AGATA Local Level Processing
(adapted from E. Clément and D. Bazzacco)

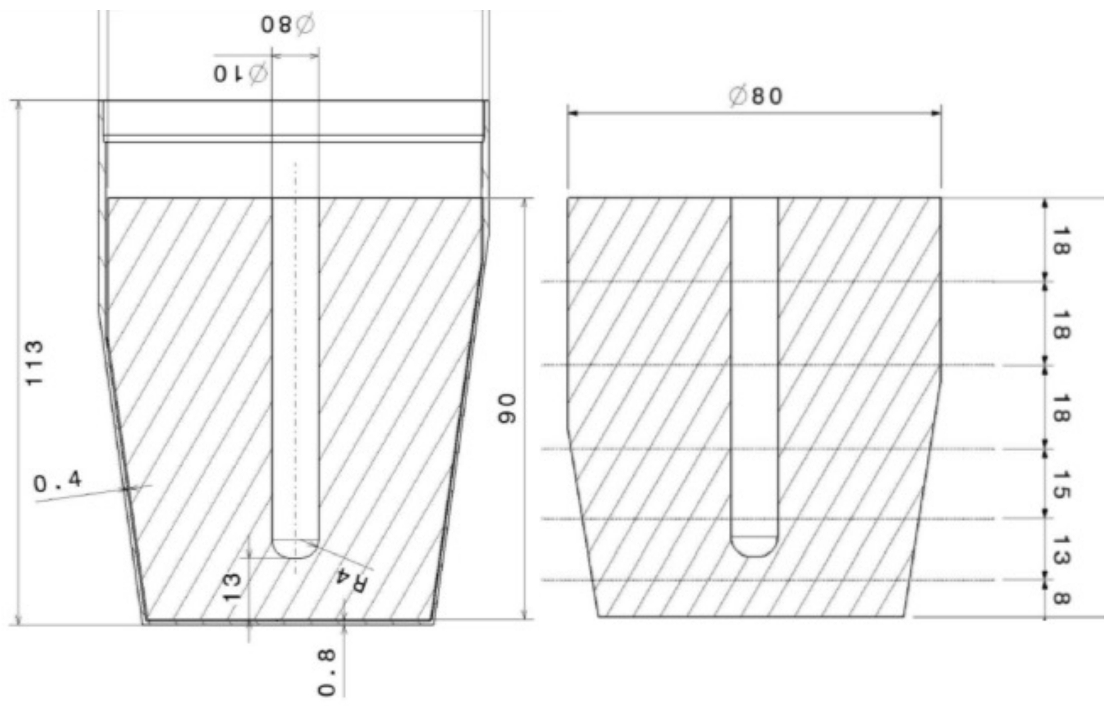
Jérémie Dudouet
on behalf of the AGATA Analysis Working Group

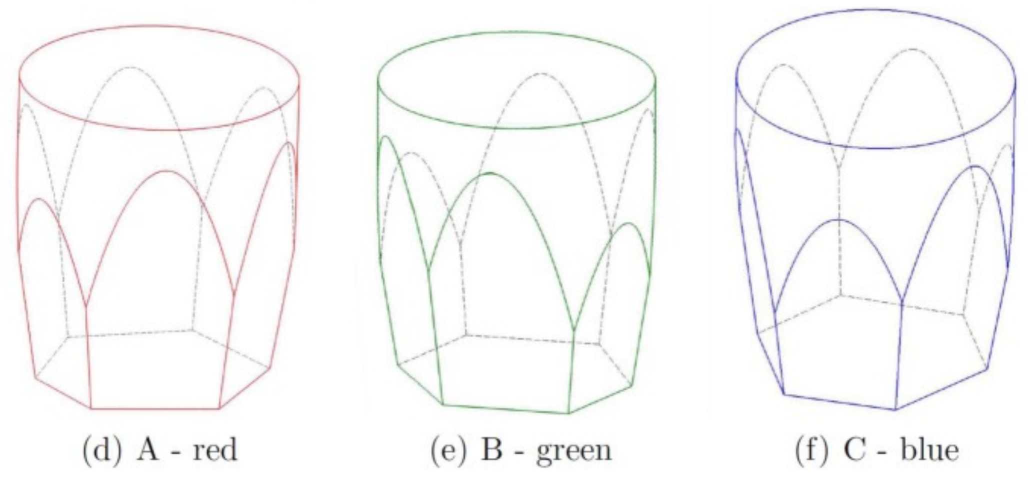
AGATA Analysis School: 11/09/2023, Legnaro

Topics

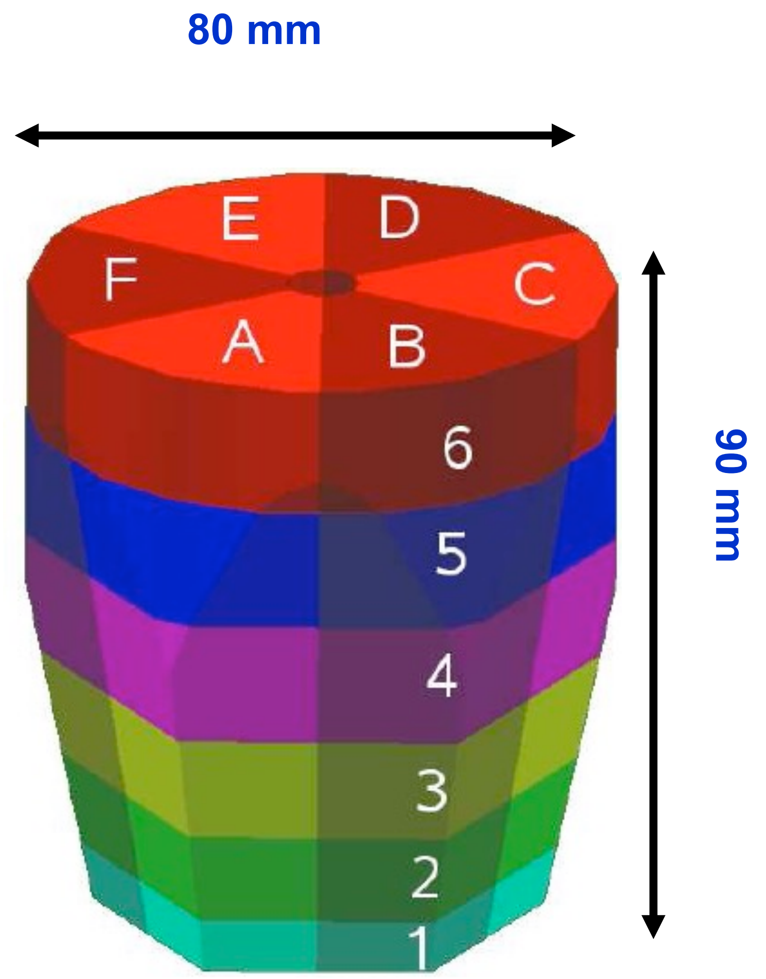
- General structure of data acquisition
- Data flow ➔ ADF
- Data processing model
- Emulators
- Narval (ADA/C/C++) and actors

AGATA detectors





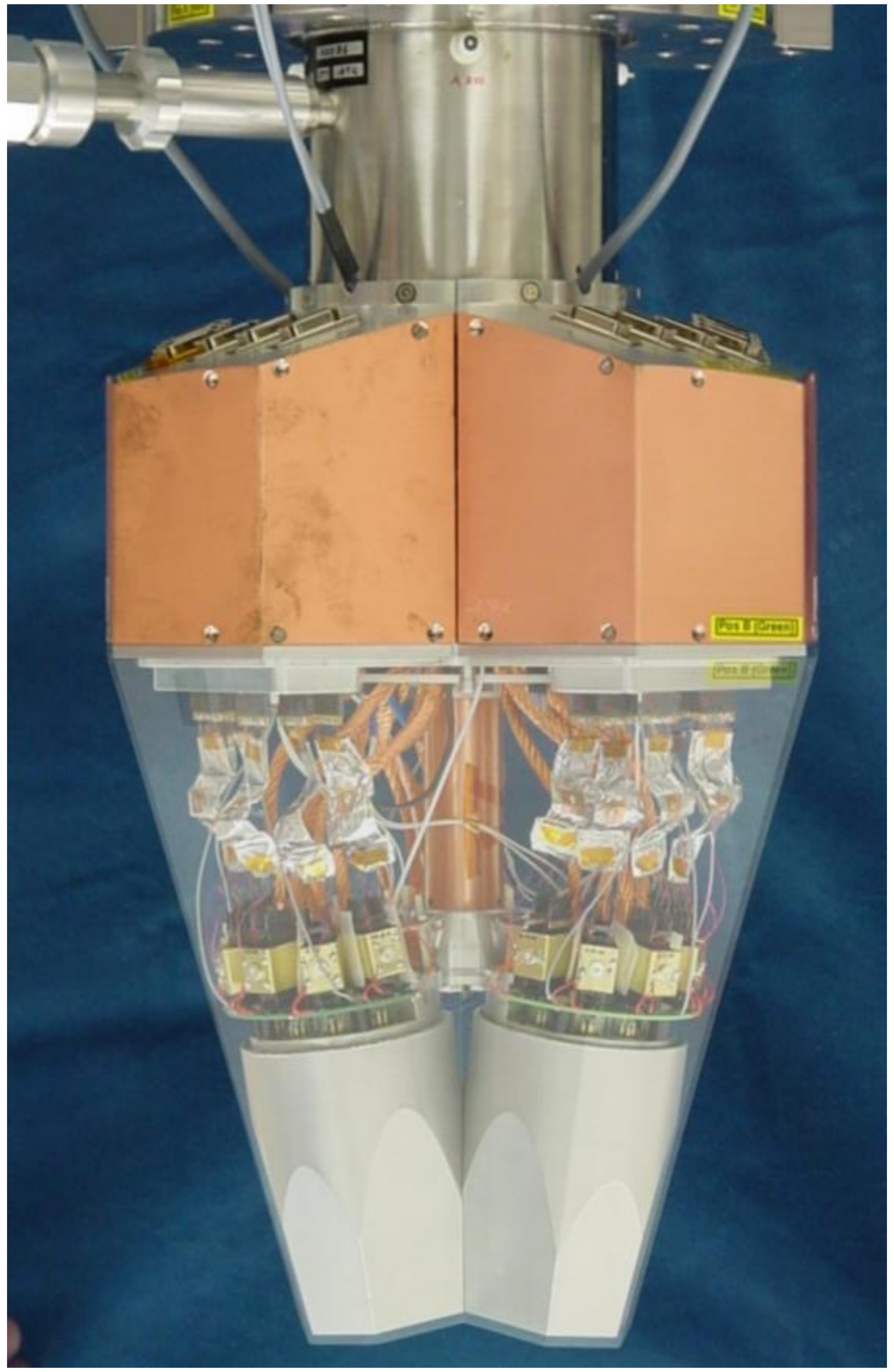
Volume ~370 cc   Weight ~2 kg
(shapes are volume-equalized to 1%)



6x6 segmented cathode
Cold FET for all signals

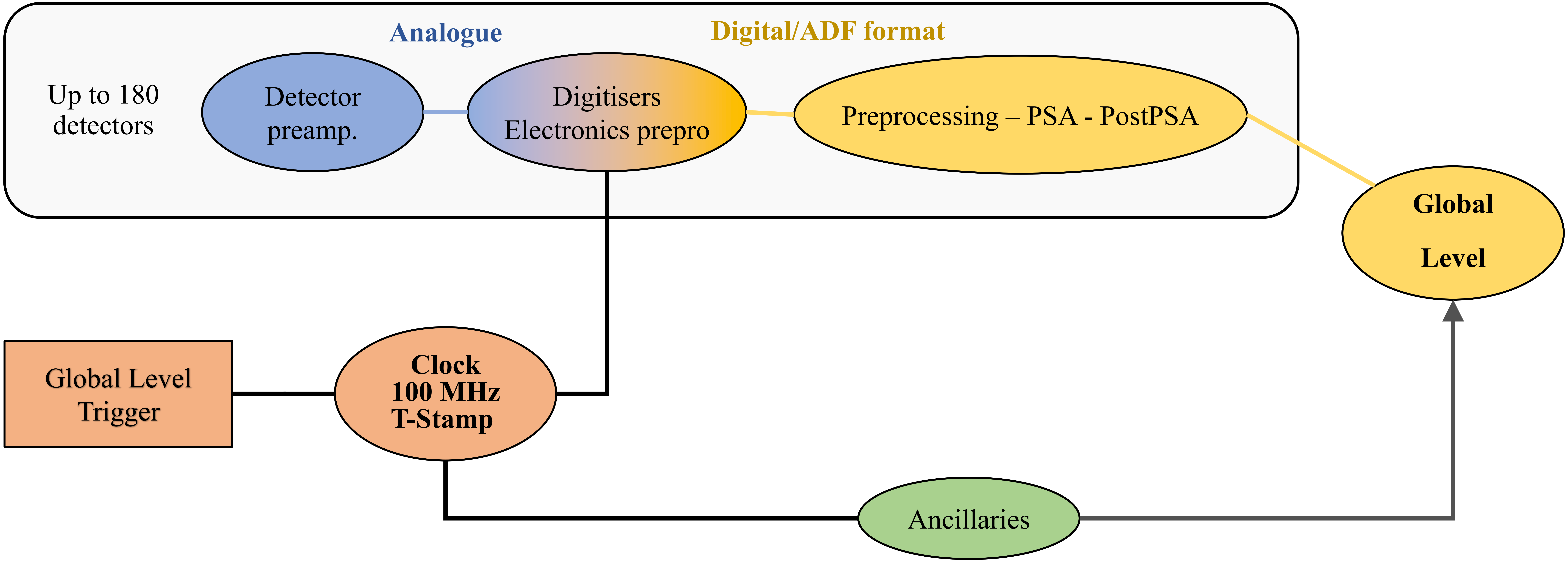
Energy resolution
Core: 2.35 keV
Segments: 2.10 keV
(FWHM @ 1332 keV)
A. Wiens et al. NIM A 618 (2010) 223
D. Lersch et al. NIM A 640(2011) 133

