areaDetector: A module for EPICS area detector support

Mark Rivers

GeoSoilEnviroCARS, Advanced Photon Source

University of Chicago

areaDetector Talk Outline

- Motivation & goals for areaDetector module
- Overview of architecture
- Drivers for detectors & cameras
- Plugins for real-time processing
- Viewers and other clients
- Demo with simDetector and Prosilica camera

areaDetector - Motivation

- 2-D detectors are essential components of synchrotron beamlines
  - Sample viewing cameras, x-ray diffraction and scattering detectors, x-ray imaging, optical spectroscopy, etc.
- EPICS is a very commonly used control system on beamlines, (APS, DLS, SLS, SLAC, NSLS-II, Shanghai, etc.)
- Need to control the detectors from EPICS (useful even on non-EPICS beamlines, since other control systems like SPEC etc. can talk to EPICS)
- Previously several packages available, each typically restricted to a small set of detectors (Flea, Pilatus, marCCD, etc.)
- Clear advantages to an architecture that can be used on any detector, re-using many software components
- Providing EPICS control allows any higher-level client to control the detector and access the data (CSS, SPEC, medm, Python scripts, IDL programs, etc)

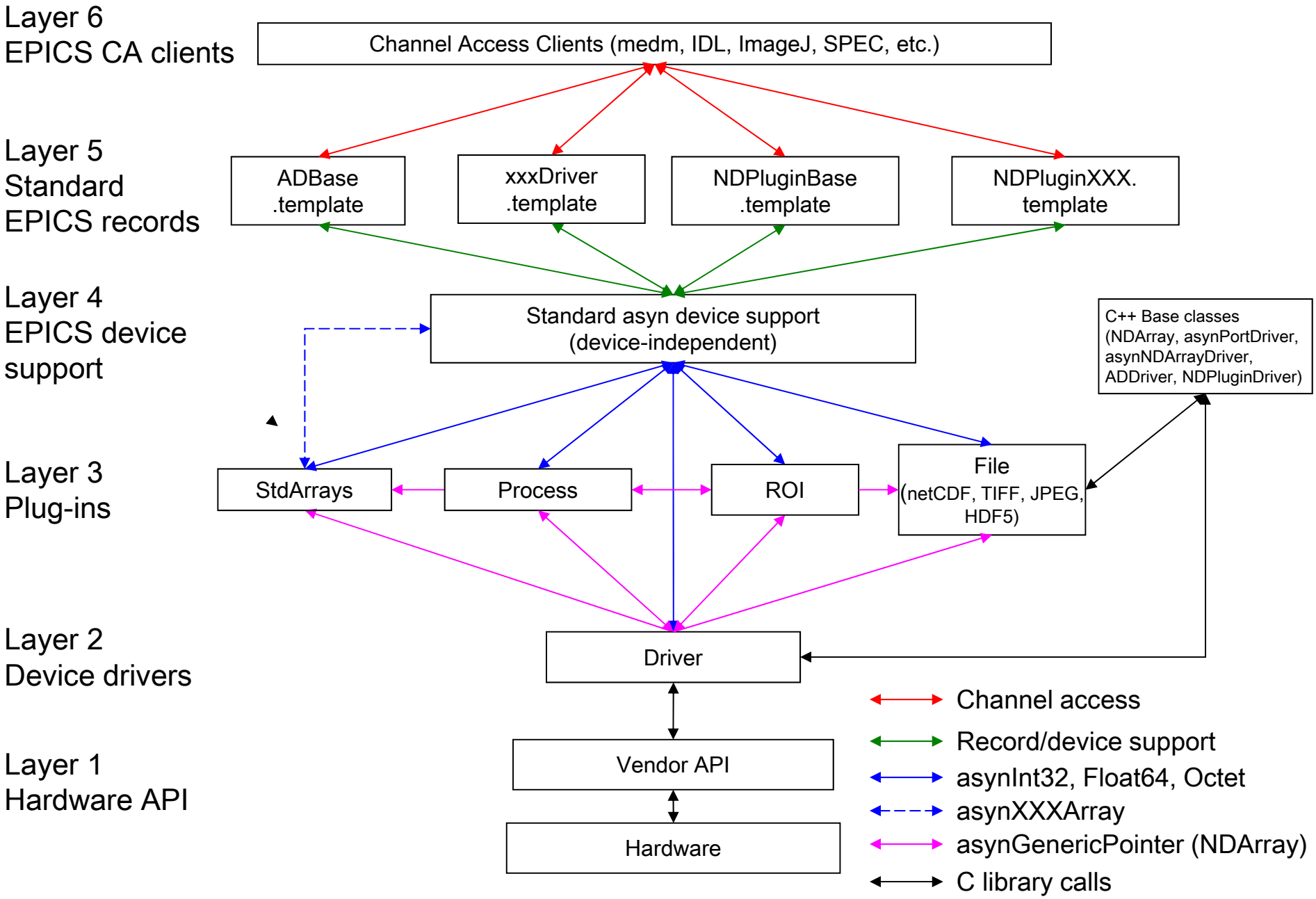
areaDetector - Goals

- Drivers for many detectors popular at synchrotron beamlines
  - Handle detectors ranging from >500 frames/second to <1 frame/second
- Basic parameters for all detectors
  - E.g. exposure time, start acquisition, etc.
  - Allows generic clients to be used for many applications
- Easy to implement new detector
  - Single device-driver C++ file to write. EPICS independent.
- Easy to implement detector-specific features
  - Driver understands additional parameters beyond those in the basic set
- EPICS-independent at lower layers.
- Middle-level plug-ins to add capability like regions-of-interest calculation, file saving, etc.
  - Device independent, work with all drivers
  - Below the EPICS layer for highest performance

areaDetector – Data structures

- NDArray
  - N-Dimensional array.
    - Everything is done in N-dimensions (up to 10), rather than 2. This is needed even for 2-D detectors to support color.
  - This is what plug-ins callbacks receive from device drivers.
- NDAttribute
  - Each NDArray has a list of associated attributes (metadata) that travel with the array through the processing pileline. Attributes can come from driver parameters, any EPICS PV, or any user-written function.
    - e.g. can store motor positions, temperature, ring current, etc. with each frame.
- NDArrayPool
  - Allocates NDArray objects from a freelist
  - Plugins access in readonly mode, increment reference count
  - Eliminates need to copy data when sending it to callbacks.

EPICS areaDetector Architecture



areaDetector – Data structures

Look at NDArray.h

Look at NDAttribute.h

Look at an XML attribute file

areaDetector Organization (R2-x and later)

areaDetector
Top-level module
RELEASE files, documentation, Makefile

ADCore	ADBinaries	ADProsilica	ADPilatus	…
Core module Base classes, plugins, simDetector, documentation	Binary libraries for Windows (HDF5, GraphicsMagick)	Prosilica driver	Pilatus driver	

- Each box above is a separate git repository
- Can be released independently
- Hosted at http://github.com/areaDetector project
- Each repository is a submodule under areaDetector/areaDetector

Source Code Organization on github

- https://github.com/areaDetector is top-level project
- Contains configure/ directory where paths and versions of supporting software are defined
- Contain .gitmodules to define submodules that will be cloned with git clone –recursive
- Contains documentation directory that builds and installs documentation
- Contains a top-level Makefile to build all or selected submodules

Detector drivers

- ADDriver (in ADCore)
  - Base C++ class from which detector drivers derive. Handles details of EPICS interfaces, and other common functions.
- Simulation driver (in ADCore)
  - Produces calculated images up to very high rates. Implements nearly all basic parameters, including color. Useful as a model for real detector drivers, and to test plugins and clients.
- Prosilica driver (ADProsilica)
  - Gigabit Ethernet cameras, mono and color
  - High resolution, high speed, e.g. 1360x1024 at 30 frames/second = 40MB/second.
- Firewire (IEEE-1396 DCAM) (ADFireWireWin, firewireDCAM)
  - Vendor-independent Firewire camera drivers for Linux and Windows
- Roper driver (ADRoper)
  - Princeton Instruments and Photometrics cameras controlled via WinView

Detector drivers (continued)

- PVCAM driver (ADPvCam)
  - Princeton Instruments and Photometrics cameras controlled via PVCAM library
- Pilatus driver (ADPilatus)
  - Pilatus pixel-array detectors.
- marCCD driver (ADmarCCD)
  - Rayonix (MAR-USA) CCD x-ray detectors
- ADSC driver (ADADSC)
  - ADSC CCD detectors
- mar345 driver (ADmar345)
  - marResearch mar345 online image plate
- Perkin-Elmer driver (ADPerkinElemer)
  - Perkin-Elmer amorphous silicon flat-panel detectors

Detector drivers (continued)

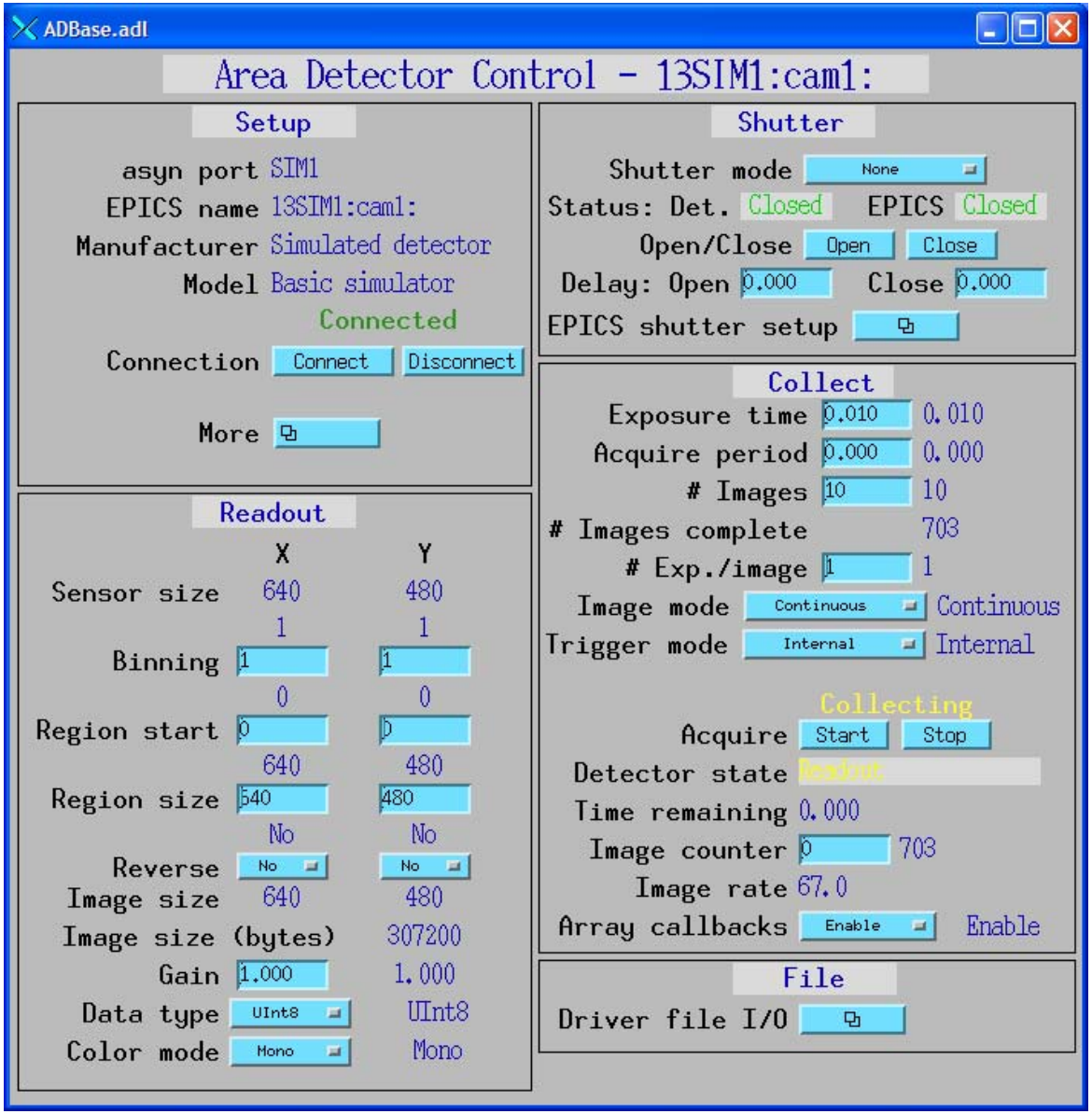
- Bruker driver (ADBruker)
  - Bruker detectors controlled via their Bruker Instrument Server (BIS)
- LightField driver (ADLightField)
  - Princeton Instruments detectors controlled via their LightField application using the Microsoft Common Language Runtime to automate it
- PSL driver (ADPSL)
  - Photonic Sciences Limited detectors
- URL driver (ADURL)
  - Driver to display images from any URL. Works with Web cameras, Axis video servers, static images, etc.
- Andor driver (ADAndor)
  - Driver for Andor CCD cameras
- Andor3 driver (ADAndor3)
  - Driver for Andor sCMOS cameras with V3 of their SDK

Detector drivers (continued)

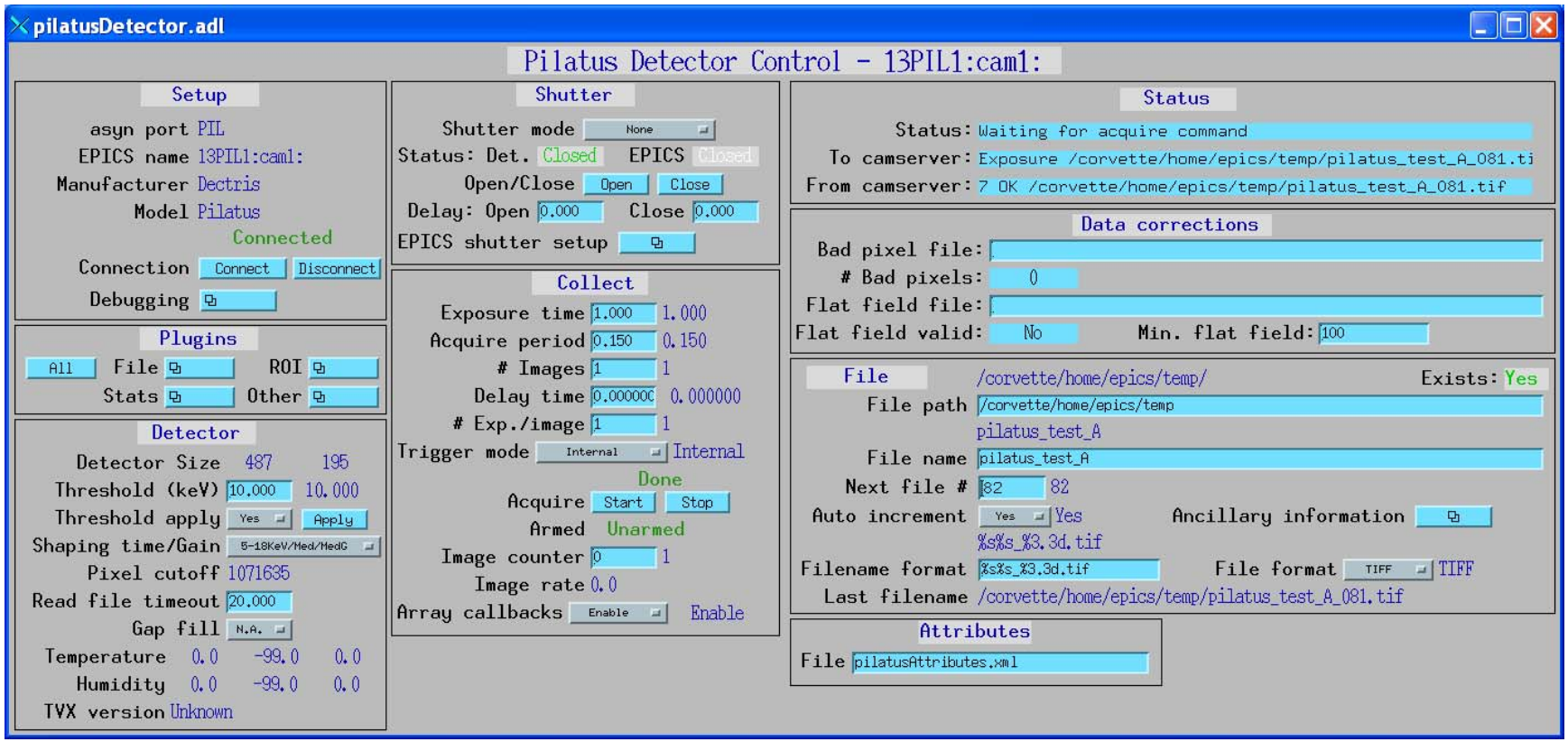
- Point Grey driver (ADPointGrey)
  - Driver for GigE, USB-3.0, USB-2.0, and Firewire cameras from Point Grey Research
- Pixirad driver (ADPixirad)
  - Driver for CdTe pixel-array detectors from Pixirad
- Generic GigE driver (aravisGigE)
  - Should work with any GigEVision compliant camera. From Tom Cobb at Diamond. Uses Aravis reverse-engineered GigEVision library
- PVAccess (EPICS V4) driver
  - Receives NTNDArrays over PVAccess
  - Allows plugins to run on other processes or machines from the areaDetector driver

