

**spi\_psd\_buildlib**

# **User Manual**

**Version 1.5.0**  
**23 September 2002**

Jürgen Knödseder  
Centre d'Etude Spatiale des Rayonnements  
knödseder@cesr.fr  
<http://www.cesr.fr/~jurgen/index.html>

#### Note to the user

This software has been written to analyse data of the SPI telescope onboard INTEGRAL. Particular care has been taken in making the software user friendly and well documented. If you appreciated this effort, and if this software and User Manual were useful for your scientific work, the author would appreciate a corresponding acknowledgment in your published work.

## **Contents**

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Getting started</b>	<b>1</b>
<b>3</b>	<b>Parameter file</b>	<b>3</b>
<b>4</b>	<b>Interface definition</b>	<b>9</b>
<b>5</b>	<b>Algorithm</b>	<b>11</b>
<b>6</b>	<b>Error codes</b>	<b>11</b>

## 1 Introduction

The executable `spi_psd_buildlib` is used to generate PSD template libraries from PSD curve data. Library templates are generally built by adding and averaging PSD pulse shapes that were downlinked from the PSD sub-assembly. Pulse shape selections may be employed to extract specific subsets from the data (e.g. energy limits, pulse shape noise, etc.). PSD pulse shape libraries consist of template pulses that are sorted by the time-to-peak (TTP) value, i.e. the time the pulse takes to rise to its maximum value. Pulse shape libraries may differ from detector to detector, hence template sets are generated for all 19 detection chains.

Normally, PSD pulse shapes are gathered in a special scientific mode called the Calibration mode. In this mode a maximum curve transmission rate of 40 curves/second may be achieved. However, PSD pulse shapes are also downlinked in Operational and Diagnostics mode. Typically, one pulse shape is received each 4 seconds in Operational mode, while the pulse shape rate in Diagnostics mode depends on the configuration (it may also reach 40 curves/second). PSD pulse shapes from all these modes may be used for template library building.

PSD library templates may be built from single detector curves (these are curves that correspond to events that triggered only a single channel of the AFEE energy chain) or from multiple detector curves (these are curves that correspond to events that triggered multiple channels of the AFEE energy chain).

Normally, multiple detector events are recognised by the PSD sub-assembly and are rejected, hence there should be not multiple detector curves. However, the fact that the AFEE generally has a lower trigger threshold than the PSD can lead to the situation that multiple detector events are detected by AFEE but not the PSD. In this case, all partners except of one must have an energy below the PSD energy threshold (what counts here is the FET energy threshold since it is this threshold that is used for multiple event rejection). Multiple detector curves are not correlated by the DPE, yet all information is available in the telemetry to perform the correlation onground (the routine `PSDReadMultCurves` of the `spi_psdlib` library automatically performs this task).

The PSD sub-assembly memory may hold 2 library template sets, yet `spi_psd_buildlib` builds the template libraries always for the set 0. To produce libraries ready for uplink and to allow for loading into PSD template set 1, the task `spi_psd_lib2moc` should be used.

The executable `spi_psd_buildlib` is written in the ANSI C++ language. It has been developed under ISDC support platform 4.1 and requires `spi_psdlib` 1.6.0 and `spi_toolslib` 1.8.0 or higher.

## 2 Getting started

Before installing `spi_psd_buildlib`, make sure that the ISDC support platform 4.1 or higher is installed on your system, and that you have installed the libraries `spi_psdlib` 1.6.0 and `spi_toolslib` 1.8.0 or higher.

After downloading the `spi_psd_buildlib.tar.gz` file, step into a directory that should hold the distribution, move the `spi_psd_buildlib.tar.gz` file into this directory and type:

```
$ gunzip spi_psd_buildlib.tar.gz
$ tar xvf spi_psd_buildlib.tar
```

The first command uncompresses the distribution file, the second unpacks the files.

Before configuration, the distribution needs to be reset to a clean state. To do this, type

```
$ make distclean
```

Then, configure the distribution. It is assumed here that you have previously installed the ISDC support platform, thus you should type

```
$ ~/bin/ac_stuff/configure
```

Finally, build the distribution by typing

```
$ make global_install
```

To perform a unit test, type

```
$ make test
```

Make sure that the test data `spi_test_data-1.0.tar.gz` are available at your site (they should reside in a directory whose name is defined by the `ISDC_TEST_DATA_DIR` environment variable).

