Data Acquisition in Particle Physics

Igor Konorov

Institute for Hadronic Structure and Fundamental Symmetries (E18)

TUM Department of Physics

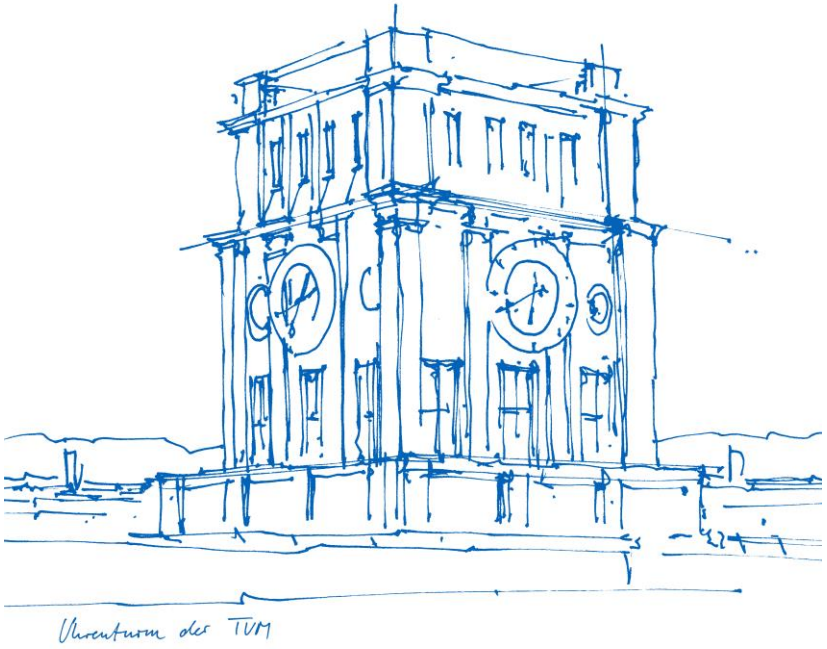
Technical University of Munich

Advanced Workshop on FPGA based

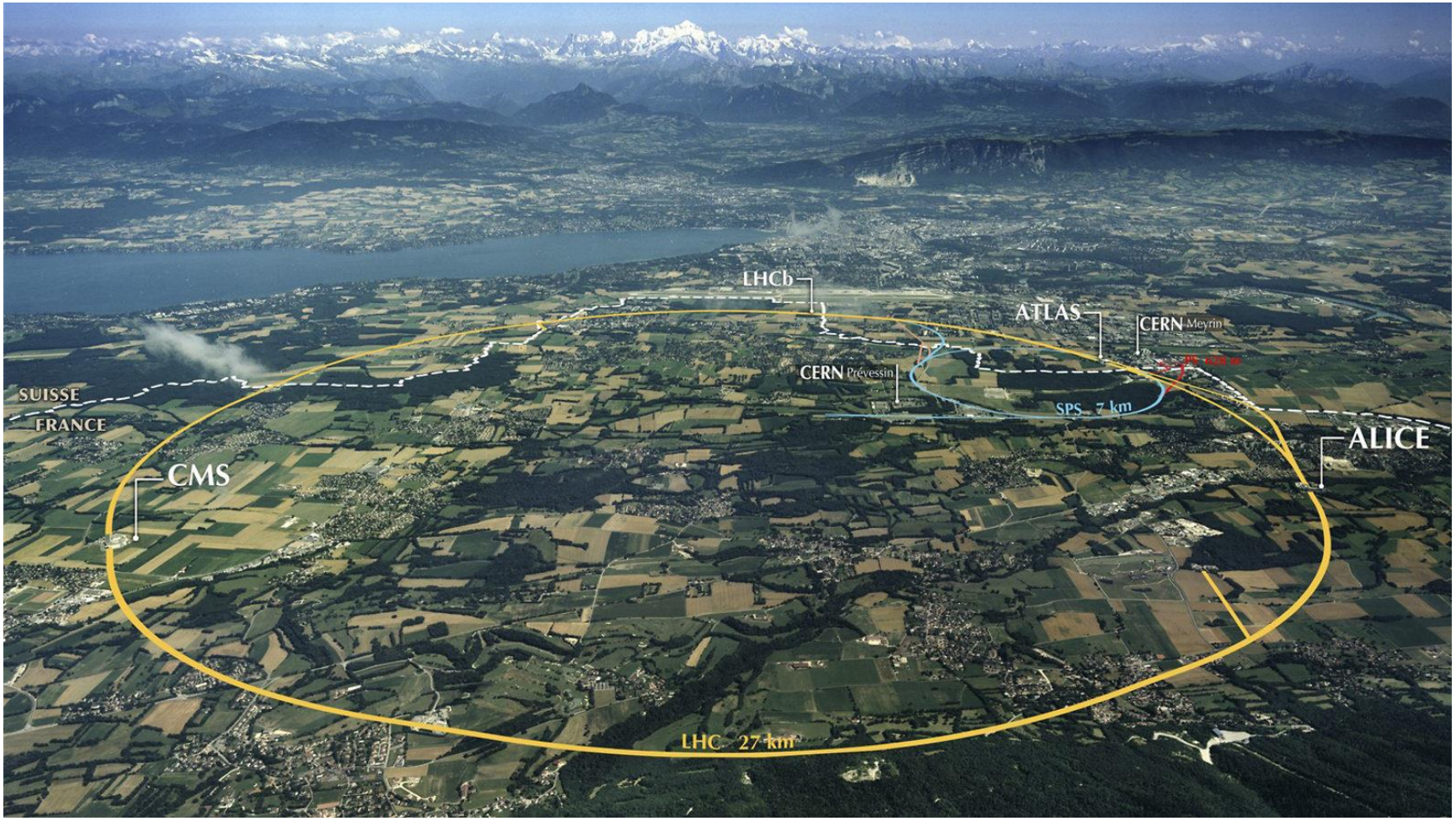
System-on-Chip for Scientific Instrumentation

and Reconfigurable Computing

ICTP Trieste



CERN Accelerator and Experiments



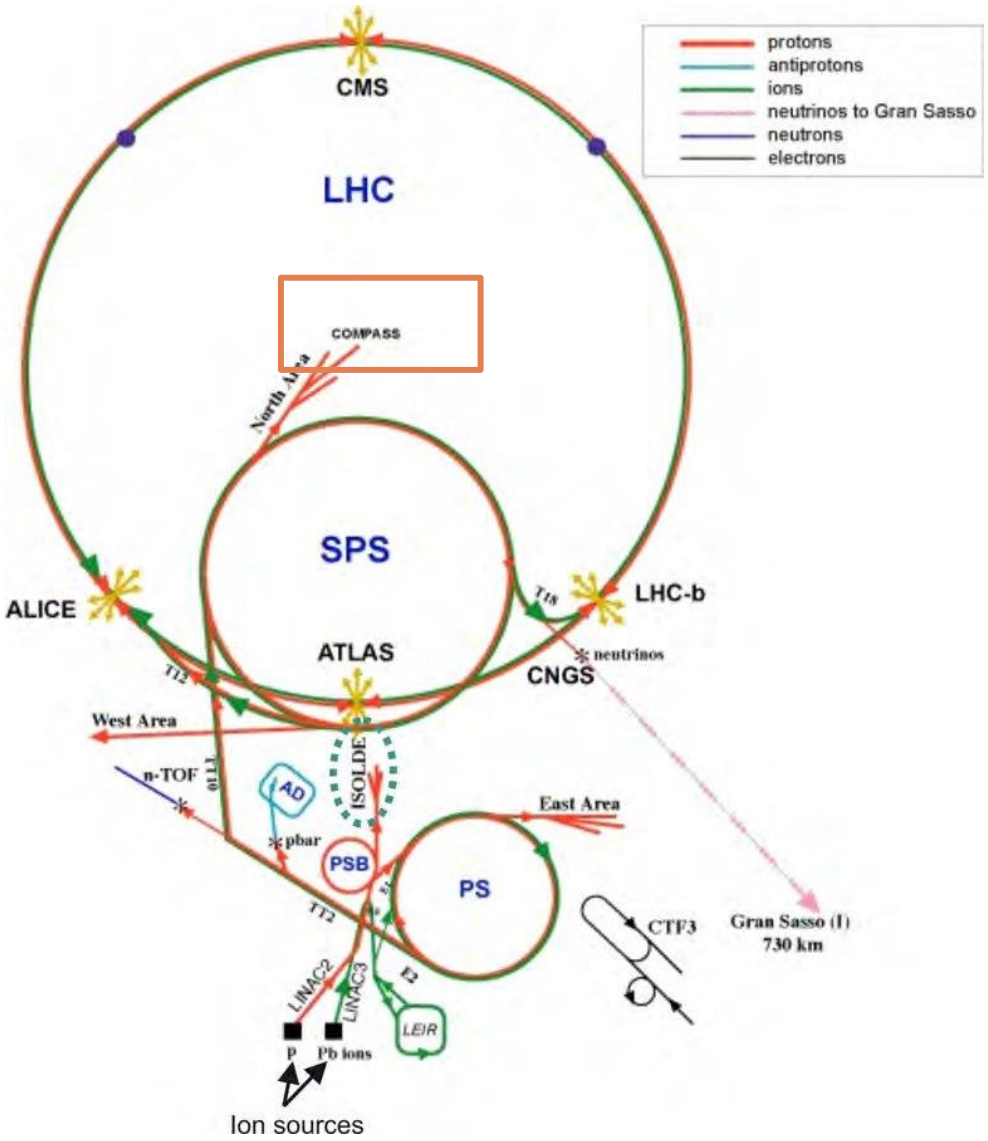
CERN Accelerators

LHC Experiments

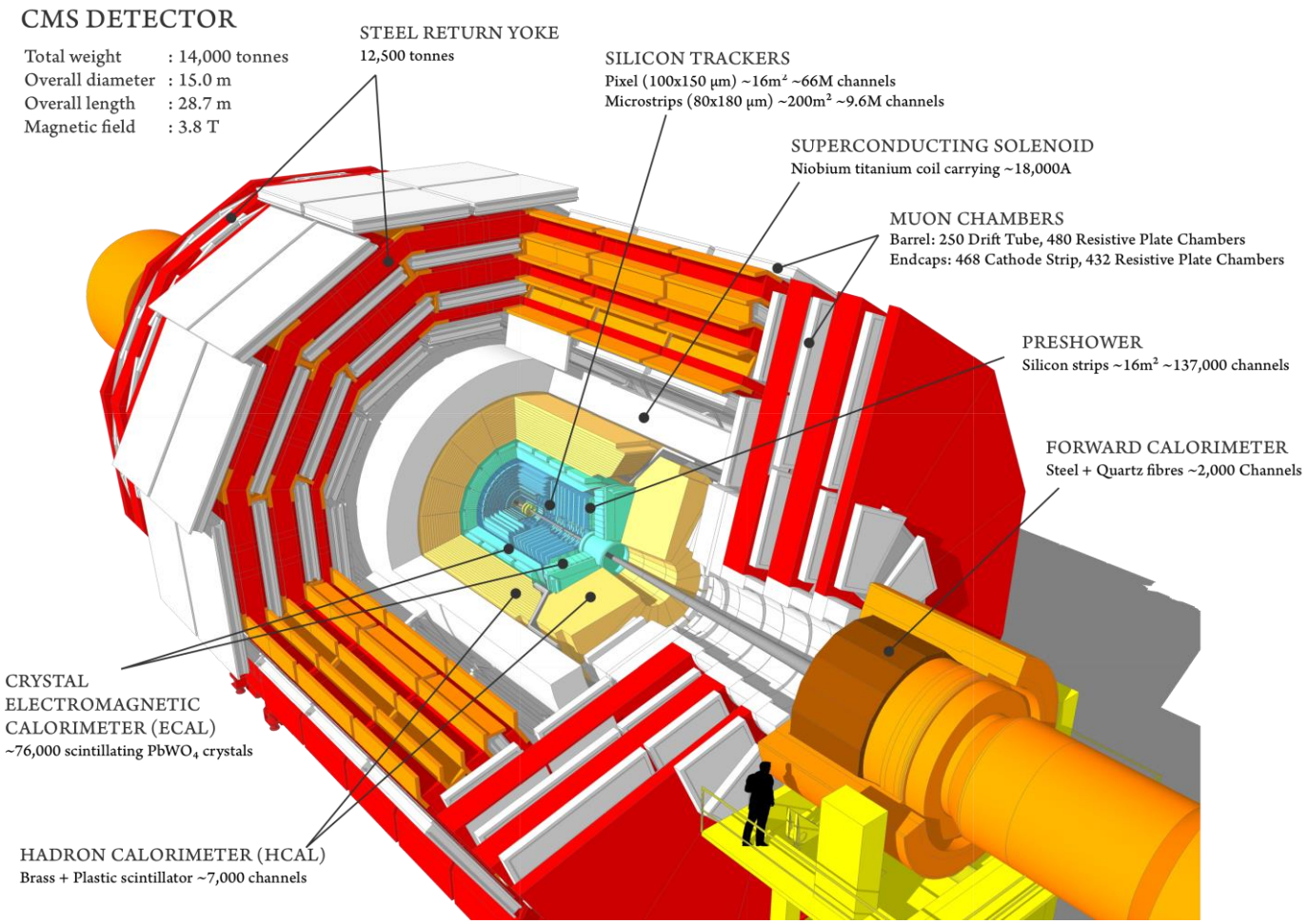
- **CMS**
- ATLAS
- LHCb
- ALICE
- TOTEM, LHCf, MeEDAL

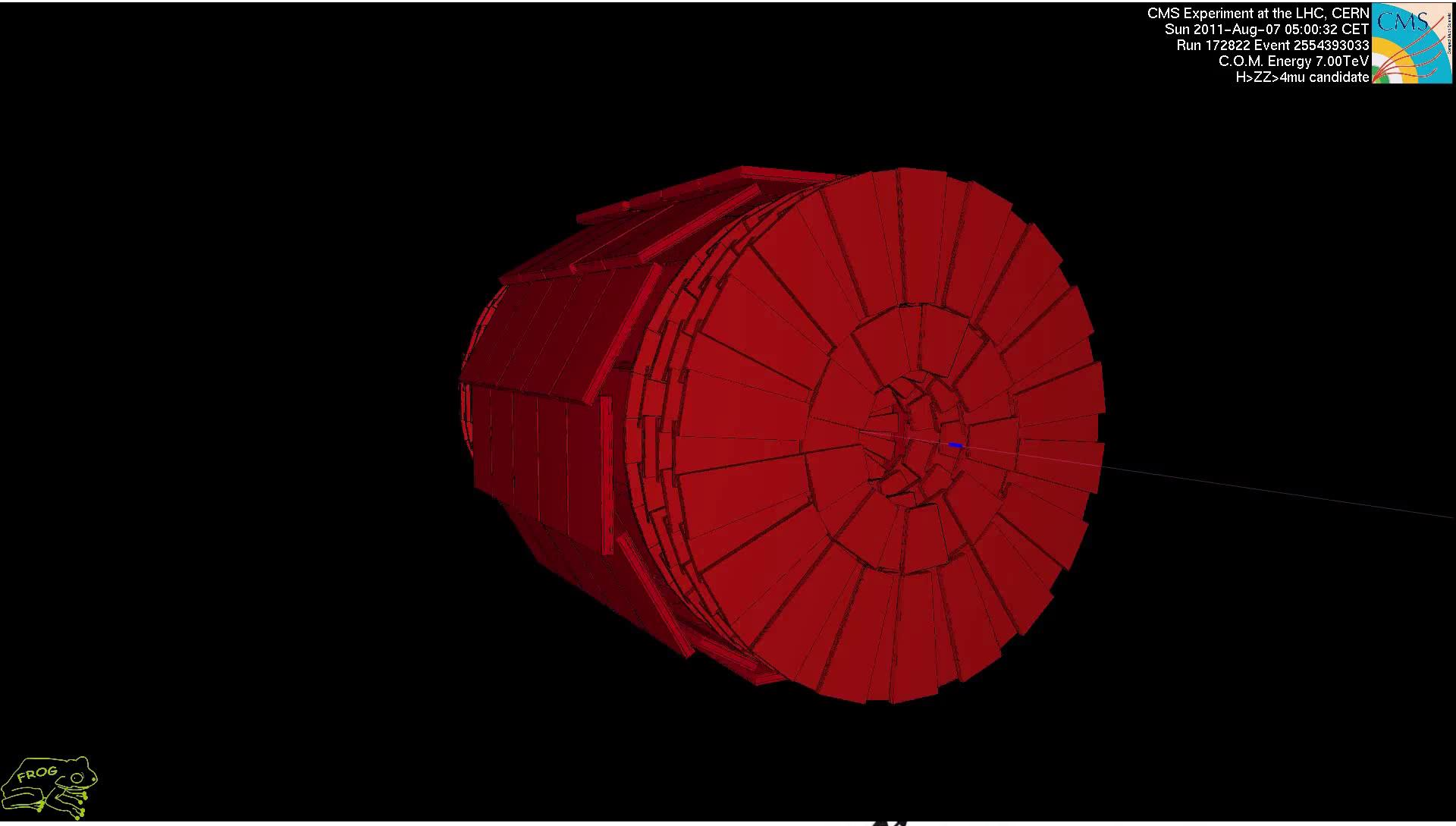
Fixed Target Experiments

- **COMPASS**
- NA61/SHINE
- NA62
- DIRAC
- LOUD
- …



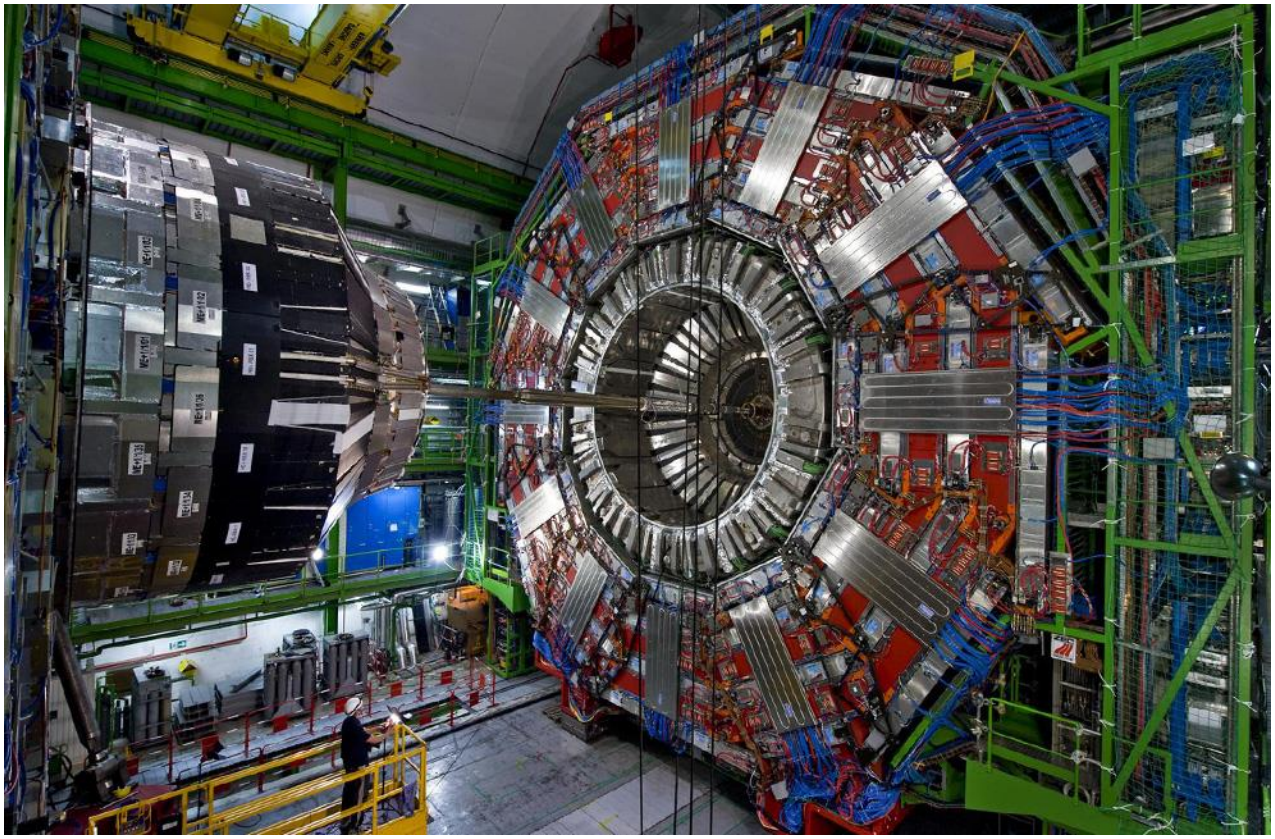
LHC CMS Experiment




CMS Experiment at the LHC, CERN
Sun 2011-Aug-07 05:00:32 CET
Run 172822 Event 2554393033
C.O.M. Energy 7.00TeV
H>ZZ>4mu candidate
CMS
FROG

