Solving common readout, DAQ and trigger problems with online digital techniques

Raúl Esteve, José Toledo

rauesbos@eln.upv.es, jtoledo@eln.upv.es

Universitat Politècnica de València – NEXT Collaboration

Scope

Those acquainted but not experts in electronics, might find some value in a short survey of problems that could be solved with reconfigurable logic, showing how far one can go even with mature technology.

Pros?

- Re-configurability and adaptability to new demands of the experiments
- Allows more powerful and sophisticated trigger by moving processing stages from the offline system

Cons?

- Those associated with FPGAs, including power dissipation

We will deal only with the following issues (this is a short 25' talk!)

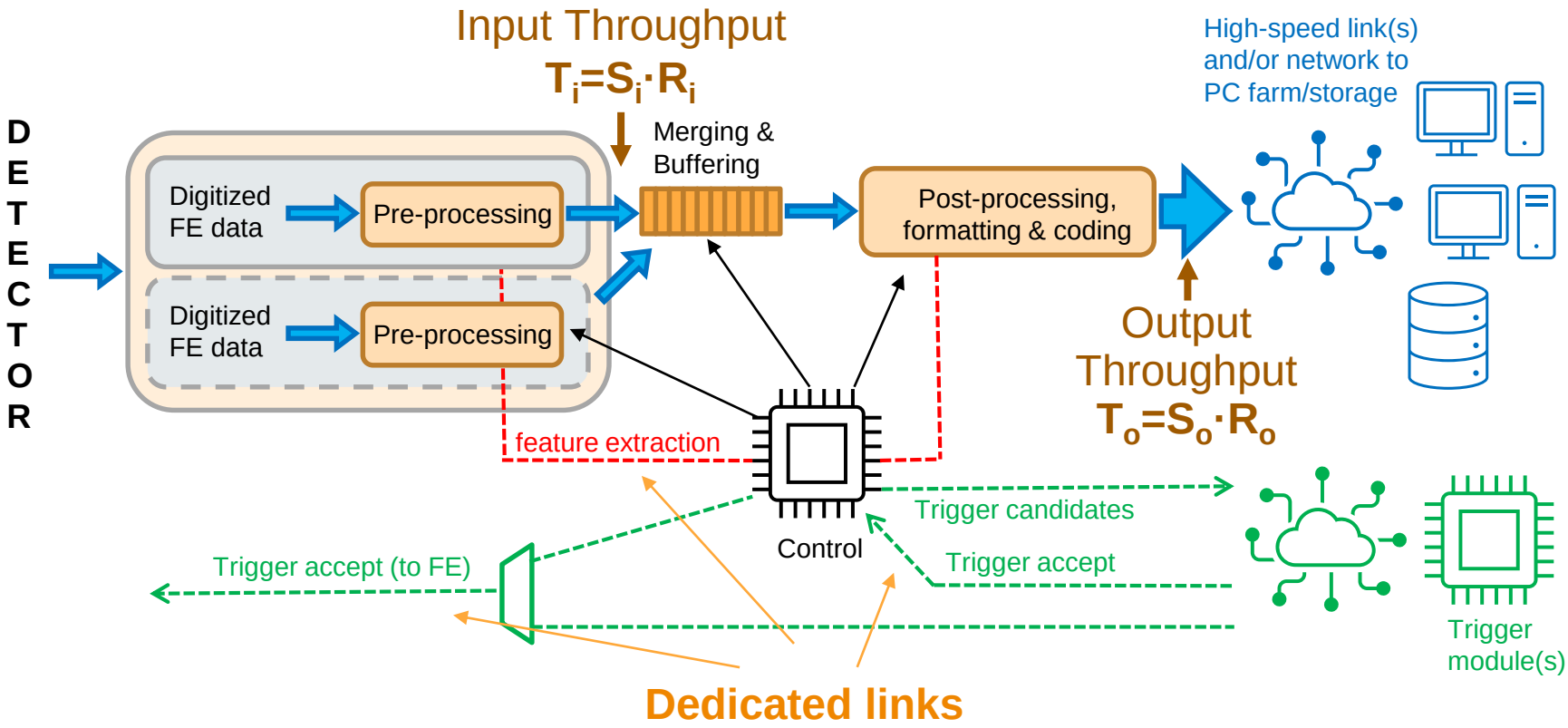
- **Dead time reduction: Bottleneck and solutions**
- **Data compression**
- **Complex and reconfigurable trigger algorithms**
- **AC coupling readout**

Dead time reduction – The problem

- System dead time occurs when the system cannot process further events

- **What is causing dead time?**
  - **Excess of data generated**:
    - Excessive sampling rate (it happens sometimes!)
    - Trigger is not efficient
  - **Only one and large buffer that must be readout**
  - **Insufficient data throughput to the PC farm**
    - Low throughput
    - Number of transmission lines to the storage system vs cost
  - **Trigger latency is rarely a problem in TPCs**

Dead time – Where are the bottlenecks?



