Trigger Architectures and Hardware

ISOTDAQ 2018
Vienna

February 14, 2018

Manfred Jeitler
Institute of High Energy Physics, Vienna

Acknowledgments

This lecture was previously prepared and presented by

Francesca Pastore and Alessandro Thea

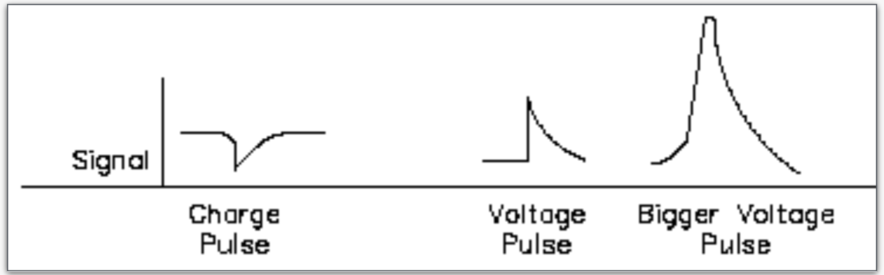
What do we want:

- Get the data we want: efficiency
- Get ***only*** the data we want: purity
- Be able to afford the system: cheap
- No breakdowns: robust
- Adjust to changing conditions: flexible

The simplest trigger system

Source: signals from detector (“detector Front-Ends”)

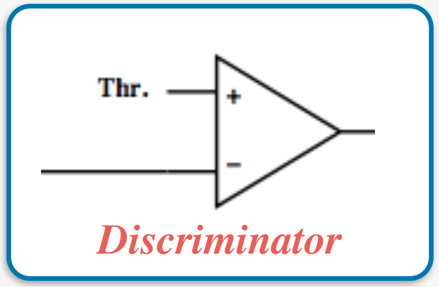
- **Binary**: e.g. tracking detectors (pixels, strips): yes/no
- **Analog**: e.g. calorimeters: pulse height



Front-End *Pre-amplifier* *Amplifier*

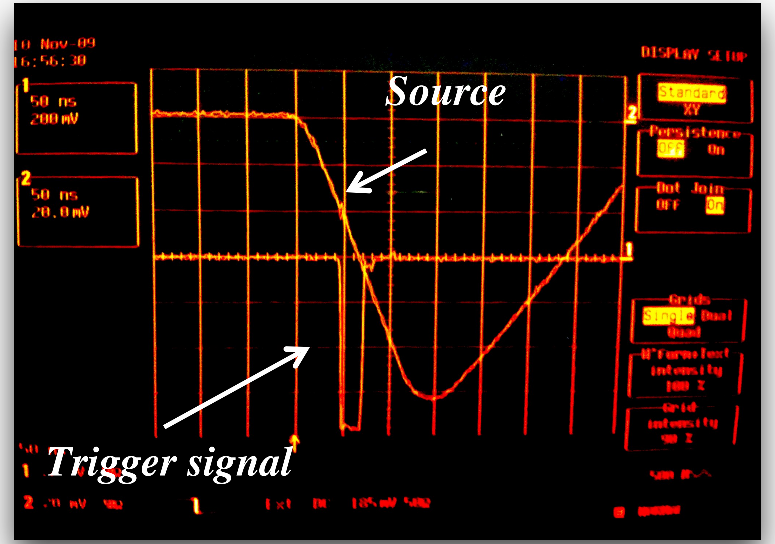
The most trivial trigger algorithm: ***Signal > Threshold***

- **lowest possible threshold**
- compromise between **signal efficiency and noise**



Accept

Reject



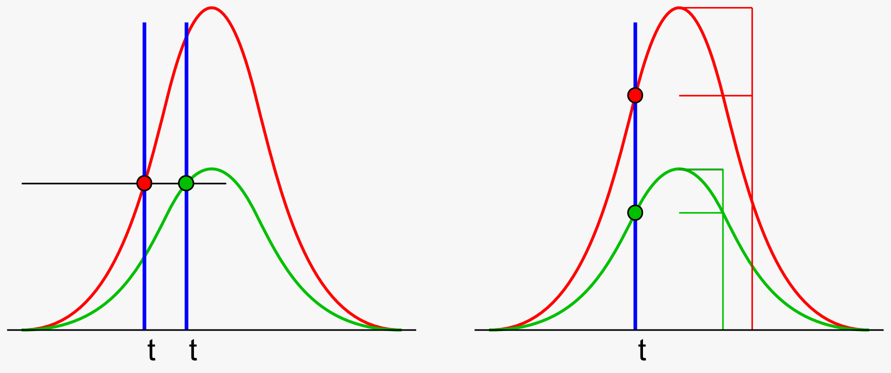
Detector signal characteristics

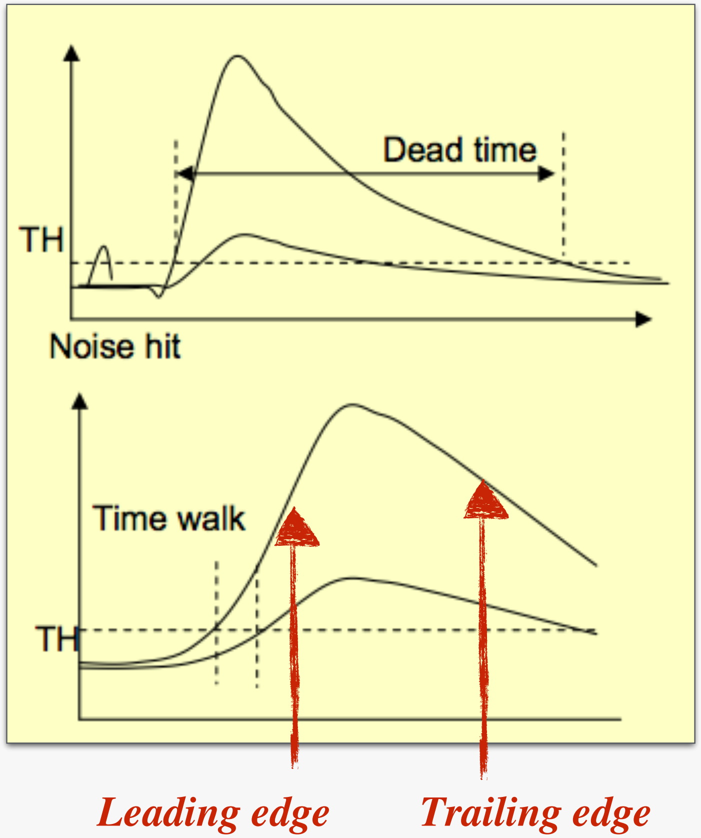
Pulse width

- Limits the usable hit rate
- Must be adapted to the desired trigger rate

Time walk

- The threshold-crossing time depends on the signal amplitude
- Affects timing resolution





Time walk can be suppressed by triggering on **total signal fraction**

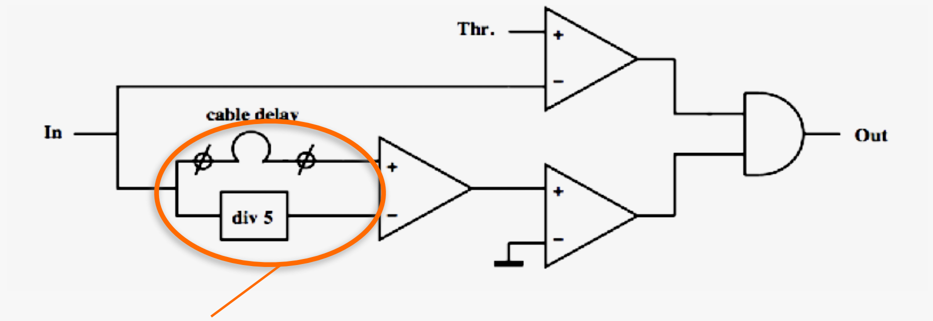
- Applicable on same-shape input signals with different amplitude
  - e.g., from scintillators

Constant fraction discriminator

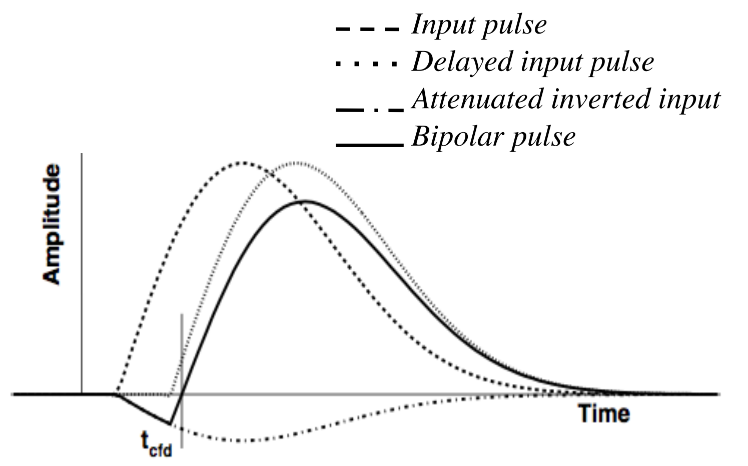
Signals with same rise time, at fraction f

$$\Delta t_f = t(f \cdot A_0) - t(A_0) = \text{const.}$$

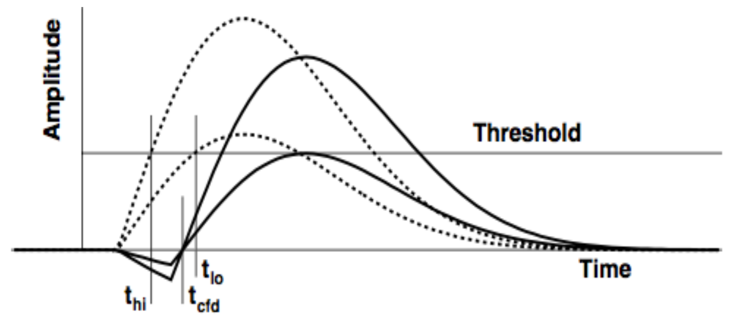
$$A(t)/f - A(t - \Delta t) = 0 \ \textbf{at} \ t = t_{CDF}$$



- **Attenuation + configurable delay**
  - applied before the discrimination determines t_{CFD}
- If delay too short, the unit works as a normal discriminator
  - the output of the normal discriminator fires later than the CFD part



The output of the CFD fires when the bipolar pulse changes polarity

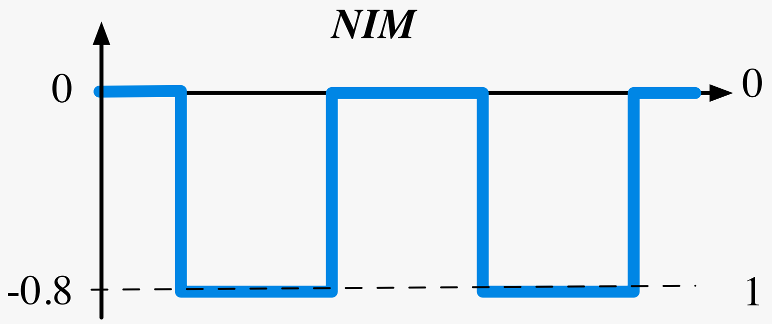


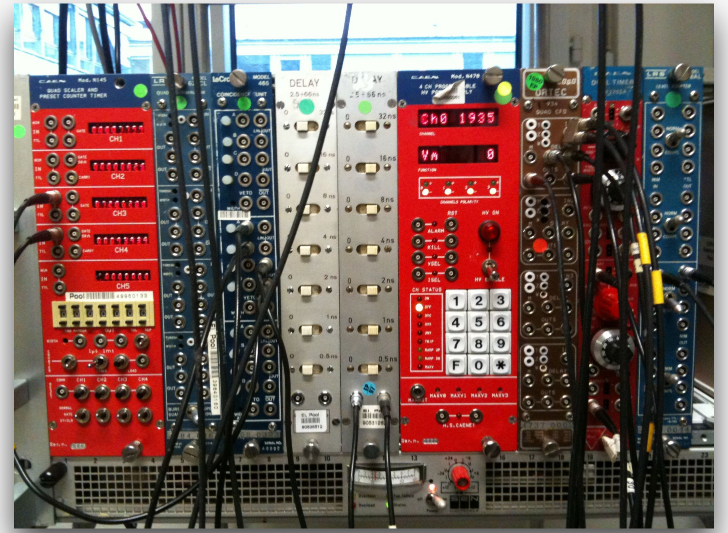
Build your own trigger system

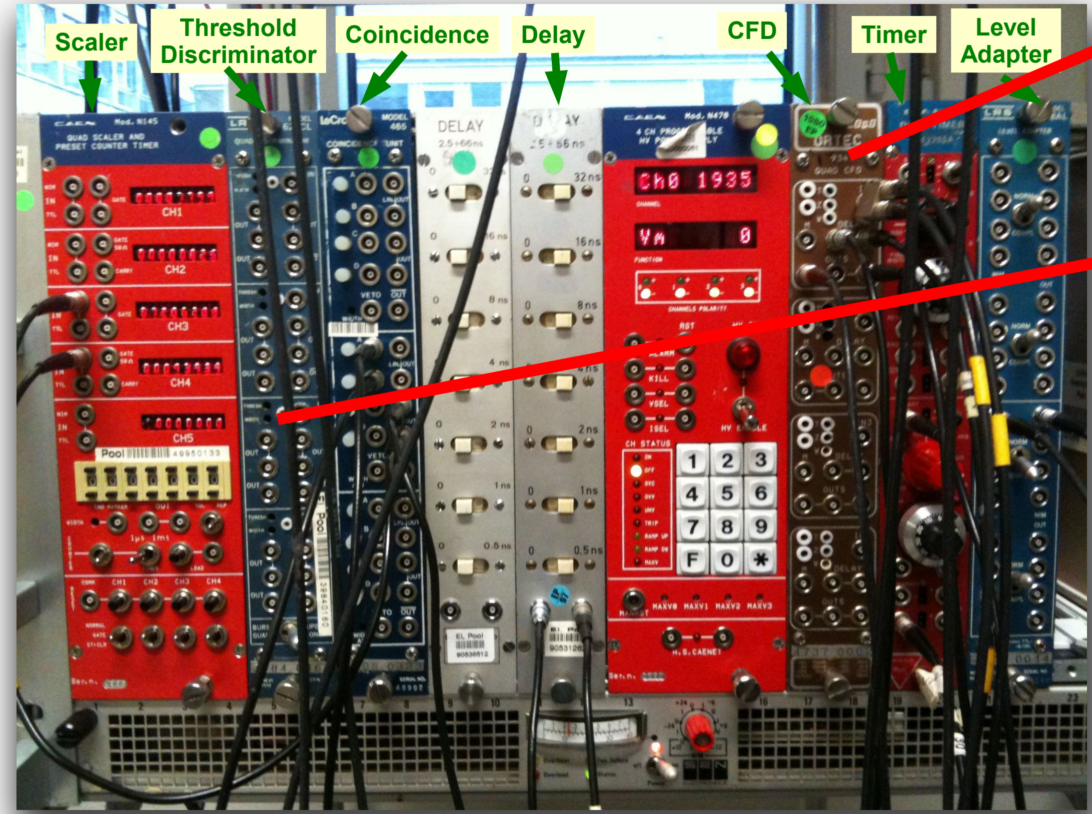
A simple trigger system can start with a **NIM crate**

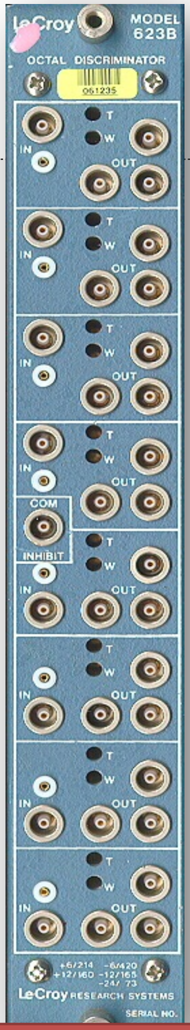
- Common support for electronic modules,
  - standard impedance, connections, logic levels (negative)

- **-16 mA into 50 Ohms = -0.8 Volts**









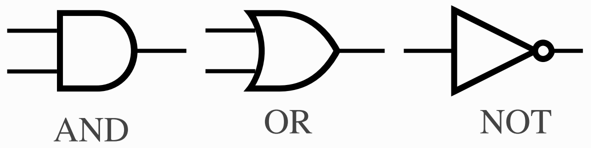
- Threshold levels configurable via screwdriver adjust

