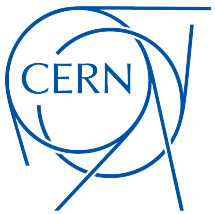
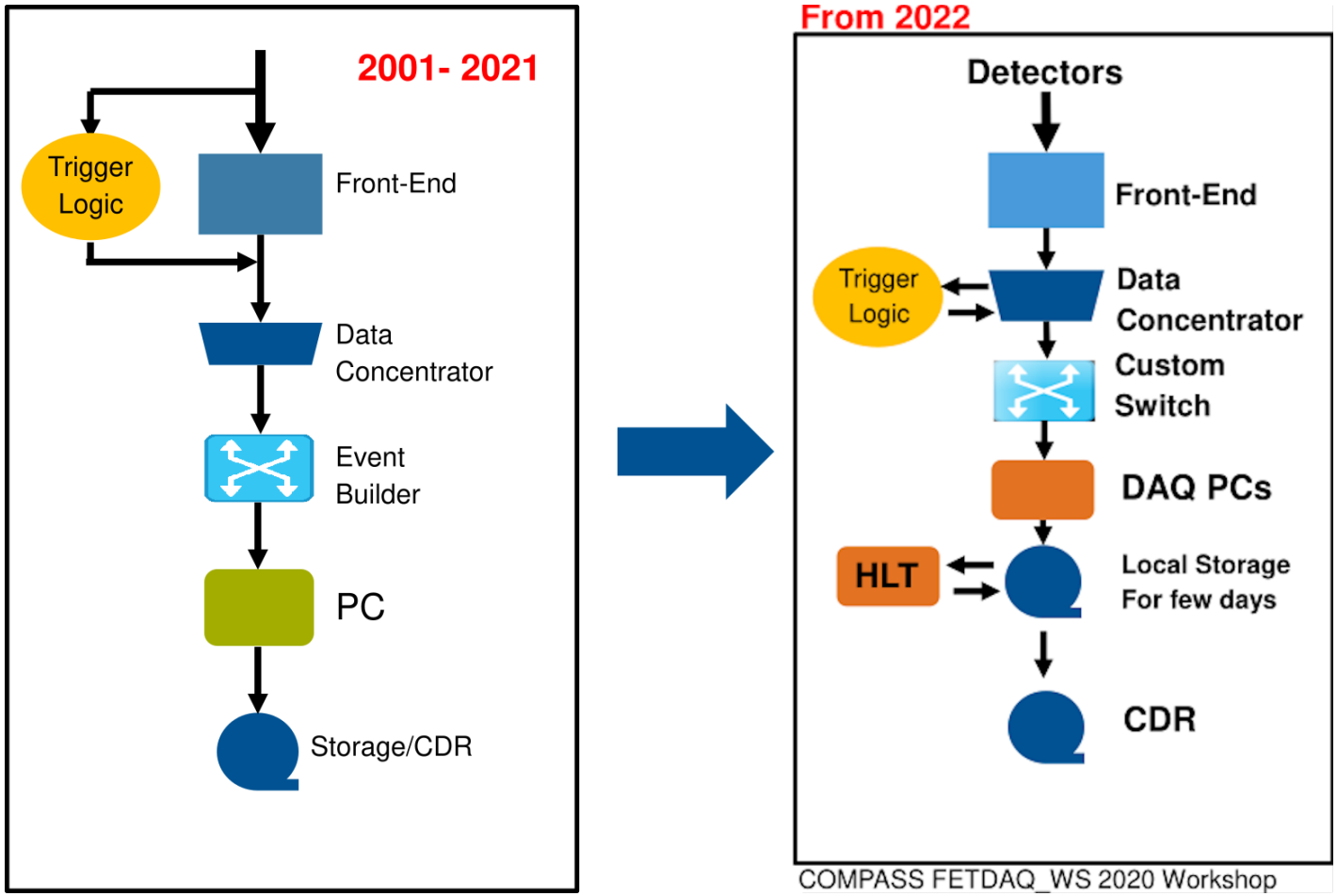
A Digital Trigger for the free running iFDAQ

Benjamin Moritz Veit

2. März 2020

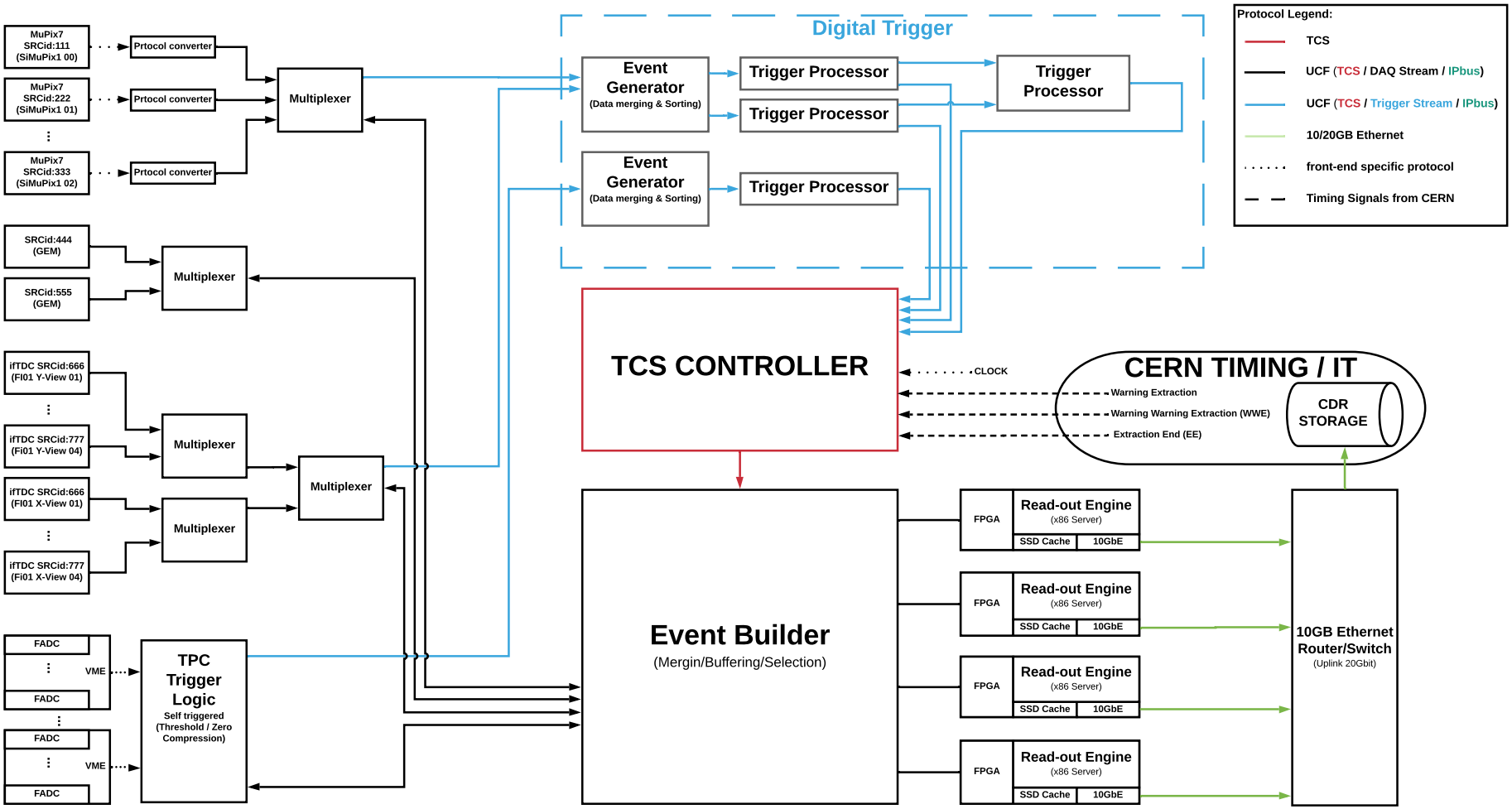


Evolution of COMPASS DAQ Architecture



Future: Splitting the continuous data stream at the level of the Data concentrators for DAQ and trigger logic.

