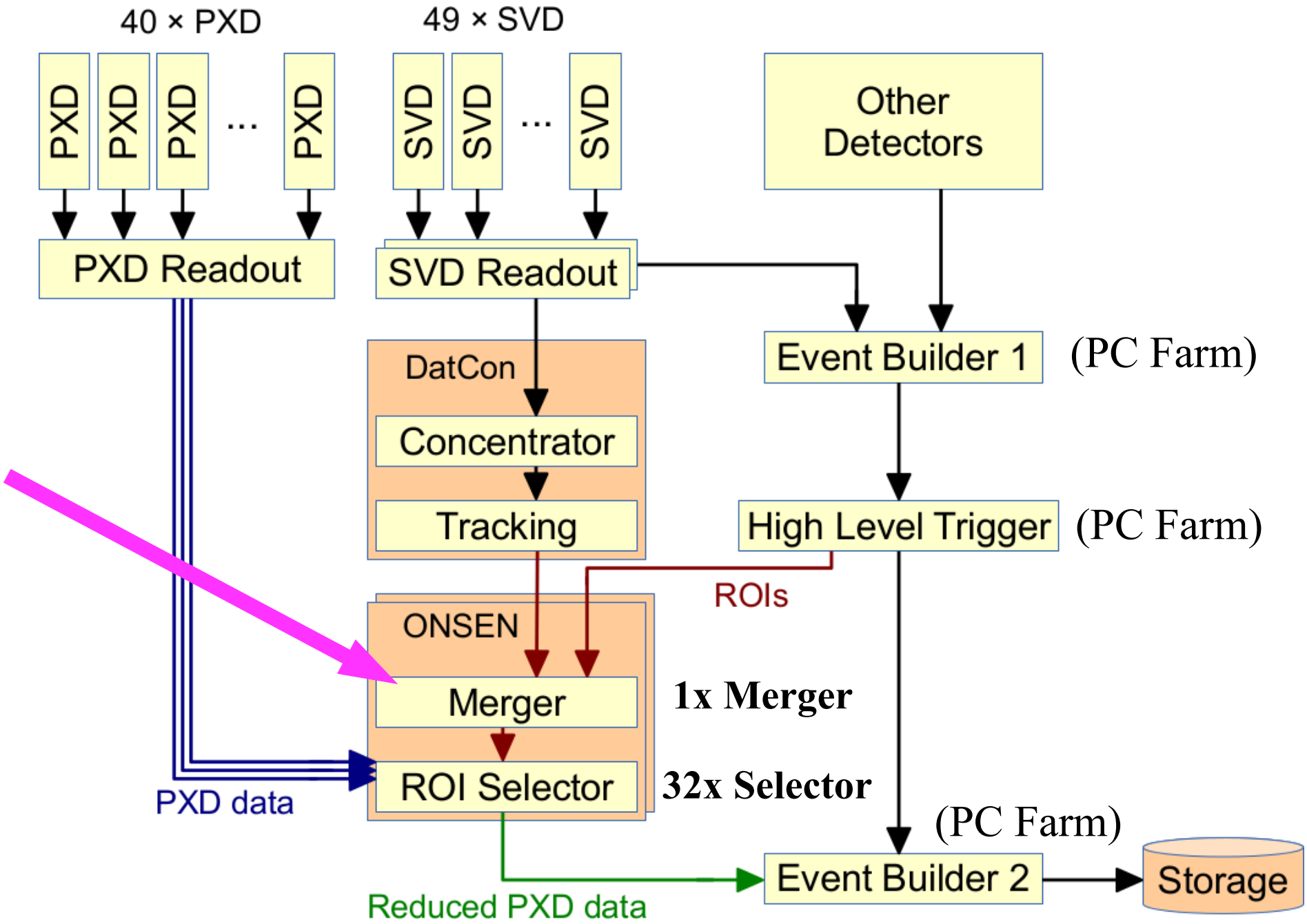
Selected ONSEN Topics - Trigger/ROI Distributor and Software/Firmware Updates

Björn Spruck and Dennis Getzkow

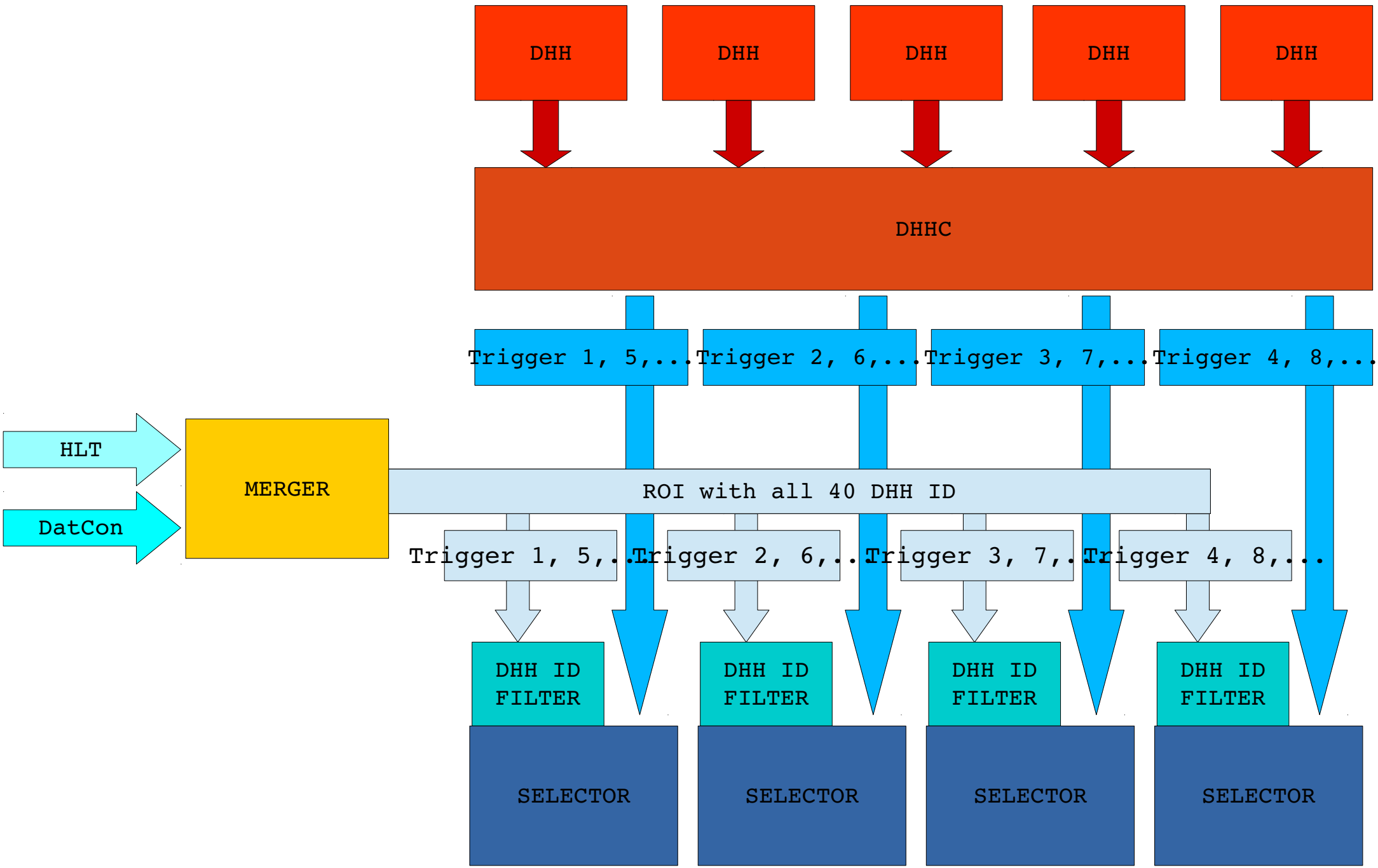
Belle II PXD/SVD Workshop, Pisa
2.10.2014

