Intro to Triggering and Data Acquisition

Samuel Silverstein, Stockholm University

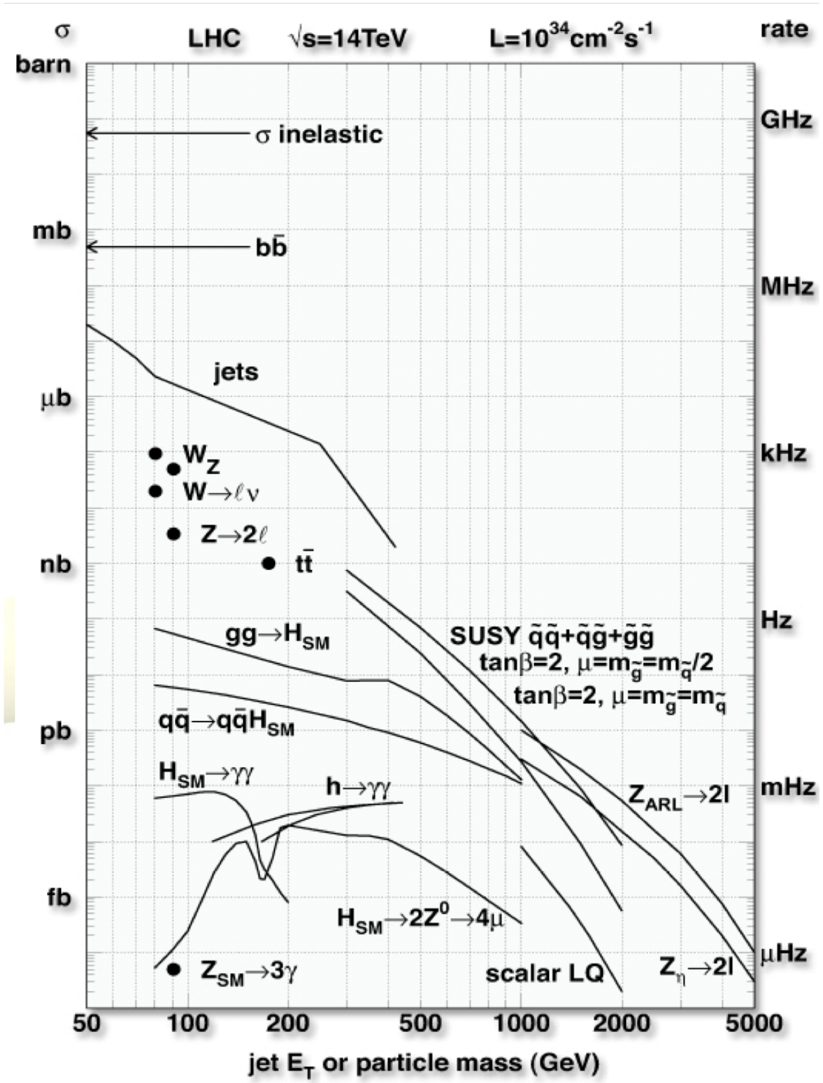
- Trigger/DAQ basics
- Collider TDAQ
- Signal processing

In this trigger/DAQ series:

- Three segments
  - Trigger/DAQ introduction, concepts
  - TDAQ architectures
  - New developments
- Exercises
  - Programmable logic (lab)
  - Dead-time simulation (homework)

Introduction

Colliders produce a <u>lot</u> of data!



- Physics at LHC:
  1 interesting event in ~10^6-10^{13}
- Need a high collision rate:
  - 40 MHz bunch crossing rate
  - 25-50 proton collisions per bunch crossing
- Collision data volume
  - About 1.5 MB/event (ATLAS)
  - ~60 TB/s at 40 MHz!
- <u>Too much data!</u>

Why is it too much?

(In reverse order…)

- Can't analyse it all…
  - Would need a million times more processing
- Can't store it all…
  - 60 TB/s is about 3.6 petabytes/minute
- Can't get it all off the detector*
  - High-bandwidth data links are expensive, take up space and consume power.
  - Practical consequences:
    - Heat dissipation
    - Cables, power, cooling, etc leave dead material, and "holes" in detector coverage

*N.B. New link technologies making this less true today, for *some* systems

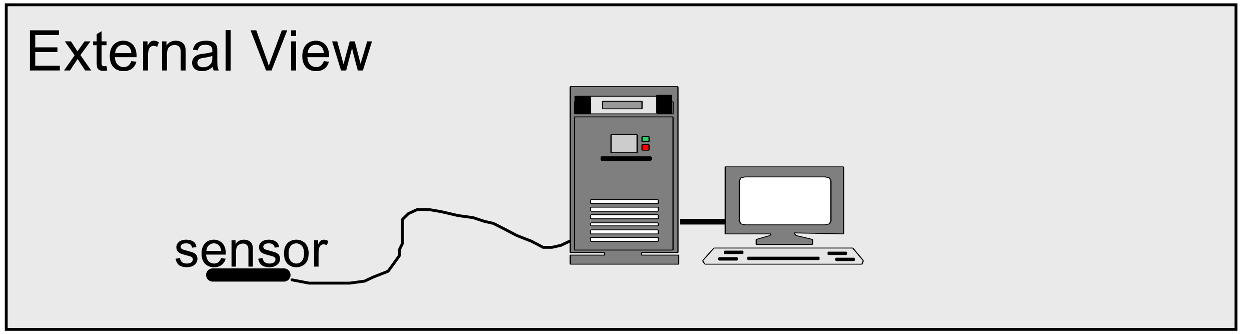
The challenge for TDAQ

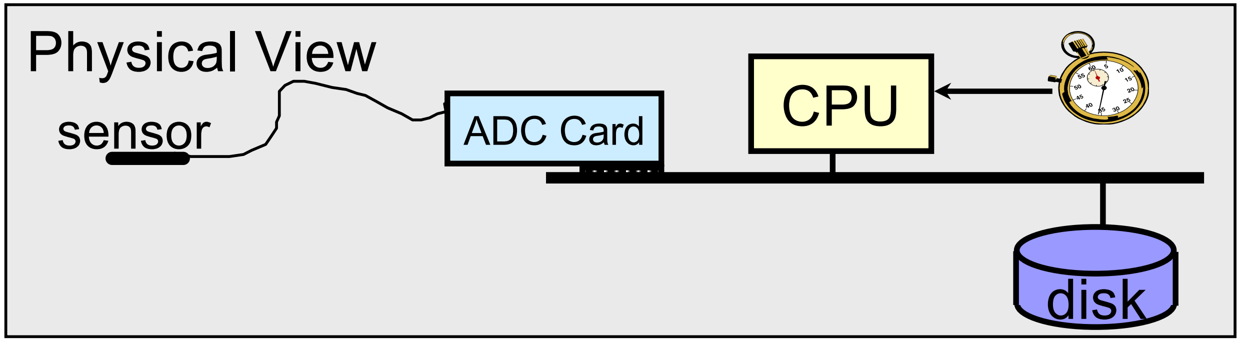
- Throw away ~99.999% of collision data
  - To keep data rates at manageable levels
- Don't throw away:
  - Interesting SM processes
  - New physics predicted by BSM models
  - Unexpected new physics signals
- Doing all of this well is hard!
  - And perfection is practically impossible

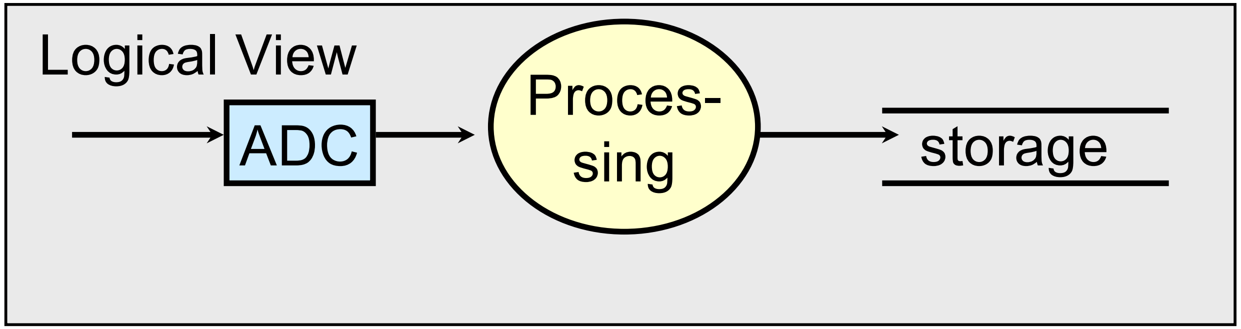
Trigger / data acquisition is a compromise between physics goals and what is technically achievable!

Data Acquisition Basics

A trivial DAQ example…







Trivial DAQ

- How it works:
  - The sensor produces an analog signal
  - An ADC periodically converts analog output to digital values
  - The CPU reads the digital ADC output data and writes them to disk (readout)
- Potential problem:
  - If the readout rate is much larger than physics rate, there is a lot of uninteresting data to store and analyze.
- Solution:
  - Use a trigger to initiate readout only if there is an interesting signal

What is a trigger?

Wikipedia: “**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. “**

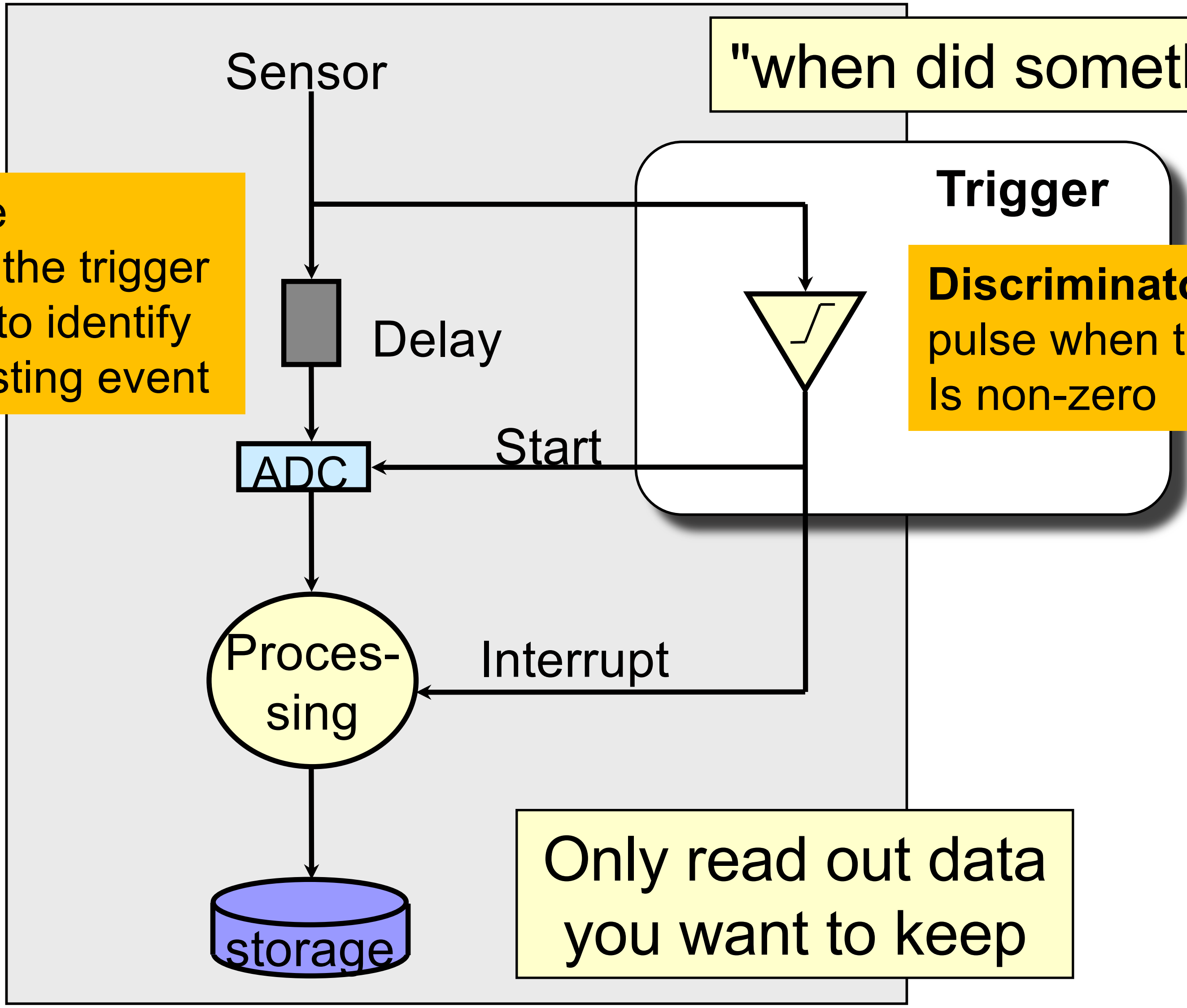
What is a trigger?

- ‘Simple criteria’
  - You just need to decide whether to keep the data. Detailed analysis comes later
- ‘Rapidly decide’
  - The data you want to keep may be lost if you wait too long to decide
- ’Only a small fraction can be recorded’
  - Selective algorithms are required, in order to record the small fraction that is interesting

Trivial DAQ with trigger

"when did something happen?"

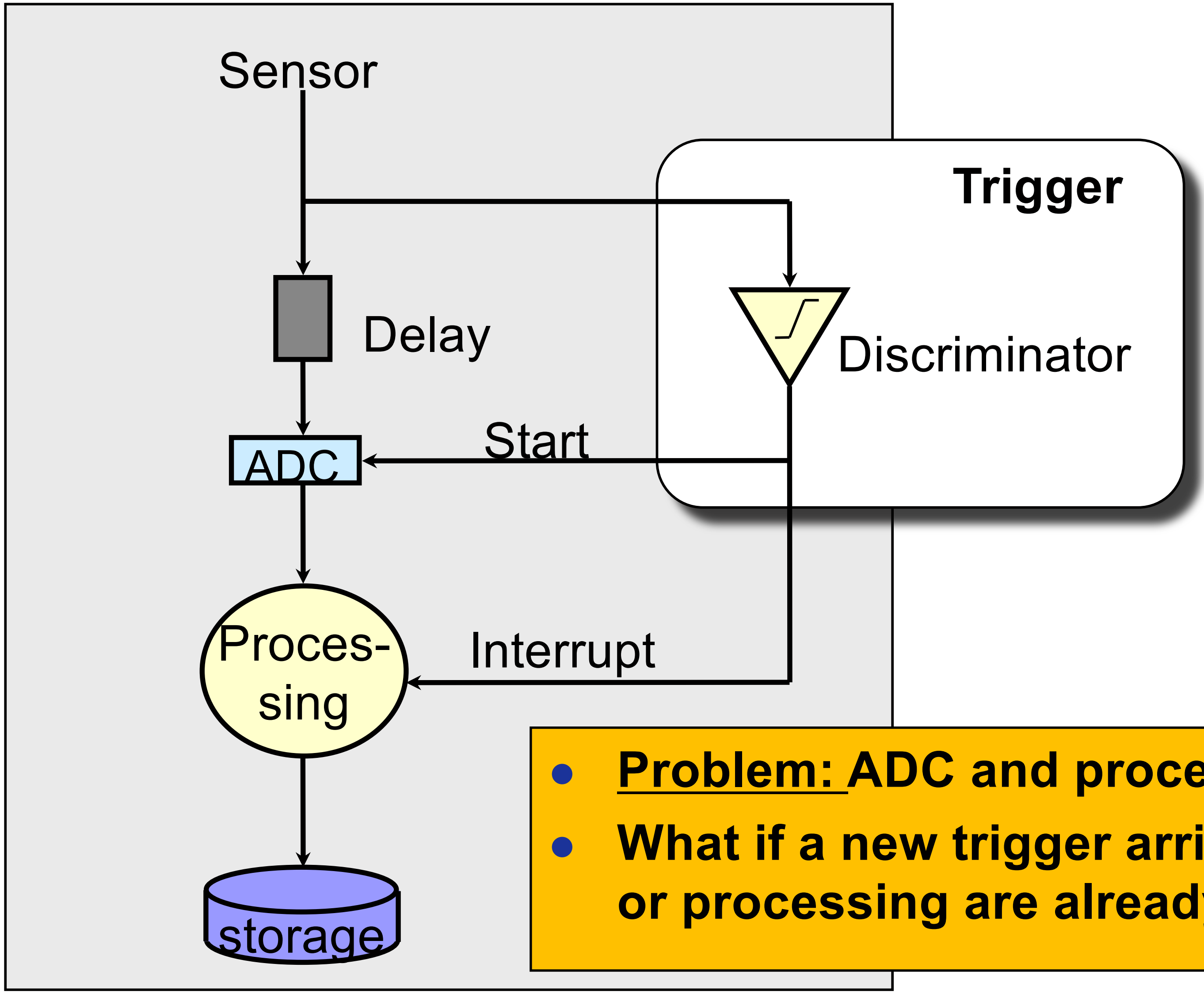
Delay the signal so the trigger has time to identify an interesting event



Discriminator: simple pulse when the signal Is non-zero

Only read out data you want to keep

Trivial DAQ with trigger

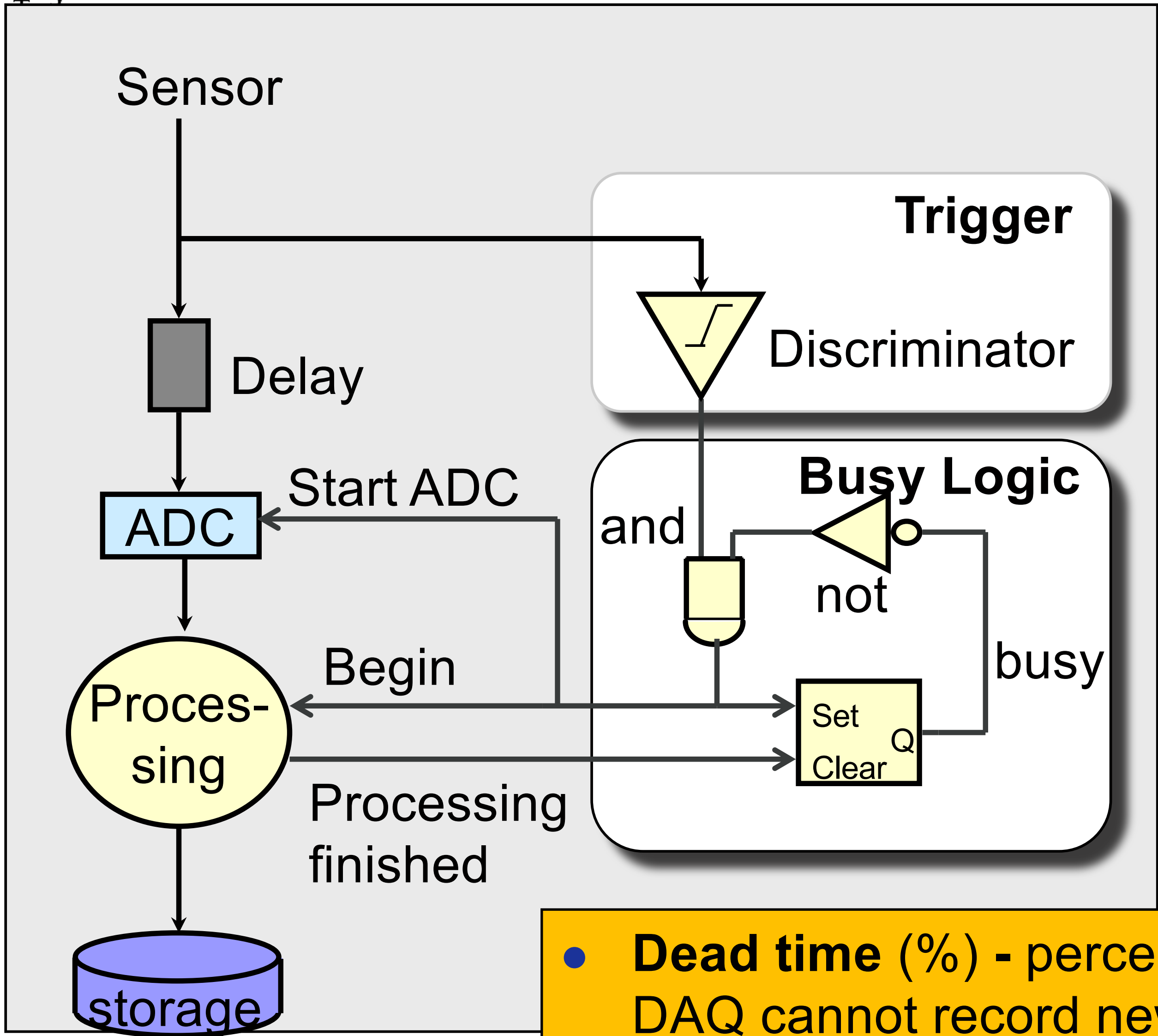


- **<u>Problem:</u> ADC and processing take time!**
- **What if a new trigger arrives when ADC or processing are already <u>busy</u>?**

Flow control

- New triggers can interfere with the readout if a previous event is still being processed
- Solution: flow control
  - ADC/readout send ‘Busy’ signal to temporarily disable the trigger
  - Events occurring during ‘Busy’ period are not recorded (dead time).

