



The HDF Group



# HDF5 Filters

## Overview



# Outline

- Overview of HDF5 filters
- Dynamically loaded filters



# OVERVIEW OF HDF5 FILTERS



# What is an HDF5 filter?

- Data transformation performed by the HDF5 library during I/O operations
  - HDF5 filters (or *built-in* filters)
    - Supported by The HDF Group
    - Come with the HDF5 library source code
  - User-defined filters
    - Filters written by HDF5 users and/or available with some applications (h5py, PyTables)
    - May be or may not be registered with The HDF Group



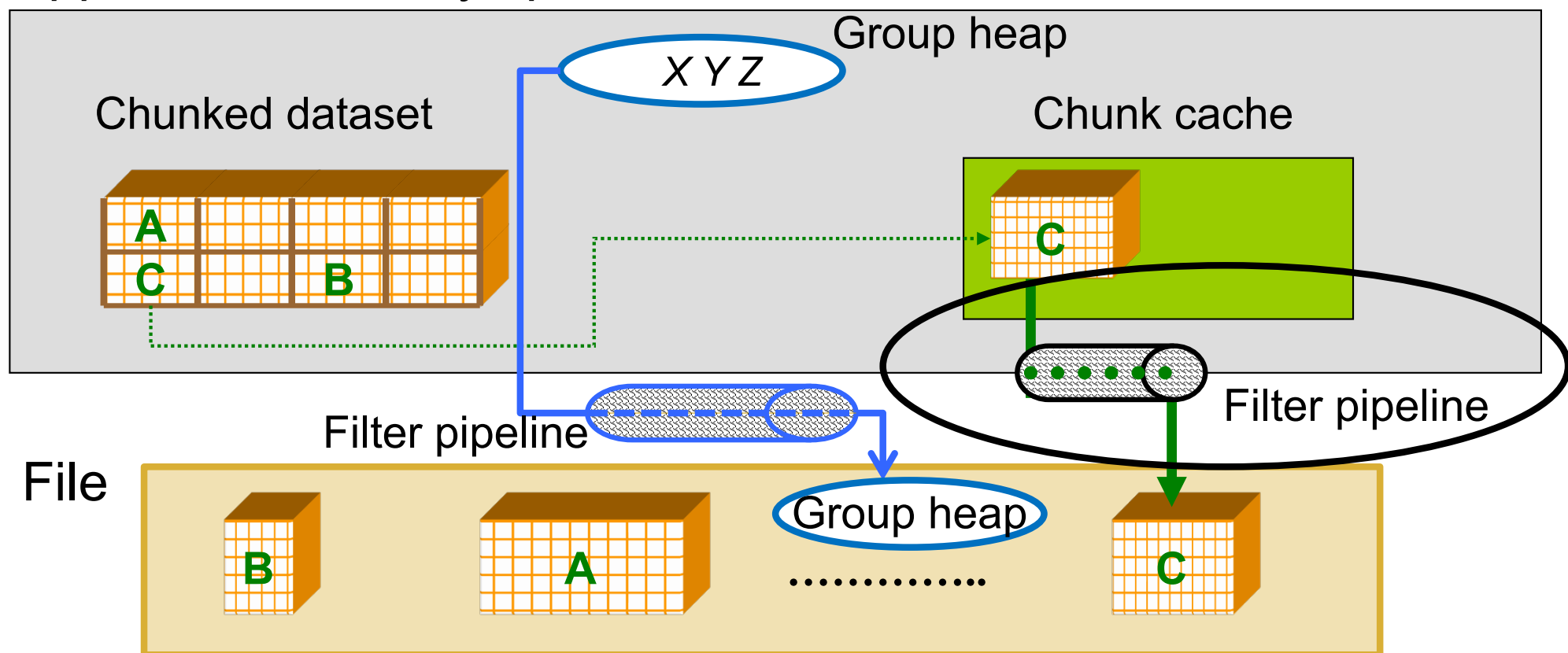
# HDF5 filters

- Filters are arranged in a pipeline so the output of one filter becomes the input of the next filter
- The filter pipeline can be only applied to
  - *Chunked* dataset
    - HDF5 library passes each chunk through the filter pipeline on the way to or from disk
  - Group
    - Link names are stored in a local heap, which may be compressed with a filter pipeline
- Filter pipeline is permanent for dataset or a group



