Status of BWEC EMC digitization in PandaRoot

Guang Zhao (*zhaog@ihep.ac.cn*)

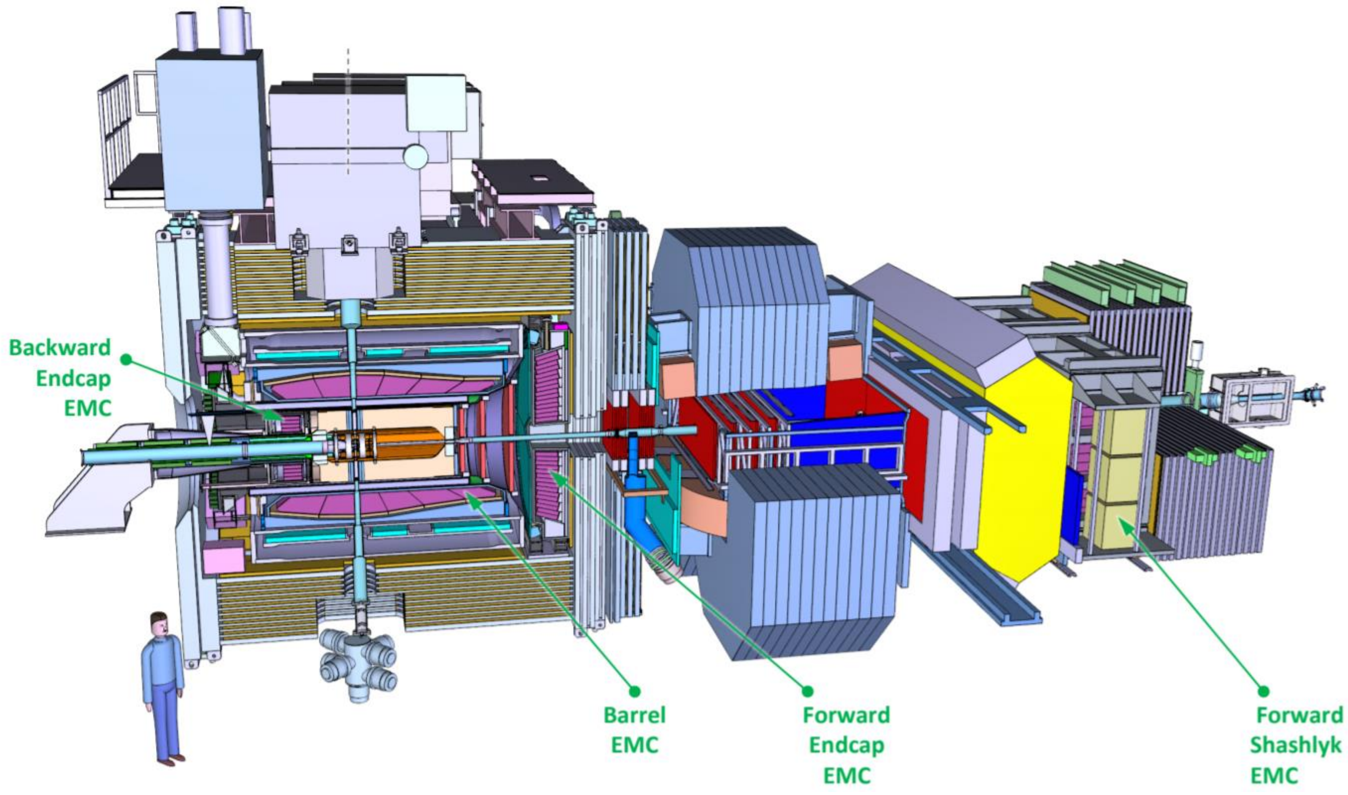
Panda China Meeting

12/12/2019 @ IMP

Outline

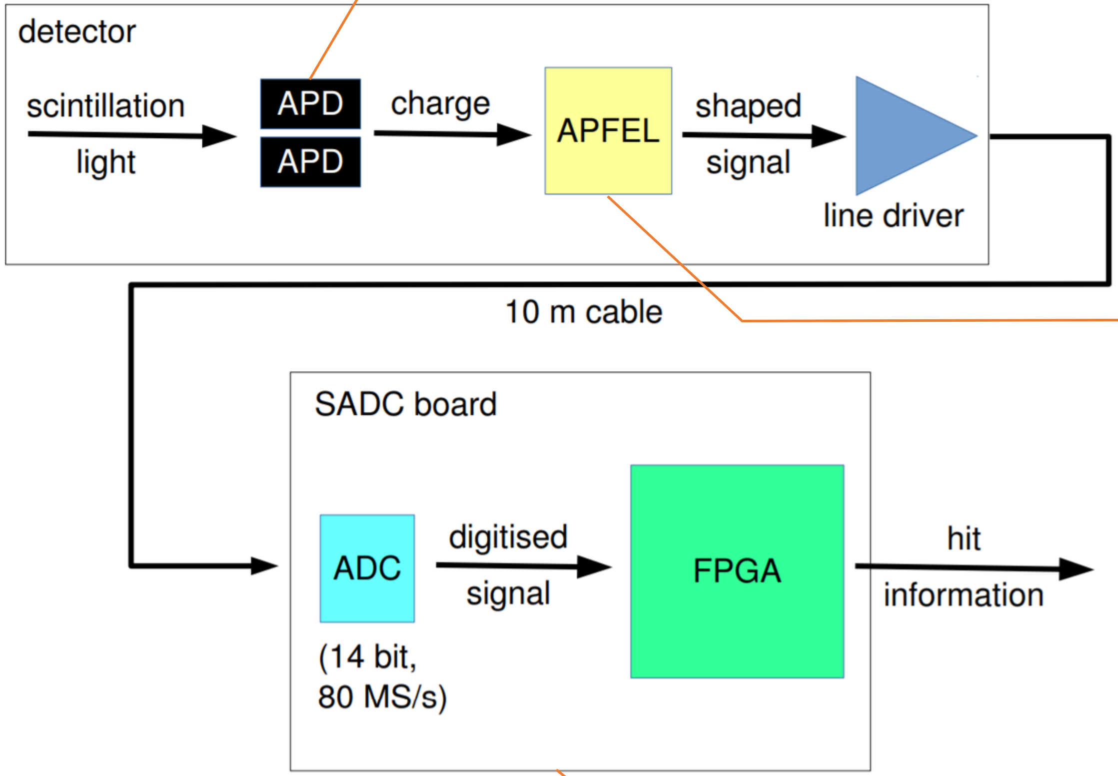
- **Introduction**
- **Digitization implementation**
  - Signal generator (Single APD)
  - Feature extraction (Single APD)
  - Duo APD readout
- **Code development in PandaRoot**
  - Design
  - Performance test
- **Summary and outlook**

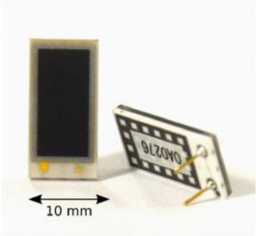
Introduction



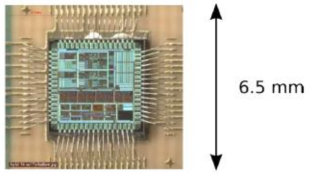
- **The target EMC detector:**
  - ~15500 high quality second-generation PWO II
  - Coverage: 99.8% of 4π
  - Energy resolution: $\leq 1\% \oplus \frac{\leq 2\%}{\sqrt{E/GeV}}$
  - Energy range (photon): 10 MeV – 14.6 GeV
  - Energy threshold (single crystal): 3 MeV
  - RMS noise: 1 MeV
  - Increased light yields at $-25°C$
- **Simulation:**
  - Need detailed simulation such as geometry description and digitization, etc
- **Main work:**
  - Geometry description for barrel EMC (published in dec18)
  - Digitization implementation for the backward endcap EMC in collaboration with Helmholtz-Institut Mainz

Introduction: BWEC readout

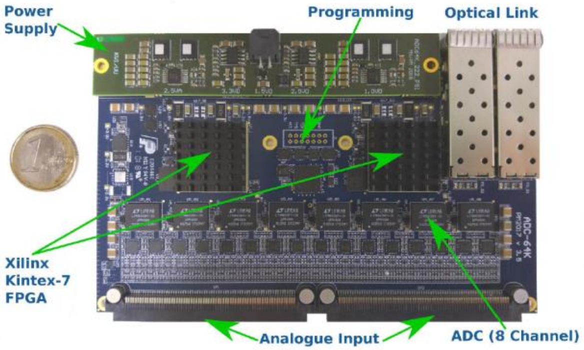




- Large area APD (7x14 mm²)
- Capacitance: 270 pF (full depletion)
- Operated at gain ~200
- Two APD per crystal



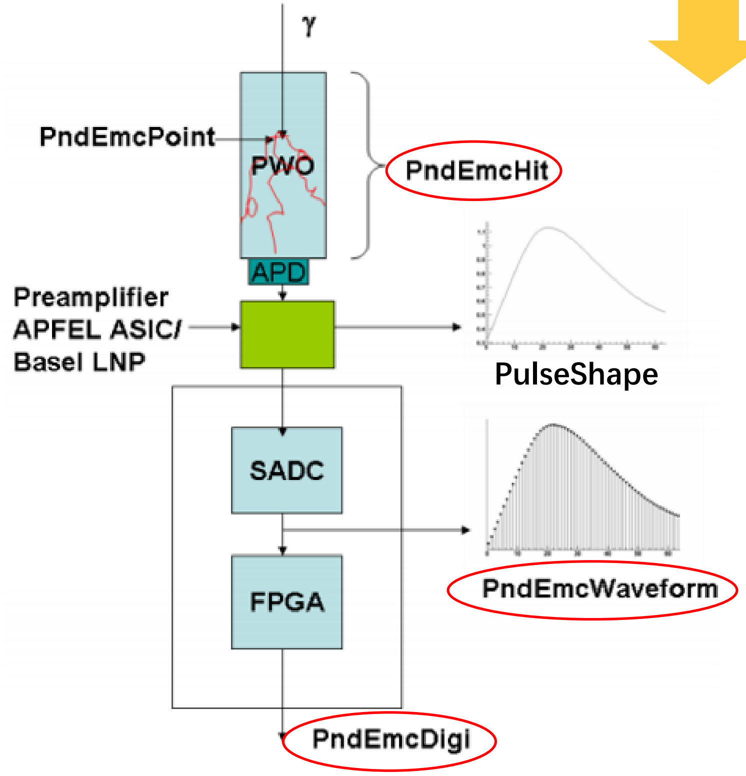
- Charge sensitive preamplifier: APFEL (ASIC for the PANDA Frontend ELectronics)
- Reads out two APD (one crystal)
- Low noise input stage
- Shaper (~1 µs shaping time)
- Two main amplifier (gain 1 and 10)
- 4 output signals (2 APD x 2 gains)
- Low power consumption (~100 mW)



