CWG4 data model

2015-11-25

M.Krzewicki

FIAS

CWG4: data model

- Goals:
  - Define and coordinate the implementation of the data model:
    - data formats for FEE, time frames, control, monitoring…
  - Touches the topics of data layout/transport/storage mechanisms:
    - (de-)serialisation, merging etc.

- Group meetings being restarted next week (december 1st at 14:00).
- experts form DAQ, HLT, detectors and PWGs (so far).
- Initially planned bi-weekly meetings.

O2 data flow

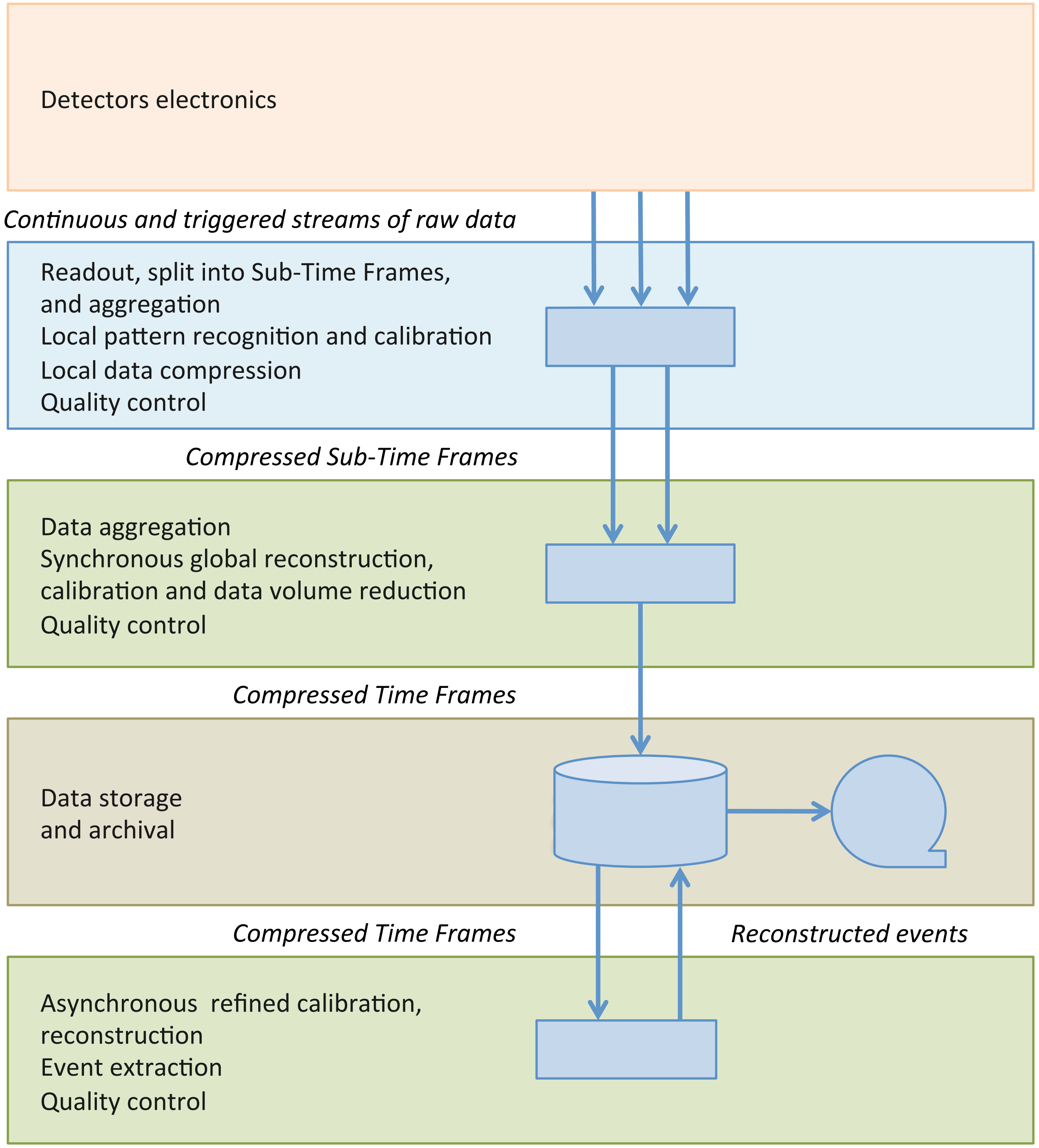

Detectors electronics
Continuous and triggered streams of raw data
Readout, split into Sub-Time Frames, and aggregation
Local pattern recognition and calibration
Local data compression
Quality control
Compressed Sub-Time Frames
Data aggregation
Synchronous global reconstruction, calibration and data volume reduction
Quality control
Compressed Time Frames
Data storage and archival
Compressed Time Frames
Reconstructed events
Asynchronous refined calibration, reconstruction
Event extraction
Quality control

Figure 1.1: Functional flow of the O^2 computing system.

