Introduction to Detector Readout and Front End Electronics

ISOTDAQ 2015
CBPF, Rio de Janeiro

Rainer Schwemmer
CERN

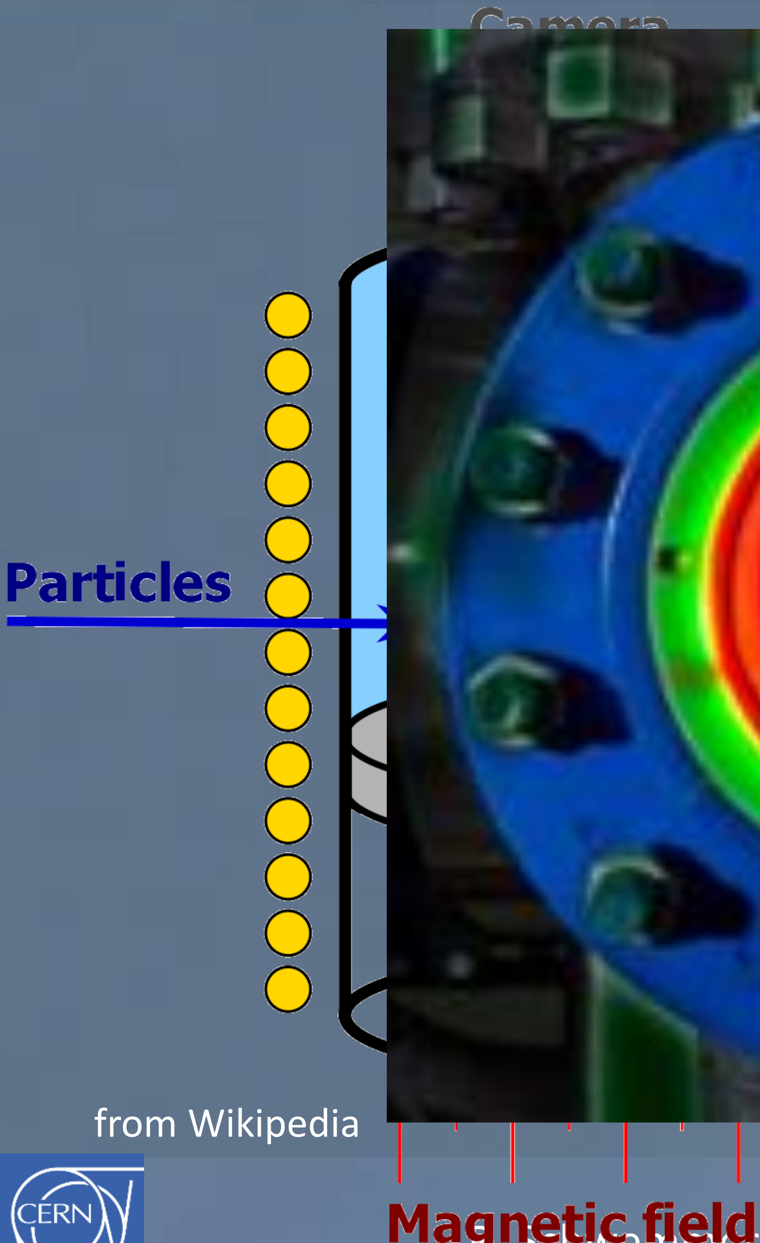
Thanks…

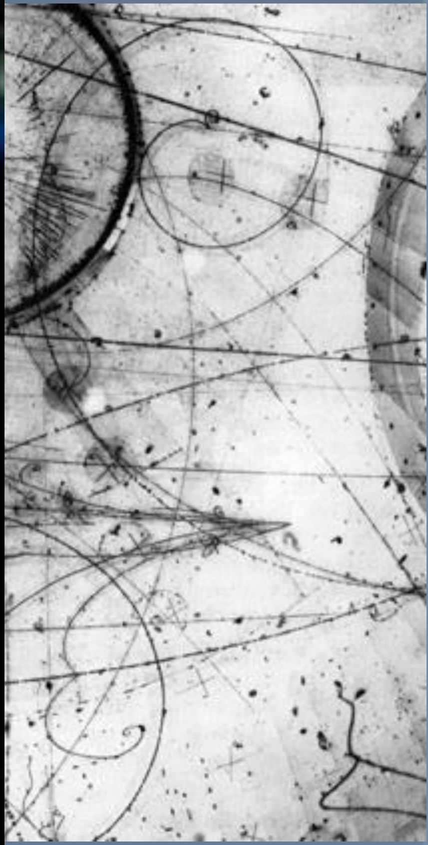
- …to all my colleagues at CERN and elsewhere who taught me what I know

Initial questions

- When working on the DAQ why should we care about Detector Readout and Frontend Electronics?
- It is important to understand our colleagues
- The Frontend electronics is the source of the data
- When designing an experiment the overall cost and complexity will be a compromise between what you can do in your electronics and what you must do in your data acquisition

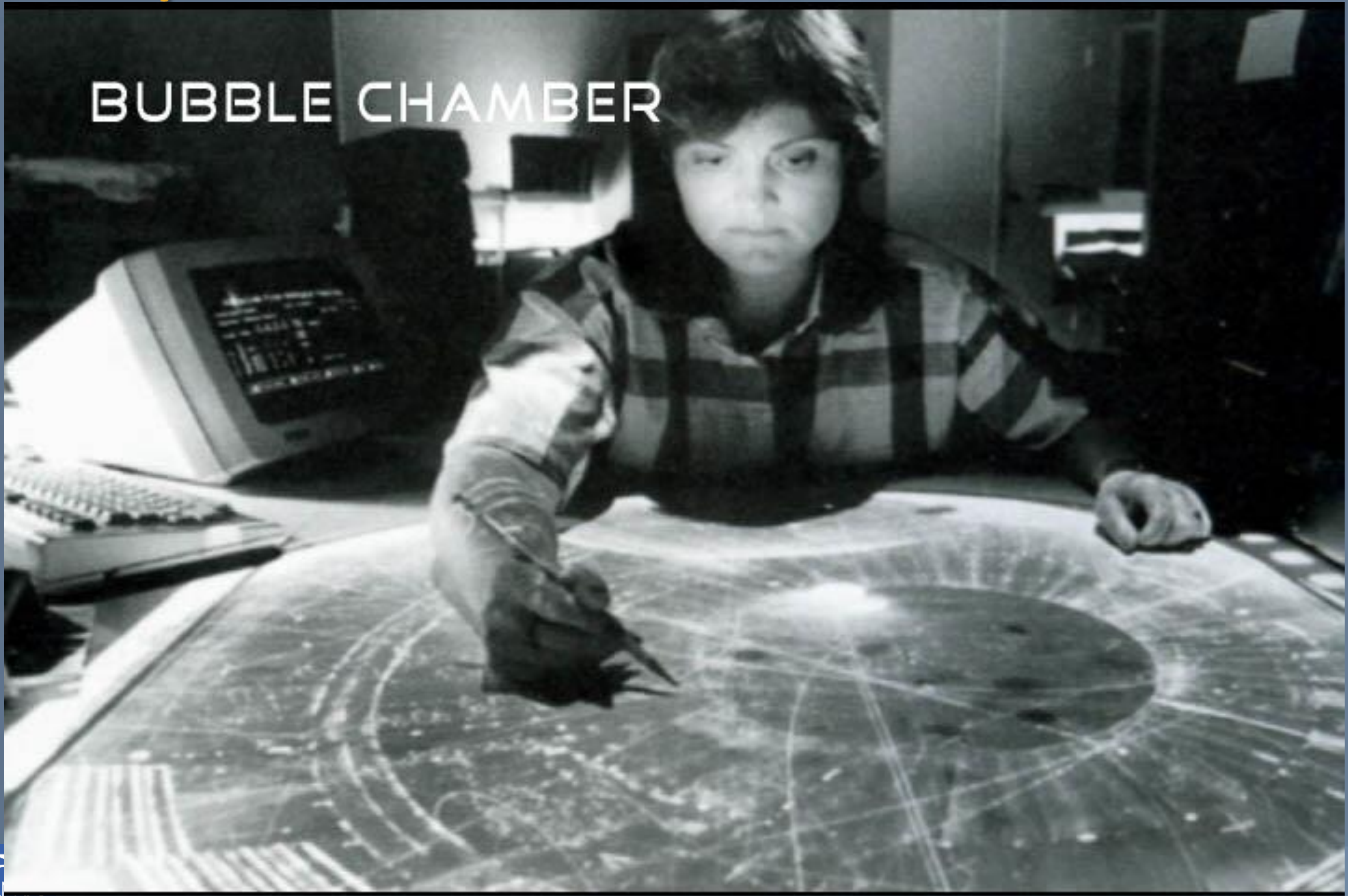
Once upon a time…



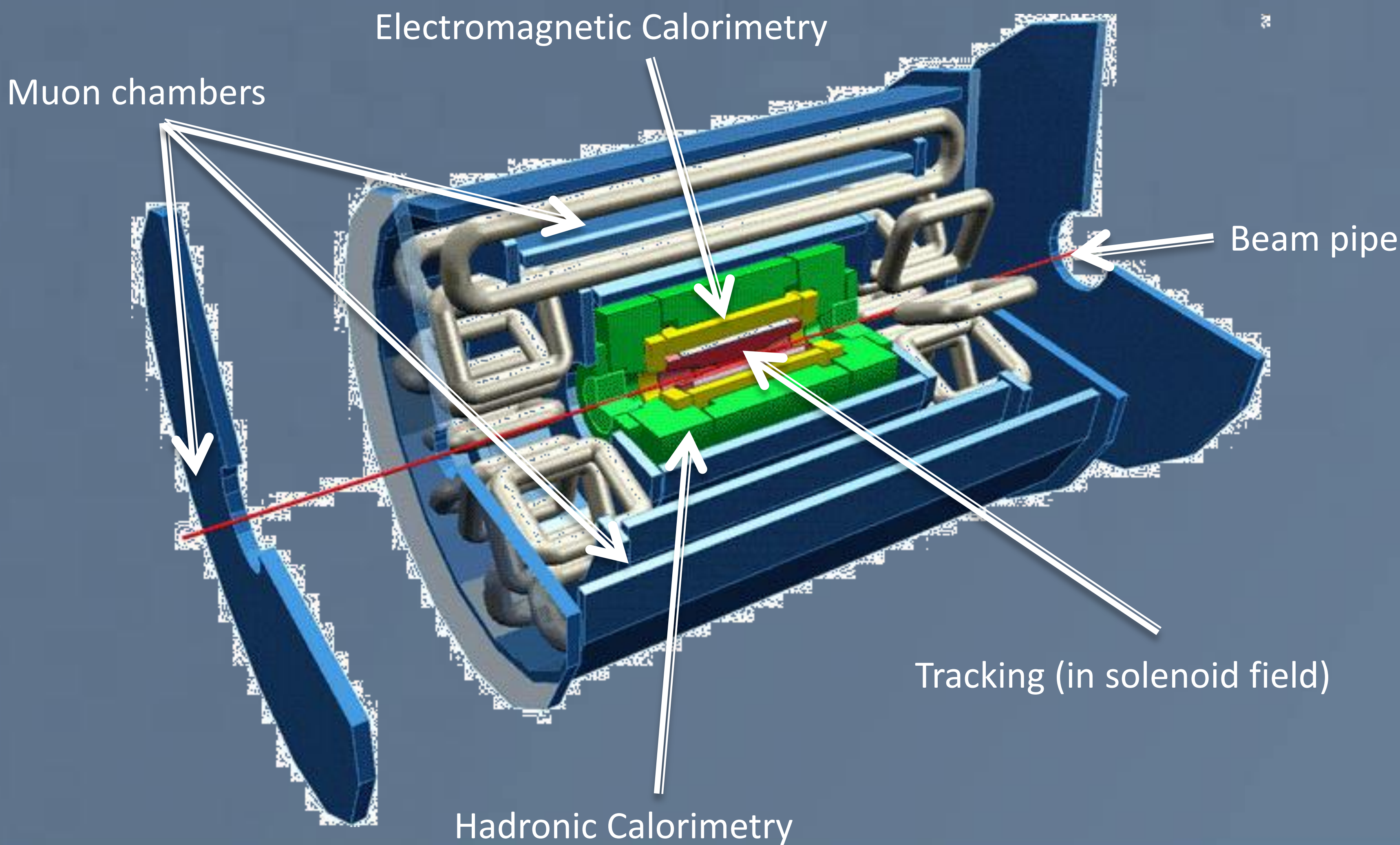


from Wikipedia

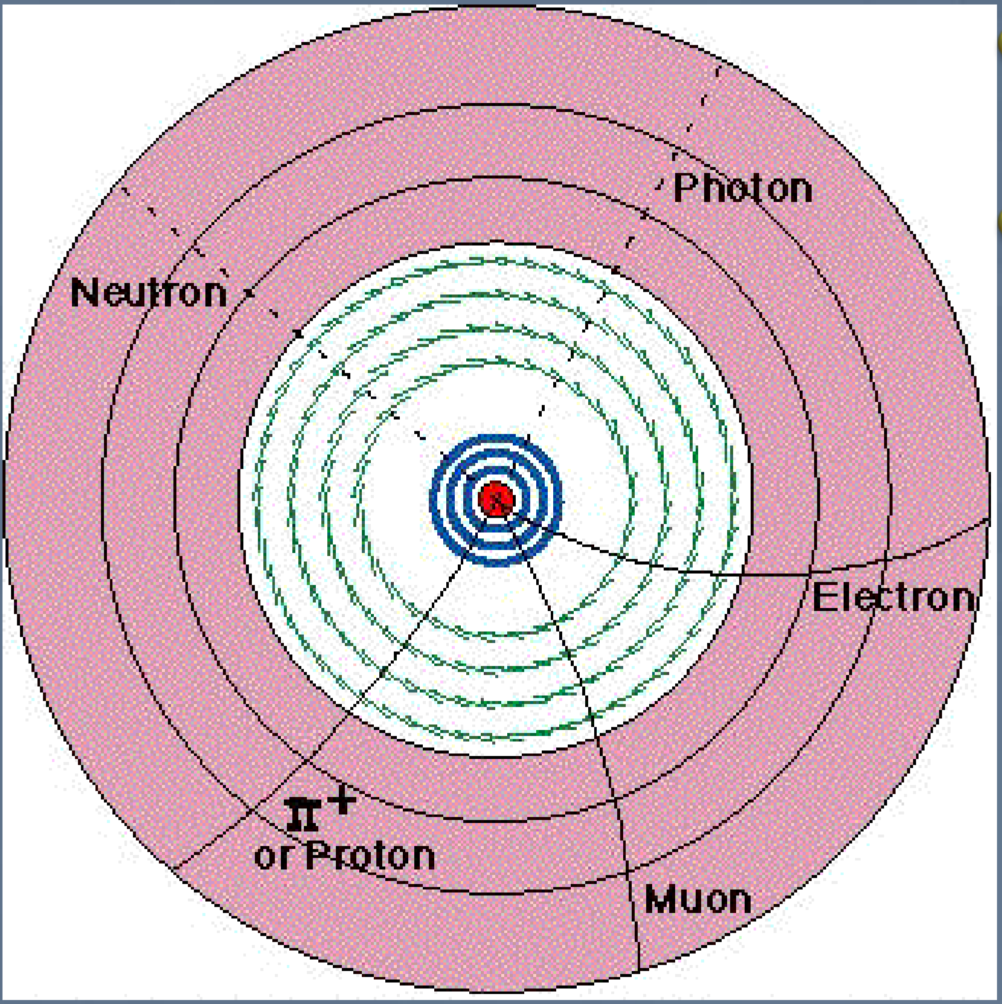
…experiment-data were read



How do we "read" ATLAS data?

