Odin - A Control and Data Acquisition Framework for Excalibur 1M and 3M Detectors

G. Yendell, U. Pedersen, N. Tartoni, S, Williams Diamond Light Source Ltd, Oxfordshire, UK
T. Nicholls, STFC/RAL, Chilton, Didcot, Oxon, UK
Alan Greer, OSL, Cambridge, UK

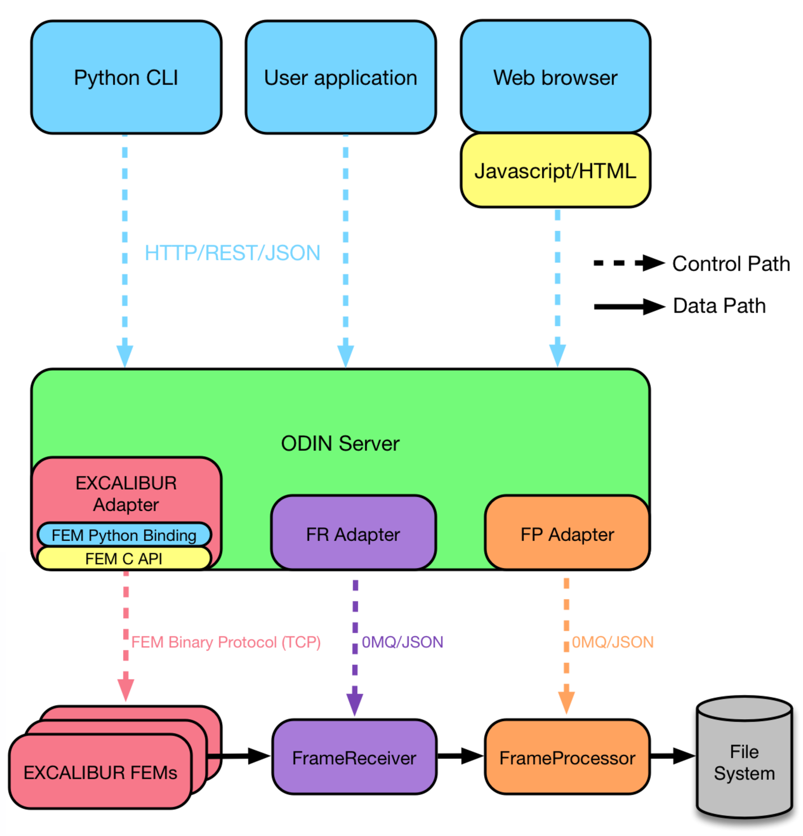
Diamond Light Source (DLS) are currently developing data acquisition and control software for several modular, high-performance detectors. Excalibur [1] is the result of a collaboration between DLS and STFC and has been implemented for the X-ray Imaging and Coherence beamline I13 to make use of the small pixel size in coherence diffraction imaging. The Hard X-ray Nanoprobe beamline I14 has more recently chosen a 3M Excalibur system for nanoscale microscopy. Another collaboration, between DLS, Elettra, the Pohang Light Source and STFC, is ongoing to develop the Percival detector [2] for soft x-ray experiments. At the same time, DLS is exploring commercial options in the Eiger from Dectris. Currently, the VMXi (Versatile Macromolecular Crystallography in-situ) beamline is commissioning a 4M Eiger X detector [3]. With a multitude of modular and scalable detector systems in development concurrently, an opportunity arose to develop shared control and data acquisition software stacks to drive the systems, designed from the very beginning to be detector agnostic, but with a set of specific use cases to guide the design process.

