

FPGA implementation of the HL-LHC CMS Drift Tubes Level-1 Trigger Algorithm

A. Navarro Tobar
on behalf of the CMS DT Upgrade group

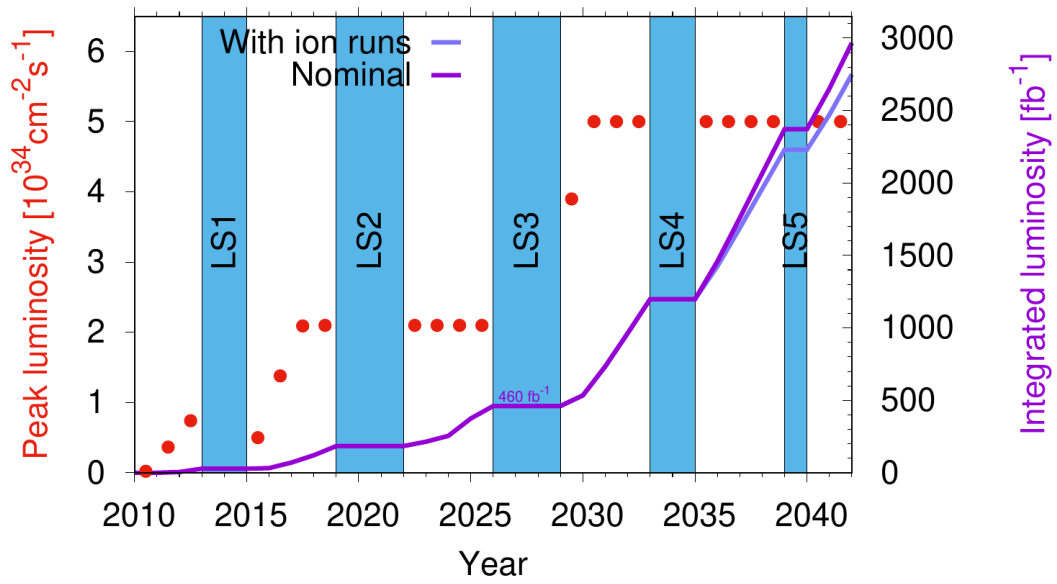
Ciemat



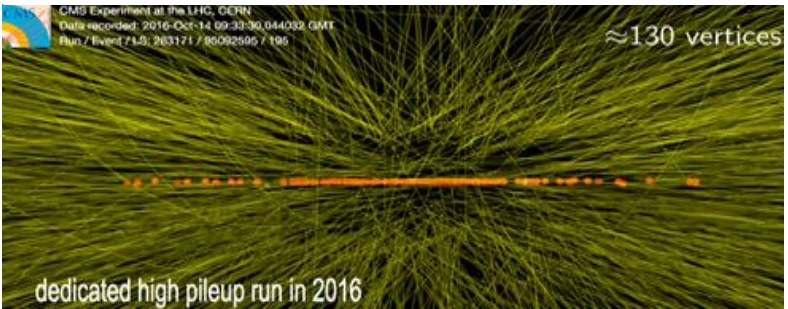
Proyecto PID2020-116262RB-C42 financiado por:



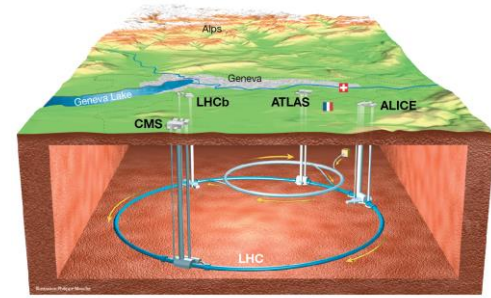
LHC High-Luminosity Upgrade



- LHC nominal design $10^{34} \text{ cm}^{-2}\text{s}^{-1}$ and 13.6 TeV (as of today)
- HL-LHC upgrade will allow to continue the physics exploration in the high energy regime
- Important goals to characterize the Higgs boson discovered at the LHC
- Increase of energy is preferred but much more difficult => integrated luminosity increase by a factor ~ 10
- (Instantaneous luminosity increase by a factor ~ 5)