Ion sources

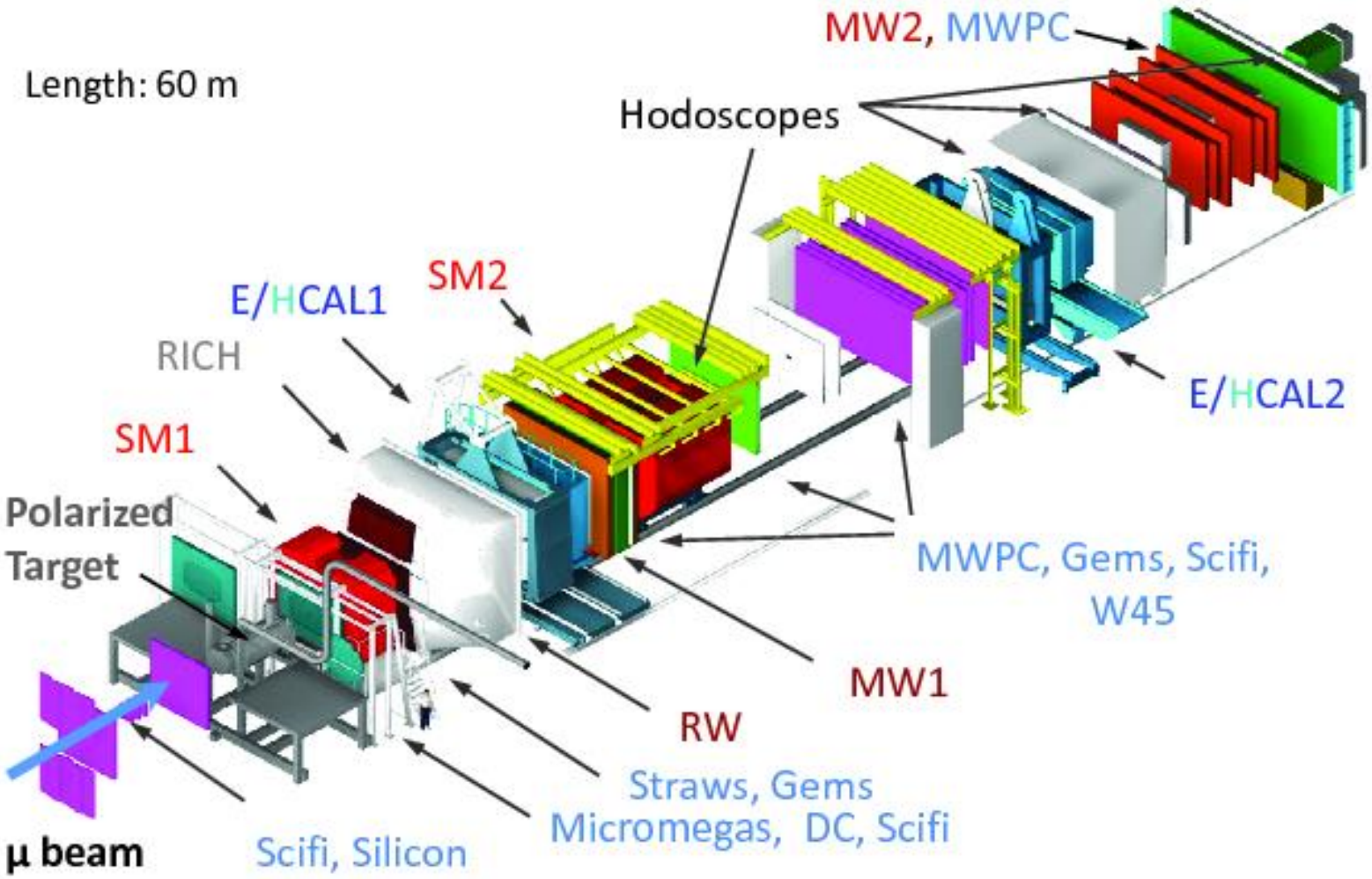
CMS Experiment



3 other LHC experiments

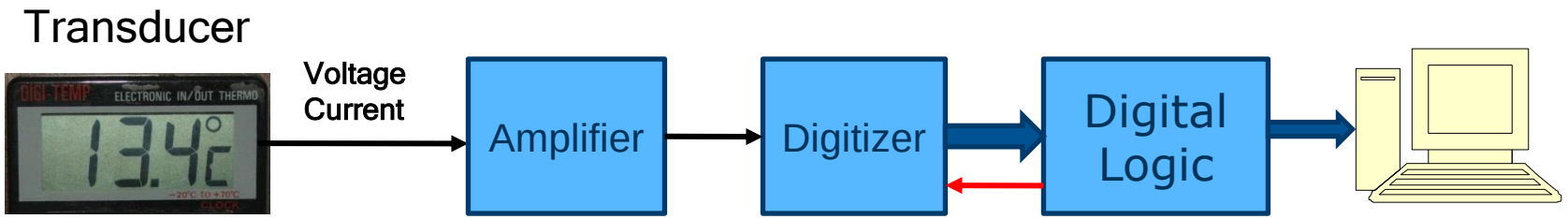
- ATLAS
- LHCb
- ALICE

COMPASS Experiment

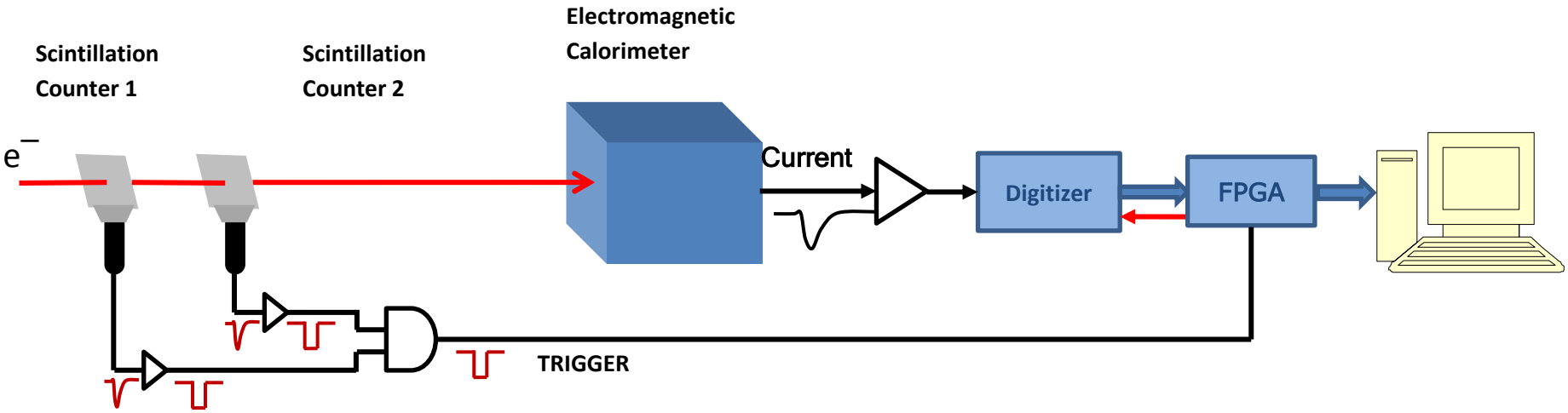


Data Acquisition System

- The process of sampling detector signals
- Conversion to digital form
- Data processing
- Transmission to PC for further processing



Electron Energy Measurement



TRIGGER – define time when amplitude value to be copied

Data path delay should be equal to trigger delay with correction for time of flight !

Triggered DAQ

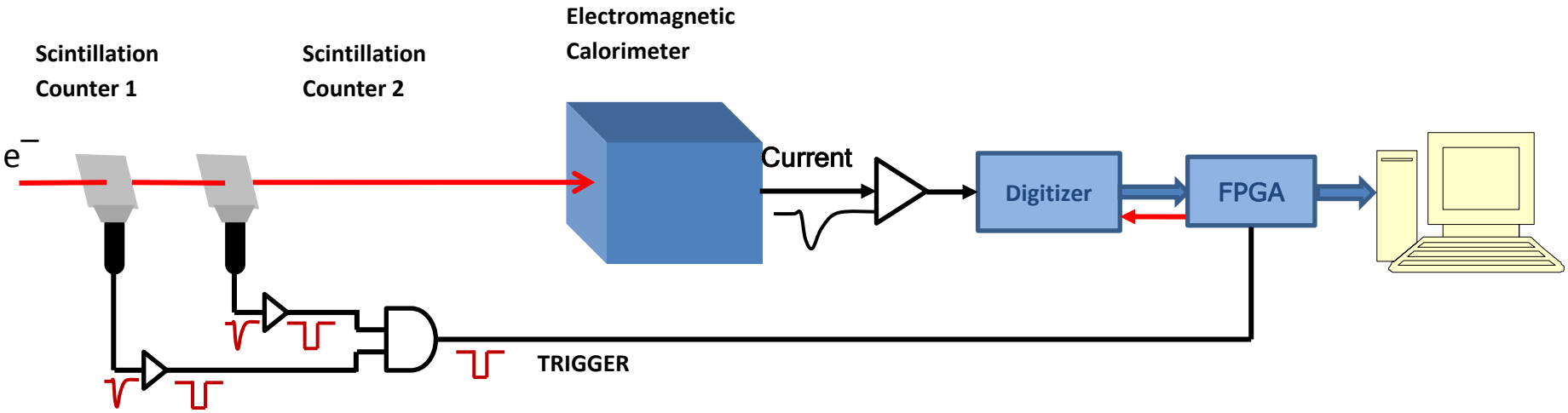
At certain time

- when something interesting happened

Take time measurement=> TRIGGERED DAQ

Photo shooting => TRIGGERED DAQ

Electron Energy Measurement



TRIGGER – define time when amplitude value to be copied

Data path delay should be equal to trigger delay with correction for time of flight !

Why Triggered and not continuous ?

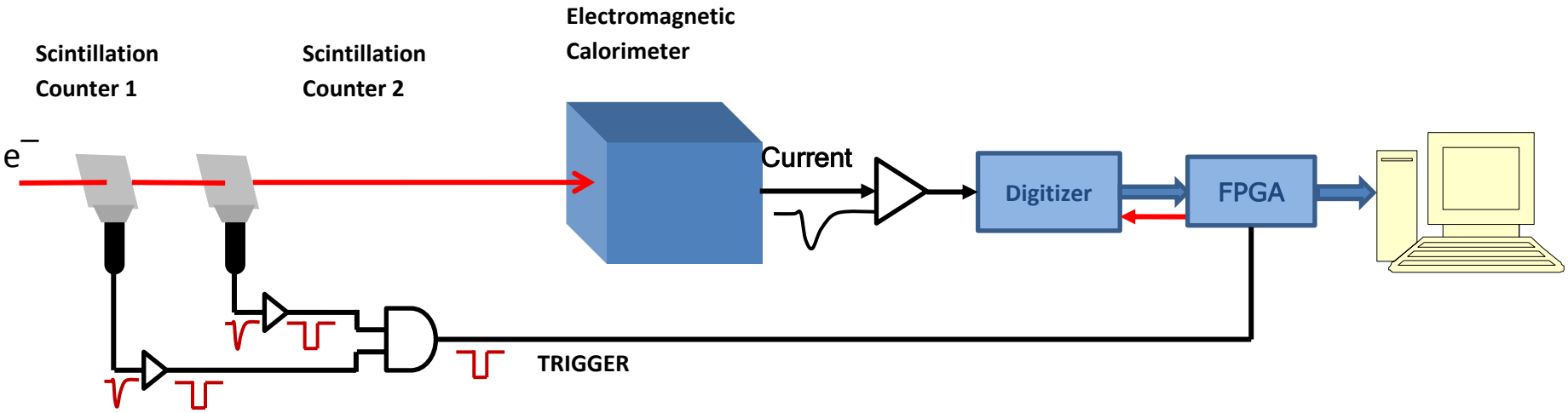
Trigger-less DAQ

Take everything

A time when something interesting happen is not easy to define

Taking video => TRIGGERLESS DAQ

Electron Energy Measurement



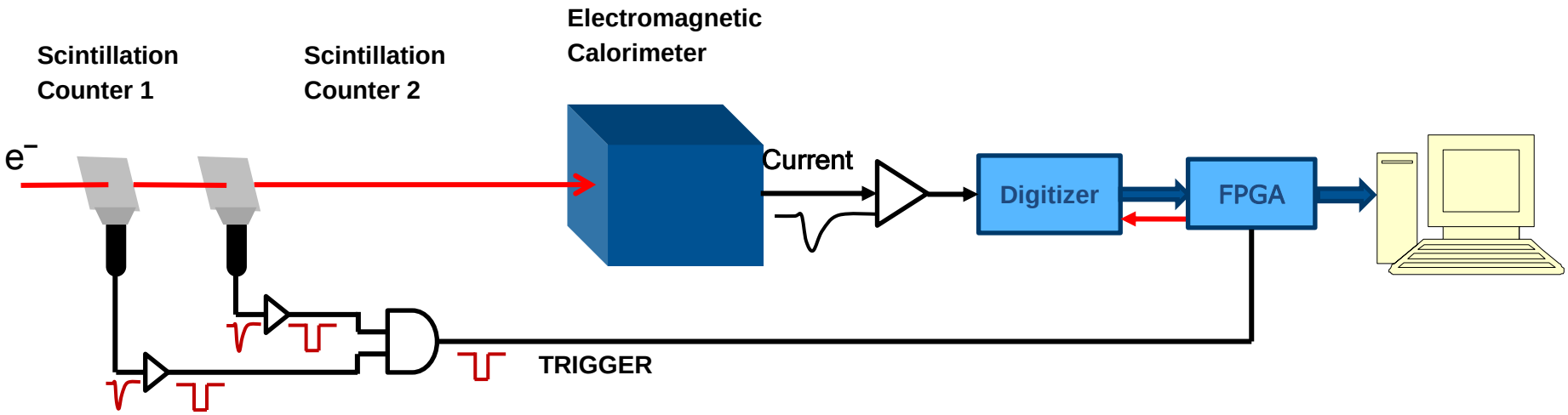
TRIGGER – define time when amplitude value to be copied

Data path delay should be equal to trigger delay with correction for time of flight !

Why Triggered and not continuous ?

- Feasibility to handle continuous stream
- No need to collect all data

Electron Energy Measurement



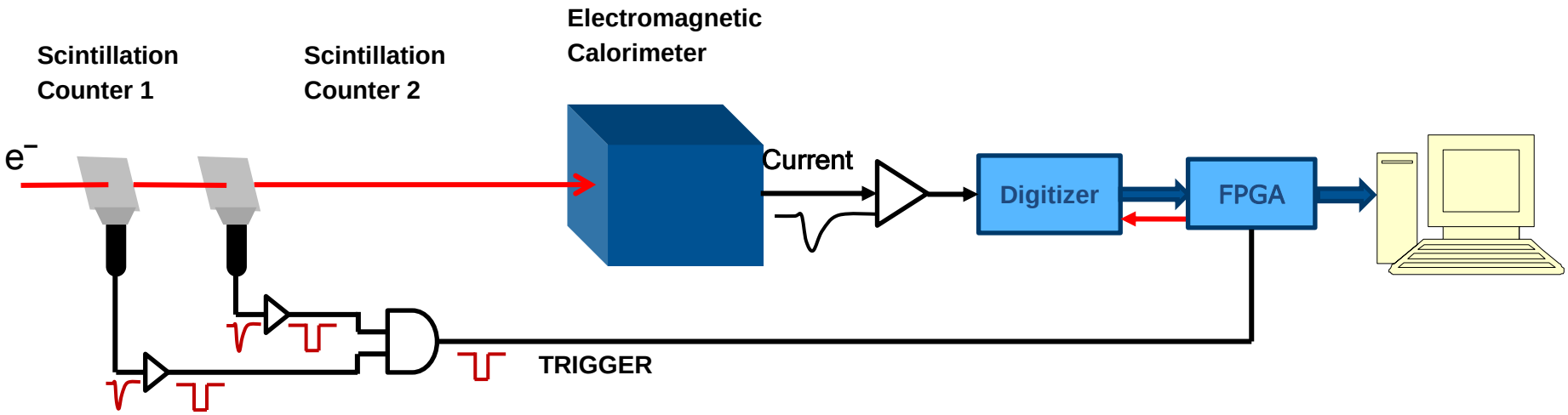
TRIGGER – define time to measure a value
Data path delay should be equal to trigger delay with correction for time of flight !

How much data the system should be able to take?

Probability mass function for Poisson distribution: $P(k) = \frac{\lambda^k e^{-\lambda}}{k!}$
Where λ – average number of events, k – number of occurred events

The system should take as often as maximum trigger frequency

Electron Energy Measurement



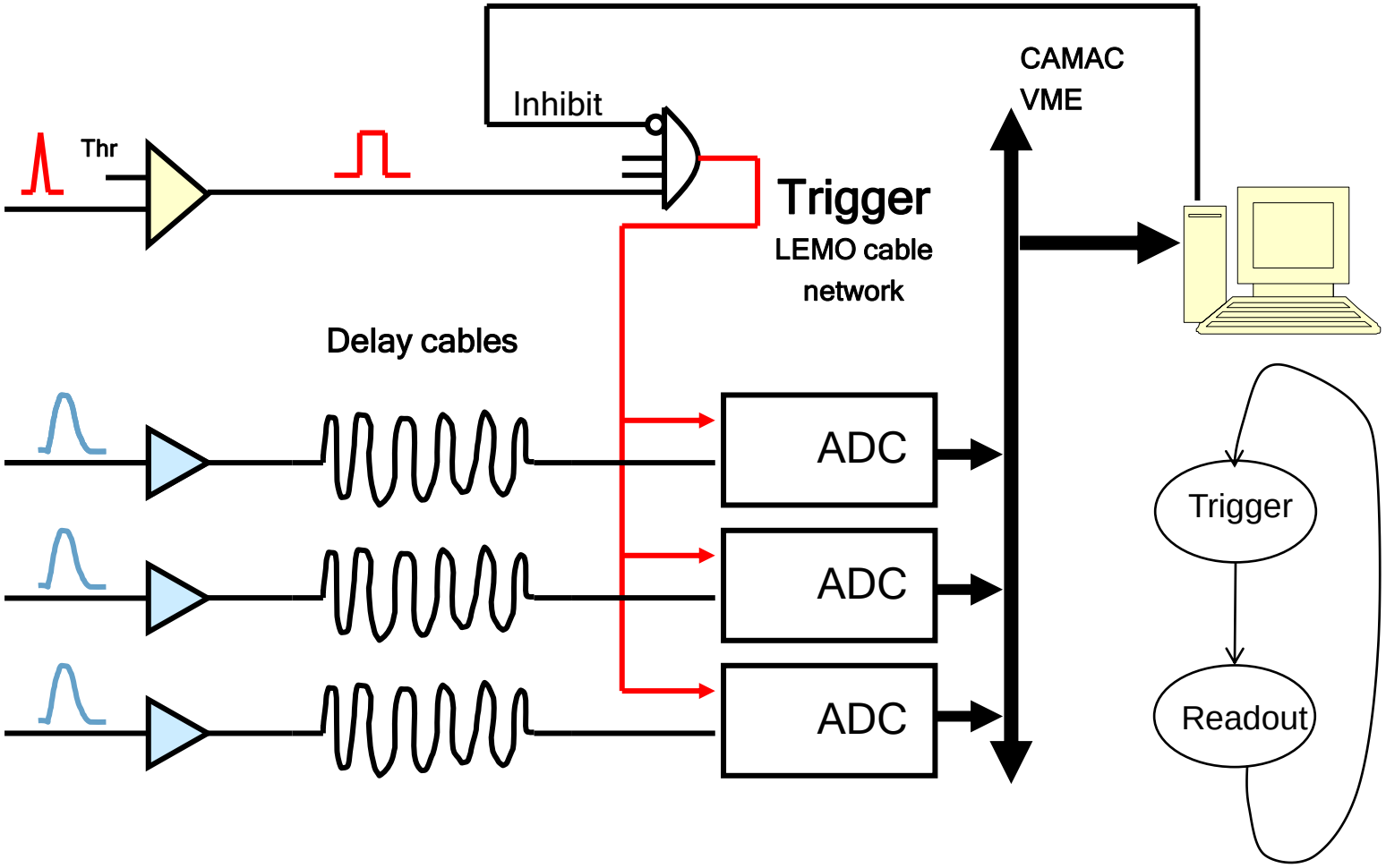
TRIGGER – define time to measure a value
Data path delay should be equal to trigger delay with correction for time of flight !

How much data the system should be able to take?

