Integrating Pulser Into Readout

Round 2

Introduction

- Many components of the HPS experiment, from trigger tuning to analysis, depend on accurate Monte Carlo simulations.

- One potential option to help improve the accuracy and quality of Monte Carlo simulation is to integrate real background data into Monte Carlo signal data.

- Previously, it was proposed that reconstructed events from pulser data be used to emulate “truth hits” from SLiC, and merged into signal Monte Carlo.
  - This is nice because it does not require modification of readout.
  - This carries some issues with accurately tuning the conversion to match the background post-readout and issues with noise handling.

Introduction

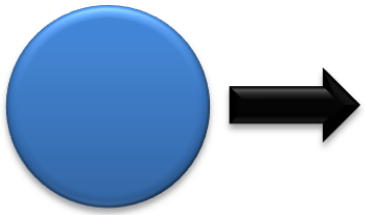
- Instead, we would prefer to directly inject pulser data ADC samples into the readout data.

- How might this be done?
  - To gain a better understanding of this, we should consider how the readout process works.

The Readout Process

- The readout process starts with SLiC output.
- SLiC (via GEANT4) produces truth hits, which contain a few relevant pieces of information:
  - Channel ID
    - On which channel that hit occurred.
  - Time
    - When the hit occurred.
  - Energy deposition
    - The energy which was deposited into the channel.

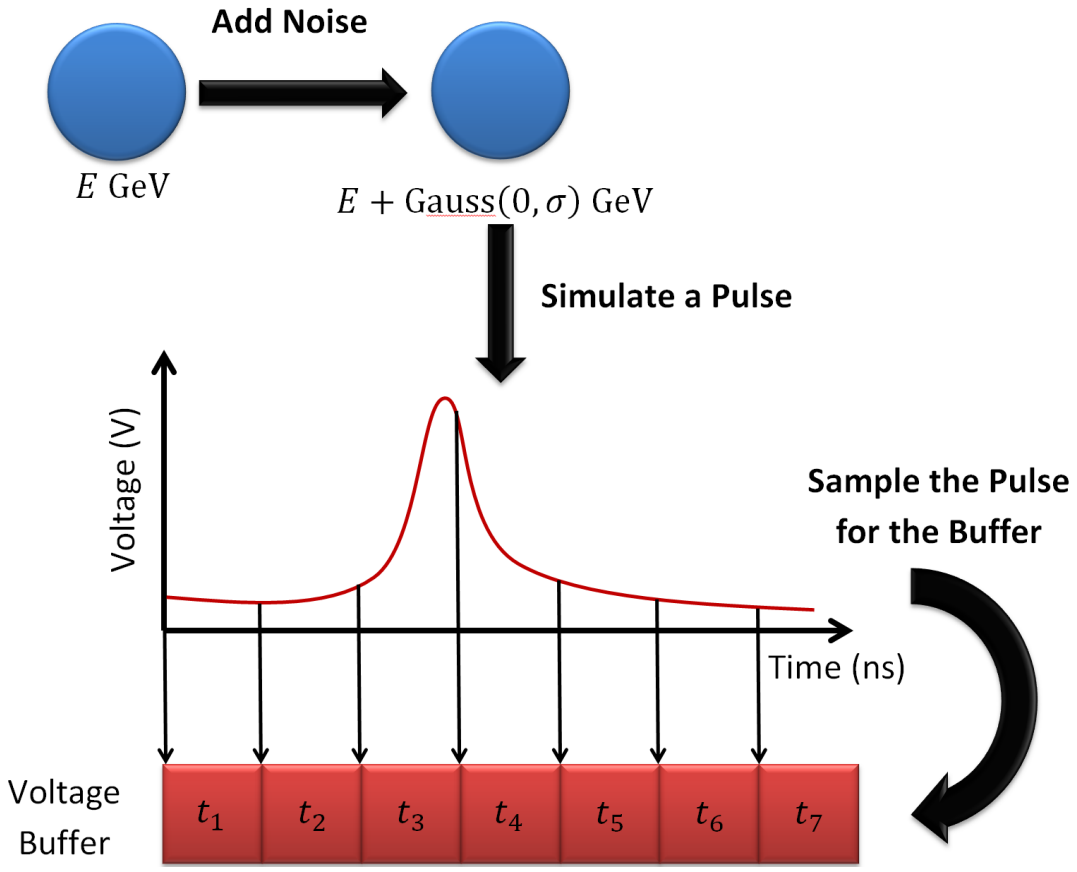
SLiC Truth Hit



- Energy Deposition (E_{truth})
- Time (t)
- Channel ID

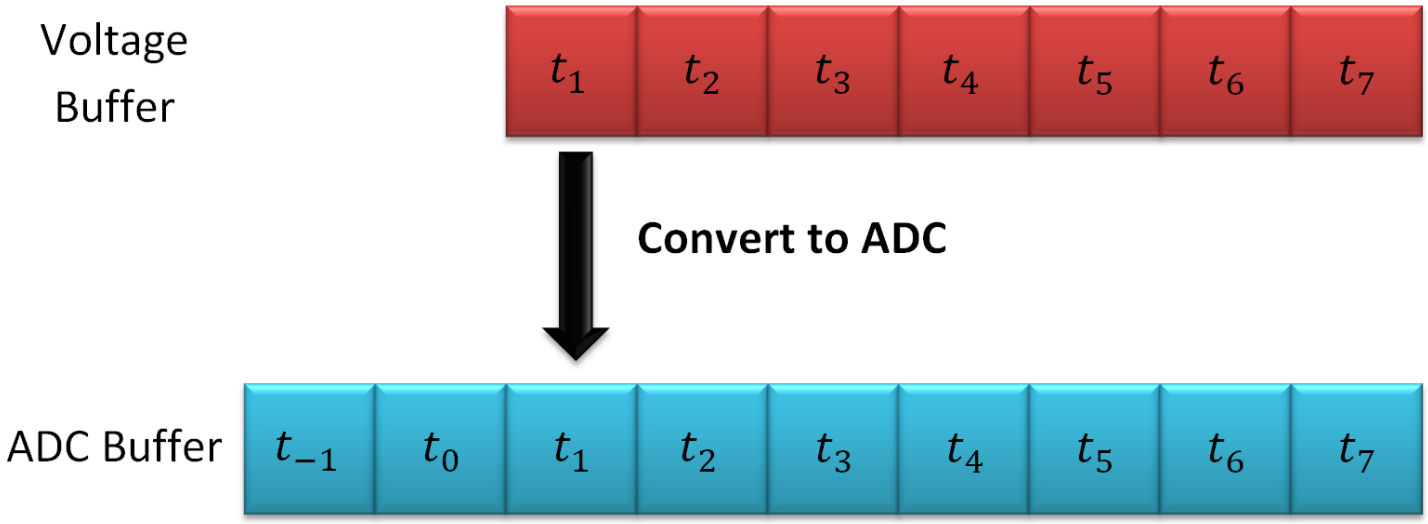
The Readout Process

- Each truth hit is read into the hit digitization driver, where it is processed.
  - The hit energy is read.
  - The energy is "fudged" by a Gaußian value.
  - A voltage pulse is emulated and sampled at various points.
  - Each of these samples is added to an internal voltage buffer.



