Readout Control specifications for FE and BE in the upgraded LHCb readout system

LHCb Electronics Upgrade Meeting

26 July 2012

Federico Alessio

Richard Jacobsson

Outline

Readout control specifications published in LHCb-INT-2012-018

→ direct impact on how FE electronics should be designed (i.e. GBT configuration)

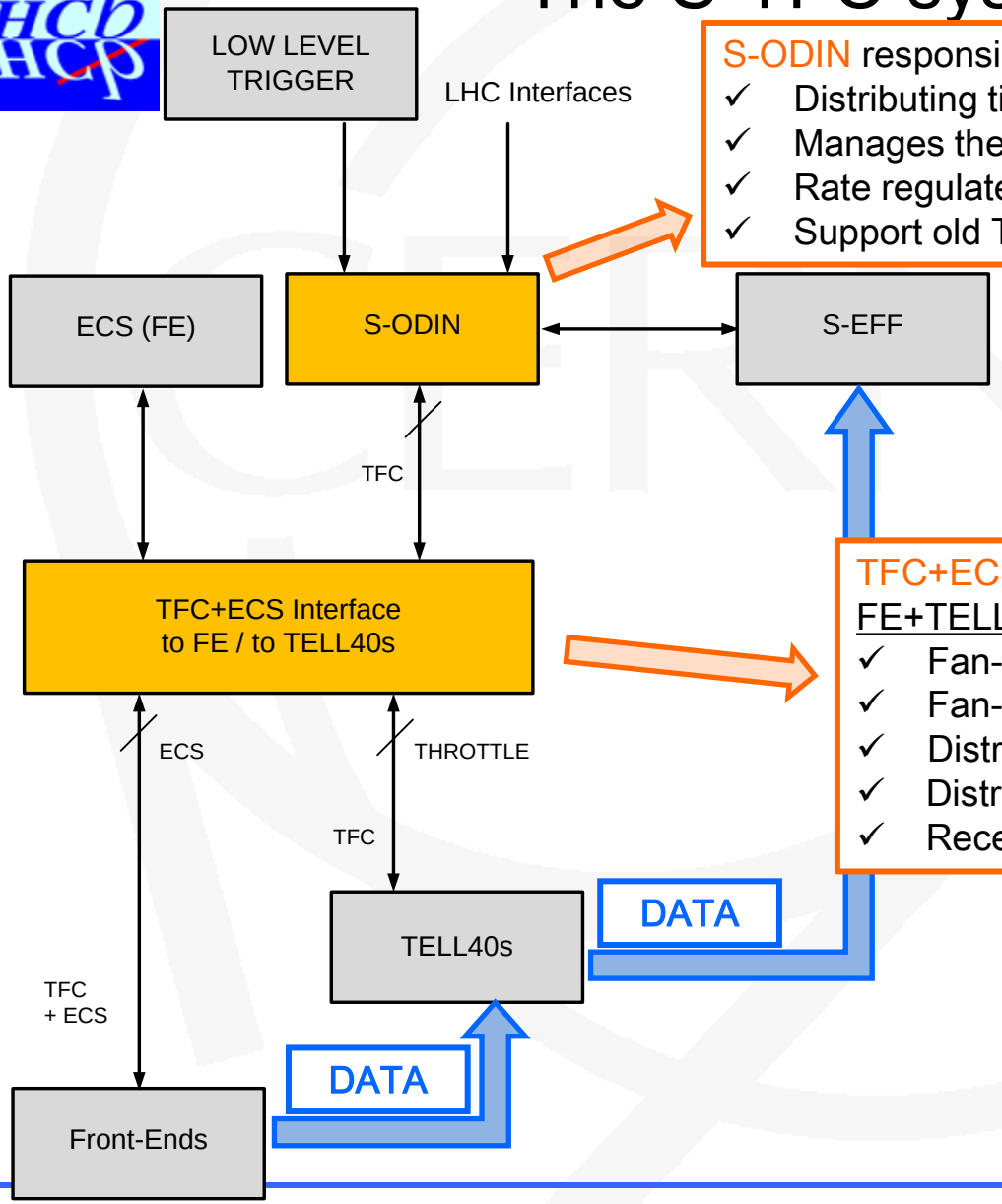
→ first iteration with experts, we need feedback! (thanks to who already commented…)

- ✓ we will re-circulate to a wider audience and we will have a dedicated meeting

In this presentation few selected topic, but document has MANY more details…:

1. Quick reminder from S-TFC system-level specifications (LHCb-PUB-2012-001)
   - ✓ Logical system
   - ✓ Physical system
2. TFC commands/resets to TELL40
   - ✓ Protocol
3. TFC commands/resets to FE
   - ✓ Protocol
4. Readout control sequences
   - ✓ Description of TFC commands+resets
5. GBT configuration in FE
   - ✓ How to use FE with current specs
6. Timing distribution to FE
7. ECS configuration and monitoring of FE via GBT
8. Q&A

The S-TFC system at a glance



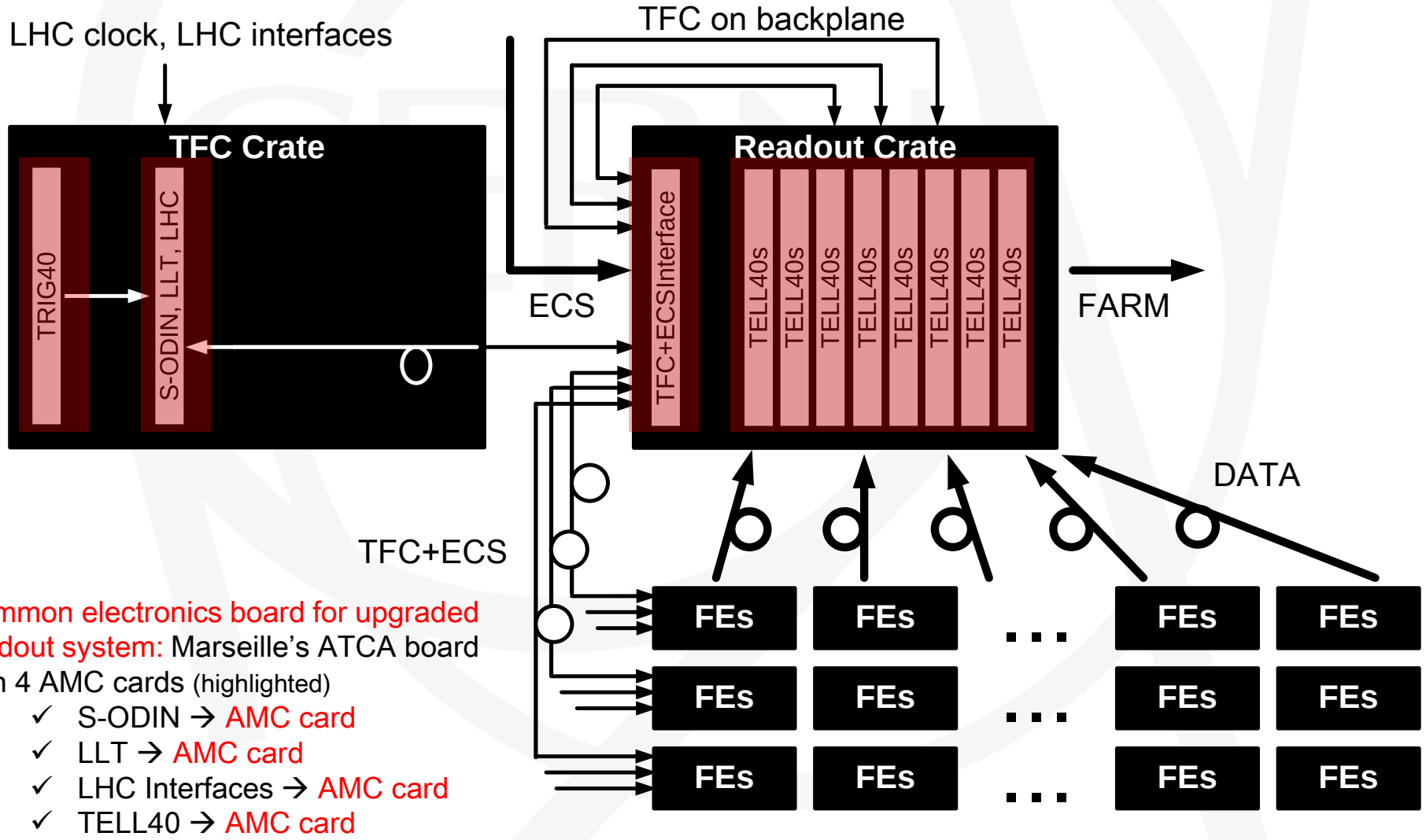
S-ODIN responsible for controlling upgraded readout system

- ✓ Distributing timing and synchronous commands
- ✓ Manages the dispatching of events to the EFF
- ✓ Rate regulates the system
- ✓ Support old TTC system: *hybrid system!*

TFC+ECSInterface responsible for interfacing FE+TELL40 slice to S-ODIN

- ✓ Fan-out TFC information to TELL40
- ✓ Fan-in THROTTLE information from TELL40
- ✓ Distributes TFC information to FE
- ✓ Distributes ECS configuration data to FE
- ✓ Receives ECS monitoring data from FE

The upgraded physical readout slice



Common electronics board for upgraded readout system: Marseille's ATCA board with 4 AMC cards (highlighted)

- ✓ S-ODIN → AMC card
- ✓ LLT → AMC card
- ✓ LHC Interfaces → AMC card
- ✓ TELL40 → AMC card

S-TFC protocol to TELL40

We will provide the TFC decoding block for the TELL40: VHDL entity with inputs/outputs

✓ TFC Word to TELL40 via TFC+ECSInterface:

→ 44 bits sent every 40 MHz = 1.76 Gb/s (on backplane)

→ all commands are associated to BXID in TFC word

43 .. 32	31 .. 16	15 .. 12	11..8
BXID(11..0)	**MEP Dest(15..0)**	**Trigger Type(3..0)**	**Calibration Type(3..0)**

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

Constant latency after S-ODIN

✓ THROTTLE Information from each TELL40 to TFC+ECSInterface:

→ 1 bit for each ATCA board connected (OR of AMC cards)

→ + BXID for which the throttle was set

S-TFC protocol to FE

✓ TFC Word on downlink to FE via TFC+ECSInterface:

→ 24 bits in each GBT frame every 40 MHz = 0.98 Gb/s

→ all commands associated to BXID in TFC word

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

Major requirement: put local configurable delays for each TFC command

→ GBT does not support individual delays

→ Need for «local» pipelining: detector delays+cables+operational logic (i.e. laser pulse?)

→ DATA SHOULD BE TAGGED WITH THE CROSSING TO WHICH IT BELONGS!

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

→ Accounting of delays in S-ODIN: for now, 16 clock cycles earlier + time to receive

→ Aligned to the furthest FE

TFC protocol to FE has implications on GBT configuration and ECS to/from FE

→ see later

S-TFC readout control sequences

"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

"FE RESETS"

→ Bit set for one clock cycle in TFC word

→ 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: S-ODIN will ensure no data is being accepted during the FE reset process (wait to the slowest by setting Header Only bit → no data is recorded at FE).
- ✓ Reset TELL40 data input logic: the same bit is sent to TELL40 for same BXID.

"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…

"BX VETO"

→ Based excusively on filling scheme, processing of that particular event is inhibited

- ✓ Only header (or few bits) in data word if this bit is set
- ✓ Allows "recuperating" buffer space in a LHC-synchronous way

S-TFC readout control sequences

"CALIBRATION TYPE" COMMAND

→ 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
- ✓ S-ODIN overrides LLT decision at TELL40
  - Periodic or calibration higher priority, but lower rate

"NZS MODE"

