Single Dish Tables

Obit:
Merx mollis mortibus nuper

version: 1.1.1 February 20, 2008

W. D. Cotton

Contents

1 Introduction

This document is intended to define the contents and meaning of the various tables developed for single dish radio astronomical data, especially "On–the–fly" (OTF) data from the Penn Array bolometer camera on the GBT. This software works inside the Obit software package.

Usage documentation uses the doxygen system and is in separate documents.

2 Obit Tables

This document uses latex macroes which are translated by a perl script (bin/PennArrayTables.pl) into the c source code defining the classes used to access these tables. The tables defined in this document are both those needed for the OTF data structure and for reading the GBT archive data files.

2.1 LaTeX Macros

Tables used in Obit are defined in this document by using LaTeX macroes to formally define the table. These macroes are:

- tabletitle{Title of table, e.g. "OTFArrayGeom"}
- tablename{Name of table, e.g. "OTFArrayGeom"}
- tableintro{Short description of class}
- tableover{Overview of usage of class}
- tablekey[{name}{type code}{ software name} {default value} {(range of indices)} {description}]
  Defines Table keyword.
- tablecol[{name}{units}{type code} {(dimensionality)} {software name} {description}]
  Defines Table column..

3 ObitOTF data

The ObitOTF data structure class and related classes are for storing "On–The–Fly" single dish radio astronomy data. The data structure is patterned after the AIPS single dish data format but, due to AIPS table naming restrictions, only a FITS binary table implementation is currently supported.

An ObitOTF data file can be thought of as a relational database. The measured sky brightness measurements are kept in the OTFScanData table together with some auxillary information such as the nominal sky pointing position, time etc. Other auxillary information and calibration and editing information is kept in other tables (see below).

3.1 ObitOTF tables

The basic calibration strategy is to manipulate tables which tell how to transform the data from "raw" data as tabulated to "calibrated" data. There are two basic calibration tables, with the same internal structure. The OTFSoln table is a differential calibration relative to a potential prior calibration. A particular calibration operation determines a OTFSoln table. A OTFCal table is

a cumulative table which is obtained from a (possible) prior calibration corrected by an OTFSoln table.

The tables in an ObitOTF data file include:

- OTFScanData
  Raw sky brightness data and auxillary information.
- OTFArrayGeom
  Table giving the geometric offsets of a feed/detector array from the pointing axis of the telescope.
- OTFTarget
  Table of sources or targets. These are referred to in the OTFScanData table as an index into this table.
- OTFIndex
  Scan table [optional] giving start and stop times and row numbers in the OTFScanData table as well as targets etc. This index is used to improve data access times.
- OTFFlag
  Table describing "flagged" data - data to be ignored.
- OTFCal
  Cumulative calibration table. This table gives multiplicative and additive corrections to the raw sky brightness measurements in the OTFScanData table as well as corrections to the nominal telescope pointing direction.
- OTFSoln
  Differential calibration ("Solution") table.

3.2 ObitOTF Data Structure

Data in the OTFScanData table are stored in table records with a row corresponding to the data obtained in a given integration. The row contains a number of descriptive columns giving time, celestial pointing etc. followed by a column containing a regular data array. The data in the OTFScanData table are all stored as floats to increase access performance.

In general, the data in the data array is a multidimensional array with different quanties along different axes (feed/detector, frequency, polarization). The dimensionality, types and axis values are given in the header of the OTFScanData table as the TDIMn (dimensionality), mCTYPn (axis type), mCDLTn (increment in axis values between pixels), mCRPXn (axis reference pixel), mCROTn (axis rotation angle) and mCRVLn (coordinate on axis at reference pixel) keywords where m is the axis number and n is the data column in the table. (This is the standard FITS convention for conveying this information.) The following illustration is for the Penn Array with one Stokes (total power), 64 detectors (FEED) and 1 frequency; the data column is number 9.

```
TDIM9   = '(1,64,1)'           / size of the multidimensional array
TZERO1  =   2.452814500000E+06 / Offset of Date from JD
1CTYP9  = 'STOKES  '           / Stokes axis
2CTYP9  = 'FEED    '           / Feed/detector axis
3CTYP9  = 'FREQ    '           / Frequency axis
1CDLT9  =         1.000000E+00 / Stokes ``increment''
```

```
2CDLT9  =         1.000000E+00 / Feed ‘‘increment’’
3CDLT9  =         1.000000E+00 / Frequency increment (Hz)
1CRPX9  =         1.000000E+00 / Stokes reference pixel
2CRPX9  =         1.000000E+00 / Feed reference pixel
3CRPX9  =         1.000000E+00 / Frequency reference pixel
1CROT9  =         0.000000E+00 / Stokes ‘‘rotation’’ (no meaning)
2CROT9  =         0.000000E+00 / Feed ‘‘rotation’’ (no meaning)
3CROT9  =         0.000000E+00 / Frequency ‘‘rotation’’ (no meaning)
1CRVL9  =   1.0000000000000E+00 / Stokes ’I’
2CRVL9  =   1.0000000000000E+00 / ‘‘Feed’’ 1
3CRVL9  =   9.0000000000000E+10 / Frequency in Hz
```

4 ObitSD software

The high level view of the Obit system is included in file OBITdoc.ps. The class documentation for the software for processing ObitOTF data is derived from the source code using doxygen and is available in html format starting at doc/doxygen/html/index.html.

5 Building ObitSD

The ObitSD package is an addon to the basic Obit package which should be installed first. See OBITdoc.ps for details.

ObitSD comes with a configure script to construct the Makefiles to build ObitSD. Note: there are a number of third party packages as well as basic Obit that should be installed first. The basic installation is thus:

```
% setenv OBIT /where/ever/you/installed/Obit
% gtar xzvf ObitSD1.0.tgz
% cd ObitSD
% ./configure
% make
```

For “/where/ever/you/installed/Obit” substitute the actual path of the Obit base directory. Alternatively, use the “–with-obit=DIR” option with configure. In addition to the packages used by basic Obit, ObitSD uses the GSL (GNU Scientific Library) package. The location of GSL can be specified to configure with the ”–with-gsl=DIR” configure option if configure cannot find it.

When using ObitSD from python, specify the PYTHONPATH environment variable as “ObitSD/python : Obit/python” where for ObitSD and Obit substitute the base directories of the ObitSD and Obit packages.

6 Class ObitTableOTFArrayGeom

ObitTableOTFArrayGeom Class

6.1 OTFArrayGeom

Table name: OTFArrayGeom

6.1.1 Introduction

[This class contains tabular data and allows access. "OTFArrayGeom" contains information about the locations and characteristics of detectors in the camera, the location of the telescope and time related information.]