DAQ with trigger & busy logic



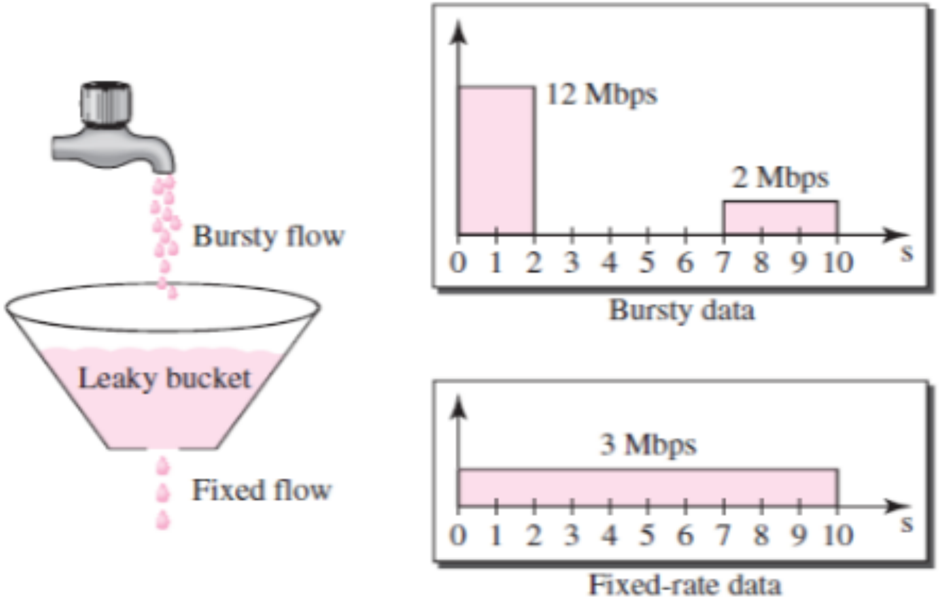
Busy is true during ADC conversion and processing time. During this time, new events cannot be triggered

- **Dead time** (%) **-** percentage of time that DAQ cannot record new events

Trigger vs. readout rates

- Triggers occur at random intervals
  - But events take ~same time to read out
- Dead time increases as the trigger rate approaches the readout rate
  - Higher chance that the DAQ will be busy when a new event arrives
  - Loss of data!
- A standard solution:
  - Add a derandomising buffer (FIFO) to even out the data flow

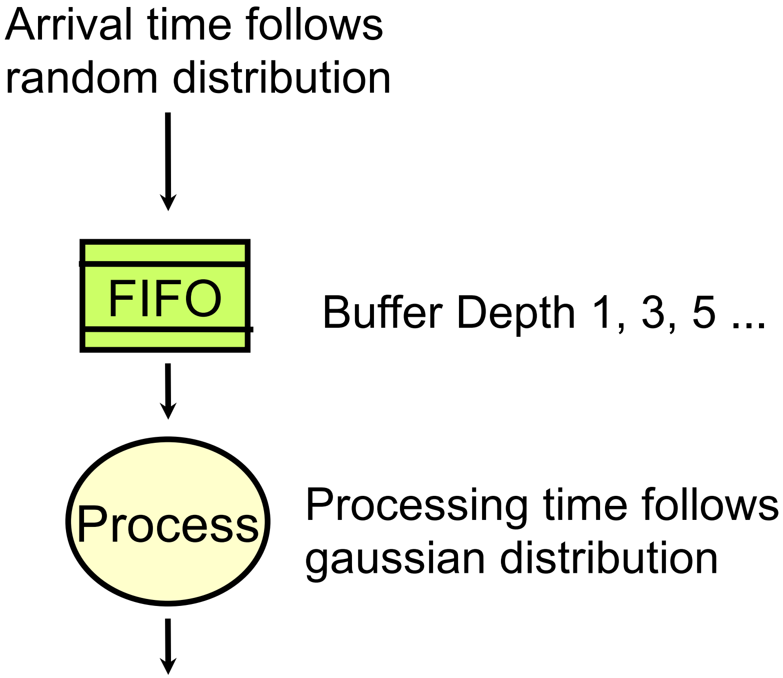
“Leaky bucket” FIFO model

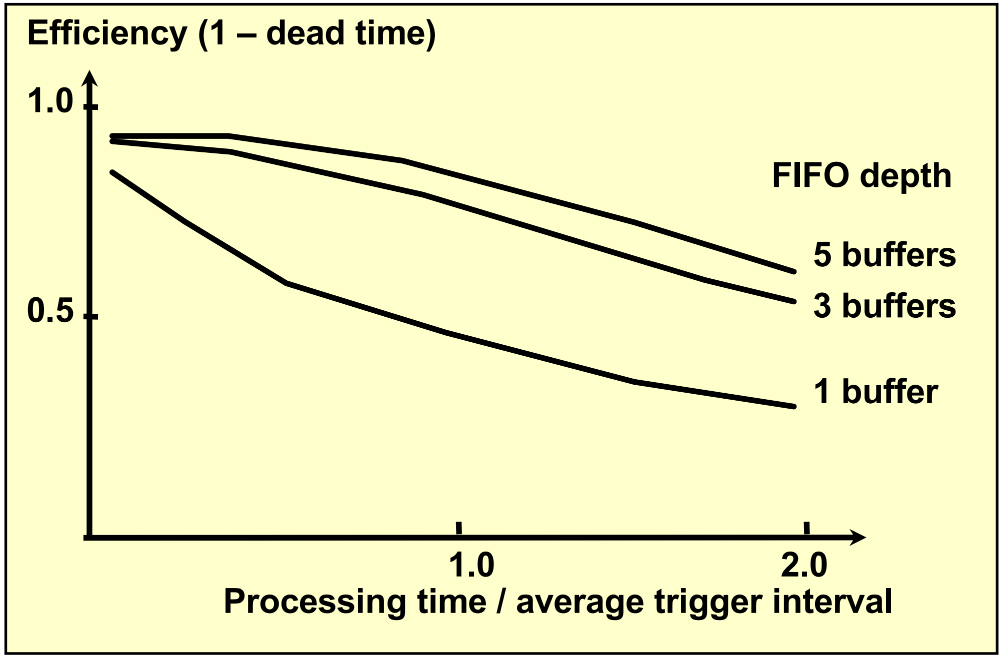


- Input arrives in “bursts”
- Output flow at a fixed rate
- No overflow if:
  - Output flow exceeds the average input flow
  - Bucket is large enough to handle short-term excess inputs

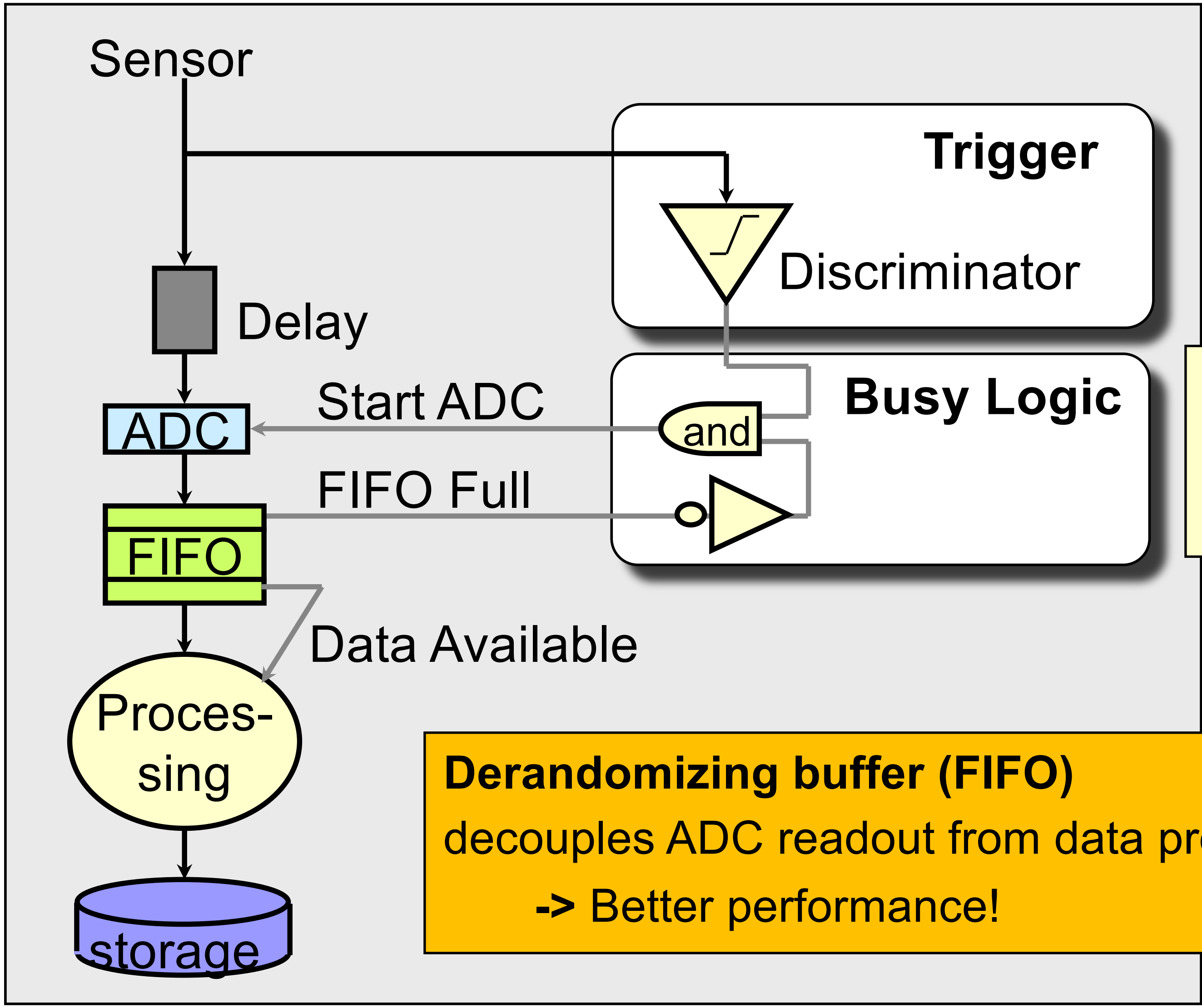
Buffers reduce dead time

- Buffers (FIFO) receive new events as they arrive, send them out when the processor is ready
  - "average out" arrival rate of new events
- Dead time depends on trigger rate, readout time, processing time per event, and FIFO depth





Buffered DAQ



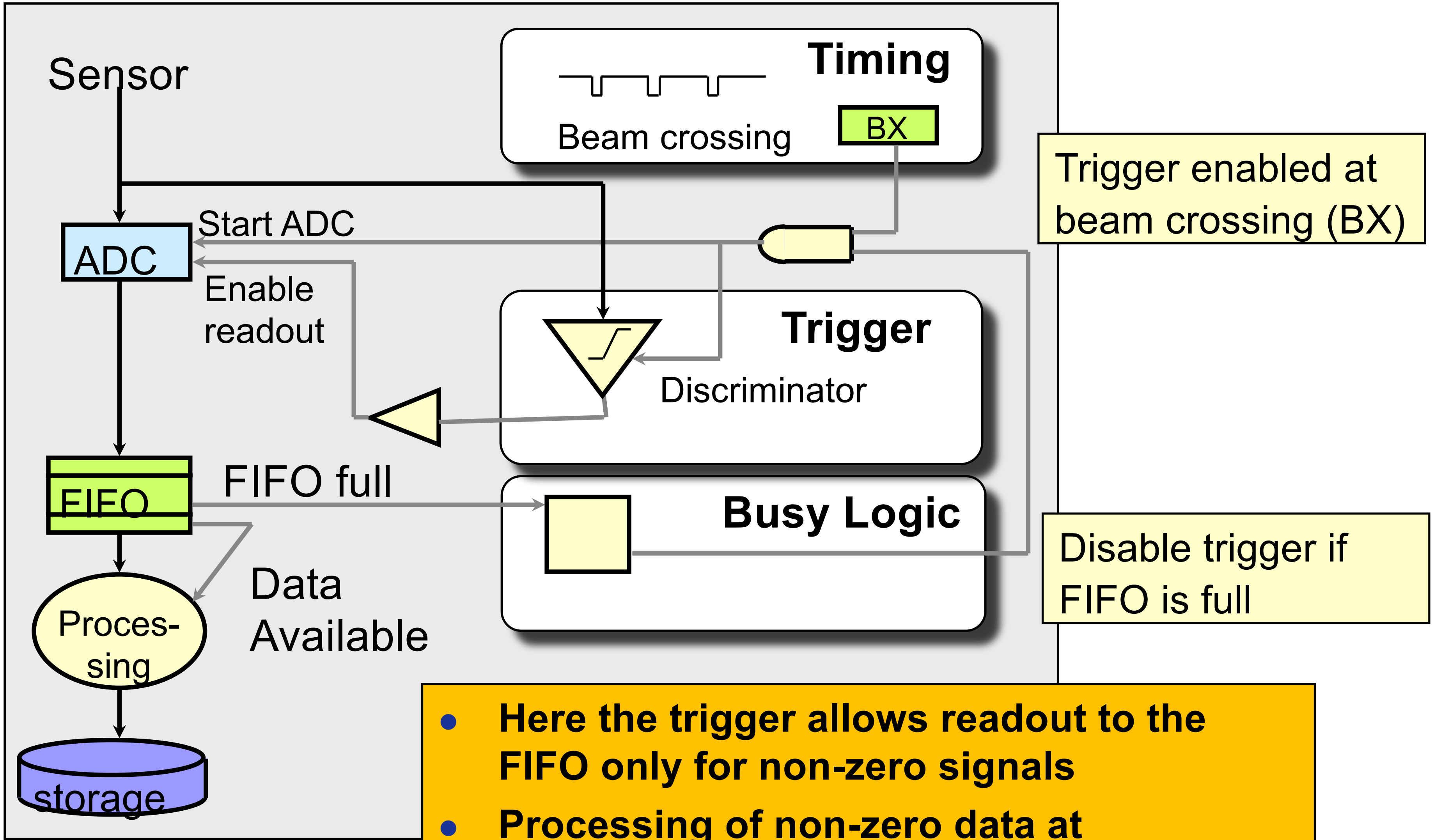
Busy time = ADC
(as long as the buffer is not full)

Derandomizing buffer (FIFO)
decouples ADC readout from data processing
-> Better performance!

Triggering on bunch crossings

- Particle colliders typically circulate particles in “bunches”
  - Collisions occur at a fixed bunch crossing (BX) interval
- The trigger analyses data from every BX
  - Produce yes/no decision to keep the event
  - Data not selected by the trigger are effectively “thrown away”.

A "trivial" collider-type trigger

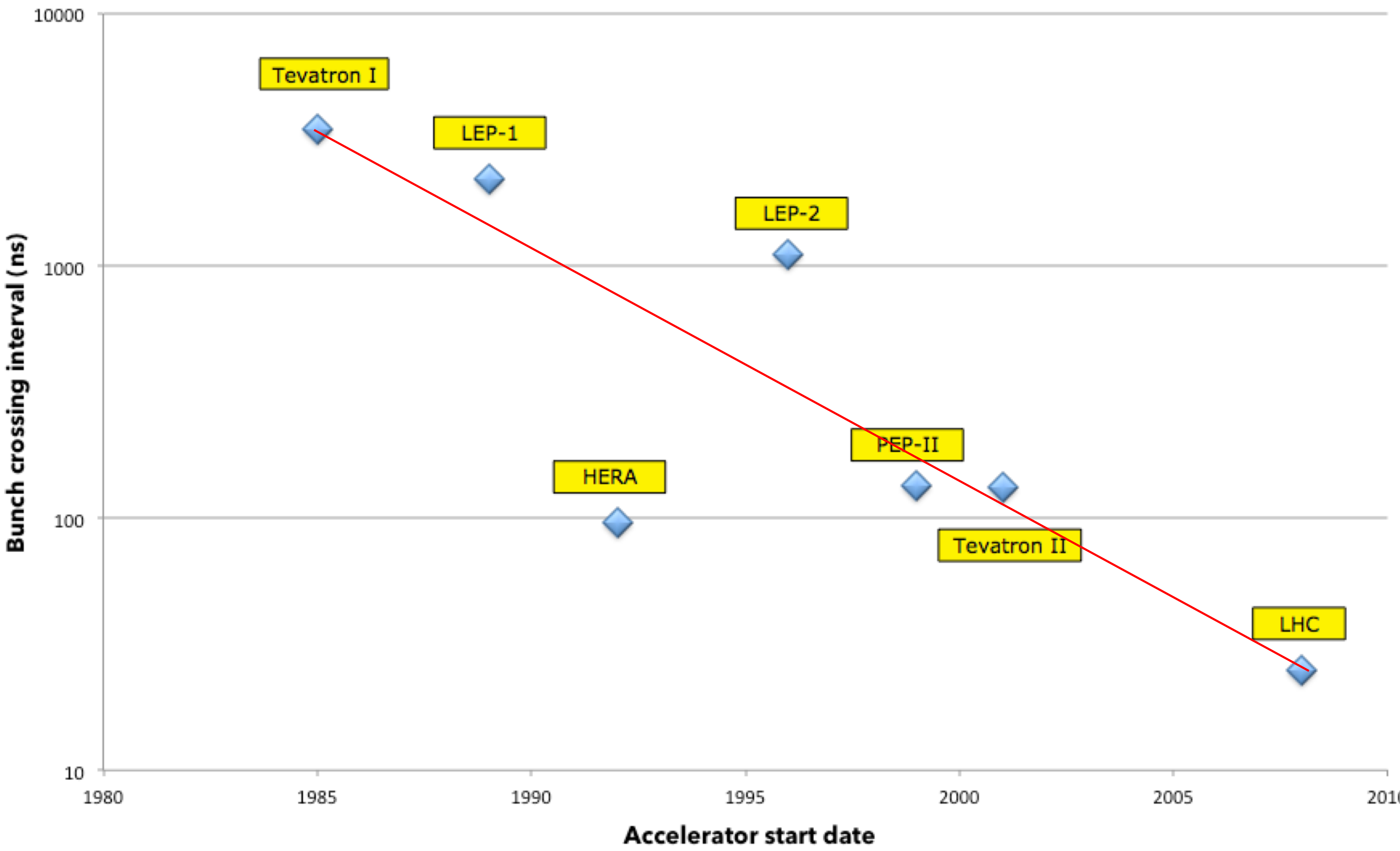


- **Here the trigger allows readout to the FIFO only for non-zero signals**
- **Processing of non-zero data at "steady" rate**

