Electronics, Trigger and Data Acquisition

lecture 2 of 3

Summer Student Programme 2019, CERN

July 11, 2019

Roberto Ferrari
Instituto Nazionale di Fisica Nucleare

roberto.ferrari@pv.infn.it

What's the problem we would like to solve ?

Have some kind of camera and want to take as many pictures as possible …

sometimes the camera is very very big

When ?

Every time something interesting happens …

usually unpredictable, i.e. uncorrelated with anything

How often may something interesting happen?

Depends …

sometimes very very often

a T/DAQ System: your Camera

Sensor (CCD+elx): Detector and data acquisition system

Memory card: Temporary storage (cache)

Display (LCD): Online and offline monitoring system

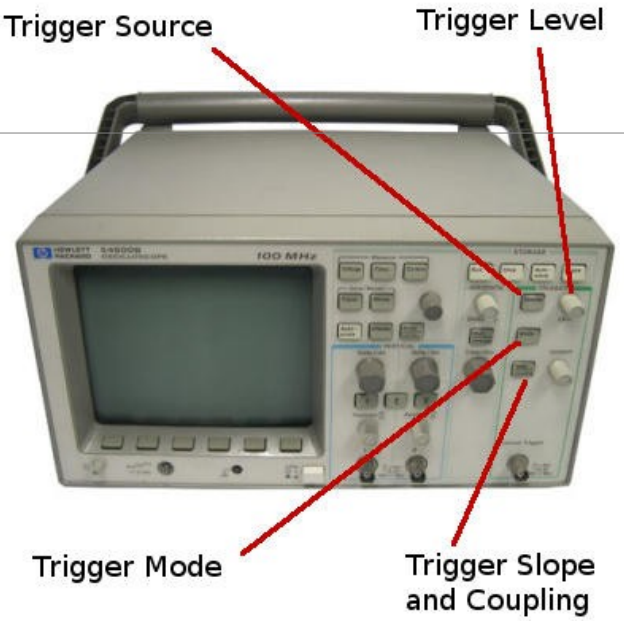
Push-button: Trigger

Trigger

did something interesting arrive ?

What does “Trigger” mean?

- *Prompt signal* starting data-acquisition processes *[“please, look at that”]*
- Keywords: *simple, rapid, selective* (as much as possible!)
  - selective = efficient for “signal” & resistant to “background”
- Actual parameters strongly dependent on operating conditions
  - in multi-level trigger system, “next” level way slower and more complex than current one



Oscilloscope trigger does exactly this:

informs the instrument to start signal acquisition and visualisation

back to square one

Do we really need a trigger ?

not obvious … triggerless DAQ systems do exist

even in HEP, e.g.:

- LHCb upgrade 40 MHz readout
- DUNE LAr TPC 2 MHz readout

but triggering may be crucial …

https://en.wikipedia.org/wiki/Coincidence_circuit :

Walther Bothe (1924-1929):
offline ⟶ online coincidence (logic AND) of 2 signals

Bruno Rossi (Nature, 1930):
"Method of Registering Multiple Simultaneous Impulses of Several Geiger Counters"
→ online coincidence of 3 signals (scalable)!

how trigger was born

https://en.wikipedia.org/wiki/Coincidence_circuit :

"Rossi coincidence circuit was rapidly adopted by experimenters around the world. It was the first practical AND circuit, precursor of the AND logic circuits of electronic computers"

Geiger-Muller counters

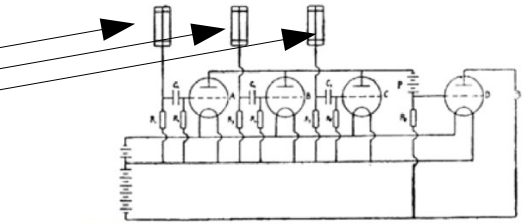


Fig. 17 – Il circuito di Rossi per rivelare coincidenze di raggi cosmici che arrivano sui contatori Geiger (i rettangoli in alto dello schema)[19].

Rossi's circuit: coincidence of signals of 3 Geiger-Muller counters

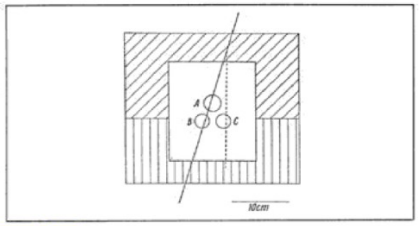
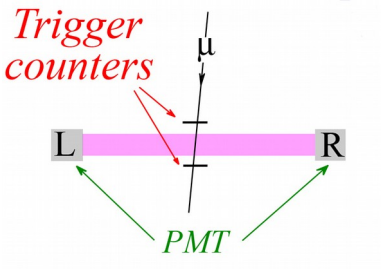


Fig. 18 – L'uso del circuito di Rossi per rivelare una coincidenza tripla che, nella disposizione in figura dei tre contatori, mostra la produzione di una radiazione secondaria (linea tratteggiata) da parte della radiazione primaria (linea continua)[20].

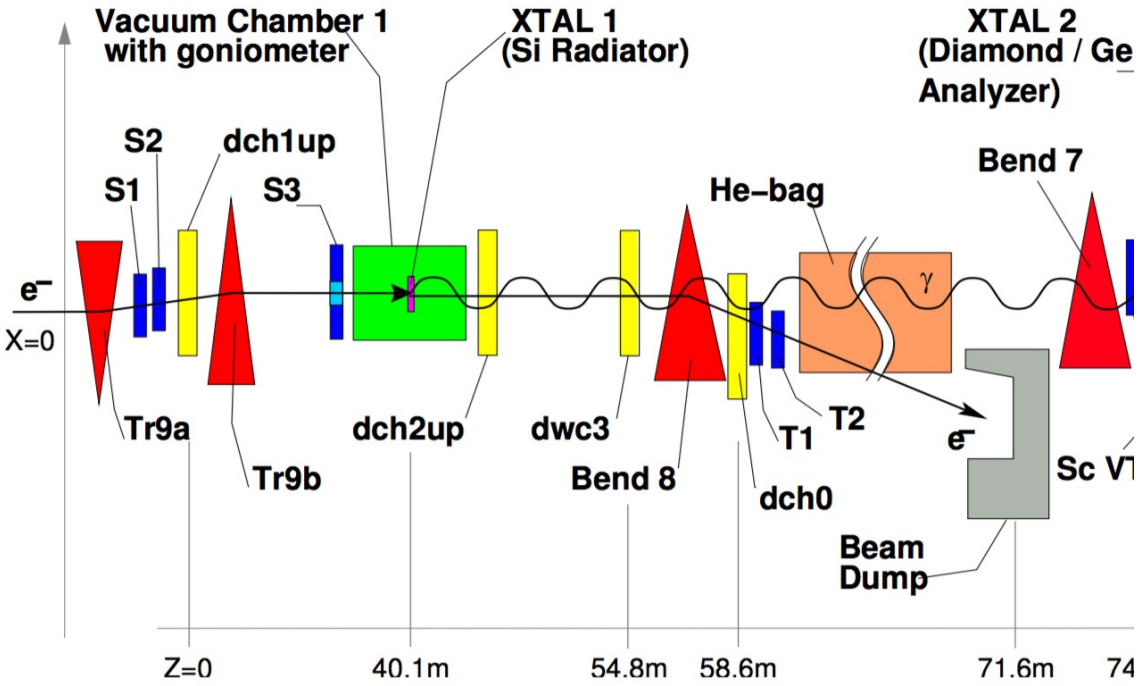


simplest case: 2-signal coincidence

a simple trigger system

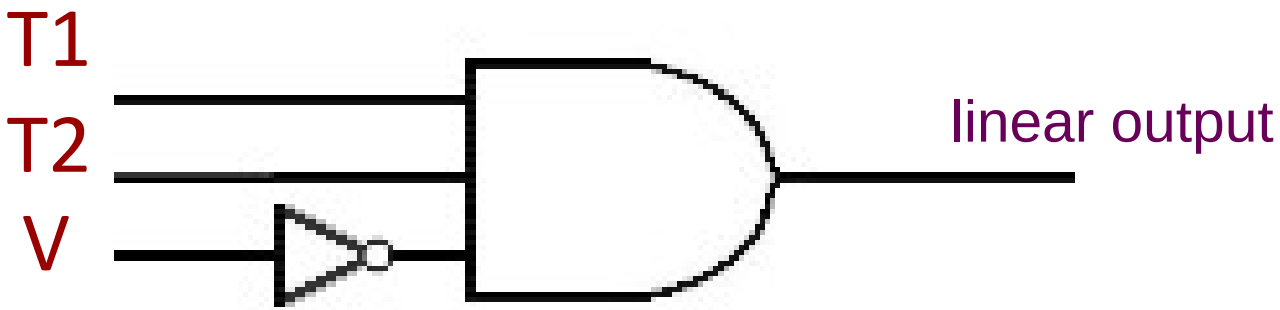
$$T = s1 \cdot s2 \cdot \overline{s3}$$

Veto (anti-coincidence)



NA59 experiment setup

… has anyway issues !

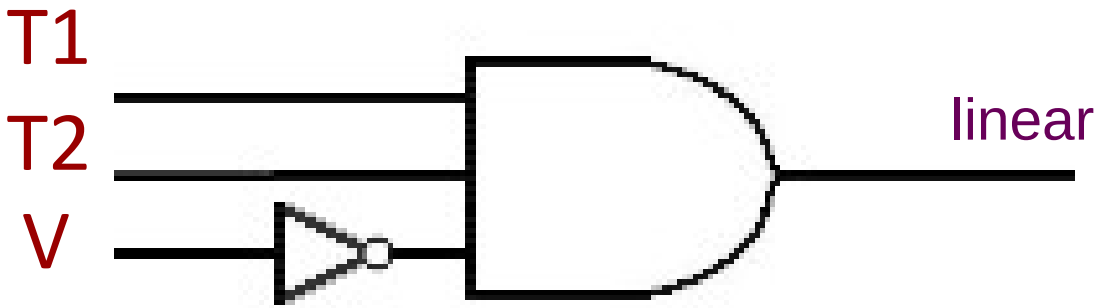


(anti-)coincidence with veto

→ simple, clear !

really doing what you think/need ?

(anti-)coincidence with veto



flawed !

output signal may:

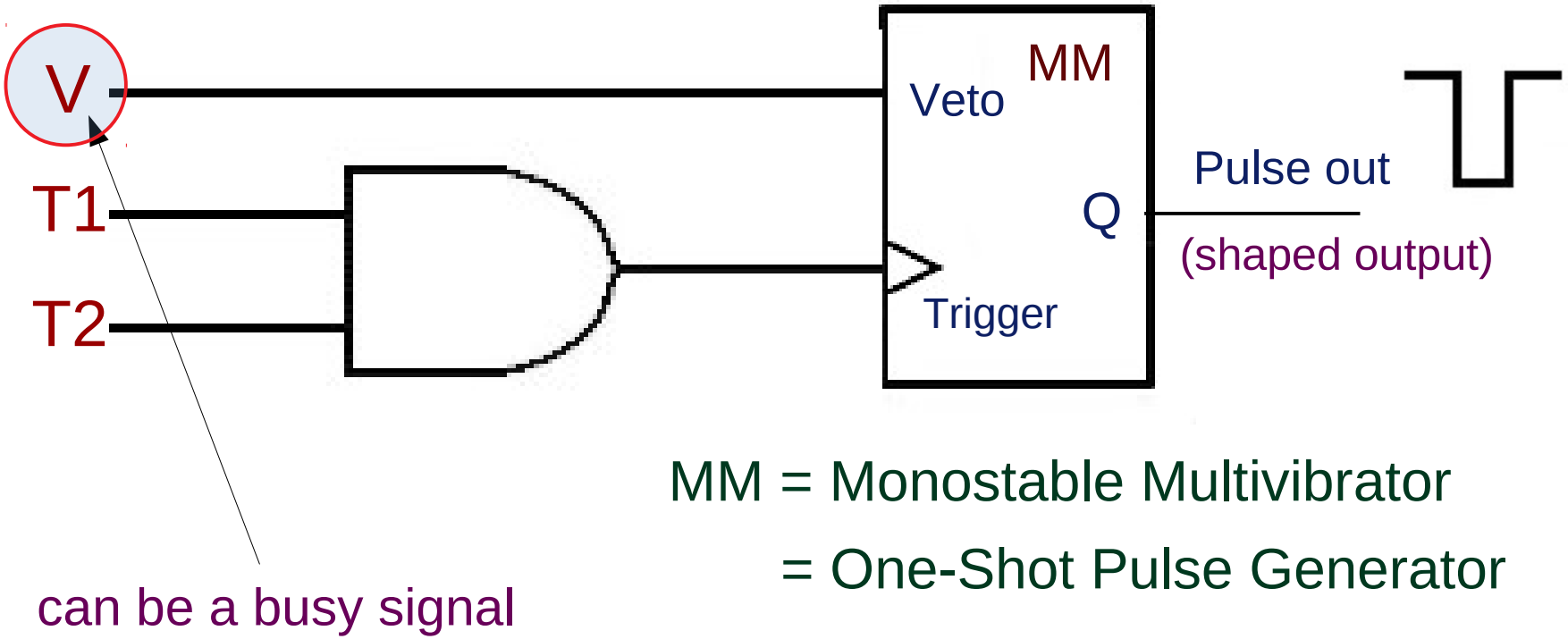
a) jitter

b) fluctuate in duration

(even better do both)

because of relative timing of T1, T2, V

(anti-)coincidence with veto



MM = Monostable Multivibrator

= One-Shot Pulse Generator

much better !

