PERCIVAL CONTROL AND DAQ SOFTWARE: status & implementation

A.S. Palaha, C. Angelsen, Q. Gu, J. Marchal, U.K. Pedersen, N. Rees, N. Tartoni, H. Yousef (Diamond Light Source)

M. Bayer, J. Correa, P. Gottlicher, H. Graafsma, S. Lange, A. Marras, S. Reza, I. Shevyakov, S. Smoljanin, J. Supra, M. Tennert, C.B. Wunderer, Q. Xia, M. Zimmer (DESY)

D. Das, N. Guerrini, B. Marsh, T. Nicholls, I. Sedgwick, T. Turchetta (STFC)

G. Cautero, D. Giuressi, A. Khromova, R. Menk, G. Pinaroli, L. Stebel (Elettra Sincrotrone Trieste)

H.J. Hyun, K.S. Kim, S. Rah (Pohang Accelerator Laboratory)

The Percival Detector Collaboration

The PERCIVAL (Pixelated, Energy Resolving CMOS Imager, Versatile And Large) detector development project is a collaboration between DESY, Elettra Sincrotrone Trieste, Pohang Accelerator Laboratory and Diamond Light Source (DLS).

Introduction

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. 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. For the designed frame-rate of **120 Hz** this leads to an overall data rate of **~6 GiB/s**.

