Data Acquisition Systems

(and a little bit about trigger & electronics)

CERN Summerstudent Programme 2009

Niko Neufeld, CERN-PH

Introduction

- Data Acquisition is a specialized engineering discipline thriving mostly in the eco-system of large science experiments, particularly in HEP
- It consists mainly of electronics, computer science, networking and (we hope) a little bit of physics
- Some material and lots of inspiration for this lecture was taken from lectures by my predecessors
- Many thanks to S. Suman for his help with the drawings!

Outline

- Introduction
  - Data acquisition
  - The first data acquisition campaign
- A simple DAQ system
  - One sensor
  - More and more sensors
- Read-out with buses
  - Crates & Mechanics
  - The VME Bus
- A DAQ for a large experiment
  - Sizing it up
  - Trigger
  - Front-end Electronics
  - Readout with networks
    - Event building in switched networks
    - Problems in switched networks
- A lightning tour of ALICE, ATLAS, CMS and LHCb DAQs

Disclaimer

- Trigger and DAQ are two vast subjects covering a lot of physics and electronics
- Based entirely on personal bias I have selected a few topics
- While most of it will be only an overview at a few places we will go into some technical detail
- Some things will be only touched upon or left out altogether – information on those you will find in the references at the end
  - Electronics (lectures by Ph. Farthouat)
  - High Level Trigger (lectures by G. Dissertori)
  - DAQ of experiments outside HEP/LHC
  - Management of large networks and farms
  - High-speed mass storage
  - Experiment Control (= Run Control + Detector Control / DCS)

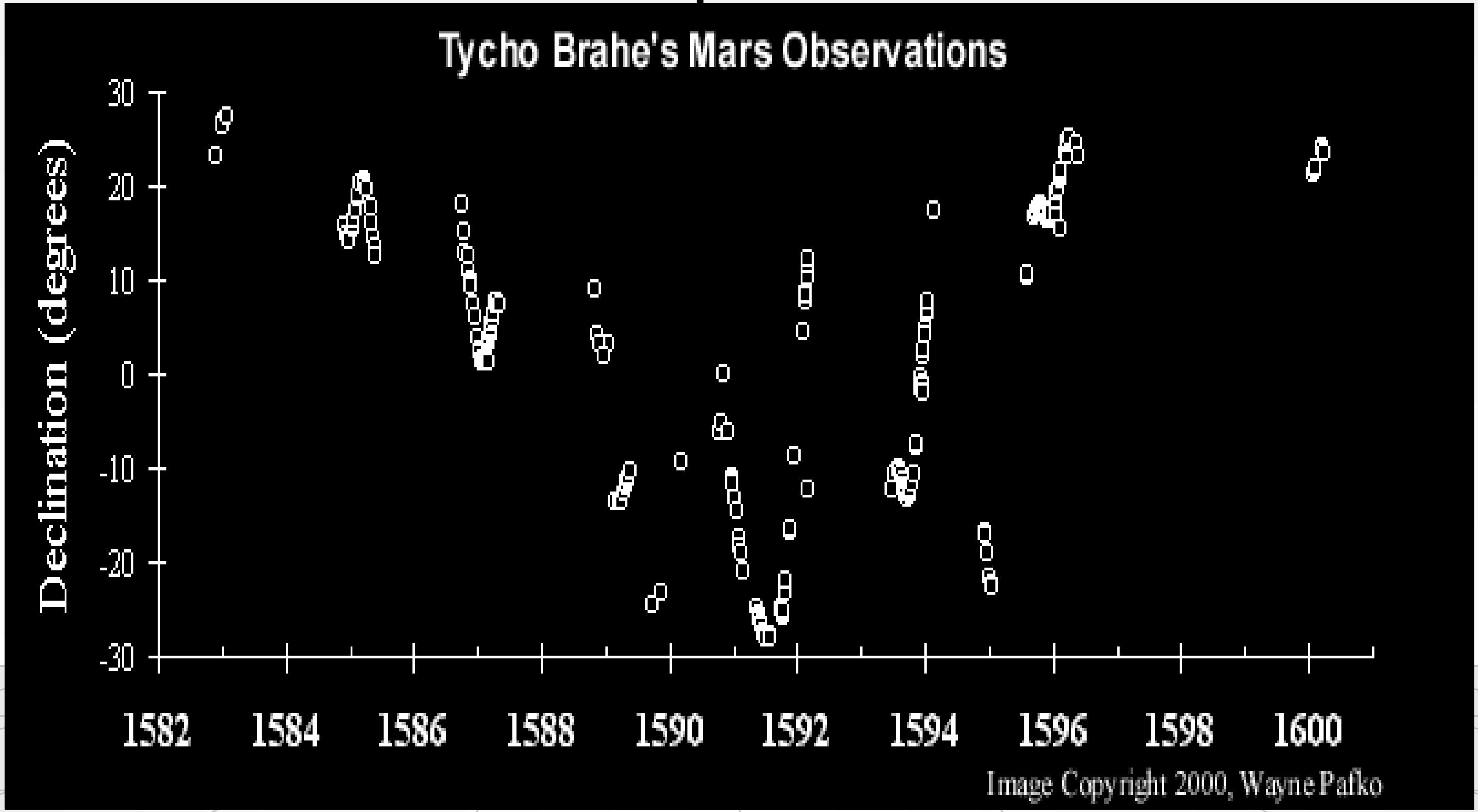
Tycho Brahe and the Orbit of Mars

I've studied all available charts of the planets and stars and none of them match the others. There are just as many measurements and methods as there are astronomers and all of them disagree. What's needed is a long term project with the aim of mapping the heavens conducted from a single location over a period of several years.

Tycho Brahe, 1563 (age 17).

- First measurement campaign
- Systematic data acquisition
  - Controlled conditions (same time of the day and month)
  - Careful observation of boundary conditions (weather, light conditions etc…) - important for data quality / systematic uncertainties

The First Systematic Data Acquisition



- Data acquired over 18 years, normally e every month
- Each measurement lasted at least 1 hr with the naked eye
- Red line (only in the animated version) shows comparison with modern theory

Tycho's DAQ in Today's Terminology

- Bandwith (bw) = Amount of data transferred / per unit of time
  - "Transferred" = written to his logbook
  - "unit of time" = duration of measurement
  - bw_{Tycho} = ~ 100 Bytes / h (compare with LHCb 40.000.000.000 Bytes / s)
- Trigger = in general something which tells you when is the "right" moment to take your data
  - In Tycho's case the position of the sun, respectively the moon was the trigger
  - the trigger rate ~ 3.85 x 10^{-6} Hz (compare with LHCb 1.0 x 10^{6} Hz)

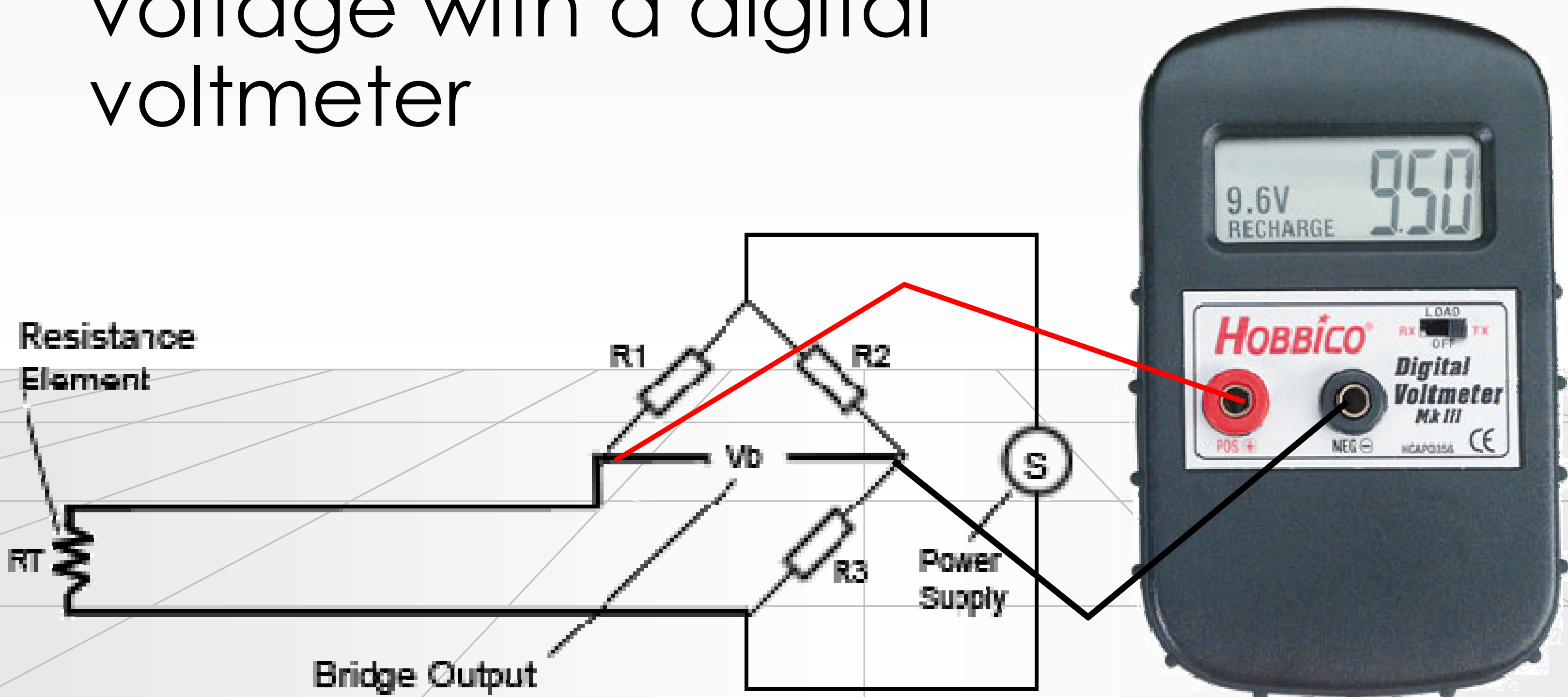
Some More Thoughts on Tycho

- Tycho did not do the correct analysis of the Mars data, this was done by Johannes Kepler (1571-1630), eventually paving the way for Newton's laws
- Morale: the size & speed of a DAQ system are not correlated with the importance of the discovery!

A Very Simple Data Acquisition System

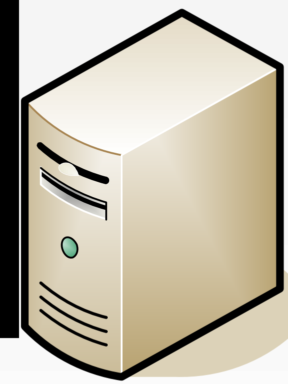
Measuring Temperature

- Suppose you are given a Pt100 thermo-resistor
- We read the temperature as a voltage with a digital voltmeter



Reading Out Automatically

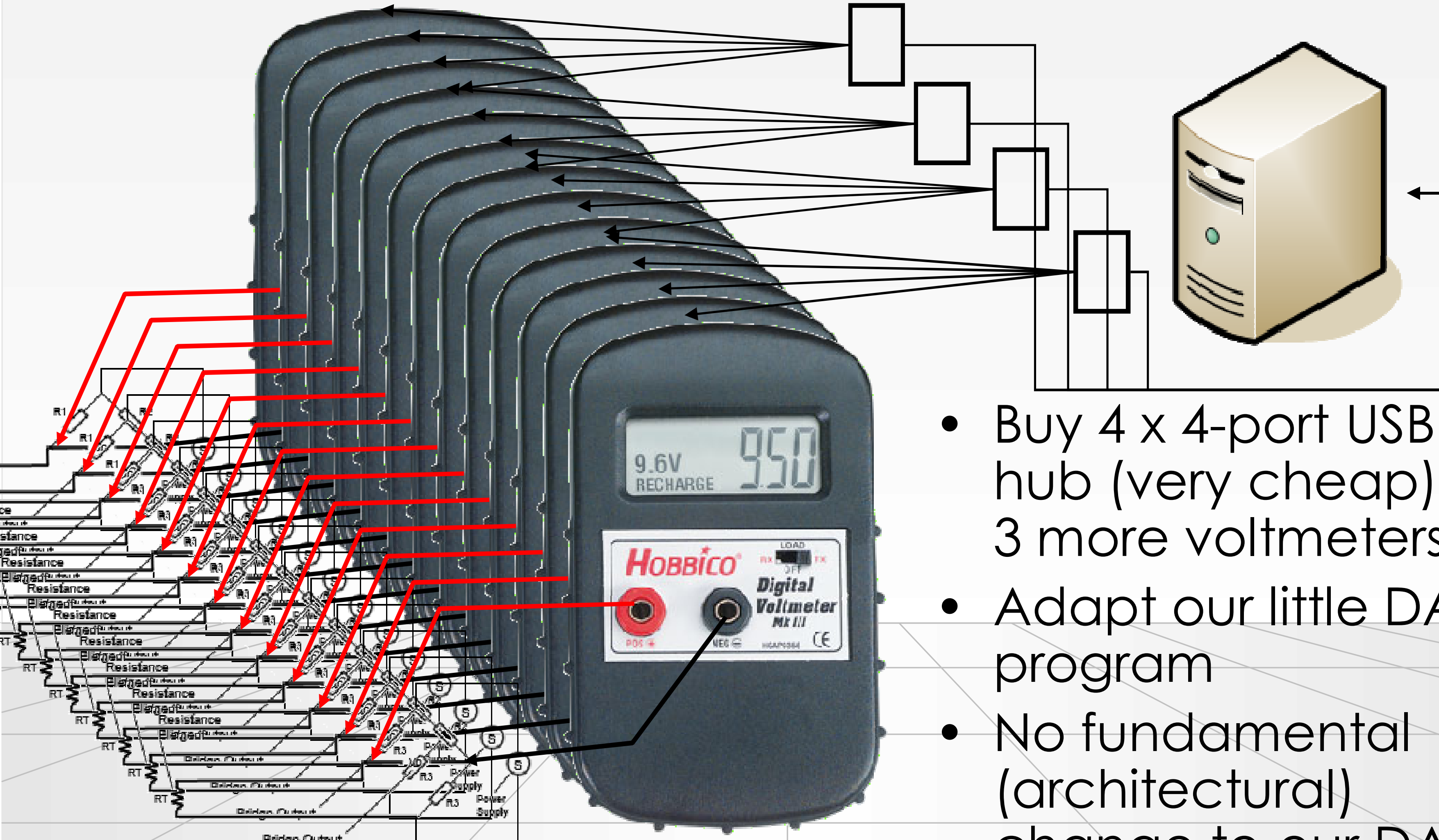
```
#include <libusb.h>
struct usb_bus *bus;
struct usb_device *dev;
usb_dev_handle *vmh = 0;
usb_find_busses(); usb_find_devices();
for (bus = usb_busses; bus; bus = bus->next)
          for (dev = bus->devices; dev; dev = dev->next)
                    if (dev->descriptor.idVendor == HOBBICO) vmh = usb_open(dev);
usb_bulk_read(vmh ,3,&u,sizeof(float),500);
```



Note how small the sensor has become.
In DAQ we normally need not worry about the details of the things we readout

USB/RS232

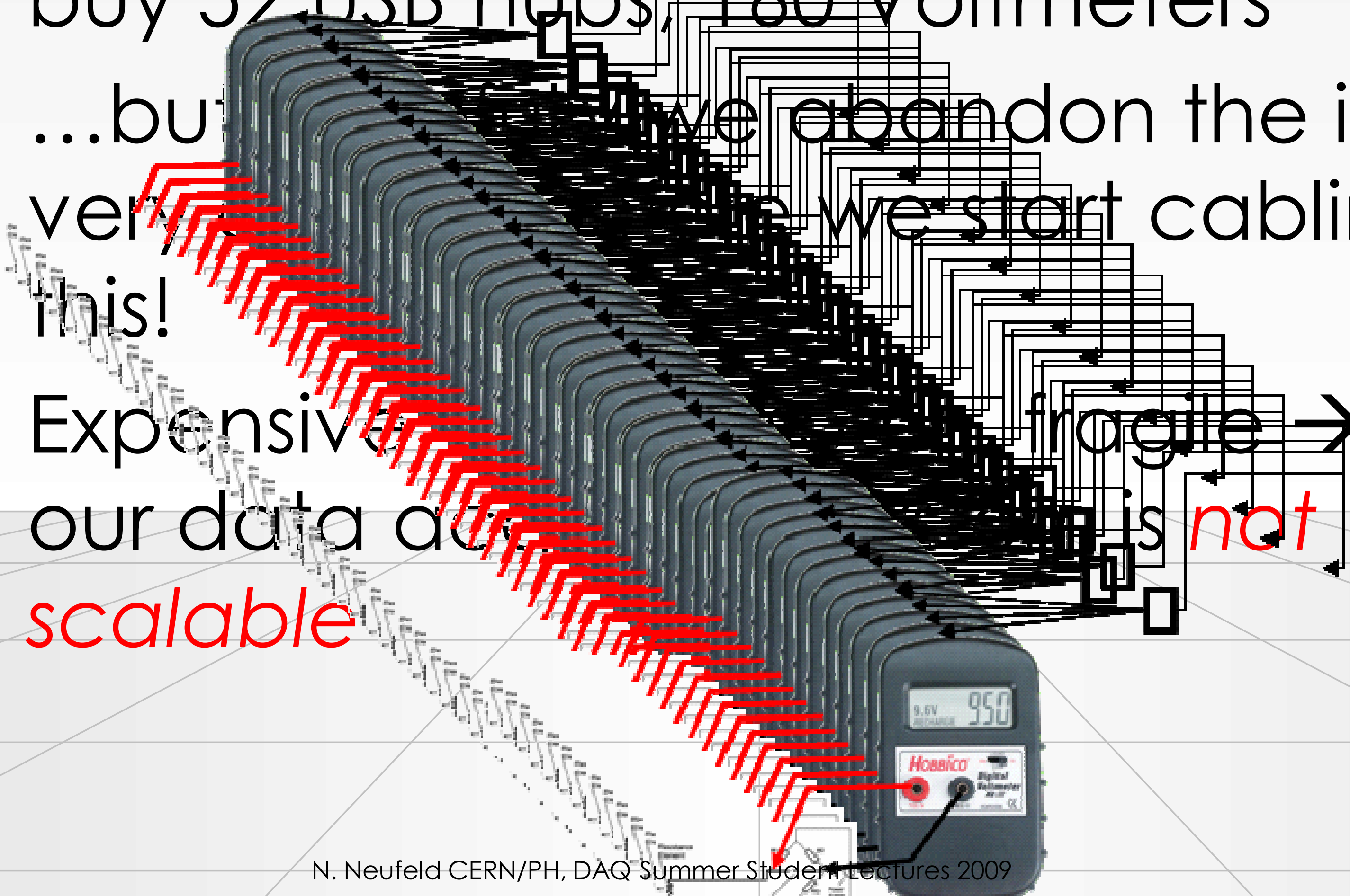
Read-out 16 Sensors



- Buy 4 x 4-port USB hub (very cheap) (+ 3 more voltmeters)
- Adapt our little DAQ program
- No fundamental (architectural) change to our DAQ

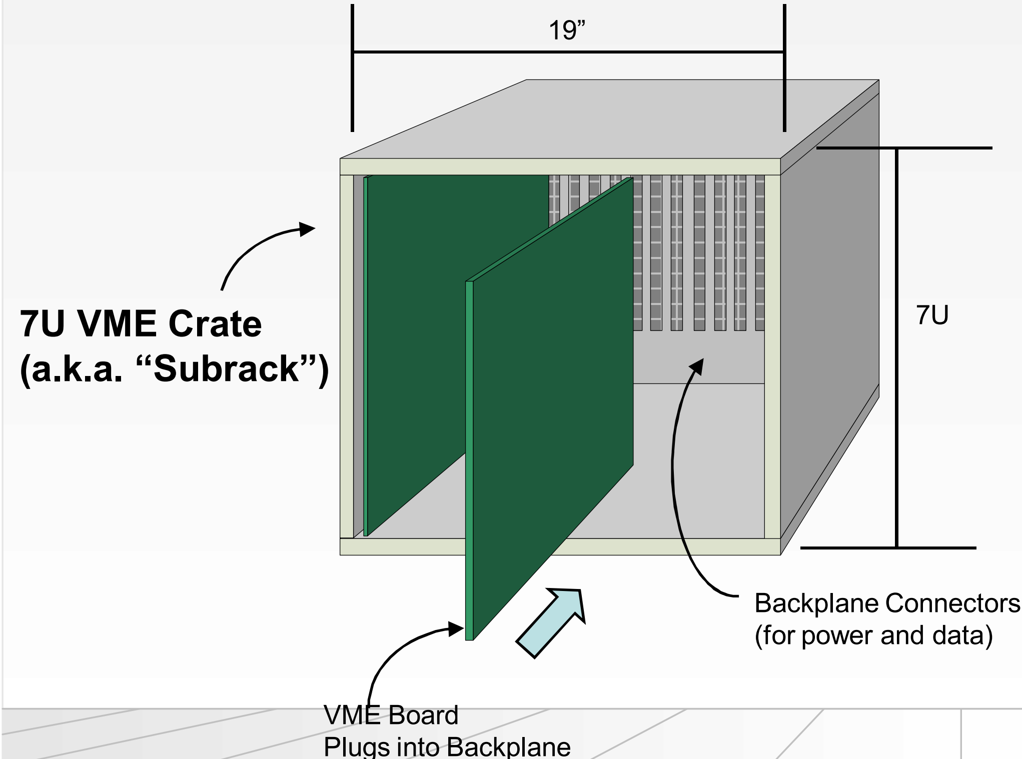
Read-out 160 Sensors

- For a moment we (might) consider to buy 52 USB hubs, 160 Voltmeters
- …but … we abandon the idea very … we start cabling this!
- Expensive … fragile → our data … is *not scalable*



