ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THMBCM031

LImA2: EDGE DISTRIBUTED ACQUISITION AND PROCESSING FRAMEWORK FOR HIGH PERFORMANCE 2D DETECTORS

S. Debionne, A. Homs Puron †, J. Kieffer, R. Ponsard, L. Claustre, A. Götz, P. Fajardo
European Synchrotron Radiation Facility (ESRF), Grenoble, France

Abstract

The continuous increase in the data rate of high-performance 2D detectors for photon science makes more and more difficult to perform data acquisition (DAQ) and online processing (ODA) on a single computer. This work presents the LImA2 framework for distributed image detector DAQ & ODA, targeting scalability and low-latency by means of efficient use of system resources, including hardware accelerators. The system controlling the PSI Jungfrau-4M detector, in operation at the ESRF ID29 Serial MX beamline is described. A dual IBM Power9 computer implementation using GPUs provide data sparsification and on-the-fly identification of diffraction peaks at a continuous rate of 500 frames/s. A LImA2 plugin for the ESRF Smartpix detector, based on the high-performance RASHPA data transfer architecture is also presented. Finally, this work includes an update on the current developments for Dectris Eiger2 support and integration in BLISS, the ESRF new beamline control system.

INTRODUCTION

LImA [1], a Library for Image Acquisition, was developed at the ESRF to simplify and standardize of the integration of 2D detectors in synchrotron beamline experiments. By abstracting the image generation capabilities of the detectors, LImA separates the manufacturer's Software Development Kit (SDK) from highly multi-threaded data processing algorithms. Such approach allowed the integration over the last twenty years of tens of different detectors with a unified interface, making LImA the de-facto 2D data acquisition (DAQ) reference software in the Tango synchrotron community similar to areaDetector [2] in the EPICS community.

During the last twenty-five years, the data throughput of 2D detectors at the ESRF has increased by about thirteen times every decade, exceeding the 10 GBytes/s by 2020. Assuming this trend, future detectors will surpass 100 GByte/s by 2030. Therefore, extracting the meaningful information from the raw data and reducing the data volumes becomes more and more critical for data manipulation and storage. Although specific implementations combine high-performance computing with hardware accelerators like FPGA and GPGPUs [3], such demanding tasks cannot be always performed by a single computer in a generic way. Alternatively, using a data centre cluster normally implies an additional latency, affecting online data analysis and fast experiment feedback. Finally, the original LImA architecture was not designed to run on multiple computers and implementing such functionality would be as difficult as starting a new project from scratch.

The LImA2 project was initiated with the aim of addressing all these challenges by providing a scalable, distributed data acquisition (DAQ) and low latency processing framework for high-performance, high-throughput 2D detectors. This paper presents the design considerations of LImA2, the details of the implementation and the current status of the project, including practical implementation such as the PSI/Jungfrau-4M application at the ESRF ID29 for Serial Macromolecular Crystallography (MX).

SYSTEM TOPOLOGIES

Three main system topologies have been identified so far, shown in Fig. 1. The partial frame dispatch topology associates one DAQ/Processing computer to each independent module in a tiled array. The full frame dispatch topology is adapted to detector systems with the capability of sending different frames to different computers.

Depending on the specific needs of the application, one topology can be better adapted to the processing algorithms than the other. For instance, tomography and XPCS techniques can have optimized pipelines for partial frame dispatch topology, while radial integration of diffraction and scattering images is easier to obtain with full frame dispatch.

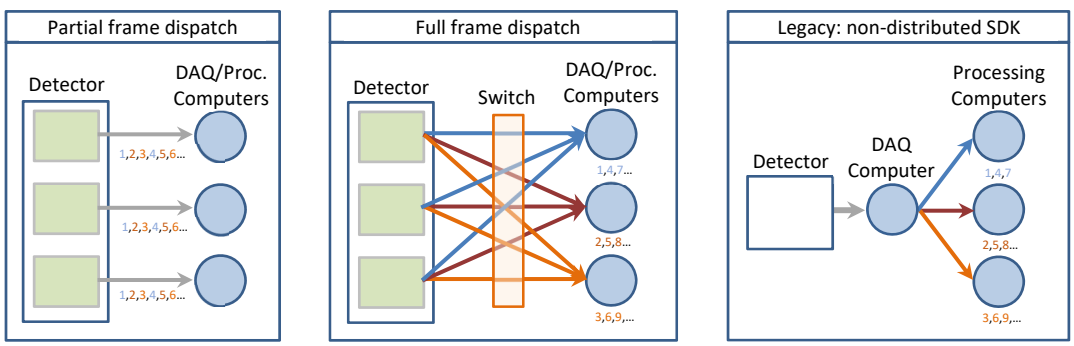

Figure 1: Currently considered LImA2 topologies.

