AHcal and TCMT Software

Reconstruction and Digitization

Niels Meyer
CALICE Software Review
18. December 2007

Reco – Digi – CALICE context

Legend

CALICE agreements:

- LCIO for data storage
- MARLIN for data processing
- Separate treatment of event data (triggered) and conditions data (slow control, ...)
- LCCD as conditions interface

- store and process LCIO files with event data in the grid
- keep conditions data in a database

Native LCIO

Calice classes inheriting LCObject

Data processing:

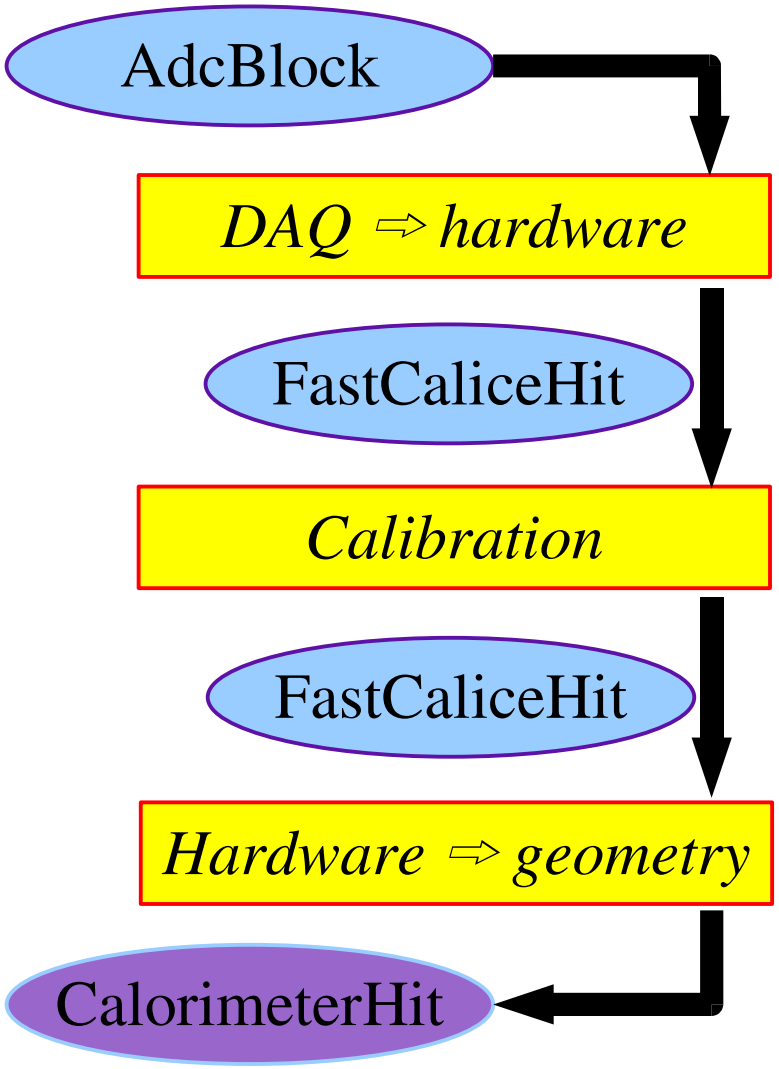
Marlin processors

Data stream:

Event data (triggered)

Conditions data

Reconstruction



Data structures:

Native LCIO

Calice classes inheriting LCObject

Data processing:

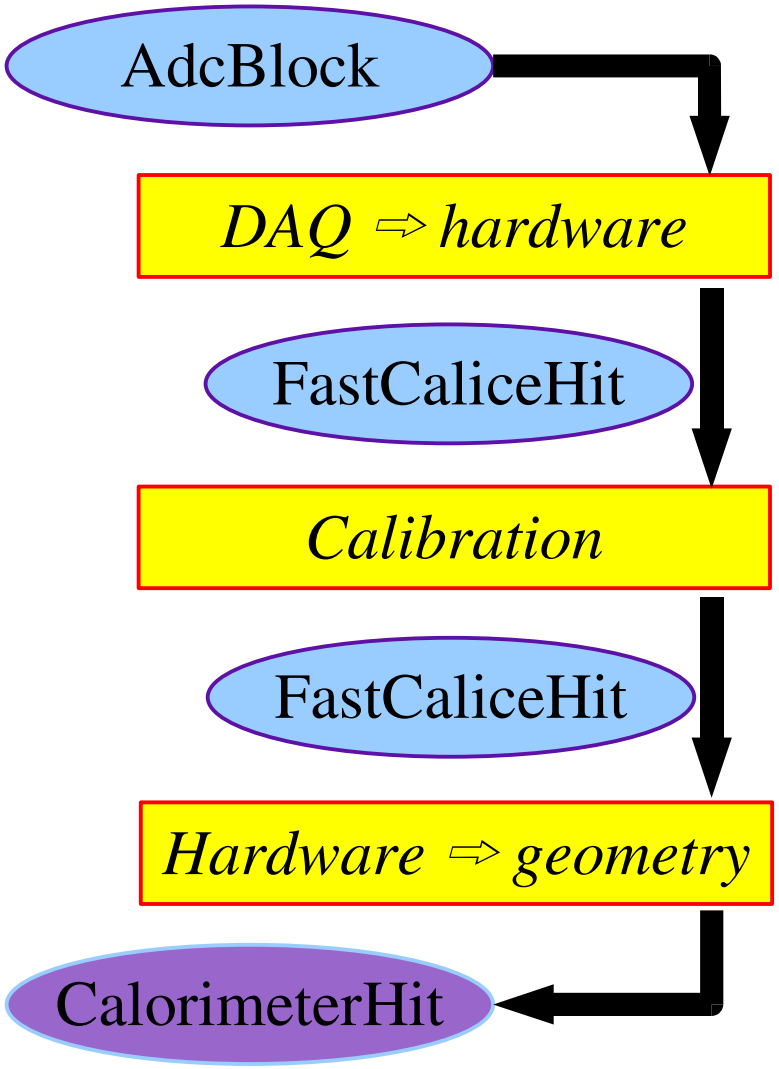
Marlin processors

Data stream:

Event data (triggered)

Conditions data

Reconstruction



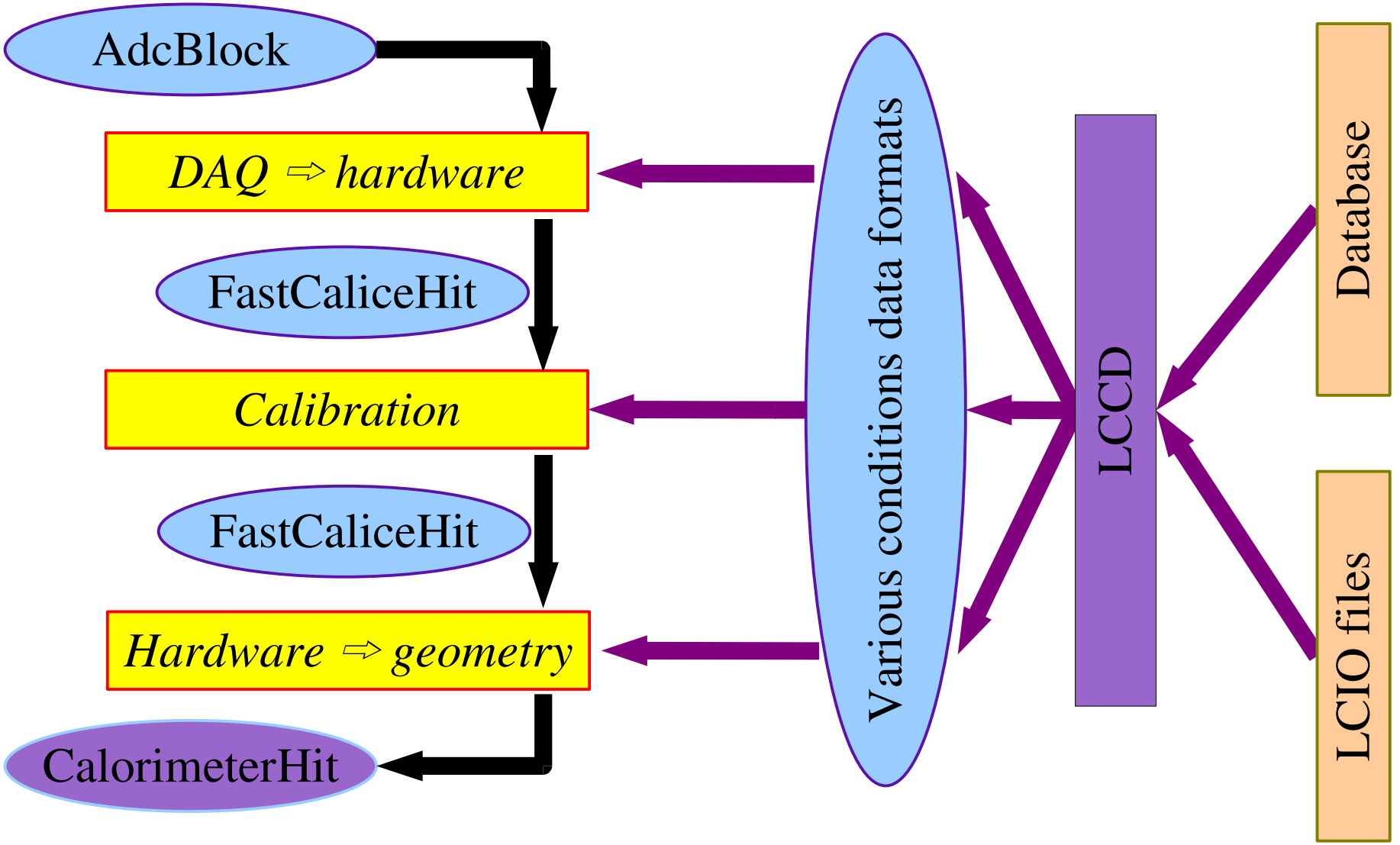
Map modules and front-ends, cretae one object per cell

