Hardware Triggered Scanning: EPICS Layer

Philip Taylor, Emma Arandjelovic

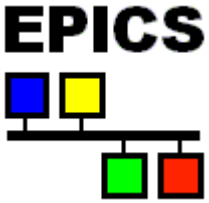
Observatory Sciences Limited

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

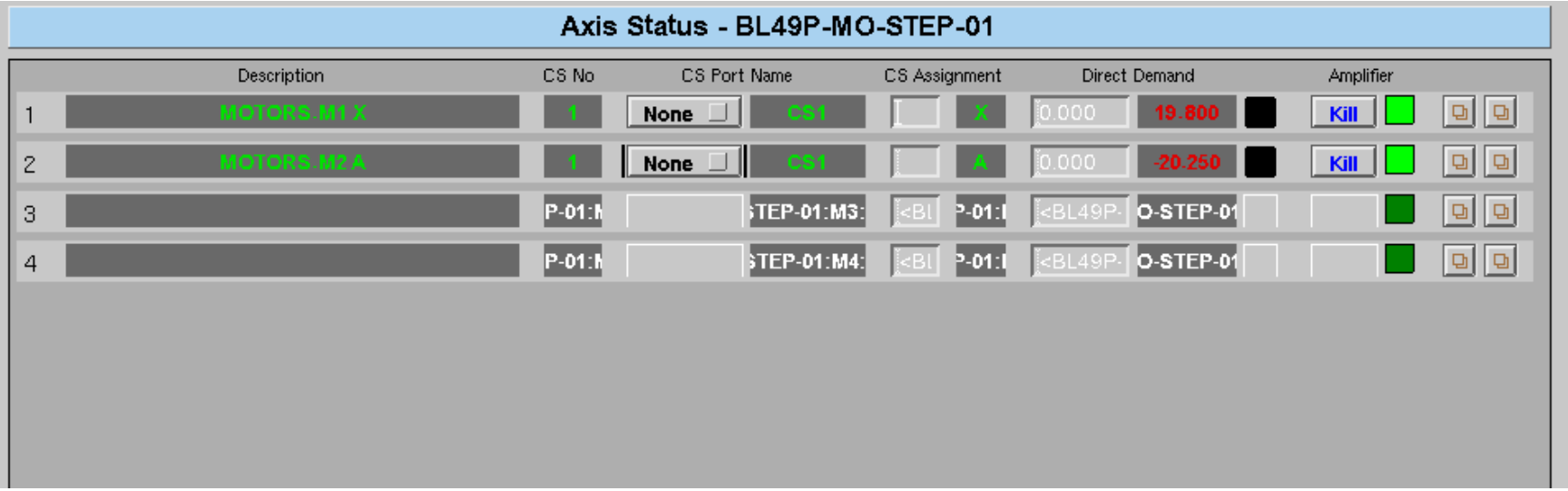
1. Motion control interface
2. Detector setup
3. File saving

