VOParis Radio-JOVE study report

February 23rd, 2014
Version 1.2

Picture of Radio-JOVE antennas © Radio-JOVE/GSFC

Authors

B. Cecconi (VOPDC-LESIA)
P. Le Sidaner (VOPDC-DIO)
R. Flagg, C. Higgins, J. Sky, J. Thieman, D. Typinski (Radio-JOVE team)

Acronyms

CDF	Common Data Format
DIO	Direction Informatique de l'Observatoire de Paris
DaCHS	Data Center Helper Suite
EPN	Europlanet FP7 project
EPN-TAP	Europlanet Table Access Protocol
GSFC	NASA Goddard Space Flight Center
HELIO	Heliophysics Intergrated Observatory
ISTP	International Solar Terrestrial Program
IVOA	International Virtual Observatory Alliance
LESIA	Laboratoire d'Etudes Spatiales et d'Instrumentation en Astrophysique
PDS	NASA Planetary Data System
SPDF	NASA Space Physics Data Facility
TAP	IVOA Table Access Protocol
TOPCAT	Tool for OPerations and CAtalogs and Tables
UCD	IVOA Unified Content Descriptor
VOPDC	Virtual Observatory Paris Data Centre

Table of contents

Context

Radio-JOVE is an educational and public outreach project developed in the USA that introduces low frequency radioastronomy concepts to students and teachers, but also the amateur radio community as well as the general public. The participants are building their own radio telescope, using a kit sold by the Radio JOVE team. This instrument can observe the sky at frequencies around 20 and 30 MHz. The users can share their observations on an archive web site, and on a mailing list.

Radio-JOVE web site: http://radiojove.gsfc.nasa.gov
Radio-JOVE data Archive : http://radiojove.org/cgi-bin/calendar/calendar.cgi

Introduction

We are proposing to set up a prototype interoperable service dedicated to the distribution of Radio-JOVE data in the Virtual Observatory (VO). This service shall:
- store the data sent by the users in a standard format,
- allow a data selection by the science team before putting the data online,
- share the data using VO standards, specifically EPN-TAP, but also those linked to the SPASE (Space Physics standards), or the HELIO project, for solar radio observations.

During this project, we test how amateur data can be shared to the scientific community, using the VO. We also want to consolidate the use of the EPN-TAP protocol, testing it with a new type a dataset, and a new type of data provider (distributed amateur community).

Scientific interest

In the Radio-JOVE frequency band, there are 2 main radio sources, wich canbe observed: the Sun and Jupiter. Other radio sources also contribute: the Galactic Background radiation, the radio counterpart of terrestrial lightnings, and local radio interferences. There are 2 large instruments in the world that routinely observe Jupiter in this frequency range: the Nançay Decameter Array in France, and the Iitate radio observatory in Japan. Other instruments can also observe Jupiter during dedicated observation campaigns, such as the UTR2 array at Kharkov in Ukraine, the LOFAR telescope in Europe, or the new LWA telescope in New Mexico, USA. These instruments do not provide a full time survey of Jovian radio emissions. Extending the temporal coverage is scientifically interesting, in particular, for the upcoming space missions that are going to explore the Jovian system (JUNO and JUICE), in which the LESIA at Observatoire de Paris in involved.
The Jovian radio emissions are appearing as “arc-shaped” structures in the time-frequency plane. This shape indicates how the observer is “beamed” by the radio source, which has a very anisotropic beaming pattern and is rotating around Jupiter, following its off-axis magnetic field. The study of these radio arcs is a powerful tool that can remotely probe the plasma is the radio emission regions. Their observed temporal variability is correlated to their intrinsic temporal variability, and the spatial variability of the emission medium. The short term variability requires a series of a radio observatories spread other the Earth, with simultaneous observations. The Radio-JOVE observer’s network is the perfect candidate for such studies. The same kind of study could be done with solar radio emissions.

Radio-JOVE data distribution

The Radio-JOVE kit is sold with the “Radio Sky Pipe” software, which drives the Radio-JOVE instrument and proposes to: save data into files or stream data to connected users. A limited series of metadata is attached to each observer. The Radio-JOVE data are distributed both ways: either using emails on the RadioJOVE-data mailing list, or on the Radio-JOVE online archive. The data format is most usually a screenshot (PNG or GIF files), as well as WAV files. A few events are shared using the native Radio Sky Pipe format.

At the occasion of the annual meeting of SARA (Society of Amateur Radio Astronomer), on July 2014, in Green Bank, USA, we have contacted the Radio-JOVE team and the Radio Sky Pipe developer. It has been decided to study the possibility of using CDF (Common Data Format) files for data distribution. The choice of the CDF as an standard format is rather natural:
- It is developed and maintained by NASA for Space Physics dataset.
- It is used as an archive format for Space Physics data at NASA (including space borne radio observation).
- It is now accepted as an archive format by NASA for planetary data.
- It has a recommended configuration and metadata description (ISTP standard and PDS guidelines).
- It has been recently added as an input format in TOPCAT.

The Radio Sky Pipe will study how to implement CDF output in his software, using the software library distributed by NASA/GSFC. We have studied the CDF file formatting, for the various Radio-JOVE data products.

File Format study

In the initial proposal, the first step to be accomplished is the definition of the file format, including the metadata standard and content.

CDF files for Radio-JOVE

The CDF standard has been adopted and the associated metadata have been selected, using existing guidelines (ISTP and PDS). Four CDF templates have been produced for the following setups:
- Single channel time series (STS): this is the basic setup, with 1 antenna and the standard single frequency Radio-JOVE receiver at ~20 MHz.
- Dual channel time series (DTS): for users that have 2 antennas (in crossed configuration, usually), with 2 Radio-JOVE single frequency receiver.
- Single channel dynamic spectrum (SDS): for users have an advanced radio receiver of a wide band spectral window (20 to 30 MHz), connected to 1 antenna.
- Dual channel dynamic spectrum (DDS): for users have an advanced radio receiver of a wide band spectral window (20 to 30 MHz), with 2 channels that are each connected to an antenna.

The CDF templates have been developed at LESIA using the CDF skeleton language. This is an ASCII file that can be ingested into the CDF library to build a master CDF, which is then filled with the metadata and the data. We have used our experience on CDF data file provision to NASA/PDS. The files produced are then compliant with NASA/PDS standards, and thus could be archived in that repository.

The CDF skeleton files for each file type is provided in Appendix A.

Radio-JOVE metadata

The CDF file contains 2 types of metadata (called attributes):
- the global attributes that refers to the data file,
- the variable attributes that refers to each variable within the file.

Global Attributes

We have included ISTP global attributes:

```
project, discipline, data_type, descriptor, data_version, instrument_type,
logical_file_id, logical_source, logical_source_description, file_naming_convention,
mission_group, pi_name, pi_affiliation, source_name, text, generated_by, generation_date,
link_text, link_title, http_link, mods, parents, rules_of_use, skeleton_version,
software_version, time_resolution, acknowledgment, adid_ref, validate
```

These keywords are defined in the ISTP guidelines available online. We have also added the relevant PDS global attributes:

Table 1. *File size at the various steps of the processing.*

original file name	*data type*	*original file type*	*original file size*	*temporary file type*	*temporary file size*	*final CDF file size*	*zipped CDF file size*
AJ4CO-FS-200B-141212100000.sps	SDS	natif SPS	2.1 MB	N/A	N/A	4.2 MB	1.6 MB
IO-B03-04-14-0254ut.mp3	DTS	MP3	3.6 MB	WAV	38.4 MB	230.3 MB	102.0 MB
raf2_25sep14.wav	SDS	WAV	20.2 MB	N/A	N/A	121.2 MB	41.7 MB
raf25sep14.wav	SDS	WAV	3.7 MB	N/A	N/A	22.2 MB	9.5 MB
raf26sep14.wav	SDS	WAV	9.3 MB	N/A	N/A	55.9 MB	24.2 MB
solar1395328196.mp3	SDS	MP3	3.6 MB	WAV	38.4 MB	230.3 MB	101.8 MB
solar1404999549.mp3	STS	MP3	2.5 MB	WAV	26.9 MB	268.8 MB	91.8 MB
solar1405001174-sol.mp3	STS	MP3	480 kB	WAV	5.1 MB	51.2 MB	17.8 MB
solar1405004070-sol.mp3	STS	MP3	719 kB	WAV	7.7 MB	76.7 MB	26.6 MB

```
observation_start_time, observation_stop_time, observation_target, observation_type
```

Relevant EPN-TAP keywords have also been added:

```
dataproduct_type, target_class, target_region, target_element, time_min, time_max,
time_sampling_min, time_sampling_max, spectral_range_min, spectral_range_max,
instrument_host_name, instrument_name, measurement_type, access_format
```

We have also defined new metadata that are specific to Radio-JOVE:

```
observer_name, observatory_location, observatory_latitude, observatory_longitude,
original_filename, n_channel
```

The definition of this keywords are the following:
- **observer_name**: Name of the observer, school or team providing the data content.
- **observatory_location**: Named location of the observatory or school.
- **observatory_latitude**: Geographic latitude of the observatory of school.
- **observatory_longitude**: Geographic longitude of the observatory of school.
- **original_filename**: Name of the original file name used to build the CDF file.
- **n_channel**: number of channel of the Radio-JOVE setup ('1' for STS and SDS files, '2' for DTS or DDS files)

An extra keyword has been identified recently and shall be added to the next release of the CDF skeletons:
- **dipole_orientation_angle**: Orientation angle of the dipole(s), in degrees, where 0° is parallel to the local meridian. Angles then range between 0° and 90°.

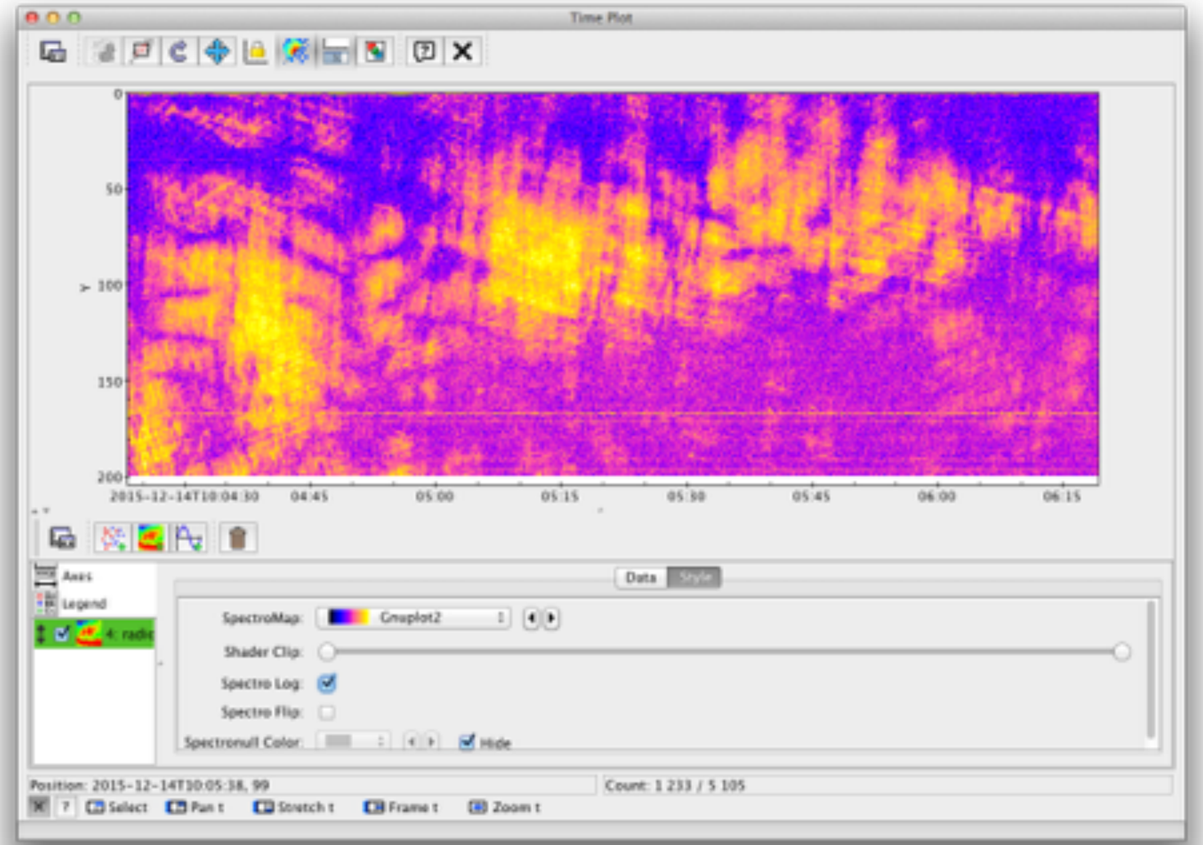
Variable Attributes

We have used the ISTP standard for variable attributes. We have simply added 1 keyword named UCD, which is providing the UCD (as defined by IVOA standard) for the given variable.

Transformation from Native or WAV format to CDF

The current data stored on the Radio-JOVE archive are either in native Radio Sky Pipe format, or in WAV and MP3 format, for single frequency setup. We have developed a prototype processing pipeline in IDL that can convert the available files into CDF files. This pipeline is using the CDAWlib (the IDL CDF library provided by the NASA/SPDF at GSFC). The IDL codes of this pipeline are presented in Appendix B.

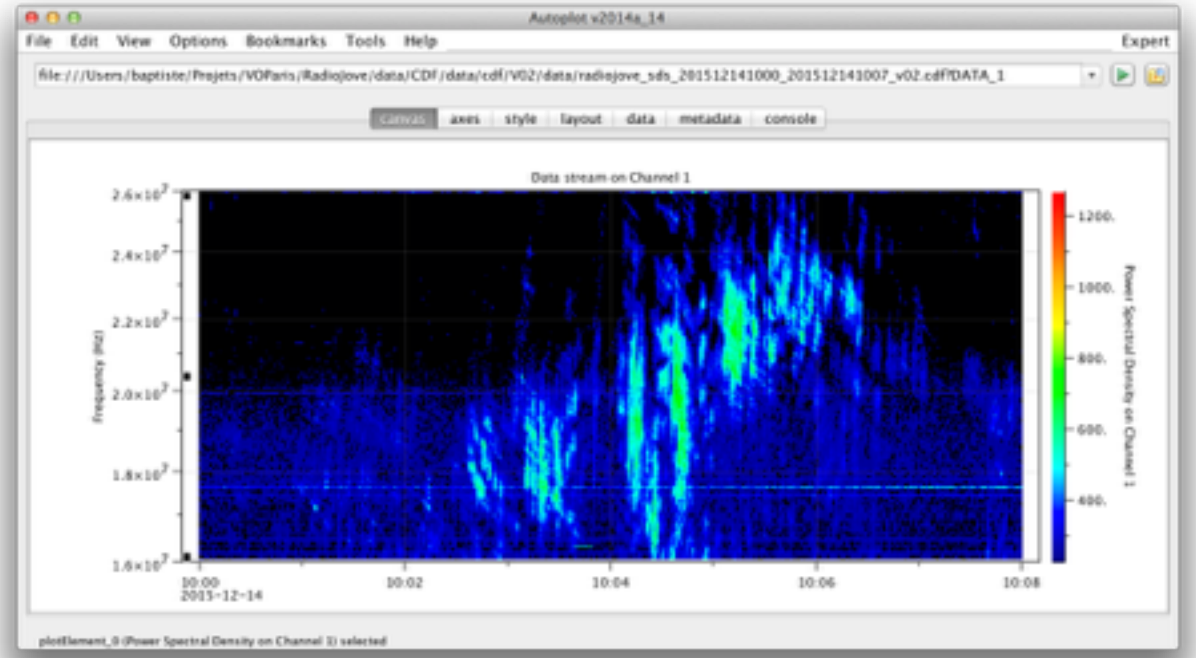
***Figure 1**. Radio-JOVE SDS data from D. Typinski displayed in TOPCAT after conversion in CDF.*



SpectroMap: Gnuplot2
Shader Clip:
Spectro Log: ☑
Spectro Flip: ☐
Spectronull Color: ☑ Hide

Position: 2015-12-14T10:05:38, 99
Count: 1 233 / 5 105

***Figure 2**. Radio-JOVE SDS data from D. Typinski displayed in Autoplot after conversion in CDF.*



file:///Users/baptiste/Projets/VOParis/Radiojove/data/CDF/data/cdf/V02/data/radiojove_sds_201512141000_201512141007_v02.cdf?DATA_1

plotElement_0 (Power Spectral Density on Channel 1) selected

Note that most of the users are providing there data in MP3 format. As it is a lossy compression algorithm, these data files are not really useable scientifically. However, for our tests, we have used these files after converting them into WAV format.

