# Data Acquisition with PSI/Eiger at the ESRF

Alejandro Homs Puron

Beamline Control Unit - Software Group

Instrumentation Services and Development Division

ESRF

Paul-Antoine Douissard, Menhart Kocsis,

Roberto Homs Regojo, Fernando Calvelo, Pablo Fajardo

## Talk outline

- Introduction
  - The ESRF BL control
  - PSI/Eiger control considerations
- Lima integration
  - PSI/Eiger-500k
  - PSI/Eiger-2M
- High frame rate acquisitions
  - Performance issues
  - Buffers & latency: CPU Affinity control

# The ESRF BL context

- Exotic experiments now run in a routine basis
  - Synchronisation with multiple devices
- More efficient X-ray source, optics, mechanics, instrumentation electronics & detectors
  - More photons / second
  - Faster experiments
- Network connectivity
  - Fiber optic links to the detectors
  - Fast data transfer links to central storage
- Expensive beamtime
  - Reliability

# The ESRF BL control system

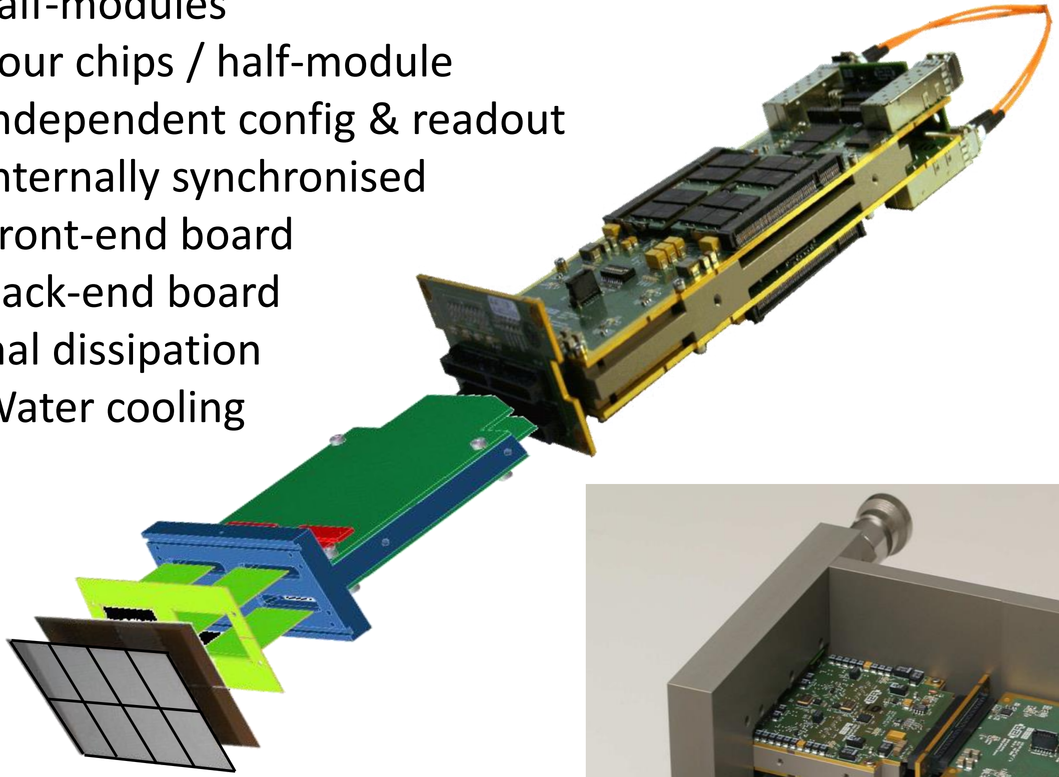
Requirements:

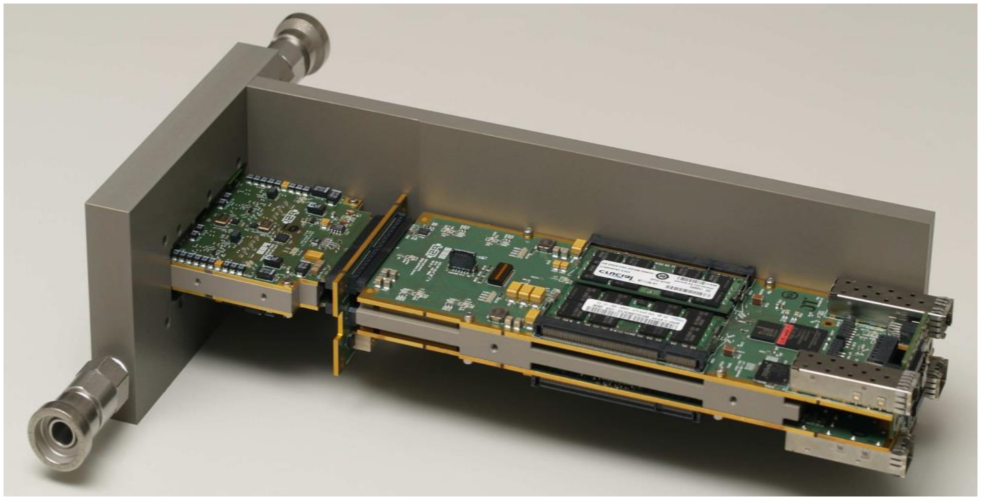
- Be reachable from SPEC / BLISS
    - Experiment “orchestrator”
- Take simple frames:
    - Snapshots, step-by-step scans
- Acquire a sequence of frames:
    - Fast/continuous scans
    - As fast as possible!
- Get basic online data reduction
- Save data in storage servers

