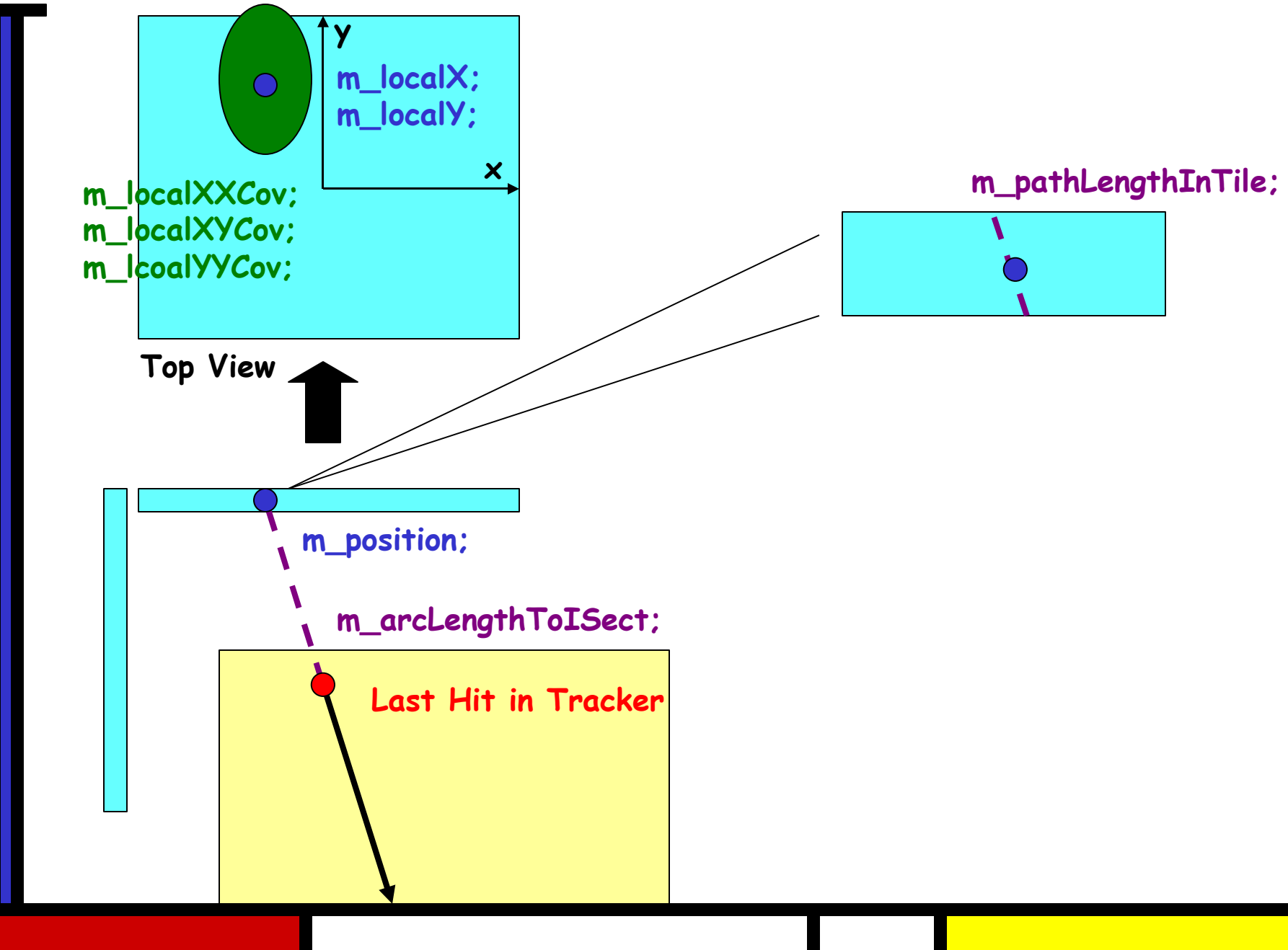
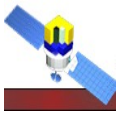
ACD Hit calibration

