Note on the Coriolis Matlab decoder of NKE Argo floats

This note describes the contents of the Coriolis Matlab decoder package, how to install, configure and use it.

[Note on the Coriolis Matlab decoder of NKE Argo floats 1](#_Toc422130505)

[1. Quick look at the decoder 2](#_Toc422130506)

[2. Description of the decoder package contents 2](#_Toc422130507)

[2.1. Note on the](#_Toc422130508) *[\_provor\_floats\_information\_co\_rt.xls](#_Toc422130508)* [file 3](#_Toc422130508)

[2.2. Note on the meta-data JSON files 4](#_Toc422130509)

[3. Matlab version to use 4](#_Toc422130510)

[4. Decoder deployment 4](#_Toc422130511)

[5. Decoder configuration 5](#_Toc422130512)

[6. How to use the decoder 6](#_Toc422130513)

[7. Note on the Delayed Mode decoding mode 7](#_Toc422130514)

[8. Additional tools 8](#_Toc422130515)

[8.1. Tools configuration 8](#_Toc422130516)

[8.2. Main tools 9](#_Toc422130517)

[8.2.1. clean\_spool\_and\_buff 9](#_Toc422130518)

[8.2.2. copy\_iridium\_mail\_files and copy\_remocean\_sbd\_files 9](#_Toc422130519)

[8.2.3. generate\_json\_decoder\_config\_labels and generate\_json\_decoder\_tech\_labels 9](#_Toc422130520)

[8.2.4. generate\_json\_float\_meta\_argos, generate\_json\_float\_meta\_ir\_sbd, generate\_json\_float\_meta\_remocean and generate\_json\_float\_meta\_remocean\_flbb 9](#_Toc422130521)

[8.2.5. get\_meta\_data\_from\_data\_base and get\_meta\_data\_from\_nc 9](#_Toc422130522)

[8.2.6. nc\_add\_rtqc\_flags 9](#_Toc422130523)

[8.2.7. nc\_check\_file\_format 10](#_Toc422130524)

[8.2.8. nc\_meta\_2\_csv, nc\_prof\_2\_csv, nc\_prof\_adj\_2\_csv, nc\_tech\_2\_csv and nc\_traj\_2\_csv 10](#_Toc422130525)

[8.2.9. nc\_trace\_disp, nc\_trace\_param and nc\_trace\_times 10](#_Toc422130526)

[9. First step procedure 10](#_Toc422130527)

[10. Second step procedure 11](#_Toc422130528)

[10.1. Processing of new incoming float data 11](#_Toc422130529)

[10.1.1. For Argos floats 11](#_Toc422130530)

[10.1.2. For Iridium SBD floats 13](#_Toc422130531)

[10.2. Declaration of new floats 13](#_Toc422130532)

[10.2.1. Find the decoder Id of the float 13](#_Toc422130533)

[10.2.2. Update the \_provor\_floats\_information\_co\_rt.xls file 13](#_Toc422130534)

[10.2.3. Create the json meta-data file of the float 13](#_Toc422130535)

[11. Third step procedure 14](#_Toc422130536)

[12. Miscellaneous information 14](#_Toc422130537)

# Quick look at the decoder

You can find in the decoder package:

* In *decPrv\_20150409\_001a\_for\_incois\decPrv\_doc\MUT\_decPrv\Matlab\_Provor\_decoder\_20150325.pptx*, a brief presentation of the decoder,
* In *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\\_VersionsLogiciellesProvor\_20150407.xlsx*, the Coriolis version of each managed float version and the associated decoder Id,
* In *decPrv\_20150409\_001a\_for\_incois\decPrv\_doc\MUT\_floats*, the NKE manuals of the float decoded by this decoder.

As you can see in the first (.pptx) presentation, there are 4 different manners to use the core decoder:

* 2 are designed for the PIs (to decode already received data): decode\_provor\_2\_csv and decode\_provor\_2\_nc,
* 2 are designed for the DACs (to decode the data incoming in a real time flux): decode\_provor\_2\_nc\_rt and decode\_provor\_2\_nc\_dm.

I will only speak in this note about the 2 PI decoders.

If you want to deploy the 2 DAC decoders at INCOIS, you will need additional perl code (developed by Laure Fontaine at Coriolis). If so, I can explain you later how we have deployed the decoders in the Coriolis real time data flux.