EXPERIMENTAL CABIN
CONTROL CABIN

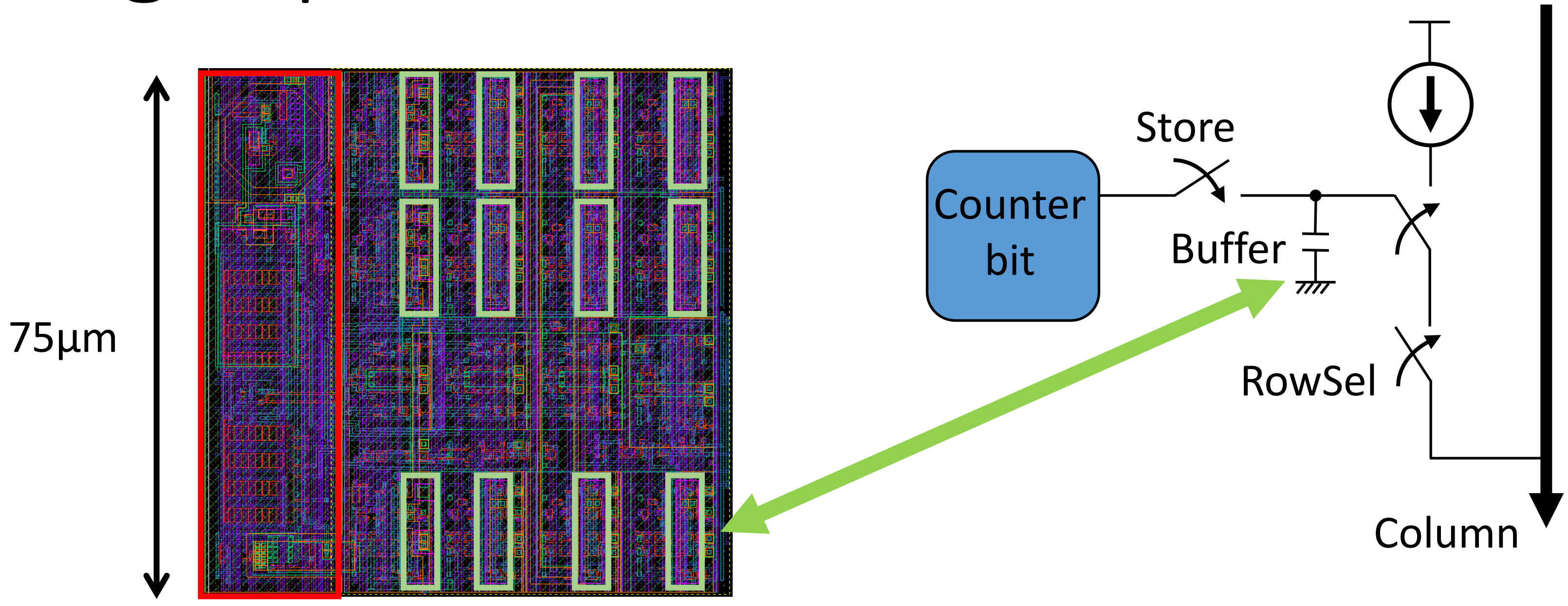
# PSI/Eiger: design

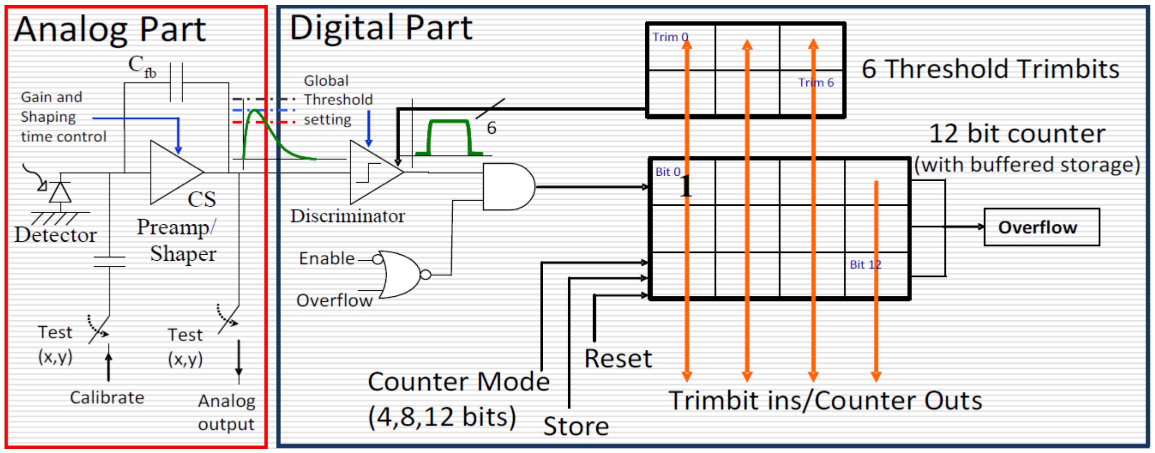
- Two half-modules
  - Four chips / half-module
  - Independent config & readout
  - Internally synchronised
  - Front-end board
  - Back-end board
- Thermal dissipation
  - Water cooling





# Eiger pixel electronics

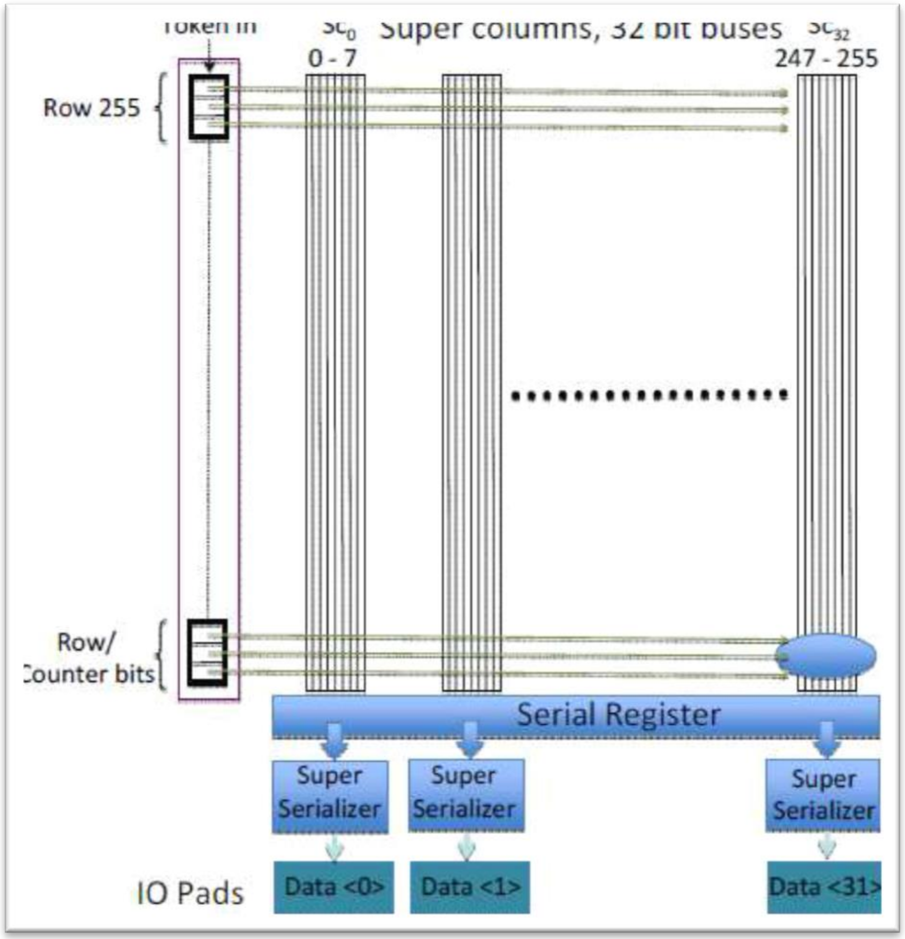




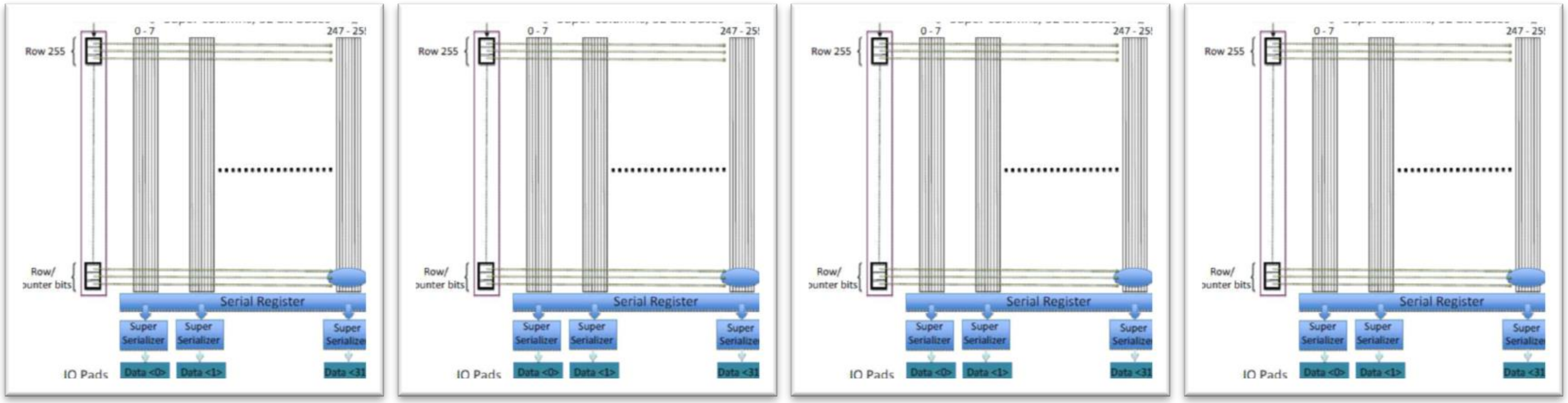
- Pixel level buffer
- Parallel readout & exposure
- Counter to buffer:
  - 3.4 µs
  - 1000x shorter than ESRF FReLon

# Eiger chip readout

- 1 super-column = 8 columns
  - 8 col * 4 bit = 32 bit/super-column
  - 32 super-columns
- Three readout speeds:
  - Normal, Half, Quarter
- Two readout modes:
  - Parallel, Non-Parallel
- Parallel + Full-Speed:
  - 4-bit: 40 µs ⇨ 22 kHz
  - 8-bit: 82 µs ⇨ 11 kHz
  - 12-bit: 164 µs ⇨ 6 kHz
    - 12-bit is called "16-bit"
  - Data rate:
    - 2.7 MByte/s/super-column
    - ~700 MByte/s/chip
- 32-bit: "auto-sum" mode
  - Hardware frame accumulation

