2017 Highlights of Hall A & C
Reconstruction Software Development

Ole Hansen

Jefferson Lab

Computing Roundtable
December 5, 2017
The Hall A & C Analyzer ("Podd"): Data Flow

1. **Reconstruction (Replay)**
   - Runs in ROOT interpreter (analyzer prompt)
   - Calls mostly Podd functions & classes
   - Scripts usually set up by experiment experts or advanced users
   - After setup, usually runs in mass replay on the farm

2. **Analysis**
   - Also runs in ROOT interpreter (analyzer prompt)
   - Calls mostly ROOT functions and classes (but may need Podd classes)
   - Done by everyone on the experiment
   - Calibration and final physics usually done here
Podd Reconstruction: Typical Steps

- Raw data **decoding**
- Calibration (gains, offsets)
- Tracking in one or more spectrometers
- Vertex reconstruction using (inverted) Transport matrices
- PID likelihood analysis
- Retrieval of beam parameters, EPICS fields, scalers & other control data
- Basic **physics** computations (kinematics)
- Basic corrections (energy loss, etc.)

Each item is only executed if requested (configured)
Example Analysis Script (Hall C)
Podd: Main Accomplishments 2017

- Consistent key/value **database** (configuration & conditions)
- **Output** system performance and efficiency improvements
- Better HRS spectrometer **VDC tracking** (noise resistance)
Podd: Database Example

Validity timestamp →
Scalar key/value pairs →
Array key/values (auto-sized) →

Retrieve parameter blocks in single call:

```c
DBRequest request[] = {
    {"detmap", &detmap, kIntV },
    {"nwires", &nwires, kIntV },
    {"wire.start", &wire.start, kIntV },
    {"wire.spacing", &wire.spacing, kIntV },
    {"wire.angle", &wire.angle, kIntV },
    {"wire.badlist", &wire.badlist, kIntV },
    {"driftvel", &driftvel, kIntV },
    {"cdc.min", &cdc.min, kIntV },
    {"cdc.max", &cdc.max, kIntV },
    {"cdc.res", &cdc.res, kIntV },
    {"tdc.offsets", &tdc_offsets, kFloatV },
//...}

err = LoadDB( file, date, request, fPrefix );
```
Podd: User-Configurable N-tuple Output

Module header file:

```c
protected:
  TClonesArray* fWires; // Wires
  TClonesArray* fHits; // Fired wires
  TClonesArray* fClusters; // Clusters

Int_t MWHits; // Total number of hits (including multihits)
Int_t MWiredHit; // Number of wires with one or more hits
```

Module DefineVariables() method:

```c
RVarDef vars[] = {
  {"hit", "Number of hits"},
  {"wire", "Active wire numbers"},
  {"wiretime", "Raw TDC values of wires"},
  {"time", "TDC values of active wires"},
  {"dist", "Drift distances"},
  {"distdist", "Dist. drift uncertainty"},
  {"rdist", "Dist. from track"},
  {"trdist", "Dist. from local track"},
  {"tknum", "Track number (0=unused)"},
  {"aenum", "Cluster number (2=unused)"},
  {"clus", "Number of clusters"},
  {"clusize", "Cluster size"},
  {"cluswire", "Cluster wire number"},
  {"clusxintercept", "Cluster intercepts (m)"},
  {"clusyintercept", "Cluster intercepts (m)"},
  {"cluszintercept", "Cluster intercepts (m)"},
  {"clusbestslope", "Cluster best slope"},
  {"clusbestlocal", "Cluster best local"},
};
```

Output definition file:

```c
# All RHCS VDC raw data (lots of information, only uncomment for debugging)
# block ${arm}.vdc.*
block ${arm}.vdc.ul.*
```

ROOT file TTree:

```c
*Dr 17 :L.vdc.ul.wire : data[Ndata.L.vdc.ul.wire]/D
*Entries : 265460 : Total Size  11978559 bytes File Size  2295160 *
*Baskets : 81 : Basket Size  2683984 bytes Compression  5.21 *

*Dr 18 :L.vdc.ul.nclus : L.vdc.ul.nclus/D
*Entries : 265460 : Total Size  7156991 bytes File Size  45865 *
*Baskets : 131 : Basket Size  64512 bytes Compression  47.36 *

*Dr 19 :L.vdc.ul.nhit : L.vdc.ul.nhit/D
*Entries : 265460 : Total Size  2136721 bytes File Size  184378 *
*Baskets : 131 : Basket Size  64512 bytes Compression  11.57 *
```

Ole Hansen (Jefferson Lab) 2017 Highlights Halls A & C Computing Roundtable 12/17 7 / 15
Podd: Output Improvements

Original:
- Only D data type
- Multiple, redundant array size variables with parallel arrays

Improved:
- Full range of data types
- Automatic detection of parallel arrays, only one size variable (with limitations)
- Automatic basket size adjustment (overriding ROOT default)
Podd: VDC Algorithm Improvements

Version 1.5.38
- Disallow UV ambiguities (configurable)
- UV fiducial cut
- Proper lower-upper matching cut
- Disallow cluster sharing

→ Guarantees clean single track at expense of slightly lower tracking efficiency

Version 1.6
- Cluster shape analysis
- Overlapping cluster splitting
- 3-parameter cluster fit
- Cluster $t_0$ cut
- UV fiducial cut
- Proper lower-upper matching cut
- Disallow cluster sharing
- Old VDC code for reference

→ Allows multi-tracks, improves tracking efficiency, high-rate capable
Improved algorithm:

- Fewer reconstruction errors: smaller tails, larger signal
- No need for explicit cuts on “clean” events
- Should be particularly helpful with high-rate/high-noise data (to be tested)
Hall C: 2017 Accomplishments

Hall C’s replay software (“hcana”) is a direct extension of Podd

- Got software and replay **ready** for, and demonstrated performance in Spring KPP run
- Many small **bugfixes** and usability improvements over past few months. Many of these have benefited Hall A as well.
- Very well organized **replay setup** (databases, configuration parameters, standard replay scripts, output & cut definitions, etc.)
Hall C Replay Overview (slides courtesy of E. Pooser)

CONFIGURATION FILES
- RUN
- PARAM
- STD
- KINEM
- HMS
- SHMS
- GEN
- TRIG
- PARAM
- DBASE
- HMS
- SHMS
- DETEC
- TRIG
- HMS
- SHMS
- DEF FILES
- MAPS

ANALYSIS FILES
- HODO
- DC+CAL
- HMS
- REPLAY SCRIPTS
- SHMS
- CER+DC
- AERO
- ROOT TREE
- DEF-FILE HISTOS
- ONLINE GUI

RESULTS
Hall C Replay Repository
Summary

- Hall A/C software has seen significant improvements in 2017
- Several long-standing annoyances with the Hall A wire chamber track reconstruction have been corrected
- Hall C’s replay setup has been very well organized and is ready for the upcoming Dec–Mar commissioning/physics runs
- Because of the code sharing between Hall A and C, general code improvements immediately benefit both Halls