38 high-resolution signals / detector



AGATA Triple/Double Cryostat

Structure of Electronics and DAQ

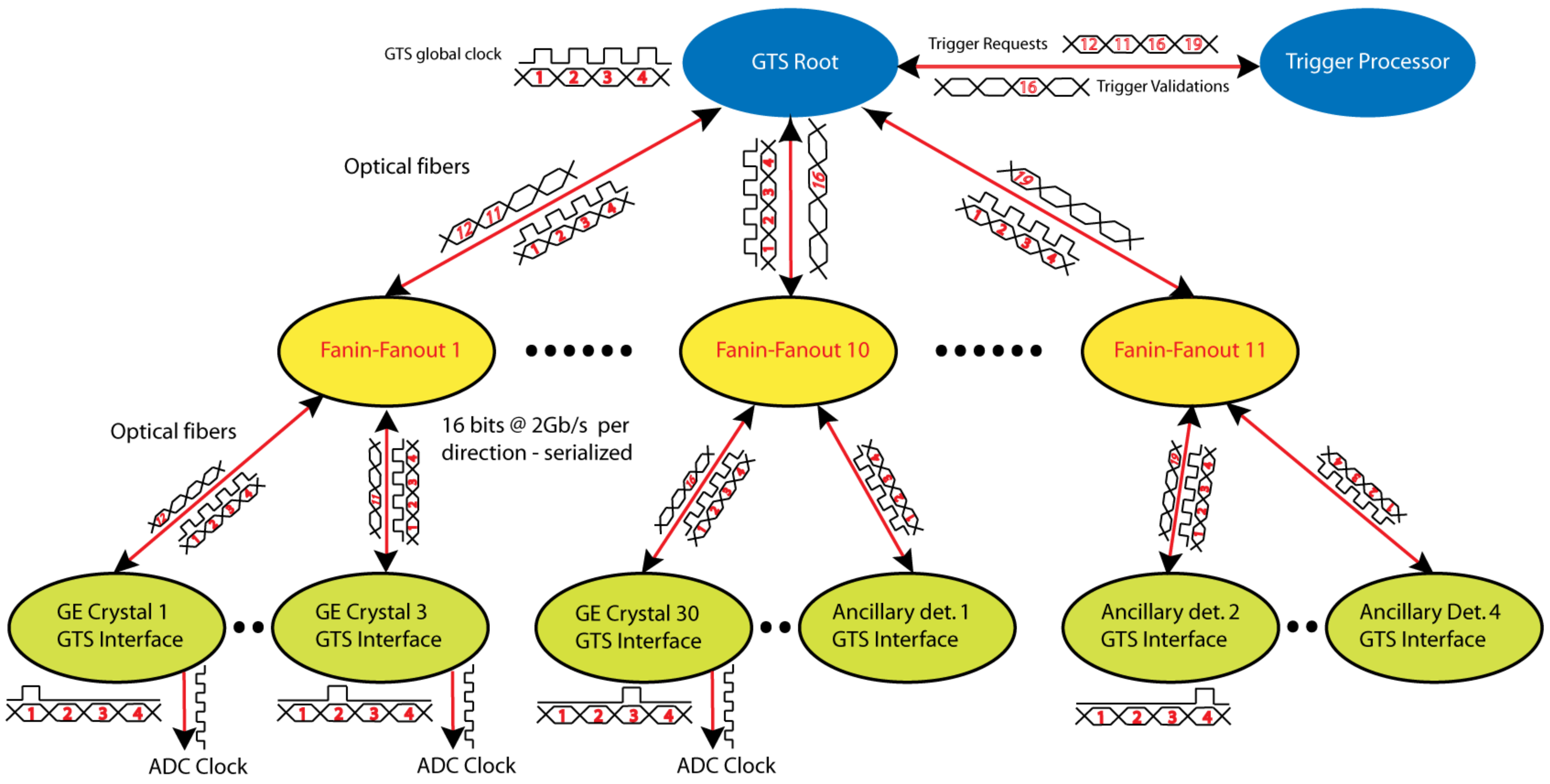


GTS : the system coordinator

All detectors operated on the same 100 MHz clock

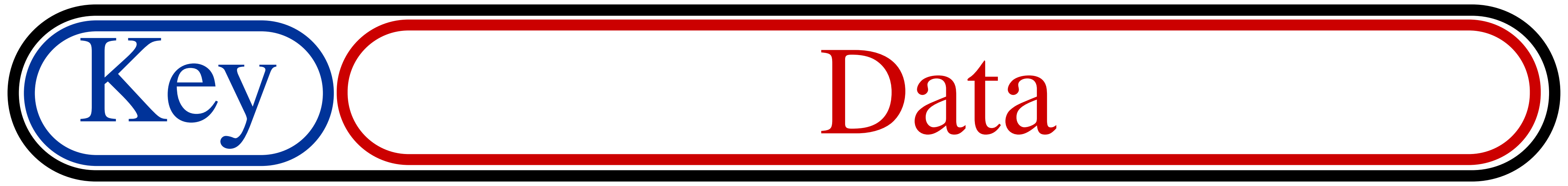
Upwards: Trigger requests, consisting of address (8 bit) and timestamp (16 bit)
max request rate 10 MHz total, 1 MHz/detector

Downwards: Validations/rejections, consisting of 48 bit timestamp + 24 bit event number



The AGATA Data Format: ADF

ADF Frame:



The key contains:
- Data lenght
- Data type
- Timestamp
- Event number

The Data contains either:
- data (energies, hits...)
- adf frames

The AGATA Data Format: ADF

The ADF.conf configuration file

```
Frame:   Agata data:crystal          4 0   Agata data:crystal          0 0
Frame:   Agata data:ccrystal         4 0   Agata data:ccrystal         65000 0
Frame:   Agata data:ranc0            4 0   Agata data:ranc0            65000 0
Frame:   Agata data:ranc1            4 0   Agata data:ranc1            65000 0
Frame:   Agata data:ranc2            4 0   Agata data:ranc2            65000 0
Frame:   Agata data:psa              4 0   Agata data:psa              0 0
Frame:   Agata data:tracked          4 0   Agata data:tracked          1 0
Frame:   Agata event:data            4 0   Agata event:data            4 0
Frame:   Agata event:ranc            4 0   Agata event:ranc            4 0
Frame:   Agata event:data:crystal    4 0   Agata event:data:crystal    4 0
Frame:   Agata event:data:ccrystal   4 0   Agata event:data:ccrystal   4 0
Frame:   Agata event:data:psa        4 0   Agata event:data:psa        4 0
Frame:   Agata event:data:ranc0      4 0   Agata event:data:ranc0      4 0
Frame:   Agata event:data:ranc1      4 0   Agata event:data:ranc1      4 0
Frame:   Agata event:data:ranc2      4 0   Agata event:data:ranc2      4 0
Frame:   Agata meta:vertex           4 0   Agata meta:vertex           0 0
Frame:   Agata meta:sync             4 0   Agata meta:sync             0 0
Frame:   Agata meta:scan             4 0   Agata meta:scan             0 0
```

Each key and frame is associated to a version number

Architecture of the system

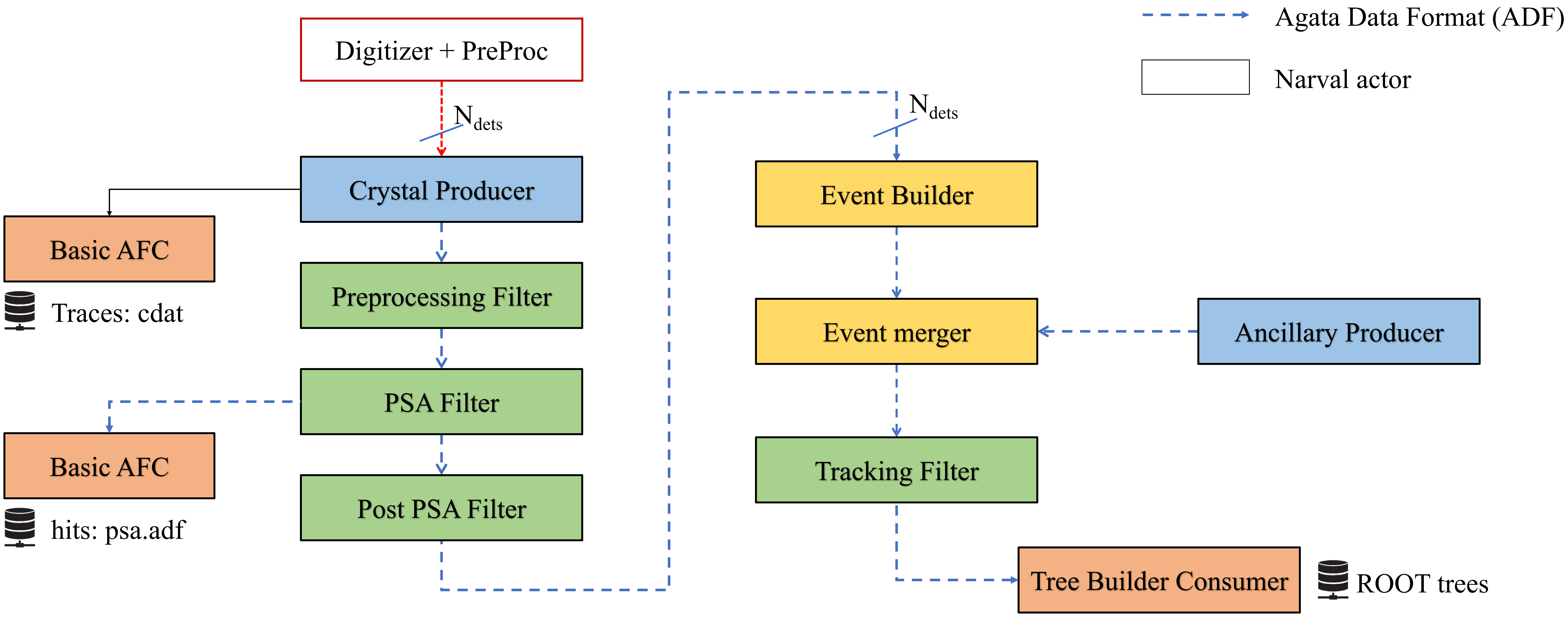
Local Level : where the individual detectors don’t know of each other.

- Electronics and computing follow a model with minimum coupling among the individual detectors, which are operated independently as long as possible
- Electronics is almost completely digital, operated on the same 100 MHz clock
- Data processing (in the electronics and in the front-end computers) is the same for all detectors and proceeds in parallel
- **Every chunk of produced data is tagged with a 48 bit number (time stamp) giving the absolute time (with a precision of 10 ns) since the last hard-reset of the system ➔ roll around takes place every 32.5 days.**

Global Level : where the detectors do know of each other

- By means of the real time trigger
- Via the event builder and merger that assemble the event fragments (including the ancillaries) into complete events that are further processed
- In the Tracking and in the Physical analysis stages

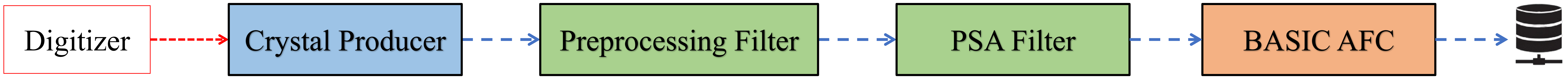
Architecture of the system



Data processing: NARVAL/DCOD

List of actors, applying successive operations on the data flow:

- **Producer:** Read data from outside (electronics/disk) to the NARVAL chain
- **Filters:** Process operations on the dataflow (calibrations/PSA/Tracking)
- **Dispatchers:** Merge multiple entries in the NARVAL chain (Event Builder/Event Merger)
- **Consumer:** Close the NARVAL chain (write data on disk)



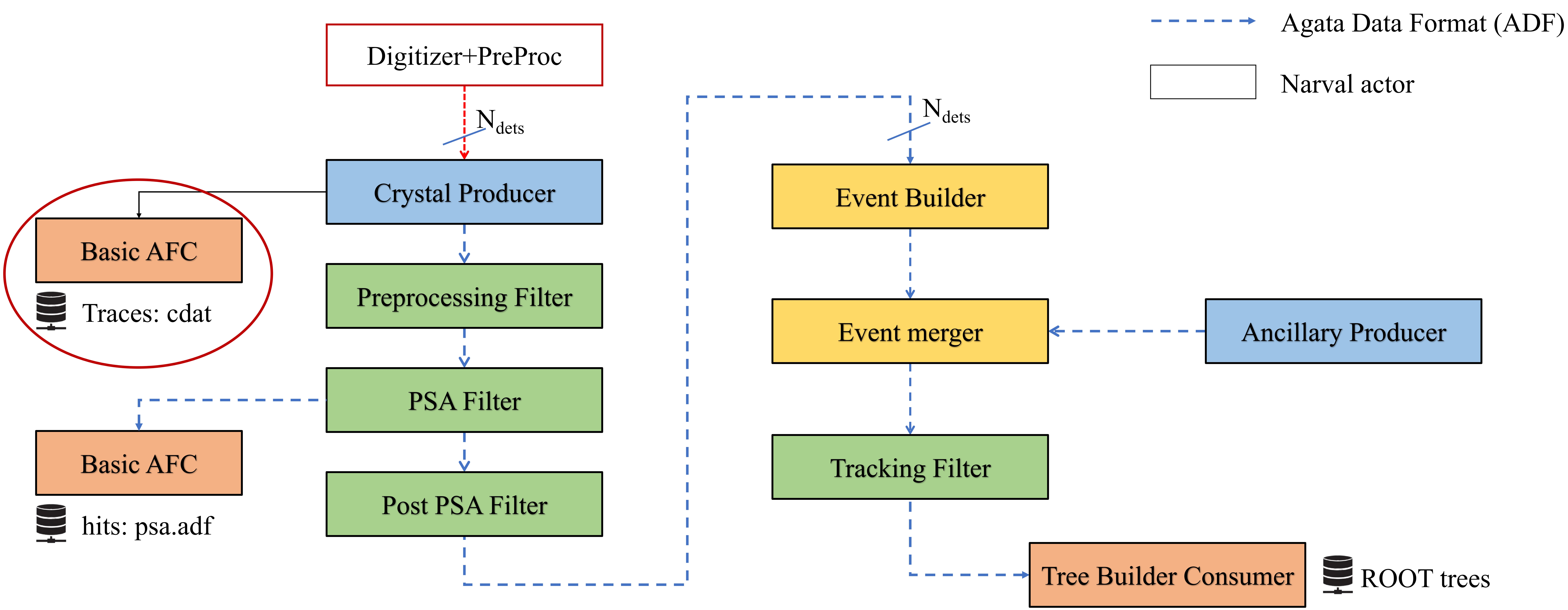
Standard actors chain for AGATA LLP

Data replay: the femul emulator

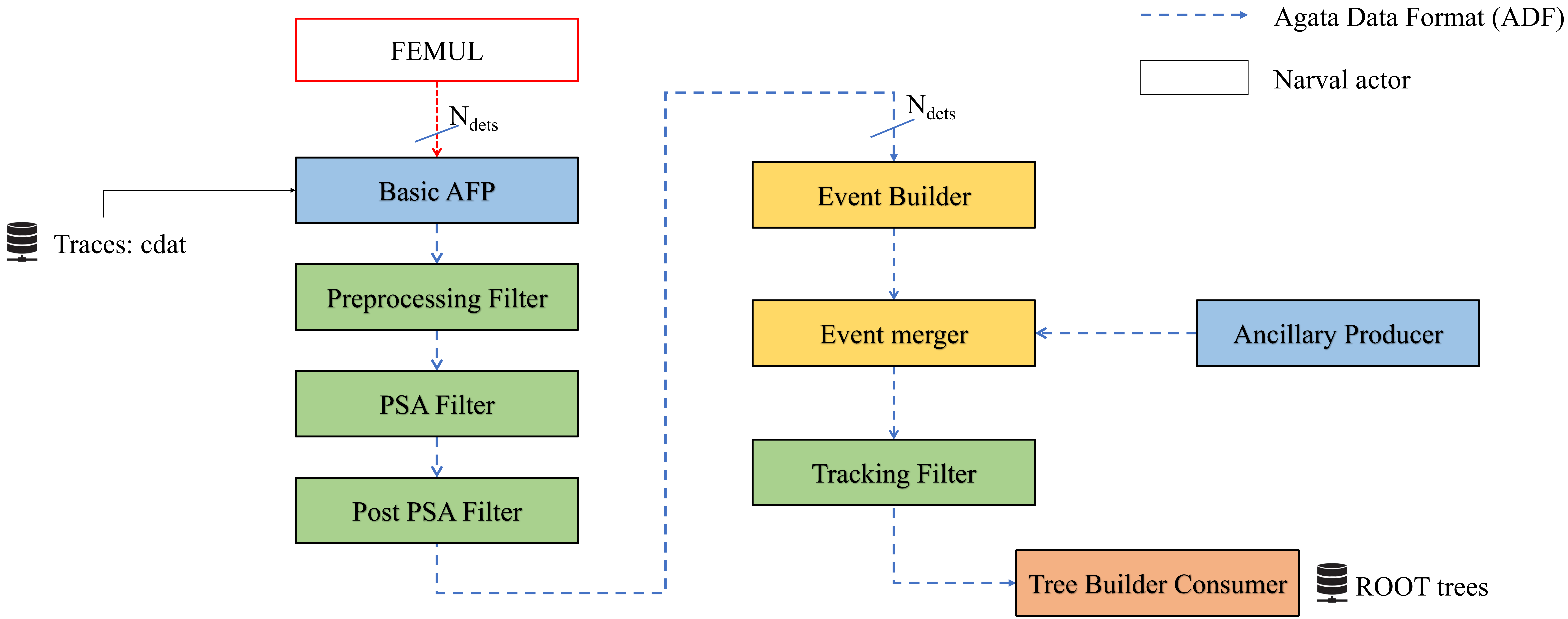
- Originally intended to help developing and debugging the user libraries which is very hard to do in a distributed processing environment like Narval.
- Gradually developed as a full emulation of the Narval framework with the limitation of being a single process running a specific machine (threads used to distribute the work on the available cores).
- Configuration (detectors, actors, ..) specified by a “topology” file.
- Generation of configuration parameters via *gen_conf.py*

