Implementing the TFC commands in the Front-End

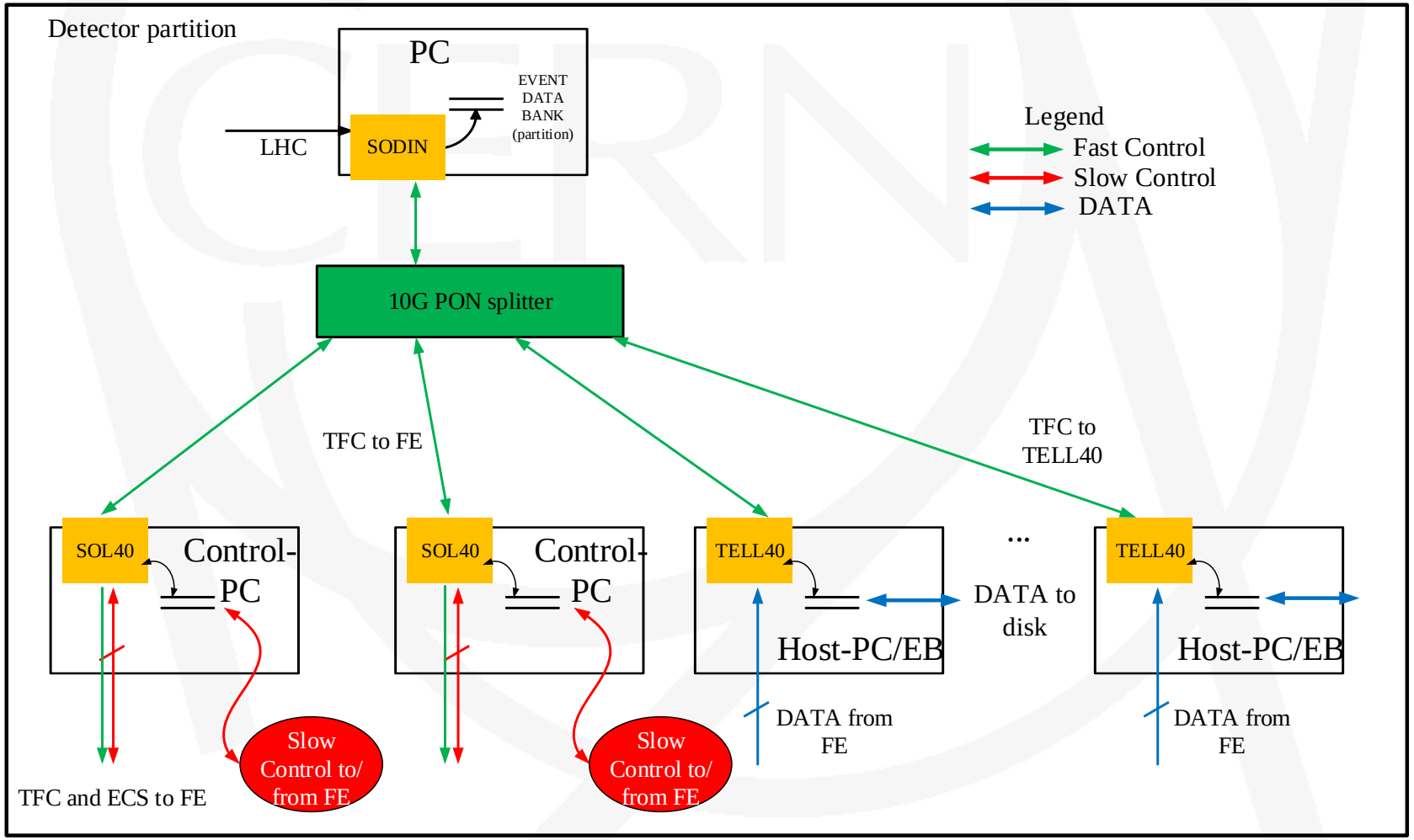
LHCb Upgrade Electronics Meeting

09 June 2016

F. Alessio, CERN

TFC system in the upgrade (as of v1.5)

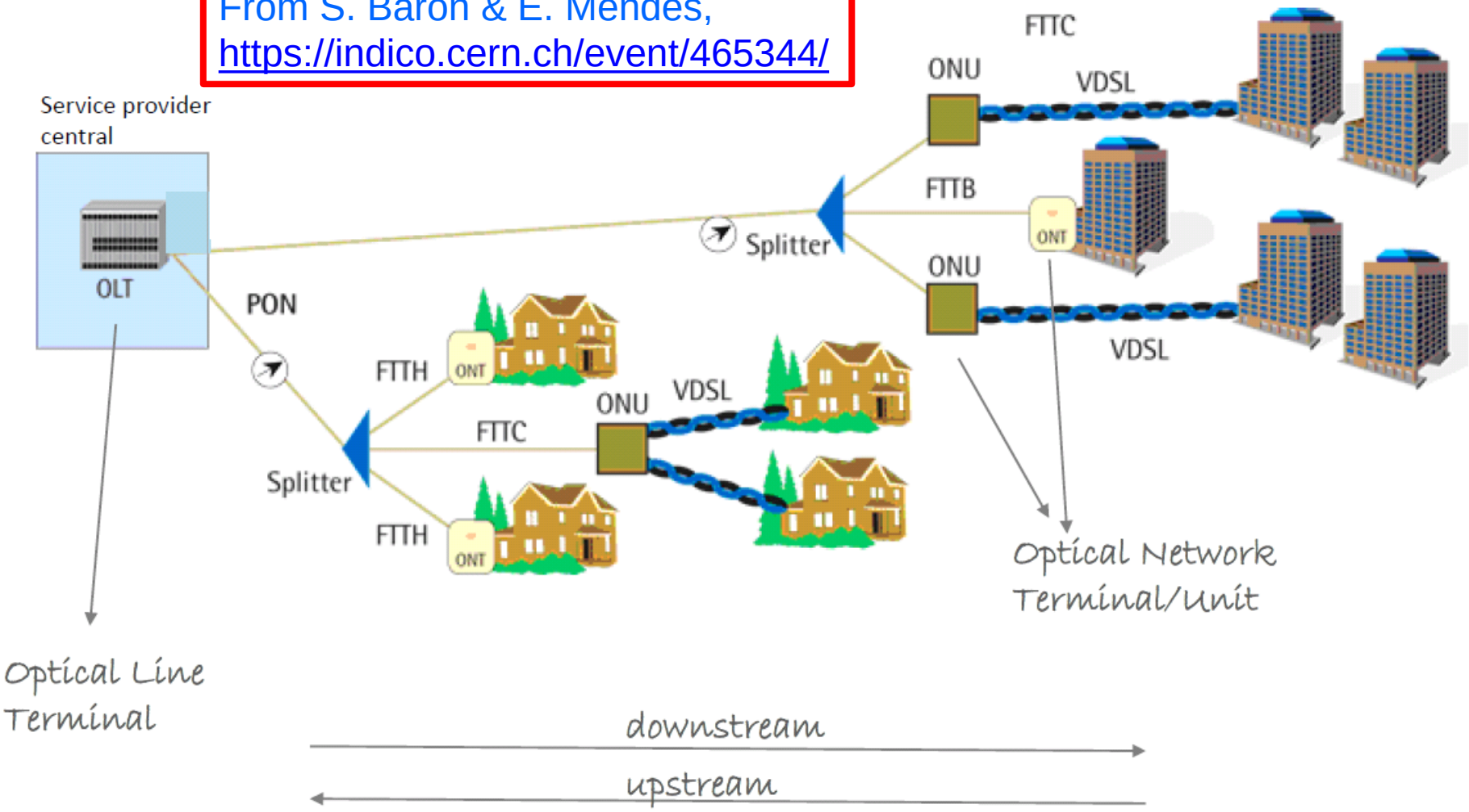
LHCb-PUB-2012-001 (v1.5)



