MMDAQ Noise Output

Oleksandr Borysov
Tel Aviv University

FCAL TB 2015 Analysis Meeting
January 25, 2016

MMDAQ root File Content

```
TFile**         /data/aborysov/tau_muon_run/run69.root
 TFile*         /data/aborysov/tau_muon_run/run69.root
  KEY: TTree raw;1      rawapvdata                          // physics run data
  KEY: TTree pedestals;1    apvpedestals                    // pedestal and noise
  KEY: TTree data;1     apvdata                             // some information from raw
  KEY: TTree run_info;1     run_info
  KEY: TDirectoryFile   config;1  parameters from config file
```

- Raw is the tree which contains signal of APV channels which exceeds the threshold;
- For Physics run pedestals tree contains only one entry with the noise and pedestals data used during the run.

MMDAQ Pedestal and Noise Calculation

For single event:

```
void CApvEvent::calculate_pedestals()
  ....
  double mean = GetVectorMean(channel_data);
  double stdev = GetVectorStdDev(channel_data);
  ....
```

```
double sum = std::accumulate(vec.begin(), vec.end(), (double)0.0);
      return sum / (double)(vec.size());

sumdiffs += (*iter - vecmean) * (*iter - vecmean);
....
return sqrt( sumdiffs / (n-1) );
```

Result of the run
(Welford algorythm)

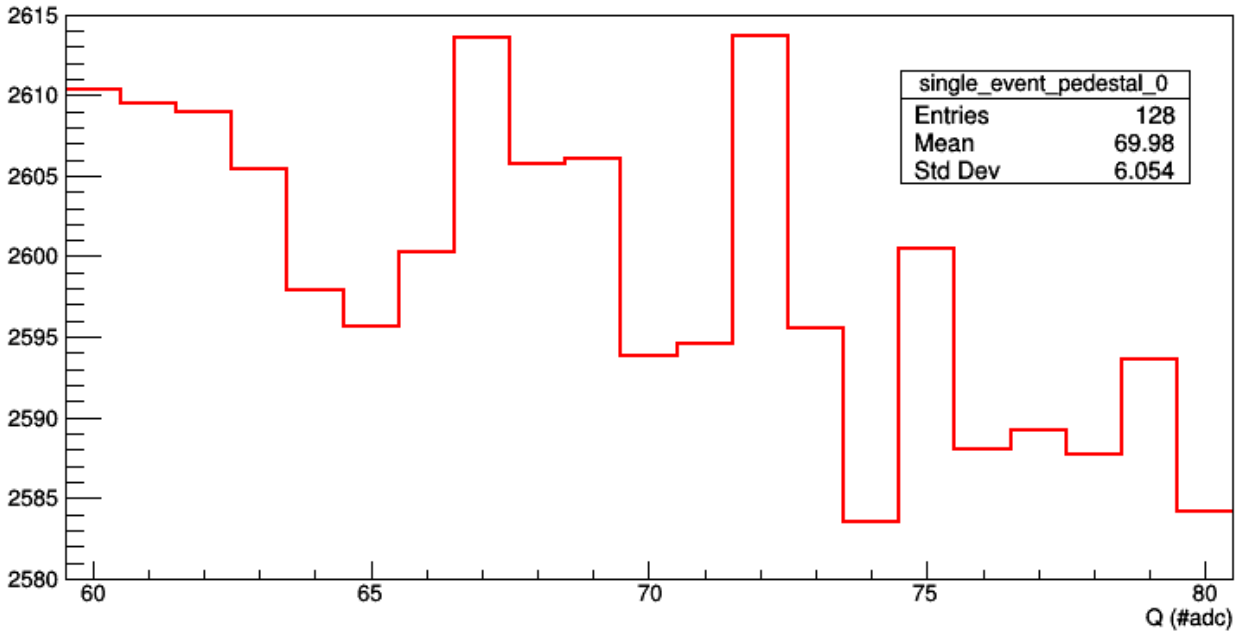
```
void CRootWriter::process_pedestal_event(const CMMEvent* const  curr_mmevent)
.........
pedestal_map[icurr_ch->first][0] += (curr_mean - prevmean) /
                                    (m_event_counter_pedestals + 1); //mean of mean

pedestal_map[icurr_ch->first][1] += (curr_stdev - prevstdev) /
                                     (m_event_counter_pedestals + 1); //mean of stdev

pedestal_map[icurr_ch->first][2] += ( (curr_mean - prevmean) *
                        (curr_mean - pedestal_map[icurr_ch->first][0] ) ); //sigma
```

