calibGenCAL Handbook
Version: v3r7

Zach Fewtrell
Aug '04, 2005
Revised: June 8, 2006

Requirements

This manual assumes you have the following software modules in working order. In some circumstances you can get away w/ recent or similar versions, but this is the configuration that I have tested the software under...

```
1)  Windows XP
2)  EngineeringModel v5r0608p3
3)  calibGenCAL v3r7
4)  MRvcmt v0r18
5)  Python 2.4.1
   a) Numeric 23.8
   b) 4Suite-XML
```

calibGenCAL does run on Linux, but that is not covered in this guide.

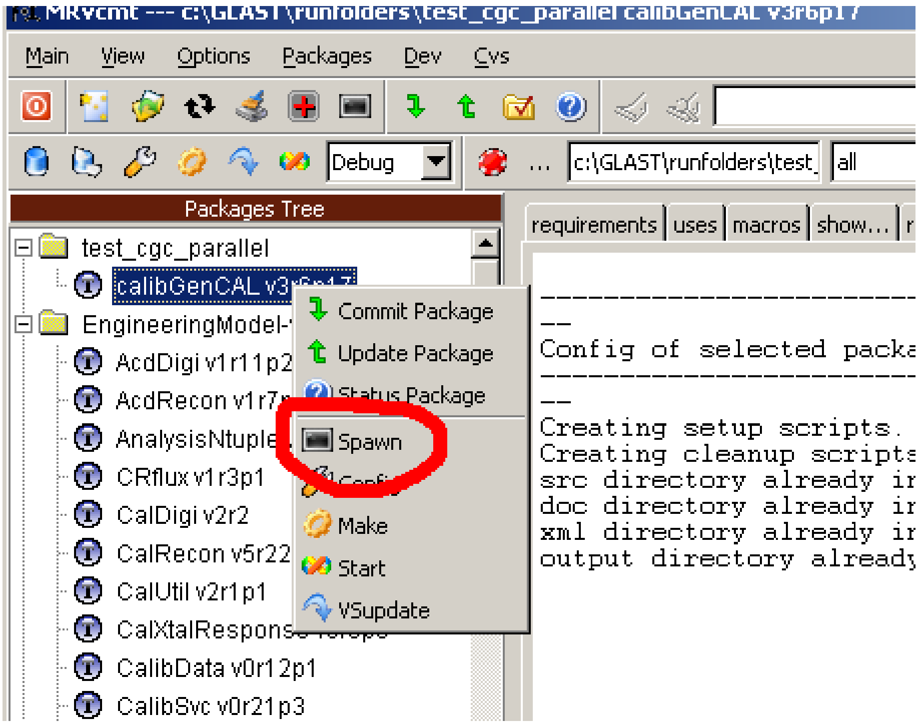
Launching Applications & Scripts

The calibGenCAL CMT package is set up so that all component scripts and applications will be accessible to the system PATH environment variable once a 'cmt config' has been performed for the package.

The simplest way to do this is via the 'spawn' button in MRvcmt.

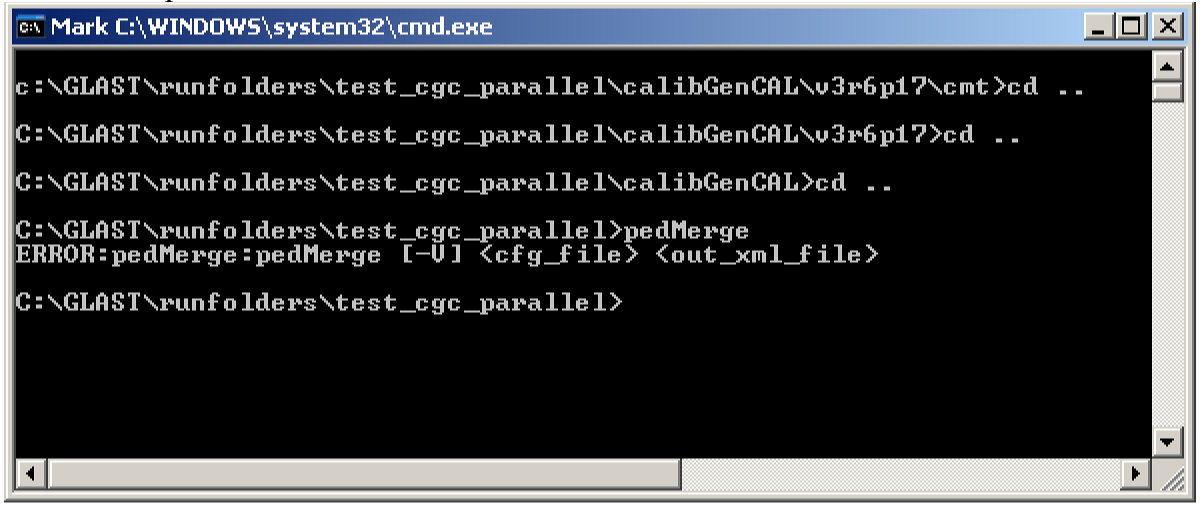
Note: You will need to properly ‘config’ and ‘make’ the calibGenCAL package and its dependents before starting. You only need to do this once. You will need to do it again if you change package versions or paths.

The 'spawn' button will spawn a system shell with the CMT environment fully configured.
The 'spawn' shell can be launched by right clicking the package name in MRvcmt.