The Readout Process

- The driver's internal clock keeps track of which point in the voltage buffer is the "current" time.
- Each event, the "current" voltage buffer cell is read and converted to ADC to be written into the ADC buffer.
  - The pedestal is added at this stage as well.

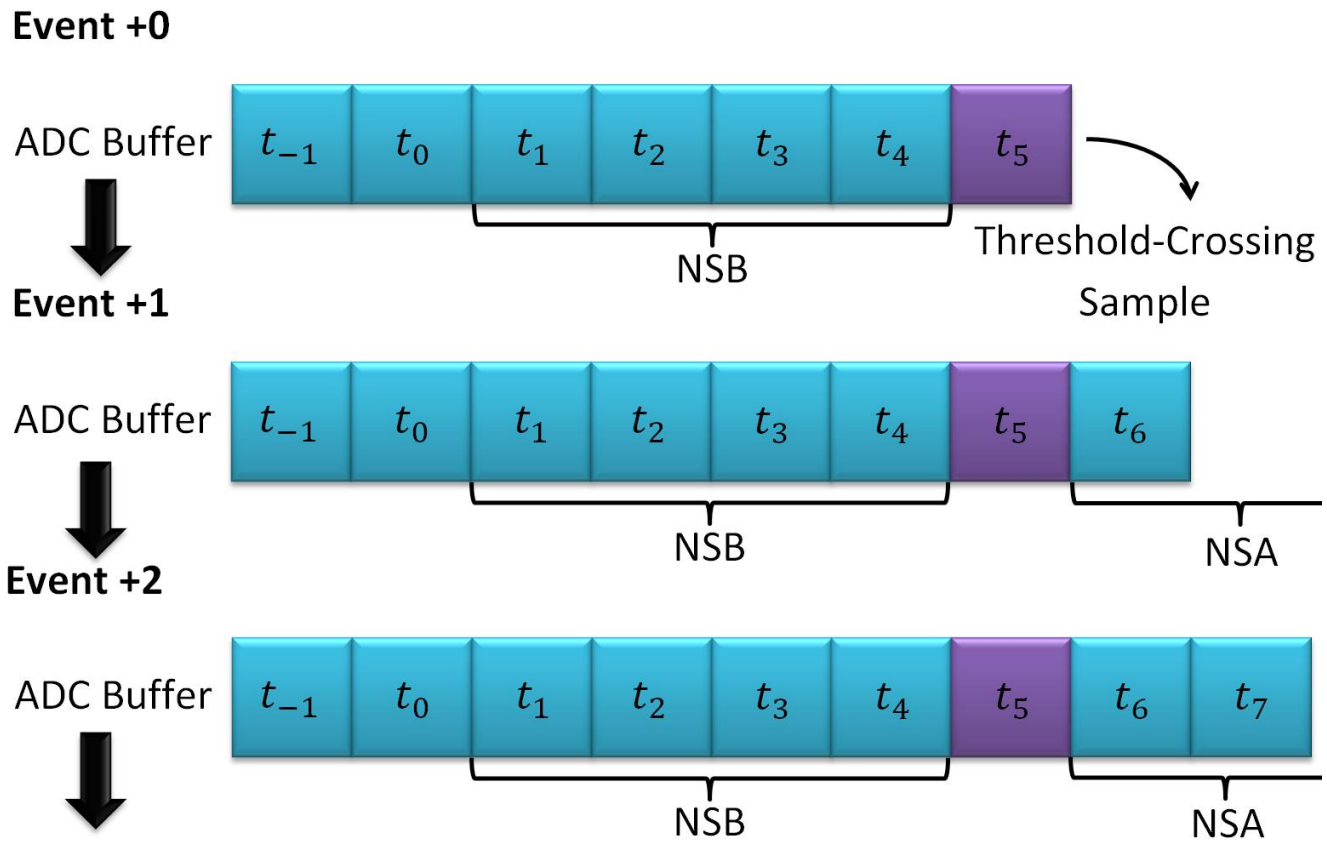


The Readout Process

- Once the ADC buffer is written for each channel, it is then itself sampled.
- If the ADC buffer value exceeds the integration threshold, integration is initiated for that channel.
  - A number of samples before the threshold-crossing sample (TCS) equal to NSB (**n**umber of **s**amples **b**efore) are then summed together.
  - The TCS itself is also added to the sum.
- Each event thereafter, when a new cell of the ADC buffer is written, that cell is added to the total integrated ADC sum for the channel.
  - This continues until a total number of samples equal to NSA (**n**umber of **s**amples **a**fter) have been added.
- The total integrated value is then output as a hit to be used by the trigger simulation.

The Readout Process

- Once the ADC buffer is written for each channel, it is then itself sampled.



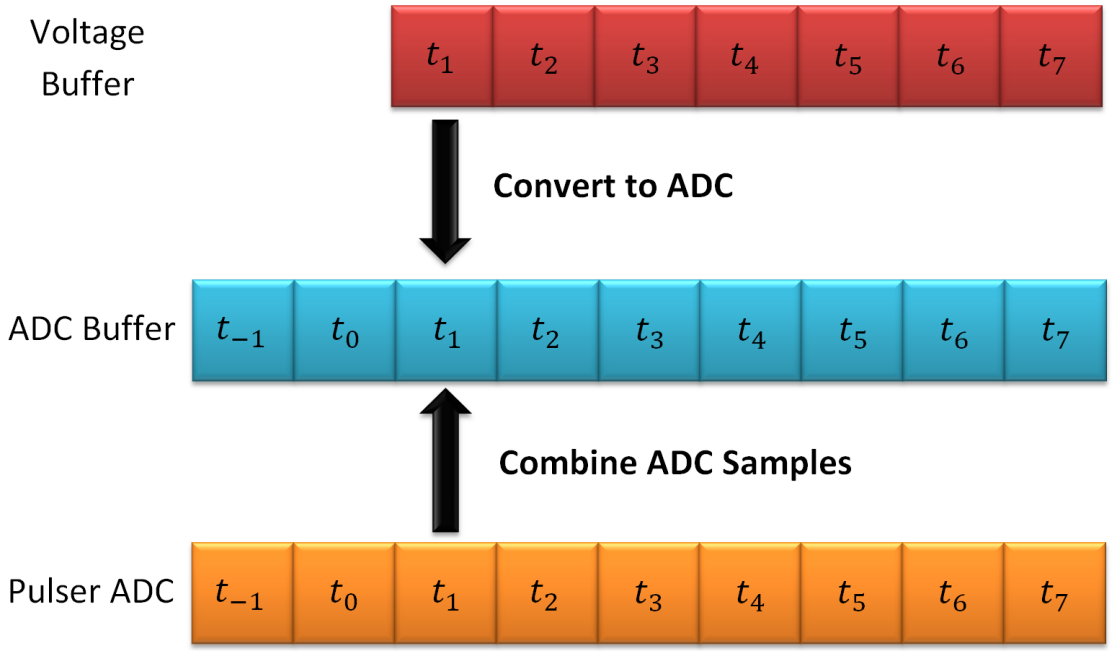
$$E_{\mathrm{ADC}} = \sum_{i=1}^{5} E_{t_i}$$

