areaDetector: What Is It, What's New and What's Next?

Mark Rivers

GeoSoilEnviroCARS, Advanced Photon Source

University of Chicago

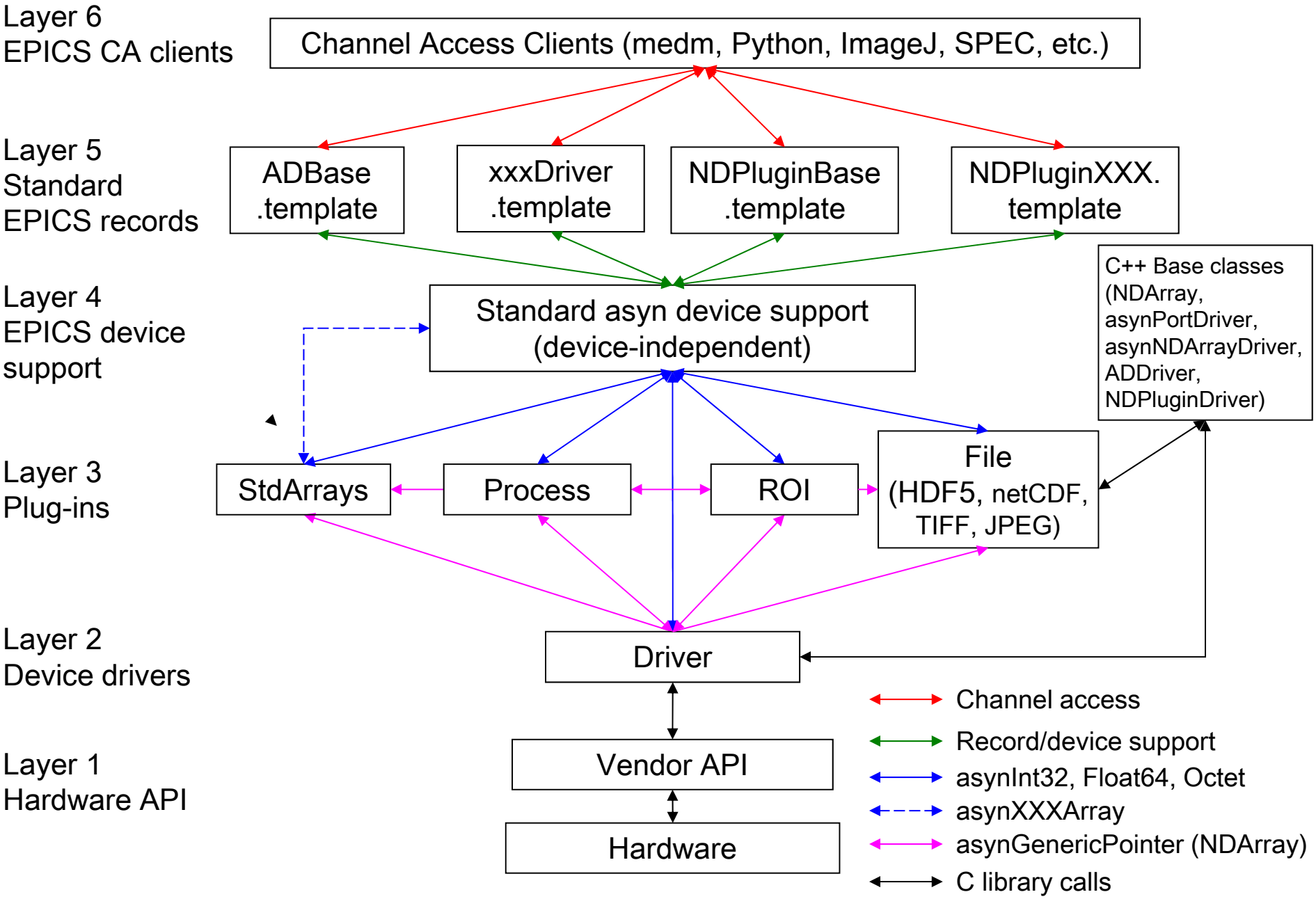
areaDetector - Goals

- Framework for 2-D detectors in EPICS
- 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
- Middle-level plug-ins to add capability like regions-of-interest calculation, file saving, etc.
  - Device independent, work with all drivers
  - Below the EPICS layer for highest performance
- EPICS-independent at lower layers.

areaDetector – Data structures

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

EPICS areaDetector Architecture



Detector Drivers

Currently 32 detector drivers covering a wide variety of detectors.

