Trigger and Data Acquisition
at colliders

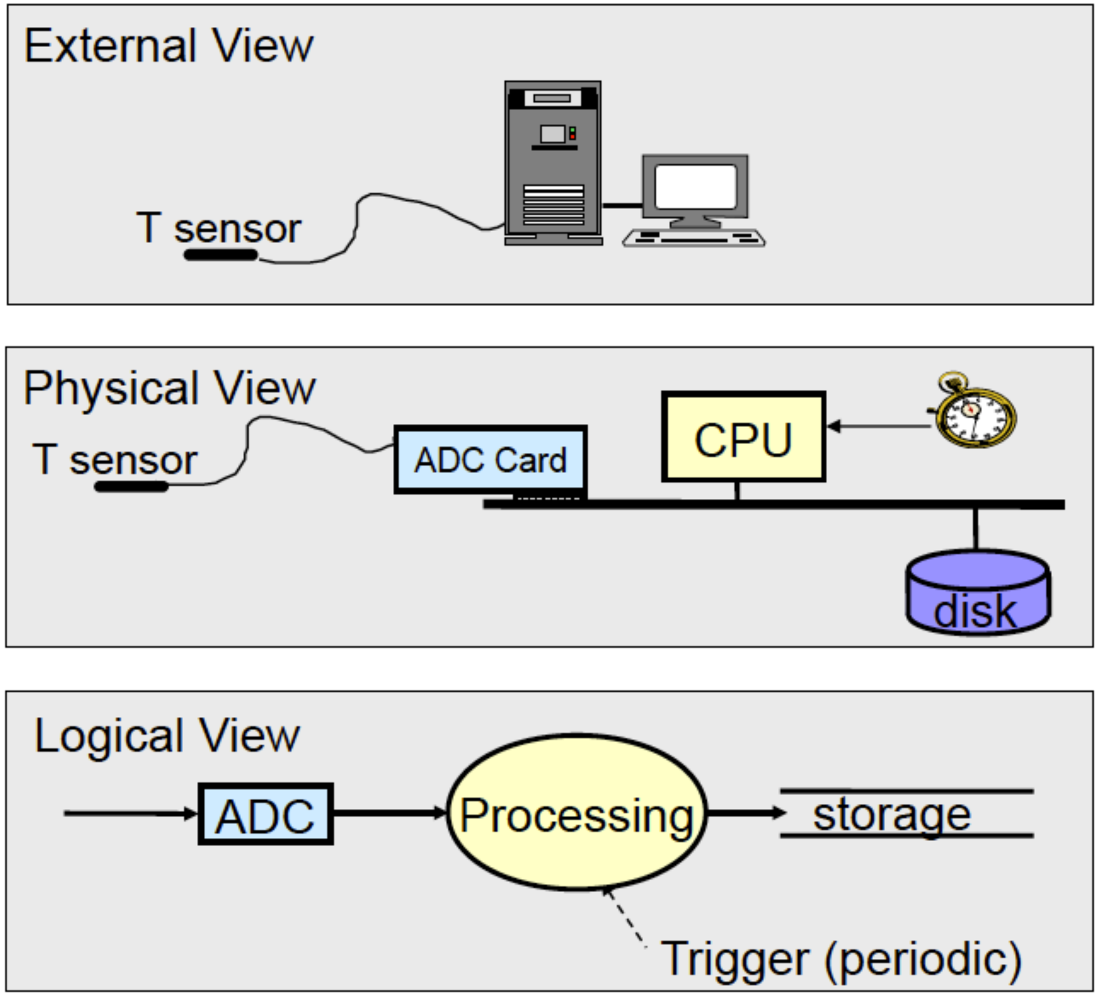
G. De Lentdecker
Université Libre de Bruxelles

DAQ ARCHITECTURE

Break

- Now we will make a break in the data flow:
  - *We have to introduce a couple of concepts about DAQ architecture and trigger before resuming with the data flow*

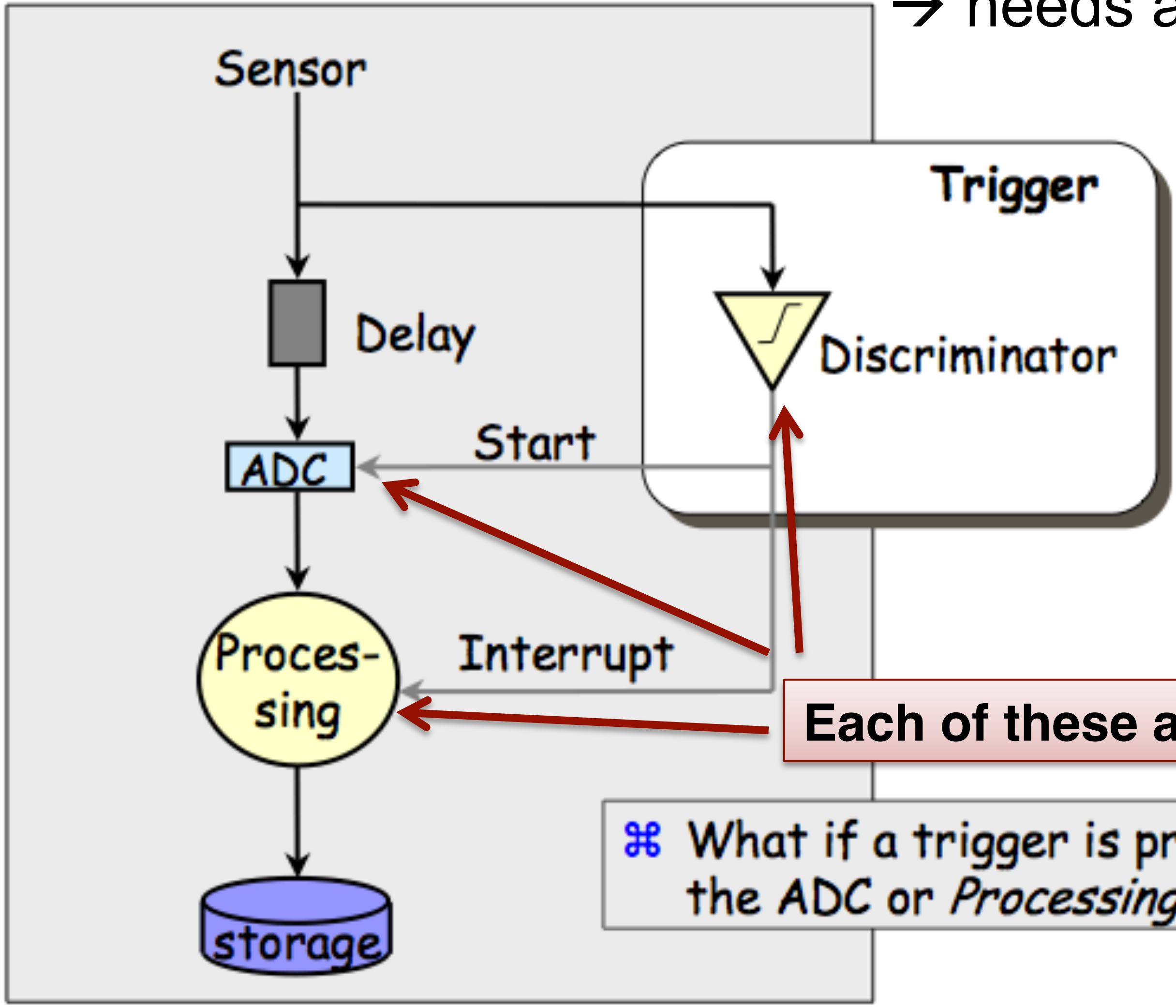
Basic DAQ – periodic trigger



- Measure temperature at a fixed frequency
- The system is clearly limited by the time to process an "event"
- Example: τ = 1ms
  - *ADC conversion, CPU processing, Storage*
- Max. rate = 1/1ms=1kHz

Basic DAQ: real trigger

- A "trigger" is a system that rapidly decides, based on simple criteria, if an interesting event took place and to initiate the data acquisition. Events are asynchronous and unpredictable
  → needs a physics trigger

