Backward endcap EMC digitization in PandaRoot

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

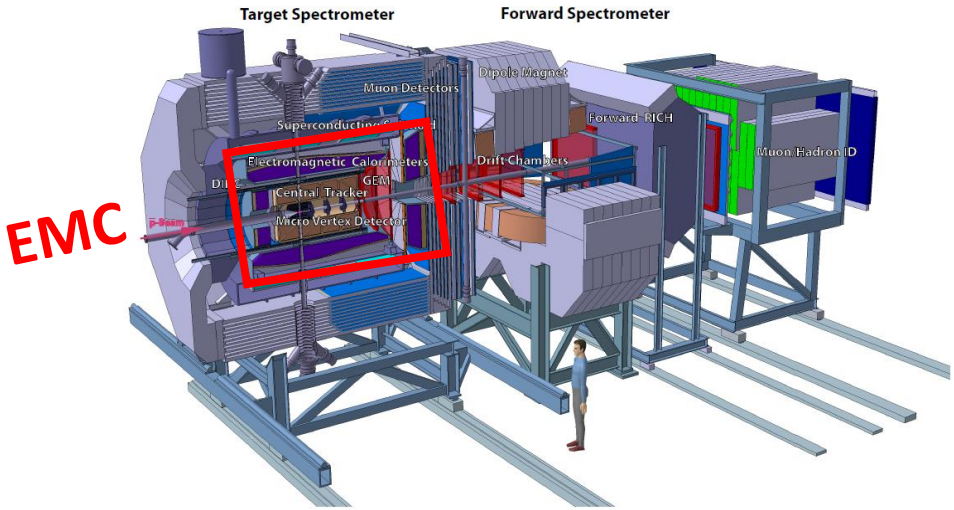
Institute of High Energy Physics (Beijing)

2019/06/03 @ USTC

Outline

- **Introduction**
  - Summary for the barrel geometry
  - Overview of the digitization software
  - The strategy of the PandaRoot implementation
- **Implementation of the backward endcap EMC digitization**
  - Signal generator
  - Feature extraction
- **Summary and outlook**

PANDA软件工作



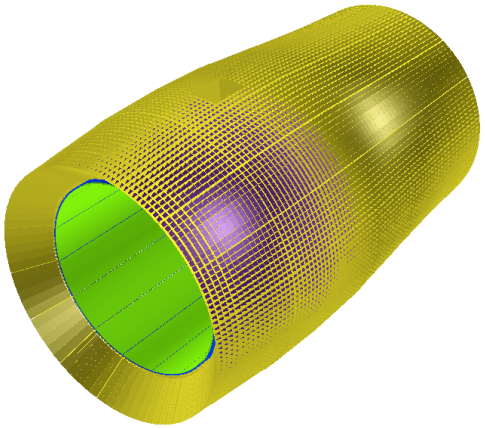
吉森大学实验室组装的桶部EMC

- **PANDA量能器：**
  - 是PANDA的主要靶向探测器
  - 单个晶体能够探测3 MeV到>10 GeV的电磁簇射
  - 工作温度：-25度（光子产额为室温的~4倍）
- **难点：**
  - 探测MeV级别的光子对探测器的精确模拟，包括几何与物质的描述、电子学模拟、簇射的重建刻度提出了极高的要求
- **主要工作：**
  - 继续完成对桶部量能器几何的构建
  - 与电子学人员进行合作（Mainz U.），开展后端盖电子学模拟的工作

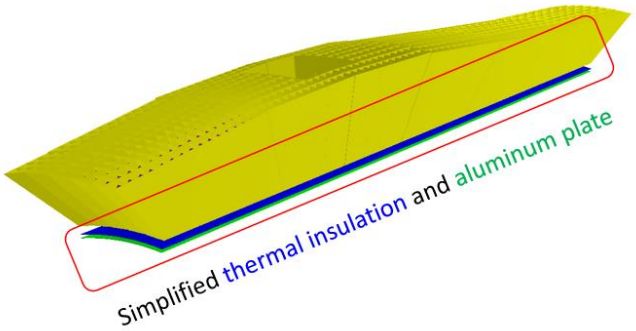
Summary for the barrel geometry

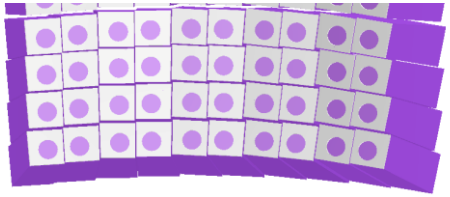
- **Have updated**
  - the whole barrel crystals/dead materials
  - the simplified front end geometries and materials

- **Have released with the PandaRoot 18dec**

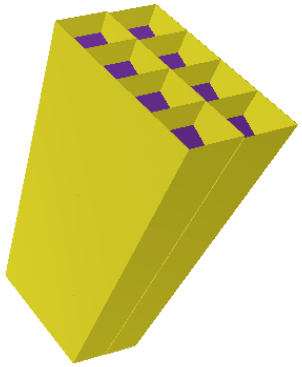


whole barrel with crystals and dead materials





wrappings and front inserts



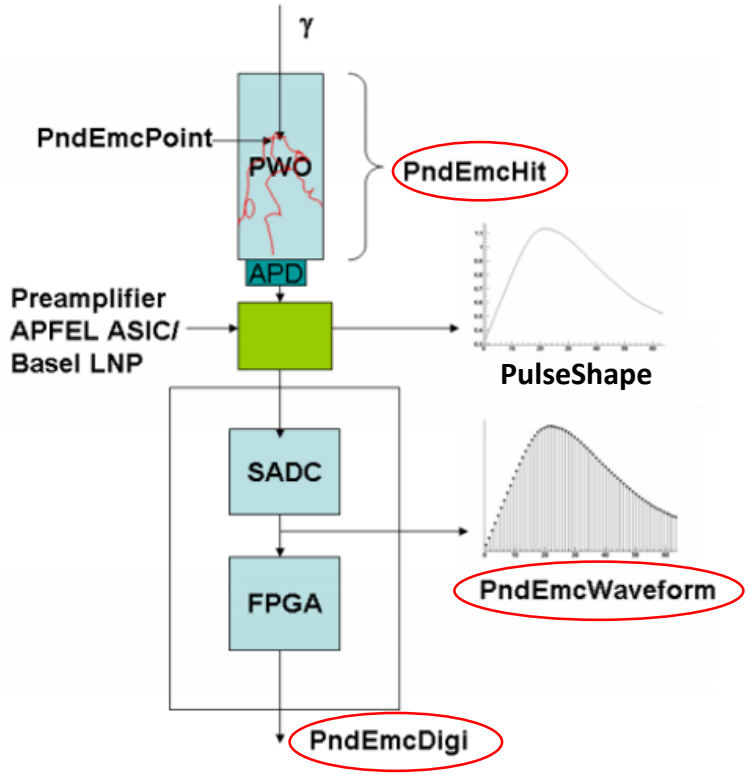
carbon alveoles

The backward endcap digitization

- **The backward endcap code by Oliver Noll**
  - Signal generator: Generate digitized waveforms from hits
    - Two-gain digitized waveform
      - High gain/low gain ratio = 10.5
    - New noise models from beam test
      - Power spectrum noise (amplifier response)
      - ADC and transmission noise (ADC/FE electronics)
  - Feature extraction: Extract time/amplitude from the waveforms, and recover pile-ups
    - TMAX filter: FIR filtering, amplitude extraction, hit detection, time extraction

- **Work to do**
  - The code is independent from PandaRoot
  - Part of the code is written in Python
  - Need to integrate to the PandaRoot framework