† alejandro.homs@esrf.fr

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THMBCM031

In any case, given the capabilities of the detector system, the main goal is to reduce as much as possible the inter-process communication and synchronization between pipelines.

The third legacy non-distributed topology is foreseen for detectors which do not support parallel data streams. It requires a single detector DAQ computer that then dispatches the data to a second stage of processing nodes

DESIGN CONSIDERATIONS

A diagram of the LImA2 application structure is shown in Fig. 2. LImA2 also separates detector DAQ, implemented in the hardware plugin, from the data processing pipelines. The detector plugin is in turn divided into a control object and one or more parallel data receiver objects. They are responsible for configuring the detector and its data transfer interfaces, starting the acquisition and injecting the data into the processing pipeline. Each data receiver object is connected to a dedicated instance of the pipeline, reducing as much as possible the latency of the results.

In contrast to the previous single pipeline model, LImA2 favors optimized pipelines for specific applications. Each pipeline defines its complete set of configuration parameters and the API for accessing to its results. Another new functionality is the possibility to start a new DAQ sequence before waiting for the end of the previous pipeline's processing. This requires a stronger decoupling between the detector plugin and the pipeline structures, allowing multiple pipeline instances from different acquisitions to coexist.

One problem encountered in LImA is the difficulty to fix the order of initialization of the different subsystems. For instance, the detector timing capabilities could depend on the size of the image, while in other detectors the pixel-depth may depend on the acquisition frame rate. In order to avoid such inter-dependence loops, LImA2 requires the complete specification of all acquisition and processing parameters when calling *prepare_acquisition*.

IMPLEMENTATION DETAILS

Like its predecessor, LImA2 is a modular library providing components intended to be used to build dedicated applications. It is written in C++-17 and uses several Boost [4] libraries: GIL, JSON, Describe, Hana, Pool, Serialization, among others. MPI was chosen for inter-process communication, providing high-performance synchronization and data transfer mechanisms. Usage of GPU for image processing is one main goal of LImA2; OpenCL is integrated through Boost.Compute and NVIDIA Cuda is also used.

Intel OneAPI [5] Thread Building Blocks (TBB) has been chosen as the default platform for implementing processing pipelines. Using graph parallelism, computations are represented by nodes and the communication channels between these computations are represented by edges. A typical processing pipeline includes an input frame FIFO, filled by the data receiver and consumed by a thread injecting the graph source node. A legacy, default pipeline implements basic operations like pixel binning, region-of-interest (ROI) selection, image flipping/rotation, RoI statistics and HDF5/Nexus file saving.

LImA2 code aims to adapt to a wide variety of detector and processing pipelines. Following modern best practices, most of the code is implemented as C++ template classes in order to ease the integration of detector and processing specific data types and algorithms. As a consequence, the generated binary code can be highly optimized by the compiler for a particular detector-processing combination. Python bindings for several classes are also provided through pybind11.

In order to simplify the implementation while ensuring scalability, a LImA2 application consists of several processes, running on the same or on different computing nodes. Considering that the manufacturer SDK can have different hardware and software requirements for detector control and for the data reception, the library separates the control process from the data receiver process(es) so they can be executed in different contexts. An SDK that requires control instances on each data receiver can still be implemented with a unique entry point control process that dispatches commands to subordinate instances.

A LImA2 client library, essential for its integration in any control system, is provided in Python. It allows the configuration of the detector and the data processing, controlling the DAQ, monitoring the pipeline status and retrieving the results.