- Important challenges for the detectors:
- Higher radiation, higher occupancy, higher rates, higher pileup, etc
- Replacement of a large fraction of CMS is required





CMS HL-LHC Upgrade: Scope

Technical proposal CERN-LHCC-2015-010 <https://cds.cern.ch/record/2020886>

Scope Document CERN-LHCC-2015-019 <https://cds.cern.ch/record/2055167/files/LHCC-G-165.pdf>

Complete replacement**

L1-Trigger/HLT/DAQ**

<https://cds.cern.ch/record/2283192>

<https://cds.cern.ch/record/2283193>

- Tracks in L1-Trigger at 40 MHz
- PFlow-like selection 750 kHz output
- HLT output 7.5 kHz

Calorimeter Endcap**

<https://cds.cern.ch/record/2293646>

- 3D showers imaging for pattern recognition
- Precision timing for PU mitigation
- Si, Scint+SiPM in Pb/W-SS

Tracker**

<https://cds.cern.ch/record/2272264>

- P_T module design for tracking in L1-Trigger
- Extended coverage to $\eta \approx 3.8$
- Much reduced material budget
- Si-Strip and Pixels increased granularity

Barrel Calorimeters*

<https://cds.cern.ch/record/2283187>

- ECAL crystal granularity readout at 40 MHz
- Precision timing for e/γ at 30 GeV, for vertex localization ($H \rightarrow \gamma\gamma$)
- ECAL and HCAL new Back-End boards

Major Electronics Upgrade/ Consolidation*

Muon systems* ***

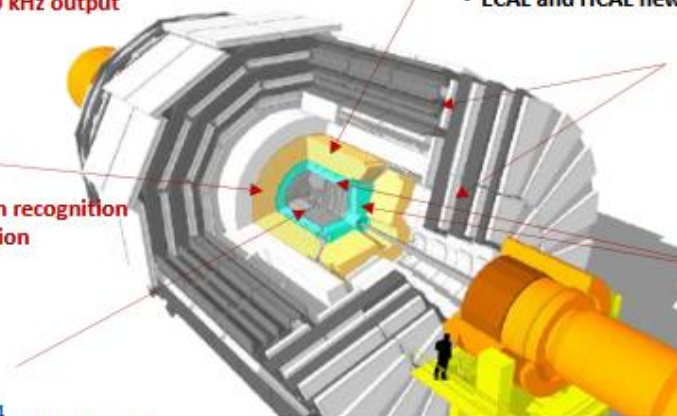
<https://cds.cern.ch/record/2283189>

- Extended GEM coverage to $\eta \approx 3$
- DT & CSC new FE/BE readout
- RPC back-end electronics
- New GEM/RPC $1.6 < \eta < 2.4$

MIP Timing Detector***

<https://cds.cern.ch/record/2296612>

- Precision timing for PU mitigation
- Barrel layer: Crystals + SiPMs
- Endcap layer: Low Gain Avalanche Diodes



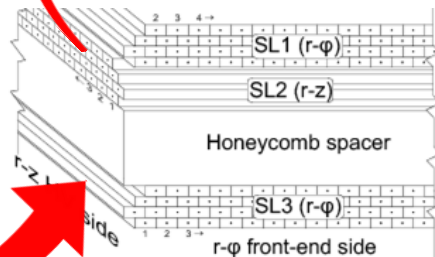
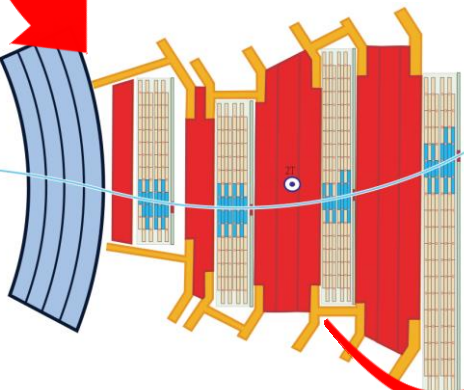
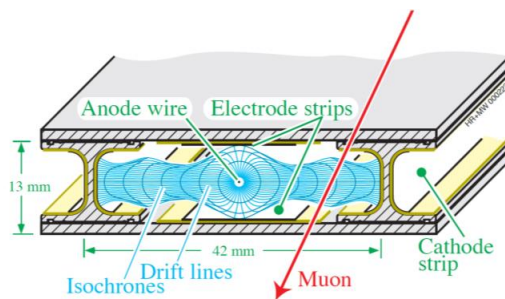
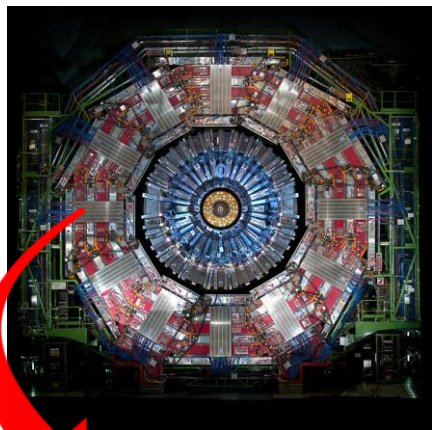
Beam Radiation Instr. and Luminosity
Common Systems and Infrastructure