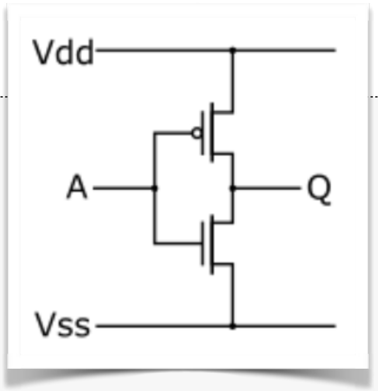
ORTEC CFD

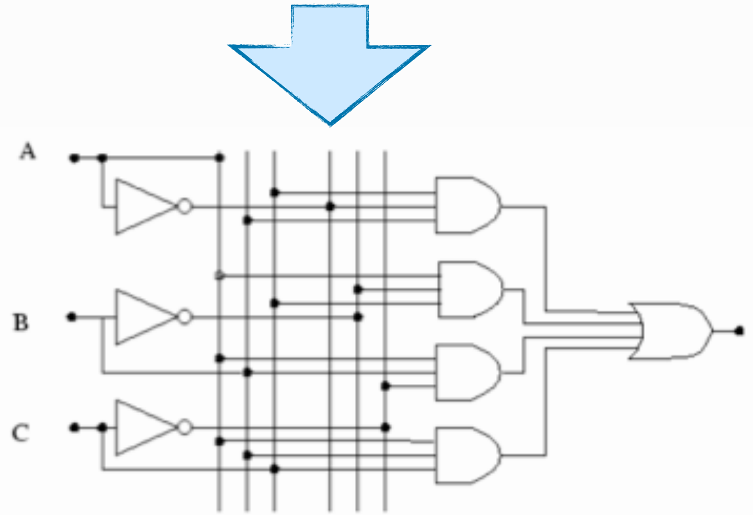
LeCroy discriminator

Trigger logic implementation

decision logic described by mathematical operators







Analog systems

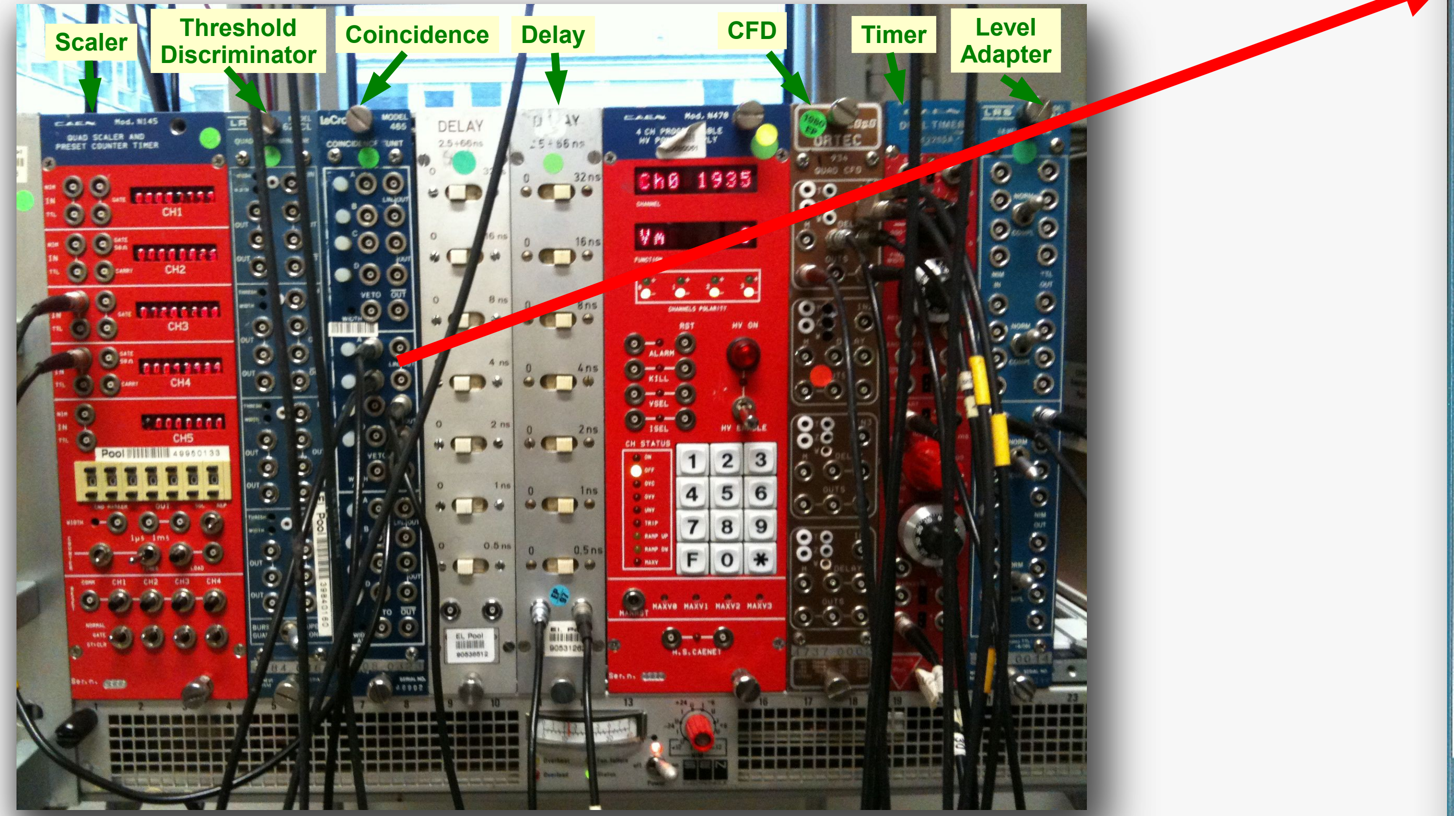
- amplifiers, filters, comparators, …

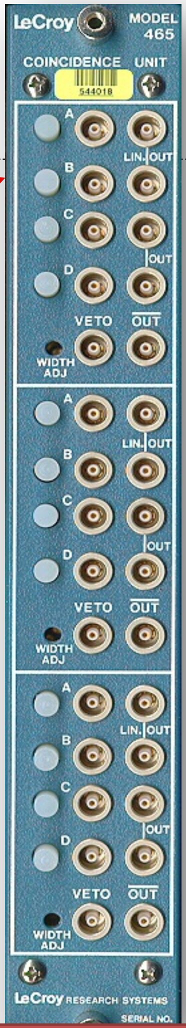
Digital systems

- **Combinatorial**
  sum, decoders, multiplexers,…
- **Sequential**
  flip-flop, registers, counters,…

Converters

- ADC, TDC, …..



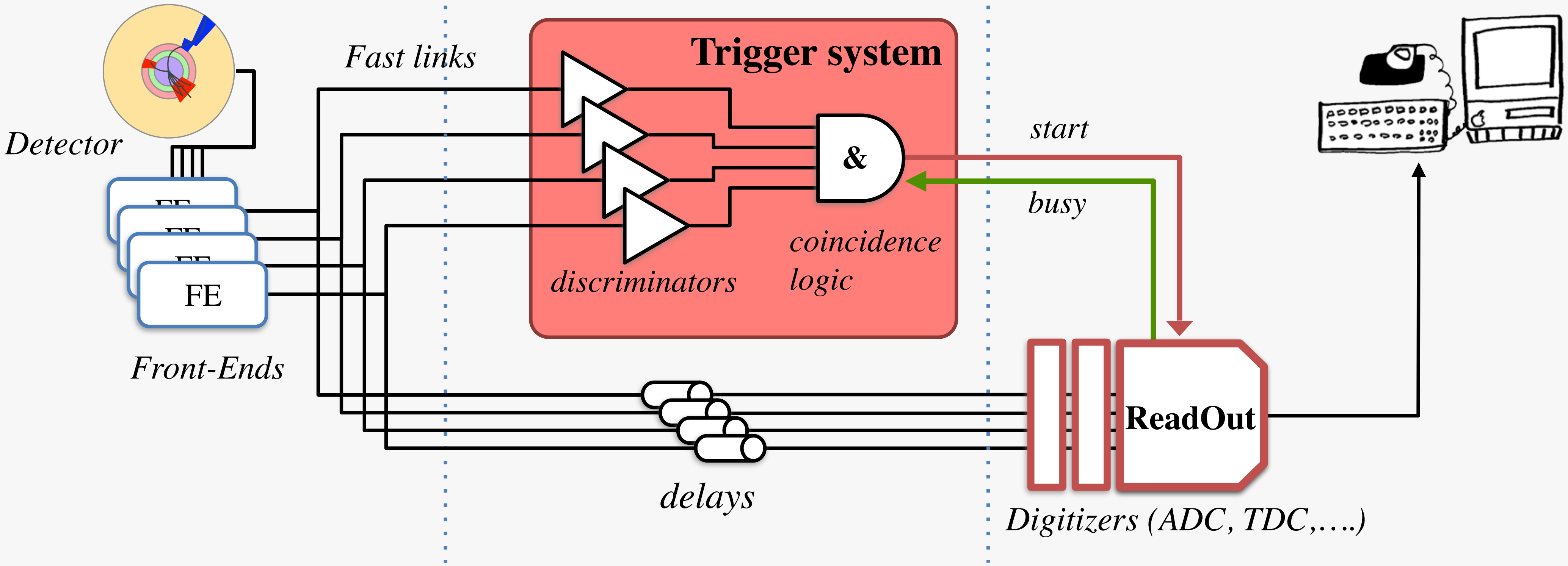


LeCroy Concidence Unit

We want:

High Efficiency
Low dead-time
Fast decision

Wait … I'm busy !!



- Incoming event rate can temporarily exceed processing rate due to fluctuations
  - Trigger signals are then rejected if busy is high, i.e. if previous event is still being processed

Dead Time

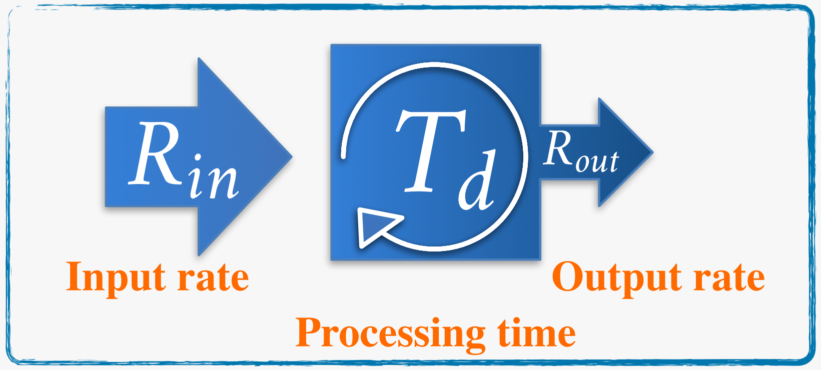
The key parameter in high speed T/DAQ systems design

- The fraction of the acquisition time when no events can be recorded.
  - Typically of the order of a **few %**
- Reduces the overall system efficiency

$$\epsilon' = \epsilon \cdot (1 - \tau_d)$$

Arises when a given processing step takes a finite amount of time

- **Readout dead-time**
- **Trigger dead-time**
- **Operational dead-time**



Maximising data-recording rate

R_{in} = Trigger rate (average)

R_{out} = Readout rate

T_d = processing time of one event

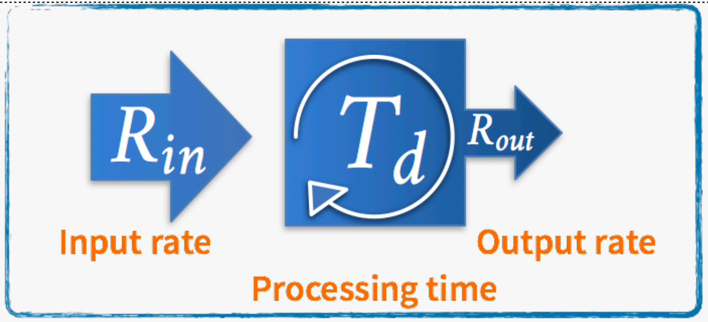
Fraction of lost events $R_{out} \cdot T_d$

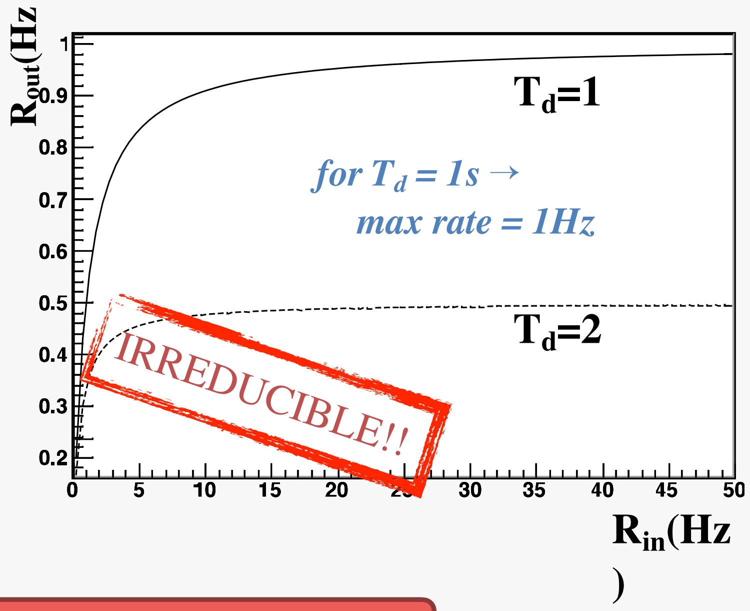
Number of events read $R_{out} = (1 - R_{out} \cdot T_d) \cdot R_{in}$

Fraction of surviving events

$$\frac{R_{out}}{R_{in}} = \frac{1}{1 + R_{in} T_d}$$

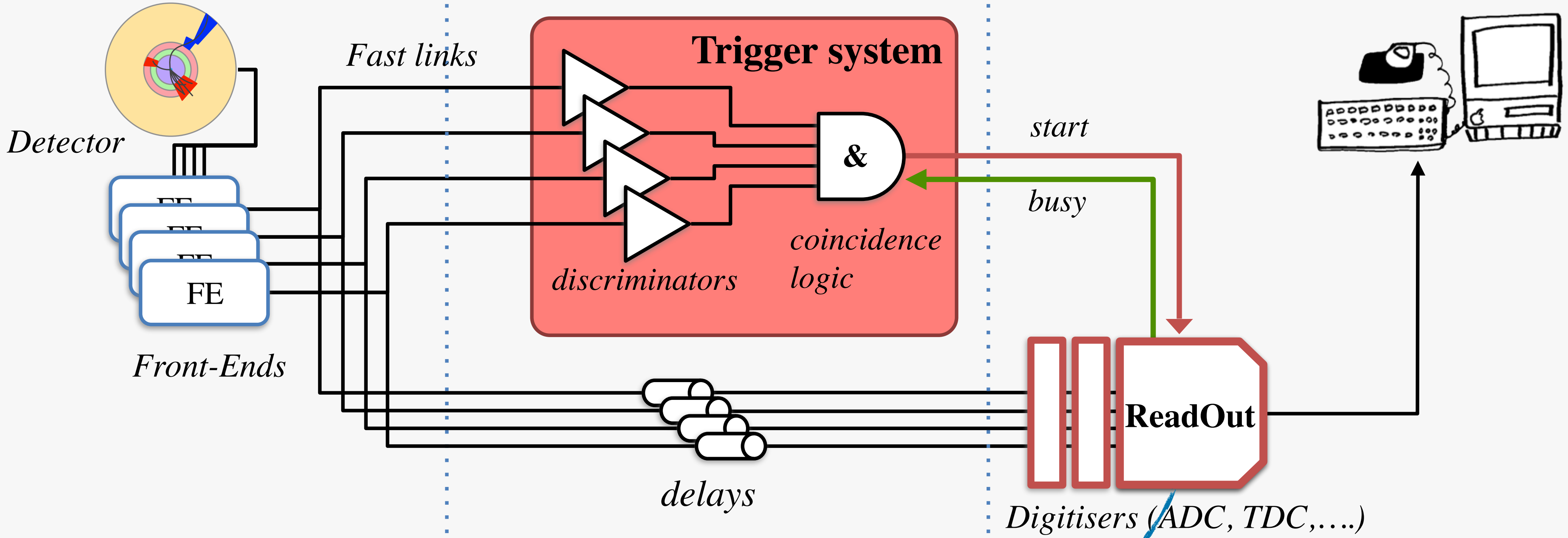
- For instance: $R_{in} = 1/T_d \rightarrow$ *dead-time = 50%*





