HCAL software framework

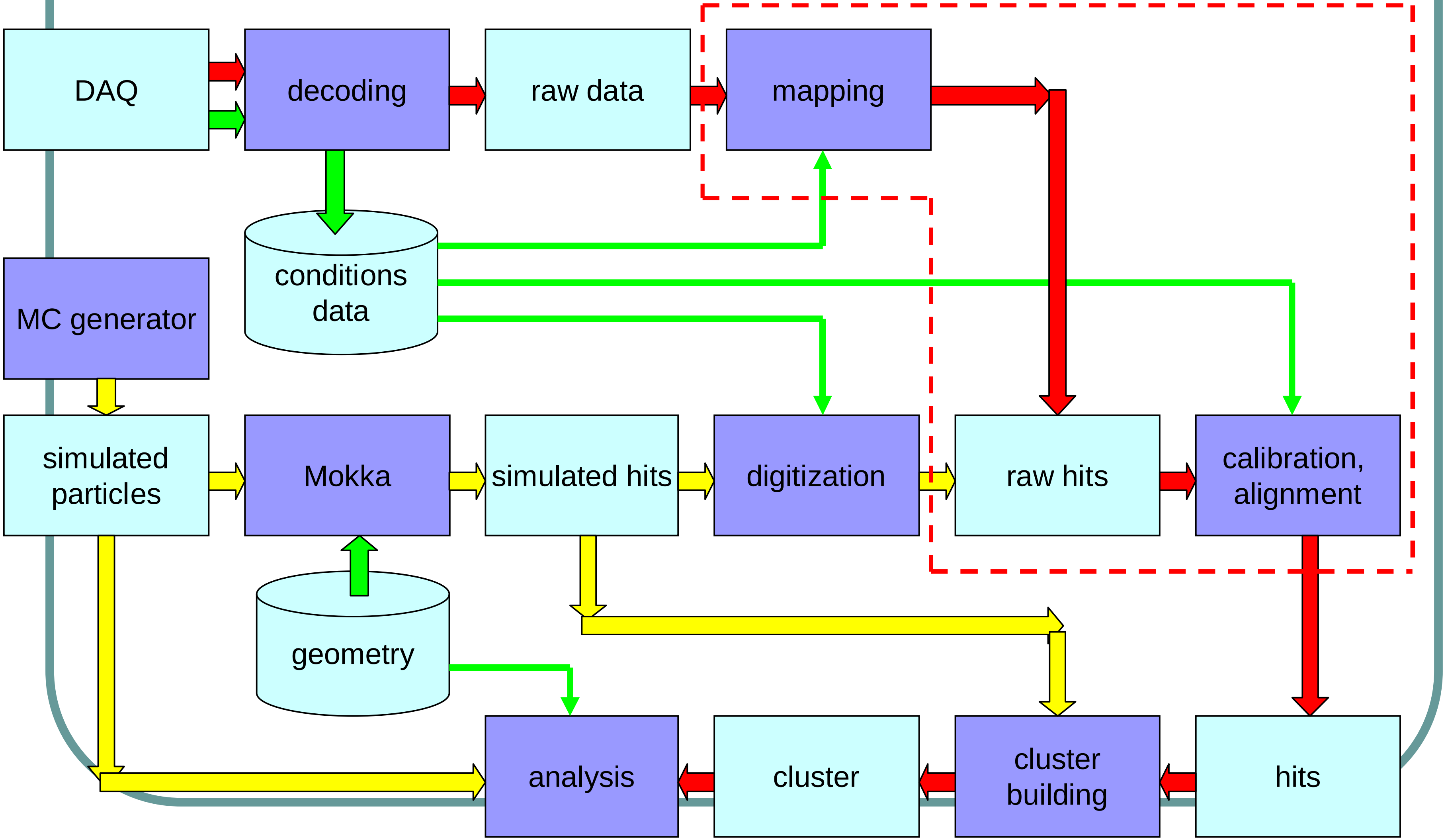
Sebastian Schmidt, DESY
ECFA meeting, Valencia

09-NOV-06

overview

- CALICE software environment
- HCAL analysis procedure
- Requirements
- HCAL reconstruction software
- Experiences with Linear Collider software

