File Formats for N-Dimensional Detector Data

Mark Rivers

GeoSoilEnviroCARS, Advanced Photon Source

University of Chicago

Talk Outline

- areaDetector module
  - Overview of architecture
  - Drivers for detectors & cameras
  - Plugins for real-time processing
  - Plugins for file saving (netCDF, Nexus/HDF, others)
  - Viewers and other clients
- Other applications
  - MCA (ASCII, HDF)
  - Tomography (netCDF, HDF)
  - EPICS sscan/saveData (MDA)
- Demo
  - Prosilica GigE camera streaming video to disk in HDF and netCDF formats

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, Shanghai, NSLS, NSLS-II, 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 (SPEC, medm, IDL programs, Python scripts, etc)

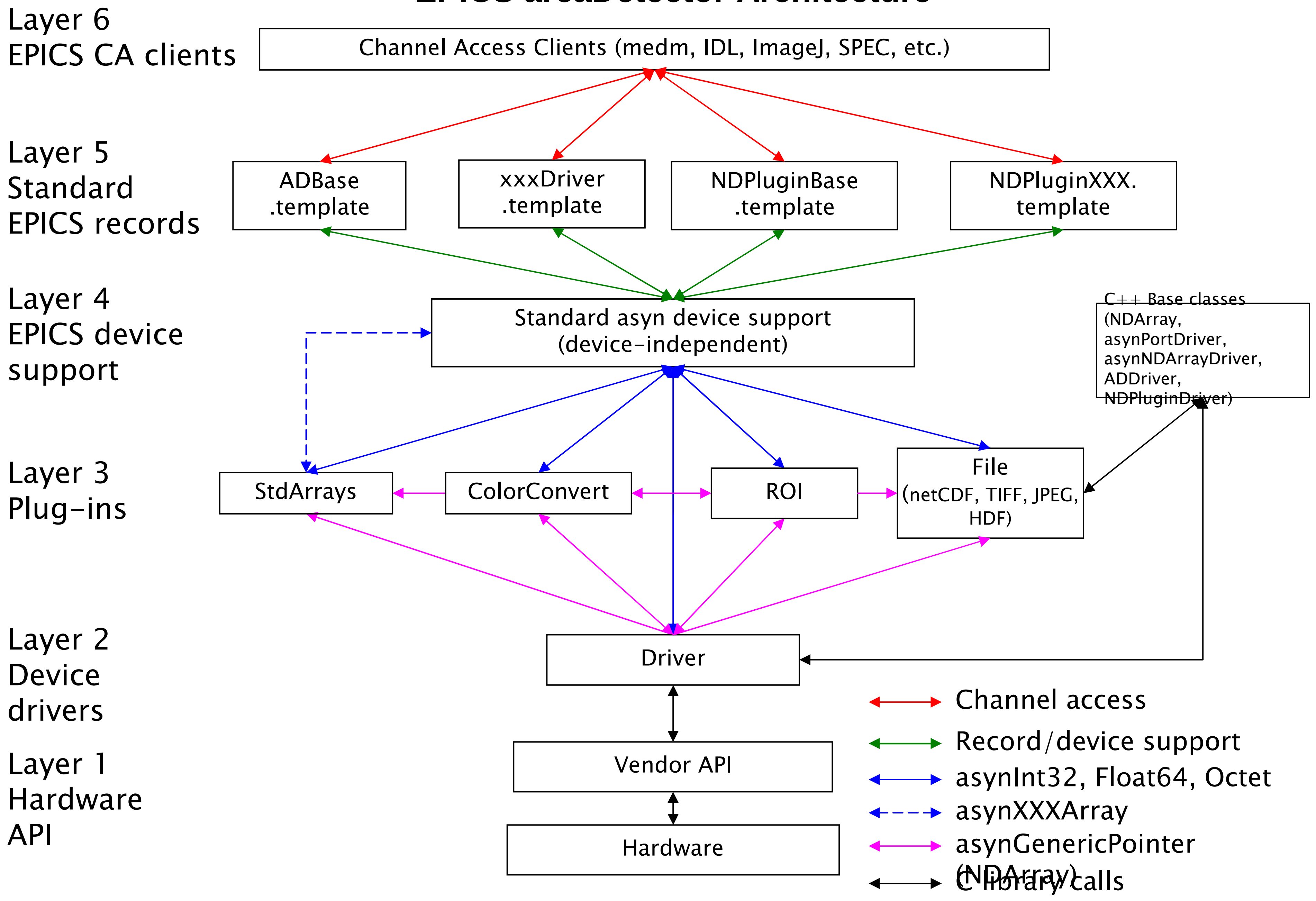
areaDetector - Goals

- 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

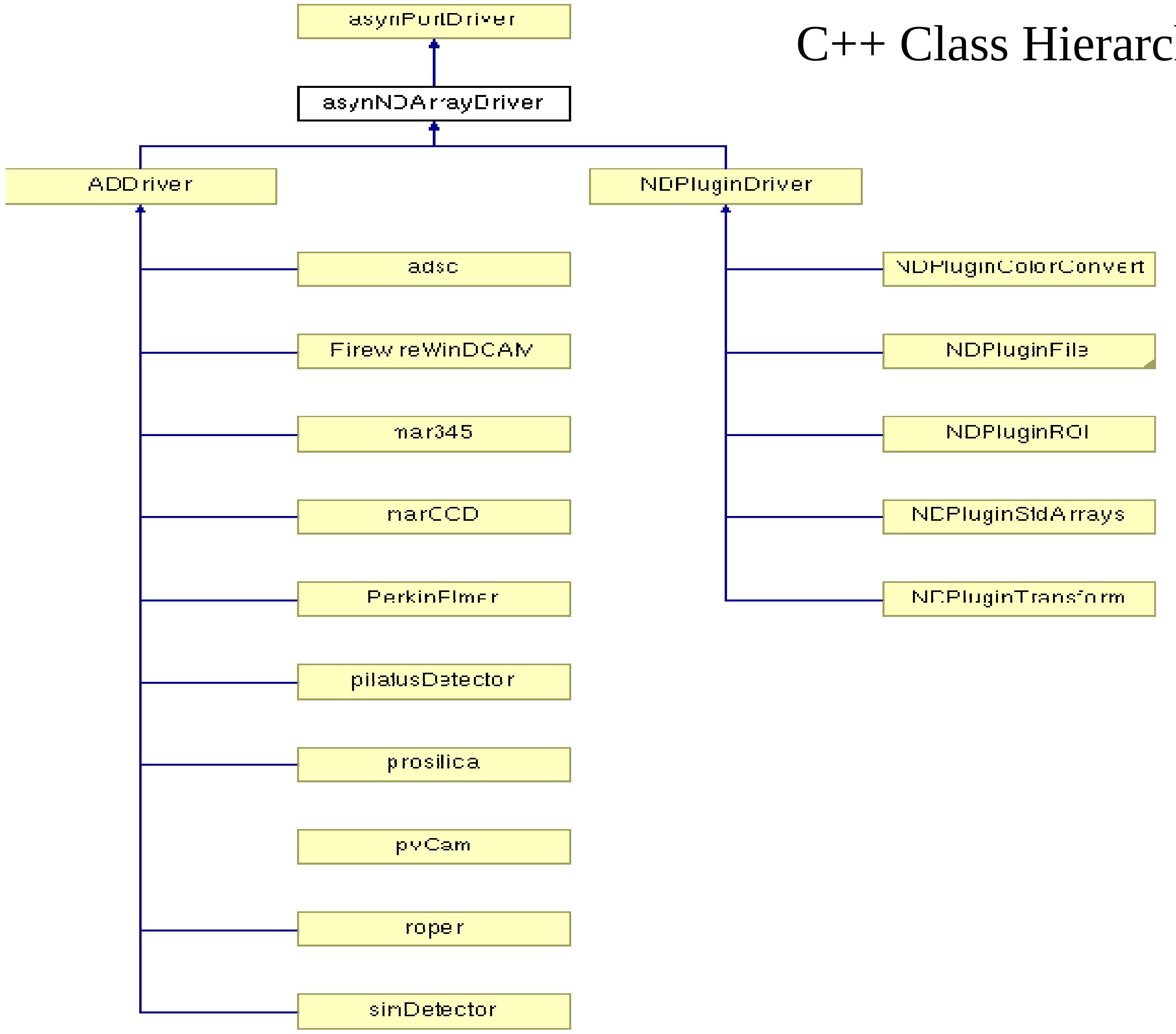
areaDetector – Data structures

- 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 or any EPICS PV; 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.
- Although written for 2-D detectors, these are completely applicable to spectroscopy systems (X, Y, E) or (E, X, Y)