$$E_{\mathrm{ADC}} = \sum_{i=1}^{6} E_{t_i}$$

$$E_{\mathrm{ADC}} = \sum_{i=1}^{7} E_{t_i}$$

Merging Real Data

- The ideal point to merge real data is concurrent with the conversion of the "current" voltage buffer sample to ADC.
  - Proper electronic noise is not simulated, so ADC noise is not repeated.
  - ADC values added in this way will automatically contribute to hit integration and energies, and be perceived by the trigger.



Merging Real Data

- How would this process be performed? There are several steps that need to be performed:

Merging Real Data

- How would this process be performed? There are several steps that need to be performed:
  - **Run filtering:** It is necessary to select only runs that contain useful pulser data. Runs that have incorrect beam energies, or where the SVT is in the wrong position, or bearing similar deviations from the simulated signal data must be excluded.

Merging Real Data

- How would this process be performed? There are several steps that need to be performed:
  - **Run filtering**
  - **Event filtering:** Criteria must be established to exclude poor quality events. Events where there is no beam or where the SVT bias is off, for instance, do not provide useful source data.

Merging Real Data

- How would this process be performed? There are several steps that need to be performed:
  - **Run filtering**
  - **Event filtering**
  - **Data randomization:** Pulser events must be placed in a random order so that each MC run that uses this technique does not have the same background signature.

Merging Real Data

- How would this process be performed? There are several steps that need to be performed:
  - **Run filtering**
  - **Event filtering**
  - **Data randomization**
  - **Pulse cleaning:** The ADC samples in the readout data need to be cleaned. (More on this in a moment.)

Merging Real Data

- How would this process be performed? There are several steps that need to be performed:
  - **Run filtering**
  - **Event filtering**
  - **Data randomization**
  - **Pulse cleaning**
  - **Readout modification:** The readout system needs to be modified to be able to read in the pulser data during readout of a signal Monte Carlo file, and to properly combine the ADC values.

Merging Real Data

- There are several important challenges that come with this method!

Merging Real Data

- There are several important challenges that come with this method!

- The readout system must be modified.
  - The digitization driver needs yet more options added into an already large driver.
    - This increases code complexity and makes maintainability more challenging.
    - This requires fairly rigorous testing; the digitization driver is the lowest-level driver in the calorimeter readout chain, and any mistake would break the entire trigger simulation.

Merging Real Data

- There are several important challenges that come with this method!

- The readout system must be modified.
  - The digitization driver needs yet more options added into an already large driver.
    - This increases code complexity and makes maintainability more challenging.
    - This requires fairly rigorous testing; the digitization driver is the lowest-level driver in the calorimeter readout chain, and any mistake would break the entire trigger simulation.

- This method requires complicated cleaning of ADC values in data to be usable.

ADC Cleaning

- There is a key difference between readout data and readout MC input.
  - Readout data is output in 200 ns blocks, which contain all of the ADC samples in that time range.
  - Monte Carlo readout input is formatted into 2 ns beam bunches, and is representative of continuous data.

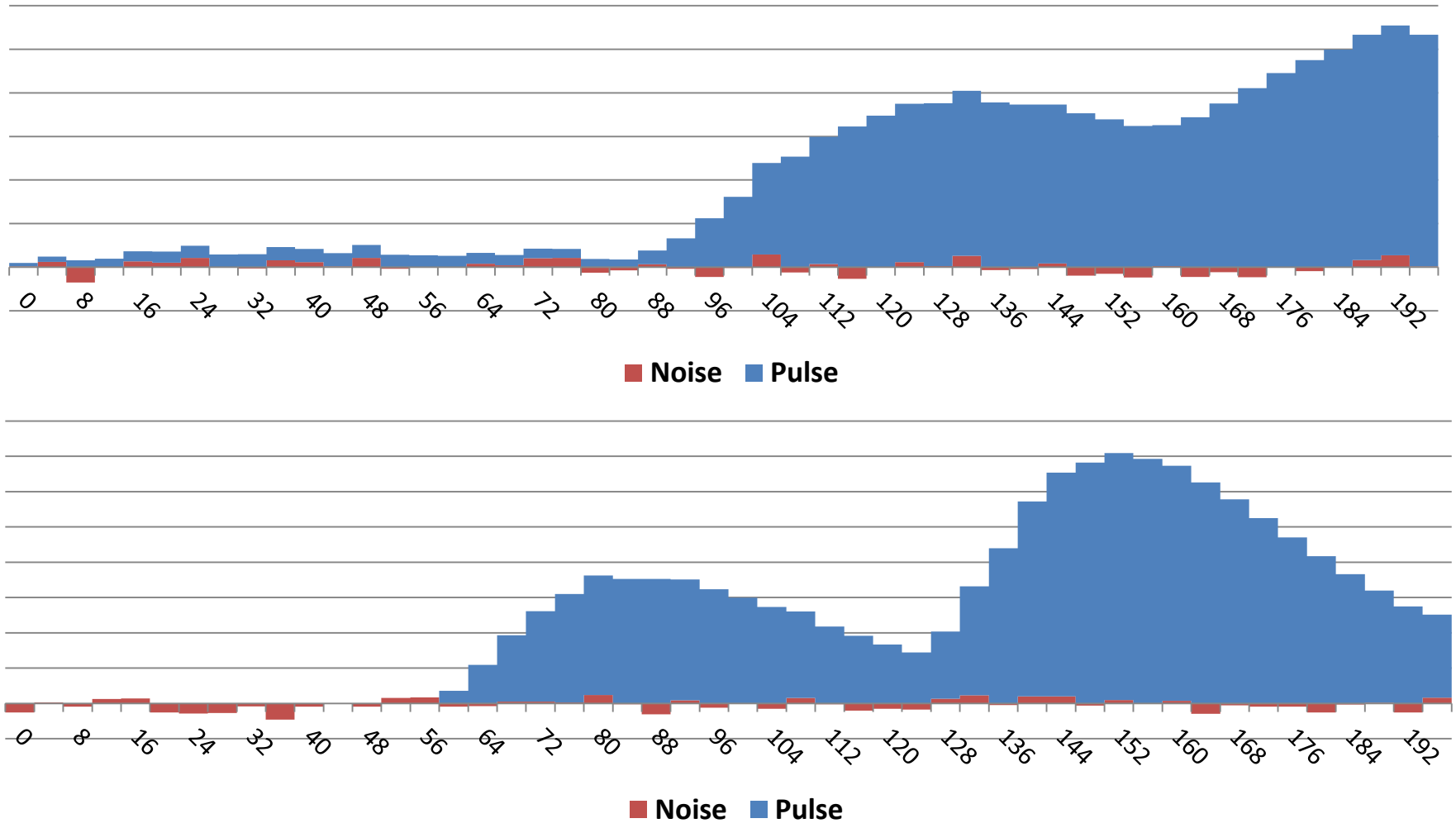
- The means that the ADC samples across multiple 200 ns readout events must be formatted so that they can be treated as one continuous chain of ADC data to work properly in the readout simulation.