### 3 Parameter file

```
#####
#
#           Centre d'Etude Spatiale des Rayonnements           #
#           (in collaboration with ISDC)                       #
#
#           SPI PSD ANALYSIS                                   #
#
# -----#
#
# File:      spi_psd_buildlib.par                               #
# Version:   1.5.0                                             #
# Component: PA                                               #
#
# Author:    Juergen Knoedlseder                               #
#            knodlseder@cesr.fr                               #
#            CESR                                             #
#
# Purpose:   Parameter file of the SPI PSD library building   #
#            executable                                         #
#
# History:   1.5.0 23-Sep-2002 First ISDC delivery (Rev. 5.0) #
#
#####
#
# The input DOLs
#=====
curveDOL, s,ql, "og_spi.fits[1]",,, "Input DOL (SWG/IDX/OG)"
algorparDOL,s,ql, "algFM280701.fits[SPI.-ALGO-PSD]",,, "PSD algorithm parameters DOL"
#
# The output DOLs
#=====
libDOL,s,ql, "lib.fits",,, "PSD result library DOL or filename"
outDOL,s,ql, "curves.fits",,, "Selected PSD curves output filename"
#
# OBT limits
#=====
minOBT,s,h, "",,, "Curve usage minimum OBT"
maxOBT,s,h, "",,, "Curve usage maximum OBT"
gtiDOL,s,h, "",,, "Curve usage GTI DOL (GNRL/CALI/OBS)"
#
# Task parameters
#=====
mult,      i,ql, 2,1, 3, "PSD curve multiplicity (1-3)"
timesteps, i,h, 64,0,64, "Number of time steps to use for library"
adc0_gain, r,h, 1.0,, "Gain PSD ADC 0"
adc1_gain, r,h, 1.0,, "Gain PSD ADC 1"
adc2_gain, r,h, 1.0,, "Gain PSD ADC 2"
adc3_gain, r,h, 1.0,, "Gain PSD ADC 3"
adc0_offset,r,h, 0.0,, "Offset PSD ADC 0"
adc1_offset,r,h, 0.0,, "Offset PSD ADC 1"
adc2_offset,r,h, 0.0,, "Offset PSD ADC 2"
```

```

adc3_offset,r,h,    0.0,,, "Offset PSD ADC 3"
#
# Energy filters
#=====
engmin,   r,ql, 1800.0,0.0,8000.0, "Total energy minimum (keV)"
engmax,   r,ql, 1900.0,0.0,8000.0, "Total energy maximum (keV)"
mp3engmin,r,ql,   0.0,0.0,8000.0, "Curve energy minimum (keV)"
mp3engmax,r,ql, 8000.0,0.0,8000.0, "Curve energy maximum (keV)"
mp4engmin,r,ql,   0.0,0.0,8000.0, "Scattered energy minimum (keV)"
mp4engmax,r,ql, 8000.0,0.0,8000.0, "Scattered energy maximum (keV)"
#
# Curve error rejection
#=====
flttonboard, b,h, no,,, "Reject curves with on-board error ?"
fltonground,b,h, no,,, "Reject curves with on-ground error ?"
#
# Single site curve usage (post processing and fitting)
#=====
fltusesingle,b,h,   no,,, "Use only single-site curves ?"
fltpost,      b,h,   no,,, "Do discrimination post-processing ?"
disDOL,       s,h,   "",,, "Discrimination parameter DOL"
fltlib,       b,h,   no,,, "Use library for on-ground fitting ?"
fltlibDOL,    s,h,   "",,, "PSD filter library DOL"
fltlibset,    i,h,   0,0, 1, "Template library set used for fitting (0/1)"
fltlibsteps,  i,h,   64,0,64, "Number of time steps used for fitting"
fltlibtpls,   i,h,   0,0,38, "Number of templates used for fitting"
#
# Starttime filter
#=====
fltstart,     b,h,           no,,, "Use starttime filter ?"
fltstartmin,  r,h,   0.0,   0.0, 80.0, "Starttime minimum (absolute)"
fltstartmax,  r,h,   80.0,   0.0, 80.0, "Starttime maximum (absolute)"
fltsstartmin,r,h,  -3.0,-100.0,100.0, "Starttime minimum (sigma)"
fltsstartmax,r,h,  +3.0,-100.0,100.0, "Starttime maximum (sigma)"
#
# Endtime filter
#=====
fltend,       b,h,           no,,, "Use endtime filter ?"
fltendmin,    r,h,   0.0,   0.0, 80.0, "Endtime minimum (absolute)"
fltendmax,    r,h,   80.0,   0.0, 80.0, "Endtime maximum (absolute)"
fltsendmin,r,h, -3.0,-100.0,100.0, "Endtime minimum (sigma)"
fltsendmax,r,h, +3.0,-100.0,100.0, "Endtime maximum (sigma)"
#
# Duration filter
#=====
fltdur,       b,h,           no,,, "Use duration filter ?"
fltdurmin,    r,h,   0.0,   0.0, 80.0, "Duration minimum (absolute)"
fltdurmax,    r,h,   80.0,   0.0, 80.0, "Duration maximum (absolute)"
fltsdurmin,r,h, -3.0,-100.0,100.0, "Duration minimum (sigma)"
fltsdurmax,r,h, +3.0,-100.0,100.0, "Duration maximum (sigma)"
#
# Baseline filter
#=====