The solution: FTTx → Fiber To The x

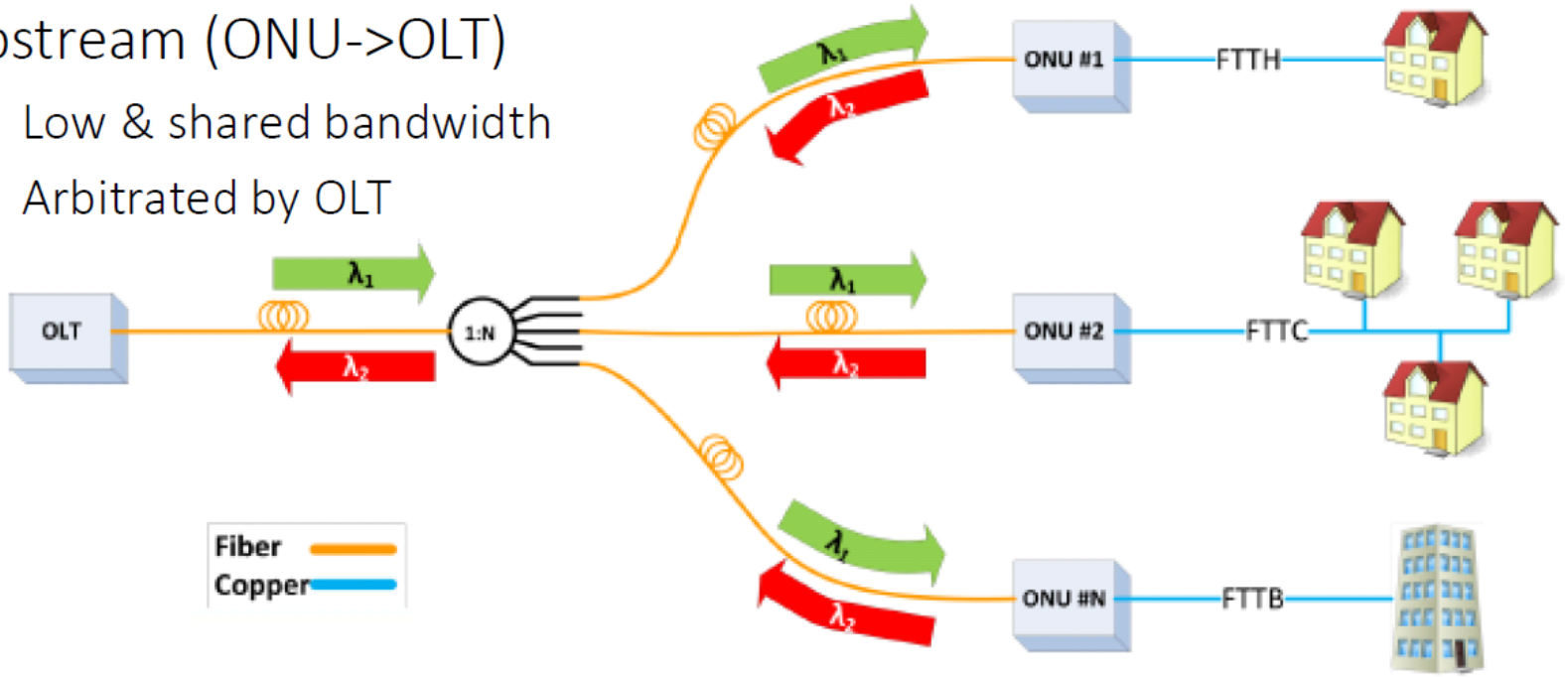
...and PON → Passive Optical Network

From S. Baron & E. Mendes, https://indico.cern.ch/event/465344/



PON Principle

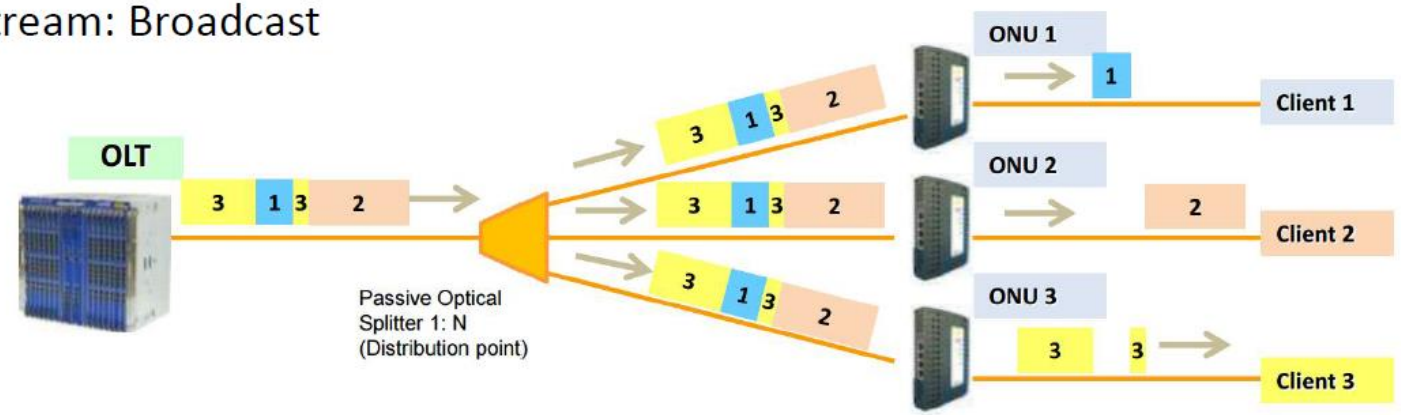
- Point to Multipoint Network (P2M)
  - Bidirectional
  - Wavelength Division Multiplexing: 1 fiber, 2 wavelengths (1 Up, 1Down)
- Downstream (OLT->ONU)
  - High bandwidth
- Upstream (ONU->OLT)
  - Low & shared bandwidth
  - Arbitrated by OLT



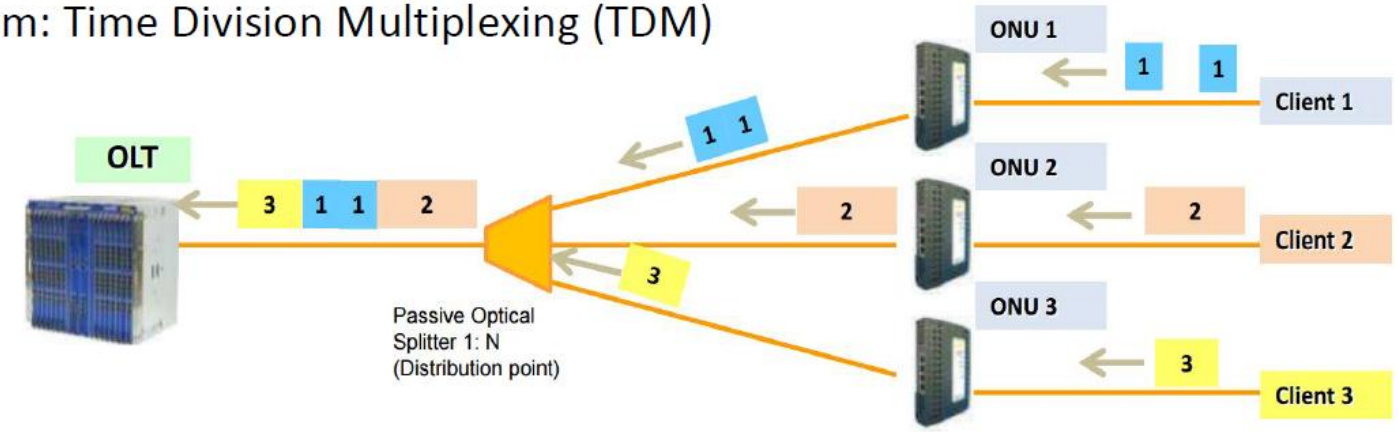
PON Principle

TTC PON

Downstream: Broadcast



Upstream: Time Division Multiplexing (TDM)

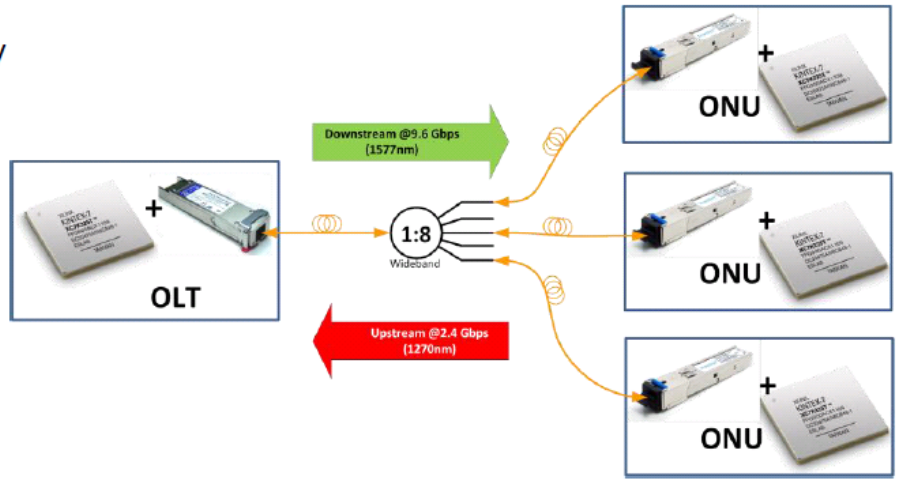


Main figures of merit of TTC-PON

- System
  - *High* Split ratio (*Large* Power Budget)
  - *Low* Level of customization wrt standard
- Downstream (broadcast)
  - *Low* & *deterministic* Trigger latency
  - *High* Bandwidth
  - *Excellent* Clock quality

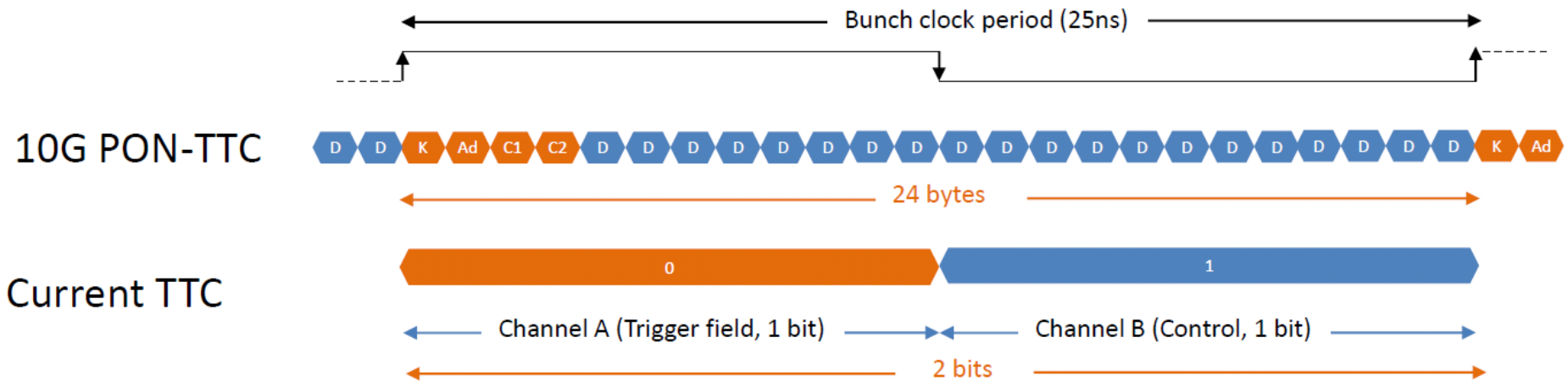
  => Classic Optical Link
- Upstream (TDM)
  - *Low* Busy latency
  - *High* Dynamic range
  - *High* Payload

  => Challenging



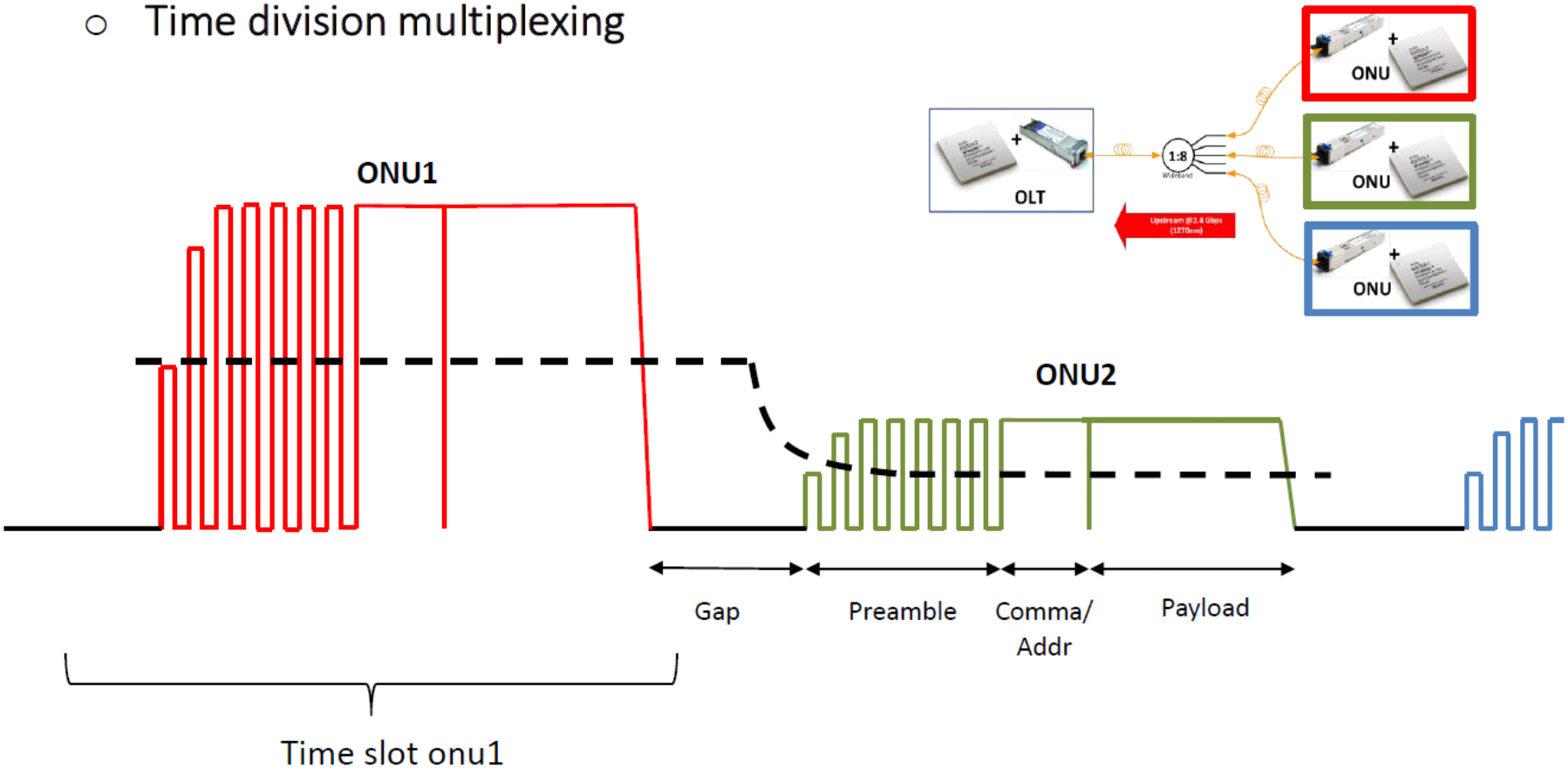
Downstream Protocol

- OLT → ONUs (broadcast)
- LHC Bunch Clock (BC) synchronous
- 9.6Gbps serial link
- 8B10B encoded, K28(.1, .5) comma
- Full Payload: 24 bytes (192 bits) per BC
- Slow Control field: 3 bytes
- **User Data Field: 20 bytes (160 bits)**