first lesson(s)

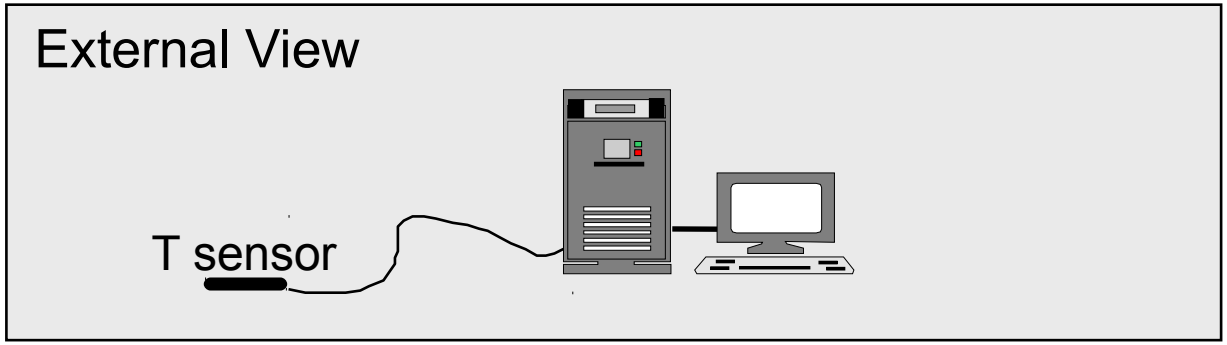
trigger signal:

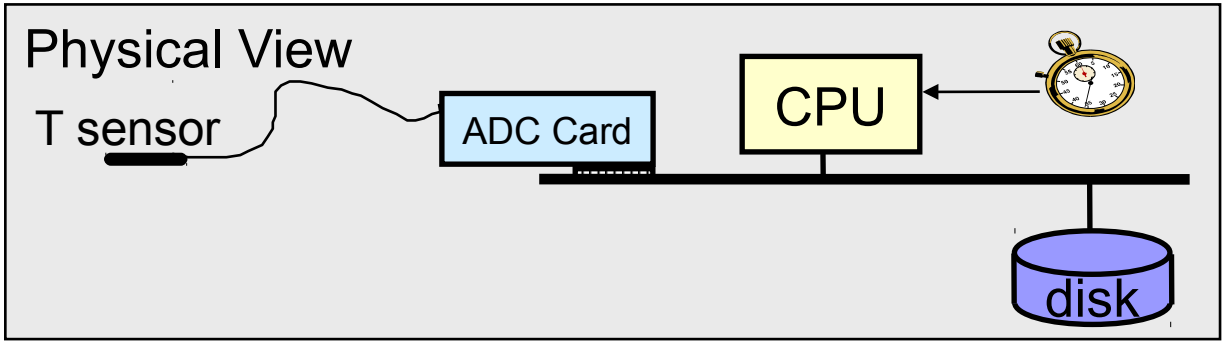
1) should be formed

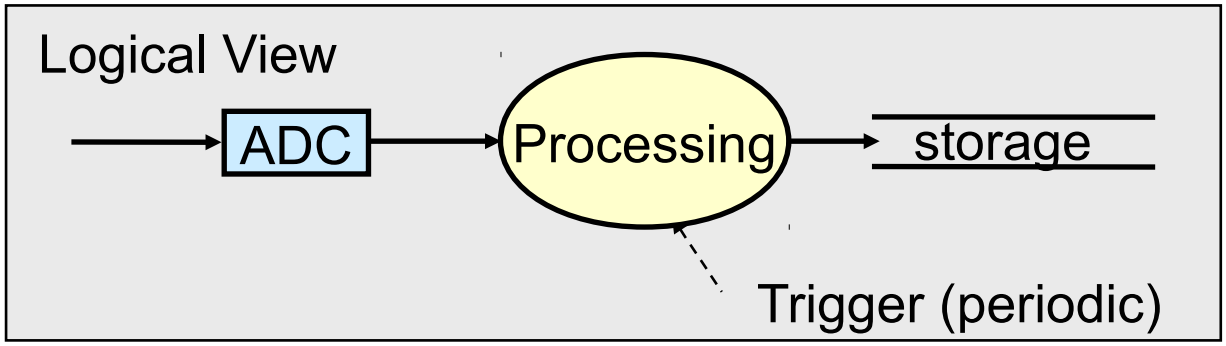
→ pulse with predefined duration

2) veto/busy should block pulse generation

Basic DAQ: Synchronous Trigger (1)

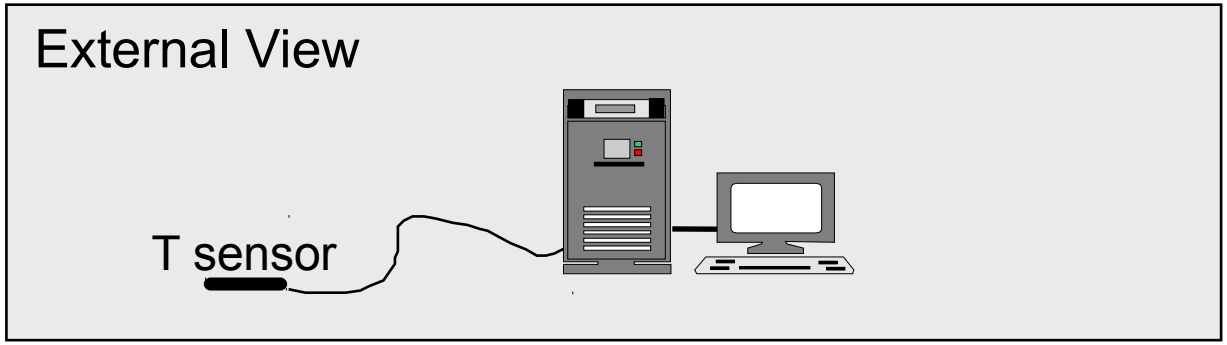


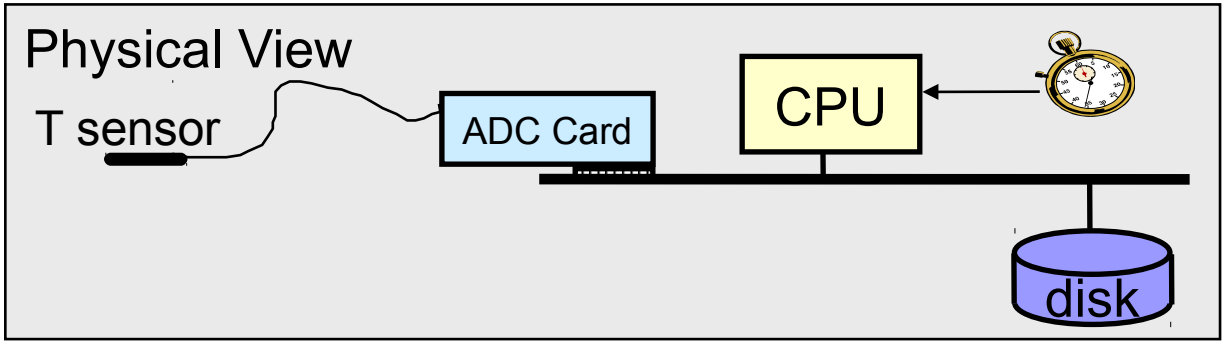


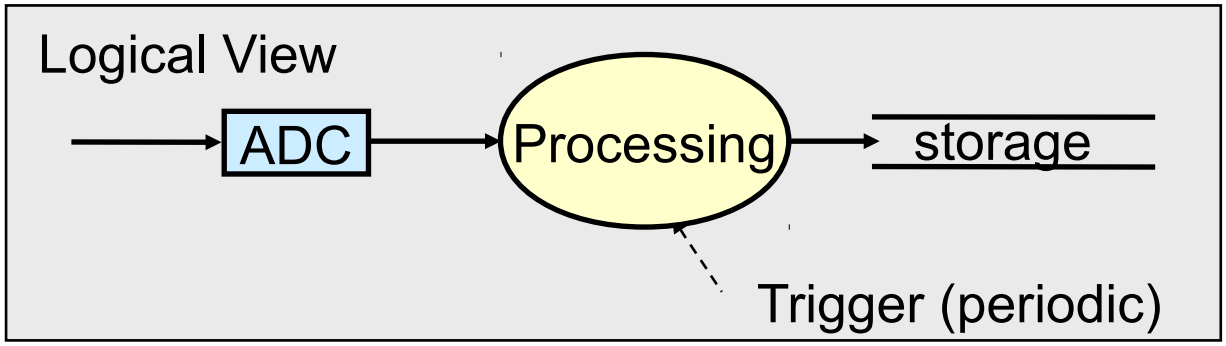


- Measure temperature at fixed frequency
- ADC performs analog-to-digital conversion
  - our “front-end electronics”
- CPU does readout and processing

Basic DAQ: Synchronous Trigger (2)