EPICS areaDetector Architecture



C++ Class Hierarchy



Detector drivers

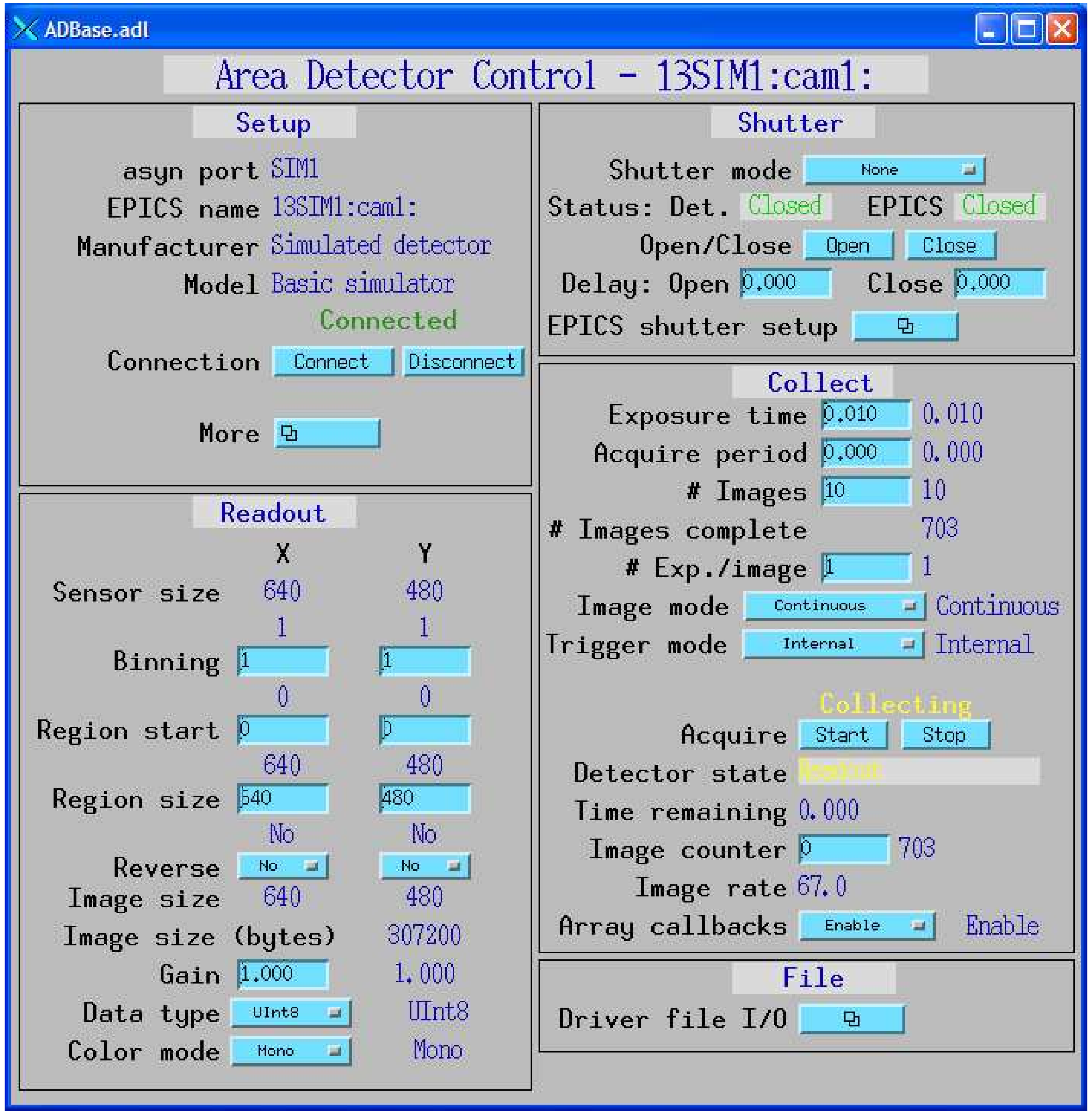
- ADDriver
  - Base C++ class from which detector drivers derive. Handles details of EPICS interfaces, and other common functions.
- Simulation driver
  - Produces calculated images up to very high rates. Implements nearly all basic parameters, including color. Useful as a model for real detector drivers,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

Detector drivers (continued)

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

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


ADBase.adl

Area Detector Control - 13SIM1:cam1:

Setup

asyn port	SIM1
EPICS name	13SIM1:cam1:
Manufacturer	Simulated detector
Model	Basic simulator
	Connected
Connection	Connect / Disconnect
More	

Readout

	X	Y
Sensor size	640	480
	1	1
Binning	1	1
	0	0
Region start	0	0
	640	480
Region size	640	480
	No	No
Reverse	No	No
Image size	640	480
Image size (bytes)		307200
Gain	1.000	1.000
Data type	UInt8	UInt8
Color mode	Mono	Mono

Shutter

Shutter mode	None		
Status: Det.	Closed	EPICS	Closed
Open/Close	Open	Close	
Delay: Open	0.000	Close	0.000
EPICS shutter setup			

Collect

Exposure time	0.010	0.010
Acquire period	0.000	0.000
# Images	10	10
# Images complete		703
# Exp./image	1	1
Image mode	Continuous	Continuous
Trigger mode	Internal	Internal
	Collecting	
Acquire	Start / Stop	
Detector state	Readout	
Time remaining	0.000	
Image counter	0	703
Image rate	67.0	
Array callbacks	Enable	Enable

File

Driver file I/O

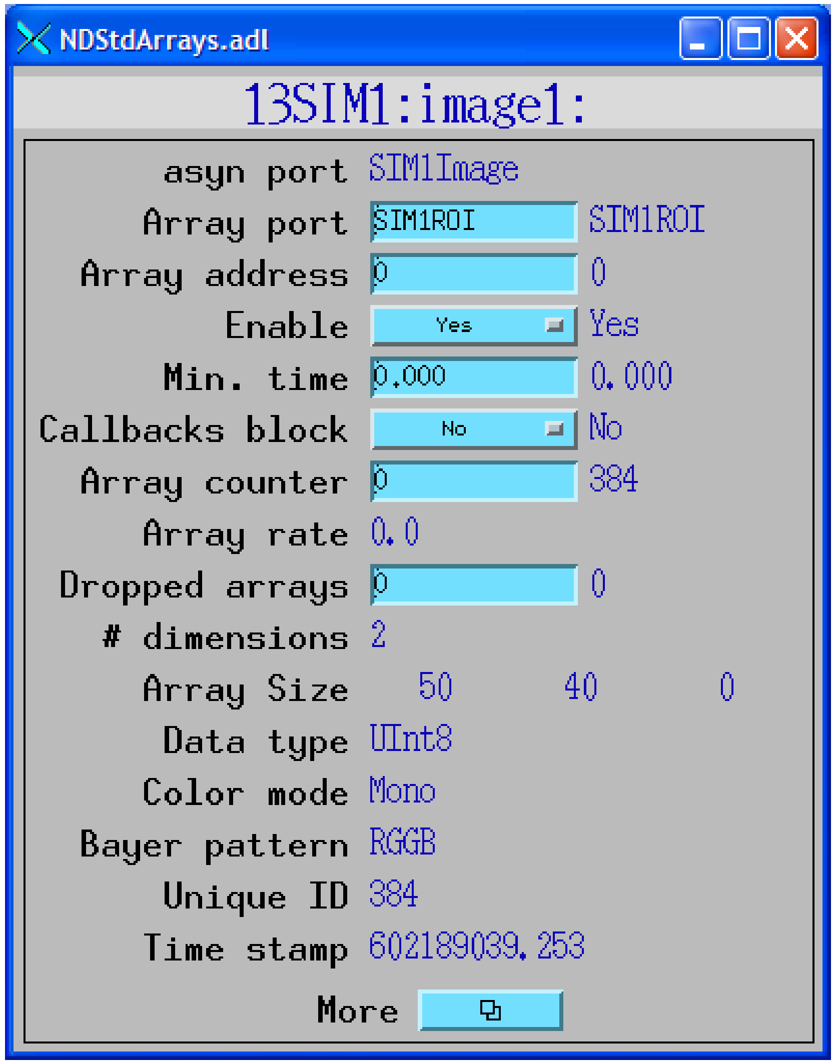
Plugins

- 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 difference thread, and hence in multiple cores on modern CPUs.