CDF File compression

The CDF standard allows internal file compression with several authorized compression algorithms. However the NASA/PDS archive guidelines are forbidding CDF file compression. We have thus presently decided to disable the CDF internal compression capabilities in order to keep the NASA/PDS compliance. The main problem with this choice is the size of the CDF files, as shown in Table 1. The decision on the Radio-JOVE CDF file compression may thus be revised in the future.

The file size increase is clear from the currently selected archive format (native, WAV and MP3) to the proposed CDF (uncompressed) or even an posteriori zipped CDF. This file size aspect shall not be disregarded as many Radio-JOVE contributors don't have high speed internet connections. We will study the possibility have raw data (in WAV format, for instance) upload with server side CDF encoding, to allow low-speed internet user to participate to the program.

Furthermore, it appears that some files are showing as dual channel receiver setups, but contain the same data in both channel. This is due to a "mono/stereo" mode misconfiguration on the users side during the conversion from the native file to the uploaded file.

Validation of the Radio-JOVE CDF files by visualisation in standard tools

The CDF files can be loaded in many tools and data processing languages. We show here spectrograms based on SDS data files provided by D. Typinski (Florida, USA), one of the main Radio-JOVE contributor. The data has been provided initially in the native Radio Sky Pipe output format. We have converted them into CDF files using the scripts presented in Appendix B.

TOPCAT

TOPCAT is a tool developed by M. Taylor at University of Bristol and is one of the main tool of the IVOA world to work with tables and catalogs. With the support of ESA, CDF input capabilities have been included. Using this tool, we have been able to load the CDF file and plot a dynamic spectrum, as shown on Figure 1.

Autoplot

Autoplot is a tool developed at University of Iowa and is dedicated to display CDF files, either user provided files from his local file system, or from a remote CDF database like the SPDF. Using this tool, we have been able to load the CDF file and plot a dynamic spectrum, as shown on Figure 2.

Server Study

The second task is to study and set up of a test server, including a data storage area, an interface for data upload, an interface for data validation, and several VO interfaces.

Server Prototyping with DaCHS

We have setup a prototype server based on the DaCHS framework, as recommended for EPN-TAP services. DaCHS is a full TAP server developed by M. Demleitner at University of Heidelberg. We recommend to use this framework, as it is well maintained, easy to deploy and to configure. Several deployment tutorials have been written by the Planetary Science team of VOParis in the course of the EPN FP7 project.

The new version of the DaCHS framework have been used (version 0.9.3), which includes CDF data collection input support. The translation of the Radio-JOVE CDF attributes into EPNTAP keywords in ongoing. The URL of the prototype server will be announced once a few data files are ingested.

The next step will be to setup the data upload, and data validation interfaces.

Conclusion

The CDF format as a data distribution format for Radio-JOVE data is well adapted, if we except the compression aspects. The foundation of the data and metadata structures has now been drafted. The next step is to implement the CDF generation support into the Radio Sky Pipe software. The server part of the study is not finished yet, and we will go on working on this direction to propose a prototype as soon as possible. The current plans for the longer term are to deliver the server to the Radio-JOVE team in the US, once it is working and they have found a sustainable hosting solution for the server. They are also looking for a data storage solution for the data files. With the NASA/PDS compliance, a possibility could be to submit the data files to that archive facility.

Finally, the collaboration with the Radio-JOVE team has been very fruitful, and we are very happy to continue this project that links a very involved amateur community with a scientific community. We hope that this collaboration will enable new studies on solar and planetary radio emissions.

Online Resources

CDF at NASA/GSFC: http://www.opendap.org
CDF archiving for PDS: http://ppi.pds.nasa.gov/doc/cdf/PDS4-Archiving-of-CDF-Files-v3.pdf
CDF ISTP guidelines: http://spdf.gsfc.nasa.gov/istp_guide/istp_guide.html

Radio-JOVE web site: http://radiojove.gsfc.nasa.gov
Radio-JOVE Archive site: http://radiojove.org/archive.html
Radio Sky Pipe software: http://www.radiosky.com/skypipeishere.html

VOParis Europlanet web resources: http://voparis-europlanetobspm.fr
TOPCAT: http://www.star.bris.ac.uk/~mbt/topcat/
AutoPlot: http://autoplot.org
UCD standard: http://www.ivoa.net/documents/latest/UCD.html

Appendix A

This appendix presents the 4 CDF skeleton files developed at LESIA for the Radio-JOVE project. CDF skeleton files are ASCII files used as CDF templates by CDF libraries. They can be built using the *ISTP CDF Skeleton Editor* provided by NASA/GSFC, or manually.

We have followed the ISTP and PDS guidelines to built these CDF skeleton files. Please refer to online ISTP and PDS documentation for more details.

Radio-JOVE STS CDF skeleton

The CDF skeleton code below shall be used for single channel time series (i.e., single frequency) Radio-JOVE setup. The CDF skeleton filename is:

    radiojove_sts_000000000000_000000000000_v02.skt

```
! Skeleton table for the "1418948900177" CDF.
! Generated: Thursday, 18-Dec-2014 16:44:07
! CDF created/modified by CDF V3.5.0
! Skeleton table created by CDF V3.5.0_2

#header

                       CDF NAME: 1418948900177
```

```
                  DATA ENCODING: PPC
                       MAJORITY: COLUMN
                         FORMAT: SINGLE

! Variables  G.Attributes  V.Attributes  Records  Dims  Sizes
! ---------  ------------  ------------  -------  ----  -----
     0/4          53            37         0/z      0
! CDF_COMPRESSION: None
! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
! CDF_CHECKSUM: MD5
! (Valid checksum: None, MD5)


#GLOBALattributes

! Attribute         Entry       Data
! Name              Number      Type       Value
! ---------         ------      ----       -----

  "Project"             1:    CDF_CHAR     { "RadioJOVE" } .

  "Discipline"          1:    CDF_CHAR     { "Space " -
                                             "Physics>Magnetospheric " -
                                             "Science" } .

  "Data_type"           1:    CDF_CHAR     { "RDR" } .

  "Descriptor"          1:    CDF_CHAR     { "STS" } .

  "Data_version"        1:    CDF_CHAR     { "02" } .

  "Instrument_type"     1:    CDF_CHAR     { "Waveform Receiver" } .

  "Logical_file_id"     1:    CDF_CHAR     { "radiojove_sts_00000000" -
                                             "0000_000000000000_v02" } .

  "Logical_source"      1:    CDF_CHAR     { "radiojove_sts" } .

  "Logical_source_description"
                        1:    CDF_CHAR     { "RadioJOVE Single Channel " -
                                             "Spectrograph" } .

  "File_naming_convention"
                        1:    CDF_CHAR     { "source_descriptor_yyyyMM" -
                                             "ddHHmm_yyyyMMddHHmm_V02" } .

  "Mission_group"       1:    CDF_CHAR     { "RadioJOVE" } .

  "PI_name"             1:    CDF_CHAR     { "RadioJOVE Project" } .

  "PI_affiliation"      1:    CDF_CHAR     { "GSFC" } .

  "Source_name"         1:    CDF_CHAR     { "RadioJOVE" } .

  "TEXT"                1:    CDF_CHAR     { "RadioJOVE Project data." } .

  "Generated_by"        1:    CDF_CHAR     { "SkyPipe" }
                        2:    CDF_CHAR     { "RadioJOVE" }
                        3:    CDF_CHAR     { "VOPDC" } .

  "Generation_date" .

  "LINK_TEXT"           1:    CDF_CHAR     { "Radio-SkyPipe Software " -
                                             "webpage" }
                        2:    CDF_CHAR     { "RadioJOVE webpage" }
                        3:    CDF_CHAR     { "VOParis Data Centre webpage" } .
```

```
"LINK_TITLE"            1:    CDF_CHAR     { "Radio-SkyPipe Software" }
                        2:    CDF_CHAR     { "RadioJOVE" }
                        3:    CDF_CHAR     { "VOParis Data Centre" } .

"HTTP_LINK"             1:    CDF_CHAR     { "http://www.radiosky.com/sk" -
                                             "ypipeishere.html" }
                        2:    CDF_CHAR     { "http://radiojove.gsfc.nasa" -
                                             ".gov" }
                        3:    CDF_CHAR     { "http://voparis-europlanet." -
                                             "obspm.fr" } .

"MODS" .

"Parents" .

"Rules_of_use" .

"Skeleton_version"
                        1:    CDF_CHAR     { "0.2" } .

"Software_version"
                        1:    CDF_CHAR     { "0.1" } .

"Time_resolution" .

"Acknowledgement" .

"ADID_ref" .

"Validate" .

"Observation_start_time"
                        1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_stop_time"
                        1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_target"
                        1:    CDF_CHAR     { "Saturn" } .

"Observation_type"
                        1:    CDF_CHAR     { "Waves" } .

"dataproduct_type"
                        1:    CDF_CHAR     { "TS>Time Series" } .

"target_class"          1:    CDF_CHAR     { "planet" } .

"target_region"         1:    CDF_CHAR     { " " } .

"target_element"        1:    CDF_CHAR     { " " } .

"time_min"              1:    CDF_REAL8    { 0.0 } .

"time_max"              1:    CDF_REAL8    { 0.0 } .

"time_sampling_min"
                        1:    CDF_REAL4    { 0.0 } .

"time_sampling_max"
                        1:    CDF_REAL4    { 0.0 } .

"spectral_range_min"
                        1:    CDF_REAL4    { 0.0 } .
```

```
  "spectral_range_max"
                      1:    CDF_REAL4     { 0.0 } .

  "instrument_host_name"
                      1:    CDF_CHAR      { " " } .

  "instrument_name"   1:    CDF_CHAR      { " " } .

  "measurement_type"
                      1:    CDF_CHAR      { "phys.flux;em.radio" } .

  "access_format"     1:    CDF_CHAR      { "cdf" } .

  "observer_name"     1:    CDF_CHAR      { " " } .

  "observatory_location"
                      1:    CDF_CHAR      { " " } .

  "observatory_latitude"
                      1:    CDF_REAL8     { 0.0 } .

  "observatory_longitude"
                      1:    CDF_REAL8     { 0.0 } .

  "original_filename"
                      1:    CDF_CHAR      { " " } .

  "n_channel"         1:    CDF_BYTE      { 0 } .


#VARIABLEattributes

  "CATDESC"
  "DEPEND_0"
  "DEPEND_1"
  "DEPEND_2"
  "DEPEND_3"
  "DICT_KEY"
  "DISPLAY_TYPE"
  "FIELDNAM"
  "FILLVAL"
  "FORMAT"
  "LABLAXIS"
  "LABL_PTR_1"
  "LABL_PTR_2"
  "LABL_PTR_3"
  "UNITS"
  "UNIT_PTR"
  "VALIDMIN"
  "VALIDMAX"
  "VAR_TYPE"
  "SCALETYP"
  "SCAL_PTR"
  "VAR_NOTES"
  "MONOTON"
  "LEAP_SECONDS_INCLUDED"
  "RESOLUTION"
  "Bin_location"
  "TIME_BASE"
  "TIME_SCALE"
  "REFERENCE_POSITION"
  "ABSOLUTE_ERROR"
  "RELATIVE_ERROR"
  "FORM_PTR"
  "DELTA_PLUS_VAR"
  "DELTA_MINUS_VAR"
  "SCALEMIN"
  "SCALEMAX"
  "UCD"


#variables

! No rVariables.


#zVariables

! Variable           Data      Number                 Record   Dimension
! Name               Type     Elements  Dims  Sizes  Variance  Variances
! --------           ----     --------  ----  -----  --------  ---------

  "Epoch"          CDF_EPOCH16     1       0              T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: 01-Jan-0000 00:00:00.000.000.000.000

  ! Attribute        Data
  ! Name             Type       Value
  ! --------         ----       -----

    "CATDESC"     CDF_CHAR     { "Default time" }
    "FIELDNAM"    CDF_CHAR     { "Epoch" }
    "FILLVAL"     CDF_EPOCH16   { 31-Dec-9999 23:59:59.999.999.999.999 }
    "LABLAXIS"    CDF_CHAR     { "Epoch" }
    "UNITS"       CDF_CHAR     { " " }
    "VALIDMIN"    CDF_EPOCH16   { 01-Jan-2000 00:00:00.000.000.000.000 }
    "VALIDMAX"    CDF_EPOCH16   { 01-Jan-2020 00:00:00.000.000.000.000 }
    "VAR_TYPE"    CDF_CHAR     { "support_data" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "MONOTON"     CDF_CHAR     { "INCREASE" }
    "TIME_BASE"   CDF_CHAR     { "J2000" }
    "TIME_SCALE"  CDF_CHAR     { "UTC" }
    "REFERENCE_POSITION"
                  CDF_CHAR     { "Earth" }
    "SCALEMIN"    CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "SCALEMAX"    CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "UCD"         CDF_CHAR     { "time.epoch" } .

  ! RV values were not requested.


! Variable           Data      Number                 Record   Dimension
! Name               Type     Elements  Dims  Sizes  Variance  Variances
! --------           ----     --------  ----  -----  --------  ---------

  "DATA_1"       CDF_FLOAT      1       0              T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE:   -1.0e+30

  ! Attribute        Data
  ! Name             Type       Value
  ! --------         ----       -----

    "CATDESC"     CDF_CHAR     { "Data stream on Channel 1" }
    "DEPEND_0"    CDF_CHAR     { "Epoch" }
    "DICT_KEY"    CDF_CHAR     { "data" }
    "DISPLAY_TYPE"
                  CDF_CHAR     { "time_series" }
    "FIELDNAM"    CDF_CHAR     { "DATA_1" }
    "FILLVAL"     CDF_FLOAT     { -1.0e+31 }
    "FORMAT"      CDF_CHAR     { "F8.3" }
    "UNITS"        CDF_CHAR     { " " }
    "LABLAXIS"    CDF_CHAR     { "Amplitude on Channel 1" }
    "VALIDMIN"    CDF_FLOAT     { -100000 }
    "VALIDMAX"    CDF_FLOAT     { 100000 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "SCALETYP"    CDF_CHAR     { "lin" }
    "SCALEMIN"    CDF_FLOAT     { -100000 }
    "SCALEMAX"    CDF_FLOAT     { 100000 }
    "UCD"         CDF_CHAR     { "phys.flux;em.radio" } .

  ! RV values were not requested.



#end
```

