CBM-TRD Feature Extraction

CBM-TRD TDR Review

SPONSORED BY THE

Federal Ministry of Education and Research

05P15RFFC1

Cruz de Jesus García Chávez
garcia@iri.uni-frankfurt.de

IRI - Goethe University
Frankfurt am Main
Prof. Dr. Udo Kebschull

The TRD Readout Concept

Main components:

- **Front-End Electronics (FEE)**
  - Conversion of TRD detector signals into discrete messages
  - Self-triggered
- **Readout ConTroller (RCT)**
  - Interface between FEE and data taking system
  - Communication to the Experiment Control System (ECS)
  - Timing and Flow Control system (TFC)
  - Feature Extraction
- **First-Level Event Selection (FLES)**

- Communication between the blocks is made by different data transport links (depending on the hardware used)

Schematic view of the TRD readout chain

The TRD Readout Proposal

- Based on the SPADIC v2.0 FEE
- AFCK as Readout controller

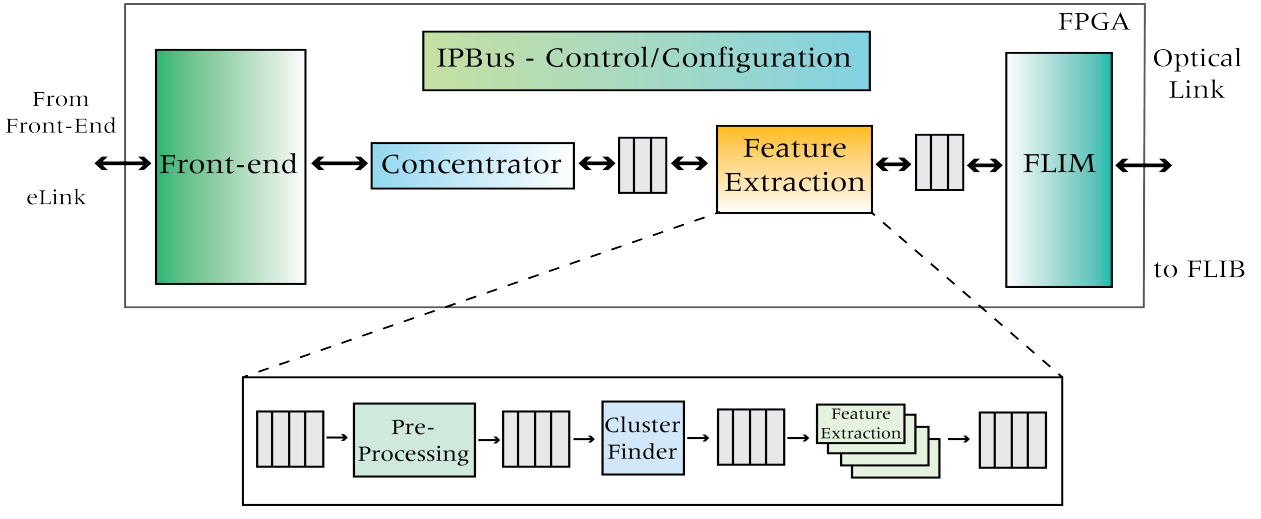
Data transport links:

- Between SPADIC 2.0 and AFCK → GBTx e-Link
- Between AFCK and FLIB/FLES → FLIM
  - Provides a data rate of 10GB/s
  - Data and control transport

TRD readout chain using the SPADIC v2.x ASIC and AFCK boards

The TRD Readout Proposal

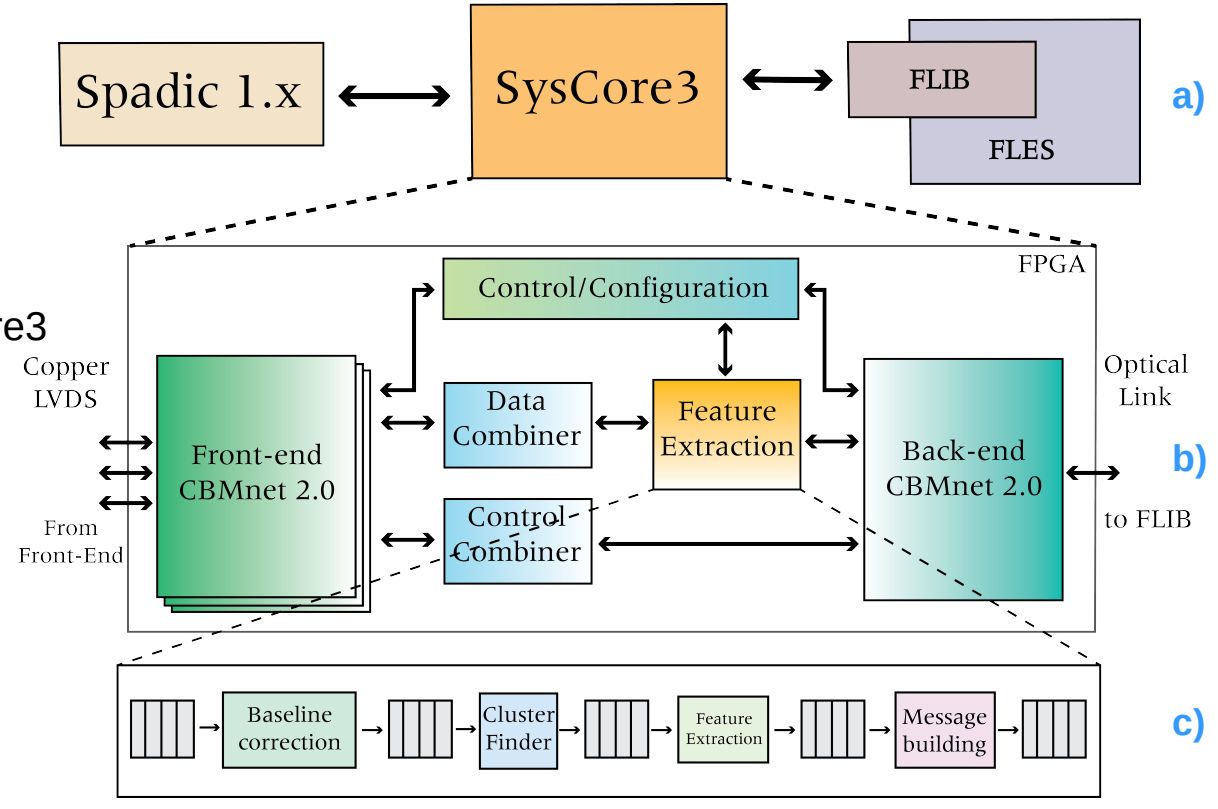
- Front-End module
  GBTx e-Link logic
- Concentrator
  Gathers data streams from multiple Front-ends
- Feature Extraction
  Data pre-processing stage
- FLIM
  Data transport link to first level event selector
- IPBUS
  Control and configuration communication protocol



Detailed view of the AFCK firmware with a feature extraction stage. Front-end based on GBTx and the Back-end based on FLIM.