Belle II PXD DAQ



Load Balancing / Multiplexing by Event Number

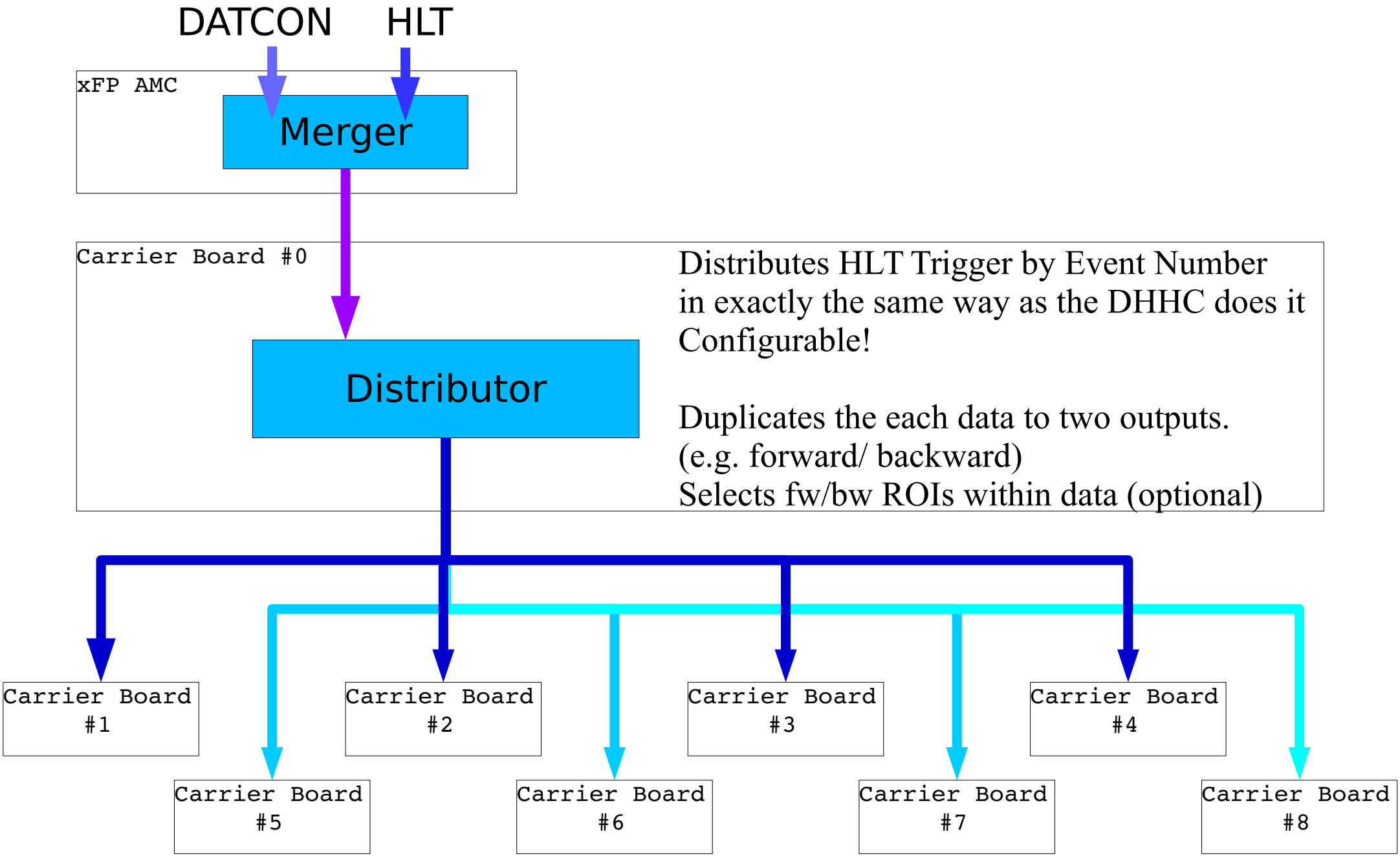
- Not every Selector Node gets PXD data for all triggered events, because
  - DHHC is doing load balancing between 5 inputs to 4 outputs
  - Each node get every 4th event (on average), but with data from 5 DHH (e.g. from 5 half-ladders)
- That means, Trigger data (ROIs) have to be send accordingly from the Merger Node to the 32 Selector Nodes.
- **We use a lookup table with 64 entries on DHHC and Merger Node for selecting the Node. (Can be modified on the fly!)**
- Done in two steps, as it involves the Carrier Boards
- In addition the ROI information can be pre-filtered to save some memory bandwidth as we need only ROIs for always the same 5 half-ladders on one Selector Node
- Setup of a test system to test the firmware for this (Dennis Getzkow)


DHH
DHH
DHH
DHH
DHH
DHHC
Trigger 1, 5,...
Trigger 2, 6,...
Trigger 3, 7,...
Trigger 4, 8,...
HLT
DatCon
MERGER
ROI with all 40 DHH ID
Trigger 1, 5,...
Trigger 2, 6,...
Trigger 3, 7,...
Trigger 4, 8,...
DHH ID FILTER
DHH ID FILTER
DHH ID FILTER
DHH ID FILTER
SELECTOR
SELECTOR
SELECTOR
SELECTOR

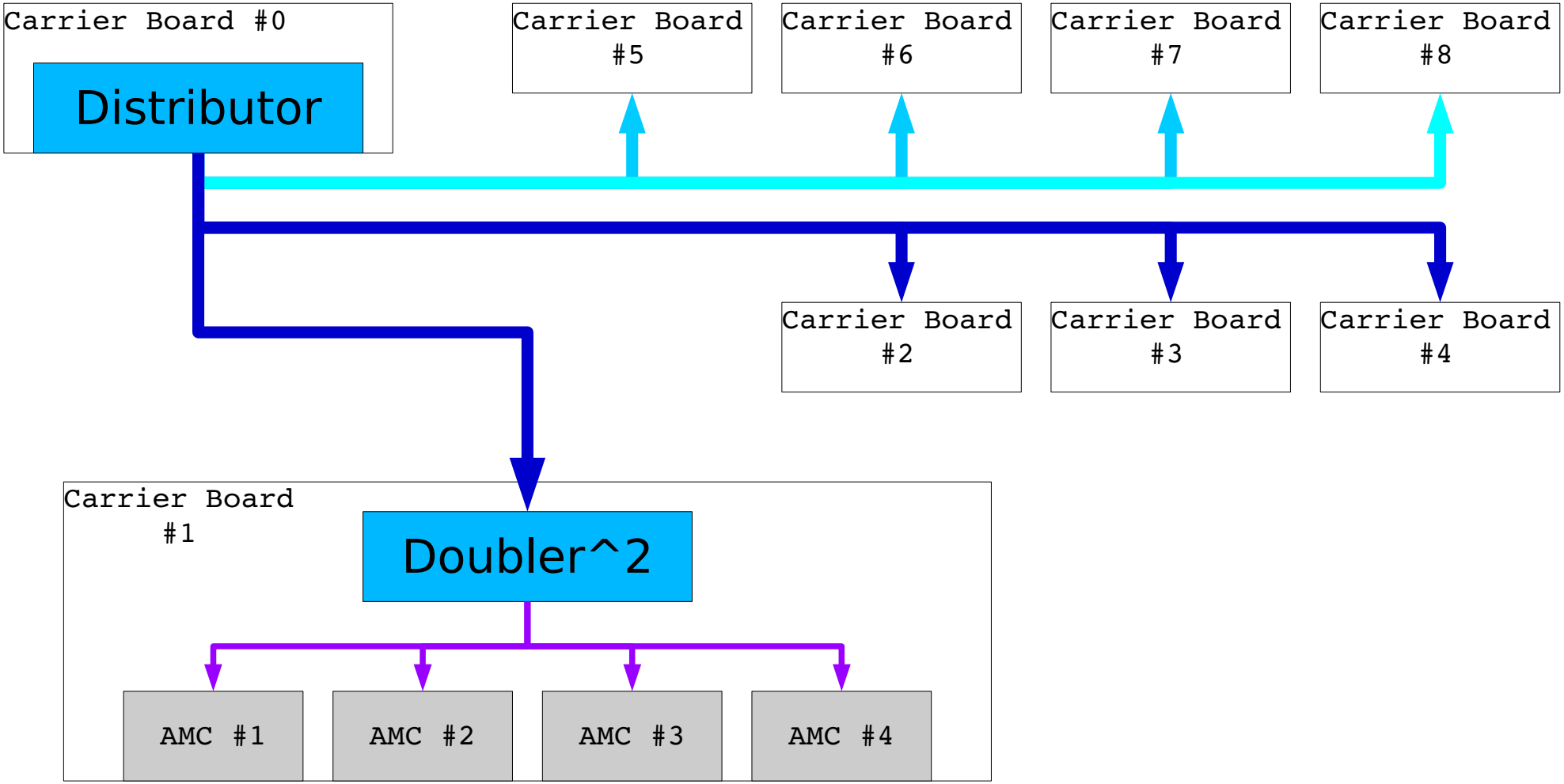
Load Balancing / Multiplexing by Event Number

- Not every Selector Node gets PXD data for all triggered events, because
  - DHHC is doing load balancing between 5 inputs to 4 outputs
  - Each node get every 4th event (on average), but with data from 5 DHH (e.g. from 5 half-ladders)
- That means, Trigger data (ROIs) have to be send accordingly from the Merger Node to the 32 Selector Nodes.
- **We use a lookup table with 64 entries on DHHC and Merger Node for selecting the Node. (Can be modified on the fly!)**
- Done in two steps, as it involves the Carrier Boards
- In addition the ROI information can be pre-filtered to save some memory bandwidth as we need only ROIs for always the same 5 half-ladders on one Selector Node
- Setup of a test system to test the firmware for this (Dennis Getzkow)

