Tracking / Digitization Update

Matt Graham and Richard Partridge

June 2, 2009

Readout Chip

- Simulates functionality of the readout chip
  - Charge to ADC count conversion
  - Adds noise to channels with hits as well as adding random noise hits
  - Encodes / decodes raw data format
- KPix readout chip developed by Tim
  - Closely modeled on KPix
  - Fairly complex code due to detailed modeling of KPix
- Occasional hit position problems traced back to KPix code
  - Noise fluctuations producing negative charge were turned into large charges
  - Fixed (we think) by not letting charge be negative
- Based on this experience, we decided to write a simpler "GenericReadoutChip" class
  - Provides basic functionality required for a readout chip
  - Added several bits of new functionality
    - Introduced hit threshold & neighbor threshold (low charge hits were over-estimating raw hit occupancy)
    - Simple linear noise formula Sig(Noise) = NoiseOffset + NoiseSlope * Capacitance
    - Made thresholds, noise parameters settable
    - Protects against underflow/overflow conditions in 16-bit raw data format

Clustering Code

- Clustering code would occasionally go into an infinite loop that would continue until all heap space was exhausted
  - Spent some time trying to find problem, but wasn’t having much luck
  - Problem most likely was in the details of the list handling
- In looking at old clustering algorithm, it appeared that a slightly different algorithm would be more efficient
  - Old algorithm appeared to have non-linear scaling with number of hits
  - Using an additional HashMap, it looked like linear scaling could be achieved
- Decided to implement new NearestNeighbor clustering algorithm class
  - Use HashMaps to achieve linear scaling with hit occupancy
  - Created new ClusteringAlgorithm interface to allow different clustering algorithms to be used by the hit maker
  - Made cluster seed and neighbor thresholds settable

Pixel Digitization

- Tim had largely coded a version of his digitization code for pixels
- A few pieces were missing
  - Mapping a 2D Gaussian charge distribution onto pixels
  - Pixel hit making / clustering software
  - Making everything work
- Decided to finish this work up
  - Single algorithm for pixel and strip digitization, clustering, and hit making
  - Potentially adaptable to ATLAS “short strip” (or long pixel) detectors
- RP developed charge distribution code
  - Created BivariateDistribution class to integrate 2D Gaussian onto an array of identical rectangular pixels based on algorithm by Genz
- RP also Coded a new Erf class to avoid in-accuracies / exceptions from Apache Erf class that was being used
  - Based on algorithm by Schonfelder
- Both algorithms should be quite fast / accurate
- Tim developed the hit making / clustering software
  - RP modified to use NearestNeighbor clustering aglorithm developed for strips

Tracking Changes

- Divided tracking volume into sectors to localize tracking searches
- Developed improved “quick check” algorithms to test seeds for consistency with pT and impact parameter constraints
- Improved algorithm for forming HelicalTrackCross hits in stereo layer
- Each of these changes has greatly improved performance in a high-occupancy environment
- Probably still need further work to deal with sLHC conditions
  - 2M raw hits / event!!

Future Projects

- Pull sensor electrode code out of GeomConverter so that a particular detector is not hard coded
  - Tim and Jeremy are working on this
  - Plan is to put detector-specific code in a driver
  - High priority – can’t check modified detectors into CVS
- Simulate fast shaping readout chips??
  - Currently assume all charge is collected on the sense electrodes and transferred to a readout chip
  - In ATLAS (for example), shaping time is comparable to longest charge collection time
- Allow other parameterization of silicon ChargeCarriers?