Digitization process in PandaRoot



Signal generator

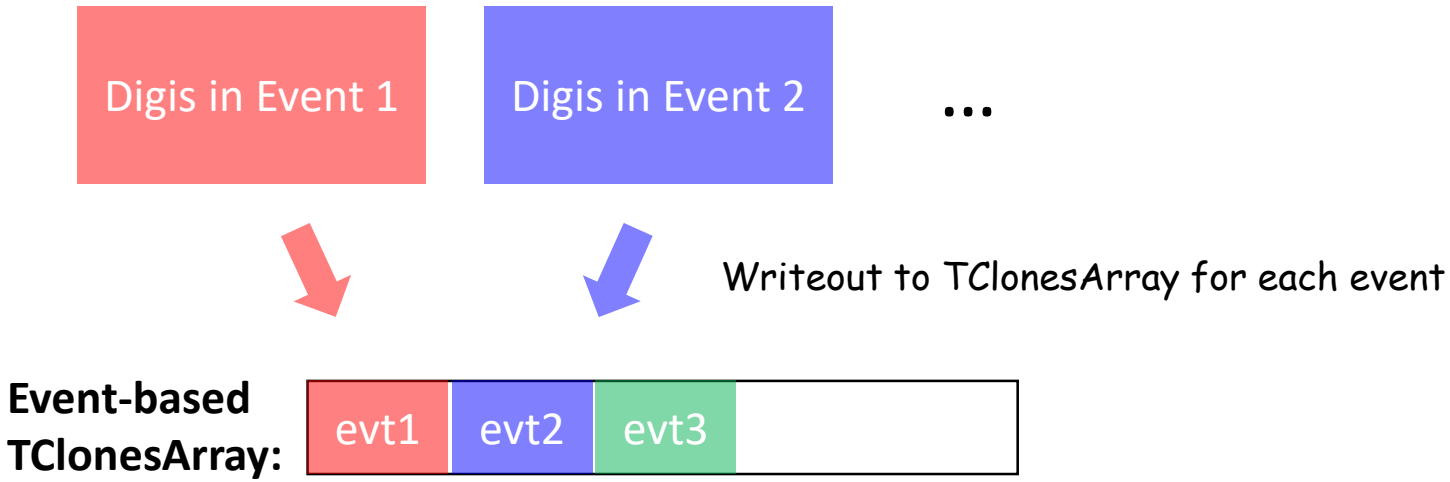
- **Create analog waveform**
- **Add electronics noise and digitize the waveform**

Feature extraction

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

Need to support both event-based and time-based simulation

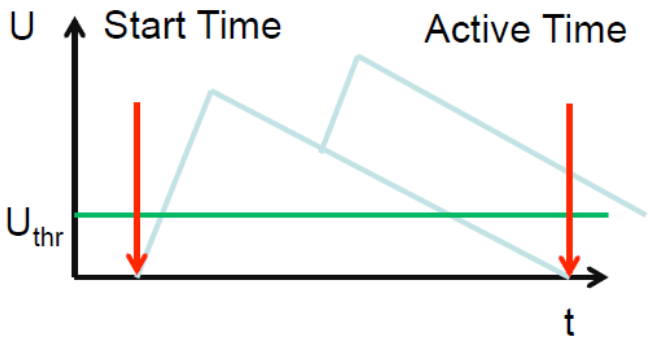
The event-based simulation



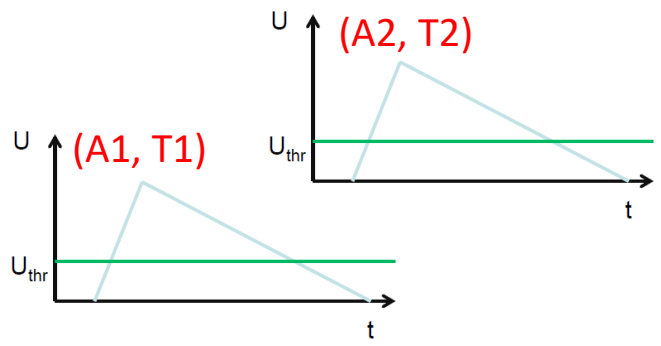
The time-based simulation

- **The digitization should support the time-based simulation**
  - Because
    - Panda readout is trigger-less
    - At 20 MHz interaction rate, pile-up occurs
  - Need to handle
    - Add up multiple hits in SADC
    - Separate waveforms as part of feature extraction

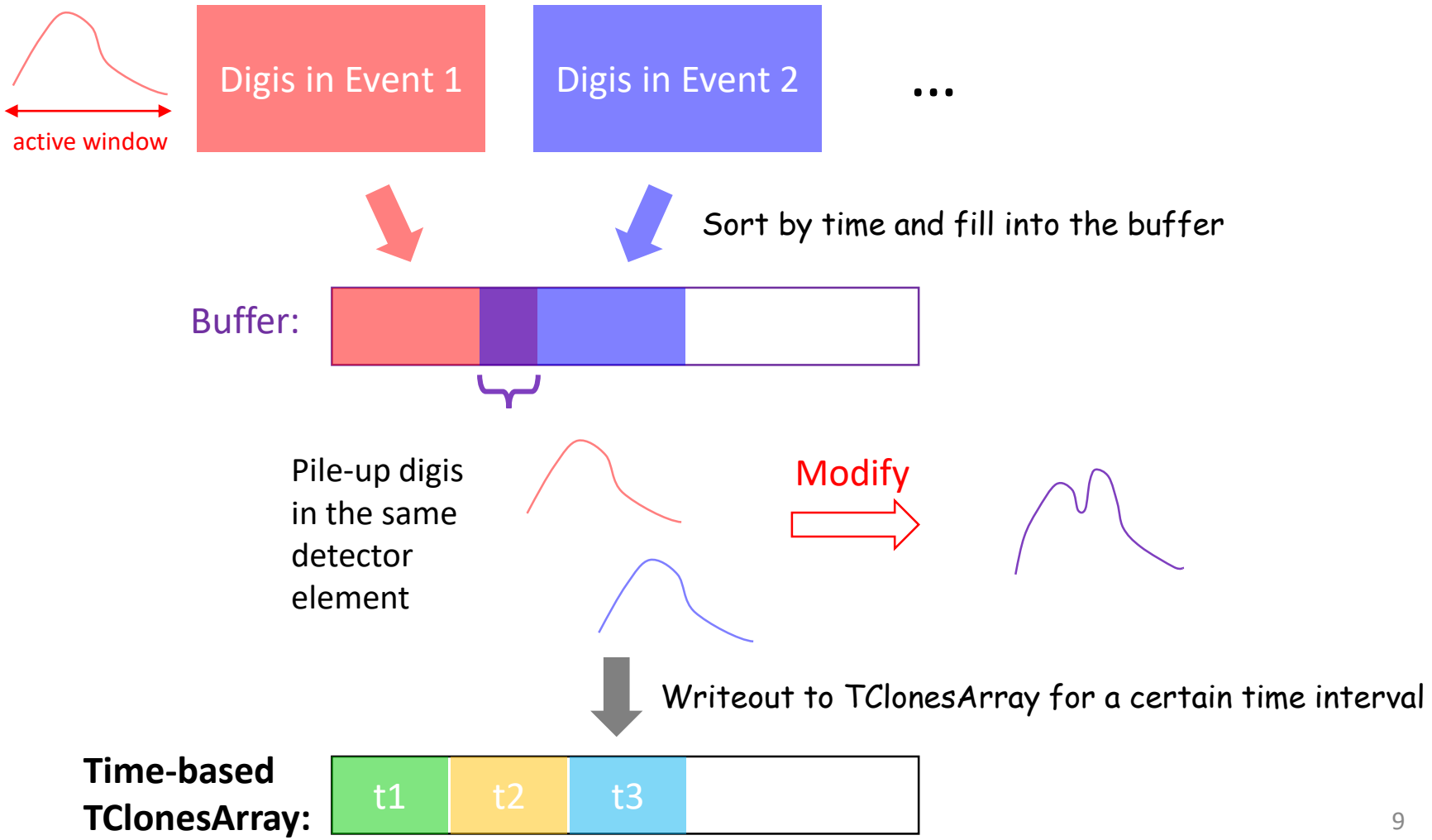
Add up hits



Extract amplitudes/times



The time-based simulation (II)



