Changes to support Pass 8

Introduction

With Pass 8, event_types are introduced. Even though these are essentially a generalization of conversion_type (i.e., front vs back converting events) in that the event types partition the data into distinct subclasses which have their own effective area, point spread, and energy dispersion functions, in Pass 8, there are alternative partitions for a given class. For example, the Pass 8 SOURCE class selection can be partitioned in one of three ways: into front vs back converting events, into events belonging to one of 4 "PSF" subclasses, or into events belonging to one of 4 "EDISP" classes.

Pass 8 Descriptions

The Pass 8 event_class and event_type descriptions and usage are described in the following links.

- Pass 8 Event Class Definitions
- Pass8 Data Analysis Reference Page

Changes to gt-tools

- gtobssim:
  - The event type partition has been added as an option for gtobssim. Examples for P8R2_SOURCE_V6:

```
gtobssim irfs=P8R2_SOURCE_V6 evtype=none
gtobssim irfs=P8R2_SOURCE_V6 evtype=PSF
gtobssim irfs=P8R2_SOURCE_V6 evtype=EDISP
```

    evtype=none is the default, in which case the application will use the front/back partition, consistent with pre-Pass 8 behavior.
  - For evtype=[PSF,EDISP], DSS keywords specifying the event_type partition that was used will be written to the FT1 files that are produced. These keywords will show the bit pattern that would be used to mask to obtain the corresponding partition, e.g., for evtype=PSF:

```
DSTYP2: BIT_MASK(EVENT_CLASS,128,P8R2)
DSUNI2: DIMENSIONLESS
DSVAL2: 1:1

DSTYP3: BIT_MASK(EVENT_TYPE,60,P8R2)
DSUNI3: DIMENSIONLESS
DSVAL3: 1:1
```

    Here 60 = 0b111100, the binary representation of the mask with bits 2-5 set. Note that the event_type DSS keywords will be omitted for the front/back partition.
  - Since there is no information in the IRFs on cross-membership between event_type partitions, only bits for the selected partition will be set in the FT1 file. This means that choosing event_types in gtselect for a different partition will result in no events passing the cut.
- gtselect
  - The evtype option has been added, and DSS keywords indicating this selection will be written.
  - For both evclass and evtype, the user must specify the integer corresponding to bit-mask that selects the desired events within a **single** partition:

```
gtselect evclass=128 evtype=3     # P8R2_SOURCE_V6, front and back (i.e., all) events
gtselect evclass=128 evtype=1     # P8R2_SOURCE_V6, front-only events
gtselect evclass=128 evtype=60    # P8R2_SOURCE_V6, PSF[0-3] (all) events
gtselect evclass=256 evtype=960   # P8R2_CLEAN_V6, EDISP[0-3] (all) events
```

    The evtype=[3,60,960] selections all return all of the events, but they also specify the irfs corresponding to the event_type partition selected. This information is written to the DSS keywords and read by downstream tools.
  - One can also do

```
gtselect evclass=none evtype=none
```

    in which case, no event_class or event_type cuts are applied and the DSS keywords associated with those cuts would not be added. This will allow one to use an FT1 file from the astroserver as the common input file for different evclass and evtype selections.
  - evtype selections that cross partitions can be specified, but the behavior of the downstream tools (i.e., which irfs are used) is undefined. At some point, we will add code to guard against invalid selections.
- gtdiffrsp
  - evtype has been added as a hidden parameter.