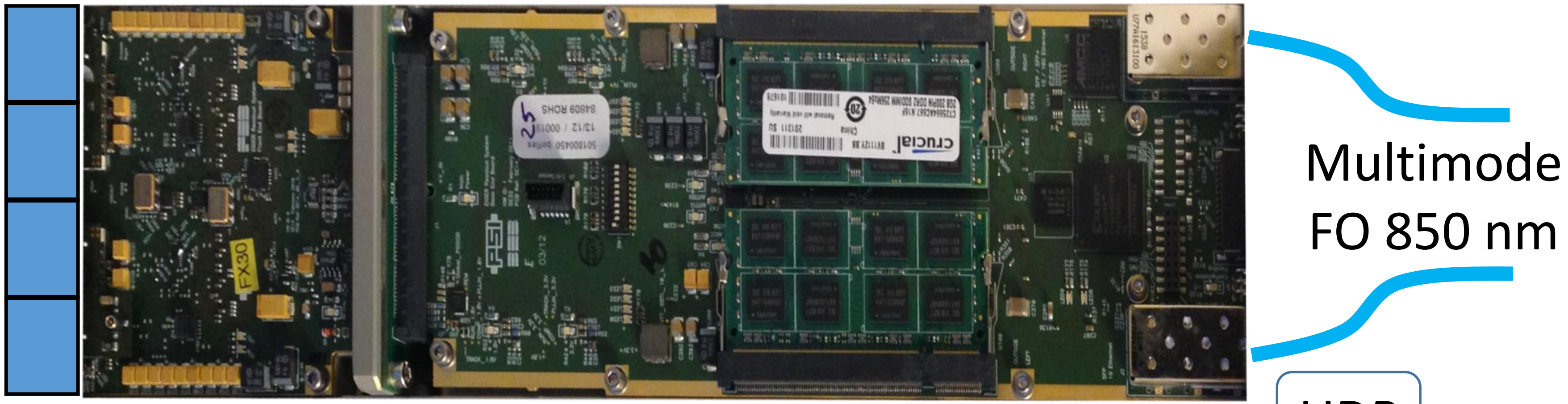


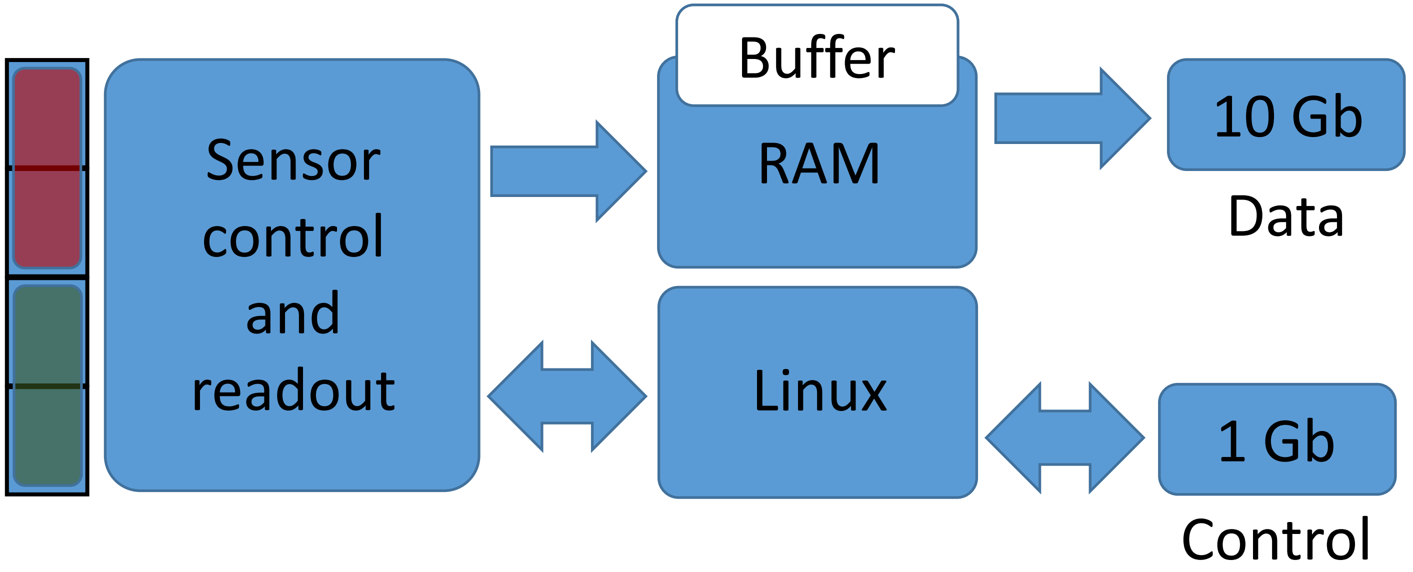
# Eiger: Half-module readout



- Parallel chip readout
- 2.7 GByte/s/half-module!
- 5.4 GByte/s/module!!

# PSI/Eiger: data acquisition





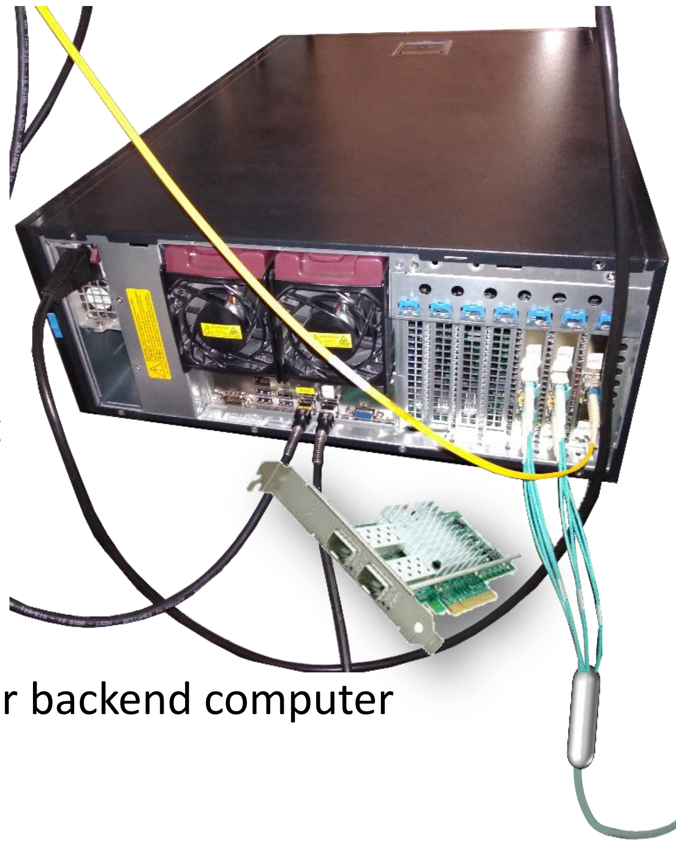
# Integration: hardware

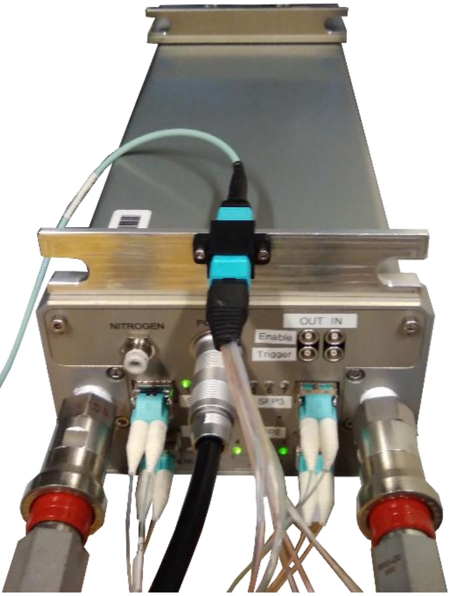
Standard Ethernet technology:

- No specific DAQ board
- State-of-the-art industry products
  - Ring buffers

Linux operating system:

- Network connectivity
- Low-level data management in the kernel
  - Kernel buffers