# Description of the decoder package contents

In the decoder zipped archive (*decPrv\_20150409\_001a\_for\_incois.7z*) you will find the following items.

In *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats*, everything that concerns float configurations:

* *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\json\_float\_info*, for DAC decoders only,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\prvFloatInfo\\_provor\_floats\_information\_co.txt*, main float information file (resulting of a copy of the 13 first columns of the *\_provor\_floats\_information\_co\_rt.xls* file) see below (in §2.1) for a short description of these columns,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\json\_float\_meta\_\**, JSON meta-data float files (one directory for each float transmission type: ‘argos’, ‘ir\_rudics\_remocean’ for French Remocean rudics floats, ‘ir\_sbd\_remocean’ for Indian FLBB Remocean floats and ‘ir\_sbd’ for common Iridium floats). See below (in §2.2) the description of their contents,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\lists*, list of floats managed at Coriolis,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\float\_dm\_buffer\_lists*, pre-defined buffer contents used by Remocean floats decoded in 'delayed mode'; see below (in §7) explanations on this mode,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\DB\_export*, Coriolis data base exports used to generate the JSON meta-data files

In *decPrv\_20150409\_001a\_for\_incois\decPrv\_doc*, the decoder, tools and float documentation. Unfortunately, most of it is in French, however in *decPrv\_20150409\_001a\_for\_incois\decPrv\_doc\MUT\_floats*, you can find the NKE manuals of the float decoded by this decoder.

In *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft*, the decoder itself:

* *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\soft*, the Matlab code,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\\_VersionsLogiciellesProvor\_20150407.xlsx*, the Coriolis version of each float version and the associated decoder Id,
* *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\configParamNames* and *techParamNames*, the JSON files (one per decoder Id) used to link CONFIG (resp. TECH) labels outside the decoder. In these files "CONF\_PARAM\_DEC\_ID" (resp. "TECH\_PARAM\_DEC\_ID") should not be modified because it is the Id used by the decoder to affect "CONF\_PARAM\_NAME" (resp. "TECH\_PARAM\_NAME") label to the concerned information in the output NetCDF files.

The two following directories are not really part of the decoder itself but you must have it on your computer to use it correctly.

In *decPrv\_20150409\_001a\_for\_incois\ressources*, you can find the files for the free ETOPO2 and SRTM30+ bathymetric atlases.

In *decPrv\_20150409\_001a\_for\_incois\m\_map* you can find the ‘m\_map’ code used for the geographical drawings (I've corrected the 1.4e version, grep 'RNU' in this code if you want to know my corrections, from the *m\_map1.4e\_ori.7z* archive downloaded in the web).

## Note on the *\_provor\_floats\_information\_co\_rt.xls* file

Most of the columns are obvious to understand, so let's only speak about:

* Column G (DELAI parameter): used to store the DELAI parameter useful for old float versions (none in this version of the decoder, thus = -1), this column is also used to store the firmware version of the Remocean floats (the time to update the configuration for some specific messages depends on float firmware version !),
* Column K (Day of the first descent): this is the reference day of the float (day #0 of its internal calendar). It is the day the float first crosses the FIRST\_GAP isobar (and then store the corresponding time as DST). It is generally the launch date or launch date + 1 day (depending on the launch hour),
* Column L (End decoding date): in this version this information only concerns common Iridium floats. It is the date (stored in the mail file name) of the last Iridium mail file to be processed for this float. This date is used to separate input raw data for floats which shared the same IMEI or for floats which have been recovered while still emitting,
* Column M (DM flag): only concerns Remocean floats, set to 1 for floats decoded in DM mode; see below (in §7) explanations on this mode.

## Note on the meta-data JSON files

In a meta-data JSON file (one file per float) you can find the information used to create the meta.nc file for this float.

Most of the items are not used by the decoder, i.e. it just reads the information in the JSON file and copies it in the meta.nc file. However, some items are used by the decoder:

* "CONFIG\_PARAMETER\_NAME"/"CONFIG\_PARAMETER\_VALUE": contain the configuration of the float at launch:
  + For floats which transmit their configuration it is the base configuration,
  + For Argos floats it is the configuration for all the float life (for NKE Argos floats we have at least 2 missions; 3 for multi-mission floats),

Then, you can easily understand that this item should be carefully filled.

