Standalone waveform analyzer & updates on the digitization of the PMT based detectors

Sayak Chatterjee

UMass Amherst

Moller counting meeting, May 5, 2026

Outline

Feature #1176 OPEN

Edit Log time Watch ...

Task #1168: Generalize Detector Decode methods
Task #1187: v1.8 Decoder improvements

Standalone waveform analyzer

Added by Ole Hansen 3 months ago. Updated 28 days ago.

Status:	In Progress	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Sayak Chatterjee	**% Done:**	70%
Target version:	1.8	**Estimated time:**	60.00 h
Responsible:	Ole Hansen		

Description Quote

Port the multi-peak waveform analysis function from hcana into the FADC code in Podd. Make it a standalone version. Input: vector of samples, Output: structure of vectors of peak parameters (leading-edge time, integral, time-over-threshold, pileup status indicator, etc).

The waveform analysis code in hcana is implemented in `void THcRawAdcHit::SetSampIntTimePedestalPeak()`
https://github.com/JeffersonLab/hcana/blob/93fed862f038aecc3fcbc7ce453b6693908f18e8/src/THcRawAdcHit.cxx#L314

Subtasks Add

Related issues Add

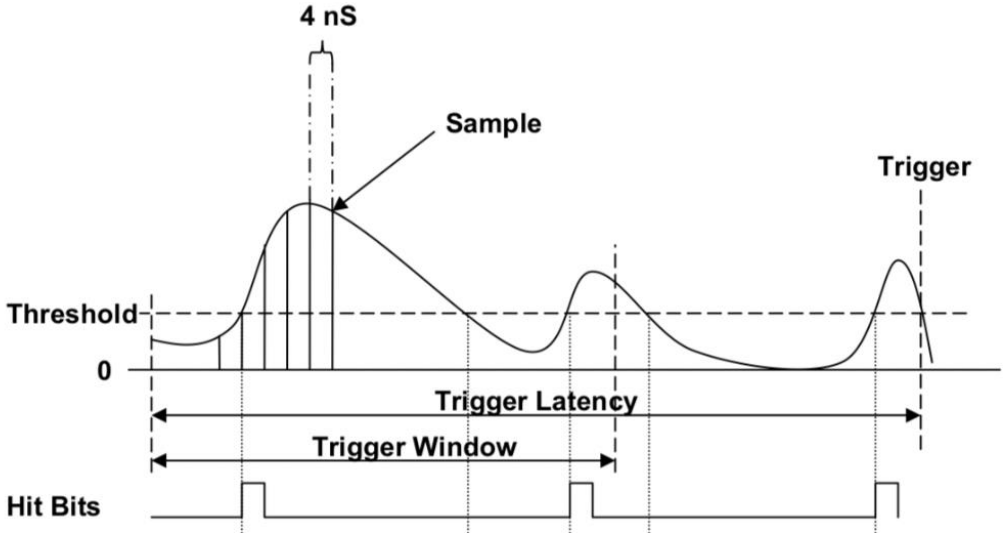
Implementation of the standalone waveform analyzer using the existing implementation in **hcana**

