"How to improve your SAX life"

The SAX LECS Data Analysis System

— Software User Manual —

April 2, 1996

Written by	:	U. Lammers
Revised by	:	F. Favata, A. Parmar

Document ID	:	SAX/LEDA/0010
Issue no.	:	0.3.0
Status	:	draft

Revision history

Revision no.	Date	Author	Comments
0.0.1	09/10/95	UL	skeleton
0.1.0	08/01/96	UL	first draft version
0.3.0	02/04/96	UL	GVs comments, updated man-pages + FITS headers

Contents

1 Introduction

1.1 The LECS on-board SAX

SAX is an Italian-Dutch X-ray astronomy satellite that will be launched into a low-earth orbit at an altitude of 600 km with an inclination of 3° in spring 1996. The payload consists of 6 scientific instruments with a total spectral coverage from 0.1 keV to 300 keV. The low-energy response is realized by means of an imaging gas-scintilation proportional counter (LEGSPC) provided by ESA's Space Science Department. In contrast to conventional GSPC's the LEGSPC possesses an ultra-thin gas cell entrance window and does dispense with separate drift and scintillation regions. Inside the satellite the instument is located at the focal plane of one out of four X-ray concentrator systems. The LEGSPC together with its concentrator is regarded as one unit named Low-Energy Concentrator Spectrometer (LECS). More details about the mission and the LECS can be found in [1, 2] and references therein.

The raw telemetry data from pointed observations of the LECS experiment are available in the form of so-called Final Observation Tapes (FOTs) from the SAX Scientific Data Centre in Rome, Italy. Prior to any actual scientific analysis with popular X-ray software packages such as *Xspec*, *Ximage*, and *Xronos* the FOT data have to be pre-processed using a system of dedicated instrument-specific programs. This software is referred to as the *SAX LECS Data Analysis System* (SAXLEDAS).

1.2 Scope of this document

This document is intended to provide all the necessary information to run the SAXLEDAS, i.e., to create from raw LEGSPC FOT data files that can be further processed with mission-independent, high-level X-ray analysis packages such as *Xspec*. It is assumed that the SAXLEDAS has been properly installed on a UNIX platform and is ready to use. General informations on how to get and install the SAXLEDAS can be found at the URL
`http://astro.estec.esa.nl/SA-general/Research/Sax/Saxint/saxmain.html`.

1.3 Intended readership

This document shall be read by everyone who intends to pre-process raw telemetry FOT data of the LEGSPC instrument with the SAXLEDAS. The reader should have a solid understanding of X-ray data analysis in general and is expected to have basic knowledges in the following fields:

- the design and functioning principle of the LEGSPC instrument (see [2] and references therein).
- the concept of FITS files (comprehensive information can be found at the URL `http://www.gsfc.nasa.gov/astro/fits/fits_home.html`)
- the FTOOLS software package to work with FITS files (see the URL `http://heasarc.gsfc.nasa.gov/docs/software/ftools/ftools_menu.html`).
- working in a UNIX environment (creating/modifying/deleting files and directories, running programs, etc.) ([3] and many others)

1.4 Contact Persons

All questions, comments or suggestions concerning technical issues on any aspect of the SAXLEDAS or this document should be addressed to

Uwe Lammers Uwe.Lammers@astro.estec.esa.nl

In case of questions related to scientific issues one of the following persons should be contacted:

Arvind Parmar Arvind.Parmar@astro.estec.esa.nl
Fabio Favata Fabio.Favata@astro.estec.esa.nl

2 Data Files in the SAXLEDAS

Regarding the format of the files that are read and written by the individual SAXLEDAS modules it has been established to utilize *exclusively* FITS, i.e., there is no other ad-hoc self-defined file format whatsoever. This decision was made, taking account of

- the wide acceptance of the FITS file format [4, 5] in the astronomical community,
- the availability of numerous S/W tools that operate on FITS files (especially the FTOOLS [6] package from NASA/HEASARC), and
- the self-descriptive nature of FITS files.

Although FITS provides a rough framework for storing data in a portable and machine-independent manner, the structural details of the data arrangements have still to be individually defined. In the following this is first done for the so-called raw event and HK files which are created by the *ingestor* module from either Final Observation Tapes (FOTs), or packet files stemming from the instrument's Electrical Ground Support Equipment (EGSE). Several of the SAXLEDAS modules process the raw files and extend and/or modify their structure, i.e., output *derived* event/HK files that slightly differ from the raw versions. These differences will be described here as well.

The SAXLEDAS FITS files are compliant with the FITS standards as defined by the NASA Office of Standards and Technology [4]. Furthermore, all available standards and recommendations issued by the NASA/OFWG (e.g. [7]) have been carefully considered in order to be compatible with existing and forthcoming S/W tools from NASA/OGIP. This mainly pertains to the choice of FITS header keywords.

2.1 Basics

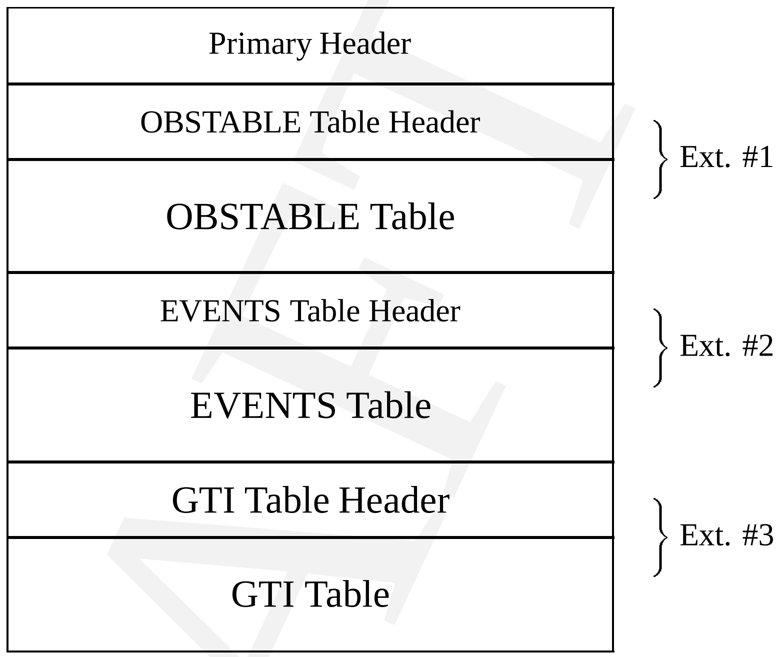
2.1.1 Event files: general contents and structure

The LEGSPC instrument can be operated in three different modes that are *Direct Mode* (DM), *Image Mode*, and *Spectral Mode*. In DM — the only one which is of relevance here — the characteristic data of each individually detected photon are recorded by the EU and, later on, stored into corresponding DM telemetry packet files. The available physical photon data are:

- arrival time with a resolution of $\approx 15.25\,\mu$s
- Pulse Height Amplitude (PHA) in units of detector channels (0–1023)
- X-coordinate in units of detector pixels (0-255)
- Y-coordinate in units of detector pixels (0-255)
- burst length (BL) signal in units of detector channels (0–255)
- VETO signal in units of detector channels (0-255)

Roughly spoken, the PHA is a measure for the photon energy, the BL value determines the penetration depth of the photon in the gas cell, and the VETO signal is a measure for the light distribution seen by the PMT. More precise definitions of those quantities can be found in [8].

A FITS event file is now a time-ordered list of those data from all photons that have been detected during a certain exposure time. The data are arranged in the form of a FITS Binary Table (FBT), also simply called Bintable, where each row has a fixed column structure and represents a valid photon event. More precisely, an event file in the SAXLEDAS consists of a primary header with an empty primary array followed by a so-called Obstable extension. This extension is likewise a FBT whose structure and semantics are discussed in Sec. 2.2.2. The actual event table forms the second extension of the FITS file. Its detailed structure is presented in Secs. 2.2.3 and 2.4.1 for the *raw* and *linearized* forms respectively. Raw event list files finally possess a Good Time Interval (GTI) table which constitutes the third extension of each corresponding FITS files (Sec. 2.2.4). The following figure gives a schematic depiction of a SAXLEDAS event file:

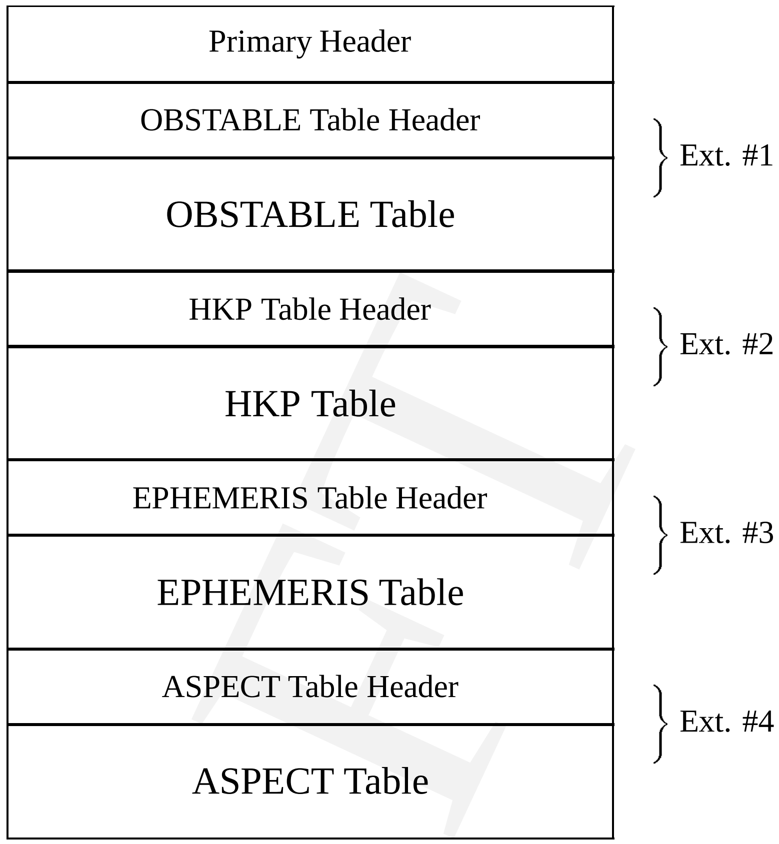


2.1.2 HK files: general contents and structure

The SAXLEDAS HK files contain all data that are not of actual scientific interest but which provide auxiliary information needed by various modules that operate on event lists. The various categories of HK parameters with relevance for the scientific data analysis are

- time and duration of observation intervals ($\longrightarrow$ OBSTABLE)
- instrumental HK parameter ($\longrightarrow$ HKP)
- spacecraft ephemeris data ($\longrightarrow$ EPHEMERIS)
- spacecraft attitude data ($\longrightarrow$ ASPECT)

Just as in the case of event data, the most natural choice for representing the various HK data sets is the FBT form. Thus, a SAXLEDAS HK file is composed of a primary header with a corresponding empty primary array, followed by a sequence of four FBT extensions. The names of those extensions are given in parenthesis in the enumeration above or can be inferred from the following schematic figure:



2.1.3 File naming convention

The name of a SAXLEDAS file consists of three parts, the basename, the type, and the format extension. The basename can be any valid Unix file name, the type extension is a '_' followed by a unique three-letter identification, and the format extension is *always* '*.fits*'. Example:

'kv183_kv551_rev.fits'

is the full name of a FITS file *'kv183_kv551'* of type *'_rev'*.

2.1.4 Types of event and HK files

The so-called *ingestor* is the most basic module of the SAXLEDAS. It can be viewed as a program to create "literal copies" of the data on a FOT or in the EGSE packet files into FITS format. These original FITS files represent the basic input to the analysis chain and are *raw* event and HK files whose structure follow the scheme above. In the course of processing the raw files it proves unavoidable to modify and/or extend their structure which leads to various *types* of event and HK files. The following table provides an overview of the various types of FITS files in the SAXLEDAS whose contents and structures are detailed in this document.

type ext.	name of 2. Bintable ext.	Description	input to module(s)	output from module(s)
_rev	`RAW EVENTS`	Raw EVents file	*evelin*, *calcgain*	*foting*, *egseing*
_lev	`EVENTS`	Linearized EVents file	*evelin*, *baryconv*	*evelin*, *baryconv*
_rhk	`RAW HKP`	Raw HouseKeeping file	*eng2phys*, *hkplot*	*foting*, *egseing*
_hkp	`HKP`	HouseKeeping file with Physical values	*calcgain*, *hkplot*	*eng2phys*
_hkg	`HKP+GAIN`	HouseKeeping file with Gain related columns	*evelin*, *hkplot*	*calcgain*
_xhk	`PANTER HK`	auXiliary HouseKeeping file	*hkplot*	*pantering*

2.2 Raw event file

2.2.1 Primary Header

The primary header is a standard FITS file header that comprises three types of keyword-values pairs. In the first instance, it contains all keywords that are mandatory for a primary header such as `BITPIX` and `NAXIS` [4]. Furthermore, values are defined for several keywords that are designated as *reserved* by the FITS standard, e.g., `INSTRUMEN` and `TELESCOP`. Finally, the third category of keywords provides additional information about the data stored in the FBT extensions following the primary HDU. An example is `N_OBIS` which holds the number of observation intervals (OIs) that have been concatenated in this event file (see Sec. 2.2.2).
The header comprises various timing related keywords which are repeated in each table header. This block of keywords defines the observation boundaries and the exact reference frame for all time-tagged data stored in the FBT. The keywords are:

keyname	description
TIMESYS	definition of mission reference time = zero point for all time tags; nominal value is '1994 1 1 00:00:00'
TIMEUNIT	the time unit for the keywords TSTART, TSTOP, TELAPSE; nominal value is 's'
TIMEREF	defines the general time reference frame — is either 'LOCAL' or 'SOLARSYSTEM' if times refer to satellite position or the barycenter of the solar system respectively; the nominal value is 'LOCAL'
TASSIGN	defines the location where the time assignment was made — will always be 'SATELLITE'
CLOCKAPP	boolean value specifying whether spacecraft clock drifts have been taken into account — will always be T
MJDREF	the Modified Julian Day (MJD) number corresponding to mission reference time defined in TIMESYS; value is 49353.0
MJD-OBS	the MJD number corresponding to the observation start
TSTART	observation start time in s relative to MJDREF
TSTOP	observation end time in s relative to MJDREF
TELAPSE	the difference TSTOP − TSTART, i.e., the total time in seconds elapsed during the whole observation; any gaps due earth occultation, SAA passages, etc. are included
SCSEQBEG	the hexadecimal value of the S/C clock [8] at the observation start in the format *XXXX.YYYY,* XXXX being the integer s-part
SCSEQEND	the hexadecimal value of the S/C clock [8] at the observation end in the same format as SCSEQBEG; note that due to wrap-arounds SCSEQEND can be lower than SCSEQBEG
DATE-OBS	UTC date of observation start
TIME-OBS	UTC time of observation start
DATE-END	UTC date of observation end
TIME-END	UTC time of observation end

The value of the CONTENT keyword in the primary header of event list files is 'EVENT LIST'. A sample primary header is given in Sec. A.1. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

The SAXLEDAS event files do not contain any primary array data, hence, the NAXIS keyword has always the value 0.

2.2.2 Observation interval table

The observation interval table (Obstable) forms the first extension in a SAXLEDAS event file and consists of a Bintable header followed by the corresponding FBT. The data stored in an Obstable have the following meaning: Ideally, the observation of a single X-ray target during a certain observation period (OP) results into merely one, possibly quite large DM file on the FOT. However, if the observation conditions alter during the OP, for instance, due to changes in the experiment configuration (SAA passages, earth eclipse, S/C roll manoeuvres, etc.) the OP is split into corresponding observation intervals (OBIs) each of which is marked by a time boundary. In many cases it is desirable to concatenate all OBIs of the actual OP and build up a single large event list from all those data. If this has been done, the Obstable contains the original time boundaries of the individual OBIs. The table has 7 columns with a total row width of 50 bytes.

col. #	col. name	form[1]	unit	description
1	OBI_NUM	I	—	observation interval number
2	OBIB	9A	s	S/C clock at beginning of OBI
3	OBIE	9A	s	S/C clock at end of OBI
4	START_DATE	8A	—	UTC date of OBI start
5	START_TIME	12A	—	UTC time of OBI start
6	STOP_DATE	8A	—	UTC date of OBI end
7	STOP_TIME	12A	—	UTC time of OBI end
		60 byt.		

In case of the *egseing* ingestor, where non-flight EGSE data are converted to FITS, the Obstable consists only of a single row. It represents the arrival times of the first and last photon in the EGSE DM packet files respectively.
A sample Obstable header is given in Sec. A.2. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.2.3 Event list table

The event list table (ELT) forms the second extension of a SAXLEDAS event file and consists of a Bintable header followed by the corresponding FBT. As already discussed in Sec. 2.1.1 the ELT contains the actual scientific data where each detected photon is represented by a separate row in the FBT. The table has 6 columns with a resulting row width of 18 bytes:

col. #	col. name	type	unit	description
1	TIME	D	s	the photon arrival time in seconds since MJDREF
2	PHA	I	chan	the Pulse Height Amplitude, i.e., energy channel of the photon
3	RAWX	I	pixel	the unlinearized X-coordinate of the photon in the focal plane
4	RAWY	I	pixel	the unlinearized Y-coordinate of the photon in the focal plane
5	BL	I	chan	the BL value of the photon in detector channels
6	VETO	I	chan	the VETO value of the photon in detector channels
		18 byt.		

In addition to the block of timing keywords discussed in Sec. 2.2.1, the ELT header contains two new time-related keywords, viz. ONTIME and TIMEDEL. The value of the former specifies the total time in seconds the S/C was actually pointing 'on-source'. The value is identical to the sum of all GTIs defined in the FBT following the ELT (see Sec. 2.2.4). Corrections for instrumental dead time effects are not included in ONTIME. TIMEDEL is set to the instrumental time resolution $15.25\,\mu$s.
A sample ELT header is given in Sec. A.3. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

[1] I: 2-byte integer, J: 4-byte integer, E: 4-byte real, D: 8-byte real, L: logical, xA: x-byte character

2.2.4 Good Time Interval table

The ELT is followed by a GTI table which forms the third FBT extension. This table consists of one or multiple time intervals defined by a start and stop time respectively. Each photons whose arrival time falls within one of those intervals is considered to be a 'scientifically valid' X-ray event. The sum of all GTI intervals is equals the value of the `ONTIME` keyword in the ELT header. The structure of the GTI table is:

col. #	col. name	type	unit	description
1	`START`	D	s	lower GTI boundary
2	`STOP`	D	s	upper GTI boundary
		16 byt.		

A sample GTI header is given in Sec. A.5. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.3 Raw Housekeeping parameter file

2.3.1 Primary Header

The primary header is identical to the one in the event list file (see Sec. 2.2.1) except for the value of the `CONTENT` keyword which is here `'HK PARAMETER'`.

2.3.2 Observation interval table

The observation interval table (header+data) is identical with the one in the event list file (see Sec. 2.2.2).

2.3.3 HK parameter table

The HK parameter table is the second extension in a SAXLEDAS HK file. It contains the values of various instrument specific quantities in the form of a time ordered list, i.e., a FBT with each row representing a set of instrument HK parameters at a certain time. In total there are 55 parameters many of which are *physical* quantities such as temperatures and voltages. Note that, nevertheless, in the raw HK file all parameters are stored in unscaled digital units as they are read out by the EU. The conversion to physical units is done at a later stage with a specialized SAXLEDAS module *eng2phys* which outputs a derived HK file (see Sec. 2.5). The following table provides a quick overview of the HK parameter table structure. The description of each parameter is deliberately kept brief — for more details on the meaning of the parameters, see [8].

col. #	colum name	type	description
1	TIME	D	time tag in seconds since MJDREF
2	GETT_ENBL_SW	L	getter enable software flag (T if enabled)
3	MEM_MOD_ENBL	L	memory modify software flag (T if enabled)
4	GETT_ENBL	L	getter enable status taken before relay RS04 (T if enabled)
5	SHUT_CLOSE	L	window protection shutter closed position status (T if shutter closed)
6	SHUT_OPEN	L	window protection shutter open position status (T if shutter open)
7	PMT_CURR	L	PMT current flag (T if excessive current on PMT)
8	GU_SAFETY	L	gas cell pressure safety flag (T if pressure exceeded limit)
9	VALVE_SUPPLY	L	GU valve supply after RS05 (T if valve supply enabled)
10	GETT_SUPPLY	L	getter supply status after relay RS04 (T if getter supply enabled)
11	HV_POWER	L	HV power supply status after relay RS03 (T if HV is enabled)
12	HV_ENBL	L	HV enable status after relay RS03 (T if 28V supply is present)
13	FEE_−15V	L	FEE -15V power status after relay RS01 (T if enabled)
14	FEE_+15V	L	FEE +15V power status after relay RS01 (T if enabled)
15	ST_EQ	I	self-test error counter; if set to 15, more than 15 errors occured since last readout
16	S_EQ	I	system error counter; if set to 15, more than 15 errors occured since last readout
17	O_EQ	I	operational error counter; if set to 15, more than 15 error occured since last readout
18	QEP_RATEMETER	I	qualified event pulse counter
19	EVENT_READ_RATEMETER	I	event input buffer counter
20	EVENT_LOST_RATEMETER	I	counter for events that have not been transfered to DE
21	BASETIME	J	basetime of the S/C clock for the actual HK parameter set
22	DEADTIME_RATEMETER	I	count of the dead time accumulated in the AE
23	E_MIN_RATEMETER	I	count of the output of the LLD
24	E_MAX_RATEMETER	I	count of the output of the Ra discriminator

25	POS_RATEMETER	I	count of the output of the Rx and Ry discriminators
26	BL_RATEMETER	I	count of the output of the BL discriminator
27	VETO_RATEMETER	I	count of the output of the Rv discriminator
28	CELL_PRESSURE	I	current pressure in the gas cell
29	RESERVOIR_PRESSURE	I	current pressure in the GU
30	GAS_VALVE1_OPERATION	I	gas valve 1 operation cycle counter
31	GAS_VALVE2_OPERATION	I	gas valve 2 operation cycle counter
32	STANDBY_MODE	L	T if LEGSPC is in STANDBY mode
33	RUNNING_MODE	L	T if LEGSPC is in RUNNING mode
34	NORM_INTERR_HANDLER	L	T if DE normal input interrupt handler is active
35	HT_INTERR_HANDLER	L	T if DE high-throughput interrupt handler is active
36	DIRECT_MODE_PROC	L	T if DE DM event processing task is active
37	SPEC1_MODE_PROC	L	T if DE spectrum mode processing task is active
38	IMAGE1_MODE_PROC	L	T if DE image mode processing task is active
39	ENG_HK_GENERATION	L	T if DE generates engineering HK packets
40	HK_COLLECTION	L	T if DE HK collection task is active
41	MACROCOM_PROC	L	T if DE macroprocessing task is active
42	PROT_TASK	L	T if DE protection task is active
43	RTMS	L	T if real-time monitoring system is active
44	CELL_TEMPERATURE	I	current temperature in the gas cell
45	RESERV_TEMPERATURE	I	current temperature in the GU
46	EU_TEMPERATURE	I	current temperature of the EU
47	CELL_HIGH_VOLTAGE	I	cell HV monitor
48	PMT_HIGH_VOLTAGE	I	PMT HV monitor
49	LOW_VOLTAGE_+5V	I	+5V DC/DC converter voltage monitor
50	LOW_VOLTAGE_+6V	I	+6V DC/DC converter voltage monitor
51	LOW_VOLTAGE_+15V	I	+15V DC/DC converter voltage monitor
52	LOW_VOLTAGE_-15V	I	-15V DC/DC converter voltage monitor
53	LOW_VOLTAGE_+28V	I	+28V DC/DC converter voltage monitor
54	COMMAND_COUNTER	I	DE processed macrocommand counter
55	PMT_CURR_FLAG_COUNTER	I	PMT current flag counter
		93 byt.	

