The ABC of XTE

Data Files

Introduction

Although you don’t need to understand fully the detailed contents of RXTE data files in order to reduce RXTE data, some knowledge of file structure and contents will certainly make your RXTE life easier. Fortunately, despite the plethora of modes and configurations for PCA and HEXTE, there are only two basic formats for science data: ***science array*** and ***science event***. These two basic formats are described in this chapter and follow a brief description of RXTE FITS files in general. Also described here are the RXTE filter files which enable "good" data to be selected on the basis of various housekeeping data.

Using just two basic formats requires that the hundreds of different configurations be described for software in a comprehensive and unified way. This task is performed by DDL - the Data Description Language in which certain configuration-identifying keywords are written. Knowing DDL is not strictly necessary, but is included here in a footnote for completeness. PCA modes and configurations are described individually in the PCA Issues chapter; HEXTE modes and configurations in the HEXTE Issues chapter.

FITS Files

Basic Structure

All RXTE data files are in FITS format and have the following parts:

1. **Primary Header** contains information about the mission, the instrument, the observation and the initial processing. As with all FITS headers, this information is in the form of keywords with

assigned values.

2. **Primary Image Array** is blank.

3. **First Extension Header** contains keywords which provide a complete and detailed description of the contents of the first extension. For convenience, it also contains some of the same information as the primary header.

4. **First Extension** contains the scientific data. In the case of of science array files, the first extension is called **`XTE_SA`**. In the case of science event files, it is called **`XTE_SE`**.

5. **Second Extension Header** contains keywords which provide a complete and detailed description of the contents of the second extension. For convenience, it also contains some of the same information as the primary header.

6. **Second Extension** lists the standard good time intervals, i.e. the start and stop times of all the potentially useable data in the file. It is called **`GTI`**.

7. **Third Extension Header** contains keywords which provide a complete and detailed description of the contents of the third extension. For convenience, it also contains some of the same information as the primary header.

8. **Third Extension** lists the good time intervals, i.e. the start and stop times of all the data in the file, regardless of whether the data are actually useable. Like the second extension, it is called **`GTI`**.

A general guide to FITS format can be obtained via anonymous FTP at the FITS home page.

Useful Ftools for Examining FITS Files

The following ftools display summaries or the full contents of FITS files. We recommend you run them on one or two sample FITS files to get an idea of the structure and contents of your data files.

1. **fstruct** provides an overview of the structure of a FITS file, i.e. the number, names and sizes of the extensions it contains.

2. **flcol** provides an overview of the structure of a specified extension, i.e. the number, names and sizes of the columns it contains.

3. **fdump** dumps to the screen the entire contents of a FITS file, starting with all the headers.

4. **fkeyprint** prints to the screen the value of a specified header keyword.

5. **fv = "FITS Viewer"** is a GUI-based tool that combines the functionality of the above four tools.

As with all ftools, typing **fhelp** followed by the name of the ftool will yield on-line help. You can also visit the ftools page.

Important Timing Keywords in the Primary Header

The keyword values in the primary header provide information about the start and stop times of the observation, the time units, the target, the Principal Investigator etc. You can view a primary header (and the rest of the FITS file) by using the ftool **fdump**. Most of these keywords, such as `OBSERVER`, have obvious meanings. However, the keywords associated with time require further explanation.

Time in RXTE data files is kept in TT (Terrestrial Time), the IAU standard and successor to Ephemeris Time (ET). TT and its relationship to other time systems is described in the Time Tutorial chapter. In general, the times stamped on data in RXTE files have the following properties:

- Time stamps are in Space Craft Clock Seconds (SCCS), the true elapsed seconds since the "start" of the mission: January 1, 1994, at 0:0:0 UTC, which corresponds to MJD=49353.0 (UTC).
- Time stamps plus the value of the **`TIMEZERO`** keyword provide time in TAI (International Atomic Time) as seconds since January 1, 1994, at 0:0:28 (TAI).
- Time stamps plus `TIMEZERO` plus **`MJDREFI`** plus **`MJDREFF`** provide the absolute time in TT. Here, the sum of the two `MJDREFI` values gives the MJD of the start of the mission *including* the conversion from UTC to TT.
- All times are in seconds, except **`MJDREFI`** and **`MJDREFF`** which are in days.