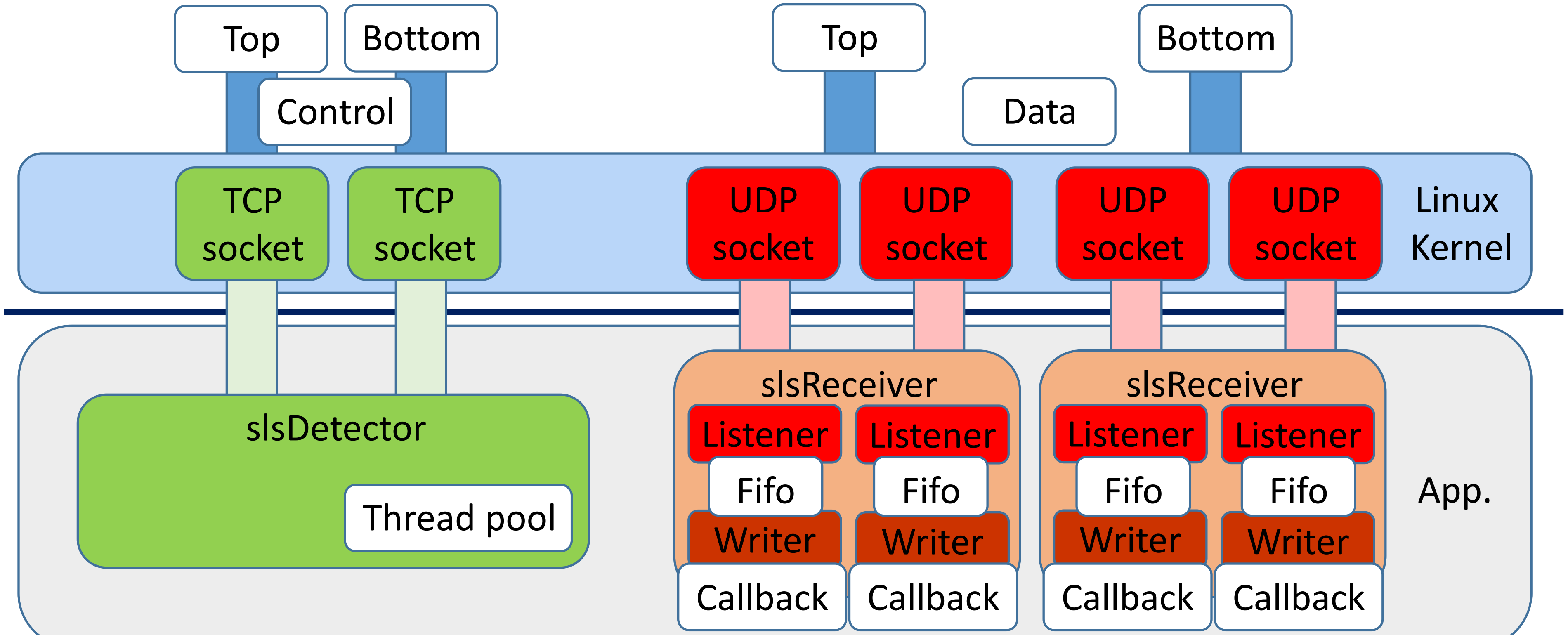
TID support:

- Server-performance detector backend computer
- Linux operating system
- Fiber optics connectivity
  - 12-fiber aggregated cables & MTP technology
  - Network adapters & transceivers
  - Cabling & patch panels
- Fast link to central storage servers

# Integration: SDK

***slsDetectorPackage***:

- PSI/SLS Detector Group
- Control and data acquisition library
  - Eiger, Jungfrau, …
- Written in C++



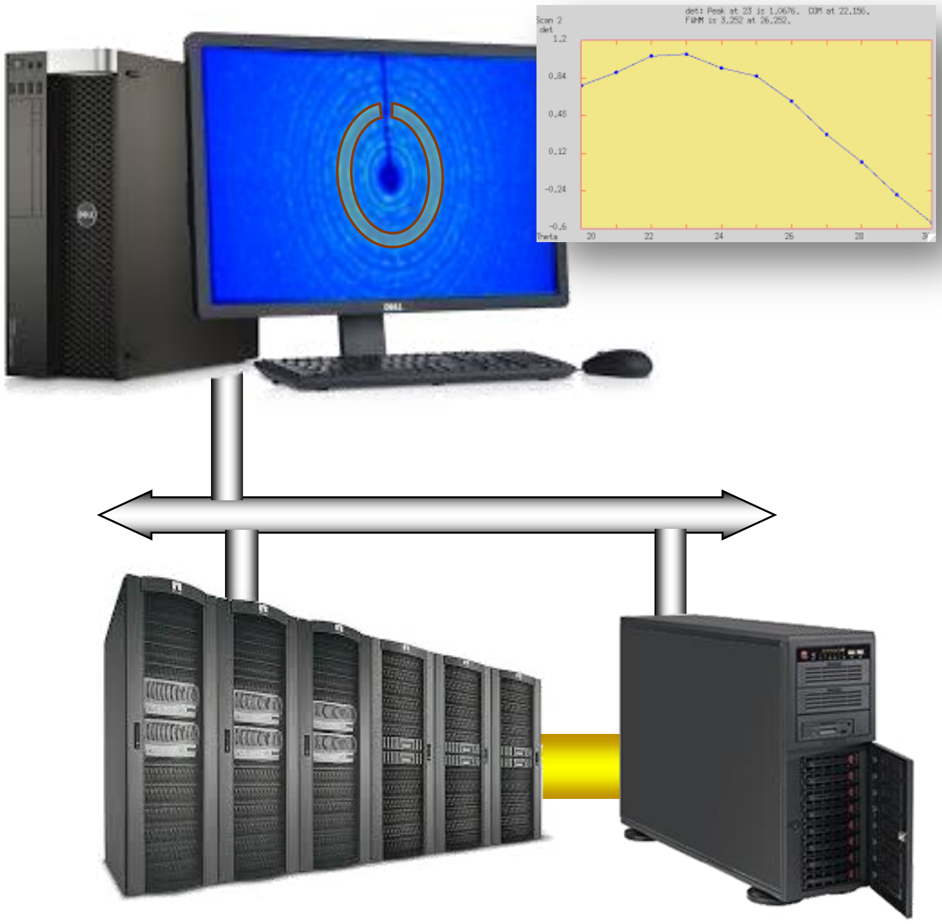
# BL control integration: Lima

***Library for IMage Acquisition***

- Developed at the ESRF
- Virtually in every beamline
- Hardware abstraction for 2D detectors
    - Control of generic parameters
    - Image generator
- Provide common image processing
    - Frame reconstruction
    - Online visualization
    - Online data-reduction
        - RoI counters (rect, arc)
- Tango middleware interface
- Control from Spec / Bliss
- Saving locally and on remote servers

Features:

- Optimum usage of computer resources
- Multi-threaded processing



# Lima plugin

Detector-specific configuration:

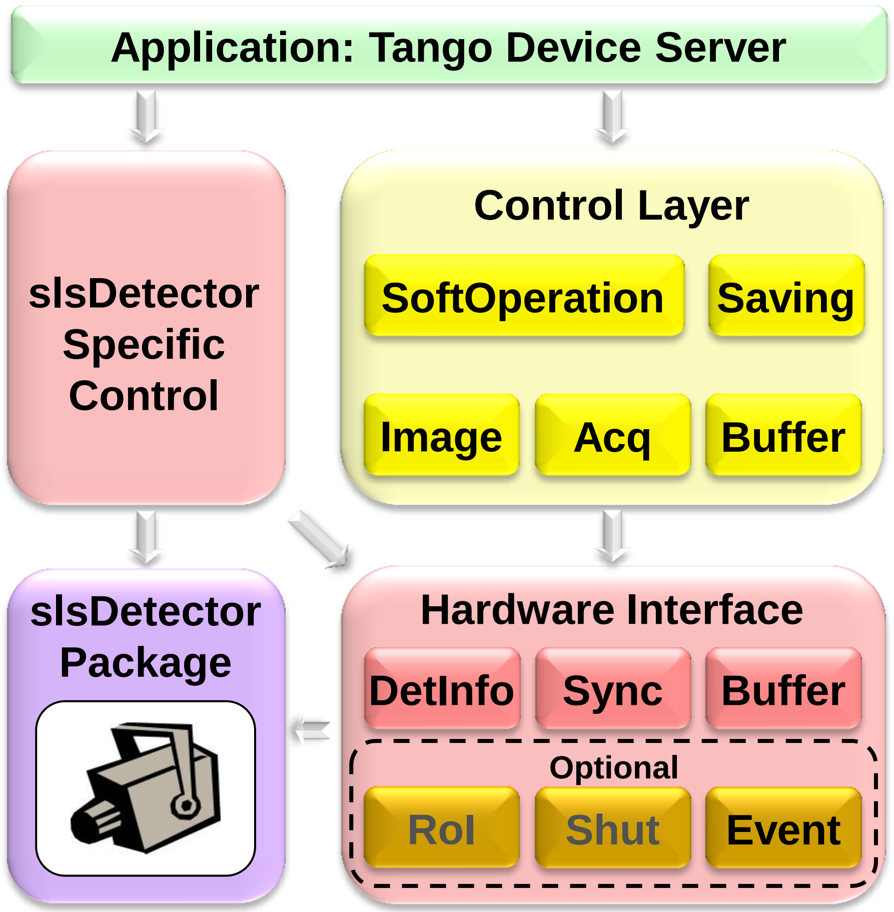
- Sensor parameters
  - Energy (threshold DACs & trim-bits)
  - Pixel-depth & read-out modes
- Monitoring (temperatures)

Synchronisation configuration:

- Scan parameters
  - Exposure time & frame rate / Ext-sync.
  - Number of frames

Data acquisition:

- Acquisition control:
  - Start, stop & status (progress)
- Inject frames into Lima

