Subject: Modifications in EMC classes
Posted by Dima Melnychuk on Wed, 28 Apr 2010 12:14:50 GMT
View Forum Message <> Reply to Message

Hi all,

I have committed some modifications to the EMC code. They required small changes in /pid/PidCorr/PndPidCorrelator.cxx, which I could not commit since I do not have write permission to pid directory. Required modifications are at the end of this post, so could somebody commit them.

Modifications are the following:

1. Till now PndEmcCluster contained the vector of PndEmcDigi from which it was created, not the indexes in TClonesArray. It resulted that those digis were stored twice, first time in a TClonesArray of PndEmcDigi and for the second time within this PndEmcCluster. With current modification the indexes are stored. And this modification required the next one.

2. PndEmcCluster was till now not the container class but provided via its methods calculation of some cluster properties such as energy, position, mass. Other properties such as different cluster moments were implemented as separate classes: PndEmcClusterEnergySums, PndEmcClusterMoments, PndEmcXClMoments. With current modification I moved most part of functionality to the new class PndEmcClusterProperties which calculate cluster energy, position and mass. For this the pointer to TClonesArray of PndEmcDigi should be passed to this class to access digis which belong to cluster with stored indexes. And cluster energy, position and three additional parameters used by PID (Zerike moments (2,0) and (5,3) and lateral energy distribution) are stored to PndEmcCluster with Set methods.
However the calculation of the corrected energy and error matrix is left at the moment within PndEmcCluster since only the information on cluster energy/position is needed for that and not the information from the digis.
Important aspect here is the calculation time. Since Zerike moments (2,0) and (5,3) and lateral energy distribution are calculated in the PndEmcMakeCluster and PndEmcMakeBump emc reconstruction macro need more time. However this time is gained during pid where these parameters were calculated and now the precalculated value can be accessed there.

3. The classes which provide information about cluster properties are not accessed from the PndEmcCluster but should be initialised separately

PndEmcClusterEnergySums esum(cluster, digi_array);
PndEmcClusterMoments moments(cluster, digi_array);
PndEmcXClMoments xmoments(cluster, digi_array);

Example with PndEmcClusterEnergySums is in /macro/emc/dedicated/reco_analys.C

4. There are some modification in bump splitting, which does not affect algorithm but only organisation of the classes. Local maxima finding and splitting are implemented as subtasks now.

5. Till now the shared digis produced in bump splitting were not stored. PndEmcSharedDigi is PndEmcDigi plus the weight with which digi belong to the given PndEmcBump. On the one hand if all the PndEmcClusters has only one local maxima and not splitted they are identical with PndEmcBumps and TClonesArray of PndEmcSharedDigi is identical with TClonesArray of

PndEmcDigi. But TClonesArray of PndEmcSharedDigi would be needed if the final object from emc reconstruction are PndEmcBumps and to calculate their properties the access to PndEmcSharedDigis will be needed.

6. Since I started these modification before Tobias provided MCTruth matching for EMC I tried to merge that part of code with my modifications and I hope it will work correctly.

Any comments are welcome.

Best regards,
Dima

svn diff PndPidCorrelator.cxx

```
Index: PndPidCorrelator.cxx
===================================================================
--- PndPidCorrelator.cxx        (revision 8485)
+++ PndPidCorrelator.cxx        (working copy)
@@ -407,7 +407,7 @@
       pidCand->SetEmcRawEnergy(bump->energy());
       pidCand->SetEmcCalEnergy(bump->GetEnergyCorrected());
       pidCand->SetEmcIndex(i);
-      pidCand->SetEmcModule(((PndEmcDigi*)bump->Maxima())->GetModule());
+       pidCand->SetEmcModule(bump->GetModule());
       pidCand->SetEmcNumberOfCrystals(bump->NumberOfDigis());

       pidCand->SetLink(FairLink((fDetectorType)emcType, i));
@@ -530,7 +530,7 @@
     }

     //if (emcHit->energy() < fCorrPar->GetEmc12Thr()) continue;
-    Int_t emcModule = ((PndEmcDigi*)emcHit->Maxima())->GetModule();
+        Int_t emcModule = emcHit->GetModule();
     if (emcModule>4) continue;

     emcPos = emcHit->where();
@@ -553,17 +553,16 @@

     Float_t dist = (emcPos-vertex).Mag2();
     if ( emcQuality > dist ){
-      const PndEmcXClMoments& clsZmom = emcHit->Xmoments();
       emcIndex = ee;
       emcQuality = dist;
       emcEloss = emcHit->energy();
       emcElossCorr = emcHit->GetEnergyCorrected();
       emcModuleCorr = emcModule;
       emcNCrystals = emcHit->NumberOfDigis();
-      Z20 = clsZmom.AbsZernikeMoment(2, 0, 15);// Z_{n = 2}^{m = 0}
-      Z53 = clsZmom.AbsZernikeMoment(5, 3, 15);// Z_{n = 5}^{m = 3}
-      secLatM = clsZmom.Lat();
```

```
-   }
+           Z20 = emcHit->Z20();// Z_{n = 2}^{m = 0}
+           Z53 = emcHit->Z53();// Z_{n = 5}^{m = 3}
+           secLatM = emcHit->LatMom();
+       }

    if (fDebugMode){
      Float_t ntuple[] = {vertex.), vertex.Y(), vertex.Z(), vertex.Phi(),
```