A sample raw HK parameter header is given in Sec. A.6. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.3.4 Ephemeris table

The ephemeris table (ET) forms the third extension of a SAXLEDAS HK file. Basically it contains the S/C's essential orbit data as a function of time, i.e., the position and velocity of the satellite sampled at a rate of 1/s. The data are arranged in a 17-column FBT with a total width of 86 bytes:

col. #	col. name	type	unit	description
1	TIME	D	s	time tag relative to MJDREF
2	SAT_X	D	km	X-component of the S/C position vector
3	SAT_Y	D	km	Y-component of the S/C position vector
4	SAT_Z	D	km	Z-component of the S/C position vector
5	VEL_X	D	km/s	X-component of the S/C velocity vector
6	VEL_Y	D	km/s	Y-component of the S/C velocity vector
7	VEL_Z	D	km/s	Z-component of the S/C velocity vector
8	LEAP_SEC	I	s	difference between TAI and UTC in s
9	SAA_FLAG	L	—	flag indicating whether S/C is within SAA
10	SAA_TIME	E	min	depending on SAA_FLAG = T/F: minutes elapsed since S/C entered/exited SAA
11	DAYSIDE_FLAG	L	—	flag indicating whether S/C is on earth dayside or not
12	DAYSIDE_TIME	E	min	depending on DAYSIDE_FLAG = T/F: minutes elapsed since S/C entered/exited from earth dayside
13	FOV_FLAG	I	—	instrument pointing direction: 0 = sky, 1 = bright earth, 2 = night earth
14	EARTH_ANG	E	deg	minimum angle between instrument axis and limb of the earth
15	BREARTH_ANG	E	deg	minimum angle between instrument axis and the day/night terminator on earth
16	SUN_ANG	E	deg	angle between instrument axis and the direction to the sun
17	COR_MIN	E	GeV/c	minimum geomagnetic cut-off rigidity at the S/C's location
		86 byt.		

In the case of ground data stemming from the EGSE the data section of the ET extension is empty, i.e., the number of rows in the FBT is 0.
A sample ET header is given in Sec. A.9. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.3.5 Aspect table

The aspect table (AT) forms the fourth extension of a SAXLEDAS HK file. It comprises the essential data concerning the S/C's attitude, i.e., pointing direction of the NFIs as a function of time (granularity: 0.5 s). As in the case of the ET the data are arranged as a FBT. Each row consists of 8 columns with a resulting width of 64 bytes:

col. #	col. name	type	unit	description
1	`TIME`	D	s	time tag relative to `MJDREF`
2	`RA_SCX`	D	deg	right ascension of the ACA X axis
3	`DEC_SCX`	D	deg	declination of the ACA X axis
4	`RA_SCY`	D	deg	right ascension of the ACA Y axis
5	`DEC_SCY`	D	deg	declination of the ACA Y axis
6	`RA_SCZ`	D	deg	right ascension of the ACA Z axis
7	`DEC_SCZ`	D	deg	declination of the ACA Z axis
8	`ROLL_ANG`	D	deg	instrumental roll angle
		64 byt.		

In the case of ground data stemming from the EGSE the data section of the AT extension is empty, i.e., the number of rows in the FBT is 0.

A sample AT header is given in Sec. A.10. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.4 Derived event list files

2.4.1 Linearized event list file

Linearized event list files (type ext.: "*_lev*") are created by the SAXLEDAS module *evelin* and have the same overall structure as raw event files, i.e., consist of a primary header/empty primary array followed by the Obstable and the actual event extension (see Sec. 2.1.1). The latter differs from the raw version in the following manner:

1. Five FBT columns have been added:

col. #	col. name	type	unit	description
1–6				same as for raw event extension
7	`PI`	I	chan	the Pulse Invariant of the photon derived from the PHA value by applying temporal and spatial gain corrections (see [9] for more information)
8	`PIC`	I	chan	the PI value from column 7 additionally corrected for the effect of gas-cell penetration (see [2] and [9] for more information
9	`DETX`	E	pixel	linearized X coordinate of the photon in the focal plane (see [9] for more information)
10	`DETY`	E	pixel	linearized Y coordinate of the photon in the focal plane (see [9] for more information)
11	`X`	I	pixel	projected X-position of photon on sky
12	`Y`	I	pixel	projected Y-position of photon on sky
		34 byt.		

2. The standard block of timing keywords (Sec. 2.2.1) plus `ONTIME` (Sec. 2.2.3) is completed by three time related keywords:

   `LIVETIME` The total time in seconds pointed 'on-source' corrected for instrumental dead-time effects.

   `EXPOSURE` The total time in seconds on source corrected for any relevant quantities to compute the correct count rate.

   `DEADC` The dead time correction factor in the range $0.0 - 1.0$ which equals the ratio `LIVETIME`/`ONTIME`.

3. Standard World-Coordinate-System (WCS) [10, 11] header keywords have been added. The values of those keywords enable corresponding FITS readers such as *Ximage* or the standard FTOOLS task *xy2sky* to compute from the X and Y sky image coordinates the R.A. and Decl. values of each photon in the ELT.

On top of that various header keywords and history records have been added to the ELT header. The values of those keywords mainly provide details about the possessing steps that have been performed by the *evelin* task [12].
A sample linearized event-file header is given in Sec. A.4. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.5 Derived HK parameter files

2.5.1 Physical HK parameter file

Physical HK parameter files (type ext.: "*_hkp*") are generated by the SAXLEDAS module *eng2phys* and have the same overall structure as raw HK files, i.e., consist of a primary header/empty primary array, followed by the Obstable extension, the actual HK-, the ephemeris, and the attitude extension (see Sec. 2.1.2). The essential difference to the raw version is that several parameters in the HK parameter extension have been scaled from digital to physical units using calibration relations provided in [8]. As a consequence, the type of certain table columns has changed:

col. #	colum name	type	unit
18	QEP_RATEMETER	E	count/s
19	EVENT_READ_RATEMETER	E	count/s
20	EVENT_LOST_RATEMETER	E	count/s
22	DEADTIME_RATEMETER	E	count/s
23	E_MIN_RATEMETER	E	count/s
24	E_MAX_RATEMETER	E	count/s
25	POS_RATEMETER	E	count/s
26	BL_RATEMETER	E	count/s
27	VETO_RATEMETER	E	count/s
28	CELL_PRESSURE	E	mbar
29	RESERVOIR_PRESSURE	E	mbar
30	GAS_VALVE1_OPERATION	E	count/s
31	GAS_VALVE2_OPERATION	E	count/s
44	CELL_TEMPERATURE	E	Celsius
45	RESERV_TEMPERATURE	E	Celsius
46	EU_TEMPERATURE	E	Celsius
47	CELL_HIGH_VOLTAGE	E	V
48	PMT_HIGH_VOLTAGE	E	V
49	LOW_VOLTAGE_+5V	E	V
50	LOW_VOLTAGE_+6V	E	V
51	LOW_VOLTAGE_+15V	E	V
52	LOW_VOLTAGE_−15V	E	V
53	LOW_VOLTAGE_+28V	E	V

A sample physical HK header is given in Sec. A.7. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

2.5.2 HK parameter gain file

The HK parameter gain (HKG) file (type ext.: "*_hkg*") is derived from the physical HK file (see last section) i.e., it has the same overall structure and the first 55 columns of the HK parameter extension are identical in the two cases. In a HKG file, however, the FBT possesses additional columns which are related to the computation of the temporal energy gain performed with the SAXLEDAS module *calcgain* [9]. In brief, the following information is provided:

col. #	col. name	type	unit	description
1–55				same as for HKP extension
56	PHA1	E	chan	centroid of fitted Gaussian for calibration source 1
57	FWHM1	E	chan	FWHM of fitted Gaussian for calibration source 1
58	CHISQ1	E	—	reduced χ^2 value of the Gaussian fit for calibration source 1
59	GAIN1	E	—	energy gain of calibration source 1
60	PHA2	E	chan	centroid of fitted Gaussian for calibration source 2
61	FWHM2	E	chan	FWHM of fitted Gaussian for calibration source 2
62	CHISQ2	E	—	reduced χ^2 value of the Gaussian fit for calibration source 2
63	GAIN2	E	—	energy gain of calibration source 2
64	GAIN12	E	—	ratio GAIN1/GAIN2
65	GAIN_AVG	E	—	average gain of the two calibration sources
66	CHISQAVG	E	—	reduced χ^2 value of GAIN_AVG
67	BCKGND1	E	count/s/cm^2	normalized background countrate in spatial window 1
68	BCKGND2	E	count/s/cm^2	normalized background countrate in spatial window 2
		191 byt.		

For more information on the additional table columns see [9].
A sample HKG file header is given in Sec. A.8. Each keyword/value has a corresponding comment, thus, the header is self-explanatory.

3 Pipeline Processing with the SAXLEDAS

3.1 General philosophy

The SAXLEDAS has been designed as an FTOOLS based, command-driven system of standalone modules, each of which is optimized for a special purpose. The following figure illustrates the basic data flow in the SAXLEDAS. Rectangular boxes represent individual modules while boxes with rounded corners stand for one or several input/output files. A thick arrows pointing towards a file box indicates that this file has been newly created or modified in any way. Thin arrows indicate that files are just being passed without any changes.

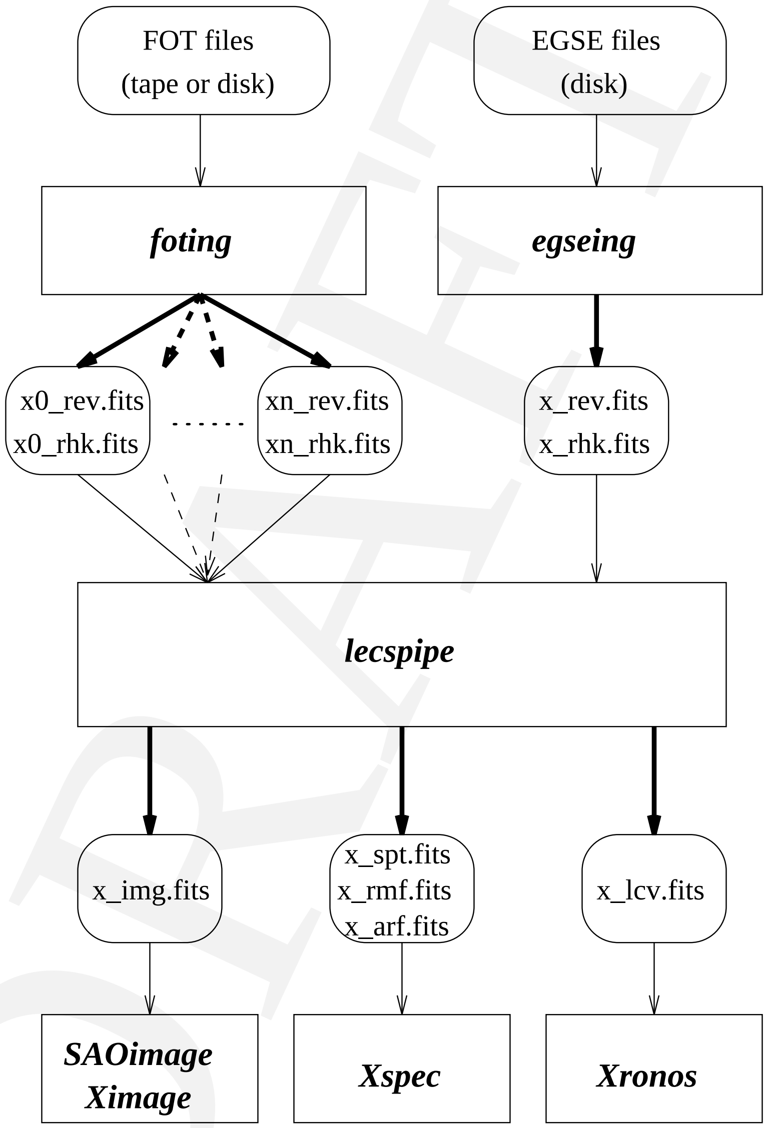


Figure 1: Basic input and output of the SAXLEDAS pipeline

foting and *egseing* are two fundamental modules which create FITS files from FOT telemetry science or EGSE packet files respectively. The raw FITS files represent the basic input to the actual SAXLEDAS symbolized through the module *lecspipe*.

3.2 The pipeline

In more detail, obtaining ”final product” files from raw telemetry files via the module *lecspipe* can be regarded as a four-stage pipeline process as depicted in the following figure:

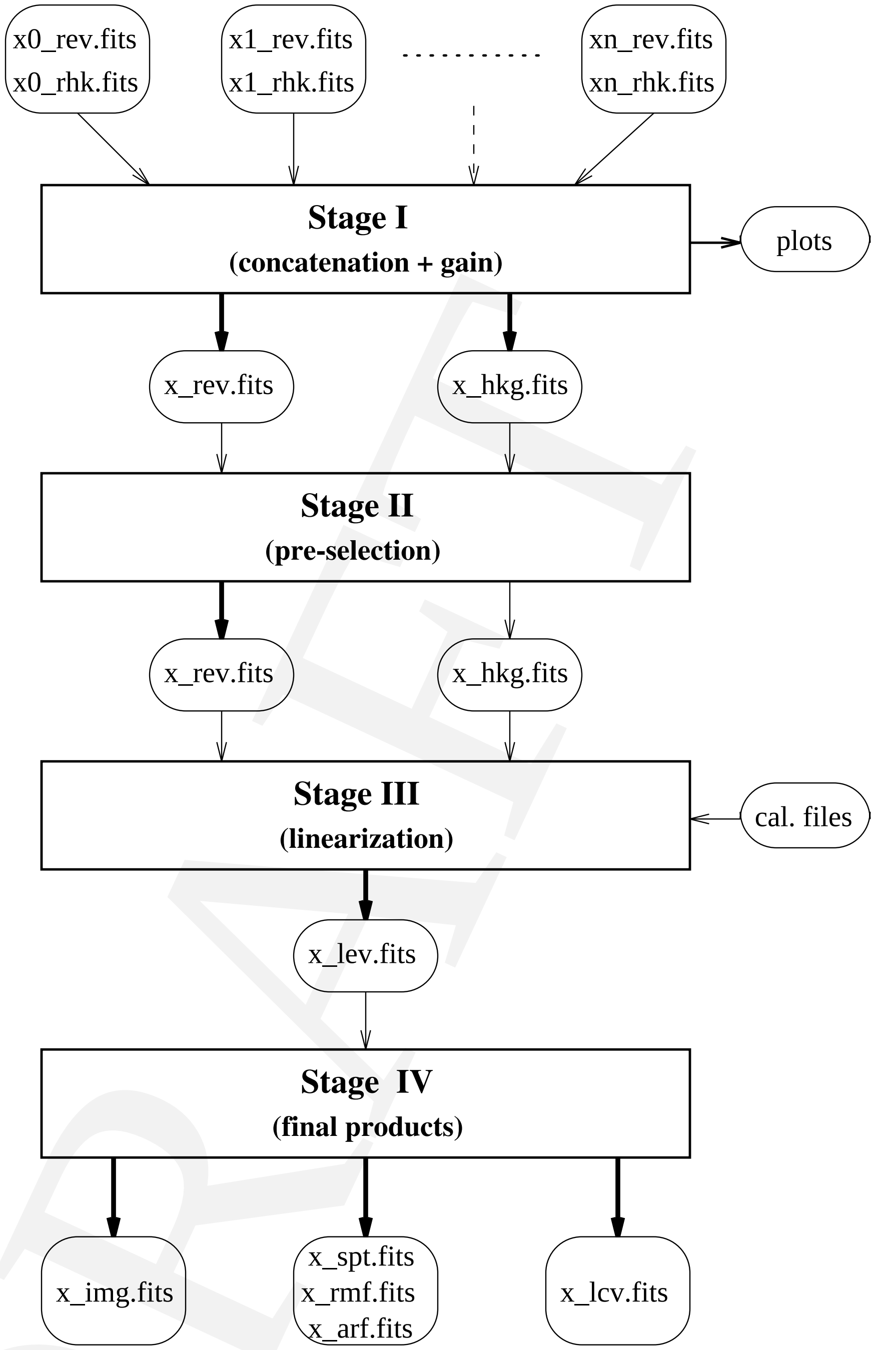


Figure 2: Data flow in the SAXLEDAS pipeline

In the following the various stages will be described in more detail.

3.2.1 Stage I

The following figure shows the data flow in stage I of the pipeline:

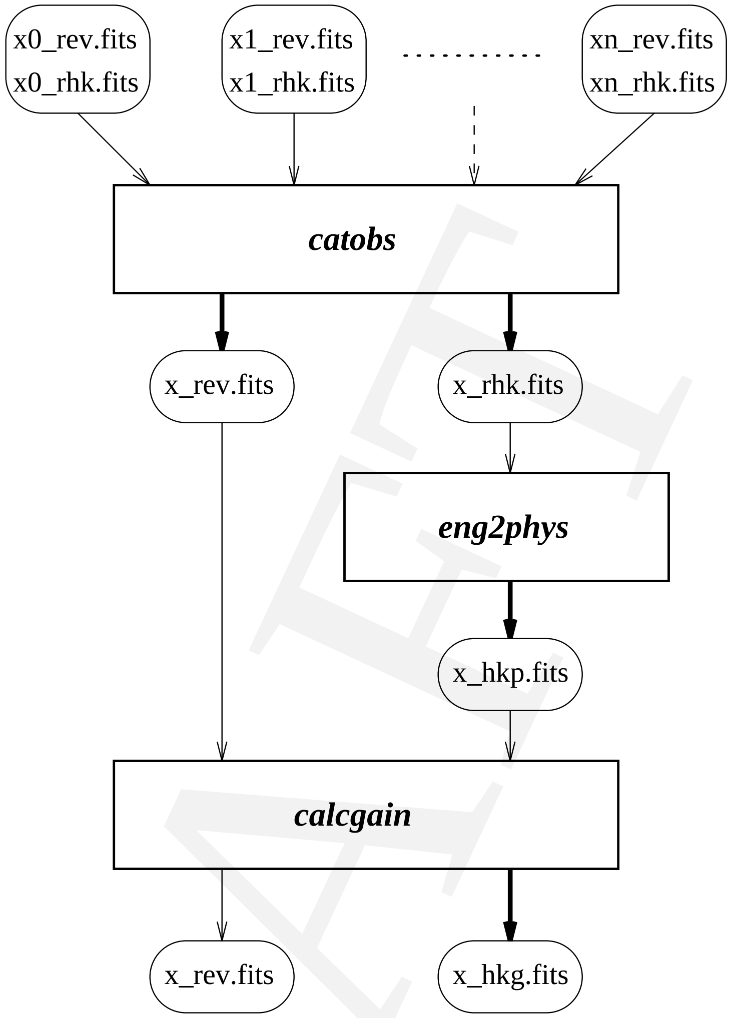


Figure 3: Stage I of the SAXLEDAS pipeline

Input

Input to stage I are one or several raw event list files that have been created with either the EGSE ingestor *egseing* or the FOT ingestor *foting* respectively. In the latter case, each single event list file of type `rev` will correspond to a LEGSPC Direct Mode FOT containing data acquired during an observing interval. Normally, for each event list file there will be a corresponding raw HK file of type `rhk`.

Description

In this stage, three major tasks are carried out:

- Concatenation of individual event list files

- Transformation of units in HK file
- Calculation of the energy gain from the calibration sources

Output The output of stage I consists of

1. A single raw event file that represents the concatenated version of the individual input event files
2. A single FITS HK file of type `hkg` containing HK parameters in physical units and information about the energy gain.

3.2.2 Stage II

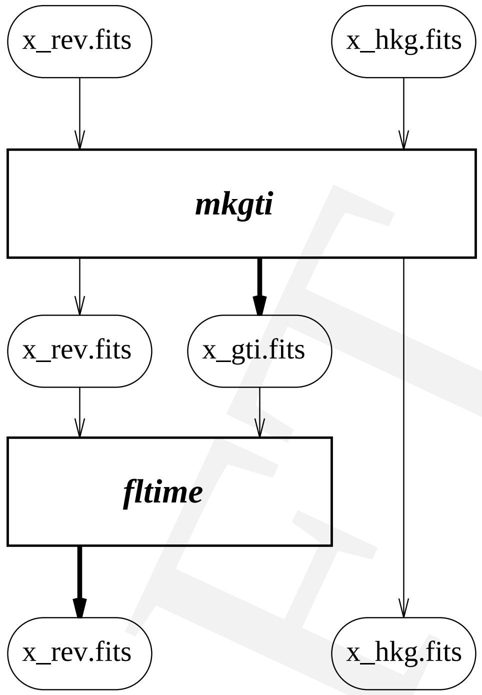


Figure 4: Stage II of the SAXLEDAS pipeline

Input

Description

Output

3.2.3 Stage III

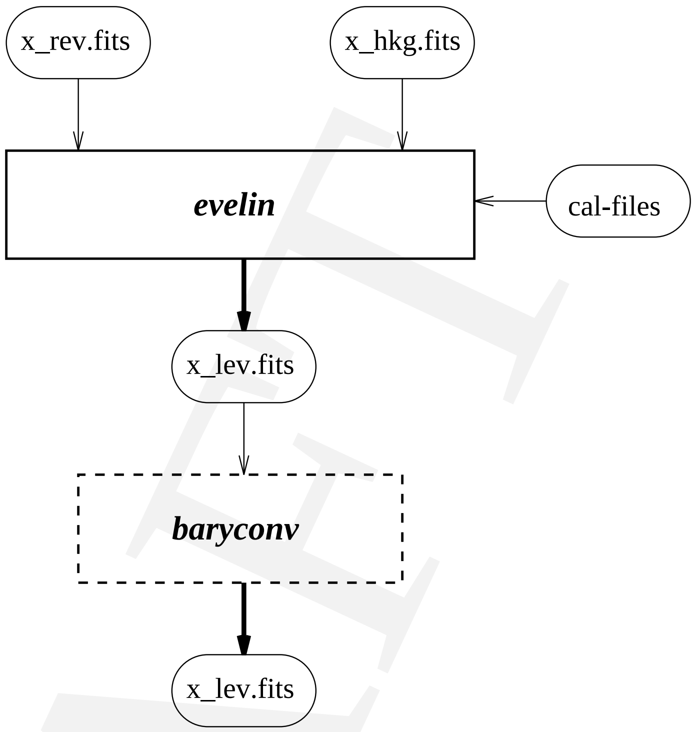


Figure 5: Stage III of the SAXLEDAS pipeline

Input

Description

Output

3.2.4 Stage IV

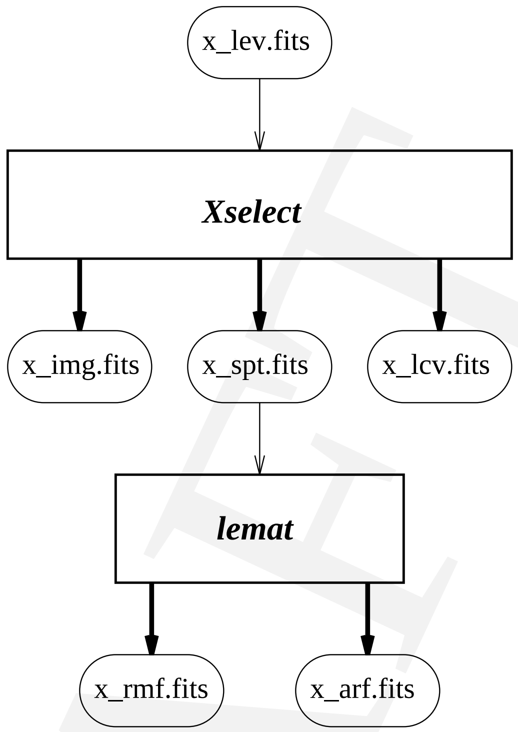


Figure 6: Stage IV of the SAXLEDAS pipeline

Input

Description

Output

3.3 The basic SAXLEDAS components

3.3.1 The control centre and FITS file browser: *scs*

NAME

scs - the SAXLEDAS Control Shell

SYNOPSIS

scs [**parameter=value** ...]

DESCRIPTION

scs is a tool to aid performing data anlysis with the SAXLEDAS interactively. It provides a simple GUI and, thus, does require a graphical environment to run. Upon invocation, *scs* comes up with a window that basically consists of three vertically arranged panels.

Command button area

Terminal window area

Below the "button" area is an ordinary text terminal window which runs the program defined in the SHELL environment variable (usually the user's prefered shell). The terminal window behaves like a common 'xterm' except that, in addition, it is linked to the other elements in the scs-window in a particular way: Whenever the current source directory and/or instrument is changed, the shell in the terminal window follows this change. In addition there is a connection to the event file browser (see below).

