L2 RMIB GERB Products
HDF Archiving Procedure

Gilles Sadowski

September 8, 2000

Contents

CHANGE RECORD

Issue	Date	Changed by	Reason for change
Version 1	05/9/2000		new document

1 Purpose

This document describes the interface between the RMIB processing end products and the archiving programs that translates from the RMIB internal data format into HDF files and vice versa.

This document is organised as follows:

- Section 2 exposes the interface between the RMIB internal data format and the L2 products in HDF format.
- Section 3 lists the source files of the interface programs and the script that helped producing them.
- Section 4 presents the test procedures that have been used to check the translation programs.

2 Translation from Internal Data Types to HDF Data Types

2.1 Context

This section describes how the Level 2 (L2) products derived at RMIB are translated from the RMIB internal data format to Hierarchical Data Format (HDF) files.

2.2 From GERB Processing End Products To HDF Archive

The translation from the internal data format used for the calculations and image processing to the HDF format is performed by a C program, using the API from the HDF5 Library. The program that performs the translation is named “`l2_write_hdf`”. It can generate any of the fourteen products described in the *“L2 RMIB GERB Products User Guide”* (MSG-RMIB-GE-UG). It is run by the following command:

```
l2_write_hdf <information file> <hdf output directory>
```

where `<information file>` is a simple text file that comprises all the information needed to gather the data to be stored in the HDF file and `<hdf output directory>` is the name of an existing directory where the HDF file will be written to.

The `<information file>` is composed from successive lines. Each line must contain a pair of character strings separated by the “=” character; thus the strings themselves must not contain an “=” character. The first string is a key identifying which HDF field corresponds to the data. the second string is either the data (number or string) to be inserted in the HDF file or the name a an existing file the contents of which will then be inserted in the HDF file. In the latter case, the name can refer either to an RMA file (containing an image) or to a text file (containing one column of data). Below is a template of the contents of the information file.

product_id=<*string*>
radiation_type_id=<*string*>
file_name=<*string*>
thermal_flux_image=<*file_name*>
solar_flux_image=<*file_name*>
thermal_radiance_image=<*file_name*>
solar_radiance_image=<*file_name*>
cloud_cover_image=<*file_name*>
cloud_phase_image=<*file_name*>
cloud_amount_image=<*file_name*>
surface_type_image=<*file_name*>
viewing_zenith_image=<*file_name*>
solar_zenith_image=<*file_name*>
relative_azimuth_image=<*file_name*>
viewing_azimuth_image=<*file_name*>
latitude_image=<*file_name*>
longitude_image=<*file_name*>
sw_correction_image=<*file_name*>
lw_correction_image=<*file_name*>
thermal_adm_image=<*file_name*>
solar_adm_image=<*file_name*>
time_row_ascii=<*file_name*>
integration_start_column_ascii=<*file_name*>
integration_end_column_ascii=<*file_name*>
gerb_a_values_ascii=<*file_name*>
gerb_c_values_ascii=<*file_name*>
imager_spectral_regression_ascii=<*file_name*>

ggsps_product_version=<*string*>
ggsps_file_name=<*string*>
rmib_sw_id=<*string*>
gerb_instrument_id=<*string*>
imager_instrument_id=<*string*>
imager_type=<*string*>
first_gerb_packet=<*string*>
last_gerb_packet=<*string*>
integration_start=<*string*>
integration_end=<*string*>
geolocation_file=<*string*>
rg_nx=<*string*>
rg_ny=<*string*>
rg_lap=<*string*>
rg_lop=<*string*>
rg_rf_ns=<*string*>
rg_rf_ew=<*string*>
rg_dx=<*string*>
rg_dy=<*string*>
rg_xp=<*string*>
rg_yp=<*string*>
rg_orientation=<*string*>
rg_nr=<*string*>
duplication=<*string*>
sw1_los_ew_histo_image=<*file_name*>
sw2_los_ew_histo_image=<*file_name*>
sw3_los_ew_histo_image=<*file_name*>
los_ns_speed=<*string*>
total1_los_ew_histo_image=<*file_name*>
total2_los_ew_histo_image=<*file_name*>
total3_los_ew_histo_image=<*file_name*>
product_confidence_ascii=<*file_name*>
thermal_adm_version=<*string*>
solar_adm_version=<*string*>
resolution_enhancement_ascii=<*file_name*>

Tables 1 and 2 summarise all the fields that appear in the L1.5 NANRG and L2 products (see MSG-RMIB-GE-IF-0002). The column labelled "Source" indicates whether the second string refers to a file (`RMA` or `ASCII`). The column labelled "HDF Structure" indicates which type of HDF structure is used to store the data (`H5D` stands for *Dataset* and `H5A` for *Attribute*).