- 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



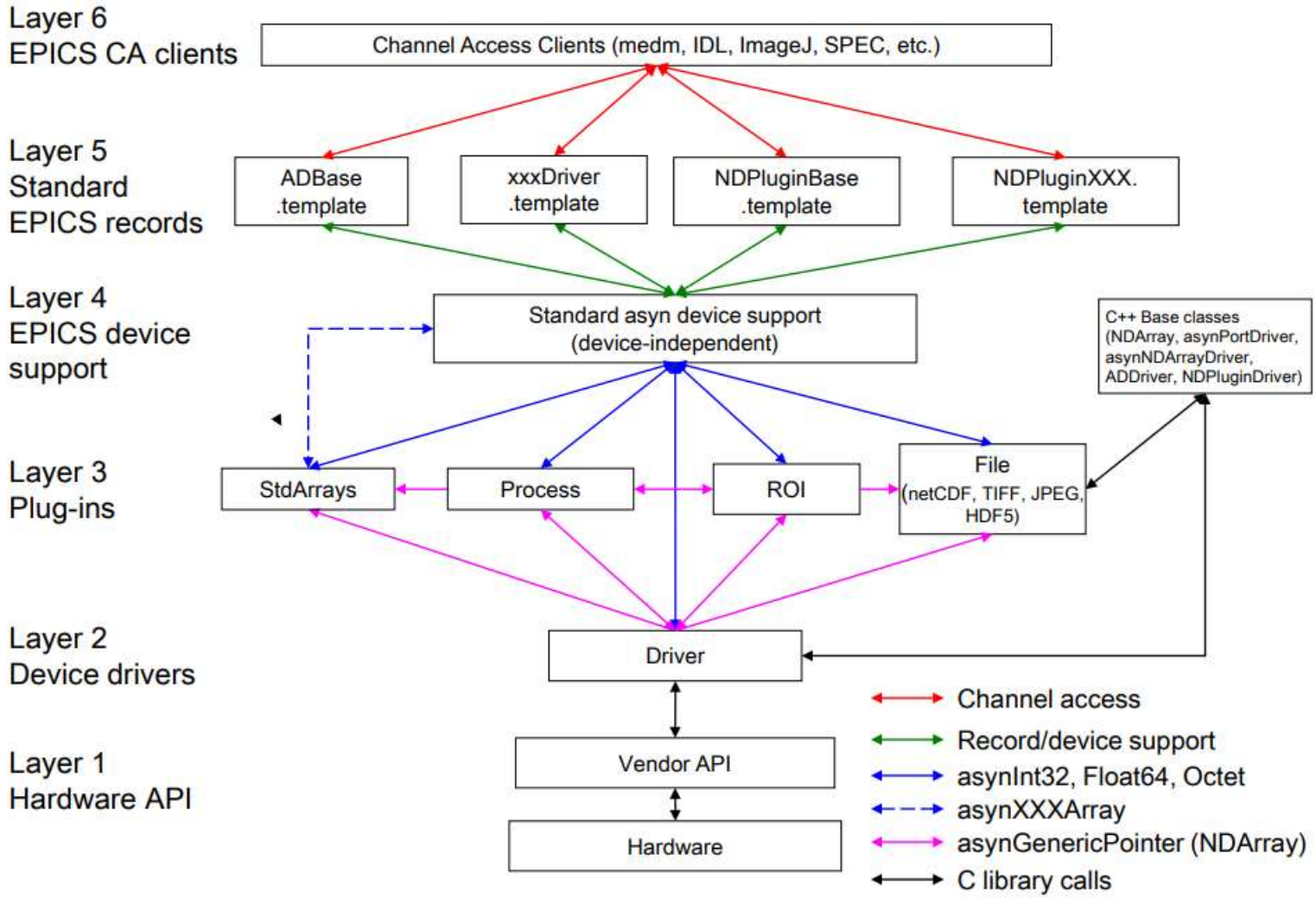
AreaDetector

- 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

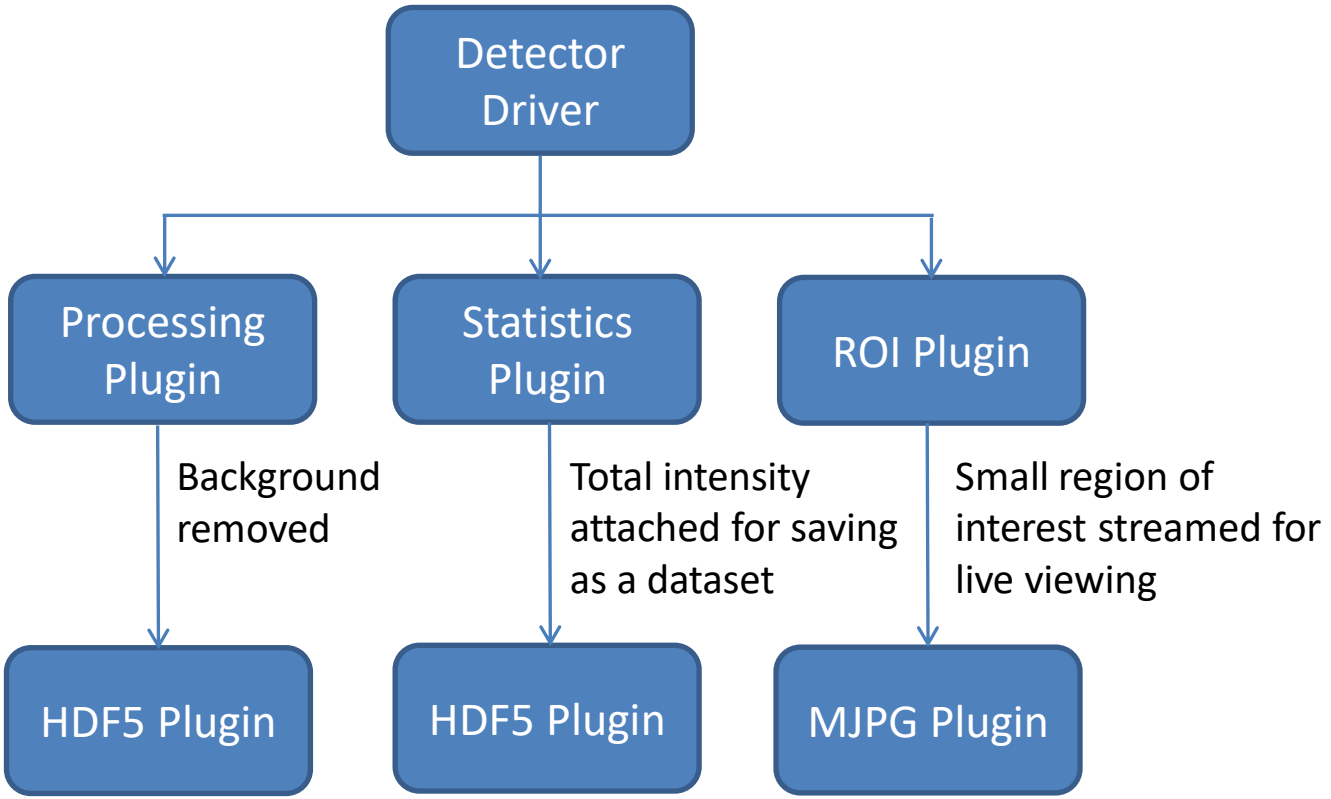
Source: Mark Rivers, Using AreaDetector



AD Plugins

