

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
- Emphasis on what is new and plans for future
- Today: Morning lecture, afternoon demonstration
- Tomorrow: Hands-on practice with simulation or real detectors
- Next week: Writing software for a new detector or a new plugin

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)
- 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)
 - The client needs only to know how to talk to EPICS, not the details of the detector

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 database and Channel Access layers for highest performance

areaDetector – Data structures

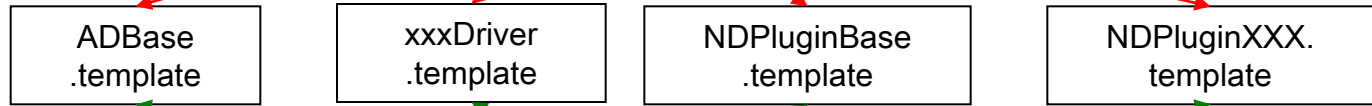
- NDArray
 - N-Dimensional array.
 - Everything is done in N-dimensions (up to 10), rather than 2. This is needed even for 2-D detectors to support color.
 - This is what plug-ins callbacks receive from device drivers.
- NDAttribute
 - Each NDArray has a list of associated attributes (metadata) that travel with the array through the processing pipeline. 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 plugins.

EPICS areaDetector Architecture

Layer 6
EPICS CA clients

Channel Access Clients (medm, IDL, ImageJ, SPEC, etc.)

Layer 5
Standard
EPICS records

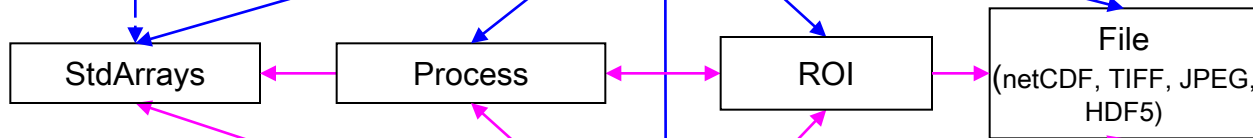


Layer 4
EPICS device
support

Standard asyn device support
(device-independent)

C++ Base classes
(NDArray, asynPortDriver,
asynNDArrayDriver,
ADDriver, NDPluginDriver)

Layer 3
Plug-ins



Layer 2
Device drivers

Driver

Layer 1
Hardware API

Vendor API

Hardware



areaDetector – Data structures

Look at NDArrary.h

Look at NDAttribute.h

Look at an XML attribute file

areaDetector R2-0 Release

- areaDetector was getting too big.
 - New releases being held up waiting for testing on one detector types, etc.
- Hard to collaborate with other sites using APS Subversion repository
 - git and github provide much better tools for multi-site collaborations

R2-0 Organization

areaDetector

Top-level module
RELEASE files, documentation, Makefile

ADCore

Core module
Base classes, plugins,
simDetector, documentation

ADB binaries

Binary libraries for
Windows (HDF5,
GraphicsMagick)

ADProsilica

Prosilica driver

ADPilatus

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

Types of Detector Drivers

- Vendor C/C++ library
- Vendor socket protocol
- Vendor application program with automation mechanism

Detector drivers (23 total)

- 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.
 - Controlled via calls to vendor PvAPI C library
- Firewire (IEEE-1396 DCAM) (ADFireWireWin, firewireDCAM)
 - Vendor-independent Firewire camera drivers for Linux and Windows
 - Controlled via open-source libraries on Windows and Linux

Detector drivers

- Roper driver (ADRoper)
 - Princeton Instruments and Photometrics cameras controlled via Microsoft COM control of WinView applications
- PVCAM driver (ADPvCam)
 - Princeton Instruments and Photometrics cameras
 - Controlled via PVCAM C library
- Pilatus driver (ADPilatus)
 - Pilatus pixel-array detectors.
 - Controlled via sockets to vendor camserver application
- marCCD driver (ADmarCCD)
 - Rayonix (MAR-USA) CCD x-ray detectors
 - Controlled by socket communication to marCCD remote control
- ADSC driver (ADADSC)
 - ADSC CCD detectors
 - Controlled by C calls to vendor library

Detector drivers (continued)

- mar345 driver (ADmar345)
 - marResearch mar345 online image plate
 - Controlled by socket communication to mar345dtb socket server
- Perkin-Elmer driver (ADPerkinElemer)
 - Perkin-Elmer amorphous silicon flat-panel detectors
 - Controlled by C calls to vendor library
- Bruker driver (ADBruker)
 - Bruker CCD detectors
 - Controlled via their Bruker Instrument Server (BIS) socket server
- LightField driver (ADLightField)
 - Princeton Instruments detectors, including all recent models
 - Controlled via their LightField application using the Microsoft Common Language Runtime to automate it

Detector drivers (continued)

- PICAM driver (ADPiCam)
 - Princeton Instruments cameras, including recent models
 - Controlled via PICAM C library
 - Currently under development by John Hammonds
- PSL driver (ADPSL)
 - Photonic Sciences Limited detectors
 - Controlled via socket connection to their Python PSLViewer server
- URL driver (ADURL)
 - Driver to display images from any URL. Works with Web cameras, Axis video servers, static images, etc.
 - Controlled by calls to GraphicsMagick library

Detector drivers (continued)

- Andor driver (ADAndor)
 - Driver for Andor CCD cameras
 - Controlled by calls to V2 of their C SDK
- Andor3 driver (ADAndor3)
 - Driver for Andor sCMOS cameras
 - Controlled by calls to V3 of their SDK
- Point Grey driver (ADPointGrey)
 - Driver for GigE, USB-3.0, USB-2.0, and Firewire cameras from Point Grey Research
 - Controlled by calls to their C SDK
- Pixirad driver (ADPixirad)
 - Driver for CdTe pixel-array detectors from Pixirad
 - Controlled by TCP and UDP communication with their detector

Detector drivers (continued)

- Generic GigE driver (aravisGigE)
 - Should work with any GigEVision compliant camera. From Tom Cobb at Diamond.
 - Controlled using the Aravis reverse-engineered GigEVision library
- QImaging driver (ADQImaging)
 - QImaging cameras.
 - Controlled using Qimaging SDK
 - Written by Arthur Glowacki
- ADPvAccess
 - Driver that receives NDArrays over EPICS V4
 - Allows plugins to run in an EPICS IOC on a different machine than the detector
 - Written by David Hickin from Diamond.

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

The screenshot shows the 'Area Detector Control - 13SIM1:cam1' window. It is divided into several panels: Setup, Shutter, Collect, Readout, and File. The Setup panel shows configuration for the asyn port (SIM1), EPICS name (13SIM1:cam1), manufacturer (Simulated detector), and model (Basic simulator). The Shutter panel shows the shutter mode (None), status (Det. Closed, EPICS Closed), and open/close buttons. The Collect panel shows acquisition parameters like exposure time (0.010), acquire period (0.000), and number of images (10). The Readout panel shows sensor size (640x480), binning (1x1), region start (0,0), region size (540x480), and other parameters. The File panel shows the driver file I/O button.

ADBase.adl

Area Detector Control - 13SIM1:cam1:

Setup

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

Shutter

Shutter mode
Status: Det. `Closed` EPICS `Closed`
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`
Trigger mode `Internal`
`Collecting`
Acquire
Detector state `Readout`
Time remaining `0.000`
Image counter `0` `703`
Image rate `67.0`
Array callbacks `Enable`

Readout

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

File

Driver file I/O

Pilatus specific control screen

pilatusDetector.adl

Pilatus Detector Control - 13PIL1:cam1:

Setup

asyn port **PIL**
EPICS name **13PIL1:cam1:**
Manufacturer **Dectris**
Model **Pilatus**
Connection **Connected**
Debugging ☐

Shutter

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

Status

Status: **Waiting for acquire command**
To camserver: **Exposure /corvette/home/epics/temp/pilatus_test_A_081.tif**
From camserver: **7 OK /corvette/home/epics/temp/pilatus_test_A_081.tif**

Data corrections

Bad pixel file:
Bad pixels: **0**
Flat field file:
Flat field valid: **No** Min. flat field: **100**

Collect

Exposure time **1.000** **1.000**
Acquire period **0.150** **0.150**
Images **1** **1**
Delay time **0.000000** **0.000000**
Exp./image **1** **1**
Trigger mode **Internal** **Internal**
Acquire **Start** **Stop**
Armed **Unarmed**
Image counter **0** **1**
Image rate **0.0**
Array callbacks **Enable** **Enable**

File

File path **/corvette/home/epics/temp/** Exists: **Yes**
File name **pilatus_test_A**
Next file # **82** **82**
Auto increment **Yes** **Yes** Ancillary information ☐
Filename format **%s%s_%3.3d.tif** File format **TIFF** **TIFF**
Last filename **/corvette/home/epics/temp/pilatus_test_A_081.tif**

Attributes

File **pilatusAttributes.xml**

Plugins

All **File** **ROI**
Stats **Other**

Detector

Detector Size **487** **195**
Threshold (keV) **10.000** **10.000**
Threshold apply **Yes** **Apply**
Shaping time/Gain **5-18KeV/Med/MedG**
Pixel cutoff **1071635**
Read file timeout **20.000**
Gap fill **N.A.**
Temperature **0.0** **-99.0** **0.0**
Humidity **0.0** **-99.0** **0.0**
TVX version **Unknown**