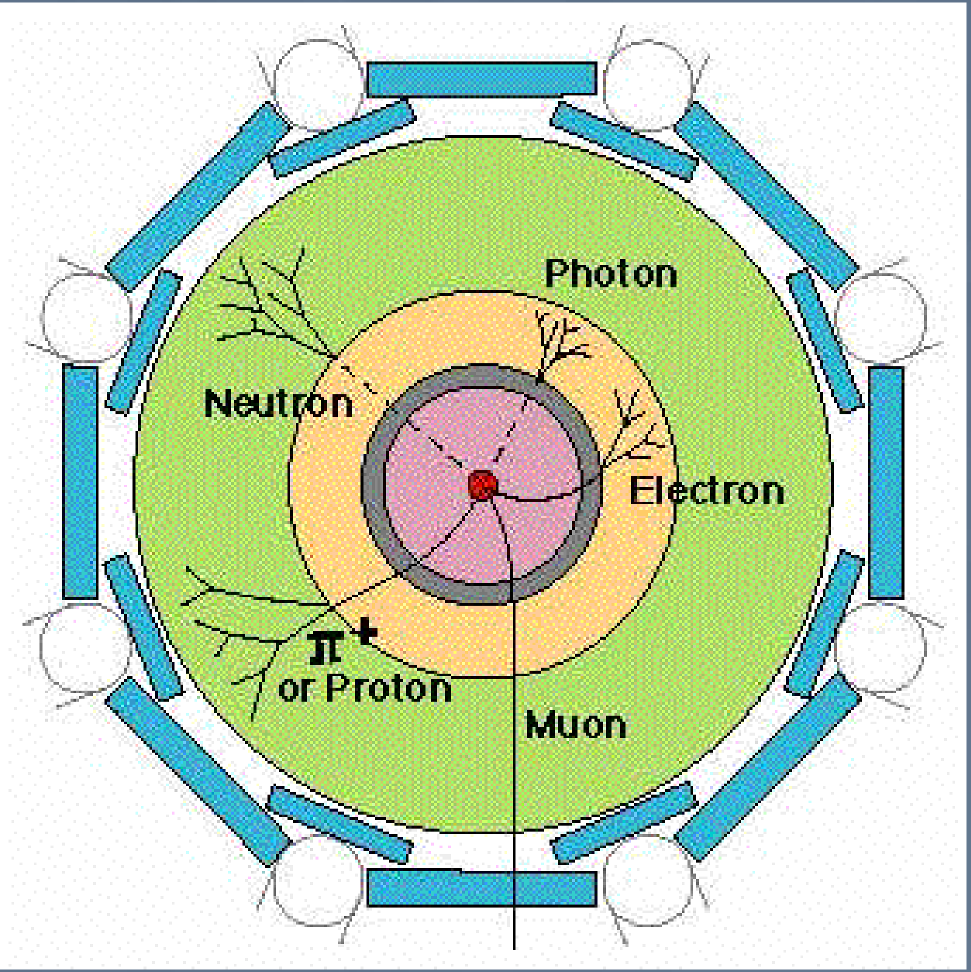


Tracking



- Separate tracks by charge and momentum
- Position measurement layer by layer
  - Inner layers: silicon pixel and strips → presence of hit determines position
  - Outer layers: “straw” drift chambers → need time of hit to determine position

Calorimetry

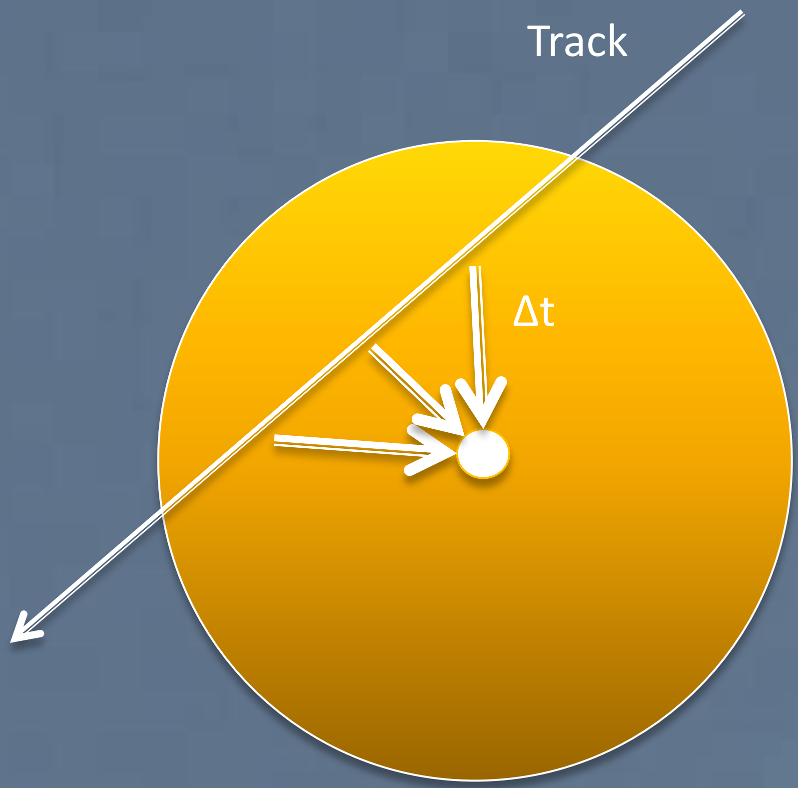


- Particles generate showers in calorimeters
  - Electromagnetic Calorimeter (yellow): Absorbs and measures the energies of all electrons, photons
  - Hadronic Calorimeter (green): Absorbs and measures the energies of hadrons, including protons and neutrons, pions and kaons

→ amplitude measurement required to find deposited charge

- position information provided by segmentation of detector

Muon System



ATLAS Muon drift chambers have a radius of 3 cm and are between 1 and 6 m long

- Electrons formed along the track drift towards the central wire.
- The first electron to reach the high-field region initiates the avalanche, which is used to derive the timing pulse.
- Since the initiation of the avalanche is delayed by the transit time of the charge from the track to the wire, the detection time of the avalanche can be used to determine the radial position(*)

→ ***need fast timing electronics***

(*) Clearly this needs some start of time t=0 (e.g. the beam-crossing)

Many different detectors – very similar goals

Although these various detector systems look very different, at the end it comes down to this:

- Sensors (“detector-elements”) must determine
  1. presence of a signal and/or
  2. magnitude of the signal created and/or
  3. time of arrival of the signal
- Some measurements depend on ***sensitivity***, i.e. detection threshold, e.g.: silicon tracker, to detect presence of a particle in a given electrode
- Others seek to determine a *signal very* ***accurately***, i.e. resolution, e.g. : calorimeter – magnitude of absorbed energy; muon chambers – time measurement yields position

The signal

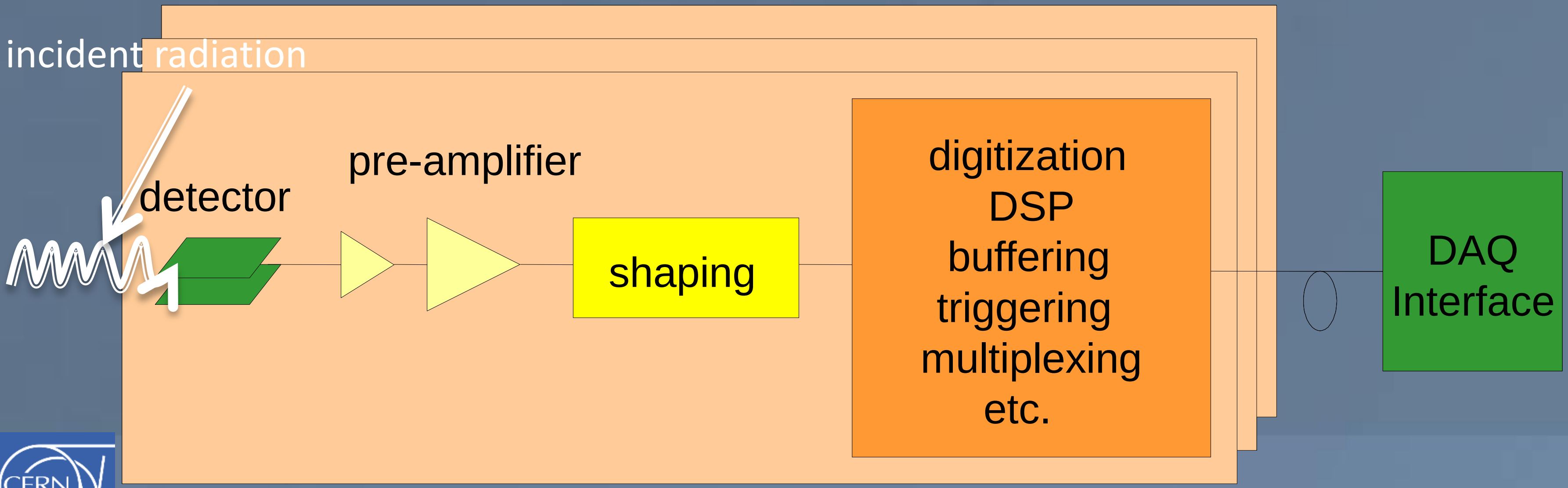
$$S = \frac{E_{absorbed}}{E_{excitation}}$$

- The signal S is usually a small current pulse varying in duration (from ~ 100 ps for a Si sensor to O(10) µs for inorganic scintillators)
- There are many sources of signals. Magnitude of signal depends on deposited signal (energy / charge) and excitation energy

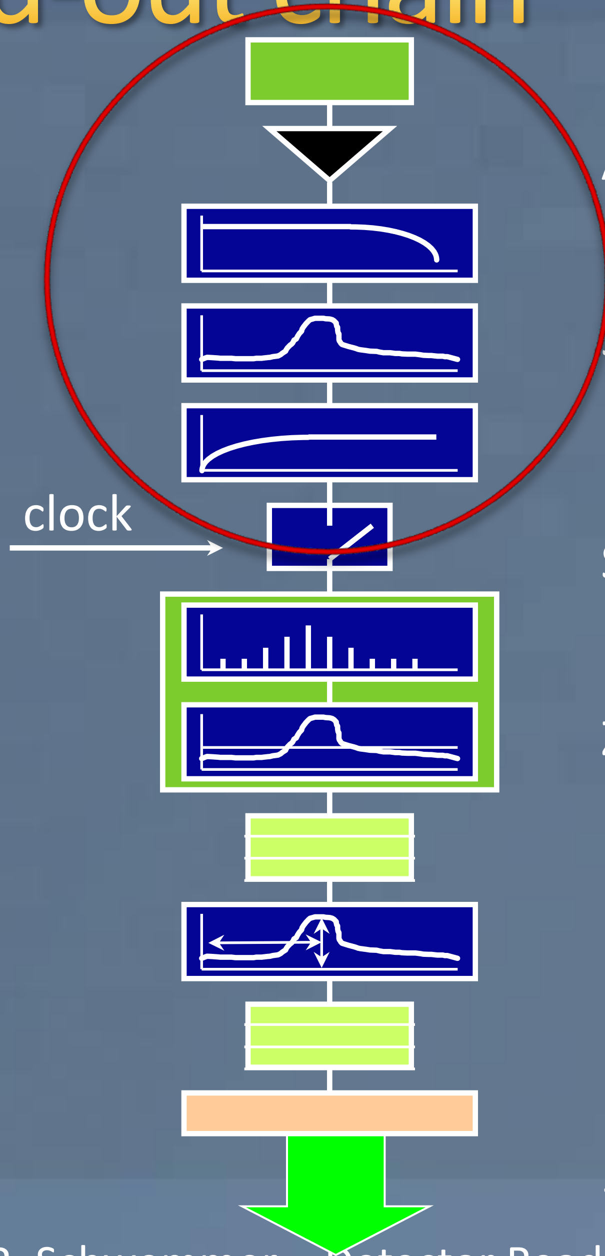
Signal	Physical effect	Excitation energy
Electrical pulse (direct)	Ionization	30 eV for gases 1- 10 eV for semiconductors
Scintillation light	Excitation of optical states	20 – 500 eV
Temperature	Excitation of lattice vibrations	meV

The "front-end" electronics`

- Front-end electronics is the electronics directly connected to the detector (sensitive element)
- Its purpose is to
  - acquire an electrical signal from the detector
  - tailor the response of the system to optimize
    - the minimum detectable signal
    - energy measurement (charge deposit)
    - event rate
    - time of arrival
    - in-sensitivty to sensor pulse shape
  - digitize the signal and store it for further treatment