- Developed at University of Uppsala by Pawel Marciniewski
- 64 ADC channel
- 14 bit
- 80 MHz
- Two FPGA's
- Two optical links

Introduction: Digitization process in PandaRoot

Event Generation	Detector Simulation	Digitization	Reconstruction	Rootification
EVGEN	HITS	DIGITS	SHOWERS	TTree/THist



Signal Generator (SG)

- Analog waveforms creation
- Noises generation
- Digitization
- Pile-up waveforms creation

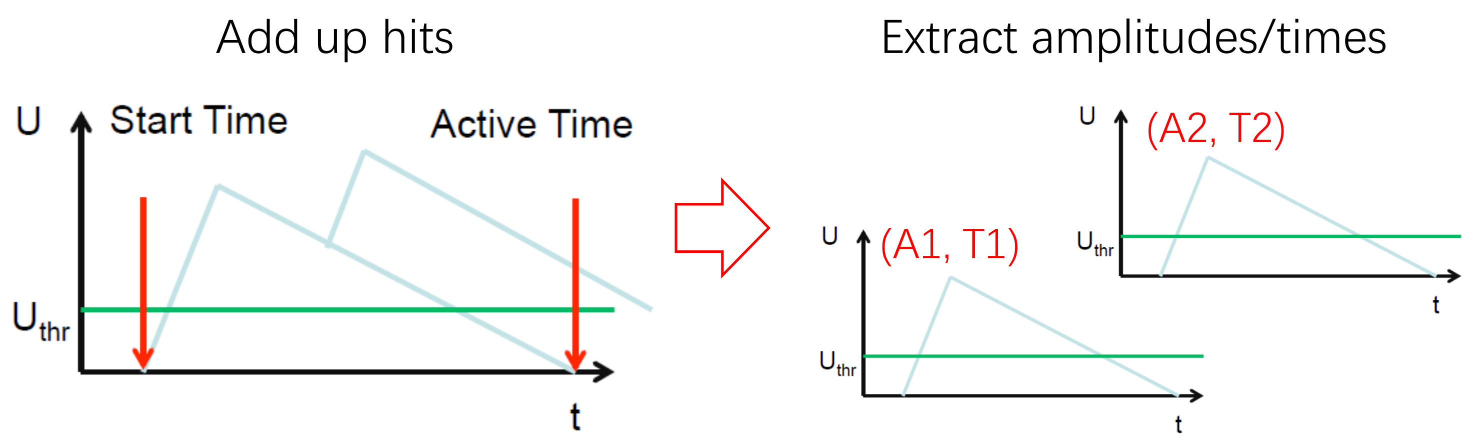
Feature Extraction (FE)

- Hit detection
- Amplitude/time extraction
- Pile-up recovery

Time-based simulation

Introduction: Time-based simulation

- **The digitization should support the time-based simulation**
  - Because
    - Panda readout is trigger-less
    - For barrel/backward endcap, a single crystal rate up to 100 kHz lead to 1% pile-up probability
  - Need to handle
    - Add up multiple hits in SADC as part of signal generator
    - Separate pile-up waveforms as part of feature extraction

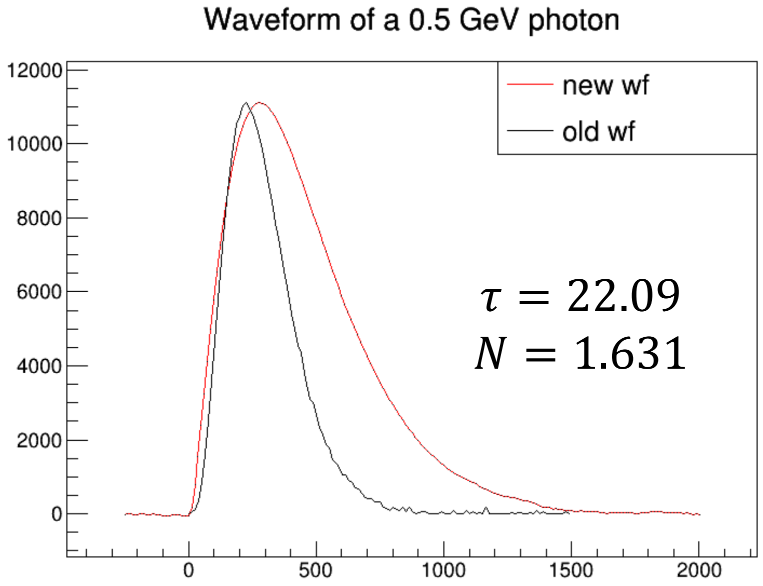


Signal Generator: Pulses

$$f(x) = -A \cdot e^{\frac{-N(x-\delta)}{\tau}} \cdot \left(\frac{x-\delta}{\tau}\right)^N \qquad (2.1)$$

Whereby τ is describing the decay behavior. N has an impact on the rising and decay ratio. δ shifts the pulse in time. A is proportional to the pulse hight H:

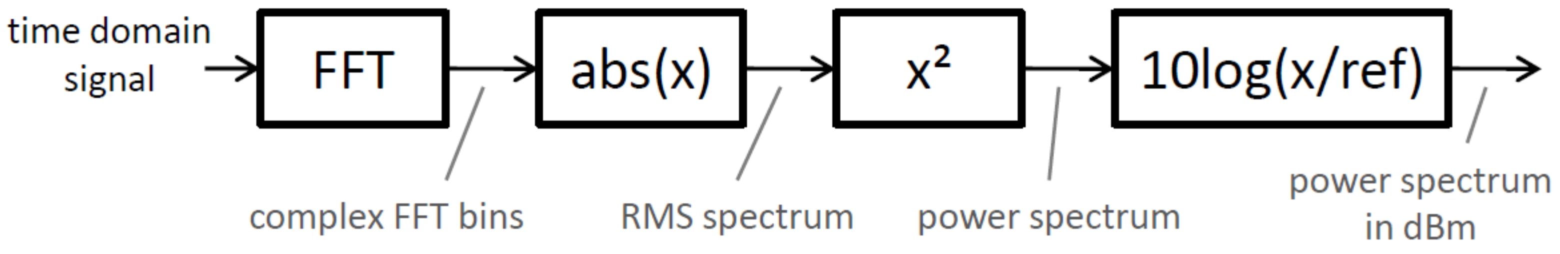
$$A = H \cdot e^N \qquad (2.2)$$

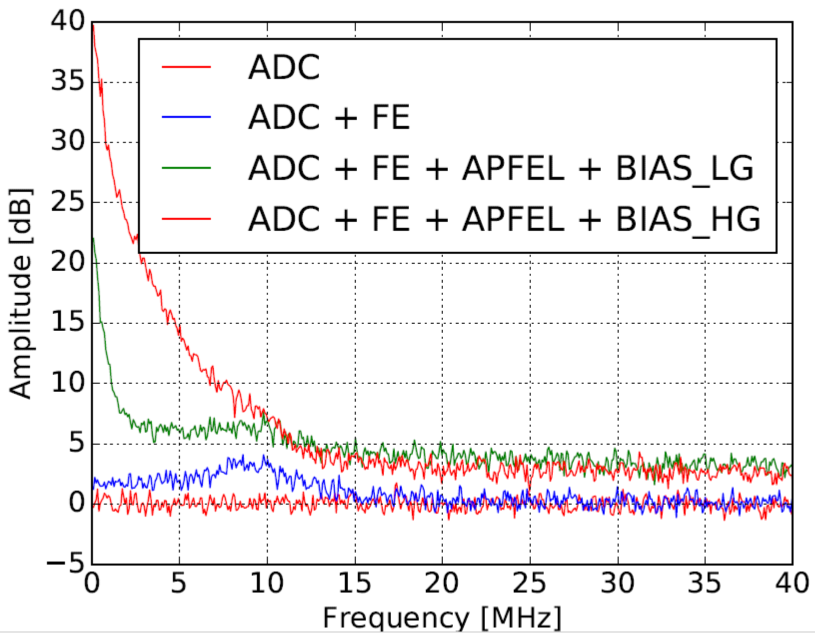


- APD gain = 200
- APFEL amplifier: 2 gains
  - HG/LG = 10.5
- Full pulse width: ~1700 ns
- Rising time: ~300 ns
- APFEL ASIC pulse digitized by the SADC

Signal Generator: Noises

FFT analysis
for the noises



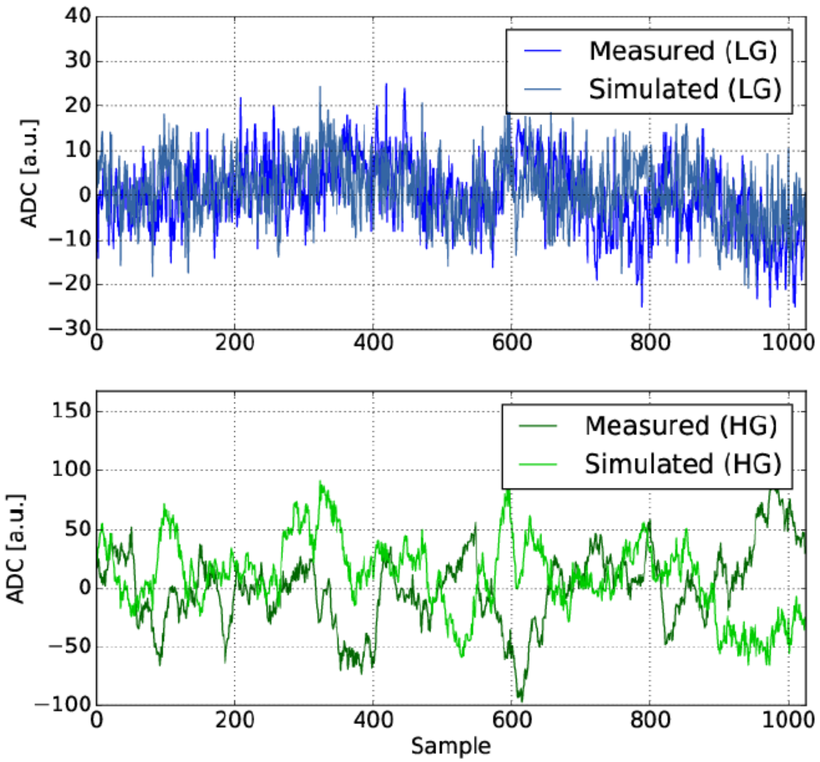


- **<u>Noise components</u>**
  - Biased APD, APFEL preamplifier at low/high gain
  - Open ADC entrance
  - Front-end electronics transmission
- **<u>Noise measurement</u>**
  - FTT analysis of the noises
- **<u>Noise simulation</u>**
  - iFFT of the power spectrum to obtain time-domain noises