FITS file browser area

PARAMETERS

The following command line parameters are recognized by the program:

shellrows **= 24 [integer: 10-30**]
The number of text rows of the terminal window

shellcols **= 80 [integer: 60-120**]
The number of text columns of the terminal window

shellfont **= "10x20" [string**]
The font for displaying text in the terminal window. The standard syntax for specifying X11 font names (including wildcards) is accepted.

shellfgcol **= "black" [string**]
The foreground color of the terminal window - must be a valid X11 color name.

shellbgcol **= "white" [string**]
The background color of the terminal window - must be a valid X11 color name.

wwwbrowser **= "netscape" [string]**
The name of a WWW browser to be spawned when requesting help via the `help` button.

helpurl **= "http://astro.estec.esa.nl/SA-general/Research/Sax/Saxint/saxmain.html" [string]**
The URL for help on SAXLEDAS issues

imgdisplayer **= "saoimage" [string]**
Name of a program capable of displaying FITS images created by the FTOOL task *f2dhisto*.

pgplotdev **= "/xwin" [string]**
The default PGPlot device - will be passed to FTOOL *fplot*.

filelistrows **= 12 [integer: 5-40]**
Number of displayed rows in the filelist browser window

filelistfont **= "-adobe-courier-bold-r-normal–12-*" [string]**
Name of font for displaying entries in the filelist browser window - the standard X11 font name syntax is accepted.

datadir **= "$env(HOME)/datsax" [string]**
The name of the primary data directory to be used. *datadir* can comprise names of environment variables in Tcl-syntax, i.e., `$env(...)` where `...` is the name of an existing environment variable. If the specified directory does not exists, *scs* will attempt to create it.

instrumentdirs **= "LECS MECS" [string]**
A blank separated list of directory names each of which is supposed to contain data from a certain SAX instrument.

instrument **= "LECS" [string]**
Name of the default instrument - must correspond to an element specified in *instrumentdirs*.

keynames **= "EXTNAME DATAMODE SUBMODE OBJECT GUPOW HVPOW AEPOW CELLON PMTON CELLHV PMTHV HANDLER DM_CONT" [string]**
A blank separated list of FITS header keyword names. Two FITS event files are regarded as being suitable for concatenation if the values of each of those keywords match.

filepatt **= "*_rev.fits" [string]**
A regular expression (Tcl syntax). All files matching *datadir*/*instrument*/*filepatt* will be displayed in the filelist browser window.

help **= no [boolean: yes—no]**
If set to yes, the program displays a brief on-screen description of all available command line parameters together with the current program version number and exits sucessfully.

EXAMPLES

1. Start *scs* with `/sax/data` as the primary data directory: scs datadir=/sax/data

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

BUGS

- The scrollbar of the central terminal window does not work

SEE ALSO

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.2 Feeding the pipeline: *egseing*, *foting*

NAME

egseing - the EGSE INGestor program for the SAX/LEGSPC

SYNOPSIS

egseing [**infiles=**] <**filelist**> [**parameter=value** ...]

DESCRIPTION

egseing is a conversion program to process packet data files created with the LEGSPC EGSE and convert them into FITS format. In the case of Direct Mode data, two output files are actually generated, viz., a FITS event list and a corresponding HK file, the structure and format of which is identical to those created with the FOT ingestor *foting* - see the document LEDA0002 for details.

For each EGSE packet file specified on the command line the following steps are performed:

1. Identification of the file type
2. Initial scan of the file
3. ...

PARAMETERS

The following command line parameters are recognized by the program:

infiles **= "" [string]**
This is a blank or colon separated list of files to convert to FITS format. The name extension of each file is used to derive the type of the file. Valid extensions are: `.drt` (Direct Mode), `.spt` (Spectrum Mode), `.imt` (Image Mode), `.hkt` (RB HK Data), `.ent` (Engineering HK Data), and `.mmt` (Memory Dump Data). For Direct-Mode data, the names of the two FITS output files are derived from the input file name, i.e. the events file will get a suffix `_rev.fits` (for Raw EVents) whilst the name of the FITS HK file will have a suffix `_rhk.fits` (for Raw HouseKeeping).

indirpath **= "" [string]**
A colon separated list of directories to seek for the specified input files. For each file given in *infiles* this is performed by first attempting to open the file as specified in *infiles*. If this fails, each directory component in *indirpath* is inspected for the location of the desired input file.

outdir **= "=" [string]**
The destination directory for the FITS output files. If "=" is specified for *outdir* (the default), the output files are written to the directory where the corresponding input file was found.

verbosity **= 2 [integer: 0—1—2—3]**
Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means "mute", only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 1.

help **= no [boolean: yes—no]**
: Display a brief on-screen description of all available command line parameters together with the current program version number.

selectevents **= no [boolean: yes—no]**
: If set to yes, the program prompts for an event selection time interval. The default is ”no” which means that all events contained in the input file will be considered for the output.

caldb **= no [boolean: yes—no]**
: If set to yes, the created FITS files are eligible for inclusion into the SAX/LEGSPC calibration database. In this case, the primary FITS headers of the output files contain the mandatory keywords for FITS calibration files. Default is ”no”.

nohk **= no [boolean: yes—no]**
: If set to yes, the generation of a HK FITS file is suppressed. Default is ”no”.

survey **= no [boolean: yes—no]**
: This boolean switch triggers whether the program should stop after the input file has been inspected for the total accumulation time and possible data integrity errors (see below). In this case, no output FITS file will be created. Default is ”no”.

xreverse/yreverse **= no [boolean: yes—no]**
: Switches to trigger the mirroring of X/Y event detection coordinates. Default is ”no”.

errorcheck **= no [boolean: yes—no]**
: If set to yes, the program inspects the input packet file for data integrity errors (such as errors in the initial and final time stamps of packet headers or block sequence count errors) and prints a summary of detected problems at the end. Default is ”no”.

origin **= ”SSD” [string]**
: Value of FITS header keyword `ORIGIN`.

obs_mode **= ”CALIBRATION” [string]**
: Value of FITS header keyword `OBS_MODE`.

observer **= ”SSD, SAX-TEAM” [string]**
: Value of FITS header keyword `OBSERVER`.

obs_id **= ”00000001” [string]**
: Value of FITS header keyword `OBS_ID`.

tape_id **= 1 [integer: >=1]**
: Value of FITS header keyword `TAPE_ID`.

obs_no **= 1 [integer]**
: Value of FITS header keyword `OBS_NO`. Default is 1.

EXAMPLES

1. Convert the Direct Mode files `sample1.drt` and `sample2.drt` and the corresponding HK file `sample1.hkt` and `sample2.hkt`, respectively, into FITS format. The directories `/sax/data/eqm2`, `/sax/data/fs`, as well as `/sax/data/pfm` are considered as possible locations for the input files. The FITS output files shall be written to the directories where the corresponding input files were found. Detector Y coordinates shall be mirrored and the input files shall be inspected for data integrity errors: egseing ”sample1.drt sample2.drt” yrev+ errorcheck+ indirpath=/sax/data/eqm2:/sax/data/fs:/sax/data/pfm

FURTHER DOCUMENTATION

1. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM,U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

2. SAX LEGSPC USER MANUAL, A. Parmar, ESA/SSD, Issue 8.0

BUGS

- The decoding of Spectrum, Image, and Memory Dump data is not supported
- In case of Engineering HK packet files, merley an integrity check of the time stamps in the secondary header is performed.

SEE ALSO

foting

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

NAME

foting - the SAX Final Observation Tape INGestor program

SYNOPSIS

foting [**parameter=value** ...]

DESCRIPTION

foting is the primary program to read and decode data from SAX Final Observation Tapes (FOTs) as supplied from the Scientific Data Center and convert them into FITS format according the specifications defined in [1]. The nominal flow of operations is the following:

1. Read the FOT tape directory and build up a list of relevant files that have to be processed.
2. Read and decode the ephemeris, attitude and OBT-UTC files which are provided on each default FOT.
3. Loop over all observations performed by a certain instrument during the observation period corresponding to the FOT. For each of these observations a FITS event and HK file is generated. A single observation is defined to be the longest uninterrupted time period of stable pointing to a distinct target. Possible interruptions of stable pointing periods can be external events such as SAA passages, or changes in the instrument configuration.

PARAMETERS

The following command line parameters are recognized by the program:

disk **= no [boolean: yes—no**]
: A boolean switch determining wether FOT files are to be read from disk or tape.

indir **= "./" [string: valid directory name**]
: If FOT files shall be read from disk (*disk*=yes), this parameter determines the source directory under which all relevant files are expected. Note that *foting* always attempts to open the tape directory file as

trimfotdir **= "none" [string: name of external program**]
: This is a name of an external program that will be invoked by *foting* in case *disk*=yes prior to open the tape directory file. This program is supposed to "trim" the names of the FOT files in the source directory *indir* in case they do not follow the default naming convention:

```
[XX]type[tape]\_YY\_Z.dat
```

where the various fields have the following meaning:

- `XX`- the instrument ID (e.g. `le`); empty (i.e. "") for non-instrument specific files
- `type`- file type field from tape directory, e.g. `ephemeris`, `ledir001`, ...
- `tape`- the word `tape` for "directory" files, e.g. `tapedir`, `instdir`, `obsdirYY`- the observation number
- `Z`- `n`, `s`, or `0` depending on type of observing period ('N', 'S', or '**')

foting will change into the right directory before it attempts to run the name-trimming program. No program will actually be run in case *trimfotdir* is given the special value "none" or no value at.

tapedev **= "-" [string: valid device file name or "-"]**
: If FOT files shall be read from tape (*disk*=no), this parameter determines the tape device to be used. "-" signifies to utilize an installation dependant default device (actual value can be checked with "foting help=yes").

instrument **= "-" [string: "LE—ME—M1-3—HP—PD—P1-4A-C—W1—W2—**—ALL—-"]**
: This parameter determines which kind of instrument specific files are to be read. The permitted values for *instrument* are indicated above, multiple choices - separated by blank or '+' - are possible. Specifying "-" means to actually use the installation dependant default value.

namestem **= "fot" [string]**
: This establishes the stem of the names for event and HK FITS files that are to be created. The actual complete name of a FITS file created by *foting* has the form

`<namestem>_<FOTID>_<InstID>_<ObsNr>_<ObsType>_<Nr>_<ID>.fits`

where `FOTID`, `InstID`, `ObsNr`, `ObsType`, `Nr`, and `ID` stand for the six-digit FOT sequence number, a unique two-letter instrument ID (`LE` for the LEGSPC), the observation interval number, a single-letter observing type ID (`i`, `f`, `n` for initial slew, final slew, and normal observing period), the running file number in the current observing period, and `ID` is a three-letter identification of the type of the SAXLEDAS FITS file, e.g. `rev` for "Raw EVent" or `rhk` for "Raw HouseKeeping".

outdir **= "./" [string]**
: The name of the directory in which the output files shall be created.

obtutcplot **= no [boolean: yes—no]**
: A boolean switch to control the creation of a plot showing the relation between the S/C clock (OBT) and UTC expressed in seconds since the the fixed mission reference time 01/01/94 00:00:00 (TAI - 28s). This relation is a regression line derived via a standard least-square fit to the distinct OBT-UTC data points from the OBT_UTC file on the FOT. After the plot is displayed the user shall be prompted for a confirmation to proceed with the current OBT-UTC relation.

plotdev **= "/xwin" [string: a valid PGPLOT device]**
: The plotting device to be used to create the OBT-UTC plot. All devices supported by the FTOOLS PGPLOT library can be specified.

origin **= "-" [string]**
: Value of FITS header keyword `ORIGIN`. "-" means to use an installation dependant default value - 'foting help=yes' shows this default value.

obs_mode **= "-" [string: "POINTING—SLEW—CALIBRATION—-"]**
: Value of FITS header keyword `OBS_MODE`. "-" means to determine the value from corresponding information on the FOT.

observer **= "-" [string]**
: Value of FITS header keyword `OBSERVER`. "-" means to determine the value from corresponding information on the FOT.

obs_no **= 1 [integer: >=1]**
Value of FITS header keyword OBS_NO. Default is 1.

xreverse/yreverse **= no [boolean: yes—no]**
Switch to trigger the mirroring of X/Y event detection coordinates. Default is ”no”.

nohk **= no [boolean: yes—no]**
If set to yes, the generation of a HK FITS file is suppressed. Default is ”no”.

verbosity **= 2 [integer: 0—1—2—3]**
Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means ”mute”, only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 2.

help **= no [boolean: yes—no]**
Display a brief on-screen description of all available command line parameters together with the current program version number.

EXAMPLES

1. Process all files related to the SAX LE and ME instrument on the current FOT. Suppress the creation of HK files: foting instrument=LE+ME nohk=yes

2. Read all LECS specific FOT files from a disk directory `/fot` and create all resulting FITS output files in user’s home directory foting disk=yes indir=/fot outdir=$HOME instrument=LE

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

2. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM,U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

3. MSC FOT LAYOUT DOCUMENT - USER MANUAL, Telespazio, Doc. TPZ/GO.SDC-100/FOT-TN/SAX

BUGS

- For the LE, merely the decoding of Direct Mode science and HK files is supported (no Indirect and Image Mode).
- The decoding of S/C HK files is not supported.

SEE ALSO

egseing

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.3 Find compatible event list files: *compevfiles*

NAME

compevfiles - find COMPatible EVent list FILES

SYNOPSIS

compevfiles [**evfilename**=] <**rev_filename**> [**parameter=value** ...]

DESCRIPTION

compevfiles is a tool to find all FITS event list files in the current working directory that are compatible with a specified one. Compatability of event list files in this context is defined as equality of the instrument configurations under which the corresponding data have been acquired. Actually, if the values of a set of header keywords in the event extension of two files are found to match, those files are considered to be compatible. The list of keywords used for this equality check can be specified via the parameter *keynames* (see below). Upon successful return the list of name of compatible event list files is printed on stdout in the proper temporal order, i.e., in ascending order of the values of the *TSTART* keyword. This resulting list can be directly used as value of the command line parameter *infiles* of the *catobs* module for concatenating event list files.

PARAMETERS

The following command line parameters are recognized by the program:

evfilename **= ”” [string: name of existing event list file**]
All files that are compatible with *evfilename* are sought.

keynames **= ”EXTNAME DATAMODE SUBMODE OBJECT GUPOW HVPOW AEPOW CELLON PMTON CELLHV PMTHV HANDLER DM_CONT” [string**]
A blank separated list of FITS keyword names in the event extension of SAX/LECS event list files. Two files are considered to be compatible if the respective values of each of those keywords are equal. If a list element in *keynames* is not a valid event extension keyword name, it will simply be ignored. For a complete list of valid keyword names, see Ref. [1] or [2]

matchpatt **= ”*_[rl** ev.fits” [string: regular expression (Tcl syntax)]]
A file matching pattern in the form of a regular expression. This pattern will be used to obtain an interim list of files that are to be checked for compatibility with the file specified as *evfilename*.

help **= no [boolean: yes—no**]
A boolean flag which, when set to *yes* triggers that the program prints out a brief on-screen description of all available command line parameters together with the current program version number and exits successfully.

EXAMPLES

1. Print on stdout a time ordered list of all files in the current working directory ending in `.fits` that are compatible with the event list file `kv620_rev.fits`: compevfiles kv620_rev.fits matchpatt='*.fits'

2. Find all event list files in the current working directory that are compatible with the file `kv620_rev.fits`. File shall be considered compatible if the values of the *OBJECT* and *EXTNAME* keywords match: compevfiles kv620_rev.fits keynames="OBJECT EXTNAME"

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

2. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM,U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

BUGS

None known.

SEE ALSO

catobs

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.4 Concatenating event list files: *catobs*

NAME

catobs - conCATenate files belonging to same OBServing period

SYNOPSIS

catobs [**infiles=**] <**file_names**> [**parameter=value** ...]

DESCRIPTION

catobs is used to concatenate LECS event list files (both, `_rev` and `_lev` types are supported) that logically belong to the same observing period into one larger sum file. For each input file it is checked whether a corresponding science HK file exists. If this is the case, the program creates a sum HK file as well. The primary and all Bintable extension headers of the sum files are essentially copies of the corresponding headers of the first file in the list of files specified for concatenation. All time related keywords, such as `TSTOP`, `TELAPSE`, etc. are updated where necessary to reflect the larger time coverage of the sum files. In addition, history keywords are added to the primary headers and the `N_OBIS` keyword value is set to the number of input files.

WARNING: The list of input files given as *infiles* is strictly processed from left to right. Therefore, to ensure a proper time order of the event list data in the sum file, the individual files have to be specified in the right temporal order, i.e., in ascending order of the values of the *TSTART* keyword. Also, the program expects the very last file in the *infiles* list to contain the largest time tag values. Under this assumption the values of various time-related header keywords in the sum events files are derived from the headers of this last file in the list. The *compevfiles* task can be used to obtain a list of compatible event list files in the proper temporal order. In fact, the output of *compevfiles* can be directly used as value for *infiles* as demonstrated in Example 2 below.

PARAMETERS

The following command line parameters are recognized by the program:

infiles **= "" [string: list of existing event list files**]

A blank, or colon-separated list of event files to concatenate. If blank is used as file name separator, the list has to be enclosed by "" in order to be considered as a single command line argument. Note that the files are strictly processed from left to right and, consequently, have to be in the proper temporal order (see Warning note above). If a file in *infiles* does not exist or is found to be impossible to be appended to the sum file, a warning message is issued and the processing continues with the next file in the list.

indirpath **= "" [string: list of directory names**]

A colon separated list of directories to seek for the specified input files. For each file given in *infiles* this is performed by first attempting to open the file as specified in *infiles*. If this fails, each directory component in *indirpath* is inspected for the location of the desired input file.

stemoutfile **= "-" [string**]

This determines the file name stem of the sum file(s) to be created. The stem of a file name is defined to be everything before any extension `_???.fits`, e.g., the name stem

of `kv492_rev.fits` is just `kv492`. Specifying "-" for *stemoutfile* means that the actual value shall be derived from the name of the first file in the *infiles* list which is expected to follow the naming scheme used by the FOT ingestor *foting*: <Stem>_<FOT_ID>_<Instrument_ID>_<ObsID>_<Number>_rev.fits
If the first file is found to match this pattern the sum file will get the name <Stem>_<FOT_ID>_<Instrument_ID>_rev.fits
otherwise, `sum_rev.fits`.

outdir **= "=" [string: name of existing directory or "="]**
The name of a directory to which the output file(s) shall be written. "=" means that the resulting sum file(s) shall be created in the directory the very first file in the *infiles* list was found. Note that write access for *outdir* is needed.

hkconcat **= yes [boolean: yes—no]**
If set to "yes", for each input event file it is checked whether a corresponding science HK file exists. If this is the case, a concatenated version of all single HK files is created similar to the event list sum file. "no" as value suppresses the HK sum file creation.

rmfiles **= no [boolean: yes—no]**
If set to "yes", the individual input files are removed after the corresponding sum file(s) has/have been successfully created.

printstem **= no [boolean: yes—no]**
If set to "yes", this triggers that the program does nothing but merely prints out the filename stem of the sum file and exits successfully.

verbosity **= 2 [integer: 0—1—2—3]**
Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means "mute", only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 2.

help **= no [boolean: yes—no]**
A boolean flag which, when set to *yes* triggers that the program prints out a brief on-screen description of all available command line parameters together with the current program version number and exits successfully.

EXAMPLES

1. Concatenate the three event files `1_rev.fits`, `2_rev.fits`, and `3_rev.fits`. The resulting output file shall be named `123_rev.fits`: catobs "1_rev.fits 2_rev.fits 3_rev.fits" outfile=123_rev.fits

2. Concatenate all event list files that are compatible with the file `kv467_rev.fits` using the task *compevfiles*. The sum file should get the default name. HK file concatenation shall not take place and the individual input files shall be removed after the sum file has been successfully created: catobs "‘compevfiles kv467_rev.fits‘" hkconcat=no rmfiles=yes

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

2. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM, U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

BUGS

None known.

SEE ALSO

compevfiles,

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.5 Physical units in HK files: *eng2phys*

NAME

eng2phys - convert HK data from ENGineering TO PHYSical units

SYNOPSIS

eng2phys [**infile=**] <**file_name**> [**parameter=value** ...]

DESCRIPTION

eng2phys is used to scale the original digital telemetry values in a raw LEGSPC HK file to physical units, i.e. pressure monitor readings to ”mbar” and digital temperature data to ”Deg Celsius”. The program simply reads a raw LEGSPC HK file (suffixed ”_rhk.fits”) and writes out the resulting new file (suffixed ”_hkp.fits”) in which the data types of certain Bintable columns have changed from ”I” to ”E”. Both types of HK files (”_rhk” + ”_hkp”) can be inspected, for instance, with the *hkplot* module.

PARAMETERS

The following command line parameters are recognized by the program:

infile **= ”” [string]**
The raw HK FITS file to be transformed. The name of the output file is derived from *infile*, i.e., a potential suffix ”_rhk” is changed to ”_hkp”.

indir **= ”” [string]**
A colon separated list of directories to seek for the specified HK input file *infile*. This is done by first attempting to open *infile* as is. If this fails, each directory component in *indir* is inspected for the location of *infile*.

outdir **= ”=” [string]**
The destination directory for the output HK file. If ”=” is specified for *outdir* (the default), the output file is written to the directory where the input file was found.

verbosity **= 2 [integer: 0—1—2—3]**
Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means ”mute”, only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 1.

help **= no [boolean: yes—no]**
Display a brief on-screen description of all available command line parameters together with the current program version number.

EXAMPLES

1. Scale contents of raw HK file ”con2_cont2_tv_dupfm_ft13_rhk.fits” to physical units and create resulting output file ”con2_cont2_tv_dupfm_ft13_hkp” in the directory where the input file was found. Verbosity level is 2: eng2phys con2_cont2_tv_dupfm_ft13_rhk.fits verb=2

2. Scale contents of raw HK file ”al_vac_dupfm_ft22_rhk.fits” to physical units, seekig it in the directories ”./”, ”../data/FuncTest”, and ”/data/legspc”. The resulting output file is generated in the current directory: eng2phys al_vac_dupfm_ft22_rhk.fits indirpath=”.:../data/FuncTest:/data/legspc” outdir=”.”

FURTHER DOCUMENTATION

1. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM, U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

BUGS

None known.

SEE ALSO

foting, *egseing*, *hkplot*

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.6 Monitoring the gain: *calcgain*

NAME

calcgain - CALCulate time-dependent energy GAIN of Fe calibration sources and monitor background count rate

SYNOPSIS

calcgain [**infile=**] <**rev_filename**> [**parameter=value** ...]

DESCRIPTION

The *calcgain* task calculates the time-dependent energy gain of the two fixed iron calibration sources inside the SAX/LEGSPC and, in addition, monitors the background count rate outside the detector's FOV. The program requires a raw event list file that comprises all calibration source events. The computed quantities are stored as Bintable columns in a new HK file of type `_hkg` whose structure closely resembles the `_hkp` type (see [2] for details). Regarding the nominal flow of operations, three major steps can be identified:

1. Read a SAX/LEGSPC raw events FITS file and select all events originating from the two Fe calibration sources. This selection is performed with the aid of two spatial windows whose position and size can be specified via command line parameters (see below).
2. Divide the total exposure time (= arrival time of last minus arrival time of first photon) into *N* equal intervals and subsequently, for each interval center, accumulate a PHA spectrum over a time period *T*. This step will generate for each calibration source a set of *N* spectra (counts vs. PHA channel number) which will be used to define the time-dependent gain $g^j(t_i) := PHA_i^j/PHA_{nom}^j$. $g^j(t_i)$ should designate the gain at time t_i (i=1,...,N) for calibration source #j, PHA_i^j is the centroid of the Gaussian curve that has been fitted to the accumulated spectrum #i for source #j, and PHA_{nom}^j is the nominal peak position (in PHA channels) of the Fe line for source #j. In case of poor statistics, i.e., if an accumulated spectrum is found to contain less than 25 counts, *calcgain* will not attempt to perform a Gaussian fit but set all gain related quantities (see below) to the value `INDEF`.
3. Create a new HK FITS file with the name extension `_hkg.fits`. The content of this file is identical to the HK file of type `hkp` (created with *eng2phys*) that corresponds to the event input file except that 13 columns have been added to the HK parameter extension. These are:

   - `PHA1`, `PHA2` - centroid of Gaussian fitted to PHA spectra of calibration source #1/#2
   - `FWHM1`, `FWHM2` - Full-Width-Half-Maximum value of fitted Gaussians for sources #1/#2
   - `CHISQ1`, `CHISQ2` - reduced Chi^2 values of Gaussian fits
   - `GAIN1`, `GAIN2` - gain values as defined above for calibration sources #1 and #2
   - `GAIN12` - the ratio `GAIN1`/`GAIN2`
   - `GAIN_AVG` - the average of `GAIN1` and `GAIN2`
   - `CHISQAVG` - reduced Chi^2 value corresponding to `GAIN_AVG`

– `BCKGND1`, `BCKGND2` - the average background count rate evaluated through event accumulation in two spatial windows outside the FOV.