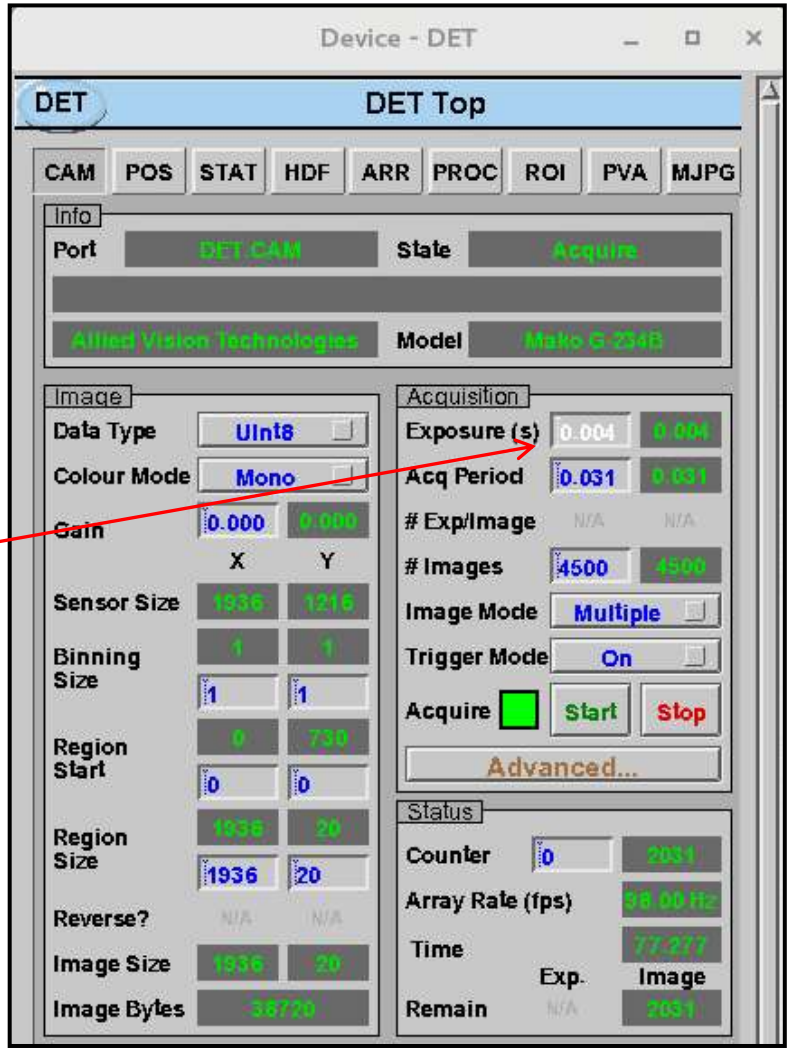
- Plugins receive a copy of each NDArray via a callback
- They can in turn be sources of NDArray callbacks
- Plugins can also attach NDAttributes to the data
- Chain can be ‘rewired’ on the fly and plugins enabled/disabled