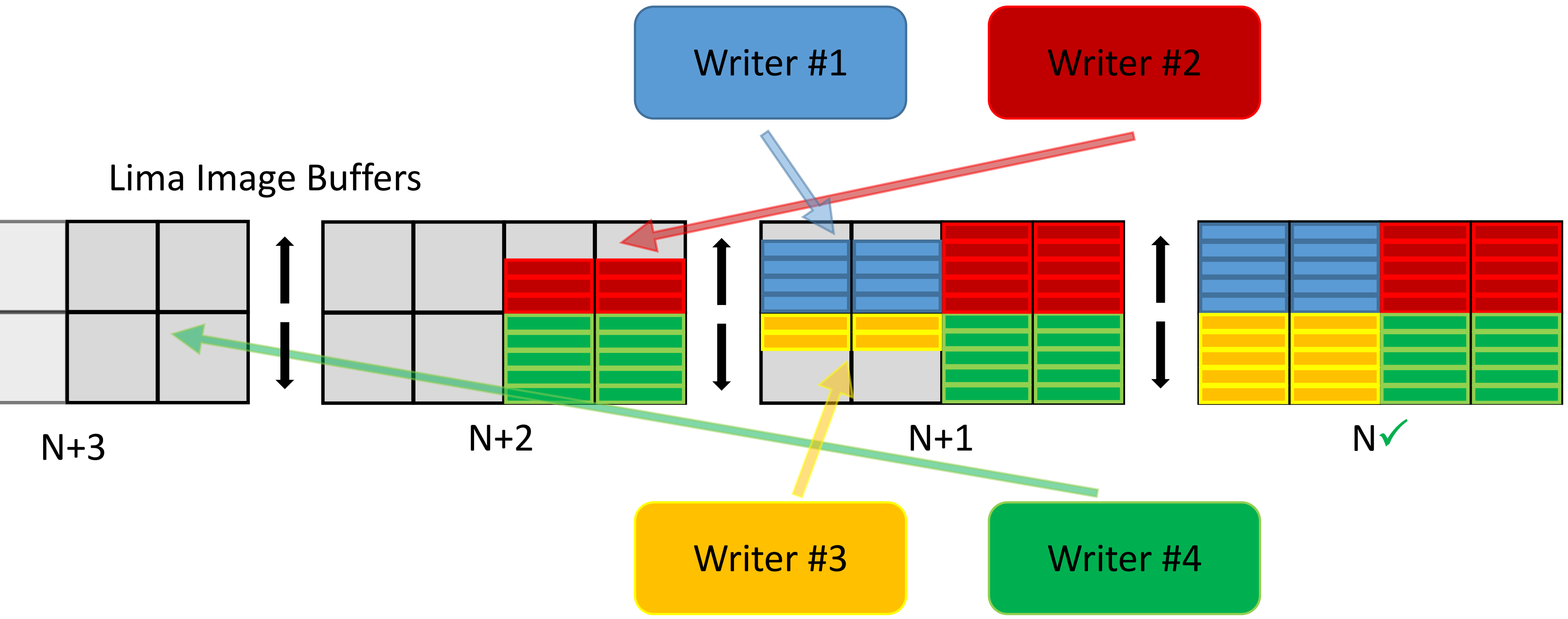


# Lima plugin: frame generation

- Frame dynamics
  - 4 parallel writer callbacks
  - Asynchronous
  - Strict data integrity
  - Must keep a track of frames for each writer
  - Last callback for a frame ⇨ new frame ready event

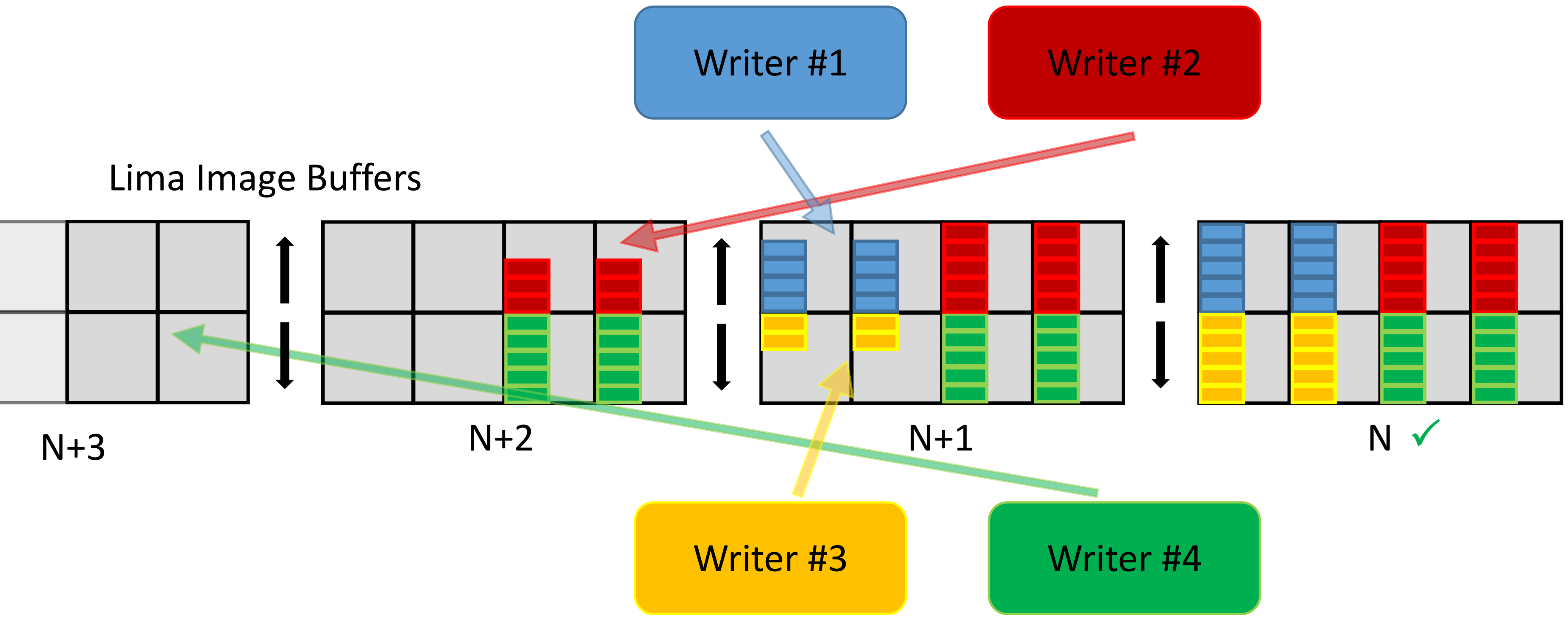
- Frame geometry
  - Readout: center ⇨ border
  - Half-modules are rotated 180°
  - FPGA does horizontal flip ⇨ flip vertically top half
  - Interchip pixels are twice larger:
    - Insert gap
    - Intensity correction

# Lima plugin: frame generation



- Copy chip-by-chip, line-by-line ⇨ vertical flip & inter-chip gap
- Inter-chip pixel intensity ⇨ frame reconstruction task
  - In Lima processing threads