<https://cds.cern.ch/record/2020886>

New Detector System***

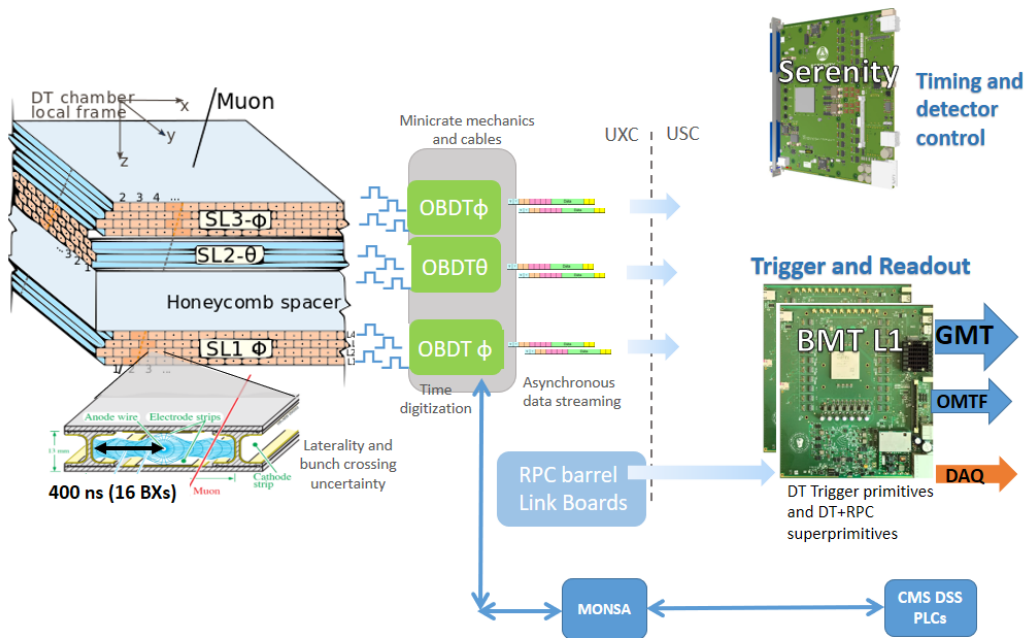
New paradigms for a HEP experiment to meet the unprecedented challenges and fully exploit the HL-LHC luminosity and physics potential

CMS muon detector - DTs



- Compact Muon Solenoid: general purpose detector at the LHC
- Muon Barrel: outermost part of the central wheels of CMS, identifies, triggers and tracks muons (minimum ionizing particles that can cross CMS iron yokes)
- Drift Tubes: 5 wheels x 12 sectors, 4 chambers, each contains 3 SuperLayers (4-layer blocks) provide information for phi (2SL) and theta (1SL) views
- DT subdetector consists on 172k Drift Cells. Drift time provides information on the position of muon (the farthest from wire, the later the hit arrives). Drift velocity $54.5 \mu\text{m/ns}$

