

# Introducing HDF5 - A new storage format for your data

Chris Hall

IMBL

# Our current storage demands

- IMBL CT requires  $> 1000$  images to be collected and stored.
- IMBL allows moderate resolution for each image over a wide area  $\Rightarrow$  many pixels per image
- The SR beam and detector limits means individual projections can be made of several images.
- $\Rightarrow$  Many arrays need to be stored.
- E.g. Elvis the rhino: raw data 1.31 TB in 30 CT sets: 336,390 files.

# Future storage issue

- IMBL is designed for moderate spatial resolution imaging of large objects. We have not yet fully exploited this capability
- As well as X, Y, theta, & Z we might also add time, and even a spectral dimension to the data stream
- => The quantity and speed of data that needs to be stored is going to increase

# Data storage efficiency

- Storing each image in a single file is very inefficient.
  - Data storage times are slow
  - Directories take long times to list
  - Management of the data is slow and awkward
  - It is separated from the ‘meta-data’ (data you would normally record in your experiment log book.)

# Towards a data store standard...

- There have been several attempts to agree on a tomography data storage format e.g.
  - APS : Scientific Data Exchange
  - International: NeXus and PANData Formats
- Many rely on the concept of a data ‘container’ which keeps relevant data together in one place.

# Hierarchical Data Format 5 (HDF5)

- An HDF5 file is a *container* for storing various data
- An HDF5 file is self describing... You can figure out where and what the data is just by looking at it
- HDF5 is mature, and used in many other areas e.g. financial services
- It is composed of two primary types of objects: groups and datasets.
  - **HDF5 group**: a grouping structure containing zero or more HDF5 objects, together with supporting metadata
  - **HDF5 dataset**: a multidimensional array of data elements, together with supporting metadata

# Introduction to HDF5

- Any HDF5 group or dataset may have an associated *attribute list*.
  - An **HDF5 attribute** is a user-defined HDF5 structure that provides extra information about an HDF5 object.
- Working with groups and datasets is similar in many ways to working with directories and files in a file system. In fact an HDF5 object in an HDF5 file is often referred to by its **full path name** (also called an **absolute path name**).
  - / signifies the root group.
  - /blah signifies a member of the root group called blah.
  - /blah/blah signifies a member of the group blah (which in turn is a member of the root group blah)

# The goal for HDF5 on IMBL:

- Each sample will have all its raw data saved in a single HDF5 file. (This will eventually include the calibration images (F&D), but probably not in the first instance)
- Serial scans and other protocols will be kept in an N-dimensional data array within the HDF5 file
- Stitched and corrected projection images will be stored in a separate HDF5 file after processing
- Reconstructed data will be stored in a third HDF5 file.



# Why this protocol?

- Raw data can be collected at the highest speeds into a single file
- Kept in the /input tree of the file store, this will be archived automatically
- Processed data requires programs to read the data. Currently these work on TIFF file stacks. It is easy to unload data from the HDF5 file to TIFF stacks

# The role of AreaDetector

- All our imagers use an EPICS AreaDetector system to control, read, and store image data
- AreaDetector includes a plugin which will take the data and save it to the HDF5 file along with the instrument attributes
- Storing to HDF5 works in either Stream or Capture mode

# Example HDF5 on IMBL

The screenshot displays the HDFView application window. The main interface is divided into several panes:

- File Tree (Left):** Shows the hierarchical structure of the HDF5 file. The selected path is `entry > data > data`. Other visible nodes include `instrument`, `NDAttributes`, `detector`, and `performance`.
- Image Viewer (Top Center):** Displays a grayscale image of a detector. The title bar indicates the path `ImageView - data - /entry/data/ - /scratch/tmp/test_1.h5 - 800.0%`. The image is currently zoomed to 800.0%.
- TableView - ImageCounter (Top Right):** Shows a small table with 10 rows and 2 columns. The data is as follows:
 

	0	1
0	1	
1	2	
2	3	
3	4	
4	5	
5	6	
6	7	
7	8	
8	9	
9	10	
- TableView - data (Bottom Center):** Shows a larger table with 11 rows and 11 columns. The data is as follows:
 

