# Feasibility Study of PERCIVAL Data Acquisition Backend Architecture

U. K. Pedersen, N. Tartoni, J. Marchal, J. Thompson, N. Rees, N. De Maio, A. Greer C. B. Wunderer, A. Marras M. Bayer, J. Correa, S. Lange, I. Shevyakov, S. Smoljanin, M. Viti, Q. Xia, M. Zimmer, H. Graafsma G. Cautero, D. Giuressi, R. Menk, L. Stebel, H. Yousef H. Yousef, T. Nicholls, R. Turchetta, I. Sedgwick, D. Das, B. Marsh

***Abstract*—PERCIVAL soft-X-ray (250 eV 1 keV) image detector project is a collaboration between DESY, STFC, ELETTRA and DIAMOND LIGHT SOURCE. The objective of the project is to develop a back-thinned CMOS detector which outperforms present soft-X ray image detector technology, in terms of sensor size, noise, dynamic range and frame rate. The size of this 13M pixel imager associated with its 120 frames per second frame rate impose very challenging requirements to the Data Acquisition Backend of the system. A DAQ backend system architecture, using a commercial deep-buffer switch to rearrange image data streams coming from different regions of the sensor via several 10Gbps Ethernet links has been proposed to reassemble image frames. Real-time data processing is to be performed on multiple, parallel commodity compute nodes. This contribution to the conference reports on benchmarking tests performed as a feasibility study, and presents the resulting recommendations for the system architecture of the PERCIVAL detector DAQ backend. The feasibility study covered three key issues: Reliably moving data in UDP packets from multiple 10Gbps Ethernet links from the DAQ front-end electronics to X86 based commodity compute nodes; Real-time processing on the compute nodes; and finally streaming data to a central parallel storage system.**

## I. Introduction

THE Percival DAQ backend feasibility study was carried out by members of the Diamond Light Source Beamline Controls group from mid-2013 to early 2014. This feasiblity study was focussed on the use of commodity X86 compute nodes for real-time collection, processing and storing of image data from the Percival detector. This paper summarise the study and its reported recommendations and conclusions for the continuing design and implementation of the Percival DAQ backend software and hardware.

### A. The Percival Detector

Figure 1 show the outline architecture of the Percival system at the time of carrying out the study (chip pixel count are subject to change).

U. K. Pedersen, N. Tartoni, J. Marchal, J. Thompson, N. Rees, N. De Maio A. Greer are with Diamond Light Source, Harwell Science and Innovation Campus, Oxfordshire, United Kingdom

C. B. Wunderer, A. Marras M. Bayer, J. Correa, S. Lange, I. Shevyakov, S. Smoljanin, M. Viti, Q. Xia, M. Zimmer H. Graafsma are with Deutsches Elektronen-Synchrotron (DESY) in Hamburg, Germany

G. Cautero, D. Giuressi, R. Menk, L. Stebel H. Yousef are with Elettra Sincrotrone Trieste, Italy

T. Nicholls, R. Turchetta, I. Sedgwick, D. Das B. Marsh are with the Science and Technology Facilities Council (STFC) at the Rutherford Appleton Laboratory, Harwell Science and Innovation Campus, Oxfordshire, United Kingdom

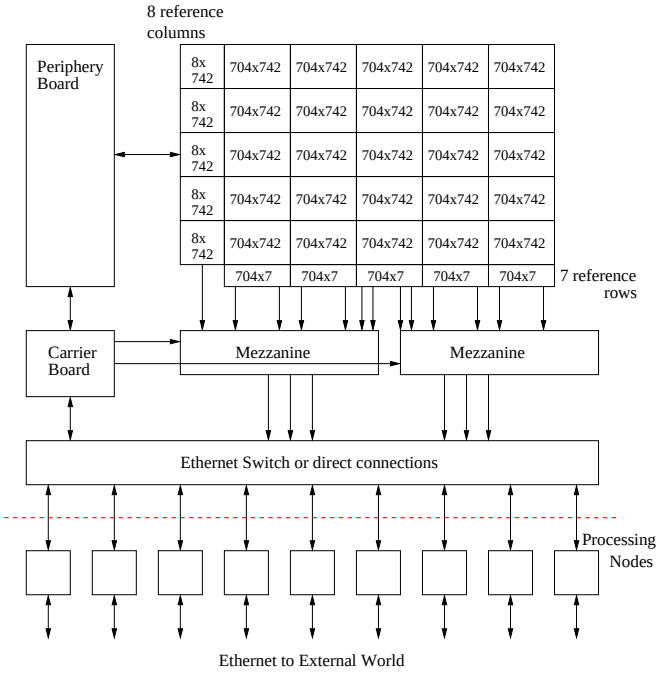


Fig. 1. Percival system overview.