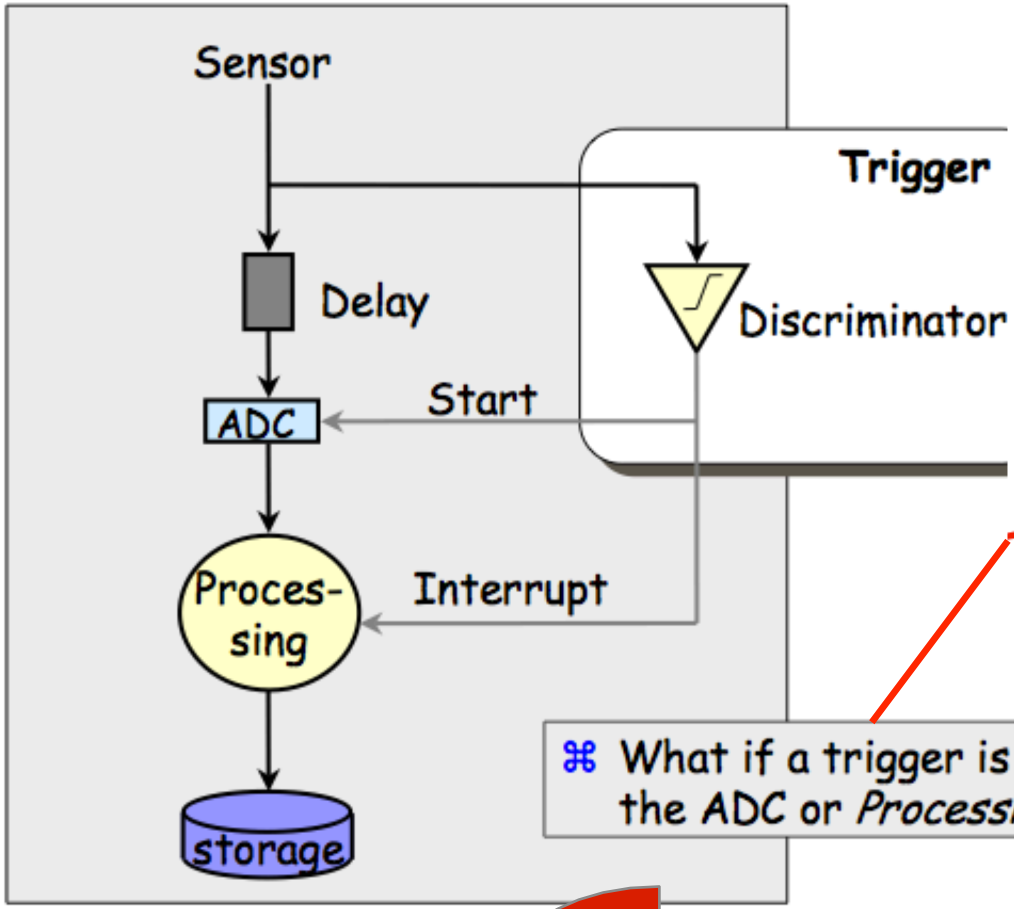


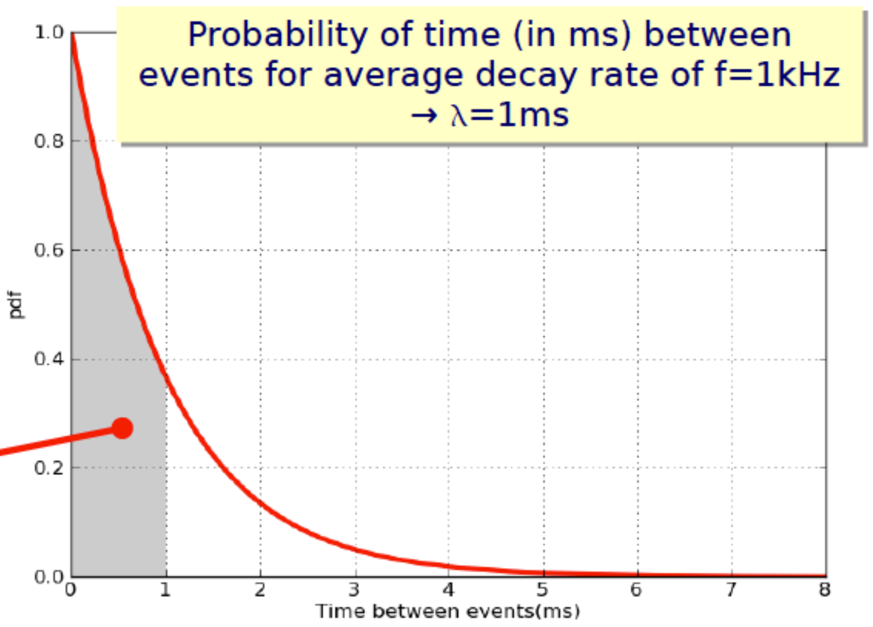
Each of these actions takes some time

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

Basic DAQ: real trigger

- Example: measure β decay properties
  - *Stochastic process*



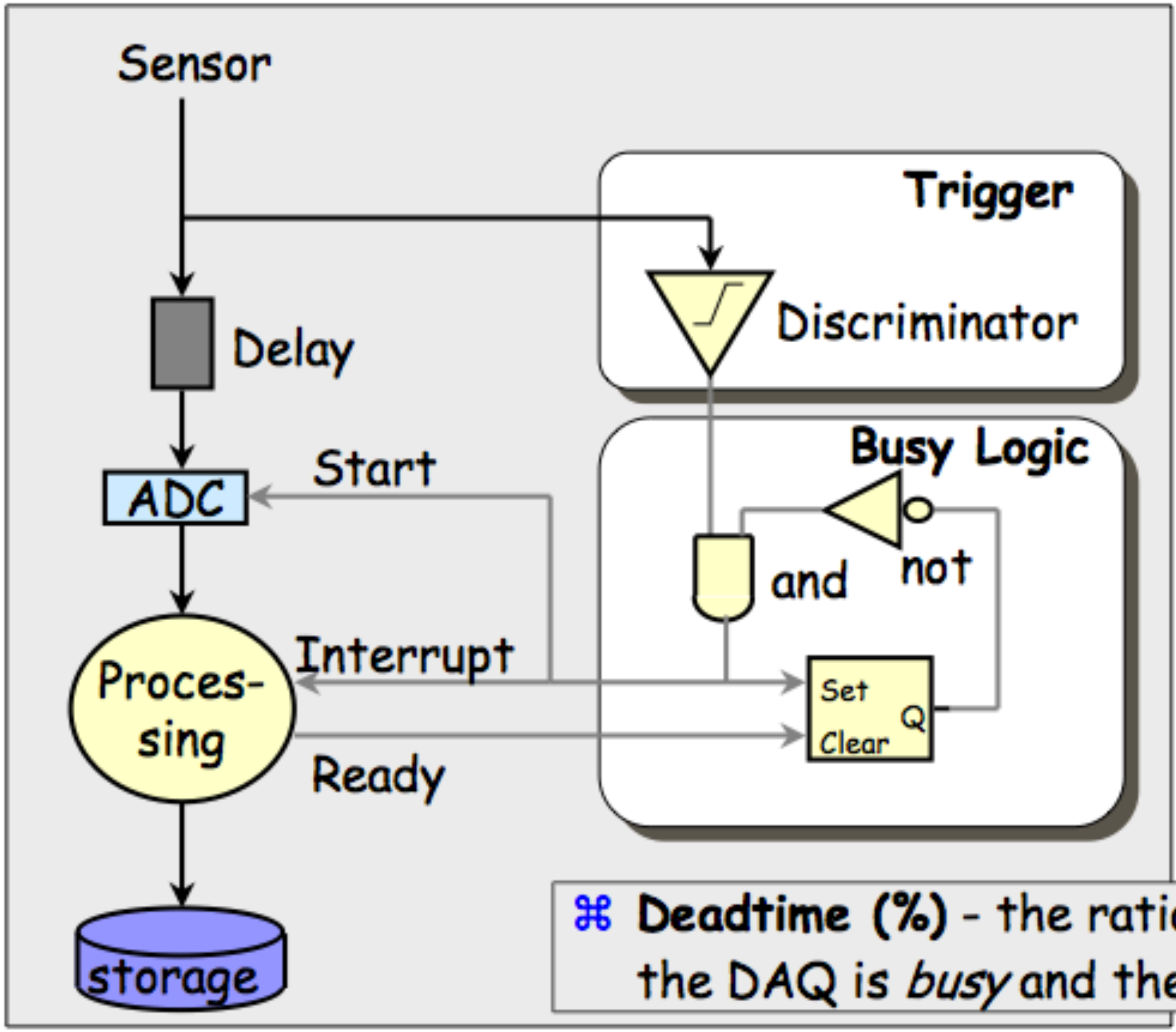


⌘ What if a trigger is produced when the ADC or *Processing* is busy ?

Unpredictable !

Busy logic

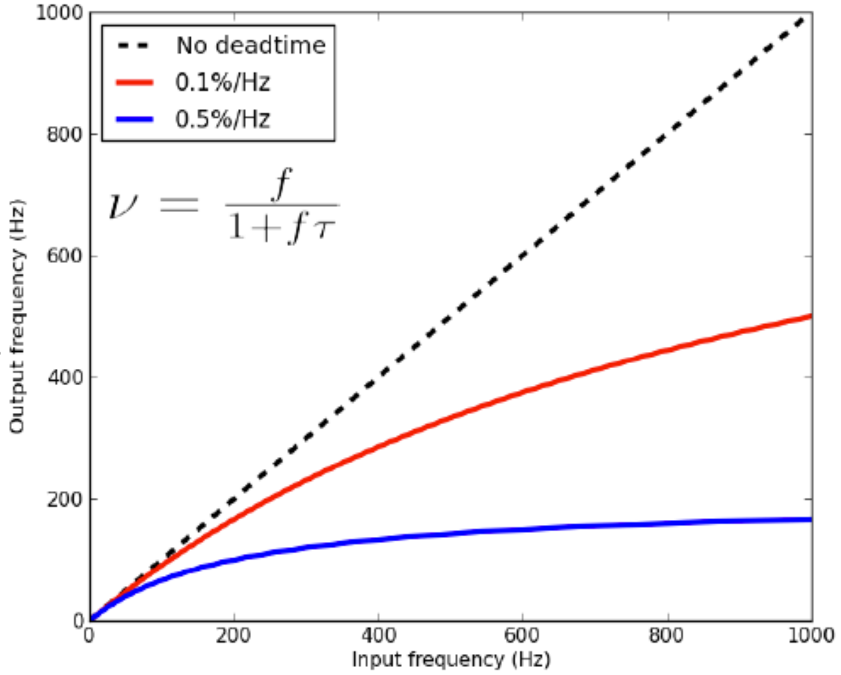
- To avoid this unpredictable state, a busy logic is introduced:



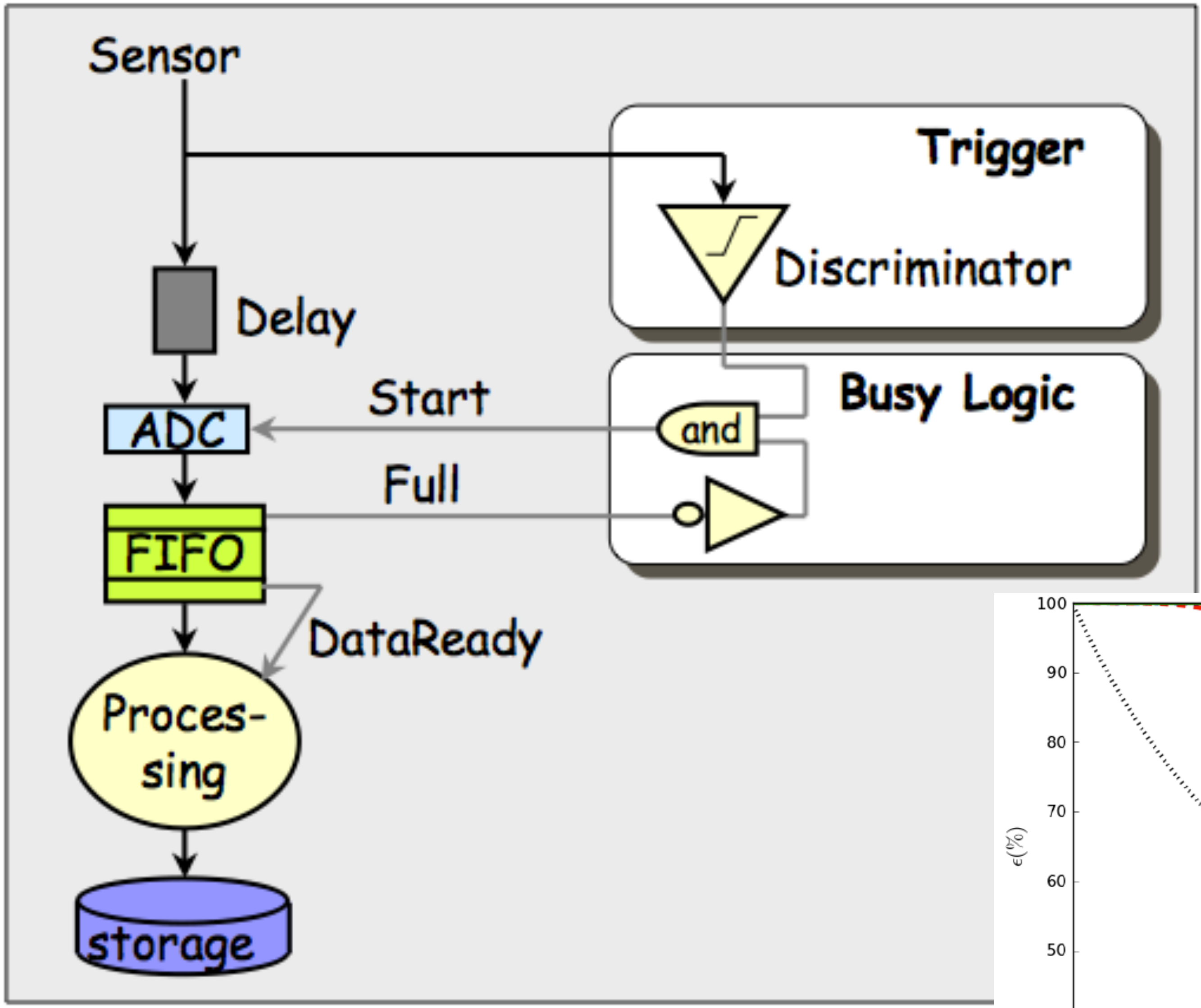
⌘ **Deadtime (%)** - the ratio between the time the DAQ is *busy* and the total time

Dead-time

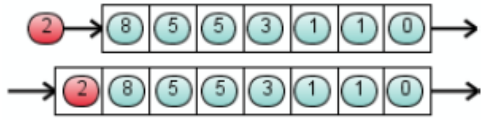
- f = average frequency of occurrence of the physics process
- ν = average frequency at which the system can acquire events
- τ = time to process one event, without being able to handle other triggers
- → $\nu\tau$ = fraction of time the DAQ is busy; $(1-\nu\tau)$ = DAQ is free
- $f \cdot (1-\nu\tau) = \nu$ → $\nu = \frac{f}{1+f\tau} < f$
- Efficiency, $\varepsilon = \frac{N^{saved}}{N^{tot}} = \frac{1}{1+f\tau} < 100\%$
- In our exemple: f = 1kHz, $\tau = 1$ ms
- → ε = 50 %, ν = 500 Hz
- The dead-time, d = 0.1% / Hz



Reducing the deadtime

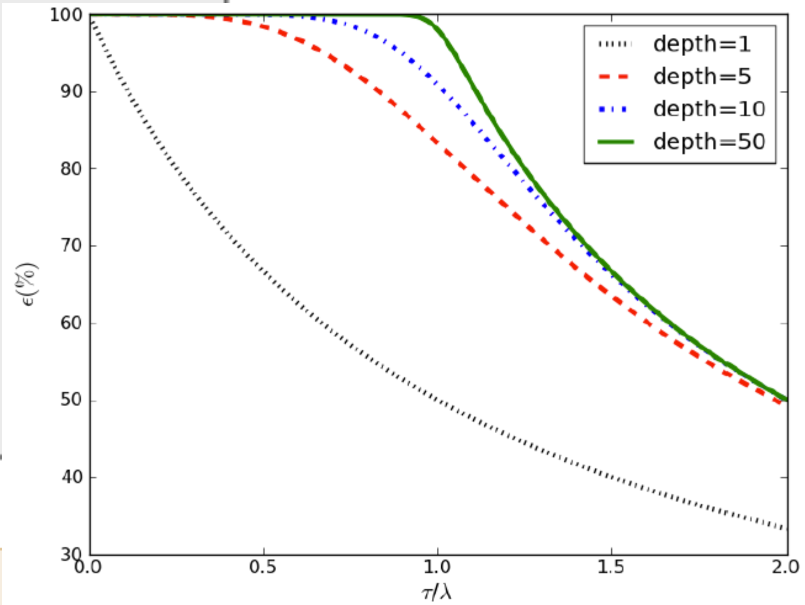


FIFO (First-In First-Out)

