



Hardware Triggered Scanning: EPICS Layer

Philip Taylor, Emma Arandjelovic
Observatory Sciences Limited



Software Stack: Reminder



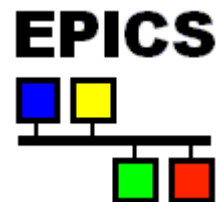
Data Analysis WorkbeNch
- Analysis and visualization



Generic Data Acquisition
- Experiment setup and supervision



Malcolm
- Scan configuration



Experimental Physics &
Industrial Control System
- Low level control of hardware



EPICS Responsibilities

1. Motion control interface
 2. Detector setup
 3. File saving
- On training rigs, a single IOC **BLxx-EA-IOC-01** takes care of everything
 - Beamlines may use multiple IOCs



Motion Control

- [EPICS pmac module](#) takes care of:
 - Co-ordinate system configuration
 - Sending trajectory profiles to the controller
- Motion parameters defined in a [motor record](#) for each axis:
 - MRES: motor resolution
 - VMAX: maximum velocity
 - ACCL: acceleration time





Coordinate system

Axis Status - BL49P-MO-STEP-01													
	Description	CS No	CS Port Name		CS Assignment		Direct Demand		Amplifier				
1	MOTORS M1 X	1	None	CS1		X	0.000	19.800		Kill			
2	MOTORS M2 A	1	None	CS1		A	0.000	-20.250		Kill			
3		P-01:M		STEP-01:M3:	<Bl	P-01:I	<BL49P-	O-STEP-01					
4		P-01:M		STEP-01:M4:	<Bl	P-01:I	<BL49P-	O-STEP-01					



AreaDetector

- EPICS framework for controlling detectors
 - Provides a standard, general-purpose interface
 - Supports a wide variety of detectors
 - Real time data analysis and processing
 - Configurable plugin architecture
 - Data passed through plugin chain in an **NDArray**
 - N-dimensions (up to 10) with attached metadata (**NDAttributes**) associated with the frame

<http://cars9.uchicago.edu/software/epics/areaDetector.html>



AD Architecture

Source: Mark Rivers, [Using AreaDetector](#)

Layer 6
EPICS CA clients

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

Layer 5
Standard
EPICS records

ADBase .template xxxDriver .template NDPluginBase .template NDPluginXXX .template

Layer 4
EPICS device support

Standard asyn device support
(device-independent)

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

Layer 3
Plug-ins

StdArrays Process ROI File
(netCDF, TIFF, JPEG,
HDF5)

Layer 2
Device drivers

Driver

Layer 1
Hardware API

Vendor API

Hardware

- Channel access
- Record/device support
- asynInt32, Float64, Octet
- asynXXXArray
- asynGenericPointer (NDArray)
- C library calls