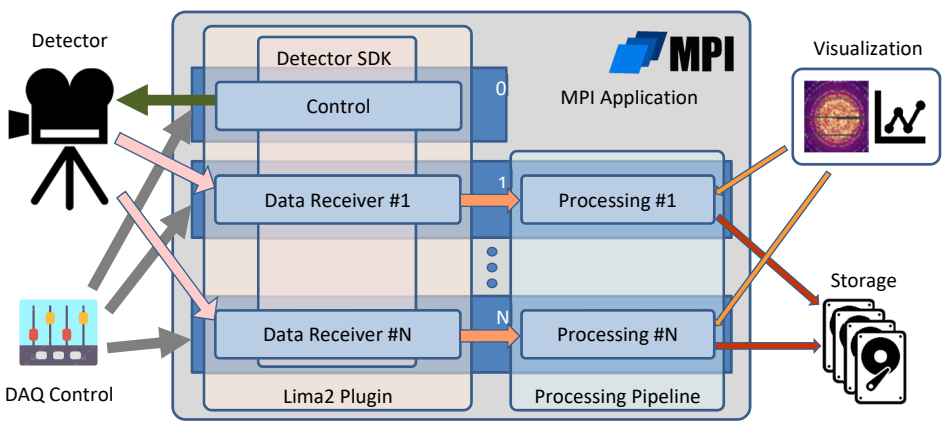

Figure 2: Structure of a LImA2 application.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THMBCM031

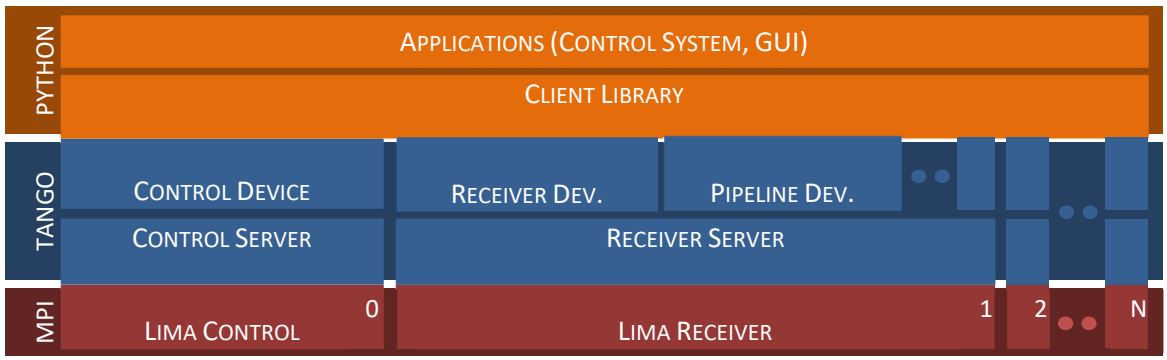

Figure 3: LImA2 software stack.

Tango is used as communication protocol between the Python client(s) and the C++ servers. The high-level application does not need to be aware of the system topology, so the system can scale to more processing nodes in a transparent manner to the control system. Fig. 3 shows the structure of the LImA2 software stack.

PSI JUNGFRAU-4M FOR SERIAL MX

The PSI Jungfrau-4M [6], installed at the ESRF ID29 beamline [7], is a charge integrating detector featuring per-pixel and per-frame automatic gain switching capabilities, attractive to x-ray photon science experiments requiring a high dynamic range at high speeds. Supporting up to 2 kHz frame-rate, Jungfrau generates an UDP data stream of 8 GByte/s at 1 kHz operation. An ESRF variant of the PSI slsDetectorPackage receiver [8] implements several optimizations that allows receiving the 1 kHz packet stream into the computer RAM without data loss. Dedicated stream listener threads and low-level packet buffers ensure reception data integrity.

In addition to image geometry reconstruction, a preliminary pedestal and gain correction is needed in order to obtain pixel values corresponding to photon flux. This is an intensive computation task requiring six calibration maps. The implementation in LImA2 uses OpenCL on GPUs to assemble the UDP packets and perform pedestal/gain correction. The raw images in UDP packet format can be independently saved from the geometrically-assembled and intensity-corrected images.

