EPICS V4 for Diamond Detector Data

Dave Hickin
Diamond Light Source
EPICS Developers Meeting
Saclay 21/10/2014

Objectives

- Lossless high-performance transfer of detector data and camera images including metadata
- Software infrastructure to support it
- Framework for high performance scientific data services
- Incremental development
  - Based on areaDetector

diamond

Why transfer detector data?

- Transferring data between platforms (Usually from Windows to Linux)
  - Cameras often have Windows only support
  - Better support for HPC in Linux, e.g. HP file system
  - Linux toolchain
  - Preference for Linux (open source, reliability etc.)
  - More expertise on Linux

- Distributing processing

diamond

Case Study – I12 Beamline

- I12 Beamline at Diamond
  - Joint Engineering, Environmental and Processing(JEEP)
  - Imaging, Tomography, X-ray diffraction, SAXS
- PCO detector, Windows-only support
- HDF5-writer, Lustre distributed files system
- ~90 10MB images per second
- 10 Gig Ethernet

diamond

Why use EPICS V4

- EPICS V4 adds structured data. Allows metadata to be kept with image data
  - Image dimensions
  - Colour attributes
  - Experiment and system metadata
  - Compression/encoding information
- Integrates with other EPICS developments, e.g. CSS.

diamond

areaDetector overview

- Provides a general-purpose interface for detectors and cameras in EPICS
- Easily extensible
- Supports wide variety of detectors and cameras
- High-performance
- Mechanism for device-independent real-time data analysis

diamond

areaDetector overview (cont)

- Camera drivers inherit from base class ADDriver
- Drivers produces NDArrays
- Run plugins

- Plugins inherit from NDPluginDriver
- Connect to asyn port on a driver
- Consume NDArrays

diamond

NDArray

- 1-d type array of numeric type
- Dimension information
- ID and time stamps
- Attributes

- Also has Array Pool (but not part of “data”)

diamond

NDArray

- Dimension
  - Converts 1-d array to N-d array
  - Describes how array is part of larger array
  - Has size, offset, reverse and binning fields
- ID and timestamps
  - Unique ID – integer unique to frame
  - Time stamp (from driver)
  - 2nd time stamp

diamond

NDArray

- Attributes
  - A linked list of heterogeneous type.
  - Each attribute has name, description, source, source type and value
  - Source type can be driver, parameter library, EPICS PV or user-defined function
  - 2 standard attributes colorMode and Bayer pattern. Color Mode turns 3-d array into 2-d colour image
  - Attributes can describe image, contain info such as camera parameters or current value of PV

diamond

areaDetector and EPICS V4

- areaDetector runs a plugin which is a pvAccess server
- Plugin translates NDArrays into EPICS V4 structured data (normative type). Closely maps to NDArray
- pvAccess used to transfer data
- V4 client implements ADDriver
- Translates V4 type back into NDArray
- Passes NDArrays to plugins
- Existing plugins run remotely

