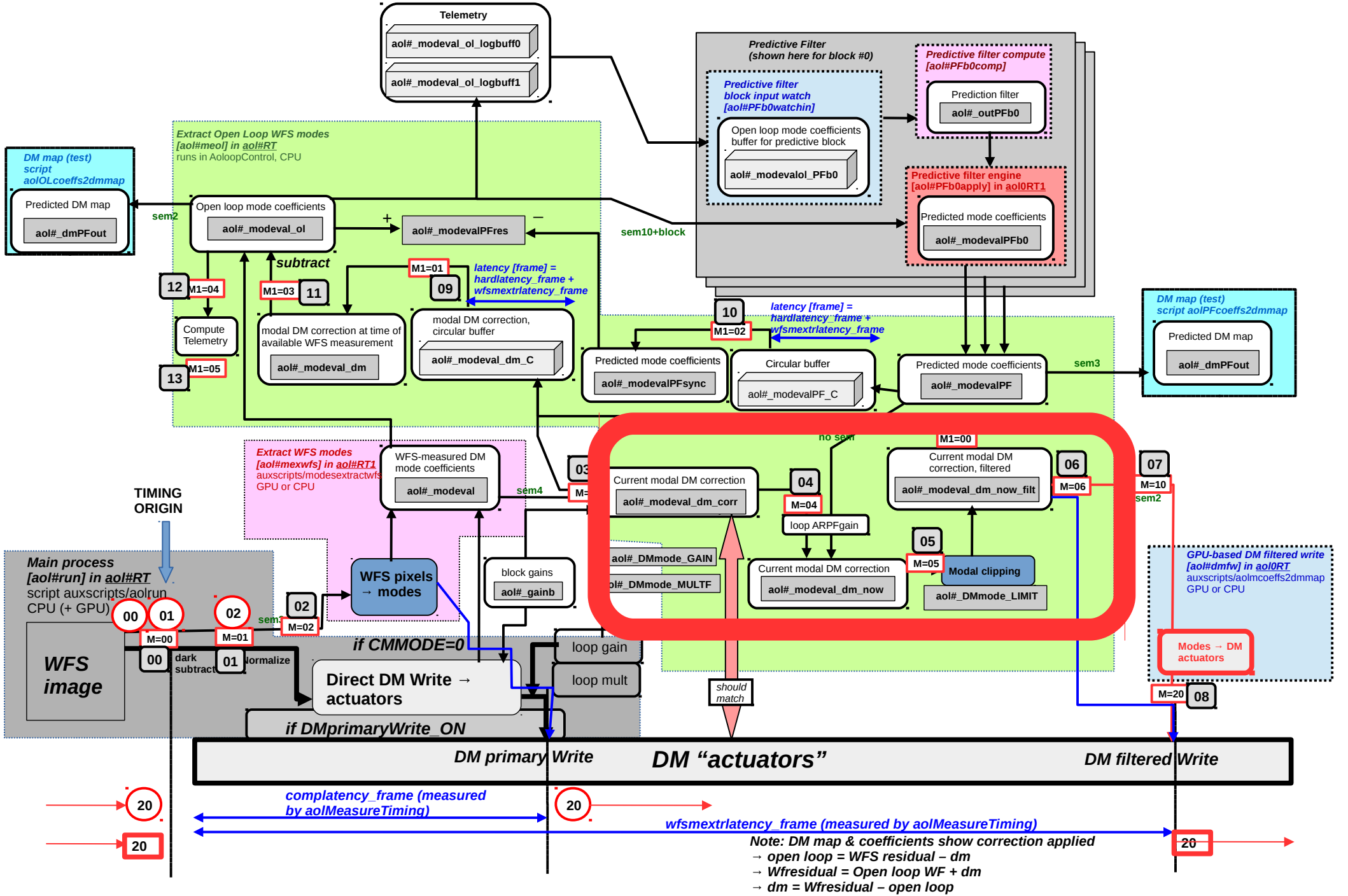
Fastest way to write on DM is to multiply WFS image by control matrix → DM actuators



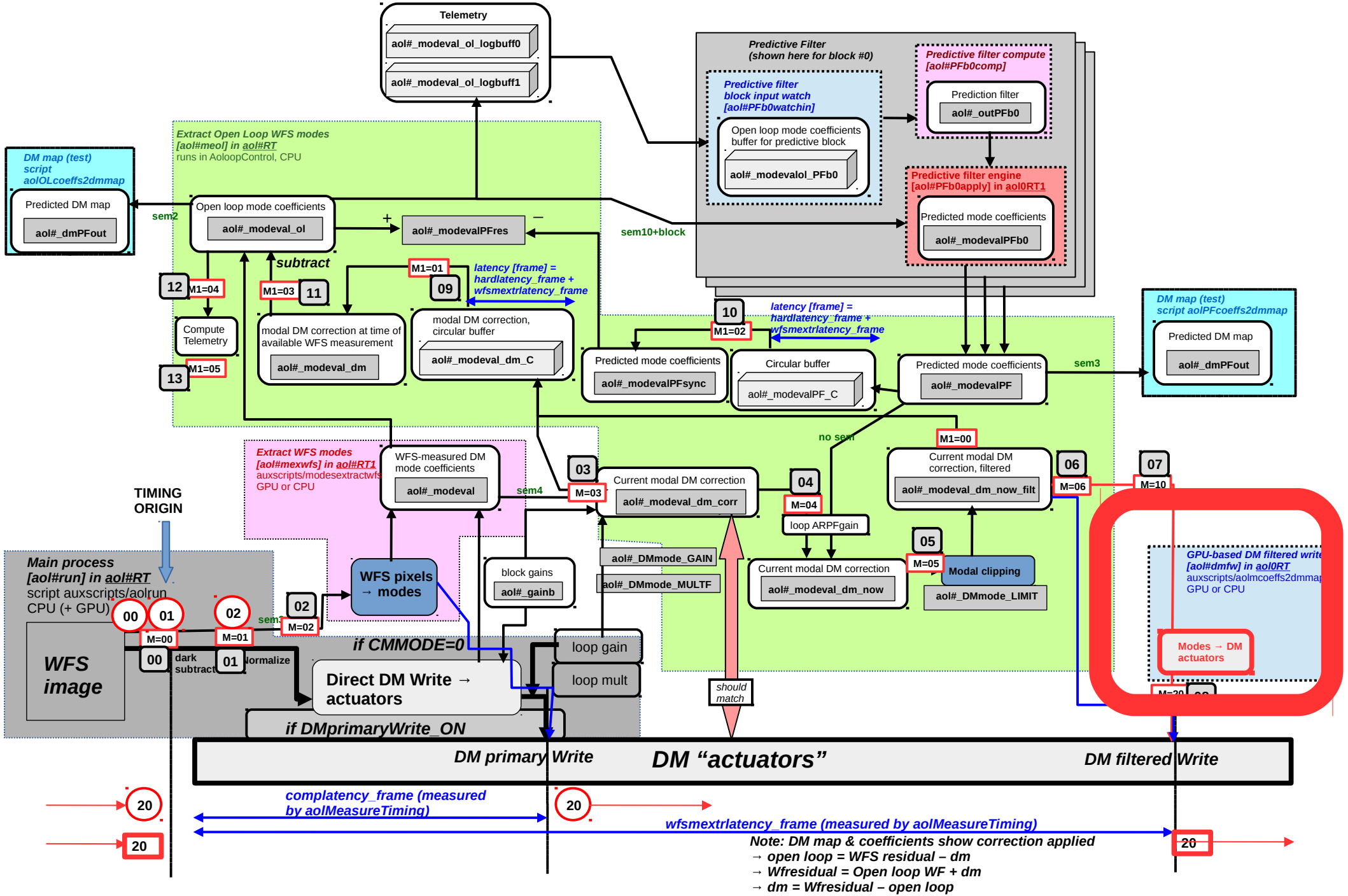
Modal reconstruction: compute WF modes from WFS image (usually GPU-based)



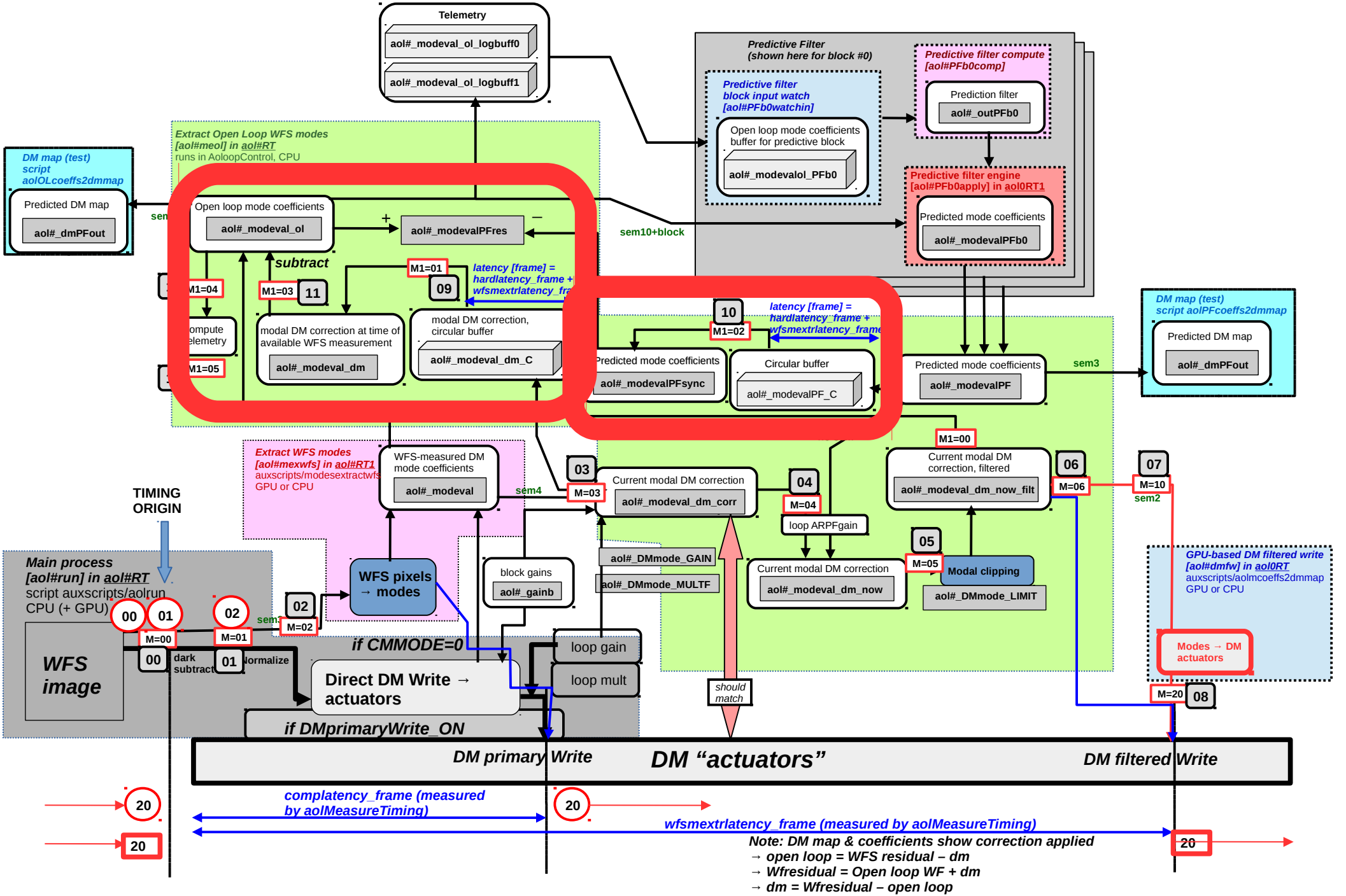
Modal filtering



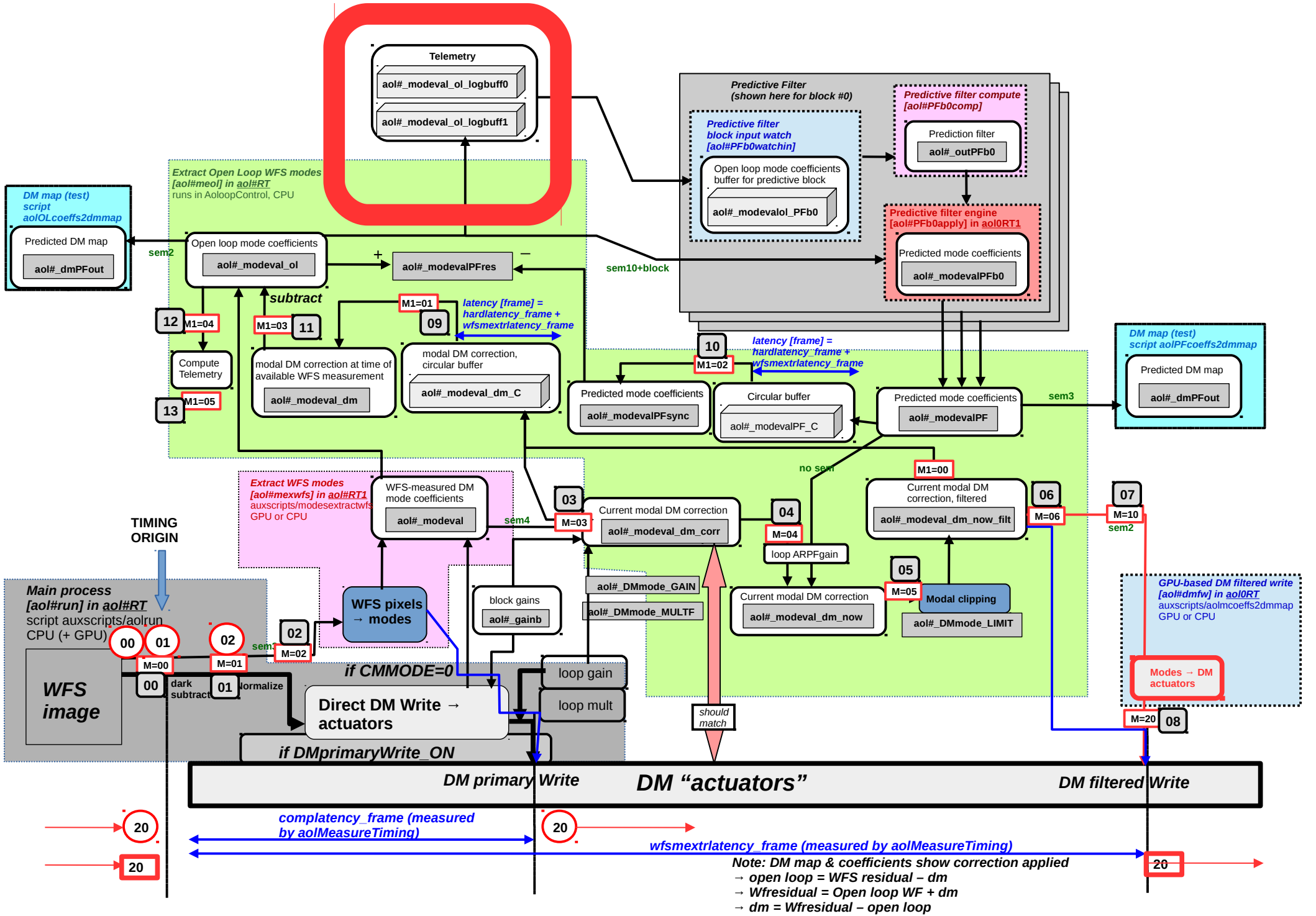
Modes → DM actuators (MVM operation)