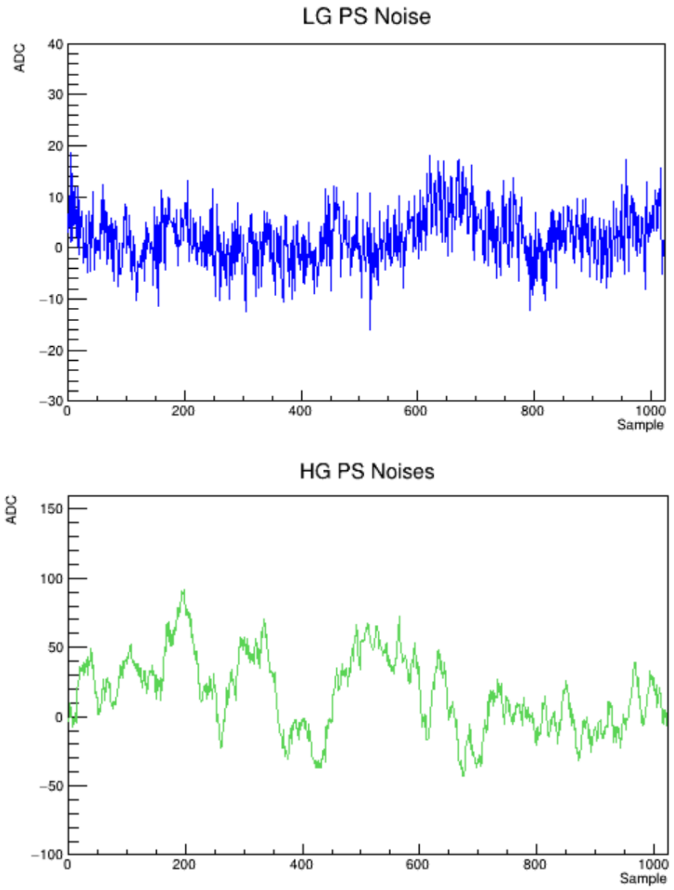
Signal Generator: Noise (II)

Biased APD, APFEL preamplifier for low/high gain (correlated)

Measurement

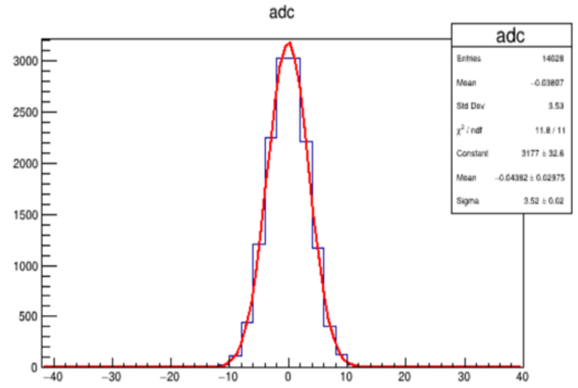


PandaRoot Sim

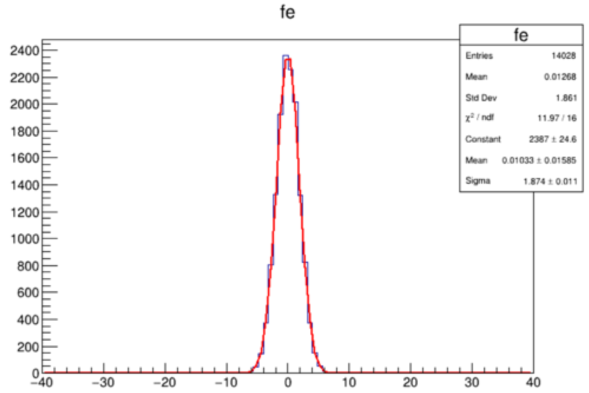


✓ Good agreement between simulation and measurement

ADC & FE Transmission



Measured value: 3.5
Simulated value: 3.52 +/- 0.02



Measured value: 1.89
Simulated value: 1.874 +/- 0.011

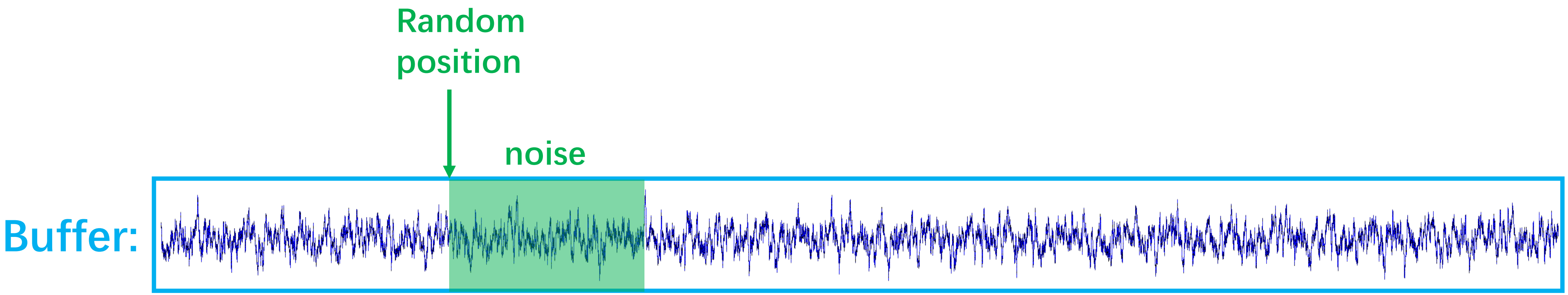
A problem: IFFT is slow

- **Time for 100 events (Core i5 in my laptop)**
  - Old algorithm: 3.88 sec
  - New algorithm: 42.29 sec

- **Time for 100 events without iFFT**
  - New algorithm: 3.49 sec ➔ Comparable to the time of the old algorithm

- **Reason**
  - Perform 511 times of iFFT for each noise generation because the power spectrum has 511 frequency bins
  - Need to optimize the noise modeling method

Solution: An approximated noise model



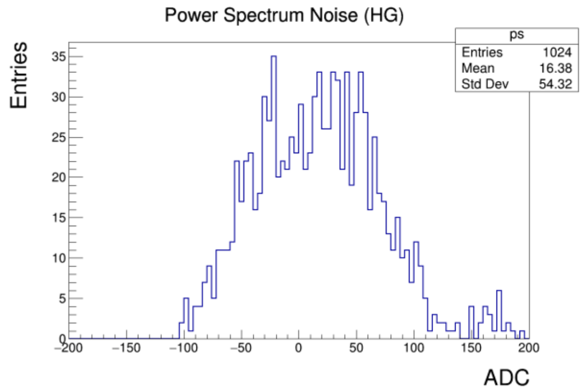
- ✓ **Noise modeling**
  - ✓ Pre-generate a big noise buffer
  - ✓ Pick up the noise of a waveform from a random position in the buffer.

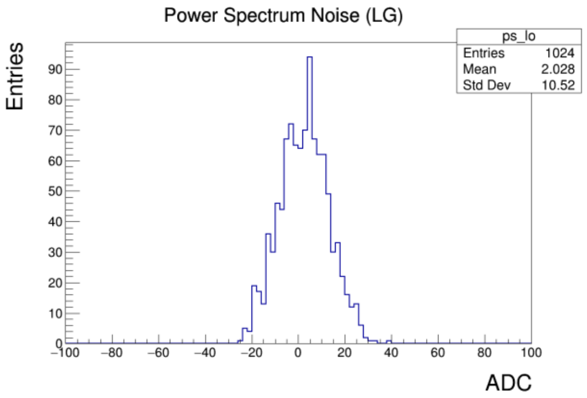
- ✓ **Pros:** Much faster
- ✓ **Cons:** Loose some randomness

# of waveforms	CPU Time (sec)	
	Full iFFT	Reduced iFFT
100	1.218	1.106
200	1.405	1.079
500	2.413	1.047
1000	4.147	1.105
5000	18.096	1.193