```

```

fltbase,    b,h,                no,,, "Use baseline filter ?"
fltbasemin, r,h, 20.0,    0.0,511.0, "Baseline minimum (absolute)"
fltbasemax, r,h, 80.0,    0.0,511.0, "Baseline maximum (absolute)"
fltsbasemin,r,h, -3.0,-100.0,100.0, "Baseline minimum (sigma)"
fltsbasemax,r,h, +3.0,-100.0,100.0, "Baseline maximum (sigma)"
#
# Baseline slope filter
#=====
fltbaseslope,    b,h,                no,,, "Use baseline slope filter ?"
fltbaseslopemin, r,h, -1.0, -10.0, 10.0, "Baseline slope minimum (absolute)"
fltbaseslopemax, r,h, +1.0, -10.0, 10.0, "Baseline slope maximum (absolute)"
fltsbaseslopemin,r,h, -3.0,-100.0,100.0, "Baseline slope minimum (sigma)"
fltsbaseslopemax,r,h, +3.0,-100.0,100.0, "Baseline slope maximum (sigma)"
#
# Baseline slope significance filter
#=====
fltbasesig,    b,h,                no,,, "Use baseline slope significance filter ?"
fltbasesigmin,r,h, -3.0,-100.0,100.0, "Baseline slope significance minimum (absolute)"
fltbasesigmax,r,h, +3.0,-100.0,100.0, "Baseline slope significance maximum (absolute)"
#
# Noise filter
#=====
fltnoise,    b,h,                no,,, "Use noise filter ?"
fltnoisemin, r,h, 0.0,    0.0,511.0, "Noise minimum (absolute)"
fltnoisemax, r,h, 20.0,    0.0,511.0, "Noise maximum (absolute)"
fltsnoisemin,r,h, -3.0,-100.0,100.0, "Noise minimum (sigma)"
fltsnoisemax,r,h, +3.0,-100.0,100.0, "Noise maximum (sigma)"
#
# Chisqr filter
#=====
fltchisqr,    b,h,                no,,, "Use Chi squared filter ?"
fltchisqrmin, r,h, 0.0,    0.0,10000.0, "Chi squared minimum (absolute)"
fltchisqrmax, r,h, 10000.0,    0.0,10000.0, "Chi squared maximum (absolute)"
fltschisqrmin,r,h, -3.0,-100.0, 100.0, "Chi squared minimum (sigma)"
fltschisqrmax,r,h, +3.0,-100.0, 100.0, "Chi squared maximum (sigma)"
#
# Standard parameters
#=====
clobber,b,ql, no,,, "Overwrite existing template library ?"

```

The following parameters have to be specified:

- **curveDOL** specifies the input group that holds the PSD curves that should be processed. The input group may be either a single science window group, an index group or an observation group. An ISDC level of **COR** is required (for access to corrected energies).
- **algotparDOL** specifies the PSD onboard algorithm parameter DOL of HDU type **[SPI.-ALGO-PSD]** that contains the parameters that were loaded to the PSD sub-assembly while the PSD curves were recorded. These algorithm parameters are required for PSD curve characterisation.
- **libDOL** specifies the output library DOL or filename. The library HDU extension **[SPI.-LIB.-PSD]** may be added to provide a complete DOL, yet it is not required.
- **outDOL** (optional) specifies the filename of the output file that should contain a science window group that holds all PSD curves that were selected for library building. If this parameter is left

blank, no output group will be created. Otherwise, a group of type [GNRL-SCWG-GRP] will be created that holds the three HDUs [SPI.-CCRV-RAW], [SPI.-CCRV-PRP], and [SPI.-CCRV-COR]. Note that irrespectively of the curve multiplicity (**mult**), the curves are stored as single detector curves (**mult=1**). This group may be fed as input to a further run of **spi\_psd\_buildlib**.

