From resistor to experiment

Design with emphasis on commissioning, operation, maintenance and evolution

R. Jacobsson

Introduction

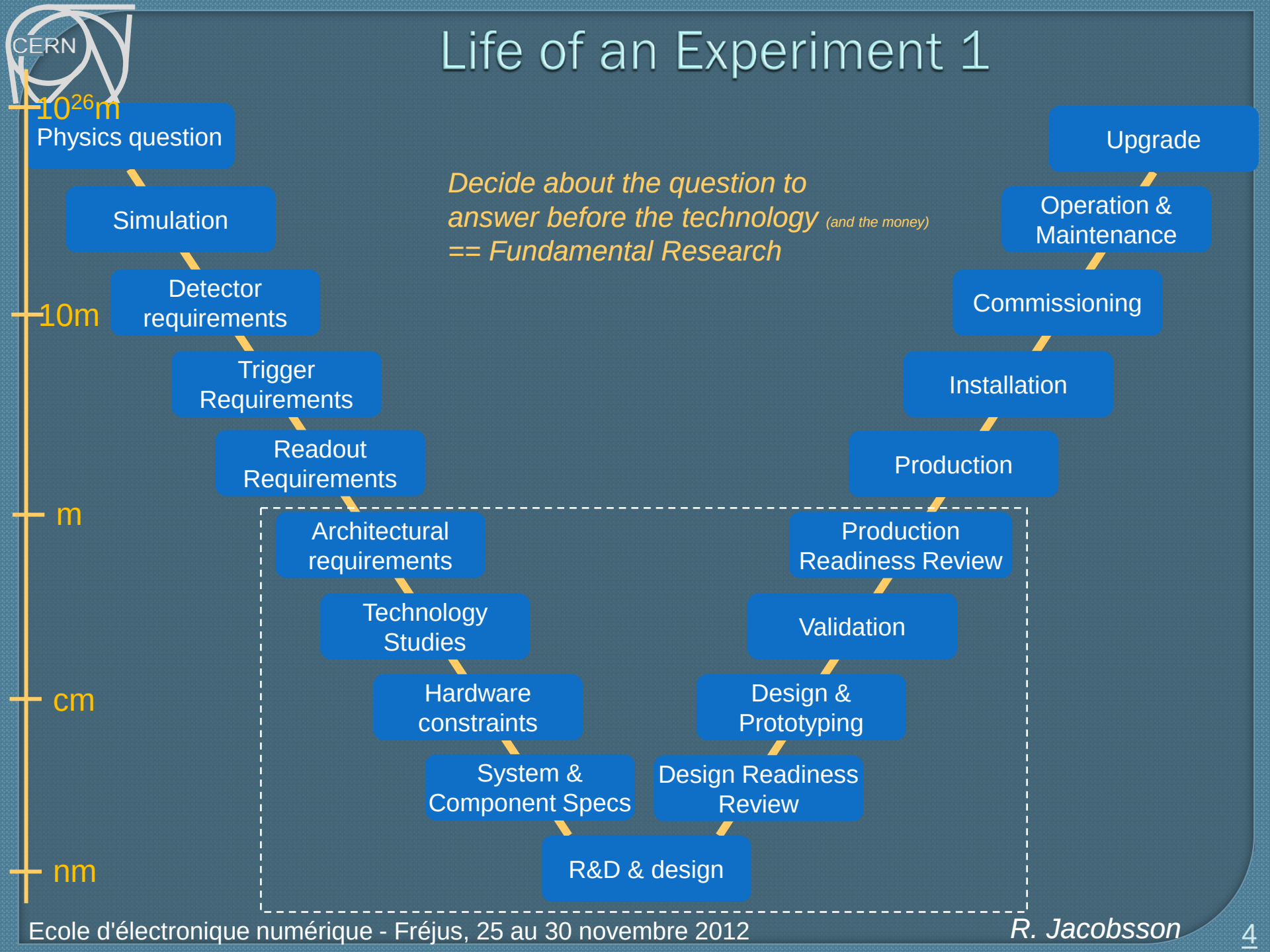
- Personal view – 20 year of experience with DELPHI & LHCb
  - Physicist view….
  - Unavoidably a bias in the examples and in the opinions
  - First three years of LHCb operation + two shifters and 96% operational efficiency
- From physics idea to PCB and back
- How the global specifications determine the choice and design of each component
- A long list of global requirements/concepts in order to commissioning and operate and maintain detector which has to be kept in mind when developing even the lowest chip
  - Common traps and missed points
- Beam time is expensive + competition is tough
- Collection of concepts to keep in mind

Contents

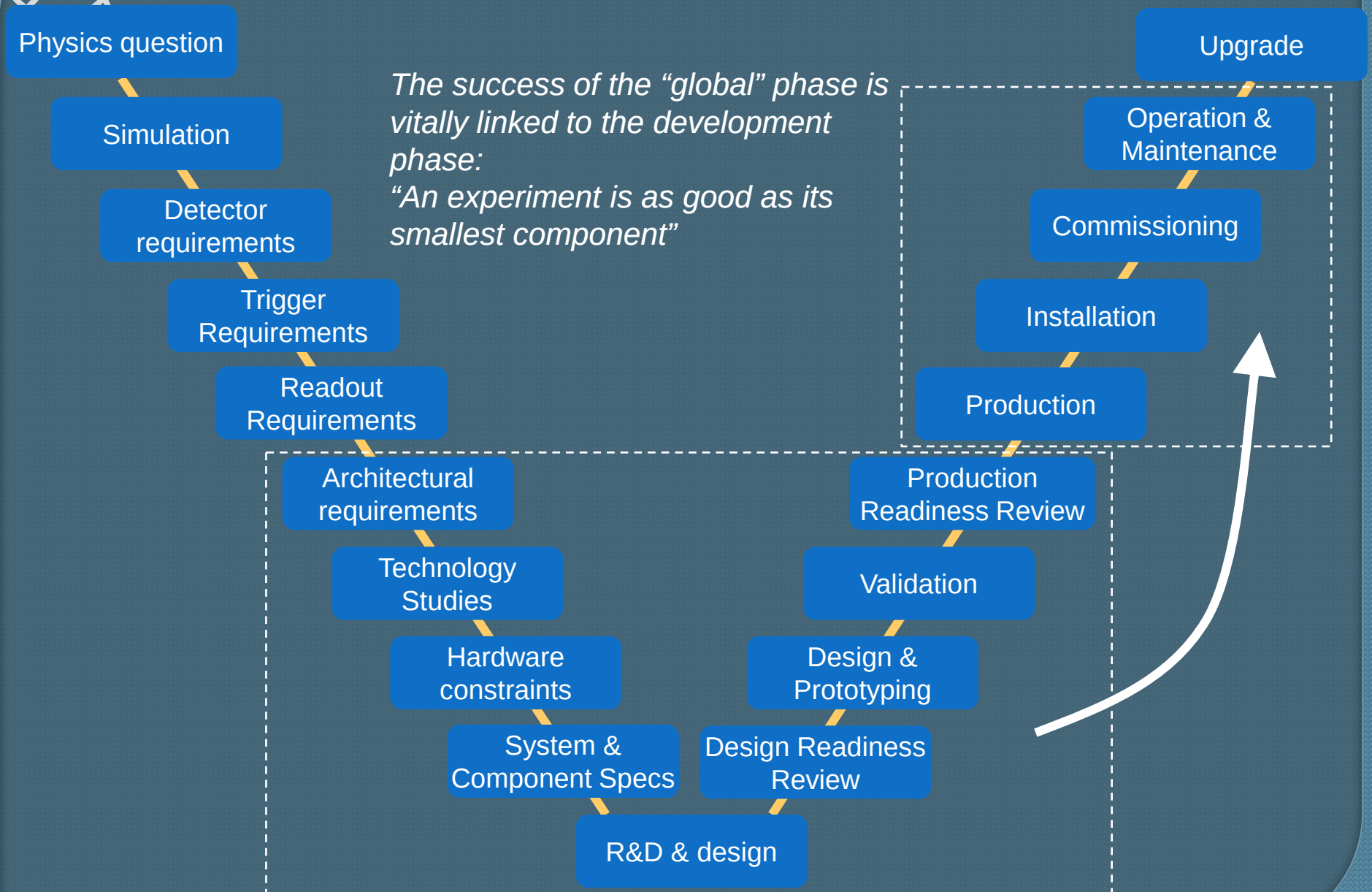
1. Introduction
   - Setting the scope
   - Rates at collider experiments
2. Basic concepts
   - Multi-level trigger systems and readout structures
3. Front-end electronics
   - General architecture
   - Digitizers
   - Signal processing
4. Trigger principles and implementations
   - Setting the scope
   - Trigger basics
   - Trigger design and performance
   - Fast and synchronous hardware triggers
   - Software triggers
5. Experiment timing
   - Timing and fast control systems
6. Event readout
   - Readout networks (buses, switches, etc)
   - Event building and online farm processing (event filters)
   - Data storage
7. Configuration, control and monitoring
   - Operating modes
   - Run control
   - Data monitoring and quality control
8. Conclusions

- Data Acquisition Systems
  - Front-end Readout
  - Event Building
- Run Control
  - Tools and Architecture

Life of an Experiment 1

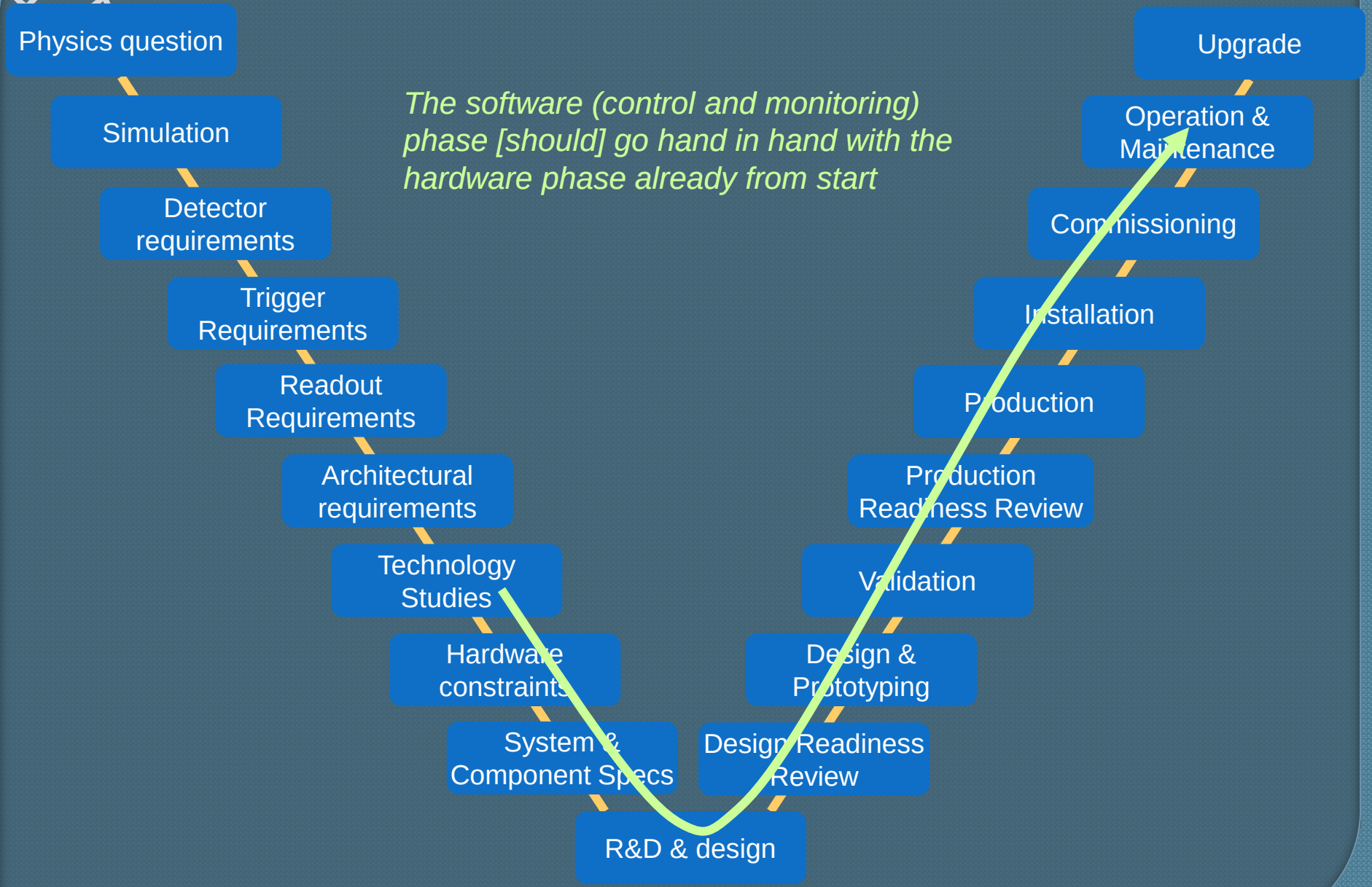


Life of an Experiment 2



The success of the "global" phase is vitally linked to the development phase:
"An experiment is as good as its smallest component"

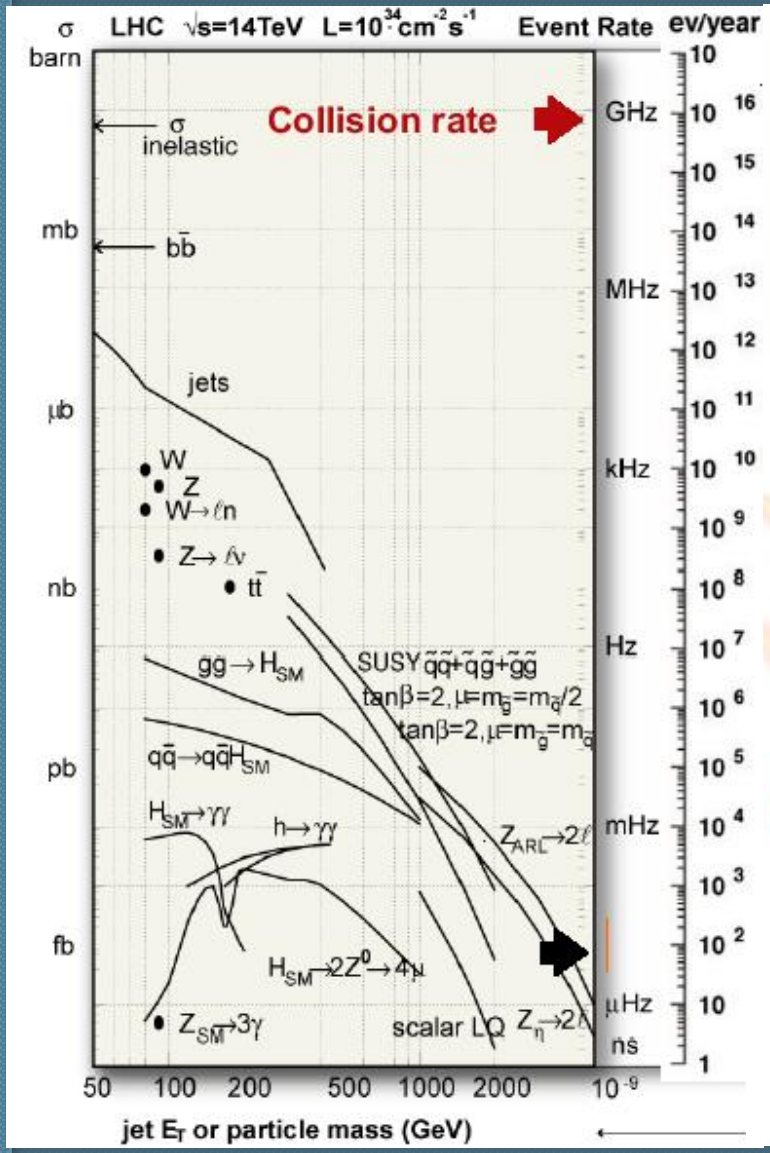
Life of an Experiment 3

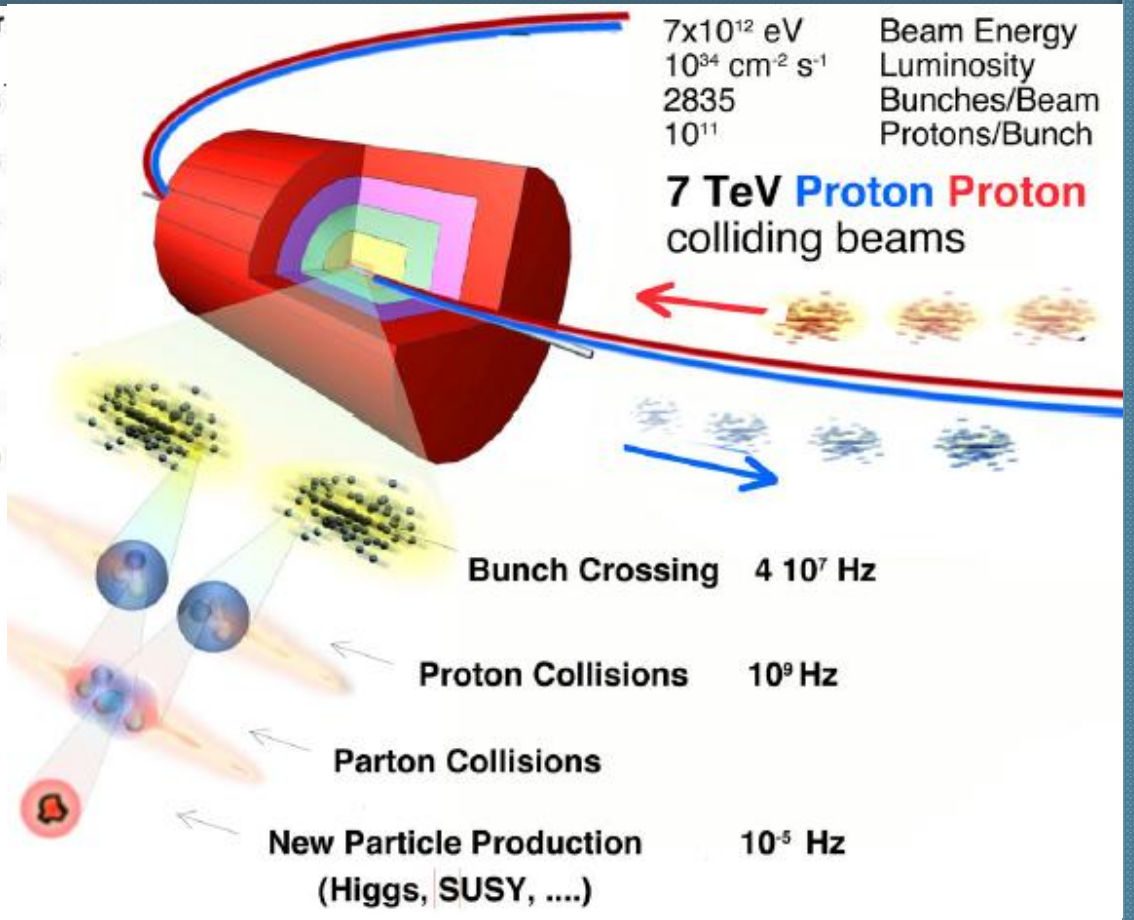


The software (control and monitoring) phase [should] go hand in hand with the hardware phase already from start