Odin

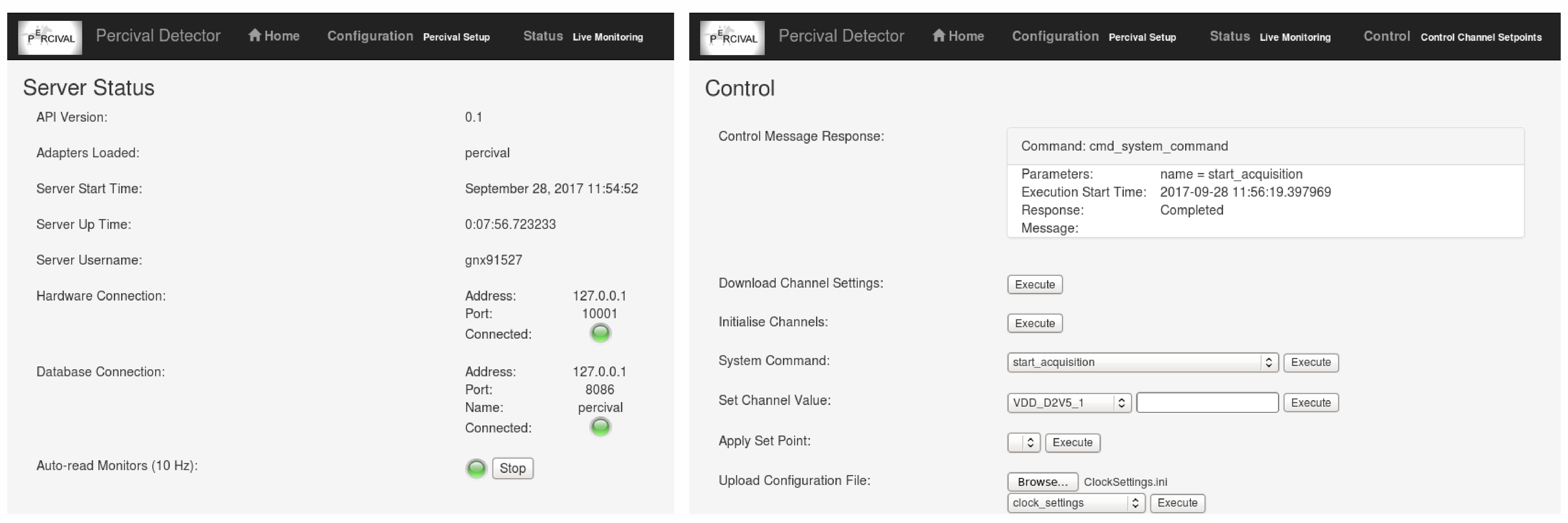
Devices consisting of multiple individual parts can lead to complications in the control layer trying to get them to operate together in unity. The Odin software framework is designed specifically for this modular architecture by mirroring the structure within its internal processes. The data acquisition modules have the perspective of being one of many nodes built into the core of their logic. This makes it straightforward to operate multiple file writers on different server nodes working together to write a single acquisition to disk, all managed by a single point of control. Given the collaborative nature of the detector development, the software framework has been designed to generic, allowing its integration with control systems used at different sites.

OdinControl

OdinControl is a HTTP based server host providing a framework that device-specific adapters can be implemented for to interface to the control channel of a device. Adapters can be loaded in an Odin Server instance, which then provides a REST API corresponding to the attributes and methods in the adapter API. This can be extended to a RESTful client library that can be integrated into a higher level control system such as EPICS. OdinControl provides a simple Python API, enabling rapid development of device adapters. With OdinControl and a device adapter, a control system agnostic, consistent API is created that can be used in a wide range of applications. The generic architecture of OdinControl also means it does not need tight coupling to OdinData; it is also interfaced via adapters, just like Excalibur or Percival.