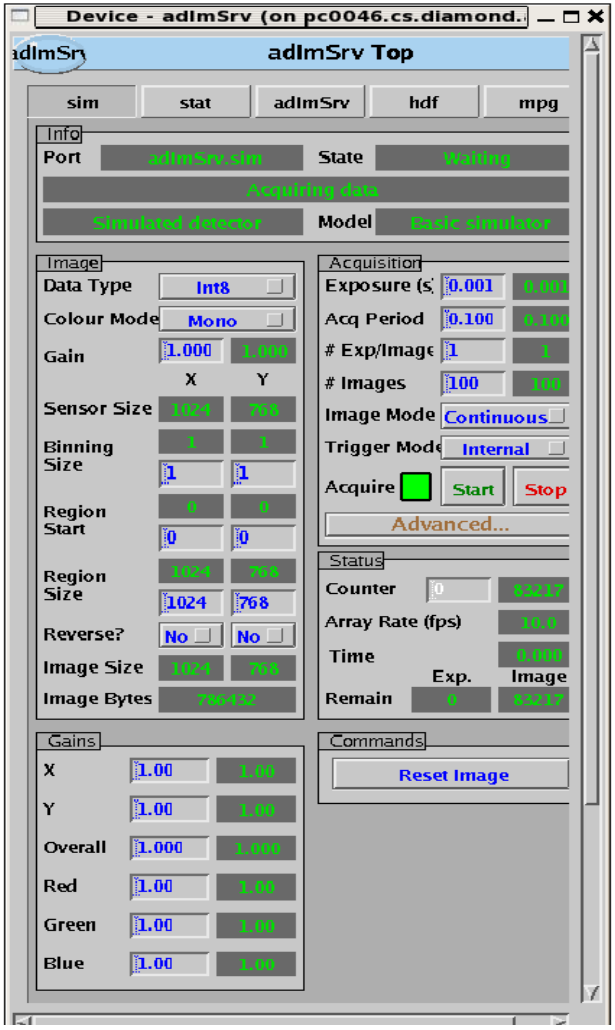
diamond

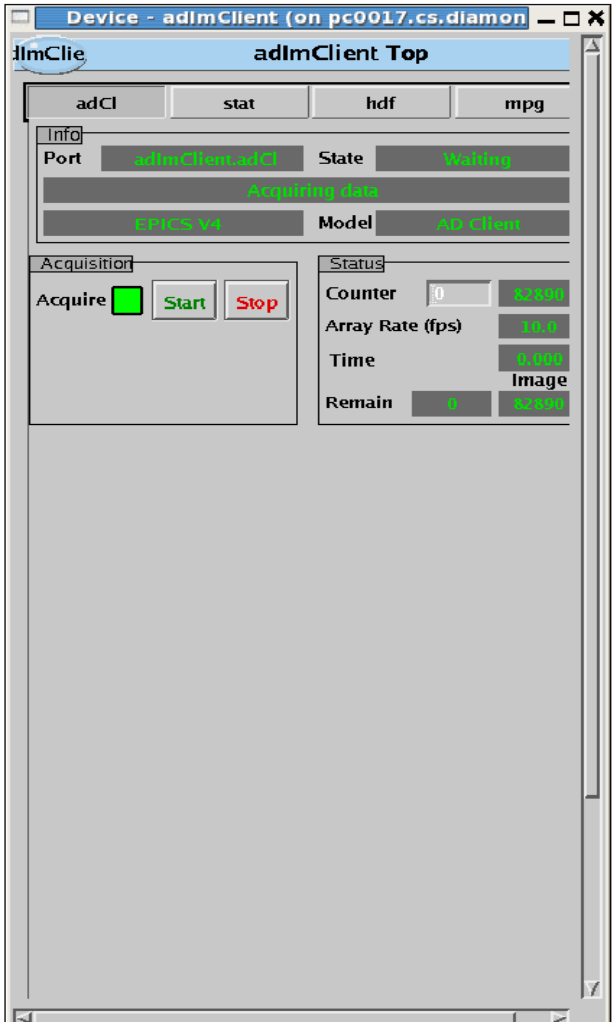
EPICS V4/AD Development

- EPICS V4 Initial Prototype
  - Server-side plugin and client-side driver
  - Uses pvAccess and EPICS V4 core code
  - Client monitors PV
  - Linux only. Not running on Windows yet
  - Initially coding based on pvAccess testServer
- Further development
  - Turn into robust solution
  - Integrate with other developments
  - Enhancements