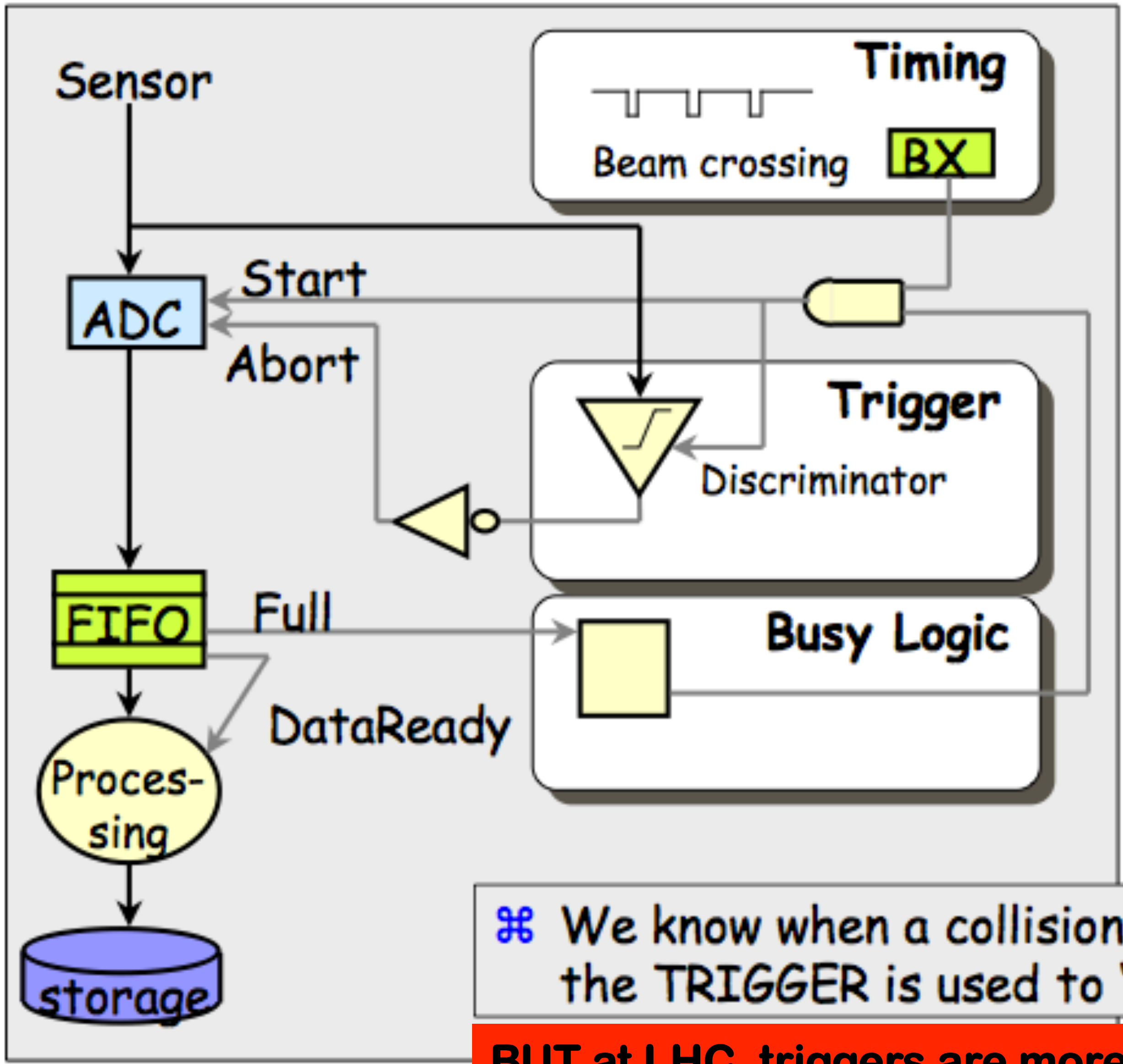


(see slide 6 of part II)

FIFO absorbs and smooths the input fluctuations, providing a ~ steady (de-randomized) output rate



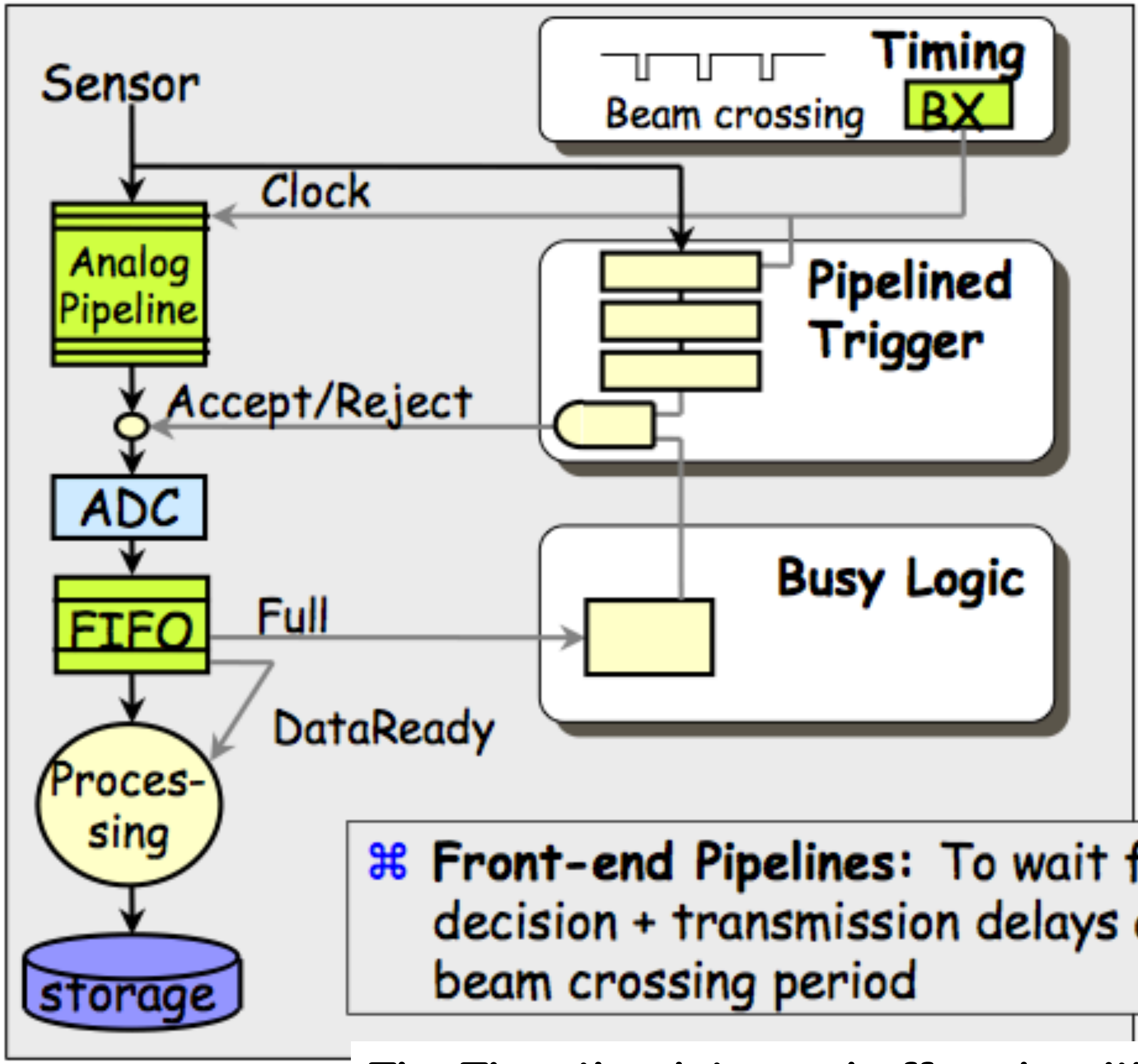
Simple DAQ in collider mode



⌘ We know when a collision happens
the TRIGGER is used to "reject" the data

BUT at LHC, triggers are more sophisticated than just a discriminator; and typically takes ~a few us while LHC bunch crossing is 25 ns ?!?

Simple DAQ at LHC



⌘ **Front-end Pipelines:** To wait for the trigger decision + transmission delays are longer than beam crossing period

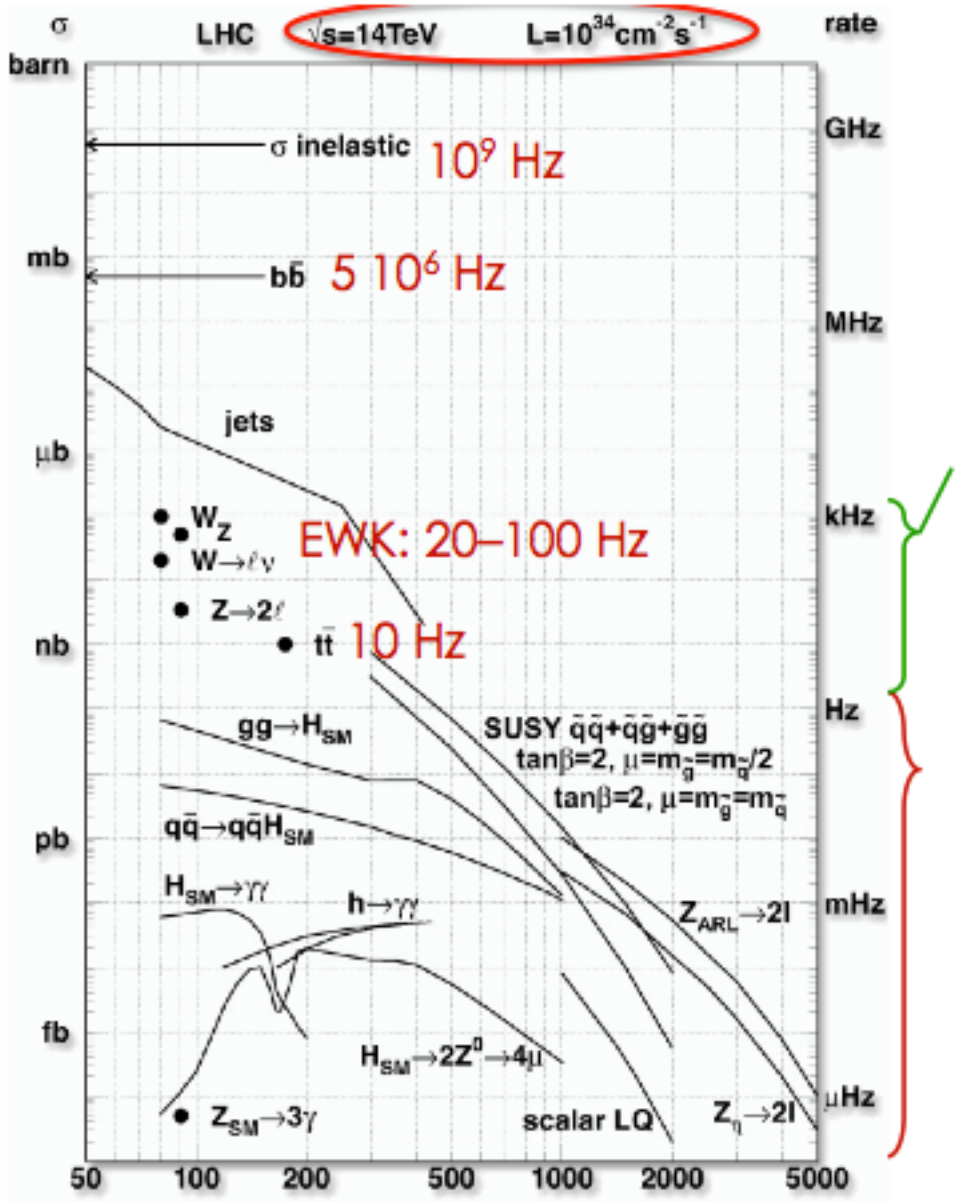
The Time the data are buffered waiting for the trigger is called trigger latency

TRIGGER

Why a trigger ?