The TRD Readout Prototype

- Front-end electronics based on SPADIC 1.x ASIC
- Readout controller implemented on a SysCore3 FPGA development platform
  - Xilinx FPGA Spartan 6
  - Up to 3 SPADIC 1.x FEEs per SysCore3
- CBMnet 2.0 data transfer link
  - Used between FEE, RCT and FLIB
  - Data transfer rate of 4.8 Gbps
- Feature extraction stage implemented in the SysCore3
  Algorithms:
  - Total charge
  - Time-over-threshold
  - Centre of gravity



a) *TRD readout chain using the SPADIC v1.x ASIC and SysCore3 boards*

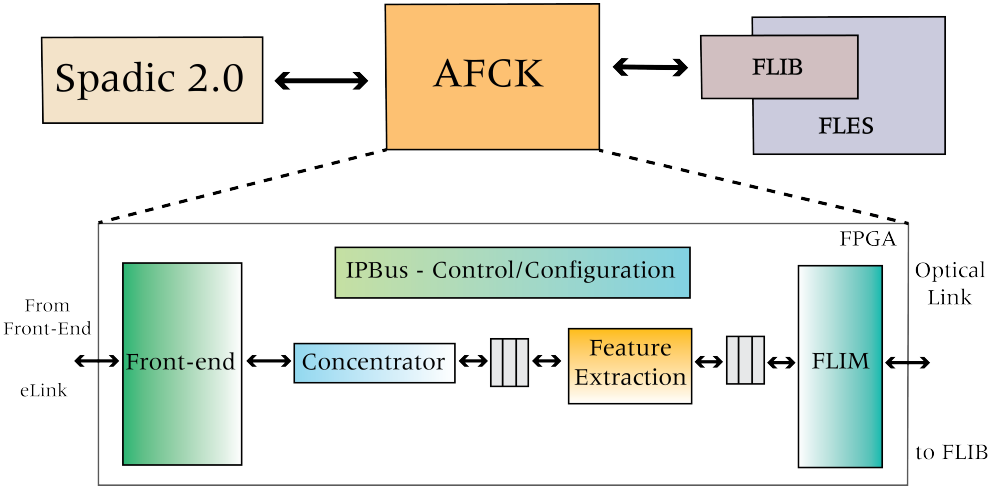
b) Detailed view of the SysCore3 firmware

c) Detailed view of the feature extraction stage

Feature Extraction

Goals

- Data processing stage
- Deliver data merged based on hit-time and bandwidth reduced to the FLES
- Find and extract regions of interest from SPADIC hit messages

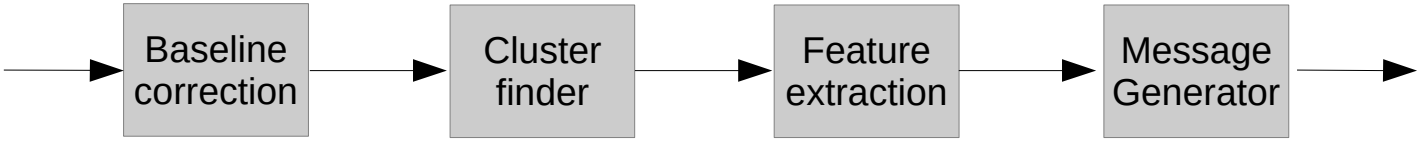


The TRD readout proposal with a detailed view of the AFCK firmware

Feature extraction requirements for CBM-TRD

Signal processing algorithms:

- Baseline correction
- Cluster finder
- Feature extraction from found clusters
- Message packing for transport to FLIB/FLES

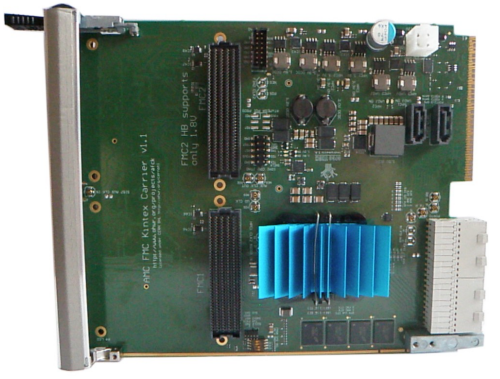


Feature extraction algorithm structure implemented in the AFCK

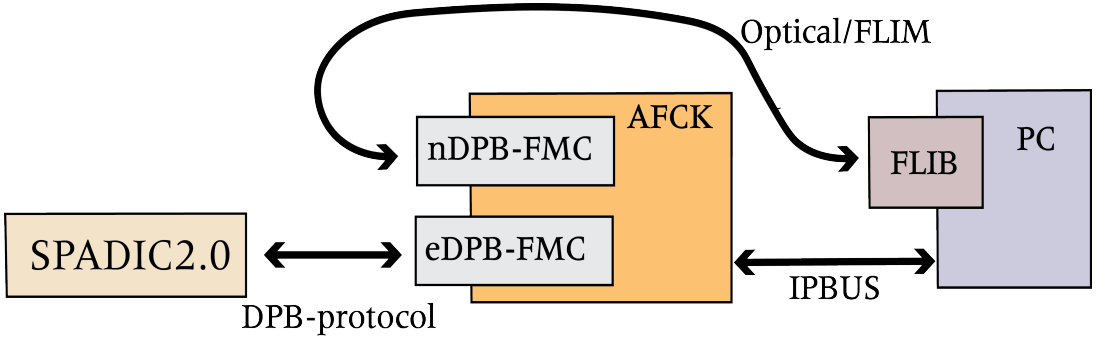
The TRD Readout Proposal at CERN-SPS 2016 testbeam

Overview

- AFCK installed in a micro TCA
- Configuration of the AFCK and SPADIC 2.0 by IPBUS
- Interface SPADIC 2.0 to AFCK by eDPB-FMC on FMC2
- Interface AFCK to FLIB by nDPB-FMC on FMC1

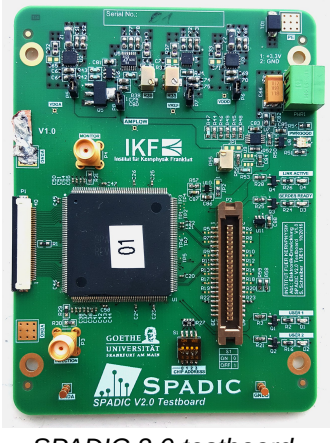


AFCK

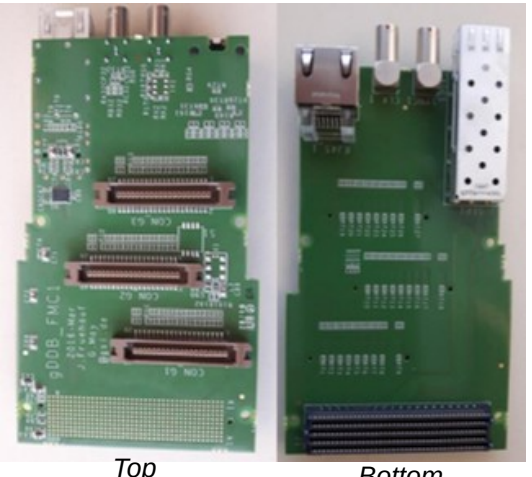


TRD readout chain using the SPADIC v2.x ASIC and AFCK boards. Setup at CERN-SPS 2016 testbeam