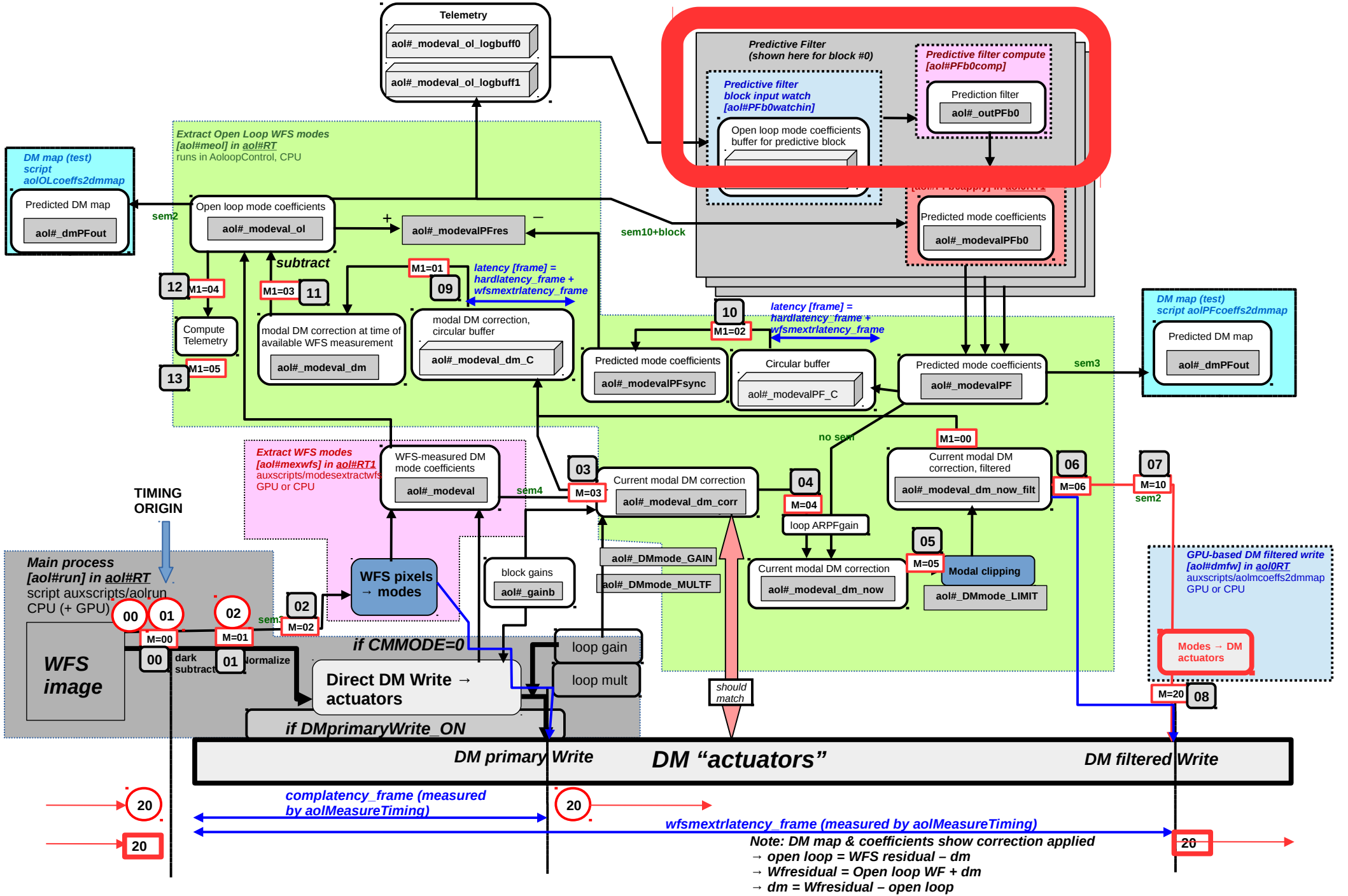
Modal pseudo-open loop reconstruction



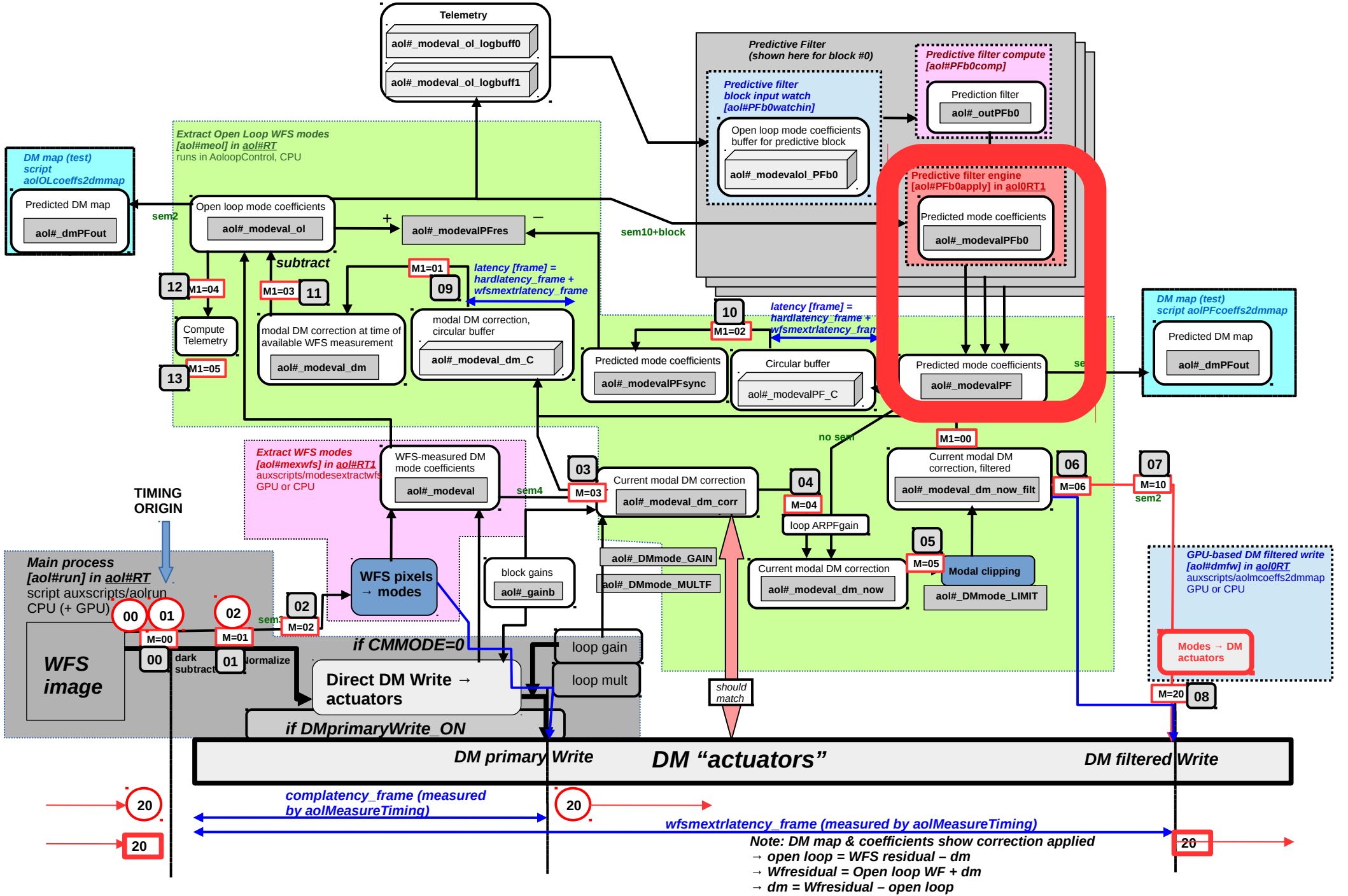
Write modal telemetry buffers



Compute Predictive Filter (soft real-time)



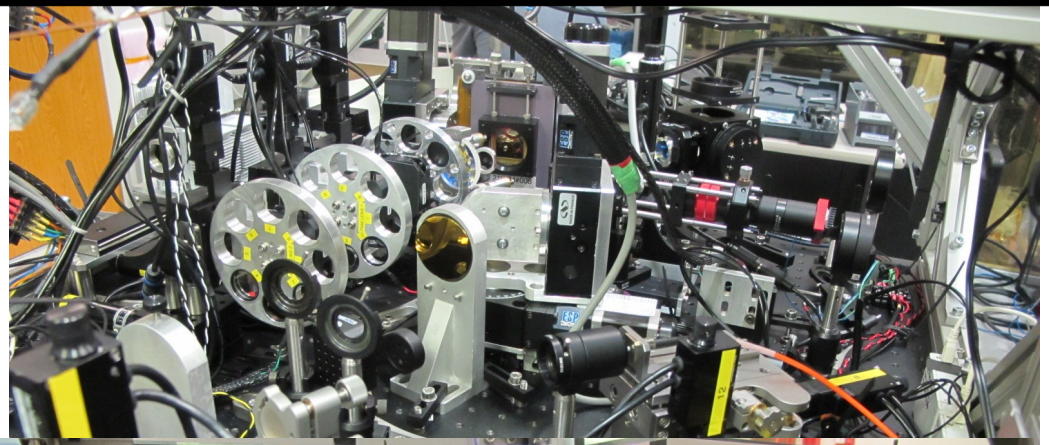
Apply Predictive Filter

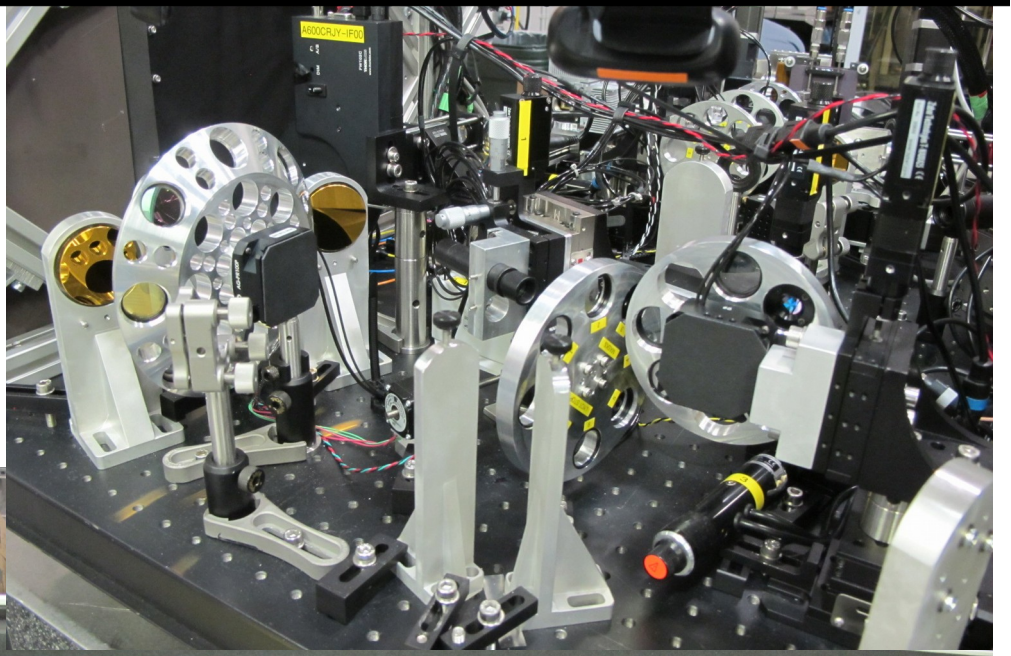


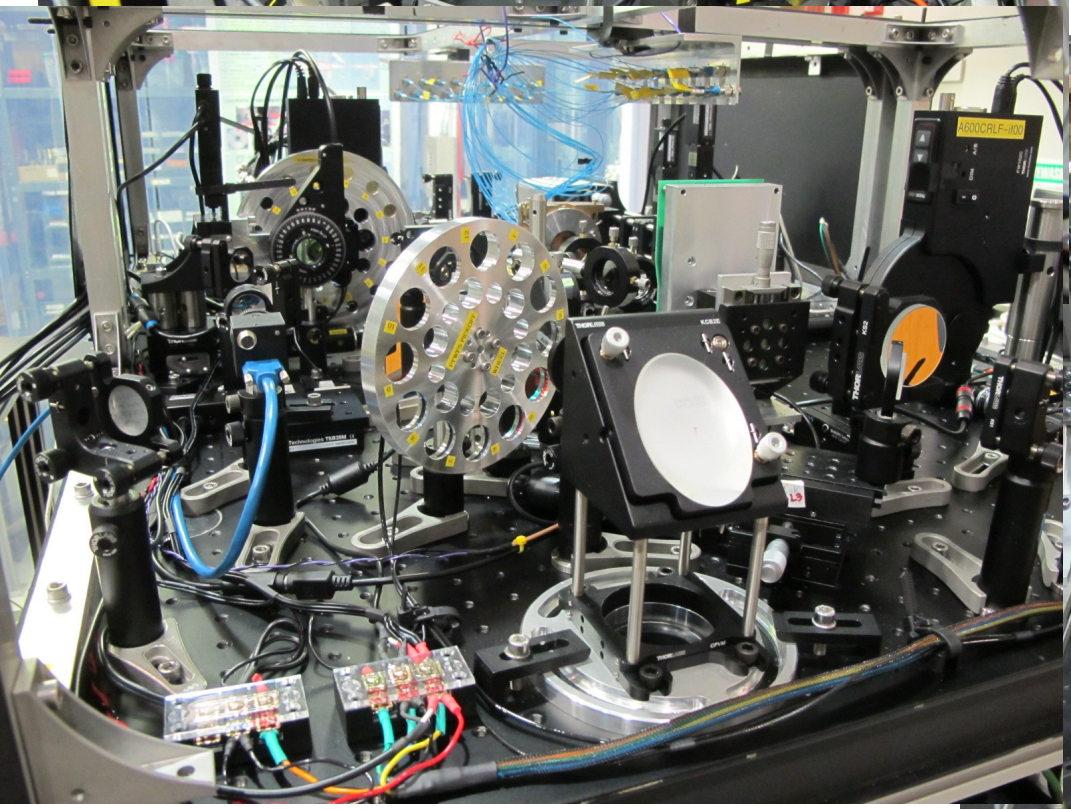
Linking loops / handling multiple WFSs and DMs

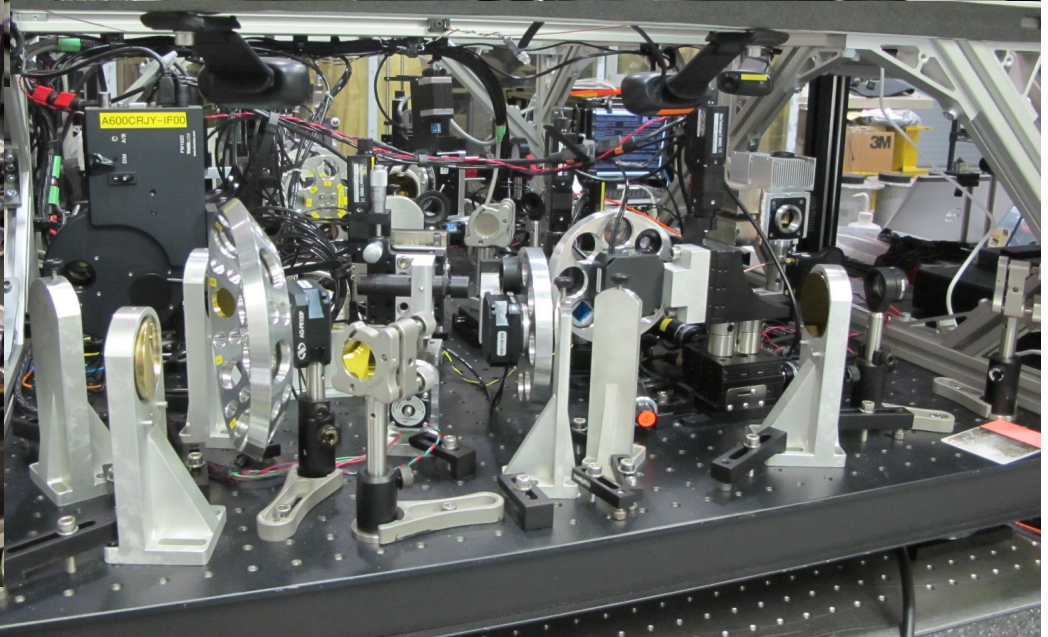
SCExAO

Subaru Coronagraphic Extreme Adaptive Optics

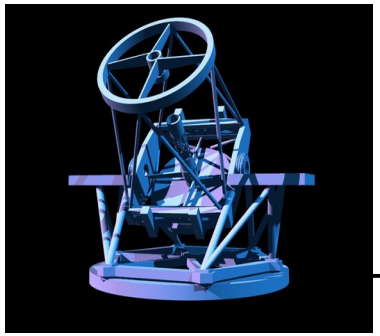








SCExAO Light path



Facility AO

2-5um imager/spectro (IRCS)