Radio-JOVE DTS CDF skeleton

The CDF skeleton code below shall be used for dual channel time series (i.e., single frequency, 2 crossed antennas) Radio-JOVE setup. The CDF skeleton filename is:

```
radiojove_dts_000000000000_000000000000_v02.skt
```

```
! Skeleton table for the "1418948900177" CDF.
! Generated: Thursday, 18-Dec-2014 16:44:07
! CDF created/modified by CDF V3.5.0
! Skeleton table created by CDF V3.5.0_2

#header

                       CDF NAME: 1418948900177
                  DATA ENCODING: PPC
                       MAJORITY: COLUMN
                         FORMAT: SINGLE

! Variables  G.Attributes  V.Attributes  Records  Dims  Sizes
! ---------  ------------  ------------  -------  ----  -----
    0/4          53            37          0/z      0
! CDF_COMPRESSION: None
! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
! CDF_CHECKSUM: MD5
! (Valid checksum: None, MD5)


#GLOBALattributes

! Attribute         Entry       Data
! Name              Number      Type       Value
! ---------         ------      ----       -----

  "Project"             1:    CDF_CHAR     { "RadioJOVE" } .

  "Discipline"          1:    CDF_CHAR     { "Space " -
                                             "Physics>Magnetospheric " -
                                             "Science" } .

  "Data_type"           1:    CDF_CHAR     { "RDR" } .

  "Descriptor"          1:    CDF_CHAR     { "DTS" } .
```

```
"Data_version"          1:    CDF_CHAR     { "02" } .

"Instrument_type"       1:    CDF_CHAR     { "Waveform Receiver" } .

"Logical_file_id"       1:    CDF_CHAR     { "radiojove_dts_00000000" -
                                             "0000_000000000000_v02" } .

"Logical_source"        1:    CDF_CHAR     { "radiojove_dts" } .

"Logical_source_description"
                        1:    CDF_CHAR     { "RadioJOVE Single Channel " -
                                             "Spectrograph" } .

"File_naming_convention"
                        1:    CDF_CHAR     { "source_descriptor_yyyyMM" -
                                             "ddHHmm_yyyyMMddHHmm_V02" } .

"Mission_group"         1:    CDF_CHAR     { "RadioJOVE" } .

"PI_name"               1:    CDF_CHAR     { "RadioJOVE Project" } .

"PI_affiliation"        1:    CDF_CHAR     { "GSFC" } .

"Source_name"           1:    CDF_CHAR     { "RadioJOVE" } .

"TEXT"                  1:    CDF_CHAR     { "RadioJOVE Project data." } .

"Generated_by"          1:    CDF_CHAR     { "SkyPipe" }
                        2:    CDF_CHAR     { "RadioJOVE" }
                        3:    CDF_CHAR     { "VOPDC" } .

"Generation_date" .

"LINK_TEXT"             1:    CDF_CHAR     { "Radio-SkyPipe Software " -
                                             "webpage" }
                        2:    CDF_CHAR     { "RadioJOVE webpage" }
                        3:    CDF_CHAR     { "VOParis Data Centre webpage" } .

"LINK_TITLE"            1:    CDF_CHAR     { "Radio-SkyPipe Software" }
                        2:    CDF_CHAR     { "RadioJOVE" }
                        3:    CDF_CHAR     { "VOParis Data Centre" } .

"HTTP_LINK"             1:    CDF_CHAR     { "http://www.radiosky.com/sk" -
                                             "ypipeishere.html" }
                        2:    CDF_CHAR     { "http://radiojove.gsfc.nasa" -
                                             ".gov" }
                        3:    CDF_CHAR     { "http://voparis-europlanet." -
                                             "obspm.fr" } .

"MODS" .

"Parents" .

"Rules_of_use" .

"Skeleton_version"
                        1:    CDF_CHAR     { "0.2" } .

"Software_version"
                        1:    CDF_CHAR     { "0.1" } .

"Time_resolution" .

"Acknowledgement" .
```

```
"ADID_ref" .

"Validate" .

"Observation_start_time"
                      1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_stop_time"
                      1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_target"
                      1:    CDF_CHAR     { "Saturn" } .

"Observation_type"
                      1:    CDF_CHAR     { "Waves" } .

"dataproduct_type"
                      1:    CDF_CHAR     { "TS>Time Series" } .

"target_class"        1:    CDF_CHAR     { "planet" } .

"target_region"       1:    CDF_CHAR     { " " } .

"target_element"      1:    CDF_CHAR     { " " } .

"time_min"            1:    CDF_REAL8    { 0.0 } .

"time_max"            1:    CDF_REAL8    { 0.0 } .

"time_sampling_min"
                      1:    CDF_REAL4    { 0.0 } .

"time_sampling_max"
                      1:    CDF_REAL4    { 0.0 } .

"spectral_range_min"
                      1:    CDF_REAL4    { 0.0 } .

"spectral_range_max"
                      1:    CDF_REAL4    { 0.0 } .

"instrument_host_name"
                      1:    CDF_CHAR     { " " } .

"instrument_name"     1:    CDF_CHAR     { " " } .

"measurement_type"
                      1:    CDF_CHAR     { "phys.flux;em.radio" } .

"access_format"       1:    CDF_CHAR     { "cdf" } .

"observer_name"       1:    CDF_CHAR     { " " } .

"observatory_location"
                      1:    CDF_CHAR     { " " } .

"observatory_latitude"
                      1:    CDF_REAL8    { 0.0 } .

"observatory_longitude"
                      1:    CDF_REAL8    { 0.0 } .

"original_filename"
                      1:    CDF_CHAR     { " " } .

"n_channel"           1:    CDF_BYTE     { 0 } .
```

```
#VARIABLEattributes

  "CATDESC"
  "DEPEND_0"
  "DEPEND_1"
  "DEPEND_2"
  "DEPEND_3"
  "DICT_KEY"
  "DISPLAY_TYPE"
  "FIELDNAM"
  "FILLVAL"
  "FORMAT"
  "LABLAXIS"
  "LABL_PTR_1"
  "LABL_PTR_2"
  "LABL_PTR_3"
  "UNITS"
  "UNIT_PTR"
  "VALIDMIN"
  "VALIDMAX"
  "VAR_TYPE"
  "SCALETYP"
  "SCAL_PTR"
  "VAR_NOTES"
  "MONOTON"
  "LEAP_SECONDS_INCLUDED"
  "RESOLUTION"
  "Bin_location"
  "TIME_BASE"
  "TIME_SCALE"
  "REFERENCE_POSITION"
  "ABSOLUTE_ERROR"
  "RELATIVE_ERROR"
  "FORM_PTR"
  "DELTA_PLUS_VAR"
  "DELTA_MINUS_VAR"
  "SCALEMIN"
  "SCALEMAX"
  "UCD"


#variables

! No rVariables.


#zVariables

! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "Epoch"         CDF_EPOCH16     1       0              T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: 01-Jan-0000 00:00:00.000.000.000.000

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----
```

```
    "CATDESC"     CDF_CHAR     { "Default time" }
    "FIELDNAM"    CDF_CHAR     { "Epoch" }
    "FILLVAL"     CDF_EPOCH16   { 31-Dec-9999 23:59:59.999.999.999.999 }
    "LABLAXIS"    CDF_CHAR     { "Epoch" }
    "UNITS"       CDF_CHAR     { " " }
    "VALIDMIN"    CDF_EPOCH16   { 01-Jan-2000 00:00:00.000.000.000.000 }
    "VALIDMAX"    CDF_EPOCH16   { 01-Jan-2020 00:00:00.000.000.000.000 }
    "VAR_TYPE"    CDF_CHAR     { "support_data" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "MONOTON"     CDF_CHAR     { "INCREASE" }
    "TIME_BASE"   CDF_CHAR     { "J2000" }
    "TIME_SCALE"  CDF_CHAR     { "UTC" }
    "REFERENCE_POSITION"
                  CDF_CHAR     { "Earth" }
    "SCALEMIN"    CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "SCALEMAX"    CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "UCD"         CDF_CHAR     { "time.epoch" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "DATA_1"       CDF_FLOAT      1       0              T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE:   -1.0e+30

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "CATDESC"     CDF_CHAR     { "Data stream on Channel 1" }
    "DEPEND_0"    CDF_CHAR     { "Epoch" }
    "DICT_KEY"    CDF_CHAR     { "data" }
    "DISPLAY_TYPE"
                  CDF_CHAR     { "time_series" }
    "FIELDNAM"    CDF_CHAR     { "DATA_1" }
    "FILLVAL"     CDF_FLOAT     { -1.0e+31 }
    "FORMAT"      CDF_CHAR     { "F8.3" }
    "UNITS"        CDF_CHAR     { " " }
    "LABLAXIS"    CDF_CHAR     { "Amplitude on Channel 1" }
    "VALIDMIN"    CDF_REAL4     { -100000 }
    "VALIDMAX"    CDF_REAL4     { 100000 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "SCALETYP"    CDF_CHAR     { "lin" }
    "SCALEMIN"    CDF_REAL4     { -100000 }
    "SCALEMAX"    CDF_REAL4     { 100000 }
    "UCD"         CDF_CHAR     { "phys.flux;em.radio" } .

  ! RV values were not requested.



! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "DATA_2"       CDF_FLOAT      1       0              T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE:   -1.0e+30

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "CATDESC"     CDF_CHAR     { "Data stream on Channel 2" }
    "DEPEND_0"    CDF_CHAR     { "Epoch" }
    "DICT_KEY"    CDF_CHAR     { "data" }
    "DISPLAY_TYPE"
                  CDF_CHAR     { "time_series" }
    "FIELDNAM"    CDF_CHAR     { "DATA_2" }
    "FILLVAL"     CDF_FLOAT     { -1.0e+31 }
    "FORMAT"      CDF_CHAR     { "F8.3" }
    "UNITS"        CDF_CHAR     { " " }
    "LABLAXIS"    CDF_CHAR     { "Amplitude on Channel 2" }
    "VALIDMIN"    CDF_FLOAT     { -100000 }
    "VALIDMAX"    CDF_FLOAT     { 100000 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "SCALETYP"    CDF_CHAR     { "lin" }
    "SCALEMIN"    CDF_FLOAT     { -100000 }
    "SCALEMAX"    CDF_FLOAT     { 100000 }
    "UCD"         CDF_CHAR     { "phys.flux;em.radio" } .

  ! RV values were not requested.


#end
```

