areaDetector

What's New and What's Next

Mark Rivers

GeoSoilEnviroCARS, Advanced Photon Source

University of Chicago

Outline

- Last update was October 2017 at ITER and ICALEPCS in Barcelona
  - areaDetector and ADCore Releases since then: R3-2, R3-3 (soon)
- Brief recap of the top items from R3-0 for those who missed it
- Major new features in 3-2 and 3-3
- Roadmap for R4-0 and R5-0

EPICS_NTNDA_Viewer ImageJ plugin (ADViewers R1-0)

- New ImageJ plugin written by Tim Madden and Marty Kraimer
- Essentially identical to EPICS_AD_Viewer.java except that it displays NTNDArrays from the NDPluginPva plugin, i.e. using pvAccess to transport the images rather than NDPluginStdArrays which uses Channel Access.

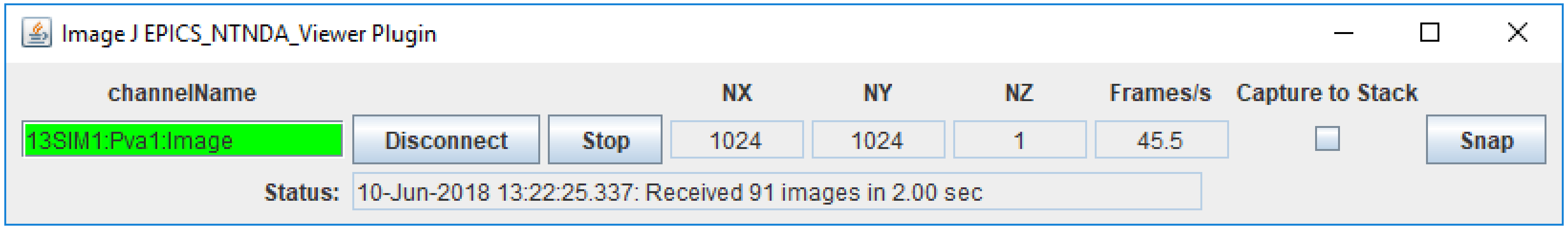
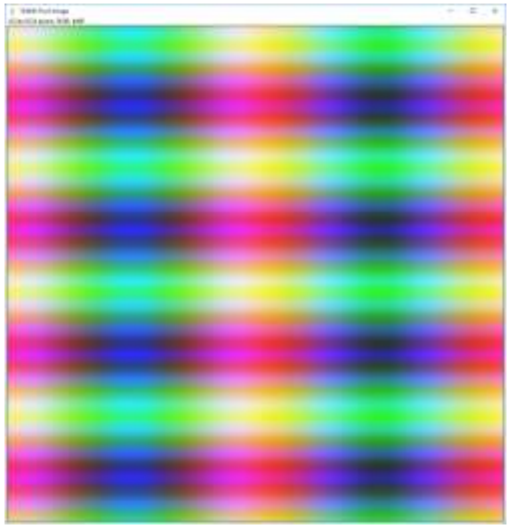


Image J EPICS_NTNDA_Viewer Plugin

channelName			NX	NY	NZ	Frames/s	Capture to Stack	
13SIM1:Pva1:Image	Disconnect	Stop	1024	1024	1	45.5	☐	Snap

Status: 10-Jun-2018 13:22:25.337: Received 91 images in 2.00 sec



EPICS_NTNDA_Viewer Advantages

- NTNDArray data transmitted "atomically" over the network
- With Channel Access size and data type of waveform record is fixed at iocInit, cannot be changed at runtime.
  - To view both 8-bit and 64-bit double FFT images waveform record needs to be 64-bit double, 8X network overhead for 8-bit. pvAccess changes the data type of the NTNDArrays dynamically at run-time.
- Channel Access requires setting EPICS_CA_MAX_ARRAY_BYTES, considerable confusion and frustration for users.
- NDPluginPva is 5X-10X faster than NDPluginStdArrays
- ImageJ is 1.5-2X faster with pvAccess than with Channel Access.