The read-out chain



Detector / Sensor

Amplifier

Filter

Shaper

Range compression

Sampling

Digital filter

Zero suppression

Buffer

Feature extraction

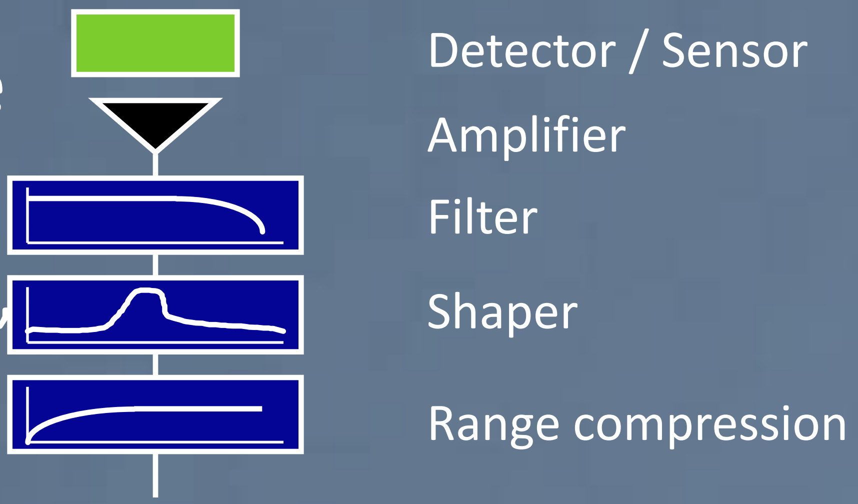
Buffer

Format & Readout

to Data Acquisition System

Acquiring the signal

- Need to match to the specific detector
- Fight noise, radiation, consume (ideally) no power, be weight-less, zero-configuration etc…
- In practice achieved using ASICs made by and for HEP experiments →



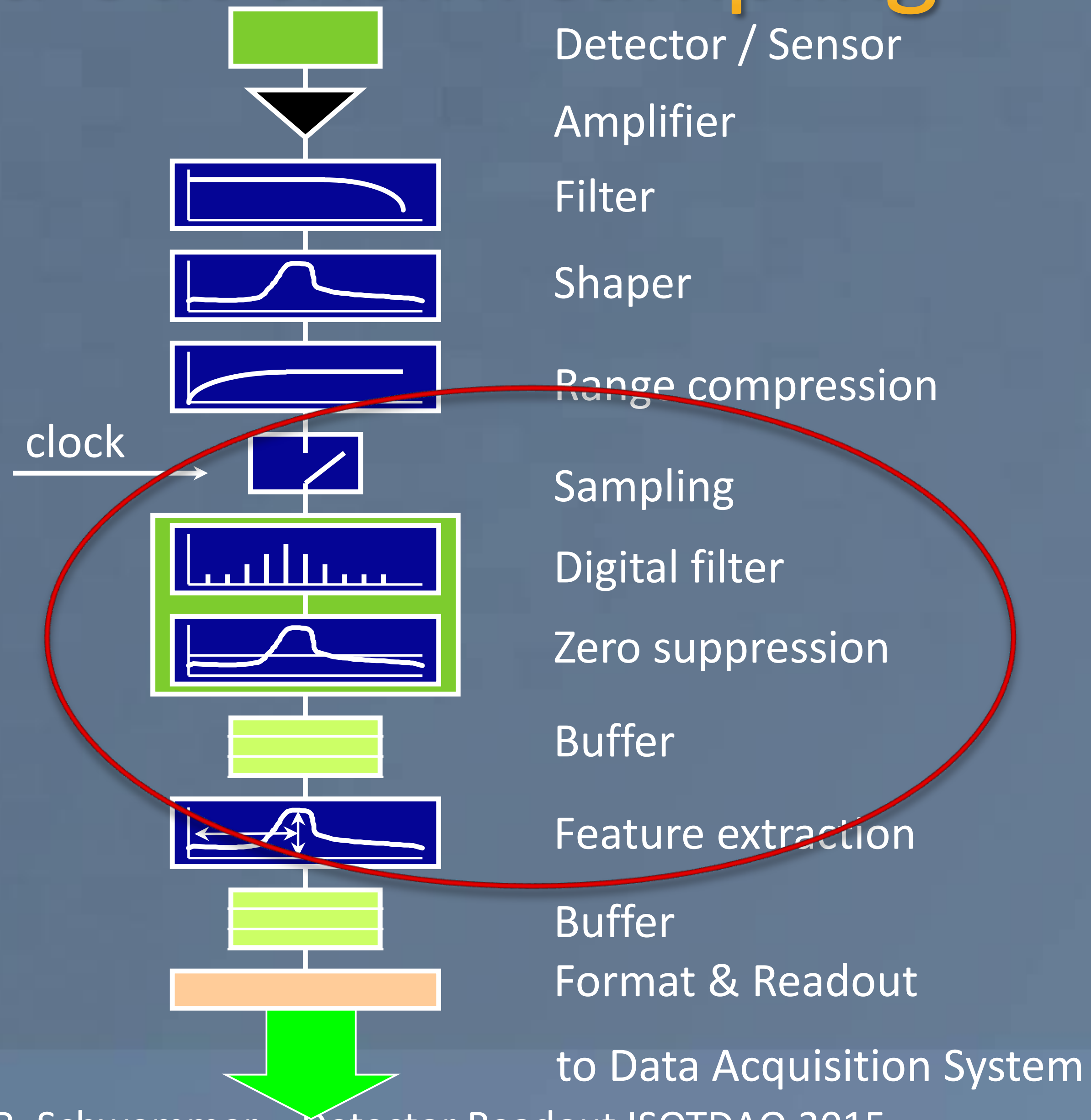
Low level front-end design
Ozgur COBANOGLU

After the black analogue magic

As usual (☺)what you do depends on many factors:

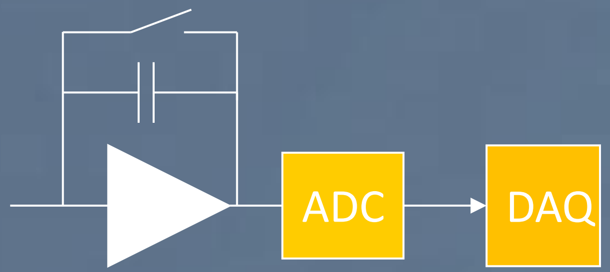
- Number of channels and channel density
- Collision rate and channel occupancies
- *Triggering:* levels, latencies, rates
- Available technology and cost
- What you can/want to do in custom made electronics and what you do in standard computers (computer farms)
- Radiation levels
- Power consumption and related cooling
- Location of digitization
- Given detector technology

The read-out chain: sampling



Single integrator

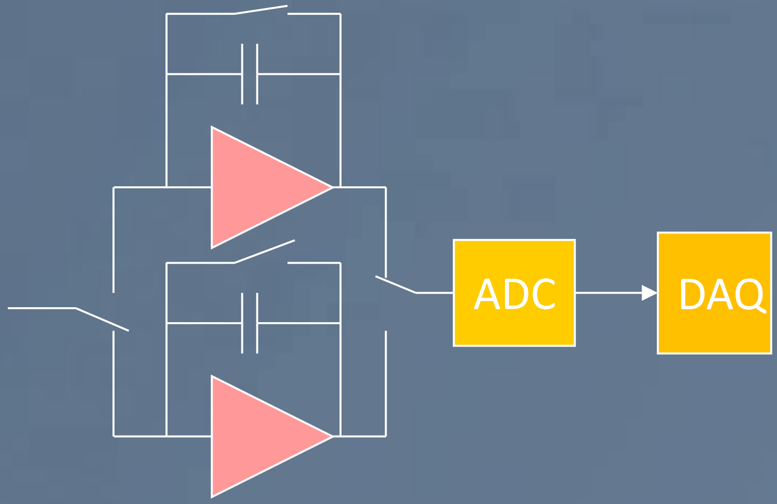
- Simple (only one sample per channel)
- Slow rate (and high precision) experiments
- Long dead time
- Nuclear physics
- Not appropriate for HEP



1. Collect charge from event
2. Convert with ADC
3. Send data to DAQ

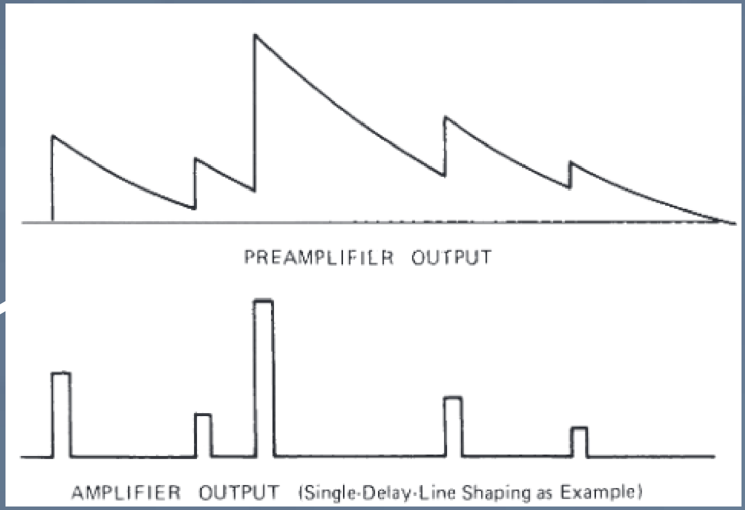
Double buffered

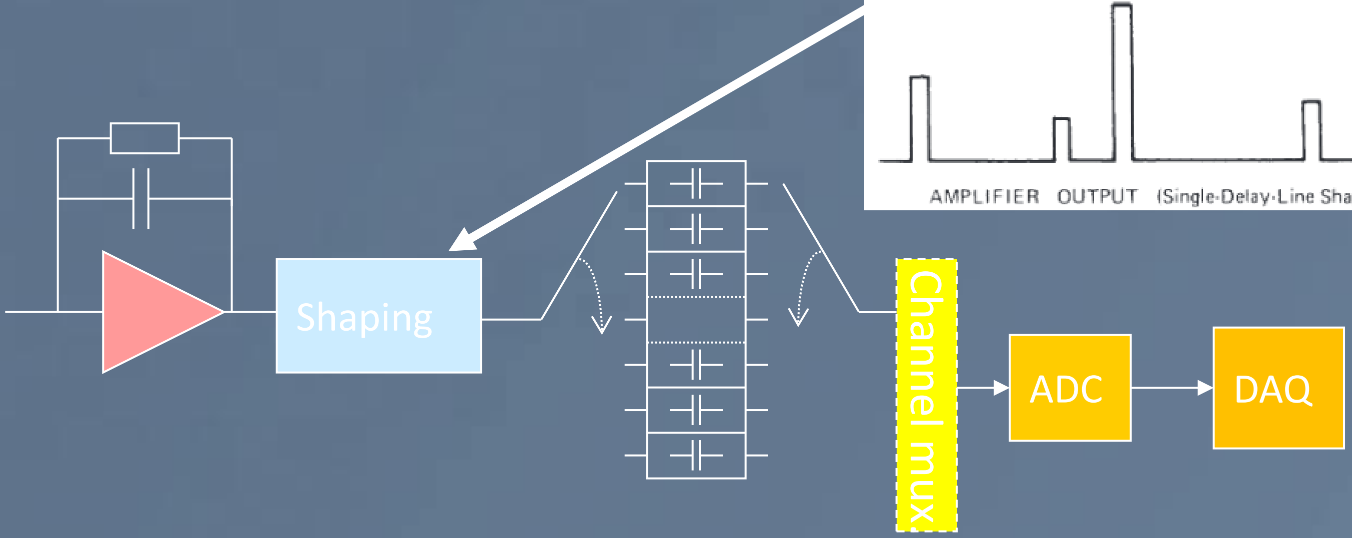
- Use a second integrator while the first is readout and reset
- Decreases dead time significantly
- Still for low rates



Multiple event buffers

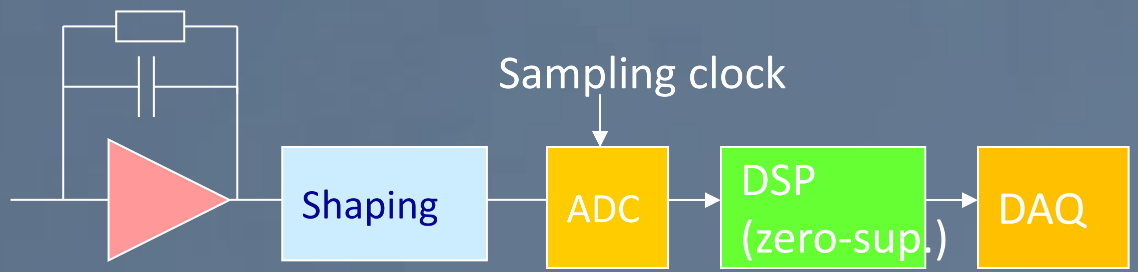
- Good for experiments with short spills and large spacing between spills (e.g. fixed target experiment at SPS)
- Fill up event buffers during spill (high rate)
- Readout between spills (low rate)
- ADC can possibly be shared across channels
- Buffering can also be done digitally (in RAM)





Constantly Sampled Readout

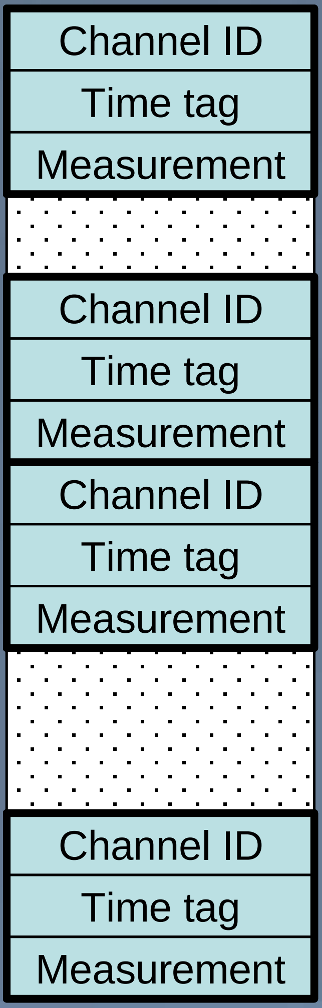
- Needed for high rate experiments with signal pileup
- Shapers and not switched integrators
- Allows digital signal processing in its traditional form (constantly sampled data stream)
- Output rate may be far to high for what following DAQ system can handle



- With local zero-suppression this is suitable for future high rate experiments (LHC, CLIC, FAIR)