Closeup of Merger Carrier – Step 1



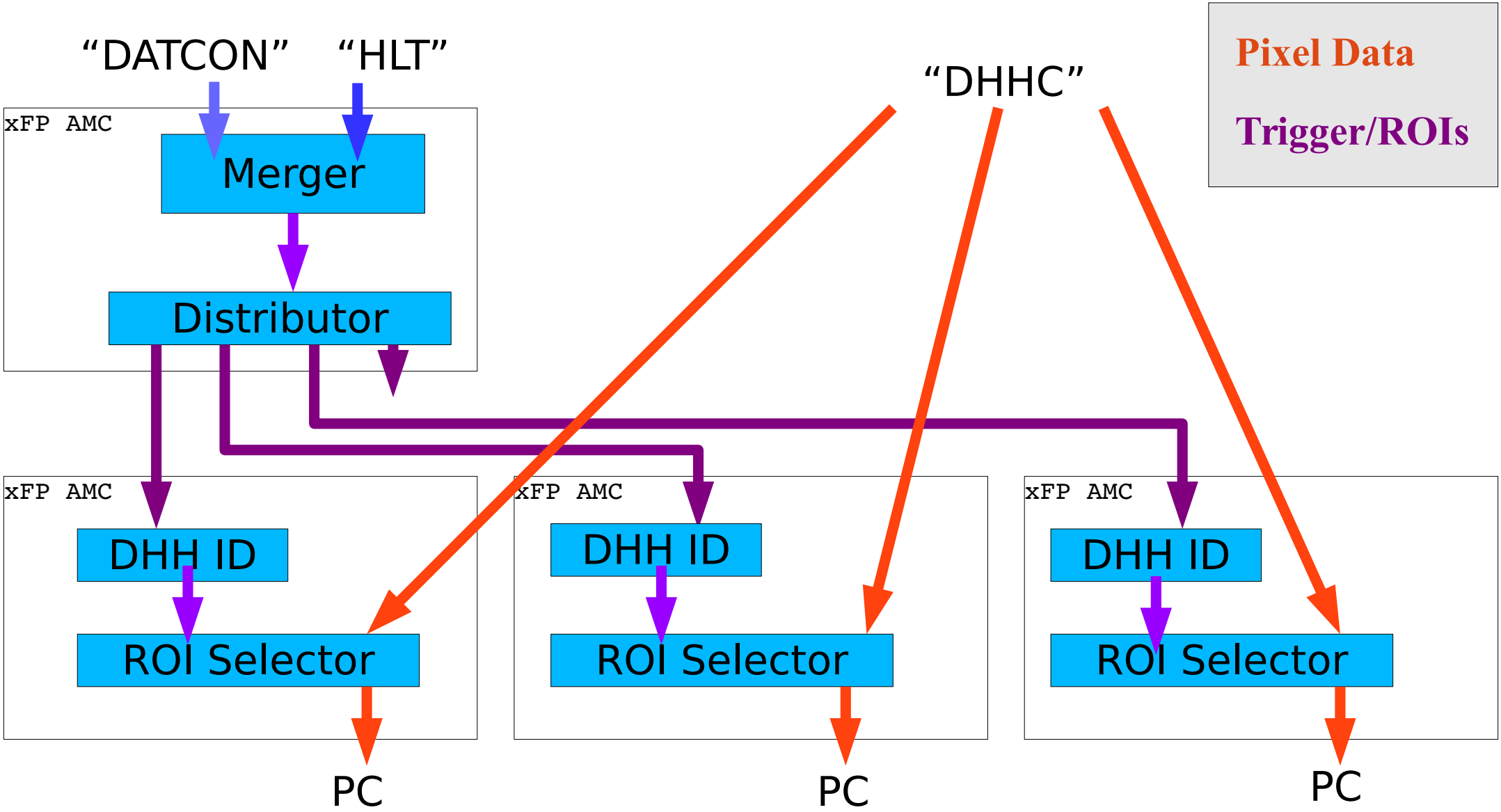
Closeup of Selector Carriers – Step 2



Remark: All AMC in one Carrier process sub-events of the same Event Number.
Thus no further selection necessary needed. The data just has to be clones to four outputs (or doubled two times; doubled like in “DESY test”)
One could select further on the DHH IDs, but thats not needed, data rate is already low.
(nevertheless, it would save memory bandwidth...)

Simplified Test Setup

- 4 AMCs in Carrier Board using passive connectivity between cards.
  - Carrier board FPGA not involved



Results

- Distributor core (standalone) – works
- DHH ID filter core (standalone) – works
- Including Distributor core into Merger firmware – done
- Including DDHH ID filter core into Selector firmware – done
- Testing the “complete” system – ongoing

Data Formats Changes in Firmware and Software

Belle II

- Change of Data Format – done
  - Important for event building (on the Event Builder) and HLT ↔ DHH matching on ONSEN esp. on run start after a “crash”.
  - **Firmware**, **Packer**, **Unpacker changed**
    - Basf2 repository contains working combination, but <u>some issues</u> in data format still remain... (endianess, checksum endianess)
    - Reason: Software performace, endian swapping a lot of data.
  - Verify changes with Event Builder and HLT
  - Full frame unpacking supported in pxdUnpacker
    - PXDRawAdc & PXDRawPedestal objects added to basf2.
    - But: Actual pedestal calculation still missing.
- Change of internal data format to support up to 60 frames per event and multiple buffers per event
  - → test still ongoing

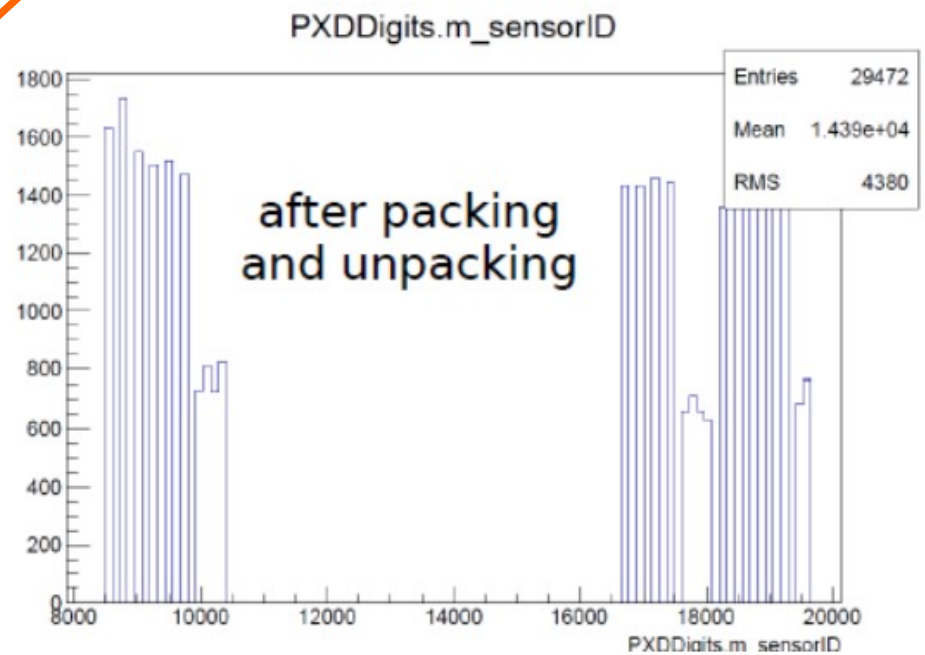
Background Overlay, Raw Data

Background overlay

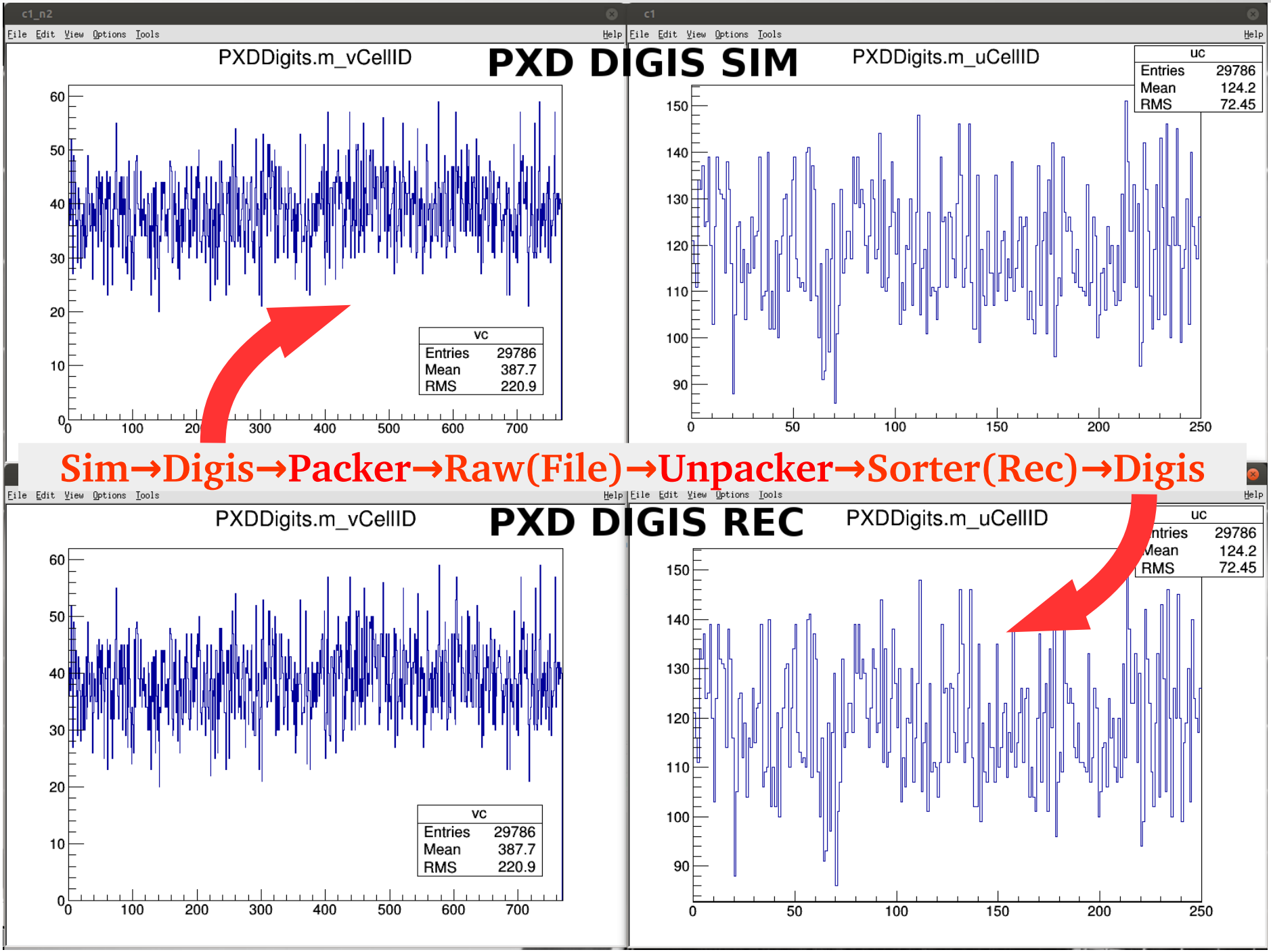
- Merging of simulated signal and real data background
- Called “addbg” at Belle
- ➔ Initial discussion at workshop