Read-out with Buses

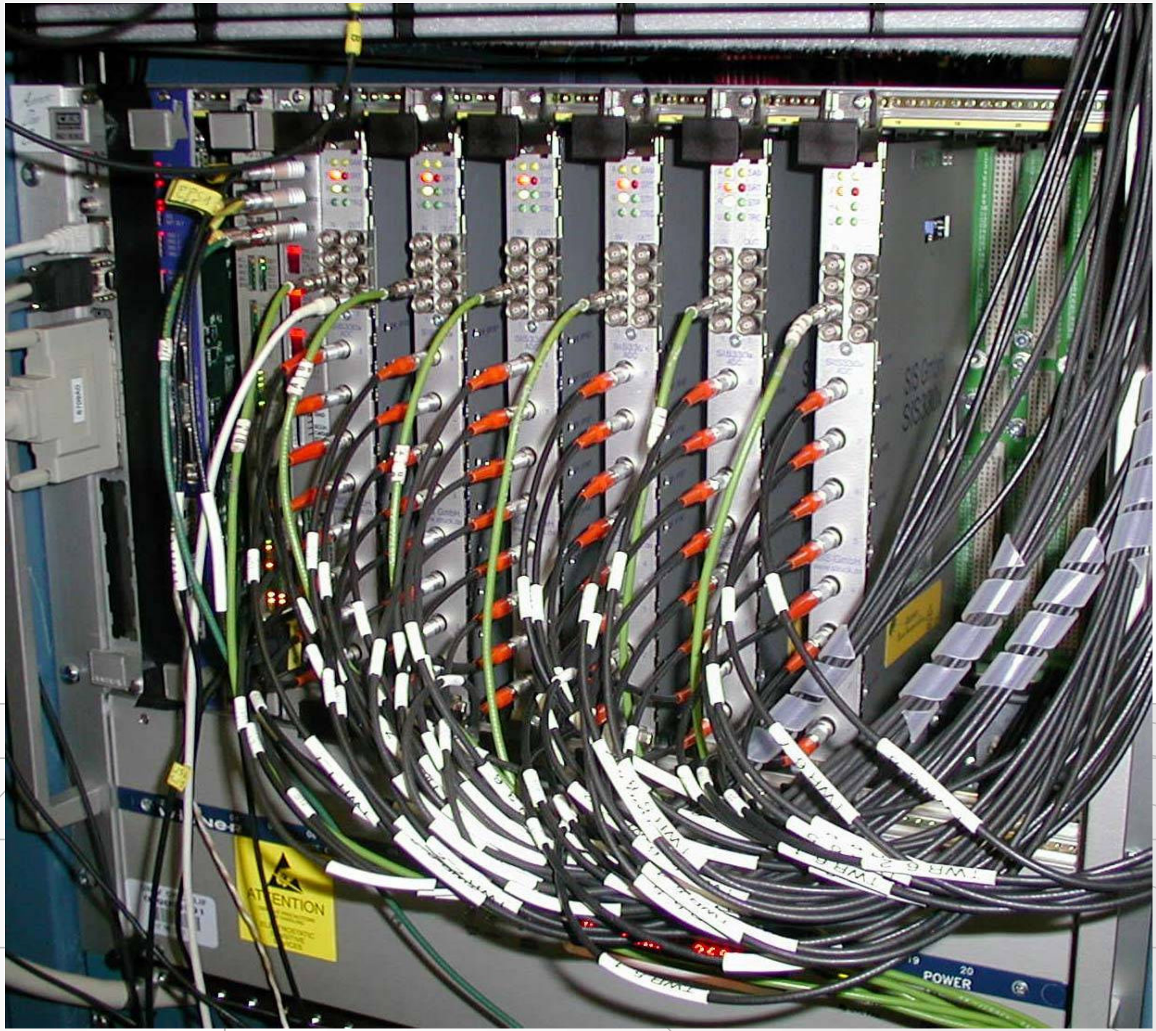
A Better DAQ for Many (temperature) Sensors



- Buy or build a compact multi-port volt-meter module, e.g. 16 inputs
- Put many of these multi-port modules together in a common chassis or crate
- The modules need
  - Mechanical support
  - Power
  - A standardized way to access their data (our measurement values)
- All this is provided by standards for (readout) electronics such as VME (IEEE 1014)

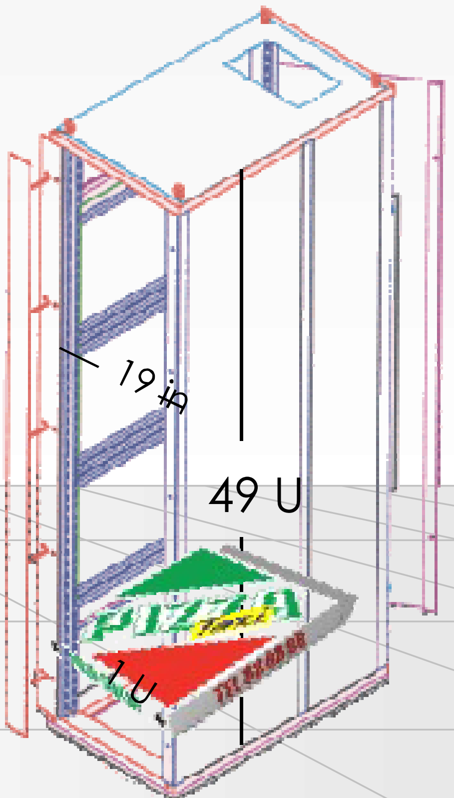
DAQ for 160 Sensors Using VME

- Readout boards in a *VME-crate*
  - mechanical standard for
  - electrical standard for power on the backplane
  - signal and protocol standard for communication on a *bus*



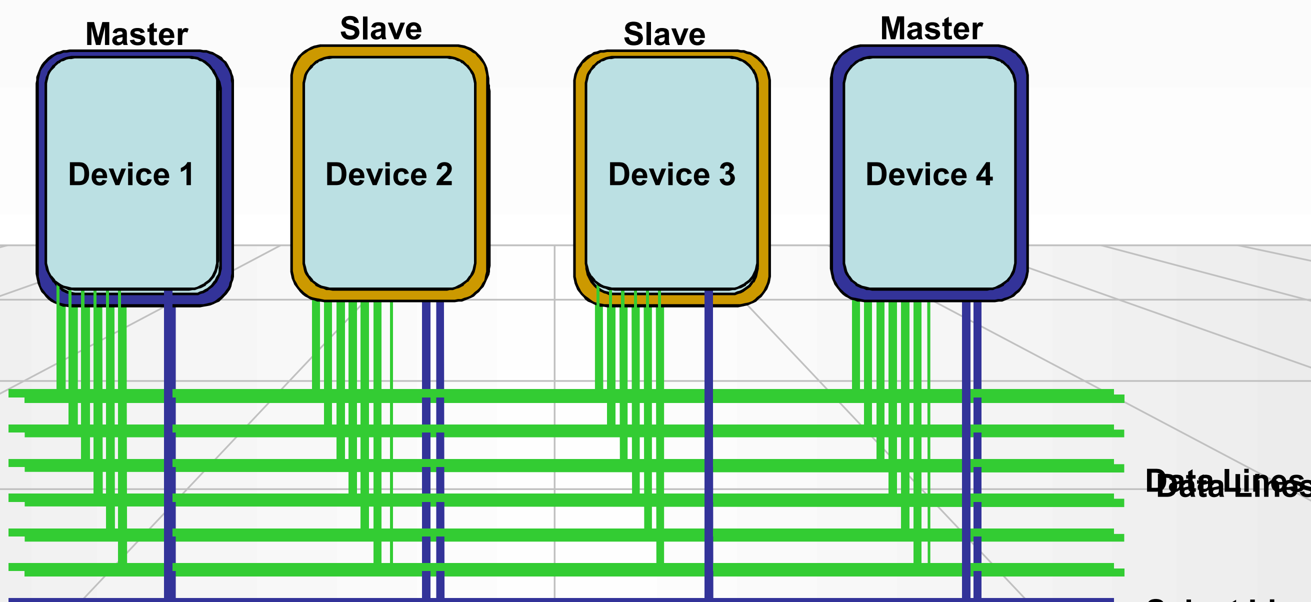
A Word on Mechanics and Pizzas

- The width and height of racks and crates are measured in US units: inches (in, '') and U
  - 1 in = 25.4 mm
  - 1 U = 1.75 in = 44.45 mm
- The width of a ''standard'' rack is 19 in.
- The height of a crate (also sub-rack) is measured in Us
- Rack-mountable things, in particular computers, which are 1 U high are often called *pizza-boxes*
- At least in Europe, the depth is measured in mm
- Gory details can be found in IEEE 1101.x (VME mechanics standard)



Communication in a Crate: Buses

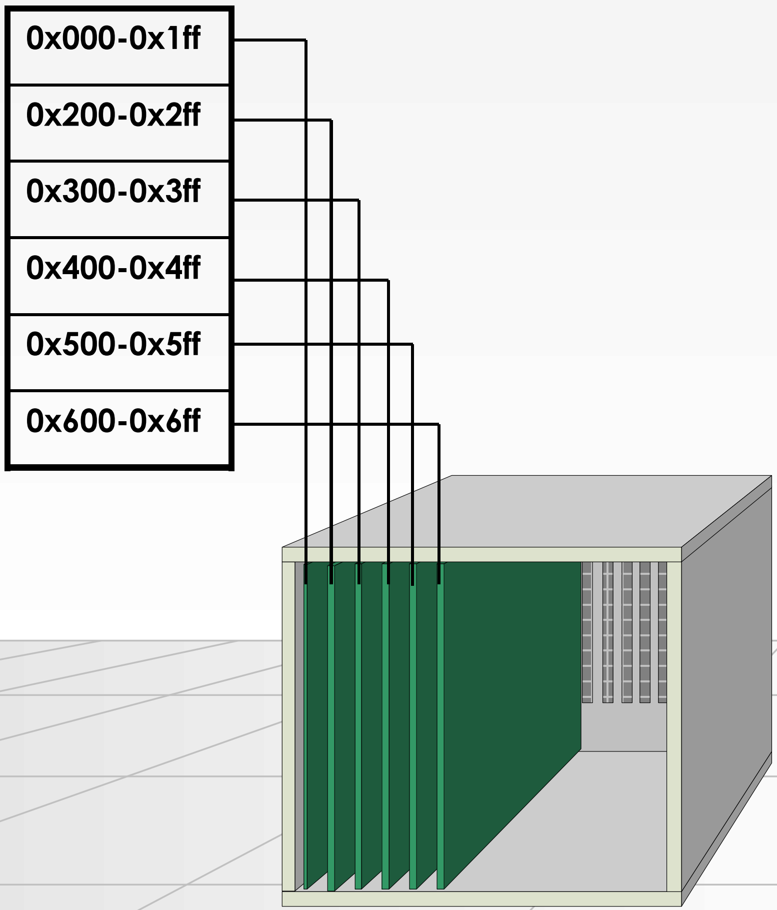
- A bus connects two or more devices and allows the to communicate
- The bus is shared between all devices on the bus → arbitration is required
- Devices can be masters or slaves (some can be both)
- Devices can be uniquely identified ("addressed") on the bus



Buses

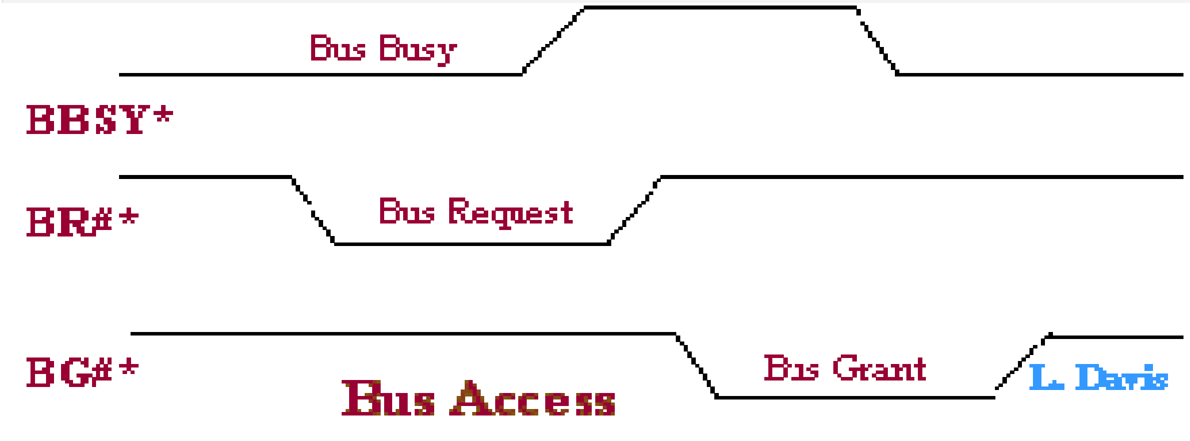
- Famous examples: PCI, USB, VME, SCSI
  - older standards: CAMAC, ISA
  - upcoming: ATCA
  - many more: FireWire, I2C, Profibus, etc…
- Buses can be
  - local: PCI
  - external peripherals: USB
  - in crates: VME, compactPCI, ATCA
  - long distance: CAN, Profibus

The VME Bus



- In a VME crate we can find three main types of modules
  - The controller which monitors and arbitrates the bus
  - Masters read data from and write data to slaves
  - Slaves send data to and receive data from masters
- Addressing of modules
  - In VME each module occupies a part of a (flat) range of addresses (24 bit to 32 bit)
  - Address range of modules is hardwired (conflicts!)

VME protocol 1) Arbitration

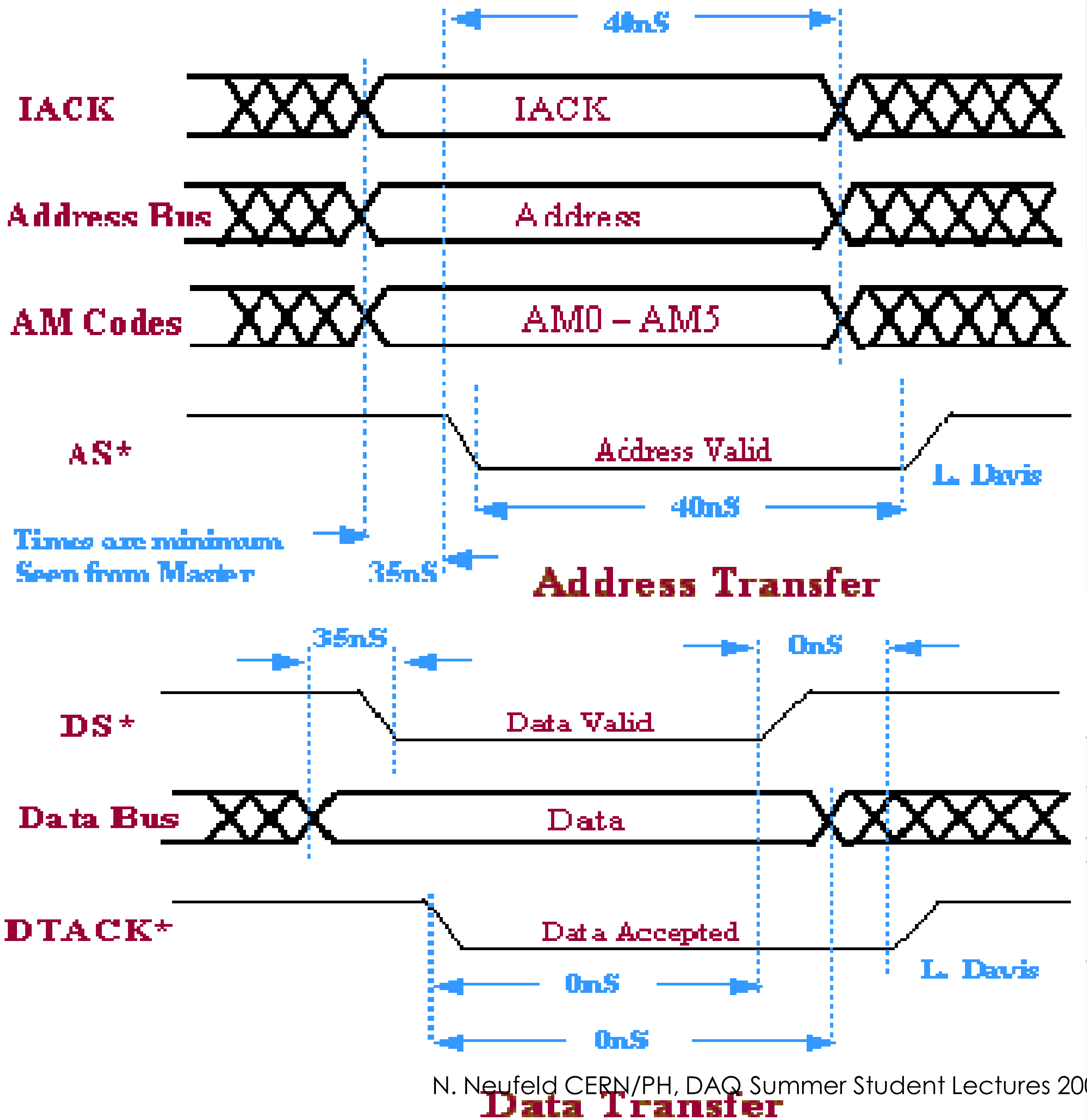


- Arbitration: Master asserts*) BR#, Controller answers by asserting BG#
- If there are several masters requesting at the same time the one physically closest to the controller wins
- The winning master drives BBSY* high to indicate that the bus is now in use

Pictures from http://www.interfacebus.com

*) assert means driving the line to logical 0 (VME control lines are inverted or active-low)

VME protocol 2) Write transfer

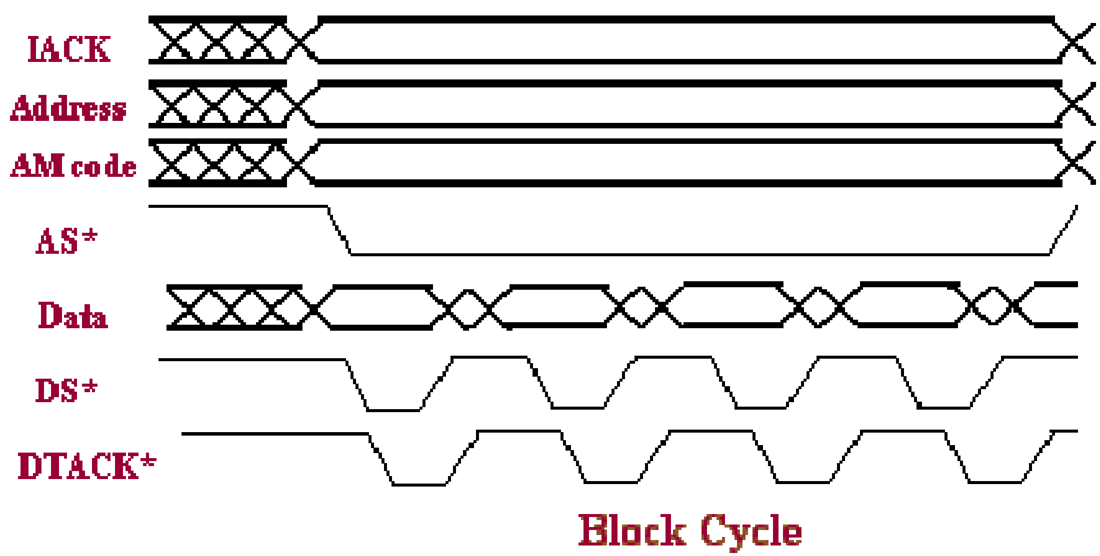


- The Master writes data and address to the data / respectively data bus
- It asserts DS* and AS* to signal that the data and address are valid
- The slave reads and acknowledges by asserting DTACK
- The master releases DS*, AS* and BSBSY*, the cycle is complete
- Note: there is no clock! The slave can respond whenever it wants. VME is an asynchronous bus

Speed Considerations

- Theoretically ~ 16 MB/s can be achieved
  - assuming the databus to be full 32-bit wide
  - the master never has to relinquish bus master ship
- Better performance by using block-transfers

VME protocol 3) Block transfer



- Block transfers are essential for Direct Memory Access (DMA)
- More performance can be gained by using the address bus also for data (VME64)

- After an address cycle several (up to 256) data cycles are performed
- The slave is supposed to increment the address counter
- The additional delays for asserting and acknowledging the address are removed
- Performance goes up to 40 MB/s
- In PCI this is referred to as "burst-transfer"

Advantages of buses

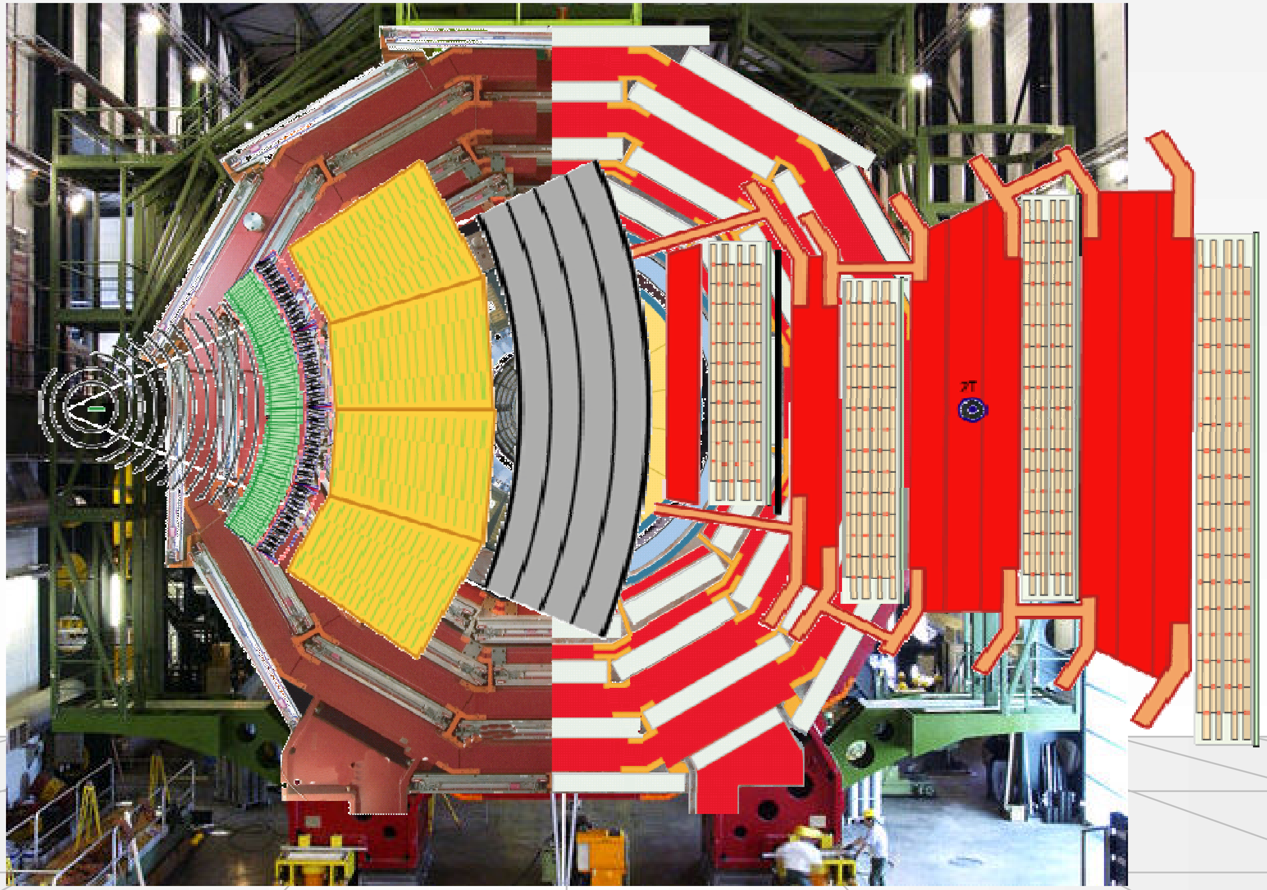
- Relatively simple to implement
  - Constant number of lines
  - Each device implements the same interface
- Easy to add new devices
  - topological information of the bus can be used for automagically choosing addresses for bus devices: this is what plug and play is all about.

Buses for DAQ at LHC?