ADC Cleaning

- The problem with doing this is edge effects.
  - Imagine if a pulse occurred very near the beginning (or end) of a readout window.
  - That window will now start (or end) with a high ADC value.
  - Imagine that the preceding window was at essentially pedestal at its end (or beginning).
  - There is now a discontinuous jump between the event values!

ADC Cleaning

- Consider two consecutive events.



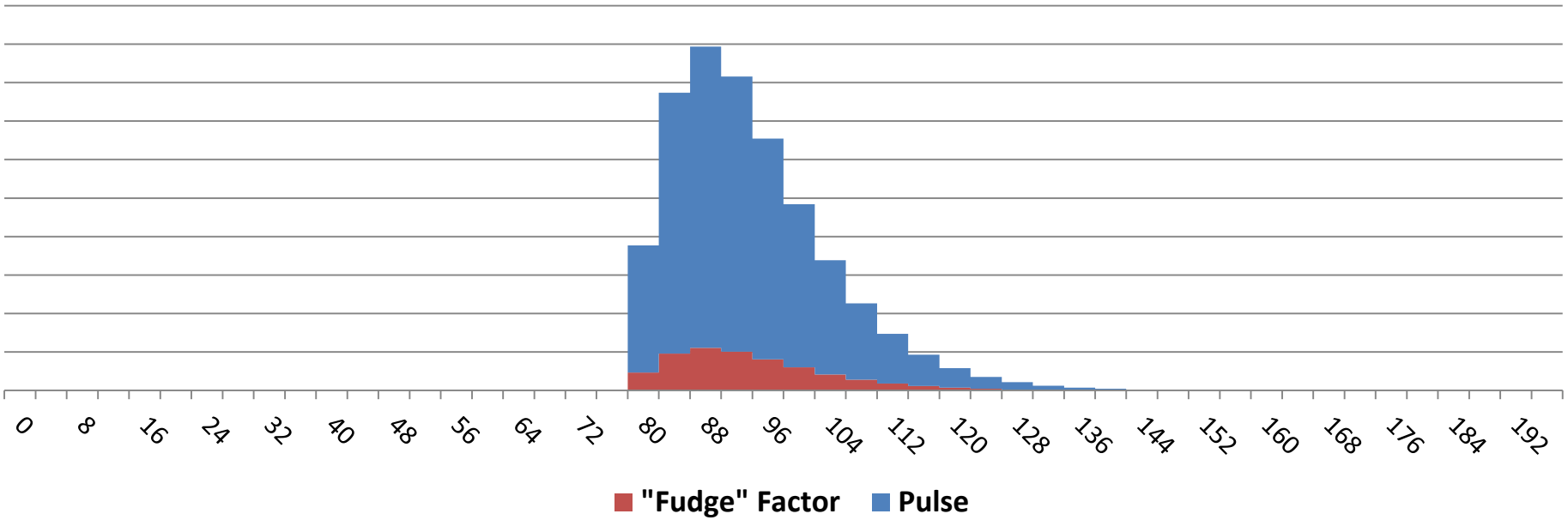
ADC Cleaning

- Combining these pulses is not a trivial task!
  - Add to this that we can not simply skip these pulses.
  - We need the ADC data from the calorimeter and the data from the SVT to align, so we can break apart a given pulser event.
  - Something must be done to allow us to obtain functionally continuous ADC data from the real data.

- It may be better to consider alternatives.

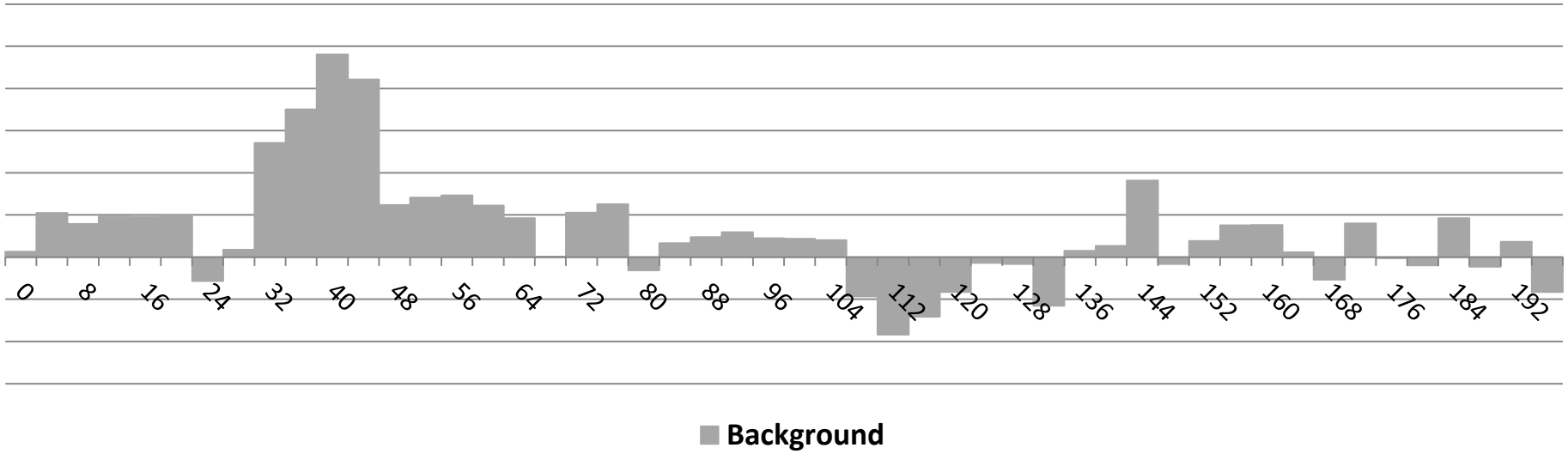
An Alternate Approach

- Since the hardest part of fully integrating ADC data into the readout simulation is the edge effects, why no skip this part entirely?
- Instead, run the readout simulation entirely as normal.
  - This gives a simulated readout event with just the MC signal hits in the ADC buffer.