ADBase.adl – Generic control screen

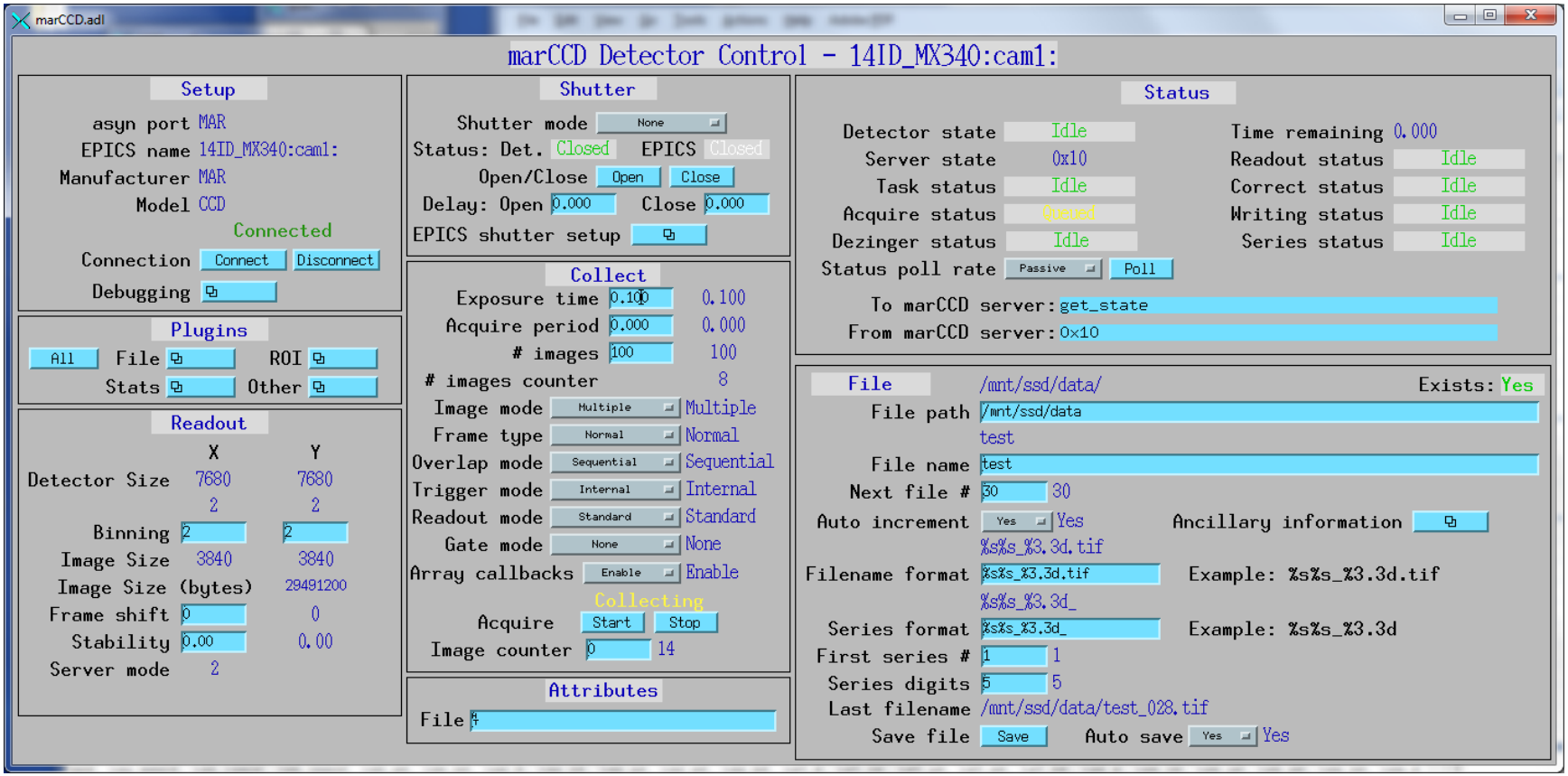
- Works with any detector
- Normally write custom control for each detector type to hide unimplemented features and expose driver-specific features