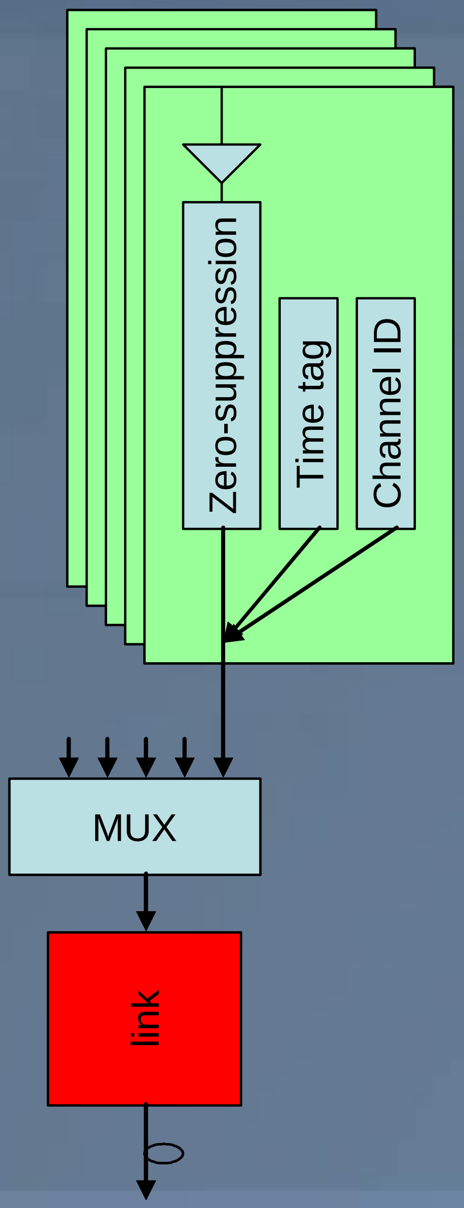
Excursion: zero-suppression

- Why spend bandwidth sending data that is zero for the majority of the time?
- Perform zero-suppression and only send data with non-zero content
  - Identify the data with a channel number and/or a time-stamp
  - We do not want to loose information of interest so this must be done with great care taking into account pedestals, baseline variations, common mode, noise, etc.
  - Not worth it for occupancies above ~10%
- Alternative: data compression
  - Huffman encoding and alike (slow, power-intensive)



TANSTAFL

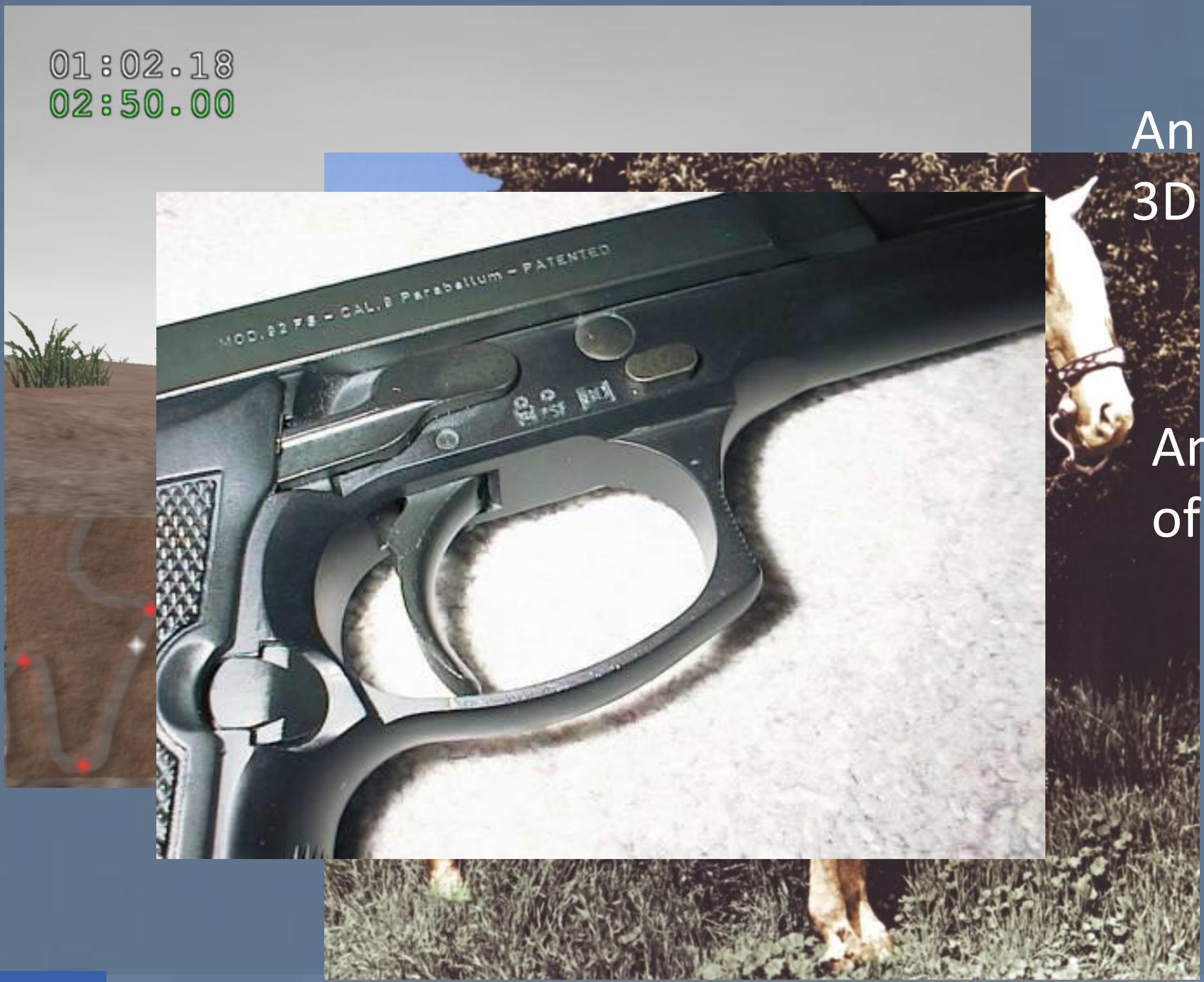
- (There Aint No Such Thing As A Free Lunch)
- Data rates fluctuate all the time and we have to fit this into links with a given bandwidth
- It is difficult to define a priori what to consider a zero channel
- Not any more event synchronous
- Complicated buffer handling (overflows)
- Before an experiment is built and running it is very difficult to give reliable estimates of data rates needed (background, new physics, etc.)
- → "They" will always want to read it out non-zero suppressed



Trigger (Sneak Preview)

SPOILER ALERT

What is a trigger?



An open-source 3D rally game?

An important part of a Beretta?

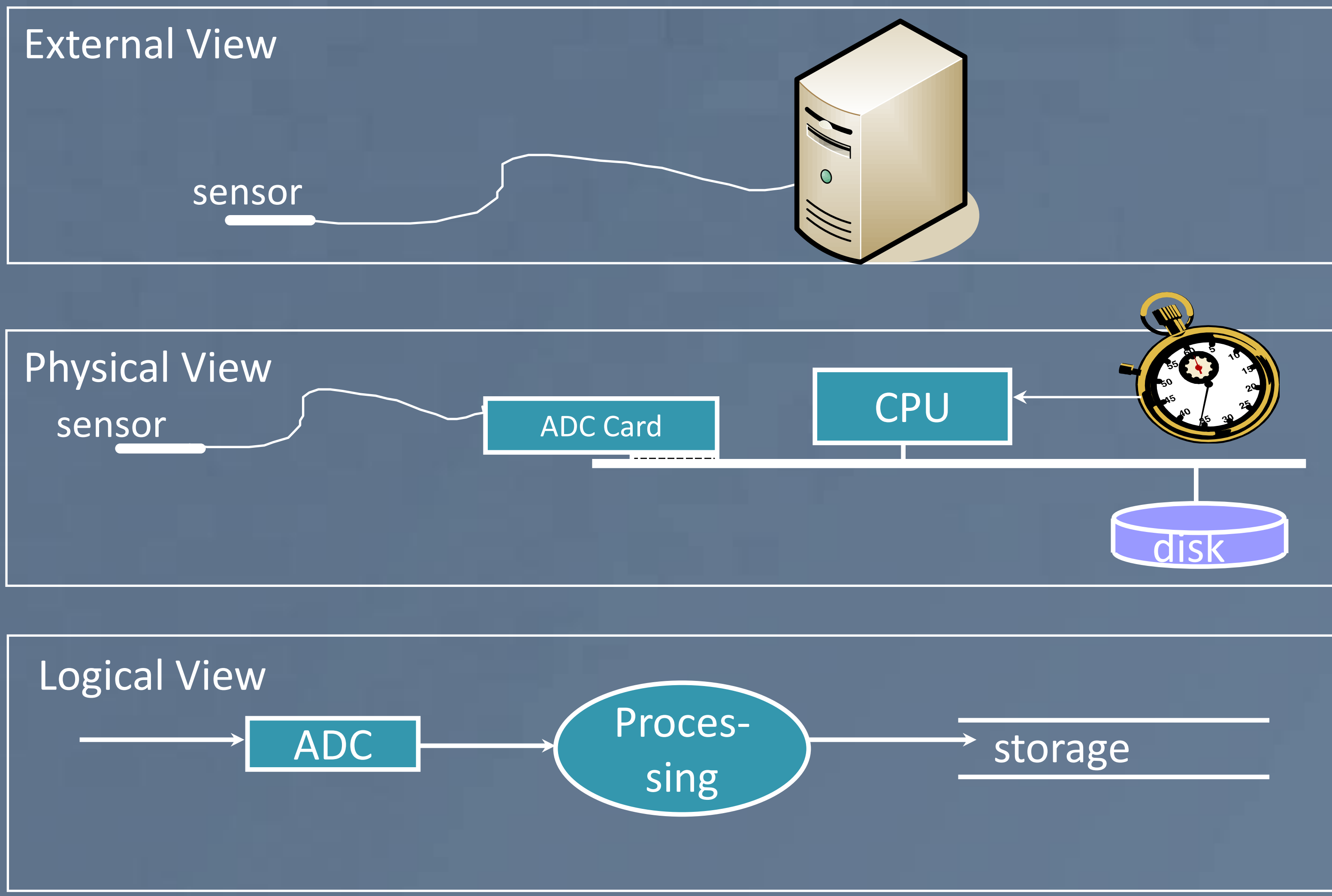
The most famous horse in movie history?

What is a trigger?

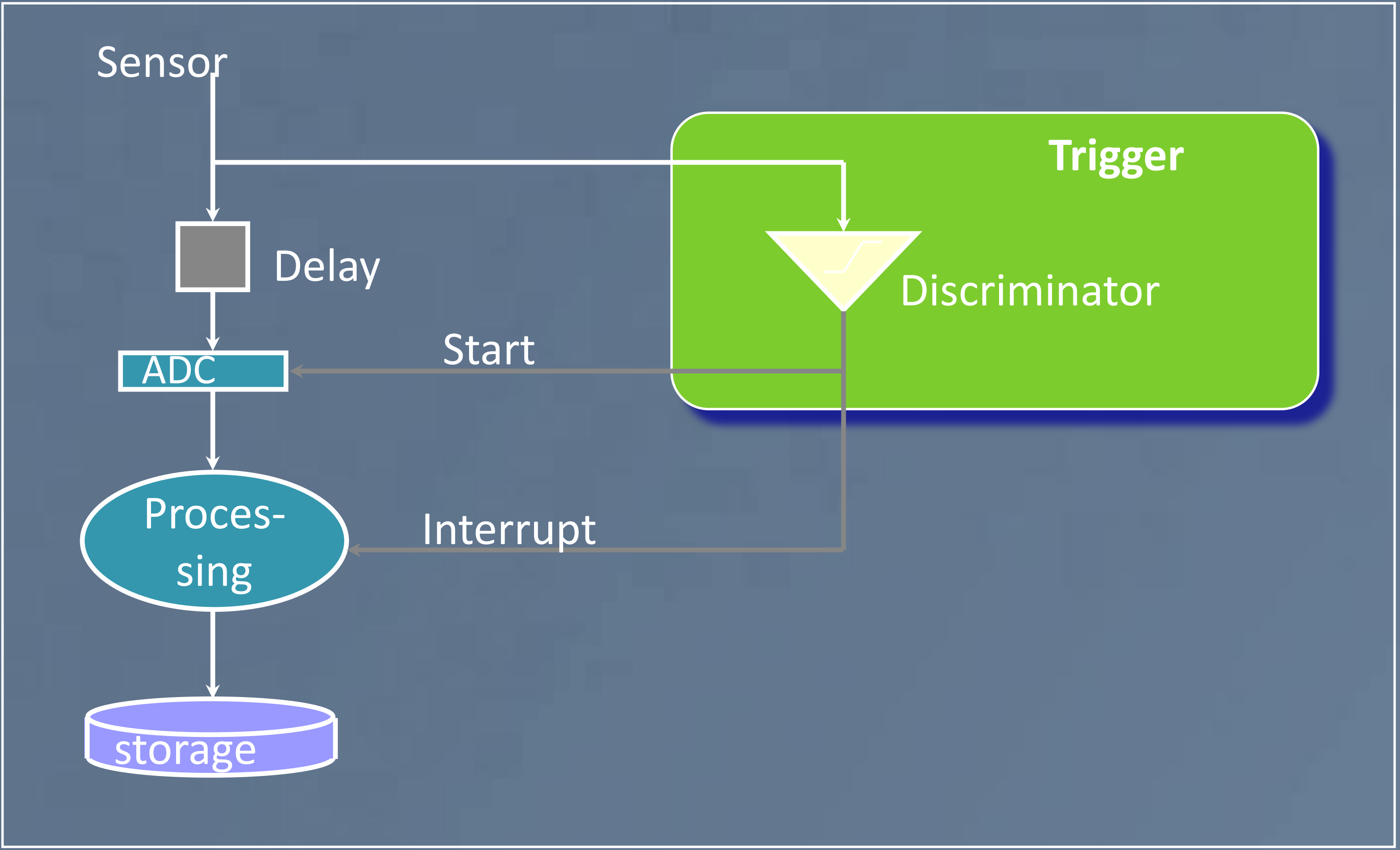
Wikipedia: "**A trigger is a system that uses simple criteria to rapidly decide which events in a particle detector to keep when only a small fraction of the total can be recorded.** "

