LAT Photons

Jump to updates of 22 Jul 2010

Jump to updates of 29 Jun 2010

Jump to updates of 19 Aug 2008

Jump to updates of 20 Nov 2007

LAT Photons (FT1, LS-002) [formerly Event Summary Data]

This is a fundamental data product used by the science tools. The current definition (version of July 25, 2006 as of this writing) is posted on the Guidelines for Science Tools Design page maintained by Masa Hirayama.

The issues and proposed changes below primarily are with respect to the 'LAT event summary' extension in the version linked above. You may want to have that page open when you read this page. The initial concept for the contents dates at least to 2001, when we were not burdened with a detailed understanding of what would be available from reconstruction and classification and how the tools would work, and so should be updated in several respects.

The issues and proposed resolutions below, or any other aspect of the definition of the contents are open for comment (probably most effectively by editing this page rather than inserting Confluence comments).

Proposed resolutions for the issues below were added 29 November 2005 by S. Digel. Some issues still outstanding were also highlighted.

1. Duplication of keywords

(*Digel*) **TELESCOP, INSTRUME, EQUINOX, RADECSYS, DATE, DATE-OBS, DATE-END** are duplicated from the primary header of the file. Is this necessary, or even a good idea? I propose putting **TELESCOP, INSTRUME, DATE, DATE-OBS, and DATE-END** in the primary header only, and **EQUIN OX** and **RADECSYS** in the Lat event summary header.

(*Foschini*) Perhaps it is better to keep all the above keywords in the events and GTI extensions and left blank the primary header. The latter is generally not used by common data analysis software.

(*Stephens*) I think they should stay in all the headers. The idea is that each extension can stand alone if necessary and have the relevant keywords. The only reason to leave them in the primary header is for informational purposes. i.e. you just have to read the primary header and you know what the file is about.

(*Ballet*) I agree that the extensions are more important than the primary, and that the best solution is to have the general keywords in both the extensions and the primary.

(*Hirayama*) The repetetion of "major" keywords (such as **TELESCOP**) was suggested by the HEASARC FITS Working Group when we met them at NASA /GSFC on May 28th, 2003. See also section titled "Answers to the questions" under topic 006 of the latest topics of FT1 and FT2 for more details. Because the FT1 definition changed quite a bit since then, it could be useful if some of us can meet the HEASARC FITS Working Group members to discuss on this topic.

Proposed resolution: Continue duplicating the keywords between headers.

2. HDUCLASS keywords

(*Digel*) I am not sure whether we adhere to the strict definition of a HDUCLASS1 = EVENTS file. The only column that we have in common with the specification of The Recommended Columns and Keywords for a FITS Event List is TIME.

Also, I think that with the HDUCLASS keywords we are supposed to supply a format-specifying document with the **HDUDOC** keyword, and a version in **HD UVERS**. We don't have such a document, and we already have (but do not yet use) a **VERSION** keyword in the primary header.

For these reasons, I'd recommend removing **HDUCLASS** keywords entirely.

(*Hirayama*) The **HDUCLASS** and **HDUCLASn** keywords were added based on the HEASARC FITS Working Group's suggestion made when we met them at NASA/GSFC on May 28th, 2003. See also topic 002 of the latest topics of FT1 and FT2 for more details. I guess they will not appreciate it very much if we drop those keywords, although I can't (and shouldn't) speak for them, of course. In any case, I suggest to ask for their opinion before making a dicision to drop those keywords.

Proposed resolution: As part of a review by the HFWG, we should ask for concurrence about dropping the **HDUCLASS** keywords. In any case we are far enough from matching the columns of a **HDUCLASS1 = EVENTS** file that we shouldn't claim to match that definition.

3. TASSIGN keyword

(*Digel*) **TASSIGN** is supposed to be where the event times were assigned, and we currently have this set equal to 'SATELLITE'. This is not strictly true, as we will need some ground processing to turn ticks of the 20 MHz clock into time since the last 1 PPS signal from the spacecraft, and we'll also need to shift GPS time to TT.

Also, I don't see what good the keyword does us. So I recommend removing it.

(*Hirayama*) For usage (and the meaning) of **TASSIGN** keyword, see the HEASARC's description. Based on the description, 'SATELLITE' is appropreate for our case, in my opinion.

It is a part of standard time-related keywords listed under section titled "B. Time Keywords" in "The Recommended Columns and Keywords for a FITS Event List" and that's why it was added, I think, although I am not sure whether it is absolutely necessary or not. Indeed, gtbary seems not to read nor write this keyword. Also, in the meeting with the HEASARC FITS Working Group at NASA/GSFC on May 28th, 2003, it was suggested (if I remember correctly) that not all of the listed time-related keywords are necessary (See also topic 004 of the latest topics of FT1 and FT2 for more details). On the other hand, (I believe) it is a commonly-used keyword among other high-energy astrophisics missions and having this keyword in our FT1 file doesn't hurt us very much. So, I would recommend to keep it "just in case."

In any case (to drop it or to keep it), the HEASARC FITS Working Group might have a different opinion on this topic, especially from a point of view of multi-mission support.

Proposed resolution: Drop it unless the HFWG wants a very broad interpretation of the meaning of **TASSIGN = 'SATELLITE'**.

4. OBS_ID and OBJECT keywords

(*Digel*) I don't think we need either of these. What **OBS_ID** was originally intended to represent is not clear, and a specific **OBJECT** is typically not relevant for the LAT.