As show in Fig. 4, the detector is connected to a 100 GbE switch through sixteen 10 GbE fiber-optic cables. The switch aggregates the raw data into two 100 Gbit links towards the ESRF data centre, where two identical IBM Power9 AC-922 computers are dedicated to the detector DAQ. Each system is equipped with two Power9 processors hosting to two NVIDIA Tesla V100 SXM2 (32 GB) GPUs with high-speed NVLink. A Mellanox ConnectX-6 100 Gbit adapter featuring the PCIe multi-host extension is installed on the shared PCIe Gen4 slot of the AC-922s. Such configuration exports an independent network interface on each Power9 for the same Ethernet port, allowing the direct routing of network packets to the corresponding PCIe bus based on the destination MAC address. The LImA2 PSI plugin exploits the round-robin functionality built in the Jungfrau detector to dispatch consecutive images to different data receivers, each running on a dedicated Power9 socket. This scheme corresponds to the *full-frame-dispatch* topology of Fig. 1. Connectivity to GPFS storage servers is implemented through independent 25 Gbit links.

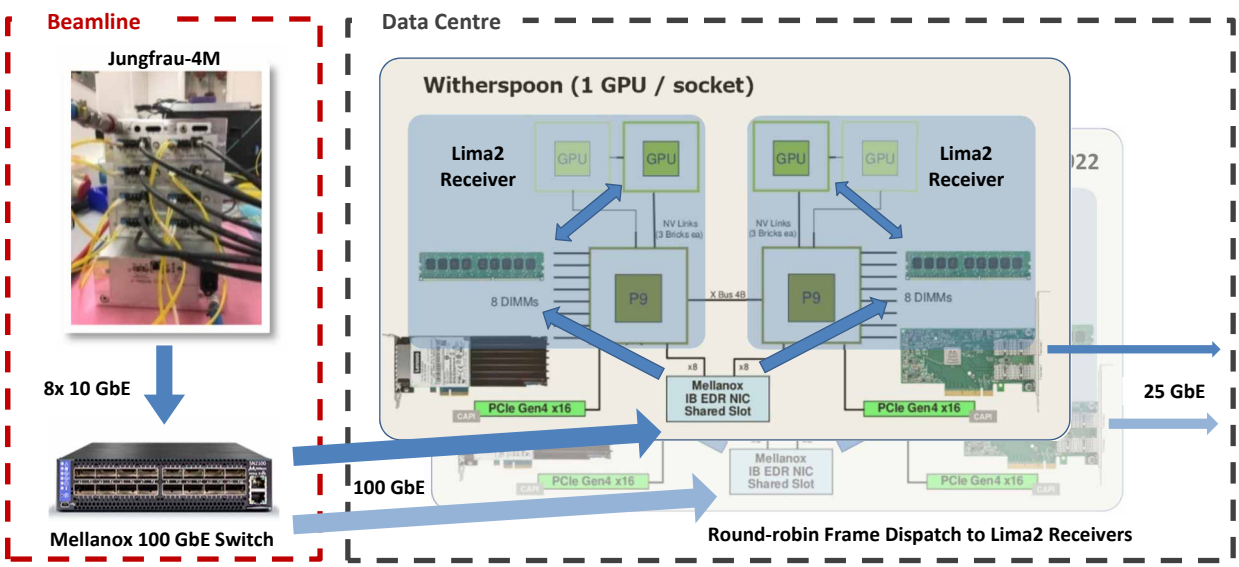

Figure 4: Connection diagram of the LImA2 system for the PSI Jungfrau-4M.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THMBCM031

Typical experiments at ID29 beamline for Serial MX include long DAQ sequences with an effective crystal hit-rate ranging from 20-50 %. Saving all the frames represents an important amount of resources for the central storage servers and it cannot be justified with such levels of meaningless data. Figure 5 shows the data processing flow for the Jungfrau-4M at ID29. A novel approach uses the peak-finding algorithm in pyFAI [9] to extract the scattering background distribution and the diffraction peaks from the corrected image. The SMX pipeline executes the pyFAI OpenCL code for image sparsification and peak-finding (spots). The results include an *is-hit* flag that indicates if the corresponding frame has enough diffraction peaks to be analyzed, this can be used as a veto mechanism in the future. Depending on the experiment conditions and the signal-to-noise ratio threshold used to discard meaningless data, the resulting sparse and spots files can represent a data reduction factor between four and one hundred relative to the corresponding dense corrected data. HDF5 master files containing Virtual Datasets are also saved in order to provide a global, continuous view of the interlaced parallel receiver streams, including the results of the peak-finder algorithm.