Internal type with ID, float amplitude, and error

Calibrate amplitude, depends on cell hardware and constants

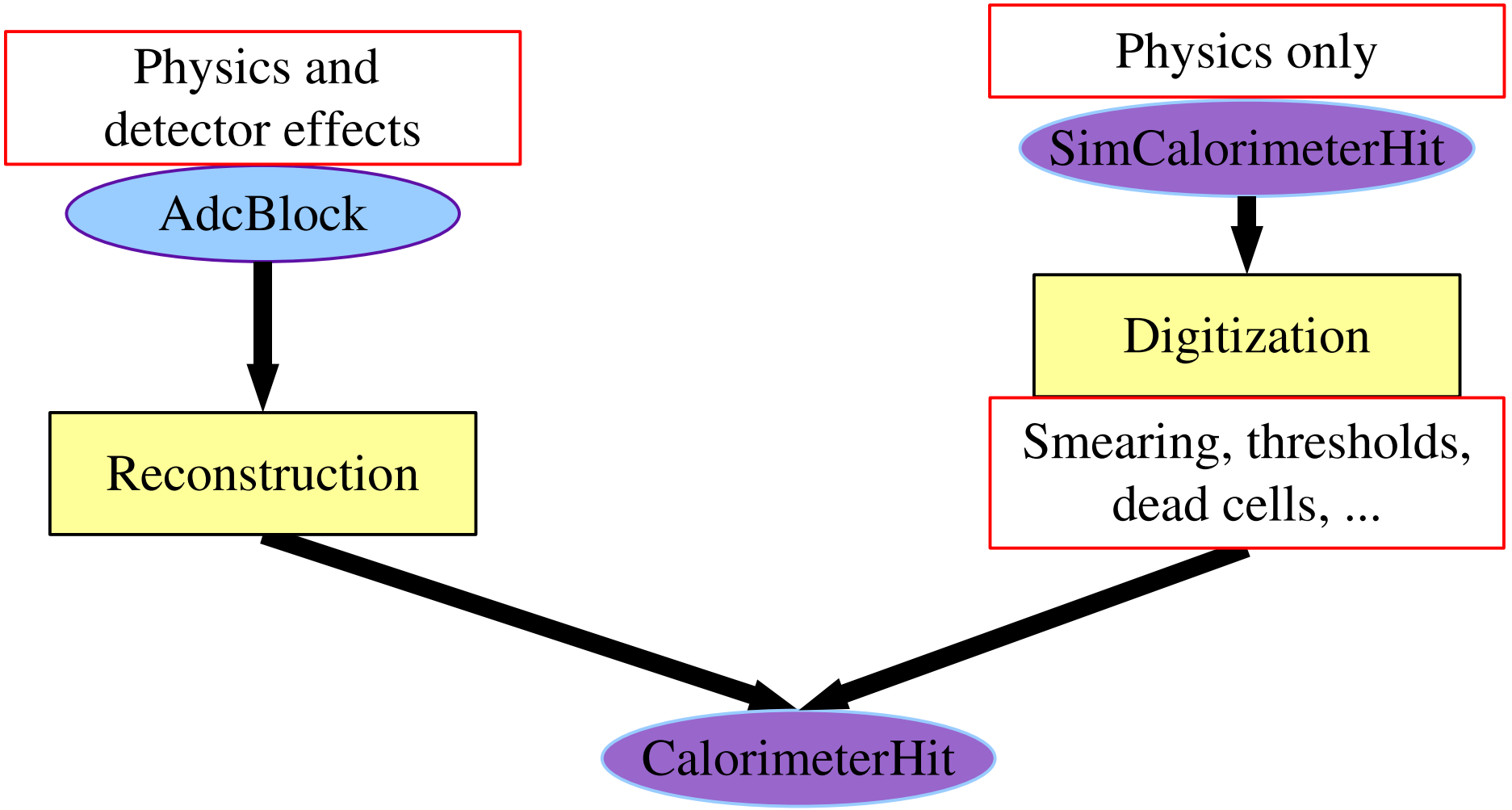
Map cells to geometry, determine 3D position

Reconstruction



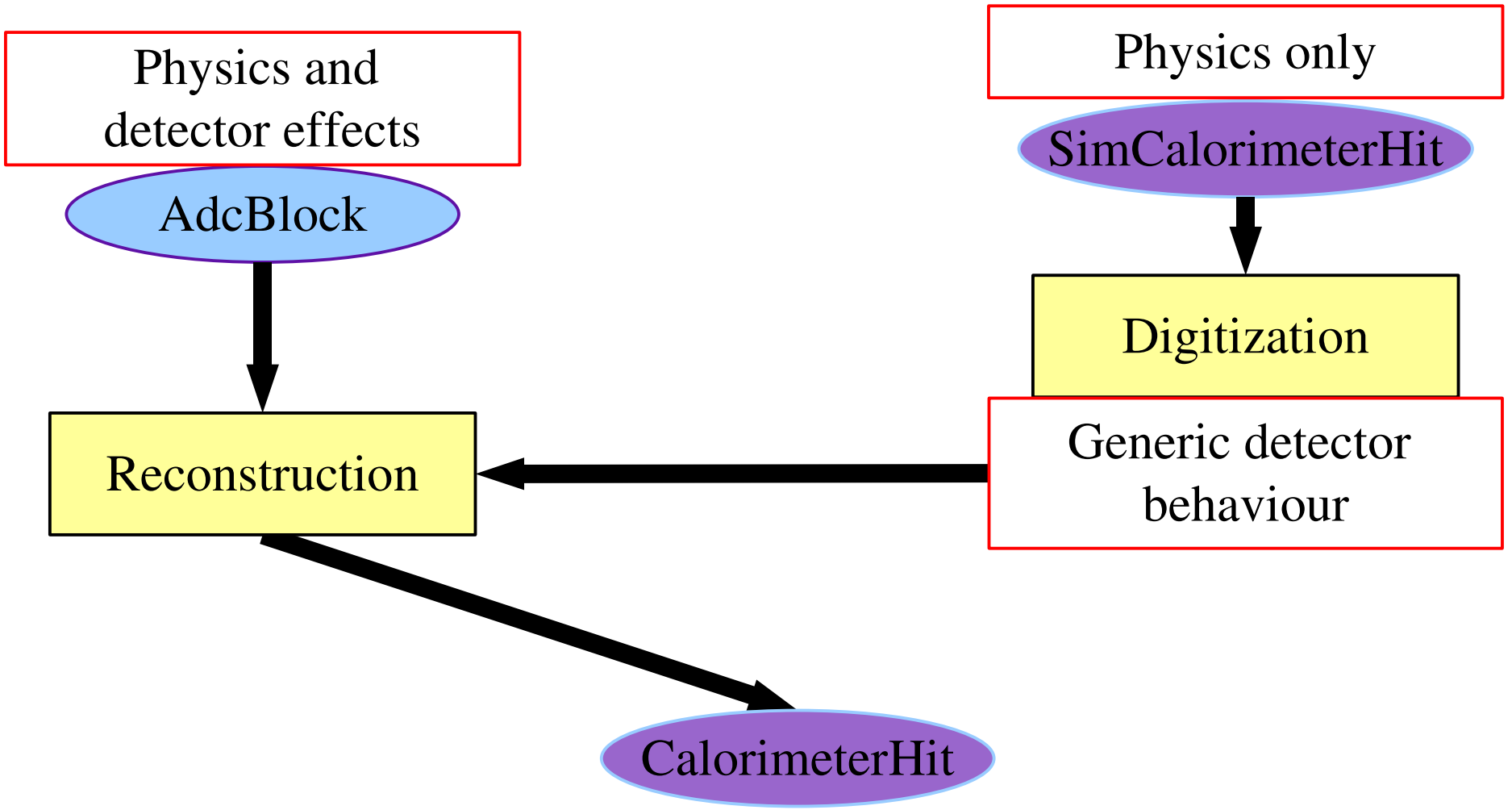
Digitization

Digitization is meant to be MC-equivalent of data reconstruction
Two genral approaches: a) parametric simulation of physics ⊗ reco

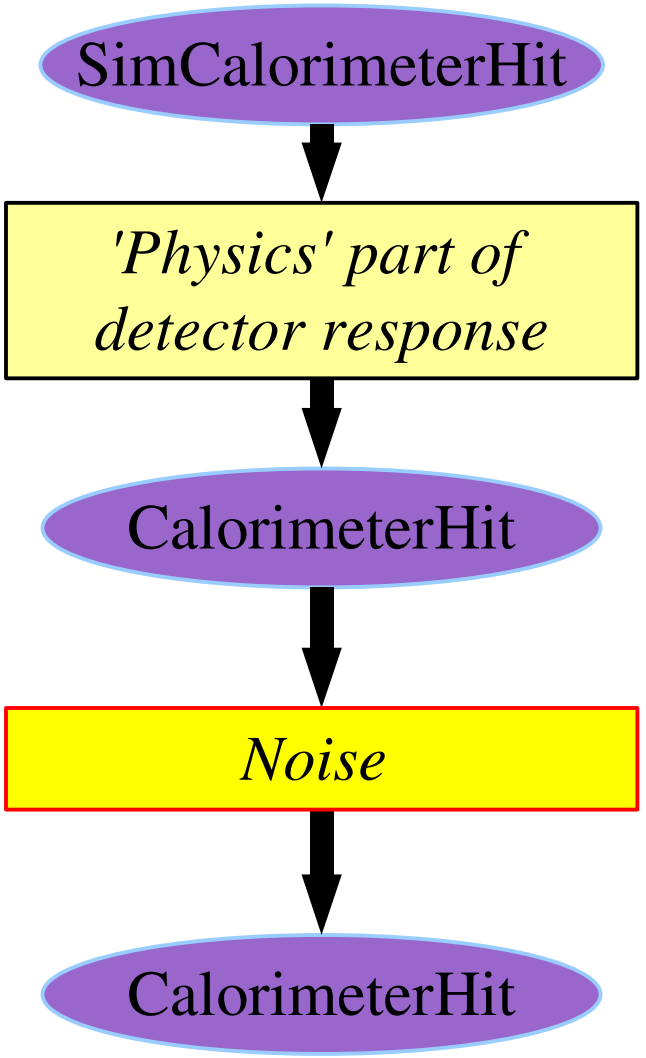


Digitization

Digitization is meant to be MC-equivalent of data reconstruction
Two genral approaches: b) simulation of detector characteristics