- Simple
- Rapid
- Selective
- When only a small fraction can be recorded

Trivial DAQ


External View
sensor
Physical View
sensor
ADC Card
CPU
disk
Logical View
ADC
Proces-
sing
storage

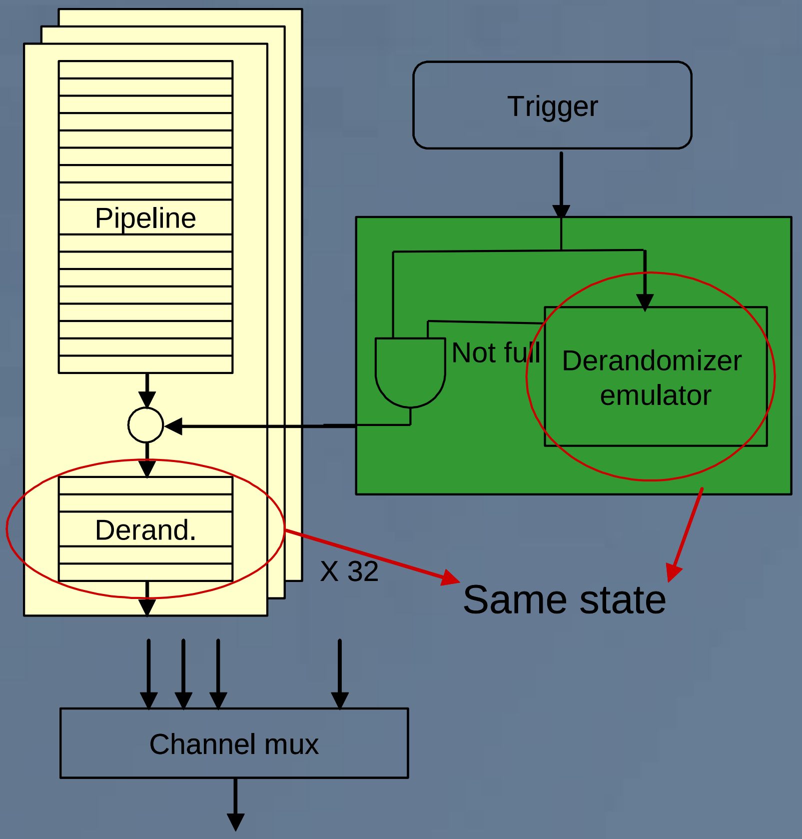
Trivial DAQ with a real trigger



What if a trigger is produced when the *ADC* or *processing* is busy?

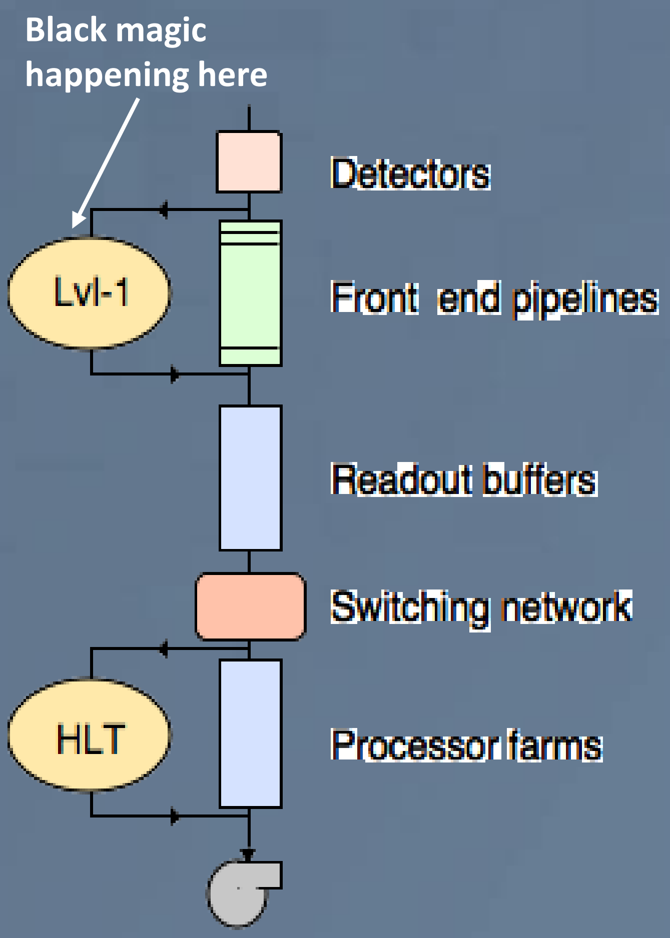
Trigger rate control

- Trigger rate determined by physics parameters used in trigger system:
  1 kHz – 1MHz for LHC experiments
  - The lower rate after the trigger allows sharing resources across channels (e.g. ADC and readout links)
- Triggers will be of random nature i.e. follow a Poisson distribution → a burst of triggers can occur within a short time window so some kind of rate control/spacing is needed
  - Minimum spacing between trigger accepts → dead-time
  - Maximum number of triggers within a given time window
- Derandomizer buffers needed in front-ends to handle this
  - Size and readout speed of this determines effective trigger rate



Trigger for LHC

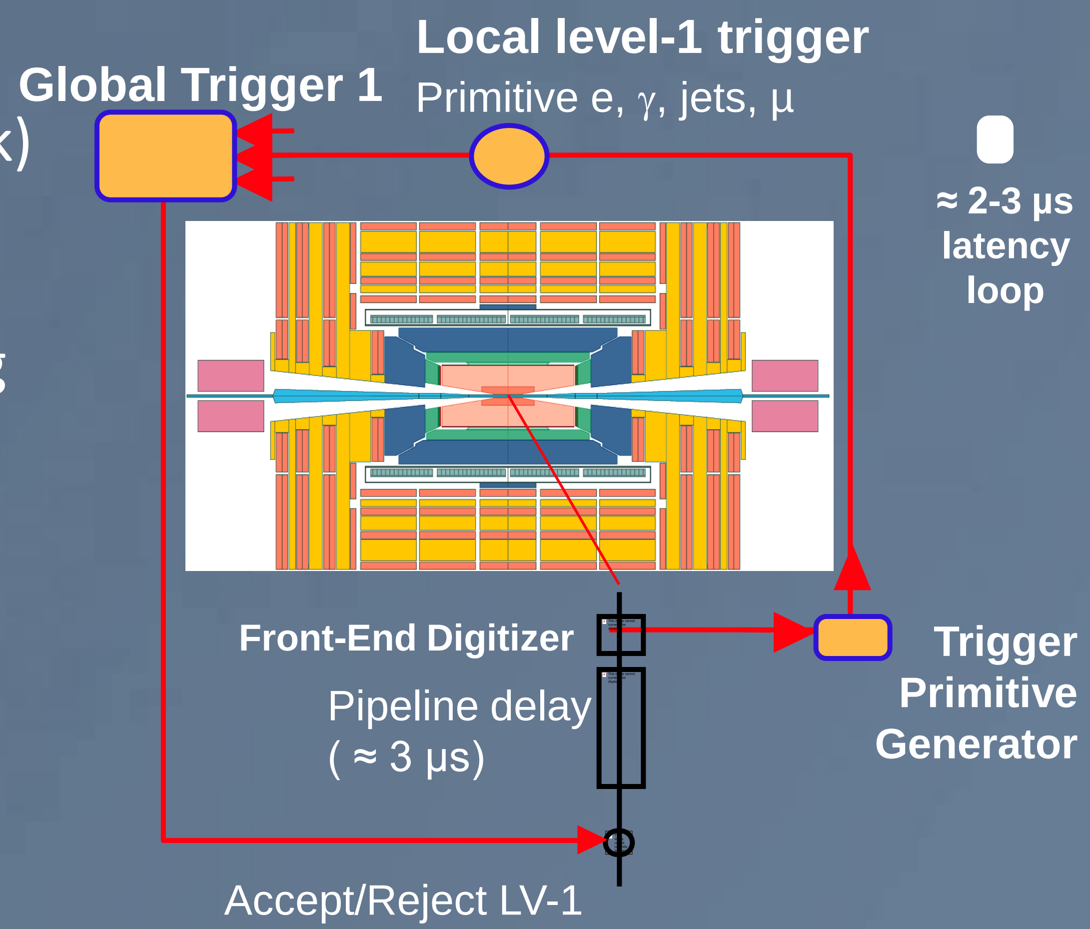
- No ((currently) affordable) DAQ system could read out $O(10^7)$ channels at 40 MHz → 400 TBit/s to read out – even assuming binary channels!
- What's worse: most of these millions of events per second are totally uninteresting: one Higgs event every 0.02 seconds
- A *first level trigger (Level-1, L1)* must somehow* select the more interesting events and tell us which ones to deal with any further



(*) See any lecture on LHC trigger systems

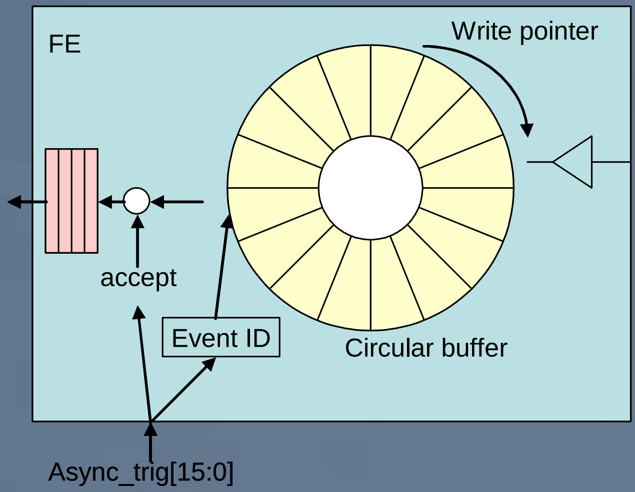
Distributing the Trigger

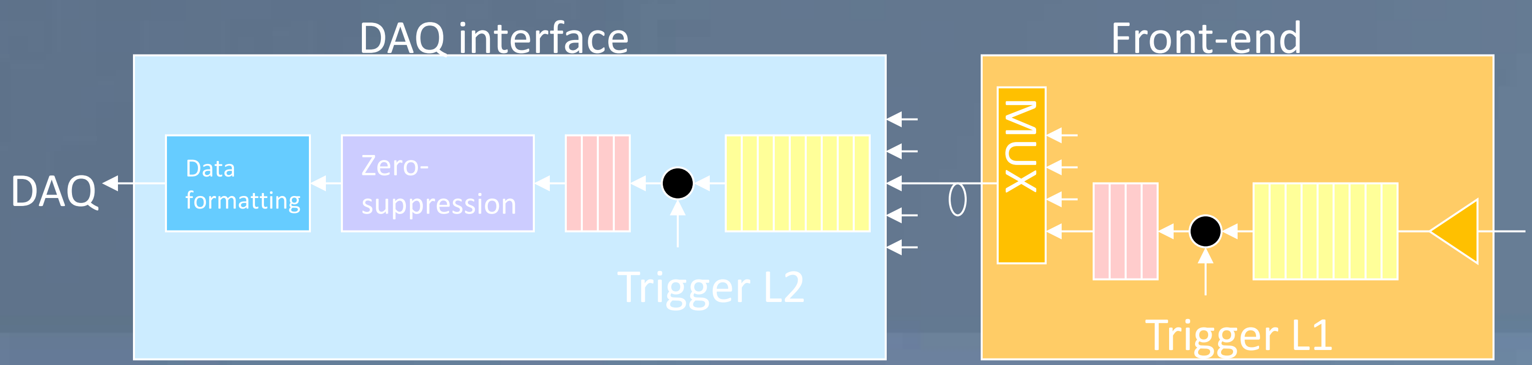
- Assuming now that a magic box tells for each bunch crossing (clock-tick) yes or no
- This decision has to be brought for each crossing to all the detector front-end electronics elements so that they can send of their data or discard it
- Use the same Timing and Trigger Control (TTC) system as for the clock distribution



Multilevel triggering

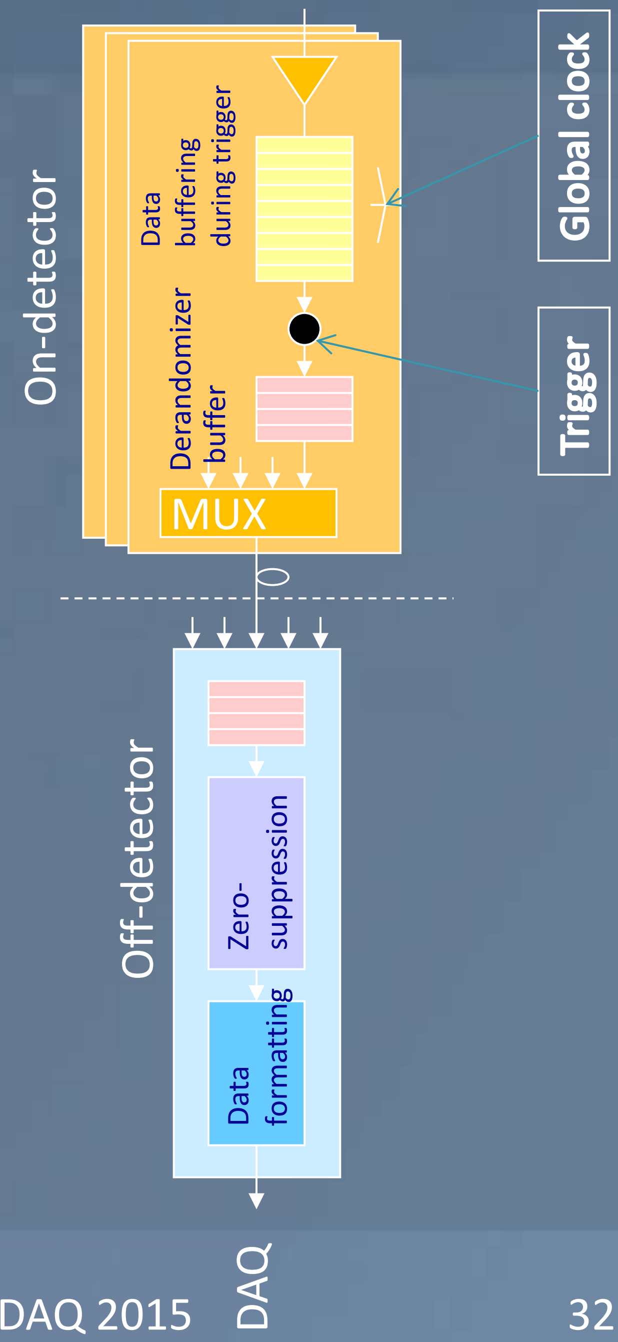
- First level triggering.
  - Hardwired trigger system to make trigger decision with short latency.
  - Constant latency buffers in the front-ends
- Second level triggering in DAQ interface
  - Processor based (standard CPU's or dedicated custom/DSP/FPGA processing)
  - FIFO buffers with each event getting accept/reject in sequential order
  - Circular buffer using event ID to extracted accepted events
    - Non accepted events stays and gets overwritten by new events
- High level triggering in the DAQ systems made with farms of CPU's: hundreds – thousands.
  (separate lectures on this)