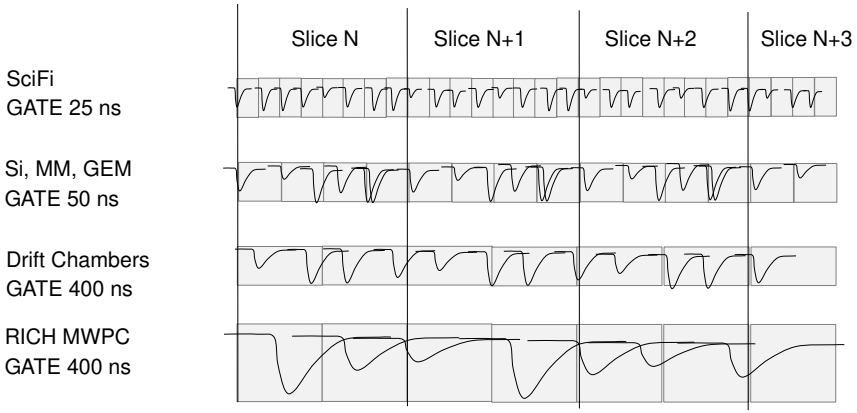
New DAQ Structure



Time Slices vs Images

Time Slices are generated by a synchronous signal which is distributed to all front-ends.

Additional partition in **images** according to detector resolution on front-ends.



Protocol Considerations

Unified protocol standard for all components of the DAQ

(Front-ends, Multiplexer, Digital trigger, ...)

- UCF as low level protocol to encapsulate different user protocols (IPBus, TCS, Data) over one physical link.
- Remove the information about the number of data words from header.
- Extension of the SRCid to carry additional information:
  - SrcID to identify Source (e.g. Multiplexer, Detector-System)
  - FrontendID replaces PortID to identify specific front-end
  - ViewID is used to group front-ends according to planes/sub-structure

  Helps to identify the origin of information for e.g. Trigger Processor.
- DAQ independent monitoring for debugging (e.g. Scaler read-out via IPBus)

Unified protocol helps for debugging since we can put a readout engine after each module to debug intermediate steps.

Running Modes of DAQ

- **Free Running Mode:** complete data stream from the detectors is saved to disk for later analysis $\rightarrow$ events have to be defined on the reconstruction level. (Pre-filtering/selection of data before physics reconstruction.)

- **Triggered/Filtered Mode:** Online information from trigger processors is used to reduce data size before it is written to disk. Remaining data is close what we call at the moment "Event".

Multi Stage Digital (FPGA) Trigger

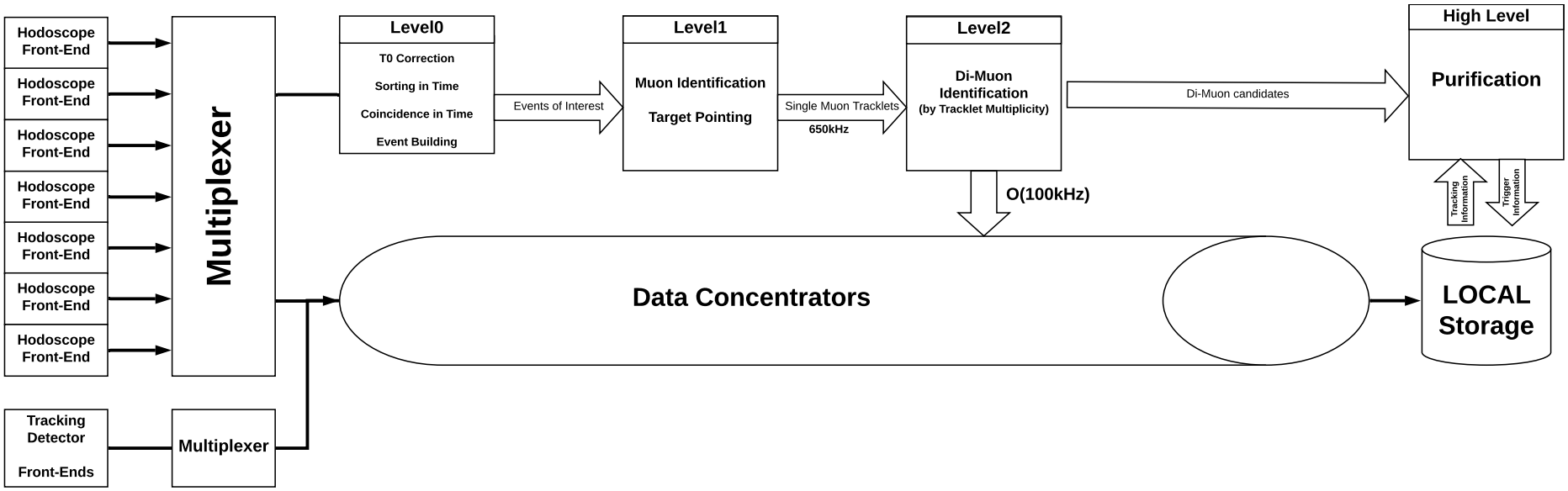
Using the continuous parallel readout stream for trigger logic to make trigger decision on geometrical/energy criteria in real-time.

Continuous-Readout:

→ One stream buffered in data concentrators for DAQ.

→ One stream of an **subset** of detectors is sent to Trigger-Logic.