# Filter pipeline

## Application memory space



Filters are applied in a user-specified order when the HDF5 library performs I/O operations on a chunk or on a group heap



# Filter pipeline programming model

- Operations on the HDF5 filter pipeline

<http://www.hdfgroup.org/HDF5/doc1.6/Filters.html>

- Defining a pipeline
  - Use a sequence of the `H5Pset_filter` calls or predefined API, e.g., `H5Pset_deflate`, on a *dataset* or *group creation* property to create a pipeline
  - On write, the filters are applied in the order they were specified
  - On read, the filters are applied in the reverse order they were specified (last one in the pipeline is applied first)
  - *It is the user's responsibility to create a meaningful pipeline*



# Filter pipeline programming model

- Operations on the HDF5 filter pipeline
  - Query
    - Number of filters in a pipeline
      - H5Pget\_nfilters
    - Information about a filter using filter identifier
      - H5Pget\_filter\_by\_id
    - Check if a filter is available in the library
      - H5Zfilter\_avail
  - Modify
    - Change properties of existing filter
      - H5Pmodify\_filter
    - Remove filter from pipeline
      - H5Premove\_filter





# Filter pipeline programming model

- *Filter pipeline is permanent for dataset or a group*
  - Filters are part of an HDF5 object (group or dataset) creation property
  - The object's filter pipeline cannot be modified after the object has been created



# Applying filters to a dataset

```
dcpl_id    = H5Pcreate(H5P_DATASET_CREATE);  
cdims[0]   = 100;  
cdims[1]   = 100;  
H5Pset_chunk(dcpl_id, 2, cdims);  
H5Pset_shuffle(dcpl);  
H5Pset_deflate(dcpl_id, 9);  
dset_id = H5Dcreate (... , dcpl_id);  
H5Pclose(dcpl_id);
```



# Applying filters to a group

```
gcpl_id    = H5Pcreate(H5P_GROUP_CREATE);  
H5Pset_deflate(dcp1_id, 9);  
group_id = H5Gcreate (... , gcpl_id, ...);  
H5Pclose(gcpl_id);
```



# Example h5ex\_d\_gzip.f90

- Review h5ex\_d\_gzip.f90

```
h5dump -pH h5ex_d_gzip.h5
HDF5 "h5ex_d_gzip.h5" {
  GROUP "/" {
    DATASET "DS1" {
      DATATYPE H5T_STD_I32LE
      DATASPACE SIMPLE { ( 64, 32 ) / ( 64, 32 ) }
      STORAGE_LAYOUT {
        CHUNKED ( 8, 4 )
        SIZE 5337 (1.535:1 COMPRESSION)
      }
      FILTERS {
        COMPRESSION DEFLATE { LEVEL 9 }
      }
      FILLVALUE {
        FILL_TIME H5D_FILL_TIME_IFSET
        VALUE H5D_FILL_VALUE_DEFAULT
      }
      ALLOCATION_TIME {
        H5D_ALLOC_TIME_INCR
      }
    }
  }
}
```



# HDF5 DEAFULT FILTERS



# External HDF5 Filters

- External HDF5 filters rely on the third-party libraries installed on the system
  - GZIP
    - By default HDF5 configure uses ZLIB installed on the system
    - Configure will proceed if ZLIB is not found on the system
  - SZIP (added by NASA request)
    - Optional; have to be configured in using `–with-szlib=/path....`
    - Configure will proceed if SZIP is not found
    - Comes with a license  
[http://www.hdfgroup.org/doc\\_resource/SZIP/Commercial\\_szip.html](http://www.hdfgroup.org/doc_resource/SZIP/Commercial_szip.html)
    - Decoder is free; for encoder see the license terms