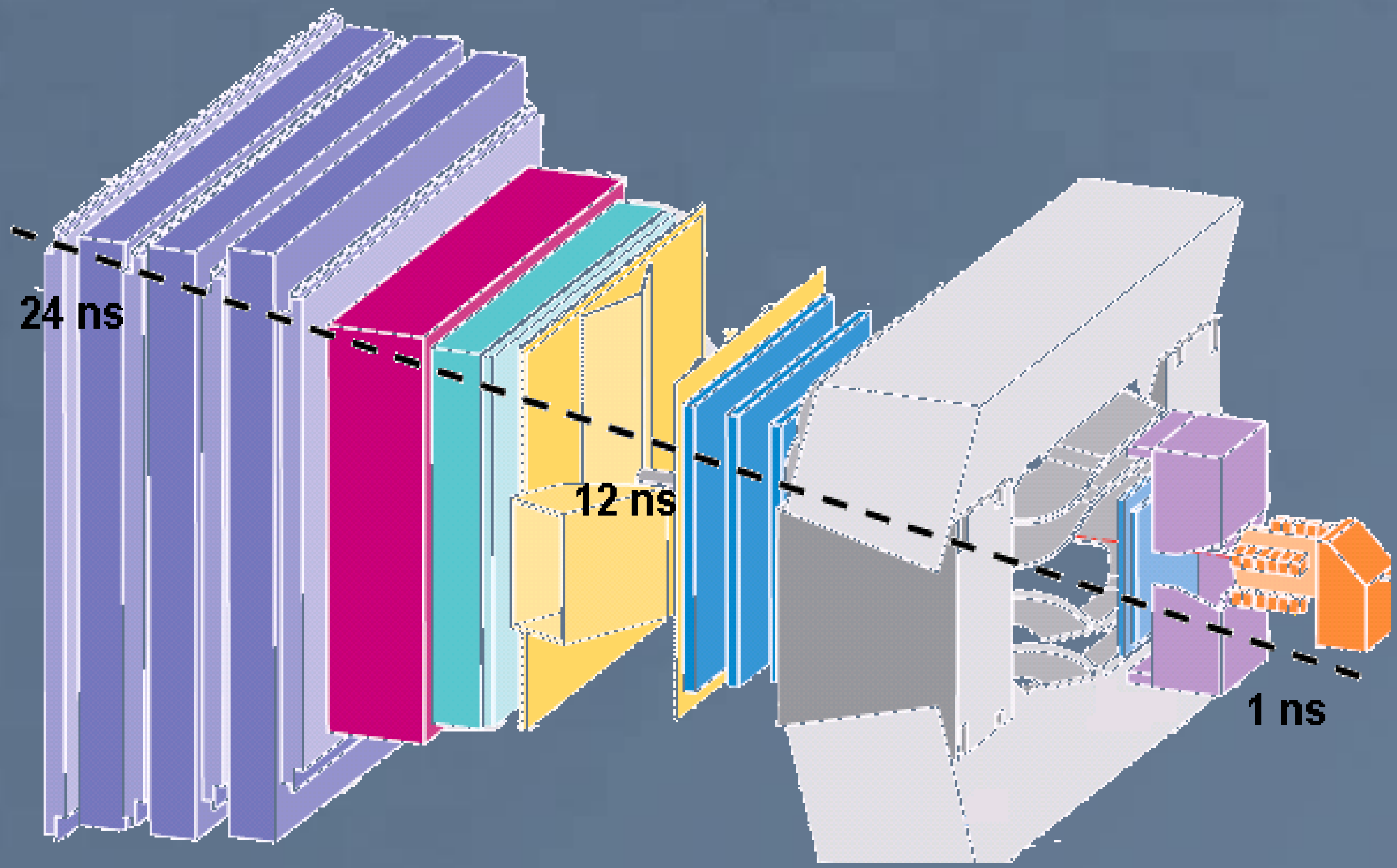


Synchronous Readout

- All channels are doing the same "thing" at the same time
- Synchronous to a global clock (bunch crossing clock)
- Data-rate on each link is identical and depends only on *trigger -rate*
- On-detector buffers (*de-randomizers*) are of same size and their occupancy ("how full they are") depends only on the *trigger-rate*
- ☹ Lots of bandwidth wasted for zero's
  - Price of links determine if one can afford this
- ☺ No problems if occupancy of detectors or noise higher than expected
  - But there are other problems related to this: spill over, saturation of detector, etc.

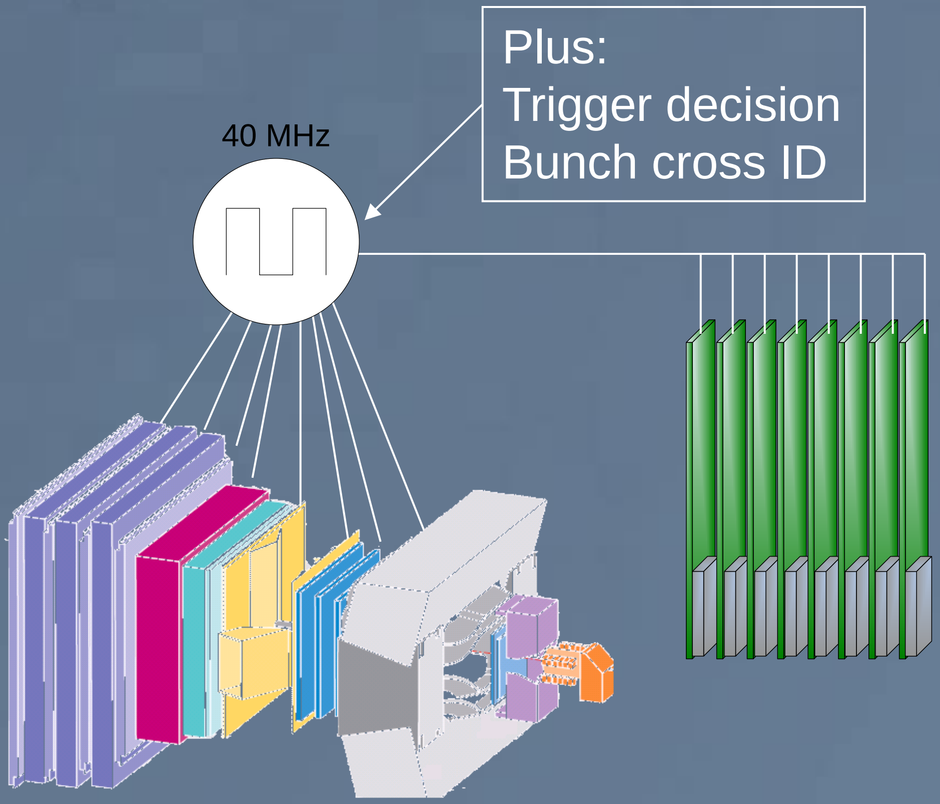


Time Alignment



Distributing Synchronous Signals @ the LHC

- An *event* is a snapshot of the values of all detector front-end electronics elements, which have their value caused by the same collision
- A common clock signal must be provided to all detector elements
  - Since c is constant, the detectors are large and the electronics are fast, the detector elements must be carefully time-aligned
- Common system for all LHC experiments TTC based on radiation-hard opto-electronics

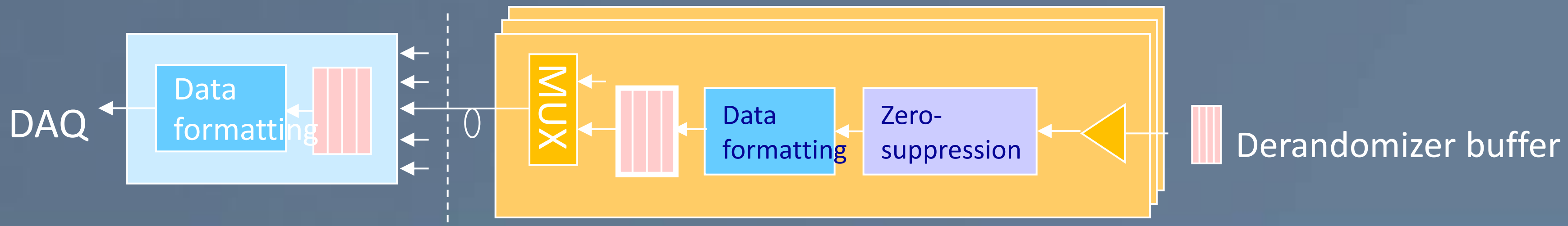


Synchronicity at the LHC

- N (channels) ~ O(10^7); Interactions every 25 ns
  - need huge number of connections
  - Need to synchronize detector elements to (better than) 25 ns
- In some cases: detector signal/time of flight > 25 ns
  - integrate more than one bunch crossing's worth of information
  - need to identify bunch crossing...
- It's On-Line (cannot go back and recover events)
  - need to monitor selection –
  - need very good control over all conditions

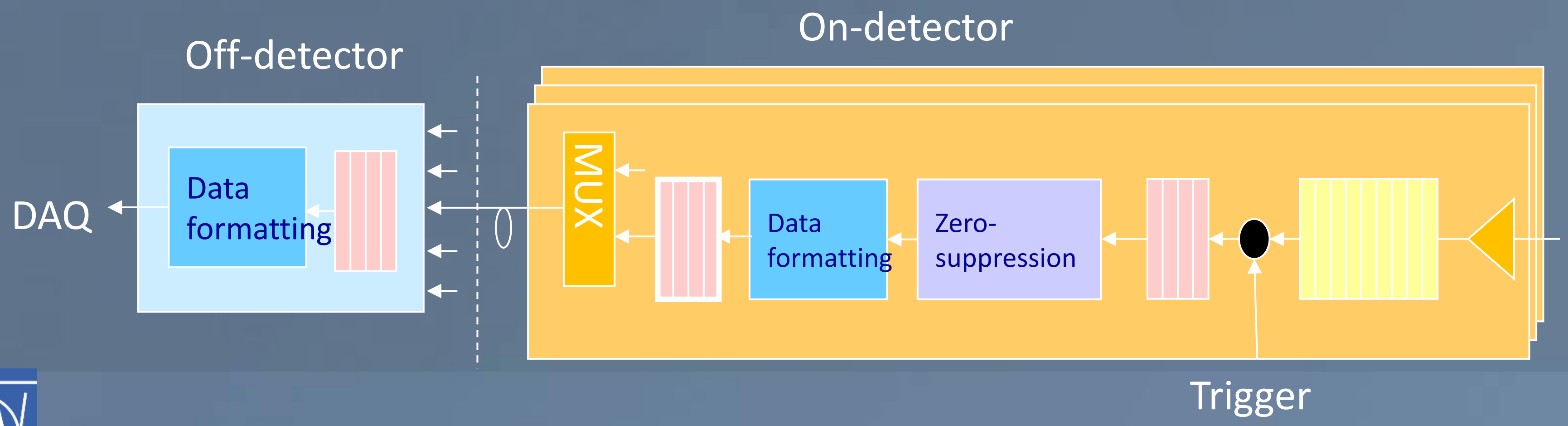
Asynchronous Readout

- Remove zeros on the detector itself
  - Lower average bandwidth needed for readout links Especially interesting for low occupancy detectors
- Each channel "lives a life of its own" with unpredictable buffer occupancies and data is sent whenever ready (asynchronous)
- In case of buffer-overflow a truncation policy is needed → BIAS!!
  - Detectors themselves do not have 100% detection efficiency either.
  - Requires sufficiently large local buffers to assure that data is not lost too often (Channel occupancies can be quite non uniform across a detector with same front-end electronics)