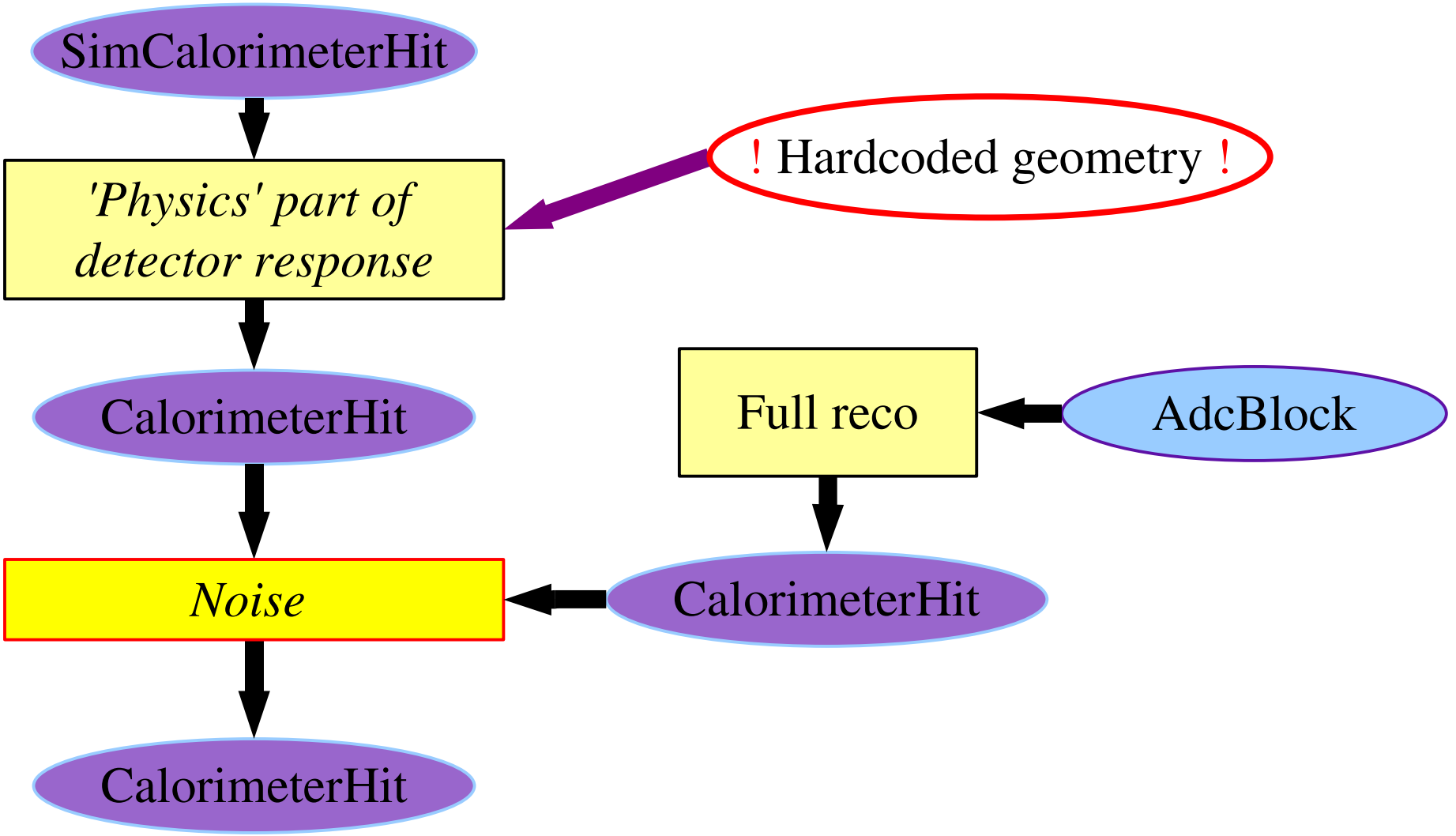
TCMT Digitization



Use DigiSim for:
real cell sizes, optical cross talk,
amplitude smearing, conversion to MIP

Marlin Processor:
overlay MIP-calibrated random trigger,
threshold cut after overlay only