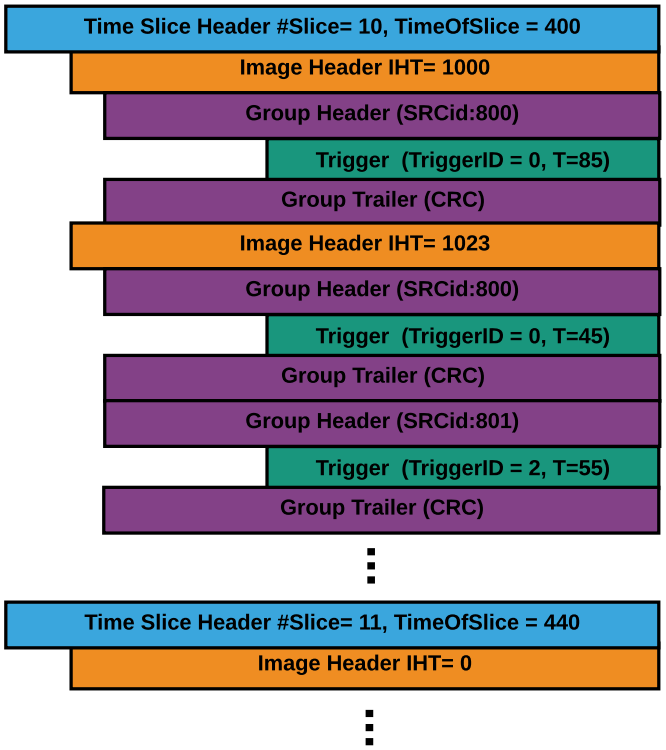
Trigger-Logic:



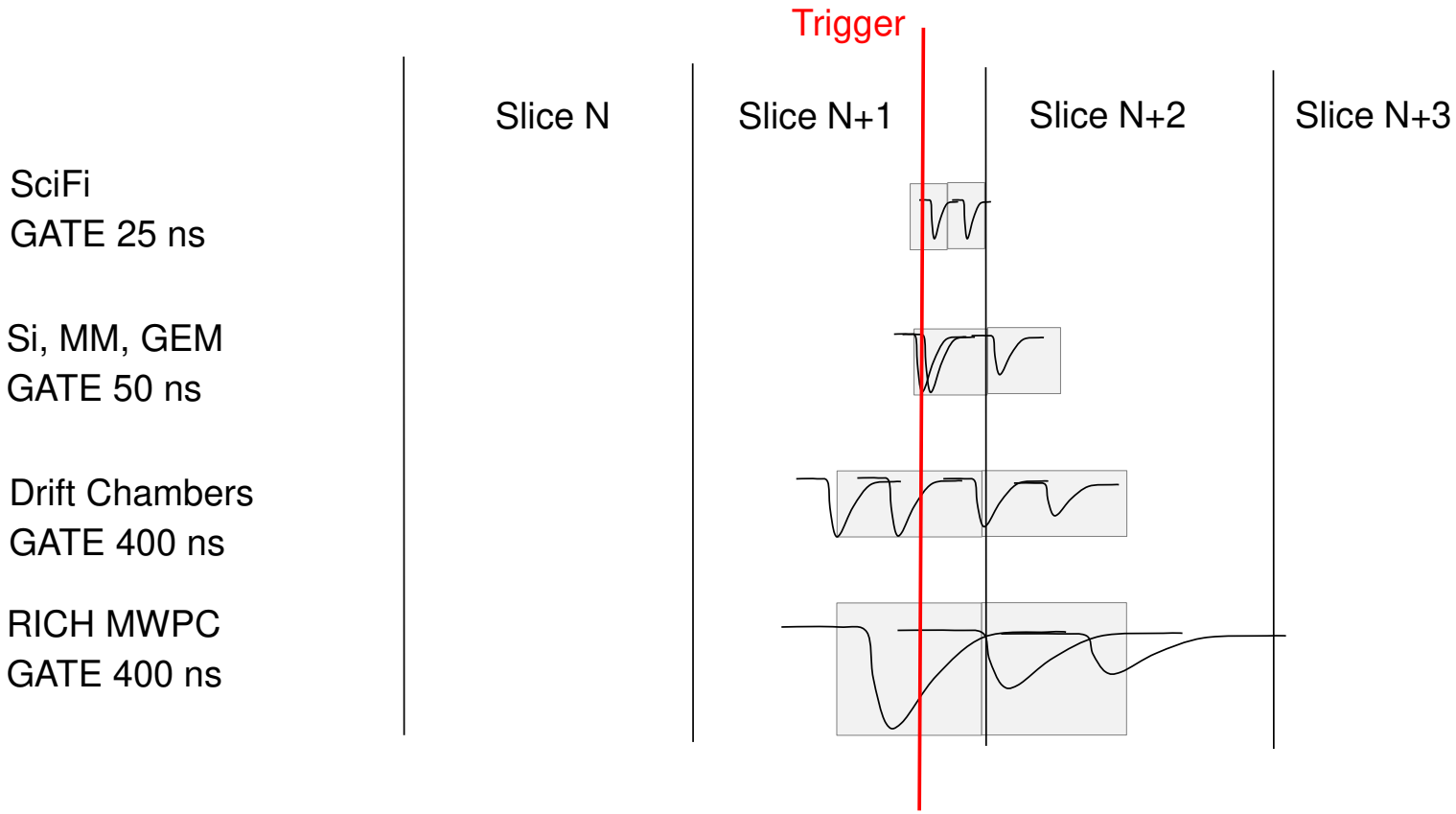
Trigger Processor Information

Two data streams:

- One on the **TCS** level distributed to the multiplexer/event generator to reduce incoming data stream size
- One on the **data stream** level. Trigger Processor represents an SRCid which contains the information of the Trigger decision



Tagging Images from DAQ stream in Buffer



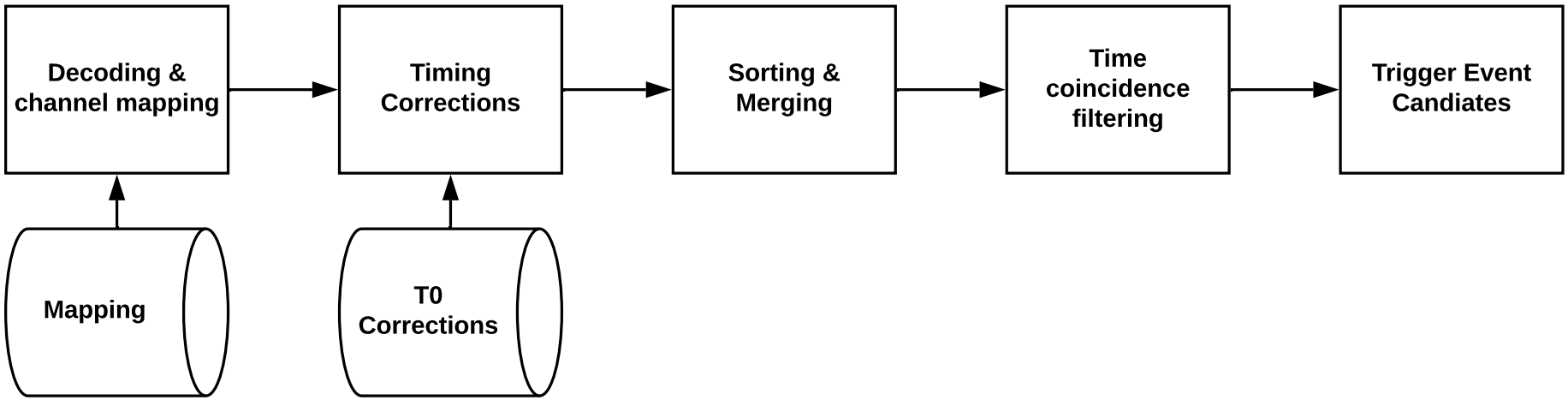
After trigger decision two consecutive images from DAQ data stream are saved as event for data reduction.

But time correction have to be applied “online”.