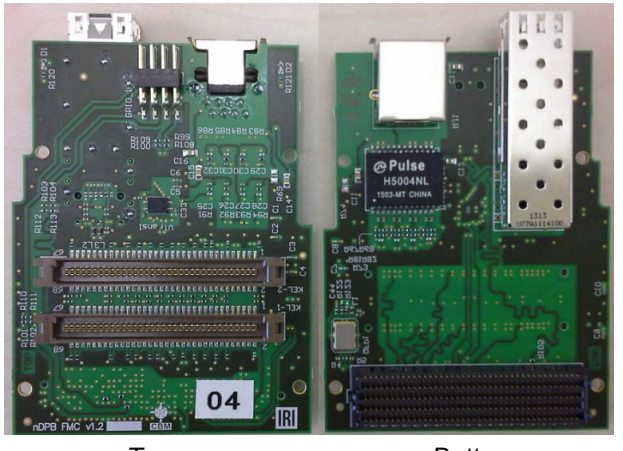
ELMA MicroTCA.4-Crate with 6 Slots



SPADIC 2.0 testboard

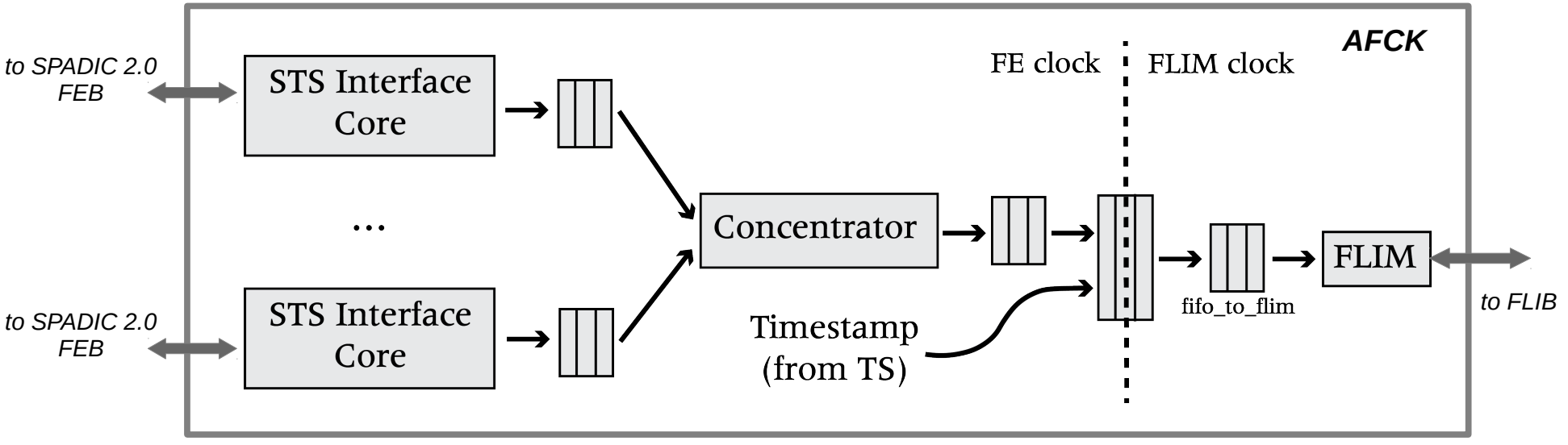


Top *eDPB-FMC* *Bottom*



Top *nDPB-FMC* *Bottom*

The TRD Readout Proposal at CERN-SPS 2016 testbeam - Firmware structure



AFCK firmware structure diagram

- IPBUS nor TS sync logic not shown in diagram, only data flow
- Used same design structure as 'nDPB_simpleMS' firmware design
- Result : ***eDPB*** firmware design
  ('***e***' stands for eLink and '***DPB***' for Data Processing Board)

<u>Simplified Microslices</u>

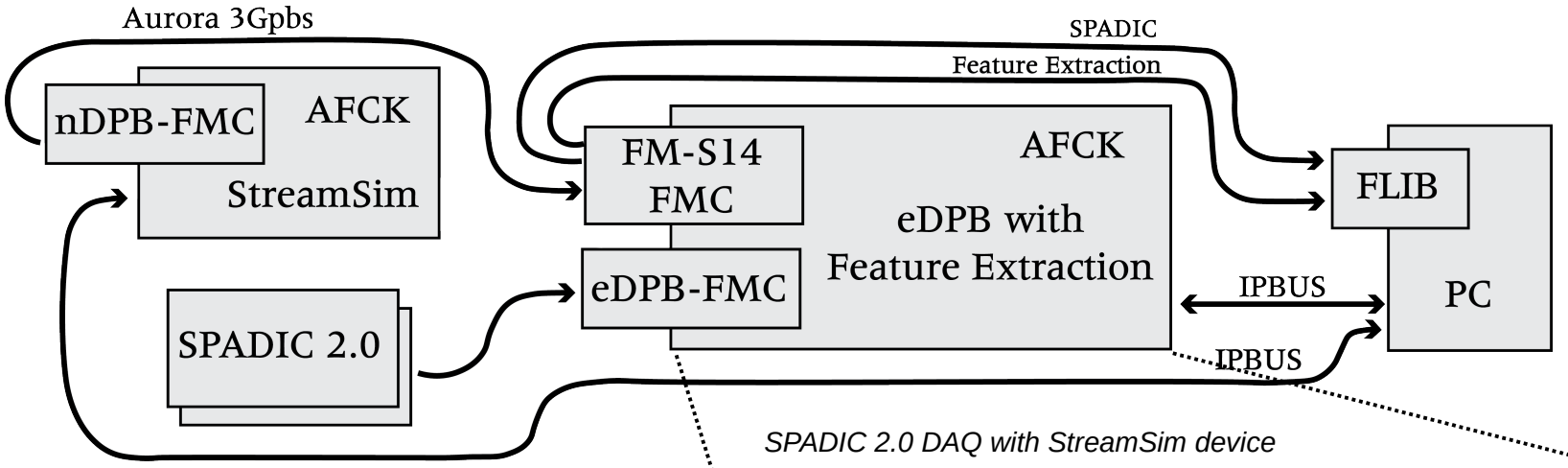
- FEE are synchronized with the TS system