- Simulation driver
- GigE cameras (AVT, Point Grey, any GigEVision camera via aravis library)
- Point Grey USB-3.x cameras
- Dectris Pilatus and Eiger pixel array detectors
- Princeton Instruments and Photometrics detectors and spectrometers
- Andor CCD and CMOS cameras
- Perkin Elmer and Dexela flat panel detectors
- Web cameras and Axis video servers
- Many more (Bruker, Pixirad, Photonic Sciences, etc.)

Plugins

- Designed to perform real-time processing of data, running in the EPICS IOC (not over EPICS Channel Access)
- Receive NDArray data over callbacks from drivers or other plugins
- Plug-ins can execute in their own threads (non-blocking) or in callback thread (blocking)
  - If non-blocking then NDArray data is queued
  - If executing in callback thread, no queuing, but slows driver
- Allows
  - Enabling/disabling
  - Throttling rate (no more than 0.5 seconds, etc)
  - Changing data source for NDArray callbacks to another driver or plugin
- Plugins can be *sources* of NDArray 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.

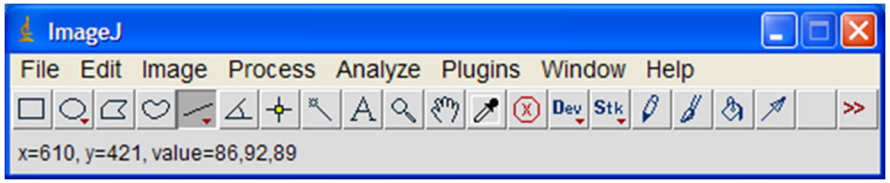
Plugins

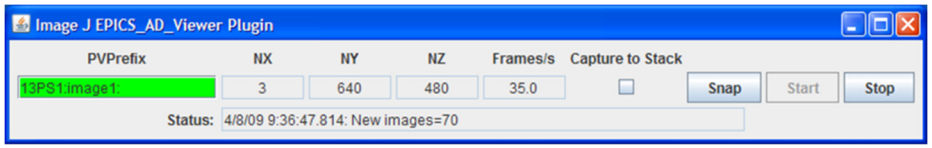
Currently 20 plugins that perform wide variety of operations.

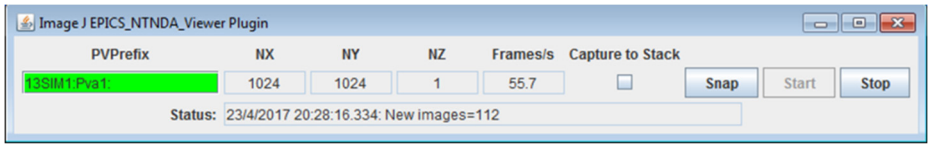
- NDPluginROI
  - Region-of-interest selection, binning, axis reversal
- NDPluginTransform
  - Geometric operations (rotate, mirror in X or Y, etc.)
- NDPluginStats
  - Min, max, mean, centroid, sigma, X/Y profiles, histogram, etc.
- NDPluginProcess
  - Image processing: background, flat field, offset, scaler, recursive filtering
- NDPluginFFT
- NDPluginFile
  - Saves files in HDF5, netCDF, TIFF, JPEG, Nexus, PNG, PDF, etc.
- NDPluginOverlay
  - Adds graphic overlays to an image
- Many more

Viewers

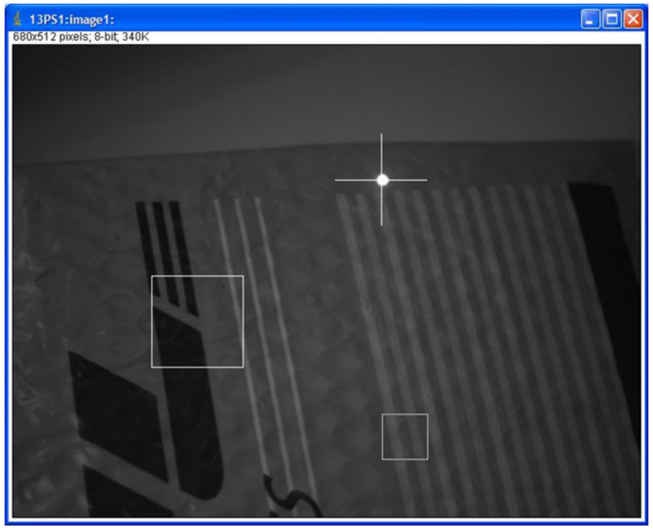
- ffmpegServer allows image display in any Web browser or ffmpegViewer high-speed Qt-based viewer for MJPEG stream
- ImageJ plugins for displaying Images from EPICS Channel Access and pvAccess (new)