Pedestal

Noise

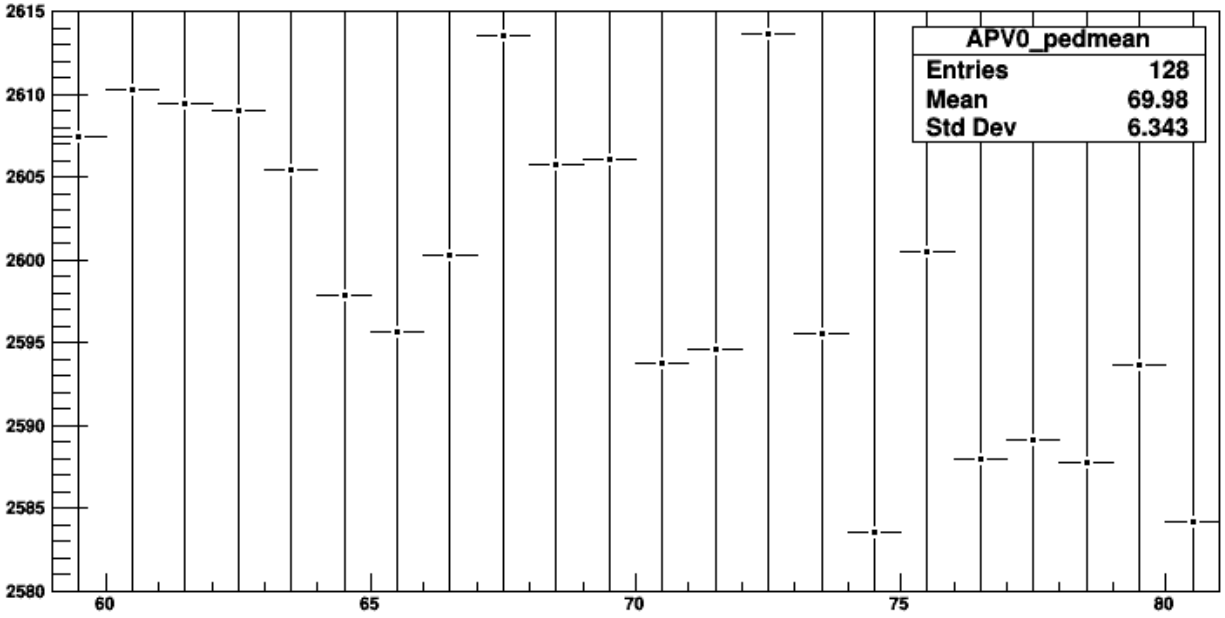
TAU run 22, root vs mmdaq event 2809

The last entry from the tree



- The last entry in the “pedestals” tree contains the result of calculation and is used by MMDAQ during the physics run.
- For the physics run “pedestals” tree has one entry with the data used fro data processing.

Loaded by MMDAQ



pedestals Tree in MMDAQ root file

```
atree->SetBranchAddress("apv_evt",    &p_m_apv_evt);
atree->SetBranchAddress("time_s",     &p_m_time_s);
atree->SetBranchAddress("time_us",    &p_m_time_us);
atree->SetBranchAddress("apv_fecNo", &p_m_apv_fec);
atree->SetBranchAddress("apv_id",     &p_m_apv_id);
atree->SetBranchAddress("apv_ch",     &p_m_apv_ch);
atree->SetBranchAddress("mm_id",      &p_m_mm_id);
atree->SetBranchAddress("mm_readout", &p_m_mm_readout);
atree->SetBranchAddress("mm_strip",   &p_m_mm_strip);
atree->SetBranchAddress("apv_pedmean",      &p_m_apv_pedmean);
atree->SetBranchAddress("apv_pedsigma",      &p_m_apv_pedsigma);
atree->SetBranchAddress("apv_pedstd",      &p_m_apv_pedstd);
```

Just to start:

- Check apv_pedmean, apv_pedsigma and apv_pedstd as a function of bias voltage;
- Distributions for the whole APV chip;
- Graphs for some channels with different size;
- List of the channels where these values have weird numbers (too big, too small, etc.);
- Note, even the beam is in a particular place of the detector, the pedestal and noise are calculated for all channels and all of them are susceptible to the bias voltage.