

# **Introduction to HDF5 for HDF4 users**

Robert E. McGrath

NCSA

December 4, 2002

# Overview

- Assumes that you know HDF4 concepts and have used HDF4
- Would like to start using HDF5
- **Goal:** Provide some principles and examples that will help understand HDF5
- A paper gives more details and a collection of “How Do I...?” suggestions.

# Three “Principles”

1. *Do what makes sense*
2. *Think of HDF5 as a completely new file format*
3. *Anything you can do in HDF4, you can do in HDF5*

# ***1. Do what makes sense***

- The tutorial and documentation are suggestions, not rules.
- Use HDF5 in ways that work best for your goals
  - Sometimes, it may not be best to exactly copy or emulate the way HDF4 was used
  - HDF5 has many new features to exploit

## *2. Think of HDF5 as a new Format*

- Despite the name, the HDF5 Format and Library are new and different.
- Shouldn't expect things to work the same as HDF4 or earlier versions.

### ***3. Anything you can do in HDF4, you can do in HDF5***

- That said, HDF5 is conceptually compatible with HDF4
- It is reasonable to expect that whatever you did with HDF4 can be done with HDF5, usually better.
- This tutorial presents some examples to illustrate common tasks in HDF4, and how to do them with HDF5

# *How to create HDF4 objects in HDF5*

- The six main types of HDF4 objects have equivalents in HDF5
  - HDF4 objects can be represented by specific cases of HDF5 objects
  - Programming model is different, so source code may have no resemblance.
- Will briefly discuss examples

# SDS

- HDF4 SDS is equivalent to an HDF5 Dataset
- HDF5 does not currently support dimension scales
  - Coming in 2003

# Example of SDS

```
int32 sd_id, sds_id;  
int32 dim_sizes[2];  
intn status;
```

```
sd_id = SDstart ("SDS.hdf", DFACC_CREATE);
```

```
dim_sizes[0] = 5;  
dim_sizes[1] = 16;
```

```
sds_id = SDcreate (sd_id, "SDSexample", DFNT_INT32, 2, dim_sizes);  
status = SDendaccess (sds_id);  
status = SDend (sd_id);
```

# Example of HDF5 Dataset

```
hid_t    file_id, dataset_id, dataspace_id, datatype_id;
hsize_t   dims[2];
herr_t    status;
file_id = H5Fcreate("DS.h5", H5F_ACC_TRUNC, H5P_DEFAULT,
                    H5P_DEFAULT);
dims[0] = 5;
dims[1] = 16;

dataspace_id = H5Screate_simple(2, dims, NULL);
datatype_id = H5Tcopy(H5T_STD_I32BE);
dataset_id = H5Dcreate(file_id, "/DSexample", datatype_id, dataspace_id,
                      H5P_DEFAULT);
status = H5Dclose(dataset_id);
status = H5Sclose(dataspace_id);
status = H5Tclose(datatype_id);
status = H5Fclose(file_id);
```

# Vgroup

- HDF4 Vgroup is equivalent to an HDF5 Group
- HDF5 is organized as a rooted, directed graph
  - There are no “lone Vgroups” or equivalent, everything is accessed by pathnames

# Example of VGroup

```
intn status_n;  
int32 status_32, vgroup_ref,  
vgroup1_id, vgroup2_id, file_id;  
  
file_id = Hopen ("vgroup.hdf", DFACC_CREATE, 0);  
status_n = Vstart (file_id);  
  
vgroup1_ref = vgroup2_ref = -1; /* create new group */  
  
vgroup1_id = Vattach (file_id, vgroup1_ref, "w");  
vgroup2_id = Vattach (file_id, vgroup2_ref, "w");  
  
status_32 = Vdetach (vgroup1_id);  
status_32 = Vdetach (vgroup2_id);  
  
status_n = Vend (file_id);  
status_n = Hclose (file_id);
```