    - If DSS keywords are in the input FT1 file (e.g., from gtselect or gtobssim), then gtdiffrsp will read those keywords to determine which event_type partition to use for computing the diffuse responses. If no DSS keywords are present, then the evtype option will used to determine the partition. If it is omitted, then front/back will be used.
    - The value of evtype must be the integer representation of the full bit-mask corresponding to the desired partition, i.e., evtype=[3,60,960] for F/B, PSF, and EDISP, respectively.
    - The DSS keywords will be updated with the evtype selection and this will indicate the event_type partition used in computing the diffuse response columns.
- gtexpcube2
    - If the cmap argument is missing, the DSS keywords from that input file will provide the information to determine the event_class and event_type selections.
    - The user can override the geometry of the cmap file. This will allow exposure maps larger than the cmap to be specified. The energy planes will be read from the cmap file, so these would no longer need to be specified by hand, but they could still be over-ridden.
- gtirfs
    - The output will be expanded to indicate event_type partitions:

```
$ gtirfs

<...output for P7* irfs...>

P8R2_CLEAN_V6 ( = P8R2_CLEAN_V6::BACK + P8R2_CLEAN_V6::FRONT )
P8R2_CLEAN_V6 (EDISP) ( = P8R2_CLEAN_V6::EDISP0 + P8R2_CLEAN_V6::EDISP1 + P8R2_CLEAN_V6::EDISP2 +
P8R2_CLEAN_V6::EDISP3 )
P8R2_CLEAN_V6 (PSF) ( = P8R2_CLEAN_V6::PSF0 + P8R2_CLEAN_V6::PSF1 + P8R2_CLEAN_V6::PSF2 +
P8R2_CLEAN_V6::PSF3 )
P8R2_CLEAN_V6::BACK
P8R2_CLEAN_V6::EDISP0
P8R2_CLEAN_V6::EDISP1
P8R2_CLEAN_V6::EDISP2
P8R2_CLEAN_V6::EDISP3
P8R2_CLEAN_V6::FRONT
P8R2_CLEAN_V6::PSF0
P8R2_CLEAN_V6::PSF1
P8R2_CLEAN_V6::PSF2
P8R2_CLEAN_V6::PSF3
P8R2_SOURCE_V6 ( = P8R2_SOURCE_V6::BACK + P8R2_SOURCE_V6::FRONT )
P8R2_SOURCE_V6 (EDISP) ( = P8R2_SOURCE_V6::EDISP0 + P8R2_SOURCE_V6::EDISP1 + P8R2_SOURCE_V6::
EDISP2 + P8R2_SOURCE_V6::EDISP3 )
P8R2_SOURCE_V6 (PSF) ( = P8R2_SOURCE_V6::PSF0 + P8R2_SOURCE_V6::PSF1 + P8R2_SOURCE_V6::PSF2 +
P8R2_SOURCE_V6::PSF3 )
P8R2_SOURCE_V6::BACK
P8R2_SOURCE_V6::EDISP0
P8R2_SOURCE_V6::EDISP1
P8R2_SOURCE_V6::EDISP2
P8R2_SOURCE_V6::EDISP3
P8R2_SOURCE_V6::FRONT
P8R2_SOURCE_V6::PSF0
P8R2_SOURCE_V6::PSF1
P8R2_SOURCE_V6::PSF2
P8R2_SOURCE_V6::PSF3

<...output for remainder of P8R2 irfs...>
```

CALDB changes

- CALDB/data/glast/lat/bcf/irf_index.fits
- CALDB/data/glast/lat/caldb.indx
- G25, DC1, DC2 irfs will be removed
- HANDOFF (Pass 4) through Pass 6 irfs will be removed unless there is a real need to retain them.
- The files in the bcf/[ea,edisp,psf] subdirectories will be aggregated by event_class and event_type partition. A script to do this for the P8_*_V5 irf files is here. irfs/handoff_response will need to be updated either by adding a generalized version of that script or by modifying makefits directly.
- The current custom IRFs mechanism (using the CUSTOM_IRF_DIR and CUSTOM_IRF_NAMES env vars) will no longer work. The foreseen alternative would be to have a modified CALDB directory tree that would be pointed to by setting the CALDB env var.

Tracking Development

- There is a trello board for coordinating this work. Send your trello userid to jchiang@slac.stanford.edu for access.
- JIRAs will also be created for the relevant items.