Control System Architecture

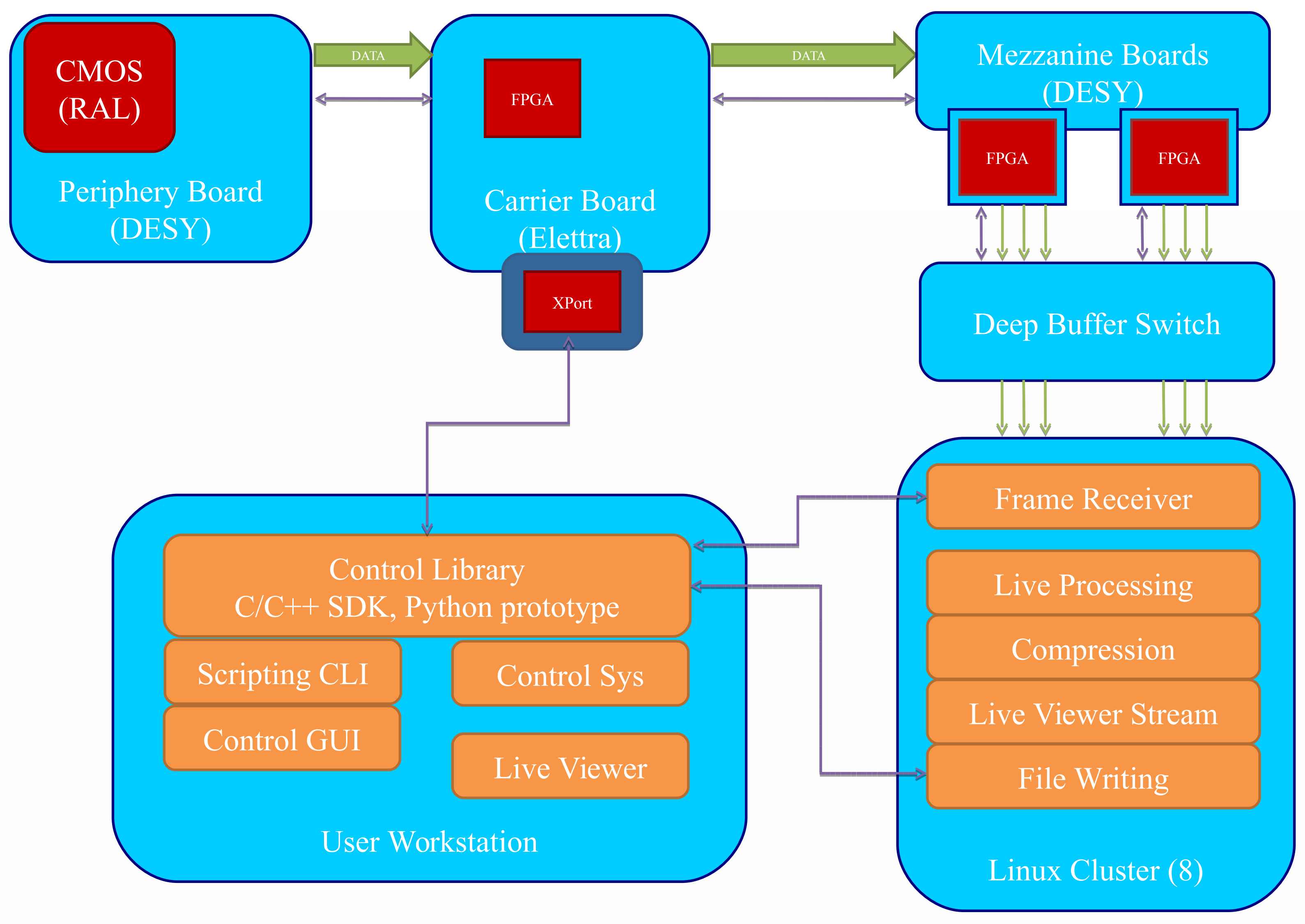

Fig 1. Percival Software Architecture: purple arrows represent control links, green arrows represent data links

The proposed system architecture shown in Figure 1 can be split into two parts: the DAQ software of the top half shown in red, and the control software of the bottom half shown in orange.

DAQ system:

Detector data is collected from the CMOS chip by the carrier board, transferred to the mezzanine boards which sends the data through 10 Gbps Ethernet links to the deep buffer switch and onto the Linux cluster nodes.

Control Software:

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.

The Linux cluster software performs all of the live processing of the incoming data collected by the Frame Receiver, providing live streaming and viewing of the data that can be monitored in the user workstation. Compression optimised for multi-core architectures and fast performance will be used to reduce bandwidth. Complete frames will be stored on a parallelised file system with consecutive frames on consecutive drives. A virtual dataset will present this data spread over several HDF5 files as one single HDF5 dataset without rearranging or rewriting the data.

Live Processing Optimisation

The optimisation of the live processing C++ library was carried out by Qiushi Gu, a summer student at Diamond Light Source in 2015. Multiple methods were used to optimise the processing bandwidth, including:

- **reducing computation time of calculations** that use floating points by optimising the formulas,
- **optimisation options** from the GCC compiler and C++ language,
- **parellelisation** using Intel Threading Building Blocks (TBB),
- **vectorising operations** to process multiple pixels per calculation using Advanced Vector eXtensions (AVX),
- **alignment of data** with the available vector sizes and with processor memory caches.