Plugins (continued)

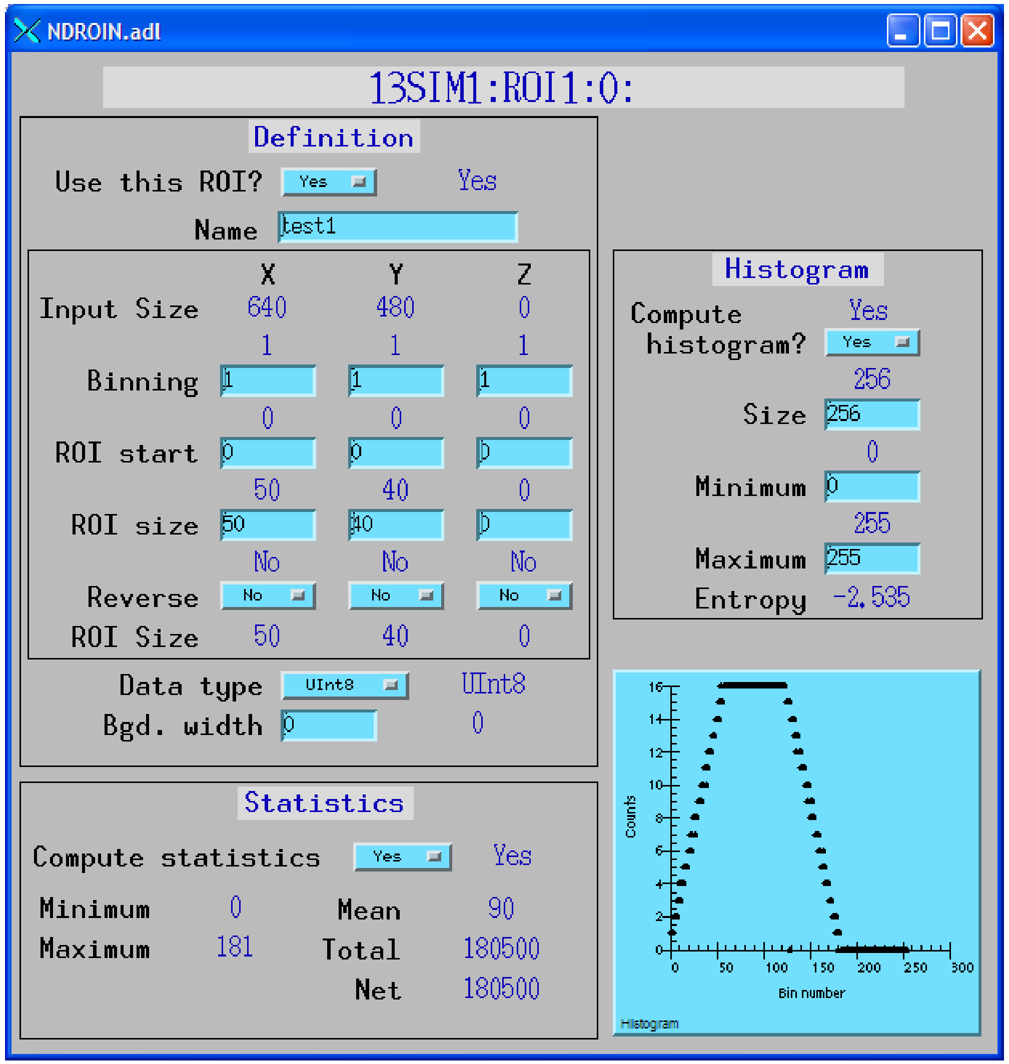
- NDPlugInStdArrays
  - Receives arrays (images) from device drivers, converts to standard arrays, e.g. waveform records.
  - This plugin is what EPICS channel access viewers normally talk to.
- NDPluginROI
  - Performs region-of-interest calculations
  - Select a subregion. Optionally bin, reverse in either direction, convert data type.
  - Optionally compute statistics on the region (min, max, mean, etc.)
  - Optionally compute histogram of the region, make available as waveform for client plotting.
- NDPluginColorConvert
  - Convert from one color model to another (Bayer, RGB pixel, row or planar interleave)
- NDPluginMJPEG
  - MJPEG server that allows viewing images in a Web browser.

StdArrays and ROI plugin displays


NDStdArrays.adl

13SIM1:image1:

asyn port	SIM1Image		
Array port	SIM1ROI	SIM1ROI	
Array address	0	0	
Enable	Yes	Yes	
Min. time	0.000	0.000	
Callbacks block	No	No	
Array counter	0	384	
Array rate	0.0		
Dropped arrays	0	0	
# dimensions	2		
Array Size	50	40	0
Data type	UInt8		
Color mode	Mono		
Bayer pattern	RGGB		
Unique ID	384		
Time stamp	602189039.253		


NDROIN.adl

13SIM1:ROI1:0:

Definition

Use this ROI?	Yes	Yes
Name	test1	

	X	Y	Z
Input Size	640	480	0
Binning	1	1	1
	1	1	1
ROI start	0	0	0
	0	0	0
ROI size	50	40	0
	50	40	0
Reverse	No	No	No
	No	No	No
ROI Size	50	40	0

Data type	UInt8	UInt8
Bgd. width	0	0

Statistics

Compute statistics	Yes	Yes	
Minimum	0	Mean	90
Maximum	181	Total	180500
		Net	180500