A pixel array feeds scrambled data to a pair of "mezzanine" cards, each of which carry an FPGA that performs bit-descrambling and formats the data into UDP packets for transmission to the processing nodes. The processing nodes are commodity X86 servers and it is the processing on these nodes which is the focus of this study.

The frame rate is expected to be 120 Hz. Each pixel is represented in a 16 bit word. For the ~13Mega-pixel large detector this yields data-rate of ~3GB/s. However, as the Percival detector also transmits a reset frame along with each data frame, the overall data-rate is doubled to ~6GB/s. This data-stream will be transmitted from the mezzanine FPGAs via two sets of three 10Gbps Ethernet fibre links.

The data frames are sent from the Mezzanine boards to the processing nodes in a round-robin fashion (i.e. "temporal mode") such that each processing node receive only every N'th full frame for processing and storing, where N equals the total number of processing nodes. Thus the expected frame-rate on the individual compute node is 15fps.

With an overall data rate of 6GB/s, continuously streaming through 8 parallel compute nodes, an external large scale

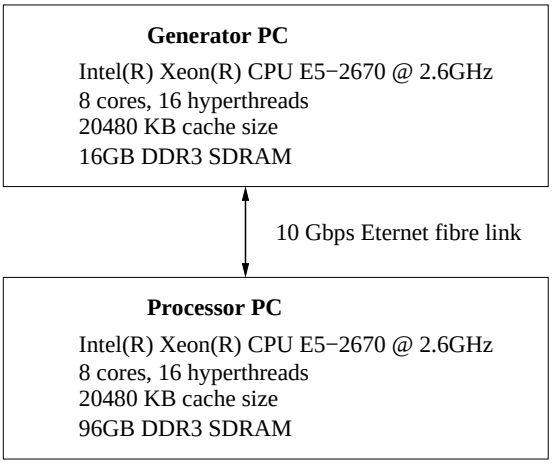


Fig. 2. Emulation system overview.

parallel file system is required to capture and store the data to disk. The choice of file system technology is beyond the scope of the Percival project and this particular study, but a very important aspect to be considered separately.

### B. Purpose of the study

The feasibility study was carried out as a risk-mitigation exercise, prior to starting up a major development project with an extensive risk-register. The purpose was to gain some knowledge and experience in transferring, processing and storing data at the rate of the Percival detector system. The continuous 6GB/s data rate is a challenge even on modern compute platforms.

The identified issues to study included:

- UDP transfer of high-rate image data resembling the Percival data stream
  - Particularly considering packet loss at the receiving end
- Real time processing of the data stream
  - Investigate the concern of the CPU load when processing the high rate data stream
  - Identify the required compute resources

The outcome of this study was a report summarising the investigations, analysis of the problem and recommendations regarding the required compute resources and hints for optimal software design.

## II. Hardware Emulation

AS the electronics hardware for the detector system is not yet available, the feasibility study software was designed to emulate the back-end electronics, the “Mezzanine Boards”. These boards transmit image frames over 10Gbps Ethernet in a UDP packet stream.

A second software application was implemented to receive the transmitted packets, detect any dropped packets and perform a basic version of the required processing.

The hardware configuration for this study is illustrated in Figure 2. The emulation system consist of two X86 based servers, connected back-to-back with a 10Gbps Ethernet link. One server runs the frame generator and transmitter - and the other one the frame receiver and processing. The investigations and tests were carried out on the receiver.

## III. Profiling

PROFILING exercises of the packet receiving and processing application were carried out in several stages with different focus.

Initially a series of experiment were carried out with the data transfer in “spatial mode” - where as opposed to the “temporal mode” only a subset of a full frame (i.e. a stripe) is transmitted to each compute node for processing. However, as the Percival detector frames contain calibration data around the edges it is clear that each processing step is most efficiently carried out by a compute node if the entire full-frame dataset is received and available on a given node. In the “temporal mode” each compute node can operate entirely parallel and independent of each other which simplifies the application greatly.

There are no requirements for processing involving sequences of frames at this point. Thus the main parts of the profiling focused on the “temporal mode” where full frames will be transmitted in a round-robin fashion to a cluster of processing compute nodes.

### A. Dropping UDP Packets

The initial tests indicated a substantial issue with packet loss where up to 10 detector image frames in an transmission of 100,000 would be detected as corrupt due to missing data. The data loss was tracked down to dropped UDP packages at the receiving end of the transmission. There were no evidence of data loss or corruption on smaller scales like byte-loss or bit-level corruption.