(*Foschini*) **OBS_ID** can be used to identify the observation proposal of Guest Investigator programs, that should be activated during the second and the following years of operations. The **OBJECT** would be the on-axis target. With this respect I suggest to add also 4 more keywords, namely **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ** for the equatorial coordinates (J2000) of the spacecraft X and Z axes during that particular pointing. This in the EVENTS HDU only. These keywords could be of help also during other observations to identify the centre of the field of view.

(*Stephens*) The GLAST data as planned to be distributed does not have pointings in the traditional sense and there will be a lot of data that doesn't correpsond to a proposal from the GI program. The **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ** keywords are mostly meaningless for the data as the spacecraft is constantly moving. The center of the "field of view" for the data extraction is already identify by the Data Subspace Selection (DSS) keywords. It would be possible and might be of use to duplicate this in the **OBJECT** keyword but I think the **OBS_ID** keyword would go away.

(*McEnery*) I think that the **OBS_ID** might not be very useful. It is ambiguous when an observation starts. The LAT does not go into a different mode or start a new run when we repoint, so it would not be obvious when to start tagging events with a new **OBS_ID**. Also, as Tom points out, we will likely spend most of our time in survey mode even after year 1. I think that it may be useful to have **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ** because they define the orientation of the LAT for each event.

(*Hirayama*) I don't remember why those keywords were introduced, but there are some records in topic 009 and topic 010 of the latest topics of FT1 and FT2, which might be of your interest.

(*Foschini*) The basic problem of these and other keywords is to define the boundaries of the data that are saved into a certain file. After a fruitfull email exchange with Seth and Julie about the observing modes, I suggest that it can be useful to separate the data obtained from slews and those from pointings. Also in survey mode, the spacecraft is not continuously slewing, but rocking between two pointings, lasting one orbit on each pointing. The pointing direction can change, but what is important is that also in survey mode the spacecraft is pointing toward a certain direction for one orbit. On the other hand, independently on the reason for which the spacecraft is moving (slew, repoint, other), the data acquired in this mode require a different treatment with respect to the data from pointings. So, we have a sequence of pointings-slews-pointings-slews... both in survey and GO modes.

I think that it could be useful to use the change between pointing and slew as boundary for the files: in this way, a lot of keywords, like **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ, OBJECT** and even **OBS_ID**, are automatically defined (for slews these keywords could refer to the middle value), and so that also for time keywords. This could be also useful to avoid an excessive load of the hardware for the science analysis: several single small files are better than one or few single huge files.

Please note that the **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ** keywords refer to the spacecraft (i.e. star tracker), but the boresight of the LAT should be calculated by applying a rototranslation from the star tracker position. The latter can be always improved, so that it is always better to have the starting point, that is the position from the star tracker.

(*McEnery*) I disagree that the data will be treated differently in inertially pointed and survey mode. The LAT is not pointing at a particular direction (in celestial coordinates) during survey mode, the direction of the LAT boresight is constantly changing. In pointed mode, there is no guarantee that the LAT will stay perfectly inertially pointed, it may move by a few degrees to satisfy contraints, and will move by a lot during an Earth avoidance maneuver. So even in a pointed observation **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ, OBJECT** are not automatically defined, and a science analysis cannot assume that the pointing direction is constant.

(*Foschini*) My notes were only suggestions, I did not state that there will be a different treatment, but I suggested a different treatment. If I have understood well, although the LAT has no FIFO, that creates problems during slews in other satellites (see, for example, the XMM Slew Survey), it remains open the fact that during slews it is necessary to give the attitude for every single photons. The sum of these errors in the attitude reconstruction changes the PSF size and pattern, that is worse than the case of pointed observation.

With reference to the fact that a satellite is not perfectly stable during a pointing, this is normal, but the keywords **RA_SCX, DEC_SCX, RA_SCZ, DEC_SCZ** define the boresight of the intruments and deviations from the required attitude during a pointing are tagged as bad-time intervals (the complementary to good-time intervals) and data discarded (or later reanalyzed with different methods). In other words, the basic philosophy in pointings is to have certain key parameters taken as reference and to consider deviations from them.

Proposed resolution: Drop **OBS_ID** and **OBJECT**.

5. MC_TRUTH keyword

(*Digel*) I think that the original intent was that this keyword indicate whether the data are entirely Monte Carlo truth values (i.e., actual directions, energies, etc.). We do not actually have files like this, and I propose removing this keyword. **gtobsssim** does add a column **MC_SRC_ID** to the files it generates, but this is easy enough to check for without the **MC_TRUTH** keyword.

(*Stephens*) Actually I thought this was a flag like the old **PSR_COLS** keyword and indicated that there were additional MC columns in the file. As we don't use this I think it can go away.

(*Hirayama*) Adding a new column to an FT1 file for a special case may not be appreciated by the HEASARC people. I remember they don't like "variants" of an FT1 file, mainly for archival purposes. I think their discussion was like: "for the GLAST team members (who know the file contents very well), it is just one additional column, but for the HEASARC members (who do NOT want to deal with the file contents as much as possible), it is a different file format." Probably it is better to talk to them about it again, once we have our conclusion about this topic, rather than trusting my rusty memory.

Proposed resolution: Drop the **MC_TRUTH** keyword.