- **minOBT** (optional) specifies the minimum OBT of the curves that should be considered for library building. If the parameter is left blank, no minimum OBT will be applied.
- **maxOBT** (optional) specifies the maximum OBT of the curves that should be considered for library building. If the parameter is left blank, no maximum OBT will be applied.
- **gtiDOL** (optional) specifies the DOL of a Good Time Interval data structure of HDU type [SPI.-GNRL-GTI] or [SPI.-OBS.-GTI] that should be used to select specific time intervals for PSD curve selection. If the parameter is left blank, no GTI selection will be applied.
- **mult** specifies the curve multiplicity, i.e. the number of Germanium detectors that triggered the AFEE subassembly while the PSD trigger occurred. The normal setting is **1** and corresponds to single detector events. However, for calibration purposes multiple detector interactions may be preferred, and one may select double detector events (**2**) or tripple detector events (**3**).
- **timesteps** specifies the number of time steps of the result library. The default value is **64** and it is not recommended to change this value.
- **adc0\_gain** specifies the gain correction that should be applied for the PSD ADC convertor 0. As default select 1.0.
- **adc1\_gain** specifies the gain correction that should be applied for the PSD ADC convertor 1. As default select 1.0.
- **adc2\_gain** specifies the gain correction that should be applied for the PSD ADC convertor 2. As default select 1.0.
- **adc3\_gain** specifies the gain correction that should be applied for the PSD ADC convertor 3. As default select 1.0.
- **adc0\_offset** specifies the offset correction that should be applied for the PSD ADC convertor 0. As default select 0.0.
- **adc1\_offset** specifies the offset correction that should be applied for the PSD ADC convertor 1. As default select 0.0.
- **adc2\_offset** specifies the offset correction that should be applied for the PSD ADC convertor 2. As default select 0.0.
- **adc3\_offset** specifies the offset correction that should be applied for the PSD ADC convertor 3. As default select 0.0.
- **engmin** together with **engmax**, specifies an energy window for the total energy of the PSD curves. Only curves with a total energy that falls into this window will be selected for library building. Total energy means that in case of a multiple detector curve, all partial energies will be summed. In the case of a single detector curve, the total energy is simply the energy of the curve.
- **engmax** together with **engmin**, specifies an energy window for the total energy of the PSD curves (see above).
- **mp3engmin** (optional) together with **mp3engmax**, specifies an energy window for the energy of PSD curves if the requested event multiplicity is **mult**  $\geq 2$ . This selection allows to select PSD curves with energies within a given window, for double or triple detector curves.