NDArrayPool Design Enhancements (R3-3)

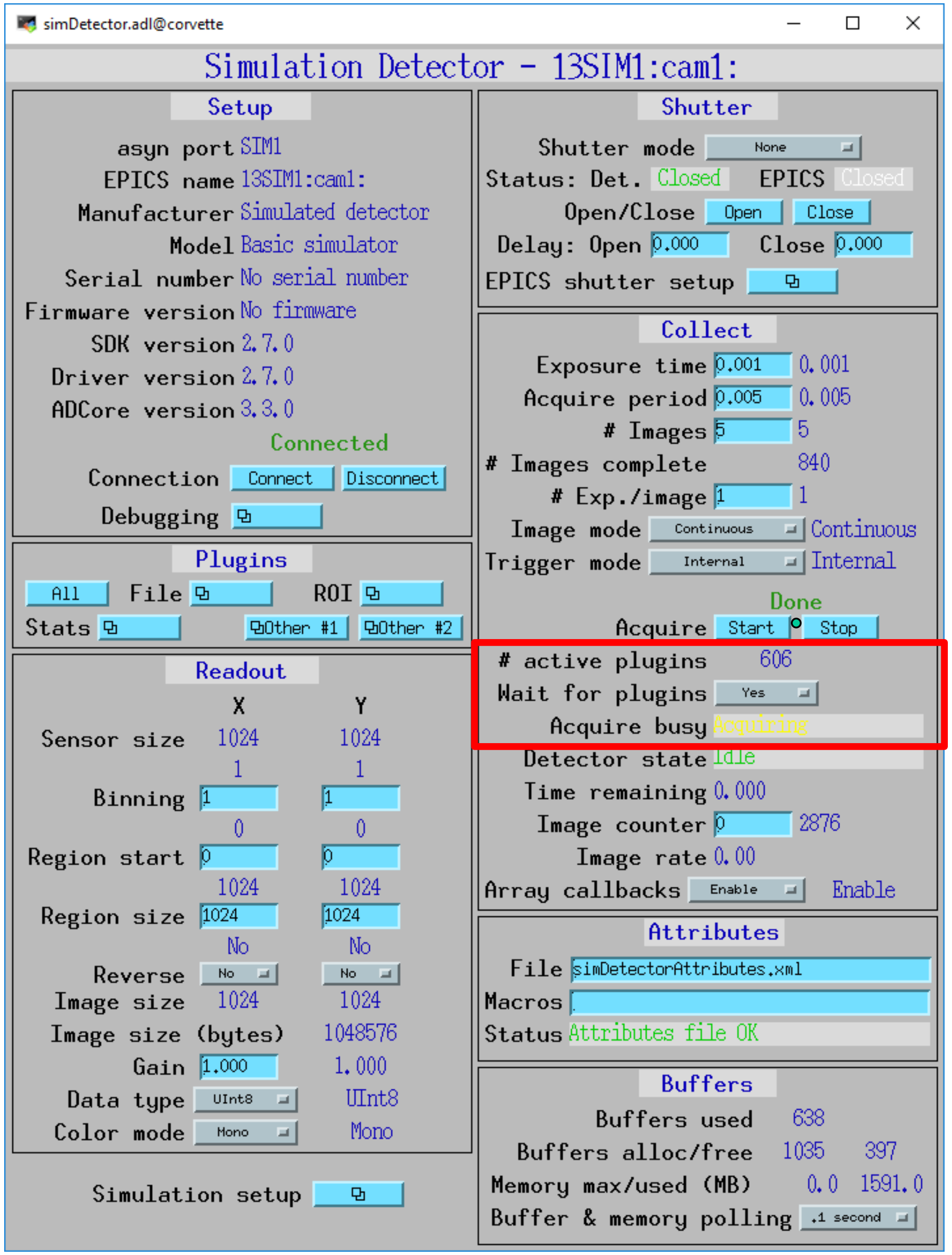
- Previously each plugin used its own NDArrayPool.
    - Problem: not really possible to enforce the maxMemory limits for the driver and plugin chain.
    - Sum of the memory use by the driver and all plugins that matters, not the use by each individual driver and plugin.
- NDPluginDriver base class changed to set its pNDArrayPool pointer to the address passed to it in the NDArray.pNDArrayPool for the NDArray in the callback.
- Ultimately all NDArrays are derived from the driver, either directly, or via the NDArrayPool.copy() or NDArrayPool.convert() methods.
    - Thus plugins now allocate NDArrays from the driver's NDArrayPool, not their own.

Active Plugin Counting and Waiting for Plugins to Complete (R3-3)

- Previously to wait for plugins to complete before the driver indicated that acquisition was complete then needed to set CallbacksBlock=Yes for each plugin in the chain.
- Waiting for plugins is needed in cases like the following
  - Doing a step scan and one of the counters for the step-scan is a PV from the statistics plugin.
    - Necessary to wait for the statistics plugin to complete to be sure the PV value is for current NDArray and not the previous one.
  - Doing a scan and writing the NDArrays to a file with one of the file plugins. N
    - Necessary to wait for the file plugin to complete before changing the file name for the next point.
- Problems with setting CallbacksBlock=Yes.
  - Slows down the driver because the plugin is executing in the driver thread and not in its own thread.
  - Complicated to change all of the required plugin settings from CallbacksBlock=No to CallbacksBlock=Yes.