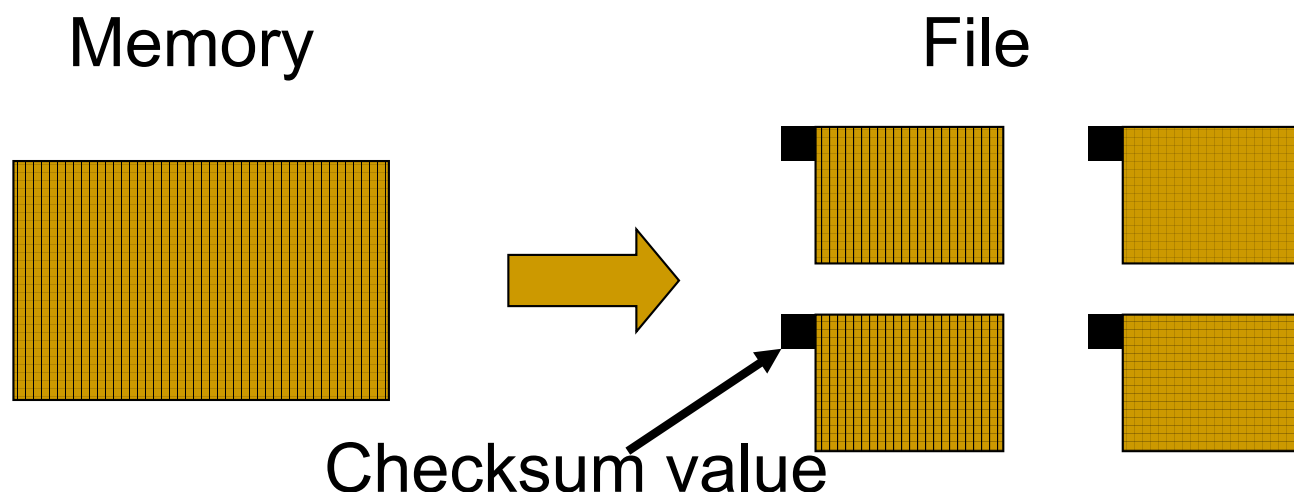
# Internal HDF5 Filters

- Internal filters are implemented by The HDF Group and come with the library
- HDF5 internal filters can be configured out using `–disable-filters=“filter1, filter2, ..”`
  - FLETCHER32
  - SHUFFLE
  - SCALEOFFSET
  - NBIT
  - CHECKSUM



# Checksum filter

- Predefined HDF5 filter (H5Pset\_fletcher32)
- Why:
  - Error detection for raw data
- What:
  - Implements Fletcher32 checksum algorithm







# Enabling Raw Data Checksum

- Review in EX-5 `h5ex_d_checksum.f90`



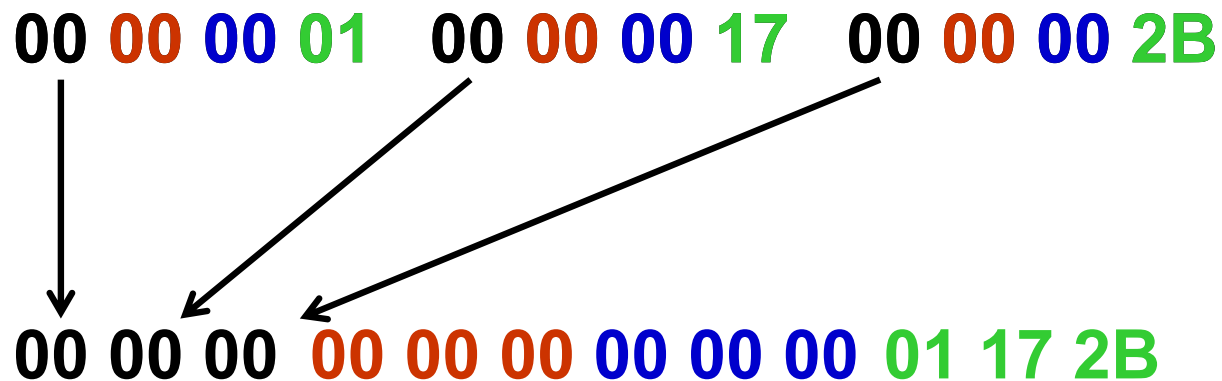
# Applying HDF5 filters

- Chunked storage is required
- Types of filters
  - Algebraic Data transformation
  - Data shuffling
  - Checksum
  - Data compression
    - Scale + offset
    - N-bit
    - GZIP (deflate)
    - SZIP
  - Compression methods supported by HDF5 User Community



# Shuffling filter

- Predefined HDF5 filter (H5Pset\_shuffle)
- Why:
  - Better compression of unused bytes
- What:
  - Changes byte order in a stream of data