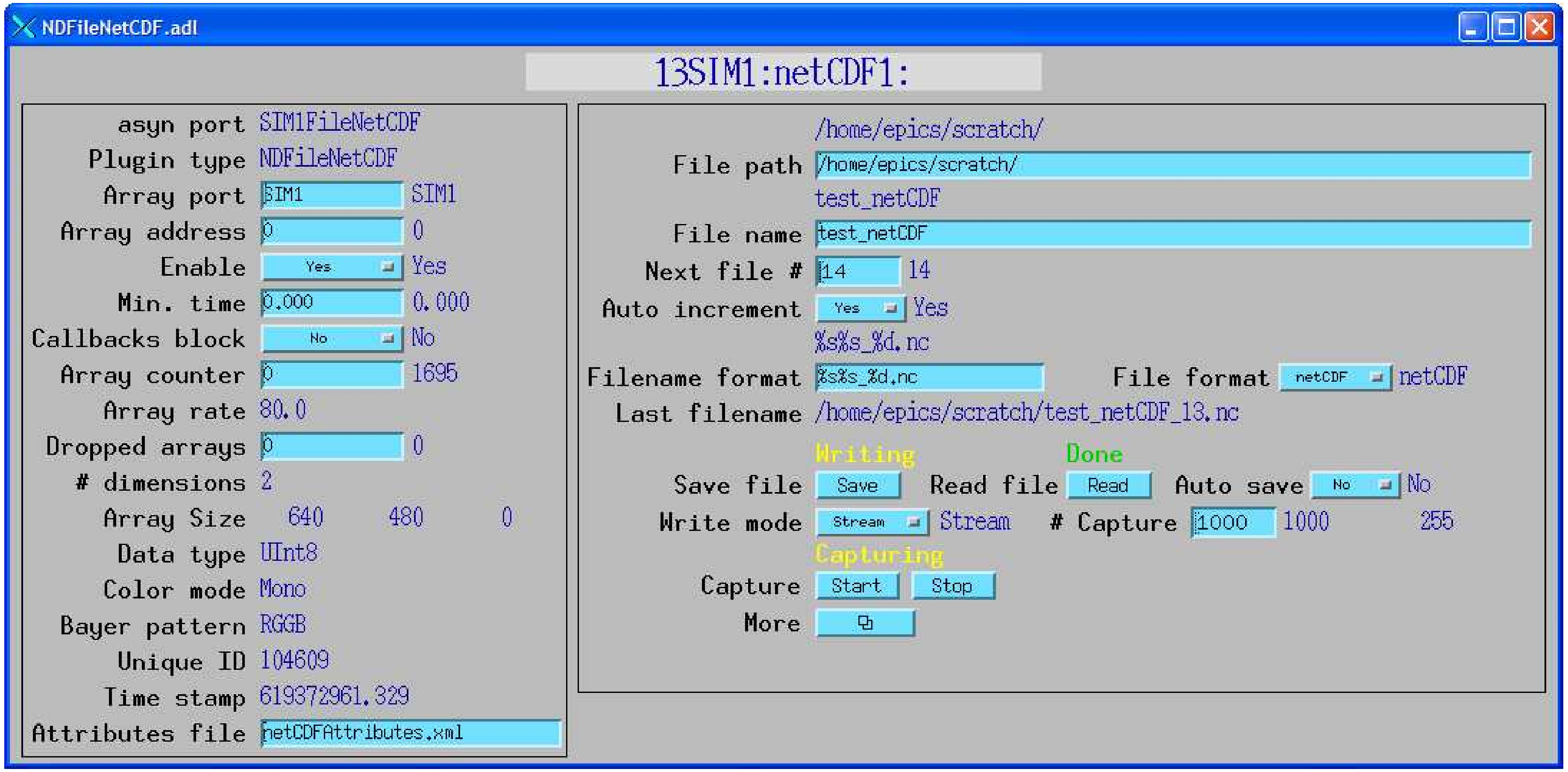
Histogram

Compute histogram?	Yes	Yes
Size	256	256
Minimum	0	0
Maximum	255	255
Entropy	-2.535	

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
- File formats currently supported
  - TIFF
  - JPEG (with compression control)
  - netCDF (popular self-describing binary format, supported by Unidata at UCAR)
  - NeXus (standard file format for neutron and x-ray communities, based on HDF, which is another popular self-describing binary format, richer than netCDF).
- 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



NDFileNetCDF.adl

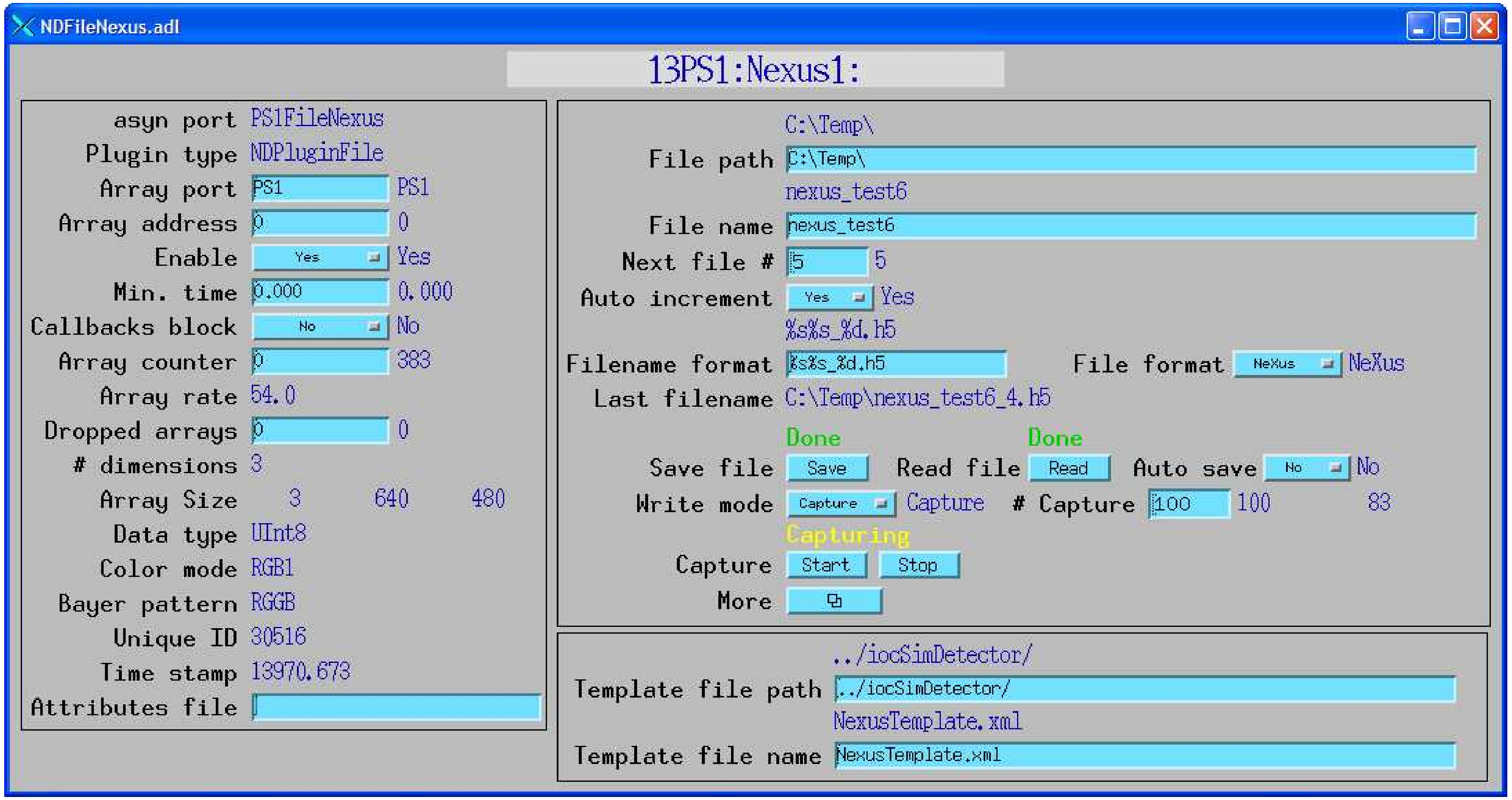
13SIM1:netCDF1:

asyn port	SIM1FileNetCDF	
Plugin type	NDFileNetCDF	
Array port	SIM1	SIM1
Array address	0	0
Enable	Yes	Yes
Min. time	0.000	0.000
Callbacks block	No	No
Array counter	0	1695
Array rate	80.0	
Dropped arrays	0	0
# dimensions	2	
Array Size	640 480 0	
Data type	UInt8	
Color mode	Mono	
Bayer pattern	RGGB	
Unique ID	104609	
Time stamp	619372961.329	
Attributes file	netCDFAttributes.xml	

File path	/home/epics/scratch/	/home/epics/scratch/
File name	test_netCDF	test_netCDF
Next file #	14	14
Auto increment	Yes	Yes
Filename format	%s%s_%d.nc	%s%s_%d.nc
File format	netCDF	netCDF
Last filename	/home/epics/scratch/test_netCDF_13.nc	
Save file	Save	Writing
Read file	Read	Done
Auto save	No	No
Write mode	Stream	Stream
# Capture	1000	1000 255
Capture	Start Stop	Capturing
More		

Example: streaming 80 frames/second of 640x480 mono video to netCDF file, no dropped frames.

NDPluginFile display



NDFileNexus.adl

13PS1:Nexus1:

asyn port	PS1FileNexus			
Plugin type	NDPluginFile			
Array port	PS1	PS1		
Array address	0	0		
Enable	Yes	Yes		
Min. time	0.000	0.000		
Callbacks block	No	No		
Array counter	0	383		
Array rate	54.0			
Dropped arrays	0	0		
# dimensions	3			
Array Size	3	640	480	
Data type	UInt8			
Color mode	RGB1			
Bayer pattern	RGGB			
Unique ID	30516			
Time stamp	13970.673			
Attributes file				

File path	C:\Temp\	C:\Temp\
File name	nexus_test6	nexus_test6
Next file #	5	5
Auto increment	Yes	Yes
Filename format	%s%s_%d.h5	%s%s_%d.h5
File format	NeXus	NeXus
Last filename	C:\Temp\nexus_test6_4.h5	
Save file	Save	Done
Read file	Read	Done
Auto save	No	No
Write mode	Capture	Capture
# Capture	100	100 83
Capture	Start Stop	Capturing
More		
Template file path	../iocSimDetector/	../iocSimDetector/
Template file name	NexusTemplate.xml	NexusTemplate.xml

Example: capturing 54 frames/second of 640x480 RGB video to Nexus/HDF file, no dropped frames.

