Subject: Bug in PndEmcHitProducer
Posted by Vanniarajan Suyam Jothi on Wed, 14 Apr 2010 09:00:41 GMT
View Forum Message <> Reply to Message

Hi,
  There was a probable problem in the PndEmcHitProducer observed.
The Energy deposited in the crystal from the PndEmcHit is fractionaly more than the energy sum from the PndEmcPoint
in the crystal.
  Mohammad Babai has found that there is a problem in the
following part of the PndEmcHitProducer::Exec() method.
the data entries of two maps fTrackEnergy and fTrackTime
are accessed un-initialised. Now He has fixed the
PndEmcHitProducer.

```
 for (Int_t iPoint=0; iPoint<nPoints; iPoint++)
  {
    point = (PndEmcPoint*) fPointArray->At(iPoint);
    fTrackEnergy[point->GetDetectorID()] += point->GetEnergyLoss();
        point_time=point ->GetTime();
        if (point_time<fTrackTime[point->GetDetectorID()]) fTrackTime[point->GetDetectorID()]
=point_time;

        // Check and save MC truth information
        // Eloss==0 tracks are only stored in point, if track is entering detector from outside
        // and thats what we are interested in...
        if(point->GetEnergyLoss()==0){
              fTrackMcTruth[point->GetDetectorID()].push_back(point->GetTrackID());
// cout << "ELoss==0 : ID " <<
point->GetTrackID()<<","<<point->GetDetectorID()<<","<<point->GetXPad()
<<","<<point->GetYPad()<<endl;
        }
  }
```

Regards,
Vanni

Subject: Re: Bug in PndEmcHitProducer
Posted by M.Babai on Wed, 14 Apr 2010 09:40:14 GMT
View Forum Message <> Reply to Message

Hi!,

This problem is now solved and the corrected code is available in trunk.
In the original code the following was declared and used without proper initialization:

```
map<Int_t, Float_t> fTrackEnergy;
map<Int_t, Float_t> fTrackTime;  //time of first point
map<Int_t, std::vector <Int_t> > fTrackMcTruth;  //McTruth
fTrackEnergy.clear();
fTrackTime.clear();
fTrackMcTruth.clear();
.........
.........
point  = (PndEmcPoint*) fPointArray->At(iPoint);
fTrackEnergy[point->GetDetectorID()] += point->GetEnergyLoss();
point_time=point ->GetTime();
if (point_time < fTrackTime[point->GetDetectorID()])
    fTrackTime[point->GetDetectorID()] = point_time;
```

In the lines above we can see a comparison and a "+=" operation on not initialized member of the map which has(might have) an undefined state.
Another point, which is (in this case) a matter of taste and beauty, is the declaration of:
PndEmcPoint* point = NULL;
map<Int_t, Float_t>::const_iterator p;
outside the loop. In this case they are not leading into wrong computations but potentially they can, as their values are changed inside the loop.

Greets,
/M

Subject: Re: Bug in PndEmcHitProducer
Posted by Elwin Dijck on Wed, 14 Apr 2010 18:01:23 GMT
View Forum Message <> Reply to Message

Hi all,

I don't think there was any problem with undefined values: if you try to access an element in a std::map that is not in the map using the []-operator, that element will be added with its default value. The default 'constructor' float() is not undefined and will simply give 0.0, so the usage of fTrackEnergy with += will work properly without explicit initialization.

However it is indeed true that fTrackTime needs initialization here, in the old code the result would always be the default value of zero since point_time is never < 0.

Regards,
Elwin

Subject: Re: Bug in PndEmcHitProducer
Posted by Johan Messchendorp on Wed, 14 Apr 2010 20:05:43 GMT
View Forum Message <> Reply to Message

Hi,

I tend to agree with Elwin. I did some tests with a simple program using a map simulating the code and indeed a first time call of the []-operator gives on all tested platforms nicely gives a zero. So, I also think that this cannot be a problem. The time issue is more nasty, but then again, if you don't use the time information in the analysis, it should not be a problem. So, honestly speaking, I cannot understand why the energy deposited using the PndEmcHitProducer should differ from looping over the points manually. The problem must be hidden somewhere else.

Mohammad A. pointed out to me another potential problem in the PndEmcHitProducer code related to the filling of the container of the hits:

Quote:// -- -
```
// -----   Private method AddDigi   --------------------------------------------
PndEmcHit* PndEmcHitProducer::AddHit(Int_t trackID,Int_t detID, Float_t energy,
                                     Float_t time, std::vector <Int_t> &mctruth)
{
  // It fills the PndEmcHit category

  //cout << "PndEmcHitProducer: track " << trackID << " evt " << eventID
  //<< " sec " << sec << " plane " << pla << " strip " << strip << "box
  //" << box << " tube " << tub << endl;
  TClonesArray& clref = *fDigiArray;
  Int_t size = clref.GetEntriesFast();
  return new(clref[size]) PndEmcHit(trackID, detID, energy, time, emcX[detID],
                                    emcY[detID], emcZ[detID], mctruth);
}
// ----
```

Note that the TClonesArray is filled with an object which can have a variable vector size: mctruth! I don't know the details of TClonesArrays, but can one fill it with a dynamic array containing a std::vector <Int_t> ???? I guess that this mctruth propagation is also obsolete with the new code of Tobias. So, maybe we should remove it anyway...

Best wishes,

Johan.