Upstream data transmission scheme

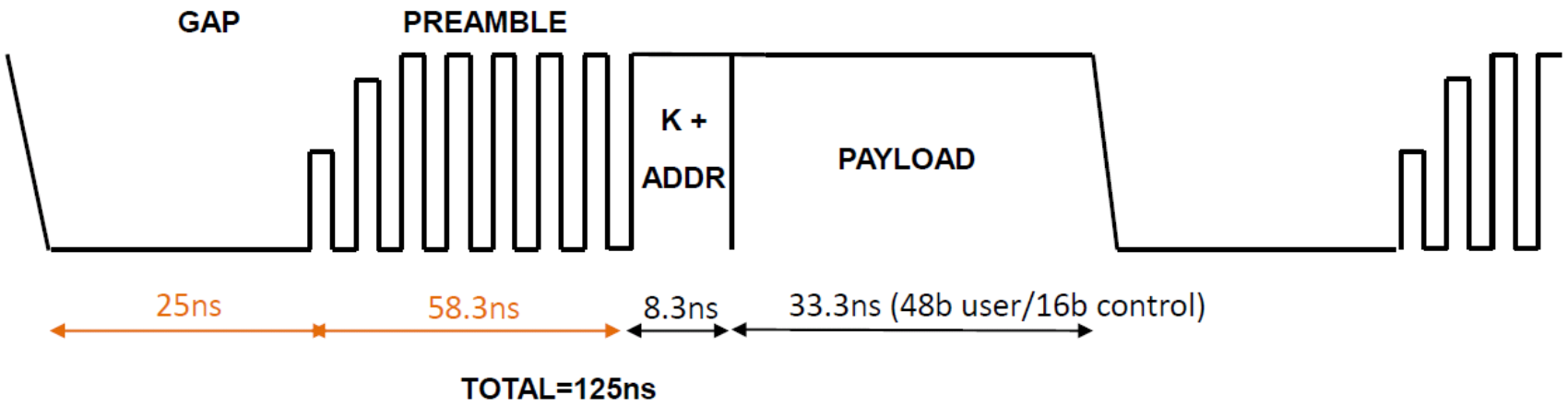
- Time division multiplexing

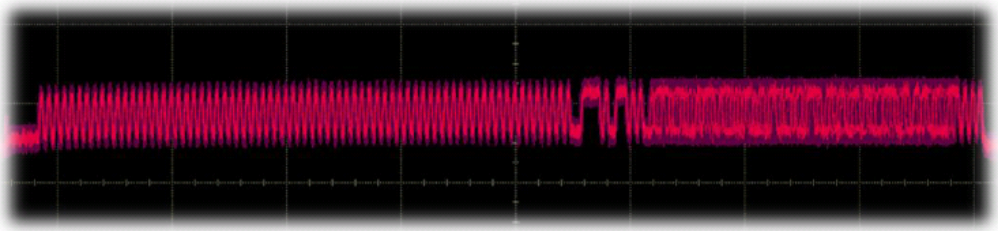


Waiting time= (Number ONU's) x (Time slot ONU)

Upstream data transmission scheme

- Burst composition:



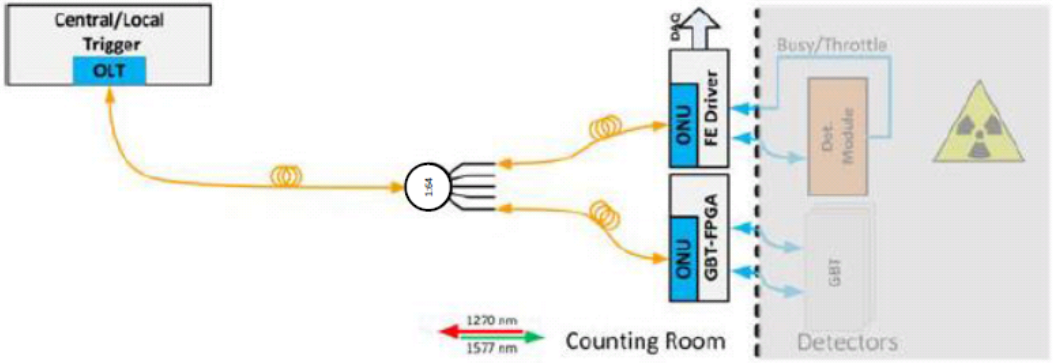


- XGPON specs:
  - Minimum Gap: 25ns
  - Minimum Reset Pulse Width: 25ns
  - Maximum Settling time: 52ns (=min. preamble length)

System Performance

System
Bidirectional
Fully scalable and flexible
Split ratio: 1:64 maximum
*Immune to temperature variations

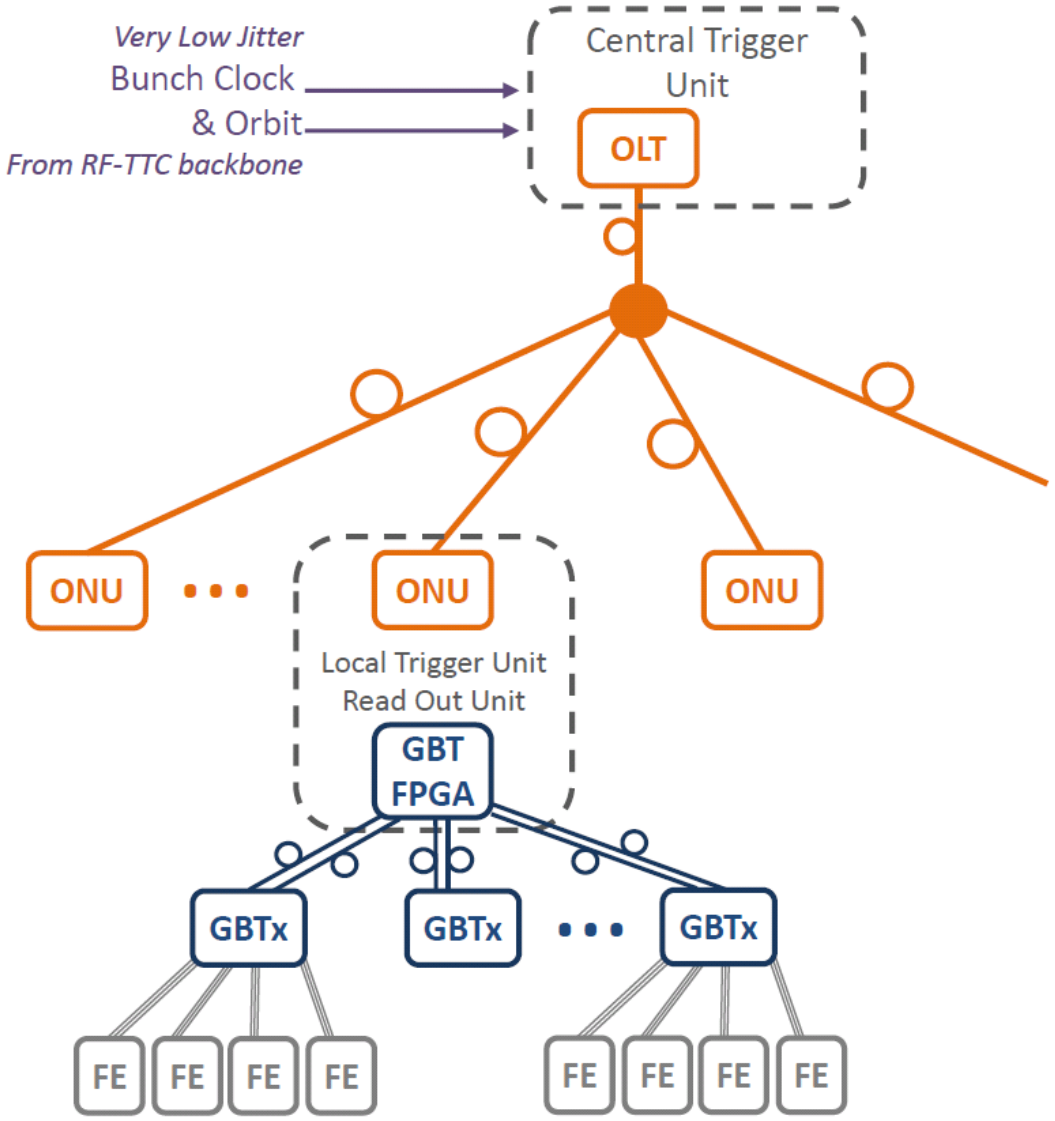
Downstream	
Bunch clock synchronous	
Data-Rate	9.6Gbps (6.4Gbps user)
Payload user	160b (per BC)
Latency	~86ns



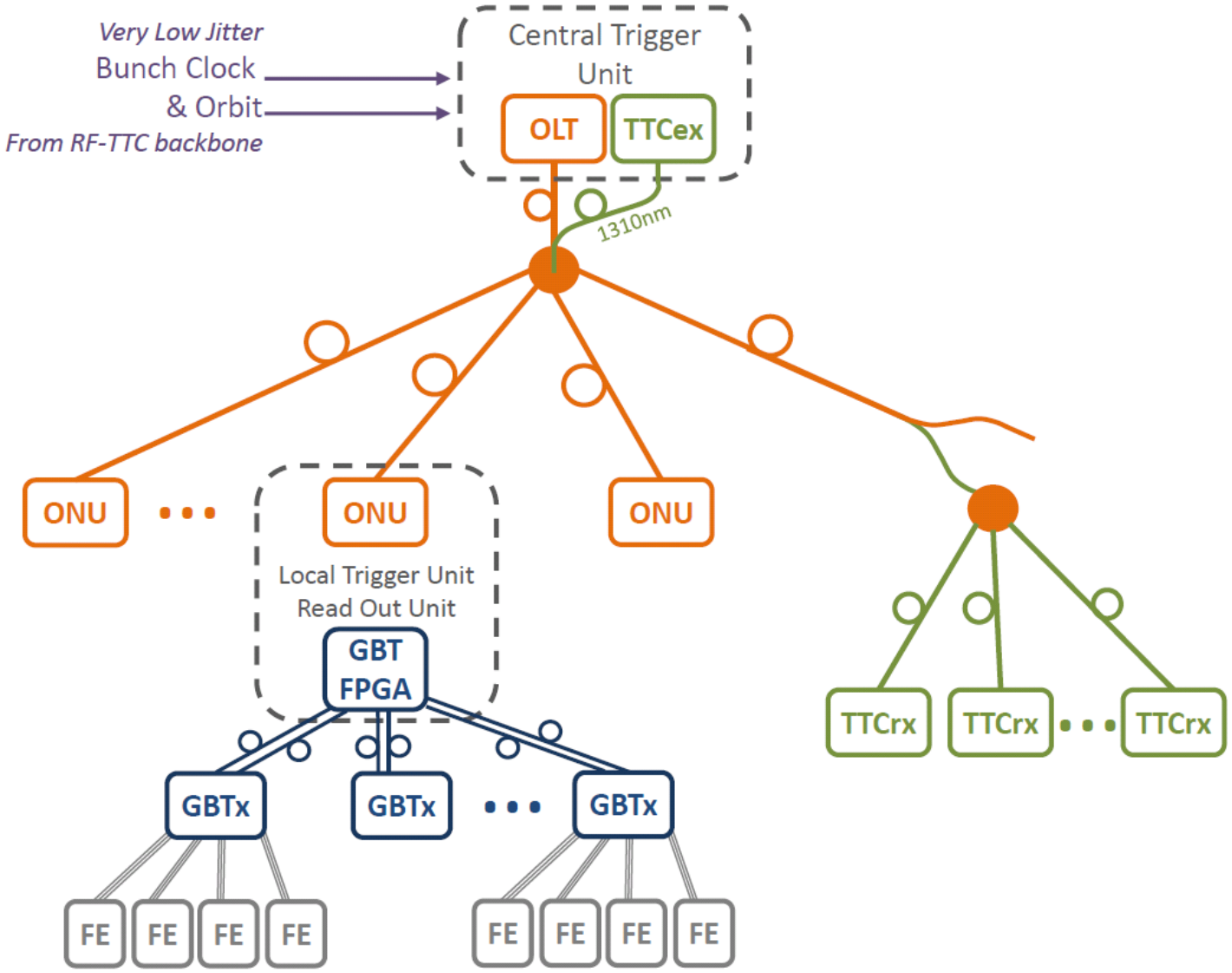
Upstream	
Data-Rate	2.4 Gbps - linerate 8Mbps (64 ONU's)
Payload user	48b (each N_ONUx125ns)
Waiting time per ONU	N_ONUx125ns 8us – 64 ONUs 4us – 32 ONUs
Dynamic range	12 dB (!)

