Overview & Update of the Hall-C Analyzer

Eric Pooser

Jefferson Lab

06/25/2018
Hall-C Analyzer Overview

- Hall-C ROOT Analysis framework (HCANA) is written in C++ and is an extension of the Hall A analyzer “podd”
- Based on previous Fortran Analyzer ENGINE infrastructure
- Hall C ROOT Analyzer Wiki
- Maintained on GitHub
- Users fork off of “develop” branch
- Contributions are made in local/remote personal repository of HCANA
- Changes are pushed back onto the development branch which (may) get merged into the development branch
- Detailed "How-To" wiki: instructions how to “Git” started with HCANA and GitHub

06/25/2018

Eric Pooser

Hall A/C Data Analysis Workshop
Steps to Install HCANA

1. Download and install **ROOT** (> 5.32)
2. Setup ROOT environment
   - `source /path/to/rootbuild/bin/thisroot.(c)sh`
3. Fork **hcana repository** (if you have not already done so)
4. Clone personal remote repository on local machine
   - `git clone https://github.com/username/hcana`
5. `cd` into “hcana” directory and setup the environment
   - `source setup.(c)sh`
6. Obtain the podd submodule which hcana points too
   - `git submodule init`
   - `git submodule update`
7. Create new branch and switch to it
   - `git checkout -b branch-name`
8. Build HCANA
   - `scons -j4`

"How-To" wiki
ROOT Analyzer/Compiling Wiki
ROOT Analyzer/Running Wiki
Hall-C Analyzer Overview

• **Doxygen page** documents HCANA source code

Hall C ROOT/C++ Analyzer (hcana)

THcDC Class Reference

This class analyzes a package of horizontal drift chambers. It uses the first letter of the apparatus name as a prefix to parameter names. The parameters, read in the Setup method, determine the number of chambers and the number of parameters per plane.

**Author**

S. A. Wood, based on Fortran ENGINE

**Public Member Functions**

```c
virtual ~THcDC ()
virtual int Decode (const THaEvData &)
virtual EStatus Init (const TDateTime &run_time)
virtual int CoarseTrack (TClonesArray &tracks)
virtual int FineTrack (TClonesArray &tracks)
virtual int ApplyCorrections (void)
    int GetNWires (int plane) const
    int GetNChamber (int plane) const
    int GetWireOrder (int plane) const
    Double_t GetPitch (int plane) const
    Double_t GetCentralWire (int plane) const
    int GetTdcWinMin (int plane) const
```
Work Flow of HCANA

- HCANA utilizes the Hall-A podd decoder to unpack raw EVIO data into THaEvData objects
  - Each readout module (F250, 1190, …) has its own decoder class \( \rightarrow \) podd/hana_decode
- THcHitList utilizes the detector maps to associate the ROC, slot, & channel number with a specific detector
  - THcHitList::DecodeToHitList(&THaEvData)
- THcDetectorMap builds an array (fTable) with one structure element per readout channel
  - THcDetectorMap::Load(const char *fName)
- Each element holds the roc, slot, channel number and module type for a given channel
- In addition, the element holds the id number of the detector, the plane, the counter (or wire number), and the signal number (ADC/TDC, +/-)
Work Flow of HCANA

- At initialization each detector class has its detector map populated with the list of readout channels belonging to it
  - `gHcDetectorMap->FillMap(fDetMap, idstring)`
- In addition, a hit list is created to hold the raw hits for each respective detector
  - `THcHitList::InitHitList(fDetMap, rawhitclassname, maxhits)`
- Detector classes then decode the THcHitList data into raw hit data associated with a specific detector’s readout channel
  - `THcDetector::Decode(&THaEvData)`
- Raw hit information is stored as TClonesArray objects which are accessible via THcAnalyzer
  - `adcPulseAmpRaw, adcPulseAmp, ...`
Work Flow of HCANA

- Raw hit data is further processed into higher level hit data objects *via* fiducial ADC & TDC cuts
  - `THcDetector::CoarseProcess(&TClonesArray)`
  - `goodAdcPed, goodAdcPulseInt, ...`
- Hit data can then be further processed into higher level physics data
  - `THcDetector::FineProcess(&TClonesArray)`
  - Fiducial tracking cuts *e.g.* $\chi^2/\text{ndf}$, $\beta$, $E/p$
  - Track matching, efficiencies, ...
- Raw, hit, & physics data is then processed by `THcAnalyzer` which makes the data available *via* ROOT histograms and TTrees
Recent Updates to HCANA: **e9d8699**