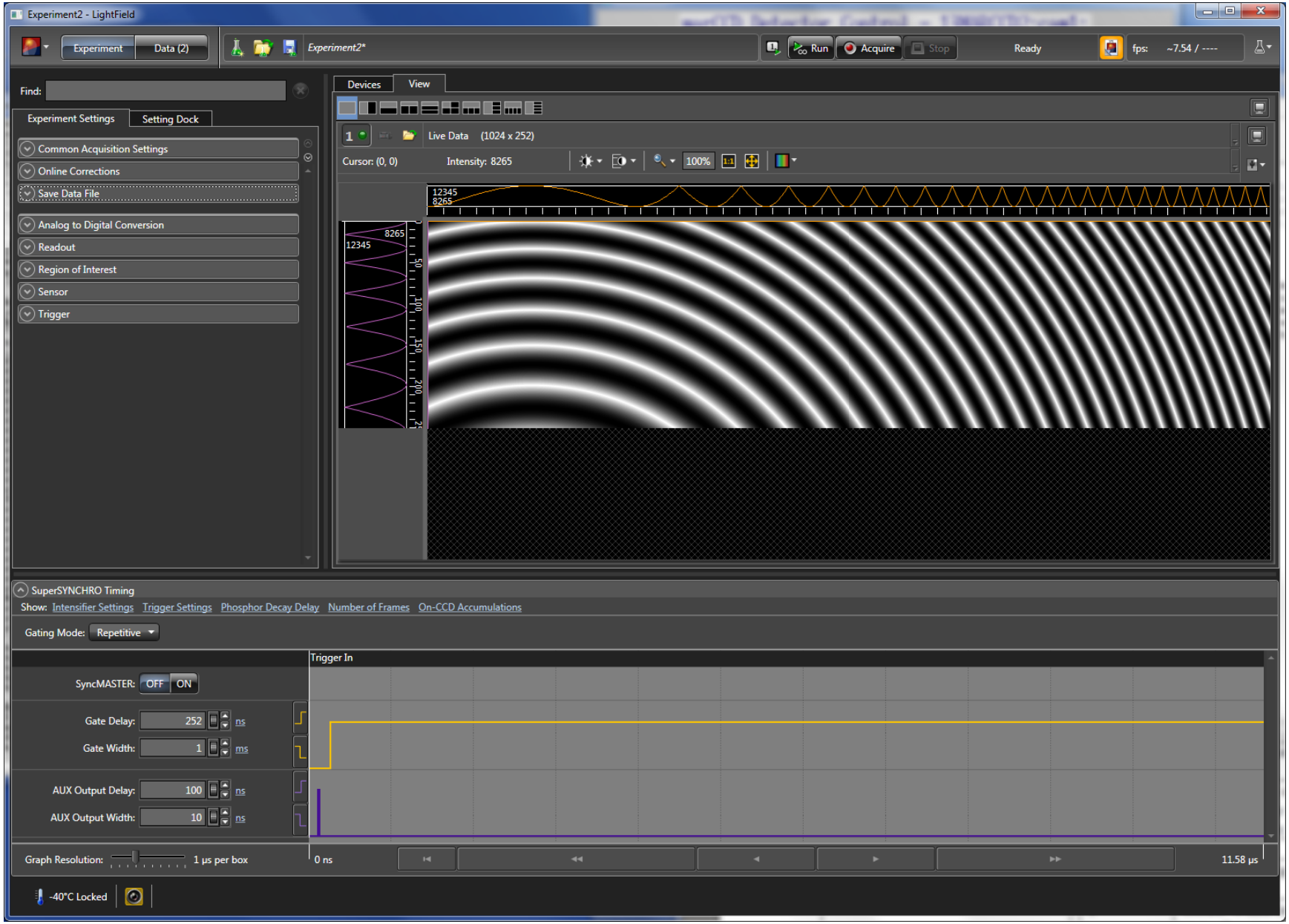
Pilatus specific control screen



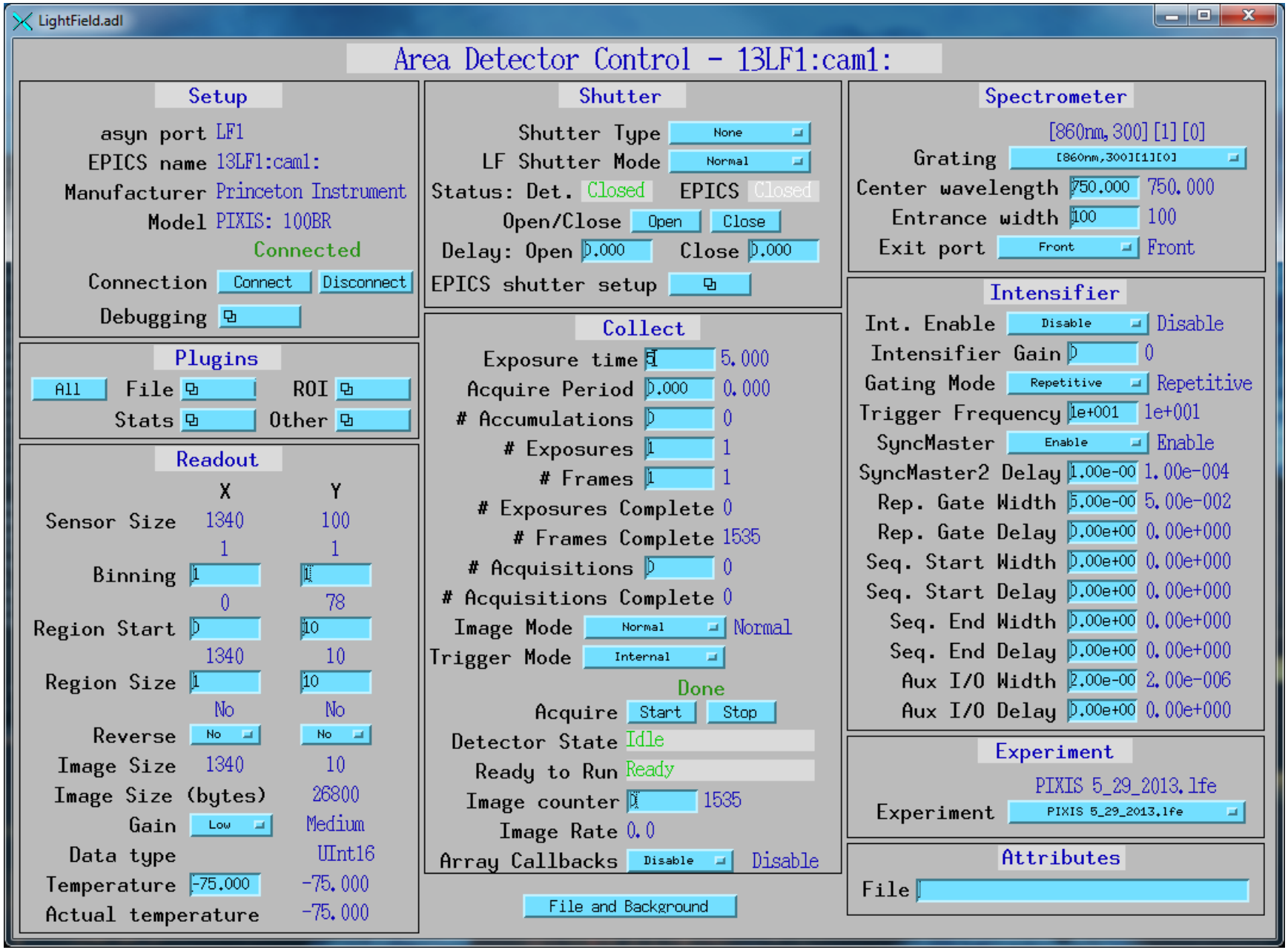
MAR-345 specific control screen



LightField driver

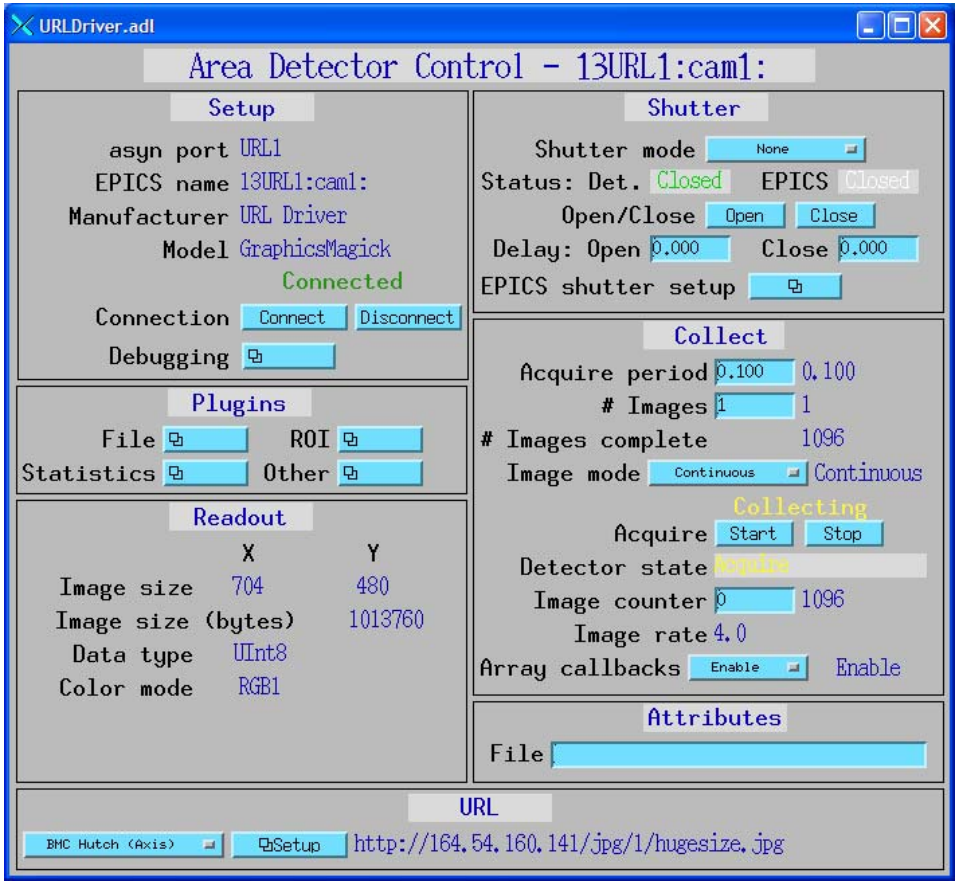


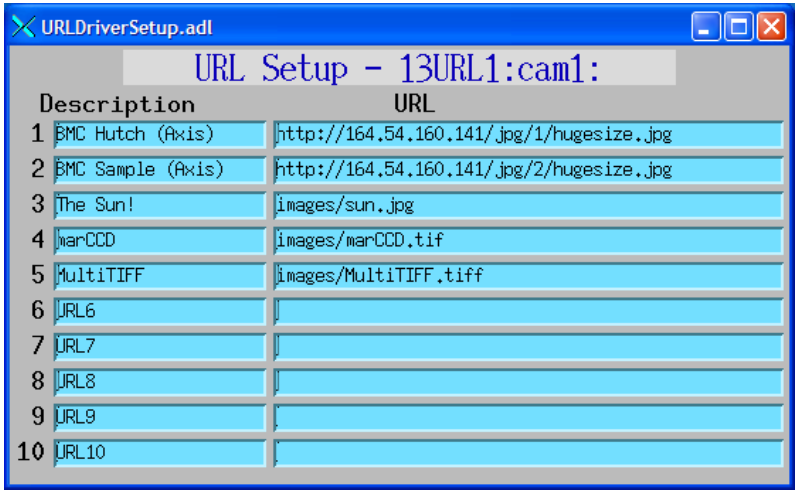
LightField driver



URL Driver

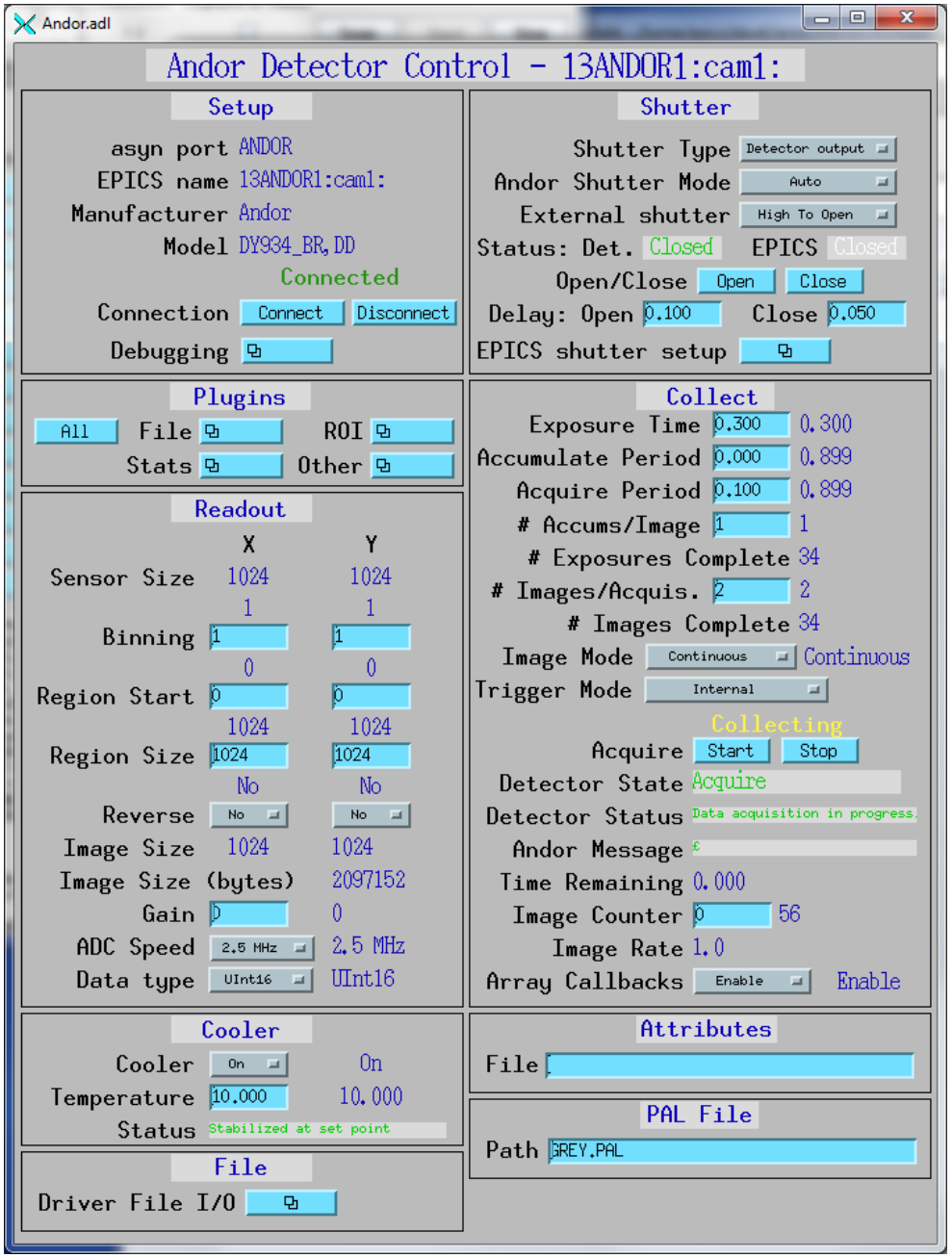
- Driver that can read images from any URL.
- Can be used with Web cameras and Axis video servers.
- Uses GraphicsMagick to read the images, and can thus handle a large number of image formats (JPEG, TIFF, PNG, etc.).



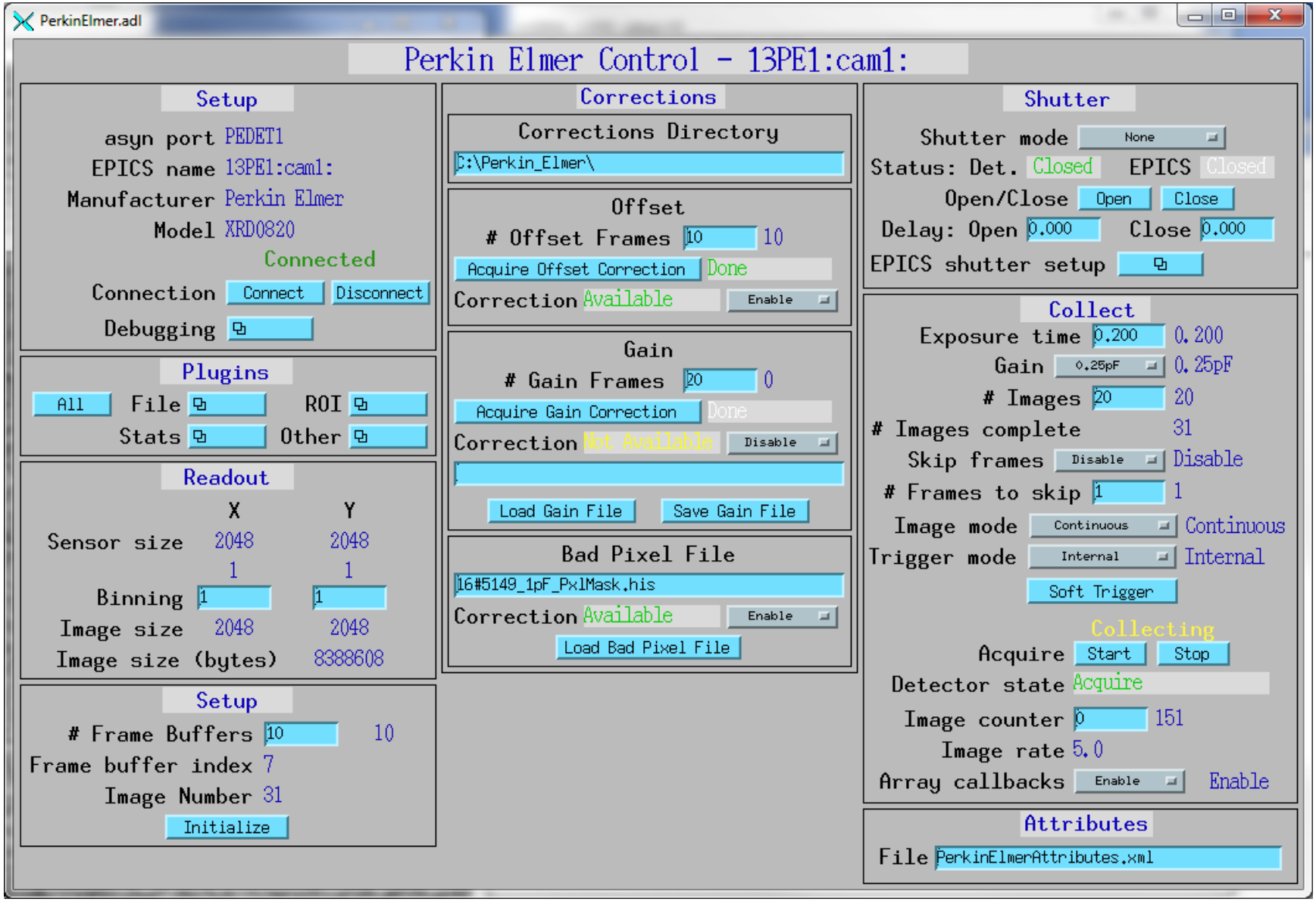


Andor Driver

- Supports USB and PCI CCD cameras from Andor.
- Runs on 32-bit and 64-bit Linux and 32-bit and 64-bit Windows.
- Original version by Matt Pearson from Diamond Light Source.



Perkin Elmer Flat Panel Driver



R2-0: Point Grey driver

- New driver for all cameras from Point Grey using their FlyCap2 SDK.
- Firewire, GigE and USB 3.0
- High performance, low cost

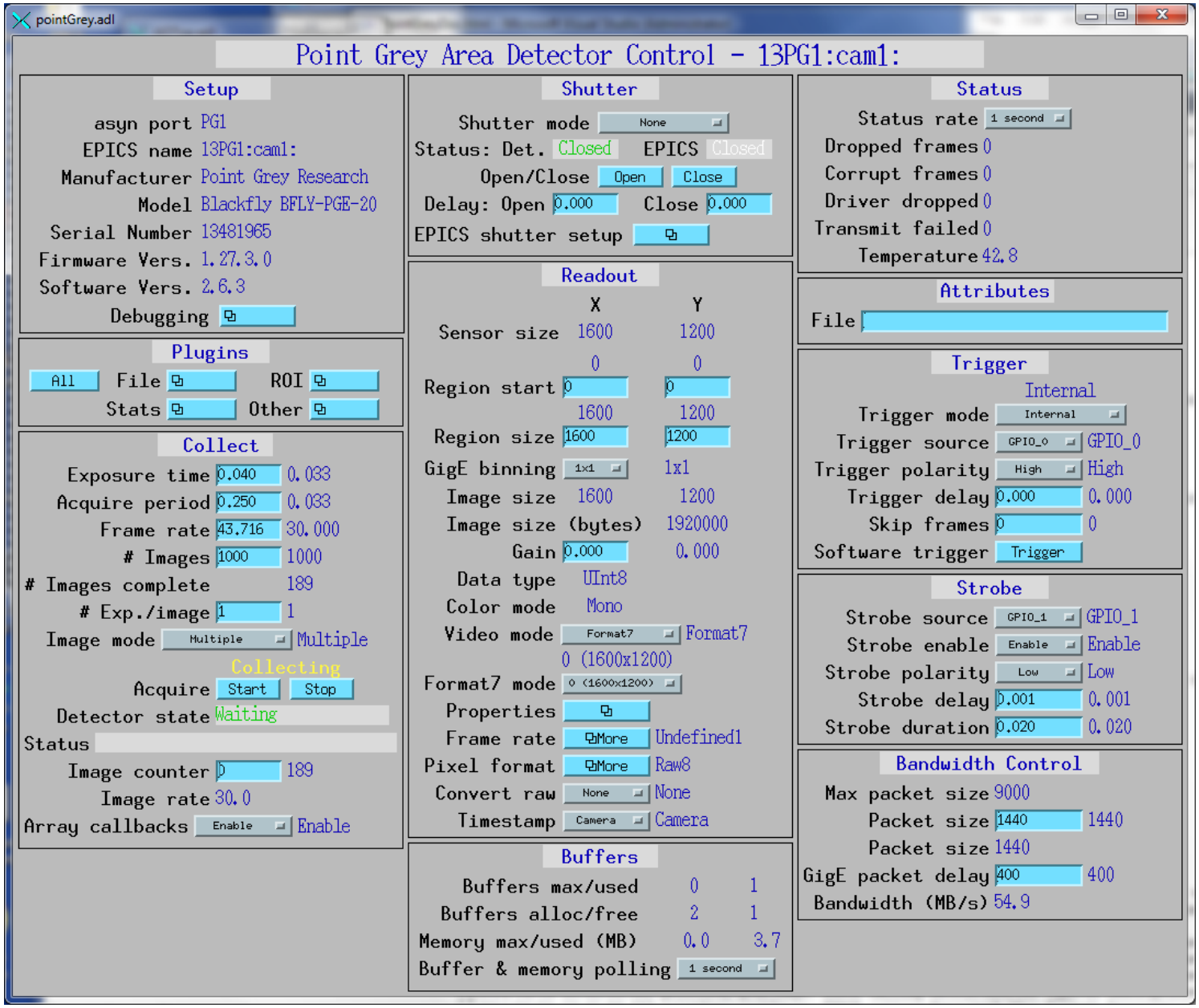
Point Grey GigE Camera
BlackFly PGE-20E4C

- e2v EV76C570 CMOS sensor
- Global shutter
- 29 x 29 x 30 mm
- Power Over Ethernet
- 4.5 micron pixels
- 1600 x 1200 pixels, color (mono)
- 47 frames/s
- $595
  - 5X cheaper than comparable Prosilica cameras we bought in the past

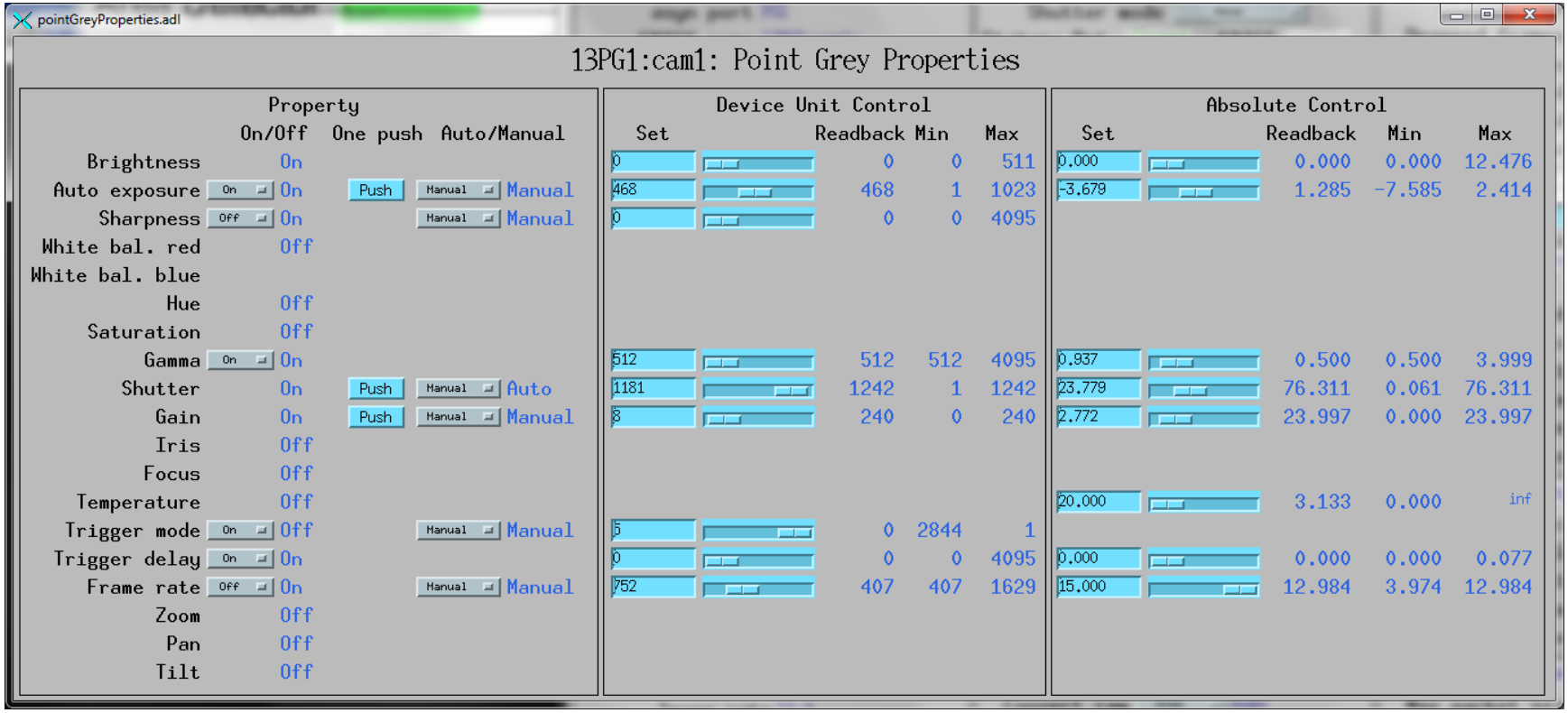
Point Grey USB-3.0 Camera Grasshopper3 GS3-U3-23S6M

- 1920 x 1200 global shutter CMOS
- Sony IMX174 1/1.2
- No smear • Distortion-free
- Dynamic range of 73 dB
- Peak QE of 76%
- Read noise of 7e-
- 12-bit or 8-bit data
- Max frame rate of 162 fps
  - ~356 MB/S, >3X faster than GigE
- USB 3.0 interface
- $1,295

Point Grey Driver

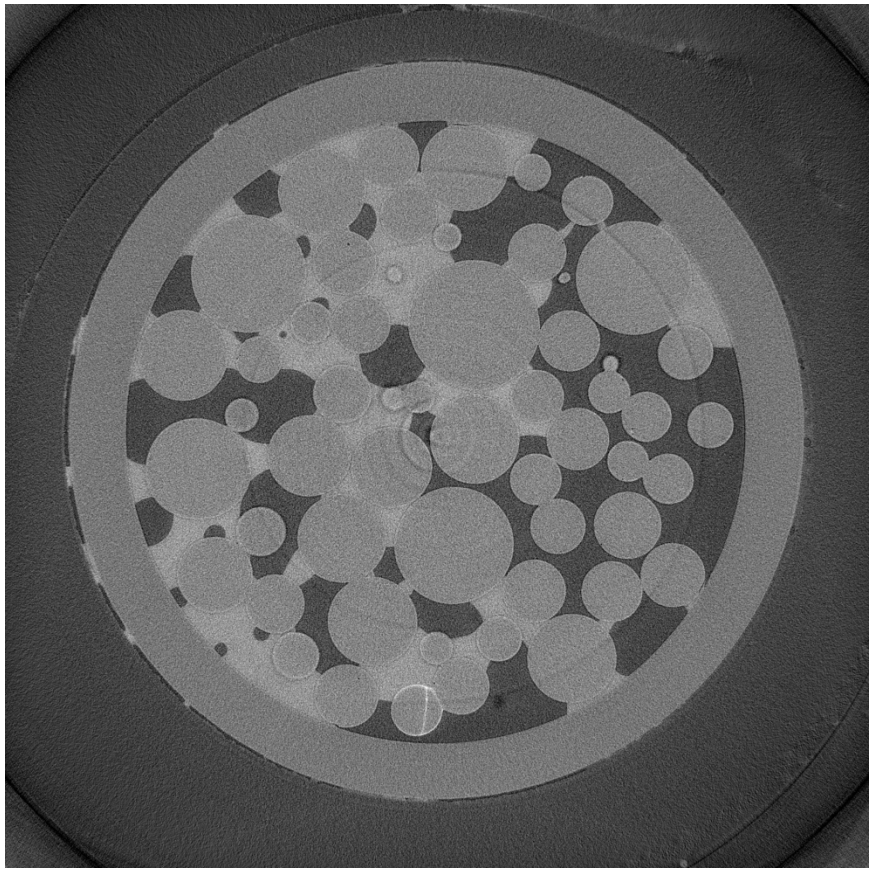


Point Grey Driver (Grasshopper3 camera)