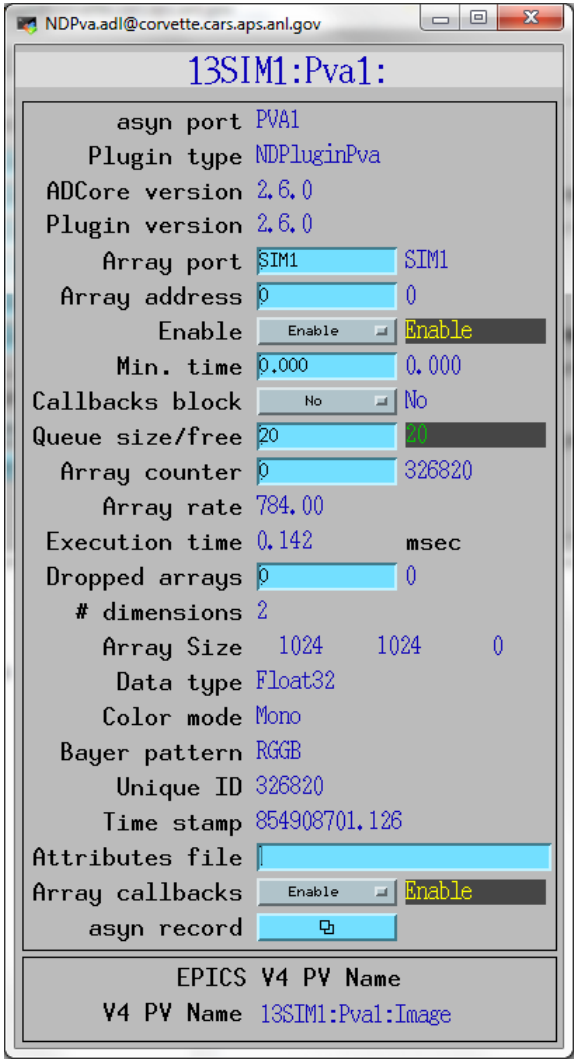


- Centroid of laser pointer calculated by statistics plugin
- Cursor overlay X, Y position linked to centroid



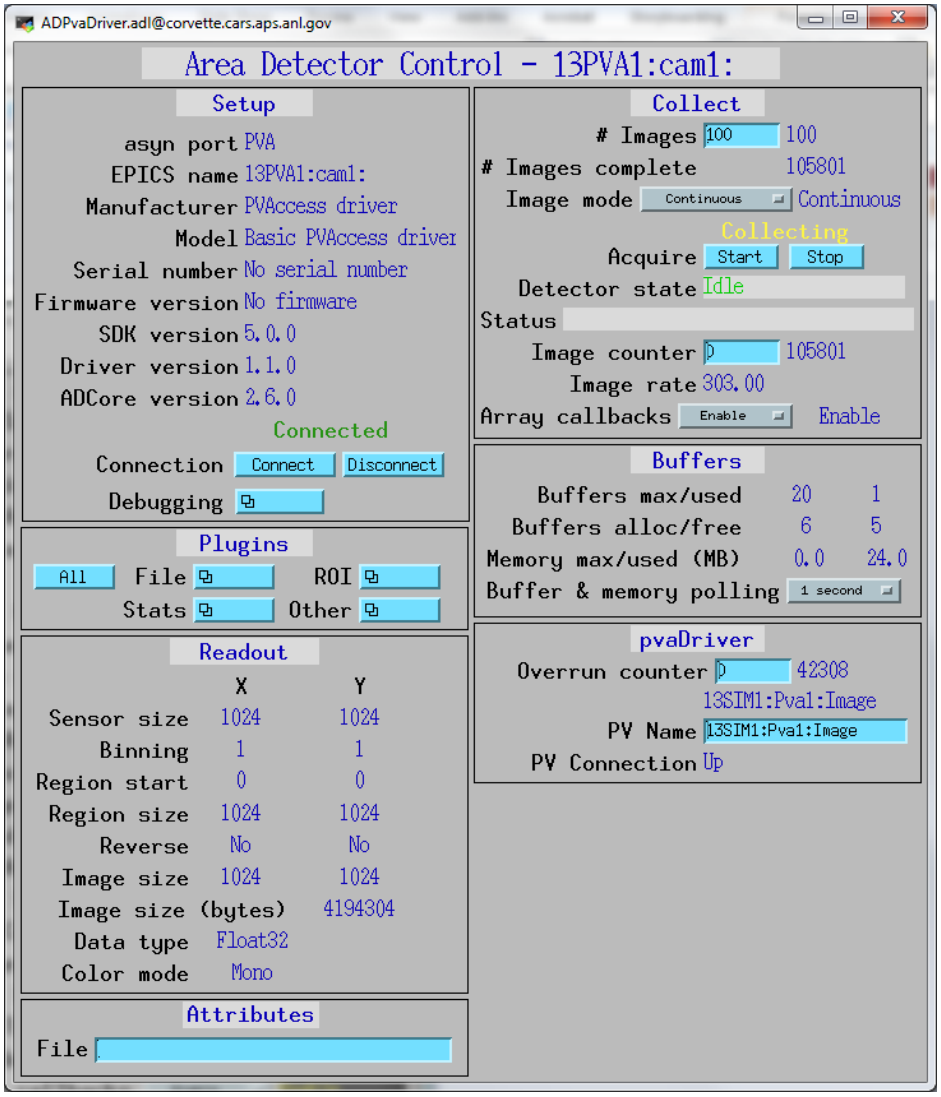
NDPluginPva (EPICS V4)