CALICE data streams



infrastructure

- Data
  - Sent from CERN to DESY using GBit link
  - Converted from raw DAQ format to LCIO, conditions data saved to conditions data base
  - Stored on DESY dcache system
  - Accessible via the grid
  - Altogether already several TB
- Conditions/calibration data stored in a MySQL data base
  - Access via LCCD
  - Two MySQL servers at DESY, reachable world wide
- Using MARLIN for reconstruction
- One of the first experiments testing the new linear collider software environment in the “real world”

HCAL analysis procedure

Pedestal subtraction: $A = A_0 - p$

Energy E deposited in one calorimeter cell [GeV]:

SiPM gain in ADC channels (taken in calibration mode)

Electronics inter-calibration between physics and calibration mode

$$E = N_{MIP} \cdot E_{MIP}^{MC} = \frac{f_{resp}\left(A \cdot \frac{I_{phys}^{calib}}{G_{pix}} \right)}{f_{resp}\left(A_{MIP} \cdot \frac{I_{phys}^{calib}}{G_{pix}} \right)} \cdot E_{MIP}^{MC} \approx \frac{A \cdot f_{corr}\left(A \cdot \frac{I_{phys}^{calib}}{G_{pix}} \right)}{A_{MIP}} \cdot E_{MIP}^{MC}$$