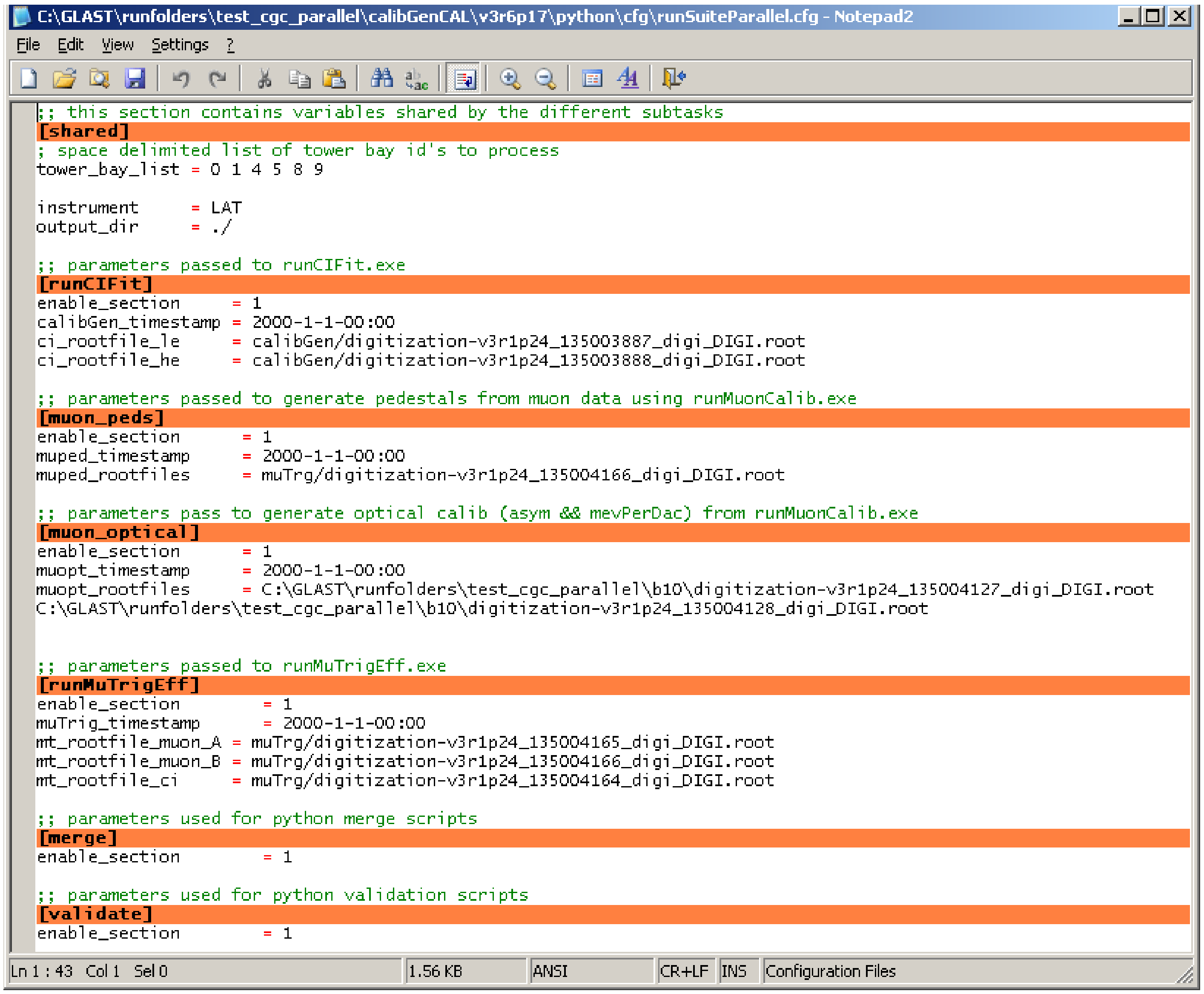
```
--------------------------
--
Config of selected packa
--------------------------
--
Creating setup scripts.
Creating cleanup scripts
src directory already in
doc directory already in
xml directory already in
output directory already
```

Once the shell has been spawned, the user may execute any calibGenCAL application or script simply by typing in the name of the script



```
c:\GLAST\runfolders\test_cgc_parallel\calibGenCAL\v3r6p17\cmt>cd ..

C:\GLAST\runfolders\test_cgc_parallel\calibGenCAL\v3r6p17>cd ..

C:\GLAST\runfolders\test_cgc_parallel\calibGenCAL>cd ..

C:\GLAST\runfolders\test_cgc_parallel>pedMerge
ERROR:pedMerge:pedMerge [-V] <cfg_file> <out_xml_file>

C:\GLAST\runfolders\test_cgc_parallel>
```

runSuiteParallel


C:\GLAST\runfolders\test_cgc_parallel\calibGenCAL\v3r6p17\python\cfg\runSuiteParallel.cfg - Notepad2
File Edit View Settings ?
Ln 1 : 43 Col 1 Sel 0
1.56 KB
ANSI
CR+LF
INS
Configuration Files

```
;; this section contains variables shared by the different subtasks
[shared]
; space delimited list of tower bay id's to process
tower_bay_list = 0 1 4 5 8 9

instrument      = LAT
output_dir      = ./

;; parameters passed to runCIFit.exe
[runCIFit]
enable_section     = 1
calibGen_timestamp = 2000-1-1-00:00
ci_rootfile_le     = calibGen/digitization-v3r1p24_135003887_digi_DIGI.root
ci_rootfile_he     = calibGen/digitization-v3r1p24_135003888_digi_DIGI.root

;; parameters passed to generate pedestals from muon data using runMuonCalib.exe
[muon_peds]
enable_section      = 1
muped_timestamp     = 2000-1-1-00:00
muped_rootfiles     = muTrg/digitization-v3r1p24_135004166_digi_DIGI.root

;; parameters pass to generate optical calib (asym && mevPerDac) from runMuonCalib.exe
[muon_optical]
enable_section      = 1
muopt_timestamp     = 2000-1-1-00:00
muopt_rootfiles     = C:\GLAST\runfolders\test_cgc_parallel\b10\digitization-v3r1p24_135004127_digi_DIGI.root
C:\GLAST\runfolders\test_cgc_parallel\b10\digitization-v3r1p24_135004128_digi_DIGI.root


;; parameters passed to runMuTrigEff.exe
[runMuTrigEff]
enable_section         = 1
muTrig_timestamp       = 2000-1-1-00:00
mt_rootfile_muon_A = muTrg/digitization-v3r1p24_135004165_digi_DIGI.root
mt_rootfile_muon_B = muTrg/digitization-v3r1p24_135004166_digi_DIGI.root
mt_rootfile_ci     = muTrg/digitization-v3r1p24_135004164_digi_DIGI.root

;; parameters used for python merge scripts
[merge]
enable_section         = 1

;; parameters used for python validation scripts
[validate]
enable_section         = 1
```

PURPOSE:
- Loops through all towers & perform intNonlin, muonCalib, muTrigEff analysis on each tower
- Merge all single tower outputs
- Validate output

INPUTS
- cal digi event root files from calibGen, calu_collect_mu, & muTrg

WHAT IT *DOESN'T* DO
- genDACsettings, tholdCIGen

EXAMPLE CFG FILE

calibGenCAL/python/cfg/runSuiteParallel.cfg

WINDOWS COMMANDLINE

“runSuiteParallel cfgfile”

NOTES:

- The command line is simplified by the fact that we created a .bat file launcher (so the user doesn't have to type "python blah.py etc.."