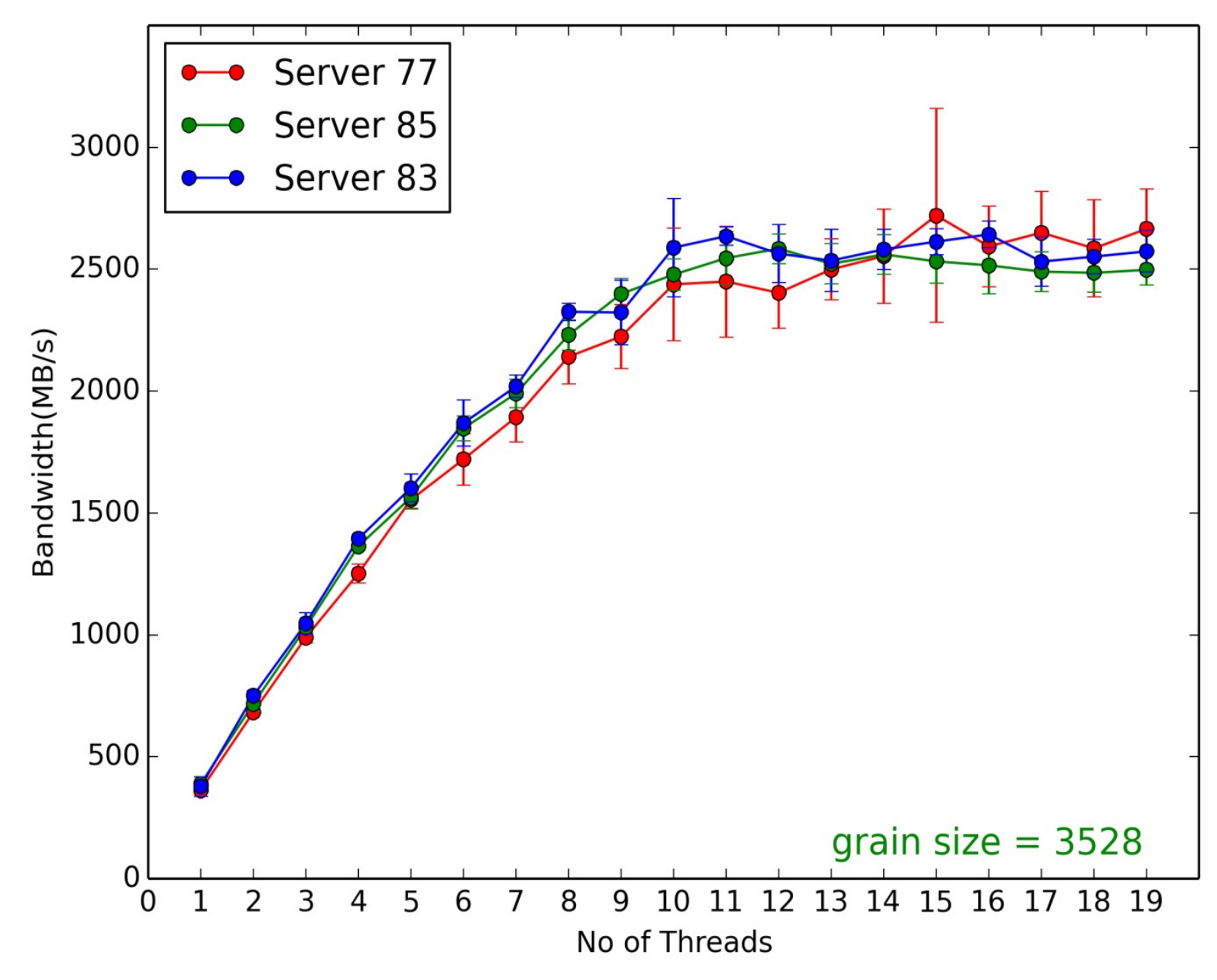

Fig 2. Processing bandwidth versus the number of threads

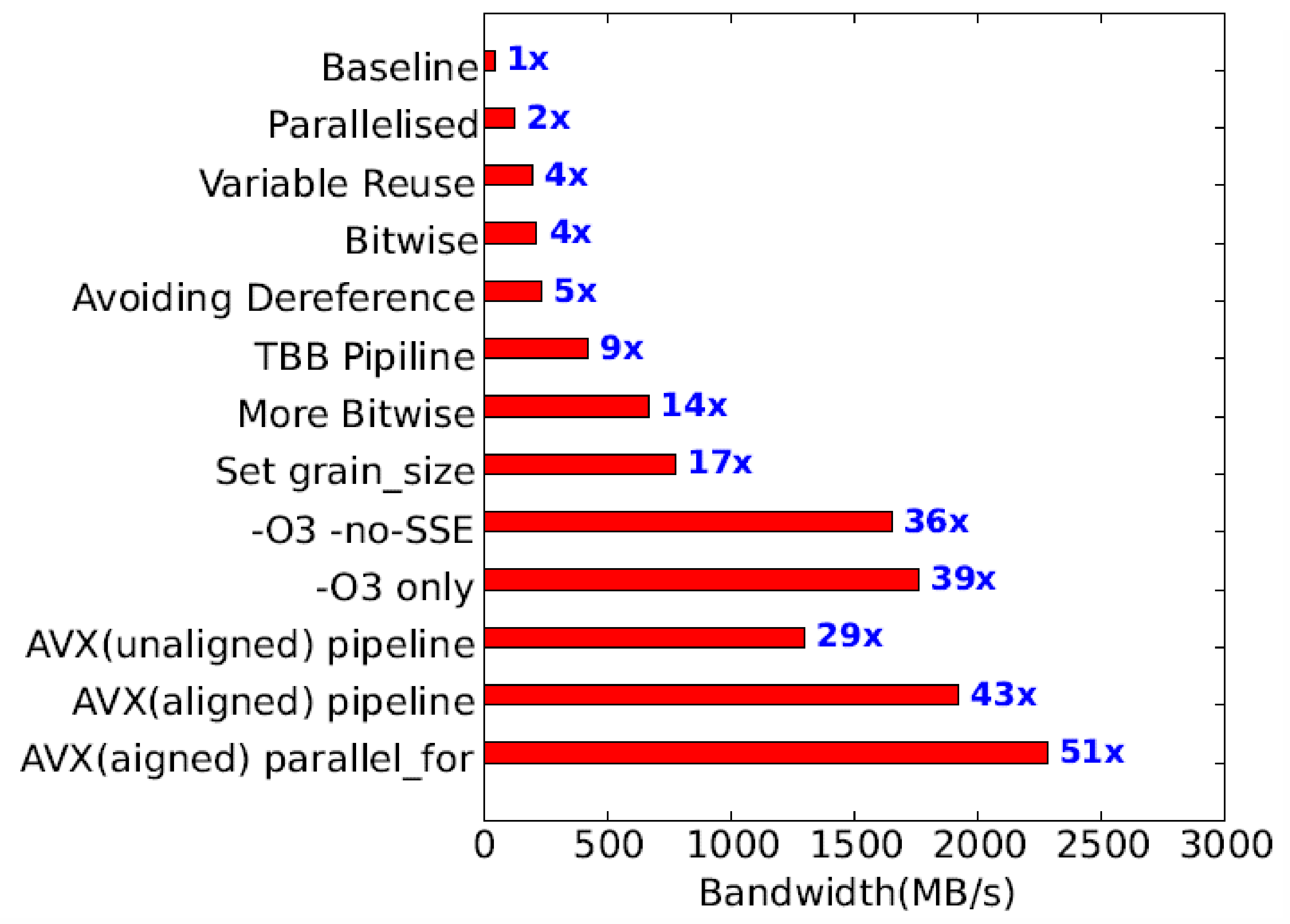

Fig 3. Optimisation steps versus processing bandwidth, with comparisons to the baseline

Conclusions

- DLS has developed a design for the system architecture
- The Frame Receiver software prototype works successfully with the current mezzanine board firmware
- Components developed at DLS are ready to be used for live streaming and viewing of data
- Writing data to a parallelised file system has proven successful using direct chunk write and the single-write multiple-read (SWMR) feature
- A prototype of the scripting and command line interface (CLI) for detector control has been tested with emulated hardware
- The live processing library written in C has been highly optimised to substantially surpass the bandwidth requirements for data processing on each cluster node!
  - Testing shows one cluster node using the library can process data at a rate of **(2.5 ± 10%) GB/s, more than 3 times required!**
- An interface for EPICS and areaDetector will be included with the controls software, and will allow use of other controls systems such as TANGO and DOOCS
- The SDK allows implementation of a GUI with the controls software

For more information please visit www.diamond.ac.uk or contact Arvinder Palaha at Arvinder.Palaha@diamond.ac.uk

diamond