Experimental Challenge

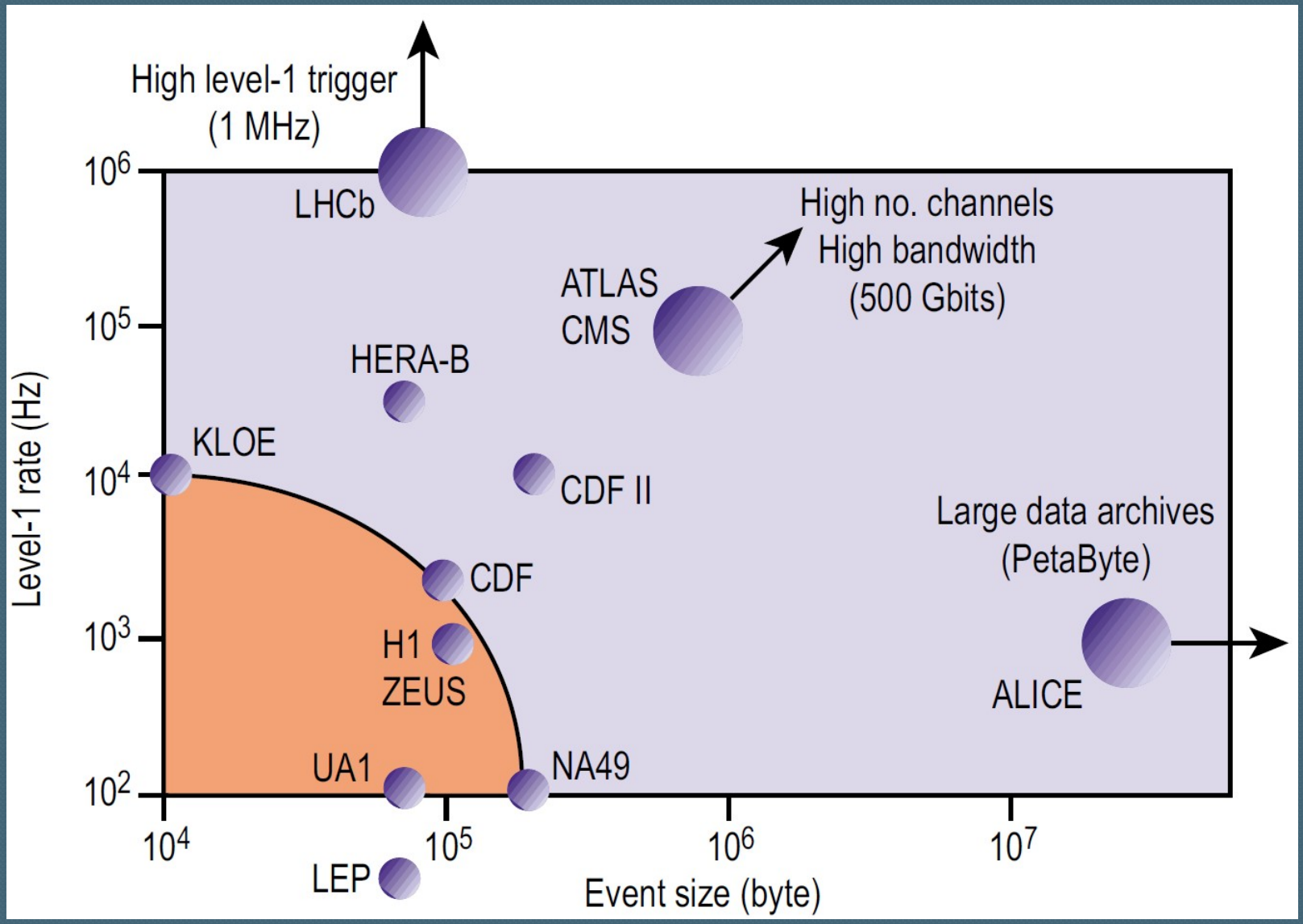
- Physics → Detector → Trigger → Readout





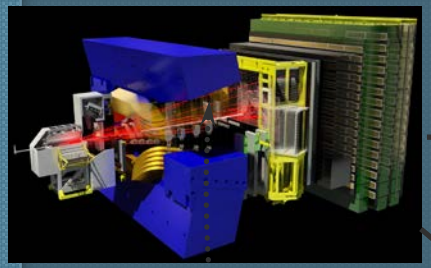
Event rate : ~ 10^9 Hz
Event selection : ~ $1/10^{13}$

Data Taking – Past and Now

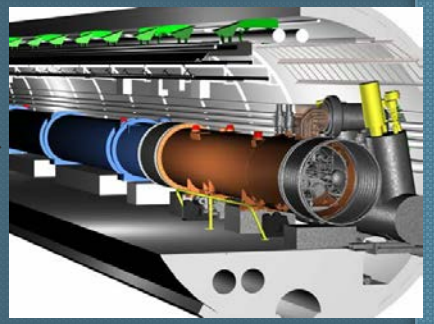


Principal blocks of an experiment

Detector



Accelerator



Experiment Control System

Trigger system

Signals

Decisions

Clock

Info

Readout control system

Control

Data acquisition system

Raw data

Mass storage

Detector Safety System

Feedback

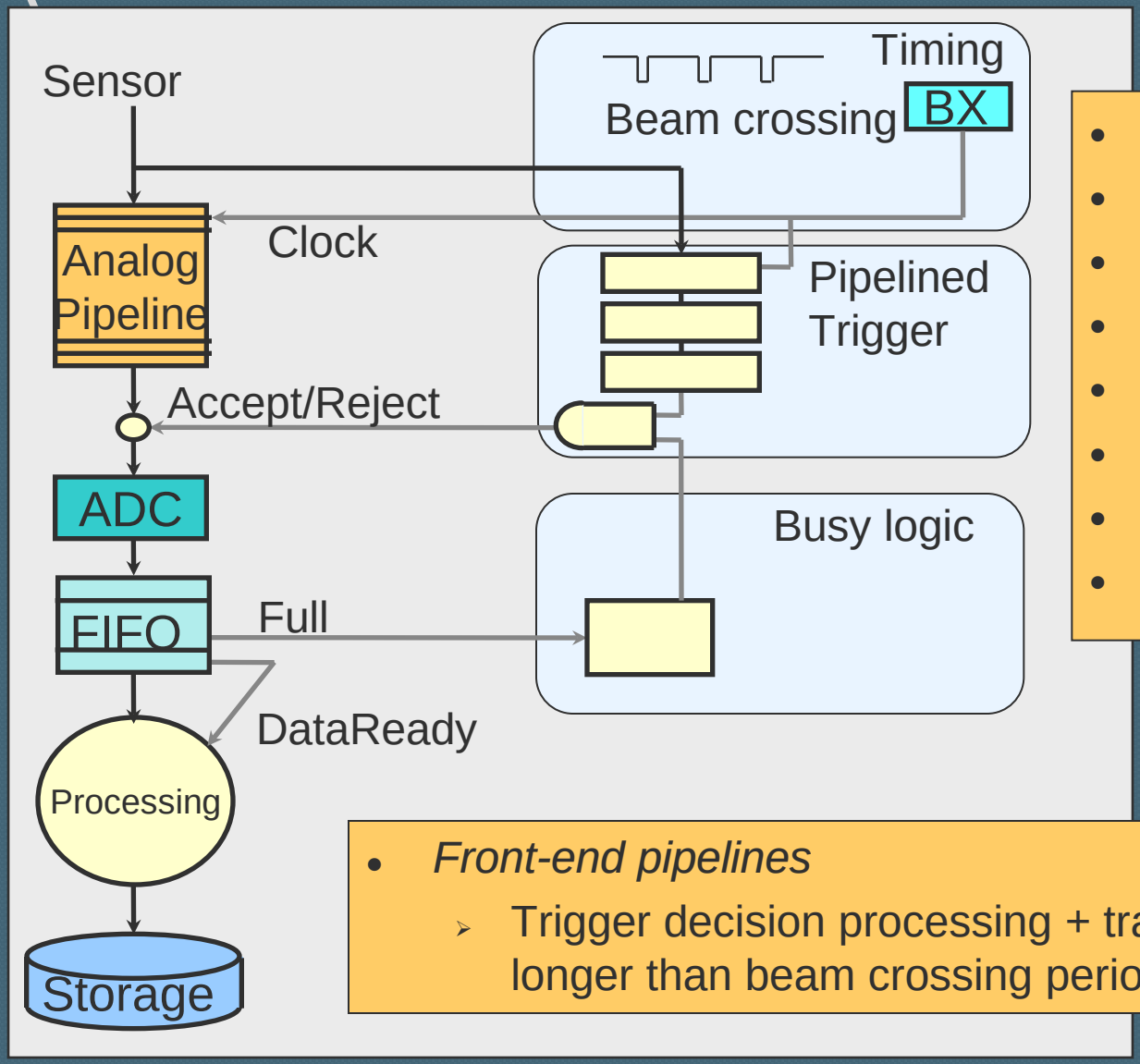
Detector & trigger simulation

Reconstruction & analysis

Physics results

Experiment Control System = HV, LV, motion, readout configuration, data taking control, monitoring, etc

Trivial DAQ system at LHC



- *Timing*
- *Trigger*
- *Front-end pipelines*
- *Derandomizer buffers*
- *Busy logic*
- *Deadtime/efficiency*
- *Parallel processing*
- *(Event building)*

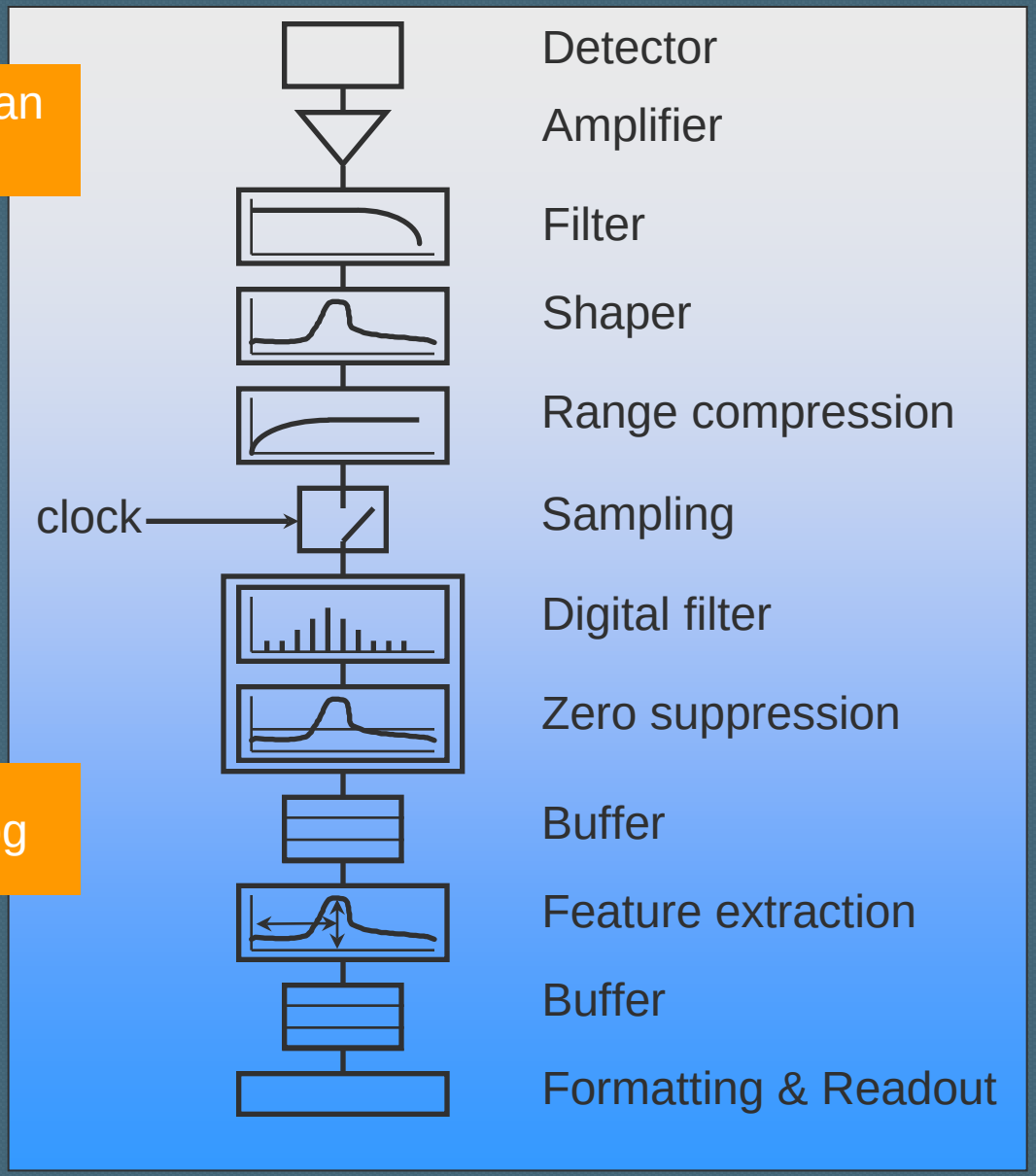
"Real time"….

- *Front-end pipelines*
  - Trigger decision processing + transmission delays are longer than beam crossing period

General readout processing flow

Clock distribution in an experiment is critical

Lim (Digital) $f_{clock} \rightarrow \infty$ = Analog



FE strategy

- Precise Front-End strategy is defined by the detector technology
- A wide variety of technologies for four purposes:
  - Tracking & momentum measurement
  - Identification
  - Energy measurement
- Analog readout / digital readout / binary readout
- ADC / TDC

Requirements

- Which are the challenges at each level of the readout system?

➔ Defines to a large extent the organization, technology, design of components and links

- Radiation
- Mechanical stress (vibrations and cables and busses)
- Magnetic field
- Distance/spread
- Size and material budget
- Accessibility
- EMI (noise)
- EMI immunity
- Power
- Heat dissipation
- Cooling
- Flexibility
- Clock quality
- Buffering
- Bandwidth
- Operational logic
- Computational performance
- Control and monitoring requirements
- Reliability
- Availability / fault tolerance
- Cost
- Redundancy
- Autonomy
- Partitioning

Requirements – Front End

- Which are the challenges at each level of the system?
- ➔ Defines to a large extent the organization, technology, design of components and links
  - Radiation / SEU immunity
  - Mechanical stress (vibrations and cables and busses)
  - Magnetic field
  - Distance/spread
  - Accessibility
  - Size and material budget
  - EMI (noise)
  - EMI immunity
  - Power
  - Heat dissipation
  - Cooling
  - Flexibility
  - Clock quality
  - Buffering
  - Bandwidth
  - Operational logic
  - Computational performance
  - Control and monitoring requirements
  - Reliability
  - Availability / fault tolerance
  - Cost
  - Redundancy
  - Autonomy
  - Partitioning

FE Challenges at the LHC

Example of vertex detector FE:
➔ Custom integrated circuits essential for vertex detectors in HEP.

- Requirements
  - 1. low mass to reduce scattering
  - 2. low noise
  - 3. fast response
  - 4. low power
  - 5. radiation tolerance
  - Powering is typically located *far* from FE, voltage drop, high current etc…

- Conflicts and compromises
  - reduction in mass → thin detector
  - radiation tolerance → thin detector
  - thin detector → less signal → lower noise required
  - lower noise → increased power
  - fast response → increased power
  - increased power → more mass in cabling + cooling
  - immunity to external pickup → shielding → mass
  - + contain costs

Requirements – FE data links

- Which are the challenges at each level of the system?
- ➔ Defines to a large extent the organization, technology, design of components and links
  - Radiation
  - Mechanical stress (vibrations and cables and busses)
  - Magnetic field
  - Distance/spread
  - Accessibility
  - Size and material budget
  - EMI (noise)
  - EMI immunity
  - Power
  - Heat dissipation
  - Cooling
  - Flexibility
  - Clock quality
  - Buffering
  - Bandwidth
  - Operational logic
  - Computational performance
  - Control and monitoring requirements
  - Reliability
  - Availability
  - Cost
  - Redundancy
  - Autonomy
  - Partitioning

Requirements – Trigger

- Which are the challenges at each level of the system?

➔ Defines to a large extent the organization, technology, design of components and links

- Radiation
- Mechanical stress (vibrations and cables and busses)
- Magnetic field
- Distance/spread
- Accessibility
- Size and material budget
- EMI (noise)
- EMI immunity
- Power
- Heat dissipation
- Cooling
- Flexibility
- Clock quality
- Buffering
- Bandwidth
- Operational logic
- Computational performance
- Control and monitoring requirements
- Reliability
- Availability
- Cost
- Redundancy
- Autonomy
- Partitioning

Requirements – Readout control

- Which are the challenges at each level of the system?

➔ Defines to a large extent the organization, technology, design of components and links

- Radiation
- Mechanical stress (vibrations and cables and busses)
- Magnetic field
- Distance/spread
- Accessibility
- Size and material budget
- EMI (noise)
- EMI immunity
- Power
- Heat dissipation
- Cooling
- Flexibility
- Clock quality
- Buffering
- Bandwidth
- Operational logic
- Computational performance
- Control and monitoring requirements
- Reliability
- Availability
- Cost
- Redundancy
- Autonomy
- Partitioning

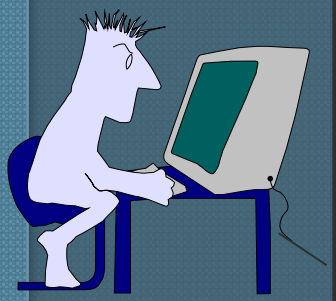
Requirements – Detector safety

- Which are the challenges at each level of the system?

➔ Defines to a large extent the organization, technology, design of components and links

- Radiation
- Mechanical stress (vibrations and cables and busses)
- Magnetic field
- Distance/spread
- Accessibility
- Size and material budget
- EMI (noise)
- EMI immunity
- Power
- Heat dissipation
- Cooling
- Flexibility
- Clock quality
- Buffering
- Bandwidth
- Operational logic
- Computational performance
- Control and monitoring requirements
- Reliability
- Availability
- Cost
- Redundancy
- Autonomy
- Partitioning

Trivial organization of large readout system



Detector sense wires

Detector Channels (FE)

