REPORT ON CONTROL/DAQ SOFTWARE DESIGN AND CURRENT STATE OF IMPLEMENTATION FOR THE PERCIVAL DETECTOR

A.S. Palaha, C. Angelsen, Q. Gu, J. Marchal, N. Rees, U.K. Pedersen, N. Tartoni, H. Yousef, Diamond Light Source, Harwell Science and Innovation Campus, Oxfordshire, UK
M. Bayer, J. Correa, P. Gnadt, P. Gottlicher, H. Graafsma, S. Lange, A. Marras, S. Reza *, I. Shevyakov, S. Smoljanin, J. Supra, M. Tennert, U. Trunk, C.B. Wunderer, Q. Xia, M. Zimmer, Deutsches Elektronen-Synchrotron (DESY), Hamburg, Germany
D. Das, N. Guerrini, B. Marsh, T. Nicholls, I. Sedgwick, R. Turchetta, Rutherford Appleton Laboratory (RAL), Harwell Science and Innovation Campus, Oxfordshire, UK
G. Cautero, D. Giuressi, A. Khromova, R. Menk, G. Pinaroli, L. Stebel, Elettra Sincrotrone Trieste, Italy
H.J. Hyun, K.S. Kim, S. Rah, Pohang Accelerator Laboratory, Pohang, Korea

Abstract

The Percival Collaboration is developing a high-speed, low X-ray energy detector capable of detecting single photons while maintaining a large dynamic range of sensitivity.

The increased brilliance of state-of-the-art Synchrotron radiation sources and Free Electron Lasers require imaging detectors capable of taking advantage of these light source facilities. The PERCIVAL ("Pixelated Energy Resolving CMOS Imager, Versatile and Large") detector is being developed in collaboration between DESY, Elettra Sincrotrone Trieste, Diamond Light Source and Pohang Accelerator Laboratory.

It is a CMOS detector targeting soft X-rays < 1 KeV, with a high resolution of up to 13 M pixels reading out at 120 Hz, producing a challenging data rate of 6 GiB/s.

The controls and data acquisition system will include a Software Development Kit to allow integration with third party control systems like Tango and DOOCS; an EPICS [1] areaDetector [2] driver will be included by default. It will make use of parallel readout to keep pace with the data rate, distributing the data over multiple nodes to create a single virtual dataset using the HDF5 file format for its speed advantages in high volumes of regular data.

This development project will culminate in a control and DAQ system capable of dealing with very high data rates while providing easy integration with site-specific control systems.

This report presents the design of the control system software for the Percival detector, an update of the current state of the implementation carried out by Diamond Light Source.

INTRODUCTION

The PERCIVAL detector is designed for high-speed and low noise detection of soft X-rays. The Percival CMOS chip is a large pixel array of ~13 mega-pixels with 15 bits per pixel (packed in 16 bit words) which encodes a substantially higher dynamic range of the sensor as described below [3]. This leads to 25 MiB per frame. For each data frame the detector will also produce a reset frame of the same dimension, so the total data produced is 50 MiB per frame. There will also be a ~2 mega-pixels version [4]. For the designed frame-rate of 120 Hz this leads to an overall data rate of ~6 GiB/s [5].

The Detector

The detector pixel is an adaptive gain design [4], allowing single photon discrimination at low flux while still being capable of high flux measurements at the cost of increased noise. It comprises a diode and three capacitors, each of increasing capacitance, to collect charge depending on whether the next smallest capacitance component has been filled; thus "adaptive-gain". As there are four "collectors" per pixel, there are four possible gain values also. A pixel array feeds scrambled data to a pair of "mezzanine" cards, each with an FPGA to perform a certain amount of low level unscrambling and formatting the data into UDP packets for transfer to the processing nodes. These nodes are a Linux cluster of 8 commodity servers, as recommended by an earlier feasibility study [3] in order to handle the processing of the expected data rate.