- Low level data formats:
  - from FEE to CRU.
  - raw data + headers.
  - trigger information.
- “On-line”, in-memory data formats:
  - between FLPs and EPNs.
  - event+trigger information, time frames, control, monitoring…
- End user - persistent - formats:
  - (C)TF
  - AOD
  - QA

End user formats

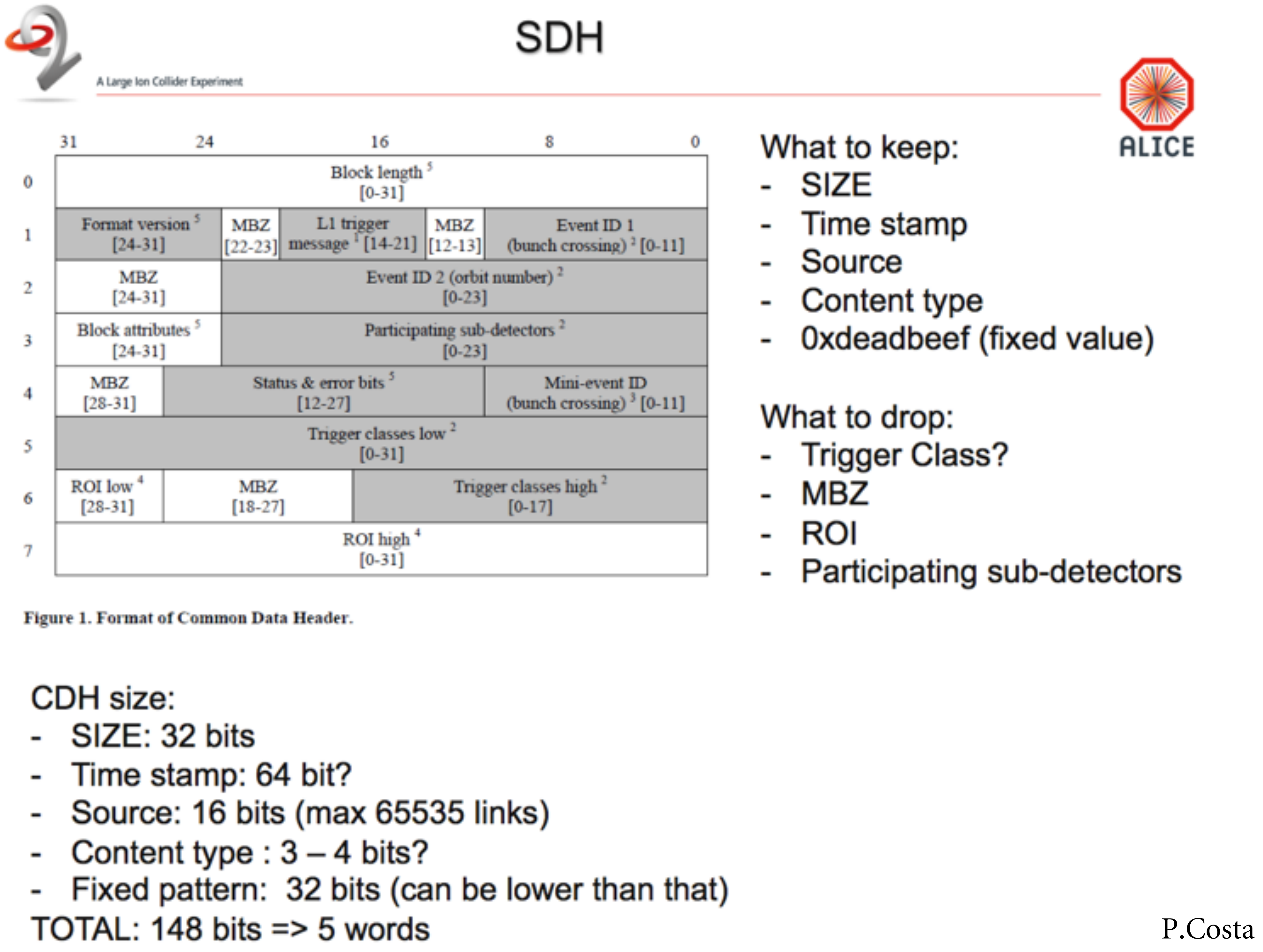
- AOD:
  - what to store there?
    - single events - as before, convenient access to single interactions
      - problems may arise with secondaries
    - all tracks from timeframe + reconstructed vertices + “default” association of tracks to interactions.
      - possible to emulate single event interface.
      - reassociation of tracks to interactions possible
      - re-reconstruction of V0’s etc.
  - discussion on the binary format needs to be continued.
  - Storage formats:
    - schema evolution/versioning?
    - etc…