Multi-level triggering

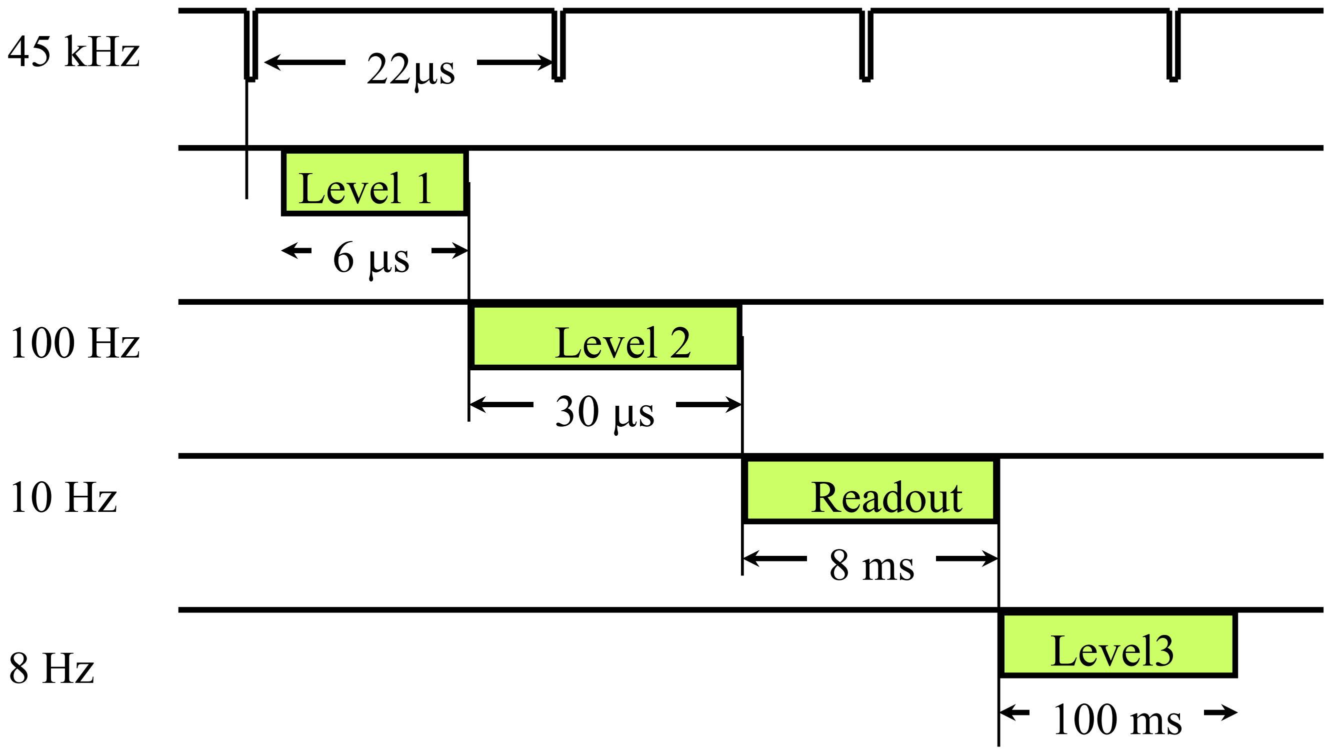
- Collider experiments need high rejection (~99.999% at LHC)
  - But don't lose valuable physics!
- Not possible with a single trigger
  - "Selective" algorithms need <u>full-resolution</u> detector data, which is <u>too much</u> to analyse at 40M BX/s
- Solution: Multi-level triggers
  - Level-1 trigger (high rate)
    - Simple algorithms on "coarse resolution" data
    - Typically implemented in custom hardware
  - High-level trigger(s)
    - More analysis-like algorithms on (fewer) events that pass the previous level(s)
    - Typically implemented in CPU farms

Collider BX rate evolution



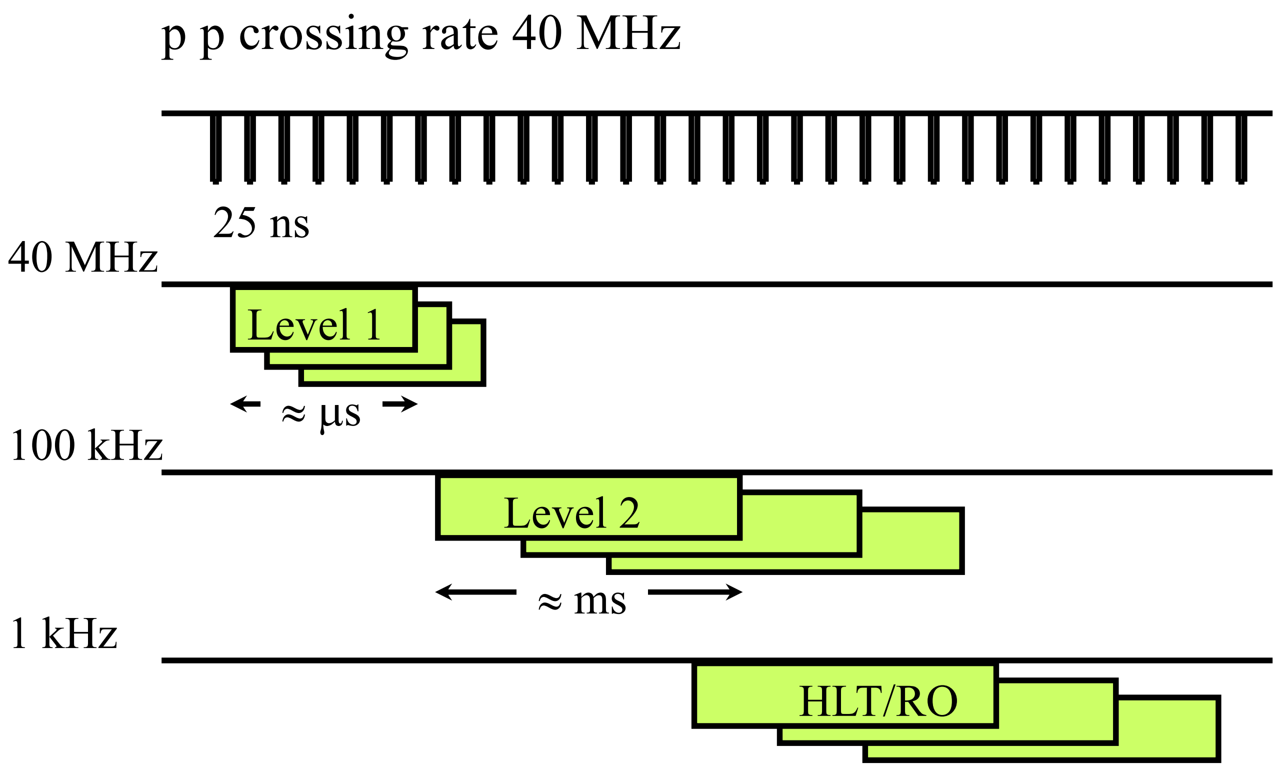
Trigger/DAQ at LEP

e^+e^- crossing rate: 45 kHz (LEP-1)



- **Level 1 trigger latency less than BX interval –> No dead time**
- **No overlapping events at any trigger level**
- **Most trigger/DAQ electronics located off-detector**

Trigger/DAQ at LHC

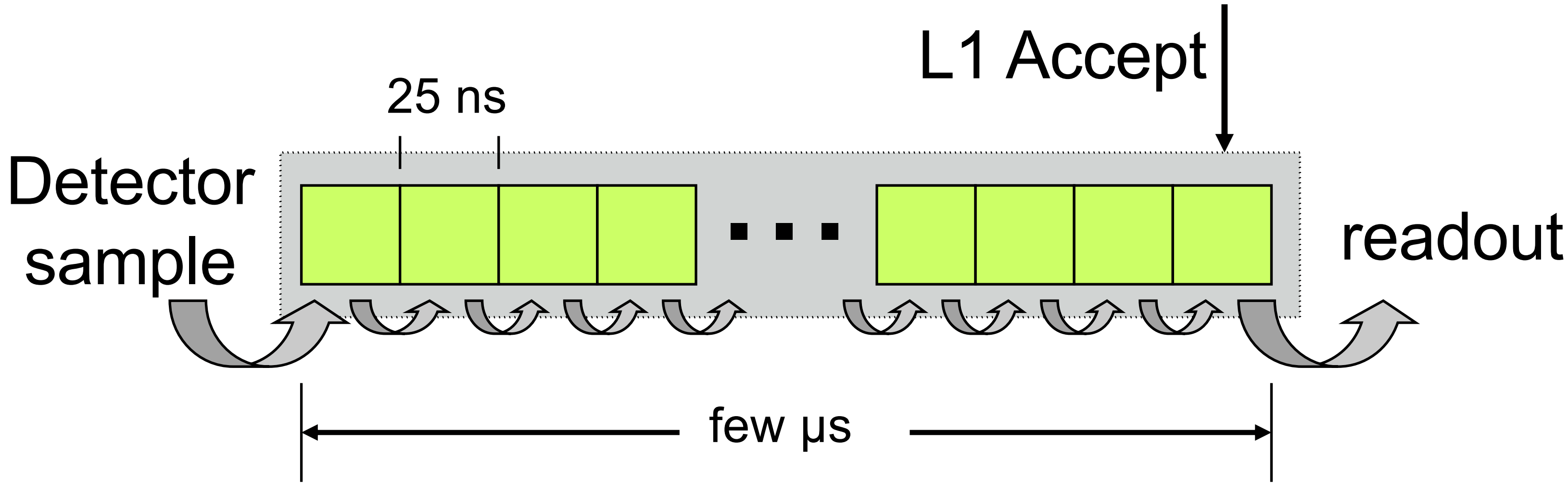


- Level 1 trigger latency >> bunch interval
- Up to 10 μs for complete readout of an event
- Need <u>pipelined</u> trigger and DAQ

Pipelined trigger

- For every bunch crossing:
  - Sample and store all detector data in fixed-length pipeline buffers (~few μs)
  - Send reduced-granularity data to the Level-1 trigger over a low-latency path
  - Level-1 does pipelined processing
    - Many consecutive stages
    - Produces a Level-1 accept decision (L1A) for every BX
- If an L1A is issued for an event:
  - Extract data for that event from the end of the pipeline buffers and read it out to DAQ

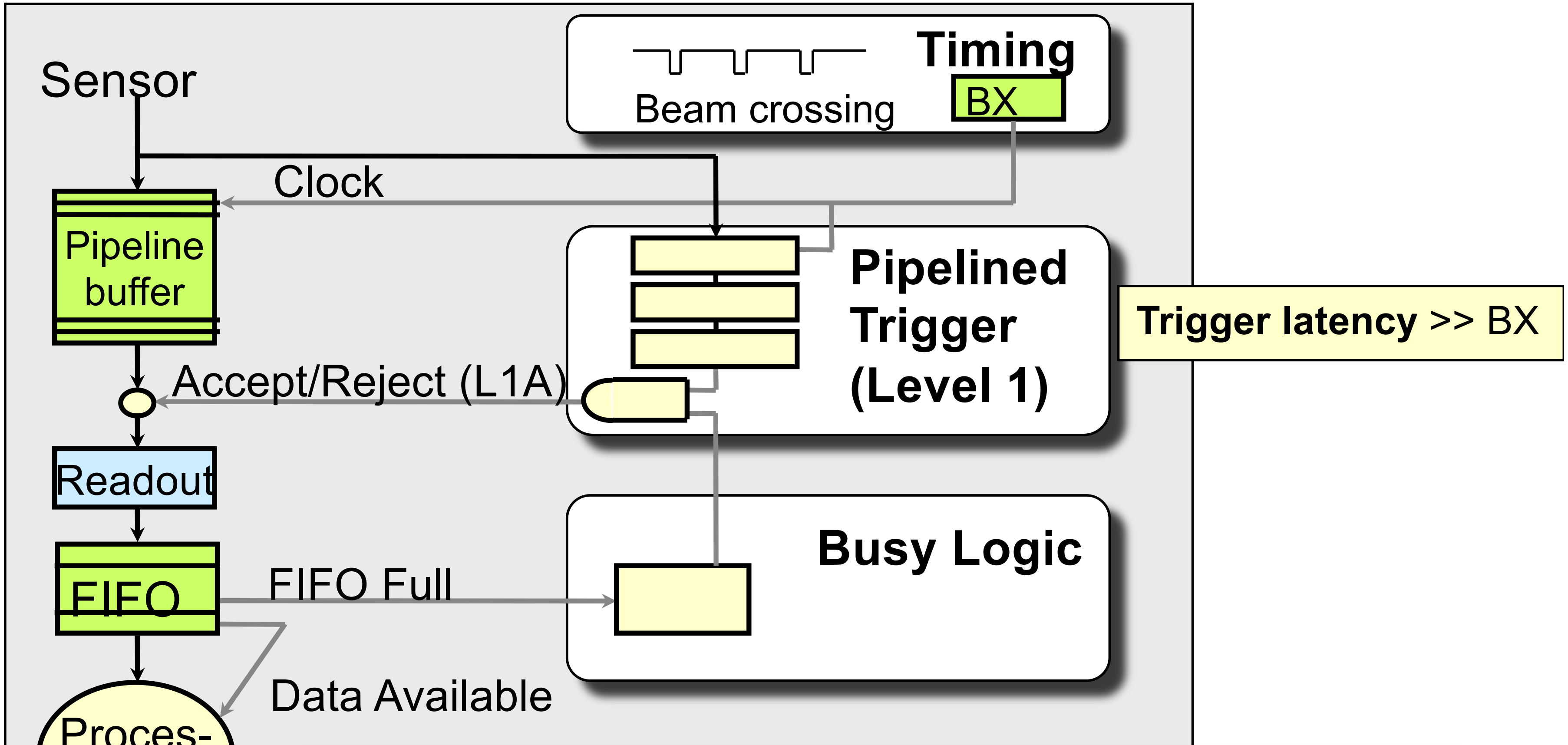
Pipeline buffer



New data enters the buffer every BX

Keep event at end of pipeline if L1A received

Simple pipelined trigger/DAQ

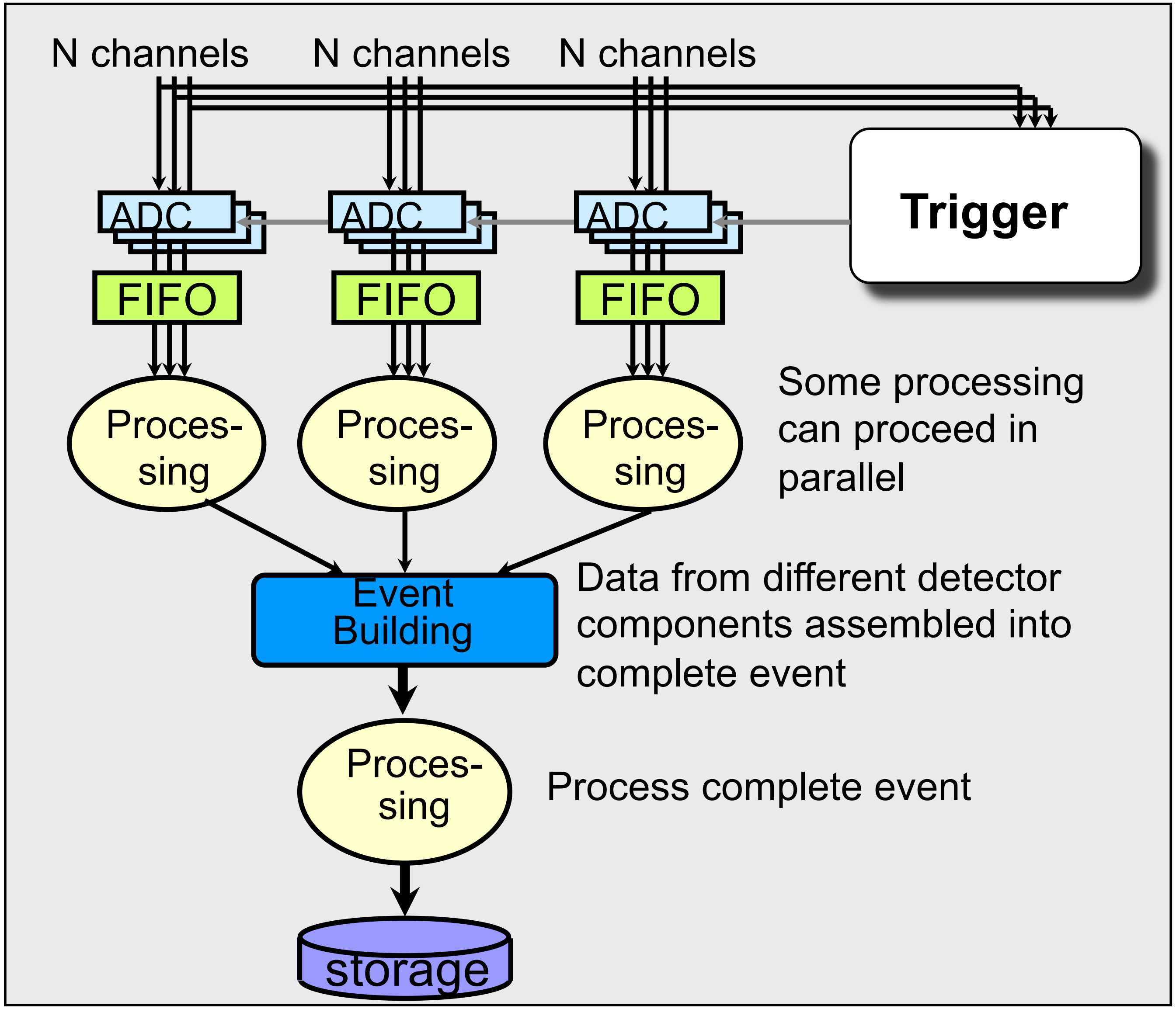


- Sensor data stored in pipeline buffer, awaiting trigger decision. Only "interesting" events are selected.
- Level-1 trigger algorithm has higher latency than BX period, so executed in many "pipelined" steps.

