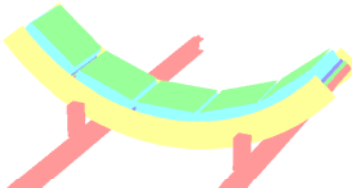
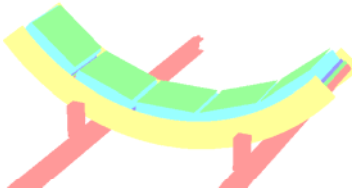


PHOS offline status

Yuri Kharlov

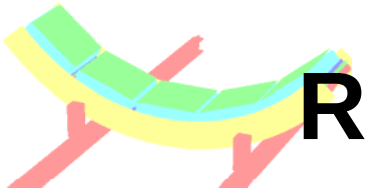
ALICE offline week

8 October 2007



Raw format (EMC)

- PHOS-EMC raw format: ALTRO samples (number of samples per channel is programmable, typically 60-100) which are amplitudes measured by 10-bit ADC with 10-MHz frequency with 2 channels simultaneously:
  - High gain: from 5 MeV to 5 GeV
  - Low gain: from 80 MeV to 80 GeV
- Note: **instead of one signal, PHOS-EMC FEE writes 120-200** amplitudes, which significantly increases the payload.
- Two modes are foreseen in ALTRO:
  - **non-zero suppression** (fixed number of samples): no information loss, but every event contains 2x3584 channels.
  - **zero suppression** (only samples above the baseline after the pedestal subtraction are registered): some information is lost (pedestals and samples below the baseline), but only signals are present in the data stream



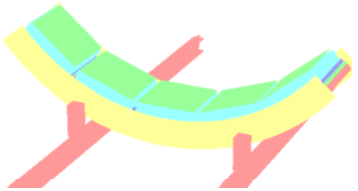
Raw format simulation (EMC)

- Simulation of ALTRO samples: AliPHOSPulseGenerator

```
AliPHOSPulseGenerator pulse;
Int_t *adcValuesLow = new
Int_t[pulse.GetRawFormatTimeBins()];
Int_t *adcValuesHigh= new
Int_t[pulse.GetRawFormatTimeBins()];
pulse.SetAmplitude(energy);
pulse.SetTZero(digit->GetTimeR());
pulse.MakeSamples();
pulse.GetSamples(adcValuesHigh, adcValuesLow) ;
```

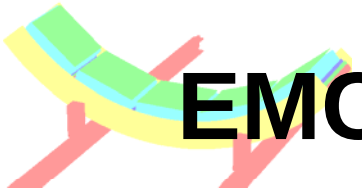
- Simulation of ALTRO payload: AliAltroBuffer
- HW address to geometrical position mapping: AliAltroMapping
- Raw data is simulated in AliPHOS::Digits2Raw().

  Only **zero-suppressed** data in "**New RCU format**" are produced in simulation.



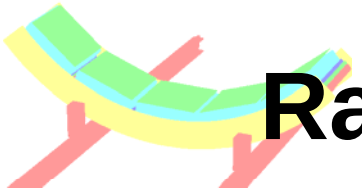
Raw format (CPV)

- PHOS-CPV consists of 5 cathode-pad proportional chambers, each has 7168 channels.
- Group of 48 pads are read by 1 DILOGIC chip and multiplexed
- Group of 10 DILOGIC chips are controlled by 1 row
- One CPV chamber is read by 1 DDL which controls 16 rows
- One word of CPV raw format is 32-bit word consisting of:
  - 6 bits for padding
  - 4 bits for row number (0-15)
  - 4 bits for DILOGIC number (0-9)
  - 6 bits for DILOGIC address (0-47)
  - 12 bits for pad amplitude
- Neither simulation nor decoding of CPV raw data is not implemented yet



EMC raw data decoding (low level)

```
AliRawReaderRoot* rf = new AliRawReaderRoot("rawFile.root");
AliCaloRawStream in(rf,"PHOS");
in.SetOldRCUFormat(kTRUE);
while (rf->NextEvent()) { // read next event
   runNum = rf->GetRunNumber();
   while ( in.Next() ) { // read next sample
      hwAddress = in.IsNewHWAddress();
      gain    = in.IsLowGain(); // 0-high, 1-low
      module  = in.GetModule(); // module number (0,4)
      column  = in.GetColumn(); // z (0,55)
      row     = in.GetRow();    // x (0,63)
      timeAll = in.GetTimeLength(); // tot.num.of samples
      time    = in.GetTime();   // sample time (0,timeAll-1)
      ampl    = in.GetSignal(); // sample amplitude (0,1023)
   }
}
```



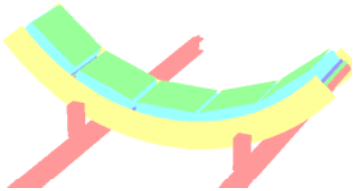
Raw data decoding (high level)

