

**spi\_psd\_calclib**

# User Manual

Version 1.1.0  
1 September 2002

Jürgen Knödseder  
Centre d'Etude Spatiale des Rayonnements  
knodseder@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>2</b>
3.1	Copy a library . . . . .	4
3.2	Detector averaging . . . . .	4
<b>4</b>	<b>Interface definition</b>	<b>5</b>
<b>5</b>	<b>Error codes</b>	<b>6</b>

## 1 Introduction

The executable `spi_psd_calclib` allows to perform arithmetics on PSD template libraries such as template addition and averaging. It may also be used to copy library templates from one detector to another.

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

## 2 Getting started

Before installing `spi_psd_calclib`, 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` version 1.6.0 or higher and `spi_toolslib` version 1.8.0 or higher.

After downloading the `spi_psd_calclib.tar.gz` file, step into a directory that should hold the distribution, move the `spi_psd_calclib.tar.gz` file into this directory and type (don't type the prompt \$):

```
$ gunzip spi_psd_calclib.tar.gz
$ tar xvf spi_psd_calclib.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
```

### 3 Parameter file

```
#####
#
#           Centre d'Etude Spatiale des Rayonnements           #
#           (in collaboration with ISDC)                       #
#
#           SPI PSD ANALYSIS                                   #
#
# -----#
#
# File:      spi_psd_calclib.par                               #
# Version:   1.1.0                                           #
# Component: PA                                              #
#
# Author:    Juergen Knoedlseder                             #
#            knodlseder@cesr.fr                              #
#            CESR                                            #
#
# Purpose:   Parameter file of the SPI PSD library calculation #
#            executable                                       #
#
# History:   1.1.0  1-Sep-2002  First ISDC delivery          #
#
#####
#
# The input library DOLs
#=====
inlibDOL0,s,ql,"lib/psd_analysis_std.fits",,, "PSD library DOL for filename #0"
inlibDOL1,s,ql,"",,, "PSD library DOL for filename #1"
inlibDOL2,s,ql,"",,, "PSD library DOL for filename #2"
inlibDOL3,s,ql,"",,, "PSD library DOL for filename #3"
inlibDOL4,s,ql,"",,, "PSD library DOL for filename #4"
#
# The output library DOL
#=====
outlibDOL,s,ql,"psd_library.fits",,, "PSD result library DOL or filename"
#
# Library arithmetics
#=====
detid0, s,ql,"#0:0",,, "Detector 0 arithmetics"
detid1, s,ql,"#0:0",,, "Detector 1 arithmetics"
detid2, s,ql,"#0:0",,, "Detector 2 arithmetics"
detid3, s,ql,"#0:0",,, "Detector 3 arithmetics"
detid4, s,ql,"#0:0",,, "Detector 4 arithmetics"
detid5, s,ql,"#0:0",,, "Detector 5 arithmetics"
detid6, s,ql,"#0:0",,, "Detector 6 arithmetics"
detid7, s,ql,"#0:0",,, "Detector 7 arithmetics"
detid8, s,ql,"#0:0",,, "Detector 8 arithmetics"
detid9, s,ql,"#0:0",,, "Detector 9 arithmetics"
detid10,s,ql,"#0:0",,, "Detector 10 arithmetics"
detid11,s,ql,"#0:0",,, "Detector 11 arithmetics"
detid12,s,ql,"#0:0",,, "Detector 12 arithmetics"
```

```

detid13,s,ql,"#0:0",,, "Detector 13 arithmetics"
detid14,s,ql,"#0:0",,, "Detector 14 arithmetics"
detid15,s,ql,"#0:0",,, "Detector 15 arithmetics"
detid16,s,ql,"#0:0",,, "Detector 16 arithmetics"
detid17,s,ql,"#0:0",,, "Detector 17 arithmetics"
detid18,s,ql,"#0:0",,, "Detector 18 arithmetics"
#
# Standard parameters
#=====
clobber,b,h, yes,,, "Overwrite existing data structures ?"