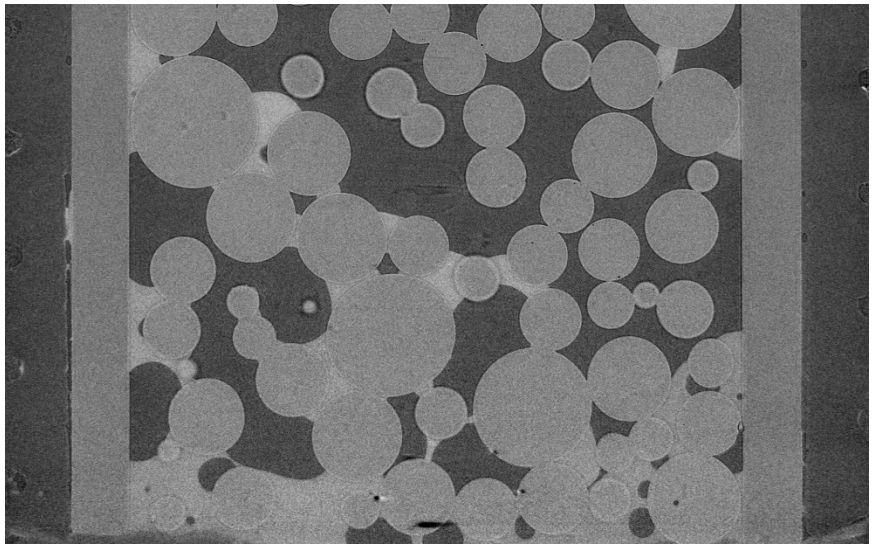


Pink Beam, Mirror=2.0 mrad

- Mirror angle=2.0 mrad (Beads_Pink_H)
- 2 mm Al absorber
- 8-bit data
- 1 ms exposure time, 124 frames/s, 900 projections, 7.3 seconds total
- Rotation axis orientation corrected for mirror angle



Horizontal slice



Vertical slice

Plugins

- Designed to perform real-time processing of data, running in the EPICS IOC (not over EPICS Channel Access)
- Receive NDArray data over callbacks from drivers or other plugins
- Plug-ins can execute in their own threads (non-blocking) or in callback thread (blocking)
  - If non-blocking then NDArray data is queued
    - Can drop images if queue is full
  - If executing in callback thread, no queuing, but slows device driver
- Allows
  - Enabling/disabling
  - Throttling rate (no more than 0.5 seconds, etc)
  - Changing data source for NDArray callbacks to another driver or plugin
- Some plugins are also sources of NDArray callbacks, as well as consumers.
  - Allows creating a data processing pipeline running at very high speed, each in a different thread, and hence in multiple cores on modern CPUs.

Plugins (continued)

- NDPlugInStdArrays
  - Receives arrays (images) from device drivers, converts to standard arrays, e.g. waveform records.
  - This plugin is what EPICS channel access viewers normally talk to.
- NDPluginROI
  - Performs region-of-interest calculations
  - Select a subregion. Optionally bin, reverse in either direction, convert data type.
  - Divide the array by a scale factor, which is useful for avoiding overflow when binning.
- NDPluginColorConvert
  - Convert from one color model to another (Mono, RGB1 (pixel), RGB2 (row) or RGB3 (planar) interleave)
  - Bayer conversion removed from this plugin, now part of Prosilica and Point Grey drivers.
- NDPluginTransform
  - Performs geometric operations (rotate, mirror in X or Y, etc.)

Plugins (continued)

- NDPluginStats
  - Calculates basic statistics on an array (min, max, sigma)
  - Optionally computes centroid centroid position, width and tilt.
  - Optionally Computes X and Y profiles, including average profiles, profiles at the centroid position, and profiles at a user-defined cursor position.
  - Optionally computes the image histogram and entropy
- NDPluginROIStat
  - Multiple ROIs with simple statistics in a single plugin
  - More efficient when many ROIs are needed, e.g. for peaks in a 1-D energy spectrum
  - Min, max, total, net, mean
  - Time-series of each of these statistics

Plugins (continued)

- NDPluginProcess
  - Does arithmetic processing on arrays
  - Background subtraction.
  - Flat field normalization.
  - Offset and scale.
  - Low and high clipping.
  - Recursive filtering in the time domain.
  - Conversion to a different output data type.
- NDPluginOverlay
  - Adds graphic overlays to an image.
  - Can be used to display ROIs, multiple cursors, user-defined boxes, text, etc.
- ffmpegServer
  - MJPEG server that allows viewing images in a Web browser. From DLS.

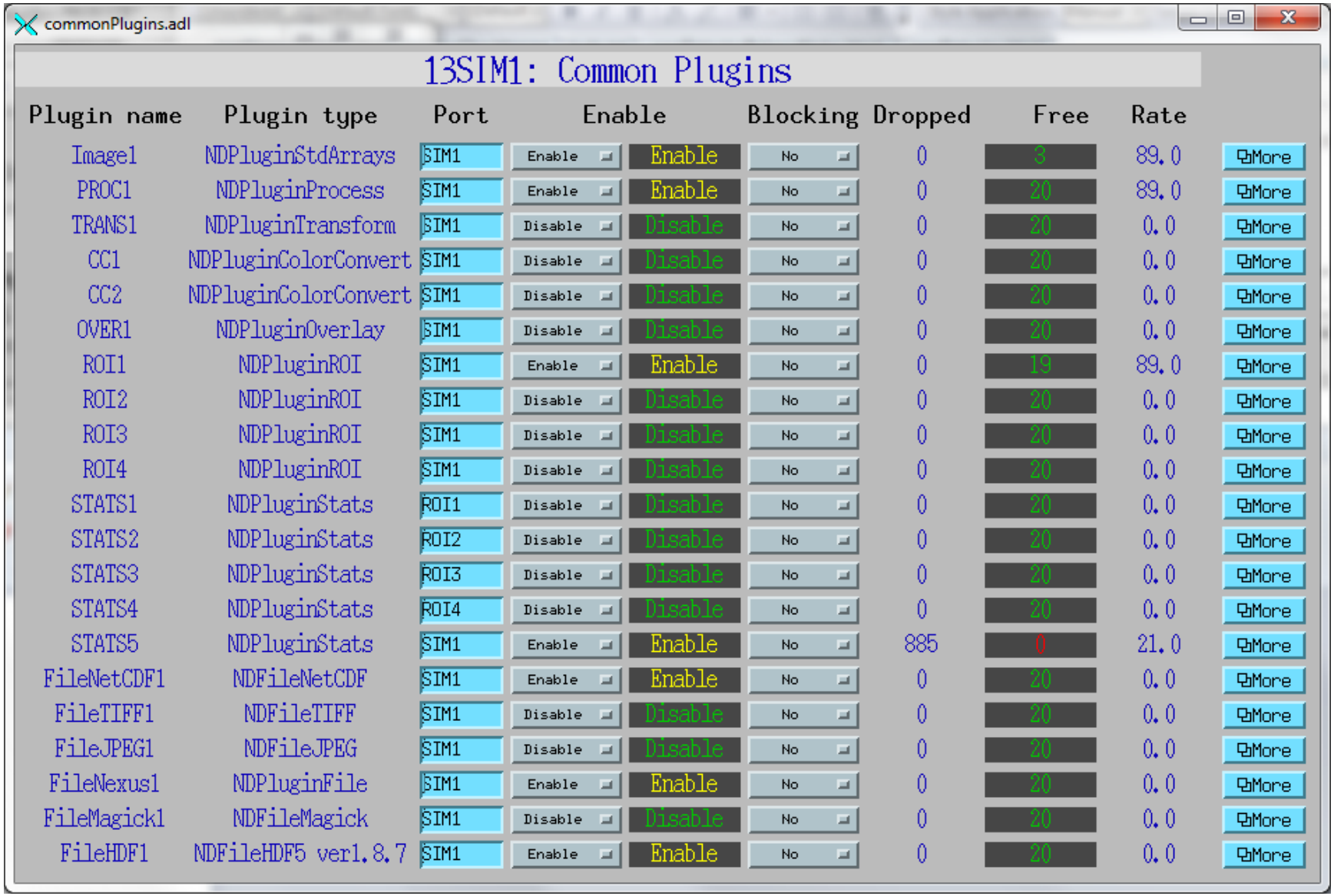
Plugins (continued)

- NDPluginAttribute
  - Extracts NDAttributes from NDArrays and publishes their values as ai records
  - Can collect time-series arrays of the attribute values
- NDPluginCircularBuff
  - Buffers NDArrays in a circular buffer
  - Computes a trigger expression using up to 2 NDAttribute values
  - When trigger condition is met then outputs NDArrays
  - User-specified number of pre-trigger and post-trigger arrays to output
- NDPluginTimeSeries
  - Accepts 1-D NDArrays[NumSignals] or 2-D [NumSignals,NewTimePoints] and appends to time-series buffer
  - Operates in fixed length (stop when full) or circular buffer modes
  - Optional time-averaging of input data

Plugins (continued)

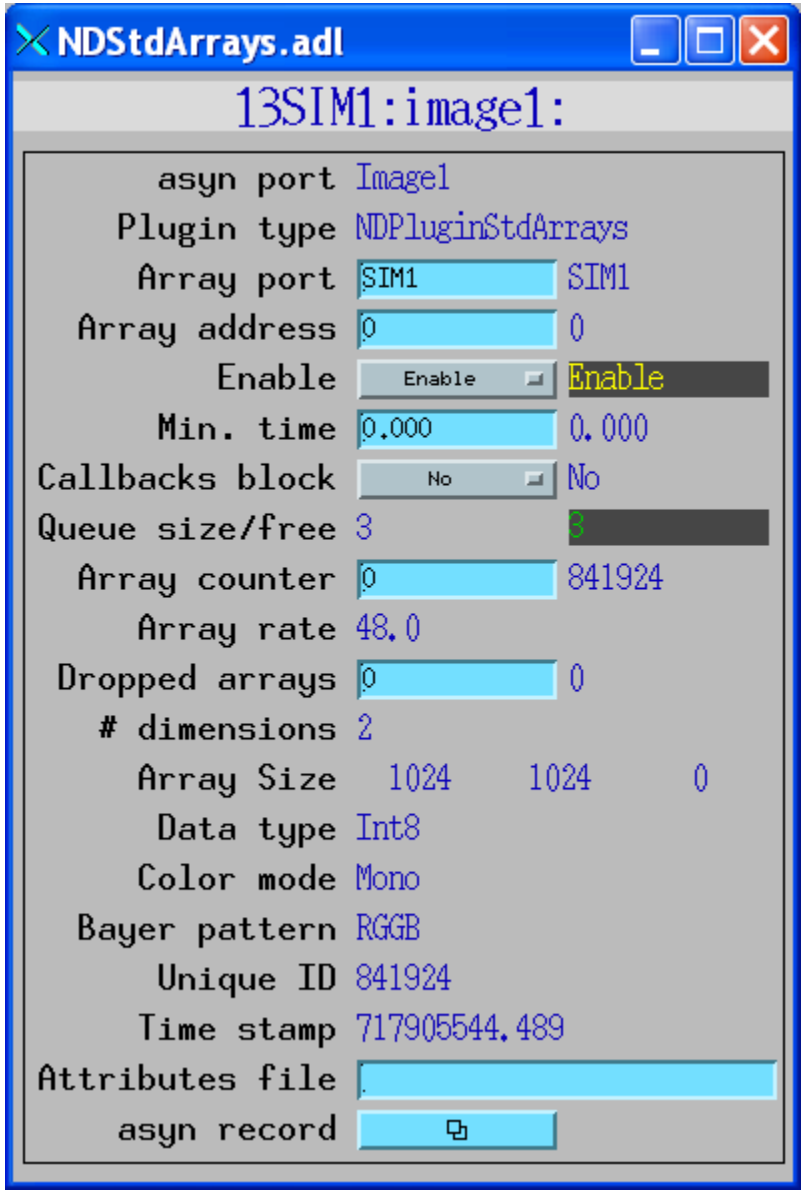
- NDPluginFFT
  - Computes FFT of 1-D or 2-D NDArrays
  - Exports NDArrays containing the absolute value (power spectrum) of the FFT
  - Exports 1-D arrays of the FFT real, imaginary, absolute values, and time and frequency data.
- NDPluginPVA
  - Converts NDArrays to EPICS V4 NTNDArrays
  - Exports the NtNDArrays over PVAccess with internal V4 server
  - Can be used to send structured data to EPICS V4 clients
  - When used with the PVAccess driver then areaDetector plugins can be run on different machine from the detector driver

commonPlugins.adl All plugins at a glance

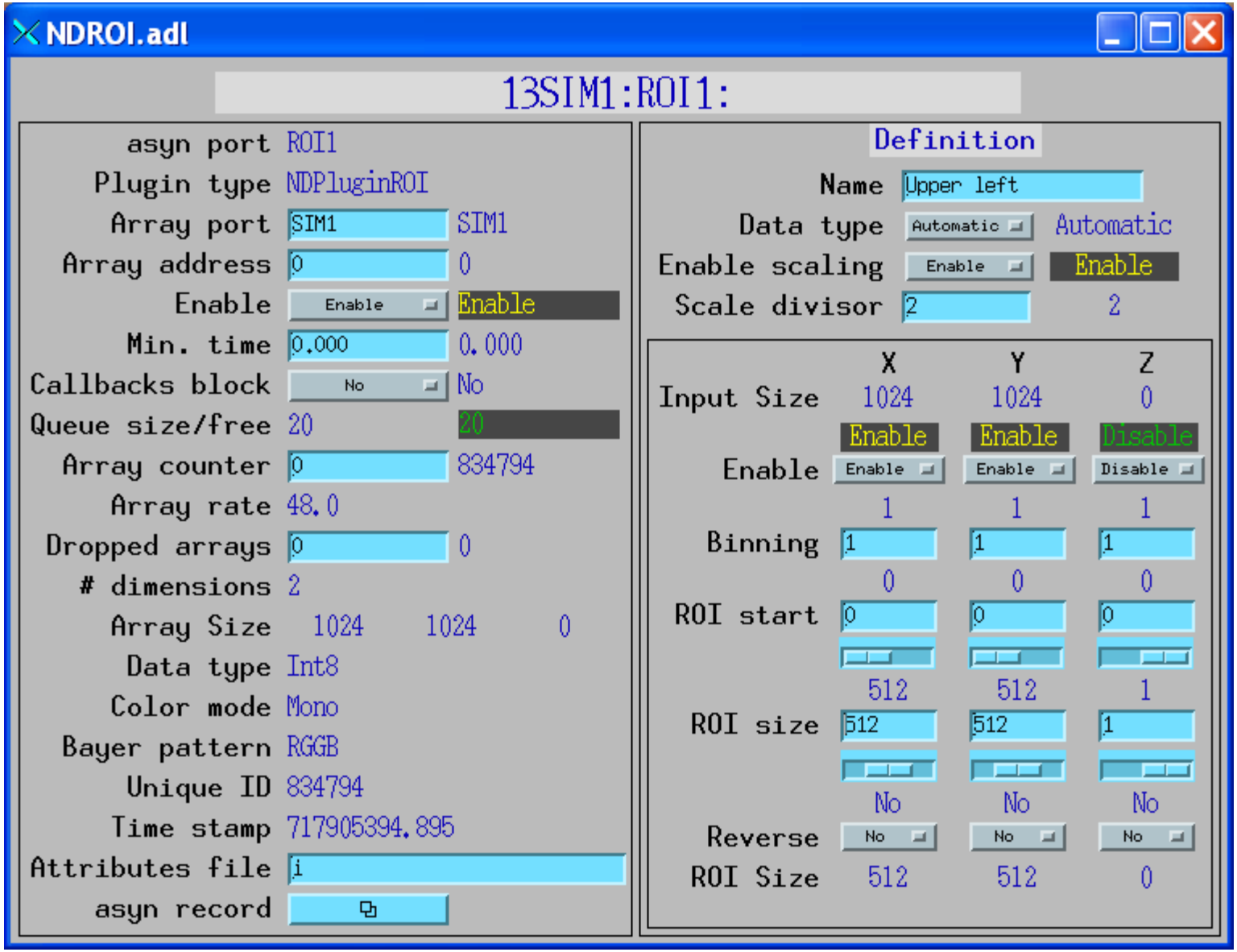


Plugin name	Plugin type	Port	Enable		Blocking	Dropped	Free	Rate	
Image1	NDPluginStdArrays	SIM1	Enable	Enable	No	0	3	89.0	More
PROC1	NDPluginProcess	SIM1	Enable	Enable	No	0	20	89.0	More
TRANS1	NDPluginTransform	SIM1	Disable	Disable	No	0	20	0.0	More
CC1	NDPluginColorConvert	SIM1	Disable	Disable	No	0	20	0.0	More
CC2	NDPluginColorConvert	SIM1	Disable	Disable	No	0	20	0.0	More
OVER1	NDPluginOverlay	SIM1	Disable	Disable	No	0	20	0.0	More
ROI1	NDPluginROI	SIM1	Enable	Enable	No	0	19	89.0	More
ROI2	NDPluginROI	SIM1	Disable	Disable	No	0	20	0.0	More
ROI3	NDPluginROI	SIM1	Disable	Disable	No	0	20	0.0	More
ROI4	NDPluginROI	SIM1	Disable	Disable	No	0	20	0.0	More
STATS1	NDPluginStats	ROI1	Disable	Disable	No	0	20	0.0	More
STATS2	NDPluginStats	ROI2	Disable	Disable	No	0	20	0.0	More
STATS3	NDPluginStats	ROI3	Disable	Disable	No	0	20	0.0	More
STATS4	NDPluginStats	ROI4	Disable	Disable	No	0	20	0.0	More
STATS5	NDPluginStats	SIM1	Enable	Enable	No	885	0	21.0	More
FileNetCDF1	NDFileNetCDF	SIM1	Enable	Enable	No	0	20	0.0	More
FileTIFF1	NDFileTIFF	SIM1	Disable	Disable	No	0	20	0.0	More
FileJPEG1	NDFileJPEG	SIM1	Disable	Disable	No	0	20	0.0	More
FileNexus1	NDPluginFile	SIM1	Enable	Enable	No	0	20	0.0	More
FileMagick1	NDFileMagick	SIM1	Disable	Disable	No	0	20	0.0	More
FileHDF1	NDFileHDF5 ver1.8.7	SIM1	Enable	Enable	No	0	20	0.0	More