* "CALIBRATION\_COEFFICIENT": contains the coefficients used by the decoder to compute derived parameters,
* "CALIB\_RT\_\*": contain all information about the RT adjustments that should be applied by the decoder,
* "SENSOR\_MOUNTED\_ON\_FLOAT": for bio floats, contains the list of sensors mounted on the float (the only way to know what is expected to be received from the float).

# Matlab version to use

To use the decoder, you need at least a Matlab version with a native NetCDF library (after R2006b).

I use MATLAB R2014a (but only because I have not updated yet the drawing programs for the Matlab R2014b release). If you don't use the drawing programs you can use the MATLAB R2014b or MATLAB R2015a versions.

# Decoder deployment

To install the decoder you should:

1. Unzip the provided archive,
2. Add the *decPrv\_20150409\_001a\_for\_incois\m\_map\m\_map1.4e* directory in your Matlab path,
3. Add the *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\soft* directory and all its sub-directories in your Matlab path.

# Decoder configuration

To configure the decoder you have to update, according to your own (Linux or PC) platform, the decoder configuration file: *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\soft\\_prv\_decoder.conf*.

The items to update are:

* FLOAT\_LIST\_FILE\_NAME: default list of floats to process,
* EXPECTED\_CYCLE\_LIST: for Argos floats, we can choose the cycles to decode. However, as TRAJ data (and profile location and time determination) may use information from previous cycles, I recommend to always set (EXPECTED\_CYCLE\_LIST = 9999 or EXPECTED\_CYCLE\_LIST = [~]) i.e. to always process existing cycles from the beginning (to process all of them or to process the first *N* of them),
* FLOAT\_TRANSMISSION\_TYPE: set to 1 for Argos, 2 for Remocean French floats, 3 for Iridium common floats and 4 for INCOIS FLBB floats,
* FLOAT\_INFORMATION\_FILE\_NAME: set to *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\prvFloatInfo\\_provor\_floats\_information\_co.txt* file,
* HEX\_ARGOS\_FILE\_FORMAT: set to 1,
* DIR\_INPUT\_HEX\_ARGOS\_FILE\_FORMAT\_1: set to the top directory of the Argos HEX files (I will send you the data for this directory),
* DIR\_INPUT\_HEX\_ARGOS\_FILE\_FORMAT\_2: unused if HEX\_ARGOS\_FILE\_FORMAT = 1,
* HEX\_ARGOS\_DATA\_DIRECTORY\_STRUCTURE: set to 3,
* DIR\_INPUT\_RSYNC\_DATA: set to the top directory of the Iridium mail files repository (I will send you the data for these directory),
* DIR\_INPUT\_RSYNC\_LOG: used by DAC decoders only,
* DIR\_INPUT\_JSON\_TECH\_LABEL\_FILE: set to *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\techParamNames* directory,
* DIR\_INPUT\_JSON\_CONF\_LABEL\_FILE: set to *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\configParamNames* directory,
* DIR\_INPUT\_JSON\_FLOAT\_META\_DATA\_FILE: set to *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\json\_float\_meta\_\** directory that matches the FLOAT\_TRANSMISSION\_TYPE information,
* DIR\_INPUT\_DM\_BUFFER\_LIST: set to *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\float\_dm\_buffer\_lists* directory,
* IRIDIUM\_DATA\_DIRECTORY: set to the directory chosen to store the float Iridium mail files (create your own empty directory),
* DIR\_OUTPUT\_LOG\_FILE: set to the directory chosen to store the decoder log files (create your own empty directory),
* DIR\_OUTPUT\_CSV\_FILE: set to the directory chosen to store the decoder csv files (create your own empty directory),
* DIR\_OUTPUT\_XML\_FILE: used by DAC decoders only,
* DIR\_OUTPUT\_NETCDF\_FILE: set to the directory chosen to store the Argo output NetCDF files (create your own empty directory),
* GENERATE\_NC\_\*: use these flags to choose the NetCDF Argo files that will be generated by the decode\_provor\_2\_nc program (1 if you want to generate it, 0 otherwise) (PROF and TRAJ files are linked so always generate both files),
* APPLY\_RTQC: set to 1 if you want to apply RTQC after the decoding, 0 otherwise,
* TESTXXX\_\*: if APPLY\_RTQC = 1, use these flags to choose the tests you want to perform in the RTQC (1 to perform the test, 0 otherwise),
* TEST004\_ETOPO2\_FILE: if test #4 has to be performed in RTQC, set this item to the *decPrv\_20150409\_001a\_for\_incois\ressources\ETOPO2\ETOPO2v2g\_i2\_MSB.bin* file,
* TEST015\_GREY\_LIST\_FILE: if test #15 has to be performed in RTQC, download the most recent greylist on the GDAC FTP server and set this item to this file,
* ADD\_THREE\_MINUTES: the Argos times provided by NKE floats have a 6 minutes resolution, if you set this item to 1, 3 minutes will be added to some of these times (we usually set this item to 0, i.e. we don't apply this correction),
* CYCLE\_2\_SBD\_DATA\_FILE\_NAME: not used in this decoder version (for ANDRO decoder only),
* FLOAT\_WITH\_DAMPING\_DISK\_FILE\_NAME: used for old FSI NKE floats (for ANDRO only) not managed by this Coriolis decoder version.