Raw data simulation and reconstruction

- ✔ Script for adding packers and unpackers to the module path
- ✔ PXD available, validated
- ➔ Packers and unpackers of other detectors to be added



Packer-Unpacker IO Verification



DQM – More Histograms

ROOT Object Browser

Browser File Edit View Options Tools

Files

Draw Option:

- root
- PROOF Sessions
- ROOT Files
  - your_histo_file.root
    - pxdraw;1
      - hrawPxdPackets;1
      - hrawPxdPacketSize;1
      - hrawPxdHitMapAll;1
      - hrawPxdAdcMapAll;1
      - hrawPxdPedestalMapAll;1
      - hrawPxdCount;1
      - hrawPxdHitMapSensor 1:0:1 (DHH ID $00);1
      - hrawPxdChargeMapSensor 1:0:1 (DHH ID $00);1
      - hrawPxdHitsChargeSensor 1:0:1 (DHH ID $00);1
      - hrawPxdHitsCommonModeSensor 1:0:1 (DHH ID $00);1
      - hrawPxdHitsTimeWindowSensor 1:0:1 (DHH ID $00);1
      - hrawPxdHitMapSensor 1:0:2 (DHH ID $01);1
      - hrawPxdChargeMapSensor 1:0:2 (DHH ID $01);1
      - hrawPxdHitsChargeSensor 1:0:2 (DHH ID $01);1
      - hrawPxdHitsCommonModeSensor 1:0:2 (DHH ID $01);1
      - hrawPxdHitsTimeWindowSensor 1:0:2 (DHH ID $01);1
      - hrawPxdHitMapSensor 1:1:1 (DHH ID $02);1
      - hrawPxdChargeMapSensor 1:1:1 (DHH ID $02);1
      - hrawPxdHitsChargeSensor 1:1:1 (DHH ID $02);1
      - hrawPxdHitsCommonModeSensor 1:1:1 (DHH ID $02);1
      - hrawPxdHitsTimeWindowSensor 1:1:1 (DHH ID $02);1
      - hrawPxdHitMapSensor 1:1:2 (DHH ID $03);1
      - hrawPxdChargeMapSensor 1:1:2 (DHH ID $03);1
      - hrawPxdHitsChargeSensor 1:1:2 (DHH ID $03);1
      - hrawPxdHitsCommonModeSensor 1:1:2 (DHH ID $03);1
      - hrawPxdHitsTimeWindowSensor 1:1:2 (DHH ID $03);1
      - hrawPxdHitMapSensor 1:2:1 (DHH ID $04);1
      - hrawPxdChargeMapSensor 1:2:1 (DHH ID $04);1
      - hrawPxdHitsChargeSensor 1:2:1 (DHH ID $04);1
      - hrawPxdHitsCommonModeSensor 1:2:1 (DHH ID $04);1
      - hrawPxdHitsTimeWindowSensor 1:2:1 (DHH ID $04);1
      - hrawPxdHitMapSensor 1:2:2 (DHH ID $05);1
      - hrawPxdChargeMapSensor 1:2:2 (DHH ID $05);1
      - hrawPxdHitsChargeSensor 1:2:2 (DHH ID $05);1
      - hrawPxdHitsCommonModeSensor 1:2:2 (DHH ID $05);1
      - hrawPxdHitsTimeWindowSensor 1:2:2 (DHH ID $05);1
      - hrawPxdHitMapSensor 1:3:1 (DHH ID $06);1
      - hrawPxdChargeMapSensor 1:3:1 (DHH ID $06);1
      - hrawPxdHitsChargeSensor 1:3:1 (DHH ID $06);1
      - hrawPxdHitsCommonModeSensor 1:3:1 (DHH ID $06);1
      - hrawPxdHitsTimeWindowSensor 1:3:1 (DHH ID $06);1
      - hrawPxdHitMapSensor 1:3:2 (DHH ID $07);1
      - hrawPxdChargeMapSensor 1:3:2 (DHH ID $07);1
      - hrawPxdHitsChargeSensor 1:3:2 (DHH ID $07);1
      - hrawPxdHitsCommonModeSensor 1:3:2 (DHH ID $07);1

Filter: All Files (*.*)

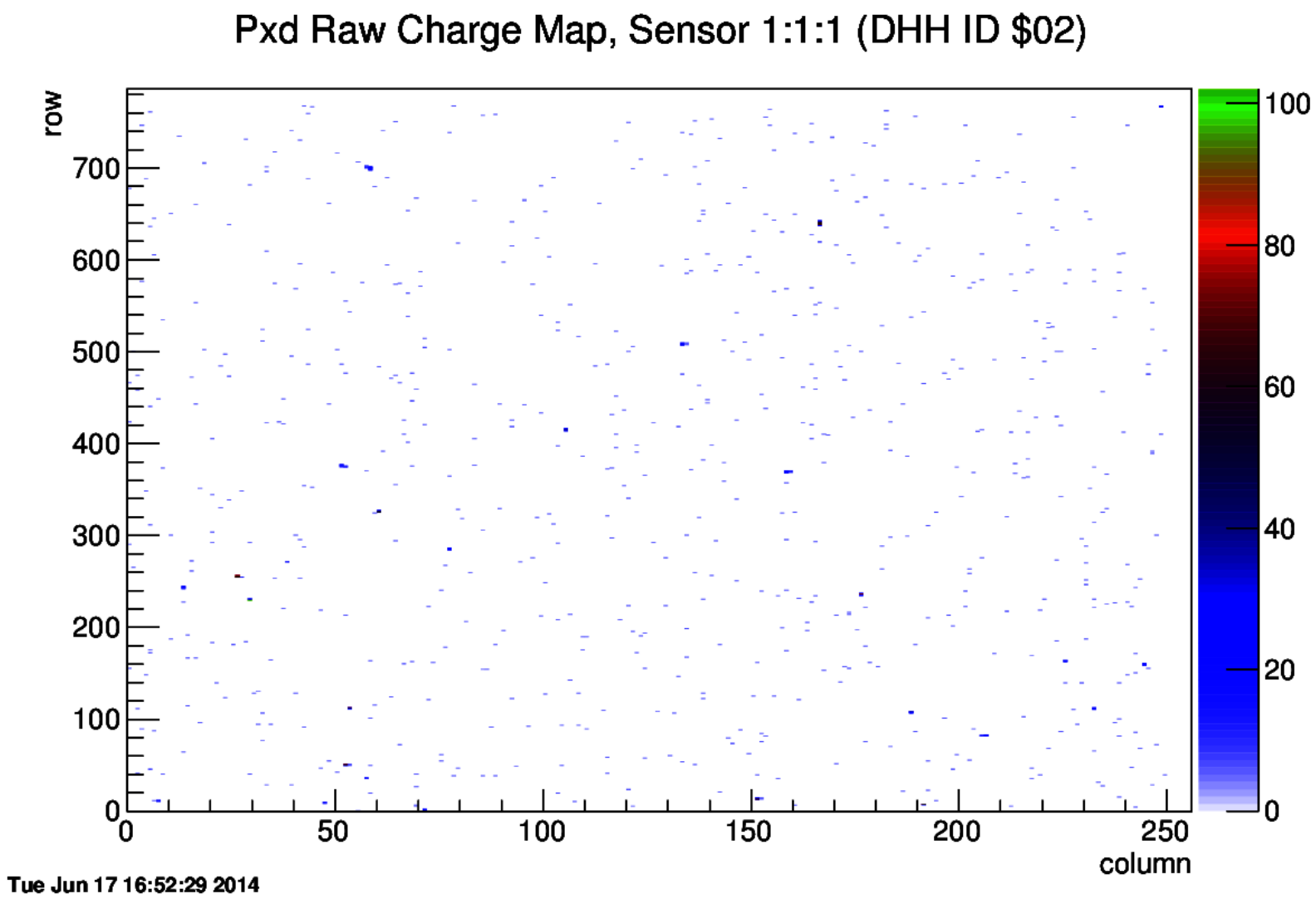
- hrawPxdHitsCommonModeSensor 2:14:1 (DHH ID $3C);1
- hrawPxdHitsTimeWindowSensor 2:14:1 (DHH ID $3C);1
- hrawPxdHitMapSensor 2:14:2 (DHH ID $3D);1
- hrawPxdChargeMapSensor 2:14:2 (DHH ID $3D);1
- hrawPxdHitsChargeSensor 2:14:2 (DHH ID $3D);1
- hrawPxdHitsCommonModeSensor 2:14:2 (DHH ID $3D);1
- hrawPxdHitsTimeWindowSensor 2:14:2 (DHH ID $3D);1
- hrawPxdHitMapSensor 2:15:1 (DHH ID $3E);1
- hrawPxdChargeMapSensor 2:15:1 (DHH ID $3E);1
- hrawPxdHitsChargeSensor 2:15:1 (DHH ID $3E);1
- hrawPxdHitsCommonModeSensor 2:15:1 (DHH ID $3E);1
- hrawPxdHitsTimeWindowSensor 2:15:1 (DHH ID $3E);1
- hrawPxdHitMapSensor 2:15:2 (DHH ID $3F);1
- hrawPxdChargeMapSensor 2:15:2 (DHH ID $3F);1
- hrawPxdHitsChargeSensor 2:15:2 (DHH ID $3F);1
- hrawPxdHitsCommonModeSensor 2:15:2 (DHH ID $3F);1
- hrawPxdHitsTimeWindowSensor 2:15:2 (DHH ID $3F);1
- pxd;1
  - hPXDFired1;1
  - hPXDFired2;1
  - hPXDClusters1;1
  - hPXDClusters2;1
  - hPXDHitmapU1;1
  - hPXDHitmapU2;1
  - hPXDHitmapV1;1
  - hPXDHitmapV2;1
  - h2PXDHitmapUV1;1
  - h2PXDHitmapUV2;1
  - hPXDDigitCharge1;1
  - hPXDDigitCharge2;1
  - hPXDClusterCharge1;1
  - hPXDClusterCharge2;1
  - hPXDSeed1;1
  - hPXDSeed2;1
  - hPXDSizeU1;1
  - hPXDSizeU2;1
  - hPXDSizeV1;1
  - hPXDSizeV2;1
  - hPXDSize1;1
  - hPXDSize2;1
  - hPXDstartRow1;1
  - hPXDstartRow2;1
  - hPXDAverageSeedByStartRow1;1
  - hPXDSeedCountsByStartRow1;1
  - hPXDAverageSeedByStartRow2;1
  - hPXDSeedCountsByStartRow2;1
  - hPXDAverageSeedByU1;1
  - hPXDSeedCountsByU1;1
  - hPXDAverageSeedByV1;1
  - hPXDSeedCountsByV1;1