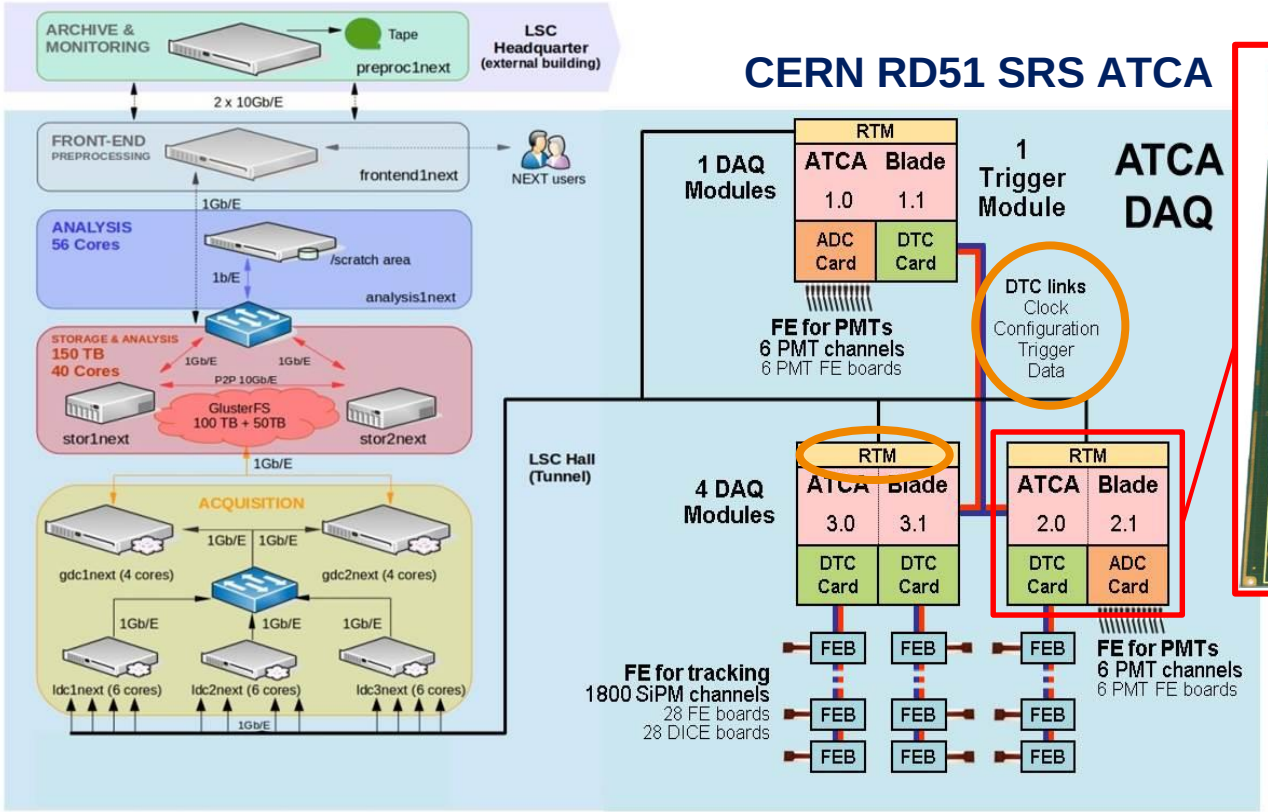
Dead time reduction – Summary of solutions