End user formats

- Time frames:
    - test time frame:
        - input data for the development/testing of first reco/calibration prototypes.
        - maybe few scenarios: zero, small and full distortions.
        - at least ITS,TPC,TRD needed to exercise the distortion calibration.
    - when do we decide?
        - test time frame(s)
        - eventual CTFs.

    - Storage formats:
        - schema evolution/ versioning?
        - etc...

Low level formats



SDH

A Large Ion Collider Experiment

	31	24	16	8	0
0	Block length [5] [0-31]				
1	Format version [5] [24-31]	MBZ [22-23]	L1 trigger message [1] [14-21]	MBZ [12-13]	Event ID 1 (bunch crossing) [1] [0-11]
2	MBZ [24-31]	Event ID 2 (orbit number) [2] [0-23]			
3	Block attributes [5] [24-31]	Participating sub-detectors [2] [0-23]			
4	MBZ [28-31]	Status & error bits [5] [12-27]	Mini-event ID (bunch crossing) [3] [0-11]		
5	Trigger classes low [2] [0-31]				
6	ROI low [4] [28-31]	MBZ [18-27]	Trigger classes high [2] [0-17]		
7	ROI high [4] [0-31]				

Figure 1. Format of Common Data Header.

What to keep:
- SIZE
- Time stamp
- Source
- Content type
- 0xdeadbeef (fixed value)

What to drop:
- Trigger Class?
- MBZ
- ROI
- Participating sub-detectors

CDH size:
- SIZE: 32 bits
- Time stamp: 64 bit?
- Source: 16 bits (max 65535 links)
- Content type : 3 – 4 bits?
- Fixed pattern: 32 bits (can be lower than that)

TOTAL: 148 bits => 5 words

P.Costa

- First discussions on the new raw data header format

On-line formats and protocols

- ZeroMQ a candiate for the data transport.
  - handles (re-)connections, various routing topologies.
  - supports several protocols: tcp, ipc, inproc, pgm.
  - just a transport - no restrictions on the underlying payload format.

  - supports multi-part messages.
  - guarantees the delivery of all parts (or none) - in order.