ID	Description	Source (Data Type)	HDF Structure (Data Type)	Quantization Range
1	Thermal Flux	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{11}, 2^{11}[\rightarrow [-2^{13}, 2^{13}[$
2	Solar Flux	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{11}, 2^{11}[\rightarrow [-2^{13}, 2^{13}[$
3	Thermal Radiance	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{11}/5, 2^{11}/5[\rightarrow [-2^{13}, 2^{13}[$
4	Solar Radiance	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{11}/5, 2^{11}/5[\rightarrow [-2^{13}, 2^{13}[$
5	Cloud Cover	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
6	Cloud Phase	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
7	Cloud Amount	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
8	Surface Type	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
9	Viewing Zenith Angle	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I8BE)	$[-2^{7}, 2^{7}[\rightarrow [-2^{7}, 2^{7}[$
10	Solar Zenith Angle	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I8BE)	$[-2^{7}, 2^{7}[\rightarrow [-2^{7}, 2^{7}[$
11	Relative Azimuth Angle	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{9}, 2^{9}[\rightarrow [-2^{9}, 2^{9}[$
12	Viewing Azimuth Angle	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{9}, 2^{9}[\rightarrow [-2^{9}, 2^{9}[$
13	Latitude	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{7}, 2^{7}[\rightarrow [-2^{14}, 2^{14}[$
14	Longitude	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I16BE)	$[-2^{8}, 2^{8}[\rightarrow [-2^{15}, 2^{15}[$
15	LW Correction	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I8BE)	$[-2^{6}, 2^{6}[\rightarrow [-2^{7}, 2^{7}[$
16	SW Correction	RMA (H5T_NATIVE_FLOAT)	H5D (H5T_STD_I8BE)	$[-2^{6}, 2^{6}[\rightarrow [-2^{7}, 2^{7}[$
17	Thermal ADM	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
18	Solar ADM	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
19	LOS E-W Speed Histo SW	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A
20	LOS E-W Speed Histo TOTAL	RMA (H5T_NATIVE_UCHAR)	H5D (H5T_STD_U8BE)	N/A