Control and Monitoring

Trigger

Timing and Readout Control

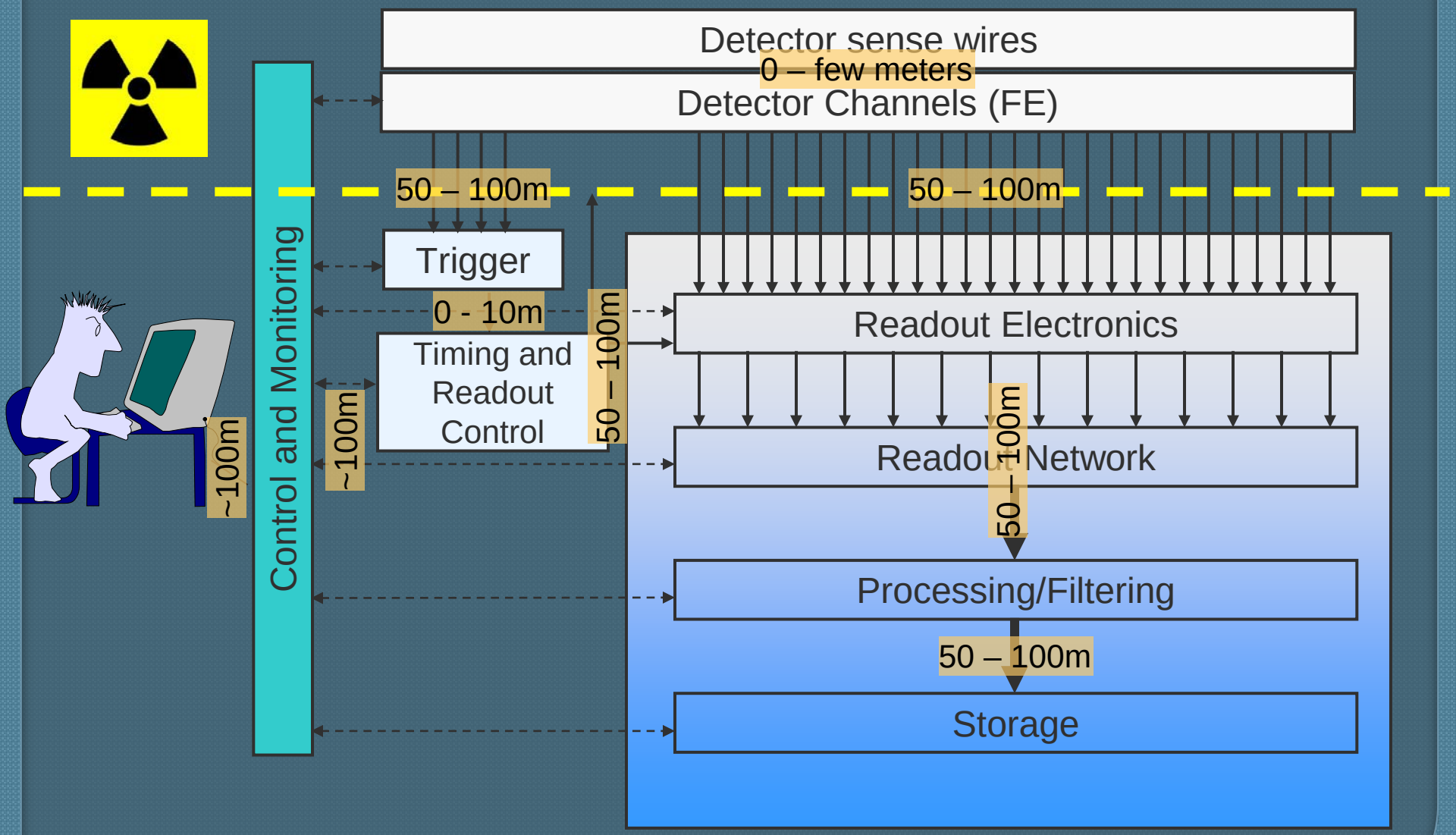
Readout Electronics

Readout Network

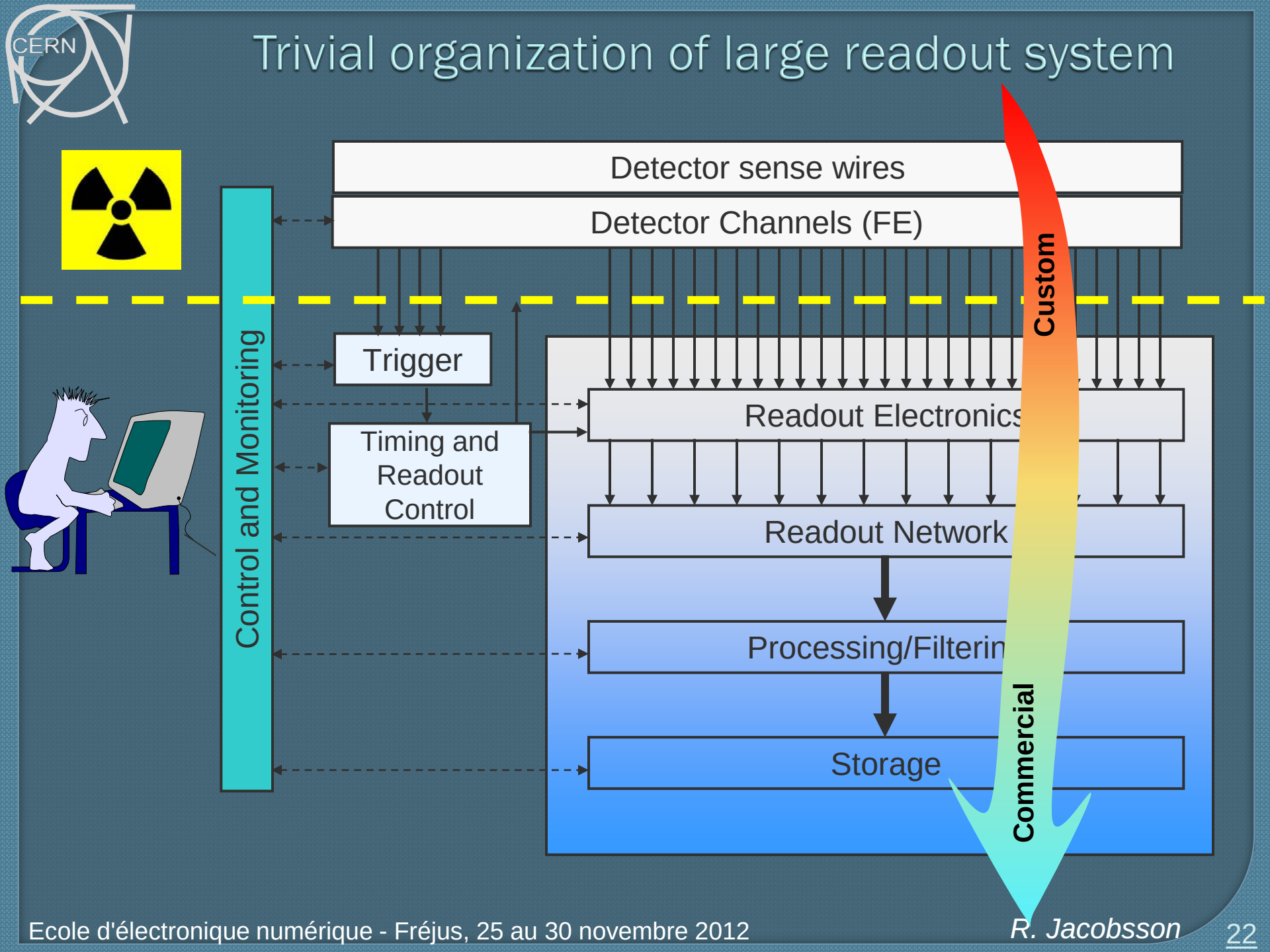
Processing/Filtering

Storage

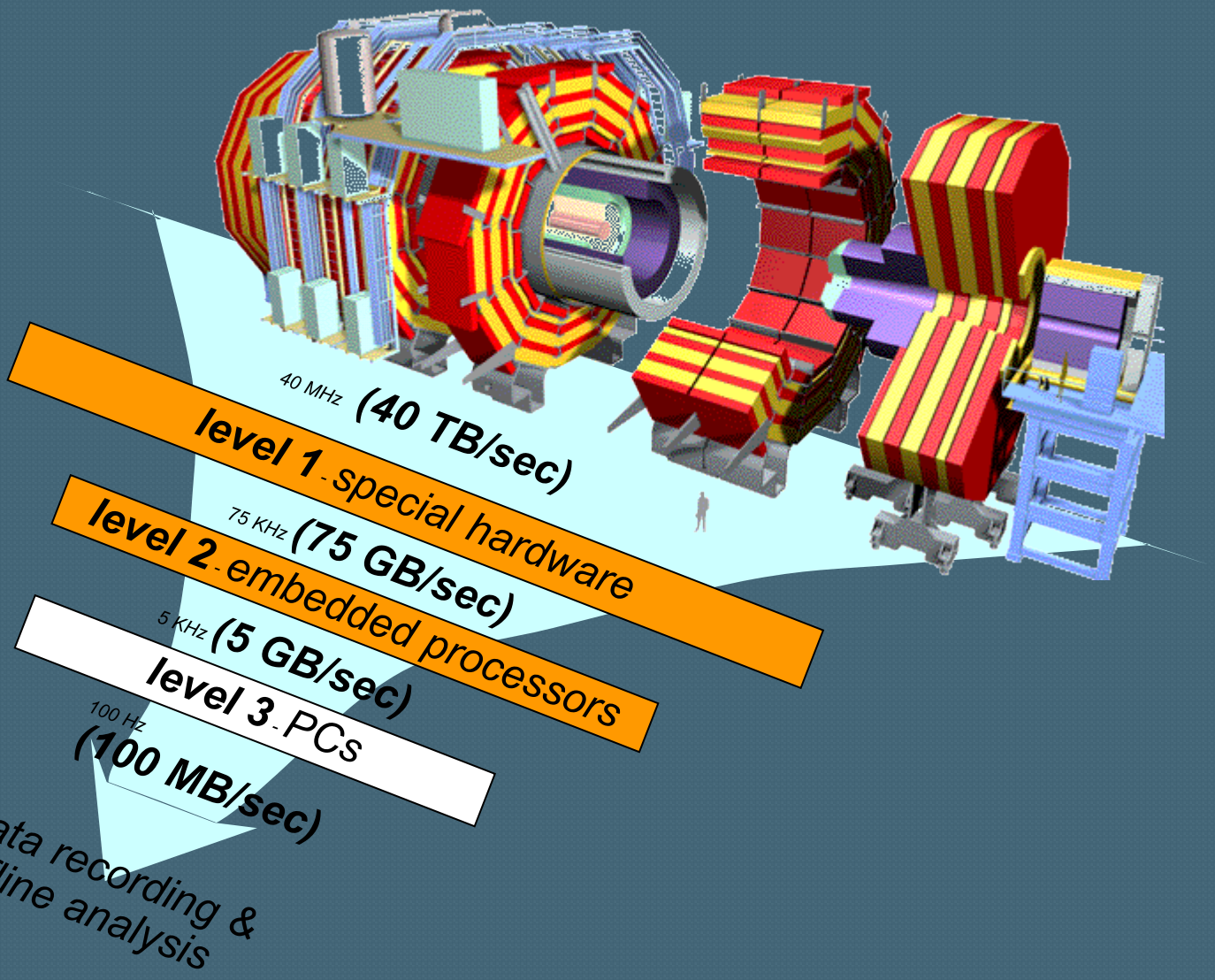
Trivial organization of large readout system



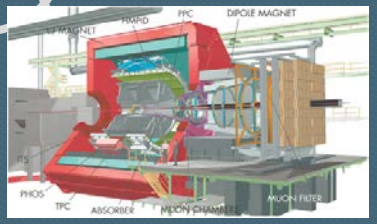
Trivial organization of large readout system



Readout Bandwidths

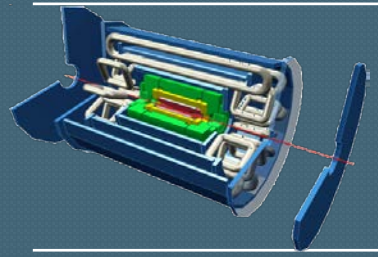


Readout links at LHC



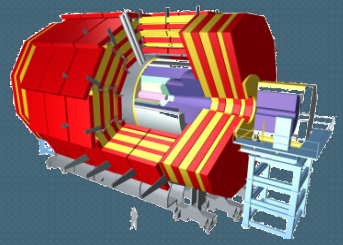
DDL

Optical 200 MB/s ≈ 400 links
Full duplex: Controls FE (commands, Pedestals, Calibration data)
Receiver card interfaces to PC



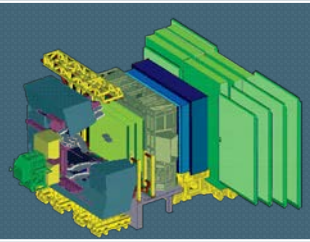
SLINK

Optical: 160 MB/s ≈ 1600 Links
Receiver card interfaces to PC.



SLINK 64

LVDS: 200 MB/s (max. 15m) ≈ 500 links
Peak throughput 400 MB/s to absorb fluctuations
Receiver card interfaces to commercial NIC (Myrinet)

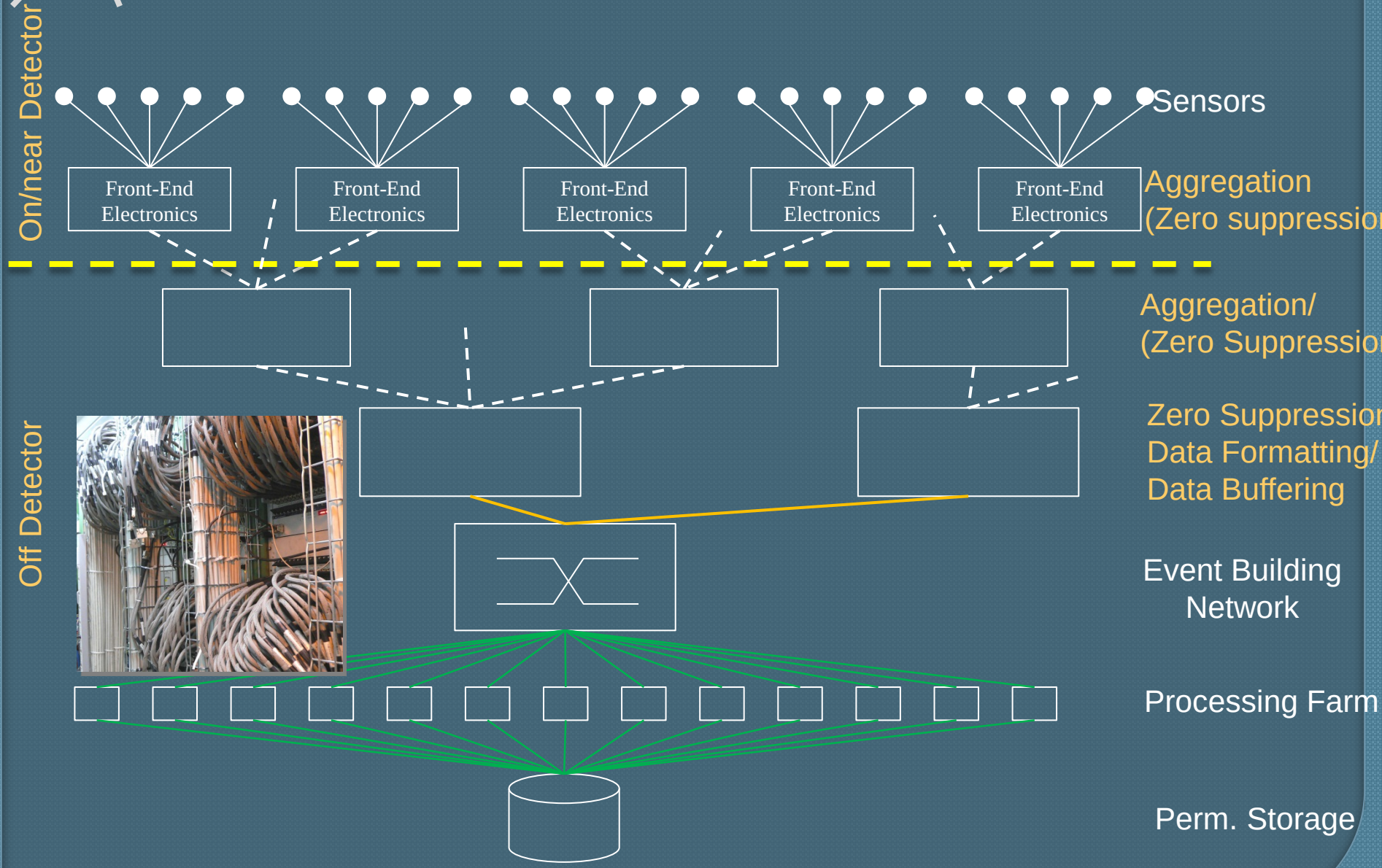


Glink (GOL)

Optical 200 MB/s ≈ 400 links
Receiver card interfaces to custom-built Ethernet NIC (4 x 1 Gbit/s over copper)

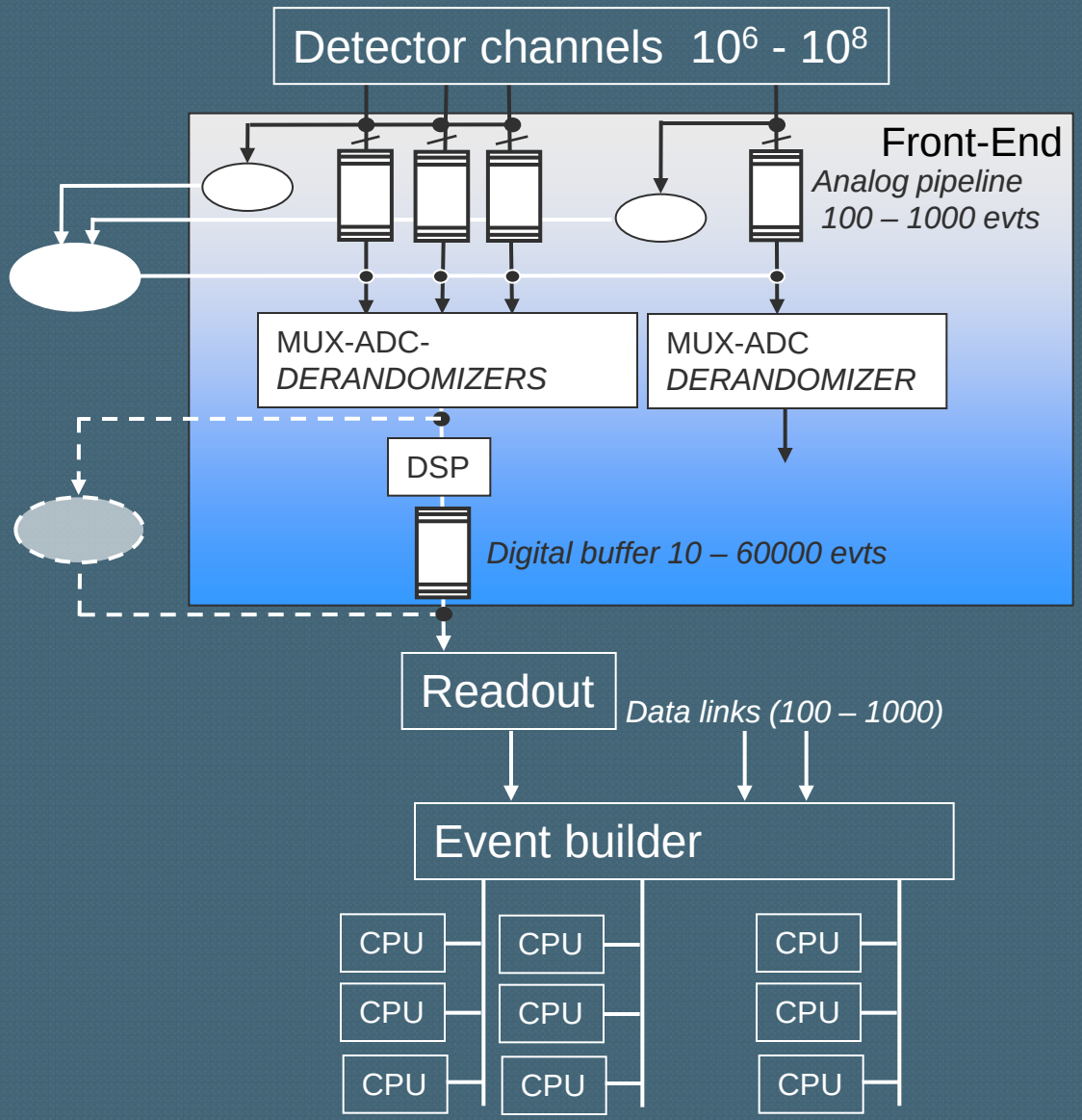
- Links between Front End – Readout Boards
  - Custom based (radiation)
- Links between the custom hardware (ROB) and commercial hardware
  - Commercial large scale networks for the event building and interface to processing farm

Readout Strategy – Data aggregation



R. Jacobsson

Readout System Parameters



L1 trigger
(1-10 μs)

L2 trigger
(10 μs - ms)

L2/L3 trigger
(Event filter)