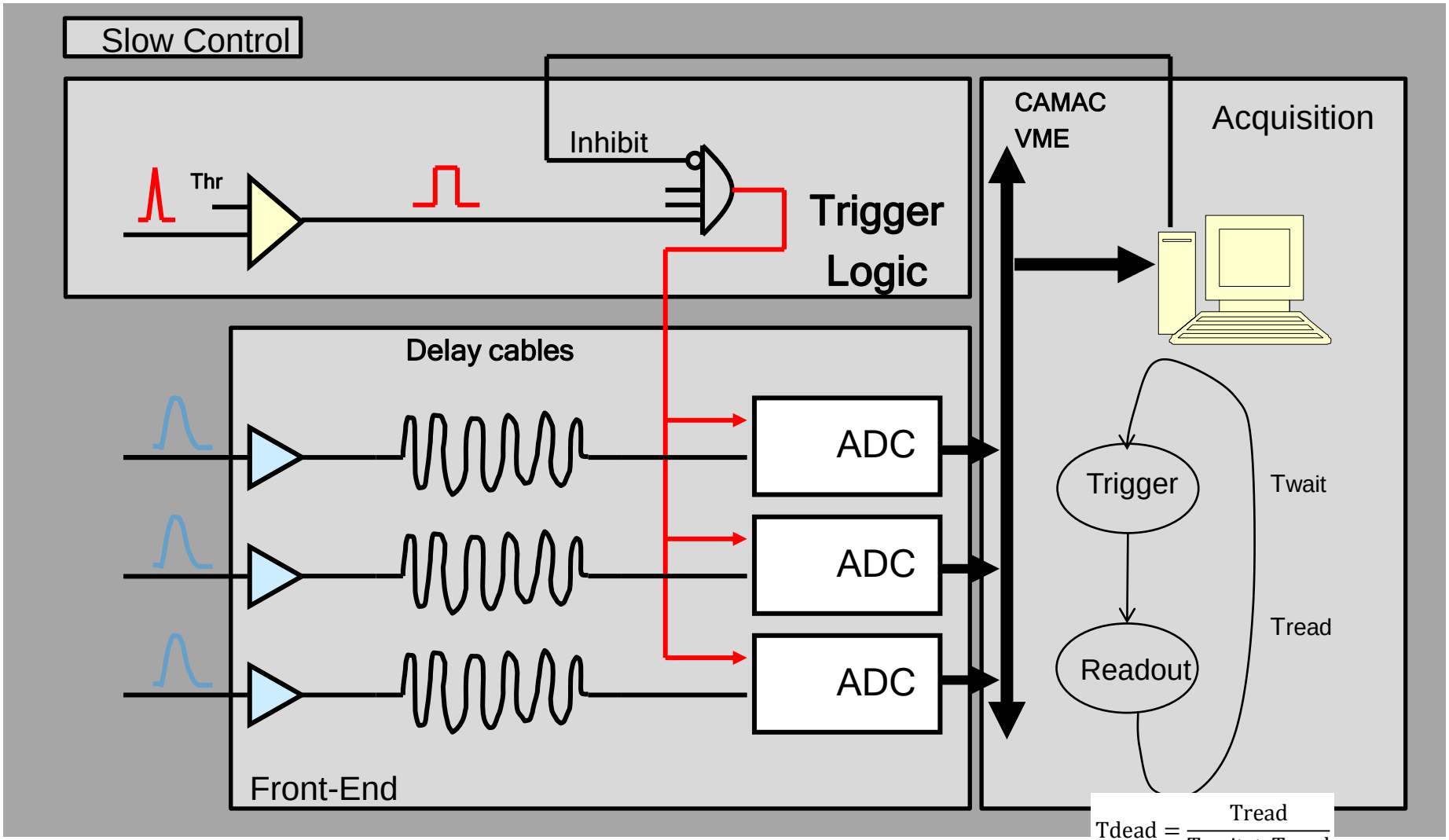
Probability mass function for Poisson distribution: $P(k) = \frac{\lambda^k e^{-\lambda}}{k!}$
Where λ – average number of events, k – number of occurred events

The system should take as often as maximum trigger frequency

DAQ Architecture in Particle Physics



DAQ Architecture in Particle Physics



Efficiency of data taking

RO Sequence : Trigger –> Busy – Read Out -> Release Busy (Ready for next event) = T busy
Probability of events described by Poisson distribution

$$q_j(t) = \frac{r(rt)^{j-1}e^{-rt}}{(j-1)!}$$

J – number of triggers and r – trigger rate

A rule of thumb:

Dead Time = $\frac{Tbusy}{\lambda}$

Tbusy – DAQ busy time

λ – average time between triggers

Tav >> Tbusy

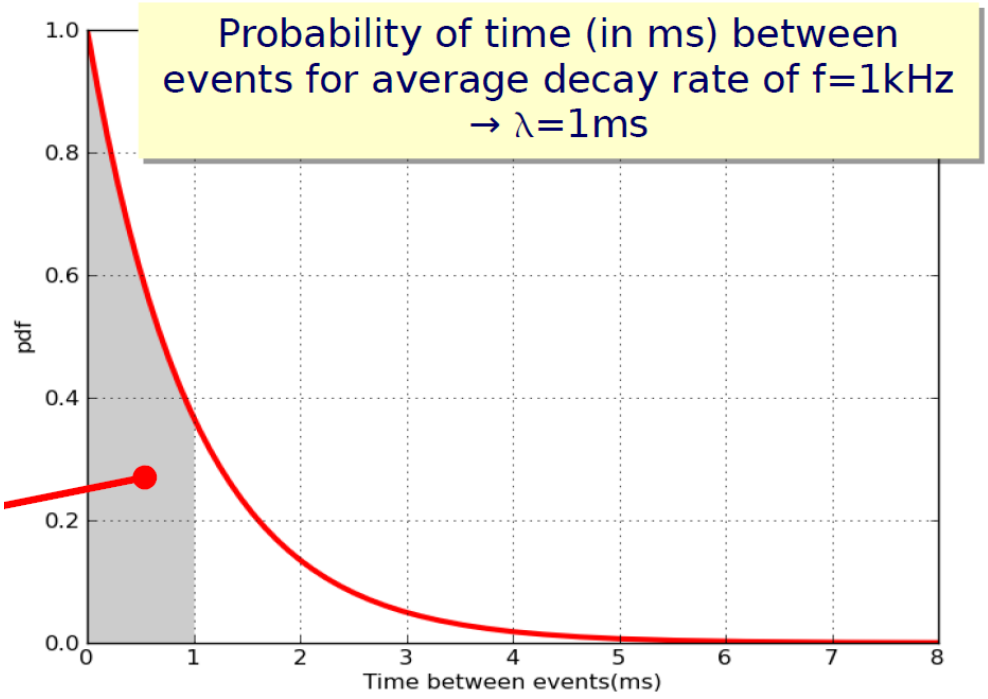
Example:

1kHz => λ = 1ms

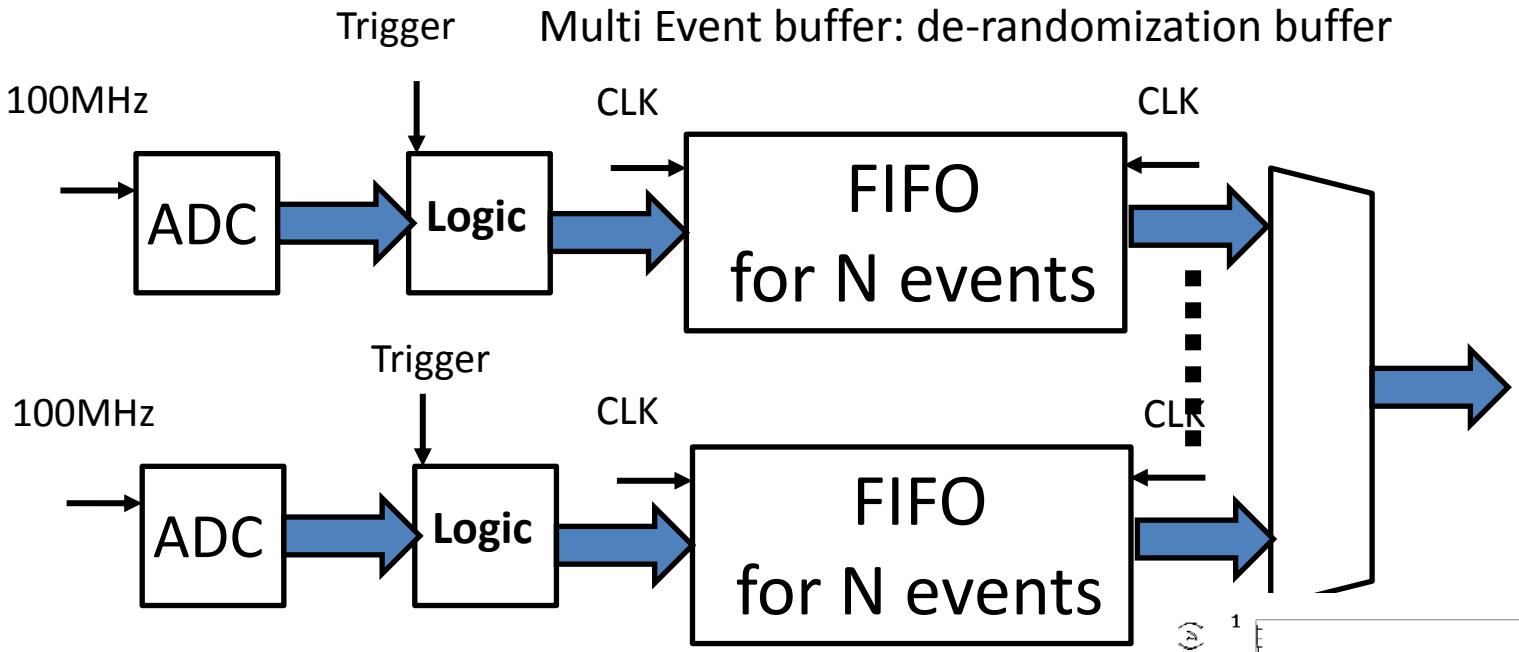
Tbusy = 50 useconds

DeadTime = 0.05/1 = 0.05 or 5%

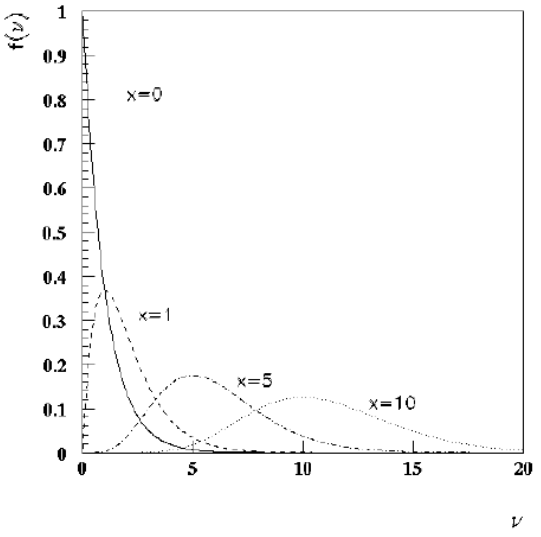
Tbusy



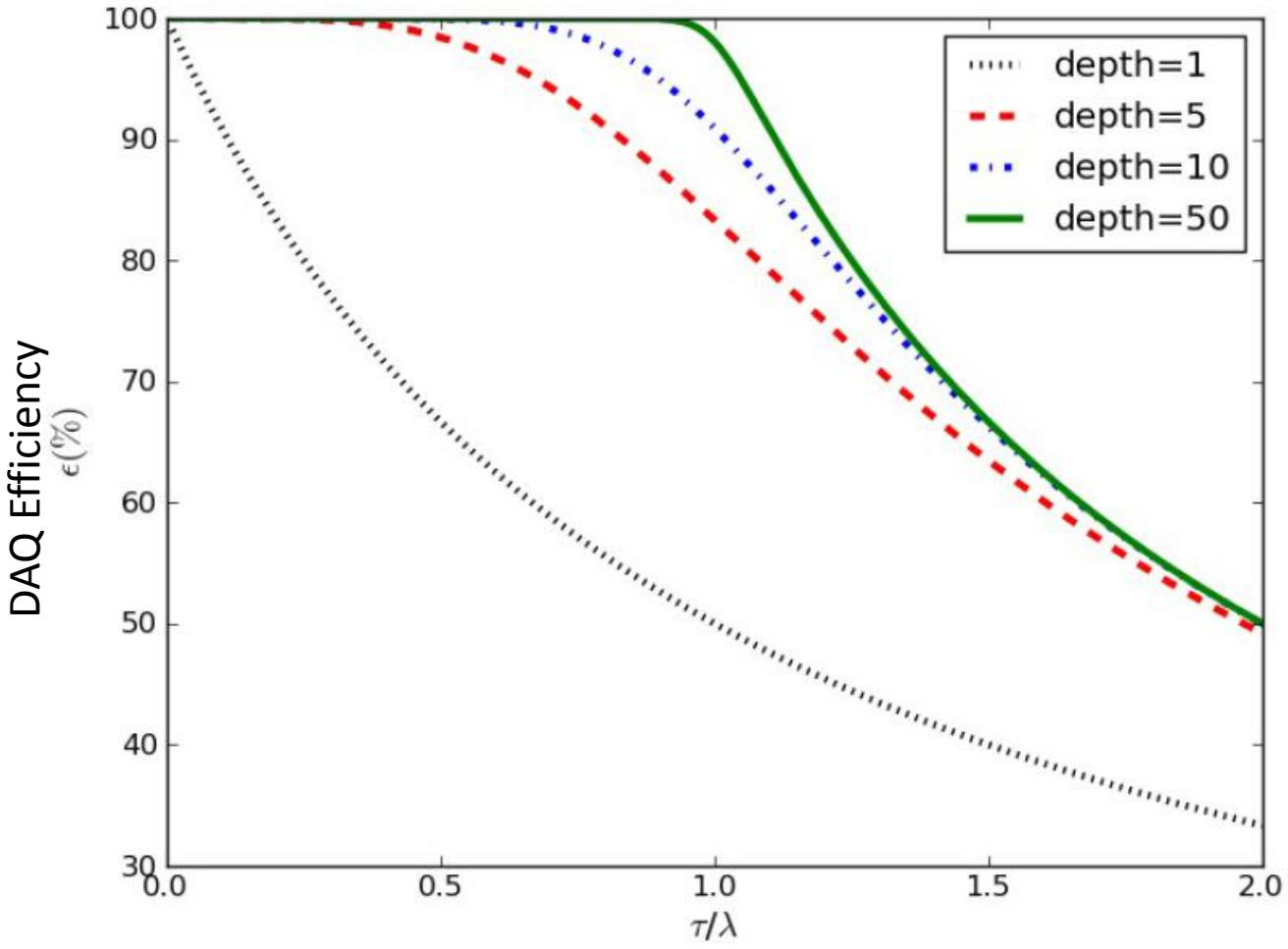
Pipe Line Front Ends



Input : Poisson distribution
Output : more like a Gaussian centered around average value