- AliPHOSRawDecoder: decode samples, pass some parameters to decoding
  - SetOldRCUFormat(Bool_t)
  - SubtractPedestals(Bool_t)

  Returns digit energy, time and geometry

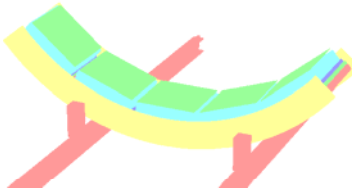
  Here sample processing procedures should be called (crude, peak finder, k-level, Minuit, LMS)
- AliPHOSRawDigiProducer: convert samples to TClonesArray of digits

- Raw data decoding (either high- or low-level) can be used in any raw data processing procedures (DA, DQM, Reconstruction)



On-line data processing

- PHOSda.cxx: one of the detector algorithms (calibration by equalization of mean deposited energy per crystal)
- AliPHOSCalibHistoProducer: creates histograms needed for calibration
- AliPHOSPreprocessor: runs from **Shuttle**, reads histograms accumulated by AliPHOSCalibHistoProducer calculates calibration parameters and creates **OCDB** objects



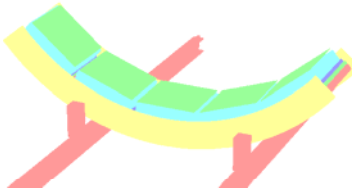
On-line: use case

```
AliRawReader* rf = new
                   AliRawReaderRoot("rawData.root");

AliPHOSCalibHistoProducer hp;
hp.SetOldRCUFormat(isOldRCUFormat);
hp.SetUpdatingRate(2000);

AliPHOSRawDecoder dc(rf);
hp.SetRawDecoder(&dc);

while(rf->NextEvent()) {
   hp.Run();
   nEvents++;
}
```



Off-line reconstruction

- AliReconstruction: steering class for reconstruction from raw data to ESD. It takes care about the event loop and provides raw reader, tree of digits, tree of clusters, ESD.
- Each detector provides a class AliDETReconstructor which has methods to make the following tasks per event:
  - Create digits from raw data
  - Make clusters from digits
  - Make ESD from clusters
- AliPHOSReconstruction has the following methods (*thanks to Cvetan*!):

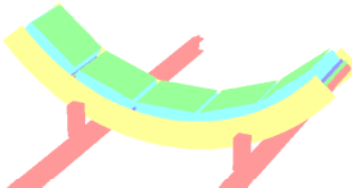
- **`AliPHOSReconstructor::ConvertDigits(AliRawReader* rawReader, TTree* digitsTree)`**
- **`AliPHOSReconstructor::Reconstruct(TTree* digitsTree, TTree* clustersTree)`**
- **`AliPHOSReconstructor::FillESD(TTree* digitsTree, TTree* clustersTree, AliESDEvent* esd)`**

Off-line reconstruction: use case

```
AliReconstruction rec ;
rec.SetOption("PHOS","OldRCUFormat");
rec.SetRunTracking("") ;
rec.SetRunVertexFinder(kFALSE) ;
rec.SetRunLocalReconstruction("PHOS") ;
rec.SetFillESD("PHOS") ;
rec.SetInput(“rawData.root”);
rec.Run();
```

PHOS reconstruction parameters classes

- Special class AliPHOSRecoParam and it's derivatives AliPHOSRecoParamEmc and AliPHOSRecoParamCpv
- Objectives:
  - need to tune reconstruction parameters depending on what kind of data we want to reconstruct and to keep this changes in OCDB
  - possibility to change reconstruction parameters from the macro
- High clusterization threshold used to suppress noise digits
- SubtractPedestal flag was set to TRUE as the data are not baseline-subtracted
- See macro with example $ALICE_ROOT/PHOS/macros/BeamTest2006/RawReconstruction.C

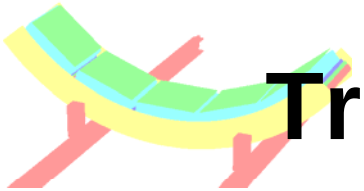


Rec.Param: use case

```
AliReconstruction rec ;
rec.SetOption("PHOS","OldRCUFormat");
rec.SetRunTracking("") ;
rec.SetRunVertexFinder(kFALSE) ;
rec.SetRunLocalReconstruction("PHOS") ;
rec.SetFillESD("PHOS") ;
rec.SetInput("rawData.root");

 //  Set non-default rec. parameters
AliPHOSRecoParam* recEmc = new
  AliPHOSRecoParamEmc();
recEmc->SetSubtractPedestals(kTRUE);
recEmc->SetMinE(0.05);
recEmc->SetClusteringThreshold(0.05);
AliPHOSReconstructor::SetRecoParamEmc(recEmc);

rec.Run();
```