- MS timestamps are synchronized with the TS system

- MS timestamps are built depending on the FEE data pre-processing time

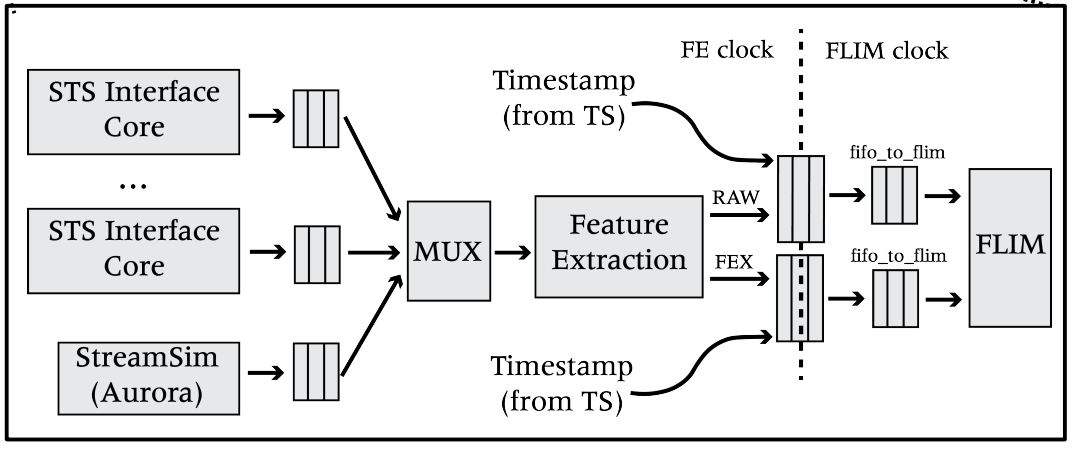
Based on Junfeng's DAQ talk, during the 28th CBM collaboration meeting

Firmware structure - eDPB simplified microslice with feature extraction



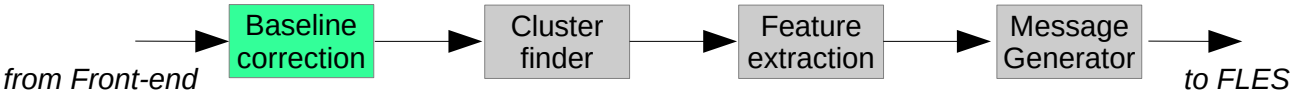
SPADIC 2.0 DAQ with StreamSim device

- Available since January 2017
  (after CERN-SPS 2016 testbeam)
- Used to:
  - → Readout SPADIC 2.0
  - → Feature extraction development and testing
- **StreamSim**
  - → Used for online/offline analysis of:
    - Algorithm tuning (e.g. Cluster finder, feature extraction)
  - → Inject testbeam data to the eDPB (feature extraction)
  - → Simulate data flow from front-ends
  - → Simulates SPADIC 2.0 e-Link frames
  - → Data transport link based on Xilinx's Aurora @3Gpbs



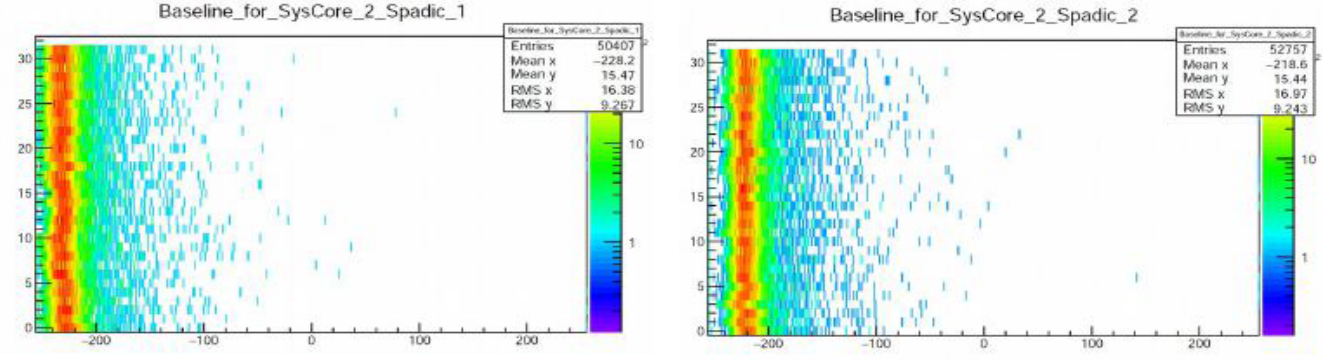
eDPB with feature extraction - firmware diagram

Baseline correction



Feature extraction algorithm diagram

- First processing stage and most important step in data reduction
- Decrease the noise and systematic effects
- Baseline equalization for all 32 channels per SPADIC done by software

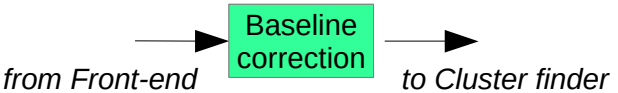


Example baseline equalization plot for all 32 channels of SPADIC_1 and SPADIC_2 readout by the SysCore_2 during the SPS CERN 2016 testbeam

- On the DPB, the baseline correction is applied on every SPADIC hit-message

Baseline correction

Operation modes



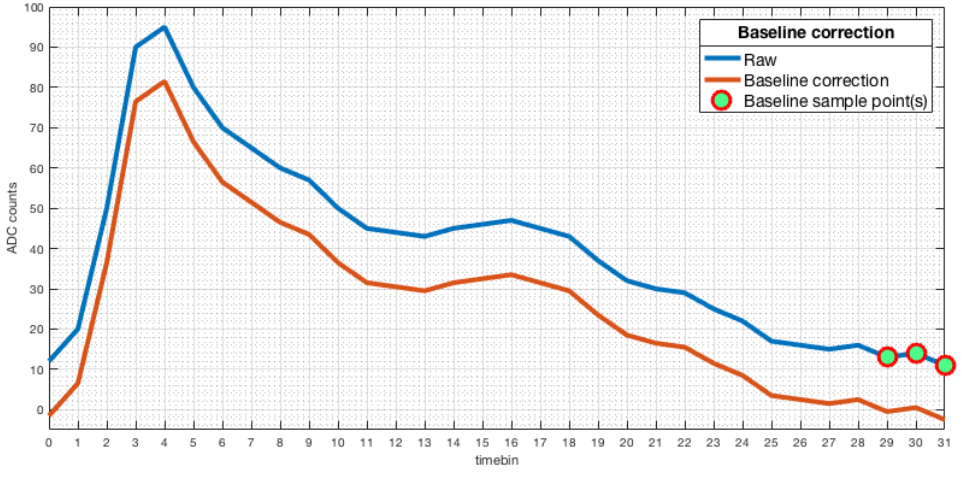
a. Fixed mode:
A constant value stored in a register is subtracted from all time bins

b. Bin dependent:
For every value of the current message, a value stored in a memory is subtracted from all time bins

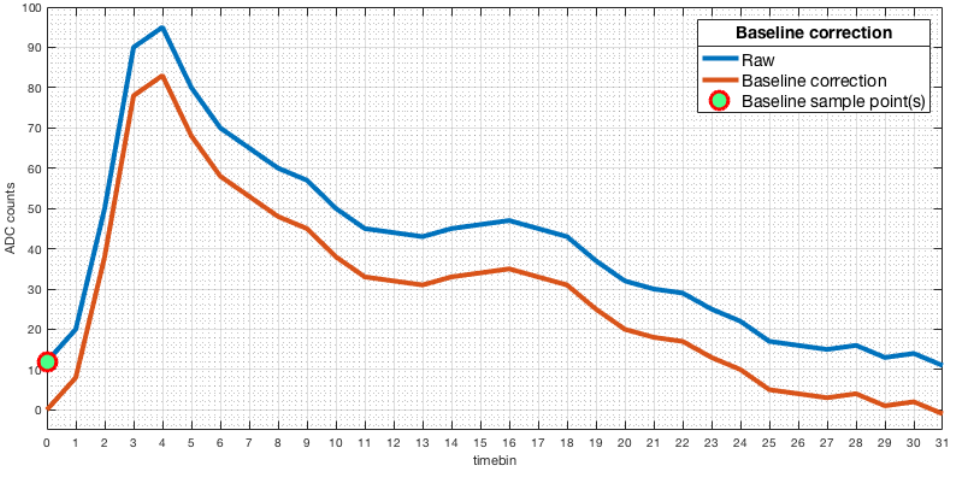
c. Variable:

c.1 Average of the last three bins of the time dependent signals is subtracted from all the message time bins
Calculation is performed for every message independently

c.2 First sample in front of the rising edge is subtracted from all message time bins
Calculation is performed for every message independently