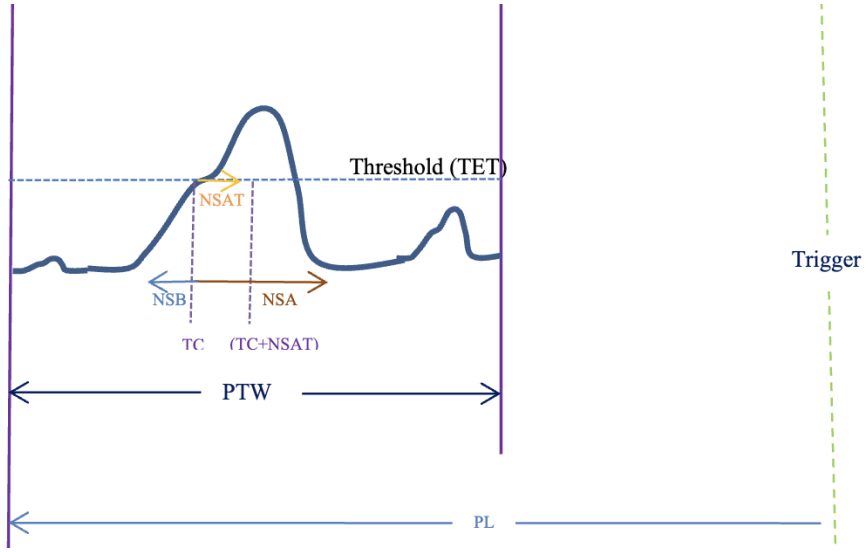
Moller-counting

- Currently moller-counting framework is being used to analyze the data from the cosmic testing of the MOLLER main detector production modules at W&M

- This framework uses the class MOLLERGenericDetector which decodes the fADC data and analyze the waveform and store them to extract amplitude, integral, pedestal and timing information of the individual detector channels

- External database is being used to set the fADC parameters such as, threshold, NSA, NSB, ADC to amplitude conversion factor, gain, integration window etc. for extracting the data





Motivation

- The existing framework stores only one good pulse and does not look for any additional pulses for a particular channel
- We used the existing standalone waveform analyzer from hcana to use with Podd to extract the amplitude, integral, time and pile-up information from the fADC waveforms
- In this study, the main goal is to compare the output of the standalone implementation from hcana with moller-counting
- W&M data has been used to validate the standalone code
- Pulse amplitude, pedestal, integral, timing information are compared using two different implementation
  - MOLLERGenericDetector class
  - THcRawAdcHit class from hcana