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

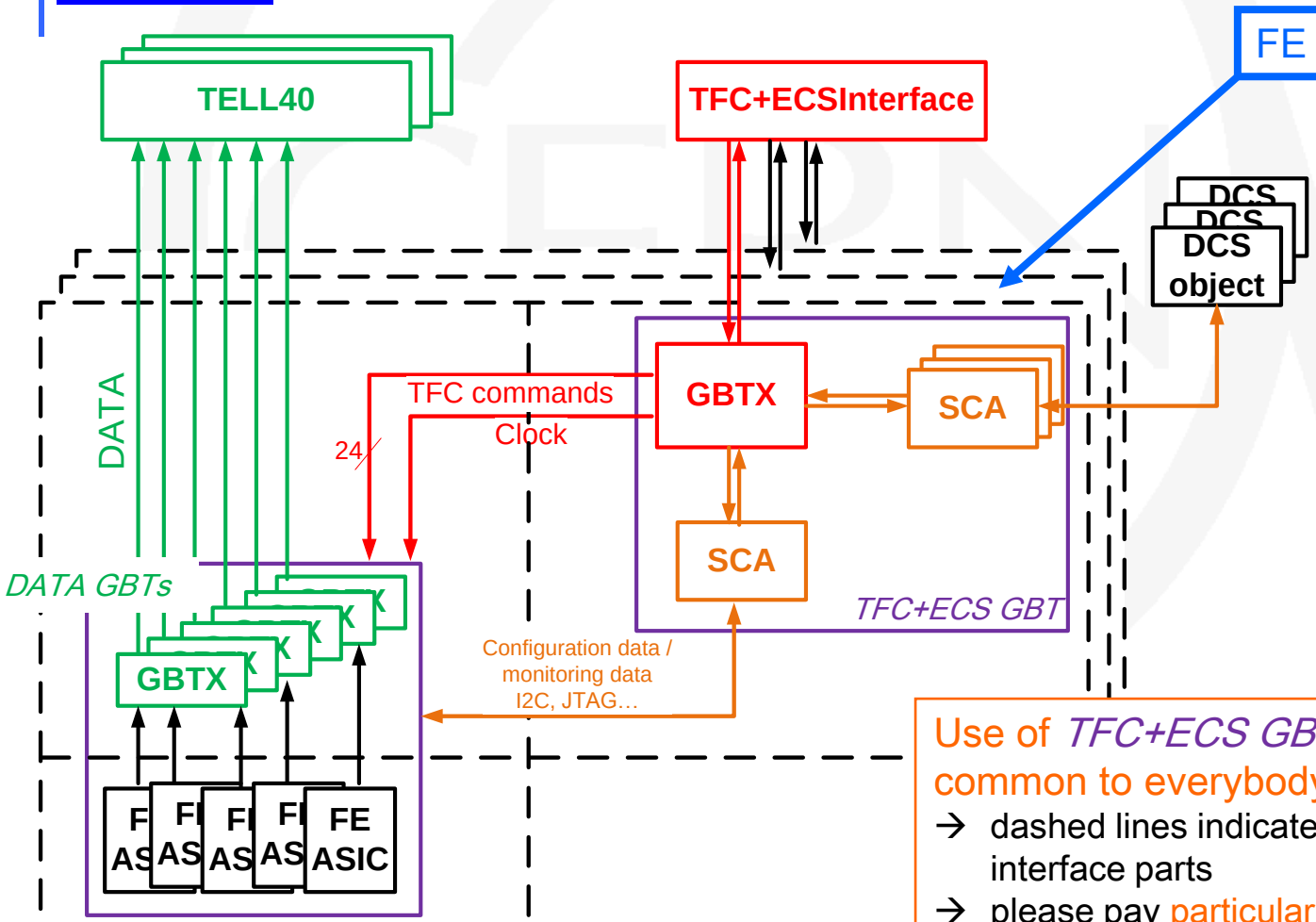
→ Possible to have also multi-NZS readout: *consecutive NZS events*
- ✓ S-ODIN will take care of sending Header Only for a defined set of clock cycles later to allow recuperating buffer space (programmable as well, to the slowest of the detector)

What are the sub-detector requests in term of NZS? What do you need and what do you want to do?

"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

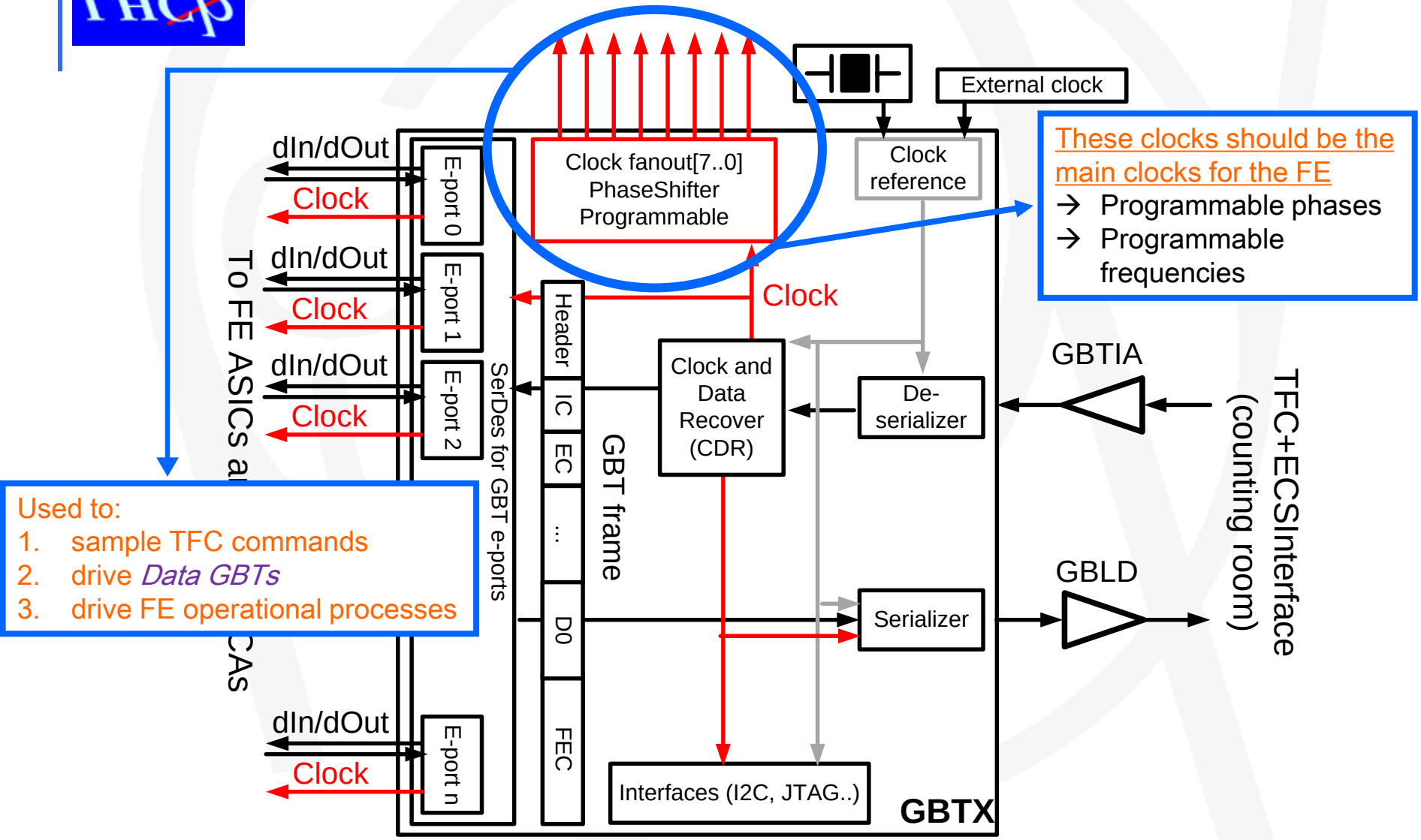
How to decode TFC in FE chips?



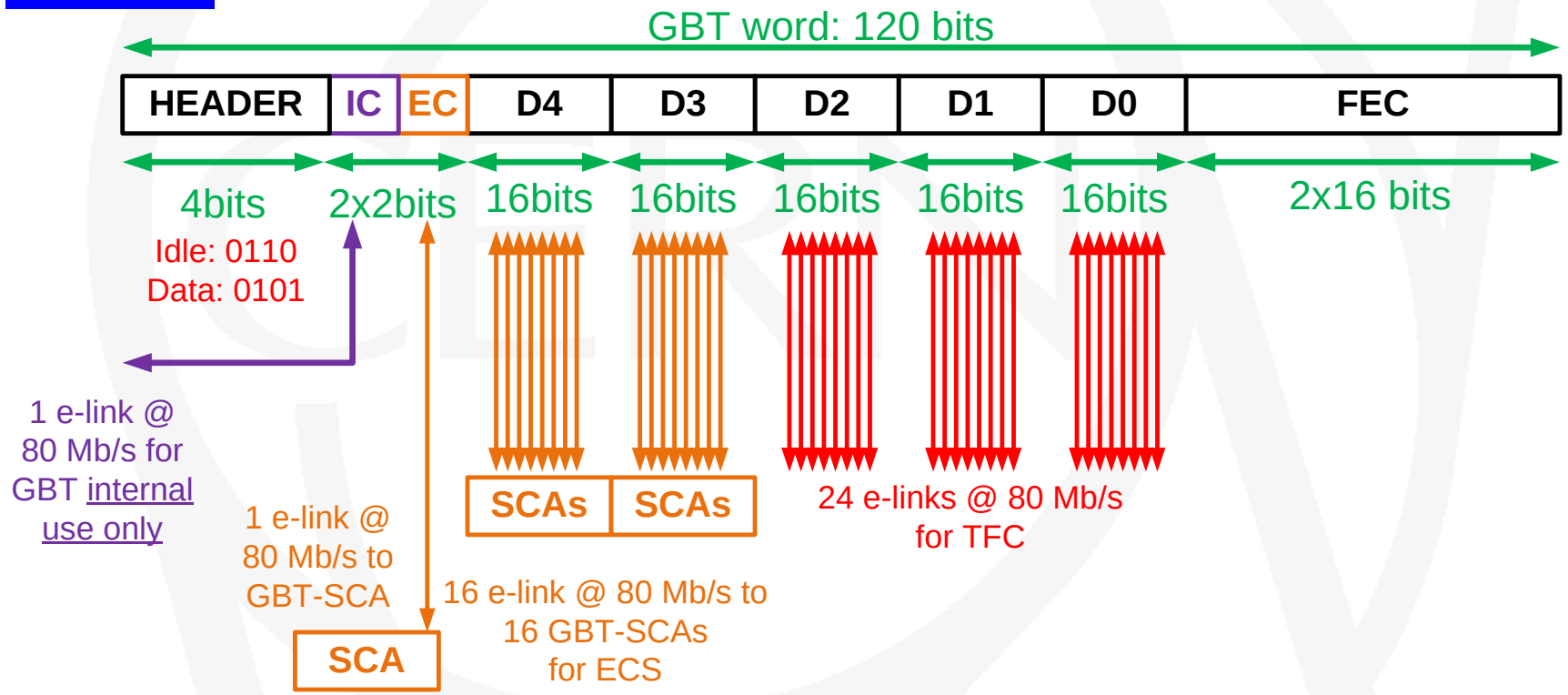
Use of *TFC+ECS GBTs* in FE is 100% common to everybody!!

- → dashed lines indicate the detector specific interface parts
- → please pay particular care in the clock transmission: the TFC clock must be used by FE to transmit data, i.e. low jitter!
  - ✓ Kapton cable, crate, copper between FE ASICs and GBTX

The TFC+ECS GBT



The TFC+ECS GBT protocol to FE

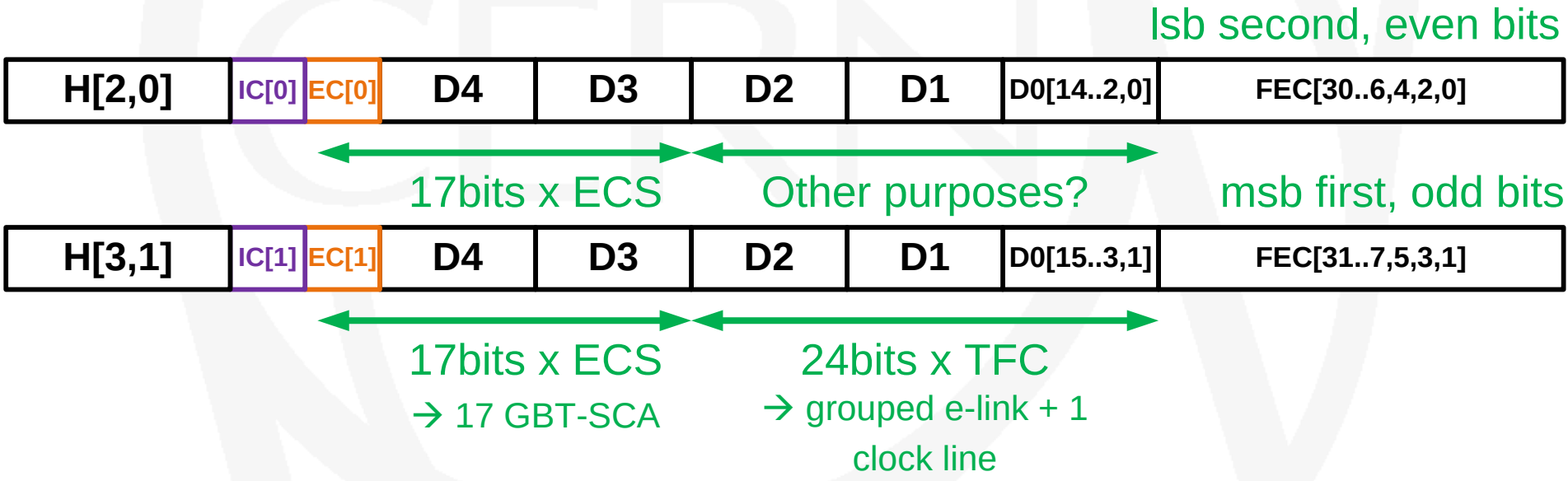


→ TFC protocol has direct implications in the way in which GBT should be used everywhere

- ✓ 24 e-links @ 80 Mb/s dedicated to TFC word:
  - use 80 MHz phase shifter clock to sample TFC parallel word
- ✓ TFC bits are packed in GBT frame so that they all come out on the same clock edge