- **BCM Module**: THcBCMCurrent
  - Sanghwa's Slides
  - Reads average BCM current values from scaler parameter file
  - Writes values into bcm#.AvgCurrent event by event
  - Compares the current readout values with the threshold and sets event flags (0, 1)
  - The threshold current can be configured in two ways
    - SetCurrentCut()
    - gBCM_Current_threshold

[Graphs and tables showing data analysis results]

Eric Pooser

06/25/2018
Recent Updates to HCANA: 4cb3bfb

- Missing reference times are now printed in the analyzer summary
  - `End(THaRunBase *run)` added to each detector class
  - `THcHitList::MissReport(const char *name)`

Normal end of file ./raw/shms_all_02484.dat encountered
Missing Ref times: T.shms 0 4152
Missing Ref times: P.ngcer 0 3484
Missing Ref times: P.dc 0 0
Missing Ref times: P.hod 0 4166
Missing Ref times: P.hgcer 0 1519
Missing Ref times: P.aero 0 2161
Missing Ref times: P.cal 0 3021
THcScalerEvtHandler::End Analyzing 3 delayed scaler events
End of file
• Reference time selection is dependent on the first TDC/FADC hit in the window which is greater than the associated cut

  \[ X_{t(a)dcrefcut} \]

• “Good” FADC hits are determined via time difference cuts for every FADC readout channel

  \[ X_{adcTimeWindowMin(\text{Max})} \]
Recent Updates to HCANA

- Creation of **THcReactionPoint**
  - Calculates the z-vertex at the target in the transport coordinate system
  - The x & y vertex are defined via. `gbeam_x(y)` offsets and the raster position

- Modification to **THcDC** & **ThcDCTrack** now provides per plane residual calculations excluding the plane for which the residual is being calculated
  - `X.dc.residualExclPlane`
  - `vector <Double_t>`

\[ \sigma \approx 256 \mu m \]

\[ 10 \, \text{cm LD}_2 \]
Recent Updates to HCANA: e06d50f

- Bug fix in `THcHodoscope`

- `fFPTimeAll` is the average FPT as calculated by all scintillator times

- `fFPTimeAll` was calculated in `EstimatedFocalPlaneTime()`, where it was set to the `starttime` defined to be the average FPT calculated without using a DC track

- For tracks, a FPT was calculated for each track by utilizing the track information in order to improve the FPT calculation and was saved for each track

- `fFPTimeAll` was being filled for each track, and thus the last track

- `fFPTimeAll` was moved to `THcHodoscope::FineProcess` and is now set to the average FPT of the golden track

- Data calibrated with run 2540

- Data shown with the same kinematics is for run 2525

\[ \sigma \approx 250 \, \mu m \]
Recent Updates to HCANA: 8a26c04

- FADC’s modules found to have their trigger times (internal 250 MHz clock) slip randomly by 4 ns with firmware present during fall 2017 and spring 2018 run
- Correction applied in `THcHitList::DecodeToHitList()` via of comparing trigger time provided by the TI module to the trigger time in each FADC module
• Creation of coincidence time module **THcCoinTime**
• **Documentation**
• Facilitates the analysis of coincidence time data
• Derived from **THaPhysicsModule**
• The spectrometers central path length and coincidence time offset are parameters in hallc-replay
• Coincidence times for various particle species are supported
Hall-C Replay

- Framework designed to facilitate the reconstruction of events in the Hall-C spectrometers
- The Hall-C Replay Structure Wiki provides general information regarding the infrastructure
- The Hall-C replay code is maintained in the JeffersonLab/hallc_replay GitHub repository

- Interfaces with THcAnalyzer to process and obtain the data
  - Raw, hit, & tracking data
  - Physics data