TDAQ for multiple systems

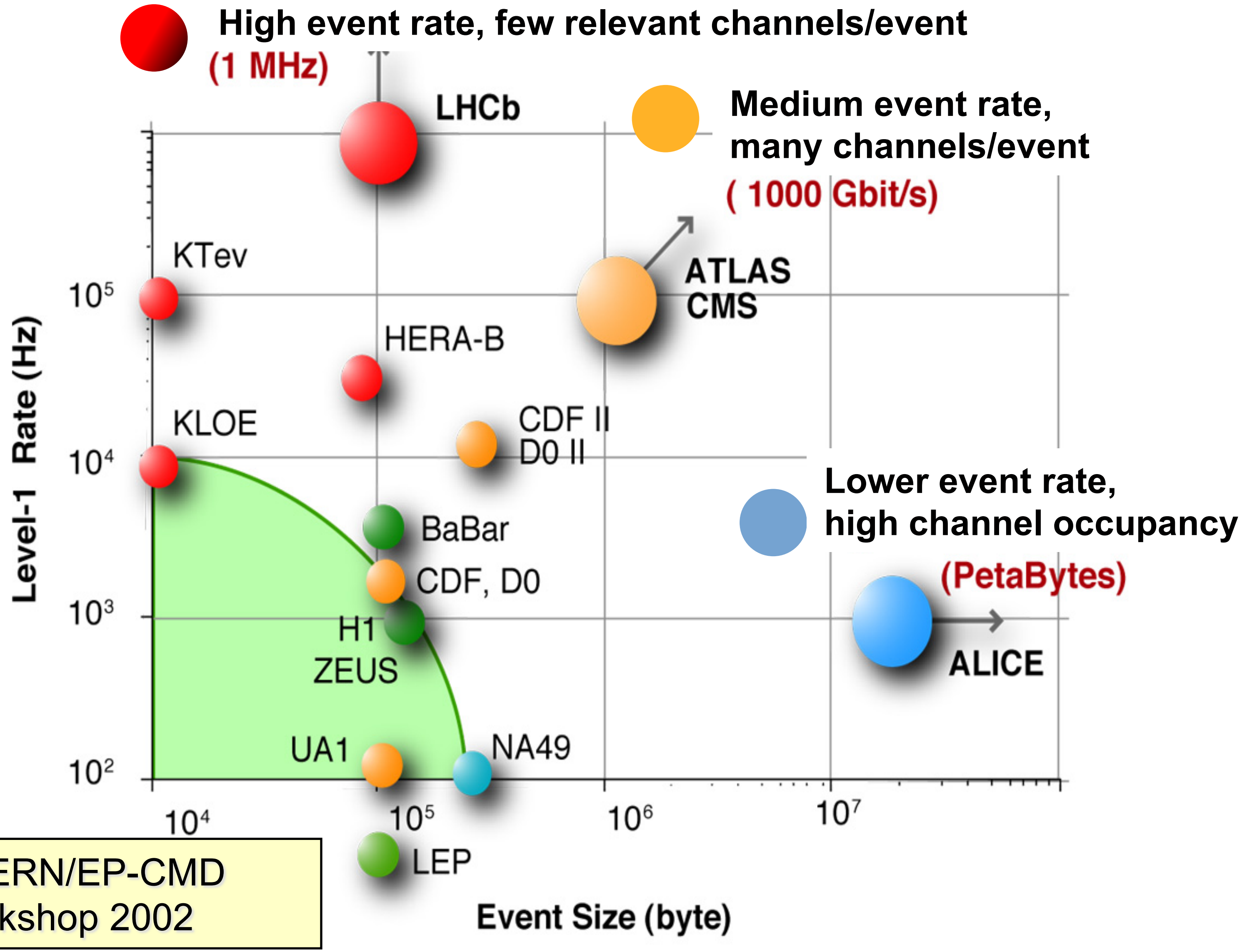
- Collider detectors are collections of detector systems working together
  - Tracking detectors
  - Calorimeters
  - Muon spectrometers
- A single Central trigger initiates readout for all detectors
- DAQ collects and records data from all detectors for each triggered event

Multi-system Trigger/DAQ



Trigger/DAQ architectures

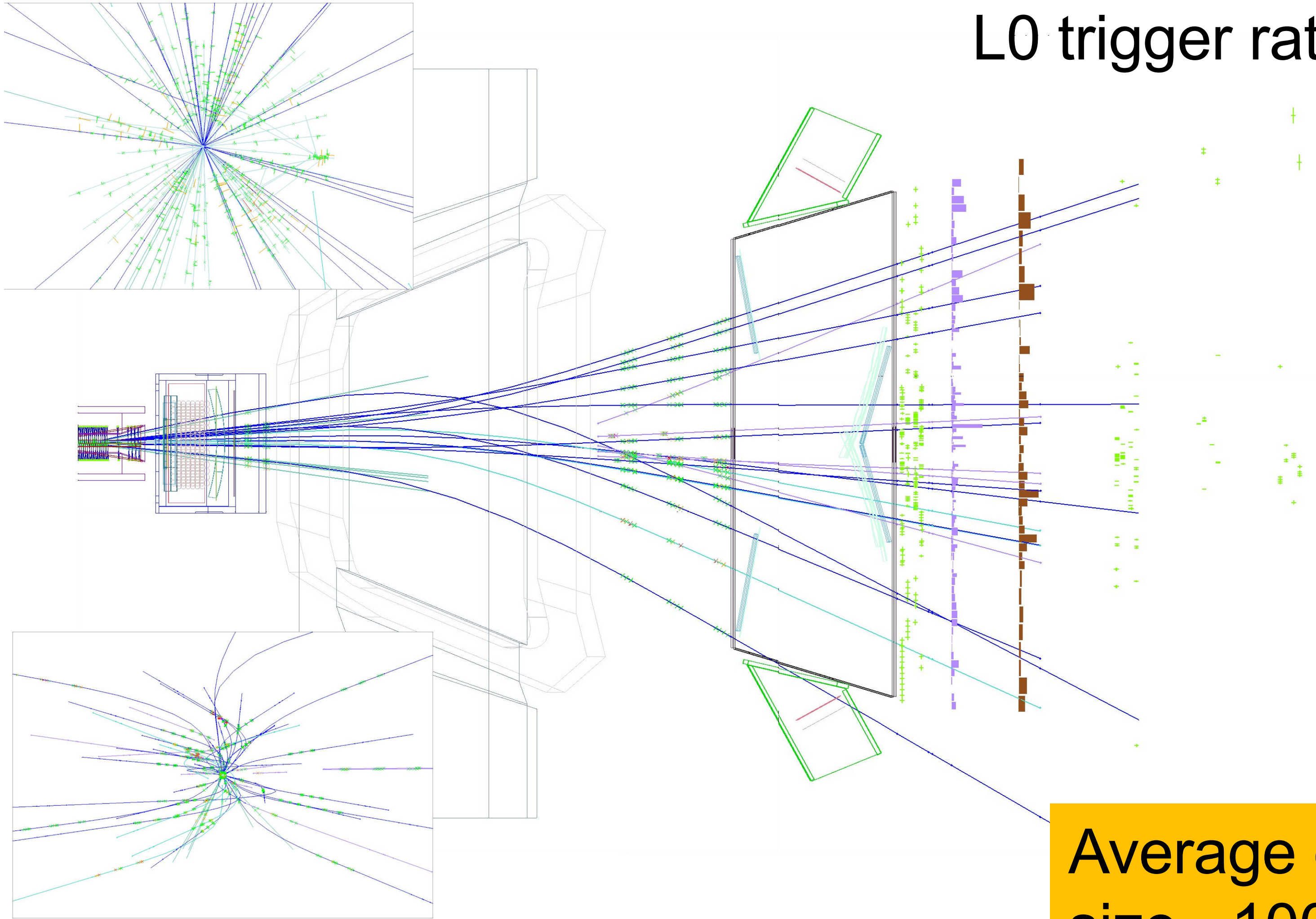
Trigger/DAQ requirements



S. Cittolin CERN/EP-CMD
LECC Workshop 2002

LHCb: High rate, small events

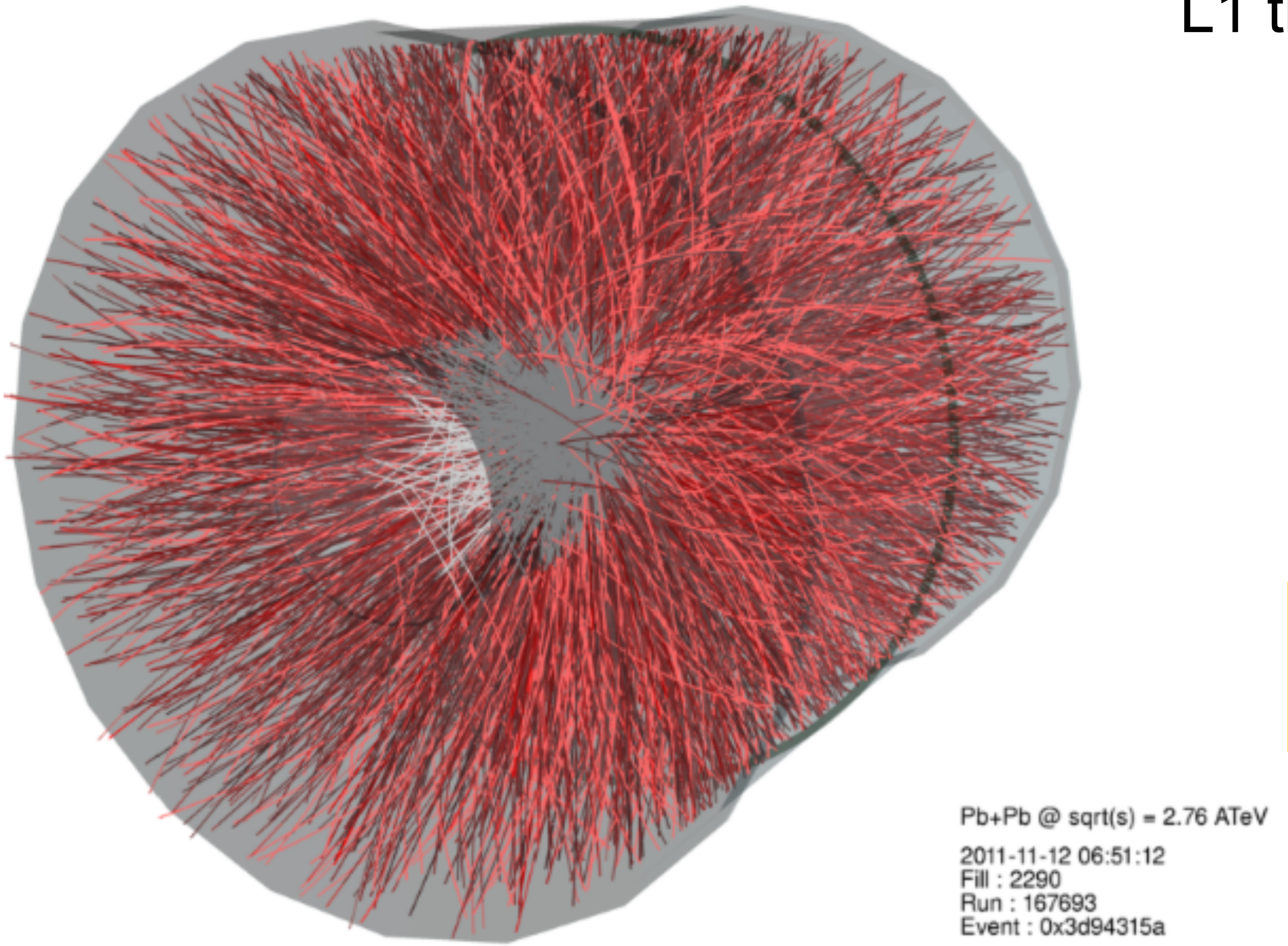
L0 trigger rate: 1 MHz



Average event size ~100kB

ALICE: Low rate, large event

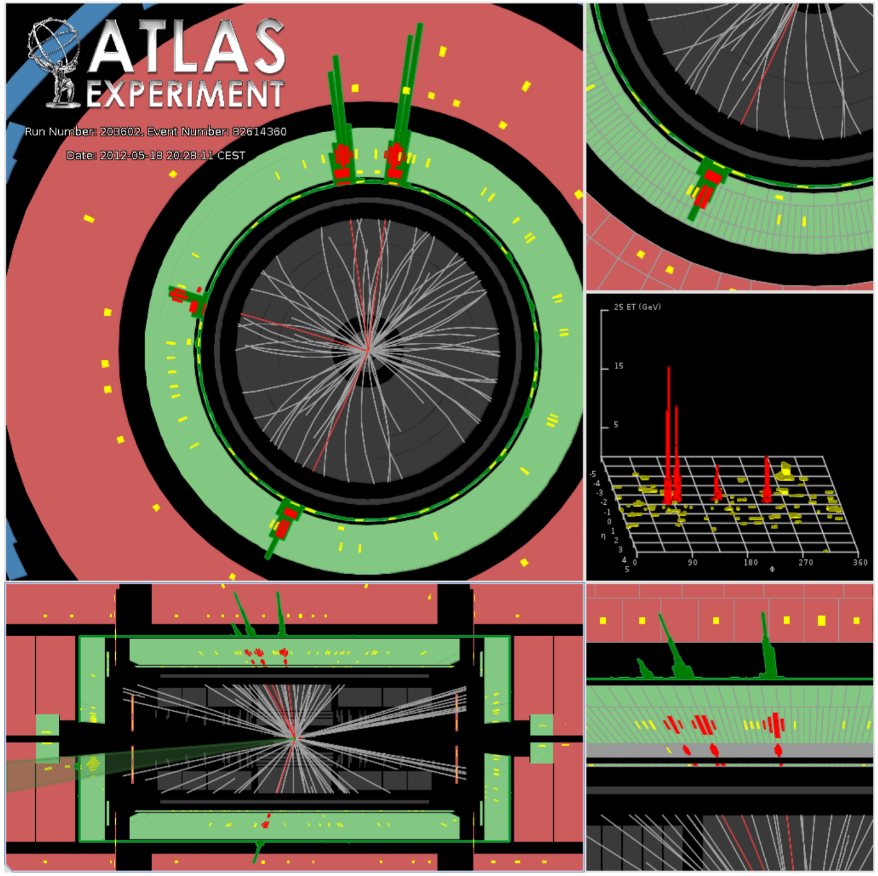
L1 trigger rate: 8 kHz

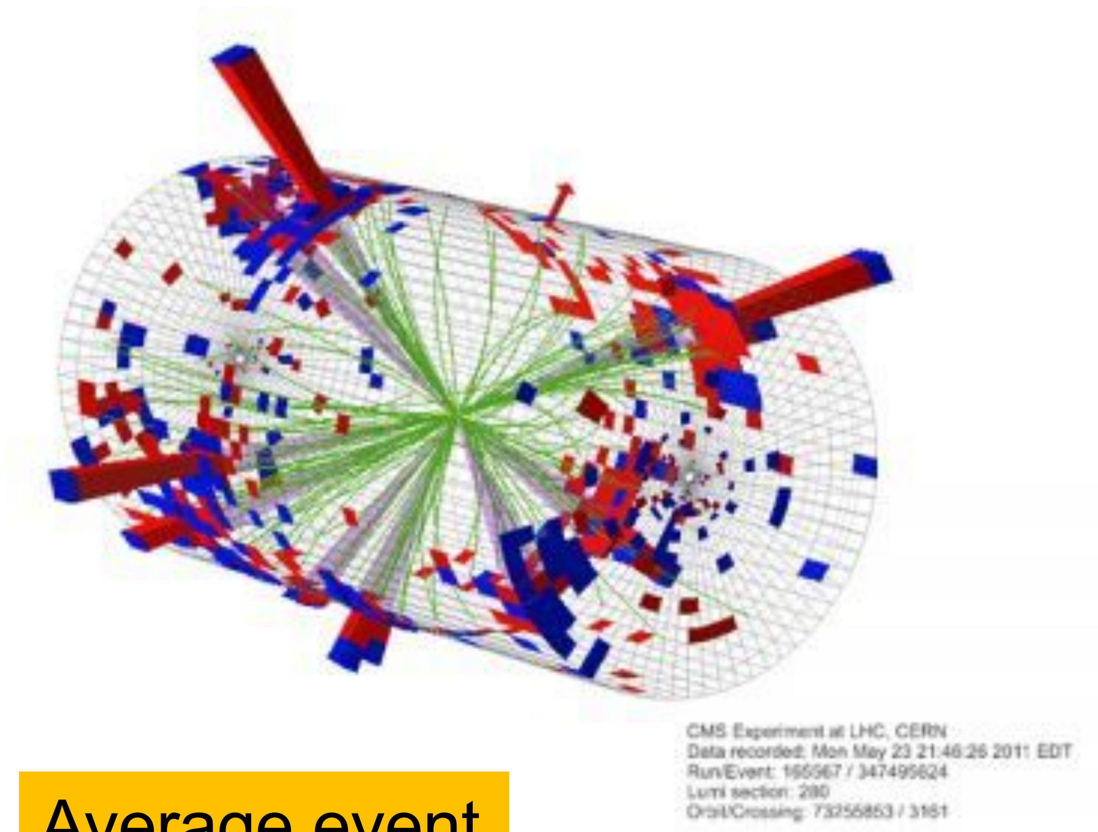


Average event size > 40 MB

ATLAS and CMS

L0 trigger rate: 75-100 kHz





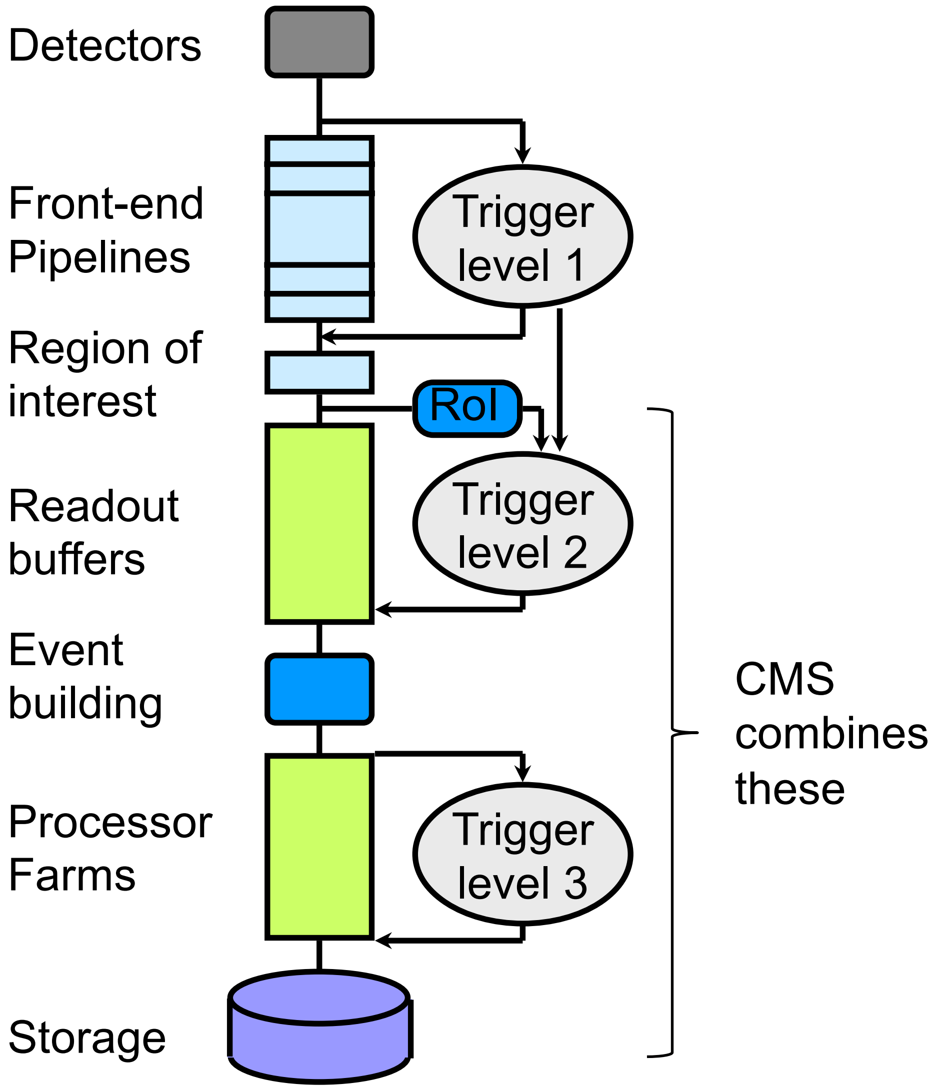
Average event
Size ~ 1 MB

Different DAQ approaches

- Lower event rate, many channels (ALICE)
  - Simpler trigger
    - Naively: “something happened” (activity)
  - Read out the entire event
- High event rate, few channels (LHCb)
  - Read out only the “active” channels
    - Move to so-called “triggerless” DAQ
- Medium event rate, many channels
  - Sophisticated, multi-level trigger
    - More “physics-like” than activity trigger
  - Read out entire event after level-1 trigger