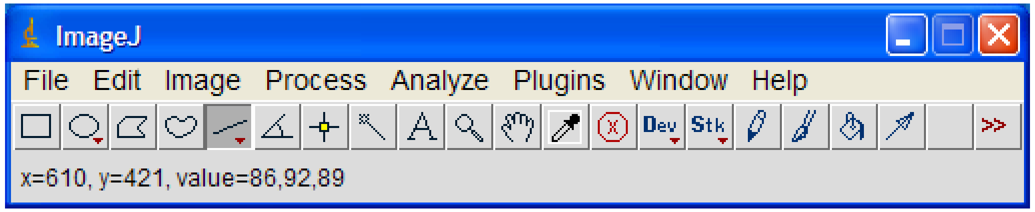
Viewers

- 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.
  - IDL areaVision (Stephen Mudie from AS)
  - MJPEG server allows image display in any Web browser

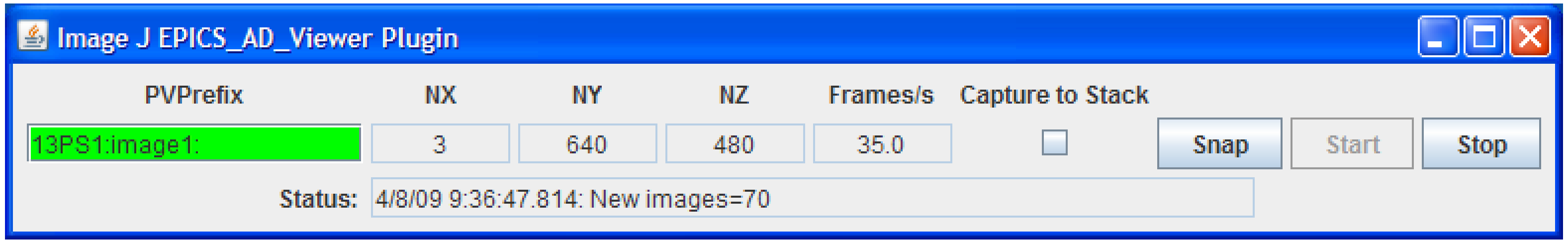
Driver and PluginAttributes Defined via User-Created XML File

```
 <?xml version="1.0" standalone="no" ?>
<Attributes>
  <Attribute name="AcquireTime"
             type="EPICS_PV"
             source="13PS1:cam1:AcquireTime"
             dbrtype="DBR_NATIVE“
             description="Camera acquire time" />
  <Attribute name="CameraModel"
             type="PARAM"
             source="MODEL“
             datatype="STRING"
             description="CameraModel" />
  <Attribute name="FramesDropped"
             type="PARAM"
             source="PS_FRAMES_DROPPED"
             datatype="INT"
             description="FramesDropped" />
</Attributes>
```

ImageJ Viewer

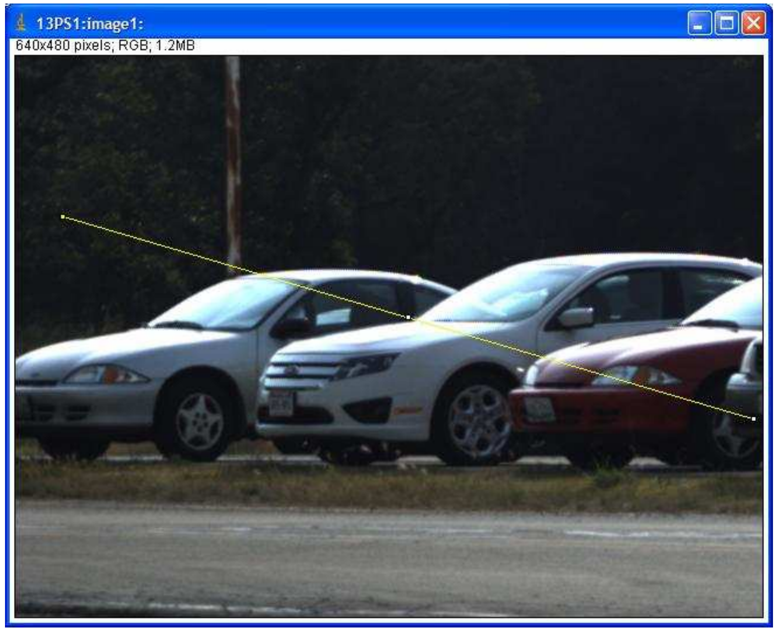


x=610, y=421, value=86,92,89

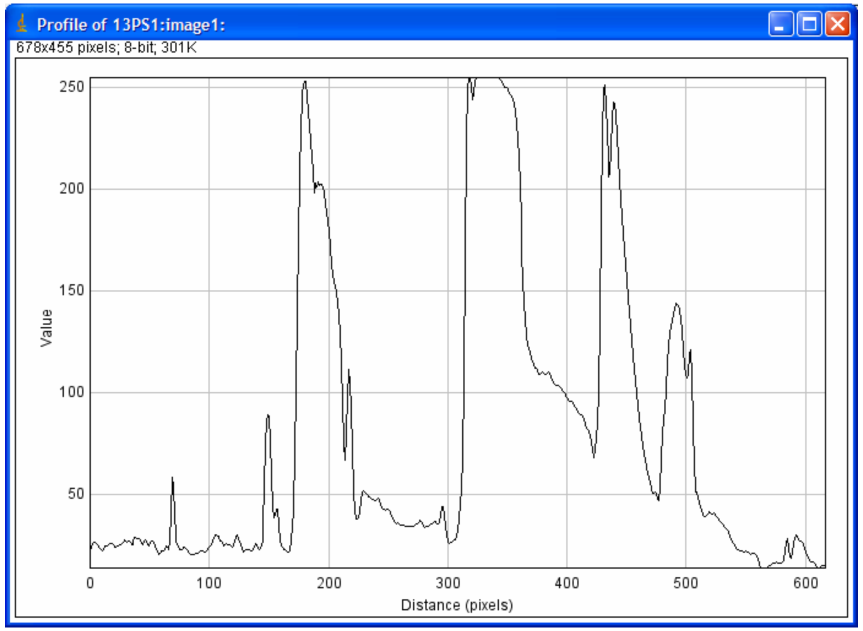


PVPrefix	NX	NY	NZ	Frames/s	Capture to Stack
13PS1:image1:	3	640	480	35.0	☐

Status: 4/8/09 9:36:47.814: New images=70



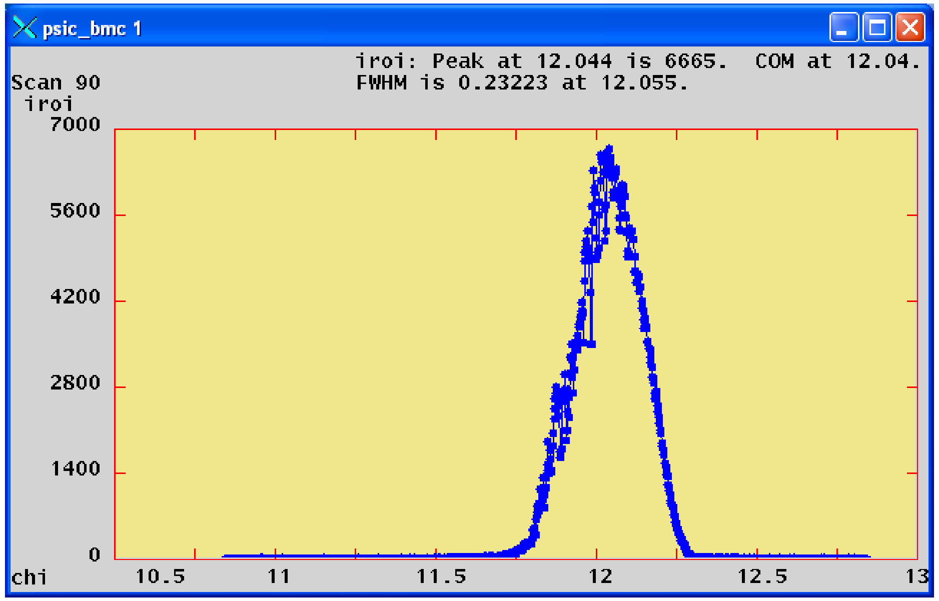
640x480 pixels; RGB; 1.2MB



678x455 pixels; 8-bit; 301K

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.



iroi: Peak at 12.044 is 6665. COM at 12.04.
FWHM is 0.23223 at 12.055.