Details on the Level0 Trigger

Trigger Event Building

Function of the Level0 Trigger

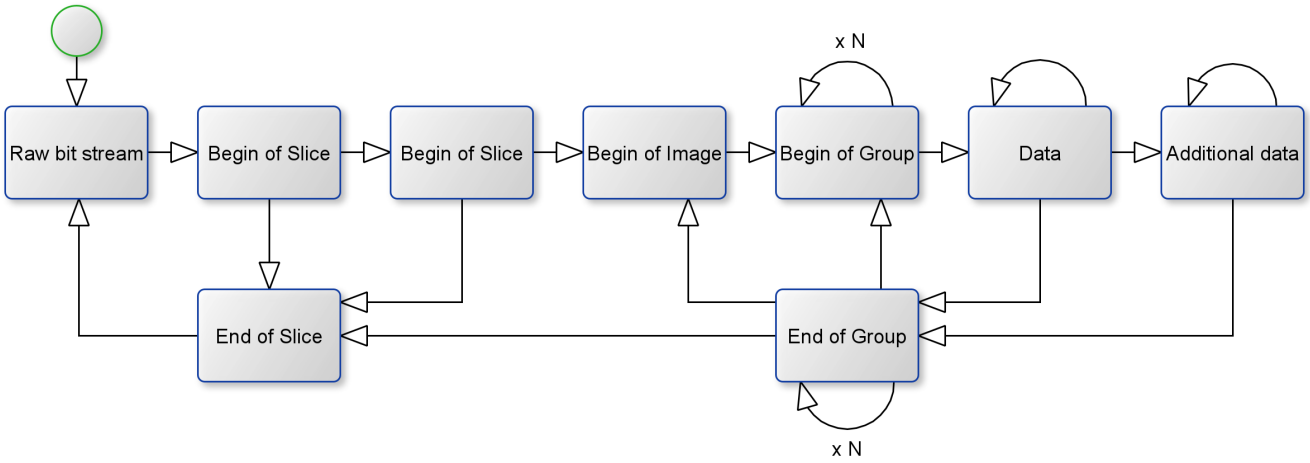


- Timing corrections
- Merging of data from different sources and sorting in time
- Filter data on time coincidence

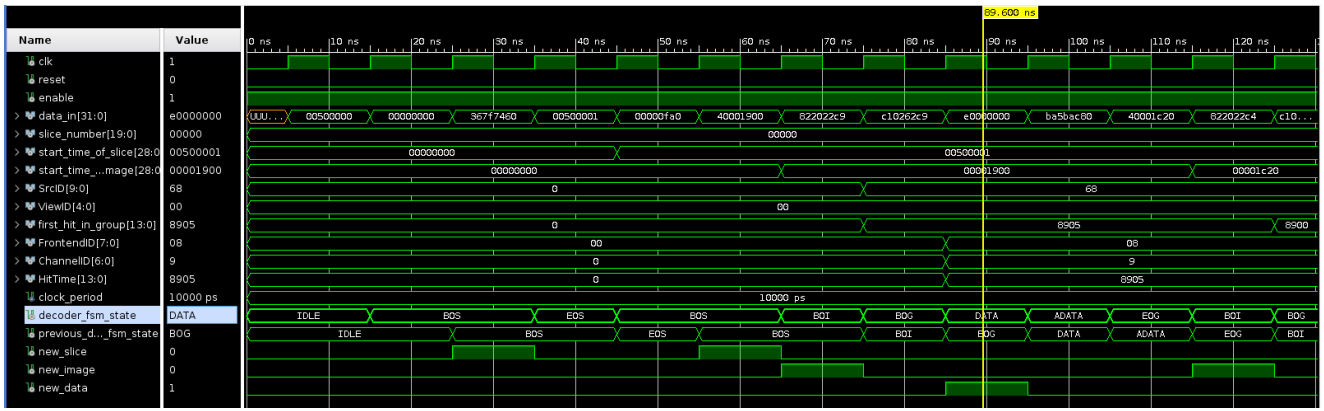
$\rightarrow$ **Trigger Event Candidates**

Raw Data Decoding

Implemented as finite state machine as seperated VHDL IPCore.

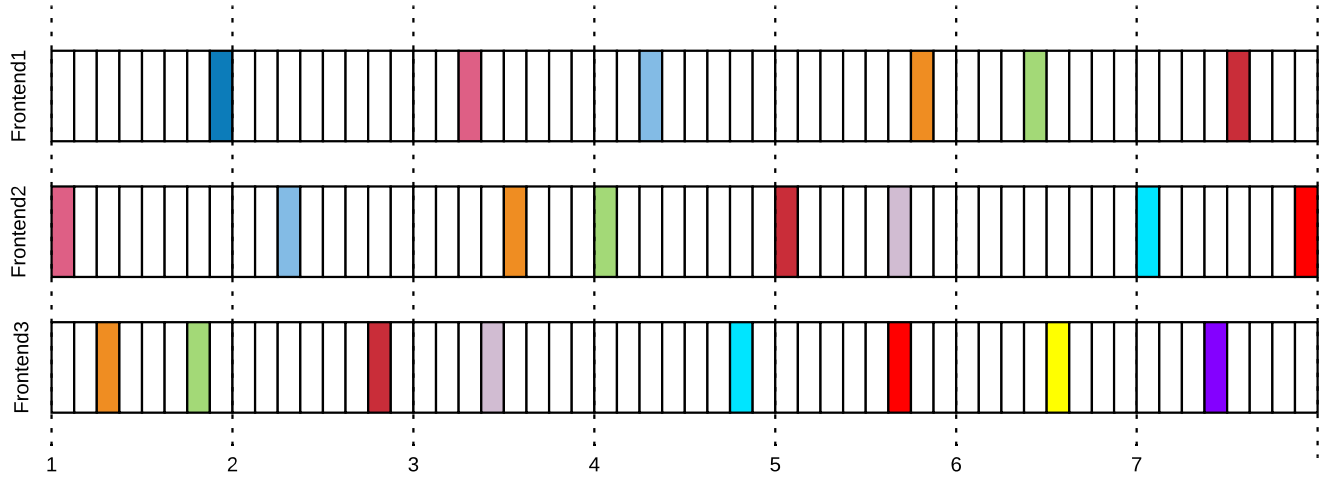


RTL simulation against Martin's C++ implementation shows 100% agreement.



More Details on Timing

Making Trigger decision and saving two consecutive images sounds easy - but how it looks in reality ?



In an continuous read-out data stream we do not have any time constraints a priori - since different detectors and even different elements have different relative latencies to each other.

Online Trigger decision depends on time aligned data

$\rightarrow$ We have to introduce an online correction!

Two Stages of Timing Corrections!

Coarse Timing Correction (CTC)

Shift of Images to make sure two consecutive Images of different front-ends contain correlated information.

- Done of the level of the front-ends
- Analog to classical TDC-delay parameter
- Unit: Multiplies of images-width

Fine Timing Correction (FTC)

Shift of the hit time to align correlated hits at one point in time .

- Done of the level of the Trigger Processor (or Multiplexer?)
- Analog to classical T0 correction
- Unit: resolution of TDC's

Both Corrections need a common time reference!

Time Reference

The start time of the slice and images is synchronized with an central clock but these clock is "free-running "against the physic signals. We have to extract the correlation against the beam time to make data meaningful!

$\rightarrow$ We select a subset of detectors/planes which act as time reference detectors!