<table>
<thead>
<tr>
<th>CALIBRATION</th>
<th>Modify readme.MD file (#149)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATFILES</td>
<td>New optics reconstruction matrix</td>
</tr>
<tr>
<td>DBASE</td>
<td>Scaler replay and online GUI</td>
</tr>
<tr>
<td>DEF-files</td>
<td>changed SHMS DC binning to b</td>
</tr>
<tr>
<td>MAPS</td>
<td>Wiremap fix (#117)</td>
</tr>
<tr>
<td>PARAM</td>
<td>Shms dc calib (#144)</td>
</tr>
<tr>
<td>SCRIPTS</td>
<td>Hms calo def file (#130)</td>
</tr>
<tr>
<td>TEMPLATES</td>
<td>Modify TEMPLATES/pstackana.f</td>
</tr>
<tr>
<td>onlineGUI</td>
<td>Hms def file develop (#125)</td>
</tr>
</tbody>
</table>
Hall-C Replay Framework: DBASE

**Configuration Files**

- **RUN**
- **PARAM**
- **STD**
- **KINEM**

**DBASE**

```plaintext
#include "PARAM/HMS/AERO/haero.param"
#include "PARAM/HMS/CAL/hcal.pos"
#include "PARAM/HMS/CAL/hcal.param"
#include "PARAM/HMS/CER/hcer.param"
#include "PARAM/HMS/DC/hdc.param"
#include "PARAM/HMS/DC/hdc.pos"
#include "PARAM/HMS/DC/hdc_tracking.param"
#include "PARAM/HMS/DC/hdriftmap.param"
#include "PARAM/HMS/HODO/hhodo.pos"
#include "PARAM/HMS/HODO/hhodo.param"

; General HMS parameter files
; Note: shmsflags.param includes spectrometers
#include "PARAM/HMS/GEN/pcana.param"
#include "PARAM/HMS/GEN/pdebug.param"
#include "PARAM/HMS/GEN/shmsflags.param"
#include "PARAM/HMS/GEN/ptracking.param"
```

- \$gpbeam\$ = 6.4
- \$gtarg_num\$ = 1
- \$htheta_lab\$ = 15
- \$pttheta_lab\$ = 15
- \$hpcentral\$ = 3
- \$ppcentral\$ = 3
- \$hpartmass\$ = 0.00051099
- \$opartmass\$ = 0.00051099
Hall-C Replay Framework: PARAM

CONFIGURATION FILES

- RUN
- PARAM
- STD
- KINEM
- HMS
- SHMS
- GEN
- TRIG

DBASE

PARAM

PARAM

Configuration files for Hall-C Replay Framework include:

- Incl. detector specific parameter files:
  - "PARAM/HMS/AERO/haero.param"
  - "PARAM/HMS/CAL/hcal.pos"
  - "PARAM/HMS/CAL/hcal.param"
  - "PARAM/HMS/CER/hcer.param"
  - "PARAM/HMS/DC/hdc.param"
  - "PARAM/HMS/DC/hdc.pos"
  - "PARAM/HMS/DC/hdc_tracking.param"
  - "PARAM/HMS/DC/hdriftmap.param"
  - "PARAM/HMS/HODO/hhodo.pos"
  - "PARAM/HMS/HODO/hhodo.param"

- General SHMS parameter files:
  - Note: shmsflags.param includes spectrometers:
  - "PARAM/SHAM/SHEMS/GEN/pcaena.param"
  - "PARAM/SHAM/SHEMS/GEN/pdebug.param"
  - "PARAM/SHAM/SHEMS/GEN/shmsflags.param"
  - "PARAM/SHAM/SHEMS/GEN/ptracking.param"

- Number of heavy gas Cherenkov PMT's:
  - phgcer_tot_pmts = 4

- Garth H. gain calibration from run 486, March 9 2017:
  - phgcer_adc_to_npe = 1/436., 1/393., 1/364., 1/372.
Hall-C Replay Framework: MAPS

Configuration Files

- RUN
- PARAM
- STD
- KINEM
- HMS
- SHMS
- GEN
- TRIG

DBASE

PARAM

SHMS

HMS

CRATE

MAPS

Sample Configuration Files:

```c
; HMSG detector specific parameter files
#include "PARAM/HMS/AERO/haero.param"
#include "PARAM/HMS/CAL/hcal_pos"
#include "PARAM/HMS/CAL/hcal.param"
#include "PARAM/HMS/CER/hcer.param"
#include "PARAM/HMS/DC/hdc.param"
#include "PARAM/HMS/DC/hdc_pos"
#include "PARAM/HMS/DC/hdc_tracking.param"
#include "PARAM/HMS/DC/hdrlftmap.param"
#include "PARAM/HMS/HODO/hhodo.pos"
#include "PARAM/HMS/HODO/hhodo.param"

; General SHMS parameter files
; Note: shmsflags.param includes spectrometers
#include "PARAM/SHMS/GEN/pcana.param"
#include "PARAM/SHMS/GEN/pdebug.param"
#include "PARAM/SHMS/GEN/shmsflags.param"
#include "PARAM/SHMS/GEN/strip_tracking.param"

; Number of heavy gas Cherenkov PMTs's
phgcer_tot_pmts = 4

; Garth H. gain calibration from run 486, May 9 2017
phgcer_adc_tmppe = 0.436, 0.393, 0.364, 0.372
```

```
 06/25/2018 Hall A/C Data Analysis Workshop
```