```

Instead of specifying complete DOLs (Data Object Locations), which are composed of a filename plus the data structure extension (HDU), `spi_psd_calclib` accepts also simple filenames and adds the appropriate data structure extensions. This means that **specified data structure extensions are ignored**.

The parameters have the following meaning:

- `inlibDOLn` specifies the input library DOLs or filenames, where  $n$  runs from 0 to 4. A blank parameter name means that no DOL has been specified. At least one DOL or filename is needed for task execution.
- `outlibDOL` specifies the library DOL or filename to which the results will be written. The performed action sensitively depends on the `clobber` parameter. If `clobber=yes`, an existing template library that resides in the output file will be overwritten. If `clobber=no`, the selected templates will be added to those that reside already in the template library.
- `detidn` specifies the arithmetics that should be done to produce the result library templates for detector  $n$  (see below for more details about the arithmetics fields).
- `clobber` specifies if existing output data structures should be overwritten or not. If `yes` is specified, the executable will notify the user about the deletion of any file (see also `outlibDOL`).

The detector arithmetic fields specify the library templates that should be added from the source template files into the result library template file. There is one such field per detector channel, and each field specifies the templates that should be added for this detector. The arithmetic field is a character field with the general syntax

```
#n:a,b,c + #m:d,e,f,g,h + ...
```

Here,  $n$  and  $m$  are DOL source numbers, hence they run from 0 to 4. For example, `#0:` corresponds to the DOL specified by the parameter `inlibDOL0`, `#1:` corresponds to the DOL specified by the parameter `inlibDOL1`, and so on.

`a` to `h` are placeholders and stand for PSD detector channels (or detector identifiers) in the range 0 to 18. For example, `detid0,s,ql,"#0:0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18"` adds the templates for all PSD channels from the source DOL `inlibDOL0` to the result library for PSD channel 0. The same is achieved using `detid0,s,ql,"#0:0-18"`, hence the `-` operator allows the specification of detector ranges. The start and/or the end detector identifier may also be omitted. The four fields `#0:0-18`, `#0:-18`, `#0:0-`, `#0:-` are identical and specify all 19 detectors from the source DOL `inlibDOL0`.

Comma separated detector identifiers have the special functionality of toggling between detectors. For example, `#0:3,3` means that no detector is selected. The first 3 signals to use detector 3, while the second 3 deletes detector 3 from the list of detectors that should be used. The above example is, of course, not very meaningful. However, `#0:-,3` makes much more sense, since now all detectors except of detector 3 will be used (the `-` selects all detectors while the 3 deletes detector 3 from the list).

Templates from different source files may be added using the + symbol. For example, `detid0,s,q1,"#0:0+#1:8"` adds the templates for detector 0 from source file 0 and the templates for detector 8 from source file 8 to the result library for detector 0.

If the arithmetic field is left empty, nothing is done for the corresponding detector (thus one could use also `spi-psd_calclib` to extract templates for a given detector from a library).

**Blanks may be added deliberately to the arithmetic field.**

The following examples illustrate the usage of the detector arithmetic fields.