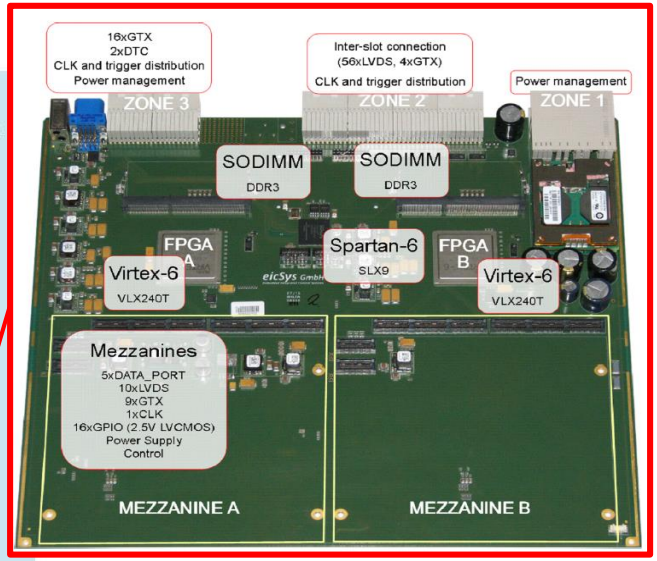
- **Multiple buffer schemes**
  - Reduce dead time since data buffering and data transfer can work in parallel
  - A double buffer is sufficient for most of applications
  - Buffer size matters!
    - Its **size can be dynamically assigned** and dependent of the event
    - Easy to implement with circular buffers where one only need to manage pointers
  - Buffer implementation:
    - Internal FPGA dual-port memories (never as large as we'd like!)
    - **External DDRx memory can be interface to emulate a dual port memory**

Dead time reduction – Summary of solutions

- **Sampling rate**
  - Avoid sampling faster that required by the application
  - Keep high sampling rate but send only sum of samples (re-binning), or just less samples
- **Improving throughput**
  - Having more transfer links rises data throughput but **also hardware cost**, (more GbE links, more ethernet cards, more storage PCs,...)
  - 10 GbE improves data throughput, reducing dead time
- **Data compression**
  - Significant reduction of data to be transferred per event (alleviates output link(s) limitations)
- **Improving trigger efficiency**
  - Improving detection algorithms to prevent “wrong” events from being sent upstream

Dead time reduction - Our solution for the NEXT TPC





Dead time reduction - Our solution for the NEXT TPC

- **A double circular buffer scheme implemented, from 80 µs to to 3.2 ms of data buffer):**
  - Not enough memory in FPGA to implement such a buffer for PMT data!
  - Double buffer scheme in external DDR3
  - Dual-port memory emulation in RAM is possible due to the fast interface, up to 1280 MB/s!

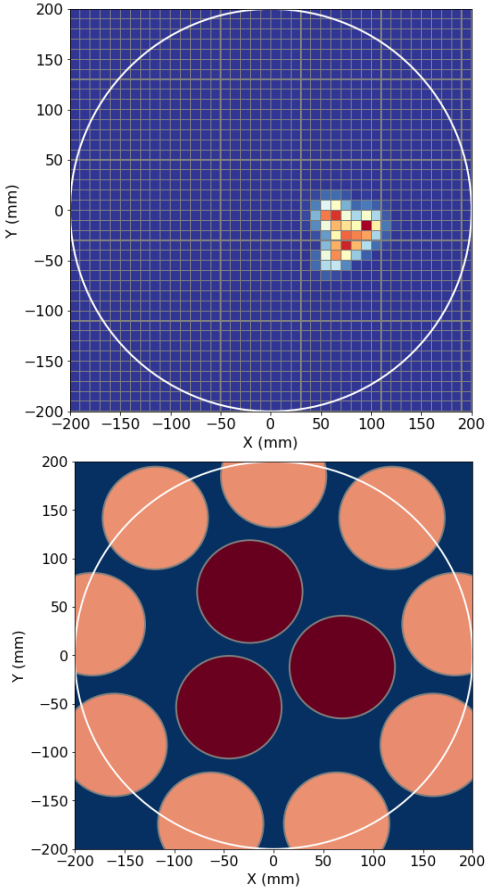
- **Our bottleneck, in the output links (PMTs)…**
  - Each buffer can accommodate up to 2.3 MB: 12-bit 12 channel at 40 MHz of PMT data for 3.2 ms
  - Available throughput: 2 link x 8 bit at 125 MHz = 250 MB/s (9.2 ms to read the maximum buffer size)