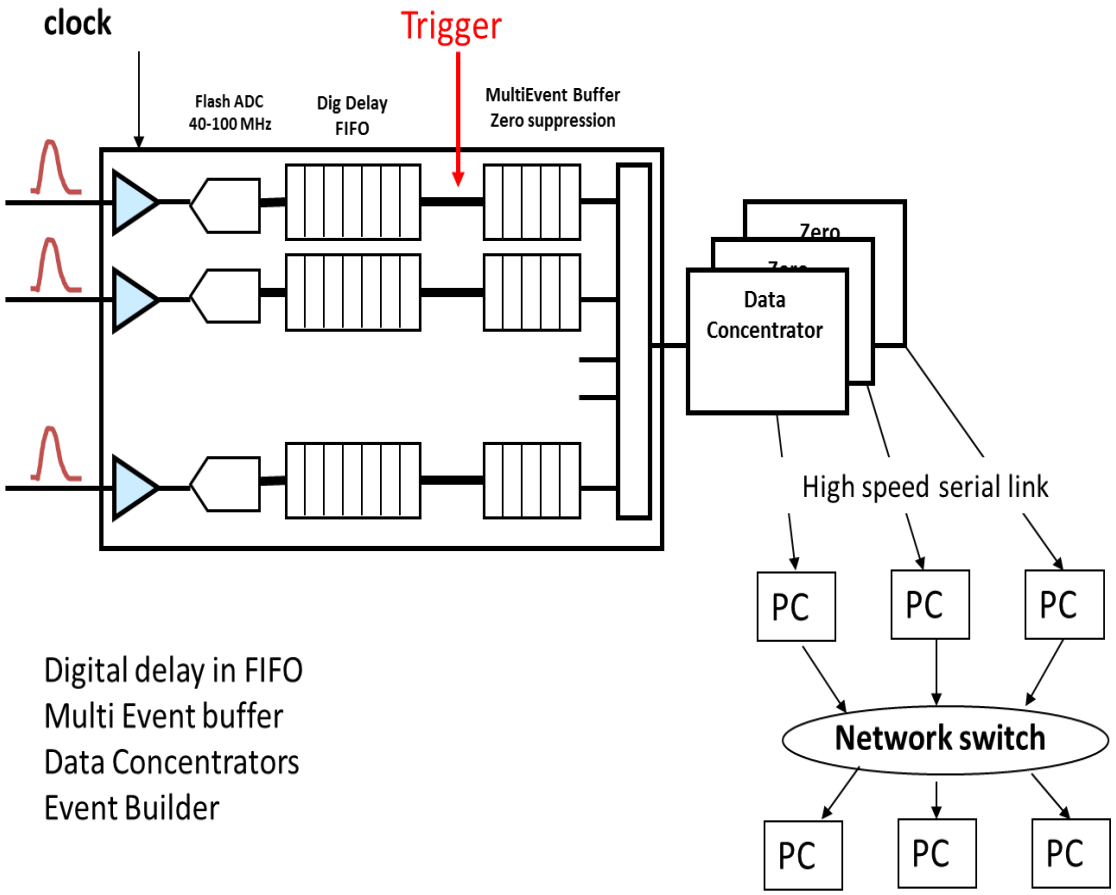


DAQ efficiency vs FIFO Depth

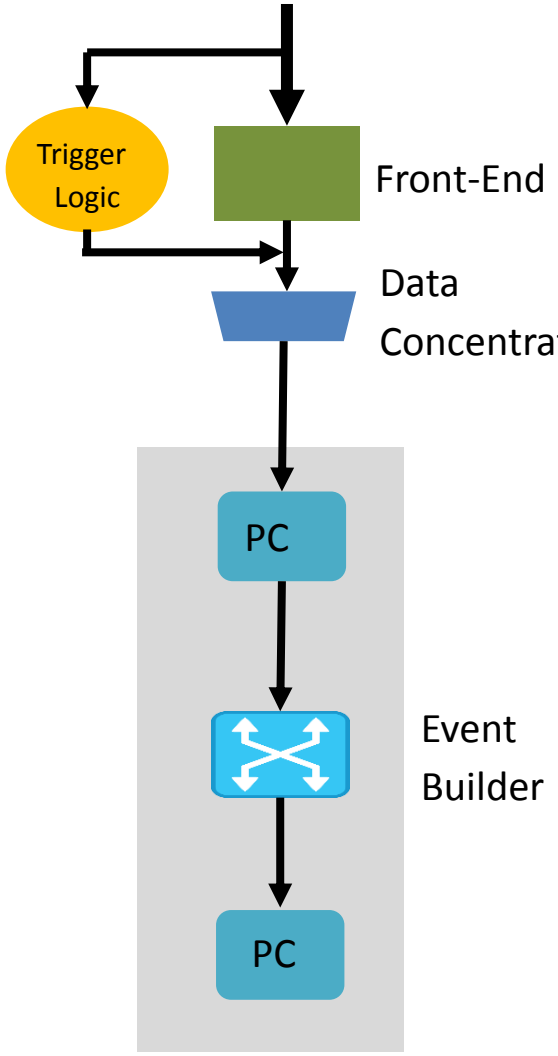


Trigger rate/DAQ rate capability

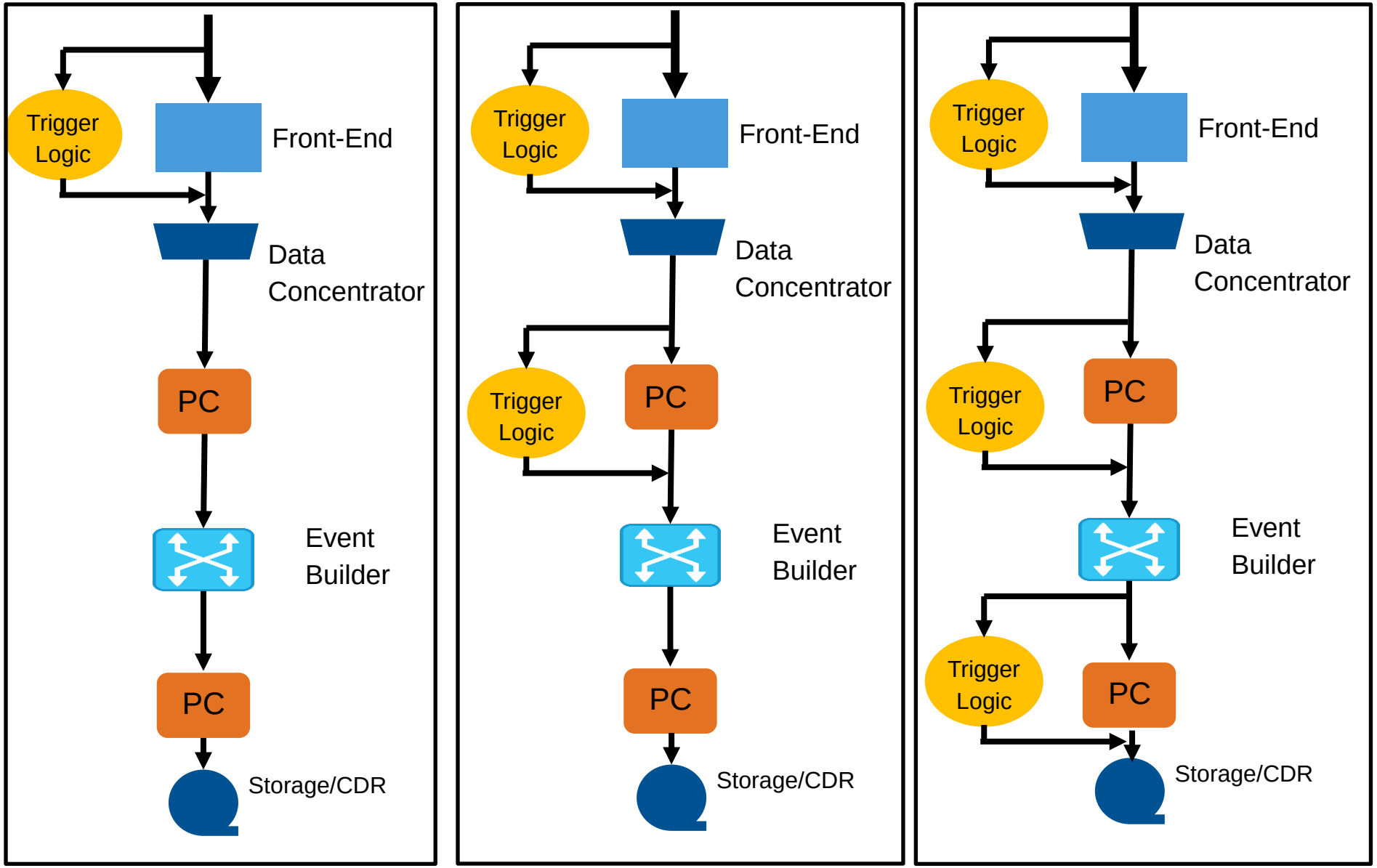
Data Flow DAQ Architecture



Digital delay in FIFO
Multi Event buffer
Data Concentrators
Event Builder



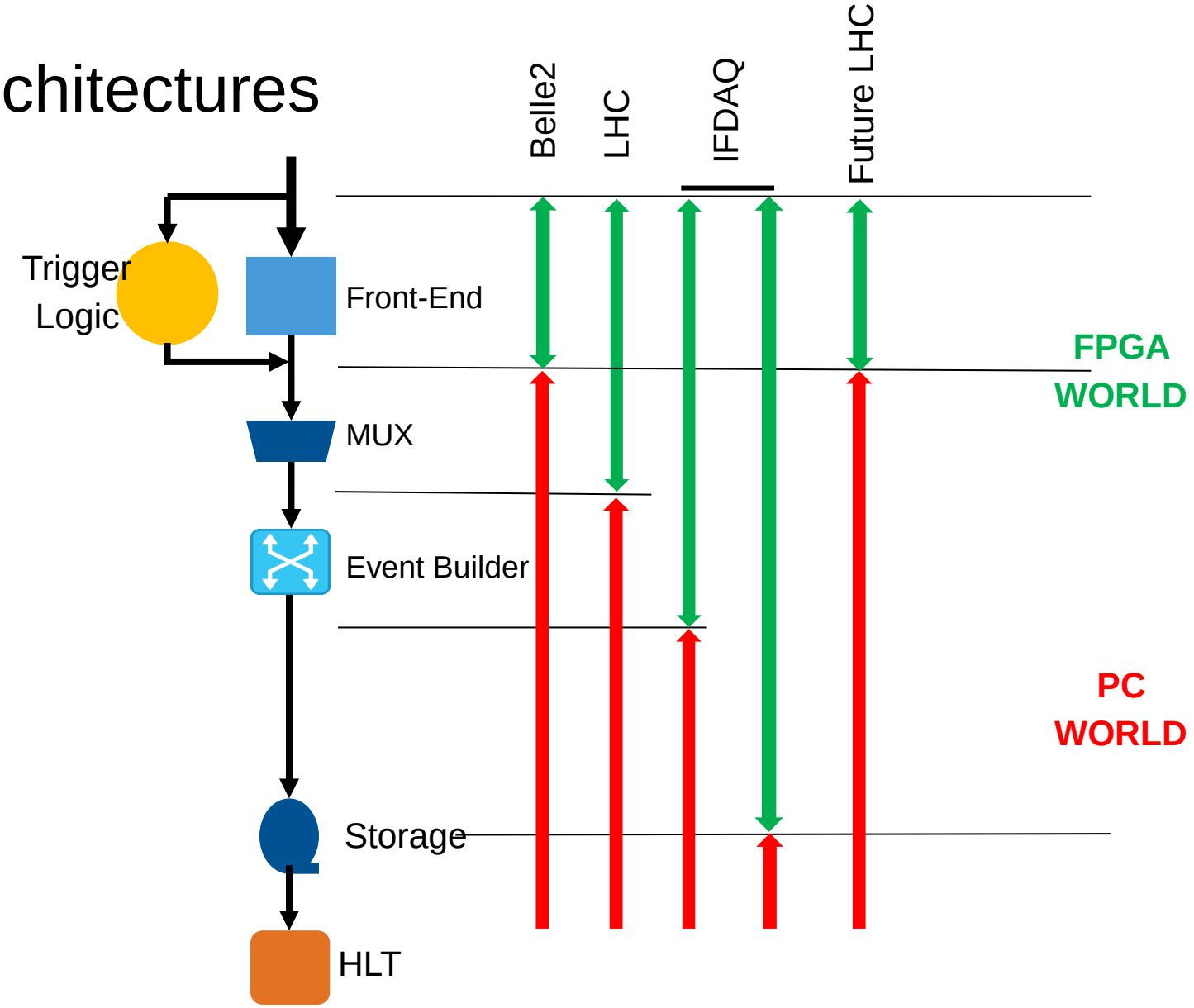
DAQ Architectures



DAQ Elements

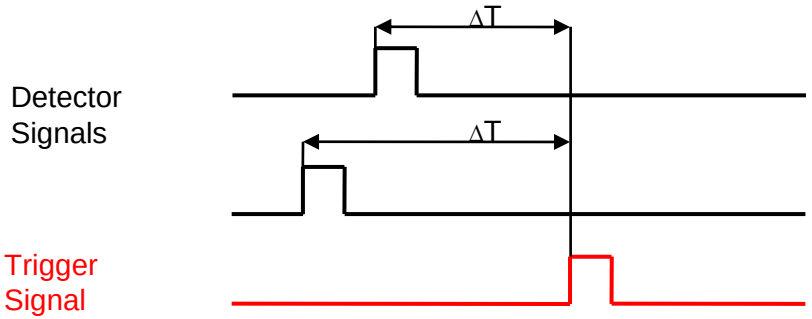
- Front-end electronics, detector specific
  - Conversion of detector analog signal to digital form
  - Derandomization
  - Data processing: signal detection, extraction of signals' parameters **Time** and/or **Amp…**
- Trigger Logic
  - reduce amount of stored data
  - define time when of interesting event
- Trigger Distribution system => Time Distribution System
- Slow Control System
  - Control and monitoring of PS, Gas system, Temperature, Humidity,…
  - Programming of Front-ends
- Acquisition System => Event builder
  - Data acquisition – moving data from FE to PCs
  - Data flow control
  - Real time Software
  - Run control

DAQ Architectures



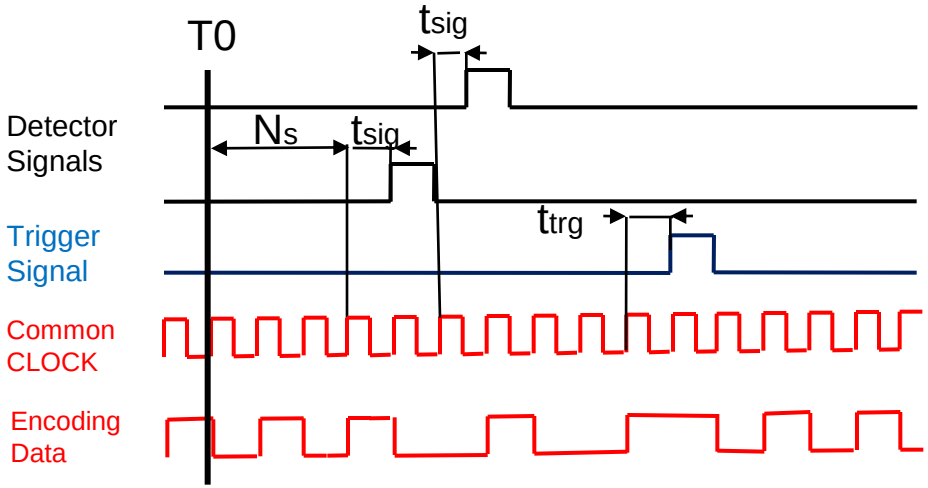
Time Distribution and Time Measurement

Time measurement



Classical method:

- TRIGGER is a reference
- SIGNAL time is measured respectively to TRIGGER signal



Alternative method for big experiments:

- Distribute CLOCK , why clock?
  - Easier to distribute with very low jitter
- Measure absolute time respectively to CLOCK phase

$T_{sig} = N_s\ T_{clk} + t_{sig}$

$T_{trg} = N_t\ T_{clk} + t_{trg}$

Clock and Data are encoded and transmitted from single source to multiple destinations

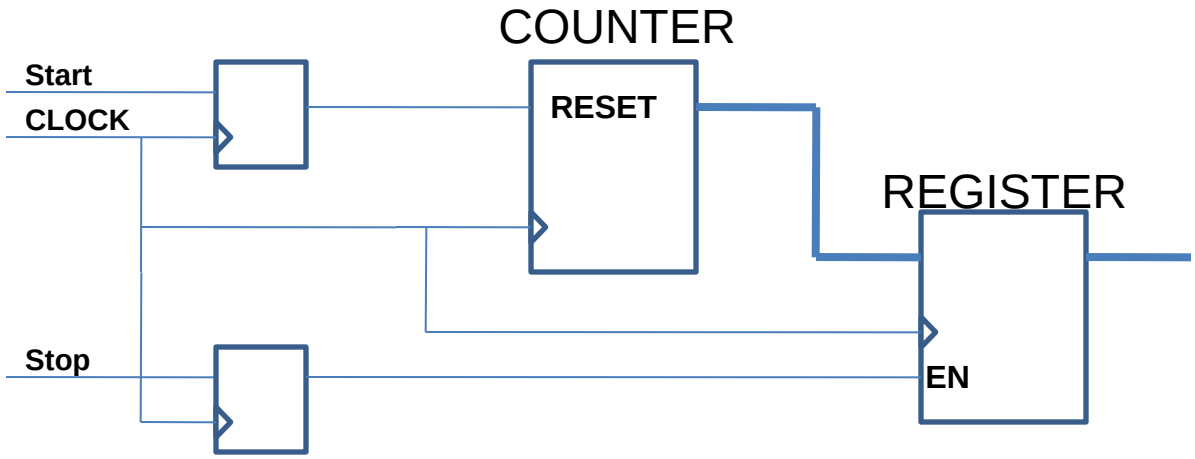
NA48, LHC->TTC, COMPASS->TCS

TDC types

- **Time stretching** :
  - Time measurement between START and STOP
  - Fast charging of capacitor with reference current => slow discharging
- **Time to amplitude converters :**
  - charging capacitor with reference current => ADC to measure amplitude
- **ASIC TDC: Delay Locked Loop based TDC**
- **FPGA Counter as a simple TDC**
- **FPGA TDC**

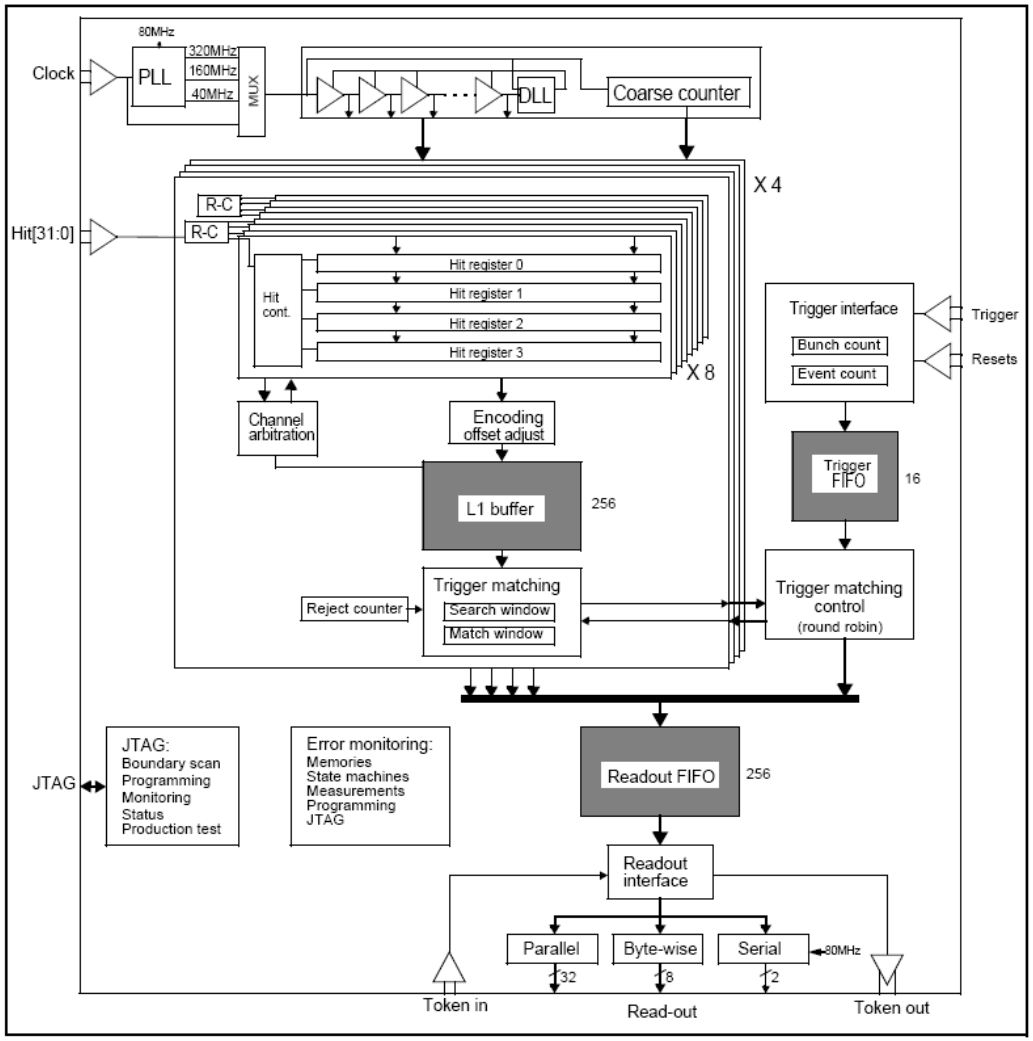
Counter as TDC

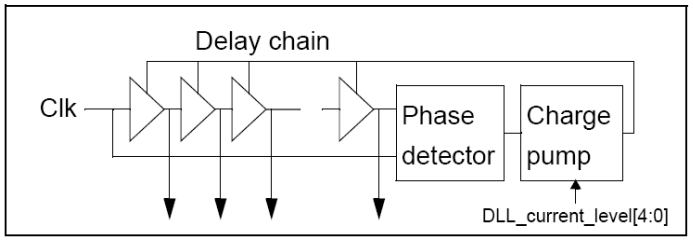
Counter based
Fclock = 400MHz



Tbin = Tclock

Delay Lock Loop TDC, HPTDC



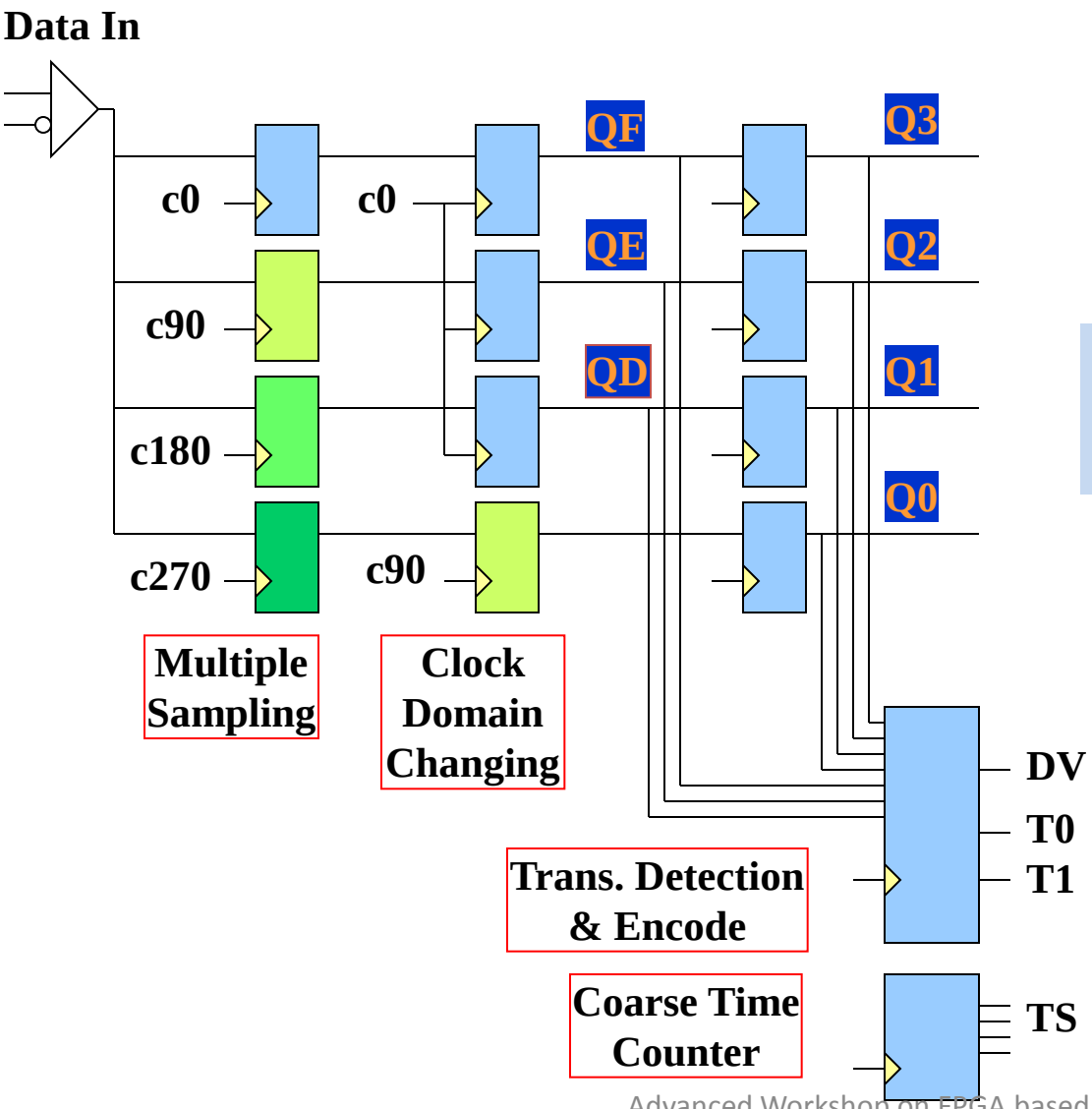


Transistor parameters vary from chip to chip and even within one chip

DLL compensates variation of transistor parameters from chip to chip and due to voltage and temperature

- Resolution 25ps
- 32channels/chip

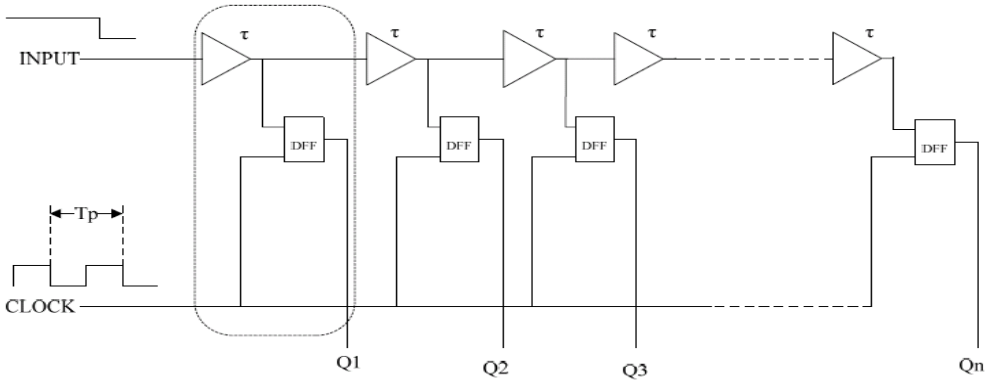
FPGA based TDC1



4x Sampling:
250 MHz: 1ns(LSB), 288ps(RMS)
400 MHz: 625ps(LSB), 180ps(RMS)

Tapped Delay TDCs

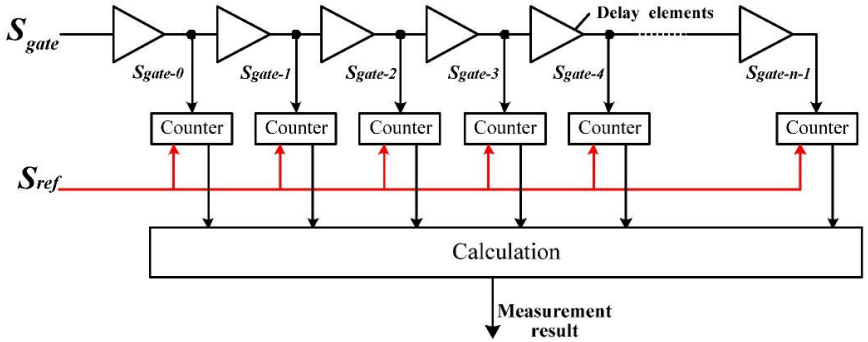
Fermilab



Jinyuan Wu and Zonghan Shi
Fermilab

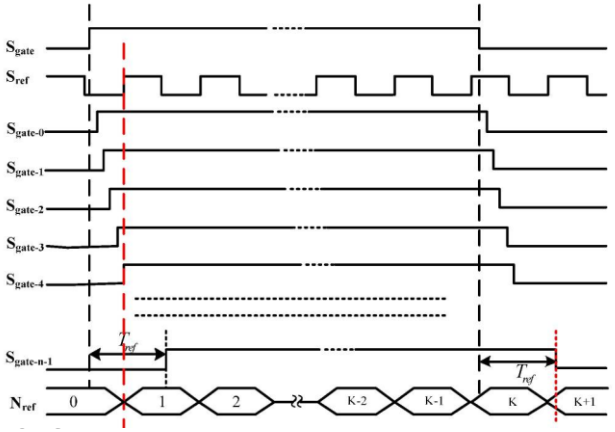
ALTERA Cyclone II EP2C8T144C6
Tap delay: 60 ps
Main clock : 400 MHz
TDC resolution : 12 ps