Plot of operation mode c.1



Plot of operation mode c.2

Baseline correction

Correction methods according to SPADIC stop-type word

- Number of samples are included in the SPADIC end-of-message word

Word type	Format	Content
SOM	1000 gggg gggg cccc	g: group ID, c: channel ID
TSW	1001 tttt tttt tttt	t: timestamp
RDA	1010 dddd dddd dddd	d: raw data
CON	0ddd dddd dddd dddd	d: raw data
⋮	⋮	⋮
CON	0ddd dddd dddd dddd	d: raw data
EOM	1011 nnnn nnhh -sss	n: number of samples contained in raw data block, h: hit type, s: stop type

Word description of a SPADIC hit-message

Stop type	Description
000	normal end of the hit message
001	aborted because the message output buffer was full
010	aborted because the ordering FIFO was full
011	multi hit: the next hit message was triggered before the current one was finished
100	output buffer was full, multi hit detected simultaneously
101	ordering FIFO was full, multi hit detected simultaneously

SPADIC stop-type description

- Messages have a defined number of samples (1-32)

Based on the SPADIC stop-type word, two methods can be applied :

- Default case:
  - Single-hit
  - Normal stop-type word

- Multi-hit case:
  - Multi-hit stop-type word

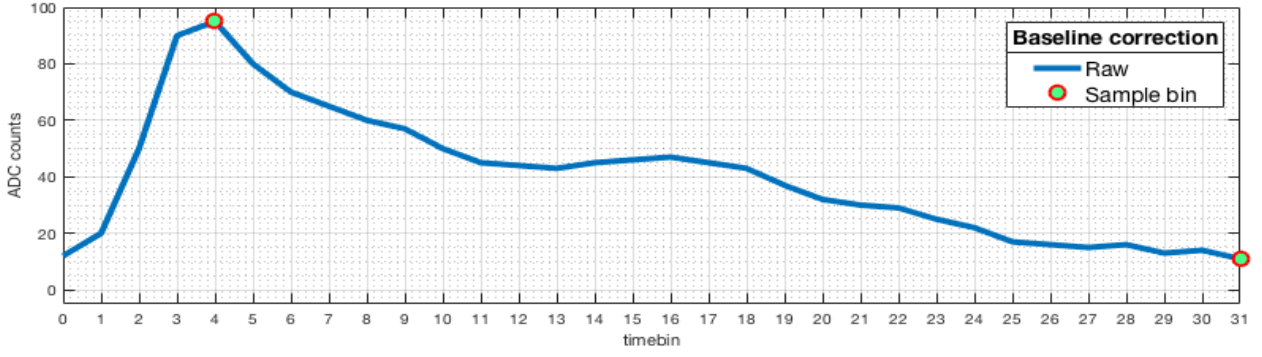
CBM

Baseline correction

Default Case

- Message with sample number larger ***two:***

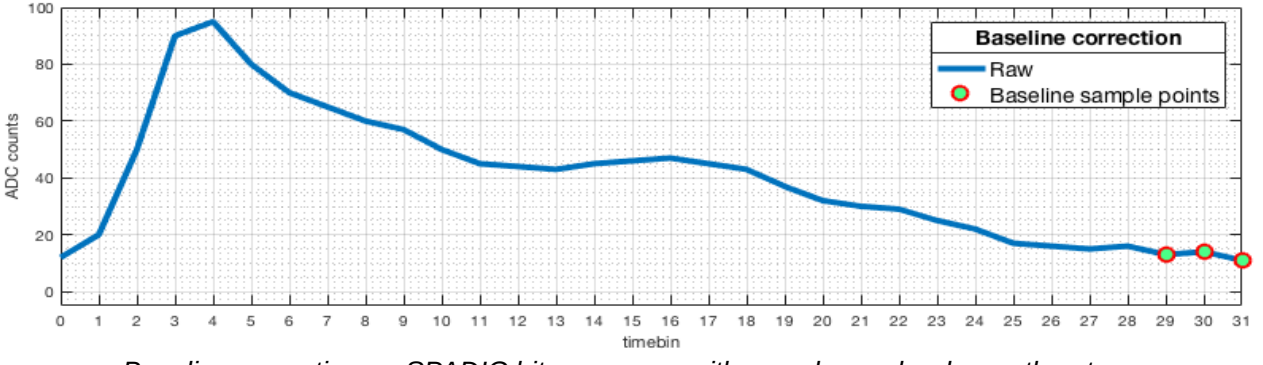
  - One sample maximum amplitude
  - One pre-sample / with maximum distance relative to the maximum
  - The pre- or last message sample(s) is/are used to calculate and subtract the channel baseline



Baseline correction on SPADIC hit-messages with sample number larger two. Sample bins on bin 4 for maximum amplitude and bin 31 with maximum distance relative to the maximum

- Message with sample number *larger* than two:

  - Average of the last three bins of the time dependent signals is subtracted from all the message time bins

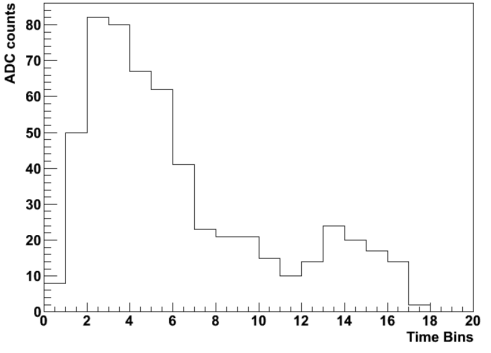


Baseline correction on SPADIC hit-messages with sample number larger than two.

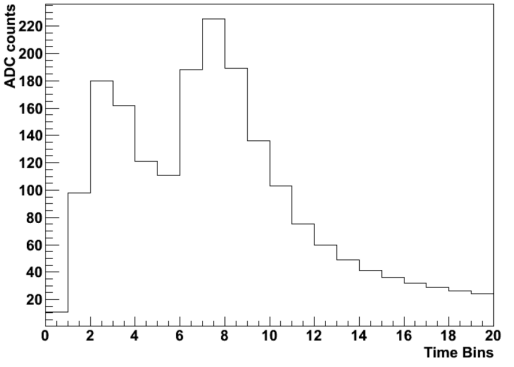
Baseline correction

Multi-hit case

- Number of samples smaller than normal predefined
- Second signal is on top of the ion-tail slope of the previous one
- Subtracting last-three bin's average → ***baseline overcompensation!***



Example single-hit event



Example double-hit event

Approaches:

- Subtraction of the pre-sample.
- Subtraction of a fixed measured baseline value.
- Subtraction a fitted baseline offset using the SPADIC response function.