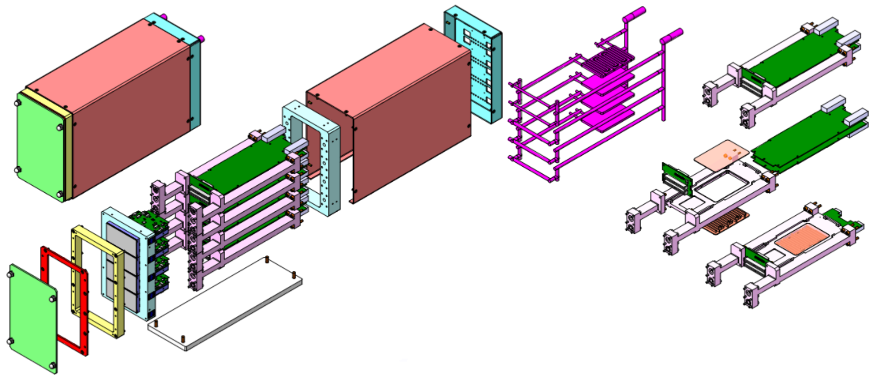
# Lima plugin: frame generation 4-bit

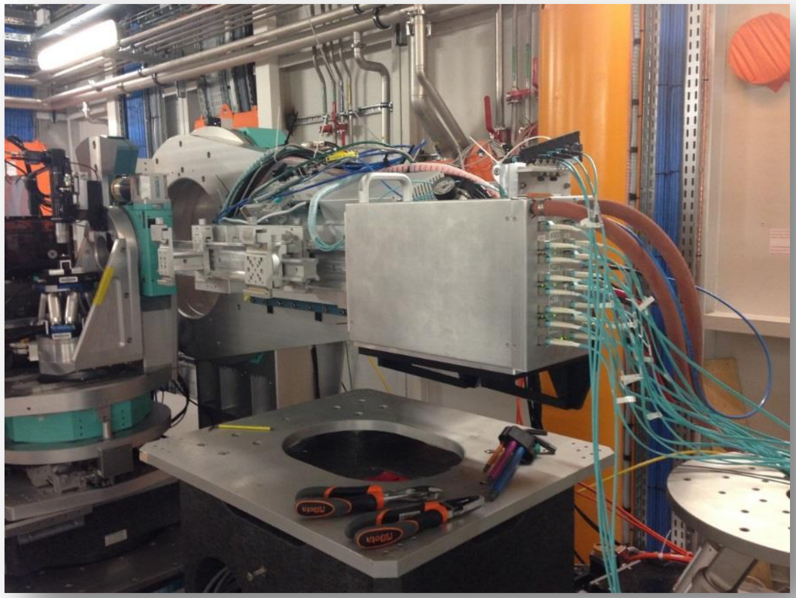


- Not supported in Lima ⇨ convert to 8-bit/pixel
- Copy (2-pixel/byte) packed data in callback, line-by-line
- Expand to 8-bit in frame reconstruction task

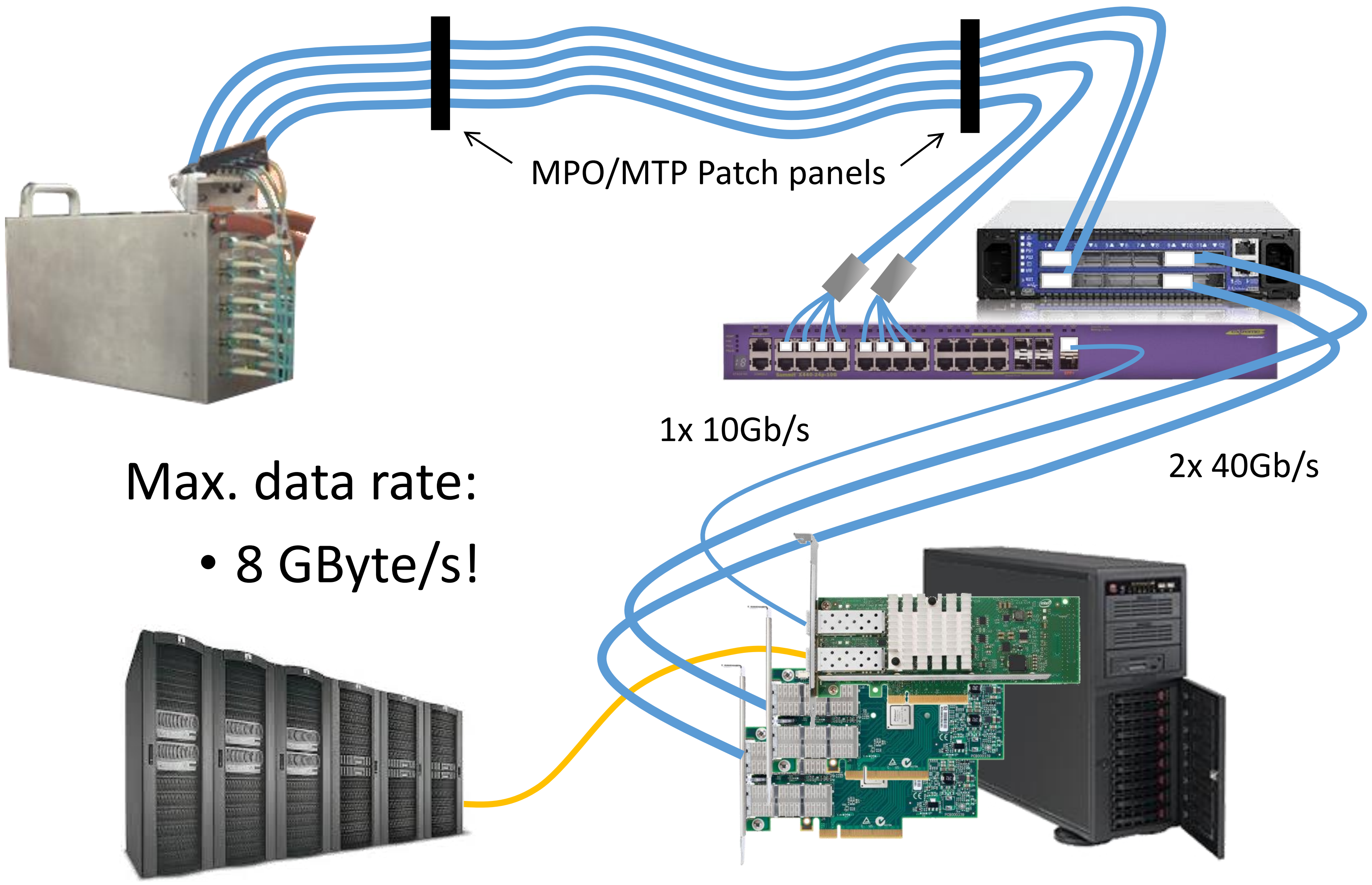
# Eiger-2M: design

- Vertical arrangement of 4x Eiger-500k modules
- 8 half-modules
  - slsDetector takes care of control
  - 4x times more slsReceivers (8)
  - Just need to add a gap in the reconstructed image
- 16 Ethernet links:
  - Control: 8x 1Gb/s
  - Data: 8x 10 Gb/s = 80 Gb/s





# Eiger-2M: network connectivity



Max. data rate:

- 8 GByte/s!

## Performance issues

- Control & image reconstruction ⇨

- Sensor readout is 2.5x faster than data transfer
- Onboard RAM absorbs that difference
- Receive data at max. link speed ⇨ 2.5x faster exposures
- Manage to read 4-bit @ 8 kHz ⇨ can acquire at 22 kHz!

But we lose UDP packets!

## Performance issues

Fixes (thanks to PSI colleagues):

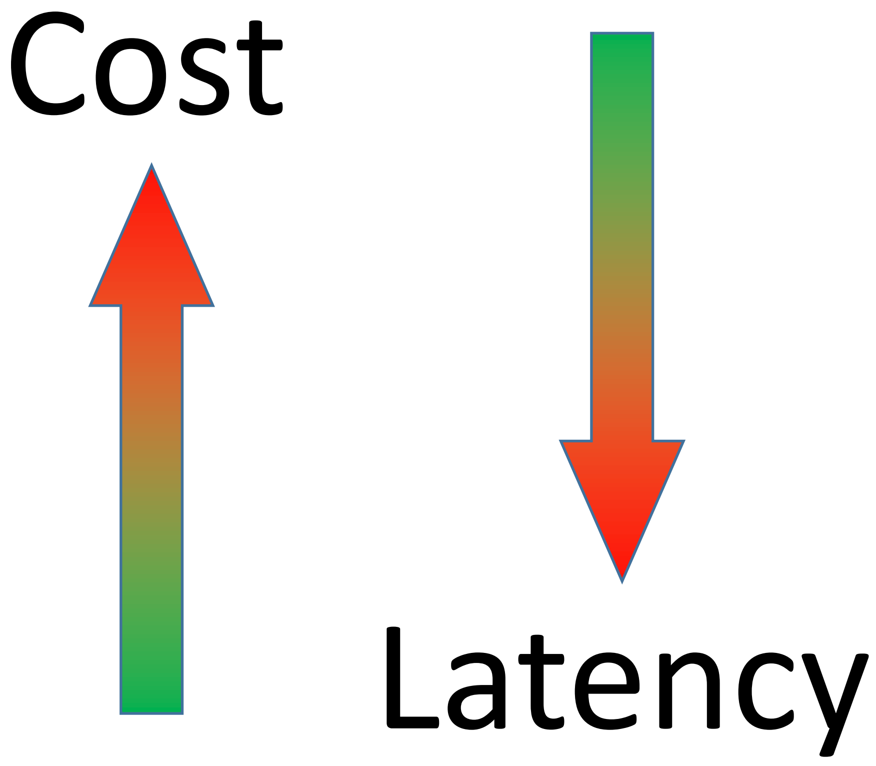
- Ethernet flow control
- IRQ balance and coalescence
- Kernel memory buffer sizes: controller & socket
- Real-time scheduling priority privileges