Active Plugin Counting and Waiting for Plugins to Complete

- NDPluginDriver base class now does the following:
    - Increments a NumActivePlugins counter in the driver that owns each NDArray as it is queued
    - Decrements the counter after the processing is done.
- All drivers have 3 new records:
    - **NumActivePlugins**: Indicates the total number of NDArrays that are currently processing or are queued for processing by this driver.
    - **WaitForPlugins**: Determines whether AcquireBusy waits for NumActivePlugins to go to 0 before changing to 0 when acquisition completes.
    - **AcquireBusy**: “busy” record that is set to 1 when Acquire changes to 1. It changes back to 0 when acquisition completes, i.e. when Acquire_RBV=0.
        - If WaitForPlugins is Yes then it also waits for NumActivePlugins to go to 0 before changing to 0.
- Should now rarely be necessary to change plugins to use CallbacksBlock=Yes.



simDetector.adl@corvette

Simulation Detector - 13SIM1:cam1:

Setup

asyn port	SIM1
EPICS name	13SIM1:cam1:
Manufacturer	Simulated detector
Model	Basic simulator
Serial number	No serial number
Firmware version	No firmware
SDK version	2.7.0
Driver version	2.7.0
ADCore version	3.3.0
	Connected
Connection	Connect / Disconnect
Debugging	

Plugins

All, File, ROI, Stats, Other #1, Other #2

Readout

	X	Y
Sensor size	1024	1024
Binning	1 / 1	1 / 1
Region start	0 / 0	0 / 0
Region size	1024 / 1024	1024 / 1024
Reverse	No / No	No / No
Image size	1024	1024
Image size (bytes)		1048576
Gain	1.000	1.000
Data type	UInt8	UInt8
Color mode	Mono	Mono

Simulation setup

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	0.001	0.001
Acquire period	0.005	0.005
# Images	5	5
# Images complete		840
# Exp./image	1	1
Image mode	Continuous	Continuous
Trigger mode	Internal	Internal
		Done
Acquire	Start	Stop
# active plugins		606
Wait for plugins	Yes	
Acquire busy	Acquiring	
Detector state	Idle	
Time remaining	0.000	
Image counter	0	2876
Image rate	0.00	
Array callbacks	Enable	Enable

Attributes

File	simDetectorAttributes.xml
Macros	
Status	Attributes file OK

Buffers

Buffers used	638	
Buffers alloc/free	1035	397
Memory max/used (MB)	0.0	1591.0
Buffer & memory polling	.1 second	

New records

NDFileTIFF (R3-3)

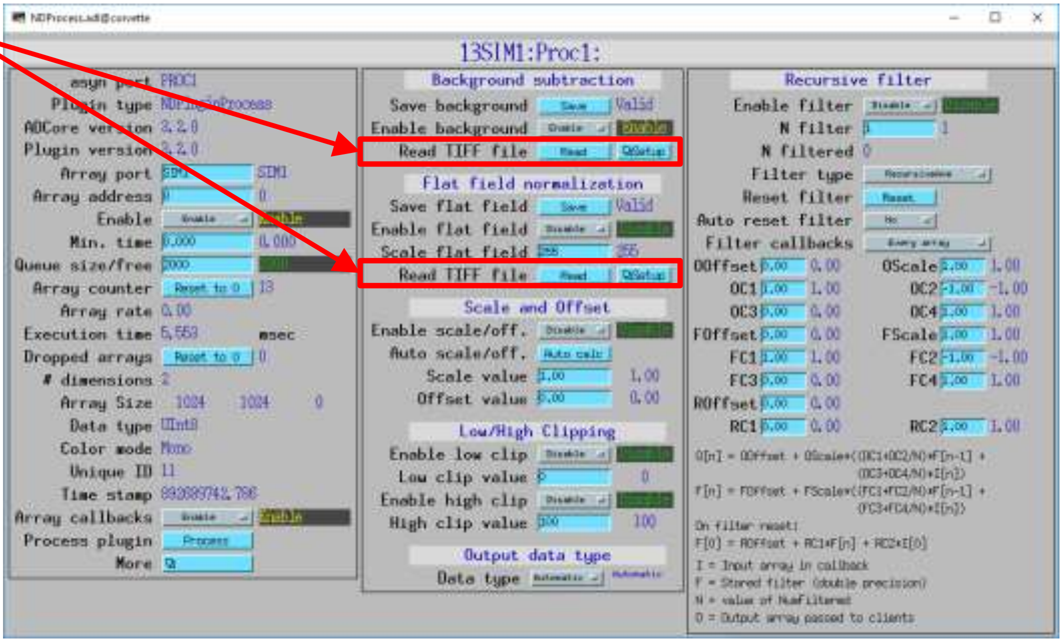
- Added support for readFile() so it is now possible to read a TIFF file into an NDArray using this plugin and do callbacks to downstream plugins.
  - All datatypes (NDDataType_t) are supported.
  - Supports Mono, RGB1, and RGB3 color modes. It also correctly reads files written with RGB2 color mode.
  - Sestores the NDArray fields uniqueID, timeStamp, and epicsTS if they are present.
  - Restores all of the NDArray NDAttributes that were written to the TIFF file.
    - Because of the way the NDAttributes are stored in the TIFF file the restored attributes are all of type NDAttrString, rather than the numeric data types the attributes may have originally used.
- One motivation is for NDPluginProcess to be able to read TIFF files for the background and flat field images, rather than needing to collect them each time it is used.