MAR-345 specific control screen

marCCD.adl

marCCD Detector Control - 14ID_MX340:cam1:

Setup

asyn port

EPICS name

Manufacturer

Model

Connected

Connection

Debugging

Shutter

Shutter mode

Status: Det. EPICS

Open/Close

Delay: Open Close

EPICS shutter setup

Status

Detector state Time remaining

Server state Readout status

Task status Correct status

Acquire status Writing status

Dezinger status Series status

Status poll rate

To marCCD server:

From marCCD server:

Plugins

Readout

	X	Y
Detector Size	7680	7680
	2	2
Binning	2	2
Image Size	3840	3840
Image Size (bytes)	29491200	
Frame shift	0	0
Stability	0.00	0.00
Server mode	2	

Collect

Exposure time

Acquire period

images

images counter

Image mode

Frame type

Overlap mode

Trigger mode

Readout mode

Gate mode

Array callbacks

Acquire

Image counter

Attributes

File

File

File path Exists:

File name

Next file #

Auto increment Ancillary information

Filename format Example: %s%s_%3.3d.tif

Series format Example: %s%s_%3.3d

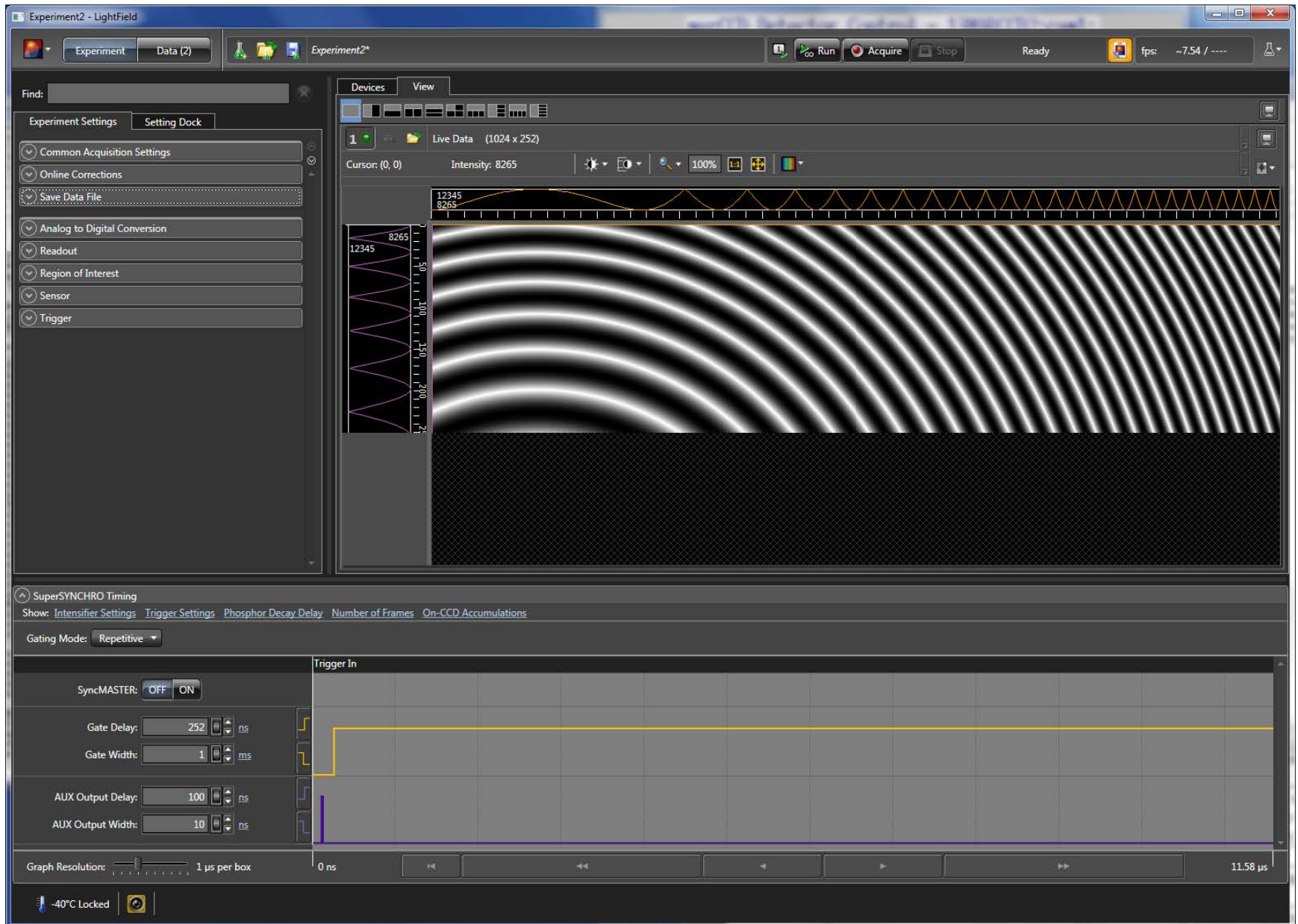
First series #

Series digits

Last filename

Save file Auto save

LightField driver



LightField driver

LightField.adl

Area Detector Control - 13LF1:cam1:

Setup

asyn port LF1
EPICS name 13LF1:cam1:
Manufacturer Princeton Instrument
Model PIXIS: 100BR
Connected
Connection
Debugging

Plugins

All
Stats

Readout

	X	Y
Sensor Size	1340	100
Binning	1	1
	0	78
Region Start	0	10
	1340	10
Region Size	1	10
	No	No
Reverse	No	No
Image Size	1340	10
Image Size (bytes)		26800
Gain	Low	Medium
Data type		UInt16
Temperature	-75.000	-75.000
Actual temperature		-75.000

Shutter

Shutter Type
LF Shutter Mode
Status: Det. Closed EPICS Closed
Open/Close
Delay: Open 0.000 Close 0.000
EPICS shutter setup

Collect

Exposure time 5.000
Acquire Period 0.000 0.000
Accumulations 0
Exposures 1
Frames 1
Exposures Complete 0
Frames Complete 1535
Acquisitions 0
Acquisitions Complete 0
Image Mode Normal
Trigger Mode

Done

Acquire
Detector State Idle
Ready to Run Ready
Image counter 1535
Image Rate 0.0
Array Callbacks Disable

File and Background

Spectrometer

[860nm, 300] [1] [0]
Grating
Center wavelength 750.000 750.000
Entrance width 100 100
Exit port Front

Intensifier

Int. Enable Disable
Intensifier Gain 0
Gating Mode Repetitive
Trigger Frequency 1e+001 1e+001
SyncMaster Enable
SyncMaster2 Delay 1.00e-00 1.00e-004
Rep. Gate Width 5.00e-00 5.00e-002
Rep. Gate Delay 0.00e+00 0.00e+000
Seq. Start Width 0.00e+00 0.00e+000
Seq. Start Delay 0.00e+00 0.00e+000
Seq. End Width 0.00e+00 0.00e+000
Seq. End Delay 0.00e+00 0.00e+000
Aux I/O Width 2.00e-00 2.00e-006
Aux I/O Delay 0.00e+00 0.00e+000

Experiment

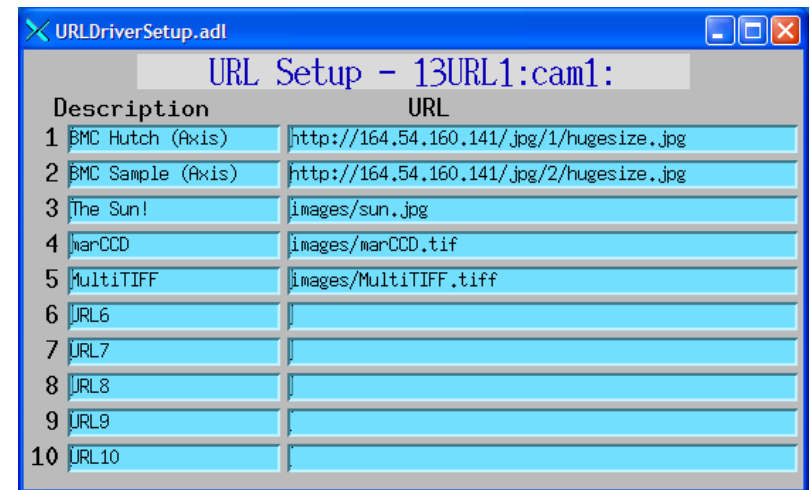
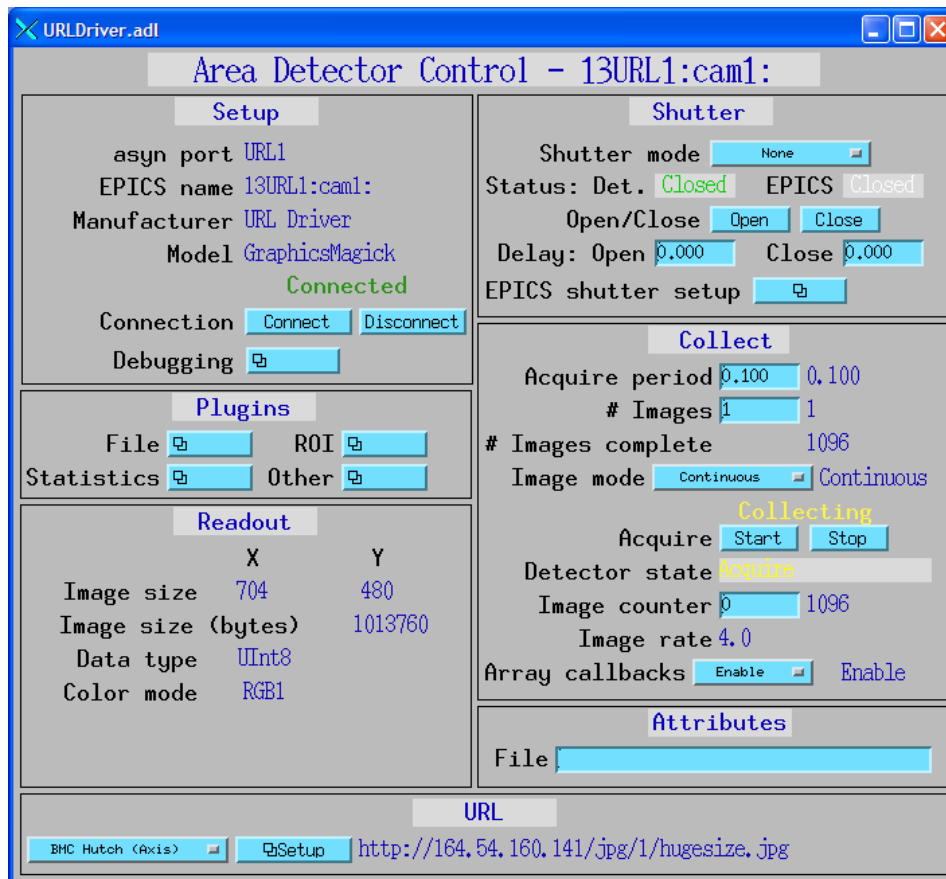
PIXIS 5_29_2013.lfe
Experiment