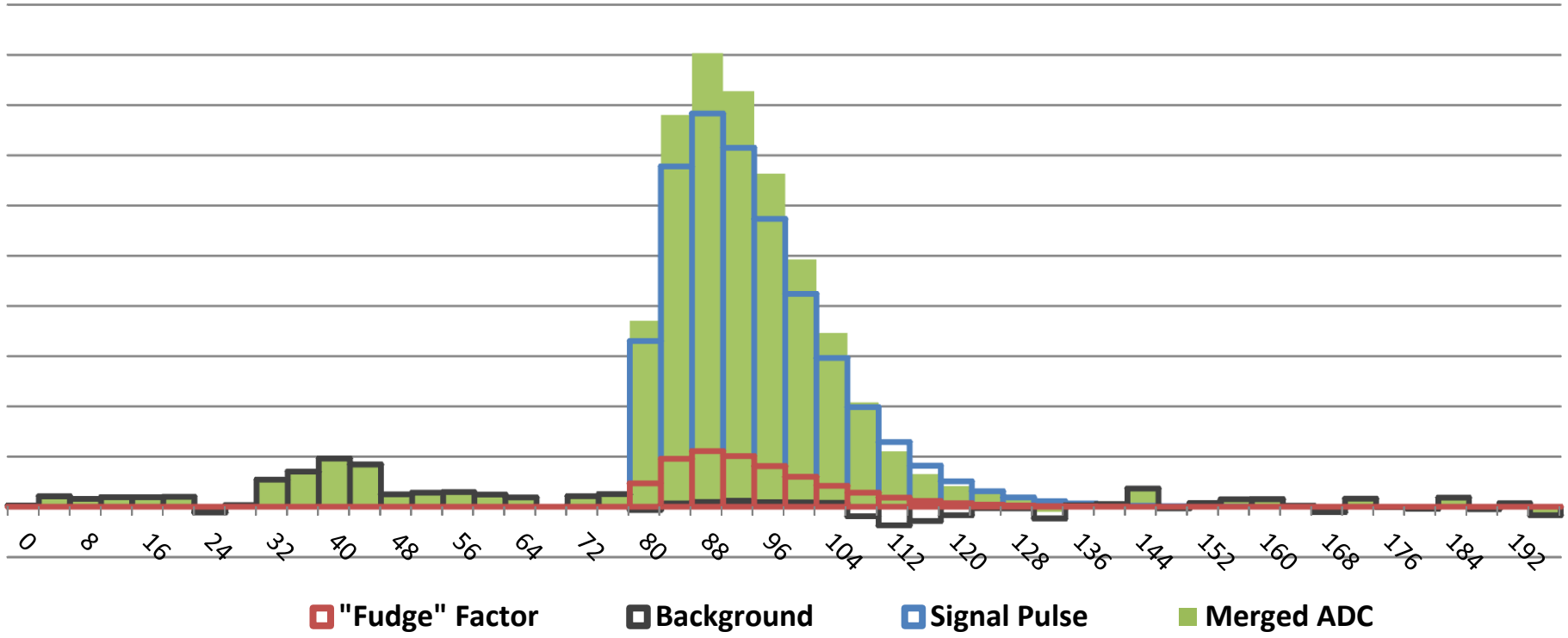
An Alternate Approach

- Next, take a random good pulser event from a good pulser and extract the ADC for the same channel as the MC readout event.



An Alternate Approach

- Then, merge the two by adding the ADC values together.
  - Since these are now both 200 ns readout events, the edge effects don't matter.
  - Electronic noise isn't simulated in the digitization driver, so the noise from the data doesn't add unnecessary additional noise.



An Alternate Approach

- This method has several distinct advantages:
  - It does not require any alteration to the readout system.
    - The merging can be easily performed as a secondary step after readout is complete.
  - It does not require any fancy processing of readout data ADC samples.
    - Edge effects will still exist, but they will manifest identically to how they appear in real data, so this is good.

- There is only one real issue.
  - The trigger does not see any background hits, so we do not observe effects like coincidental triggers.

An Alternate Approach

- This requires most of the same preprocessing steps as the more robust version:
  - **Run filtering**
    - We still need to select only good runs that match the simulated data conditions.
  - **Event filtering**
    - It is still important that we drop pulser events where there I no beam or similar problems are present.
  - **Data randomization**
    - We still want to make sure that the same background signature is not used for each MC run.

Conclusion

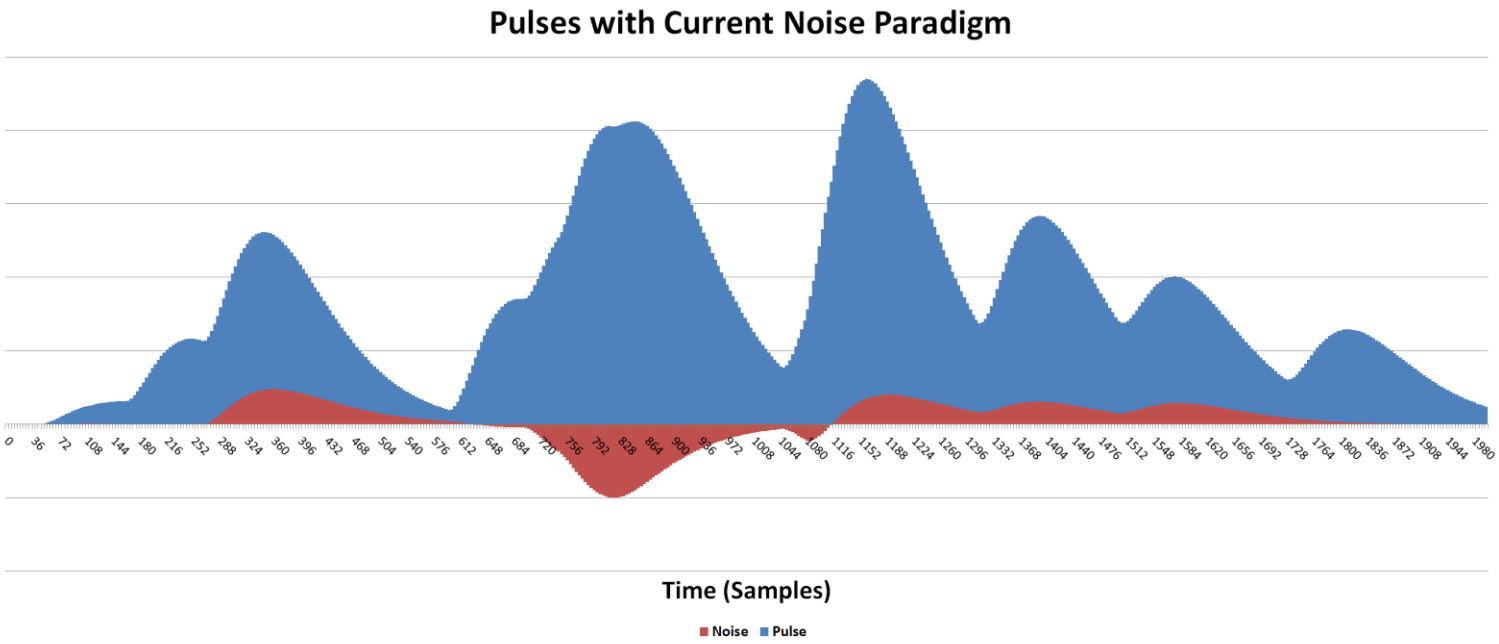
- There are two ways to directly merge readout ADC data with Monte Carlo simulation data.