To achieve high efficiency $\Longrightarrow R_{in} \cdot T_d \ll 1$

A simple trigger system

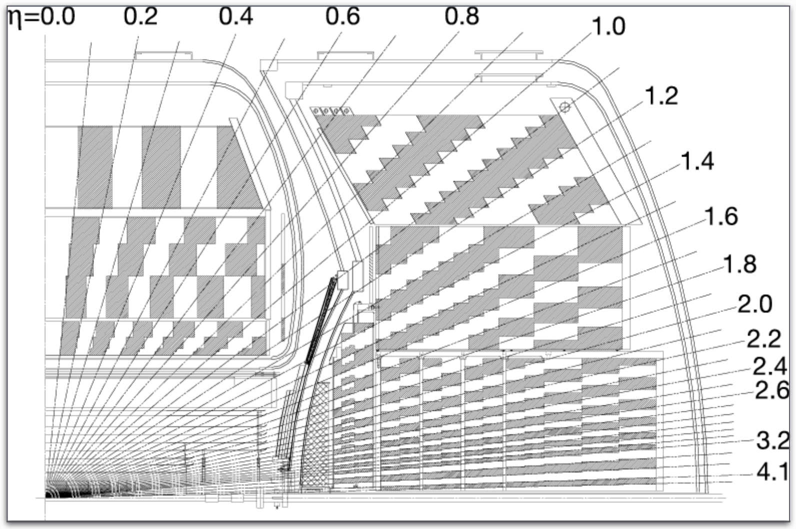


$$\tau_d = R_T \cdot T_{digi+RO}$$

Fraction of lost events due to finite digitisation & readout time

Approaches to minimise dead time

DZero calorimeters



transverse and longitudinal segmentation pattern

1. Parallelism

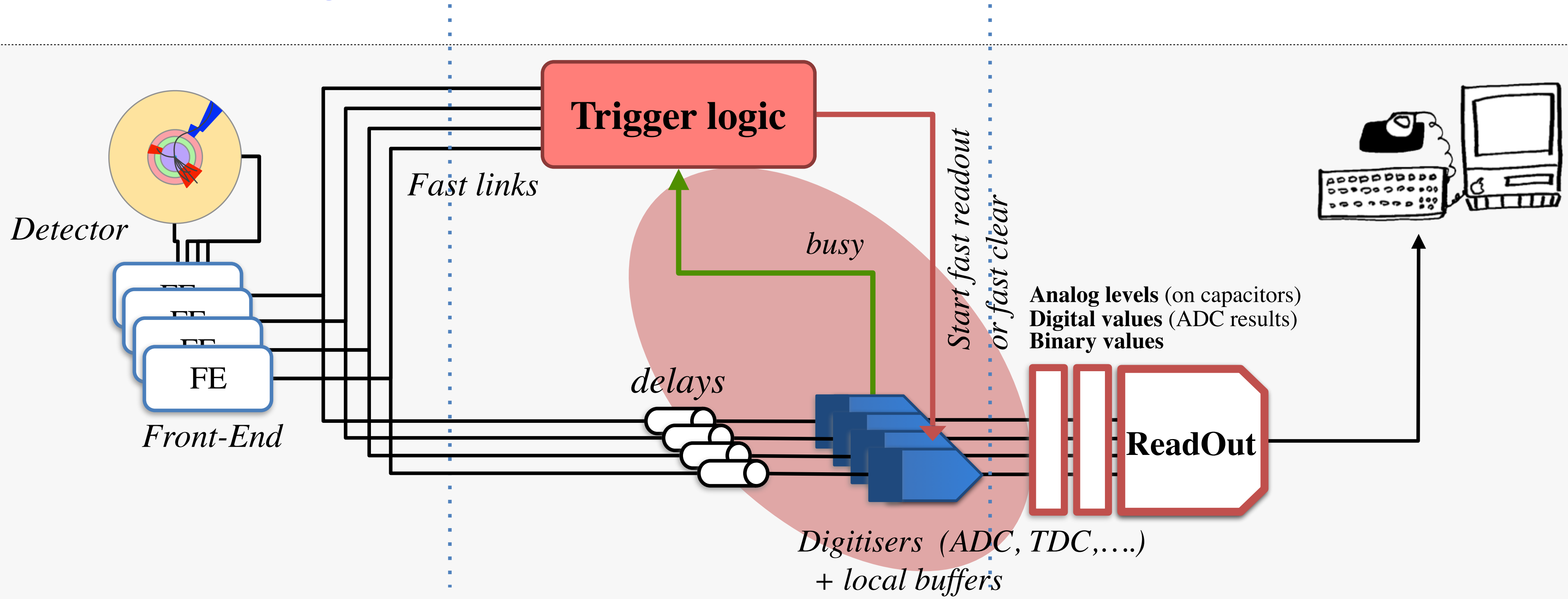
- Independent readout and trigger paths, one for each sensor element
- Digitisation and DAQ processed in parallel (as much as affordable!)

2. Pipelining to absorb rate fluctuations

- Processing structured in independent steps

$$\begin{tikzcd}
 & & \boxed{\text{Step2}} \arrow[dr] & & & \\
\boxed{\text{Step1}} \arrow[r] & \boxed{\text{FIFO}} \arrow[ur] \arrow[dr] & & \boxed{\text{FIFO}} \arrow[r] & \boxed{\text{Step3}} \arrow[r] & \boxed{\text{FIFO}} \\
 & & \boxed{\text{Step2}} \arrow[ur] & & &
\end{tikzcd}$$

Minimising readout deadtime



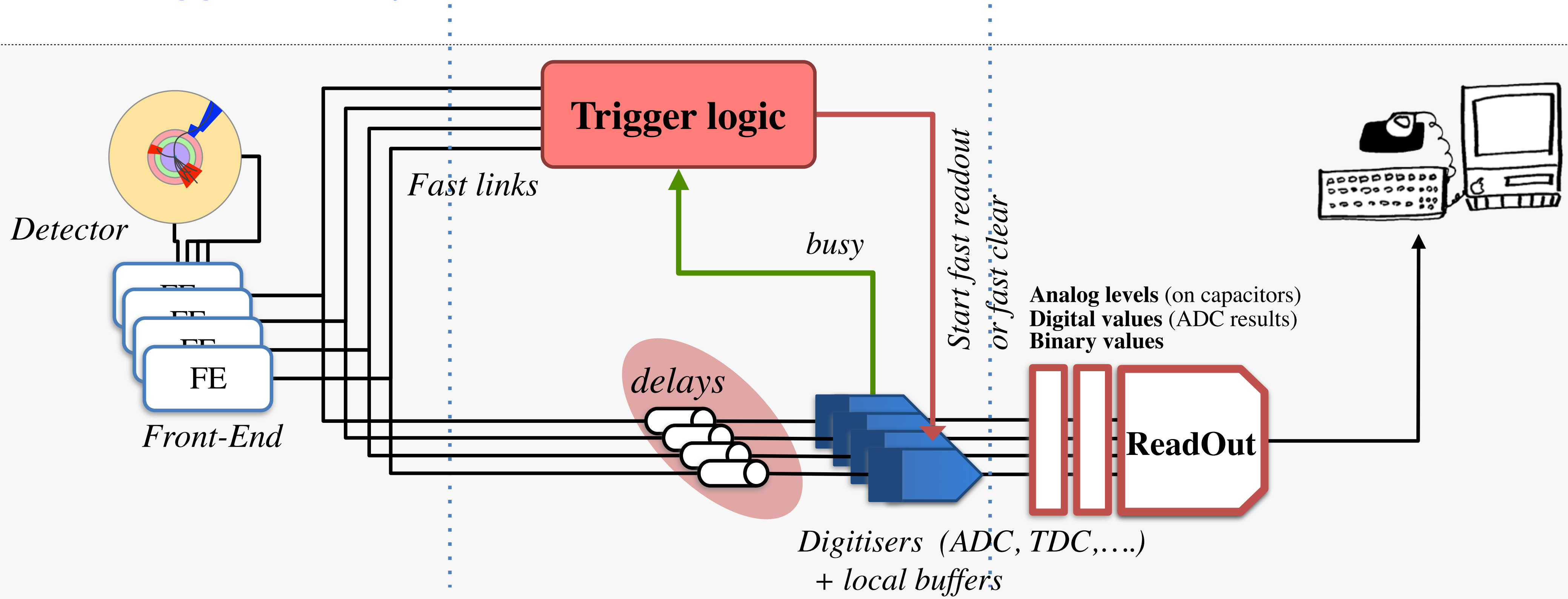
Parallelism: Use multiple digitisers

Pipelining: Different stages of readout

- **local readout (fast) + global event readout** (slow)

$$\tau_d = R_T \cdot T_{LRO}$$

Trigger latency & deadtime



Latency: time to form the trigger decision and distribute to the digitisers

- Signals must be delayed until the trigger decision is available
- The more complex is the selection, the longer is the latency

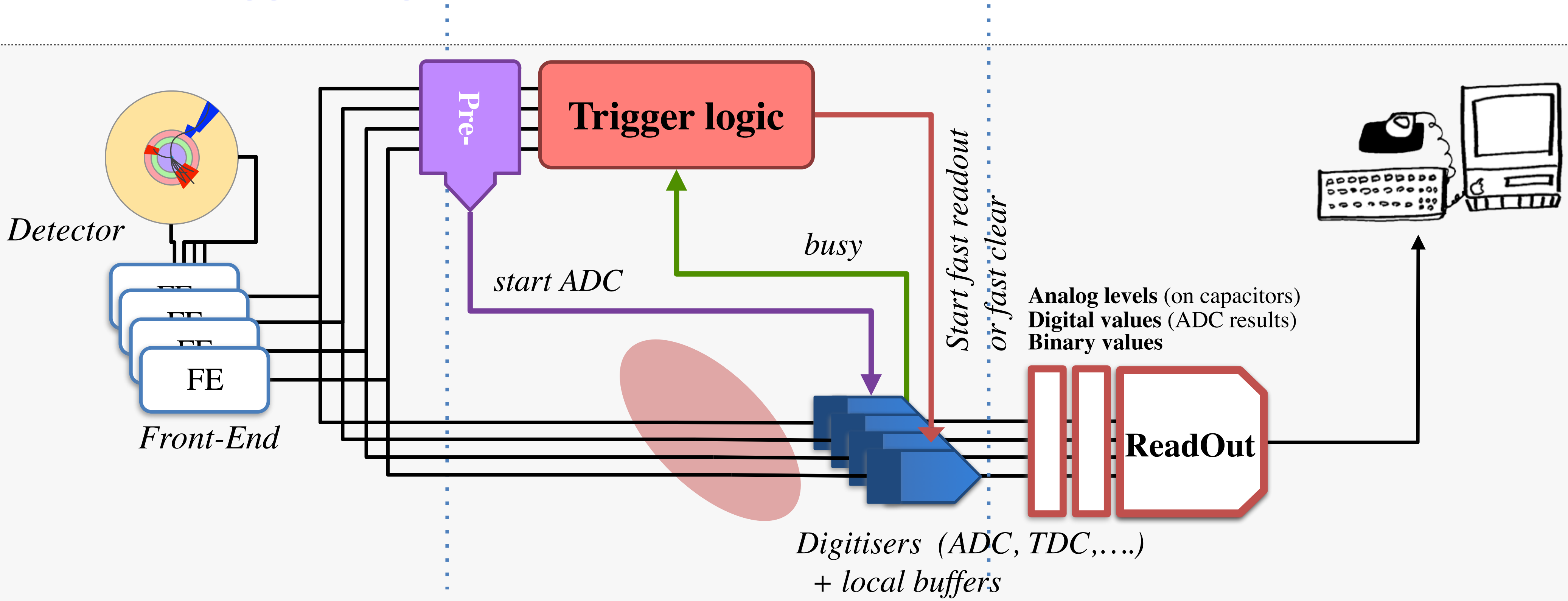
$$\tau_d = R_T \cdot T_{LRO}$$

Latency

I'm late! I'm late!

- latency is an important constraint on trigger architecture
- pipeline memory is expensive
  - in terms of money, space, energy consumption
- → need fast algorithms
- no iterative loops
- small propagation times → put trigger electronics close to detector
  - but not on detector (radiation protection!)

Pre-trigger stage



Pre-Trigger stage: **very fast**, indicating presence of minimal activity in the detector

- Used to START the digitisers, with no delay
- The complex trigger decision comes later
- L_T : pre-trigger processing time ("Latency")

$$\tau_d = R_{pT} \cdot L_T + R_T \cdot T_{LRO}^{fast}$$

Trigger and Readout dead time coupling

Extend the idea… **multiple trigger levels**

- Complexity of algorithms increases at each level
- Each stage further reduces the rate
- Later stages have longer latency

Dead-time is the sum of the trigger dead-time, summed over the trigger levels, and the readout dead-time

$$\tau_d^{multi} = \left(\sum_{i=1}^{N} R_{i-1} \cdot L_i \right) + R_N \cdot T_{LRO}$$

Pre-trigger
$i=0$

R_i = Rate after the i-th level
L_i = pre-trigger processing time for the i-th level
T_{LRO} = Local readout time

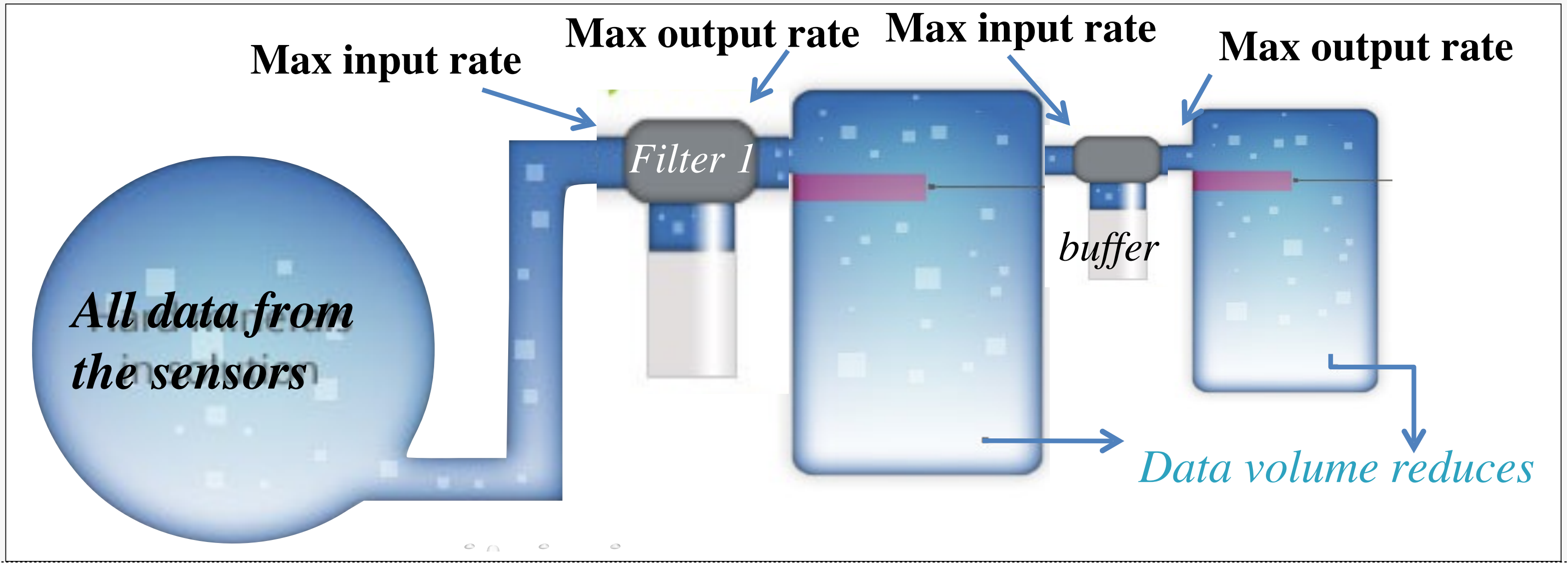
Readout dead-time is driven by the final-level trigger rate

Buffering

At each stage, data volume is reduced

- **input rate** constrains the filter processing time and the buffer size
- **output rate** limits the maximum latency allowed in the next step