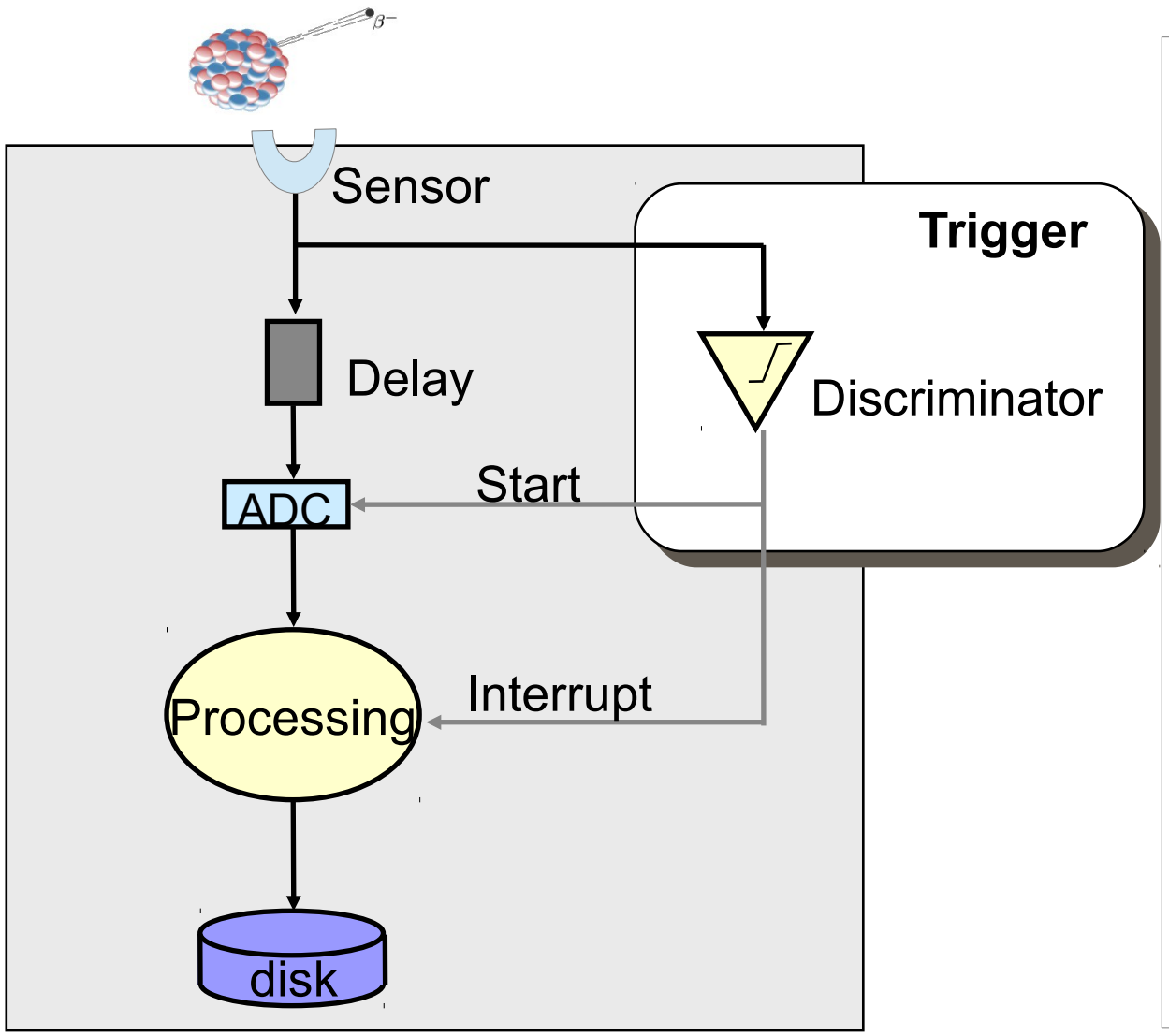






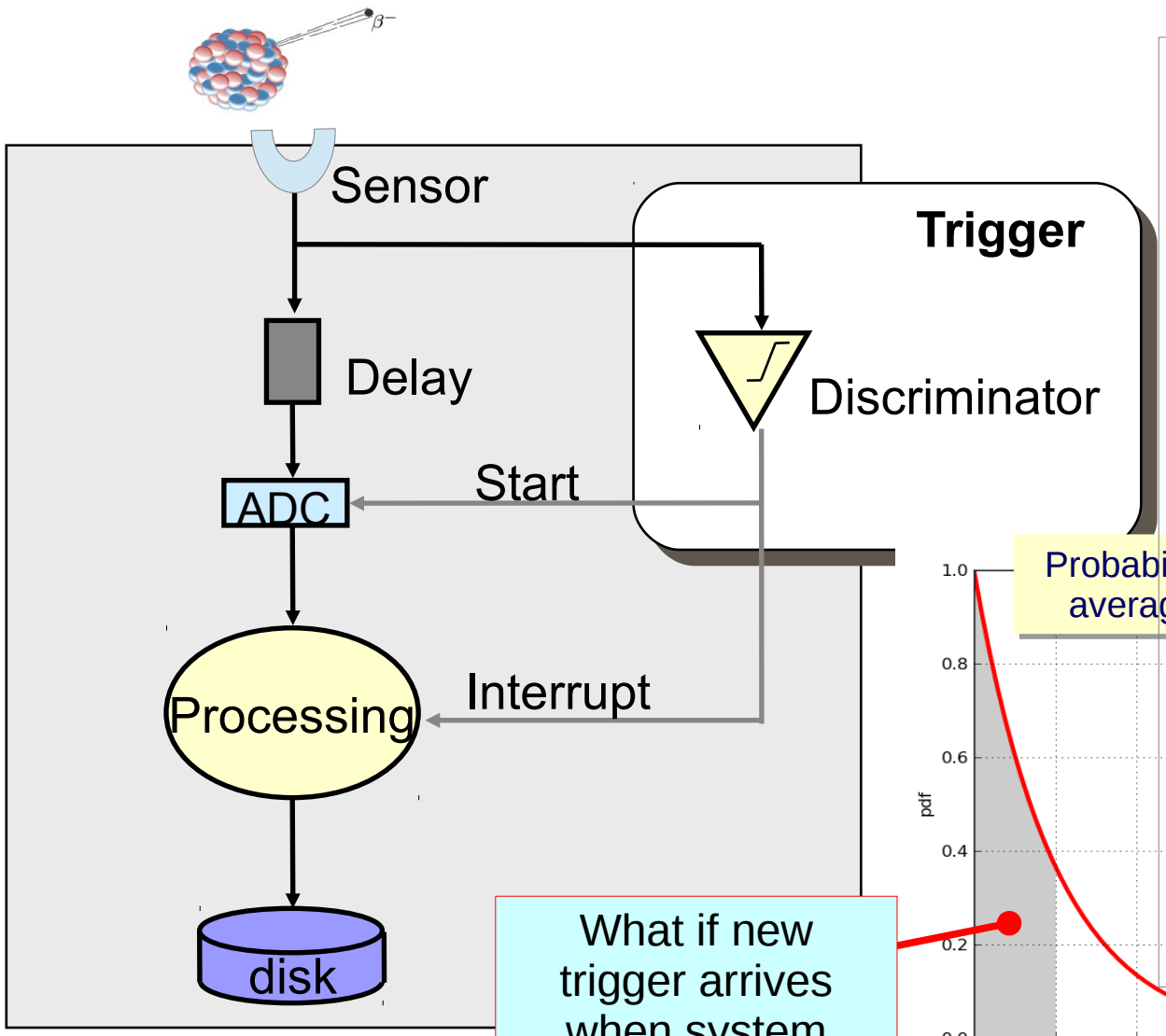
- Fully sequential system
- System limited by single-event processing time
- If τ ~ 1 ms for
  ADC conversion
  +CPU processing
  +storage
  → can sustain up to ~ 1 kHz of ***periodic (synchronous) trigger*** rate

Basic DAQ: Physics Trigger



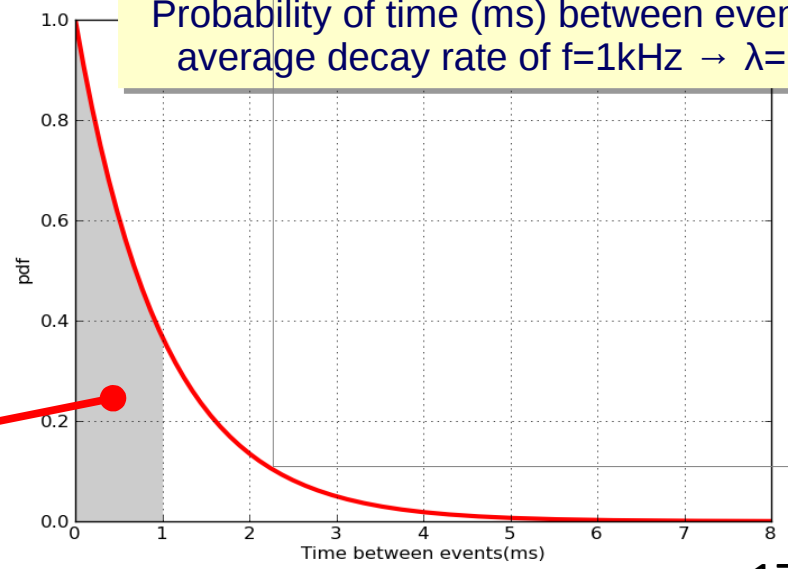
- Measure β decay properties
- Asynchronous and unpredictable events
  - need **physics** trigger
- Delay compensates for trigger latency
  - time to reach decision
- When system busy (=not ready to handle triggers) → dead time

Basic DAQ: Physics Trigger



- Measure β decay properties
- Stochastic (i.e. fully uncorrelated process)
  - fluctuations

Probability of time (ms) between events for average decay rate of f=1kHz → λ=1ms



What if new trigger arrives when system busy?

Basic DAQ: don't loose any event ?

a) Retriggerable DAQ system: any new trigger accepted, each causing processing (dead-time) restart, regardless of DAQ state

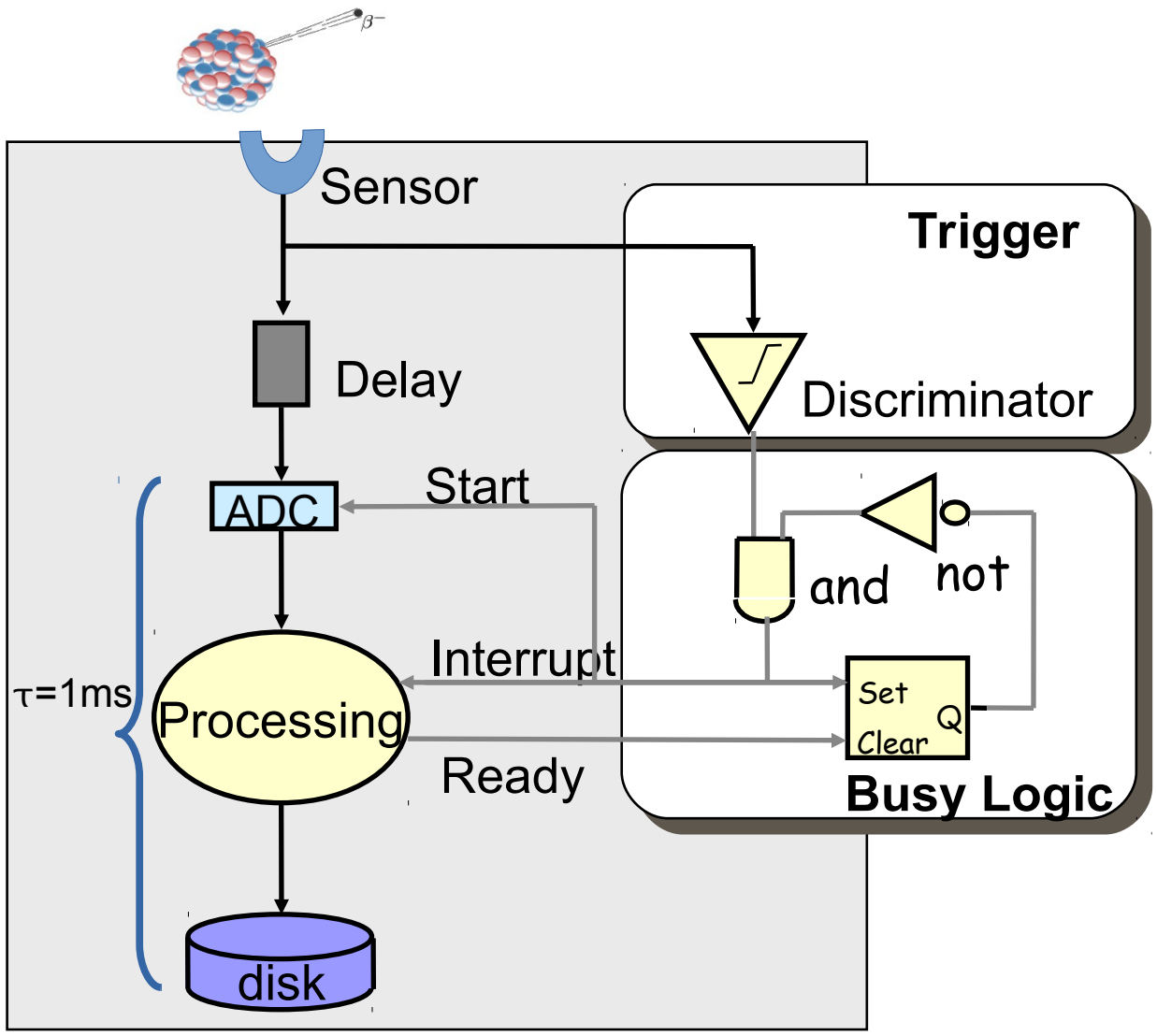
⇒ paralysable DAQ

b) Non retriggerable : dead time blocking trigger (no new trigger until dead time elapsed)

⇒ non-paralysable DAQ

- going to describe non-paralysable DAQ systems only

Basic DAQ: Real Trigger & Busy



- Busy logic blocks triggers while processing
- Which (average) DAQ rate can be achieved ?

  Reminder: τ = 1 ms sufficient for 1 kHz synchronous trigger

Dead Time

how many events may I loose ?