- A bus is shared between all devices (each new active device slows everybody down)
  - Bus-width can only be increased up to a certain point (128 bit for PC-system bus)
  - Bus-frequency (number of elementary operations per second) can be increased, but decreases the physical bus-length
- Number of devices and physical bus-length is limited (scalability!)
  - For synchronous high-speed buses, physical length is correlated with the number of devices (e.g. PCI)
  - Typical buses have a lot of control, data and address lines (look at a SCSI or ATA cable)
- Buses are typically useful for systems < 1 GB/s

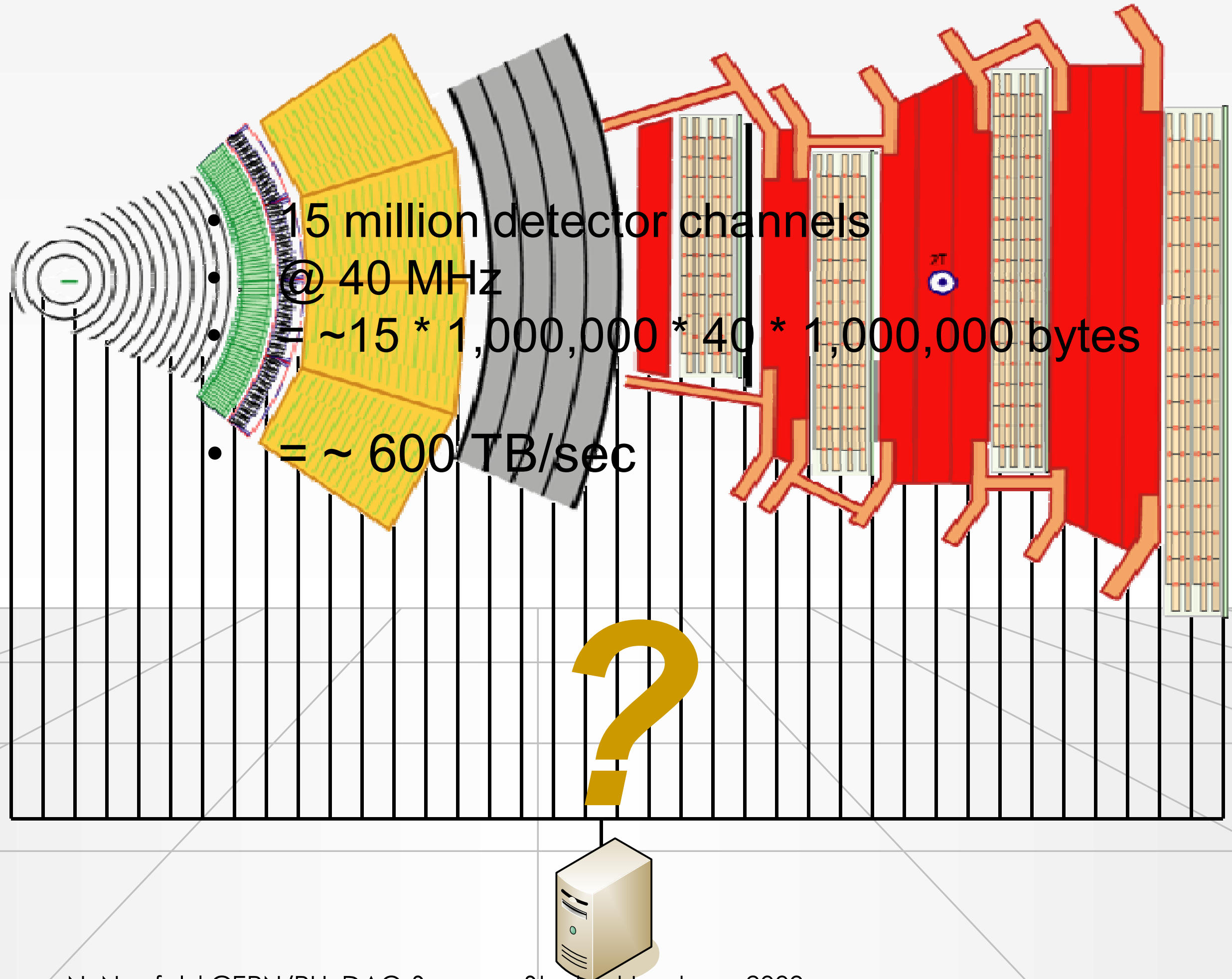
Data Acquisition for a Large Experiment

Moving on to Bigger Things…



The CMS Detector

Moving on to Bigger Things…



- 15 million detector channels
- @ 40 MHz
- = ~15 * 1,000,000 * 40 * 1,000,000 bytes
- = ~ 600 TB/sec

Designing a DAQ System for a Large HEP Experiment

- What defines "large"?
  - The number of channels: for LHC experiments $O(10^7)$ channels
    - a (digitized) channel can be between 1 and 14 bits
  - The rate: for LHC experiments everything happens at 40.08 MHz, the LHC bunch crossing frequency (This corresponds to 24.9500998 ns or 25 ns among friends)
- HEP experiments usually consist of many different sub-detectors: tracking, calorimetry, particle-ID, muon-detectors

First Questions

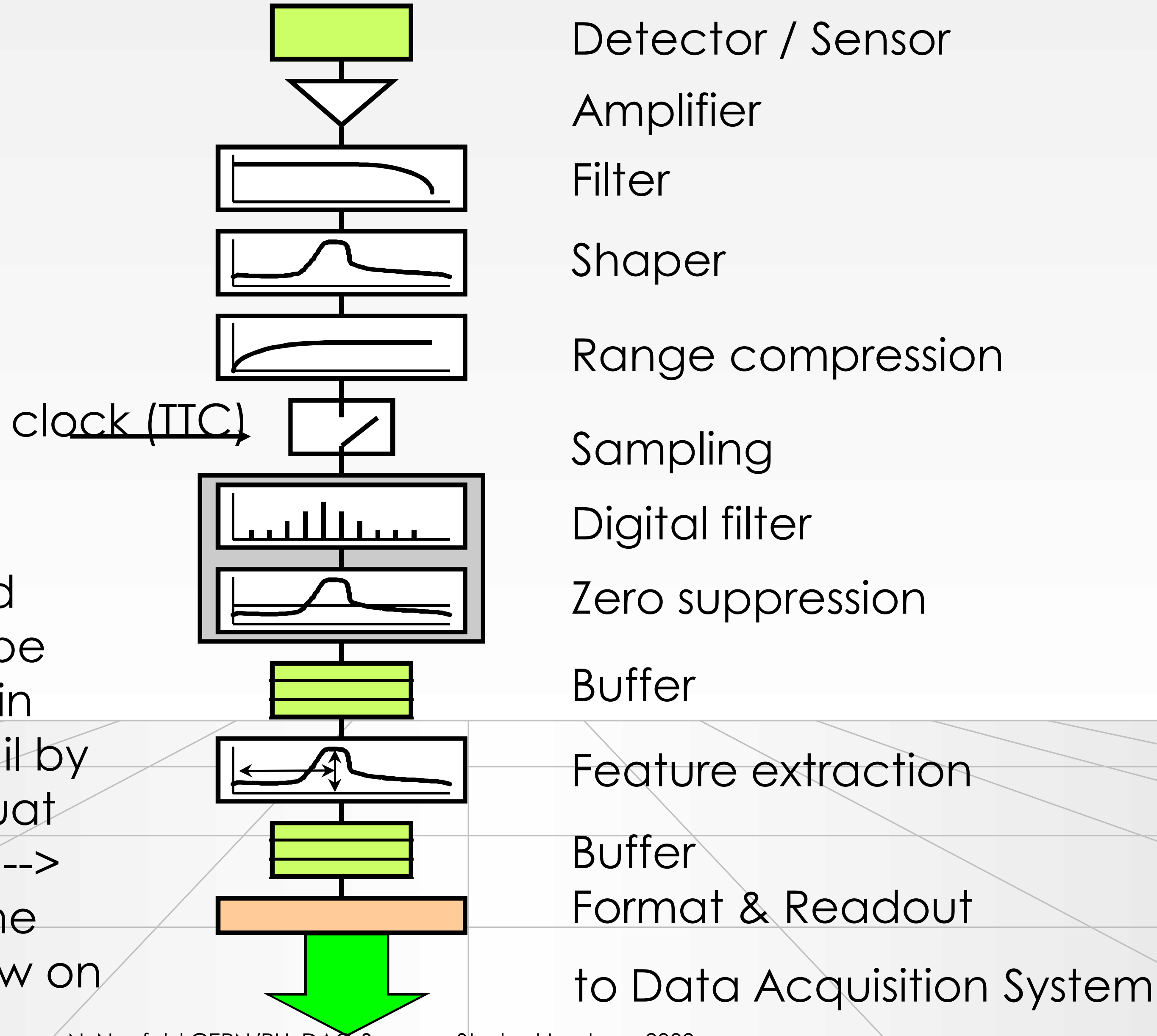
- Can we or do we want to save all the data?
- How do we select the data
- Is continuous read-out needed, i.e. an experiment in a collider? Or are there idle periods mixed with periods with many events – this is typically the case for fixed-target experiments
- How do we make sure that the values from the many different channels refer to the same original event (collision)

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

Frontend Electronics (FEE)

Bird's-Eye View on (front-end) Electronics

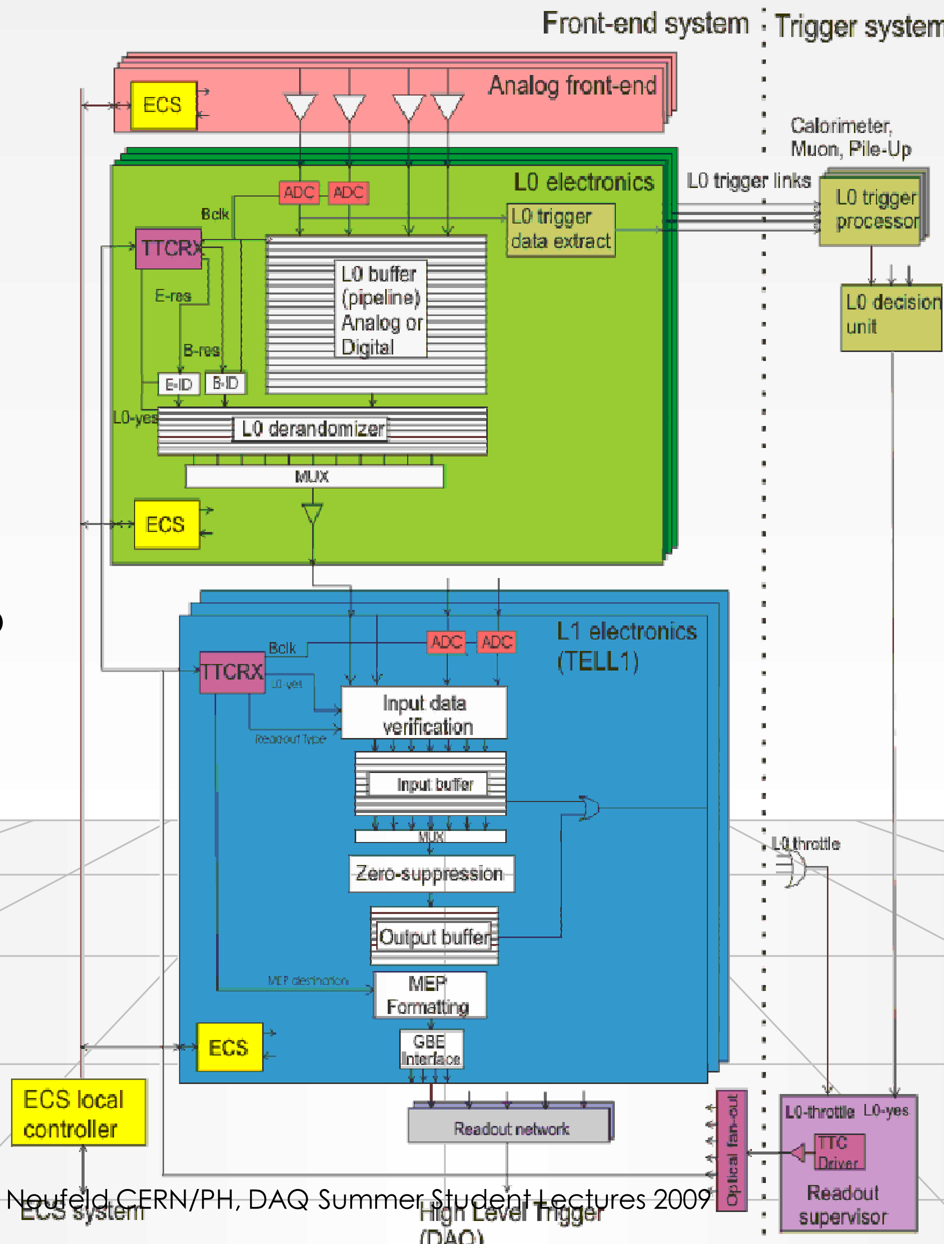


All this (and more) will be explained in great detail by Ph. Farthouat next week --> focus on the green arrow on beyond

FEE & DAQ by electronics engineers

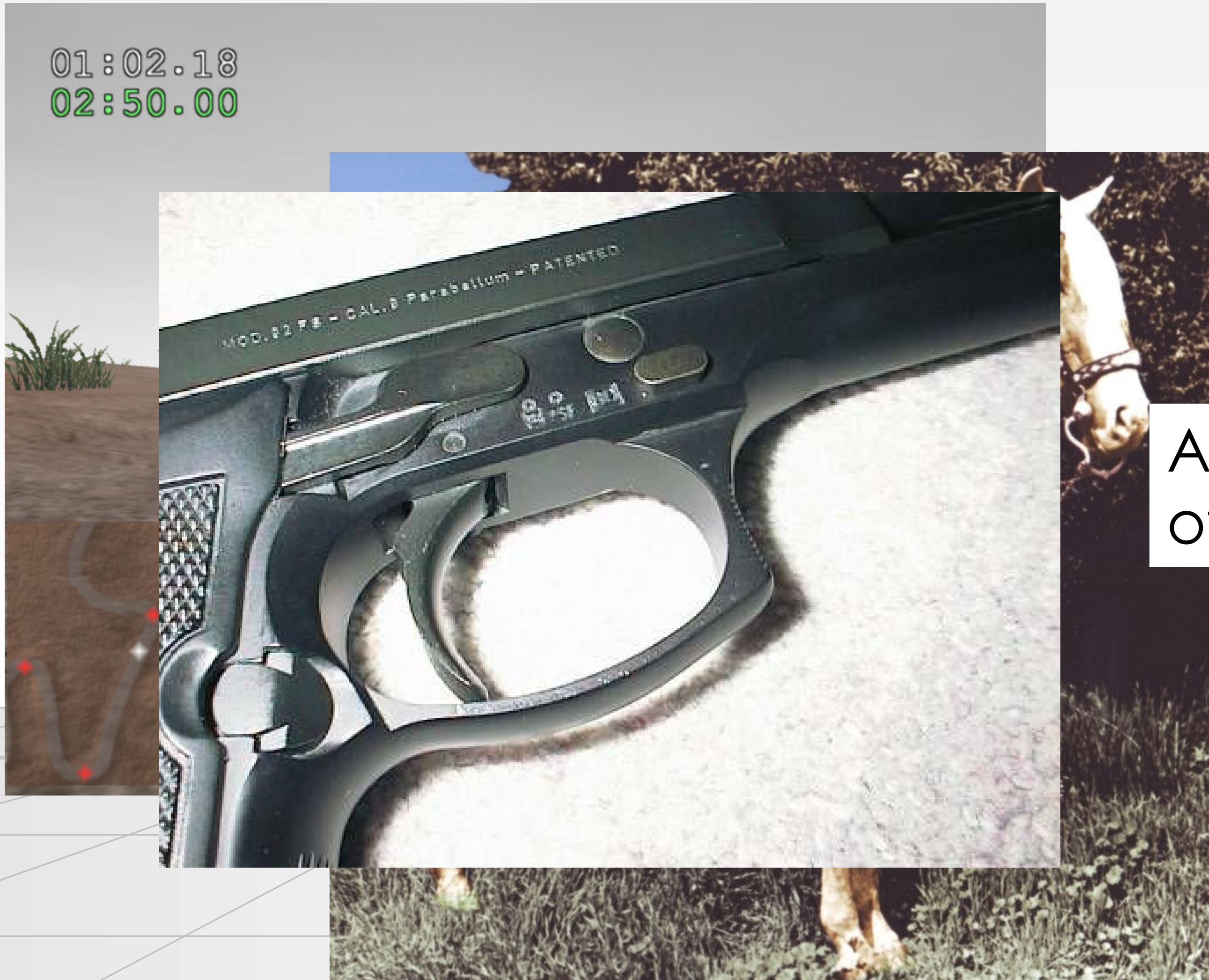
FEE = Front End Electronics

Example from LHCb



Trigger

What is a trigger?



n open-source
D rally game?

An important part
of a Beretta

The most famous
horse in
movie history?

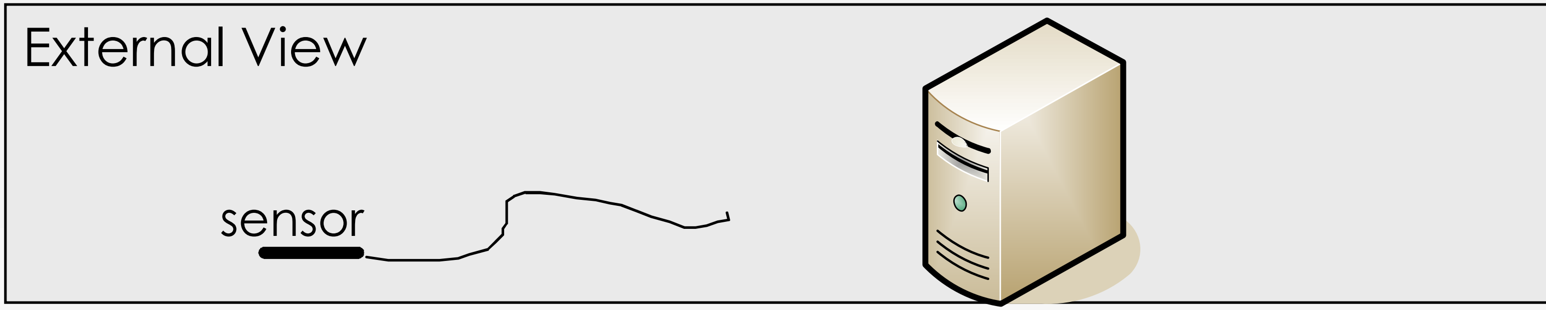
What is a trigger?