NDPluginProcess (R3-3)

- Load a dedicated TIFF plugin for the NDPluginProcess plugin in commonPlugins.cmd.
  - Used for reading background or flatfield TIFF files.
- Add an sseq record to load the background image from a TIFF file. Executes all the following steps:
  1. Saves the current NDArrayPort fo the Process plugin to a temporary location
  2. Sets the NDArrayPort to the TIFF plugin.
  3. Enables ArrayCallbacks for the TIFF plugin in case they were disabled.
  4. Process the ReadFile record in the TIFF plugin. This reads the TIFF file and does callback to the Process plugin.
  5. Loads the NDArray from the callback into the background image.
  6. Restores the previous NDArrayPort from the temporary location.
- Add an sseq record to load the flatfile from a TIFF file.
  - Executes the same steps as for the background above, except that in step 5 it loads the NDArray into the flatfile image.

NDPluginProcess (R3-3)

New records



NDProcess.adl@corvette

13SIM1:Proc1:

asyn port	PROC1	
Plugin type	NDPluginProcess	
ADCore version	3.2.0	
Plugin version	3.2.0	
Array port	SIM1	SIM1
Array address	0	0
Enable	Enable	Enable
Min. time	0.000	0.000
Queue size/free	2000	2000
Array counter	Reset to 0	13
Array rate	0.00	
Execution time	5.553	msec
Dropped arrays	Reset to 0	0
# dimensions	2	
Array Size	1024 1024 0	
Data type	UInt8	
Color mode	Mono	
Unique ID	11	
Time stamp	892889742.786	
Array callbacks	Enable	Enable
Process plugin	Process	
More		

Background subtraction

Save background	Save	Valid
Enable background	Disable	Disable
Read TIFF file	Read	Setup

Flat field normalization

Save flat field	Save	Valid
Enable flat field	Disable	Disable
Scale flat field	255	255
Read TIFF file	Read	Setup

Scale and Offset

Enable scale/off.	Disable	Disable
Auto scale/off.	Auto calc	
Scale value	1.00	1.00
Offset value	0.00	0.00

Low/High Clipping

Enable low clip	Disable	Disable
Low clip value	0	0
Enable high clip	Disable	Disable
High clip value	100	100