The strategy

- **There are two digitization algorithms in PandaRoot**

	The default package	The forward package
Scope	All EMC	Only FW Endcap
Multi-gain waveform	No	Yes
Time-based simulation / Waveform activate window	Support / Fixed	Support / Dynamic
Scalability	OK	Easier

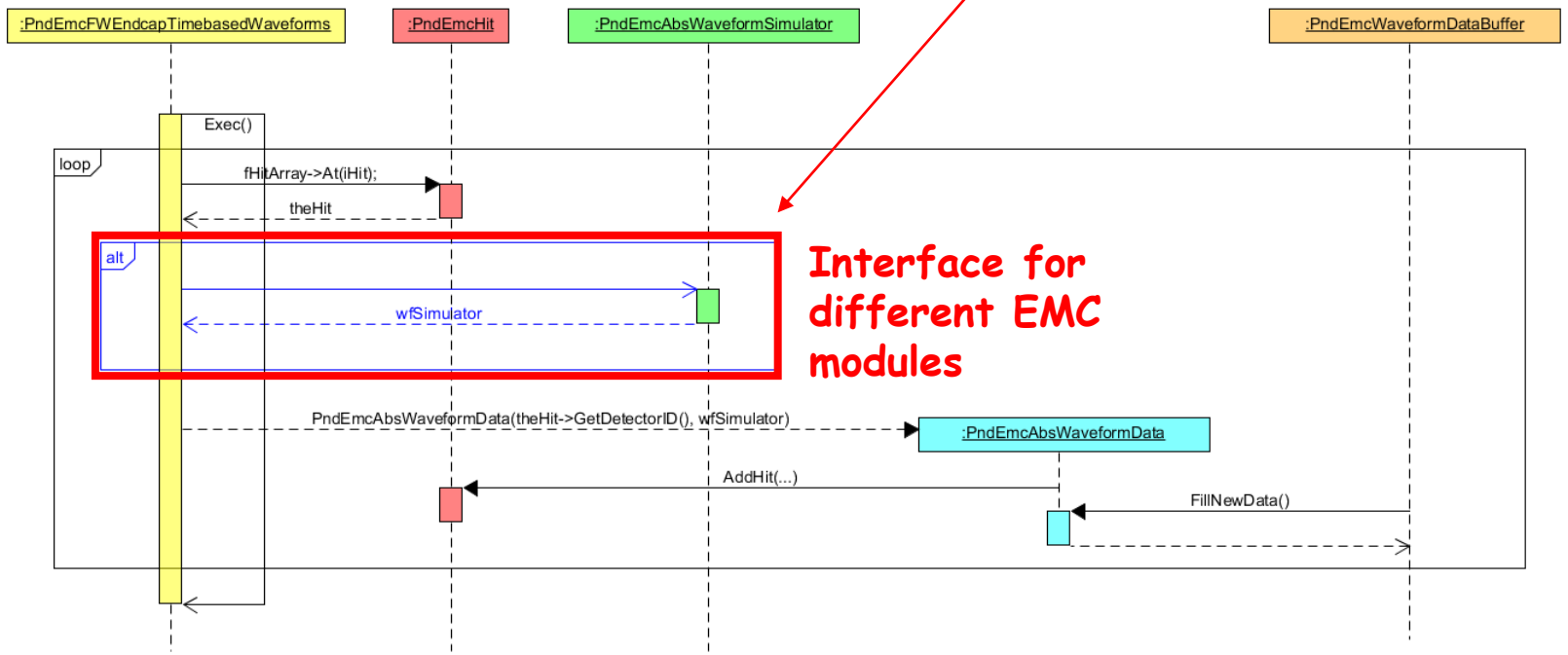
- **Implement a standalone backward endcap digitization package based on the forward package. Will study and test the digitization algorithms with this workable package.**

- **Need to consider the integration with other EMC modules**

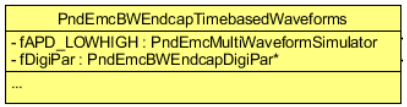
Signal generator

The hits to waveforms process

Different simulators for different EMC parts (e.g. different noise model/digitization parameters). Need to implement a new simulator for the backward endcap.

