;

TITLE: # The Zodiacal Light Emission Module in LevelS

DOC. TYPE: TECHNICAL NOTE

PROJECT REF.: PL-LFI-OAT-TN–045 **PAGE:** I of IV, 10

ISSUE/REV.: 1.0 **DATE:** 3 December 2007

Issued by	Michele Maris	Date: 03 December Signature:
Agreed by	A. ZACCHEI LFI DPC Manager	Date: Signature:
Approved by	R.C. BUTLER LFI Program Manager	Date: Xxx Signature: _____
Approved by	N. MANDOLESI LFI Principal Investigator	Date: Xxx Signature: _____

DISTRIBUTION LIST

Recipient	Company / Institute	E-mail address	Sent

CHANGE RECORD

Issue	Date	Sheet	Description of Change	Release
0	2007 Noc 20		Starting of the document	0
1	2006 Dec 03		First Issue	0
				0

1 SCOPE

This document describes the Zodiacal Light Emission module (ZLE) in the Level S.

Zodiacal Light Emission is the thermal emission from dust dispersed within the Solar System.

The author acknowledge for their support C.Burigana, M.Fraillis, S.Galeotta, M.Reinecke.

1.1 LIMITS OF APPLICABILITY

This document describes Issue 1.0 of the code, 3 Dec 2007.

2 APPLICABLE/REFERENCE DOCUMENTS

2.1 APPLICABLE DOCUMENTS

[AD-1] How to use the Zodiacal Light Emission maps for Planck
M. Maris, S. Fogliani, C. Burigana
PL-LFI-OAT-TN-031, 2004 Dec 31

2.2 REFERENCE DOCUMENTS

[RD–1] Zodiacal light emission in the PLANCK mission
M. Maris, C. Burigana and S. Fogliani
A&A, 2006, 452, 685

[RD–2] The COBE Diffuse Infrared Background Experiment Search for the Cosmic Infrared Background. II. Model of the interplanetary Dust Cloud
Kelsall, T.; Weiland, J. L.; Franz, B. A., et al.
ApJ, 1998, 508, 44

2.3 ACRONYMS LIST

CSV	Comma Separated Values
TOD	Time Ordered Data
ZLE	Zodiacal Light Emission

3 THE LEVELS/ZLE MODULE

`LevelS/ZLE` is a module devoted to the simulation of Zodiacal Light Emission (ZLE) within the framework of the Planck LevelS.

Details on the ZLE and how the calculation is accomplished can be found in [RD-1].

Here, It is enough to recall that the flux of ZLE, I_f, observed at a given frequency channel, f, from a given pointing direction, $\boldsymbol{P}$, is a function both of the instantaneous position of Planck within the Solar System, $\boldsymbol{R}_p$, as well as of the position of the Sun, $\boldsymbol{R}_{Sun}$.

The flux distribution as observer in the sky, $I_f(\boldsymbol{P}, \boldsymbol{R}_p, \boldsymbol{R}_{sun})$, is computed from the model of ZLE thermal emission described in [RD-2] and implemented into the code `ZOD_F90`. The code is able to predict the ZLE flux for any observer located within the Solar System and for any frequency. However, the calculation of the ZLE is time expensive and requires an appropriated fine tuning of many parameters, at the same time the range of variability for $\boldsymbol{P}$, $\boldsymbol{R}_p$ and $\boldsymbol{R}_{sun}$ is limited for a specific mission like Planck. Consequently an approximated method based on expansions in series of $I_f(\boldsymbol{P}, \boldsymbol{R}_p, \boldsymbol{R}_{sun})$ has been described in [RD-1] and implemented in the `IDL` package `ZLE_IDL` described in [AD-1].