HIGH PERFORMANCE OPTIMIZATIONS

Several optimizations have been applied to the running systems to improve the overall performance. In order to avoid CPU scheduling collisions with the tasks receiving UDP packets, the CPU affinity is tuned for the players actively participating in the DAQ context: the network device IRQ handlers and kernel dispatching, the detector stream listener threads, the LImA2 processing threads and the GPFS client daemon. NUMA affinity is also managed for the data receiver processes, in particular the low-level packet buffers. The CPU frequency scaling governor is set to *performance* and the CPU Idle Power Management is disabled. All these system-wide settings are initialized by a dedicated *lima_launcher* utility, also used for tuning system running LImA servers. Finally, the mitigation of the CPU security vulnerabilities is disabled in order to exploit all the optimizations in the hardware.

The IBM Power9 processor include the NX unit performing HW-accelerated GZIP compression and decompression. The Linux kernel distributed by Ubuntu 20.04 does not include the patches for exporting this functionality to user-space applications. The AC-922 systems dedicated to LImA2 are running a patched kernel, notably reducing the CPU load associated to HDF5 dense image saving and improving overall system performance [10]. This ensures a high level of compression with the speed of the fastest compressors on the market (LZ4).

ESRF SMARTPIX & RASHPA

The ESRF Smartpix [11] is a photon-counting pixel detector based on the Medipix3RX chip, composed of one or more functional modules of up to eight chips (512 kPixels). Featuring frame rates as high as 6 kHz @ 8-bit (6-bit pixel capacity), a 1 Mpixel detector can generate up to 6 GByte/s data streams. Smartpix data transfer is based on RASHPA [12], a framework developed at the ESRF for addressing the issues of high-throughput data transfer from the detector to the first layer of computing nodes with a powerful, versatile and scalable architecture. Multi-module and multi-computer systems are an essential part of the RASHPA design considerations, as well as the goal to offload as many tasks as possible from the CPUs. The framework consists on a toolbox of FPGA components and its associated low-level software modules that exploit efficient Remote Direct Memory Access (RDMA), to achieve zero-copy data transfers in modern CPUs, supporting RDMA over Converged Ethernet (RoCEv2).