6. EVENT_ID column

(*Digel*) This is specified as a 32-bit integer. Right now, most likely the event IDs will be assigned by the LAT as a time stamp. Most likely we will need more than 32 bits to represent them, and of course they should look more or less like some kind of integer representation of the value in **TIME**. *Right now, I don't know what to recommend for **EVENT_ID**.*

(*Foschini*) Yes, it appears that **EVENT_ID** is a duplication of **TIME** column. It can be removed.

(*Hirayama*) The topic is also outlined in topic 010 as well as other range problems that might interest you. Also, note that it may not be very easy to identify an event by **TIME** column after **gtbary** overwrites its contents for a barycentric correction.

Proposed resolution: Find out how the event numbers will be assigned onboard. From the ISOC-FSW ICD it isn't obvious to me. The **EVENT_ID** column will be distinct from time and should stay.

[anders] The extended GEM sequence counter (which is your event ID) is assigned onboard by FSW and it's a 64-bit number. Note that to define an event uniquely you need the combination Run ID and Event ID.

7. RECON_VERSION column

(*Digel*) **RECON_VERSION** is defined as a 2-byte integer to define the version of reconstruction (and classification) algorithm applied for a given event. This is probably more general than we need. In principle, a given events file could have events processsed with different versions of the software, but in practice, I do not expect that this will happen. I recommend that we make RECON_VERSION a header variable instead of a column.

Also, somebody needs to think of how we will translate versions of the geometry, calibration files, reconstruction, and classification, into some fairly compact representation. Certainly we will be closely keeping track of this information for Level 1 processing, and will have a database someplace that can tell us.

Proposed resolution: I'm not sure. We probably could adopt a 'v1r2p3' string for designating the version of reconstruction - say from the version of Gleam used to make the reconstruction, but I don't think that any versioning has been worked out yet for event classification.

[anders] There is also makeFT1 which is part of ScienceTools. And for completeness you may also want the version of the Halfpipe which merges and time orders the events. In any case, I suggest you explicitly use 'GlastRelease-v1r2p3' instead of just 'v1r2p3' to avoid any confusion as to what the version number refers to.

8. CALIB_VERSION column

(*Digel*) As for **RECON_VERSION** we do not need this as a column. Actually, I'd recommend combining it with **RECON_VERSION**.

Proposed resolution: Actually, on second thought, I do not recommend combining it with **RECON_VERSION**, but this is another example where I don't think we know yet how calibrations will be versioned. I'll at least ask the ISOC.

[anders] Currently we do not have any versioning of the calibration files. Also note that each calibration file is updated independently of any other. They will also have different validity periods. There is an ID number associated with each calibration type. This number currently only lives in the calibration meta-database.

9. IMGOODCALPROP, IMVERTEXPROB, IMCOREPROB, IMPSFERRPRED, CALENERGYSUM, CALTOTRLN, and IMGAMMAPROB columns

(*Digel*) These columns relate to the classifications of the events and date to DC1, in particular to implementing the Atwood cuts for background rejection. Many things came together at once in the runup to DC1 and we ended up incorporating each of these variables in the event summary file. The DC1 response functions were derived post-Atwood cuts.

For DC2, we will have a cleaner way to specify the results of the event classification, although I do not know what it is yet. We will have some kind of analog of **IMGAMMAPROB**. Ideally, we'll also have distilled other classifications into a few sets that map into response functions that we'll provide (e.g., **fro nt, back**, and **cal-only** events, with **good_pdf**, **good_energy**, or **dont_care**).

What do you recommend here?

Speaking of classification, will we also have a flag like 'HEAVY_CR'?

(*Burnett*) The "IM" prefix was inserted by me when I absorbed Bill's values. Since it stands for Insightful Miner, and it is not clear that we will always use exactly Bill's trees for this, I would suggest that we drop it from the names. "COREPROB" is confusing, it really means goodpsfprob. And there is no longer anything to correspond to IMPSFERRPED.

More practically, since there would be actually cuts on each of the IM variables to define "gammas" to get into the FT1 gamma file, those need to be clear. Should they be strict, so that the events are highly likely to be gammas, or loose, to allow a user to choose the level of contamination with respect to poorly measured energy, direction, or presense of background? Given this possibility, how many different Aeff and PSFs will we calculate.

In the case where there are multiple parallel trees for the classification analysis, the best cuts are probably dependent on which tree was used: this implies more variables, or fields in a variable, to describe which path the analysis took.

Proposed resolution: We need a discrete assignment into event classes that correspond to the response functions we ultimately adopt. Right now all we have is front vs. back, and which class an event belongs to can be determined solely by conversion layer. Having a 'gamma ray-ness' column would be tempting, too, but in principle this is already related to the event classification and we wouldn't have response functions that corresponded to further cuts on the quantity.

10. CONVERSION_POINT

(*Digel*) This was originally imagined as a way for end users to decide whether they wanted to believe whether a particular event was not a charged particle and was well reconstructed. Eventually, we will have an event display server available that will make this much easier for users, who would otherwise have to find the geometry of the LAT some place.

Also, so far we have not invented a need for filtering the data on **CONVERSION_POINT**. In principle, we may some day have a solar flare mode, for which we will pay attention only to the inner towers and layers, but this is probably much better implemented as a kind of flag (like "**INTERIOR**").

I recommend removing **CONVERSION_POINT**.

(*Burnett*) I agree. The filtering implied by this is already incorporated into the classification tree variables.