Data compression – Summary of solutions

- **Online data compression**
  - Lossy compression:
    - Traditional Zero-Suppression
    - Simple algorithm
  - Lossless compression:
    - Huffman codes.
    - More complex algorithms

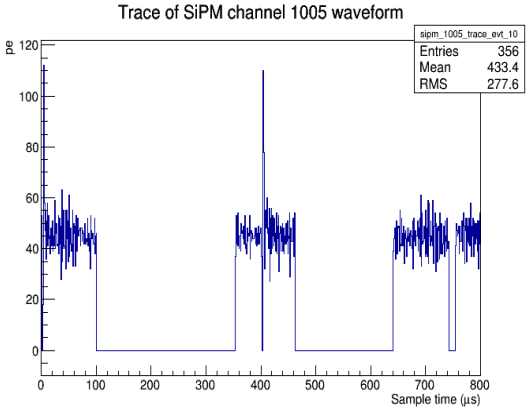
- When apply each type of compression?
  - Lossy compression is suitable when less channels see signal per event (tracking planes), it is important not to introduce a bias in the analysis
  - Lossless compression preserves the whole signal, and can reach similar compression rates when more sensors see signal and events are large in comparison with the capture window (energy planes)

- In both cases, and according to the signal topology, compression rates from 60 % to 80 % can be achieved



Data compression - Our solution for the NEXT TPC

- Two factors have been taken into account:
  - Dead time reduction for interesting events vs increase of trigger rate for calibration RUNs
  - Two different type of sensors: SiPM (tracking plane) and PMTs (Energy plane with AC-coupling readout, RAW data)



- Two different approaches implemented:
  - SiPM (low channel occupancy):
    - **Traditional Zero-Suppression**
    - Data compression rate of approx. 74 %
  - PMT (high channel occupancy)
    - **RAW signal AC-coupling readout makes zero-suppression less efficient due to the long tail of the signal**
    - **Lossless data compression applied (Huffman encoding) to delta codification signal** achieves a data compression rate of approx. 80 % with fixed codes

- **Data compression is clearly feasible with few FPGA resources (**Xilinx FPGA Virtex-6 LX240T):
  - Zero-Suppression: 2 % Slice LUT, 3 % of RAM (64 SiPM channels) at 100 MHz working frequency
  - Huffman encoding: Less than 1 % Slice LUT, less than 1 % of RAM (per PMT DAQ module, up to 12 PMT channels) at 200 MHz working frequency

Complex trigger algorithms – The problem

- **Dead Time increases if trigger is not efficient**, excess of data transmission must be avoided

- Most TPCs searching for rare events must trigger using the readout signals (dark matter, double beta), **online trigger needed**!

- **Calibration** is always a problem:
  - Takes lot of time and data
  - DAQ must work at very high rate as opposed to the lower event data rate!