Time frame format

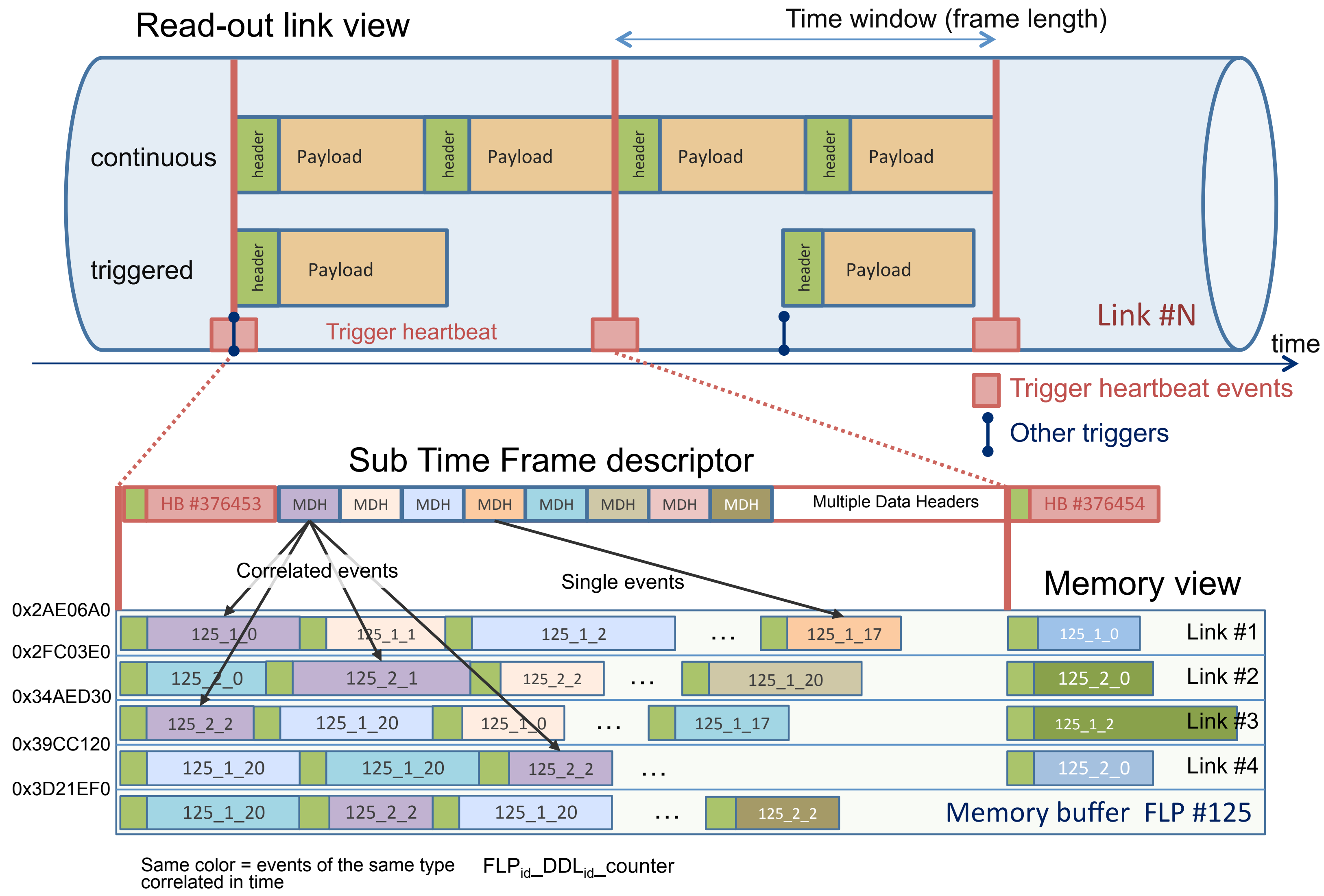


Figure 7.9: The Time Frame descriptor assembled on FLPs. Top: schematic flow of data blocks (events) as sent by the FEE. Bottom: The aggregation of individual data blocks into MDH headers.

- The time frame on the FLP, first iteration (TDR).
- Using ZMQ multi-part messaging foreseen from the start…