Science Array Files

Basic Structure

The science array format is used for data binned at regular intervals by the spacecraft electronics. Examples are the PCA Standard 2 configuration, which contains 129-channel spectra accumulated every 16 seconds, and HEXTE multi-scalar Bin mode, which contains light curves in 1-8 spectral bands. The data occupy the `XTE_SA` extension in the form of regularly accumulated arrays or histograms.

The XTE_SA Extension

The `XTE_SA` extension contains the science data arranged in columns (for different kinds of data) and rows (for each accumulation). The individual column-row cells may contain single values or arrays, depending on the kind of data. For example, the `Time` column contains one value per row, whereas the `XeCnt` column in certain PCA files contains, per row, a histogram of count rates in different energy bands. For modes with data histogrammed according to time, this means that the histogram step size - not the accumulation time - represents the actual time resolution of the data.

Important point: The times associated with binned data in science array files always refer to the beginning of the bin. This convention makes it much easier for software to keep track of time within the histograms. Likewise, in the light curves produced by the current extractors, the time also refers to the beginning of the bin, as reflected in the TIMEPIXR keyword value of 0.0. (Older versions of the extractors may have used TIMEPIXR=0.5.)

Since the Science Array format is a broad scheme, rather than a detailed prescription, the actual data format for a given configuration is described in the file itself - in the form of keywords in the extension header which you can view, like the rest of the FITS file, by using the ftool **fdump**. The most important of these are as follows:

- **`DATAMODE:`** In the case of RXTE, this keyword gives the configuration, e.g. `B_250us_2A_0_17_Q,`.
- **`TDDES:`** The value is a DDL string giving the name of the Observatory (XTE) and Instrument (ASM, HEXTE or PCA), e.g. `O[XTE] & I[PCA].`
- **`TTYPEn:`** This keyword gives the name of the nth column. You can also find out the column names using the ftool **flcol**. Note that not all science array files have the same column names (apart from `Time`, that is): the names depend on instrument and mode.
- **`TDDESn:`** In general, this keyword describes the data in the nth column. For the first, `Time` column, the value is simply `Time`. However, for second and higher columns it contains a DDL string. For example, `TDDES2` in a PCA Standard 1 data file has the value:

  ```
  D[0] & E[X1L^X1R^X2L^X2R^X3L^X3R] & C[0~255] & T[0.0;0.125;1024]
  ```

  which means that the data come from Detector 0 (i.e. PCU0), from any one of the six detector Elements (xenon anodes), in one band spanning Channels 0-255, and packaged in a Time histogram with zero offset, and 1024 0.125-second steps.
- **`DELTAT:`** This keyword gives the accumulation time of the histograms, i.e. the time between each row in the file. In the above example, it is 128 seconds (1024 times 0.125 s). In general, the accumulation time of the histograms is *not* a key piece of information for most users, as it only dictates the arrangement of the data in the file, rather than their scientific properties. The RXTE ftool saextrct transparently extracts data without your having to know how many steps each histogram contains or how often the histograms are accumulated. However, if you intend to write your own data reduction software, the accumulation time is crucial.

Note: The `XTE_SA` extension in PCA science array files also contains additional columns devoted to various EDS flags, as well as a collection of keywords called the coda which summarize EDS activity. The flags and coda are usually not important, but are described in a footnote for completeness.

The Two GTI Extensions

RXTE science array files have two additional extensions containing good time intervals (GTI). Both are called **`GTI`**.

The first `GTI` extension, the second extension in the file, contains the ANDed GTI corresponding to the times when:

1. Telemetred data are present
2. The satellite is pointing at the nominal source position, as derived from the spacecraft attitude
3. The nominal source position is not occulted by the Earth, as predicted by mission operations
4. The satellite is outside the South Atlantic Anomaly, as predicted by mission operations

The second `GTI` extension, the third extension in the file, contains the GTI corresponding to when telemetred data are present, i.e. just the first of the four criteria in the first `GTI` extension.

Science Event Files

Basic Structure

The science event format is used for unbinned data, i.e. for individual events. An example is the PCA Good Xenon configuration which contains time-stamped events with 256-channel resolution, PCU ID and anode ID. The science data occupy the `XTE_SE` extension in the form of event words - binary-encoded descriptions of the individual events. This format is more compact than the column-based format used for ROSAT and ASCA event-lists, and is fundamentally different.