Output data type

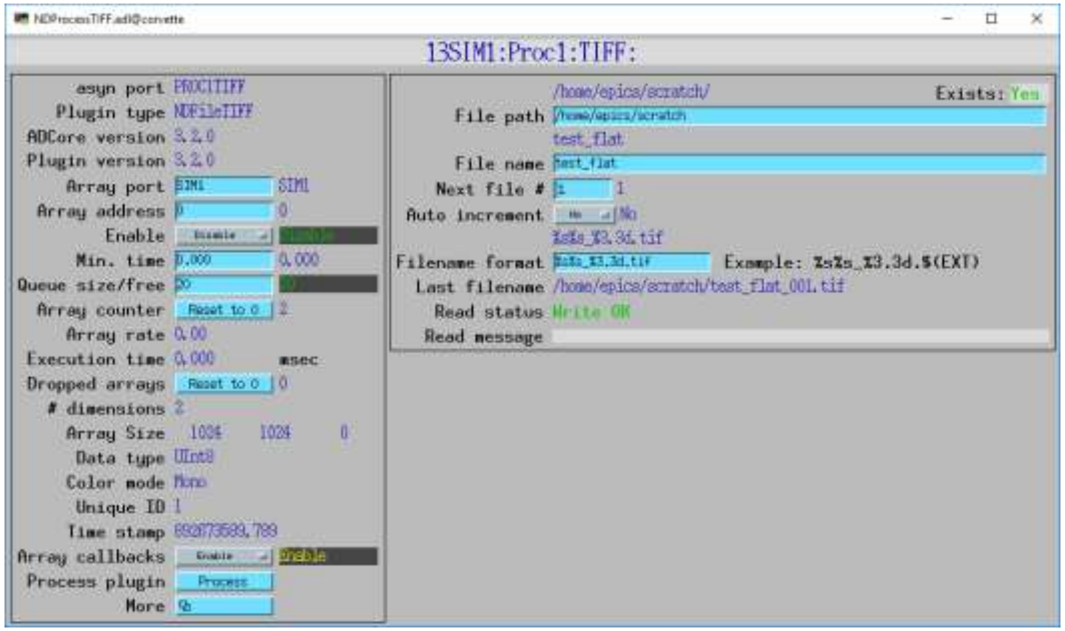
Data type	Automatic	Automatic

Recursive filter

Enable filter	Disable	Disable
N filter	1	1
N filtered	0	
Filter type	RecursiveAve	
Reset filter	Reset	
Auto reset filter	No	
Filter callbacks	Every array	

OOffset	0.00	0.00	OScale	1.00	1.00
OC1	1.00	1.00	OC2	-1.00	-1.00
OC3	0.00	0.00	OC4	1.00	1.00
FOffset	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
ROffset	0.00	0.00			
RC1	0.00	0.00	RC2	1.00	1.00

O[n] = OOffset + OScale*((OC1+OC2/N)*F[n-1] +
(OC3+OC4/N)*I[n])
F[n] = FOffset + FScale*((FC1+FC2/N)*F[n-1] +
(FC3+FC4/N)*I[n])
On filter reset:
F[0] = ROffset + RC1*F[n] + RC2*I[0]
I = Input array in callback
F = Stored filter (double precision)
N = value of NumFiltered
O = Output array passed to clients



NDProcessTIFF.adl@corvette

13SIM1:Proc1:TIFF:

asyn port	PROC1TIFF	
Plugin type	NDFileTIFF	
ADCore version	3.2.0	
Plugin version	3.2.0	
Array port	SIM1	SIM1
Array address	0	0
Enable	Disable	Disable
Min. time	0.000	0.000
Queue size/free	20	20
Array counter	Reset to 0	2
Array rate	0.00	
Execution time	0.000	msec
Dropped arrays	Reset to 0	0
# dimensions	2	
Array Size	1024 1024 0	
Data type	UInt8	
Color mode	Mono	
Unique ID	1	
Time stamp	892873589.789	
Array callbacks	Enable	Enable
Process plugin	Process	
More		

	/home/epics/scratch/	Exists: Yes
File path	/home/epics/scratch	
	test_flat	
File name	test_flat	
Next file #	1	1
Auto increment	No	No
	%s%s_%3.3d.tif	
Filename format	%s%s_%3.3d.tif	Example: %s%s_%3.3d.$(EXT)
Last filename	/home/epics/scratch/test_flat_001.tif	
Read status	Write OK	
Read message		

NDFileHDF5 (R3-2)

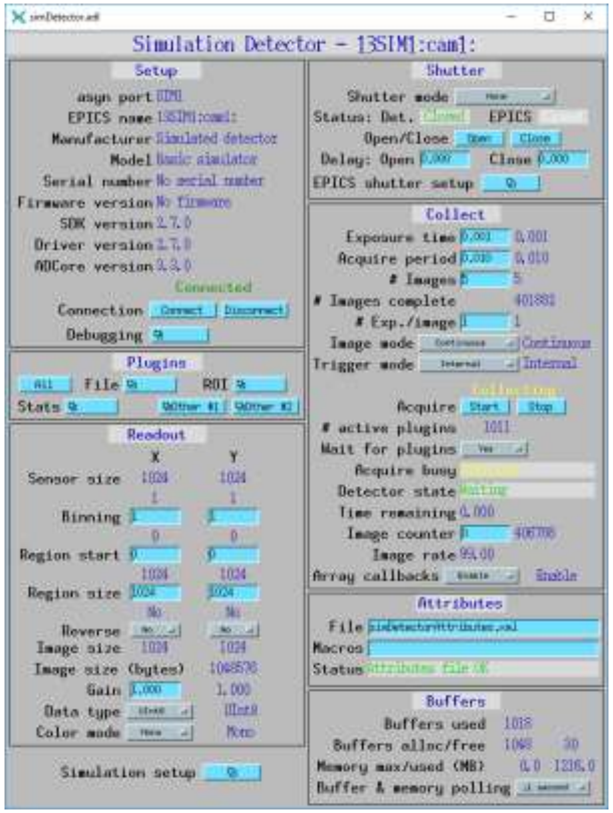
- Added support for blosc compression library.
  - Compressors include blosclz, lz4, lz4hc, snappy, zlib, and zstd.
  - Also support for ByteSuffle and BitShuffle.
  - ADSupport now contains the blosc library, so it is available for most architectures.
  - The build flags WITH_BLOSC, BLOSC_EXTERNAL, and BLOSC_LIB have been added, similar to other optional libraries. Thanks to Xiaoqiang Wang for this addition.
- Changed all output records in NDFileHDF.template to have PINI=YES. This is how other plugins all work.