- Today a typical PC runs at several GHz, so couldn't we just use the LHC Clock (40 MHz) to trigger the DAQ and use a large number of (relatively) cheap PC for data processing ?
- The answer is no, of course:
  - *No (affordable) DAQ system could read out O(10^7) channels at 40 MHz → 600 TBit/s to read out – even assuming binary channels!*
  - *In addition most of these millions of events per second are totally uninteresting: one Higgs event every O(100) seconds*
- → we need event selection, that is a "trigger"

- Hence new questions arise:
  - *What trigger rate can we afford ?*
  - *How do we select the data ?*
  - *How do we make sure that the values from the many different channels refer to the same original event (collision)?*



- Total collision rate ~ 1 GHz @ L = 10^{34}cm^{-2}s^{-1} and 14 TeV

- “Interesting” physics is about 6-8 orders of magnitude smaller

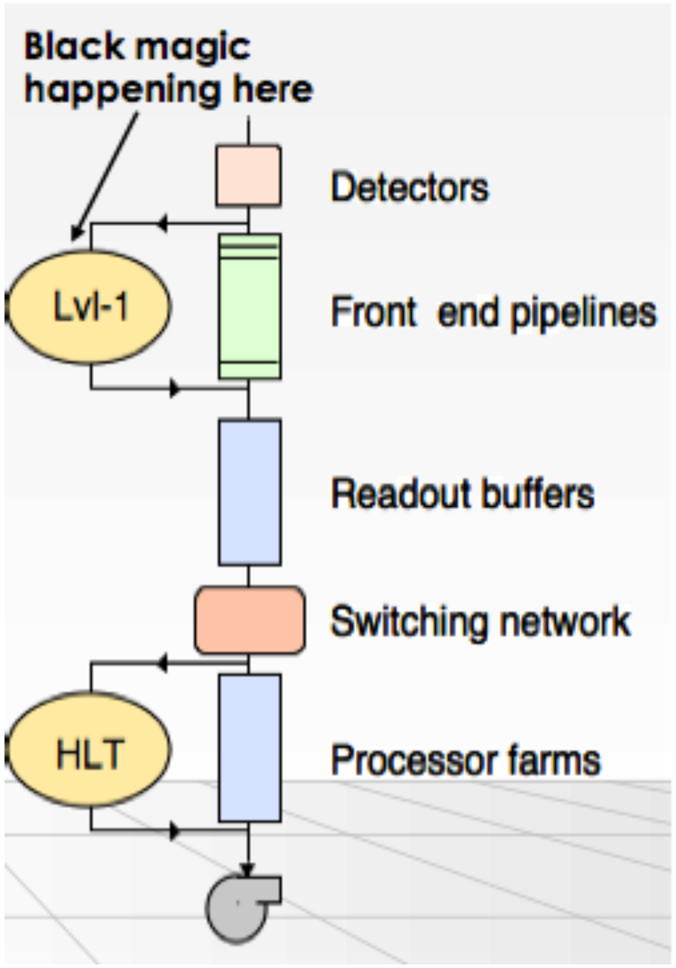
- New physics is 9 orders of magnitude smaller

- We “just” need to identify and select these rare processes from the overwhelming background before reading out and storing the whole event.

Trigger

- Usually the reduction of the data rate to a reasonable rate of O(100 Hz) that can be archived cannot be achieved in one step
- Therefore the trigger is split in several levels (usually called level-1, 2, 3,…). Each level performs a reconstruction of the data before applying a selection.
- Each trigger level reduces the data rate → subsequent trigger levels have more time to perform the data processing.
- Usually the first level (Lvl-1) is hardwired
  - *Short latency: a few μs.*
  - *In HEP Lvl-1 usually uses data from calorimeters and muon detectors*
- The second level (Lvl-2)
  - *Processor based (standard CPU's or dedicated custom/FPGA processing)*
  - *Latency : a few ms.*
- The higher levels are usually software.
  - *The reconstruction program is run in parallel on a large number of CPUs O(100-1000).*
  - *Having more time (~1s) the algorithms can be more complex and can use all the detector data, including the tracker data.*

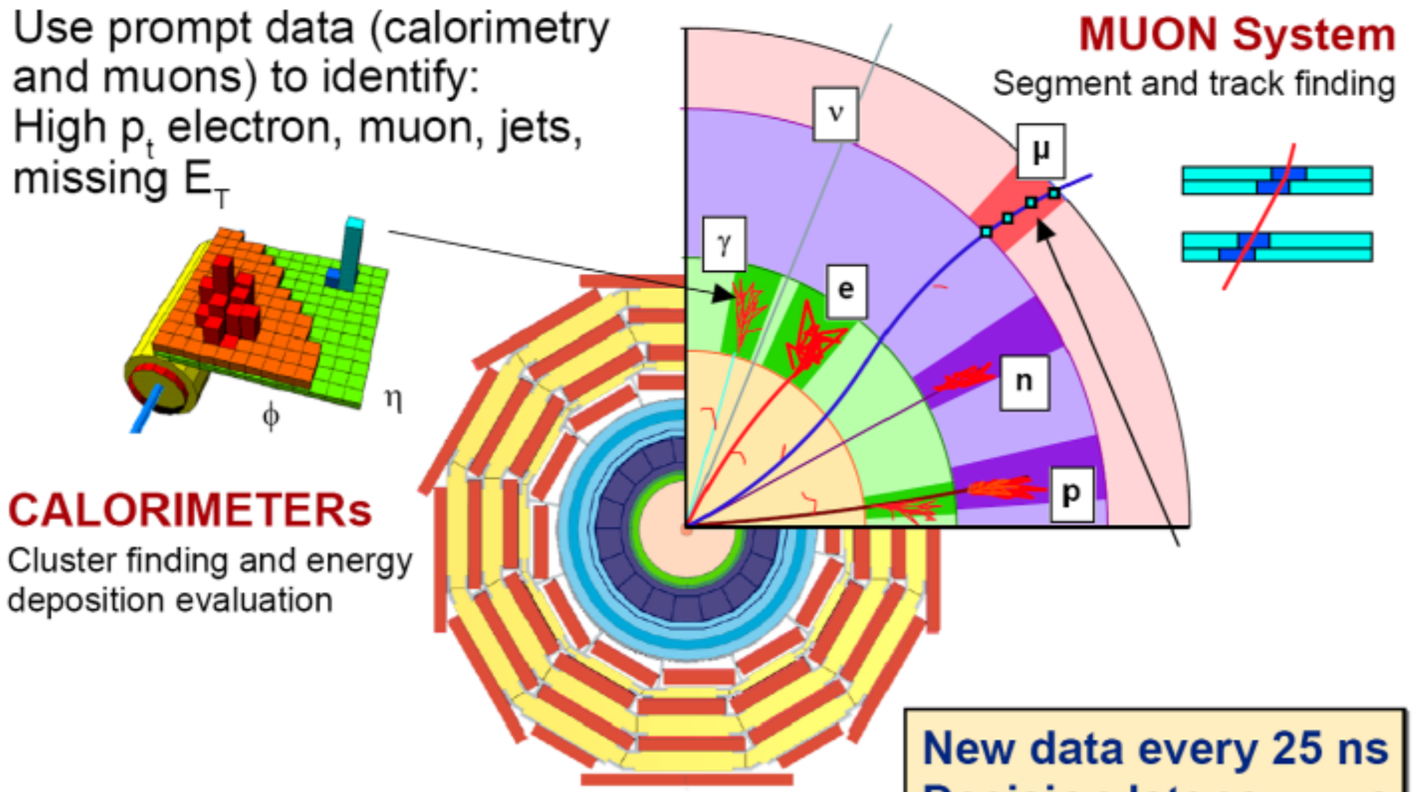
Multilevel trigger



- Level1 :
  - *latency ~ 3μs*
  - *O(10^7) channels*
- Work with local information to avoid many interconnections
- Must be fast and robust
  - *Look for simple signatures*
  - *implemented in hardware*
  - *Use Calo and muon data*

Fortunately Mother Nature is kind

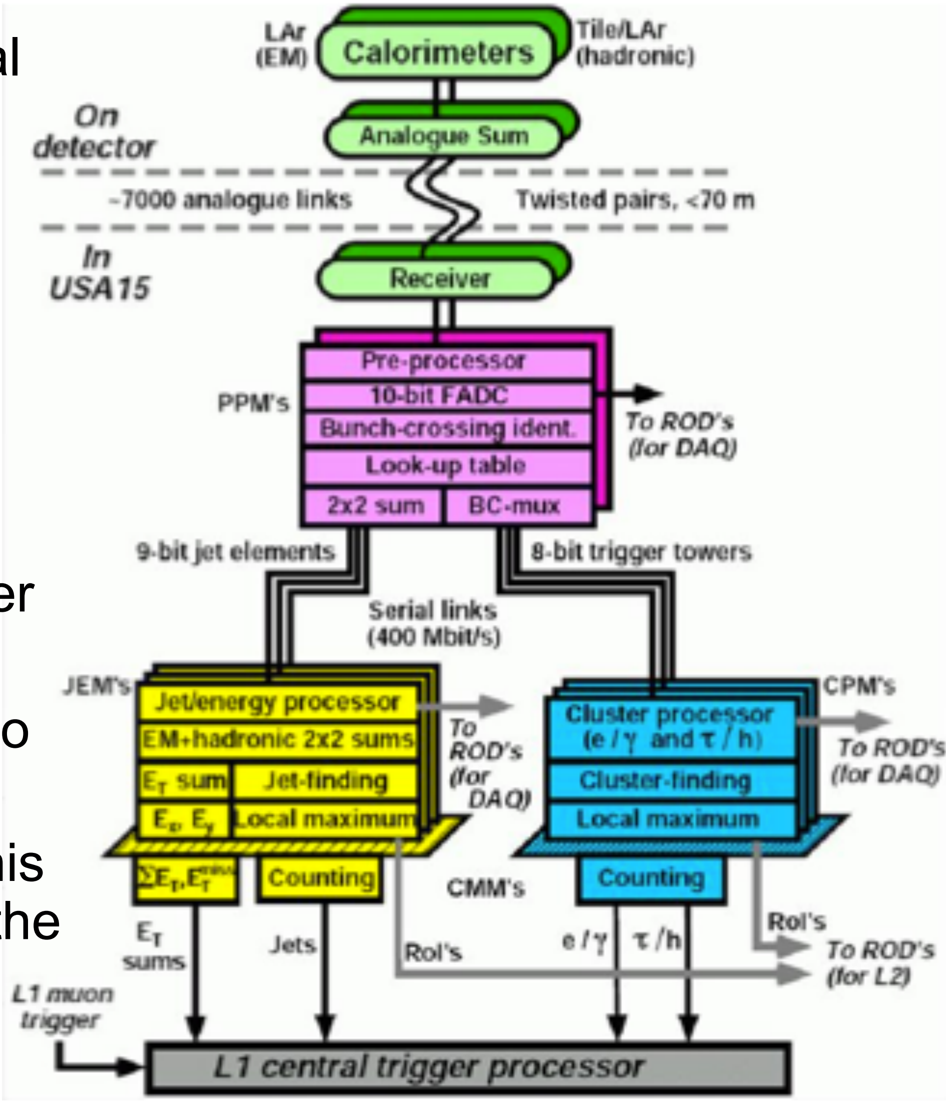
- p-p collisions mostly produce low p_t particles ~1 GeV/c
- Physicists are interested in heavy particles (W, Z, Top, Higgs, Z', W', sparticles,…) often decaying in high pt leptons or jets (p_t > 25 GeV/c)
- Therefore Level1 triggers will often look for such high p_t leptons and jets (as well as large MET):