The XTE_SE Extension

The `XTE_SE` extension contains the science data arranged in rows (one per event) and in two columns: the first, `Time`, for the time; the second, `Event`, for the event word. Event words comprise strings of ones and zeros, the combinations of which define the properties of each event with respect to a template of all possible properties within the configuration. This template is broken up into sections which, depending on the particular configuration, refer to things like PCU ID, PHA channel band, etc. Thus, an individual event word, with its particular combination of ones and zeros, picks out, say, one PCU ID, one PHA channel band etc.

Since the Science Event format is a broad scheme, rather than a detailed prescription, the actual data format for a given configuration is described in the file itself - in the form of keywords in the extension header which you can view by using the ftool **fdump** (note, however, that fdump will not display event words). The most important of these are as follows:

- **`DATAMODE:`** In the case of RXTE, this keyword gives the configuration, e.g. `E_1us_1M_0_1s`.
- **`TDDES:`** The value is a DDL string giving the name of the Observatory (XTE) and Instrument

(ASM, HEXTE or PCA), e.g. `O[XTE] & I[PCA]`.

- **`TDDES2`**: This keyword *in PCA Event files* contains a DDL string which describes where the data in the second, `Event` column came from, i.e. from which detector, anode, channels, etc. For example, for files containing data in the `E_1us_1M_0_1s` PCA configuration, the value of `TDDES2` is:

  ```
  D[0~4] & E[X1L^X1R^X2L^X2R^X3L^X3R] & C[0~249]
  ```

  which means that the data come from Detector 0-4 (i.e. PCU0-4), from any one of the six detector Elements (xenon anodes), from Channels 0-249 (but not necessarily in one band).

- **`TDDES4`**: This keyword *in HEXTE Event files* contains a DDL string which describes where the data in the fourth, `Event` column came from, i.e. from which detector, pulse height channels, PHA channels, etc. For example, for files containing data in the `E_8us_256_DX1F` HEXTE configuration, the value of `TDDES4` is:

  ```
  D[0~3] & E[0~131071] & C[0~255] & T[0~16]
  ```

  which means that the data come from Detectors 0-3 , with 17 bits of additional event information (see section on HEXTE Event List configurations and from Channels 0-255.

- **`TEVTB2`**: This keyword *in PCA Event files* contains a DDL string which defines the event word template. For example, `TEVTB2` in a PCA `E_1us_1M_0_1s` data file has the value:

  ```
  (M[1]{1},D[0:4]{3},T[0.0:0.00390625;9.5367431640625e-07]{12})^(M[12&
  7]{8},S[One5]{5},S[FirstFlag]{1},S[SpillFlag]{1},S[Zero1]{1})^(M[1]&'
  {8},S[LostEvents0]{8})^(M[2]{8},S[LostEvents1]{8})^(M[4]{8},S[Zero]&'
  {7},S[Spillage]{1})^(M[8]{8},S[Zero]{5},S[ModeSpecific]{3})
  ```

  This looks intimidatingly verbose. However, the keyword actually contains six alternative event words, each denoted by an M-token (emboldened here for clarity) which tells the RXTE software which of the six applies to the row. It is the first event word which describes the science data: the last five event words are for various EDS flags.

  Like science array files, event files also have a coda of keywords summarizing EDS activity.

  For a detailed explanation of this particular event word, please refer to the section in the *PCA Issues* chapter on Generic Event Mode configurations. For more about DDL, see the DDL footnote to this chapter.

- **`TEVTB4`**: This keyword *in HEXTE Event files* contains a DDL string which defines the event word template. For example, `TEVTB4` in a HEXTE `E_8us_256_DX1F` data file has the value:

  ```
  E[0:63] {6}, E[CAL] {1}, T[0:15;1] {4}, D[0:3] {2}, E[LE1,LE0] {2},
  T[0:1;7.62939453125e-6] {17}, C[0:255] {8}
  ```

  and is described in more detail in the section HEXTE Event List configurations.

  Unlike PCA Event modes, there are no M-tokens and no "alternative" event words in HEXTE Event files. This is because the non-events that would require alternative event words are *not* included in telemetry. This means, for example, that a HEXTE event file will not contain an instance of a non-zero calibration flag, i.e. `E[CAL]` will not equal 1. It also means that there is no need to select events based on the calibration flag.

The Two GTI Extensions

RXTE science array files have two additional extensions containing good time intervals (GTI). Both are called **`GTI`**.