Plugin Chain Example



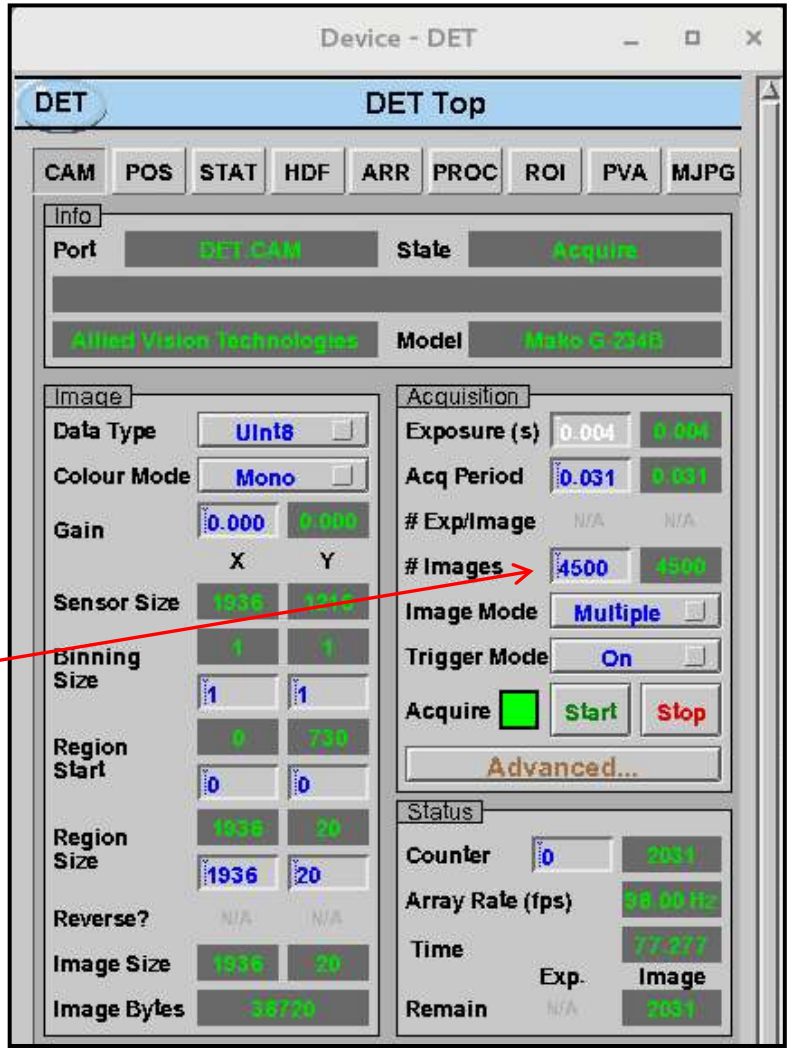
Detector Setup

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



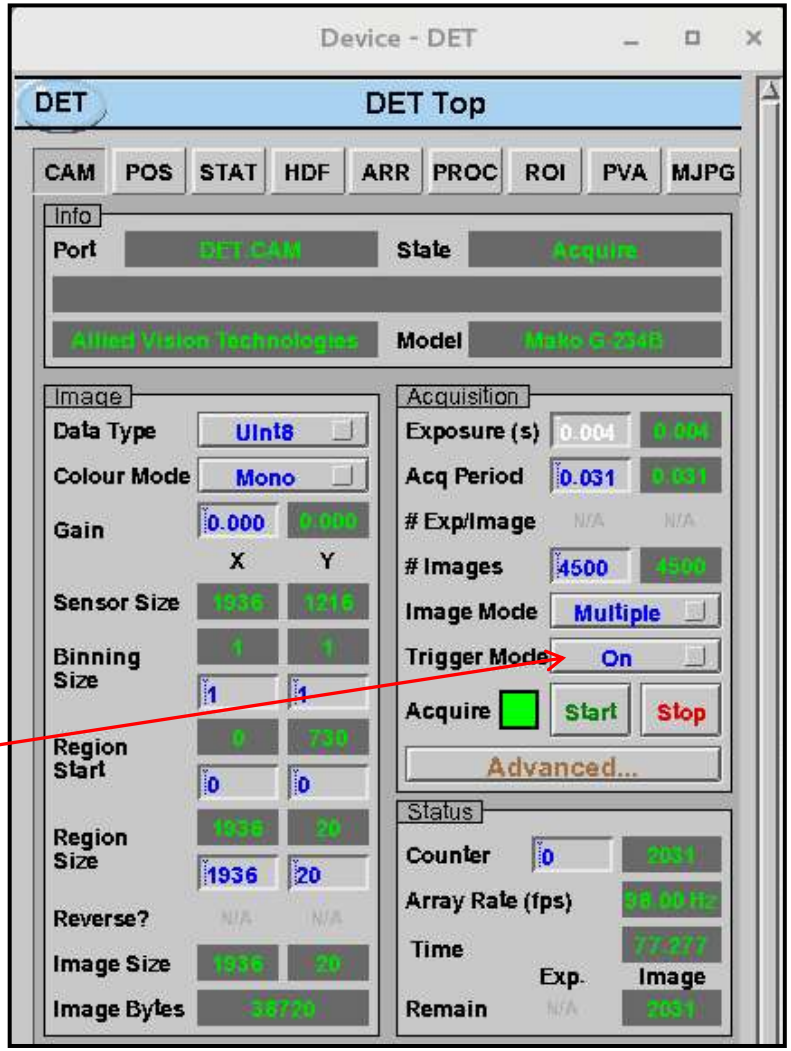
Detector Setup

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



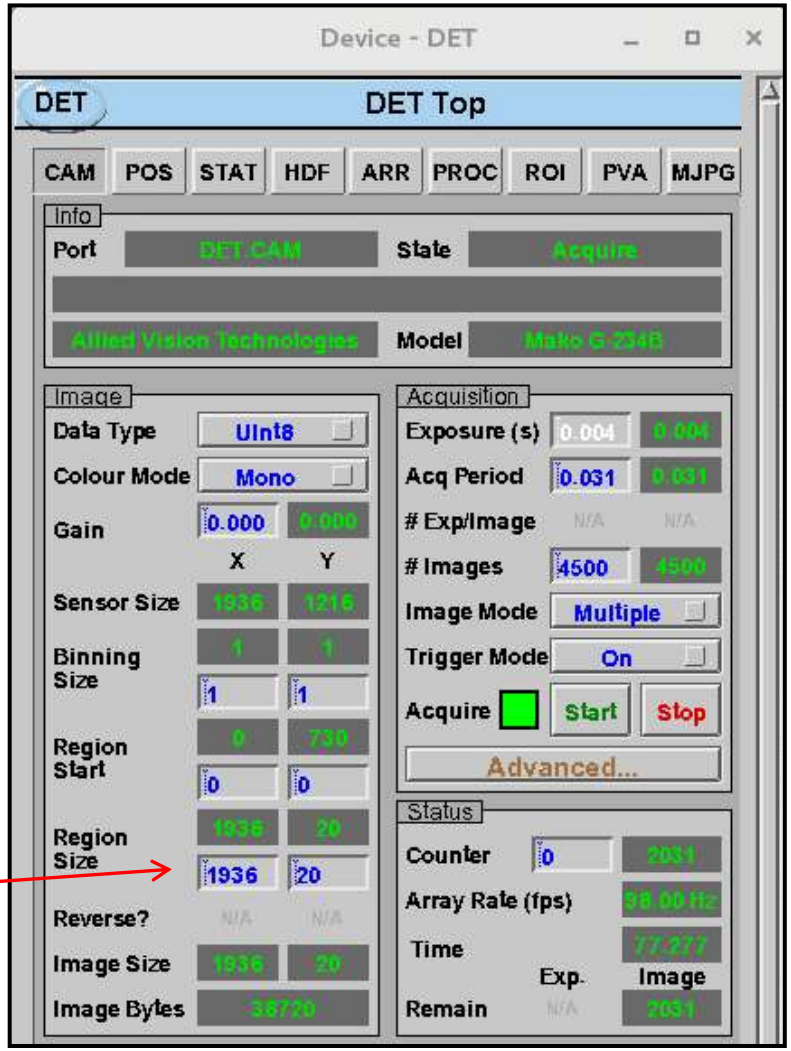
Detector Setup