Attributes

File

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.

Andor.adl

Andor Detector Control - 13ANDOR1:cam1:

Setup

asyn port **ANDOR**
EPICS name **13ANDOR1:cam1:**
Manufacturer **Andor**
Model **DY934_BR_DD**
Connected
Connection **Connect** **Disconnect**
Debugging ☐

Shutter

Shutter Type **Detector output**
Andor Shutter Mode **Auto**
External shutter **High To Open**
Status: Det. **Closed** EPICS **Closed**
Open/Close **Open** **Close**
Delay: Open **0.100** Close **0.050**
EPICS shutter setup ☐

Plugins

All **File** ☐ **ROI** ☐
Stats ☐ **Other** ☐

Readout

	X	Y
Sensor Size	1024	1024
Binning	1	1
Region Start	0	0
Region Size	1024	1024
Reverse	No	No
Image Size	1024	1024
Image Size (bytes)	2097152	
Gain	0	0
ADC Speed	2.5 MHz	2.5 MHz
Data type	UInt16	UInt16

Collect

Exposure Time **0.300** **0.300**
Accumulate Period **0.000** **0.899**
Acquire Period **0.100** **0.899**
Accums/Image **1**
Exposures Complete **34**
Images/Acquis. **2** **2**
Images Complete **34**
Image Mode **Continuous** **Continuous**
Trigger Mode **Internal**
Collecting
Acquire **Start** **Stop**
Detector State **Acquire**
Detector Status **Data acquisition in progress.**
Andor Message
Time Remaining **0.000**
Image Counter **0** **56**
Image Rate **1.0**
Array Callbacks **Enable** **Enable**

Cooler

Cooler **On** **On**
Temperature **10.000** **10.000**
Status **Stabilized at set point**

File

Driver File I/O ☐

Attributes

File

PAL File

Path **GREY.PAL**

Perkin Elmer Flat Panel Driver

PerkinElmer.adl

Perkin Elmer Control - 13PE1:cam1:

Setup

asyn port PEDET1
EPICS name 13PE1:cam1:
Manufacturer Perkin Elmer
Model XRD0820
Connected
Connection
Debugging

Corrections

Corrections Directory
C:\Perkin_Elmer\
Offset
Offset Frames 10 10
Acquire Offset Correction Done
Correction Available

Gain
Gain Frames 20 0
Acquire Gain Correction Done
Correction Not Available
Load Gain File Save Gain File

Bad Pixel File
16#5149_1pF_PxlMask.his
Correction Available
Load Bad Pixel File

Shutter

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

Collect

Exposure time 0.200 0.200
Gain 0.25pF 0.25pF
Images 20 20
Images complete 31
Skip frames Disable
Frames to skip 1 1
Image mode Continuous
Trigger mode Internal

Collecting
Acquire
Detector state Acquire
Image counter 0 151
Image rate 5.0
Array callbacks Enable

Plugins

All
Stats

Readout

	X	Y
Sensor size	2048	2048
Binning	1	1
Image size	2048	2048
Image size (bytes)	8388608	

Setup

Frame Buffers 10 10
Frame buffer index 7
Image Number 31

Attributes

File PerkinElmerAttributes.xml

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



Mono
Sensor



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

pointGrey.adl

Point Grey Area Detector Control - 13PG1:cam1:

Setup

asyn port **PG1**
EPICS name **13PG1:cam1:**
Manufacturer **Point Grey Research**
Model **Blackfly BFLY-PGE-20**
Serial Number **13481965**
Firmware Vers. **1.27.3.0**
Software Vers. **2.6.3**
Debugging ☒

Shutter

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

Status

Status rate **1 second**
Dropped frames **0**
Corrupt frames **0**
Driver dropped **0**
Transmit failed **0**
Temperature **42.8**

Attributes

File

Trigger

Internal
Trigger mode **Internal**
Trigger source **GPIO_0** **GPIO_0**
Trigger polarity **High** **High**
Trigger delay **0.000** **0.000**
Skip frames **0** **0**
Software trigger **Trigger**

Strobe

Strobe source **GPIO_1** **GPIO_1**
Strobe enable **Enable** **Enable**
Strobe polarity **Low** **Low**
Strobe delay **0.001** **0.001**
Strobe duration **0.020** **0.020**

Bandwidth Control

Max packet size **9000**
Packet size **1440** **1440**
Packet size **1440**
GigE packet delay **400** **400**
Bandwidth (MB/s) **54.9**

Collect

Exposure time **0.040** **0.033**
Acquire period **0.250** **0.033**
Frame rate **43.716** **30.000**
Images **1000** **1000**
Images complete **189**
Exp./image **1** **1**
Image mode **Multiple** **Multiple**
Collecting
Acquire **Start** **Stop**
Detector state **Waiting**
Status **Waiting**
Image counter **0** **189**
Image rate **30.0**
Array callbacks **Enable** **Enable**

Readout

	X	Y
Sensor size	1600	1200
Region start	0	0
Region size	1600	1200
GigE binning	1x1	1x1
Image size	1600	1200
Image size (bytes)		1920000
Gain	0.000	0.000
Data type	UInt8	
Color mode	Mono	
Video mode	Format7	Format7
Format7 mode	0 (1600x1200)	
Properties	☒	
Frame rate	☒ More	Undefined1
Pixel format	☒ More	Raw8
Convert raw	None	None
Timestamp	Camera	Camera

Buffers

Buffers max/used	0	1
Buffers alloc/free	2	1
Memory max/used (MB)	0.0	3.7
Buffer & memory polling	1 second	

Plugins

All ☒ File ☒ ROI ☒
Stats ☒ Other ☒

Point Grey Driver (Grasshopper3 camera)

pointGreyProperties.adl

13PG1:cam1: Point Grey Properties

Property	Device Unit Control			Absolute Control			
	On/Off	One push	Auto/Manual	Set	Readback	Min	Max
Brightness	On			0	0	0	511
Auto exposure	On	Push	Manual	468	468	1	1023
Sharpness	Off		Manual	0	0	0	4095
White bal. red	Off						
White bal. blue							
Hue	Off						
Saturation	Off						
Gamma	On			512	512	512	4095
Shutter	On	Push	Manual	1181	1242	1	1242
Gain	On	Push	Manual	8	240	0	240
Iris	Off						
Focus	Off						
Temperature	Off						
Trigger mode	On		Manual	5	0	2844	1
Trigger delay	On			0	0	0	4095
Frame rate	Off		Manual	752	407	407	1629
Zoom	Off						
Pan	Off						
Tilt	Off						

Note: The 'Absolute Control' section in the original image contains additional rows for properties like Readback, Min, and Max, which are not explicitly labeled in the table above. These values are provided for reference.

Plugins

- Designed to perform real-time processing of data, running in the EPICS IOC (not over EPICS Channel Access)
- Receive NDAarray 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 NDAarray 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 NDAarray callbacks to another driver or plugin
- Some plugins are also sources of NDAarray 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 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
- 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.