DAQ Dead Time & Efficiency (1)

Being:

τ = <DAQ dead time>, f = <signal rate>, ν = <acquisition rate>

$\nu \cdot \tau$ = total DAQ dead time $\Rightarrow$ $(1-\nu \cdot \tau)$ = total DAQ available time

$\rightarrow$ $f \cdot (1-\nu \cdot \tau) = \nu \Rightarrow \nu = f/(1+f \cdot \tau) < f, 1/\tau$

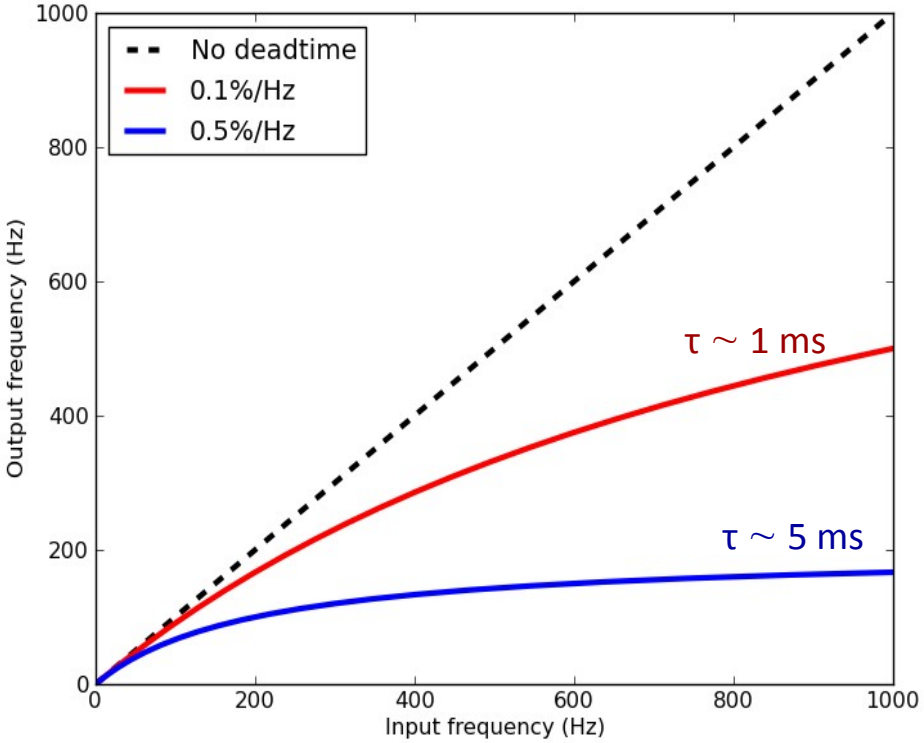
Efficiency $\varepsilon = \nu/f = 1/(1+f \cdot \tau) < 100\%$

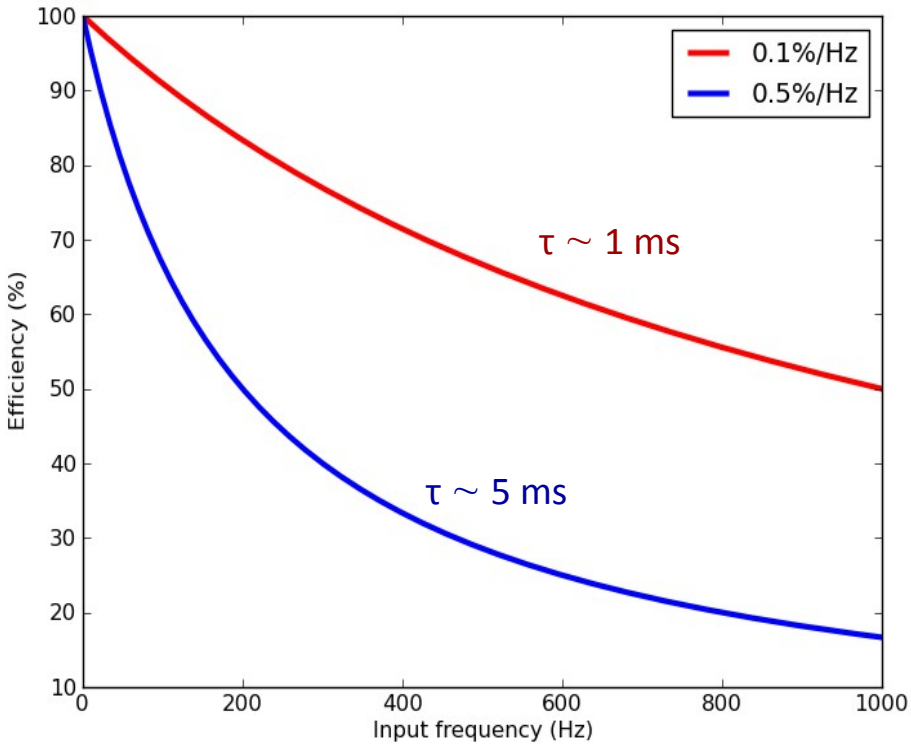
Dead time $(1-\varepsilon) = f \cdot \tau/(1+f \cdot \tau) \Rightarrow f \cdot \tau < (1-\varepsilon) < 1$

- Max acquisition speed ($f \rightarrow \infty$) $\nu \rightarrow 1/\tau$
- Due to stochastic fluctuations, efficiency always < 100%

  if τ = 1 ms, f = 1 kHz $\Rightarrow$ ν = 500 Hz, ε = 50%

DAQ Dead Time & Efficiency (2)



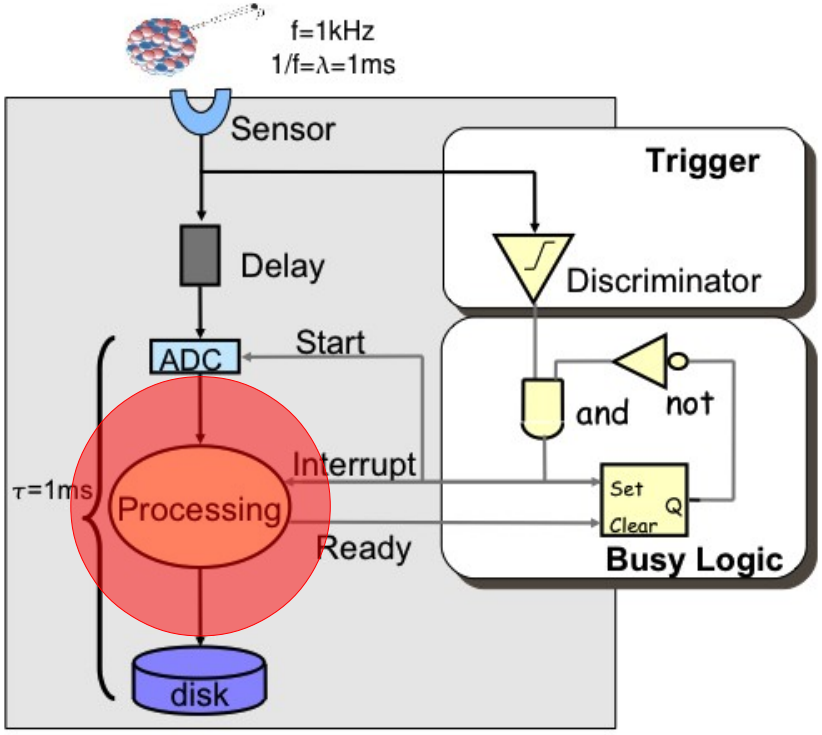


- Want: $\varepsilon \sim 99\% \Rightarrow f \cdot \tau \sim 0.01 \Rightarrow 1/\tau \sim 100 \cdot f$

  $f = 1$ kHz $\Rightarrow$ $1/\tau =$ 100 kHz !
- To cope with input rate fluctuations, we have to over-design our DAQ system by a factor 100. Very inconvenient ! How to mitigate this effect ?

Dead Time → de-randomise

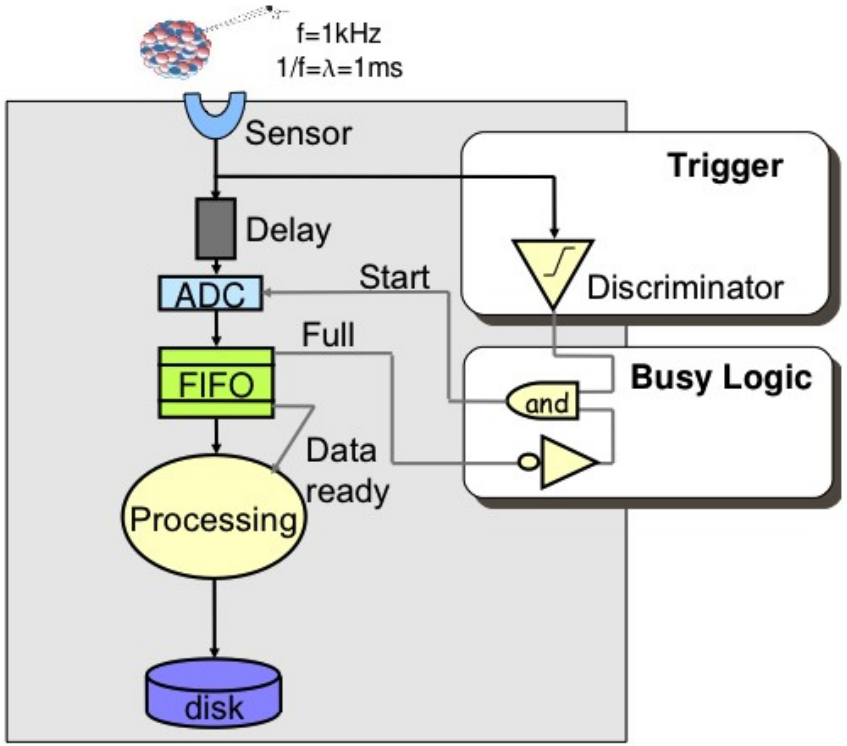
- Processing → bottleneck



Dead time ~ $(1+x)^{-1}$ ~ 50%

for $x = 1/(f \cdot \tau) \sim 1$

- Buffering allows to decouple problems

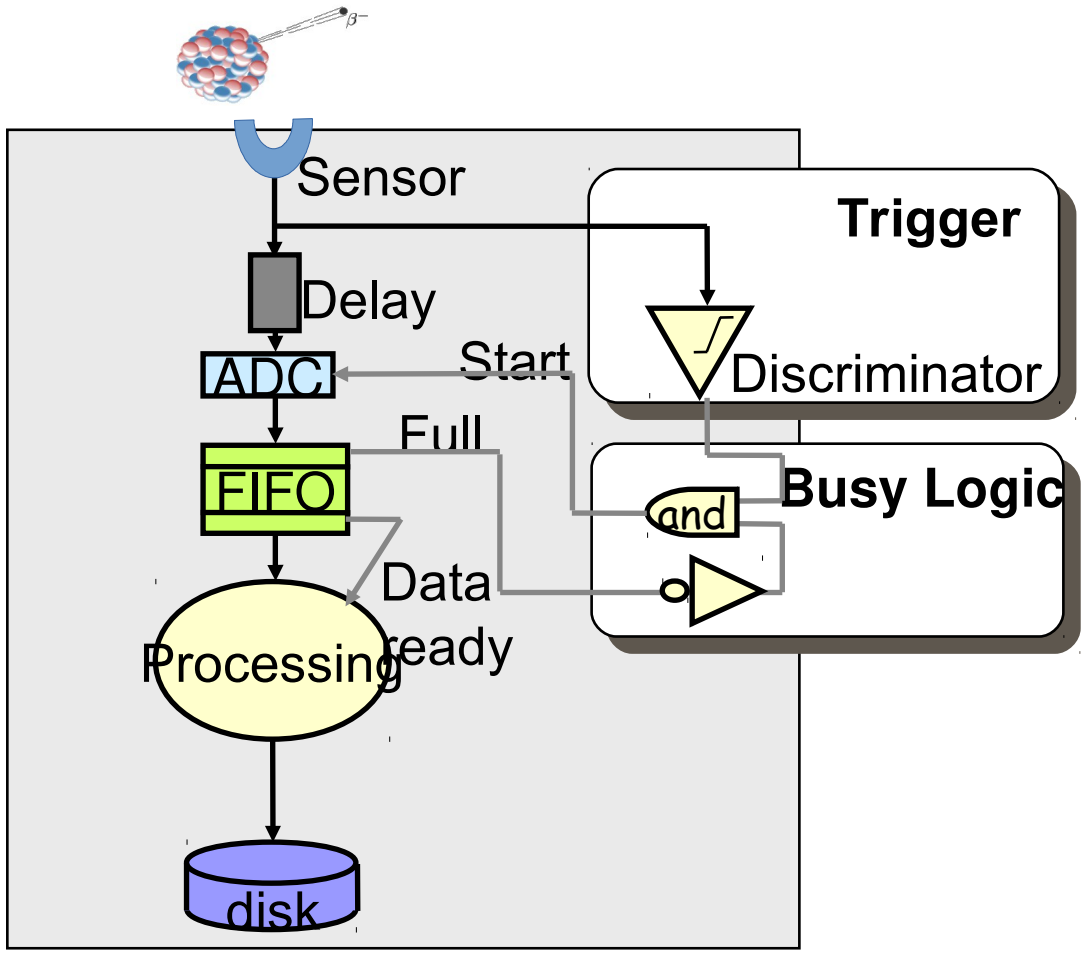


Dead time ~ 1/(N+1)