- New plugin that converts NDArrays into the EPICS v4 normative type NTNDArray
- Embedded EPICSv4 server serves the new NTNDArray structure as an EPICSv4 PV
- High performance, ~3.2GB/s shown here
- Can be received by any EPICS v4 client
  - Java, Python, C++ versions of pvAccess
  - CSS has a widget that can display NTNDArrays
  - New ImageJ plugin
  - Can include an NTNDArray receiver in another IOC
- From Bruno Martins



pvAccess Driver (EPICS V4)

- Logical inverse of NDPluginPva
- Receives NTNDArrays over the network, converts to NDArrays and calls plugins
- Can be used to run areaDetector IOC and plugins on another machine or in another process
- High performance:
  - ~1.2 GB/s shown here with interprocess communication
  - Saturating 10 Gb Ethernet links has been demonstrated
- From Bruno Martins



Multiple Threads per Plugin

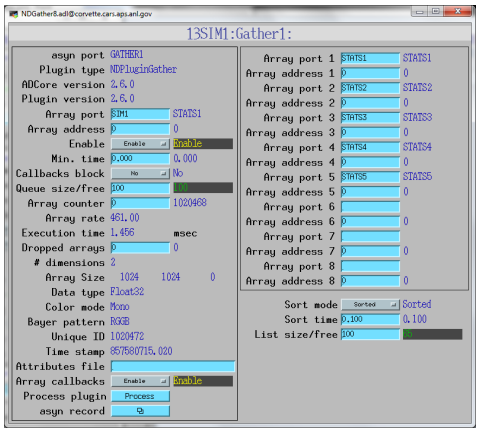
- New support for multiple threads running the processCallbacks() function in a single plugin.
- Can improve the performance of the plugin by a large factor.
- Linear scaling with up to 5 threads (the largest value tested) observed for most of the plugins that now support multiple threads.
- Maximum number of threads that can be used for the plugin is set in constructor and in IOC startup script.
- Actual number of threads to use controlled via an EPICS PV at run time, up to the maximum value.
- Optional sorting of NDArrays by uniqueId to attempt to output them in the correct order.
  - Several new parameters to control this option
- Plugins needed minor modifications to be thread-safe for multiple threads running in a single plugin object.
- Most compute-intensive plugins now support multiple threads.

NDPluginScatter

- Used to distribute (scatter) the processing of NDArrays to multiple downstream plugins
  - Allows multiple instances of a plugin to process NDArrays in parallel, utilizing multiple cores to increase throughput.
  - Utilizes modified round-robin for choosing next output plugin
- More complex than multiple threads in a single plugin, but allows the plugins running in parallel to have different configurations or even be different plugins