➔ **Online == Offline processing**

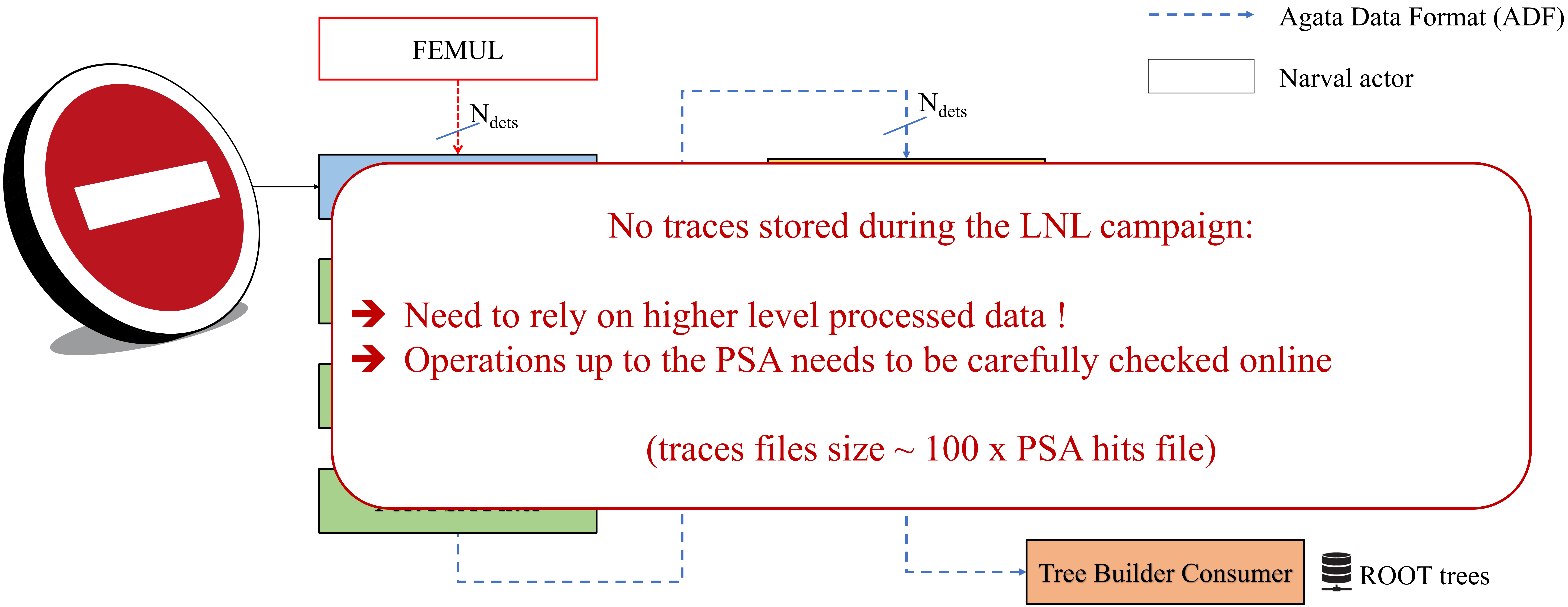
Data replay: the femul emulator



Data replay: the femul emulator



Data replay: the femul emulator



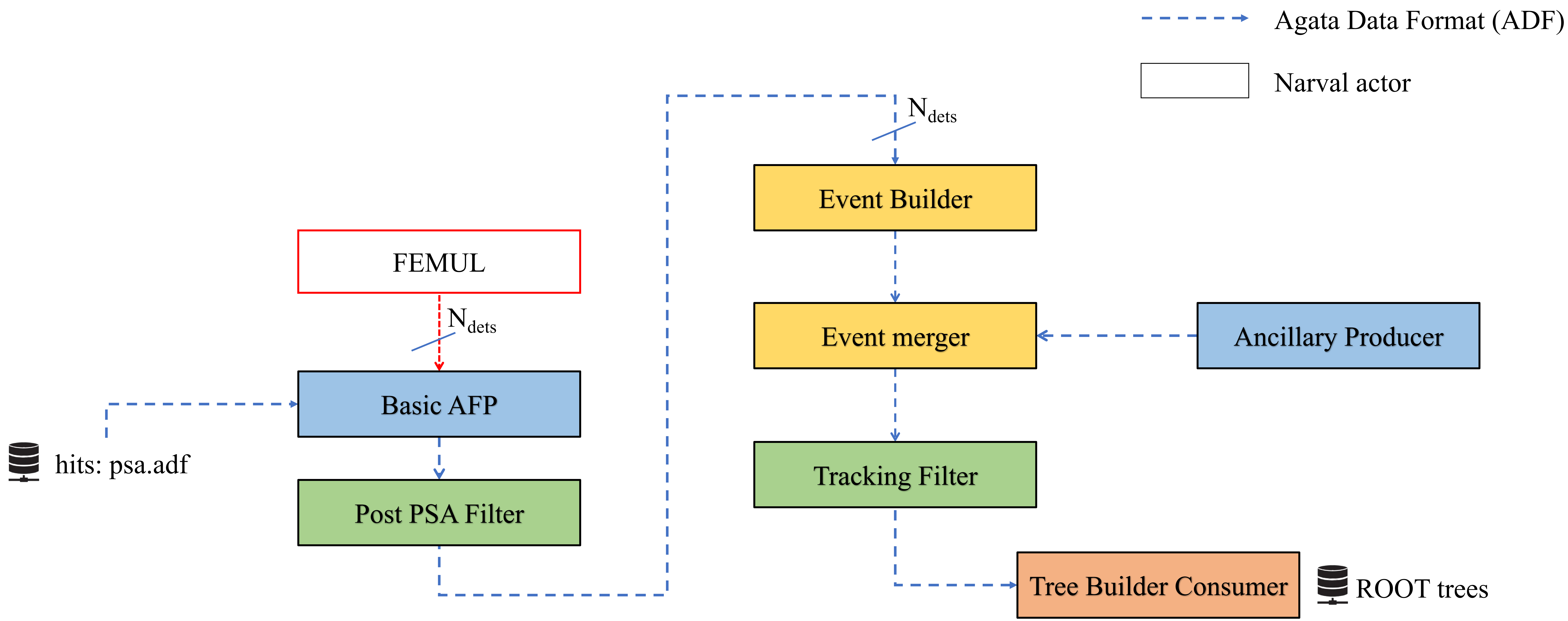
No traces stored during the LNL campaign:

- ➔ Need to rely on higher level processed data !
- ➔ Operations up to the PSA needs to be carefully checked online

(traces files size ~ 100 x PSA hits file)

Data replay: the femul emulator

Data replay: the femul emulator



Structure of analysis directories

The directory where you produce your data contains some standard sub-directories
(e.g. /agatadisks/exptname (EXXX) /(Config EXXX)/run_XXX_date)

Conf: Configuration of actors, calibrations, ... for each detector
→ **00A, 00B, 00C ... Ancillaries, Global, Merger**
with minimal differences between online and offline

Data: Data and spectra produced during the experiment
→ Online writes data here
→ Offline replay takes data from here

Out: Data and spectra produced during data replay
→ Offline writes data here

Typical configuration directories

Conf/12A

- CrystalProducer.conf
- CrystalProducerATCA.conf
- PreprocessingFilter.conf
- PreprocessingFilterPSA.conf
- PSAFilter.conf
- PostPSAFilter.conf
- xdir_1325-1340.cal
- xinv_1325-1340.cal
- BasicAFC.conf
- BasicAFP.conf

Conf/Builder

- EventBuilder.conf

Conf/Merger

- EventMerger.conf
- TrackingFilter.conf
- CrystalPositionLookUpTable
- TreeBuilder.conf

Binary spectra

- Simple C-style multidimensional (max 6) arrays written mostly in binary format
- For historical reasons the format is not recorded in the file
  → Often written as part of the file name:
  Prod__4-38-32768-UI__Ampli.spec is a file dump of an array defined as:
  unsigned integer Ampli[4][38][32768],
  containing **4*38** = 152 spectra of **32768** channels

- The viewers **TkT** and **Mat** can decode and interpret the format.
- Other programs (e.g. RecalEnergy) can interpret the spectrum length and type but the user have to specify the number of spectra to act upon.

Some Useful programs

The number of channels (38 x number of detectors) to be calibrated and checked at each analysis level is too large to be done one by one: **automatic tools and procedures are distributed**

→ **TkT spectrum viewer**: to plot any spectrum produced all along the actors chain

→ **RecalEnergy:** Analysis of spectra looking for peaks

→ **SortPsaHits:** Sort of PSA hits (special format) to determine neutron damage correction parameters

→ **gen_conf.py:** Unified procedure to produce configuration files for all actors

→ **solveTT.py:** Optimize time alignment of "equal" detectors

Local actors

- **CrystalProducer**
  - Readout of electronics, or get raw data from file
  - Local event builder
  - Save original data to be able to replay experiment
  - Raw projections

- **PreprocessingFilter**
  - Energy calibrations
  - Retrigger and Time calibration
  - Cross talk correction
  - Amplitude calibration and time alignment of traces
  - Improved pile-up rejection

- **PSAFilter**
  - Decomposition of calibrated experimental traces by comparison with a calculated signal basis
  - In principle more than one algorithm available but only one used in practice

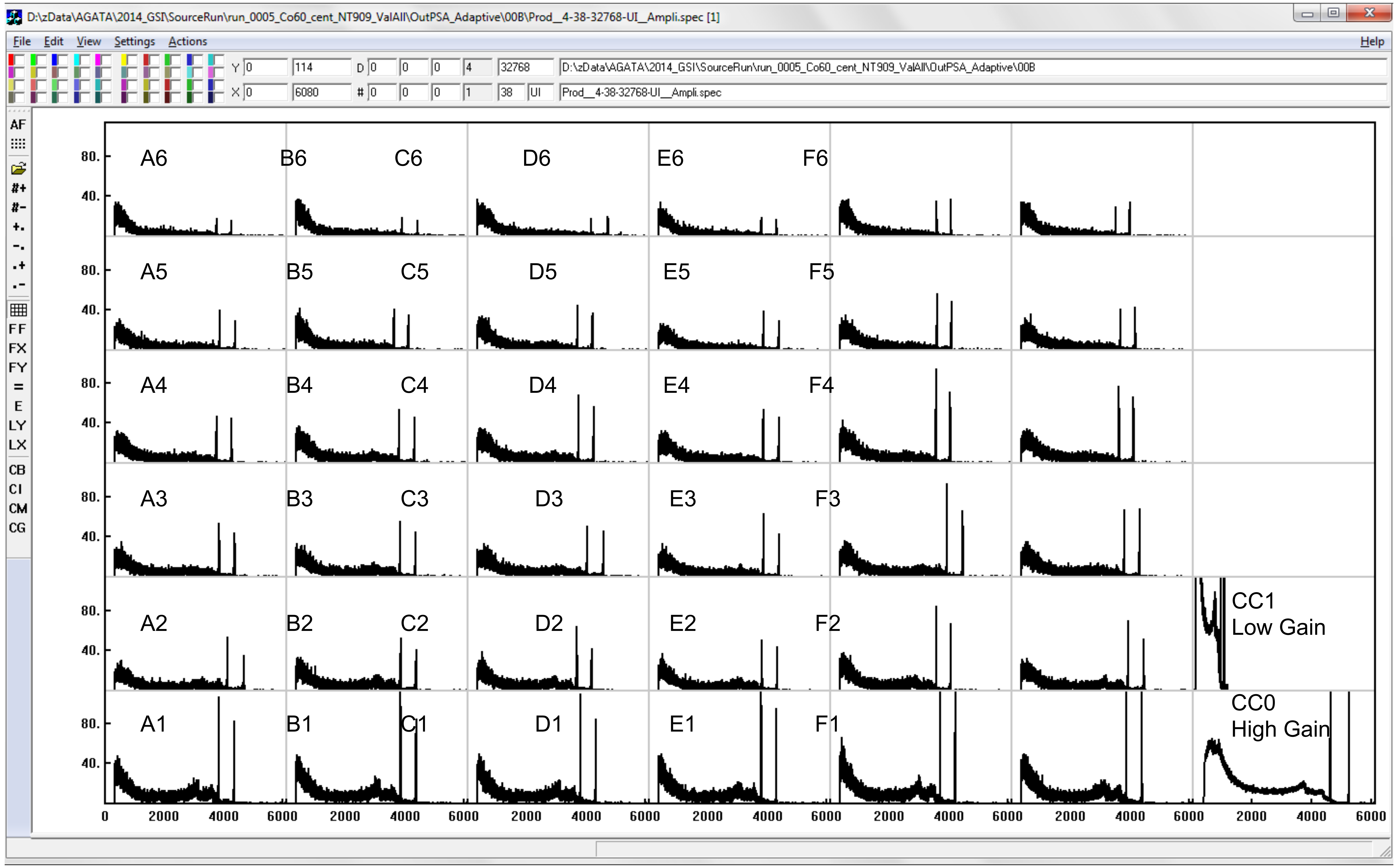
- **PostPSAFilter**
  - Neutron deficit corrections
  - Recalibrations of energy and time
  - *Smearing of positions (not recommended)*

Crystal Producer

What does it read and write:

- Raw event: data from GGP electronics
- Traces length (100 samples of 10ns):
- Write original data (in compressed format)
- data files typically split after 1 M events ~ 3.6 GB
  - event_mezzdata.cdat.0000 event_mezzdata.cdat.0001 …
- Generate raw spectra for amplitudes and baselines
- Format data into an data:crystal adf frame and send it to the data flow.

Prod__4-38-32768-UI__Ampli.spec
[1]-38-32768 used for energy calibrations



PreprocessingFilter

Performs:

- **Energy calibrations** and **cross talk corrections**
- Analysis of traces
  - **Calculation of T0** from the core signal
  - **Time alignment** of the segments
  - Vertical normalization of traces
  - Define the segment multiplicity

- After Preprocessing:
  - energies are stored in units of keV
  - times are in units of samples (10 ns) (but time calibration parameters are in ns)
  - positions are given in mm, when they show up after the PSA

PreprocessingFilter

Calibrations:

- **Energy**
  - **Gain-only, no offset** coefficient needed because of the way the amplitude is generated in the preprocessing electronics.