6.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc.

6.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"TELEX ": Type: D**, Variable name: TeleX,
  Default value: 0.0, Range:
  Telescope X coordinate. (meters, earth center)
- **"TELEY ": Type: D**, Variable name: TeleY,
  Default value: 0.0, Range:
  Telescope Y coordinate. (meters, earth center)
- **"TELEZ ": Type: D**, Variable name: TeleZ,
  Default value: 0.0, Range:
  Telescope Z coordinate. (meters, earth center)
- **"RDATE ": Type: A**, Variable name: RefDate,
  Default value: "YYYYMMDD", Range:
  Reference date as "YYYYMMDD"
- **"DEGPDY ": Type: D**, Variable name: DegDay,
  Default value: 360.0, Range:
  Earth rotation rate (deg/IAT day)

- **"POLARX ": Type: E**, Variable name: PolarX,
  Default value: 0.0, Range:
  Polar position X (meters) on ref. date
- **"POLARY ": Type: E**, Variable name: PolarY,
  Default value: 0.0, Range:
  Polar position Y (meters) on ref. date /**
- **"GSTIA0 ": Type: D**, Variable name: GSTiat0,
  Default value: 0.0, Range:
  GST at time=0 (degrees) on the reference date
- **"UT1UTC ": Type: E**, Variable name: ut1Utc,
  Default value: , Range:
  UT1-UTC (time sec.)
- **"DATUTC ": Type: E**, Variable name: dataUtc,
  Default value: , Range:
  data time-UTC (time sec.)
- **"IATUTC ": Type: E**, Variable name: iatUtc,
  Default value: , Range:
  IAT - UTC (sec).
- **"TIMSYS": Type: A**, Variable name: TimeSys,
  Default value: "UTC", Range:
  Time system, 'IAT' or 'UTC'

6.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"DETECTOR" : Type J** " " (1)
  Variable name root: detector
  Detector number
- **"AZ_OFF" : Type E** "DEGREE " (1)
  Variable name root: azOff
  "Azimuth" offset from nominal pointing, this is formally "cross elevation" in GBT-speak and is the offset on the sky in the direction of azimuth.
- **"EL_OFF" : Type E** "DEGREE " (1)
  Variable name root: elOff
  Elevation offset from nominal pointing

6.1.5 Modification History

1. W. D. Cotton 14/03/2003
   Revision 1: Initial definition

7 Class ObitTableOTFCal

ObitTableOTFCal Class

7.1 OTFCal

Table name: OTFCal

7.1.1 Introduction

[This class contains tabular data and allows access. "OTFCal" contains calibration information for OTF data. Calibrated data for each detector are:

$$\text{cal\_data} = \text{mult}(\text{raw\_data} - \text{cal} - \text{add} - \text{poly})$$

where cal is the calibration noise value for "Cal on" data and 0 for "Cal off" date and poly is evaluated in the direction of the detector.]

7.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc.

7.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"NO_DETEC": Type: J**, Variable name: numDet,
  Default value: 1, Range: ()
  Number of detectors)
- **"NO_POLY": Type: J**, Variable name: numPoly,
  Default value: 1, Range: ()
  Number of polynomial coefficients describing atmospheric emission. 0 =¿ no polynomial model.

7.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable

name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"TIME " : Type E** "DAYS " (1)
  Variable name root: Time
  The center time.
- **"TIME_INT" : Type E** "DAYS " (1)
  Variable name root: TimeI
  The integration time.
- **"TARGET" : Type J** " " (1)
  Variable name root: Target
  Celestial target, as index in target table.
- **"DELTA_AZ" : Type E** "DEGREE " (1)
  Variable name root: dAz
  Correction to the "Azimuth" for all detectors. This is formally "cross elevation" in GBT-speak and is the offset on the sky in the direction of azimuth.
- **"DELTA_El" : Type E** "DEGREE " (1)
  Variable name root: dEl
  Correction to the Elevation for all detectors.
- **"CAL " : Type E** " " (numDet)
  Variable name root: cal
  Cal value in units of raw data per detector, to be subtracted from "Cal on" data.
- **"ADD " : Type E** " " (numDet)
  Variable name root: add
  Additive (subtractive actually) term per detector
- **"MULT " : Type E** " " (numDet)
  Variable name root: mult
  Additive (subtractive actually) term per detector
- **"WEIGHT " : Type E** " " (numDet)
  Variable name root: wt
  Weight value per detector
- **"POLY " : Type E** " " (numPoly)
  Variable name root: poly
  Polynomial atmosphere model, expansion in RA, dec about pointing position

7.1.5 Modification History

1. W. D. Cotton 03/04/2003
   Revision 1: Initial definition

8 Class ObitTableOTFModel

ObitTableCC Class

8.1 CLEAN Components Table

Table name: OTFModel

8.1.1 Introduction

[This class contains tabular data and allows access. "OTFModel" table contains image model components which may be derived via a CLEAN or other fitting process. An OTFModel is the front end to a persistent disk resident structure. Only FITS data are supported. This class is derived from the ObitTable class.]

8.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc. A number of model types are supported as described with their parameters in the following:

- Point
  A Point model has a position and a flux but no extent on the sky. This model is indicated by the absence of the Type column or a value of 0. No additional parameters are needed.
- Gaussian on Sky
  This is a Gaussian shaped model. This model is indicated by a value in the Type column of 1. The extra model parameters are:.
  1. Major axis size in asec.
  2. Minor axis size in asec.
  3. position angle on sky in deg.
- Convolved Gaussian
  This is a Gaussian shaped model. This model is indicated by a value in the Type column of 2. The extra model parameters are:.
  1. Major axis size in asec.
  2. Minor axis size in asec.
  3. position angle on sky in deg.
- Uniform optically thin sphere
  This corresponds to a uniformly filled sphere model which is optically thin. This model is indicated by a value in the Type column of 3. The extra model parameters are:.
  1. Radius in aseconds.

8.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"NO_PARM": Type: J**, Variable name: numParm,
  Default value: 0, Range: ()
  The number of IFs

8.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"X " : Type E** "Degree " (1)
  Variable name root: X
  "X" position of component centroid as offset from reference position
- **"Y " : Type E** "Degree " (1)
  Variable name root: Y
  "Y" position of component centroid as offset from reference position
- **"FLUX " : Type E** "Jansky " (1)
  Variable name root: Flux
  Flux density of component
- **"TYPE " : Type J** " " (1)
  Variable name root: Type
  Component type: 0: (or not present) point, 1=Gaussian on sky, 2= convolved Gaussian, 3=Uniform optically thin sphere
- **"PARMS " : Type E** " " (numParm)
  Variable name root: Parms
  Model components as needed by model

8.1.5 Modification History

1. W. D. Cotton 17/12/2003
   Revision 1: Initial definition

9 Class ObitTableSkyModel

ObitTableSkyModel Class

9.1 SkyModel

Table name: SkyModel

9.1.1 Introduction

[This class contains tabular data and allows access. "SkyModel" contains a sky brightness model in terms of discrete components.]