The HK parameter table contains a HK record for each second, however, changes in the energy gain will typically occur on timescales larger than 1s. Thus, for the sake of efficiency the default time granularity for the gain computation is set to a larger value *Tg* and filling-up the new HK table columns is done by repeated usage of the respective actual values. The effective value for *Tg* - in other words the above number *N* of time intervals - can be specified through a command line parameter *granularity*.

The primary output of *calcgain*, i.e., the gain related columns in the newly created HK file will be used by the *evelin* task to apply the temporal gain correction to scientific event list data.

PARAMETERS

The following command line parameters are recognized by the program:

infile **= "" [string: name of existing** `_rev` **file]**
The raw events FITS input file which shall be processed. This file must contain calibration source event data, otherwise, the energy gain cannot be calculated. In such a case, all gain related HK table columns will bet set to the value `INDEF`.

granularity **= 10 [real: >= 1]**
This parameter determines the granularity of the gain computations in s. Note that the actual granularity taken will slightly differ from the one specified in order to get an integer number of equal time intervals. Lower values than 1s - the HK parameter time granularity - are not accepted. Default is 10s.

winsize **= 20 [real: >= 10]**
Size in s of the time window which is used for accumulating spectra. Note that with a given calibration source strength of approx. 25 cnts/s, *winsize* values lower than 10s would lead to very poor event statistics and possibly meaningless Gauss-fits. Default is 20.

fitfun **= "S" [string: S—D]**
A switch to control the type of Gaussian fit function to use. "D" stands for a double Gaussian of the form f(x) = p1 * exp(-(x-p2)^2/p3^2) + p4 * exp(-(x-const*p2)^2/p3^2)

where the constant is the ratio of the energies of the two Mn lines the calibration source iron lines consist of. Specifying "S" as value for *fitfun* forces *p4* to be zero, i.e., a simple Gaussian will be used.

xmin1/xmin2 **= 1/214 [integer: 0-255]**
X coordinate of lower left corners of spatial windows used to select events from calibration sources #1 and #2 respectively.

ymin1/ymin2 **= 210/1 [integer: 0-255]**
Y coordinate of lower left corners of spatial windows used to select events from calibration sources #1 and #2 respectively.

xsize1/xsize2 **= 44/40 [integer: 1-256]**
Pixel sizes in X direction of calibration source windows #1 and #2.

ysize1/ysize2 **= 44/40 [integer: 1-256]**
Pixel sizes in Y direction of calibration source windows #1 and #2.

xbmin1/xbmin2 **= 1/180 [integer: 0-255]**
X coordinate of lower left corners of background monitor windows #1 and #2.

ybmin1/ybmin2 **= 1/180 [integer: 0-255]**
Y coordinate of lower left corners of background monitor windows #1 and #2.

xbsize1/xbsize2 **= 75/75 [integer: 1-256]**
Pixel sizes in X direction of background monitor windows #1 and #2.

ybsize1/ybsize2 **= 75/75 [integer: 1-256]**
Pixel sizes in Y direction of background monitor windows #1 and #2.

fe1_peak_pha **= 410 [integer: 350-450]**
The nominal Fe line peak position of calibration source #1 in PHA channels. Default is 410.

fe2_peak_pha **= 410 [integer: 350-450]**
The nominal Fe line peak position of calibration source #2 in PHA channels. Default is 410.

hkgfile **= yes [boolean: yes—no]**
A switch to control the creation of a new HK FITS file containing the calculated quantities, e.g., energy gain and background count rates. If *hkgfile* is set to ”no”, the generation of the `hkg` file is suppressed and the essential informations will be printed on the screen.

plotspec **= -1 [integer: $>$=-1]**
If *plotspec* is assigned a value $i >= 0$, a plot of the accumulated spectrum #*i* together with the fitted Gaussian will be created. This enables a simple visual control of the quality of the fit.

plotdev **= ”/XWIN” [string: name of valid PGPLOT device]**
A string determining a plot device on which a requested spectral plot (see above) shall be created. All devices supported by the FTOOLS PGPLOT library can be specified.

verbosity **= 2 [integer: 0—1—2—3]**
Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means ”mute”, only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 2.

help **= no [boolean: yes—no]**
A boolean flag which, when set to *yes* triggers that the program prints out a brief on-screen description of all available command line parameters together with the current program version number and exits successfully.

EXAMPLES

1. Calculate the energy gain from the raw events file `kv626_rev.fits` with a time granularity of 100s and a time window size of 200s; program verbosity level is 1: calcgain kv626_rev.fits gran=100 win=200 verb=1

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

2. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM, U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

BUGS

None known.

SEE ALSO

eng2phys, *evelin*

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.7 Define good time intervals: *hkplot*

NAME

hkplot - HK parameter PLOTting and GTI creation task

SYNOPSIS

hkplot [**infile=**] <**file_name**> [**paramlist=**] <**col_names**> [**parameter=value** ...]

DESCRIPTION

hkplot is used to create time series plots of SAX/LEGSPC HK parameters and, optionally, define GTIs from those plots by making time range selections with the mouse cursor. Up to 8 different parameters can be simultaneously displayed in different, vertically arranged panels. In each of those panels the user may define an arbitrary number of time intervals from which a standard GTI FITS file is generated that can subsequently be used for event selection. The QDP/PLT package which is based on the graphics library PGPLOT is used for plotting purposes. Thus, *hkplot* supports all graphics output devices for which the underlying PGPLOT module has been configured.

After the corresponding data have been read in from the HK input file, *hkplot* outputs the requested plots to the specified device and enters the interactive PLT environment, allowing to modify the appearance of the plot (rescaling, adding labels, etc.). The user exits from PLT with 'Quit' and then has the possibility to define individual time intervals with the mouse cursor (if input device has this capability). Actions are triggered by pressing keys/mouse buttons:

```
Key / Mouse Button   Function
----------------------------------------------------
'A' / Left           begin/finish an interval
'D' / Middle         delete last point/time interval
'X' / Right          delete all GTIs defined so far
'x','q'              exit GTI definition
```

After exiting with 'x' or 'q', a standard GTI file comprising the interval definitions made is generated. Its name is derived from the name of the HK input file.

PARAMETERS

The following command line parameters are recognized by the program:

infile **= "" [string]**
: The name of the HK input file.

paramlist **= "" [string]**
: A list of HK parameters represented by their corresponding column names in the HK Bintable extension of the input file. Case is not significant and all names have at least one unique abbreviation, e.g. 'ct' for CELL_TEMPERATURE. To get a list of all valid HK parameter names together with possible abbreviations, invoke *hkplot* with paramlist = '?'.

merge **= "OR" [string: "AND—OR"]**

This parameter determines how GTIs defined in different HK plot panels are merged together. In the default OR mode, the final set of GTIs is constructed by logically ORing all single intervals. In AND mode, the final set of GTIs consists merely of those intervals that are common to all GTIs from the various HK panels. Panels for which no GTIs have been defined are not considered. Example:

```
GTI 1:                 |-------------|
GTI 2:                          |----------|
resulting OR-GTI:      |--------------------|
resulting AND-GTI:              |---|
```

times **= "-" [string]**

A comma or semi-colon separated list of time intervals referring to the time tags in the HK input file. For each interval specified, a separate plot window will be generated displaying merely the HK parameter evolution during this period. The default is times = "-" meaning to display the whole time period covered by the input file in only one window. "-" given as an lower or upper limit of a time interval specification is taken as the smallest and largest time tag of the input file respectively. Example: times = -100,130-140,150- will pop up three plot windows (one after another) displaying the HK parameter time evolution during the periods [min_t, 100], [130, 140], and [150, max_t].

device **= "/XWIN" [string]**

A string specifying the desired output device for plotting. Each device for which the PGPLOT library has been configured is valid.

pltcmd **= "" [string]**

Any legal PLT command that is appended to the list of default PLT commands automatically generated to label the axes and provide a title of the plot based on the input file name. If *pltcmd* is in the form @filename the generation of default commands is suppressed. Instead, the file ".pco" in the current working directory is taken as a source for PLT commands to be interpreted.

gtidef **= yes [boolean: yes—no]**

A boolean switch pertaining to the interactive definition of GTIs. If set to "no", the program merely creates the requested HK time series plots without prompting the user to make interval selections.

verbosity **= 1 [integer: 0—1—2—3]**

Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means "mute", only fatal error messages will be printed. Default is level 1.

help **= no [boolean: yes—no]**

If set to "yes", issue a brief on-screen description of all available command line parameters together with the current program version number.

EXAMPLES

1. Display the time evolution of the three HK parameters cell temperature, cell pressure, and EU temperature in the interval t = 1000 - 2000 s from file "cont5_hkp.fits". GTIs defined in different

plot panels should be AND-merged in the end: hkplot cont5_hkp.fits ”ct cp eut” times=1000-2000 merge=and

2. Plot the HK parameters CELL_HIGH_VOLTAGE and PMT_HIGH_VOLTAGE for the whole time period spanned by the input file ”sample_hkp.fits”. The definition of GTIs is not required: hkplot sample_hkp.fits ”CELL_HIGH_VOLTAGE PMT_HIGH_VOLTAGE” gtidef-

FURTHER DOCUMENTATION

1. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM,U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

BUGS

None known.

SEE ALSO

fplot, fltime (both FTOOLS)

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.8 Refine raw event list data: *evelin*

NAME

evelin - the main SAX/LECS EVEnt LINearization module

SYNOPSIS

evelin [infiles=] <**file_names**> [**parameter=value** ...]

DESCRIPTION

evelin is the main linearization module of the SAX/LEGSPC data analysis system. Basically, it transforms raw event FITS files (type: "_rev") into linearized forms (type: "_lev") in which several basic corrections have been applied to the data. These are:

1. All events outside the detector's FOV have been removed.
2. The raw PHA values have been corrected for temporal changes in the energy gain (based on continuous monitoring of Fe calibration sources).
3. The PHA values are corrected for spatial variations in the energy gain across the detector's focal plane (based on spatial gain maps from calibration database).
4. A basic event filtering has been performed to remove "non-scientific" events from the data.
5. The focal-plane coordinates of detected photons have been "linearized" to account for intrinsic distortions that are inherent to the detector.
6. Instrumental dead-time effects have been taken into account.
7. The corrected focal-plane coordinates have been mapped onto sky coordinates (standard tangent-plane projection)

The default behavior is to perform all tasks in one run, although, each individual correction can be suppressed by setting a corresponding boolean command line switch. The program is able to deal with two different types of event files that are symbolized by the name extensions `_rev` and `_lev`. `_rev` is the default extension for a raw event file which is the primary output from the ingestor modules *foting* and *egseing* respectively. `_lev` stands for "linearized events" and this represents the primary output of *evelin*. The `_lev` type files differ from the raw version by five additional Bintable columns that have been added to the event extension. These are (see [2] for further details):

- `PI`- the Pulse Invariant of the event in units of detector channels
- `PIC`- PI additionally corrected for gas-cell penetration effect
- `DETX`- the linearized X-coordinate of the event
- `DETY`- the linearized Y-coordinate of the event
- `X`- sky-projected X coordinate of photon
- `Y`- sky-projected Y coordinate of photon

Thus, *evelin* accepts its own output again as input which is needed if one wants to do the aforementioned multi-step linearization process successively in more than one run. If the user requests a linearization step that has already been performed in a previous run, a warning message is issued and the request is ignored. The various build-in linearization stages of *evelin* are described in more detail in the following sections.

EVENTS REMOVAL This first processing step simply removes all events outside the detector's FOV. Note that this pertains also to the events from the calibration sources which ly in the outermost corners of the focal plane. The detector's FOV is defined as a circular region with a raw pixel radius of 100 (alterable via command line parameter *fovradius*) centered around the point where the optical axis intersects the focal plane (approx. raw coordinates: 132/125).

TEMPORAL GAIN CORRECTION This correction accounts for temporal changes in the detector's energy gain as a consequence of changes in operating conditions, e.g. temperature, pressure etc. The actual time-dependent energy gain is calculated using the pipeline module *calcgain* and contained as column data in a HK FITS file of type `_hkg`. Thus, *evelin* first attempts to open a `_hkg` file corresponding to the current event list file and read the gain information from a Bintable column named `GAIN_AVG` (can be changed via command line parameter *tgaincol*). The time granularity of those data is always greater than 10s and, therefore, they have to be interpolated which can either be done by means of a linear regression function or a cubic spline (controlled via command line parameter *tgainint*). Prior to actually performing the temporal gain correction on the PHA values, the interpolation function together with the gain data can be plotted (see parameter *tgainplot*). After the plot the user is prompted for an acceptance confirmation. If this is answered positively, the actual correction is done, i.e., the PHA value of each event i is modified according to

$$\mathrm{PI}_i = \mathrm{PI}_i/\mathrm{gain}_t(t_i) \tag{1}$$

where PI is a new Bintable column in the event extension of the newly created output file. Since the PHA and PI values are integer quantities performing the above prescription *as is* leads to an artificial "integer quantization effect". In order to avoid this, a random value in the range [-0.5, 0.5] is added to each PHA value before it is divided by the actual gain factor. Further information about gain related issues can be found in [4].

SPATIAL GAIN CORRECTION In addition to the time dependence, the energy gain is spatially not homogeneous, i.e., varies across the detector's focal plane. Hence, the energy channel of each photon has to be corrected according to its detection coordinates. This is done with the aid of a pre-processed spatial gain map file which, either, stems from the SAX/LEGSPC calibration database, or can be explicitly specified (see command line parameters *sgaincaldb* and *sgainfile*). The spatial gain map file contains basically a two-dimensional array which, when indexed with raw X and Y coordinates, yields the spatial energy gain at that location. This is used to correct the PI values of each photon according to the simple prescription:

$$\mathrm{PI}_i = \mathrm{PI}_{i,\mathrm{old}}/(\mathrm{const}\cdot\mathrm{gain}_s(x_i, y_i)) \tag{2}$$

where PI_i_old is the already temporally corrected energy channel of photon #i and "const" is an alterable correction factor with default value 1 (see command line parameter *sgainfactor*).

Similar to the temporal gain correction, a random value in the range [-0.5, 0.5] is added to the integer PI values before they get divided by the spatial gain factors. Further information about gain related issues can be found in [4].

EVENT SELECTION Not all events that are detected inside the instrument's FOV are of actual interest for the further scientific analysis of the data. Examples are particle events originating from cosmic rays that have been absorbed in the gas cell or spurious, so-called double events that are detected on a line joining the two calibration sources. Those events can be identified through peculiarities in their Veto and BL signals, and be removed via a discrimination against appropriately chosen interval thresholds of those quantities. This is done by performing a standard one-dimensional Veto cut in combination with a two-dimensional selection in PI-BL space. The event-selection stage is controlled via several command line parameters, among which the most important one is blvselfract. It determines the fraction of photons that will be retained by the combined selection at all energies. More detailed information on the event selection can be found in [6].

COORDINATE LINEARIZATION Due to intrinsic distortions inherent to the detection mechanism, the raw spatial event coordinates as yielded by the instrument's electronic unit do not directly reflect the physical location where the photon has hit the focal plane. To gain this information, a non-linear correction of the form

$$\begin{aligned} C_i &= a_i^0 + a_i^1 x + a_i^2 y + a_i^3 xy + a_i^4 x^2 + a_i^5 y^2 + a_i^6 x^2 y + \\ &\quad + a_i^7 xy^2 + a_i^8 x^3 + a_i^9 y^3 \end{aligned}$$

has to be applied. Here, *i* is either x or y, *C*_{*x/y*} are the desired physical coordinates of the photon in the focal plane (measured from the center of the FOV), *x/y* are the raw detection coordinates referred to the center of the FOV, and *a0_i* to *a9_i* are constant coefficients. For more details on the transformation and related issues, see [5]. Per default, *evelin* attempts to read the two sets of *a*-coefficients from a corresponding FITS data file in the SAX/LECS calibration database. If this is not available the file of an alternative coordinate linearization file can be specified (see command line parameters *lincrdcaldb* and *lincrdfile*). The above transformation gives the physical X and Y event coordinates measured in mm from the center of the FOV which will be used to derive "linearized" pixel coordinates. These will finally be stored as `DETX` and `DETY` table entries in the event extension of `_lev` files. Linearized coordinates run in each dimension from 0.0 to a maximal pixel value *Max* which depends on the actually chosen plate scale (see command line parameter lincrdplatescale). The point (0.0/0.0) is defined to refer to the corner in the focal plane with physical coordinates (0mm/0mm) and the point (*Max*/*Max*) represents the diagonally opposite corner at (32.79mm/32.79mm). Using the "intrinsic" plate scale of 7.8 pixels/mm (256x256 raw pixels on an active window area of 32.8x32.8 mm) the dimension of the linearized coordinate space will be [0.0-255.0] x [0.0-255.0]. Another example: A plate scale of 30 pixels/mm leads to an pixel range in X and Y of [0.-30*32.79] = [0.-984.]. The center of the FOV will become the central pixel in the linearized image, i.e., in the last example the center of the FOV will be assigned the coordinates (492.0/492.0).

DEAD-TIME CALCULATION In this processing step the instrumental dead-time correction factor *Fd*, i.e., 1 minus the fraction of the total observing time during which the instrument was insensitive to incoming X-rays, is calculated. The actual computation is carried out using the instrument's deadtime ratemeter counter which is tabulated in the HK file as a standard parameter with a 1s time granularity. The corresponding Bintable column name is `DEADTIME_RATEMETER`, see [2]. The value of *Fd* is then simply Fd = 1 - M(DEADTIME_RATEMETER)

where *M(DEADTIME_RATEMETER)* is the value of the dead-time ratemeter averaged over the time period covered by the observation. *Fd* will be stored in the event extension of the output file as value of a new header keyword `DEADC`. In addition, the standard deviation of *Fd* and the dead-time corrected observation time will be calculated (keyword names `STDDEADC` and `LIVETIME`/ `EXPOSURE`). The latter quantity is needed to allow the true X-ray source strength and variance to be determined.

DETECTOR TO SKY COORDINATE MAPPING In this stage the program calculates for each event the pixel position within a two-dimensional sky image and saves this information in the `X` and `Y` table columns of the event extension. In addition, standard World-Coordinate-System (WCS) header keywords (see [2] and [7] for details) are added to the event extension header which enables corresponding FITS readers such as *Ximage* or the standard FTOOLS task *xy2sky* to compute for each event the actual R.A. and Dec. value. The generated image is a standard tangent plane projection of the sky with a X and Y pixel range corresponding to the dimensions of the linearized detector image. Other than in this, however, X and Y pixel values start at 1. The central pixel of the sky image shall be the projection's fixed-point, i.e., the point where the tangent plane touches the celestial sphere. Via a command line parameter *skypointing*) it can be chosen among the following three alternatives:

1. the current nominal pointing direction of the satellite
2. the mean pointing of the satellite during the observation
3. any fixed point in the sky given as a user-supplied R.A./Dec. coordinate pair

In the first case, the celestial coordinates of the reference point are taken as the values of the `RA_OBJ` and `DEC_OBJ` keywords from the primary FITS header. In the second case, they are calculated as the mean R.A. and Dec. values of the satellite's pointing axis from the attitude data in the HK file corresponding to the current event file. In case an arbitrary reference point is requested, the coordinates are specified via command line parameters *skyranom* and *skydecnom*.

Once the sky image reference point is known, the calculation of each event's sky coordinates is carried out in the following manner:

1. The satellite's attitude at the photon arrival time is determined. This is normally done by reading the attitude data from the Aspect extension in the HK file corresponding to the current event list file. Those data have a time granularity of 0.5s and the satellite's attitude at the requested time is calculated using a cubic-spline interpolation. It is possible to dispense with using the actual attitude data from the HK file and assume a fixed spacecraft attitude instead (see command line parameter *skyfixedatt*). This must then be given as set of 3-1-3 Euler angles in the celestial, earth-centered reference frame.
2. The actual pointing direction is corrected for the effect of abberation and the current satellite roll angle is calculated.
3. The event is projected onto the tangent plane using its linearized focal-plane coordinates and the quantities obtained in step 2. Hereby, the effect of a potential boresight-misalignment,

is taken into account. This is done by means of a 3x3 rotation matrix to align the LECS's optical axis to the satellite's Z-axis, i.e., the current pointing direction. Per default the boresight-misalignment matrix is attempted to read from a corresponding file in the calibration database. This can be suppressed by specifying an alternative file or by forcing the program to use a hard-coded default matrix (see command line parameters *opticscaldb* and *opticscalfile*).

The final sky image is north-oriented, i.e., the top of the image points towards the celestial north pole.

PARAMETERS

The following command line parameters are recognized by the program:

infiles **= "" [string: list of existing event list files]**
A blank or colon separated list of event files (of type `_rev`, or `_lev`) to be processed. Note that if blank is used as file name separator, the list has to be enclosed by "" in order to be considered as a single command line argument. If a file in *infiles* does not exist or is found to be a valid LECS event file a warning message is issued and the processing continues with the next file in the list.

indirpath **= "" [string: list of directory names]**
A colon separated list of directories to seek for the specified input files. For each file given in *infiles* this is performed by first attempting to open the file as specified in *infiles*. If this fails, each directory component in *indirpath* is inspected for the location of the desired input file.

outdir **= "=" [string: name of existing directory or "="]**
The destination directory for the FITS output file. If "=" is specified for *outdir* (the default), the output file is written to the directory where the corresponding input file was found. Note that write access for *outdir* is needed.

clobber **= no [boolean: yes—no]**
In case of a raw event input file the name of the output file is simply derived by changing `rev` to `lev`, e.g. `x_rev.fits` becomes `x_lev.fits`. If a `_lev` file is to be processed the name is retained causing a name conflict. *clobber* = yes in this situation triggers that the input file will get overwritten by the new output file. If *clobber* is set to "no" an input file `x_lev.fits` will be renamed to `x_lev.old.fits` prior to creating the output file. In case a file `x_lev.old.fits` exists already from a previous run it will be lost.

rmevents **= no [boolean: yes—no]**
A boolean switch determining whether the events removal stage shall be carried out.

fovradius **= 0. [real: >=0.]**
The radius in raw pixels of the circular region defining the detector's FOV. If it is given as 0. (the default), the actual value will be obtained from the LECS-optics geometry calibration file ([3]). The value will be close to 100 pixels.

tgain **= yes [boolean: yes—no]**
A boolean switch determining whether the temporal gain correction shall be carried out.

tgaincol **= "GAIN_AVG" [string: valid HK parameter column name]**
The name of the Bintable column in the HK file (type: `hkg`) containing the time-dependent gain values to be used for the temporal gain correction.

tgainint **= ”LINEAR” [string: ”LINEAR”—”SPLINE”]**
: This determines the gain interpolation method to be used. ”LINEAR” selects a linear regression while ”SPLINE” signifies that the available gain values should be interpolated using a cubic spline.

tgainplot **= no [boolean: yes—no]**
: If set to ”yes”, a plot showing the interpolating gain function is produced and the program asks for acceptance before proceeding with the actual correction.

sgain **= yes [boolean: yes—no]**
: A boolean switch determining whether the spatial gain correction shall be carried out.

sgaincaldb **= yes [boolean: yes—no]**
: If set to ”yes”, the spatial gain map file used for performing the spatial gain correction is located by querying the SAX calibration database. In case this is not desired, *sgaincaldb* = no signals that a default map file should be used instead.

sgainfile **= ”” [string: name of existing gain map file]**
: The name of a spatial gain map file to be used for performing the spatial gain correction. Only in effect if *sgaincaldb* is set to ”no”.

sgainfactor **= 1 [real: 1-100]**
: A multiplicative constant used in the spatial energy gain correction.

blvsel **= yes [boolean: yes—no]**
: A boolean switch determining whether the standard event selection shall be carried out.

blvseldovetocut **= yes [boolean: yes—no]**
: If set to ”no” the Veto selection part in the combined Veto/PI-BL selection is not done.

blvselfract **= 0.9 [real: 0.5-1.0]**
: This is the fraction of photons that shall be retained in the standard Veto/PI-BL event selection at all energies. Default is to retain 90% of the genuine X-ray events.

blvsele2picaldb **= yes [boolean: yes—no]**
: If set to ”yes”, the keV->PI conversion file needed for the event selection is located by querying the SAX calibration database. If this is not desired or the calibration database is not available, *blvsele2picaldb* = no signals that a hard-coded default relation shall be used in converting energies on keV to PI values.