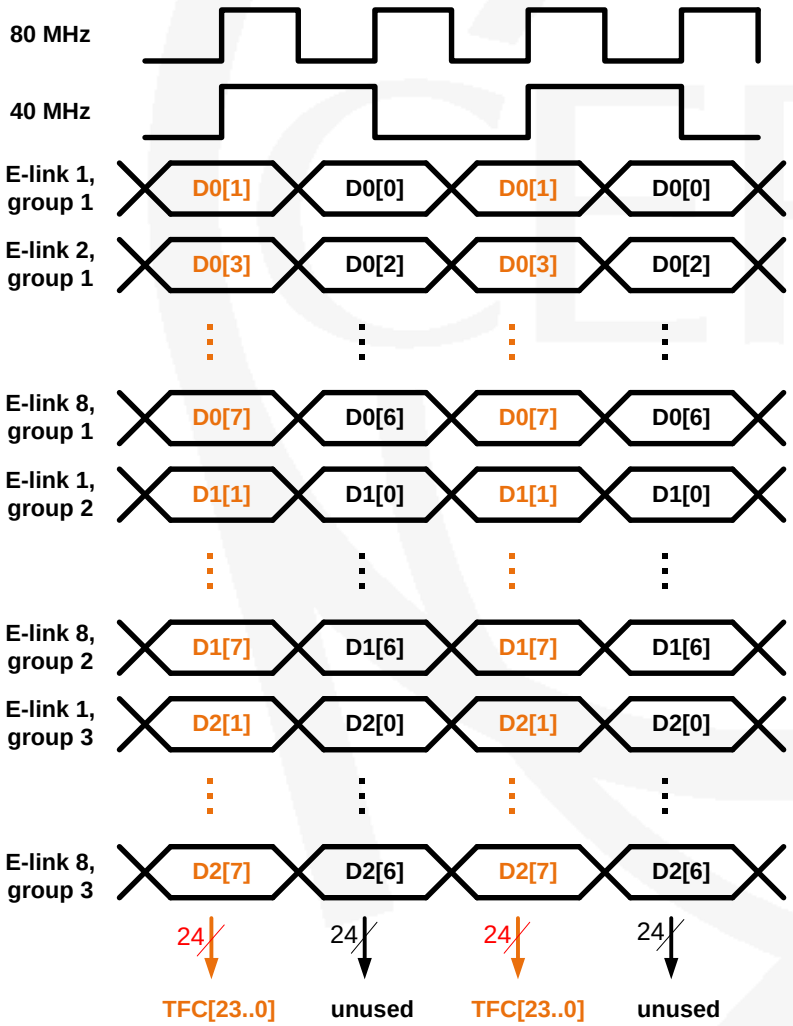
→ Leftover 17 e-links dedicated to GBT-SCAs for ECS configuring and monitoring (see later)

Words come out from GBT at 80 Mb/s



→ Odd bits of GBT protocol on rising edge of 40 MHz clock (first, msb),
→ Even bits of GBT protocol on falling edge of 40 MHz clock (second, lsb)

TFC decoding at FE after GBT



This is crucial!!

→ we can already specify where each TFC bit will come out on the GBT chip

→ this is the only way in which FE designers still have minimal freedom with GBT chip

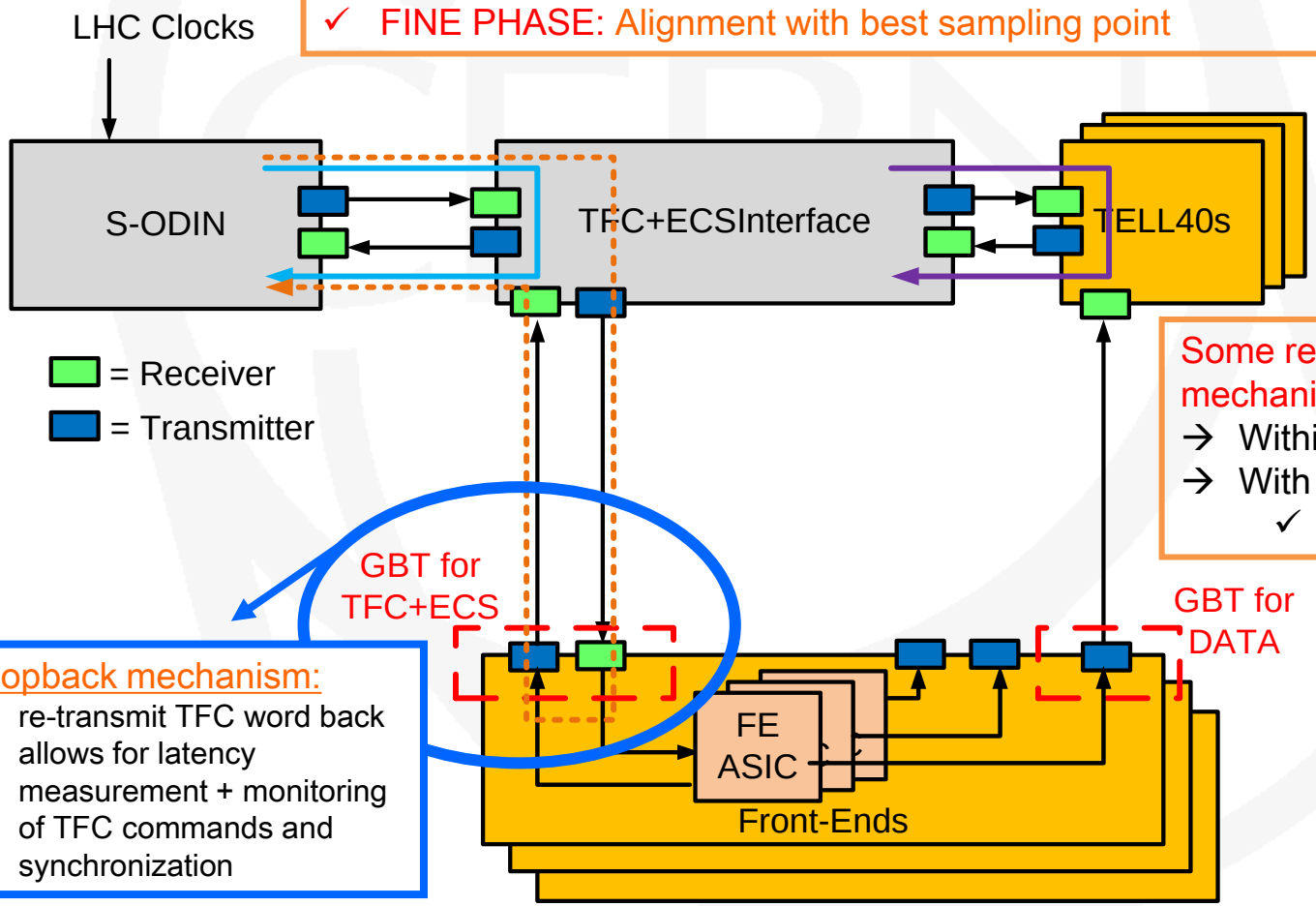
- ✓ if TFC info was packed to come out on only 12 e-links (first odd then even), then decoding in FE ASIC would be mandatory!
- ✓ which would mean that the GBT bus would have to go to each FE ASIC for decoding of TFC command

→ there is also the idea to repeat the TFC bits on even and odd bits in TFC protocol

- ✓ would that help?
- ✓ FE could tie logical blocks directly on GBT pins…

Timing distribution

From TFC point of view, we ensure constant:
- ✓ LATENCY: Alignment with BXID
- ✓ FINE PHASE: Alignment with best sampling point



Some resynchronization mechanisms envisaged:
- → Within TFC boards
- → With GBT
  - ✓ No impact on FE itself

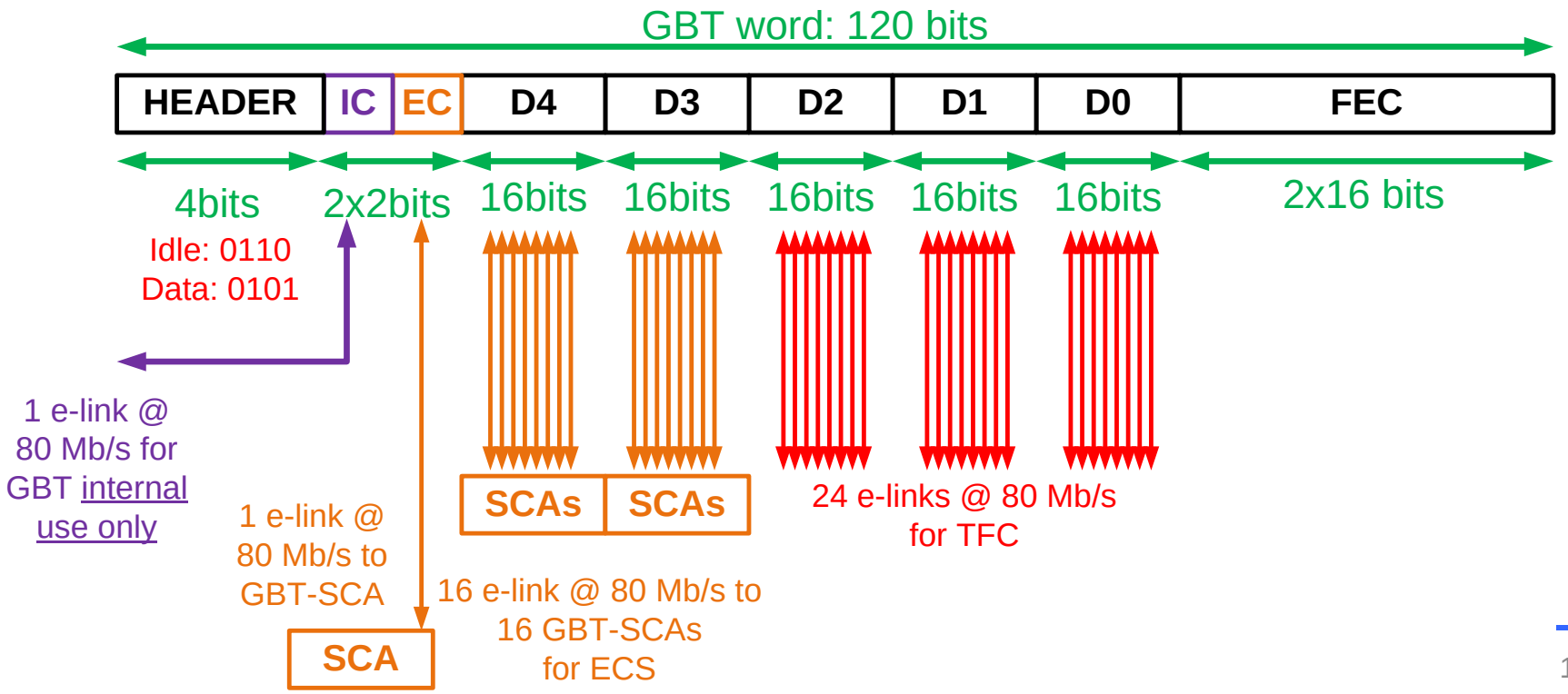
Loopback mechanism:
- → re-transmit TFC word back
- → allows for latency measurement + monitoring of TFC commands and synchronization

Now, what about the ECS part?

We are not forgetting it, but it is *slightly easier*.

Each pair of bit from ECS field inside GBT can go to a GBT-SCA

- ✓ One will be needed to configure the *Data GBTs* (that could be the EC one).
- ✓ The rest can go to either FE ASICs or DCS objects (temperature, pressure) via SCAs
- ✓ GBT-SCA chip has already everything for us: interfaces, e-links ports ..



Now, what about the ECS part?

We are not forgetting it, but it is *slightly easier*.

Each pair of bit from ECS field inside GBT can go to a GBT-SCA

- ✓ One will be needed to configure the *Data GBTs* (that could be the EC one).
- ✓ The rest can go to either FE ASICs or DCS objects (temperature, pressure) via SCAs
- ✓ GBT-SCA chip has already everything for us: interfaces, e-links ports ..