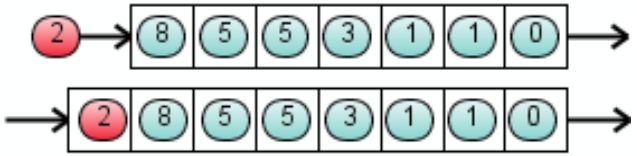
[N = buffer depth]

for $(f \cdot \tau) \sim 1$

Basic DAQ: De-Randomisation



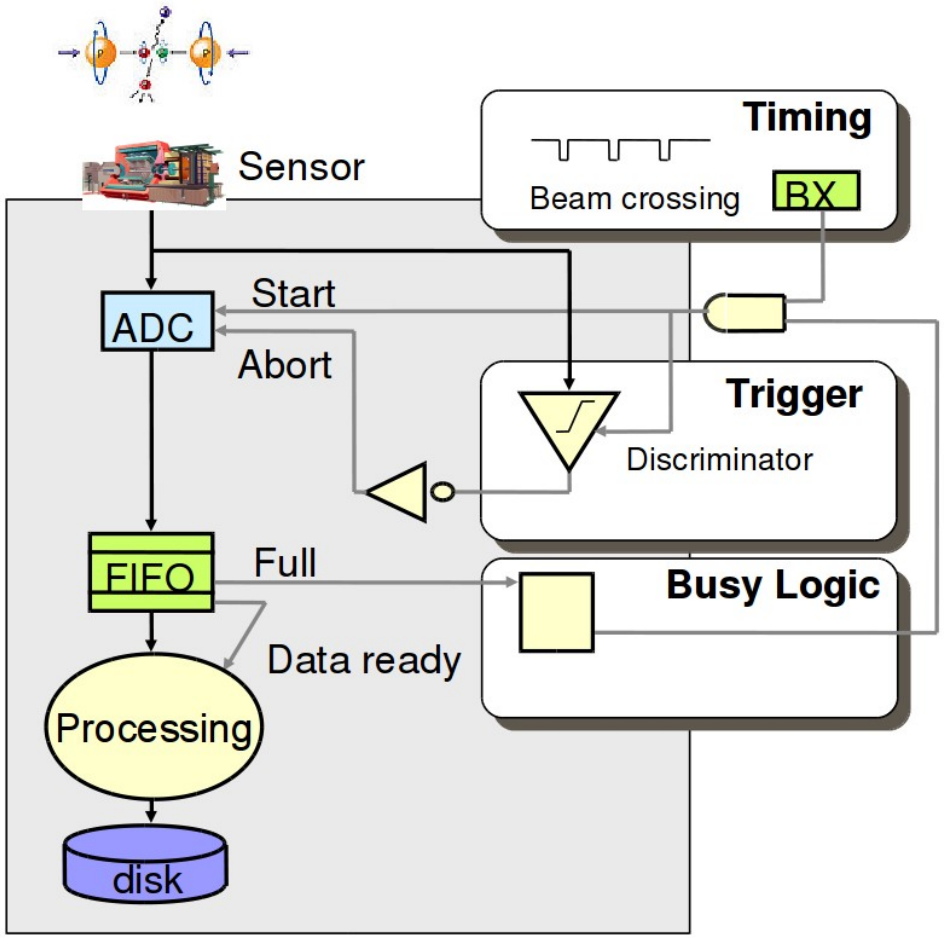
- First-In First-Out
  - buffer area organized as a queue
  - depth: number of memory cells
  - implemented in HW and SW



- Buffering introduces additional latency on data path

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

Basic DAQ: Collider Mode



- Synchronous particle collision rate
- Trigger rejects (= does not select) uninteresting events
- Synchronous collisions but unpredictable and uncorrelated triggers
- De-randomisation still needed

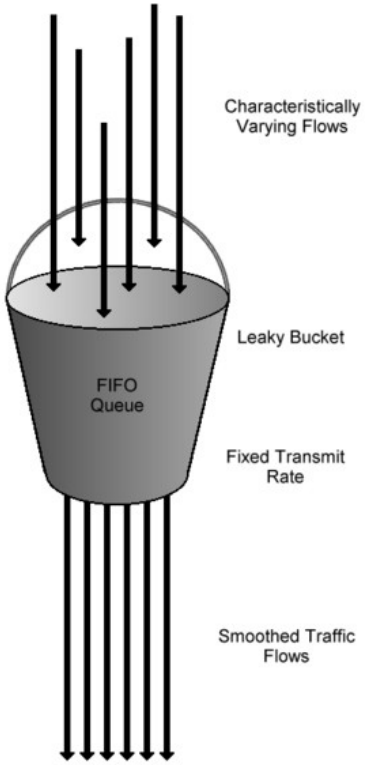
Buffering

does buffering solve all problems ?

- FIFO
  - filled with variable input flow
  - emptied at smoothed output flow

    → the Leaky-Bucket problem

- Q: how often may overflow (= FIFO full) ?



some very simple queueing theory

- N-event buffer ... single queue size N:

P_k : % time with k events in ; P_N = no space available → dead time

$\sum P_k = 1$ [k=0..N]

rate [j→j+1] = $\lambda \cdot P_j$ (fill at rate λ)

rate [j+1→j] = $\mu \cdot P_{j+1}$ (empty at rate μ > λ)

steady state: $\mu \cdot P_{j+1} = \lambda \cdot P_j \Rightarrow P_{j+1} = \rho \cdot P_j = \rho^{j+1} \cdot P_0$ [$\rho = (\lambda/\mu) <\sim 1$]

for $\rho \sim 1 \Rightarrow P_j \sim P_{j+1} \Rightarrow \sum P_k \sim (N+1) \cdot P_0 = 1 \Rightarrow P_0 \sim P_N \sim 1/(N+1)$

$\Rightarrow$ dead time ~ 1/(N+1)

want ~ 1% $\Rightarrow$ N ~ 100

some very simple queueing theory

- N-event buffer ... single queue size N:

P_k : % time with k events in ; P_N = no space available → dead ti

$\sum P_k = 1$ [k=0..N]

rate(j→j+1) = $\lambda \cdot P_j$ (fill at rate

rate(j+1→j) = $\mu \cdot P_{j+1}$ (em ate μ > λ)

steady state: $\mu \cdot P_{j+1} = \lambda$ $P_{j+1} = \rho \cdot P_j = \rho^{j+1} \cdot P_0$ [$\rho = (\lambda/\mu) <\sim 1$]

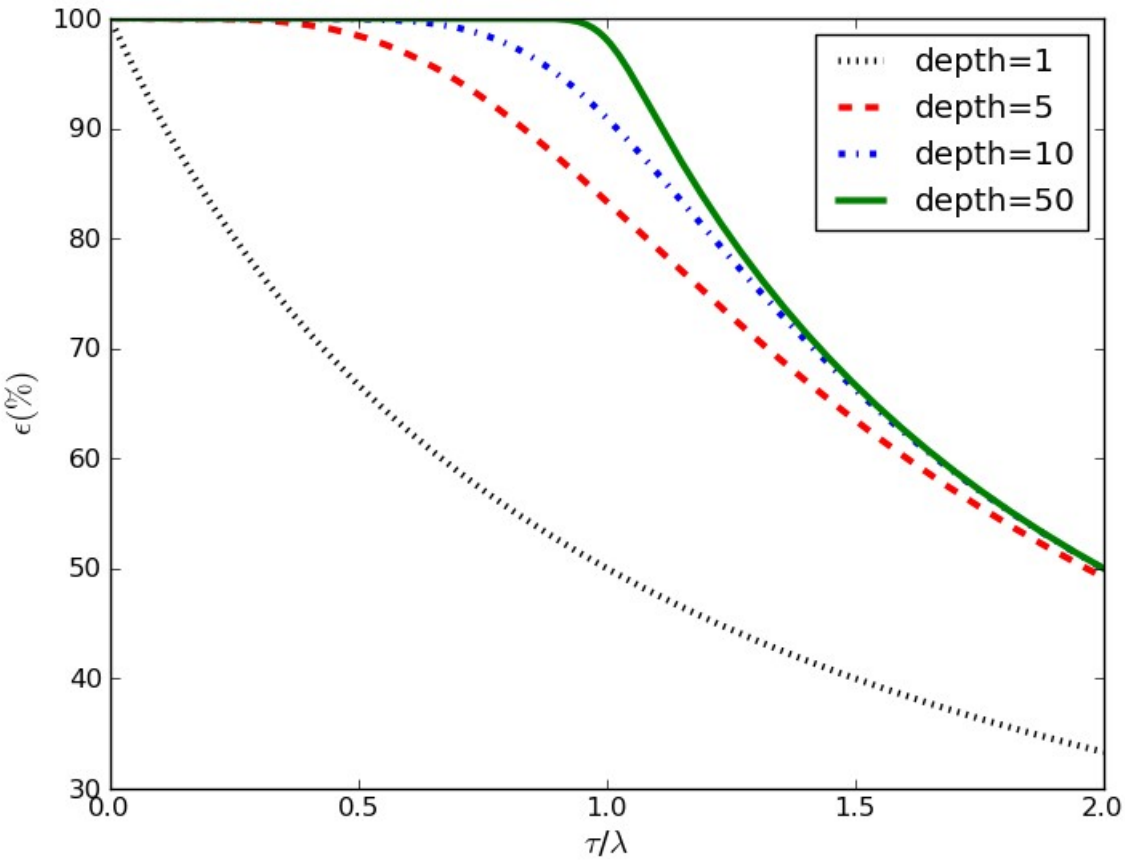
for $P_j \sim P_{j+1} \Rightarrow \sum P_k \sim (N+1) \cdot P_0 = 1 \Rightarrow P_0 \sim P_N \sim 1/(N+1)$

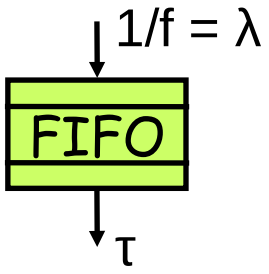
$\Rightarrow$ dead time ~ 1/(N+1)

want ~ 1% $\Rightarrow$ N ~ 100

Take care: analytic calculation possible for pretty simple systems only

de-randomisation





- DAQ ε ~100% with:
  - τ ~ 1/f
  - “moderate” buffer size
- Two degrees of freedom to play with
- This dead time often managed by trigger system itself (“complex dead time”)