Use prompt data (calorimetry and muons) to identify:
High p_t electron, muon, jets, missing E_T
MUON System
Segment and track finding
ν
μ
γ
e
n
p
η
ϕ
CALORIMETERs
Cluster finding and energy deposition evaluation

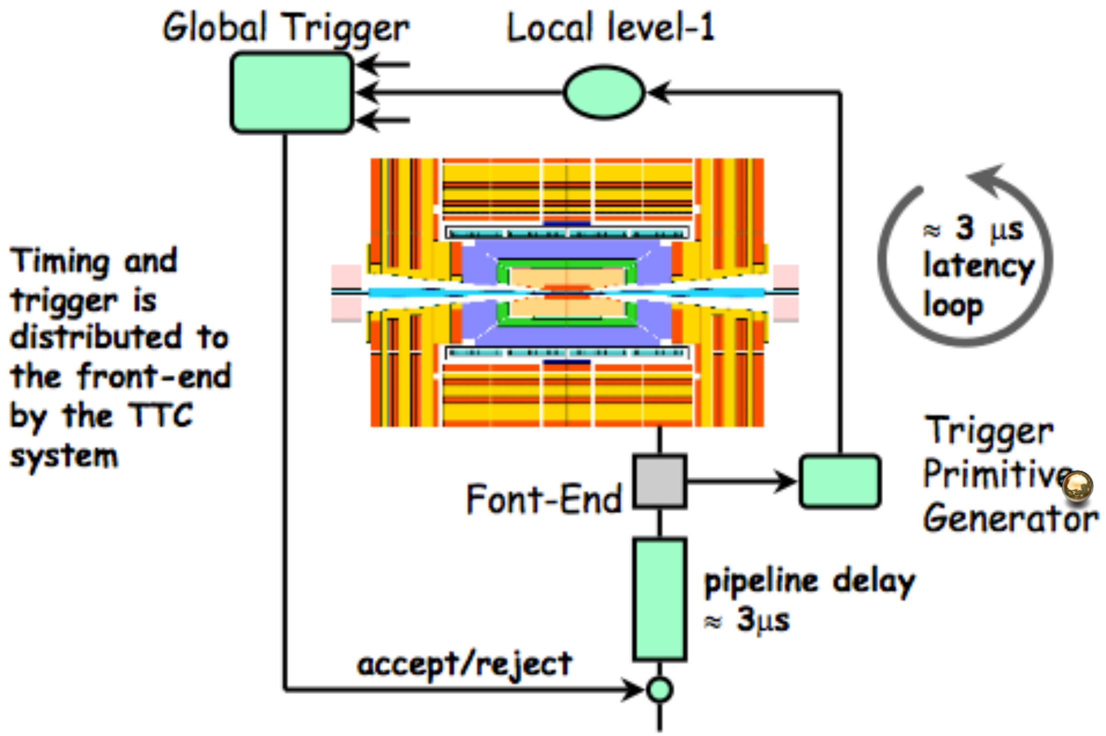
New data every 25 ns
Decision latency ~ μs

ATLAS Level1 Calo Trigger

- 7000 trigger towers (EM+HA) 0.1x0.1 in eta, phi space
- On FEE, analogue signals of individual cells are summed to form tower signal
- Signal transmitted to Pre-processor (PPM): digitization and E_t estimate at 40 MHz
- Tower data are transmitted to cluster processor (CPM); there are 4 crates
- e/γ processing on large FPGA
- e/γ candidates sent to L1 central trigger processor 1.5 μs after bunch crossing
- Note the remaining L1 latency is due to signal transmission !
- After Level1 has made his decision, this information has to be sent back to all the FEE



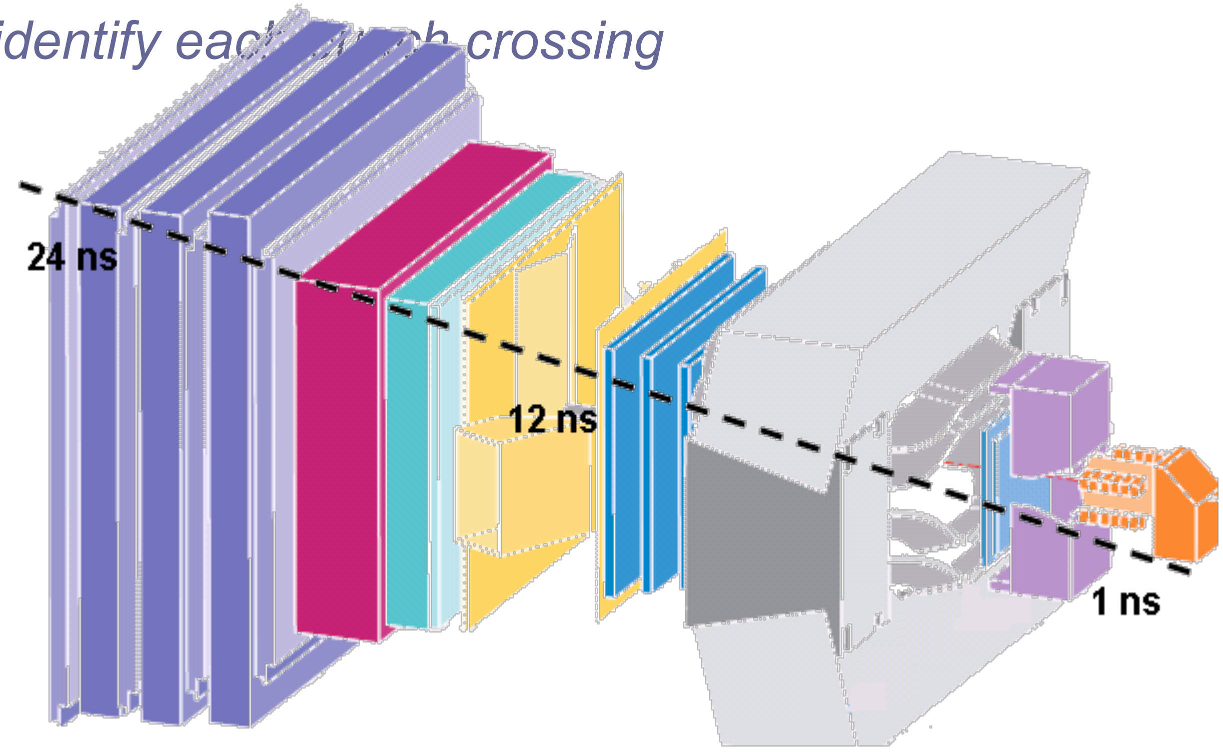
Level1 Trigger distribution


Global Trigger
Local level-1
Timing and trigger is distributed to the front-end by the TTC system
≈ 3 µs latency loop
Trigger Primitive Generator
Font-End
pipeline delay ≈ 3µs
accept/reject

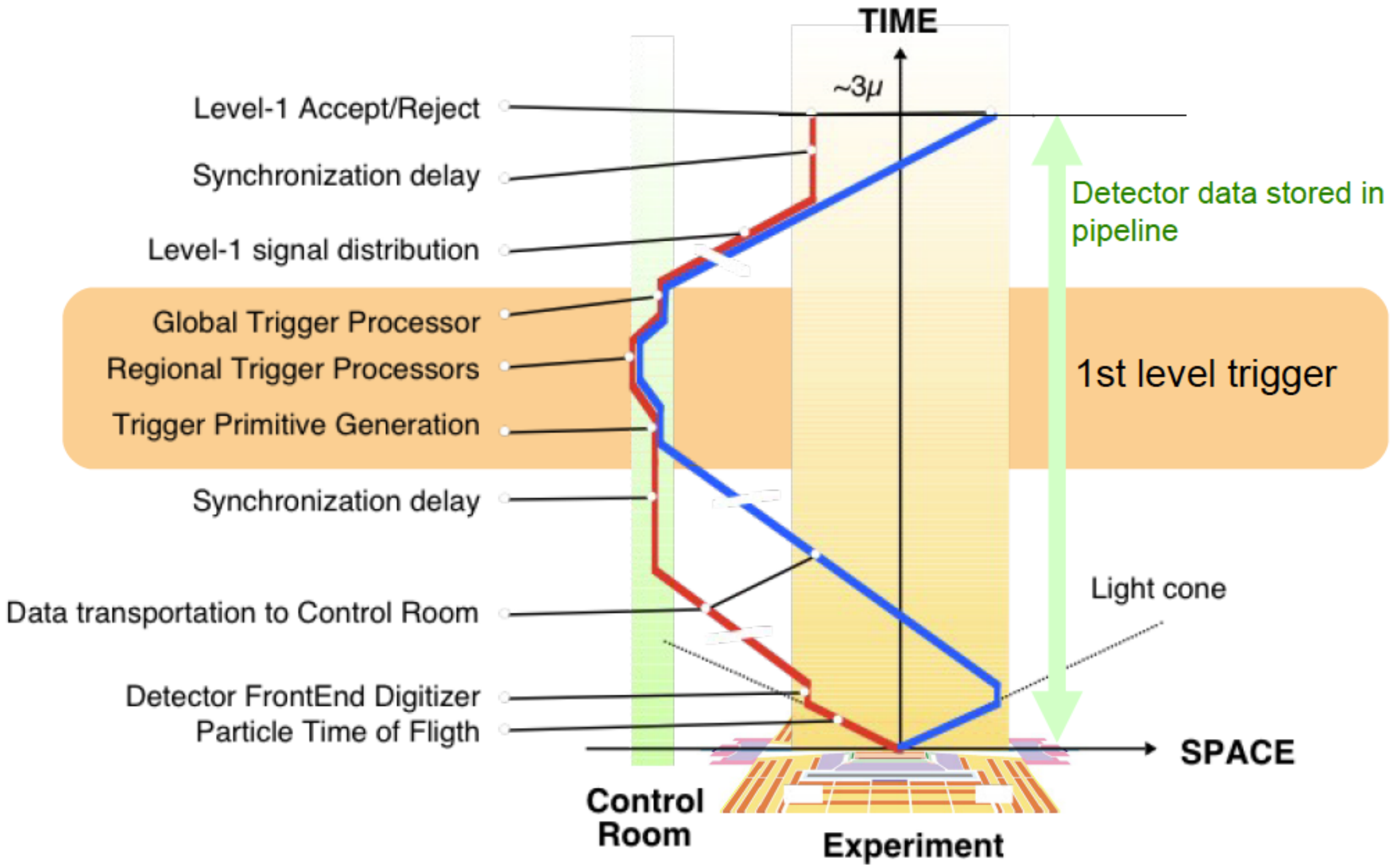
- Now that the magic box tells for each Bunch Crossing if the event has to be kept or rejected, this information has to be sent back to the FEE so that they can send or discard their data.
- Synchronisation at the output of the pipeline is not trivial => needs timing calibration

Level1 Timing Issues

- Need to synchronize all detectors components to better than 25ns
  - *The system need a reference clock*
- But for some detectors, signal development/time of flight > 25 ns
  - *Integrates more than one bunch crossing information*
  - *Particles travel 7.5 m in 25 ns*
  - *Need to identify each bunch crossing*



Signal path during L1 Trigger


TIME
~3μ
Level-1 Accept/Reject
Synchronization delay
Level-1 signal distribution
Global Trigger Processor
Regional Trigger Processors
Trigger Primitive Generation
Synchronization delay
Data transportation to Control Room
Detector FrontEnd Digitizer
Particle Time of Fligth
Detector data stored in pipeline
1st level trigger
Light cone
SPACE
Control Room
Experiment