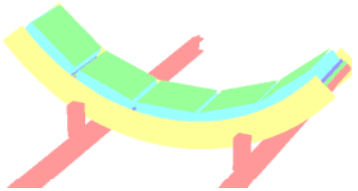
Track-PHOS matching and PID

AliPHOSPIDv1::MakePID()

- Calculates probabilities of PHOS cluster identification:
  - Flight time
  - Charged particle ID with CPV
  - Shower shape
- The product of all probabilities for all particle species identification is stored in AliESDCaloCluster

AliPHOSTracker::PropagateBack(esd)

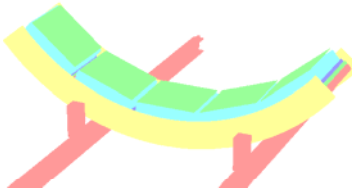
- Finds the best (closest) cluster matched by a track
- It should be implemented in PID for charged track identification
- *Thanks to Iouri Belikov for implementation*



ESD analysis: use case

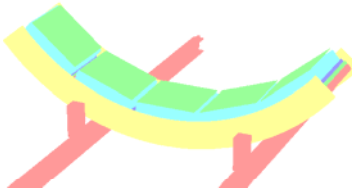
```
TFile f("AliESDs.root");
TTree *esdTree = (TTree*)f.Get("esdTree");
AliESDEvent *event = new AliESDEvent;
event->ReadFromTree(esdTree);
for(Int_t iEvent=0; iEvent<esdTree->GetEntries(); iEvent++){
   esdTree->GetEvent(iEvent);
   Int_t multClu = event->GetNumberOfCaloClusters();
   for (Int_t i=0; i<multClu; i++) {
      AliESDCaloCluster *clu = event->GetCaloCluster(i);
      Float_t xyz[3];
      clu->GetPosition(xyz);
      Float_t energy  = clu->E();
      Float_t *pid    = clu->GetPid();
      Int_t nDig      = clu->GetNumberOfDigits()
   }
}
```

See AliESDCaloCluster for more methods



Trigger

- Simulation of a trigger: AliPHOSTrigger.
- Recent development of PHOS TRU introduced a new trigger algorithm which should be implemented
- PHOS TRU will write its data to the raw data stream.
- The format of TRU raw data has been proposed by the PHOS/EMCAL trigger group (Hans Muller), it is waiting for approval by the offline team.
- Need for TRU raw data simulation and reader
- Trigger efficiency: simulation studies are needed



DQM and QA

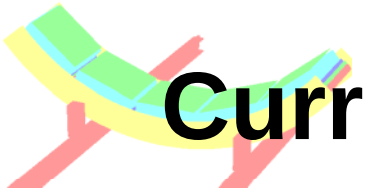
Data Quality Monitor:

- Now **HLT** DQM is available
- To define the list of histograms to store by online monitoring program
- To define the histograms to monitor by a run shift

Online Quality Assurance:

- To define the reference objects which the monitored one should be compared with
- Time evolution of basic run parameters should be monitored (average digit multiplicity, total deposited energy, etc.)
- Integration of the HLT DQM and offline QA framework is needed

DQM and QA strategy is to be defined and documented



Current status of PHOS data taking

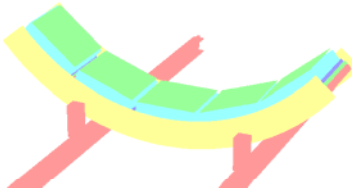
- PHOS module #2 is installed in the PHOS lab (bldg.167), cooled to –20°C, plugged and takes data almost continuously.
- Trigger:
  - Cosmic events with 110x70 scintillator telescope (detects cosmics coming from the top ±25°)
  - LED monitoring system
- Taken data is collected by DAQ to a local disk by 100-Gb chunks, after filling the chunk the data migrates to CASTOR and AliEn.
- So far (5.10.2007) total 4.3 Tbyte of data is stored in CASTOR
- The path to PHOS data in CASTOR:

  /castor/cern.ch/alice/phos/2007/
- The path to PHOS data in AliEn:

  /alice/data/2007/LHC07a_PHOS/
- More details about PHOS data taking see in FDR session on Friday.

Forthcoming PHOS commissioning

- Plans to be defined (what and when to measure)
  - Underground cosmic
  - LED
- Online DA: not fully tested, not fully functional
- Online DQM: ready in HLT, but still needs more improvements
- Do we expect half-year continuous PHOS operation before the LHC start?



Summary

- Generally speaking, PHOS is able to process data from raw to physics spectra
- The real data processing is not as smooth as we though in simulation
- Online monitoring and calibration still need to be validated in the real running conditions
- Hardware is still not perfect, it still requires manual intervention of the analizers
- Trigger data is still not implemented in analysis
- HLT data is still not linked to off-line
- DQM strategy is to be defined
- All the raw data analysis should be developed in a standard aliroot framework, macros and classes should be committed to CVS