CONTROL SYSTEM ARCHITECTURE

Shown in Fig. 1 is an overview of the proposed Percival software architecture, where the green single direction arrows represent the flow of data and purple bidirectional arrows represent control links. The top half represents the data acquisition system, with the sensor board having its data collected by the carriers boards via LVDS lines and transferred to the mezzanine boards before it is sent through the deep buffer switch and onto the Linux cluster.

Each FPGA of the mezzanine board has four 10 GBps links to the deep buffer switch, one for control and three for transmitting data. Since there is a chance of UDP packet loss, and the goal for maximum packet loss is less than 1 in 10^6, a lightweight retry protocol will be used to ensure this aim.

The Linux cluster software stack and user workstation software suite comprise the control software that is being

* Mittuniversitetet, Sundsvall, Sweden

ISBN 978-3-95450-148-9

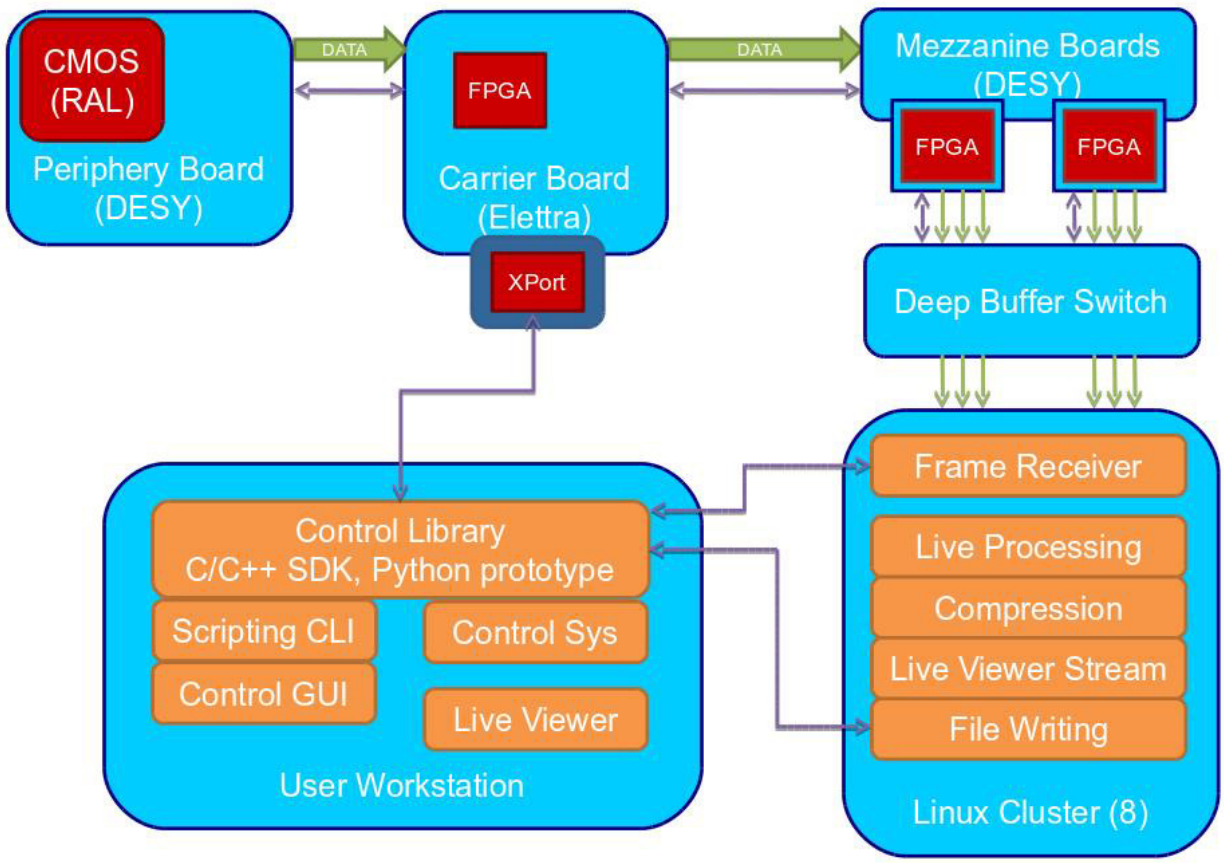

Figure 1: Percival Software Architecture: purple arrows represent control links, green arrows represent data links.