# Effect of data shuffling

- H5Pset\_shuffle followed by H5Pset\_deflate
- Write 4-byte integer dataset 256x256x1024 (256MB)
- Using chunks of 256x16x1024 (16MB)
- Values: random integers between 0 and 255

	File size	Total time	Write Time
No Shuffle	102.9MB	671.049	629.45
Shuffle	67.34MB	83.353	78.268



# Enabling Raw Data Checksum

- Review in EX-5 `h5ex_d_checksum.f90`
- Add GZIP compression to pipeline



# N-bit compression filter

- Predefined HDF5 filter (`H5Pset_nbit`)
- Why:
  - Compact storage for user-defined datatypes
- What:
  - When data stored on disk, padding bits chopped off and only significant bits stored
  - Supports most datatypes
  - Works with compound datatypes



# N-bit compression example

- In memory, one value of N-Bit datatype is stored like this:

```
| byte 3 | byte 2 | byte 1 | byte 0 |  
| ???????? | ???SPPP | P P P P P P P P | P P P P ???? |
```

S-sign bit    P-significant bit    ?-padding bit

- After passing through the N-Bit filter, all padding bits are chopped off, and the bits are stored on disk like this:

```
|    1st value    |    2nd value    |  
| S P P P P P P P | S P P P P P P P | ...
```

- Opposite (decompress) when going from disk to memory



# Enabling Nbit filter

- Review in EX-5 `h5ex_d_nbit.f90`





## “Scale+offset” filter

- Predefined HDF5 filter  
(H5Pset\_scaleoffset)
- Why:
  - Use less storage when less precision needed
- What:
  - Performs scale/offset operation on each value
  - Truncates result to fewer bits before storing
  - Currently supports integers and floats



# Example with floating-point type

- Data: {104.561, 99.459, 100.545, 105.644}
- Choose scaling factor: decimal precision to keep  
E.g. **scale factor D = 2**
  1. Find minimum value (offset): 99.459
  2. Subtract minimum value from each element  
Result: {5.102, 0, 1.086, 6.185}
  3. Scale data by multiplying  $10^D = 100$   
Result: {510.2, 0, 108.6, 618.5}
  4. Round the data to integer  
Result: {510, 0, 109, 619}
  5. Pack and store using min number of bits



# Checking available HDF5 Filters

- Use API (`H5Zfilter_avail`)
- Check `libhdf5.settings` file

Features:

Parallel HDF5: no

.....

I/O filters (external): deflate(zlib),szip(encoder)

I/O filters (internal): shuffle,fletcher32,nbit,scaleoffset

.....