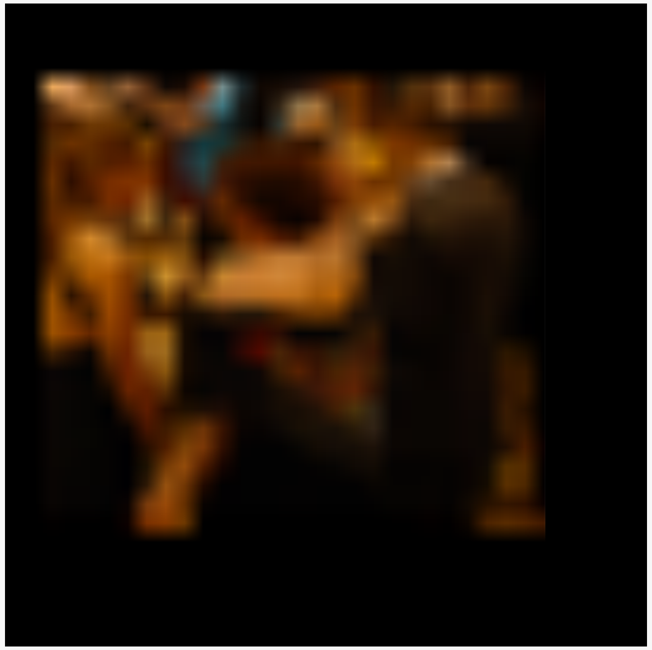
No additional dead-time is introduced,
unless buffers fill up (overflow)



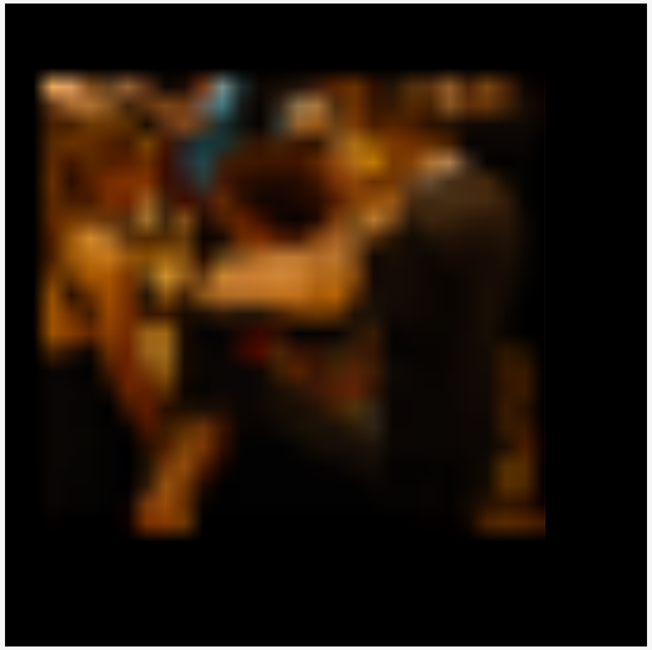
Multi-level triggers

- More and more complex algorithms are applied on lower and lower data rates
  - First level with short latency, working at higher rates
  - Higher levels apply further rejection, with longer latency (more complex algorithms)

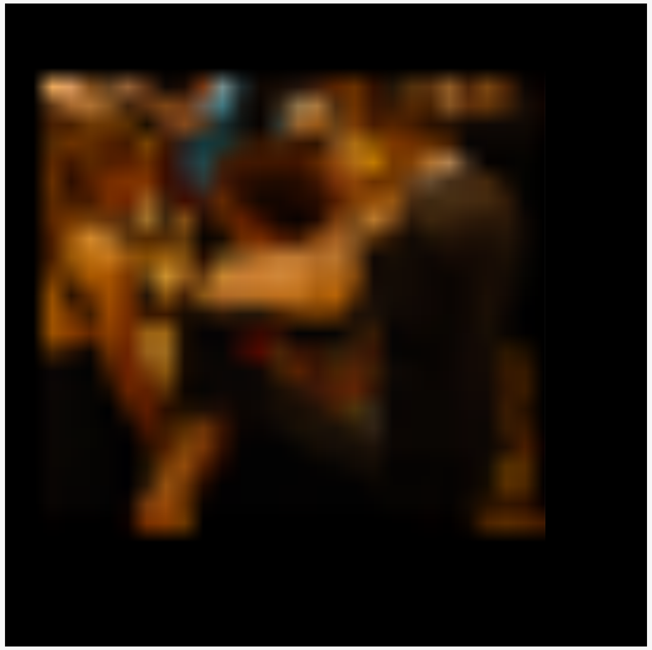
Multi-level triggers



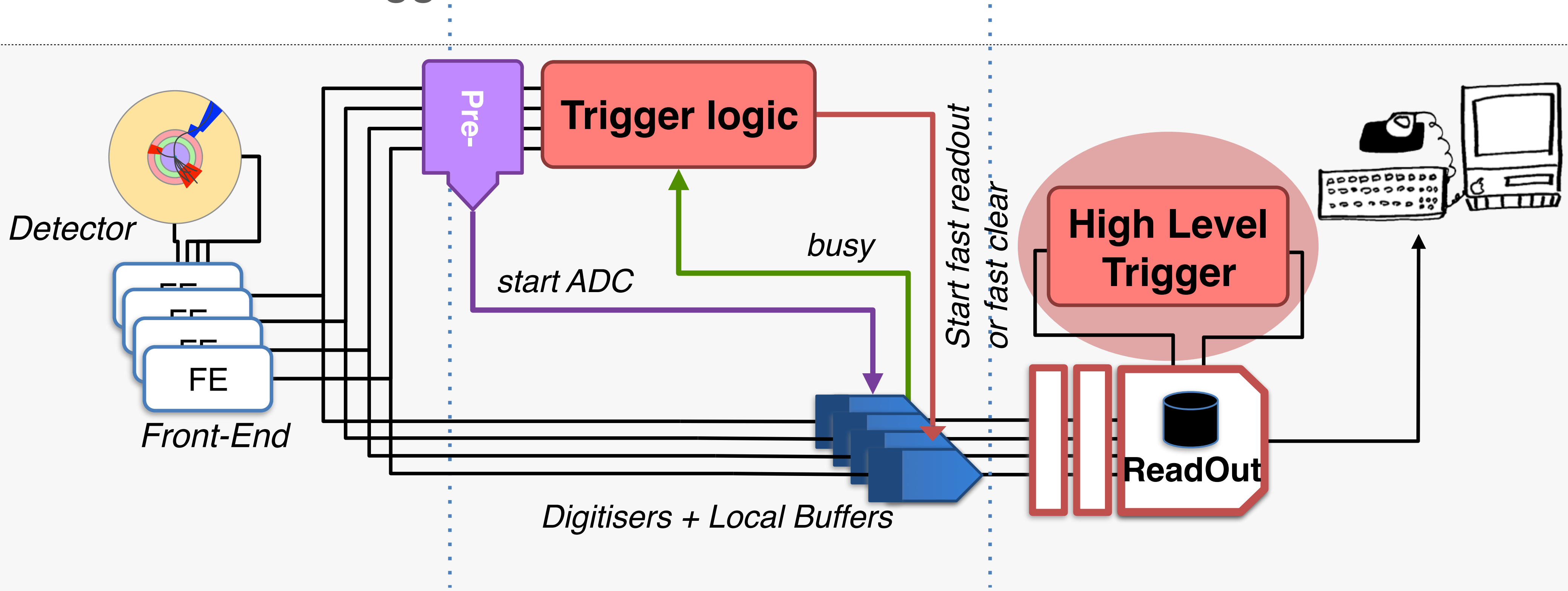
Multi-level triggers



Multi-level triggers



Multi-level trigger architecture

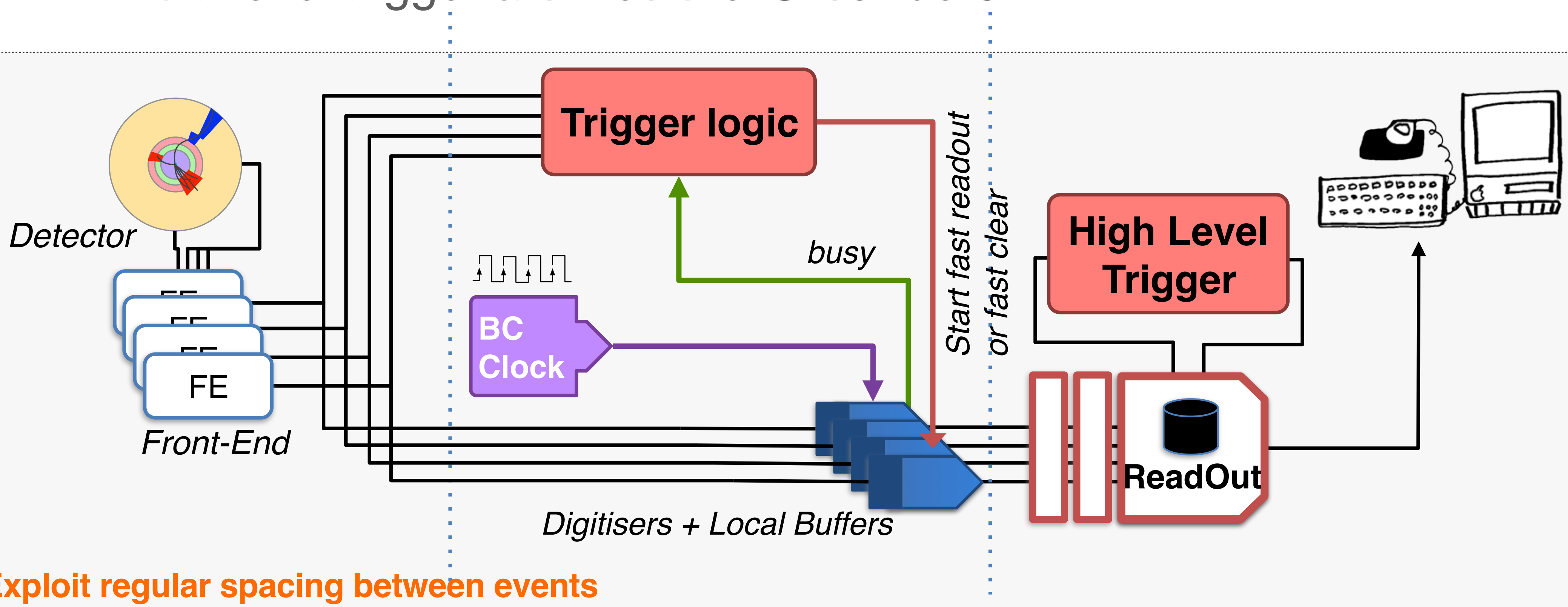


Multi-level architecture

- different levels of trigger, accessing different buffers
- The pre-trigger starts the digitisation

$$\tau_d^{multi} = \left(\sum_{i=1}^{N} R_{i-1} \cdot L_i \right) + R_N \cdot T_{LRO}$$

Multi-level trigger architecture @ colliders



Exploit regular spacing between events

- BC clock starts digitisation - **No Pre-trigger dead time**
- L1 trigger synchronous to BC clock.
  - No Level-1 dead time if $L_{L1} < T_{BC}$

$$\tau_d^{multi} = \left(\sum_{i=1}^{N} R_{i-1} \cdot L_i \right) + R_N \cdot T_{LRO}$$

synchronous vs asynchronous trigger processing

- some calculations are harder, others easier
  - example: there may be many or just a few tracks
- if you put data onto a computer: some events take longer to calculate than others
  - overall computing resources will be optimally used
  - so, is this fine?

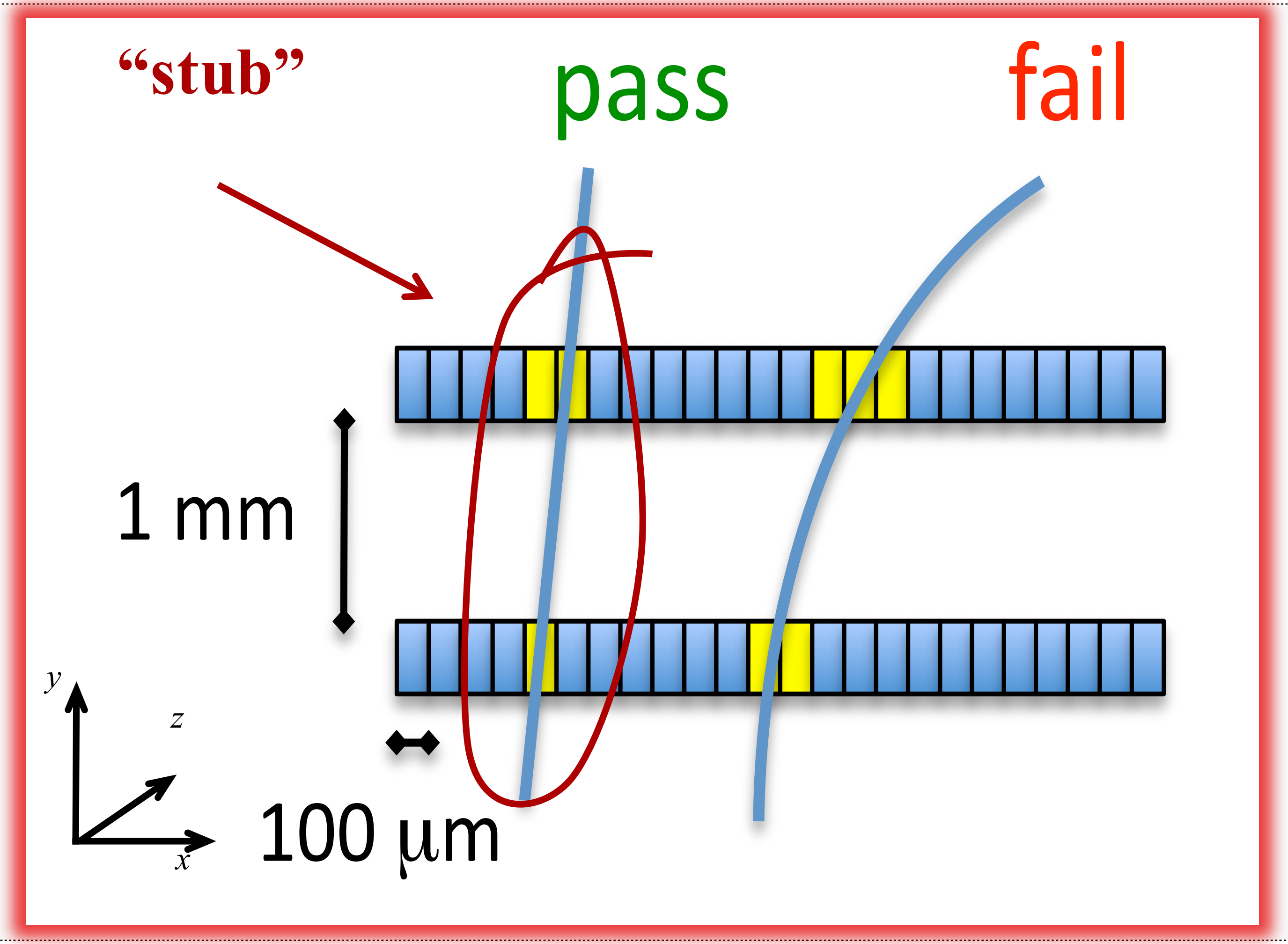
synchronous vs asynchronous trigger processing

- some calculations are harder, others easier
  - example: there may be many or just a few tracks
- if you put data onto a computer: some events take longer to calculate than others
  - overall computing resources will be optimally used
  - so, is this fine? - NO!
- danger! what if an event takes too long to process and is outside latency?
  - "watchdog" events: the watchdog will bark if you take too long!
  - just take all such events? - But there may be far too many of them!
  - just drop them? - But these may be the most interesting events! You might be killing all the "New Physics" events!
  - just take the percentage of them that you can afford? - Compromise, but may be a nightmare to analyze!