Eric Pooser
Hall-C Replay: DEF-files

- **THaOutput Analysis Output** (DEF-files)
- Interfaces with THcAnalyzer to output user defined histograms and ROOT TTrees
- Variables are created in detector classes in the DefineVariables() method and can be written to the TTree output and histogrammed via DEF-files
  - `P.hodo.goodAdcPulseAmp`
- Block variables write multiple variables to the TTree with a single call
  - `block P.gtr.*`
- Histograms can be defined for specific variables with and without cuts (1, 2, & 3D supported)
  - `TH1F h1Name 'Title; X-Title; Y-Title' var nBins xLow xHigh`
  - `TH1F h1Name 'Title; X-Title; Y-Title' var nBins xLow xHigh cut1&cut2||cut3`
- DEF-files defining cuts create global cut objects for histograms and interactive analysis
  - `time_cut1 P.dc.1u1.time>0&&P.dc.1u1.time<250`
Hall-C Replay Framework: Analysis Files

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

**ANALYSIS FILES**
- HMS
- DETEC
- TRIG
- SHMS
- MAPS

- REPLAY SCRIPTS
  - HODO
  - DC+CAL

- Files: PARAM, SHMS, HMS, GEN, TRIG, DBASE, DETEC, TRIG, MAPS

Jefferson Lab
Hall-C Replay Scripts: Report Files

```c
void replay_workshop_example(Int_t RunNumber=0, Int_t MaxEvent=0) {

    // Get RunNumber and MaxEvent if not provided.
    if(RunNumber == 0) {
        cout << "Enter a Run Number (-1 to exit): ";
        cin >> RunNumber;
        if( RunNumber==0 ) return;
    }

    if(MaxEvent == 0) {
        cout << "Number of Events to analyze: ";
        cin >> MaxEvent;
        if(MaxEvent == 0) {
            cerr << "Invalid entry\n";
            exit;
        }
    }

    // Create file name patterns.
    const char* RunFileNamePattern = "raw/shms_all_@05d.dat";
    const char* ROOTFileNamePattern = "ROOTfiles/shms_replay_4k4k_.root";
    // Add variables to global list.
    gHcParms->Define("gen_run_number", "Run Number", RunNumber);
    gHcParms->AddString("g_c_rp_database_filename", "DBASE/standard.database");
    // Load variables from files to global list.
    gHcParms->Load(gHcParms->GetString("g_c_rp_database_filename", RunNumber));
    // Decode par file name and decode map filename should now be defined.
    gHcParms->Load(gHcParms->GetString("g_c_rp_database_filename", RunNumber));
    gHcParms->Load(gHcParms->GetString("g_c_rp_1Par_filename", RunNumber));
    // Load parameters for SHMS trigger configuration
    gHcParms->Load("PARAM/TRIG/shmsparameters.param");

    // Create the hall C style detector map
    gHcDetectorMap = new THDetectorMap();
    gHcDetectorMap->Load("MAPS/SHMS/DETEC/shms_stack.map");

    // Add trigger apparatus
    THApparatus* TRG = new THApparatus("T", "TRG");
    gHcApps->Add(TRG);
    // Add trigger detector to trigger apparatus
    THTrigDet* shms = new THTrigDet("shms", "SHMS Trigger Information");
    TRG->AddDetector(shms);
    // Set up the equipment to be analyzed.
    THApparatus* SHMS = new THHallSpectrometer("P", "SHMS");
    gHcApps->Add(SHMS);
    // Add drift chambers to SHMS apparatus
    THDC* dc = new THDC("dc", "Drift Chambers");
    SHMS->AddDetector(dc);
    // Add hodoscope to SHMS apparatus
    THHodoscope* hod = new THHodoscope("hod", "Hodoscope");
    SHMS->AddDetector(hod);
    // Add heavy gas cherenkov to SHMS apparatus
    THCherenkov* hgcer = new THCherenkov("hgcer", "Heavy Gas Cherenkov");
    SHMS->AddDetector(hgcer);
    // Include golden track information
    THaGoldenTrack rtr = new THaGoldenTrack("gtr", "SHMS Golden Track", "P");
    gHcPhysics->Add(gtr);