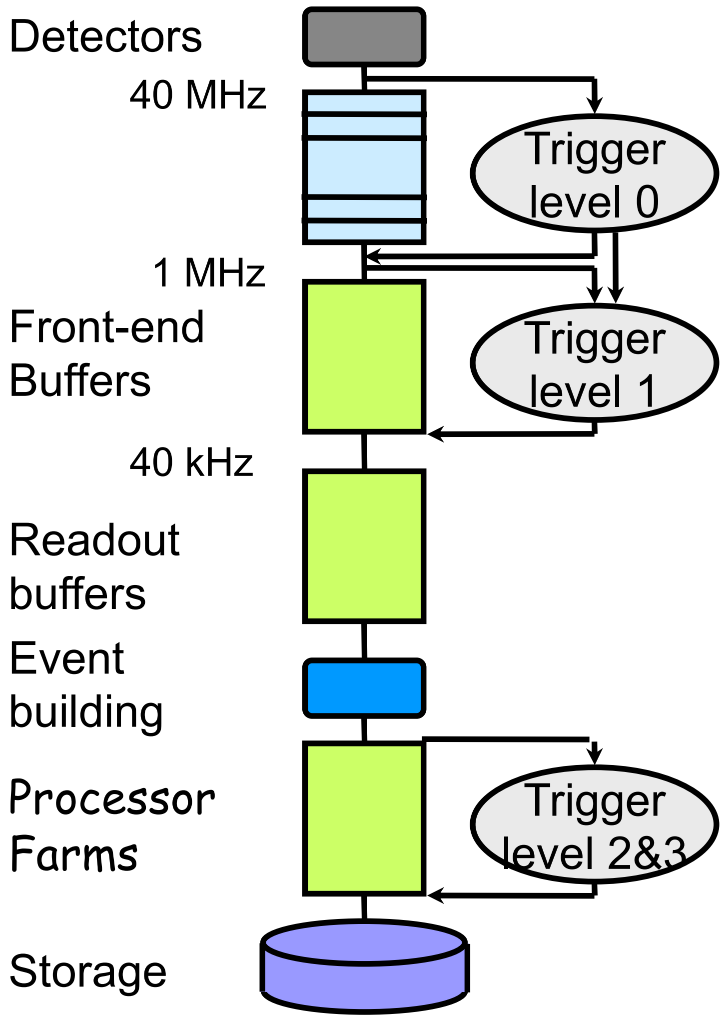
I will *mainly* focus on these… (ATLAS, CMS)

ATLAS trigger



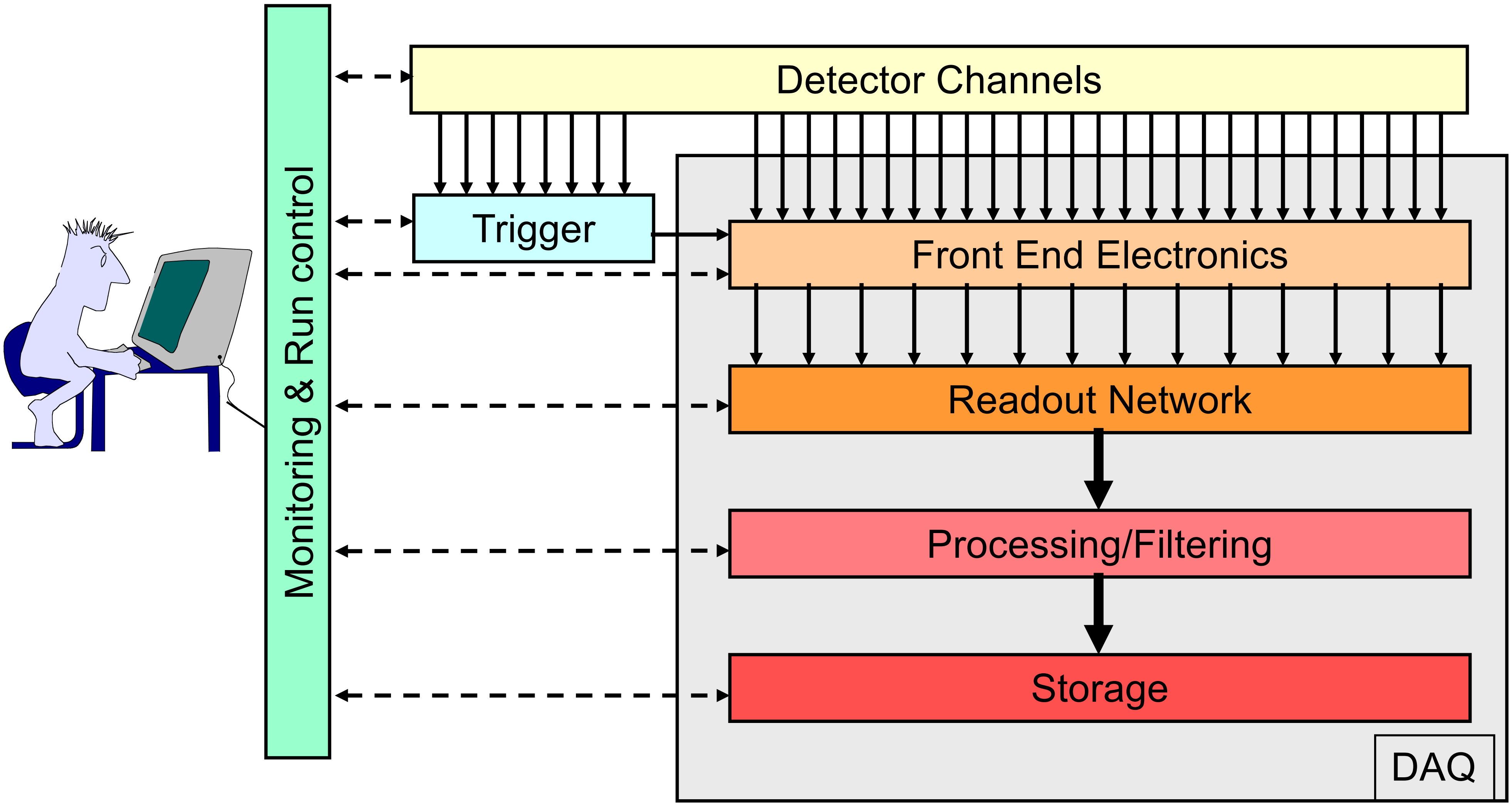
- **Level-1 (3.5 μs) (custom processors)**
  - Isolated clusters, jets, ET in calorimeters
  - Muon trigger: tracking coincidence matrix.
- **Level-2 (100 μs) (processor farm)**
  - Guided by Regions Of Interest (RoI) identified by Level-1
  - Select detector data routed to CPUs by routers and switches
  - Feature extractors (DSP or specialized) perform refined object ID algorithms
  - Staged local and global processors
- **Level-3 (≈ms) (commercial processors)**
  - Reconstruct the event using all data
  - Select of interesting physics channels

LHCb trigger system



- **Level-0 (4 μs) (custom hardware)**
  - High p_T electrons, muons, hadrons
  - Pile-up veto.
- **Level-1 (1000 μs) (specialized processors)**
  - Vertex topology (primary & secondary vertices)
  - Tracking (connecting calorimeter clusters with tracks)
- **Level-2 (≈ms) (commercial processors)**
  - Refinement of Level-1. Background rejection.
- **Level-3 (≈ms) (commercial processors)**
  - Event reconstruction. Select physics channels.

TDAQ and run control



Putting it all together…

Front-end electronics and signal processing

Front end electronics

- Input <u>conditioning</u>
  - Amplifiers, shapers, integrators...
  - Convert detector input signals to a form useful for the trigger & readout
- <u>Sampling</u> and digitization (ADC)
- Buffering and readout
- Signal processing (for trigger)
  - Amplitude
  - Timing

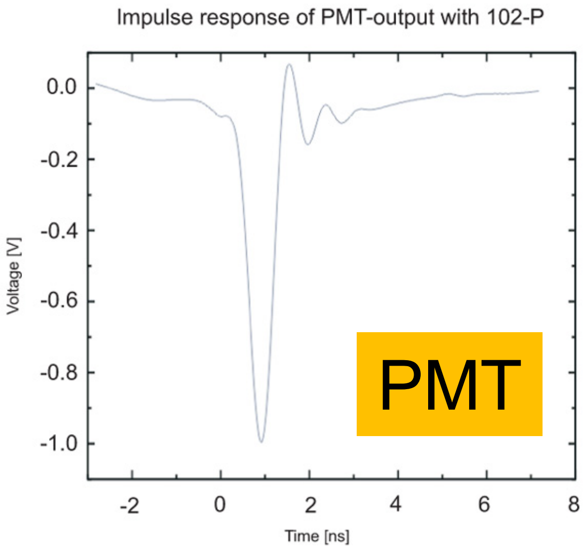
Closely related, often on-detector (Input conditioning; Sampling and digitization)

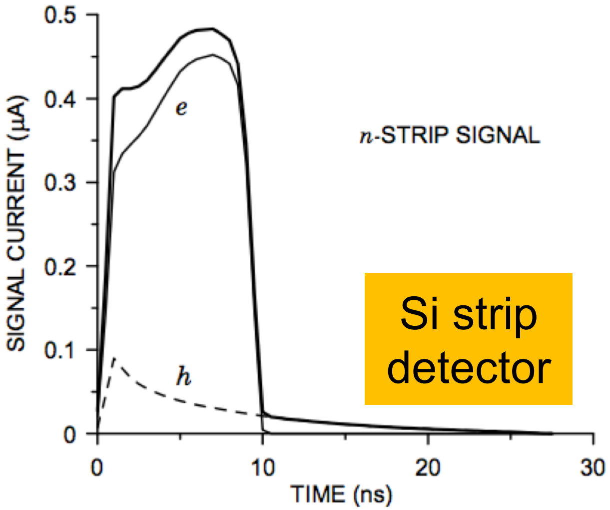
Beginning to migrate off-detector (Buffering and readout)

Usually off-detector (Signal processing)

Also: calibration, monitoring, and other services

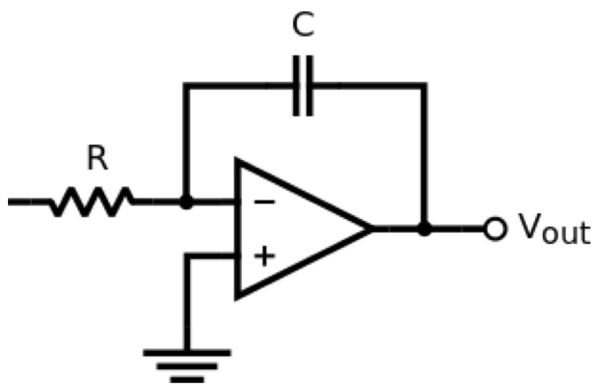
Input conditioning/sampling



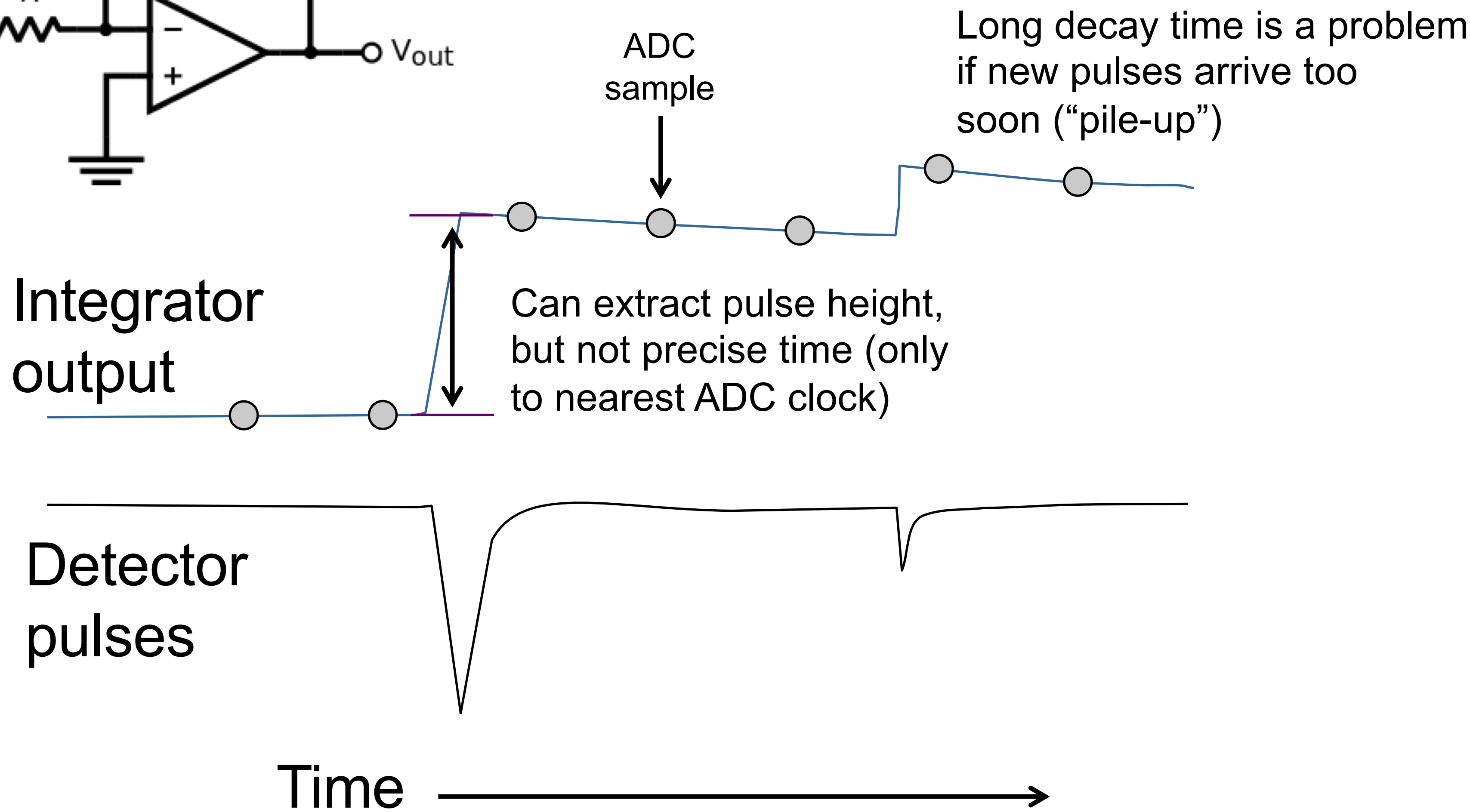


- Raw pulses can be fast and have complex shapes
  - Would need fast ADCs to directly measure them (expensive, power-hungry, low dynamic range)
- A solution is <u>pulse shaping</u>
  - Convert fast pulses into a slower, well-defined shape
  - Ideally amplitude-independent
  - Match to affordable ADC with the desired dynamic range (# bits)

Simple charge integrator

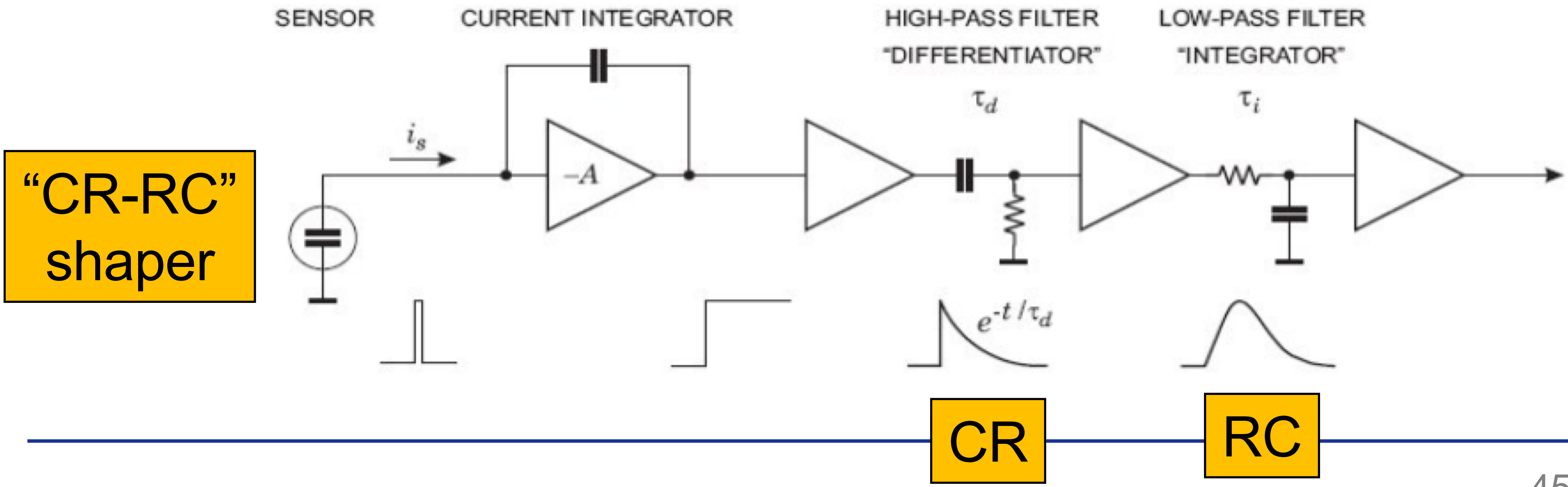


(Simplest pulse shaping circuit)



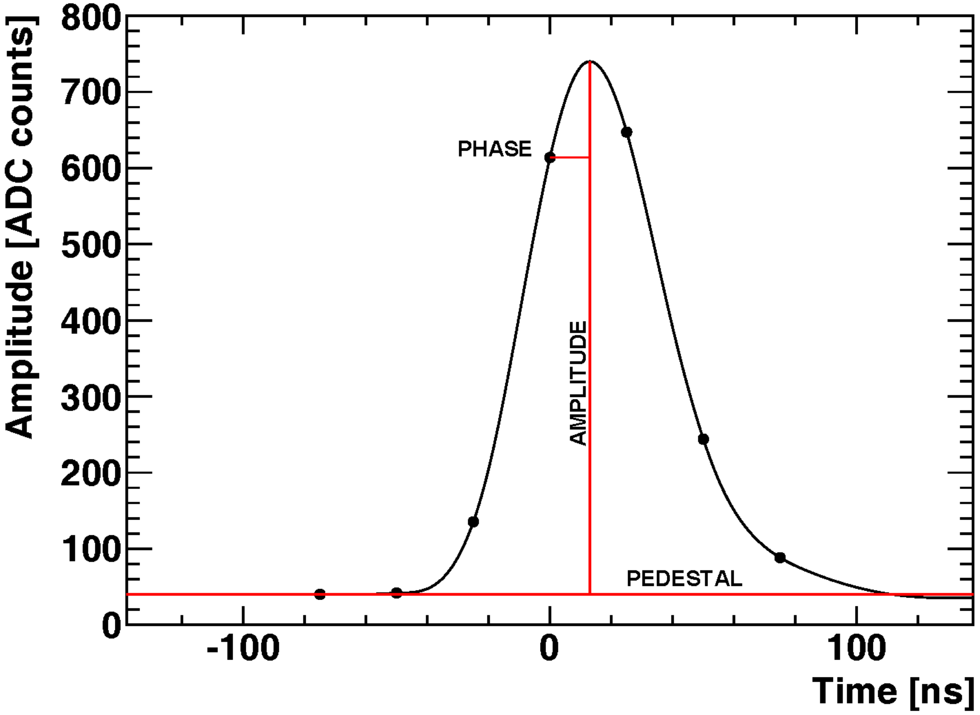
Differentiator/integrator

- Commonly-used shaper in particle detectors
- Differentiator (high-pass filter)
  - Maximum amplitude of shaped pulse
  - Pulse duration (decay time)
- Integrator (low-pass filter)
  - Slows down rise-time of pulse