Integration with GBT & legacy TTC

TTC PON



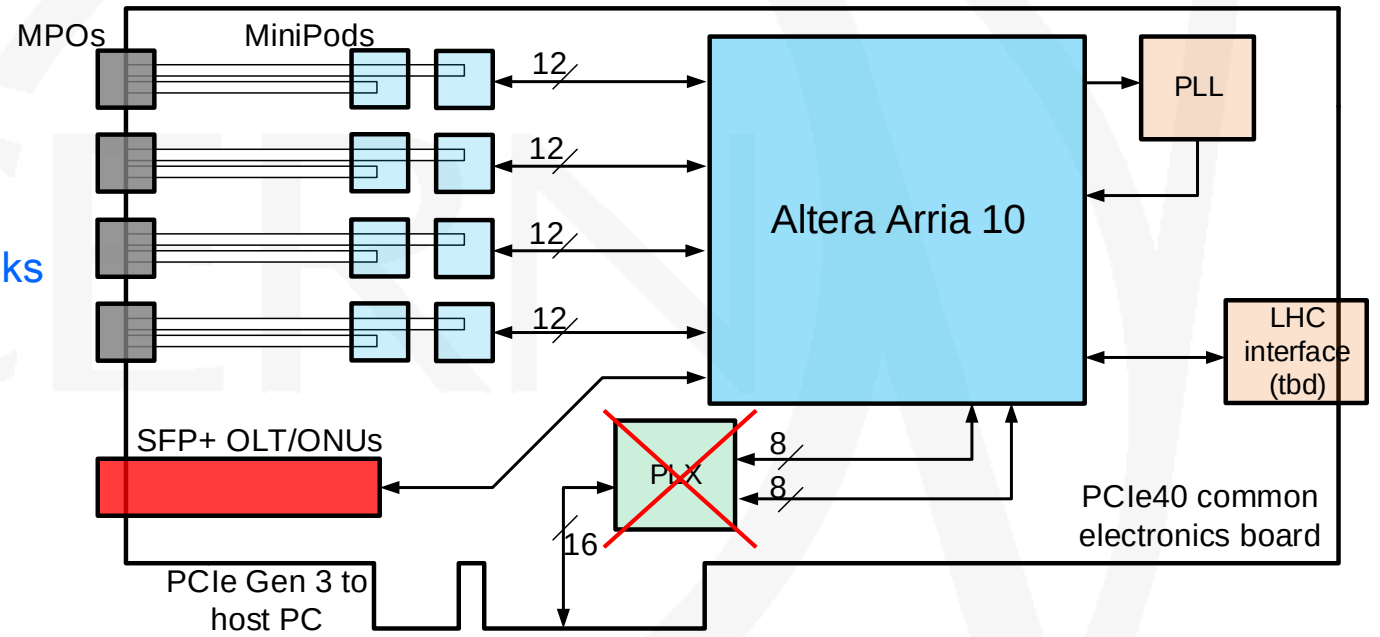
Integration with GBT & legacy TTC



TFC on common hardware backbone

In principle:

- Firepower of 48 bidir links
- TTC-PON interface OLT/ONUs with SFP+
- Deterministic on board external PLL for resynchronization



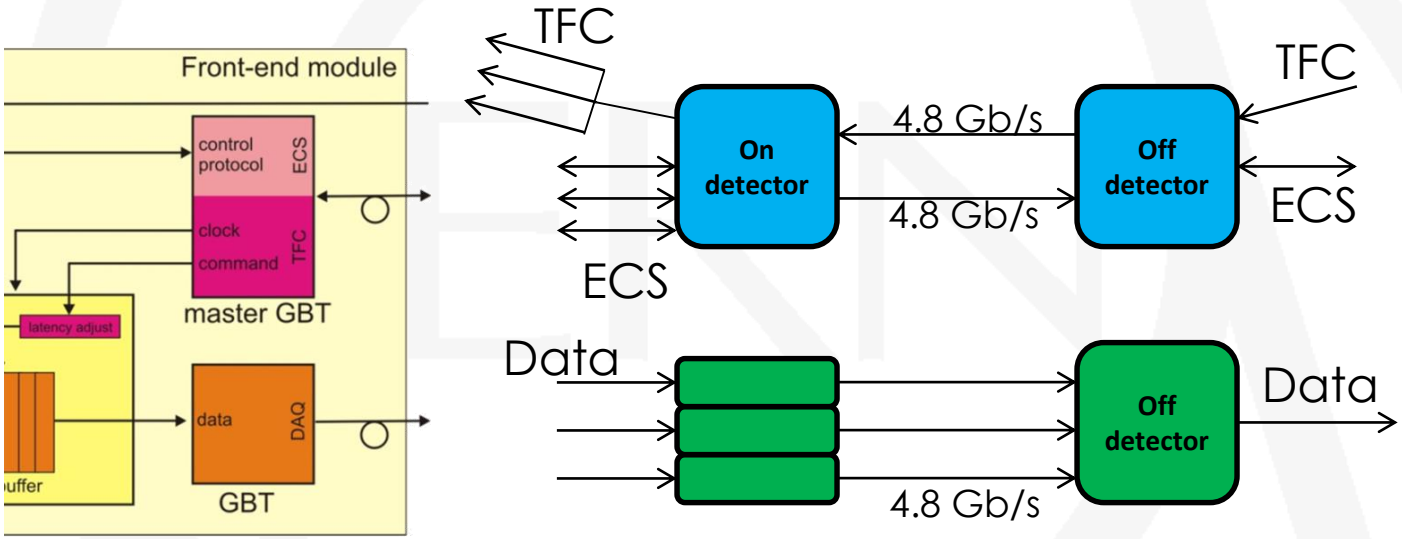
Same exact hardware (PCIe40), firmware defines the functionality: SODIN, SOL40, TELL40

- The Host PC is the PC controlling the board
- In principle ~100 Gbps of ECS access

The only difficulty in the system is the PON splitter, currently limited to 64 destinations

- May need cascading or may need a different SODIN hardware (with many SFP+ OLT/ONUs instead of MiniPods) → to be discussed later on
- As usual, tight and proficuous collaboration with CPPM and EP-ESE

Fast & Slow Control to FE



Separate links between controls and data

- A lot of data to collect
- Controls can be fanned-out (especially fast control)

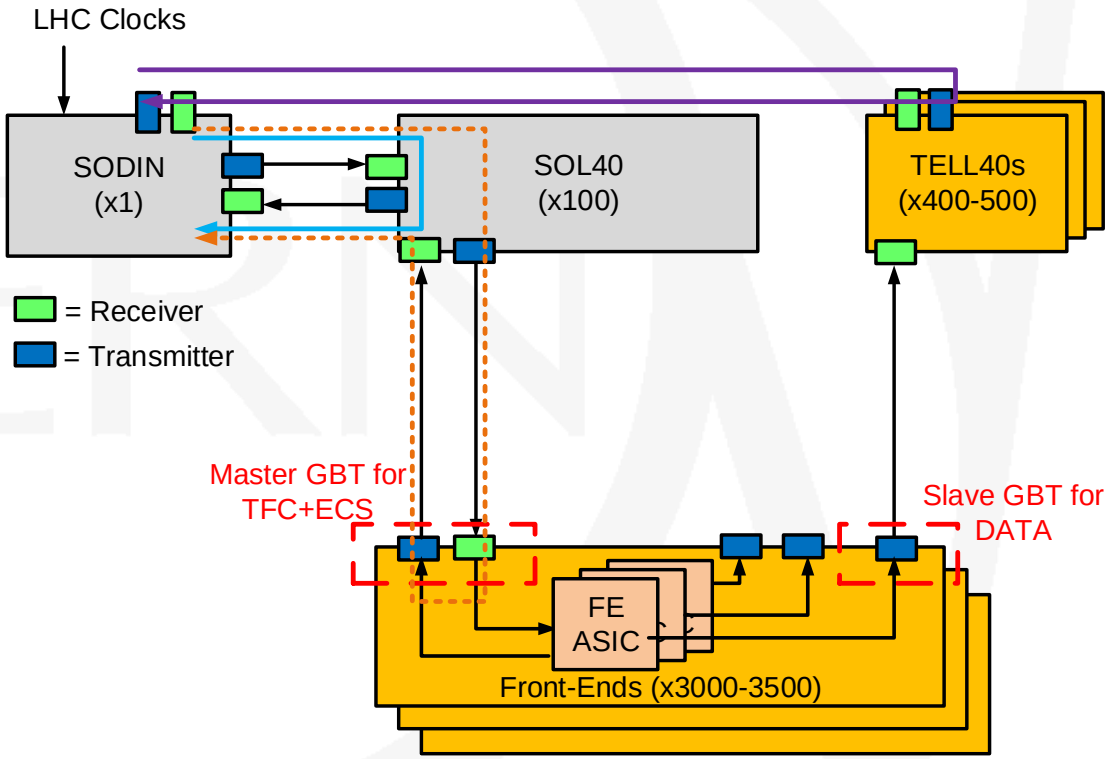
Compact links merging Timing, Fast and Clock (TFC) and Slow Control (ECS).