2-5um IFU

ADC
1 Hz

64x64 DM
2 kHz

BEAM SWITCHER

50x50 DM
3.5 kHz

Modulated Visible PyWFS
0.4-1.0 um

Weakly/un-modulated NearIR PyWFS
0.8-2.0 um

Weakly/un-modulated Visible PyWFS
0.6-1.0 um

VAMPIRES (2 cameras)
Polarimetry
Dual band
Aperture masking

111 DM
segmented

FIRST
Polarimetry
Interferometry

RHEA
visible IFU

coronagraphic LOWFS

Coronagraph

CHARIS
nearIR IFU

Photonic nuller

MKIDS
focal plane WFS

IRD
1-2 um HR spectrograph

SAPHIRA
Imager

Sci path viewing cam

Active WF correction

Dedicated science instrument

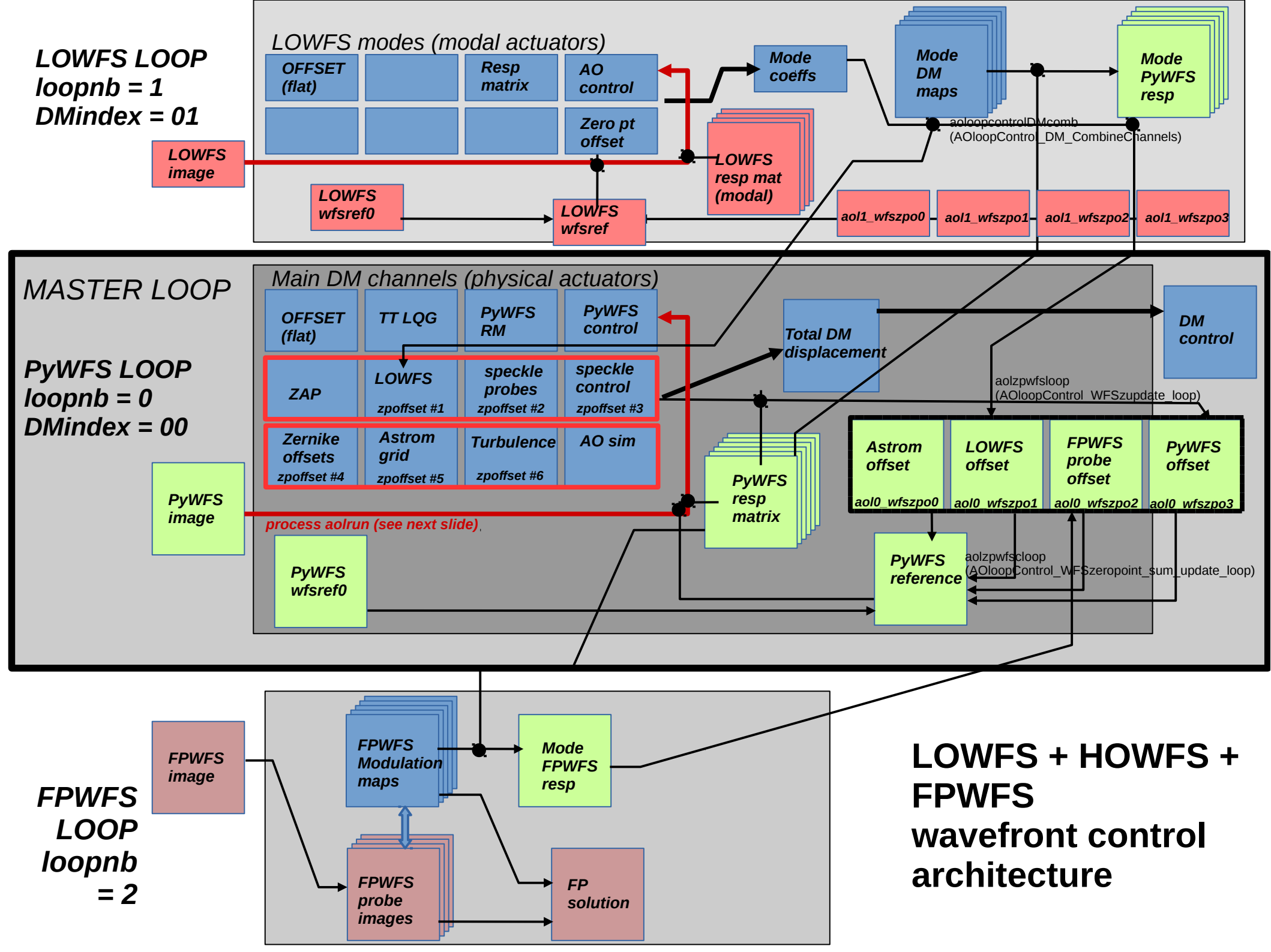
Mixed science/WFS

Dedicated WFS

Visitor port

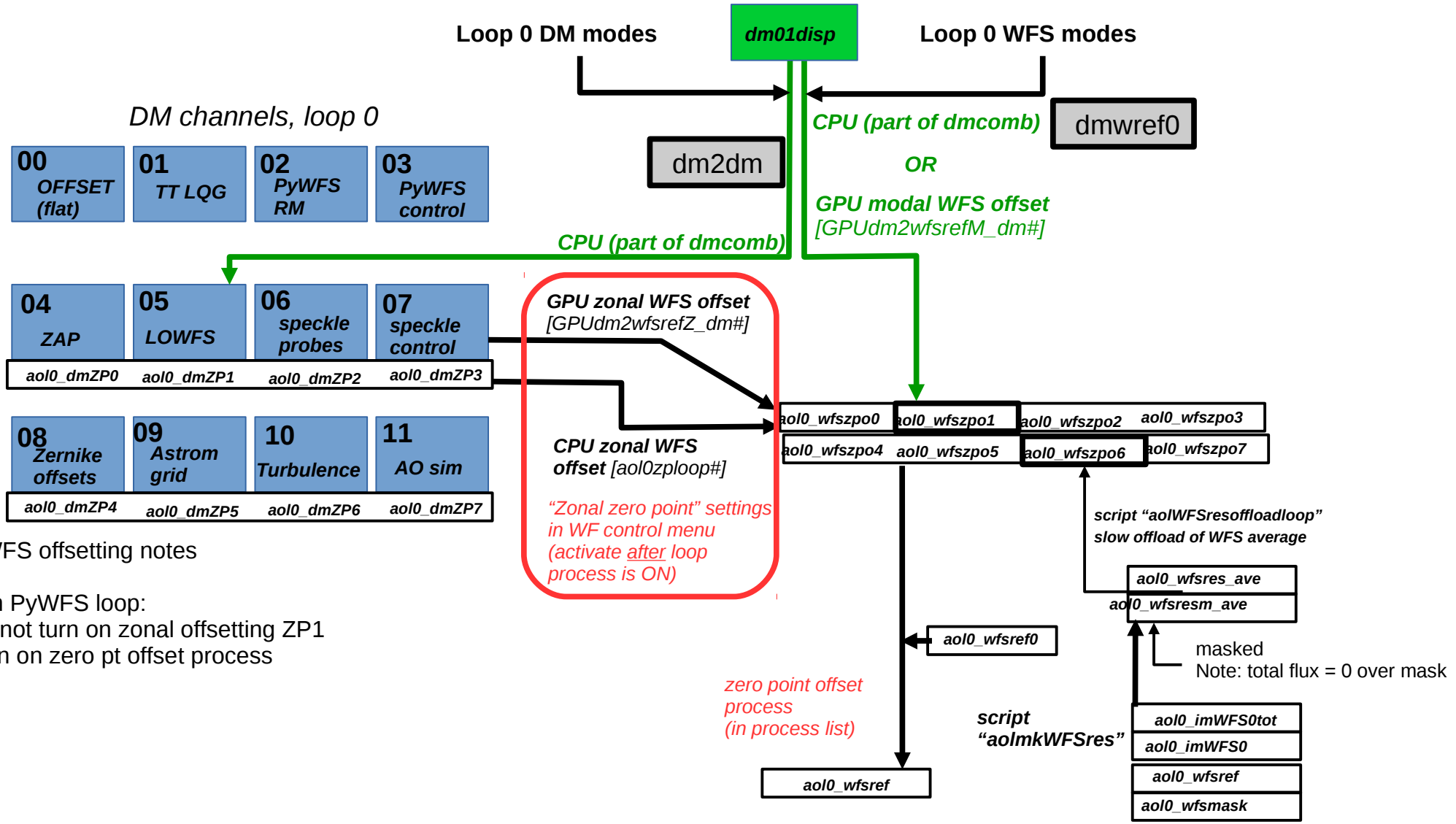
dichroic

beam switch


LOWFS LOOP
loopnb = 1
DMindex = 01
LOWFS modes (modal actuators)
OFFSET (flat)
Resp matrix
AO control
Zero pt offset
Mode coeffs
Mode DM maps
Mode PyWFS resp
aoloopcontrolDMcomb
(AOloopControl_DM_CombineChannels)
LOWFS image
LOWFS resp mat (modal)
LOWFS wfsref0
LOWFS wfsref
aol1_wfszpo0
aol1_wfszpo1
aol1_wfszpo2
aol1_wfszpo3
MASTER LOOP
Main DM channels (physical actuators)
OFFSET (flat)
TT LQG
PyWFS RM
PyWFS control
Total DM displacement
DM control
PyWFS LOOP
loopnb = 0
DMindex = 00
ZAP
LOWFS
zpoffset #1
speckle probes
zpoffset #2
speckle control
zpoffset #3
aolzpwfsloop
(AOloopControl_WFSzupdate_loop)
Zernike offsets
zpoffset #4
Astrom grid
zpoffset #5
Turbulence
zpoffset #6
AO sim
Astrom offset
aol0_wfszpo0
LOWFS offset
aol0_wfszpo1
FPWFS probe offset
aol0_wfszpo2
PyWFS offset
aol0_wfszpo3
PyWFS image
PyWFS resp matrix
process aolrun (see next slide)
PyWFS reference
aolzpwfscloop
(AOloopControl_WFSzeropoint_sum_update_loop)
PyWFS wfsref0
FPWFS LOOP
loopnb = 2
FPWFS image
FPWFS Modulation maps
Mode FPWFS resp
FPWFS probe images
FP solution
LOWFS + HOWFS + FPWFS
wavefront control architecture

Linking multiple control loops (zero point offsetting)

A control loop can offset the convergence point of another loop @> kHz (GPU or CPU)
Example: speckle control, LOWFS need to offset pyramid control loop
THIS IS DONE TRANSPARENTLY FOR USER → don't pay attention to the diagram below !



LOWFS offsetting notes

From PyWFS loop:
- Do not turn on zonal offsetting ZP1
- Turn on zero pt offset process

Timing is everything

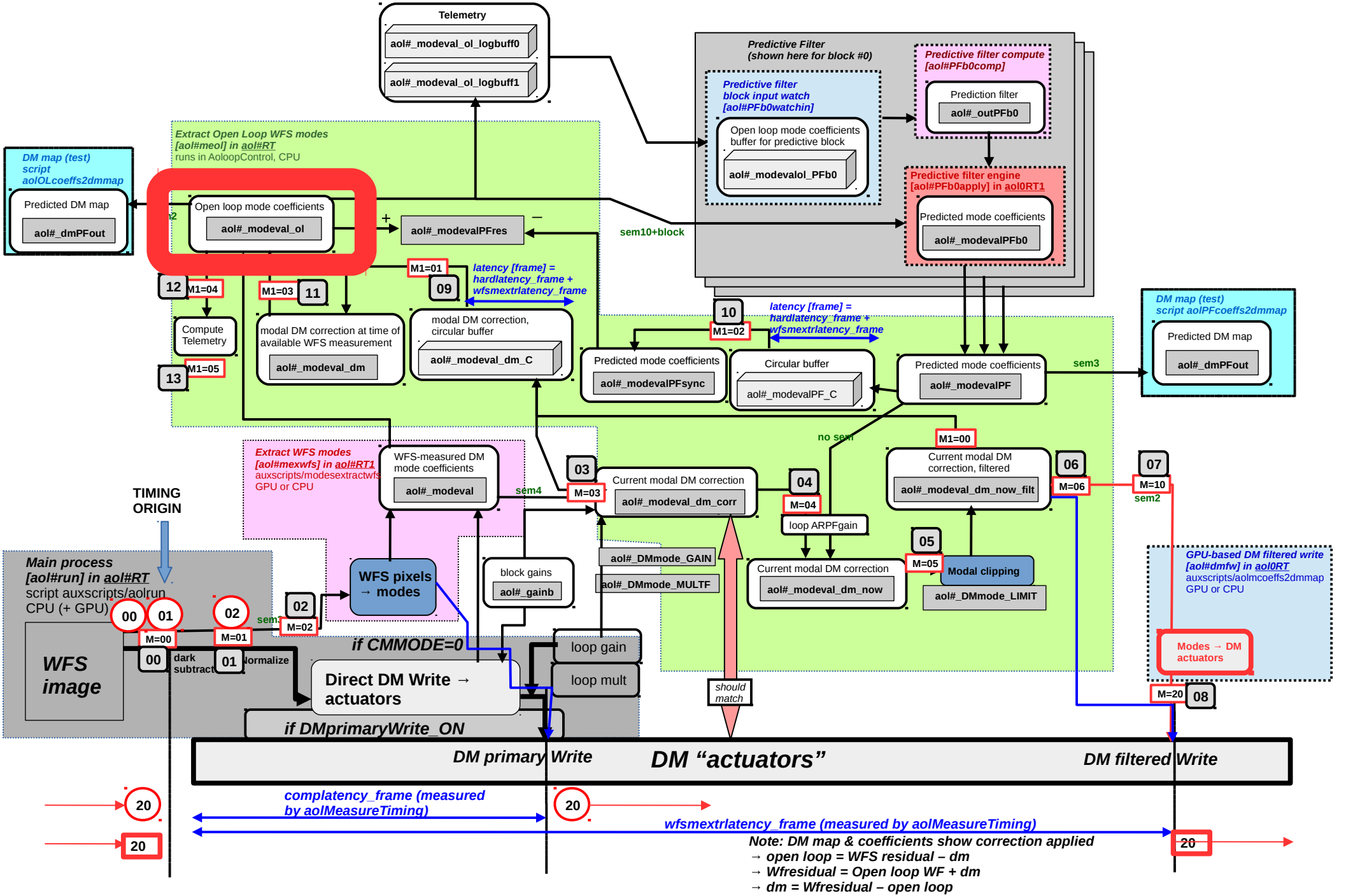
Good timing **knowledge and stability** is essential for:

→ Pseudo-open loop reconstruction

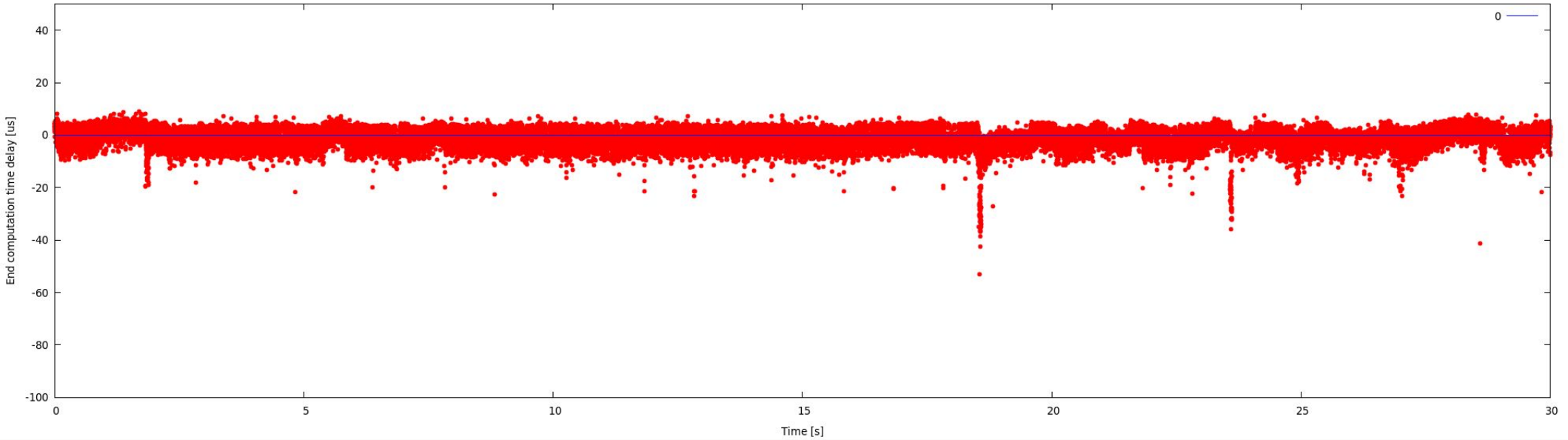
→ Fast response matrix acquisiton

→ Predictive control

End of real-time computation processes



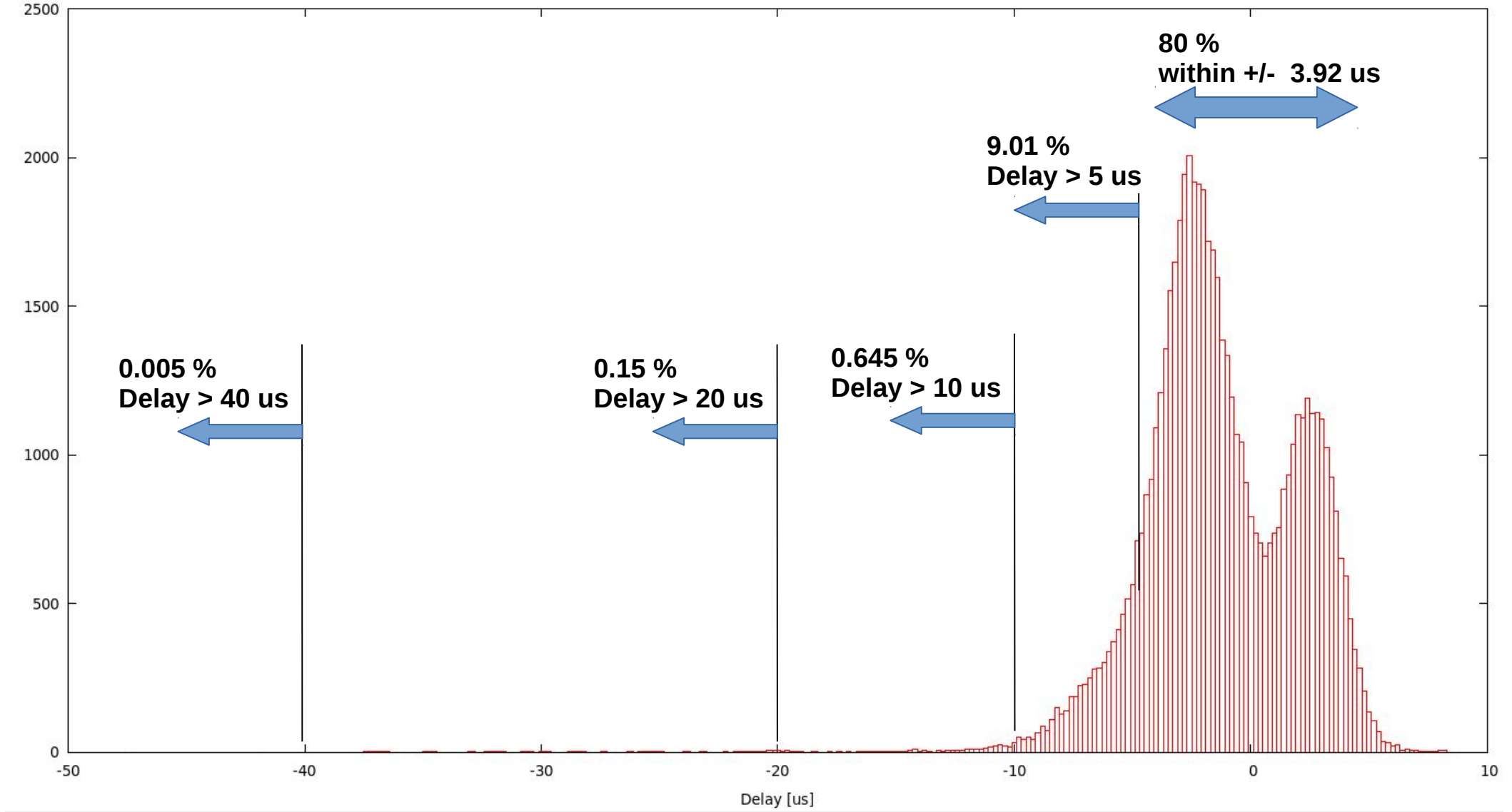
End-to-end Timing Jitter
RMS < 5 us, max delay ~50us



End-to-end timing jitter measured by monitoring completion time of last real-time stream: modal pseudo-open loop coefficients.