Rate

40 MHz

100 kHz – 1 MHz

1 - 40 kHz
~ 1-100 Gbytes/s

100 Hz
~ 1- 100 Mbytes/s

Timing and synchronization

- *LHC*
  - *ATLAS*
  - *CMS*
  - *LHCb*
  - *ALICE (Pb + Pb)*

10^6 – 10^8 electronics channels

LEP : e^+e^- crossing rate 45 kHz

22 μs

SPS : $p\bar{p}$ crossing rate 260 kHz

3.8 μs

Tevatron : $p\bar{p}$ crossing rate 2.5 MHz

396 ns

HERA : ep crossing rate 10.4 MHz

96 ns

LHC : pp crossing rate 40 MHz

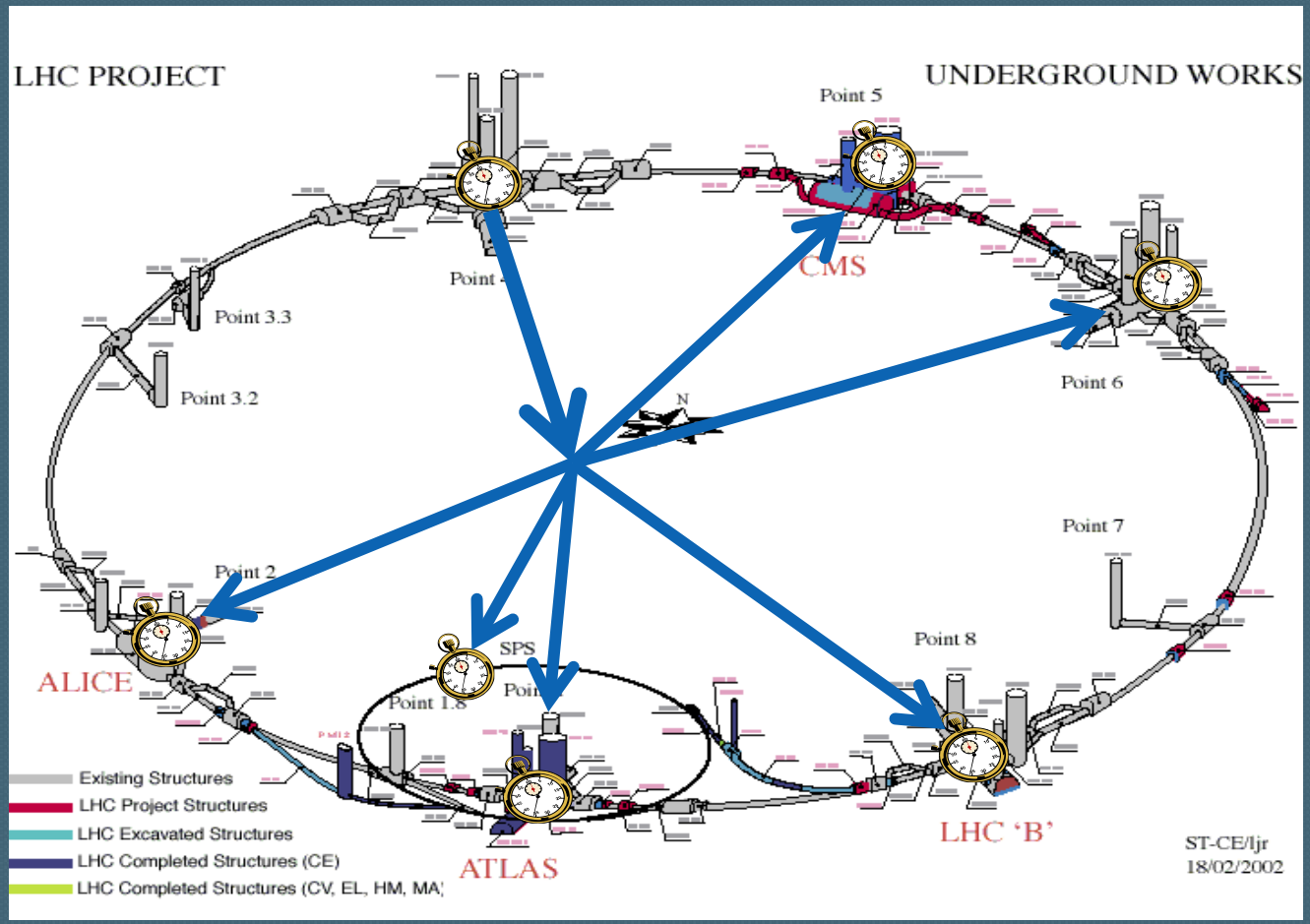
25 ns

CLIC : e^+e^- crossing rate 2 GHz (pulsed)!

0.5 ns

Timing, timing, timing

- The master clock of the entire complex:
  - Proton bunch crossings == Accelerating radiofrequency
  - Distributed to all equipment over many kilometers of fibers



Timing, timing, timing

- Key points
  - Experiment requires a global clock receiver interfaced to the timing distribution in the experiment
    - Should be able to adjust the global phase of the experiment
    - Presence and stability of the clocks must be monitored locally
  - The accelerator clock(s) are the global master clocks of the entire readout system
    - Sampling the detector signal at the optimal point at the FE
    - Sample the fast readout control commands
    - Drive the operational logic
    - Drive the data links

    ➔The phase between the clock and the bunch arrival time must be monitored locally
  - All stages of clock distribution must respect
    - Reproducible and locally controllable fine phase(O(100ps))
    - Reproducible and locally controllable transmission delay / latency (clock cycles)
    - Low jitter

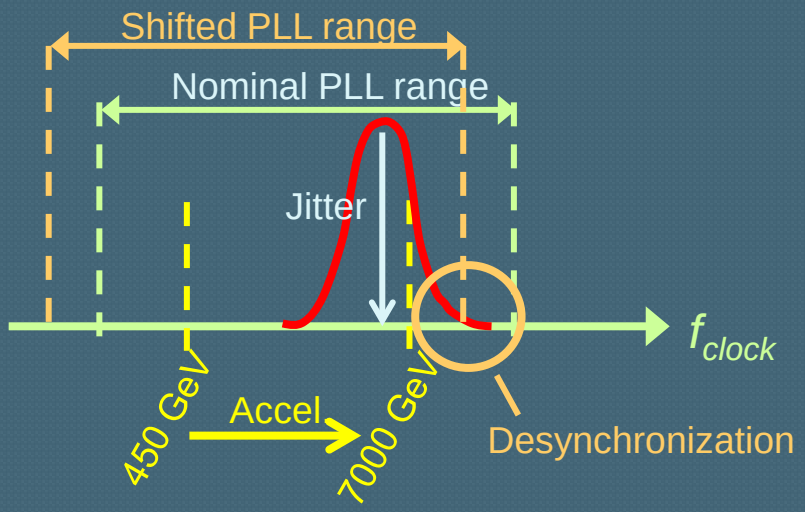
    ➔ Local quartz-based PLL circuits are generally needed to clean-up clocks
  - Ultimate set up of all phase adjustments and latencies with beam and consecutive forced sampling of bunch crossings (“timing scans”)

Timing, timing, timing

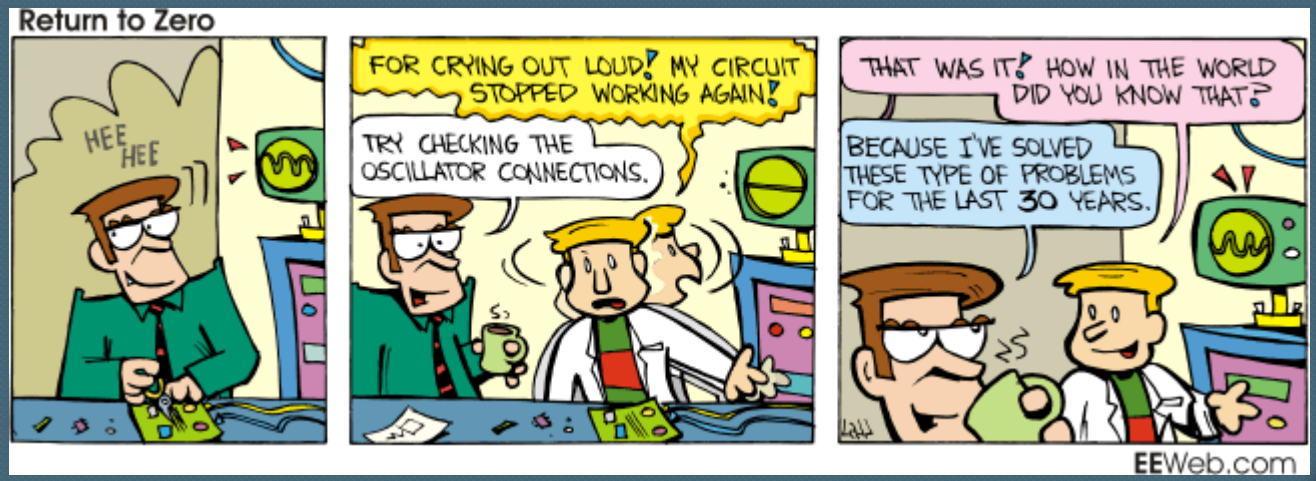
! Watch acceptance range of PLL circuits!!

- Affected by environment

➔ Should include the entire energy range of the beam (injection → flat top)



Clock and network are blamed for everything that goes wrong…!



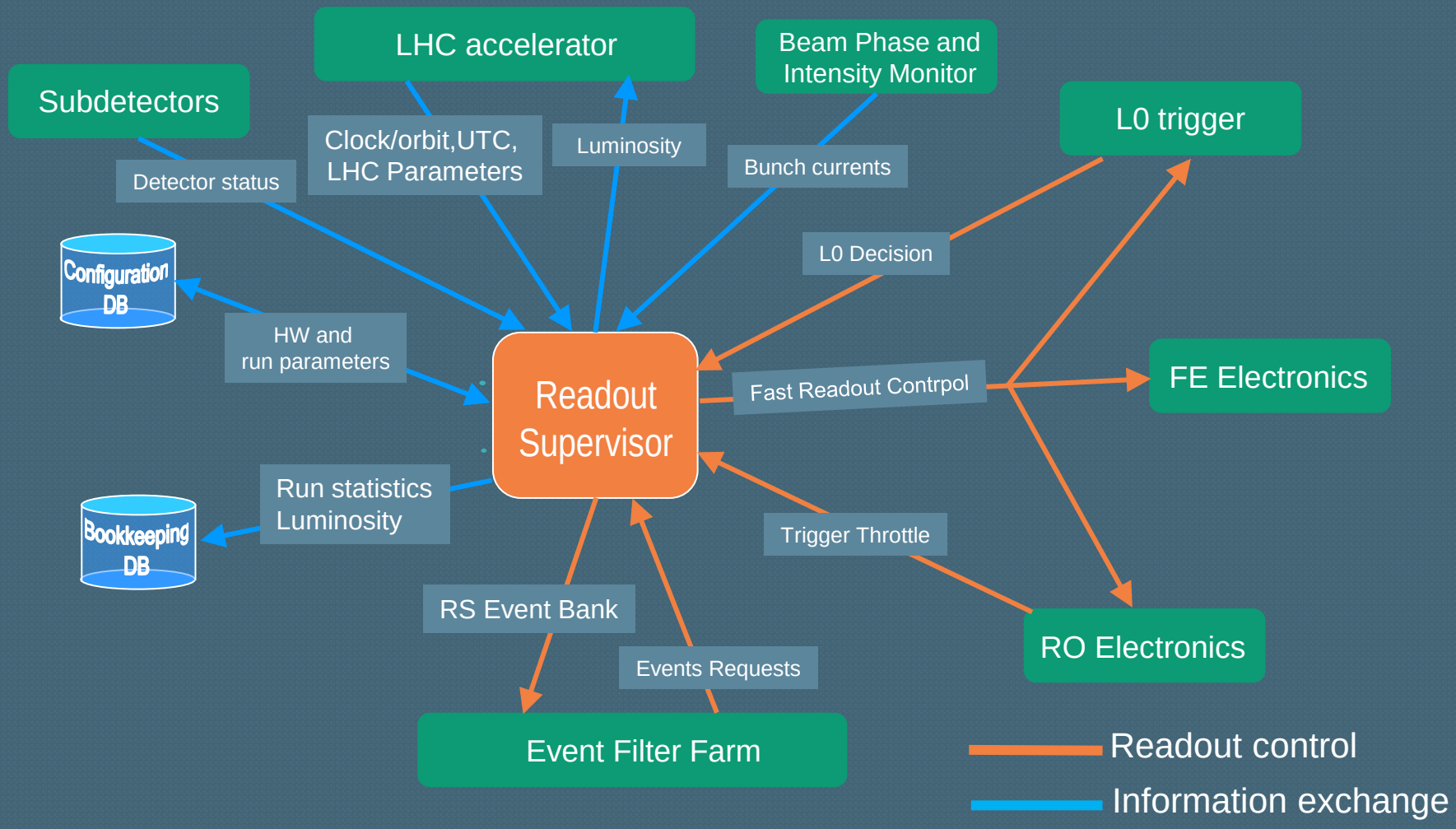
Readout Electronics Functionalities

- Data transport and local processing is a relatively “simple” task
  - Similar challenges in any development
  - But in a HEP experiment it is far from a local task, i.e. *stand-alone*
  - ➔ The challenge is the global functioning with a maximum of intact, calibrated, good data
- Fundamental additional tasks of the readout electronics
  - Timing
  - Synchronization
  - Event (data) rejection based on trigger decision
  - Calibration and special triggers
  - “Special readout modes”
  - Throttling (= detecting and signaling overflow situation)
  - Truncation
  - Data destination assignment
  - Monitoring
  - Fault detection
  - Error recovery
  - ➔ All of these tasks has to follow a coordinated behavior
- Synchronization is the task of ensuring that the data fragments belonging to the same bunch crossing is treated the same way and that they all carry the same identifier in order to be structured together before storage
  - Heartbeat, counters and sequenced resets in synch with accelerator

Timing and Readout Control

- The additional fundamental tasks driven by a central “readout controller”
  - In many experiments, this is associated with the central first level trigger
  - However, trigger is just another subsystem which produces the physics decision
    - Only one - expensive set of links with FE
  - Readout controllers needs to be flexible, modifiable, redundant, reliable etc
- Readout control commands needed by FE, BE (readout boards) and farm
  - Synchronous distribution
- Adjustable delays on each command at Front-End
  - Timing aligning with time-of-flight, signal cable length, processing times

Readout Control Environment



FE/BE Timing and Readout Control

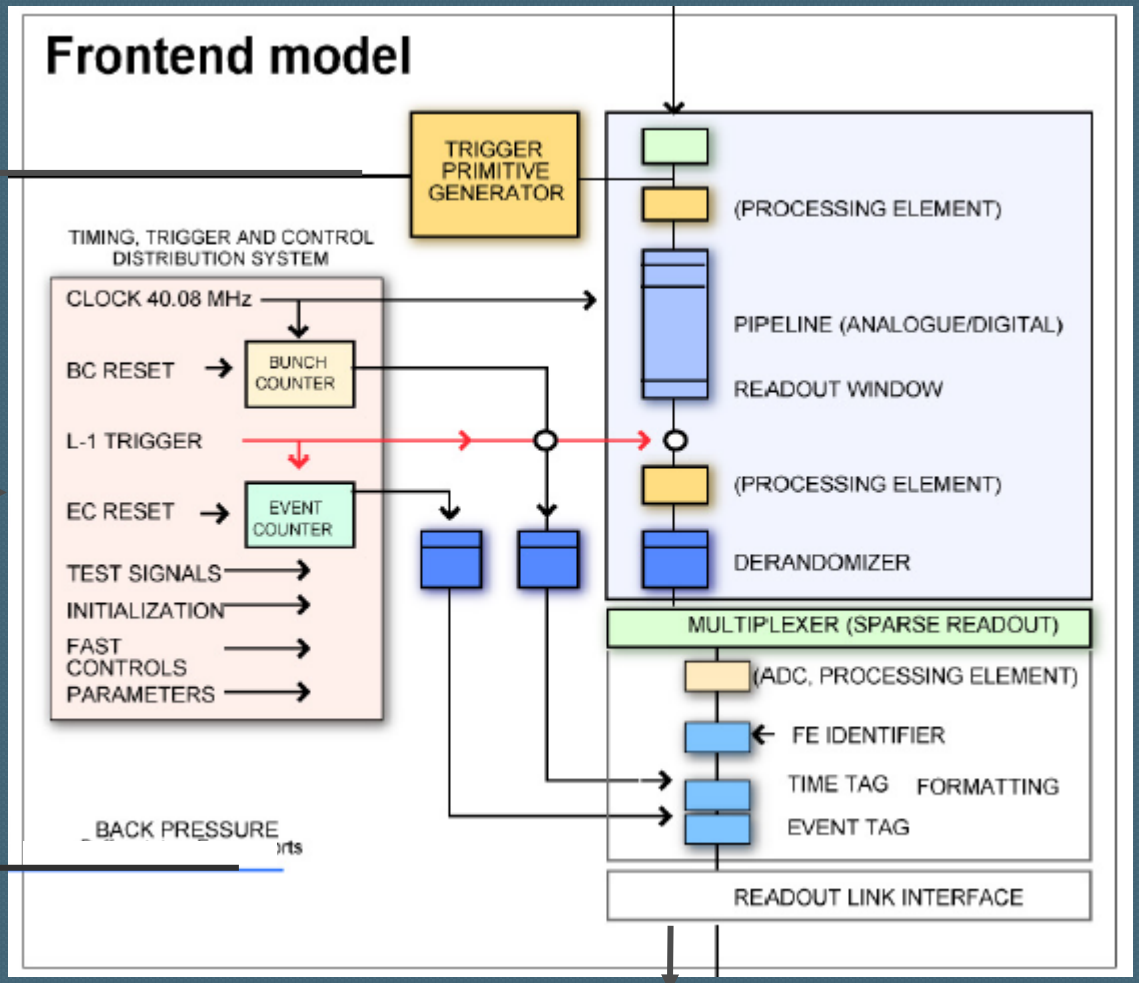
- Readout control → Common specifications!
  - FE is custom electronics
  - Often ASIC = carved in stone
  - Needs a common behavior
  - High speed control

Detector channel

Trigger link

Timing, trigger and fast control

Trigger throttling

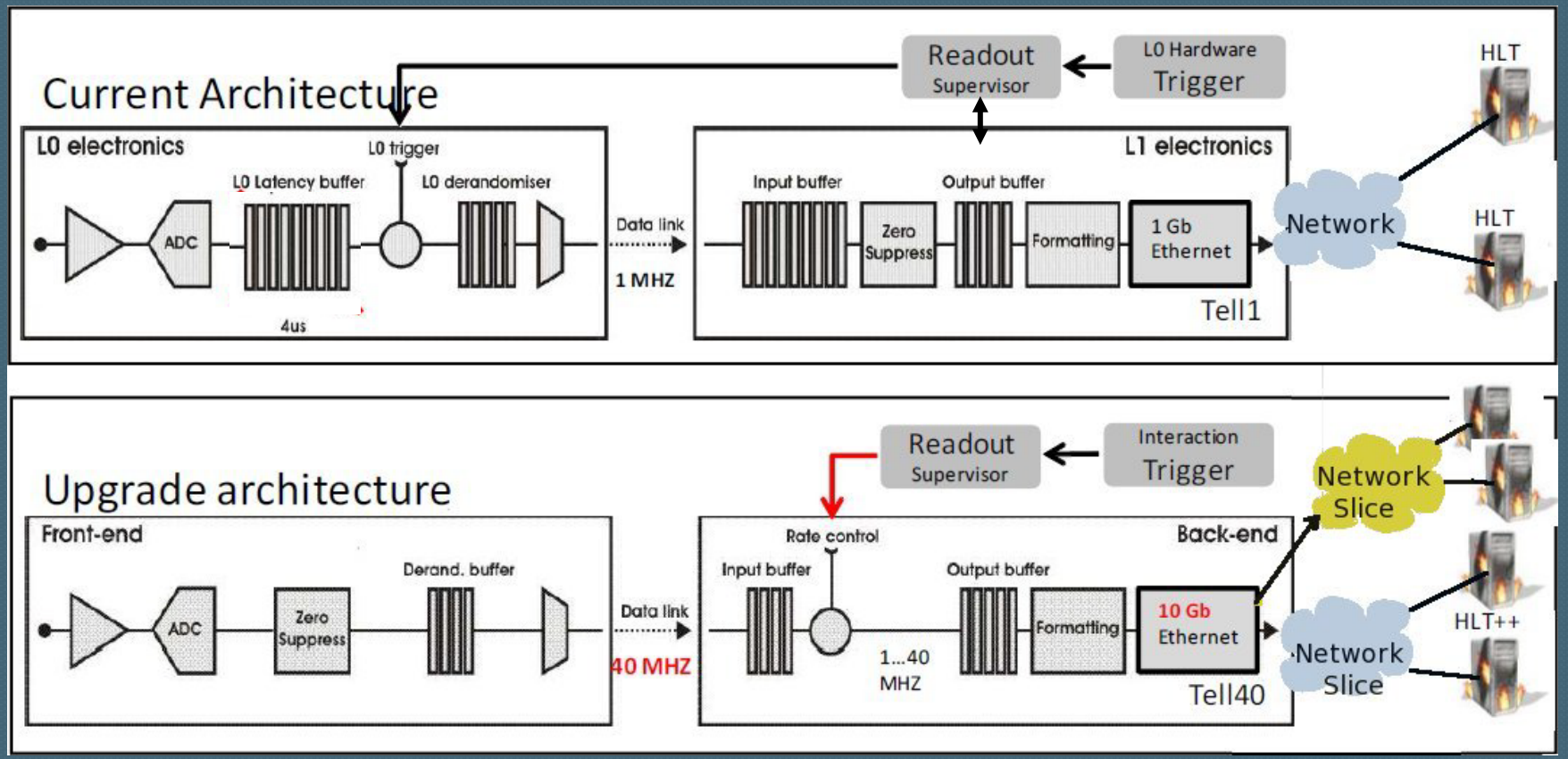


Readout system

R. Jacobsson

Alternative architectures

- Rate reduction at FE or complete FE readout
  - Allows more time and more advanced event selection with a full software trigger