9.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc.

9.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"RA ": Type: E**, Variable name: RA,
  Default value: 0.0, Range:
  Tangent point RA (deg)
- **"DEC ": Type: E**, Variable name: Dec,
  Default value: 0.0, Range:
  Tangent point Dec (deg)
- **"PROJ ": Type: A**, Variable name: Proj,
  Default value: "-SIN", Range:
  Projection code '-SIN', '-ARC', '-TAN'

9.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"RA_OFF " : Type E** "DEGREE " (1)
  Variable name root: RAOff
  Right ascension offset from tangent point
- **"DEC_OFF " : Type E** "DEGREE " (1)
  Variable name root: DecOff
  Declination offset from tangent point
- **"FLUX " : Type E** "JY " (1)
  Variable name root: Flux
  Flux density

9.1.5 Modification History

1. W. D. Cotton 14/03/2003
   Revision 1: Initial definition

10 Class ObitTableOTFSoln

ObitTableOTFSoln Class

10.1 OTFSoln

Table name: OTFSoln

10.1.1 Introduction

[This class contains tabular data and allows access. "OTFSoln" contains calibration solution information for OTF data. Calibrated data for each detector are:

$$\text{cal\_data} = \text{mult}(\text{raw\_data} - \text{cal} - \text{add} - \text{poly})$$

where cal is the calibration noise value for "Cal on" data and 0 for "Cal off" date and poly is evaluated in the direction of the detector. OTFSoln tables may be applied to either an OTFCal table or directly the data in a self-cal mode.]

10.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc.

10.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"NO_DETEC": Type: J**, Variable name: numDet,
  Default value: 1, Range: ()
  Number of detectors)
- **"NO_POLY": Type: J**, Variable name: numPoly,
  Default value: 1, Range: ()
  Number of polynomial coefficients describing atmospheric emission. 0 =¿ no polynomial model.

10.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.

The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"TIME " : Type E** "DAYS " (1)
  Variable name root: Time
  The center time.
- **"TIME_INT" : Type E** "DAYS " (1)
  Variable name root: TimeI
  The integration time.
- **"TARGET" : Type J** " " (1)
  Variable name root: Target
  Celestial target, as index in target table.
- **"DELTA_AZ" : Type E** "DEGREE " (1)
  Variable name root: dAz
  Correction to the "Azimuth" for all detectors. This is formally "cross elevation" in GBT-speak and is the offset on the sky in the direction of azimuth.
- **"DELTA_EL" : Type E** "DEGREE " (1)
  Variable name root: dEl
  Correction to the El for all detectors.
- **"CAL " : Type E** " " (numDet)
  Variable name root: cal
  Cal value in units of raw data per detector, to be subtracted from "Cal on" data.
- **"ADD " : Type E** " " (numDet)
  Variable name root: add
  Additive (subtractive actually) term per detector
- **"MULT " : Type E** " " (numDet)
  Variable name root: mult
  Additive (subtractive actually) term per detector
- **"WEIGHT " : Type E** " " (numDet)
  Variable name root: wt
  Weight value per detector
- **"POLY " : Type E** " " (numPoly)
  Variable name root: poly
  Polynomial atmosphere model, expansion in RA, dec about pointing position

10.1.5 Modification History

1. W. D. Cotton 05/04/2003
   Revision 1: Initial definition

11 Class ObitTableOTFScanData

ObitTableOTFScanData Class

11.1 OTFScan data

Table name: OTFScanData

11.1.1 Introduction

[This class contains tabular data and allows access. An OTFScanData table has "on the fly" mode observational data from the bolometer array.]

11.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc.

11.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"NO_DETEC": Type: J**, Variable name: numDet,
  Default value: , Range: ()
  The number of detectors.
- **"ORIGIN": Type: A**, Variable name: origin,
  Default value: , Range:
  Originator of file
- **"OBJECT": Type: A**, Variable name: object,
  Default value: , Range:
  Name of object
- **"TELESCOP": Type: A**, Variable name: teles,
  Default value: , Range:
  Telescope used
- **"DATE-OBS": Type: A**, Variable name: obsdat,
  Default value: , Range:
  Date (yyyy-mm-dd) of observation

- **"EPOCH": Type: E**, Variable name: epoch,
  Default value: , Range:
  Celestial coordiate equinox
- **"BUNIT": Type: A**, Variable name: bunit,
  Default value: , Range:
  Data units
- **"OBSRA": Type: D**, Variable name: obsra,
  Default value: , Range:
  Observed Right Ascension in deg.
- **"OBSDEC": Type: D**, Variable name: obsdec,
  Default value: , Range:
  Observed declination in deg.
- **"BEAMSIZE": Type: E**, Variable name: beamSize,
  Default value: 0.00111, Range:
  Gaussian FWHM of telescope beam size.
- **"DIAMETER": Type: E**, Variable name: diameter,
  Default value: 100.0, Range:
  Diameter of telescope in meters.
- **"OTFTYPE": Type: A**, Variable name: OTFType,
  Default value: "Unknown", Range:
  Type of data: "DCR": GBT DCR, "SP": GBT Spectral processor, "CCB":CalTech Continuum Backend, "PAR":Penn Array Receiver

11.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"TIME " : Type E** "DAYS " (1)
  Variable name root: Time
  The center time.
- **"TIME_INT" : Type E** "DAYS " (1)
  Variable name root: TimeI
  The integration time.
- **"TARGET" : Type E** " " (1)
  Variable name root: Target
  Celestial target, as index in target table.

- **"Scan" : Type E** " " (1)
  Variable name root: Scan
  Observing scan index.
- **"RA " : Type E** "DEGREE " (1)
  Variable name root: RA
  Nominal RA of array center
- **"DEC " : Type E** "DEGREE" (1)
  Variable name root: Dec
  Nominal Dec of array center
- **"ROTATE " : Type E** " " (1)
  Variable name root: rotate
  Rotation of array on sky (parallactic angle)
- **"CAL " : Type E** " " (1)
  Variable name root: cal
  if ¿ 0 then the cal source is on.
- **"DATA " : Type E** " " (numDet)
  Variable name root: data
  Detector sample data per detector)

11.1.5 Modification History

1. W. D. Cotton 14/03/2003
   Revision 1: Initial definition
2. W. D. Cotton 14/09/2003
   Added diameter, change name to OTFScanData

12 Class ObitTableOTFTarget

ObitTableOTFTarget Class

12.1 Target table for OTF data documentation

Table name: OTFTarget

12.1.1 Introduction

[This class contains tabular data and allows access. OTFTarget contains information about astronomical sources. An ObitTableOTFTarget is the front end to a persistent disk resident structure. Only FITS cataloged data are supported. This class is derived from the ObitTable class.]