developed at DLS. The Linux cluster will run a chain of software for grabbing the data off the switch through 10 GBps Ethernet links for viewing and writing the processed data to file. The user workstation software will provide slow control of the detector and data acquisition, made of a core C/C++ library, with a Software Development Kit (SDK) to provide a means to interface with control systems.

Software Development Kit

The SDK will allow interfacing with controls systems such as EPICS, TANGO, DOOCS and others. A Graphical User Interface (GUI) can be applied with it to the control interface. The control software will come with EPICS and areaDetector support built in.

LINUX CLUSTER SOFTWARE

The Linux Cluster Software encompasses all the software involved in the live processing of the detector data. Live processing is necessary as it is required to be able to run the detector continuously, that is not only in burst or buffering mode, but also to take data almost indefinitely. The architecture is of a parallel processing pipeline with each piece of the software processing the data in turn. Each complete frame of detector data will be sent to a separate node in sequential order.

Frame Receiver

The Frame Receiver enables reception of frames of detector data transmitted over a network connection as a stream. It constructs data frames in shared memory buffers and communicates with external applications to allow processing and storage. It collects the data from the deep buffer switch into the cluster of Linux nodes, placing them in memory buffers to be accessed by the next step in the chain.

Live Processing

The live processing of the detector data into use-able images is implemented in a highly optimised C++ library.

The detector data at this stage has the format of a 16-bit integer per pixel. This is split into three parts; 2 bits for the gain multiplier, 8 bits for the fine Analogue Digital Converter (ADC) code and 5 bits for the coarse ADC code, with 1 bit left unused. The fine and coarse Analogue Digital Unit (ADU) codes represent the analogue voltage from each pixel as converted by the detector ADCs.

ISBN 978-3-95450-148-9

The first processing step is to decode the integer coarse, fine and gain codes from the packed 16-bit integer. The coarse and fine codes are converted to a floating point number representing the intensity measured by each pixel. This is called the ADU calibration, and requires a static array of offset and gain values for each of the seven coarse and fine ADCs.

Multiplying the ADU calibrated float by the pixel gain is the next step, the gain is one of four values encoded by the two gain bits of the now unpacked 16-bit integer. The gain values depend on which of the capacitors or the diode is read out from the pixel to the ADC, and depends on which pixel is read out (as the capacitances of the pixel components are not completely uniform over all the pixels).

Next, if the gain corresponds to the photo-diode of the pixel, i.e. the highest gain value, then the reset frame that has also undergone the previous processing steps must be subtracted from the sample frame. This is called the Correlated Double Sampling (CDS) subtraction step, reducing systematic errors in the signal.

Finally, dark image subtraction is performed by removing a constant frame from the result.

The coarse and fine offsets and gains, as well as the pixel gains and the dark frame are all intrinsic to the detector, acquired before measurements are taken and stored in static arrays available in memory for the processing steps to use.

In developing and optimising this library [6], the primary goal is achieving the fastest processing time, indicated by the bandwidth, i.e. the amount of data processed per unit time for both the sample and reset frames. Profiling tools were used to identify the most significant contributions to processing time and potential bottlenecks, and the library was developed as a processing chain so as to swap in and out different versions of the processing steps within the profiling and optimisation framework. Much of the tuning applied here is specific to the hardware and micro-architecture of the system.