40 half-ladders
→ 40x histograms

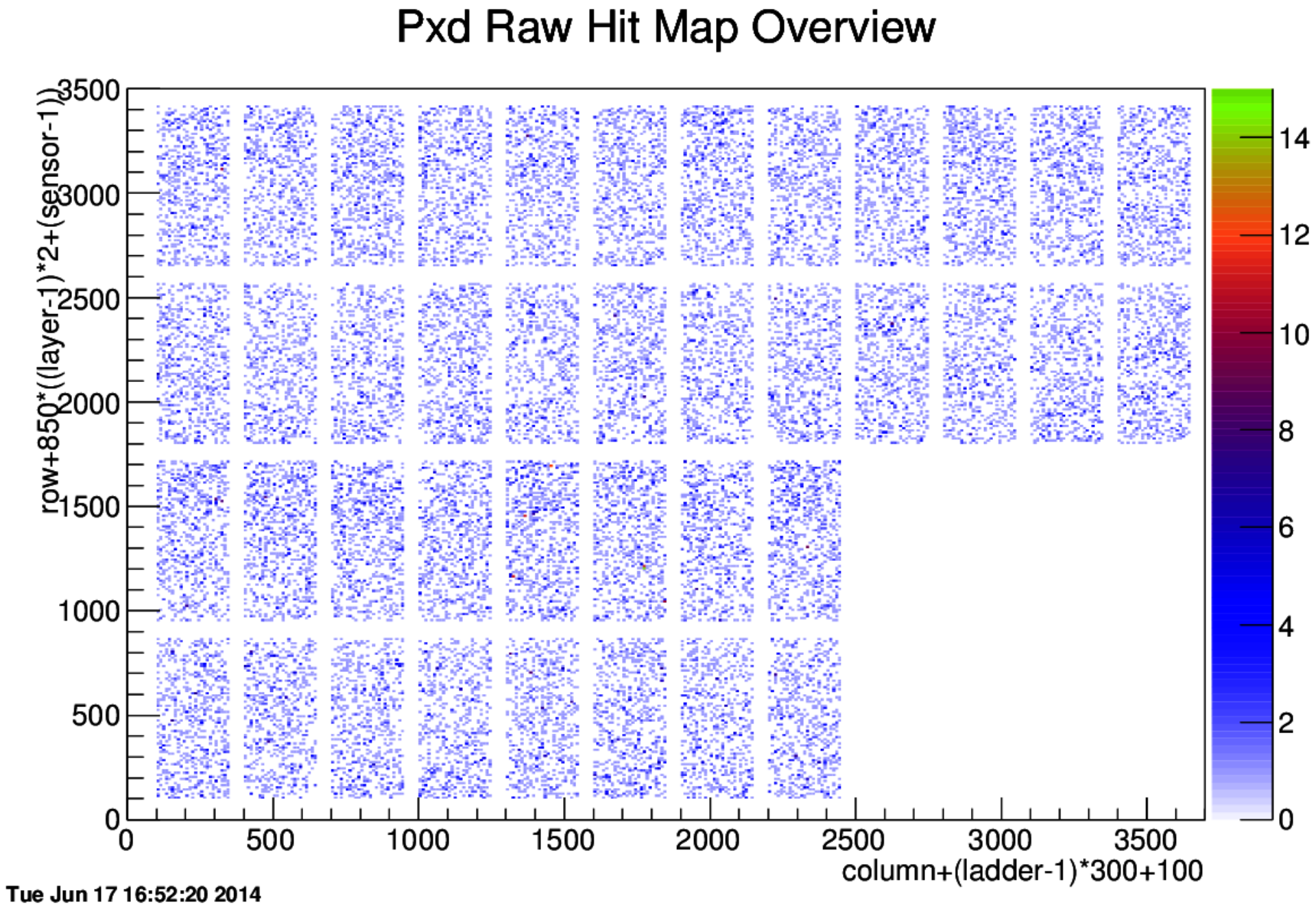
- hPXDClusters2;1
- hPXDHitmapU1;1
- hPXDHitmapU2;1
- hPXDHitmapV1;1
- hPXDHitmapV2;1
- h2PXDHitmapUV1;1
- h2PXDHitmapUV2;1
- hPXDDigitCharge1;1
- hPXDDigitCharge2;1
- hPXDClusterCharge1;1
- hPXDClusterCharge2;1
- hPXDSeed1;1
- hPXDSeed2;1
- hPXDSizeU1;1
- hPXDSizeU2;1
- hPXDSizeV1;1
- hPXDSizeV2;1
- hPXDSize1;1
- hPXDSize2;1
- hPXDstartRow1;1
- hPXDstartRow2;1
- hPXDAverageSeedByStartRow1;1
- hPXDSeedCountsByStartRow1;1
- hPXDAverageSeedByStartRow2;1
- hPXDSeedCountsByStartRow2;1
- hPXDAverageSeedByU1;1
- hPXDSeedCountsByU1;1
- hPXDAverageSeedByV1;1
- hPXDSeedCountsByV1;1
- hPXDAverageSeedByU2;1
- hPXDSeedCountsByU2;1
- hPXDAverageSeedByV2;1
- hPXDSeedCountsByV2;1
- hrawROIcount;1
- hrawROIytpe;1
- hrawROIHLTmap;1
- hrawROIHLTsize;1
- hrawROIHLT_DHHID;1
- hrawROIHLTrow1;1
- hrawROIHLTrow2;1
- hrawROIHLTcol1;1
- hrawROIHLTcol2;1
- hrawROIDCmap;1
- hrawROIDCsize;1
- hrawROIDC_DHHID;1
- hrawROIDCrow1;1
- hrawROIDCrow2;1
- hrawROIDCcol1;1
- hrawROIDCcol2;1
- /

Filter: All Files (*.*)

Just one example – Charge Maps



Hit Maps


Pxd Raw Hit Map Overview
row+850*((layer-1)*2+(sensor-1))
column+(ladder-1)*300+100
0
500
1000
1500
2000
2500
3000
3500
0
2
4
6
8
10
12
14
Tue Jun 17 16:52:20 2014