- **mp3engmax** together with **mp3engmin**, specifies an energy window for the energy of the PSD curves (see above).
- **mp4engmin** (optional) together with **mp4engmax**, specifies an energy window for the energy deposited in the auxiliary detector if the requested event multiplicity is **mult = 3**. This selection allows to select special interaction combinations, in particular related to pair creation within a given detector.
- **mp4engmax** together with **mp4engmin**, specifies an energy window for the auxiliary detector (see above).
- **fltonboard** specifies if curves with onboard analysis error should be used or not. Note that for multiple detector curves, no onboard error information is available, hence in this case no onboard error filtering should be done. It is anyways recommended to use onground filtering, hence as default this parameter should be set to **0**.
- **fltonground** specifies if curves should be filtered onground. This is the recommended option (hence set the parameter to **1**). Onground filtering means that the curves are checked against several criteria (pulse duration, baseline level, pulse saturation, etc.) that allow to reject curves that can not be analysed by the PSD software. The criteria of the check are defined by the algorithm parameters **DOL** specified by **algotparDOL**.
- **fltusesingle** specifies if only single-site events should be used to build the library. In principle a template library should only contain single-site events and one is tempted to set this option to **0** by default. Yet the judgment if an event is single-site or multiple-site is already based on another library, and it should be expected that selecting only single-site events for library building should introduce some bias towards a new library that resembles the old one. This domain is still a field of ongoing studies, and it may be useful to use this option with some relaxed discrimination criteria.
- **fltpost** (optional) if only single-site events should be used (**fltusesingle = yes**), this parameter specifies if the curves should be post-processed (i.e. if the pulse shape discrimination should be done onground using user-specified discrimination parameters) or if the onboard decision should be used (**fltpost = yes** leads to post-processing, **no** means using the onboard discrimination).
- **disDOL** (optional) if post-processing is selected for single-site event identification (**fltpost = yes**) this parameter specifies the **DOL** of the discrimination parameters. One may use the task **spi\_psd\_optimise** to obtain optimum discrimination parameters.
- **fltlib** (optional) if post-processing is selected for single-site event identification (**fltpost = yes**) this parameter specifies if the onboard library fitting results should be used for discrimination (**fltlib = no**) or if the pulse shapes should be fitted onground with a user-specified library (**fltlib = yes**). In general, if post-processing is selected the user wants to specify the library that is used for fitting, in particular if the library should be determined iteratively.
- **fltlibDOL** (optional) if pulse shapes should be fitted onground for post-processing (**fltlib = yes**), this parameter specifies the **DOL** of the library that should be used for fitting.
- **fltlibset** (optional: if pulse shapes should be fitted onground for post-processing (**fltlib = yes**), this parameter specifies the template set number of the library that should be used for fitting (the setting is identical for all detectors).
- **fltlibsteps** (optional) if pulse shapes should be fitted onground for post-processing (**fltlib = yes**), this parameter specifies the number of time-steps that should be used for fitting. Note that this number may not exceed the number of time-steps available in the library (the setting is identical for all detectors). The default value is **64**.
- **fltlibtpls** (optional) if pulse shapes should be fitted onground for post-processing (**fltlib = yes**), this parameter specifies the number of templates that should be used for fitting. Note that this

number may not exceed the number of templates available in the library (the setting is identical for all detectors). If this parameter is set to 0, all available library templates will be used for fitting (i.e. different number of templates may be used for different detectors).

- **fltstart** specifies if pulse shapes with starttime outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltstartmin**, **fltstartmax**] or by specifying relative limits using the parameters [**fltsstartmin**, **fltsstartmax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltstartmin} = \text{mean} - \text{rms} \times \text{fltsstartmin}$  and  $\text{fltstartmax} = \text{mean} + \text{rms} \times \text{fltsstartmax}$ .
- **fltend** specifies if pulse shapes with endtime outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltendmin**, **fltendmax**] or by specifying relative limits using the parameters [**fltsendmin**, **fltsendmax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltendmin} = \text{mean} - \text{rms} \times \text{fltsendmin}$  and  $\text{fltendmax} = \text{mean} + \text{rms} \times \text{fltsendmax}$ .
- **fltdur** specifies if pulse shapes with duration outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltdurmin**, **fltdurmax**] or by specifying relative limits using the parameters [**fltsdurmin**, **fltsdurmax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltdurmin} = \text{mean} - \text{rms} \times \text{fltsdurmin}$  and  $\text{fltdurmax} = \text{mean} + \text{rms} \times \text{fltsdurmax}$ .
- **fltbase** specifies if pulse shapes with baseline outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltbasemin**, **fltbasemax**] or by specifying relative limits using the parameters [**fltsbasemin**, **fltsbasemax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltbasemin} = \text{mean} - \text{rms} \times \text{fltsdurmin}$  and  $\text{fltbasemax} = \text{mean} + \text{rms} \times \text{fltsdurmax}$ .
- **fltbaseslope** specifies if pulse shapes with baseline slope outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltbaseslopemin**, **fltbaseslopemax**] or by specifying relative limits using the parameters [**fltsbaseslopemin**, **fltsbaseslopemax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltbaseslopemin} = \text{mean} - \text{rms} \times \text{fltsbaseslopemin}$  and  $\text{fltbaseslopemax} = \text{mean} + \text{rms} \times \text{fltsbaseslopemax}$ .
- **fltbasesig** specifies if pulse shapes with baseline slope significance outside a specific interval should be rejected. The interval is defined by specifying absolute limits using the parameters [**fltbasesigmin**, **fltbasesigmax**].
- **fltnoise** specifies if pulse shapes with noise outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltnoisemin**, **fltnoisemax**] or by specifying relative limits using the parameters [**fltsnoisemin**, **fltsnoisemax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltnoisemin} = \text{mean} - \text{rms} \times \text{fltsnoisemin}$  and  $\text{fltnoisemax} = \text{mean} + \text{rms} \times \text{fltsnoisemax}$ .
- **fltchisqr** specifies if pulse shapes with Chi squared outside a specific interval should be rejected. The interval is defined by either specifying absolute limits using the parameters [**fltchisqrmin**, **fltchisqrmax**] or by specifying relative limits using the parameters [**fltschisqrmin**, **fltschisqrmax**]. The relative limits are calculated channel by channel, and are specified in units of rms around the mean values, i.e.  $\text{fltchisqrmin} = \text{mean} - \text{rms} \times \text{fltschisqrmin}$  and  $\text{fltchisqrmax} = \text{mean} + \text{rms} \times \text{fltschisqrmax}$ .
- **clobber** specifies if existing library templates should be overwritten (**clobber** = **yes**) or if curves should be added onto an existing library (**clobber** = **no**). The same logic applies also for the PSD curves that were selected and are optionally written to the filename specified by **outDOL**. **clobber** = **yes** will delete an existing output data structure, while **clobber** = **no** will lead to adding of additional data.