- **Cross-talk**
  - 36*36=1296 coefficients to correct capacitive coupling correlations between segments and core
  - Used also to recover up to one broken or missing segment per crystal

- **Local Time**
  - 36 coefficients to align segments to the core (**great influence on the performance of PSA)**

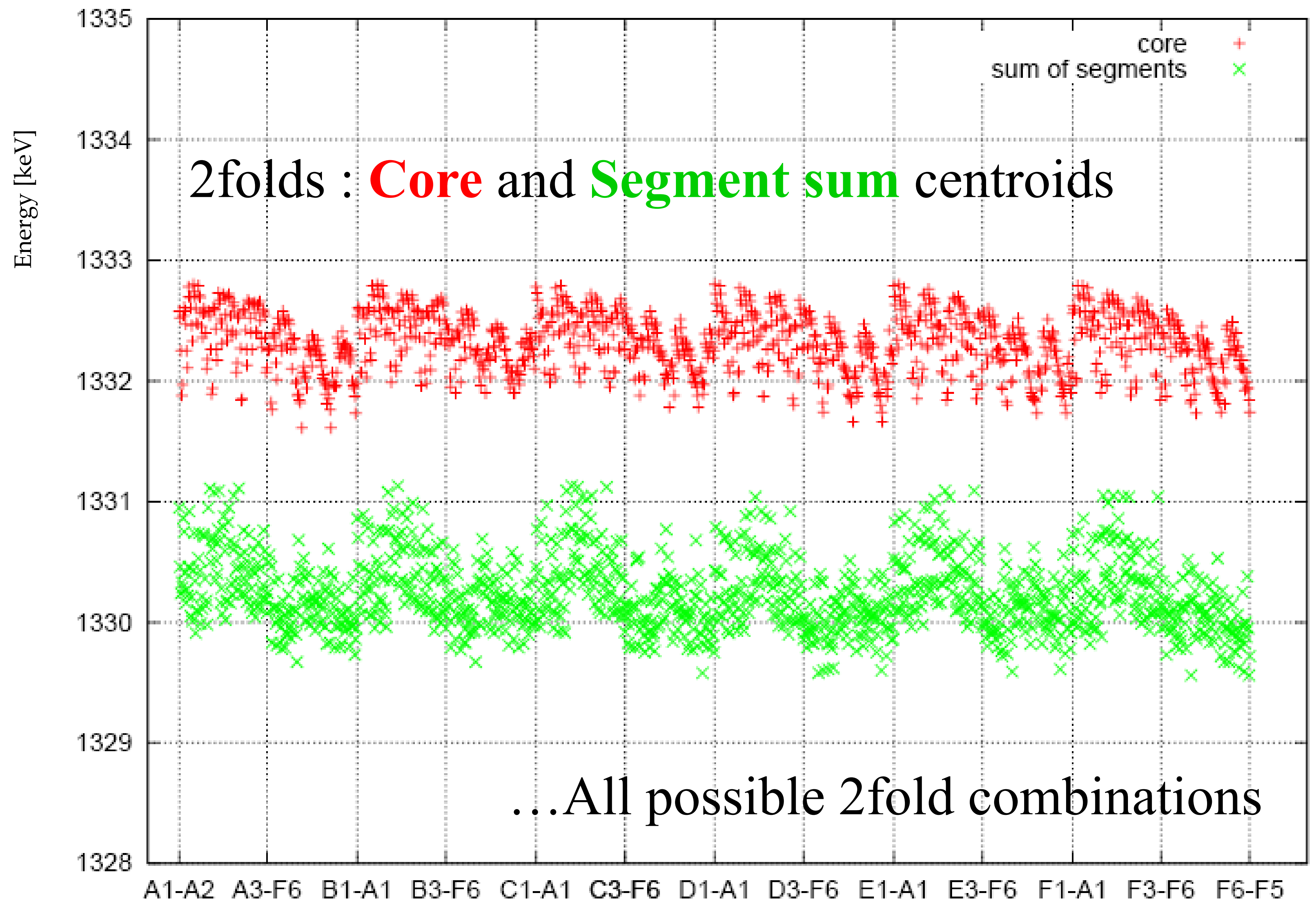
PreprocessingFilterPSA.conf

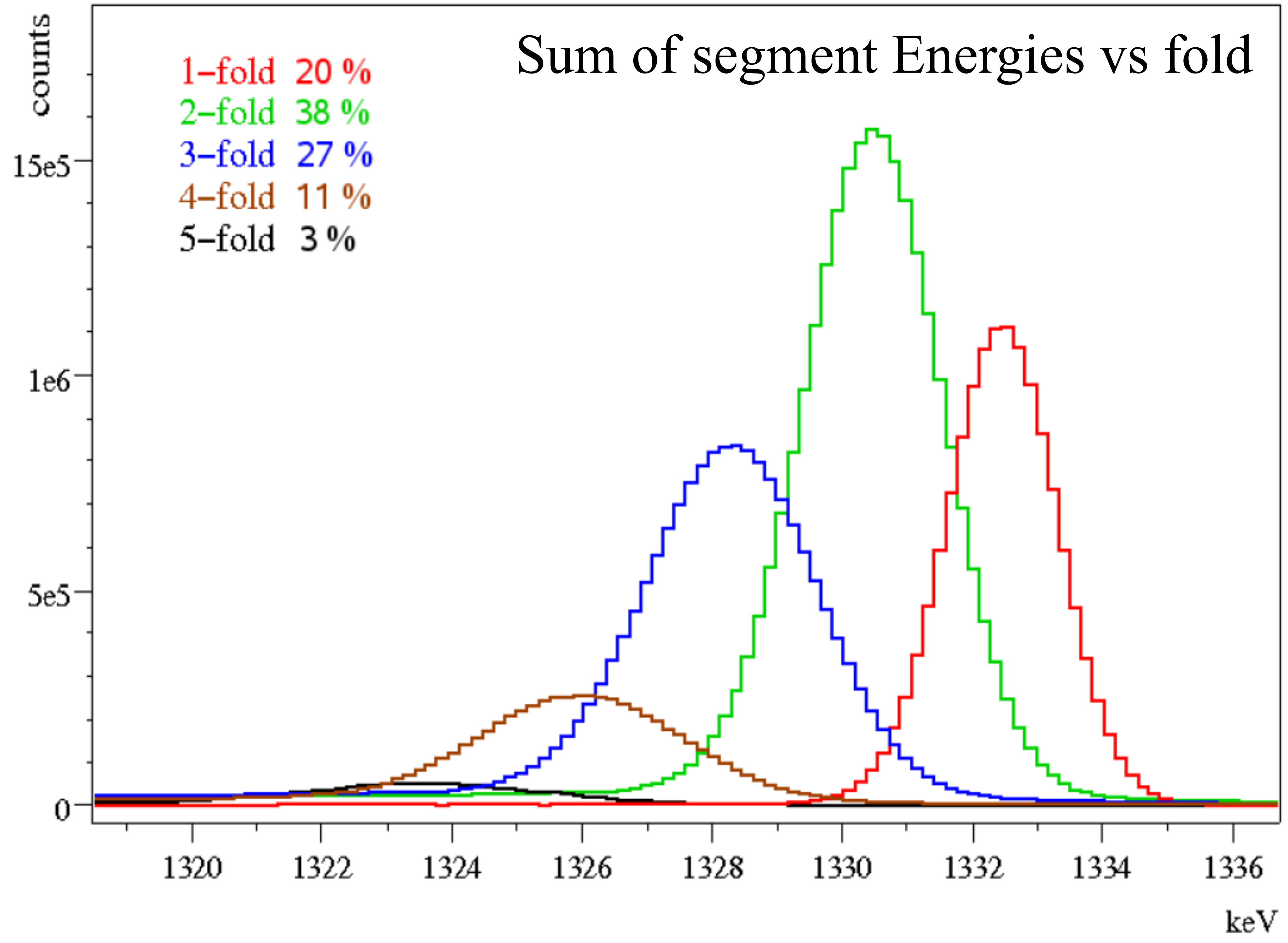
```
#segm/core %d(id) %f(tfall) %f(trise)  %f(egain) %f(emink) %f(tmove)

segm          0     4600      1000      0.162979    10.0      9.891
segm          1     4600      1000      0.170079    10.0     10.463
segm          2     4600      1000      0.163820    10.0     10.361
segm          3     4600      1000      0.169401    10.0      9.485
segm          4     4600      1000      0.158867    10.0      7.971
segm          5     4600      1000      0.155504    10.0     10.077
segm          6     4600      1000      0.170291    10.0      9.050
segm          7     4600      1000      0.165092    10.0      9.263
segm          8     4600      1000      0.145804    10.0      7.420
segm          9     4600      1000      0.168806    10.0      8.448
segm         10     4600      1000      0.143493    10.0      6.188
segm         11     4600      1000      0.159609    10.0     10.510
segm         12     4600      1000      0.153815    10.0     10.251
segm         13     4600      1000      0.155996    10.0      9.448
segm         14     4600      1000      0.168760    10.0      9.537
segm         15     4600      1000      0.175860    10.0      8.866
segm         16     4600      1000      0.185031    10.0     13.873
segm         17     4600      1000      0.157300    10.0     10.564
segm         18     4600      1000      0.169683    10.0      9.836
segm         19     4600      1000      0.168100    10.0      9.683
segm         20     4600      1000      0.170233    10.0      9.677
segm         21     4600      1000      0.174663    10.0      9.472
segm         22     4600      1000      0.174109    10.0      8.942
segm         23     4600      1000      0.165021    10.0     11.498
segm         24     4600      1000      0.152862    10.0     10.267
segm         25     4600      1000      0.169911    10.0     11.067
segm         26     4600      1000      0.165142    10.0      9.910
segm         27     4600      1000      0.159595    10.0      9.393
segm         28     4600      1000      0.168353    10.0      8.347
segm         29     4600      1000      0.167807    10.0     11.519
segm         30     4600      1000      0.163006    10.0      9.945
segm         31     4600      1000      0.159887    10.0     10.383
segm         32     4600      1000      0.155449    10.0      9.600
segm         33     4600      1000      0.143345    10.0      8.487
segm         34     4600      1000      0.150043    10.0      7.216
segm         35     4600      1000      0.176351    10.0     12.197
core          0     4200      1000      0.347055    20.0     25.000
core          1     4200      1000      0.069358    20.0     -0.285
tntf 2097152
```

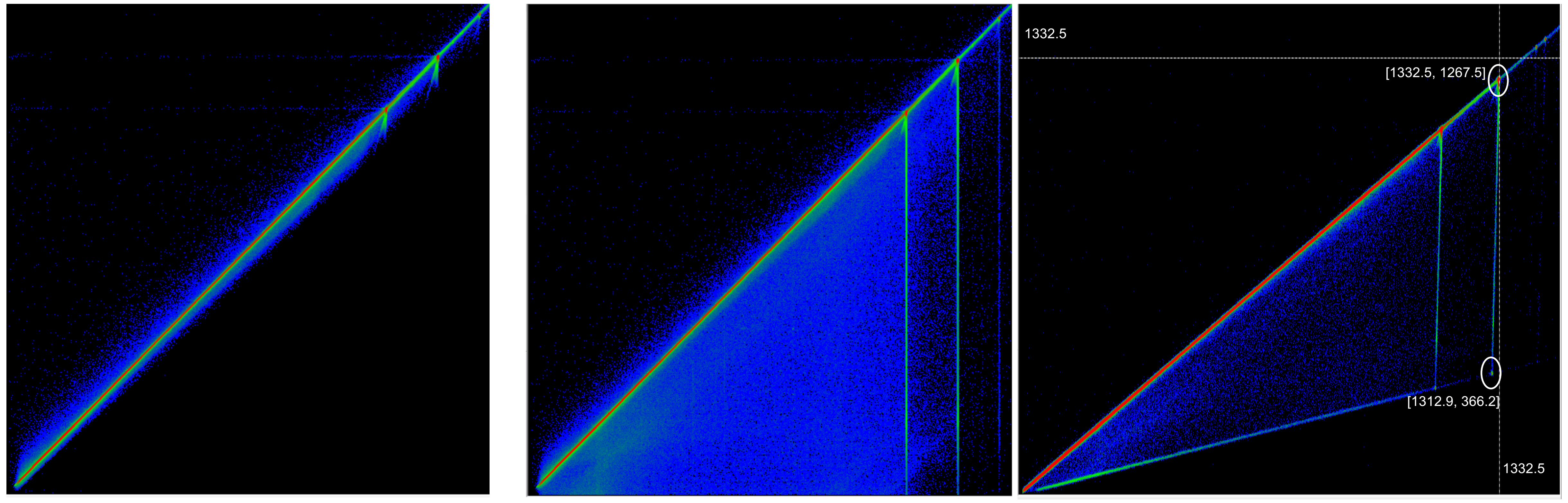
Cross-Talk correction: Motivations

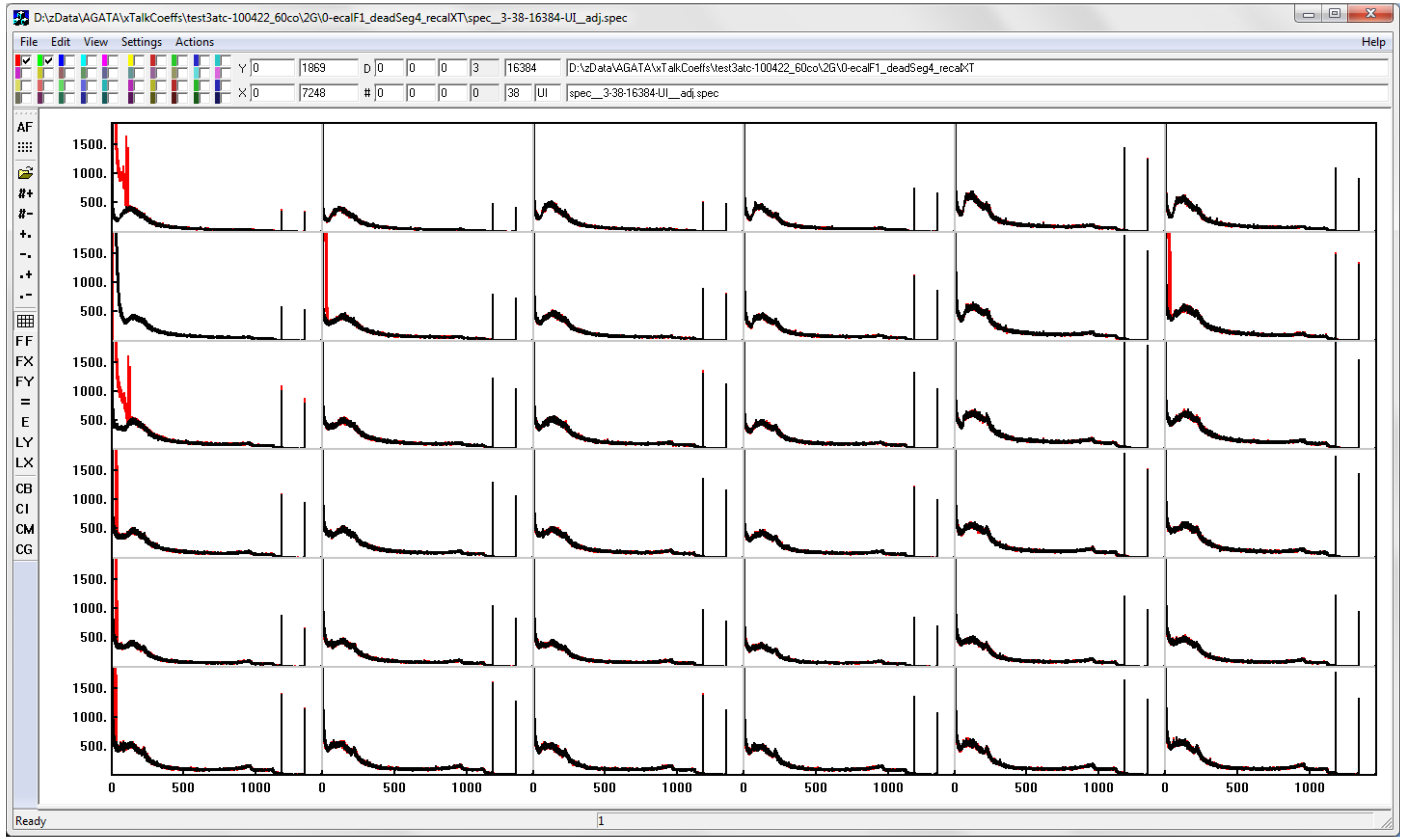
- Crosstalk is present in **any segmented detector**
- Creates strong energy shifts proportional to fold
- Tracking needs segment energies !