- Issues in heavy multi-thread environments in:
  - glibc, Lima & slsDetectorPackage (v2.3)

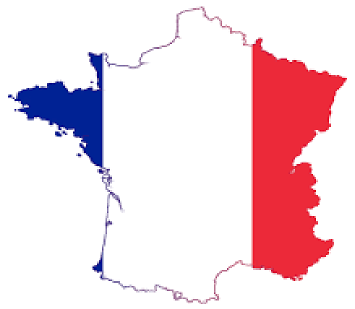
# Frame transfer chain: buffers

- Sensor pixel
- Onboard RAM
- Ethernet adapter
- Kernel: device
- Kernel: socket
- Receiver
- Lima
- Storage servers

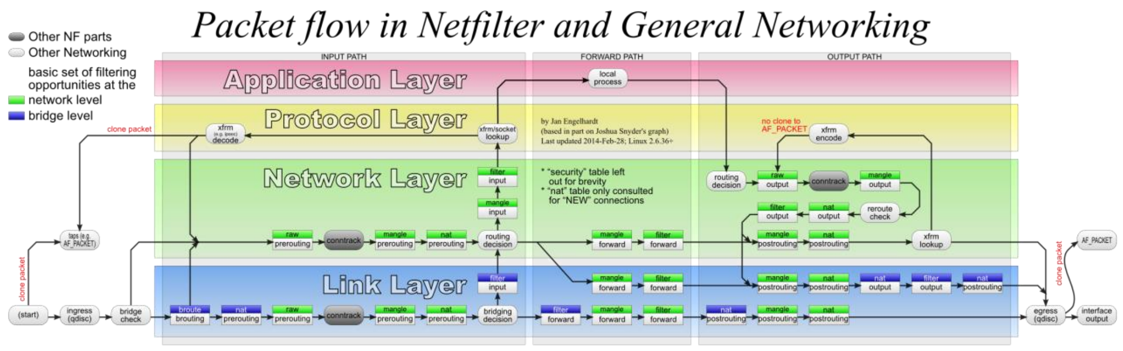
Same bandwidth



# OS vs state administration



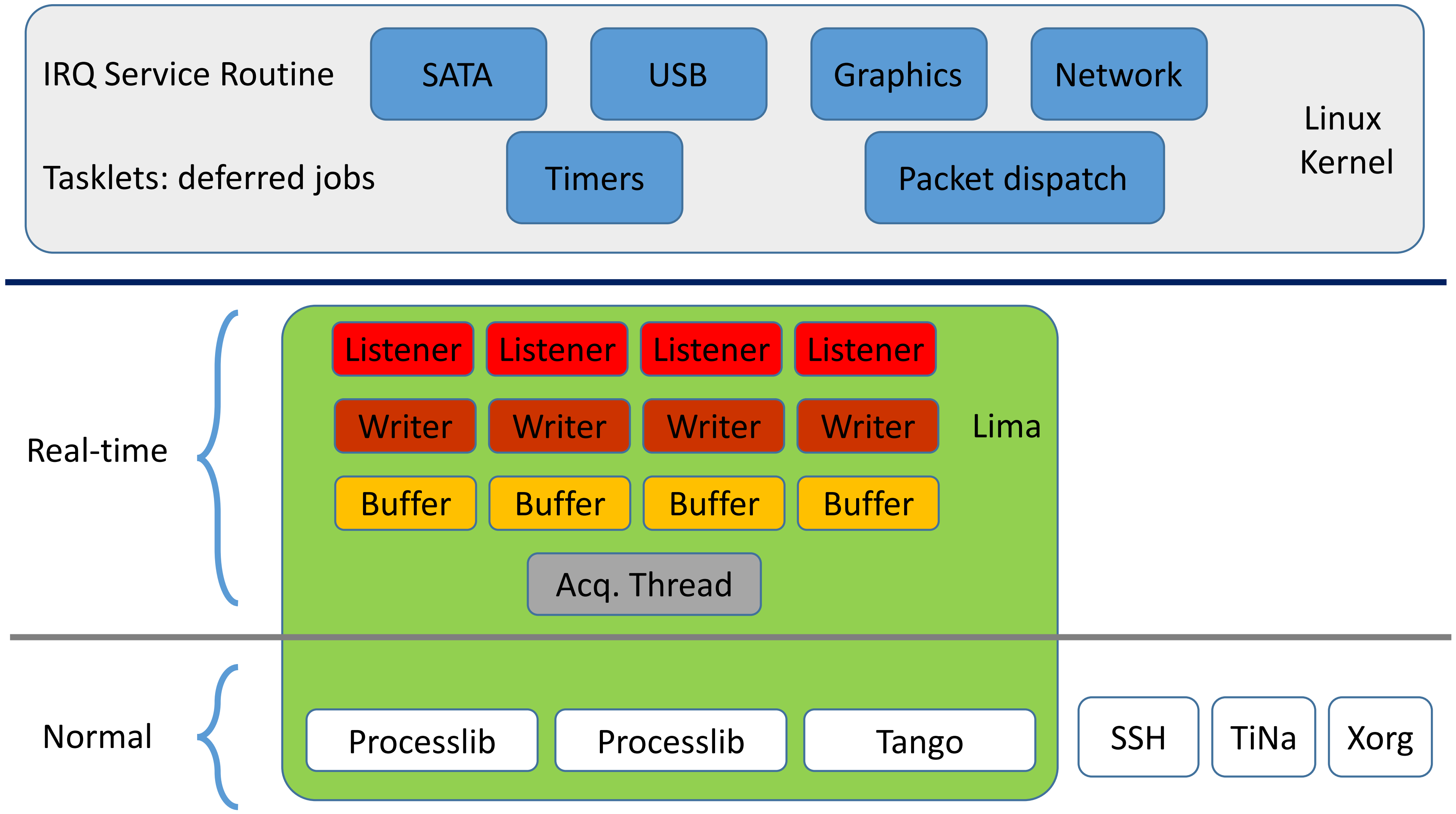
| | | |
|---|---|---|
| You get | Social security | Network & multi-processing |
| You must | Pay taxes | Share your resources: CPU |
| You suffer | Bureaucracy | Packet dispatching in network stack |