# Example of HDF5 Group

```
hid_t    file_id, group1_id, group2_id;  
herr_t   status;  
file_id = H5Fcreate("groups.h5", H5F_ACC_TRUNC, H5P_DEFAULT,  
                    H5P_DEFAULT);  
  
group1_id = H5Gcreate(file_id, "/MyGroup1", 0);  
group2_id = H5Gcreate(file_id, "/MyGroup2", 0);  
  
status = H5Gclose(group1_id);  
status = H5Gclose(group2_id);  
status = H5Fclose(file_id);
```

# Vdatas (Tables)

- HDF4 Vdata is equivalent to an HDF5 Dataset with a Compound Datatype.
  - Because the HDF5 Model is more general, the table can be multidimensional, can have complex records, etc.
- The HDF High Level library provides a relatively simple interface to manage table data.
  - [http://hdf.ncsa.uiuc.edu/HDF5/hdf5\\_hl/doc/](http://hdf.ncsa.uiuc.edu/HDF5/hdf5_hl/doc/)

# Example of Vdata

```
typedef struct s1_t {  
    int   a;  
    float b;  
    double c;  
} s1_t;  
s1_t    data_buf[10];  
  
file_id = Hopen ("vdata.hdf", DFACC_WRITE, 0);  
status_n = Vstart (file_id);  
  
vdata_id = VSattach (file_id, vdata_ref, "w");  
  
status_32 = VSsetname (vdata_id, "Table 1");  
status_32 = VSsetclass (vdata_id, "Some Data");  
  
status_n = VSfdefine (vdata_id, "a_name", DFNT_INT32, 1 );  
status_n = VSfdefine (vdata_id, "b_name", DFNT_FLOAT32, 1 );  
status_n = VSfdefine (vdata_id, "c_name", DFNT_FLOAT64, 1 );  
  
status_n = VSsetfields (vdata_id, "a_name","b_name","c_name");  
  
num_of_records = VSwrite (vdata_id, (uint8 *)data_buf, 10,  
                         FULL_INTERLACE);  
  
status_32 = VSdetach (vdata_id);  
status_n = Vend (file_id);  
status_32 = Hclose (file_id);
```

# Example of HDF5 Compound Data

```
typedef struct s1_t {  
    int  a;  
    float b;  
    double c;  
} s1_t;  
s1_t    s1[10];  
hid_t    s1_tid;  
hid_t    file, dataset, space;  
herr_t   status;  
hsize_t  dim[] = {10};  
space = H5Screate_simple(1, dim, NULL);  
file = H5Fcreate("table.h5", H5F_ACC_TRUNC, H5P_DEFAULT, H5P_DEFAULT);  
s1_tid = H5Tcreate (H5T_COMPOUND, sizeof(s1_t));  
H5Tinsert(s1_tid, "a_name", HOFFSET(s1_t, a), H5T_NATIVE_INT);  
H5Tinsert(s1_tid, "c_name", HOFFSET(s1_t, c), H5T_NATIVE_DOUBLE);  
H5Tinsert(s1_tid, "b_name", HOFFSET(s1_t, b), H5T_NATIVE_FLOAT);  
dataset = H5Dcreate(file, "/Table 1", s1_tid, space, H5P_DEFAULT);  
status = H5Dwrite(dataset, s1_tid, H5S_ALL, H5S_ALL, H5P_DEFAULT, s1);  
H5Tclose(s1_tid);  
H5Sclose(space);  
H5Dclose(dataset);  
H5Fclose(file);
```

# Example using HDF5 “Table” API

```
typedef struct s1_t {  
    int   a;  
    float b;  
    double c;  
} s1_t;  
s1_t    dst_buff[10];  
  
/* Calculate the size and the offsets of the struct members in memory */  
size_t dst_size = sizeof( s1_t );  
size_t dst_offset[3] = { HOFFSET( s1_t ),  
                      HOFFSET( s1_t, b ),  
                      HOFFSET( s1_t, c )};  
  
size_t dst_sizes[3] = { sizeof( dst_buf[0].a ),  
                      sizeof( dst_buf[0].b ),  
                      sizeof( dst_buf[0].c )};  
  
const char *field_names[3] = { "a_name", "b_name", "c_name" };  
hid_t    field_type[3];  
  
field_type[0] = H5T_NATIVE_INT;  
field_type[1] = H5T_NATIVE_FLOAT;  
field_type[2] = H5T_NATIVE_DOUBLE;  
  
file = H5Fcreate(FILE, H5F_ACC_TRUNC, H5P_DEFAULT, H5P_DEFAULT);  
  
err = H5TBmake_table( "Table 1", file_id, "Table1", 3, 10, dst_size, field_names, dst_offset,  
                     field_type, 10, NULL, 0, dst_buff );
```