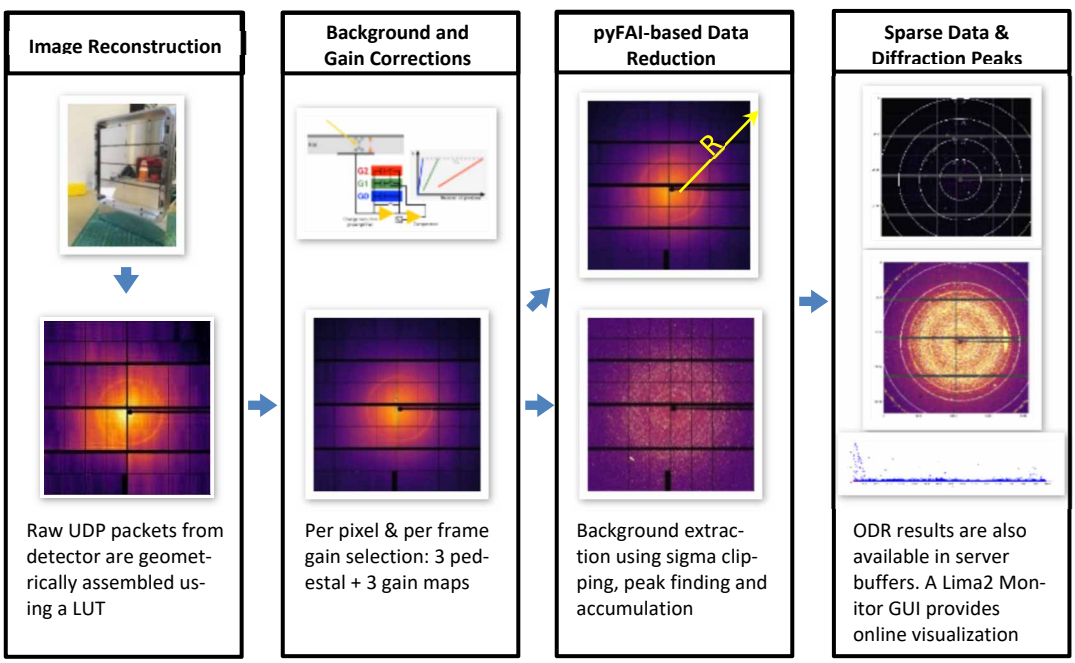

Figure 5: Data processing flow in ID29 PSI Jungfrau-4M for Serial MX.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THMBCM031

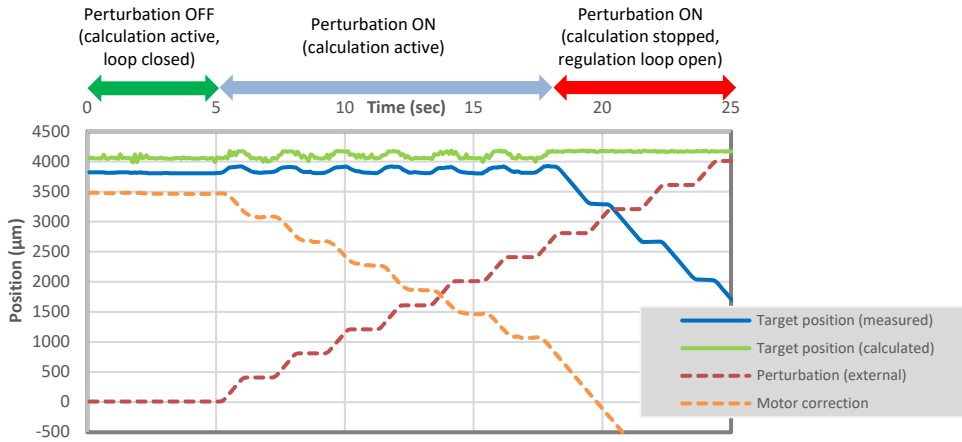

Figure 6: Preliminary real-time GPU-assisted position tracking.

RASHPA intrinsically incorporates advanced functionality like multiple concurrent transfer streams, featuring spatial ROIs and temporal sub-sampling. GPUDirect technology, providing direct transfer to GPU memory, is also supported through Mellanox 100 Gbit RNICs. Two RASHPA hardware implementations are available for the Smartpix detector: PCIe and RoCEv2.

The low-latency capabilities inherent to RDMA data transfer, in the submillisecond range in the case of Smartpix, makes RASHPA particularly suitable for implementing schemes for fast experimental feedback using information extracted from the 2D detector data. A preliminary demonstration of real-time feedback capabilities of the Smartpix detector with RASHPA/RoCEv2 and GPUDirect for synchrotron experiments consisted in the position tracking of a moving sample from X-ray transmission images at 1 kfps, as shown in Fig. 6.

From the conceptual point of view, RASHPA and LImA2 are the hardware and software counterparts of a common, consistent framework for distributed DAQ for 2D high-throughput detectors. In concordance, Smartpix/RASHPA has a natural integration into LImA2.

DECTRIS EIGER2 & INTEGRATION INTO BLISS

Dectris Eiger2 detectors implement the Stream2 protocol, offering multi-band images associated to the dual-threshold pixel capabilities as well as multi-receiver configuration. An Eiger2 LImA2 plugin is in an advanced development stage. The definitive Nexus/HDF5 file format for the generated four-dimensions datasets of multi-band images is under discussion.

A basic LImA2 integration into BLISS was first made in order to control sequences for the Serial MX experiments, lacking the interface to the BLISS scanning engine. The complete integration, including support for *blissdata* (an in-memory data access API), is under development, also in an advanced stage.