Notes:

* The others *\_prv\_decoder\_\*.conf* files, provided in *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\soft*, are my used configuration files (when I use the decoder to process a given FLOAT\_TRANSMISSION\_TYPE set of floats, I move the appropriate *\_prv\_decoder\_\*.conf* file to *\_prv\_decoder.conf* before processing the floats),
* The \_*prv\_decoder\_conf.json* file is the configuration file for the DAC decoders.

# How to use the decoder

To use the decoder you can type decode\_provor\_2\_csv or decode\_provor\_2\_nc in the Matlab command window.

Doing that, all the float of the FLOAT\_LIST\_FILE\_NAME file will be decoded in a unique csv file (decode\_provor\_2\_csv) or in V3.1 NetCDF Argo files (decode\_provor\_2\_nc).

You can also choose to decode only few floats by providing their WMO numbers as decoder parameters: decode\_provor\_2\_csv(2902077) or decode\_provor\_2\_nc(2902089, 2902118, 2902086) for example.

# Note on the Delayed Mode decoding mode

This mode only concern Remocean floats.

Sometimes, these floats don't transmit all expected messages (or transmit erroneous information), in that case, the real time buffer cannot be processed at the correct time (it is finally processed after a TIME OUT) and the result can be erroneous (particularly when the float is transmitting more than one sub-cycles, i.e. the data collected during more than one ascent).

We haven't succeed in solving this case in real time, thus we decided to create a Delayed Mode decoding mode.

Once a float died (or has been retrieved), the set of received float data packets can be definitely grouped in buffers by the user.

The tool split\_remocean\_sbd\_mail\_files (for INCOIS FLBB floats) or split\_remocean\_rudics\_sbd\_files (for french Remocean floats) can be used to split the received SBD files in mono-packet files (one file per float packet) and to create a csv file containing the first version of the buffers. The user can then checks this proposal and modify it as needed.

As an example, see *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\float\_dm\_buffer\_lists\xls\split\_remocean\_sbd\_mail\_files\_2902089\_20150216T085137.xlsx* file.

In the 'Rank' column you have the buffer number, in the 'File' column you have the mono-packet file name.

All the files with the same rank are processed together (in the same buffer), the buffer are processed in the order of the rank numbers.

Files with rank = -1 are ignored.

This csv file should be carefully studied and the rank numbers modified to modify, if needed, the buffers contents (my modifications are in yellow in these Excel files). Of course this work needs a perfect knowledge of the theoretical transmission strategy of each float version.

Once this is done, the resulting Excel file is used to create the *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\float\_dm\_buffer\_lists\2902089\_buffers.txt* file and the DM mode flag of this float #12902089 is set to 1 in the *decPrv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\prvFloatInfo\\_provor\_floats\_information\_co.txt* configuration file.

When the decoder detects a float to be processed in DM:

1. It first checks if the *IRIDIUM\_DATA\_DIRECTORY/IMEI\_WMO/archive\_dm* directory is empty (see in §9, for explanation on this directory).

If so (first decoding of the float in DM), it splits the files of the *IRIDIUM\_DATA\_DIRECTORY/IMEI\_WMO/archive* directory in mono-packet files and store these files in the *IRIDIUM\_DATA\_DIRECTORY/IMEI\_WMO/archive\_dm* directory.

1. It then decodes the float according to the *Prv\_20150409\_001a\_for\_incois\decPrv\_config\_floats\float\_dm\_buffer\_lists\wmo\_buffers.txt* strategy.

