RFC: Supporting Display of Quality Flags in H5dump Tool

Allen Byrne

This RFC proposes a method to display quality flags associated with a dataset using the h5dump tool.

1. Introduction

The concept of NPOESS Quality Flags is to provide for consistently stored, high density, quality information about the delivered data – simplifying usability while maintaining storage efficiency. Quality flags are qualifications of one or more consecutive bits in each byte. Quality flag arrays follow the structure of the data product.

The size of the arrays are equal to or less than the size of the data to which the quality information applies (dimensions correspond to the data product arrays).

Quality flags are stored in the HDF5 files as *n* number(s) of two or three dimensional, 1-byte arrays. The number of arrays is dependent on the quality flag definitions, specific to each data product. Each byte may contain multiple bit-level flags. Quality flags will be ordered such that each flag is entirely contained within a single byte, occasionally resulting in a byte with reserved or meaningless bits.

Byte alignment is the same for every quality flag array. First bit (left-most) is the LSB.

Figure 1 is an example of how quality flags can be referenced. There are five datasets with dataset 2, 3, and 4 containing quality flags. Each datasets has three regions being referenced. Reference VIIRS-IST-EDR_Gran_1[ ] is associated with the second region of each dataset.

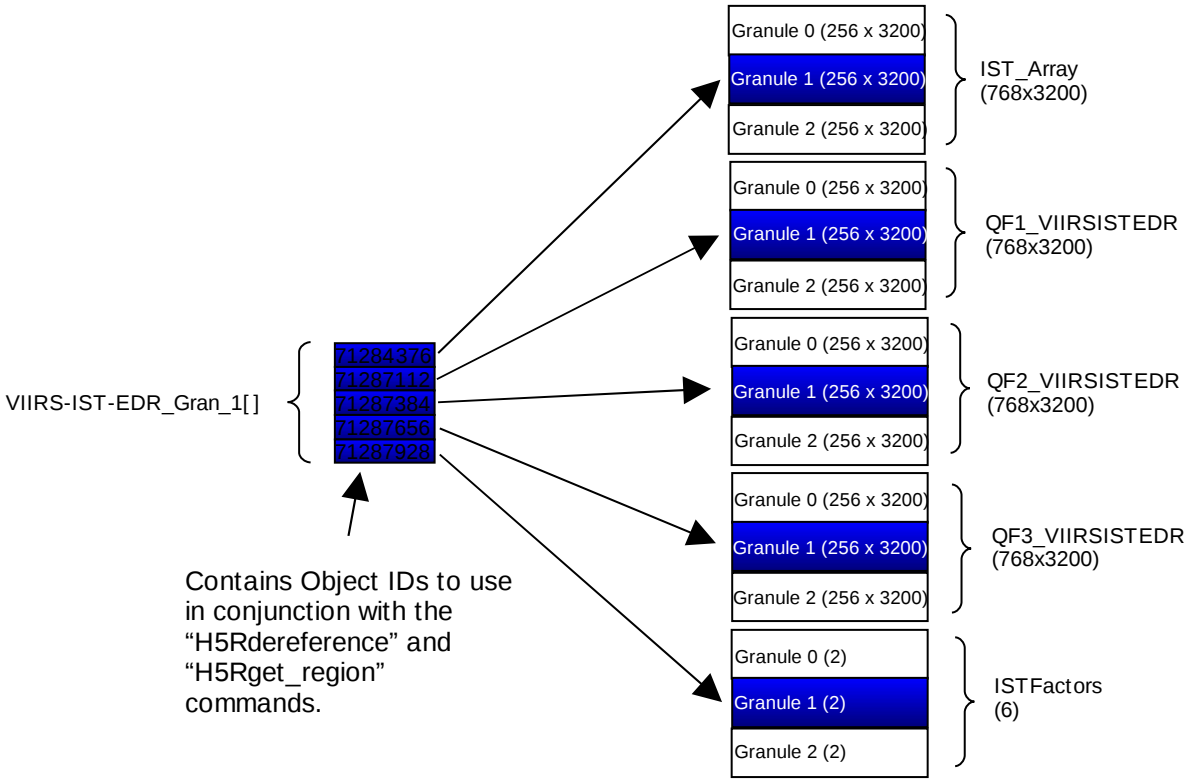

Figure 1

2. Command Option, Display Format

In order for h5dump to display the quality flags associated with a dataset, there must be a command line option and data display format defined. The following command line option is recommended:

```
     -M M, --packed-bits=M    Print packed bits as numbers using format M
for dataset P given in option -d. Where M is offset,length[,offset,length]


  M - is a paired list of integers the first number of which is the offset
and the second number is the length of the its being queried
```

