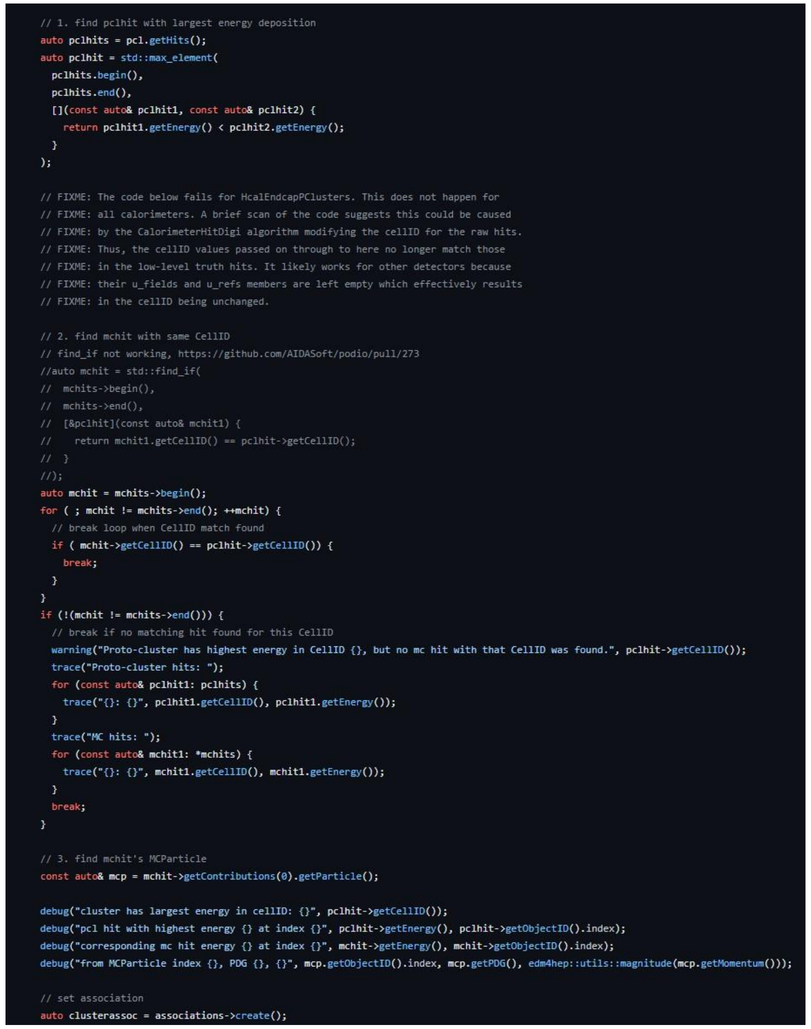
ePIC LFHCAL Meeting | Truth-Cluster Assoc.s



```cpp
// 1. find pclhit with largest energy deposition
auto pclhits = pcl.getHits();
auto pclhit = std::max_element(
  pclhits.begin(),
  pclhits.end(),
  [](const auto& pclhit1, const auto& pclhit2) {
    return pclhit1.getEnergy() < pclhit2.getEnergy();
  }
);

// FIXME: The code below fails for HcalEndcapPClusters. This does not happen for
// FIXME: all calorimeters. A brief scan of the code suggests this could be caused
// FIXME: by the CalorimeterHitDigi algorithm modifying the cellID for the raw hits.
// FIXME: Thus, the cellID values passed on through to here no longer match those
// FIXME: in the low-level truth hits. It likely works for other detectors because
// FIXME: their u_fields and u_refs members are left empty which effectively results
// FIXME: in the cellID being unchanged.

// 2. find mchit with same CellID
// find_if not working, https://github.com/AIDASoft/podio/pull/273
//auto mchit = std::find_if(
//  mchits->begin(),
//  mchits->end(),
//  [&pclhit](const auto& mchit1) {
//    return mchit1.getCellID() == pclhit->getCellID();
//  }
//);
auto mchit = mchits->begin();
for ( ; mchit != mchits->end(); ++mchit) {
  // break loop when CellID match found
  if ( mchit->getCellID() == pclhit->getCellID()) {
    break;
  }
}
if (!(mchit != mchits->end())) {
  // break if no matching hit found for this CellID
  warning("Proto-cluster has highest energy in CellID {}, but no mc hit with that CellID was found.", pclhit->getCellID());
  trace("Proto-cluster hits: ");
  for (const auto& pclhit1: pclhits) {
    trace("{}: {}", pclhit1.getCellID(), pclhit1.getEnergy());
  }
  trace("MC hits: ");
  for (const auto& mchit1: *mchits) {
    trace("{}: {}", mchit1.getCellID(), mchit1.getEnergy());
  }
  break;
}

// 3. find mchit's MCParticle
const auto& mcp = mchit->getContributions(0).getParticle();

debug("cluster has largest energy in cellID: {}", pclhit->getCellID());
debug("pcl hit with highest energy {} at index {}", pclhit->getEnergy(), pclhit->getObjectID().index);
debug("corresponding mc hit energy {} at index {}", mchit->getEnergy(), mchit->getObjectID().index);
debug("from MCParticle index {}, PDG {}, {}", mcp.getObjectID().index, mcp.getPDG(), edm4hep::utils::magnitude(mcp.getMomentum()));

// set association
auto clusterassoc = associations->create();
```

- Current logic in EICrecon:
  1) Identify **highest energy hit** (i.e. cell) in cluster
  2) Grab **1st contributing particle** of corresponding imulated hit, **regardless of energy, origin, status, species, etc.**
  3) Assign that contributor as the associated particle of the cluster

- Draft PR opened for updated algorithm
  - [eic/EICrecon#1382]
  - A few to-do's:
    - Improve old method (e.g. use highest-energy contributor rather than just the 1st)
    - Add old method as option for algorithm
    - Use TGeo volumes to check if vertex is in calorimeter (currently using named constant)
  - Aiming to be done with these by sometime next week