The first `GTI` extension, the second extension in the file, contains the ANDed GTI corresponding to the times when:

1. Telemetred data are present
2. The satellite is pointing at the nominal source position, as derived from the spacecraft attitude
3. The nominal source position is not occulted by the Earth, as predicted by mission operations
4. The satellite is outside the South Atlantic Anomaly, as predicted by mission operations

The second `GTI` extension, the third extension in the file, contains the GTI corresponding to when telemetred data are present, i.e. just the first of the four criteria in the first `GTI` extension.

Filter Files

RXTE filter files contain the values of various housekeeping data and derived quantities with which good data can be selected. For instructions on how to use filter files, please see the recipe on *Reduction and Analysis of PCA Spectra*. For instructions on how to create a new filter file, see the recipe *Creating XTE Filter Files*. In this section we describe the files themselves.

Properties

Conventionally, there is one filter file for each Observation ID, i.e. for each temporally contiguous collection of data from a single pointing. The filter file for a given ObsId can be found in the Standard Products subsystem - unless, that is, your data tape was made in the first few months of AO-1, in which case you have to make your own. In fact, you'll also have to make a filter file if the exisiting one does not contain the `BKGD_THETA`, `BKGD_PHI`, `TIME_SINCE_SAA`, and `ELECTRONn` columns needed by the background generator pcabackest.

Note that whilst a filter file spans an entire ObsId, the corresponding PCA and HEXTE data trains can contain gaps - for instance, when the instruments were turned off to protect them from the SAA.

The files themselves contain regularly spaced rows in time (the default bin size is 16 seconds) made up of fifty or so columns, each containing a housekeeping or derived parameter. Examples are the high voltage across the three Xenon layers in PCU0 (with the column name **`hvXE_PCU0`**) and the Right Ascension of the instantaneous pointing direction (**`POINT_RA`**). You can list all the column names by using the ftool **flcol** on the filter file.

How to Make a Filter File

We provide here brief instructions for creating a filter file. For further details, see the recipe *Creating XTE Filter Files*.

The Perl script **xtefilt** will make a filter file, automatically consulting the appropriate housekeeping files. It requires, as input:

1. The ObsID (e.g. 10367-02-02-00). Note that for realtime data, the ObsId is different. Please check the recipe *Working with Realtime Data* for details.

2. The path to the directory containing your FITS Master Index (FMI) file , (e.g. `/home1/day/xte/P10367` if that's where the file `FMI` is).

3. A file listing the Application IDs used to generate the filter file. This list changes as experience with data screening grows, and as modifications are made to estimating the PCA background. The current version of the list can be found by typing **fhelp xtefilt**. Please cut and paste its contents into your own file.

Additional, optional inputs are possible, including changing the size of the time bins from 1 second (the default). Typing **fhelp xtefilt** will display complete instructions for using xtefilt. An example of its use:

```
> xtefilt -o "10367-02-02-00" -p "/home1/day/xte/" -a "applist" -t 100
```

where `applist` contains the list of AppIds and the time bins are set to 100 seconds.

Footnote 1 - DDL: Data Description Language

- Purpose
- Basic Syntax: DDL for TDDES Keywords
- DDL For Science Event Data

Purpose

One of RXTE's innovations is its flexibility in selecting and packaging raw data for telemetry, as reflected in the hundreds of different instrument configurations. To describe these configurations in a compact, comprehensive and comprehensible manner, the Data Description Language (DDL) was devised by Arnold Rots of the RXTE GOF. Specifically, DDL serves three purposes:

1. To enumerate the selection criteria used by the on-board electronics.

2. To describe the structure of the data in the file.

3. To convey the meaning of the data.

It is also designed to be read by software and by humans. For a more comprehensive guide to DDL, please see *XTE FITS Formatter Design Documents* by Rots, Hilldrup & Barnette.

Basic Syntax: DDL for TDDES Keywords

DDL was devised ab initio as a flexible language extensible to other missions. Its flexibility and terseness do, however, make it easier to gain an understanding by specific example rather than by general exposition.