Radio-JOVE SDS CDF skeleton

The CDF skeleton code below shall be used for single channel dynamic spectrum (i.e., wide band, 1 antenna) Radio-JOVE setup. The wide band receiver is here tuned to 200 frequency steps. The CDF skeleton filename is:

```
    radiojove_sds_000000000000_000000000000_v02.skt
```

```
! Skeleton table for the "1418948900177" CDF.
! Generated: Thursday, 18-Dec-2014 16:44:07
! CDF created/modified by CDF V3.5.0
! Skeleton table created by CDF V3.5.0_2

#header

                       CDF NAME: 1418948900177
                  DATA ENCODING: PPC
                       MAJORITY: COLUMN
                         FORMAT: SINGLE

! Variables  G.Attributes  V.Attributes  Records  Dims  Sizes
! ---------  ------------  ------------  -------  ----  -----
     0/4          53            37          0/z      0
! CDF_COMPRESSION: None
! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
! CDF_CHECKSUM: MD5
! (Valid checksum: None, MD5)


#GLOBALattributes

! Attribute         Entry       Data
! Name              Number      Type       Value
! ---------         ------      ----       -----

  "Project"               1:    CDF_CHAR     { "RadioJOVE" } .

  "Discipline"            1:    CDF_CHAR     { "Space " -
                                               "Physics>Magnetospheric " -
                                               "Science" } .

  "Data_type"             1:    CDF_CHAR     { "RDR" } .

  "Descriptor"            1:    CDF_CHAR     { "SDS" } .

  "Data_version"          1:    CDF_CHAR     { "02" } .

  "Instrument_type"       1:    CDF_CHAR     { "Spectrogram Receiver" } .

  "Logical_file_id"       1:    CDF_CHAR     { "radiojove_sds_00000000" -
                                               "0000_000000000000_v02" } .

  "Logical_source"        1:    CDF_CHAR     { "radiojove_sds" } .

  "Logical_source_description"
                          1:    CDF_CHAR     { "RadioJOVE Single Channel " -
                                               "Spectrograph" } .

  "File_naming_convention"
                          1:    CDF_CHAR     { "source_descriptor_yyyyMM" -
                                               "ddHHmm_yyyyMMddHHmm_V02" } .

  "Mission_group"         1:    CDF_CHAR     { "RadioJOVE" } .

  "PI_name"               1:    CDF_CHAR     { "RadioJOVE Project" } .

  "PI_affiliation"        1:    CDF_CHAR     { "GSFC" } .

  "Source_name"           1:    CDF_CHAR     { "RadioJOVE" } .

  "TEXT"                  1:    CDF_CHAR     { "RadioJOVE Project data." } .

  "Generated_by"          1:    CDF_CHAR     { "SkyPipe" }
                          2:    CDF_CHAR     { "RadioJOVE" }
                          3:    CDF_CHAR     { "VOPDC" } .

  "Generation_date" .

  "LINK_TEXT"             1:    CDF_CHAR     { "Radio-SkyPipe Software " -
                                               "webpage" }
                          2:    CDF_CHAR     { "RadioJOVE webpage" }
                          3:    CDF_CHAR     { "VOParis Data Centre webpage" } .

  "LINK_TITLE"            1:    CDF_CHAR     { "Radio-SkyPipe Software" }
                          2:    CDF_CHAR     { "RadioJOVE" }
                          3:    CDF_CHAR     { "VOParis Data Centre" } .

  "HTTP_LINK"             1:    CDF_CHAR     { "http://www.radiosky.com/sk" -
                                               "ypipeishere.html" }
                          2:    CDF_CHAR     { "http://radiojove.gsfc.nasa" -
                                               ".gov" }
                          3:    CDF_CHAR     { "http://voparis-europlanet." -
                                               "obspm.fr" } .

  "MODS" .

  "Parents" .

  "Rules_of_use" .
```

```
"Skeleton_version"
                      1:    CDF_CHAR     { "0.2" } .

"Software_version"
                      1:    CDF_CHAR     { "0.1" } .

"Time_resolution" .

"Acknowledgement" .

"ADID_ref" .

"Validate" .

"Observation_start_time"
                      1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_stop_time"
                      1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_target"
                      1:    CDF_CHAR     { "Saturn" } .

"Observation_type"
                      1:    CDF_CHAR     { "Waves" } .

"dataproduct_type"
                      1:    CDF_CHAR     { "DS>Dynamic Spectra" } .

"target_class"        1:    CDF_CHAR     { "planet" } .

"target_region"       1:    CDF_CHAR     { " " } .

"target_element"      1:    CDF_CHAR     { " " } .

"time_min"            1:    CDF_REAL8    { 0.0 } .

"time_max"            1:    CDF_REAL8    { 0.0 } .

"time_sampling_min"
                      1:    CDF_REAL4    { 0.0 } .

"time_sampling_max"
                      1:    CDF_REAL4    { 0.0 } .

"spectral_range_min"
                      1:    CDF_REAL4    { 0.0 } .

"spectral_range_max"
                      1:    CDF_REAL4    { 0.0 } .

"instrument_host_name"
                      1:    CDF_CHAR     { " " } .

"instrument_name"     1:    CDF_CHAR     { " " } .

"measurement_type"
                      1:    CDF_CHAR     { "phys.flux;em.radio" } .

"access_format"       1:    CDF_CHAR     { "cdf" } .

"observer_name"       1:    CDF_CHAR     { " " } .

"observatory_location"
                      1:    CDF_CHAR     { " " } .
```

```
  "observatory_latitude"
                      1:    CDF_REAL8    { 0.0 } .

  "observatory_longitude"
                      1:    CDF_REAL8    { 0.0 } .

  "original_filename"
                      1:    CDF_CHAR     { " " } .

  "n_channel"         1:    CDF_BYTE     { 0 } .


#VARIABLEattributes

  "CATDESC"
  "DEPEND_0"
  "DEPEND_1"
  "DEPEND_2"
  "DEPEND_3"
  "DICT_KEY"
  "DISPLAY_TYPE"
  "FIELDNAM"
  "FILLVAL"
  "FORMAT"
  "LABLAXIS"
  "LABL_PTR_1"
  "LABL_PTR_2"
  "LABL_PTR_3"
  "UNITS"
  "UNIT_PTR"
  "VALIDMIN"
  "VALIDMAX"
  "VAR_TYPE"
  "SCALETYP"
  "SCAL_PTR"
  "VAR_NOTES"
  "MONOTON"
  "LEAP_SECONDS_INCLUDED"
  "RESOLUTION"
  "Bin_location"
  "TIME_BASE"
  "TIME_SCALE"
  "REFERENCE_POSITION"
  "ABSOLUTE_ERROR"
  "RELATIVE_ERROR"
  "FORM_PTR"
  "DELTA_PLUS_VAR"
  "DELTA_MINUS_VAR"
  "SCALEMIN"
  "SCALEMAX"
  "UCD"


#variables

! No rVariables.


#zVariables

! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "Epoch"         CDF_EPOCH16     1        0              T
  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: 01-Jan-0000 00:00:00.000.000.000.000

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "CATDESC"     CDF_CHAR     { "Default time" }
    "FIELDNAM"    CDF_CHAR     { "Epoch" }
    "FILLVAL"     CDF_EPOCH16   { 31-Dec-9999 23:59:59.999.999.999.999 }
    "LABLAXIS"    CDF_CHAR     { "Epoch" }
    "UNITS"       CDF_CHAR     { " " }
    "VALIDMIN"    CDF_EPOCH16   { 01-Jan-2000 00:00:00.000.000.000.000 }
    "VALIDMAX"    CDF_EPOCH16   { 01-Jan-2020 00:00:00.000.000.000.000 }
    "VAR_TYPE"    CDF_CHAR     { "support_data" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "MONOTON"     CDF_CHAR     { "INCREASE" }
    "TIME_BASE"   CDF_CHAR     { "J2000" }
    "TIME_SCALE"  CDF_CHAR     { "UTC" }
    "REFERENCE_POSITION"
                  CDF_CHAR     { "Earth" }
    "SCALEMIN"    CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "SCALEMAX"    CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "UCD"         CDF_CHAR     { "time.epoch" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "DATA_1"        CDF_FLOAT      1       1    200       T         T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: -1.0e+30

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "CATDESC"     CDF_CHAR     { "Data stream on Channel 1" }
    "DEPEND_0"    CDF_CHAR     { "Epoch" }
    "DEPEND_1"    CDF_CHAR     { "FREQUENCY" }
    "DICT_KEY"    CDF_CHAR     { "data" }
    "DISPLAY_TYPE"
                  CDF_CHAR     { "spectrogram" }
    "FIELDNAM"    CDF_CHAR     { "DATA_1" }
    "FILLVAL"     CDF_FLOAT    { -1.0e+31 }
    "FORMAT"      CDF_CHAR     { "F8.3" }
    "LABLAXIS"    CDF_CHAR     { "Power Spectral Density on Channel 1" }
    "UNITS"       CDF_CHAR     { " " }
    "VALIDMIN"    CDF_FLOAT    { 0.0 }
    "VALIDMAX"    CDF_FLOAT    { 5000.0 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "SCALETYP"    CDF_CHAR     { "lin" }
    "SCALEMIN"    CDF_FLOAT    { 0.0 }
    "SCALEMAX"    CDF_FLOAT    { 5000.0 }
    "UCD"         CDF_CHAR     { "phys.flux;em.radio" } .
  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ----------

  "FREQUENCY"     CDF_FLOAT      1       1    200       F         T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: -1.0e+30

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "CATDESC"     CDF_CHAR     { "Frequency value" }
    "DICT_KEY"    CDF_CHAR     { "support_data" }
    "DISPLAY_TYPE"
                  CDF_CHAR     { "time_series" }
    "FIELDNAM"    CDF_CHAR     { "DATA_R" }
    "FILLVAL"     CDF_FLOAT    { -1.0e+31 }
    "FORMAT"      CDF_CHAR     { "F15.3" }
    "LABLAXIS"    CDF_CHAR     { "Frequency" }
    "UNITS"       CDF_CHAR     { "Hz" }
    "VALIDMIN"    CDF_FLOAT    { 0.0 }
    "VALIDMAX"    CDF_FLOAT    { 1.0e+08 }
    "VAR_TYPE"    CDF_CHAR     { "support_data" }
    "SCALETYP"    CDF_CHAR     { "log" }
    "SCALEMIN"    CDF_FLOAT    { 1.0e+07 }
    "SCALEMAX"    CDF_FLOAT    { 4.0e+07 }
    "UCD"         CDF_CHAR     { "em.freq" } .

  ! NRV values follow...

    [1] = 0.0
    [2] = 0.0
    [3] = 0.0
    [4] = 0.0
    [5] = 0.0
    [6] = 0.0
    [7] = 0.0
    [8] = 0.0
    [9] = 0.0
    [10] = 0.0
    [11] = 0.0
    [12] = 0.0
    [13] = 0.0
    [14] = 0.0
    [15] = 0.0
    [16] = 0.0
    [17] = 0.0
    [18] = 0.0
    [19] = 0.0
    [20] = 0.0
    [21] = 0.0
    [22] = 0.0
    [23] = 0.0
    [24] = 0.0
    [25] = 0.0
    [26] = 0.0
    [27] = 0.0
    [28] = 0.0
    [29] = 0.0
    [30] = 0.0
    [31] = 0.0
    [32] = 0.0
    [33] = 0.0
    [34] = 0.0
    [35] = 0.0
    [36] = 0.0
    [37] = 0.0
    [38] = 0.0
    [39] = 0.0
    [40] = 0.0
    [41] = 0.0
    [42] = 0.0
    [43] = 0.0
    [44] = 0.0
    [45] = 0.0
    [46] = 0.0
    [47] = 0.0
    [48] = 0.0
    [49] = 0.0
    [50] = 0.0
    [51] = 0.0
    [52] = 0.0
    [53] = 0.0
    [54] = 0.0
    [55] = 0.0
    [56] = 0.0
    [57] = 0.0
    [58] = 0.0
    [59] = 0.0
    [60] = 0.0
    [61] = 0.0
    [62] = 0.0
    [63] = 0.0
    [64] = 0.0
    [65] = 0.0
    [66] = 0.0
    [67] = 0.0
    [68] = 0.0
    [69] = 0.0
    [70] = 0.0
    [71] = 0.0
    [72] = 0.0
    [73] = 0.0
    [74] = 0.0
    [75] = 0.0
    [76] = 0.0
    [77] = 0.0
    [78] = 0.0
    [79] = 0.0
    [80] = 0.0
    [81] = 0.0
    [82] = 0.0
    [83] = 0.0
    [84] = 0.0
    [85] = 0.0
    [86] = 0.0
    [87] = 0.0
    [88] = 0.0
    [89] = 0.0
    [90] = 0.0
    [91] = 0.0
    [92] = 0.0
    [93] = 0.0
    [94] = 0.0
    [95] = 0.0
    [96] = 0.0
    [97] = 0.0
    [98] = 0.0
    [99] = 0.0
    [100] = 0.0
    [101] = 0.0
    [102] = 0.0
    [103] = 0.0
    [104] = 0.0
    [105] = 0.0
    [106] = 0.0
    [107] = 0.0
    [108] = 0.0
    [109] = 0.0
    [110] = 0.0
    [111] = 0.0
    [112] = 0.0
    [113] = 0.0
    [114] = 0.0
    [115] = 0.0
    [116] = 0.0
    [117] = 0.0
    [118] = 0.0
    [119] = 0.0
    [120] = 0.0
    [121] = 0.0
    [122] = 0.0
    [123] = 0.0
    [124] = 0.0
    [125] = 0.0
    [126] = 0.0
    [127] = 0.0
    [128] = 0.0
    [129] = 0.0
    [130] = 0.0
    [131] = 0.0
    [132] = 0.0
    [133] = 0.0
    [134] = 0.0
    [135] = 0.0
    [136] = 0.0
    [137] = 0.0
    [138] = 0.0
    [139] = 0.0
    [140] = 0.0
    [141] = 0.0
    [142] = 0.0
    [143] = 0.0
    [144] = 0.0
    [145] = 0.0
    [146] = 0.0
    [147] = 0.0
    [148] = 0.0
    [149] = 0.0
    [150] = 0.0
    [151] = 0.0
    [152] = 0.0
    [153] = 0.0
    [154] = 0.0
    [155] = 0.0
    [156] = 0.0
    [157] = 0.0
    [158] = 0.0
    [159] = 0.0
    [160] = 0.0
    [161] = 0.0
    [162] = 0.0
    [163] = 0.0
    [164] = 0.0
    [165] = 0.0
    [166] = 0.0
    [167] = 0.0
    [168] = 0.0
    [169] = 0.0
    [170] = 0.0
    [171] = 0.0
    [172] = 0.0
    [173] = 0.0
    [174] = 0.0
    [175] = 0.0
    [176] = 0.0
    [177] = 0.0
    [178] = 0.0
    [179] = 0.0
    [180] = 0.0
    [181] = 0.0
    [182] = 0.0
    [183] = 0.0
    [184] = 0.0
    [185] = 0.0
    [186] = 0.0
    [187] = 0.0
    [188] = 0.0
    [189] = 0.0
    [190] = 0.0
    [191] = 0.0
    [192] = 0.0
    [193] = 0.0
    [194] = 0.0
    [195] = 0.0
    [196] = 0.0
    [197] = 0.0
    [198] = 0.0
    [199] = 0.0
    [200] = 0.0


#end
```