# THIRD PARTY HDF5 FILTERS



## Third-party HDF5 filters

- Compression methods supported by HDF5 user community

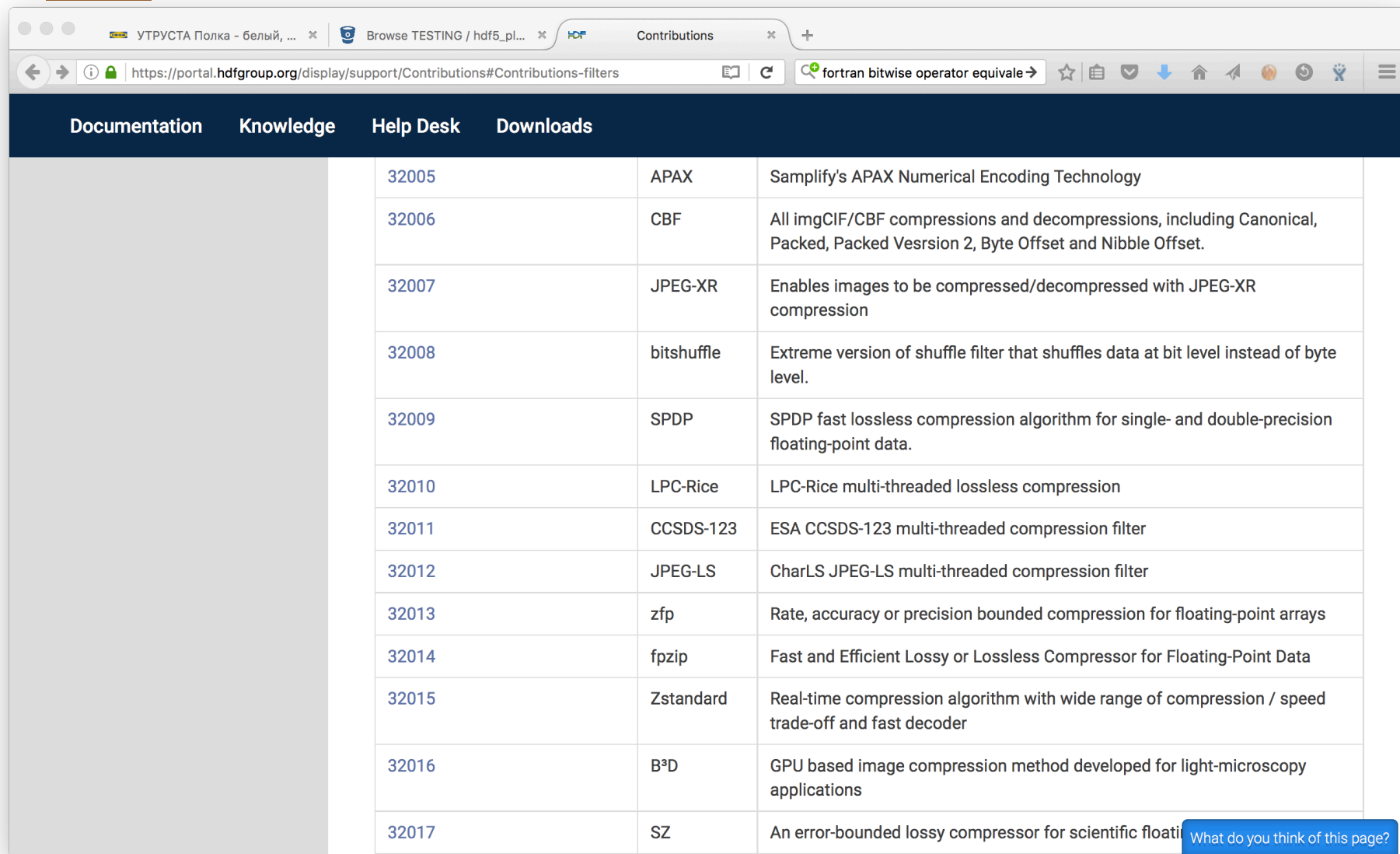
<http://www.hdfgroup.org/services/contributions>

- LZO, BZIP2, BLOSC (PyTables)
- LZF (h5py)
- MAFISC
  - The Website has a patch for external module loader
- Registration process
  - Helps with filter's provenance



# Registered third-part filters

<https://portal.hdfgroup.org/display/support/Contributions#Contributions-filters>



Documentation	Knowledge	Help Desk	Downloads
	<a href="#">32005</a>	APAX	Samplify's APAX Numerical Encoding Technology
	<a href="#">32006</a>	CBF	All imgCIF/CBF compressions and decompressions, including Canonical, Packed, Packed Verssion 2, Byte Offset and Nibble Offset.
	<a href="#">32007</a>	JPEG-XR	Enables images to be compressed/decompressed with JPEG-XR compression
	<a href="#">32008</a>	bitshuffle	Extreme version of shuffle filter that shuffles data at bit level instead of byte level.
	<a href="#">32009</a>	SPDP	SPDP fast lossless compression algorithm for single- and double-precision floating-point data.
	<a href="#">32010</a>	LPC-Rice	LPC-Rice multi-threaded lossless compression
	<a href="#">32011</a>	CCSDS-123	ESA CCSDS-123 multi-threaded compression filter
	<a href="#">32012</a>	JPEG-LS	CharLS JPEG-LS multi-threaded compression filter
	<a href="#">32013</a>	zfp	Rate, accuracy or precision bounded compression for floating-point arrays
	<a href="#">32014</a>	fpzip	Fast and Efficient Lossy or Lossless Compressor for Floating-Point Data
	<a href="#">32015</a>	Zstandard	Real-time compression algorithm with wide range of compression / speed trade-off and fast decoder
	<a href="#">32016</a>	B <sup>3</sup> D	GPU based image compression method developed for light-microscopy applications
	<a href="#">32017</a>	SZ	An error-bounded lossy compressor for scientific floati

What do you think of this page?