- Currently the energy deposition information in the ACD is only in terms of raw ADC counts
  - 0-4095 counts, low & hi range
- Ideally this should be in terms of MIPs
- To convert we need for each channel:
  - Pedestal values & gains for both ranges,
    - Low range pedestals come from random triggers
    - Low range gains come from MIP calibration
    - Hi range gains come from CNO/ charge injection
    - Hi range pedestals could come from charge injection
      - Need to investigate hi-range calibration more
  - For the time being we wiil focus on low range calibration, could treat hi-range as saturated values for now

Track Extrapolation to ACD



Track Extrapolation to ACD



- Currently, each track is extrapolated to ACD some information is stored for each tile the track crosses

```
Event::AcdTkrIntersection {

  AcdId m_tileId;          // which tile was hit
  int   m_trkId;           // which track did the hitting

  Point  m_location;       // 3D global postion of Tkr ACD element i-sect

  double m_localX;         // Position of hit in ACD element plane
  double m_localY;

  double m_localXXCov;     // Error ellipse of track project onto plane
  double m_localXYCov;
  double m_localYYCov;

  double m_arcLengthToIntersection; // Distance from last track hit to i-sect

  double m_pathLengthInTile;  // Distance track travels in tile
}
```

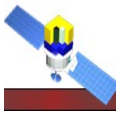
Caveat: Near misses are NOT factored into these calculations

Calibrated ACD Hits

- Just want to keep track of the pulses in terms of MIPs, rather than ADC counts

```
Event::AcdHit {

  AcdId m_tileId;            // which tile was hit

  unsigned m_rawData;        // Digi level data, including Veto flags
  bool pha(Pmt AorB);        // Access to above information
  bool hasHit(Pmt AorB);
  bool hasVeto(Pmt AorB);

  float m_mipsPmtA;          // calibrated values
  float m_mipsPmtB;
  float mips();              // average of PMT values
}
```

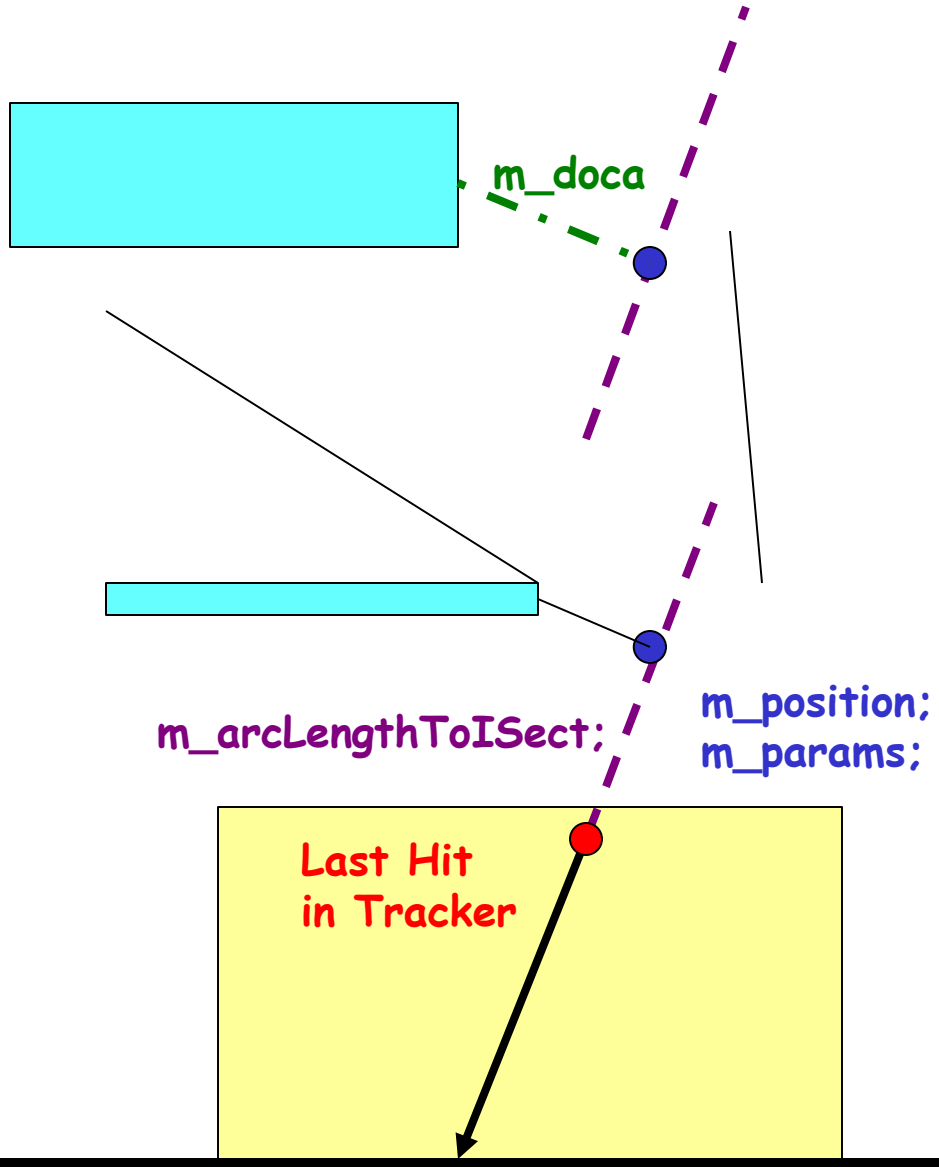
Associations Between Tracks and ACD hits