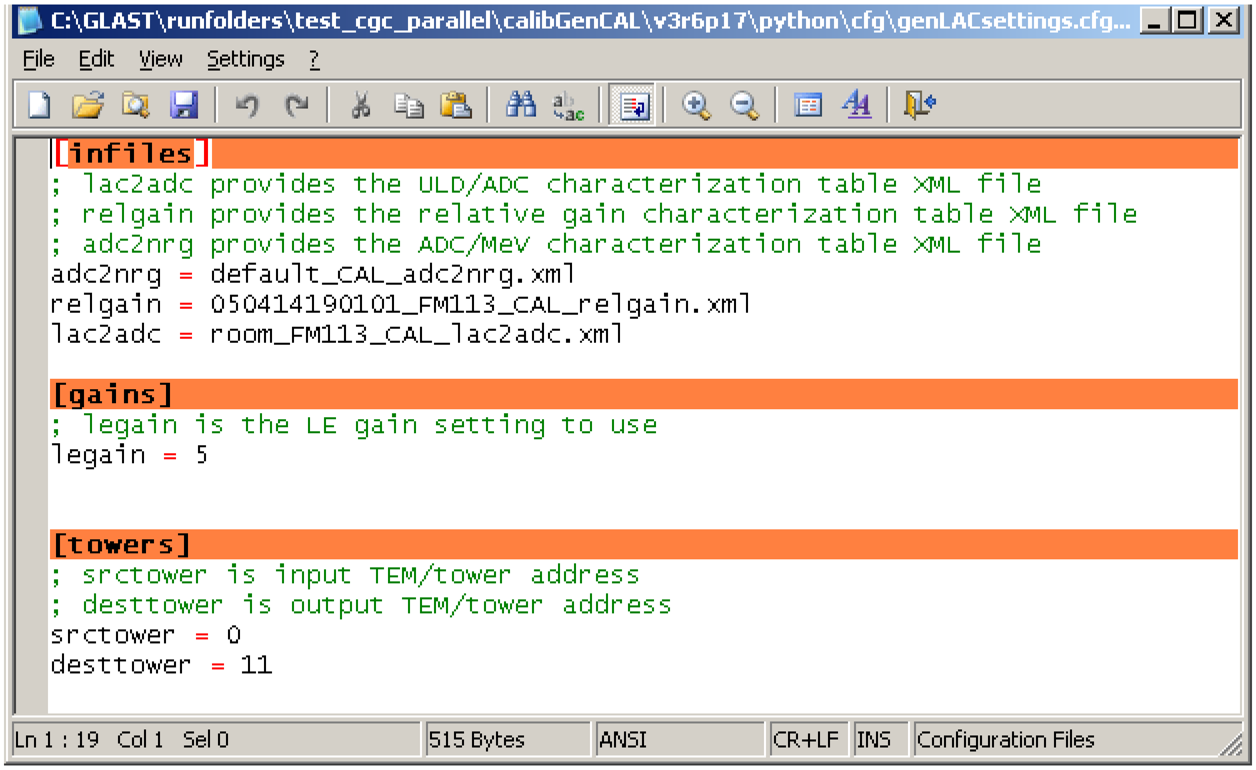
- The validation scripts are not official tests, they are mostly simple range checking. A failure does not necessarily mean that the data is bad, but you should look at the outlying values which are selected by the val script and make your own determination of their validity.

- Each section is *modular* and may be individually enabled/disabled. You should still fill out the parameters for all sections though, as successive phases will use info from previous phases to generate meta filenames & find their input files.

- Outputs are all written to the same folder & some are read back in by later stages from the same folder…. Don't get fancy ☺

- Muon calibration is broken into 2 phases.
    - pedestals, which can be performed on 10k/tower event file w/ 4-range readout, non zero-suppressed
    - optical (asym, mevPerDac), which requires ideally 500k/tower events, 4-range readout. Zero suppression is preferred but not required for speed & disk space reasons.

genFLEsettings, genFHEsettings, genULDsettings, genLACsettings


C:\GLAST\runfolders\test_cgc_parallel\calibGenCAL\v3r6p17\python\cfg\genLACsettings.cfg...
File Edit View Settings ?

```
[infiles]
; lac2adc provides the ULD/ADC characterization table XML file
; relgain provides the relative gain characterization table XML file
; adc2nrg provides the ADC/Mev characterization table XML file
adc2nrg = default_CAL_adc2nrg.xml
relgain = 050414190101_FM113_CAL_relgain.xml
lac2adc = room_FM113_CAL_lac2adc.xml

[gains]
; legain is the LE gain setting to use
legain = 5


[towers]
; srctower is input TEM/tower address
; desttower is output TEM/tower address
srctower = 0
desttower = 11
```

PURPOSE:

- Generate single tower online instrument configuration XML file for a single tower.

INPUTS

- **FLE**
  - fle2adc characterization xml file
  - adc2nrg xml file
  - relgain xml file
  - fle_fhe_bias xml file
- **FHE**
  - fhe2adc characterization xml file
  - adc2nrg xml file
  - relgain xml file
  - fle_fhe_bias xml file
- **LAC**
  - lac2adc characterization xml file
  - adc2nrg xml file
  - relgain xml file
- **ULD**
  - uld2adc characterization xml file

EXAMPLE CFG FILE

calibGenCAL/pythong/cfg/genXXXsettings.cfg

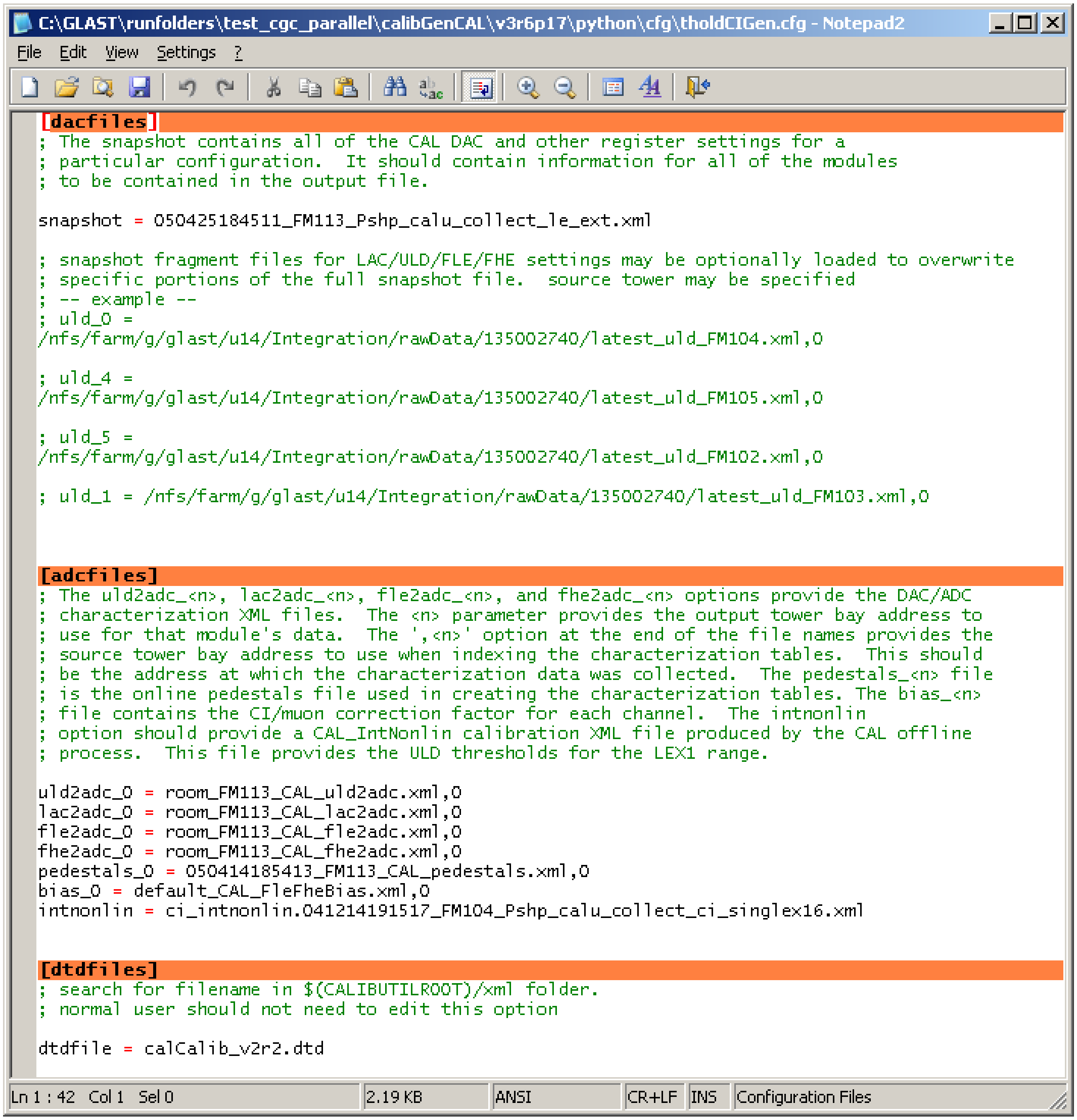
NOTES

This tool & its inputs are still single tower; you will have to execute it once per installed LAT tower.
To see an official command line usage statement, simply execute the program w/ no args.