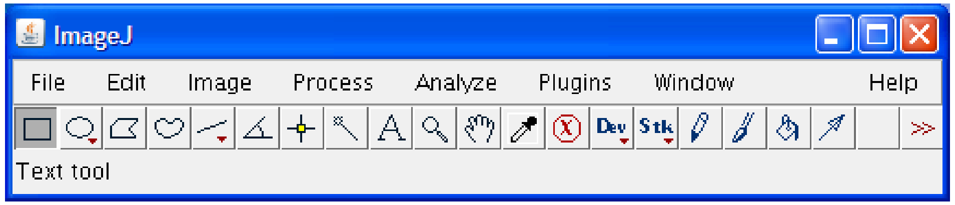
Tomography Data

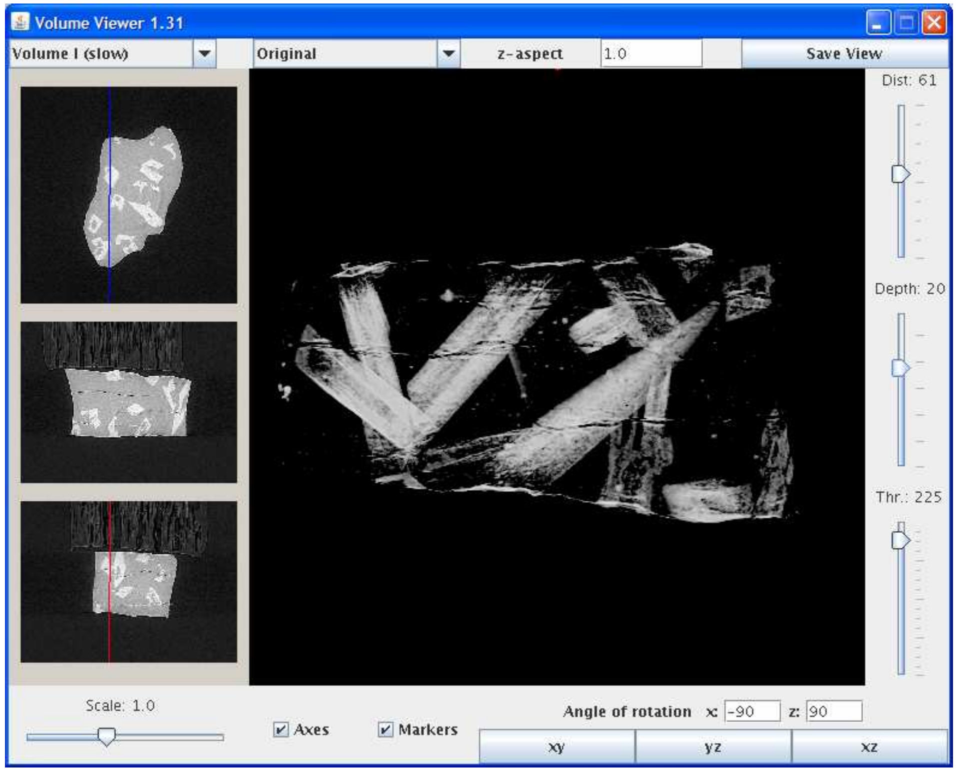
- Sector 13 at APS uses netCDF to store 3-D tomography data, both normalized raw data and reconstructed images
- Portable, self-describing, very well supported by Unidata at UCAR
  - We currently use netCDF “classic” (netCDF version 3)
  - netCDF version 4 uses HDF5 as underlying format, but with netCDF API
- Use vendor format (Princeton SPE) for raw data – not ideal

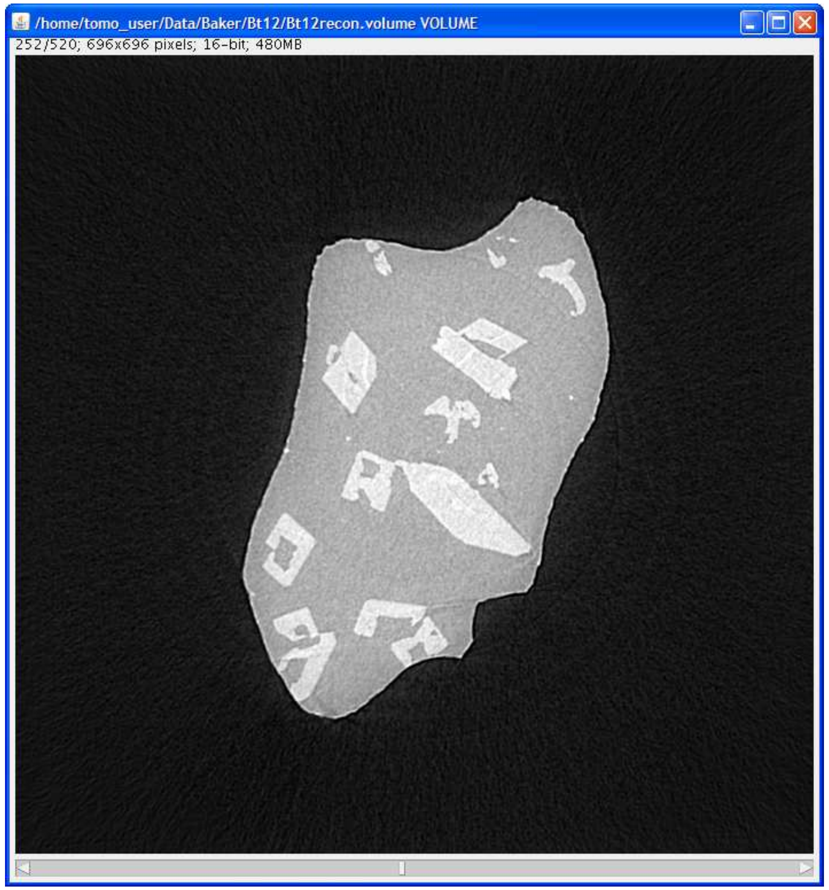
- Sector 2 at APS uses HDF4
  - Not satisfied with peformance relative to simple binary
  - Not happy with trouble users have with translating to more common formats
  - ImageJ requires C library to read, not available on Mac?
  - Considering abandoning HDF …

- Trivial to import netCDF data into ImageJ with netCDF plugin and 4 line script
- USERS LIKE THIS!

```
baja:~/Data/Baker/Bt12>more /usr/local/ImageJ/plugins/_GSETomo.txt
run("Load NetCDF File");
run("XOR...", "value=1000000000000000 stack");
run("Calibrate...", "function=[Straight Line] unit=[Gray Value] text1=[0 65535] text2=[-32768 32767 ]");
run("Enhance Contrast", "saturated=0.5");
```






/home/tomo_user/Data/Baker/Bt12/Bt12recon.volume VOLUME
252/520; 696x696 pixels; 16-bit; 480MB

Tomography Data

```
baja:~/Data/Baker/Bt12>/usr/local/netcdf-3.6.3/ncdump/ncdump -h Bt12recon.volume
netcdf Bt12recon {
dimensions:
        NX = 696 ;
        NY = 696 ;
        NZ = 520 ;
variables:
        short VOLUME(NZ, NY, NX) ;
                VOLUME:scale_factor = 2.e-06f ;

// global attributes:
                :title = "Before heating" ;
                :operator = "Rivers, Bai" ;
                :camera = "CoolSnap, 5x objective, 65mm tube" ;
                :sample = "Bt12" ;
                :image_type = "RECONSTRUCTED" ;
                :energy = 25.f ;
                :dark_current = 101.f ;
                :center = 342.701f ;
                :x_pixel_size = 5.22f ;
                :y_pixel_size = 5.22f ;
                :z_pixel_size = 0.f ;
                :angles = 0.f, 0.25f, 0.5f, 0.75f, 1.f, 1.25f, 1.5f, 1.75f, 2.f, 2.25f, 2.5f, 2.75f, 3.f, 3.25f, 3.5f, 3.75f, 4.f, 4.25f,
    4.5f, 4.75f, 5.f, 5.25f, 5.5f, 5.75f, 6.f, 6.25f, 6.5f, 6.75f, 7.f, 7.25f, 7.5f, 7.75f, 8.f, 8.25f,…
```

MDA files

Binary, portable (XDR)

Non-standard, not well documented

Commonly used at APS for general scan data (sscan record)

Metadata is not well organized, requires setup by user to capture everything required for interpretation (MCA calibration, etc.)