A typical ***string*** of DDL is a concatenation of ***tokens*** (denoted by single letters) with ***assigned values*** (enclosed in square brackets). For example, the `TDDES2` keyword in science array files is used to describe the data in the second column. In the case of the `SB_500us_50_249_2s` PCA configuration, the value of `TDDES2` is a DDL string which looks like this:

```
D[0~4] & E[X1L^X1R^X2L^X2R^X3L^X3R] & C[50~249] & T[0.0;0.00048828125;4096]
```

It has four tokens:

1. **`D`** – for Detector (PCU in this case)
2. **`E`** – for detector Element (xenon anode in this case)
3. **`C`** – for detector Channel
4. **`T`** – for Time

These tokens have values enclosed in square brackets and are concatenated with DDL's `&` symbol which means AND.

Let's look at the token values:

1. **`D[0~4]`** means that events were selected from PCU0-4 (DDL's `~` symbol delimits an accumulation range).

2. **`E[X1L^X1R^X2L^X2R^X3L^X3R]`** means that each event came from only one of the six xenon anodes (DDL's `^` symbol means EOR - exclusive or).

3. **`C[50~249]`** means that events were selected from channels 50-249.

4. **`T[0.0;0.00048828125;4096]`** means that each accumulation contains 4096 time-steps, 0.00048828125 seconds wide, with 0 offset.

It should be clear from this example that the meaning of a particular DDL string depends on context to some extent. We could associate the token `D[0~4]` with PCU0-4 because the file contains RXTE PCA data. In fact, this information is contained in the `TDDES` keyword, the value in this case being `O[XTE] & I[PCA]` (`O` and `I` are the Observatory and Instrument tokens, respectively).

DDL For Science Event Data

When used for `TDDES` keywords, DDL enumerates and describes selection criteria as used by the on-board electronics. When used for science event data - specifically for the `TEVTB2` keyword - DDL describes how the data are structured in the file. Although DDL does not change meaning with usage, it is advisable to bear this distinction in mind when looking at a DDL string.

In general, when DDL is used for event words, each token corresponds to pieces of the event word template. The size (in bits) of each piece follows the token, enclosed in curly brackets. For example, `D[0:4]{3}` means that three bits are used to identify one of the five PCU (i.e., 000 for PCU0, 001 for PCU1, 010 for PCU2 etc). In this way, event word templates can be straightforwardly built up from various tokens. However, there are two necessary additions which complicate the template somewhat and which require additional explanation: M-tokens ("markers") and Z-tokens ("enumerators").

- **M-tokens:** RXTE science event files are not wholly filled with valid events: some rows contain error flags. Since these "non-events" do not have the same bit structure as the events, alternative event words must be used. DDL marks these alternatives by means of M-tokens.

  In the context of RXTE event words, M-tokens can be thought of as prefixes which distinguish alternative DDL strings. Thus the `TEVTB2` keyword may consist of several alternative DDL strings, each prefixed by an M-token. By examining the M-token of a given row, software can tell which alternative event word template describes the structure of the row.

  M-tokens work like this. The M-token **`M[127]{8}`** means: *The first 8 bits are used as a marker: if its value is 127 (01111111), then the immediately following template is the one that applies to the row.* Similarly, the M-token **`M[4]{8}`** means: *The first 8 bits are used as a marker: if its value is 4 (00000100), then the immediately following template is the one that applies to the row.* By convention, the first marker in a string identifies a valid event. Its form is **`M[1]{1}`**, and, unlike the others in the string, it begins with 1. In practice, when software encounters a 1 in the very first bit of a row, it assumes the "event" template and doesn't interpret the rest of the string as a marker.

  Now let's look at an example - the `TEVTB2` keyword for the `E_1us_1M_0_1s` configuration. For clarity, the M-tokens are emboldened and rearranged on the page to be at the beginning of each line:

  ```
  (M[1]{1},D[0:4]{3},T[0.0:0.00390625;9.5367431640625e-07]{12})^
  ```

```
(M[127]{8},S[One5]{5},S[FirstFlag]{1},S[SpillFlag]{1},S[Zero1]{1})^
(M[1]{8},S[LostEvents0]{8})^
(M[2]{8},S[LostEvents1]{8})^
(M[4]{8},S[Zero]{7},S[Spillage]{1})^
(M[8]{8},S[Zero]{5},S[ModeSpecific]{3})
```

The meaning should now be clear, but please note the following, additional points about M-tokens:

- M-tokens and their associated DDL sub-strings are enclosed in parentheses, e.g. `(M[8]{8},S[Zero]{5},S[ModeSpecific]{3})`.
- The sub-strings in this example are concatenated with DDL's `^` symbol, meaning EOR (exclusive or) and denoting that each string is an exclusive alternative.