Jitter includes following components:
- Hardware synchronization (PyWFS tip-tilt mirror)
- Camera readout
- Data transfer over TCP link
- All real-time computations, CPU and GPU
- Time measurement errors

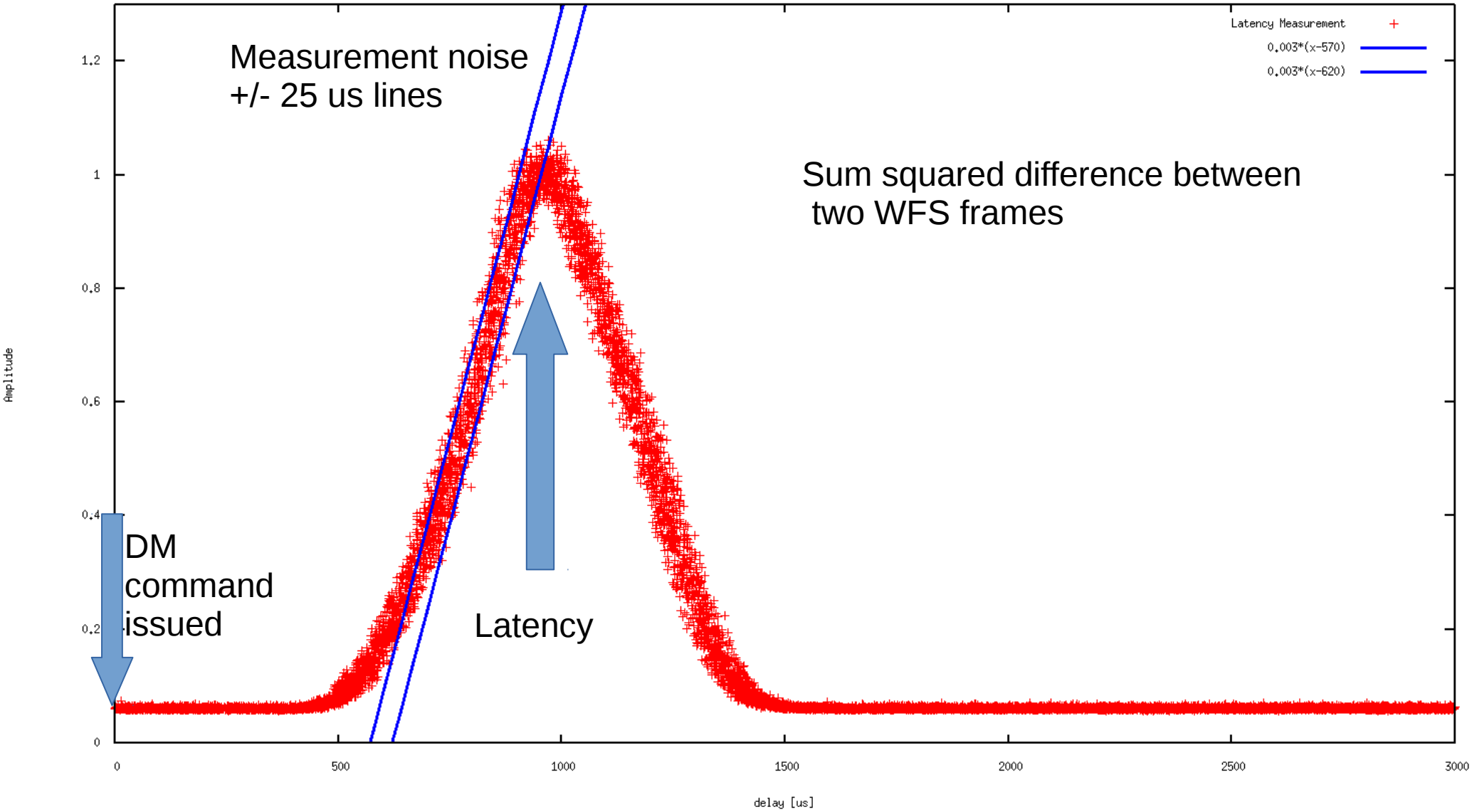
End-to-end Timing Jitter Histogram, measured @ 2kHz



10% of loop iteration @ 2 kHz

Hardware Latency measured on SCExAO

Time between DM command issued and corresponding WFS signal observed
(Camera readout + TCP transfer + processing + DM electronics)



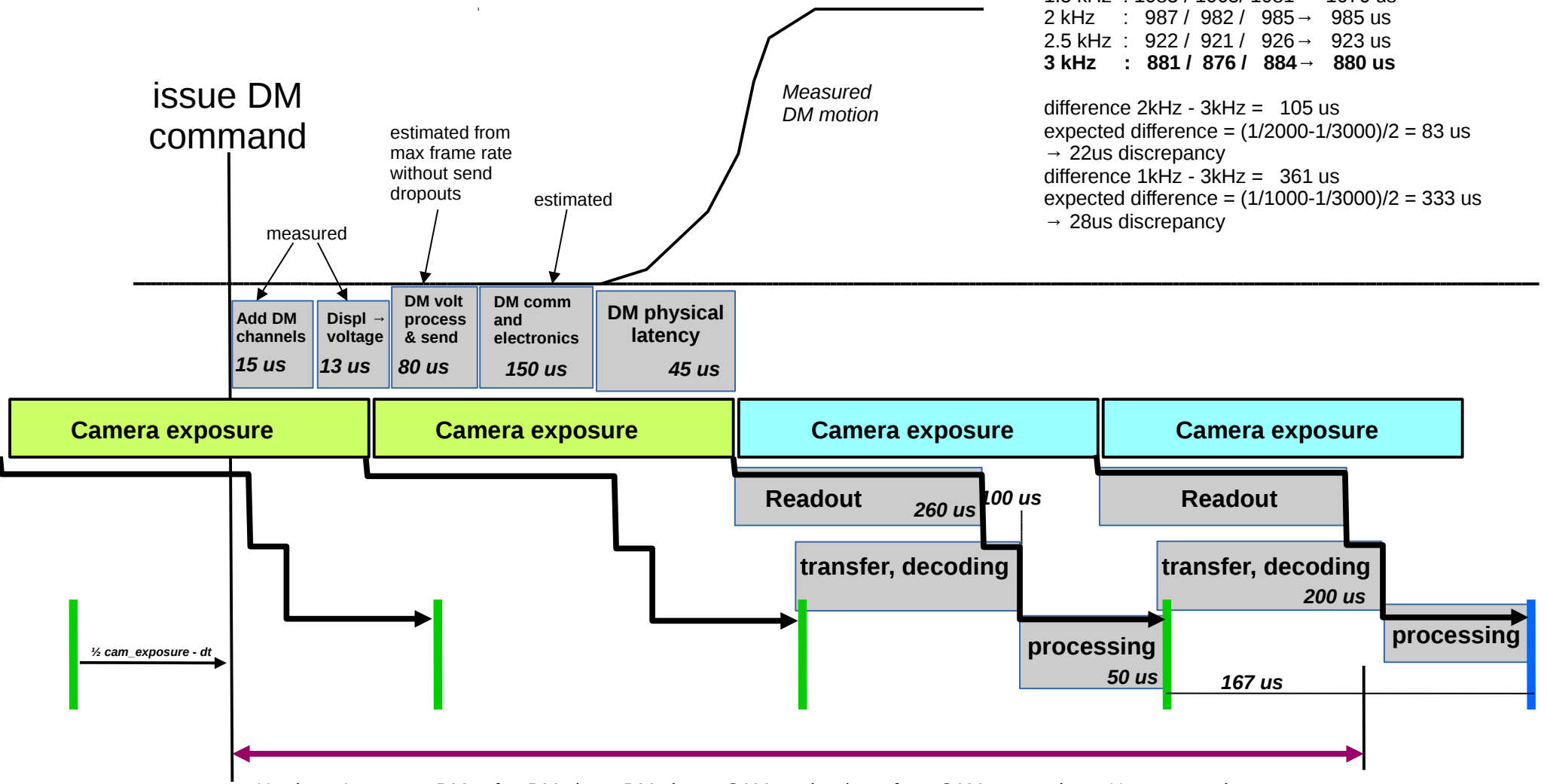
Hardware Latency

Definition:
Time offset between ***DM command issued****, and* ***mid-point between 2 consecutive WFS frames with largest difference***

SCExAO measured hardware latencies:

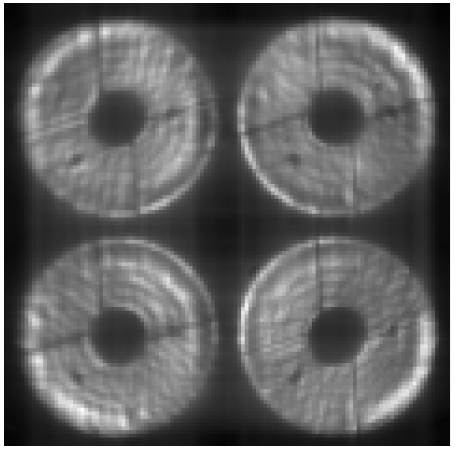
1 kHz : 1253 / 1260/ 1269 → 1261 us
1.5 kHz : 1083 / 1065/ 1081 → 1076 us
2 kHz : 987 / 982 / 985 → 985 us
2.5 kHz : 922 / 921 / 926 → 923 us
3 kHz : 881 / 876 / 884 → 880 us

difference 2kHz - 3kHz = 105 us
expected difference = (1/2000-1/3000)/2 = 83 us
→ 22us discrepancy
difference 1kHz - 3kHz = 361 us
expected difference = (1/1000-1/3000)/2 = 333 us
→ 28us discrepancy



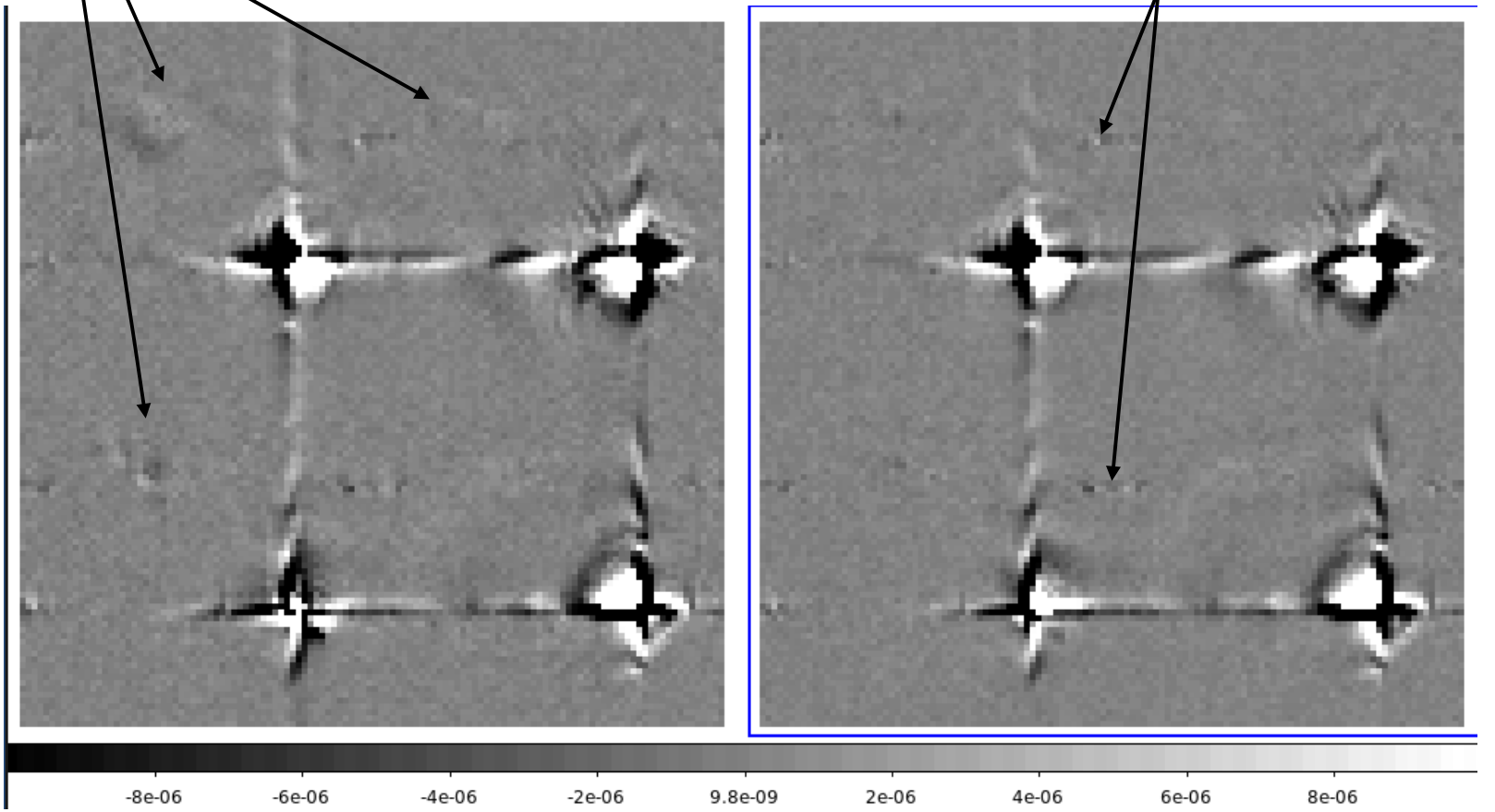
HardwareLatency = DM soft + DM elec + DM phys + CAM readout/transfer + CAM processing + ½ exposure time
HardwareLatency = N x cam_exposure + dt

Fast RM acquisition (4000 Hadamard pokes in 2s @ 2 kHz) + Removing temporal DM response from response matrix by using two poke sequences



Temporal bleeding from previous poke pattern (should be removed from RM)

Camera readout RF coupling between pixels ~1% electronic ghosts at 2kHz frame rate Needs to be kept in RM

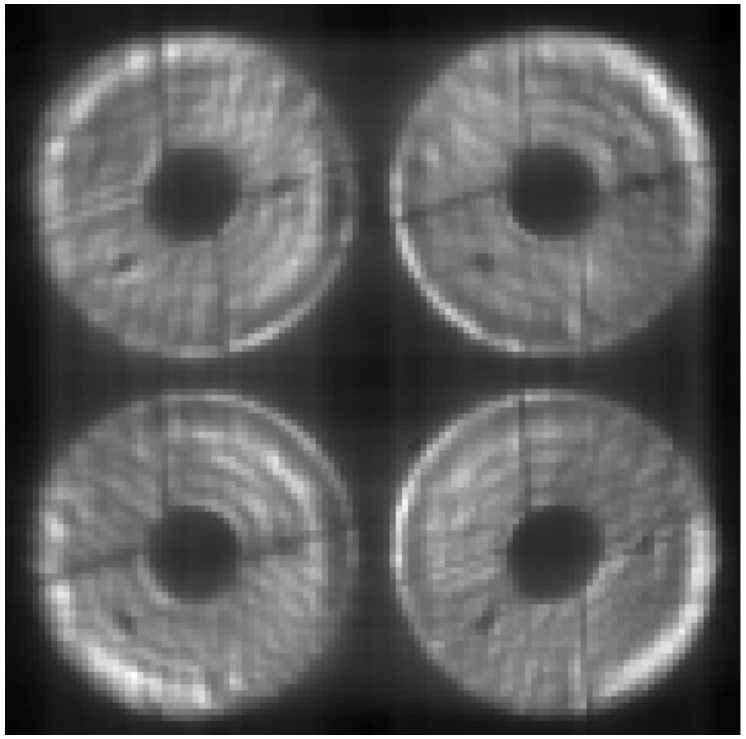


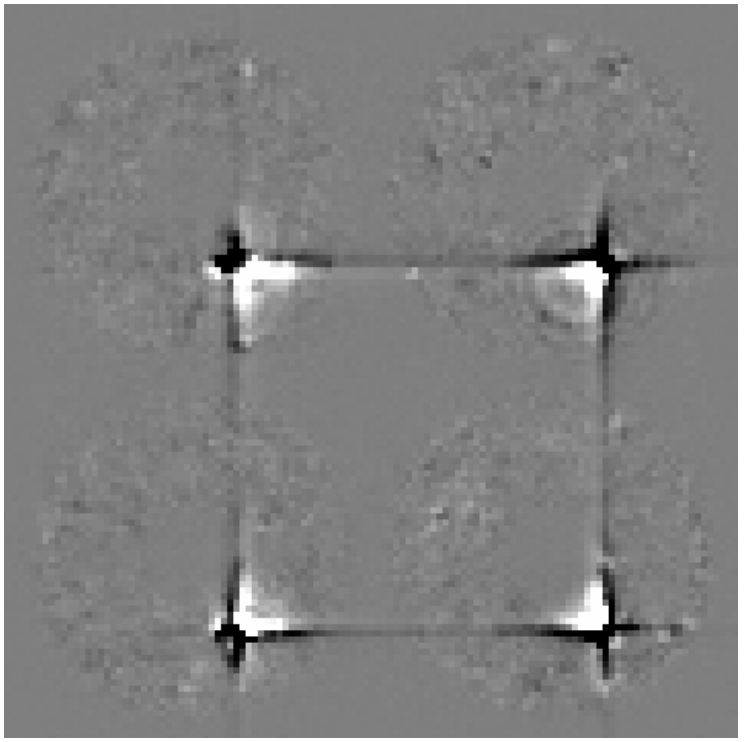
RM assembled from single poke sequence:
+- +- +- +-