Proposed resolution: Remove **CONVERSION_POINT**.

11. PULSE_PHASE and ORBITAL_PHASE

(*Digel*) These are needed (and filled) by the pulsar timing analysis tools. I don't know whether they belong in the definition per se of the event file, because they can be added by the pulsar tools, but I don't feel particularly strongly if they stay. I do recommend that we make them floating point values instead of their current doubles.

(*Stephens*) They belong in the definition as they can be in the file, Even if they are not there in the
files delivered from the data server.

(*Hirayama*) Those columns became "permanent" to avoid multiple variants of the FT1 definition. In fact, it is not recommended (by the HEASARC FITS Working Group members) for a tool to change a file format by adding those columns. (See also my comment for "5. MC_TRUTH keyword.") Personally I like it better if pulsar tools add those columns when necessary, but I also see their point, especially thinking about archiving those files for future use. So, if we need/want to drop them, probably we should talk to them about our plan.

(*Digel*) I think that we should consider the FT1 format to define what the data servers deliver. The PULSE_PHASE and ORBITAL_PHASE columns are specific to an analysis, and would be wasted space in the server. These columns should be added to files by the pulsar tools that fill them, just as gtdiffresp adds a column for each diffuse source in the particular source model under consideration. None of these analysis-specific columns is fundamentally part of the FT1 data.

Proposed resolution: Remove the **PULSE_PHASE** and **ORBITAL_PHASE** columns. They are analysis specific, and will not be in the event summaries delivered to or from the server. Actually, Jim has pointed out that if the response functions are known and the standard diffuse emission model is also defined, the **gtdiffresp** columns corresponding to them could be precomputed and stored with the events. This is apparently a big computational savings from the user perspective, and I think should be done. But we aren't in a position now to even say how may different sets of response functions we will have.

12. SKYX/Y issue

(*Digel*) The Guidelines for Science Tools Design page lists one outstanding issue for the definition of the events file, whether to add columns that give the coordinates of the events in some coordinate projection.

The justification given for this is that with projected values of the coordinates (presumably with respect to a sensibly chosen center), tools like **ds9** can interpret and bin the events into maps, and then (possibly complicated) regions can be defined for generation of response matrices. This is an analysis path (multiple overlapping point sources in **Xspec**) that we are not intending to pursue.

Jim has pointed out that having **ds9** able to make a binned map directly with real coordinates on it (as opposed to just sky pixels) is a big convenience. Otherwise, a map first has to be generated with **gtbin**.

I don't know what to recommend. Having coordinates available in some (which?) coordinate projection would be convenient, as would, say, having them available in Galactic coordinates, but I am not sure where it is sensible to draw the line.

(*Foschini*) Surely having **ds9** able to make images directly from the event list is very useful, particularly to have a quick look. In this case, it is necessary to have full **WCS** keywords in the EVENTS header. But, for what I see the event list files are expected to be huge, so it will be necessary to have an executable able to make a selection of events centered on certain coordinates, with a certain radius of extraction, in a certain time region, within a certain energy band (that is already available). In that case, we can avoid putting the full **WCS** keywords in the EVENTS header and the executable extracting the selected photon list can add the necessary **WCS** keywords in the subset of data.

Proposed resolution: Omit **SKYX/Y** columns. Reluctantly, I also recommend that we should not include Galactic coordinates either.

(*Chiang*) I think it is probably better to draw the line **after** adding Galactic coordinates. I am often asked by people who are browsing the data using fv, ds9, ROOT, etc. why we don't have Galactic coordinates in the FT1 file, especially since in gamma-ray astronomy it is incredibly useful to know the location of a source relative to the Milkway.

Revised proposed resolution: Omit **SKYX/Y** columns. Let's include Galactic coordinates, for the reason that Jim gives. We do not have, and are not planning to provide, a coordinate conversion utility for event files, and Galactic coordinates certainly would be convenient to have available.

13. Live time

(*Digel*) Way back when (until early 2003), the events file was defined to include a column called 'Deadtime' which was to be the 'Deadtime accumulated since the start of the mission.' It was removed, partially because Masa pointed out that information that relates specifically to the LAT belong in the FT2 file. Also, live time turns out to be more convenient to work with.

I propose that we include accumulated livetime since the start of the mission as a column. It was always imagined that for very short time intervals (shorter than the update interval of the FT2 file) we would need to calculate accumulated livetimes between events in the event summary file. This also will be important for studies of solar flares and (if we are lucky) very bright GRBs, when we may be dead time limited for short periods of time.

(*Foschini*) Perhaps it can be useful to have a **DEADC** keyword with the average deadtime correction (i.e. the complementary to deadtime value) applied to the events in the file.

(*McEnery*) I agree that we should have livetime included as a column in the events file.

Proposed resolution: Add the **LIVETIME** column.

[anders] Long story short: Only the livetime per run will be available.

14. Event summary++

(*Digel*) **LS-002** is an event summary, intended to have information for higher-level analyses. The GSSC expects (reasonably) to receive this summary for every event that is telemetered to the ground (and that is not discarded early in the process as obviously background to save disk space at the ISOC).

They would also like to receive an 'extended' event summary that includes all of the variables that are used by the classification trees. That sounds fine to me, too. Right now, the specification of variables actually used by the trees has not converged, and sending the whole Analysis Ntuple would be overkill. I think that we should be able to define this data product in good detail within the next month.