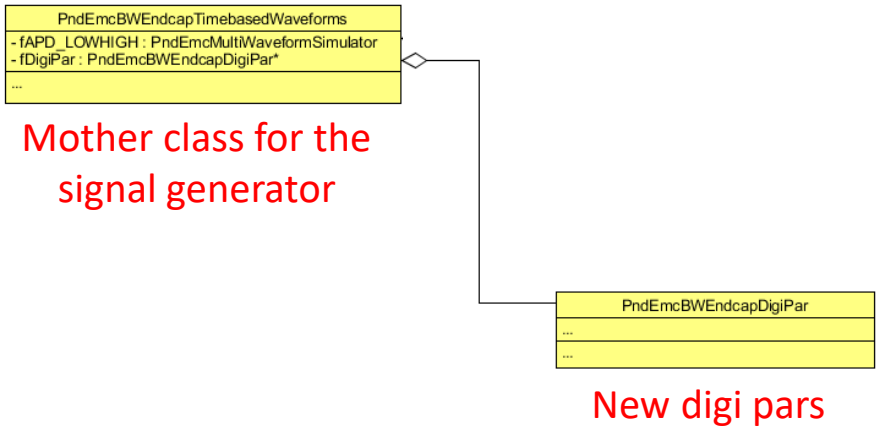


The main class



Define a new main class for the signal generator

The digitization parameters



Mother class for the signal generator

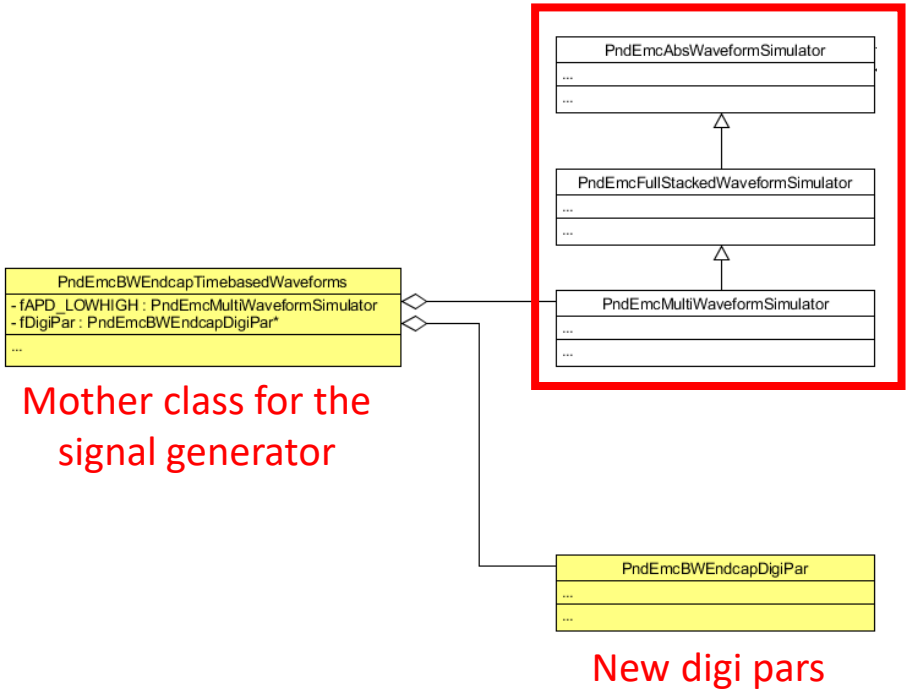
New digi pars

PndEmcBWEndcapDigiPar

+ GetTimeBeforeHit() : Double_t
+ GetTimeAfterHit() : Double_t
+ GetWfCutOffEnergy() : Double_t
+ GetSampleRate() : Double_t
+ GetNBits() : Int_t
+ GetPulseshapeTau() : Double_t
+ GetPulsesapeN() : Double_t
+ GetEnergyRangeHigh() : Double_t
+ GetEnergyRangeLow() : Double_t
+ GetGHLGRatio() : Double_t
+ GetADCNoiseHigh() : Double_t
+ GetADCNoiseLow() : Double_t
+ GetFENoiseHigh() : Double_t
+ GetFENoiseLow() : Double_t
+ GetPssigmaHigh() : Double_t
+ GetPsSigmaLow() : Double_t
+ GetGesSigmaHigh() : Double_t
+ GetGesSigmaLow() : Double_t
+ GetFIRCoeffFile() : TString
+ GetTmaxTaps() : Double_t
...

All parameters for signal generation and feature extraction

The simulator

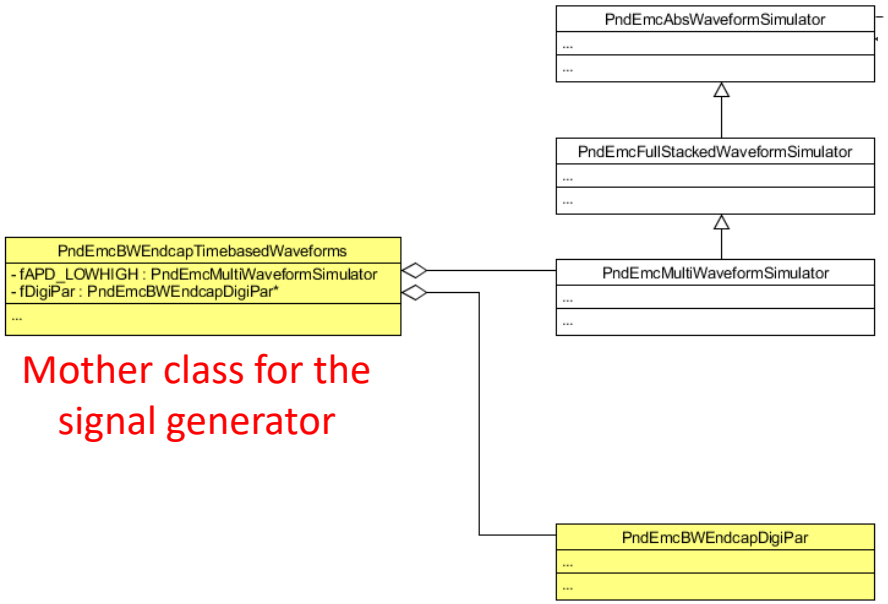


Mother class for the signal generator

New digi pars

- This is the interface
- **Input:** PndEmcHit
- **Output:** Digitized Waveform
- **Inner algorithms:** Dependent on EMC modules
- We implement a simulator for the backward endcap