12.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc.

12.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"VELTYP ": Type: A**, Variable name: velType,
  Default value: , Range:
  Velocity type,
- **"VELDEF ": Type: A**, Variable name: velDef,
  Default value: , Range:
  Velocity definition 'RADIO' or 'OPTICAL'
- **"FREQID ": Type: J**, Variable name: FreqID,
  Default value: 0, Range:
  The Frequency ID for which the source parameters are relevant.

12.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable

name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"ID. NO. " : Type J** " " (1)
  Variable name root: TargID
  Target ID
- **"TARGET " : Type A** " " (16)
  Variable name root: Target
  Target name
- **"QUAL " : Type J** " " (1)
  Variable name root: Qual
  Target qualifier
- **"CALCODE " : Type A** " " (4)
  Variable name root: CalCode
  Calibrator code
- **"IFLUX " : Type E** "JY " (1)
  Variable name root: IFlux
  Total Stokes I flux density
- **"QFLUX " : Type E** "JY " (1)
  Variable name root: QFlux
  Total Stokes Q flux density
- **"UFLUX " : Type E** "JY " (1)
  Variable name root: UFlux
  Total Stokes U flux density
- **"VFLUX " : Type E** "JY " (1)
  Variable name root: VFlux
  Total Stokes V flux density
- **"FREQOFF " : Type D** "HZ " (1)
  Variable name root: FreqOff
  Frequency offset (Hz) from nominal
- **"BANDWIDTH" : Type D** "HZ " (1)
  Variable name root: Bandwidth
  Bandwidth
- **"RAEPO " : Type D** "DEGREES " (1)
  Variable name root: RAMean
  Right ascension at mean EPOCH (actually equinox)
- **"DECEPO " : Type D** "DEGREES " (1)
  Variable name root: DecMean
  Declination at mean EPOCH (actually equinox)

- **"EPOCH " : Type D** "YEARS " (1)
  Variable name root: Epoch
  Mean Epoch (really equinox) for position in yr. since year 0.0
- **"RAAPP " : Type D** "DEGREES " (1)
  Variable name root: RAApp
  Apparent Right ascension
- **"DECAPP " : Type D** "DEGREES " (1)
  Variable name root: DecApp
  Apparent Declination
- **"LSRVEL " : Type D** "M/SEC " (1)
  Variable name root: LSRVel
  LSR velocity per IF
- **"RESTFREQ" : Type D** "HZ " (1)
  Variable name root: RestFreq
  Line rest frequency per IF
- **"PMRA " : Type D** "DEG/DAY " (1)
  Variable name root: PMRa
  Proper motion (deg/day) in RA
- **"PMDEC " : Type D** "DEG/DAY " (1)
  Variable name root: PMDec
  Proper motion (deg/day) in declination

12.1.5 Modification History

1. W. D. Cotton 10/07/2003
   Revision 1: Initial version

13 Class ObitTableOTFIndex

ObitTableOTFIndex Class

13.1 Index table for OTF data

Table name: OTFIndex

13.1.1 Introduction

[This class contains tabular data and allows access. ObitTableOTFIndex contains an index for a OTF data file giving the times, target and datum range for a sequence of scans. A scan is a set of observations in the same mode and on the same target. An ObitTableOTFIndex is the front end to a persistent disk resident structure. This class is derived from the ObitTable class.]

13.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc. No Keywords in table.

13.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"SCAN_ID " : Type J** " " (1)
  Variable name root: ScanID
  Scan ID number
- **"TIME " : Type E** "DAYS " (1)
  Variable name root: Time
  The center time of the scan.
- **"TIME_INTERVAL " : Type E** "DAYS " (1)
  Variable name root: TimeI
  Duration of scan
- **"TARGET_ID " : Type J** " " (1)
  Variable name root: TargetID
  Target ID as defined in the OTFTarget table

- **"START_REC " : Type J** " " (1)
  Variable name root: StartRec
  First record number (1-rel) in scan
- **"END_REC " : Type J** " " (1)
  Variable name root: EndRec
  Last record number (1-rel) in scan

13.1.4 Modification History

1. W. D. Cotton 08/06/2003
   Revision 1: Initial version

14 Class ObitTableOTFFlag

ObitTableOTFFlag Class

14.1 Flag table for OTF data documentation

Table name: OTFFlag

14.1.1 Introduction

[This class contains tabular data and allows access. ObitTableOTFFlag contains descriptions of data to be ignored An ObitTableOTFFlag is the front end to a persistent disk resident structure. This class is derived from the ObitTable class.]

14.1.2 Overview

In memory tables are stored in a fashion similar to how they are stored on disk - in large blocks in memory rather than structures. Due to the word alignment requirements of some machines, they are stored by order of the decreasing element size: double, float long, int, short, char rather than the logical order. The details of the storage in the buffer are kept in the ObitTableDesc. No Keywords in table.

14.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"TARGET " : Type J** " " (1)
  Variable name root: TargetID
  Target ID as defined in the OTFTarget table
- **"FEED " : Type J** " " (1)
  Variable name root: Feed
  Feed number to flag, 0=>all
- **"TIME RANGE " : Type E** "DAYS " (2)
  Variable name root: TimeRange
  Start and end time of data to be flagged
- **"FREQ " : Type J** " " (2)
  Variable name root: chans
  First and last frequency channel numbers to flag

- **"PFLAGS " : Type X** " " (4)
  Variable name root: pFlags
  Polarization flags, same order as in data, T=>flagged
- **"REASON " : Type A** " " (24)
  Variable name root: reason
  Reason for flagging

14.1.4 Modification History

1. W. D. Cotton 08/06/2003
   Revision 1: Initial

15 Class ObitTableGBTBEAM_OFFSETS

ObitTableGBTBEAM_OFFSETS Class

15.1 Template ObitTableGBT document

Table name: BEAM_OFFSETS

15.1.1 Introduction

[Table in GBT archive/Antenna file. This class contains tabular data and allows access.
This class is derived from the ObitTable class.]