Xidian University

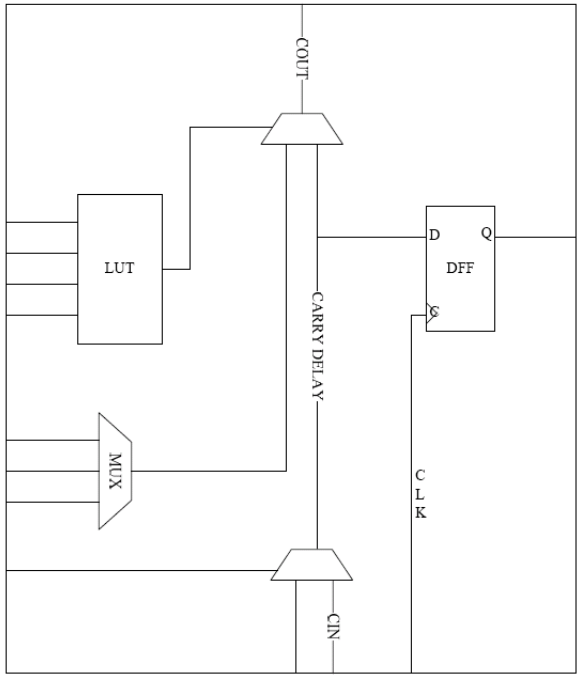


Min Zhang et all
Xidian University

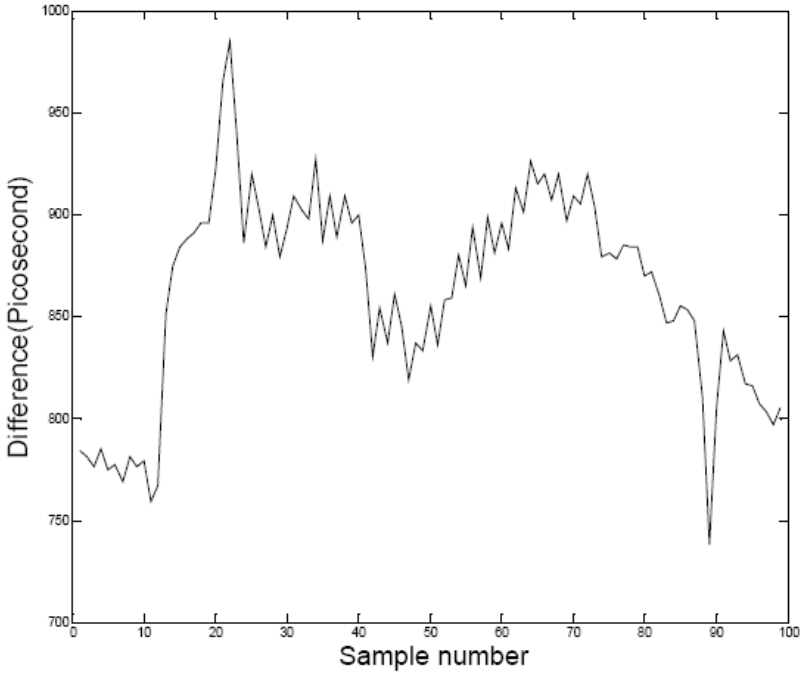
Xilinx Virtex-5 XC5VLX110T FPGA
7.6 ps TDC resolution



Tapped TDC



Differential nonlinearity

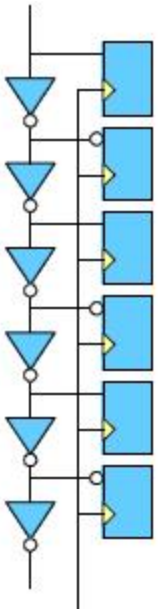


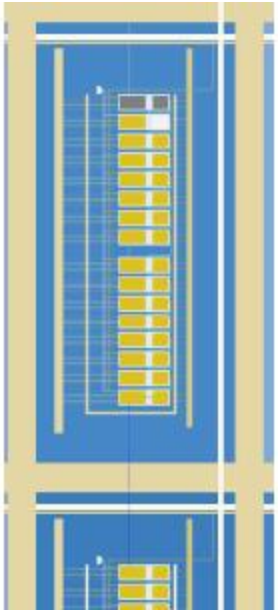
Time Resolution is 0 ps using Virtex 4 chip

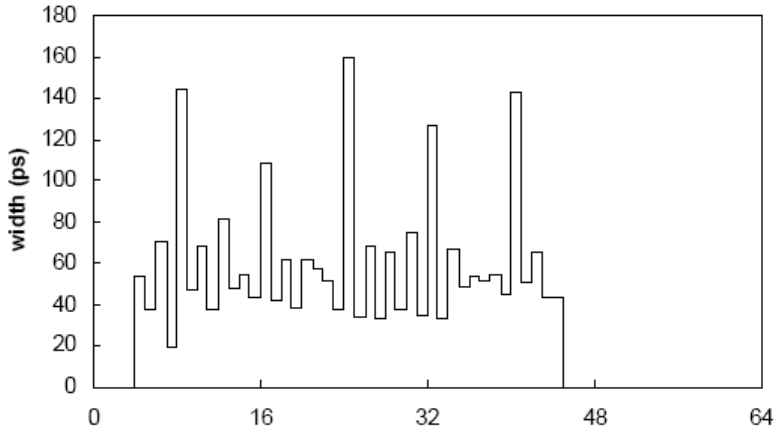
FPGA based TDC, next step

Jinyuan Wu and Zonghan Shi
Fermilab

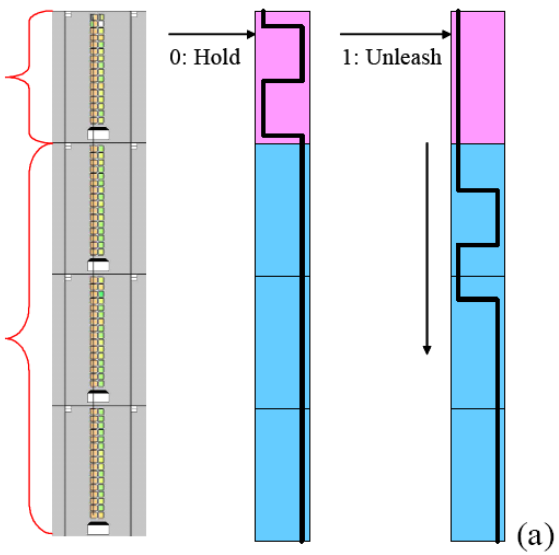
ALTERA Cyclone II EP2C8T144C6
Tap delay: 60 ps
Ultra-wide bin: 165 ps
Main clock : 400 MHz



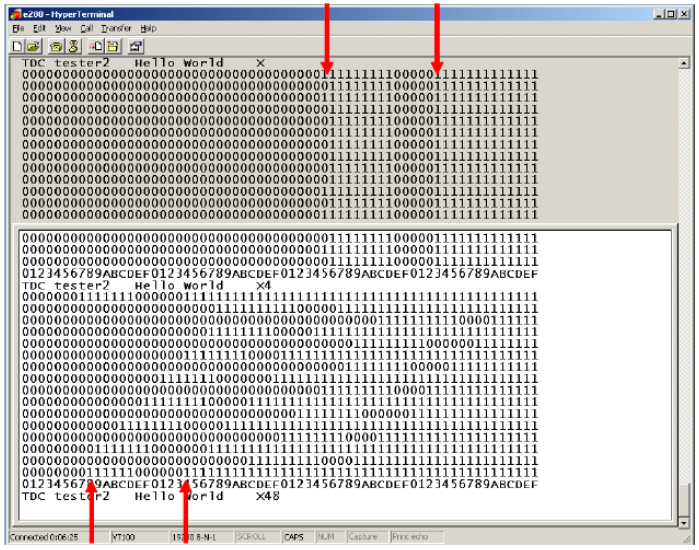




FPGA based TDC next step



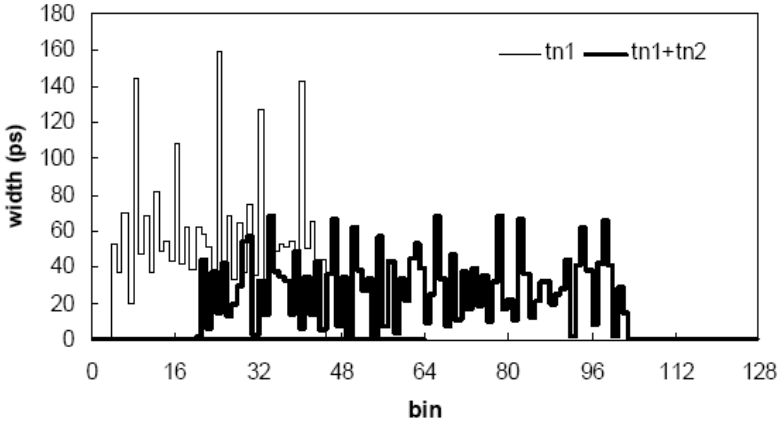
(a)



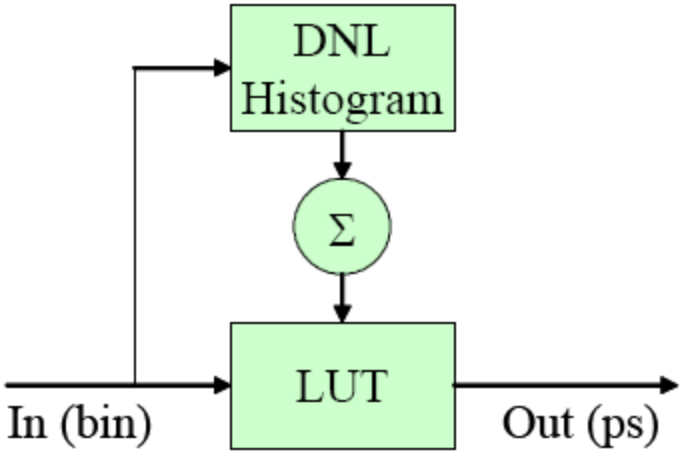
(b)

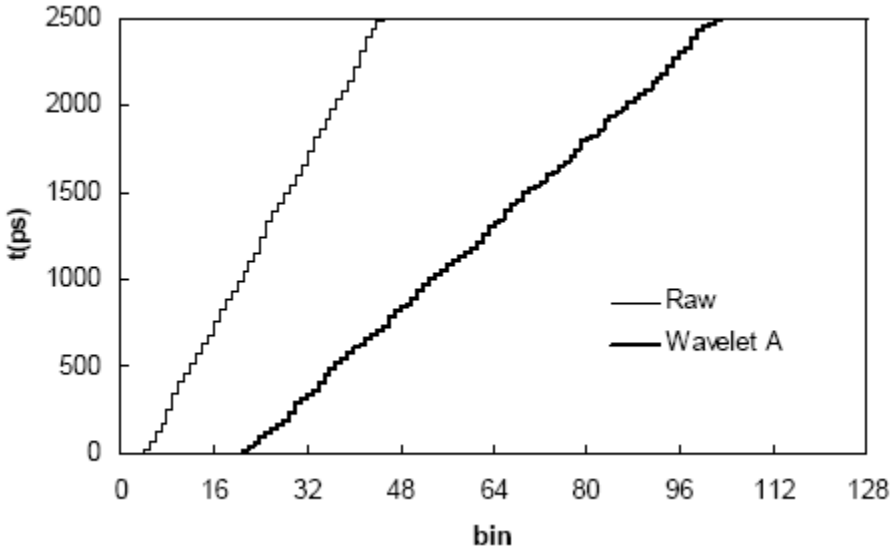
Wavelet launcher:

- Input pulse unleash bit pattern
- Multiple measurement



FPGA based TDC next step





Tapped TDC Resource Utilization

Device: EP2C8T144C6, Price: $28 (April 2008), Operating Frequency: 400MHz, Total Logic Elements: 8256						
	Max. bin width	Av bin width	ΔT RMS error	Dead time	Delay Chain Length	Logic Element Used
Raw TDC, Non-calibrated	165ps	60ps	58ps	2.5ns	64	1621 (20%)
Raw TDC, calibrated	165ps	60ps	40ps	2.5ns		
Wavelet TDC A, calibrated	65ps	30ps	25ps	5ns		
Wavelet TDC B, calibrated			12ps	45ns		1988 (24%)

Device utilization summary and power consumption.

		1024-Unit TDC		512-Unit TDC	
	Available	**Used**	**Utilization**	**Used**	**Utilization**
Slice Registers	69,120	1410	2%	602	0.9%
Slice LUTs	69,120	666	1%	327	0.5%
Occupied Slices	17,280	1265	7%	652	3.7%
Bonded IOBs	640	25	3%	25	3%
Block RAM/FIFO	148	2	1%	2	1%
Clock Resources	32	4	12%	2	6%
Number of routed lines		13,127		4937	
Dynamic Power Consumption		23 mW		9 mW	
Total Power Consumption		1.113 W		1.087 W	

Problems of Tapped FPGA TDC

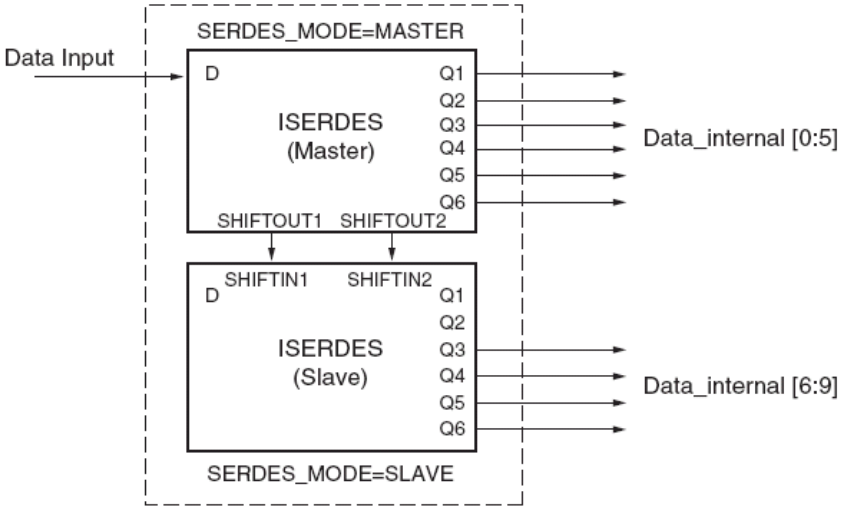
- Detailed analysis of FPGA circuit layout
- Advanced usage of placement constrains
- Variation of bin sizes due to differences in propagations through logic elements
- Consumes quite a lot of FPGA fabric resources

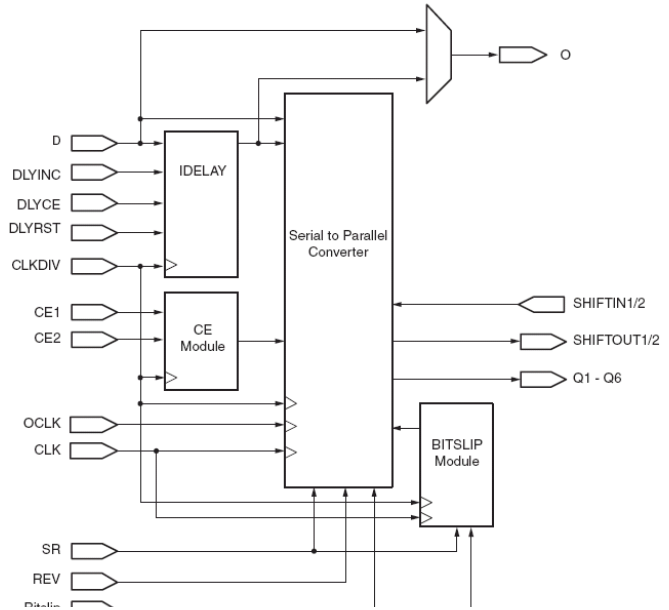
References :

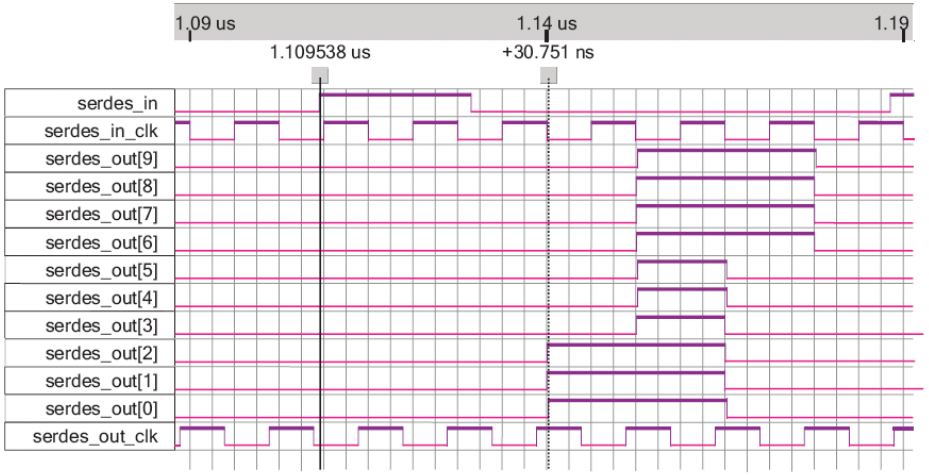
1. Jinyuan Wu et all. The 10-ps wave union TDC: Improving FPGA TDC resolution beyond its cell delay. November 2008 IEEE Nuclear Science Symposium conference record. Nuclear Science Symposium. DOI: 10.1109/NSSMIC.2008.4775079

2. Min Zhang et all. A 7.4 ps FPGA-based TDC with a 1024-unit measurement matrix April 2017. Sensors 17(4):865. DOI: 10.3390/s17040865

DeSerializer as TDC





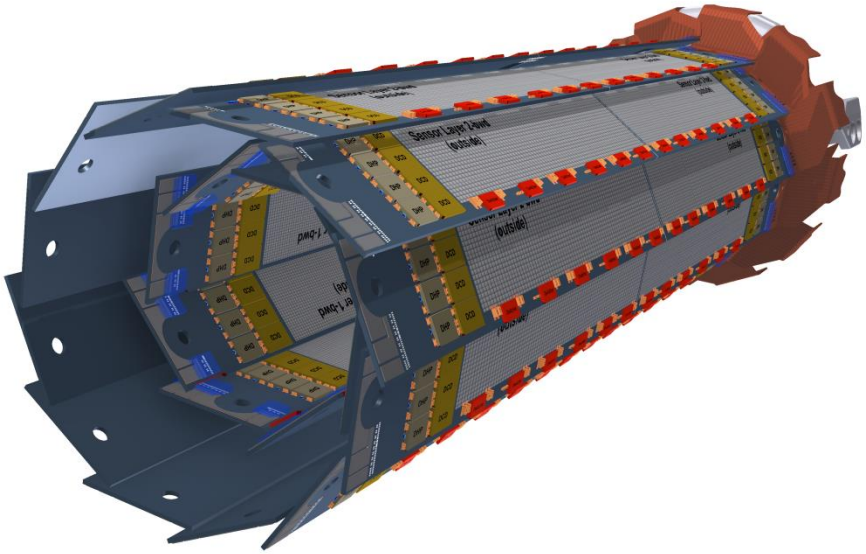


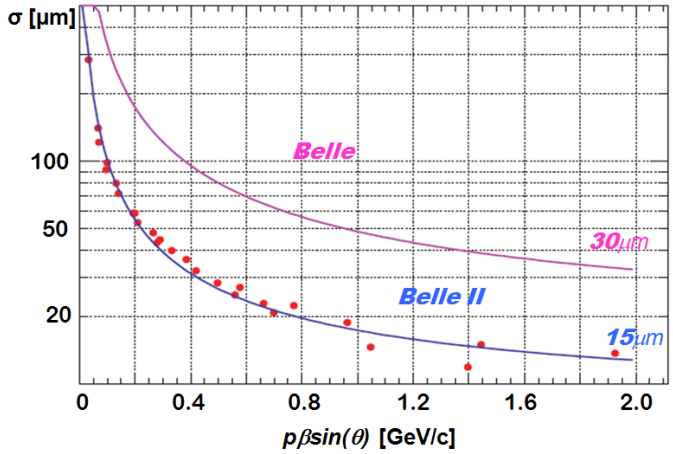
Features:

- LVDS input
- 64 taps for delay adjusting, one tap 7
- Delay controlled by Reference Clock