Operator displays

medm, edm, caQtDM, CSS-BOY (R3-2)

- Added ADApp/op/Makefile.
  - Runs the conversion tools to convert the medm adl files to edl for edm, ui for caQtDM, and opi for CSS-BOY.
- Lightning talk on this later today.

medm



simDetector.adl

Simulation Detector - 13SIM1:cam1:

Setup
asyn port SIM1
EPICS name 13SIM1:cam1:
Manufacturer Simulated detector
Model Basic simulator
Serial number No serial number
Firmware version No firmware
SDK version 2.7.0
Driver version 2.7.0
ADCore version 3.3.0
Connected
Connection Connect | Disconnect
Debugging

Plugins
All | File | ROI
Stats | Other #1 | Other #2

Readout

	X	Y
Sensor size	1024	1024
	1	1
Binning	1	1
	0	0
Region start	0	0
	1024	1024
Region size	1024	1024
	No	No
Reverse	No	No
Image size	1024	1024
Image size (bytes)		1048576
Gain	1.000	1.000
Data type	UInt8	UInt8
Color mode	Mono	Mono

Simulation setup

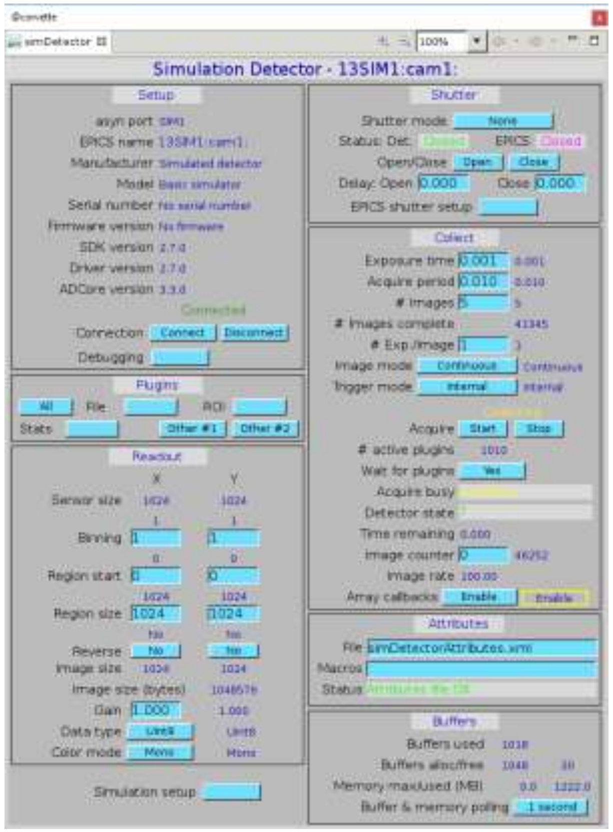
Shutter
Shutter mode None
Status: Det. Closed EPICS
Open/Close Open | Close
Delay: Open 0.000 Close 0.000
EPICS shutter setup

Collect
Exposure time 0.001 0.001
Acquire period 0.010 0.010
# Images 5 5
# Images complete 401881
# Exp./image 1 1
Image mode Continuous Continuous
Trigger mode Internal Internal
Collecting
Acquire Start | Stop
# active plugins 1011
Wait for plugins Yes
Acquire busy
Detector state Waiting
Time remaining 0.000
Image counter 0 406706
Image rate 99.00
Array callbacks Enable Enable

Attributes
File simDetectorAttributes.xml
Macros
Status Attributes file OK

Buffers
Buffers used 1018
Buffers alloc/free 1048 30
Memory max/used (MB) 0.0 1216.0
Buffer & memory polling 1 second

CSS-BOY



CSS-BOY

simDetector

100%

Simulation Detector - 13SIM1:cam1:

Setup
asyn port SIM1
EPICS name 13SIM1:cam1:
Manufacturer Simulated detector
Model Basic simulator
Serial number No serial number
Firmware version No firmware
SDK version 2.7.0
Driver version 2.7.0
ADCore version 3.3.0
Connected
Connection Connect | Disconnect
Debugging