Proposed resolution: We may be able to define now the additional variables that are actually used in the classification(s).

15. Time Issues

(*Foschini*) I suggest to adopt the days as time unit, that is of immediate use for scientist. This means to change the **TIMEUNIT** value to 'd'.

I suggest also to add to the EVENTS and GTI header two keywords indicading the on-board time value corresponding to the first and last event in the file. That is, the keywords **TSTART,TSTOP** should have corresponding keywords **OBTSTART,OBTSTOP**. This can be useful if the time correlation is missing or with errors.

I would add also a keyword **TELAPSE**, namely just **TSTOP-TSTART**, both in the EVENTS and GTI header, just to say the time extension of the whole observation. This is not the exposure or the sum of the GTI, but simply the whole uncorrected time of observation.

Two more keywords can be useful for timing accuracy: **TIERRELA,TIERABSO** see The Proposed Timing FITS File Format for High Energy Astrophysics Data.

(*Hirayama*) I don't think changing **TIMEUNIT** value from 's' to 'd' is a good idea. Almost all times are expressed in units of seconds, with an exception of **MJDREF** (which is in days), in event files for various astrophysics missions as long as I know. Also, according to Glossary of Keywords commonly used in OGIP FITS Files, **TIMEUNIT** governs **TSTART**, **TSTOP** and **TIMEZERO** keywords, but not **TIME** column (which is governed by **TUNITn** keyword). That means **TSTART** cannot be directly compared with a **TIME** column value, for example, if **TIMEUNIT**='d' and **TUNITn**='s'. It is technically feasible, but rather confusing. **TUNITn** can be 'd' to avoid the confusion, but then **TELAPSE** (and some other keywords) are in units of seconds no matter what. Looking at definitions of those other keywords and columns, it seems to me that 's' is a more natural choice for **TIMEUNIT** than 'd'.

For **TELAPSE**, **OBTSTART**, and **OBTSTOP** keywords, see my comment on "17. GTI."

Proposed resolution: Keep the time units as seconds. I need to look into **TIERRELA** and **TIERABSO** still.

[anders] Minor detail: The start and end time of the run (ultimately deciding the livetime) may not be the time of the first and last events. The reason is that the instrument is alive for a short time before the first event and also after the last event. This is not captured in the livetime counters. I may have a solution to this using 'sweep' events which are issued just before and after the first and last physics triggers. The advantage is that there is a fixed and calculable time between the sweep event and when the instrument is alive/dead (independently of when the first trigger arrives).

16. Name of tables

(*Foschini*) Presently there is only one keyword to identify the template of EVENTS and GTI data structure. To take into account that templates can change (particularly during the first months after the launch), it is better to add a keyword **EXTVERS** or something similar with the version number of the used template. Moreover, the data structure of LAT and GBM can require different keywords, so I propose to give at the **EXTNAME** more complicate names to take into account the different uses: for example, for the LAT events there can be **EXTNAME**='GLAST-LAT_EVT' and the corresponding GTI can be **EXTNAME**='GLAST-LAT-GTI'.

(*Ballet*) I don't think it is a good idea to change the names. The INSTRUME and TELESCOP keywords are there to indicate which intruments we are dealing with. Better keep standard names for the EVENTS and GTI extensions, as was done for previous missions such as Chandra or XMM-Newton. I agree to the EXTVERS suggestion.

Proposed resolution: Keep the extension names as **EVENTS** and **GTI**.

17. GTI

(*Foschini*) I suggest to add a keyword **GTI_NAME** to the GTI header to explain the origin of GTI: for example, there could be GTI due to attitude, telemetry, or other. The final GTI can be a merging of the whole types of GTI.

The keyword **TELAPSE** in GTI should indicate the elapsed time of the whole observation (see the notes at n. **15 Time Issues**) and not the difference of GTI STOP-START.

I suggest also to add two more columns in GTI table, **OBTSTART,OBTSTOP**, with the onboard time corresponding to **START,STOP** columns.

Perhaps it could be useful also to add two more columns with START and STOP in UTC.

Note: the keyword **HDUCLAS2** for GTI header (if kept, see n. **2 HDUCLASS keywords**) I think should be 'STANDARD'.

(*Hirayama*) About the **TELAPSE** keyword, I remember somebody (most likely a member of the HEASARC FITS Working Group) told us the definition is "time between START of the first GTI and STOP of the last," which is the current FT1 definition. However, I couldn't find such statement on the web. Instead, HFWG Recommendation R11 states "**TELAPSE** is the time interval (in seconds) obtained as difference between the start and stop times of an observation." It sounds like they assume a pointed observation, as usual, where the assumption is not quite appropriate for GLAST. My guess is that, when they explained the definition to us, they gave us a traditional explanation that works for pointed observations, without much consideration on a continuously scanned observation that GLAST will perform, although we should confirm it with the HEASARC people before we conclude so.

If my understanding is correct, I would agree that **TELAPSE** = **TSTOP** - **TSTART** and should apppear in an EVENTS extension, too. In the current definition, it shows up only in a GTI extension because it is defined as a derived quantity from the contents of a GTI extension.

For **GTI_NAME**, if we need it at all, it should cooperate with DSS keywords, I think. In order to compute and revise GTI's upon data subselection, not all DSS keywords will be taken into account of. So, one solution could be (although I don't think it is pretty) to list DSS keyword numbers that are used to compute GTI's in **GTI_NAME** keyword value.