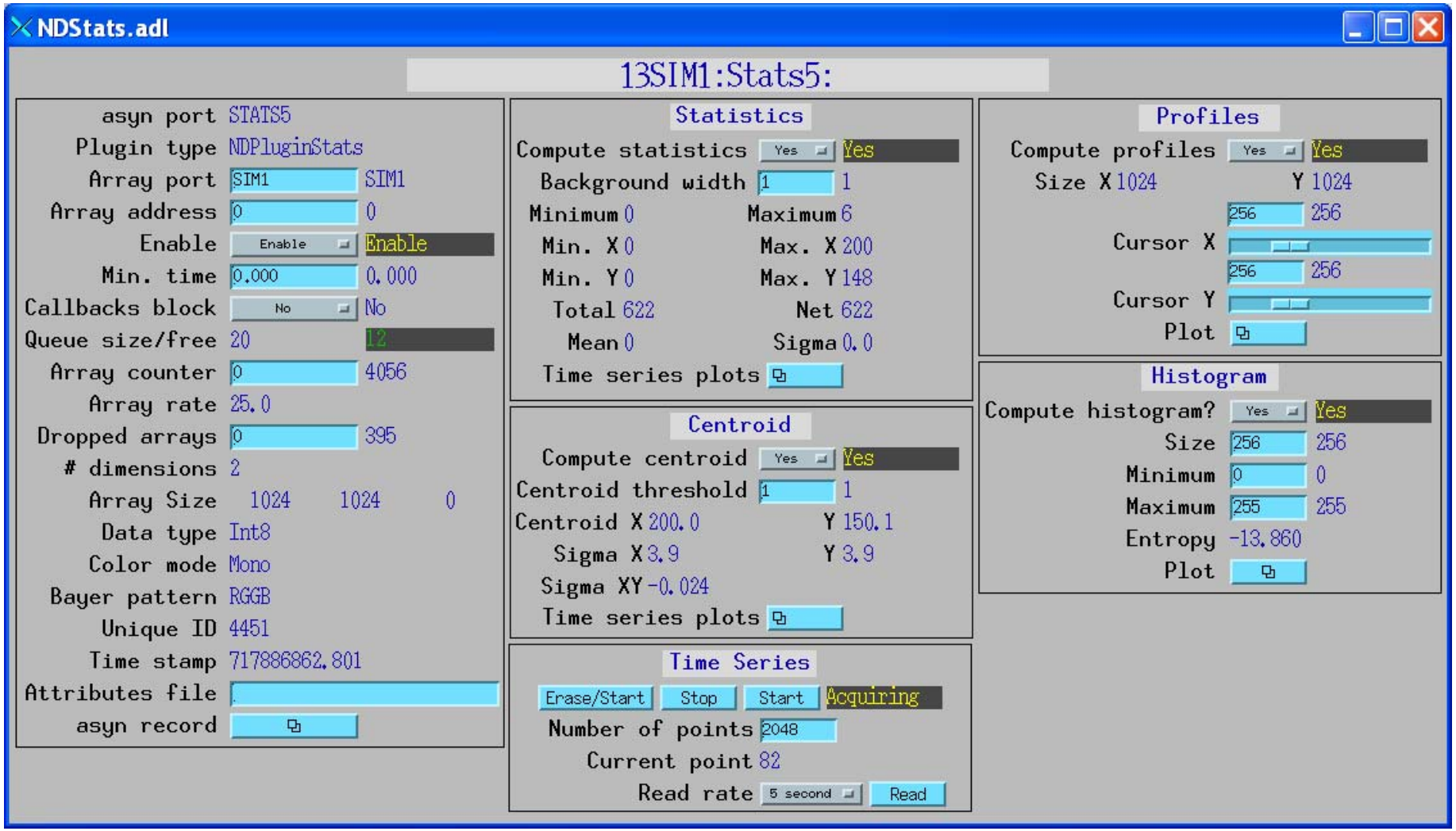
NDStdArrays plugin



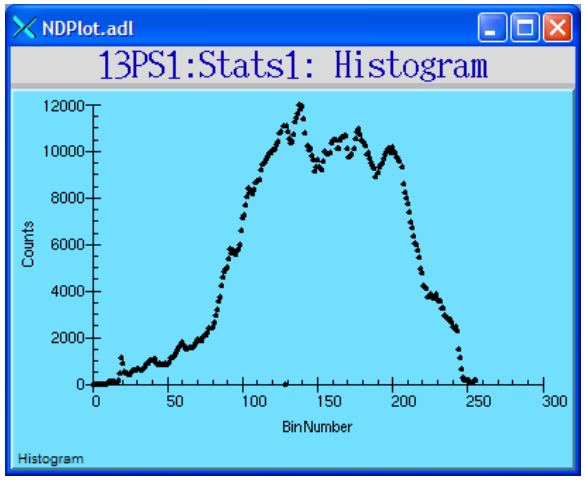
ROI plugin

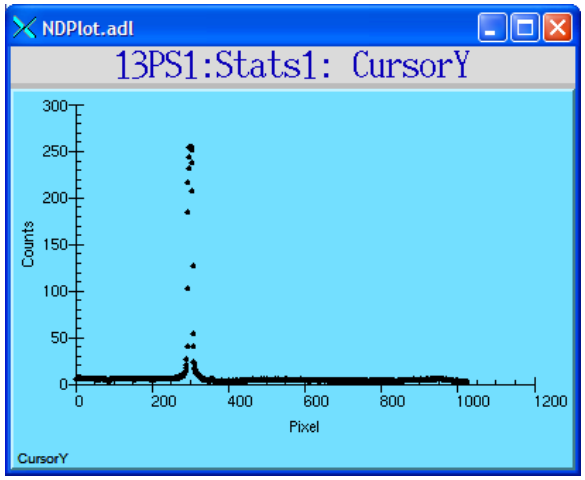

NDROI.adl
13SIM1:ROI1:
asyn port ROI1
Plugin type NDPluginROI
Array port SIM1 SIM1
Array address 0 0
Enable Enable Enable
Min. time 0.000 0.000
Callbacks block No No
Queue size/free 20 20
Array counter 0 834794
Array rate 48.0
Dropped arrays 0 0
dimensions 2
Array Size 1024 1024 0
Data type Int8
Color mode Mono
Bayer pattern RGGB
Unique ID 834794
Time stamp 717905394.895
Attributes file i
asyn record
Definition
Name Upper left
Data type Automatic Automatic
Enable scaling Enable Enable
Scale divisor 2 2
X Y Z
Input Size 1024 1024 0
Enable Enable Enable Disable
Enable Enable Enable Disable
1 1 1
Binning 1 1 1
0 0 0
ROI start 0 0 0
512 512 1
ROI size 512 512 1
No No No
Reverse No No No
ROI Size 512 512 0

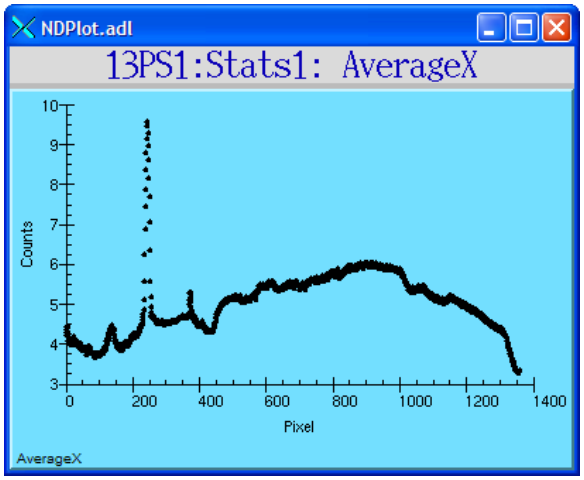
Statistics plugin

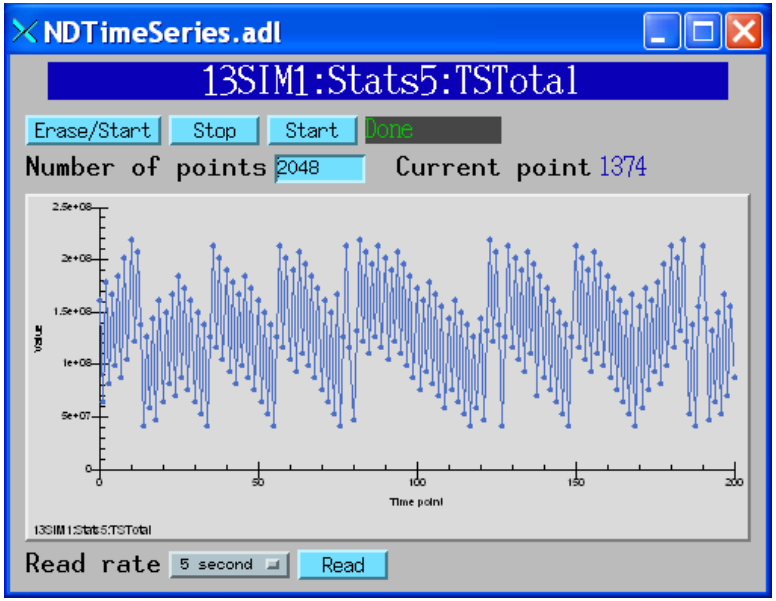


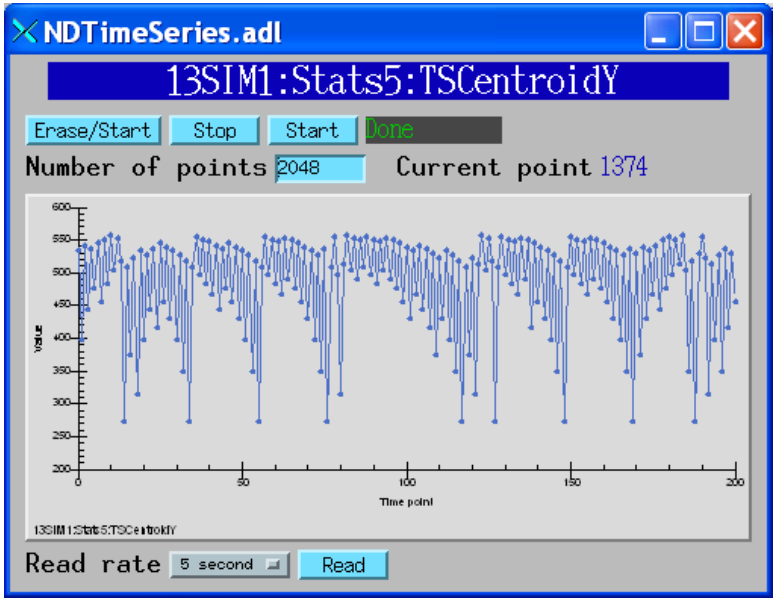
Statistics plugin (continued)