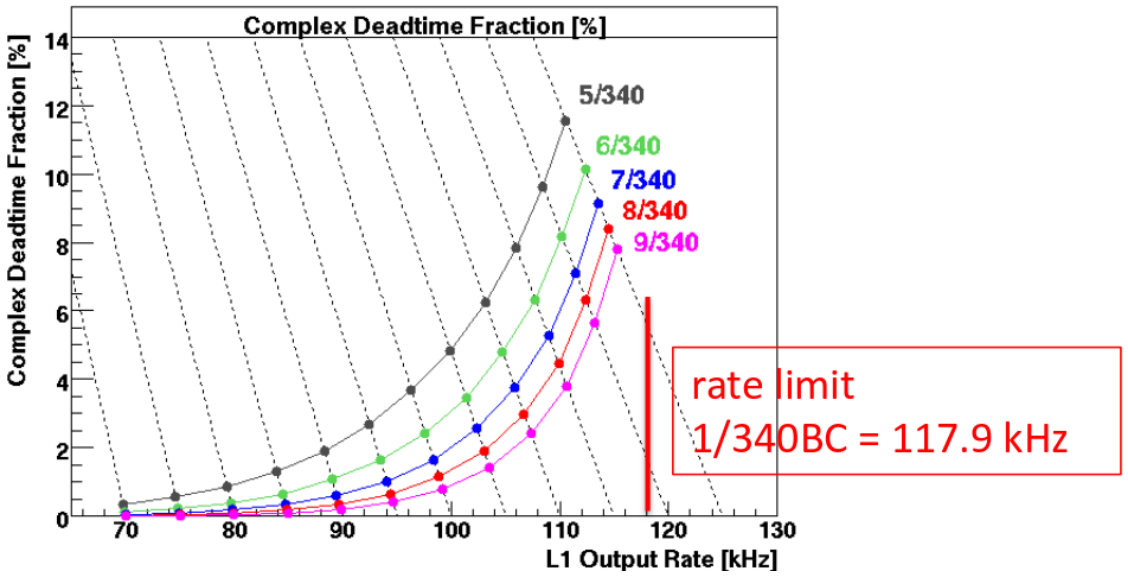
Complex Dead Time

1) Simple dead time: avoid overlapping (conflicting) readout window

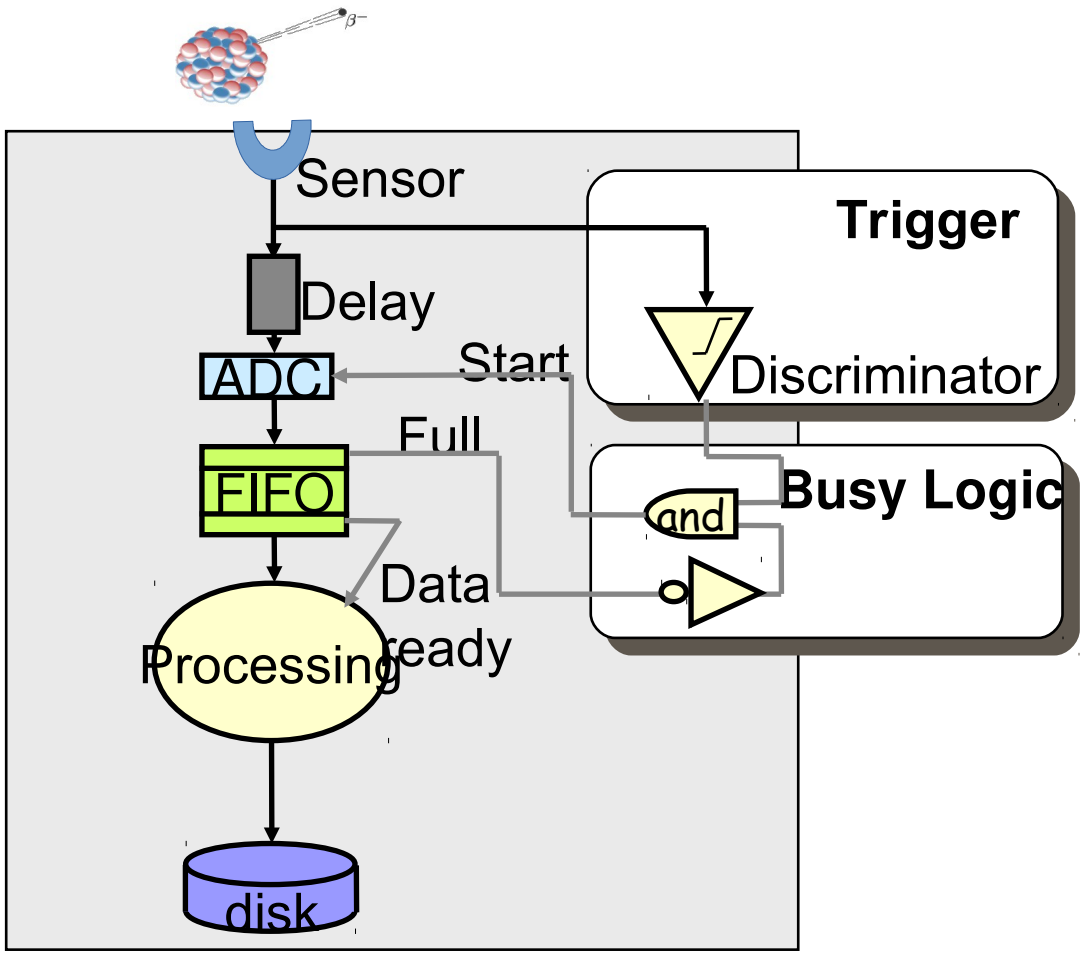
2) Complex dead time: avoid overflow in front-end buffers (protection against trigger bursts)

e.g. ATLAS uses simple leaky-bucket algorithms with 2 parameters:

max X triggers (X = FIFO depth) in any (sliding) time window = (X*readout time)



De-Randomisation: Summary



- ~100% efficiency and minimal deadtime may be achieved if
  - ADC rate >> f
  - data processing and storing at rate ~f
- FIFO decouples low latency front-end from data processing
  - minimal amount of “unnecessary” fast components
- Could “Delay” be replaced with “FIFO”?
  - analog pipelines heavily used in LHC DAQs

Multi-Level Trigger Architecture

Multi-Level Triggering

Often implemented over 3 levels (alternative is 2) :

L1 (sometimes called L0) → detector-specific front-end electronics

L2 (sometimes L1) → regional information processing (dedicated processing units, working with limited resolution information)

L3 → typically some “standard” processing farm (working with full resolution information)

L2 & L3 → HLT.s (High-Level Triggers)

Event Selection Latency

- L1 : O(1 μs in real-time) → let say ~ 3 μs
- L2 : O(10 ms) → let say ~ 20 ms
- L3 (HLT) : O(s) → let say ~ 1 s

Q: do these 3 numbers mean the same thing ?

Well … depends ...

Latency and Real-Time

real time: system must respond within some fixed delay

→ Latency = Max Latency (constrained)

→ over fluctuations bad, will create dead time

non-real-time: system responds as soon as it's available

→ Latency = Mean Latency (not constrained)

→ over fluctuations ok, shouldn't create dead time

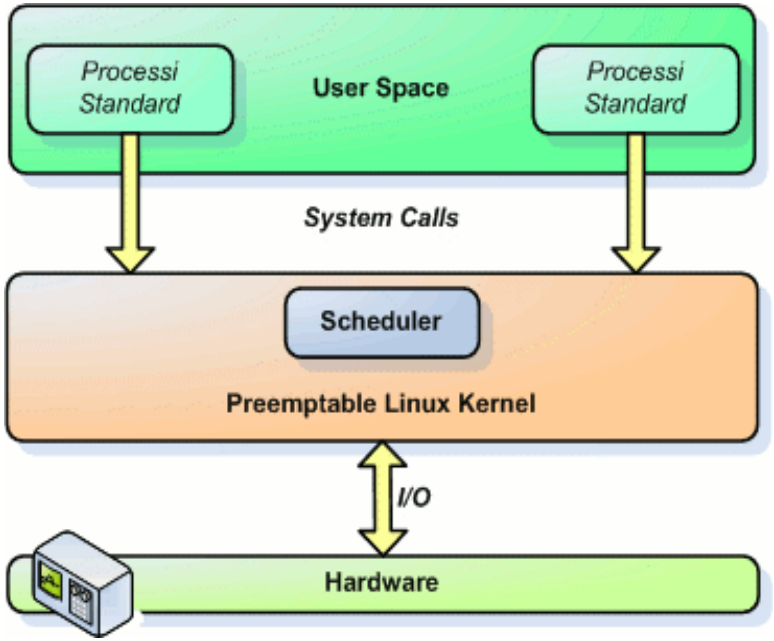
real time o.s. :

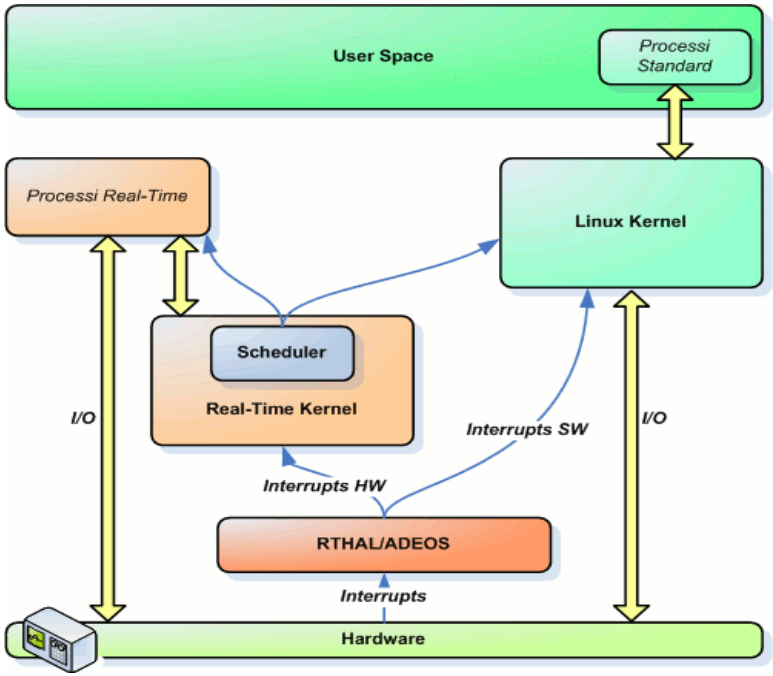
very stable time delay in responding to events

standard unix kernels are not real time:

no duration constraints for system calls

Off-Topic: Real-Time Linux





Low-latency Ubuntu patch

(soft real time) :

Interruptible linux kernel

https://help.ubuntu.com/community/UbuntuStudio/RealTimeKernel

RTAI (hard real time) :

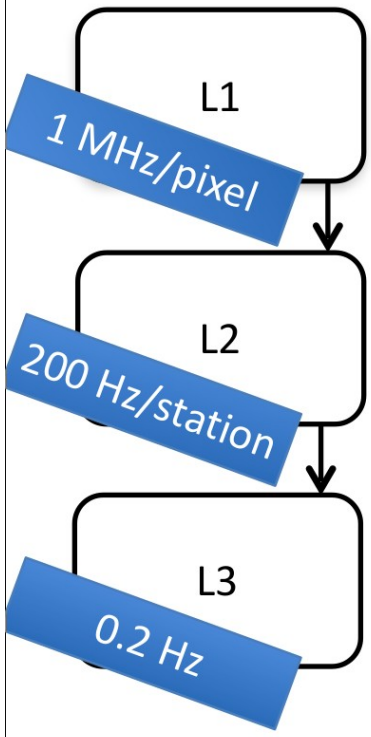
linux kernel as high-priority application

https://www.rtai.org/

Working trigger systems

Auger Observatory : Simple Signatures

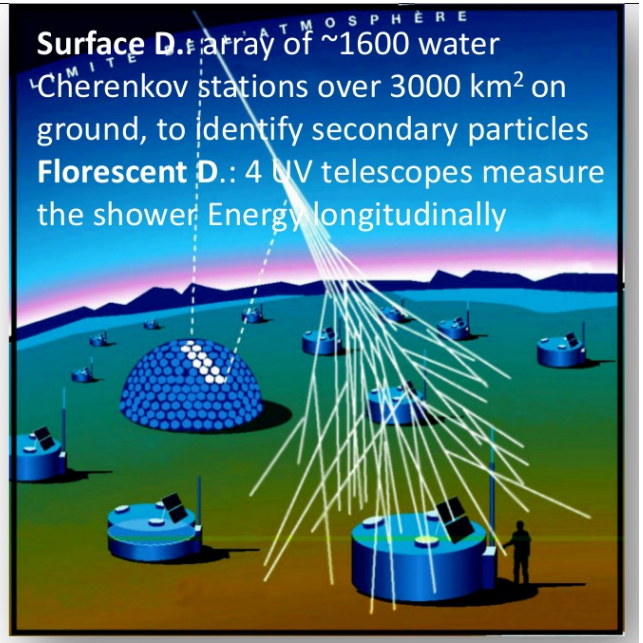
- **Detect air showers generated by cosmic rays above 10^{17} eV**
  - Expected rate < 1/km²/century. Two large area detectors
- On each detector, a 3-level trigger operates at a wide range of primary energies, for both vertical and very inclined showers

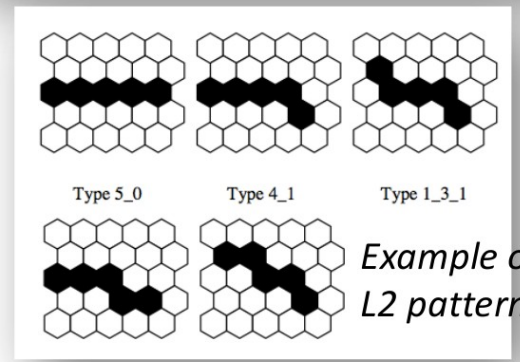


L1: (local) decides the pixel status (on/off)
- ADC counts > **threshold**
- ADC digitizes any 100 ns (**time** resolution)
- ADC values stored for 100 μs in **buffers**
- **Synchronized** with a signal from a GPS clock

L2: (local) identifies track segments
- Geometrical criteria with recognition algorithms on programmable patterns