Requirements:

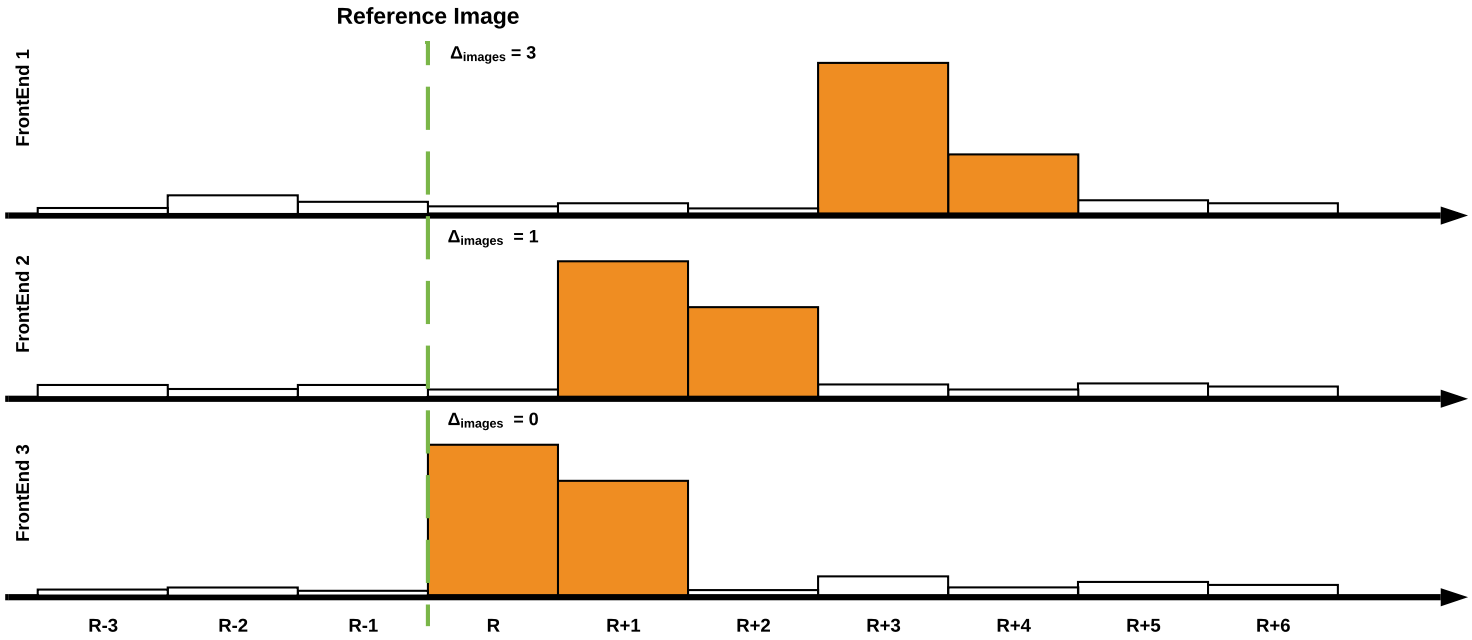
- Very good timing and rate capabilities.
- Detector geometries also defines type of incomming particle acceptance.
  - Selection of Beam Particle: BMS/SciFi (small).
  - Selection of HALO Particle: Vetos/Hodoscopes (large).
- Detector acts as Virtual Monolithic Plane $\rightarrow$ must be calibrated first!

  (Either on the analog side or with special TDC front-end firmware which allows a channel by channel fine calibration).

Any hit in any of these detectors can be used as reference point and therefore represents a beam-time measurement!

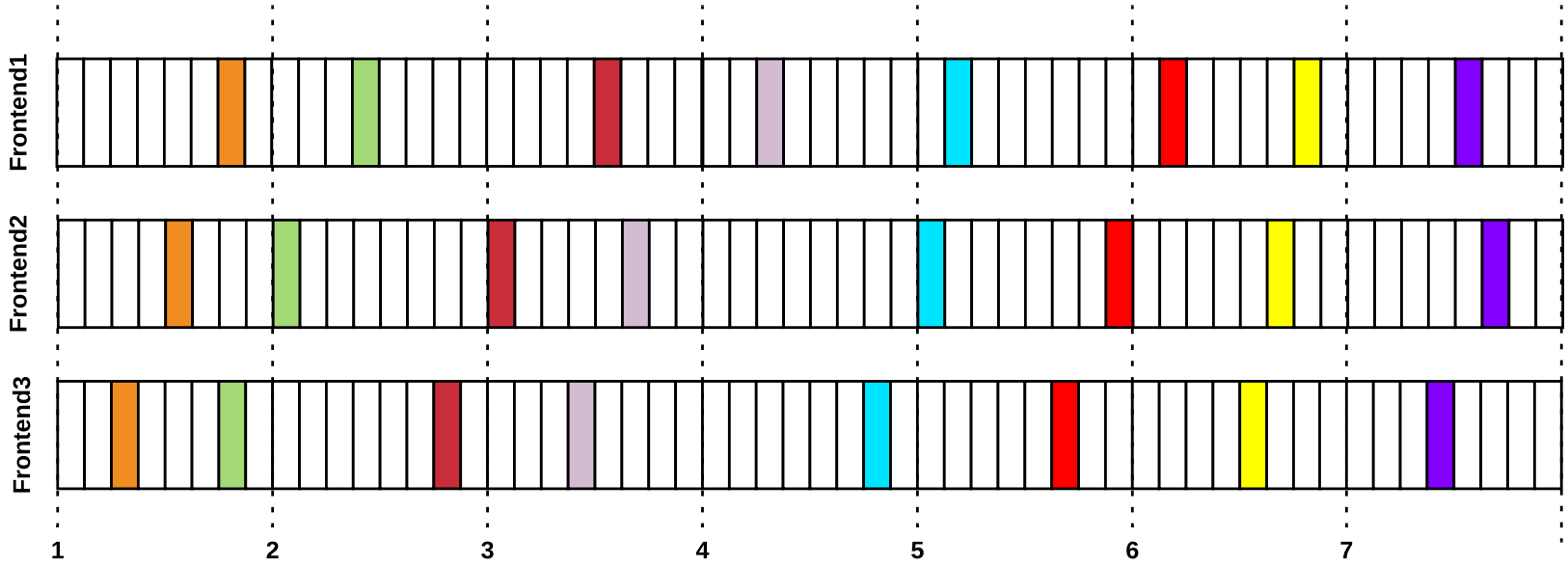
Determination of Time Corrections

The DAQ is in free-running-mode with low intensity beam so that only one beam particle is present in the spectrometer during the length of an image.



