Argonne
NATIONAL
LABORATORY

... for a brighter future

U.S. Department
of Energy

UChicago ▶
Argonne LLC

A U.S. Department of Energy laboratory
managed by UChicago Argonne, LLC

CALICE DAQ for the DHCAL

J.L.Schlereth

March 17, 2008

Motivation

- Need for more structured DAQ for 1m^3 prototype
  - 5760 vs 32 front-end ASICs to configure and readout.
- Advantages of CALICE DAQ software
  - scope of features appropriate to project of this size
    - *Run Control*
    - *Monitoring*
    - *Data recording*
  - extensible framework accommodates DHCAL hardware
  - provides possibility of easier integration with other detectors in same test beam

History and Milestones

- November, 2007
  - Study CALICE DAQ code and documentation
- December, 2007
  - Produce design report specifying modifications and additions needed for DHCAL
- January, 2008
  - Code development begun on notebook PC with no VME hardware
- February, 2008
  - Testing done with DAQ hardware
  - Charge Injection scan run type implemented
  - Configuration based on simple XML files developed
  - Analysis code developed to perform quick checks
  - Integration with CALICE DAQ

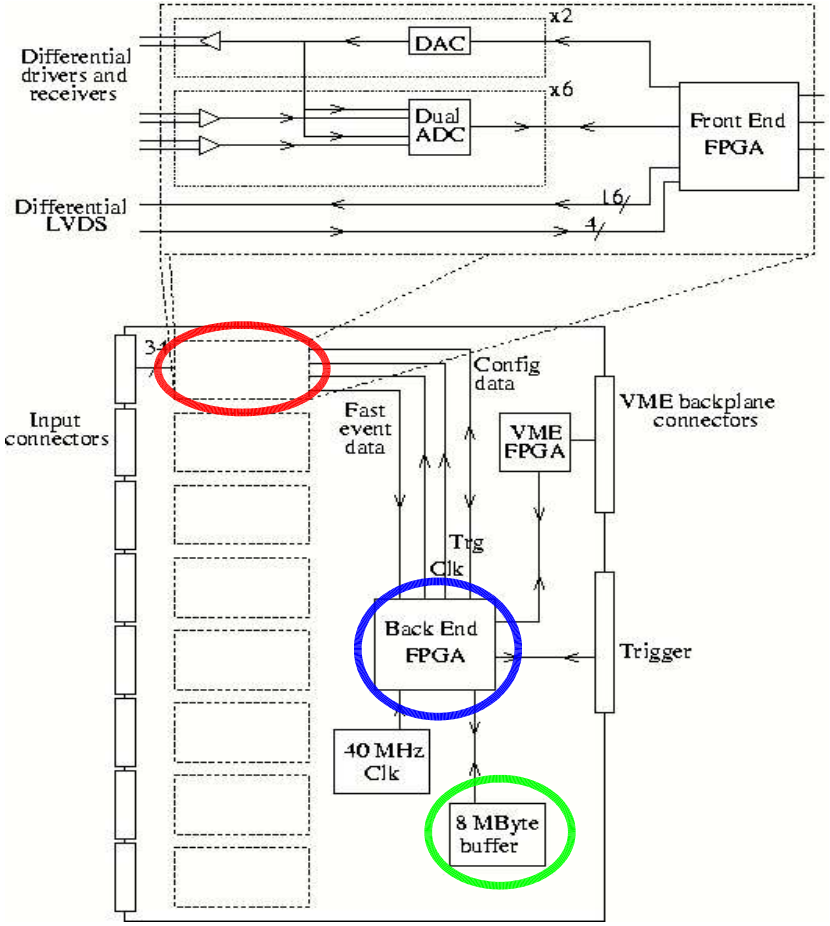
DHCAL - CALICE DAQ Differences

- CALICE DAQ based around CRC
  - Ahc, Dhe derive from Crc base classes
- CRC module also used for triggering
- Trigger data always read out along with event data in separate sub-records