Plugins
All | File | ROI
Stats | Other #1 | Other #2

Readout

	X	Y
Sensor size	1024	1024
	1	1
Binning	1	1
	0	0
Region start	0	0
	1024	1024
Region size	1024	1024
	No	No
Reverse	No	No
Image size	1024	1024
Image size (bytes)		1048576
Gain	1.000	1.000
Data type	UInt8	UInt8
Color mode	Mono	Mono

Simulation setup

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 0.001 0.001
Acquire period 0.010 0.010
# Images 5 5
# Images complete 41345
# Exp./image 1 1
Image mode Continuous Continuous
Trigger mode Internal Internal
Collecting
Acquire Start | Stop
# active plugins 1010
Wait for plugins Yes
Acquire busy
Detector state
Time remaining 0.000
Image counter 0 46252
Image rate 100.00
Array callbacks Enable Enable

Attributes
File simDetectorAttributes.xml
Macros
Status Attributes file OK

Buffers
Buffers used 1018
Buffers alloc/free 1048 30
Memory max/used (MB) 0.0 1222.0
Buffer & memory polling 1 second

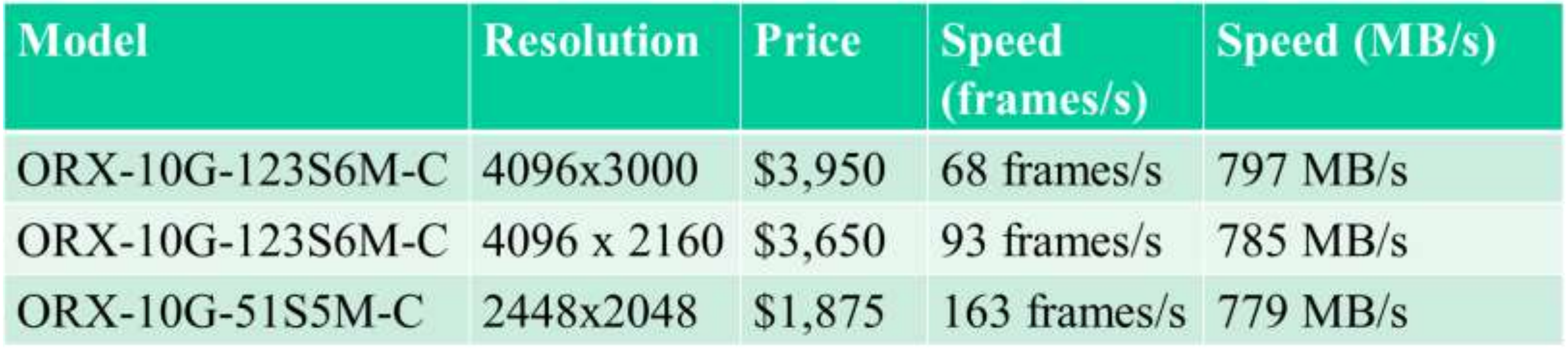
Other Changes (R3-3)

- NDArrayPool Enhancements
  - Changes to allow inheriting it from derived classes. Thanks to Sinesa Veseli for this.
  - Optimization to memory allocation mechanism. Original work by Michael Huth. I am currently modifying to use std::multiset, same as used for plugin output sorting.

- ntndArrayConverter.cpp
  - Added conversion of the NDArray.timeStamp and NDArray.epicsTS fields from EPICS epoch (Jan. 1 1990) to Posix epoch (Jan. 1, 1970).
  - Needed because NDArrays use EPICS epoch but pvAccess uses Posix epoch and the timestamps shown by pvGet were incorrect for the NTNDArrays.