Plugins (recent additions)

- NDPluginCircularBuffer
 - Buffers NDArrays in a circular buffer.
 - Outputs the arrays on receipt of a trigger, either as PV or NDArray attribute.
 - Supports pre-trigger and post-trigger samples
 - Written by Alan Greer at Observatory Sciences
- NDPluginAttribute
 - Extracts an attribute from an NDArray and publishes as a scalar and an array
 - Written by Matt Pearson at ORNL
- ADPluginEdge
 - Does edge detection using the OpenCV Canny function
 - Written by Keith Brister at LS-CAT

Plugins (recent additions)

- NDPluginROIStat
 - Computes simple statistics on multiple ROIs.
 - More efficient than chaining an NDPluginROI and NDPluginStats when there are many ROIs
 - Written by Matt Pearson at ORNL
- ffmpegServer
 - MJPEG server that allows viewing images in a Web browser. From DLS.
 - Puts compressed images on the network, greatly reducing bandwidth compared to uncompressed channel access arrays.
- ADPvAccess
 - Plugin that sends NDArrays over EPICS V4
 - Allows plugins to run in an EPICS IOC on a different machine than the detector
 - Written by David Hickin from Diamond.

commonPlugins.adl All plugins at a glance

13SIM1: Common Plugins									
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

NDStdArrays.adl

13SIM1:image1:

asyn port	Image1		
Plugin type	NDPluginStdArrays		
Array port	SIM1	SIM1	
Array address	0	0	
Enable	Enable	Enable	
Min. time	0.000	0.000	
Callbacks block	No	No	
Queue size/free	3	3	
Array counter	0	841924	
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	841924		
Time stamp	717905544.489		
Attributes file			
asyn record			

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
Binning	1	1	1
ROI start	0	0	0
ROI size	512	512	1
Reverse	No	No	No
ROI Size	512	512	0

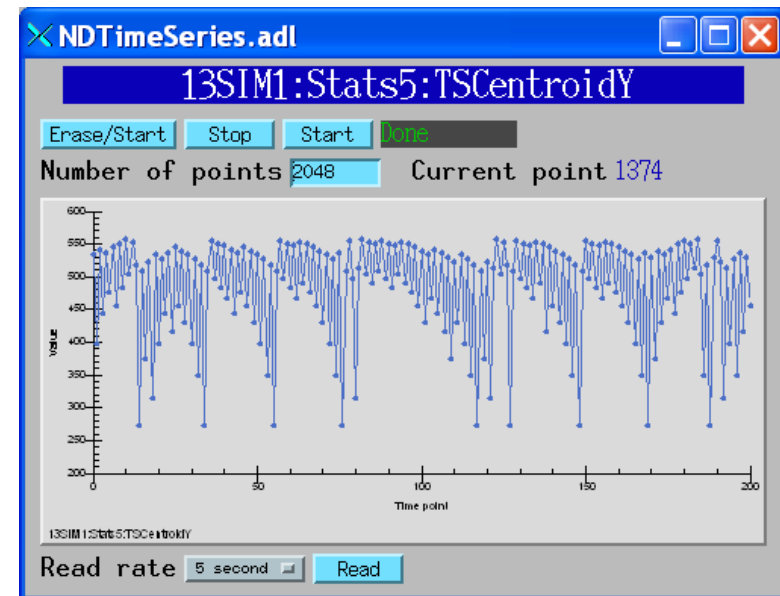
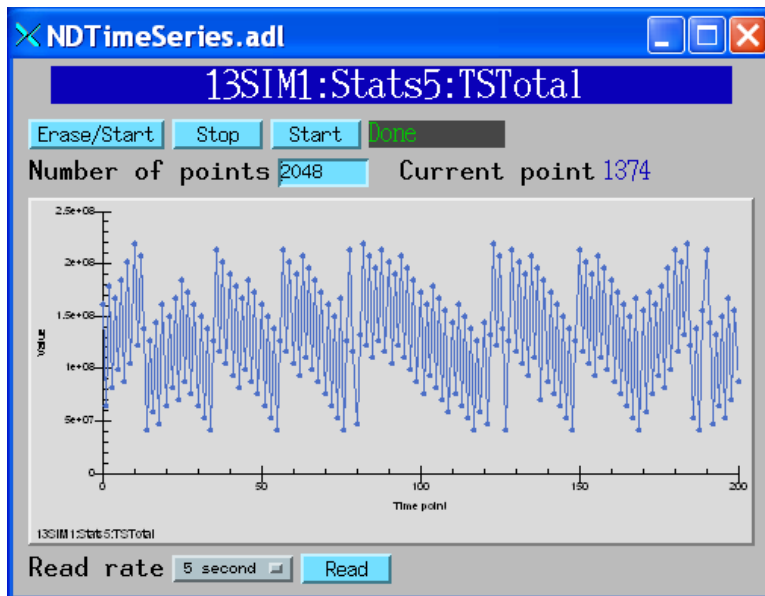
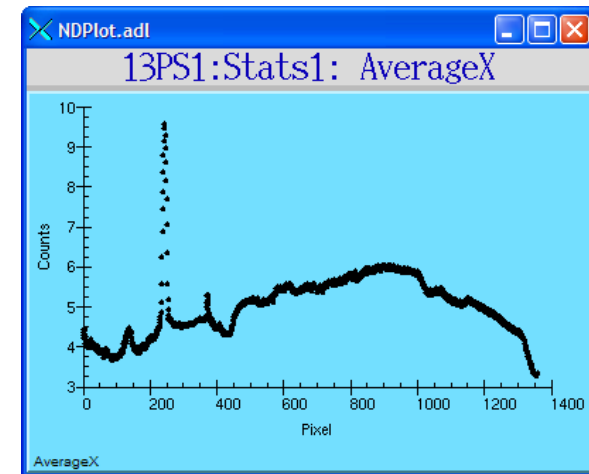
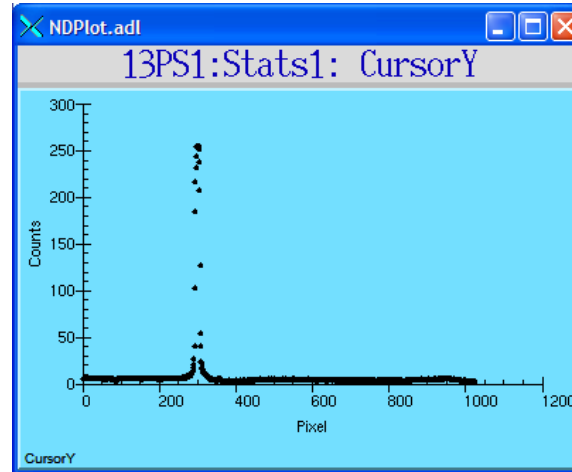
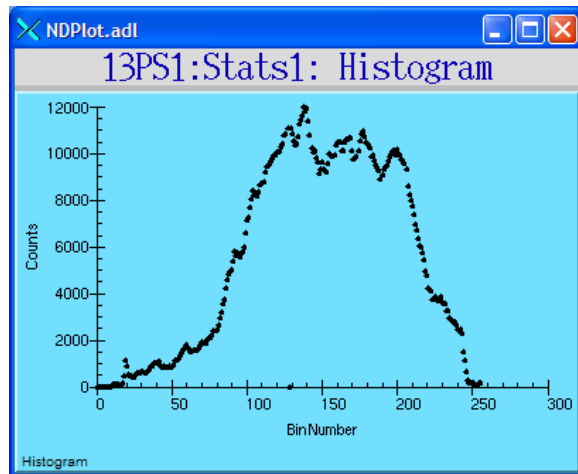
Statistics plugin

NDStats.adl

13SIM1:Stats5:

asyn port STATS5 Plugin type NDPluginStats Array port SIM1 SIM1 Array address 0 0 Enable ☐ Enable Min. time 0.000 0.000 Callbacks block ☐ No Queue size/free 20 12 Array counter 0 4056 Array rate 25.0 Dropped arrays 0 395 # dimensions 2 Array Size 1024 1024 0 Data type Int8 Color mode Mono Bayer pattern RGGB Unique ID 4451 Time stamp 717886862.801 Attributes file . asyn record ☐	Statistics Compute statistics ☐ Yes Background width 1 1 Minimum 0 Maximum 6 Min. X 0 Max. X 200 Min. Y 0 Max. Y 148 Total 622 Net 622 Mean 0 Sigma 0.0 Time series plots ☐	Profiles Compute profiles ☐ Yes Size X 1024 Y 1024 256 256 Cursor X ☐ 256 256 Cursor Y ☐ Plot ☐
	Centroid Compute centroid ☐ Yes Centroid threshold 1 1 Centroid X 200.0 Y 150.1 Sigma X 3.9 Y 3.9 Sigma XY -0.024 Time series plots ☐	Histogram Compute histogram? ☐ Yes Size 256 256 Minimum 0 0 Maximum 255 255 Entropy -13.860 Plot ☐
	Time Series Erase/Start ☐ Stop ☐ Start ☐ Acquiring Number of points 2048 Current point 82 Read rate ☐ 5 second ☐ Read ☐	

Statistics plugin (continued)



Overlay plugin

NDOverlay.adl

13SIM1:Over1:

asyn port	OVER1		
Plugin type	NDPluginOverlay		
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	2055	
Array rate	10.0		
Dropped arrays	0	0	
# dimensions	3		
Array Size	3	1024	1024
Data type	UInt8		
Color mode	RGB1		
Bayer pattern	RGGB		
Unique ID	3042		
Time stamp	778440667.851		
Attributes file			
asyn record			

Overlay definitions

Individual 0-7	Individual Overlays
Combined	Combined Overlays

Overlay plugin

NDOverlay8.adl

13PS1:Over1:

	Use?	Name	Shape	Draw mode	Red	Green (mono)	Blue	Pos. X	Size X	Pos. Y	Size Y	
	Yes	ROI1	Rectangle	Set	0	255	0	150	100	250	100	
1	☒ Yes	ROI1	Rectangle	Set	0	255	0	150	100	250	100	More
								150	100	250	100	
	No	ROI2	Rectangle	XOR	0	255	0	0	100	0	100	
2	☐ No	ROI2	Rectangle	XOR	0	255	0	0	100	0	100	More
								0	100	0	100	
	No	ROI3	Rectangle	Set	0	255	0	0	0	0	0	
3	☐ No	ROI3	Rectangle	Set	0	255	0	0	0	0	0	More
								0	0	0	0	
	No	ROI4	Rectangle	Set	0	255	0	0	0	0	0	
4	☐ No	ROI4	Rectangle	Set	0	255	0	0	0	0	0	More
								0	0	0	0	
	Yes	Cursor1	Cross	Set	0	255	0	420	50	196	50	
5	☒ Yes	Cursor1	Cross	Set	0	255	0	420	50	196	50	More
								420	50	196	50	
	No	Cursor2	Cross	XOR	0	255	0	227	199	171	128	
6	☐ No	Cursor2	Cross	XOR	0	255	0	227	199	171	128	More
								227	199	171	128	
	Yes	Box1	Rectangle	XOR	0	255	0	400	50	400	50	
7	☒ Yes	Box1	Rectangle	XOR	0	255	0	400	50	400	50	More
								400	50	400	50	
	No	Box2	Rectangle	XOR	0	255	0	142	368	107	128	
8	☐ No	Box2	Rectangle	XOR	0	255	0	142	368	107	128	More
								142	368	107	128	

Overlay plugin

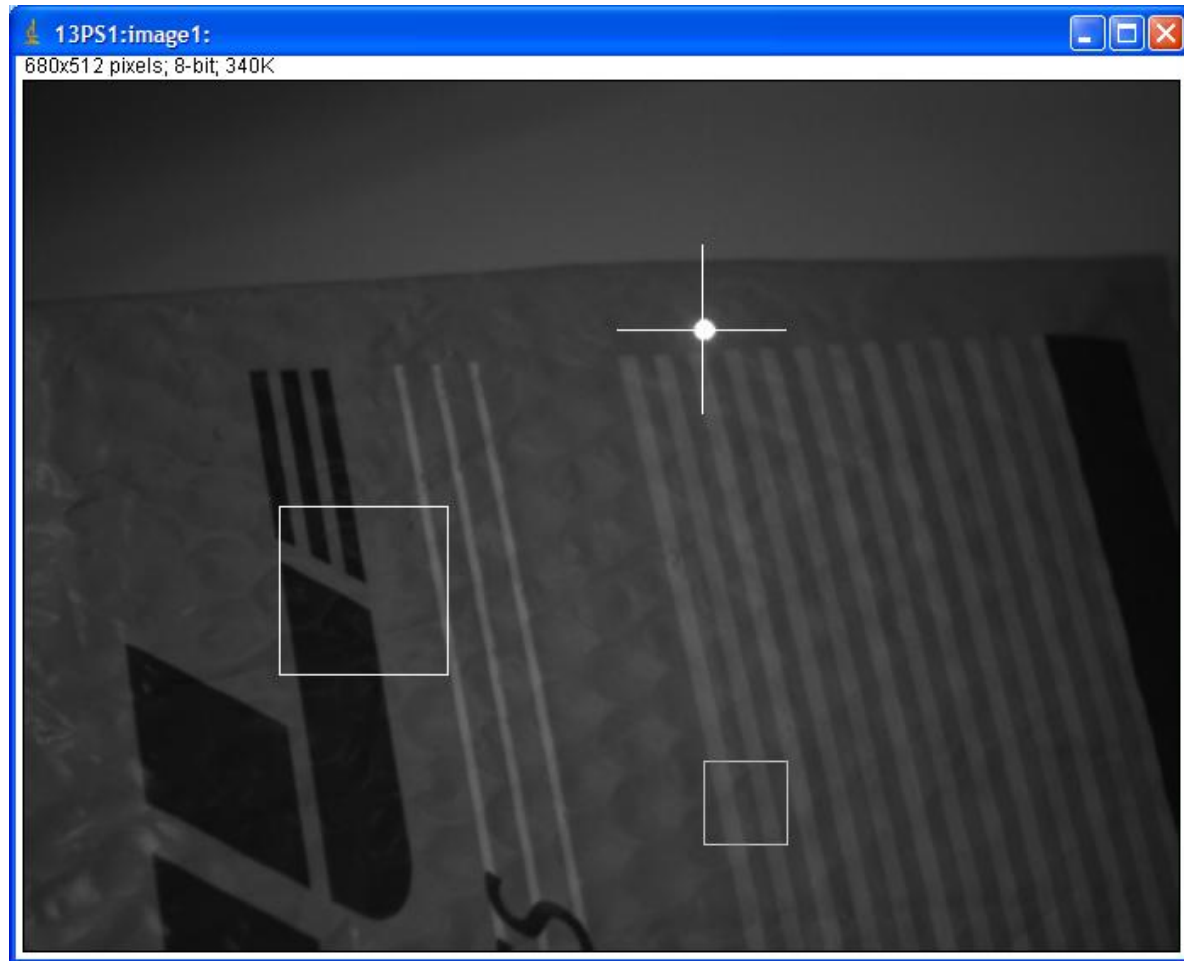
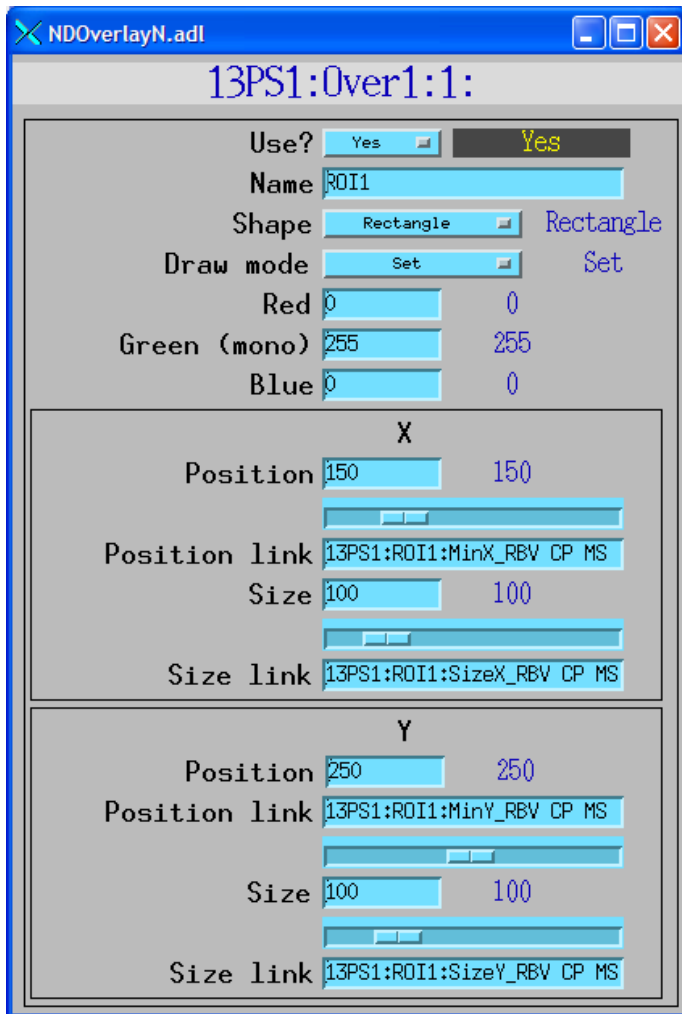
NDOverlayN.adl

13SIM1:Over1:1:

Use?	☐ Yes ☒ Yes
Name	Text 1
Shape	Text Text
Draw mode	Set Set
Red	255 255
Green (mono)	255 255
Blue	255 255
Display Text	This is some text
Time format	%Y-%m-%d %H:%M:%S.%03f
Format example	%Y-%m-%d %H:%M:%S.%03f
Font	6x13

	X	Y
Position	50 50	102 102
Position link	13SIM1:ROI1:MinX_RBV	13SIM1:ROI1:MinY_RBV
Size	200 200	200 200
Size link	13SIM1:ROI1:SizeX_RBV	13SIM1:ROI1:SizeY_RBV
Width	1 1	1 1
Width link	CP NPP MS	CP NPP MS

Overlay plugin



Centroid of laser pointer calculated by statistics plugin

Cursor overlay X, Y position linked to centroid

Processing plugin

NDProcess.adl

13SIM1:Proc1:

asyn port

PROC1

Plugin type

NDPluginProcess

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

11572

Array rate

47.0

Dropped arrays

0

0

dimensions

2

Array Size

1024

1024

0

Data type

Int8

Color mode

Mono

Bayer pattern

RGGB

Unique ID

12032

Time stamp

717887092.888

Attributes file

asyn record

Background subtraction

Save background

Save

Invalid

Enable background

Disable

Disable

Flat field normalization

Save flat field

Save

Invalid

Enable flat field

Disable

Disable

Scale flat field

255

255

Scale and Offset

Enable scale/off.

Disable

Enable

Auto scale/off.