→ Many FE chips can be connected to same GBT-SCA

- ✓ Provided they use the same bus (and same protocol)
- ✓ Provided we have a proper addressing scheme at the FE side (and its mapping)

→ Or (viceversa) many GBT-SCAs can be connected to the same FE chip

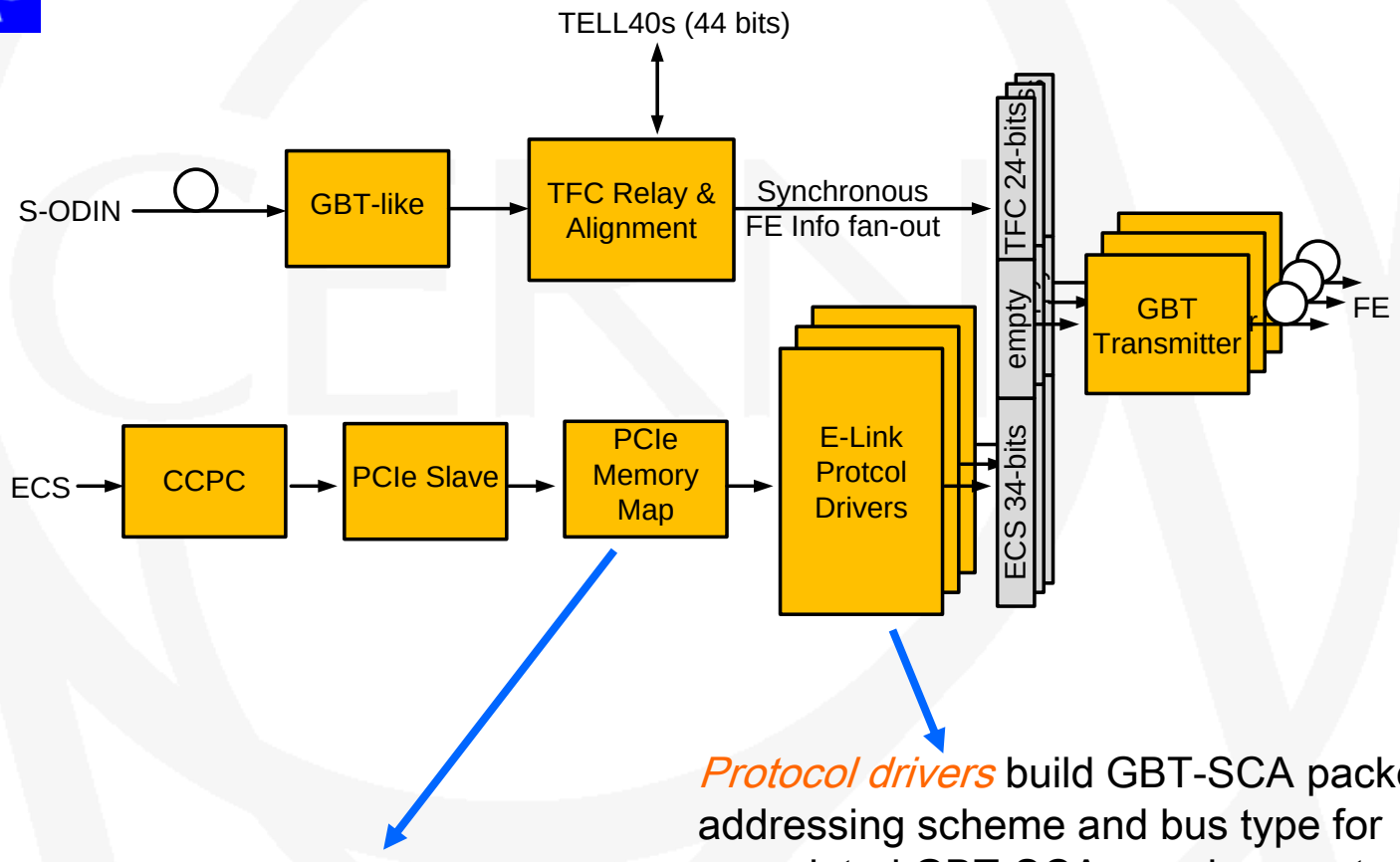
- ✓ If needed to have more than 80 Mb/s for each FE chip ECS… (really?!?)

To make this works, the only thing we need is to be able to:

- ✓ *drive the right FE chip at the right address*
- ✓ *with the right GBT-SCA*
- ✓ *with the right protocol for the chosen bus*

→TFC+ECSInterface will do that in the most flexible way possible

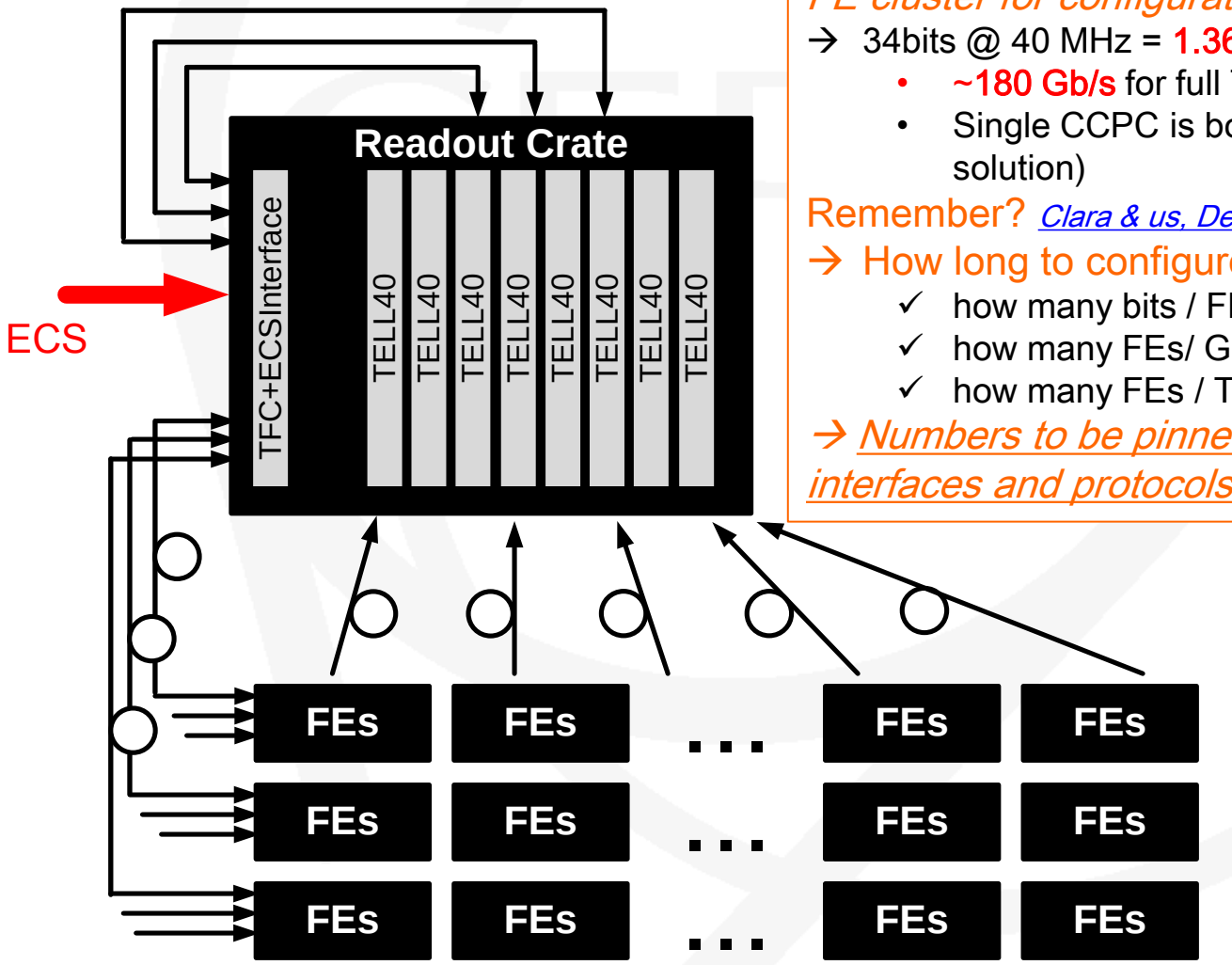
TFC+ECS Interface encoding block to FE!



Memory Map with internal addressing scheme for GBT-SCA chips + FE chips addressing, e-link addressing and bus type: content of memory loaded from ECS

Protocol drivers build GBT-SCA packets with addressing scheme and bus type for associated GBT-SCA user busses to selected FE chip (GBT-SCA supported protocols)

Usual considerations …



TFC+ECSInterface has the ECS load of an entire FE cluster for configurating and monitoring

→ 34bits @ 40 MHz = **1.36Gb/s** on single GBT link

- **~180 Gb/s** for full TFC+ECSInterface (132 links)
- Single CCPC is bottleneck… (local storage as solution)

Remember? *Clara & us, December 2011*

→ How long to configure FE cluster?

- ✓ how many bits / FE?
- ✓ how many FEs/ GBT link?
- ✓ how many FEs / TFC+ECSInterface?