Point Grey 10-Gbit Ethernet Camera Oryx ORX-10G-51S5C-C

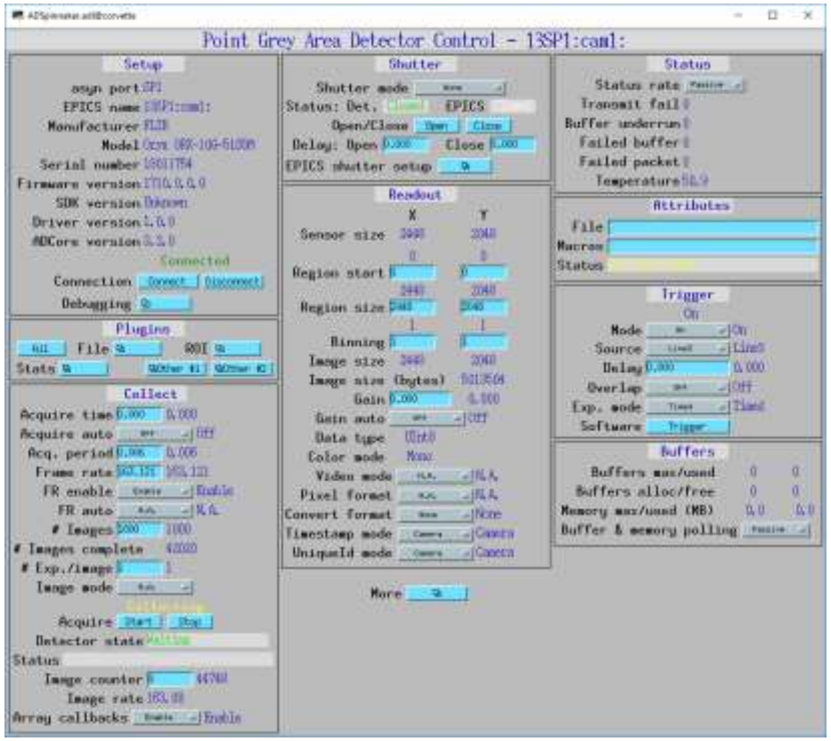
- 2448 x 2048 global shutter CMOS
- Sony IMX250 2/3”
- Dynamic range of 72 dB
- Peak QE of 62%
- Read noise of 2.2e-
- 12-bit, 10-bit, or 8-bit data
- Max frame rate of 162 fps
  - 779 MB/S, >8X faster than GigE
- $1,875



Model	Resolution	Price	Speed (frames/s)	Speed (MB/s)
ORX-10G-123S6M-C	4096x3000	$3,950	68 frames/s	797 MB/s
ORX-10G-123S6M-C	4096 x 2160	$3,650	93 frames/s	785 MB/s
ORX-10G-51S5M-C	2448x2048	$1,875	163 frames/s	779 MB/s

ADSpinnaker

- New driver for Point Grey GeniCAM cameras using their Spinnaker SDK (10 GigE, GigE, USB-3)
  - Currently working on Windows
  - Linux requires Ubuntu 16 (gcc 5.4, special release of ffmpeg)
- Some work beginning on aravisGigE driver
  - Guabao Shen at APS and Neil O’Brien at Diamond)



asyn port SP1
EPICS name 13SP1:cam1:
Manufacturer FLIR
Model Oryx ORX-10G-51S5M
Serial number 18011754
Firmware version 1710.0.0.0
SDK version Unknown
Driver version 1.0.0
ADCore version 3.1.0
Connected
Connection
Debugging

File
ROI
Stats

Acquire time 0.000 0.000
Acquire auto Off Off
Acq. period 0.006 0.006
Frame rate 162.121 165.121
FR enable Enable Enable
FR auto N.A. N.A.
Images 5000 1000
Images complete 42000
Exp./image 1 1
Image mode N.A.
Acquire
Detector state Waiting
Status
Image counter 0 44760
Image rate 163.00
Array callbacks Enable Enable

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

Sensor size 2448 2048
Region start 0 0 0 0
Region size 2448 2048 2448 2048
Binning 1 1 1 1
Image size 2448 2048
Image size (bytes) 5013504
Gain 0.000 0.000
Gain auto Off Off
Data type UInt8
Color mode Mono
Video mode N.A. N.A.
Pixel format N.A. N.A.
Convert format None None
Timestamp mode Camera Camera
UniqueId mode Camera Camera

Status rate Passive
Transmit fail 0
Buffer underrun 0
Failed buffer 0
Failed packet 0
Temperature 50.9

File
Macros
Status

On
Mode On On
Source Line0 Line0
Delay 0.000 0.000
Overlap Off Off
Exp. mode Timed Timed
Software

Buffers max/used 0 0
Buffers alloc/free 0 0
Memory max/used (MB) 0.0 0.0
Buffer & memory polling Passive

Roadmap: ADCore R4-0 ??

- Put more functionality into ADDriver base class
  - Derived class would call ADDriver::doPluginCallbacks(), which would handle setting attributes, getting timestamp, calling plugins, etc.
- Simplify file saving modes (no more Single, Capture, Stream) and eliminate AutoSave
- Add flag to prevent overwriting files
- Support compression in NDArrays

Roadmap: ADCore R5-0 ??

- Change NDArray to NTNDArray for passing data to plugins
- Use PVDatabase
  - "local" provider within IOC
  - "pva" provider between IOCs
- Smart pointers automatically eliminate all unnecessary copying
- Eliminates need for NDPluginPva
- V4 clients can immediately receive data from any point in plugin chain
- Distribute load to multiple IOCs without pvaDriver
- Bruno Martins has demonstrated this working for ADSimDetector and NDPluginStats