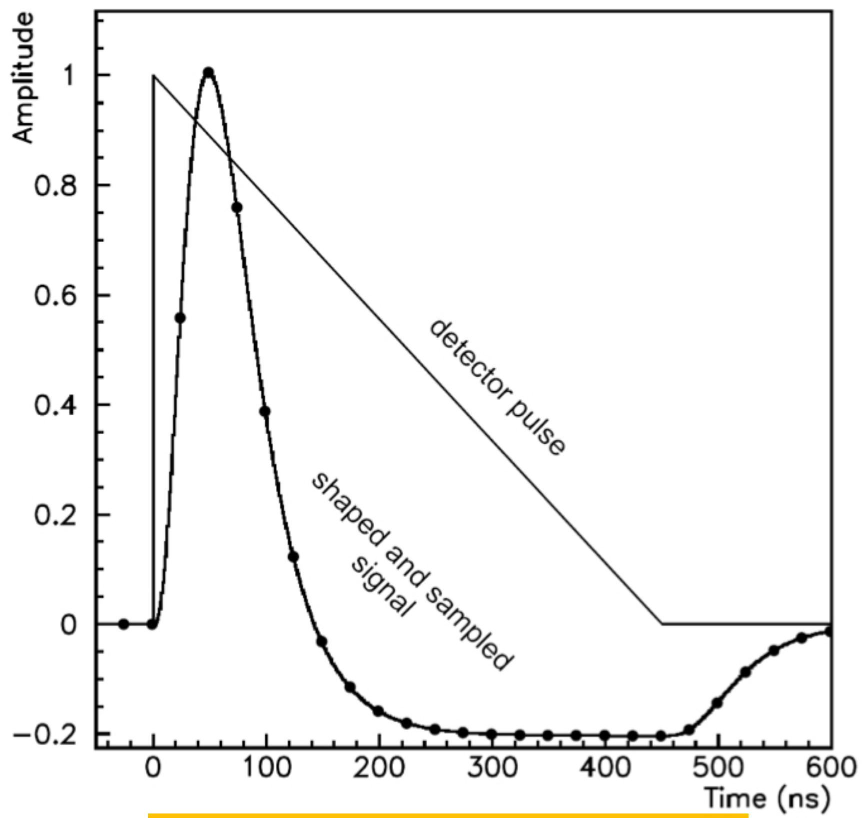
ATLAS calorimeter shapers

Tile (hadronic) calorimeter



Unipolar shaper, makes a short PMT pulse longer

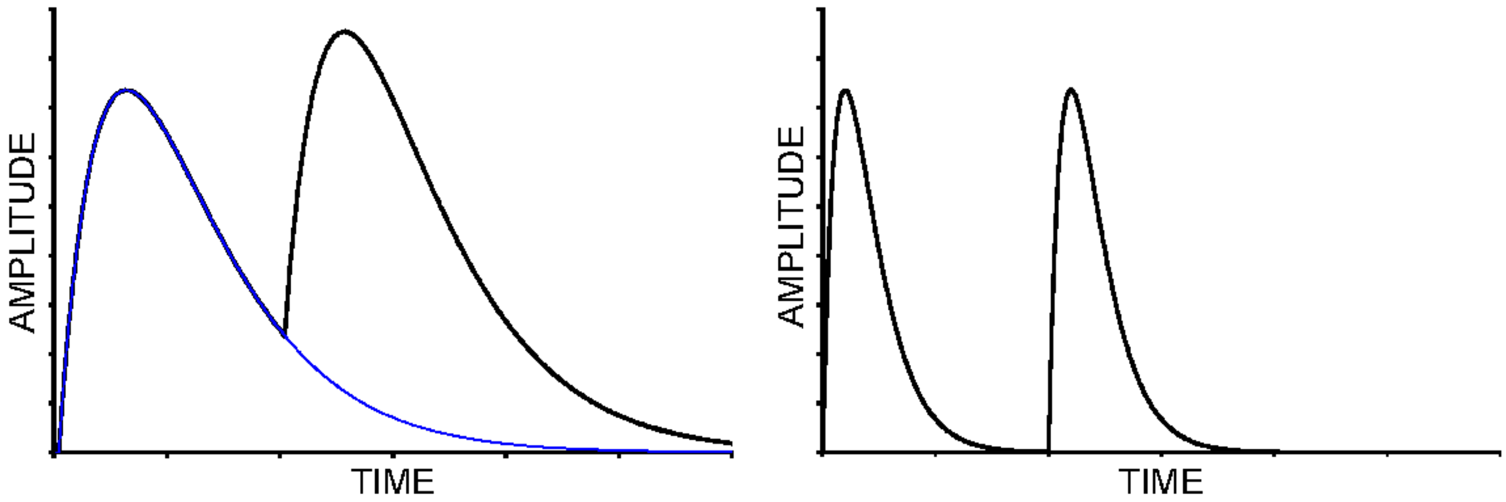
LAr (EM) calorimeter



Bipolar shaper, shortens a long ionization signal

Pile-up

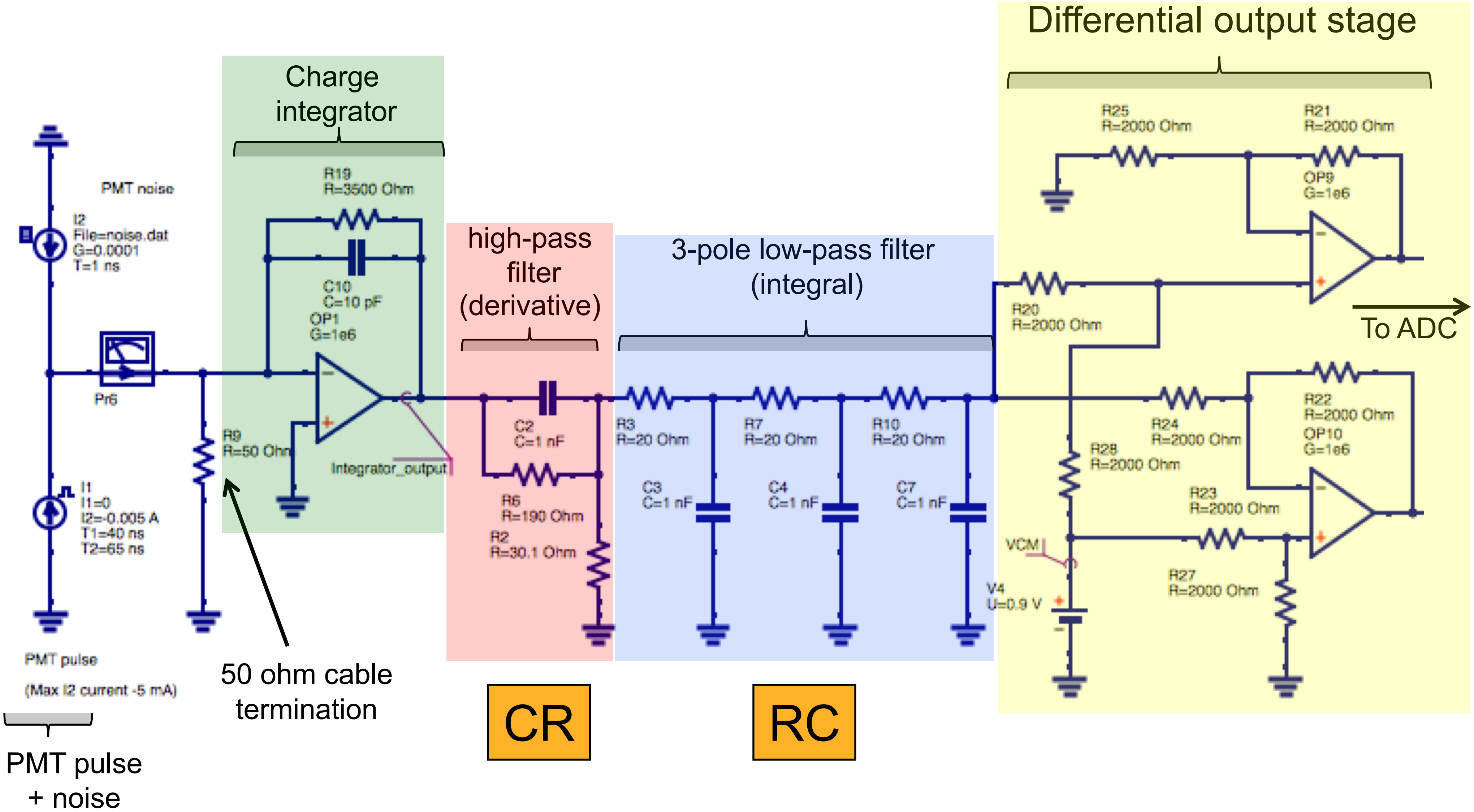
- Broad pulses good for digital processing
  - Better amplitude and timing estimates
  - Less sensitive to random nose
- But too broad pulses increase the pile-up rate
- Need to <u>compromise</u>



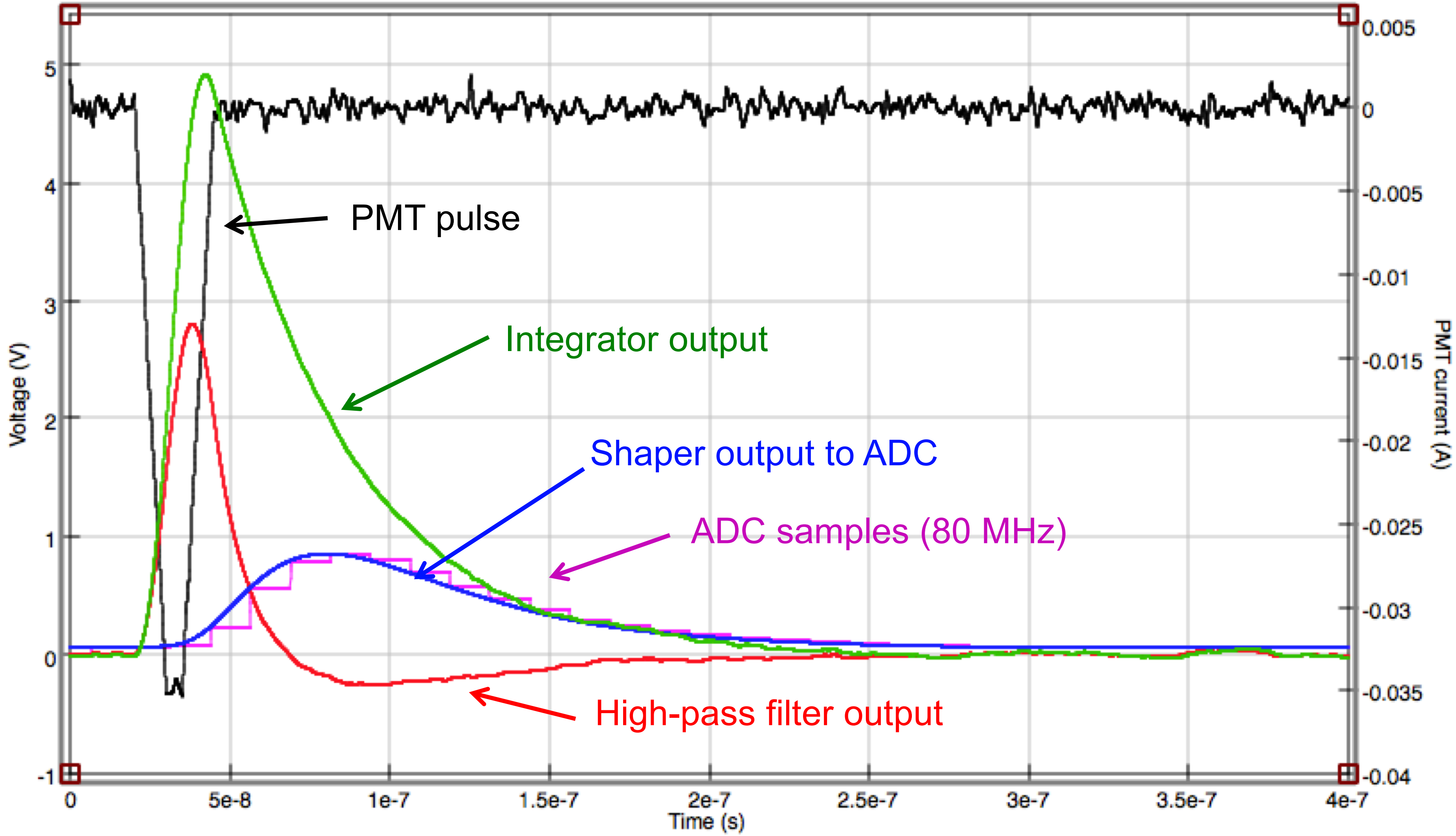
Case study

- The following is a simulation of a slow shaper idea for a hadronic calorimeter
- Design constraints:
  - Fast pulse (PMT)
    - Unipolar shaper (CR-RC)
  - Want a large dynamic range
    - Assumed a 16-bit ADC (13.5 effective bits) with up to 80 MHz sampling rate
  - Low channel occupancy (low pile-up)
    - Pulse can be slow (~150 ns)
    - Oversampling increases # of effective bits