15.1.2 Overview

15.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"NAME " : Type A** " " (32)
  Variable name root: Name

- **"BEAMXELOFFSET" : Type D** "DEGREE " (1)
  Variable name root: xeloff

- **"BEAMELOFFSET" : Type D** "DEGREE " (1)
  Variable name root: eloff

- **"SRFEED1 " : Type J** " " (1)
  Variable name root: srfeed1

- **"SRFEED2 " : Type J** " " (1)
  Variable name root: srfeed2

15.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

16 Class ObitTableGBTANTPOSGR

ObitTableGBTANTPOSGR Class

16.1 Template ObitTableGBT document

Table name: ANTPOSGR

16.1.1 Introduction

[Table in GBT archive/Antenna file. This class contains tabular data and allows access. This class is derived from the ObitTable class. Secondary Focus receivers.]

16.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

16.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"DMJD " : Type D** "DAY " (1)
  Variable name root: dmjd

- **"RAJ2000 " : Type D** "DEGREE " (1)
  Variable name root: raj2000

- **"DECJ2000" : Type D** "DEGREE " (1)
  Variable name root: decj2000

- **"MNT_AZ " : Type D** "DEGREE " (1)
  Variable name root: mntAaz

- **"MNT_EL " : Type D** "DEGREE " (1)
  Variable name root: mntEl

- **"REFRACT " : Type D** "DEGREE " (1)
  Variable name root: refract

- **"MAJOR " : Type D** "DEGREE " (1)
  Variable name root: major

- **"MINOR " : Type D** "DEGREE " (1)
  Variable name root: minor

- **"SR_XP" : Type D** "MM " (1)
  Variable name root: srXp

- **"SR_YP" : Type D** "MM " (1)
  Variable name root: srYp

- **"SR_ZP " : Type D** "MM " (1)
  Variable name root: srZp

- **"SR_XT " : Type D** "DEGREE " (1)
  Variable name root: srXt

- **"SR_YT " : Type D** "DEGREE " (1)
  Variable name root: srYt

- **"SR_ZT " : Type D** "DEGREE " (1)
  Variable name root: srZt

16.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

17 Class ObitTableGBTANTPOSPF

ObitTableGBTANTPOSPF Class

17.1 Template ObitTableGBT document

Table name: ANTPOSPF

17.1.1 Introduction

[Table in GBT archive/Antenna file. This class contains tabular data and allows access. This class is derived from the ObitTable class. Prime Focus receivers.]

17.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

17.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"DMJD " : Type D** "DAY " (1)
  Variable name root: dmjd

- **"RAJ2000 " : Type D** "DEGREE " (1)
  Variable name root: raj2000

- **"DECJ2000" : Type D** "DEGREE " (1)
  Variable name root: decj2000

- **"MNT_AZ " : Type D** "DEGREE " (1)
  Variable name root: mntAaz

- **"MNT_EL " : Type D** "DEGREE " (1)
  Variable name root: mntEl

- **"REFRACT " : Type D** "DEGREE " (1)
  Variable name root: refract

- **"MAJOR " : Type D** "DEGREE " (1)
  Variable name root: major

- **"MINOR " : Type D** "DEGREE " (1)
  Variable name root: minor

- **"PF_FOCUS" : Type D** "MM " (1)
  Variable name root: pfFocus
  Prime focus focus.

- **"PF_ROTATION" : Type D** "DEGREE " (1)
  Variable name root: pfRotation
  Prime focus rotation

- **"PF_X " : Type D** "MM " (1)
  Variable name root: pfX

17.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

18 Class ObitTableGBTDCRSTATE

ObitTableGBTSTATE Class

18.1 ObitTableGBTDCRState document

Table name: STATE

18.1.1 Introduction

[Table in GBT archive/DCR file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

18.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

18.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"MASTER ": Type: A**, Variable name: master,
  Default value: "DCR", Range:
  Switching signals master
- **"SCAN ": Type: J**, Variable name: scan,
  Default value: , Range:
  Scan number
- **"UTDATE ": Type: J**, Variable name: utdate,
  Default value: , Range:
  MJD of start time
- **"UTCSTART ": Type: D**, Variable name: utcstart,
  Default value: , Range:
  Start time

18.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"BLANKTIM" : Type D** "SECOND " (1)
  Variable name root: blanktim

- **"PHASETIM" : Type D** "SECOND " (1)
  Variable name root: phasetim

- **"SIGREF" : Type B** "" (1)
  Variable name root: sigref

- **"CAL" : Type B** "" (1)
  Variable name root: cal

- **"SWSIG1 " : Type B** "" (1)
  Variable name root: swsig1

- **"SWSIG2 " : Type B** "" (1)
  Variable name root: swsig2

- **"SWSIG3 " : Type B** "" (1)
  Variable name root: swsig3

- **"SWSIG4" : Type B** "" (1)
  Variable name root: swsig4

- **"SWSIG5" : Type B** "" (1)
  Variable name root: swsig5

18.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

19 Class ObitTableGBTDCRRECEIVER

ObitTableGBTDCRRECEIVER Class

19.1 ObitTableGBTDCRRECEIVER document

Table name: RECEIVER

19.1.1 Introduction

[Table in GBT archive/DCR file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

19.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

19.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"SCAN ": Type: J**, Variable name: scan,
  Default value: , Range:
  Scan number
- **"UTDATE ": Type: J**, Variable name: utdate,
  Default value: , Range:
  MJD of start time
- **"UTCSTART ": Type: D**, Variable name: utcstart,
  Default value: , Range:
  Start time

19.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"CHANNELID" : Type I** "" (1)
  Variable name root: channelid

- **"TESTDATA" : Type B** "" (1)
  Variable name root: testdata

19.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

20 Class ObitTableGBTDCRDATA

ObitTableGBTDCRDATA Class

20.1 ObitTableGBTDCRDATA document

Table name: DATA

20.1.1 Introduction

[Table in GBT archive/DCR file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

20.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

20.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"SCAN ": Type: J**, Variable name: scan,
  Default value: , Range:
  Scan number
- **"UTDATE ": Type: J**, Variable name: utdate,
  Default value: , Range:
  MJD of start time
- **"UTCSTART ": Type: D**, Variable name: utcstart,
  Default value: , Range:
  Start time
- **"BACKEND": Type: A**, Variable name: backend,
  Default value: "DCR", Range:
  Which backend
- **"CTYPE1": Type: A**, Variable name: ctype1,
  Default value: "STATE", Range:
  First data axis is State
- **"CTYPE2": Type: A**, Variable name: ctype2,
  Default value: "RECEIVER", Range:
  Second data axis is Receiver

20.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"IFFLAG " : Type I** "CODE " (1)
  Variable name root: ifflag