- **Z-tokens:** The best way to explain Z-tokens in the RXTE context is by example. Let's consider a PCA configuration that offers PCU and anode ID. Based on the above, the corresponding DDL tokens in the `TEVTB2` keyword would be:

```
D[0:4]{3},E[X1L^X1R^X2L^X2R^X3L^X3R]{6}
```

where, say, 100100000 would mean an event occurred in the X1L anode of PCU4. But given that there are 30 possible combinations of PCU and anode, using a total of nine bits is perhaps excessive. The purpose of Z-tokens is to enumerate these 30 combinations in a systematic way, making it easier for software to identify anode and PCU combinations - and saving disk space.

The way Z-tokens are used is simple. In general:

```
Z[A0, A1, A2... AN-1]{M}
```

means that M bits are reserved in the template to enumerate a list of N items, where N is not greater than 2**M. In our example, the list items are the thirty combinations of PCU and anode:

```
Z[D[0] & E[X1L],D[0] & E[X1R],D[0] & E[X2L],D[0] & E[X2R],D[0] & E[X3L],D[0] & E
  D[1] & E[X1L],D[1] & E[X1R],D[1] & E[X2L],D[1] & E[X2R],D[1] & E[X3L],D[1] & E
  D[2] & E[X1L],D[2] & E[X1R],D[2] & E[X2L],D[2] & E[X2R],D[2] & E[X3L],D[2] & E
  D[3] & E[X1L],D[3] & E[X1R],D[3] & E[X2L],D[3] & E[X2R],D[3] & E[X3L],D[3] & E
```

Footnote 2 - EDS Flags and Coda

EDS Flags: Spillage, ModeSpecific, Lost Events

In addition to columns for time and binned events, PCA science array files also contain columns for flags called `Spillage` and `ModeSpecific`. In science event files, these flags appear in the `Event` column as alternative event word templates, with the addition of the `LostEvents` flag

`Spillage` is used to flag instances when the EDS received more events than it could process and more than zero events were dropped. If no spillage occurred, its value is zero. If spillage occurred in the previous accumulation, then its value is 1.

`LostEvents` (in Event Mode only) contains the number of events that did not fit into the buffer anymore. Beware, it's an unsigned short and may have rolled over.

`ModeSpecific` is only meaningful for Trigger modes. It contains the three trigger flags: low, high and HEXTE.

EDS Coda

Before a new PCA configuration is run by an Event Analyzer, the EDS ends the old run by attaching a ***coda***. The coda contains information about the status of the EDS during the run, including errors and discrepancies between what was processed and what was recorded and telemetred.

The various pieces of coda information end up as keywords at the end of the header of the XTE_SA extension. They are prefaced by the `DDESC` keyword which contains a DDL string indicating the origin of the coda. For example:

```
O[XTE] & I[EDS] & D[0] & E[EA1]
```

shows that the coda in question refers to EA1 on the "A" side of the EDS. Next come the individual coda items comprising:

1. **`ConfigId:`** This field is identical to the Configuration Id in the Parameter Block used by the observation.

2. **`EARunStart:`** This specifies the actual (as opposed to scheduled) starting time of the observation (in MajorTicks).

3. **`PartCount:`** This specifies how many partitions of data were generated (but not necessarily telemetered) by the observation.

4. **`TyperErrCount:`** This is the total number of discrepancies between the encoded Typer Table and the actual Typer Table, after the observation (Value has meaning for PCA Modes Only).

5. **`TyperErrOffset:`** This identifies the Typer Table offset which contained the first discovered discrepancy (Value has meaning for PCA Modes Only).

6. **`TyperErrValue:`** This identifies the value that was read at the Typer Table offset, which didn't match the encoded table (Value has meaning for PCA Modes Only).

7. **`RecodErrCount:`** This is the total number of discrepancies between the encoded Recoder Tables and the actual Recoder Table, after the observation (Value has meaning for PCA Modes Only).

8. **`RecodErrOffset:`** This identifies the Recorder Table offset which contained the first discovered discrepancy (Value has meaning for PCA Modes Only).

9. **`RecodErrValue:`** This identifies the ,value that was read at the Recoder Table offset, which didn't match the encoded table (Value has meaning for PCA Modes Only).

Please proceed to the *PCA Issues* chapter. Or return to the *Table of Contents.*