Should be replaced – not too difficult

IDL and Python browsers and visualization tools available

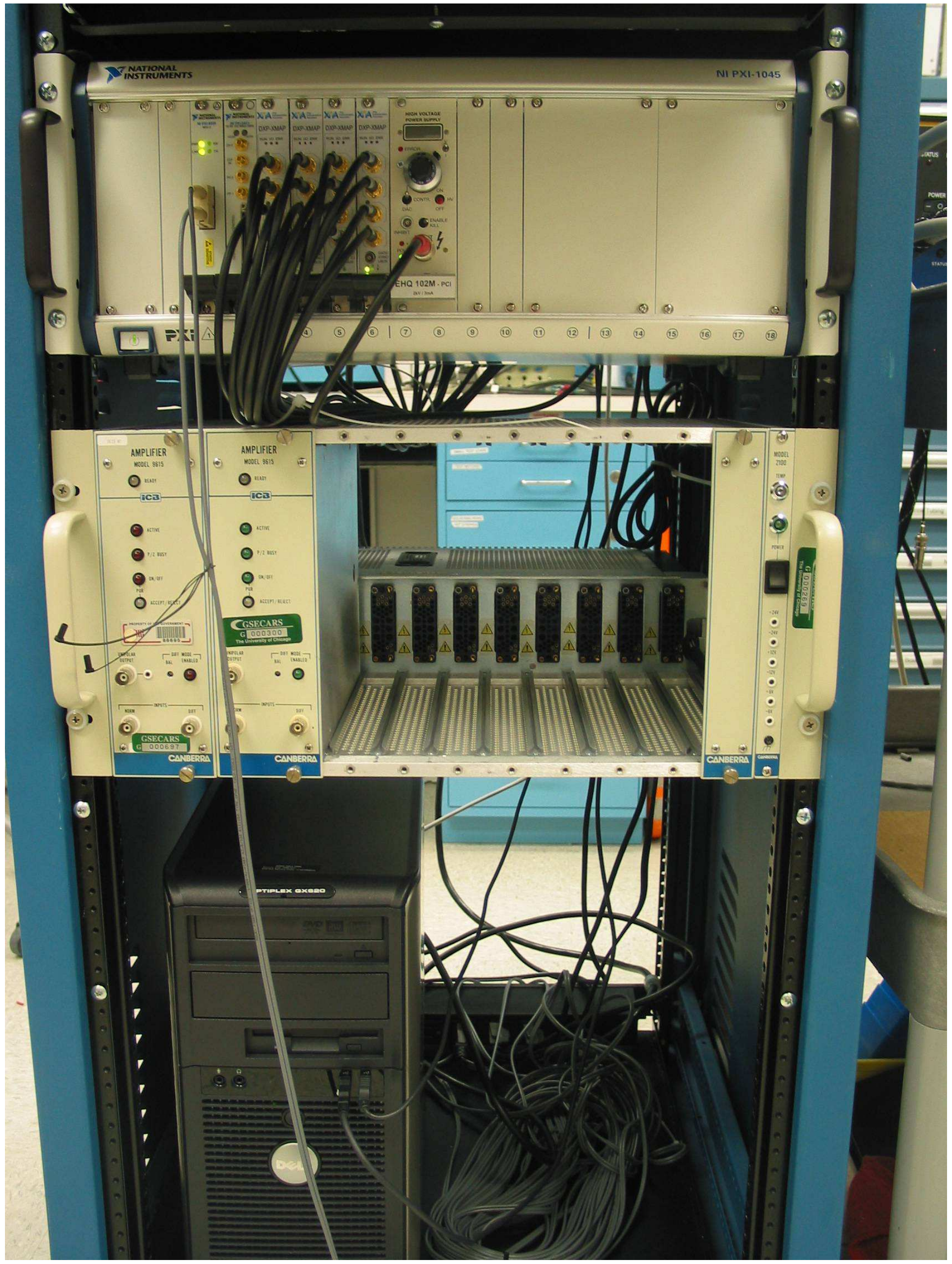
HDF4 was originally tried as file format but abandoned for poor performance – use HDF5?

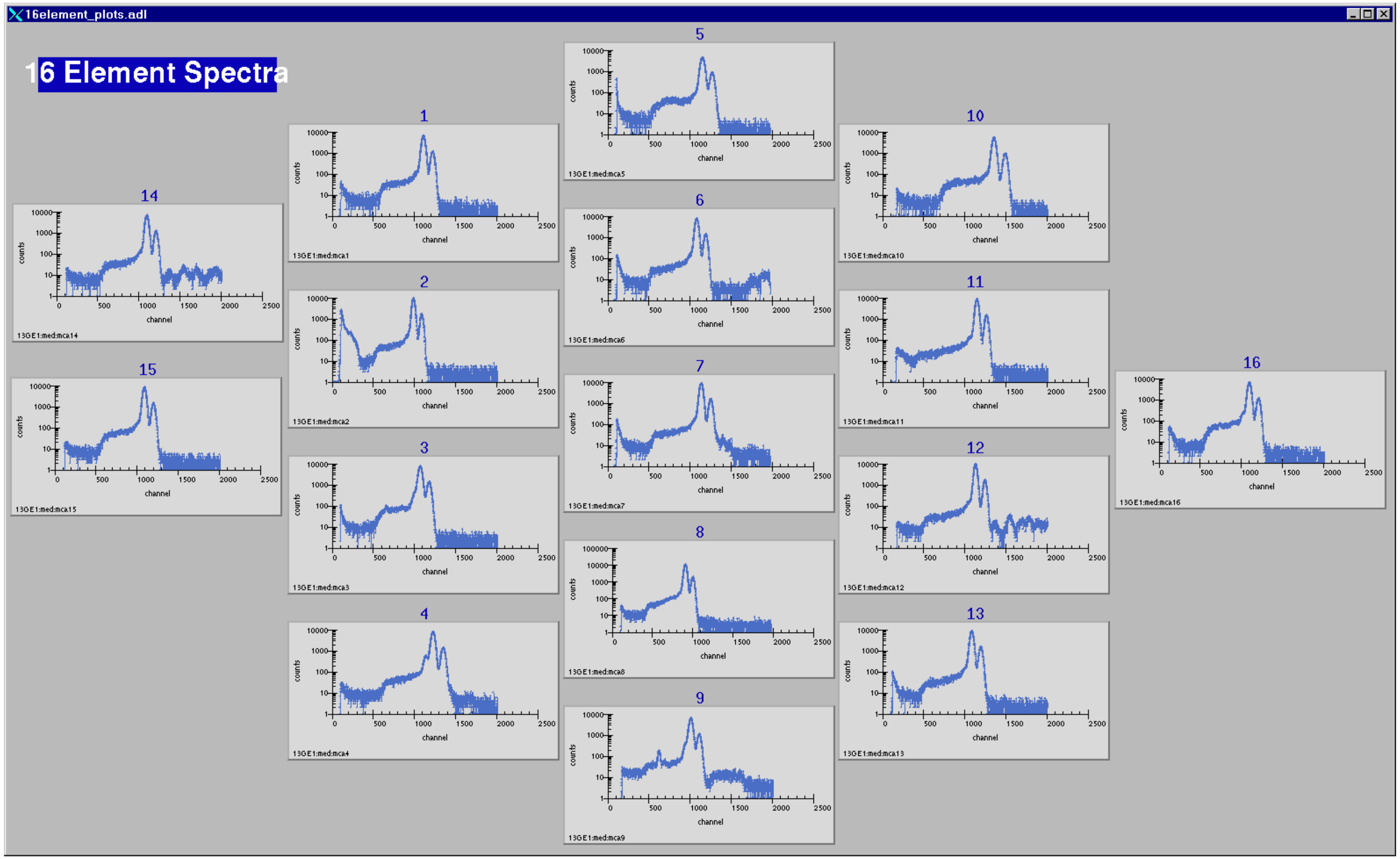
MCA files – legacy, ASCII (easy for users)

```
VERSION:    3.1
ELEMENTS:            16
DATE:       APR 19, 2009 20:00:00.101
CHANNELS:           2048
ROIS:       12 12 12 12 12 12 12 12 12 13 12 12 12 2 12 12
REAL_TIME:   300.0005188 300.0005188 300.0005188 300.0005188 300.0005188 300.0005188 300.0005188 300.0005188 300.0005188 300.0005188
         300.0005188 300.0005188 300.0005188 300.0005188 300.0005188 300.0005188
LIVE_TIME:   300.1702576 294.7783508 293.2825623 295.9739075 288.4481506 291.3387756 300.1826782 292.4120483 290.4937744 292.0257568
         293.1321106 292.1801147 290.0241699 300.1603699 294.2463989 292.7752991
CAL_OFFSET: -3.4999999e-001 -3.6593691e-002 -1.5307420e-001 5.1209070e-002 -1.2942410e-001 -7.9422943e-002 -1.6341200e-001 3.2899791e-004 -
         1.6128260e-001 -4.7699928e-002 -1.3841900e-001 -6.9478489e-002 -1.1549980e-001 0.0000000e+000 3.6675561e-002 -1.8902750e-001
CAL_SLOPE:   1.7000001e-002 1.7440230e-002 1.5586490e-002 1.6970981e-002 1.4973060e-002 1.5056320e-002 1.6039200e-002 1.8031770e-002 1.6510570e-
         002 1.5173850e-002 1.8110130e-002 1.4946350e-002 1.5717430e-002 2.2104000e+001 1.4986820e-002 1.5494210e-002
CAL_QUAD:    0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
         0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000 0.0000000e+000
TWO_THETA:   10.0000000 10.0000000 10.0000000 10.0000000 10.0000000 10.0000000 10.0000000 10.0000000 10.0000000 10.0000000 10.0000000
         10.0000000 10.0000000 10.0000000 10.0000000 10.0000000
ROI_0_LEFT:   226 202 233 202 241 237 228 193 221 233 200 238 229 1 230 237
ROI_0_RIGHT:   246 222 256 223 265 260 249 213 242 256 219 261 251 1 253 260
ROI_0_LABEL: Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Ca Ka &  Cd
         Ka &  Ca Ka &  Ca Ka &
ENVIRONMENT: S:SRcurrentAI.VAL="102.364" (Storage Ring Current)
ENVIRONMENT: 13BMA:m17.RBV="5.6729" (FOE Mono angle readback)
DATA:
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 732 2 0 13 0 0 0 0 1 15 2 32 0 0 1 3
 702 2 0 18 0 0 0 0 0 18 2 20 0 0 2 3
 790 1 0 18 1 0 0 0 0 10 3 8 0 0 4 0
 722 3 0 10 0 0 0 0 0 12 2 8 0 0 1 3
```