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.** “

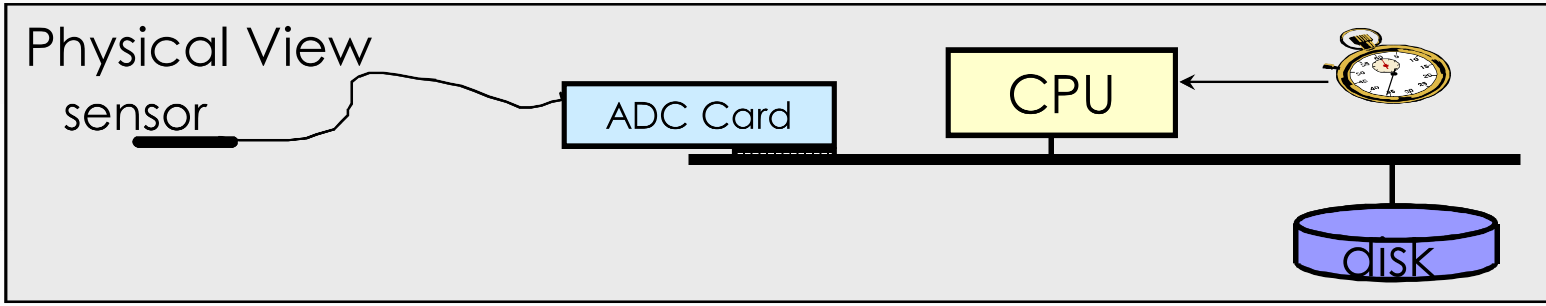
Trigger

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

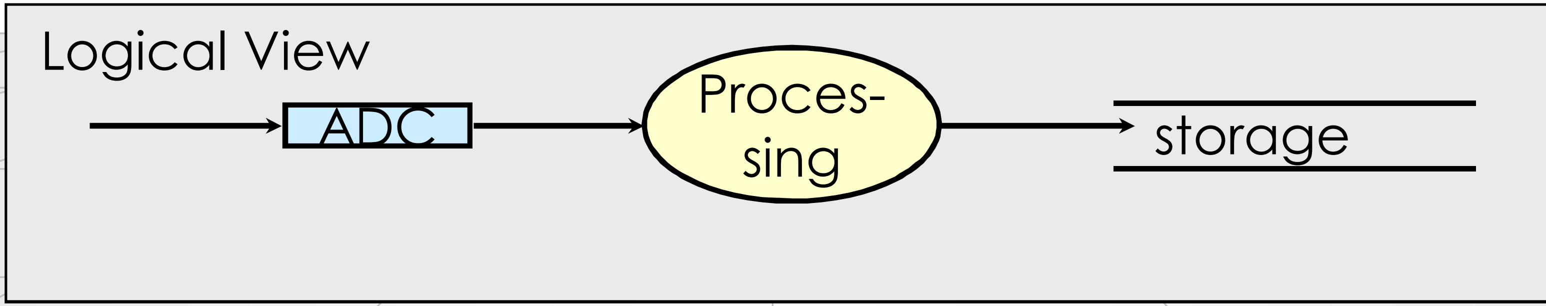
Trivial DAQ



External View
sensor

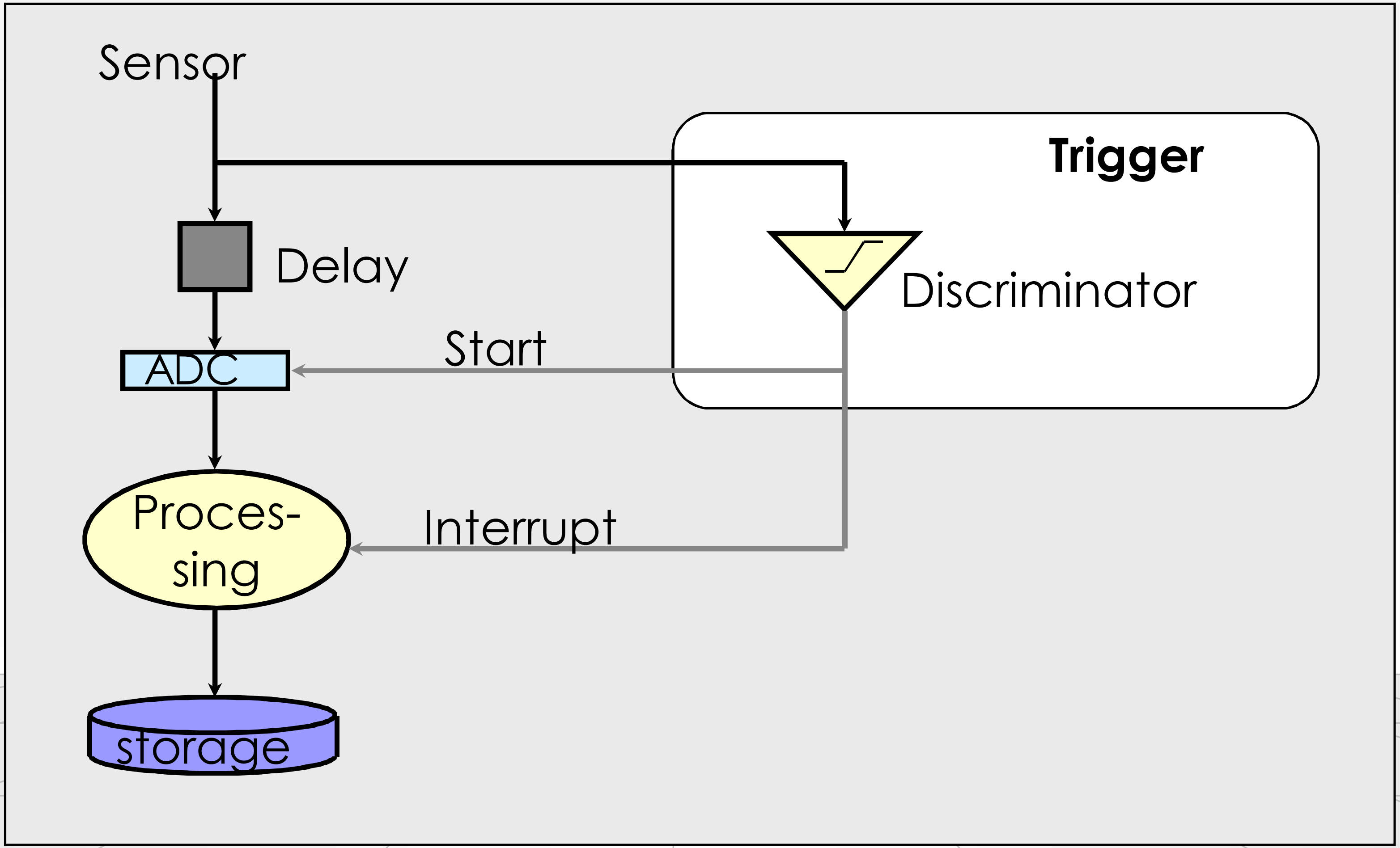


Physical View
sensor
ADC Card
CPU
disk



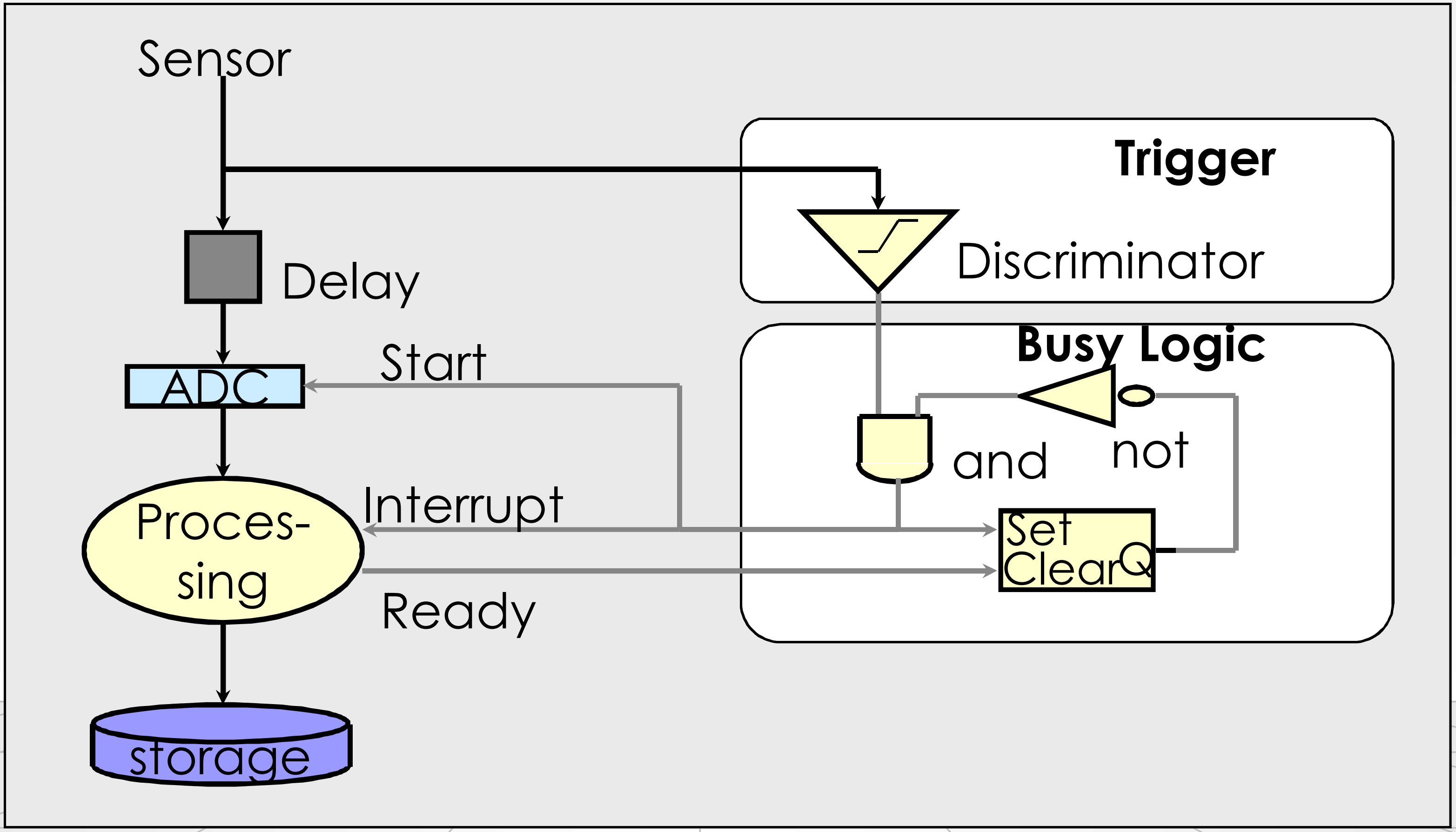
Logical View
ADC
Proces-
sing
storage

Trivial DAQ with a real trigger



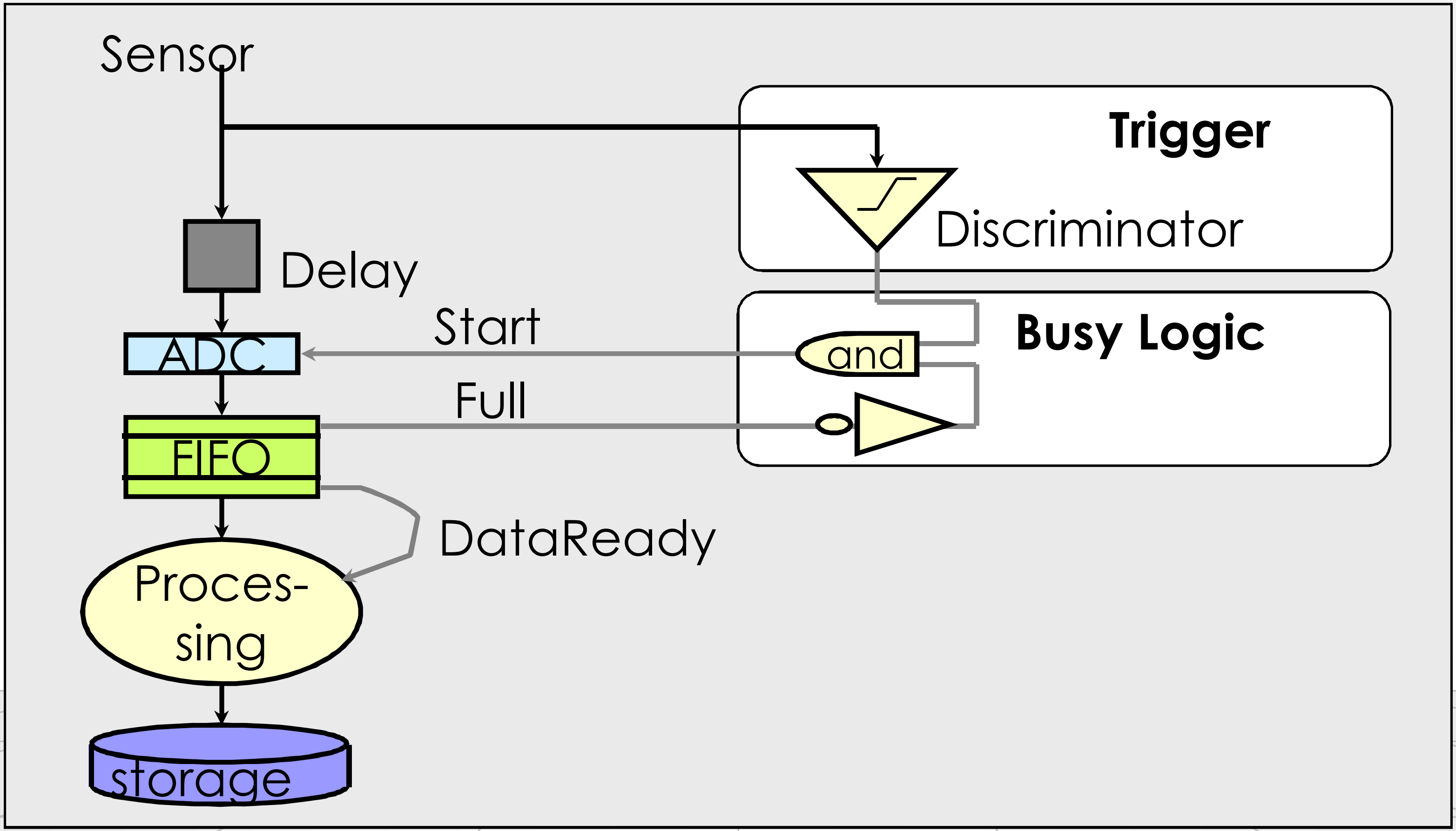
What if a trigger is produced when the *ADC* or *processing* is busy?

Trivial DAQ with a real trigger 2



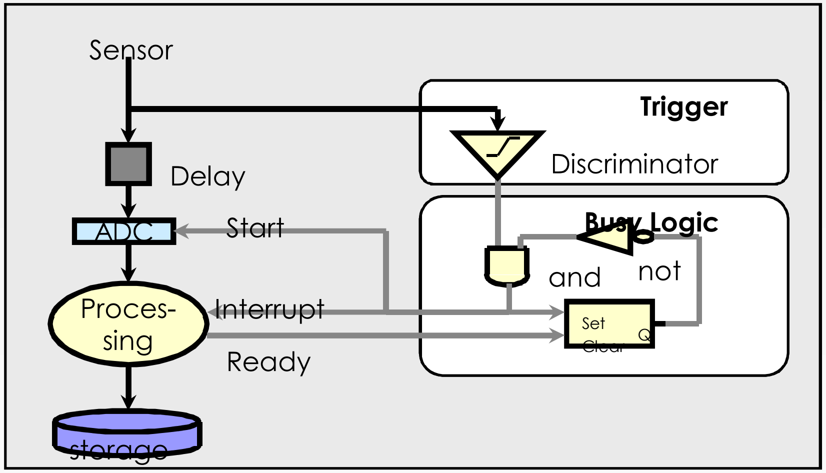
Deadtime (%) is the ratio between the time the DAQ is *busy* and the total time.

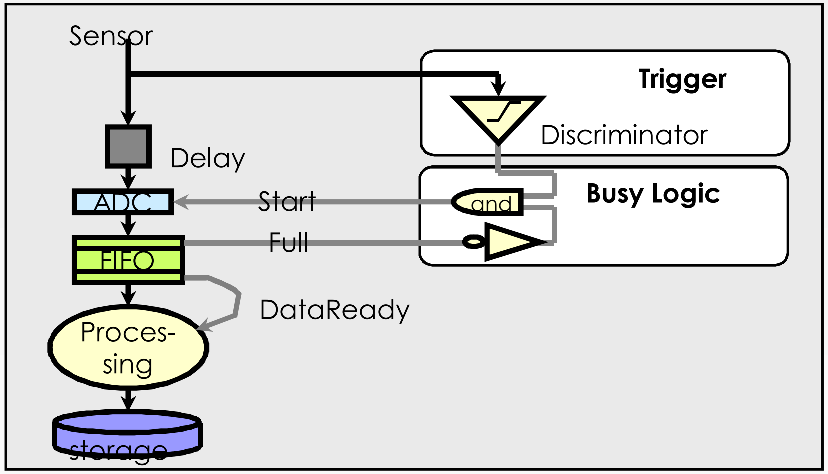
Trivial DAQ with a real trigger 3



Buffers are introduced to de-randomize data, to decouple the data production from the data consumption. **Better performance**.

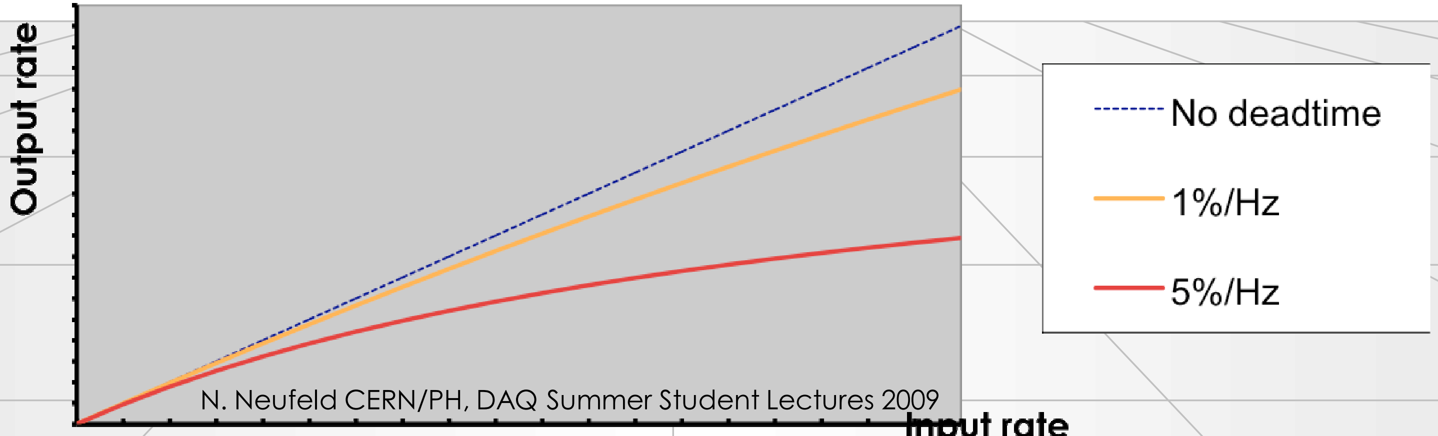
Effect of derandomizing





The system is *busy* during the ADC conversion time + processing time until the data is written to the storage

The system is *busy* during the ADC conversion time if the FIFO is not full (assuming the storage can always follow!)

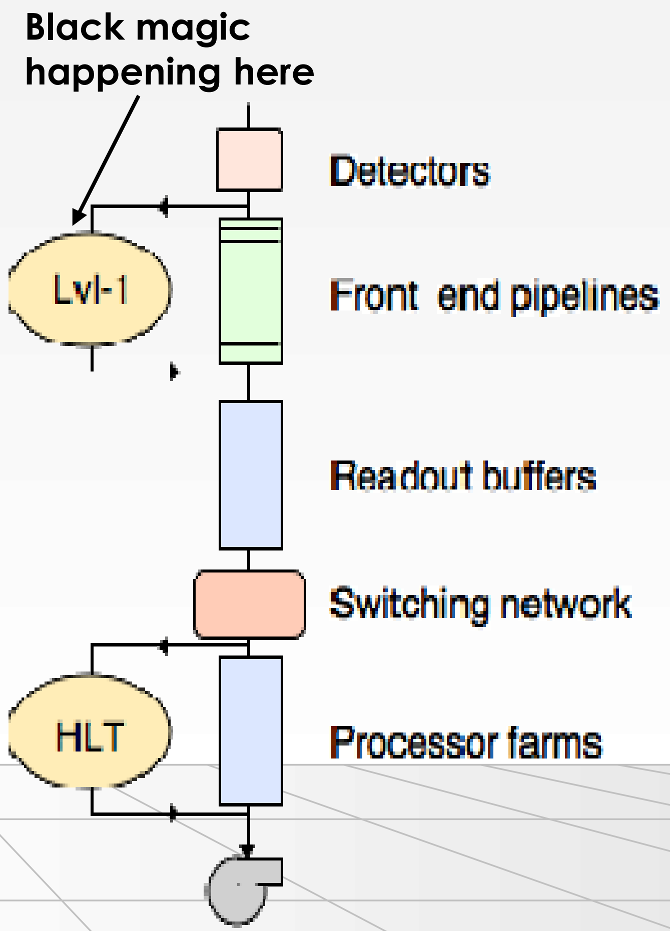


Choosing a trigger

- Keep it simple! (Remember Einstein: "As simple as possible, but not simpler")
- Even though "premature optimization is the root of all evil", think about efficiency (buffering)
- Try to have few adjustable parameters: scanning for a good working point will otherwise be a night-mare

Trigger for LHC

- No (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

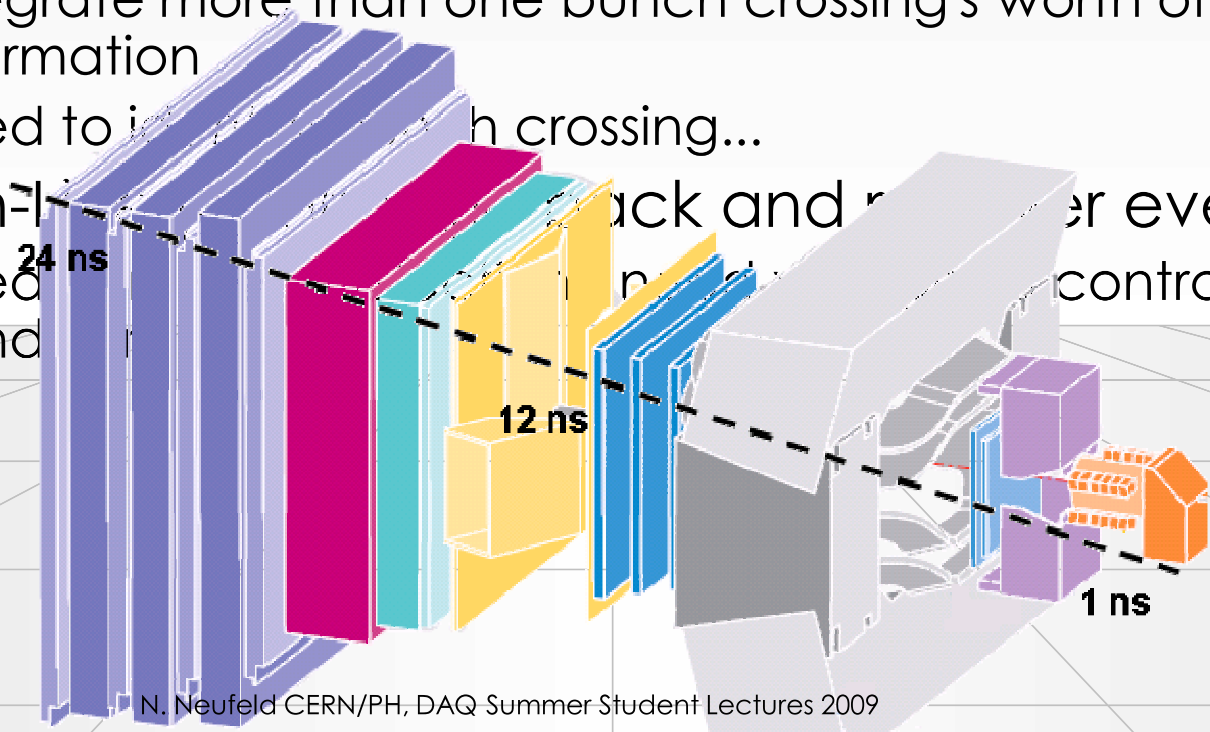


Inside the Box: How does a Level-1 trigger work?