Open Issues

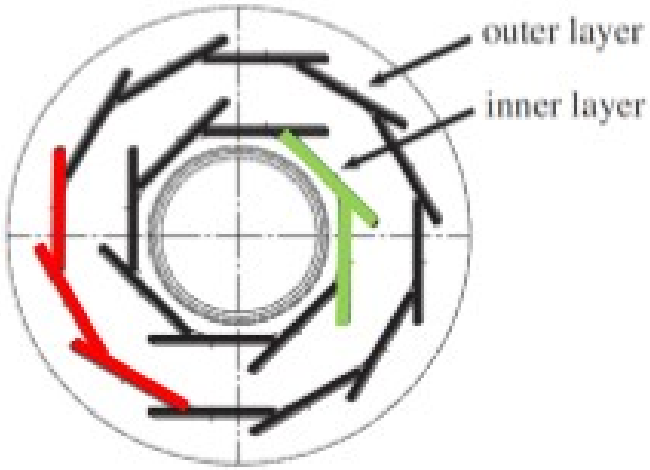
- Firmware
  - Clustered data (new format!) not supported.
    - Unpacking of FPGA based Clusterizer within Selector Core not started yet.
- Software
  - Clustered data not supported yet.
    - Unclear how to interface with the cluster objects in basf2
  - Pedestal calculation in basf2 missing
  - DQM for RAW data
    - Monitor 8M channels (pixels)... not really possible within DQM framework (~20GB/task)

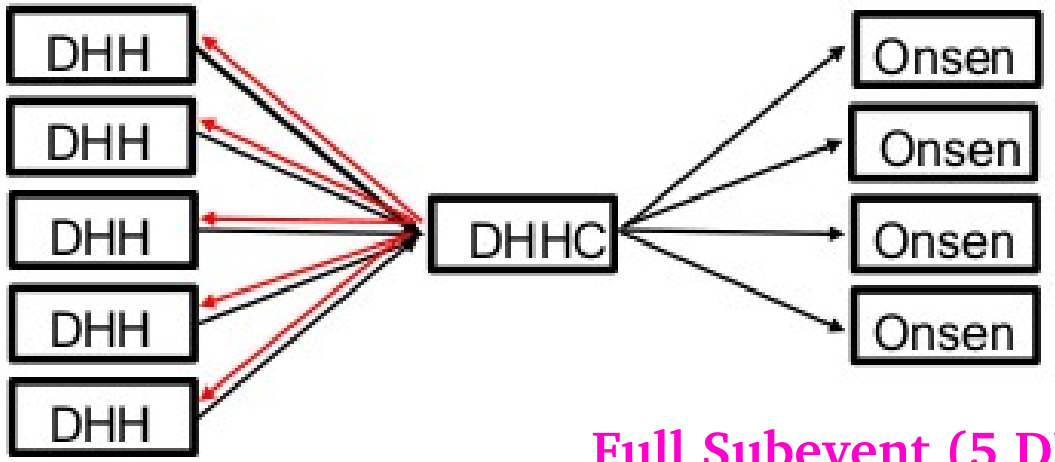
Backups

Data Load Balancing

- Proposal to average data for 5 half ladders:
  - 2 inner half ladders
  - 3 outer half ladders on opposite side
- Data rate reduction:

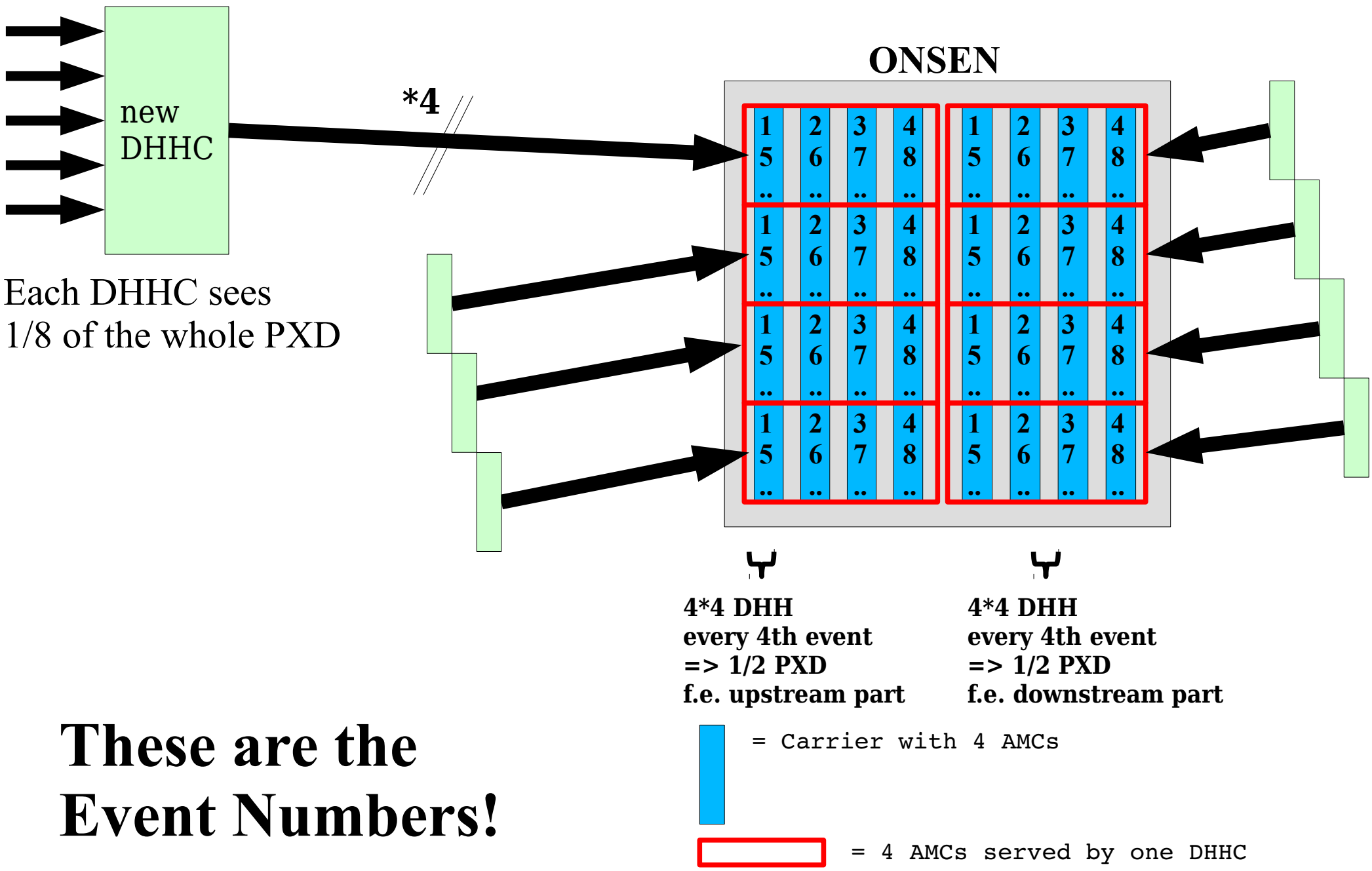
Rate = (2x1 + 3x0.5)/5 = 0.7





Full Subevent (5 DHH) to one output. Selected by trigger number!

Event Wise View



These are the Event Numbers!

Load Balancing / Multiplexing by Event Number

Belle II

- Be flexible, don't use “just” the lower two bits as selection.
- The following was decided by Igor, Dima and Björn:
- **We use a lookup table with 64 entries.**
  - 6 lower bits of trigger number as index in table which selects then one of the four outputs
  - Table can be changed “on the fly“ by Slow Control
  - Easily adaptable if we cannot use all four outputs in a test
  - Deactivate one output w/o loosing data
  - Generate lower or higher data rates on one output
- Because of the delayed HLT trigger, modifying the table is only possible during a stopped DAQ (e.g. HLT queue/fifos must be completely empty!)

Results

- Implemented in VHDL including PLB registers for LUT configuration
- Tested with 1GEvents (32GB) with different LUT setup
  - Checksums, events number selection and continuity are O.K.
- SITCP in → Distributor → (SITCP out 1, nirvana, nirvana, SITCP out 2) on an AMC board (Virtex 5).
- But
  - Not checked with four **real** outputs (can only use two or three for sitcp)
  - Not checked on Virtex 4, not checked on carrier board (connectivity...)
  - Not checked with backplane (AURORA)

Forward/Backward ROI Sorting

- ROIs for forward/backward part of PXD are distinguishable by their DHH-ID
  - We can reduce the HLT data which is send to each node by sending only ROI data which the node needs
  - Remark: Data rate is already low, just double the output (aka DESY test) will work, too.
  - But: We would save some memory bandwidth on the Selector nodes.
- Results:
  - Works. Checked with 1,6 GEvts Input (=570 GB) → 2*400 MEvts Output (=2*70 GB), Rate ~112 MB/s Input
  - Checksum is recalculated as HLT/ROI content of package changes by selecting FW/BW ROIs only.
- But:
  - Not checked with eight **real** outputs (can only use two or three for sitcp)
  - Not checked on Virtex 4, not checked on carrier board (connectivity...)
  - Not checked with backplane (AURORA)

Data Formats - in Details

- (Major) changes:
  - HLT frame in front of DHHC Start frame; contains trigger nr and run nr (for "HLT trigger without DHH data"); + Endianess of Header
    - Change affects Event Builder 2 !!!
  - Adding run nr and subrun nr to DHHC frames (needed for matching when run nr changes)
    - Change affects DHH(C).
  - Adding run nr and subrun nr to HLT and DATCON data (needed for matching when run nr changes)
    - Change affects DATCON and HLT code.
  - Endianess of data (only on output). Adjust such that we dont have to do it in software (saves some CPU time)
    - Endianess of checksums calculation? (DHH)
- Small changes not discussed here...