For **OBTSTART** and **OBTSTOP**, I don't think it is a good idea. Noting GTI's will change upon data subselection, a data subselection tool must update **OBT START** and **OBTSTOP** keyword values, too. The GLAST tools can be modified to do so, but external tools such as XSELECT will not. At the least, that necessitates a special handling of a GLAST FT1 file, wihch is different from event files of other astrophysics missions.

I think we should adhere to a standard GTI format, unless a deviation from it will significantly improve users' benefits.

For **START** and **STOP** in UTC, I simply don't know how to do it. A moment in time in the UTC system cannot be expressed by a single number because of leap seconds, if I understand correctly. Also, for the same reason as for **OBTSTART** and **OBTSTOP**, I don't think inventing a new GTI extension format is not a good idea, either.

(*Foschini*) The **OBTSTART,OBTSTOP** keywords are useful only in case of problems, rather than for a direct use. That is, if there are problems in time correlation or anything else, the only way to reconstruct on ground the events sequence is to have the on board time (that is the only direct measurement of time of an event) and restart again the conversion to user friendly time values. So it is just to have a backup option.

For UTC keywords, these are generally expressed as a string: e.g. "2005-11-22T11:43:00". In this case, it is just for the end user, to have something more friendly than JD or anything else. To make the conversion, it is used a time correlation, that is a function that linearly correlates the UTC to the onboard time.

Proposed resolution: There's a lot here. I don't feel strongly about **TELAPSE** but I recommend that we do not adopt it because as Masa pointed out, it is not so relevant for scanning observations. I need to think more about **OBTSTART** and **OBTSTOP**. I am reminded that according to the letter of the law for GTIs, our servers should return corresponding GTIs for each query.

18. CONVERSION_LAYER

(*McEnery*) I think that we should define this so that 0 is the bottom (thick) of the tracker and 15 is the top (thin) layer. This is consistent with the definition used in the reconstruction code, but is opposite to the current definition (where 0 is the top of the tracker).

(*Chiang*) If we keep CONVERSION_LAYER as a column, then I would go even further and include in the numbering scheme the tracker layers that do not have radiators. This would help guard against any ambiguity regarding where the conversion occurred since we would be using the same layer ids that are used in the reconstruction code. However, ultimately, we may choose to omit the CONVERSION_LAYER information in favor of an EVENT_CLASS variable.

Proposed resolution: Omit **CONVERSION_LAYER** (see below).

19. EVENT_CLASS

(*Chiang*) There would be a one-to-one correspondence between IRF "subtypes" and possible EVENT_CLASS values. Presently, there are only two possible event classes, Front and Back, indicating in which set of radiators the conversion occurred. As the instrument calibration and IRF implementation evolve, there will likely be more classes than just two. An advantage of using EVENT_CLASS over something like CONVERSION_LAYER is that such a scheme helps ensure that the user will only be able to make cuts that are supported by our IRF implementations.

Proposed resolution: Include **EVENT_CLASS** as Jim describes in place of **CONVERSION_LAYER** and as a way to encode other classes that we come up with. After discussion with Jim, I think we should also have a column called something like **CONVERSION** that would indicate, e.g., by value 1,2, or 3 whether the event is FRONT, BACK, or CAL-ONLY. This would be in addition to **EVENT_CLASS**, although until we have other ways defined to discriminate other kinds of response functions it will have the same value.

(*30 June 2006)* Here are a few more issues that have arisen in the run-up to Ground Readiness Test 5, which has the ISOC transferring a realistic FT1 file to the GSSC, who will verify that it is consistent in every testable way with the detailed specifications in the Science Data Products File Format Document (Word file). The FFD has not been finalized and several loose ends have been recognized regarding the specification of the FT1 headers and columns.

20. Precision of time in TSTART/TSTOP

From a note by Tom Stephens:

Unknown macro: {bgcolor}

An interesting question came up as we were working on testing our next software release concerning the precision of the values in the FITS file headers related to time. In the data tables of all the files, columns related to time are stored as double precision which gives us 15 significant digits (microsecond precision) which is the requirement. I've never seen any such specification for the keyword values in the headers. (Although I would assume it should be the same.) The problem has to do with rounding. Here's what we found.

As part of our data ingest system, we validate the incoming FITS files. One of those checks is to make sure that the time values in the tables all fall within the specified TSTART and TSTOP values given in the header. In several files we were seeing things like the following:

In the header we have:
TSTART = 1.540365873394E+08 => 154036587.3394

In the data we have
TIME = 1.54036587339366E+08 => 154036587.339366

which is the same if we round the data to the ten thousandths of a second but since both are read into a double precision variable, a simple comparison causes the file to fail verification since the data time is earlier than TSTART. Effectively the files are correct to the level or precision in the header but not to the level of precision in the data.

The GSSC can only check the files to the level of precision in the header keywords, so my question is what level of precision should we have there? It doesn't really matter to me what the answer is, but we need to know to develop the software and I think it should be consistent across all the data files created by both LAT and GBM.

(*Digel*) The consensus seems to be just to increase the number of decimal places in the ASCII representations of TSTART and TSTOP in the header to reach the microsecond level. This is probably good enough, although we are right at the limit of precision of doubles and with floating point representations, if a comparison comes down to the last digit of precision, it can be hit or miss.