The beauty of synchronous trigger processing

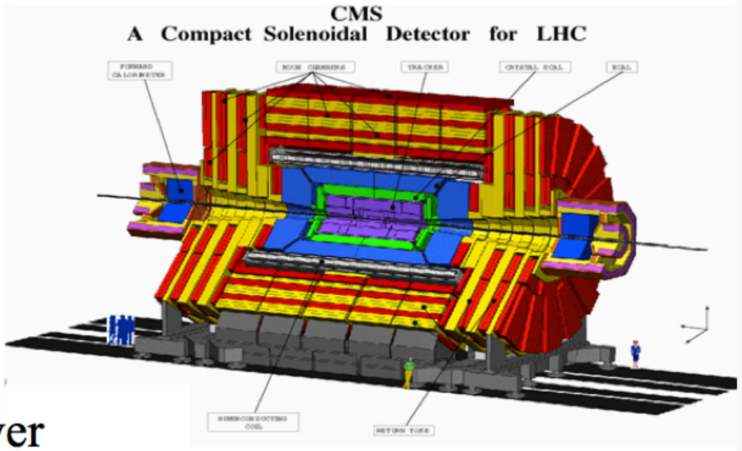
- guaranteed latency – even the most complicated calculations fit into the available processing time
  - you are just "wasting resources" in case of "simple" events
  - like an assembly line: if a worker is fast, he will be idle part of the time and you lose salary money; if he is too slow, the whole production process will crash!
- enormous resources of present-day integrated circuits (ASICs and FPGAs) make this possible
- take care to choose correct programming style!
  - no loops
  - no conditional jumps
  - make everything parallel as much as possible

Silicon tracker trigger: local intelligence



Why a hardware trigger?

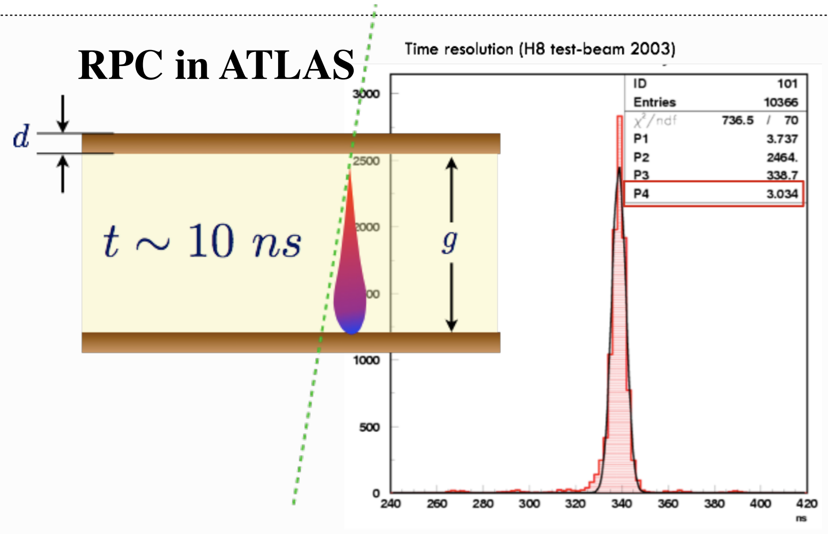
- Ideal: read out everything
  - read out detector data for every "bunch crossing": every 25 ns, so read out at 40 MHz
  - reconstruct events using all detector data in computers
  - discard data without interest before writing to tape
- **Why not work without hardware trigger?**
  - need very big computer farms (money problem)
  - *but also:*
  - have to get all data out from detector
  - have to supply detector with much power
  - not only money problem but resolution degradation due to amount of material in detector ("copper tracker")

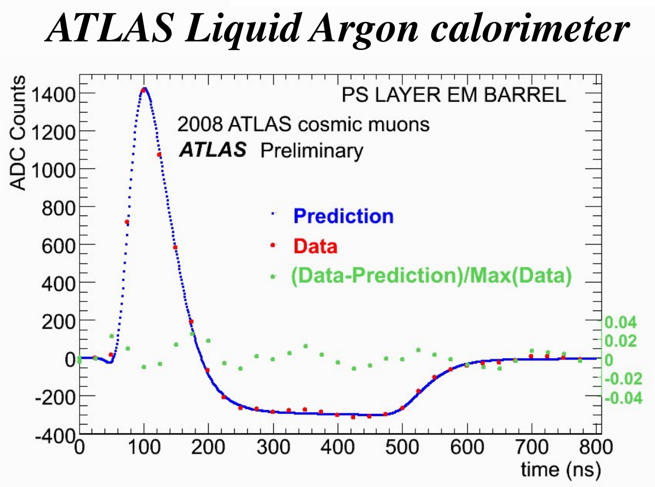


Level-1 trigger technologies

Pipelined trigger
Fast processors
Fast data transfer

How does the trigger receive input data?





Typically 'parasitic' on the main detector readout system

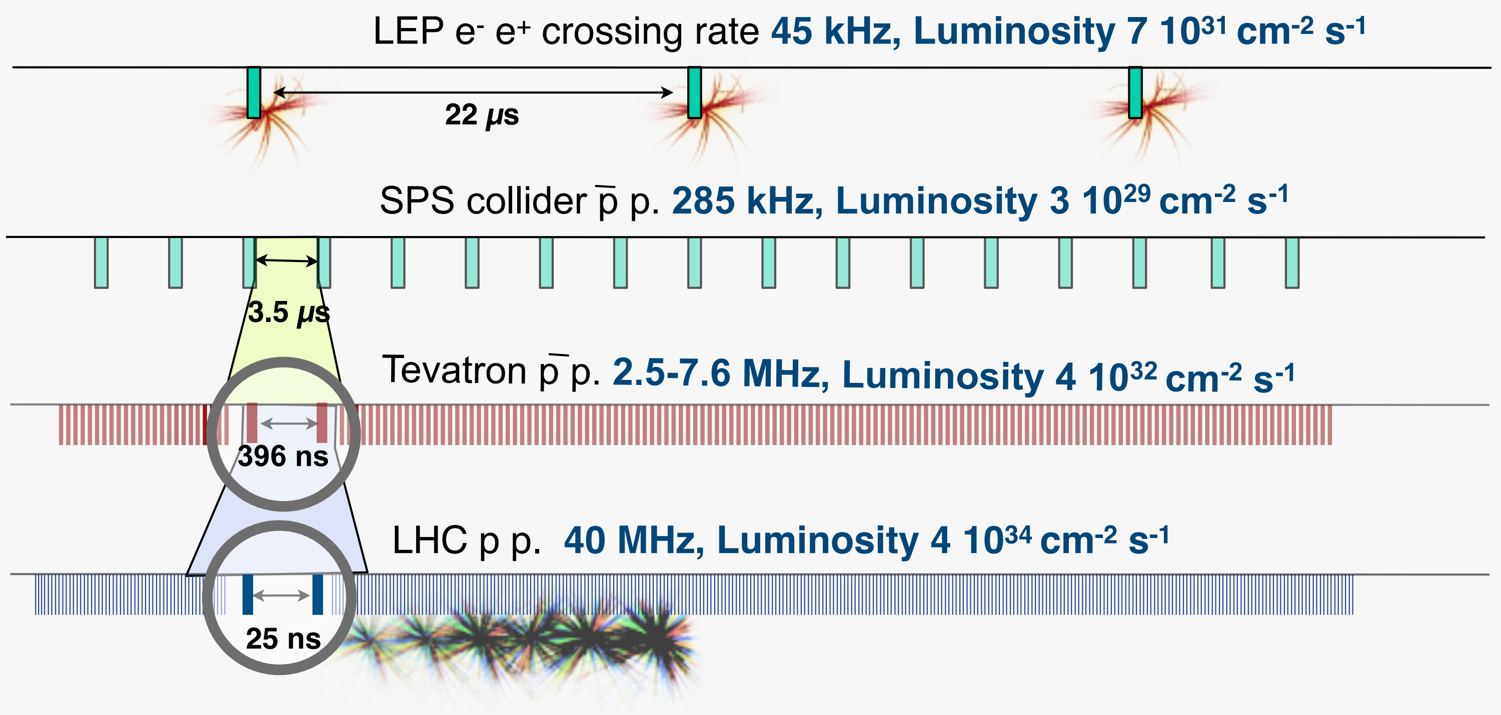
- Exception is when dedicated trigger detectors are used
  (e.g. ATLAS RPCs for muons)
- Organic scintillators
- Electromagnetic calorimeters
- Proportional chambers (short drift)
- Cathode readout detectors (RPC, TGC, CSC)

Typical requirement

- Fast signal: good time resolution and low jittering
- Shaping and on-board peak finding for slower detectors
- High efficiency
- (often) High rate capability

Need high-performance FE/trigger electronics for fast signal processing

Synchronous level-1 triggers @ colliders


LEP e⁻ e⁺ crossing rate 45 kHz, Luminosity 7 10³¹ cm⁻² s⁻¹
22 μs
SPS collider p̄ p. 285 kHz, Luminosity 3 10²⁹ cm⁻² s⁻¹
3.5 μs
Tevatron p̄ p. 2.5-7.6 MHz, Luminosity 4 10³² cm⁻² s⁻¹
396 ns
LHC p p. 40 MHz, Luminosity 4 10³⁴ cm⁻² s⁻¹
25 ns

$$R=\sigma\cdot\mathscr{L}=\mu\cdot f_{BX}$$

@LEP, BC interval **22 *μs***: complex trigger processing was possible

- **In modern colliders**: required high luminosity is driven by high rate of BC
  - **BC spacing too short for final trigger decision!**

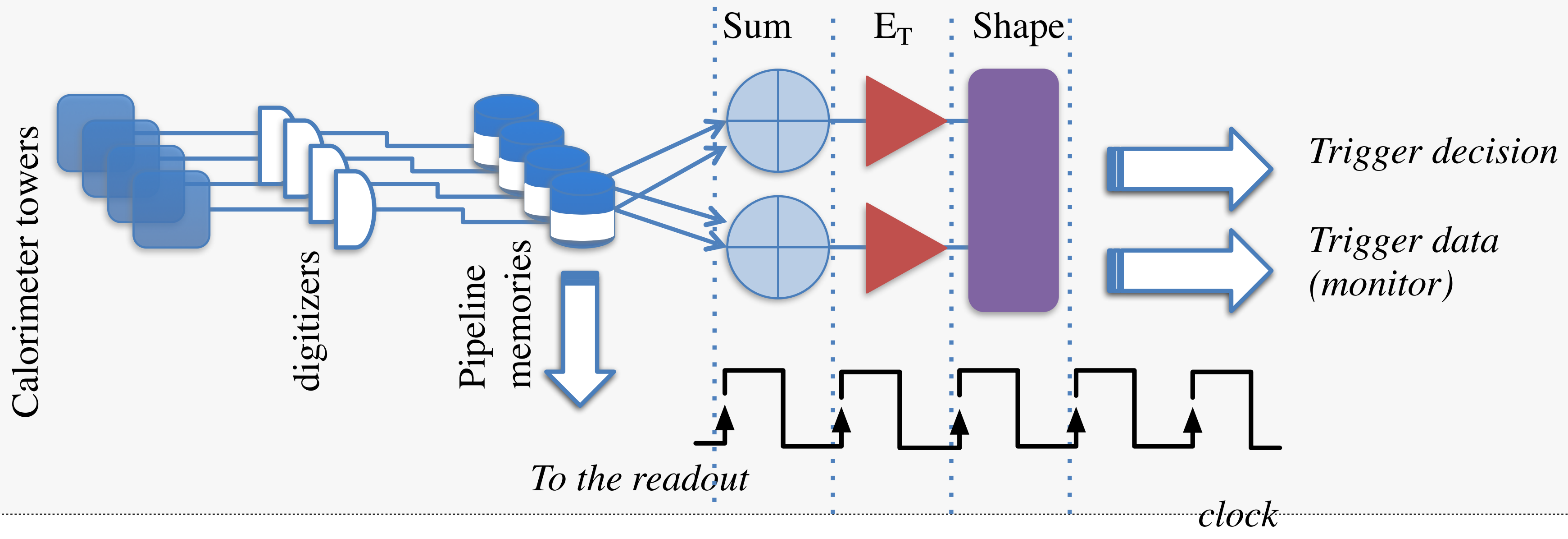
Pipelining & buffers

With a synchronous system and large buffer pipelines, longer fixed trigger latency O(μs) becomes accessible

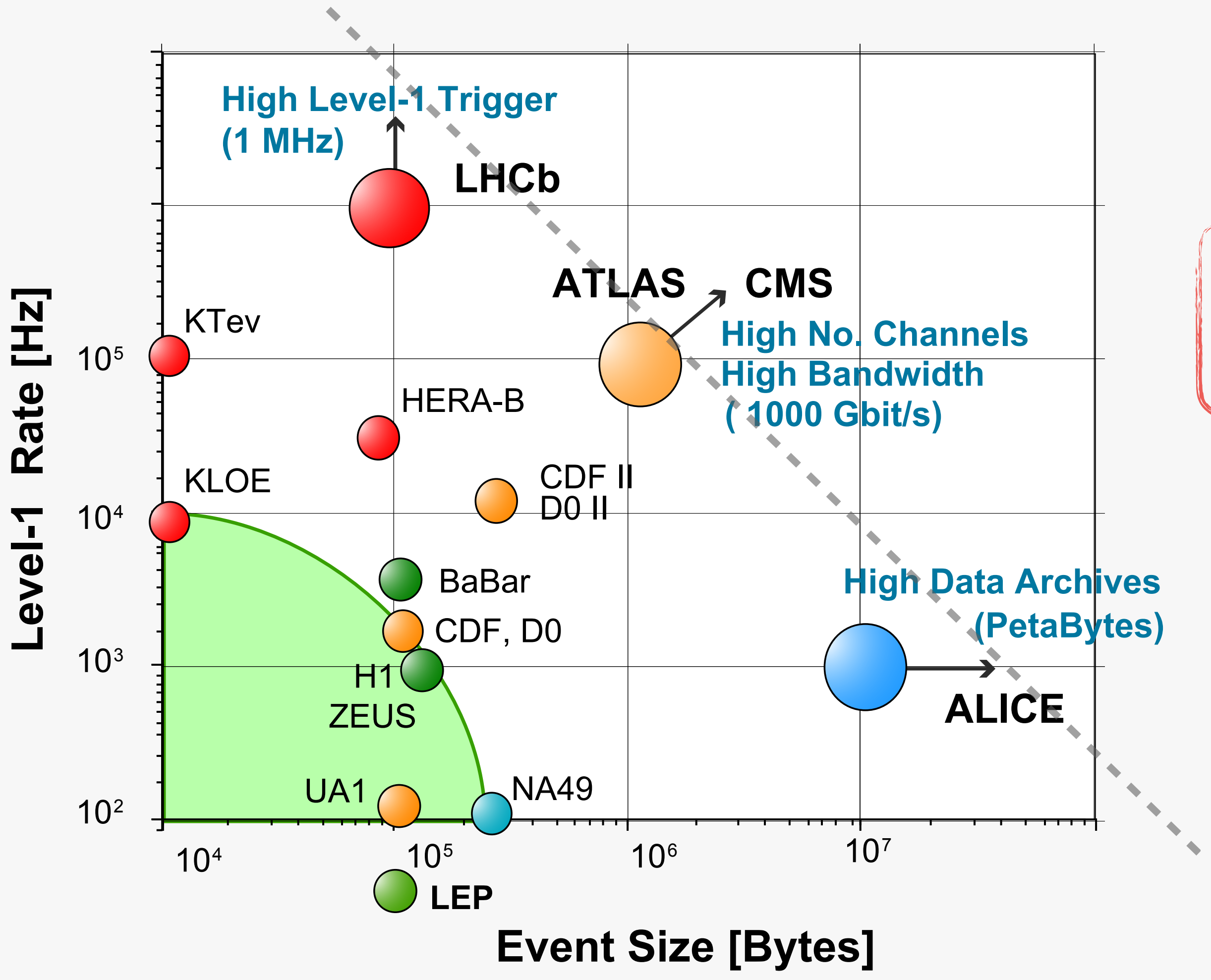
- Latency is the sum of each step processing and data transmission time

