GLAST CAL Software Tasks
12 Nov 2002

1) Calibration and Testing
   a) Develop calibration algorithm infrastructure so both GSE and SAS developed algorithms can be used in both environments
      i) Requires root scripts called by Python and vice versa
      ii) Requires intermediate data formats in each case
      iii) Requires analysis and i/o code to be kept separate
   b) Develop muon calibration analysis suite
      i) Already available in C++
      ii) Test case for shared calibration algorithm scheme
   c) Develop charge injection calibration analysis suite
   d) Improve light taper model
      i) Current best model results in asymmetry curvature of wrong sign
      ii) Use EM crystal measurements to either verify current model or derive new model
      iii) Implement new model for use in CALDigi and CALRecon (see 2 below)
      iv) Investigate effect of computing nonlinear taper function from first and second moment information. More accurate to use terms up to squared of series expansion of nonlinear function?

2) Simulation
   a) Automate CAL test plot production
      i) CAL performance review plot set originally produced manually
      ii) Produce similar plots with automated scripts
      iii) Allows easier playing with parameters
   b) Continue to update CAL geometry

3) CAL Digi
   a) Implement Integral Nonlinearity in CALDigi
      i) Lookup table (i.e. vector of pairs of values) in each case
      ii) Denser at lower energies?
   b) Implement CAL calibration classes in TDS
      i) May want to split classes for digi from classes for recon to allow use of simulation to investigate failure detection
ii) If classes split between digi and recon, default is data are identical
c) Implement i/o to load CAL calibration TDS classes from calibration file
   i) If “official” calibration storage not available, implement temporary simple format
   ii) When “official” calibration storage format is available, transfer to it
   iii) When metadata database available, integrate database query and update with calibration file i/o
d) Modify CALDigi to use TDS calibration classes for calibration data
   i) Apply “flight-like” calibration data i.e. unique for each crystal
   ii) If flight-like data not available, replicate single crystal data to serve for all crystals
   iii) Use for pedestals, gain, integral nonlinearity, light taper parameters, etc
e) Modify CALDigi, TDS classes to store energy deposited in diodes
f) Continue implementation of infrastructure to simulate failure modes
   i) Also want to simulate ability to detect failures
   ii) Need ability to turn off components realistically
   iii) Need ability to modify calibration constants and track modifications

4) CAL Recon
   a) Implement light taper models as Gaudi tool in CALRecon
      i) Allows joboptions selection between multiple models
      ii) Already done in CALDigi
      iii) Make tool set in CALRecon match that in CALDigi (currently linear and const+exponentials)
      iv) Allow different choices for Digi and Recon
   b) Implement Integral Nonlinearity in CALRecon
      i) Lookup table (i.e. vector of pairs of values) in each case
      ii) Denser at lower energies?
   c) Investigate use of log(L/R) asymmetry rather that ratio of differences for exponential taper function
d) Modify CALRecon to use TDS calibration classes for calibration data
   i) Apply “flight-like” calibration data i.e. unique for each crystal
   ii) If flight-like data not available, replicate single crystal data to serve for all crystals
   iii) Use for pedestals, gain, integral nonlinearity, light taper parameters, etc
e) Modify single crystal position recon to forbid unphysical position result/ Deal with energy deposition in diode
   i) Currently, recon detects this and makes position at crystal center
   ii) Determine most likely crystal end with diode deposition (usual cause of this problem) and determine energy from opposite end using light taper
   iii) Ultimately, use more sophisticated solution, perhaps using tracker info, clustering, and/or other outlier detection. Range of actions from current solution to throwing out crystal for that event

f) Improve shower profile fitting model
   i) Disagreement between model and data at 1GeV
   ii) Off axis incident angles. Need to deal with fact that each CAL layer samples a range of shower longitudinal distances. This introduces dependence on radial profile (see Atwood work?)
   iii) Evaluate and refine Atwood geometric tower edge loss technique.
      (1) Where does energy lost in gaps go? Actually may reappear (at least in part) in subsequent layers
      (2) More realistic gap material density
      (3) Improve shape of shower model projected onto layer (oval instead of circle)
      (4) Improved radial profile (vary with longitudinal depth)
   iv) Coping with tower edge effects. Does energy deposited in diodes compensate for energy lost in gap material (“last layer correlation” from tower face)?

g) Recompute last layer correlation parameters for current CAL geometry

h) Implement energy recon uncertainty calculation
   i) Infrastructure first, then algorithm for each reconstruction technique
   ii) Details will need to be redone when algorithms modified or recomputed
i) Implement position recon uncertainty calculation

j) Implement direction recon uncertainty calculation

k) Design and implement modifications to recon classes to handle uncertainties

l) Implement improved energy recon at low energies
   i) Use improved TKR energy deposition estimate
   ii) Use Atwood or TKR team algorithm?
   iii) TKR deposition must be but in TDS from TKRRecon (?)

m) Implement clustering algorithm for CALRecon
   i) Fuzzy clustering implementation?
n) Study uses of CAL clustering
   i) Rejection of outlier events in shower identification
   ii) Aid tracking of showers in CAL across tower boundaries
   iii) Identify multiple events
o) Review CALRecon class structure
   i) Reimplement with abstract classes?
   ii) Review TDS class design

5) Management
   a) Revised CAL software plan
   b) CAL geometry document
   c) Failure mode simulation test plan
      i) Should include required failure simulations
      ii) Should include plan for studying failure detection
   d) CAL software requirements document