# GR (Raster Images) and Palettes

- Raster images can be stored as HDF5 Datasets.
- The HDF5 “Image and Palette Specification” defines a standard profile.
  - <http://hdf.ncsa.uiuc.edu/HDF5/doc/ADGuide/ImageSpec.html>
- The HDF High Level library provides a relatively simple interface to manage image data

# Example HDF4 GR image

```
int16 image_buf[2][5][3];  
  
file_id = Hopen ("image.hdf", DFACC_CREATE, 0);  
  
gr_id = GRstart (file_id);  
  
data_type = DFNT_INT8;  
  
dim_sizes[0] = 5;  
dim_sizes[1] = 2;  
  
ri_id = GRcreate (gr_id, "Image 1", 3, data_type,  
                  MFGR_INTERLACE_PIXEL, dim_sizes);  
start[0] = start[1] = 0;  
edges[0] = 5;  
edges[1] = 2;  
  
status = GRwriteimage(ri_id, start, NULL, edges, (VOIDP)image_buf);  
  
status = GRendaccess (ri_id);  
status = GRend (gr_id);  
status = Hclose (file_id);
```

# Example using HDF5 Dataset

```
hsize_t    dims[2];
herr_t    status;
hsize_t    comp_dims[1] = { 3 };
unsigned char data[5][2][3];
file_id = H5Fcreate("image.h5", H5F_ACC_TRUNC, H5P_DEFAULT, H5P_DEFAULT);
dims[0] = 5;
dims[1] = 2;
dataspace_id = H5Screate_simple(2, dims, NULL);
datatype_id = H5Tarray_create( H5T_NATIVE_UINT8, 1, comp_dims, NULL );
dataset_id = H5Dcreate(file_id, "/Image 1", datatype_id, dataspace_id,
                      H5P_DEFAULT);
/* Create the required attributes */
attr_type = H5Tcopy( H5T_C_S1 );
attr_size = strlen( "IMAGE" ) + 1;
H5Tset_size( attr_type, (size_t)attr_size );
H5Tset_strpad( attr_type, H5T_STR_NULLTERM );
attr_space_id = H5Screate( H5S_SCALAR );

attr_id = H5Acreate( dataset_id, "CLASS", attr_type, attr_space_id, H5P_DEFAULT );
H5Awrite( attr_id, attr_type, "IMAGE" );
H5Aclose( attr_id );
/* And so on for the required attributes:
   IMAGE_VERSION="1.2", IMAGE_SUBCLASS="IMAGE_TRUECOLOR", and
   INTERLACE_MODE="INTERLACE_PIXEL" */
H5Sclose( attr_space_id );
H5Dwrite( dataset_id, datatype_id, H5S_ALL, H5S_ALL, H5P_DEFAULT, data);
```

# Example using HDF5 “Image” API

```
hid_t      file_id;  
unsigned char image_data[2][5];  
  
file_id = H5Fcreate( "image.h5", H5F_ACC_TRUNC, H5P_DEFAULT, H5P_DEFAULT );  
  
status = H5IMmake_image_24bit( file_id, "Image 1", 5, 2, image_data )  
  
H5Fcclose(file_id);
```

# Annotations, Global Attributes, etc.

- HDF5 Attributes are attached to an object (usually a Dataset or Group)
  - No equivalent to global attributes or file annotations
  - Recommended that these be made attributes of the root group or similar convention
- HDF5 does not have any predefined attributes