Each trigger processor concurrently processes many events

- Divide processing in steps, each performed within one BC



Trigger and data acquisition trends



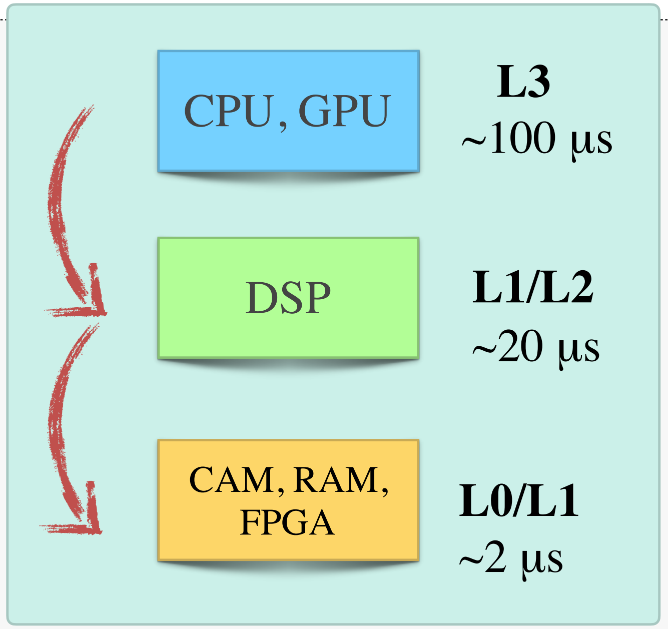
$$\boxed{B_{DAQ} = R_T^{max} \times S_E}$$

As the data volumes and rates increase, new architectures need to be developed

Programmable devices

Key requirements for high rate triggers

- Fast processing
- Flexible/programmable algorithms
- Data compression and formatting
- Monitor and automatic fault detection



Digital integrated circuits (IC)

- Reliability, reduced power usage, reduced board size and better performance

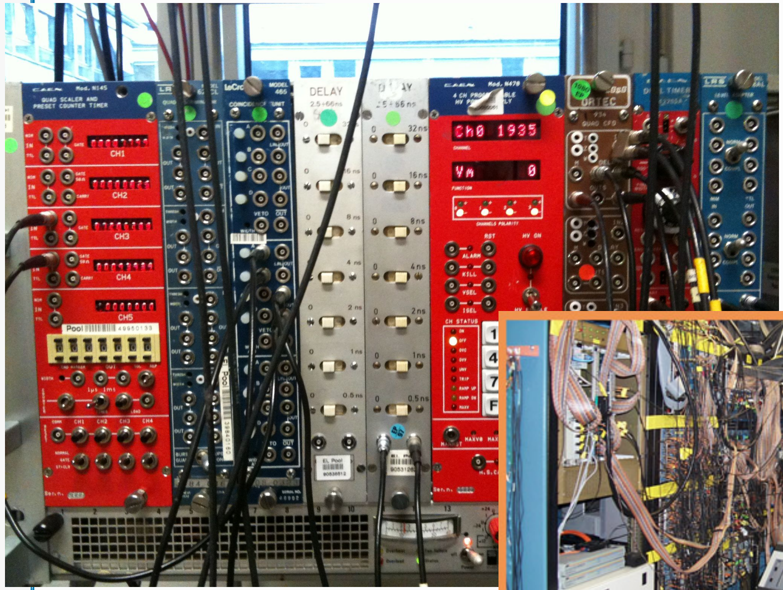
Different families on the market:

- Microprocessors (CPU, GPU, ARM, DSP=digital signal processors)
  - Available on the market or specific
- Programmable logic devices (FPGA, CAM)
  - More operations/clock cycle, but costly and difficult software developing

Choose your L1 trigger technology

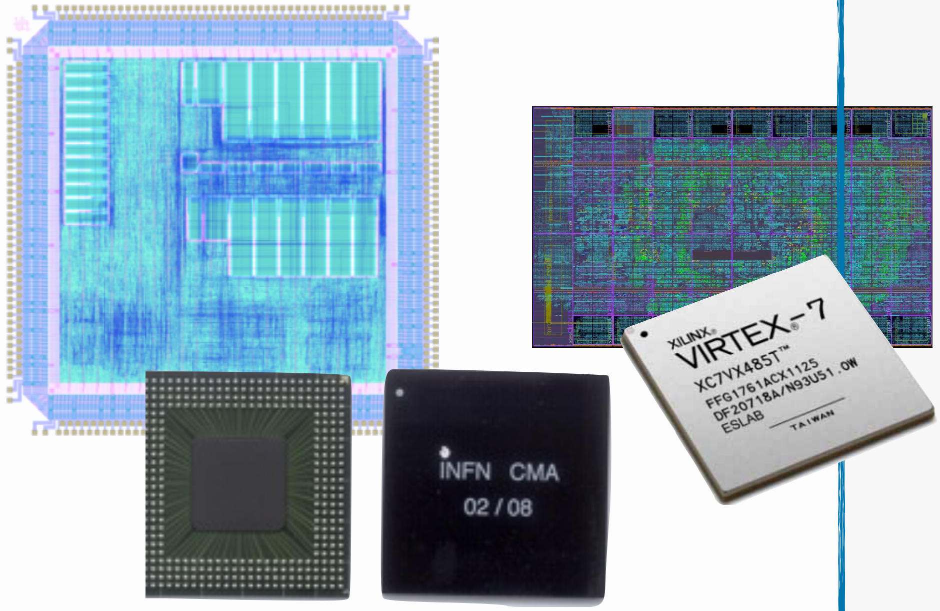
Modular electronics

- Simple algorithms
- Low-cost
- Intuitive and easy-to-use



Digital integrated systems

- Highly complex algorithms
- Fast signals processing
- Specific knowledge of digital systems



Manfred Jeitler

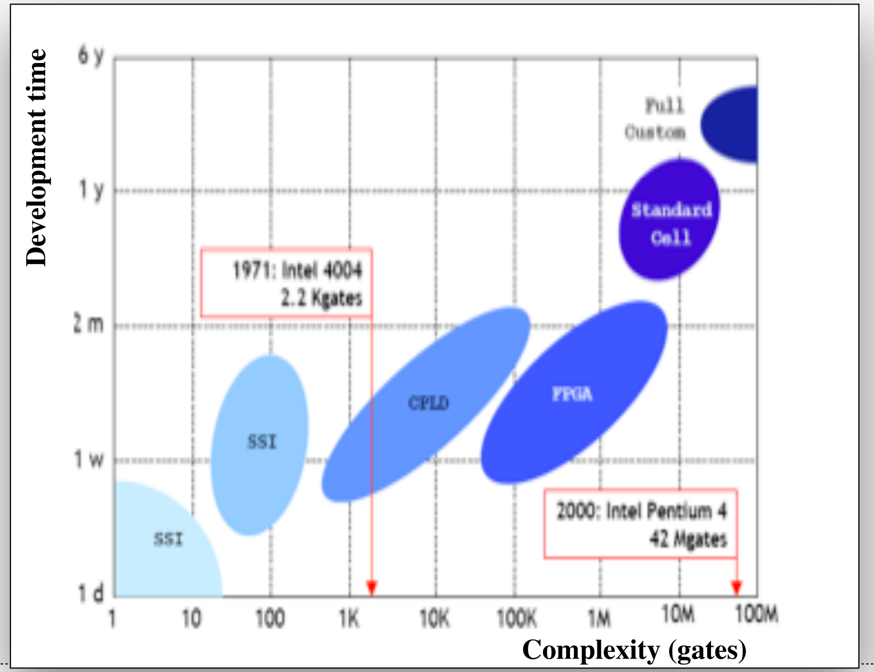
ASICs vs FPGAs: unit cost vs flexibility

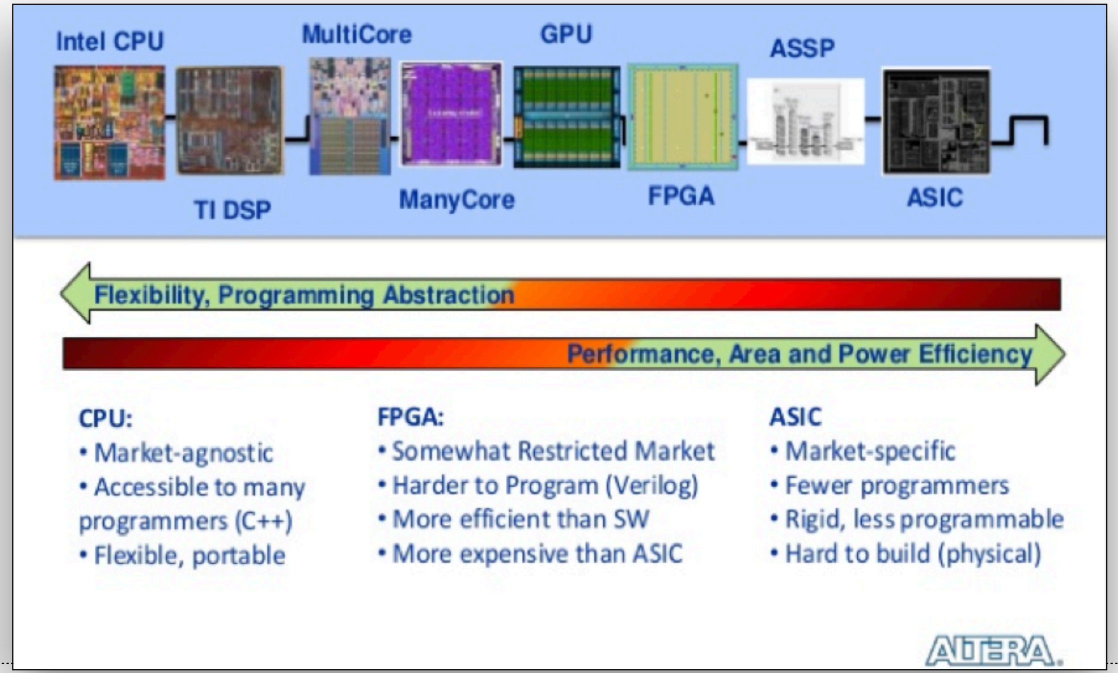
Application-specific integrated circuits (ASICs): optimised for fast processing (Standard Cells, full custom)

- Intel processors, ~ GHz

Field-programmable gate arrays (FPGAs)

- Processors @ 100 MHz easily available on the market (1/10 speed of full custom ASICs)





Intel CPU
MultiCore
GPU
ASSP
TI DSP
ManyCore
FPGA
ASIC
Flexibility, Programming Abstraction
Performance, Area and Power Efficiency
ALTERA

CPU:
- Market-agnostic
- Accessible to many programmers (C++)
- Flexible, portable

FPGA:
- Somewhat Restricted Market
- Harder to Program (Verilog)
- More efficient than SW
- More expensive than ASIC

ASIC
- Market-specific
- Fewer programmers
- Rigid, less programmable
- Hard to build (physical)

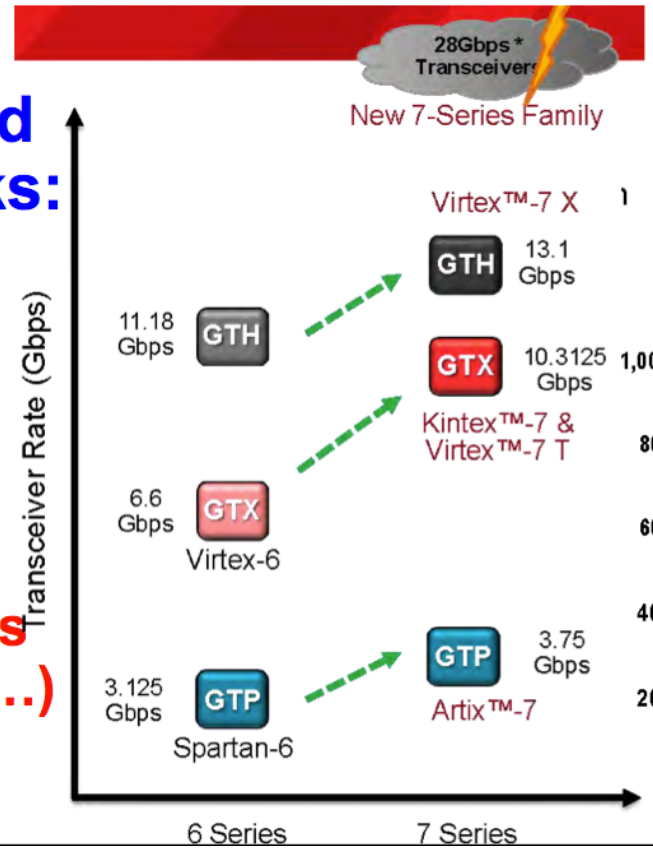
Progress in FPGAs

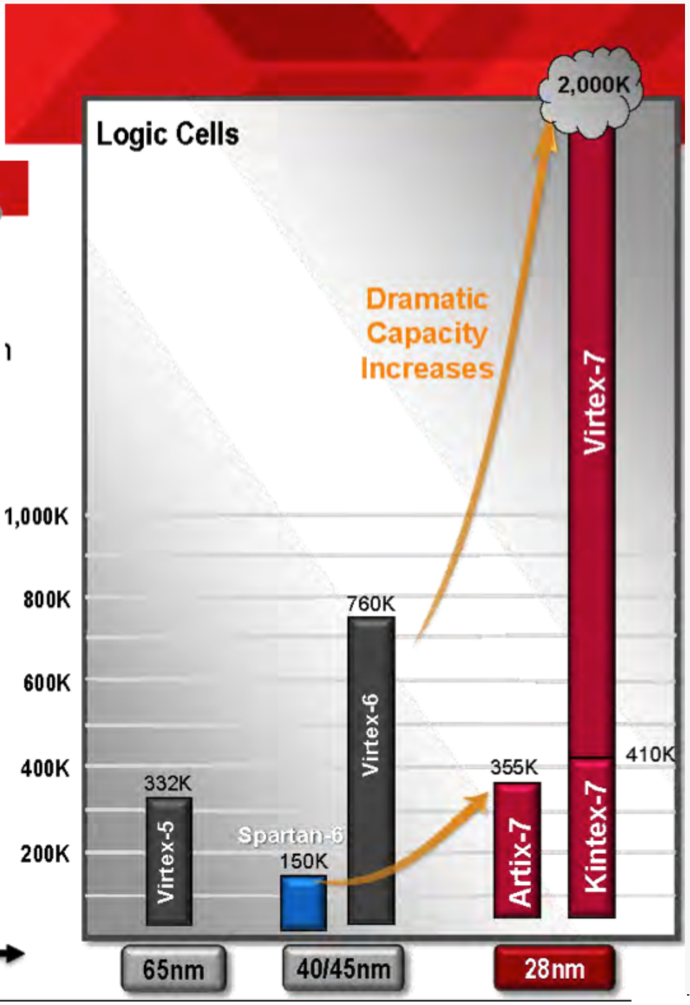
Logic Cells

- ➢28 nm: > 2X gains over 40 nm→

On-Chip High Speed Serial Links:

- ➢Connect to new compact high density optical connectors (SNAP-12…)