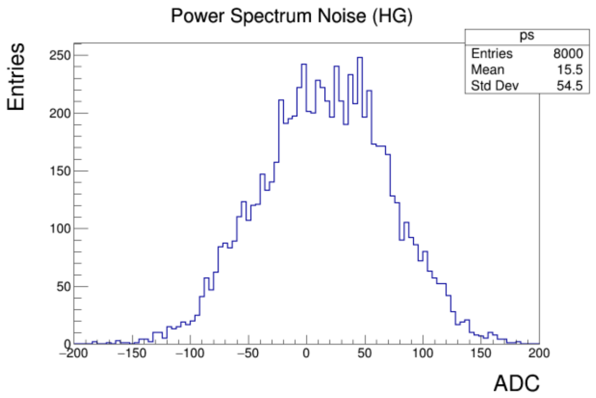
Noise comparisons

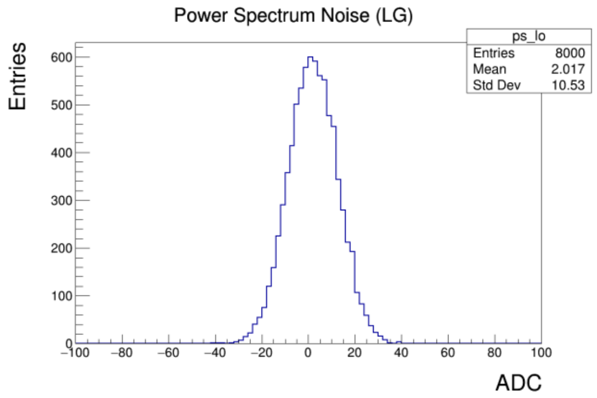
From Oliver



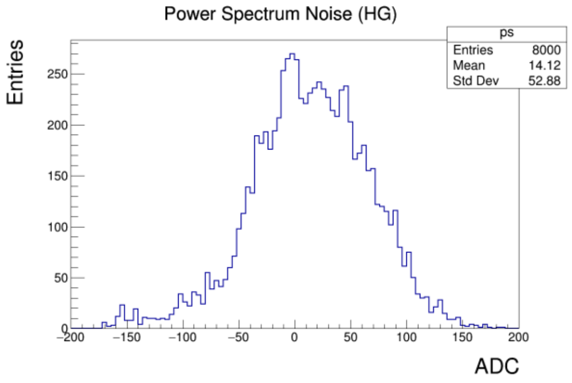


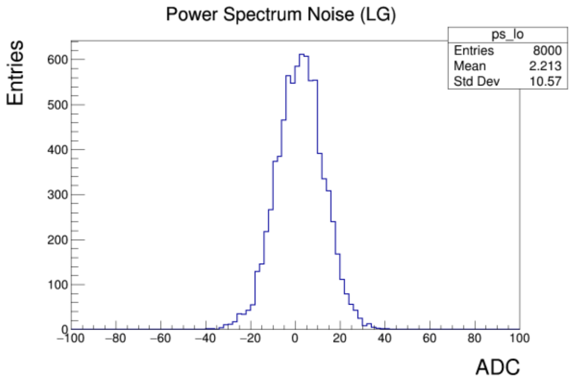
w/ full iFFT





w/ reduced iFFT



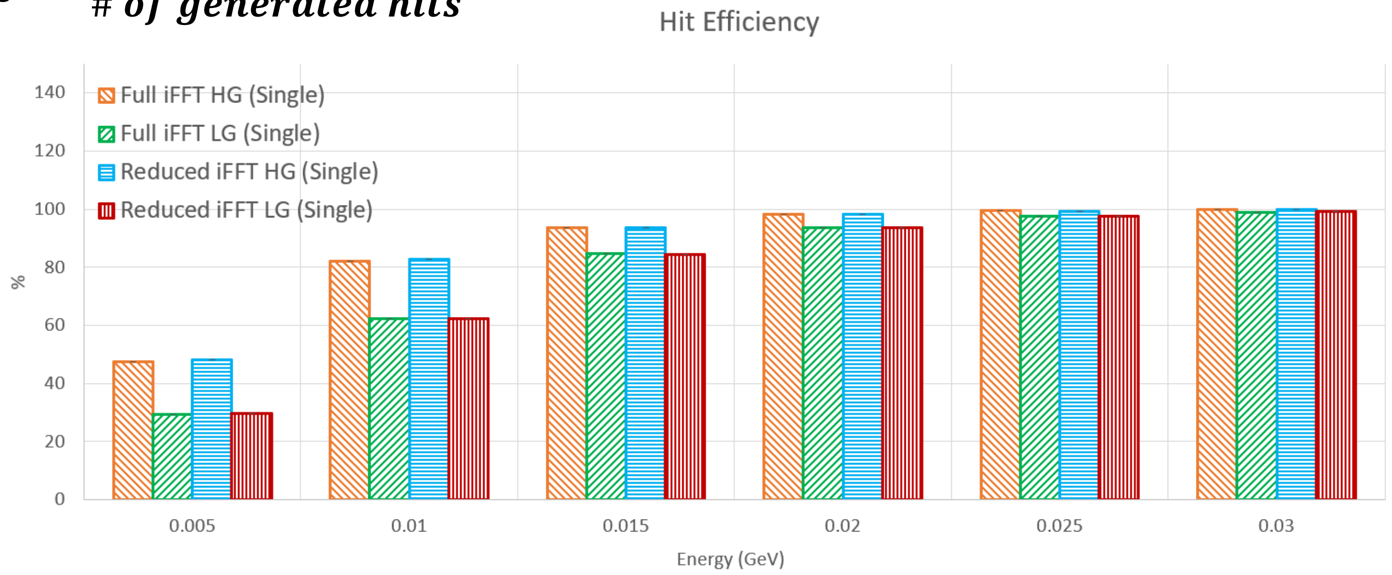


No obvious discrepancies on mean/rms distributions for the reduced iFFT

Noise rate/hit efficiency check (single APD)

- **Noise rate = # of noises / sec [Hz]**
  - Full iFFT: 160.5 +/- 1.3 kHz (HG), 332.4 +/- 2.6 kHz (LG)
  - Reduced iFFT: 166.0 +/- 1.3 kHz (HG), 332.9 +/- 2.6 kHz (LG)

- **Hit efficiency =** $\dfrac{\#\ \boldsymbol{of\ detected\ hits}}{\#\ \boldsymbol{of\ generated\ hits}}$



Similar noise rates and hit efficiencies for the reduced iFFT

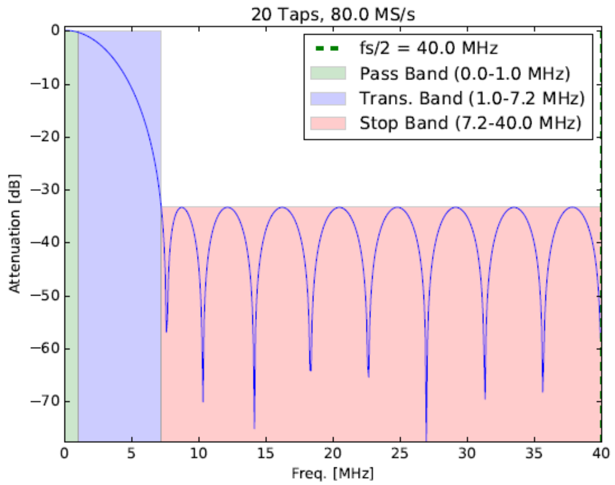
Feature Extraction: FIR filtering

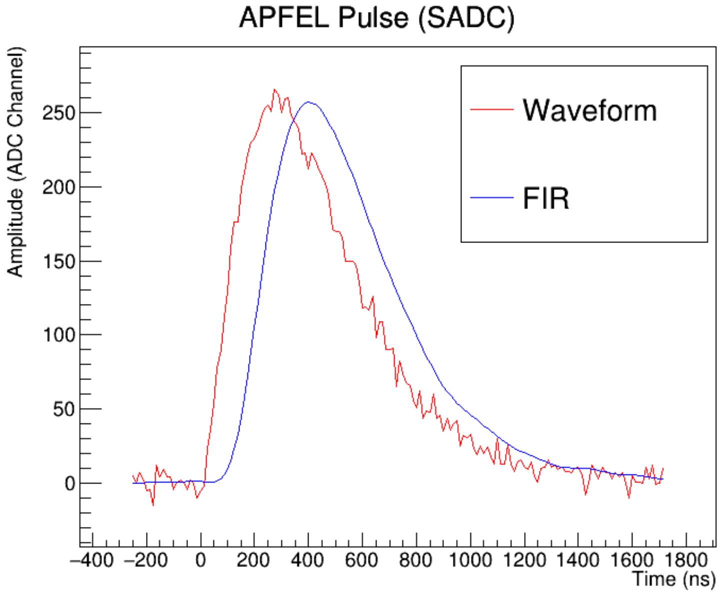
PndEmcPSATmaxAnalyser

- fEnergyList : vector<Double_t>
- fTimeList : vector<Double_t>

+ Process(const PndEmcWaveform*) : Int_t
+ GetHit(Int_t, Double_t&, Double_t&)
- fir(Int_t*, Int_t) : Double_t*
- hit_det(Int_t, Int_t, Int_t) : Double_t

- Transfer function suppressed HF noise (low pass)
- Z transformation of impulse response
- $H(z) = \sum_{n=0}^{N} h(n) \cdot z^{-n}$
  - $h(n)$: Filter Koeffizienten
  - $z = e^{i\omega T}$
- Each output value is weighted sum of most recent input values
- $\mathbf{out}[n] = h_0\mathbf{in}[n] + h_1\mathbf{in}[n-1] + \ldots + h_N\mathbf{in}[n-N]$





Low pass filter to smooth the waveform (20 coefficients, ~10 cycle clocks latency)

Feature extraction: Time extraction

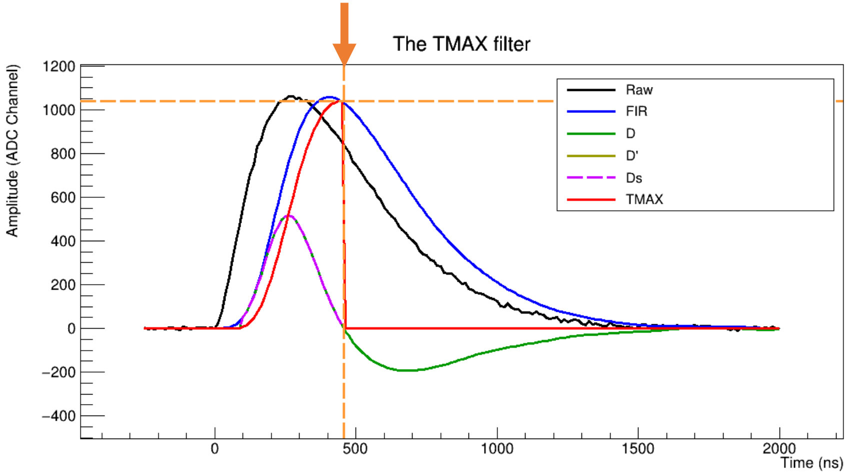
PndEmcPSATmaxAnalyser
- fEnergyList : vector<Double_t> - fTimeList : vector<Double_t>
+ Process(const PndEmcWaveform*) : Int_t + GetHit(Int_t, Double_t&, Double_t&) - fir(Int_t*, Int_t) : Double_t* - hit_det(Int_t, Int_t, Int_t) : Double_t

The TMAX filter:

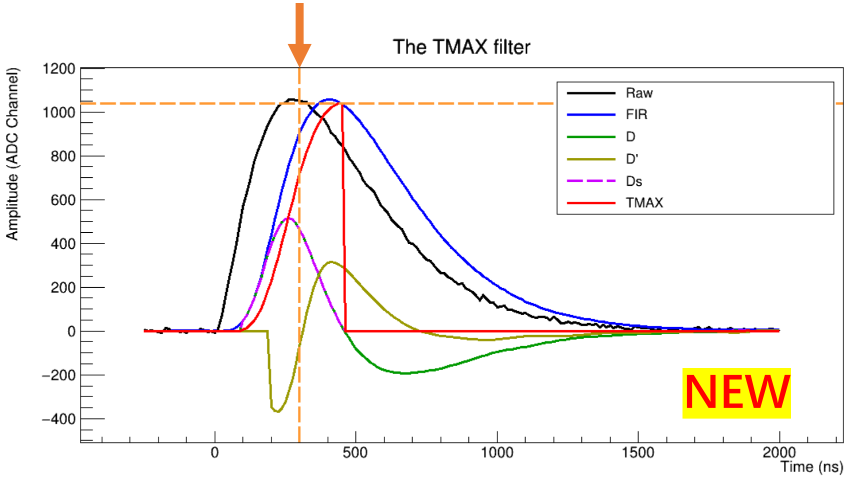
Deviation: D[i] = T[i + r] – T[i]

Second Deviation: D’[i] = D[i + r] – D[i]

Falling edge cancelling: $D_s[i] = D[i] + D^*_{inv}[i]$



First deviation



Second deviation

✓ **Times are extracted at the zero transition of the second deviation of the FIR, which are closer to the times of the raw waveforms’ peak**