`ZLE_IDL` could be used to generate directly a full TOD of ZLE samples, provided a list of $(\boldsymbol{P}, \boldsymbol{R}_p, \boldsymbol{R}_{sun})$ as a function of time is provided. However, for series of observations organized in scan circles more ore less normal to the ecliptic, the $I_f(\boldsymbol{P}, \boldsymbol{R}_p, \boldsymbol{R}_{sun})$ is a double peaked, periodic function, with peaks occurring near the ecliptic. Along the scan circle the ZLE varies over scales of tens of degrees. Scan circles taken in consecutive days, i.e. with small changes of $\boldsymbol{R}_p$, and $\boldsymbol{R}_{sun}$, will show a similar ZLE pattern. Hence, in simulating the ZLE, it is useless to generate TODs with a resolution of less then half degree. However, LevelS simulates the sampling of sky with angular resolutions of about 1 or 2 arcmin and roughly repeats many times the scan of the same region of sky within each pointing period. For this reason the ZLE in LevelS is generated in two steps.

In the first step, the output of the `LevelS/simmission` code is used by `LevelS/ZLE/zle_tod.pro` to generate a *`ZLE_Micromap`* file.

A `ZLE_Micromap` file contains, for any pointing period specified in the `simmission` output file, and a given detector in the focal plane, a circular strip of sky wider than the averaged scan circle drawn by the detector of choice in the given pointing period.

Typically a micromap is centerd on a circle drawn in the sky around the averaged spin axis of the given pointing period. The radius is equal to the angular distance between the line-of-sight of the

given detector and the spin axis, β_{dtc}. At least two other strips, representing sky circles with radii $\beta_{dtc} \pm 1^{\circ}$, are also computed. Each circle is sampled at a fraction of degree longitudinally.

It is needless to say that the `simmission` file so generated will be specific for a given mission profile (launch date, scanning strategy, Planck orbit and so on) and the resulting `ZLE_Micromap` files will be specific for each `simmission` file and each detector in the focal plane.

The second step is performed within the `LevelS/multimod` program by the module `zle.cc` with header `zle.h`. The `zle.h` module defines the class ZLE.

When the a ZLE object is instantiated the creator reads the file of `ZLE_Micromap` specified in the `multimod` par file and fill the object with its content.

Upon request from `LevelS/multimod`, for each simulated pointing direction, the method `ZLE::Addintensity` provides an interpolated ZLE flux. The correct micromap from the `ZLE_Micromap` file is selected according to the pointing period.

The `LevelS/ZLE` is contains the following packages:

`LevelS/ZLE/DB`	Data bases used by the package
`LevelS/ZLE/ZLE_IDL`	the series expansions package
`LevelS/ZLE/LIB_LS`	the interface to the LevelS of the IDL module
`LevelS/ZLE/CXX`	the C++ interface to the `LevelS/multimod`

The ZOD_F90 and the code to generate the data bases of ZLE emission to be stored in the `LevelS/ZLE/DB` are not in the present issue of the ZLE module due to the complexity in their use and integration, will be introduced in a next issue.

It has to be noted that `LevelS/ZLE/zle_tod.pro` uses a data base parameterizing the focal plane which is stored in `LevelS/ZLE/DB/FH` the same used by `LevelS/simmission` and `LevelS/multimod`. In case this database in the LevelS is changed the database in `LevelS/ZLE/DB/FH` will have to be updated accordingly.

4 INSTALLATION AND USAGE

An installation script, `install.sh`, is provided as described in the accompayning `README` files.

Installation of the IDL part requires no else than storing in `ZLE_IDL/zle_idl_start.pro` and `zle_start.pro` the proper absolute path to the libraries. This is automatically done by the installation program.

The `IDL` part requires `HEALPix/IDL` support and the support for `TOODI`.

To run the `IDL` part setup the global variable `IDL_DLM_PATH` to point to the location of `TOODI` libraries. An example is given in the script `source_start.sh` to be runned by

```
;source source.zle
```

edit this script by hand if your library is located in another directory.

Then simply enter IDL with HEALPix support (in most cases simply run `HIDL`) and start the zle_start.pro program which will link all the needed paths to the `IDL` environment.

The C++ part has instead to be copied inside the `LevelS/cxx` directory, a way is to use the `export_to_levels.sh` script within `LevelS/cxx`. This is not done by the installation program. Then has to be linked by recompiling multimod.