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



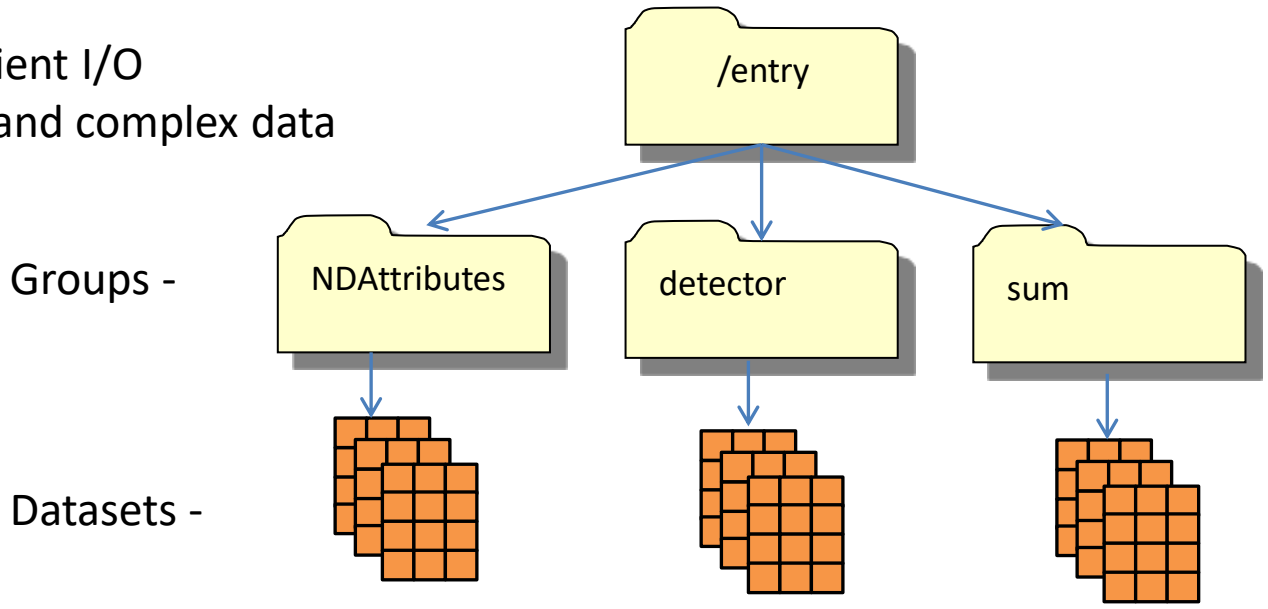
Detector Setup

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



Data Files

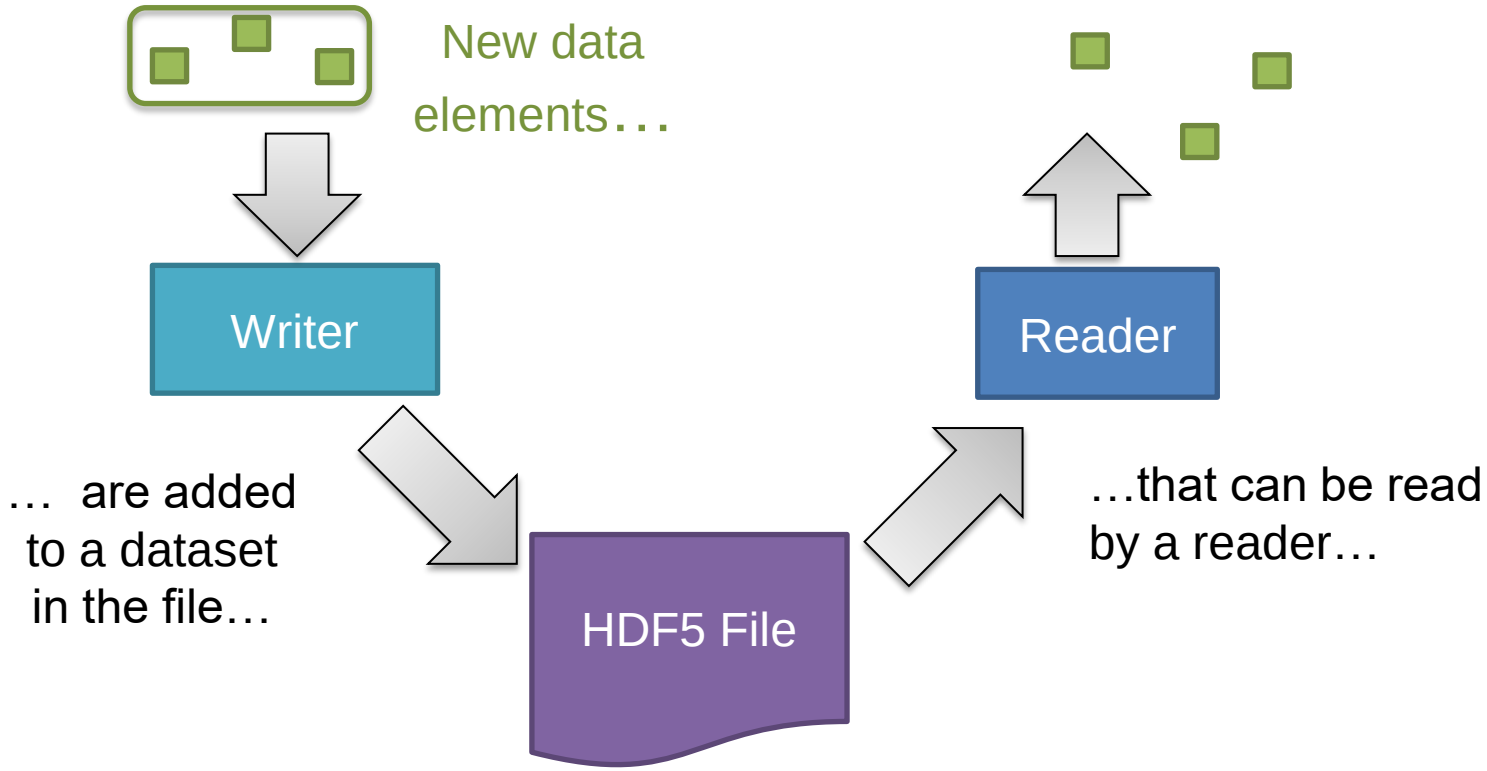
- Data is saved in the hierarchical HDF format
- Designed for:
  - Flexible, efficient I/O
  - high volume and complex data