Feature Extraction: Amplitude extraction

PndEmcPSATmaxAnalyser

- fEnergyList : vector<Double_t>
- fTimeList : vector<Double_t>

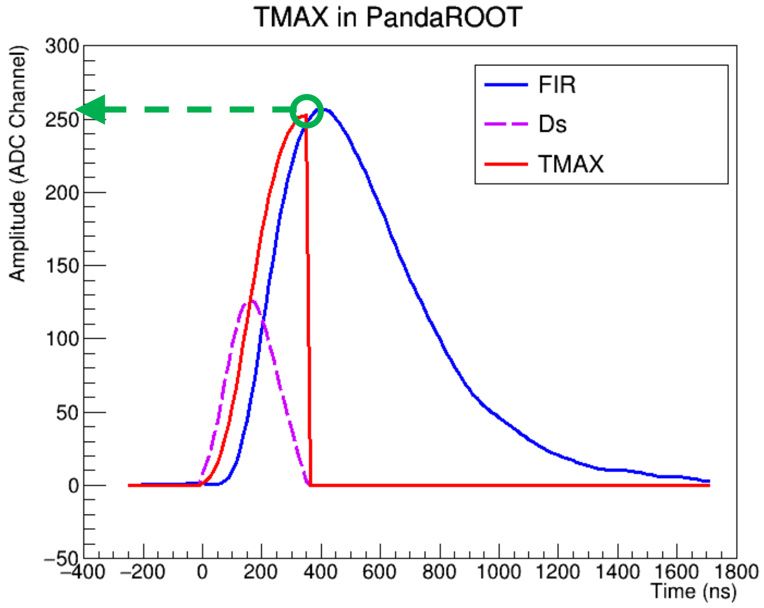
+ Process(const PndEmcWaveform*) : Int_t
+ GetHit(Int_t, Double_t&, Double_t&)
- fir(Int_t*, Int_t) : Double_t*
- hit_det(Int_t, Int_t, Int_t) : Double_t

The TMAX filter:

$$D_s[i] \mapsto \begin{cases} F_{TMAX}[i] = F_{TMAX}[i-1] + \frac{D_s[i]}{r} & : D_s[i] < 0 \\ F_{TMAX}[i] = 0 & : D_s[i] = 0 \end{cases}$$

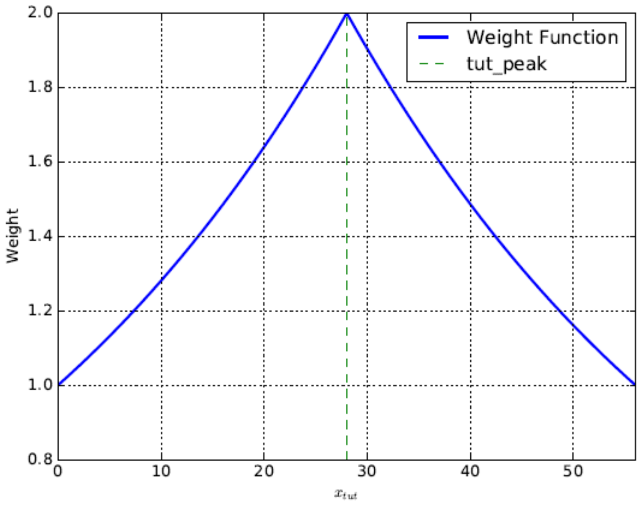
By integrating D_s, we can obtain the amplitude of the rising edge of the pulse

Extract amplitude at the TMAX peak



Feature Extraction: Hit detection

```
PndEmcPSATmaxAnalyser
- fEnergyList : vector<Double_t>
- fTimeList : vector<Double_t>
+ Process(const PndEmcWaveform*) : Int_t
+ GetHit(Int_t, Double_t&, Double_t&)
- fir(Int_t*, Int_t) : Double_t*
- hit_det(Int_t, Int_t, Int_t) : Double_t
```



- True hits should be detected from noises
- A function to weight the hit detection with the time under threshold is derived
- The weight function is convoluted with the extraction function (TMAX), and a hit is detected when its value passes a threshold

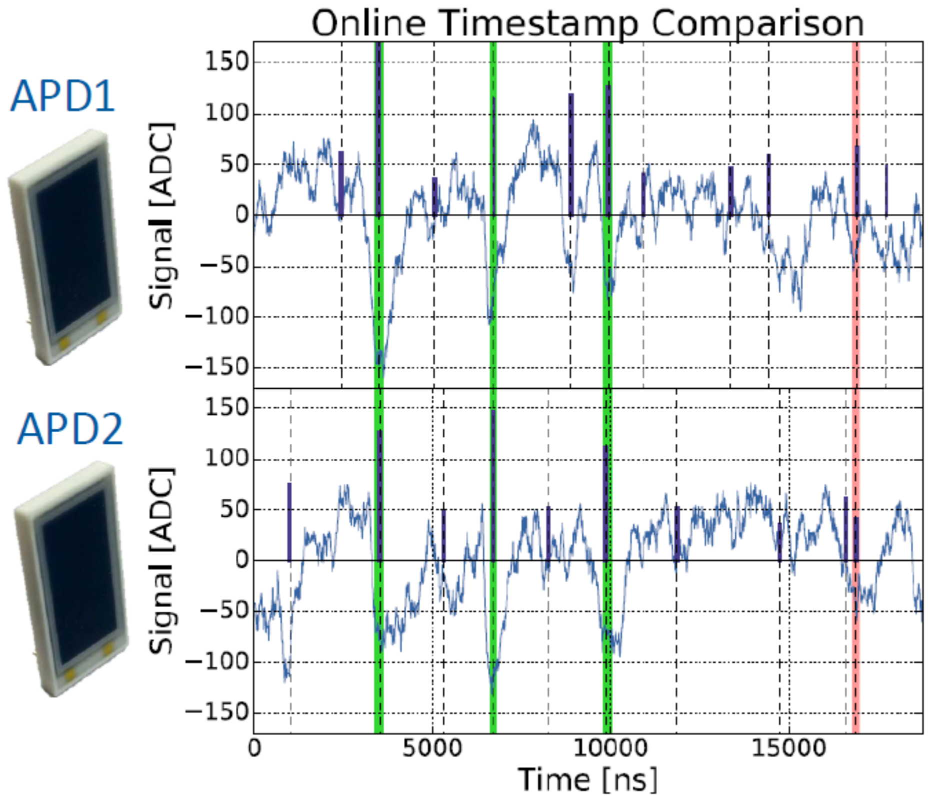
$$x_{tut} \mapsto \begin{cases} e^{a \cdot x_{tut}} & : x_{tut} < tut_{peak} \\ hit_{val} \cdot e^{-a \cdot (x_{tut} - tut_{peak})} & : x_{tut} \geq tut_{peak} \end{cases}$$

$$(tut_{peak}, hit_{val}) = (28{,}2)$$

Hit threshold: ~1.35 MeV

The duo-APD output

Scintillation light



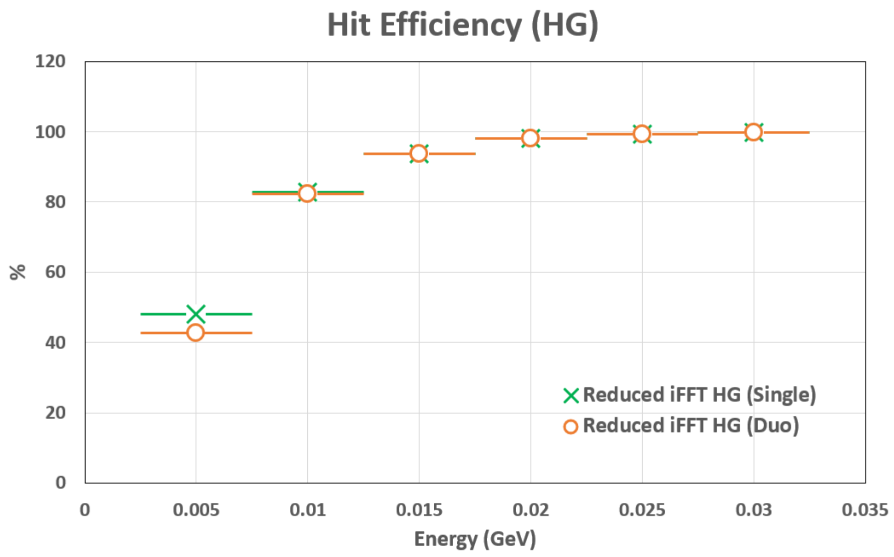
- Noise hit rate reduction by online timestamp comparison
- Noises are uncorrelated
- Hit merge
  - Times are matched
    - $|T1 - T2| < dT$
  - Average amplitudes/times
    - $A_{OUT} = (A1 + A2)/2$
    - $T_{OUT} = (T1 + T2)/2$

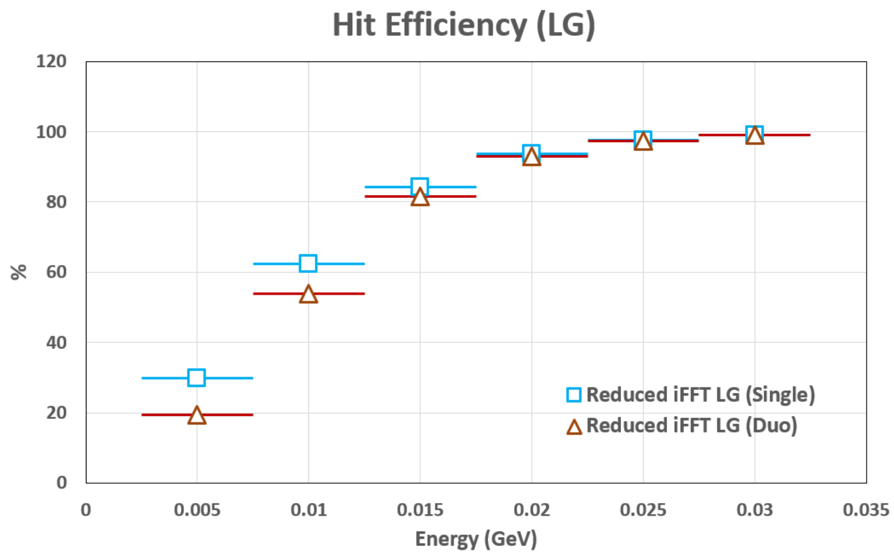
Noise rate/hit efficiency check (duo APD)

- Noise rate = # of noises / sec [Hz]
  - Single APD: 166.0 +/- 1.3 kHz (HG), 332.9 +/- 2.6 kHz (LG)
  - Duo APD: 22.1 +/- 0.2 kHz (HG), 84.4 +/- 0.7 kHz (LG)