5 DESCRIPTION OF ZLE_MICROMAP

In defininining micromaps the following conventions are assumed:

1. The reference frame is ecliptical, centered on the Solar System Barycentre.
2. Units for angles are radians.
3. Units for distances are Astronomical Units.
4. Units for fluxes are either MJy/sr or K.

The content of a ZLE_Micromap file is described in the follow.

5.1 KEYWORDS

Name	Type	Description
Version	string	DDL version
Created	string	creation date
Creator	string	name and version of the program generating the DDL
Frequency	double	frequenc channell in GHz
FH_ID	int	Feed Horn identifier (usually the row number in the FH_DB)
FH_NAME	string	Feed Horn name
FH_DB	string	version of the Feed Horn database
SimFile	string	File name or handler for the original simmission file
SunDB	string	Solar positions DB
date_start	int	Start date (in yyyymmdd format)
time_start	double	Start time (in hhmmssdsss format)
date_end	int	int End date (in yyyymmdd format)
time_end	double	End time (in hhmmssdsss format)
Components	string	list of ZLE components
ZLE_METHOD	string	method or program used for the ZLE calculation
ZLE_DB	string	file name of zle coefficients (if ZLE_IDL is used)
NPNT	int	number of pointing periods
Ef	double	Ef coefficient
NLatMap	int	number of colatitudes in each micromap (minimum 3)

Name	Type	Description
lat_even_sampled	int	1 if colatitudes are sampled at a constant step
DTheta	double	Step in colatitude in the micromap (rad)
ThetaC	double	Colatitude for the central circle in the micromap (rad)
Theta0	double	Colatitude for the fist circle in the micromap (rad)
NLongMap	int	number of longtudes in each micromap
long_even_sampled	int	1 if colatitudes are sampled at a constant step
Dphi	double	step in longitude of the micromap (rad)
phi0	double	longitude of the first sample of the micromap (rad)
FluxUnit	string	unit for the ZLE flux in the micromap
Flux2Tant	double	conversion factor from Flux (MJy/sr) to Tant (mK)

5.2 COLUMNS

Name	Type	Description
I_SPIN	int	Number of pointing period in the simmission output file
t_startpt_int	int	time of start for the spin period in sec from 1970, Jan 1st, 00:00
t_startpt_frac	double	fractions of seconds for the time of start of the spin period
t_endpt_int	int	time of end for the spin period in sec from 1970, Jan 1st, 00:00
t_endpt_frac	double	fractions of seconds for the time of end of the spin period
SPIN_X	double	X component of the Spin axis for the pointing period
SPIN_Y	double	Y component of the Spin axis for the pointing period
SPIN_Z	double	Z component of the Spin axis for the pointing period
OBS_X	double	X baricentric position for the observer (A.U.)
OBS_Y	double	Y baricentric position for the observer (A.U.)
OBS_Z	double	Z baricentric position for the observer (A.U.)
SUN_X	double	X baricentric position for the Sun (A.U.)
SUN_Y	double	Y baricentric position for the Sun (A.U.)
SUN_Z	double	Z baricentric position for the Sun (A.U.)
IDX0	int	index of the first element of the micromap for the given pointing

Name	Type	Description
		period in the ZLE
ZLE	double	list of micromaps contents

Columns are filled according to the following rules:

1. All the columns, apart from ZLE, has a size of (NPNT+1).
2. All the micromaps are stored in ZLE.
3. A single micromap is assigned at each pointing period. The index of the first element of each micromap in ZLE is in IDX0.
4. All the Micromaps has the same size: NLatMap NLongMap.
5. ZLE has size (NPNT+1) NLatMap NLongMap
6. The Micromaps are arranged in ZLE as a list of rows, each row being filled with values for constant colatitude and increasing longitude. Colatitude increases for increasing rows. See the scheme below:

   θ = colatitude, λ = longitude, N_θ = number of colatitudes, N_λ = number of longitudes, Z = ZLE flux

end of previous micromap	
Address	**Content**
IDX0(period)	$[0,Z(\theta_0,\lambda_0)]$
IDX0(period)+1	$[1,Z(\theta_0,\lambda_1)]$
IDX0(period)+2	$[2,Z(\theta_0,\lambda_2)]$
...	...
IDX0(period)+NLongMap-1	$[N_\lambda\text{-}1,Z(\theta_0,\lambda_{N\lambda\text{-}1})]$
IDX0(period)+NLongMap	$[N_\lambda,Z(\theta_1,\lambda_0)]$
	...
IDX0(period)+2 NLongMap-1	$[2N_\lambda\text{-}1, Z(\theta_1,\lambda_{N\lambda\text{-}1})]$,
IDX0(period)+2 NLongMap	$[2N_\lambda,Z(\theta_2,\lambda_0)]$,
	...
IDX0(period)+3 NLongMap-1	$[3N_\lambda\text{-}1,Z(\theta_2,\lambda_{N\lambda\text{-}1})]$
begin of next micromap	

7. The first micromap is filled with the sampled colatitudes and longitudes.
   1. the first raw is filled with colatitudes of each row

2. the second row is filled with longitudes
3. all the unused elements are filled with IEEE NaN values.
4. correspondingly in all the other columns the first element of each column is meaningless and takes value NaN when columns are *double*.

6 VALIDATION

Validation is performed by tests in the `LevelS/ZLE/validation` directory.

Validation tests are performed by scripts operated under `bash`.

To operate a test script (under `bash`) call it with the command `source test_name.sh`.

All the files related to a test have names beginning with prefix `test0`, `test1`, ..., and so on, according to the test.

Needed data sets are in `LevelS/ZLE/validation/reference/simmission` and `LevelS/ZLE/validation/reference/DB`.

Expected outputs are in `LevelS/ZLE/validation/reference/outputs`.

Output files are `.log`, `.csv`, `.fits` and `.ps`.

6.1 TEST 0

Scope of this test is to verify that the ZLE_Micromaps are properly written by the zle_idl_ls.pro command and properly readed and mapped within the `multimod.cc` code.

The script for the test is `test0.sh`

When displayed the ZLE_Micromap should contain a consecutive list of numbers ordered in rows like:

0,1,2,3, ..., 9
1000,1001, ..., 1009
2001,2002, ..., 2009
10000,10001, ..., 10009
11000,11001, ..., 11009
12000,12001, ..., 12009

and so on.

Note that this test is slow given every intermediate step is written on disk on a `csv` file.

6.2 TEST 1

This test is designed to check proper pointing reconstruction and interpolation. In place of the signal a simple analytical model is used.

The script for the test is `test1.sh`.

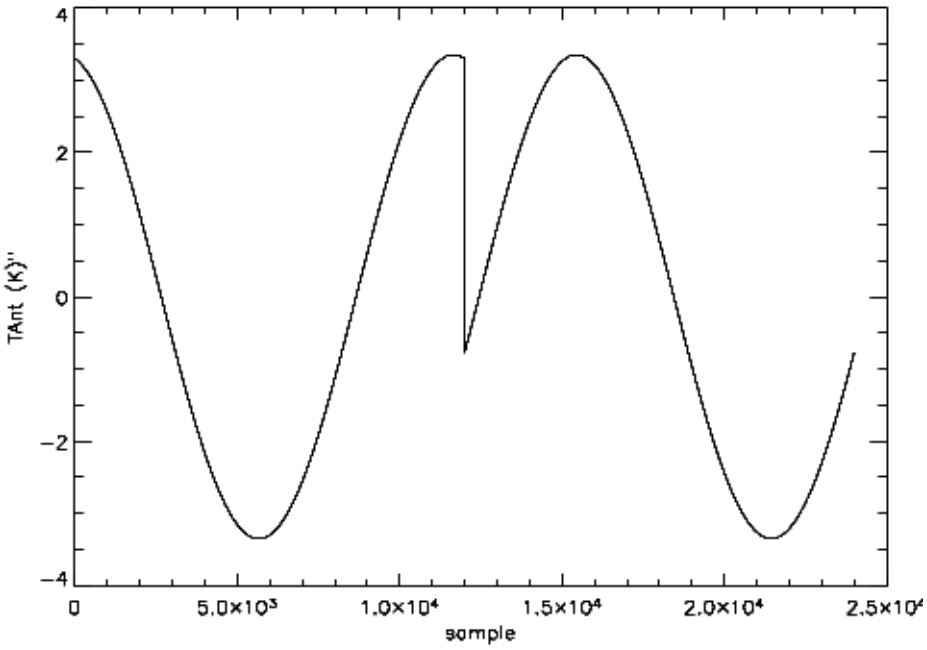
The output is the error in reconstructing pointing and interpolating the function which would be below some 1e-5.