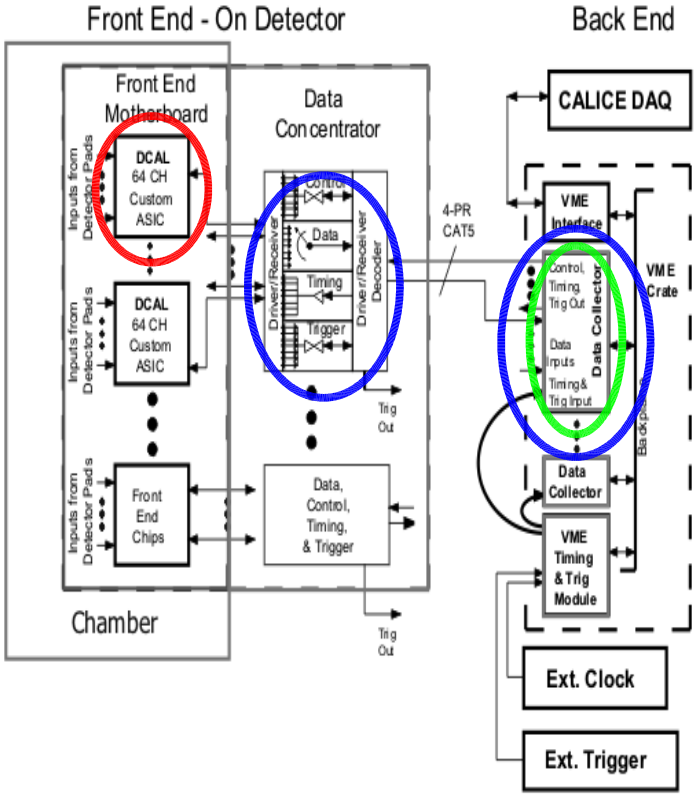
- DHCAL based around Dcal (front-end) ASIC, Data Concentrator, Data Collector
- Trigger & Timing Module used for triggering
- Trigger time stamps and hit data all part of event data sub-record

CRC and DHCAL Block Diagrams

CRC



DHC



DHCAL – CALICE DAQ Common Features

- Configuration needs are similar
  - Front-end and back-end parameters
  - serial protocol used for front-end configuration
  - VME slot broadcast, FE slot broadcast concepts apply
- Run types
  - Noise measurements (polling trigger)
  - Charge injection scans (program controlled trigger)
  - Cosmic ray and beam spill operations (either external or polling trigger)
- VME Access
  - Use HAL in Readout modules
  - Use DaqBusAdapter concept, define implementation at compile time
- Even though code development was required to implement details specific to DHCAL readout hardware, same concepts deployed in CRC specific software were applicable.