# HOW TO ADD YOUR OWN FILTER



## How to proceed?

- Implement a filter (See `H5Zregister` in RM)
  - See `H5Zdeflate.c` in the HDF5 src directory for ideas
- Application will need to
  - Register filter with the HDF5 library using `H5Zregister`
  - Add filter to pipeline using `H5Pset_filter`
  - Follow the HDF5 programming model as usual
- Or
  - Library needs to be modified with the new filter





## Example: h5dump output on BZIP2 data

```
HDF5 "h5ex_d_bzip2.h5" {  
  GROUP "/" {  
    DATASET "DS-bzip2" {  
      ...  
    }  
    FILTERS {  
      UNKNOWN_FILTER {  
        FILTER_ID 307  
        COMMENT bzip2  
        PARAMS { 9 }  
      }  
    }  
    .....  
  }  
  DATA {h5dump error: unable to print data  
}
```



## Problem with using custom filter

- “Off the shelf” HDF5 tools do not work with the third-party filters
  - h5dump, MATLAB and IDL, etc.
- Solution
  - Modify HDF5 source with your code and distribute it
    - And what will happen if a user wants filters from the different distributions???? Oh.... No....
  - Use a 1.8.11 and later ☺



# **DYNAMICALLY LOADED FILTERS IN HDF5**



# Sponsors and Stakeholders

---

- DESY
- Synchrotron Community
- PyTables, H5Py and others



# Requirements

- There are no changes to the HDF5 file format.
- There are no changes to the HDF5 interface and the programming model in order to use the feature
- The HDF5 third-party filters are available as shared libraries or DLLs on the user's system.
- The minimum content of the shared library satisfies the requirements as shown on the next slide.



# Requirements

- There are predefined default locations where the HDF5 library searches the shared libraries or DLLs with the HDF5 filter functions.
- The default location may be overwritten by an environment variable.
- Once a filter plugin library is loaded, it stays loaded until the HDF5 library is closed.



# Programming Model

- As before but doesn't require H5Zregister
- On write

```
dcpl = H5Pcreate (H5P_DATASET_CREATE);
status = H5Pset_filter (dcpl, (H5Z_filter_t)307,
                        H5Z_FLAG_MANDATORY, (size_t)6, cd_values);
dset = H5Dcreate (file, DATASET, H5T_STD_I32LE, space,
                 H5P_DEFAULT, dcpl,...);
status = H5Dwrite (dset, H5T_NATIVE_INT, H5S_ALL, H5S_ALL,
                  H5P_DEFAULT, wdata[0]);
```



# Programming Model

- On read

```
file = H5Fopen (FILE, H5F_ACC_RDONLY, H5P_DEFAULT);  
dset = H5Dopen (file, DATASET, H5P_DEFAULT);  
      H5Dread (dset, H5T_NATIVE_INT, H5S_ALL, H5S_ALL,  
              H5P_DEFAULT, rdata[0]);
```





# Plugin Library Requirements

- Built as shared library (\*.so, DLL)
- The plugin source includes:
  - H5PLextern.h
  - Definition of the filter structure
  - The filter function
  - Two “helper” functions :
    - Find plugin type  
`H5PL_type_t H5PLget_plugin_type(void)`
    - Return plugin structure  
`void *H5PLget_plugin_info(void)`
- It may also include other functions, for example, the source of the compression library.



## Plugin Example

[https://bitbucket.hdfgroup.org/projects/TEST/repos/hdf5\\_plugins/browse](https://bitbucket.hdfgroup.org/projects/TEST/repos/hdf5_plugins/browse)

### BZIP2

- Bzip2 filter implemented in PyTables
- Built with configure or CMake
- Used as acceptance test for the feature
- More plugins can be added in the future



## Steps to create an HDF5 filter plugin

- When compiling, point to the HDF5 include files.
- Use the appropriate linking flags.
- Link with any required external libraries.
- Adopt the example above we provide