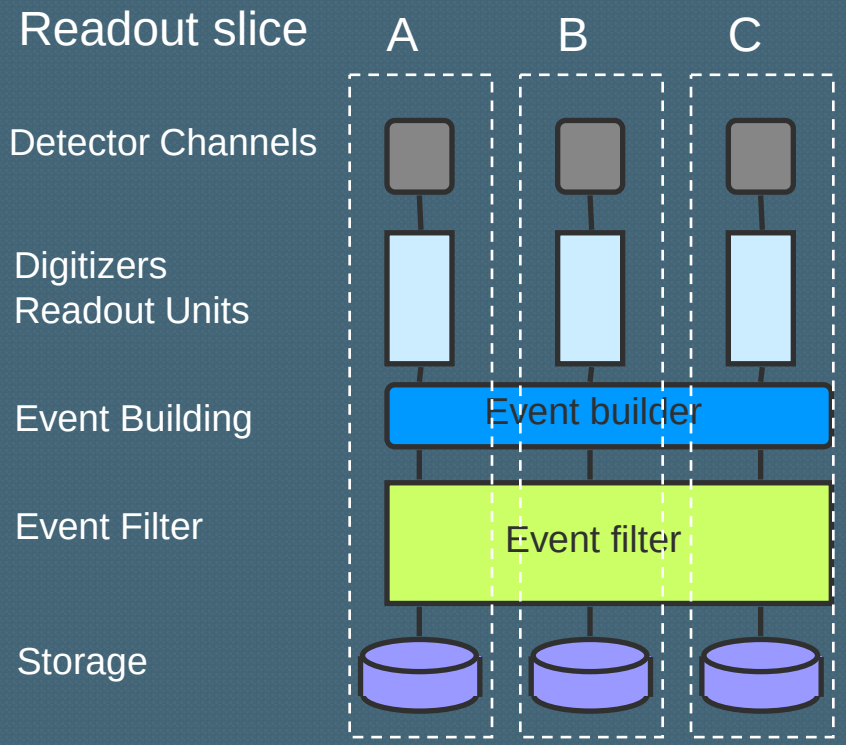


System Dynamics

- Partitioning
  - *Possibility of operating a part of the detector autonomously with its own powering, configuration, triggering scheme, and readout independently from all other parts of the detector.*
  - ➔ Implication on infrastructure and architecture
    - Control and readout slices
- List of use cases
  - Installation phase
  - Commissioning phase
  - Testing and debugging
  - Stand-alone calibrations
  - Problem solving during data taking
- Partitioning is different from *masking*
  - *Disabling a part of the system which causes problems to readout, trigger, event processing*
  - System should allow masking at any level and configure the appropriate dependencies (automatically) in the rest of the system chain

Partitioning

- Partitioning the DAQ imposes strong constraints in the software and hardware. Some resources are shareable and some not



- A trigger source (local trigger or trigger supervisor)
- A set of readout units (data sources) attached to the detector parts to be run
- Some bandwidth on the event building.
- Processing resources.
- Data storage.

- One or several readout slices together makes a partition
- Implications on control
  - The control and monitoring should respect the same borders

System Dynamics cont'd

- Operational fault diagnostics and tolerance
  - Faults are inevitable…. SEU, jitter, etc…
  - Faults must be handled as swiftly and as transparently as possible to the data taking
    - E.g. desynchronizations must not block the data transport process
  - A failure in a FE chip should not stop the data taking of 99 999 other chips…
    - Equivalent to channel inefficiency → very limited impact on physics data quality

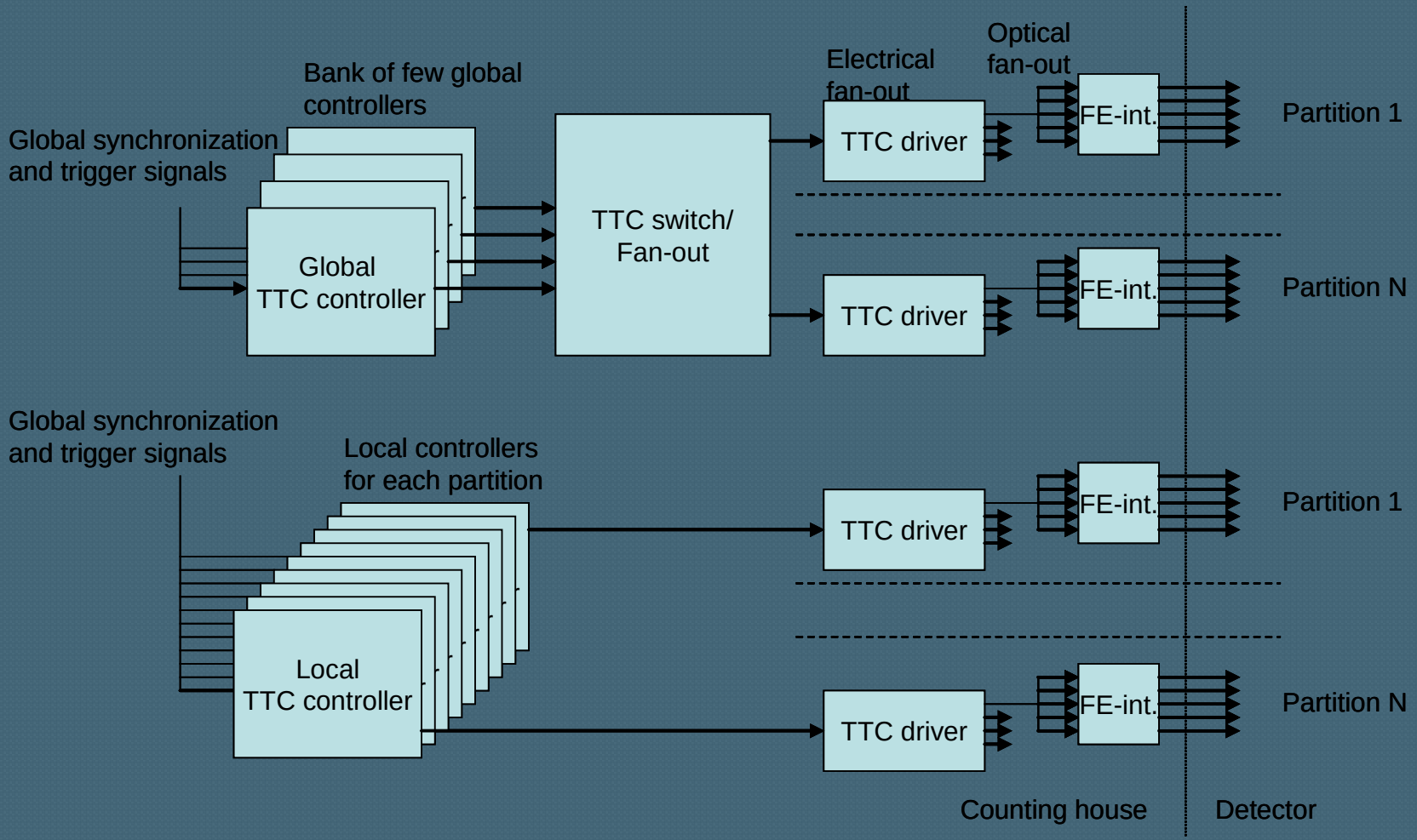
  ➔ Limited error bits to flag data
    - Error banks should not cause 30% deadtime because they increase the readout load through a part of the system

  ➔ Raise status flags via control system as soon as possible
    - Depends on the information needed to actually diagnose the fault

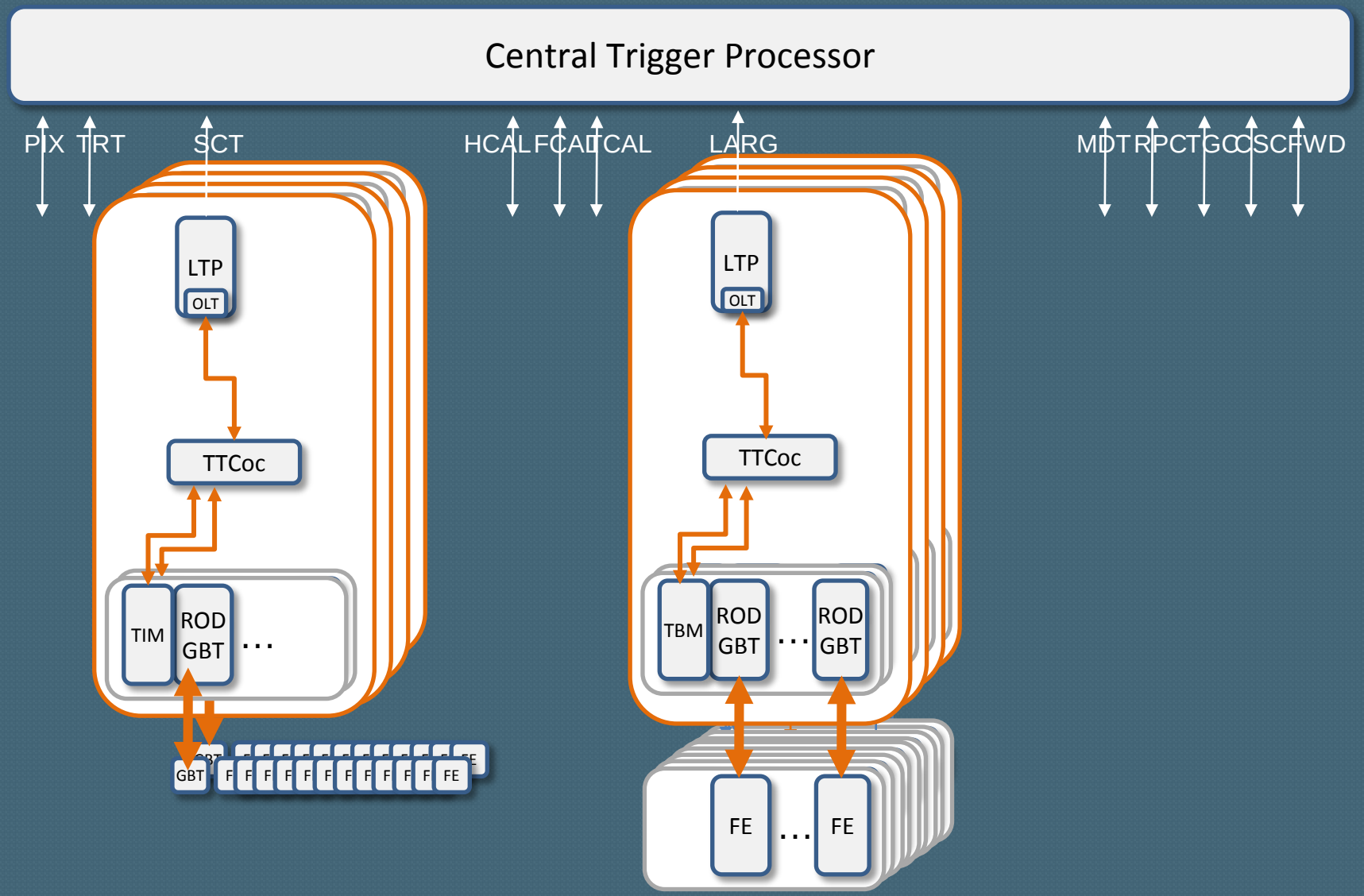
- Loss of control and monitoring communication
  - Should not stop data taking - Only DAQ system is needed to take data
    - Monitoring system is needed to take good data
    - Temporary loss of control or monitoring is not a problem
  - Should be solved "on the fly"

  - Ex. Bypass capacitors and distance to powering!!