Common DAQ Modules Modified

- DaqBusAdapter
  - include CAENLinuxBusAdapter
- DaqConfiguration
  - add RunTypes DhcQinj, DhcQinjScan
  - add operations under RunTypes DhcNoise, DhcCosmics
- DaqRunType
  - add operations for RunTypes DhcQinj, DhcQinjScan
- SubRecordType
  - add DhcLocationData, TtmlocationData types
  - add DhcReadoutConfigurationData, DhcBeConfigurationdata, DhcFeConfigurationData and TtmConfigurationData types

DHCAL specific Modules Added

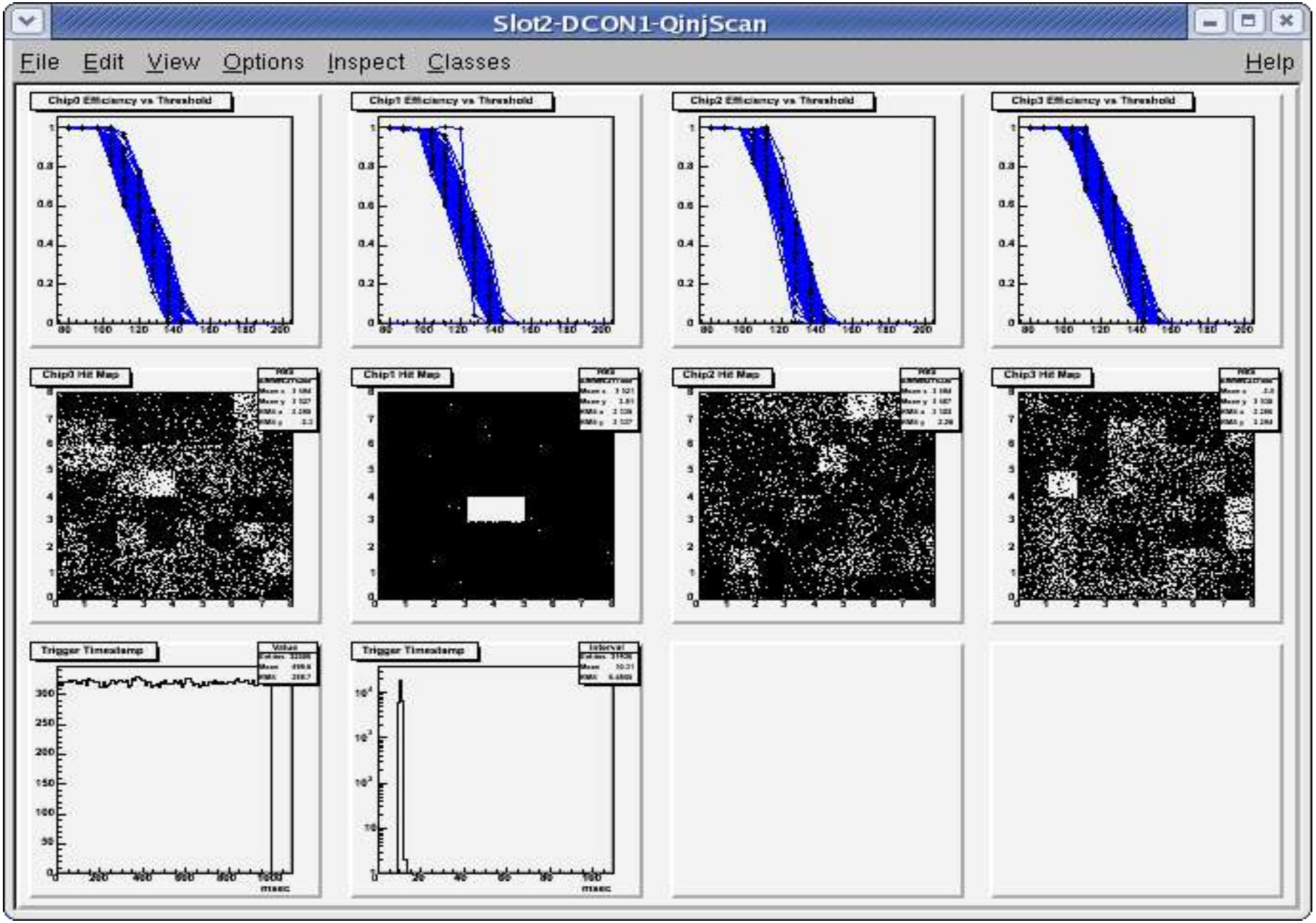
- daquser/inc/dhc
  - DhcConfiguration
- online/inc/dhc
  - DhcReadout
  - DhcSerialCommand
  - DhcSerialHeader
  - DhcVmeDevice
- online/inc/ttm
  - TtmReadout
  - TtmVmeDevice
- records/inc/dhc
  - DhcLocation
  - DhcLocationData
  - DhcFeConfigurationData
  - DhcBeConfigurationData
  - DhcReadoutConfigurationData
  - DhcEventData
  - DhcFeHitData
- records/inc/ttm
  - TtmConfigurationData
  - TtmLocationData
  - TtmLocation