RM assembled from average of two poke sequences:
+- +- +- +-
+- -+ +- -+

RMs reconstructed from Hadamard pokes, 2kHz modulation (DM moves during EMCCD frame transfer)

Multi-channel DM virtualization & timing knowledge/stability
→ on-sky response matrix acquision, while ExAO loop running





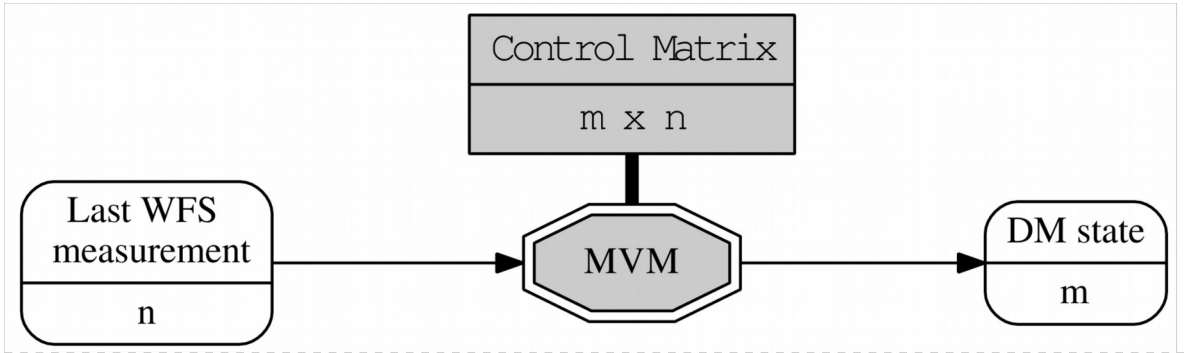
Left: WFS reference
Right: Response to single actuator poke (one of 2000)

RM measurement @ 2kHz takes 4000 pokes = 2 sec
Multiple RMs averaged to increase SNR

Self-learning AO control

Conventional AO:

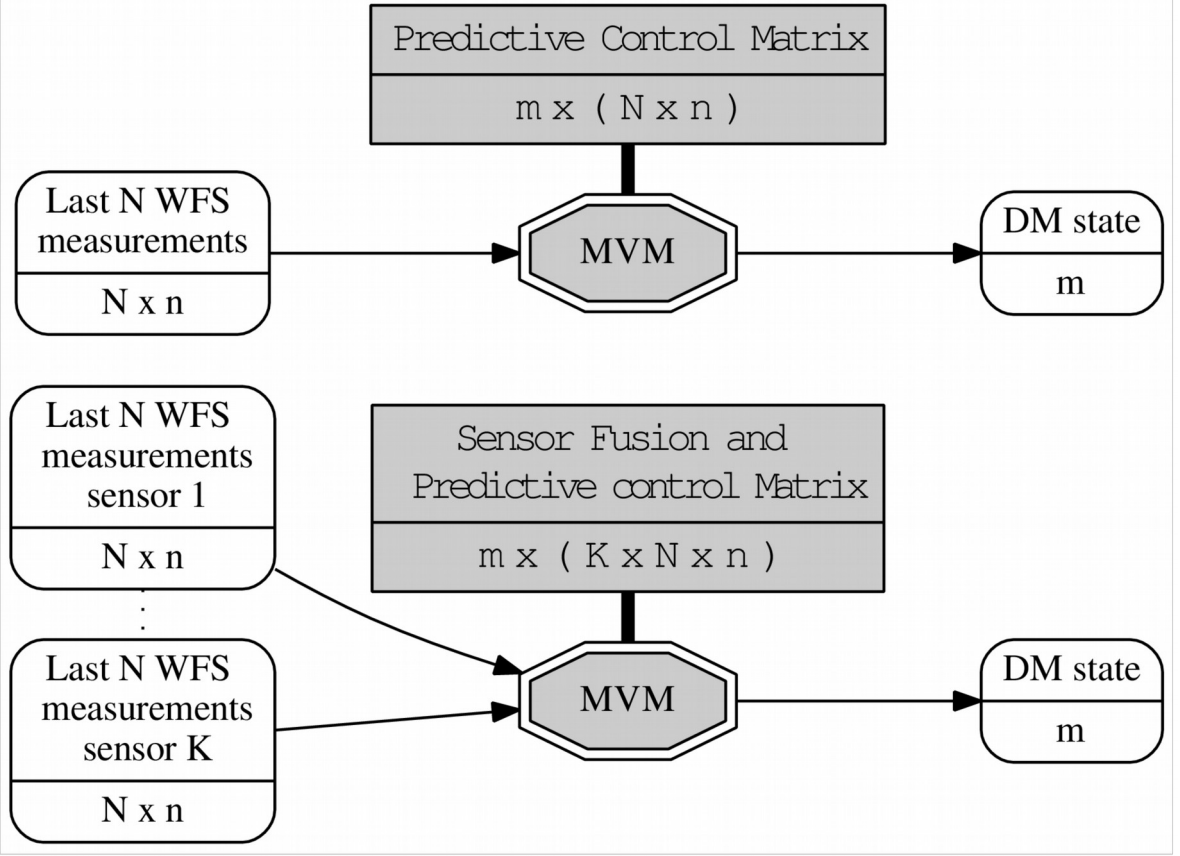
Resp Matrix is **measured**
CM computed as pseudo-inverse of RM



Self-learning AO control:

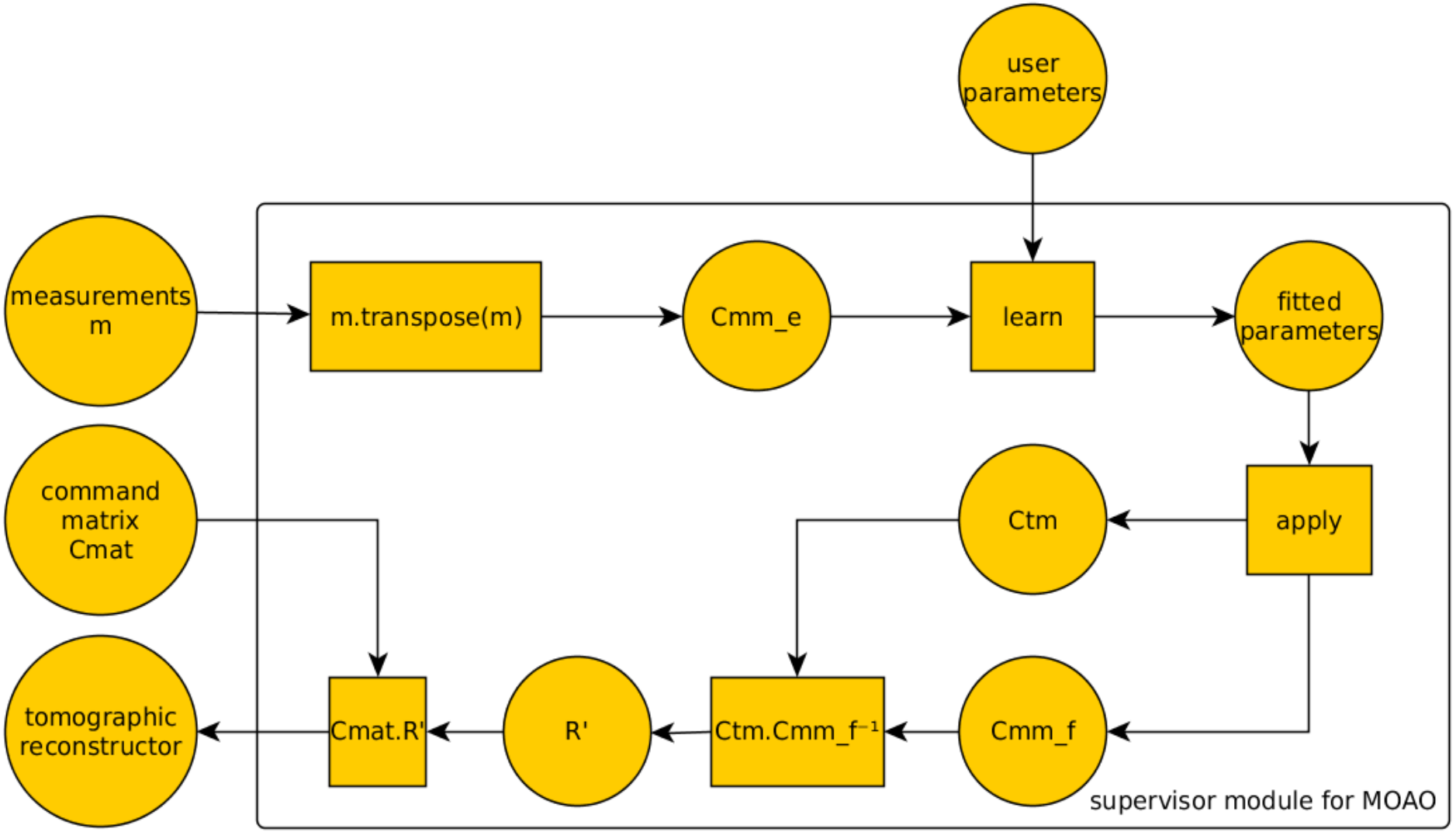
Optimally use recent (predictive control) and auxiliary (sensor fusion) measurements → control matrix is very big, and usually impossible to measure

CM is **derived** from WFS(s) telemetry using machine learning approaches



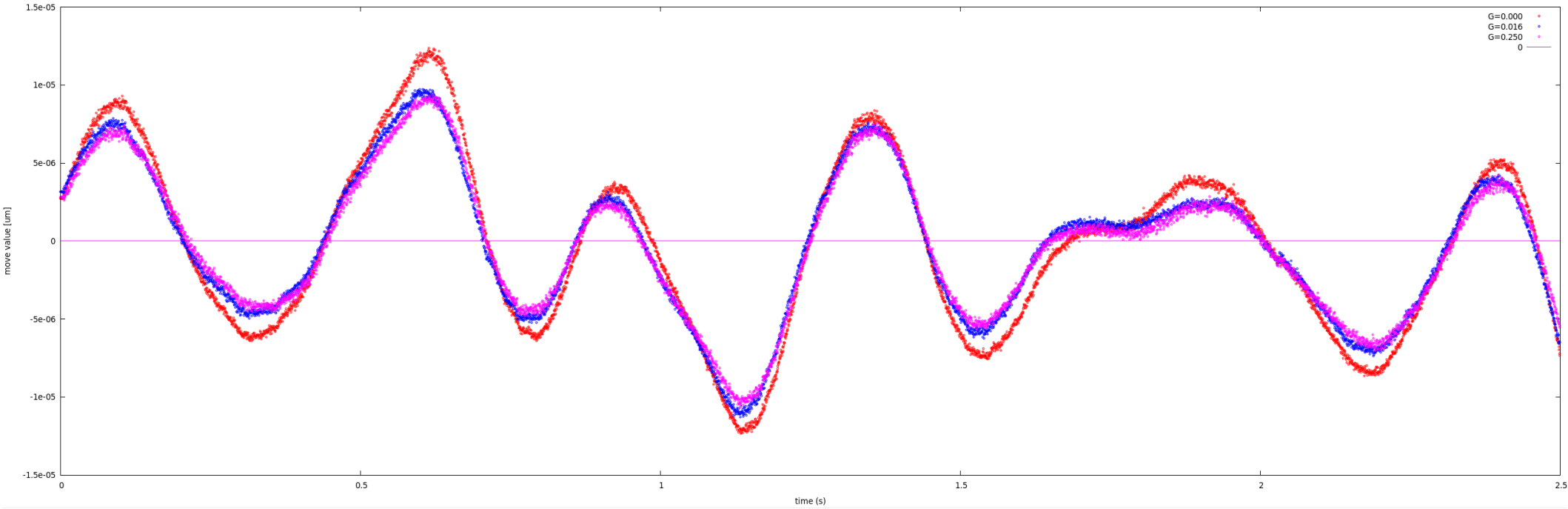
Loop supervision module

Mix of cost function optimization for parameters identification (“Learn” process) and linear algebra for reconstructor matrix computation (“apply” process)



Open loop reconstruction
Comparison between gain values

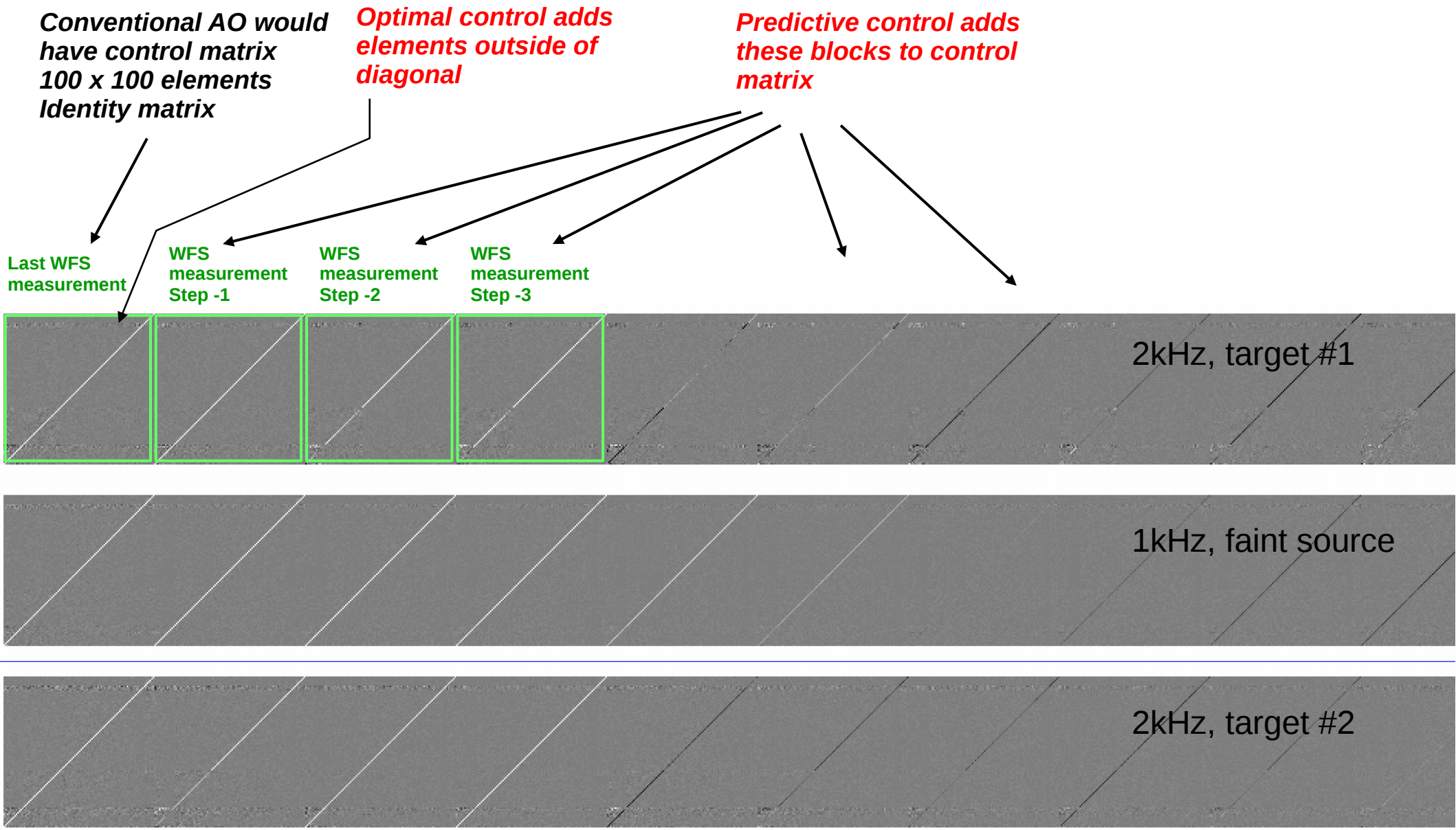
G=0.000 → over-estimates OL values
All G>0.0 reconstructions match at %-level