# **DYNAMICALLY LOADED FILTERS DESIGN**

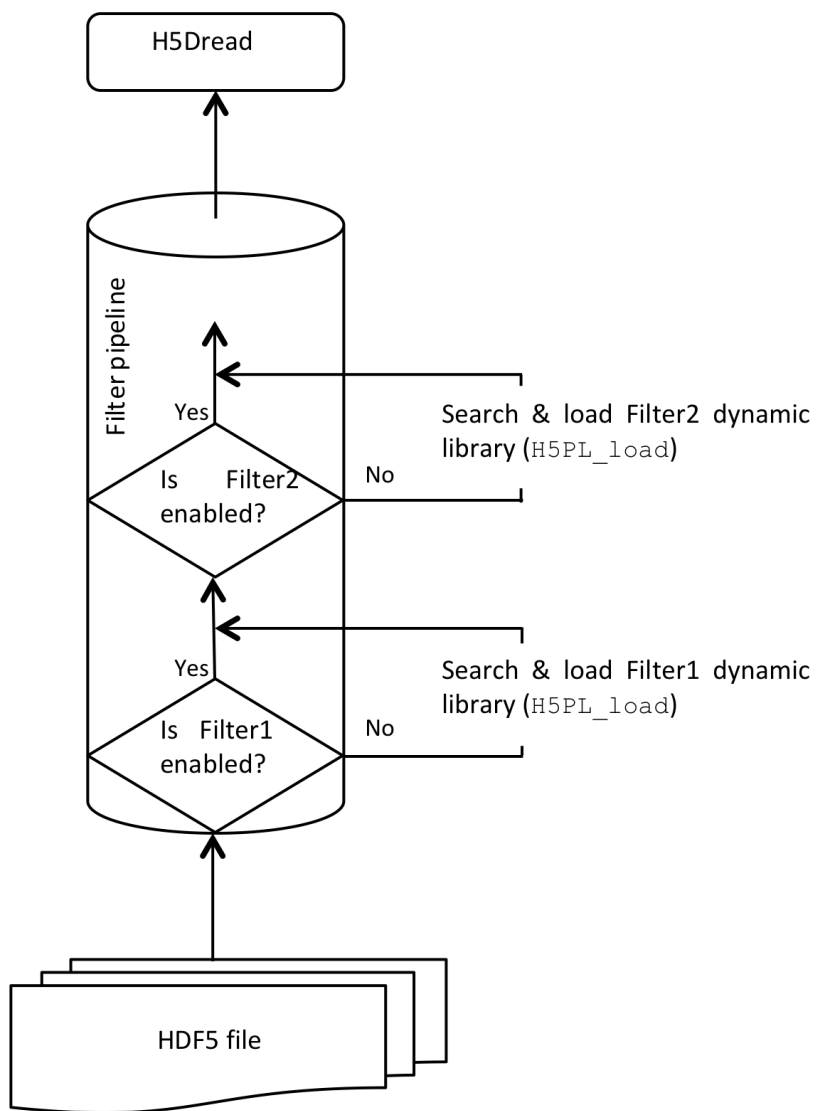


## On write

- H5Pset\_filter triggers search, loading and registering of the filter.
- Where to search for write and read?
  - Linux/UNIX
    - “/usr/local/hdf5/lib/plugin”
  - Windows
    - "%ALLUSERSPROFILE%/hdf5/lib/plugin"
  - Default path can be overwritten with the HDF5\_PLUGIN\_PATH environment variable



# On read





# Changes to the source code

- H5Pocpl.c (H5Pset\_filter), H5Z.c (register filter, etc.), some H5E\* files.
- New
  - H5PL.c, H5PLprivate.h
    - Manage external plugins
      - Search
      - Load
      - Register
      - Unload
  - H5PLextern.h
    - Included by plugins
- plugin.c
  - Test code; uses three “dummy” plugins



**DO NOT DO THIS**





## Word of caution

- Do not unregister **any** custom filter before
  - Closing all objects that use the filter
  - Flushing the objects to the file
- We will need to fix H5Zunregister to be more conscious to avoid failures:
  - Fail when objects are still open
  - Flush MD and chunk caches if objects are closed



# WHAT ABOUT TOOLS?



# HDF5 Tools

- No need to modify “reading tools” such as h5dump, h5ls, and h5copy, and HDFView

- h5repack

h5repack -f UD={ID:k; N:m; CD\_VAL:[n<sub>1</sub>,...,n<sub>m</sub>]}.....

- BZIP2 example

h5repack -f UD={ID:307; N:1; CD\_VAL:[9]} file1.h5 file2.h5



The HDF Group



# Thank You!

## Questions?