Typical good outputs are:

Maximum error in colatitude: 4.7683716e-06
Maximum error in longitude : 5.7316105e-06
Maximum error in flux : 2.83718e-05

Note that this test is slow given every intermediate step is written on disk on a `csv` file.

An example of signal output is in: test1.Tant.ps



6.3 TEST 2

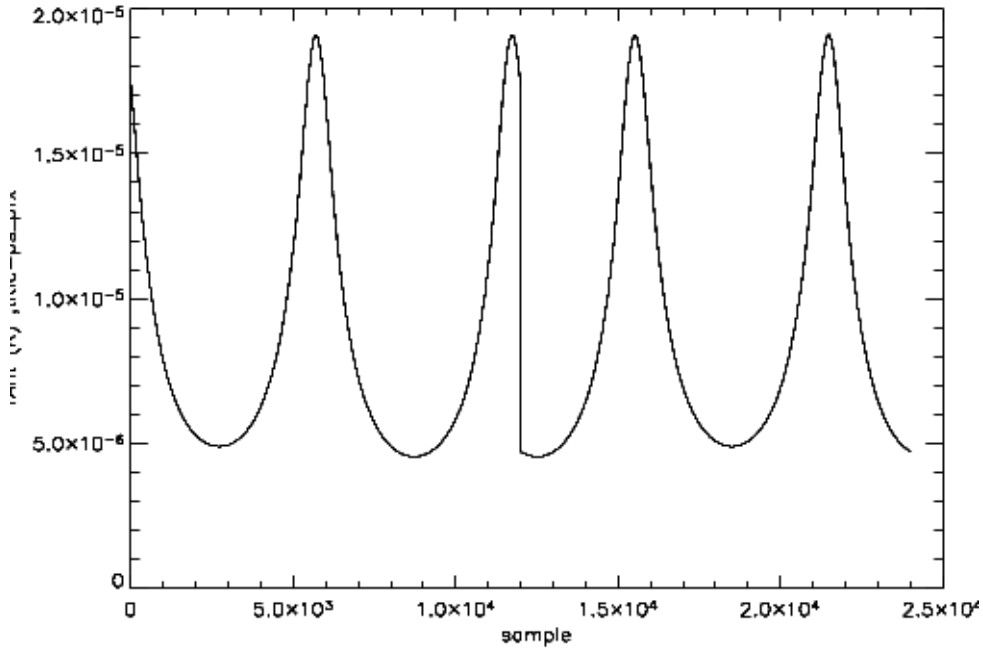
Finally a true ZLE generation.

Typical outputs are:

Min ZLE : 4.51558e-06 K
Mean ZLE: 8.66770e-06 K
Max ZLE : 1.91318e-05 K

An example of output is: test2.Tant.ps

Note that this test is slow given every intermediate step is written on disk on a `csv` file.



6.4 TEST 3

Example of massive production of ZLE over 1000 spin periods.

Typical outputs are:

Min ZLE : 4.42067e-06 K
Mean ZLE: 8.66427e-06 K
Max ZLE : 1.91318e-05 K

An example of output is: test3.Tant.ps.