DT Electronics for High-Luminosity LHC

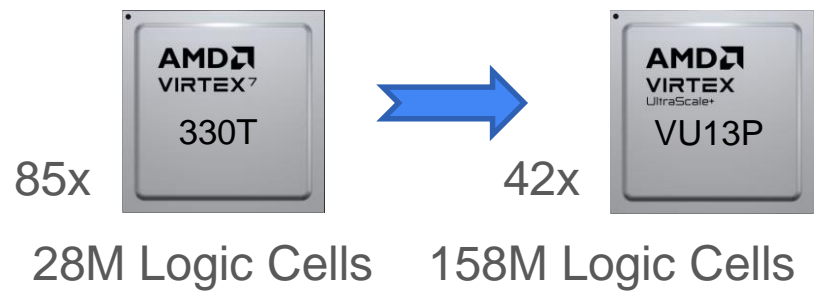


Trigger/readout architecture changes

Full data streaming to Underground Service Cavern (USC) of all DT data (no filtering)

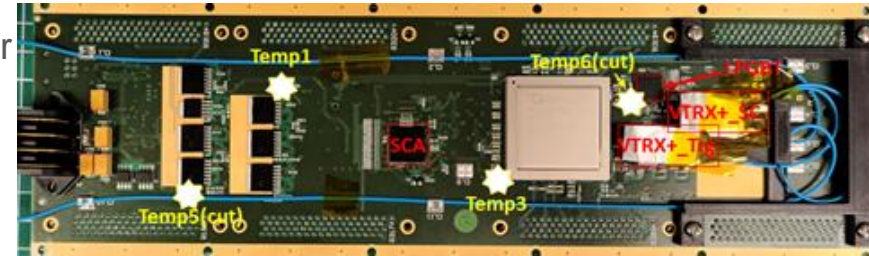
Trigger can done at USC (no radiation): bigger and faster FPGAs (before on-detector ASIC + medium sized FPGAs)

Achieve offline-grade (SW) performance at Level-1 Trigger (HW)



On-detector electronics: OBDT-theta

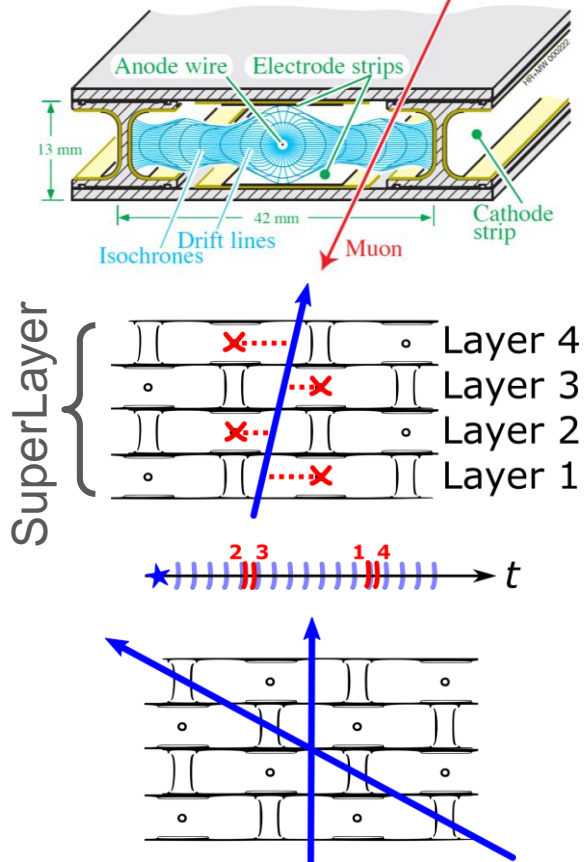
- Replaces current electronics: separate boards for Trigger and Readout, both based on ASICs
- CIEMAT has designed the board that takes care of the theta view of the DT chambers
- Designed around the Polarfire FPGA, mild radiation tolerance
 - Plus some ASICs from CERN for services and communication
 - 14 layers, halogen free material
- 228 time digitization channels (0.78 ns), large output optical bandwidth (~60 Gbps)
- We have tested under radiation at CHARM (CERN) with a high energy hadrons mixed field for 100 Gy