WINDOWS COMMANDLINE

```
“genFLEsettings [-V] <MeV> <cfg_file> <out_xml_file>”
“genFHEsettings [-V] <GeV> <cfg_file> <out_xml_file>”
“genLACsettings [-V] <MeV> <cfg_file> <out_xml_file>”
“genULDsettings [-V] <cfg_file> <out_xml_file>”
```

tholdCIGen


C:\GLAST\runfolders\test_cgc_parallel\calibGenCAL\v3r6p17\python\cfg\tholdCIGen.cfg - Notepad2
File Edit View Settings ?

```
[dacfiles]
; The snapshot contains all of the CAL DAC and other register settings for a
; particular configuration.  It should contain information for all of the modules
; to be contained in the output file.

snapshot = 050425184511_FM113_Pshp_calu_collect_le_ext.xml

; snapshot fragment files for LAC/ULD/FLE/FHE settings may be optionally loaded to overwrite
; specific portions of the full snapshot file.  source tower may be specified
; -- example --
; uld_0 =
/nfs/farm/g/glast/u14/Integration/rawData/135002740/latest_uld_FM104.xml,0

; uld_4 =
/nfs/farm/g/glast/u14/Integration/rawData/135002740/latest_uld_FM105.xml,0

; uld_5 =
/nfs/farm/g/glast/u14/Integration/rawData/135002740/latest_uld_FM102.xml,0

; uld_1 = /nfs/farm/g/glast/u14/Integration/rawData/135002740/latest_uld_FM103.xml,0



[adcfiles]
; The uld2adc_<n>, lac2adc_<n>, fle2adc_<n>, and fhe2adc_<n> options provide the DAC/ADC
; characterization XML files.  The <n> parameter provides the output tower bay address to
; use for that module's data.  The ',<n>' option at the end of the file names provides the
; source tower bay address to use when indexing the characterization tables.  This should
; be the address at which the characterization data was collected.  The pedestals_<n> file
; is the online pedestals file used in creating the characterization tables. The bias_<n>
; file contains the CI/muon correction factor for each channel.  The intnonlin
; option should provide a CAL_IntNonlin calibration XML file produced by the CAL offline
; process.  This file provides the ULD thresholds for the LEX1 range.

uld2adc_0 = room_FM113_CAL_uld2adc.xml,0
lac2adc_0 = room_FM113_CAL_lac2adc.xml,0
fle2adc_0 = room_FM113_CAL_fle2adc.xml,0
fhe2adc_0 = room_FM113_CAL_fhe2adc.xml,0
pedestals_0 = 050414185413_FM113_CAL_pedestals.xml,0
bias_0 = default_CAL_FleFheBias.xml,0
intnonlin = ci_intnonlin.041214191517_FM104_Pshp_calu_collect_ci_singlex16.xml


[dtdfiles]
; search for filename in $(CALIBUTILROOT)/xml folder.
; normal user should not need to edit this option

dtdfile = calCalib_v2r2.dtd
```

PURPOSE

- generates full set (ULD,LAC,FLE,FHE) of threshold levels for offline simulation/reconstruction

INPUTS

- snapshot instrument configuration xml file (one per lat)
- (optional) override parts of snapshot configuration w/ fragment xml files
- uld2adc,fle2adc,fhe2adc,lac2adc characterization xml files (per tower)

- online pedestal xml file (per tower)
- fle_fhe bias files (per tower)
- intNonlin xml file (per lat)

WINDOWS COMMANDLINE

“tholdCIGen [-V] <cfg_file> <out_xml_file>”

