

Detector algorithms and preprocessors

Boris Polishchuk

PHOS Detector Algorithms

- **DA1** (“**Calibration DA**”)
 - fills histograms (if histograms from the previous run exists, continue to fill them):
 - amplitude vs time for High and Low Gain.
TH2F* fTimeEnergy[64][56][2]
 - HG/LH ratios (one TH1F per cell).
TH1F* fHgLgRatio[64][56]
 - **EOR**: export file PHOS_Module2_Calib.root to FXS
- **DA2** (“**Bad channels map**”)
 - fills histograms:
 - “Quality” for High and Low Gain (one per cell)
TH1F* fHQuality[64][56][2]
 - Agreement for DAQ and HLT: $0 < \langle \text{Quality} \rangle < 1 \Rightarrow$ good
 - **EOR**: export file PHOS_Module2_BCM.root to FXS

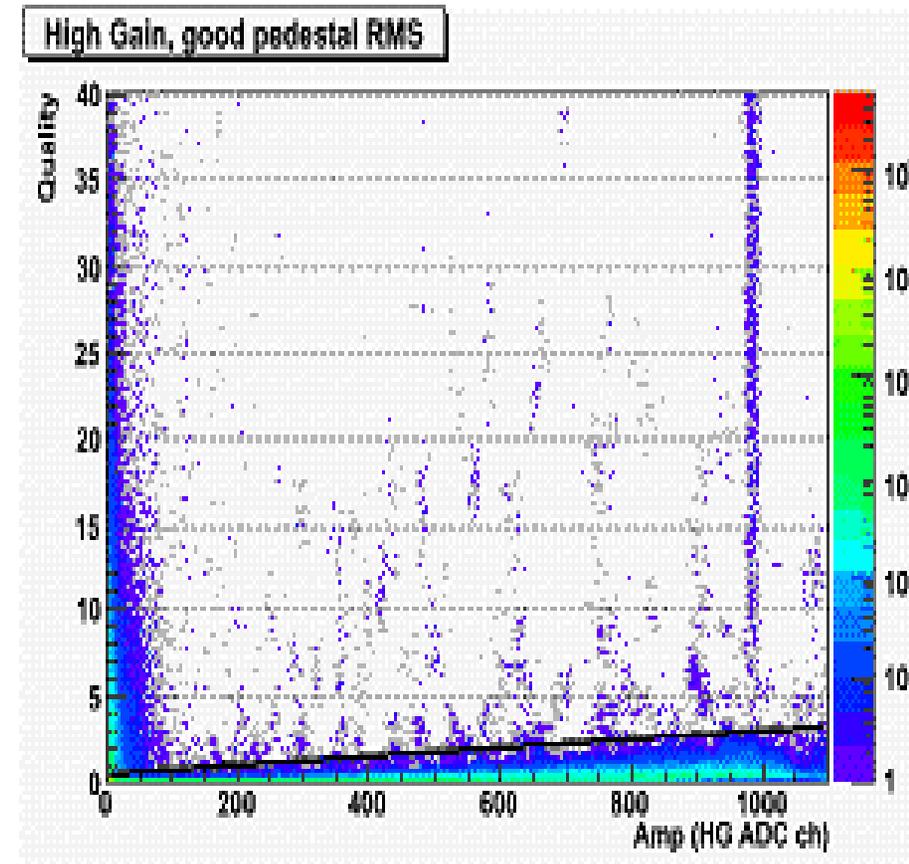
PHOSda2.cxx for DAQ

- Histograms fills by the “Quality” estimations as returned by

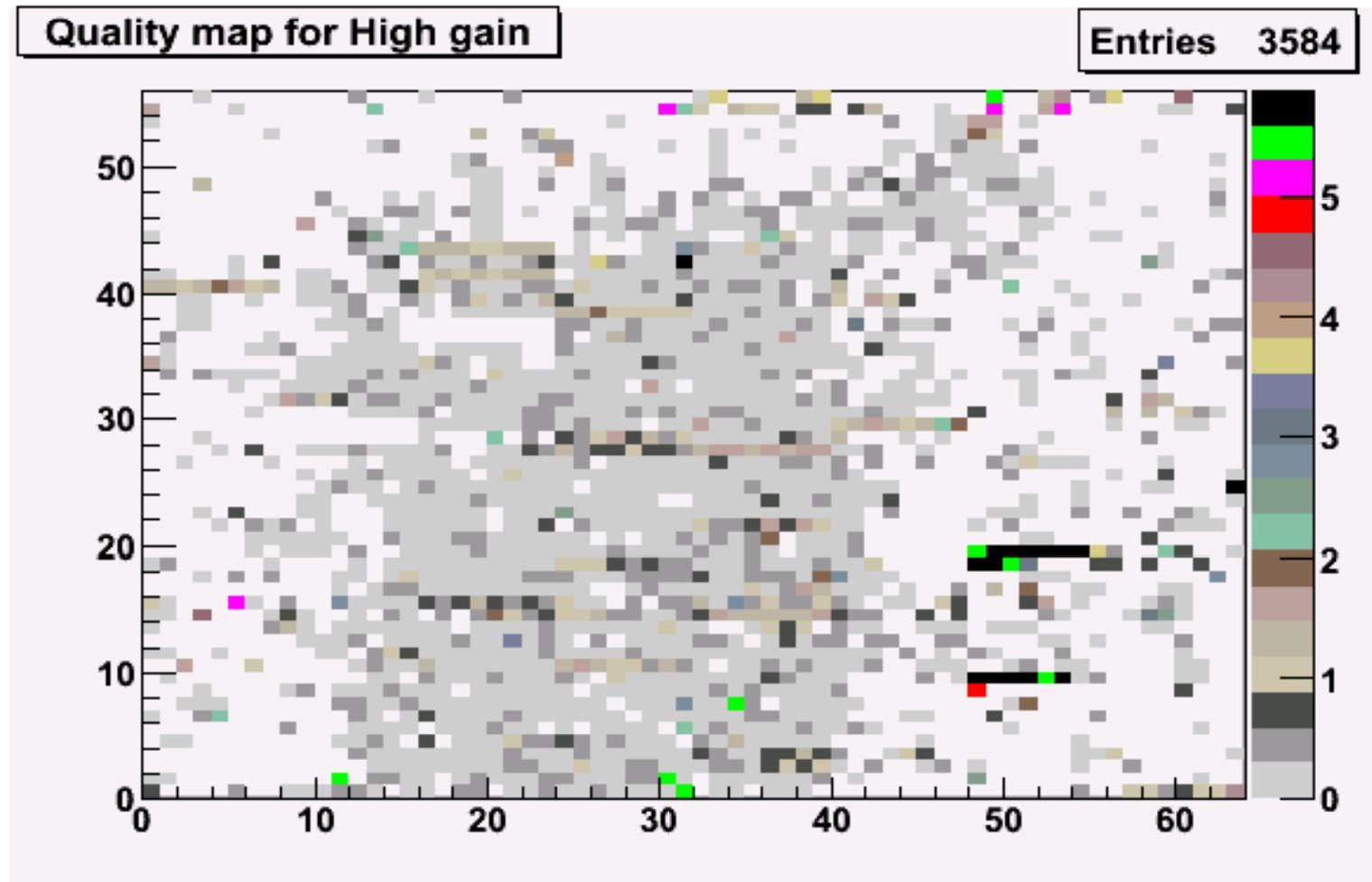
`AliPHOSRawDecoderv1::GetQuality()`:

- if `Amp < 5` return **999** (bit flip) or **0**;
- if `fit fails` return **999**;
- if **OK**: return `[chi2/NDF]/cut`

$$\text{cut} = 0.75 + 0.0025 * \text{Amp}$$



Quality Map for Run #8249



DA's status

- **AliPHOSDA1** and **AliPHOSDA2** tested in a standalone macros on the raw data files from 2006 beam test and 2007 cosmic runs
- **PHOSda1.cxx** and **PHOSda2.cxx** committed to the AliRoot SVN trunk and ready to launch during the PHOS commissioning at P2.

Not tested in a real DATE stream yet!

PHOS Preprocessor (I)

- **PHYSICS** run:
 - Retrieve “Quality” histograms from FXS and build BCM for the current run: $0 < \langle \text{Quality} \rangle < 1 \Rightarrow \text{good}$. Store BCM to OCDB.
 - Retrieve calibration and HG/LG ratio histograms from FXS
 - Randomly pick a PHOS crystal
 - check for BCM; if cell is good and has enough statistics – choose a correspondent (High Gain) *fTimeEnergy* histogram as a reference