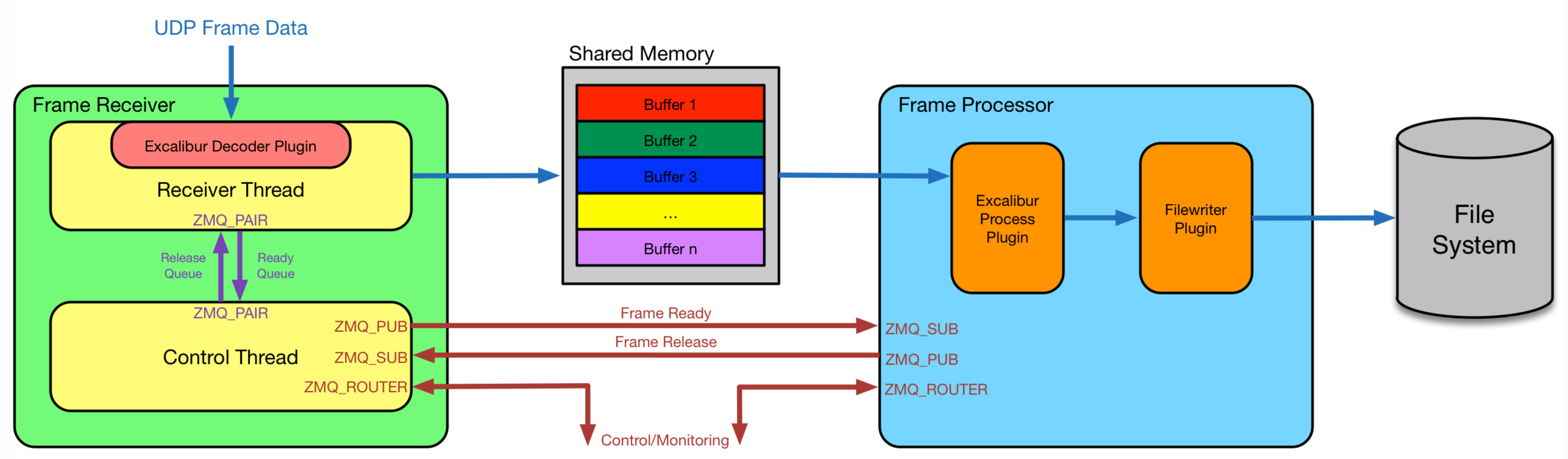
OdinControl architecture for Excalibur.



Example OdinControl client webpages for Percival.

OdinData

OdinData gathers incoming frames from a data stream and writes them to disk as quickly as possible. It has a modular architecture making it simple to extend its use for new detectors. The function of the software itself is relatively simple, allowing a higher-level supervisory control process to do the complex logic, exchanging simple configuration messages to perform specific operations. OdinData consists of two separate processes. These are the FrameReceiver (FR) and the FrameProcessor (FP). The FR can collect data packets on various input channel types, for example UDP and ZeroMQ [6], construct data frames and pass them on to the FP through a shared memory interface. The FP can then construct data chunks in the correct format and write them to disk.



OdinData architecture showing Excalibur use case.

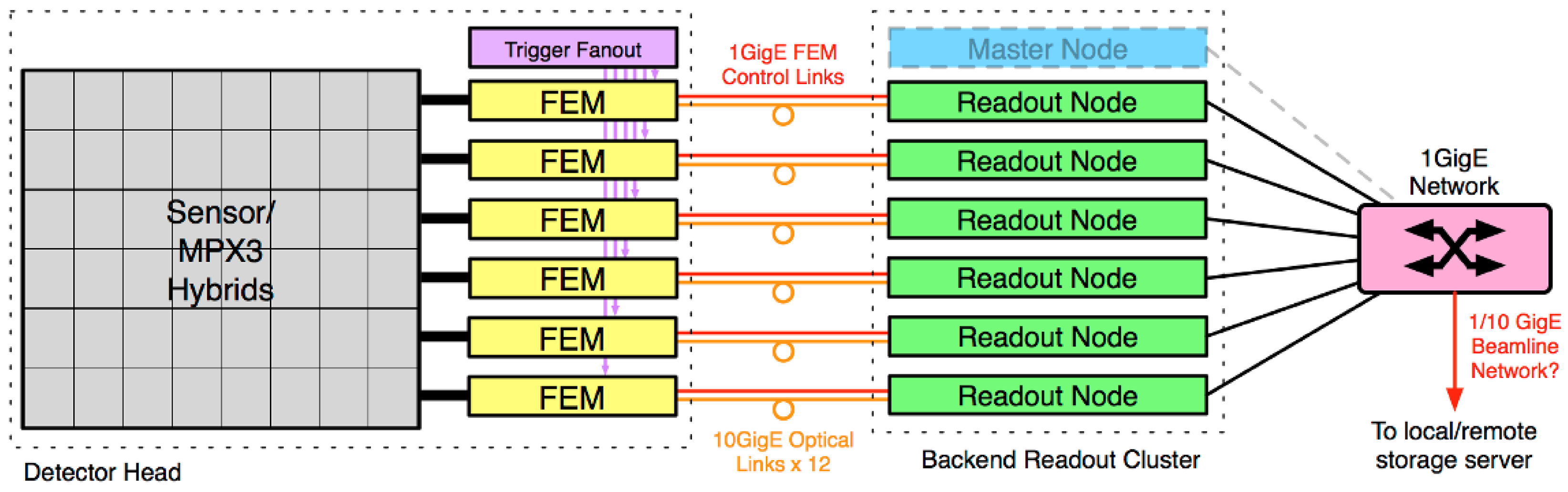
Excalibur has two plugins to provide the implementation of a decoder of the raw frame data as well as the processing required to define the data structure written to disk. The Excalibur use case is shown in the schematic above. The implementation of any other detector would simply require the two plugins to be replaced with equivalents, to process the output data stream.