- The first step taken to optimise the library was reducing the computation time in the first processing step by changing the floating point division to a multiplication of the denominator's inverse.
- The GCC compiler offers three levels of optimisation, and various techniques offered by the C++ language were employed.
- Parallelisation is employed using Intel Threading Building Blocks (TBB) in order to spread the processing of one node over multiple threads. It was found that the optimal number of logical threads was to use the number of physical threads, although some subtleties were encountered.
- Vectorising the operations such that calculations are performed on multiple pixels at the same offers further improvement in processing bandwidth, using the AVX (Advanced Vector eXtensions) instruction set that is enabled on Intel Ivybridge and Sandybridge architectures.
- Alignment of the data with the available vector sizes also improved throughput, requiring that the multiples of 7 pixels (due to the detector chip having 7 ADCs each with which to read out rows of pixel data) be padded with an empty slot.

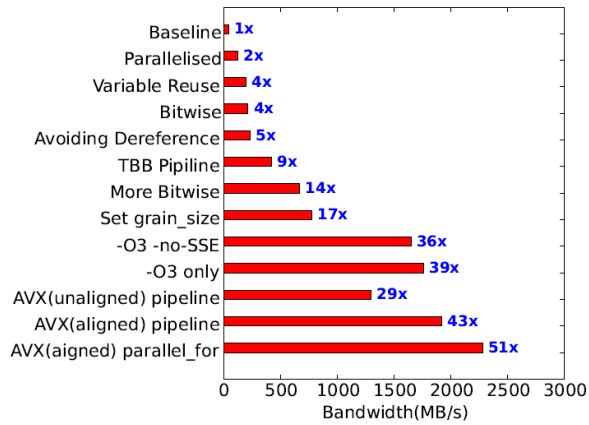

Figure 2: Optimisation steps versus processing bandwidth, with comparisons to the baseline [6].

The results of the various optimisation steps and the impacts on bandwidth are shown in Fig. 2. Improvements were optimal when parallelisation, compiler optimisation, replacement with bitwise operations and the AVX with a particular template better suited to large bandwidths is used.

Figure 3 shows the linear dependence of processing bandwidth with up to roughly 10 threads, after which there appears to be a saturation at ~2.5 GiB/s. The result shown is using a grain size, or chunk of data to process from the cache at one time, to be a factor of the number of pixels in the frame. The grain size is optimised to be a multiple of 7 (to match the number of ADCs, and the number of pixels rows is a multiple of 7), and is optimised to fill as much of the memory cache of the server node without saturating it. A grain size of 3528 was used for the result measured in Fig. 3.

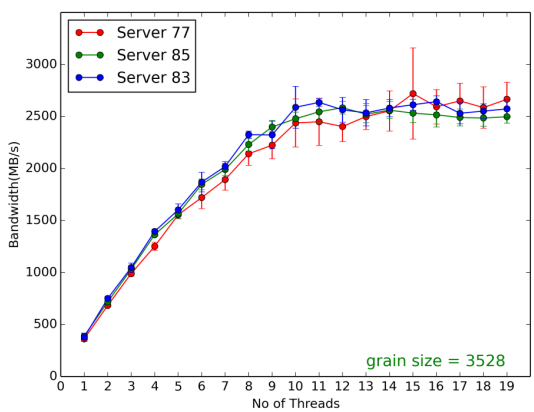

Figure 3: Processing bandwidth versus the number of threads [6].

ISBN 978-3-95450-148-9

Overall, the optimised library is found to run stably on a single node at (2.5 ± 10%) GiB/s, which is 59× the baseline bandwidth, and more than 3× the required bandwidth to achieve the required data processing bandwidth.

Compression

Compression of data before writing to file has been investigated, with optimised parallel compression of interest in particular, to exploit the Linux cluster multi-core architecture. The "blosc" compressor tool [7] was used as it specialises in fast compression and decompression, use of multi-threading capabilities, and it can be applied to HDF5 data when writing to or reading from a disk. Initial and very basic measurement has yielded encouraging results but require further tests and development.

Live Viewer Stream

The live viewer requirement is only a reduced data-rate, sacrificing combinations of frame-rate, resolution and/or region-of-interest. Components for streaming the live data after processing on the Linux cluster already exist so will be incorporated into the Linux cluster software. This is also true for the live viewer; making use of an in-house developed version of an areaDetector plug-in that utilises the ffmpeg libraries to stream live data.