Missing/Broken segments





- Exploiting the redundancy:

$$\sum Segs == Core$$

- Compensate loss of energy in core
- Assign energy to missing segment

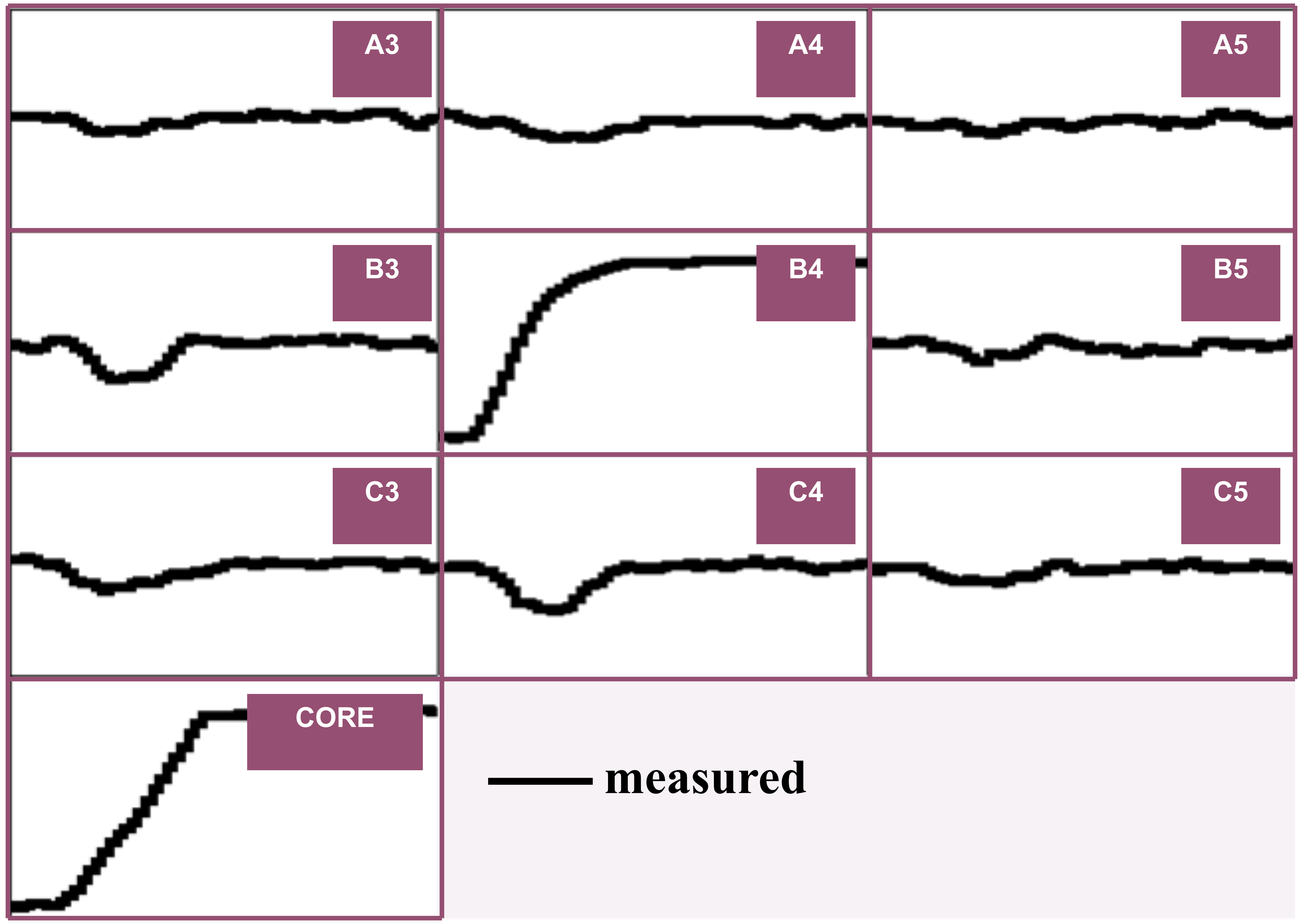
$$E_{missing} = E_{core} - Sum_{Other}$$

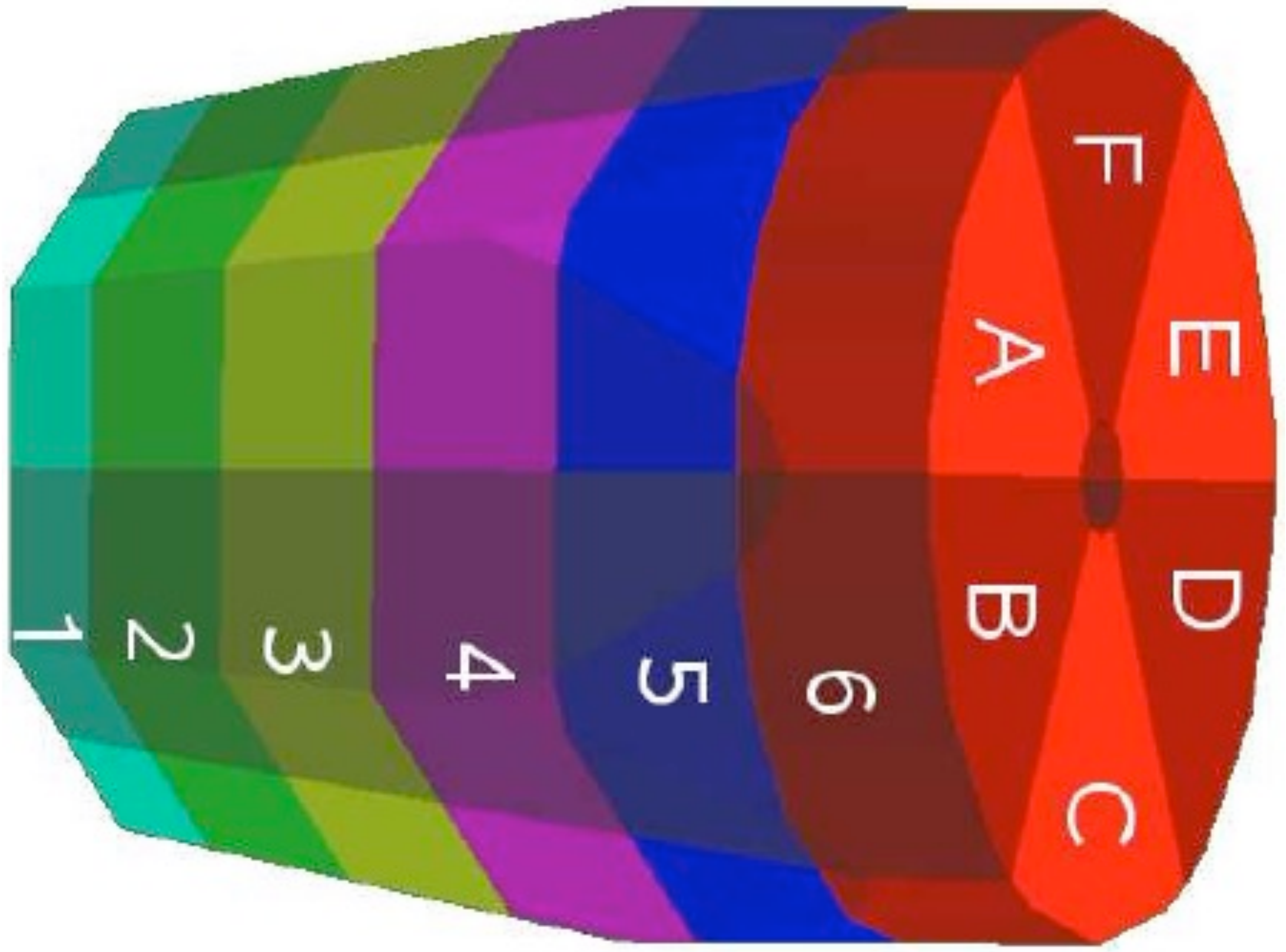
PSAFilter

Performs:

- Read the simulated basis (ADL)
- Apply the preamp response function to the simulated traces
- **Perform the signal decomposition:**
  - Implemented algorithm is the Grid Search
    - As a full grid search
    - As a coarse/fine search (AGS)
- **Reduces size of data by factor ~80**
- Provides the parameters for the correction of neutron damage
- **Takes ~95 % of total CPU time**
- Is the critical point for the processing speed of online and offline analyses