- The more formally accurate way is to perform this step during readout.
  - This requires complicated processing of the data ADC samples to get data into the right format to work in readout.

- The somewhat looser way to merge them is to simply combine a pulser readout event with a MC signal readout event after the simulation ends.
  - This avoids any complicated data processing or code alteration, but the trigger simulation never sees the background.

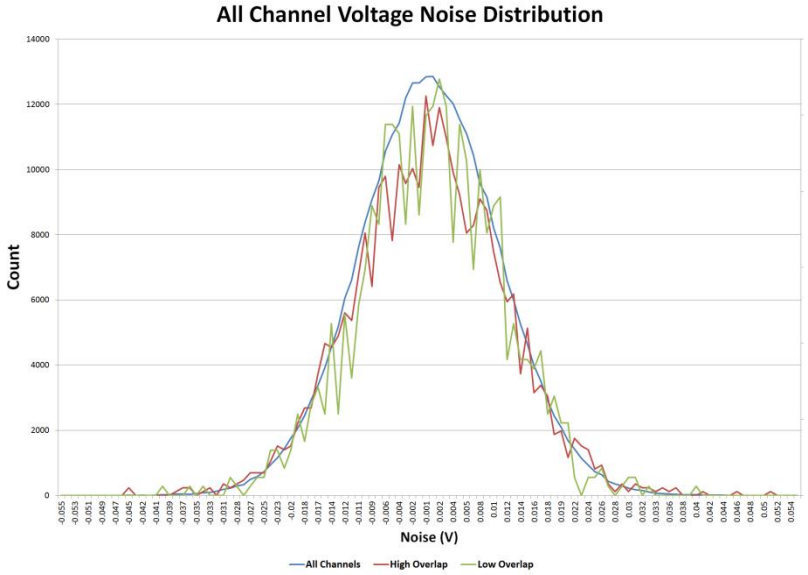
Extra – On Noise

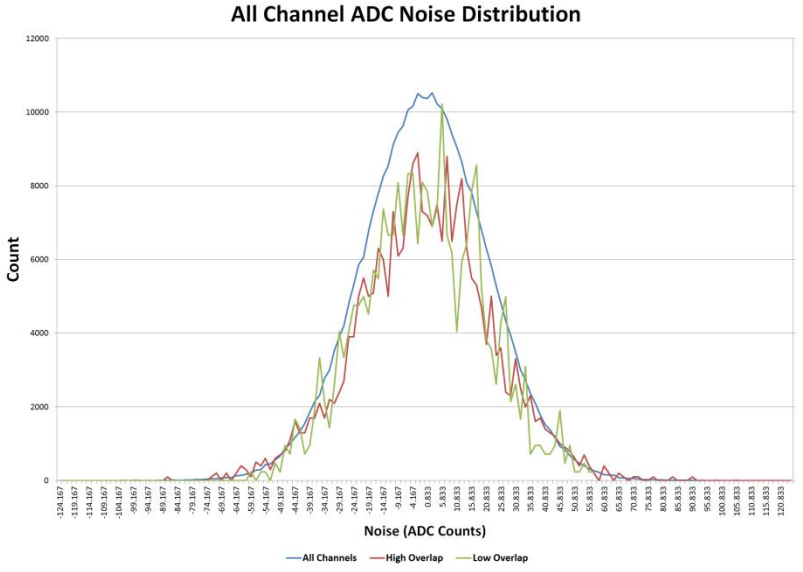
- The readout simulation does not simulate electronic noise at all.
- Currently, it simulates a “noise effect” simply by skewing the input truth hit energy by a Gaußian value.
- This does introduce come randomization, but it skews the entire pulse in the same manner, rather than on a sample-by-sample basis.



Extra – On Noise

- How does this affect the noise on a given ADC sample?
  - It is first worth considering how much total noise is added per 2 ns beam bunch. This is plotted below for all channel (**BLUE**), a high-occupancy channel (**RED**), and a low-occupancy channel (**GREEN**).