Radio-JOVE DDS CDF skeleton

The CDF skeleton code below shall be used for dual channel dynamic spectrum (i.e., wide band, 2 antenna) Radio-JOVE setup. The wide band receiver is here tuned to 200 frequency steps. The CDF skeleton filename is:

```
radiojove_dds_000000000000_000000000000_v02.skt
```

```
! Skeleton table for the "1418948900177" CDF.
! Generated: Thursday, 18-Dec-2014 16:44:07
! CDF created/modified by CDF V3.5.0
! Skeleton table created by CDF V3.5.0_2

#header

                       CDF NAME: 1418948900177
                  DATA ENCODING: PPC
                       MAJORITY: COLUMN
                         FORMAT: SINGLE
```

```
! Variables  G.Attributes  V.Attributes  Records  Dims  Sizes
! ---------  ------------  ------------  -------  ----  -----
    0/4          53            37          0/z      0
! CDF_COMPRESSION: None
! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
! CDF_CHECKSUM: MD5
! (Valid checksum: None, MD5)


#GLOBALattributes

! Attribute         Entry       Data
! Name              Number      Type       Value
! ---------         ------      ----       -----

  "Project"             1:    CDF_CHAR     { "RadioJOVE" } .

  "Discipline"          1:    CDF_CHAR     { "Space " -
                                             "Physics>Magnetospheric " -
                                             "Science" } .

  "Data_type"           1:    CDF_CHAR     { "RDR" } .

  "Descriptor"          1:    CDF_CHAR     { "DDS" } .

  "Data_version"        1:    CDF_CHAR     { "02" } .

  "Instrument_type"     1:    CDF_CHAR     { "Spectrogram Receiver" } .

  "Logical_file_id"     1:    CDF_CHAR     { "radiojove_dds_00000000" -
                                             "0000_000000000000_v02" } .

  "Logical_source"      1:    CDF_CHAR     { "radiojove_dds" } .

  "Logical_source_description"
                        1:    CDF_CHAR     { "RadioJOVE Dual Channel " -
                                             "Spectrograph" } .

  "File_naming_convention"
                        1:    CDF_CHAR     { "source_descriptor_yyyyMM" -
                                             "ddHHmm_yyyyMMddHHmm_V02" } .

  "Mission_group"       1:    CDF_CHAR     { "RadioJOVE" } .

  "PI_name"             1:    CDF_CHAR     { "RadioJOVE Project" } .

  "PI_affiliation"      1:    CDF_CHAR     { "GSFC" } .

  "Source_name"         1:    CDF_CHAR     { "RadioJOVE" } .

  "TEXT"                1:    CDF_CHAR     { "RadioJOVE Project data." } .

  "Generated_by"        1:    CDF_CHAR     { "SkyPipe" }
                        2:    CDF_CHAR     { "RadioJOVE" }
                        3:    CDF_CHAR     { "VOPDC" } .

  "Generation_date" .

  "LINK_TEXT"           1:    CDF_CHAR     { "Radio-SkyPipe Software " -
                                             "webpage" }
                        2:    CDF_CHAR     { "RadioJOVE webpage" }
                        3:    CDF_CHAR     { "VOParis Data Centre webpage" } .

  "LINK_TITLE"          1:    CDF_CHAR     { "Radio-SkyPipe Software" }
                        2:    CDF_CHAR     { "RadioJOVE" }
                        3:    CDF_CHAR     { "VOParis Data Centre" } .
```

```
"HTTP_LINK"            1:    CDF_CHAR     { "http://www.radiosky.com/sk" -
                                            "ypipeishere.html" }
                       2:    CDF_CHAR     { "http://radiojove.gsfc.nasa" -
                                            ".gov" }
                       3:    CDF_CHAR     { "http://voparis-europlanet." -
                                            "obspm.fr" } .

"MODS" .

"Parents" .

"Rules_of_use" .

"Skeleton_version"
                       1:    CDF_CHAR     { "0.2" } .

"Software_version"
                       1:    CDF_CHAR     { "0.1" } .

"Time_resolution" .

"Acknowledgement" .

"ADID_ref" .

"Validate" .

"Observation_start_time"
                       1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_stop_time"
                       1:    CDF_CHAR     { "0000-01-01T00:00:00.000Z" } .

"Observation_target"
                       1:    CDF_CHAR     { "Saturn" } .

"Observation_type"
                       1:    CDF_CHAR     { "Waves" } .

"dataproduct_type"
                       1:    CDF_CHAR     { "DS>Dynamic Spectra" } .

"target_class"         1:    CDF_CHAR     { "planet" } .

"target_region"        1:    CDF_CHAR     { " " } .

"target_element"       1:    CDF_CHAR     { " " } .

"time_min"             1:    CDF_REAL8    { 0.0 } .

"time_max"             1:    CDF_REAL8    { 0.0 } .

"time_sampling_min"
                       1:    CDF_REAL4    { 0.0 } .

"time_sampling_max"
                       1:    CDF_REAL4    { 0.0 } .

"spectral_range_min"
                       1:    CDF_REAL4    { 0.0 } .

"spectral_range_max"
                       1:    CDF_REAL4    { 0.0 } .

"instrument_host_name"
```

```
                      1:    CDF_CHAR     { " " } .

  "instrument_name"     1:    CDF_CHAR     { " " } .

  "measurement_type"
                      1:    CDF_CHAR     { "phys.flux;em.radio" } .

  "access_format"       1:    CDF_CHAR     { "cdf" } .

  "observer_name"       1:    CDF_CHAR     { " " } .

  "observatory_location"
                      1:    CDF_CHAR     { " " } .

  "observatory_latitude"
                      1:    CDF_REAL8    { 0.0 } .

  "observatory_longitude"
                      1:    CDF_REAL8    { 0.0 } .

  "original_filename"
                      1:    CDF_CHAR     { " " } .

  "n_channel"           1:    CDF_BYTE     { 0 } .


#VARIABLEattributes

  "CATDESC"
  "DEPEND_0"
  "DEPEND_1"
  "DEPEND_2"
  "DEPEND_3"
  "DICT_KEY"
  "DISPLAY_TYPE"
  "FIELDNAM"
  "FILLVAL"
  "FORMAT"
  "LABLAXIS"
  "LABL_PTR_1"
  "LABL_PTR_2"
  "LABL_PTR_3"
  "UNITS"
  "UNIT_PTR"
  "VALIDMIN"
  "VALIDMAX"
  "VAR_TYPE"
  "SCALETYP"
  "SCAL_PTR"
  "VAR_NOTES"
  "MONOTON"
  "LEAP_SECONDS_INCLUDED"
  "RESOLUTION"
  "Bin_location"
  "TIME_BASE"
  "TIME_SCALE"
  "REFERENCE_POSITION"
  "ABSOLUTE_ERROR"
  "RELATIVE_ERROR"
  "FORM_PTR"
  "DELTA_PLUS_VAR"
  "DELTA_MINUS_VAR"
  "SCALEMIN"
  "SCALEMAX"
  "UCD"
```

```
#variables

! No rVariables.


#zVariables

! Variable           Data      Number                 Record   Dimension
! Name               Type     Elements  Dims  Sizes  Variance  Variances
! --------           ----     --------  ----  -----  --------  ---------

  "Epoch"          CDF_EPOCH16     1       0              T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: 01-Jan-0000 00:00:00.000.000.000.000

  ! Attribute        Data
  ! Name             Type       Value
  ! --------         ----       -----

    "CATDESC"      CDF_CHAR     { "Default time" }
    "FIELDNAM"     CDF_CHAR     { "Epoch" }
    "FILLVAL"      CDF_EPOCH16   { 31-Dec-9999 23:59:59.999.999.999.999 }
    "LABLAXIS"     CDF_CHAR     { "Epoch" }
    "UNITS"        CDF_CHAR     { " " }
    "VALIDMIN"     CDF_EPOCH16   { 01-Jan-2000 00:00:00.000.000.000.000 }
    "VALIDMAX"     CDF_EPOCH16   { 01-Jan-2020 00:00:00.000.000.000.000 }
    "VAR_TYPE"     CDF_CHAR     { "support_data" }
    "SCALETYP"     CDF_CHAR     { "linear" }
    "MONOTON"      CDF_CHAR     { "INCREASE" }
    "TIME_BASE"    CDF_CHAR     { "J2000" }
    "TIME_SCALE"   CDF_CHAR     { "UTC" }
    "REFERENCE_POSITION"
                   CDF_CHAR     { "Earth" }
    "SCALEMIN"     CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "SCALEMAX"     CDF_EPOCH16   { 01-Jan-0000 00:00:00.000.000.000.000 }
    "UCD"          CDF_CHAR     { "time.epoch" } .

  ! RV values were not requested.


! Variable           Data      Number                 Record   Dimension
! Name               Type     Elements  Dims  Sizes  Variance  Variances
! --------           ----     --------  ----  -----  --------  ---------

  "DATA_1"         CDF_FLOAT      1       1     200      T          T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: -1.0e+30

  ! Attribute        Data
  ! Name             Type       Value
  ! --------         ----       -----

    "CATDESC"      CDF_CHAR     { "Data stream on Channel 1" }
    "DEPEND_0"     CDF_CHAR     { "Epoch" }
    "DEPEND_1"     CDF_CHAR     { "FREQUENCY" }
    "DICT_KEY"     CDF_CHAR     { "data" }
    "DISPLAY_TYPE"
```

```
                 CDF_CHAR     { "spectrogram" }
   "FIELDNAM"    CDF_CHAR     { "DATA_1" }
   "FILLVAL"     CDF_FLOAT    { -1.0e+31 }
   "FORMAT"      CDF_CHAR     { "F8.3" }
   "LABLAXIS"    CDF_CHAR     { "Power Spectral Density on Channel 1" }
   "UNITS"       CDF_CHAR     { " " }
   "VALIDMIN"    CDF_FLOAT    { 0.0 }
   "VALIDMAX"    CDF_FLOAT    { 5000.0 }
   "VAR_TYPE"    CDF_CHAR     { "data" }
   "SCALETYP"    CDF_CHAR     { "lin" }
   "SCALEMIN"    CDF_FLOAT    { 0.0 }
   "SCALEMAX"    CDF_FLOAT    { 5000.0 }
   "UCD"         CDF_CHAR     { "phys.flux;em.radio" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "DATA_2"        CDF_FLOAT      1       1    200       T         T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: -1.0e+30

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

   "CATDESC"     CDF_CHAR     { "Data stream on Channel 2" }
   "DEPEND_0"    CDF_CHAR     { "Epoch" }
   "DEPEND_1"    CDF_CHAR     { "FREQUENCY" }
   "DICT_KEY"    CDF_CHAR     { "data" }
   "DISPLAY_TYPE"
                 CDF_CHAR     { "spectrogram" }
   "FIELDNAM"    CDF_CHAR     { "DATA_2" }
   "FILLVAL"     CDF_FLOAT    { -1.0e+31 }
   "FORMAT"      CDF_CHAR     { "F8.3" }
   "LABLAXIS"    CDF_CHAR     { "Power Spectral Density on Channel 2" }
   "UNITS"       CDF_CHAR     { " " }
   "VALIDMIN"    CDF_FLOAT    { 0.0 }
   "VALIDMAX"    CDF_FLOAT    { 5000.0 }
   "VAR_TYPE"    CDF_CHAR     { "data" }
   "SCALETYP"    CDF_CHAR     { "lin" }
   "SCALEMIN"    CDF_FLOAT    { 0.0 }
   "SCALEMAX"    CDF_FLOAT    { 5000.0 }
   "UCD"         CDF_CHAR     { "phys.flux;em.radio" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "FREQUENCY"     CDF_FLOAT      1       1    200       F         T

  ! VAR_COMPRESSION: None
  ! (Valid compression: None, GZIP.1-9, RLE.0, HUFF.0, AHUFF.0)
  ! VAR_SPARSERECORDS: None
  ! (Valid sparserecords: None, sRecords.PAD, sRecords.PREV)
  ! VAR_PADVALUE: -1.0e+30
  ! Attribute       Data
  ! Name            Type       Value
  ! ---------       ----       -----

   "CATDESC"     CDF_CHAR     { "Frequency value" }
   "DICT_KEY"    CDF_CHAR     { "support_data" }
   "DISPLAY_TYPE"
                 CDF_CHAR     { "time_series" }
   "FIELDNAM"    CDF_CHAR     { "DATA_R" }
   "FILLVAL"     CDF_FLOAT    { -1.0e+31 }
   "FORMAT"      CDF_CHAR     { "F15.3" }
   "LABLAXIS"    CDF_CHAR     { "Frequency" }
   "UNITS"       CDF_CHAR     { "Hz" }
   "VALIDMIN"    CDF_FLOAT    { 0.0 }
   "VALIDMAX"    CDF_FLOAT    { 1.0e+08 }
   "VAR_TYPE"    CDF_CHAR     { "support_data" }
   "SCALETYP"    CDF_CHAR     { "log" }
   "SCALEMIN"    CDF_FLOAT    { 1.0e+07 }
   "SCALEMAX"    CDF_FLOAT    { 4.0e+07 }
   "UCD"         CDF_CHAR     { "em.freq" } .

  ! NRV values follow...

   [1] = 0.0
   [2] = 0.0
   [3] = 0.0
   [4] = 0.0
   [5] = 0.0
   [6] = 0.0
   [7] = 0.0
   [8] = 0.0
   [9] = 0.0
   [10] = 0.0
   [11] = 0.0
   [12] = 0.0
   [13] = 0.0
   [14] = 0.0
   [15] = 0.0
   [16] = 0.0
   [17] = 0.0
   [18] = 0.0
   [19] = 0.0
   [20] = 0.0
   [21] = 0.0
   [22] = 0.0
   [23] = 0.0
   [24] = 0.0
   [25] = 0.0
   [26] = 0.0
   [27] = 0.0
   [28] = 0.0
   [29] = 0.0
   [30] = 0.0
   [31] = 0.0
   [32] = 0.0
   [33] = 0.0
   [34] = 0.0
   [35] = 0.0
   [36] = 0.0
   [37] = 0.0
   [38] = 0.0
   [39] = 0.0
   [40] = 0.0
   [41] = 0.0
   [42] = 0.0
```

```
    [43] = 0.0
    [44] = 0.0
    [45] = 0.0
    [46] = 0.0
    [47] = 0.0
    [48] = 0.0
    [49] = 0.0
    [50] = 0.0
    [51] = 0.0
    [52] = 0.0
    [53] = 0.0
    [54] = 0.0
    [55] = 0.0
    [56] = 0.0
    [57] = 0.0
    [58] = 0.0
    [59] = 0.0
    [60] = 0.0
    [61] = 0.0
    [62] = 0.0
    [63] = 0.0
    [64] = 0.0
    [65] = 0.0
    [66] = 0.0
    [67] = 0.0
    [68] = 0.0
    [69] = 0.0
    [70] = 0.0
    [71] = 0.0
    [72] = 0.0
    [73] = 0.0
    [74] = 0.0
    [75] = 0.0
    [76] = 0.0
    [77] = 0.0
    [78] = 0.0
    [79] = 0.0
    [80] = 0.0
    [81] = 0.0
    [82] = 0.0
    [83] = 0.0
    [84] = 0.0
    [85] = 0.0
    [86] = 0.0
    [87] = 0.0
    [88] = 0.0
    [89] = 0.0
    [90] = 0.0
    [91] = 0.0
    [92] = 0.0
    [93] = 0.0
    [94] = 0.0
    [95] = 0.0
    [96] = 0.0
    [97] = 0.0
    [98] = 0.0
    [99] = 0.0
    [100] = 0.0
    [101] = 0.0
    [102] = 0.0
    [103] = 0.0
    [104] = 0.0
    [105] = 0.0
    [106] = 0.0
    [107] = 0.0
    [108] = 0.0
    [109] = 0.0
    [110] = 0.0
    [111] = 0.0
    [112] = 0.0
    [113] = 0.0
    [114] = 0.0
    [115] = 0.0
    [116] = 0.0
    [117] = 0.0
    [118] = 0.0
    [119] = 0.0
    [120] = 0.0
    [121] = 0.0
    [122] = 0.0
    [123] = 0.0
    [124] = 0.0
    [125] = 0.0
    [126] = 0.0
    [127] = 0.0
    [128] = 0.0
    [129] = 0.0
    [130] = 0.0
    [131] = 0.0
    [132] = 0.0
    [133] = 0.0
    [134] = 0.0
    [135] = 0.0
    [136] = 0.0
    [137] = 0.0
    [138] = 0.0
    [139] = 0.0
    [140] = 0.0
    [141] = 0.0
    [142] = 0.0
    [143] = 0.0
    [144] = 0.0
    [145] = 0.0
    [146] = 0.0
    [147] = 0.0
    [148] = 0.0
    [149] = 0.0
    [150] = 0.0
    [151] = 0.0
    [152] = 0.0
    [153] = 0.0
    [154] = 0.0
    [155] = 0.0
    [156] = 0.0
    [157] = 0.0
    [158] = 0.0
    [159] = 0.0
    [160] = 0.0
    [161] = 0.0
    [162] = 0.0
    [163] = 0.0
    [164] = 0.0
    [165] = 0.0
    [166] = 0.0
    [167] = 0.0
    [168] = 0.0
    [169] = 0.0
    [170] = 0.0
    [171] = 0.0
    [172] = 0.0
    [173] = 0.0
    [174] = 0.0
    [175] = 0.0
    [176] = 0.0
    [177] = 0.0
    [178] = 0.0
    [179] = 0.0
    [180] = 0.0
    [181] = 0.0
    [182] = 0.0
    [183] = 0.0
    [184] = 0.0
    [185] = 0.0
    [186] = 0.0
    [187] = 0.0
    [188] = 0.0
    [189] = 0.0
    [190] = 0.0
    [191] = 0.0
    [192] = 0.0
    [193] = 0.0
    [194] = 0.0
    [195] = 0.0
    [196] = 0.0
    [197] = 0.0
    [198] = 0.0
    [199] = 0.0
    [200] = 0.0


#end
```

Appendix B

This appendix presents the IDL code developed for the Radio-JOVE prototype processing pipeline. It is using the SPDF IDL CDF library (CDAWlib), that must be compiled before execution of the IDL scripts.