diamond

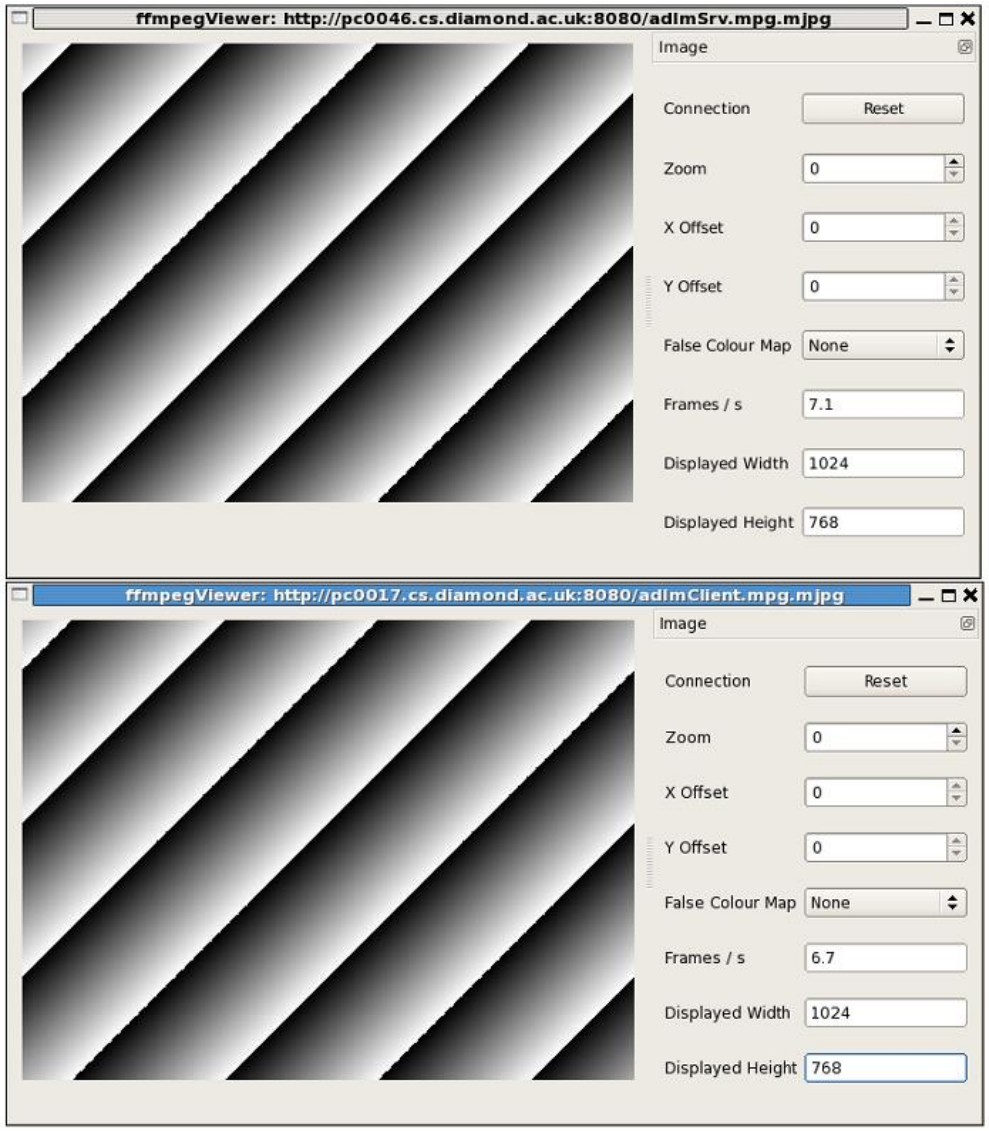
EPICS V4 Server and Client Prototype





diamond

Prototype: Images



diamond

Recent Work

- Increased performance. Tested on 10 Gig Ethernet
- Investigation and implementation of compression
- Server-side plugin uses pvDatabase instead of testServer and new client written
- Improved reliability. Helping to debug EPICS V4 core
- Defined new normative type (NTNDArray) to replace old type (NTImage)
- CSS integration and pvaSrv
- Prototype V4 SimDetector

diamond

Compression

- Compression algorithm applied to image data
- Add compression info to V4 structure
- Allows effective bandwidth to exceed physical
- Used freely available compression libraries.
  - An LZ4 library
  - Blosc (multi-threaded compression)
- Combined LZ4 algorithm with Blosc