Irradiation at CHARM

The DT Level-1 Algorithm

DT: problem to solve



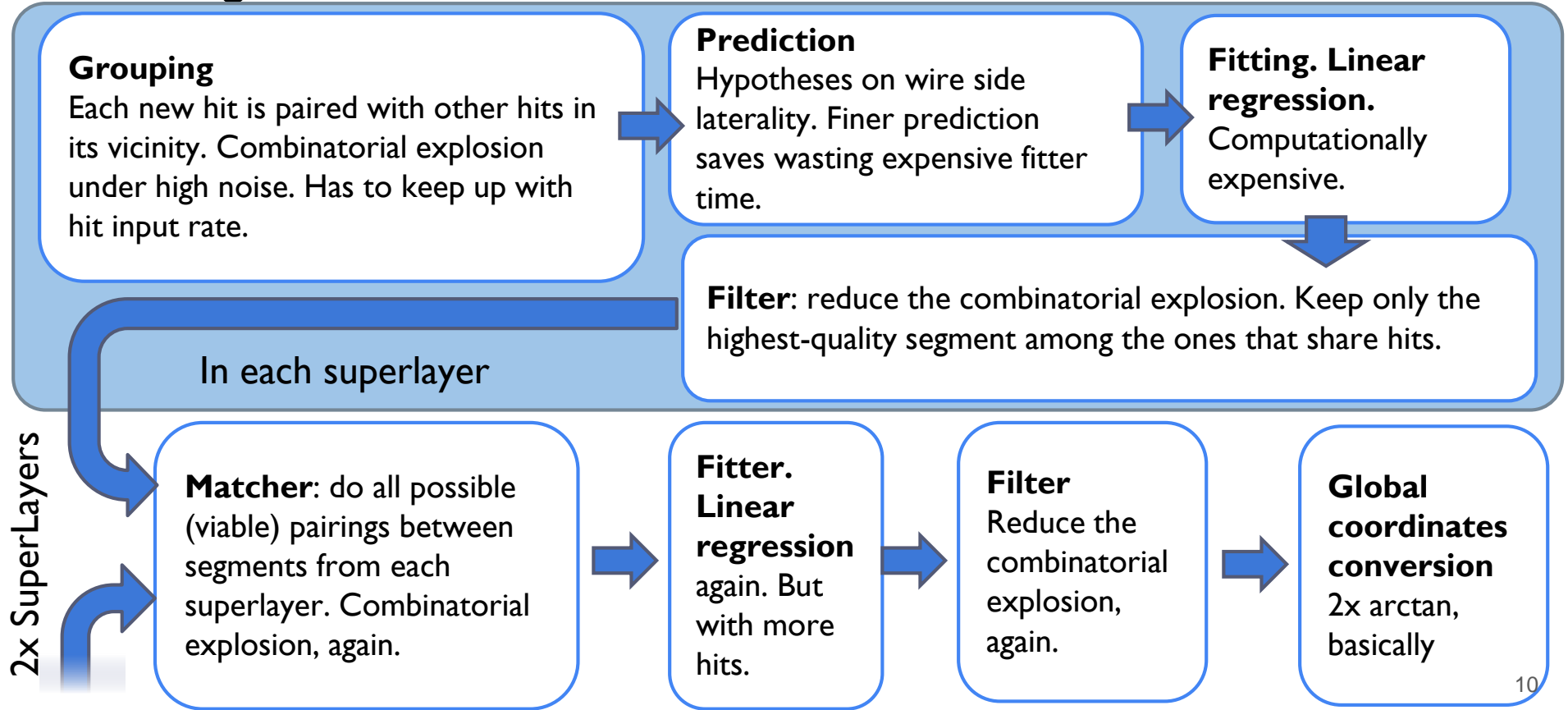
- Drift Time is up to 400 ns = 16 BX
 - LHC Bunch Crossing period = 25 ns
- Muon hits can mix with hits from other muons, even from other events
- Laterality: we don't know a priori if a muon track passed to the right or left of the wire
 - Tracks with same hits but different laterality are called **ghosts**
- Sliding window, continuous processing: we cannot do event-based processing. A hit can form a group with hits 32 BX apart
- When pileup/noise increase \rightarrow combinatorial explosion (number of possible pairings of hits grows very fast)
- This is for one superlayer: then we match each segment with the ones produced in the other 2 superlayers (new combinatorial explosion, new linear regression)