The following points have been identified as effecting the level of packet loss, and improvements implemented and tested in the software which was developed throughout this study:

- **Kernel network buffers:** By default the buffers used by the Linux kernel for incoming network traffic are relatively small. When network traffic is high these buffers quickly overflow, resulting in packet loss. Improvements were noticed when increasing the network buffers to 64MB, which is large enough to contain a full frame, including the reset-frame.
- **Processor input queue:** This parameter is also relatively small by default. Packets arriving after the queue is full are discarded before they can be processed. Increasing the queue length to 250,000 prevent the overflow.
- **Thread CPU scheduling:** On occasion, the Linux kernel decides to move a running thread from one CPU core to another. In some circumstances the overhead of moving threads leads to a loss of UDP packets. The recommendation is to pin the UDP receiving threads or processes to specific CPU cores to prevent this scheduling.
- **Data transmitter timing characteristics:** The initial transmit emulator would burst out all packets for an individual image frame (about 8K packets per frame) in a tight loop; then a gap before transmitting the next frame.

TABLE I
SINGLE CHANNEL PROCESSING TIMES

| Processing Mode | Raw | Reordering | Descramble |
|---|---|---|---|
| Image Frame | 92ms | 135ms | 92ms |
| Reset Frame | 37ms | 30ms | 37ms |
| Packet Service Thread | 51ms | 51ms | 51ms |

TABLE II
4 CHANNELS PROCESSING TIMES

| Processing Mode | Raw | Reordering | Descramble |
|---|---|---|---|
| Image Frame | 93-95ms | 130-136ms | 360-430ms |
| Reset Frame | 28-31ms | 29-33ms | 24-28ms |
| Packet Service Thread | 135ms | 135-138ms | 135-137ms |

This timing characteristic would result in very high levels of packet loss. Spreading the transmission of individual UDP packets within a full frame, while still retaining the overall frame-rate is proving more reliable and causing fewer packet loss overall.

- **Network card:** Two models of network cards were used in these tests from Mellanox and Broadcom. In this particular use-case the better performance in terms of packet loss was with the Broadcom card where packet loss was around 1 in 100,000. With the Mellanox card the loss was recorded at around 1 in 10,000.

### B. Processing Load

The study investigated the capability of commodity X86 servers to perform the required processing at the rate of the Percival data stream.

The data receiver application also include the data processing steps. All tests were carried out on image frames of size 4096x4096 which is slightly larger than the real 13M detector frames.

The required processing steps with increasing complexity include:

- **Raw:** receiving the UDP packet stream and generating image frames
- **Reordering:** reconstruct the correct sequence of pixels within the frame
- **Descrambling:** decoding gain, fine and coarse ADC bits and reset frame subtraction

The algorithms used in the feasibility study have only been through a single iteration of optimisation; namely a relatively simple re-ordering of if statements and loops to be more efficient. Even this simple optimisation yielded a performance increase of over 40%. Further efforts in this domain will be pursued to implement efficient algorithms.

Each processing step for the image frame and reset frame ran in a separate thread. A packet service thread present a certain overhead for receiving frames and servicing the processing threads. Table I lists the measurements of each processing step and the service overhead.

The processing was then scaled up by 4 to run the same steps in 4 parallel instances, still on the same compute node. Table II lists the range of measurements for each step.

## IV. CONCLUSIONS

THE feasibility study report has issued a number of conclusions and recommendations for the further work of designing and implementing the Percival DAQ backend software architecture.

### A. UDP Packet Retransmit Protocol

A loss of even a few packages in a transmission of 100,000 frames is unacceptable. The goal is to achieve 1,000,000 frames without any corruption. To meet this goal the study report recommended to add a small-scale re-transmission protocol to the existing data transfer protocol. The use of a more advanced protocol like TCP/IP would also improve the situation - but this is undesired due to the extra complexity and FPGA resources required on the Mezzanine board.

More recently a basic re-transmit protocol has been designed to allow re-sending dropped packets if the data is still available in the memory of the Mezzanine board.

### B. Processing Performance

The following conclusions were made based on the profiling exercises:

- The bottleneck is not in the network transfer rate, but rather in the time required to process the incoming frames.
- A processing data-rate equivalent to a 10gbps network link is reached when 8 receiver and processing processes run in parallel.
- Pinning threads with heavy processing load to dedicated CPUs is essential for reliable performance.
- The Broadcom 10gpbs network card performed better than the Mellanox card with respect to packet loss.
- There is no evidence of data loss or corruption on scales smaller than a complete UDP packet.

Given the profiling results on a smaller scale of compute nodes and CPU cores in this study; the recommendation is that a minimum of 60 cores are required to perform the processing - and a similar number of cores required in addition for other work like packet receiving, file writing, etc.

The recommendation is to scale the system to run across 8 X86 servers with 32 cores. Each server will require a 10gbps Ethernet link to receive frames and another 10gbps link to transport images off the system or write to an external filesystem. This recommendation is intended to provide a reasonable processing resource to cover additional computation requirements which have not yet been identified.

Further work must be undertaken to optimise the algorithm implementations. The study recommends further investigations into loop unrolling and other techniques to take advantage of the modern CPUs parallel capabilities together with compiler optimisations.