Time frame format

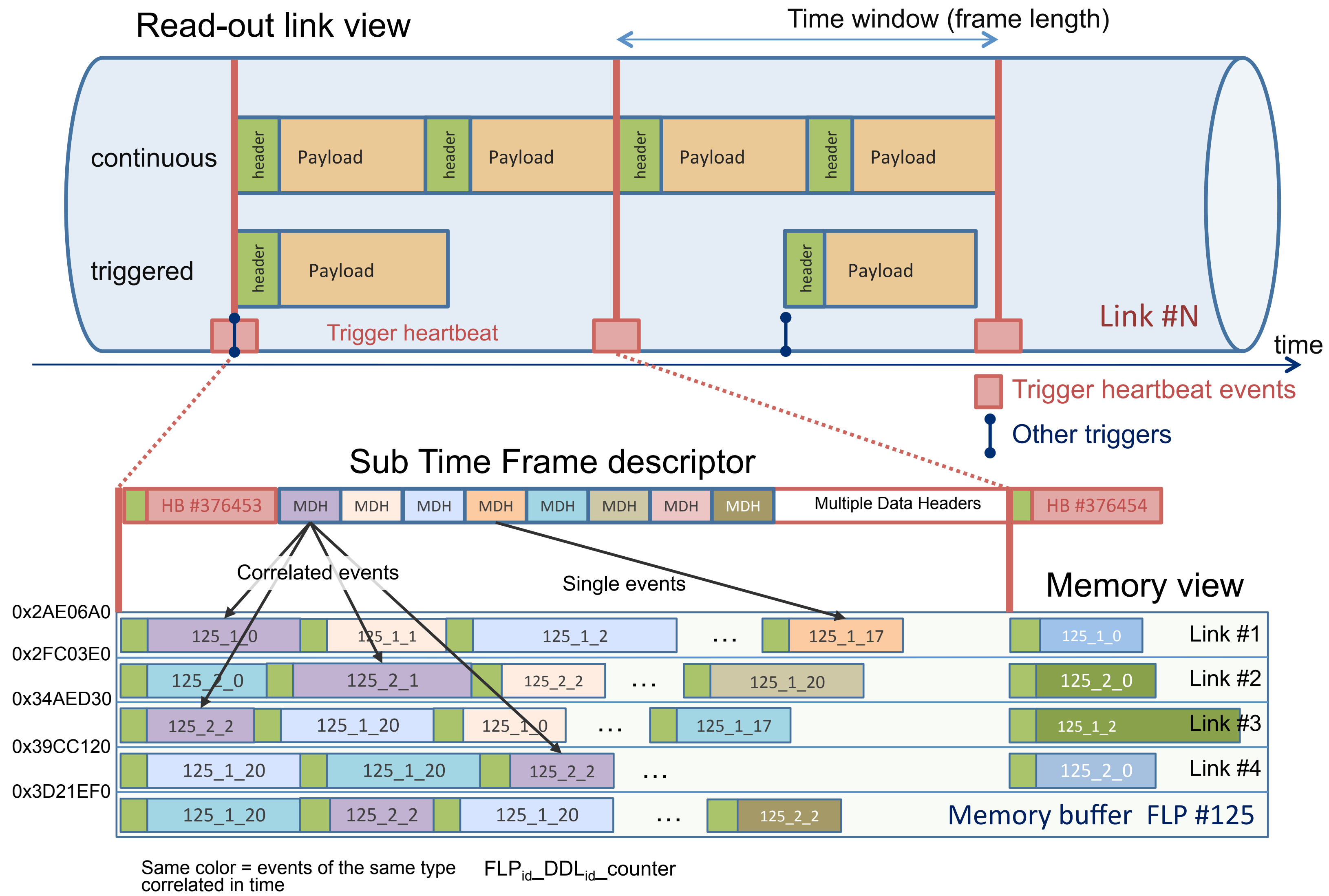


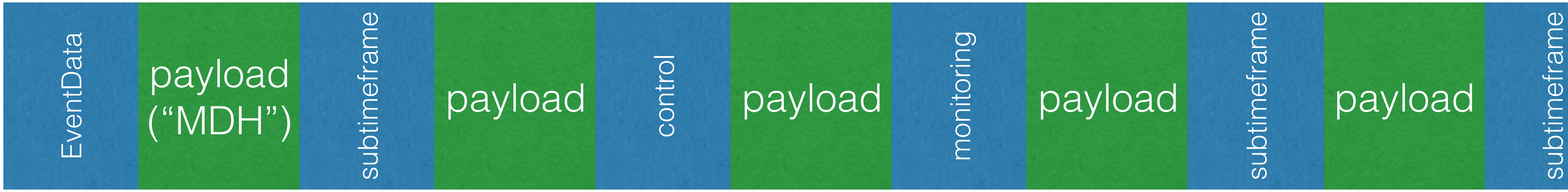
Figure 7.9: The Time Frame descriptor assembled on FLPs. Top: schematic flow of data blocks (events) as sent by the FEE. Bottom: The aggregation of individual data blocks into MDH headers.

- Intricate protocol with MDH, MDT
- deals with lots of low level information and explicit memory layout.

- The time frame on the FLP, first iteration (TDR).
- Using ZMQ multi-part messaging foreseen from the start…

Time frame format - first try

- … we can take it a bit farther:
- Leverage the multi part protocol also to the headers:
  - the message (a full (sub-)time frame) then is a collection of (header-payload) pairs.
  - no need for MDH (nor the trailers) - the global event information is just another data block.
  - it becomes trivial to add/remove/re-tag data (zero-copy).



EventData

payload ("MDH")