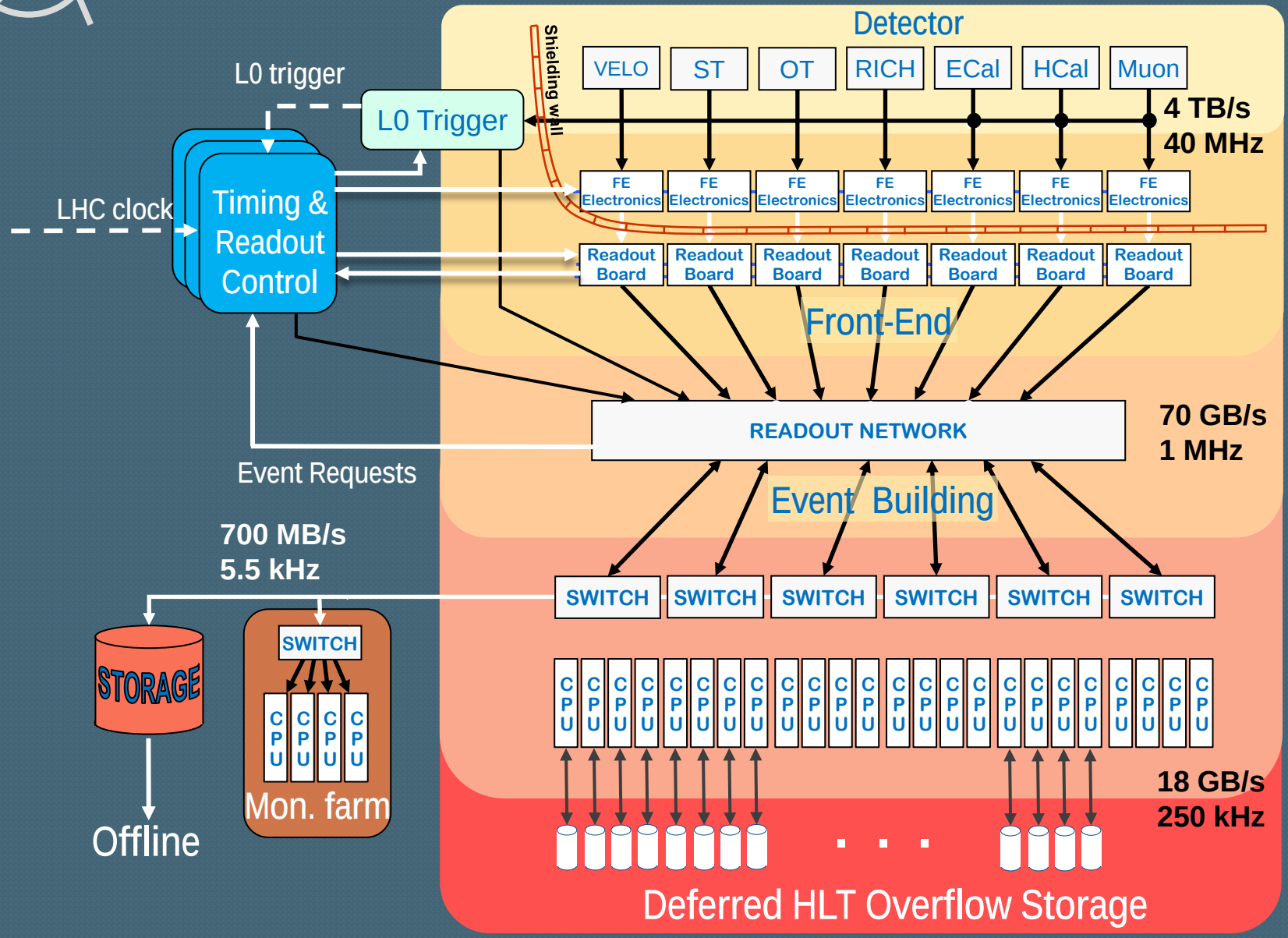
- Dynamic system design



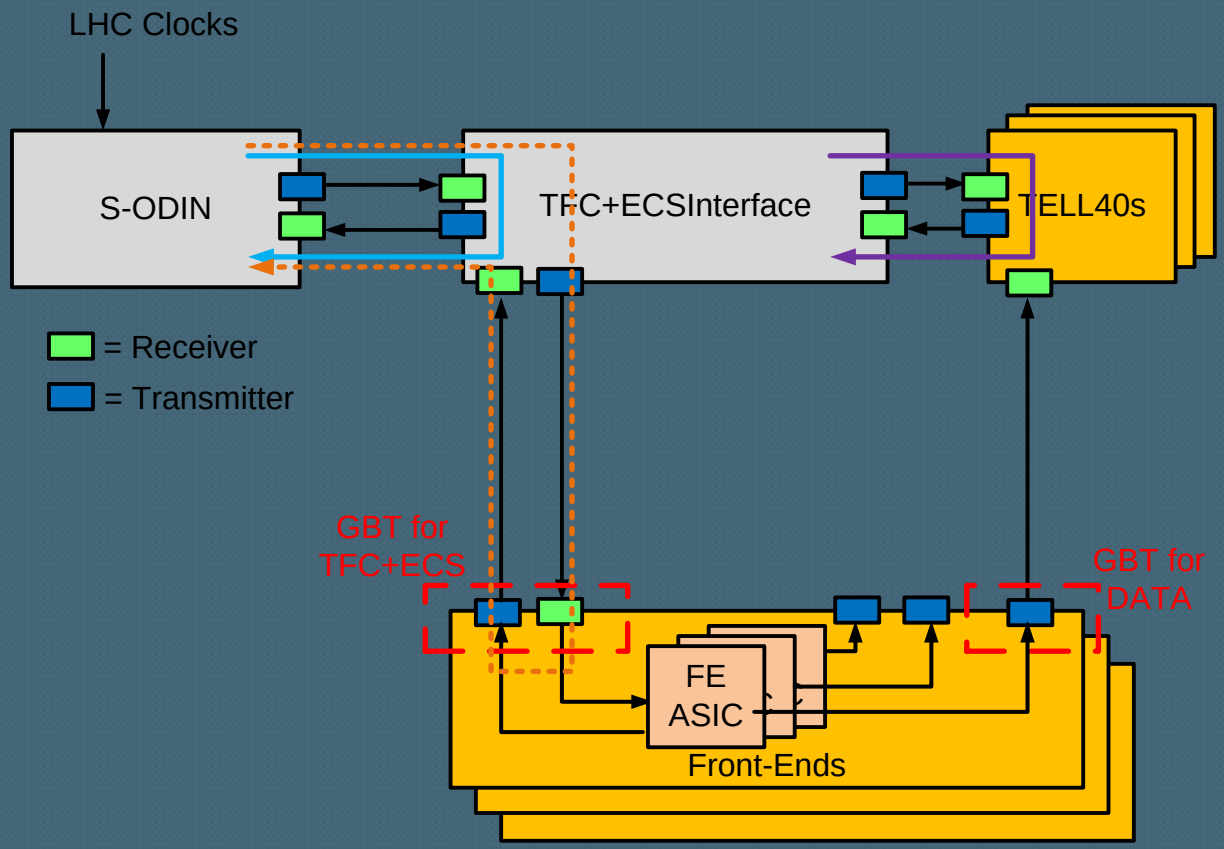
Atlas Readout Control Architecture



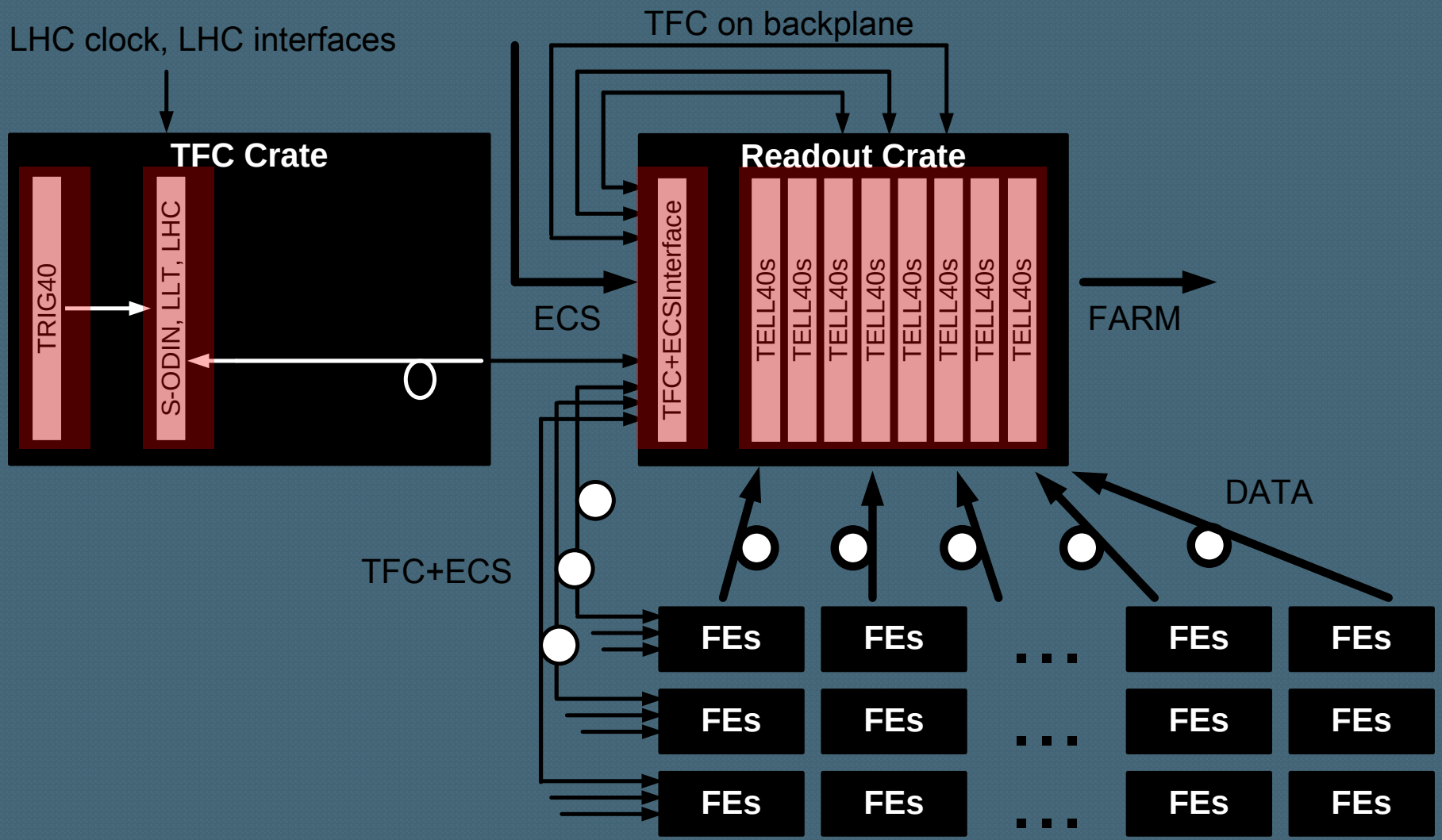
LHCb Readout Architecture



Readout Control Distribution

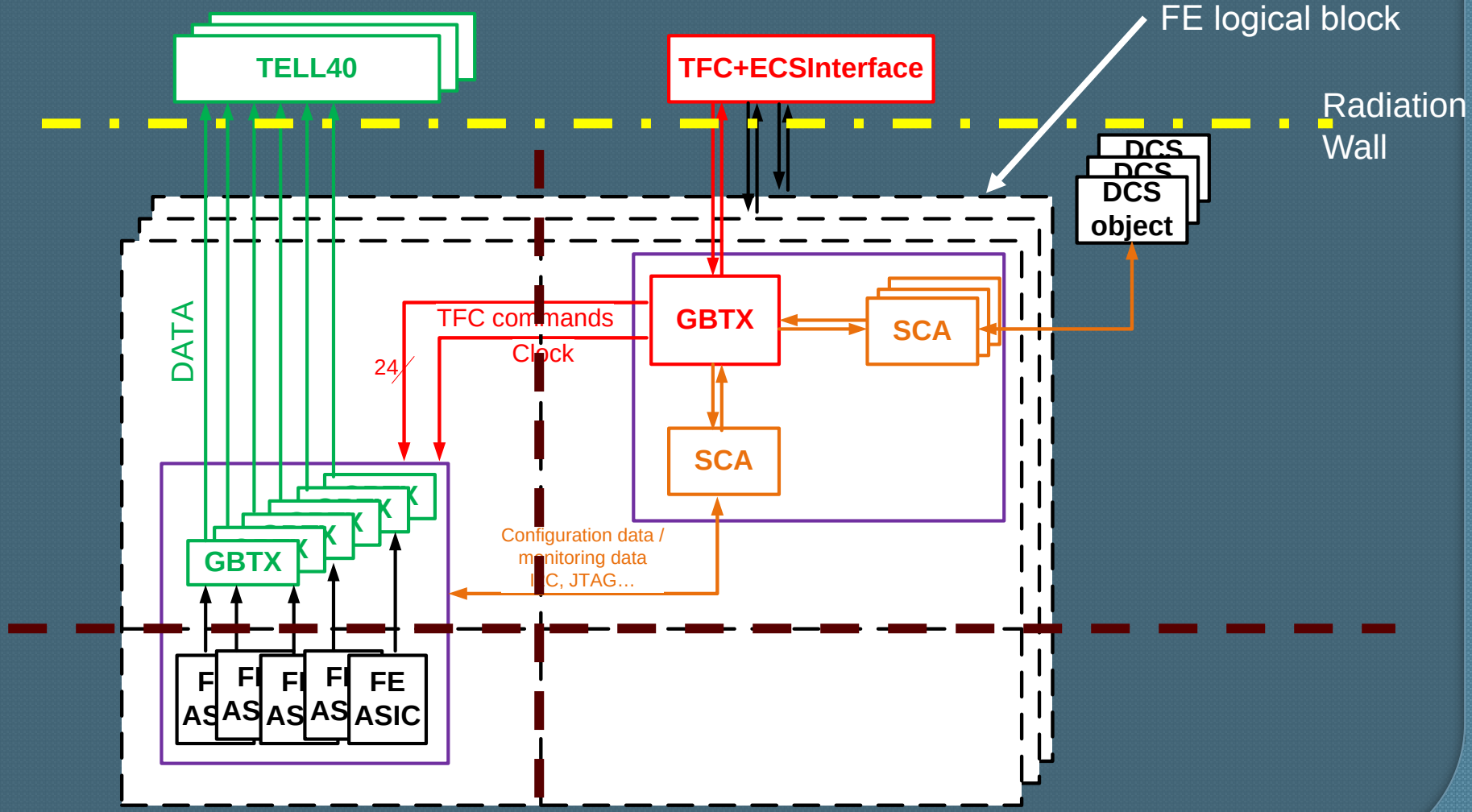


Readout Electronics Physical Architecture



Front-End Architecture

- The splitting between the blocks depend on the requirements



Busses can be complicated...

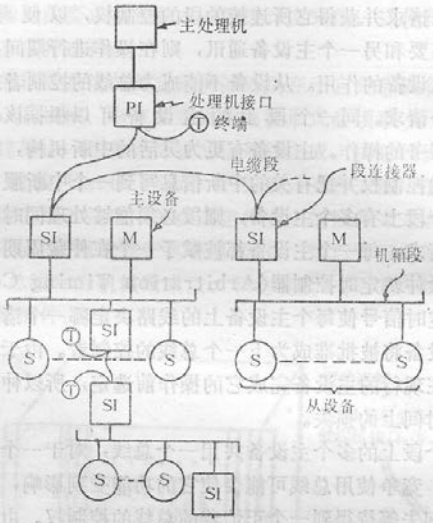


图1.2(b) 快总线系统示例

即使用电缆段。除SI外，设备也可以接到电缆段上，这样的设备同样要服从快总线规范，它的地理地址由开关确定，同时自身必须备有电源。

连接段的其它方法是采用 **缓冲段连接器** (Buffered Interconnect) 或 **段扩展器** (Segment Extender)。缓冲段连接器是一种贮存并传递快总线操作的设备，因此它将破坏主设备和从设备间的同步关系。扩展段的概念允许多个段使用同一个组地址，这是靠段扩展器实现的。和段连接器相比，段扩展器要简单得多。当然这种简化是以牺牲系统及扩展的那些段的灵活性为代价的。

4

数据响应定时器所允许的时间更长时，它可以设置WT。不过实用上并不推荐这种作法，因为这会增加出现不可恢复性错误的危险。用WT脉冲可以实现总线的单次操作以达到诊断的目的。

5·2 主地址周期

快总线操作的主地址周期建立起主设备和一个或多个从设备之间的联系。这些从设备保持同主设备的连接并对数据周期作出响应直到主设备切断连接时为止。

主地址周期在主设备获得总线控制权并在总线上设置GK=1之后启动。然后主设备设置RD=0、MS线，AD线以及选择性地设置PA、PE和EG线。主设备发出地址同步信号AS之后，收到

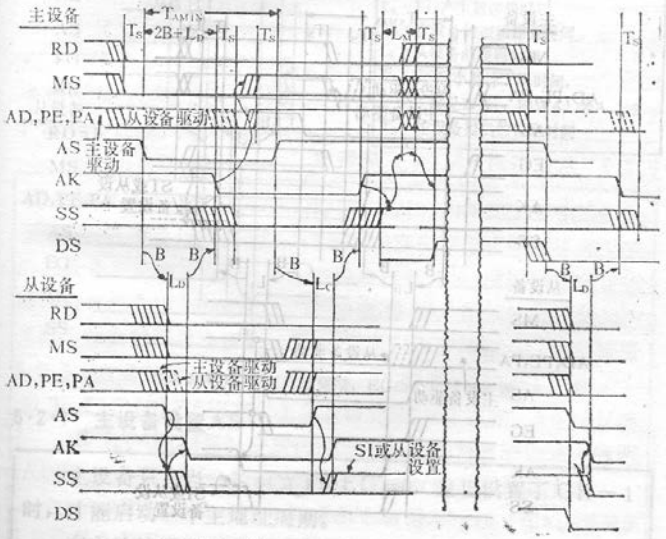


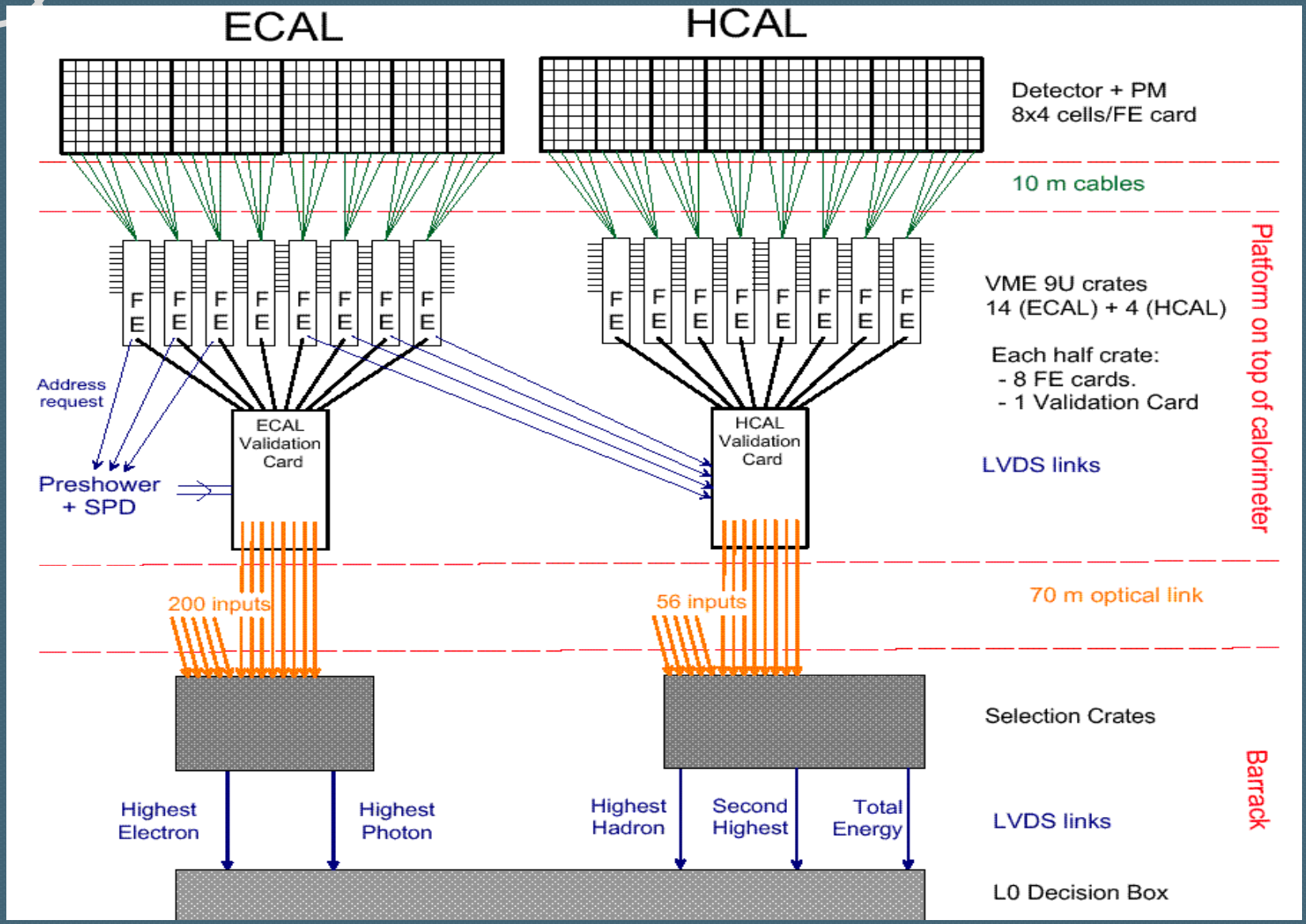
图5.2(a) 逻辑地址周期（符号见图5.2(c)）

63

Busses – to be or not to be

- Reliability issue….
- Data collection or global trigger decision

LHCb Calorimeter trigger

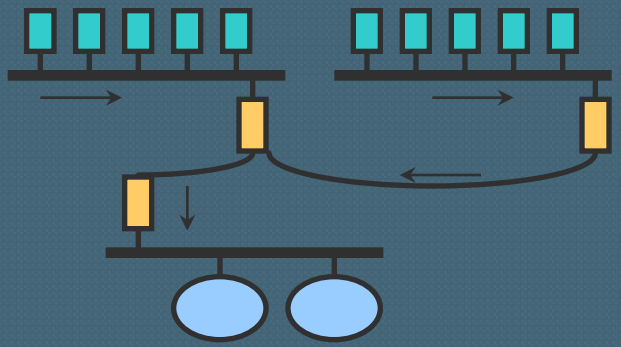


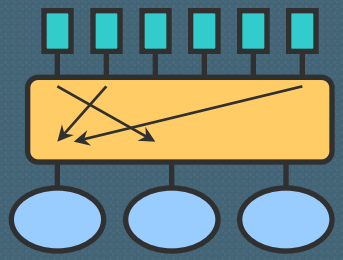
Readout Systems

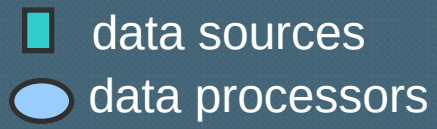
- Since the early 70's there have been a need for a standard for building big readout systems with many hundred thousands of electronics channels (*And we are still waiting for that standard….)*

- Basic components needed:
  - FE boards (digitizers, etc)
  - Readout controllers
  - Crates
  - Crate interconnects

- With these components you can build networks using ***buses*** or ***switches***







- Buses and switches have different ***data transfer characteristics*** and ***protocols***

Experiment Control System

"The one who delivers hardware, delivers software"

- In charge of the control and monitoring of:
  - Data acquisition and trigger (Run control)
    - Configuration
      - Type of RUN, loading of parameters, enabling/disabling parts of the experiment
    - Partitioning
      - Ability to run parts of the experiment in stand-alone mode simultaneously
    - Error reporting and recovery
    - System and data monitoring
  - Detector Control (Slow Control)
    - Gas, HV, LV, temperatures, ...
  - Experimental Infrastructures
    - Cooling, ventilation, electricity distribution, ...
  - Interaction with the outside world
    - Magnet, accelerator system, safety system, etc.

Control System Requirements

- Short configuration time!!
  - No bottlenecks
- Decouple data path from control path
- Common approach in the design and implementation of all parts of the system
  - Easy inter-subsystem integration
- Scalable & Flexible
  - Allow for the integration of new detectors
- Integrate the different activities
  - Such that rules can be defined (ex: Stop DAQ when Slow Controls is in Error)
- Allow Stand-alone control of sub-systems
  - For independent development and concurrent usage.
- Automation
  - Avoids human mistakes and speeds up standard procedures
- Easy to operate
  - Two to three operators (non-experts) should be able to run the experiment.
- Maintainable
  - Experiments run for many years

- GUIs run nothing vital, pull the plug concept

System Configuration

- All the components of the system need to be configured before they can perform their function
  - Detector channels: thresholds, calibration constants need to be downloaded
  - Processing elements: programs and parameters
  - Readout elements: destination and source addresses
  - Trigger elements: Programs, thresholds, parameters
- Configuration needs to be performed in a given sequence
- Databases
  - The data to configure the hardware and software is retrieved from a ***database system***. No data should be hardwired in the code (addresses, names, parameters, etc.)
- Data driven code
  - Generic software should be used wherever possible (it is the data that changes)
  - (Readout flow and control flow are (should be) decoupled!
  - Loss of communication doesn't mean the the data taking has to be stopped, just recover communication on the fly!

Readout Electronics Monitoring/Diagnostics

- Often stated: “FE ASICs are expensive!”
  - Resource economy on spy buffers, counters and status…. NO!
  - The cost of difficult and long diagnostics downtime, and recovery procedures is sizeable

  ➔ Multiplied by the number of components = $$$!