→ *Numbers to be pinned down soon + GBT-SCA interfaces and protocols.*

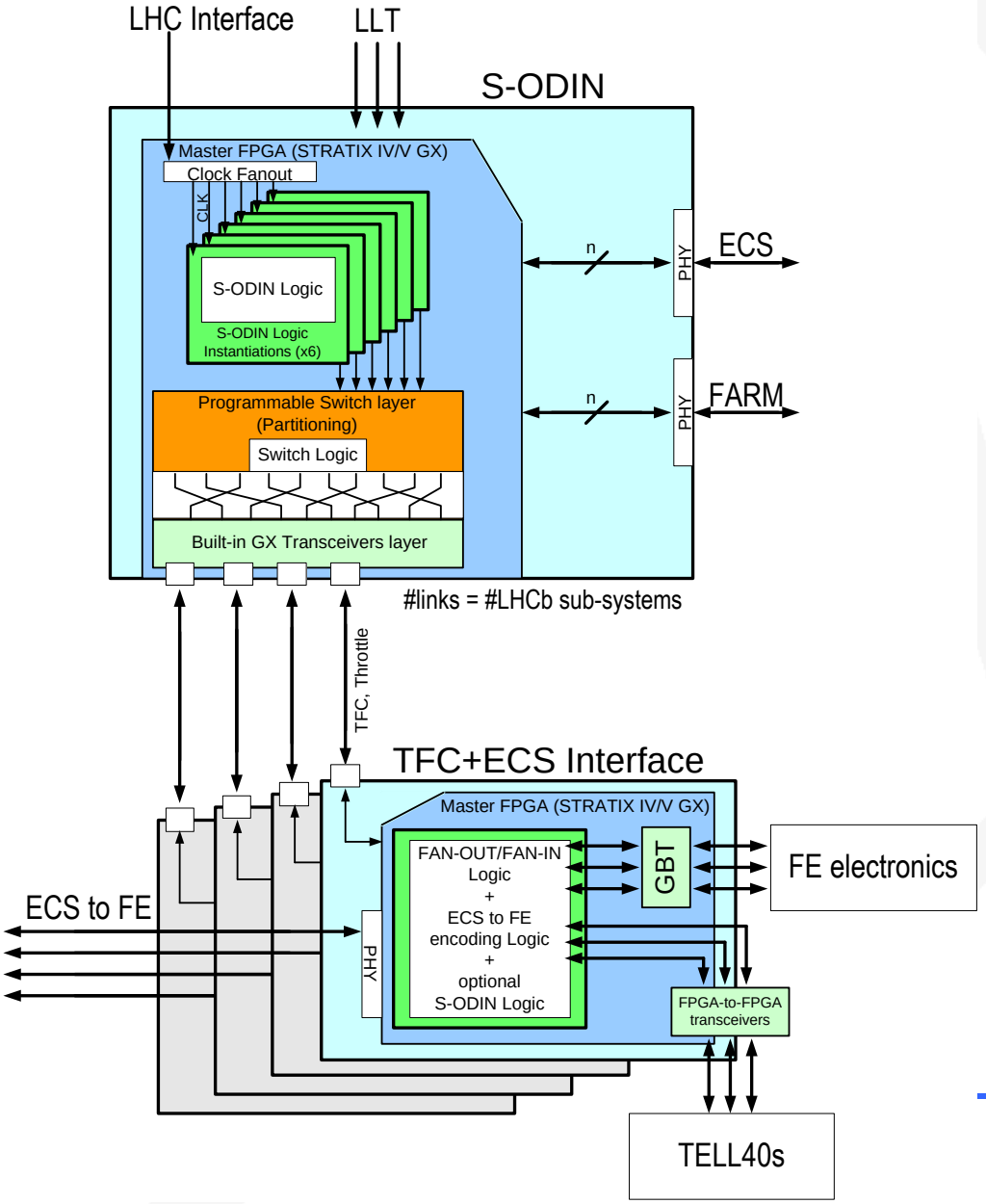
Questions & (maybe) Answers

Backups

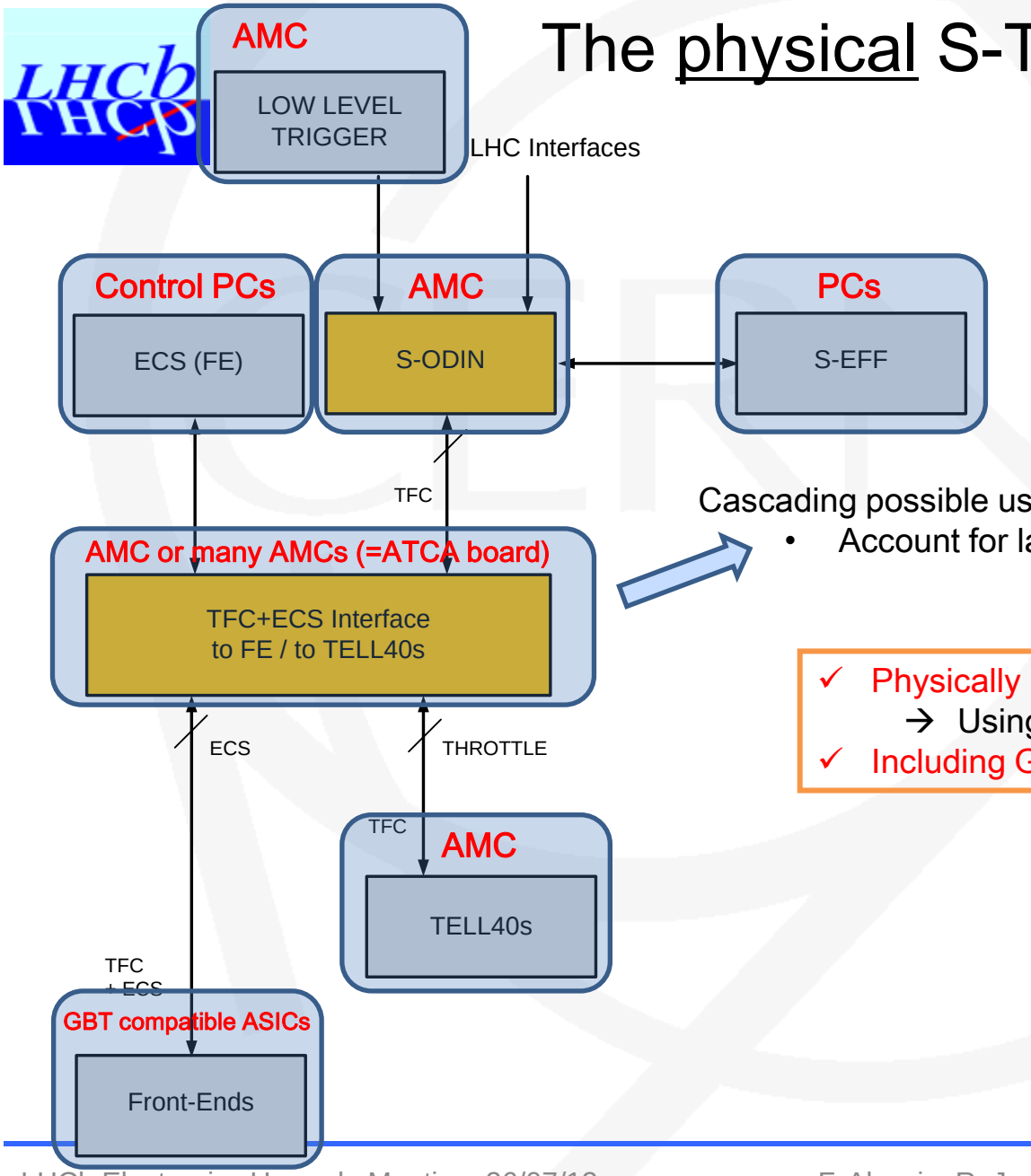
System and functional requirements

1. Bidirectional communication network
2. Clock jitter, and phase and latency control
   - ✓ At the FE, but also at TELL40 and between S-TFC boards
3. Partitioning to allow running with any ensemble and parallel partitions
4. LHC interfaces
5. Events rate control
6. Low-Level-Trigger input
7. Support for old TTC-based distribution system
8. Destination control for the event packets
9. Sub-detectors calibration triggers
10. S-ODIN data bank
    - ✓ Infomation about transmitted events
11. Test-bench support

S-TFC concept reminder



The physical S-TFC at a glance

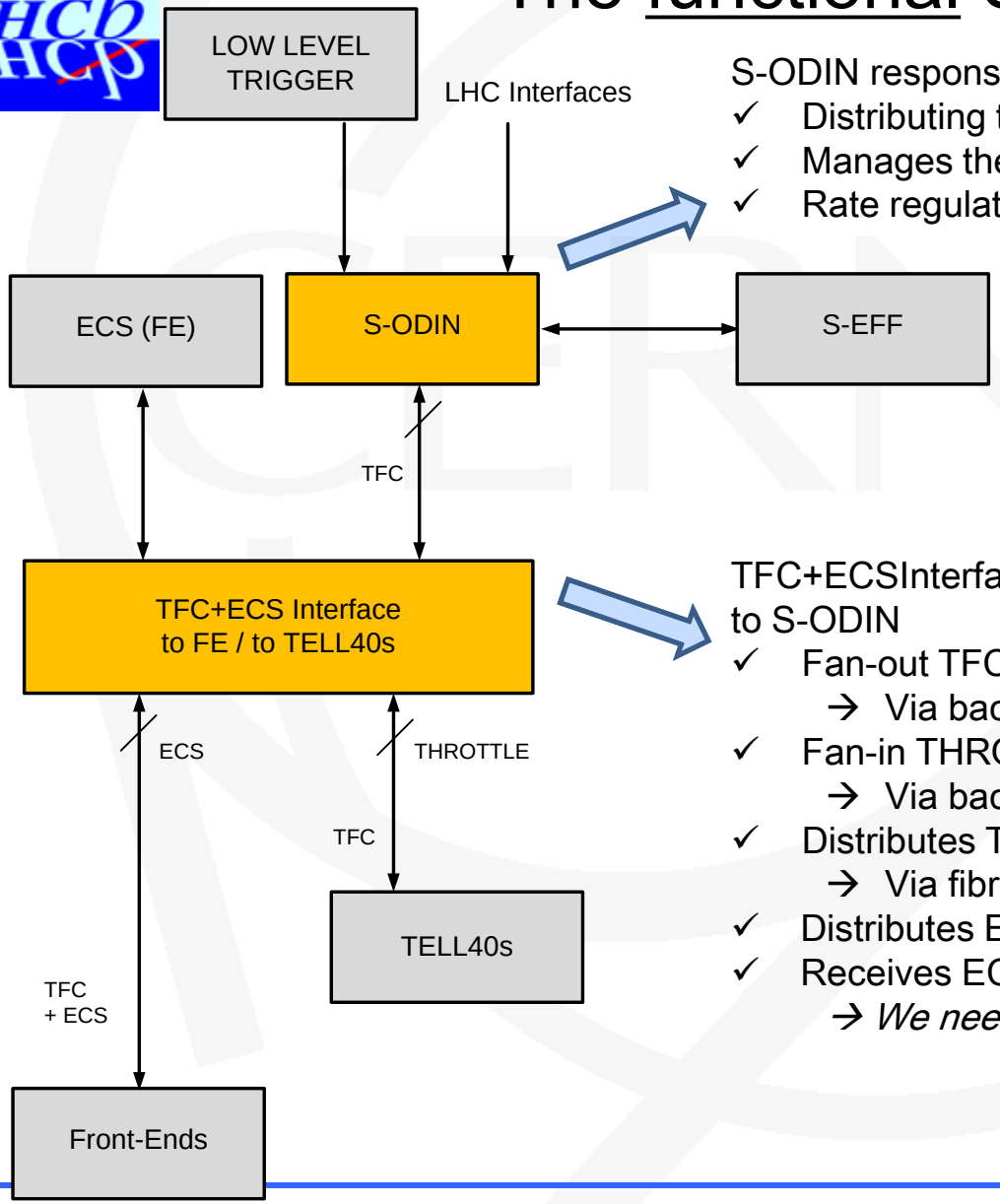


Cascading possible using a dedicated bidirectional link

- Account for latency at each cascaded stage

✓ Physically mapped on ATCA
→ Using Marseille's boards
✓ Including GBT for FE

The functional S-TFC at a glance



S-ODIN responsible for controlling entire upgraded readout

- ✓ Distributing timing and synchronous commands
- ✓ Manages the dispatching of events to the EFF
- ✓ Rate regulates the system

TFC+ECSInterface responsible for interfacing FE+BE to S-ODIN

- ✓ Fan-out TFC information to TELL40
  - → Via backplane
- ✓ Fan-in THROTTLE information from TELL40
  - → Via backplane
- ✓ Distributes TFC information to FE
  - → Via fibres + GBT
- ✓ Distributes ECS configuration data to FE
- ✓ Receives ECS monitoring data from FE
  - → *We need counters in FE!*

Partitioning

Partitioning is assured by having:

- ✓ Many instances of S-ODIN codes inside main FPGA
- ✓ Switching is done inside main FPGA
  - Simply assure that TFC information are sent to right output
- ✓ TFC+ECSInterface «interfaces» S-ODIN with partitioned FE+TELL40 slice(s)
  - Logical distribution of TFC+ECSInterfaces in TELL40s crates
  - Important to respect the «logical concept» of partitioning
    → Should not span over different sub-systems with same TFC+ECSInterface…
    → Need at least one dedicated TFC+ECSInterface for each sub-system

TFC Back-End control commands

- ✓ Control functions for Back-End
  - Same as Front-End
    - Bunch ID for synchronization check with internal counter and data from FE
    - Bunch Counter Reset
    - Event Counter Reset (reset of same counters as FE + all event related counters)
    - Header Only → Force FE to transmit only header and no data (Informative)
    - Calibration pulsing (informative)
    - Non-zero suppressed readout of current crossing (Informative)
    - Bunch Crossing Type Veto (Informative)
    - Front-End electronics reset (Expect only header from FE)
  - Back-End Reset (Header Only from FE during reset)
  - Trigger
    - Reject data (Header still sent to farm or not?)
    - Attention: In TFC word, the trigger (& MEP destination) is not associated to the transmitted BunchID and the rest of the TFC word
    - S-ODIN pipes the asynchronous local trigger information for the maximum latency possible for BE (How much buffering is available in BE, number of events?)
    - Realignment of all data for BE is done in TFC+FE interface via pipeline logic
  - Trigger Type to define type of event, processing type, destination type etc
  - Multi-Event Packet Destination IP
    - Transmitted when MEP should be closed and transmitted
  - Any other needs? → Reserve bits

TFC Front-End control commands

- ✓ Control functions for Front-End
  - Bunch ID for synchronization check with internal counter
  - Bunch Counter Reset
  - Event Counter Reset
    - Reset of counter for accepted crossings = crossings for which header+data was sent
    - Reset of counter of truncated events
    - And all other event related counters (TFC command counters, etc!)
  - Header Only → Force FE to transmit only header and no data
  - Calibration pulsing (How many types do we need?)
  - Non-zero suppressed readout of current crossing
    - Following *n* crossing will receive "Header Only" → Header only transmission
  - Bunch Crossing Type Veto based on crossing scheme from LHC
    - Send header only for empty crossings and most single beam crossings
  - Front-End electronics reset
    - During the time of the reset (common duration) Front-End receives "Header Only" command and should transmit header only
  - Any other needs? → Reserve bits
- ✓ All TFC commands (individual signal) require local configurable delay

«BX VETO»

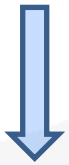
@ 40 MHz

S-ODIN vetoes the readout of an event

Based on filling scheme

→ Used to control the rate of readout while < 30MHz

→ INDEPENDENT FROM LLT DECISION!



FE can use this info to recuperate time for processing events

→ Only header for vetoed events

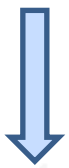
→ Flexible packing of data into GBT frame

Sending a «LLTyes»

@ 40 MHZ

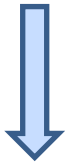
S-ODIN receives decisions from LLT

Special triggers



S-ODIN aligns and applies priority scheme on trigger types

Rate regulation
(next slide)

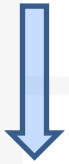


S-ODIN sends out a “LLTyes” to TELL40 at
a fixed latency wrt BXID!

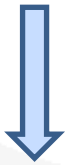
Rate regulation

@ 40 MHz

TELL40 raises the throttle bit



TFC Interfaces compiles a throttle word with BXID and sends it to S-ODIN

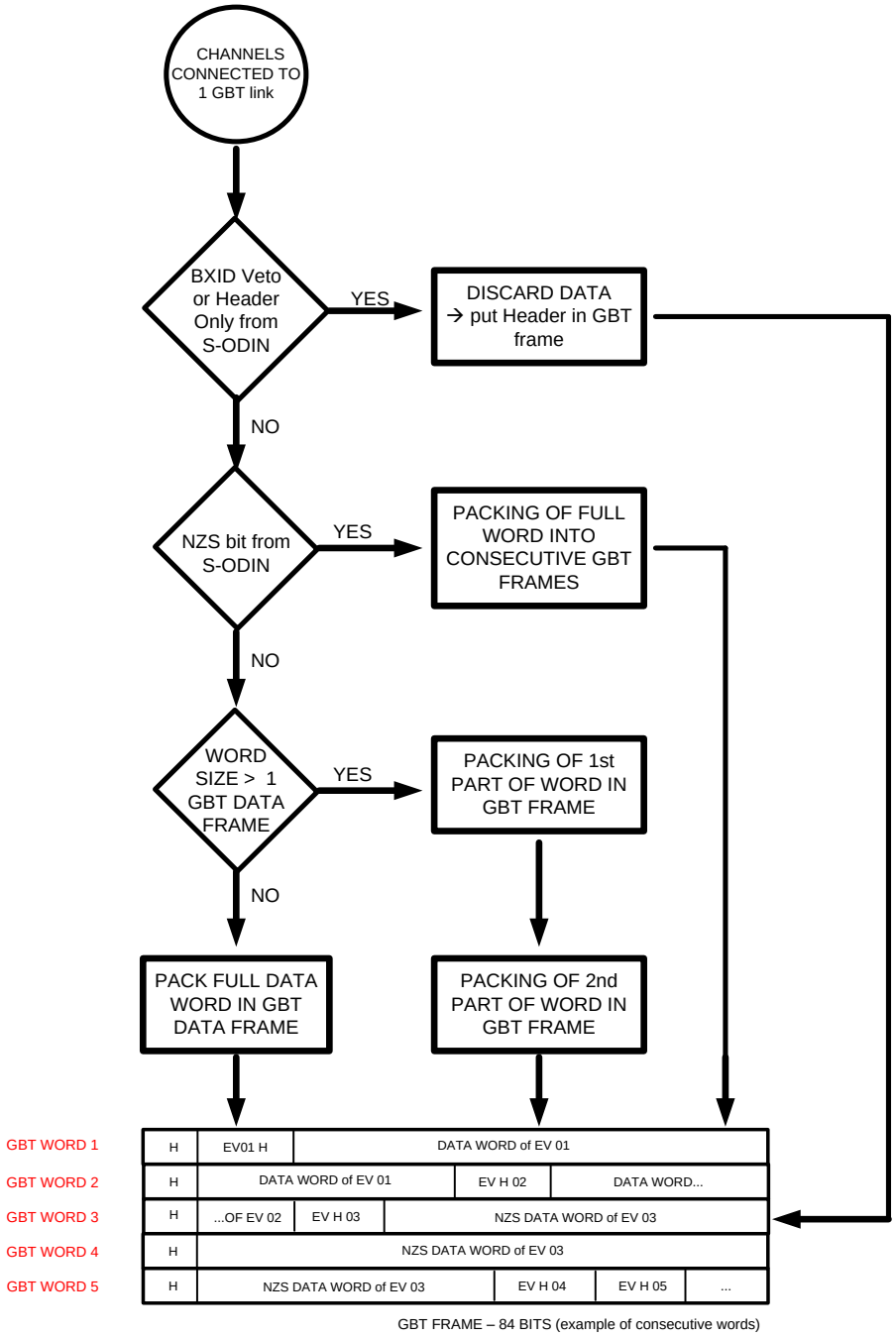


MEP request scheme
(next slide)

S-ODIN rejects event(s) until throttle is released

→ In practice: the subsequent “LLTyes”(s) become “LLTno”(s)!