Unpacker, Packer, Data Formats

- Change of Data Format proposed (and accepted)
  - Important for event building (on the Event Builder) and HLT ↔ DHH matching on ONSEN esp. on run start after a “crash”.
  - Unpacker changes (besides the firmware)
  - → pxdUnpacker renamed to pxdUnpackerDesy1314 for test beam data
  - Overhaul of unpacker code (conventions, naming, documentation etc)
  - pxdPacker changed accordingly
  - Repository contains working combination, but some issues in data format still remain... (endianess, checksum endianess)
    - (ONSEN output → Event Builder → Storage/Unpacking)
- Clustered data
  - Unpacking of FPGA based Clusterizer not started.
  - Data format not finalized yet(?)
  - Unclear how to interface with the cluster objects in basf2

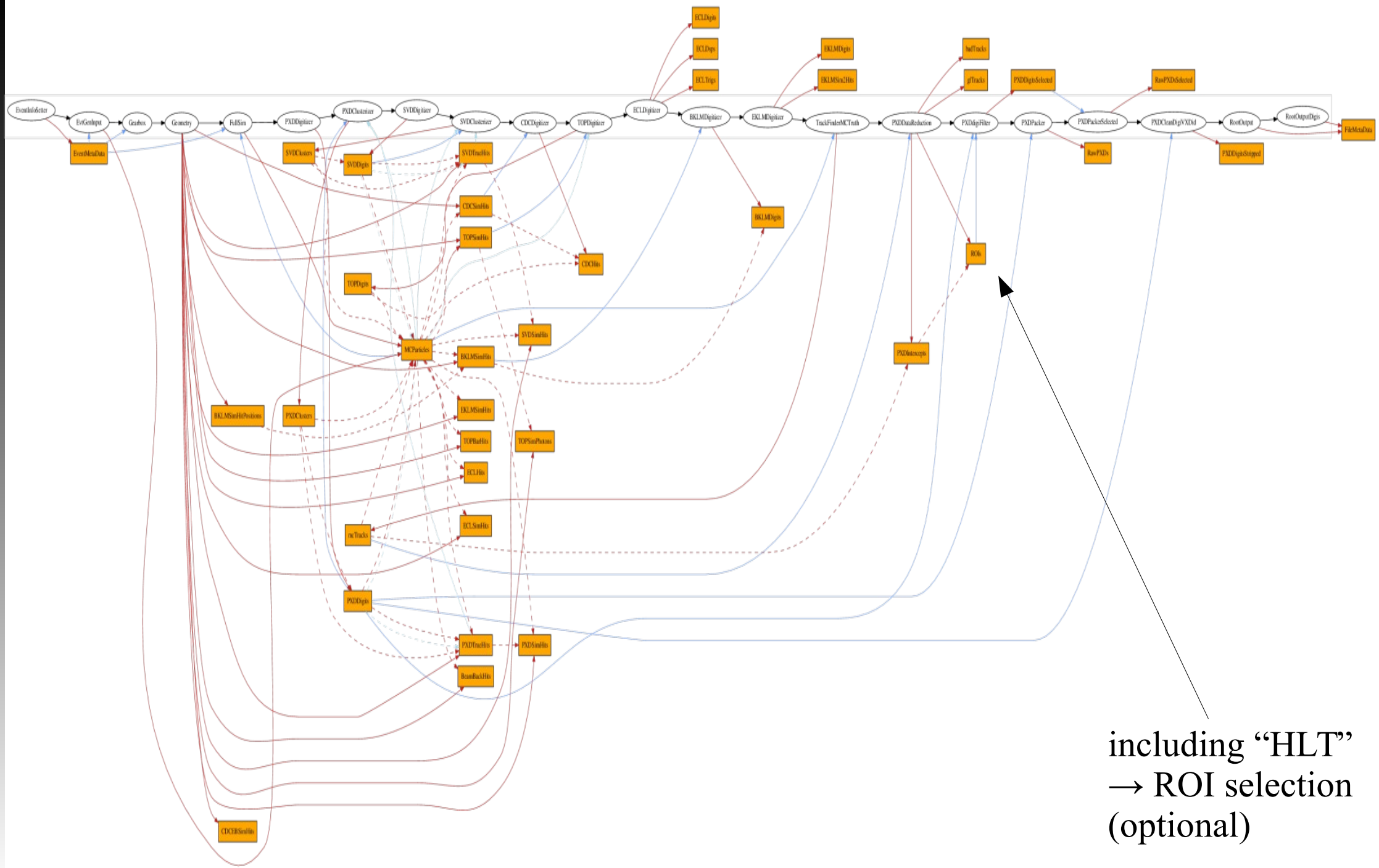
Full Frames

- Full Frames needed for pedestal calculation
- Will be written within the normal data stream
  - DHHC → ONSEN → Event Builder 2
  - Basf2 unpacking into special objects (PXDRawAdc/Pedestal)
- But where to calculate the pedestals?
  - Express Reco (DQM) is not getting all events
  - Will by chance get or miss the event with the Full Frames
  - Calculation need several FF → thus we have to collect them over some time at **one** place (like DQM histograms!)
- Solution?
  - Flag the event (Trigger Bits?) → Tell the event builder to always send it to Express Reco(?). Or better somewhere else?
- **Who?**

PXD Packer

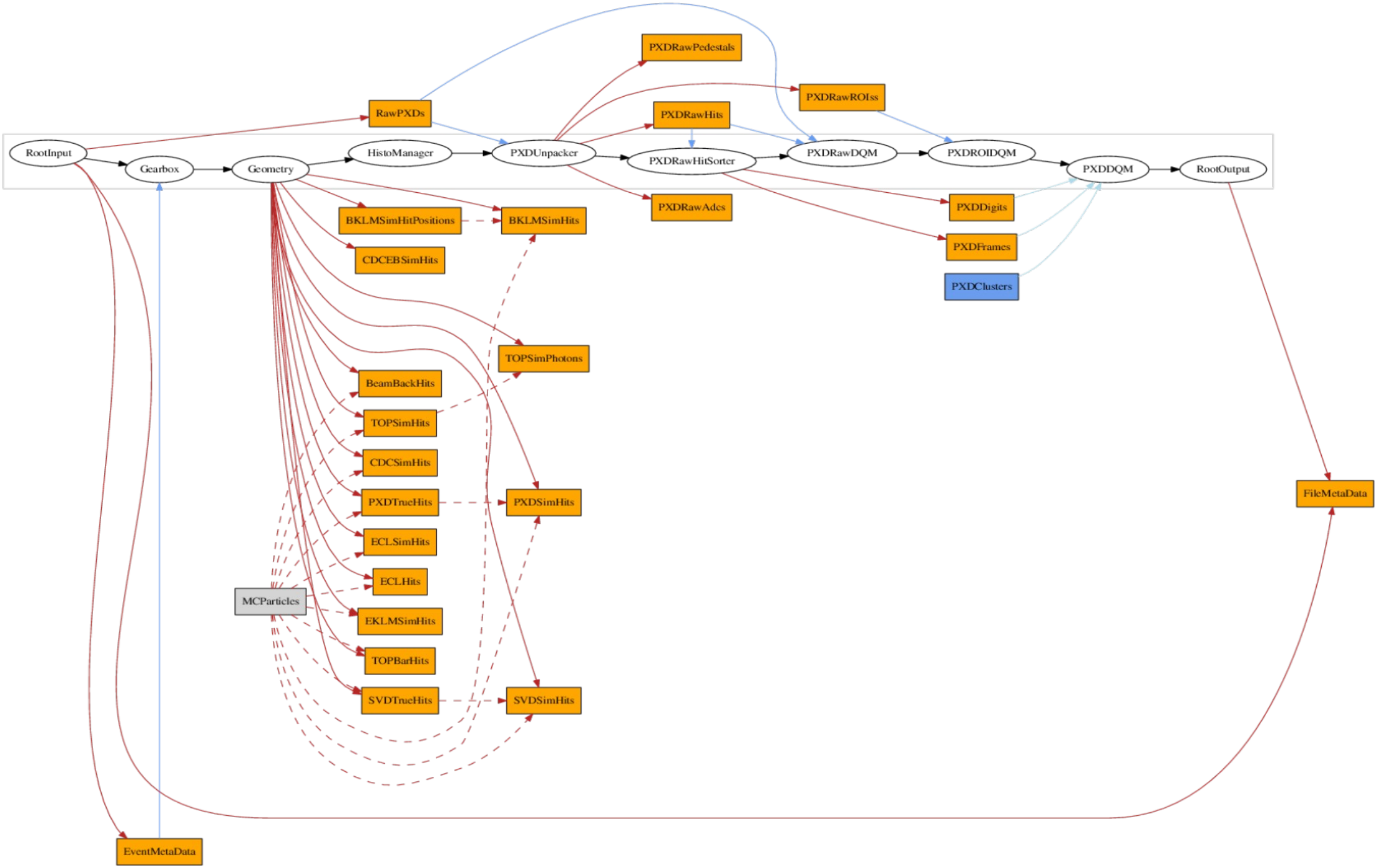
- Request for Packer on last software meeting (B2GM Jan 2014).
  - Take (simulated) Pxd Hits ("Digis") and create a data stream for ONSEN input.
  - Reasons:
    - Use simulated data as input for ONSEN.
    - Reprocess data with new tracking/HLT code.
    - (all this could be done in plain software, but thats not the point)
- Using our existing simulation code, the first version could be checked in the same day.
- (small) Problem: Pixel sorted by VxdId , but VxdId is not the same for all pixels in one half-ladder → time consuming repetitive iterating over PxdDigits data category.
- Todo: Read from framework which detectors/half ladders are existing. Which DHH are connected to which DHHC?
- Todo: 32 ONSEN inputs means to write 32 separate streams.