	0	1	2	3	4	5	6	7	8	9	10
0	24	25	26	27	28	29	30	31	32	33	34
1	25	26	27	28	29	30	31	32	33	34	35
2	26	27	28	29	30	31	32	33	34	35	36
3	27	28	29	30	31	32	33	34	35	36	37
4	28	29	30	31	32	33	34	35	36	37	38
5	29	30	31	32	33	34	35	36	37	38	39
6	30	31	32	33	34	35	36	37	38	39	40
7	31	32	33	34	35	36	37	38	39	40	41
8	32	33	34	35	36	37	38	39	40	41	42
9	33	34	35	36	37	38	39	40	41	42	43
10	34	35	36	37	38	39	40	41	42	43	44
- TableView - NDArrayTimeStam... (Bottom Right):** Shows a table with 10 rows and 2 columns. The data is as follows:
 

	0	1
0	7.959504498582994E8	
1	7.959504498633747E8	
2	7.959504498684314E8	
3	7.95950449873486E8	
4	7.959504498785405E8	
5	7.959504498836211E8	
6	7.959504498886929E8	
7	7.959504498937742E8	
8	7.959504498988541E8	
9	7.959504499039342E8	
- Metadata (Bottom):** Displays technical details for the selected data (5644):
 

```

      data (5644)
      8-bit unsigned character, 10 x 40 x 60
      Number of attributes = 6
      NDArrayDimBinning = 1, 1
      NDArrayDimOffset = 0, 0
      NDArrayDimReverse = 0, 0
      NDArrayNumDims = 2
      NX_class = SDS
      signal = 1
      
```

## Take home message:

- Using HDF5 is essential if you want to collect a CT set with a short exposure times and/or a lot of projections
- If you are interested in switching to a more manageable data format:
- **IMBL will be using HDF5 as an option in 2018/3**



# An example HDF5 set-up

NDFileHDF5.adl

Simulation Detector - 13SIM1 (real)

## 13SIM1:HDF1:

asyn port	FileHDF1
Plugin type	NDFileHDF5 ver1.10.0
ADCore version	2.6.0
Plugin version	2.6.0
Array port	SIM1
Array address	0
Enable	<input checked="" type="checkbox"/> Enable
Min. time	0.000
Callbacks block	<input type="checkbox"/> No
Queue size/free	20 / 0
Array counter	0 / 720
Array rate	197.00
Execution time	2.199 msec
Dropped arrays	0 / 0
# dimensions	2
Array Size	1024 1024 0
Data type	UInt8
Color mode	Mono
Bayer pattern	RGGB
Unique ID	61668
Time stamp	849993152.877
Attributes file	
Array callbacks	<input type="checkbox"/> Disable <input checked="" type="checkbox"/> Enable
asyn record	<input type="checkbox"/>

File path	/home/epics/scratch/	Exists: Yes
File name	test	Create dir. depth 0 0 <input type="button" value="Help"/>
Next file #	21	Temp. suffix
Auto increment	<input checked="" type="checkbox"/> Yes	Lazy open <input type="checkbox"/> No
Filename format	%s_%3.3d.h5	Example: %s_%3.3d.h5
Last filename	/home/epics/scratch/test_020.h5	
Save file	<input checked="" type="button" value="Save"/>	Read file <input type="button" value="Read"/>
Write mode	<input type="checkbox"/> Stream <input checked="" type="checkbox"/> Stream	# Capture 1000 1000 720
Capture	<input checked="" type="button" value="Start"/> <input type="button" value="Stop"/>	Delete driver file <input type="checkbox"/> No
Write status	Write OK	
Write message		

Compression	<input type="checkbox"/> zlib <input checked="" type="checkbox"/> None	SWMR Support
# data bits	8	SWMR supported
Data bits offset	0	SWMR mode
SZip # pixels	16	SWMR active
Zlib level	6	SWMR callbacks
Store performance	<input checked="" type="checkbox"/> Yes	More <input type="button" value="..."/>
Store attributes	<input checked="" type="checkbox"/> Yes	
Run time	0.012	
I/O speed	664.3	
Default layout selected		Exists: Yes
XML File name		

Rows per chunk	0	1024
Columns per chunk	0	1024
Frames cached per chunk	0	1
Boundary alignment	0	0
Boundary threshold	65536	65536
Flush on N'th frame	0	1
Fill value	0.0	0.0