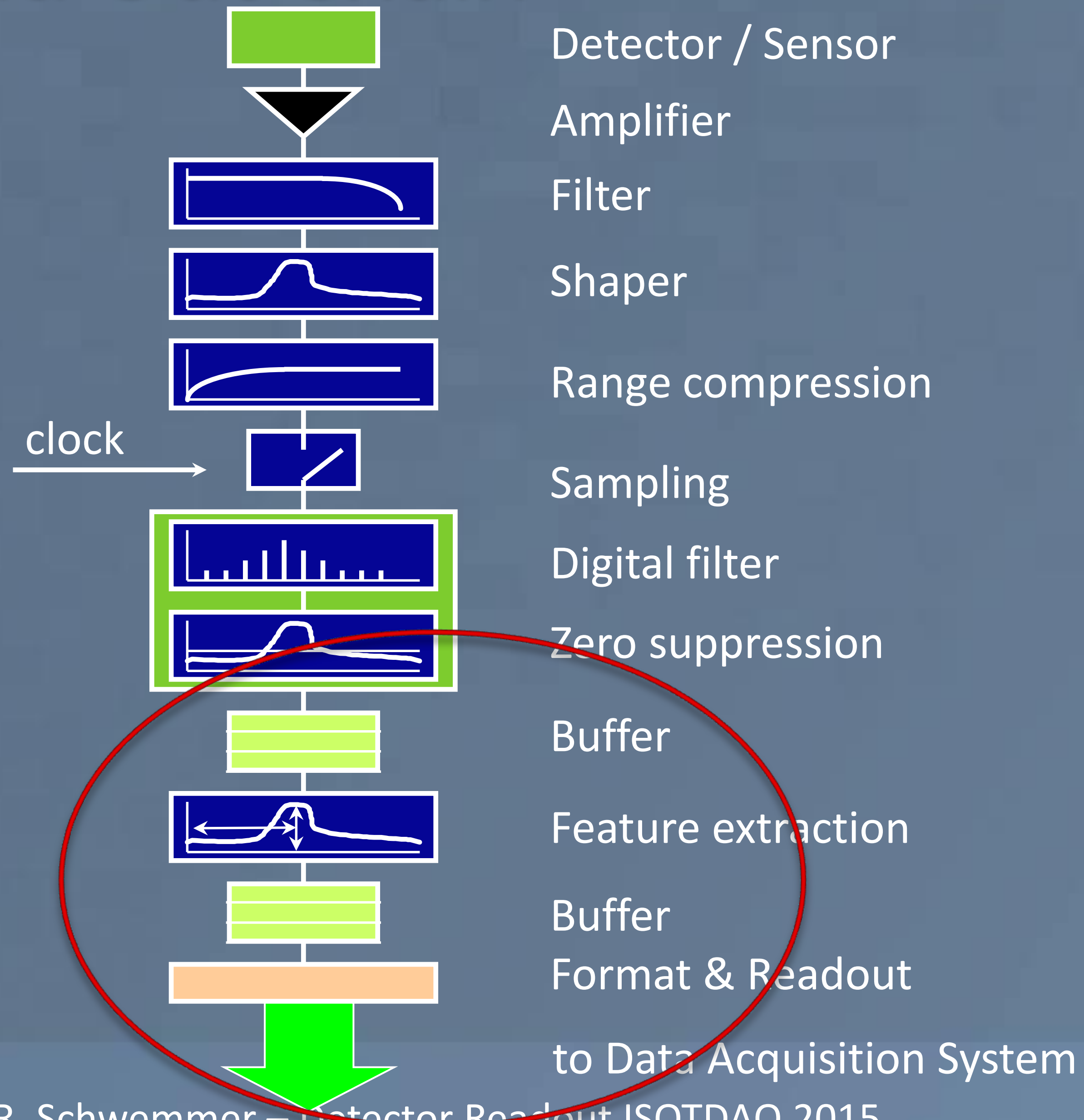


Asynchronous Readout - continued

- DAQ must be able to handle this (buffering!)
- Async. readout of detectors in LHC: ATLAS and CMS muon drift tube detectors, ATLAS and CMS pixel detectors, ATLAS SCT, several ALICE detectors with a relatively low trigger rate (few kHz).



The Read Out Chain

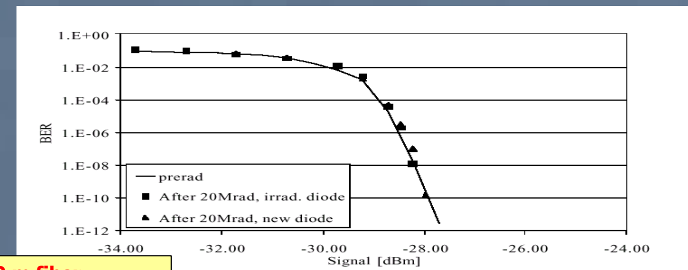


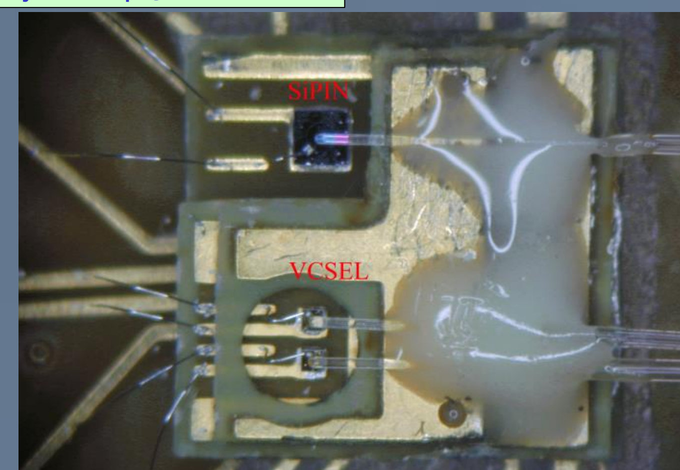
To the DAQ: The Readout Link

- Large amount of data to bring out of detector
  - Large quantity: ~ 100k links in large experiments (cost!)
  - High speed: Gbits/s
- Point to point unidirectional
- Transmitter side has specific constraints
  - Radiation
  - Magnetic fields
  - Power/cooling
  - Minimum size and mass
  - Must collect data from one or several front-end chips
- Receiver side can be commercially available module/components (use of standard link protocols whenever possible, e.g. 64/66 bit encoding like in Ethernet)

Digital optical links

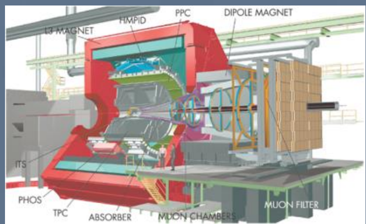
- High speed: 1 Ghz – 10 GHz – 40 GHz
- Extensively used in telecommunications (expensive) and in computing (“cheap”)
- Encoding
  - Inclusion of clock for receiver PLL’s
  - DC balanced
  - Special synchronization characters
  - Error detection and or correction
- Reliability and error rates strongly depending on received optical power and timing jitter
- Multiple serializers and deserializers directly available in modern high end FPGA’s.
- Used everywhere in the LHC experiments, will also be standard for future upgrades (Versatile Link / GBT and similar)





Readout Links of LHC Experiments

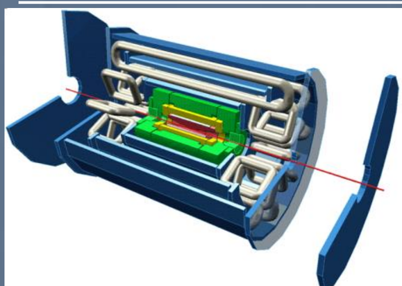
Flow Control



DDL

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

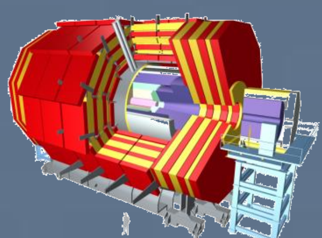
Flow Control: yes



SLINK

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

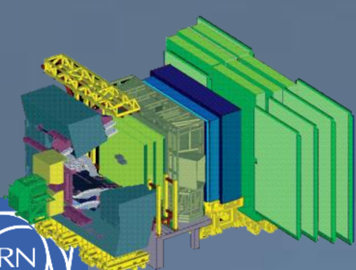
Flow Control: yes



SLINK 64

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

Flow Control: yes



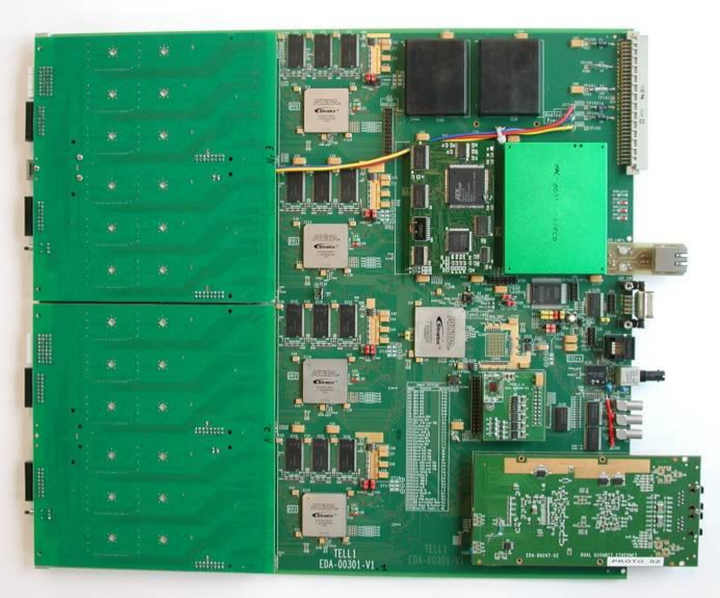
Glink (GOL)

Optical 200 MB/s ≈ 4000 links
Receiver card interfaces to custom-built Ethernet NIC (4 x 1 Gbit/s over copper)

Flow Control: (no)

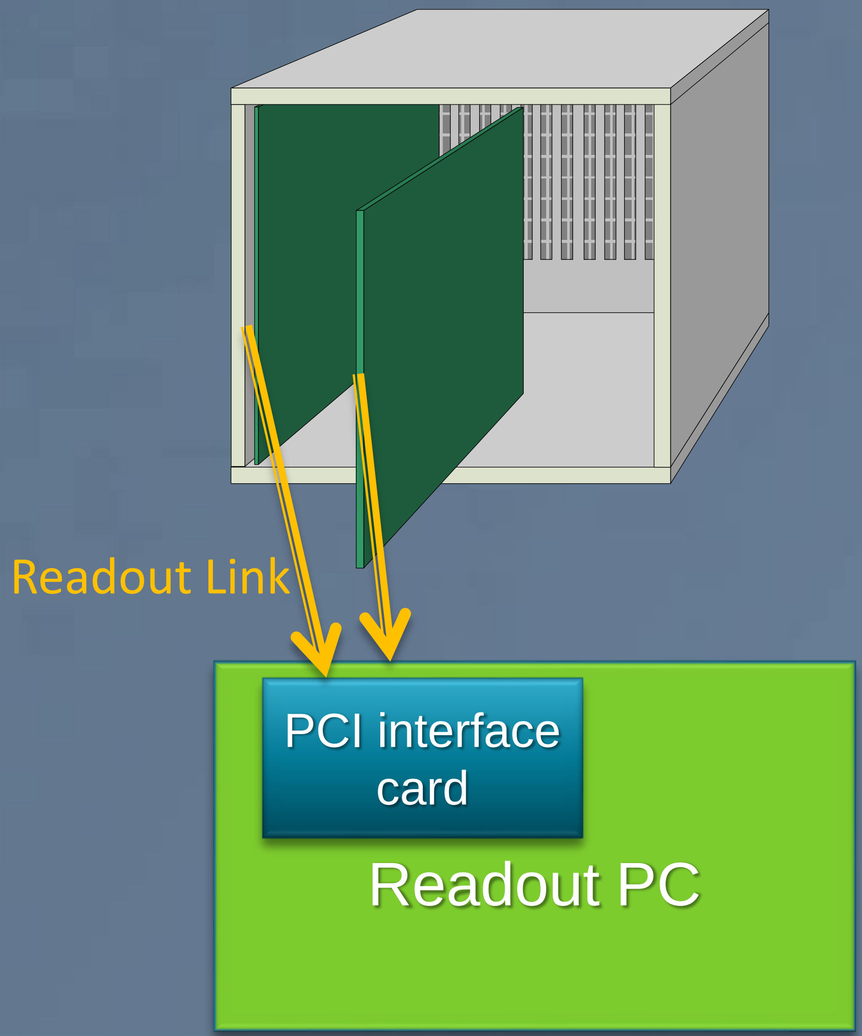
DAQ interfaces / Readout Boards

- Front-end data reception
  - Receive optical links from multiple front-ends: 24 - 96
  - Located outside radiation
- Event checking
  - Verify that data received is correct
  - Verify correct synchronization of front-ends
- Extended digital signal processing to extract information of interest and minimize data volume
- Event merging/building
  - Build consistent data structures from the individual data sources so it can be efficiently sent to DAQ CPU farm and processed efficiently without wasting time reformatting data on CPU.
  - Requires significant data buffering



DAQ interfaces / Readout Boards 2

- High level of programmability needed
- Send data to CPU farm at a rate that can be correctly handled by farm
  - 1 Gbits/s Ethernet (next is 100Gbits/s)
  - In house link with PCI interface: S-link
- Requires a lot of fast digital processing and data buffering: **FPGA's**, DSP's, embedded CPU
- Use of ASIC's not justified
- Complicated modules that are only half made when the hardware is there: FPGA firmware (from HDL), DSP code, on-board CPU software, etc.
- PC provides standardised interface and buffering!

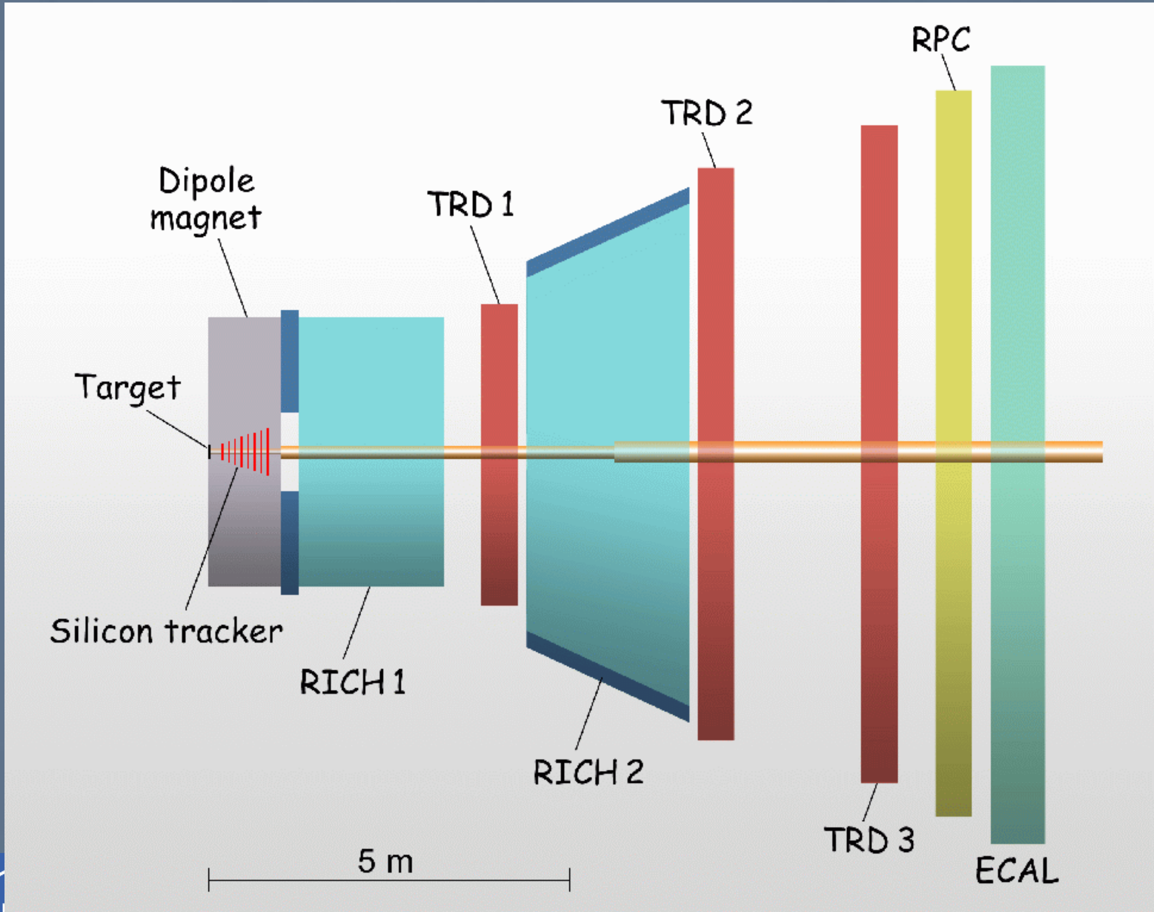


The Undiscovered Country

…("the future of detector readout")