- Millions of channels →: try to work as much as possible with "local" information
  - Keeps number of interconnections low
- Must be fast: look for "simple" signatures
  - Keep the good ones, kill the bad ones
  - Robust, can be implemented in hardware (fast)
- Design principle:
  - fast: to keep buffer sizes under control
  - every 25 nanoseconds (ns) a new event: have to decide within a few microseconds (μs): trigger-latency

Challenges for the L1 at LHC

- N (channels) ~ O(10^7); ≈20 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 i… h crossing...
- It's On-l… ack and … er events)
  - need … control over all cond…

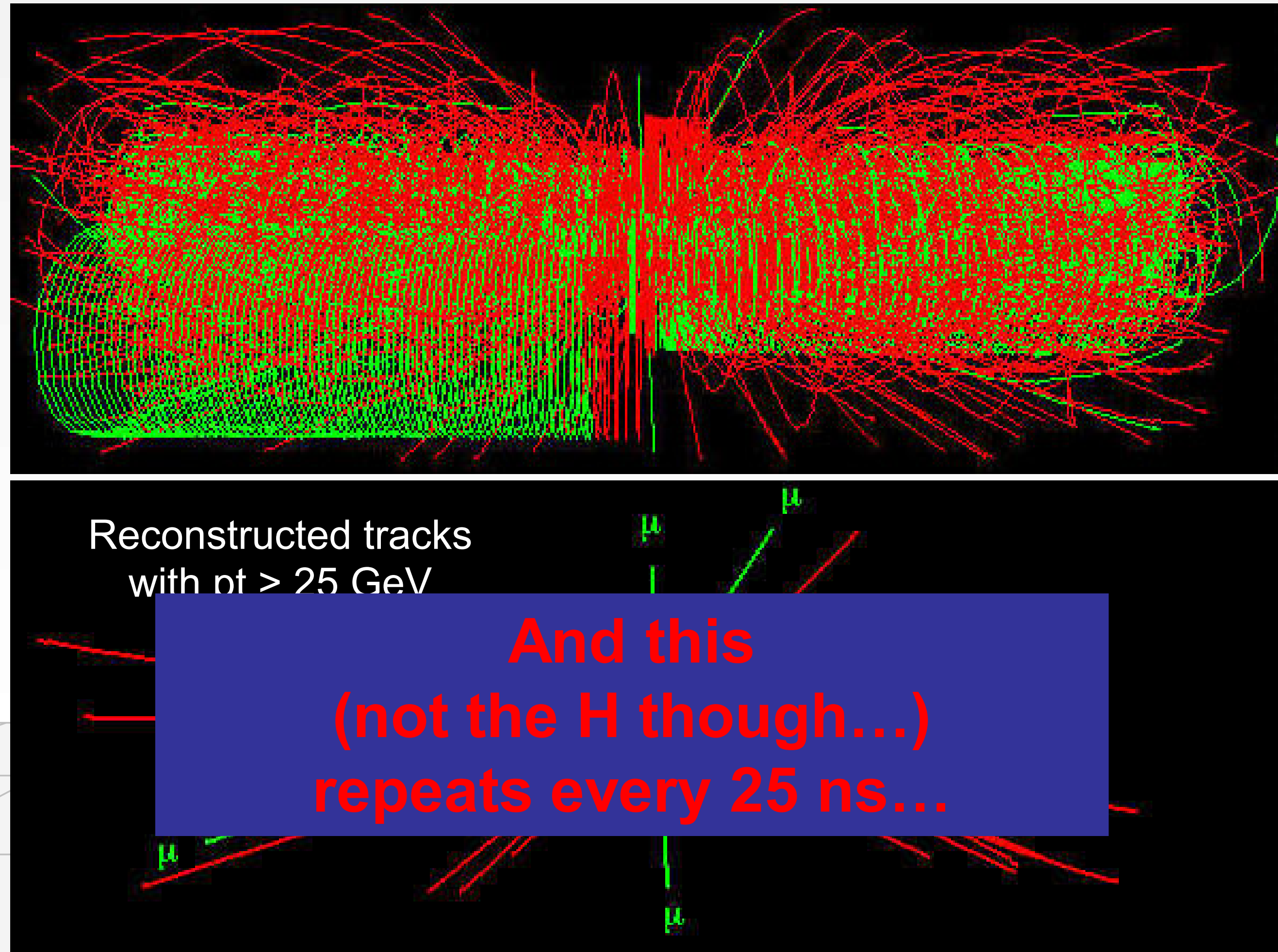


Know Your Enemy: pp Collisions at 14 TeV at 10^{34} $cm^{-2}s^{-1}$

- $\sigma(pp)$ = 70 mb --> >7 x 10^8 /s (!)
- In ATLAS and CMS* 20 min bias events will overlap
- $H \rightarrow ZZ$

$Z \rightarrow \mu\mu$

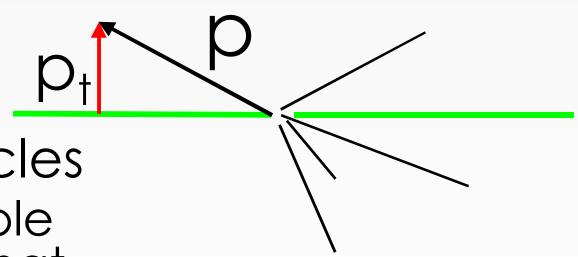
$H \rightarrow$ 4 muons: the cleanest (“golden”) signature



*)LHCb @2x10^{33} cm^{-2}-1 isn’t much nicer and in Alice (PbPb) it will be even worse

Mother Nature is a … Kind Woman After All

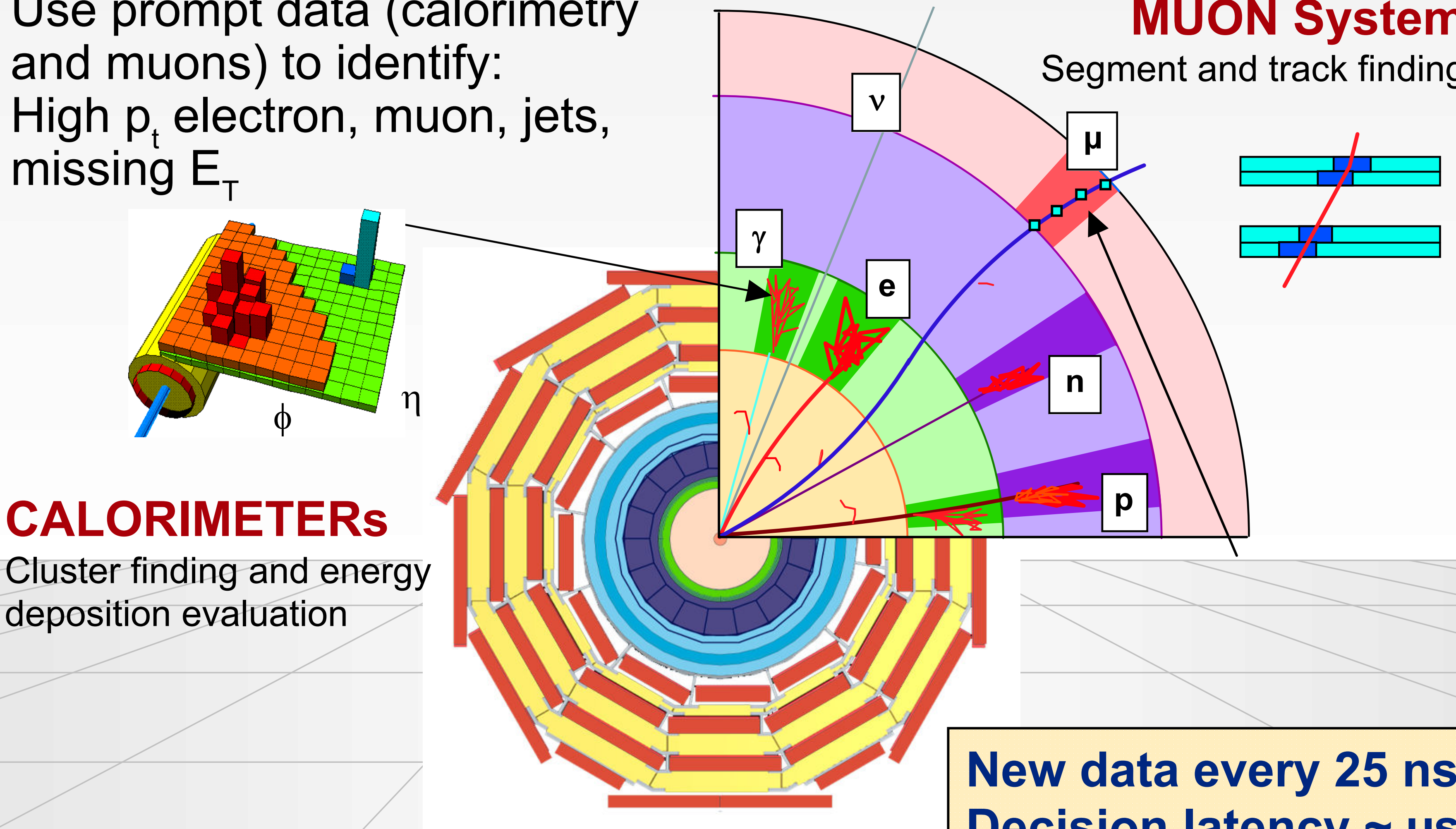
- pp collisions produce mainly hadrons with transverse momentum "p_t" ~1 GeV
- Interesting physics (old and new) has particles (leptons and hadrons) with large p_t:
  - $W \rightarrow e\nu$: $M(W)=80$ GeV/c^2; $p_t(e)$ ~ 30-40 GeV
  - H(120 GeV)$\rightarrow\gamma\gamma$: $p_t(\gamma)$ ~ 50-60 GeV
  - $B \rightarrow \mu D^{*+}\nu$ $p_t(\mu)$ ~ 1.4 GeV



- Impose high thresholds on the p_t of particles
  - Implies distinguishing particle types; possible for electrons, muons and "jets"; beyond that, need complex algorithms
- Conclusion: in the L1 trigger we need to watch out for high transverse momentum electrons, jets or muons

How to defeat minimum bias: transverse momentum p_t

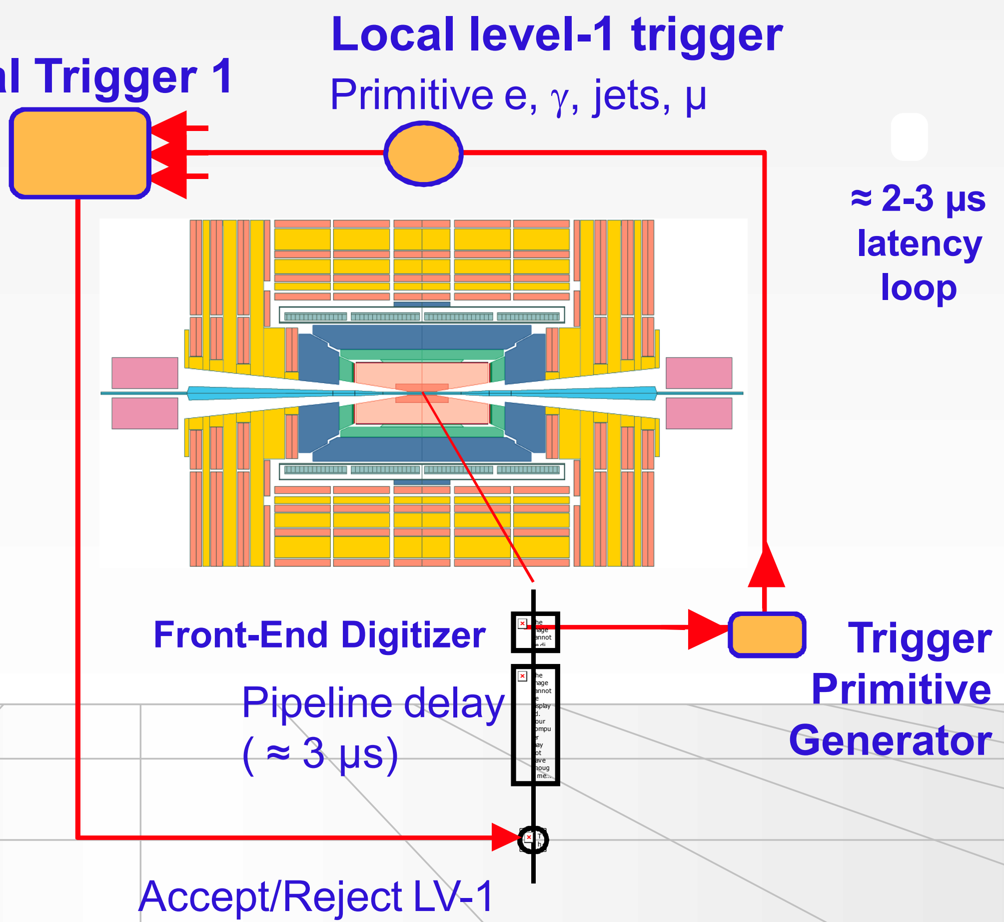
Use prompt data (calorimetry and muons) to identify:
High p_t electron, muon, jets, missing E_T



New data every 25 ns
Decision latency ~ μs

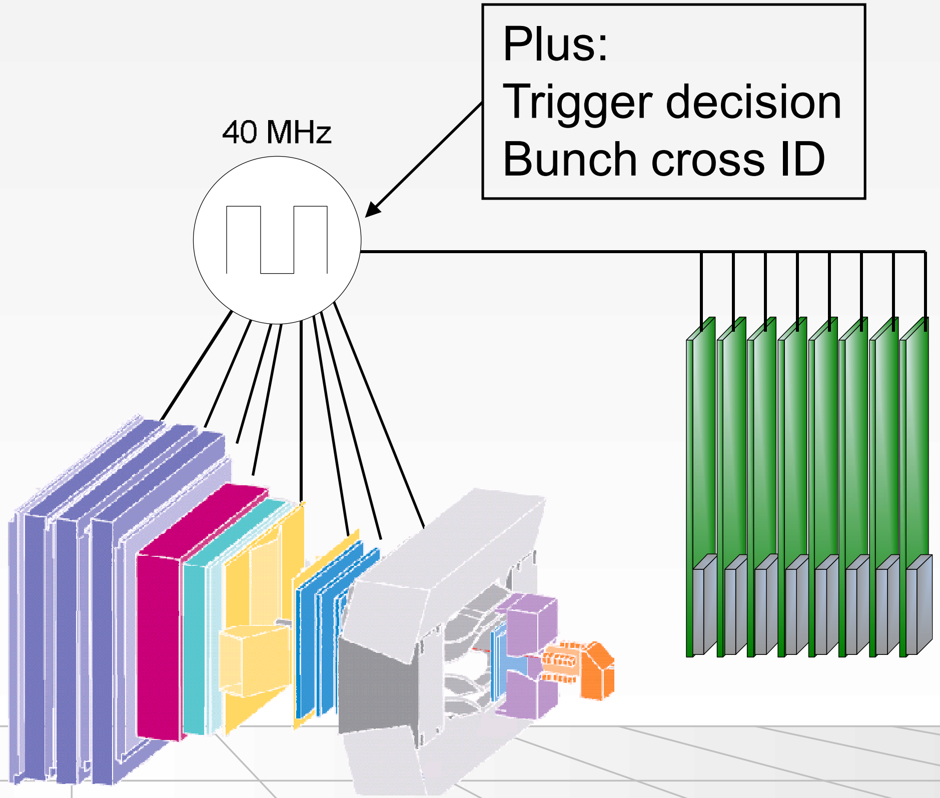
Distributing the L1 Trigger

- Assuming now that a magic box tells for each bunch crossing (clock-tick) yes or no
  - Triggering is not for philosophers – "perhaps" is not an option
- 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

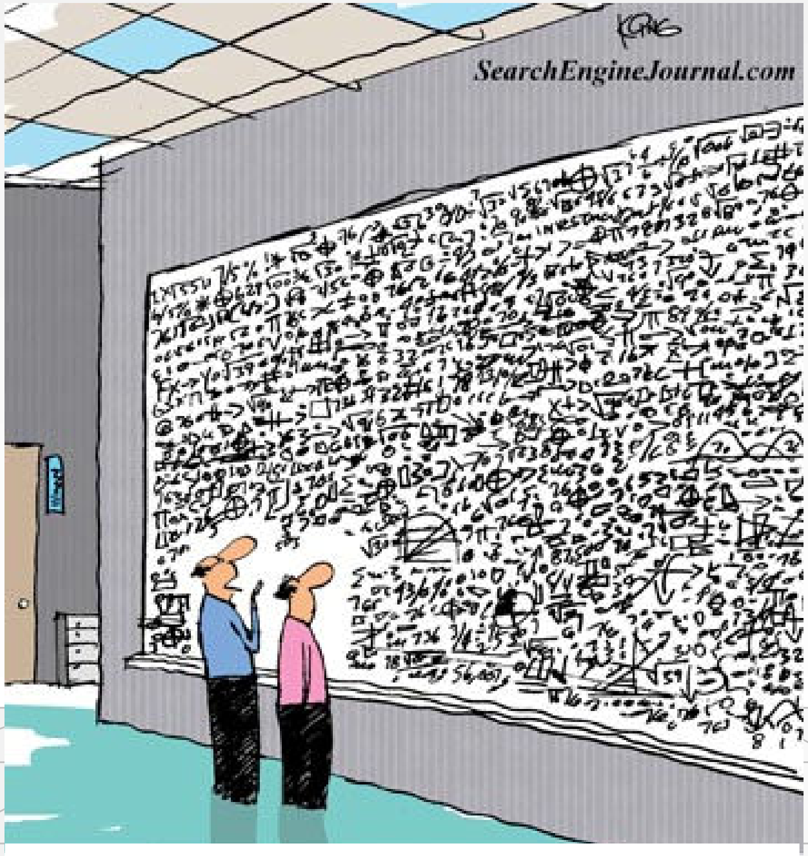


Clock Distribution and Synchronisation

- 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 the c is constant, the detectors are large and the electronics is fast, the detector elements must be carefully time-aligned
- Common system for all LHC experiments TTC based on radiation-hard opto-electronics

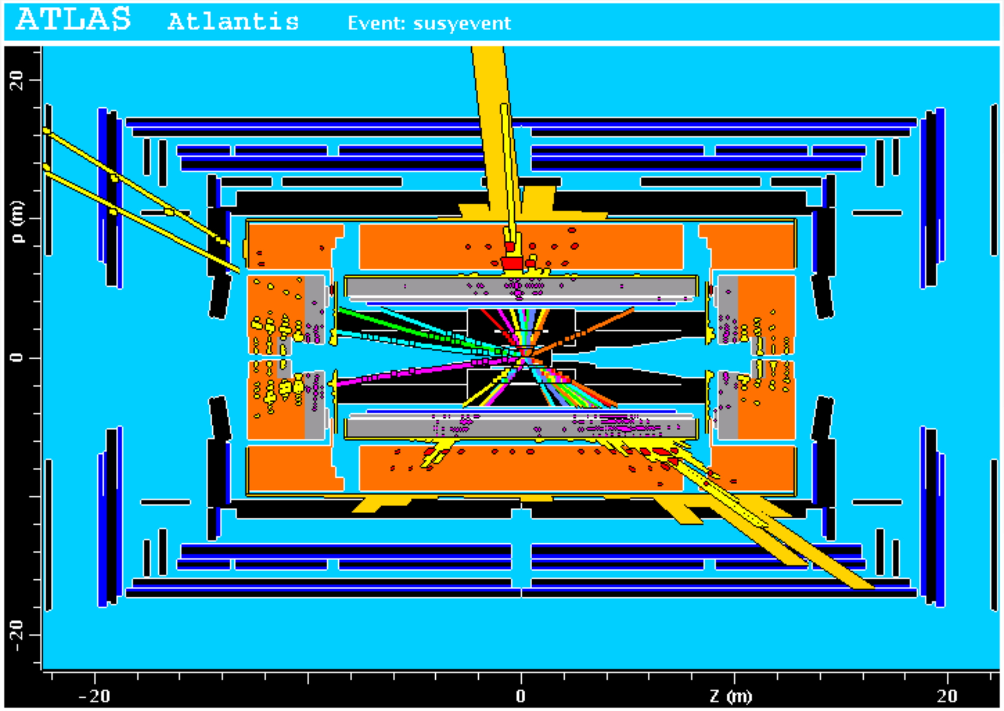


High Level Trigger

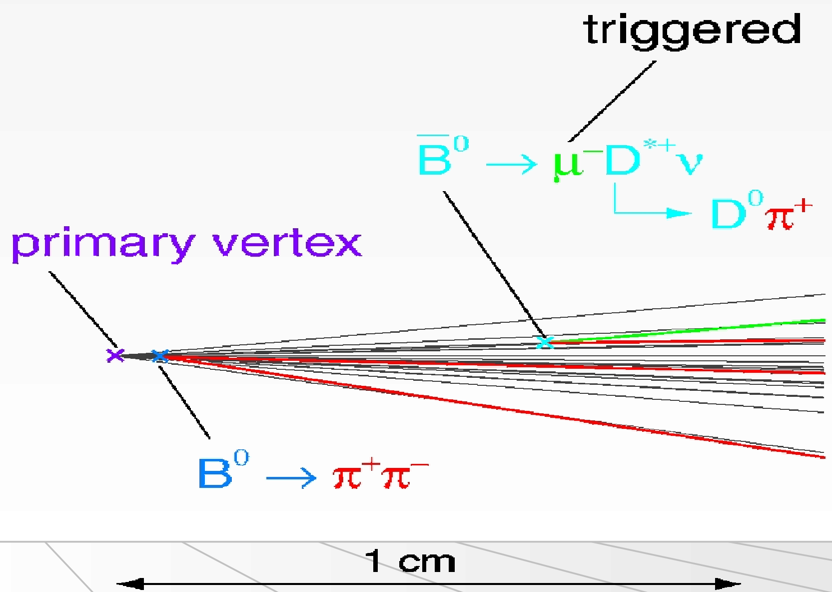


And that, in simple terms, is what we do in the High Level Trigger

High Level Trigger



Complicated Event structure with hadronic jets (ATLAS) or secondary vertices (LHCb) require full detector information



Methods and algorithms are the same as for offline reconstruction (Lecture "From raw data to physics")

Online Trigger Farms 2009

	ALICE	ATLAS	CMS	LHCb	CERN IT
# servers	81 (1)	837	900	550	5700
# cores	324	~ 6400	7200	4400	~ 34600
total available power (kW)		~ 2000(2)	~ 1000	550	2.9 MW
currently used power (kW)		~ 250	450(3)	~ 145	2.0 MW
total available cooling power	~ 500	~ 820	800 (currently)	525	2.9 MW
total available rack-space (Us)	~ 2000	2449	~ 3600	2200	n/a
CPU type(s)	AMD Opteron	Intel Hapertown	Intel (mostly) Harpertown	Intel Harpertown	Mixed (Intel)

(1) 4-U servers with powerful FPGA preprocessor cards H-RORC
(2) Available from transformer (3) PSU rating

Large DAQ

Data Acquisition

- Event-data are now digitized, pre-processed and tagged with a unique, monotonically increasing number
- The event data are distributed over many *read-out boards* ("sources")
- For the next stage of selection, or even simply to write it to tape we have to get the pieces together: enter the DAQ

Network based DAQ

- In large (HEP) experiments we typically have thousands of devices to read, which are sometimes very far from each other → *buses can not do that*
- Network technology solves the scalability issues of buses
  - In a network devices are equal ("peers")
  - In a network devices communicate directly with each other
    - no arbitration necessary
    - bandwidth guaranteed
  - data and control use the same path
    - much fewer lines (e.g. in traditional Ethernet only two)
  - At the signaling level buses tend to use parallel copper lines. Network technologies can be also optical, wire-less and are typically (differential) serial

Network Technologies

- Examples:
  - The telephone network
  - Ethernet (IEEE 802.3)
  - ATM (the backbone for GSM cell-phones)
  - Infiniband
  - Myrinet
  - many, many more
- Note: some of these have "bus"-features as well (Ethernet, Infiniband)
- Network technologies are sometimes functionally grouped
  - Cluster interconnect (Myrinet, Infiniband) 15 m
  - Local area network (Ethernet), 100 m to 10 km
  - Wide area network (ATM, SONET) > 50 km

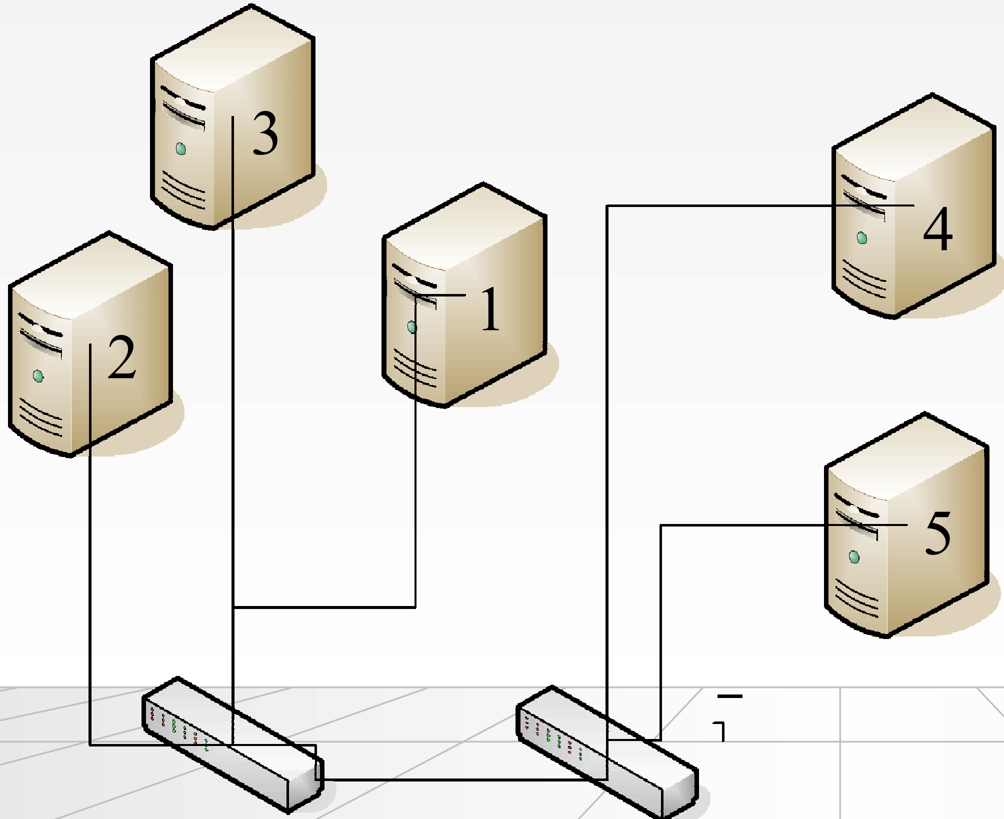
Connecting Devices in a Network

- On an network a device is identifed by a network address
  - eg: our phone-number, the MAC address of your computer
- Devices communicate by sending messages (frames, packets) to each other
- Some establish a connection like the telephone network, some simply send messages
- Modern networks are *switched with point-to-point links*
  - circuit switching, packet switching