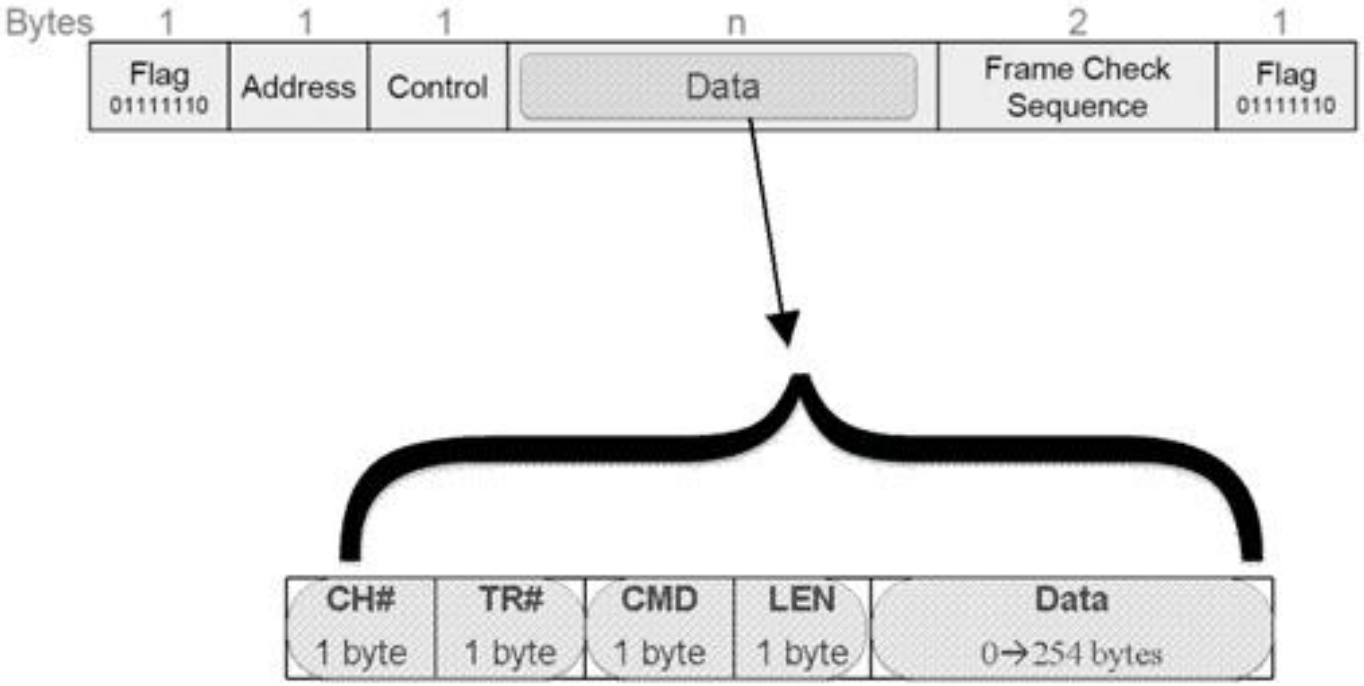
Example of readout mechanism



This scheme implies that the GBT bandwidth is exploited to its maximum:

- → Pack of events consecutively independently from GBT boundaries
- → Needs buffering to keep events stored waiting to be sent off
- → Header needs more information than just BXID
  - ✓ length of word, type of word…
- → Needs a well-defined truncation scheme and boundaries

Extra slide - GBT-SCA Packet

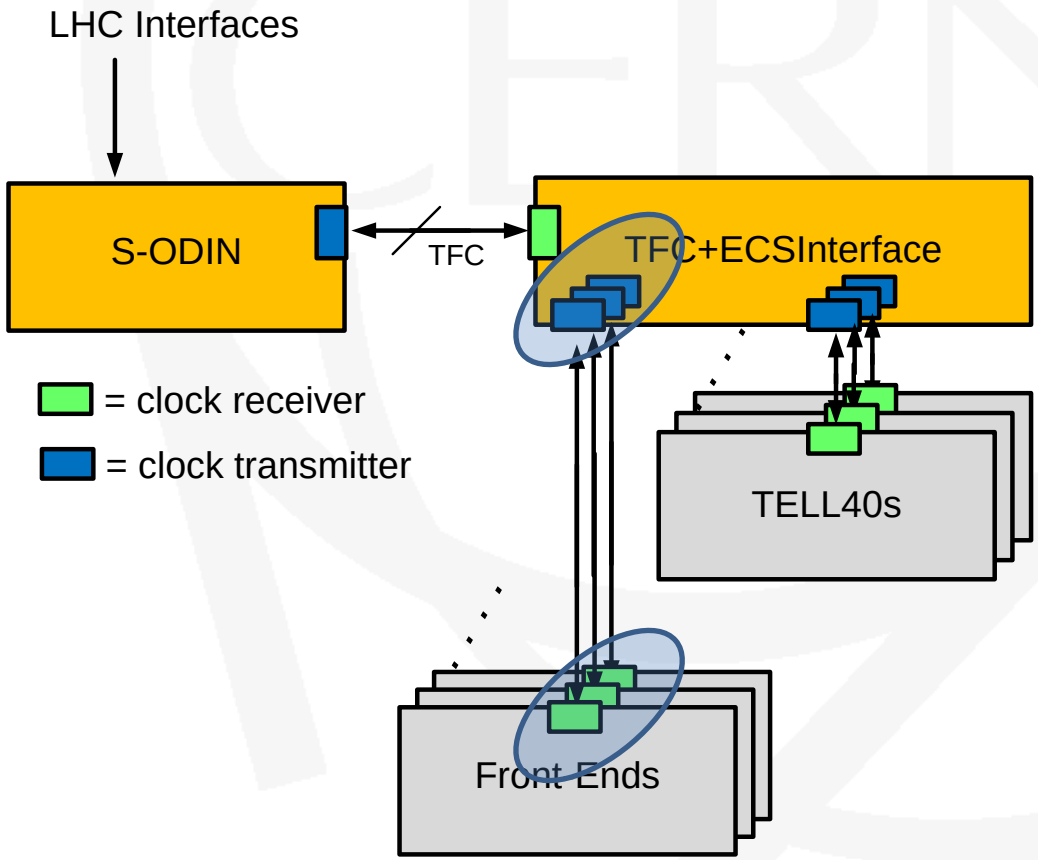


Easy: the GBT-SCA has everything set up (on paper…)

→ Only thing is to build proper protocol

→ FE designer selects the bus + addressing scheme, we built a generic entity in TFC+ECSInterface to drive it.

Clock distribution and phase/latency control



1. at the FE: GBT

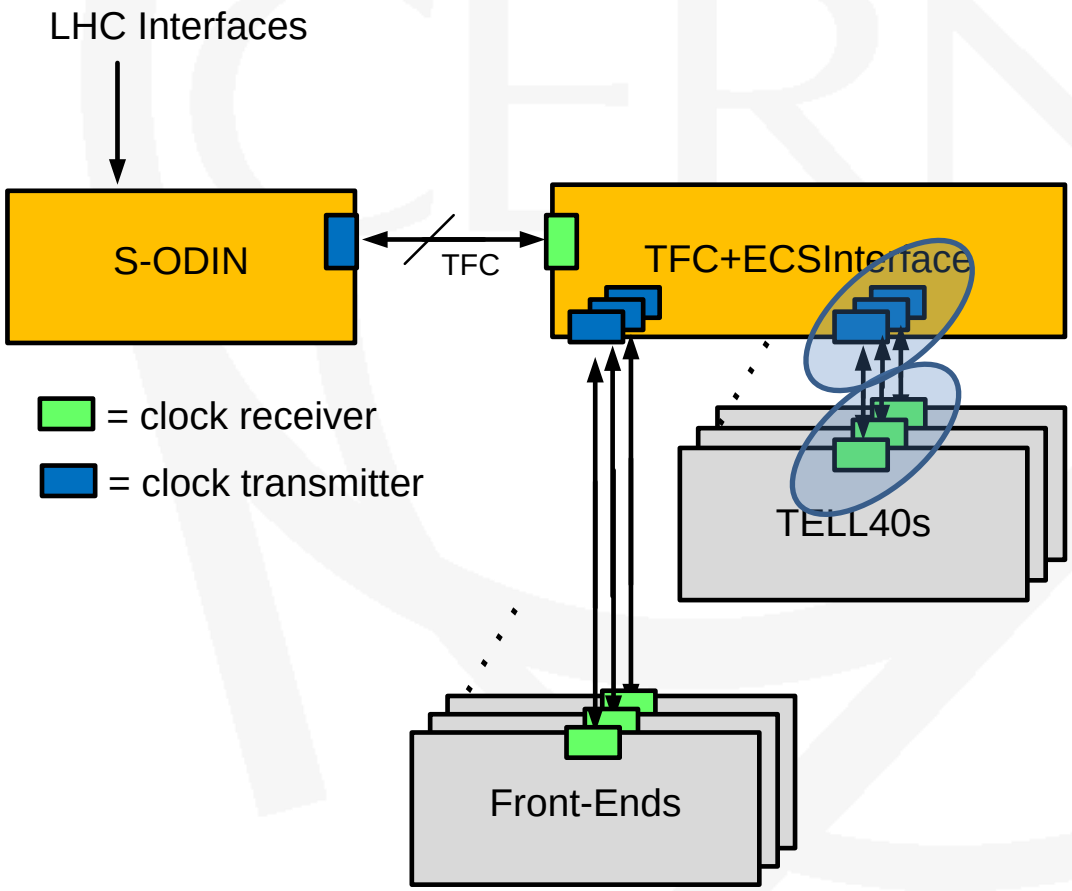
→ Does the job for us

→ control of fine phase + latency at the FE + minimize jitter

→ No problem in ECS monitoring

- Simply decoding GBT protocol in TFC+ECSInterface FPGA
  → No need of fine phase or latency control for ECS.

Clock distribution and phase/latency control



2. ATCA backplane

→ Does the job for us

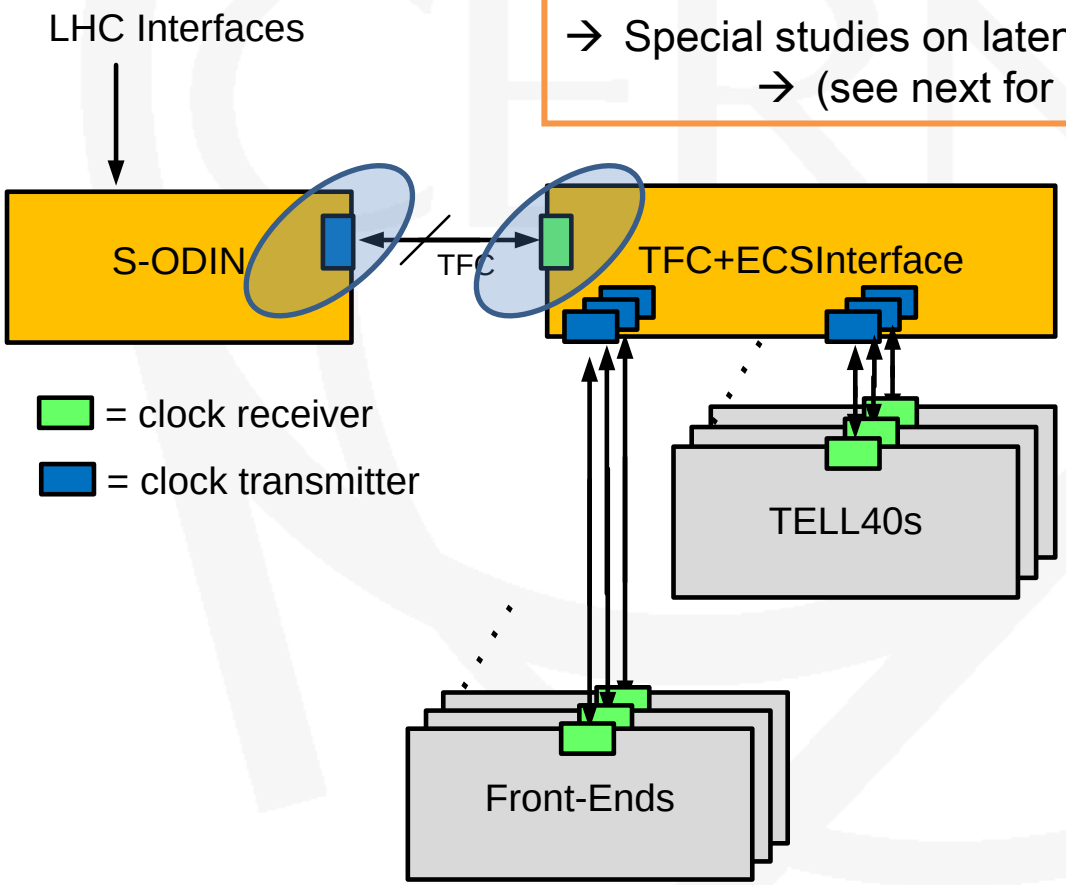
- → control of latency
- → jitter and fine phase less of an issue

- → Effectively is an FPGA-to-FPGA link on backplane dedicated lines
- → To be checked: jitter on backplane!