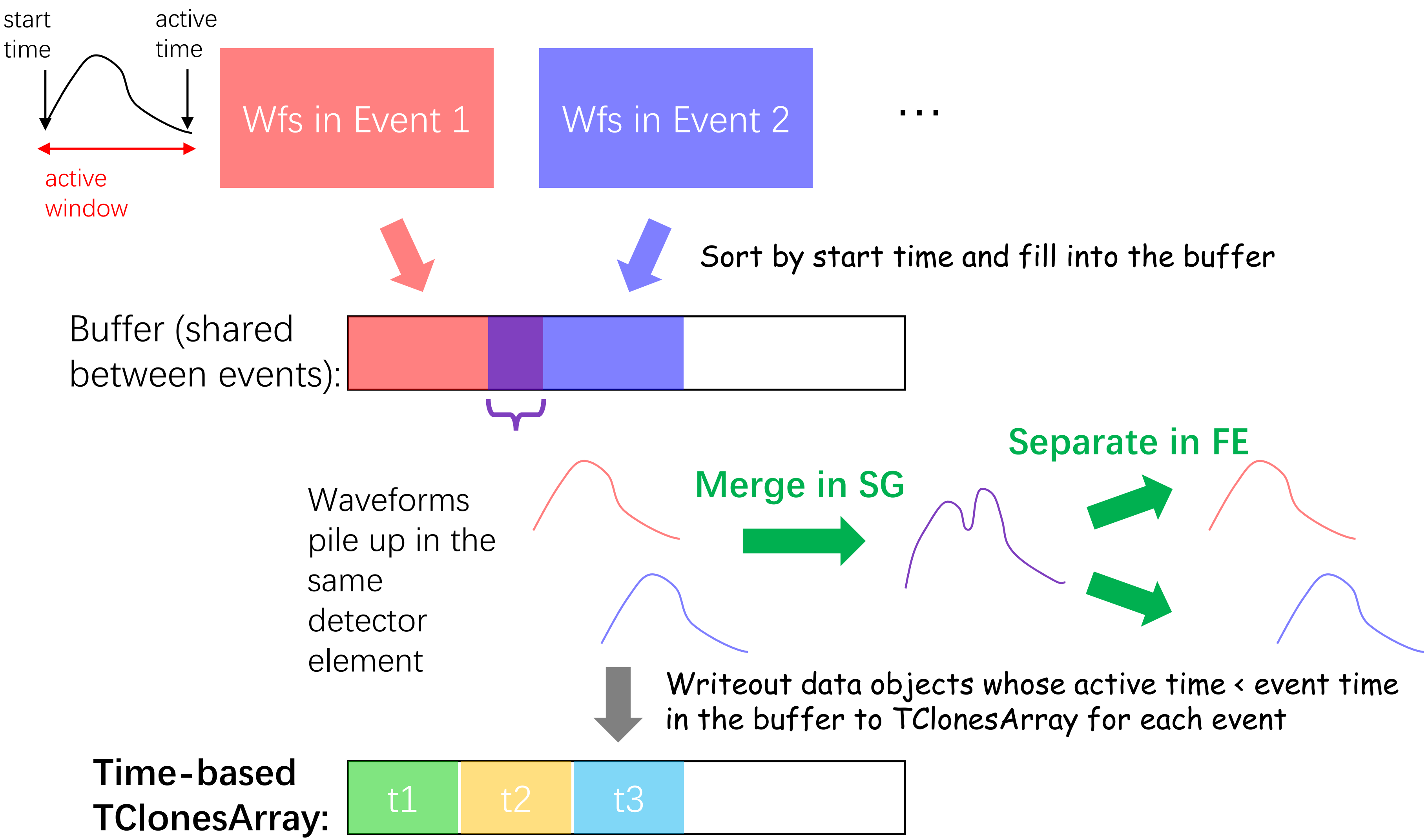
The duo-APD output can effectively suppress the noise rate

- Hit efficiency = $\frac{\#\ of\ detected\ hits}{\#\ of\ generated\ hits}$

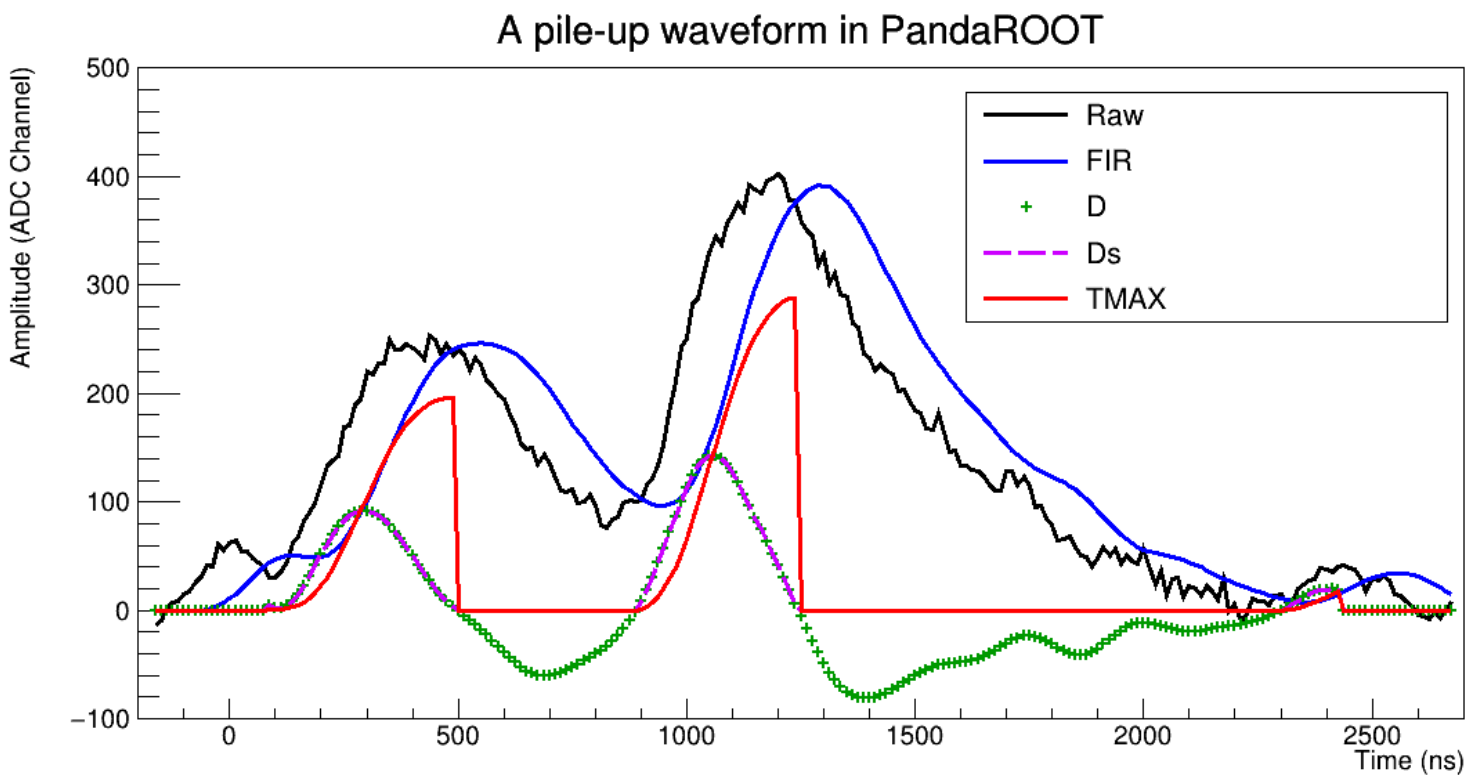




Time-based simulation in PandaRoot



Time-based simulation: Pile-up waveforms



- ✓ We are able to produce the pile-up waveforms, and are able to separate them
- ✓ For instance, two digis are detected from this pile-up waveform
- ✓ The amplitude of the secondary waveforms need to be corrected, because the amplitude of the rising edge does not start from 0

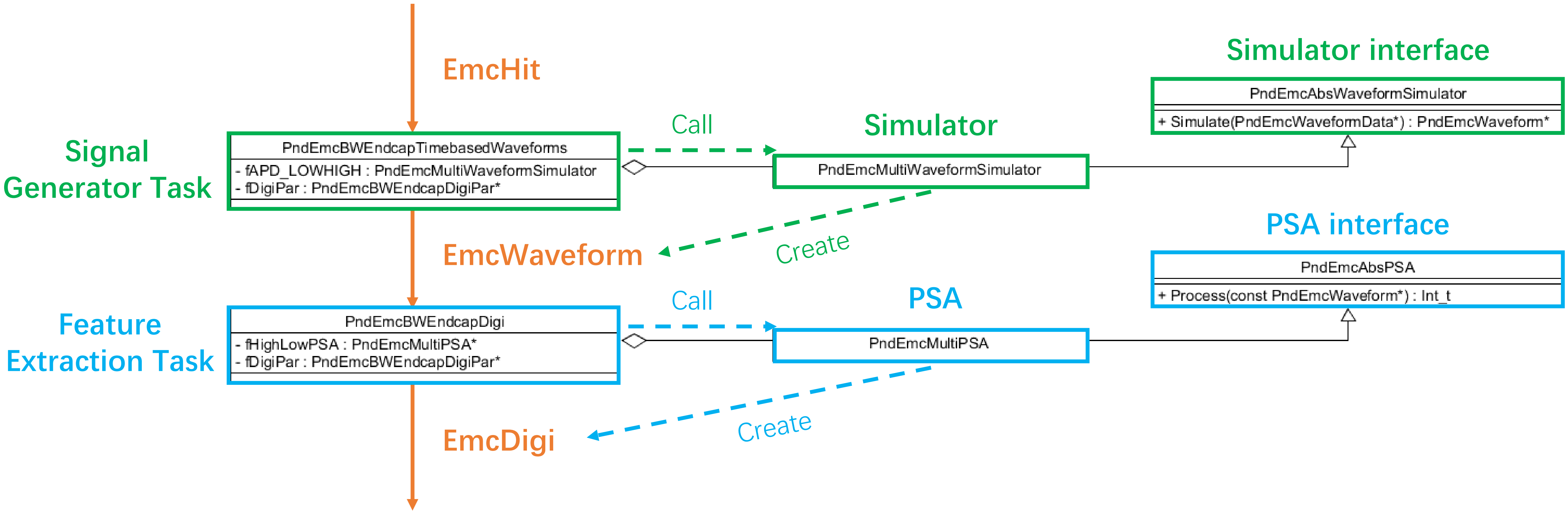
Code development in PandaRoot

- **Major requirements for the digitization package**
  - Main function: signal generator + feature extraction
  - Support time-based simulation
- **Compare the two existing digitization algorithms in PandaROOT**