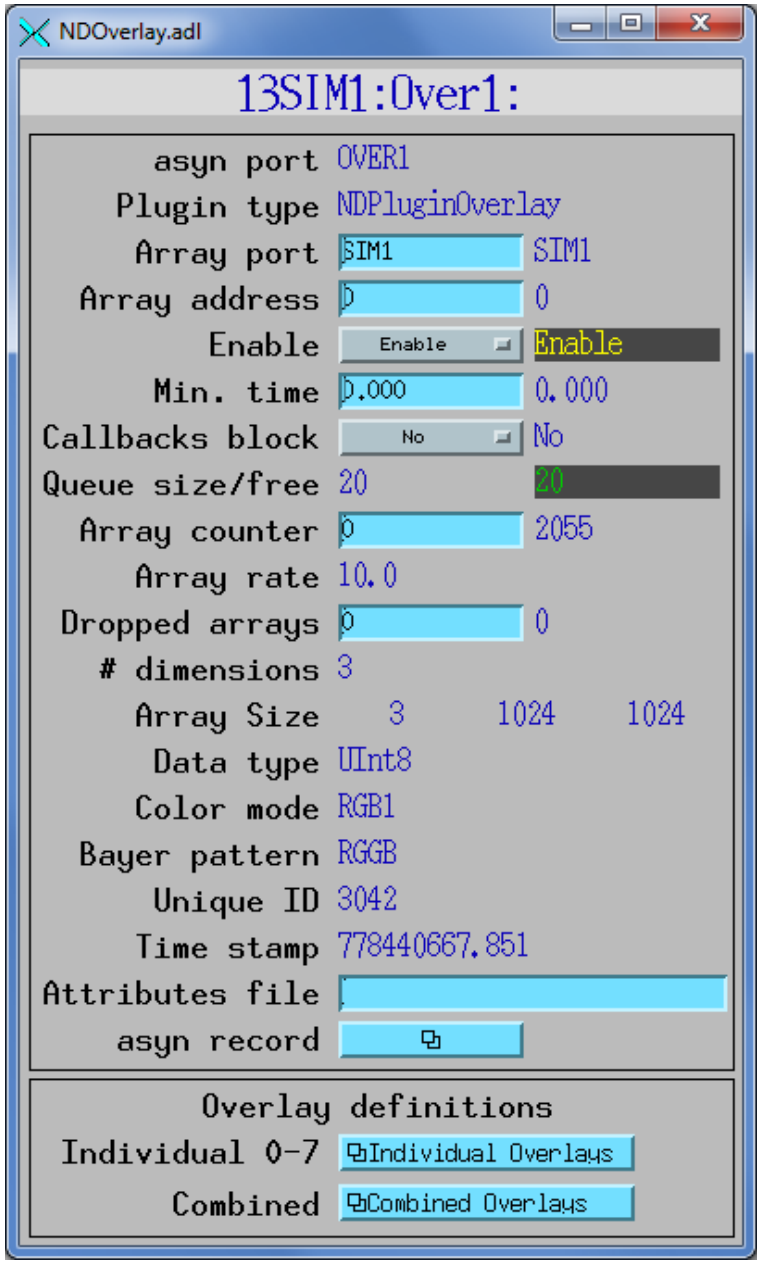






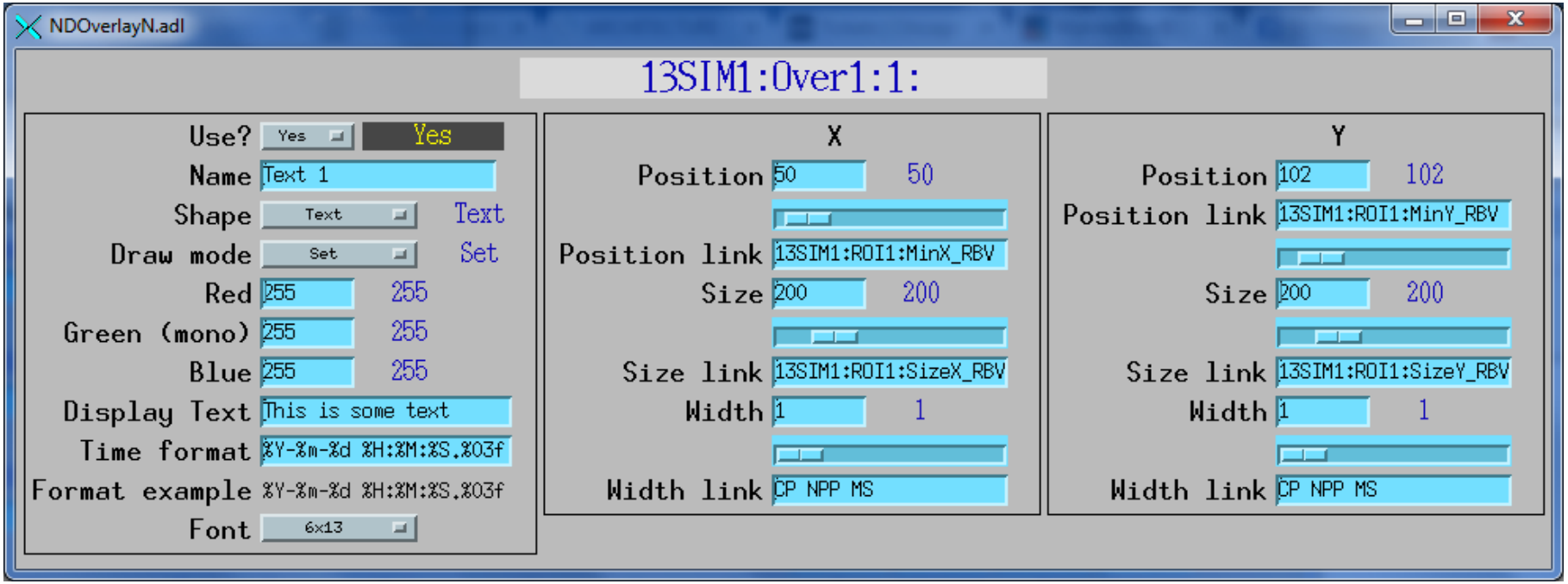


Overlay plugin

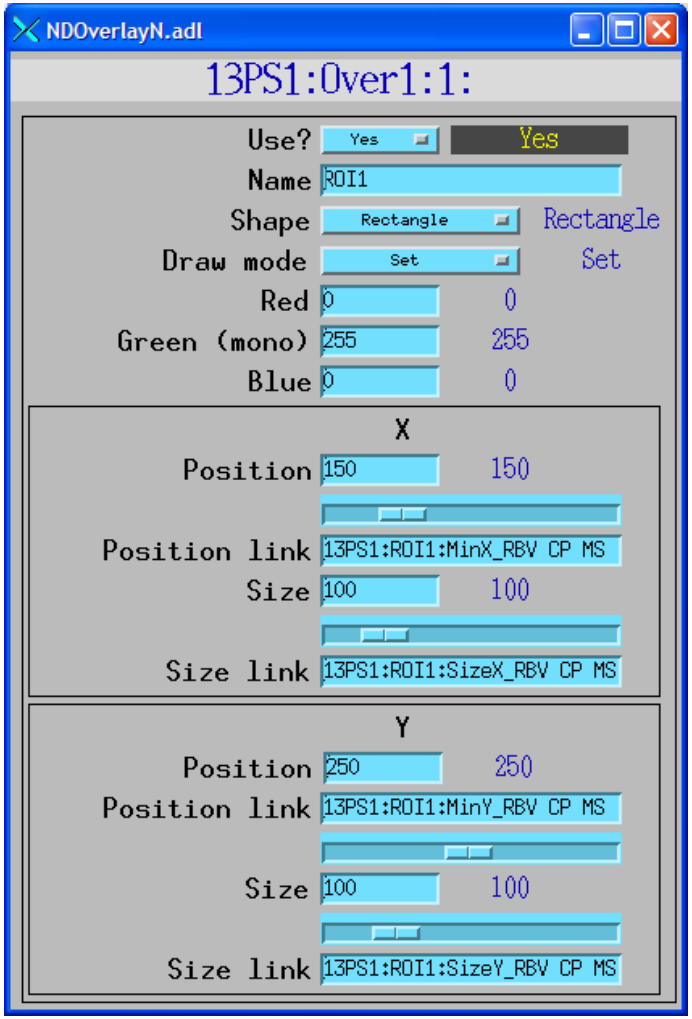


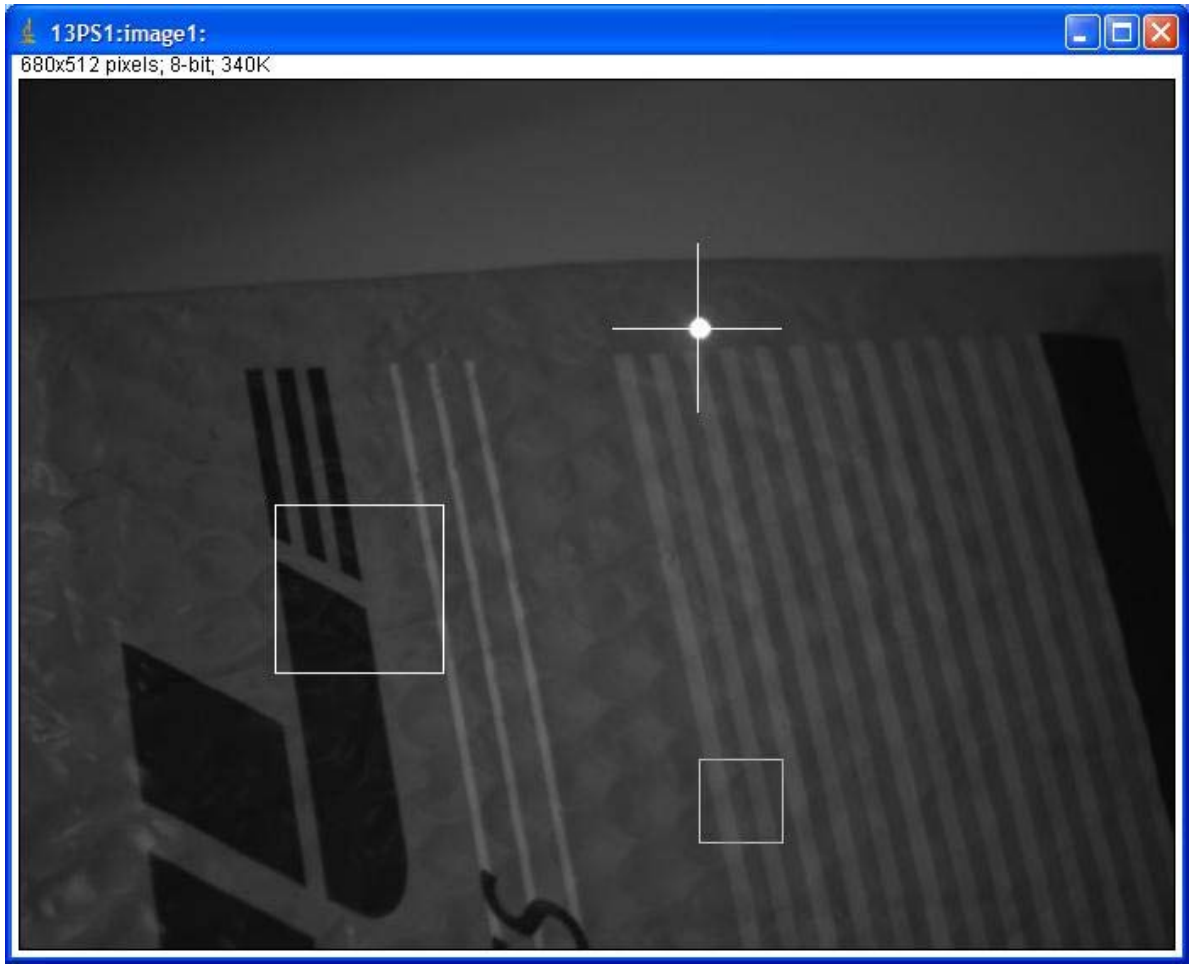
Overlay plugin

Overlay plugin



Overlay plugin

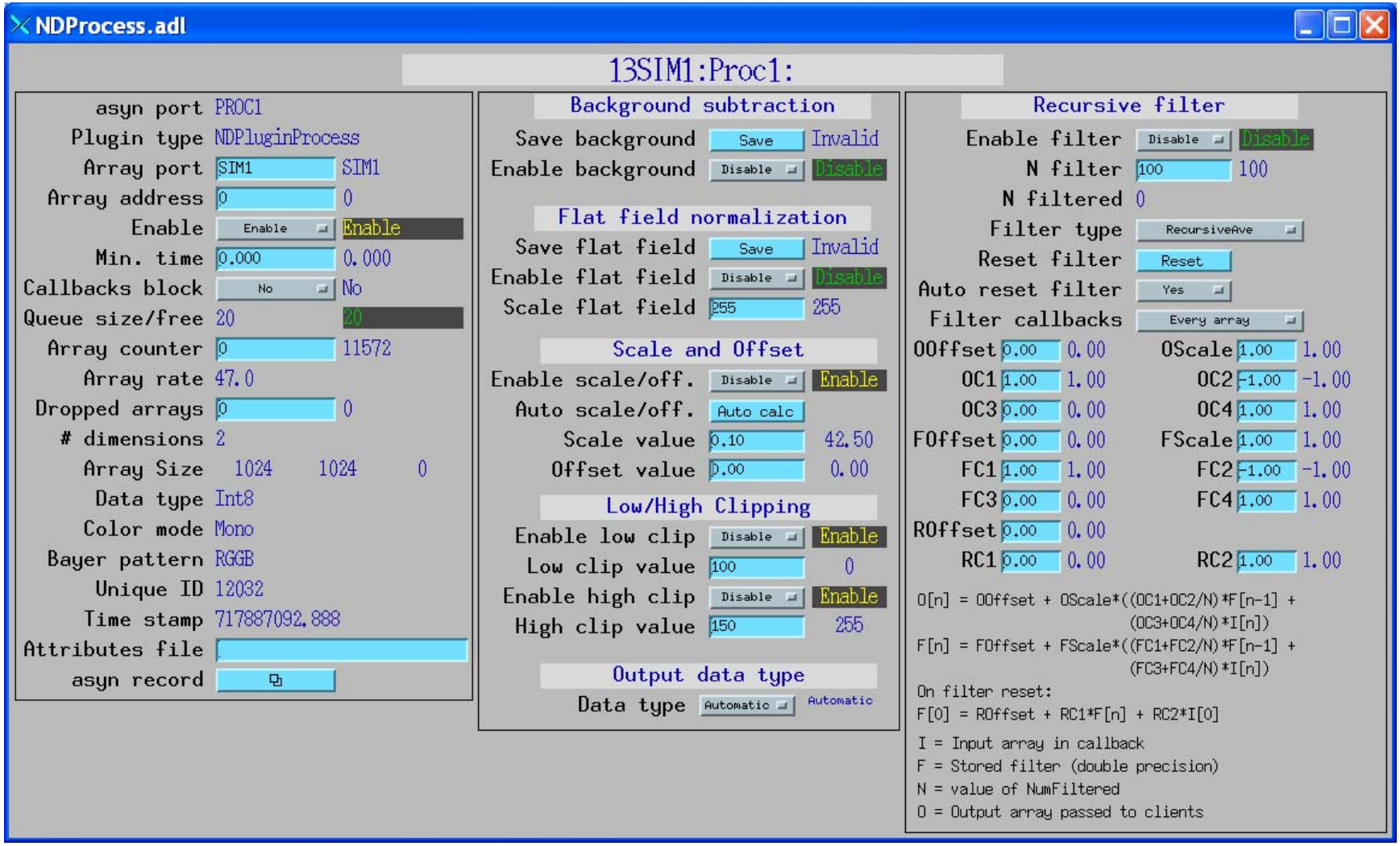




Centroid of laser pointer calculated by statistics plugin

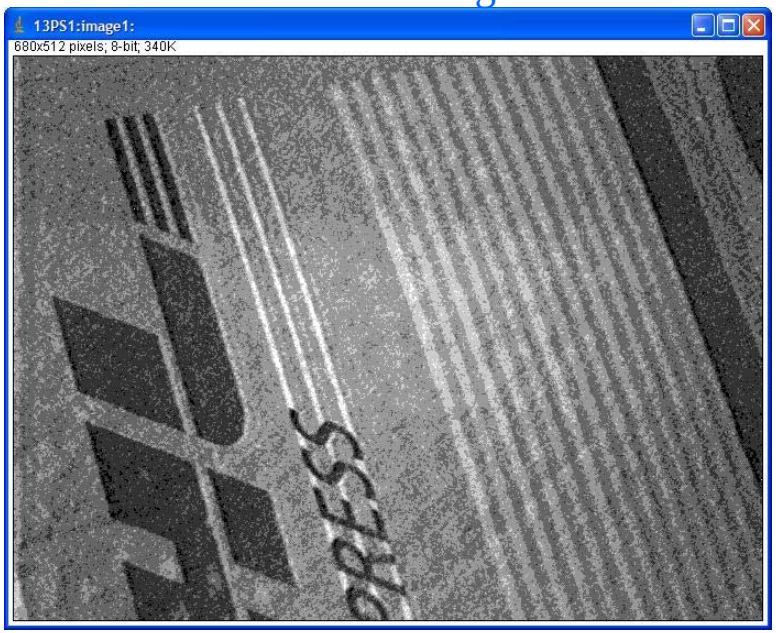
Cursor overlay X, Y position linked to centroid

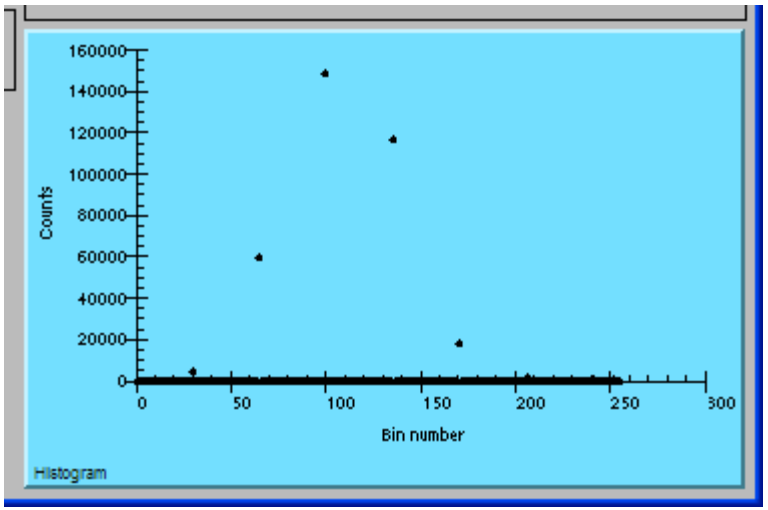
Processing plugin



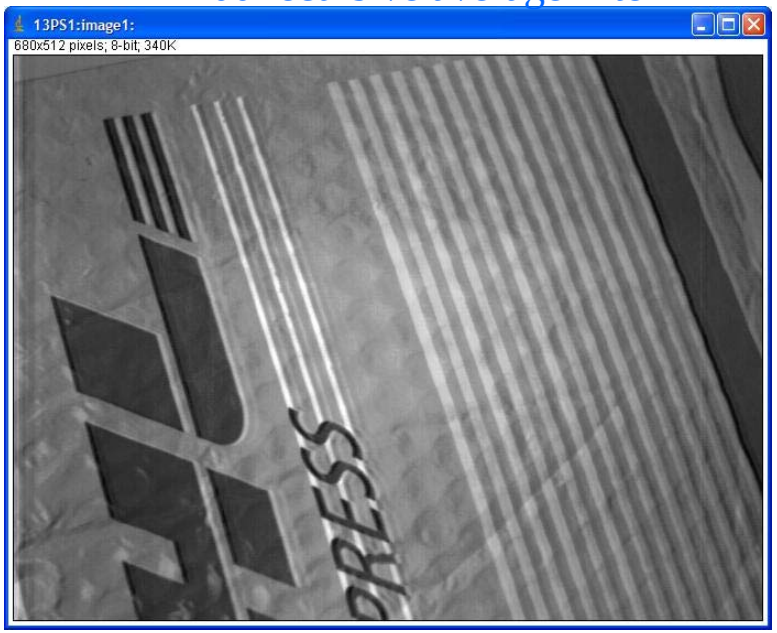
Processing plugin
30 microsec exposure time