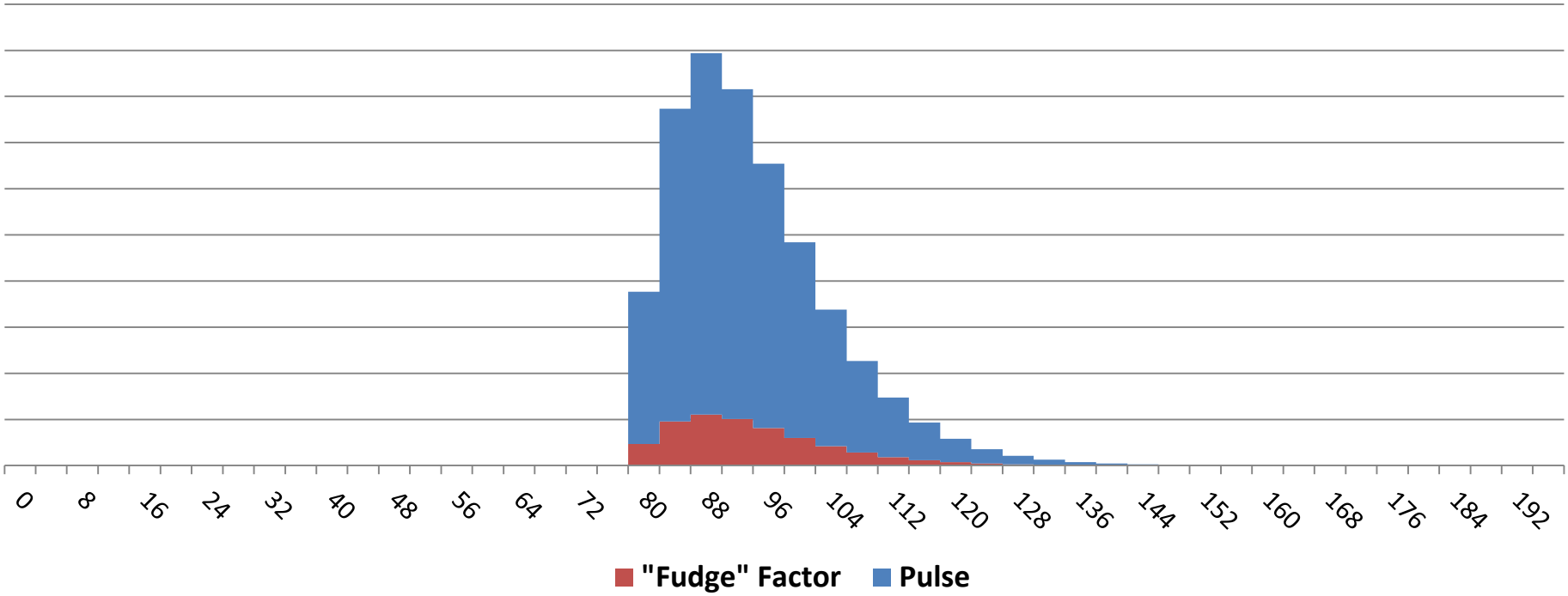




- About the same amount of noise is added, on average, from each 2 ns beam bunch, with a solid Gaußian distribution.

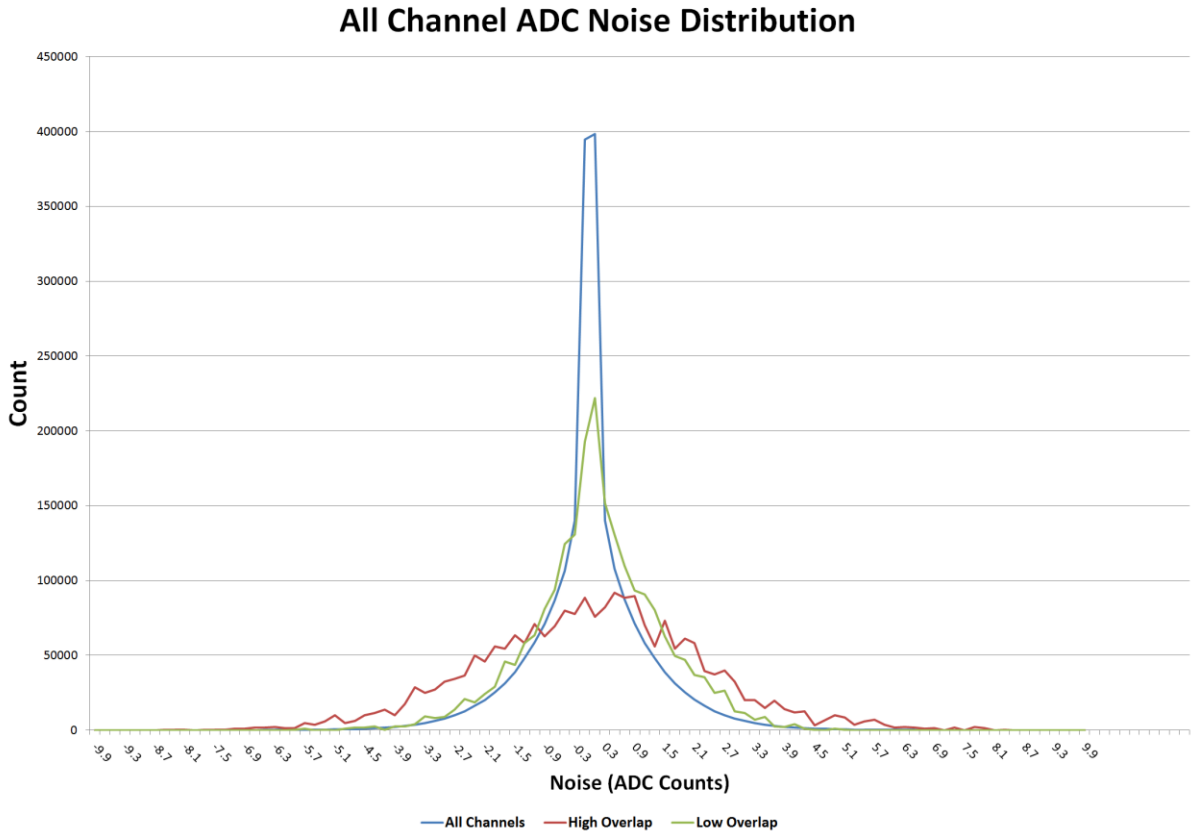
Extra – On Noise

- So it's okay? Not exactly.
- Because the noise is added to the *hit*, and not the actual sample, it is in truth spread out.
  - It is spread across multiple ADC samples because it is effectively built into the simulated voltage pulse.