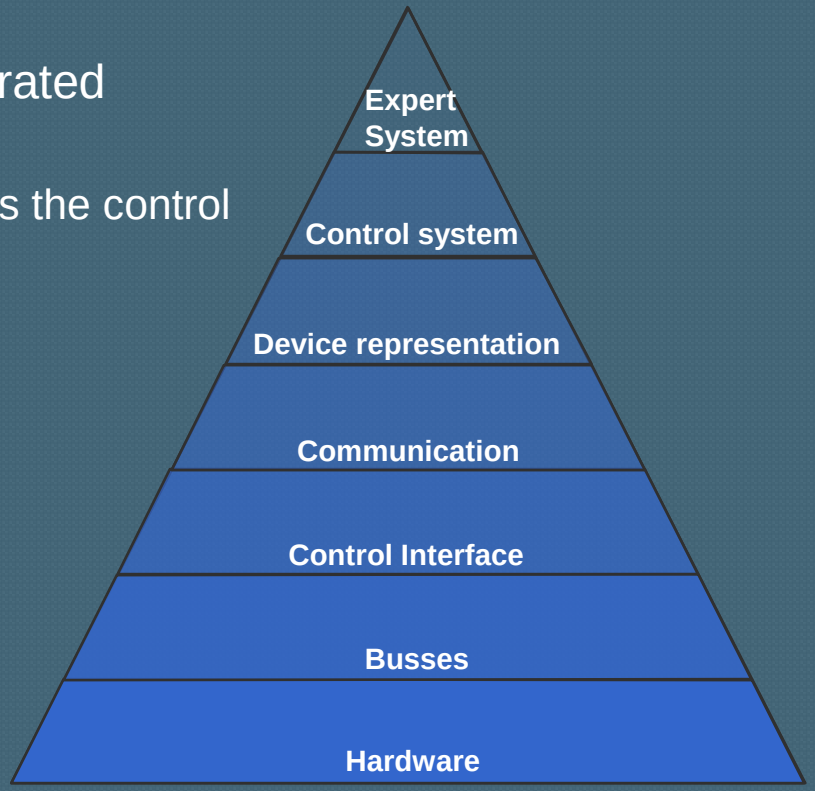
- List of advice
  - No write only registers
  - Automatic checking of write actions (write – read)
  - Counters of sufficient width to avoid wrap-around
  - Simultaneously sampled read buffers for status and counter registers
  - Status registers reflecting the states of circuits and pieces of operational logic should be available on the ECS bus in three versions
    1. Live status
    2. Sampled status by snapshot to be coherent with counters
    3. Latched value upon change to the abnormal state

Control System Hierachy

- Aspects of implementing generic and integrated remote control of electronics devices
  - Define a generic data structure which reflects the control configuration of a board and

    which interfaces to
    - Local control actions
    - GUI display
    - Overall expert system for automation
  - Provide an interface between the functional

    view and the hardware view of the system



  - Provide a simple and economical remote access protocol to any board resource type independent of the bus type.
  - Provide a simple and economical protocol which allows monitoring counter and status information

- Generic view of a distributed control system (ex PVSS)

Device Representation

- A device type is represented by a dynamic data base structure
  - Each board is an instantiation of the structure
- A full representation and storage of all resources for control and monitoring
  - Version
    - Cross-check with board/firmware version
  - FPGA code
    - File pointers
  - Hardware view – registers
    - Readings and settings to verify control operations on hardware (writing always followed by reading)
  - Functional view – parameters
    - Readings and settings to display separately in GUIs
  - State
    - Global status information used by expert system
  - Actions
    - Dynamic structures associated with the server commands and services

```
BoardType {
    struct Version { }
    struct FPGAcode { }
    struct State {
        int RunState
        bool WriteError
        bool StatusError
        bool Monitored
        bool Owner } }
    struct Registers {
        struct Readings { }
        struct Settings { } }
    struct Parameters {
        struct Readings { }
        struct Settings { } }
    struct Action {
        struct ReadWriteRegisters {}
        struct UpdateRegisters {}
        struct ReadWriteTable {}
        struct UpdateTable {}
        struct SubscribeRegisters {}
        struct UpdateSubscribedRegisters {}
        struct DownloadFPGA {}
        struct FPGALoadStatus {} }
    }
```

Device Representation – HW View

- Obviously requires some convention on the type of devices or rather the different access modes
  - E.g. Q1 means FPGA on Local Bus with base address 0x1000
  - E.g. I2C_40 means an I2C device with base address 0x40
- Created with Device Type Editor

Hardware view

```
struct Registers {
        struct Readings {
                struct Q1 {
                        int R000
                        int R004
                        …}
                struct Q2 {
                        int R000
                        int R004
                        …}
                struct I2C_40 {
                        int R00
                        …}
                }
        struct Settings {
                struct Q1 {
                        int R000
                        int R004
                        …}
                struct Q2 {
                        int R000
                        int R004
                        …}
                struct I2C_40 {
                        int R00
                        …}
                }
```

Device Representation – Functional View

- Motivations for a device representation in terms of functional parameters
  - Things depend on functional parameters, not physical registers
  - Economic in terms of gate and access
  - User interfaces – intuitive
  - Saved configurations
  - Settings of functions are applied together

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P_IP_PROTOCOL								P_IP_TTL								P_IP_SERVICE								P_IP_VERSION				P_IP_HDR_LEN			

- Grouping of functional parameters in functional blocks to which they belong
  - Applied together

- Mapping between the functional view of a device and the hardware view
  - Registers → Parameter decoding
  - Parameters of function → Register encoding

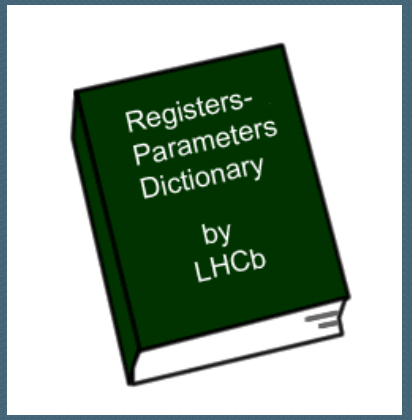
Functional view

```
struct Parameters {
    struct Readings {
        struct HW {
            int H_CLK_EXT
            …}
        struct System {
            int P_L0_LATENCY
            …}
        struct Status {
            int S_ERR_PWR
            …}
        struct Enable {
            bool R_L0_EXT_ENB
            …}
        …}
    struct Settings {
        struct HW {
            int H_CLK_EXT
            …}
        struct System {
            int P_L0_LATENCY
            …}
        struct Enable {
            bool R_L0_EXT_ENB
            …}
        …}
```

Translation Registers ←→ Parameters

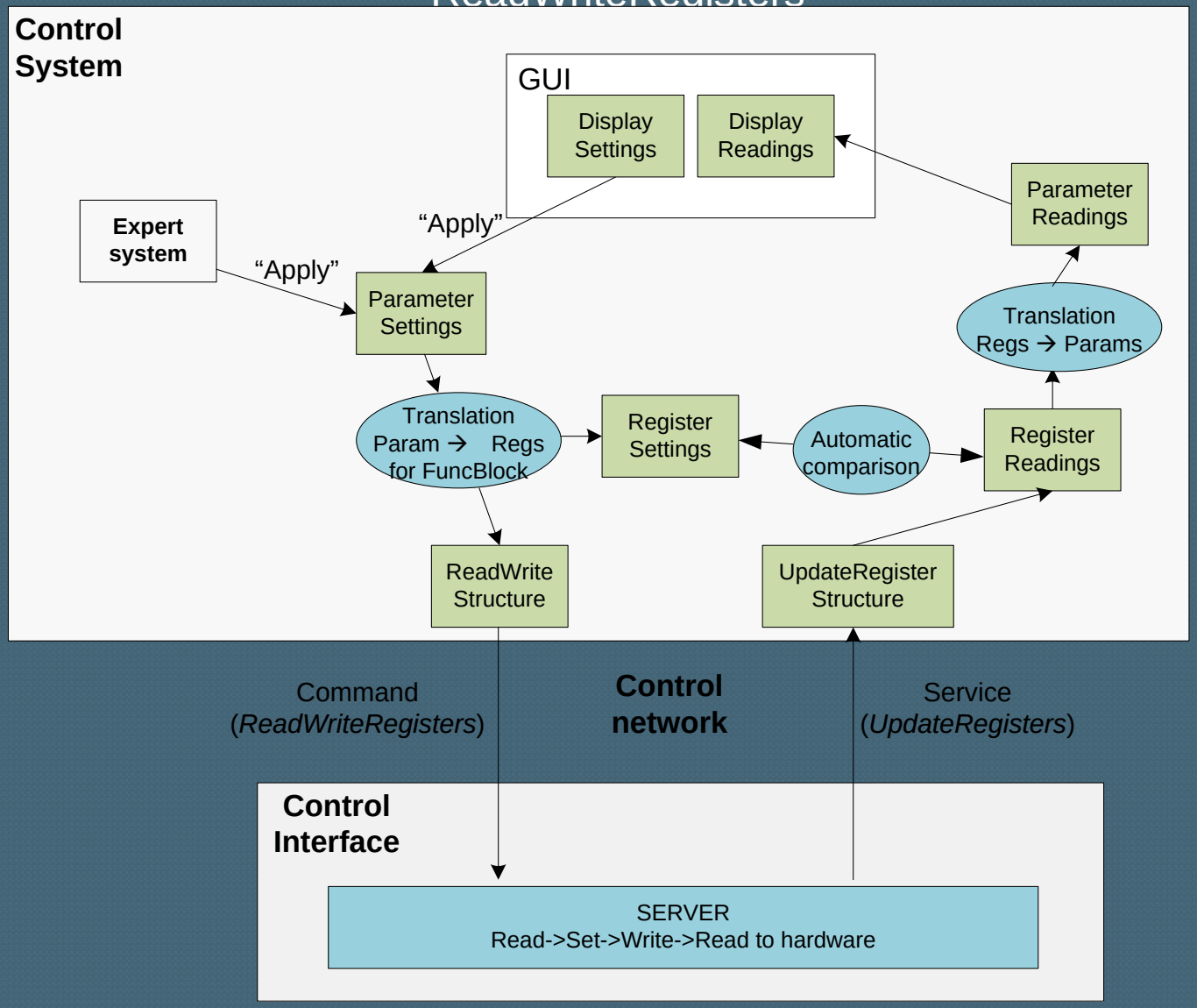
- Instead of hard-coded → Dynamic translation whenever there is a read/write operation using descriptor information
  - Register Descriptor
    - Same structure for any device type
    - One instantiation per device type
    - Information for Register → Parameter, Functional block → Registers, Data Subscription
    - Parameter info: {Addr, Method, Type, ParamName, FuncBlock, Width, Position, Check}

```
Device1 {
      struct Version { }
      struct Registers {
            struct Q1 { }
            struct Q2 { }
            struct Q3 { }
            struct Q4 {
                  . . .
                  string R01C [ ] ={ {0x4034, 1, 1, P_IP_HDR_LEN, FrontEnd, 4, 0, 1}
                                     {0x4034, 1, 1, P_IP_VERSION, FrontEnd, 4, 4, 1}
                                     {0x4034, 1, 1, P_IP_SERVICE, FrontEnd, 8, 8, 1}
                                     {0x4034, 1, 1, P_IP_TTL, FrontEnd, 8, 16, 1}
                                     {0x4034, 1, 1, P_IP_PROTOCOL, FrontEnd, 8, 24, 1} }
                  . . . } }
      struct FuncBlocks { }
      struct DataSubscribe { }
            }
```



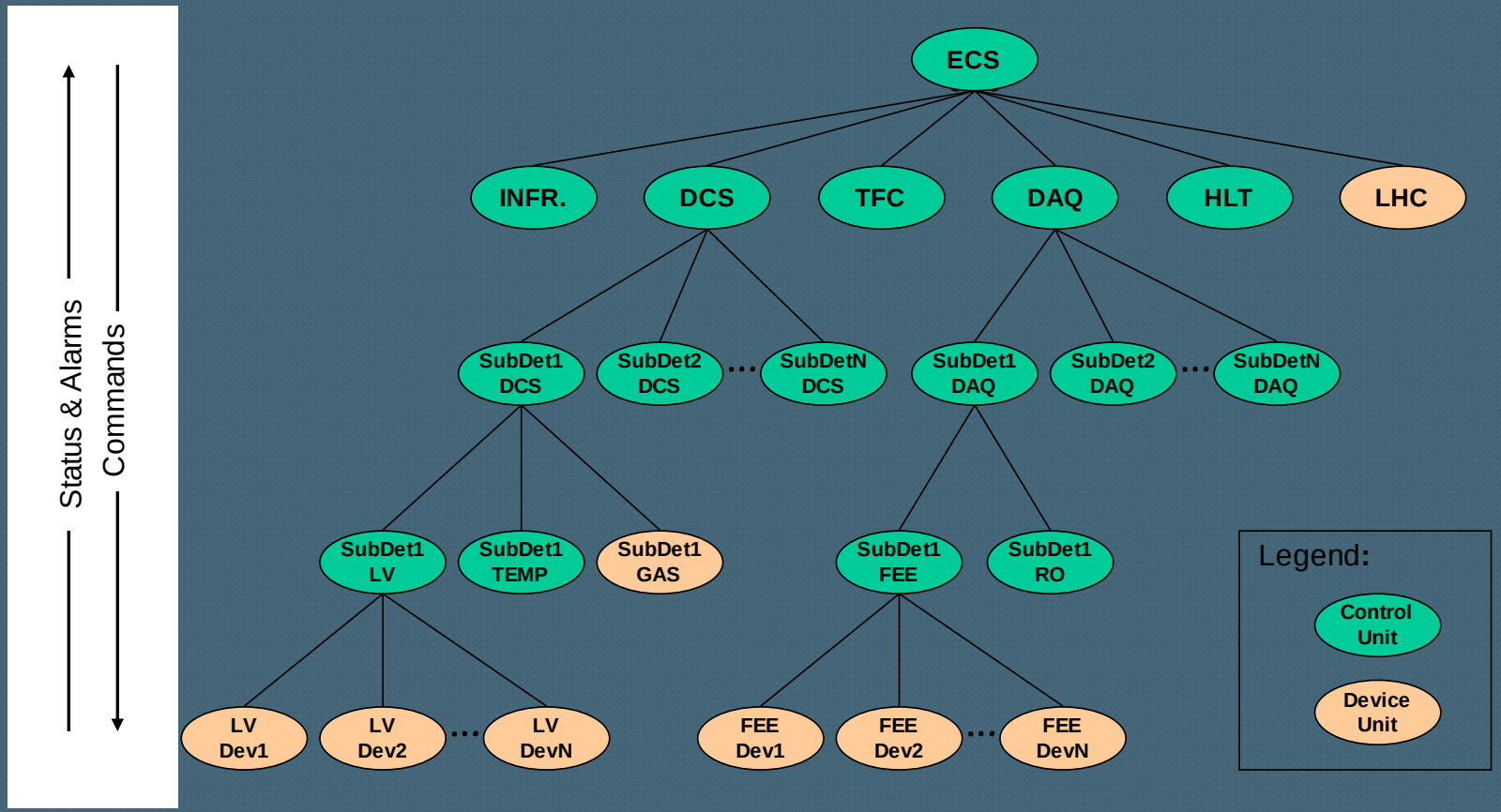
- Descriptor Editor
- Generic Translation API Manager

ReadWriteRegisters



Control System Integration

ex. LHCb Controls Architecture



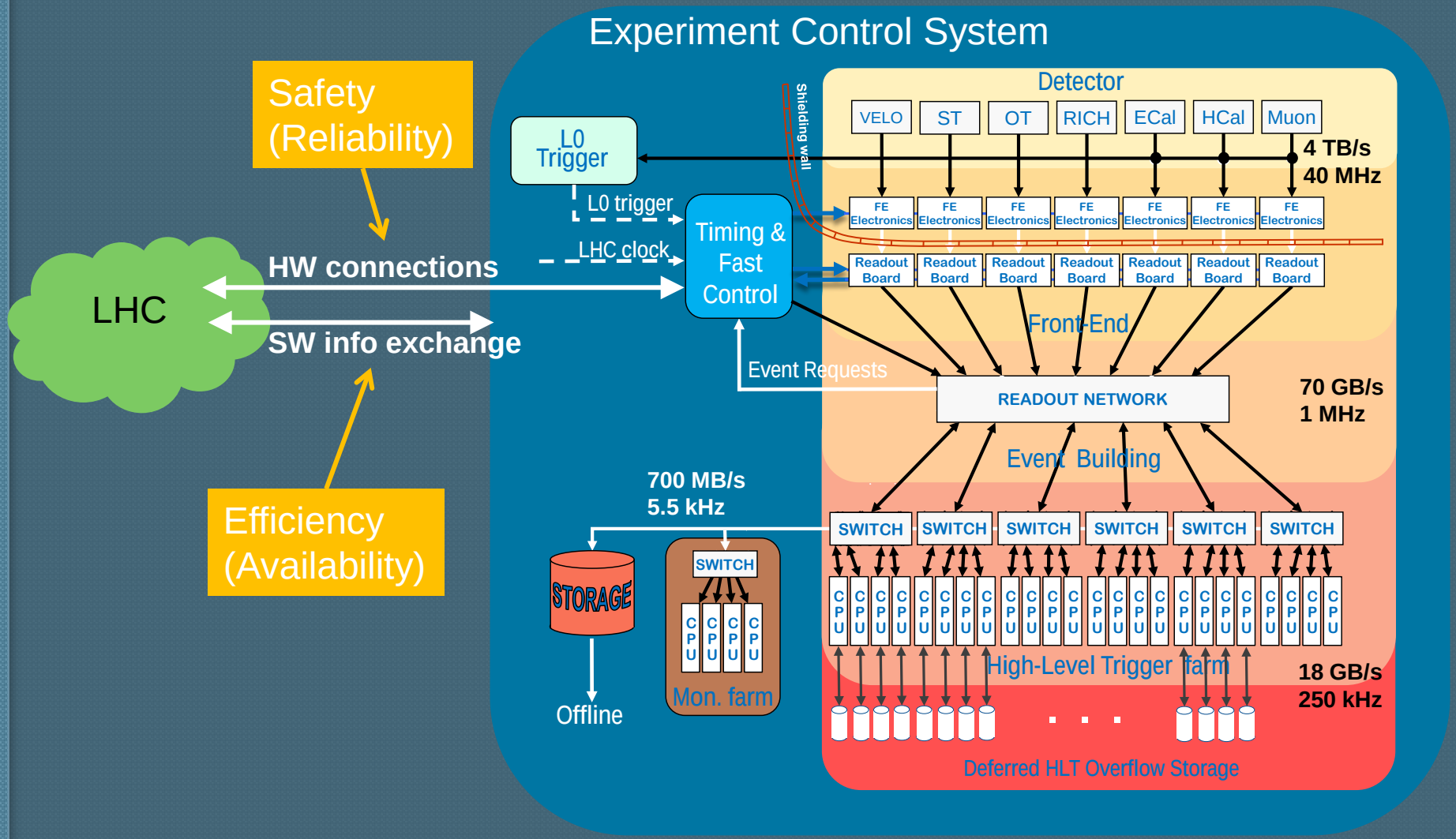
Control System Hierarchies

- General purpose strategy for a large control system
  - *Commonality*
  - Generic control interface server
  - Dynamic communication protocol
  - Representation of resources and storage in control system
  - Automatic handling of parameters and registers
  - Interface to supervisor control system and automated expert system

  - Fault detection and automatic recoveries

- Operation slogan
  - “Produce your own data” = Physicist non-expert shifter to operate experiment 24h

Global Operational Control



System Validation and non-conformities

- Specifications, and Non-conformities, non-conformities, non-conformities

- Consequences of common denominator
  - ➔ Detector is as good as its smallest constituents
  - ➔ Less performance (TOF, RICH)

- System validation in final environment
- Validation and review process

- Testing phases
  - Unit testing => one or a few classes/procedures
  - Integration testing => different components together
  - System testing => overall functionality through all layers

- By the way, pipelined logic is strongly preferred, limited use of combinational logic
  - More tolerant to code changes and environmental changes, faster re-validation process


Return to Zero
OUR NEW OSCILLOSCOPE HAS EVERYTHING BUT THE KITCHEN SINK!
YEAH, TOO BAD WE CAN'T ACTUALLY OWN IT...
BEEP-BEEP-BEEP-B
SUBSCRIPTION EXPIRED!
PLAY ENDS IN --
14 SEC.
ADD $$
YOU'VE GOT THE NEXT ONE, RIGHT?
EEWeb.com

Simulation and Validation

- Levels of simulation
  - Behavioural/functional simulation - emulation
  - Clock level simulation – synthesizable code = development framework
  - Timing simulation – after placing&routing = resource usage