- 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

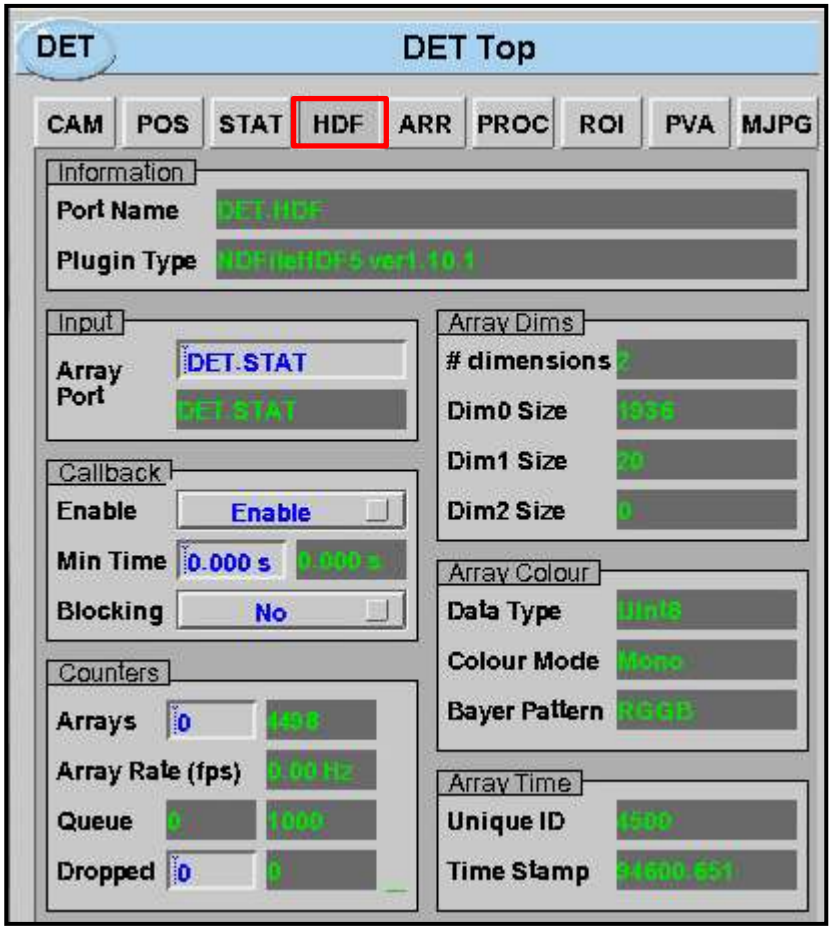
- *Flush rate* determines how often data is flushed to disk (controlled by Malcolm)
- HDF Datasets are *chunked*: divided into equally sized blocks
  - Each chunk stored contiguously in the file
  - Avoids need to read in the whole file
  - Allows datasets to be extendible
  - Enables compression
  -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