Output

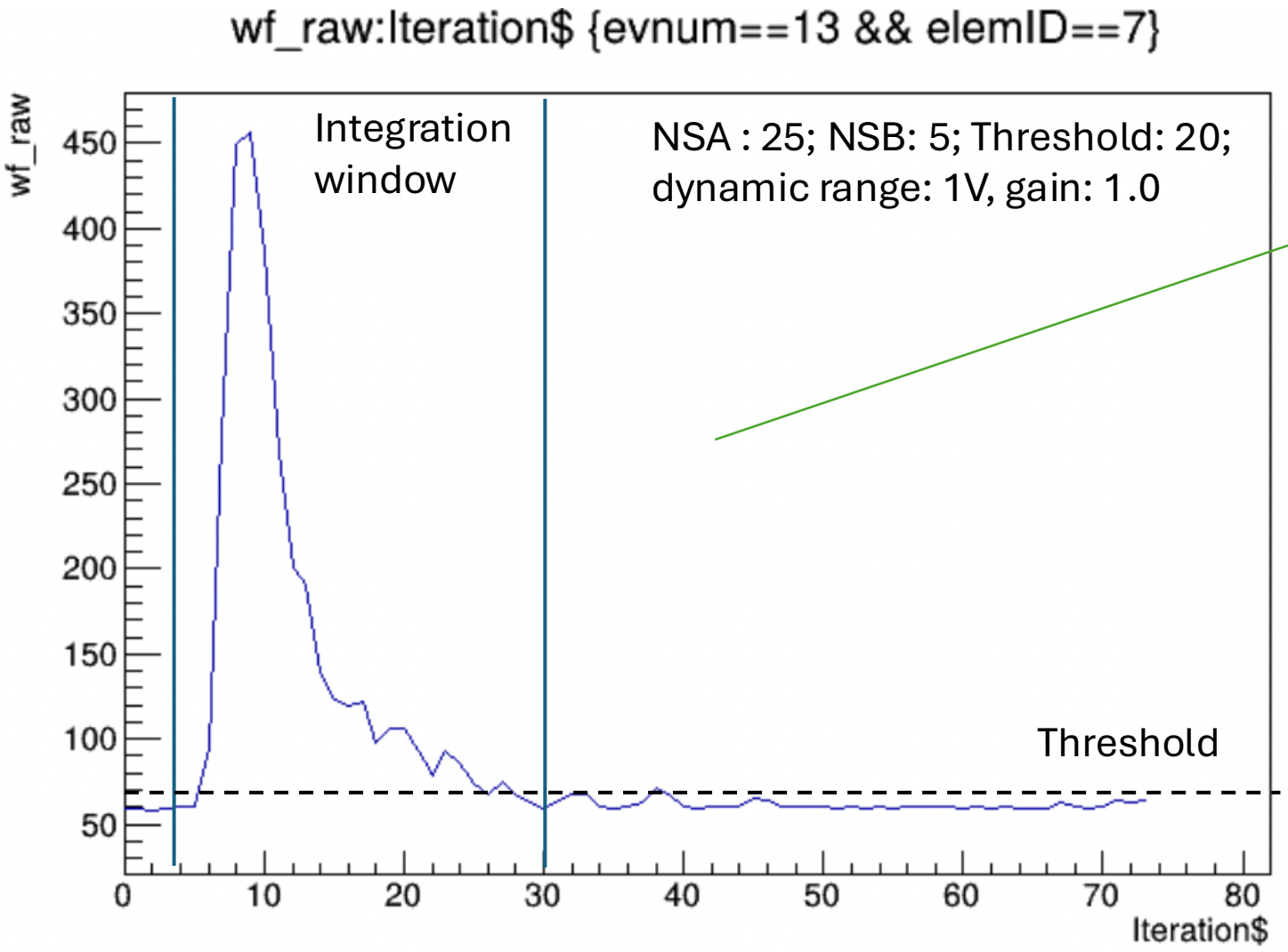
Example from a few events

Element ID	Parameter	Moller-counting (a.u.)	Standalone (a.u.)
7	Raw Amplitude Time Raw integral Integral	582.667 30.8587 382.202 245.155	582.667 30.8125 388.412 246.639
12	Raw Amplitude Time Raw integral Integral	143.531 45.2448 173.643 30.8191	143.531 45.1875 178.584 30.8347

```
root [1] T->Scan("Entry$:evnum:elemID:npulses");
*************************************************************
*    Row    *    Entry$ *     evnum *    elemID *   npulses *
*************************************************************
*        0 *         0 *         5 *         3 *         2 *
*        1 *         1 *         5 *        21 *         3 *
*        2 *         2 *         6 *         2 *         2 *
*        3 *         3 *         9 *         1 *         2 *
*        4 *         4 *        10 *         0 *         2 *
*        5 *         5 *        10 *         1 *         2 *
*        6 *         6 *        10 *         7 *         2 *
*        7 *         7 *        10 *        21 *         2 *
*        8 *         8 *        10 *        23 *         2 *
*        9 *         9 *        13 *         7 *         2 *
*       10 *        10 *        14 *         1 *         2 *
*       11 *        11 *        16 *         7 *         2 *
*       12 *        12 *        17 *         1 *         2 *
*       13 *        13 *        17 *         2 *         3 *
*       14 *        14 *        17 *        14 *         2 *
*       15 *        15 *        17 *        21 *         3 *
*       16 *        16 *        18 *        21 *         2 *
*       17 *        17 *        19 *         0 *         2 *
*       18 *        18 *        19 *         1 *         2 *
*       19 *        19 *        20 *        21 *         2 *
*       20 *        20 *        21 *         7 *         2 *
*       21 *        21 *        23 *         1 *         3 *
*       22 *        22 *        24 *         7 *         2 *
*       23 *        23 *        26 *         2 *         2 *
*       24 *        24 *        26 *         3 *         2 *
Type <CR> to continue or q to quit ==>
```