subtimeframe

payload

control

payload

monitoring

payload

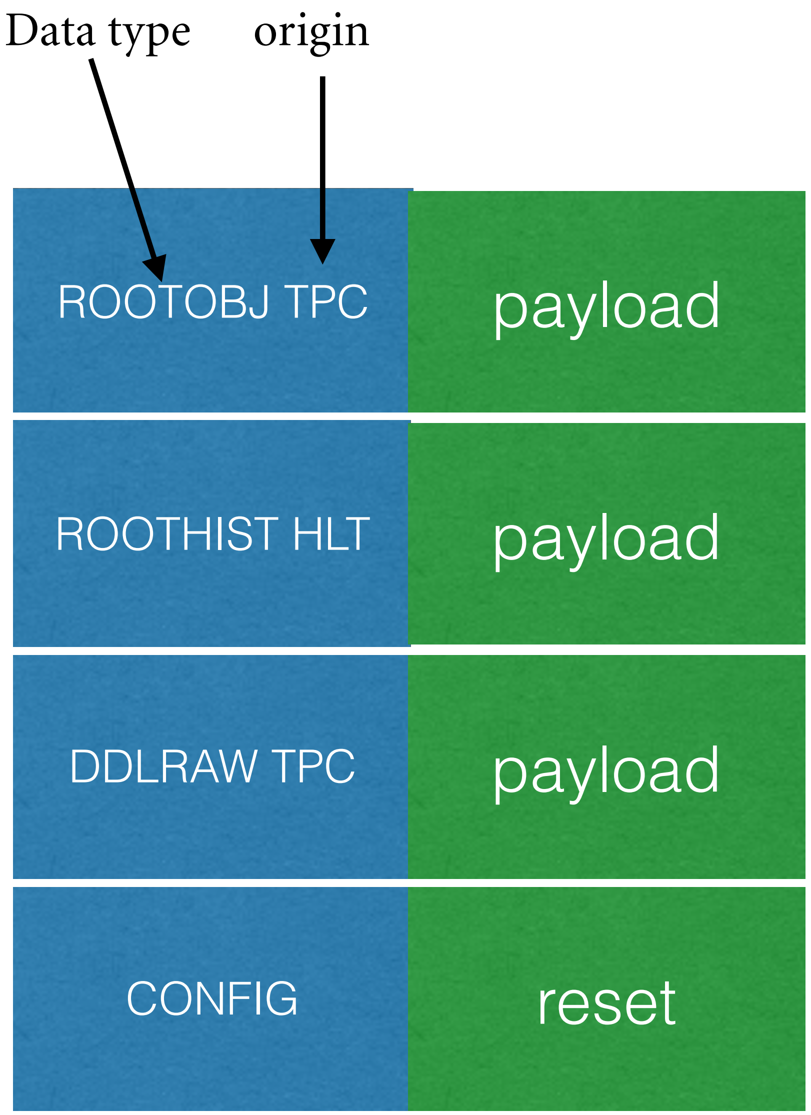
subtimeframe

payload

subtimeframe

On-line protocols - first try

- example header format:
  - data type+origin (char array)
  - equipment id (uint)



- monitoring/QA
- control/config/private data

- User decides what to do based on header info:
  - e.g. ROOTOBJ -> deserialize using ROOT

On-line protocols - first try

- Headers are easy to parse.
  - no need to look into the payload.
- Easy to add additional information to the header.
  - no restrictions on header size/format - to be agreed.
- Makes it trivial to embed control/monitoring in the data stream.
  - no need for additional sockets etc.
- Proposal: use it for all communications via a framework abstraction.

- This scheme is implemented in the HLT for run 2.
  - Based on the HLT experience from Run 1.
  - In aliroot:
    - ZMQ* standalone binaries - merger, proxy, viewer, etc.
    - AliZMQhelpers.h - a minimal C-like interface.
  - Can be used to interface to alfa development during run2?