Alternatives:

- Using SPADIC IIR filter
- No online processing of multi-hits → *Transporting multi-hit messages directly to FLES for detailed analysis*

Cluster finder

- Data from *each* SPADIC is already time sorted

  → Memory consumption is greatly reduced compared to another ASICs

- SPADIC word *hit-type* is a useful feature:

  → Explains why the hit message was generated

Hit type	Description
00	triggered by global signal (DLM)
01	self-triggered
10	triggered by neighbor channel
11	both of the above simultaneously

SPADIC hit-type description

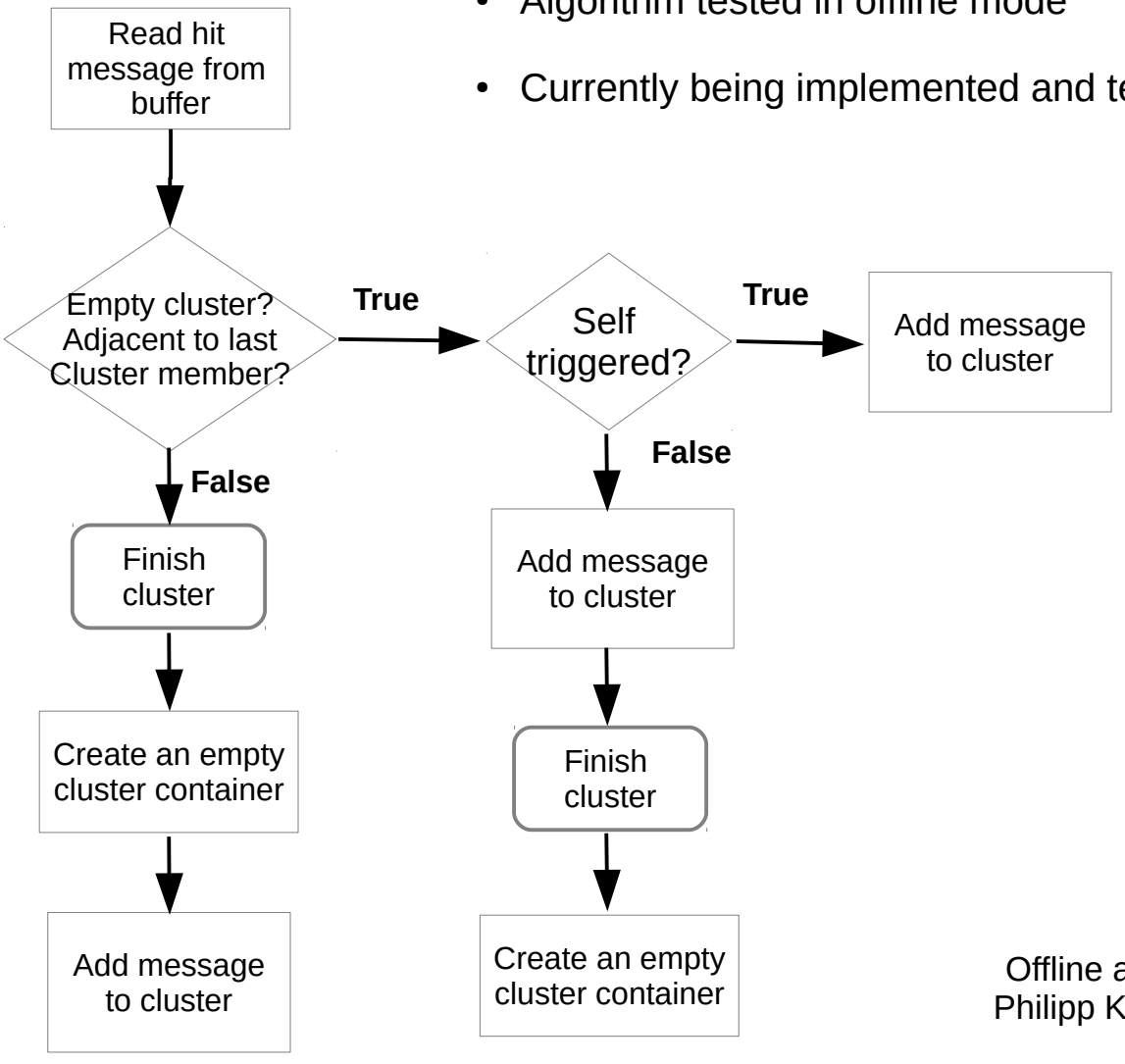
Goals:

- Create hit-message clusters in space
- Apply feature extraction algorithms to the found clusters
  - Extract:
    - Total cluster charge
    - Centre-of-gravity in time and space direction
  - In other words, extract hit-position along the pad-row, cluster charge and time
- Data reduction

