H4CF Conversion Library

1.0.beta

Generated by Doxygen 1.6.1

Tue Jun 4 12:24:57 2013

Contents

Chapter 1

The H4CF Conversion Library

Version:

1.0.beta

1.1 Introduction

The H4CF Conversion Library converts both HDF-EOS2 and HDF4 files by following the CF conventions. The variables and attributes of the converted HDF-EOS2 and HDF4 files are accessible through a set of high-level APIs described here.

Chapter 2

File Documentation

2.1 h4cf.h File Reference

Has all functions for the H4CF Conversion Library.

Functions

- void **h4cf_open** (char *filename)
  Opens an existing HDF-EOS2 or HDF4 file.
- const map< string, int > **h4cf_get_dims** ()
  Retrieves pairs of name and size of dimension in the file.
- const list< var * > **h4cf_get_vars** ()
  Retrieves variables in the file.
- const string **h4cf_get_var_name** (var *v)
  Retrieves the name of a variable pointed by v.
- const vector< map< string, int > > **h4cf_get_var_dims** (var *v)
  Retrieves the dimensions of a variable pointed by v.
- const h4cf_data_type **h4cf_get_var_type** (var *v)
  Retrieves the variable type of a variable pointed by v.
- const int **h4cf_get_var_rank** (var *v)
  Retrieves the rank of a variable pointed by v.

- void **h4cf_get_var_value** (vector< char > *buf, var *v)

  Retrieves data values of a variable pointed by v and stores them into buf.

- void **h4cf_get_var_value** (vector< char > *buf, var *v, int32 *start, int32 *stride, int32 *edge)

  Retrieves subset data values of a variable pointed by v and stores them into buf.

- const list< attr * > **h4cf_get_file_attrs** ()

  Retrieves file attributes.

- const list< attr * > **h4cf_get_var_attrs** (var *v)

  Retrieves the attributes of a variable pointed by v.

- void **h4cf_get_attr_value** (vector< char > *buf, attr *a)

  Retrieves the data values of an attribute pointed by a and stores them into buf.

- const string **h4cf_get_attr_name** (attr *a)

  Retrieves the name of an attribute pointed by a.

- const h4cf_data_type **h4cf_get_attr_type** (attr *a)

  Retrieves the attribute type of an attribute pointed by a.

- const int **h4cf_get_attr_count** (attr *a)

  Retrieves the number of elements stored in an attribute pointed by a.

- const attr * **h4cf_get_var_attr_by_name** (string str, var *v)

  Retrieves the attribute that has str name from a variable pointed by v.

- void **h4cf_close** ()

  Closes the access to the opened file.

2.1.1 Detailed Description

The **h4cf.h** **(p. 3)** contains all APIs that user need to know to access the variables and attributes in HDF4 files following the CF conventions.

2.1.2 Function Documentation

2.1.2.1 void h4cf_close ()

h4cf_close terminates access to the opened file by releasing the resources held by the library. The opened file should be closed by calling this function when it is no longer needed.

Returns:

none

2.1.2.2 const int h4cf_get_attr_count (attr * *a*)

h4cf_get_attr_count returns the number of elements stored in an attribute pointed by *a*.

For example, if `coordsys` attribute has type `CHAR8` and value "Cartesian", the count will be 9. If `valid_range` attribute has type `INT8` and value "0, -2", the count will be 2.

Parameters:

a a pointer to an attribute

Returns:

a number of elements.

2.1.2.3 const string h4cf_get_attr_name (attr * *a*)

h4cf_get_attr_name returns the name of an attribute pointed by *a*. The attribute name follows the CF conventions.

Parameters:

a a pointer to an attribute

Returns:

a string

2.1.2.4 const h4cf_data_type h4cf_get_attr_type (attr * a)

h4cf_get_attr_type returns the data type of an attribute pointed by *a*. The data type can be:

- CHAR8
- UCHAR8
- INT8
- UINT8
- INT16
- UINT16
- INT32
- UINT32
- FLOAT
- DOUBLE

Parameters:

a a pointer to an attribute

Returns:

a data type

2.1.2.5 void h4cf_get_attr_value (vector< char > * *buf*, attr * *a*)

h4cf_get_attr_value reads the data values stored in an attribute pointed by *a* and saves them into *buf* vector.

Remarks:

For the first parameter *buf*, user does not need to specify its capacity. The storage of vector will be allocated by the library, and the actual size of vector is equal to the number of bytes the attribute holds.

Parameters:

→ ***buf*** a pointer to store values
← ***a*** a pointer to an attribute

Returns:

none

2.1.2.6 const map<string, int> h4cf_get_dims ()

h4cf_get_dims retrieves the dimension information and returns the pairs of name and size of dimension in map. The name of dimension follows the CF conventions.