- See HDF5 Advanced Topics Tutorial

Read/Write Performance

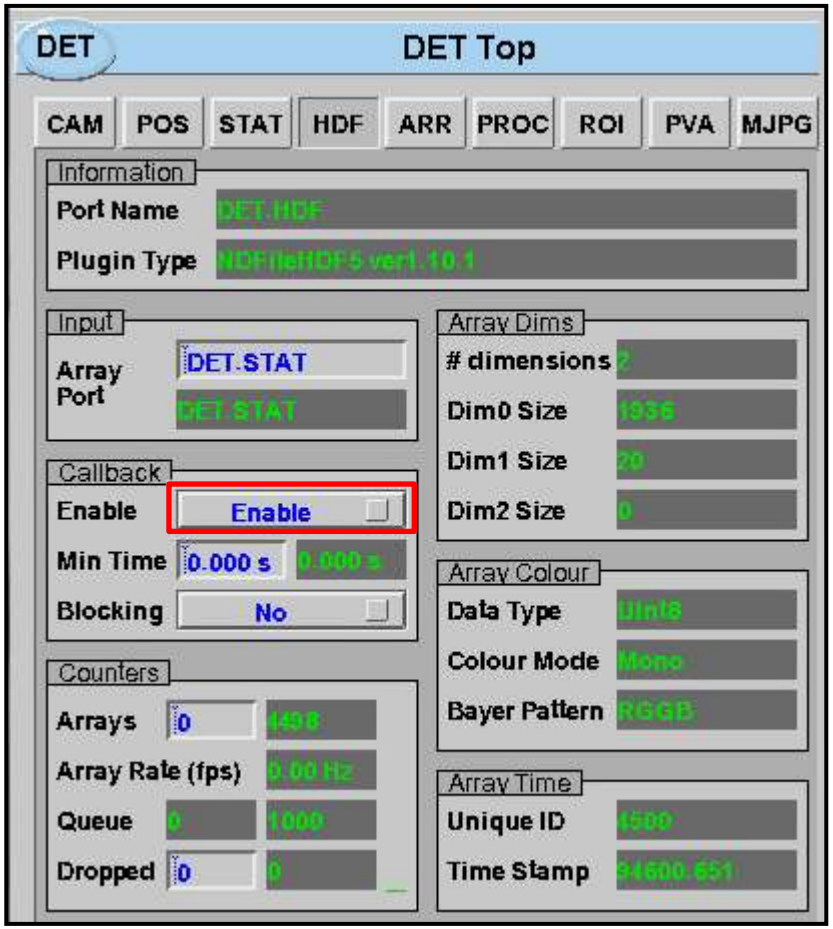
- Performance **GREATLY (up to 1000 times!!)** affected by:
  - Chunking settings
  - Compression
  - Flush rate
- Settings must be optimised for the experiment
- Consider BOTH read and write performance
- This is a complex issue!
- Never set a chunk size of 1!

File Saving



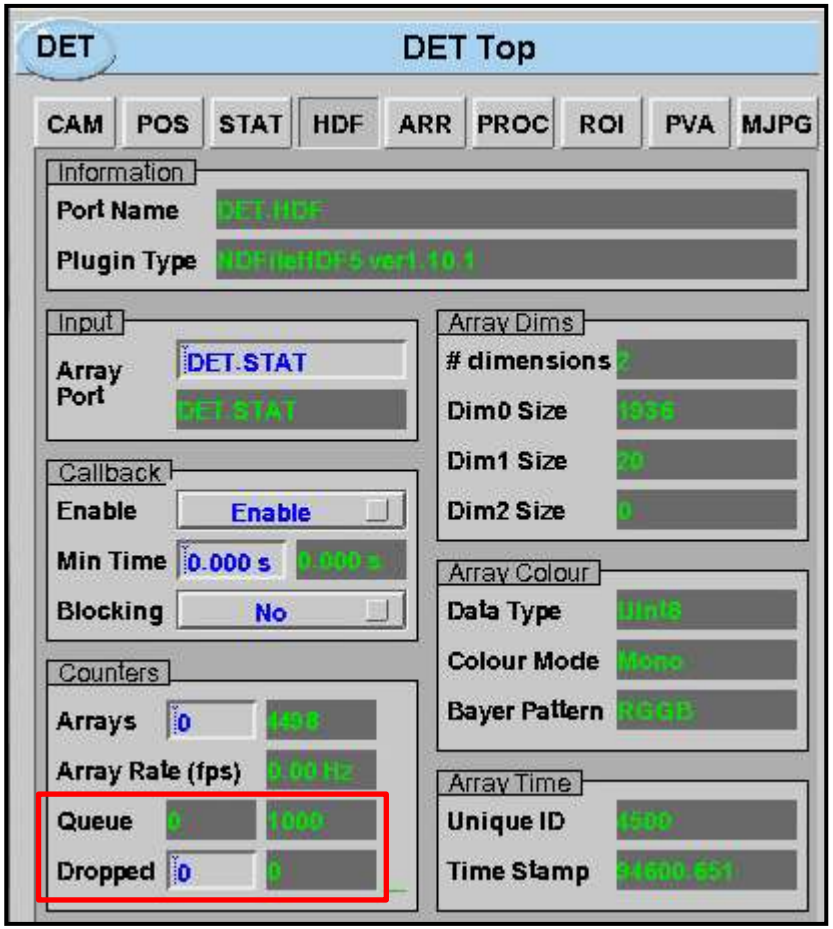
- *NDFileHDF5* plugin saves the files to disk
- Two files are generated:
  - <name>-DET.h5 – detector data
  - <name>-PANDABOX.h5 – position data
- These are saved in a folder for each scan