# Example of HDF4 attribute

```
int attr_values[2];  
  
sd_id = SDstart ("attr.hdf", DFACC_WRITE);  
  
sds_index = 0;  
sds_id = SDselect (sd_id, sds_index);  
  
n_values = 2;  
status = SDsetattr (sds_id, "Valid_Range", DFNT_FLOAT32, n_values,  
(VOIDP)attr_values);  
  
status = SDendaccess (sds_id);  
status = SDend (sd_id);
```

# Example of HDF5 attribute

```
hid_t    file_id, dataset_id, attribute_id, dataspace_id;
hsize_t   dims;
int      attr_data[2];
herr_t   status;

file_id = H5Fopen("attr.h5", H5F_ACC_RDWR, H5P_DEFAULT);

dataset_id = H5Dopen(file_id, "/dset");

dims = 2;
dataspace_id = H5Screate_simple(1, &dims, NULL);

attribute_id = H5Acreate(dataset_id, "ValidRange", H5T_STD_I32BE, dataspace_id,
H5P_DEFAULT);

status = H5Awrite(attribute_id, H5T_NATIVE_INT, attr_data);

status = H5Aclose(attribute_id);
status = H5Sclose(dataspace_id);
status = H5Dclose(dataset_id);
status = H5Fclos(file_id);
```