Clustering

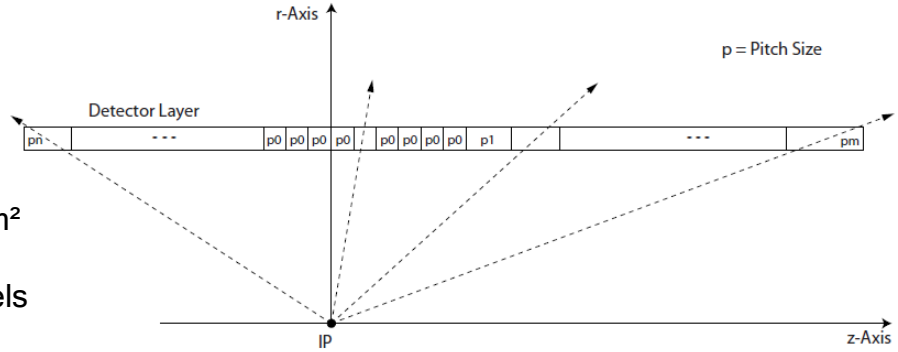
DEPFET PXD Detector for Belle2 Experiment





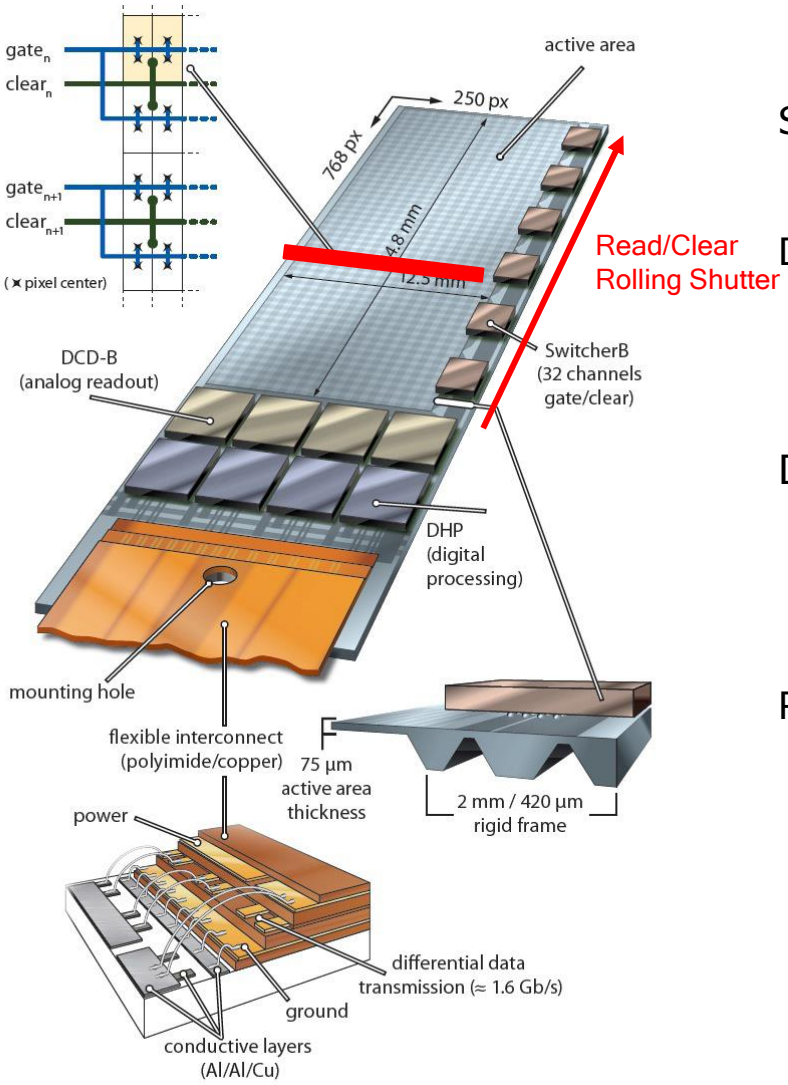
PXD

- Two layers :
  - L1 : 8 inner layers, 1.4 cm from IP, 44.8 x 12.5 mm²
  - L2 : 12 outer layers, 2.2 cm from IP, 61.44 x 12.5 mm²
- 40 half ladders => Half ladder 250x768 pixels => 7.68 Mpixels
- 75 um thick



Total material budget 0.2% $\mathbf{X}_0$

Detector Readout



Switcher :

- HV, controls DEPFET transistors and asserts Read/Clear signals

DCD (Drain Current Digitizer):

- 256 x PreAmp
- 256 x 8 bit ADC at 10 MHz
- MUX 1024 : 256 to DHPT running at 320Mbps each => 81.92Gbps

DHPT (Data Handling Processor)

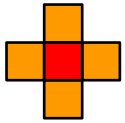
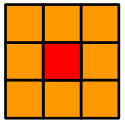
- Common mode correction
- Zero suppression
- Serial Link 1.6Gbps to DHE

Rolling shutter readout :

- 4 rows(1000 pixels) are readout and then cleared at once
- Speed 100ns for 4 rows => 19 us for full sensor or 2 KEKB revolution cycles

Clustering algorithm for FPGA

Simple requirements:

- Merging direct neighbors :  or 
- Pixel array 768x250
- Real time processing
  - 4 streams 50 10^6 pixel/second = 2 10^8 pixel/s
- Latency is not important
- Cluster data processing :
  - Center of gravity for ROI
  - Marking clusters created by low momentum particles

Literature

- **Most common case:**

Clustering for calorimeters with predefined shape 2x2,3x3, 5x5…

- **General purpose real time DCE3 clustering algorithm**
  - Parallel clustering
- **General purpose clustering(A.Annovi, M.Beretta) for ATLAS pixel detector**

Algorithm:

Each detector pixel is presented as FSM(Finite State Machine)

Detector of NxM pixels requires NxM FSMs

Clustering procedure:

Initialization , loading FSMs by hit information : EMPTY, HIT

Readout :

- external FSM selects first not empty Pixel and reads it
- SELECT signal propagates to neighboring FSMs for further readout
- this procedure is repeated till all neighboring pixels are readout

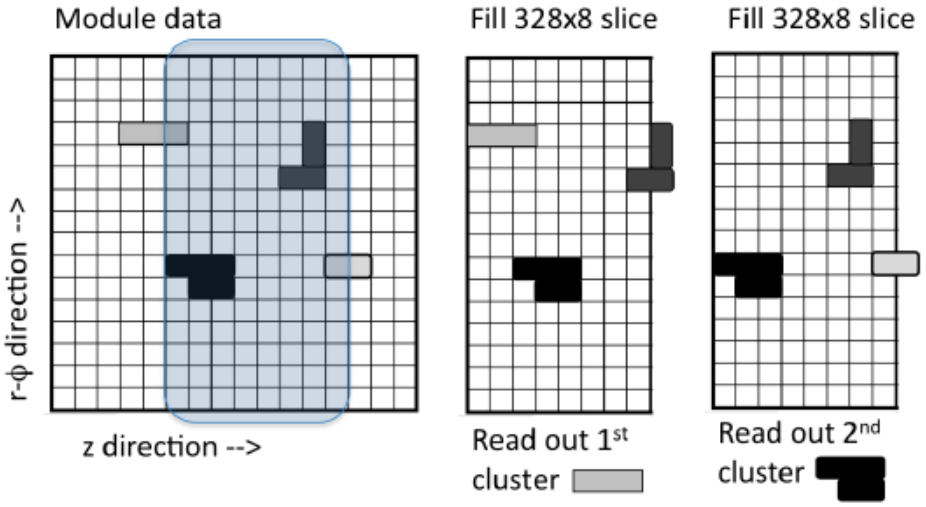
Literature

Problem: amount of hardware scales linearly with number of pixels and very fast uses up all FPGA resources:

grid size	clock period	area usage
8x8	6ns	1%
120x8	13ns	5%
32x32	13ns	6%
64x32	16ns	11%
256x8	15ns	11%
328x8	17ns	16%
120x32	20ns	21%

Table 1: Algorithm performances on a xc5vlx330 FPGA.

Solution to the problem : "Sliding Window". Window is bigger than any cluster



FPGA : XC5VLX155
Window : 328x8
30 % FPGA resources
Speed : >20 Mhits/s

New clustering algorithm for FPGA

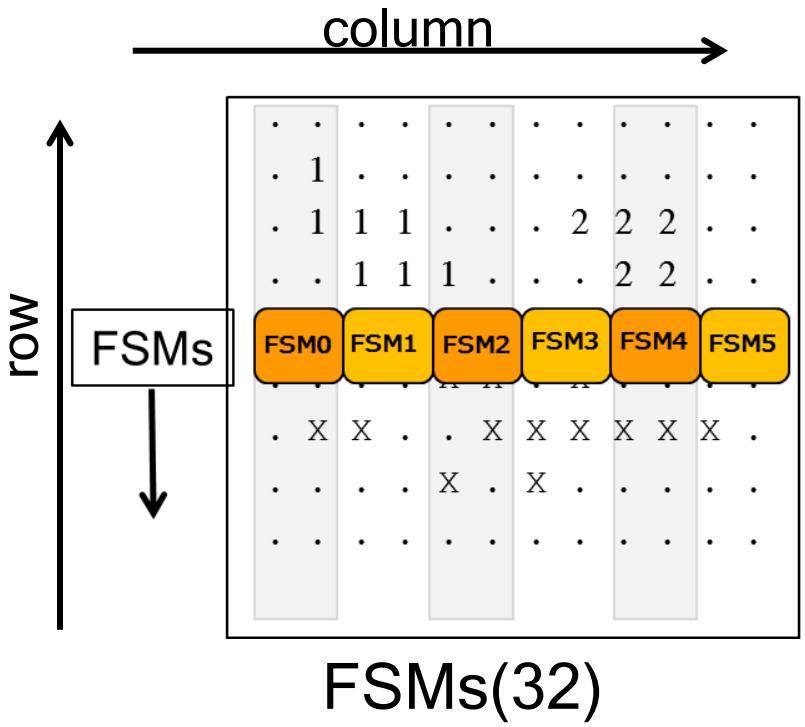
Algorithm takes advantage of detector readout feature:

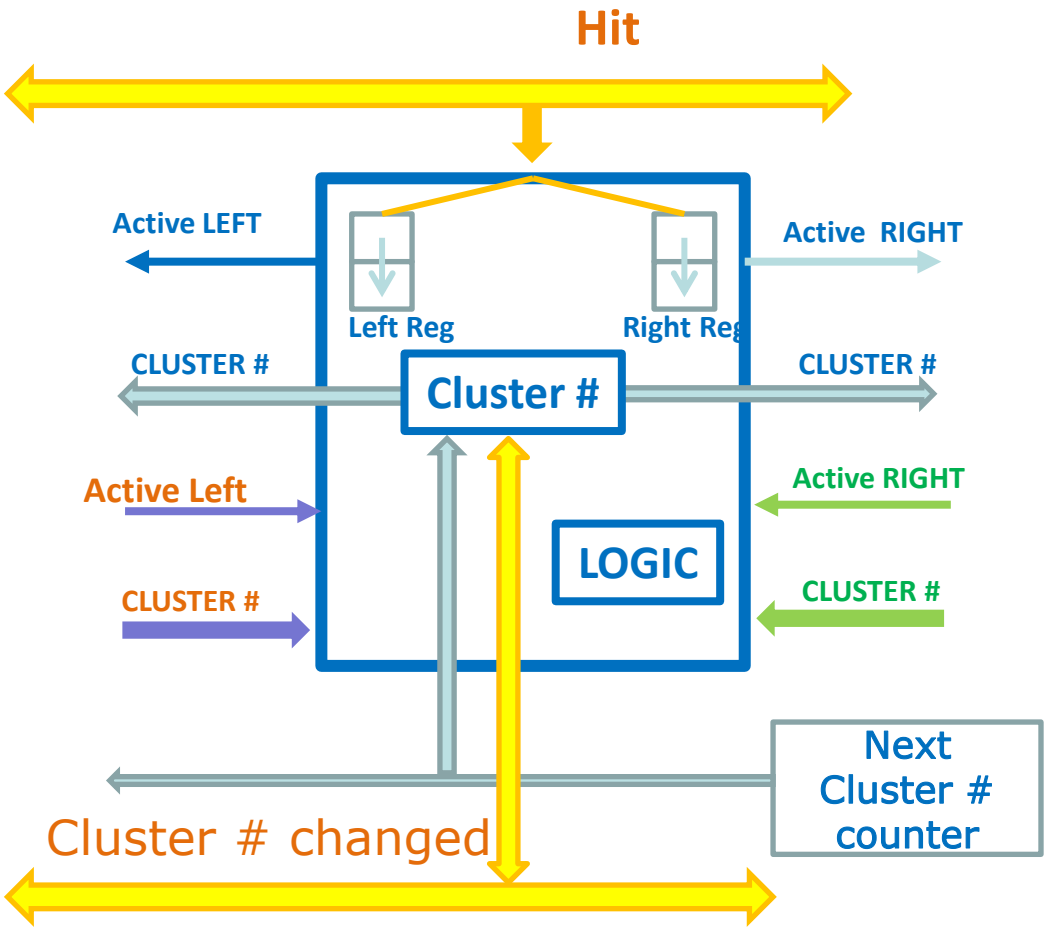
- sequential data transmission
- limited data rate: not more than 4 x 76 MPix/s
- Ordered hits readout sequence, almost row wise:
  data mixed within 4 consecutive rows
- no latency requirements

Clustering algorithm features:

- Hit information is analyzed once and cluster number assigned
- Following processing steps shuffle hits using cluster number information
- Clustering algorithm reconstructs any cluster shape within half ladder
- Pipeline design – real time operation

Clustering FSMs





FSM behavior

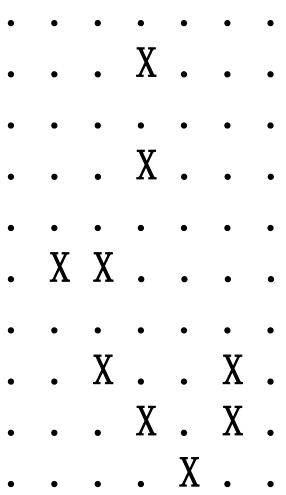
What **FSMs** do?

- Each FSM responsible for hits of two columns
- Process one hit in one clock cycle
- Evaluate **hit cluster number**
- Write hit together with cluster number to hit memory
- Store cluster number in **cluster memory**
- When two clusters touch each other the lowest cluster number is taken over

FSM behavior is described for all cases

1. FSM is not active, hit arrives
2. FSM is active, no hit
3. FSM is active and new hit arrives
4. FSM is active, there was no hit belonging to any of these two rows within this column and current hit is a first belonging to new column

Examples of FSMs actions 1



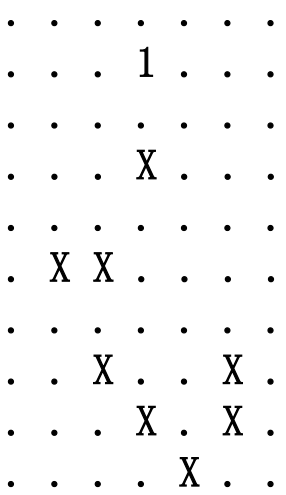
Next cluster counter

1

Cluster memory

Addr.	Value
1	-
2	-
3	-
4	-
5	-

Examples of FSMs actions 1



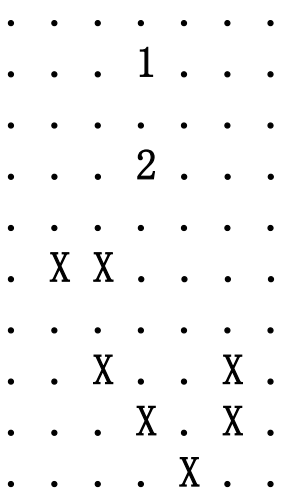
Next cluster counter

2

Cluster memory

Addr.	Value
1	1
2	-
3	-
4	-
5	-

Examples of FSMs actions 1



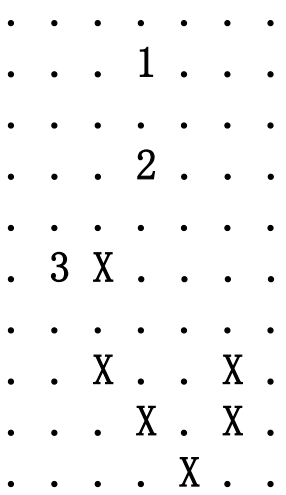
Next cluster counter

3

Cluster memory

Addr.	Value
1	1
2	2
3	-
4	-
5	-

Examples of FSMs actions 1



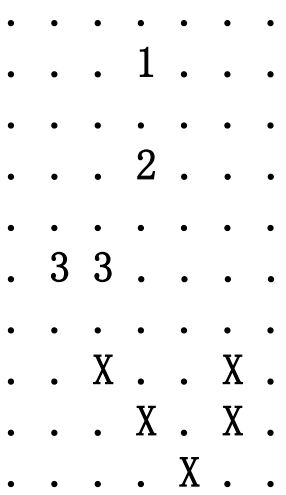
Next cluster counter

4

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	-
5	-

Examples of FSMs actions 1



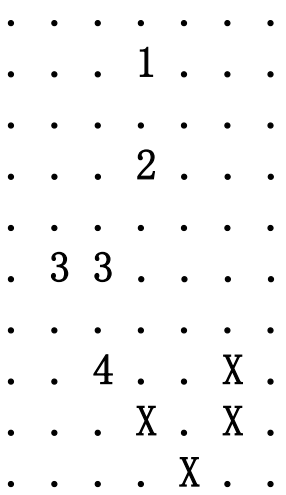
Next cluster counter

4

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	-
5	-

Examples of FSMs actions 1



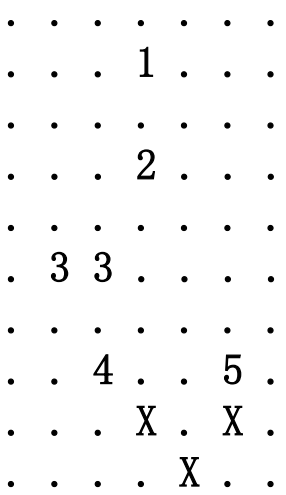
Next cluster counter

5

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	4
5	-

Examples of FSMs actions 1



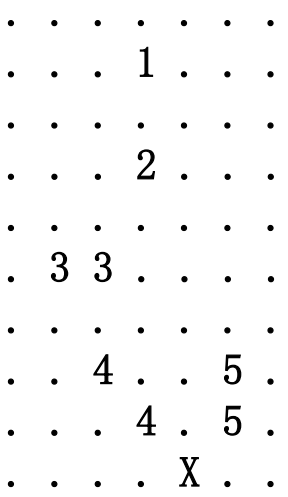
Next cluster counter

6

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	4
5	5

Examples of FSMs actions 1



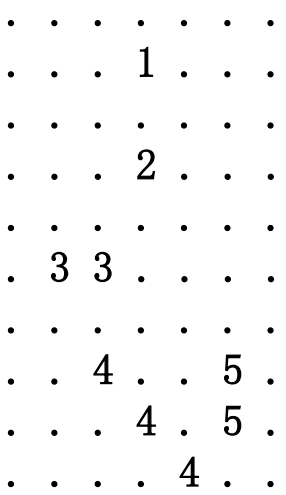
Next cluster counter

6

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	4
5	5

Examples of FSMs actions 1



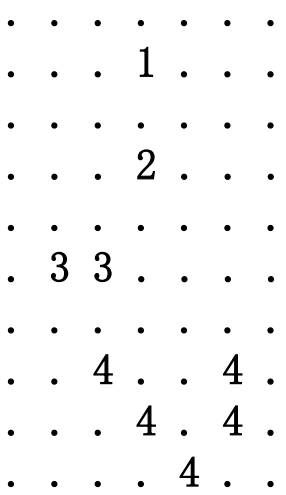
Next cluster counter

6

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	4
5	4

Examples of FSMs actions 1



Next cluster counter

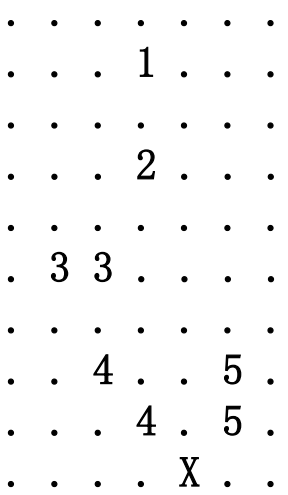
6

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	4
5	4

Cluster numbers are updated using cluster memory during hit readout.