File Saving



- *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*

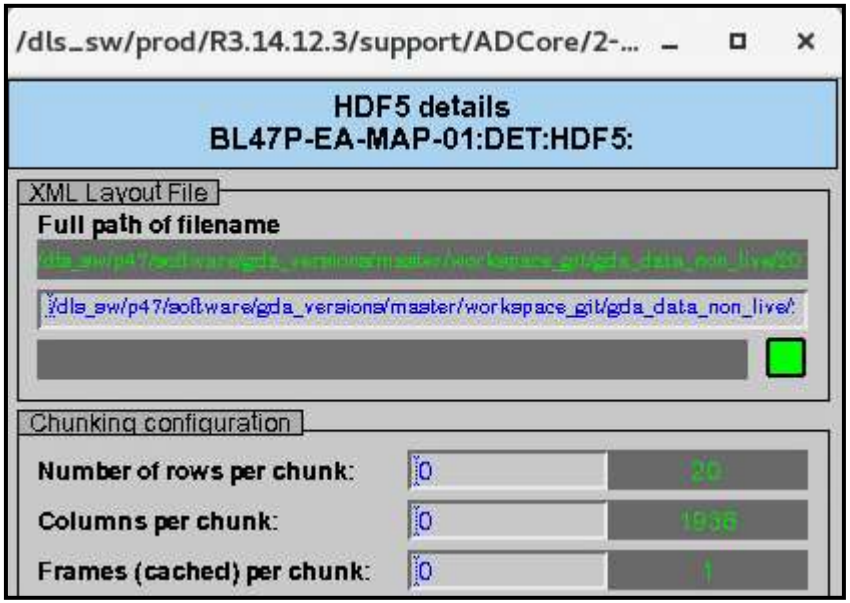
File Saving



- 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



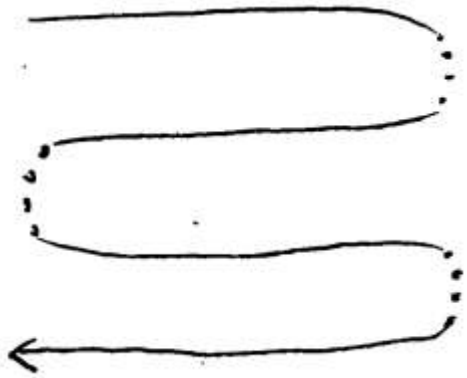
File Structure

- 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” />
```

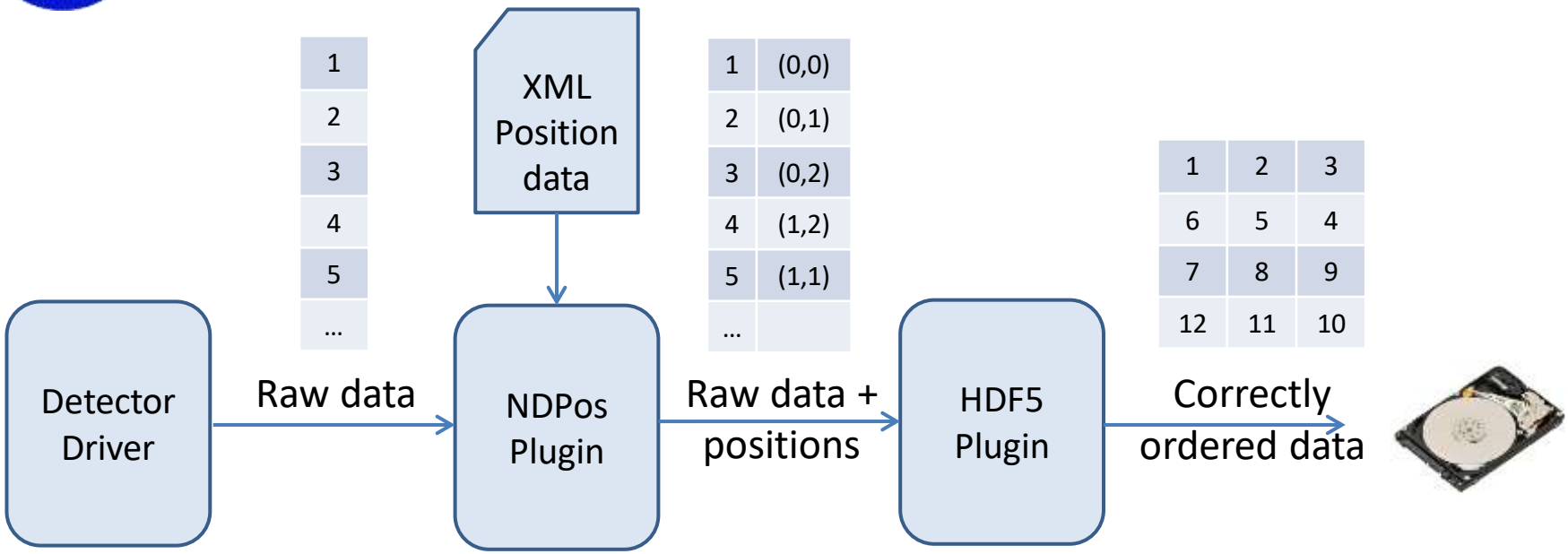
Frame Ordering

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/NDArrayUniqueId

.h5 filename

HDF5 path to dataset

```
Opened "p45-164-DET.h5" with sec2 driver.
Monitoring dataset
/entry/NDAttributes/NDArrayUniqueId...
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
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
PVDATACPP = $(SUPPORT)/pvDataCPP/5-0-4
# EPICS Base appears last
EPICS_BASE = /dls_sw/epics/R3.14.12.3/base
```

AreaDetector modules