Actually, the particular example that Tom cited was from an FT1 file that was generated before we took care about setting TSTART and TSTOP properly. **makeFT1** was given no information about those values and so just used the actual times of the first and last events in the file. In DC2 we set the TSTART and TSTOP values to be integal numbers of seconds, but even in the flight data when runs and downlinks start and end at whatever times they do, we'll only rarely have an event within a microsecond of TSTART or TSTOP for a given downlink or run.

Proposed resolution: Modify **makeFT1** to write TSTART and TSTOP to 6 decimal places.

21. Names of columns for diffuse responses

(*Digel*) For analyses with **gtlikelihood** with models that diffuse emission terms (i.e., most unbinned analyses with **gtlikelihood**), the overall execution time can be sped up with precomputation of parts of the likelihood function using **gtdiffresp**. This tool writes new columns to the FT1 files that it processes. The columns are named for the response functions used and the names of the diffuse emission terms in the source model (XML) file. For DC2, the diffuse responses were precomputed in this way.

Tom Stephens has pointed out that the resulting column names (**DC2::Extragalactic Diffuse** and **DC2::Galactic Diffuse** for the files that we made for DC2) violate HEASARC standards in a couple of ways. (Note that in these column names DC2 stands for the set of response functions used.) The standards do not allow spaces or colons; the only non_alphabetic or numeric character allowed is '_'.

The space in the source name is easy to get rid of, by using a source model XML file that gives the diffuse sources names without spaces. Actually, in anticipation of the Galactic diffuse emission model being updated fairly often, the names should include a version number anyway, like GalacticDiffuse_v0.

The delimiter between the response function name and the source name can be changed as well (at the startup cost of breaking the use of existing diffuse response columns or maybe writing a tool to change column names in existing FT1 files). Jim confirms that whatever reads the diffuse responses constructs the column name that it is looking for and checks to see whether the FT1 file contains it. For this '_' is just as good as '::'.

We should go ahead with making the change to the delimiter that **gtdiffresp** writes and **gtlikelihood** assumes in the diffuse response columns, but because the response functions and standard diffuse models will be changing fairly frequently between now and at least the first year of the mission (making existing diffuse response precomputations only of historical interest, I think that we should not include any diffuse response columns in the FT1 files that we send to the GSSC from here on out.

Proposed resolution: Change the response function-source name delimiter from '::' to '_'. In future, omit diffuse responses in FT1 files that are delivered to the GSSC.

22. AUTHOR, CREATOR, ORIGIN, SOFTWARE and VERSION keywords

(Digel) We are currently carrying these keywords in the definition of the primary header for the FT1 format used by the science tools (ft1.tpl), but are not usually assigning values to them. In order to pass the verification test of GRT 5, we will either have to remove them or give them (meaningful) values. Here are my suggestions:

AUTHOR - According to the HEASARC recommendations for the CREATOR keyword, AUTHOR is supposed to be a citation of a publication related to the contents of the file. This should be removed; I think that David has already done this in the FFD.

CREATOR - This is supposed to be the name and version of the program that created the file. In the case of FT1 files that are delivered to the GSSC the name would be 'makeFT1'. I suppose that the best choice of version number would be the version of fitsGen (currently v2r4), so the CREATOR string would be 'makeFT1 v2r4'. Can **makeFT1** determine its own version number? Otherwise, I think that having to manually edit the template file would be a maintenance problem and the better solution would be to omit CREATOR.

ORIGIN - This should be 'LISOC' for the files that **makeFT1** writes.

SOFTWARE - This is defined as the version number of the tool that generated the file; if **makeFT1** can determine what its version number is, then we should have it set SOFTWARE accordingly.

VERSION - This is defined as the release version of the file. We used this for the first time with the regenerated DC2 data (assigning them VERSION = 2). I don't think that **makeFT1** currently has a way to know what version number to assign. It should optionally take a version number as input, and if none is specified, should set VERSION = 1 in the output file.

These issues are resolved. See STGEN-30. [Jim pointed out that **SOFTWARE** is redundant with **CREATOR**. After checking to be sure that HEASARC did not care about this keyword, **SOFTWARE** has been removed from the definition of this data product.]

---

20 November 2007

The current version of the FT1 template is here. Since the last update 13 months ago we have gained more experience with the kinds of analyses we be making with the FT1 files and the L1 pipeline has reached an advanced stage of development. Some aspects of the FT1 files need to be updated as a result.

23. GPS_OUT

(Digel) In the current iteration of the definition, we had assumed that whether or not the absolute times are based on the 1 pulse per second signal being synched with the spacecraft's GPS clock could be indicated with one flag in the header of the file. Anders points out that we really should be carrying this as a column in the FT1 files because the answer can be different for different events. **GPS_OUT** as a header keyword could be interpreted to indicate whether **any** event in the FT1 file had a time based on the internal oscillator, but that in itself would not be particularly informative.

For reasons that I don't quite grasp right now, correcting event times after GPS lock has been re-established for oscillator drift during periods when lock was lost is not really practical. Also, except in extreme circumstances (e.g., the fastest ms pulsar considered over a very long time range) the potential drifts are probably negligible. So the **GPS_OUT** flag being set for a given event would be just a warning to be careful.

Anders also suggests that we consider including the number of seconds since GPS lock was lost as a way of allowing users to estimate the relative qualities of the time accuracy. These times could range up to thousands of seconds in the worst case

I propose that we make **GPS_OUT** be a column as a 2-byte integer; 0 means that the event time was assigned while the GPS system was in lock. Values >=1 indicate the number of seconds since lock was lost. Anders points out that this quantity is always an integer, the number of 1 pulse-per second counts lost.