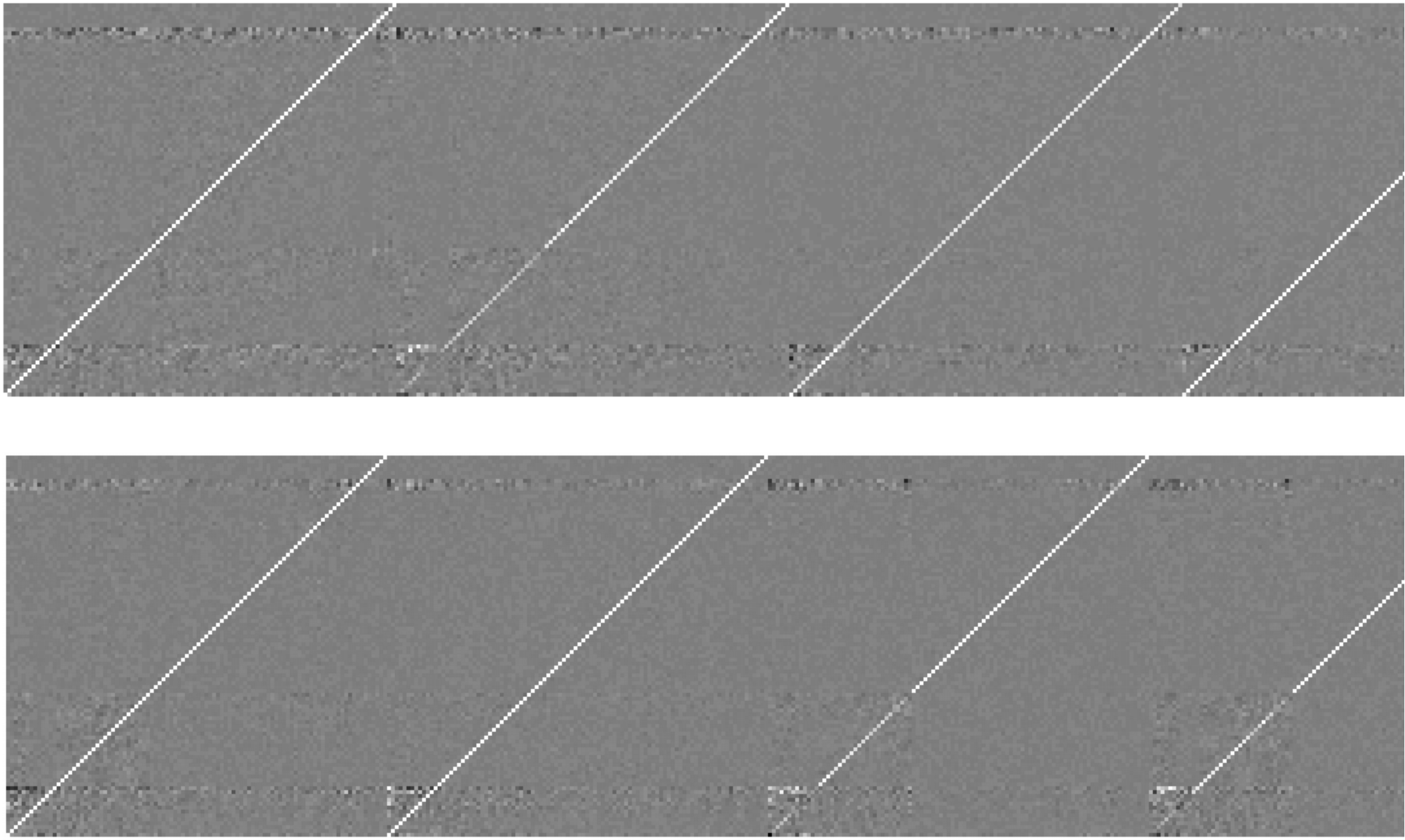
G=0.000 test relies entirely on WF residuals for OL estimation
G>0.000 tests rely mostly on DM values for OL estimation

Test shown here uses full speed RM acquisition which underestimates RM by ~15% due to DM time-of-motion → reconstructed WFs from WFS are over-estimated by ~15%

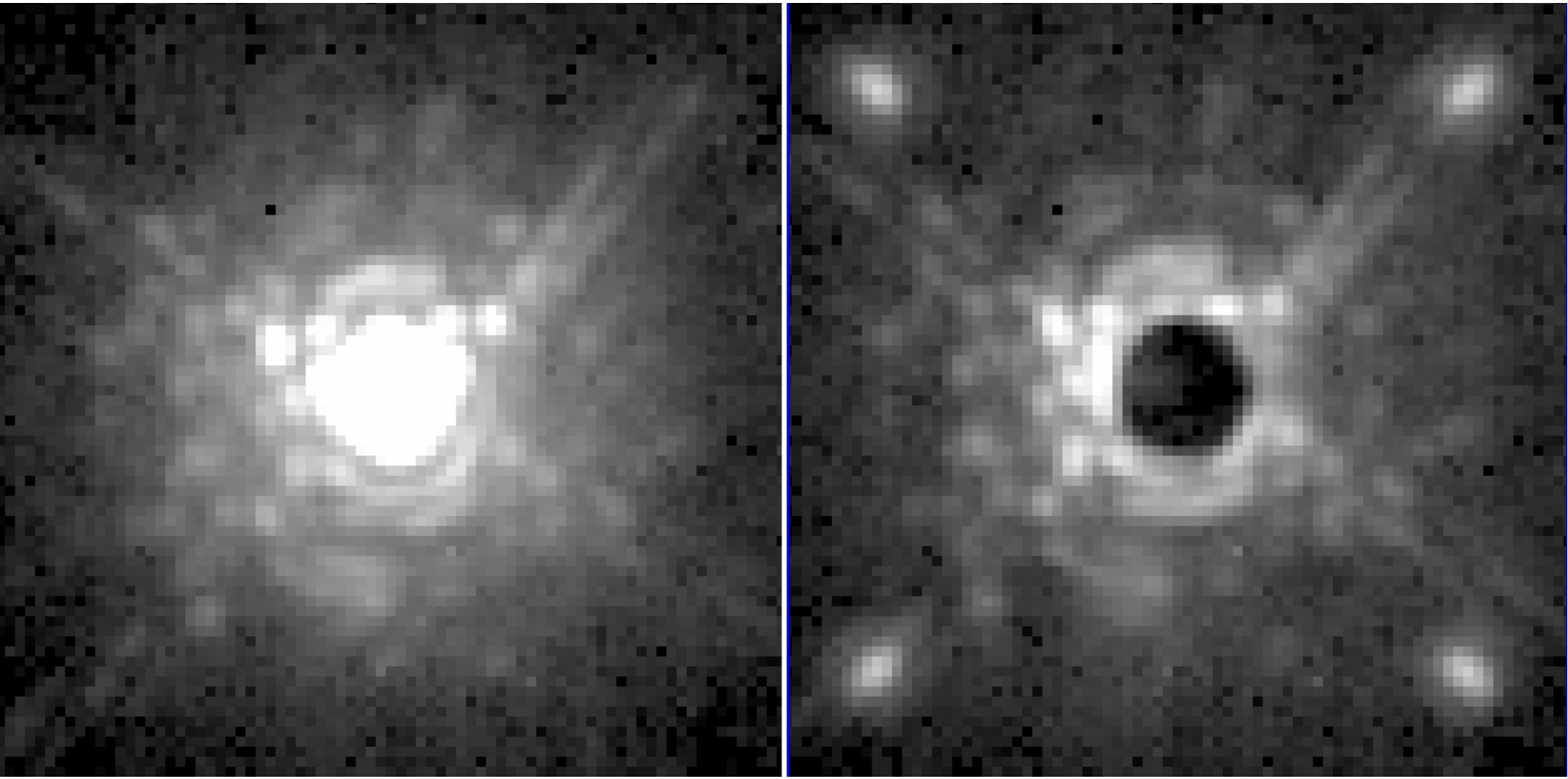
On-sky predictive control matrix (modal representation, 100 modes shown)



Prediction control matrix



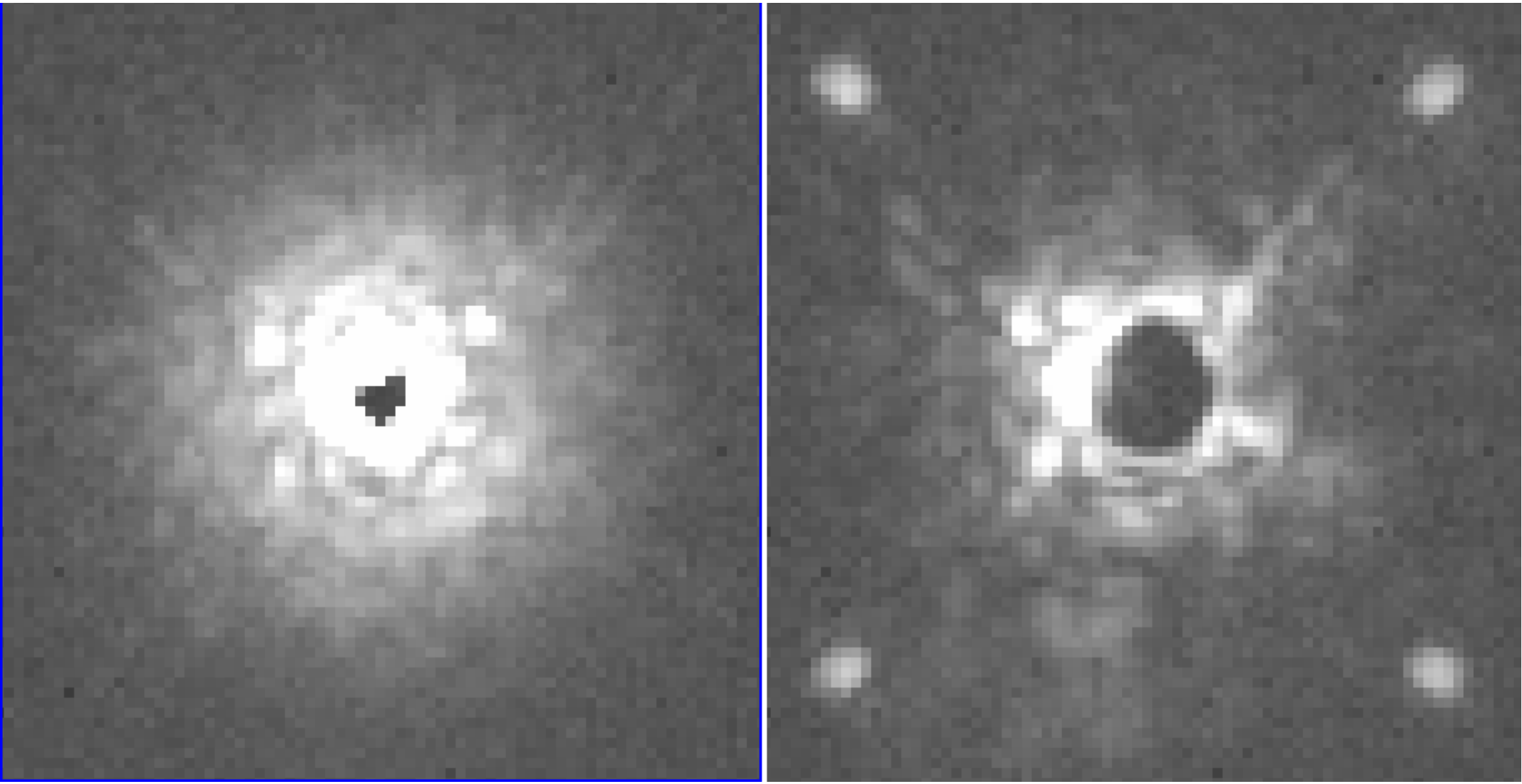
On-sky results (2 kHz, 50 sec update)



OFF (integrator, gain=0.2) ***ON***

Average of 54 consecutives 0.5s images (26 sec exposure), 3 mn apart
Same star, same exposure time, same intensity scale

Standard deviation improved by 2.5x

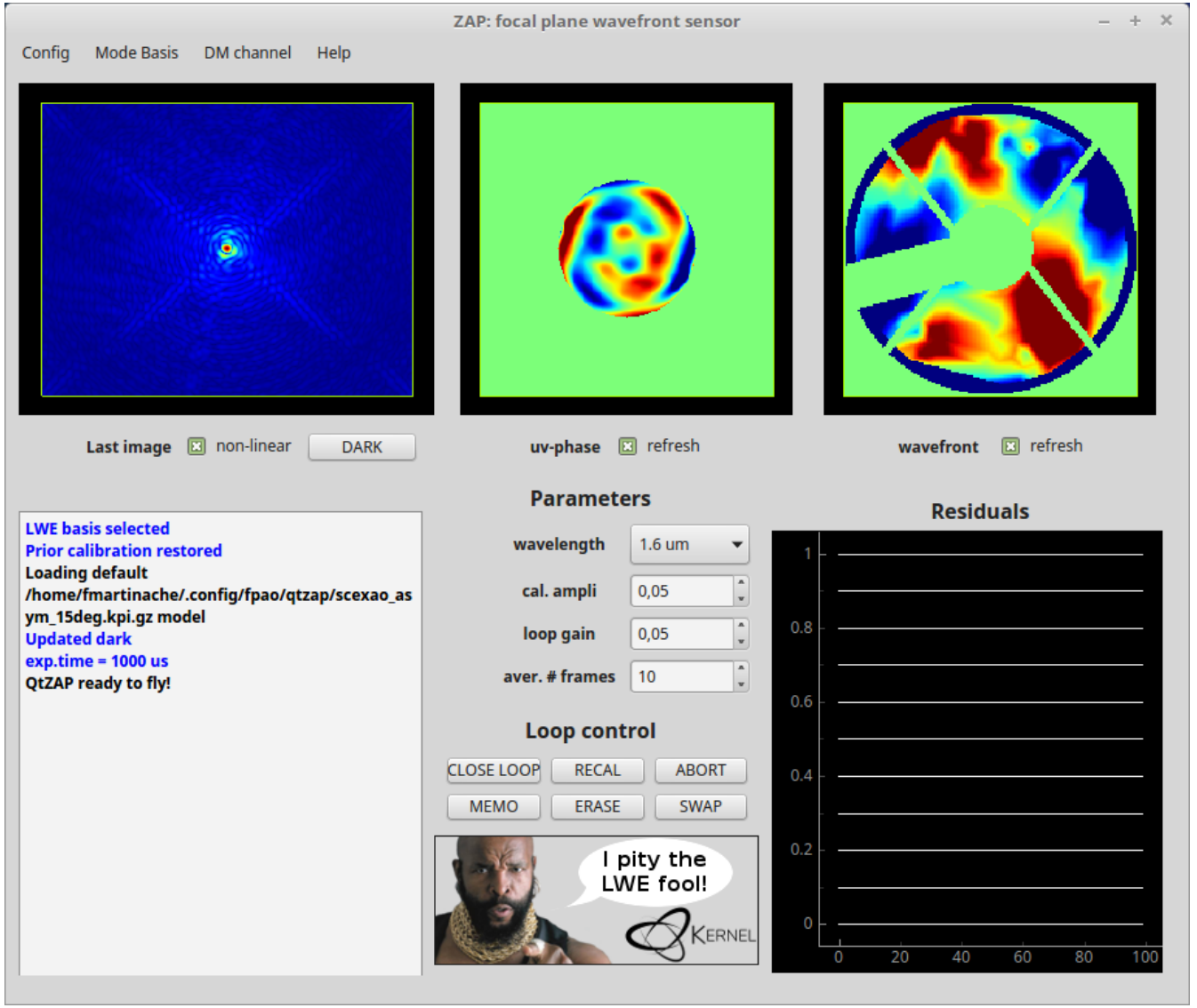


OFF (integrator, gain=0.2) ***ON***

Standard deviation of 54 consecutives 0.5s images (26 sec exposure), 3 mn apart
Same star, same exposure time, same intensity scale

Focal Plane WFS/C

OCA/KERNEL – developed software



- Address NCPA

- Asymmetric mask (pupil)

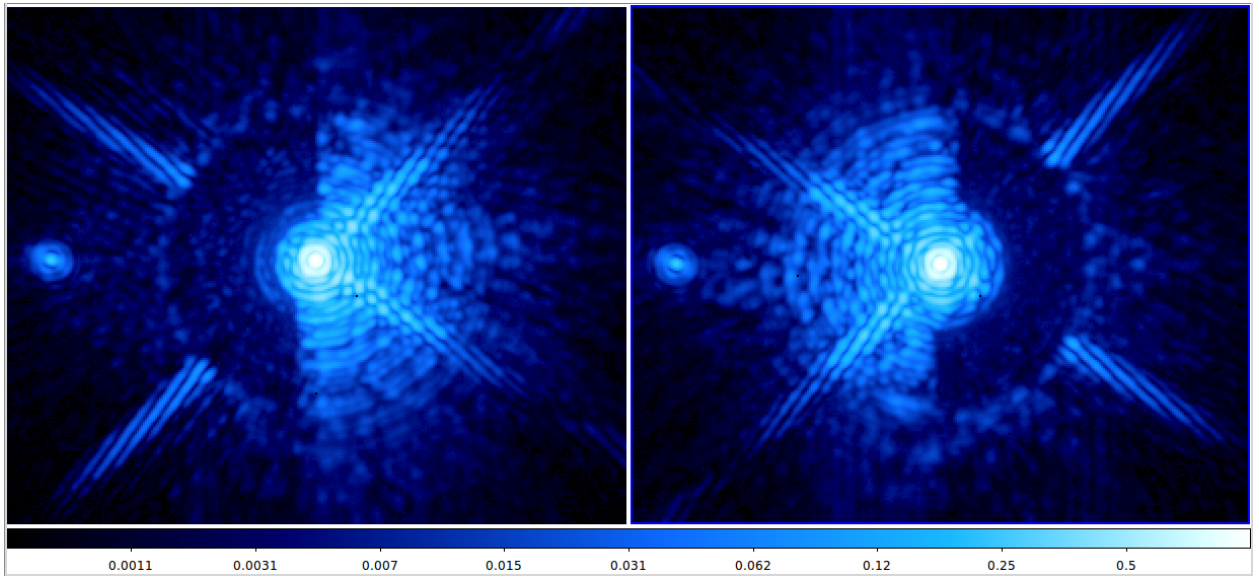
- On-sky closed-loop control

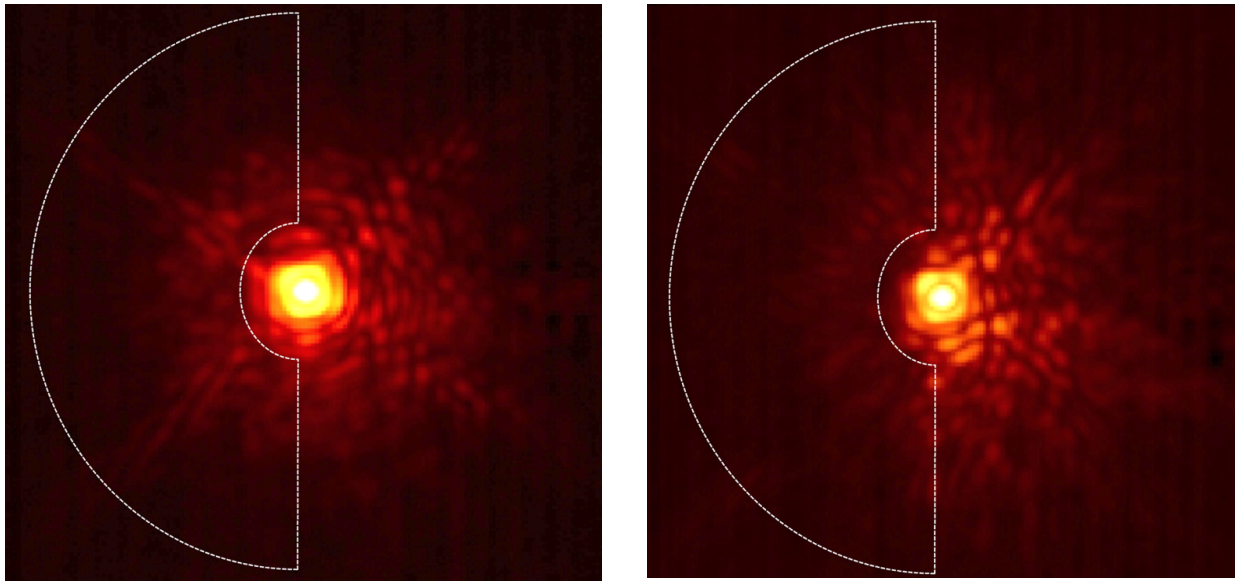
- Focal plane based WFS Low-order (Zernike and LWE) modes.