Auto calc

Scale value

0.10

42.50

Offset value

0.00

0.00

Low/High Clipping

Enable low clip

Disable

Enable

Low clip value

100

0

Enable high clip

Disable

Enable

High clip value

150

255

Output data type

Data type

Automatic

Automatic

Recursive filter

Enable filter

Disable

Disable

N filter

100

100

N filtered

0

Filter type

RecursiveAve

Reset filter

Reset

Auto reset filter

Yes

Filter callbacks

Every array

00offset

0.00

0.00

0Scale

1.00

1.00

OC1

1.00

1.00

OC2

-1.00

-1.00

OC3

0.00

0.00

OC4

1.00

1.00

F0ffset

0.00

0.00

FScale

1.00

1.00

FC1

1.00

1.00

FC2

-1.00

-1.00

FC3

0.00

0.00

FC4

1.00

1.00

R0ffset

0.00

0.00

RC1

0.00

0.00

RC2

1.00

1.00

0[n] = 00ffset + 0Scale*((OC1+OC2/N)*F[n-1] + (OC3+OC4/N)*I[n])

F[n] = F0ffset + FScale*((FC1+FC2/N)*F[n-1] + (FC3+FC4/N)*I[n])

On filter reset:

F[0] = R0ffset + RC1*F[n] + RC2*I[0]

I = Input array in callback

F = Stored filter (double precision)

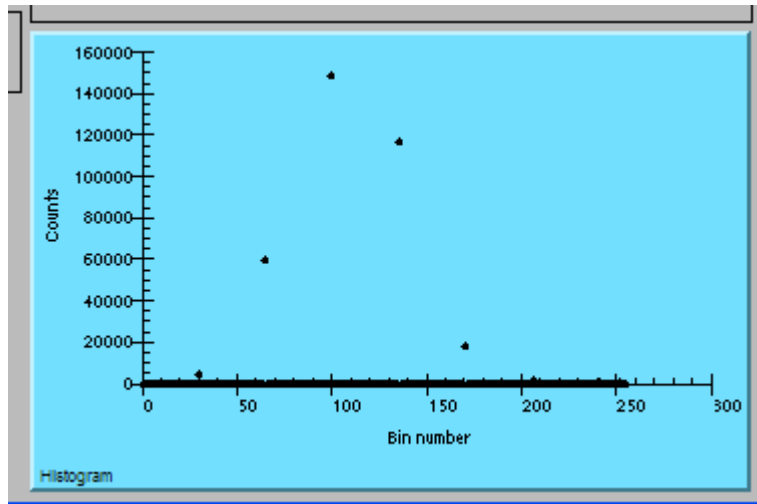
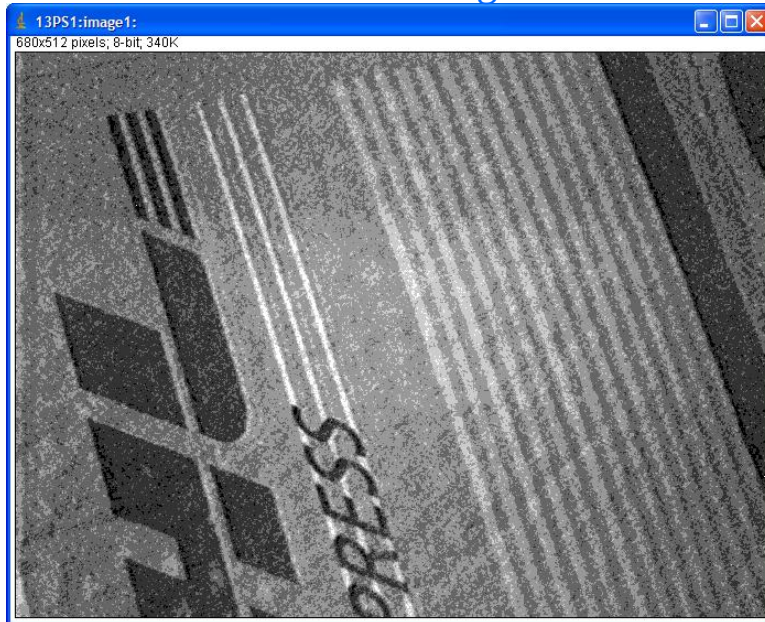
N = value of NumFiltered

0 = Output array passed to clients

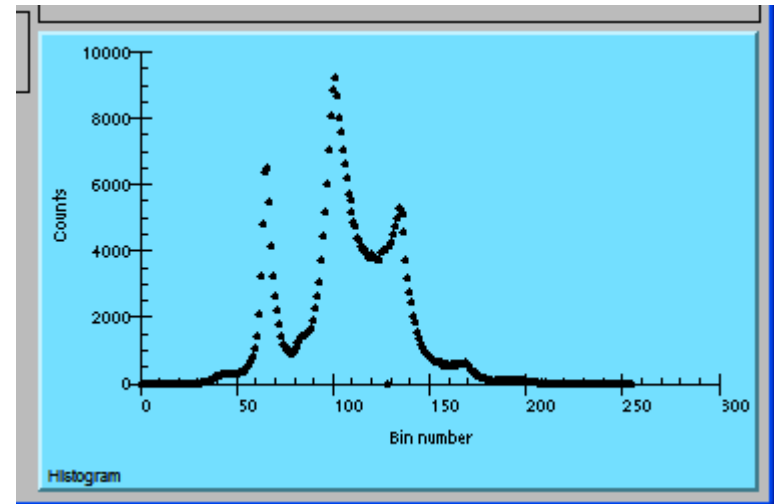
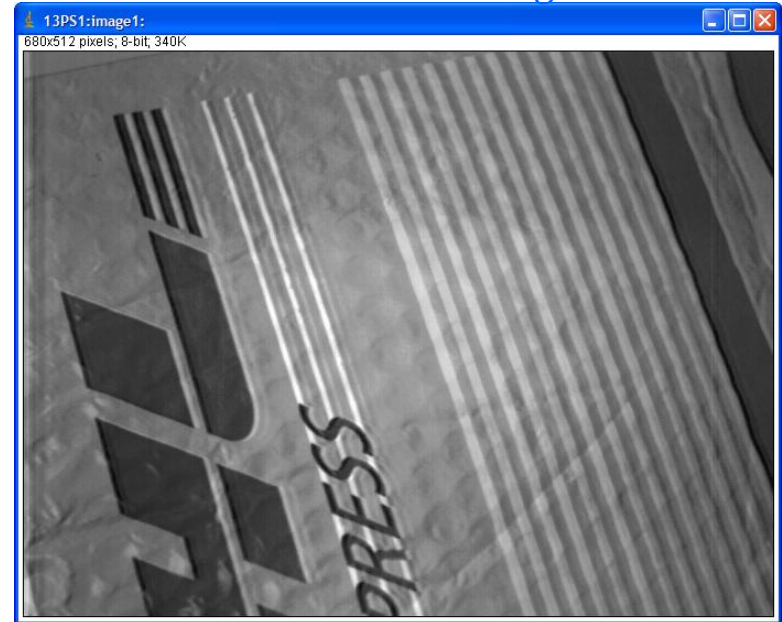
Processing plugin