}
```

```
// Add handler for prestart event 125.
THConfigEvtHandler* evi125 = new THConfigEvtHandler("HC", "Config Event type 125");
gHcEvtHandlers->Add(evi125);

// Set up the analyzer - we use the standard one,
// but this could be an experiment-specific one as well.
// The Analyzer controls the reading of the data, executes
// tests/cuts, loops over Acpparatus's and PhysicsModules,
// and executes the output routines.
THAnalyzer* analyzer = new THAnalyzer;

// A simple event class to output to the resulting tree.
// Creating your own descendant of THaEvent is one way of
// defining and controlling the output.
THaEvent* event = new THaEvent;

// Define the run(s) that we want to analyze.
// We just set up one, but this could be many.
char RunFileName[100];
sprintf(RunFileName, RunFileNamePattern, RunNumber);
THaRun* run = new THaRun(RunFileName);

// Eventually need to learn to skip over, or properly analyze
// the pedestal events
run->SetEventRange(1, MaxEvent); // Physics Event number, does not
run->SetNscan(1);
run->SetDataRequired(0x7);
run->Print();

// Define the analysis parameters
TString ROOTFileName = Form(ROOTFileNamePattern, RunNumber, MaxEvent); analyzer->SetCountMode(2); // 0 = counter is # of physics triggers
 analyzer->SetEvent(event);
 analyzer->SetCrateMapFileName("MAPS/db_crateemap.dat");
 analyzer->SetOutFile(ROOTFileName.Data());
 analyzer->SetDefFile("DEF-files/SHMS/GEN/workshop_example.def");
 analyzer->SetDefFile("DEF-files/SHMS/GEN/pstackana_cuts.def");

// File to record cuts accounting information
 analyzer->SetSummaryFile("summary_example.log");
 analyzer->Process(run);
 analyzer->CreateReportFile from template.
 analyzer->PrintReport();
 analyzer->PrintReport( "templates/chaina.template", "REPORT_OUTPUT/replay_shms_@05d_report", RunNumber )
```
Hall-C Replay: Results

CONFIGURATION FILES

RUN
PARAM
STD
KINEM
HMS
SHMS
GEN
TRIG

PARAM
DBASE
HMS
DETEC
TRIG

SHMS
DETEC
TRIG

ANALYSIS FILES

DEF FILES

HODO
DC+CAL

HMS

ROOT TREE
DEF-FILE
HISTOS
ONLINE
GUI

RESULTS

REPLAY SCRIPTS

SHMS

CER+DC
AERO

MAPS
Hall-C Replay: ROOT TTree
Hall-C Replay: DEF-Files Histograms

SHMS 1X+ Raw TDC vs. Paddle Number

Filter: All Files (*)

Command:
Command (local):
Recent Updates to Hall-C Replay

• Hall-C replay reconfigured to specifically support online analysis: **#403**

• Run specific calibrations and parameter configuration removed so that the replay is specific to experiment which is currently on the floor

• The previous run specific configuration is preserved with the **tagged release 1.0**

• Submodules configured for specific run groups so they have complete control of the online & offline replays

---

Eric Pooser

06/25/2018

Hall A/C Data Analysis Workshop
Recent Updates to Hall-C Replay

• The experiment specific submodules is where the following should reside:
  • Shell scripts
  • Custom replay scripts
  • DEF-files
  • Monitoring scripts
  • Macros
  • Symbolic links to data files stored in the experiments directory on the RAID disk
  • Parameter files, etc.
  • UTIL OL is general utility submodule for online operations
    • Pedestal monitoring, report file monitoring, …

Eric Pooser
06/25/2018
Hall A/C Data Analysis Workshop
Recent Updates to Hall-C Replay

- New calibration code exists for the calorimeter, drift chambers, and hodoscope’s
  - See various talks during Tuesday morning (10:45 - 12:15) session
- Helicity gated scalers and FADC channels included in ROOT tree
Questions?
Backup Slides