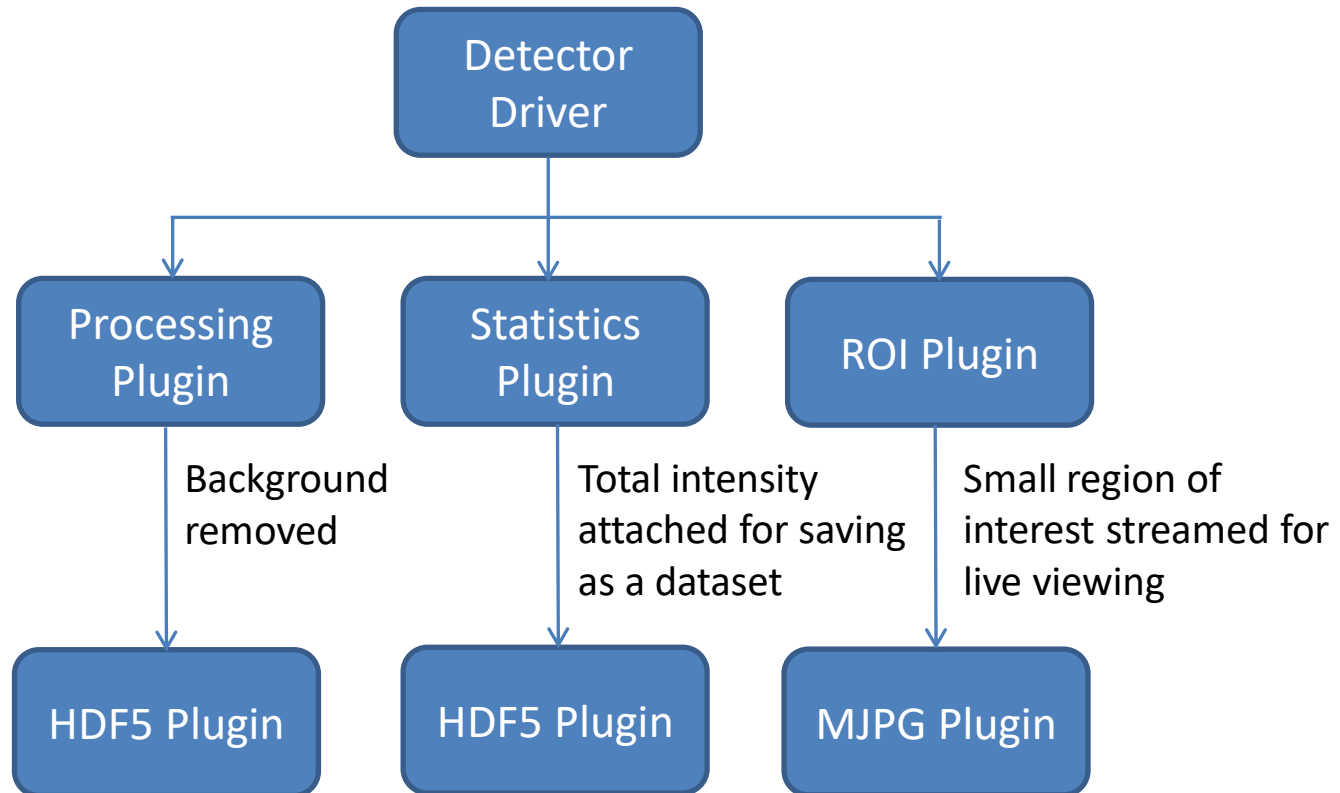


AD Plugins

- Plugins receive a copy of each NDArray via a callback
- They can in turn be sources of NDArray callbacks
- Plugins can also attach NDAttributes to the data
- Chain can be 'rewired' on the fly and plugins enabled/disabled



Plugin Chain Example





Detector Setup

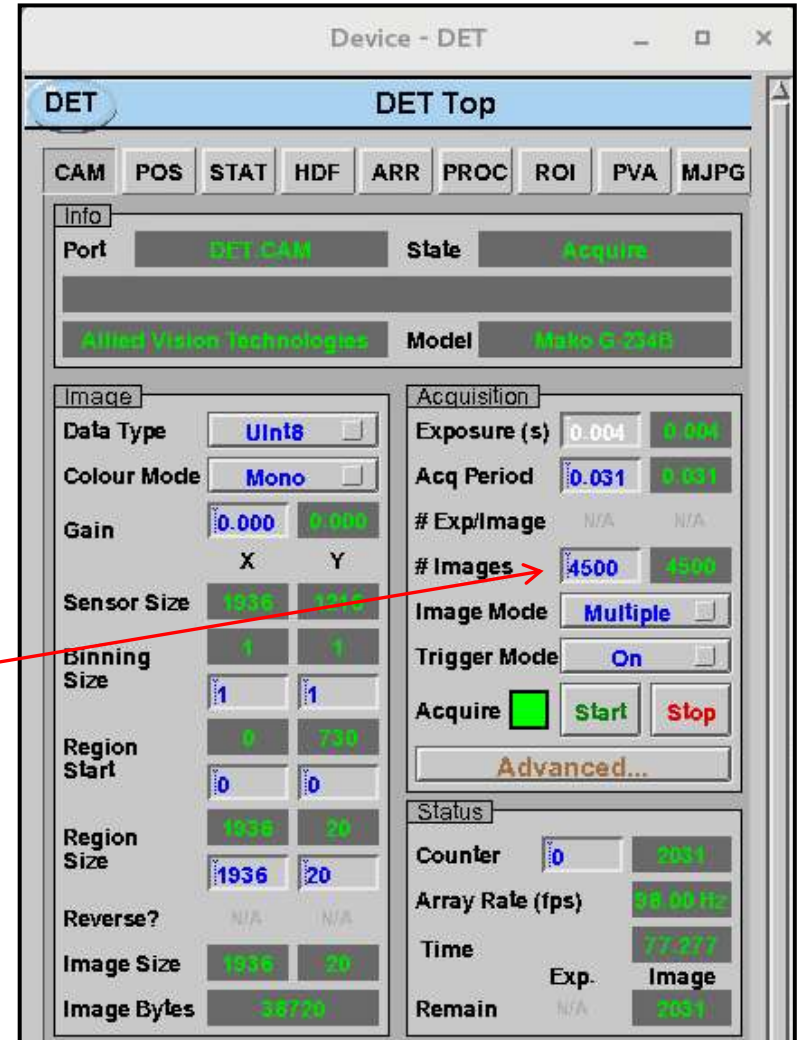
- Each detector has an AreaDetector driver
- At scan start, this sets up parameters such as:
 - Exposure time