blvsele2pifile **= ”” [string: name of existing keV->PI conversion file]**
: The name of a FITS file from which the keV->PI conversion coefficients shall be read. Only in effect if *blvsele2picaldb* is set to ”no”.

blvselcaldb **= yes [boolean: yes—no]**
: If set to ”yes”, the BL-spectra calibration file needed for the event selection is located by querying the SAX calibration database. If this is not desired or the calibration database is not available an alternative file has to be specified via the command line parameter *blvselcalfile*

blvselcalfile **= ”” [string: name of existing BL-spectra calibration file]**
: The name of a FITS file containing preprocessed BL-spectra. Such a file is needed for the standard event selection.

lincrd **= yes [boolean: yes—no]**
: A boolean switch determining whether the coordinate linearization shall be carried out.

lincrdplatescale **= ”0.” [string: number followed by ”m” or ”s”]**

This defines the actual plate scale for the coordinate linearization, i.e., the number of linearized pixels per mm on the focal plane. This parameter is supposed to be given as a number optionally followed by either of the letters ”m” or ”s”. In the first case or without any suffix, the numeric value shall be interpreted a pixel-per-mm value, otherwise as arcsec/pixel. If 0 is assigned to this parameter the program will use the intrinsic value 7.8 pixels/mm (256 raw pixels on 32.8mm active entrance window length) which corresponds to a pixel size of 17.4”. This will lead to a linearized pixel range of [0.0-255.0] in both directions (see above) and the X/Y sky image coordinates shall run from [1.0-256].

lincrdcaldb **= yes [boolean: yes—no]**

If set to yes, the file containing the coefficients needed for the coordinate linearization is located by querying the SAX calibration database. In case this is not desired, *lincrdcaldb* = no signals that a set of default coefficients shall be used instead.

lincrdfile **= ”” [string: name of existing FITS file]**

The name of a FITS file from which the coordinate linearization coefficients ought to be read. Only in effect if *lincrdcaldb* is set to ”no”.

deadtime **= yes [boolean: yes—no]**

A boolean switch determining whether the dead-time calculation shall be carried out.

sky **= yes [boolean: yes—no]**

A boolean switch determining whether the detector-to-sky coordinate mapping shall be carried out.

skyfixedatt **= no [[boolean: yes—no]**

A boolean switch determining whether the spacecraft’s attitude shall be assumed to be fixed during the observation. If set to ”yes” the actual attitude must be specified as set of three 3-1-3 Euler angles in the celestial, earth-centered reference frame via command line parameters *skyphi*, *skytheta*, and *skyphi*.

skyphi **= 0. [real: 0.-360.]**

The first 3-1-3 Euler angle to specify the satellite’s attitude if *skyfixedatt* is set to ”yes”.

skytheta **= 0. [real: 0.-180.]**

The second 3-1-3 Euler angle to specify the satellite’s attitude if *skyfixedatt* is set to ”yes”.

skypsi **= 0. [real: 0.-360.]**

The third 3-1-3 Euler angle to specify the satellite’s attitude if *skyfixedatt* is set to ”yes”.

skypointing **= ”KEYWORD” [string: ”KEYWORD”—”MEANATT”—”USER”]**

This determines the method by which the sky image reference point with the defined coordinates (128.5/128.5) will be chosen. ”KEYWORD” signifies that the celestial coordinates of the reference point shall be taken as the values of the header keywords `RA_OBJ` and `DEC_OBJ` in the event file’s primary header. ”MEANATT” means that the coordinates will be computed as the mean values of R.A. and Dec. of the spacecraft’s pointing axis during the observation. Finally, if ”USER” is given as value of *skypointing* the celestial coordinates of the image reference point must be passed to the program via command line parameters *skyranom* and *skydecnom* respectively.

skyranom **= 0. [real: 0.-360.]**

The value of the R.A. of the sky image reference point if *skypointing* was chosen as ”USER”.

skydecnom **= 0. [real: -90.-90.]**
The value of the Dec. of the sky image reference point if *skypointing* was chosen as ”USER”.

opticscaldb **= yes [boolean: yes—no]**
If set to ”yes” *evelin* attempts to locate a LECS-optics geometry calibration file through querying the calibration database. The calibration file contains:

1. 3x3 bore-sight misalignment matrix needed for the creation of sky images
2. detector rotation angle referred to the spacecraft’s X/Y reference frame
3. focal plane coordinates of the center of the FOV
4. radius of the circle defining the FOV

Setting *opticscaldb* to ”no” signals not to query the database for a LECS-optics geometry file but, unless an alternative file is specified via the *opticsfile* parameter, to use hard-coded default values for those quantities.

opticsfile **= ”” [string: name of existing FITS file]**
The name of an alternative LECS-optics geometry calibration file to use. Only in effect if *opticscaldb* is set to ”no”.

caldb **= yes [boolean: yes—no]**
If set to ”yes” all calibration files needed by the various stages of *evelin* will be attempted to locate by querying the SAX/LECS calibration database. This requires that

1. the SAX/LECS calibration database is accessible from the machine on which *evelin* is running
2. the standard FTOOLS task *quzcif* is available
3. the environment to use the calibration database has been properly set up (see general HEASARC calibration database documents)

Setting *caldb* equal to ”yes” is equivalent to setting all of the above command line parameters ending in *...caldb* equal to yes.

verbosity **= 2 [integer: 0—1—2—3]**
Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means ”mute”, only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 2.

plotdev **= ”/XWIN” [string]**
The default PGPLOT device. All devices supported by the FTOOLS PGPLOT library can be specified.

help **= no [boolean: yes—no]**
A boolean flag which, when set to *yes* triggers that the program prints out a brief on-screen description of all available command line parameters together with the current program version number and exits successfully.

EXAMPLES

1. Perform event linearization of the two raw event files `kv183_rev.fits` and `kv550_rev.fits`. Suppress the computation of sky coordinates, use cubic-spline interpolation for temporal gain values and show the resulting function before proceeding: evelin ”kv183_rev.fits kv550_rev.fits” sky=no tgainint=spline tgainplot=yes

2. Perform event linearization of the file `xxx_lev.fits` except for the calculation of the total dead time. The input file shall be sought under `/data/sax1/` and `/data/sax2/` and program shall give maximum diagnostic messages as it proceeds through the various stages: evelin xxx_lev.fits indir=/data/sax1:/data/sax2 verb=3 deadtime=no

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010
2. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM,U. Lammers and M. Busetta, Doc. SAX/LEDA/0002
3. SAX/LEGSPC CALIBRATION FILES: FORMATS AND CONTENTS,M. Busetta, F. Favata, G. Vacanti, U. Lammers, Doc. SAX/LEDA/0003
4. THE GAIN MAPS OF THE SAX/LEGSPC: FUNCTIONAL TESTS AND PANTER CALIBRATION,G. Vacanti, Doc. SAX/LEDA/0005
5. THE PLATE SCALE OF THE SAX/LEGSPC,G. Vacanti, Doc. SAX/LEDA/0007
6. LECS EVENT SELECTION,G. Vacanti, Doc. SAX/LEDA/0012
7. REPRESENTATIONS OF CELESTIAL COORDINATES IN FITS, E.W. Greisen and M. Calabretta, A&A, 1995

BUGS

- Sky image creation not yet well tested.

SEE ALSO

egseing, *foting*, *calcgain*

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.9 Convert photon arrival times: *baryconv*

NAME

baryconv - perform BARYcentric time CONVersion of photon arrival times

SYNOPSIS

baryconv [**infiles=**] <**file_names**> [**parameter=value** ...]

DESCRIPTION

The *baryconv* task converts photon arrival times from the local satellite frame to Barycentric Dynamical Time (TDB) whose spatial origin lies at the solar system barycenter. In more detail, the program expects a list of names of linearized SAX/LECS event list files (type `_lev`) on the command line, and, for each valid file in this list, the following actions are carried out:

1. Check whether the barycentric conversion has been performed already (based on value of header keyword `TIMEREF`) and, if found true, exit
2. Move to event list extension and correct all event time tags stored in the `TIME` Bintable column
3. If a GTI extension exists, convert all interval start and stop times to TDB as well
4. Update the values of the keywords `TIMEREF`, `TSTART`, `TSTOP`, and `TELAPSE` in all FITS headers

In accordance with NASA/OGIP recommendations, the value of the `TIMEREF` keyword is either `LOCAL` or `SOLARSYSTEM` depending on whether the times are expressed in local satellite time or TDB. Note that the conversion to TDB does not affect the actual *time system* in use, i.e., all times keep being referred to the fixed mission reference time 01/01/1994 00:00:00 UTC.

The conversion of an event's arrival time to TDB requires the knowledge of the satellite's position at that time. Therefore, the program attempts to open the HK file of type `_hkg` corresponding to the event list file currently being processed and to get the satellite position data from the ephemeris extension. Per default, the name of the HK file is derived from the event list file name by simply substituting `_lev` with `_hkg`. However, there is also the possibility to specify an alternative HK file name via a command line parameter hkfile.

In addition to the satellite position data, the time conversion algorithm in use by *baryconv* requires the barycentric vectors for the time period in question to be known a priori. This information is comprised in an ASCII file called `earth.dat` which is part of the standard FTOOLS reference data set (see command line parameter earthfile).

PARAMETERS

The following command line parameters are recognized by the program:

infiles **= ”” [string: list of event file names]**
This is a blank or colon separated list of event files of type ”_lev” to be processed. Note that if blank is used as file name separator, the entire list has to be enclosed by double-quotes for being recognized as single command line argument.

indirpath **= "" [string: list of directory names]**

A colon separated list of directories to seek for the specified input files. For each file given in *infiles* this is performed by first attempting to open the file as specified in *infiles*. If this fails, each directory component in *indirpath* is inspected for the location of the desired input file.

hkfile **= "=" [string: name of SAX/LECS HK file (type** `_hkg`**)]**

The name of the HK file of type `_hkg` corresponding to the current event list file that is being processed. This HK file is supposed to contain ephemeris data for the time period covered by the event arrival times to be converted to TDB. If *hkfile* is given no value or is set to "=", the actual name is derived from the event list file name by simply substituting `_lev` with `_hkg`.

earthfile **= "" [string: full path name to file** `earth.dat` **]**

The path and name of the file `earth.dat` containing the barycentric vectors for various Julian days. This file is part of the standard FTOOLS reference data set. If *earthfile* is assigned no value, the file will be accessed by *baryconv* as which has an installation dependant default value.

ra **= 0.0 [real: 0 <=** *ra* **<= 360**

dec **= 0.0 [real: -90 <=** *dec* **<= 90]**

The Right Ascension and Declination in decimal degrees of the point in the celestial sky which shall be taken as origin of the photons whose arrival times are to be converted to TDB. If both *ra* and *dec* are specified as 0, the actual values are taken from the FITS header keywords `RA_OBJ` and `DEC_OBJ` respectively. These keywords contain the celestial coordinates of the object the satellite is currently pointing at.

verbosity **= 2 [integer: 0—1—2—3]**

Specifies the program verbosity level, i.e. the amount of screen output that will be issued during program execution. Level 0 means "mute", only fatal error messages will be printed. Verbosity level 3 results into detailed diagnostic messages. Default is level 2.

help **= no [boolean: yes—no]**

A boolean flag which, when set to *yes* triggers that the program prints out a brief on-screen description of all available command line parameters together with the current program version number and exits successfully.

EXAMPLES

1. Process the file `kv618_lev.fits` assuming a source position of R.A. = 23.4 deg, Dec. = 88.9 deg. The file `earth.dat` is contained in the current directory; the name of the HK file corresponding to the event file will be implicitly taken as `kv618_hkg.fits`: baryconv kv618_lev.fits ra=23.4 dec=88.9 earthfile=./earth.dat

2. Process the two files `a_lev.fits` and `b_lev.fits` - the source positions shall be obtained from the respective FITS headers and the `earth.dat` file is found in a non-standard location: baryconv "a_lev.fits b_lev.fits" earthfile=/usr/local/data/earth.dat

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

2. CONTENTS AND STRUCTURES OF EVENT AND HK FILES IN THE SAX/LEGSPC DATA ANALYSIS SYSTEM,U. Lammers and M. Busetta, Doc. SAX/LEDA/0002

BUGS

None known.

SEE ALSO

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.3.10 Creating response matrices: *lemat*

NAME

lemat - the SAX/LECS Matrix Generation Program

SYNOPSIS

lemat [**infile=**] <**filename**> [**parameter=value** ...]

DESCRIPTION

lemat is the program to generate response matrices for the SAX LEGSPC. It needs as input a spectral file created by XSELECT. The output from LEMAT is the area response file (ARF) and redistribution matrix file (RMF) appropriate to the input. These three files can then be read into XSPEC for spectral analysis. The program logic is as follows:

1. Read Input parameters from command line or parameter file.
2. Open and read input PHA File.
3. Check PHA file structure and contents compatible with program.
4. Read and decode WMAP (weighted image map) primary array data.
5. Calculate source position centroid
6. Populate Energy Boundary Data Arrays
7. Get Calibration Data from CALDB
   - > Gain for BL corrected and non-corrected cases
   - > Energy Resolution for BL corrected and non-corrected cases
8. Get effective area data.
   - > Mirror efficiency through ray-tracing
   - > Strongback and grid Transmission through raytracing
   - > Window transmission
   - > protection window transmission
   - > Detector PSF losses through ray-tracing
   - > detector absorption efficiency
9. Write RMF output file
   - > matrix extension
   - > ebounds extension
10. Write ARF output file

The current calibration status is as follows:

Item	Status	Complete
Window Transmission	Bessy measurements	Yes
Prot. Window Transmission	Bessy measurements	Yes
Gain Curve	Panter measurements	Yes
Escape Data	Bessy/Panter measurements	Yes
Resolution shape	Measured shape	Yes
Resolution E dependence	Measured dependence	Yes
Mirror Area	From ray tracing	Yes
Detector PSF losses	From ray tracing	Yes
Grid and strongback trans	From ray tracing	Yes
Detector Efficiency	Theoretical	Yes

The following files types need to be in the CALDB in order for LEMAT to work:

Name	Contents
WTRANS	Window area densities for 5 elements
WATOM	Window mass absorption coefficients for 5 elements
WINFUD	Window fudge factors for near edges
XEATOM	Xenon absorption coefficients
FTRANS	Protection window transmission measurements
LINESHAPE	KeV to PI (and PIC), FWHM, etc relations

PARAMETERS

The following command line parameters are recognized by the program:

infile **= "" [string]**
: This is the input FITS spectral file for which the response is to be generated. The file should be in accordance with the format described in 'The OGIP Spectral File Format', OGIP/92.007.

outfile1 **= "INFILE.RMF" [string]**
: This is the output redistribution matrix file (RMF) that will be produced. If it is not specified, a file with the same name as the input name but with extension .RMF will be created. The format of the file is described in 'The Calibration Requirements for Spectral Analysis', OGIP calibration memo CAL/GEN/92-002.

outfile2 **= "INFILE.ARF" [string]**
: This is the output area response file (ARF) that will be produced. If it is not specified, a file with the same name as the input name but with extension .ARF will be created. The format of the file is described in 'The Calibration Requirements for Spectral Analysis', OGIP calibration memo CAL/GEN/92-002.

chatter **= 5 [integer: 0-10]**
: Specifies the program chattiness level, i.e. the amount of screen output that will be issued during program execution. Level 0 means "mute" except for error messages 1 will output filenames only, 2 will add input spectrum details, 3 will include processing details, 5 is the default, 6 includes overwritten file info, and 10 is too verbose for most.

maxsep **= 5.0 [real: 0.0-100.]**
The maximum separation between the source position determined by centroiding and the pixel where the peak counts occur.

point **= yes [boolean: yes—no]**
Boolean switch to trigger point source or extended source analysis. The latter is not yet enabled.

Xposition **= [real: 0.0-255.]**
This hidden parameter can be used to override the source X position obtained by assuming that the source is located at the location given by the peak counts. The units are raw pixels. The parameter is only overriden if point=no

Yposition **= [real: 0.0-255.]**
This hidden parameter can be used to override the source Y position obtained by assuming that the source is located at the location given by the peak counts. The units are raw pixels. The parameter is only overriden if point=no

clobber **= yes [boolean: yes—no]**
This hidden parameter can be used to enable or disable overwriting of already existing files.

seed **= 0 [integer: 0-65535]**
This hidden parameter can be used to specify a particular seed value for the random number generator used in the ray tracing algorithm. If it is set to 0, the system clock is used to calculate a seed.

smooth **= 100.0 [real: 0.0-1000.0]**
This hidden parameter is used to control the amount of smoothing applied to the output of the ray-tracing algorithm. Please leave at the default value unless you are an expert.

nphotons **= 5000 [integer: 500-1000000]**
This hidden parameter is the number of photons exiting from the Mirror Units at each energy in the ray tracing. Please leave at the default value unless you are an expert.

nrayen **= 41 [integer: 5-1025]**
This hidden parameter is the number of energies for which the ray tracing will be performed. Please leave at the default value unless you are an expert.

plot_arf **= no [boolean: yes—no]**
This hidden paramater can be used to plot the various contributions to the ARF using QDP/PLT for diagnostic purposes. After plotting the program remains with the PLT> prompt. Type 'exit' to complete the LEMAT run.

plot_rmf **= 0.0 [real: 0.0-15.0]**
This hidden paramater can be used to plot the Response of the LECS to monochromatic X-rays with energy =plot_rmf using QDP/PLT. This is primarily useful for diagnostic purposes. After plotting the program remains with the PLT> prompt. Type 'exit' to complete the LEMAT run.

plotdev **= "/xwin" [string]**
This hidden parameter can be used to set the PGPLOT device used with the plot_arf parameter. All devices supported by the FTOOLS PGPLOT library can be specified.

pltcmd **= " " [string]**
This parameter represents any legitimate PLT command that is to be added to the standard

QDP commands used in plotting either the ARF or RMF. Pltcmd can also be of the form @filename in which case the commands within file filename.pco are added to the default commands.

origin **= "SSD/ESA" [string]**
This hidden parameter can be used to set the output FITS files "ORIGIN" parameter.

help **= no [boolean: yes—no]**
This hidden parameter can be used to produce a short listing of the input parameters for LEMAT.

EXAMPLES

1. To create RMF and ARF files for an input file 'herx1.pha', created using XSELECT using all default parameters. The output files will be automatically named 'herx1.rmf' and 'herx1.arf', type: "lemat herx1.pha"
2. To create RMF and ARF files for an input file 'herx1.pha' for a position in the FOV of RAWX,RAWY = 130,142, type: "lemat herx1.pha point=no xpos=130. ypos=142."
3. To create RMF and ARF files for an input file 'herx1.pha' but using 100 raytracing energies and 2000 photons exiting the mirrors at each energy (this will take some time....) type: "lemat herx1.pha nphotons=2000 nrayen=100"
4. To create RMF and ARF files for an input file 'herx1.pha' and plot the LECS spectral resolution at 5.3 keV type: "lemat herx1.pha plot_rmf=5.3"

FURTHER DOCUMENTATION

1. XSPEC User Manual
2. XSELECT User Manual

SEE ALSO

calcgain, *evelin*, xspec, xselect

A.N. Parmar (Arvind.Parmar@astro.estec.esa.nl)

3.3.11 Creating final product files: *lecspipe*

NAME

lecspipe - run the LECS data analysis PIPEline

SYNOPSIS

lecspipe [**firstevent**=] <**rev_filename**> [**parameter=value** ...]

DESCRIPTION

lecspipe realizes the four-stage LEGSPC data analysis pipeline as described in Ref. [1]. It is intended to be used to create final product files from raw (event list) FITS files in a non-interactive, standardized manner. The entry and exit stage at which pipeline processing shall start and end respectively can be determined via command line parameters (*entrystage* and *exitstage*). Per default, all four stages are executed. An initial raw event list file must be specified on invocation of *lecspipe*. If this file exists and all four pipeline stages shall be executed the following sequential steps are performed:

1. The *compevfiles* task is invoked with the specified initial file to get a list of files that are suitable for concatenation.
2. Unless explicitely suppressed via a boolean command line switch (*concat* = "no") the *catobs* task in invoked with the file list obtained in step 1. This will create one raw event file and a corresponding HK file.
3. The pipeline stages I - IV are executed.

If pipeline processing is requested not to start with stage I, the initial event file name is just used to derive a unique name stem. This value will then be used to create the names of files that have to exist prior to run a particular pipeline stage. If during execution of any individual pipeline module a fatal error condition occurs, the pipeline processing is aborted. Per default all software activities will be logged in a log file, which can be inspected to control the success of a performed pipeline processing run.

PARAMETERS

The following command line parameters are recognized by the program:

firstevent **= "" [string: name of an existing raw event list file (wildcards allowed)]**
: The name of the initial raw event list file with which to start the pipeline. Actually, the file specified will be passed to the *compevfiles* task as value of the command line parameter *evfilename* to get a list of event list files that are to be concatenated with the *catobs* task. *firstevent* can be a regular expression (Tcl syntax) which must match the name of exactly one file in the current working directory. If more than one file is matched, an error message will be issued and the execution is aborted.

filenamestem **= "-" [string]**
: This string parameter will be passed to the *catobs* task as value of the command line parameter *stemoutfile*. It determines the file name stem of the files that will be created by *catobs*

and will be used for all future references to any files created in this *lecspipe* run. The stem of a file name is defined to be everything before any extension `_???.fits`, e.g., the name stem of `kv492_rev.fits` is just `kv492`.

entrystage **= 1 [integer: 1-4, must be <= *exitstage*]**

The entry point at which the non-interactive pipeline processing shall begin. Prior to actually start the data processing at the requested point, the existence of all necessary files to run stage *entrystage* is checked. If this check proves negative the execution of *lecsipe* is aborted. Execution is likewise aborted if the illegal condition *entrystage* > *exitstage* (see below) is met.

exitstage **= 4 [integer: 1-4, must be >= *entrystage*]**

The stage after which the non-interactive pipeline processing shall be stopped. If *exitstage* < *entrystage* execution is aborted.

log **= yes [boolean: yes—no]**

If set to *yes* all screen I/O will be logged to a file. The name of this log file is controlled via the *logfile* parameter (see below).

logfile **= ”-” [string: a valid file name or ”-”]**

The name of the logfile if logging is enabled with *log* = *yes* (see above). *logfile* must be a valid file name or ”-”. The latter means that the logfile will get the name `xxx.log` where `xxx` is the stem of the initial event list file name (see description of the *filenamestem* parameter). If the specified log file cannot be created, an error message will be issued and *lecspipe* exits.

concat **= ”yes” [boolean: yes—no]**

This switch determines whether or not concatenation of compatible event list files is carried out as the first step of the pipeline processing. In the latter case the initial event list file *lecspipe* was started with is used as the primary file to be processed.

printout **= yes [boolean: yes—no]**

At the end of stage I a set of standard plots and images will be created in the form of Postscript files. This switch specifies whether or not these files shall be automatically send to a Postscript printer. The way this is done has actually to be specified via a command line parameter *psprintcmd* (see below).

psprintcmd **= ”lp %s” [string: command to print PS file]**

This specifies a command on UNIX shell-level to print PostScript files. The special token ”%s” serves as a substitute for the name of the file to be printed (up to three occurences of ”%s” are recognized in *psprintcmd*).

baryconv **= yes [boolean: yes—no]**

A boolean switch determining whether the *baryconv* task will be executed at the end of the pipeline stage III.

help **= no [boolean: yes—no]**

A boolean flag which, when set to *yes* triggers that the program prints out a brief on-screen description of all available command line parameters together with the current program version number and exits successfully.

EXAMPLES

1. Run the whole LECS data analysis pipeline starting with the event list file `kv492_rev.fits` and log all screen I/O in a file `figaro.log`. *baryconv* shall not be run after stage III: lecspipe

kv492_rev.fits logfile=figaro.log baryconv=no

2. Run stage II and III of the LECS data analysis pipeline starting with the event list file `kv492_rev.fits`:
   lecspipe kv492_rev.fits entrystage=2 exitstage=3

   All Screen I/O will be logged in the file `kv492.log`.

FURTHER DOCUMENTATION

1. THE SAX LEGSPC DATA ANALYSIS SYSTEM - SOFTWARE USER MANUAL, U. Lammers, Doc. SAX/LEDA/0010

BUGS

- Currently, merely stage I+III processing is supported (to be fixed soon).
- Not yet well tested.

SEE ALSO

catobs, *foting*, *egseing*

U. Lammers (Uwe.Lammers@astro.estec.esa.nl)

3.4 Useful FTOOLS tasks

to be continued ...