Consistent results from the standalone waveform analyzer

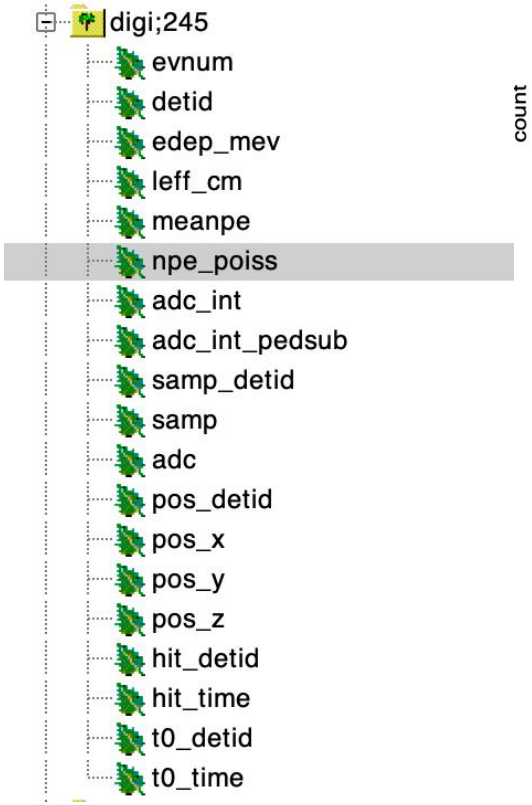
Output

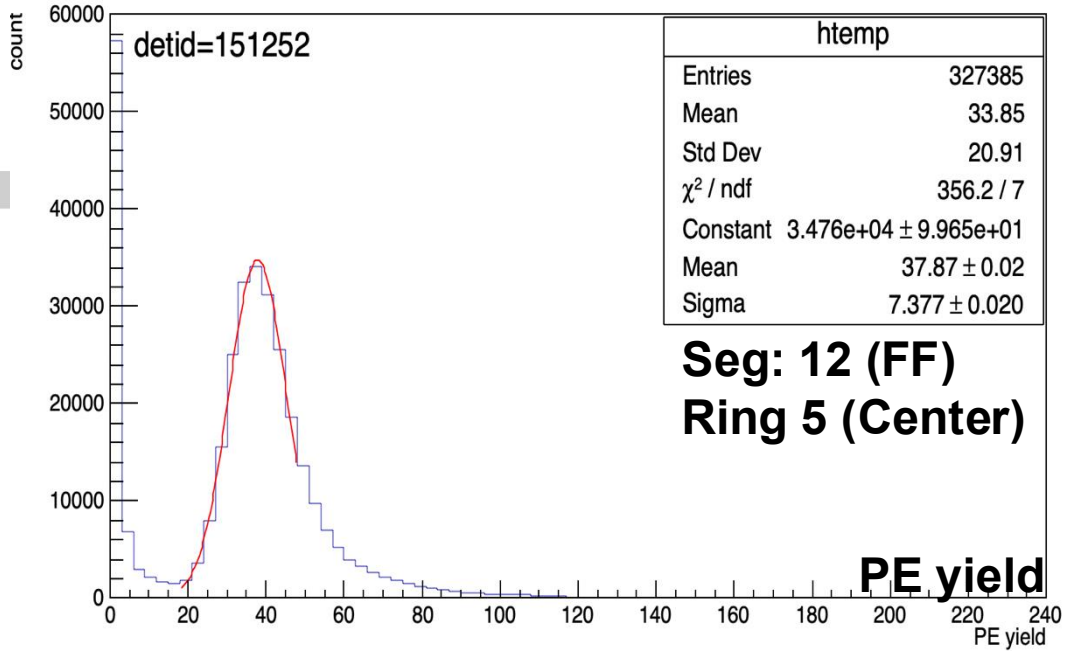


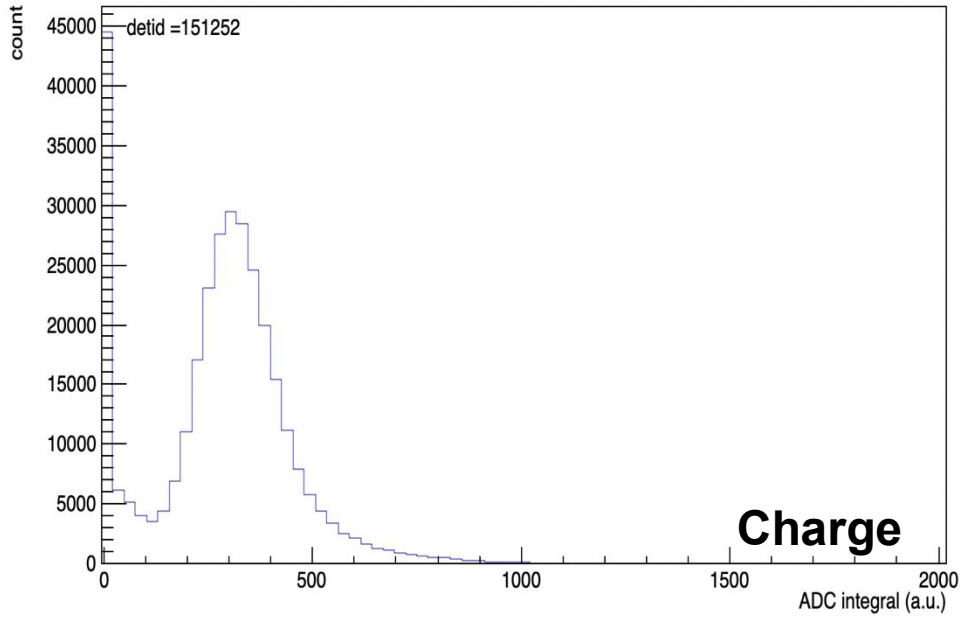

```
root [1] T->Scan("Entry$:evnum:elemID:npulses");
************************************************************
*    Row    *    Entry$ *     evnum *    elemID *   npulses *
************************************************************
*        0 *         0 *         5 *         3 *         2 *
*        1 *         1 *         5 *        21 *         3 *
*        2 *         2 *         6 *         2 *         2 *
*        3 *         3 *         9 *         1 *         2 *
*        4 *         4 *        10 *         0 *         2 *
*        5 *         5 *        10 *         1 *         2 *
*        6 *         6 *        10 *         7 *         2 *
*        7 *         7 *        10 *        21 *         2 *
*        8 *         8 *        10 *        23 *         2 *
*        9 *         9 *        13 *         7 *         2 *
*       10 *        10 *        14 *         1 *         2 *
*       11 *        11 *        16 *         7 *         2 *
*       12 *        12 *        17 *         1 *         2 *
*       13 *        13 *        17 *         2 *         3 *
*       14 *        14 *        17 *        14 *         2 *
*       15 *        15 *        17 *        21 *         3 *
*       16 *        16 *        18 *        21 *         2 *
*       17 *        17 *        19 *         0 *         2 *
*       18 *        18 *        19 *         1 *         2 *
*       19 *        19 *        20 *        21 *         2 *
*       20 *        20 *        21 *         7 *         2 *
*       21 *        21 *        23 *         1 *         3 *
*       22 *        22 *        24 *         7 *         2 *
*       23 *        23 *        26 *         2 *         2 *
*       24 *        24 *        26 *         3 *         2 *
Type <CR> to continue or q to quit ==>
```

Digitization of the PMT based detector hits

- A stand alone script (similar to libsbsdig) to take remoll hit input and give photon yield (PE) and hit time for respective MD rings
- I have started using the existing look-up table for the MD response (DocDb: 1508, 1533)
- I have started using the shower-max lookup table to develop the code for shower-max







Digitization of the PMT based detector hits

- Immediate next step is to integrate the standalone code to the moller-dig framework developed by Chandan which can be used by the moller-counting framework directly

    - moller-dig: Input from REMOLL and produces an output that can be analysed by moller-counting directly (**work in progress**)

- Similar implementation for Shower-max (started), pion detector and trigger scintillators

- Initially have the PE and hit time information from the individual detector for tracking and later depending on the need, waveform can also be included as a part of the framework

Thank you for your attention!!!