LHC Clock

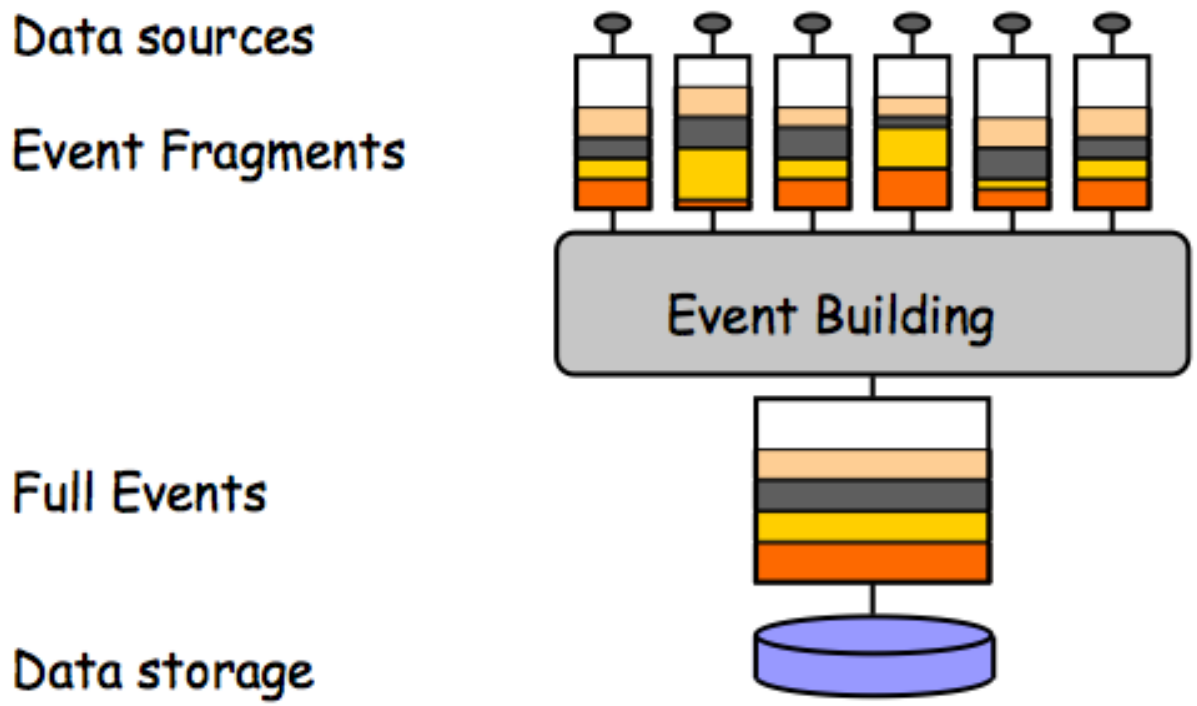
- At LHC, there is common system for all LHC experiments:
  - *The Timing, Trigger and Control (TTC)*
- TTC is based on the LHC Clock
  - *The clock is extracted by monitoring the passage of the proton bunches*
- It is sent to millions of components in ALICE, ATLAS, CMS & LHCb (by optical fibers)
- TTC also sends the trigger signal to the FEE
- Before starting to data taking, all the components had to be aligned in time
- System has to be robust :
  - *A loss of synchronization may be very difficult to be detected…*

Where are we so far ?

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

  → Need and Event Builder (EVB)
- But don't forget:
  - *At LHC the Level1 will typically accept 1/10000 events*
    - Output L1 rate is still very high ~100kHz
  - *Amount of data is still very large:*
    - Typical event size ~1Mbyte (after zero suppression or data compression)
- → need a bandwidth of 100 Gbytes/s (=800 Gbits/s) to extract the data from the Front End.

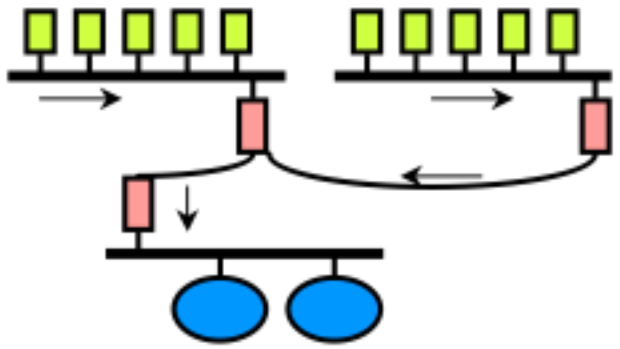
Event Builder Basic's

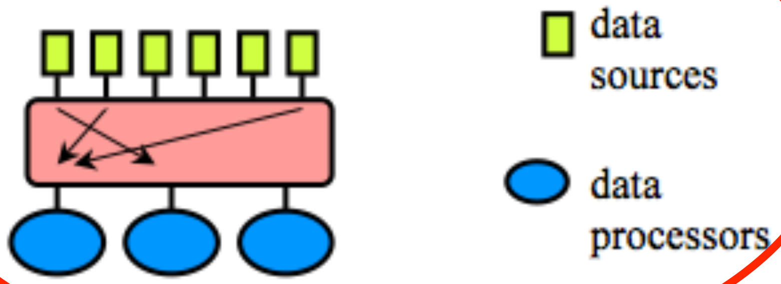


- But which technology to use?

Network-based EVB is the choice of all LHC experiments

Bus *OR* *switch/network ??*

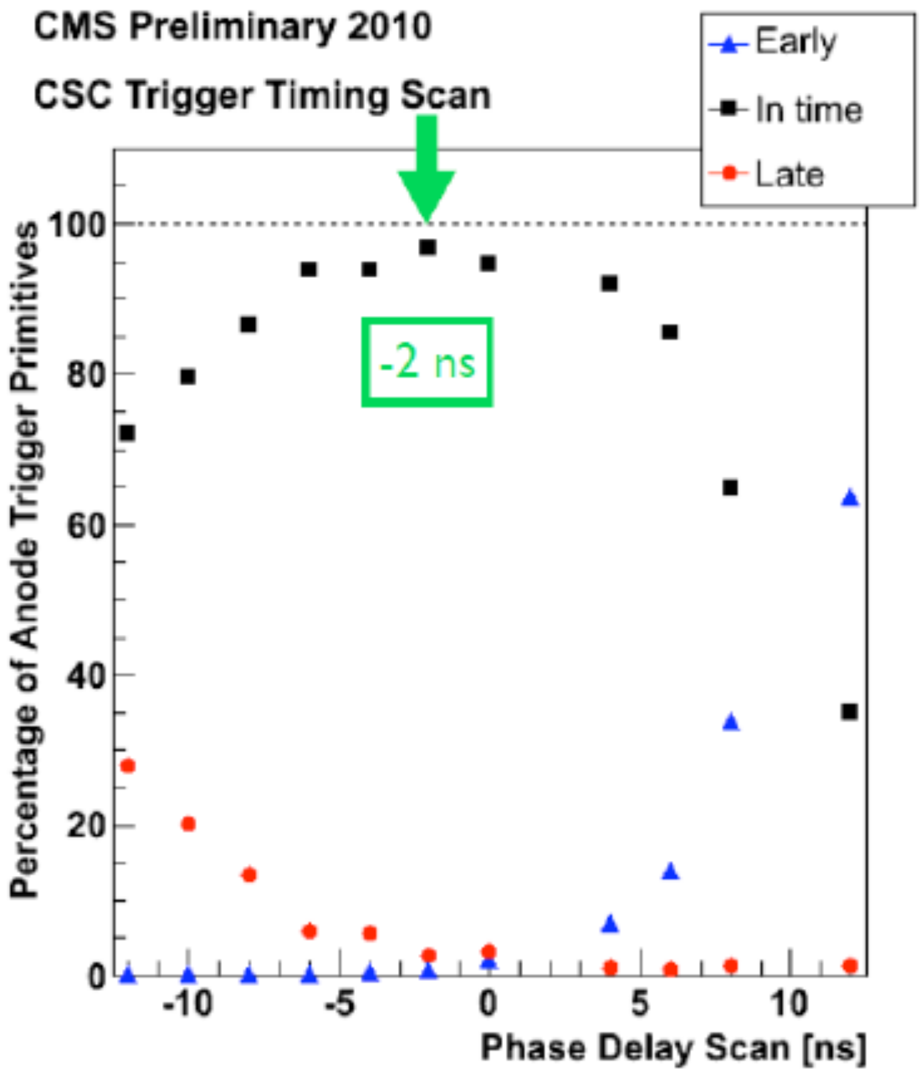




BACK-UP

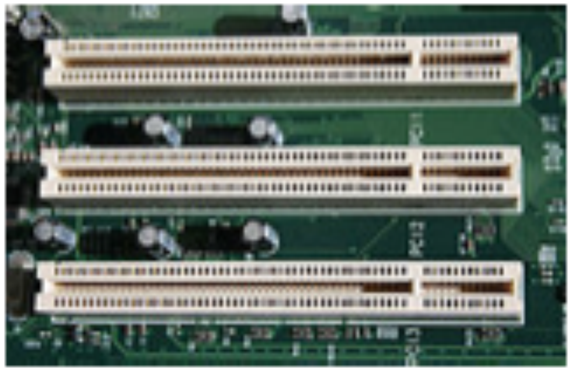
Timing synchronization

- Example : CMS CSC chambers
  - *Perform Timing scan:*
    - Scan a range of timing delays
    - Find best alignment:
      – highest % of Anode Trigger Primitives

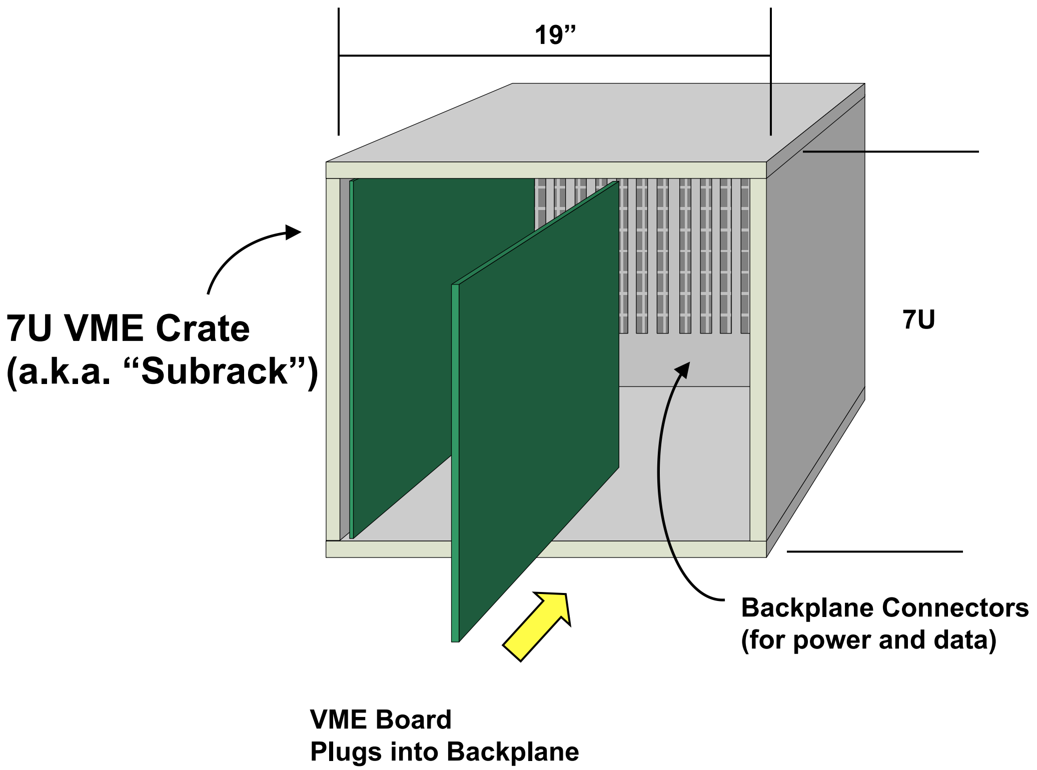


Buses

- Let’s do a parenthesis here on common buses used in HEP, because even old ones are still very popular in our labs…
- What is a bus ?
  - *A bus is a common electrical connection between several electrical devices*
  - *A bus can either allow signals to be transferred between devices, the summing (mixing) of output signals from the devices or the distribution of input signals or power amongst the devices.*
  - *A bus often takes the form of a wire or printed circuit conductor that terminate at multiple connectors which allows devices to be plugged into the bus*
  - *A common example is the PCI bus in PCs:*