EXAMPLE CFG FILE

calibGenCAL/python/cfg/tholdCIGen.cfg

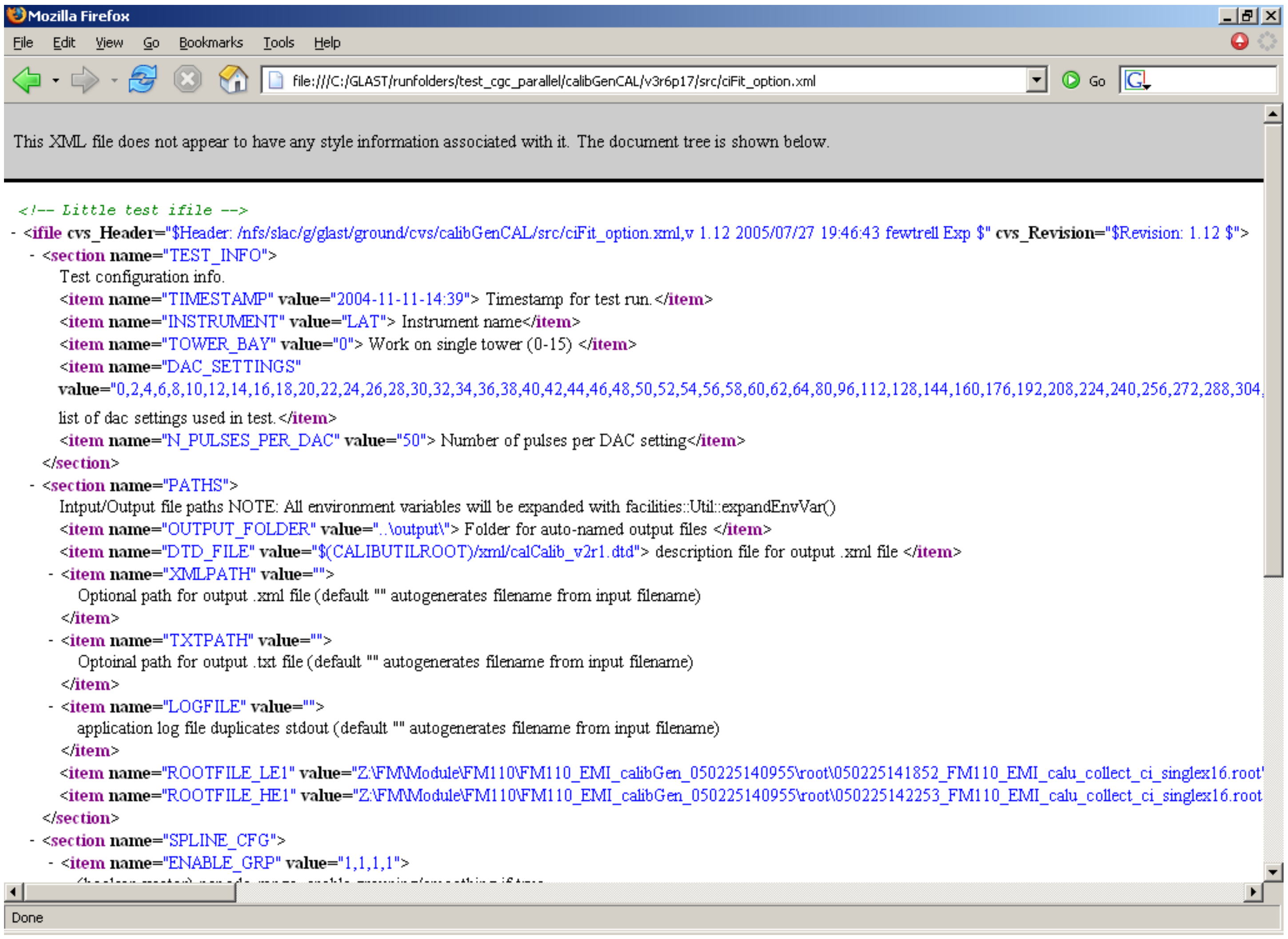
NOTES

It is difficult to run this along w/ the other calibGenCAL analysis as (of late) calibGenCAL analysis has been performed after calibration runs but before muon collection runs. To work properly tholdCIGen needs a snapshot file w/ properly calibrated config for muon collection.

Component Applications

There is a top level script, *runSuiteParallel* which will automate the process of launching most of these applications... It is modular to some extent... But if you need to repeat certain sub-stages, or if you have non-parallel input data, you will need to execute the component applications individually.

runCIFit.exe



```
This XML file does not appear to have any style information associated with it. The document tree is shown below.

<!-- Little test ifile -->
- <ifile cvs_Header="$Header: /nfs/slac/g/glast/ground/cvs/calibGenCAL/src/ciFit_option.xml,v 1.12 2005/07/27 19:46:43 fewtrell Exp $" cvs_Revision="$Revision: 1.12 $">
  - <section name="TEST_INFO">
      Test configuration info.
      <item name="TIMESTAMP" value="2004-11-11-14:39"> Timestamp for test run.</item>
      <item name="INSTRUMENT" value="LAT"> Instrument name</item>
      <item name="TOWER_BAY" value="0"> Work on single tower (0-15) </item>
      <item name="DAC_SETTINGS"
      value="0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38,40,42,44,46,48,50,52,54,56,58,60,62,64,80,96,112,128,144,160,176,192,208,224,240,256,272,288,304,
      list of dac settings used in test.</item>
      <item name="N_PULSES_PER_DAC" value="50"> Number of pulses per DAC setting</item>
    </section>
  - <section name="PATHS">