CBM

Cluster finder

Algorithm

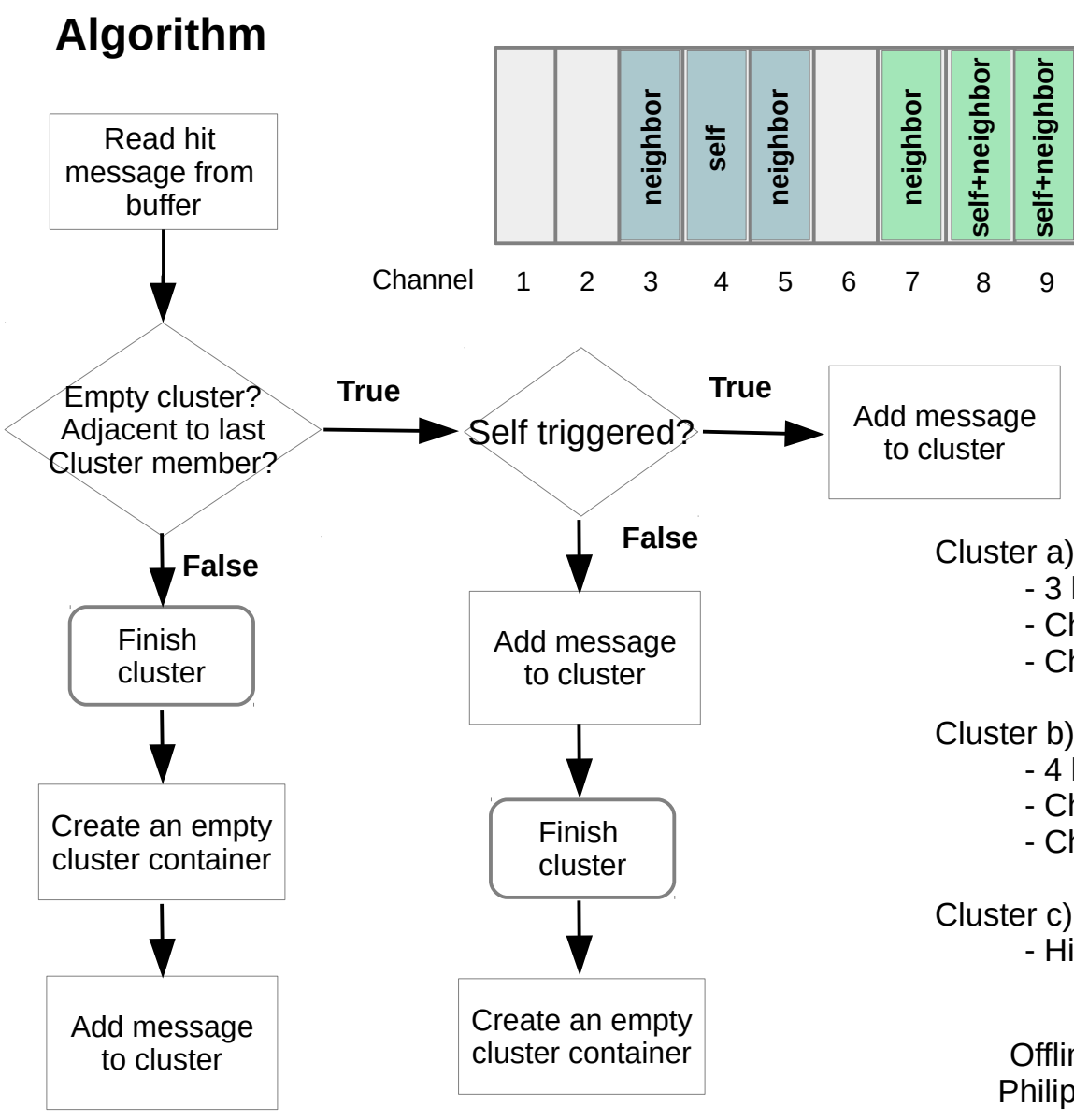


- Algorithm tested in offline mode
- Currently being implemented and tested in the online feature extraction layer

Offline algorithm results, please see Philipp Kähler's *Test beam results talk*

Cluster finder

Algorithm



Cluster a)
- 3 hit message cluster
- Channel 4 self-triggered
- Channels 3 and 5 triggered by neighbor

Cluster b)
- 4 hit message cluster
- Channels 8 and 9 self and neighbor triggered
- Channels 7 and 10 neighbor triggered

Cluster c)
- Hit message on border pad, sent to FLES without processing

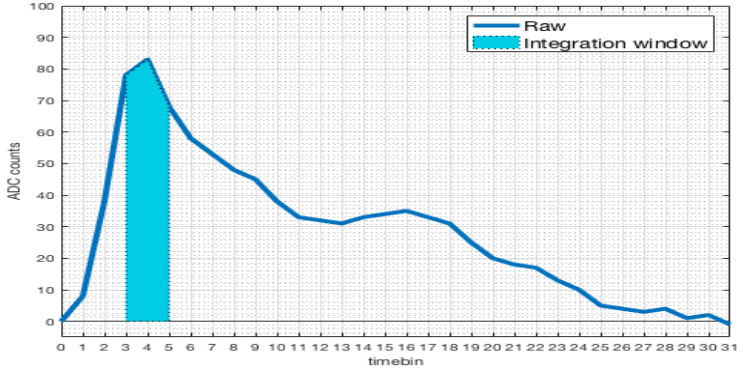
Offline algorithm results, please see
Philipp Kähler's *Test beam results talk*

Offline cluster finder algorithm analysis and results in Philipp Kähler Test Beam Results talk

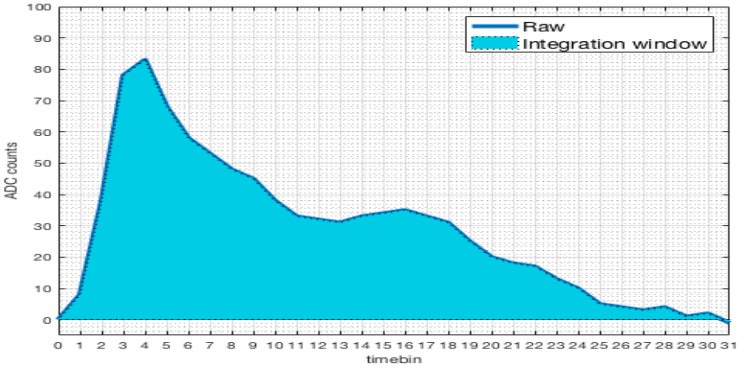
Signal pre-processing

Feature extraction algorithms:

- Integrating signal amplitude over fixed number of samples

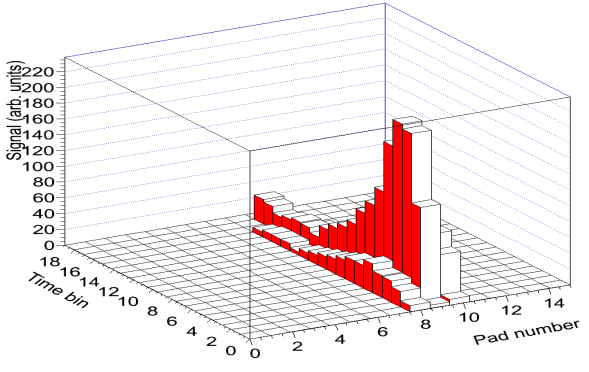


Example of signal integration around the maximum peak

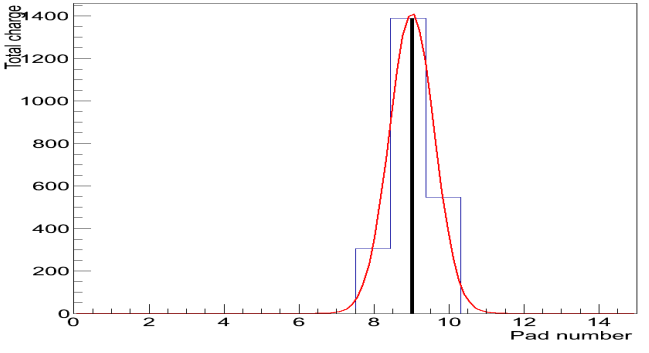


Example of signal integration of the complete signal

- Center-of-Gravity (Spatial)



Example 3-dimension plot of a 3 pad cluster



Center of gravity calculation of a (signal integrated) 3 pad cluster. Black line marks the position of the cluster centre along the pad

- Fitting the measured signal samples to the SPADIC response function

  → Approach still under development

Data reduction

Considerations:

- One SPADIC hit-message containing 20 time bins

  → Total of 256 bit

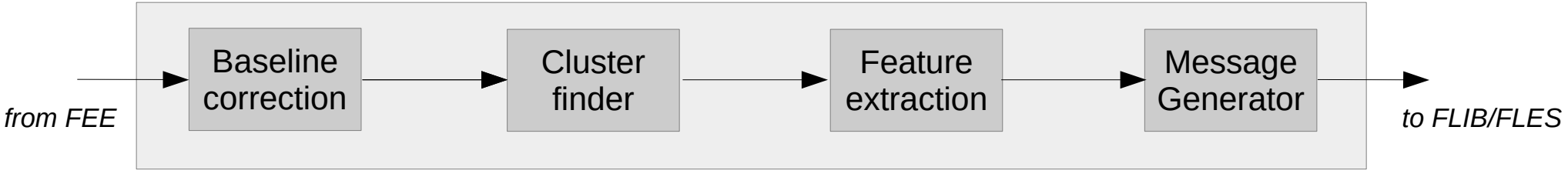
  → Hit-message samples can be between 1-31 time bins