- Extensive use of GBT as Master GBT to drive Data GBT (especially for clock)
- Extensive use of GBT-SCA for FE configuration and monitoring

Timing and command distribution

Master GBT @ FE controls the FE ASIC + Slave GBTs

- Clock from Master GBT
- TFC commands on e-links
- ECS configuration and monitoring through SCA
- Slave GBTs controlled through SCA from Master GBT



Fixed latency and deterministic phase recovery of TFC commands is ensured by combination of TTC-PON and GBT features

→ Customization is needed to synchronize links and make sure that TELL40s decodes data properly

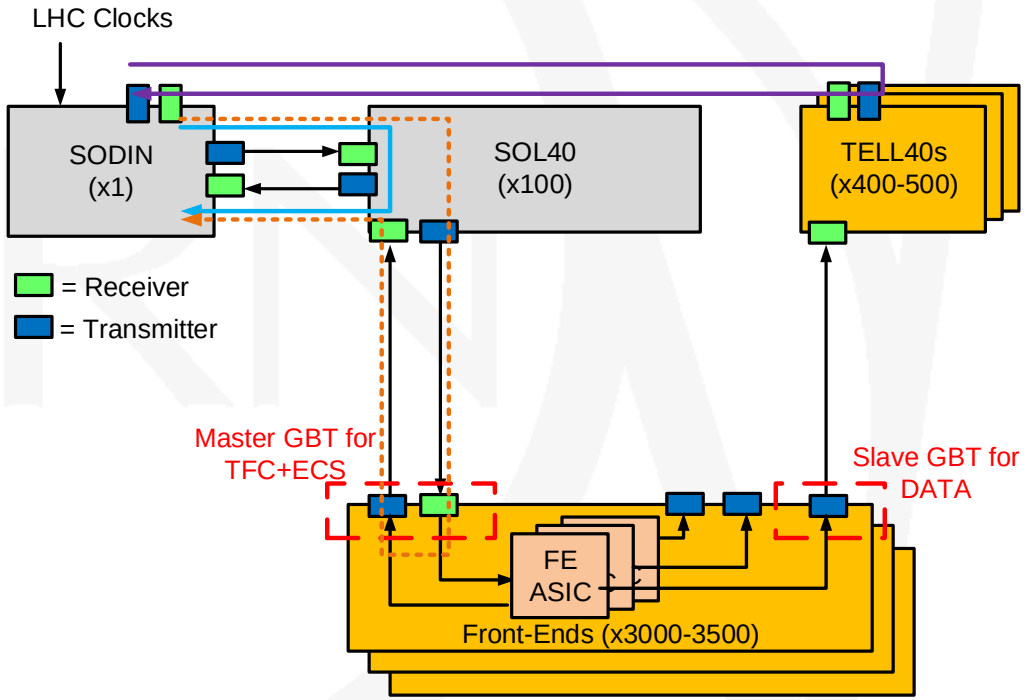
→ Customization is needed to properly decode TFC commands

→ ECS to FE through TFC links via SOL40

Do not forget the synchronicity checks

In the specs and at the reviews we asked you to implement few features to check synchronicity and latency/cable delays/sources

→ See the paths in the picture



There should be means to loopback input data to each link:

1. Loop back TFC downlink data onto the TFC uplink
   - ✓ Desirable if this goes through the FE ASIC/FPGA
   - ✓ If not, GBT has loopback capabilities
2. Loop back TFC downlink data onto the DATA uplink
   - ✓ Check for all cable delays
3. Same mechanisms to eb put in place between SODIN-SOL40 and SODIN-TELL40

General philosophy for flow control

Flow control is centralized and synchronous

- One set of TFC commands per BXID, at 40 MHz
- TFC will make sure that the system is synchonized or that the usage of command is consistent. Programmability of the firmware allows to create procedures.
  - ✓ Resiliant to create «ad-hoc» procedures
  - ✓ Generalization as a mean to cover for all scenarios (increase complexity, but covers more cases)

Data readout is distributed and asynchronous

- No need to distribute a trigger so data links can be either out of synch or offset in time
  - ✓ TELL40 has logic to align data from all input links
- Data across links should be consistent
  - ✓ If it's a calibration command at a BXID, it should be calibration data for all fragments from the whole detector on the same BXID.
- TELL40 data decoding at input should be independent from TFC
  - ✓ FE runs anyway at 40 MHz and too much memory would be needed to pipe TFC
  - ✓ Only applies a trigger and open/close MEPs (other information is for redundancy)

→ Need at least a minimum set of common solutions to ensure synchronicity, consistency and that bandwidth matches

TFC commands to TELL40, list

Protocol on information to tell the TELL40 what each event is:

63 .. 52	51	50	49 .. 18	17 .. 14	13 .. 10
BXID(11..0)	Reserve	MEP Accept	MEP Dest(31..0)	Trigger Type(3..0)	Calibration Type(3..0)

9	8	7	6	5	4	3	2	1	0
Synch	Snapshot	Trigger	BX Veto	NZS Mode	Header Only	BE Reset	FE Reset	EID Reset	BXID Reset

Trigger command to accept events for that BXID:

→ Trigger acting as a VETO to rate regulate the system

- Not based on physics decision (NO LLT)
- Unbiased rate regulation of the system by reducing the rate at which trigger is set

MEP accept command when MEP ready:

- Take MEP address and send all fragments to that address
- Dynamic mechanisms based on well-oiled mechanism used today
- Investigations ongoing to see if a different mechanism is more suited for upgraded system

Rate regulation can also help to reduce output bandwith

→ Rate regulation in SODIN was also considered as the safest mechanism to reduce output bandwidth should your FE send too much data wrt to the TELL40 output bandwidth

→ Out of the full 40 MHz, only ~28 MHz contains beam-beam, ~4 MHz contains beam-empty/empty-beam, ~8 MHz contains emtpy-empty

- ✓ SODIN can have a programmable mask that defines the rate of crossing types to be kept
- ✓ 100% for bb, 25% for eb/be and 10% for ee?
- ✓ Effectively the full input rate is 30 MHz.

→ This was described in EDMS 1606939 as Renaud mentioned a while ago.

TFC commands to FE, list

Protocol on information to FE for flow control:

23 .. 12	11 .. 10	9	8 .. 5	4	3	2	1	0
BXID(11..0)	Reserve	Snapshot	Calibration Type(3..0)	BX Veto	NZS Mode	Header Only	FE Reset	BXID Reset

Each command should have a local configurable delay

- → GBT does not support individual delays
- → Need for «local» pipelining: detector delays+cables+operational logic (i.e. laser pulse?)

To allow use of commands/resets for particular BXID, TFC word will arrive before the actual event takes place

- → Accounting of delays in SODIN: for now, 16 clock cycles earlier + time to receive
- → Aligned to the furthest FE

Each sub-detector can choose whichever set of TFC commands they need and in which specific position (e-link) → configurable mapping done in SOL40

TFC commands explained

"BXID" and "BXID Reset"

→ Every TFC word carries a BXID for synchronicity checks of the system

→ A BXID Reset is sent at every turn of the LHC (orbit pulse)

- ✓ Only reset the internal bunch counter of the FE
- ✓ This must be in your FE chip

→ BXID can be ignored by sub-detectors if this is compensated by a mean to check the synchronicity of the system

"FE RESET"

→ Bit set for one clock cycle in TFC word sent to FE and TELL40

→ Reset of FE operational logic for data processing, formatting, transmission…

- ✓ Should not touch the internal bunch counter
- ✓ FE electronics should be back as soon as possible: SODIN will ensure no data is being accepted during the FE reset process (by setting Header Only and veto trigger).
- ✓ Reset TELL40 data input logic: the same bit is sent to TELL40 for same BXID.
- ✓ Followed by a SYNCH command.

"BE RESET"

→ Bit set for one clock cycle in TFC word, only to TELL40

→ Reset of TELL40 operational logic for data processing, formatting, transmission…

- ✓ TELL40 should be back as soon as possible: SODIN will ensure no data is being accepted during the BE reset process (by setting Header Only and veto trigger).
- ✓ May be or may not be followed by a SYNCH command. In principle is not needed if TELL40 can still monitor the BXID at the input.

TFC commands explained

"HEADER ONLY"

→ Idling the system: only header (or few bits) in data word if this bit is set

- ✓ Multiple purposes: set it during reset sequence, during NZS transmission, during TAE mode…
- ✓ Header Only also can be set in case of back-pressure. If rate regulation is applied it could be chosen to sent also Header Only to FE

"BX VETO"

→ Based excusively on filling scheme

- ✓ Only header (or few bits) in data word if this bit is set
- ✓ Allows "recuperating" buffer space in a LHC-synchronous way
- ✓ Load filling scheme in SODIN, then apply recipe (which BX Type to keep, can also define a specific rate for it)

✓ BX Veto and Header Only commands are identical from FE point of view → *ORed*

"EID RESET"

→ Reset the Event Counter ID (64 bits), only sent to TELL40

- ✓ This is the only unambiguous identifier of an event and its fragments
- ✓ EID monotonically increases every time an event is accepted
- ✓ Reset only if Run number changes or if triggered via ECS

A word on the Event ID

The Event ID (64 bits) is the unique identifier of an event for the “time it is sitting in the system”

- → Events are recorded asynchronously, so event fragments (MEPs) can come out of the TELL40 at very different times → this is the only way an event is uniquely identified
- → Event ID shall increase monotonically for each run.
  - ✓ Only the Reset issued by SODIN resets it.
  - ✓ And the process is triggered by either ECS (asynchronously) or by a change run (STOP Trigger/START Trigger)
- → First Event ID is 0

The Event ID should be in the header of the MEP packet sent out to a destination

- → All other events in the MEP will have Event ID reconstructed by knowing its position in the MEP → can reduce overhead by packing a lot of fragments in a MEP (O(1000))
- → All other events in the MEP carry a BXID which wraps around every orbit
- → The TELL40 simply relays the BXID received from FE
  - ✓ If it’s wrong at the FE, it is wrong at the TELL40 and will be wrong in the DAQ!
  - ✓ It’s the role of the FE to maintain synchronicity, TELL40 can only monitor and possibly recuperate few clock cycles (16), but not more.
- → In current system its size is reduced in TELL1 packets to cover for the time it sits in the system
  - → A simple calculation shows that keeping it to 64 bits would be advisable (even @ 28 MHz, it needs at least 35 bits..)
- → Only SODIN transmits the full 64 bits Event ID together with all the information of that event (trigger - event - calibration type, UTC timestamp, orbit number, run number, etc).

TFC commands explained

"CALIBRATION TYPE"

→ Used to take data with special trigger pulses (periodic, calibration)
- ✓ Dedicated 4 bits: i.e. 4 different calibration commands possible
- ✓ Dynamic association to be used for calibration and monitoring
  - Absolute need of delays to account for each individual delay in the detectors
- ✓ SODIN overrides internal trigger decision at TELL40
  - Periodic or calibration higher priority, can also have a fast rate, programmable

"NZS MODE"

→ Read out (all) FE channels non-zero suppressed
- ✓ Packing of full set of bits in many consecutive GBT frames: needs buffering

→ Possible to have also multi-NZS readout: *consecutive NZS events*
- ✓ SODIN will take care of sending Header Only for a defined set of clock cycles later to allow recuperating buffer space (programmable as well, to the slowest of the detector)
- ✓ And reject events thereafter to avoid creating bottlenecks

"SNAPSHOT"