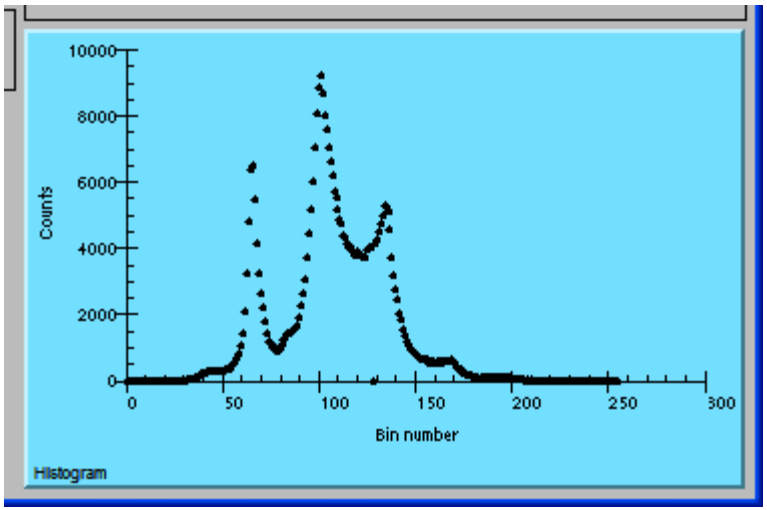
No filtering





N=100 recursive average filter

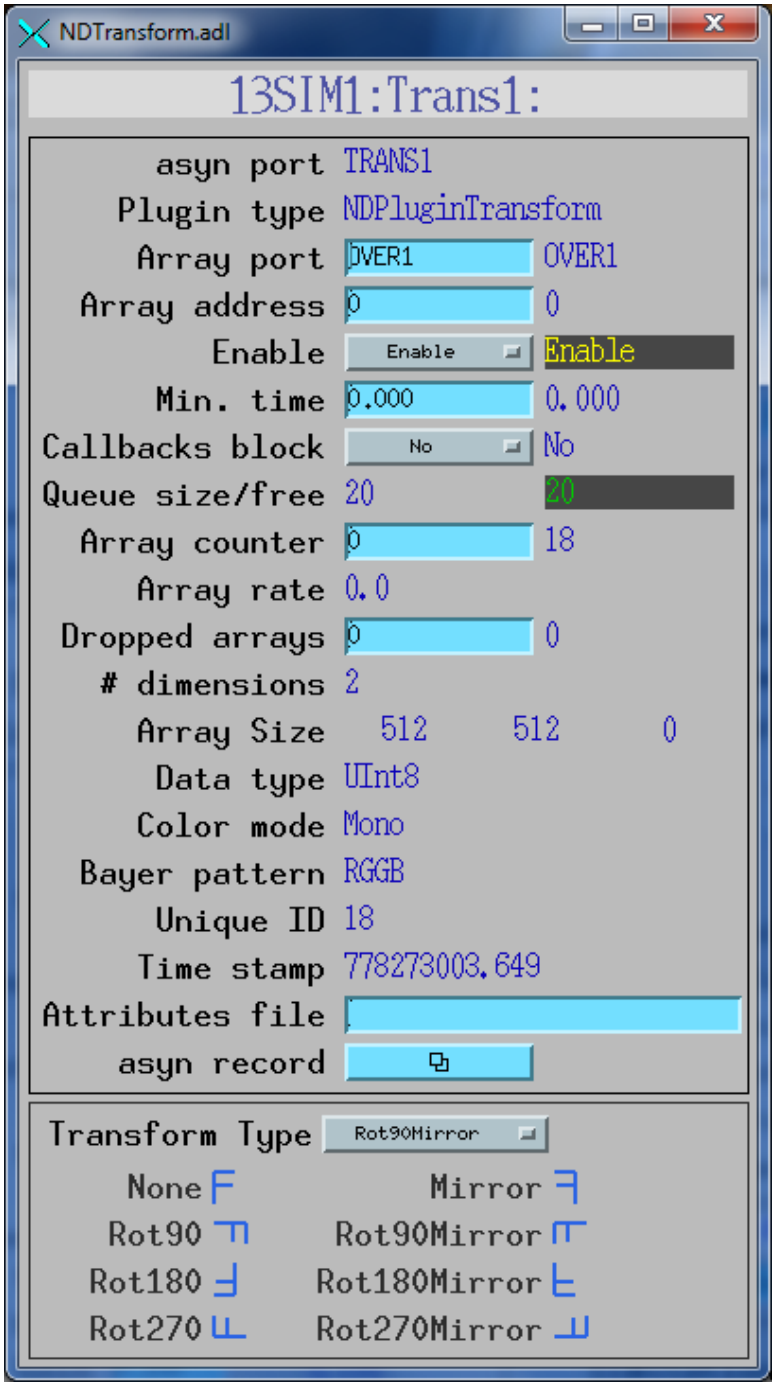




Transform plugin

R2-1 changes

- Greatly simplified: just 8 operations including null operation
- 13-85 times faster than previous releases depending on data type, color mode



Plugins: NDPluginFile

- Saves NDArrays to disk
- 3 modes:
  - Single array per disk file
  - Capture N arrays in memory, write to disk either multiple files or as a single large file (for file formats that support this.)
  - Stream arrays to a single large disk file
- For file formats that support it, stores not just NDArray data but also NDAttributes

Plugins: NDPluginFile

- File formats currently supported
  - NDFileTIFF
    - Supports any NDArray data type
    - Stores NDAttributes as ASCII user tags
  - NDFileJPEG
    - With compression control
  - NDFileNetCDF
    - Popular self-describing binary format, supported by Unidata at UCAR
  - NDFileHDF5
    - Writes HDF5 files with the native HDF5 API, unlike the NeXus plugin which uses the NeXus API. Supports 3 types of compression.
    - Supports using an XML file to define the layout and placement of NDArrays and NDAttributes in the HDF5 file
    - R2-5 will support Single Writer Multiple Reader (SWMR). Only supported on local file systems, GPFS, and Lustre (not NFS or SMB)

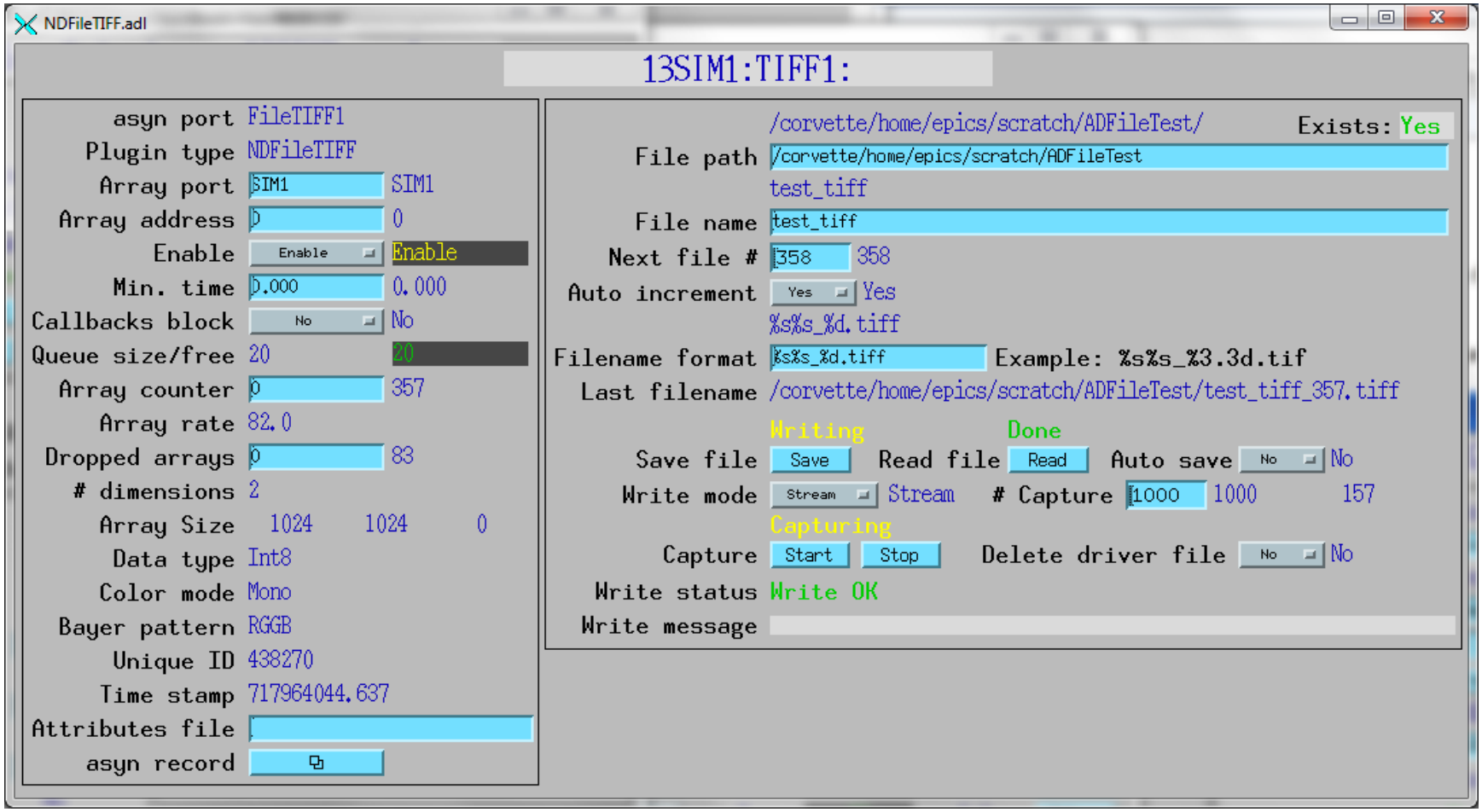
Plugins: NDPluginFile

- File formats currently supported
  - NDFileNeXus
    - Standard file format for neutron and x-ray communities, based on HDF5, which is another popular self-describing binary format; richer than netCDF
    - May be deprecated in a future release since NeXus files can now be produced with the NDFileHDF5 plugin using an appropriate XML layout file
  - NDFileMagick
    - Uses GraphicsMagick to write files, and can write in dozens of file formats, including JPEG, TIFF, PNG, PDF, etc.
  - NDFileNull
    - Used only to delete original driver files when no other file plugin is running

File saving with driver

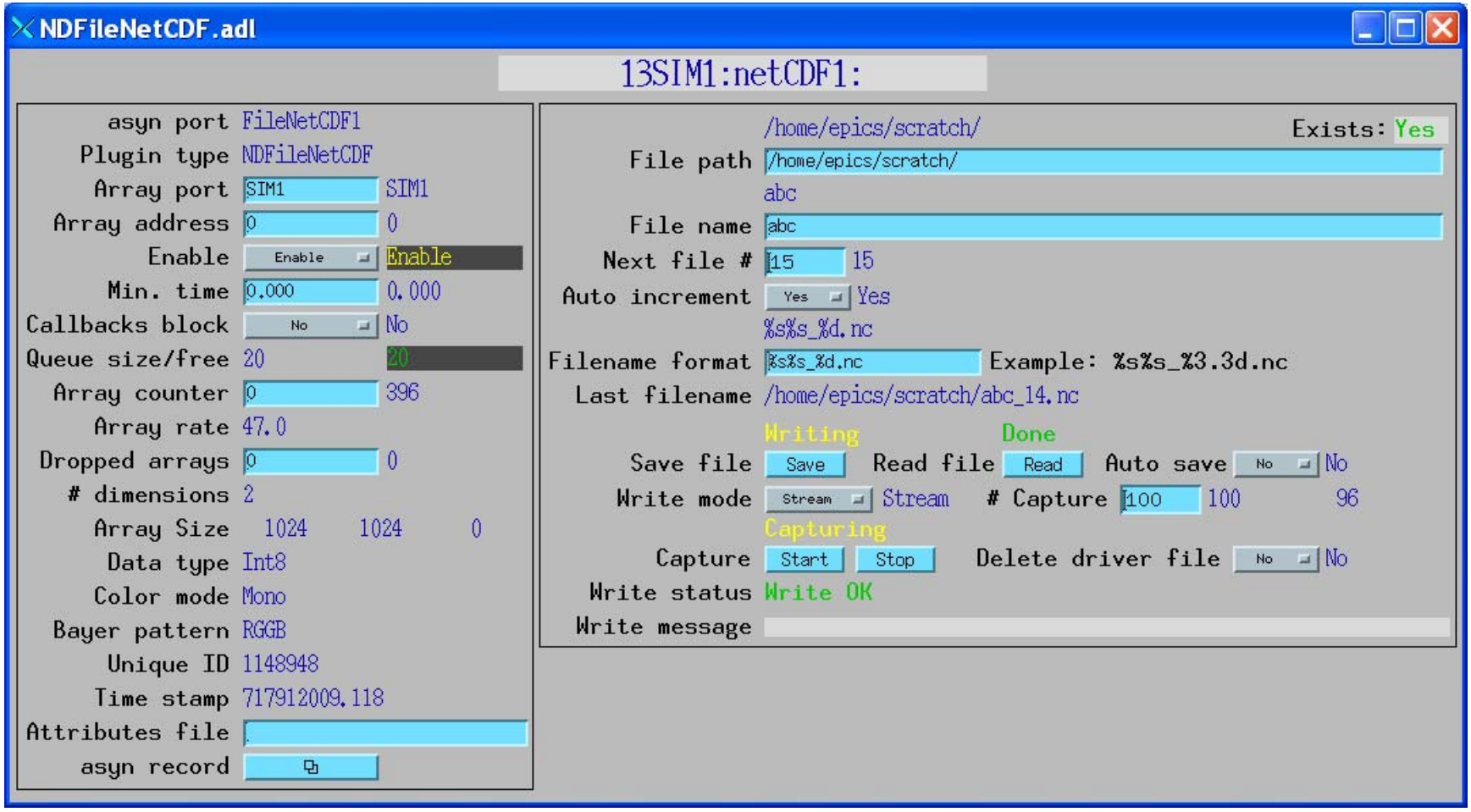
- In addition to file saving plugins, many vendor libraries also support saving files (e.g. marCCD, mar345, Pilatus, etc.) and this is supported at the driver level.
- File saving plugin can be used instead of or in addition to vendor file saving
  - Can add additional metadata vendor does not support
  - Could write JPEGS for Web display every minute, etc.

NDPluginFile display: TIFF



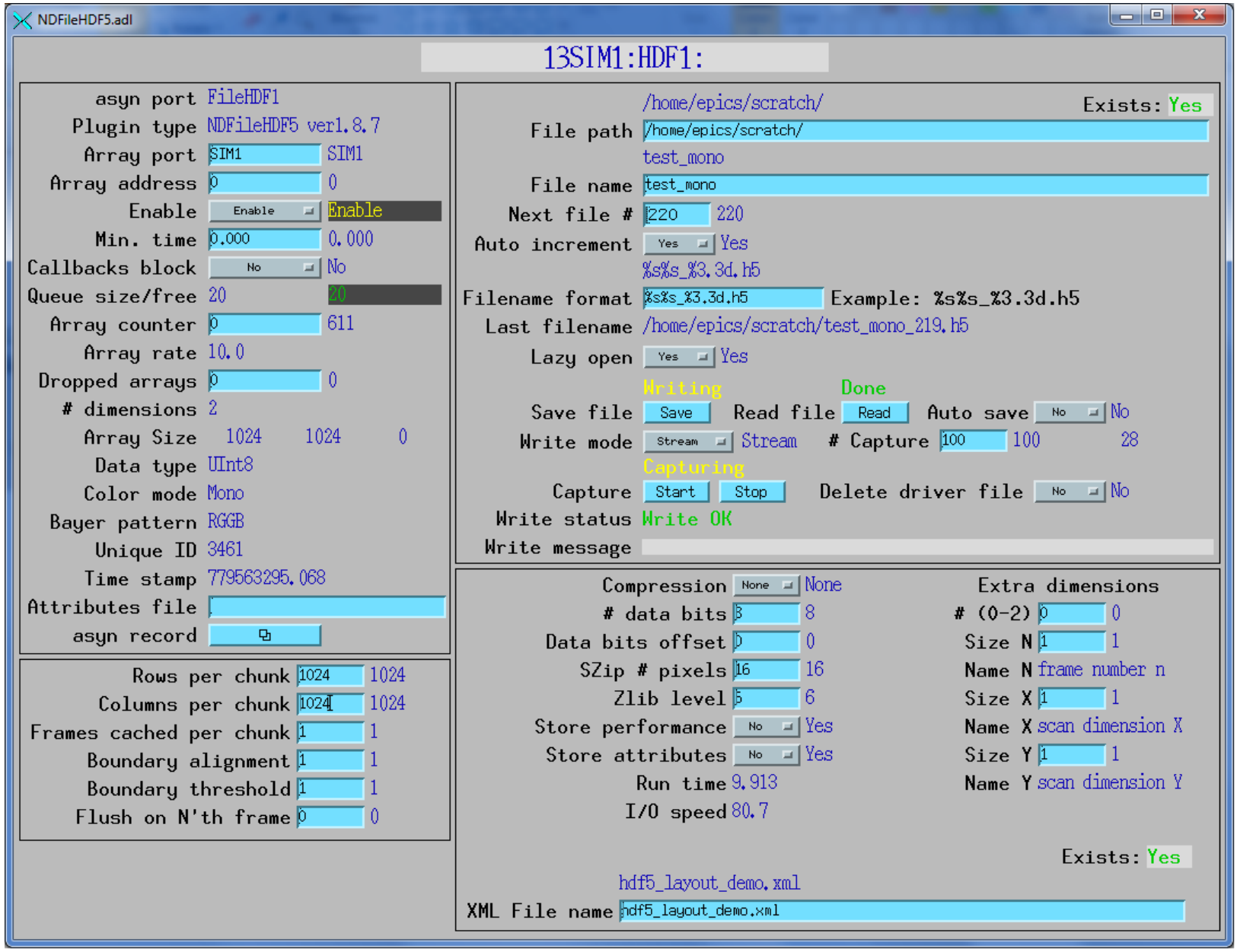
Example: saving 82 frames/second of 1024x1024 video to TIFF files, a few dropped frames.

NDPluginFile display: netCDF



Example: streaming 47 frames/second of 1024x1024 video to netCDF files, no dropped frames.