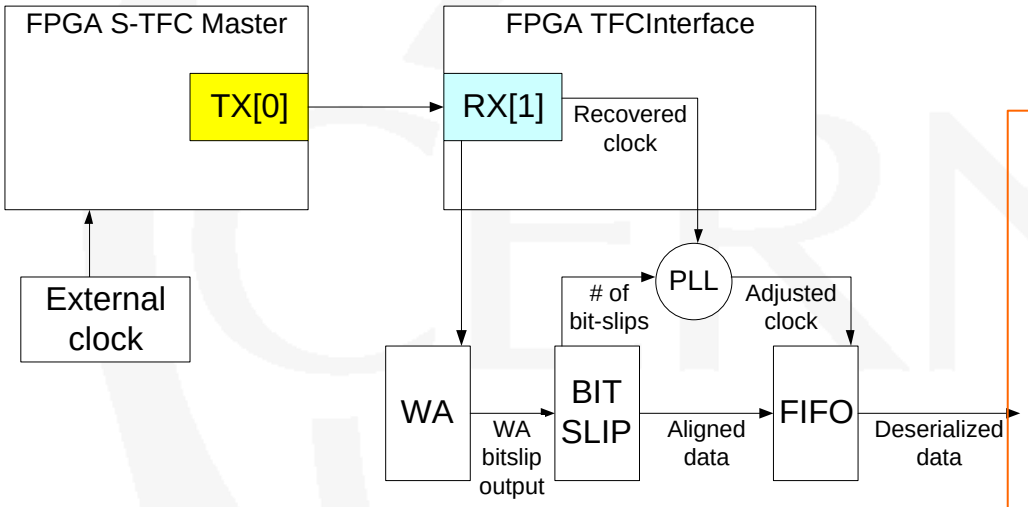
Clock distribution and phase/latency control

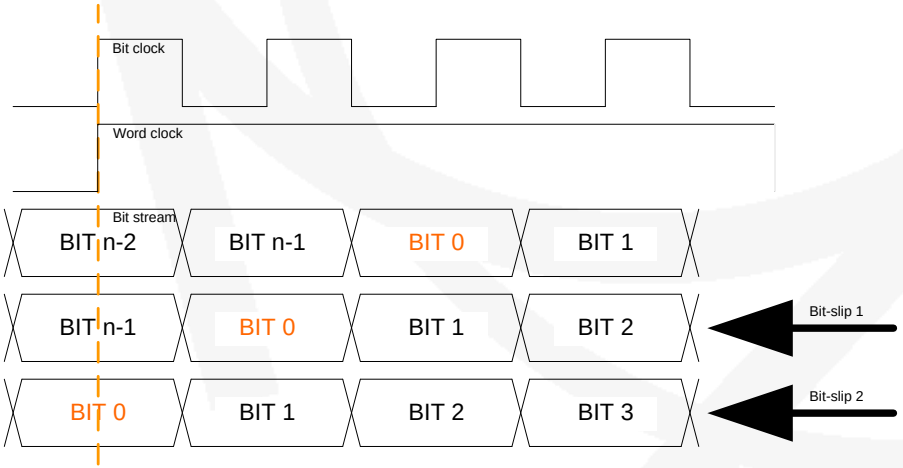
3. FPGA to FPGA transceivers
→ Special studies on latency and phase alignment
→ (see next for preliminary tests)

→ control of fine phase and latency
→ minimize jitter



Latency and phase control





Investigate with ALTERA and Jean-Pierre two years ago

→ Trick is to use the bitslip management already in FPGA
  ✓ Issue: need 8b/10b encoding

→ Set a reference value

→ Bitslip the word by the difference between the measured value and the reference value to re-align the words

First implementation seems to work

→ Stress tests various clock loss/recovering situations

→ Estimation of maximum delay in recovering

Event Destination and Farm Load Control

The current system operates in a powerful mixture of *push and pull protocol* controlled by ODIN :

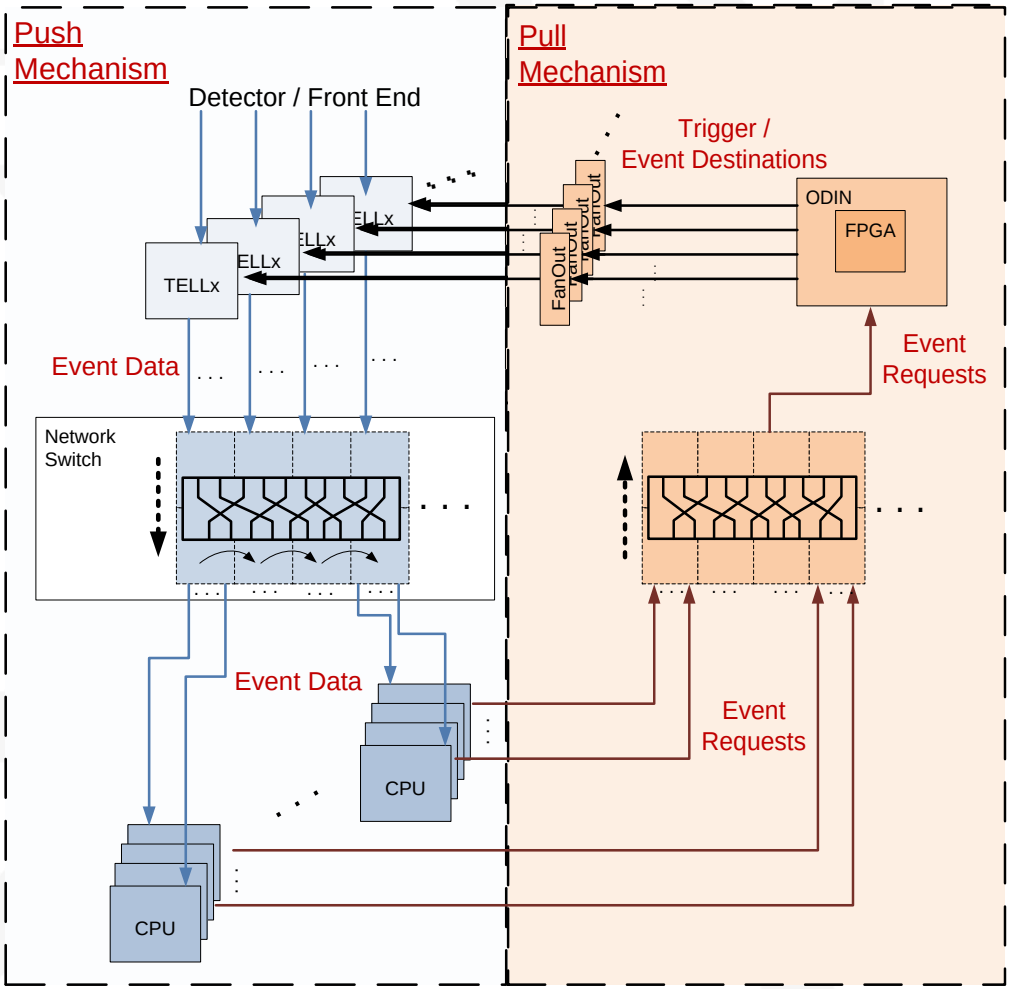
- → Asynchronous pull mechanism
- → "Push" driven by trigger type and destination command

→ Note: LHCb-PUB-2011-023

→ 4 years faultless operation

Similar proposal for upgrade

→ Comments in backup



Event Destination and Farm Load Control

Central FPGA based implementation

- Extreme reliability, flexibility, speed, controllable latencies

→ Central event packing control

- Different trigger types and destination types
- Variable MEP packing factor

→ Dynamic rate regulation as function of farm rate capacity

- Accounts for statistical variations in processing time

→ Dynamic handling of farm nodes in-flight

- Processing blockages, failures, interventions
- All impacts on rate capacity handled automatically
- As soon as nodes are recovered, included automatically in-flight by event request mechanism

→ Minimal event loss and accurate dead-time counting

Contrary to conventional pull scheme, this is robust against event request packet losses

Event Destination and Farm Load Control

Buffer requirement trivia

- Readout boards: ~1.5 MEPs per link
- Network: Some naïve assumptions
  - Rate: 30 MHz
  - MEP packing factor 10 → 3 MHz MEPs and 3 MHz MEP Requests

    → Current ODIN can handle 1.8 MHz of MEP Requests (ODIN <-> FARM is 1 GbE…)
  - Event size 100 kB → 1 MB / MEP
  - Farm nodes 5000 → 600 MEPs/node/s → 1.7ms / MEP
  - Switch subunit sharing resources: 50 links / subunit → 100 subunit

  → 30 kHz of MEPs per switch subunit

  → Every 1.7ms, 50 MEPs to juggle with → <buffer> = O("50 MB")

  → True need of memory depends on statistical variation of HLT processing time and "local farm derandomizer"
- Farm nodes: few MEPs in local derandomizing buffer

In our view, this looks like a straight-forward implementation…

S-ODIN data bank

S-ODIN transmits a data bank for each accepted event in a MEP

→ Run number, event identifier, orbit number, bunch identifier, UTC time, event type, trigger mask, bunch crossing information

+

S-ODIN data bank and LLT data bank is merged

(reminder: LLT is in same board as new S-ODIN)

→ Info about timestamp, trigger type, bxid, trigger decision…

- Mostly like now

→ Will need at least 10GbE connection directly to FARM

- what about 40GbE…? ☺
- has to allow bandwidth partitioning as well
  - In fact «several» 10GbE (n*10GbE…),

→ reduced bank size for local tests

- No LLT for instance

Old TTC system support and running two systems in parallel

We already suggested the idea of an hybrid system:

reminder: L0 electronics relying on TTC protocol

→ part of the system runs with old TTC system

→ part of the system runs with the new architecture

How?

1. Need connection between S-ODIN and ODIN (bidirectional)
   → use dedicated RTM board on S-ODIN ATCA card
2. In an early commissioning phase ODIN is the master, S-ODIN is the slave
   → S-ODIN task would be to distribute new commands to new FE, to new TELL40s, and run processes in parallel to ODIN
   → ODIN tasks are the ones today + S-ODIN controls the upgraded part
   ✓ In this configuration, upgraded slice will run at 40 MHz, but positive triggers will come only at maximum 1.1MHz…
   - Great testbench for development + tests + apprenticeship…
   - Bi-product: improve LHCb physics programme in 2015-2018…
3. In the final system, S-ODIN is the master, ODIN is the slave
   → ODIN task is only to interface the L0 electronics path to S-ODIN and to provide clock resets on old TTC protocol