Compressed Baryonic Matter (CBM)

- Heavy Ion experiment planned at future FAIR facility at GSI (Darmstadt)
- Timescale: ~2018



Detector Elements

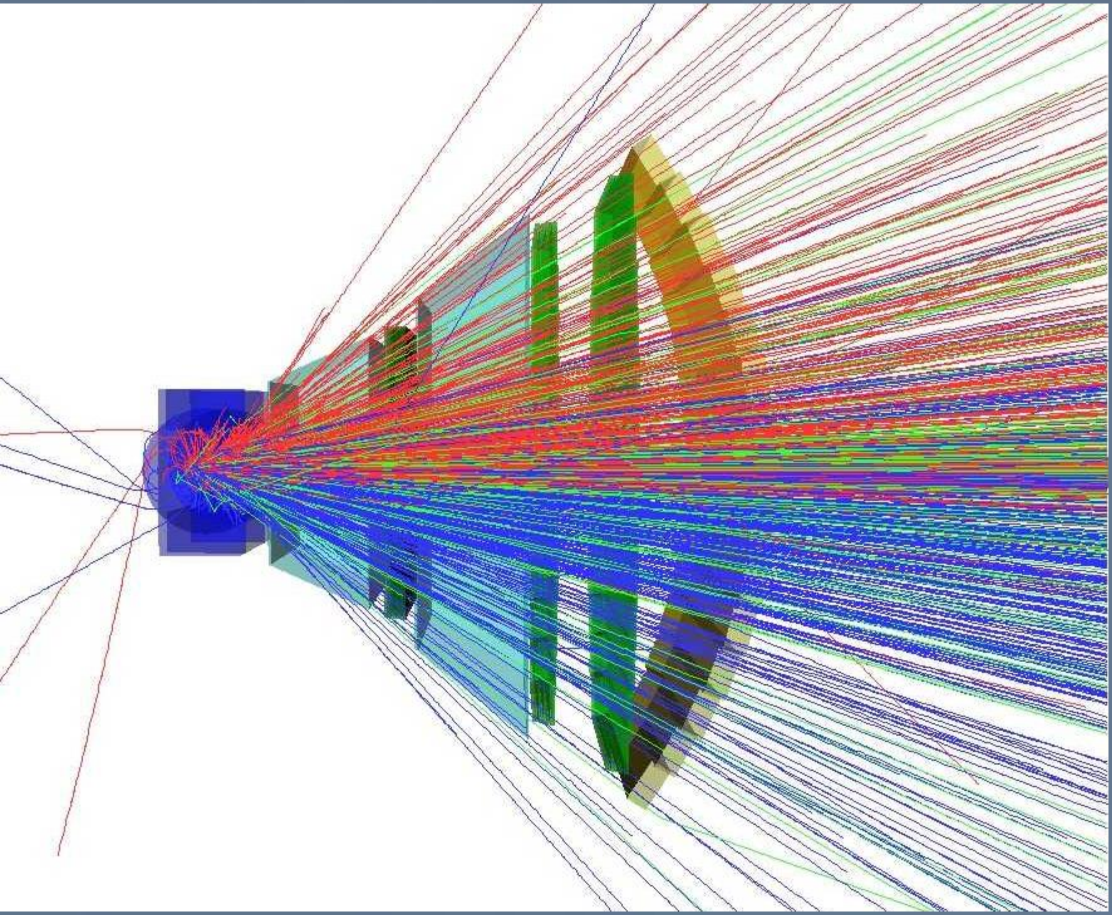
- **Si for Tracking**
- **RICH and TRDs for Particle identification**
- **RPCs for ToF measurement**
- **ECal for Electromagnetic Calorimetry**

Average Multiplicities:

160 p
400 π^-
400 π^+
44 K^+
13 K
800 g
1817 total at 10 MHz

High Multiplicities

Quite Messy Events… (cf. Alice)

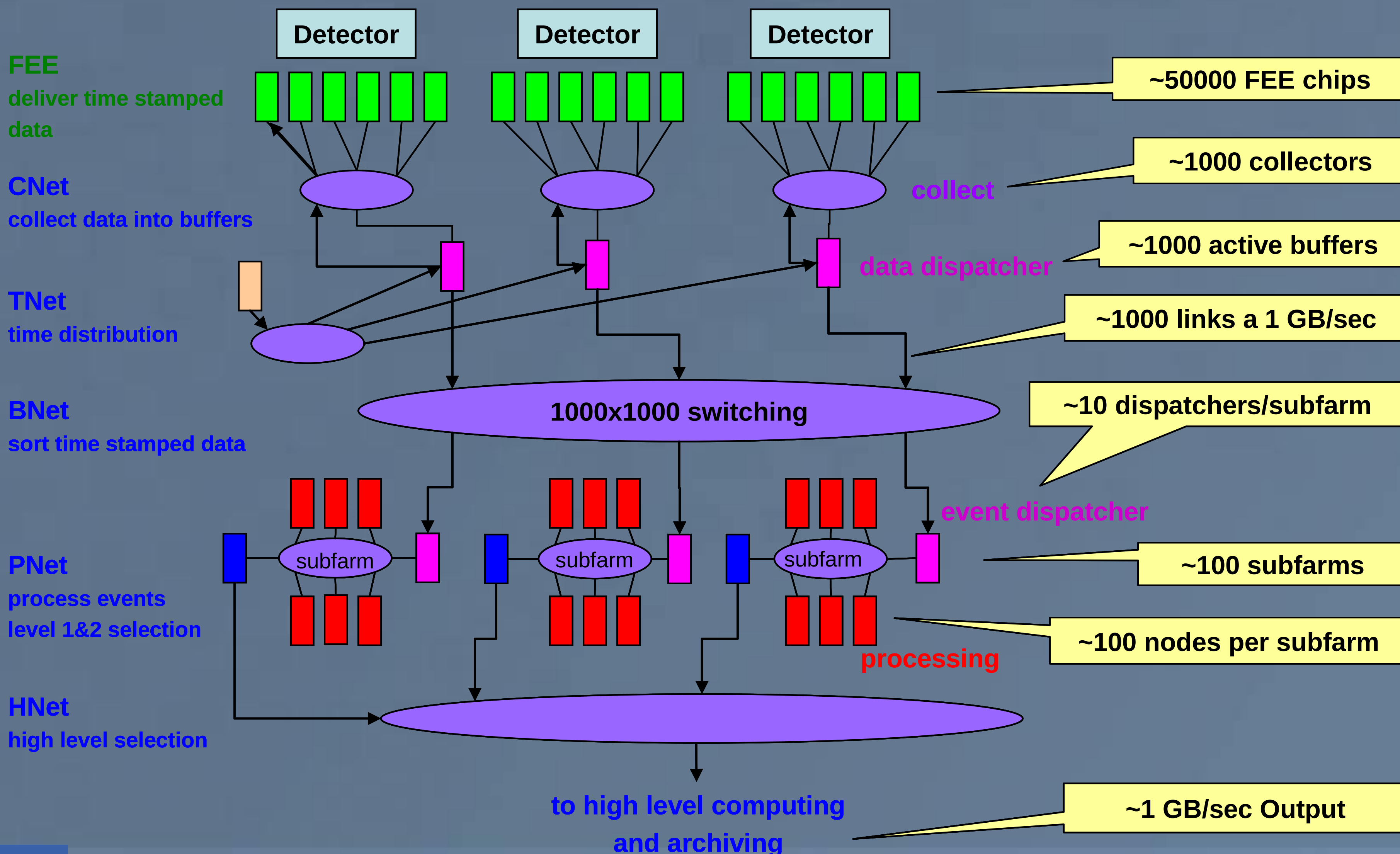


- Hardware triggering problematic
  - Complex Reconstruction
  - ‘Continuous’ beam
- Trigger-Free Readout
  - ‘Continuous’ beam
  - Self-Triggered channels with precise time-stamps
- Correlation and association later in CPU farm

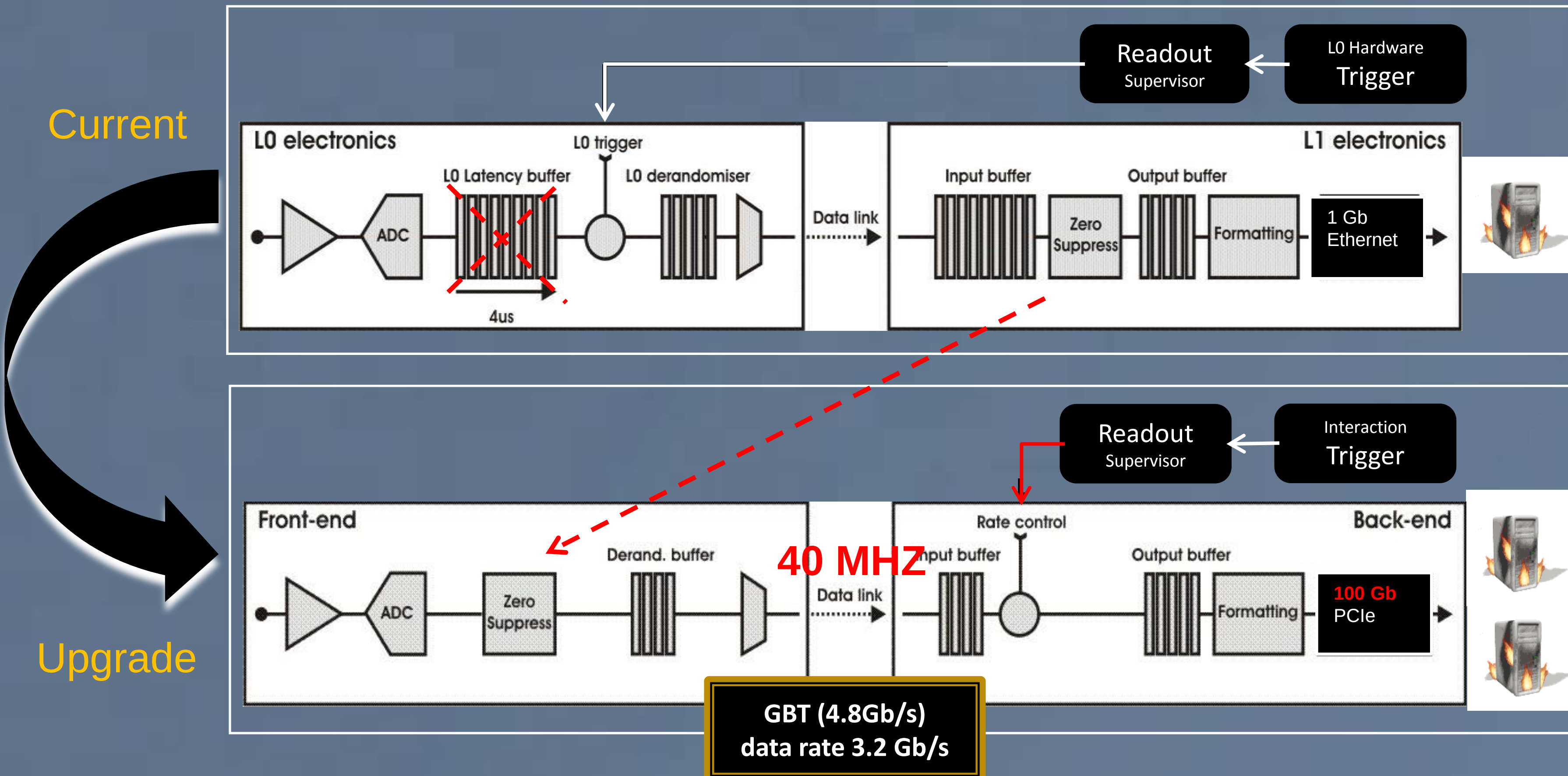
CBM Characteristics/Challenges

- Very **low-jitter (10 ps)** timing distribution network
- Data collection network to link detector elements with front-end electronics (**link speed O(GB/s)**)
- **High-performance** (~**O(TB/s)**) event building switching network connecting O(1000) Data Collectors to O(1000) Filter Nodes

CBM DAQ Architecture



LHCb DAQ Architecture for 2019



- All data will be readout @ collision rate 40 MHz by all frontend electronics (FEE) → ***a trigger-free read-out***!

- Zero-suppression will be done in FEEs to reduce the number of the GigaBit Transceiver (GBT) links

Expected Output BW: 40 Tbit/s

In conclusion…

What Do We Need to Read Out a Detector (successfully)?

- A selection mechanism ("trigger")
- Electronic readout of the sensors of the detectors ("front-end electronics")
- A system to keep all those things in sync ("clock")
- A system to collect the selected data ("DAQ")
- A Control System to configure, control and monitor the entire DAQ
- Time, money, students (lots of them, I mean YOU!)

Further Reading

- H. Spieler, “Semiconductor Detector Systems”, Oxford Univ. Press, 2005
- A. Sedra, K. Smith, “Microelectronic Circuits”, Oxford Univ. Press, 2009
- W. R. Leo, “Techniques for Nuclear and Particle Physics Experiments”, Springer, 1994
- O. Cobanoglu “Low-level front-end design”, this school
- Wikipedia!
- Conferences
  - IEEE Realtime
  - CHEP
  - TWEPP
  - IEEE NSS-MIC
- Journals
  - IEEE Transactions on Nuclear Science, in particular the proceedings of the IEEE Realtime conferences