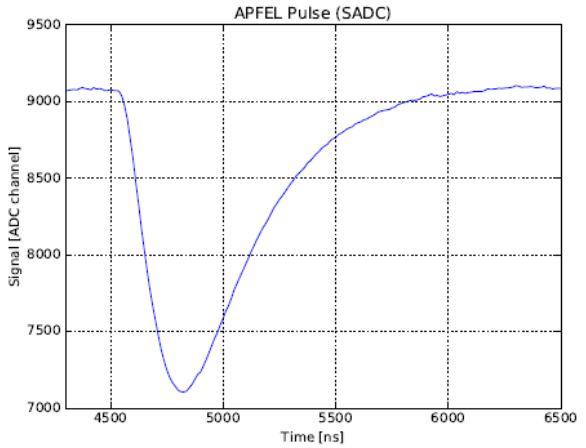
The pulse shape



Mother class for the signal generator

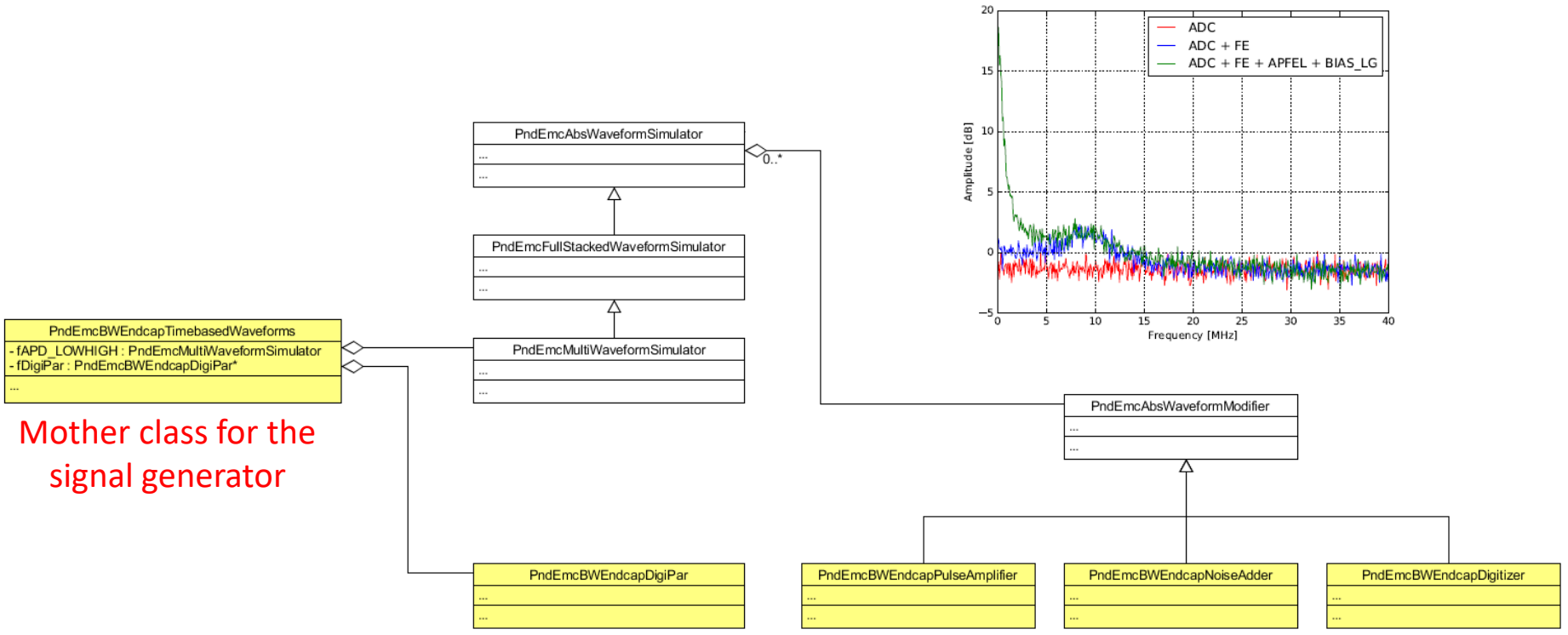
New pulse shape

New digi pars



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

The modifiers



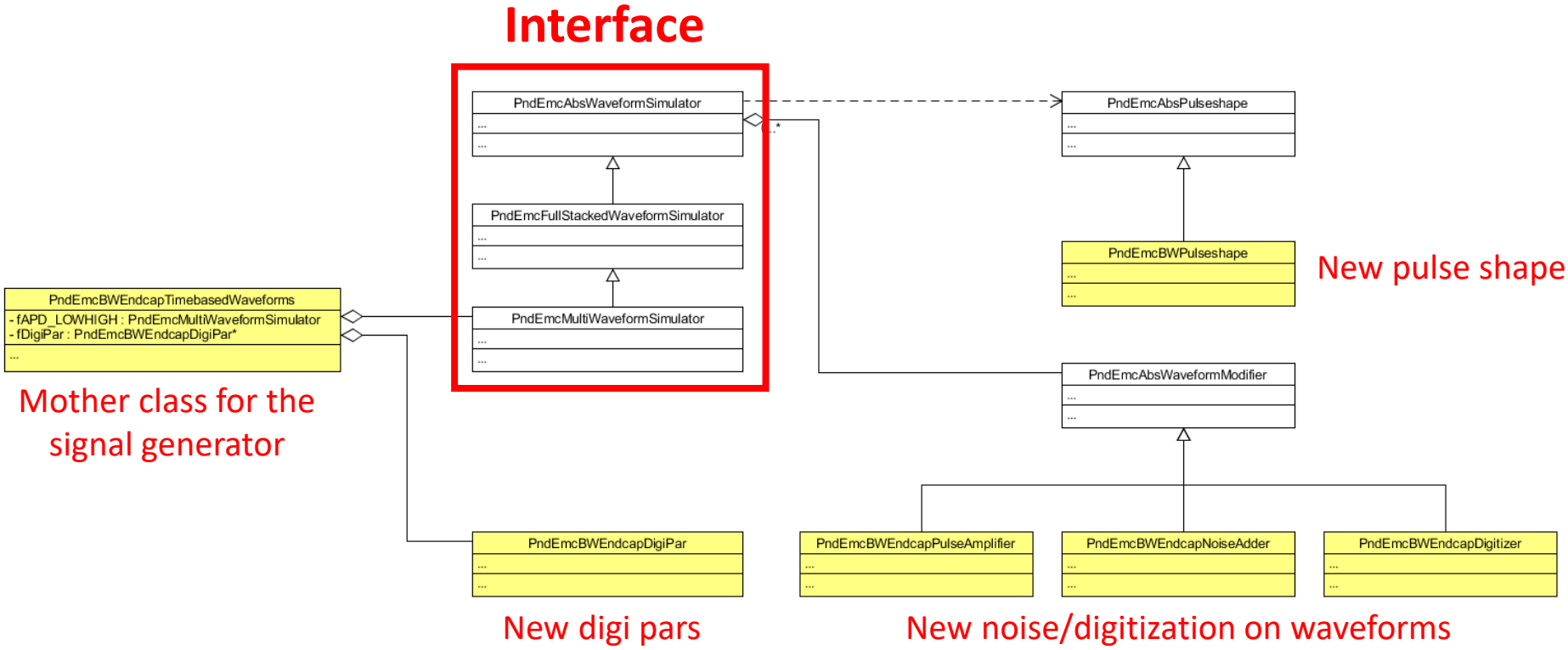
Mother class for the signal generator

New digi pars

New noise/digitization on waveforms

- **Power spectrum noise**
  - w/ FFT analysis, smearing in frequency domain
- **ADC and transmission noise**
  - Randomly smearing in time domain

Class diagram for signal generator

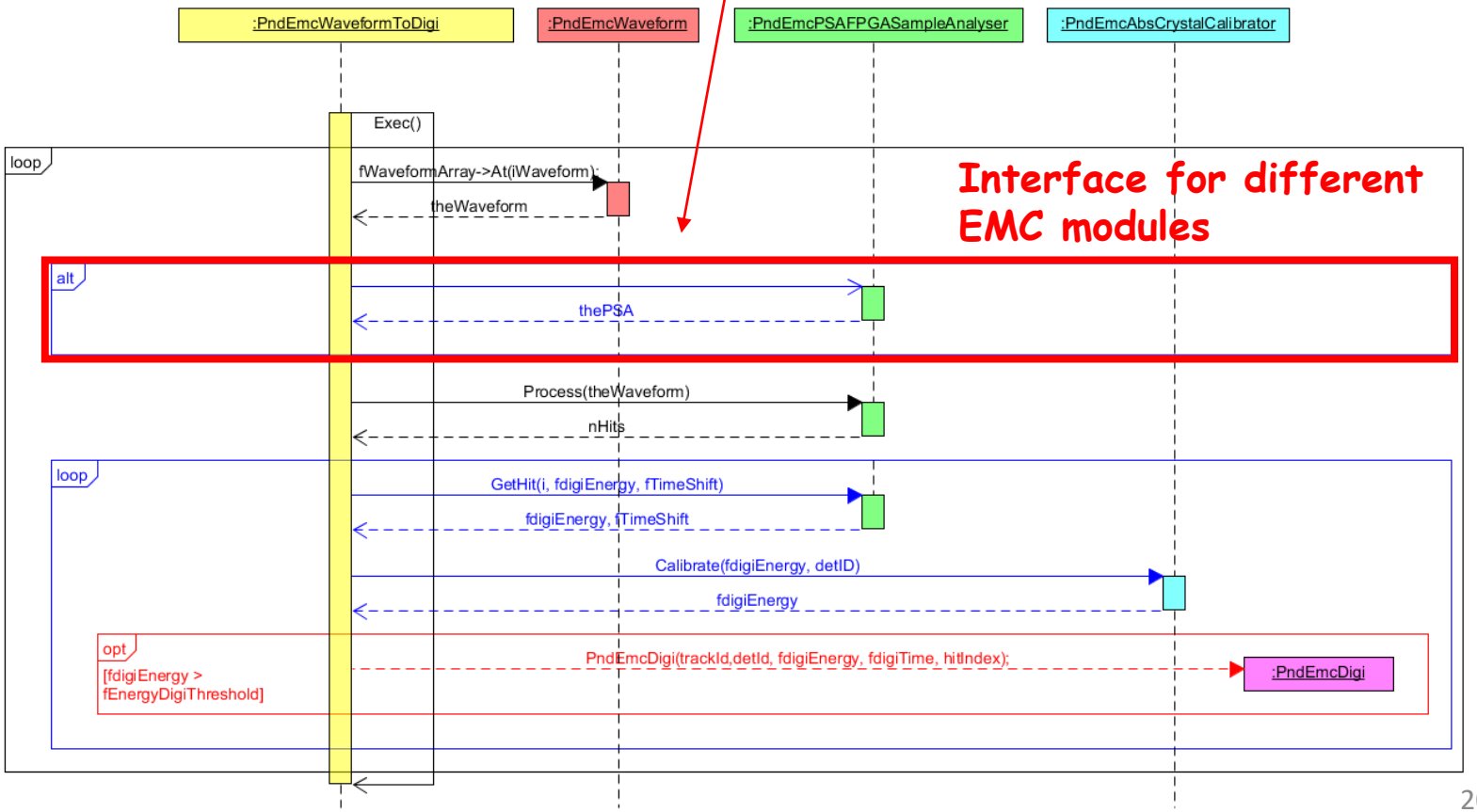


Newly added for BW

Feature extraction

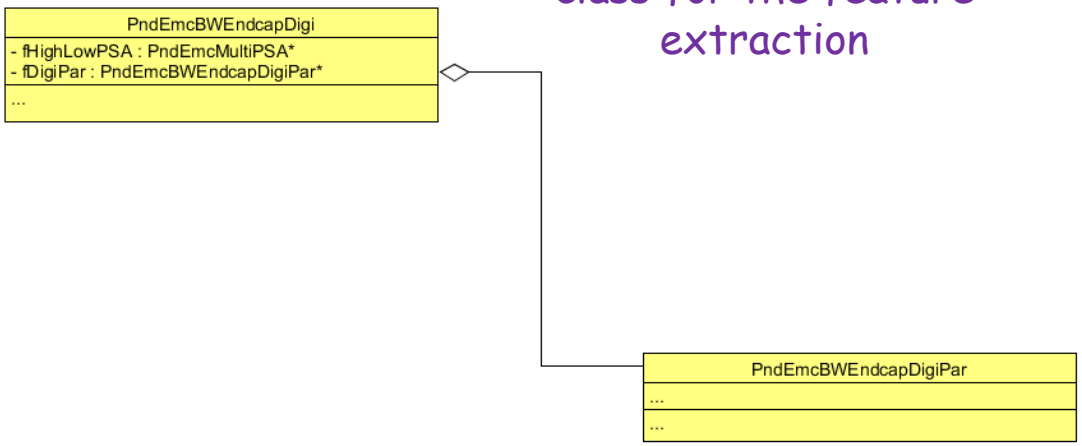
The waveforms to digis process

Different analyzers for different EMC parts (e.g. different feature extraction algorithms). Need to implement a new simulator for the backward endcap.