- mode compatible with coronagraphy in development

KERNEL

Speckle Control





Speckle nulling, in the lab and on-sky (no XAO).

Experience limited by detector readout noise and speed.

KERNEL project: C-RED-ONE camera.

From:
- 114 e- RON
- 170 Hz frame rate

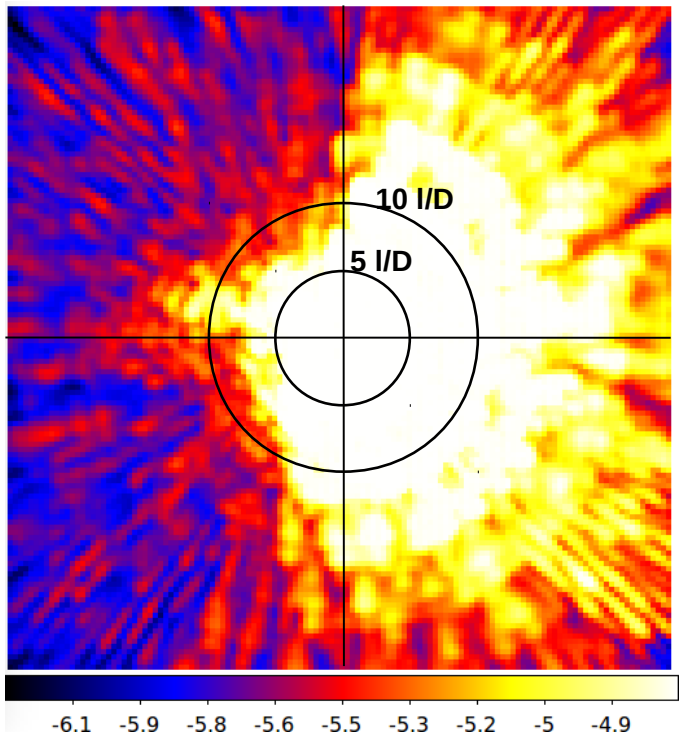
To:
- 0.8 e- RON
- 3500 Hz frame rate

Expect some updates

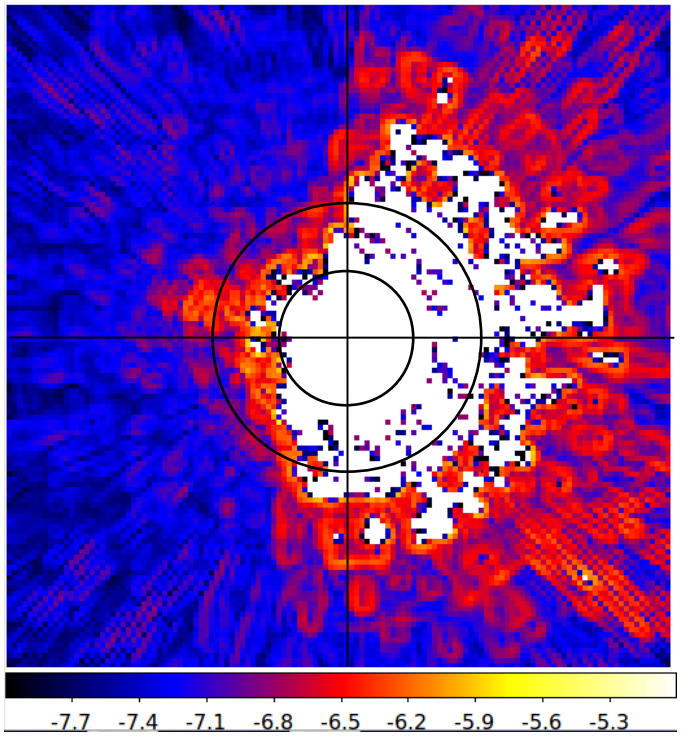
Observatoire de la CÔTE d'AZUR

Conventional Lyot Coronagraph, Broadband light: 0.9-1.7 um (62% wide band)

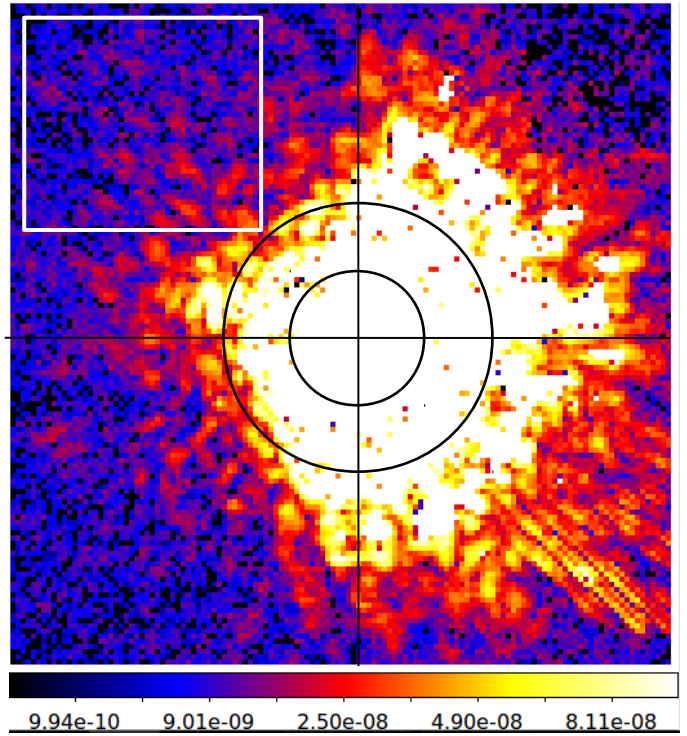
Raw Contrast



Open-loop contrast stability



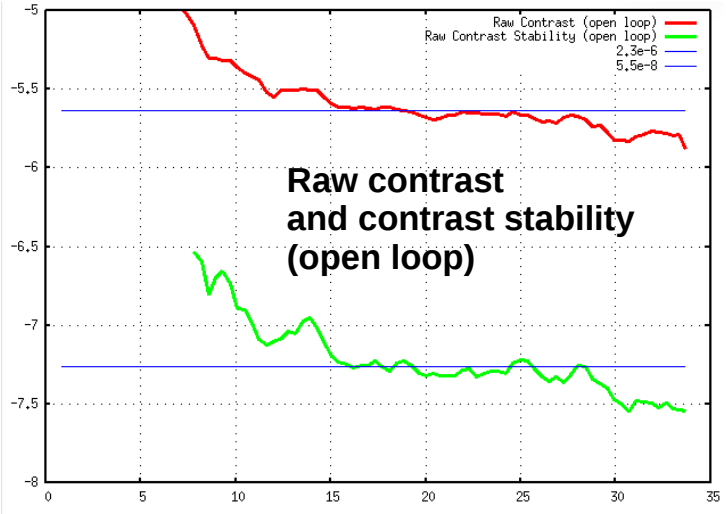
Contrast stability over 0.6 sec

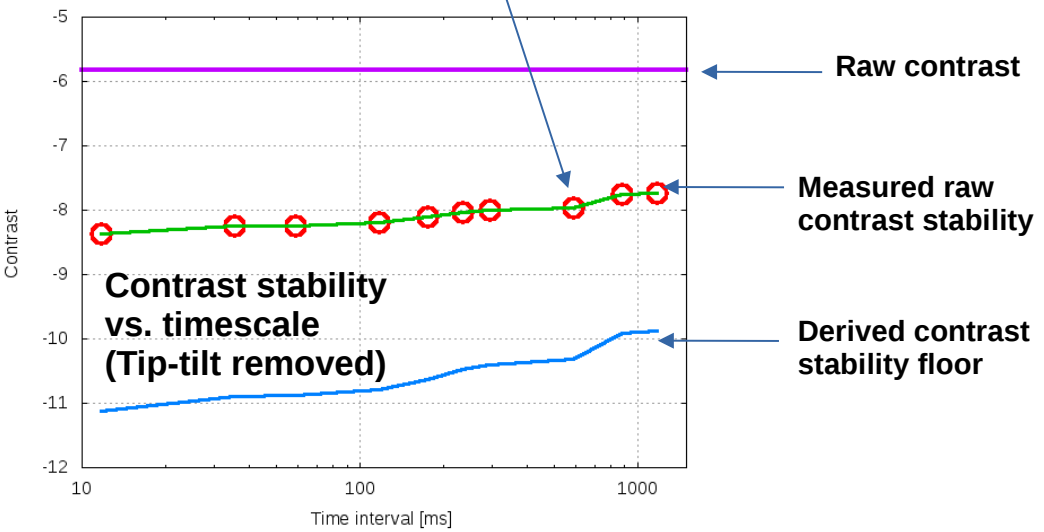


Average raw contrast [15-20 l/D] = 2.3e-6

Average raw contrast stability [15-20 l/D] = 5.5e-8

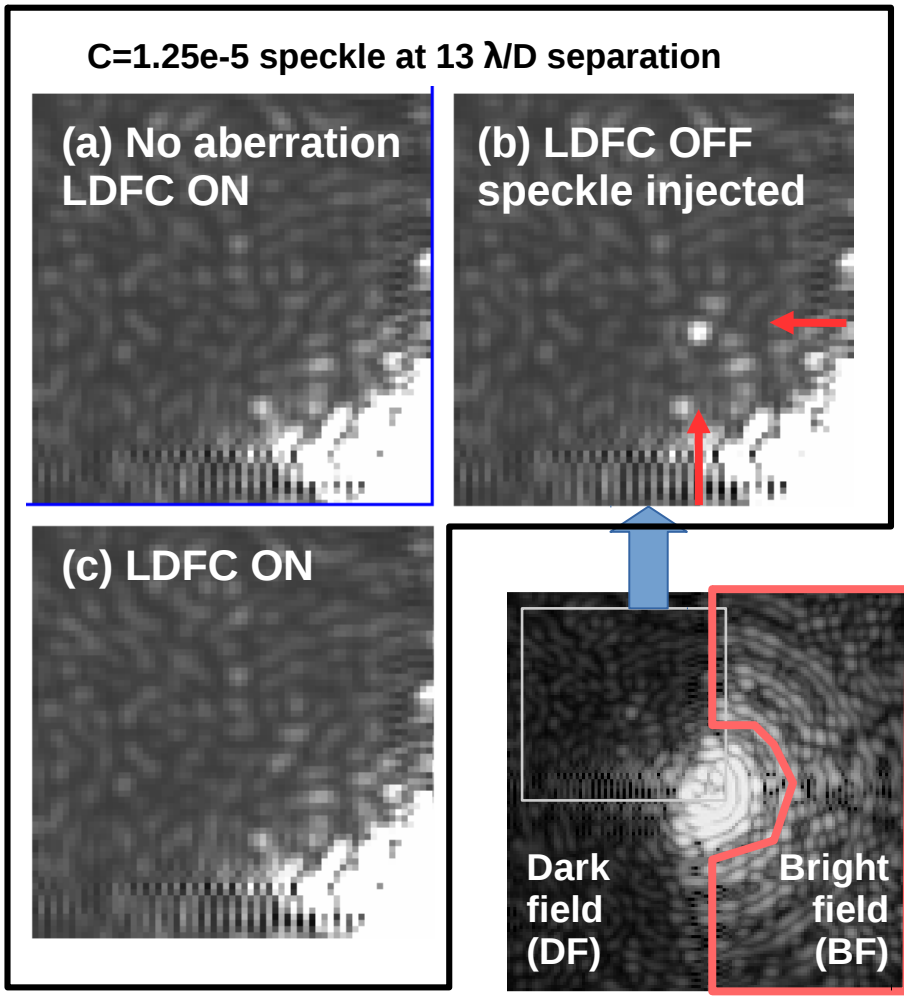
Average raw contrast stability [11-34 l/D] = 1.1e-8 (averaged value within white box)

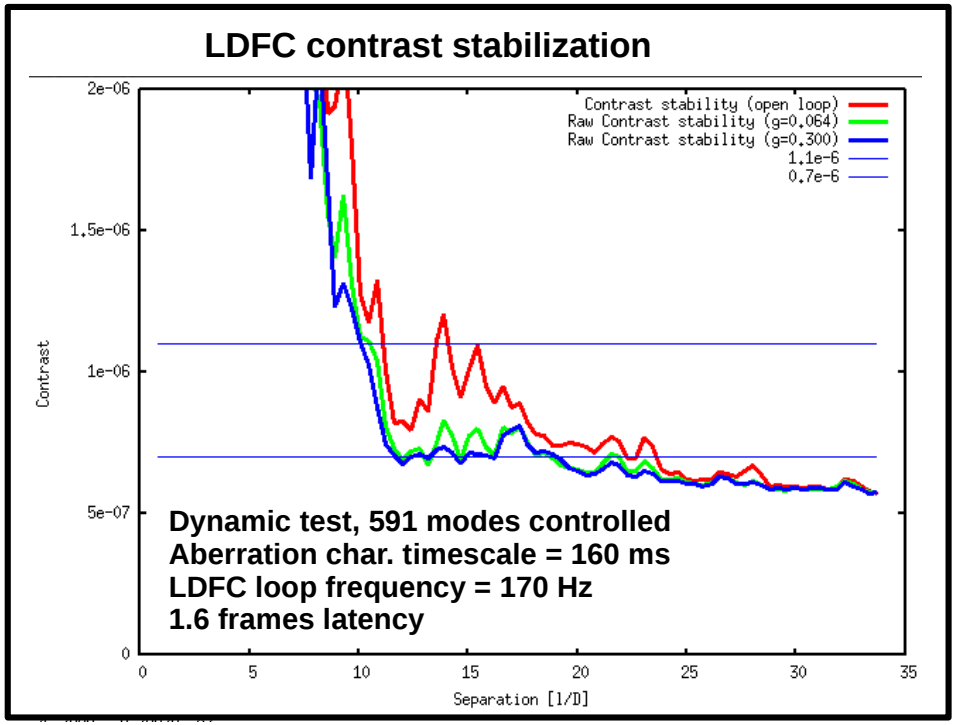


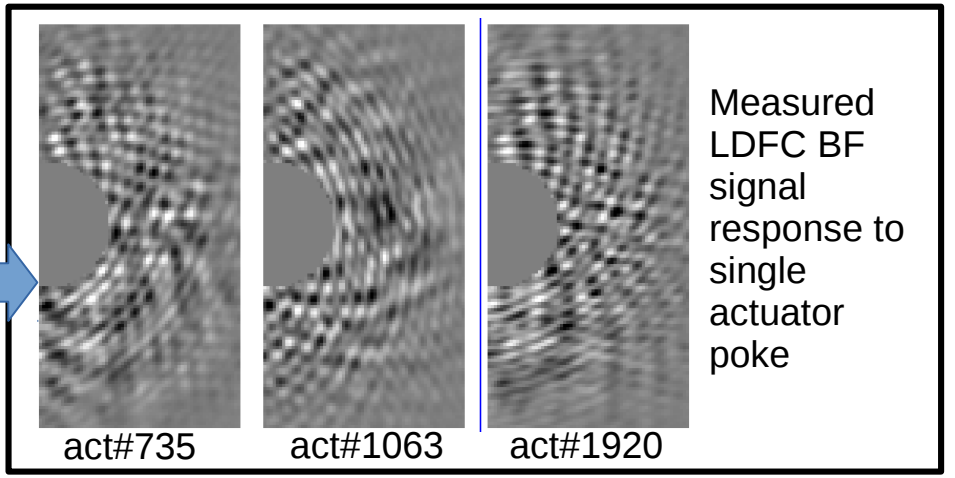


Linear Dark Field Control (internal source)

Near-IR spatial LDFC validation @ SCExAO
Frame rate = 170 Hz, Lyot coronagraph in near-IR (1.55um, 50nm wide band)