In this mode, the files stay in the *IRIDIUM\_DATA\_DIRECTORY/IMEI\_WMO/archive\_dm* directory and as the buffers are already defined the decoding of the float is more efficient.

# Additional tools

Some useful tools are provided with the decoder in the directory *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\soft\util*.

## Tools configuration

To configure the access to bathymetric atlases used by the drawing tools, you have:

* To update the line #31 of the *decPrv\_20150409\_001a\_for\_incois\m\_map\m\_map1.4\m\_etopo2.m* file with the directory *decPrv\_20150409\_001a\_for\_incois\ressources\ETOPO2*,
* To update the line #77 of the *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\soft\util\sub\get\_srtm\_data.m* file with the directory *decPrv\_20150409\_001a\_for\_incois\ressources\SRTM30+\data*.

To configure a tool you generally have to edit it:

* In the few first lines some items should be updated,
* Some tools use some information of the decoder configuration file (*\_prv\_decoder.conf*), through the function get\_config\_dec\_prv.

For example, see the tool check\_argos\_cycle\_files:

To configure this tool you have to update lines #25, #28, #31(but only if you want to use a MIN\_NON\_TRANS\_DURATION\_FOR\_GHOST different from our) and #41.

You can also see in line #50 and #51 that FLOAT\_LIST\_FILE\_NAME and FLOAT\_INFORMATION\_FILE\_NAME are retrieved from the *\_prv\_decoder.conf* configuration file (at line #56).

FLOAT\_INFORMATION\_FILE\_NAME is unique but you must update FLOAT\_LIST\_FILE\_NAME in the *\_prv\_decoder.conf* file to choose the floats to process if you use this tool without argument (you can also use it with WMO float numbers as parameters of the tool).

## Main tools

I will not comment all the tools but only the most important (useful) ones.

### clean\_spool\_and\_buff

When the decoder crashes while decoding Iridium floats, you can be in an unstable situation where files remain in the 'spool' or 'buff' directories of the float. If so, use this tool to delete SBD files and move back mail files to the 'archive' directory. Be careful, some files can be locked by the Matlab process and cannot be moved or deleted by the tool. If so, unlock the files (I used the Unlocker (http://unlocker.softonic.fr/) tool in my window platform) and run clean\_spool\_and\_buff again.

### copy\_iridium\_mail\_files and copy\_remocean\_sbd\_files

Tools used to make a copy of Iridium or Rudics files from their repository to the directory associated to each float (copy\_iridium\_mail\_files is used for common Iridium floats and FLBB INCOIS foats, copy\_remocean\_sbd\_files for French Remocean Rudics floats).

### generate\_json\_decoder\_config\_labels and generate\_json\_decoder\_tech\_labels

Tools used to generate JSON TECH and CONF reference files for each decoder Id.

My processing mode is:

1. Study each new format and create an Excel file with 'what to store and where' answers (see Excel files of the *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\configParamNames* and *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\techParamNames* directories).
2. From each Excel files (using Excel filters), create a csv file for the main decoder Id (some float versions can share the same set of TECH or CONF information).
3. Generate one JSON TECH and CONF file for each decoder ID using these 2 tools.

### generate\_json\_float\_meta\_argos, generate\_json\_float\_meta\_ir\_sbd, generate\_json\_float\_meta\_remocean and generate\_json\_float\_meta\_remocean\_flbb

Tools used to generate JSON meta-data files from data base export (and for Remocean floats, from files containing a dump of the float configuration at launch).

### get\_meta\_data\_from\_data\_base and get\_meta\_data\_from\_nc

Tools used to generate parts of the *\_provor\_floats\_information\_co\_rt.xls* file from a Coriolis data base export or from already existing floats (i.e. from their meta.nc file)

### nc\_add\_rtqc\_flags

The RTQC flags can be added by the decoder just after the decoding (if APPLY\_RTQC = 1).

You can also apply RTQC tests to existing V3.1 NetCDF Argo files using this tool (the tests to be performed should be set to 1 in the testToPerformList list on line #69).

### nc\_check\_file\_format

This tool is used to check that the NetCDF decoded files are compliant to the Argo V3.1 format with the GDAC Argo checker (the last version of the checker should be first downloaded at http://usgodae.org/pub/outgoing/argo/etc/FileChecker/).

### nc\_meta\_2\_csv, nc\_prof\_2\_csv, nc\_prof\_adj\_2\_csv, nc\_tech\_2\_csv and nc\_traj\_2\_csv

Tools used to convert Argo NetCDF V3.1 files to csv files (easy to study using Excel filters for example).