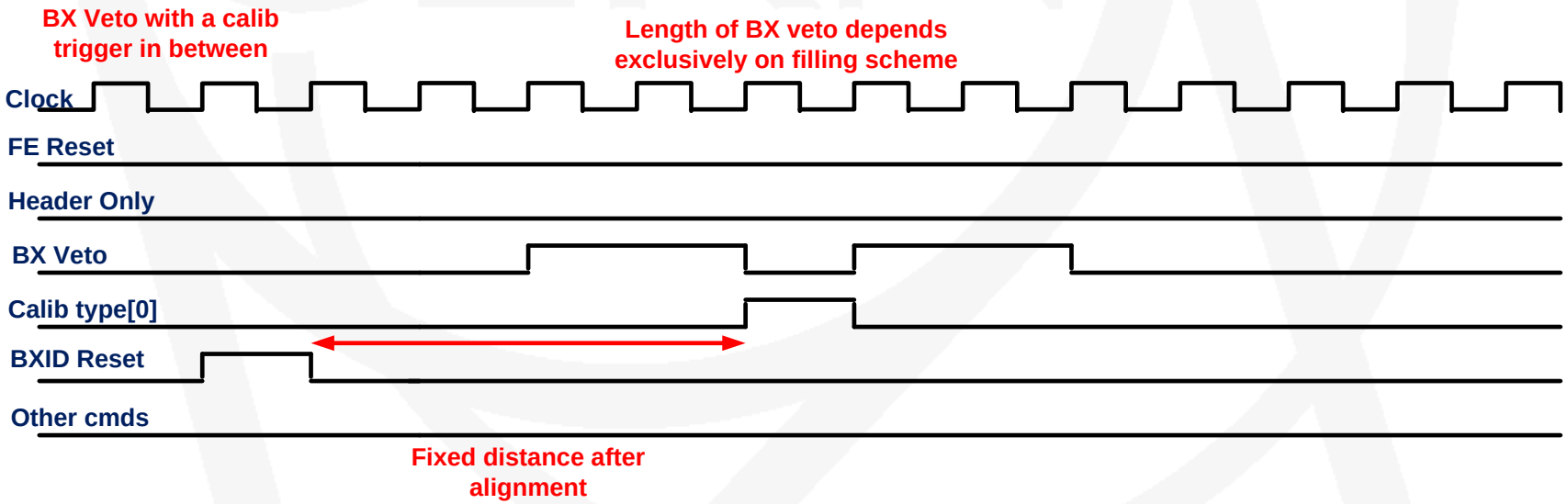
→ Read out all status and counter registers in a "latched" way
- ✓ Latch monitoring registers on snapshot bit, which is set periodically (programmable) and also single shot
- ✓ When snapshot bit is received, send all data via ECS field in TFC

"SYNCH"

→ See after ☺

Data flow control scheme

- TFC commands are synchronous wrt to BXID Reset
  - → once we align BXID Reset with beam, TFC commands come ALWAYS at the same latency
    - (wrt to BXID Reset, hence BXID)!
  - → Compression/suppression logic should act accordingly to TFC command
    - (why would you want to compress/suppress if that crossing is rejected a priori? Especially if your pre-processing is dynamic…)

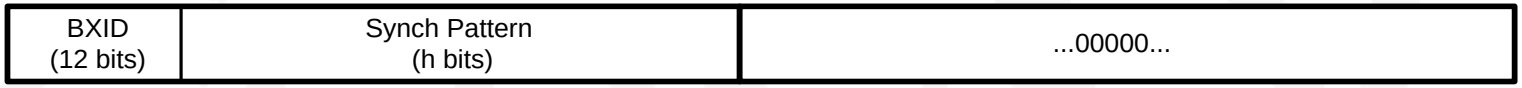


- Data is filtered according to TFC commands and the FE buffer status
- Data is packed onto the GBT link in a continuous fashion

What to do on SYNCH command?

When a Synch command is received

→ replace data packet by a specific Synch Pattern with full BXID (12 bits)



→ Synch Pattern should be programmable via ECS (in length and content)

Synch command is meant to be sure that system is synchronized… in a synchronous way!

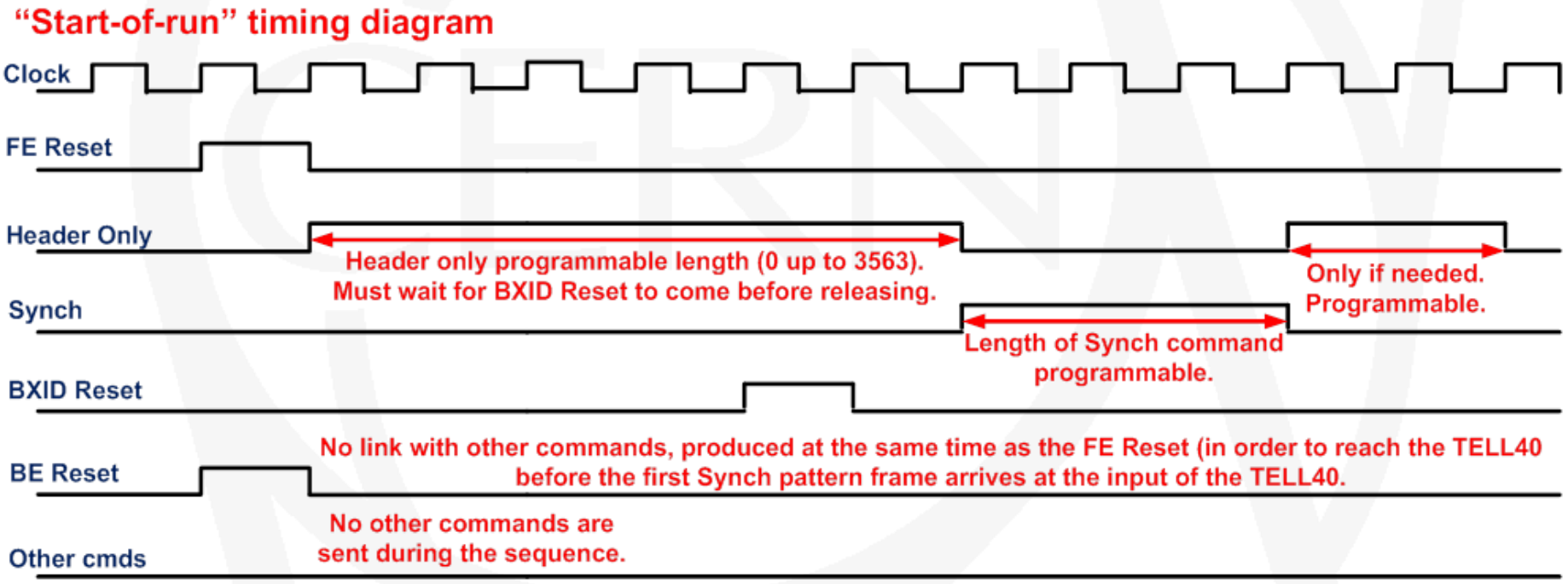
Double usage (in AND or in OR):

1. *Periodically:* i.e., SYNCH command sent every *n* Hz
   → this is intended as a safe synchronicity check!

2. *Asynchronously*: i.e. when a desynch is detected
   → TELL40 detects wrong frames, wrong packing, fast diagnostics in TELL40 specific sub-detectors' codes.
   → makes sense to clear the FE buffer
   → could be sent only for a local sub-detector from SOL40
   - i.e. could be fast triggered either by ECS or by TELL40 via SODIN through SOL40….

→ FEs send Synch Pattern for the same BXIDs everywhere
- TELL40 aligns to corresponding frame and BXID
- FE frees its memory : delete its content, read and write pointers back to empty
- FE sends Synch Pattern
  → TELL40 naturally goes on packing the preceding events in the buffers

TFC *start-of-run* fast sequence

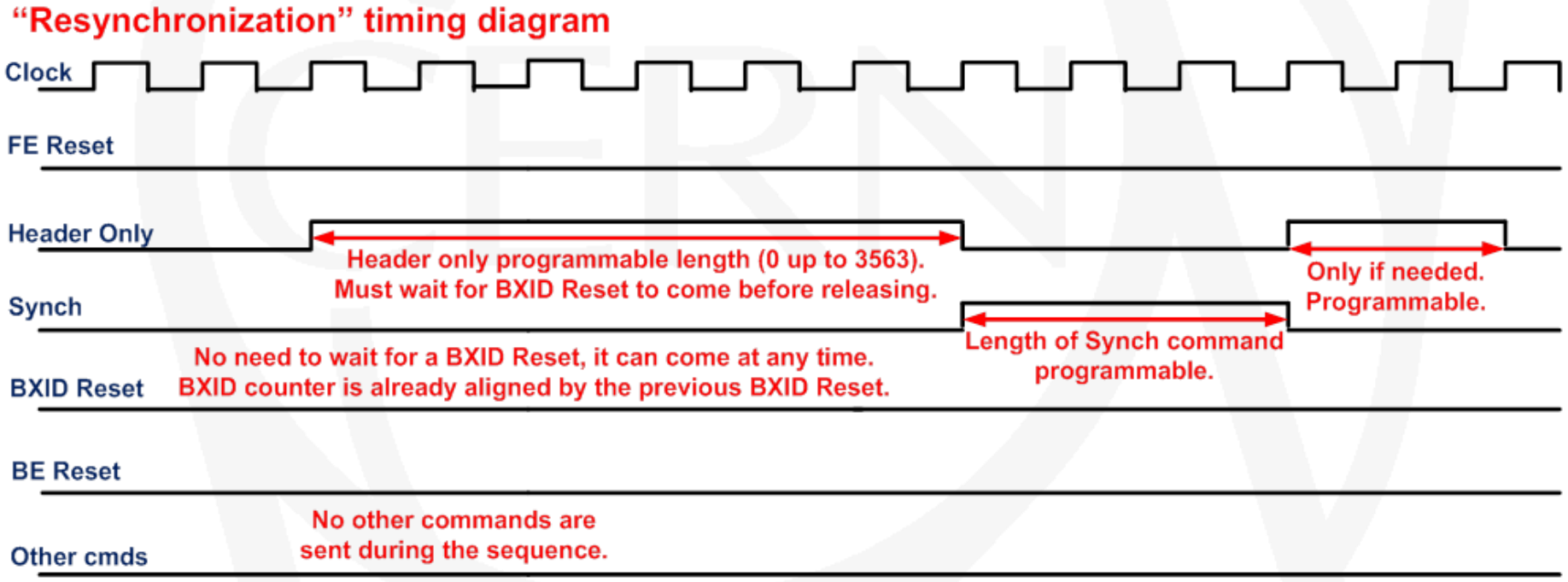


FE Reset is by definition completely asynchronous:

- Can come from control system (ECS) at change Run
- Can be periodic in the firmware
- Can be enabled by another processes (if programmed)

This mechanism will be generated everytime a FE Reset is issued.

Re-synchronization sequence



Example of a synch command being requested

→ From ECS or fast via TELL40.

→ Synch command is entirely programmable in frequency and length, while its position can be completely asynchronous or programmable.

→ Header Only is entirely programmable in length.

On the re-synchronization sequence ...

The Synch command is of outmost importance. Without it the TELL40 cannot decode your input data format!

Take good care in where you implement your Synch command logic!

- If you implement it at the output of your buffer, you must clear the buffer, reset the read/write pointers and then start from a clean sheet (as of specs).
  - ✓ In this case the TELL40 must truncate all other events in between (it will anyway as it doesn't receive the deleted events).
  - ✓ TFC can minimize the losses by sending Header Only to FE, reject events with trigger at TELL40 before a Synch, in any case if this is the implementation than we have necessarily a loss of data if the Synch command is transmitted periodically.
  - ✓ On the other hand, the rate can be pretty low (O(10 Hz)), so the loss can be minimal with the payback of having a system which is regularly re-synchronized.
  - ✓ However: how are you going to ensure that the synch pattern is for the correct BXID if you apply it at the output of your FE buffer?

- If you implement it at the input of your buffer, then a regular synch command can help fast diagnose de-synchronization of the data link.
  - ✓ Data will be consecutive and the synch frame can just be a cross-check mechanism.
  - ✓ This feature would be desirable especially if you are sending a shortened BXID.

On the re-synchronization sequence ...