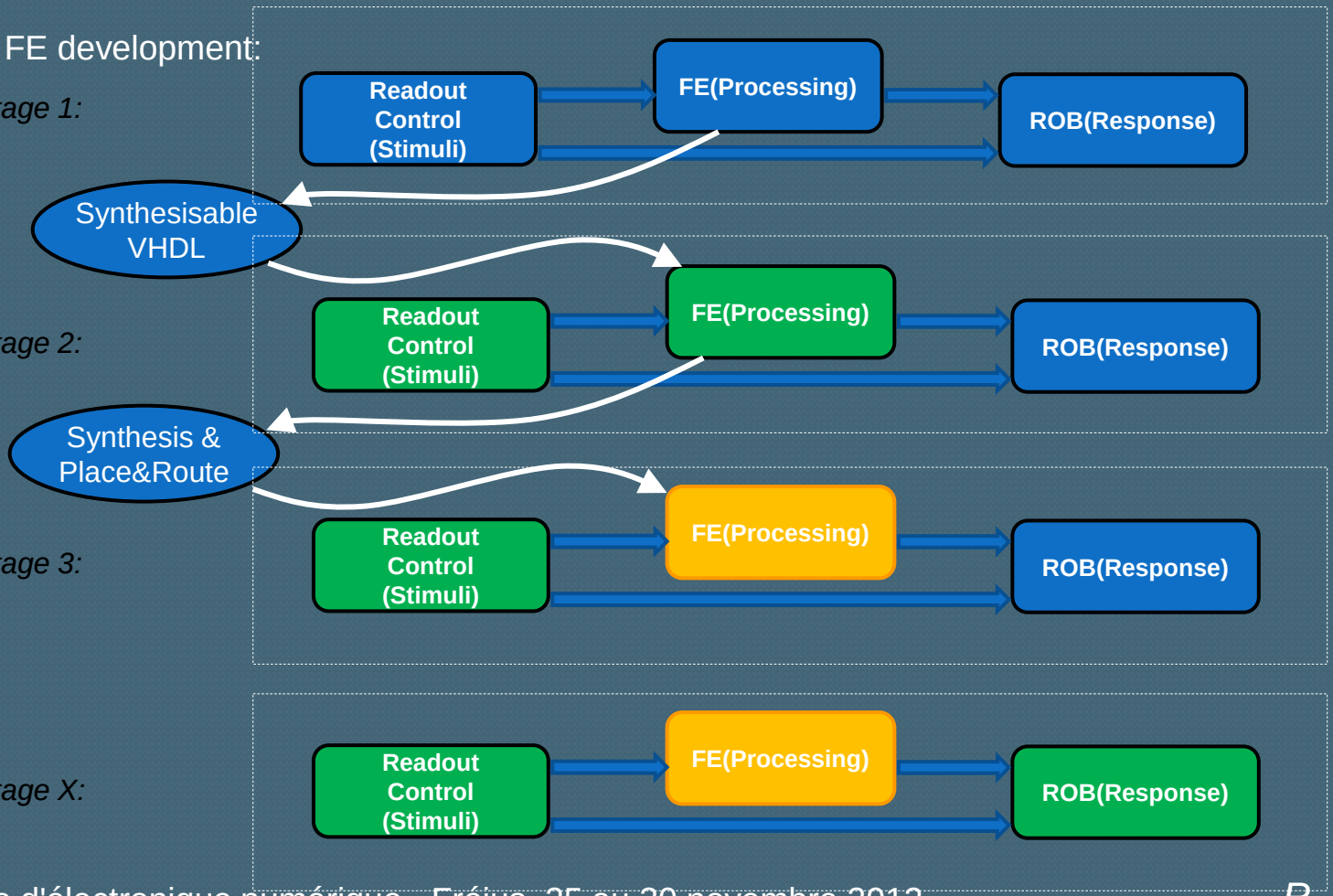
➔ Slice simulation

Emulation

Clock level

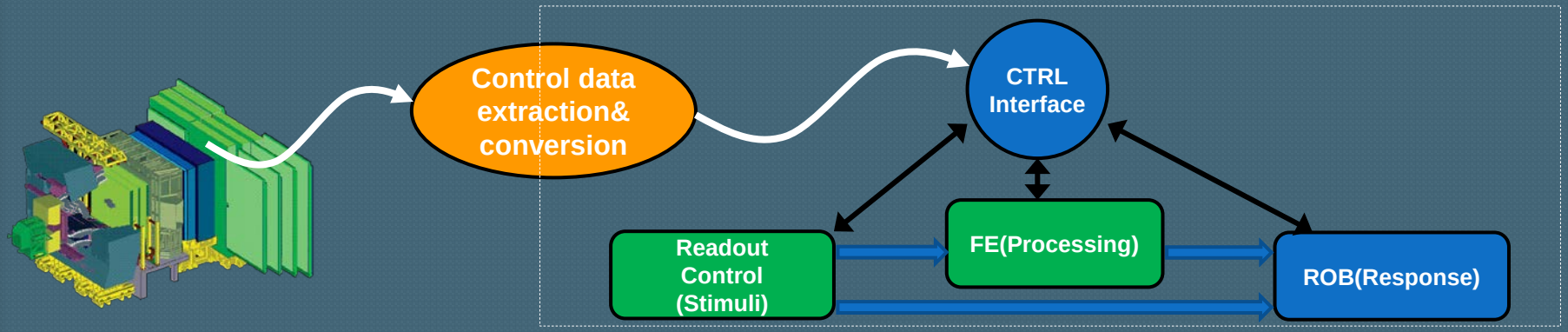
Timing

Ex. FE development:



Simulation and Validation

- Well choses system level simulation tool
  - Personal favourite: VisualElite from Mentor Graphics
  - Make sure models are portable!
- Example shows another fact:
  - ➔ Readout control must be well specified and developed ahead of the rest
  - ➔ Thus, contain flexibility and important reserve resources to accommodate changes
- Control and monitoring resources should be simulated !
  - Emulation of the actual control interface (μController, NIOS, PC….)
  - *Direct extraction and generation of system configuration from real system at single click*
    - *Allows snap-shooting and reproducing the Situation*



- Simulation framework maintained through life of the experiment
  - Mode of operation is bound to evolve in High Energy Physics

A real example

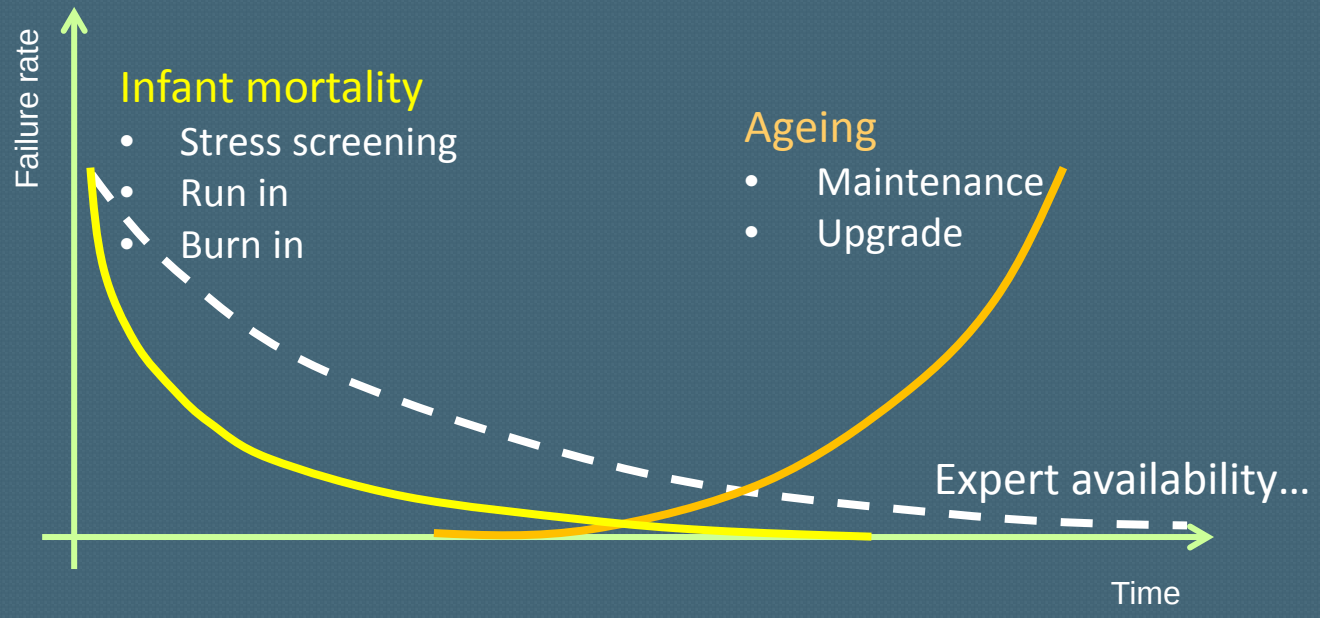
- FE model can be exchanged to check compatibility with common specs

Underestimating The Real Environment

- Grounding
- Shielding
- Heat dissipation

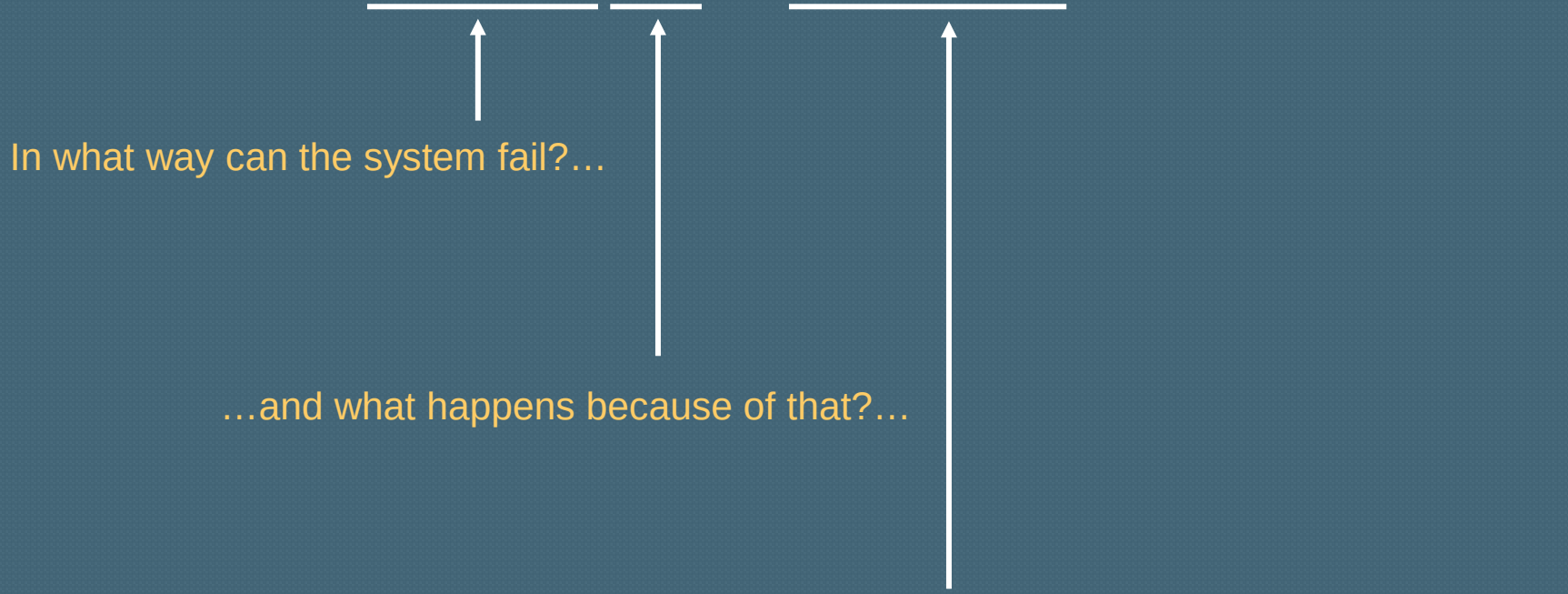
Design with globalism in mind

- Failure rate > probability of failure * number of pieces
- (LHC accelerator 250 kCHF / hour !)
- Availability, reliability, fault tolerance, error behaviour
  - Handling errors, not stop data taking!
  - Low probability of failure multiplied by enormous number of systems
  - Parallel continued validation, problem isolation
- Are the board on the shelf functional?
  - Swapping concept against plugging fatigue



Availability and Reliability

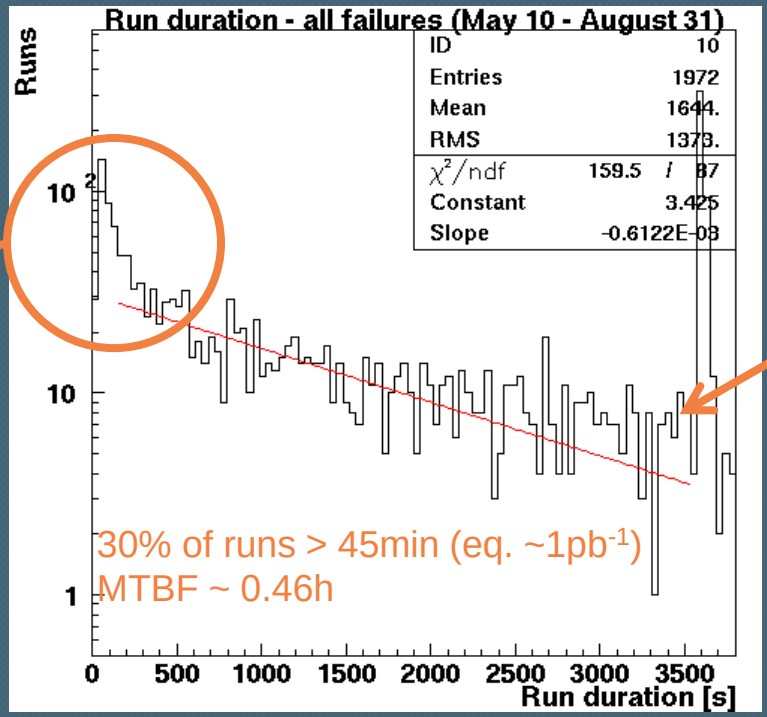
Failure Modes, Effects and Criticality Analysis



Ex. Operational Stability – Statistical analysis

- Main recurrent failure consists of desynchronizations of different types
  - Consequence: deadtime (partial or 100%), incomplete events, missing data, etc
- Assuming system is “as good as new” at start of run
  - Should allow MTBF analysis

System is not “as good as new” on start run



Max run duration

- Stop/start trigger provokes desynchronizations
  - (Removed all runs <30s to avoid polluting the plot with non-solved problems)
  - Often chain reaction of several desynchronizations of different subdetectors in rapid succession
  - Suspect phase lock loop frequency range, jitter tolerance, and insufficient bypassing/decoupling capacitance

Long experiment life cycle

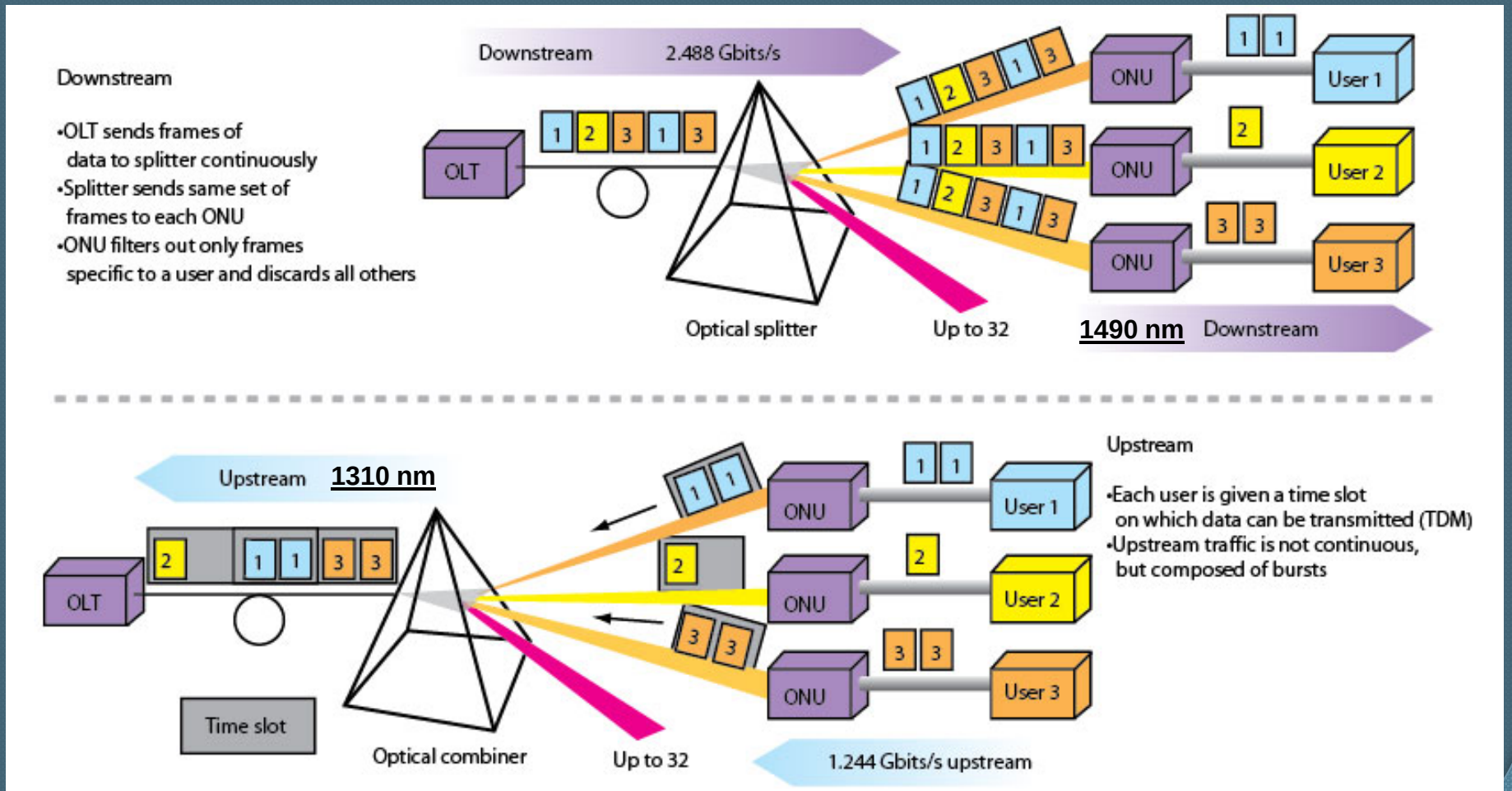
- Why do we need to upgrade?
  - “It works - don’t touch anything!”
- Upgrades are needed because of:
  - Request for improvements and new functionality
  - Changes in other parts of the system, both HW and SW
  - Expiration of support contracts
  - Obsolescence of components
  - Ageing
  - (Keep experts hooked and motivated….)
- Holding back upgrades for too long is not sane
  - Batching many changes makes troubleshooting much more difficult
  - People forget how their code works
  - Modern engineering advocates one small change at a time (a.k.a. “continuous delivery”)

Conclusions

- Trigger, data acquisition and control systems are becoming increasingly complex
- They are not static
  - It is a system that is expected to change with time, accelerator and experiment conditions
  - Provide maximum flexibility in functionality and for upgrades
- Luckily the requirements of telecommunications and computing in general have strongly contributed to the development of standard technologies:
  - Hardware: FPGAs, Flash ADCs, analog memories, PCs, networks, helical scan recording, data compression, image processing, ...
  - Software: distributed computing, software development environments, supervisory systems, ...
- We can now build a large fraction of our systems using commercial components (customization is still needed in the front-end)
- It is essential that we keep up-to-date with the progress being made by industry
- But it is also essential that we go beyond industry!
  - Basic research is what we need to build a long-term potential for technical progress

What is a PON?

- Passive Optical Network
  - Point-to-MultiPoint (PMP) optical network
    - One single fibre in charge of both downstream and upstream transmissions
      - Basis of all the growing Access Network market (also called FTTH/B/C/x)



In a GPON optical network, users are allocated time slots. During these times, they can transmit their data from remote terminals.