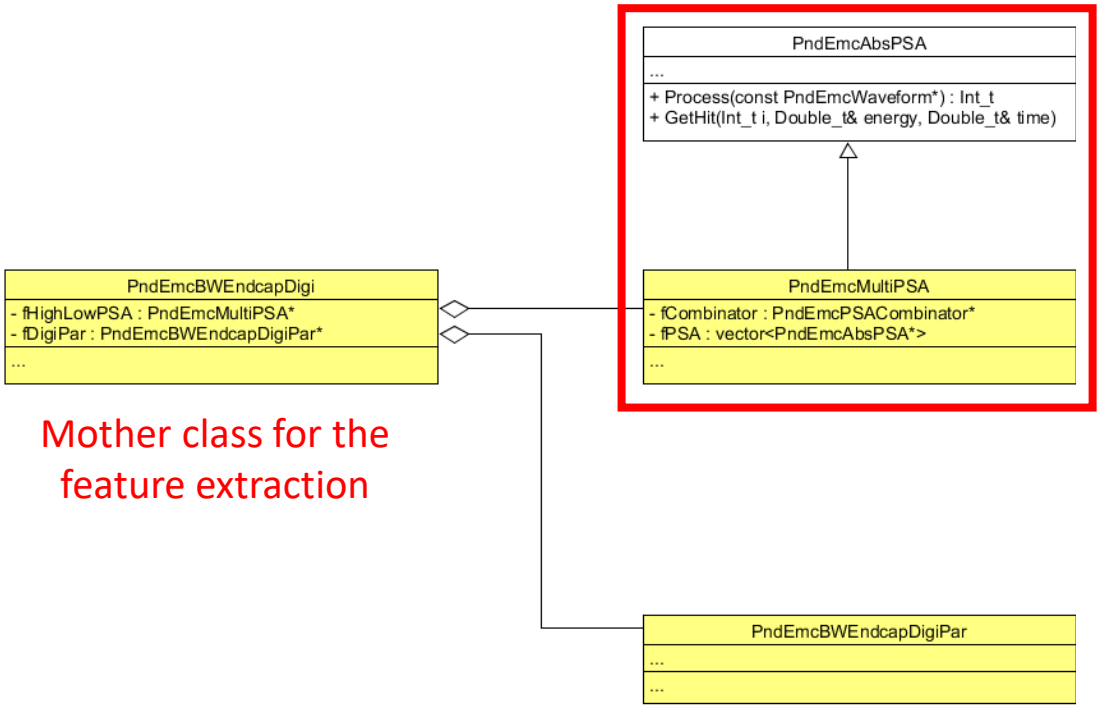
The main class

Define a new main class for the feature extraction



Digi pars (the same to the signal generator)

The PSA



Mother class for the feature extraction

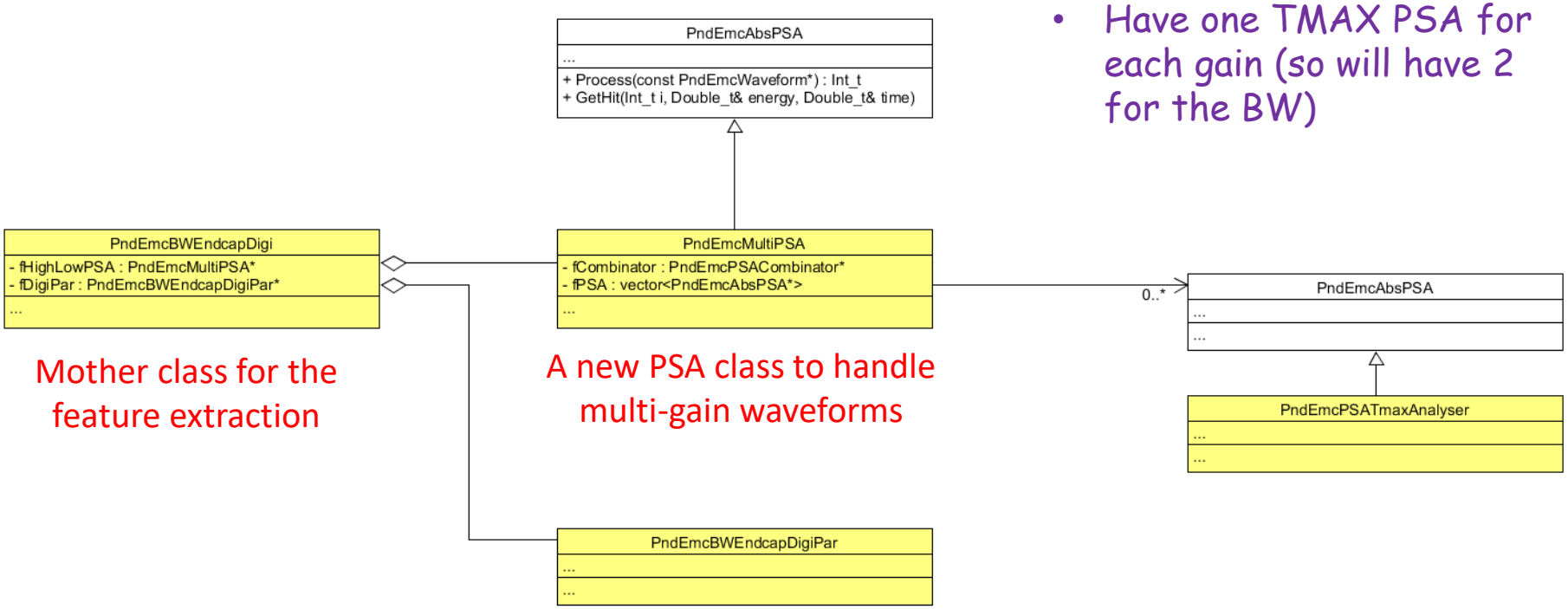
Digi pars (the same to the signal generator)

Newly added for BW

- The PSA (pulse shape analyser) is the interface to analysis waveforms.
- **Input:** PndEmcWaveform
- **Output:** Hits with energy/time information of the input waveform
- **Inner algorithms:** Depend on EMC modules
- We implement a new PSA to handle the multi-gain waveform

The TMAX PSA

- The TMAX PSA contains all the feature extraction algorithms
- Have one TMAX PSA for each gain (so will have 2 for the BW)