Complex trigger algorithm – Summary of solutions

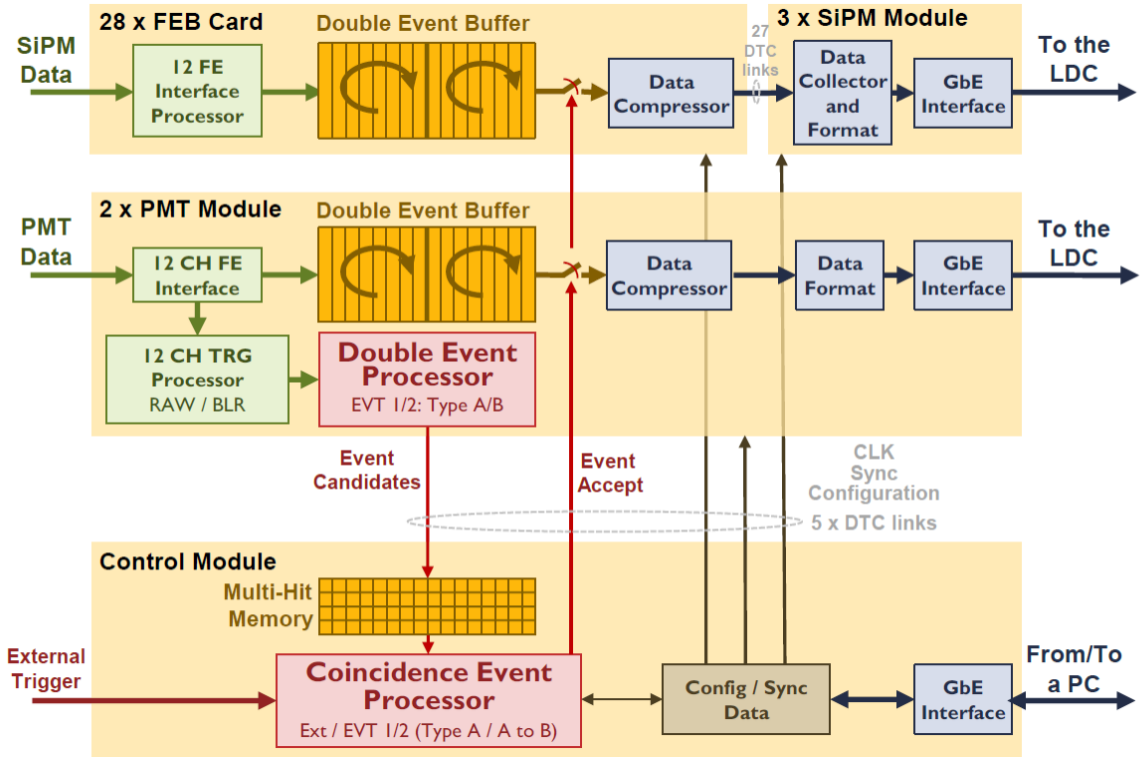
- Efficient trigger algorithms require lot of FPGA resources, so **trigger can be distributed among DAQ resources**:
  - Event selection at the level of sensor data processing
  - Trigger decision in a dedicated module: signal detection algorithms, memory hit concept
  - Fast links are needed to share trigger information among modules

- A **double trigger processor** allows simultaneous detection of different type of events: Simultaneous calibration and data of interest!

- **Complex detection algorithms can be implemented**

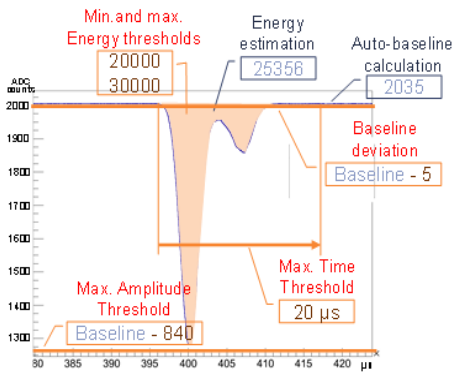
Complex trigger algorithm - Our solution for the NEXT TPC

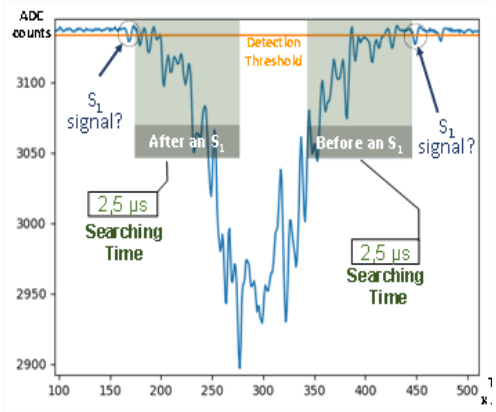
- Distributed among DAQ modules (Virtex-6 LX240T):
  - **Double Event Processors** (up to 12):
    - 7 % Slice LUT
    - 40 MHz
  - **Double Coincidence Event Processor** in a specific trigger module (up to 48 channels):
    - 26 % Slice LUT, 60 % RAM
    - 200 MHz