make_radiojove_ts_cdf_v02.pro

```
; IDL Version 7.0, Mac OS X (darwin i386 m32)
; Journal File for baptiste@macbookbc.obspm.fr
; Working directory: /Users/baptiste/Projets/CDPP/Archivage/Cassini:RPWS/HFR/SND
; Date: Thu Jun 12 12:04:25 2014
;
; RUN @compile_IDLmakecdf
;
; NB: compile_IDLmakecdf.pro script is part of the CDF CDAWlib distribution
; available here: http://spdf.gsfc.nasa.gov/CDAWlib.html

; DECLARING VERSIONS
sft_version = '02'
cdf_version = '02'
dat_version = '01'

; DECLARING PATHS
cdf_datapath = 'data/cdf/V'+sft_version+'/'
dat_datapath = 'data/dat/V'+dat_version+'/timeseries/'
master_cdf_sts = cdf_datapath+'master/radiojove_sts_000000000000_000000000000_v'+ $
                 cdf_version+'.cdf'
skelet_cdf_sts = cdf_datapath+'master/radiojove_sts_000000000000_000000000000_v'+ $
                 cdf_version+'.skt'
master_cdf_dts = cdf_datapath+'master/radiojove_dts_000000000000_000000000000_v'+ $
```

```
                 cdf_version+'.cdf'
skelet_cdf_dts = cdf_datapath+'master/radiojove_dts_000000000000_000000000000_v'+ $
                 cdf_version+'.skt'

; CLEANING AND REBUILDING MASTER CDF
spawn,'rm -f '+master_cdf_sts
spawn,'skeletoncdf -cdf '+master_cdf_sts+' '+skelet_cdf_sts
spawn,'rm -f '+master_cdf_dts
spawn,'skeletoncdf -cdf '+master_cdf_dts+' '+skelet_cdf_dts

; OPENING 2 FILES FOR HOUSEKEEPING OUTPUT
openw,lun1,cdf_datapath+'misc/interval_listing.txt',/get_lun
openw,lun2,cdf_datapath+'misc/cdf_pds_check.txt',/get_lun

; BUILDING LIST OF DATA FILES
files = file_search(dat_datapath+"*.txt")

; ============================================================================
; LOOPING ON DATA FILES
; ============================================================================

for j=0,N_ELEMENTS(files)-1 do begin

  ; LOADING LABEL DATA
  raw_data = read_ascii_file(files[j])
  header = {radiojove_wav_header}
  header.first_name = raw_data[0]
  header.last_name = raw_data[1]
  header.school_obs = raw_data[2]
  header.start_ymd = [fix(strmid(raw_data[3],6,4)), $
                      fix(strmid(raw_data[3],0,2)), $
                      fix(strmid(raw_data[3],3,2))]
  header.start_hms = [fix(strmid(raw_data[4],0,2)), $
                      fix(strmid(raw_data[4],2,2)), $
                      (strlen(raw_data[4]) eq 4) ? $
                      0 : fix(strmid(raw_data[4],4,2))]
  header.stop_ymd  = [fix(strmid(raw_data[5],6,4)), $
                      fix(strmid(raw_data[5],0,2)), $
                      fix(strmid(raw_data[5],3,2))]
  header.stop_hms  = [fix(strmid(raw_data[6],0,2)), $
                      fix(strmid(raw_data[6],2,2)), $
                      (strlen(raw_data[6]) eq 4) ? $
                      0 : fix(strmid(raw_data[6],4,2))]
  header.object = raw_data[7]
  header.storm_type = raw_data[8]
  header.frequency = float(strmid(raw_data[9],0,4))
  header.file_name = raw_data[10]

  if strmid(header.file_name,strlen(header.file_name)-3) eq 'mp3' then begin
    wav_file_name = strmid(header.file_name,0,strlen(header.file_name)-3)+'wav'
    spawn,'ffmpeg -y -i '+dat_datapath+header.file_name+' '+dat_datapath+wav_file_name
  endif else wav_file_name = header.file_name

  wav_data = read_wav(dat_datapath+wav_file_name,rate)
  if (size(wav_data))[0] eq 2 then nchannel = 2b else nchannel = 1b
  ndata = n_elements(wav_data)/nchannel

  ; BUILDING EMPTY DATA STRUCTURE
  data = replicate({radiojove_wav_data},ndata)

  ; FILLING DATA STRUCTURE
  data.time = dindgen(ndata)/rate
  if (size(wav_data))[0] eq 2 then begin
    data.chan1 = reform(wav_data[0,*])
    data.chan2 = reform(wav_data[1,*])
  endif else begin
    data.chan1 = wav_data
    data.chan2 = 0
  endelse

  ; BUILDING TIME VARIABLES
  time_epoch = dcomplexarr(ndata)
  start_jd = julday(header.start_ymd[1],header.start_ymd[2],header.start_ymd[0],$
                    header.start_hms[0],header.start_hms[1],header.start_hms[2])
  time_jd = data.time/86400d0 + start_jd
  time_iso = strarr(ndata)
  tmp0 = intarr(8)
  for i=0l,ndata-1 do begin
    data_day = fix(data[i].time/86400.)
    data_hrs = fix((data[i].time mod 86400.d0)/3600.d0)
    data_min = fix((data[i].time mod 3600.d0)/60.d0)
    data_sec = fix(data[i].time mod 60.d0)
    data_mmm = fix((data[i].time mod 1.d0)*1d3)
    data_uuu = fix((data[i].time mod 1.d-3)*1d6)
    data_nnn = fix((data[i].time mod 1.d-6)*1d9)
    data_ppp = fix((data[i].time mod 1.d-9)*1d12)
    tmp1 = [data_day,data_hrs,data_min,data_sec,data_mmm,$
          data_uuu,data_nnn,data_ppp]
    test=0b
    for k=0,7 do if tmp0[k] gt tmp1[k] then test=1b

    if test eq 1b then begin
      print,'###### Time error at step ',i
      print,string(format='(I2.2,"T",I2.2,":",I2.2,":",I2.2,".",I3.3,".",I3.3,'+ $
          '".",I3.3,".",I3.3)',tmp0)
      print,string(format='(I2.2,"T",I2.2,":",I2.2,":",I2.2,".",I3.3,".",I3.3,'+ $
          '".",I3.3,".",I3.3)',tmp1)
    endif

    cdf_epoch16,epoch_tmp,header.start_ymd[0],header.start_ymd[1], $
                          header.start_ymd[2]+data_day, $
                          header.start_hms[0]+data_hrs, $
                          header.start_hms[1]+data_min, $
                          header.start_hms[2]+data_sec, $
                          data_mmm,data_uuu,data_nnn,data_ppp,/compute
    time_epoch[i] = epoch_tmp
    time_iso[i] = cdf_encode_epoch(epoch_tmp,epoch=3)
  endfor

  ; CONSTRUCTING CDF_DATA STRUCTURE
  case nchannel of
    1:  cdf_data = {epoch:dcomplexarr(ndata), data_1:fltarr(ndata)}
    2:  cdf_data = {epoch:dcomplexarr(ndata), data_1:fltarr(ndata),data_2:fltarr(ndata)}
  endcase

  ; FILLING CDF_DATA STRUCTURE
  cdf_data.epoch=time_epoch
  cdf_data.data_1=data.chan1
  if nchannel eq 2 then cdf_data.data_2=data.chan2

  ; SOME STATUS OUTPUT
  print,"File    = "+files[j]
  print,"N_data  = "+string(ndata)
  print,"N_chann = "+string(nchannel)
  print,"T_start = "+time_iso[0]
  print,"T_stop  = "+time_iso[ndata-1]

  ; PREPARING CDF FILE
  STR_TIME_START = time_iso[0]
  STR_TIME_STOP  = time_iso[ndata-1]

  STR_AJDHM_BEG = strjoin([strsplit(strmid(STR_TIME_START,0,10),'-',/extract), $
                  strmid(STR_TIME_START,11,2),strmid(STR_TIME_START,14,2)])
  STR_AJDHM_END = strjoin([strsplit(strmid(STR_TIME_STOP,0,10),'-',/extract), $
                  strmid(STR_TIME_STOP,11,2),strmid(STR_TIME_STOP,14,2)])

  ; PREPARING CDF GLOBAL ATTRIBUTES
  CDF_GATTR = {OBSERVATION_START_TIME:"", OBSERVATION_STOP_TIME:"" ,$
               TIME_MIN:0.0d0, TIME_MAX:0.0d0, TIME_SAMPLING_MIN:0.0, $
               TIME_SAMPLING_MAX:0.0d0, $
               spectral_range_min:0., spectral_range_max:0., $
               DATA_VERSION:"", SKELETON_VERSION:"", SOFTWARE_VERSION:"", $
               INSTRUMENT_NAME:"", INSTRUMENT_HOST_NAME:"", $
               Observation_target:"", TARGET_CLASS:"", $
               TARGET_REGION:"", TARGET_ELEMENT:"", $
               Observer_name:"", Observatory_location:"", $
               Observatory_latitude:0.d0, Observatory_longitude:0.d0, $
               original_filename:"", N_CHANNEL:0b}

  ; SETTING PDS GLOBAL ATTRIBUTES
  CDF_GATTR.OBSERVATION_START_TIME = STR_TIME_START
  CDF_GATTR.OBSERVATION_STOP_TIME = STR_TIME_STOP
  CDF_GATTR.OBSERVATION_TARGET = strupcase(strmid(header.object,0,1))+ $
                 strlowcase(strmid(header.object,1))

  ; SETTING RADIOJOVE GLOBAL ATTRIBUTES
  CDF_GATTR.OBSERVER_NAME = header.first_name+' '+header.last_name
  CDF_GATTR.ORIGINAL_FILENAME = header.file_name
  CDF_GATTR.OBSERVATORY_LOCATION = " "
  CDF_GATTR.OBSERVATORY_LATITUDE = 0.d0
  CDF_GATTR.OBSERVATORY_LONGITUDE = 0.d0
  CDF_GATTR.N_CHANNEL = nchannel

  ; SETTING VESPA GLOBAL ATTRIBUTES
  CDF_GATTR.TIME_MIN = time_jd[0]
  CDF_GATTR.TIME_MAX = time_jd[ndata-1]
  CDF_GATTR.TIME_SAMPLING_MIN= 1./rate
  CDF_GATTR.TIME_SAMPLING_MAX= 1./rate
  CDF_GATTR.spectral_range_min = header.frequency
  CDF_GATTR.spectral_range_max = header.frequency
  CDF_GATTR.INSTRUMENT_HOST_NAME = header.school_obs
  if CDF_GATTR.Observation_target eq "Sun" then begin
    CDF_GATTR.TARGET_CLASS = "star"
    CDF_GATTR.TARGET_REGION = "Solar Wind"
  endif else if CDF_GATTR.Observation_target eq "Jupiter" then begin
    CDF_GATTR.TARGET_CLASS = "planet"
    CDF_GATTR.TARGET_REGION = "Aurora"
  endif else stop,'Unknown target class'
  case nchannel of
    1: CDF_GATTR.INSTRUMENT_NAME = 'RadioJOVE Single Channel'
    2: CDF_GATTR.INSTRUMENT_NAME = 'RadioJOVE Dual Channel'
  endcase
  if header.storm_type ne "" then begin
    CDF_GATTR.TARGET_ELEMENT = header.storm_type
  endif else begin
    CDF_GATTR.TARGET_ELEMENT = " "
  endelse

  CDF_GATTR.DATA_VERSION = strmid(dat_version,0,1)+'.'+strmid(dat_version,1)
  CDF_GATTR.SKELETON_VERSION = strmid(cdf_version,0,1)+'.'+strmid(cdf_version,1)
  CDF_GATTR.SOFTWARE_VERSION = strmid(sft_version,0,1)+'.'+strmid(sft_version,1)

  ; PREPARING CDF VARIABLE ATTRIBUTES
  CDF_VATTR = {EPOCH:{VALIDMIN:0.D0,VALIDMAX:0.D0,SCALEMIN:0.D0,SCALEMAX:0.D0}}

  ; SETTING CDF EPOCH VARIABLE ATTRIBUTES
  CDF_VATTR.EPOCH.VALIDMIN = cdf_data.epoch[0]
  CDF_VATTR.EPOCH.VALIDMAX = cdf_data.epoch[ndata-1]
```

```
  CDF_VATTR.EPOCH.SCALEMIN = floor(cdf_data.epoch[0] / 21.6D6)*21.6D6
  CDF_VATTR.EPOCH.SCALEMAX = ceil(cdf_data.epoch[ndata-1] / 21.6D6)*21.6D6


  ; BUILDING CDF FILENAME
  case nchannel of
    1: begin
      cdf_file_name = cdf_datapath+'data/radiojove_sts_'+STR_AJDHM_BEG+'_'+ $
                      STR_AJDHM_END+'_v'+sft_version+'.cdf'
      master_cdf = master_cdf_sts
    end
    2: begin
      cdf_file_name = cdf_datapath+'data/radiojove_dts_'+STR_AJDHM_BEG+'_'+ $
                      STR_AJDHM_END+'_v'+sft_version+'.cdf'
      master_cdf = master_cdf_dts
    end
  endcase

  ; DATA PERIOD OUTPUT
  printf,lun1,"    "+strmid(time_iso[0],0,10) + string(format='(" ",I5,"  ",A, $
              "  ",A)',ndata,strmid(time_iso[0],0,16),strmid(time_iso[ndata-1],0,16))

  ; CLEANING PREVIOUS CDF FILE
  print,'deleting previous CDF file'
  spawn,'rm -rf '+cdf_file_name

  ; BUILDING AND FILLING CDF FILE
  make_cdf,master_cdf,cdf_file_name,cdf_data,GATTR=CDF_GATTR,VATTR=CDF_VATTR;,/VERBOSE

  ; CDF SCRIPT TRICK TO REMOVE UNUSED RECORDS
  spawn,'cdfconvert -d '+cdf_file_name+' '+cdf_file_name

  ; CDF SCRIPT TO CHECK PDS4 COMPATIBILITY
  print,'checking PDS4 compliance:'
  spawn,'/Users/baptiste/Development/pds-cdf/bin/cdfcheck -v '+cdf_file_name
  spawn,'/Users/baptiste/Development/pds-cdf/bin/cdfcheck '+cdf_file_name,txt_result
  printf,lun2,header.file_name+': '+txt_result

endfor
; ==============================================================================
; END MAIN LOOP
; ==============================================================================


; CLOSING HOUSEKEEPING FILES
close,lun1
close,lun2
free_lun,lun1
free_lun,lun2

end
```