The screenshot shows the "Device - DET" software window. It has a tabbed interface with "DET" selected. Below the tabs are sections for "Info", "Image", "Acquisition", and "Status". The "Info" section shows "Port: DET-CAM" and "State: Acquire". The "Image" section shows "Data Type: UInt8", "Colour Mode: Mono", "Gain: 0.000", "Sensor Size: 1936 x 1216", "Binning Size: 1 x 1", "Region Start: 0 x 736", "Region Size: 1936 x 20", "Reverse?: N/A", "Image Size: 1936 x 20", and "Image Bytes: 38720". The "Acquisition" section shows "Exposure (s): 0.004", "Acq Period: 0.031", "# Exp/Image: N/A", "# Images: 4500", "Image Mode: Multiple", "Trigger Mode: On", and "Acquire" status with "Start" and "Stop" buttons. The "Status" section shows "Counter: 0", "Array Rate (fps): 98.00 Hz", "Time: 77.277", and "Remain: 2631". A red arrow points from the "Exposure time" text in the list to the "Exposure (s)" field in the "Acquisition" section.



Detector Setup

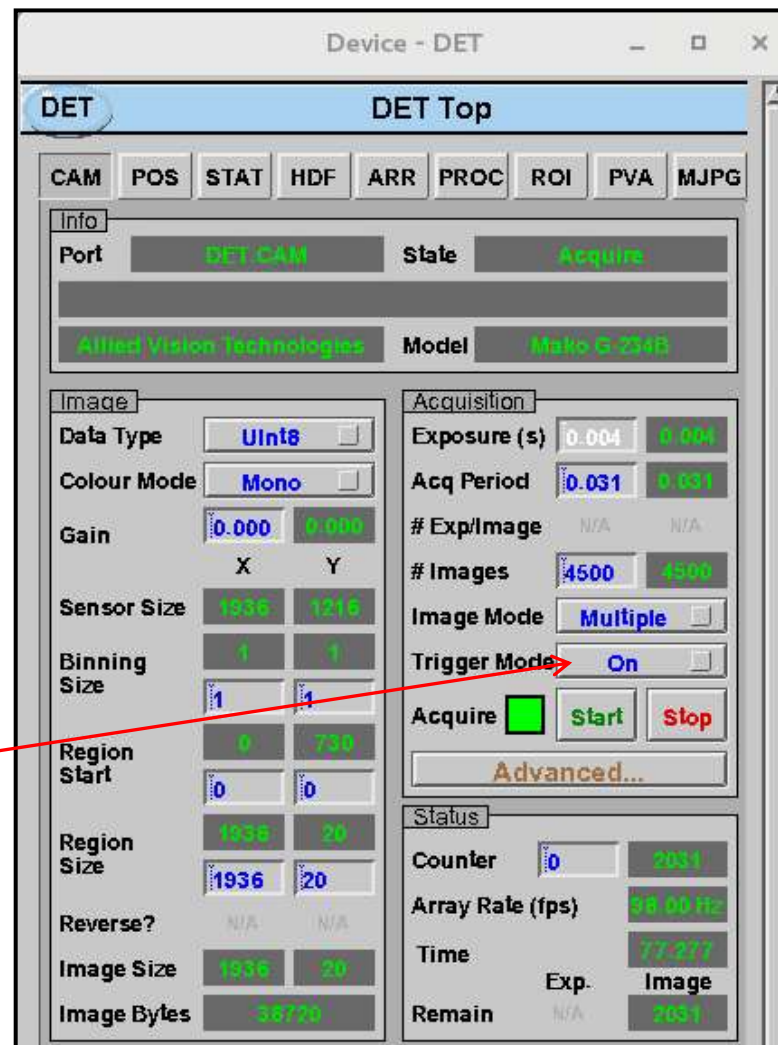
- Each detector has an AreaDetector driver
- At scan start, this sets up parameters such as:
 - Number of images