For example, if the opening file has two dimensions `XDim` and `YDim` with their size 360 and 180 respectively, the returned map will be

- map[XDim] = 360
- map[YDim] = 180

Returns:

a map containing dimension definitions. The key value in map is the name of the dimension and the mapped value is the size of the dimension.

2.1.2.7 const list<attr*> h4cf_get_file_attrs ()

h4cf_get_file_attrs returns the list of attributes.

Returns:

a list of attributes.

2.1.2.8 const attr* h4cf_get_var_attr_by_name (string *str*, var * *v*)

h4cf_get_var_attr_by_name returns the pointer to the attribute in a variable pointed by *v* if the attribute's name matches the *str* parameter.

Parameters:

str an attribute name to be searched for

v a pointer to a variable

Returns:

a pointer to the matching attribute if present, otherwise NULL.

2.1.2.9 const list<attr*> h4cf_get_var_attrs (var * v)

h4cf_get_var_attrs returns the list of attributes in a variable pointed by v.

Parameters:

v a pointer to a variable

Returns:

a list of attributes.

2.1.2.10 const vector< map<string, int> > h4cf_get_var_dims (var * v)

h4cf_get_var_dims retrieves the dimensions of a given variable and returns them in a C++ vector. Each dimension is a pair of name and size. The name of dimension follows the CF conventions.

For example, for a variable with two dimensions `XDim` and `YDim` with their size `360` and `180` respectively, the returned vector will be:

- vector[0] = <XDim, 360>
- vector[1] = <YDim, 180>

Parameters:

v a pointer to a variable

Returns:

a vector of maps that have dimension name and size

2.1.2.11 const string h4cf_get_var_name (var * v)

h4cf_get_var_name returns the name of a variable pointed by v. The variable name follows the CF conventions.

Parameters:

v a pointer to a variable

Returns:

a string

2.1.2.12 const int h4cf_get_var_rank (var * v)

h4cf_get_var_rank returns the number of dimensions of a variable pointed by *v*. For example, if *v* is O3[10][20][30], this function will return 3.

Parameters:

v a pointer to a variable

Returns:

the rank of variable

2.1.2.13 const h4cf_data_type h4cf_get_var_type (var * v)

h4cf_get_var_type returns the data type of a variable pointed by *v*. The data type can be:

- `CHAR8`
- `UCHAR8`
- `INT8`
- `UINT8`
- `INT16`
- `UINT16`
- `INT32`
- `UINT32`
- `FLOAT`
- `DOUBLE`

Parameters:

v a pointer to a variable

Returns:

a data type

2.1.2.14 void h4cf_get_var_value (vector< char > * *buf*, var * *v*, int32 * *start*, int32 * *stride*, int32 * *edge*)

h4cf_get_var_value returns the *subset* data values stored in a variable pointed by *v* and saves them into *buf* vector. The subsetting is controlled by the parameters stored in *start*, *stride*, and *edge*.

For example, if *v* has values like:

- v[0] = 0
- v[1] = 1
- v[2] = 2
- v[3] = 3

specifying start[0] = 1, stride[0] = 2, and edge[0] = 2 to this function will return

- buf[0] = 1
- buf[1] = 3.

Remarks:

For the first parameter *buf*, user does not need to specify its capacity. The storage of vector will be allocated by the library, and the actual size of vector is equal to the number of bytes the variable holds.

Parameters:

→ ***buf*** a pointer to store values

← ***v*** a pointer to a variable

← ***start*** a pointer to array containing the position at which this function will start for each dimension

← ***stride*** a pointer to array specifying the interval between the data values that will be read along each dimension

← ***edge*** a pointer to array containing the number of data elements along each dimension

Returns:

none

2.1.2.15 void h4cf_get_var_value (vector< char > * *buf*, var * *v*)

h4cf_get_var_value reads the data values stored in a variable pointed by *v* and saves them into the *buf* vector.

Remarks:

For the first parameter *buf*, a user does not need to specify its capacity. The storage of vector will be allocated by the library, and the actual size of vector is equal to the number of bytes the variable holds.

Parameters:

→ ***buf*** a pointer to store values

← ***v*** a pointer to a variable

Returns:

none

2.1.2.16 const list<var*> h4cf_get_vars ()

h4cf_get_vars returns a list of pointers of all variables in the file.

Returns:

a list containing variable pointers.

2.1.2.17 void h4cf_open (char * *filename*)

h4cf_open opens *filename* file and initializes the library.

Parameters:

filename name of the file to be opened.

Index