make_radiojove_sp_cdf_v02.pro

```
; IDL Version 7.0, Mac OS X (darwin i386 m32)
; Journal File for baptiste@macbookbc.obspm.fr
; Working directory: /Users/baptiste/Projets/CDPP/Archivage/Cassini:RPWS/HFR/SND
; Date: Thu Jun 12 12:04:25 2014
;
; RUN @compile_IDLmakecdf
;
; NB: compile_IDLmakecdf.pro script is part of the CDF CDAWlib distribution
; available here: http://spdf.gsfc.nasa.gov/CDAWlib.html
; DECLARING VERSIONS
sft_version = '02'
cdf_version = '02'
dat_version = '01'

; DECLARING PATHS
cdf_datapath = 'data/cdf/V'+sft_version+'/'
dat_datapath = 'data/dat/V'+dat_version+'/spectrogram/'
master_cdf_sds = cdf_datapath+'master/radiojove_sds_000000000000_000000000000_v'+ $
                 cdf_version+'.cdf'
skelet_cdf_sds = cdf_datapath+'master/radiojove_sds_000000000000_000000000000_v'+ $
                 cdf_version+'.skt'
master_cdf_dds = cdf_datapath+'master/radiojove_dds_000000000000_000000000000_v'+ $
                 cdf_version+'.cdf'
skelet_cdf_dds = cdf_datapath+'master/radiojove_dds_000000000000_000000000000_v'+ $
                 cdf_version+'.skt'

; CLEANING AND REBUILDING MASTER CDF
spawn,'rm -f '+master_cdf_sds
spawn,'skeletoncdf -cdf '+master_cdf_sds+' '+skelet_cdf_sds
spawn,'rm -f '+master_cdf_dds
spawn,'skeletoncdf -cdf '+master_cdf_dds+' '+skelet_cdf_dds

; OPENING 2 FILES FOR HOUSEKEEPING OUTPUT
openw,lun1,cdf_datapath+'misc/interval_listing.txt',/get_lun
openw,lun2,cdf_datapath+'misc/cdf_pds_check.txt',/get_lun

; BUILDING LIST OF DATA FILES
files = file_search(dat_datapath+"*.sp[sd]")

; ============================================================================
; LOOPING ON DATA FILES
; ============================================================================

for j=0,N_ELEMENTS(files)-1 do begin

  ; LOADING DATA
  data = read_radiojove_sp_file(files[j],header,frequency)

  ; BUILDING TIME VARIABLES
  time_epoch = dcomplexarr(header.nsweep)
  time_jd = dindgen(header.nsweep)/header.nsweep*(header.stop_jdtime-$
                 header.start_jdtime)+header.start_jdtime
  time_iso = strarr(header.nsweep)
  for i=01,header.nsweep-1 do begin
    caldat, time_jd[i], MO, DD, YY, HH, MI, SS
    cdf_epoch16,epoch_tmp,YY, MO, DD, HH, MI, SS,0,0,0,0,/compute
    time_epoch[i] = epoch_tmp
    time_iso[i] = cdf_encode_epoch(epoch_tmp,epoch=3)
  endfor


  ; CONSTRUCTING CDF_DATA STRUCTURE
  case header.nchan of
    1: cdf_data = {epoch:dcomplexarr(header.nsweep), $
            data_1:uintarr(header.nfreq,header.nsweep), frequency:fltarr(header.nfreq)}
    2: cdf_data = {epoch:dcomplexarr(header.nsweep), $
            data_1:uintarr(header.nfreq,header.nsweep), $
            data_2:uintarr(header.nfreq,header.nsweep), frequency:fltarr(header.nfreq)}
  endcase

  ; FILLING CDF_DATA STRUCTURE
  cdf_data.epoch=time_epoch
  cdf_data.data_1=reform(data[0,*,*])
  if header.nchan eq 2 then cdf_data.data_2=reform(data[1,*,*])
```

```
cdf_data.frequency=frequency

; SOME STATUS OUTPUT
print,"File    = "+files[j]
print,"N_data  ="+string(header.nsweep)
print,"N_freq  ="+string(header.nfreq)
print,"N_chan  ="+string(header.nchan)
print,"T_start = "+time_iso[0]
print,"T_stop  = "+time_iso[header.nsweep-1]

; PREPARING CDF FILE
STR_TIME_START = time_iso[0]
STR_TIME_STOP  = time_iso[header.nsweep-1]

STR_AJDHM_BEG = strjoin([strsplit(strmid(STR_TIME_START,0,10),'-',/extract), $
               strmid(STR_TIME_START,11,2),strmid(STR_TIME_START,14,2)])
STR_AJDHM_END = strjoin([strsplit(strmid(STR_TIME_STOP,0,10),'-',/extract), $
               strmid(STR_TIME_STOP,11,2),strmid(STR_TIME_STOP,14,2)])

; PREPARING CDF GLOBAL ATTRIBUTES
CDF_GATTR = {OBSERVATION_START_TIME:"", OBSERVATION_STOP_TIME:"" ,$
             TIME_MIN:0.0d0, TIME_MAX:0.0d0, TIME_SAMPLING_MIN:0.0, $
             TIME_SAMPLING_MAX:0.0d0, $
             spectral_range_min:0.0,  spectral_range_max:0.0, $
             spectral_sampling_step_min:0.0,       spectral_sampling_step_max:0.0, $
             spectral_resolution_min:0.0,    spectral_resolution_max:0.0, $
             DATA_VERSION:"", SKELETON_VERSION:"", SOFTWARE_VERSION:"", $
             INSTRUMENT_NAME:"", INSTRUMENT_HOST_NAME:"", $
             TARGET_NAME:"", TARGET_CLASS:"", $
             TARGET_REGION:"", TARGET_ELEMENT:"", $
             Observer_name:"", Observatory_location:"", $
             Observatory_latitude:0.d0, Observatory_longitude:0.d0, $
             original_filename:"", N_CHANNEL:0b}

; SETTING PDS GLOBAL ATTRIBUTES
CDF_GATTR.OBSERVATION_START_TIME = STR_TIME_START
CDF_GATTR.OBSERVATION_STOP_TIME = STR_TIME_STOP

; SETTING VESPA GLOBAL ATTRIBUTES
CDF_GATTR.TIME_MIN = time_jd[0]
CDF_GATTR.TIME_MAX = time_jd[header.nsweep-1]
CDF_GATTR.TIME_SAMPLING_MIN= min(time_jd[1:header.nsweep-1]-$
               time_jd[0:header.nsweep-2])*86400.
CDF_GATTR.TIME_SAMPLING_MAX= max(time_jd[1:header.nsweep-1]-$
               time_jd[0:header.nsweep-2])*86400.
CDF_GATTR.spectral_range_min = min(frequency)
CDF_GATTR.spectral_range_max = max(frequency)
CDF_GATTR.spectral_sampling_step_min = min(frequency[1:*]-frequency)
CDF_GATTR.spectral_sampling_step_max = max(frequency[1:*]-frequency)
case header.nchan of
  1: CDF_GATTR.INSTRUMENT_NAME = 'RadioJOVE Single Channel Spectrograph'
  2: CDF_GATTR.INSTRUMENT_NAME = 'RadioJOVE Dual Channel Spectrograph'
endcase
CDF_GATTR.INSTRUMENT_HOST_NAME = header.name
CDF_GATTR.TARGET_NAME = "Jupiter"
CDF_GATTR.TARGET_CLASS = "planet"
CDF_GATTR.TARGET_REGION = "Aurora"
CDF_GATTR.TARGET_ELEMENT = "Io-B DAM"

CDF_GATTR.DATA_VERSION = strmid(dat_version,0,1)+'.'+strmid(dat_version,1)
CDF_GATTR.SKELETON_VERSION = strmid(cdf_version,0,1)+'.'+strmid(cdf_version,1)
CDF_GATTR.SOFTWARE_VERSION = strmid(sft_version,0,1)+'.'+strmid(sft_version,1)

; SETTING RADIOJOVE GLOBAL ATTRIBUTES
CDF_GATTR.OBSERVER_NAME = header.author
CDF_GATTR.OBSERVATORY_LOCATION = header.location
```

```
  CDF_GATTR.OBSERVATORY_LATITUDE = header.latitude
  CDF_GATTR.OBSERVATORY_LONGITUDE = header.longitude
  CDF_GATTR.ORIGINAL_FILENAME = header.file_name
  CDF_GATTR.N_CHANNEL = header.nchan

  ; PREPARING CDF VARIABLE ATTRIBUTES
  CDF_VATTR = {EPOCH:{VALIDMIN:0.D0,VALIDMAX:0.D0,SCALEMIN:0.D0,SCALEMAX:0.D0}}

  ; SETTING CDF EPOCH VARIABLE ATTRIBUTES
  CDF_VATTR.EPOCH.VALIDMIN = cdf_data.epoch[0]
  CDF_VATTR.EPOCH.VALIDMAX = cdf_data.epoch[header.nsweep-1]
  CDF_VATTR.EPOCH.SCALEMIN = floor(cdf_data.epoch[0] / 21.6D6)*21.6D6
  CDF_VATTR.EPOCH.SCALEMAX = ceil(cdf_data.epoch[header.nsweep-1] / 21.6D6)*21.6D6

  ; BUILDING CDF FILENAME
  case header.nchan of
    1: begin
      cdf_file_name = cdf_datapath+'data/radiojove_sds_'+STR_AJDHM_BEG+'_'+ $
                   STR_AJDHM_END+'_v'+sft_version+'.cdf'
      master_cdf = master_cdf_sds
    end
    2: begin
      cdf_file_name = cdf_datapath+'data/radiojove_dds_'+STR_AJDHM_BEG+'_'+ $
                   STR_AJDHM_END+'_v'+sft_version+'.cdf'
      master_cdf = master_cdf_dds
    end
  endcase

  ; DATA PERIOD OUTPUT
  printf,lun1,"    "+strmid(time_iso[0],0,10) + string(format='(" ",I5,"  ",A, $
              "  ",A)',header.nsweep,strmid(time_iso[0],0,16), $
              strmid(time_iso[header.nsweep-1],0,16))

  ; CLEANING PREVIOUS CDF FILE
  print,'deleting previous CDF file'
  spawn,'rm -rf '+cdf_file_name

  ; BUILDING AND FILLING CDF FILE
  make_cdf,master_cdf,cdf_file_name,cdf_data,GATTR=CDF_GATTR,VATTR=CDF_VATTR;,/VERBOSE

  ; CDF SCRIPT TRICK TO REMOVE UNUSED RECORDS
  spawn,'cdfconvert -d '+cdf_file_name+' '+cdf_file_name

  ; CDF SCRIPT TO CHECK PDS4 COMPATIBILITY
  print,'checking PDS4 compliance:'
  spawn,'/Users/baptiste/Development/pds-cdf/bin/cdfcheck -v '+cdf_file_name
  spawn,'/Users/baptiste/Development/pds-cdf/bin/cdfcheck '+cdf_file_name,txt_result
  printf,lun2,cdf_file_name+': '+txt_result

endfor
; =============================================================================
; END MAIN LOOP
; =============================================================================


; CLOSING HOUSEKEEPING FILES
close,lun1
close,lun2
free_lun,lun1
free_lun,lun2

end
```