24. Replacing RECON_VERSION and CALIB_VERSION

(Digel) These columns are 2-byte and 3x 2-byte quantities that were intended to represent the version of Gleam and the versions of the ACD, CAL, and TKR calibrations (however exactly they are defined) that were used in the processing.

I propose to replace these with a single 2-byte integer **PIPELINE_VERSION** that is indexed to the configuration of the L1 pipeline (versions of Gleam and other software and versions of the calibrations) that were active at the time that a given event was run through the pipeline. When the L1 pipeline is under configuration control, these versions will be carefully tracked.

Anders: "We don't really have such a uber-version number yet. But we do need one."

With reprocessing, we'll want a way to tell whether a given event or file of events are in their most current processing versions. We could maintain a table or graph of most current PIPELINE_VERSION vs. event time.

25. Revisiting CONVERSION_TYPE and EVENT_CLASS

(*Digel*) These are 2-byte integer columns that define whether an event converted in the front or back section of the tracker and what event class the event satisfies.

As we currently manage the response functions, front and back are *de facto* different event classes. The **EVENT_CLASS** column is intended to make other distinctions, such as classes A and B for DC2. Right now for Pass5 we have still another column (**CTBCLASSLEVEL**) that is used to specify the event classes - so **EVENT_CLASS** is not actually being used.

What should do here depends in part on whether we rearrange the Pass 5 IRFs so that each event can be a member of one and only one event class; currently that is not the case.

In the future we can look forward to having still different sets of response functions - obviously - that will correspond to different versions of reconstruction and classification. In principle we could keep incrementing **EVENT_CLASS** (if we had a table someplace that mapped the response functions and processing versions into **EVENT_CLASS** values). Or we could have some kind of string column related to the name of the IRF that the event qualifies for. Or we could not try to protect people from themselves and let **EVENT_CLASS** have different meanings for different sets of response functions (possibly including a header keyword to name the response function family that applies to the file).

I don't have a recommendation about which option would work best.

---

19 August 2008

26. Including GEOMAG_LAT

(*Digel*) This is a quantity that depends on position of the spacecraft and time, and so years ago was relegated to the FT2 file. The thinking at the time was that filtering on geomagnetic quantities could be done with GTIs using **gtmktime**. Now that geomagnetic latitude appears to be of more direct interest to the analysis, having it available in FT1 files would be a convenience. The quantity would not have to be geomagnetic latitude per se - the current implementation derives it from McIlwain L - but either one or the other should be in the FT1 files.

(*Jim C*) If this is needed for correlation studies with other FT1 quantities, then I suppose there is a reason to have it here, although it would make more sense just to use the merit file directly. However, if this is needed for making selections for ranges of geomagnetic latitude for use in analyses with IRF-based analyses, whether using the ScienceTools or not, then one really should use gtmktime, since the GTIs enable one to compute the livetimes.

(*McEnery*) Geomagnetic latitude is already included in the event data definition (which is FT1 + a bunch of extra variables). Someone who wants to work with a fits file (and the FT1 quantities) could just use the event data directly.

---

29 June 2010

27. Including PROC_VER keyword

(*Digel*) This is summarizing what was worked out between Maria-Elena and Don Horner to allow the FSSC to handle receiving reprocessed data without interrupting ingest of output from the L1 pipeline. This 'dual ingest pipeline' scheme will allow reprocessed data to be transferred in the background, for little or no downtime when a switch is made to a new version. Incidentally the naming scheme of the FT1 files is also being modified to include the processing version as part of the name; that change is not covered here but will be documented in the File Format Document. The addition would go in LS1 as well. The keyword will go into the primary headers of the files

The value starts at 0 and will be incremented (perhaps not necessarily sequentially) in reprocessings.

This has been implemented in the File Format Document

28. Including PASS_VER keyword

(*Digel*) This keyword would have a numerical value indicating the analysis pass used to classify the events (e.g., Pass 6 or Pass 7.3), and would be used by the Science Tools to avoid mixing FT1 files with different analysis pass values. Like the above, this keyword would be implemented in a reprocessing. Initial thinking had been that this would need to be a new column, but Jim is thinking that having the Science Tools make a consistency check using just a header keyword (e.g., if given a list of input files) would not be difficult. This keyword would be set by makeFT1 and the value to assign would be passed to makeFT1. It would go in the EVENTS extensions of FT1 (and also LS1) files

(*JC*, 09Jul2010) This keyword will probably have a string value rather than be a number. It will be taken from the xml definition for the event classes.

This has been implemented in the File Format Document (22 July 2010 update, pdf to current version . Note the complete lack of version control on this site).

---

22 July 2010

27. Making EVENT_CLASS a long integer

(*Digel*) This is just to note that for Pass 7.3, EVENT_CLASS will need to be changed to a 4-byte integer (J in FITS binary table format) from a 2-byte integer (I). This is because in Pass 7.3 the event class designator will become a bit mask in order to allow complete flexibility in specifying which classes each event belongs to. With Pass 7.3, the event classes will not all be nested, so simple range selections on EVENT_CLASS will no longer work.

This has been implemented in the File Format Document (22 July 2010 update, pdf to current version . Note the complete lack of version control on this site).