Detector Setup

- Each detector has an AreaDetector driver
- At scan start, this sets up parameters such as:
 - Trigger mode





Detector Setup

- Each detector has an AreaDetector driver
- At scan start, this sets up parameters such as:

– Frame size

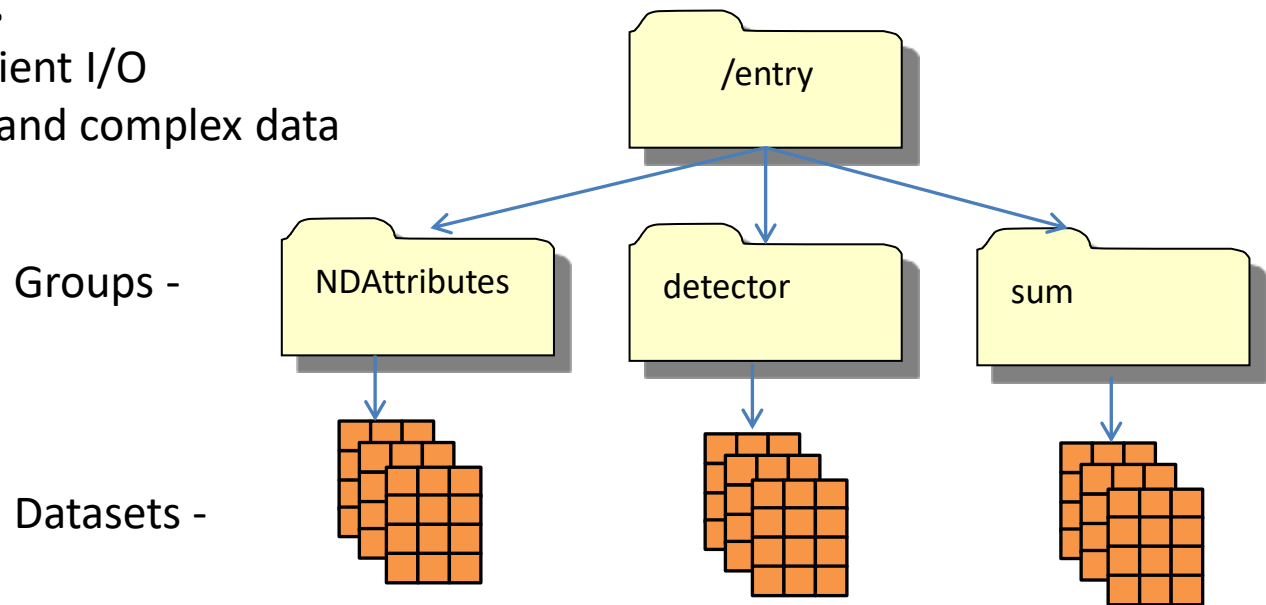
The screenshot shows the 'Device - DET' window with the following sections:

- DET Top** (Tabbed interface with CAM, POS, STAT, HDF, ARR, PROC, ROI, PVA, MJPG)
- Info**: Port: DET-CAM, State: Acquire, Manufacturer: Allied Vision Technologies, Model: Mako G-234B
- Image**: Data Type: UInt8, Colour Mode: Mono, Gain: 0.000, Sensor Size: 1936 x 1216, Binning Size: 1 x 1, Region Start: 0 x 736, Region Size: 1936 x 20 (highlighted with a red arrow), Reverse?: N/A, Image Size: 1936 x 20, Image Bytes: 38720
- Acquisition**: Exposure (s): 0.004, Acq Period: 0.031, # Exp/Image: N/A, # Images: 4500, Image Mode: Multiple, Trigger Mode: On, Acquire: Start/Stop buttons, Advanced... button
- Status**: Counter: 0, Array Rate (fps): 38.00 Hz, Time: 77.277, Remain: 2031