make_cdf.pro

```
FUNCTION cdf_getattval, cdf_file, attname_or_number, EntryNum, $
                                       cdf_type=cdf_type, $
                                       ZVARIABLE=ZVARIABLE

;+
; NAME:
;   cdf_getattval
;
; PURPOSE:
;   Get an attribute value.
;
; CATEGORY:
;   I/O
;
; CALLING SEQUENCE:
;   cdf_getattval,cdf_file,attname_or_number
;
; INPUTS:
;   cdf_file                    - cdf file to read.
;   attname_or_number - String or integer of the attribute.
;   EntryNum - The entry number as defined in CDF_ATTGET procedure.
;
; OPTIONAL INPUTS:
;   None.
;
; KEYWORD PARAMETERS:
;   /ZVARIABLE - Specify if the variable for the
;                          current attribute is a ZVARIABLE type.
;
; OUTPUTS:
;   attval - Value of the attribute.
;
; OPTIONAL OUTPUTS:
;   None.
;
; COMMON BLOCKS:
;   None.
;
; SIDE EFFECTS:
;   None.
;
; RESTRICTIONS/COMMENTS:
;   none.
;
; CALL:
;   None.
;
; EXAMPLE:
;   None.
;
; MODIFICATION HISTORY:
;   Written by X.Bonnin (LESIA, CNRS)
;
;-

attval=0b & cdf_type=''
if (n_params() lt 2) then begin
    message,/INFO,'Usage:'
    print,'attval = get_attval(cdf_file, attname_or_number, EntryNum, $'
    print,'                             cdf_type=cdf_type, /ZVARIABLE)'
    return,0b
endif
if not (keyword_set(EntryNum)) then EntryNum = 0
```

```
ZVARIABLE=keyword_set(ZVARIABLE)

cdfid = cdf_open(cdf_file, /READONLY)
if (cdf_attexists(cdfid, attname_or_number)) then begin
    cdf_attget, cdfid, attname_or_number, EntryNum, attval , $
                        cdf_type=cdf_type, ZVARIABLE=ZVARIABLE
endif
cdf_close,cdfid

return, attval
END
; ================================
; ================================
PRO cdf_getvar,cdf_file,rvar,zvar

;+
; NAME:
;   cdf_getvar
;
; PURPOSE:
;   Get information about the variables
;   defined in a cdf file.
;
; CATEGORY:
;   I/O
;
; CALLING SEQUENCE:
;   cdf_getvar,cdf_file,rvar,zvar
;
; INPUTS:
;   cdf_file - cdf file to read.
;
; OPTIONAL INPUTS:
;   None.
;
; KEYWORD PARAMETERS:
;   None.
;
; OUTPUTS:
;   rvar - Structure containing information about the regular variables.
;   zvar - Structure containing information about the Z variables.
;
; OPTIONAL OUTPUTS:
;   None.
;
; COMMON BLOCKS:
;   None.
;
; SIDE EFFECTS:
;   None.
;
; RESTRICTIONS/COMMENTS:
;   none.
;
; CALL:
;   None.
;
; EXAMPLE:
;   None.
;
; MODIFICATION HISTORY:
;   Written by X.Bonnin (LESIA, CNRS)
;
;-

if not (keyword_set(cdf_file)) then begin
```

```
    message,/INFO,'Usage:'
    print,'cdf_getvar,cdf_file,rvar,zvar'
    return
endif

if not (file_test(cdf_file)) then message,'ERROR - Input cdf_file does not exist:
'+cdf_file+'!'

cdfid=cdf_open(cdf_file,/READONLY)

inq=cdf_inquire(cdfid)
nvars=inq.nvars
nzvars=inq.nzvars

if (nvars gt 0) then begin
    rvar={name:'',id:0l,datatype:'',numelem:0l, $
          recvar:'',dimvar:''}
    rvar=replicate(rvar,nvars)
    for i=0,nvars-1 do begin
        vinq_i=cdf_varinq(cdfid,i)
        rvar[i].name=vinq_i.name
        rvar[i].id=i
        rvar[i].datatype=vinq_i.datatype
        rvar[i].numelem=vinq_i.numelem
        rvar[i].recvar=vinq_i.recvar
        rvar[i].dimvar=strjoin(strtrim(vinq_i.dimvar,2),',')
    endfor
endif

if (nzvars gt 0) then begin
    zvar={name:'',id:0l,datatype:'',numelem:0l, $
          recvar:'',dimvar:'',dim:''}
    zvar=replicate(zvar,nzvars)
    for i=0,nzvars-1 do begin
        vinq_i=cdf_varinq(cdfid,i,/ZVAR)
        zvar[i].name=vinq_i.name
        zvar[i].id=i
        zvar[i].datatype=vinq_i.datatype
        zvar[i].numelem=vinq_i.numelem
        zvar[i].recvar=vinq_i.recvar
        zvar[i].dimvar=strjoin(strtrim(vinq_i.dimvar,2),',')
        zvar[i].dim=strjoin(strtrim(vinq_i.dim,2),',')
    endfor
endif
cdf_close,cdfid

END
; ================================
; ================================
PRO cdf_getatt,cdf_file,gattrs,vattrs

;+
; NAME:
;   cdf_getatt
;
; PURPOSE:
;   Get information about the global and variable
;   attributes defined in a cdf file.
;
; CATEGORY:
;   I/O
;
; CALLING SEQUENCE:
;   cdf_getatt,cdf_file,gattrs,vattrs
;
; INPUTS:
```

```
;   cdf_file - cdf file to read.
;
; OPTIONAL INPUTS:
;   None.
;
; KEYWORD PARAMETERS:
;   None.
;
; OUTPUTS:
;   gattrs - Structure containing information about the global attributes.
;   vattrs - Structure containing information about the variable attributes.
;
; OPTIONAL OUTPUTS:
;   None.
;
; COMMON BLOCKS:
;   None.
;
; SIDE EFFECTS:
;   None.
;
; RESTRICTIONS/COMMENTS:
;   none.
;
; CALL:
;   None.
;
; EXAMPLE:
;   None.
;
; MODIFICATION HISTORY:
;   Written by X.Bonnin (LESIA, CNRS)
;
;-

if not (keyword_set(cdf_file)) then begin
    message,/INFO,'Usage:'
    print,'cdf_getatt,cdf_file,gattrs,vattrs'
    return
endif

if not (file_test(cdf_file)) then message,'ERROR - Input cdf_file does not exist:
'+cdf_file+'!'

cdfid=cdf_open(cdf_file,/READONLY)

inq=cdf_inquire(cdfid)
natts=inq.natts

if (natts gt 0) then begin
    gattrs={name:'',id:0l,scope:'',maxentry:0l,maxzentry:0l}
    vattrs={name:'',id:0l,scope:'',maxentry:0l,maxzentry:0l}
    gattrs=replicate(gattrs,natts)
    vattrs=replicate(vattrs,natts)
    gcount=0 & vcount=0
    for i=0,natts-1 do begin
        cdf_attinq,cdfid,i,name_i,scope_i,maxentry_i,maxzentry_i
        if (scope_i eq 'GLOBAL_SCOPE') then begin
            gattrs[gcount].name=name_i
            gattrs[gcount].id=i
            gattrs[gcount].scope=scope_i
            gattrs[gcount].maxentry=maxentry_i
            gattrs[gcount].maxzentry=maxzentry_i
            gcount++
        endif else begin
            vattrs[vcount].name=name_i
            vattrs[vcount].id=i
            vattrs[vcount].scope=scope_i
            vattrs[vcount].maxentry=maxentry_i
            vattrs[vcount].maxzentry=maxzentry_i
            vcount++
        endelse
    endfor
    if (gcount gt 0) then gattrs=gattrs[0:gcount-1]
    if (vcount gt 0) then vattrs=vattrs[0:vcount-1]
endif
cdf_close,cdfid

END
; ================================
; ================================
PRO make_cdf,master_cdf,output_cdf,variables, $
            vattributes=vattributes, $
            gattributes=gattributes, $
            VERBOSE=VERBOSE

;+
; NAME:
;   make_cdf
;
; PURPOSE:
;   Produces a CDF format file from a
;   given master cdf file.
;
; CATEGORY:
;   I/O
;
; CALLING SEQUENCE:
;   make_cdf,master_cdf,output_cdf,variables
;
; INPUTS:
;   master_cdf  - Path of the template cdf file.
;   output_cdf  - Path and name of the output cdf file.
;   variables   - Structure providing data values
;                 to be written into the output_cdf file,
;                 for each cdf variable defined in the master file.
;                 Structure contains the following item:
;                       .NAME_OF_CDF_VARIABLE
;                 , where "NAME_OF_CDF_VARIABLE" is the name of the cdf variable for
;                 which data must be written.
;
; OPTIONAL INPUTS:
;   vattributes - Structure providing variable attribute values
;                 to be written into the output_cdf file,
;                 for one or more cdf variable defined in the master file.
;                 Structure contains the following item:
;                       .NAME_OF_CDF_VARIABLE.NAME_OF_VATTRIBUTE
;                 , where "NAME_OF_VATTRIBUTE" is the name of the variable attribute
;                 to edit, and "NAME_OF_CDF_VARIABLE" is the name of the corresponding
;                 cdf variable.
;
;   gattributes - Structure containing global attribute values to be written into the
;                 output cdf. If it is not provided, then the global attribute values
;                 will be copied from the master to the output cdf, and the
;                 GENERATION_DATE global attribute will be update to the current date
;                 and time in the output cdf. Structure contains the following item:
;                       .NAME_OF_GATTRIBUTE
;                 , where "NAME_OF_GATTRIBUTE" is name of the global attribute to edit.
;
; KEYWORD PARAMETERS:
;   /VERBOSE - Verbose mode.
;
```

```
; OUTPUTS:
;   The routine will produce a cdf format file.
;
; OPTIONAL OUTPUTS:
;   None.
;
; COMMON BLOCKS:
;   None.
;
; SIDE EFFECTS:
;   None.
;
; RESTRICTIONS/COMMENTS:
;   Only works with Z variables.
;
;   Make sure that the IDL CDWAlib
;   (http://spdf.gsfc.nasa.gov/CDAWlib.html)
;   is correctly installed, and that the
;   IDLmakecdf routines are compiled
;   (the routine called compile_IDLmakecdf.pro
;   in CDAWlib directory
;   can be used for that purpose.)
;
; CALL:
;   cdf_getatt
;   cdf_getvar
;   read_master_cdf
;   write_data_to_cdf
;
; EXAMPLE:
;   None.
;
; MODIFICATION HISTORY :
;   Written by X.Bonnin (LESIA, CNRS)
;
;   24-MAR-2014, X.Bonnin (LESIA, CNRS):
;
;                 Rename var_values input structure to variables.
;                 Add the possibility to modify global/variable attribute values
;                 in the output cdf file, using vattributes and gattributes optional
;                 inputs.
;                 Remove results optional output.
;
;-

if (n_params() lt 2) then begin
    message,/INFO,'Usage:'
    print,'make_cdf,master_cdf_output_cdf,variables, $'
    print,'         gattributes=gattributes, $'
    print,'         vattributes=vattributes, $'
    print,'         /VERBOSE'
    return
endif
VERBOSE=keyword_set(VERBOSE)

pref='['+output_cdf+']: '

; Retrieve the variables in master_cdf and copy master_cdf to output_cdf
if (VERBOSE) then print,'Reading '+master_cdf+' and creating '+output_cdf+'...'
if not (file_test(master_cdf)) then $
    message,'ERROR - Master cdf file ('+master_cdf+') does not exist!'
buf1=read_master_cdf(master_cdf,output_cdf)
if not (file_test(output_cdf)) then $
    message,'ERROR - Output cdf file ('+output_cdf+') does not exist!'
nvar=n_tags(variables)
```

```
; Get list of variables in output_cdf
cdf_getvar,output_cdf,rvar,zvar
nzvar=n_elements(zvar)

; Get list of global and variable attributes in output_cdf
cdf_getatt,output_cdf,gattrs,vattrs
ngatt=n_elements(gattrs) & nvatt=n_elements(vattrs)

; Fill variables with input data (and update variable attributes if required)
if (nvar gt 0) and (nzvar gt 0) then begin
    buf1_varnames=strupcase(tag_names(buf1))
    varnames=strupcase(tag_names(variables))

    for i=0l,nvar-1l do begin
        where_i=(where(varnames[i] eq buf1_varnames))[0]
        if (where_i eq -1) then begin
            if (VERBOSE) then message,/CONT,pref+'Warning - '+$
                    varnames[i]+' variable not found in the master cdf!'
            continue
        endif
        if (VERBOSE) then print,pref+'Copying values of '+strtrim(varnames[i],2)+ $
            ' cdf variable...'
        *buf1.(where_i).data=variables.(i)
    endfor
endif
isdata=write_data_to_cdf(output_cdf,buf1,/DEBUG)
isok=(file_test(output_cdf)) and (isdata)
if (VERBOSE) then begin
   if (isok) then print,'Input data have been correctly written into '+output_cdf $
   else message,'ERROR - Input data have not been correctly written into '+output_cdf+'!'
endif

; Modify variable attributes in output_cdf
if (keyword_set(vattributes)) and (nvatt gt 0) and (nzvar gt 0) then begin
    varnames=strupcase(tag_names(vattributes))

    for i=0,n_elements(varnames)-1 do begin
        where_var=(where(varnames[i] eq strupcase(zvar.name)))[0]
        if (where_var eq -1) then begin
            message,/CONT,'Warning - Variable '+varnames[i]+' not found in '+ $
                output_cdf+'!'
            continue
        endif
        vattnames=strupcase(tag_names(vattributes.(i)))

        cdfid=cdf_open(output_cdf)
        for j=0,nvatt-1 do begin
            where_j=(where(strupcase(vattrs[j].name) eq vattnames))[0]
            if (where_j ne -1) then begin
                cdf_attput,cdfid,vattrs[j].name,zvar[where_var].name, $
                    vattributes.(i).(where_j)
                if (VERBOSE) then print,pref+'New value for the variable attribute '+ $
                    zvar[where_var].name+'.'+vattrs[j].name+' --> '+ $
                    strtrim(vattributes.(i).(where_j),2)
            endif
        endfor
        cdf_close,cdfid
    endfor
endif

; Modify global attributes in output_cdf
if (keyword_set(gattributes)) and (ngatt gt 0) then begin
    gattnames=strupcase(tag_names(gattributes))
    cdfid=cdf_open(output_cdf)
    for j=0,ngatt-1 do begin
        where_j=(where(strupcase(gattrs[j].name) eq gattnames))[0]
        if (where_j ne -1) then begin
            for k=0,n_elements(gattributes.(where_j))-1 do begin
                cdf_attput,cdfid,gattrs[j].name,k,gattributes.(where_j)[k]
                if (VERBOSE) then print,pref+'New value for the global attribute '+ $
                        gattrs[j].name+'['+strtrim(k,2)+'] --> '+ $
                        strtrim(gattributes.(where_j)[k],2)
            endfor
        endif
    endfor
    cdf_close,cdfid
endif

END
```

radiojove_spd_data__define.pro

```
PRO co_rpws_hfr_snd_l2_data__define
tmp = {co_rpws_hfr_snd_l2_data, START_TIME:0.d0, mode_flag:0b, active_flag:0b, $
       antenna_value:0b,INDEX_SPECTRUM_VALUE:0b, CYCLE_SPECTRUM_VALUE:0b, $
       T1_VALUE:0b, T2_VALUE:0b, T3_VALUE:0b, SPECTRUM_AGC:bytarr(270), $
       SPECTRUM_CALIBRATED:dblarr(270), SPECTRUM_FFT:dblarr(270)}
end
```

radiojove_wav_data__define.pro

```
PRO radiojove_wav_data__define
tmp = {radiojove_wav_data, time:0.d0, chan1:0, chan2:0}
end
```

radiojove_wav_header__define.pro

```
PRO radiojove_wav_header__define
tmp = {radiojove_wav_header, first_name:"", last_name:"", school_obs:"", $
  start_ymd:intarr(3), start_hms:intarr(3), stop_ymd:intarr(3), stop_hms:intarr(3), $
  object:"", storm_type:"", frequency:0., file_name:""}
end
```

read_ascii_file.pro

```
FUNCTION read_ascii_file,file,nlines

openr,lun,file,/get_lun
skip_lun,lun,/eof,/lines,transfer_count=nlines
str_raw_data = strarr(nlines)
point_lun,lun,0
readf,lun,str_raw_data
close,lun
free_lun,lun

return, str_raw_data
end
```

read_radiojove_sp_file.pro

```
PRO radiojove_sp_header__define
tmp = {radiojove_sp_header, sft_version:"", start_jdtime:0.d0, stop_jdtime:0.d0, $
  latitude:0.d0, longitude:0.d0, chartmax:0.d0, chartmin:0.d0, timezone:0, $
  source:"", author:"", name:"", location:"", file_name:"", $
  nchan:0b, nfreq:0, note_length:0, note:"", fmin:0., fmax:0., nsweep:0l}
end

FUNCTION read_radiojove_sp_file,filename,header,freq_table

openr,lun,filename,/get_lun
; reading header
header = {radiojove_sp_header}
header.file_name = filename
case strmid(filename,strlen(filename)-3) of
  'spd' : header.nchan = 2b
  'sps' : header.nchan = 1b
endcase

tmp = bytarr(10)
readu,lun,tmp
header.sft_version = string(tmp)

tmp = 0.d0
readu,lun,tmp
header.start_jdtime = julday(1,1,1901)+tmp-0.5d0
readu,lun,tmp
header.stop_jdtime = julday(1,1,1901)+tmp-0.5d0
readu,lun,tmp
header.latitude = tmp
readu,lun,tmp
header.longitude = tmp
readu,lun,tmp
header.chartmax = tmp
readu,lun,tmp
header.chartmin = tmp

tmp = bytarr(10)
readu,lun,tmp
header.source = string(tmp)

tmp = 0
readu,lun,tmp
header.timezone = tmp

tmp = bytarr(20)
readu,lun,tmp
header.author = string(tmp)
readu,lun,tmp
header.name = string(tmp)

tmp = bytarr(40)
readu,lun,tmp
header.location = string(tmp)

tmp = 0
readu,lun,tmp
header.nfreq = tmp
tmp = 0l
readu,lun,tmp
header.note_length = tmp

tmp = bytarr(header.note_length)
readu,lun,tmp
header.note = string(tmp)


notes = strsplit(strmid(header.note,4,header.note_length-8),string(255b),/extract)

header.nchan=1b
```

```
fs=fstat(lun)
header.nsweep = long((fs.size-fs.cur_ptr)/((header.nfreq+1)*2.))

for i=0,n_elements(notes)-1 do begin
  if strpos(notes(i),'SWEEPS') ne -1 then if header.nsweep ne long(strmid(notes(i),6)) $
    then message,/info,"WARNING: Inconsistent NSWEEP value in HEADER"
  if strpos(notes(i),'LOWF') ne -1 then header.fmin = float(strmid(notes(i),4))
  if strpos(notes(i),'HIF') ne -1 then header.fmax = float(strmid(notes(i),3))
  if strpos(notes(i),'STEPS') ne -1 then if header.nfreq ne long(strmid(notes(i),5)) $
    then message,/info,"WARNING: Inconsistent NCHAN value in HEADER"
  if strpos(notes(i),'DUALSPECFILE') ne -1 then if strmid(notes(i),12) ne 'False' $
    then message,/info,"WARNING: Inconsistent Single/Dual Channel flag"
endfor

freq_table=(header.nfreq-findgen(header.nfreq))/header.nfreq*(header.fmax-header.fmin)+ $
  header.fmin


data=uintarr(header.nchan,header.nfreq,header.nsweep)
tmp = bytarr(2,(header.nfreq+1))
for i=0l,header.nsweep-1 do begin
  readu,lun,tmp
  data[0,*,i] = transpose(tmp[*,0:header.nfreq-1]##[256,1])
endfor

close,lun
free_lun,lun

return,data
end
```