L3: (central) makes spatial and temporal correlation between L2 triggers



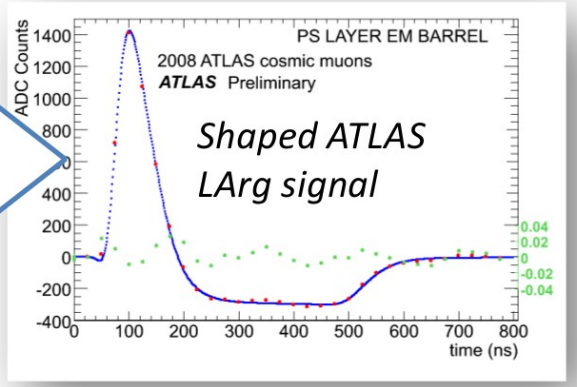


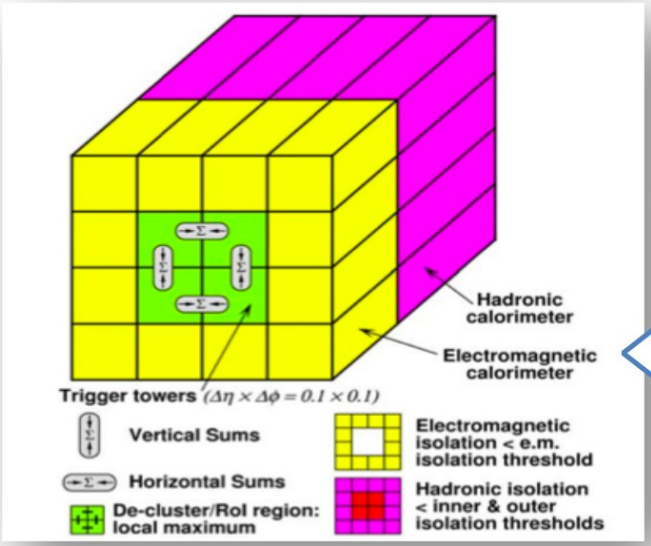
Example of L2 patterns

One event ~ 1MB → 0.2 MB/s bandwidth needed for the DAQ system

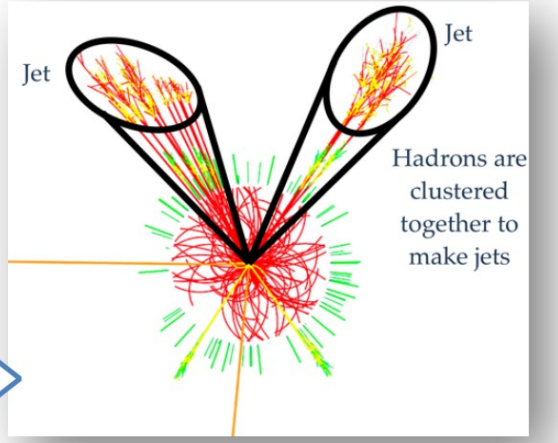
ATLAS Calorimeter : Multiple Signatures

- Identify high energy e, γ, τ, jets, missing E_T, ΣE_T
  - 1: Dedicated Front-End electronics
    - Front-End of cells sends shaped analog signals

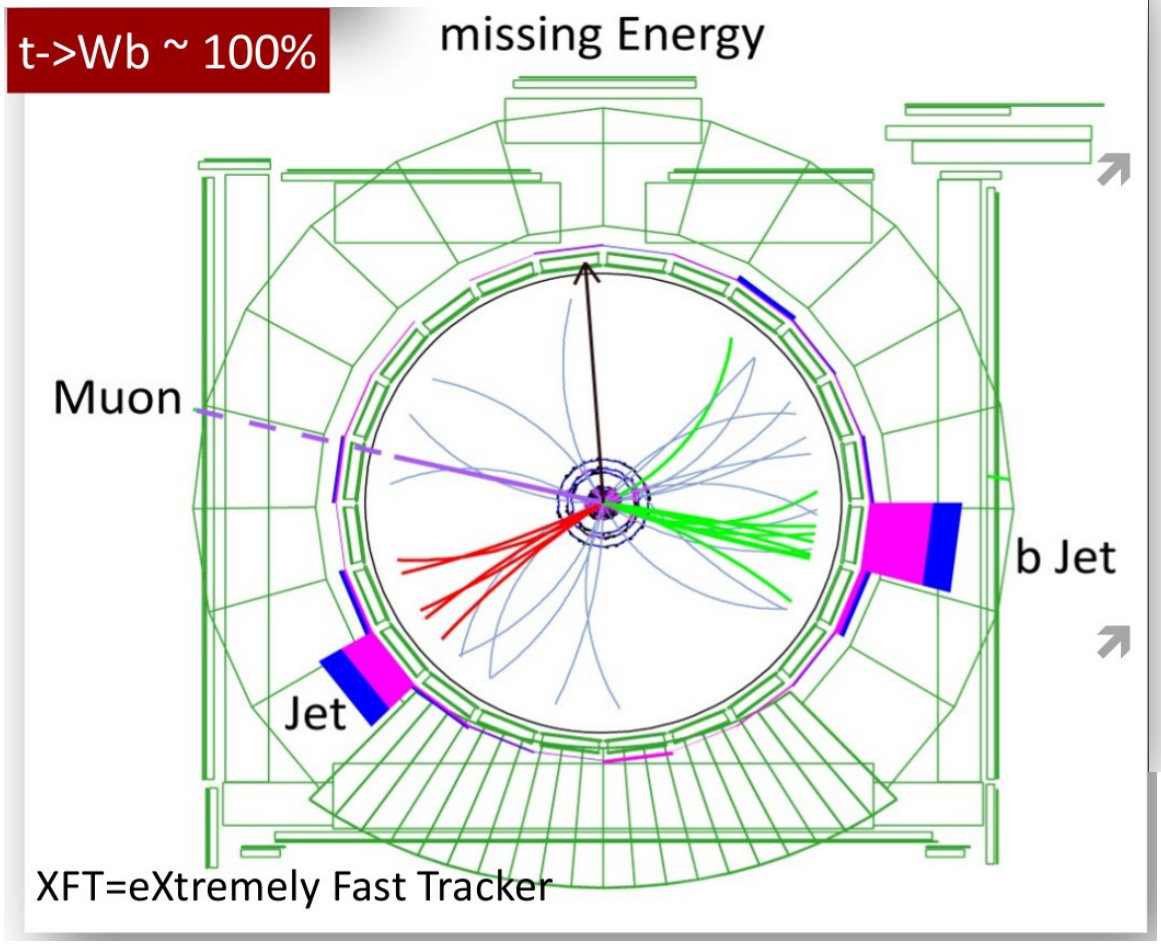




  - 2: Level-1 trigger
    - Dedicated **processors** apply simple cluster algorithms over cells and programmable E_T thresholds
  - 3: High-Level triggers
    - **Topological** variables and **tracking** information
      - e/jet separation using cluster shapes
      - e/γ separation using tracking
    - **Isolation** criteria



CDF Single Top : Multi-Object Trigger



Signal characterization:
- 1 high p_T lepton, in general isolated
- Large MET from high energy neutrino
- 2 jets, 1 of which is a b-jets

Trigger objects at L1
- Central tracking (XFT p_T>1.5GeV)
- Calorimeter
  - Electron (Cal +XFT)
  - Photon (Cal)
  - Jet (Cal EM+HAD)
- Missing E_T, SumE_T
- Muon (Muon + XFT)

Trigger objects at L2:
- L1 information
- SVT (displaced track, impact parameter)
- Jet cluster
- Isolated cluster
- Calorimeter ShowerMax (CES)

Trigger Efficiency

$$BR(\ Signal\ ) = \frac{(N_{candidates} - N_{bg})}{\alpha \cdot \varepsilon_{total} \cdot \sigma_{Bs} \cdot \int Ldt}$$

$$\alpha \cdot \varepsilon_{total} = \alpha \cdot \varepsilon_{Tracking} \cdot \varepsilon_{Reco} \cdot \varepsilon_{L1\text{-}Trig} \cdot \varepsilon_{L2\text{-}Trig} \cdot \varepsilon_{L3\text{-}Trig} \cdot \varepsilon_{vertex} \cdot \varepsilon_{analysis}$$

Must be precisely known (w/ its systematics)

Independent trigger selections allows cross-calibration
→ need of additional triggers

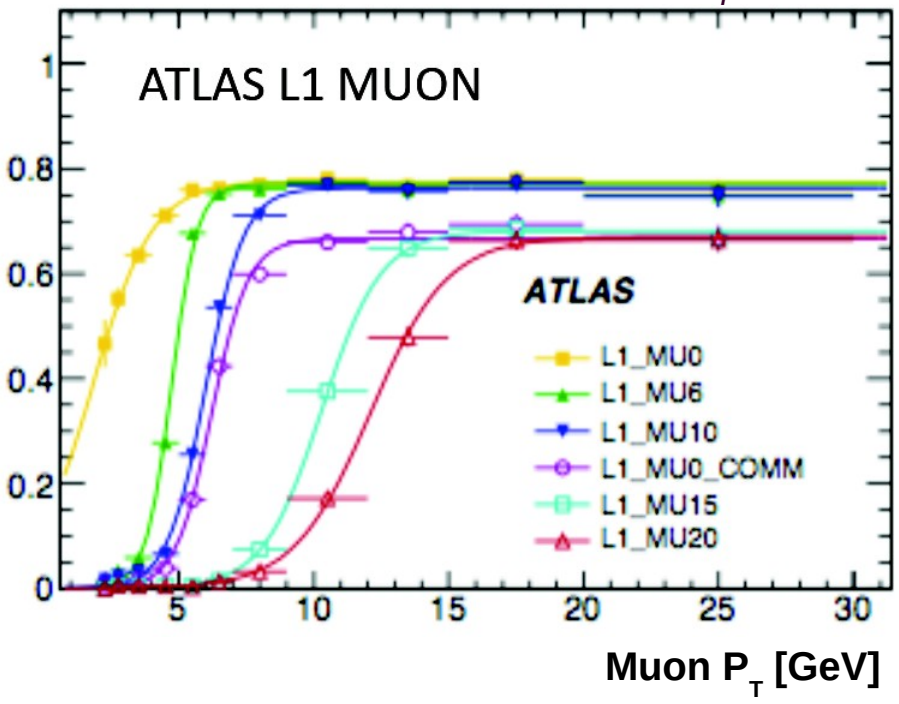
High-Level Triggers → pass-through triggers (release selection criteria)
Level-1 Triggers:
a) zero or minimum bias samples
b) Tag&Probe → trigger on “Tag” and measure “Probe”

Trigger Thresholds

Finite resolution of trigger detectors affects measurements near threshold
→ efficiency badly defined in transition (turn-on) region
→ need margins to get offline thresholds at plateau

LHC (ATLAS):
thresholds dynamically tuned as function of luminosity
[keep throughput constant]

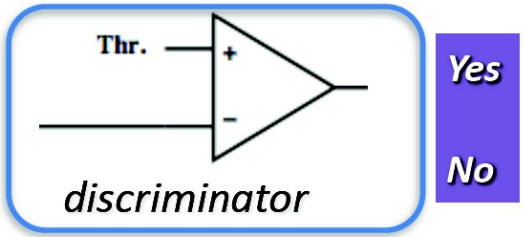
Efficiency vs. Muon P_T



Zoology: Some Building Blocks

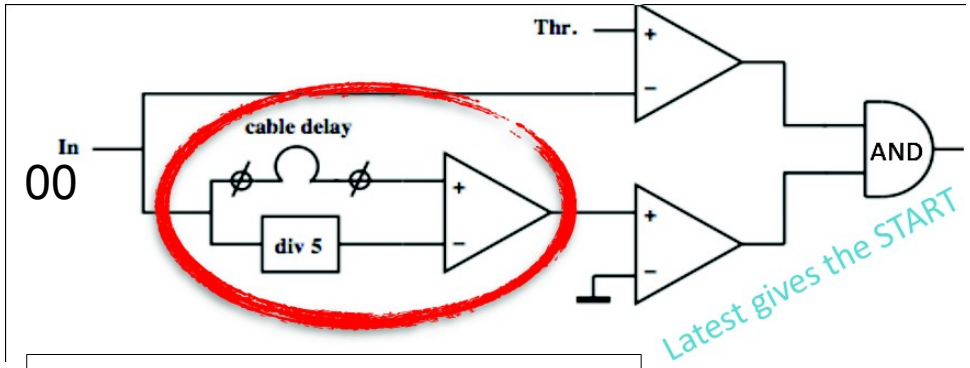
Discriminators

Discriminator: Schmitt Trigger + Monostable

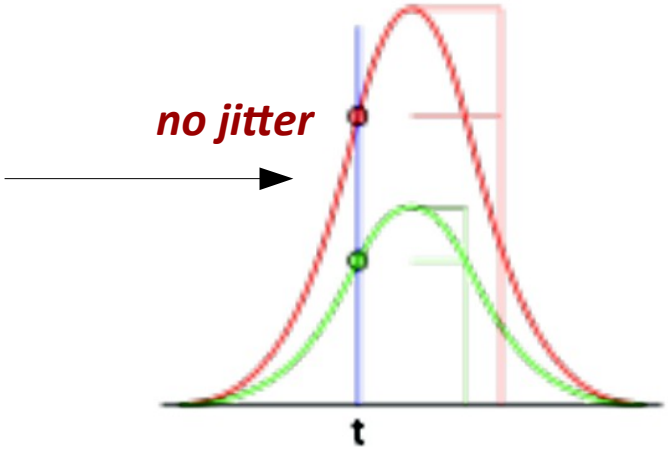


Yes

No



Constant Fraction Discriminator

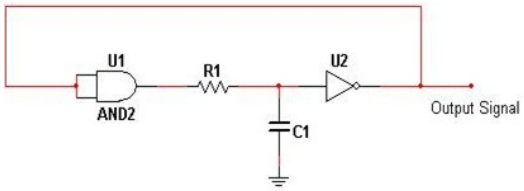


Digital part: combinatorial logic and flip-flops (finite-state machine)