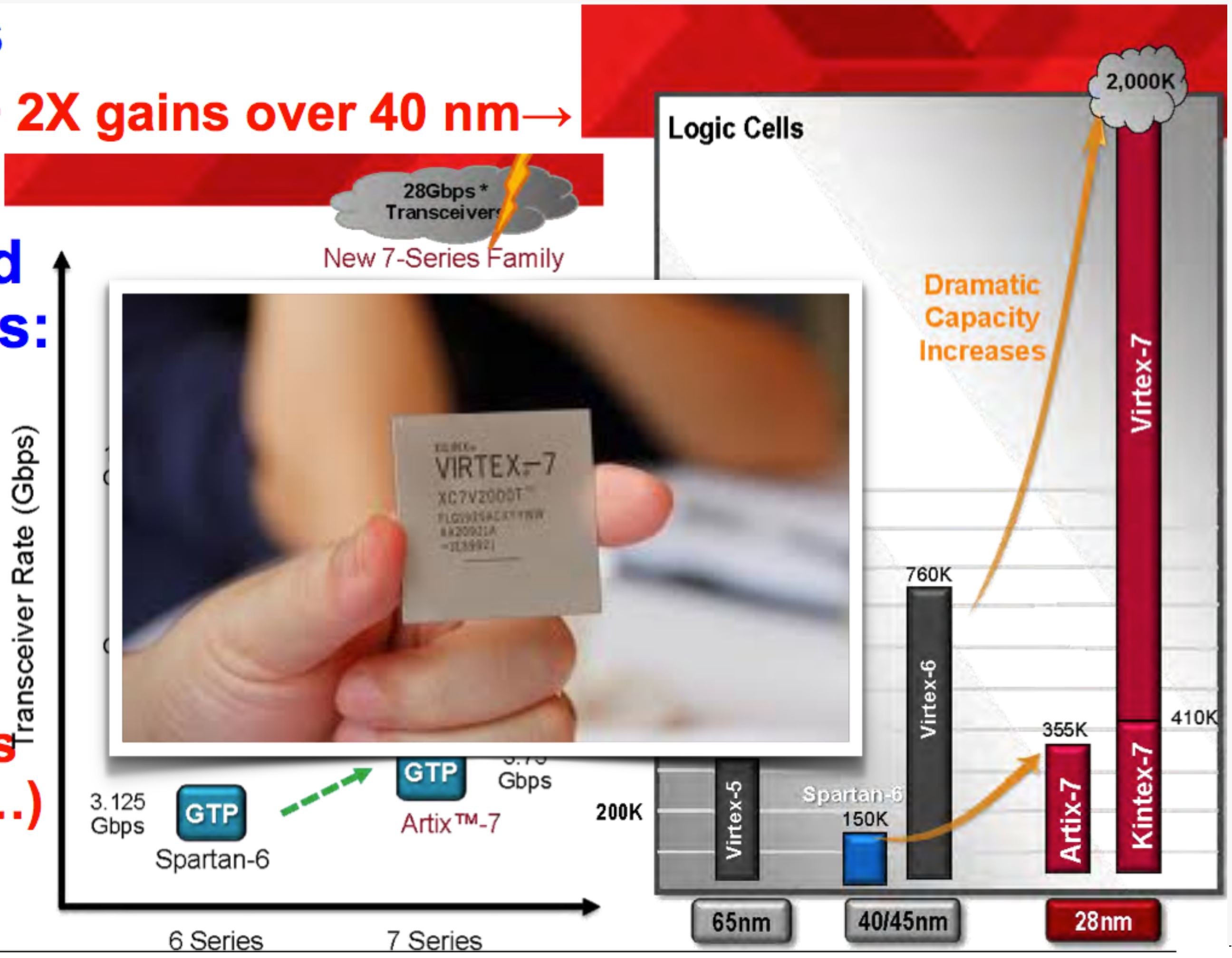
Progress in FPGAs

Logic Cells

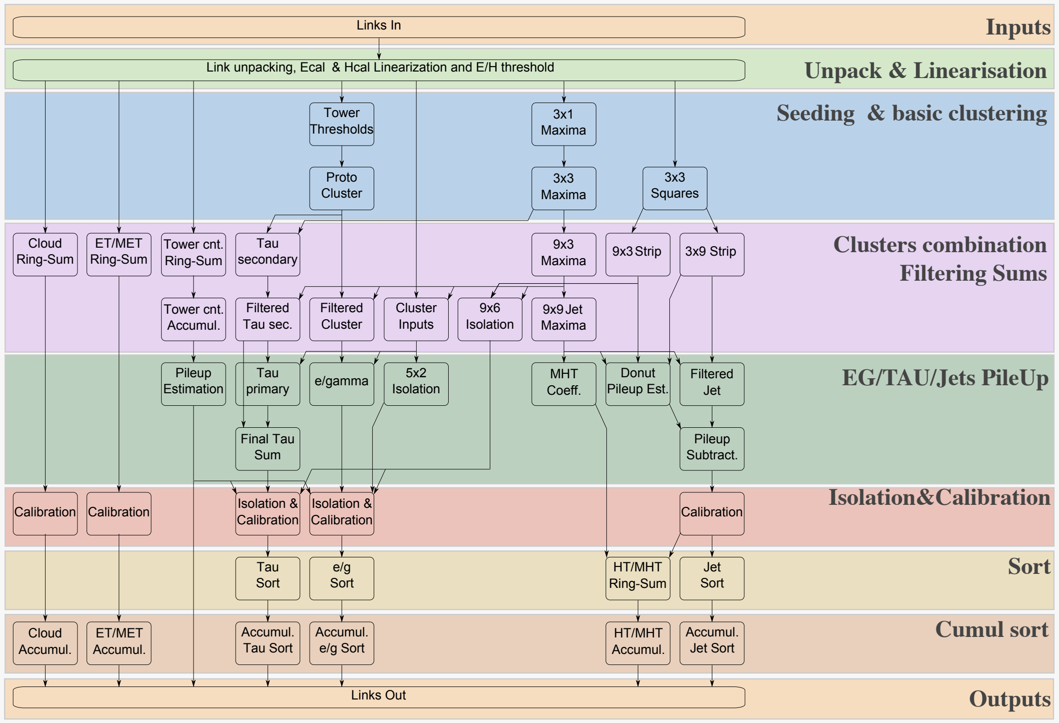
- ➢**28 nm: > 2X gains over 40 nm→**

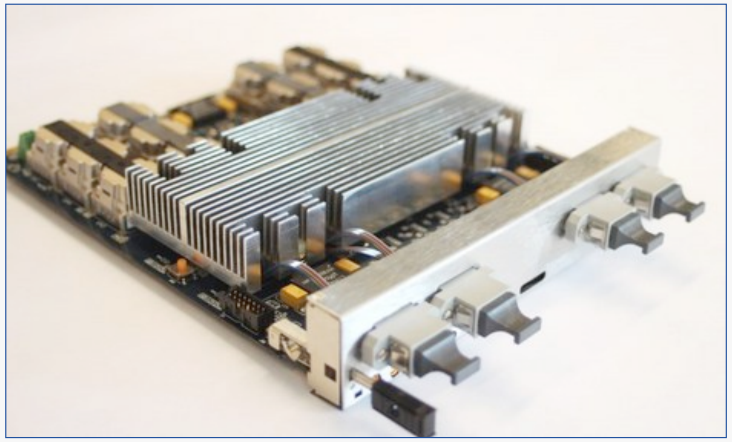
On-Chip High Speed Serial Links:

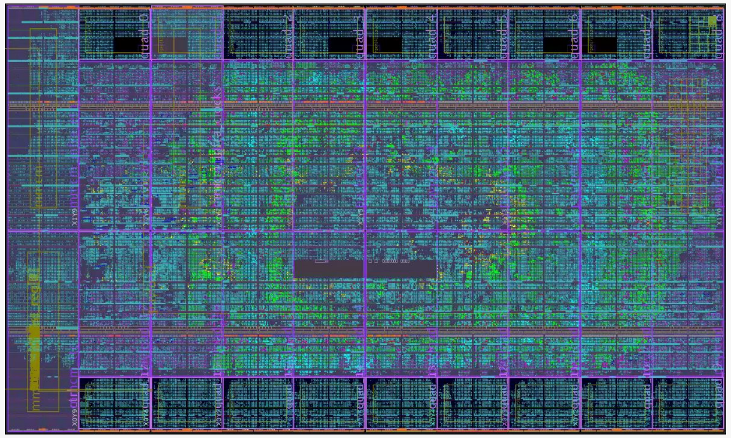
- ➢**Connect to new compact high density optical connectors (SNAP-12…)**

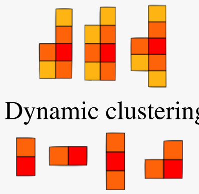


Example: CMS Calorimeter trigger in FPGA

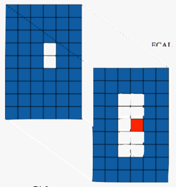




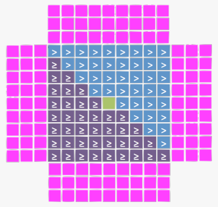




Dynamic clustering

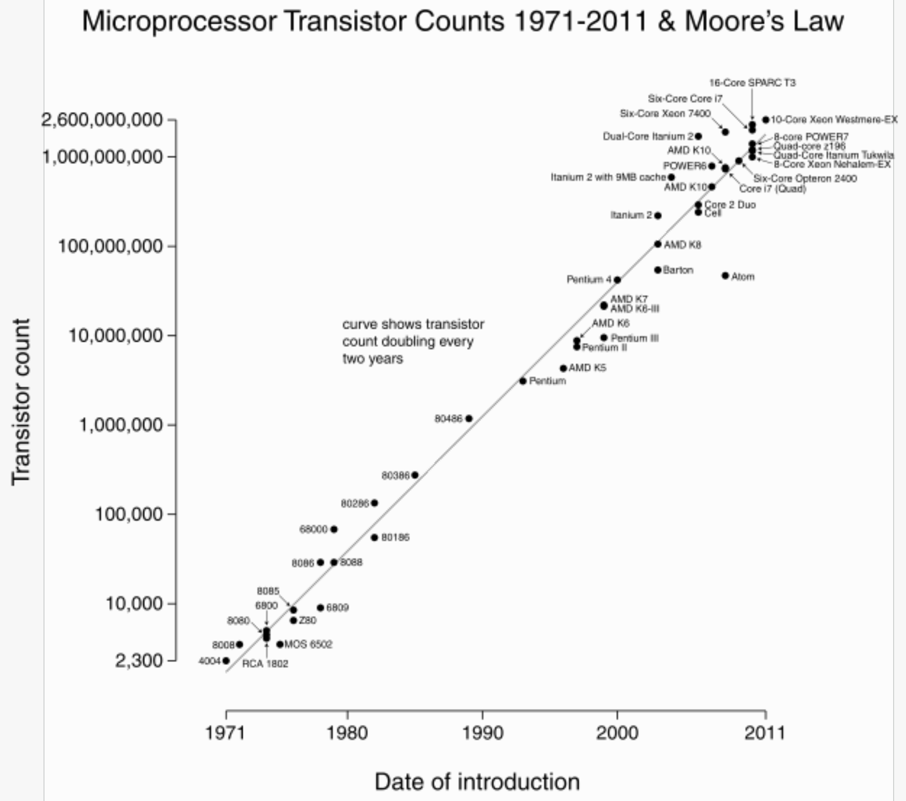


Shape veto,
H/E, isolation, calibration



jet building with
pileup subtraction

Trends in processing technologies



Moore's Law: the number of transistors that can be placed inexpensively on an integrated circuit doubles approximately every two years (Wikipedia)

Demand of **higher complexity** → **higher chip density** → **smaller structure size** (for transistors and memory size):

- Nvidia **GPUs**: 3.5 B transistors
- Virtex-7 **FPGA**: 6.8 B transistors
- **14 nm** CPUs/FPGAs in **2014**

32 nm → 10 nm

2010 **2017**

For FPGAs, smaller feature size means **higher-speed and/or less power consumption**

Multi-core evolution

- Accelerated processing **GPU+CPU**

Moore's law expected to hold at least until 2020, for FPGAs and co-processors as well

Market driven by cost effective components for Smartphones, Phablets, Tablets, Ultrabooks, Notebooks ....

Data communication

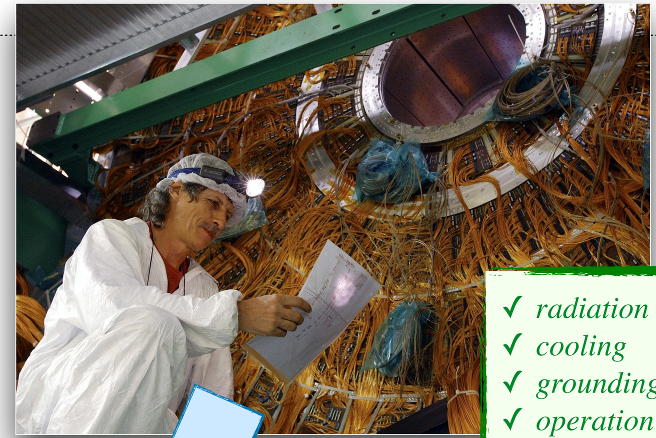
Processing technology has now reached very high densities and speeds

High-speed serial links, electrical and optical

- Low cost and low-power LVDS links, @400 Mbit/s (up to 10 m)
- Optical GHz-links for longer distances (up to 100 m)

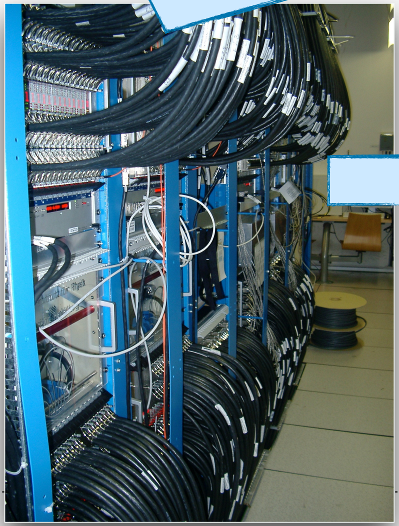
High density backplanes for data exchanges within crates

On-detector

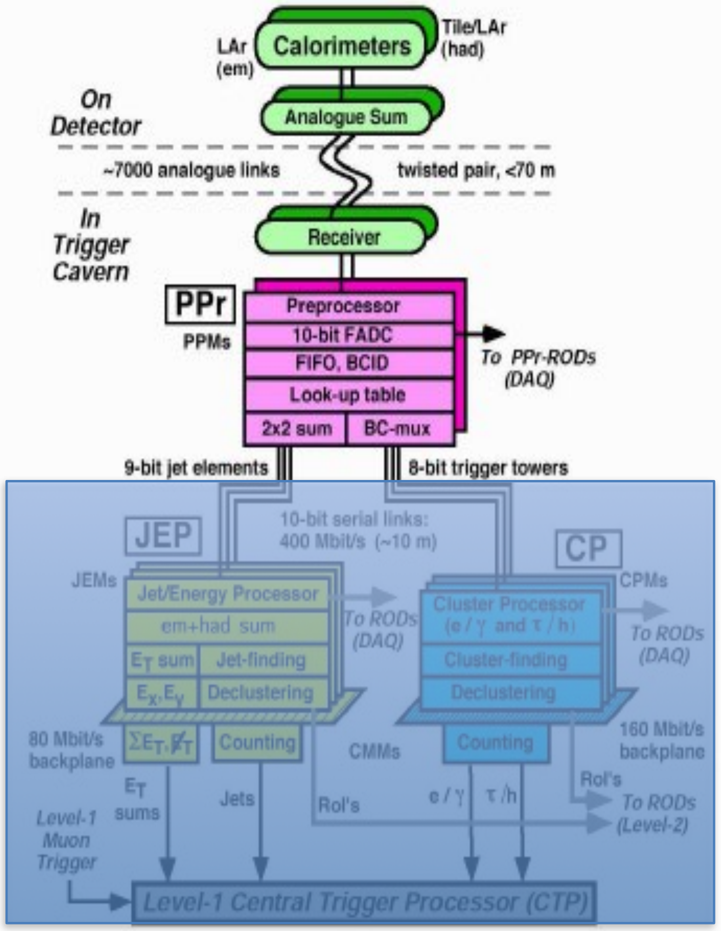


- ✓ *radiation tolerance*
- ✓ *cooling*
- ✓ *grounding*
- ✓ *operation in magnetic field*
- ✓ *very restricted access*