## 4 Interface definition

On input, `spi_psd_buildlib` reads a science window group, an index group, or an observation group of level `COR`. If the library DOL exists already and the `clobber` parameter is set to `no`, the library is loaded into memory and the new templates that are gathered from the input group are added (i.e. an additional run is performed, see below).

On output, `spi_psd_buildlib` produces library templates for all detectors it has found in the input group and creates a `SPI.-LIB.-PSD` structure. If the output DOL already exists it is either replaced (if `clobber` is set to `yes`) or a new run is added (if `clobber` is set to `no`). By default, `spi_psd_buildlib` always creates library templates for the PSD template set 0. Before uplinking the libraries they should be processed by the task `spi_psd_lib2moc` which is the only task that attributes template set numbers different from 0.

The library templates are stored in a single result file with a single HDU of type `SPI.-LIB.-PSD` to which the following keywords are added by `spi_psd_buildlib`:

```

CREATOR = 'spi_psd_buildlib' / Program that created this FITS file
CONFIGUR= '1.5.0' / Software configuration
NRUNS = 1 / Number of runs in library
...
CREATO01= 'spi_psd_buildlib' / Run 1 creating task
CONFIG01= '1.5.0' / Run 1 task configuration
SOURCE01= 'og_spi.fits[1]' / Library source DOL
ALGDOL01= 'alg.fits[SPI.-ALGO-PSD]' / PSD algorithm parameter DOL
MULT01 = 1 / PSD curve multiplicity (1-3)
ADC_G001= 1. / ADC #0 gain
ADC_G101= 1. / ADC #1 gain
ADC_G201= 1. / ADC #2 gain
ADC_G301= 1. / ADC #3 gain
ADC_0001= 0. / ADC #0 offset
ADC_0101= 0. / ADC #1 offset
ADC_0201= 0. / ADC #2 offset
ADC_0301= 0. / ADC #3 offset
ENGMIN01= 0. / [keV] Total energy minimum
ENGMAX01= 8000. / [keV] Total energy maximum
M3EMIN01= 0. / [keV] Curve detector energy minimum
M3EMAX01= 8000. / [keV] Curve detector energy maximum
M4EMIN01= 0. / [keV] Auxillary detector energy minimum
M4EMAX01= 8000. / [keV] Auxillary detector energy maximum
FLTSGLO1= 0 / Use only single-site events (0=no, 1=yes)
DISONG01= 0 / Onground pulse shape discrimination (0=no, 1=yes)
DISDOL01= 'dis.fits[SPI.-DISC-PSD]' / PSD discrimination parameters DOL
FLTLIB01= 0 / Use filter library (0=no, 1=yes)
LIBDOL01= 'lib.fits[SPI.-LIB.-PSD]' / Filter library DOL
LIBSET01= 0 / Filter library template set
LIBSTP01= 64 / Filter library number of time steps
LIBTPL01= 23 / Filter library number of templates
FLTONB01= 0 / Use onboard error filter (0=no, 1=yes)
FLTONG01= 1 / Use onground error filter (0=no, 1=yes)
FLTSTA01= 1 / Use starttime filter (0=no, 1=yes)
STAMIN01= 0.0 / Starttime filter minimum
STAMAX01= 80.0 / Starttime filter maximum
STASMNO1= -3.0 / Starttime filter minimum (sigma)
STASMX01= 3.0 / Starttime filter maximum (sigma)