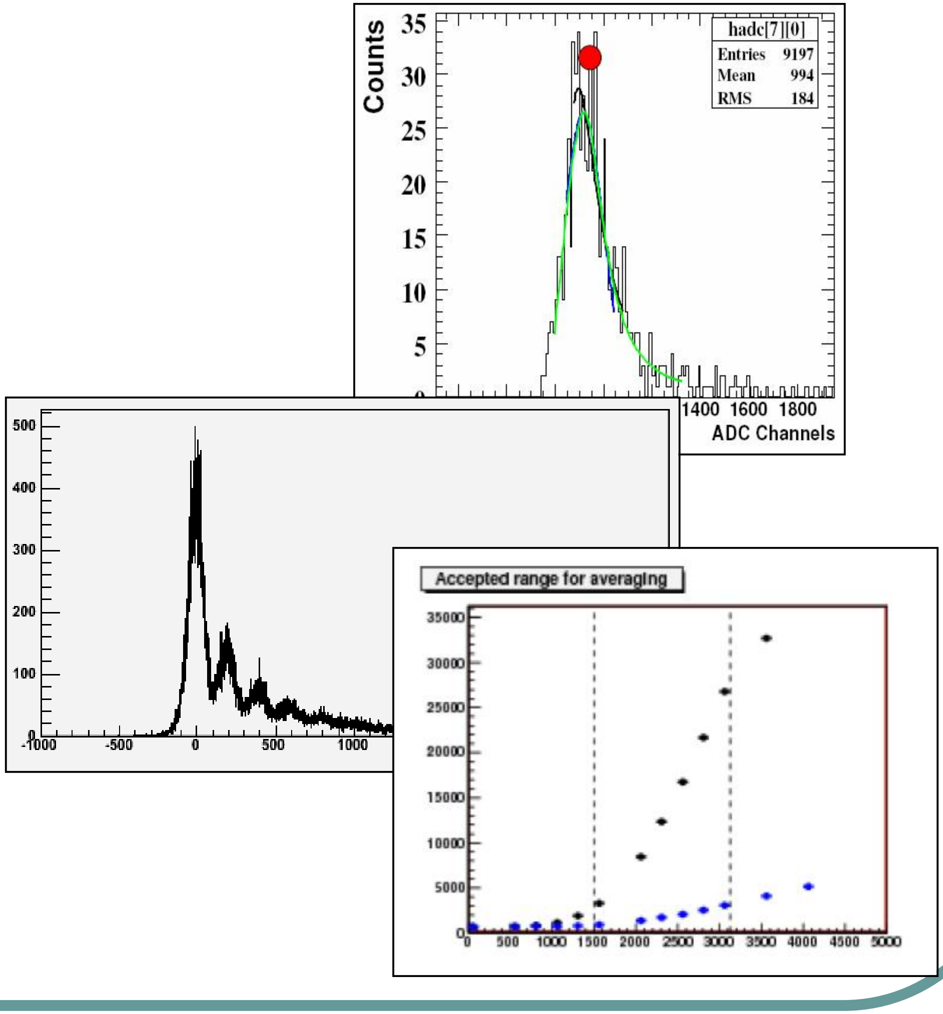
SiPM response function

Light yield of one cell

$$N_{ph.e.} = f_{resp}(N_{pix}) = N_{pix} \cdot f_{corr}(N_{pix})$$

calibration measurements

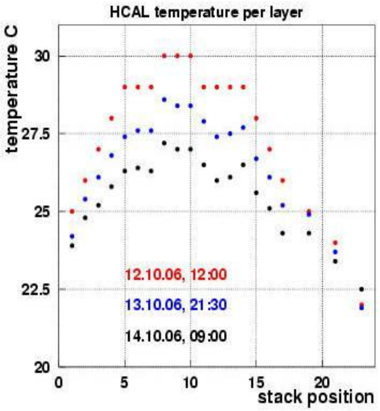
- Pedestal subtraction
  - Estimated from pedestal events
- MIP calibration
  - E.g. from dedicated muon beam runs
  - A_{MIP} [ADC]
- SiPM gain calibration
  - Fits to single photon peaks
  - Electronics operates in calibration mode
  - Time consuming (about 2h/module)
  - Runs on the grid
  - G_{pix} [ADC/pixel]
- Electronics intercalibration between physics mode and calibration mode
  - Using LED light scans



prerequisites

- Finally about 8000 cell for the HCAL only
- Every cell has to be calibrated individually
- Keep track of changing cables, module positions, detector parameters
- Temperature dependencies
- Multi step calibration: Pedestal subtraction, MIP calibration, gain calibration, intercalibration, saturation corrections
  - Different algorithms available for each step
  - Different people working on every step

➡ Implementation of a modular framework
  - MARLIN based (module = processor, set of hits/objects = collection)
  - Plug-in different modules
  - Time (and version) dependent conditions data and calibration constants provided by central data base (LCCD)



general structure of HCAL reconstruction

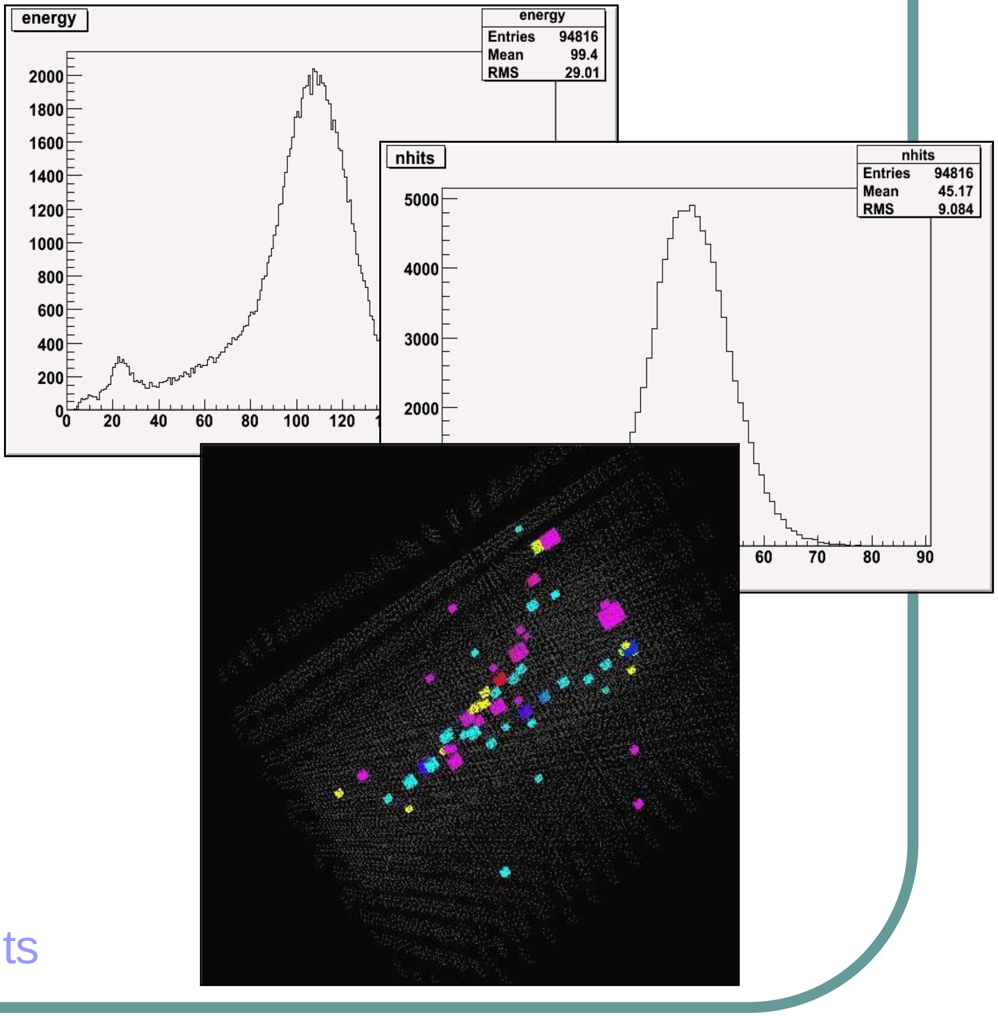
- Calibration depends on moduleID/chip/channel
  - ModuleID: unique identifier for a piece of hardware
  - Every cell on module identified by chip/channel
  - Every module connected to a certain DAQ crate/slot/frontend
  - ➡ Two step mapping between electronic channel and x/y/z necessary
- mappingIProcessor
  - crate/slot/frontend/chip/channel → moduleID/chip/channel
  - ADCBlocks → CaliceHits
- *n* CalibrationProcessors
  - CaliceHits → CaliceHits
- mappingIIProcessor
  - moduleID/chip/channel → x/y/z
  - CaliceHits → CalorimeterHits