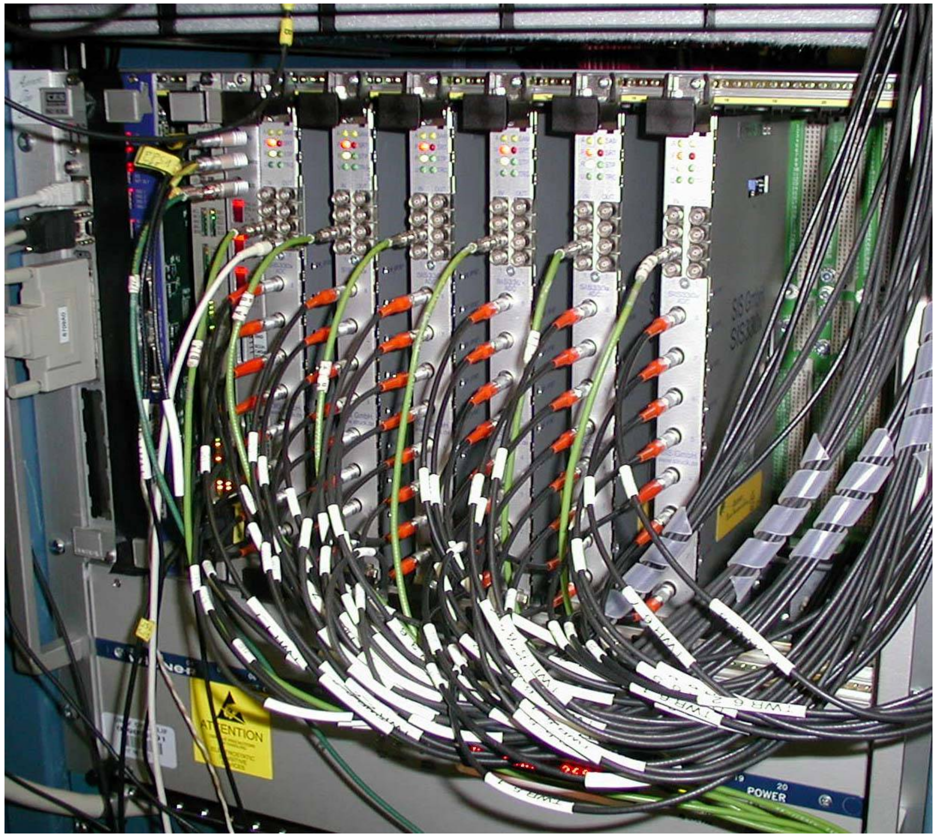
Buses in HEP (example VME)



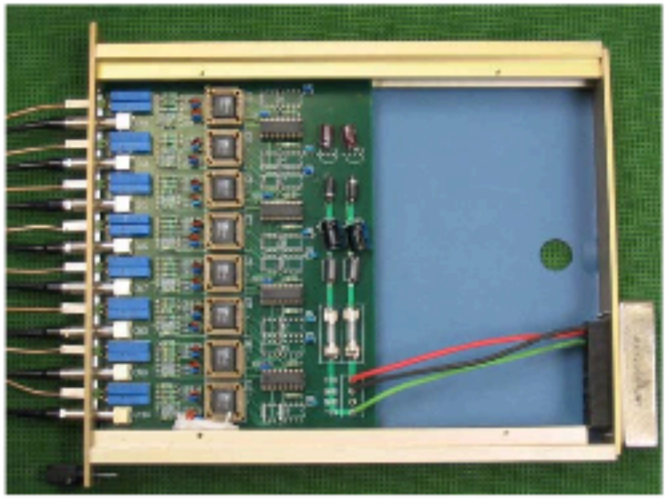
- Buy or build readout boards with some inputs
- Put many of these multi-port modules together in a common chassis or crate
- The modules need
  - *Mechanical support*
  - *Power*
  - *A standardized way to access their data (our measurement values)*
- All this is provided by standards for (readout) electronics such as VME (IEEE 1014)

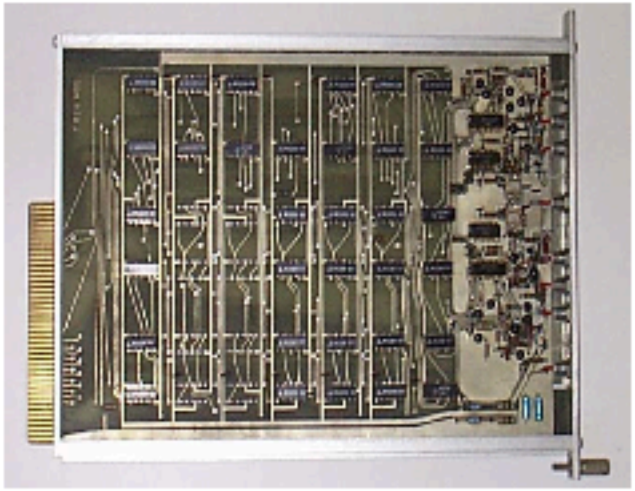
VME Example:

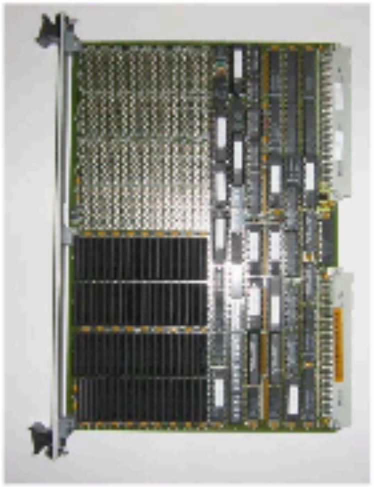
- The standard (here VME) defines:
  - *mechanical standard*
  - *electrical standard for power on the backplane*
  - *signal and protocol standard for communication on the bus*



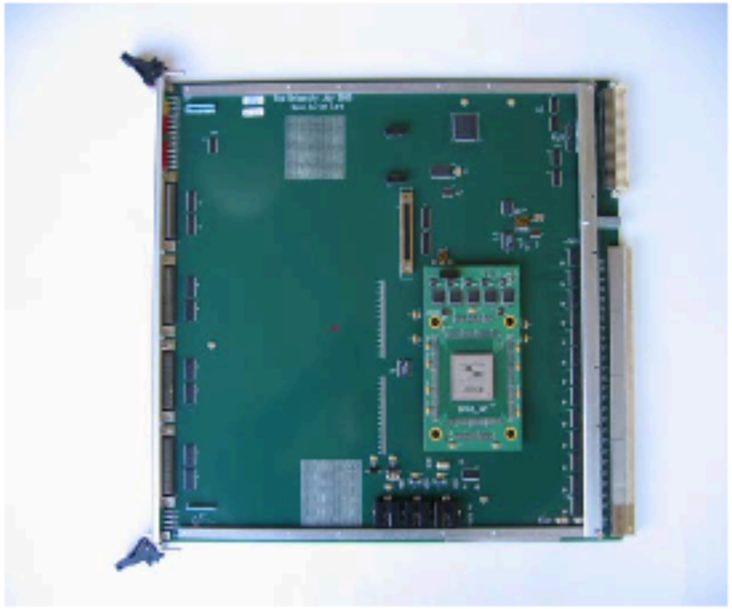
Some famous buses in HEP


NIM, 1964


CAMAC, 1969


VME 6U, 1981

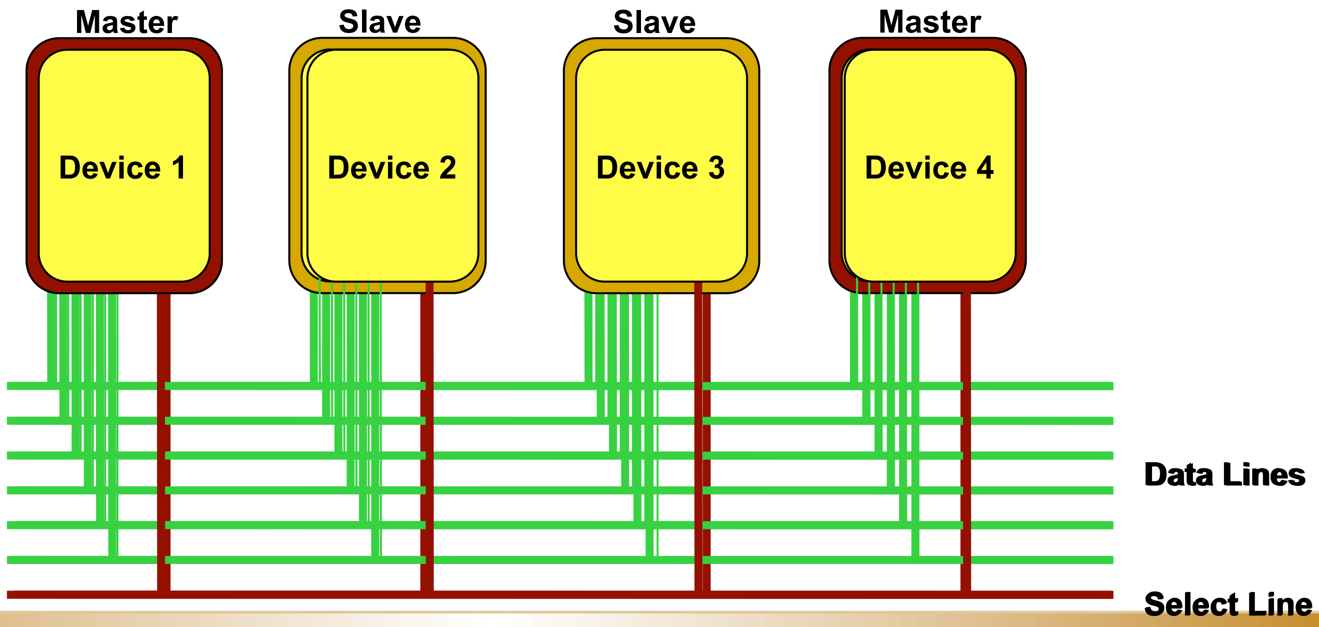
FASTBUS, 1986


VME 9U, 1994

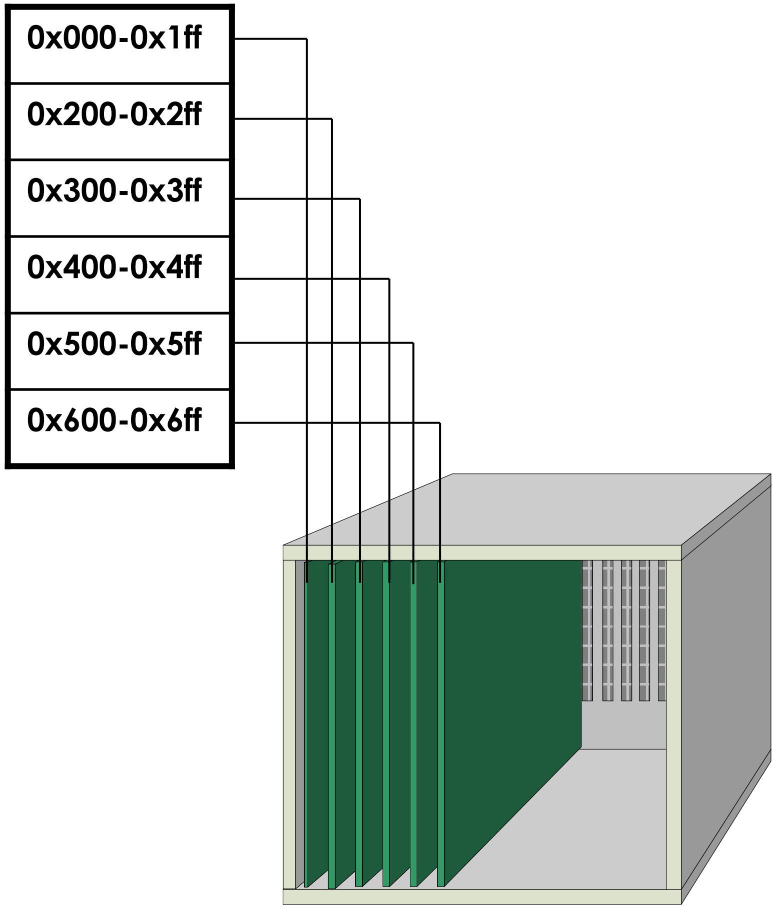
CompactPCI, 1995

Bus communication in a VME crate

- A bus connects two or more devices and allows them 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