The recommendation for the quality flag display format is that the data be displayed as usual in the data block.

```
>H5dump -d /Data_Products/VIIRS-IST-EDR/VIIRS-IST-EDR_Gran_1 sds.h5


HDF5 "sds.h5" {
   DATASET "/Data_Products/VIIRS-IST-EDR/VIIRS-IST-EDR_Gran_1" {
      DATATYPE  H5T_STD_U8BE
      DATASPACE  SIMPLE { ( 768, 3200 ) / ( H5S_UNLIMITED, H5S_UNLIMITED ) }
      DATA {
      (0,0): 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
      (0,19): 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
      (0,38): 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
...
      (767,3191): 65533, 65533, 65533, 65533, 65533, 65533, 65533,
      (767,3198): 65533, 65533
      }
   }
}

>H5dump -d /Data_Products/VIIRS-IST-EDR/VIIRS-IST-EDR_Gran_1 -M 0,2 sds.h5


HDF5 "sds.h5" {
   DATASET "/Data_Products/VIIRS-IST-EDR/VIIRS-IST-EDR_Gran_1" {
      DATATYPE  H5T_STD_U8BE
      DATASPACE  SIMPLE { ( 768, 3200 ) / ( H5S_UNLIMITED, H5S_UNLIMITED ) }
      PACKED_BITS OFFSET=0 LENGTH=2
      DATA {
      (0,0): 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
      (0,19): 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
      (0,38): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
...
      (767,3191): 3, 3, 3, 3, 3, 3, 3,
      (767,3198): 3, 3
      }
   }
}
```

3. Additional Considerations

The above example has the entire dataset with one quality flag. The subsetting option would be used to specify a portion of the dataset. Additional quality flags would be displayed as a separate data block within a dataset block. For example:

```
>H5dump -d /Data_Products/VIIRS-IST-EDR/VIIRS-IST-EDR_Gran_1 -M 0,2,2,1 sds.h5


HDF5 "sds.h5" {
   DATASET "/Data_Products/VIIRS-IST-EDR/VIIRS-IST-EDR_Gran_1" {
      DATATYPE  H5T_STD_U8BE
      DATASPACE  SIMPLE { ( 768, 3200 ) / ( H5S_UNLIMITED, H5S_UNLIMITED ) }
      PACKED_BITS OFFSET=0 LENGTH=2
      DATA {
      (0,0): 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
      (0,19): 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
      (0,38): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
...
      (767,3191): 3, 3, 3, 3, 3, 3, 3,
      (767,3198): 3, 3
      }
      PACKED_BITS OFFSET=2 LENGTH=1
```

```
      DATA {
      (0,0): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      (0,19): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      (0,38): 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
...
      (767,3191): 0, 0, 0, 0, 0, 0, 0,
      (767,3198): 0, 0
      }
   }
}
```

4. BNF Changes

The BNF description would change as follows:

```
<dataset_info> ::= <dataset_type>  <dataset_space> <storagelayout>opt
                   <compression>opt <dataset_attribute>* <object_id>opt
                   <packed_bits>opt
                   <data>opt

<packed_bits> ::= PACKED_BITS OFFSET=<int_value> LENGTH=<int_value>
```


Acknowledgements

This work was partially supported by The HDF GROUP and NPOESS contract.

Revision History

April 30, 2009:	*Version 1 first draft.*
May 6, 2009:	*Version 2, second draft.*
May 7, 2009:	*Version 2, changed the mask parameter to M*
	added PACKED_BITS OFFSET=N LENGHT=M
May 8, 2009:	*Version 3, added mutiple quality flags*
	added BNF changes
May 8, 2009:	*Version 3.2, labeled and described Figure 1.*
	corrected format name from F to M

[References]

1. “Profile of National Polar-Orbiting Operational Satellite System (NPOESS) HDF5 Files”, *Kim Tomashosky, Ken Stone, Pat Purcell, Ron Andrews, HDF and HDF-EOS Workshop X, 2006, Landover, Maryland,* *http://www.hdfeos.net/workshops/ws10/presentations/day3/Profile_of_NPOESS_HDF5_Files.ppt*

2. “NPP/ NPOESS Product Data Format”, *Richard E. Ullman, HDF and HDF-EOS Workshop XI, 2007, Landover, Maryland, http://www.hdfeos.net/workshops/ws11/presentations/day2/NPOESS-Format-Talk.ppt*

3. “HDF Group Support for NPP/NPOESS”, *Mike Folk, HDF and HDF-EOS Workshop XII, 2008, Aurora, Colorado, http://www.hdfeos.net/workshops/ws12/presentations/day3/mxf.ppt*

Appendix: Background Material

This appendix contains background material, including assumptions and high-level design decisions, which guided the final approach presented in the body of the RFC.

This is not required, but is often a good idea. Especially as you work through the various versions of the RFC, or even before you begin to write – if you note the reasons you opted for one approach over another, or perhaps include content from an earlier version that was changed based on feedback, it's good to have a record of those things. Memories are not perfect, and sometime thought processes that are documented help readers (and developers) understand better why you are going with the approach suggested.