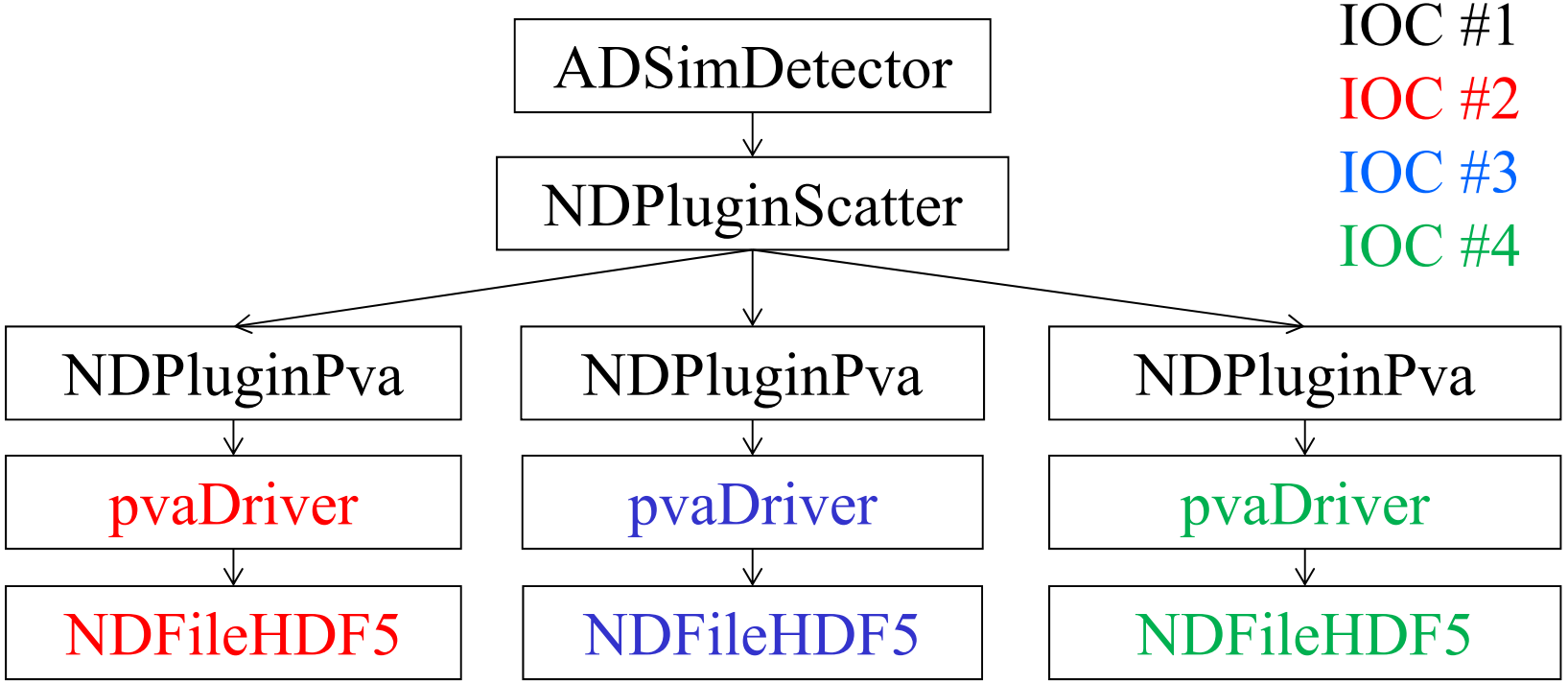
NDPluginGather

- Merges NDArrays from multiple upstream plugins into a single output stream.
- Designed to work with NDPluginScatter
- Optional sorting by uniqueId



Distributed Processing with NDPluginScatter + EPICS V4

Distribute HDF5 file writing to multiple IOCs (4096 x 3078 8-bit)



# IOCs	Files/sec	GB/sec
1	101.0	1.19
2	195.2	2.29
3	217.5	2.55

Roadmap: ADCore R5-0

- Use EPICS V4 inside drivers and plugins
- 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

Conclusions

- Architecture works well, easily extended to new detector drivers, new plugins and new clients
- Widely adopted
  - APS, SLAC, NSLS-II, CHESS, DLS, PSI, ESS, Australian Synchrotron, many others
- Base classes, asynPortDriver, asynNDArrayDriver, NDPluginDriver actually are generic, nothing “areaDetector” specific about them.
  - Used to implement other N-dimension detectors, e.g. the XIA xMAP (16 detectors x 2048 channels x 512 scan points) and quadEM (electrometers with 4 detectors x N time samples)
- Collaborative effort
  - Major contributions from Diamond, NSLS-II, SLAC, PSI, many others
- Code available on Github: https://github.com/areaDetector
- Thanks for your attention