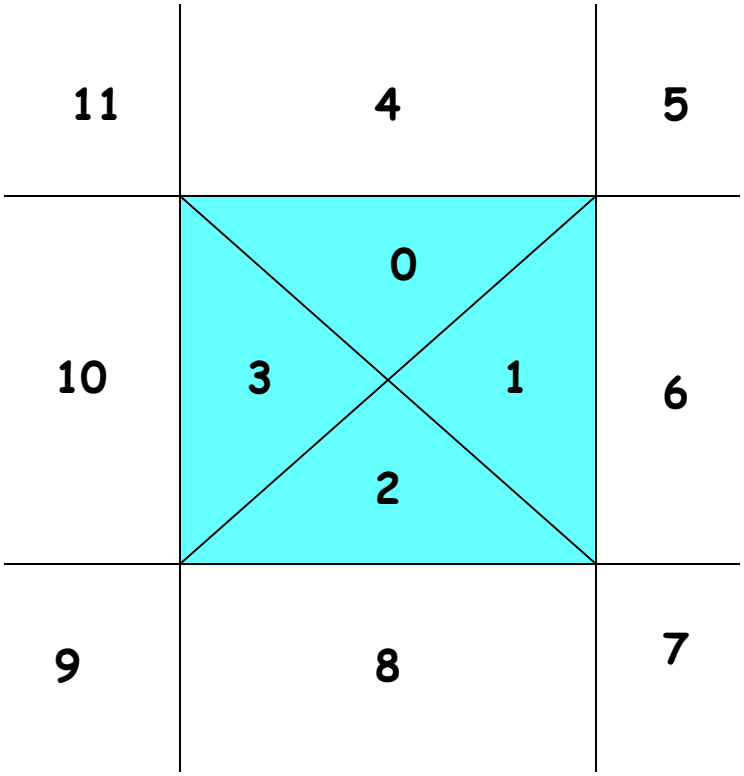


- We want to keep track of which track come close to which hit ACD element
- For each track with an arbitrary distance of a hit ACD element we can store

```
Event::AcdTkrPocaData {

  AcdId m_tileId;          // which tile was hit
  int     m_trkId;         // which track did the hitting

  Point   m_location;      // 3D global postion of POCA

  TkrTrackParams m_params;  // track params at the POCA
  double m_arcLengthToISect;  //

  enum DocaStatusCode { see next page; } m_docaStatus;
  double m_doca;
  double m_docaErr;
}
```

Associations Between Tracks and ACD hits





TopView

POCA status code