File Writing

File Writing is performed by an in-house developed file writer based on the HDF5 format. The HDF5 group is implementing support [1] for Single Write Multiple Read (SWMR) to the HDF5 libraries, allowing for live viewing of the data while the detector is capturing, processing and writing images [8]. This is an essential feature for long measurements, to inspect the data early throughout the data acquisition.

Also implemented is direct chunk writing [2]. This direct chunk write feature is also key in allowing us to do high-performance (i.e. outside of the HDF5 library) compression with something like blosc.

As each node of the Linux cluster will receive one complete frame in turn, and it has been found that parallel writing to one file is not as efficient as writing to multiple files. This will be addressed by adding Virtual DataSet (VDS) support to the HDF5 library. DLS, DESY and the Percival collaboration have jointly contracted the HDF5 group for this feature also. The HDF5 VDS will present data stored in several HDF5 datasets and files as a single HDF5 dataset and to access the data via HDF5 APIs without rewriting and rearranging the data [9].

CONCLUSION

The controls and data acquisition software for the Percival detector under development at DLS already has a design for the system architecture, and has achieved a number of significant milestones.

The Frame Receiver software can successfully receive data frames over a network switch, with a basic prototype working with the current mezzanine firmware. Live control and region-of-interest support is being developed.

Components developed at DLS for the live streaming and live viewing of processed data are ready to be used within the control framework for the Percival detector.

With the direct chunk write and SWMR feature, tests have shown acceptable performance writing to a parallelised file system.

A prototype of the scripting and CLI for detector control has been tested operating with emulated hardware.

A library written in C containing the live processing algorithms has been developed and extensively optimised to not only meet but substantially exceed the bandwidth requirements for processing data on each cluster node. Testing has shown it is capable of processing at a rate of (2.5 ± 10%) GiB/s on one of the Linux cluster nodes.

An interface for EPICS and areaDetector will be included in the developed controls software, and will be allow use with other control systems such as TANGO and DOOCS. The SDK also allows for the implementation of a GUI with the controls software.

REFERENCES

[1] Experimental Physics and Industrial Control System website: `http://www.aps.anl.gov/epics/`

[2] areaDetector website: `http://cars9.uchicago.edu/software/epics/areaDetector.html`, "areaDetector: EPICS software for area detectors", (22 September 2015).

[3] U.K. Pedersen et al., "Feasibility Study of PERCIVAL Data Acquisition Backend Architecture", IEEE NUCLEAR SCIENCE SYMPOSIUM & MEDICAL IMAGING CONFERENCE, (2014).

[4] C.B. Wunderer et al., "The PERCIVAL soft X-ray imager", 16TH INTERNATIONAL WORKSHOP ON RADIATION IMAGING DETECTORS, Trieste, Italy (2014).

[5] B. Marsh et al., "PERCIVAL: Design and Characterisation of a CMOS Image Sensor for Direct Detection of Low-Energy X-Rays", PSD10: 10th International Conference on Position Sensitive Detectors, University of Surrey, UK (2014).

[6] Q, Gu, "High Performance Detector Software for PERCIVAL Detector", *Diamond Light Source Summer Internship Program Report*, Unpublished, (2015).

[7] Blosc website: `http://www.blosc.org`, "Blosc, an extremely fast, multi-threaded, meta-compressor library", (22 September 2015).

[8] The HDF Group website, Single-Writer/Multiple-Reader (SWMR) Documentation: `https://www.hdfgroup.org/HDF5/docNewFeatures/NewFeaturesSwmrDocs.html`, (22 September 2015).

[9] The HDF Group website, Virtual DataSet Documentation: `http://www.bigdata.org/HDF5/docNewFeatures/NewFeaturesVirtualDatasetDocs.html`, (22 September 2015).

[1] contracted by DLS and Dectris

[2] funded by Dectris

ISBN 978-3-95450-148-9