Data Files

- Data is saved in the hierarchical [HDF](#) format
- Designed for:
 - Flexible, efficient I/O
 - high volume and complex data

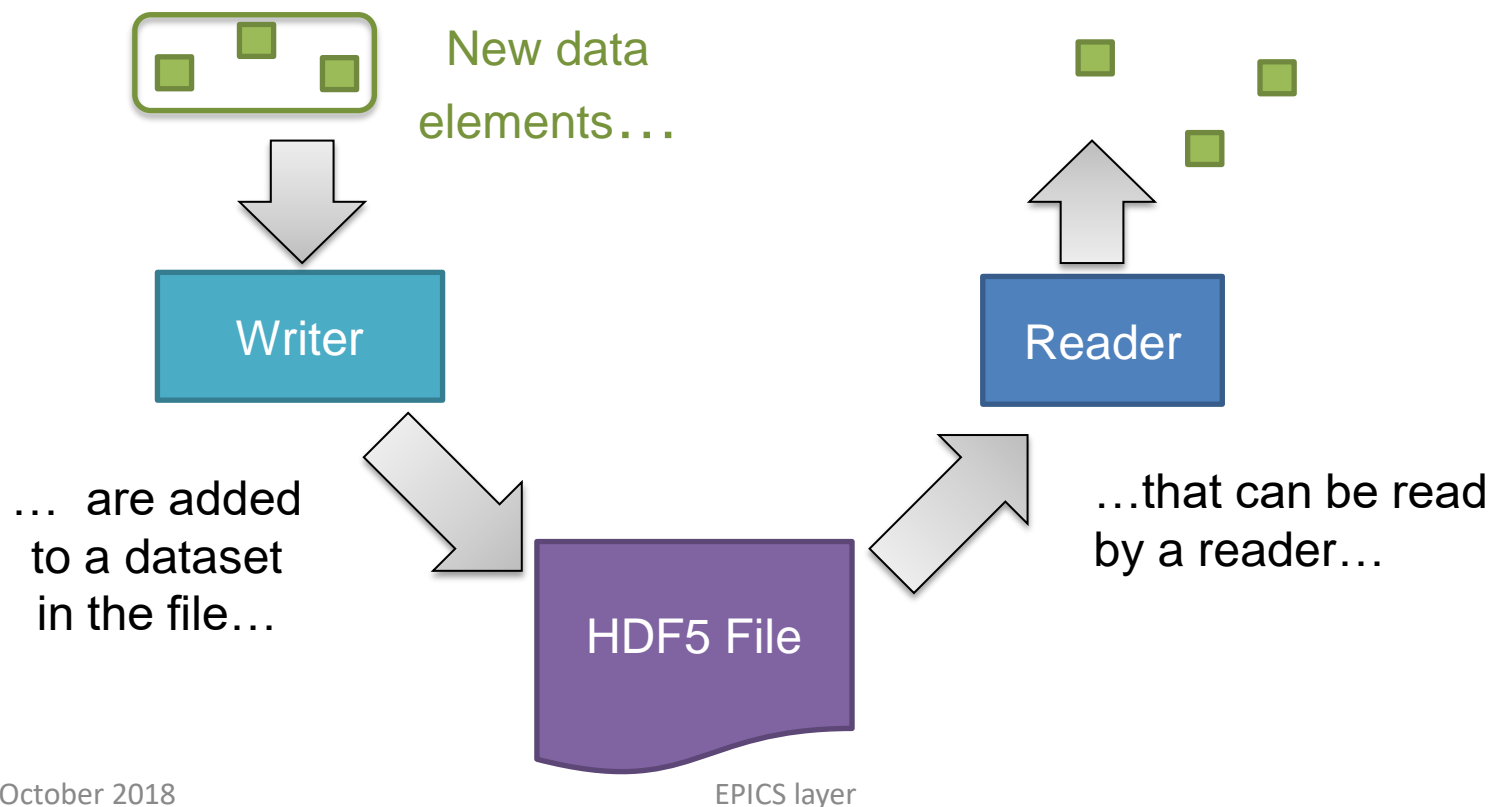


- Metadata is stored in the form of user-defined, named attributes attached to *groups* and *datasets*