TCMT Digitization

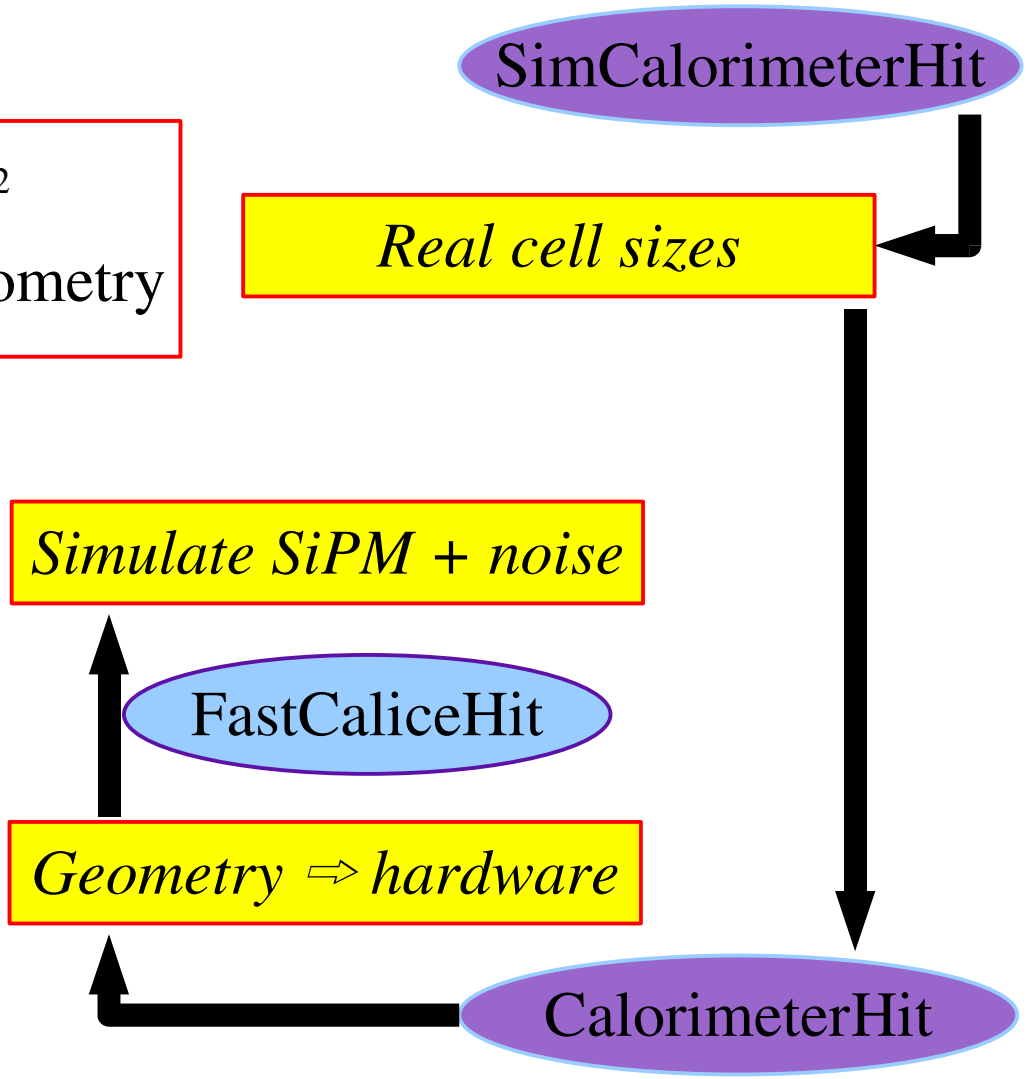


AHcal Digitization

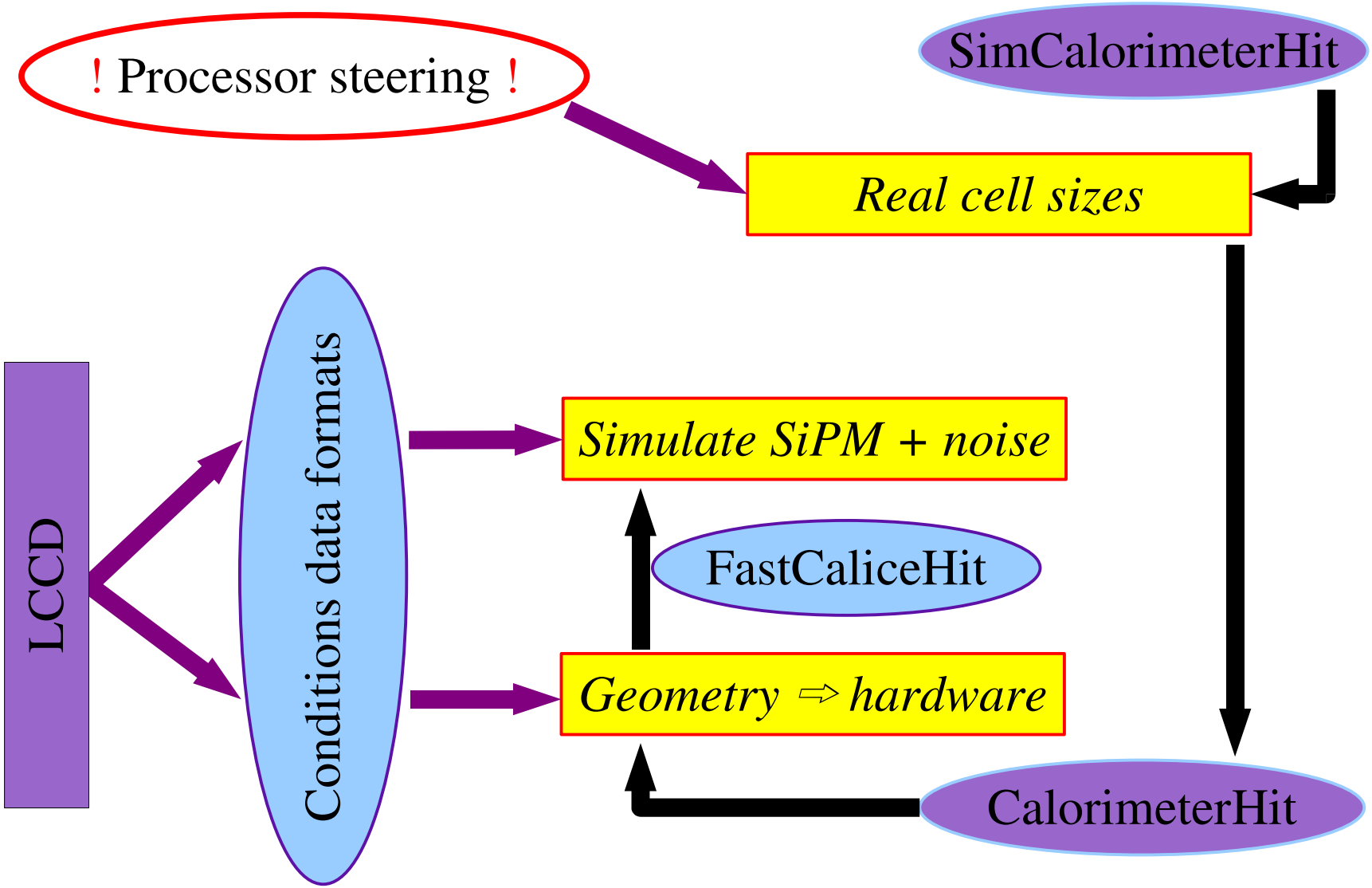
1) Convert virtual 1x1cm^2 MOKKA grid to actual cell geometry

3) Simulate cell behaviour and overlay raw rand. trig.