### nc\_trace\_disp, nc\_trace\_param and nc\_trace\_times

These are the 3 main drawing tools.

Start a tool and make the drawing window active. Then press the 'h' key of the keyboard. The help on the tool will appear in the Matlab command window.

# First step procedure

Please install and configure the decoder as explained above.

Then unzip the data archive (*raw\_data\_incois\_201504113.7z*) and:

* Set DIR\_INPUT\_HEX\_ARGOS\_FILE\_FORMAT\_1 to the *raw\_data\_incois\_201504113\archive\_cycle\_incois\_20150409* directory,
* Set DIR\_INPUT\_RSYNC\_DATA to the *raw\_data\_incois\_201504113\iridium\_mail\_repository\_20150331* directory.

You can then use the decoder to decode the Argos floats.

Before decoding Iridium floats you have to copy the Iridium mail files from the repository (DIR\_INPUT\_RSYNC\_DATA) to the float Iridium directories (IRIDIUM\_DATA\_DIRECTORY).

This can be done using the tool copy\_iridium\_mail\_files.

After that, you should have, in the IRIDIUM\_DATA\_DIRECTORY, one directory per float (called *IMEI*\_*WMO*).

In each *IMEI*\_*WMO* directory, the mail files are stored in an *archive* sub-directory.

You can then use the decoder to decode the Iridium floats.

During the first use, the decoder will create, in each *IMEI*\_*WMO* directory:

* A *spool*, a *buffer* and a *rsync\_log\_processed* sub-directory for common Iridium floats,
* A *spool*, a *buffer*, a *rsync\_log\_processed*, a *archive\_dm* and a *mat* sub-directory for Remocean Iridium floats.

# Second step procedure

In this second step we will explain how to:

* Process incoming float data to be ready for decoding,
* Declare new floats to be decoded by existing decoder Ids.

## Processing of new incoming float data

The processing steps for incoming data depend on float transmission type.

### For Argos floats

The Argos Hex data, coming from CLS by e-mails or CD-ROM, need to be prepared to be used by the decoder. This process is done in the following steps.

##### Step #0: copy all received Argos data in a unique directory

All received data (e-mail files, CD-ROM contents, archived data (of the DIR\_INPUT\_HEX\_ARGOS\_FILE\_FORMAT\_1 directory) should be first copied in a unique directory.

All these files should be at the same level of the directory (no sub-directories are allowed). Don’t worry about duplicated data (step #2 will delete the duplicates).

You can use the tool *copy\_argos\_files\_in\_archive\_cycle* to duplicate the Argos data of the DIR\_INPUT\_HEX\_ARGOS\_FILE\_FORMAT\_1 directory for a given list of floats.

##### Step #1: split the data

In this step we split the data by Argos Id number and by satellite pass.

The input directory should be the one created on step #0. The output directory will contain one directory for each Argos Id and within each of these sub-directories, one file per satellite pass for the concerned Argos Id.

The tool split\_argos\_cycle is used for step #1.

##### Step #2: delete duplicated data

In this step we check the satellite pass files generated from step #1 and delete duplicated data.

The tool delete\_double\_argos\_split is used for step #2.

##### Step #3: create Argos cycle files

In this step we create Argos cycle file (containing all the data transmitted by the float after each cycle) from satellite pass files obtained in step #2.

Each Argos cycle file contains the data of the satellite pass files concatenated and chronologically sorted. A new Argos cycle file is created each time we find a 18 hours delay without any data transmission.

The tool create\_argos\_cycle\_files is used for step #3.

##### Step #4: name Argos cycle files

In this step we compute the cycle number associated to each Argos cycle file and create the final name of the file.

The name of the Argos cycle file should be:

*ArgosId\_YYYY-MM-DD-hh-mm-ss\_WMO\_CyNum.txt*, where

* *ArgosId*: is the float PTT number (on 6 digits),
* *YYYY-MM-DD-hh-mm-ss*: is the date of the earlier float message of the file,
* *WMO*: is the float WMO number,
* *CyNum*: is the cycle number.