 - Calculate the $[56] \times [64] - 1$ ratios $\mathbf{R}[i][j]$ of mean HG amplitudes in the rest of the module to the mean amplitude of the reference histogram, thus obtaining the relative calibration parameters. Fill AliPHOSEmcCalibData object.
 - Fit HG/LG ratio histograms and fill AliPHOSEmcCalibData obj
 - Store AliPHOSEmcCalibData object to OCDB

PHOS Preprocessor (II)

- **STANDALONE** run (i.e. **LED**)
 - Retrieve “Quality” histograms from FXS
 - Build BCM and fill AliPHOSEmcBadChannelsMap
 - Retrieve HG/LG ratio histograms from FXS
 - Fit HG/LG ratios and fill AliPHOSEmcCalibData obj
 - Store AliPHOSEmcCalibData object to OCDB

PHOS Preprocessor (III)

- **PEDESTAL** run (?)
 - Retrieve the last AIPHOSEmcCalibData obj **o1** available from OCDB
 - Retrieve from FXS the AIPHOSEmcCalibData obj **o2** filled with pedestals
 - Rewrite pedestals in **o1** by values from **o2**
 - Write **o1** to OCDB
 - Now pedestals calculated “on demand” from the non-zero suppressed raw data

PHOS Preprocessor status

- Test with the SHUTTLE test facility and preinstalled histogram files: **passed**
- Whole chain (DAs + preprocessor in a real data taking): **not tested**

CPV Detector Algorithm

- **DA1** (“**Calibration DA**”)
 - amplitudes per pad: `TH1F* fCharge[128][56]`
 - file for FXS: `CPV_Module2_Calib.root`

CPV Preprocessor

- **PHYSICS** run
 - Retrieve calibration histograms from FXS
 - Randomly pick a pad and choose a correspondent fCharge histogram as reference
 - Calculate the ratios R of mean amplitudes to the mean amplitude of the reference histogram, thus obtaining the relative calibration parameters. Fill AliPHOSCpvCalibData object.
 - Store AliPHOSCpvCalibData object to OCDB