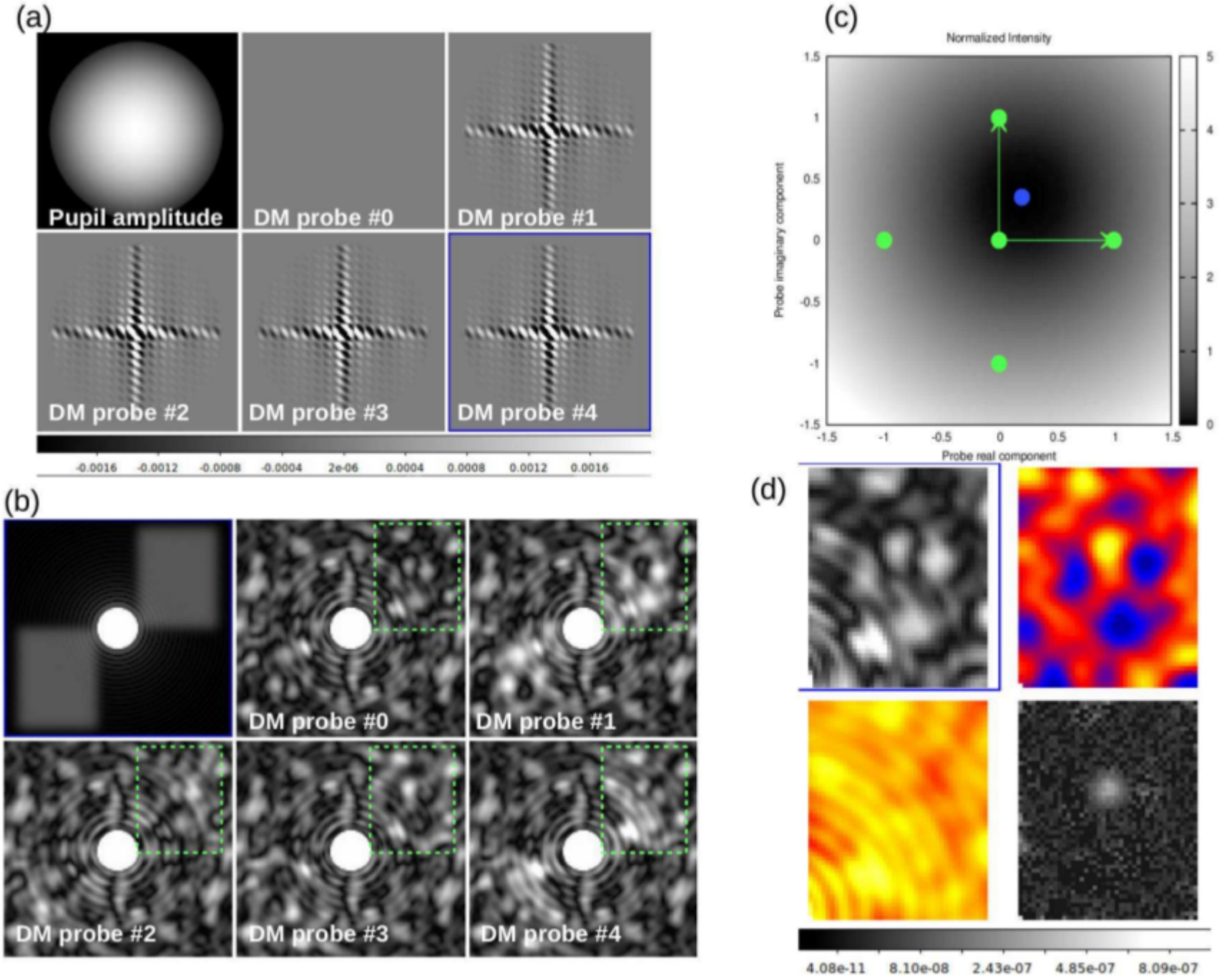






Measured LDFC BF signal response to single actuator poke

Coherent Speckle Differential Imaging



User Interface and programing / scripting

Top-level ASCII-based menus launch cacao programs in tmux sessions

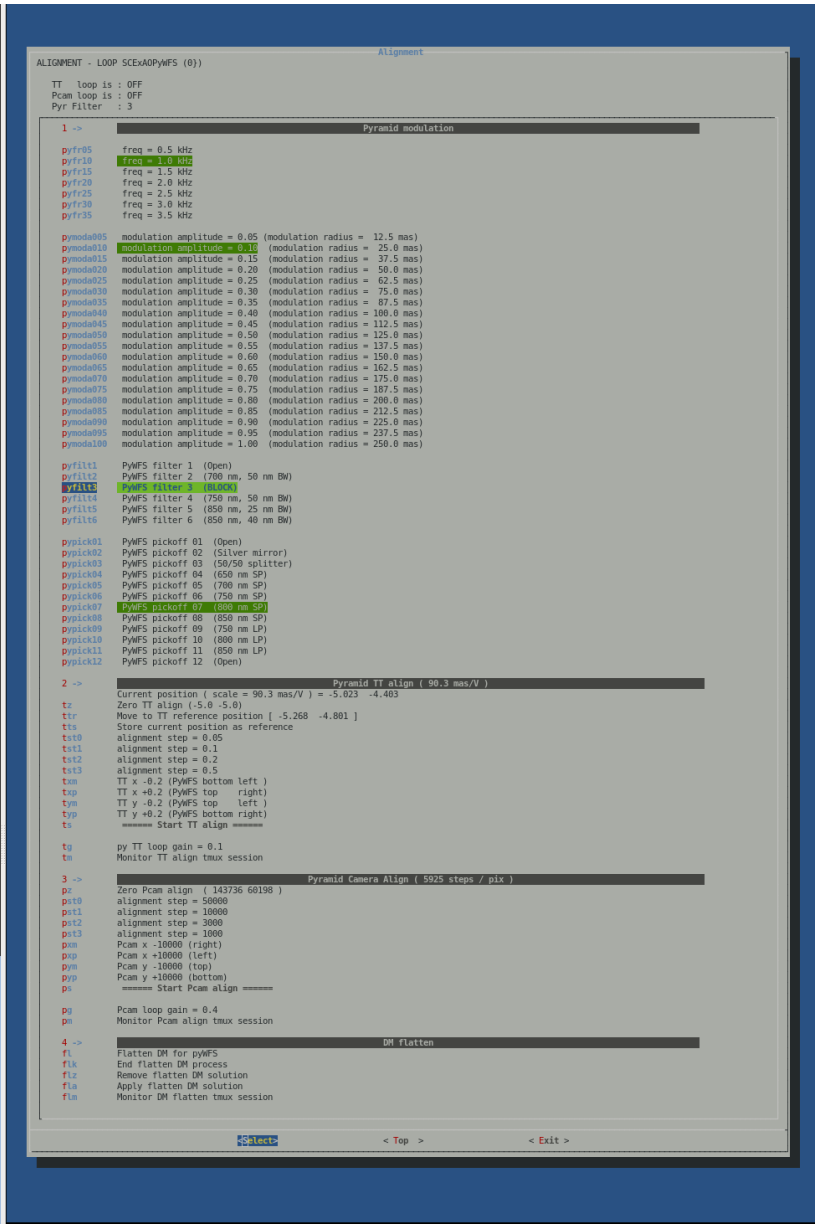
bash scripts assist with:
- sequencing between tasks
- hardware resources allocation (CPU cores, GPUs, memory)

To be upgraded by more capable KRAKENS python-based tool (currently in development)

Users can code new processes in C or Python

Developing well-documented examples users can modify

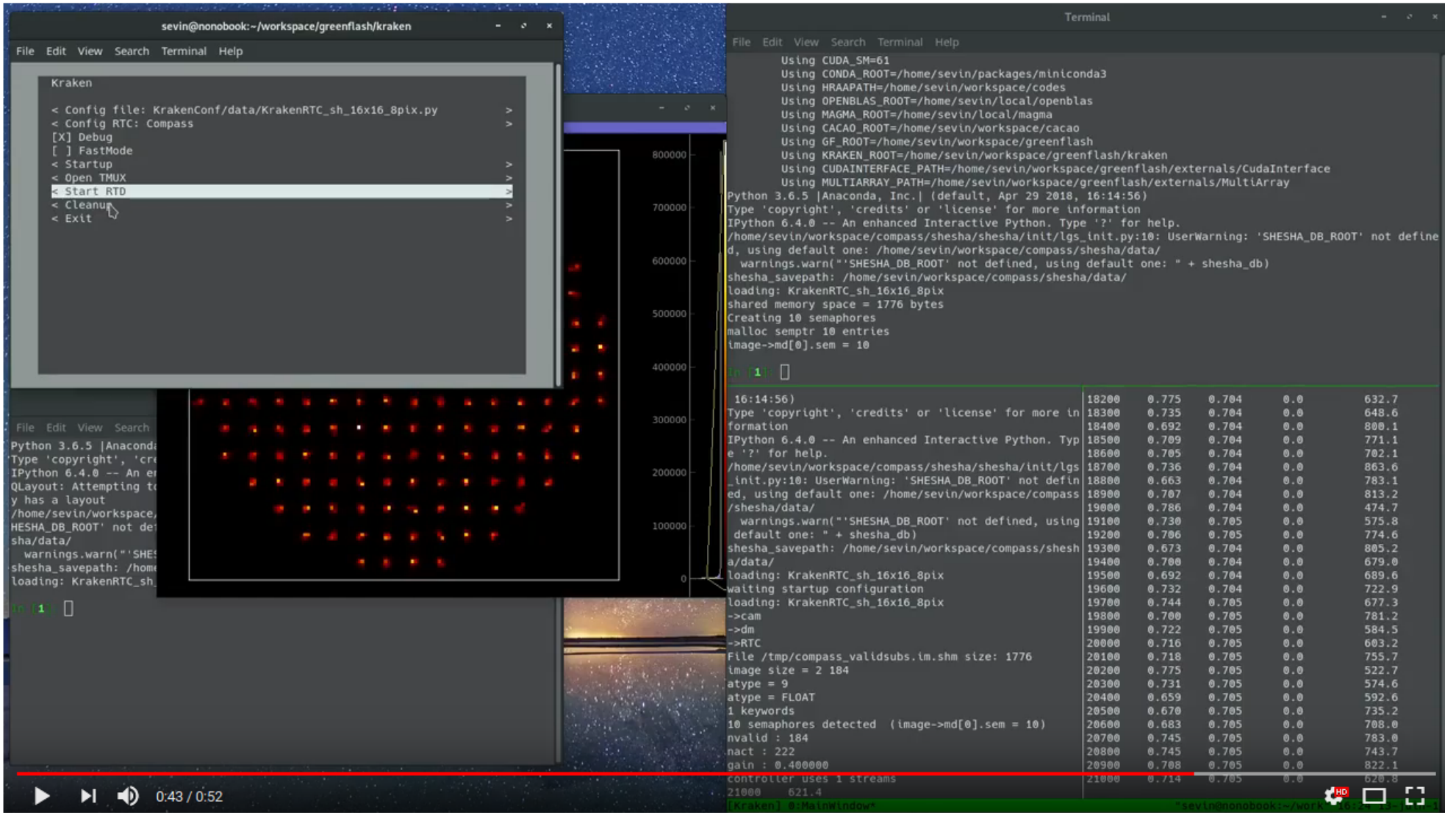
Example control GUI (bash scripts)



Software ecosystem integration & High performance hardware abstraction

l'Observatoire de Paris | LESIA

Laboratoire d'Études Spatiales et d'Instrumentation en Astrophysique

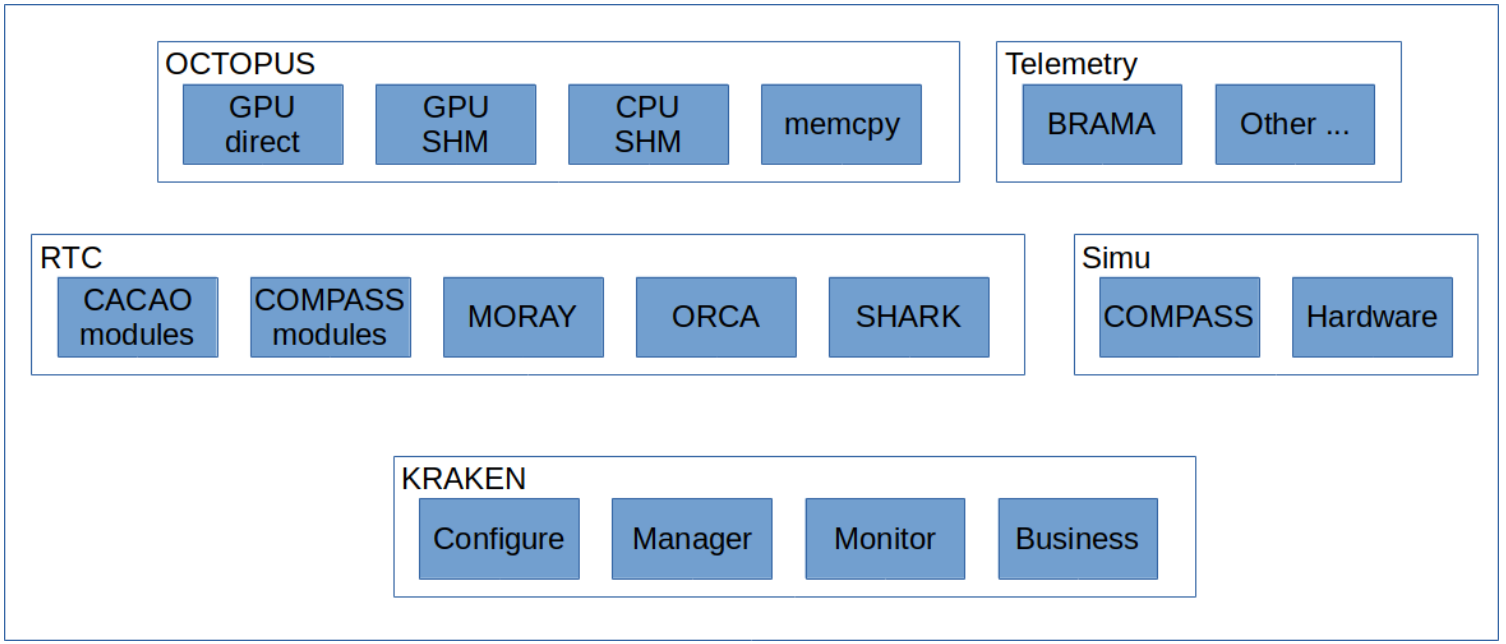


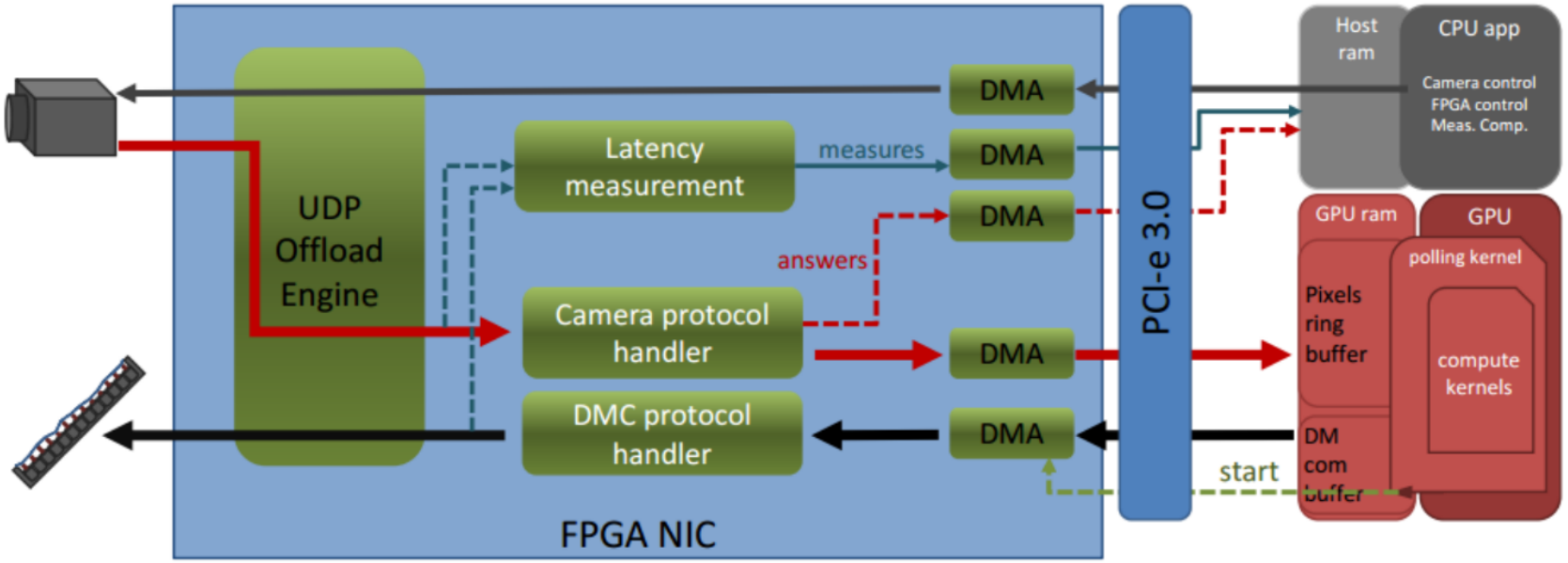
https://youtu.be/BMHeGeKfEs0

Software ecosystem integration & High performance hardware abstraction

l'Observatoire de Paris | LESIA

Laboratoire d'Études Spatiales et d'Instrumentation en Astrophysique





Thank you !

https://github.com/cacao-org/cacao