3.5 A sample SAXLEDAS session

to be continued ...

4 The SAXLEDAS FAQ list

4.1 Technical issues

4.1.1 ”How do I install the SAXLEDAS?”

to be continued ...

4.1.2 ”A certain pipeline module crashes, why?”

to be continued ...

4.2 Scientific issues

4.2.1 ”How do I create an image?”

to be continued ...

4.2.2 ”How do I create a lightcurve?”

to be continued ...

4.2.3 ”How do I create a spectrum?”

to be continued ...

A Sample FITS headers

A.1 Primary header

```
SIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                   32 / number of bits per data pixel
NAXIS   =                    0 / number of data axes
EXTEND  =                    T / FITS dataset may contain extensions
CONTENT = 'EVENT LIST'         / this is an event list file
REVISION=                    1 / revision number of processed data
ORIGIN  = 'ESA/SSD '           / processing site
DATE    = '19/03/96'           / FITS file creation date (DD/MM/YY)
CREATOR = 'foting v1.6.1'      / name of code that generated this file
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
OBSERVER= 'Giuseppe Settanni'  / name of observer or principal investigator
OBS_NO  =                    1 / observer ID sequence number
TAPE_ID = '000025  '           / FOT sequence number
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =   6.164496357785E+07 / data start time relative to MJDREF (s)
TSTOP   =   6.165244067493E+07 / data end time relative to MJDREF (s)
TELAPSE =   7.477097076416E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
N_OBIS  =                    1 / number of observation intervals (OBIs)
END
```

A.2 OBSTABLE Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   60 / width of table in bytes
NAXIS2  =                    1 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                    7 / number of fields in each row
EXTNAME = 'OBSTABLE'           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'TEMPORALDATA'       / these are temporal data ...
HDUCLAS2= 'OBSTABLE'           / ... forming the Obstable extension
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
TTYPE1  = 'OBI_NUM '           / Observation Interval Number
TFORM1  = 'I       '           / data format of the field: 2-byte INTEGER
TTYPE2  = 'OBIB    '           / start time of the OBI
TFORM2  = '9A      '           / data format of the field: ASCII character
TUNIT2  = 's       '           / SC time of OBI start (hex)
TTYPE3  = 'OBIE    '           / end time of the OBI
TFORM3  = '9A      '           / data format of the field: ASCII character
TUNIT3  = 's       '           / SC time of OBI end (hex)
TTYPE4  = 'START_DATE'         / UT date of OBI start (DD/MM/YY)
TFORM4  = '8A      '           / data format of the field: ASCII character
TTYPE5  = 'START_TIME'         / UT time of OBI start (HH:MM:SS)
TFORM5  = '12A     '           / data format of the field: ASCII character
TTYPE6  = 'STOP_DATE'          / UT date of OBI end (DD/MM/YY)
TFORM6  = '8A      '           / data format of the field: ASCII character
TTYPE7  = 'STOP_TIME'          / UT time of OBI end (HH:MM:SS)
TFORM7  = '12A     '           / data format of the field: ASCII character
END
```

A.3 RAW EVENTS Binary Table header

```
XTENSION= ’BINTABLE’           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   18 / width of table in bytes
NAXIS2  =               202819 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                    6 / number of fields in each row
EXTNAME = ’RAW EVENTS’         / name of this binary table extension
HDUCLASS= ’SAXLEDAS’           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = ’SAX/LEDA/0002’      / format specifications
HDUVERS = ’1.2.0   ’           / version of format
HDUCLAS1= ’EVENTS  ’           / event list data
HDUCLAS2= ’ALL     ’           / all events included
HDUCLAS3= ’RAW     ’           / raw data - no processing steps performed
TELESCOP= ’SAX     ’           / mission name
INSTRUME= ’LECS    ’           / instrument name
OBS_MODE= ’POINTING’           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = ’Capricorno’         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= ’FK5     ’           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = ’1       ’           / observation number in current observing period
DATAMODE= ’DIRECT  ’           / instrument operational mode
SUBMODE =                    1 / submode of DATAMODE
TIMESYS = ’1994 1 1 00:00:00’  / all times measured from this reference date
TIMEUNIT= ’s       ’           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = ’LOCAL   ’           / reference frame for reported time values
TASSIGN = ’SATELLITE’          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644963577850E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652440674927E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4770970764160E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= ’15/12/95’           / UTC date of observation start (DD/MM/YY)
TIME-OBS= ’11:36:01.000’       / UTC time of observation start (HH:MM:SS)
DATE-END= ’15/12/95’           / UTC date of observation end (DD/MM/YY)
TIME-END= ’13:41:00.000’       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= ’98F9.0000’          / spacecraft clock (s) at observation start
SCSEQEND= ’B644.0000’          / spacecraft clock (s) at observation end
ONTIME  =  7.4770970764160E+03 / total on-source time (s)
TIMEDEL =       .0000152500000 / time granularity of data (size of time bin)
INSTRACT=                    F / Is the instrument active?
GUPOW   =                    F / Is the GU powered?
HVPOW   =                    F / Is the HV on?
AEPOW   =                    F / Is the AE powered?
CELLON  =                    F / Is the gas cell on?
PMTON   =                    F / Is the PMT on?
CELLHV  =               -12000 / nominal value of cell HV
PMTHV   =                  800 / nominal value of PMT HV
LED     =                    F / Is LED on?
GETTER  =                    F / Is the getter active?
DIRECT  =                    F / Is the DM task active?
SPEC1   =                    F / Is the SPEC1 task active?
IMAGE1  =                    F / Is the IMAGE1 task active?
SPEC2   =                    F / Is the SPEC2 task active?
```

```
IMAGE2  =                    F / Is the IMAGE2 task active?
HANDLER =                    F / BL correction performed?
DM_CONT = '0x2FF   '           / the DM contents word
PAR_SEL =                    T / parameter selection (T if all present)
THR_SEL =                    T / threshold selection (T if applied) see below
BL_CORR =                    F / BL correction by DM task (T if applied)
T_FLAG  =                    T / event time flag (T if the parameter is present)
BL_FLAG =                    T / event BL flag (T if present)
X_FLAG  =                    T / event X flag (T if present)
Y_FLAG  =                    T / event Y flag (T if present)
V_FLAG  =                    T / event VETO flag (T if present)
FT_FLAG =                    T / event fine time flag (T if present)
BABT_FLA=                    T / event burst amplitude & base time flag
AE_EMIN =                    9 / most frequent AE discriminator value for Emin
AE_EMAX =                  253 / most frequent AE discriminator value for Emax
AE_XMIN =                    0 / most frequent AE discriminator value for Xmin
AE_XMAX =                  253 / most frequent AE discriminator value for Xmax
AE_YMIN =                    0 / most frequent AE discriminator value for Ymin
AE_YMAX =                  253 / most frequent AE discriminator value for Ymax
AE_VMIN =                    0 / most frequent AE discriminator value for Vmin
AE_VMAX =                  253 / most frequent AE discriminator value for Vmax
AE_BLMIN=                    0 / most frequent AE discriminator value for BLmin
AE_BLMAX=                  253 / most frequent AE discriminator value for BLmax
DM_EMIN =                    1 / DM discriminator value for Emin
DM_EMAX =                  255 / DM discriminator value for Emax
DM_XMIN =                    1 / DM discriminator value for Xmin
DM_XMAX =                  255 / DM discriminator value for Xmax
DM_YMIN =                    1 / DM discriminator value for Ymin
DM_YMAX =                  255 / DM discriminator value for Ymax
DM_VMIN =                    1 / DM discriminator value for Vmin
DM_VMAX =                  253 / DM discriminator value for Vmax
DM_BLMIN=                    1 / DM discriminator value for BLmin
DM_BLMAX=                  253 / DM discriminator value for BLmax
TTYPE1  = 'TIME    '           / event arrival time
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'PHA     '           / Pulse Height Amplitude
TFORM2  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT2  = 'chan    '           / physical unit of field
TLMIN2  =                    0 / minimum legal value for parameter 2
TLMAX2  =                 1023 / maximum legal value for parameter 2
TNULL2  =                   -1 / illegal value for column 2
TTYPE3  = 'RAWX    '           / event raw X-coordinate
TFORM3  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT3  = 'pixel   '           / physical unit of field
TLMIN3  =                    0 / minimum legal value for parameter 3
TLMAX3  =                  255 / maximum legal value for parameter 3
TNULL3  =                   -1 / illegal value for column 3
TTYPE4  = 'RAWY    '           / event raw Y-coordinate
TFORM4  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT4  = 'pixel   '           / physical unit of field
TLMIN4  =                    0 / minimum legal value for parameter 4
TLMAX4  =                  255 / maximum legal value for parameter 4
TNULL4  =                   -1 / illegal value for column 4
TTYPE5  = 'BL      '           / event burst length
TFORM5  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT5  = 'chan    '           / physical unit of field
TLMIN5  =                    0 / minimum legal value for parameter 5
TLMAX5  =                  255 / maximum legal value for parameter 5
TNULL5  =                   -1 / illegal value for column 5
TTYPE6  = 'VETO    '           / event VETO signal
```

```
TFORM6  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT6  = 'chan    '           / physical unit of field
TLMIN6  =                    0 / minimum legal value for parameter 6
TLMAX6  =                  255 / maximum legal value for parameter 6
TNULL6  =                   -1 / illegal value for column 6
END
```

A.4 EVENTS Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   34 / width of table in bytes
NAXIS2  =                    1 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   12 / number of fields in each row
EXTNAME = 'EVENTS  '           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'EVENTS  '           / event list data
HDUCLAS2= 'ALL     '           / all events included
HDUCLAS3= 'LINEARIZED'         / linearized data
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =  3.7600000000000E+01 / nominal RA of target (deg)
DEC_OBJ =  8.9300000000000E+01 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
DATAMODE= 'DIRECT  '           / instrument operational mode
SUBMODE =                    1 / submode of DATAMODE
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644963577850E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652440674927E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4770970764160E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
ONTIME  =  7.4770970764160E+03 / total on-source time (s)
TIMEDEL =  1.5250000000000E-05 / time granularity of data (size of time bin)
INSTRACT=                    F / Is the instrument active?
GUPOW   =                    F / Is the GU powered?
HVPOW   =                    F / Is the HV on?
AEPOW   =                    F / Is the AE powered?
CELLON  =                    F / Is the gas cell on?
PMTON   =                    F / Is the PMT on?
CELLHV  =               -12000 / nominal value of cell HV
PMTHV   =                  800 / nominal value of PMT HV
LED     =                    F / Is LED on?
GETTER  =                    F / Is the getter active?
DIRECT  =                    F / Is the DM task active?
SPEC1   =                    F / Is the SPEC1 task active?
IMAGE1  =                    F / Is the IMAGE1 task active?
SPEC2   =                    F / Is the SPEC2 task active?
```

```
IMAGE2  =                    F / Is the IMAGE2 task active?
HANDLER =                    F / BL correction performed?
DM_CONT = '0x2FF   '           / the DM contents word
PAR_SEL =                    T / parameter selection (T if all present)
THR_SEL =                    T / threshold selection (T if applied) see below
BL_CORR =                    F / BL correction by DM task (T if applied)
T_FLAG  =                    T / event time flag (T if the parameter is present)
BL_FLAG =                    T / event BL flag (T if present)
X_FLAG  =                    T / event X flag (T if present)
Y_FLAG  =                    T / event Y flag (T if present)
V_FLAG  =                    T / event VETO flag (T if present)
FT_FLAG =                    T / event fine time flag (T if present)
BABT_FLA=                    T / event burst amplitude & base time flag
AE_EMIN =                    9 / most frequent AE discriminator value for Emin
AE_EMAX =                  253 / most frequent AE discriminator value for Emax
AE_XMIN =                    0 / most frequent AE discriminator value for Xmin
AE_XMAX =                  253 / most frequent AE discriminator value for Xmax
AE_YMIN =                    0 / most frequent AE discriminator value for Ymin
AE_YMAX =                  253 / most frequent AE discriminator value for Ymax
AE_VMIN =                    0 / most frequent AE discriminator value for Vmin
AE_VMAX =                  253 / most frequent AE discriminator value for Vmax
AE_BLMIN=                    0 / most frequent AE discriminator value for BLmin
AE_BLMAX=                  253 / most frequent AE discriminator value for BLmax
DM_EMIN =                    1 / DM discriminator value for Emin
DM_EMAX =                  255 / DM discriminator value for Emax
DM_XMIN =                    1 / DM discriminator value for Xmin
DM_XMAX =                  255 / DM discriminator value for Xmax
DM_YMIN =                    1 / DM discriminator value for Ymin
DM_YMAX =                  255 / DM discriminator value for Ymax
DM_VMIN =                    1 / DM discriminator value for Vmin
DM_VMAX =                  253 / DM discriminator value for Vmax
DM_BLMIN=                    1 / DM discriminator value for BLmin
DM_BLMAX=                  253 / DM discriminator value for BLmax
TTYPE1  = 'TIME    '           / event arrival time
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'PHA     '           / Pulse Height Amplitude
TFORM2  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT2  = 'chan    '           / physical unit of field
TLMIN2  =                    0 / minimum legal value for parameter 2
TLMAX2  =                 1023 / maximum legal value for parameter 2
TNULL2  =                   -1 / illegal value for column 2
TTYPE3  = 'RAWX    '           / event raw X-coordinate
TFORM3  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT3  = 'pixel   '           / physical unit of field
TLMIN3  =                    0 / minimum legal value for parameter 3
TLMAX3  =                  255 / maximum legal value for parameter 3
TNULL3  =                   -1 / illegal value for column 3
TTYPE4  = 'RAWY    '           / event raw Y-coordinate
TFORM4  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT4  = 'pixel   '           / physical unit of field
TLMIN4  =                    0 / minimum legal value for parameter 4
TLMAX4  =                  255 / maximum legal value for parameter 4
TNULL4  =                   -1 / illegal value for column 4
TTYPE5  = 'BL      '           / event burst length
TFORM5  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT5  = 'chan    '           / physical unit of field
TLMIN5  =                    0 / minimum legal value for parameter 5
TLMAX5  =                  255 / maximum legal value for parameter 5
TNULL5  =                   -1 / illegal value for column 5
TTYPE6  = 'VETO    '           / event VETO signal
```

```
TFORM6  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT6  = 'chan    '           / physical unit of field
TLMIN6  =                    0 / minimum legal value for parameter 6
TLMAX6  =                  255 / maximum legal value for parameter 6
TNULL6  =                   -1 / illegal value for column 6
TTYPE7  = 'PI      '           / Pulse Invariant
TFORM7  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT7  = 'chan    '           / physical unit of field
TLMIN7  =                    0 / minimum legal value for parameter 7
TLMAX7  =                 1023 / maximum legal value for parameter 7
TNULL7  =                   -1 / illegal value for column 7
TTYPE8  = 'PIC     '           / Burst-length corrected PI
TFORM8  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT8  = 'chan    '           / physical unit of field
TLMIN8  =                    0 / minimum legal value for parameter 8
TLMAX8  =                 1023 / maximum legal value for parameter 8
TNULL8  =                   -1 / illegal value for column 8
TTYPE9  = 'DETX    '           / linearized photon X-coordinate
TFORM9  = 'E       '           / data format of the field: REAL
TUNIT9  = 'pixel   '           / physical unit of field
TLMIN9  =       .00000000E+00 / minimum legal value for parameter 9
TLMAX9  =      2.56000000E+02 / maximum legal value for parameter 9
TCRPX8  =    128.0000000000000 / detector X reference pixel
TCRVL8  =       .0000000000000 / detector X reference pixel value
TCDLT8  =       .1281000000000 / detector X pixel scale (mm/pixel)
OPTIC8  =    128.0000000000000 / opt. axis/fcocal-plane intersection point - X
TTYPE10 = 'DETY    '           / linearized photon Y-coordinate
TFORM10 = 'E       '           / data format of the field: REAL
TUNIT10 = 'pixel   '           / physical unit of field
TLMIN10 =       .00000000E+00 / minimum legal value for parameter 10
TLMAX10 =      2.56000000E+02 / maximum legal value for parameter 10
TCRPX9  =    128.0000000000000 / detector Y reference pixel
TCRVL9  =       .0000000000000 / detector Y reference pixel value
TCDLT9  =       .1281000000000 / detector Y pixel scale (mm/pixel)
OPTIC9  =    128.0000000000000 / opt. axis/fcocal-plane intersection point - Y
TTYPE11 = 'X       '           / projected X position of photon on sky
TFORM11 = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT11 = 'pixel   '           / physical unit of field
TNULL11 =                   -1 / illegal value for column 11
TTYPE12 = 'Y       '           / projected Y position of photon on sky
TFORM12 = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT12 = 'pixel   '           / physical unit of field
TNULL12 =                   -1 / illegal value for column 12
CORRMODE=                  255 / coded value for applied corrections
RMEVOFOV=                    T / events outside FOV removed?
TEMPGAIN=                    T / temporal gain correction performed?
SPATGAIN=                    T / spatial gain correction performed?
BLVTSELE=                    T / BL-Veto selection performed?
BLCORR  =                    T / BL correction performed?
LINCOORD=                    T / linearized detector coordinates?
DEADTIME=                    T / Dead-time correction performed?
SKYCOORD=                    T / sky coordinates created?
PHTFRAC =       .9900000000000 / fraction of photons retained in PI-BL selection
HISTORY
HISTORY   The following block of parameters refers to the boundaries used in
HISTORY   the performed Veto/PI-BL selection. If there are more than one
HISTORY   block, the last one reflects the current values.
HISTORY
HISTORY      Veto selection interval        : [30, 65]
HISTORY      Number of PI-BL selection boxes: 11
HISTORY      Boxes' coordinates             : #     PI-Int       BL-Int
```

```
HISTORY                                           1 [0000, 0022], [121, 249]
HISTORY                                           2 [0023, 0038], [137, 247]
HISTORY                                           3 [0039, 0068], [145, 236]
HISTORY                                           4 [0069, 0112], [147, 223]
HISTORY                                           5 [0113, 0150], [148, 214]
HISTORY                                           6 [0151, 0222], [150, 212]
HISTORY                                           7 [0223, 0357], [144, 200]
HISTORY                                           8 [0358, 0462], [123, 196]
HISTORY                                           9 [0463, 0539], [147, 194]
HISTORY                                          10 [0540, 0660], [142, 187]
HISTORY                                          11 [0661, 1023], [124, 194]
HISTORY
DEADC   =  9.9061901993741E-01 / dead-time correction factor = LIVETIME/ONTIME
STDDEADC=  4.6397107548218E-03 / standard deviation of DEADC
LIVETIME=  7.4069545778161E+03 / dead-time corrected total on-source time (s)
EXPOSURE=  7.4069545778161E+03 / fully corrected total on-source time (s)
TCTYP11 = 'RA---TAN'           / coordinate projection: TAN
TCRPX11 =         129.00000000 / sky X axis reference pixel
TCRVL11 =     37.6000000000000 / sky X coordinate at reference pixel (degrees)
TCDLT11 =      -.0048145265888 / sky X pixel scale (deg/pixel) = -17.3 "/pixel
TLMIN11 =                    1 / first X pixel in aspected sky image
TLMAX11 =                  257 / last X pixel in aspected sky image
OPTIC10 =                   -1 / opt. axis/focal-plane intersection point  - X
TCTYP12 = 'DEC--TAN'           / coordinate projection: TAN
TCRPX12 =         129.00000000 / sky Y axis reference pixel
TCRVL12 =     89.3000000000000 / sky Y coordinate at reference pixel (degrees)
TCDLT12 =       .0048145265888 / sky Y pixel scale (deg/pixel) = 17.3 "/pixel
TLMIN12 =                    1 / first Y pixel in aspected sky image
TLMAX12 =                  257 / last Y pixel in aspected sky image
OPTIC11 =                   -1 / opt. axis/focal-plane intersection point  - Y
RA_PNT  =  3.7720098134859E+01 / average right ascension of S/C Z-axis
SDDRAPNT=  1.3711934520376E+00 / standard deviation of RA_PNT
DEC_PNT =  8.9269861574389E+01 / average declination of S/C Z-axis
SDDDCPNT=  1.0049682206466E-03 / standard deviation of DEC_PNT
ROLLANG = -5.4712638339913E+01 / average S/C roll angle
SDDROLL =  1.3663981856788E+00 / standard deviation of ROLLANG
END
```