Many people have contributed: J.M. Cella, J. León (CIEMAT TFM & Thesis) S. Goy (CIEMAT enior), A. Navarro (CIEMAT Senior) D. Moran (CIEMAT Senior) C. F. Bedoya (CIEMAT Senior) I. Redondo (CIEMAT Senior) D. Fernández del Val (CIEMAT PhD) J. Llorente (CIEMAT Atracción T.) C. Martín (CIEMAT Atracción T.) J. Troconiz (UAM) F. Frias (UAM) J. Fernández (Oviedo) C. Vico (Oviedo) S. Folgueras (Oviedo) B. González (Oviedo) and more...



<https://doi.org/10.48550/arXiv.2302.01666>

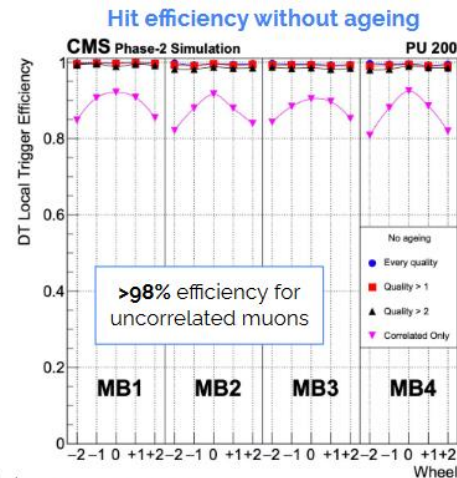
Our algorithm



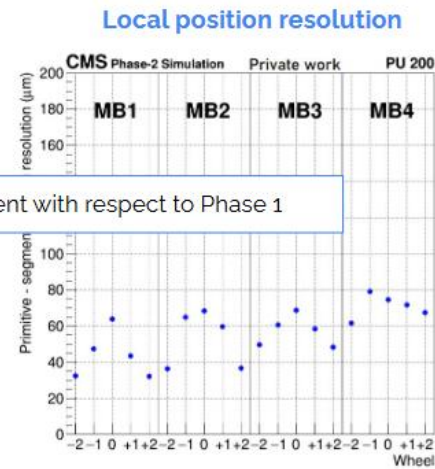
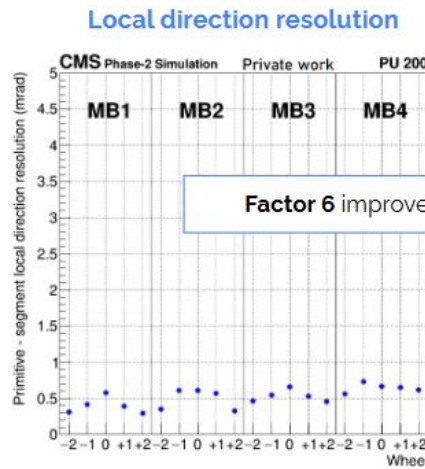
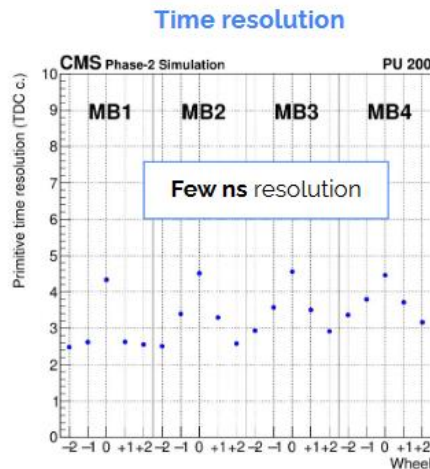
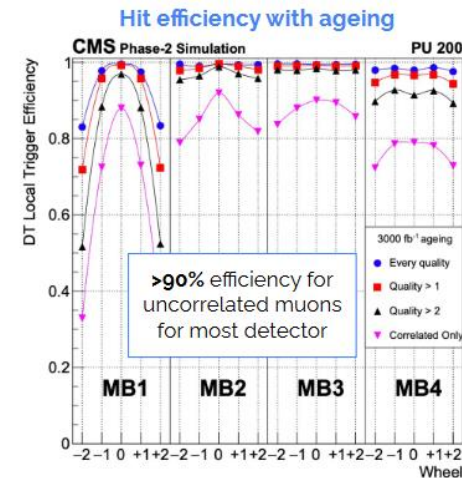
2x SuperLayers