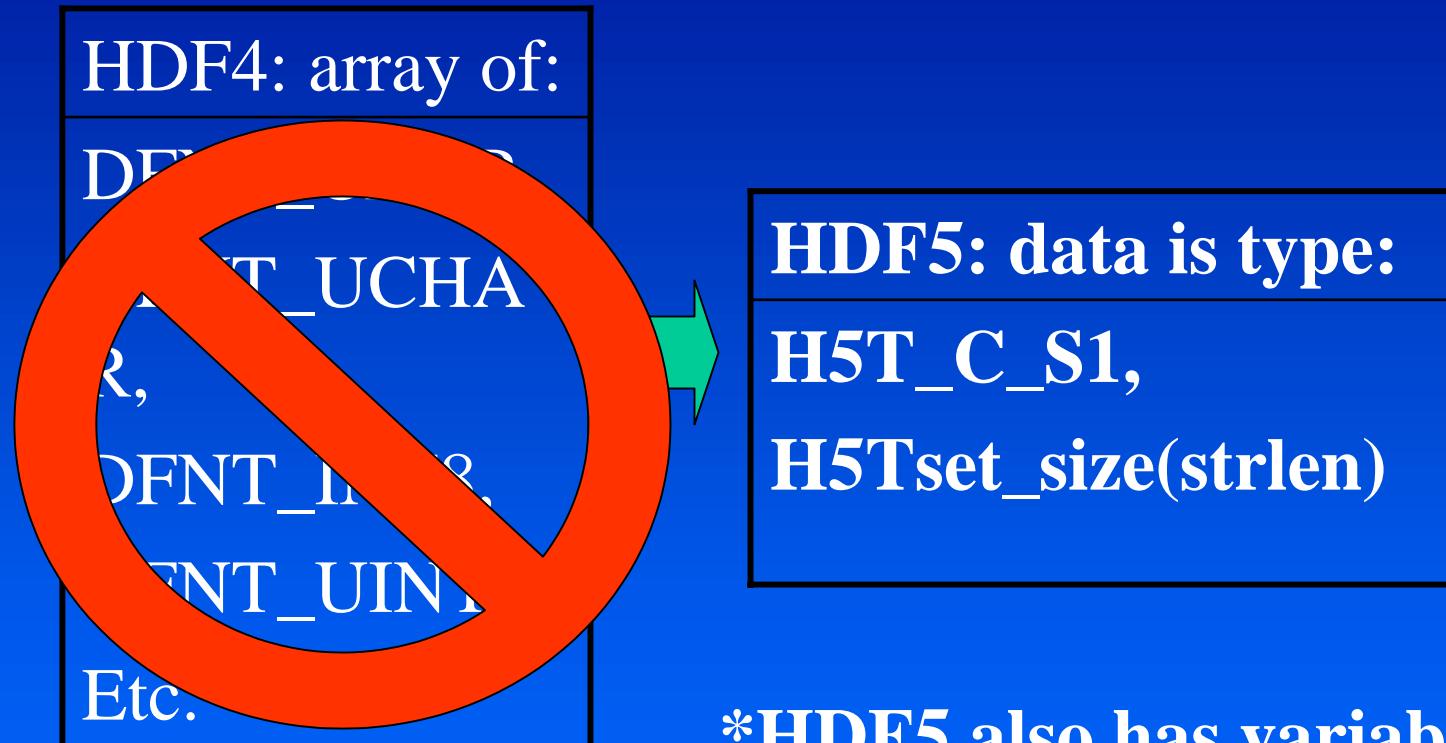
# Data Types

- HDF5 has a much richer model of datatypes than HDF4.
  - Can specify the layout of the data at the source and destination (memory and disk)
- HDF5 has a String datatype
  - *Should not store strings as arrays of characters*

# Corresponding Datatypes

HDF4 type	Corresponding HDF5 type	HDF5 Memory type
DFNT_INT8	H5T_STD_I8BE	H5T_NATIVE_INT8
DFNT_UINT8	H5T_STD_U8BE	H5T_NATIVE_UINT8
DFNT_LINT8	H5T_STD_I8LE	H5T_NATIVE_INT8
DFNT_LUINT8	H5T_STD_U8LE	H5T_NATIVE_UINT8
DFNT_INT16	H5T_STD_U8LE	H5T_NATIVE_UINT8
DFNT_UINT16	H5T_STD_U16BE	H5T_NATIVE_UINT16
DFNT_LINT16	H5T_STD_I16LE	H5T_NATIVE_INT16
DFNT_LUINT16	H5T_STD_U16LE	H5T_NATIVE_UINT16
DFNT_INT32	H5T_STD_I32BE	H5T_NATIVE_INT32
DFNT_UINT32	H5T_STD_U32BE	H5T_NATIVE_UINT32
DFNT_LINT32	H5T_STD_I32LE	H5T_NATIVE_INT32
DFNT_LUINT32	H5T_STD_U32LE	H5T_NATIVE_UINT32
DFNT_FLOAT32	H5T_IEEE_F32BE	H5T_NATIVE_FLOAT
DFNT_LFLOAT32	H5T_IEEE_F32LE	H5T_NATIVE_FLOAT
DFNT_FLOAT64	H5T_IEEE_F64BE	H5T_NATIVE_DOUBLE
DFNT_DFNT_LFLOAT64	H5T_IEEE_F64LE	H5T_NATIVE_DOUBLE

# String Datatypes: Important



\***HDF5 also has variable length strings.**

# Pointers

The Paper: HDF5 for HDF4 Users: a Short Introduction:  
<http://hdf.ncsa.uiuc.edu/....>

1. **The HDF web site:** <http://hdf.ncsa.uiuc.edu>
2. **The helpdesk:** [hdfhelp@ncsa.uiuc.edu](mailto:hdfhelp@ncsa.uiuc.edu)
3. **HDF4 to HDF5 information:** <http://hdf.ncsa.uiuc.edu/h4toh5/>
4. **HDF5 High Level APIs:**  
[http://hdf.ncsa.uiuc.edu/HDF5/hdf5\\_hl/doc/](http://hdf.ncsa.uiuc.edu/HDF5/hdf5_hl/doc/)
5. **“Image and Palette Specification”,**  
<http://hdf.ncsa.uiuc.edu/HDF5/doc/ADGuide/ImageSpec.html>
6. **“HDF5 Table Specification”,**  
[http://hdf.ncsa.uiuc.edu/HDF5/hdf5\\_hl/doc/RM\\_hdf5tb\\_spec.html](http://hdf.ncsa.uiuc.edu/HDF5/hdf5_hl/doc/RM_hdf5tb_spec.html)

# Acknowledgements

This report is based upon work supported in part by a Cooperative Agreement with NASA under NASA grant NAG 5-2040 and NAG NCCS-599. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Aeronautics and Space Administration.  
Other support provided by NCSA and other sponsors and agencies  
(<http://hdf.ncsa.uiuc.edu/acknowledge.html>).