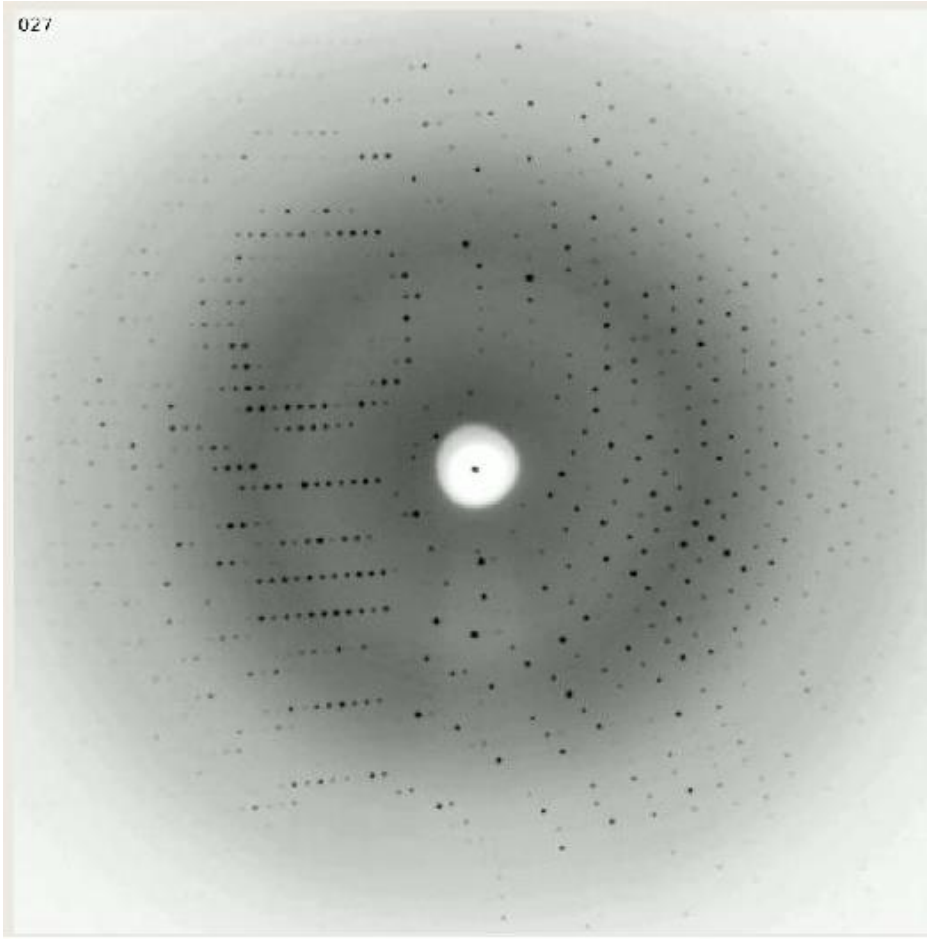
diamond

Performance on 10 Gig Ethernet

- Setup
  - 2 High Performance Linux PCs, connected by direct fibre 10 Gig Ethernet
  - SimDetector (modified) producing 3192x3192 images running V4 plugin on one machine
  - V4 client driver on other machine running stats hdf5 and mpeg plugins

diamond

simDetector Image



Performance on 10 Gig Ethernet

- Uncompressed
  - 120-122 frames per second (97-99+% bandwidth)
- Compressed
  - With compression image reduced to 36% of original size using lz4 and 38% by Blosc
  - Single threaded compression reduces performance
  - Blosc-based compression (multithreaded) increases rate
  - Blosc + lz4 best. Up to ~230 frames per second (190% of bandwidth)

diamond

NTNDArray

- EPICS V4 structure for detector data
- Normative type
- 1 NTNDArray gives 1 frame
- Maps closely to NDArray
- Dimension data, time stamps, uniqueId, attributes
- Adds codec information
- Uses unions for image data and attribute values
- Uses structure arrays for dimensions and attributes

diamond

V4 SimDetector

- Prototype V4 SimDetector
- Puts image direct into V4 structures instead of NDArray. Publishes image as PV
- V4 areaDetector client monitors and runs plugins
- Clients can run remotely or locally in the same or different process
- V4 can provide many of the functions provided by areaDetector (monitor queues, reference counted arrays)

diamond

Current and near future work

- Complete move to NTNDArray from NTImage
- Move to GitHub Module - ADPvAccess
- Package and release
- Integration with other EPICS developments especially CSS
- Windows build
- Deploy on beamline (I12 is candidate)

diamond

Possible future development of areaDetector/V4

1. Group images from multiple detectors into single image
2. Modify areaDetector drivers to put images directly into NTNDArrays instead and plugins consume these.
3. Other compression options
4. Alternatives: GStreamer

diamond