The VME bus

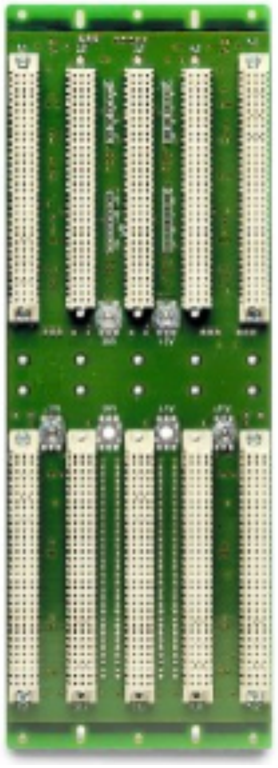


- In a VME crate we can find three main types of modules
  - *The controller which monitors and arbitrates the bus*
  - *Masters read data from and write data to slaves*
  - *Slaves send data to and receive data from masters*
- Addressing of modules
  - *In VME each module occupies a part of a (flat)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

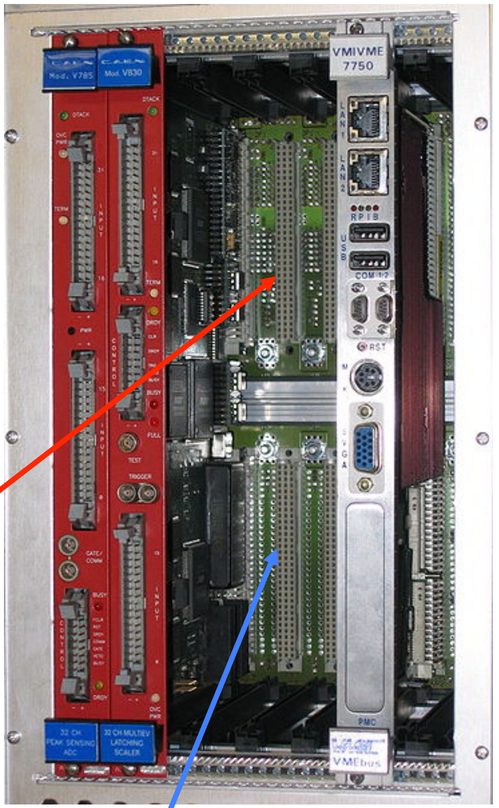
VME specifications

Mandatory bus

Extension bus

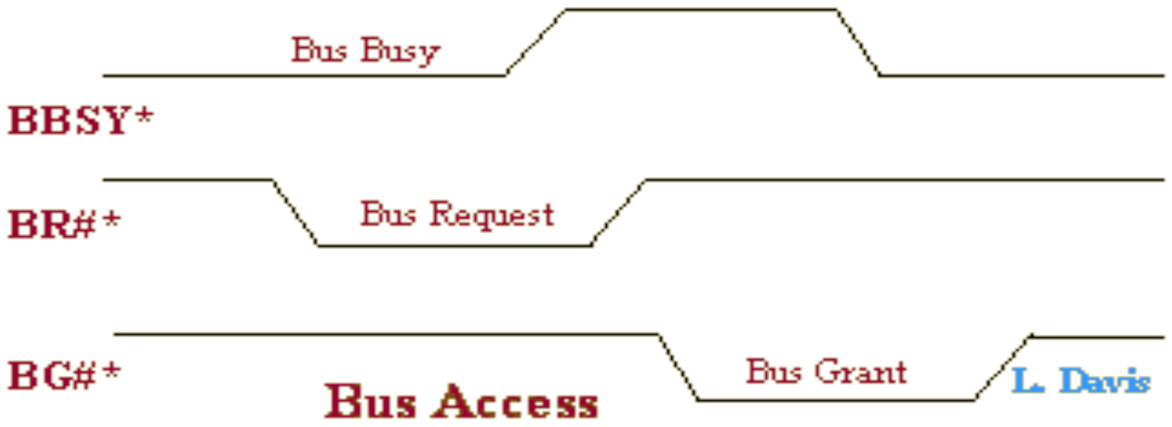


- <15:0> 16-bit Data Bus
- <23:0> 24-bit Address Bus
- 9 lines for Arbitration Bus
- Clock, Control, Status signals
- Power: +5V, +12 V, -12V
- GND: 5 lines



- <31:16> 16-bit Data Bus Extension
- <31:24> 8-bit Address Bus Extension
- Additional power lines and GND lines

VME Bus at Work

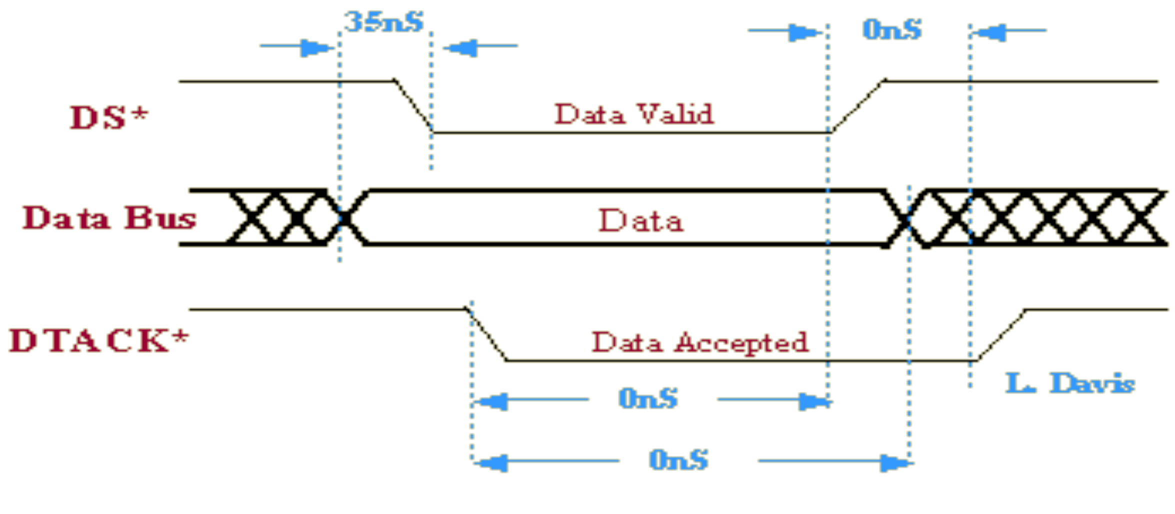


- Arbitration: Master asserts(*) BR# (Bus Request), Controller answers by asserting BG# (Bus Grant)
- If there are several masters requesting at the same time the one physically closest to the controller wins
- The winning master drives BBSY* high to indicate that the bus is now in use

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

VME data transfer

- The Master writes data and address to the data / address bus respectively
- It asserts the lines DS* (DATA Strobe) and AS* (ADDRESS Strobe –not shown) to signal that the data and address are valid
- The slave reads and acknowledges by asserting DTACK* (DATA Transfer Acknowledge)
- 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



Data Transfer

Speed

- Theoretically ~ 16 MB/s can be achieved
  - *assuming the databus to be full 32-bit wide*
- Better performance by using block-transfer
  - *After an address cycle several (up to 256) data cycles are performed*
  - *Performance goes up to 40 MB/s*

- Since 1994 VME64 (64bits)
  - *use the address bus also for data*
  - *Twice the bandwidth of VME (up to 80 MB/s)*
- More recently VME64x
  - *New 160-pin 5-row connector*
  - *Twice the bandwidth of VME64 (up to 160 MB/s)*

Disadvantages of parallel buses

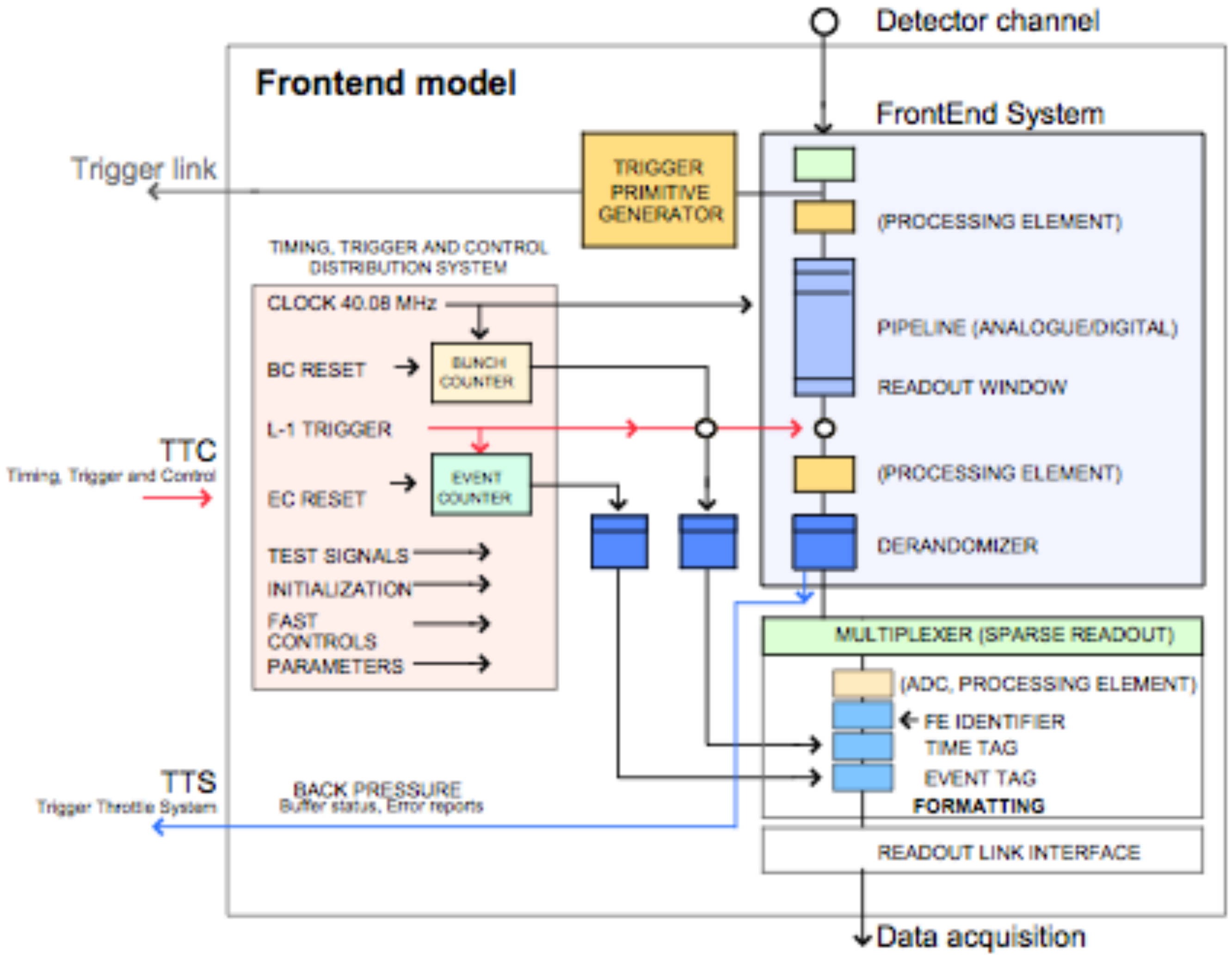
- A bus is shared between all devices (each new active device slows everybody down)
  - *Bus-width can only be increased up to a certain point (128 bit for PC-system bus)*
  - *Bus-frequency (number of elementary operations per second) can be increased, but decreases the physical bus-length*
- Nbr of 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

Network based DAQ

- Network technology solves the scalability issues of buses
  - *In a network all devices are equal ("peers")*
  - *In a network all 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*

LHC Front-End examples:

- CMS FE model:



Detector channel
Frontend model
FrontEnd System
Trigger link
TRIGGER PRIMITIVE GENERATOR
(PROCESSING ELEMENT)
TIMING, TRIGGER AND CONTROL DISTRIBUTION SYSTEM
CLOCK 40.08 MHz
PIPELINE (ANALOGUE/DIGITAL)
BC RESET
BUNCH COUNTER
READOUT WINDOW
L-1 TRIGGER
TTC
Timing, Trigger and Control
EC RESET
EVENT COUNTER
(PROCESSING ELEMENT)
DERANDOMIZER
TEST SIGNALS
INITIALIZATION
FAST CONTROLS
PARAMETERS
MULTIPLEXER (SPARSE READOUT)
(ADC, PROCESSING ELEMENT)
FE IDENTIFIER
TIME TAG
EVENT TAG
FORMATTING
TTS
Trigger Throttle System
BACK PRESSURE
Buffer status, Error reports
READOUT LINK INTERFACE
Data acquisition