Simulation results (resolution & efficiency)

- Very good performance in simulation
- Even with aged and high pile up scenarios
- Resolutions comparable to offline muon reconstruction
- **Also with real data on demonstrators!!**



Cristina Martín Pérez (CIEMAT) - 2nd COMCHA October 2024

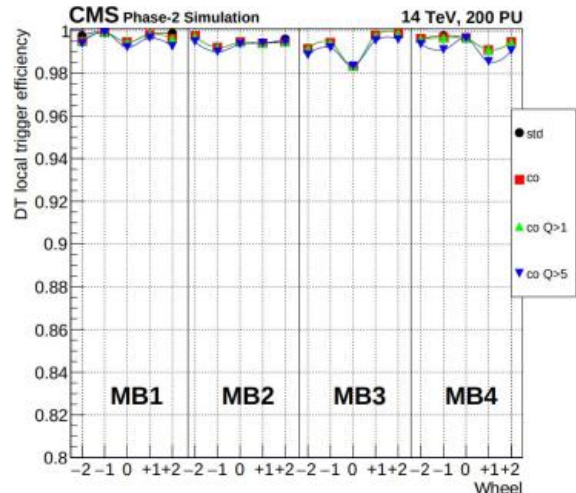
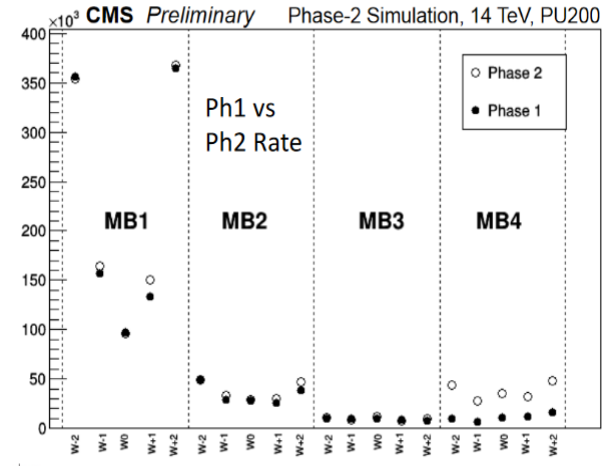
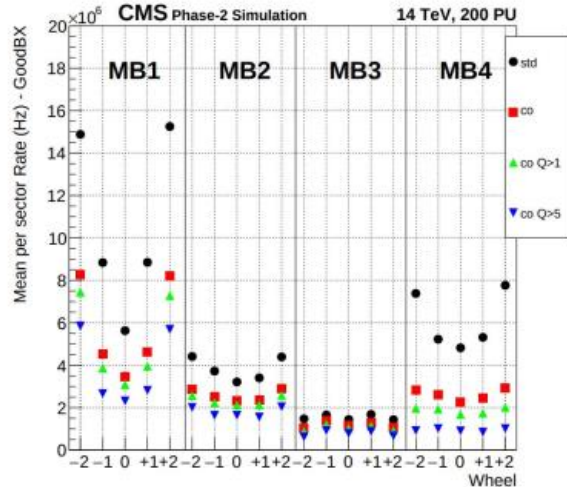
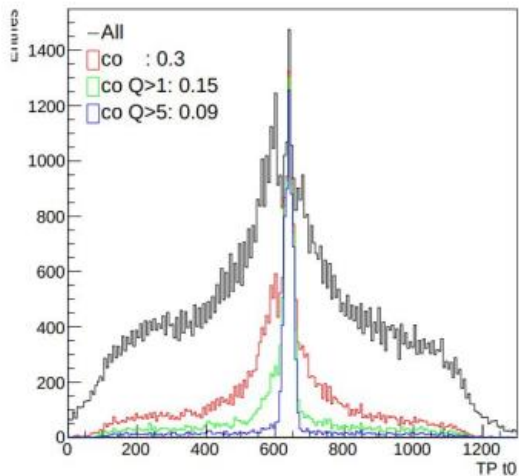