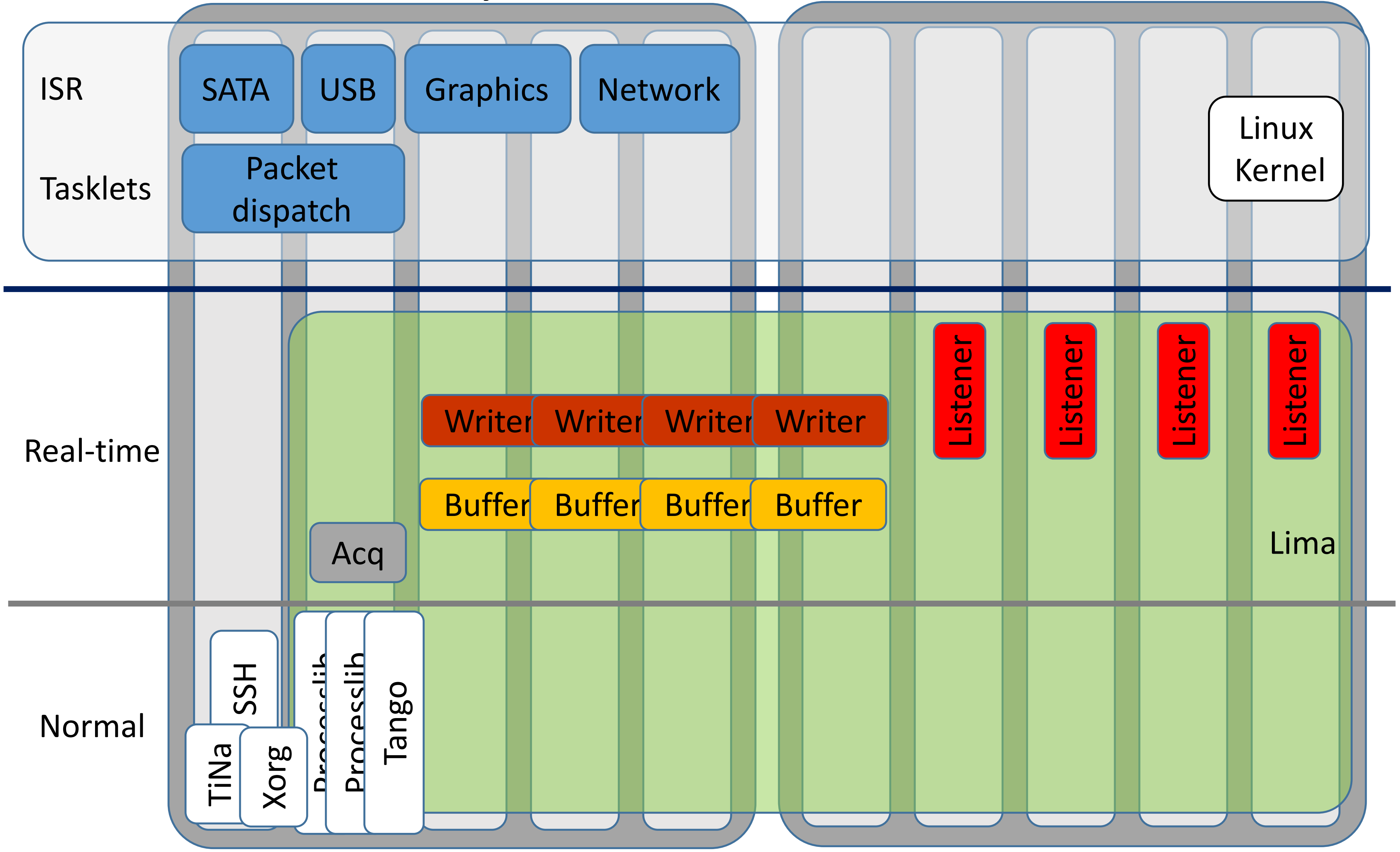
## Frame transfer: latency jitter

Buffers and
latency and jitter
explained

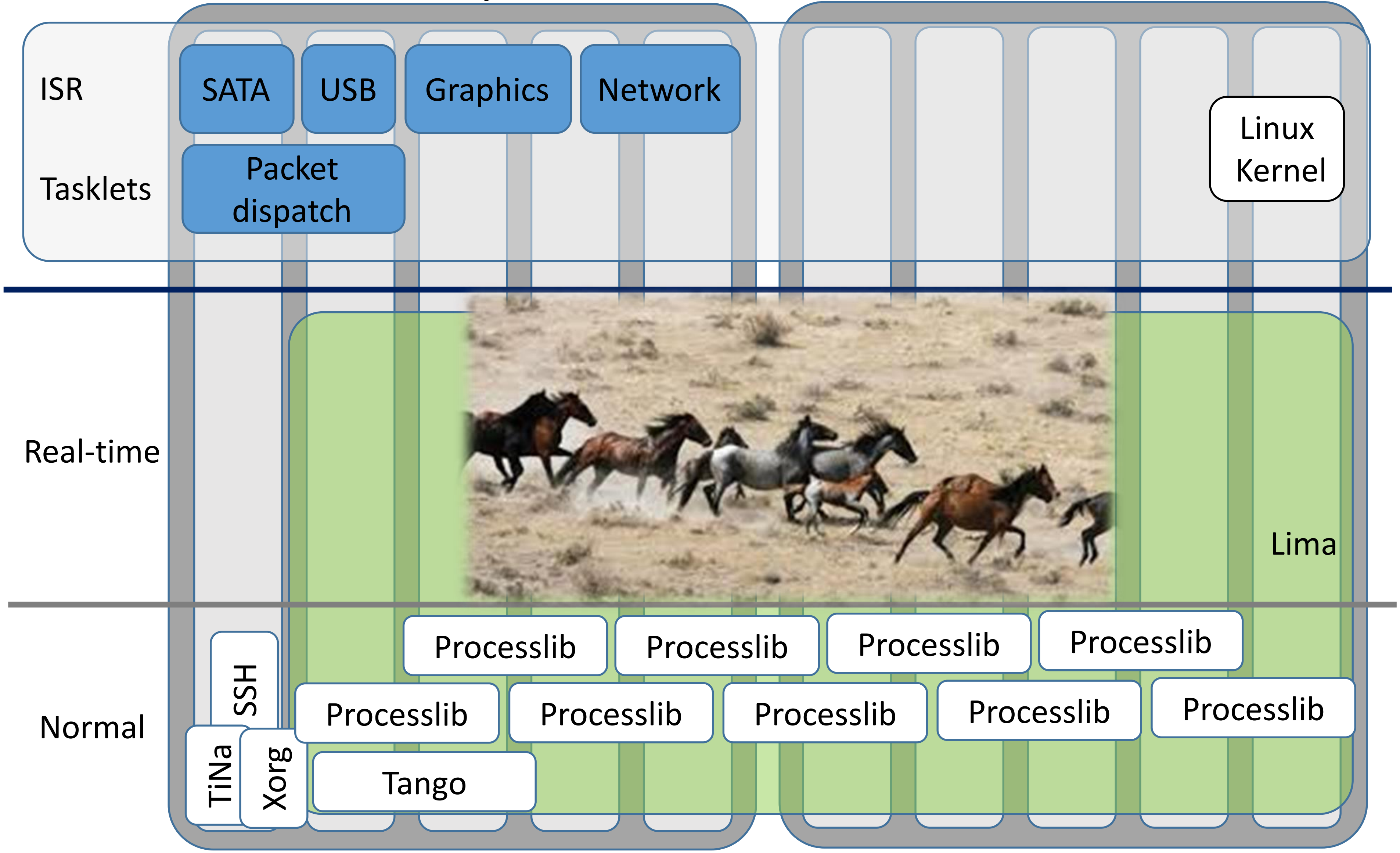
# Task scheduling priority



# CPU affinity control



# CPU affinity control



## CPU affinity control

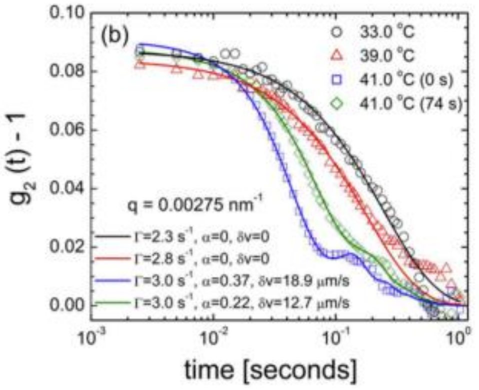
We do pay taxes!

- A watchdog sub-process restore original CPU affinity when Lima ends

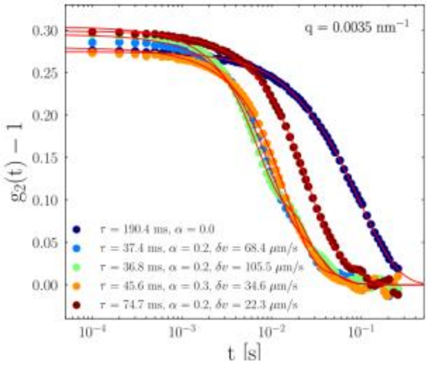
# Results

- Fastest frame acquisition with Eiger-500k
  - 4-bit @ 22 kHz: 30,000 frames
- Data transfer:
  - 8 kHz – 2 GByte/s
- No packet lost!
  - < $10^{-6}$

From T. Zinn and al., **ID02**:



Active dynamics of Janus particles in a phase separating medium
Pilatus 300k detector.



Active dynamics of silica particles in a phase separating medium
EIGER 500k detector.

**20 times faster than Pilatus 300k Detector!**

# Future I

- Port to *slsDetectorPackage* v3.1
- Optimise Eiger-2M performance
  - Now can work in 12-bit @ 750 Hz (3 GByte/s)
  - Try a more powerful backend computer
- Use SIMD instructions for 4-to-8-bit expansion
- Try *readv* multi-buffer system call for UDP input
- Explore other alternatives: DPDK
- Pseudo-flow control: TX frame delay

## Future II

- Improve local storage speed: ZFS
- Test with HDF5 / compression / GPFS
- Hardware based solutions
  - Help in DMA management
  - Frame reconstruction 4-to-8-bit expansion
- Lima v2:
  - Optimised buffer management
  - Decoupled data acquisition & processing
- Lima v3:
  - Multi-backend systems

## Acknowledgements

- PSI/SLS Detector Group:
  - Dhanya Thattil, Martin Brückner, Gemma Tinti, Erik Fröjdh
- ID01, ID02, ID10 and BM05 staff
- Lima developers:
  - Laurent Claustre, Sebastien Petitdemange, Sammuel Debionne
- ESRF ISDD Detector Unit

## Conclusions

- The PSI/Eiger detectors have been integrated in the BLs
- Data acquisition can fully exploit the Eiger-500k
- Eiger-2M full performance is a big challenge
- Hardware development should consider software problems in the solution
- UDP protocol is “delicate”
- High-level flow control protocols are desirable
- Frameworks like RASHPA are strongly encouraged

¡Muchas gracias!