SWMR Mode

- Single Writer Multiple Reader
- Process data as it's written to file
- Requires POSIX compliant file system (N.B. NFS is NOT POSIX compliant)





HDF5 Considerations

- *Flush rate* determines how often data is flushed to disk (controlled by Malcolm)
- HDF Datasets are *chunked*: divided into equally sized blocks
 - Each chunk stored contiguously in the file
 - Avoids need to read in the whole file
 - Allows datasets to be extendible
 - Enables compression
 - Chunk size is configured in the HDF5 plugin
- See [HDF5 Advanced Topics Tutorial](#)



Read/Write Performance

- Performance **GREATLY (up to 1000 times!!)** affected by:
 - Chunking settings
 - Compression
 - Flush rate
- Settings must be optimised for the experiment
- Consider BOTH read and write performance
- This is a complex issue!
- **Never set a chunk size of 1!**



File Saving

DET Top

Information

Port Name: DET.HDF

Plugin Type: NDFileHDF5 ver1.10.1

Input

Array Port: DET.STAT

Callback

Enable: ☒ Enable

Min Time: 0.000 s

Blocking: ☒ No

Counters

Arrays: 0 / 4096

Array Rate (fps): 3.00 Hz

Queue: 0 / 1000

Dropped: 0 / 0

Array Dims

dimensions: 2

Dim0 Size: 1935

Dim1 Size: 20

Dim2 Size: 0

Array Colour

Data Type: UInt8

Colour Mode: Mono

Bayer Pattern: RGGB

Array Time

Unique ID: 4500

Time Stamp: 34600.651

- [NDFileHDF5](#) plugin saves the files to disk
- Two files are generated:
 - <name>-DET.h5 – detector data
 - <name>-PANDABOX.h5 – position data
- These are saved in a folder for each scan



File Saving

A screenshot of the "DET Top" configuration window. The window has a title bar "DET Top" and a menu bar with "CAM", "POS", "STAT", "HDF", "ARR", "PROC", "ROI", "PVA", and "MJPG". The "HDF" menu is selected. The window is divided into several sections: "Information" (Port Name: DET.HDF, Plugin Type: NDFileHDF5 ver1.10.1), "Input" (Array Port: DET.STAT), "Callback" (Enable: ☒, Min Time: 0.000 s, Blocking: ☐), "Counters" (Arrays: 0, Array Rate (fps): 3.00 Hz, Queue: 0, Dropped: 0), "Array Dims" (# dimensions: 2, Dim0 Size: 1935, Dim1 Size: 30, Dim2 Size: 0), "Array Colour" (Data Type: UInt8, Colour Mode: Mono, Bayer Pattern: RGGB), and "Array Time" (Unique ID: 4500, Time Stamp: 34600.651). The "Enable" checkbox in the "Callback" section is highlighted with a red rectangle.

- [NDFileHDF5](#) plugin saves the files to disk
- Two files are generated:
 - <name>-DET.h5 – detector data
 - <name>-PANDABOX.h5 – position data
- These are saved in a folder for each scan
- Callbacks must be *enabled*



File Saving

DET DET Top

CAM POS STAT HDF ARR PROC ROI PVA MJPG

Information

Port Name DET.HDF

Plugin Type HDFFileHDF5 ver1.10.1

Input

Array DET.STAT

Port DET.STAT

Callback

Enable ☐

Min Time 0.000 s 0.000 s

Blocking No ☐

Counters

Arrays 0 1450

Array Rate (fps) 9.00 Hz

Queue 0 1000

Dropped 0 0

Array Dims

dimensions 2

Dim0 Size 1935

Dim1 Size 20

Dim2 Size 0

Array Colour

Data Type UInt8

Colour Mode Mono

Bayer Pattern RGGB

Array Time

Unique ID 4500

Time Stamp 34600.651

- Images are buffered into a queue
- If queue gets full -> frames are dropped!