	The default package	The forward package
Avalability	All EMC	Only FW Endcap
Time-based simulation	Support	Support
Multi waveform	No	Yes
Scalability	OK	Easier

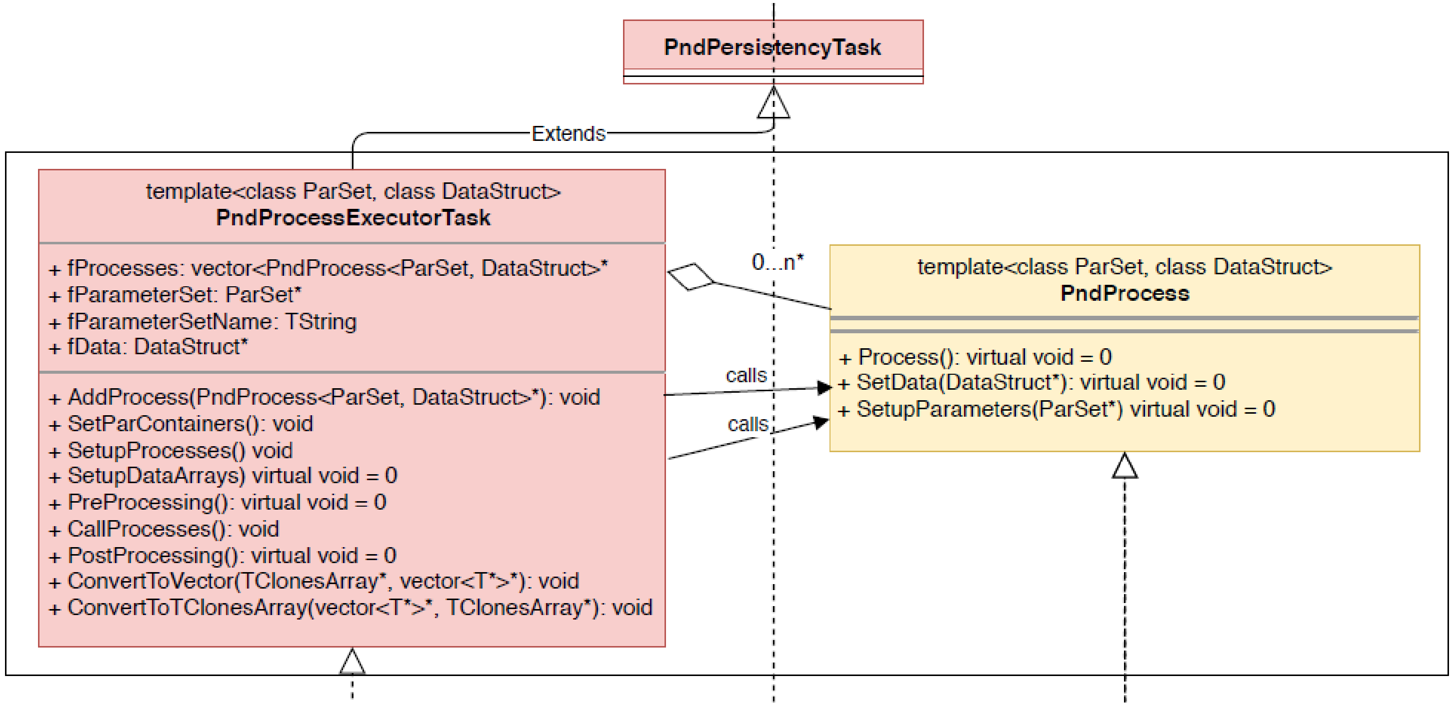
- **Will develop a new BWEC package based on the forward package**
  - Separate the tasks and the algorithms
  - Use common interfaces for the algorithm classes
  - Easy to scale to other sub-detectors

Code structure



- Two tasks for signal generator and feature extraction respectively
- Simulator for creating waveforms from hits
- Pulse Shape Analyzer (PSA) for creating digis from waveforms

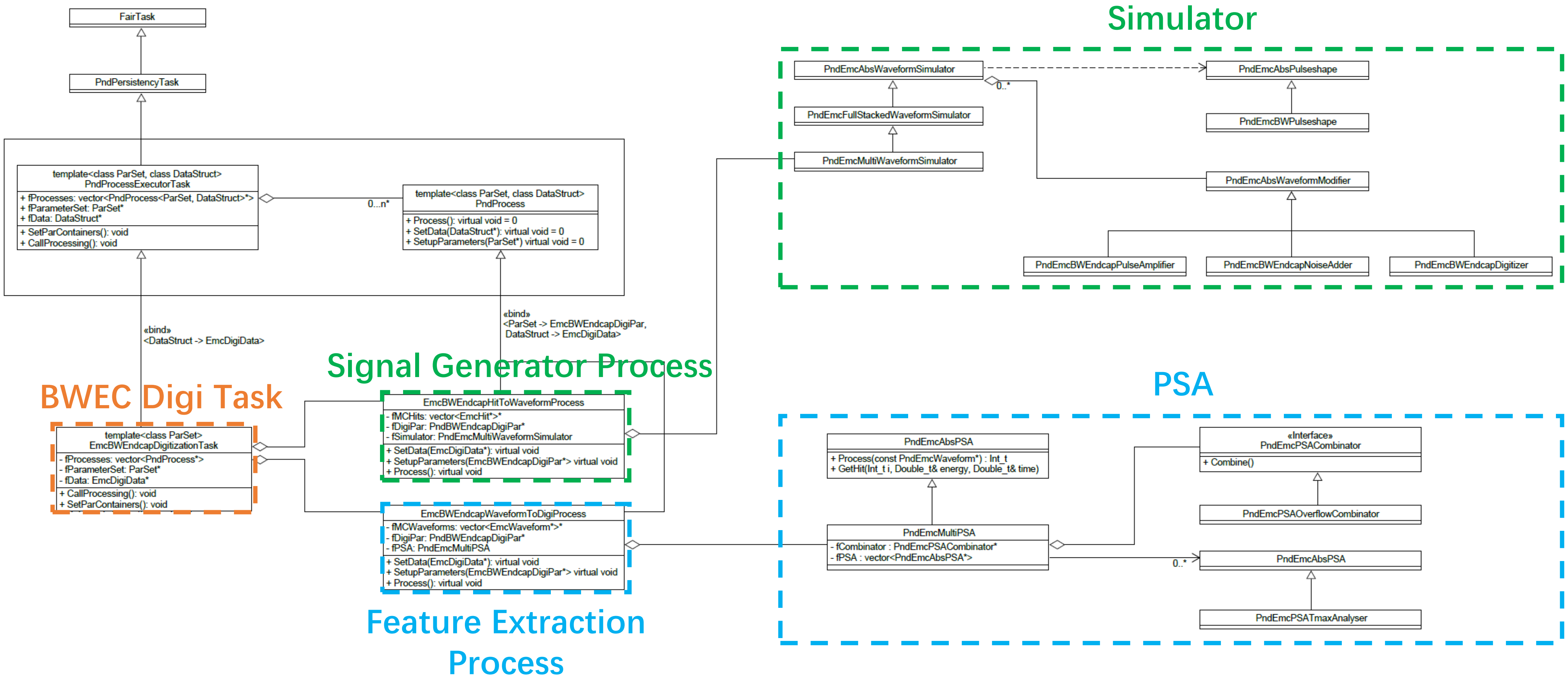
The new “ExecutorTask” framework



- A new design by Ben who is optimizing the whole EMC code
- Tasks that do similar jobs are encapsulated into a single “PndProcessExecutorTask” (e.g. there should be a single EmcDigiTask)
- The encapsulated tasks now become several “PndProcess”s