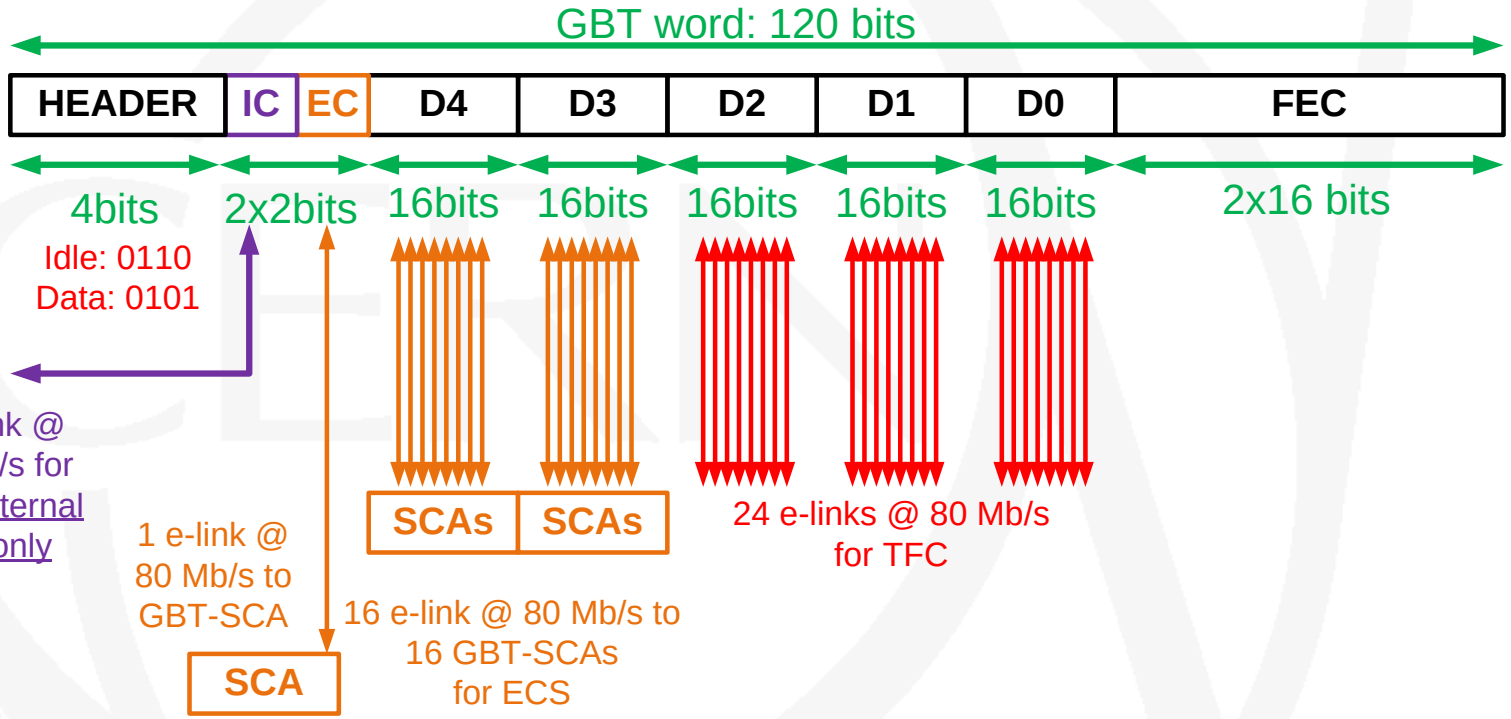
The Synch command is of outmost importance. Without it the TELL40 cannot decode your input data format!

Take good care in where you implement your Synch command logic!

- If you implement it at the output of your buffer, you must clear the buffer, reset the read/write pointers and then start from a clean sheet (as of specs).
  - ✓ In this case the TELL40 must truncate all other events in between (it will anyway as it doesn't receive the deleted events).
  - ✓ TFC can minimize the losses by sending Header Only to FE, reject events with trigger at TELL40 before a Synch, in any case if this is the implementation than we have necessarily a loss of data if the Synch command is transmitted periodically.
  - ✓ On the other hand, the rate can be pretty low (O(10 Hz)), so the loss can be minimal with the payback of having a system which is regularly re-synchronized.
  - ✓ However: how are you going to ensure that the synch pattern is for the correct BXID if you apply it at the output of your FE buffer?

- If you implement it at the input of your buffer, then a regular synch command can help fast diagnose de-synchronization of the data link.
  - ✓ Data will be consecutive and the synch frame can just be a cross-check mechanism.
  - ✓ This feature would be desirable especially if you are sending a shortened BXID.

The TFC+ECS GBT protocol to FE



→ Mapping of TFC commands on GBT bits can be customized to your needs
- Number of SCAs, speed of e-link and position
- Also which TFC commands and their position and if they need to be copied many times
- Make you sure you get your mapping right

→ Come to me and let's add it in the SOL40 firmware, this is needed to compile a firmware and see how many FPGA resources it may take

→ It may have an impact on how many SOL40 you need

Conclusion

The FE and BE specifications were already finalized and published > three years ago.

→ A new version, LHCb-PUB-2012-017 v1.4 is available now
- ✓ No major differences, only fixing the changes in the TFC architecture and clearing up some incomplete sentences.

→ In agreement with Ken, we think it is important to have each sub-system responsible/experts read the document and have a meeting session where we «sign-off» the document (virtually ☺)
- ✓ This meeting should happen relatively soon between June and July with me, Ken and Guillaume to make sure we are all on the same page.

→ Not to forget: all features I described here are already in the TFC/MiniDAQ firmware, can be tried out in simulations and can be controlled with the WinCC panels we published with the framework.
- ✓ Contact me if you need to know anything.

Backup

A word on rate regulation

- ✓ Throttle mechanism is meant to «pull the brake» if driving too fast
  - → Simple idea: veto events by putting Trigger bit sent to TELL40 to 0 synchronously across all TELL40 for the same BXID (or same EvID)

- ✓ Today, throttle is completely asynchronous and effectively rejects «L0yes» decisions from hardware trigger
  - → It is applied as long as a TELL1 does not release the throttle
  - → The event is rejected at the FE (for all FEs) which simply do not store fragments of that particular BXID in the derandomizer buffer

- ✓ In the upgrade scenario, events fragments have already been recorded and sent to the TELL40 asynchronously
  - → TELL40 must not «lag» behind otherwise its buffer will get full
    - Its input stage must be able to cope with a fully trigger-less, throttle-less readout system
    - But if the TELL40 has trouble (see next), a *throttle* must be applied
    - See next slides

Throttle mechanism in the upgrade

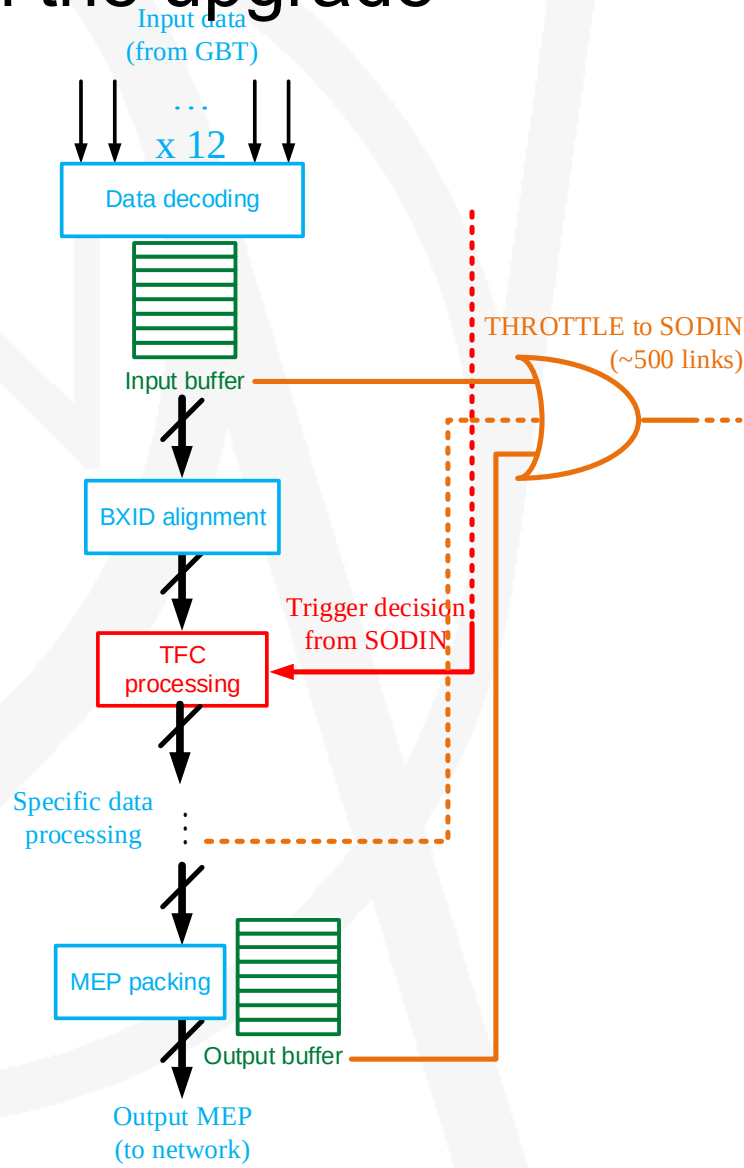
Throttle could come in three main places in TELL40

1. Input buffer
   - ✓ When full, it cannot accept events at the input anymore
2. Specific processing block
   - ✓ «A la ZS of today», if dedicated processing taking too much time or getting stuck
3. Output buffer
   - ✓ When full, it cannot accept MEPs to be sent out

What to do then?

→ TELL40 should raise a flag to go to SODIN, who will then «veto» events

But already discussed this and we concluded that ...



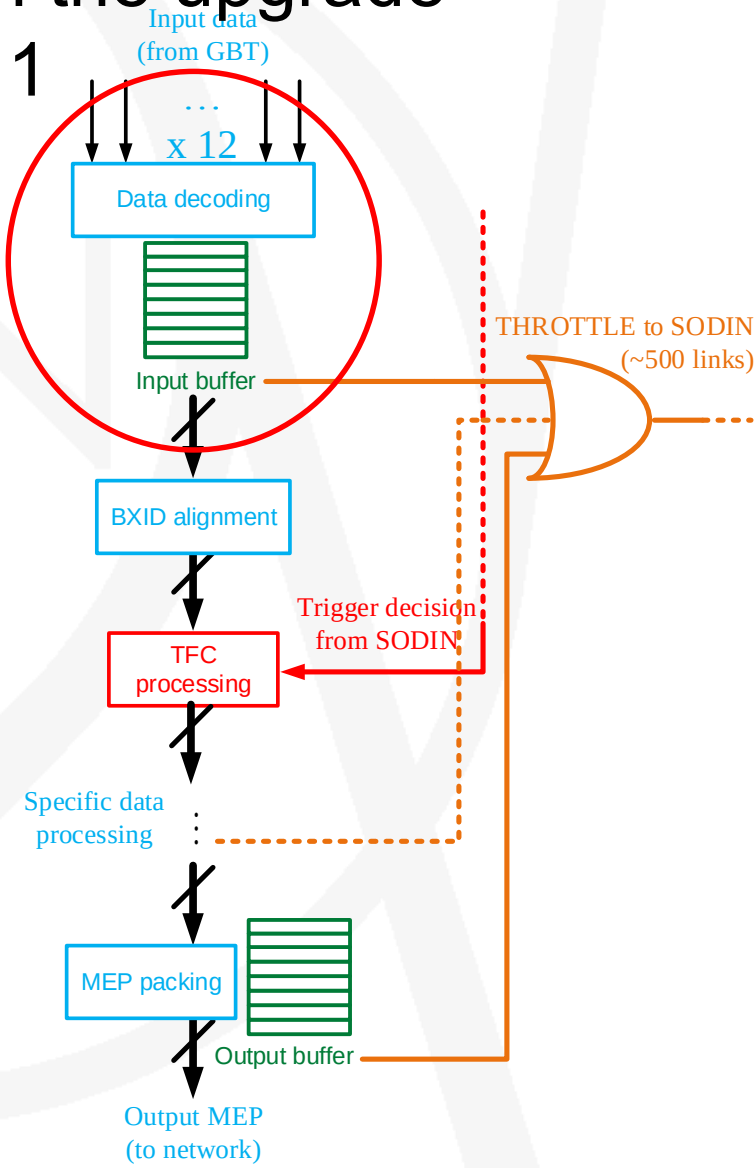
Throttle mechanism in the upgrade Scenario 1

... but ...