```

```

FLTEND01=          1 / Use endtime filter (0=no, 1=yes)
ENDMIN01=          0.0 / Endtime filter minimum
ENDMAX01=          80.0 / Endtime filter maximum
ENDSMN01=         -3.0 / Endtime filter minimum (sigma)
ENDSMX01=          3.0 / Endtime filter maximum (sigma)
FLTDUR01=          1 / Use duration filter (0=no, 1=yes)
DURMIN01=          0.0 / Duration filter minimum
DURMAX01=          80.0 / Duration filter maximum
DURSMN01=         -3.0 / Duration filter minimum (sigma)
DURSMX01=          3.0 / Duration filter maximum (sigma)
FLTBAS01=          1 / Use baseline filter (0=no, 1=yes)
BASMIN01=          20.0 / Baseline filter minimum
BASMAX01=          80.0 / Baseline filter maximum
BASSMN01=         -3.0 / Baseline filter minimum (sigma)
BASSMX01=          3.0 / Baseline filter maximum (sigma)
FLTSLP01=          1 / Use baseline slope filter (0=no, 1=yes)
SLPMIN01=         -1.0 / Baseline slope filter minimum
SLPMAX01=          1.0 / Baseline slope filter maximum
SLPSMN01=         -3.0 / Baseline slope filter minimum (sigma)
SLPSMX01=          3.0 / Baseline slope filter maximum (sigma)
FLTSLS01=          1 / Use baseline slope sig. filter (0=no, 1=yes)
SLSMIN01=         -3.0 / Baseline slope sig. filter minimum
SLSMAX01=          3.0 / Baseline slope sig. filter maximum
FLTNOI01=          1 / Use noise filter (0=no, 1=yes)
NOIMIN01=          0.0 / Noise filter minimum
NOIMAX01=          20.0 / Noise filter maximum
NOISMN01=         -3.0 / Noise filter minimum (sigma)
NOISMX01=          3.0 / Noise filter maximum (sigma)
FLTCHI01=          1 / Use Chi squared filter (0=no, 1=yes)
CHIMIN01=          0.0 / Chi squared filter minimum
CHIMAX01=         10000.0 / Chi squared filter maximum
CHISMN01=         -3.0 / Chi squared filter minimum (sigma)
CHISMX01=          3.0 / Chi squared filter maximum (sigma)
NCRV0001=          723 / Number of curves used for detector 0
NCRV0101=          723 / Number of curves used for detector 1
...
NCRV1801=          723 / Number of curves used for detector 18

```

In general, these keywords reflect the task parameters of `spi_psd_buildlib` which greatly enhances the tracability of PSD library building.

A library HDU may contain several runs which indicates that several library sources were added together (this may happen if `clobber` is set to `no`). The number of runs that are within a library is specified by the `NRUNS` keyword. Following `NRUNS`, there are keywords which are added for each run (the run number is given by the last two digits of the keywords). The meaning of the keywords can be understood from the above discription of the task parameters.

Following the task parameter keywords, `NCRV $xyy$`  summarises the total number of pulse shapes that were gathered for a run from the data for each detector.  $xx$  specifies the detector ID (0-18) while  $yy$  is the run number. These keywords are identical to the keywords that are set by the task `spi_psd_calclib`, and allow a global tracking of the library template content.

`spi_psd_buildlib` may optionally also create a science window group of level `COR` that holds all PSD curves that were selected for library building. This science window group may be fed also as input to `spi_psd_buildlib`, allowing for quick iterative PSD template library bulding. Thus, a first run of

`spi_psd_buildlib` will be used to gather all relevant curves from the observation group (by applying some energy and Good Time Interval filter), while the subsequent runs (with pulse filtering, ADC gain correction, etc.) may then read the selected curves from the science window group.

## 5 Algorithm

No special algorithm is defined so far.

## 6 Error codes

The following error codes are defined:

`SPI_PSD_BUILDLIB_ERROR_MEM_ALLOC`            `-230400`

They have the following meaning:

- `SPI_PSD_BUILDLIB_ERROR_MEM_ALLOC` : the dynamical allocation of memory failed. Probably you do not have sufficient system resources to run this executable.