Switched Networks

- In a switched network each node is connected either to another node or to a switch
- Switches can be connected to other switches
- A path from one node to another leads through 1 or more switches (this number is sometimes referred to as the number of "hops")

A Switched Network



- While 2 can send data to 1 and 4, 3 can send at full speed to 5
- 2 can distribute the share the bandwidth between 1 and 4 as needed

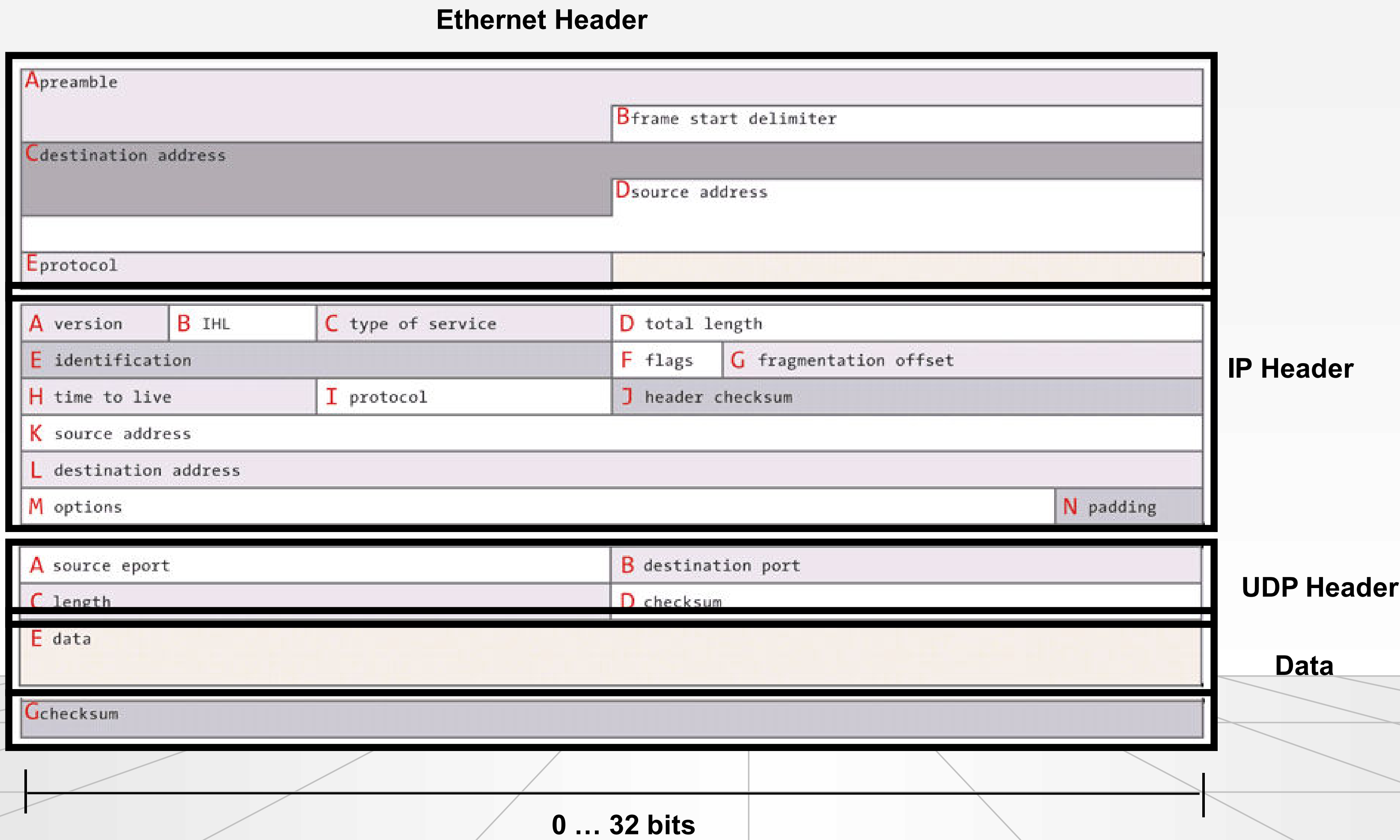
Switches

- Switches are the key to good network performance
- They must move frames reliably and as fast as possible between nodes
- They face two problems
  - Finding the right path for a frame
  - Handling congestion (two or more frames want to go to the same destination at the same time)

Ethernet

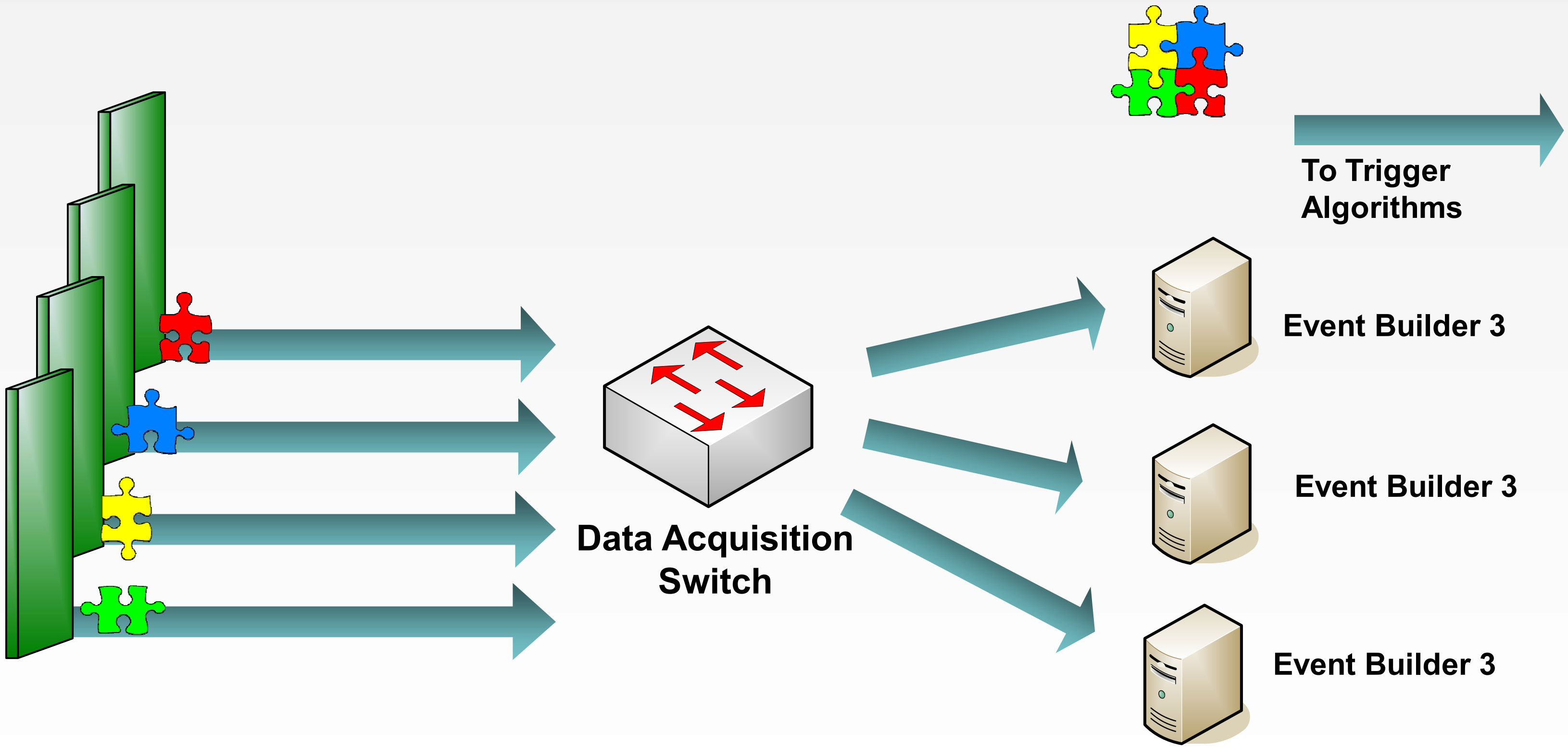
- Cheap
- Unreliable – but in practice transmission errors are very low
- Available in many different speeds and physical media
- We use IP or TCP/IP over Ethernet
- By far the most widely used local area network technology (even starting on the WAN)

IP Packets over Ethernet



Event Building

Event Building



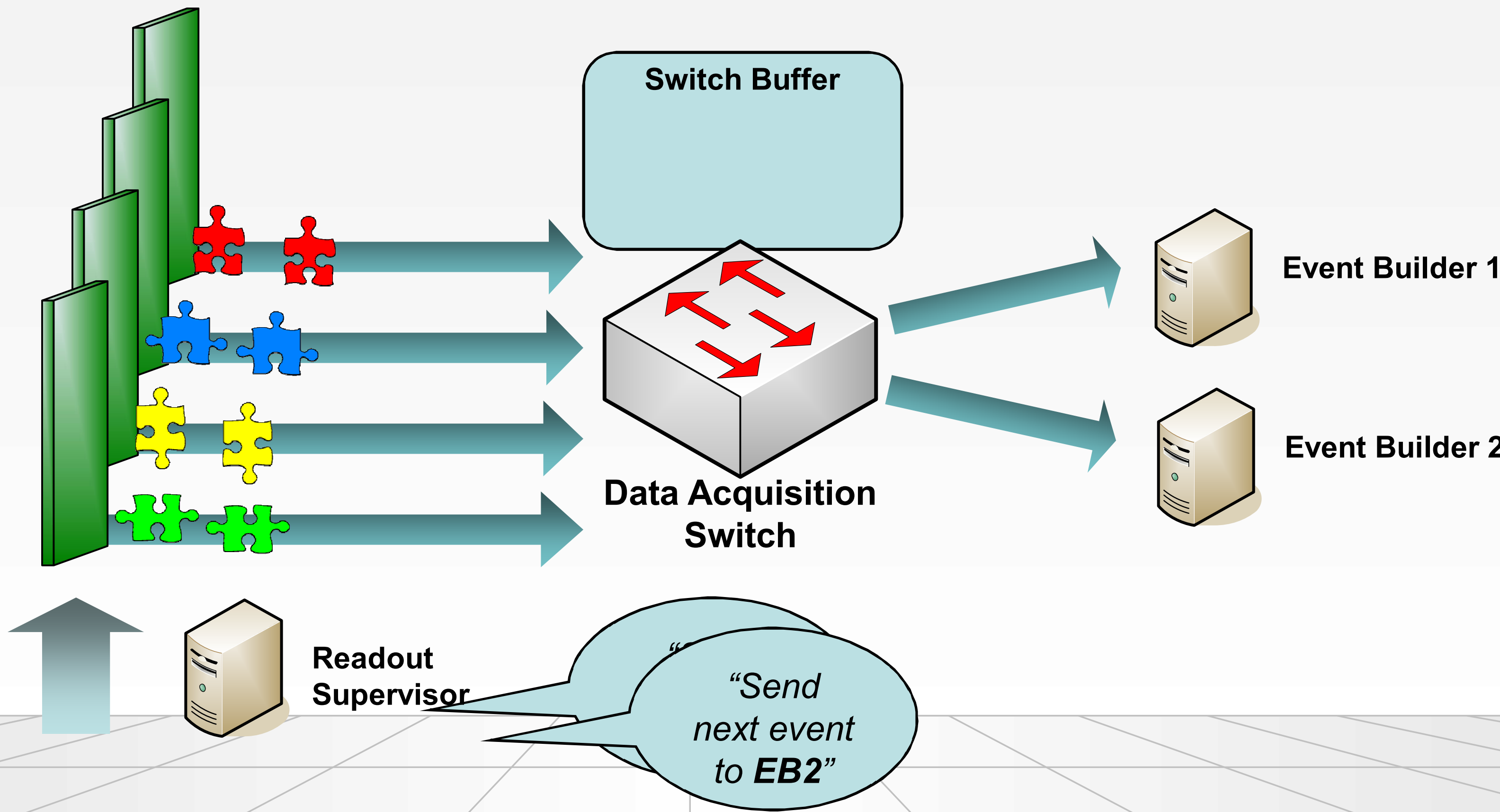
1 Event fragments are received from detector front-end

2 Event fragments are read out over a network to an event builder

3 Event builder assembles fragments into a complete event

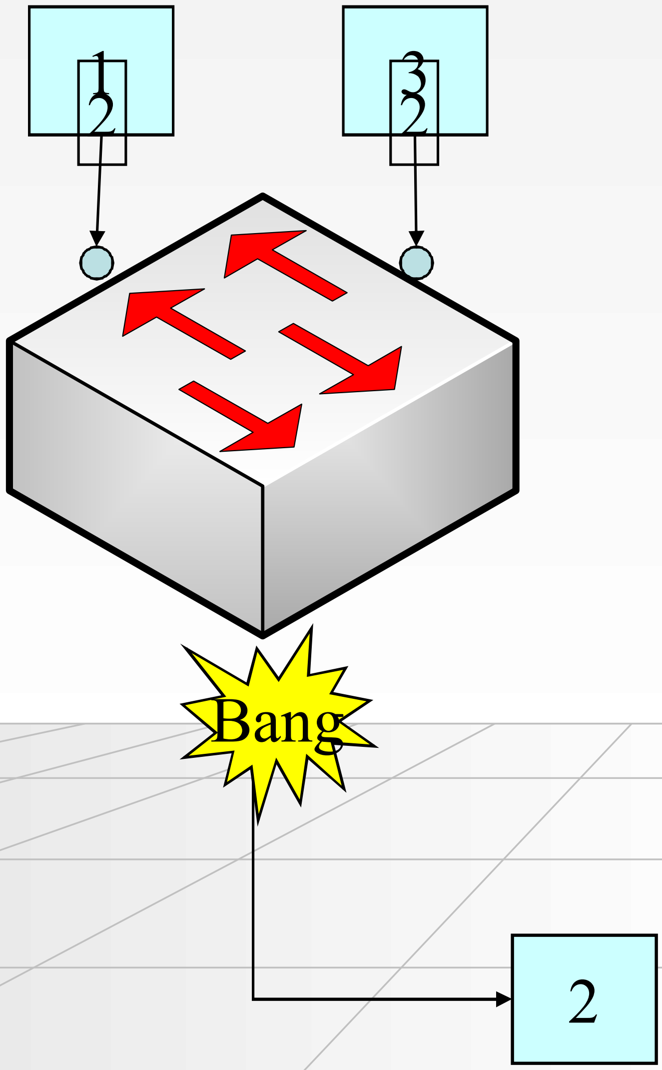
4 Complete events are processed by trigger algorithms

Push-Based Event Building



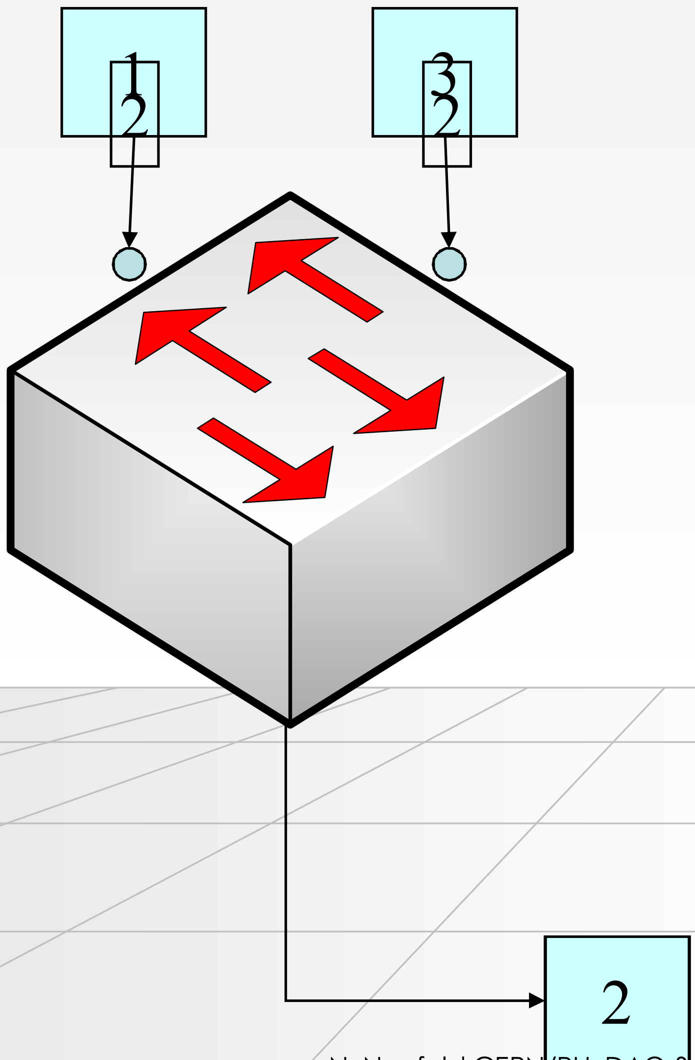
1. **Readout Supervisor tells readout boards where events must be sent (round-robin)**
2. **Readout boards do not buffer, so switch must**
3. **No feedback from Event Builders to Readout system**

Congestion



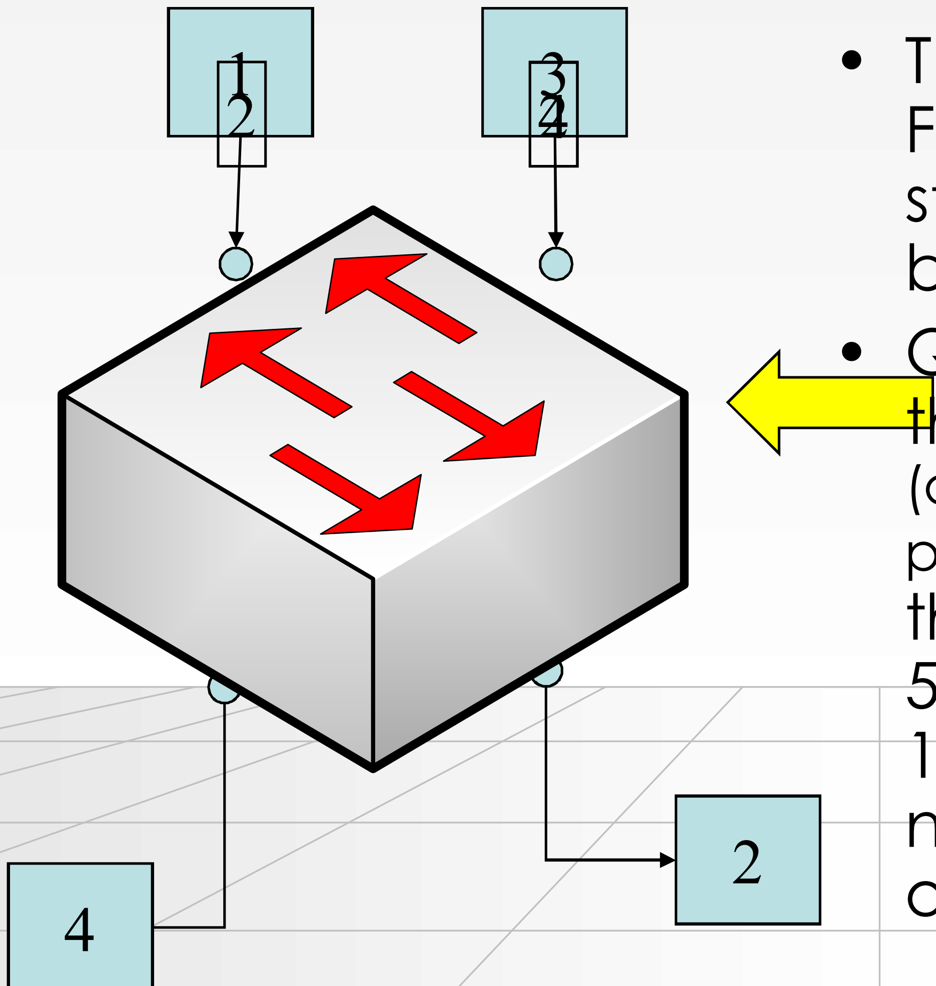
- "Bang" translates into random, uncontrolled packet-loss
- In Ethernet this is perfectly valid behavior and implemented by very cheap devices
- Higher Level protocols are supposed to handle the packet loss due to *lack of buffering*
- This problem comes from ***synchronized*** sources ***sending*** to the same destination at the ***same time***

Overcoming Congestion: Queuing at the Input



- Two frames destined to the same destination arrive
- While one is switched through the other is waiting at the input port
- When the output port is free the queued packet is sent