Off-detector



Example : ATLAS calorimeter trigger



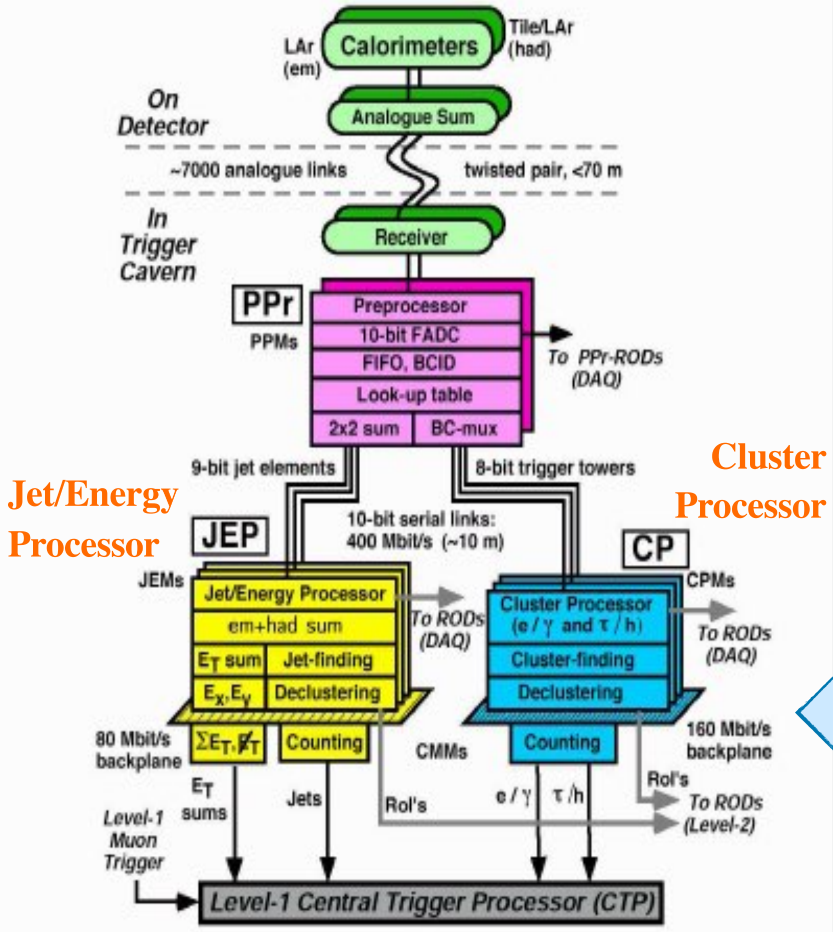
On-detector

- Sum of analog signals from cells to form towers

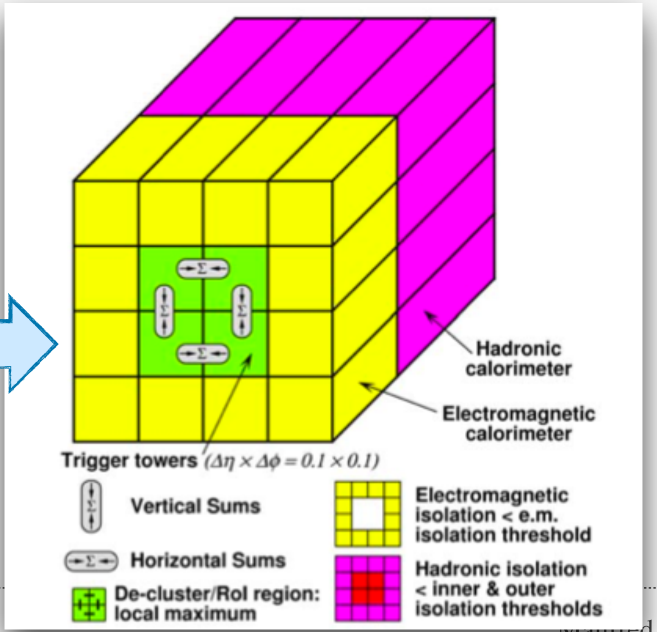
Off-detector - L1 Trigger

- Pre-processor board
- ADCs with 10-bit resolution
- ASICs to perform the trigger algorithm
  - Assign energy (ET) via Look-Up tables
  - Apply threshold on ET
  - Peak-finder algorithm to assign the BC

Example : ATLAS calorimeter trigger



- Implemented in FPGAs, the parameters of the algorithms can be easily changed
- Total of 5000 digital links connect PPr to JEP and CP, 400 Mb/s


Hadronic calorimeter
Electromagnetic calorimeter
Trigger towers (Δη × Δφ = 0.1 × 0.1)
Vertical Sums
Horizontal Sums
De-cluster/RoI region: local maximum
Electromagnetic isolation < e.m. isolation threshold
Hadronic isolation < inner & outer isolation thresholds

High-level trigger technologies

Can we use the offline algorithms online?

High Level Trigger Architecture

	Levels	L1 rate	Event size	Readout bandwidth	HLT rate
LEP	2/3	1 kHz	100 kB	a few 100 kB/s	~5 Hz
ATLAS	2/3	**100 kHz** **(L2: 10 kHz)**	1.5 MB	30 GB/s (incremental Ev. Building)	~1 kHz
CMS	2	**100 kHz**	1.5 MB	200 GB/s	~1 kHz

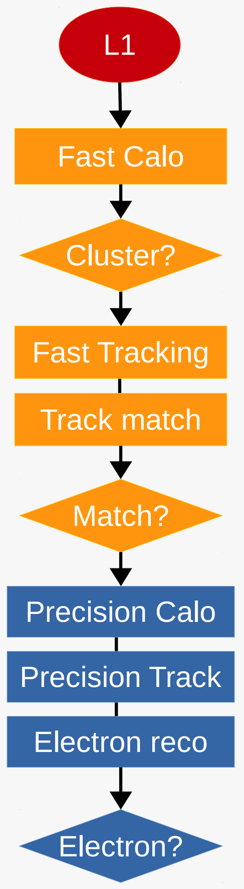
LEP: 40 Mbyte/s VME bus sufficient for bandwidth needs

LHC: cutting-edge processors, high-speed network interfaces, high-speed optical links

Different approaches possible

- Network-based event building (LHC example: **CMS**)
- Seeded reconstruction (LHC example: **ATLAS**)

HLT design principles



Offline reconstruction too slow to be used directly

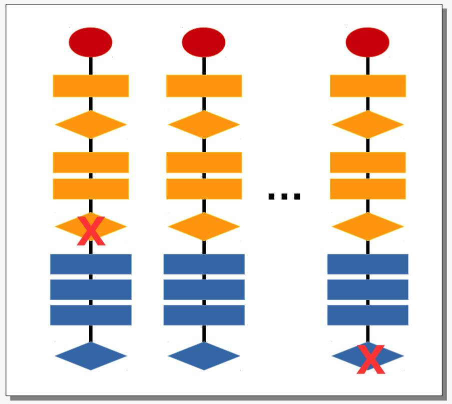
- Takes >10s per event but HLT usually needs << 1s L1

Requires **step-wise** processing with **early rejection**

1. Fast reconstruction
   - Trigger-specific or special configurations of offline algorithms
   - L1-guided regional reconstruction
2. Precision reconstruction
   - Offline (or very close to) algorithms
   - Full detector data available

Stop processing as soon as one step fails

Event accepted if any of the trigger passes



Manfred Jeitler

HLT design principles

Early rejection

- Reduce data and resources (CPU, memory….)

Event-level parallelism

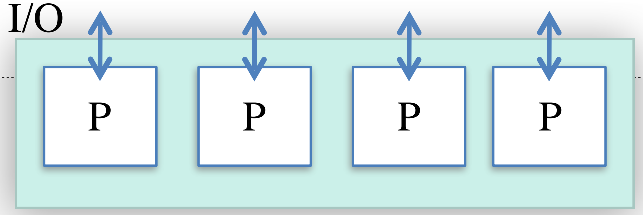
- Process more events in parallel
- Multi-processing multi-threading

Algorithm-level parallelism

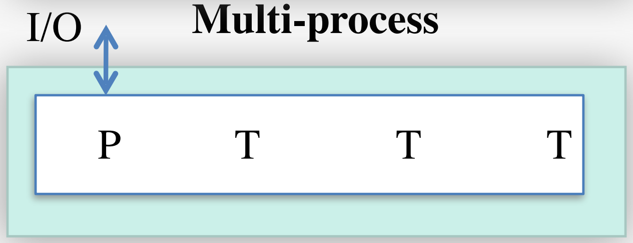
- multi-threading
- **GPUs** effective whenever large amount of data can be processed concurrently

Algorithms developed and optimized offline

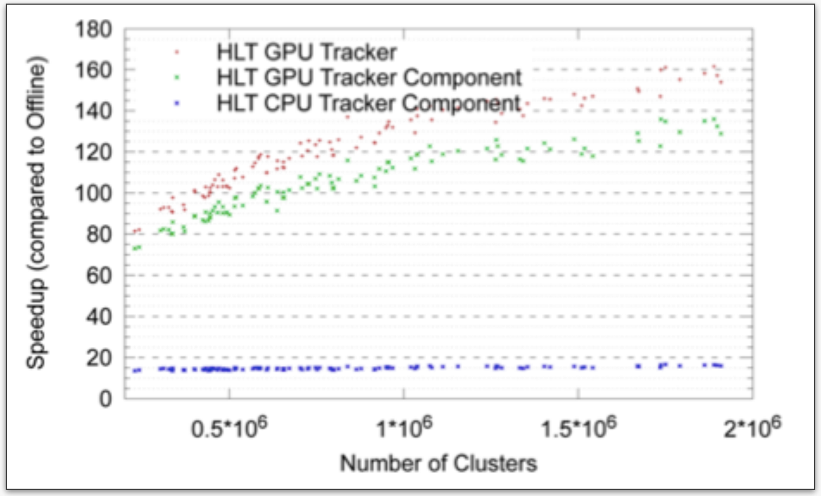
Common HLT-reconstruction software framework **reduces maintenace** and **increases reliability**



Multi-process



Multi-threading



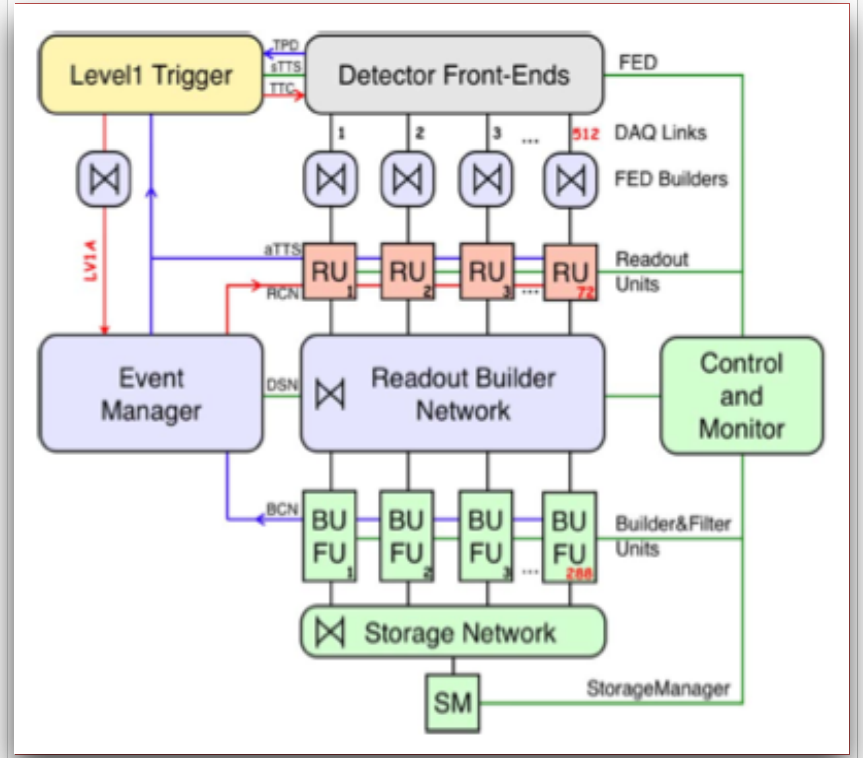
Now …
try it out in the lab!

Now time to build your own trigger system!

- Trigger and DAQ systems use many new technologies —> contact with industry
- Microelectronics, networking, computing expertise are required to build an efficient trigger system
  - But always in close contact with the physics measurements we want to study
- Here were presented some general problems, that will be discussed in detail during other lessons

Profit of this school to understand these connections!!

Network-based HLT: CMS

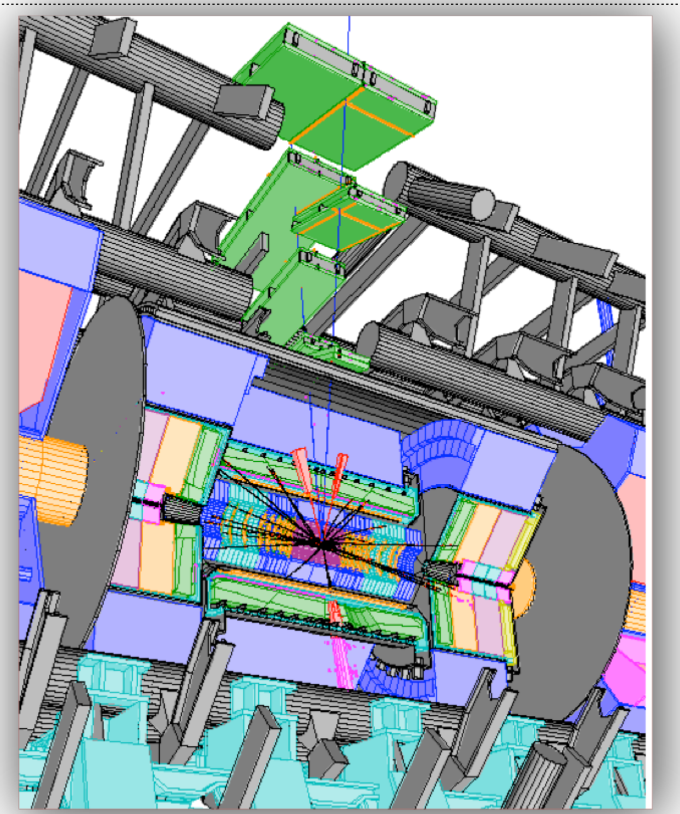


FU = several CPU cores = several filtering processes executed in parallel

Data from the readout system (RU) are transferred to the filters (FU) through a builder network

Each filter unit processes only a fraction of the events

Seeded reconstruction HLT: ATLAS



Typically, there are less than 2 RoIs per event accepted by LVL1

Level-2 uses the information seeded by level-1 trigger

- Only the data coming from the region indicated by the level-1 is processed, called Region-of-Interest (RoI)
- The resulting total amount of RoI data is minimal: a few % of the Level-1 throughput
- Level-2 can use the full granularity information of only a part of the detector

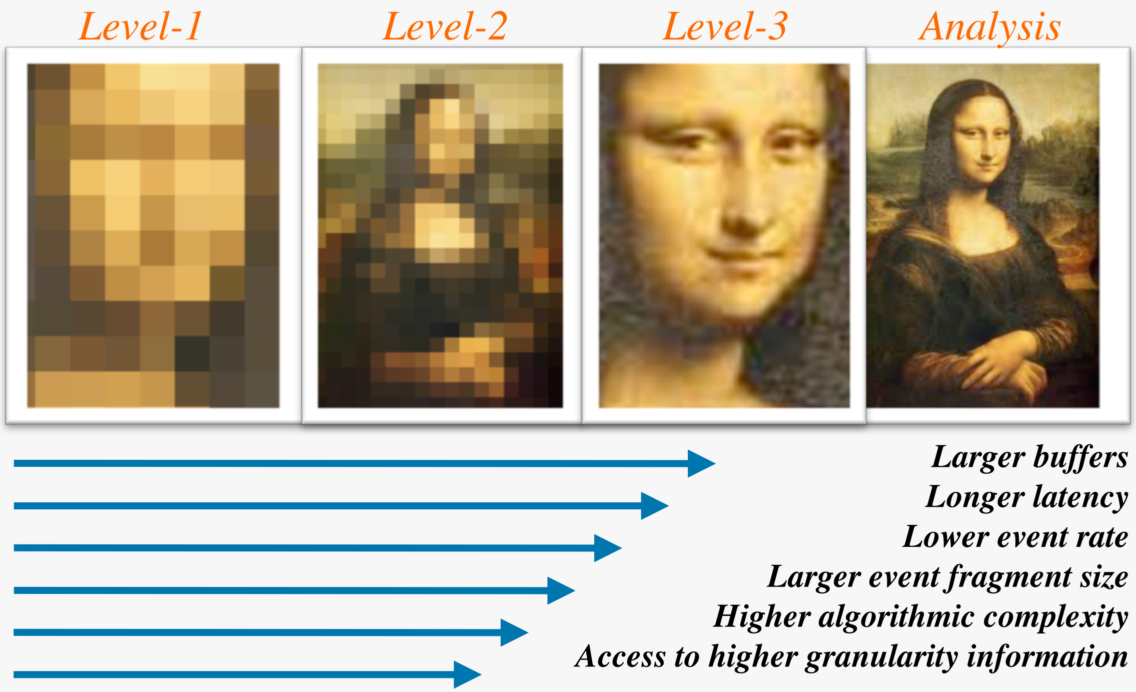
No need for large bandwidth

Complicate mechanism to serve the data selectively to the L2 processing

Multi-level triggers

Adopted in large experiments

- More and more complex algorithms are applied on lower and lower data rates
  - First level with short latency, working at higher rates
  - Higher levels apply further rejection, with longer latency (more complex algorithms)



Level-1
Level-2
Level-3
Analysis
Larger buffers
Longer latency
Lower event rate
Larger event fragment size
Higher algorithmic complexity
Access to higher granularity information

LHC experiments @ Run1

Exp	N. of Levels
ATLAS	3
CMS	2
LHCB	3
ALICE	4

Efficiency for the desired physics must be kept high at all levels, since rejected events are lost for ever