On-line data formats

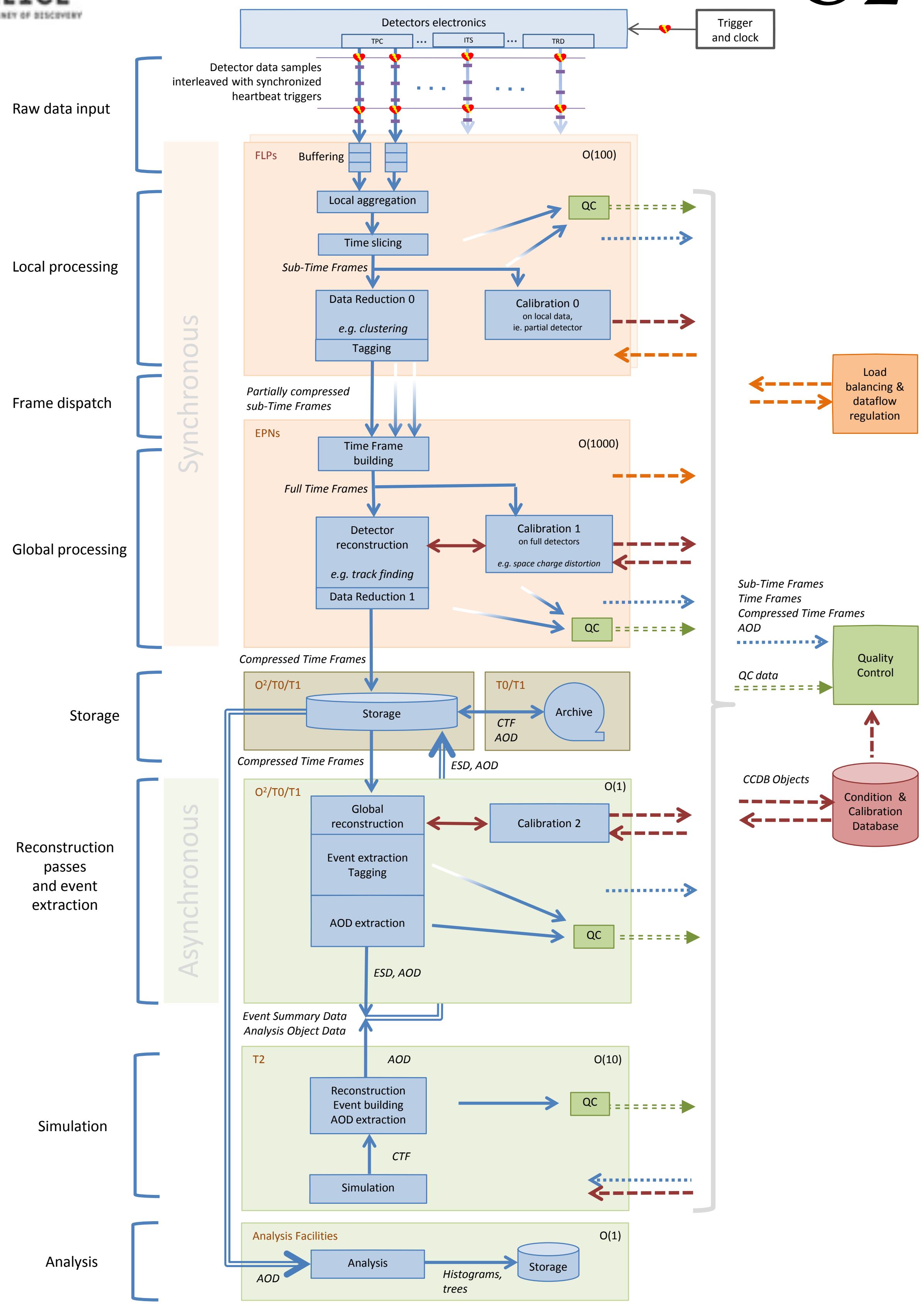
- Proposed data scheme does not impose any payload format
  - Formats can be mixed. e.g:
    - ROOT objects for monitoring.
    - Flat structures for timeframe data.
  - data format for the time frames/ compressed time frames needs to be discussed.
    - strong preference towards a zero-copy approach
      - data stays in transport buffers, we just access it.
      - Flat structures a la HLT AliFlatESD ?
      - completely flat buffers? (structs of arrays?)

Summary

- Input wanted.
- the meeting schedule will be circulated on the CWG4 mailing list.
- some first steps made, need to make strides...