Examples of FSMs actions



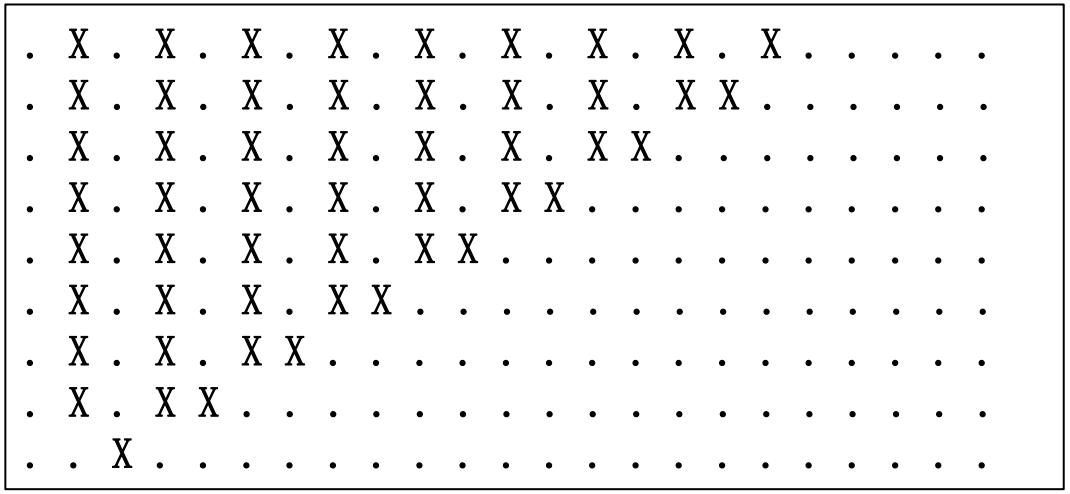
Next cluster counter

6

Cluster memory

Addr.	Value
1	1
2	2
3	3
4	4
5	5

Extreme case



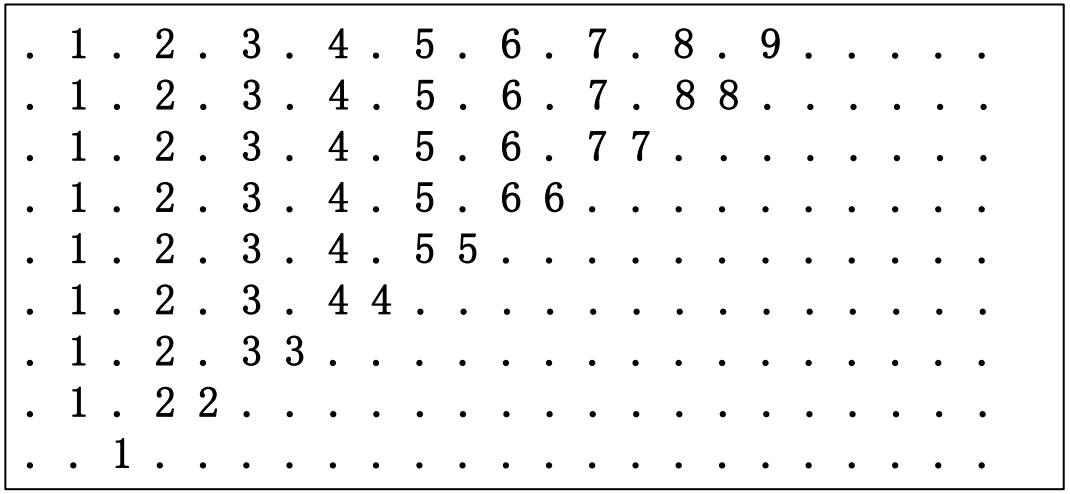
Next cluster counter

1

Cluster memory

Addr.	Value
1	-
2	-
3	-
4	-
5	-
6	-
7	-
8	-
9	-

Extreme case



Next cluster counter

10

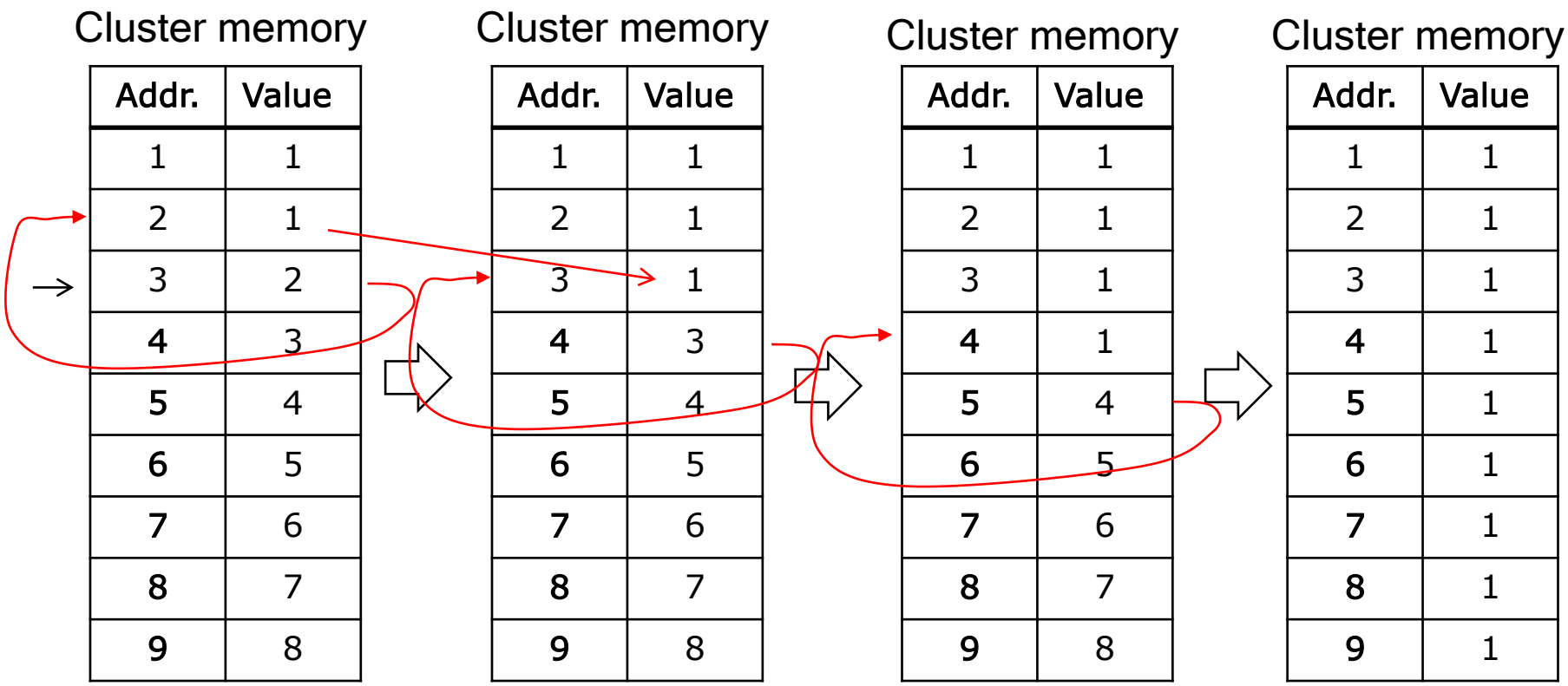
Cluster memory

Addr.	Value
1	1
2	1
3	2
4	3
5	4
6	5
7	6
8	7
9	8

Cluster memory content has to be updated to map all cluster numbers to smallest number

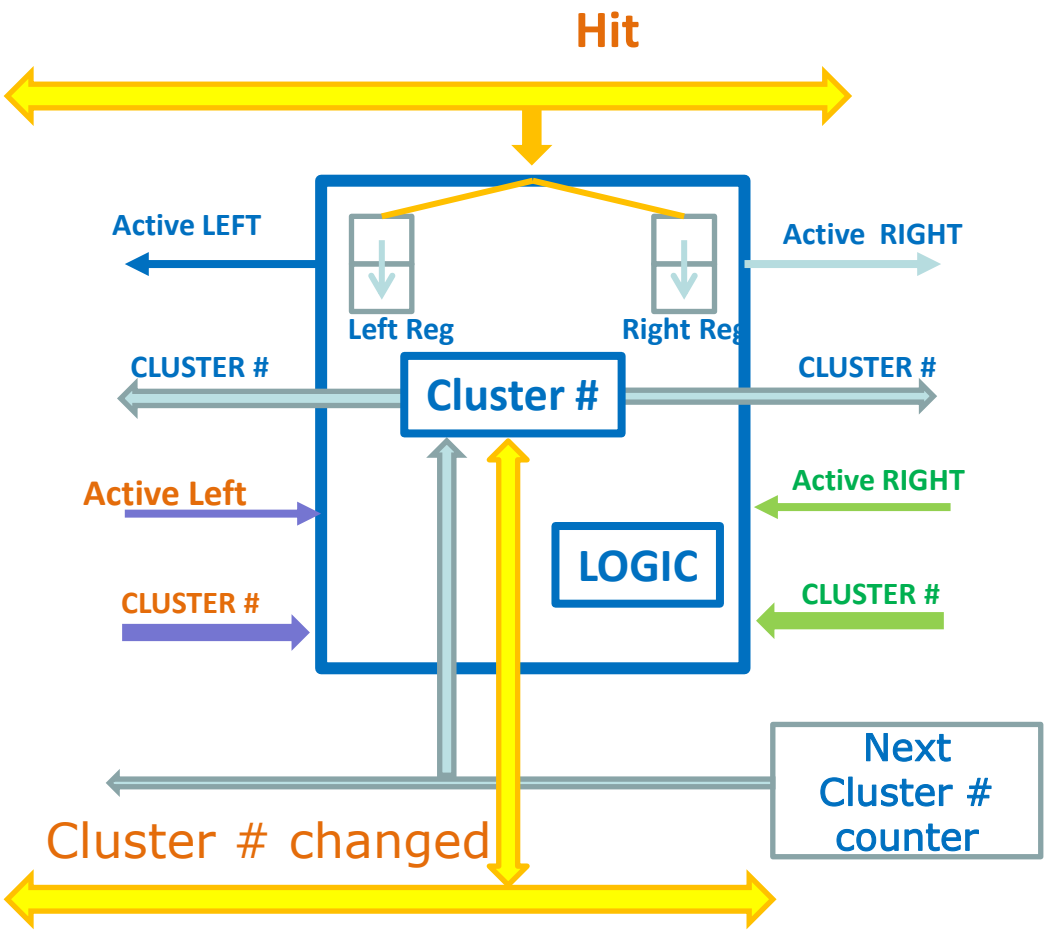
Cluster memory update

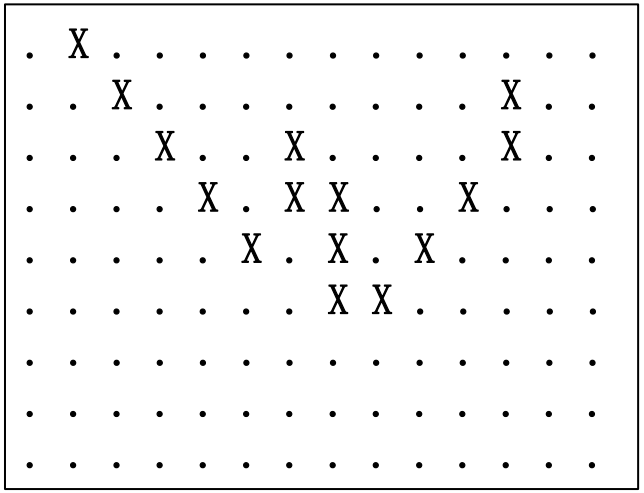
Update starts from cluster #3 to maximum cluster #
Maximum look down length is 1 clusters



One update takes 3 clock cycles
Using DP memory allows to reach 2 clock cycles per update

Broadcast Cluster Number change (1)



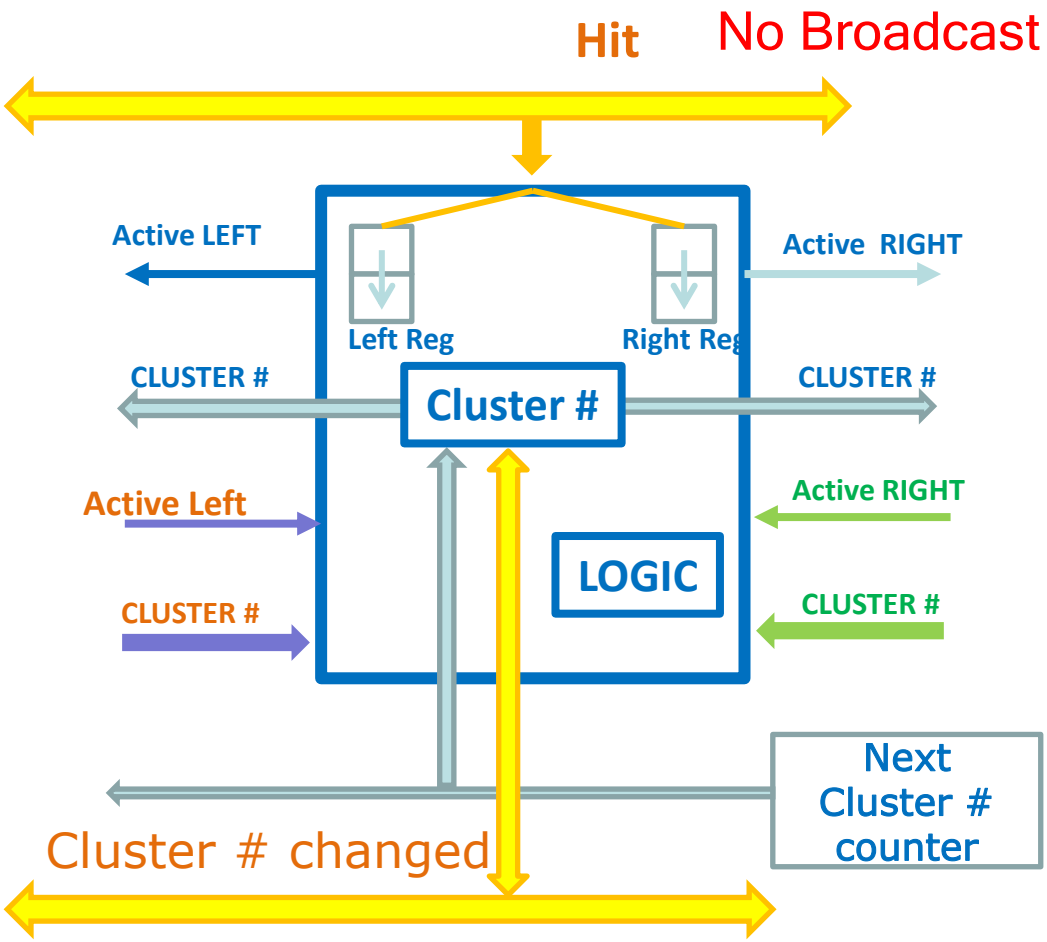


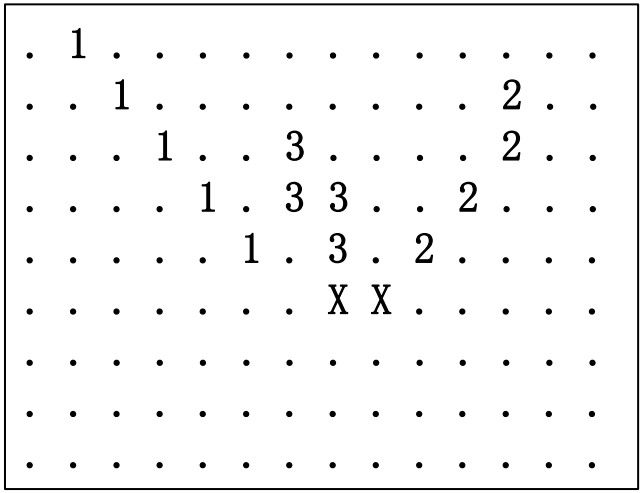
Cluster memory

Addr.	Value
1	-
2	-
3	-
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Broadcast Cluster Number change (2)



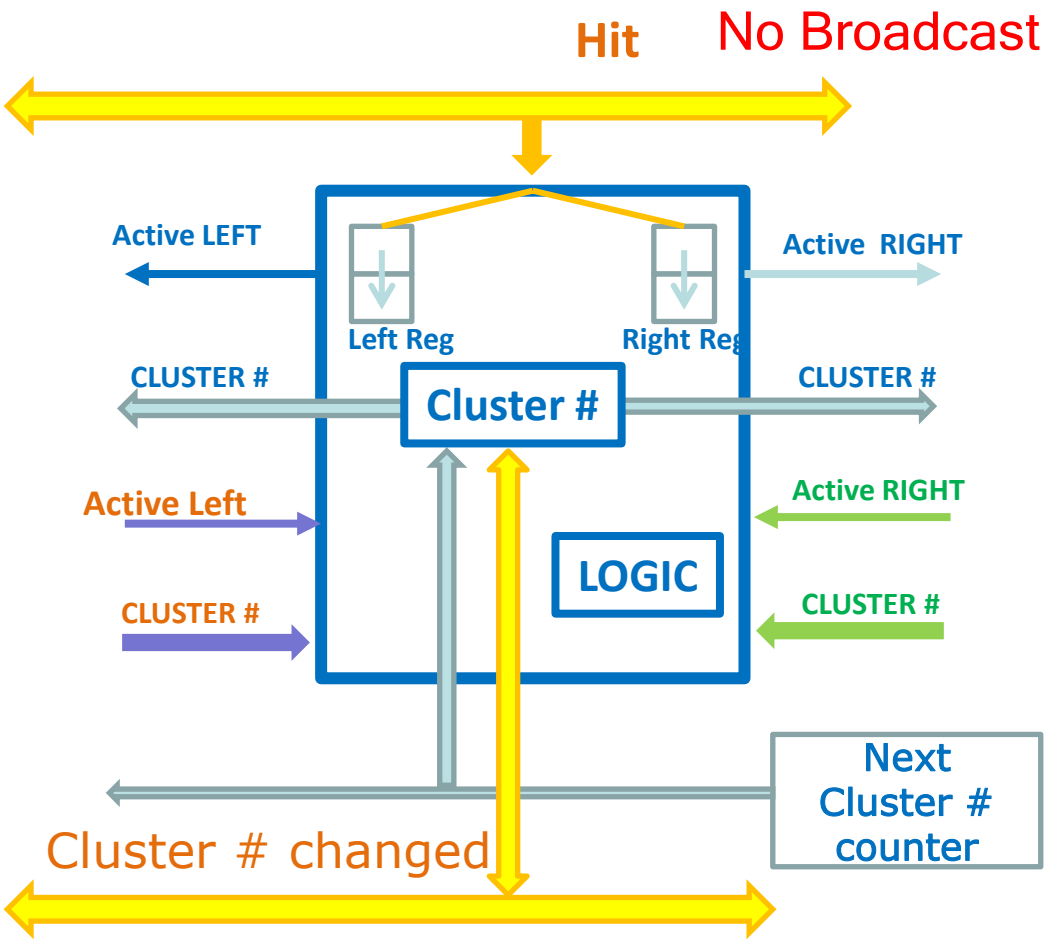


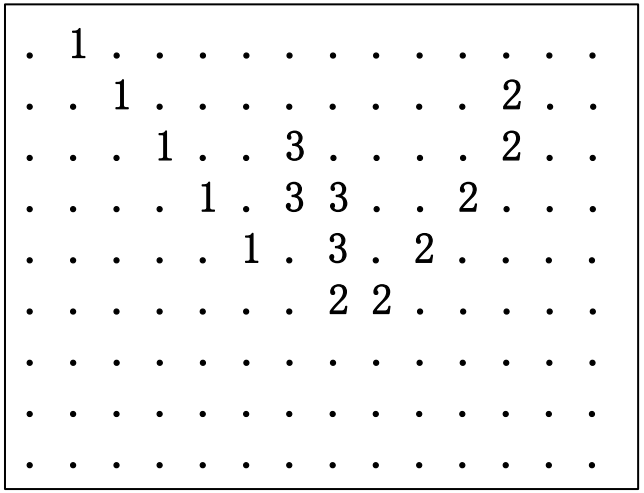
Cluster memory

Addr.	Value
1	1
2	2
3	1
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Broadcast Cluster Number change (3)



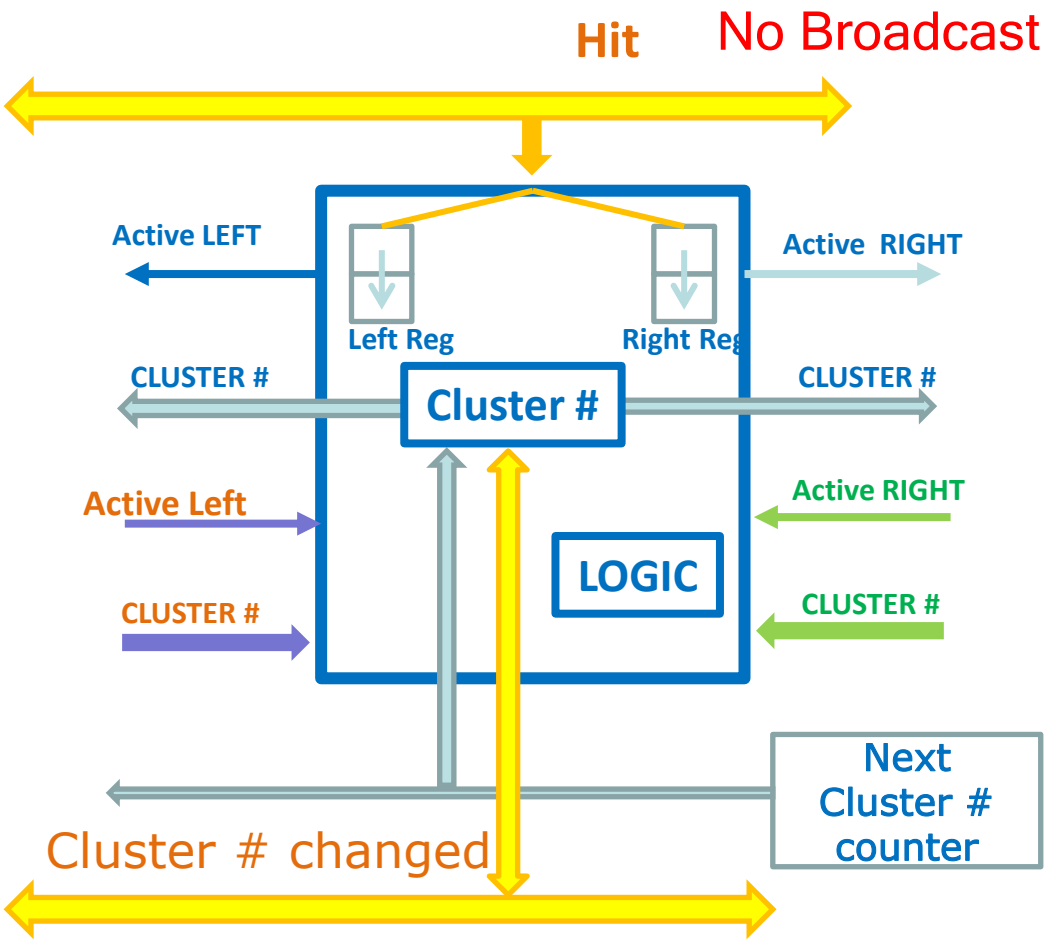


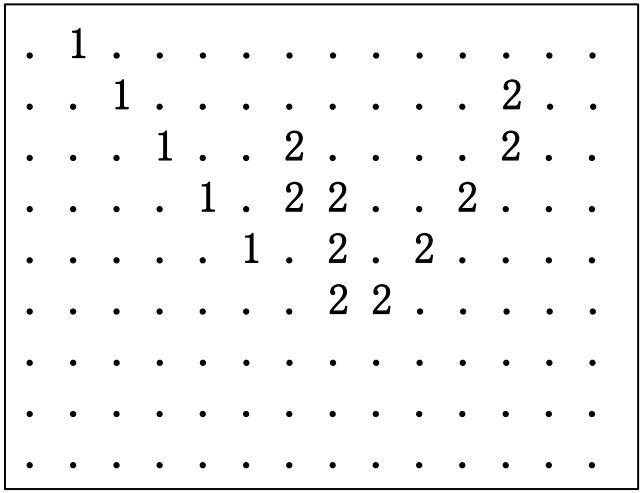
Cluster memory

Addr.	Value
1	1
2	2
3	2
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Broadcast Cluster Number change (4)