OdinData employs some of the latest features of the HDF5 library. The use of Virtual Dataset (VDS) along with Single Writer Multiple Reader (SWMR) [7] functionality means that, as the parallel writers begin writing to each raw file, the data appears in single dataset as if the processes were all writing to the same file. This enables live processing to be carried out while frames are still being captured. Direct Chunk Write [7] is used to skip the HDF5 processing pipeline and write chunks straight to disk, reducing the processing required and limiting data copying. Direct Chunk Write allows OdinData to handle the pre-compressed frames from the Eiger. Reader applications can use Dynamically Loaded Filters [7] to read the datasets.

For more information please contact Gary Yendell at gary.yendell@diamond.ac.uk

Excalibur Detectors

Excalibur [1] detectors are made up of identical sensor 'stripes', with 8 Medipix3 readout chips. Each stripe has its own FPGA data acquisition card, known as a front-end module (FEM), with a 10Gbit/s optical link. These stripes are combined into a pair to create what is called a 'module'; a 1M 2048 x 512 pixel sensor. A 3M simply consists of 3 stacked modules producing a 2048 x 1536 sensor. The Excalibur is currently operated at DLS with an EPICS areaDetector driver [4] controlling and acquiring data from each individual FEM, with a top level IOC presenting PVs wired through to the underlying processes. However, this system is limited both in its control flexibility and its data throughput and is intended to be replaced by the Odin software stack.