### 3.1 Copy a library

There is one source file (source file 0) which is copied into the result file. No template adding is performed.

```
inlibDOL0,s,q1,"libFM20.fits[SPI.-LIB.-PSD]",,,"PSD library DOL for filename #0"
inlibDOL1,s,q1,"",,,"PSD library DOL for filename #1"
inlibDOL2,s,q1,"",,,"PSD library DOL for filename #2"
inlibDOL3,s,q1,"",,,"PSD library DOL for filename #3"
inlibDOL4,s,q1,"",,,"PSD library DOL for filename #4"
#
outlibDOL,s,q1,"libFM.fits[SPI.-LIB.-PSD]",,,"PSD result library DOL or filename"
#
detid0, s,q1,"#0:0",,,"Detector 0 arithmetics"
detid1, s,q1,"#0:1",,,"Detector 1 arithmetics"
detid2, s,q1,"#0:2",,,"Detector 2 arithmetics"
detid3, s,q1,"#0:3",,,"Detector 3 arithmetics"
detid4, s,q1,"#0:4",,,"Detector 4 arithmetics"
detid5, s,q1,"#0:5",,,"Detector 5 arithmetics"
detid6, s,q1,"#0:6",,,"Detector 6 arithmetics"
detid7, s,q1,"#0:7",,,"Detector 7 arithmetics"
detid8, s,q1,"#0:8",,,"Detector 8 arithmetics"
detid9, s,q1,"#0:9",,,"Detector 9 arithmetics"
detid10,s,q1,"#0:10",,,"Detector 10 arithmetics"
detid11,s,q1,"#0:11",,,"Detector 11 arithmetics"
detid12,s,q1,"#0:12",,,"Detector 12 arithmetics"
detid13,s,q1,"#0:13",,,"Detector 13 arithmetics"
detid14,s,q1,"#0:14",,,"Detector 14 arithmetics"
detid15,s,q1,"#0:15",,,"Detector 15 arithmetics"
detid16,s,q1,"#0:16",,,"Detector 16 arithmetics"
detid17,s,q1,"#0:17",,,"Detector 17 arithmetics"
detid18,s,q1,"#0:18",,,"Detector 18 arithmetics"
```

### 3.2 Detector averaging

There is one source file (source file 0) from which the templates for all detectors are added. This results in an average template for all 19 detectors which is stored in the result file.

```
inlibDOL0,s,q1,"libFM20.fits[SPI.-LIB.-PSD]",,,"PSD library DOL for filename #0"
inlibDOL1,s,q1,"",,,"PSD library DOL for filename #1"
inlibDOL2,s,q1,"",,,"PSD library DOL for filename #2"
inlibDOL3,s,q1,"",,,"PSD library DOL for filename #3"
inlibDOL4,s,q1,"",,,"PSD library DOL for filename #4"
```

```

#
outlibDOL,s,ql,"libFM.fits[SPI.-LIB.-PSD]",,, "PSD result library DOL or filename"
#
detid0, s,ql,"#0:-",,, "Detector 0 arithmetics"
detid1, s,ql,"#0:-",,, "Detector 1 arithmetics"
detid2, s,ql,"#0:-",,, "Detector 2 arithmetics"
detid3, s,ql,"#0:-",,, "Detector 3 arithmetics"
detid4, s,ql,"#0:-",,, "Detector 4 arithmetics"
detid5, s,ql,"#0:-",,, "Detector 5 arithmetics"
detid6, s,ql,"#0:-",,, "Detector 6 arithmetics"
detid7, s,ql,"#0:-",,, "Detector 7 arithmetics"
detid8, s,ql,"#0:-",,, "Detector 8 arithmetics"
detid9, s,ql,"#0:-",,, "Detector 9 arithmetics"
detid10,s,ql,"#0:-",,, "Detector 10 arithmetics"
detid11,s,ql,"#0:-",,, "Detector 11 arithmetics"
detid12,s,ql,"#0:-",,, "Detector 12 arithmetics"
detid13,s,ql,"#0:-",,, "Detector 13 arithmetics"
detid14,s,ql,"#0:-",,, "Detector 14 arithmetics"
detid15,s,ql,"#0:-",,, "Detector 15 arithmetics"
detid16,s,ql,"#0:-",,, "Detector 16 arithmetics"
detid17,s,ql,"#0:-",,, "Detector 17 arithmetics"
detid18,s,ql,"#0:-",,, "Detector 18 arithmetics"

```

## 4 Interface definition

On input, `spi_psd_calclib` reads a list of library templates of HDU type `SPI.-LIB.-PSD`. No specific keyword is required in these data structures.