calibration interface

- Calibration is stored in module wise collections in the data base
  - No need to calibrate all modules at the same time
  - Different calibration methods can be applied for different modules
- Calibration is stored in LCHcalCalibrationObjects
  - Contains certain calibration data for a single cell
  - One collection of HcalCalbrationObjects exists in the database per module and per type of calibration
  - The interface does not define any data structure (user has free choice)
  - Interface allows to apply the calibration to a cell hit
    - `float applyCalibration (inputValue)`
      - Applies a calibration to a cell “energy”
    - `float applyCalibrationError (inputValue, inputError)`
      - Error propagation considering the errors of calibration and “energy”
    - `bool calibrationValid`
      - Is a calibration value available for a cell?
    - `bool keepEvent (resultValue, resultError)`
      - Zero suppression after applying the calibration
- Application of a calibration to a collection by a CalibrationProcessor

complete reconstruction chain

- MappingI
  - ADCBlocks → CaliceHits1
- PedestalCalibration
  - CaliceHits1 → CaliceHits2
- GainCalibration
  - CaliceHits2 → CaliceHits3
- InterCalibration
  - CaliceHits3 → CaliceHits4
- SaturationCorrection
  - CaliceHits2, CaliceHits4 → CaliceHits5
- MIPCalibration
  - CaliceHits5 → CaliceHits6
- MappingII
  - CaliceHits6 → CalorimeterHits



technical issues

- Precious experience with LC software in production environment
- LCIO performance
  - Lower performance compared to a stand-alone analysis on raw files
  - Poor performance of user defined types (based on LCGenericObject), i.e. those are not suited to store large amounts of hits, tracks, …
- LCCD scaling
  - Opens a separate connection to the global data base for every conditions data collection (here: one per module and per calibration)
  - Does not know about “connection idle times”, which are needed on MySQL server side in a production environment (problems together with dcache)
- Overall: LC software has proven to be suitable for “production environment”, nevertheless still possibilities to improve

Fix in next version

summary

- CALICE HCAL is a quite complex and large scale experiment from the software point of view
- We (as one of the first real experiments) are using the Linear Collider software framework for data storage and reconstruction (LCIO, LCCD, Marlin)
- A modular reconstruction framework has been developed, accessing a MySQL data base for conditions data