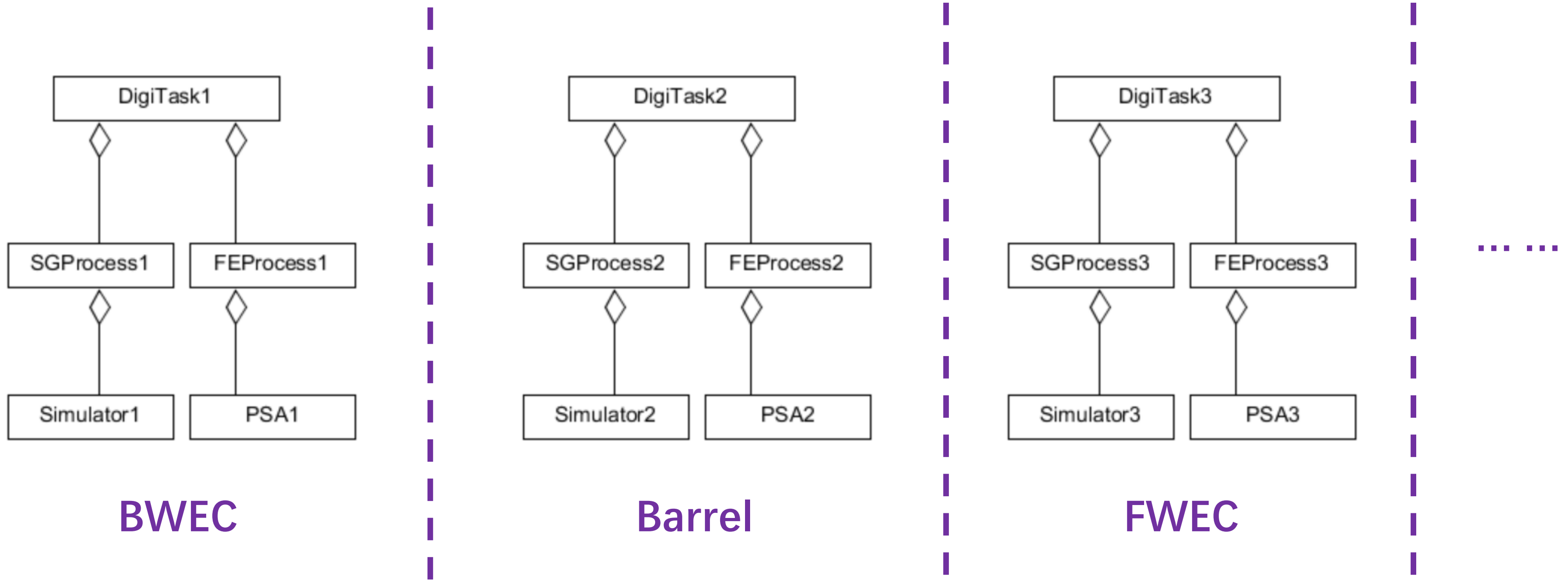
Code structure for the “new” framework

The “Simulator” and “PSA” can be re-used



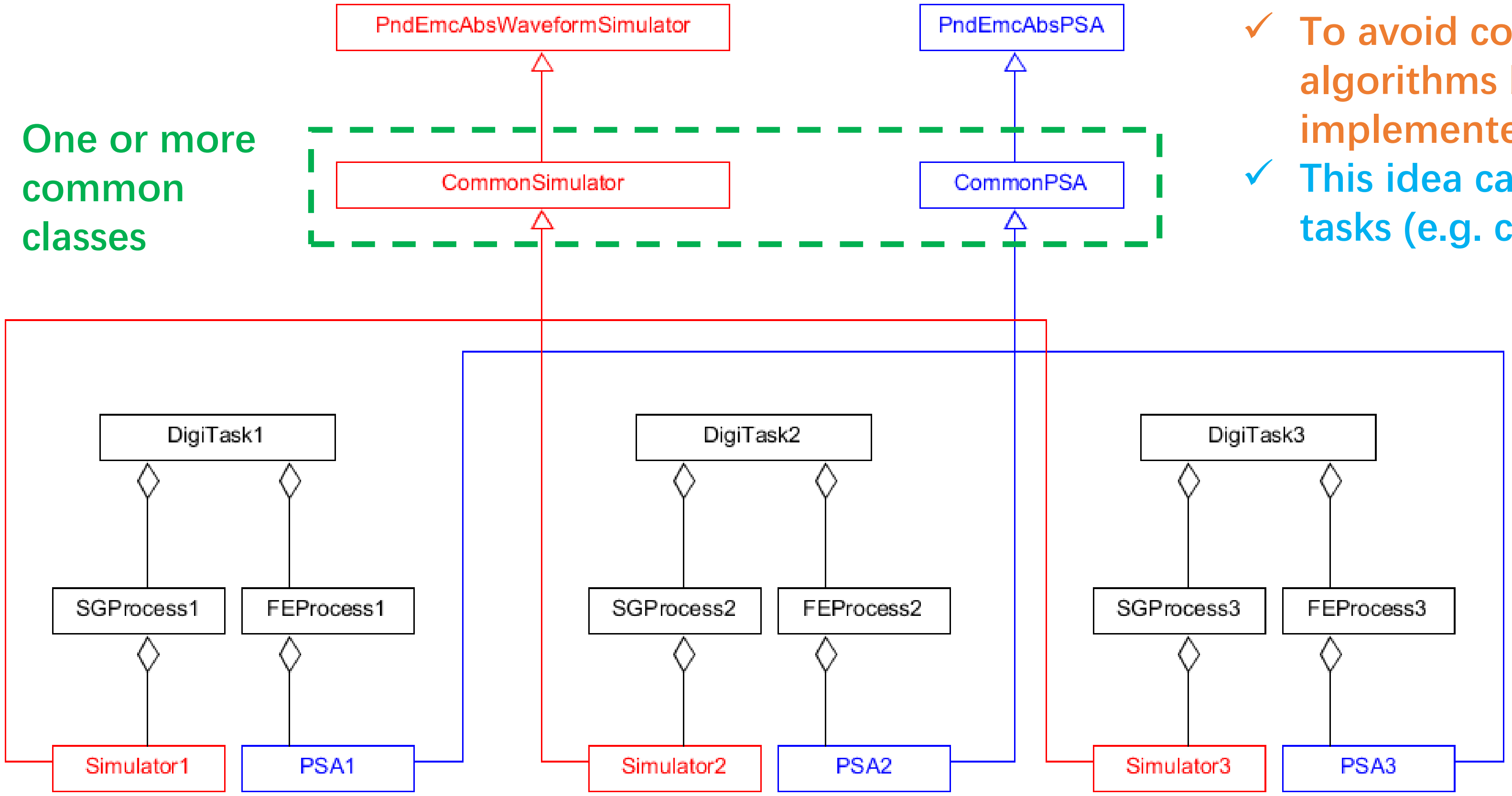
Code structure for different EMC modules

- ✓ In the new framework, for EMC, simulations are separated for EMC modules. Therefore, like BWEC, we should have several more "DigiTasks"
- ✓ For each task, it handles its own data object
- ✓ The simplified diagram are shown below (The "class names" and "structures" are also simplified for demonstration)



Common functionality among modules

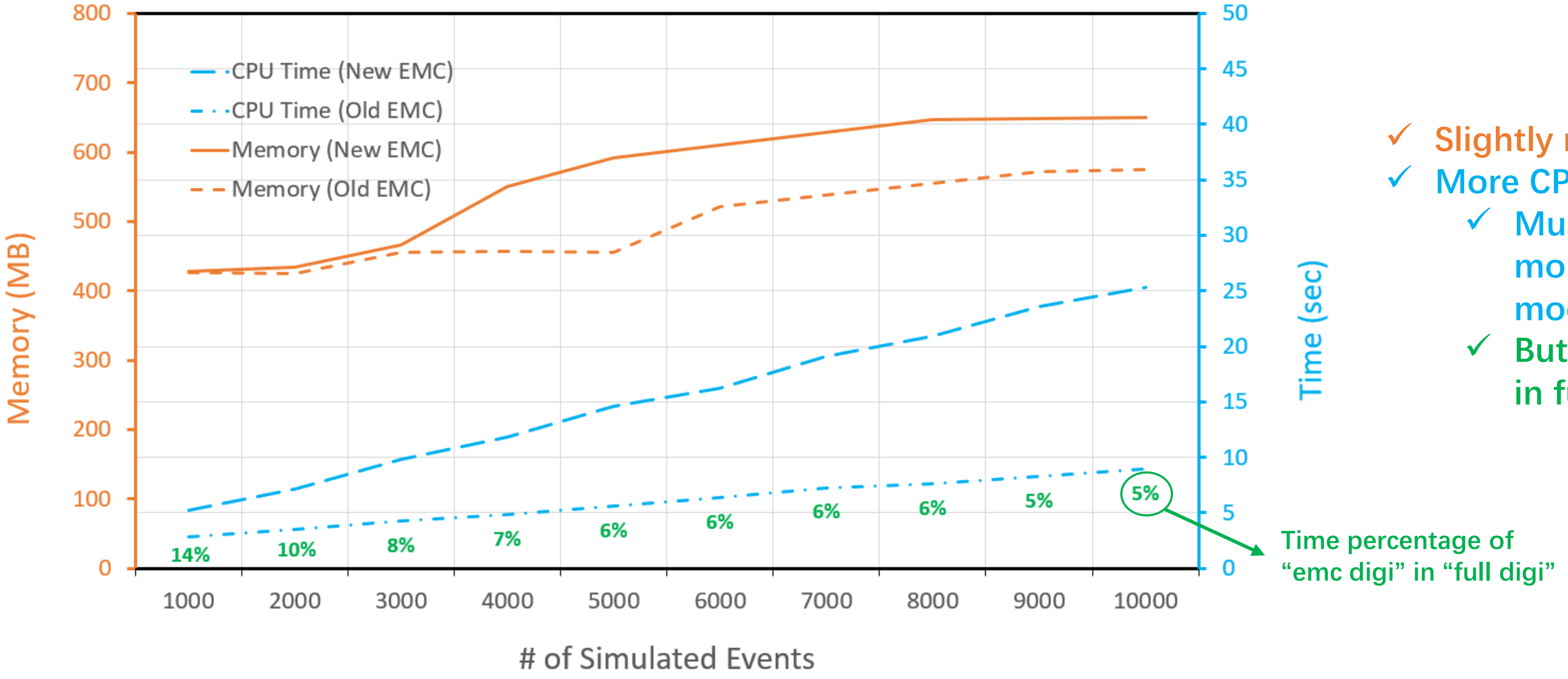
One or more common classes



- ✓ To avoid code duplication, common algorithms between modules are implemented in one or more base class
- ✓ This idea can also be applied to other tasks (e.g. clustering task)

… …

Performance test



- ✓ Slightly more memory
- ✓ More CPU time
  - ✓ Must pay for the much more complicated noise model ☹
  - ✓ But only a small fraction in full digitization ☺

Code in git repository

https://git.panda.gsi.de/zhaog/PandaRoot/tree/emc_digi_bwec

Guang Zhao > PandaRoot > **Repository**

emc_digi_bwec | PandaRoot / + | History | Find file | Web IDE

implemention of 2-apd data processing method
Guang Zhao authored 1 week ago fb806cfc

Name	Last commit	Last update
PndMCMatchNewLinks	Remove Warnings & Adjust FairLogger usage	1 year ago
analysis	Missing #include<array> added	11 months ago
config	bugfix/pndsim tree	1 year ago
detectors	implemention of 2-apd data processing method	1 week ago
eventdisplay	Updated stt geometry	1 year ago
external	Always compile the old version of Vc.	5 months ago
fastsim	fixing some paths for fsim & QA	1 year ago
field	Remove Warnings & Adjust FairLogger usage	1 year ago
gconfig	Fixing test fails when running with new root	2 years ago
genfit	Some include/lib dirs added by Radek	2 years ago
genfit2	genfit2 patch for includes / for upcominf FairRoot versio...	2 years ago
genfit2-remote	Made a hard-copy of genfit2, moved remote genfit2 to ...	2 years ago

- ✓ Latest code is uploaded to my development branch
- ✓ Will perform more tests on the time-based simulation before check in to the main repository

Summary

- **Digitization implementation**
  - Signal generator: provide digitized waveforms with realistic noises
  - Feature extraction: extract digi information from the waveforms using the TMAX filter
  - Duo APD signal processing to suppress noises
  - Capable of the time-based simulation

- **Code development in PandaRoot**
  - A new package for the bwec digitization
  - OO design
    - Task/algorithm separation: Easy to migrate to the new framework
    - Algorithms w/ well defined interfaces: Easy to scale to other sub-detectors
  - Performance test: acceptable speed and memory use

Plan

- **BWEC EMC digitization code**
  - Almost ready to check in to the main repository
  - Will perform more tests on the time-based simulation before checking in

- **EMC digitization unification**
  - Have proposed a code design
  - Need to contact hardware persons for different sub-detectors (not sure how deep we can participate)

EMC module	Readout system	Contact person
FWEC inner	VPTT + Basel preamplifier + shaper + SADC	Viktor/Myroslav
FWEC outer	APD*2 + Basel preamplifier + shaper + SADC	
Barrel/BWEC	APD*2 + APFEL ASIC (shaper) + SADC	Oliver(bwec)/Hans(barrel)
Shashlyk	PMT + SADC	Markus/Per-Erik