- **"SUBSCAN " : Type J** "CODE " (1)
  Variable name root: subscan

- **"TIMETAG " : Type D** "DMJD" (1)
  Variable name root: timetag

- **"DATA " : Type J** "COUNTS " (2,2))
  Variable name root: data

20.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

21 Class ObitTableGBTCCBSTATE

ObitTableGBTSTATE Class

21.1 ObitTableGBTCCBState document

Table name: STATE

21.1.1 Introduction

[Table in GBT archive/CCB file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

21.1.2 Overview

A binary table extension called CCBSTATE records the physical definitions of the phases of the data. There are a number of rows equal to the number of phases NPHASES returned for each CCB input port for each integration. Columns PHIA and PHIB records the values of the phase switches A & B at each phase state. Valid values are 0 or 1. Data type of each entry in the column is an unsigned byte. The ordering of rows in the CCBSTATE table corresponds to ordering of the phase columns in the DATA table. The number of rows in the CCBSTATE table is equal to the number of phases in the phase switch cycle (which is in turn equal to 2 to the power of the number of active phase switches hence 1,2, or 4). The number of phases in the phase switch cycle is referred to as NPHASES elsewhere in this document.
Comments:

- NPHASES is equal to 2NACTPSW where NACTPSW is the number of active phase switches. Since NACTPSW has valid values of 0, 1, and 2, NPHASES can be 1, 2, or 4.
- This table is analagous to GBT Backends' STATE tables but differs due to the different implementations of CALs (individual integrations are Cal On or Cal off, rather than having sub-integrations or "phases" be Cal On or Cal Off as for other GBT backends) and SIGREF (next bullet point) for this backend.
- The REF state corresponds to "PHIA XOR PHIB". A SIG state is "NOT(PHIA XOR PHIB)". With two phase switches active there will be two physically distinct rows of the CCBSTATE table that correspond to SIG and two that correspond to REF.

The details of the storage in the buffer are kept in the ObitTableDesc.

21.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"NPHASES": Type: J**, Variable name: nphases,
  Default value: , Range:
  Scan number

21.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"PHIA" : Type J** "state " (1)
  Variable name root: phia
  Value of phase switch A

- **"PHIB" : Type J** "state " (1)
  Variable name root: phib
  Value of phase switch B

21.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

22 Class ObitTableGBTCCBPORT

ObitTableGBTCCBPORT Class

22.1 ObitTableGBTCCBPORT document

Table name: RECEIVER

22.1.1 Introduction

[Table in GBT archive/CCB file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

22.1.2 Overview

A standard (SPN/004) PORT binary table extension is recorded in order to allow the CCB inputs to be crossindexed with the physical descriptions provided in the IF manager IF table. There are two columns: BANK (a character), and PORT (a non-zero integer). A given value of PORT uniquely identifies a physical input to the CCB, and may be used to index physical descriptions (frequency, feed, polarization ,etc) in the IF manager IF table. It also uniquely defines a row in the PORT table. The BANK column is retained for compliance with the GBT FITS standard and are set to a fiducial value of 'A'. Additionally there is a SLAVE column indicating which daughter card a given input port is associated with. Data are unsigned 8-bit integers with valid values 0,1,2,3. The order of the rows of the PORT table correspond to the ordering of PORT columns in the DATA table. The number of rows NPORTS of the PORT table is equal to the number of ports input ports selected as active in the manager for the given scan. The details of the storage in the buffer are kept in the ObitTableDesc.

22.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"BANK" : Type A** " " (1)
  Variable name root: bank
  Always has value of "A" but included for compliance with GBT standards.
- **"PORT" : Type I** " " (1)
  Variable name root: port
  Identifier for a physical input to the CCB
- **"SLAVE" : Type I** " " (1)
  Variable name root: slave
  Which daughter card a given input port is associated with.

22.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

23 Class ObitTableGBTCCBDATA

ObitTableGBTCCBDATA Class

23.1 ObitTableGBTCCBDATA document

Table name: DATA

23.1.1 Introduction

[Table in GBT archive/CCB file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

23.1.2 Overview

The DATA binary table extension contains raw accumulated total power integrations for each phase of each CCB input port that was used for a given scan. The first (DMJD) column of the data array contains the MJD of the integration start. The DATA column is a multidimensional column with dimensions (NPORTS,NPHASES). Each datum is recorded as a 32 bit two's complement integer; subsequent transformation to unsigned values is facilitated by recording a TZERO keyword with a value of 2^31. The order of the PORT and PHASE sub-columns should correspond to the order of the rows in the PORT and CCBSTATE tables respectively. The number of phases NPHASES is determined by the number of active switches and is 1, 2, or 4. The number of ports NPORTS is equal to the number of ports selected in the manager as active for the given scan.
A second multi-dimensional OVRFLOW column, of the same dimensions as the DATA column, comprises LOGICAL data with 'T' indicating integrations that overflowed and 'F' indicating integrations that did not. The value at subcolumn M row N in the OVRFLOW column denotes the overflow status of the integration datum at subcolumn M row N of the DATA column. One multi-dimensional LOGICAL column contains the four SLAVEOK flags. Two LOGICAL columns contain CAL A and CAL B ON flags indicating whether, for a given integration, a given call was on or not; the "integration usable" flag is a separate SHORT-INT column, indicating whether each integration is usable based on the cal diode rise and fall time flags applied by the CCB.
Comments

- Integration data are returned by the CCB as unsigned 32 bit integers. Conversion from signed 32 bit two's complements values, to unsigned 32 bit values, may require use of double precision on the data processing end.
- The "integration usable columns" is short int not logical in order to more closely line up the columns with machine byte boundaries, for better performance.
- The SLAVEOK flags can be associated with individual columns of data (ie input ports) using the information in the SLAVE column of the PORT table.
- The cabling-dependent mapping of the CCB's "Cal A" and "Cal B" to a physical Cal diode (nominally Left and Right, or perhaps, tags in the calibration FITS file database) is presently unspecified.

The details of the storage in the buffer are kept in the ObitTableDesc. The ordering of "PORTS" is given in the IF table (Freq, feed, poln). CCBSTATE given in CCBSTATE table; guessing PHIA = sig/ref (ref state swap the two feeds), PHIB = cal, 1=on.