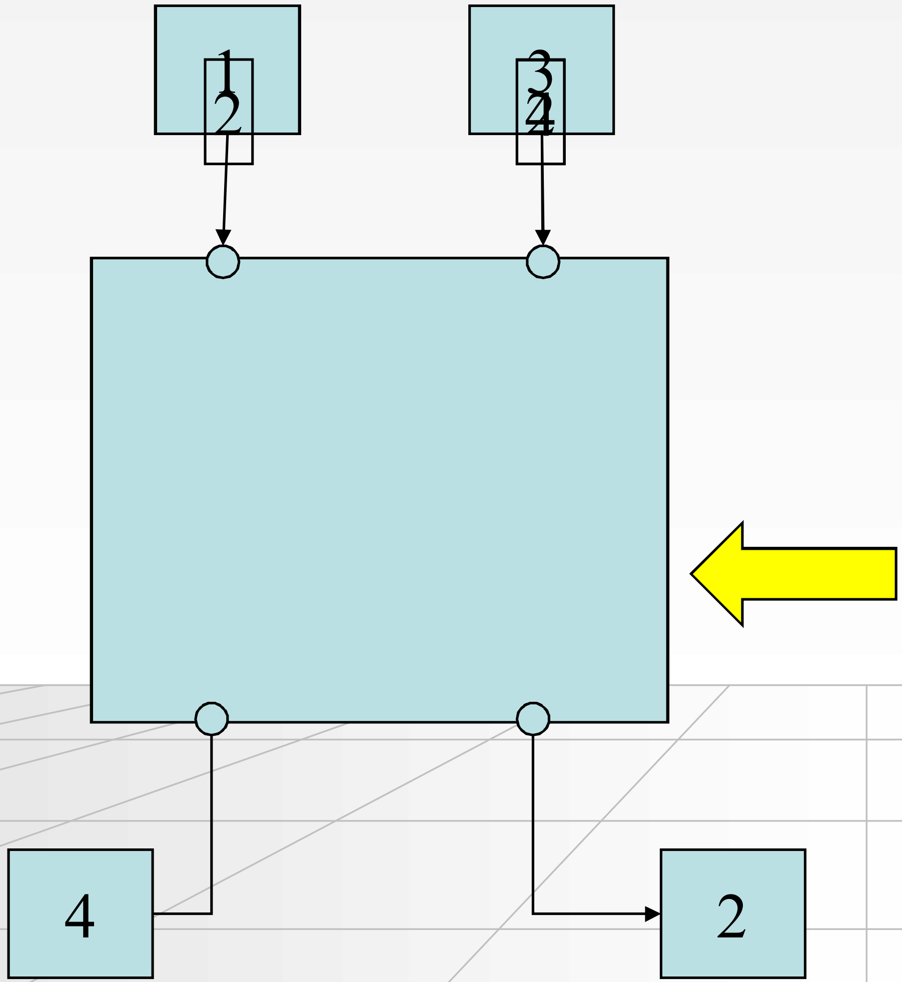
Head of Line Blocking



- The reason for this is the First in First Out (FIFO) structure of the input buffer
- Queuing theory tells us* that for random traffic (and infinitely many switch ports) the throughput of the switch will go down to 58.6% → that means on 100 MBit/s network the nodes will "see" effectively only ~ 58 MBit/s

*) "Input Versus Output Queueing on a Space-Division Packet Switch"; Karol, M. et al. ; IEEE Trans. Comm., 35/12

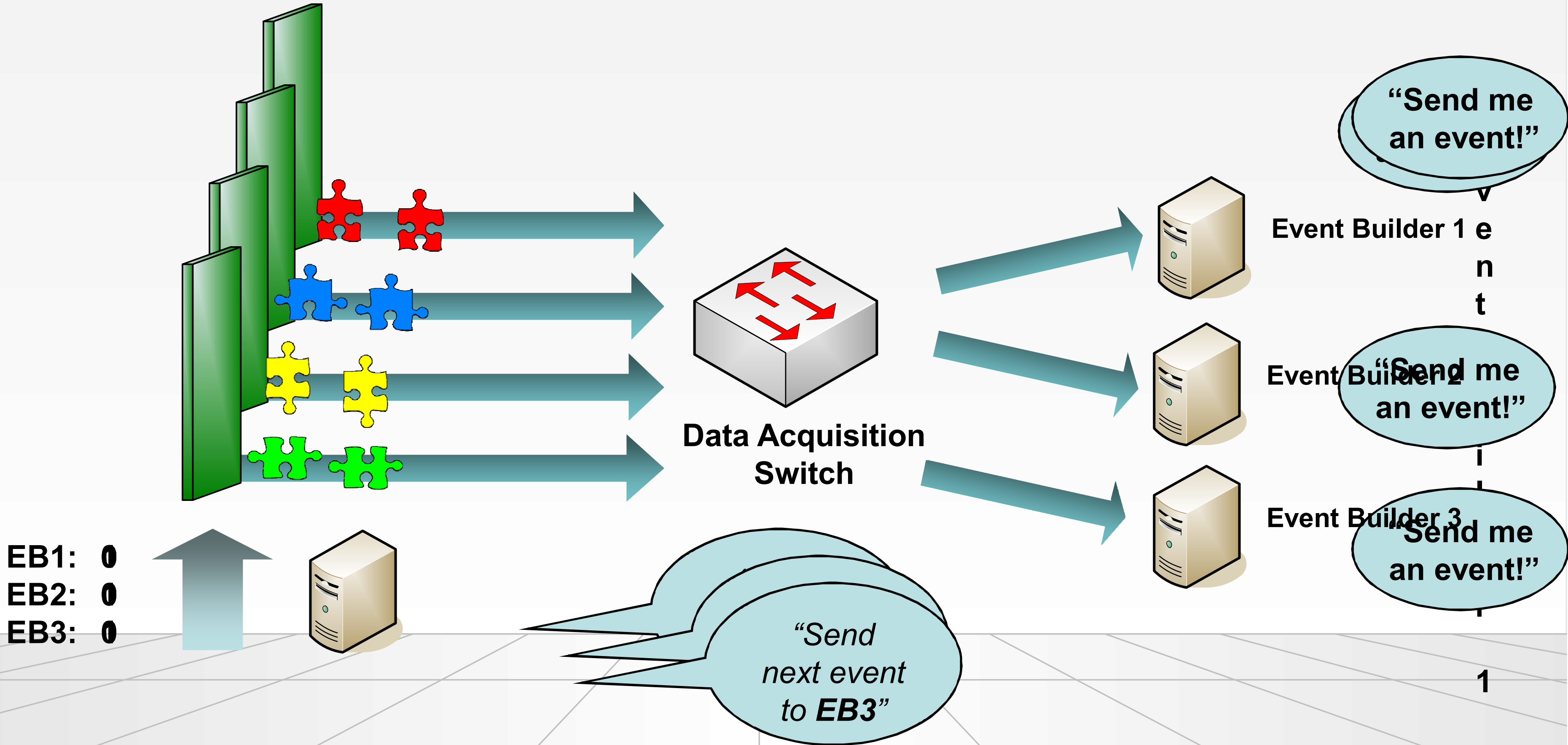
Output Queuing



- In practice virtual output queueing is used: at each input there is a queue → for n ports $O(n^2)$ queues must be managed
- Assuming the buffers are large enough(!) such a switch will sustain random traffic at 100% nominal link load

Packet to node 2 waits at output to port 2. Way to node 4 is free

Pull-Based Event Building



1 Event Builders notify Readout Supervisor of available capacity

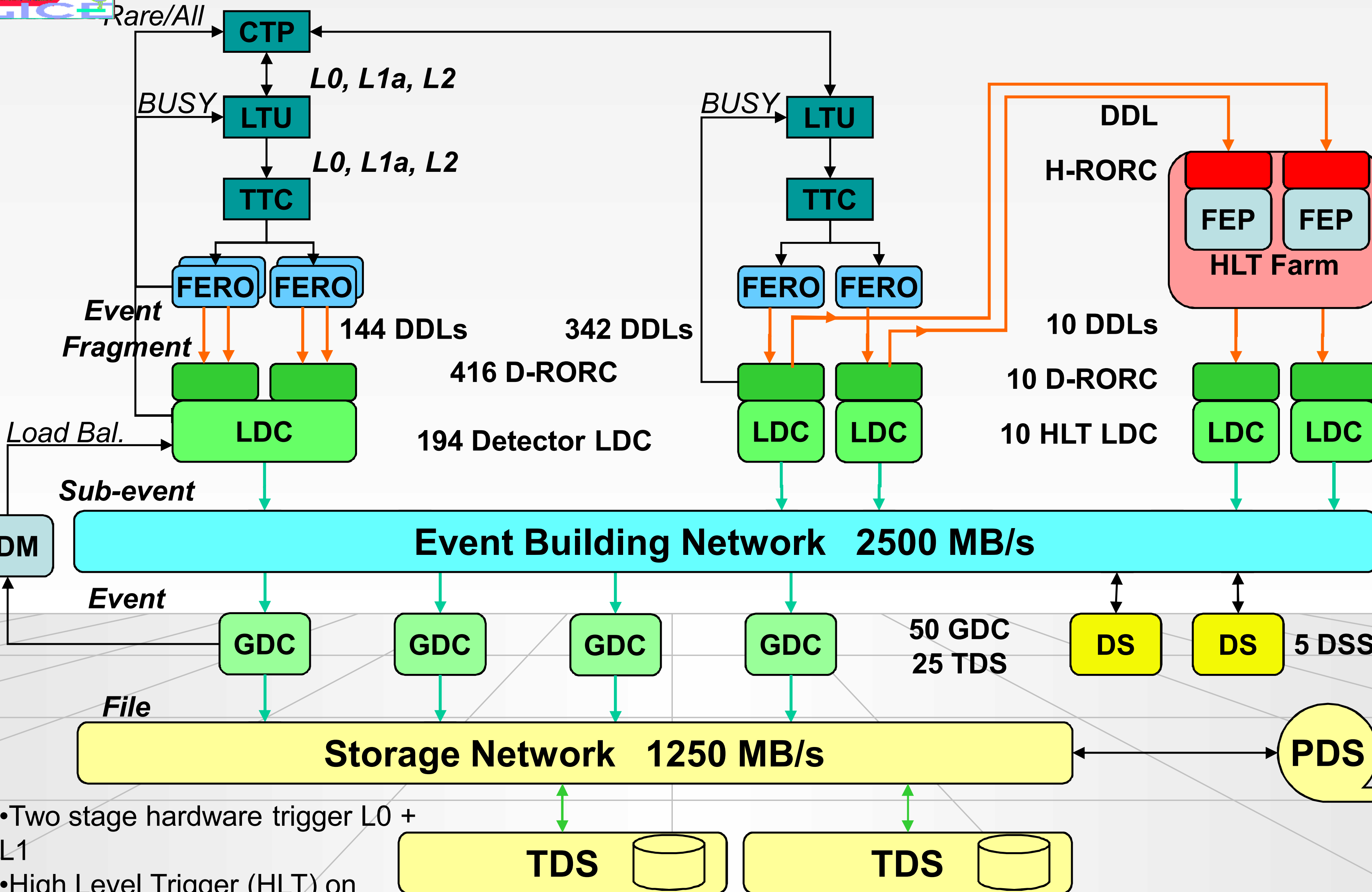
2 Readout Supervisor ensures that data are sent only to nodes with available capacity

3 Readout system relies on feedback from Event Builders

AACL

ALICE, ATLAS, CMS, LHCb
DAQs in 4 slides

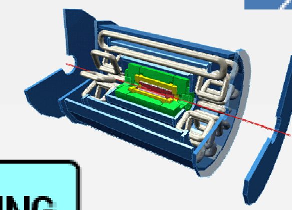
ALICE DAQ



- Two stage hardware trigger L0 + L1
- High Level Trigger (HLT) on separate farm

ATLAS DAQ

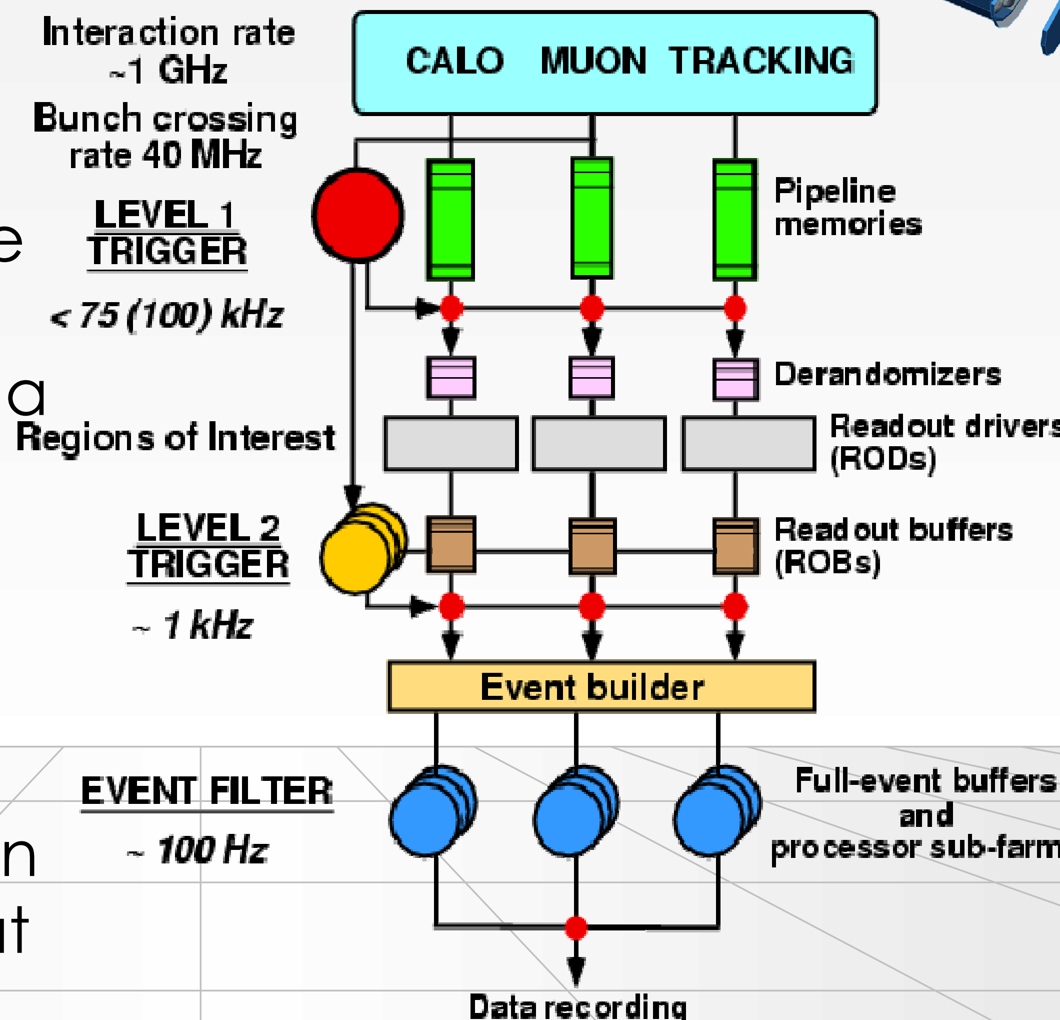
ATLAS



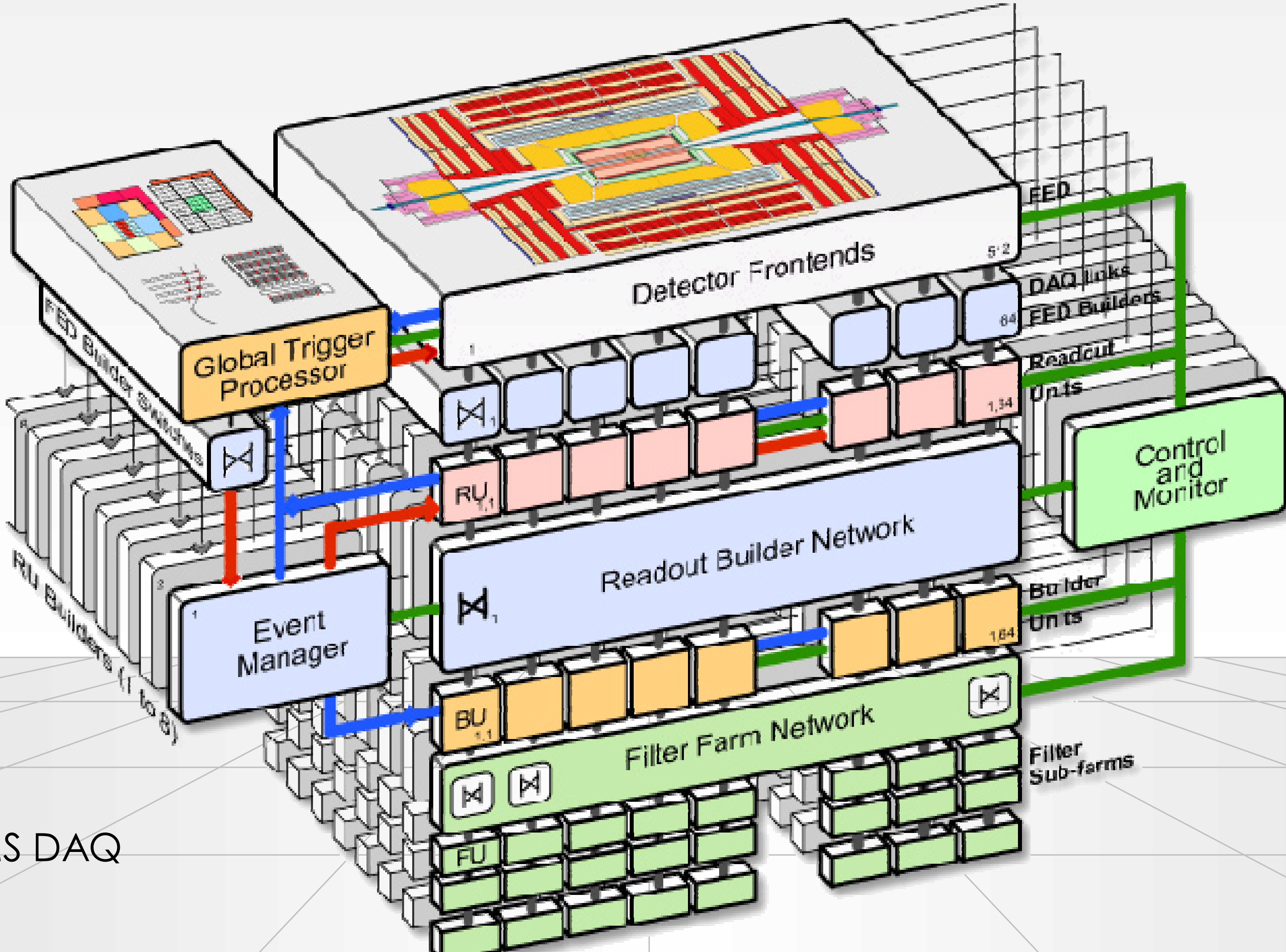
- L1 selects events at 100 kHz and defines *regions of interest*
- L2 pulls data from the region of interest and processes the data in a farm of processors

L2 accepts data at ~ 1 kHz

- Event Filter reads the entire detector (pull), processes the events in a farm and accepts at 100 Hz

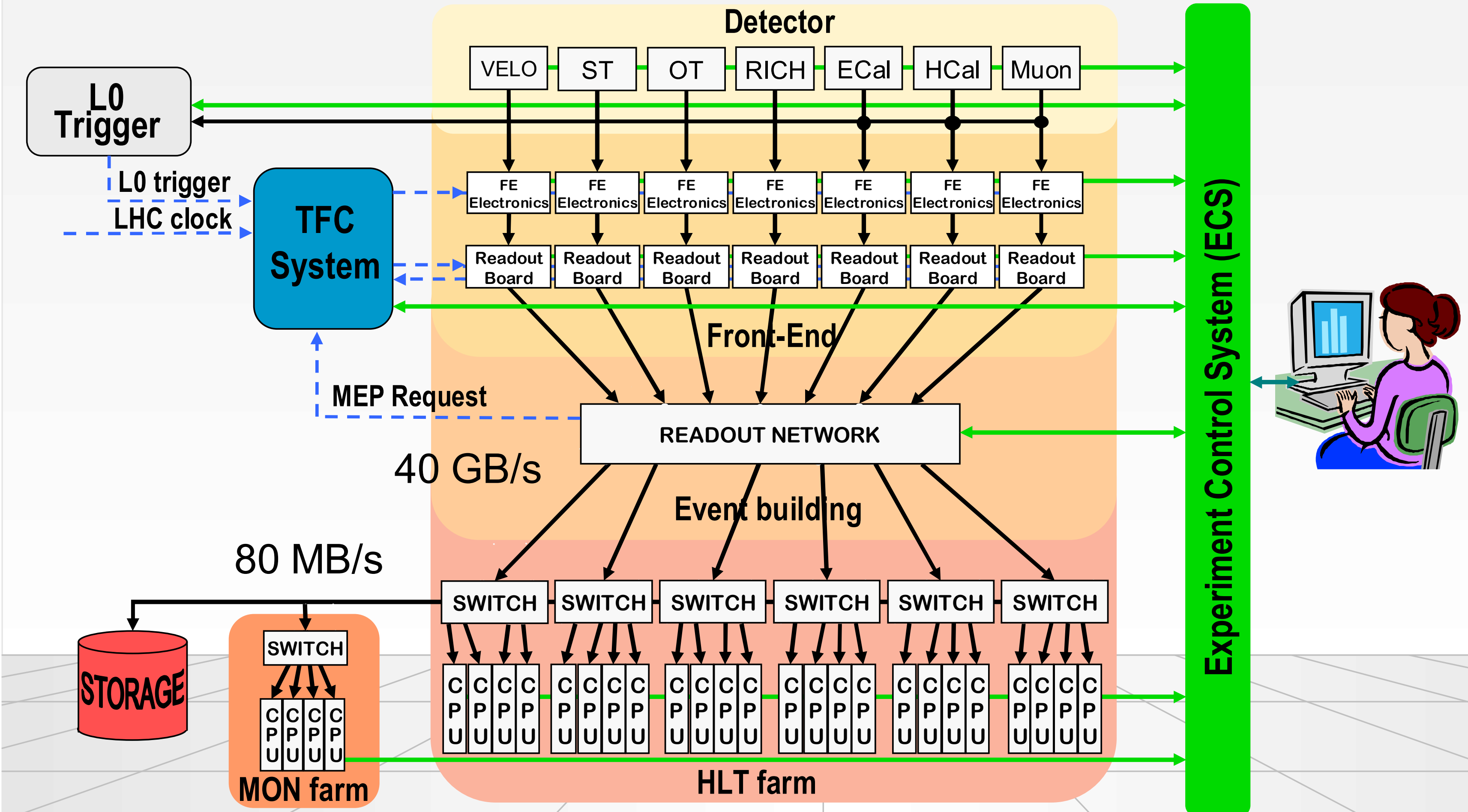


Readout Architectures



CMS DAQ

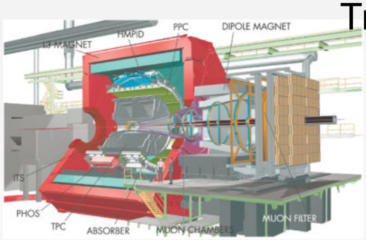
LHCb DAQ

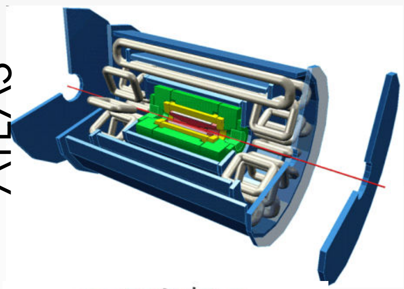


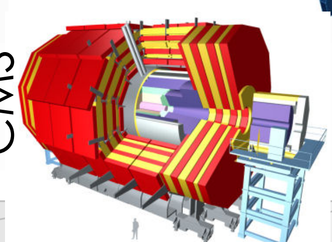
Event data
Timing and Fast Control Signals
Control and Monitoring data

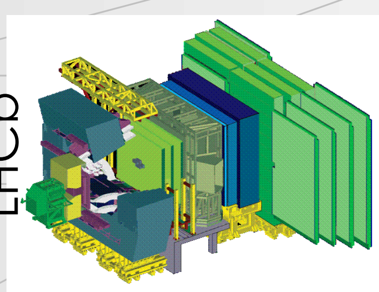
Average event size 40 kB
Average rate into farm 1 MHz
Average rate to tape 2 kHz