1. Throttling at input is useless, because TFC decision to TELL40 will come after
   - ✓ TELL40 must be able to cope with maximum full load at input stage

→ Only way to help would be if SODIN will reduce rate at FE

- ✓ By sending BX_VETOs or HEADER ONLY
  - But not all sub-detectors will implement them
- ✓ And SODIN will be 150m away from FE, so the TELL40 would need to throttle at least 30-40 clock cycles before to actually be effective....

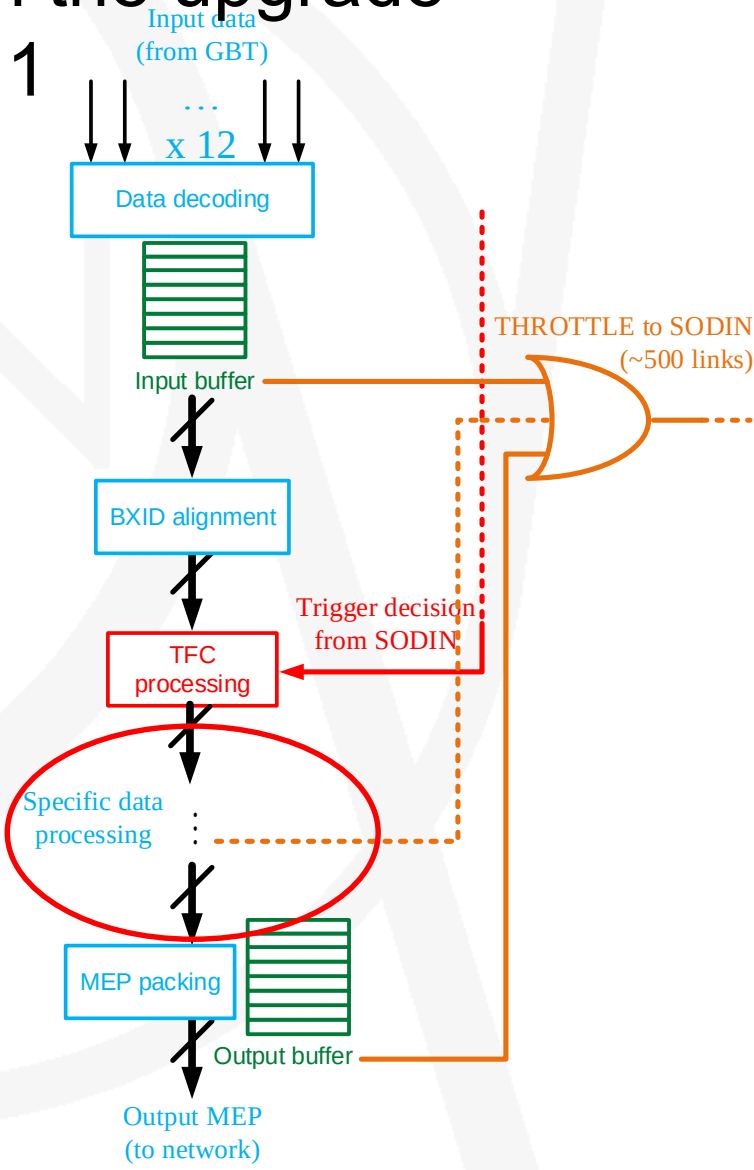


Throttle mechanism in the upgrade Scenario 1

... but ...

2. Throttling at data processing could be useful
   - ✓ it would allow emptying the input buffer quickly by veto-ing events
   - ✓ recuperate possible time lost during specific data processing or later (output link down?)

→ Question is:
   - ✓ Do sub-detectors really need to do something inside the TELL40?
     - We are already struggling with resources...
     - Everything will be in the FARM anyway...
     - In any case, always suggested to have fixed latency processing block



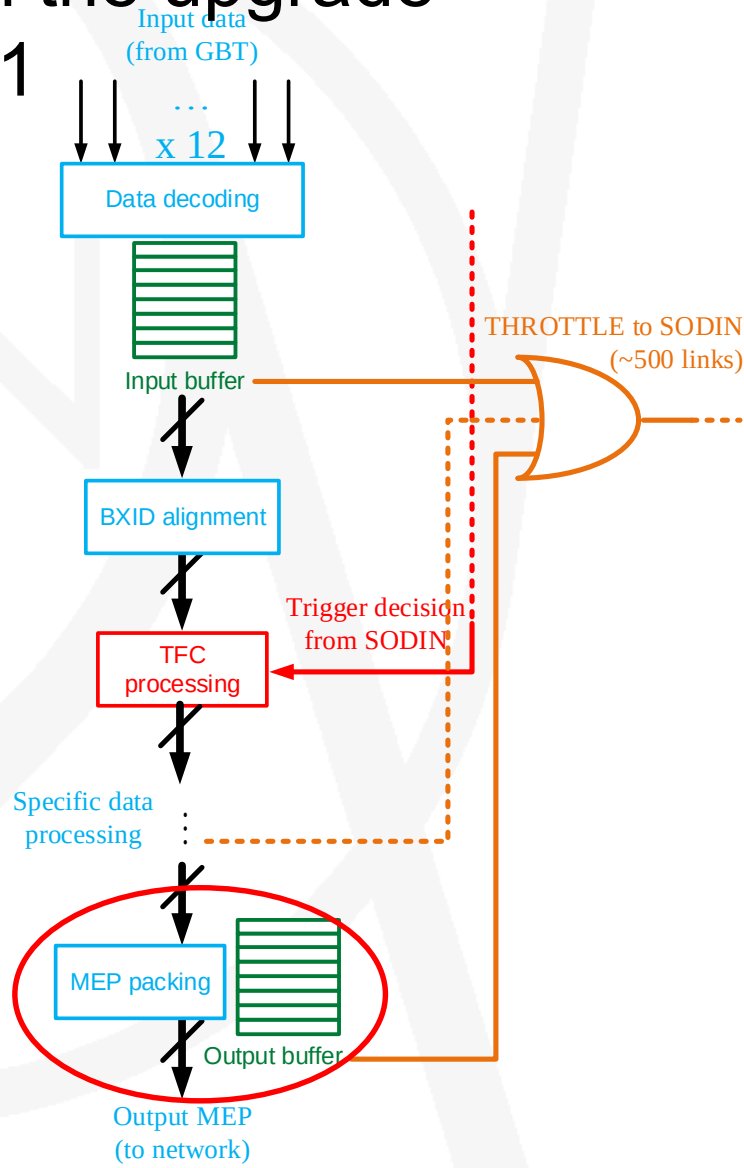
Throttle mechanism in the upgrade Scenario 1

... but ...

2. Throttling at output stage would alleviate backpressure coming from having a problem in flushing the output buffer

→ But in this case:

- ✓ This is not a real throttle, likely the system has a problem
  - Not just few clock cycles, but lots
- ✓ Probably blockage in one output buffer which will need a physical action
  - Symptom is event builder can't build full event cause missing some MEPs
    - HLT rate drops to 0
    - Stop trigger, reset, restart.



Throttle mechanism in the upgrade Scenario 2

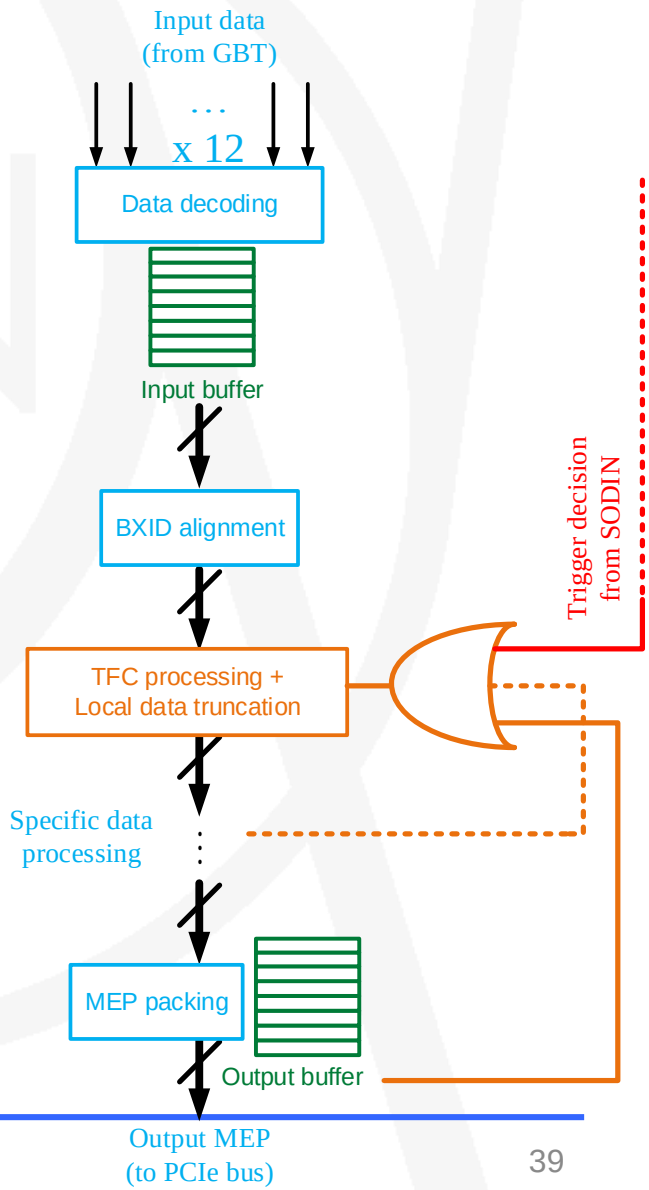
... it doesn't seem to be necessary to implement a throttle that is transmitted to SODIN by all TELL40 to rate regulate the system

New proposal is that the TELL40 behaves exaclty like a FE chip:

→ TRUNCATION when throttle is applied

- ✓ Strip data, keep header, pass event along
- ✓ This must be signalled in the Ev header

- ✓ Such a scenario will allow each TELL40 to unblock quickly and locally
- ✓ What is really needed is a lot of monitoring to be able to monitor if something happens too often
- ✓ It is then a matter of «fine tuning» the system to find the right balance
  - The risk is having many truncated events all over the place, so ...



«Gears» in data taking

→ If throttle/truncation information is sent to SODIN anyway, then SODIN can monitor this and can apply some kind of «intelligent» rate regulation
  - ✓ If truncation happens too often, there is a problem of some sort, probably the best way is to break and shift down a gear
  - ✓ If it's a real blockage, system will just stop sending events
  - ✓ It it's a momentaneous lag, then SODIN can help unblock the system

→ A possibility is to implement «gears» in SODIN, where at each gear a different (avg) prescale on the input rate is applied
  - This can be driven internally in the firmware
  - Each gears is not a fixed rate, but changes according to the current «rpm»
  - (Suggest to start with 6 gears ☺).