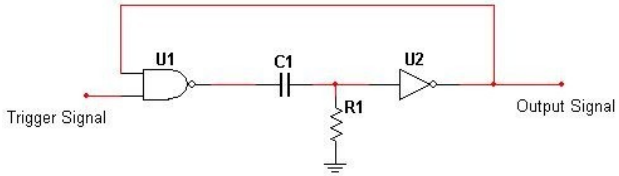
Multivibrators

Output signal (Q) may have:

No stable level → astable (oscillator)

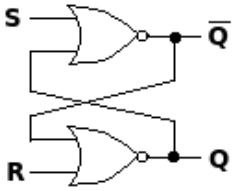


One stable level → monostable (one-shot single pulse)

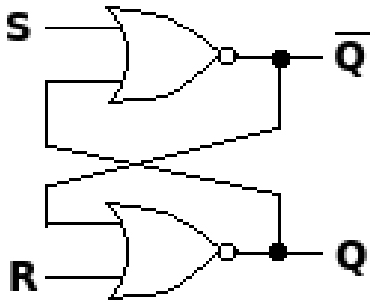


Two stable levels → bistable

Set/Reset Latch:



Set/Reset Latch



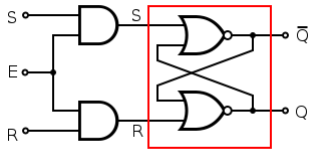
Truth table

S	R	Q	State
0	0	Previous State	No change
0	1	0	Reset
1	0	1	Set
1	1	?	Forbidden

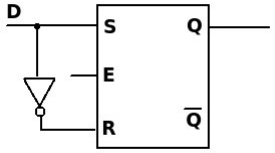
Q may (may!) change only when either S or R goes to 1

Latches / Flip-Flops

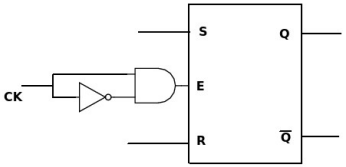
Gated S/R Latch:



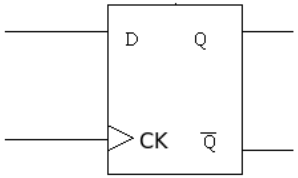
Gated D Latch:



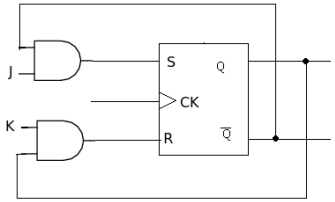
S/R Flip-Flop:



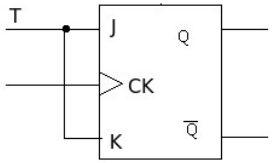
D FF (1-bit memory):



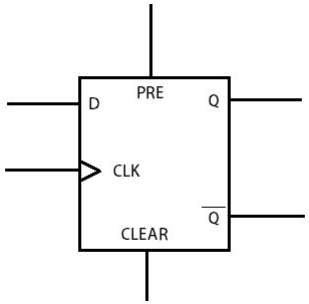
J/K FF:



T (Toggle) FF (counter):



Asynchronous Inputs:



Memories

RAM : Random Access Memory (RWM, ROM, PROM, EPROM, EEPROM, …)

CAM : Content Addressable Memory (Associative Memory)

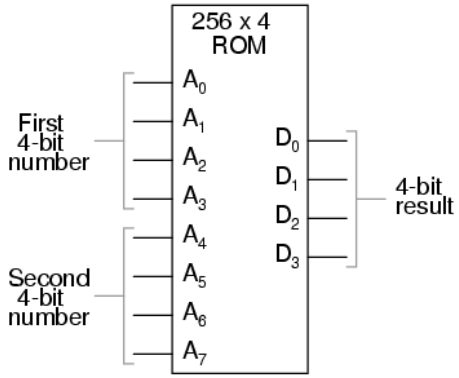
SAM : Sequentially Accessible Memory (FIFO, LIFO)

Look-Up Tables (LUT)

Read-Only Memory: Address → INPUT & Data → OUTPUT:

used to implement simple/complex logic functions

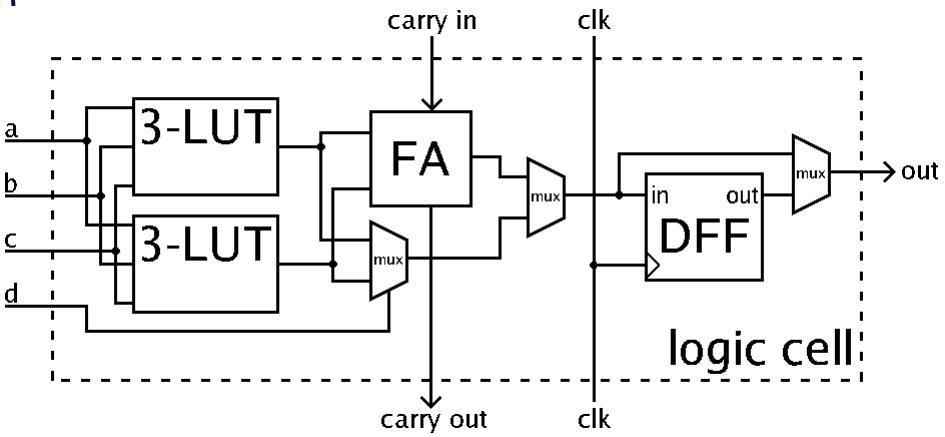
all results MUST be loaded in memory



FPGA: Field-Programmable Gate Array

array of configurable logic blocks providing tons (thousands → millions) of dynamically connected logic units, made of:

- look-up tables (to perform complex combinatorial functions)
- flip-flops (to synchronously store results)



incredibly competitive wrt. ASIC until don't need millions of chips

- may start your own project, in few days, with O(50-100) €
- "you can design a circuit on your computer and have it running on your desk in minutes" → http://www.fpga4fun.com

Trigger Hardware

What to use, today, mainly depends on latency requirements:

[my_basic_daq : modular NIM/VME electronics]

L0/L1 O(1 µs + real time) : custom ASIC or/and LUT (FPGA, CAM)

L1/L2 O(10 µs) : FPGA, CAM, DSP, fast processors

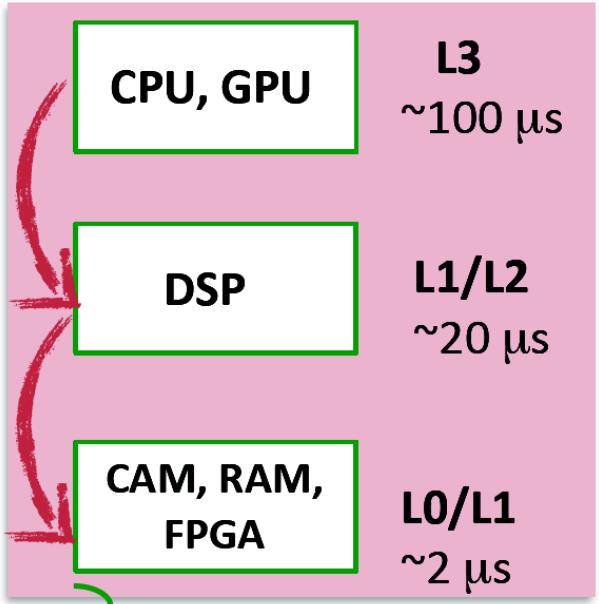
L3 O(100 µs) : CPU, GPU

real time: max latency (i.e. max delay) fixed
→ over fluctuations bad, will create dead time

non-real-time: average latency fixed
→ over fluctuations fine, shouldn't create dead time

(off-the-shelf) Programmable Devices

- Requirements at high trigger rates
  - Fast processing
  - Flexible/programmable algorithms
  - Data compression and formatting
  - Monitor and automatic fault detection

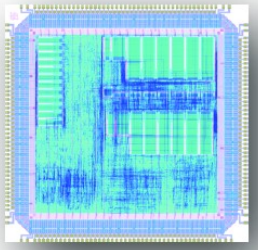


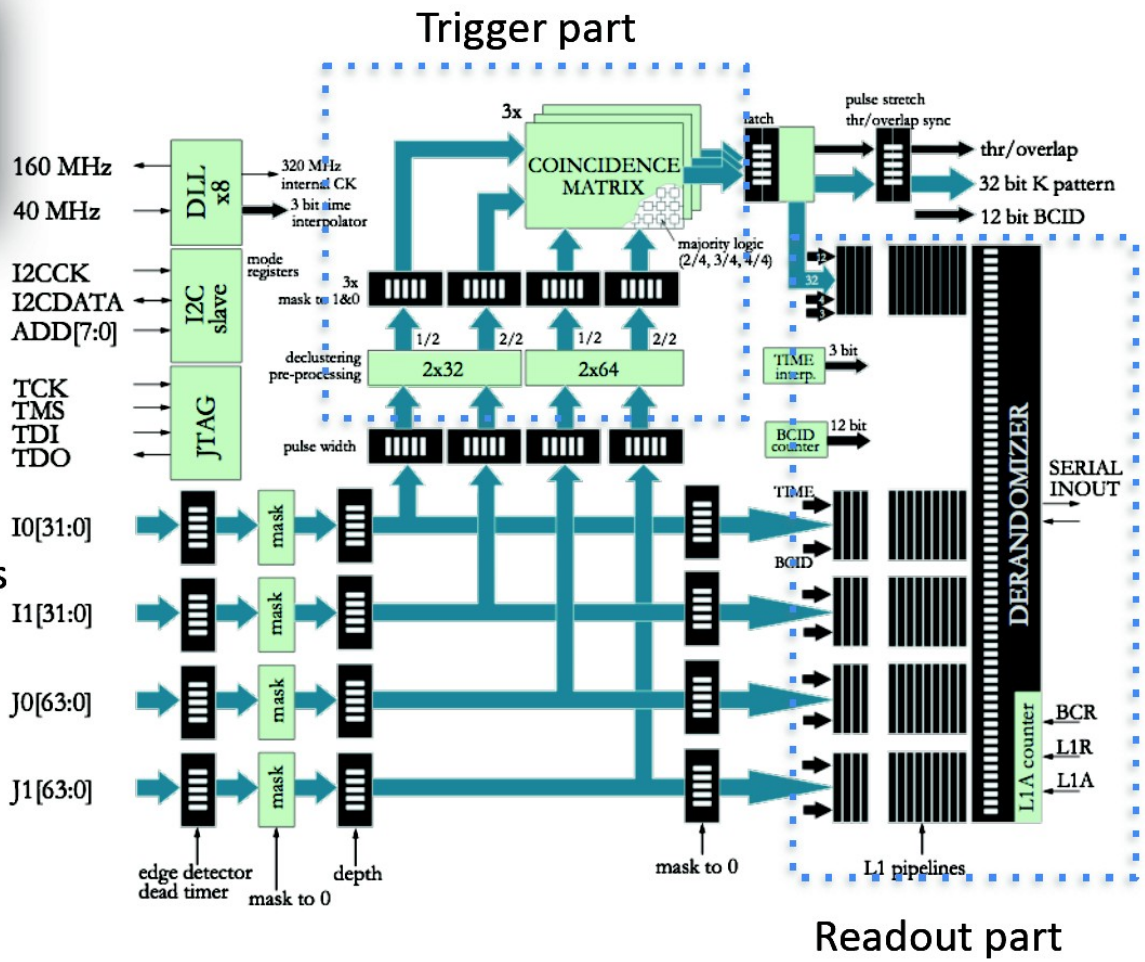
- Digital integrated circuits (IC)
  - Reliability, reduced power usage, reduced board size and better performance
- Different families on the market:
  - **Microprocessors** (CPUs, GPGPUs, ARMs, DSP=digital signal processors..)
    - Available on the market or specific, programmed only once
  - **Programmable logic devices** (FPGAs, CAMs,...)
    - More operations/clock cycle, but costly and difficult software developing
  - **New trend is the integration of both:**
    - Using standard interface (ethernet), can profit of standard software tools (like for Linux or real-time) and development time is reduced

need instructions

already learned the task

Custom ASIC





Digital inputs from detector

Trigger output

Readout output

Input: synchronization signals

ATLAS Muon Barrel Coincidence Matrix

to be continued...