Optional DHCAL Add-ons

- DhcConfigReader
  - Driven by need to be able to change small number of parameters from run to run
  - Most parameters applied globally, but some components may need unique settings
  - provides interface between external file and internal data objects
  - parses XML format file, fills RcdRecord with appropriate SubRecords, stored in file used in 'runner' application
  - relies on external library, TinyXML package.
- DaqElogger
  - provides interface to ELOG package of Stefan Ritt
  - entries automatically created with run type, run number, start, stop time
  - entries can be edited using ELOG WeBlog interface.
- DhcWriterBin
  - limited lifetime code that converts RcdRecords from DHCAL to format usable by existing analysis code.

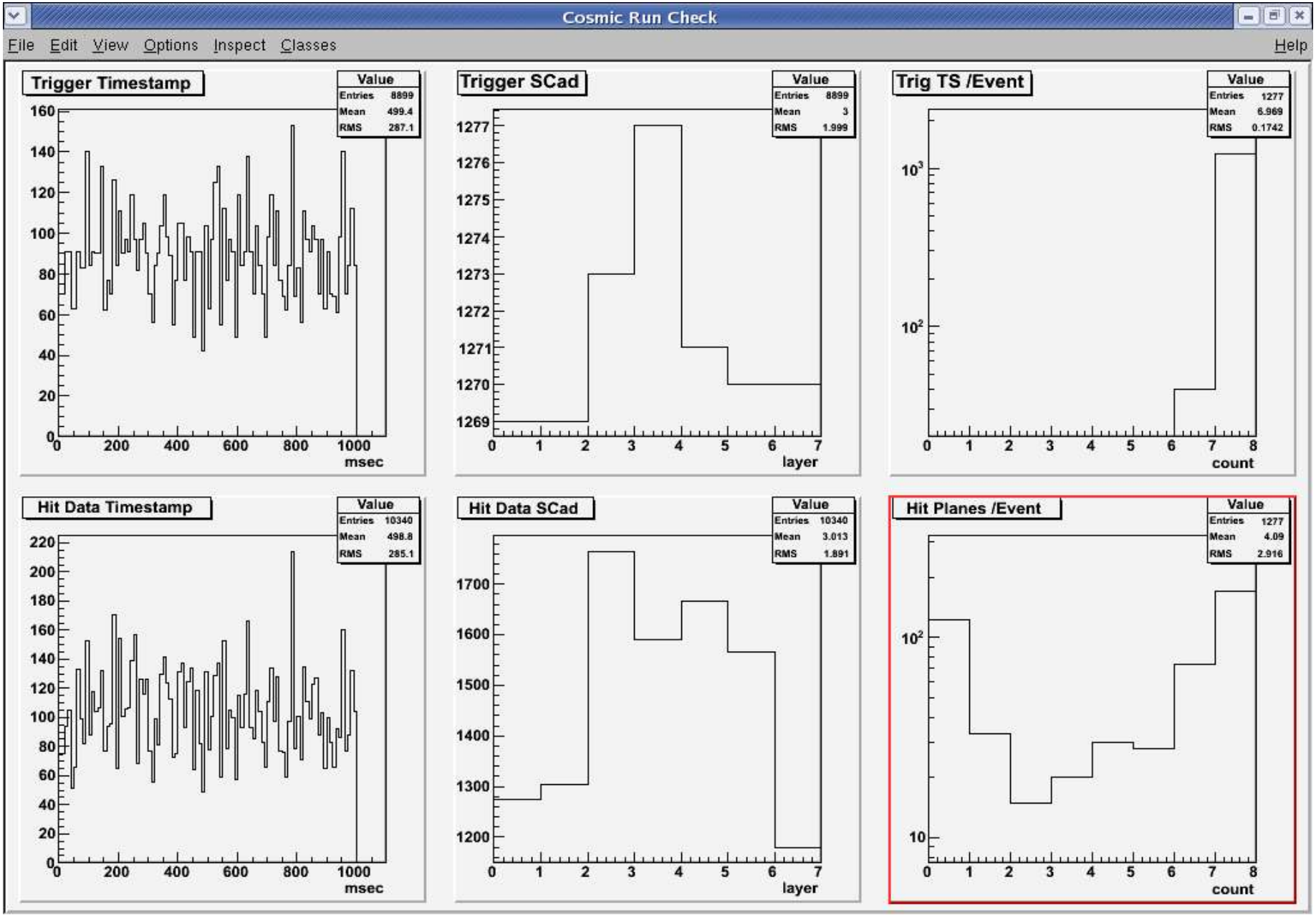
Use case - QinjScan

- Charge injection circuit included on front-end ASIC
  - check threshold behavior
  - check for dead channels
- Single run with multiple configurations (typically 64)
  - vary threshold setting for selected number of steps of selected step size
  - for each threshold setting, run 4 configurations with charge injected into ¼ of the channels.
  - run time 5-6 mins at 100 Hz trigger rate
- Analysis of results
  - checks threshold behavior, dead channels and trigger time stamp integrity
  - currently done offline on data file (~16 Mbytes for 32 ASICs), full prototype (5760 ASICs) scales to ~3 Gbyte data file
  - plan to develop online analysis, storing summary of results.


Slot2-DCON1-QinjScan
File Edit View Options Inspect Classes Help
Chip0 Efficiency vs Threshold
Chip1 Efficiency vs Threshold
Chip2 Efficiency vs Threshold
Chip3 Efficiency vs Threshold
Chip0 Hit Map
Chip1 Hit Map
Chip2 Hit Map
Chip3 Hit Map
Trigger Timestamp
Trigger Timestamp

Use Case - Cosmics

- Tried with 2 triggering methods
  - fixed poll interval, based on average rate
    - *sleep during interval, other processes get CPU time*
  - poll of Trigger Time stamp FIFO
    - *uses 100% of CPU*
- Results not sensitive to trigger method
  - events can be built in either case using time stamps


Cosmic Run Check
File Edit View Options Inspect Classes Help
Trigger Timestamp
Value
Entries 8899
Mean 499.4
RMS 287.1
msec
Trigger SCad
Value
Entries 8899
Mean 3
RMS 1.999
layer
Trig TS /Event
Value
Entries 1277
Mean 6.969
RMS 0.1742
count
Hit Data Timestamp
Value
Entries 10340
Mean 498.8
RMS 285.1
msec
Hit Data SCad
Value
Entries 10340
Mean 3.013
RMS 1.891
layer
Hit Planes /Event
Value
Entries 1277
Mean 4.09
RMS 2.916
count

Tasks / Issues Remaining

- Online monitoring of DHCAL
  - clear what to do, need to devote effort
- Include Slow Monitoring
  - High voltage
  - environmental conditions
- LCIO conversion of DHCAL event data
  - define what needs to be done
  - does raw data file contain adequate information
- Integration with other CALICE detectors
  - determine method to trigger different systems coherently

Summary

- CALICE DAQ framework provides flexible framework for DHCAL needs whether the detector is used standalone or with other CALICE detectors
- Many of the concepts originally developed to support CRC control and readout have been adapted to the DHCAL software
- A small scale system has been developed and tested and could easily be adapted to support 1m^3 prototype