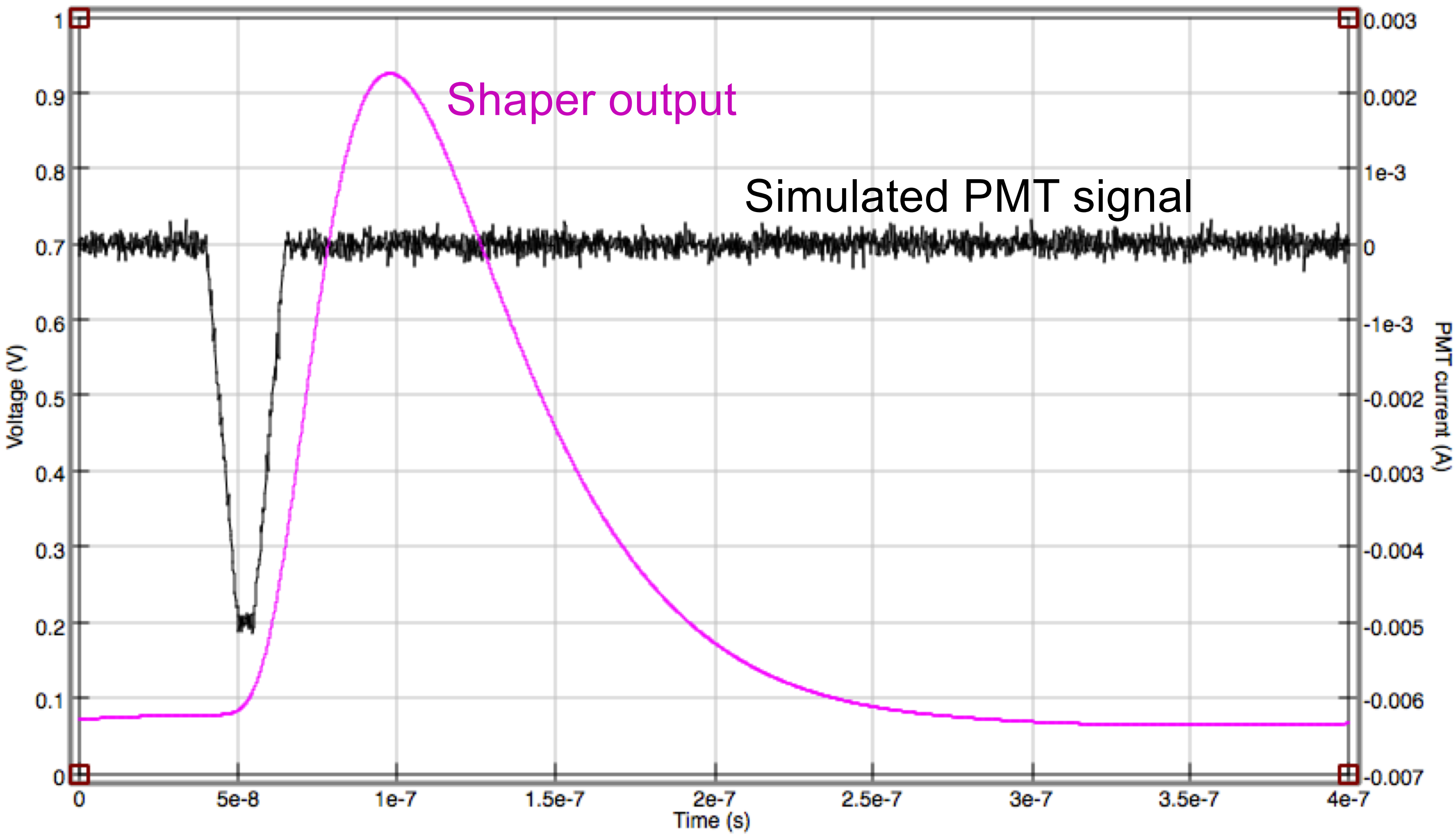
Analog input and shaper



Transient simulation

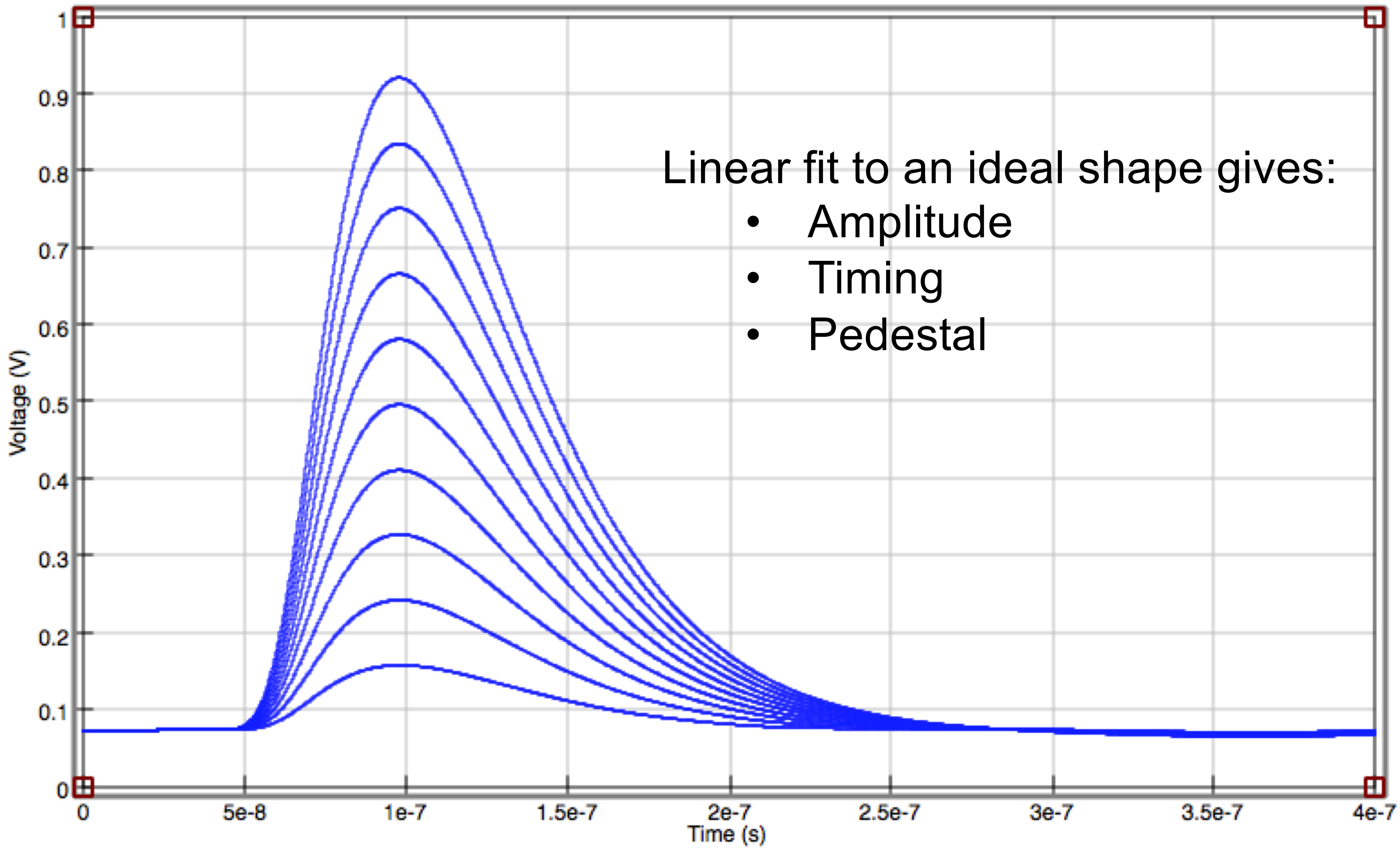


PMT input signal shaping



Shaper output

Amplitude-independent

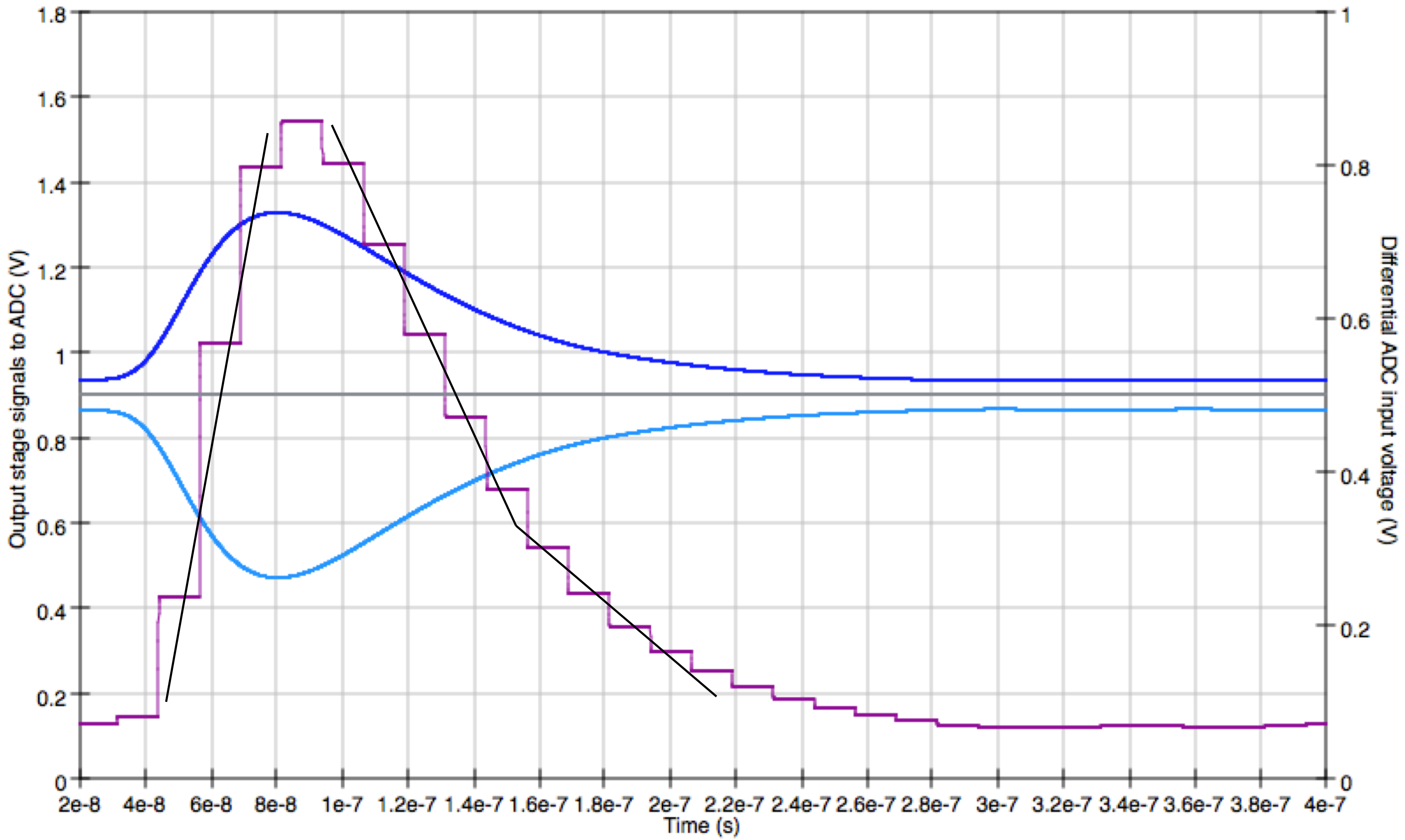


Linear fit to an ideal shape gives:

- Amplitude
- Timing
- Pedestal

Sampled shape

Multiple samples on rising and falling edges give good timing estimates



Signal processing

Signal processing

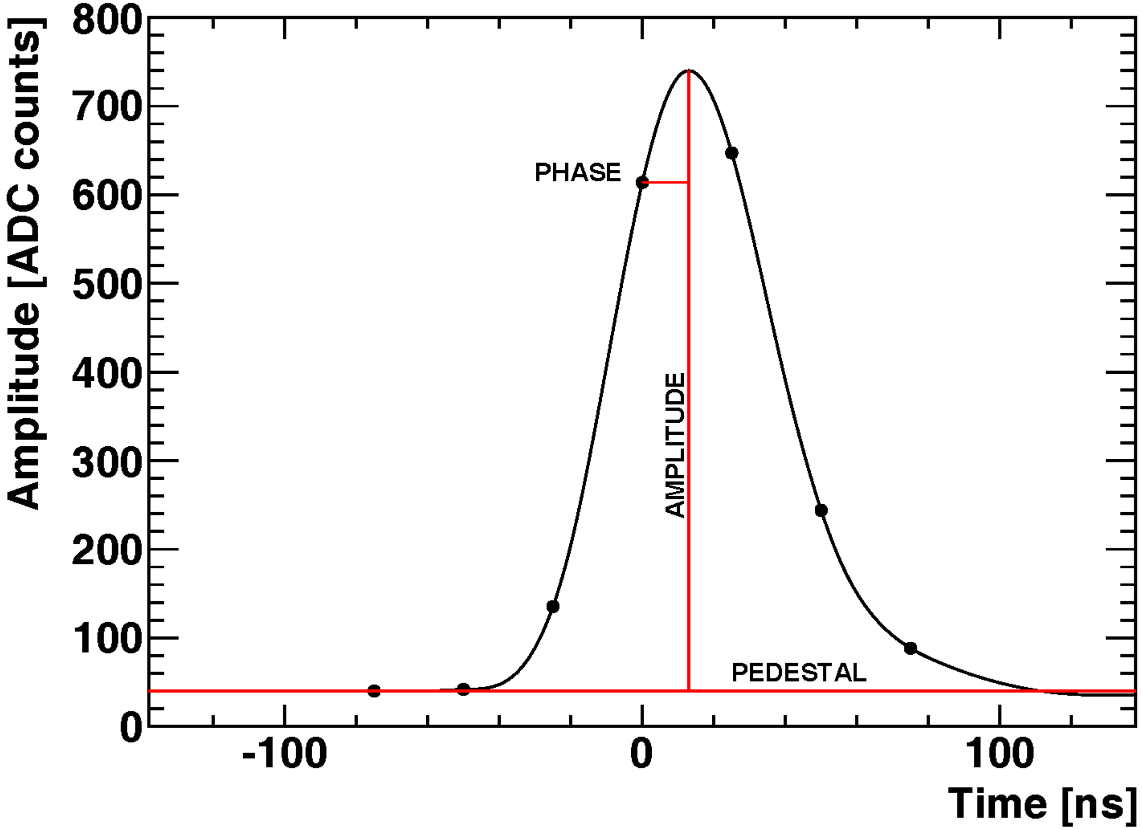
- From sampled signal, need to extract:
  - Pulse amplitude
  - Timing of the pulse
    - Coarse timing (which BX?)
    - Fine timing (ns level)
      - Good for eliminating some backgrounds
- Common approach: digital filter
  - There are different approaches
  - I will show finite-impulse-response (FIR) filter with matched coefficients…

Finite Impulse Response (FIR)

- Process data with finite length
  - Fixed-width “window” of data points
  - For example: detector pulses measured in a few consecutive ADC samples
- A FIR filter output is a weighted sum of the data points in the window
- FIR output is independent of data outside the window
  - No “memory” of earlier iterations
  - FIR filters are simple to implement and inherently stable

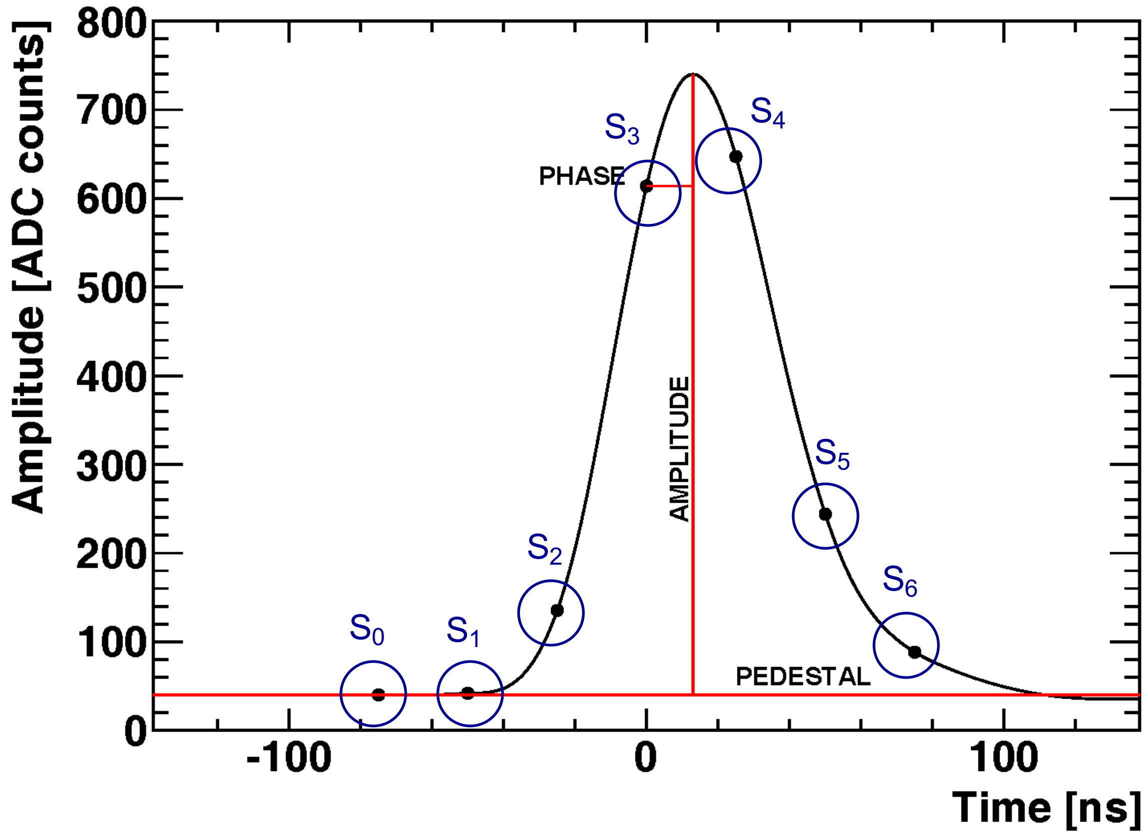
Example: calorimeter pulse

- ATLAS hadronic Tile calorimeter
- Unipolar pulse (Only positive amplitude)
- Width ~150ns
- 40 MHz ADC (5-6 samples)



Applying a FIR filter

- "Sliding window" of 7 samples: Sn
- Each sample multiplied by coefficient Cn
- Sum products to produce the filter output:



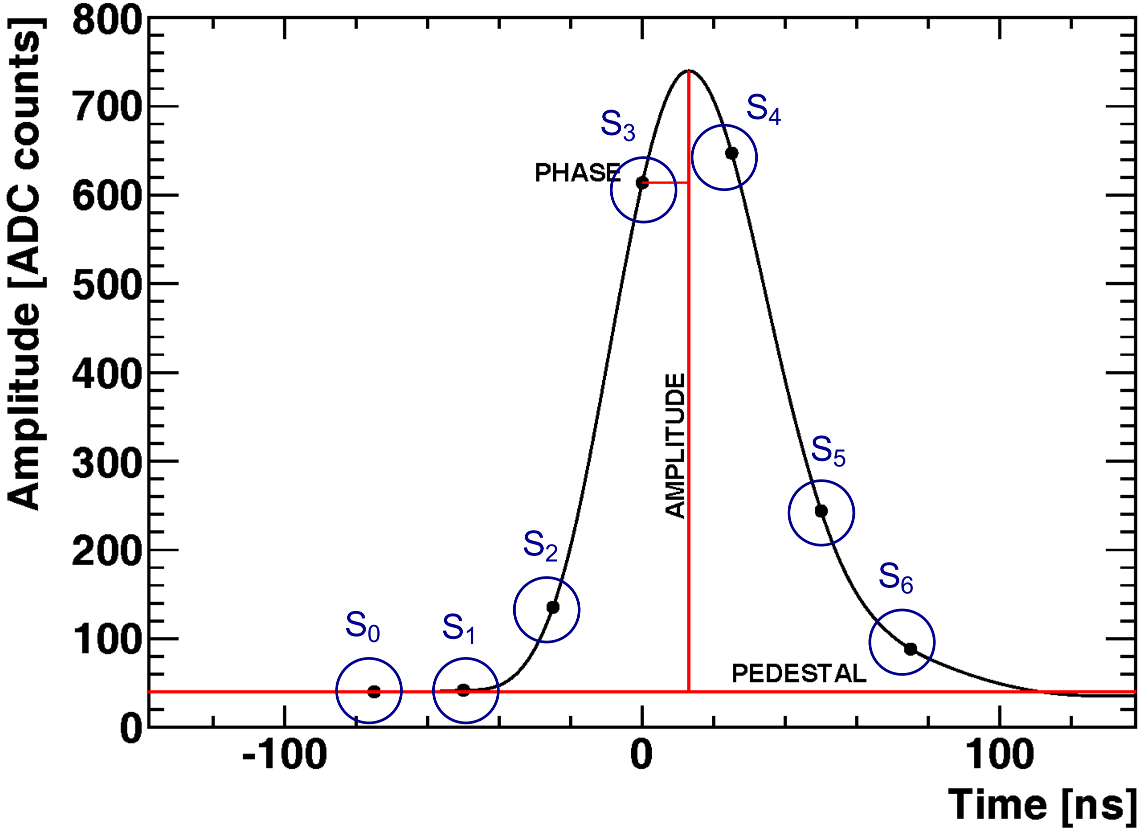
$$F = S_0 C_0 + S_1 C_1 + S_2 C_2 + \cdots + S_6 C_6$$

Coefficients C_n determine the filter response

Example: pulse integral

- Samples have equal weights
  - “Area under the curve”
- Filter coefficients Cn:

Sample	Weight
S_0	0
S_1	1
S_2	1
S_3	1
S_4	1
S_5	1
S_6	1

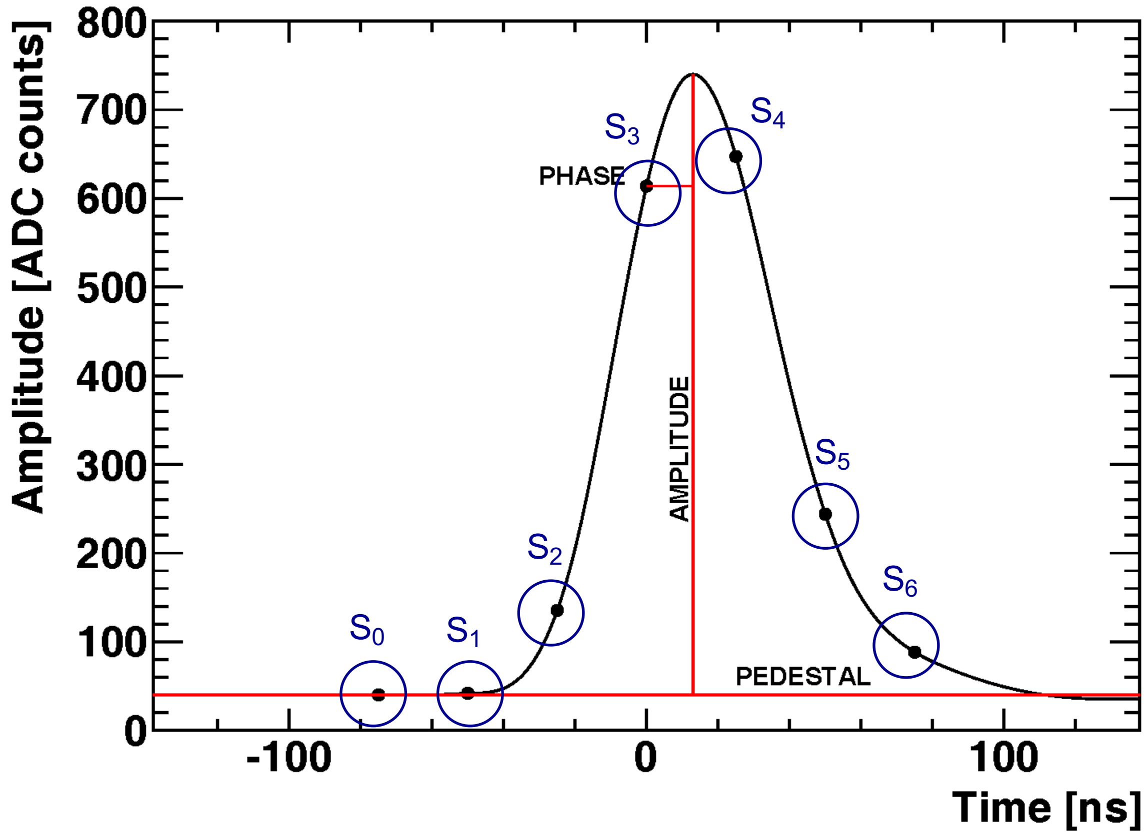


$$F = S_1 + S_2 + \cdots + S_6$$

Integral w/ pedestal subtraction

- Subtract pedestal (S0) from the other six samples.
- Filter coefficients Cn:

Sample	Weight
S_0	-6
S_1	1
S_2	1
S_3	1
S_4	1
S_5	1
S_6	1

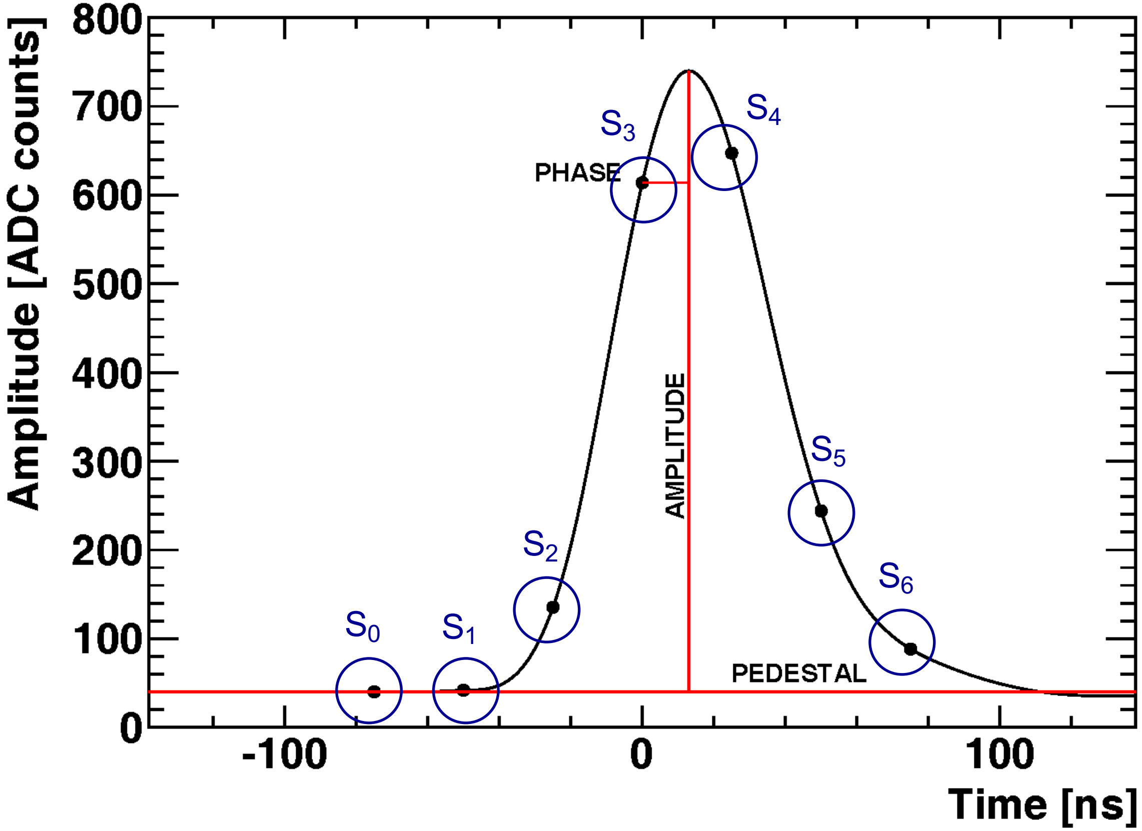


$$F = (S_1 + S_2 + \cdots + S_6) - (6 \times S_0)$$

Peak amplitude

- Use only the maximum sample
  - For example: S4
- Filter coefficients Cn:

Sample	Weight
S_0	0
S_1	0
S_2	0
S_3	0
S_4	1
S_5	0
S_6	0

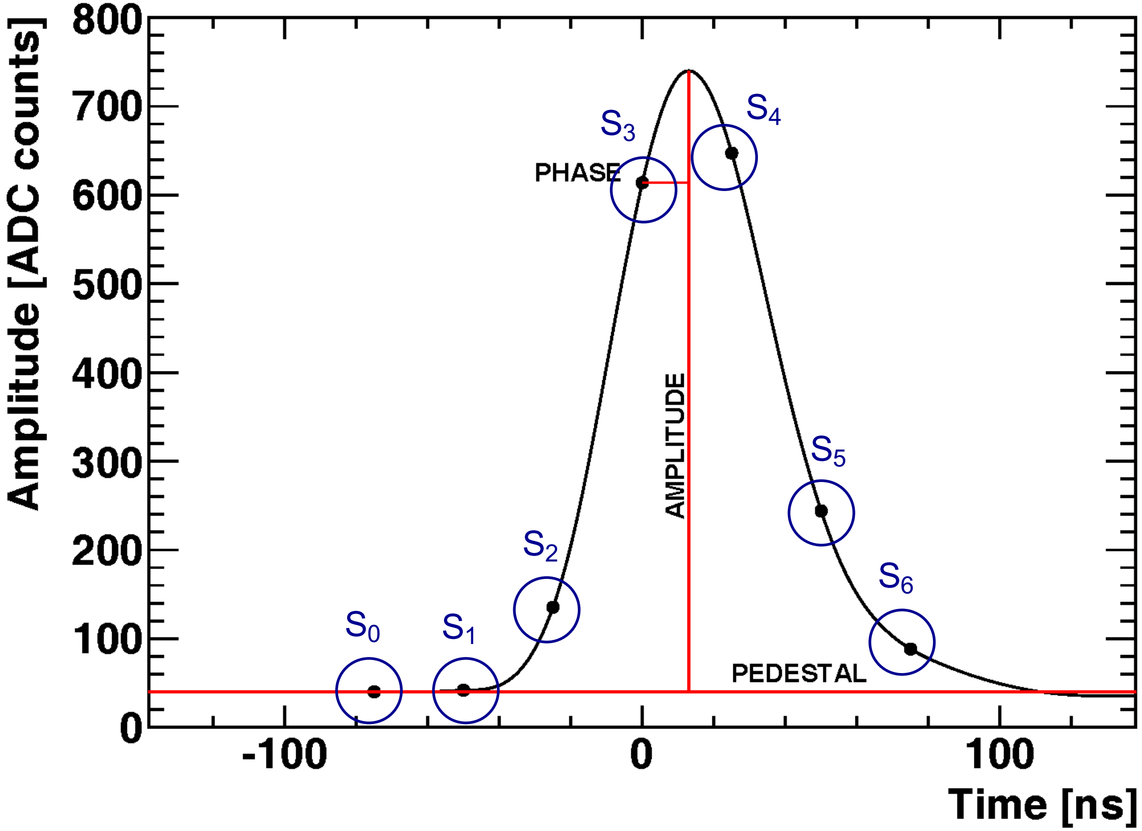


$$F = S_4$$

“Optimal filter”

- Match coefficients to the ideal pulse shape (with noise)
- Gives best resolution and signal/noise performance
- Filter coefficients C_n:

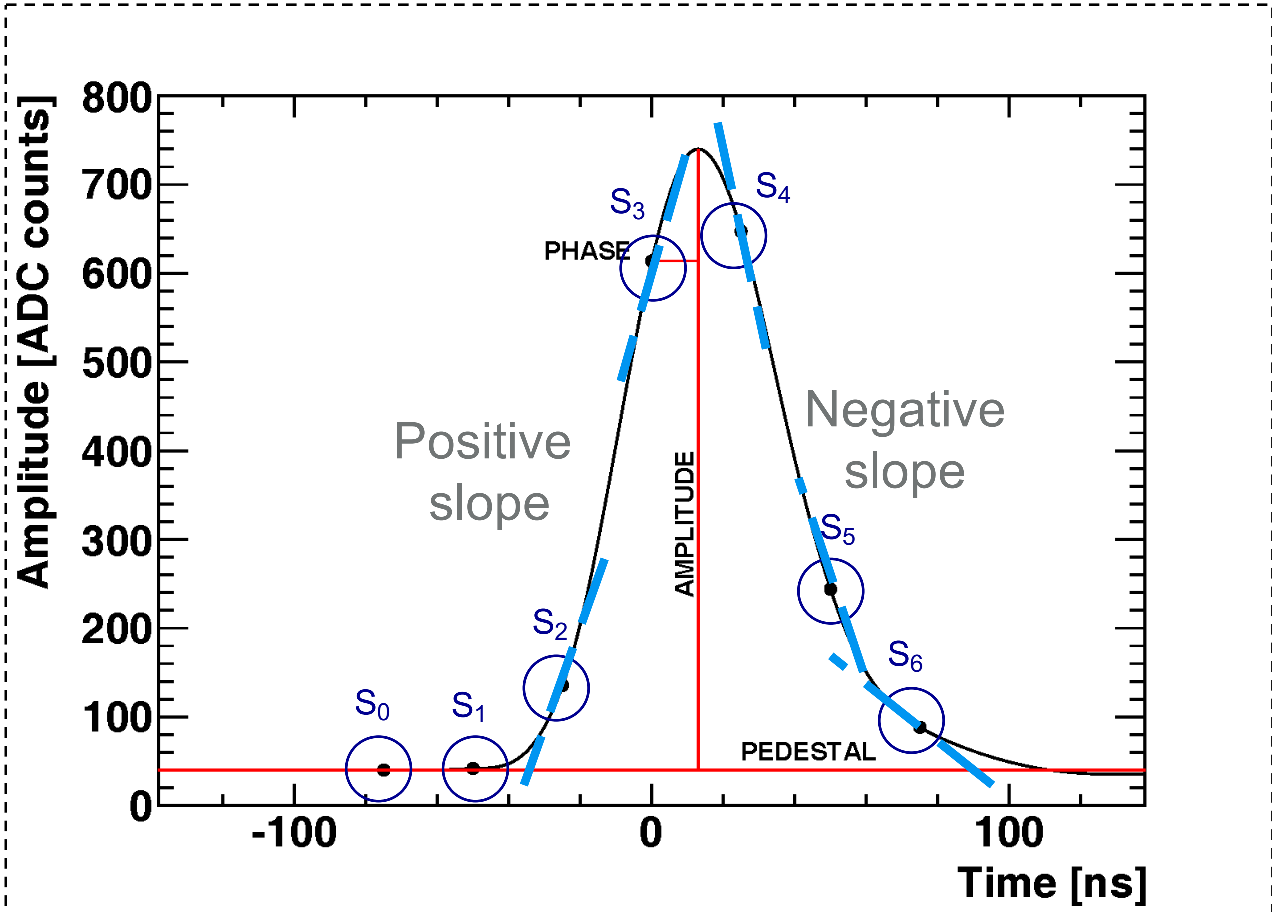
Sample	Weight
S_0	-172
S_1	0
S_2	14
S_3	62
S_4	64
S_5	24
S_6	8



Maximum response when samples are aligned with the ideal pulse shape

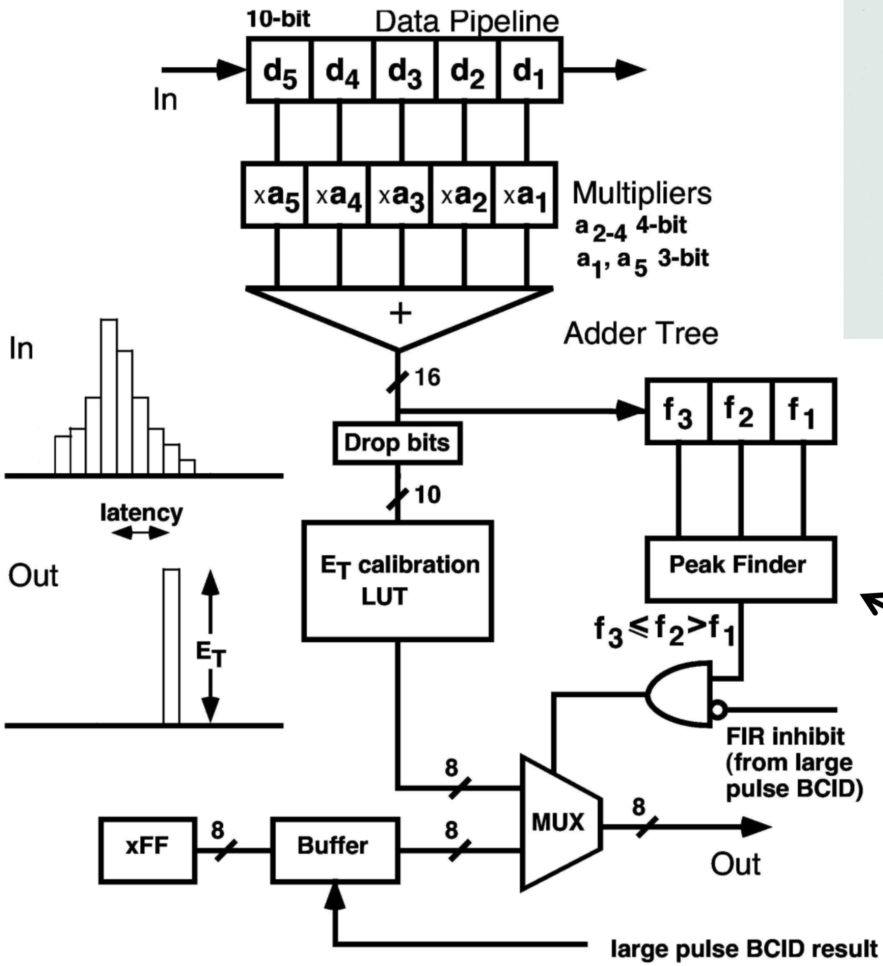
Timing (phase) measurement

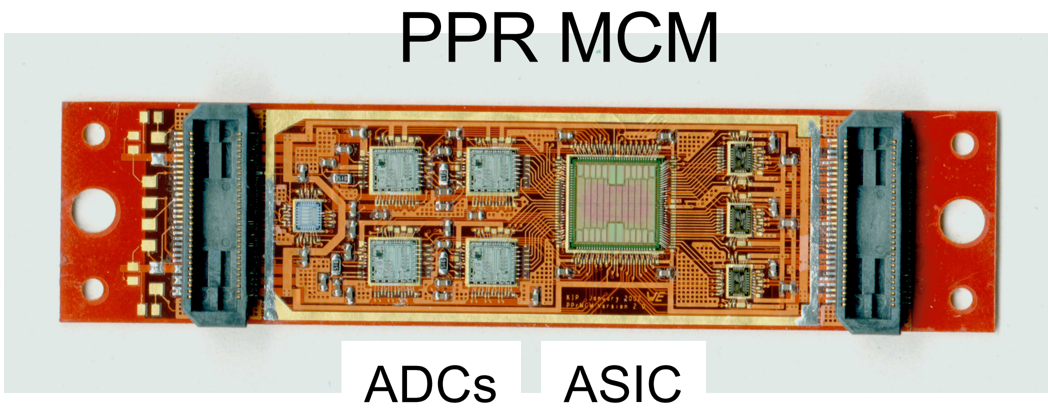
- Use <u>derivatives</u> of ideal curve as coefficients
- Divide the timing filter output by the pulse amplitude
- Can achieve sub-ns timing resolution with 25 ns sampling rate

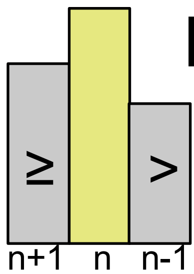


Hardware implementation

Example: ATLAS L1Calo preprocessor ASIC



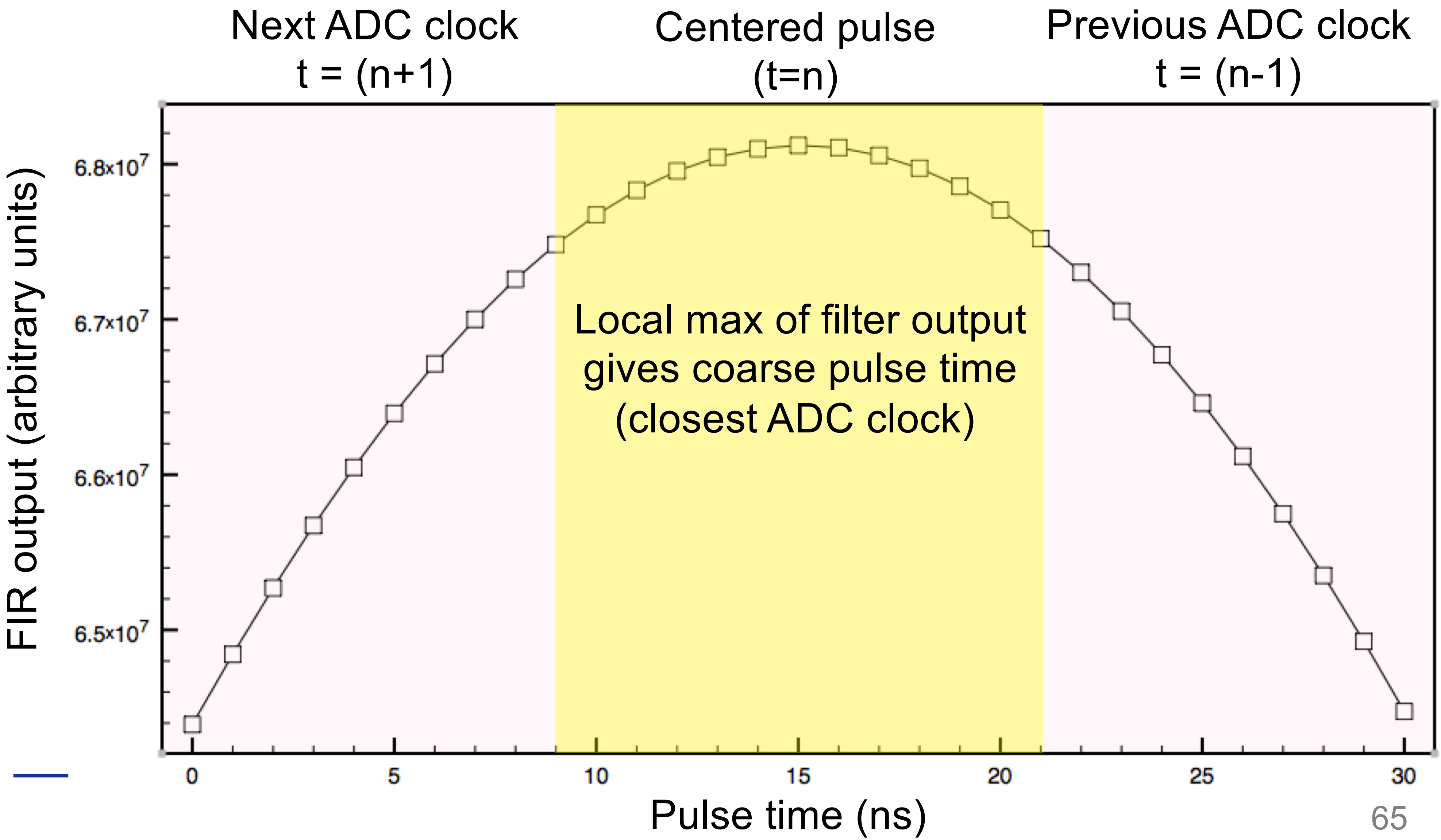




Peak finder identifies BX of pulse

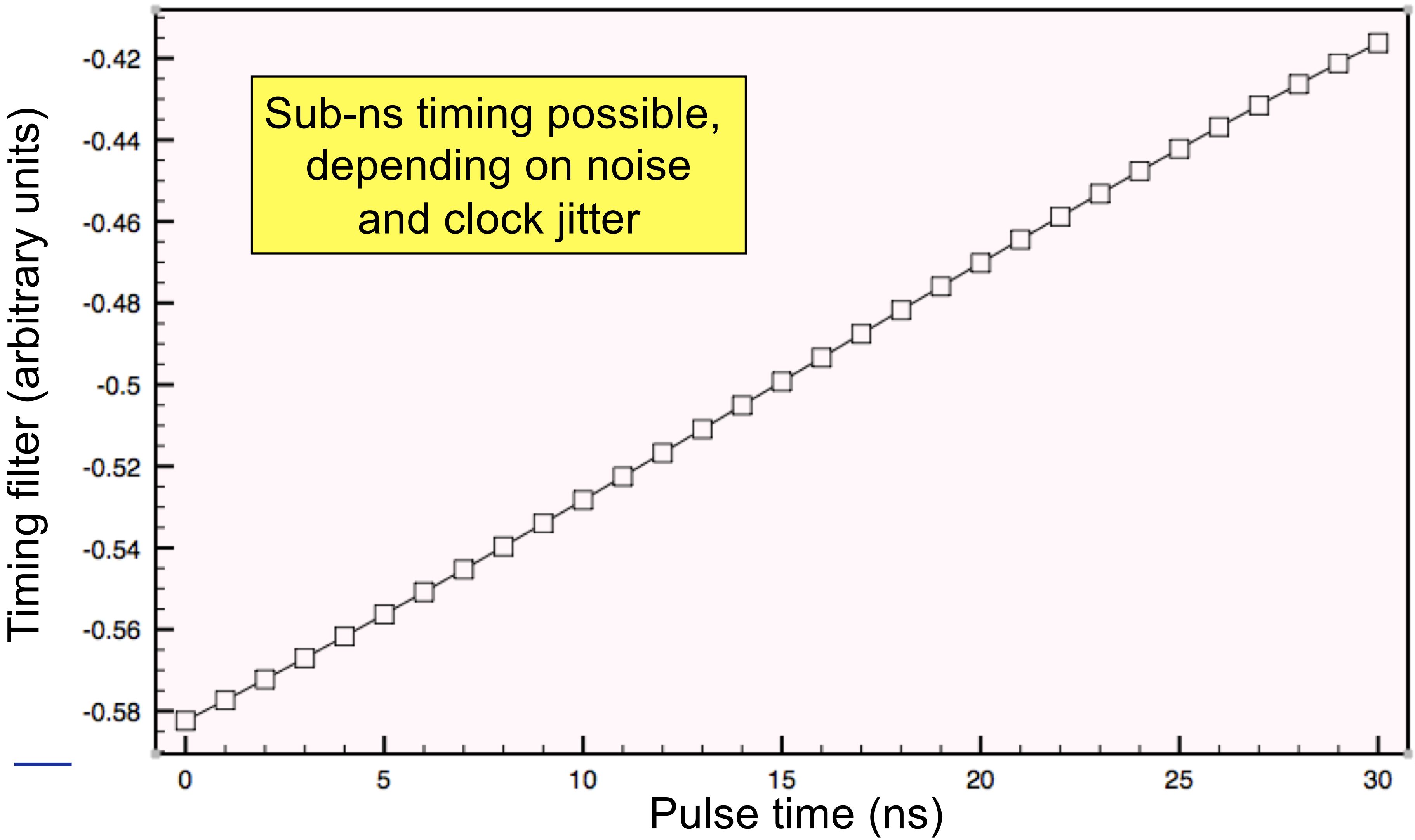
FIR response of shaper

15 samples with fixed coefficients



Fine-timing estimate (offline)

A second FIR filter, coefficients calculated from the derivative of the curve at each sample



In my next two lectures…

- Algorithms and architectures
  - More detailed examples of trigger systems, and how they are built
  - “tools” available for collider detector triggers, and how they can be used
- HL-LHC and future experiments
  - Upgrade planning and implications
  - TDAQ upgrades for SLHC
  - Future directions

Questions?