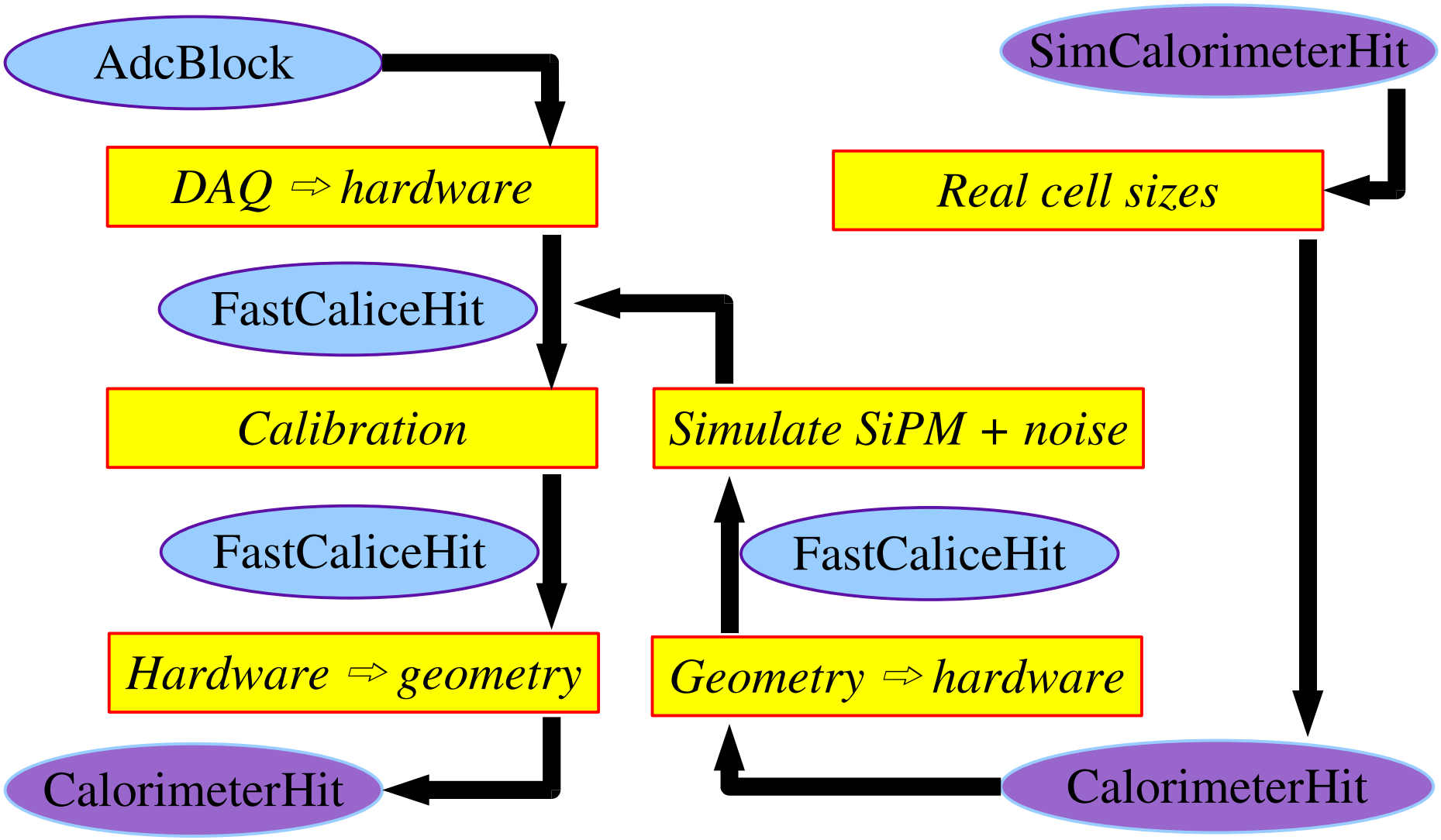
2) Map geometry identifier to SiPM



AHcal Digitization

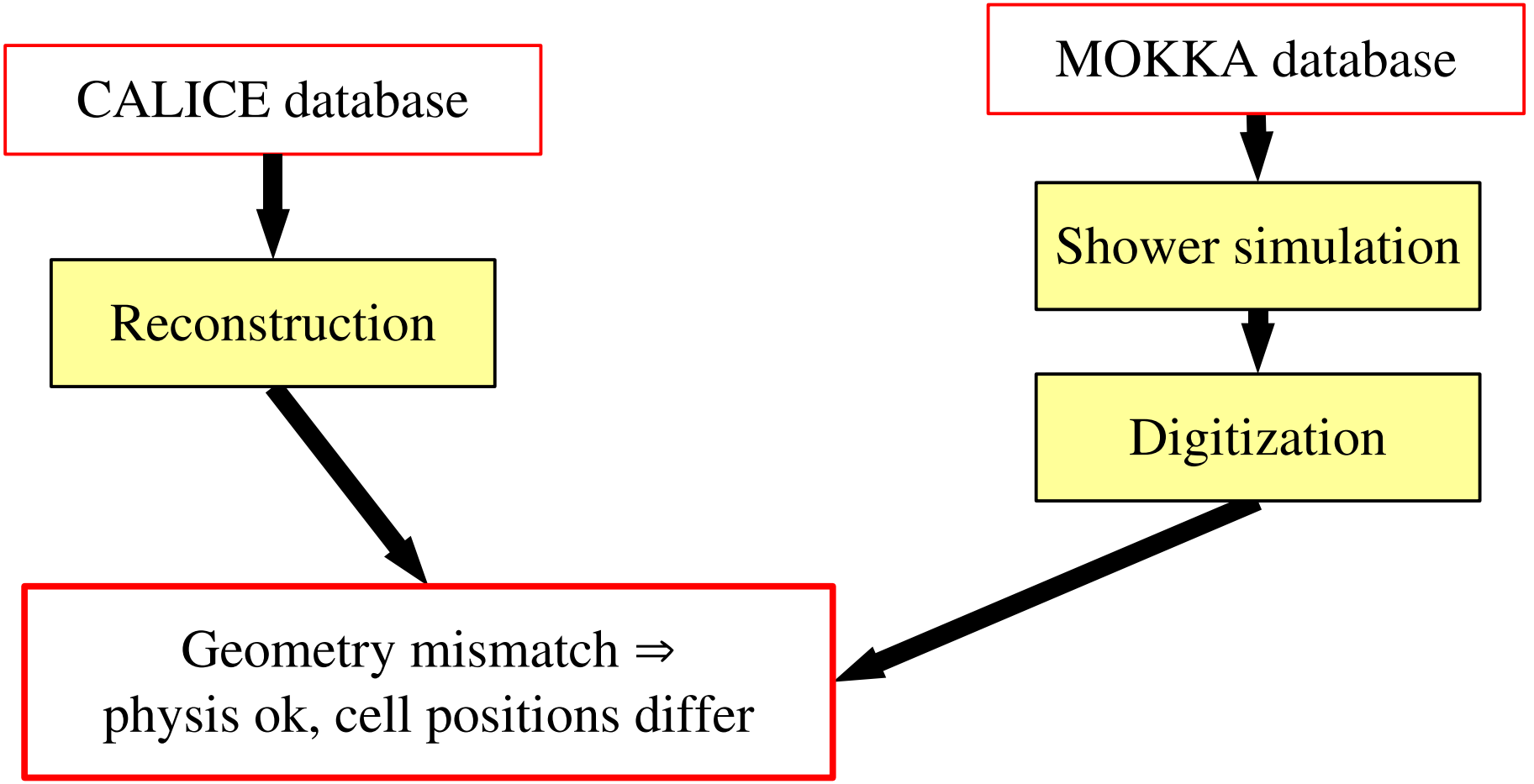


AHcal Digitization



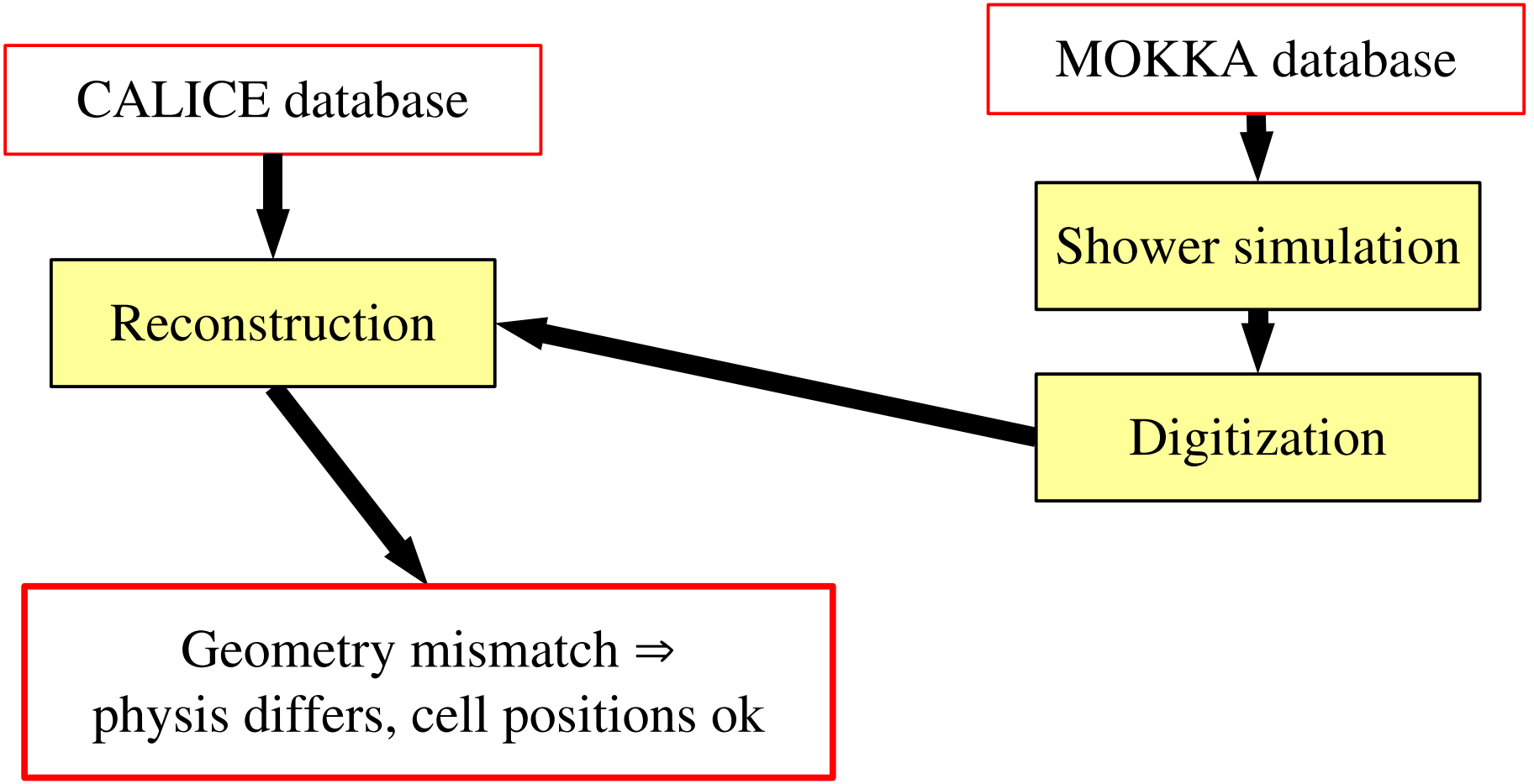
Geometry in Data vs. MC

No unique source of geomtry data for both reco and MC since MOKKA requires own database, which is not suitable for reco needs



Geometry in Data vs. MC

No unique source of geomtry data for both reco and MC since MOKKA requires own database, which is not suitable for reco needs



AHcal/TCMT and CALICE

Follow CALICE decision for data storage and processing

Still some implementation holes in TCMT reco and digitization

Database is unique storage place for all approved conditions for reco

Have to beware of geometry inconsistencies between data and MC

Only one synergy with central CALICE code: mapping and alignment
- clearly inspired by SiW ECal, not always as intuitive for others
- stored values not intuitive -> hard debugging of complex geometries (since regular and complete cell pattern is implied)
- patched for TCMT (written for unique module ⇔ CRC connection)

All operations embedded in processors, not in universal code with processors as interfaces to data structures ⇒ difficult to transfer algorithms to other applications (full simulation, other experiments)

Additional information: Detailed charts of classes and data-flow

FastCaliceHit

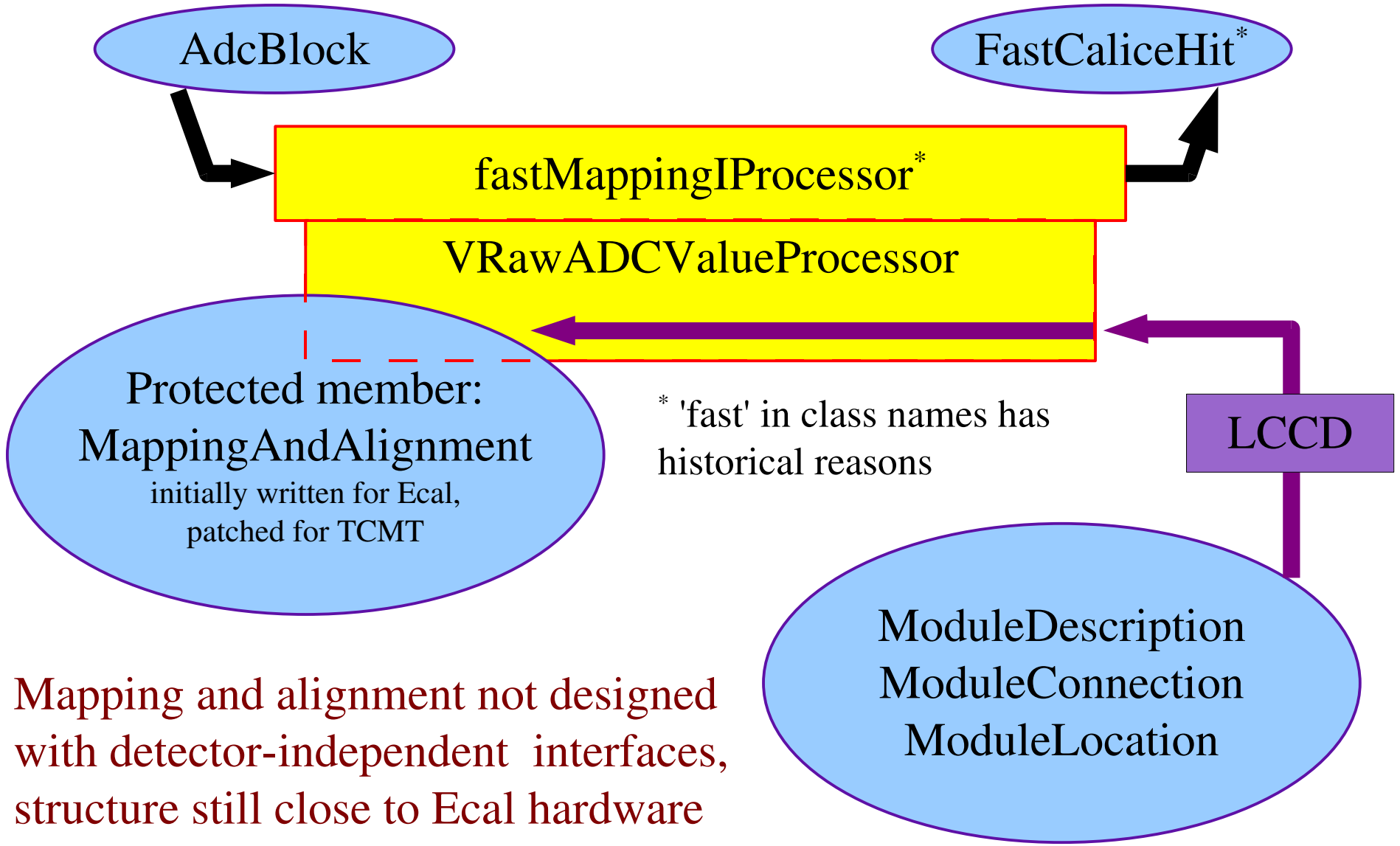
Introduced since no LCIO class offers functionality required:

- originally, all calibration steps were planned as individual processors with same hit type as input and output
- requires float amplitude
- amplitude error desirable for propagation of calibration uncertainties

Candidates:

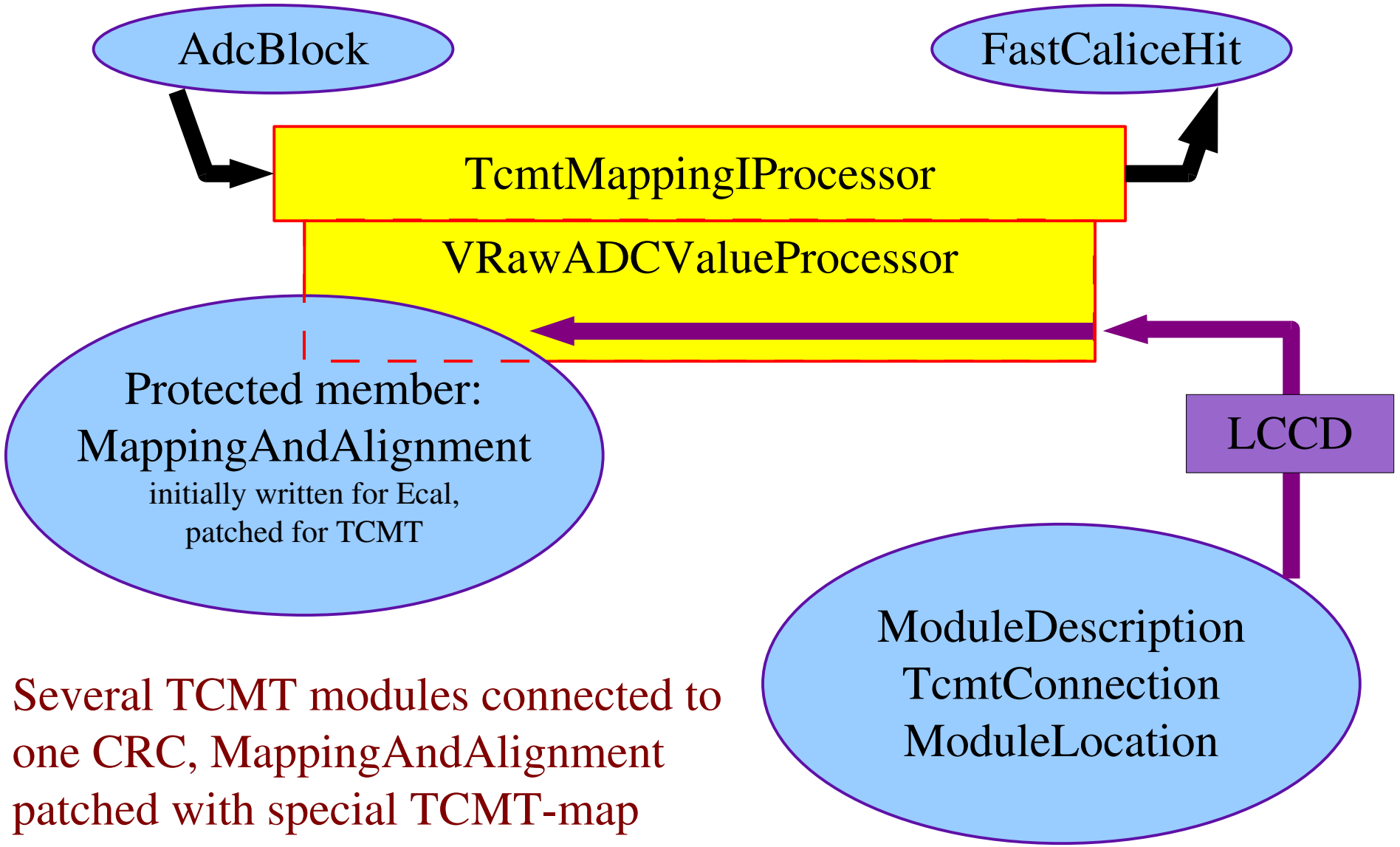
RawCalorimeterHit has int amplitude (designed for ADC channels)
CalorimeterHit has many more fields (position, type)
both lack amplitude error

DAQ ⇨ Hardware: AHcal



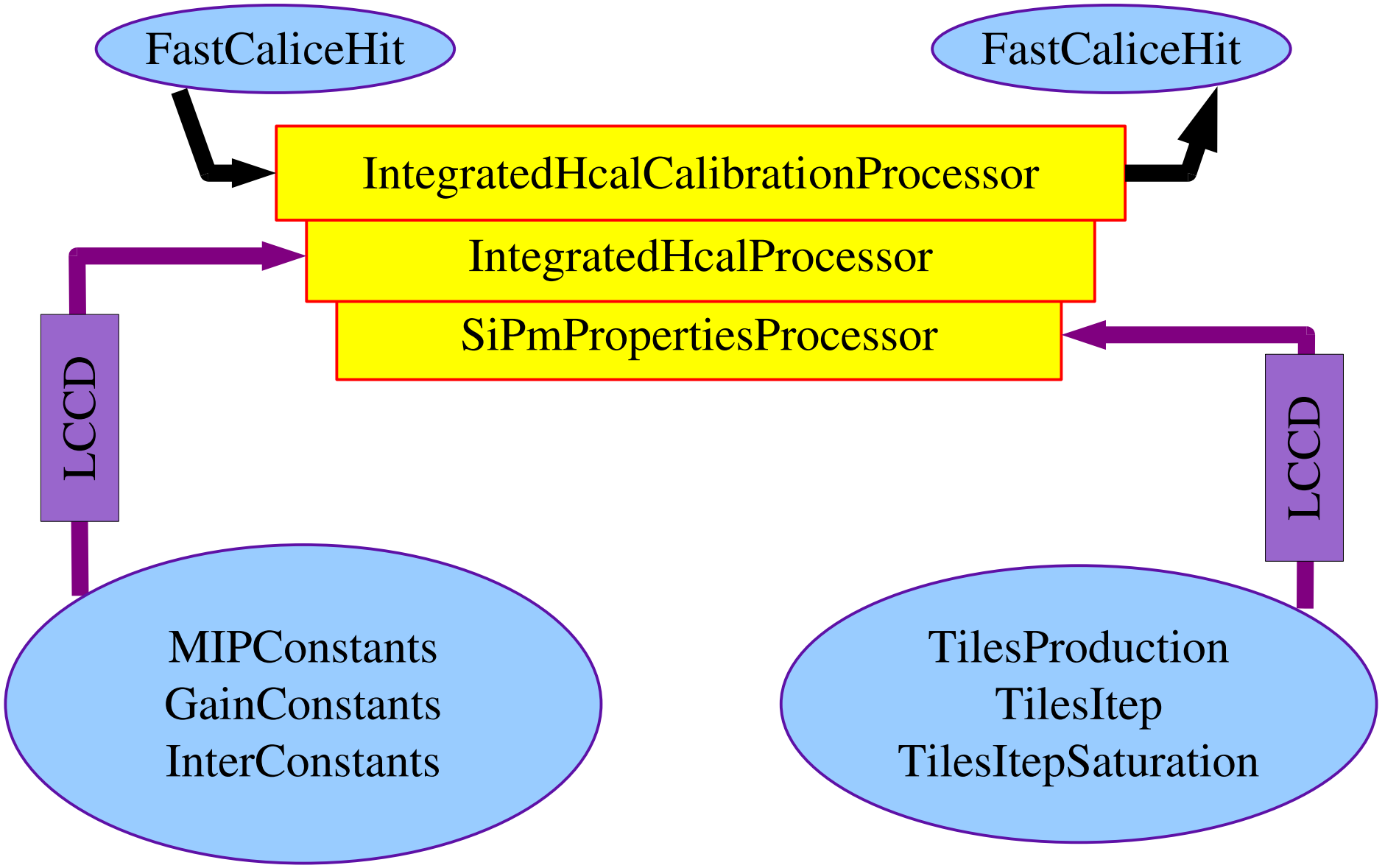
Mapping and alignment not designed with detector-independent interfaces, structure still close to Ecal hardware

DAQ ⇨ Hardware: TCMT

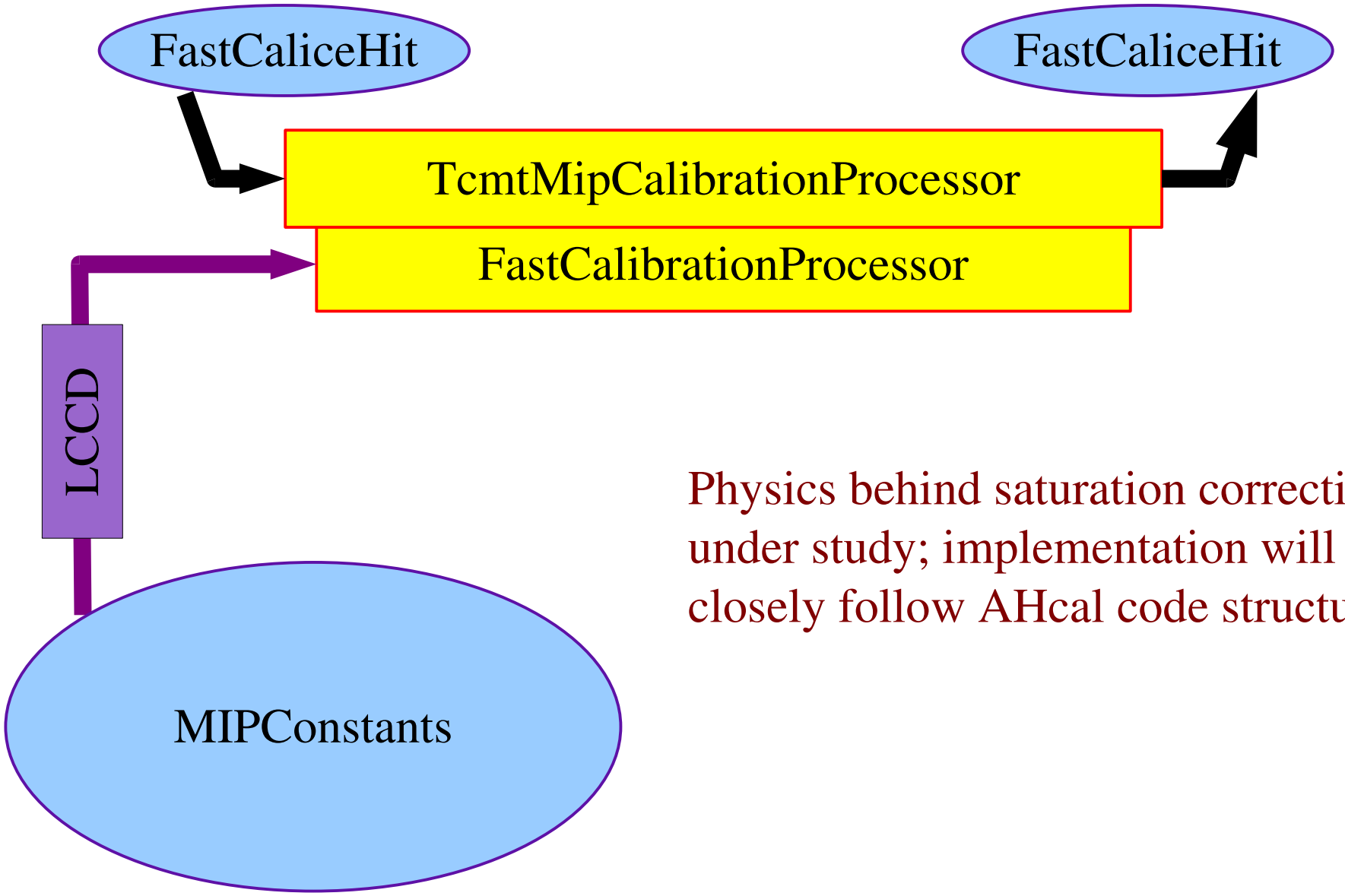


Several TCMT modules connected to one CRC, MappingAndAlignment patched with special TCMT-map