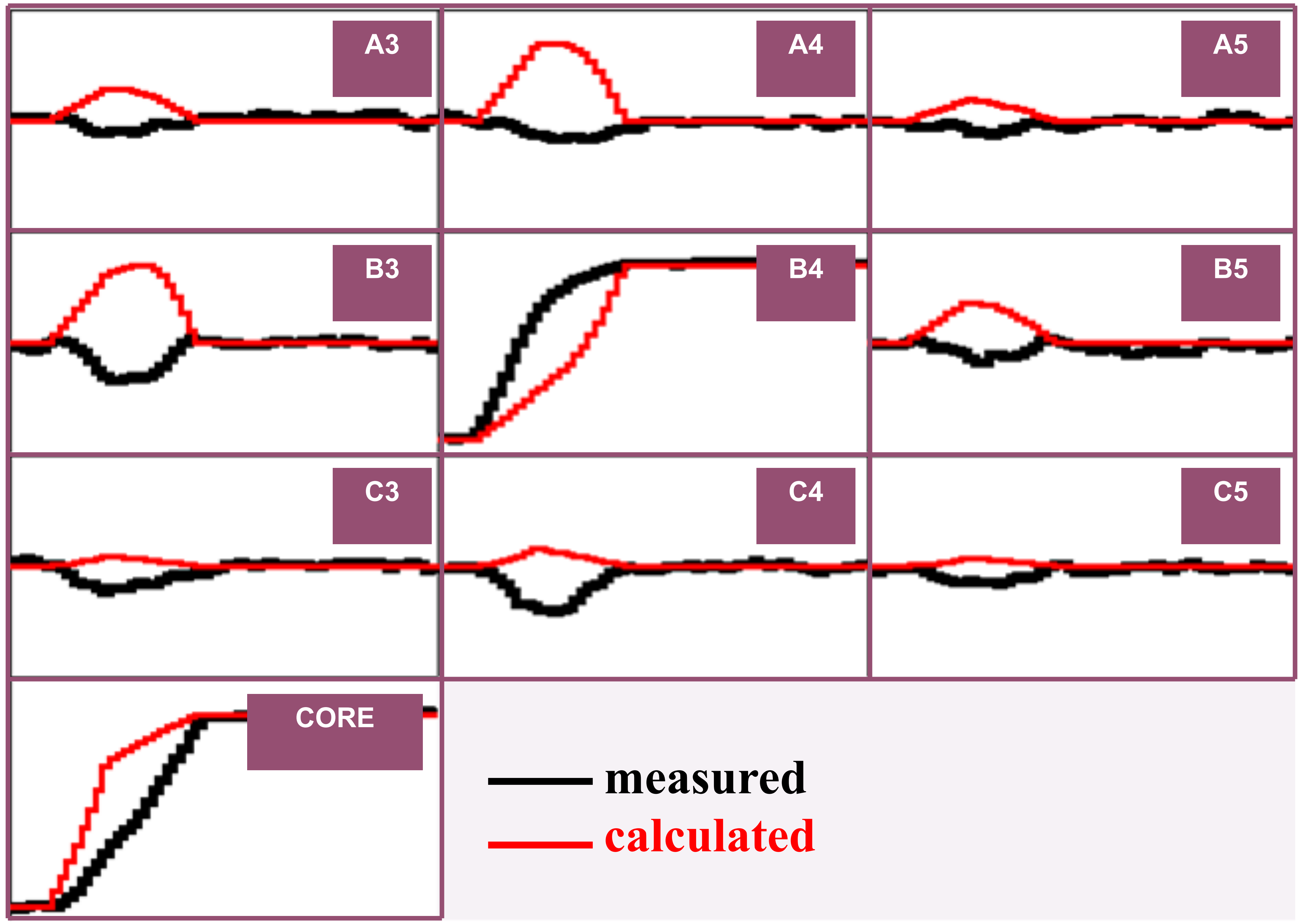
Pulse Shape Analysis concept

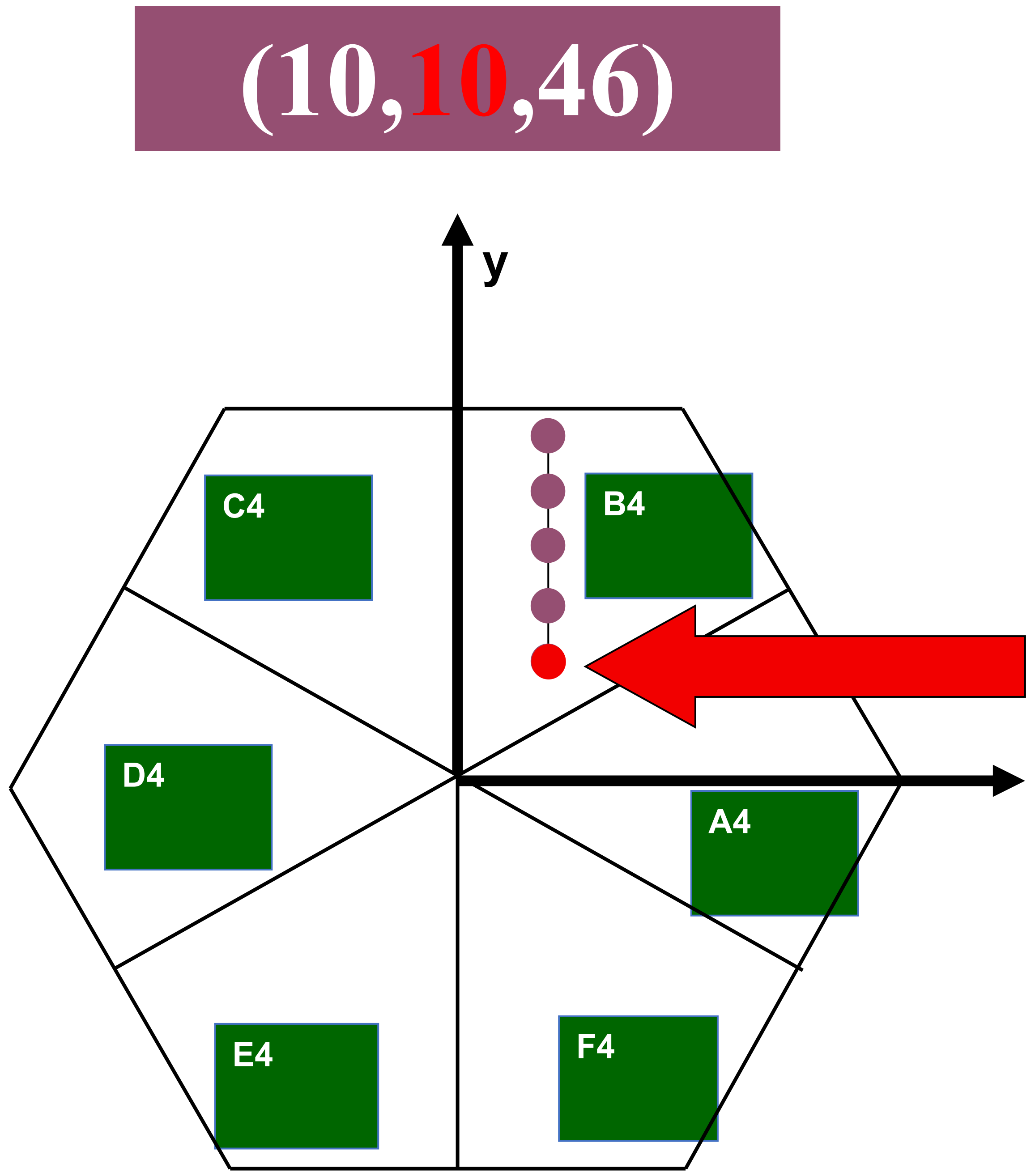




791 keV deposited in segment B4

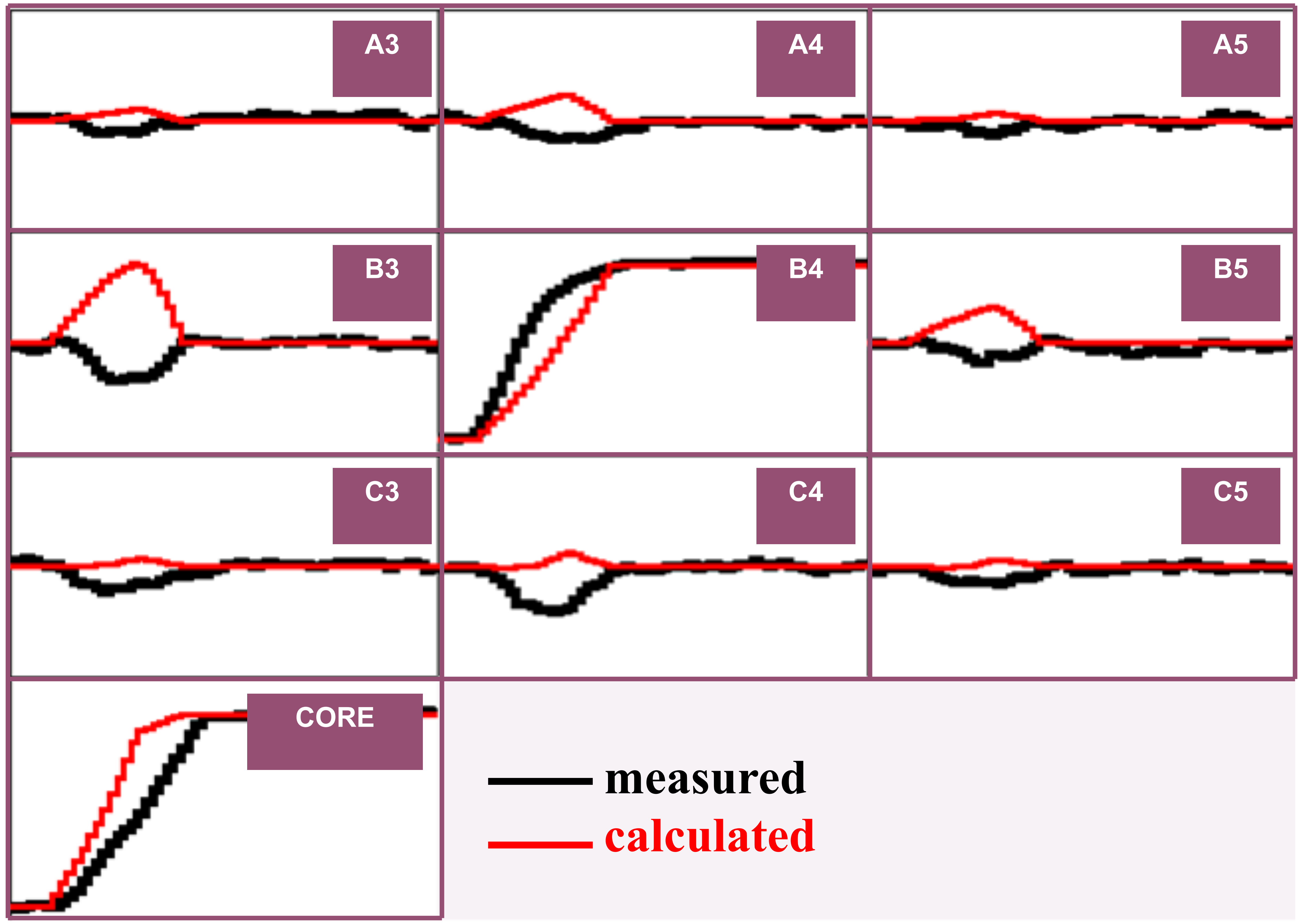
Pulse Shape Analysis concept

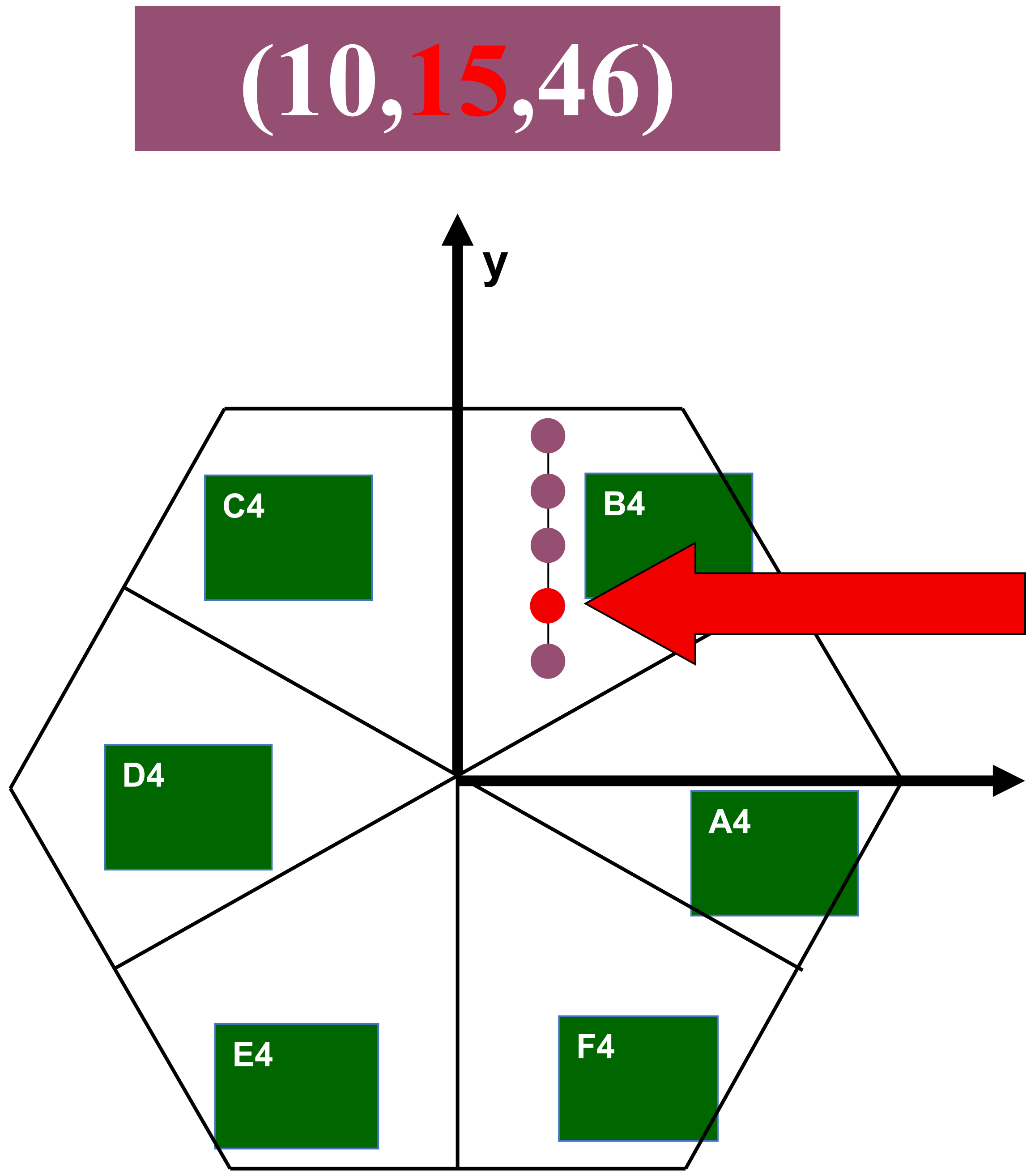




791 keV deposited in segment B4

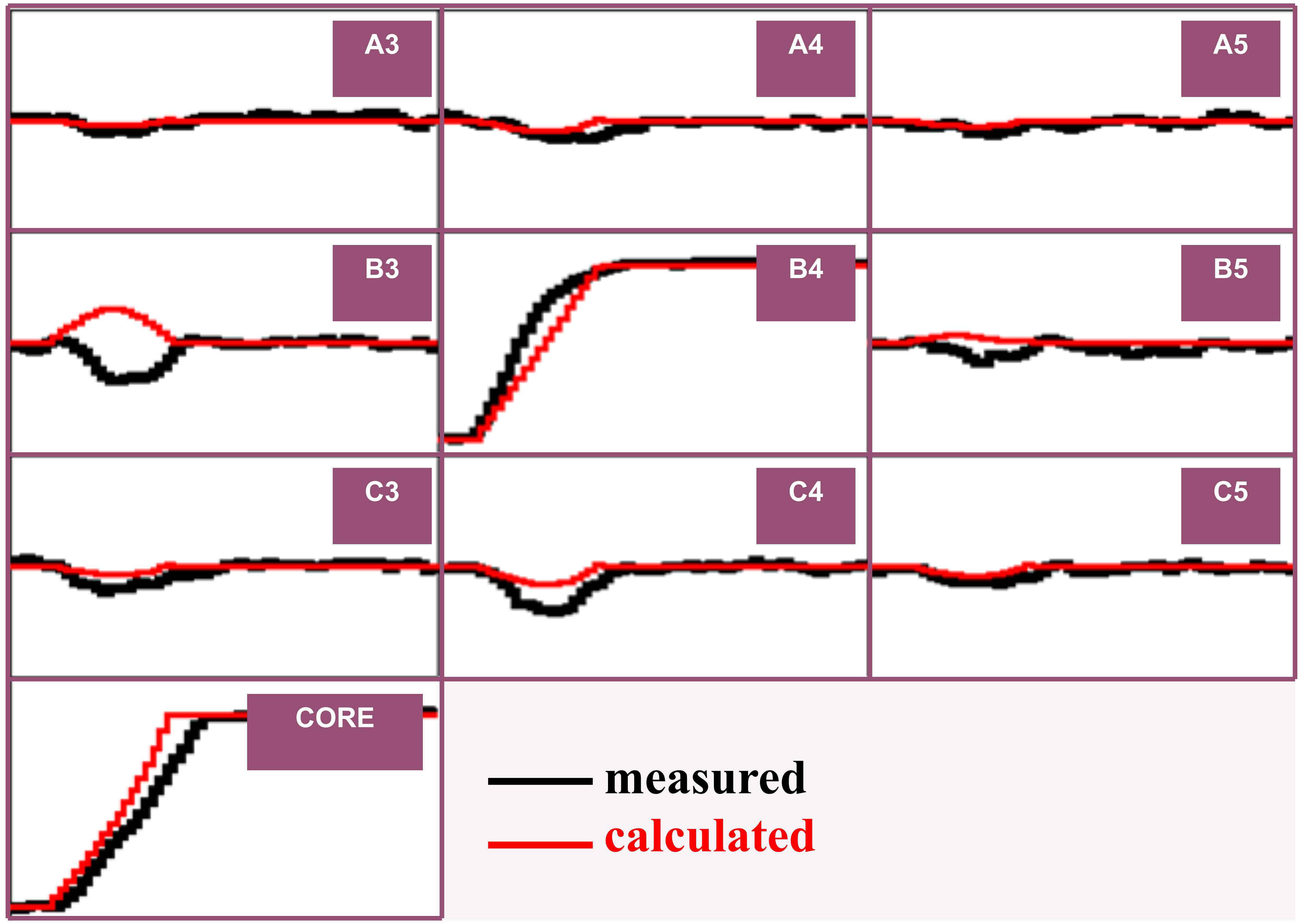
Pulse Shape Analysis concept

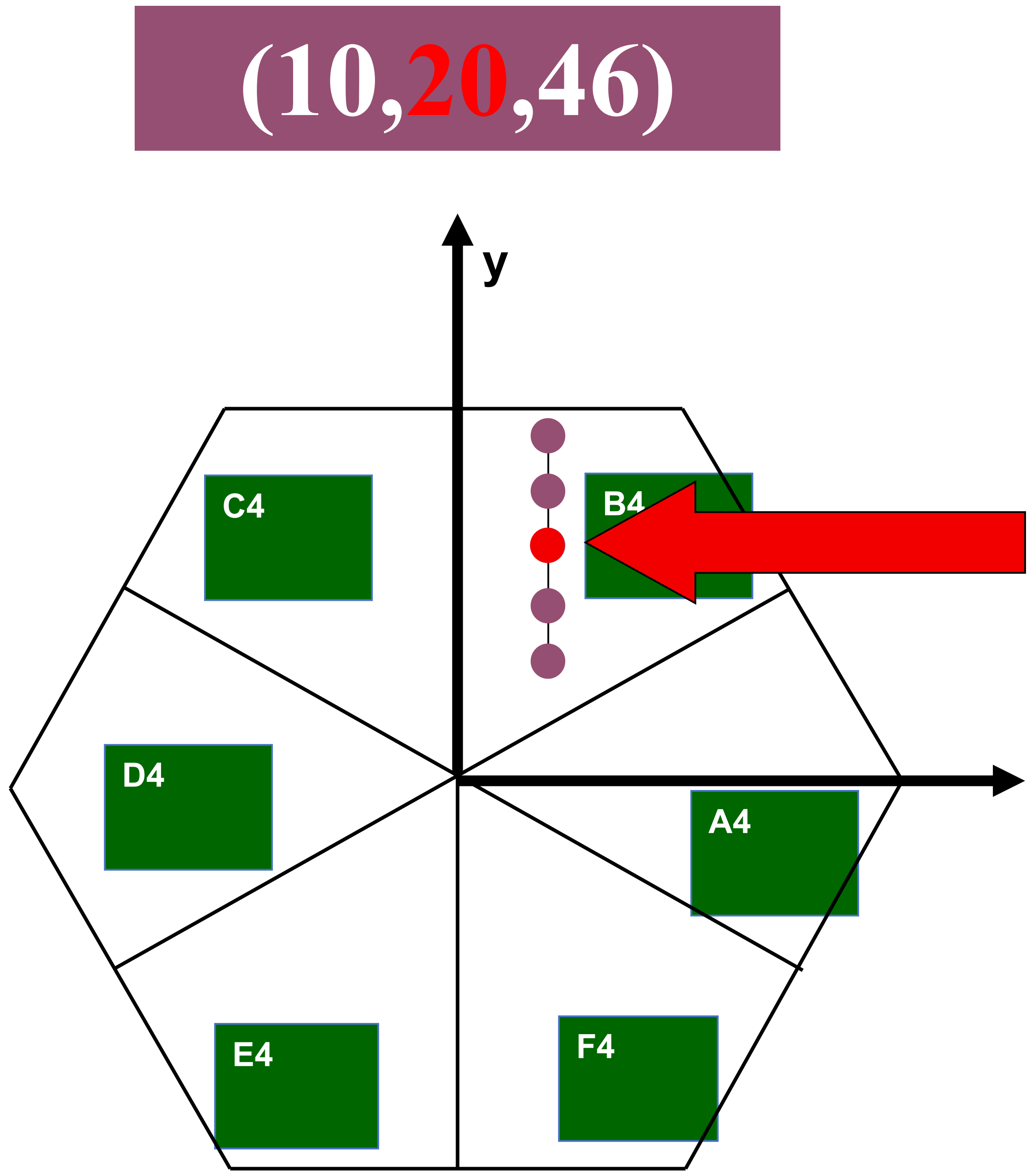




791 keV deposited in segment B4

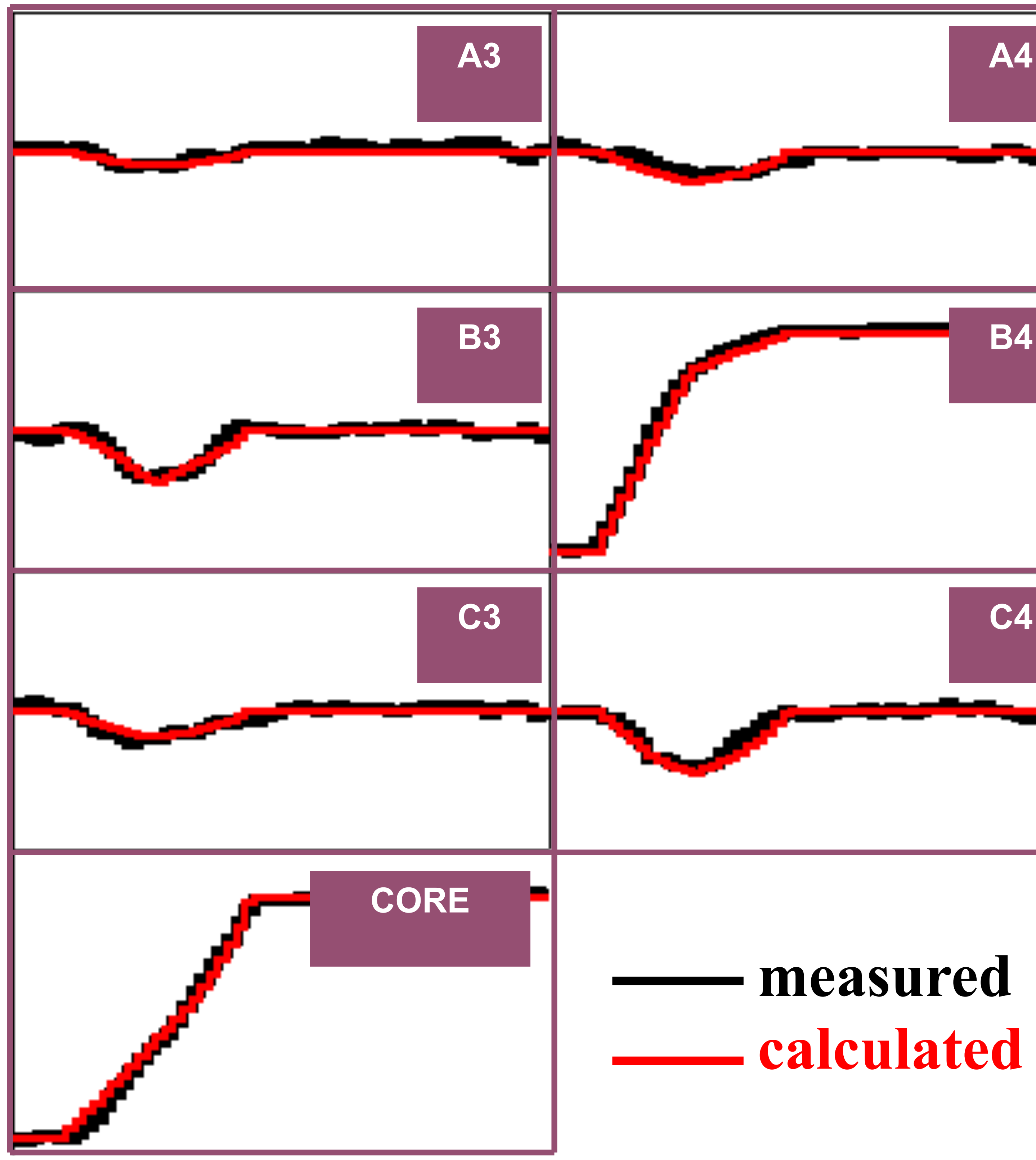
Pulse Shape Analysis concept



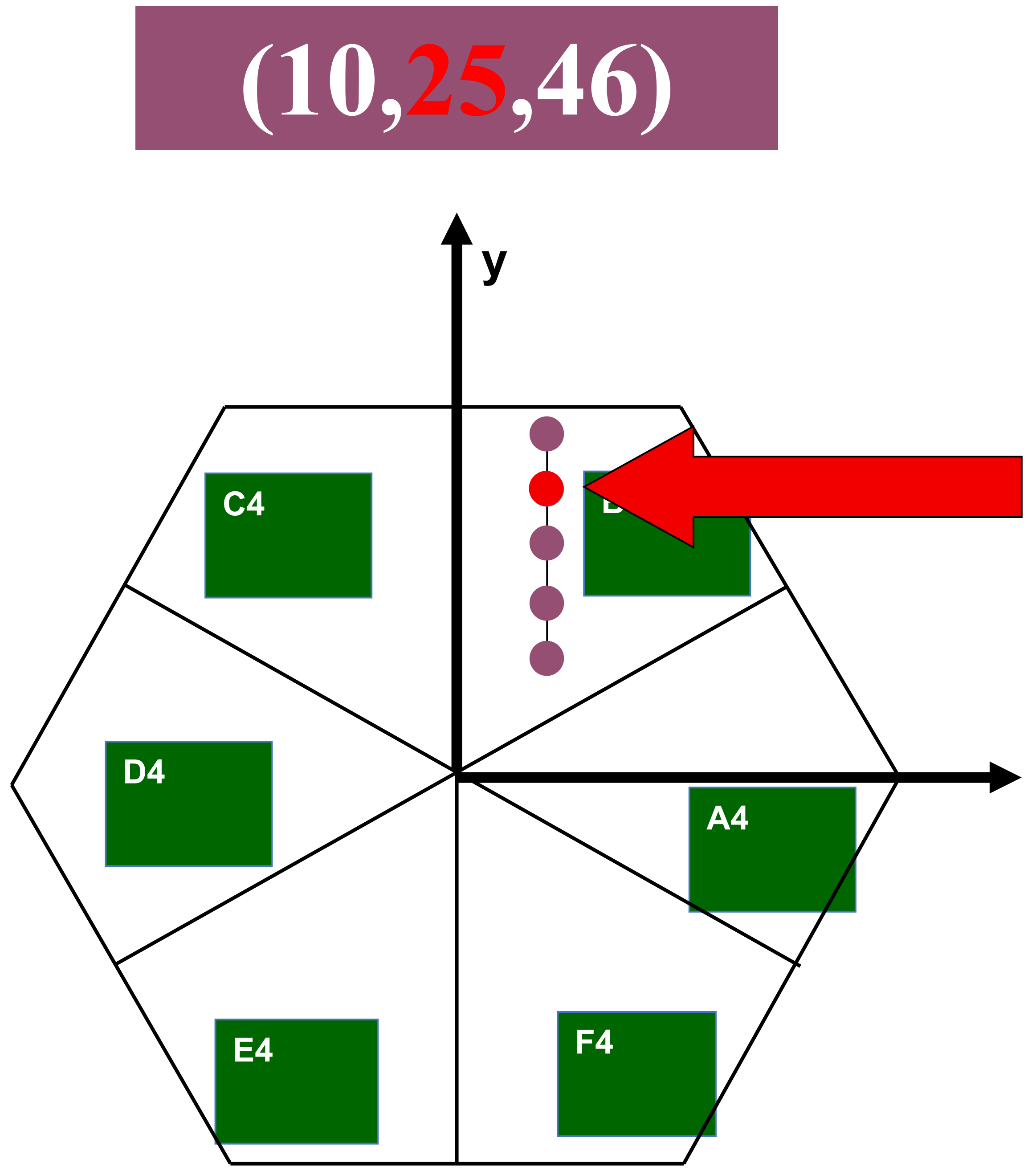


791 keV deposited in segment B4

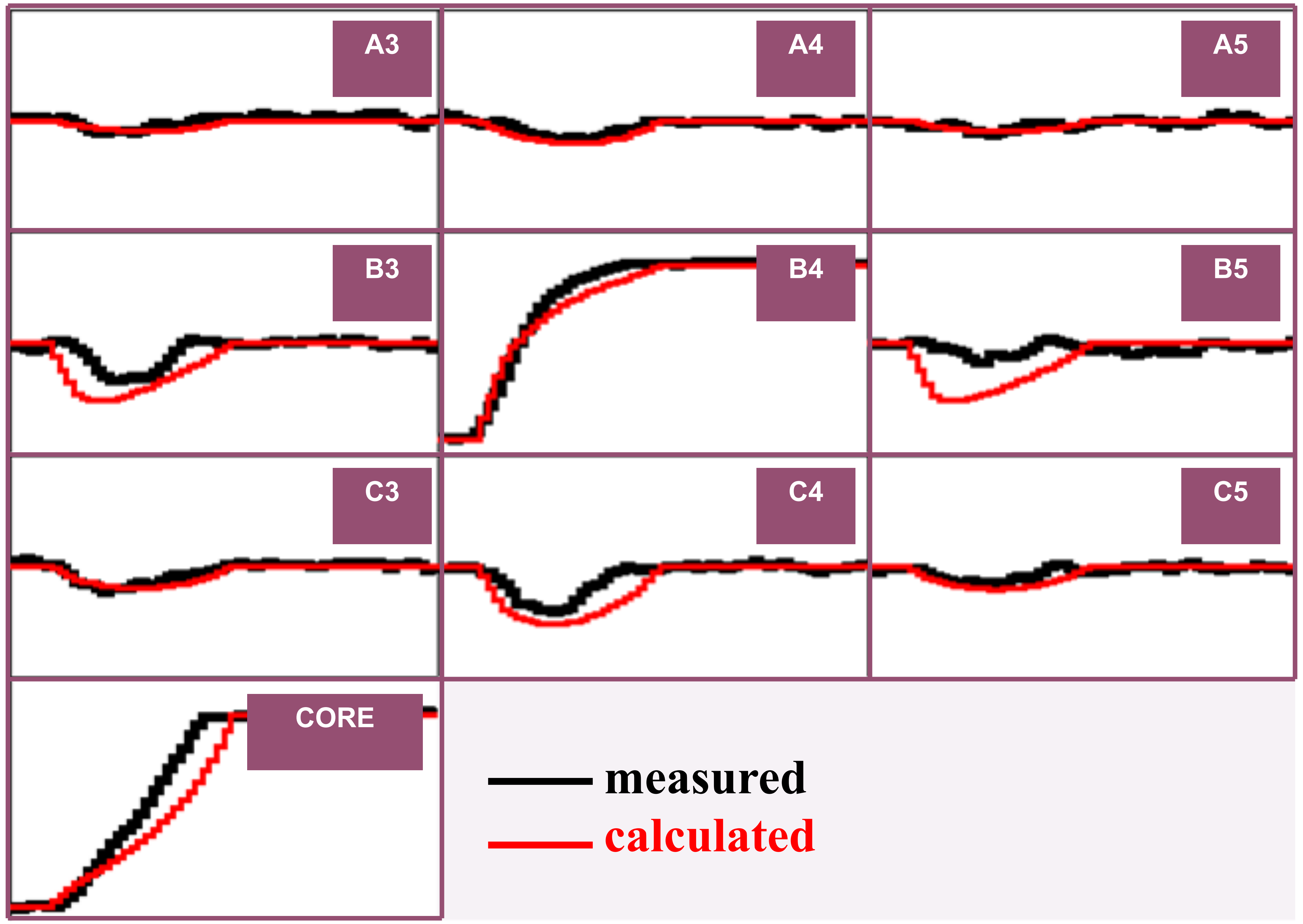
Pulse Shape Analysis concept

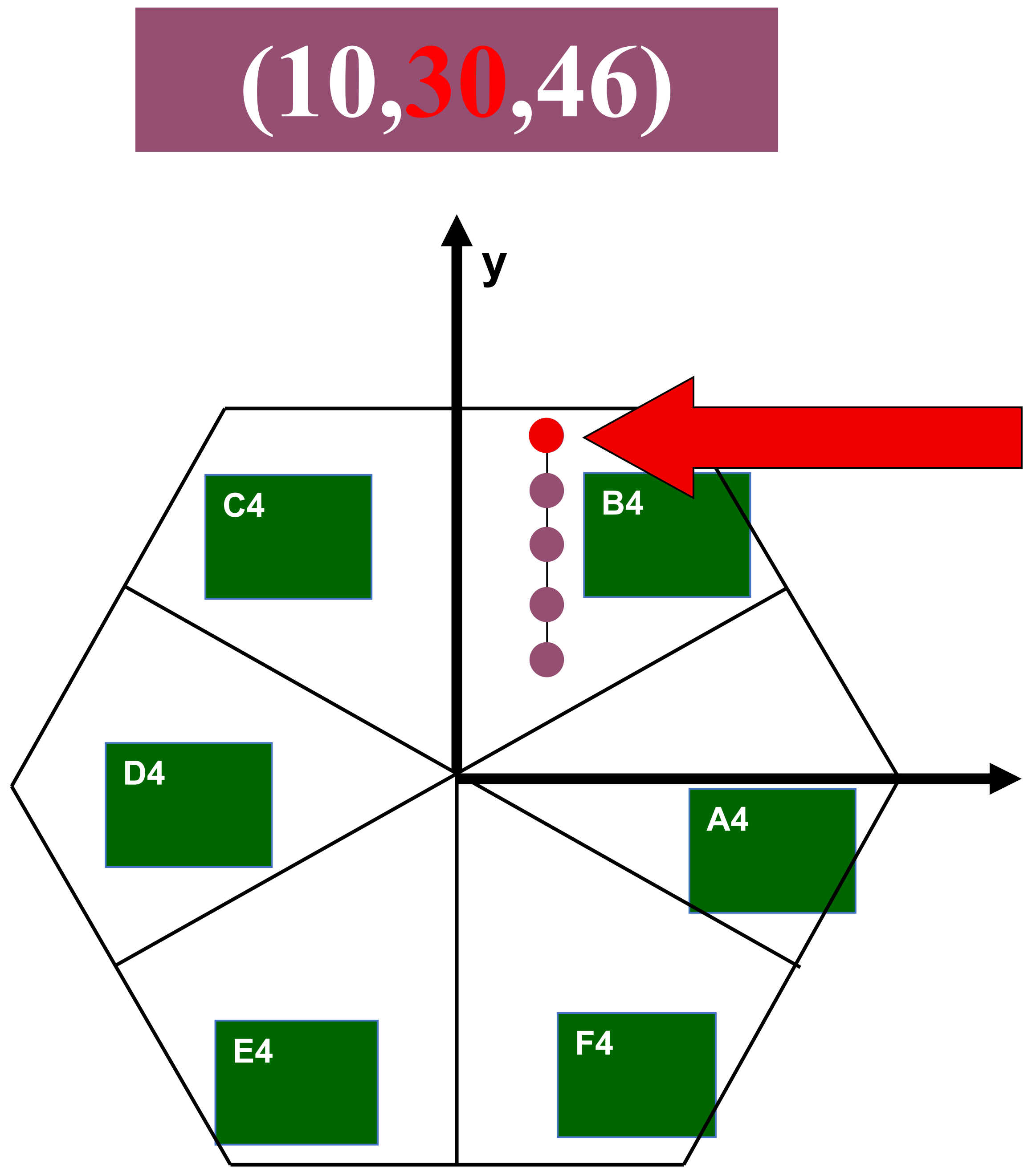


791 keV deposited in segment B4



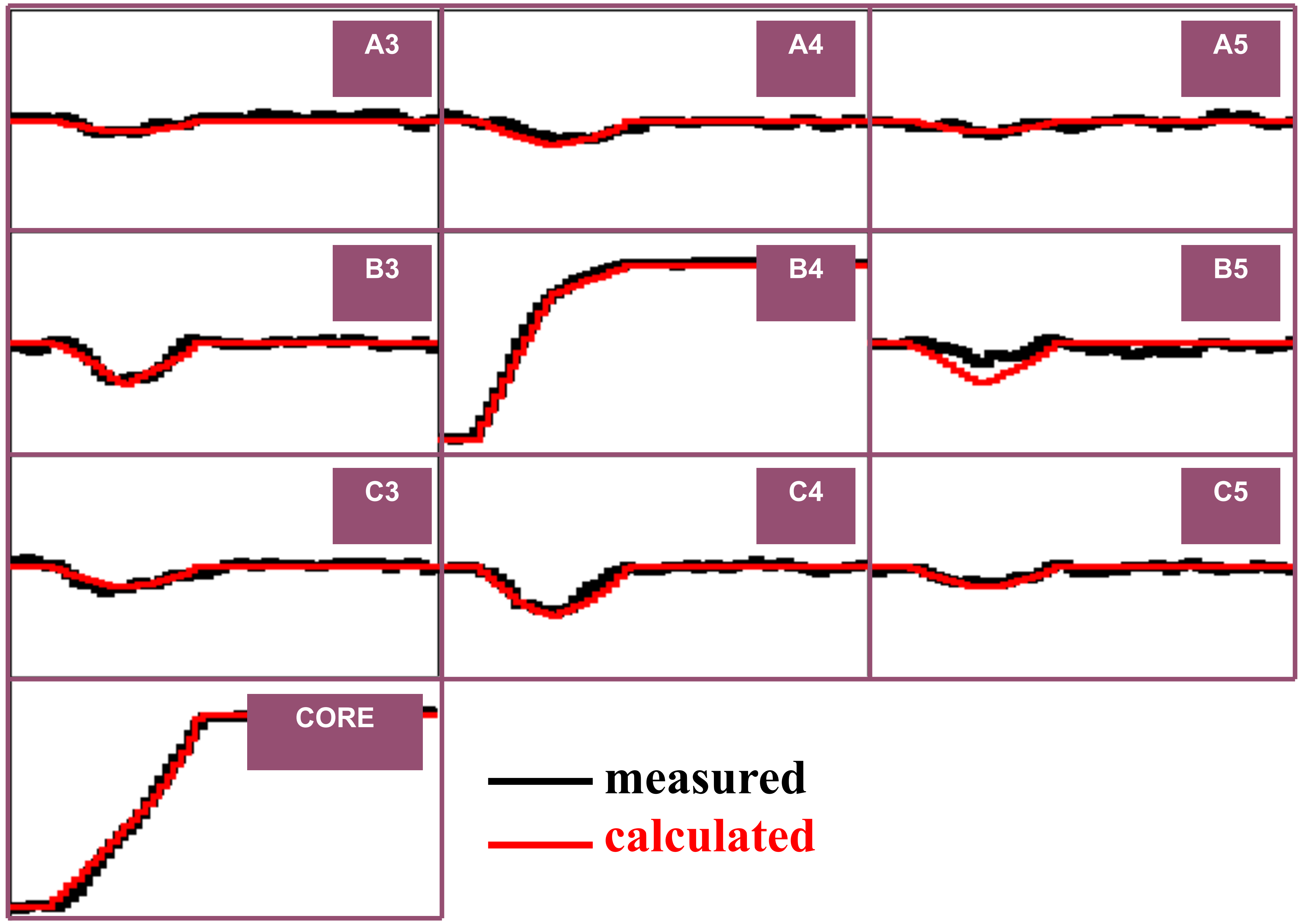
Pulse Shape Analysis concept





791 keV deposited in segment B4

Pulse Shape Analysis concept

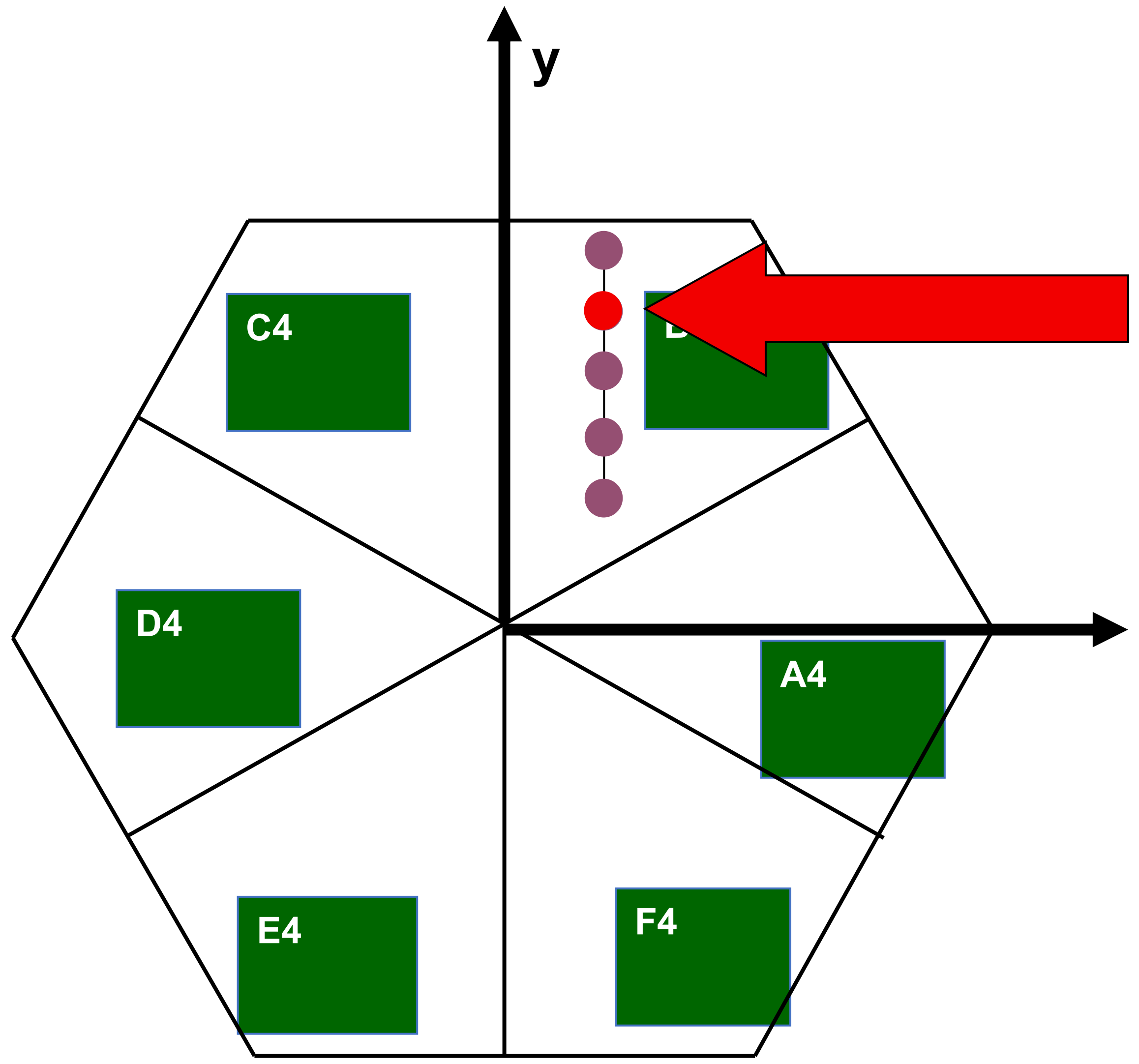


measured
calculated

791 keV deposited in segment B4

Result of
Grid Search
Algorithm

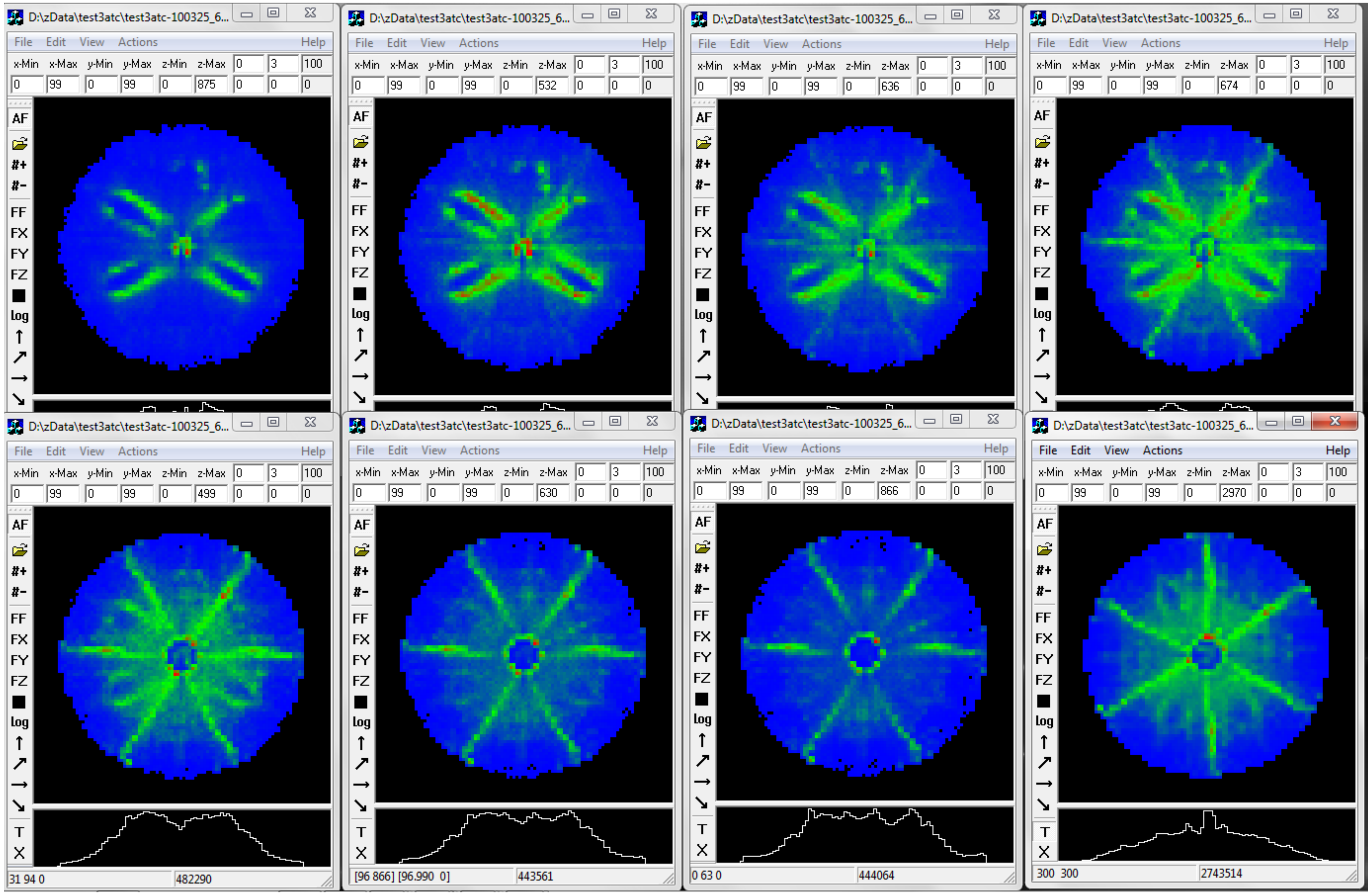
(10,25,46)



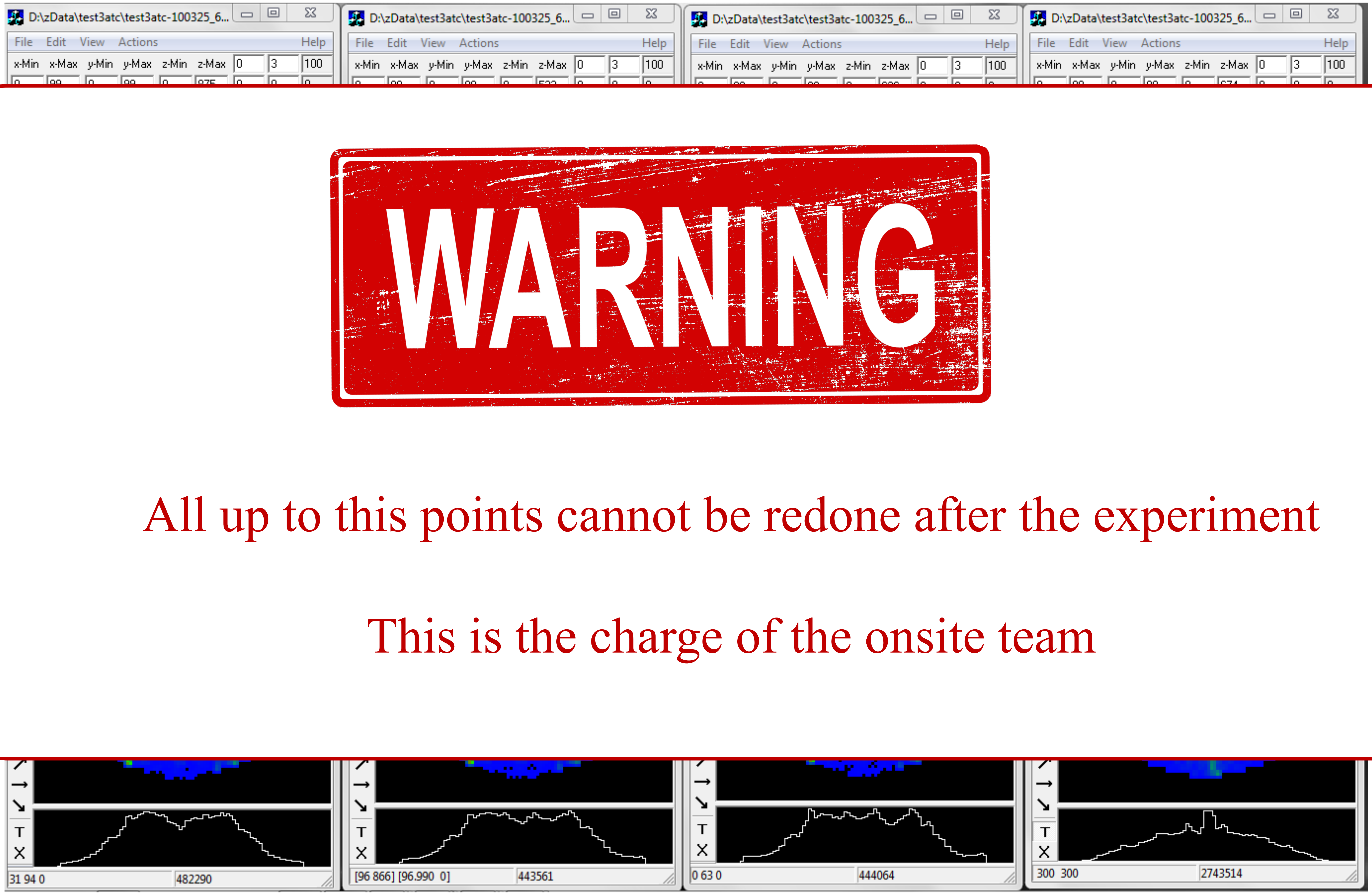
The Grid Search algorithm

- Signal decomposition assumes **one interaction per segment**
- The decomposition uses the transients and a differentiated version of the net charge pulse