Trigger/DAQ parameters



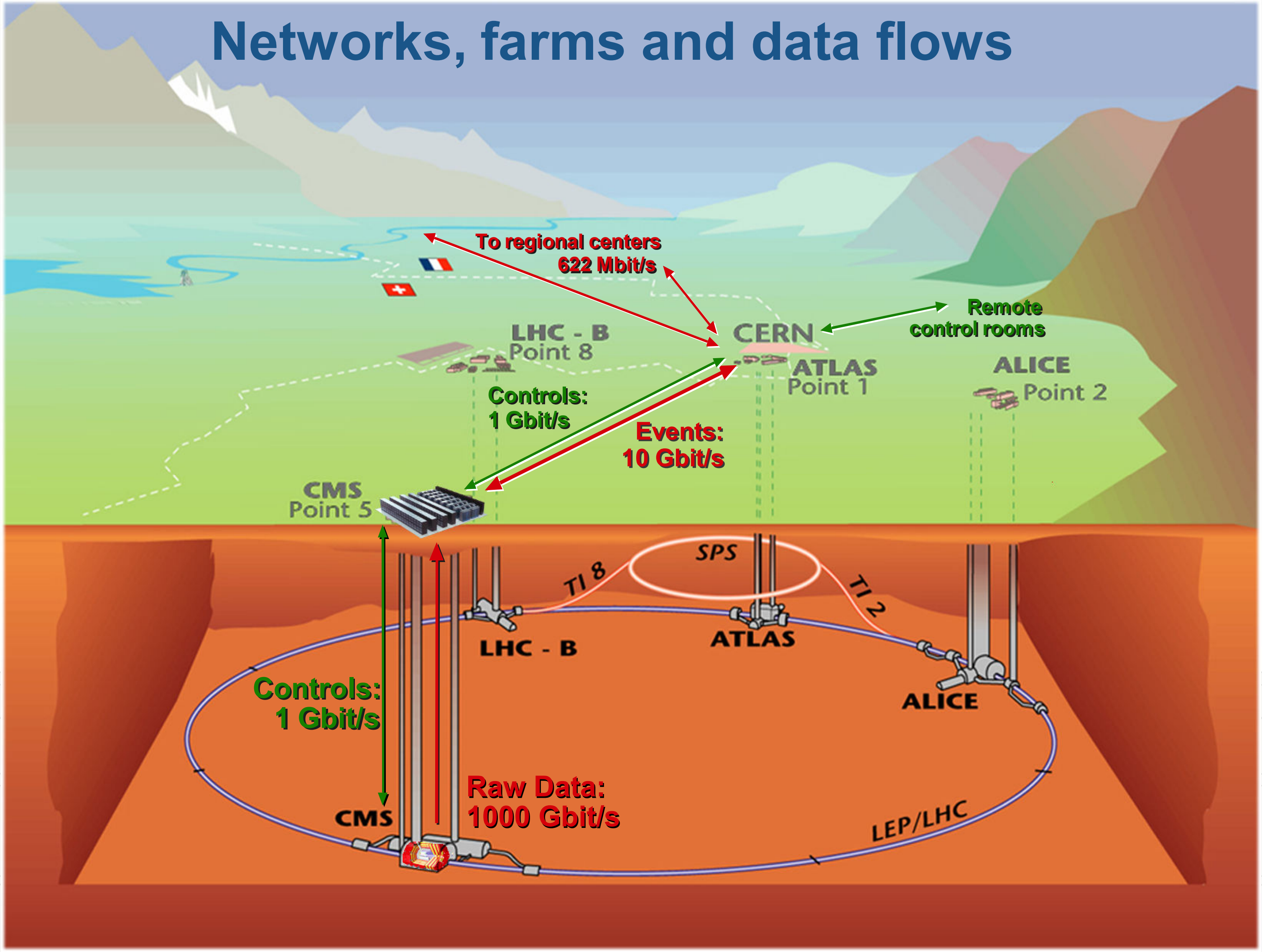






	No.Levels Trigger	**Level-0,1,2** Rate (Hz)	**Event** Size (Byte)	**Readout** Bandw.(GB/s)	**HLT Out** MB/s (Event/s)
ALICE	**4**	Pb-Pb **500** p-p **10^3**	**5×10^7** **2×10^6**	**25**	**1250** (10^2) **200** (10^2)
ATLAS	**3**	LV-1 **10^5** LV-2 **3×10^3**	**1.5×10^6**	**4.5**	**300** (2×10^2)
CMS	**2**	LV-1 **10^5**	**10^6**	**100**	**~1000** (10^2)
LHCb	**2**	LV-0 **10^6**	**3.5×10^4**	**35**	**70** (2×10^3)

On to tape...and the GRID



Further Reading

- Buses
  - VME: http://www.vita.com/
  - PCI
    http://www.pcisig.com/
- Network and Protocols
  - Ethernet
    "Ethernet: The Definitive Guide", O'Reilly, C. Spurgeon
  - TCP/IP
    "TCP/IP Illustrated", W. R. Stevens
  - Protocols: RFCs
    www.ietf.org
    in particular RFC1925
    http://www.ietf.org/rfc/rfc1925.txt
    "The 12 networking truths" is required reading
- Wikipedia (!!!) and references therein – for all computing related stuff this is usually excellent
- Conferences
  - IEEE Realtime
  - ICALEPCS
  - CHEP
  - IEEE NSS-MIC
- Journals
  - IEEE Transactions on Nuclear Science, in particular the proceedings of the IEEE Realtime conferences
  - IEEE Transactions on Communications

More Stuff

Data format, DIY DAQ, run-control

Raw data format

There are 10 kinds of people in the world


```
0000240 2828 2828 2828 2828 c4ff a201 0000 0501
0000260 0101 0101 0101 0000 0000 0000 0000 0201
0000300 0403 0605 0807 0a09 010b 0300 0101 0101
0000320 0101 0101 0001 0000 0000 0100 0302 0504
0000340 0706 0908 0b0a 0010 0102 0303 0402 0503
0000360 0405 0004 0100 017d 0302 0400 0511 2112
0000400 4131 1306 6151 2207 1471 8132 a191 2308
0000420 b142 15c1 d152 24f0 6233 8272 0a09 1716
0000440 1918 251a 2726 2928 342a 3635 3837 3a39
0000460 4443 4645 4847 4a49 5453 5655 5857 5a59
0000500 6463 6665 6867 6a69 7473 7675 7877 7a79
0000520 8483 8685 8887 8a89 9392 9594 9796 9998
0000540 a29a a4a3 a6a5 a8a7 aaa9 b3b2 b5b4 b7b6
```



```
<ADCVALUE>
<TIME>00:04:10</TIME>
<VALUE>0.2334</VALUE>
<PCISTATUS>OK</PCISTATUS>
</ADCVALUE>
<ADCVALUE>
<TIME>00:05:10</TIME>
<VALUE>0.9999</VALUE>
<PCISTATUS>ERROR</PCISTATUS>
</ADCVALUE>
<ADCVALUE>
<TIME>00:06:10</TIME>
<VALUE>0.6334</VALUE>
<PCISTATUS>OK</PCISTATUS>
</ADCVALUE>
<ADCVALUE>
<TIME>00:07:10</TIME>
<VALUE>0.8334</VALUE>
<PCISTATUS>OK</PCISTATUS>
</ADCVALUE>
```

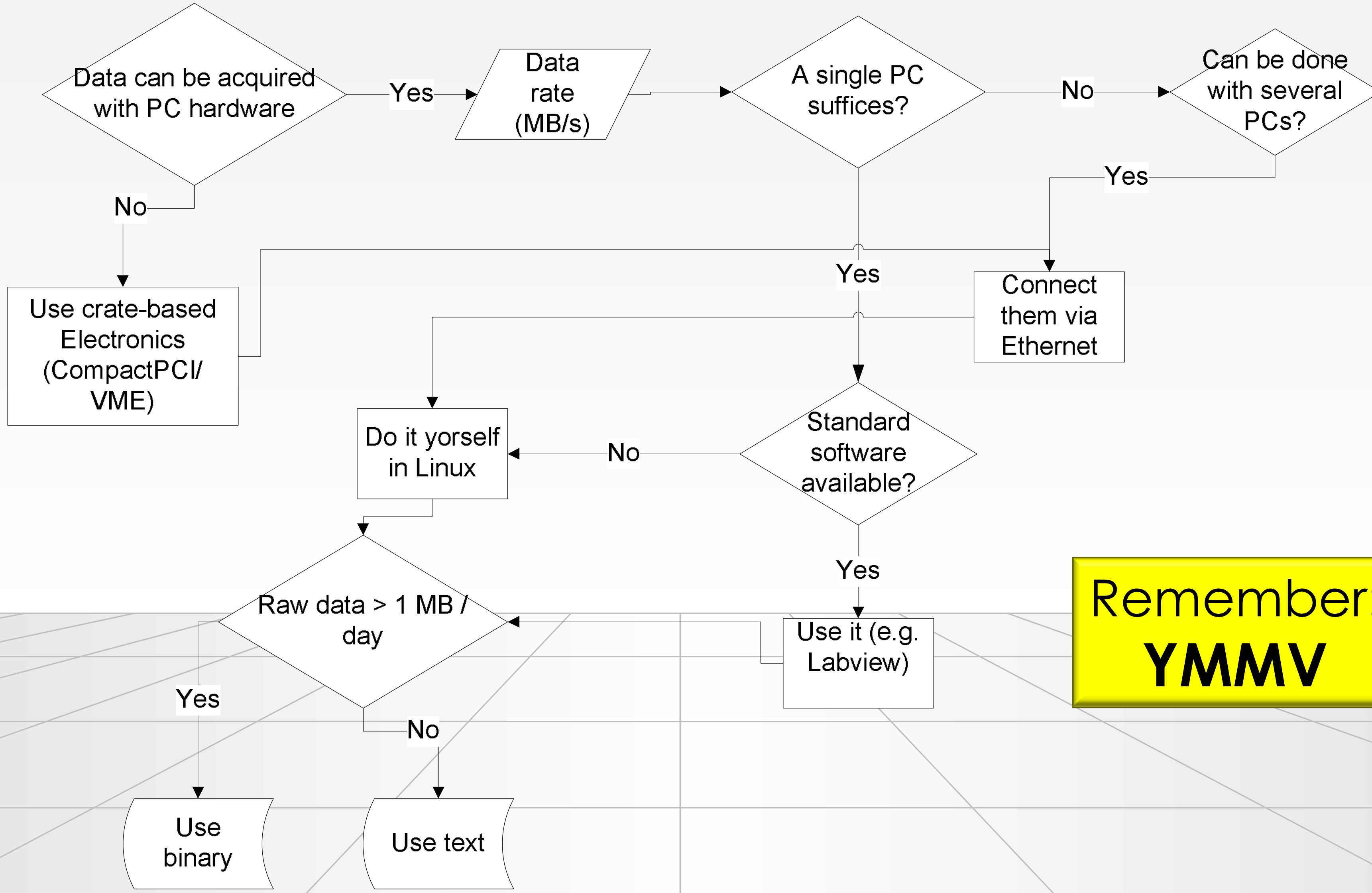

Those who can read binary and those who cannot

Binary vs Text

- 11010110 Pros:
  - compact
  - quick to write & read (no conversion)
- Cons:
  - opaque (humans need tool to read it)
  - depends on the machine architecture (endinaness, floating point format)
  - life-time bound to availability of software which can read it

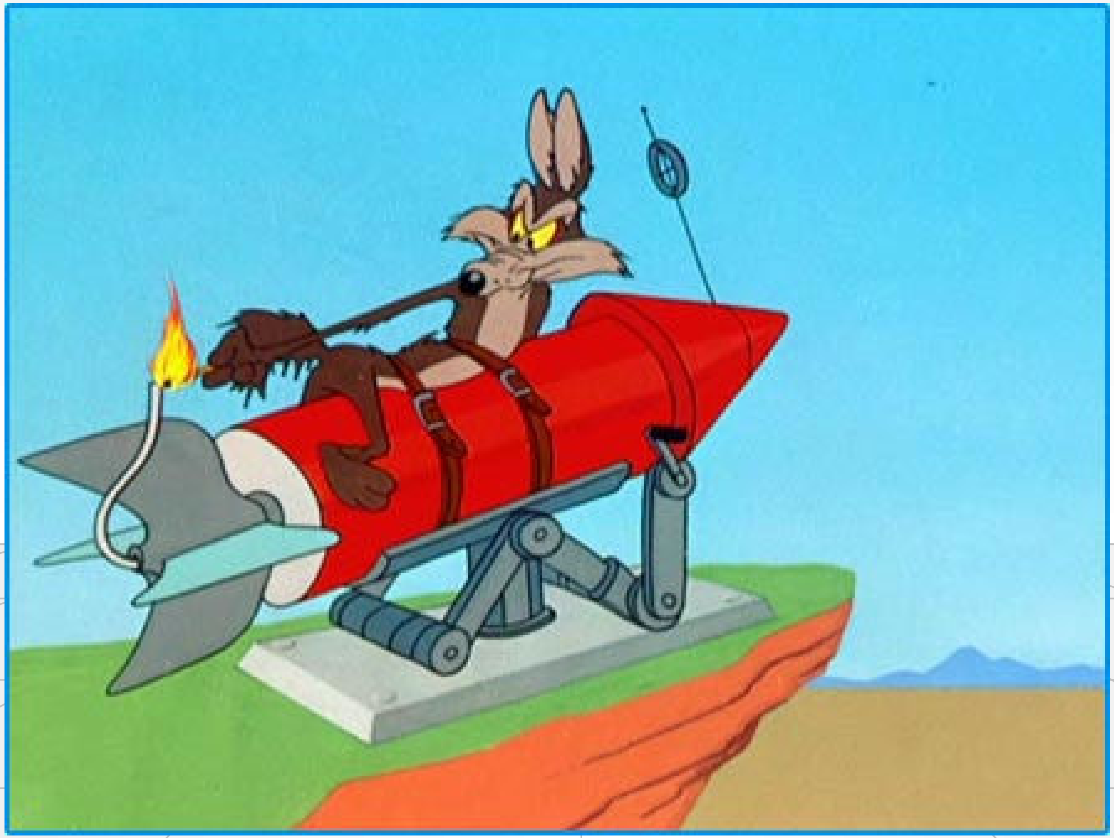
- <TEXT></TEXT> Pros:
  - universally readable
  - can be parsed and edited equally easily by humans and machines
  - long-lived (ASCII has not changed over decades)
  - machine independent
- Cons:
  - slow to read/write
  - low information density (can be improved by compression)

A little checklist for your DAQ



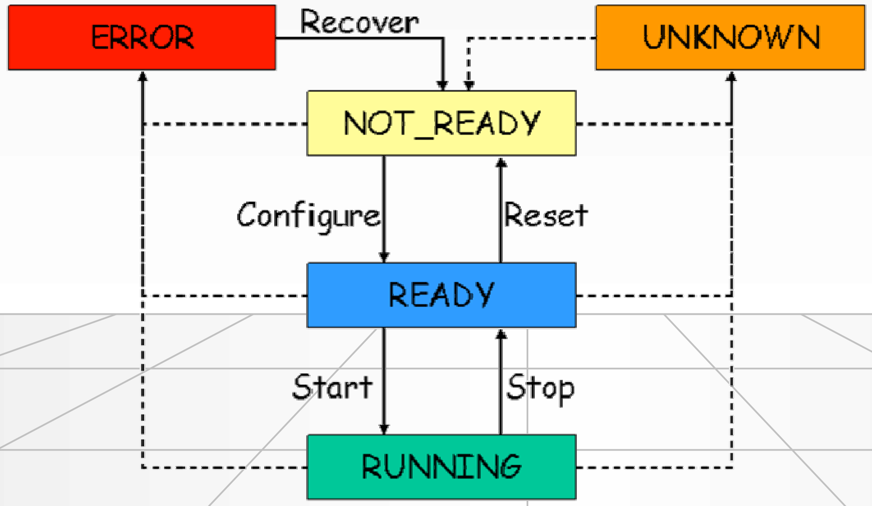
Remember: **YMMV**

Runcontrol



Run Control

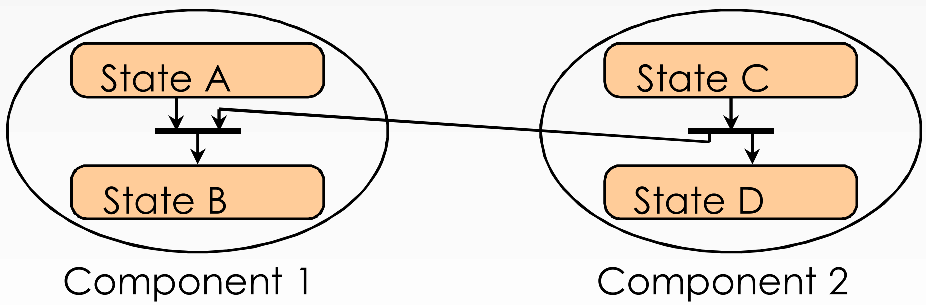
- The run controller provides the control of the trigger and data acquisition system. It is the application that interacts with the operator in charge of running the experiment.
- The operator is not always an expert on T/DAQ. The **user interface** on the Run Controller plays an important role.
- The complete system is modeled as a **finite state machine**. The commands that run controller offers to the operator are state transitions.



LHCb DAQ /Trigger Finite State Machine diagram (simplified)

Finite State Machine

- Each component, sub-component of the system is modeled as a *Finite State Machine*. This abstraction facilitates the description of each component behavior without going into detail
- The control of the system is realized by inducing transitions on remote components due to a transition on a local component



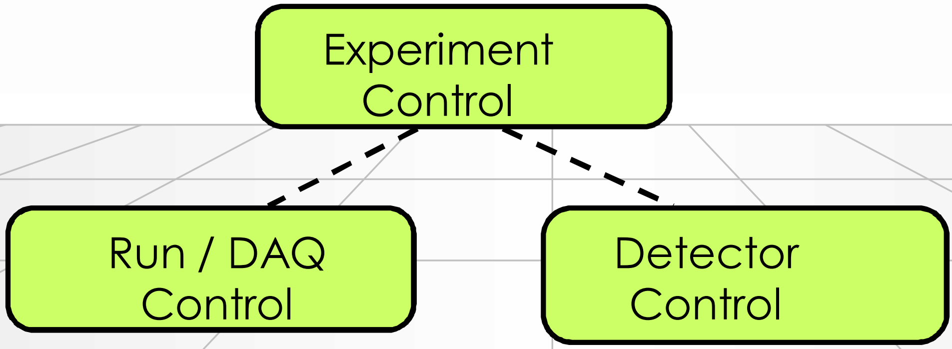
Component 1 can only complete the transition to State B if Component 2 is in state D.

- Each transition may have actions associated. The action consist of code which needs to be executed in order to bring the component to its new state
- The functionality of the FSM and state propagation is available in special software packages such as SMI

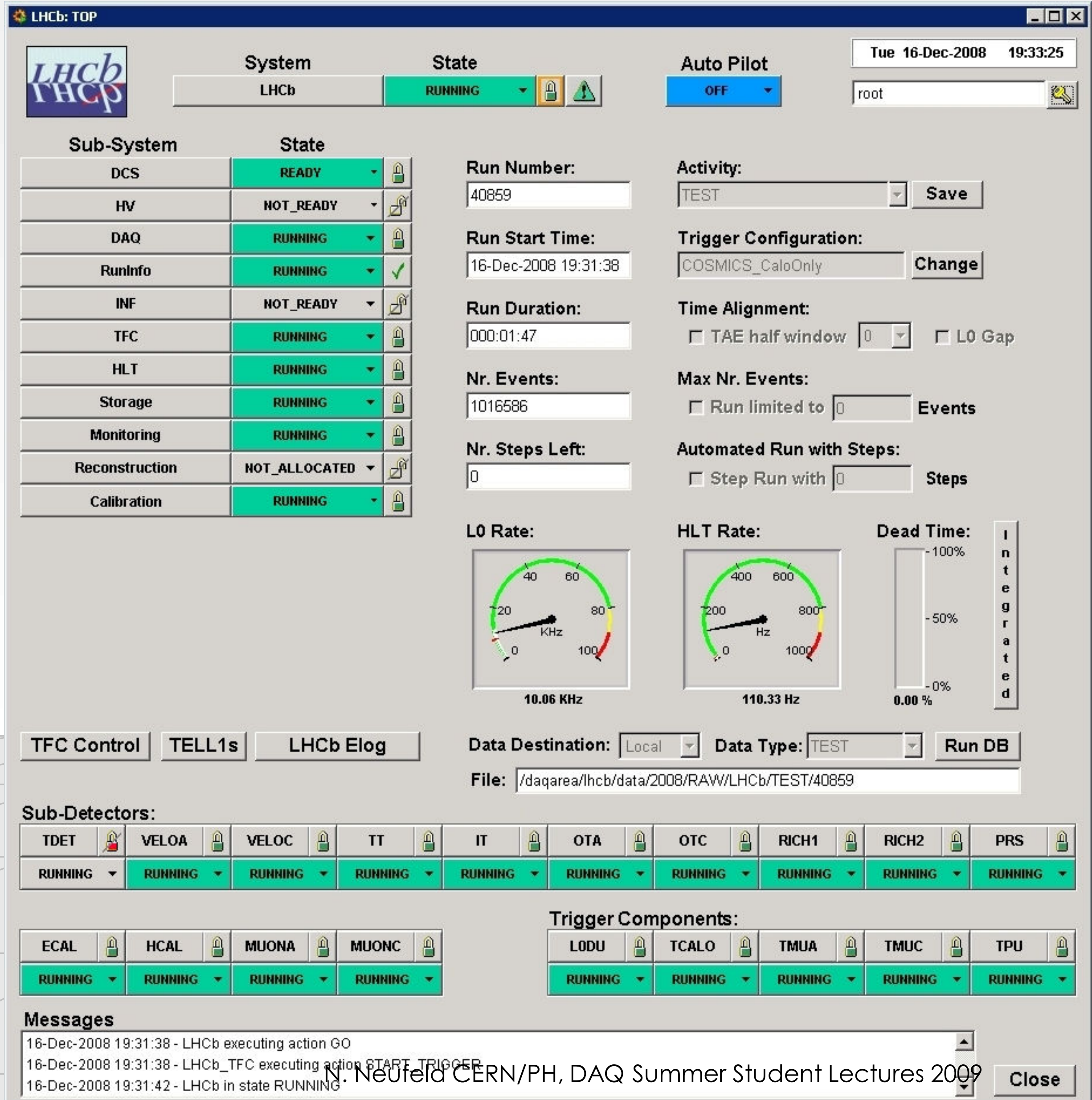
Detector Control

- The detector control system (DCS) (also Slow Control) provides the monitoring and control of the detector equipment and the experiment infrastructure.
- Due to the scale of the current and future experiments is becoming more demanding: for the LHC Experiments: ≈ 100000 parameters

Control hierarchy



Run Control GUI



Main panel of the LHCb run-control (PVSS II)

Gallery

ALICE Storage System

Online Network Infrastructure