Calibration: AHcal

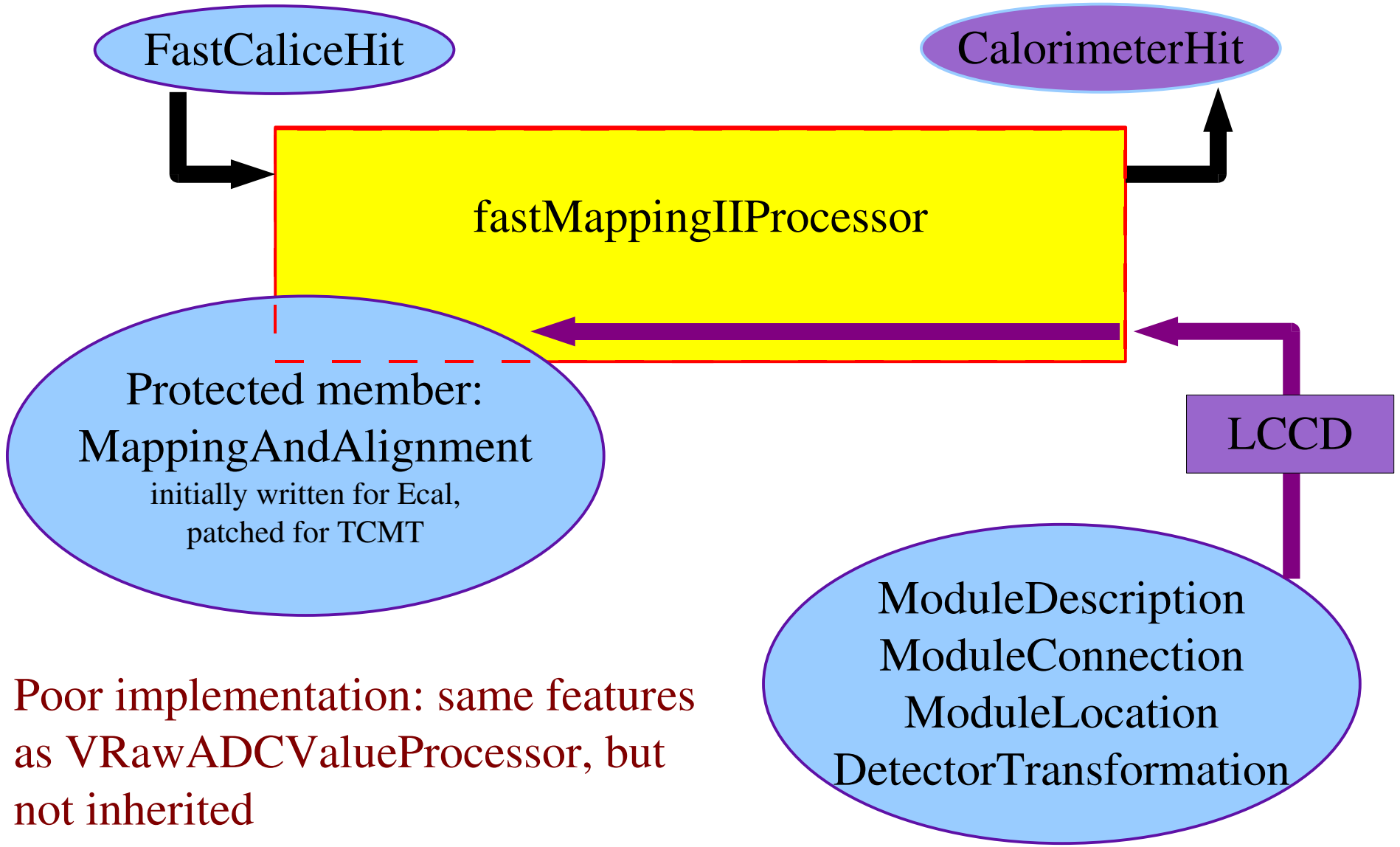


Calibration: TCMT



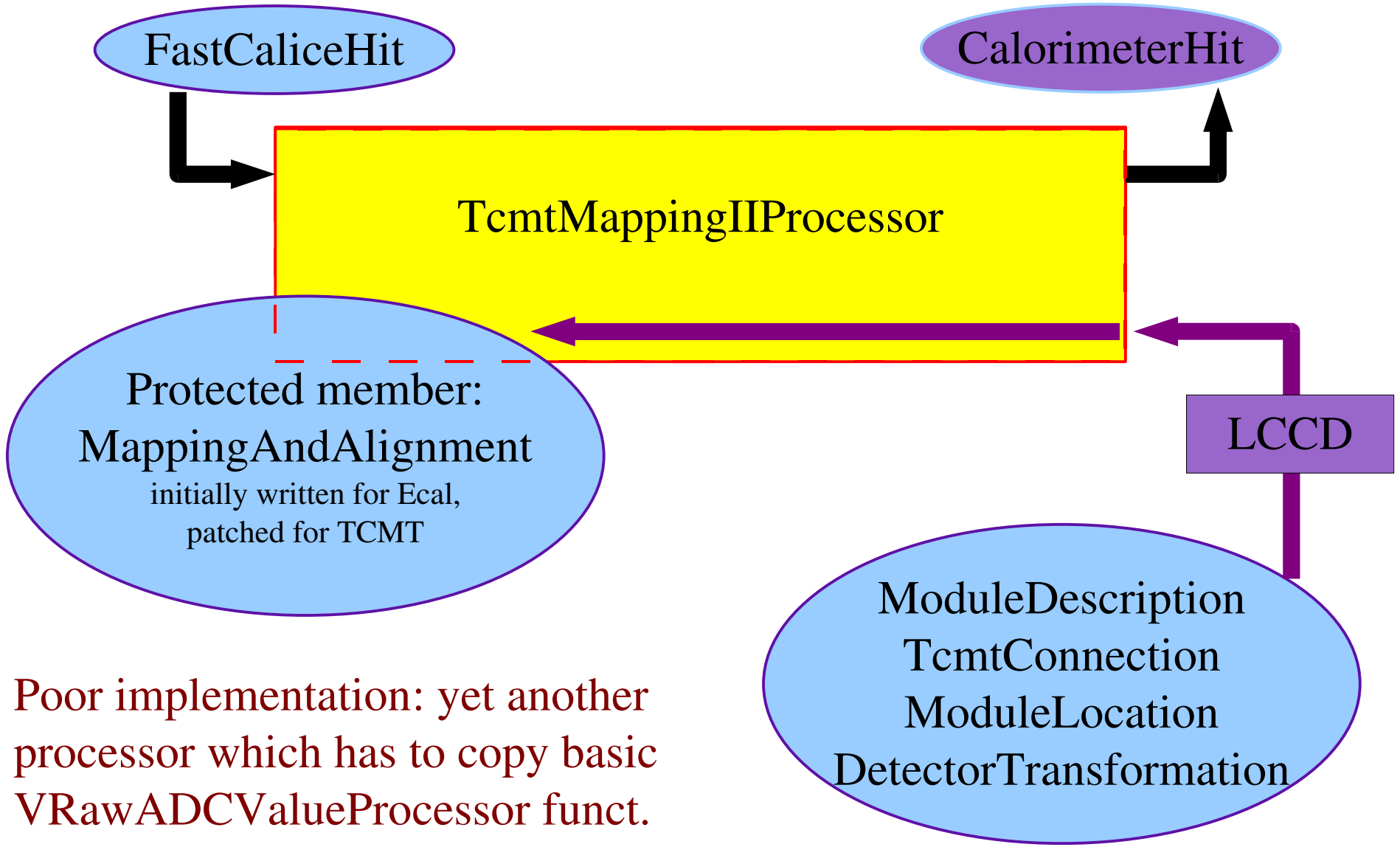
Physics behind saturation correction under study; implementation will closely follow AHcal code structure