- **Proportional** and **differential cross-talk** are included using the xTalk coefficients of the preprocessing
- The minimum energy of the "hit" segments is a parameter in the PreprocessingFilter → 10 keV
- No limit to the number of fired segments (i.e. up to 36)
- The number of used neighbours is a compile time parameter (usually 2 as Manhattan distance)
- The algorithm cycles through the segments in order of decreasing energy:
  → the result of the decomposition is removed from the remaining signal
- Using ADL bases (Bart Bruyneel)
- Using 2 mm grids: → ~ **48000 grid points in a crystal** (700-2000 points/segments)
- Speed is:
  → ~ 150 events/s/core for the Full Grid Search
  → ~ 1000 events/s/core for the Adaptive Grid Search
- To speedup execution, **parallelism has been implemented** with
  → blocks of ~ 300 events passed to 5 parallel threads of execution

Effect of time alignment



Effect of time alignment



WARNING

All up to this points cannot be redone after the experiment

This is the charge of the onsite team

PostPSAFilter

Performs:

- Final energy calibrations of cores and segments (with offset)
- Force segments to core (optional)
- Recovery of neutron damage
- Global time alignment of crystals → important to reduce random coincidences

Global actors

- **Event Builder**
  - Merge all AGATA crystal events on the basis of timestamps
- **Event Merger**
  - Merge AGATA built events with ancillaries
- **Tracking Filter**
  - Apply the tracking algorithm using segments energies and positions
- **Tree Builder**
  - Dump the data on disk using ROOT tree format

AGATA user guides

AGATA Data Analysis User's guide for Local Level Processing

AGATA Data Analysis Team

September 20, 2019

This document provides a guide to help the users analyzing the AGATA data produced at the local level processing i.e. before any building of events. It includes energy calibrations, time alignments, cross talk corrections and any other corrections to improve the quality of the data. Criteria for bad events rejection are also highlighted.

11/2018

AGATA Data Analysis User's guide Global Level Processing

AGATA Data Analysis Team

This document provides a guide to help the users analyzing the AGATA data produced at the global level processing i.e. for building or merging events and store root Trees.

https://atrium.in2p3.fr/8f552ad6-188b-401a-b72b-1fd7bab12351

Thank you !
And have fun with your data analysis 😉