Mother class for the feature extraction

A new PSA class to handle multi-gain waveforms

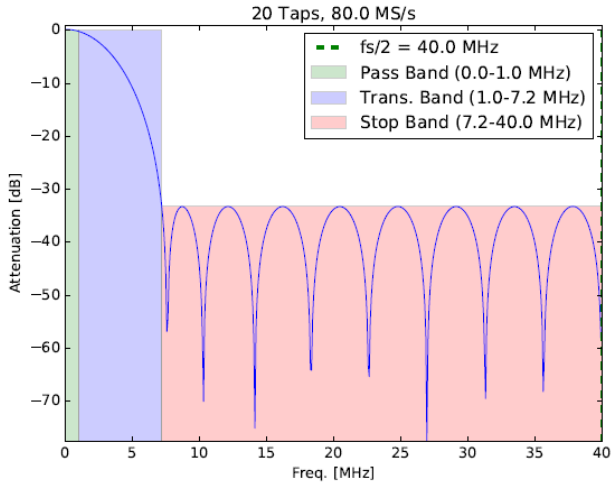
Digi pars (the same to the signal generator)

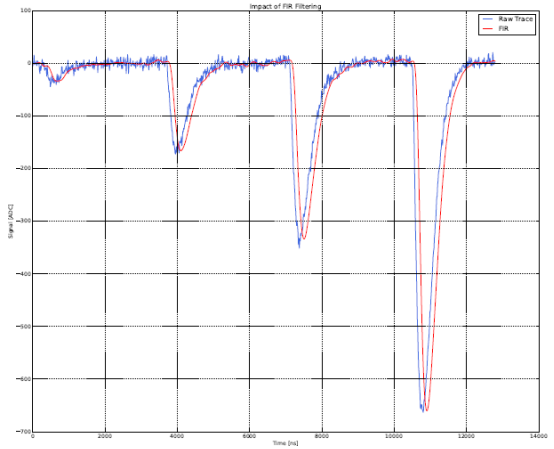
Newly added for BW

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
- $\mathbf{out}[n] = h_0\mathbf{in}[n] + h_1\mathbf{in}[n-1] + ... + h_N\mathbf{in}[n-N]$





Amplitude/Time path

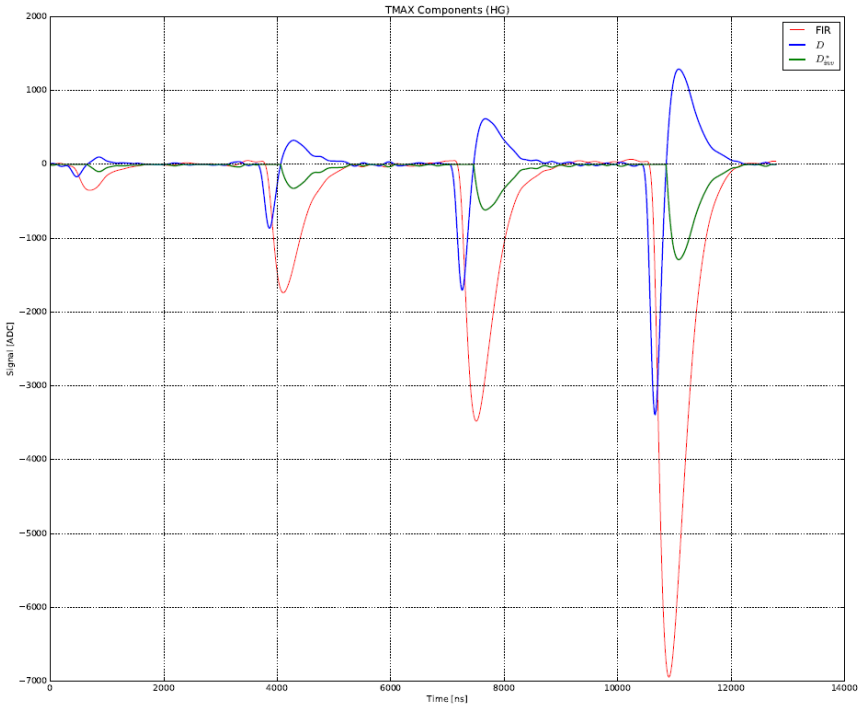
PndEmcPSATmaxAnalyser
- fEnergyList : vector<Double_t> - fTimeList : vector<Double_t>
+ Process(const PndEmcWaveform*) : Int_t + GetHit(Int_t, Double_t&, Double_t&) - fir(Int_t*, Int_t) : Double_t* - hit_det(Int_t, Int_t, Int_t) : Double_t

The T_{MAX} filter:

Deviation: $$D[i] = T[i + r] - T[i]$$

$$D^*_{inv}[i] = -\Theta(-D[i]) \cdot D[i]$$

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



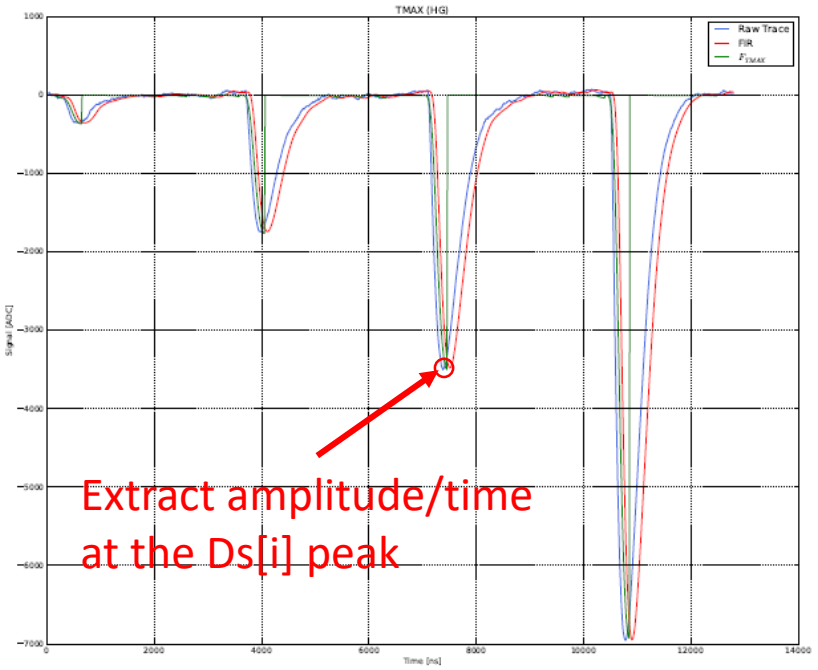
Amplitude/Time path (II)

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 T_{MAX} filter:

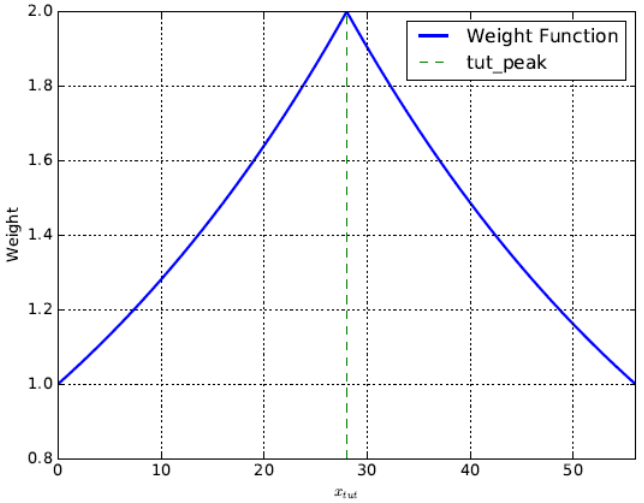
$$D_s[i] \mapsto \begin{cases} F_{TMAX}[i] = F_{TMAX}[i-1] + \frac{D_s[i]}{r} & : D_s[i] < 0 \\ F_{TMAX}[i] = 0 & : D_s[i] = 0 \end{cases}$$