A.5 GTI Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   16 / width of table in bytes
NAXIS2  =                    1 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                    2 / number of fields in each row
EXTNAME = 'ALLGTI  '           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'GTI     '           / good time intervals
HDUCLAS2= 'ALL     '           / standard GTI table
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =  1.0000000000000E+00 / nominal RA of target (deg)
DEC_OBJ =   .0000000000000E+00 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644963577850E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652440674927E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4770970764160E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
ONTIME  =  7.4770970764160E+03 / total on-source time (s)
TTYPE1  = 'START   '           / label for field 1
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'STOP    '           / label for field 2
TFORM2  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT2  = 's       '           / physical unit of field
END
```

A.6 RAW HKP Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   93 / width of table in bytes
NAXIS2  =                 7175 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   55 / number of fields in each row
EXTNAME = 'RAW HKP '           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'TEMPORALDATA'       / non-science quantities versus time
HDUCLAS2= 'HKP     '           / housekeeping parameters
HDUCLAS3= 'FULL    '           / non-compressed
HDUCLAS4= 'ENGINEERING'        / parameters in engineering units
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644971000000E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652438000000E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4670000000000E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
TIMEDEL =      1.0000000000000 / time granularity of data (size of time bin)
SCALFLAG=                    F / HK parameter values scaled to physical units?
TTYPE1  = 'TIME    '           / time tag (since MJDREF)
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'GETT_ENBL_SW'       / Getter Enable Software flag (T if enabled)
TFORM2  = 'L       '           / data format of the field: LOGICAL
TTYPE3  = 'MEM_MOD_ENBL'       / Memory Modify Software flag (T if enabled)
TFORM3  = 'L       '           / data format of the field: LOGICAL
TTYPE4  = 'GETT_ENBL'          / Getter Enable before RS04 (T if enabled)
TFORM4  = 'L       '           / data format of the field: LOGICAL
TTYPE5  = 'SHUT_CLOSE'         / Shutter Closed Position (T if closed)
TFORM5  = 'L       '           / data format of the field: LOGICAL
TTYPE6  = 'SHUT_OPEN'          / Shutter Open Position (T if open)
TFORM6  = 'L       '           / data format of the field: LOGICAL
TTYPE7  = 'PMT_CURR'           / PMT Current Flag (T if set)
TFORM7  = 'L       '           / data format of the field: LOGICAL
```

```
TTYPE8  = 'GU_SAFETY'           / Gas Unity Safety flag (T if set)
TFORM8  = 'L       '            / data format of the field: LOGICAL
TTYPE9  = 'VALVE_SUPPLY'        / Valve Supply after RS05 (T if enabled)
TFORM9  = 'L       '            / data format of the field: LOGICAL
TTYPE10 = 'GETT_SUPPLY'         / Getter Supply after RS04 (T if enabled)
TFORM10 = 'L       '            / data format of the field: LOGICAL
TTYPE11 = 'HV_POWER'            / HV Power Supply after RS03 (T if enabled)
TFORM11 = 'L       '            / data format of the field: LOGICAL
TTYPE12 = 'HV_ENBL '            / HV Enable status before RS03 (T if enabled)
TFORM12 = 'L       '            / data format of the field: LOGICAL
TTYPE13 = 'FEE_-15V'            / FEE -15V power after RS01 (T if enabled)
TFORM13 = 'L       '            / data format of the field: LOGICAL
TTYPE14 = 'FEE_+15V'            / FEE +15V power after RS01 (T if enabled)
TFORM14 = 'L       '            / data format of the field: LOGICAL
TTYPE15 = 'ST_EQ   '            / Self Test Errors (0xf: > 14 errors occured)
TFORM15 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN15 =                     0 / minimum legal value for parameter 15
TLMAX15 =                    15 / maximum legal value for parameter 15
TNULL15 =                    -1 / illegal value for column 15
TTYPE16 = 'S_EQ    '            / System Errors (0xf: > 14 errors occured)
TFORM16 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN16 =                     0 / minimum legal value for parameter 16
TLMAX16 =                    15 / maximum legal value for parameter 16
TNULL16 =                    -1 / illegal value for column 16
TTYPE17 = 'O_EQ    '            / Operational Errors (0xf: > 14 errors occured)
TFORM17 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN17 =                     0 / minimum legal value for parameter 17
TLMAX17 =                    15 / maximum legal value for parameter 17
TNULL17 =                    -1 / illegal value for column 17
TTYPE18 = 'QEP_RATEMETER'       / Qualified Event Pulse counter
TFORM18 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN18 =                     0 / minimum legal value for parameter 18
TLMAX18 =                  2000 / maximum legal value for parameter 18
TNULL18 =                    -1 / illegal value for column 18
COMMENT
COMMENT      The QEP Ratemeter counts for all the events with:
COMMENT
COMMENT         Emin < E < Emax
COMMENT         Xmin < X < Xmax
COMMENT         Ymin < Y < Ymax
COMMENT         Vmin < Veto < Vmax
COMMENT         BLmin < BL < BLmax
COMMENT
TTYPE19 = 'EVENT_READ_RATEMETER' / Event Input Buffer counter
TFORM19 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN19 =                     0 / minimum legal value for parameter 19
TLMAX19 =                  2000 / maximum legal value for parameter 19
TNULL19 =                    -1 / illegal value for column 19
TTYPE20 = 'EVENT_LOST_RATEMETER' / DE not transferred event counter
TFORM20 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN20 =                     0 / minimum legal value for parameter 20
TLMAX20 =                  2000 / maximum legal value for parameter 20
TNULL20 =                    -1 / illegal value for column 20
TTYPE21 = 'BASETIME'            / BT value at the Essential HK param. sampling
TFORM21 = 'J       '            / data format of the field: 4-byte INTEGER
TLMIN21 =                     0 / minimum legal value for parameter 21
TLMAX21 =                 32767 / maximum legal value for parameter 21
TNULL21 =                    -1 / illegal value for column 21
TTYPE22 = 'DEADTIME_RATEMETER'  / Dead Time counter
TFORM22 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN22 =                     0 / minimum legal value for parameter 22
```

```
TLMAX22 =                32767 / maximum legal value for parameter 22
TNULL22 =                   -1 / illegal value for column 22
TTYPE23 = 'E_MIN_RATEMETER'    / LLD output counter
TFORM23 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN23 =                    0 / minimum legal value for parameter 23
TLMAX23 =                 8000 / maximum legal value for parameter 23
TNULL23 =                   -1 / illegal value for column 23
TTYPE24 = 'E_MAX_RATEMETER'    / Ra discriminator output counter
TFORM24 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN24 =                    0 / minimum legal value for parameter 24
TLMAX24 =                 8000 / maximum legal value for parameter 24
TNULL24 =                   -1 / illegal value for column 24
TTYPE25 = 'POS_RATEMETER'      / Ry & Rx discriminator counter
TFORM25 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN25 =                    0 / minimum legal value for parameter 25
TLMAX25 =                 8000 / maximum legal value for parameter 25
TNULL25 =                   -1 / illegal value for column 25
TTYPE26 = 'BL_RATEMETER'       / Burst-Length discriminator counter
TFORM26 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN26 =                    0 / minimum legal value for parameter 26
TLMAX26 =                 8000 / maximum legal value for parameter 26
TNULL26 =                   -1 / illegal value for column 26
TTYPE27 = 'VETO_RATEMETER'     / Rv discriminator counter
TFORM27 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN27 =                    0 / minimum legal value for parameter 27
TLMAX27 =                 8000 / maximum legal value for parameter 27
TNULL27 =                   -1 / illegal value for column 27
TTYPE28 = 'CELL_PRESSURE'      / Cell pressure monitor
TFORM28 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN28 =                    0 / minimum legal value for parameter 28
TLMAX28 =                  255 / maximum legal value for parameter 28
TNULL28 =                   -1 / illegal value for column 28
TTYPE29 = 'RESERVOIR_PRESSURE' / Reservoir pressure monitor
TFORM29 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN29 =                    0 / minimum legal value for parameter 29
TLMAX29 =                  255 / maximum legal value for parameter 29
TNULL29 =                   -1 / illegal value for column 29
TTYPE30 = 'GAS_VALVE1_OPERATION' / gas valve 1 operation cycle counter
TFORM30 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN30 =                    0 / minimum legal value for parameter 30
TLMAX30 =                  255 / maximum legal value for parameter 30
TNULL30 =                   -1 / illegal value for column 30
TTYPE31 = 'GAS_VALVE2_OPERATION' / gas valve 2 operation cycle counter
TFORM31 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN31 =                    0 / minimum legal value for parameter 31
TLMAX31 =                  255 / maximum legal value for parameter 31
TNULL31 =                   -1 / illegal value for column 31
TTYPE32 = 'STANDBY_MODE'       / LEGSPC in STANDBY mode
TFORM32 = 'L       '           / data format of the field: LOGICAL
TTYPE33 = 'RUNNING_MODE'       / LEGSPC in RUNNING mode
TFORM33 = 'L       '           / data format of the field: LOGICAL
TTYPE34 = 'NORM_INTERR_HANDLER' / Normal Input Interrupt (T if active)
TFORM34 = 'L       '           / data format of the field: LOGICAL
TTYPE35 = 'HT_INTERR_HANDLER'  / High Throughput Input Interrupt Handler
TFORM35 = 'L       '           / data format of the field: LOGICAL
TTYPE36 = 'DIRECT_MODE_PROC'   / DIRECT Mode Event Processing (T if active)
TFORM36 = 'L       '           / data format of the field: LOGICAL
TTYPE37 = 'SPEC1_MODE_PROC'    / SPEC1 Mode Processing (T if active)
TFORM37 = 'L       '           / data format of the field: LOGICAL
TTYPE38 = 'IMAGE1_MODE_PROC'   / IMAGE1 Mode Processing (T if active)
TFORM38 = 'L       '           / data format of the field: LOGICAL
```

```
TTYPE39 = 'ENG_HK_GENERATION'  / Engineering HK Packet generation (T if active)
TFORM39 = 'L       '           / data format of the field: LOGICAL
TTYPE40 = 'HK_COLLECTION'      / HK Data collection (T if active)
TFORM40 = 'L       '           / data format of the field: LOGICAL
TTYPE41 = 'MACROCOM_PROC'      / Macrocommand Processing (T if active)
TFORM41 = 'L       '           / data format of the field: LOGICAL
TTYPE42 = 'PROT_TASK'          / Protection Task (T if active)
TFORM42 = 'L       '           / data format of the field: LOGICAL
TTYPE43 = 'RTMS    '           / Real Time Monitoring System (T if active)
TFORM43 = 'L       '           / data format of the field: LOGICAL
TTYPE44 = 'CELL_TEMPERATURE'   / Cell Temperature monitor (-55 to 150 C)
TFORM44 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN44 =                    0 / minimum legal value for parameter 44
TLMAX44 =                  255 / maximum legal value for parameter 44
TNULL44 =                   -1 / illegal value for column 44
COMMENT
COMMENT       Specific Conversion Table required for parameter 44
COMMENT
TTYPE45 = 'RESERV_TEMPERATURE' / Reservoir Temperature monitor (-55 to 150 C)
TFORM45 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN45 =                    0 / minimum legal value for parameter 45
TLMAX45 =                  255 / maximum legal value for parameter 45
TNULL45 =                   -1 / illegal value for column 45
COMMENT
COMMENT       Specific Conversion Table required for parameter 45
COMMENT
TTYPE46 = 'EU_TEMPERATURE'     / EU Temperature monitor (-55 to 150 C)
TFORM46 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN46 =                    0 / minimum legal value for parameter 46
TLMAX46 =                  255 / maximum legal value for parameter 46
TNULL46 =                   -1 / illegal value for column 46
COMMENT
COMMENT       Specific Conversion Table required for parameter 46
COMMENT
TTYPE47 = 'CELL_HIGH_VOLTAGE'  / Cell HV monitor  (0 - 20 kV)
TFORM47 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN47 =                    0 / minimum legal value for parameter 47
TLMAX47 =                  255 / maximum legal value for parameter 47
TNULL47 =                   -1 / illegal value for column 47
TTYPE48 = 'PMT_HIGH_VOLTAGE'   / PMT HV monitor (0 - 1.5kV)
TFORM48 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN48 =                    0 / minimum legal value for parameter 48
TLMAX48 =                  255 / maximum legal value for parameter 48
TNULL48 =                   -1 / illegal value for column 48
TTYPE49 = 'LOW_VOLTAGE_+5V'    / +5V DC/DC Converter Voltage monitor
TFORM49 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN49 =                    0 / minimum legal value for parameter 49
TLMAX49 =                  255 / maximum legal value for parameter 49
TNULL49 =                   -1 / illegal value for column 49
TTYPE50 = 'LOW_VOLTAGE_+6V'    / +6V DC/DC Converter Voltage monitor
TFORM50 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN50 =                    0 / minimum legal value for parameter 50
TLMAX50 =                  255 / maximum legal value for parameter 50
TNULL50 =                   -1 / illegal value for column 50
TTYPE51 = 'LOW_VOLTAGE_+15V'   / +15V DC/DC Converter Voltage monitor
TFORM51 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN51 =                    0 / minimum legal value for parameter 51
TLMAX51 =                  255 / maximum legal value for parameter 51
TNULL51 =                   -1 / illegal value for column 51
TTYPE52 = 'LOW_VOLTAGE_-15V'   / -15V DC/DC Converter Voltage monitor
TFORM52 = 'I       '           / data format of the field: 2-byte INTEGER
```

```
TLMIN52 =                    0 / minimum legal value for parameter 52
TLMAX52 =                  255 / maximum legal value for parameter 52
TNULL52 =                   -1 / illegal value for column 52
TTYPE53 = 'LOW_VOLTAGE_+28V'   / +28V DC/DC Converter Voltage monitor
TFORM53 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN53 =                    0 / minimum legal value for parameter 53
TLMAX53 =                  255 / maximum legal value for parameter 53
TNULL53 =                   -1 / illegal value for column 53
TTYPE54 = 'COMMAND_COUNTER'    / DE processed macrocommand counter
TFORM54 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN54 =                    0 / minimum legal value for parameter 54
TLMAX54 =                32767 / maximum legal value for parameter 54
TNULL54 =                   -1 / illegal value for column 54
TTYPE55 = 'PMT_CURR_FLAG_COUNTER' / PMT CurrentFlag counter
TFORM55 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN55 =                    0 / minimum legal value for parameter 55
TLMAX55 =                32767 / maximum legal value for parameter 55
TNULL55 =                   -1 / illegal value for column 55
END
```

A.7 HKP Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                  139 / width of table in bytes
NAXIS2  =                 7175 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   55 / number of fields in each row
EXTNAME = 'HKP     '           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'TEMPORALDATA'       / non-science quantities versus time
HDUCLAS2= 'HKP     '           / housekeeping parameters
HDUCLAS3= 'FULL    '           / non-compressed
HDUCLAS4= 'PHYSICAL'           / parameters in physical units
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644971000000E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652438000000E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4670000000000E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
TIMEDEL =  1.0000000000000E+00 / time granularity of data (size of time bin)
SCALFLAG=                    T / HK parameter values scaled to physical units?
TTYPE1  = 'TIME    '           / time tag (since MJDREF)
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'GETT_ENBL_SW'       / Getter Enable Software flag (T if enabled)
TFORM2  = 'L       '           / data format of the field: LOGICAL
TTYPE3  = 'MEM_MOD_ENBL'       / Memory Modify Software flag (T if enabled)
TFORM3  = 'L       '           / data format of the field: LOGICAL
TTYPE4  = 'GETT_ENBL'          / Getter Enable before RS04 (T if enabled)
TFORM4  = 'L       '           / data format of the field: LOGICAL
TTYPE5  = 'SHUT_CLOSE'         / Shutter Closed Position (T if closed)
TFORM5  = 'L       '           / data format of the field: LOGICAL
TTYPE6  = 'SHUT_OPEN'          / Shutter Open Position (T if open)
TFORM6  = 'L       '           / data format of the field: LOGICAL
TTYPE7  = 'PMT_CURR'           / PMT Current Flag (T if set)
TFORM7  = 'L       '           / data format of the field: LOGICAL
```

```
TTYPE8  = 'GU_SAFETY'           / Gas Unity Safety flag (T if set)
TFORM8  = 'L       '            / data format of the field: LOGICAL
TTYPE9  = 'VALVE_SUPPLY'        / Valve Supply after RS05 (T if enabled)
TFORM9  = 'L       '            / data format of the field: LOGICAL
TTYPE10 = 'GETT_SUPPLY'         / Getter Supply after RS04 (T if enabled)
TFORM10 = 'L       '            / data format of the field: LOGICAL
TTYPE11 = 'HV_POWER'            / HV Power Supply after RS03 (T if enabled)
TFORM11 = 'L       '            / data format of the field: LOGICAL
TTYPE12 = 'HV_ENBL '            / HV Enable status before RS03 (T if enabled)
TFORM12 = 'L       '            / data format of the field: LOGICAL
TTYPE13 = 'FEE_-15V'            / FEE -15V power after RS01 (T if enabled)
TFORM13 = 'L       '            / data format of the field: LOGICAL
TTYPE14 = 'FEE_+15V'            / FEE +15V power after RS01 (T if enabled)
TFORM14 = 'L       '            / data format of the field: LOGICAL
TTYPE15 = 'ST_EQ   '            / Self Test Errors (0xf: > 14 errors occured)
TFORM15 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN15 =                     0 / minimum legal value for parameter 15
TLMAX15 =                    15 / maximum legal value for parameter 15
TNULL15 =                    -1 / illegal value for column 15
TTYPE16 = 'S_EQ    '            / System Errors (0xf: > 14 errors occured)
TFORM16 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN16 =                     0 / minimum legal value for parameter 16
TLMAX16 =                    15 / maximum legal value for parameter 16
TNULL16 =                    -1 / illegal value for column 16
TTYPE17 = 'O_EQ    '            / Operational Errors (0xf: > 14 errors occured)
TFORM17 = 'I       '            / data format of the field: 2-byte INTEGER
TLMIN17 =                     0 / minimum legal value for parameter 17
TLMAX17 =                    15 / maximum legal value for parameter 17
TNULL17 =                    -1 / illegal value for column 17
TTYPE18 = 'QEP_RATEMETER'       / Qualified Event Pulse counter
TFORM18 = 'E       '            / data format of the field: REAL
TUNIT18 = 'count/s '            / physical unit of the field
TLMAX18 =                  2000 / maximum legal value for parameter 18
COMMENT
COMMENT      The QEP Ratemeter counts for all the events with:
COMMENT
COMMENT         Emin < E < Emax
COMMENT         Xmin < X < Xmax
COMMENT         Ymin < Y < Ymax
COMMENT         Vmin < Veto < Vmax
COMMENT         BLmin < BL < BLmax
COMMENT
TTYPE19 = 'EVENT_READ_RATEMETER' / Event Input Buffer counter
TFORM19 = 'E       '            / data format of the field: REAL
TUNIT19 = 'count/s '            / physical unit of the field
TLMAX19 =                  2000 / maximum legal value for parameter 19
TTYPE20 = 'EVENT_LOST_RATEMETER' / DE not transferred event counter
TFORM20 = 'E       '            / data format of the field: REAL
TUNIT20 = 'count/s '            / physical unit of the field
TLMAX20 =                  2000 / maximum legal value for parameter 20
TTYPE21 = 'BASETIME'            / BT value at the Essential HK param. sampling
TFORM21 = 'J       '            / data format of the field: 4-byte INTEGER
TLMIN21 =                     0 / minimum legal value for parameter 21
TLMAX21 =                 32767 / maximum legal value for parameter 21
TNULL21 =                    -1 / illegal value for column 21
TTYPE22 = 'DEADTIME_RATEMETER'  / Dead Time counter
TFORM22 = 'E       '            / data format of the field: REAL
TUNIT22 = 'count/s '            / physical unit of the field
TLMAX22 =                 32767 / maximum legal value for parameter 22
TTYPE23 = 'E_MIN_RATEMETER'     / LLD output counter
TFORM23 = 'E       '            / data format of the field: REAL
```

```
TUNIT23 = 'count/s '           / physical unit of the field
TLMAX23 =                 8000 / maximum legal value for parameter 23
TTYPE24 = 'E_MAX_RATEMETER'    / Ra discriminator output counter
TFORM24 = 'E       '           / data format of the field: REAL
TUNIT24 = 'count/s '           / physical unit of the field
TLMAX24 =                 8000 / maximum legal value for parameter 24
TTYPE25 = 'POS_RATEMETER'      / Ry & Rx discriminator counter
TFORM25 = 'E       '           / data format of the field: REAL
TUNIT25 = 'count/s '           / physical unit of the field
TLMAX25 =                 8000 / maximum legal value for parameter 25
TTYPE26 = 'BL_RATEMETER'       / Burst-Length discriminator counter
TFORM26 = 'E       '           / data format of the field: REAL
TUNIT26 = 'count/s '           / physical unit of the field
TLMAX26 =                 8000 / maximum legal value for parameter 26
TTYPE27 = 'VETO_RATEMETER'     / Rv discriminator counter
TFORM27 = 'E       '           / data format of the field: REAL
TUNIT27 = 'count/s '           / physical unit of the field
TLMAX27 =                 8000 / maximum legal value for parameter 27
TTYPE28 = 'CELL_PRESSURE'      / Cell pressure monitor
TFORM28 = 'E       '           / data format of the field: REAL
TUNIT28 = 'mbar    '           / physical unit of the field
TTYPE29 = 'RESERVOIR_PRESSURE' / Reservoir pressure monitor
TFORM29 = 'E       '           / data format of the field: REAL
TUNIT29 = 'mbar    '           / physical unit of the field
TTYPE30 = 'GAS_VALVE1_OPERATION' / gas valve 1 operation cycle counter
TFORM30 = 'E       '           / data format of the field: REAL
TUNIT30 = 'count/s '           / physical unit of the field
TTYPE31 = 'GAS_VALVE2_OPERATION' / gas valve 2 operation cycle counter
TFORM31 = 'E       '           / data format of the field: REAL
TUNIT31 = 'count/s '           / physical unit of the field
TTYPE32 = 'STANDBY_MODE'       / LEGSPC in STANDBY mode
TFORM32 = 'L       '           / data format of the field: LOGICAL
TTYPE33 = 'RUNNING_MODE'       / LEGSPC in RUNNING mode
TFORM33 = 'L       '           / data format of the field: LOGICAL
TTYPE34 = 'NORM_INTERR_HANDLER' / Normal Input Interrupt (T if active)
TFORM34 = 'L       '           / data format of the field: LOGICAL
TTYPE35 = 'HT_INTERR_HANDLER'  / High Throughput Input Interrupt Handler
TFORM35 = 'L       '           / data format of the field: LOGICAL
TTYPE36 = 'DIRECT_MODE_PROC'   / DIRECT Mode Event Processing (T if active)
TFORM36 = 'L       '           / data format of the field: LOGICAL
TTYPE37 = 'SPEC1_MODE_PROC'    / SPEC1 Mode Processing (T if active)
TFORM37 = 'L       '           / data format of the field: LOGICAL
TTYPE38 = 'IMAGE1_MODE_PROC'   / IMAGE1 Mode Processing (T if active)
TFORM38 = 'L       '           / data format of the field: LOGICAL
TTYPE39 = 'ENG_HK_GENERATION'  / Engineering HK Packet generation (T if active)
TFORM39 = 'L       '           / data format of the field: LOGICAL
TTYPE40 = 'HK_COLLECTION'      / HK Data collection (T if active)
TFORM40 = 'L       '           / data format of the field: LOGICAL
TTYPE41 = 'MACROCOM_PROC'      / Macrocommand Processing (T if active)
TFORM41 = 'L       '           / data format of the field: LOGICAL
TTYPE42 = 'PROT_TASK'          / Protection Task (T if active)
TFORM42 = 'L       '           / data format of the field: LOGICAL
TTYPE43 = 'RTMS    '           / Real Time Monitoring System (T if active)
TFORM43 = 'L       '           / data format of the field: LOGICAL
TTYPE44 = 'CELL_TEMPERATURE'   / Cell Temperature monitor (-55 to 150 C)
TFORM44 = 'E       '           / data format of the field: REAL
TUNIT44 = 'Celsius '           / physical unit of the field
TTYPE45 = 'RESERV_TEMPERATURE' / Reservoir Temperature monitor (-55 to 150 C)
TFORM45 = 'E       '           / data format of the field: REAL
TUNIT45 = 'Celsius '           / physical unit of the field
TTYPE46 = 'EU_TEMPERATURE'     / EU Temperature monitor (-55 to 150 C)
```