NDFileHDF5


NDFileHDF5.adl
13SIM1:HDF1:
asyn port FileHDF1
Plugin type NDFileHDF5 ver1.8.7
Array port SIM1 SIM1
Array address 0 0
Enable Enable Enable
Min. time 0.000 0.000
Callbacks block No No
Queue size/free 20 20
Array counter 0 611
Array rate 10.0
Dropped arrays 0 0
dimensions 2
Array Size 1024 1024 0
Data type UInt8
Color mode Mono
Bayer pattern RGGB
Unique ID 3461
Time stamp 779563295.068
Attributes file
asyn record
Rows per chunk 1024 1024
Columns per chunk 1024 1024
Frames cached per chunk 1 1
Boundary alignment 1 1
Boundary threshold 1 1
Flush on N'th frame 0 0
/home/epics/scratch/ Exists: Yes
File path /home/epics/scratch/
test_mono
File name test_mono
Next file # 220 220
Auto increment Yes Yes
%s%s_%3.3d.h5
Filename format %s%s_%3.3d.h5 Example: %s%s_%3.3d.h5
Last filename /home/epics/scratch/test_mono_219.h5
Lazy open Yes Yes
Writing Done
Save file Save Read file Read Auto save No No
Write mode Stream Stream # Capture 100 100 28
Capturing
Capture Start Stop Delete driver file No No
Write status Write OK
Write message
Compression None None
data bits 8 8
Data bits offset 0 0
SZip # pixels 16 16
Zlib level 6 6
Store performance No Yes
Store attributes No Yes
Run time 9.913
I/O speed 80.7
Extra dimensions
(0-2) 0 0
Size N 1 1
Name N frame number n
Size X 1 1
Name X scan dimension X
Size Y 1 1
Name Y scan dimension Y
Exists: Yes
hdf5_layout_demo.xml
XML File name hdf5_layout_demo.xml

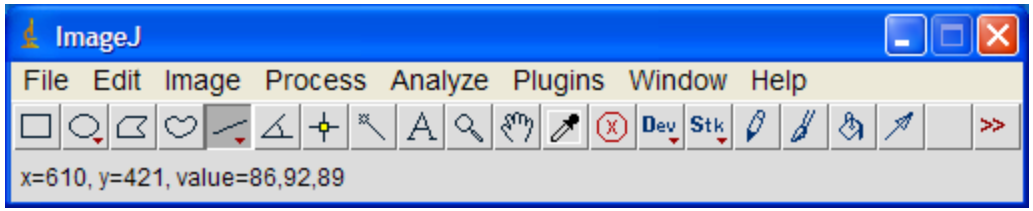
NDFileHDF5
XML file to define file layout

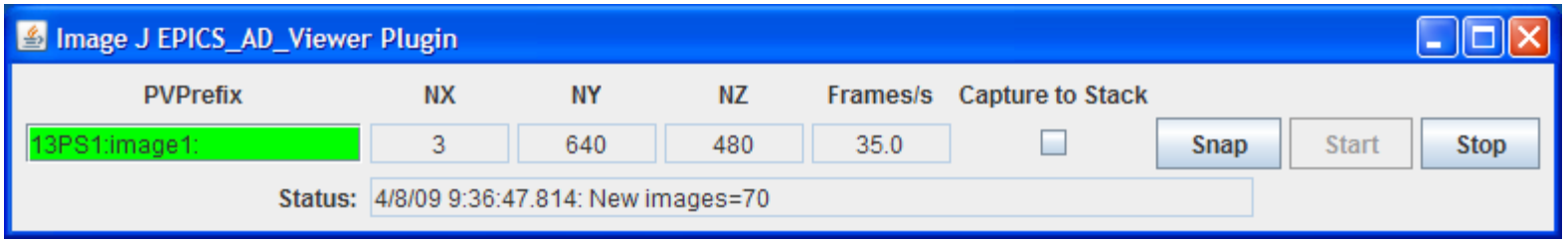
```
<xml>
  <group name="entry">
    <attribute name="NX_class" source="constant" value="NXentry" type="string"></attribute>
    <group name="instrument">
      <attribute name="NX_class" source="constant" value="NXinstrument" type="string"></attribute>
      <group name="detector">
        <attribute name="NX_class" source="constant" value="NXdetector" type="string"></attribute>
        <dataset name="data" source="detector" det_default="true">
          <attribute name="NX_class" source="constant" value="SDS" type="string"></attribute>
          <attribute name="signal" source="constant" value="1" type="int"></attribute>
          <attribute name="target" source="constant" value="/entry/instrument/detector/data"
                     type="string"></attribute>
        </dataset>
        <group name="NDAttributes">
          <attribute name="NX_class" source="constant" value="NXcollection" type="string"></attribute>
          <dataset name="ColorMode" source="ndattribute" ndattribute="ColorMode">
          </dataset>
        </group>          <!-- end group NDAttribute -->
      </group>            <!-- end group detector -->
      <group name="NDAttributes" ndattr_default="true">
        <attribute name="NX_class" source="constant" value="NXcollection" type="string"></attribute>
      </group>            <!-- end group NDAttribute (default) -->
      <group name="performance">
        <dataset name="timestamp" source="ndattribute"></dataset>
      </group>            <!-- end group performance -->
    </group>              <!-- end group instrument -->
    <group name="data">
      <attribute name="NX_class" source="constant" value="NXdata" type="string"></attribute>
      <hardlink name="data" target="/entry/instrument/detector/data"></hardlink>
      <!-- The "target" attribute in /entry/instrument/detector/data is used to
           tell Nexus utilities that this is a hardlink -->
    </group>              <!-- end group data -->
  </group>                <!-- end group entry -->
</xml>
```

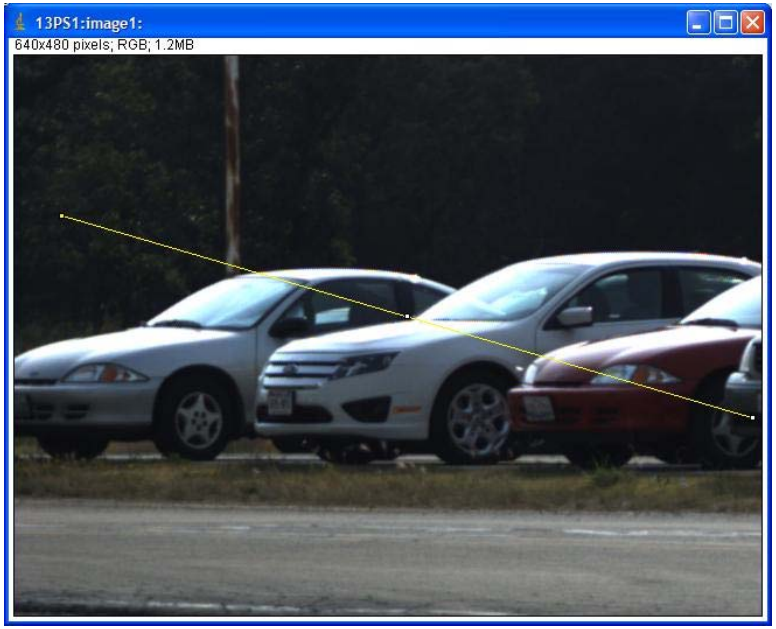
Viewers

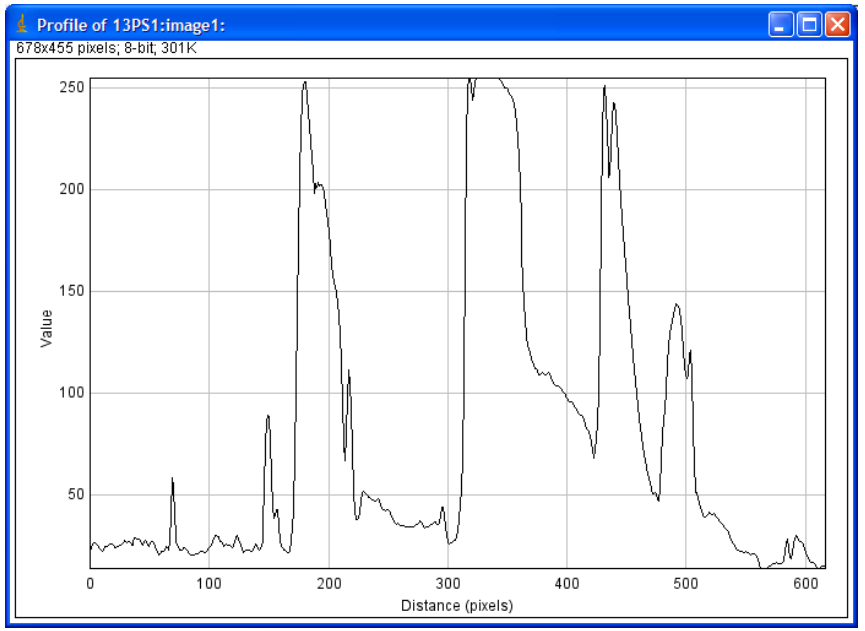
- areaDetector allows generic viewers to be written that receive images as EPICS waveform records over Channel Access
- Current viewers include:
  - ImageJ plugin EPICS_AD_Display. ImageJ is a very popular image analysis program, written in Java, derived from NIH Image.
  - IDL EPICS_AD_Display.
  - ffmpegServer allows image display in any Web browser
  - ffmpegViewer high-performance Qt-based viewer for MJPEG stream

ImageJ Viewer



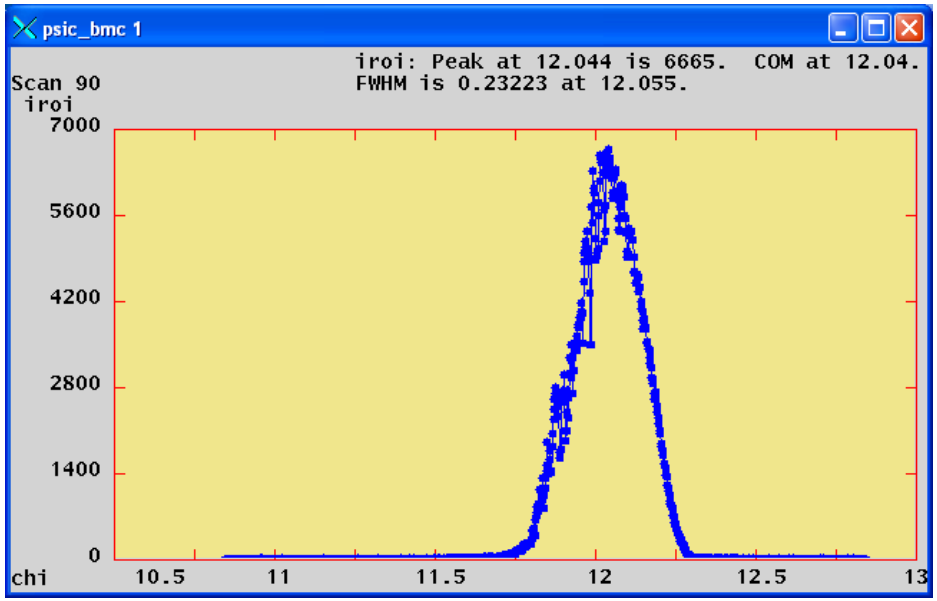




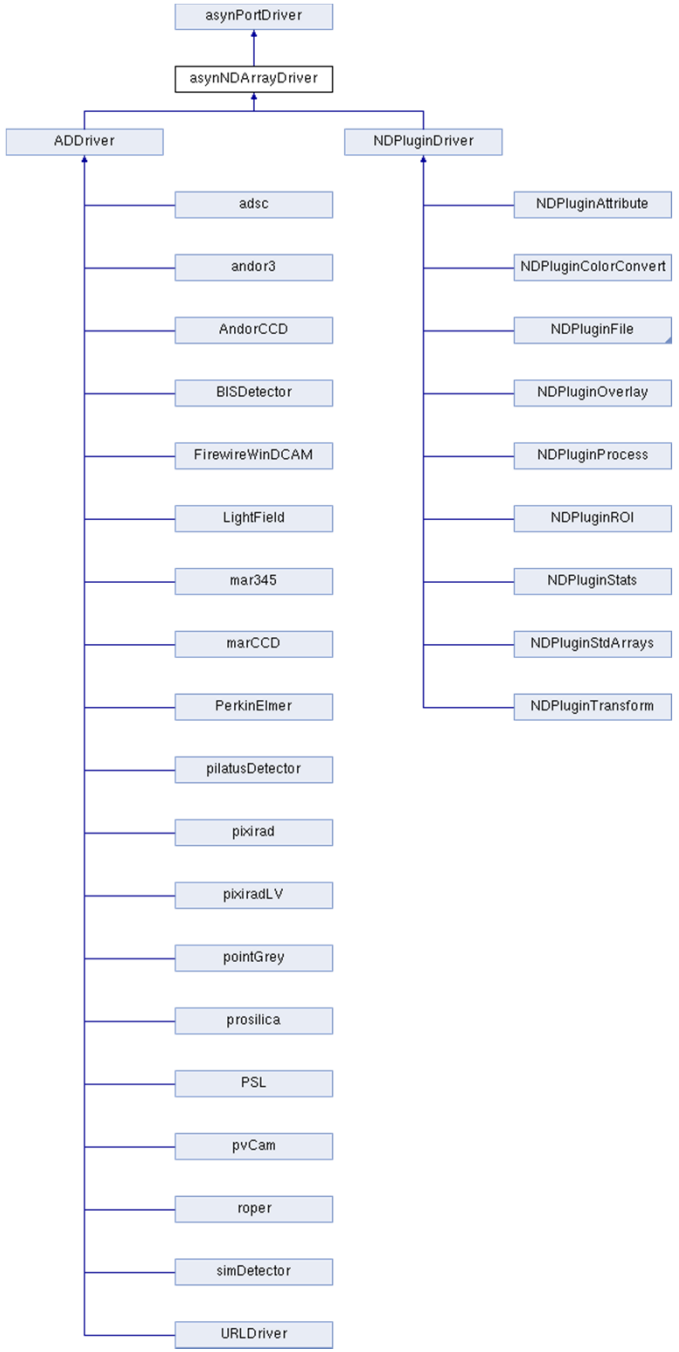


Performance Example with Pilatus driver

- SPEC used to collect 1000 points using trajectory scanning mode with the Newport XPS motor controller. Hardware trigger of Pilatus from XPS.
- Relative scan of the chi axis from -2 degrees to +2 degrees with 1000 points at .02 seconds/point
- Coordinated motion of the phi, kappa and omega axes.
- Theoretical time 20.0 second, actual time 20.8 seconds
- Includes time to save all 1000 images to disk (366 MB), Pilatus driver to read each file, correct bad pixels and flat field, compute ROIs, and post the ROIs and 1000 images to EPICS.



Internals
Class hierarchy



ADCore R3-0

- Simplify NDPluginFile base class and way file saving works
  - Remove the Single/Stream/Capture mode.
- Two parameters
  - # NDArrays to save (already present)
  - # NDArrays per file (new)
  - This allows saving only 1 array per HDF5 file, which is not possible now in Stream mode.
- Capture mode can be replaced:
  - Make input queue large enough OR
  - Use new NDPluginCircularBuffer
- Will require modifying clients that are doing file saving, hence a major release number

Future Ideas

- Put more functionality into ADDriver base class
  - Currently it does not do much, all code is in each driver for:
    - Doing callbacks to plugins
    - Processing new exposure time with writeFloat64 function
  - writeFloat64 in ADDriver base class would call setExposure() in derived class
  - Derived class would call ADDriver::doPluginCallbacks(), which would handle setting attributes, getting timestamp, calling plugins, etc.
- This is the way the Model 3 motor driver, which also uses asynPortDriver, is written
- Demultiplexor/multiplexor plugin
  - Allow multiple plugins to work on the same data stream when it saturates a single core

Conclusions

- Architecture works well, easily extended to new detector drivers, new plugins and new clients
- Base classes, asynPortDriver, asynNDArrayDriver, asynPluginDriver actually are generic, nothing “areaDetector” specific about them.
- They can be used to implement any N-dimension detector, e.g. the XIA xMAP (16 detectors x 2048 channels x 512 points in a scan line)
- Can get documentation and pre-built binaries (Linux, Windows, Cygwin) from our Web site:
  - http://cars.uchicago.edu/software/epics/areaDetector
- Can get code from github
  - https://github.com/areaDetector

Acknowledgments

- Brian Tieman, John Hammonds (APS) Perkin-Elmer driver, NeXus file saving plugin, Roper PVCAM driver
- Tim Madden (APS) initial version of ImageJ viewer, PCO drivers
- Ulrik Pedersen, Matt Pearson, Tom Cobb, David Hickin, Alan Greer (Diamond and Observatory Sciences) ffmpeg plugin, aravisGigE driver, Linux Firewire driver, HDF5 plugin
- Bruno Martins (BNL) V4 plugin and driver
- Lewis Muir (APS IMCA CAT) ADSC CCD driver
- Chris Roehrig (APS) new NDPluginTransform
- Many others for enhancements and bug fixes
- NSF-EAR and DOE-Geosciences for support of GSECARS where most of this work was done