D_s integration



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

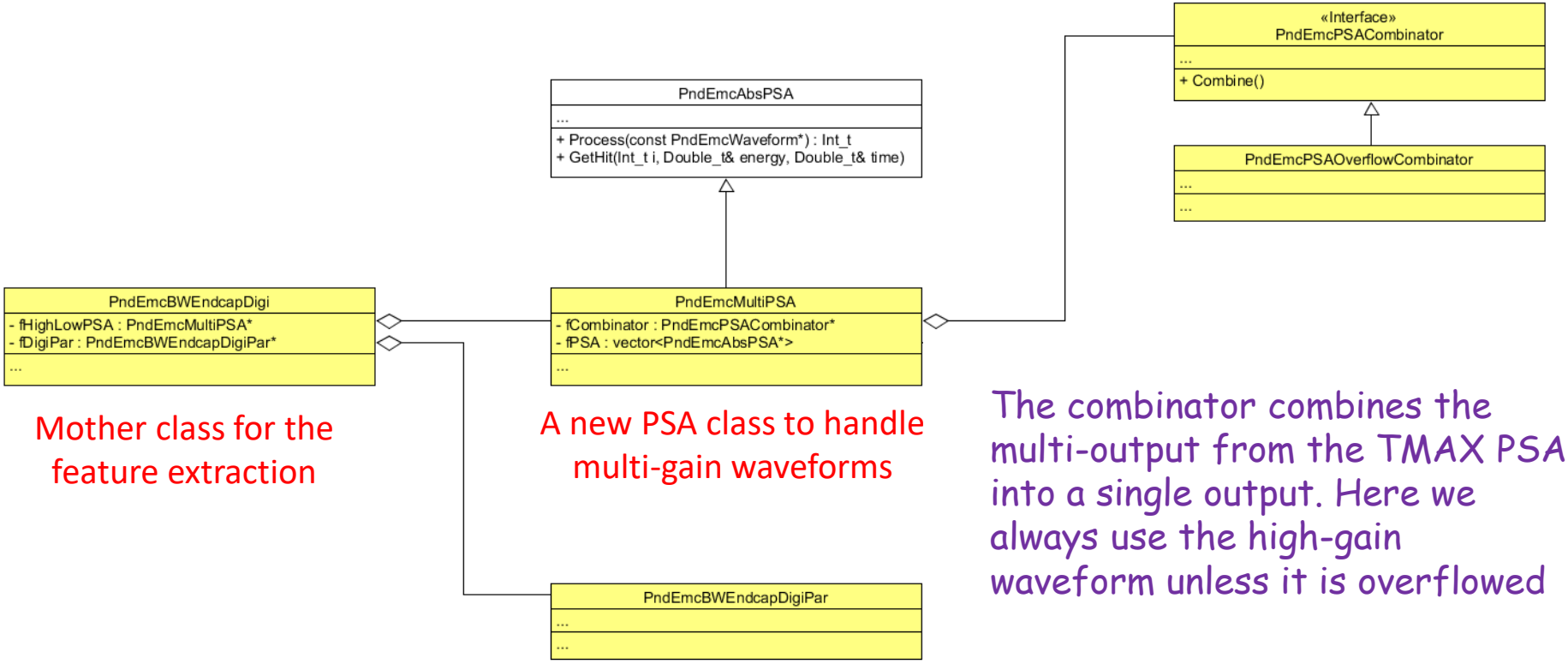


- A function to weight the hit detection with the time under threshold
- The weight function is convoluted with the extraction function, and a hit is detected when its value passes a threshold

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

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

The combinator



Mother class for the feature extraction

A new PSA class to handle multi-gain waveforms

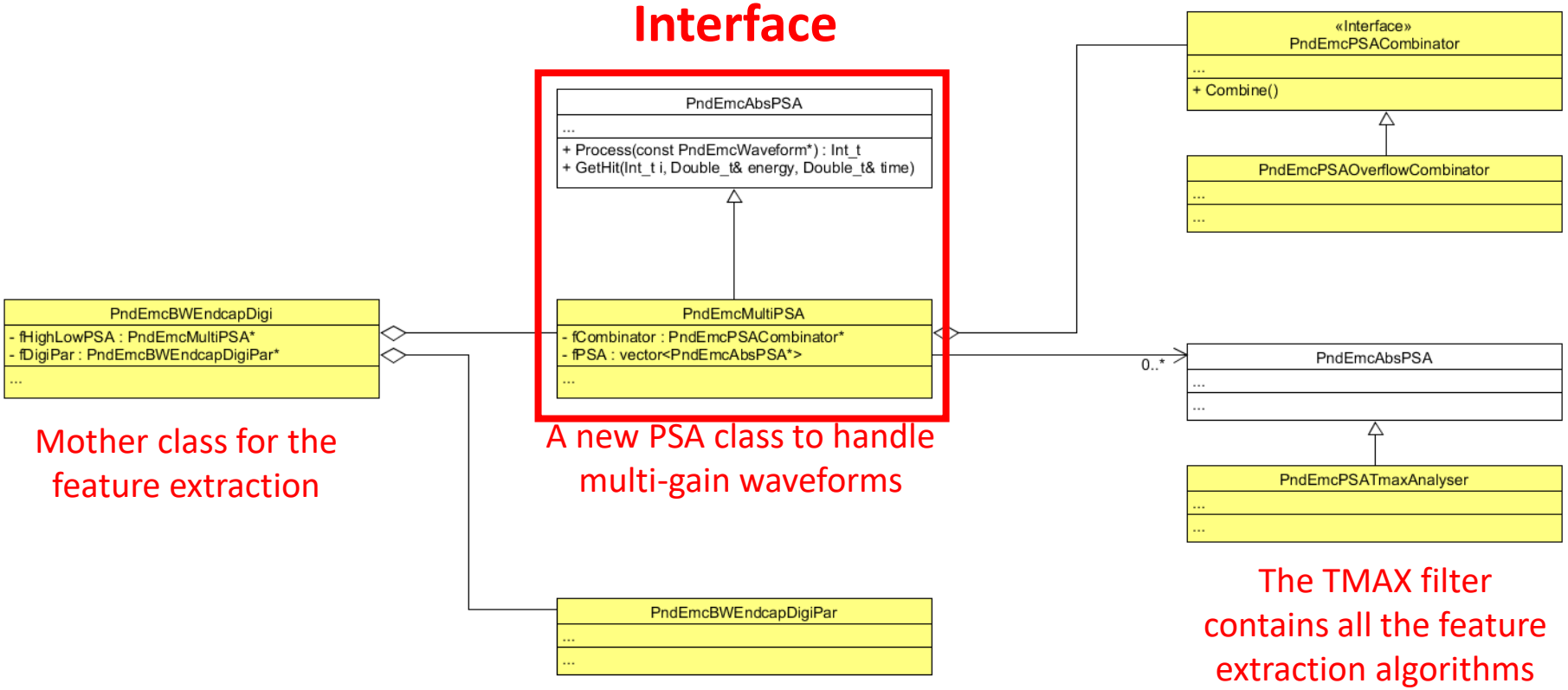
The combinator combines the multi-output from the TMAX PSA into a single output. Here we always use the high-gain waveform unless it is overflowed

Digi pars (the same to the signal generator)

Newly added for BW

Class diagram of the BW package

The combinator combines the multi-waveform input to a single output digi. Now we always use the high-gain waveform unless it is overflowed



Mother class for the feature extraction

A new PSA class to handle multi-gain waveforms

The TMAX filter contains all the feature extraction algorithms

Digi pars (the same to the signal generator)

Newly added for BW

Summary and outlook

- **Have developed a digitization package for the backward endcap**
- **The new code is finished and compiled**
- **Will start the full test and give some test results in the next collaboration meeting**
- **Need to integrate the code with other EMC modules. Further discussion will be needed**

Backups

PndEmcFullStackedWaveformSimulator

- determines length of simulation window via threshold
- if $threshold_{simulation} < threshold_{feature\ extraction}$: feature extraction will see "complete" pulses
- remaining drawback: rate of false-positive hits drastically reduced