Note also that

* *WMO* can be equal to ‘WWWWWWW’ if the ArgosId to WMO link is unknown at the time of reception of the data,
* *CyNum* can be equal to:
  + ‘EEE’: empty file (not at least one float message),
  + ‘WWW’: ArgosId to WMO link is unknown at the time of reception of the data,
  + ‘MMM’: meta-data unavailable to compute cycle number,
  + ‘TTT’: test data (dated before float launch date),
  + ‘GGG’: ghost messages,
  + ‘UUU’: cycle number value (manually) disabled by the user.

Only ‘identified’ files (i.e. with valid *WMO* and *CyNum* numbers) are processed by the decoder.

The tool move\_and\_rename\_argos\_files is used for step #4. It uses the contents of the json meta-data files.

The output files can be used for decoding (they should replace the existing ones in the DIR\_INPUT\_HEX\_ARGOS\_FILE\_FORMAT\_1 directory).

##### Step #5: check the processed output files

Use the tool check\_argos\_cycle\_files to check the work done and the file contents (particularly in columns ‘D’ to ‘G’ of the CSV file generated by this tool).

**Note also that the tool process\_argos\_data can be used to process step #1 to #4.**

### For Iridium SBD floats

The incoming Iridium mail files should be renamed to be used by the decoder. This can be done with the tool move\_and\_rename\_iridium\_sbd\_mail\_files.

Each processed file will be renamed:

*co\_YYYYMMDDThhmmss\_IMEI\_MOMSN\_MTMSN\_PID.txt*, where

* *YYYYMMDDThhmmss*: is the date of the session,
* *IMEI*: is the float IMEI number,
* *MOMSN*, *MTMSN* : are the MOMSN and MTMSN numbers of the transmission,
* *PID*: is the PID of the process that collected the mail (unused).

The newly named files can then be move to the repository (DIR\_INPUT\_RSYNC\_DATA directory).

## Declaration of new floats

The declaration of a new float in the decoder is done in 3 steps:

1. Identify the Coriolis float version and the associated decoder Id,
2. Update the *\_provor\_floats\_information\_co\_rt.xls* file,
3. Create the json meta-data file of the new float.

### Find the decoder Id of the float

The Coriolis float version can be deduced from the float User’s Manual (compare it to the Manuals of the float versions managed by the decoder (provided in *decPrv\_20150409\_001a\_for\_incois\decPrv\_doc\MUT\_floats*) the associated Coriolis float versions are at the beginning of the file name) and then from the *decPrv\_20150409\_001a\_for\_incois\decPrv\_soft\config\\_VersionsLogiciellesProvor\_20150407.xlsx* file. This file is also used to find the decoder Id (column ‘C’) from the Coriolis version (column ‘B’).

### Update the \_provor\_floats\_information\_co\_rt.xls file

Add a new entry for each new float in the *\_provor\_floats\_information\_co\_rt.xls* file.

At Coriolis, we use the tool get\_meta\_data\_from\_data\_base which retrieve the needed information from a Coriolis data base export.

### Create the json meta-data file of the float

This is the most difficult step of the float declaration because the needed file should contain information directly coming or computed from float meta-data.

These meta-data storage depends on each DAC infrastructure. At Coriolis, we use an export of the data base (and additional files for Remocean floats).

**For Argos floats** we use the tool generate\_json\_float\_meta\_argos which uses only the data base export.

**For Iridium floats** we use the tool generate\_json\_float\_meta\_ir\_sbd which uses only the data base export.

**For Iridium FLBB floats** we use the tool generate\_json\_float\_meta\_remocean\_flbb which uses:

1. The data base export,
2. The calibration coefficients transmitted by the float (*calib\_coef.xlsx* file),
3. The configuration of the float dumped in a file just before launch (*ConfigAtLaunch* directory).

Note that, to update the *calib\_coef.xlsx* (and *calib\_coef.txt*) files you first need to do a previous decoding (with decode\_provor\_2\_csv) and to get the *ScaleFactChloroA*, *DarkCountChloroA*, *ScaleFactBackscatter700*, *DarkCountBackscatter700* and *KhiCoefBackscatter* coefficient from the CSV output file.

# Third step procedure

In this last step we will explain how the core decoder can be deployed in a Real Time data flux (with the 2 DAC decoders: decode\_provor\_2\_nc\_rt and decode\_provor\_2\_nc\_dm).

To be continued (if needed by INCOIS) …

# Miscellaneous information

Note that, to correctly use the loadjson.m Matlab function on a Linux platform you should set:

setenv LANG C

in the Linux user configuration.