30 microsec exposure time

No filtering



N=100 recursive average filter



Processing plugin

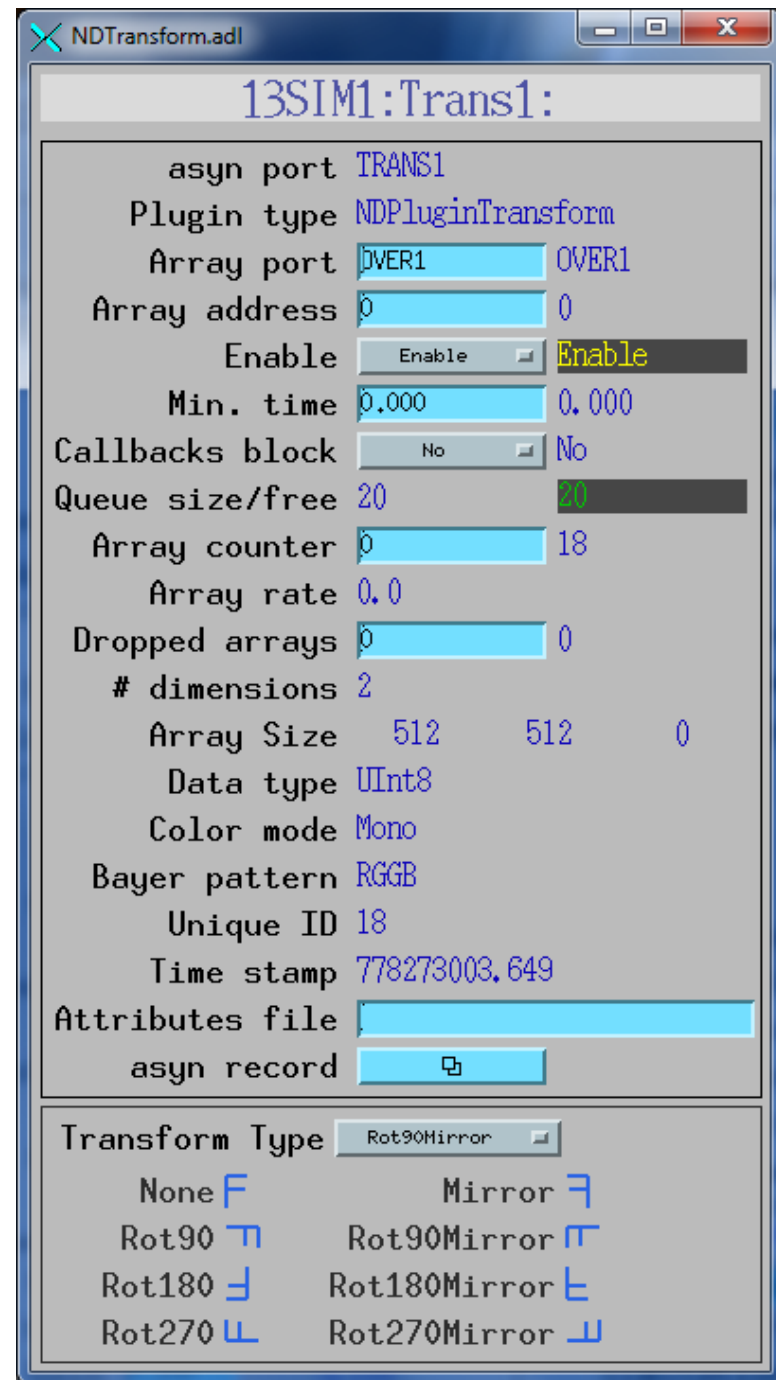
Pre-defined recursive filters

- RecursiveAve
 - Recursive average.
 - Computes running average with $1/N$ contribution from new frame
- Average
 - True average, averages next N frames
- Sum
 - Sum of the next N frames
- Difference
 - Difference of frame N and $N-1$
- RecursiveAveDiff
 - Difference of recursive average and frame N
- CopyToFilter
 - Make the last stored frame be the next frame

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 (R2-1)
 - One array per file
 - NDFileJPEG
 - With compression control
 - One array per file
 - NDFileNetCDF
 - Popular self-describing binary format, supported by Unidata at UCAR
 - Multiple arrays per file

Plugins: NDPluginFile

– NDFileHDF5

- Writes HDF5 files with the native HDF5 API, unlike the NeXus plugin which uses the NeXus API. Supports 3 types of compression.
- R2-1 supports using an XML file to define the layout and placement of NDArrays and NDAttributes in the HDF5 file

– 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

NDPluginFile Recent Features

- New record, DeleteDriverFile. Allows file writing plugins to delete the "original" file that the driver created for that array if the following are all true:
 - The DeleteDriverFile record is "Yes".
 - The file plugin has successfully written a new file.
 - The array contains an attribute called "DriverFileName" that contains the full file name of the original file. The driver attributes XML file should contain a line like the following:

```
<Attribute name="DriverFileName" type="PARAM" source="FULL_FILE_NAME" datatype="string" description="Driver file name"/>
```
- Support for getting the file name and file number from array attributes rather than from the normal EPICS PVs. Allows the driver to control which plugin saves a particular array. (Ulrik Pedersen)
- 2 new records, WriteStatus and WriteMessage to display status of file writing operations.

NDPluginFile Recent Features

- “Lazy-open” (R2-1)
 - Normally files in stream mode are opened when Capture PV is set to 1
 - This requires that there have already been an NDArray received by that plugin with the correct dimensions and attributes
 - “Lazy-open” is selected the file is not opened until the first NDArray callback happens after Capture is set to 1.
 - Simpler for users, but poorer performance, can lead to dropped arrays
- File plugins can now create directories (R2-2)
- File plugins can write files with a temporary suffix and then rename the file after writing is complete. (R2-2)
 - Allows rsync, etc. to be used to copy files, with guarantee that they are complete

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

NDFileTIFF.adl

13SIM1:TIFF1:

asyn port	FileTIFF1
Plugin type	NDFileTIFF
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	0 357
Array rate	82.0
Dropped arrays	0 83
# dimensions	2
Array Size	1024 1024 0
Data type	Int8
Color mode	Mono
Bayer pattern	RGGB
Unique ID	438270
Time stamp	717964044.637
Attributes file	
asyn record	

File path	/corvette/home/epics/scratch/ADFileTest/	Exists: Yes
File name	test_tiff	
Next file #	358 358	
Auto increment	Yes Yes	
Filename format	%s%s_%.tiff	Example: %s%s_%.tiff
Last filename	/corvette/home/epics/scratch/ADFileTest/test_tiff_357.tiff	
Save file	Save Save	Read file Read Read
Write mode	Stream Stream	# Capture 1000 1000 157
Capture	Start Start Stop Stop	Delete driver file No No
Write status	Write OK	
Write message		

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

NDPluginFile display: netCDF

NDFileNetCDF.adl

13SIM1:netCDF1:

asyn port	FileNetCDF1
Plugin type	NDFileNetCDF
Array port	SIM1 SIM1
Array address	0 0
Enable	☒ Enable
Min. time	0.000 0.000
Callbacks block	☐ No
Queue size/free	20 20
Array counter	0 396
Array rate	47.0
Dropped arrays	0 0
# dimensions	2
Array Size	1024 1024 0
Data type	Int8
Color mode	Mono
Bayer pattern	RGGB
Unique ID	1148948
Time stamp	717912009.118
Attributes file	
asyn record	☒

File path	/home/epics/scratch/	Exists: Yes
File name	abc	
Next file #	15 15	
Auto increment	☒ Yes	
Filename format	%s%s_%.d.nc	Example: %s%s_%.3d.nc
Last filename	/home/epics/scratch/abc_14.nc	
Save file	☒ Save	
Read file	☒ Read	
Auto save	☐ No	
Write mode	☒ Stream	
# Capture	100 100	96
Capture	☒ Start	
Stop	☐ Stop	
Delete driver file	☐ No	
Write status	Write OK	
Write message		

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 10
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	

File path	/home/epics/scratch/	Exists: Yes
File name	test_mono	
Next file #	220 220	
Auto increment	Yes Yes	
Filename format	%s_%3.3d.h5	Example: %s_%3.3d.h5
Last filename	/home/epics/scratch/test_mono_219.h5	
Lazy open	Yes Yes	
Save file	Save Save	Read file Read Read
Write mode	Stream Stream	# Capture 100 100 28
Capture	Start Start Stop Stop	Delete driver file No No
Write status	Write OK	
Write message		

Compression	None None	Extra dimensions
# data bits	8	# (0-2) 0 0
Data bits offset	0	Size N 1 1
SZip # pixels	16 16	Name N frame number n
Zlib level	6	Size X 1 1
Store performance	No Yes	Name X scan dimension X
Store attributes	No Yes	Size Y 1 1
Run time	9.913	Name Y scan dimension Y
I/O speed	80.7	

