RXTE GOF	***Selecting PCA Data by PCU, Layer and Anode*** ***Recipes from the RXTE Cook Book***	***RXTE FAQ***

Introduction: Why Select by PCU, Layer and Anode?

- **PCU:** There are two reasons why it's sometimes desirable to select data by PCU:

  1. For various operational reasons, not all PCUs may be turned on at the same time. If this is the case with your data, it's prudent to extract light curves and spectra only from periods when the number of operating PCUs doesn't change. For spectral analysis, this separation is necessary in order to form the correct response matrix.

  2. The individual PCU have slightly different gains. For most PCA configurations - i.e. those which combine PCU - the individual gains are aligned to a common scale by on-board software before telemetry. However, for Standard-1, Standard-2, Good Xenon and Transparent configurations, the individual gains are uncorrected. This means that to avoid blurring the response, thereby reducing spectral resolution, data from the individual PCU should be analyzed separately. [But note that combining PCU obviously improves signal-to-noise which is often more important than improving resolution by separating the PCUs. In this case, pcarsp will take these factors into account.]

- **Layer/Anode:** The PCU have three xenon layers (X1, X2, X3) each consisting of two anode chains (Left, Right). Although there is some variation with energy, in the low band (i.e. < 10 keV) the top layer detects roughly 90 percent of the cosmic photons and 50 percent of the internal instrumental background. It's easy to see that you can improve signal-to-noise for weak sources by selecting only events from the top layer (layer-1) and excluding events from the mid and bottom layers (layer-2, layer-3). The definition of "weak" is necessarily fuzzy, since it depends on source spectrum, but as a rule of thumb you can use a rate of 20 counts per second per PCU. Obviously, this does not apply to hard sources where proportionately more source events end up in the lower layers. [Note that the proposal tool recommd actually provides estimates for the expected signal-to-noise ratio by layer.]

Science Array Data:

For science array data selecting by PCU, layer or anode amounts to the same thing: specifying to saextrct the columns corresponding to your selection.

1. Verify the names of the columns in your data files by running the ftool flcol on a sample file. In the case of a Standard-2 file, you'll get:

```
Name of FITS file and [ext#][] standard2.fits
___Column_Names__________Formats______Dims______Units___
Time                       D                     s
X1LSpecPcu0                129I      (129)      count
X1RSpecPcu0                129I      (129)      count
X2LSpecPcu0                129I      (129)      count
X2RSpecPcu0                129I      (129)      count
X3LSpecPcu0                129I      (129)      count
X3RSpecPcu0                129I      (129)      count
X1LSpecPcu1                129I      (129)      count
X1RSpecPcu1                129I      (129)      count
X2LSpecPcu1                129I      (129)      count
X2RSpecPcu1                129I      (129)      count
X3LSpecPcu1                129I      (129)      count
X3RSpecPcu1                129I      (129)      count
.                          .          .          .
.                          .          .          .
.                          .          .          .
```

   Standard-2 is the only science array configuration that offers full PCU and anode ID. The column X1LSpecPcu0, for example, corresponds to the left anode of the first (top) layer of PCU0. By contrast, Standard-1 only provides PCU ID. Its columns have names like XeCntPcu0.

2. Decide from which PCU, layers or anodes you want to extract data.

3. Create an ASCII file listing the columns you want. One name per line, ending with a carriage return. For example, the list of Standard-2 columns

```
X1LSpecPcu0
X1RSpecPcu0
X1LSpecPcu1
X1RSpecPcu1
X1LSpecPcu2
X1RSpecPcu2
```

   corresponds to the top-layer of PCU0-2

4. Enter the name of the column list, prefaced with an @-sign at the saextrct prompt

```
Name of COLUMNS to be accumulated (GOOD):[GOOD]
```

Science Event Data:

For science event data selecting a particular PCU or anode entails applying the corresponding bitmask either in **fselect** or in **seextrct**. For your convenience, we have produced a set of the most common bitmasks needed for GoodXenon and other event modes. If one of these will suit your purpose, you may avoid using sefilter, which is the script which will produce these if you know the correct event token values. This recipe will describe how to use one of these bitmasks for the novice user.

If you don’t find the bitmask you need, we have a general recipe for using sefilter and the DDL which is meant for users who need something we haven’t anticipated. This script calls **sebitmask** to create, using the values you specify, an expression which either **fselect** or **seextrct** may read. In order to get the correct result from this tool, it is important that you understand both the DDL in general and your data descriptor in specific.

- If you go to our common bitmasks page, you will first see a table of event modes telling you what you can select in that mode and which table of bitmasks to use. Those tables have the appropriate bitmasks for the various common combinations of PCU’s and layers which are available for the given modes. Find the combination you want and download the bitfile into your working directory. This bitfile is an expression which basically says which values of the event word you want. Some can be read either by fselect, which will produce a FITS file in the same format but only including the desired events, or by seextrct, which will extract light curves and spectra from only those events.

- If the bitfile is marked with an asterisk next to it, then this bitmask has a format which cannot be used in seextrct. If your aim, then, is to produce a spectrum or light curve, you must filter the raw data with fselect first. Then you can run the extractor on the resulting, filtered event list. If the bitfile does not have an asterisk, then you may immediately run seextrct using that bitfile.

- **fselect** is a very simple FTOOL which takes an input FITS file and an expression in terms of one of the columns; it then outputs another FITS file in the same format but shorter, including only those rows where the values in that column match the expression. In the case of event lists, the expression (called a "bitmask") is stored in the bitfile, which you enter at the prompt preceded by an ’@’ character. In the following example, the GoodXenon1 and GoodXenon2 files have been combined using **make_se** to form the FITS file obs_01_01_gx0, which is the file we want to filter for detectors 0,1,2, and 4 (for instance, if detector 3 is off for part of the time range and we want a consistent light curve):

  ```
  % fselect
  Name of FITS file and [ext#][] obs_01_01_gx0
  Name of output FITS file[] obs_01_01_gx0_d0124
  Selection Expression[] @bitfile_d0124
  ```

  This may take some time, as fselect must compare every event in that file with the expression stored in the bitfile.

- **seextrct** has a hidden parameter appropriately called "bitfile" for entering this expression. Remember that it can only handle certain types of expressions. For a simpler expression than in the above fselect example, we can use "**seextrct bitfile=bitfile__lr12**" and then answer the prompts as usual.

If you have a question about RXTE, please send email to one of our help desks. This page is maintained by the RXTE GOF and was last modified on Friday, 19-Jun-1998 17:59:57 EDT.

Part of the NASA OSS *Structure and Evolution of the Universe* theme.

XTE Project Scientist Jean Swank, swank@pcasun1.gsfc.nasa.gov
Technical Rep: Sherri Calvo, sherri.calvo@gsfc.nasa.gov