Table 1: Image Fields

ID	Description	Source (Data Type)	HDF Structure (Data Type)
1	File name	string (H5T_C_S1)	H5A (H5T_C_S1)
2	GGSPS file name	string (H5T_C_S1)	H5A (H5T_C_S1)
3	GGSPS product version	string (H5T_C_S1)	H5A (H5T_STD_I32BE)
4a	GGSPS table names	ASCII (H5T_C_S1)	H5A (H5T_C_S1)
4b	GGSPS table version	ASCII (H5T_C_S1)	H5A (H5T_C_S1)
4c	GGSPS sw version	string (H5T_C_S1)	H5A (H5T_C_S1)
4d	GGSPS input GL0 files	ASCII (H5T_C_S1)	H5A (H5T_C_S1)
4e	GGSPS input seviri files	ASCII (H5T_C_S1)	H5A (H5T_C_S1)
5	RMIB sw identifier	string (H5T_C_S1)	H5A (H5T_C_S1)
6	Time of file creation	N/A	H5A (H5T_C_S1)
7	GERB instrument identifier	string (H5T_NATIVE_INT)	H5A (H5T_STD_I32BE)
8a	Imager type	string (H5T_C_S1)	H5A (H5T_C_S1)
8b	Imager identifier	string (H5T_NATIVE_INT)	H5A (H5T_STD_I32BE)
9a	First GERB packet	string (H5T_NATIVE_INT)	H5A (H5T_C_S1)
9b	Start of integration	string (H5T_NATIVE_DOUBLE)	H5A (H5T_C_S1)
10	Start of integration (per column)	ASCII (H5T_NATIVE_DOUBLE)	H5A (H5T_C_S1)
11a	Last GERB packet	string (H5T_NATIVE_INT)	H5A (H5T_C_S1)
11b	End of integration	string (H5T_NATIVE_DOUBLE)	H5A (H5T_C_S1)
12	End of integration (per column)	ASCII (H5T_NATIVE_DOUBLE)	H5A (H5T_C_S1)
13	Time (per row)	ASCII (H5T_NATIVE_DOUBLE)	H5A (H5T_C_S1)
14	File name for geolocation	string (H5T_C_S1)	H5A (H5T_C_S1)
15	Rectified grid data (RG): Nx	string (H5T_NATIVE_INT)	H5A (H5T_STD_I32BE)
16	RG: Ny	string (H5T_NATIVE_INT)	H5A (H5T_STD_I32BE)
17	RG: Lap	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
18	RG: Lop	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
19a	RG: Resolution flag NS	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
19b	RG: Resolution flag EW	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
20	RG: dx	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
21	RG: dy	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
22	RG: Xp	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
23	RG: Yp	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
24	RG: Grid orientation	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
25	RG: Nr	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
26	Product confidence flags	ASCII (H5T_NATIVE_INT)	H5D (H5T_STD_I32BE)
27	Duplication flag	string (H5T_NATIVE_INT)	H5A (H5T_STD_I32BE)
28	LOS N-S speed	string (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
30	A values (per GERB d.c.)	ASCII (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
31	C values (per GERB d.c.)	ASCII (H5T_NATIVE_DOUBLE)	H5A (H5T_IEEE_F64BE)
32	Spectral regression	ASCII (H5T_NATIVE_DOUBLE)	H5D (H5T_IEEE_F64BE)
33	Resolution enhancement	ASCII (H5T_NATIVE_DOUBLE)	H5D (H5T_IEEE_F64BE)
34	Thermal ADM Set Version	string (H5T_C_S1)	H5A (H5T_C_S1)
35	Solar ADM Set Version	string (H5T_C_S1)	H5A (H5T_C_S1)

Table 2: Auxiliary Fields

3 Source Code Management

3.1 Files

- Interface with HDF5 Library:

  “l2_hdf.h”
  “l2_hdf.c”

  These files provide the interface between the RMIB internal data format and HDF, for writing (and reading) all the HDF fields used in the GERB L1.5 NANRG and L2 products. They include all the direct calls to the functions provided in the HDF5 library.

- Definition of information file keys:

  “l2_hdf_keys.h”

  This file provides the definition of the character strings used to indicate the value or file location of the data that must be archived into (or read from) the HDF files.

- Name of HDF fields:

  “l2_hdf_fields.h”

- Templates:

  “L2_ALL_w.h”
  “L2_ALL_r.h”
  “L2_ALL_w.c”
  “L2_ALL_r.c”

  These files contain all the code used to make the programs for writing (and reading) HDF files for the L1.5 NANRG and L2 products.

3.2 Tool

- The script “l2_generate.pl” generates the C code of the functions contained in each of the following files:

  “l15_nanrg_w.c”
  “l15_nanrg_r.c”
  “l2_50_f_sol_w.c”
  “l2_50_f_sol_r.c”
  “l2_50_f_th_w.c”
  “l2_50_f_th_r.c”
  “l2_50_geo_w.c”
  “l2_50_geo_r.c”
  “l2_50_l_sol_w.c”
  “l2_50_l_sol_r.c”
  “l2_50_l_th_w.c”
  “l2_50_l_th_r.c”
  “l2_archive_w.c”
  “l2_archive_r.c”
  “l2_f_sol_w.c”
  “l2_f_sol_r.c”
  “l2_f_th_w.c”
  “l2_f_th_r.c”
  “l2_geo_w.c”

"l2_geo_r.c"
"l2_h_f_sol_w.c"
"l2_h_f_sol_r.c"
"l2_h_f_th_w.c"
"l2_h_f_th_r.c"
"l2_h_geo_w.c"
"l2_h_geo_r.c"
"l2_h_l_sol_w.c"
"l2_h_l_sol_r.c"
"l2_h_l_th_w.c"
"l2_h_l_th_r.c"

For each of these files, the corresponding header file is also created. The needed functions are extracted from the template files (see above). Each ".c" file must contain the definitions of 2 static variables:

- `static char const * const mandatory_keys[]`

  The values of the elements must be taken from the `#define`s ending in "`_KEY`" in the file "`l2_hdf_keys.c`".