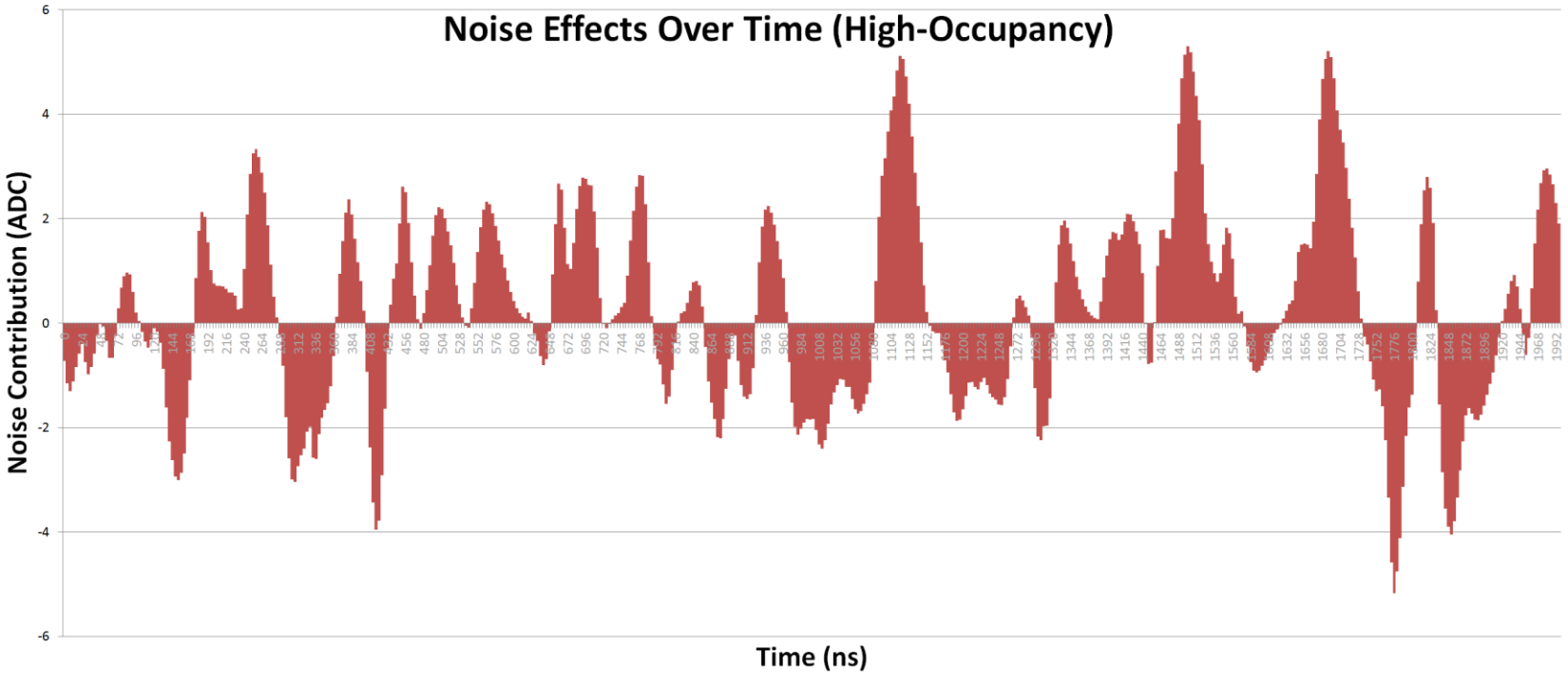
Extra – On Noise

- What does this mean in practice?
  - The amount of noise added to a given ADC channel is actually a function of the occupancy of that channel.
  - If a channel has more hits, there are more "fudged" noise additions overlapping a given ADC sample.



Extra – On Noise

- This gives an actual ADC buffer like the below (actual simulation):



- Basically, there's a noise "pulse" every time there is a hit on the channel.

Extra – On Noise

- This is not realistic.
- Real electronic noise should be random and change each ADC sample; it should have any correlation to the pulse ADC from signal.
- This can also cause issues with time resolution; without the random fluctuation of the ADC samples, time resolution can be artificially good.

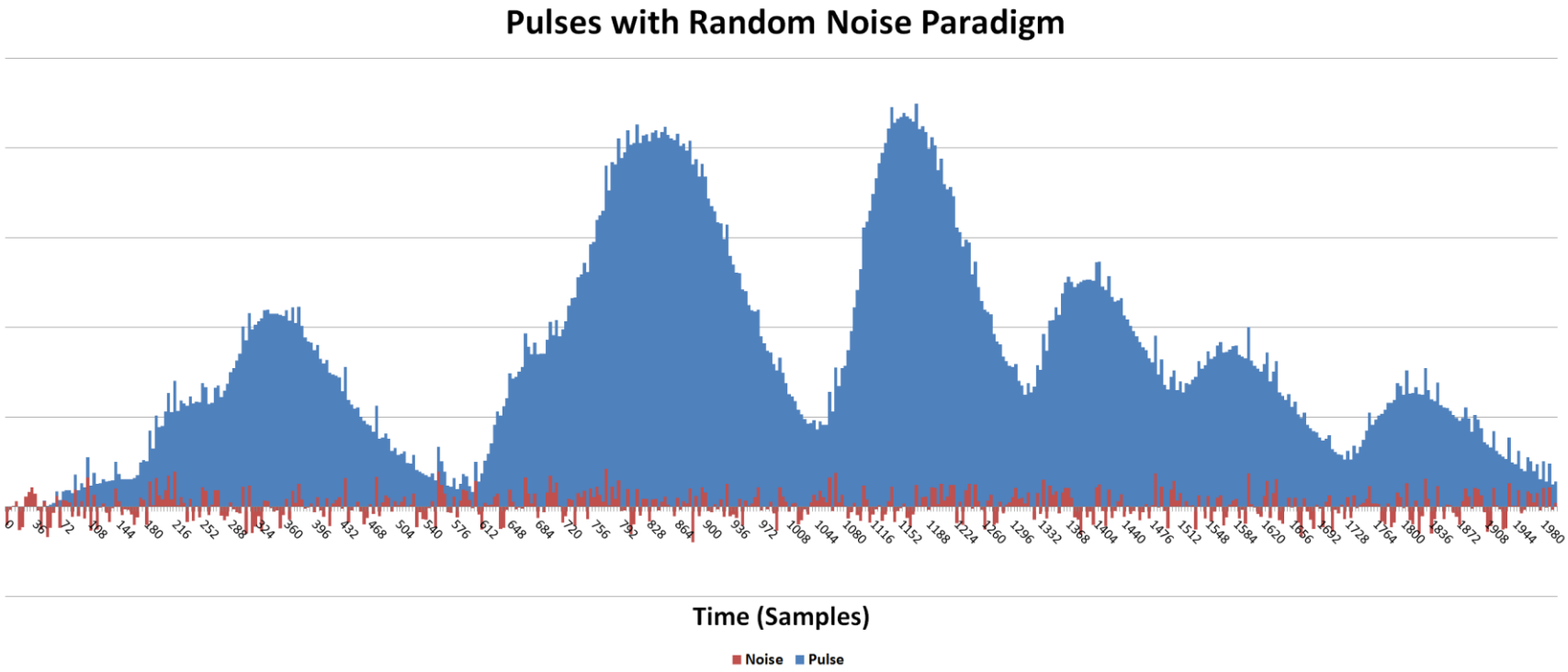
Extra – On Noise

- What should be done?
  - Pedestal runs should be employed to calculate what the actual electronic noise for each channel really looks like.
  - These calculated values can be used by readout to calculate a random electronic noise value every time an ADC buffer cell is written.
  - The noise value will then be added to the cell once regardless of the presence or absence of hits.

- The random Gaußian smearing to the truth hit energy can potentially still be retained.
  - This could be tweaked to represent inefficiencies in photoelectron detection in the detector which could cause deviations in the detected energy versus the true energy.

Extra – On Noise

- This modification would then result in a more accurate representation of a noisy ADC buffer.



Extra – On Noise – Conclusion

- The current representation of electronic noise in the calorimeter Monte Carlo is not accurate.
- It produces “noise” that is not random, but rather correlates to the voltage pulses of input truth hits.
- It also, consequently, is dependent on channel occupancy.
- Random electronic noise should be implemented instead, based on analysis of pedestal run data.