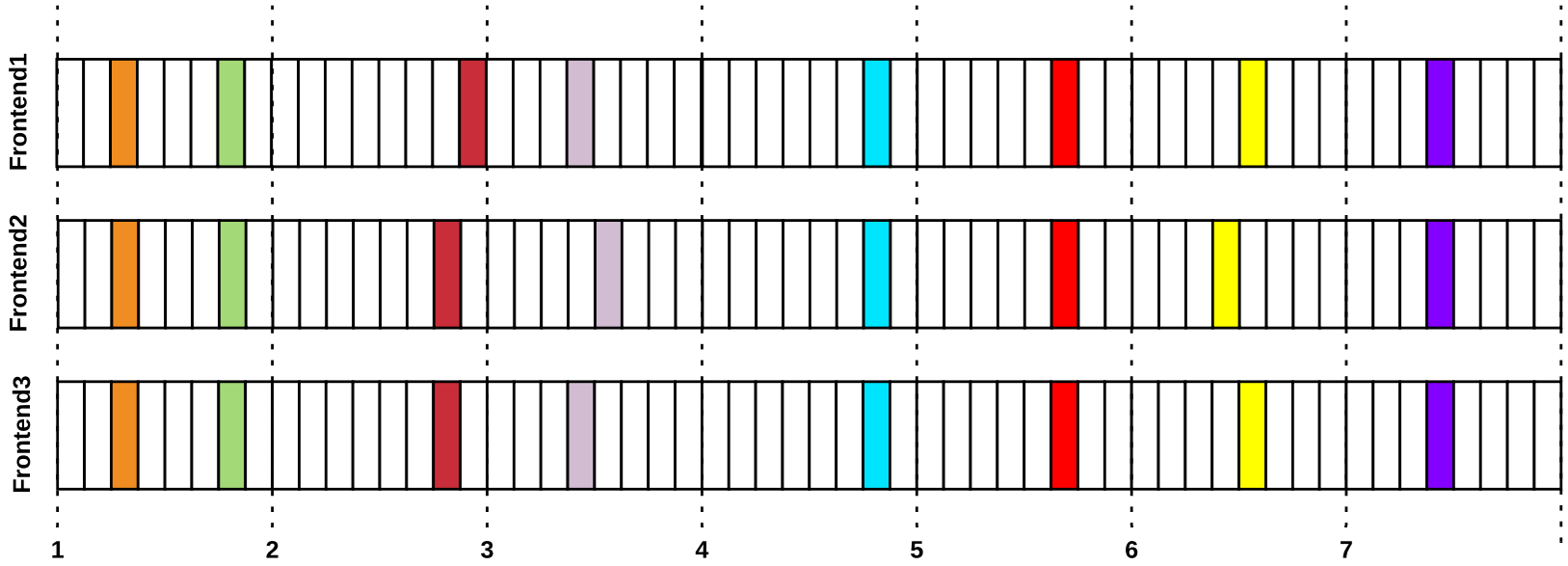
A hit in an image of an reference detector marks the image as start image.
These image number is subtracted from the surrounding images.
$\rightarrow$ the maximum multiplicity shows the CTC value.

Determination of FRC correction



We histogram the difference of the CRC corrected hit-time from the beam-time and extract the most probable values for FRC channel by channel.

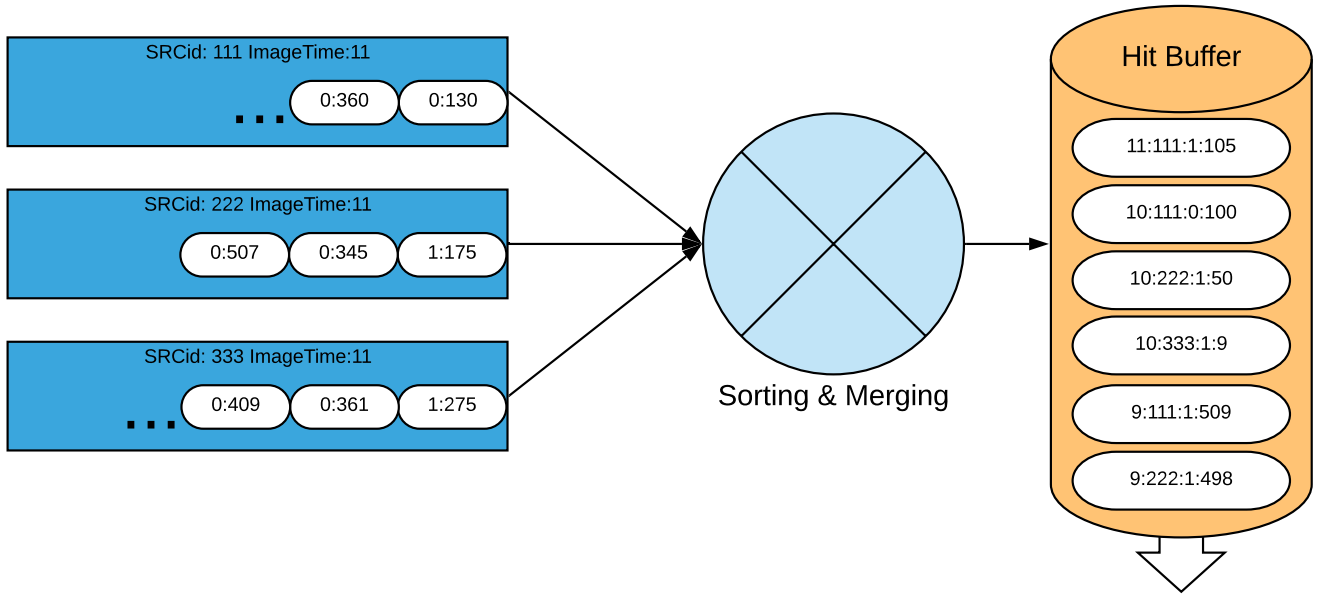
Data Stream after time corrections



Data stream fully calibrated and ready for further processing of Level0 trigger for coincidence detection!

Event-Generator: Sorting of Data

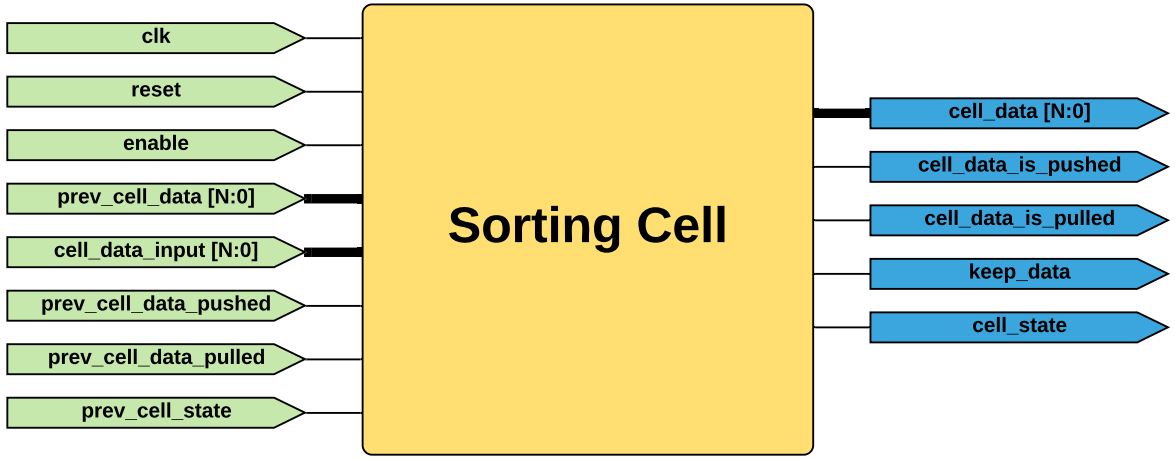
Images from different front-ends are sorted and merged in hit buffer according to global hit time.



Hit buffer (sorting cell memory) holds information of three consecutive images. First images is read-out, second image is hold for sorting and merging with the third image which is coming in.