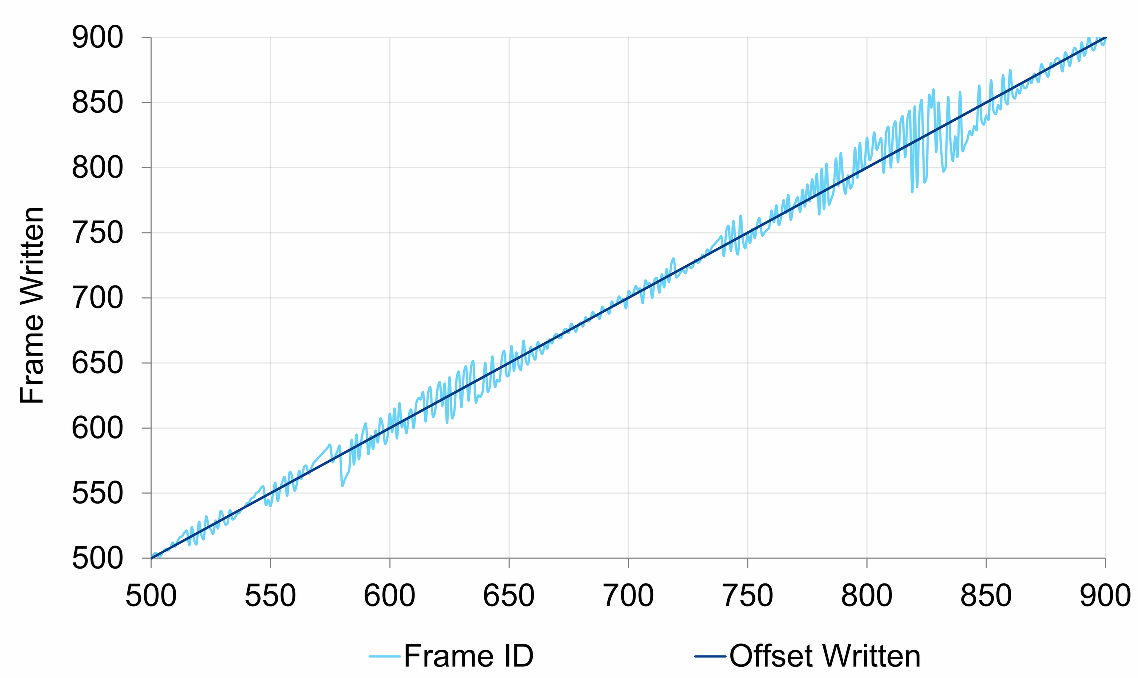


A schematic of the Excalibur 3M system [5].

Use Cases

Odin is now starting to be integrated with beamline and lab detector systems at DLS. Excalibur is the first detector to be fully supported by Odin. It has FR and FP plugins allowing an OdinData processes to serve each FEM of the detectors, as well as an adapter for OdinControl to provide a complete HTTP API for controlling the nodes. The latest milestone reached was using OdinData and OdinControl to acquire 10,000 frames from a 1M detector at 100Hz. The Surface and Interface Diffraction beamline I07 will be the first to commission an Excalibur operated with this software stack.

The first Eiger detector at DLS is currently being commissioned on the VMXi beamline using OdinData, alongside an EPICS areaDetector [8] driver used for control only. In this system, frames are fanned out systematically from the Eiger ZeroMQ [6] stream to four process nodes, each running an OdinData stack writing one quarter of the frames. These raw datasets are then wrapped in a VDS along with the meta data captured in parallel by a fifth, separate process. This produces a final HDF5 file containing a single, coherent view of all the data from an acquisition. It is able to fully support the 10 Gbit link of the 4M; the bottleneck being pushing frames down the data link from the detector, rather than writing them to disk.



A graph showing the order and offset of frames written during an Eiger acquisition. The Frame ID shows that frames are written out of order, because some nodes fall behind others, while the linear Offset Written proves that the frames are still written to the correct offsets in the HDF5 dataset.

Conclusions

Odin has the scope to become an integral part of beamline controls at DLS. It serves many currently missing use cases, such as those that involve very high data rates or require more flexible control. The usage to up to this point in time has been successful, but there is some way to go before it can be deployed more widely. The next steps are to improve stability and to develop some new features required for future use cases. One of these being the ability to rewind a scan and continue from part way through, which will require overwriting specific chunks of data through computation of relative offsets in the HDF5 datasets based on configuration parameters. This will eventually be integrated into Malcolm [9], another ongoing development at Diamond to allow concurrent control of multiple different systems, specifically for, but not limited to, hardware-triggered scans and Mapping [10] of samples.

References

[1] J. Marchal et al., “Excalibur: a small-pixel photon counting area detector for coherent x-ray diffraction - front-end design, fabrication and characterisation,” in Journal Of Physics: Conference Series, vol. 425, pp. 530–533, 2013.
[2] C. B. Wunderer et al., “The percival soft x-ray imager,” in JINST, vol. 9, 2014.
[3] EIGER X 4M, https://www.dectris.com/products/ eiger/eiger-x-for-synchrotron/details/ eiger-x-4m/.
[4] J. Thompson et al., “Controlling the excalibur detector,” in Proc. ICALEPCS’11, pp. 894–897.
[5] N. Tartoni et al., “Excalibur: A three million pixels photon counting area detector for coherent diffraction imaging based on the medipix3 asic,” in IEEE Nuclear Science Symposium Conference Record, 2012.
[6] ZeroMQ, zeromq.org/.
[7] N. Rees et al., “Developing hdf5 for the synchrotron community,” in Proc. ICALEPCS’15, pp. 845–848.
[8] Seivam B. ADEiger, github.com/brunoseivam/ ADEiger/.
[9] T. Cobb, M. Basham, G. Knap, C. Mita, and G. Yendell, “Malcolm: A middlelayer framework for generic continuous scanning,” in Proc. ICALEPCS’17. paper TUPHA159, this conference.
[10] R. Walton et al., “Mapping developments at diamond,” in Proc. ICALEPCS’15, pp. 1111–1114.