- Cluster size of 3 pads (default configuration)

  → Configurable cluster size in firmware

Baseline correction

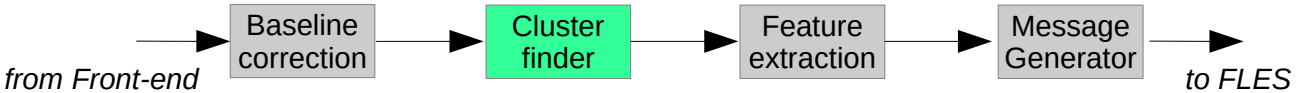
- No data reduction at this point
- Signal conditioning for later processing



Feature extraction algorithm structure implemented in the AFCK

Data reduction

Cluster finder



Feature extraction algorithm diagram

- **Hit-message gathering**

  → Gathers SPADIC hit-messages that belongs to a cluster

  → No data reduction at this level

- **Cluster merger**
  - Merges meta-data from SPADIC hit-message clusters
  - A cluster can not have empty pads

    → Metadata can be reduced
  - Cluster timestamp

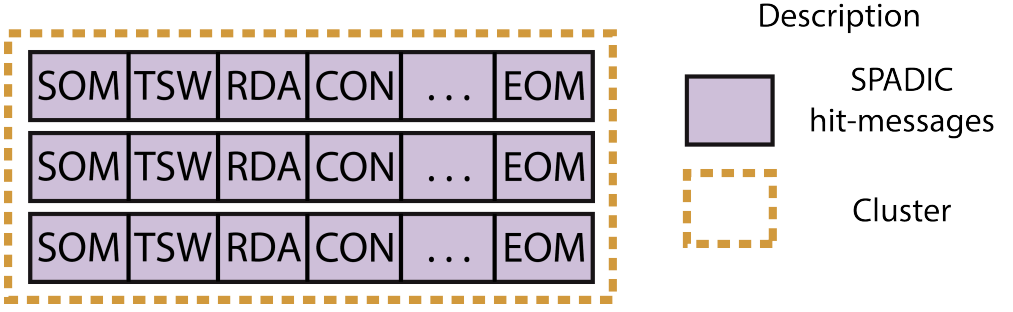
    → Selected from central pad

    or

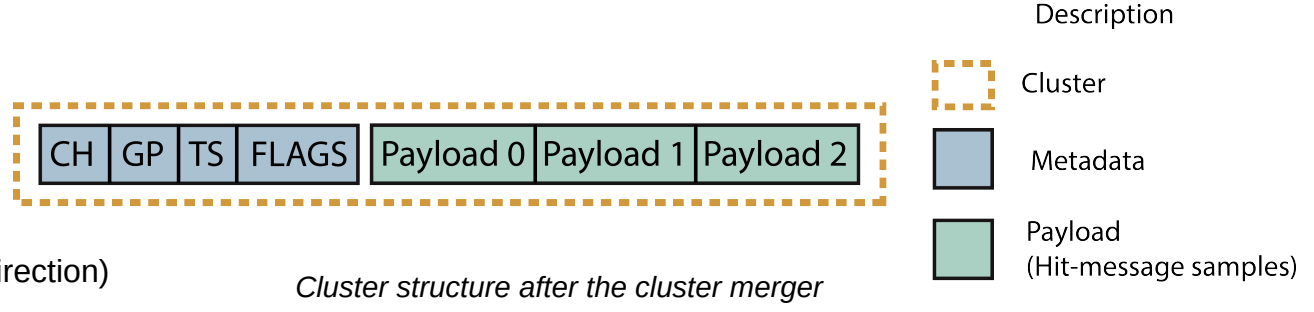
    → Calculated by Centre-of-gravity (time direction)
  - Data reduction ratio of **~1.3**

    Output of hit-message gathering : ~768 bit

    Output of cluster merger : ~576 bit
  - *Data reduction optimization under development*



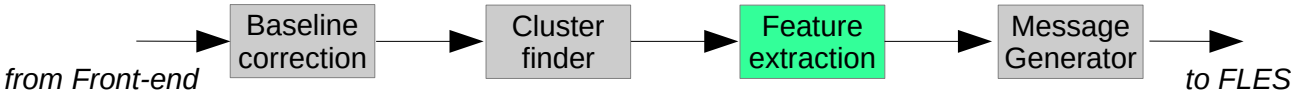
Hit-message gathering of the cluster finder algorithm.
A cluster instance groups multiple SPADIC hit-messages



Cluster structure after the cluster merger

Consider:
→ SPADIC hit-message containing 20 time bins
→ 3 pad cluster

Data reduction



Feature extraction algorithm diagram

Feature extraction

- Processing over found clusters

- **Algorithms used**

  → Total charge integration

  → Centre-of-gravity (Spatial)

  → Signal fit (*under development*)

Consider:
→ SPADIC hit-message containing 20 time bins
→ 3 pad cluster

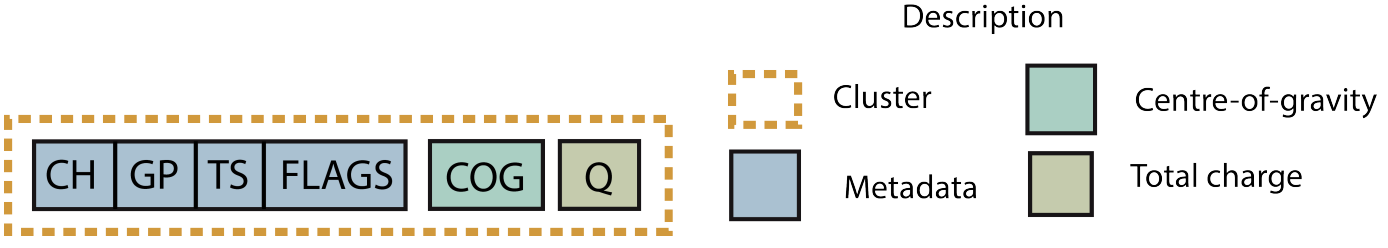
- **Output vector**

  → Data reduction ratio of **~4.5** (*preliminary*)

  For an input cluster of ~576bit → output vector of 128 bit (COG and Q of 32 bit words each)

  → Algorithm output payloads between 32 to 64 bit (according to accuracy/error)

  → On-going analysis of algorithm accuracy against offline simulation



Example feature extraction output vector

Conclusions

- **eDPB firmware for the TRD readout proposal available** → *OK*

  → Developed during the CERN SPS 2016 testbeam

  → Integration of the SPADIC 2.0

  → Includes feature extraction algorithms

- **Optimization for TRD readout proposal** → *Under development (close to final)*

  → On-going online/offline simulations in order to compare data quality

  → FPGA resource consumption based on the AFCK

  → Online algorithm tuning based on simulations and beamtest data

Thank you