23.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"DMJD " : Type D** "d " (1)
  Variable name root: dmjd
  MJD of the integration start.
- **"SLAVEOK" : Type L** "bool " (4)
  Variable name root: slaveok
  Is each daughter card in an OK state?
- **"USABLE " : Type I** "status" (1)
  Variable name root: usable
- **"CALA " : Type L** "status" (1)
  Variable name root: cala
  Is Cal A "on"?
- **"CALB " : Type L** "status" (1)
  Variable name root: calb
  Is Cal B "on"?
- **"OVRFLOW " : Type L** "bool " (16,4))
  Variable name root: ovrflow
  DATA overflowed? - same order as DATA
- **"DATA " : Type J** "ulong " (16,4))
  Variable name root: data
  unsigned (PORT,CCBSTATE)

23.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

24 Class ObitTableGBTPARDATA

ObitTableGBTPARDATA Class

24.1 ObitTableGBTPARDATA document

Table name: DATA

24.1.1 Introduction

[Table in GBT archive/Penn Array Camera file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

24.1.2 Overview

The DATA binary table extension contains raw bolometer data. The "TimeStamp" column contains the number of seconds since the beginning of Unix Time (1 Jan 1976?).
The main file HDU contains the following keywords:

- "DATE-OBS"
  String in form yyyy-mm-dd giving UTC start date
- "INSTRUME"
  String 'PennArrayReceiver'
- "UTCSTART"
  String: UTC start time in seconds since midnight
- "UTDSTART"
  String: UTC starttime in MJD, "unknown" = unspecified
- "UTCEND"
  String: UTC of exposure end, "unknown" = unspecified
- "SCANNUM"
  Integer: scan number(?).
- "PROJID"
  String: Project ID

24.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"CFGVALID": Type: L**, Variable name: cfgvalid,
  Default value: , Range:
  If true, configuration has not changed during scan

24.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"TimeStamp" : Type D** "Second" (1)
  Variable name root: TimeStamp
  Unix time in seconds.
- **"daccounts" : Type D** "Count" (72)
  Variable name root: daccounts
  daccounts
- **"saecounts" : Type D** "Count" (72)
  Variable name root: saecounts
  saecounts
- **"DigInput" : Type B** "bool" (6)
  Variable name root: DigInput
  DigInput

24.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

25 Class ObitTableGBTPARSENSOR

ObitTableGBTPARSENSOR Class

25.1 ObitTableGBTPARSensor document

Table name: Sensor

25.1.1 Introduction

[Table in GBT archive/PAR file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

25.1.2 Overview

A binary table extension called PARSENSOR Gives the row and column numbers of each sensor. NB this does not appear to correspond to sky position. Eight of the sensors are "dark", i.e. don't see the sky.

25.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"Row" : Type J** " " (1)
  Variable name root: row
  Row number (0-rel) of corresponding sensor, row number=index in data arrays
- **"Col" : Type J** " " (1)
  Variable name root: col
  Column number (0-rel) of corresponding sensor, row number=index in data arrays

25.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

26 Class ObitTableGBTSPSTATE

ObitTableGBTSTATE Class

26.1 ObitTableGBTSPSTATE document

Table name: STATE

26.1.1 Introduction

[Table in GBT archive/SpectralProcessor file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

26.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

26.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"FORMATID": Type: A**, Variable name: formatid,
  Default value: "GBSDD007", Range:
  SDD_FORMAT_ID
- **"SCAN": Type: J**, Variable name: scan,
  Default value: , Range:
  Scan number
- **"SUBSCAN": Type: J**, Variable name: subscan,
  Default value: , Range:
  Scan record number
- **"UTDATE": Type: J**, Variable name: utdate,
  Default value: , Range:
  MJD of start time
- **"UTCSTART": Type: D**, Variable name: utcstart,
  Default value: , Range:
  UTC start time seconds.
- **"UTCSTOP": Type: D**, Variable name: utcstop,
  Default value: , Range:
  Stop time seconds.

- **"RCVRS": Type: J**, Variable name: rcvrs,
  Default value: , Range:
  Each item is index by RCVRS.

26.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"BLANKTIM" : Type E** "SECOND " (2)
  Variable name root: blanktim

- **"PHASETIM" : Type E** "SECOND " (2)
  Variable name root: phasetim

- **"SIGREF" : Type B** "" (2)
  Variable name root: sigref

- **"CAL" : Type B** "" (2)
  Variable name root: cal

- **"FFTS " : Type J** "" (2)
  Variable name root: ffts

- **"DELETED " : Type J** "" (2)
  Variable name root: deleted

26.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

27 Class ObitTableGBTSPRECEIVER

ObitTableGBTSPRECEIVER Class

27.1 ObitTableGBTSPRECEIVER document

Table name: RECEIVER

27.1.1 Introduction

[Table in GBT archive/SpectralProcessor file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

27.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

27.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"SCAN ": Type: J**, Variable name: scan,
  Default value: , Range:
  Scan number
- **"UTDATE ": Type: J**, Variable name: utdate,
  Default value: , Range:
  MJD of start time
- **"UTCSTART ": Type: D**, Variable name: utcstart,
  Default value: , Range:
  Start time

27.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"RCVRID" : Type J** " " (1)
  Variable name root: rcvrid

- **"TAPER" : Type A** " " (8)
  Variable name root: taper

- **"OBSFREQ" : Type D** "HZ" (1)
  Variable name root: obsfreq

- **"IFF" : Type D** "HZ" (1)
  Variable name root: iff

- **"FREQRES" : Type D** "HZ" (1)
  Variable name root: freqres

- **"BANDWD" : Type E** "HZ" (1)
  Variable name root: bandwd

- **"TCAL" : Type E** "DEGREE" (1)
  Variable name root: tcal

- **"TPLEVEL" : Type E** " " (1)
  Variable name root: tplevel

- **"FASTTIM" : Type E** "SECOND" (1)
  Variable name root: fasttim

- **"SLOWTIM" : Type E** "SECOND" (1)
  Variable name root: slowtim

- **"CLIP" : Type E** "SECOND" (1)
  Variable name root: clip

- **"THRESH" : Type E** "SECOND" (1)
  Variable name root: thresh

- **"SYNTHL" : Type B** "CODE" (1)
  Variable name root: synthl

- **"OVERL" : Type B** "CODE" (1)
  Variable name root: overl

- **"IMODF" : Type B** "CODE" (1)
  Variable name root: imodf

- **"IFSYNTH" : Type B** "CODE" (1)
  Variable name root: ifsynth

- **"TAPEROFF" : Type B** "CODE" (1)
  Variable name root: taperoff

- **"RFIEXC" : Type B** "CODE" (1)
  Variable name root: rfiexc

- **"CLKSRC" : Type B** "CODE" (1)
  Variable name root: clksrc

- **"IFLO" : Type B** "CODE" (1)
  Variable name root: iflo

- **"IFSIDE" : Type B** "CODE" (1)
  Variable name root: ifside

- **"RFSIDE" : Type B** "CODE" (1)
  Variable name root: rfside

27.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

28 Class ObitTableGBTSPDATA

ObitTableGBTSPDATA Class

28.1 Template ObitTableGBTSPDATA document

Table name: DATA

28.1.1 Introduction

[Table in GBT archive/SpectralProcessor file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

28.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc.