TFC fits well on xTCA and Marseille hardware

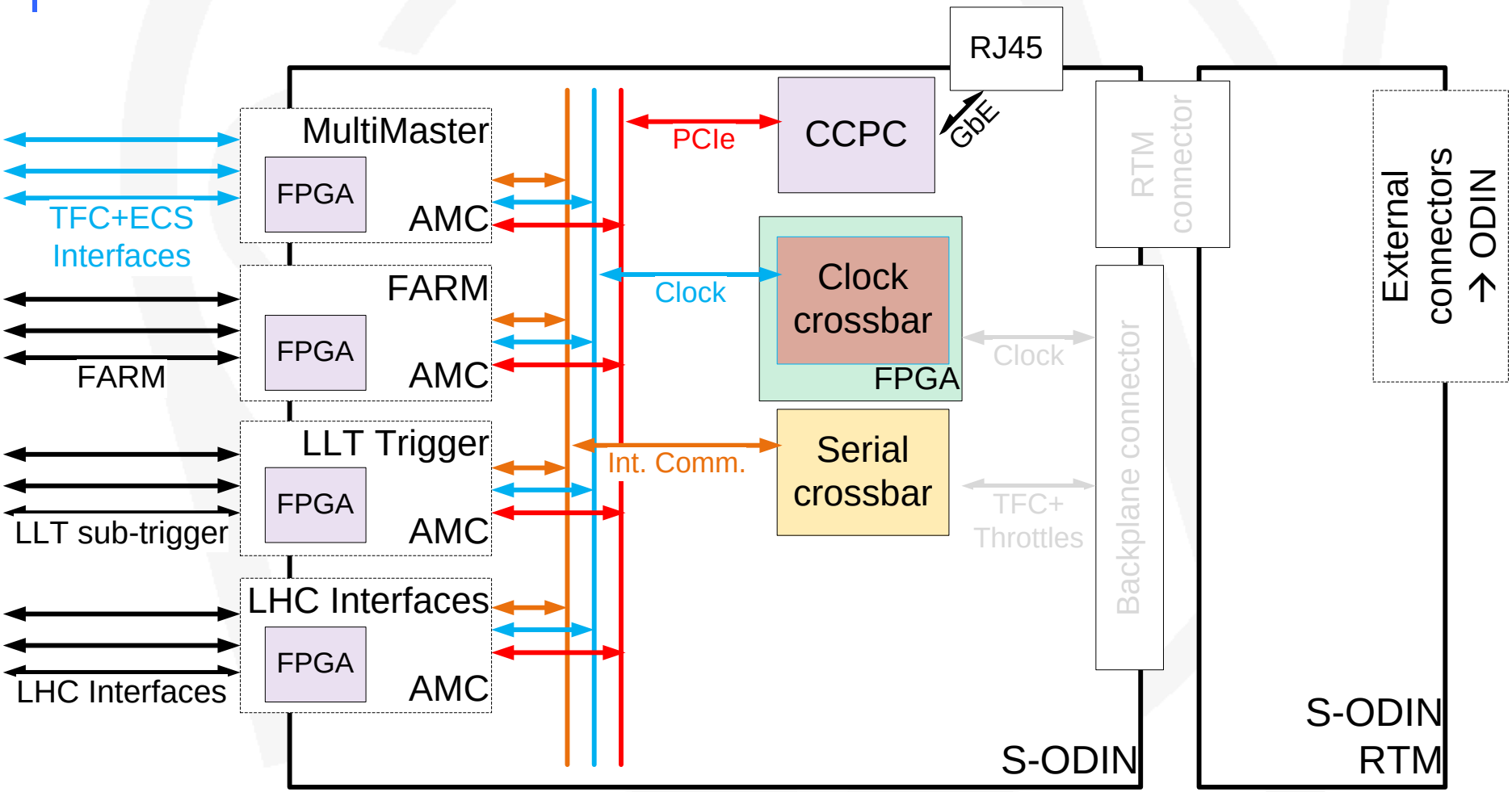
TFC development work at CERN apart from hardware testing and high level control

- ✓ All FPGA code (TFC logic including trigger, link protocols, board control)
- ✓ Special mezzanine(s) (Clock reception, LHC Interfaces, TTC, etc)
- ✓ Relay logic for TFC+FE Interface including optional ODIN logic for stand-alone activities
- ✓ Challenges:
  - TFC+FE interfaces with optional ODIN need to be ready and produced in quantities before any SD test setup → here we might be already late!
  - Validation of clock phase control and reproducibility of latency and jitter
  - Validation of tranmission protocols S-ODIN+TFCInterface and TFCInterface+FE
    - ➢ Try 8b/10b decoder to start with
      - ❑ Simple, as part of the ALTERA decoding
    - ➢ Validate GBT with TFC+ECS to FE information

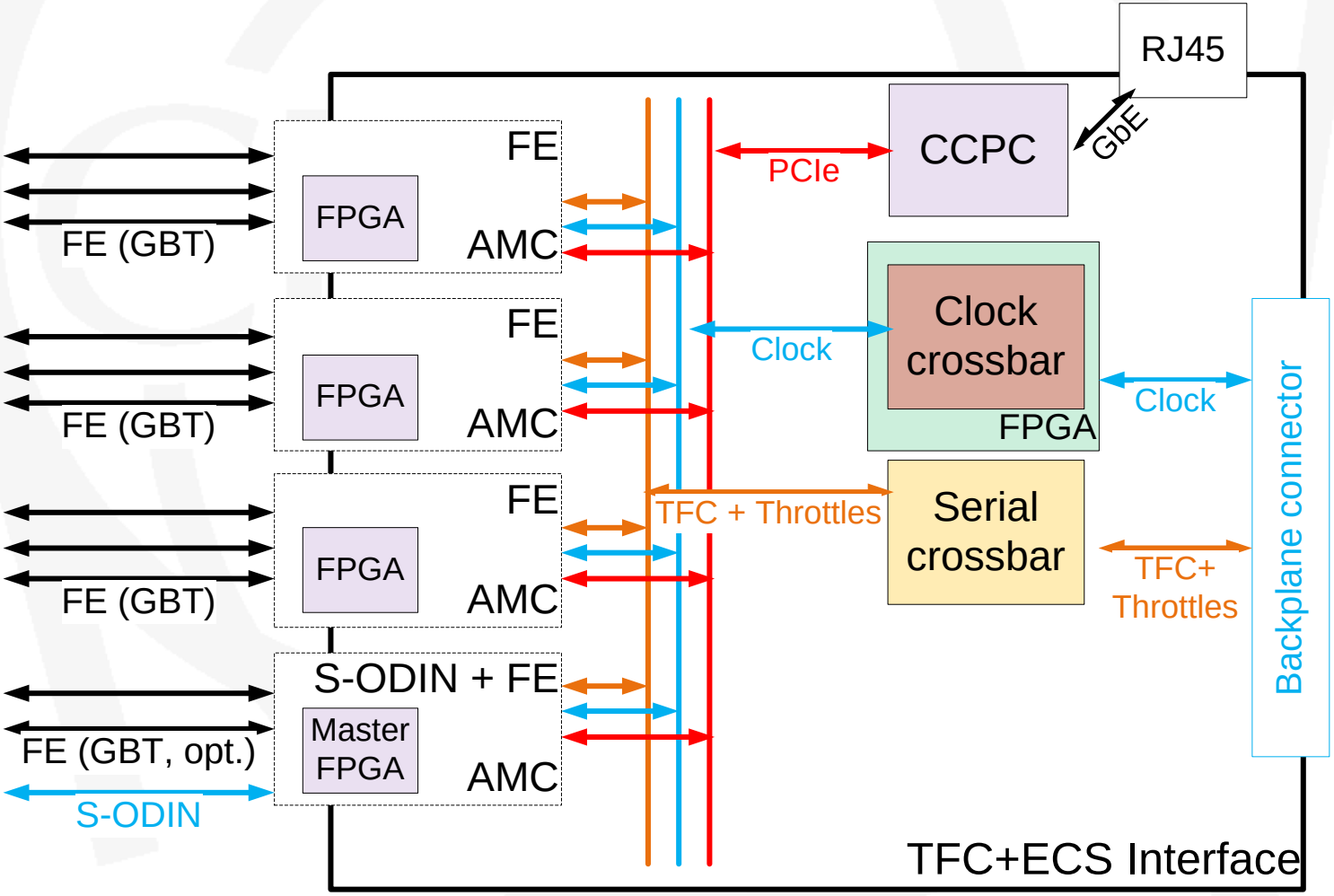
Considerations

- ✓ In ATCA Dual Star technology, boards in Hub1 and Hub2 talks to all the boards in the crate
  - TFC Interface as Hub1 board
- ✓ S-ODIN located near the LHC interfaces (clock, machine timing and control information).
  - Bi-directional fibres connections to TFC Interface
- ✓ If no backplane, BE connection via TFC Interface must be done via fibres
  - How many receivers/transmitter per mezzanine/ how many mezzanines?

S-ODIN on Marseille's ATCA board



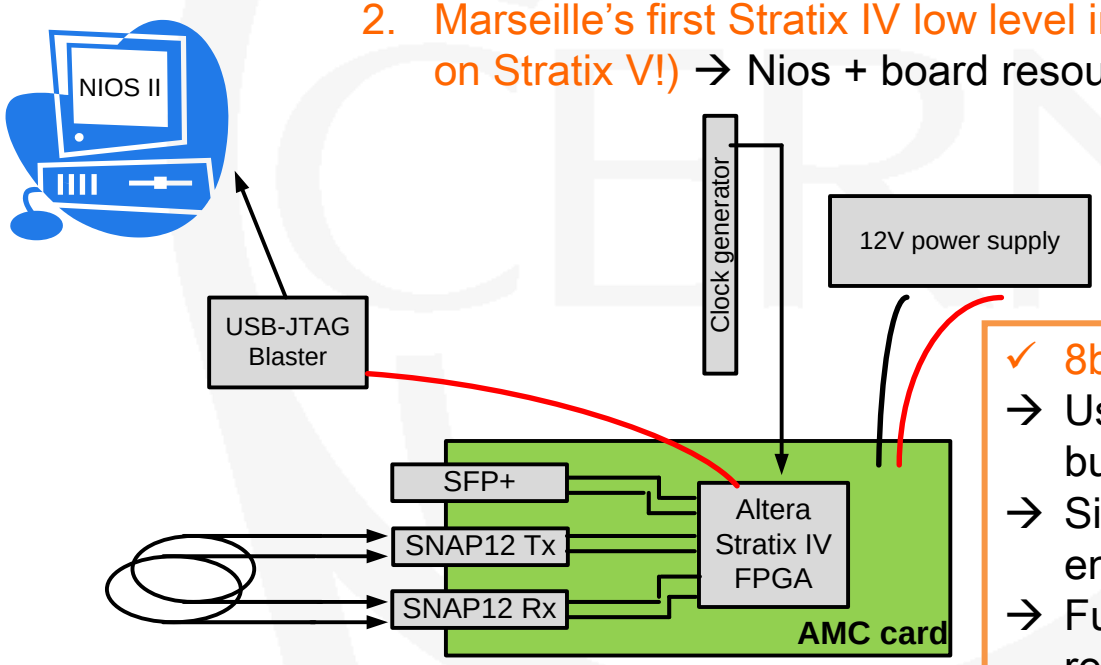
TFC+ECSInterface on Marseille's ATCA board



Clock distribution and phase/latency control

First preliminary tests on phase/latency control using:

1. Marseille's first AMC prototype with ALTERA Stratix IV
2. Marseille's first Stratix IV low level interfaces (will need re-validation on Stratix V!) → Nios + board resources interfaces + thanks to Marseille team!



✓ 8b/10b protocol: no problem
→ Using «word alignment» from Altera GX buffer + «deterministic latency»
→ Simply add Ctrl word for the 8b/10b encoder: 2bits more
→ Full reproducibility upon power-up and resets and reconfiguration

✓ FPGA-to-FPGA GBT protocol: ok, but needs special frame alignment
→ No deterministic latency if no special words are sent!
→ Needs a special word (10 bits minimum) at power-up/after reset/after reconfiguration for the GX buffer to realign to the beginning of the frame + «deterministic latency»
- First preliminary tests were ok, ***but needs more validation under stress tests!***

Single AMC test stand for sub-detectors

If needed, early first sub-detectors test stands could use a single AMC board

→ single Stratix IV FPGA, 12 links in/out @ 4.8 Gb/s + 1 bidir 8 Gb/s link (SFP+)

→ see Jean-Pierre's presentation

Need first firmware version of TFC+ECS Interface

→ including main S-ODIN functionalities

- triggers, resets, commands, throttle, destination assignments, etc…

Need first firmware version of TELL40s

→ decoding and packing to a computing node

→ this would allow first tests on buffer occupancies

Need first software version of control software (for simply controlling the hardware)

→ this have to be done in NIOS as there is no space for CCPC on an AMC

With this, you can test any FE chip and test that it is compliant with specs…

→ Implement many many counters in the FE, we need them!

Would need a production of O(10) AMCs needed

For the future test stands, a full ATCA board with all functionalities will be available

→ see Jean-Pierre presentation

Timeline + work organization + costs

First version of a single TFC+ECSInterface AMC board for test stands around March-April 2012

It is mostly the time dedicated to

- → compile a working first firmware version
- → test/simulate functionalities
- → learn to control the board

Work will be organized as usual between me and Richard

- → all development done at CERN
  - Richard's experience with the development of the current TFC system is an invaluable add-on
  - Our experience with the running of the current TFC system is another invaluable add-on
- → will provide entirely:
  - → firmware for S-ODIN and TFC+ECS Interface
  - → firmware core for TELL40 to strip out TFC information (only a core!)

Costs:

- → Not much as we are using the Marseille's hardware
- → Possible expenses:
  - A uTCA/xTCA crate to test TFC functionalities with this technology
    They come with everything inside: power, controller cards, Ethernet.. no additional costs on this.
  - Fibres and fibres breakouts
- → Not more than 10/15kCHF

Lab work and planned (first) tests

Some lab work has already started:

1. Set up a S-TFC test stand at the pit ✔ Done!
2. Get Marseille's AMC board first prototype ✔ Done!
   - First version of firmware with GBT protocol in it (developed in Marseille)
   - First version of control software in NIOS (developed in Marseille)
     - One word of acknowledgment for the work done by Jean-Pierre and Frederic Hachon in Marseille
3. Get acquainted with the hardware ✔ Done!
4. Send first trigger words ✔ Done!
   - 8b/10b protocol: using ALTERA features
   - GBT protocol: not possible to use ALTERA features
5. Stress tests to validate latency control
6. Stress tests to validate phase control ✔ Done (but will need revalidation on Stratix V + more tests!)
7. Implement first version of TFC+ECS firmware with functionalities presented here
   - Including first simple version of S-ODIN
8. Develop first single-AMC test stand board with firmware + software to control the hardware
   1. Need dedicated simulation framework
      → Already developed in 2009 (see https://indico.cern.ch/conferenceDisplay.py?confId=65307)
      → Needs mostly adaptations …
   - Work to interface the single AMC card to PVSS (we have a Summer Student for that!)