On output, `spi_psd_calclib` produces library templates for all 19 detectors, based on the input templates and combined following the arithmetic parameters. The 19 libraries are stored in a single file of HDU type `SPI.-LIB.-PSD` to which the following keywords are added:

```

CREATOR = 'spi_psd_calclib' / Program that created this FITS file
CONFIGUR= '1.1.0' / Software configuration
NRUNS = 1 / Number of runs in library
...
CREAT001= 'spi_psd_calclib' / Run 1 creating task
CONFIG01= '1.1.0' / Run 1 task configuration
SRCE0001= 'libFM20.fits[SPI.-LIB.-PSD]' / Source 0
DU000001= '11111111111111111111' / Source 0 detector usage for detector 0
DU010001= '11111111111111111111' / Source 0 detector usage for detector 1
...
DU180001= '11111111111111111111' / Source 0 detector usage for detector 18
NCRV0001= 723 / Number of curves in library for detector 0
NCRV0101= 723 / Number of curves in library for detector 1
...
NCRV1801= 723 / Number of curves in library for detector 18

```

The keyword `NRUNS` indicates how many runs have contributed to build the library templates that are in this file. If the output library file already existed, templates are added to the existing templates, hence the run history keeps track of the various sources of the templates. Following `NRUNS`, there are keywords which are added for each run (the run number `rn` is given by the last two digits of the keywords).

**CREATOR $rn$**  and **CONFIG $rn$**  specify the creating application and version number. If you used **spi\_psd\_calclib** to create the template, you'll find '**spi\_psd\_calclib**' and the actual version number after these keywords.

**SRCE $snrn$**  specifies the source DOL for source number  $sn$  and run number  $rn$ . These keywords reflect the DOLs you specified by the parameters **inlibDOL $rn$**  of the **spi\_psd\_calclib.par** parameter file.

**DU $idsnrn$**  reflects the detector selection that has been made for detector identifier  $id$ , source number  $sn$  and run number  $rn$ . These keywords are character strings of 19 digits in length, where 0 and 1 signifies that a detector was not selected or selected, respectively, for building the library template for detector  $id$  from source  $sn$ . Each of the 19 digits stands for an individual source detector, with detector identifier 0 as the left-most digit. These keywords are built by interpreting the arithmetic fields **detid $id$**  of the **spi\_psd\_calclib.par** parameter file.

**NCRV $idrn$**  summarises the total number of pulse shapes that are finally at the end of the run in all templates of each detector  $id$ . These keywords are identical to the keywords that are set by the task **spi\_psd\_buildlib**, and allow a global tracking of the library template content.

## 5 Error codes

The executable **spi\_psd\_calclib** may stop with the following error codes:

<b>SPI_PSD_CALCLIB_ERROR_MEM_ALLOC</b>	-230500
<b>SPI_PSD_CALCLIB_ERROR_NO_INDOL</b>	-230501
<b>SPI_PSD_CALCLIB_ERROR_NO_DELETE</b>	-230502

They have the following meaning:

- **SPI\_PSD\_CALCLIB\_ERROR\_MEM\_ALLOC** : the allocation of dynamical memory has failed. Probable your system resources are too limited to run this task.
- **SPI\_PSD\_CALCLIB\_ERROR\_NO\_INDOL** : no input DOL has been specified. **spi\_psd\_calclib** needs at least one input DOL to work properly.
- **SPI\_PSD\_CALCLIB\_ERROR\_NO\_DELETE** : although **clobber=yes** has been specified, **spi\_psd\_calclib** was unable to delete an existing library DOL using DAL routines. Probably, the library DOL is a child of another group. Therefore, delete the output library DOL by hand before executing **spi\_psd\_calclib**.

In addition, all errors that may occur in calls to ISDC support functions (such as for example DAL, RIL, PIL, **spi\_psdlib** or **spi\_toolslib**) are forwarded. Please consult the ISDC web pages and the corresponding User Manuals for getting information about these error codes.