backup

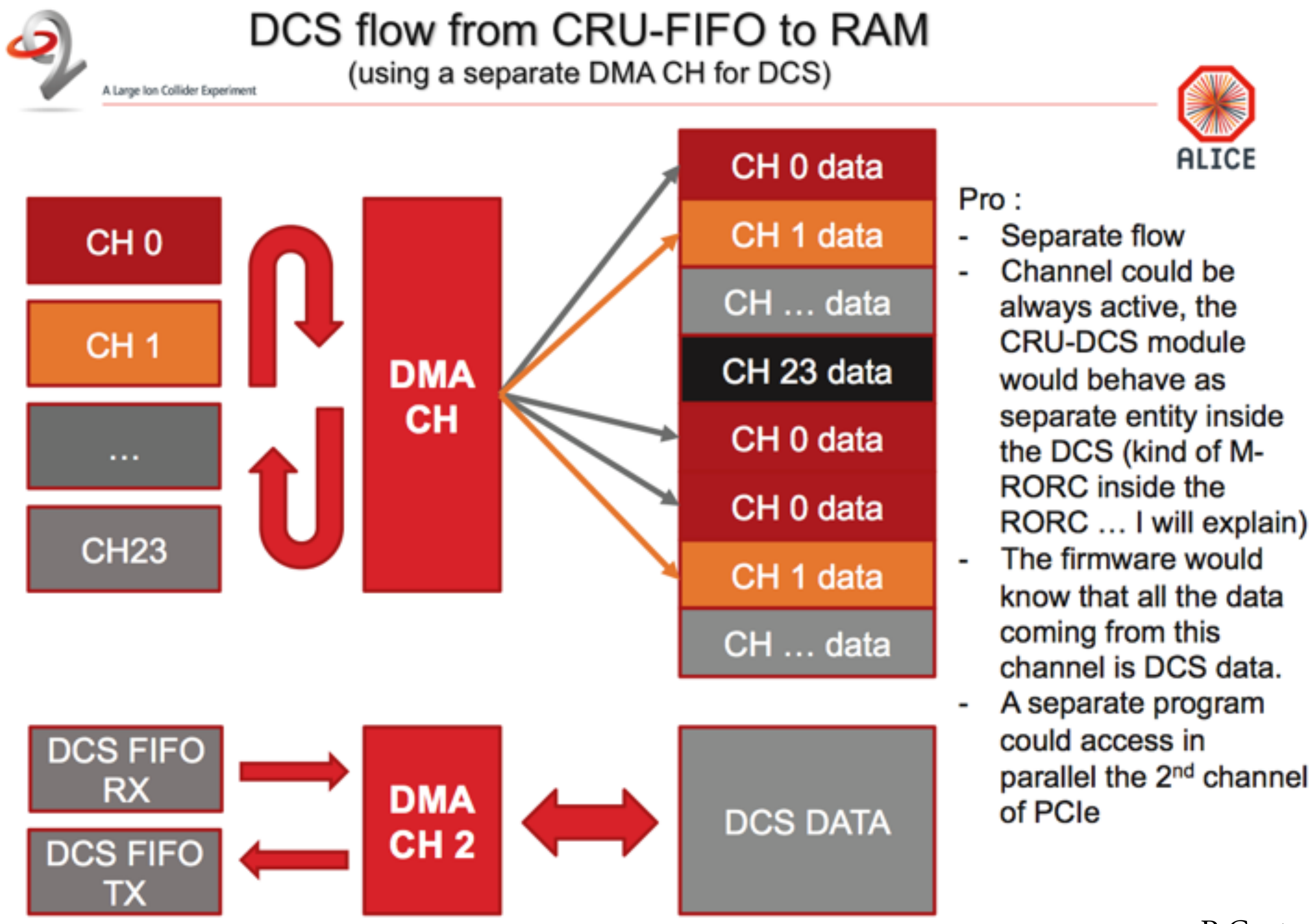
O2 data flow



Detectors electronics
TPC
...
ITS
...
TRD
Trigger and clock
Detector data samples interleaved with synchronized heartbeat triggers
Raw data input
Local processing
Frame dispatch
Global processing
Storage
Reconstruction passes and event extraction
Simulation
Analysis
Synchronous
Asynchronous
FLPs
Buffering
O(100)
Local aggregation
QC
Time slicing
Sub-Time Frames
Data Reduction 0
e.g. clustering
Tagging
Calibration 0
on local data, ie. partial detector
Partially compressed sub-Time Frames
Load balancing & dataflow regulation
EPNs
O(1000)
Time Frame building
Full Time Frames
Detector reconstruction
e.g. track finding
Data Reduction 1
Calibration 1
on full detectors
e.g. space charge distortion
QC
Sub-Time Frames
Time Frames
Compressed Time Frames
AOD
QC data
Quality Control
Compressed Time Frames
O²/T0/T1
Storage
T0/T1
Archive
CTF
AOD
Compressed Time Frames
ESD, AOD
CCDB Objects
Condition & Calibration Database
O²/T0/T1
O(1)
Global reconstruction
Calibration 2
Event extraction
Tagging
AOD extraction
QC
ESD, AOD
Event Summary Data
Analysis Object Data
T2
AOD
O(10)
Reconstruction
Event building
AOD extraction
QC
CTF
Simulation
Analysis Facilities
O(1)
AOD
Analysis
Histograms, trees
Storage

Figure 5.1: Data flow and processing pipeline of the O^2 system.

Low level formats



DCS flow from CRU-FIFO to RAM

(using a separate DMA CH for DCS)

A Large Ion Collider Experiment

CH 0

CH 1

…

CH23

DMA CH

CH 0 data

CH 1 data

CH … data

CH 23 data

CH 0 data

CH 0 data

CH 1 data

CH … data

DCS FIFO RX

DCS FIFO TX

DMA CH 2

DCS DATA

Pro :

- Separate flow
- Channel could be always active, the CRU-DCS module would behave as separate entity inside the DCS (kind of M-RORC inside the RORC … I will explain)
- The firmware would know that all the data coming from this channel is DCS data.
- A separate program could access in parallel the 2nd channel of PCIe

P.Costa

- First discussions on low level data organization.