File Saving Configuration

- Chunking parameters configured manually
 - Defaults to the frame size if left at 0
- HDF file structure defined in XML
 - Configured by Malcolm
 - Stored in the same directory as the scan data
 - Deleted when the scan ends

A screenshot of a software window titled "HDF5 details" with the subtitle "BL47P-EA-MAP-01:DET:HDF5:". The window has a light blue header and a white body. It contains two main sections: "XML Layout File" and "Chunking configuration". The "XML Layout File" section has a label "Full path of filename" and a text input field containing the path "/dls_sw/p47/software/gda_versions/master/workspace_git/gda_data_non_live/". Below the input field is a green square button. The "Chunking configuration" section has three rows of input fields: "Number of rows per chunk:" with a value of 0 and a green "20" next to it; "Columns per chunk:" with a value of 0 and a green "1056" next to it; and "Frames (cached) per chunk:" with a value of 0 and a green "1" next to it.

/dls_sw/prod/R3.14.12.3/support/ADCore/2-... - □ x

HDF5 details
BL47P-EA-MAP-01:DET:HDF5:

XML Layout File

Full path of filename

/dls_sw/p47/software/gda_versions/master/workspace_git/gda_data_non_live/

/dls_sw/p47/software/gda_versions/master/workspace_git/gda_data_non_live/

Chunking configuration

Number of rows per chunk:	0	20
Columns per chunk:	0	1056
Frames (cached) per chunk:	0	1



File Structure

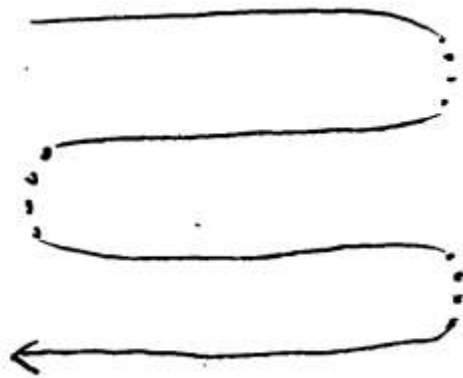
- The STATS plugin computes the total sum and attaches it as an NDAttribute
- This is used to create the *sum* dataset
- XML definition:

```
<dataset name="sum" ndattribute="STATS_TOTAL" source="ndattribute" />
```