- `static char const * const groups[]`

  The values of the elements must be taken from the `#define`s ending in "`_GROUP`" in the file "`l2_hdf_fields.c`".

The rest of the file in generated automatically by the script.

4 Testing

This section describes the tests performed to check the programs that perform the translation between the RMIB internal data format and HDF.

4.1 L2 RMIB GERB Products

1. Purpose:

   Check the execution of the program "`l2_write_hdf`".

2. Inputs:

   - 1 information file (see section 2.2).
   - All the necessary data files (as specified in the information file) in RMIB internal format.

3. Method:

   (a) Command line:

   `l2_write_hdf <information file name> <directory name>`

   where `<information file name>` is the full path to the information file and `<directory name>` is the directory where the HDF file must be created.

   (b) The created HDF file is read back by a program called "`l2_read_hdf`" designed to perform the operation inverse to "`l2_write_hdf`". Indeed it will read the HDF file and recreate the input data files in RMIB internal format as well as an information file (see section 4.2 below).

   (c) Data which has been read back is compared to the input data. Script "`txtdiff.pl`" compares ascii text files, script "`rmadiff.pl`" compares image binary files and script "`infodiff.pl`" compares the data contained in the information files.

   - A retrieved text file should contains identical lines as the input file except, maybe, for trailing white space and fractional number format (number of digits after fractional point).
   - A retrieved image should have identical values as the input image unless a quantisation was performed. In that case the difference between input value and retrieved value should be less (in absolute value) than half the quantisation factor.
   - An information file should be identical to the input information file except, maybe, for trailing white space, fractional number format (number of digits after fractional point) and comment lines (with a '#' as the first character of the line).

4. Outputs:

   - 1 HDF file, containing the L2 RMIB GERB Product specified by the "`product_id`" key in the information file.
   - Error messages are managed by the RMIB library code.

5. Pass/Fail Criteria:

   - This program is expected to create any of the 14 L2 RMIB GERB Products, where each product consists in 1 HDF file.
   - The contents of the HDF files must correspond to the description provided in the RAL-RMIB ICD (MSG-RAL-GE-IF-0011) and the *"L2 RMIB GERB Products User Guide"* (MSG-RMIB-GE-UG).
   - The outputs of test scripts "`txtdiff.pl`", "`rmadiff.pl`" and "`infodiff.pl`" should be as expected (see above).

6. Date: July 2000.

7. Status: **OK**.

4.2 L1.5 NANRG RAL GERB Product

1. Purpose:

   Check the execution of the program "`l2_read_hdf`", when reading an L1.5 NANRG product HDF file.

2. Inputs:

   - 1 information file (see section 2.2).
   - 1 HDF file obtained from RAL and containing the L1.5 NANRG RAL GERB Product.

3. Method:

   (a) Command line:

   ```
   l2_read_hdf <information file name> <directory name>
   ```

   where `<information file name>` is the full path to the information file and `<directory name>` is the directory where the HDF file is located.

4. Outputs:

   - 1 information file. Lines containing keys the values of which refer to file paths are copied from the input information file. Lines containing keys the values of which are strings are filled with the values retrieved from the L1.5 NANRG HDF file.
   - Text and binary files containing data, in RMIB internal format, retrieved from the L1.5 NANRG HDF file.
   - Error messages are managed by the RMIB internal library code.

5. Pass/Fail Criteria:

   - This program is expected to read the L1.5 NANRG RAL GERB Product, which consists in 1 HDF file.
   - The contents of the HDF file must correspond to the description provided in the RAL-RMIB ICD (MSG-RAL-GE-IF-0011).

6. Date: ?

7. Status: **TBD** (ongoing).