$\rightarrow$ Going from slice/image structure to hit-based data structure.

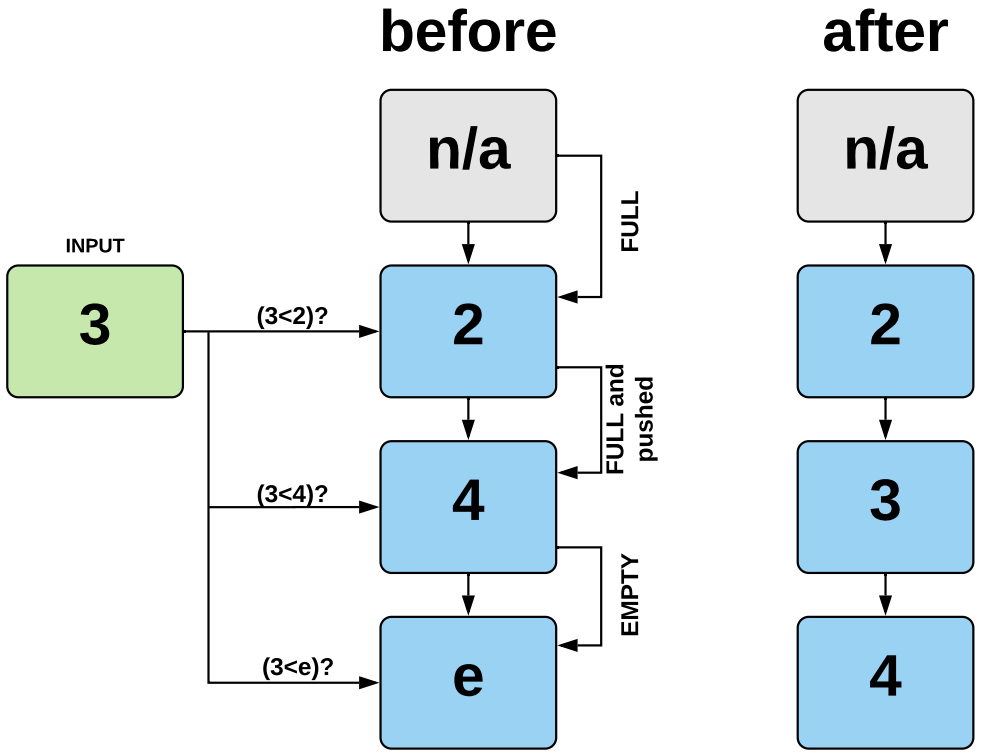
Parallel Sorting Algorithm I



Basic Rule-set:

- If cell is empty, claim the incoming element if the above cell is not empty.
- If a cell is occupied, it will claim the incoming element if the incoming data is less than the stored element AND the occupied cell above don't kick out its element.
- If previous cell kicks out it's stored element, then the current cell MUST claim the above cell's element, regardless of the state of the current cell.
- If a cell is occupied and accepts new data, it must kick out its current element.

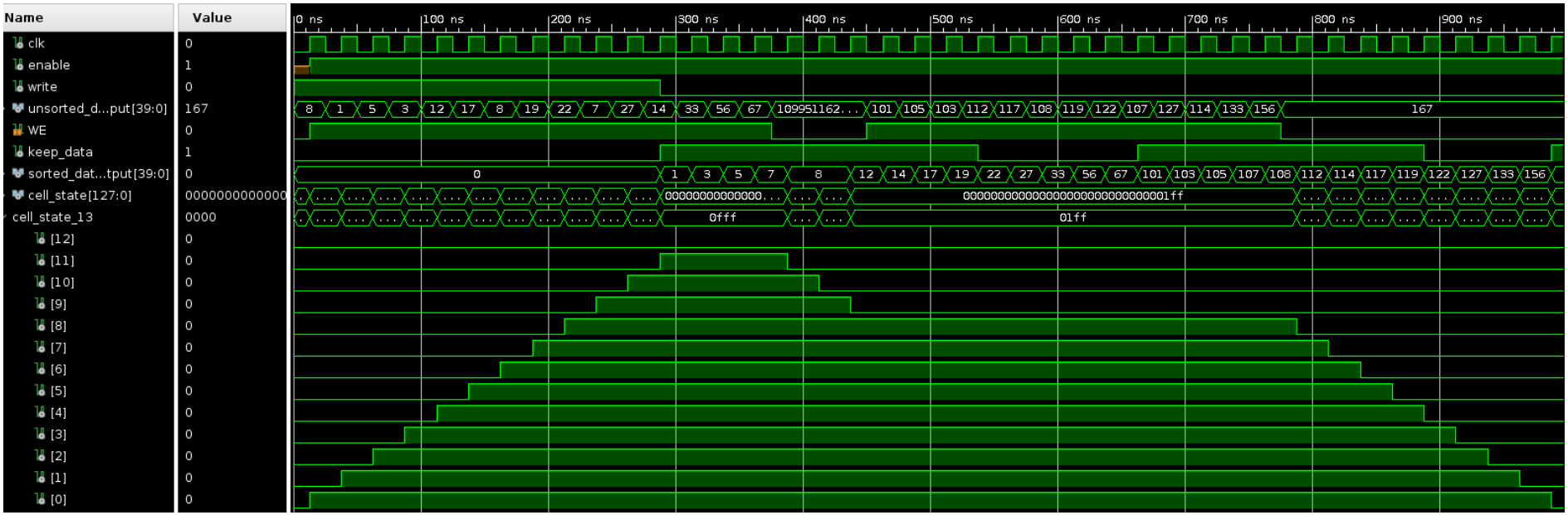
Parallel Sorting Algorithm II



- Running time of O(N) but we buy it with space on FPGA.
- Scheme extended by parallel read/write functionality.
- Marking cells which has neighbors in certain time gate for further processing.

Parallel Sorting Cells Implementation

Test implementation as modular IPCores:



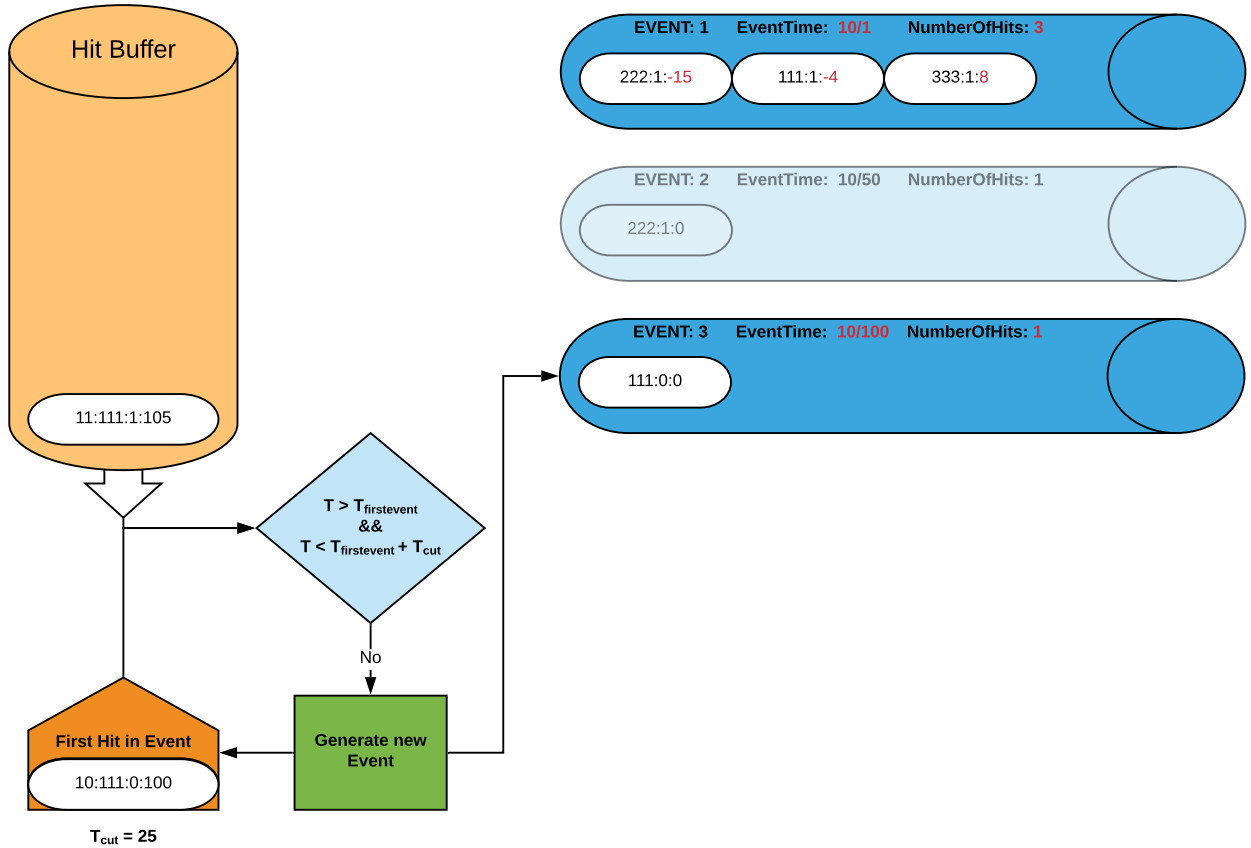
Resources for 128 Sorting cells a 40bit in XCKU040:

- LUT: 12519 (5.16%)
- FF: 5248 (1.08%)

Exact dimensions have to specified from simulations!

Event Building: Generating Events

Output is pushed out every cycle and a time gate is applied to to find coincidence in time to generate Events of Interests.



→ Generated EoI are sent to further trigger-processor stages.

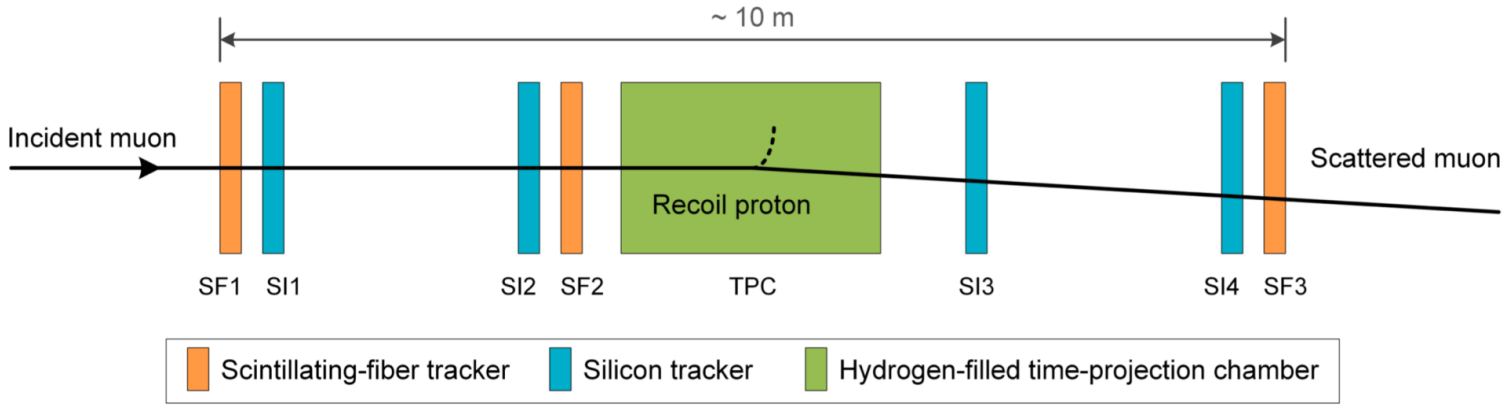
High Level Trigger Processor

Separated FPGA for pattern recognition (L1 Trigger)!

- Pattern recognition on trigger event basis (EOI).
- More Information as before: more detectors, finer granularity.
  $\rightarrow$ More sophisticated trigger decision

Example: Kink-Trigger for Proton Radius:

Measurement of low-Q^2 elastic-scattering of muons on protons at 100 GeV.

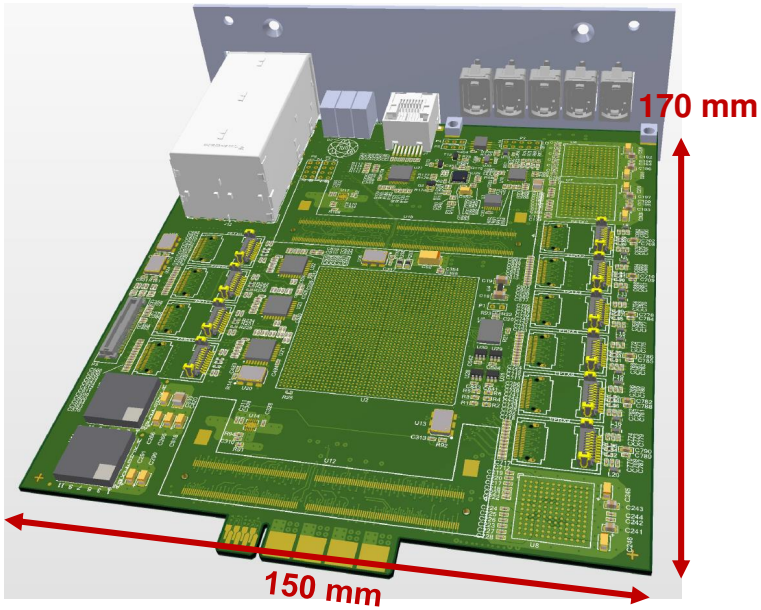


Algorithm: Cluster generation and simple tracking through space points

(e.g. look up table approach in FPGA or direct track reconstruction)

Possible Hardware Platforms

Event-Builder: **iFDAQ XCKU Card** (Developed by TUM):



- Xilinx Kintex UltraScale XCKU095.
- 64 optical links (up to 16 Gb/s):
  - 60 via MPO connectors.
  - 4 via SFP+ connector.
- 2 DDR4 memory interfaces.
  - Up to 16 GB total storage.
- Clock Jitter Cleaning.
- Custom Form factor with crate for two cards.

Trigger Processor: **XCKU-PCIe Card** (like: Xilinx KCU116, Alveo U250 , ...)

- "Cheaper" since no custom hardware.
- PCIe interface could be used to interface with CPUs or GPUs (DMA) and for monitoring/debugging.

or also **ifDAQ XCKU Card**.

Thank You !