CURRENT PROJECT STATUS

LImA2 entered in production at ID29 on January 2023 with a single IBM computer and two parallel data receivers offering dense & sparse saving. The peak-finder functionality and the corresponding online monitor GUI were gradually added in parallel to beamline operation. The system is currently using two IBM systems with four data receivers; it can perform continuous acquisitions at ~500 Hz as well as limited sequences at higher speeds (~150 kframes @ 1 kHz). Dense frames and sparse data are saved in HDF5 files, taking benefit from the [NX] hardware compression engine available in the IBM Power9 processors. The next goal is to reach the continuous 1 kHz operation.

Despite all the optimizations in place, the GPU data throughput associated to dense image saving currently affects the Jungfrau UDP packet reception integrity. Incomplete frames due to packet loss are marked as invalid and ignored by the processing pipeline; the statistical nature of the SMX data collection tolerates small amounts of data loss (~ 1-5 %). The remarkable results obtained with the FPGA-assisted implementation of the Jungfraujoch system [2] motivate its integration into a second variant of the LImA2 Jungfrau plugin.

CONCLUSIONS

LImA2, a framework for distributed high-performance 2D detector DAQ and processing, has been developed, targeting scalable data acquisition systems with low-latency data reduction. An implementation for the PSI Jungfrau-4M detector is in production with two parallel computers at the ESRF ID29 Serial MX beamline, providing online peak-finding and data sparsification at ~500 Hz. The ESRF Smartpix detector, using the high-performance RDMA-based RASHPA data transfer architecture, has been integrated with its two variants: PCIe and RoCEv2, featuring GPUDirect technology, and its real-time capabilities for active feedback experiments have been proven. A plugin for the Dectris Eiger2 supporting multi-band images is in an advanced development stage, as well as its integration into BLISS. By separating the detector plugin from the data processing pipelines, the integration efforts of new high-performance hardware into advanced experimental endstations can be leveraged.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-THMBCM031

REFERENCES

[1] S. Petitdemange, L. Claustre, A. Homs, R. Homs Regojo, and E. Papillon, "The LIMA Project Update", in *Proc. ICALEPCS'13*, San Francisco, CA, USA, Oct. 2013, paper FRCOAAB08, pp. 1489-1492.

[2] areaDetector, https://areadetector.github.io/areaDetector/index.html

[3] F. Leonarski *et al*, "Jungfraujoch: hardware-accelerated dataacquisition system for kilohertz pixel-array X-ray detectors", *J. Synchrotron Radiat.*, vol. 30, pp. 227-234, 2023. doi:10.1107/S1600577522010268

[4] Boost, https://www.boost.org

[5] Intel® OneAPI, https://www.intel.com/content/www/us/en/developer/tools/oneapi/overview.html

[6] A. Mozzanica *et al.*, "The JUNGFRAU detector for applications at synchrotron light sources and XFELs", *Synchrotron Radiat. News*, vol. 31, no. 6, pp. 16-20, 2018. doi:10.1080/08940886.2018.1528429

[7] J. Orlans, S. Basu, D. De Sanctis, "ID29, a versatile beamline for time-resolved serial crystallography at the EBS-ESRF", *Acta Crystallographica A-Foundation and Advances*, vol. 78, pp. E398-E399, 2022.

[8] slsDetectorPackage, https://github.com/slsdetectorgroup/slsDetectorPackage

[9] G. Ashiotis, A. Deschildre, Z. Nawaz, J.P. Wright, D. Karkoulis, F.E. Picca, J. Kieffer, "The fast azimuthal integration Python library: pyFAI", *J. Appl. Crystallogr*, vol. 48, part 2, pp. 510-519, 2015.

[10] NX GZIP, https://github.com/libnxz/power-gzip

[11] C. Ponchut *et al.*, "SMARTPIX, a photon-counting pixel detector for synchrotron applications based on Medipix3RX readout chip and active edge pixel sensors", *J. Instrum.*, vol. 10, p. C01019, 2015. doi:10.1088/1748-0221/10/01/C01019

[12] W. Mansour, R. Biv, C. Ponchut, R. Ponsard, N. Janvier, P. Fajardo, "FPGA Based Real-Time Image Manipulation and Advanced Data Acquisition For 2D-XRAY Detectors", *IEEE Trans. Nucl. Sci.*, vol. 68, issue 8, pp. 1927-1932, 2021. doi:10.1109/TNS.2021.3086416