```
TFORM46 = 'E       '           / data format of the field: REAL
TUNIT46 = 'Celsius '           / physical unit of the field
TTYPE47 = 'CELL_HIGH_VOLTAGE'  / Cell HV monitor  (0 - 20 kV)
TFORM47 = 'E       '           / data format of the field: REAL
TUNIT47 = 'V       '           / physical unit of the field
TTYPE48 = 'PMT_HIGH_VOLTAGE'   / PMT HV monitor (0 - 1.5kV)
TFORM48 = 'E       '           / data format of the field: REAL
TUNIT48 = 'V       '           / physical unit of the field
TTYPE49 = 'LOW_VOLTAGE_+5V'    / +5V DC/DC Converter Voltage monitor
TFORM49 = 'E       '           / data format of the field: REAL
TUNIT49 = 'V       '           / physical unit of the field
TTYPE50 = 'LOW_VOLTAGE_+6V'    / +6V DC/DC Converter Voltage monitor
TFORM50 = 'E       '           / data format of the field: REAL
TUNIT50 = 'V       '           / physical unit of the field
TTYPE51 = 'LOW_VOLTAGE_+15V'   / +15V DC/DC Converter Voltage monitor
TFORM51 = 'E       '           / data format of the field: REAL
TUNIT51 = 'V       '           / physical unit of the field
TTYPE52 = 'LOW_VOLTAGE_-15V'   / -15V DC/DC Converter Voltage monitor
TFORM52 = 'E       '           / data format of the field: REAL
TUNIT52 = 'V       '           / physical unit of the field
TTYPE53 = 'LOW_VOLTAGE_+28V'   / +28V DC/DC Converter Voltage monitor
TFORM53 = 'E       '           / data format of the field: REAL
TUNIT53 = 'V       '           / physical unit of the field
TTYPE54 = 'COMMAND_COUNTER'    / DE processed macrocommand counter
TFORM54 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN54 =                    0 / minimum legal value for parameter 54
TLMAX54 =                32767 / maximum legal value for parameter 54
TNULL54 =                   -1 / illegal value for column 54
TTYPE55 = 'PMT_CURR_FLAG_COUNTER' / PMT CurrentFlag counter
TFORM55 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN55 =                    0 / minimum legal value for parameter 55
TLMAX55 =                32767 / maximum legal value for parameter 55
TNULL55 =                   -1 / illegal value for column 55
END
```

A.8 HKP+GAIN Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                  191 / width of table in bytes
NAXIS2  =                 7175 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   68 / number of fields in each row
EXTNAME = 'HKP+GAIN'           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'TEMPORALDATA'       / non-science quantities versus time
HDUCLAS2= 'HKP     '           / housekeeping parameters
HDUCLAS3= 'FULL    '           / non-compressed
HDUCLAS4= 'PHYSICAL+GAIN'      / physical unit + gain columns
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644971000000E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652438000000E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4670000000000E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
TIMEDEL =  1.0000000000000E+00 / time granularity of data (size of time bin)
SCALFLAG=                    T / HK parameter values scaled to physical units?
TTYPE1  = 'TIME    '           / time tag (since MJDREF)
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'GETT_ENBL_SW'       / Getter Enable Software flag (T if enabled)
TFORM2  = 'L       '           / data format of the field: LOGICAL
TTYPE3  = 'MEM_MOD_ENBL'       / Memory Modify Software flag (T if enabled)
TFORM3  = 'L       '           / data format of the field: LOGICAL
TTYPE4  = 'GETT_ENBL'          / Getter Enable before RS04 (T if enabled)
TFORM4  = 'L       '           / data format of the field: LOGICAL
TTYPE5  = 'SHUT_CLOSE'         / Shutter Closed Position (T if closed)
TFORM5  = 'L       '           / data format of the field: LOGICAL
TTYPE6  = 'SHUT_OPEN'          / Shutter Open Position (T if open)
TFORM6  = 'L       '           / data format of the field: LOGICAL
TTYPE7  = 'PMT_CURR'           / PMT Current Flag (T if set)
TFORM7  = 'L       '           / data format of the field: LOGICAL
```

```
TTYPE8  = ’GU_SAFETY’           / Gas Unity Safety flag (T if set)
TFORM8  = ’L       ’            / data format of the field: LOGICAL
TTYPE9  = ’VALVE_SUPPLY’        / Valve Supply after RS05 (T if enabled)
TFORM9  = ’L       ’            / data format of the field: LOGICAL
TTYPE10 = ’GETT_SUPPLY’         / Getter Supply after RS04 (T if enabled)
TFORM10 = ’L       ’            / data format of the field: LOGICAL
TTYPE11 = ’HV_POWER’            / HV Power Supply after RS03 (T if enabled)
TFORM11 = ’L       ’            / data format of the field: LOGICAL
TTYPE12 = ’HV_ENBL ’            / HV Enable status before RS03 (T if enabled)
TFORM12 = ’L       ’            / data format of the field: LOGICAL
TTYPE13 = ’FEE_-15V’            / FEE -15V power after RS01 (T if enabled)
TFORM13 = ’L       ’            / data format of the field: LOGICAL
TTYPE14 = ’FEE_+15V’            / FEE +15V power after RS01 (T if enabled)
TFORM14 = ’L       ’            / data format of the field: LOGICAL
TTYPE15 = ’ST_EQ   ’            / Self Test Errors (0xf: > 14 errors occured)
TFORM15 = ’I       ’            / data format of the field: 2-byte INTEGER
TLMIN15 =                     0 / minimum legal value for parameter 15
TLMAX15 =                    15 / maximum legal value for parameter 15
TNULL15 =                    -1 / illegal value for column 15
TTYPE16 = ’S_EQ    ’            / System Errors (0xf: > 14 errors occured)
TFORM16 = ’I       ’            / data format of the field: 2-byte INTEGER
TLMIN16 =                     0 / minimum legal value for parameter 16
TLMAX16 =                    15 / maximum legal value for parameter 16
TNULL16 =                    -1 / illegal value for column 16
TTYPE17 = ’O_EQ    ’            / Operational Errors (0xf: > 14 errors occured)
TFORM17 = ’I       ’            / data format of the field: 2-byte INTEGER
TLMIN17 =                     0 / minimum legal value for parameter 17
TLMAX17 =                    15 / maximum legal value for parameter 17
TNULL17 =                    -1 / illegal value for column 17
TTYPE18 = ’QEP_RATEMETER’       / Qualified Event Pulse counter
TFORM18 = ’E       ’            / data format of the field: REAL
TUNIT18 = ’count/s ’            / physical unit of the field
TLMAX18 =                  2000 / maximum legal value for parameter 18
COMMENT
COMMENT      The QEP Ratemeter counts for all the events with:
COMMENT
COMMENT         Emin < E < Emax
COMMENT         Xmin < X < Xmax
COMMENT         Ymin < Y < Ymax
COMMENT         Vmin < Veto < Vmax
COMMENT         BLmin < BL < BLmax
COMMENT
TTYPE19 = ’EVENT_READ_RATEMETER’ / Event Input Buffer counter
TFORM19 = ’E       ’            / data format of the field: REAL
TUNIT19 = ’count/s ’            / physical unit of the field
TLMAX19 =                  2000 / maximum legal value for parameter 19
TTYPE20 = ’EVENT_LOST_RATEMETER’ / DE not transferred event counter
TFORM20 = ’E       ’            / data format of the field: REAL
TUNIT20 = ’count/s ’            / physical unit of the field
TLMAX20 =                  2000 / maximum legal value for parameter 20
TTYPE21 = ’BASETIME’            / BT value at the Essential HK param. sampling
TFORM21 = ’J       ’            / data format of the field: 4-byte INTEGER
TLMIN21 =                     0 / minimum legal value for parameter 21
TLMAX21 =                 32767 / maximum legal value for parameter 21
TNULL21 =                    -1 / illegal value for column 21
TTYPE22 = ’DEADTIME_RATEMETER’  / Dead Time counter
TFORM22 = ’E       ’            / data format of the field: REAL
TUNIT22 = ’count/s ’            / physical unit of the field
TLMAX22 =                 32767 / maximum legal value for parameter 22
TTYPE23 = ’E_MIN_RATEMETER’     / LLD output counter
TFORM23 = ’E       ’            / data format of the field: REAL
```

```
TUNIT23 = 'count/s '           / physical unit of the field
TLMAX23 =                 8000 / maximum legal value for parameter 23
TTYPE24 = 'E_MAX_RATEMETER'    / Ra discriminator output counter
TFORM24 = 'E       '           / data format of the field: REAL
TUNIT24 = 'count/s '           / physical unit of the field
TLMAX24 =                 8000 / maximum legal value for parameter 24
TTYPE25 = 'POS_RATEMETER'      / Ry & Rx discriminator counter
TFORM25 = 'E       '           / data format of the field: REAL
TUNIT25 = 'count/s '           / physical unit of the field
TLMAX25 =                 8000 / maximum legal value for parameter 25
TTYPE26 = 'BL_RATEMETER'       / Burst-Length discriminator counter
TFORM26 = 'E       '           / data format of the field: REAL
TUNIT26 = 'count/s '           / physical unit of the field
TLMAX26 =                 8000 / maximum legal value for parameter 26
TTYPE27 = 'VETO_RATEMETER'     / Rv discriminator counter
TFORM27 = 'E       '           / data format of the field: REAL
TUNIT27 = 'count/s '           / physical unit of the field
TLMAX27 =                 8000 / maximum legal value for parameter 27
TTYPE28 = 'CELL_PRESSURE'      / Cell pressure monitor
TFORM28 = 'E       '           / data format of the field: REAL
TUNIT28 = 'mbar    '           / physical unit of the field
TTYPE29 = 'RESERVOIR_PRESSURE' / Reservoir pressure monitor
TFORM29 = 'E       '           / data format of the field: REAL
TUNIT29 = 'mbar    '           / physical unit of the field
TTYPE30 = 'GAS_VALVE1_OPERATION' / gas valve 1 operation cycle counter
TFORM30 = 'E       '           / data format of the field: REAL
TUNIT30 = 'count/s '           / physical unit of the field
TTYPE31 = 'GAS_VALVE2_OPERATION' / gas valve 2 operation cycle counter
TFORM31 = 'E       '           / data format of the field: REAL
TUNIT31 = 'count/s '           / physical unit of the field
TTYPE32 = 'STANDBY_MODE'       / LEGSPC in STANDBY mode
TFORM32 = 'L       '           / data format of the field: LOGICAL
TTYPE33 = 'RUNNING_MODE'       / LEGSPC in RUNNING mode
TFORM33 = 'L       '           / data format of the field: LOGICAL
TTYPE34 = 'NORM_INTERR_HANDLER' / Normal Input Interrupt (T if active)
TFORM34 = 'L       '           / data format of the field: LOGICAL
TTYPE35 = 'HT_INTERR_HANDLER'  / High Throughput Input Interrupt Handler
TFORM35 = 'L       '           / data format of the field: LOGICAL
TTYPE36 = 'DIRECT_MODE_PROC'   / DIRECT Mode Event Processing (T if active)
TFORM36 = 'L       '           / data format of the field: LOGICAL
TTYPE37 = 'SPEC1_MODE_PROC'    / SPEC1 Mode Processing (T if active)
TFORM37 = 'L       '           / data format of the field: LOGICAL
TTYPE38 = 'IMAGE1_MODE_PROC'   / IMAGE1 Mode Processing (T if active)
TFORM38 = 'L       '           / data format of the field: LOGICAL
TTYPE39 = 'ENG_HK_GENERATION'  / Engineering HK Packet generation (T if active)
TFORM39 = 'L       '           / data format of the field: LOGICAL
TTYPE40 = 'HK_COLLECTION'      / HK Data collection (T if active)
TFORM40 = 'L       '           / data format of the field: LOGICAL
TTYPE41 = 'MACROCOM_PROC'      / Macrocommand Processing (T if active)
TFORM41 = 'L       '           / data format of the field: LOGICAL
TTYPE42 = 'PROT_TASK'          / Protection Task (T if active)
TFORM42 = 'L       '           / data format of the field: LOGICAL
TTYPE43 = 'RTMS    '           / Real Time Monitoring System (T if active)
TFORM43 = 'L       '           / data format of the field: LOGICAL
TTYPE44 = 'CELL_TEMPERATURE'   / Cell Temperature monitor (-55 to 150 C)
TFORM44 = 'E       '           / data format of the field: REAL
TUNIT44 = 'Celsius '           / physical unit of the field
TTYPE45 = 'RESERV_TEMPERATURE' / Reservoir Temperature monitor (-55 to 150 C)
TFORM45 = 'E       '           / data format of the field: REAL
TUNIT45 = 'Celsius '           / physical unit of the field
TTYPE46 = 'EU_TEMPERATURE'     / EU Temperature monitor (-55 to 150 C)
```

```
TFORM46 = 'E       '           / data format of the field: REAL
TUNIT46 = 'Celsius '           / physical unit of the field
TTYPE47 = 'CELL_HIGH_VOLTAGE'  / Cell HV monitor  (0 - 20 kV)
TFORM47 = 'E       '           / data format of the field: REAL
TUNIT47 = 'V       '           / physical unit of the field
TTYPE48 = 'PMT_HIGH_VOLTAGE'   / PMT HV monitor (0 - 1.5kV)
TFORM48 = 'E       '           / data format of the field: REAL
TUNIT48 = 'V       '           / physical unit of the field
TTYPE49 = 'LOW_VOLTAGE_+5V'    / +5V DC/DC Converter Voltage monitor
TFORM49 = 'E       '           / data format of the field: REAL
TUNIT49 = 'V       '           / physical unit of the field
TTYPE50 = 'LOW_VOLTAGE_+6V'    / +6V DC/DC Converter Voltage monitor
TFORM50 = 'E       '           / data format of the field: REAL
TUNIT50 = 'V       '           / physical unit of the field
TTYPE51 = 'LOW_VOLTAGE_+15V'   / +15V DC/DC Converter Voltage monitor
TFORM51 = 'E       '           / data format of the field: REAL
TUNIT51 = 'V       '           / physical unit of the field
TTYPE52 = 'LOW_VOLTAGE_-15V'   / -15V DC/DC Converter Voltage monitor
TFORM52 = 'E       '           / data format of the field: REAL
TUNIT52 = 'V       '           / physical unit of the field
TTYPE53 = 'LOW_VOLTAGE_+28V'   / +28V DC/DC Converter Voltage monitor
TFORM53 = 'E       '           / data format of the field: REAL
TUNIT53 = 'V       '           / physical unit of the field
TTYPE54 = 'COMMAND_COUNTER'    / DE processed macrocommand counter
TFORM54 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN54 =                    0 / minimum legal value for parameter 54
TLMAX54 =                32767 / maximum legal value for parameter 54
TNULL54 =                   -1 / illegal value for column 54
TTYPE55 = 'PMT_CURR_FLAG_COUNTER' / PMT CurrentFlag counter
TFORM55 = 'I       '           / data format of the field: 2-byte INTEGER
TLMIN55 =                    0 / minimum legal value for parameter 55
TLMAX55 =                32767 / maximum legal value for parameter 55
TNULL55 =                   -1 / illegal value for column 55
COMMENT
COMMENT   The following columns contain information concerning the computed
COMMENT   time-dependent energy gain of the two fixed Fe calibration sources
COMMENT   inside the SAX/LEGSPC. For a description of the utilized algorithm
COMMENT   and computed quantities see the SAX Data Analysis Software User
COMMENT   Manual.
COMMENT
TIMEGRAN=  9.9694627685547E+01 / actual time granularity of gain values
NMPHAFE1=  4.1000000000000E+02 / nominal PHA of calibration source #1
NMPHAFE2=  4.1000000000000E+02 / nominal PHA of calibration source #2
PHAFITFN= 'SNGGAUSS'           / type of function to fit cal. src. spectra
TTYPE56 = 'PHA1    '           / center of fitted spectrum for cal. source #1
TFORM56 = 'E       '           / data format of the field: REAL
TUNIT56 = 'chan    '           / physical unit of field
TTYPE57 = 'FWHM1   '           / FWHM of fitted Gaussian for cal. source #1
TFORM57 = 'E       '           / data format of the field: REAL
TUNIT57 = 'chan    '           / physical unit of field
TTYPE58 = 'CHISQ1  '           / chi^2 of Gaussian fit to calculate GAIN1
TFORM58 = 'E       '           / data format of the field: REAL
TTYPE59 = 'GAIN1   '           / gain of calibration source #1
TFORM59 = 'E       '           / data format of the field: REAL
TTYPE60 = 'PHA2    '           / center of fitted spectrum for cal. source #2
TFORM60 = 'E       '           / data format of the field: REAL
TUNIT60 = 'chan    '           / physical unit of field
TTYPE61 = 'FWHM2   '           / FWHM of fitted Gaussian for cal. source #2
TFORM61 = 'E       '           / data format of the field: REAL
TUNIT61 = 'chan    '           / physical unit of field
TTYPE62 = 'CHISQ2  '           / chi^2 of Gaussian fit to calculate GAIN2
```

```
TFORM62 = 'E       '           / data format of the field: REAL
TTYPE63 = 'GAIN2   '           / gain of calibration source #2
TFORM63 = 'E       '           / data format of the field: REAL
TTYPE64 = 'GAIN12  '           / ratio GAIN1/GAIN2
TFORM64 = 'E       '           / data format of the field: REAL
TTYPE65 = 'GAIN_AVG'           / average gain
TFORM65 = 'E       '           / data format of the field: REAL
TTYPE66 = 'CHISQAVG'           / chi^2 of Gaussian fit to calculate GAIN_AVG
TFORM66 = 'E       '           / data format of the field: REAL
TTYPE67 = 'BCKGND1 '           / background flux outside FOV in box #1
TFORM67 = 'E       '           / data format of the field: REAL
TUNIT67 = 'count/s/cm**2'      / physical unit of field
TTYPE68 = 'BCKGND2 '           / background flux outside FOV in box #2
TFORM68 = 'E       '           / data format of the field: REAL
TUNIT68 = 'count/s/cm**2'      / physical unit of field
END
```

A.9 EPHEMERIS Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   86 / width of table in bytes
NAXIS2  =                 7500 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   17 / number of fields in each row
EXTNAME = 'EPHEMERIS'          / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'TEMPORALDATA'       / non-science quantities versus time
HDUCLAS2= 'EPHEM   '           / spacecraft orbit data
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1644961000000E+07 / data start time relative to MJDREF (s)
TSTOP   =  6.1652460000000E+07 / data end time relative to MJDREF (s)
TELAPSE =  7.4990000000000E+03 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
TIMEDEL =      1.0000000000000 / time granularity of data (size of time bin)
TTYPE1  = 'TIME    '           / time tag (time since MJDREF)
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'SAT_X   '           / X component of the S/C position vector
TFORM2  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT2  = 'km      '           / physical unit of field
TTYPE3  = 'SAT_Y   '           / Y component of the S/C position vector
TFORM3  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT3  = 'km      '           / physical unit of field
TTYPE4  = 'SAT_Z   '           / Z component of the S/C position vector
TFORM4  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT4  = 'km      '           / physical unit of field
TTYPE5  = 'VEL_X   '           / X component of the S/C velocity vector
TFORM5  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT5  = 'km/s    '           / physical unit of field
TTYPE6  = 'VEL_Y   '           / Y component of the S/C velocity vector
TFORM6  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT6  = 'km/s    '           / physical unit of field
```

```
TTYPE7  = 'VEL_Z   '           / Z component of the S/C velocity vector
TFORM7  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT7  = 'km/s    '           / physical unit of field
TTYPE8  = 'LEAP_SEC'           / difference between TAI and UTC
TFORM8  = 'I       '           / data format of the field: 2-byte INTEGER
TUNIT8  = 's       '           / physical unit of field
TTYPE9  = 'SAA_FLAG'           / is S/C inside SAA?
TFORM9  = 'L       '           / data format of the field: LOGICAL
TTYPE10 = 'SAA_TIME'           / time elapsed since S/C entered/exited SAA
TFORM10 = 'E       '           / data format of the field: REAL
TUNIT10 = 'min     '           / physical unit of field
TTYPE11 = 'DAYSIDE_FLAG'       / is S/C on earth's dayside?
TFORM11 = 'L       '           / data format of the field: LOGICAL
TTYPE12 = 'DAYSIDE_TIME'       / time elapsed since S/C entered/exited dayside
TFORM12 = 'E       '           / data format of the field: REAL
TUNIT12 = 'min     '           / physical unit of field
TTYPE13 = 'FOV_FLAG'           / pointing direction: sky/bright-/dark earth
TFORM13 = 'I       '           / data format of the field: 2-byte INTEGER
TTYPE14 = 'EARTH_ANG'          / angle: instrument axis <-> earth limb
TFORM14 = 'E       '           / data format of the field: REAL
TUNIT14 = 'deg     '           / physical unit of field
TTYPE15 = 'BREARTH_ANG'        / angle: instrument axis <-> day/night terminator
TFORM15 = 'E       '           / data format of the field: REAL
TUNIT15 = 'deg     '           / physical unit of field
TTYPE16 = 'SUN_ANG '           / angle: instrument angle <-> direction to sun
TFORM16 = 'E       '           / data format of the field: REAL
TUNIT16 = 'deg     '           / physical unit of field
TTYPE17 = 'COR_MIN '           / minimum geomagnetic cut-off rigidity
TFORM17 = 'E       '           / data format of the field: REAL
TUNIT17 = 'GeV/c   '           / physical unit of field
END
```

A.10 ATTITUDE Binary Table header

```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   64 / width of table in bytes
NAXIS2  =                 3240 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                    8 / number of fields in each row
EXTNAME = 'ASPECT  '           / name of this binary table extension
HDUCLASS= 'SAXLEDAS'           / SAX/LECS DA System (follows OGIP conventions)
HDUDOC  = 'SAX/LEDA/0002'      / format specifications
HDUVERS = '1.2.0   '           / version of format
HDUCLAS1= 'TEMPORALDATA'       / non-science quantities versus time
HDUCLAS2= 'ASPECT  '           / spacecraft aspect data
TELESCOP= 'SAX     '           / mission name
INSTRUME= 'LECS    '           / instrument name
OBS_MODE= 'POINTING'           / observ. mode: POINTING, SLEW, or CALIBRATION
OBJECT  = 'Capricorno'         / name of object pointed at
RA_OBJ  =                 37.6 / nominal RA of target (deg)
DEC_OBJ =                 89.3 / nominal DEC of target (deg)
RADECSYS= 'FK5     '           / World Coordinate System
EQUINOX =  2.0000000000000E+03 / equinox
OBS_ID  = '1       '           / observation number in current observing period
TIMESYS = '1994 1 1 00:00:00'  / all times measured from this reference date
TIMEUNIT= 's       '           / unit for TSTART, TSTOP, TELAPSE, ONTIME
TIMEREF = 'LOCAL   '           / reference frame for reported time values
TASSIGN = 'SATELLITE'          / location of time assignment
CLOCKAPP=                    T / Time corrected for clock drift?
MJDREF  =  4.9353000000000E+04 / mission reference time: 01/01/94 00:00:00 UTC
MJD-OBS =  5.0066483344907E+04 / observation UTC start time (MJD)
TSTART  =  6.1670761000000E+07 / data start time relative to MJDREF (s)
TSTOP   =   .0000000000000E+00 / data end time relative to MJDREF (s)
TELAPSE =  2.5800000000000E+04 / difference: TSTOP - TSTART (s)
DATE-OBS= '15/12/95'           / UTC date of observation start (DD/MM/YY)
TIME-OBS= '11:36:01.000'       / UTC time of observation start (HH:MM:SS)
DATE-END= '15/12/95'           / UTC date of observation end (DD/MM/YY)
TIME-END= '13:41:00.000'       / UTC time of observation end (HH:MM:SS)
SCSEQBEG= '98F9.0000'          / spacecraft clock (s) at observation start
SCSEQEND= 'B644.0000'          / spacecraft clock (s) at observation end
TIMEDEL =       .5000000000000 / time granularity of data (size of time bin)
TTYPE1  = 'TIME    '           / time tag (time since MJDREF)
TFORM1  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT1  = 's       '           / physical unit of field
TTYPE2  = 'RA_SCX  '           / right ascension of ACA X axis
TFORM2  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT2  = 'deg     '           / physical unit of field
TTYPE3  = 'DEC_SCX '           / declination of ACA X axis
TFORM3  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT3  = 'deg     '           / physical unit of field
TTYPE4  = 'RA_SCY  '           / right ascension of ACA Y axis
TFORM4  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT4  = 'deg     '           / physical unit of field
TTYPE5  = 'DEC_SCY '           / declination of ACA Y axis
TFORM5  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT5  = 'deg     '           / physical unit of field
TTYPE6  = 'RA_SCZ  '           / right ascension of ACA Z axis
TFORM6  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT6  = 'deg     '           / physical unit of field
```

```
TTYPE7  = 'DEC_SCZ '           / declination of ACA Z axis
TFORM7  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT7  = 'deg     '           / physical unit of field
TTYPE8  = 'ROLL_AN '           / instrumental roll angle
TFORM8  = 'D       '           / data format of the field: DOUBLE PRECISION
TUNIT8  = 'deg     '           / physical unit of field
END
```

B List of acronyms

ACA	Attitude Control Axis
AE	Analogue Electronics assembly
AT	Attitude Table
BL	Burst Length
DC	Direct Current
DE	Digital Electronics assembly
DM	Direct Mode
EGSE	Electrical Ground Support Equipment
ELT	Event List Table
ET	Ephemeris Table
EU	Electronics Unit
F	False
FAQ	Frequenlty Asked Questions
FBT	FITS Binary Table
FEE	Front End Electronics
FITS	Flexible Image Transport System
FOT	Final Observation Tape
FOV	Field Of View
FWHM	Full Width Half Maximum
GU	Gas Unit
HEASARC	High Energy Astrophysics Science Archive Research Center
HDU	HeaDer Unit
HK	HouseKeeping
HV	High Voltage
LECS	Low-Energy Concentrator Spectrometer
LEGSPC	Low-Energy Gas Scintillation Proportional Counter
LLD	Lower Level Discriminator
MJD	Modified Julian Day (JD - 2400000.5)
NASA	National Aeronautics and Space Administration
NFI	Narrow Field Instrument
OBIs	OBservation Intervals
OFWG	OGIP FITS Working Group
OGIP	Office for Guest Investigation Programs
OI	Observation Interval
OP	Observation Period
PHA	Pulse Height Amplitude
PMT	PhotoMultiplier Tube
SAA	South Atlantic Anomaly
SAX	Satellite per Astronomia X (X-ray Astronomy Satellite)
SAXLEDAS	SAX/LEGSPC Data Analysis System
S/C	SpaceCraft
S/W	Software
T	True
TAI	International Atomic Time
URL	Uniform Resource Locator
UT	Universal Time
UTC	Coordinated Universal Time

References

[1] L. Piro et al. *SAX Observers' Handbook*, July 1995. Issue 1.0.

[2] A.N. Parmar, D.D.E. Martin, M. Bavdaz, F. Favata, E. Kuulkers, G. Vacanti, U. Lammers, A. Peacock, and B.G. Taylor. The low-energy concentrator spectrometer on-board the SAX X-ray astronomy satellite. *A&A*, 1996.

[3] B.W. Kernighan and P. Pike. *The UNIX Programmning Environment*. Prentice-Hall, Inc., 1984.

[4] NASA/Science Office of Standards and Technology. *Definition of the Flexible Image Transport System (FITS)*, nost 100-1.0 edition, June 1993.

[5] NASA/Science Office of Standards and Technology. *A User's Guide for the Flexible Image Transport System (FITS)*, 3.1 edition, May 1994.

[6] J.K. Blackburn, E.A. Greene, and B. Pence. *Mini-Users' Guide to FTOOLS*. NASA/HEASARC, 2.6 edition, October 1993.

[7] HEASARC. Legacy. Technical Report 5, NASA/GSFC, November 1994.

[8] D.D.E Martin. *SAX/LEGSPC User Manual*. ESA/SSD, Nov 1995. Issue 9.0.

[9] U. Lammers. *The SAX LECS Data Analysis System - Online Manual Pages*. ESA/SSD, 1995.

[10] W. Pence, A. Rots, L. Angelini, and J. McDowell. Naming conventions for fits coordinate system keywords. Technical Report OGIP/94-006, NASA/GSFC, 1994.

[11] E.W. Greisen and M. Calabretta. Representations of celestial coordinates in fits. Technical report, NRAO, 1994.

[12] U. Lammers. *The SAX LEGSPC Data Analysis System - Software User Manual*. ESA/SSD, 1996. Doc-ID: LEDA/0010.