      Intput/Output file paths NOTE: All environment variables will be expanded with facilities::Util::expandEnvVar()
      <item name="OUTPUT_FOLDER" value="..\output\"> Folder for auto-named output files </item>
      <item name="DTD_FILE" value="$(CALIBUTILROOT)/xml/calCalib_v2r1.dtd"> description file for output .xml file </item>
    - <item name="XMLPATH" value="">
        Optional path for output .xml file (default "" autogenerates filename from input filename)
      </item>
    - <item name="TXTPATH" value="">
        Optoinal path for output .txt file (default "" autogenerates filename from input filename)
      </item>
    - <item name="LOGFILE" value="">
        application log file duplicates stdout (default "" autogenerates filename from input filename)
      </item>
      <item name="ROOTFILE_LE1" value="Z:\FM\Module\FM110\FM110_EMI_calibGen_050225140955\root\050225141852_FM110_EMI_calu_collect_ci_singlex16.root"
      <item name="ROOTFILE_HE1" value="Z:\FM\Module\FM110\FM110_EMI_calibGen_050225140955\root\050225142253_FM110_EMI_calu_collect_ci_singlex16.root
    </section>
  - <section name="SPLINE_CFG">
    - <item name="ENABLE_GRP" value="1,1,1,1">
```

OUTPUTS:

- integral nonlinearity (ADC->DAC) xml file for offline use
- txt file for internal use.

INPUT:

- 2 singlex16 files from calibGen online suite w/ the following configurations
    1) le_only, he_gain=muon, le_gain=nominal, tack_delay=nominal,triggers=nominal, calib_gain=on
    2) he_only, he_gain=muon, le_gain=nominal, tack_delay=nominal,triggers=nominal, calib_gain=off

    Note: these files correspond respectively to the 3rd & 4th singlex16 runs of an 8 run calibGen suite.

EXAMPLE CFG:

calibGenCAL/src/ciFit_option.xml

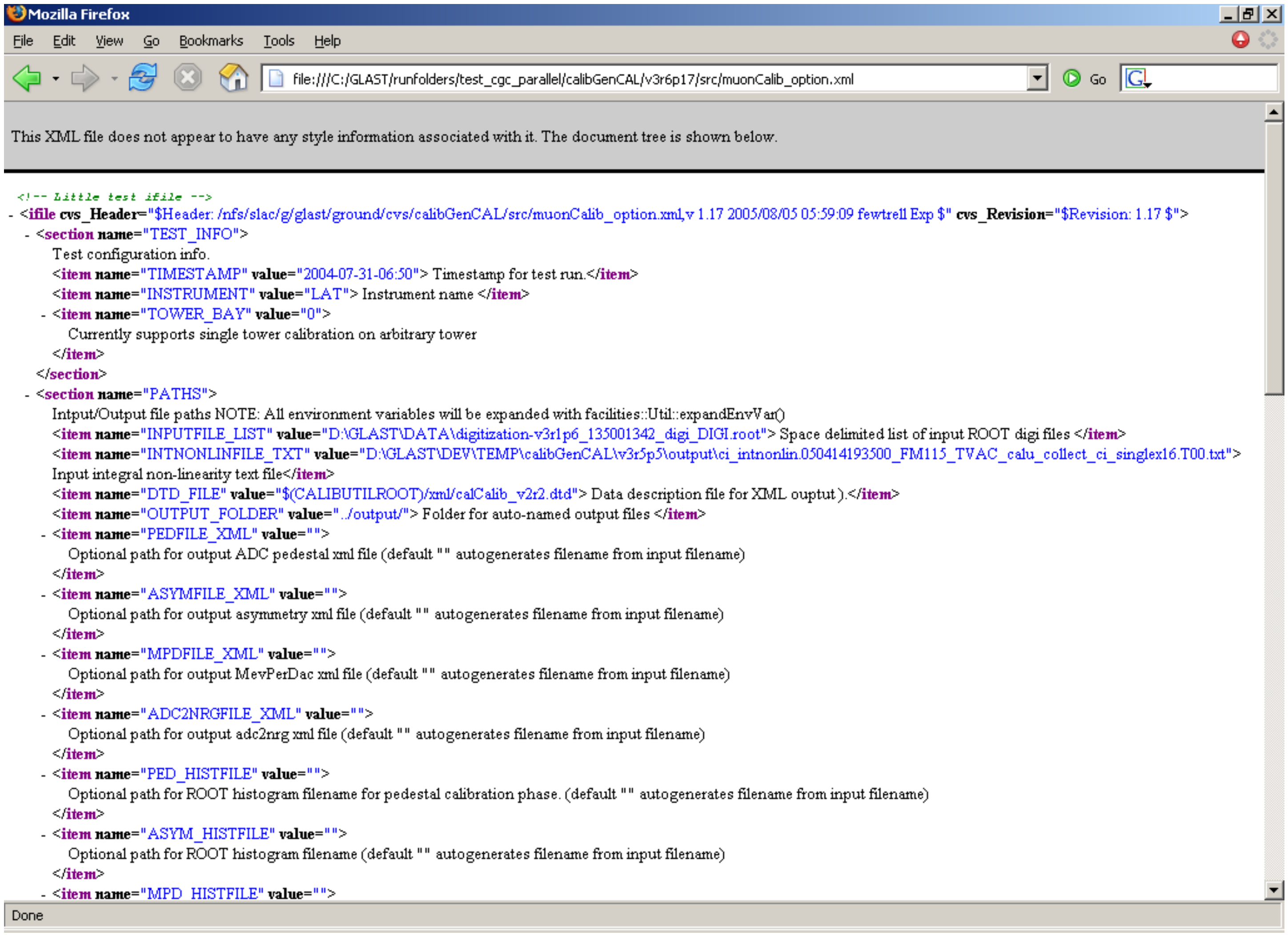
NOTES:

- Supports multi tower input, but processes only 1 tower @ a time
- Don't forget to set TOWER_BAY config parameter to tower you wish to process.
- set the BROADCAST_MODE config option to true if your calibGen online script sampled all crystal columns in parallel instead of series. (This would be evident by a lower event count if nothing else)

WINDOWS COMMAND LINE

"runCIFit.exe config.xml"

runMuonCalib.exe


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
file:///C:/GLAST/runfolders/test_cgc_parallel/calibGenCAL/v3r6p17/src/muonCalib_option.xml
Go
Done

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
 <!-- Little test ifile -->
- <ifile cvs_Header="$Header: /nfs/slac/g/glast/ground/cvs/calibGenCAL/src/muonCalib_option.xml,v 1.17 2005/08/05 05:59:09 fewtrell Exp $" cvs_Revision="$Revision: 1.17 $">
  - <section name="TEST_INFO">
      Test configuration info.
      <item name="TIMESTAMP" value="2004-07-31-06:50"> Timestamp for test run.</item>
      <item name="INSTRUMENT" value="LAT"> Instrument name </item>
    - <item name="TOWER_BAY" value="0">
        Currently supports single tower calibration on arbitrary tower
      </item>
    </section>
  - <section name="PATHS">
      Intput/Output file paths NOTE: All environment variables will be expanded with facilities::Util::expandEnvVar()
      <item name="INPUTFILE_LIST" value="D:\GLAST\DATA\digitization-v3r1p6_135001342_digi_DIGI.root"> Space delimited list of input ROOT digi files </item>
      <item name="INTNONLINFILE_TXT" value="D:\GLAST\DEV\TEMP\calibGenCAL\v3r5p5\output\ci_intnonlin.050414193500_FM115_TVAC_calu_collect_ci_singlex16.T00.txt">
      Input integral non-linearity text file</item>
      <item name="DTD_FILE" value="$(CALIBUTILROOT)/xml/calCalib_v2r2.dtd"> Data description file for XML ouptut ).</item>
      <item name="OUTPUT_FOLDER" value="../output/"> Folder for auto-named output files </item>
    - <item name="PEDFILE_XML" value="">
        Optional path for output ADC pedestal xml file (default "" autogenerates filename from input filename)
      </item>
    - <item name="ASYMFILE_XML" value="">
        Optional path for output asymmetry xml file (default "" autogenerates filename from input filename)
      </item>
    - <item name="MPDFILE_XML" value="">
        Optional path for output MevPerDac xml file (default "" autogenerates filename from input filename)
      </item>
    - <item name="ADC2NRGFILE_XML" value="">
        Optional path for output adc2nrg xml file (default "" autogenerates filename from input filename)
      </item>
    - <item name="PED_HISTFILE" value="">
        Optional path for ROOT histogram filename for pedestal calibration phase. (default "" autogenerates filename from input filename)
      </item>
    - <item name="ASYM_HISTFILE" value="">
        Optional path for ROOT histogram filename (default "" autogenerates filename from input filename)
      </item>
    - <item name="MPD_HISTFILE" value="">
```

OUTPUTS

- pedestal, asymmetry, mevPerDAC xml files for offline use
- adc2nrg xml file for online use
- pedestal txt file for internal use

INPUT

1 or more 4-range muon collection files.

EXAMPLE CFG

calibGenCAL/src/muonCalib_option.xml

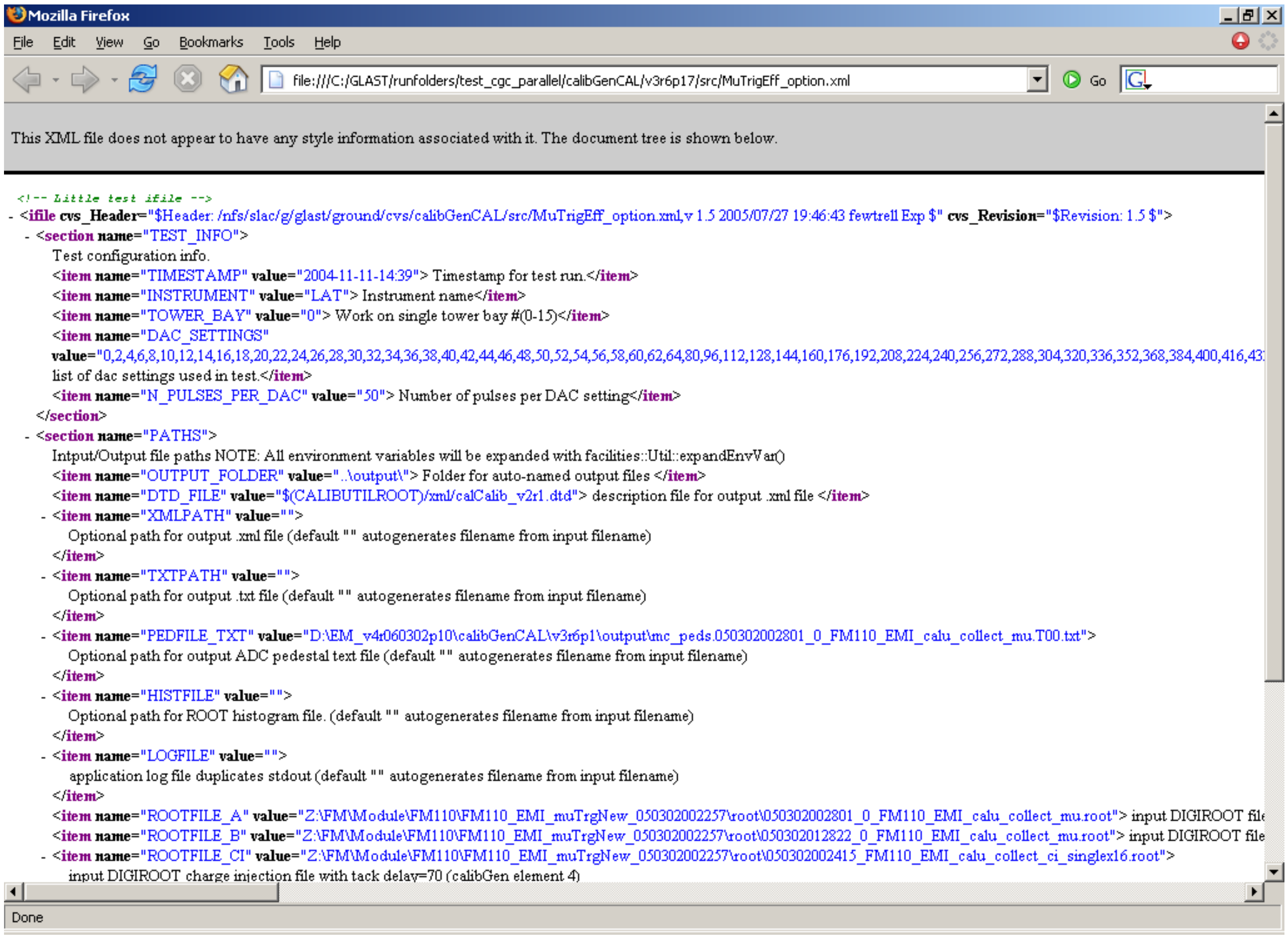
NOTES

-supports multi tower input, but processes only 1 tower @ a time
-each section (ped, asym, mevPerDAC) can be run separately by enabling/disabling appropriate Booleans in config file.
Note: Don't forget to set TOWER_BAY config parameter to tower you wish to process.

WINDOWS COMMANDLINE

“runMuonCalib config.xml”

runMuTrigEff.exe



File Edit View Go Bookmarks Tools Help

file:///C:/GLAST/runfolders/test_cgc_parallel/calibGenCAL/v3r6p17/src/MuTrigEff_option.xml

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
 <!-- Little test ifile -->
- <ifile cvs_Header="$Header: /nfs/slac/g/glast/ground/cvs/calibGenCAL/src/MuTrigEff_option.xml,v 1.5 2005/07/27 19:46:43 fewtrell Exp $" cvs_Revision="$Revision: 1.5 $">
  - <section name="TEST_INFO">
      Test configuration info.
      <item name="TIMESTAMP" value="2004-11-11-14:39"> Timestamp for test run.</item>
      <item name="INSTRUMENT" value="LAT"> Instrument name</item>
      <item name="TOWER_BAY" value="0"> Work on single tower bay #(0-15)</item>
      <item name="DAC_SETTINGS"
      value="0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38,40,42,44,46,48,50,52,54,56,58,60,62,64,80,96,112,128,144,160,176,192,208,224,240,256,272,288,304,320,336,352,368,384,400,416,43
      list of dac settings used in test.</item>
      <item name="N_PULSES_PER_DAC" value="50"> Number of pulses per DAC setting</item>
    </section>
  - <section name="PATHS">
      Intput/Output file paths NOTE: All environment variables will be expanded with facilities::Util::expandEnvVar()
      <item name="OUTPUT_FOLDER" value="..\output\"> Folder for auto-named output files </item>
      <item name="DTD_FILE" value="$(CALIBUTILROOT)/xml/calCalib_v2r1.dtd"> description file for output .xml file </item>
    - <item name="XMLPATH" value="">
        Optional path for output .xml file (default "" autogenerates filename from input filename)
      </item>
    - <item name="TXTPATH" value="">
        Optional path for output .txt file (default "" autogenerates filename from input filename)
      </item>
    - <item name="PEDFILE_TXT" value="D:\EM_v4r060302p10\calibGenCAL\v3r6p1\output\mc_peds.050302002801_0_FM110_EMI_calu_collect_mu.T00.txt">
        Optional path for output ADC pedestal text file (default "" autogenerates filename from input filename)
      </item>
    - <item name="HISTFILE" value="">
        Optional path for ROOT histogram file. (default "" autogenerates filename from input filename)
      </item>
    - <item name="LOGFILE" value="">
        application log file duplicates stdout (default "" autogenerates filename from input filename)
      </item>
      <item name="ROOTFILE_A" value="Z:\FM\Module\FM110\FM110_EMI_muTrgNew_050302002257\root\050302002801_0_FM110_EMI_calu_collect_mu.root"> input DIGIROOT file
      <item name="ROOTFILE_B" value="Z:\FM\Module\FM110\FM110_EMI_muTrgNew_050302002257\root\050302012822_0_FM110_EMI_calu_collect_mu.root"> input DIGIROOT file
    - <item name="ROOTFILE_CI" value="Z:\FM\Module\FM110\FM110_EMI_muTrgNew_050302002257\root\050302002415_FM110_EMI_calu_collect_ci_singlex16.root">
        input DIGIROOT charge injection file with tack delay=70 (calibGen element 4)
```

Done

OUTPUTS:

fle_fhe_bias file for online use.

INPUT:

1 singlex16 CI file from muTrg suite
2 muon collection files from muTrg suite
1 pedestal txt file from muonCalib

EXAMPLE CFG:

calibGenCAL/src/MuTrigEff_option.xml

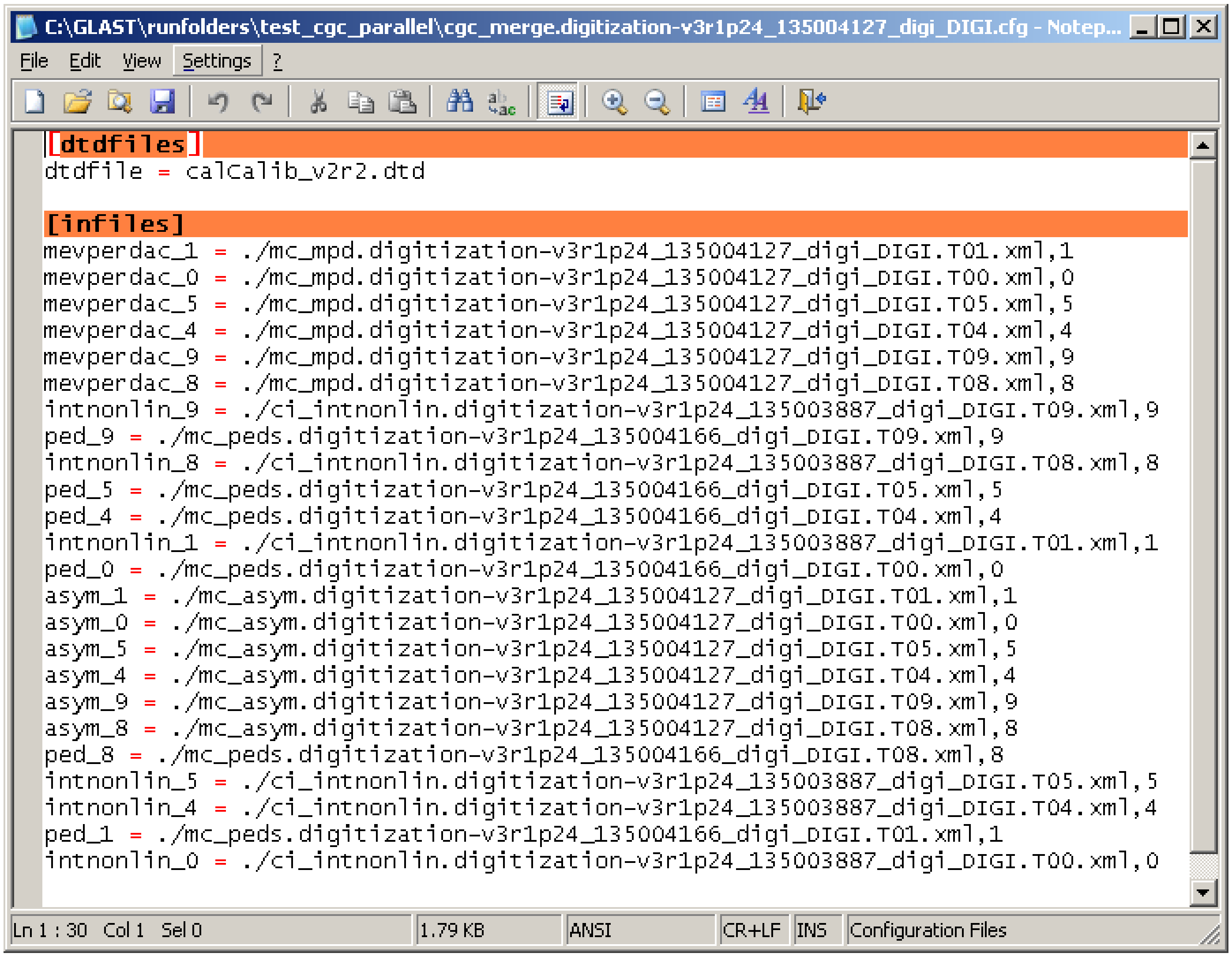
NOTES:

Supports multi tower input, but processes only 1 tower @ a time
Note: Don’t forget to set TOWER_BAY config parameter to tower you wish to process.

WINDOWS COMMANDLINE

“runMuTrigEff config.xml”

pedMerge, asymMerge, mevPerDacMerge, intNonlinMerge


C:\GLAST\runfolders\test_cgc_parallel\cgc_merge.digitization-v3r1p24_135004127_digi_DIGI.cfg - Notep...
File Edit View Settings ?
Ln 1 : 30 Col 1 Sel 0
1.79 KB
ANSI
CR+LF
INS
Configuration Files

```
[dtdfiles]
dtdfile = calCalib_v2r2.dtd

[infiles]
mevperdac_1 = ./mc_mpd.digitization-v3r1p24_135004127_digi_DIGI.T01.xml,1
mevperdac_0 = ./mc_mpd.digitization-v3r1p24_135004127_digi_DIGI.T00.xml,0
mevperdac_5 = ./mc_mpd.digitization-v3r1p24_135004127_digi_DIGI.T05.xml,5
mevperdac_4 = ./mc_mpd.digitization-v3r1p24_135004127_digi_DIGI.T04.xml,4
mevperdac_9 = ./mc_mpd.digitization-v3r1p24_135004127_digi_DIGI.T09.xml,9
mevperdac_8 = ./mc_mpd.digitization-v3r1p24_135004127_digi_DIGI.T08.xml,8
intnonlin_9 = ./ci_intnonlin.digitization-v3r1p24_135003887_digi_DIGI.T09.xml,9
ped_9 = ./mc_peds.digitization-v3r1p24_135004166_digi_DIGI.T09.xml,9
intnonlin_8 = ./ci_intnonlin.digitization-v3r1p24_135003887_digi_DIGI.T08.xml,8
ped_5 = ./mc_peds.digitization-v3r1p24_135004166_digi_DIGI.T05.xml,5
ped_4 = ./mc_peds.digitization-v3r1p24_135004166_digi_DIGI.T04.xml,4
intnonlin_1 = ./ci_intnonlin.digitization-v3r1p24_135003887_digi_DIGI.T01.xml,1
ped_0 = ./mc_peds.digitization-v3r1p24_135004166_digi_DIGI.T00.xml,0
asym_1 = ./mc_asym.digitization-v3r1p24_135004127_digi_DIGI.T01.xml,1
asym_0 = ./mc_asym.digitization-v3r1p24_135004127_digi_DIGI.T00.xml,0
asym_5 = ./mc_asym.digitization-v3r1p24_135004127_digi_DIGI.T05.xml,5
asym_4 = ./mc_asym.digitization-v3r1p24_135004127_digi_DIGI.T04.xml,4
asym_9 = ./mc_asym.digitization-v3r1p24_135004127_digi_DIGI.T09.xml,9
asym_8 = ./mc_asym.digitization-v3r1p24_135004127_digi_DIGI.T08.xml,8
ped_8 = ./mc_peds.digitization-v3r1p24_135004166_digi_DIGI.T08.xml,8
intnonlin_5 = ./ci_intnonlin.digitization-v3r1p24_135003887_digi_DIGI.T05.xml,5
intnonlin_4 = ./ci_intnonlin.digitization-v3r1p24_135003887_digi_DIGI.T04.xml,4
ped_1 = ./mc_peds.digitization-v3r1p24_135004166_digi_DIGI.T01.xml,1
intnonlin_0 = ./ci_intnonlin.digitization-v3r1p24_135003887_digi_DIGI.T00.xml,0
```

***note**: this file was generated by runSuiteParallel & contains merge parameters for all 4 calibration types. Each merge script will read in only the parameters it requires.

OUTPUTS

Merged offline xml calibration files for full LAT

INPUTS

Single tower offline xml calibration files.

EXAMPLE CFG FILE(S)

calibGenCAL/python/cfg/xxxMerge.cfg

pedVal, asymVal, intNonlinVal, mevPerDacVal

PURPOSE

-validates offline calibration xml files

WINDOWS COMMANDLINE

```
pedVal [-V] [-L <log_file>] [-E <err_limit>] [-W <warn_limit>] [-R <root_file>] <xml_file>
asymVal [-V] [-L <log_file>] [-E <err_limit>] [-W <warn_limit>] [-R <root_file>] <xml_file>
mevPerDacVal [-V] [-L <log_file>] [-E <err_limit>] [-W <warn_limit>] [-R <root_file>] <xml_file>
intNonlinVal [-V] [-L <log_file>] [-E <err_limit>] [-W <warn_limit>] [-R <root_file>] <xml_file>
```

err_limit, warn_limit, root_file, log_file are all *optional* parameters

EXAMPLE CFG

None, all parameters are command line

NOTES

- The validation scripts are not official tests, they are mostly simple range checking. A failure does not necessarily mean that the data is bad, but you should look at the outlying values which are selected by the validation script and make your own determination of their validity.