“Full Chain”: Sim → Raw



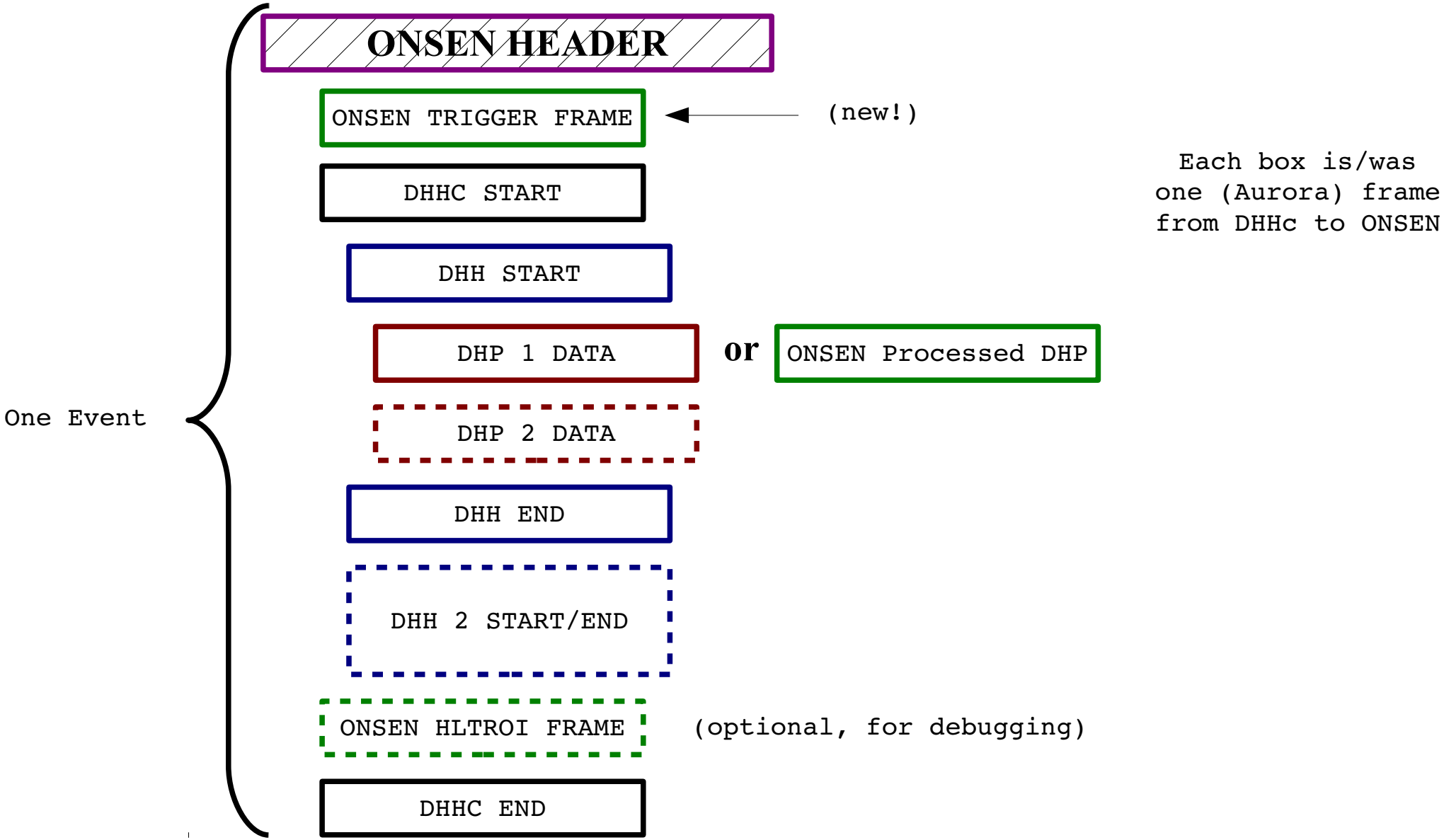
including “HLT”
→ ROI selection
(optional)

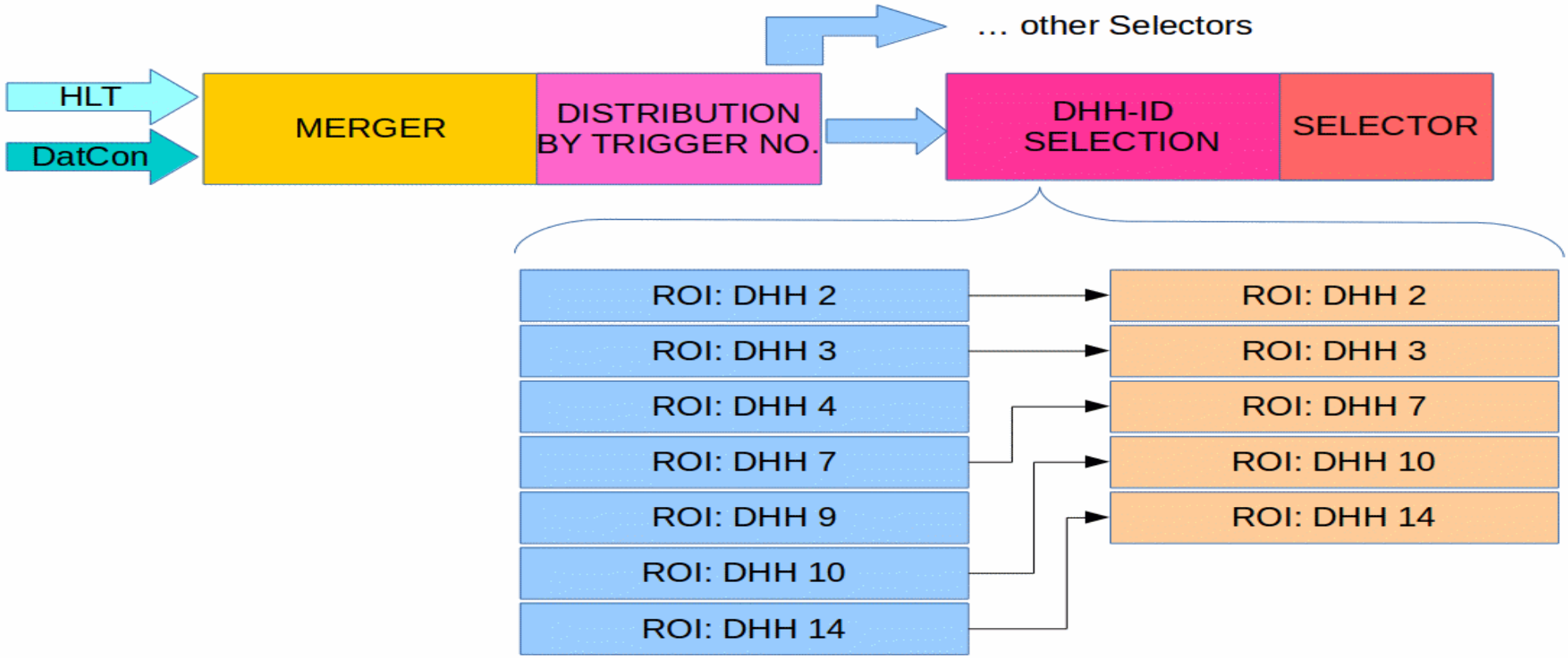
“Full Chain”: Raw → Rec



Data Format

- We use “DHH/DHHC” data format as output of ONSEN, too
- Output/Splitter Node adds header with frame sizes (no “frames” in TCP/IP)



ONSEN: Selection by Trigger No. and DHH ID

- Merger distributes ROI in dependence of Trigger Number for that we use LUT with 64 entries
- Each Selector gets ROI with specific DHH ID