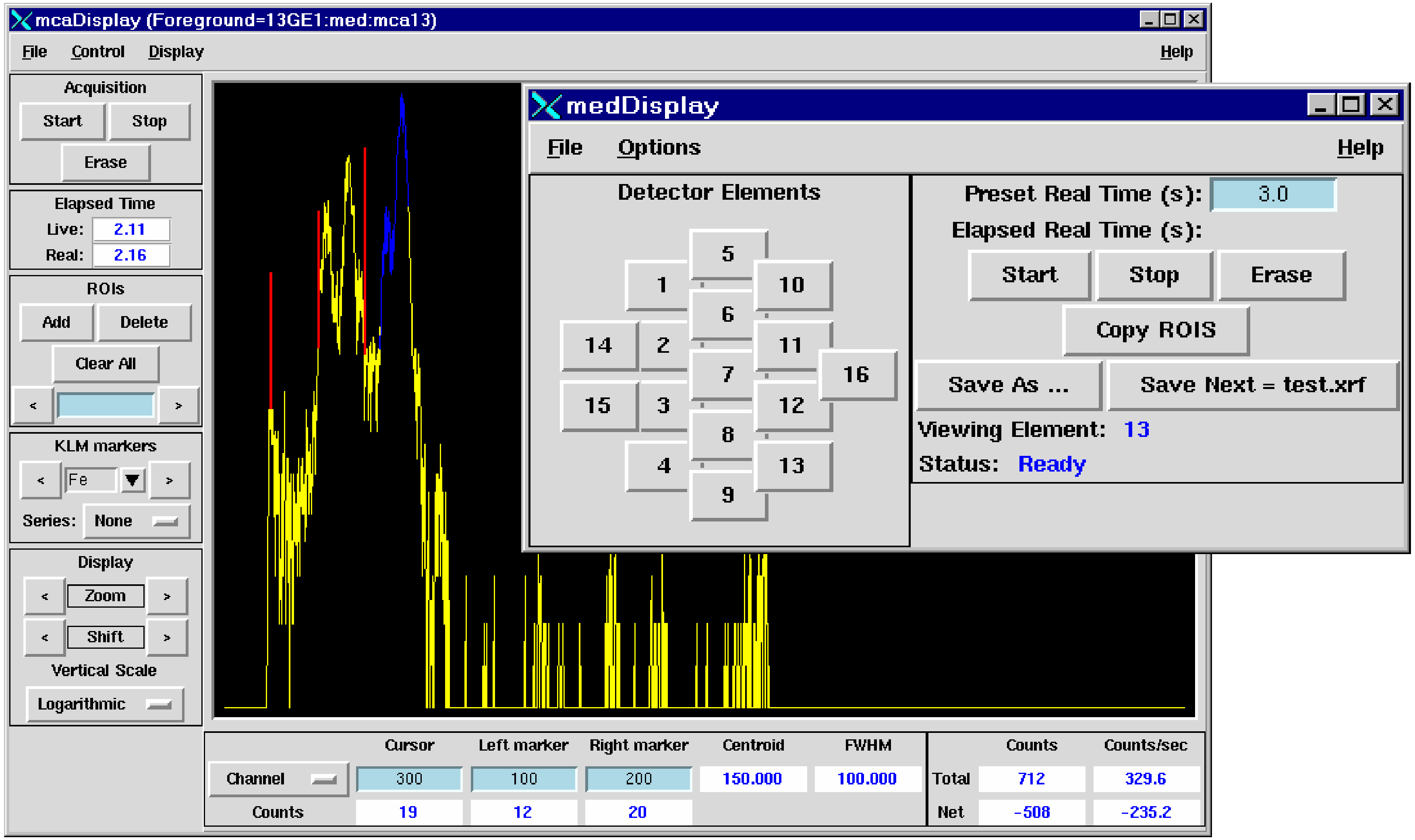
XIA Fast DSP Electronics for EDS Detectors

- X-MAP PXI detector electronics
- Large on-board memory for on-the-fly spectral data collection
- Almost completed driver in time for this meeting that uses asynNDArrayDriver class
- Can thus use any plugins, including HDF and netCDF file writers




16element_plots.adl
16 Element Spectra
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
counts
channel
13GE1:med:mca1
13GE1:med:mca2
13GE1:med:mca3
13GE1:med:mca4
13GE1:med:mca5
13GE1:med:mca6
13GE1:med:mca7
13GE1:med:mca8
13GE1:med:mca9
13GE1:med:mca10
13GE1:med:mca11
13GE1:med:mca12
13GE1:med:mca13
13GE1:med:mca14
13GE1:med:mca15
13GE1:med:mca16

medDisplay (IDL and Python)



Preset Real Time (s): 3.0
Elapsed Real Time (s):
Viewing Element: 13
Status: Ready

	Cursor	Left marker	Right marker	Centroid	FWHM
Channel	300	100	200	150.000	100.000
Counts	19	12	20		

	Counts	Counts/sec
Total	712	329.6
Net	-508	-235.2

Future directions

- Need to benchmark various formats: binary, netCDF, HDF5/Nexus for various applications. There are remaining doubts about performance, but little hard data I am aware of.
- Larger collaborations on workbench for data collection and analysis?

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 source code and pre-built binaries (Linux, Windows, Cygwin) from our Web site:
  – http://cars.uchicago.edu/software/epics/areaDetector
- Can also get code on SourceForge
  – http://epics.svn.sourceforge.net/viewvc/epics/applications/trunk/areaDetector

Acknowledgments

- Brian Tieman (APS) Roper PVCAM driver
- John Hammonds (APS) Perkin-Elmer driver and NeXus file saving plugin
- Tim Madden (APS) initial version of ImageJ viewer
- Stephen Mudie (Australian Synchrotron) areaViewer IDL-based viewer
- Ulrik Pedersen and Tom Cobb (Diamond) MJPEG plugin, Linux Firewire driver
- Lewis Muir (U Chicago) ADSC CCD driver
- NSF-EAR and DOE-Geosciences for support of GSECARS where most of this work was done

- Thanks for your attention!!!