28.1.3 Keywords

The following keywords must follow the required bintable keywords but the order is otherwise arbitrary. Allowed data types are: **I**=integer(16-bit), **J**=integer(32-bit), **E**=real, **D**=double, **A**=character, **L**=logical. "Variable name" is the name the keyword is given in software and "Range" if present indicates a structural keyword, "()" indicates all values are allowed and "(n,m)" indicates values from n to m. This latter is used to indicate sufixes for column lables.

- **"OBJECT ": Type: A**, Variable name: object,
  Default value: , Range:
  Source name
- **"SCAN ": Type: J**, Variable name: scan,
  Default value: , Range:
  Scan number
- **"UTDATE ": Type: J**, Variable name: utdate,
  Default value: , Range:
  MJD of start time
- **"UTCSTART ": Type: D**, Variable name: utcstart,
  Default value: , Range:
  Start time in seconds.
- **"UTCSTOP ": Type: D**, Variable name: utcstop,
  Default value: , Range:
  Stop time in seconds.
  tablekey["INTTIME "Dinttime Integration time in seconds.] tablekey["BACKEND"Abackend"DCR"
  Which backend]
- **"CTYPE1": Type: A**, Variable name: ctype1,
  Default value: "STATE", Range:
  First data axis is State

- **"CTYPE2": Type: A**, Variable name: ctype2,
  Default value: "RECEIVER", Range:
  Second data axis is Receiver

28.1.4 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X**=bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"SUBSCAN " : Type J** "CODE " (1)
  Variable name root: subscan

- **"UTDATE " : Type J** "DAY" (1)
  Variable name root: utdate

- **"UTCSTART " : Type D** "SECOND" (1)
  Variable name root: utcstart

- **"PSRPER " : Type D** " " (1)
  Variable name root: psrper
  Pulsar period.

- **"DATA " : Type E** "COUNT " (1024,2,2))
  Variable name root: data
  Data

28.1.5 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT

29 Class ObitTableGBTIF

ObitTableGBTIF Class

29.1 ObitTableGBTIF document

Table name: IF

29.1.1 Introduction

[Table in GBT archive/IF file. This class contains tabular data and allows access. This class is derived from the ObitTable class.]

29.1.2 Overview

The details of the storage in the buffer are kept in the ObitTableDesc. IF controler information about frequency and polarization setup. Sky Frequency Formula:

$$sky = SFF\_SIDEBAND * IF + SFF\_MULTIPLIER * LO1 + SFF\_OFFSET$$

Signed Sum of the LOs:

$$sum = -(SFF\_MULTIPLIER * LO1 + SFF\_OFFSET)/SFF\_SIDEBAND$$

29.1.3 Columns

The order of the columns is arbitrary. Allowed data types are: **I**=16 bit integer, **J**=32 bit integer, **E**=real, **D**=double, **C**=Complex (32 bit each), **M**=double complex (64 bit each), **A**=character, **L**=logical, **X=**bit array, **B**=byte array. Multidimensional arrays use the `TDIMnnn` keyword convention.
The first line of each entry gives the name:type, units and dimensionality of the entry. "Variable name" is the name the keyword is given in software, possibly with a suffix if the column label ends in #xxx where xxx is a keyword.

- **"BACKEND" : Type A** " " (32)
  Variable name root: backend
  Name of the terminating backend.
- **"BANK " : Type A** " " (2)
  Variable name root: bank
  Name of the backend's set of inputs.
- **"PORT " : Type J** " " (1)
  Variable name root: port
  Index of the backend's input.
- **"RECEIVER" : Type A** " " (32)
  Variable name root: receiver
  Name of the receiver of origin.

- **"FEED" : Type J** " " (1)
  Variable name root: feed
  Index of receiver RF entry point (0 indicates none.
- **"SRFEED1" : Type J** " " (1)
  Variable name root: srfeed1
  Index of first FEED of a sig/ref pair.
- **"SRFEED2" : Type J** " " (1)
  Variable name root: srfeed2
  Index of second FEED of a sig/ref pair.
- **"RECEPTOR" : Type A** " " (8)
  Variable name root: receptor
  Name of the receiver's detector.
- **"LO_CIRCUIT" : Type A** " " (32)
  Variable name root: loCircuit
  Circuit producing the tracking frequency.
- **"LO_COMPONENT" : Type A** " " (32)
  Variable name root: loComponent
  component producing the tracking frequency.
- **"SIDEBAND" : Type A** " " (2)
  Variable name root: sideband
  Resulting sideband: upper or lower.
- **"POLARIZE" : Type A** " " (2)
  Variable name root: polarize
  Resulting polarization ('X', 'Y', 'R', 'L').
- **"CENTER_IF" : Type E** "HZ" (1)
  Variable name root: CenterIF
  Approximate physical center frequency.
- **"CENTER_SKY" : Type E** "HZ" (1)
  Variable name root: CenterSky
  Approximate center frequency on the sky.
- **"BANDWDTH" : Type E** "HZ" (1)
  Variable name root: bandwdth
  Approximate resulting bandwidth. BANDWDTH of 0 denotes the bandpass is outside the optimal range.
- **"HIGH_CAL" : Type J** " " (1)
  Variable name root: highCal
  Indicates a high powered calibrator was used.
- **"TEST_TONE_IF" : Type E** "HZ" (1)
  Variable name root: testToneIF
  Approximate physical test tone frequency, if any.

- **"TEST_TONE_SKY" : Type E** "HZ" (1)
  Variable name root: YestToneSky
  Approximate test tone frequency on the sky, if any.
- **"TEST_TONE_CIRCUIT" : Type A** " " (32)
  Variable name root: TestToneCircuit
  Circuit producing the test tone, if any.
- **"TEST_TONE_COMPONENT" : Type A** " " (32)
  Variable name root: testToneComponent
  Component producing the test tone, if any.
- **"SFF_MULTIPLIER" : Type D** " " (1)
  Variable name root: sffMultiplier
  Sky Frequency Formula multiplier coefficient.
- **"SFF_SIDEBAND" : Type D** " " (1)
  Variable name root: sffSideband
  Sky Frequency Formula sideband coefficient.
- **"SFF_OFFSET" : Type D** " " (1)
  Variable name root: sffOffset
  Sky Frequency Formula offset coefficient.
- **"TRANSFORM_COUNT" : Type J** " " (1)
  Variable name root: transformCount
  Number of transform.
- **"TRANSFORMS" : Type A** " " (4096)
  Variable name root: transforms
  Matrix of transform descriptions (frequencies in MHz).

29.1.4 Modification History

1. A. N. Author 99/99/9999
   Revision 1: Copied from GBT