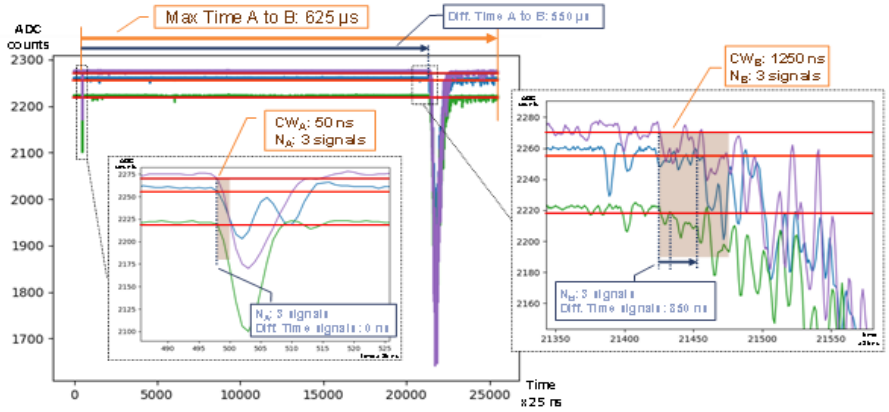


Complex trigger algorithm - Our solution for the NEXT TPC

- Features implemented:
  - Configurable coincidence trigger: Selectable number of events in a time window
  - Configurable fast event detection: Amplitude, time and energy thresholds
  - Event rejection system: Thought to reject false S1 detected
  - Two event types: Calibration while taking data with, and for the later, priority in the use of the second buffer, …
  - S1 to S2 trigger







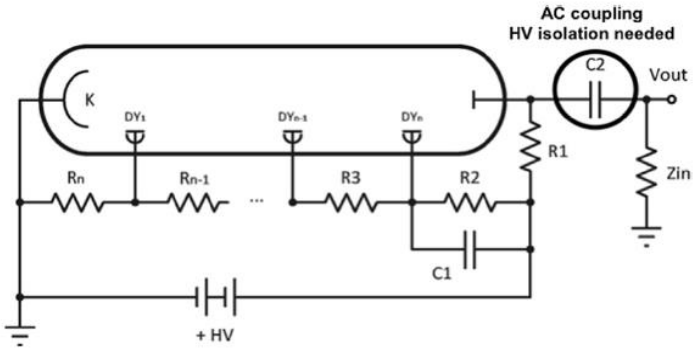
Complex trigger algorithm - Our solution for the NEXT TPC

- Three run statistics (24 h) with general configuration:
  - Two type of S2 events: ^{83m}Kr calibration data (EVT1) and “high energy” events (EVT2)
  - Buffer priority for beta decay events
  - At least 2 PMT events in a coincidence window of 1.2 us, data compression applied
- Back to reality!
  - **At the end, online settings determine Dead Time, throughput, trigger rate and priorities**!

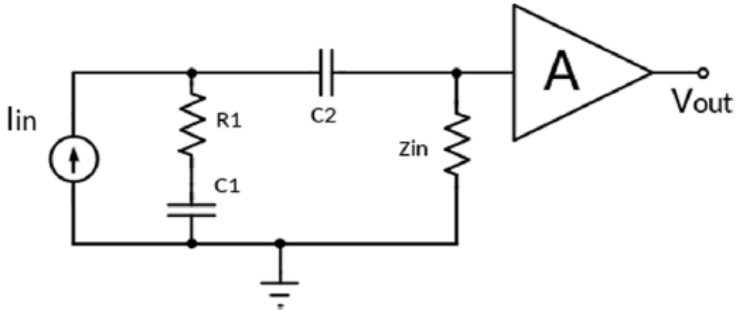
RUN	Events Searched	Total Event Rate	Data Rate	EVT1 Rate	Dead Time Rate	
				EVT2	EVT1	EVT2
7520	2	20 Hz	31.78 MB/s	0.47 %	9.4 %	3.8 %
7512	2	27 Hz	41.66 MB/s	0.37 %	16.4 %	5.3 %
7502	2	76 Hz	77.28 MB/s	0.11 %	39.2 %	15.15 %

AC-coupling readout – The problem

- In a grounded cathode PMT, the AC-coupling scheme:
  - Simplifies detector mechanics, on cable for supply and signal
  - Enhances safety (the alternative is the insulation of the PMT from ground)
  - Reduces of low frequency noise effects.

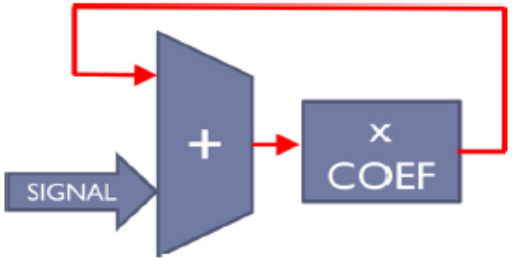


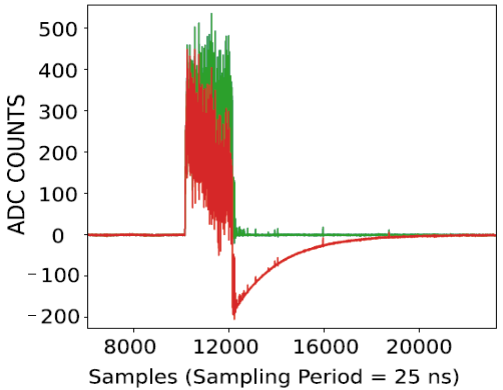
- **The problem is that it creates a High Pass Filter (HPF)**:
  - It affects the signal received in the DAQ, with a null total area
  - The input pulse area must be recovered, since the energy is proportional to this area and energy resolution must be preserved