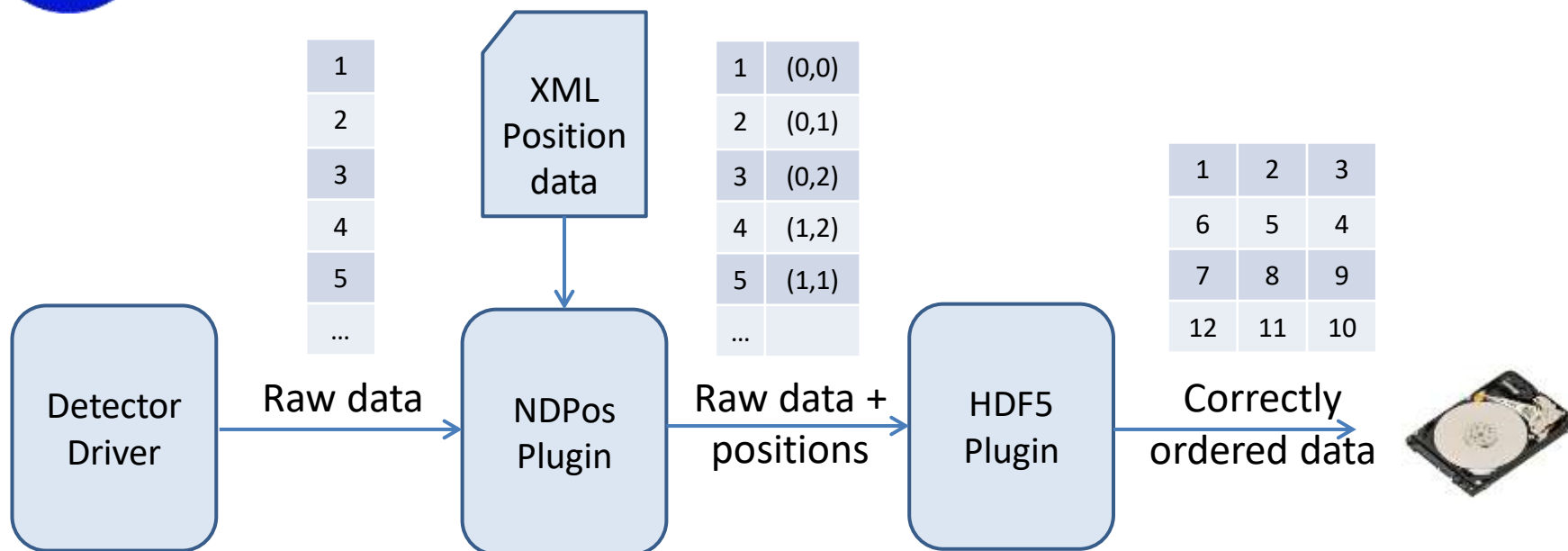
Frame Ordering

Remember: In a snake scan, alternate rows are captured backwards!



1	2	3
6	5	4
7	8	9
12	11	10

Solution



- The [NDPosPlugin](#) intercepts the image frame and attaches the corresponding position as an attribute
- The file saving plugin reads the position attributes and ensures the frames are stored in the correct order



Unique Keys

When a dataset is extended, a 'fill' value is used

Q: How do we distinguish between fill and real data?

A: A special *Unique Keys* dataset is created:

- Same size as the scan dataset, with fill value 0
- A non-zero value is written when the corresponding scan data has been flushed to disk
- DAWN monitors this data set in order to know when to start processing a scan point

h5watch



- Command line tool for watching data added to a dataset
- `--dim` option monitors for dimension size changes

h5watch --dim p45-164-DET.h5/entry/NDAttributes/NDArrayUniqueld

.h5 filename

HDF5 path to dataset

Opened "p45-164-DET.h5" with sec2 driver.

Monitoring dataset

/entry/NDAttributes/NDArrayUniqueld...

dimension 0: 46->48 (increases)

dimension 1: 180->180 (unchanged)

dimension 2: 1->1 (unchanged)

dimension 3: 1->1 (unchanged)



IOC Utilities

- To reboot an IOC:
 - console <IOC name> (remote IOCs)
 - ioc-connect <IOC name> (test rig)
 - Ctrl-X
- To check what module versions are used:
 - configure-ioc show <IOC name>
 - Check the configure/RELEASE file

```
[p49user@p49-pw001 ~]$ configure-ioc show BL49P-EA-IOC-01
BL49P-EA-IOC-01 /dls_sw/work/R3.14.12.7/support/BL46P-BUILDER/iocs/BL49P-EA-IOC-01/bin/linux-x86_64/stBL49P-EA-IOC-01.sh
[p49user@p49-pw001 ~]$
[p49user@p49-pw001 ~]$ cat /dls_sw/work/R3.14.12.7/support/BL46P-BUILDER/iocs/BL49P-EA-IOC-01/configure/RELEASE
```



IOC RELEASE File

Common prefixes

SUPPORT = /dls_sw/prod/R3.14.12.3/support

WORK = /dls_sw/work/R3.14.12.3/support

Module definitions

ADCORE = \$(SUPPORT)/ADCore/2-6dls5

ADPANDABLOCKS = \$(SUPPORT)/ADPandABlocks/2-0

ADSUPPORT = \$(SUPPORT)/ADSupport/1-2

ARAVISGIGE = \$(SUPPORT)/aravisGigE/2-1dls9

} AreaDetector modules

ASYN = \$(SUPPORT)/asyn/4-31

BL46P_BUILDER = \$(WORK)/BL46P-BUILDER

BUSY = \$(SUPPORT)/busy/1-6-1dls1

CALC = \$(SUPPORT)/calc/3-1

FFMPEGSERVER = \$(SUPPORT)/ffmpegServer/3-1dls1-5

MOTOR = \$(SUPPORT)/motor/6-9dls14

NORMATIVETYPESCPP = \$(SUPPORT)/normativeTypesCPP/5-0-2

PMAC = \$(SUPPORT)/pmac/2-1

PROCSERVCONTROL = \$(SUPPORT)/procServControl/1-17-1

PVACCESSCPP = \$(SUPPORT)/pvAccessCPP/4-1-3

PVDATABASECPP = \$(SUPPORT)/pvDatabaseCPP/4-1-1

PVDATA CPP = \$(SUPPORT)/pvDataCPP/5-0-4

EPICS Base appears last

EPICS_BASE = /dls_sw/epics/R3.14.12.3/base