Exists: Yes

hdf5_layout_demo.xml

XML File name hdf5_layout_demo.xml

NDFileHDF5

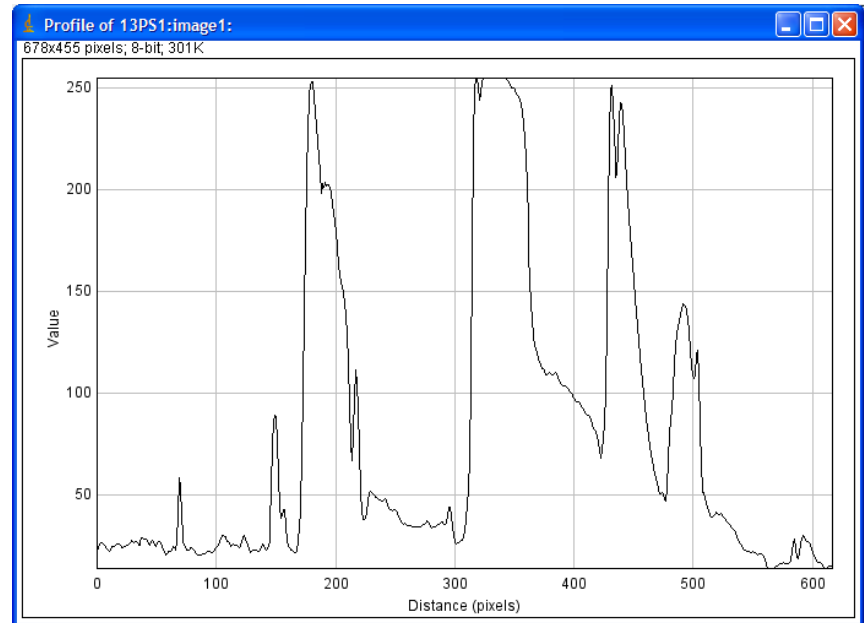
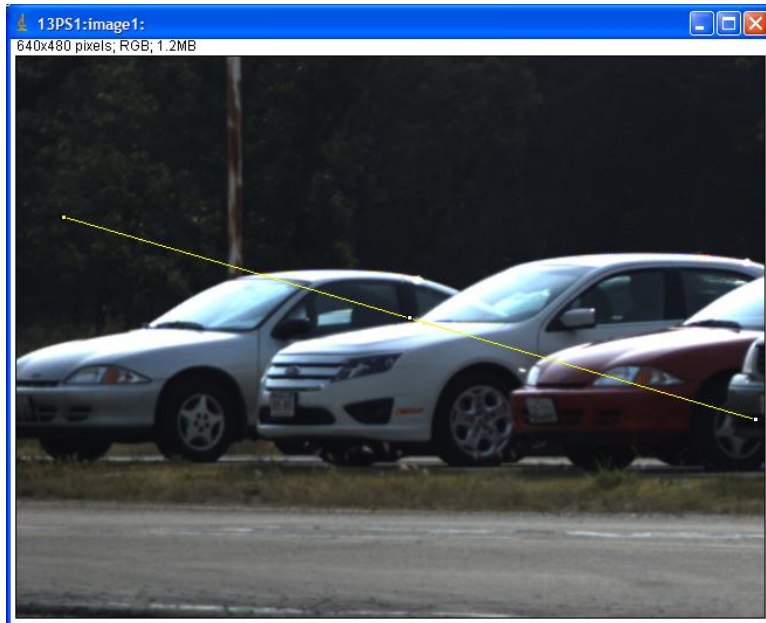
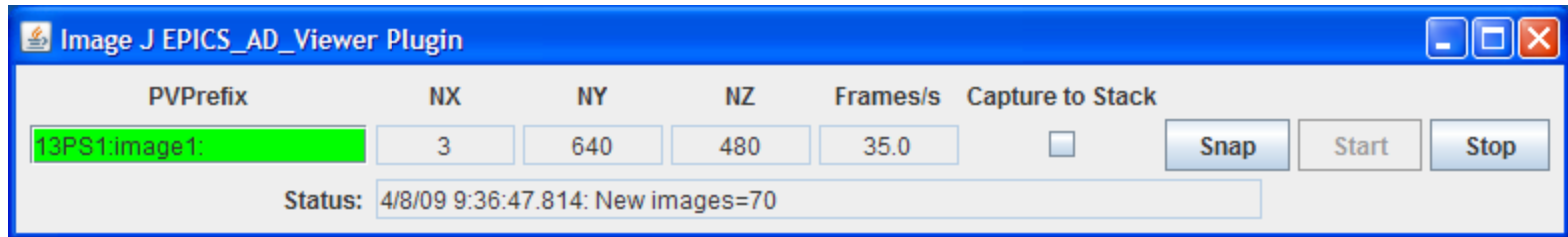
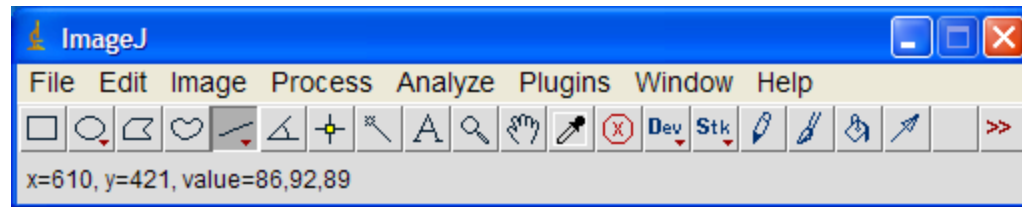
XML file to define file layout

```
<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

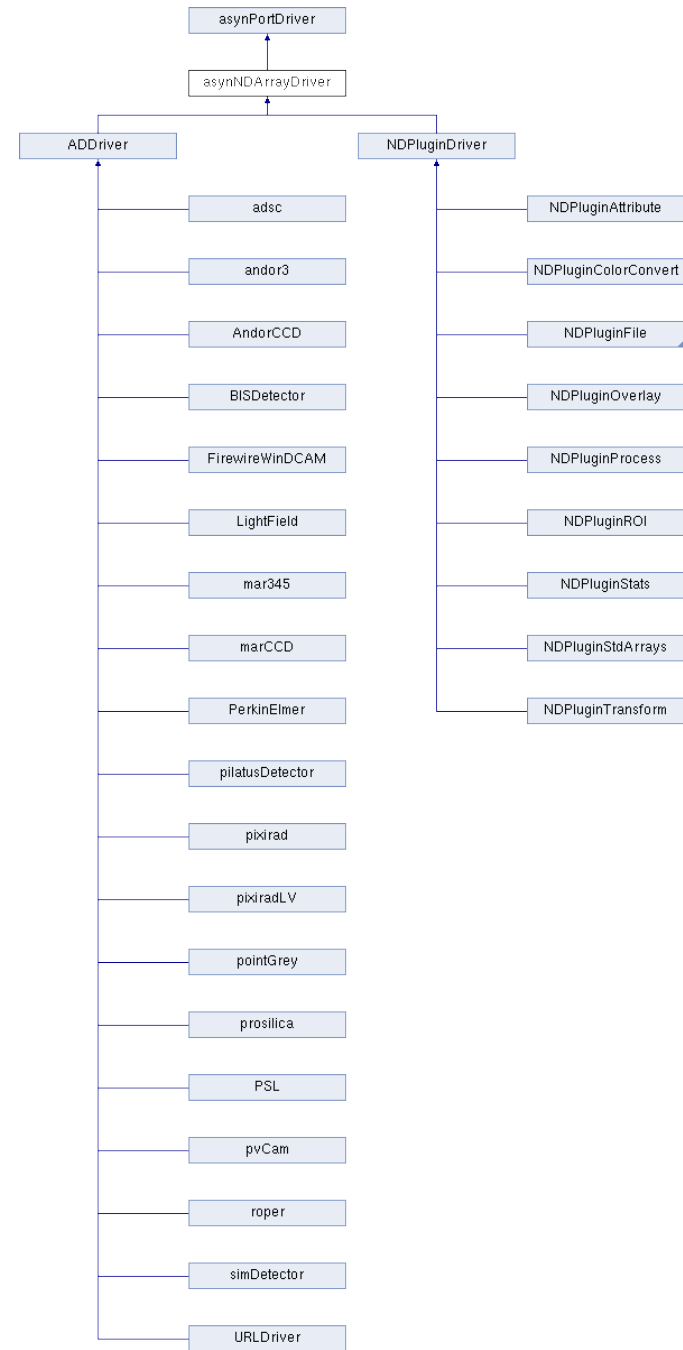
- areaDetector allows generic viewers to be written that receive images as EPICS waveform records over Channel Access
- Current viewers include:
 - ImageJ plugin EPICS_AD_Display. ImageJ is a very popular image analysis program, written in Java, derived from NIH Image.
 - IDL EPICS_AD_Display.
 - ffmpegServer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

ImageJ Viewer



Internals

Class hierarchy



Recent Changes (R2-0)

- Added EPICS timestamp field to NDArray
- EPICS records can get the timestamp when the image was collected by setting TSE=-2.
- Added NDAttributeFunct, allows user-written function to set the value of an attribute

Recent Changes (R2-1)

- NDPluginTransform
 - Greatly simplified and performance improved
- NDPluginFile
 - Added support for “lazy-open”
- NDFileHDF5
 - Support for defining the layout of the HDF5 file via an XML file
- NDFileTIFF
 - NDAttributes are written to user ASCII TIFF tags (up to 490)
- NDPluginOverlay
 - Support for text overlays
 - Support for defining the line width in rectangle and cross overlays

Recent Changes (R2-2, not released)

- NDPluginFile
 - Added support for creating directories
 - Added support for using a temporary suffix and renaming on close
- NDPluginROIStat
 - New plugin to do simple statistics on multiple ROIs
- simDetectorNoIOC
 - Example standalone C++ application that instantiates a simDetector without running an EPICS IOC
 - Shows that areaDetector drivers and plugins only depend on libCom and asyn libraries. Can be used from other control systems.

Future Ideas (R2-3?)

- Make trigger modes in ADBase be Free Run, Fixed Rate, External. Now it is Internal, External.
- AcquirePeriod would only apply in fixed rate, not in Free Run. This is easier for users.

Future Ideas (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

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

Future Ideas

- Extend areaDetector concepts to other types of detectors:
 - ADCs
 - Electrometers
 - Waveform digitizers
 - Oscilloscopes?
- They all produce 1-D (or 2-D for multi-channel inputs) arrays that could benefit from plugins for file saving, FFTs, ROI extraction, digital filtering, etc.

Future Ideas

- Export NDArrays via EPICS V4
- David Hickin (DLS) has demonstrated:
 - A plugin that exports NDArrays as V4 objects over Channel Access
 - An ADDriver that receives the V4 objects on another machine and has its own set of plugins
- Allows using multiple machines, and multiple processes, not just multiple cores in a single IOC for plugin processing

areaDetector Collaboration

- The move to GitHub has really helped areaDetector become a collaborative effort
- Many more people are contributing via additions and bug fixes.
- Make changes in their fork on github and then issue a “pull request”.
- Collaboration meeting ~monthly on Google Hangout (U. Pedersen, M. Rivers, A. Glowacki, M. Pearson, M. Krammer, N. Rees, D. Hickin, T. Cobb)
- In-person meetings ~2 times/year.
- Developed a road-map, following it pretty well.

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, Tim Madden, Tim Mooney, Arthur Glowacki, John Hammonds, Chris Roehrig (APS)
- Ulrik Pedersen, Tom Cobb, Nick Rees (Diamond)
- Alan Greer (Observatory Sciences)
- Matthew Pearson (ORNL)
- Emma Sheppard (Australian Synchrotron)
- Lewis Muir (IMCA CAT)
- Keith Brister (LS-CAT)
- Bruce Hill (SLAC)
- Many others for enhancements and bug fixes
- NSF-EAR and DOE-Geosciences for support of GSECARS where most of this work was done
- Thanks for your attention!!!