AC-coupling readout – Summary of solutions

- A BLR (Baseline Restoration) applies **an inverse function of a HPF**
- **Analogue solutions:**
  - Gated BLR (Helmuth Spieler's short course at IEEE NSS 2002)
  - Closed-loop solutions (C. Liguori, G. Pessina, NIMA A 437 1999 557)
- **Online digital solution**
  - Pros:
    - Simple algorithm to implement in FPGA
    - Per-channel filter fine tuning
  - Cons:
    - Sensitive to non-zero mean noise
  - Avoiding low frequency introduced in the BLR:
    - The algorithm can be activated when a pulse is detected and switched off when the pulse ends
    - A cleaning filter (HPF) can be applied to cancel this DC effect

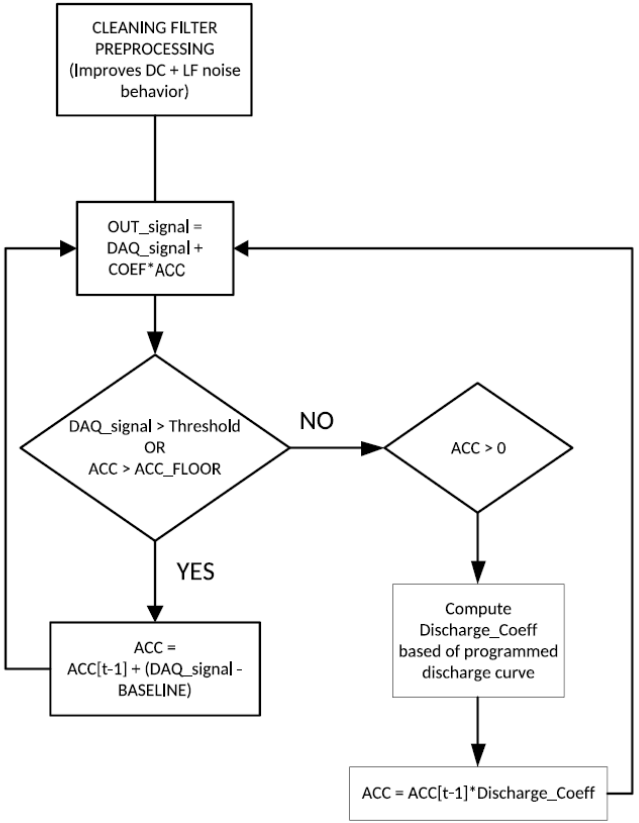




AC-coupling readout - Our solution for the NEXT TPC

- **BLR is applied on-line**, needed for the early event detection based on energy!

- Our BLR algorithm in Virtex-6 LX240T:
  - One per channel implemented (12), since all of them can be used for trigger
  - The on-line BLR algorithm is a 42-bit fixed point format BLR version
  - The BLR algorithm is activated when a pulse is detected
  - A moving average filter is needed to calculate the signal baseline
  - FPGA resources:
    - 1.1% Slice LUT, 0.42 % of RAM, per BLR
    - 40 MHz



Summary

- Nowadays, the DAQ bottleneck is the output data link(s):
  - Choose a DAQ module that can easily scale up in output throughput
  - Don´t lose sight of the cost!
- Buffer is not currently a bottleneck:
  - DDRx can easily emulate large multi-buffer dual-port memory
- Online data compression can alleviate the requirements:
  - It takes little FPGA resources
  - It can provide a factor 3-4 improvement in rate or in dead time
- Many TPCs need online trigger generation:
  - Choose an architecture that favours FPGA-based distributed online trigger: Modular, with dedicated links, …
  - Full control over operational parameters: Trigger rate, throughput, dead time,…
- FPGAs can efficiently accommodate functionalities in early readout stages like BLR that were previously at the FE or offline