Hardware ⇨ Geometry: AHcal



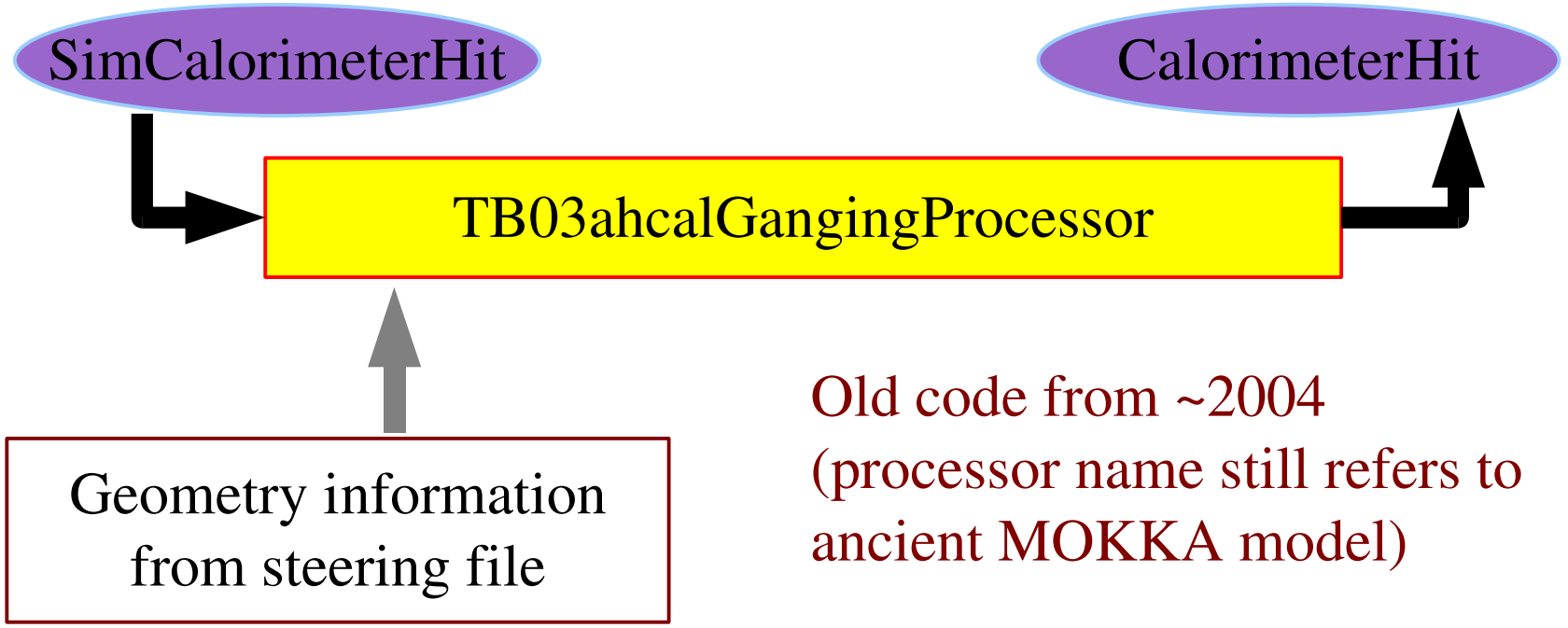
Poor implementation: same features as VRawADCValueProcessor, but not inherited

Hardware ⇨ Geometry: TCMT



Poor implementation: yet another processor which has to copy basic VRawADCValueProcessor funct.

Real Cell Sizes: AHcal

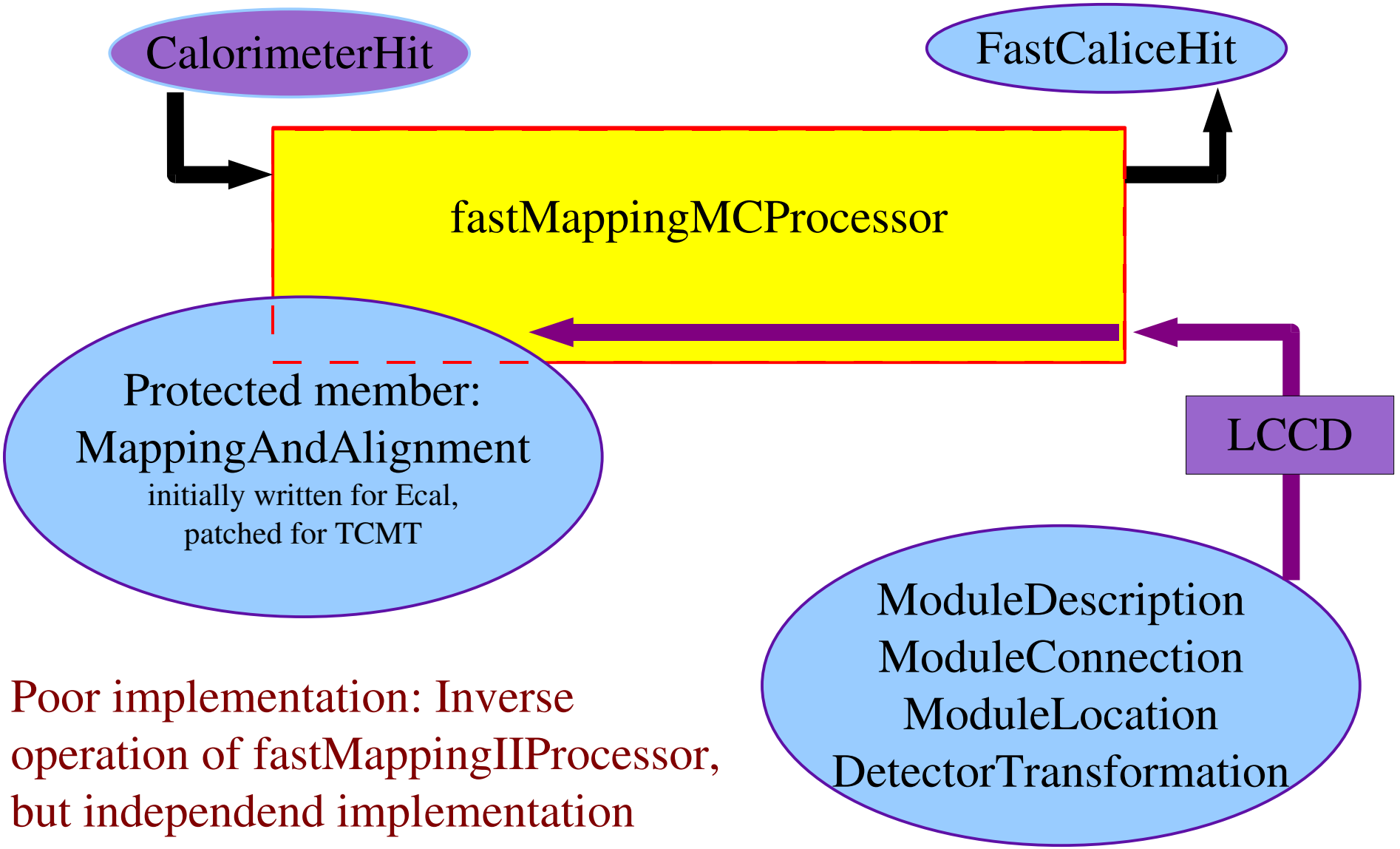


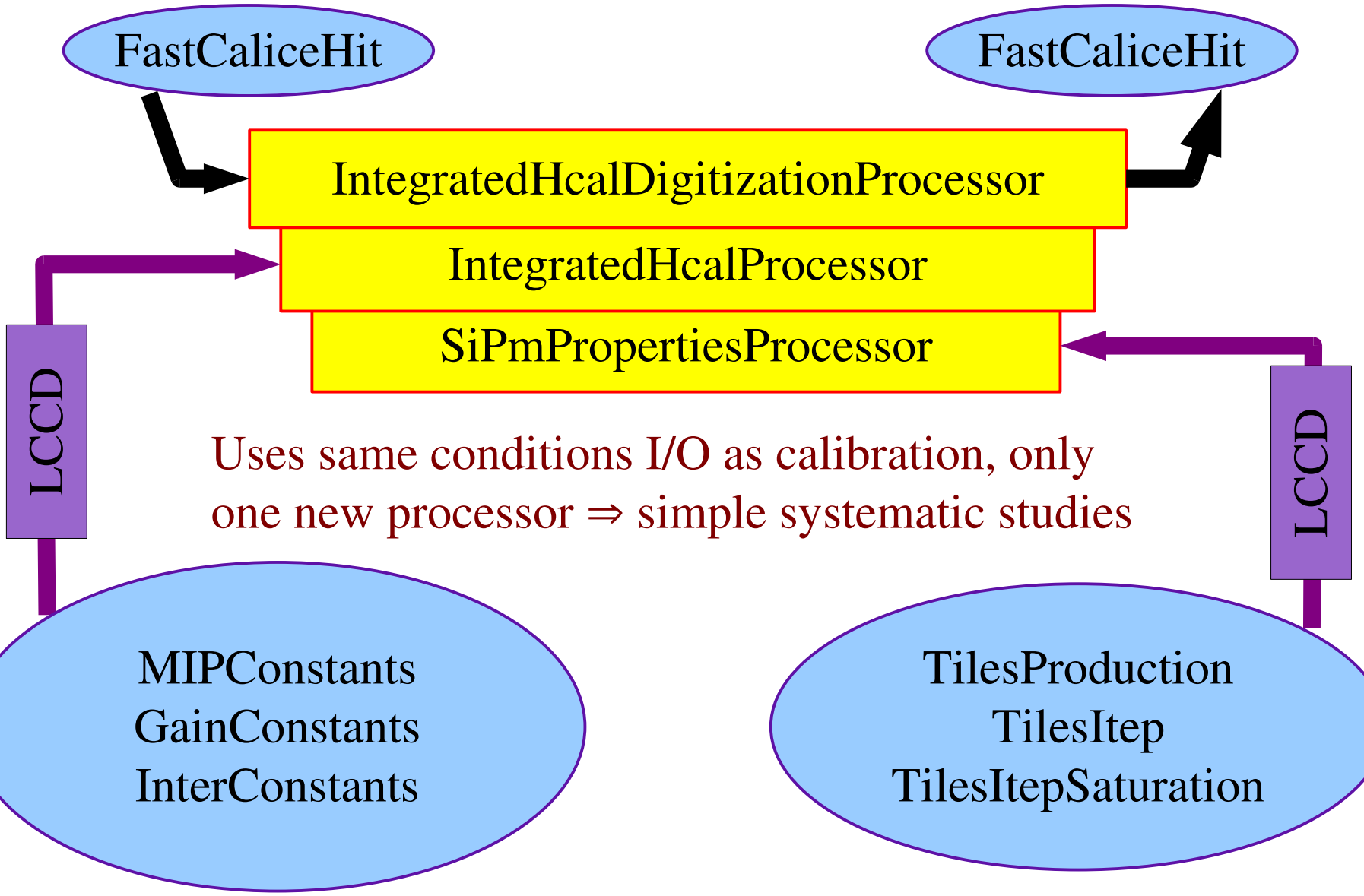
Old code from ~2004 (processor name still refers to ancient MOKKA model)

No template steering for coarse modules, unexperienced user can easily screw up

Not updated since module building plans usually do not change – stable after initial debugging

Geometry ⇨ Hardware: AHcal



Poor implementation: Inverse operation of fastMappingIIProcessor, but independend implementation

Simulate SiPM + Noise: AHcal