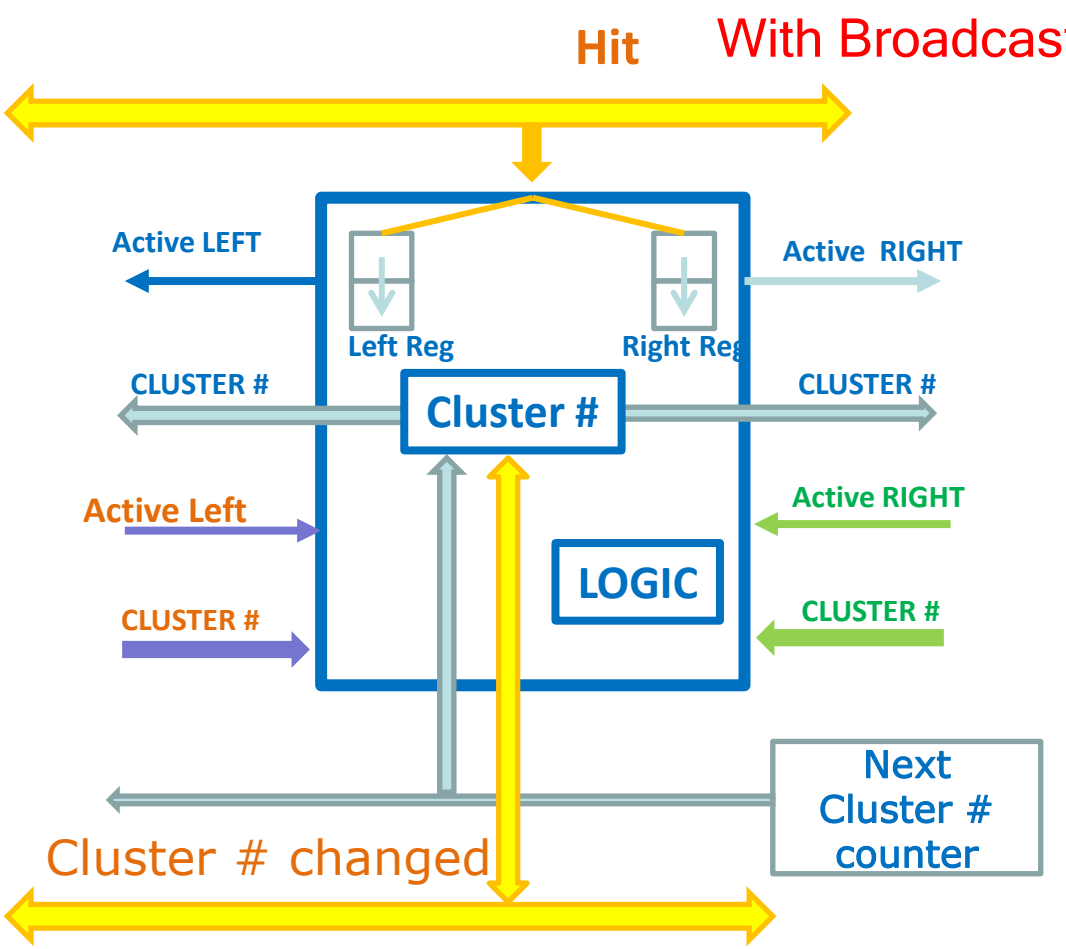


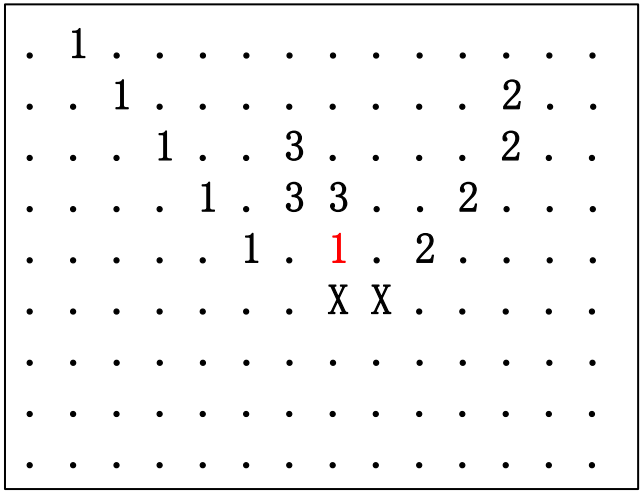
Cluster memory

Addr.	Value
1	1
2	2
3	2
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Broadcast Cluster Number change (5)



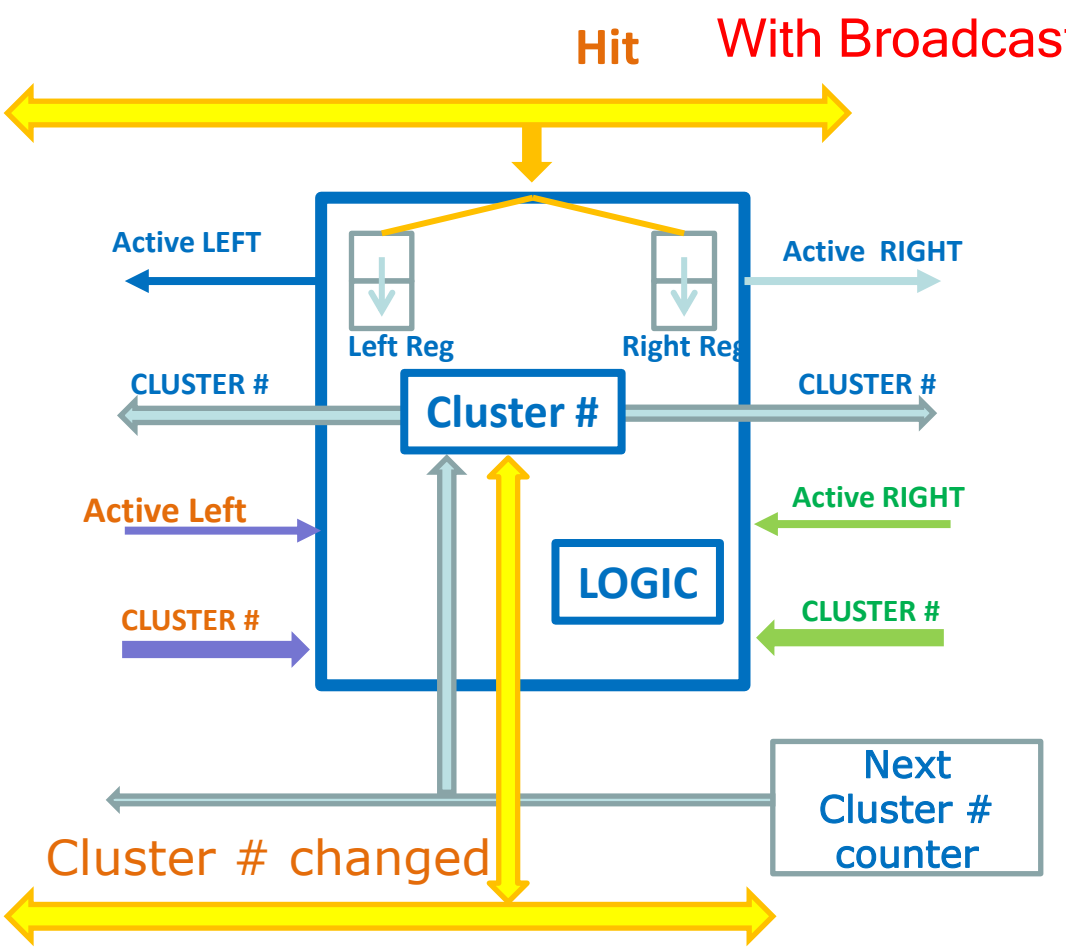


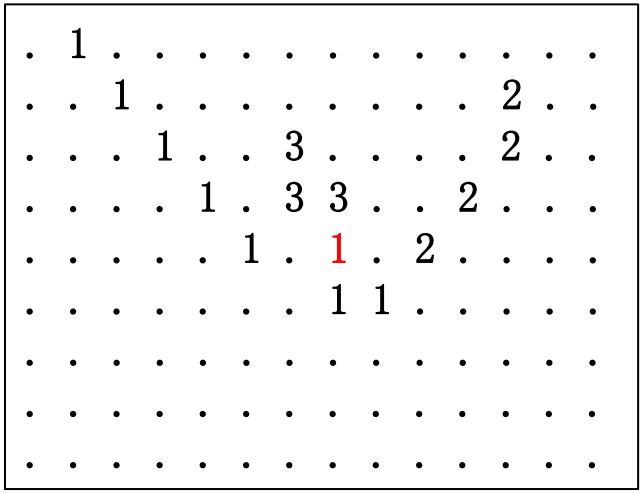
Cluster memory

Addr.	Value
1	1
2	2
3	1
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Broadcast Cluster Number change (6)



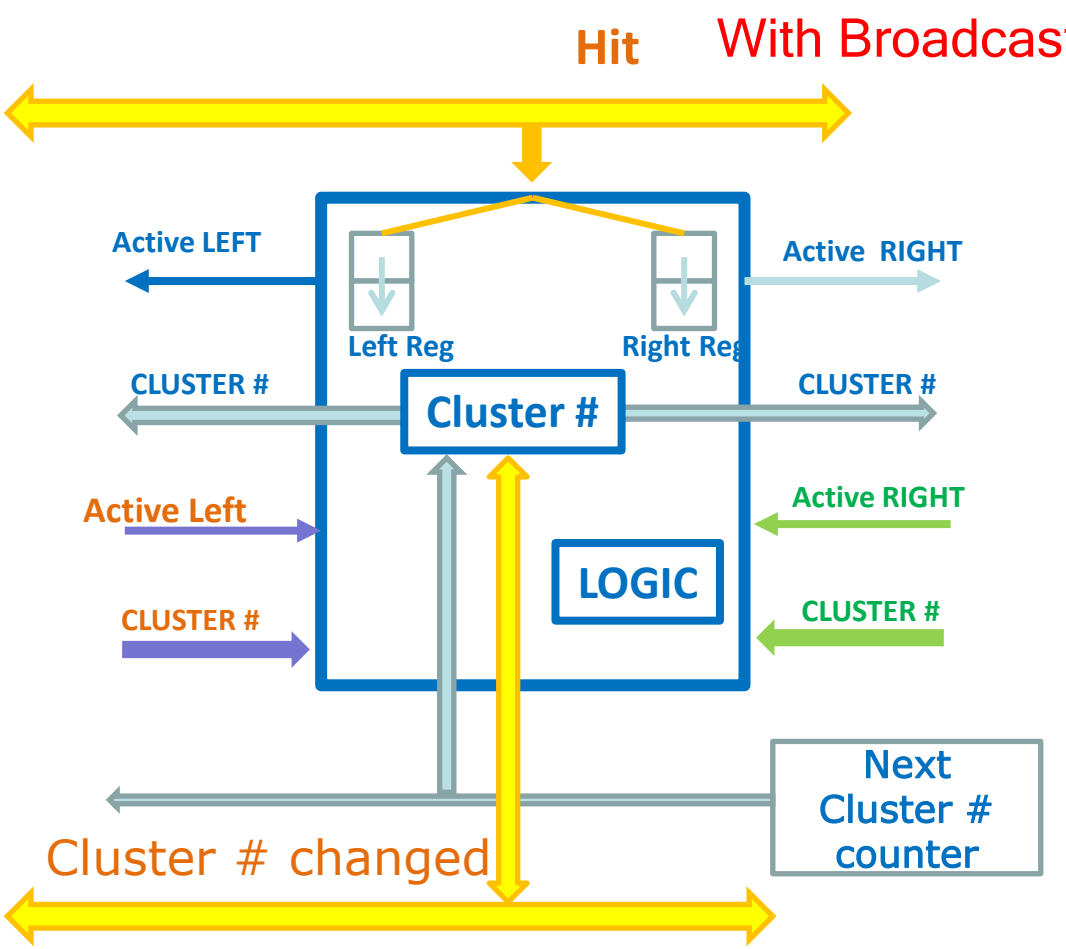


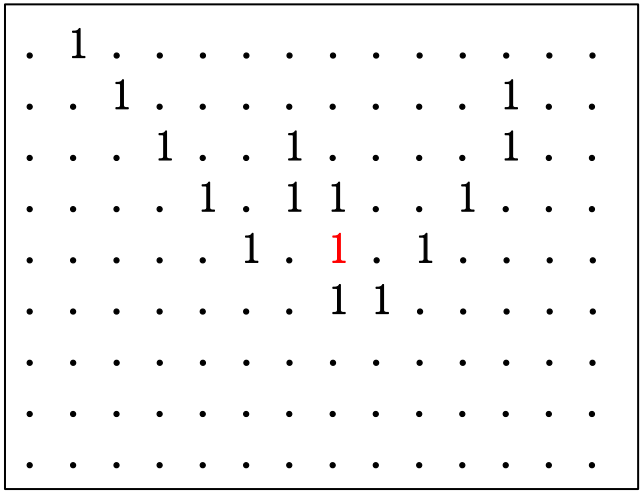
Cluster memory

Addr.	Value
1	1
2	1
3	1
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Broadcast Cluster Number change (6)





Cluster memory

Addr.	Value
1	1
2	1
3	1
4	-
5	-

When FSM changes Cluster# from B to A it broadcast change to all FSMs.
Active FSMs with cluster number B also change its' cluster number to A.

Resources and speed

Clustering FSMs 64 columns and 4k clusters:

- 7% of Slices of XC6VLX130T
- 1% of Memory blocks (6 out of 264)
- 100 Mhits/s clock

Clustering FSMs 128 columns and 4k clusters:

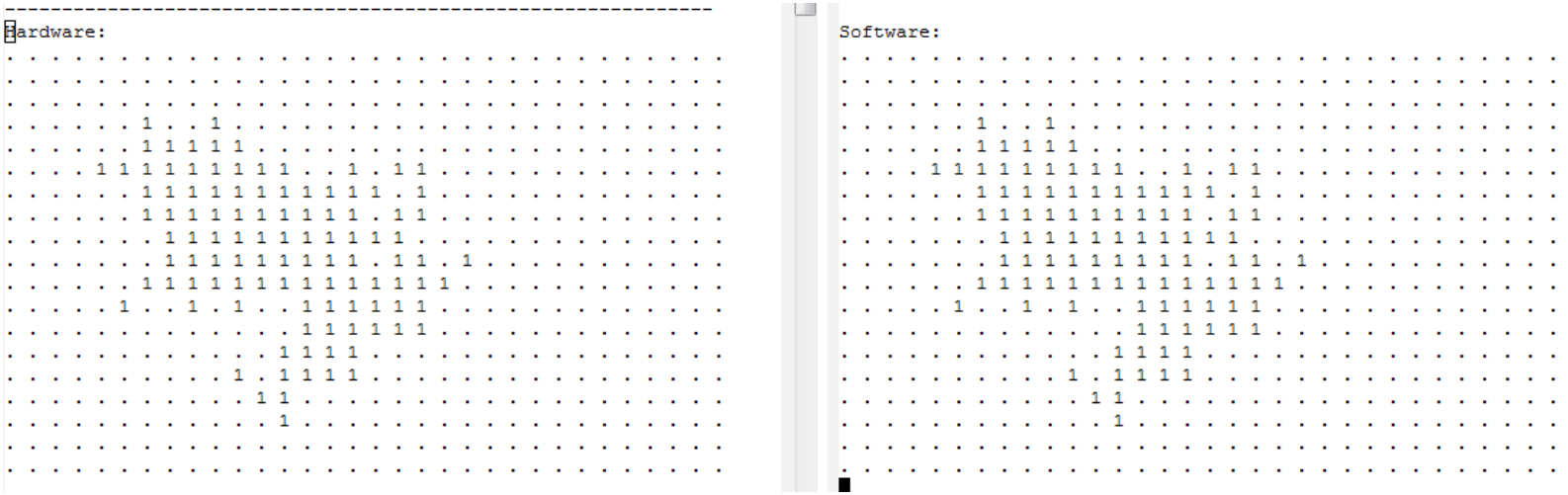
- 13% of Slices
- 1% of Memory
- 80 Mhits/s

Clustering for 250 columns and 8k clusters:

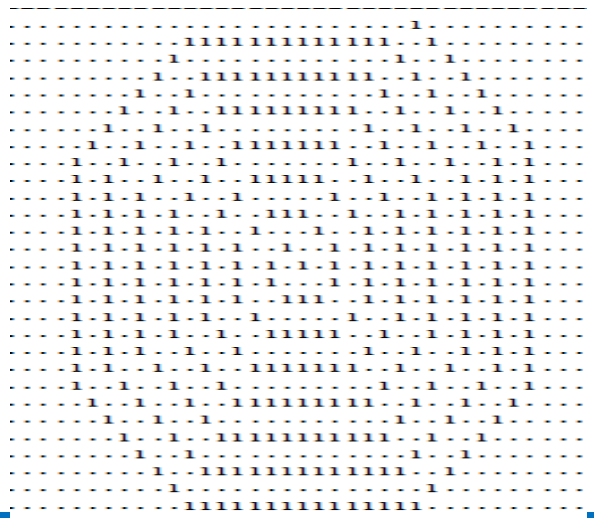
- 30% of slices
- 30% memory blocks
- 100 Mhits/s

Test in hardware

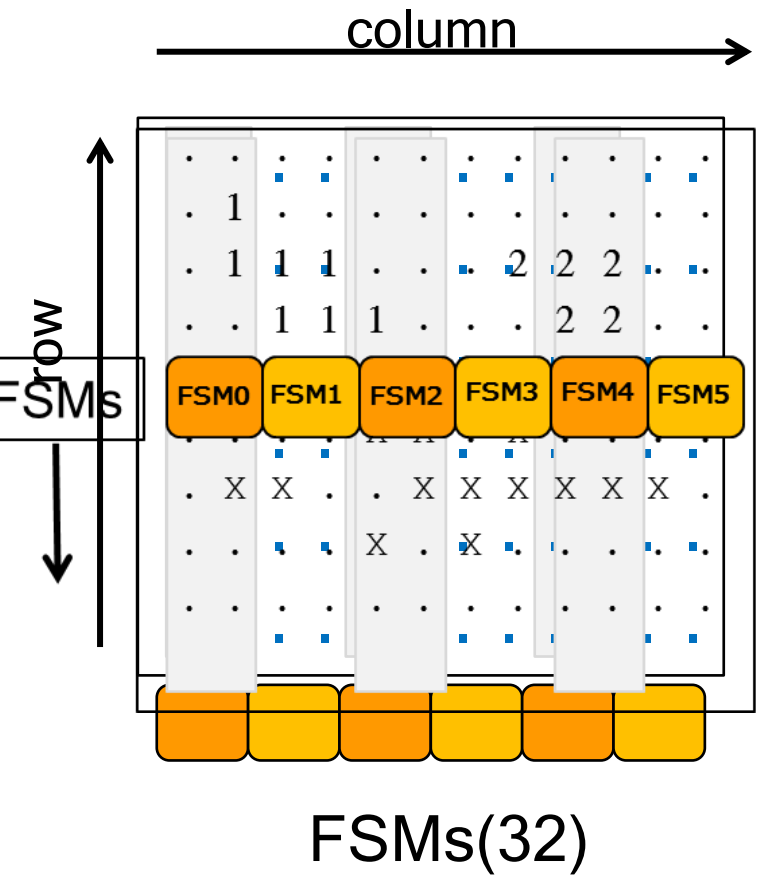
Compare hardware and software results



Test of algorithm



Clustering FSMs



FSMs(32)