C. Martín Pérez
J. Lorente
J. León
S. Goy

Simulation results (rates)

- Rate reduction strategies being put in place (to select best low-quality hits)
- Coincidences algorithm

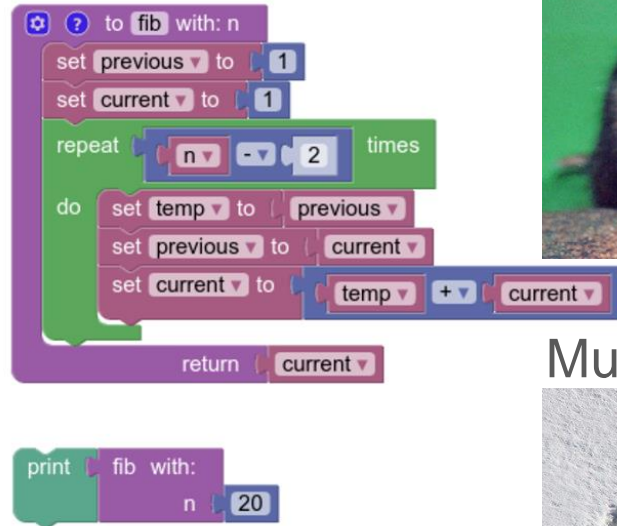
J. Troconiz
D. Moran



FPGA implementation of the DT Trigger Algorithm

What isn't an FPGA?

Linear or sequential program: a succession of instructions that run one after the other in a processor. It might have loops, conditional branching, but there's always a "pointer" that marks the position in the program that is being executed in any particular moment.



```
to fib with: n
  set previous to 1
  set current to 1
  repeat (n - 2) times
    do
      set temp to previous
      set previous to current
      set current to temp + current
  return current
end

print fib with:
  n 20
```

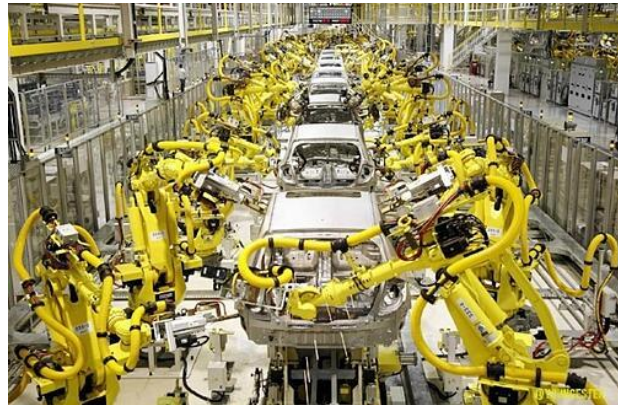
Arduino



Multi-core CPU

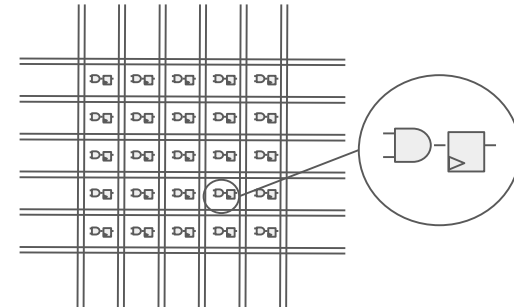


What is an FPGA?



- Array of basic configurable digital components (LUTs and FFs)
- With configurable inter-connection resources
- And with other purpose-specific resources (RAM, DSP, IO blocks, Gigabit Transceivers, Clock management)
- Parallel or sequential: your choice (but for sequential you already have CPUs)

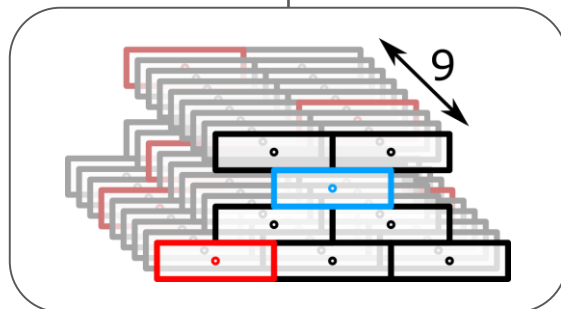
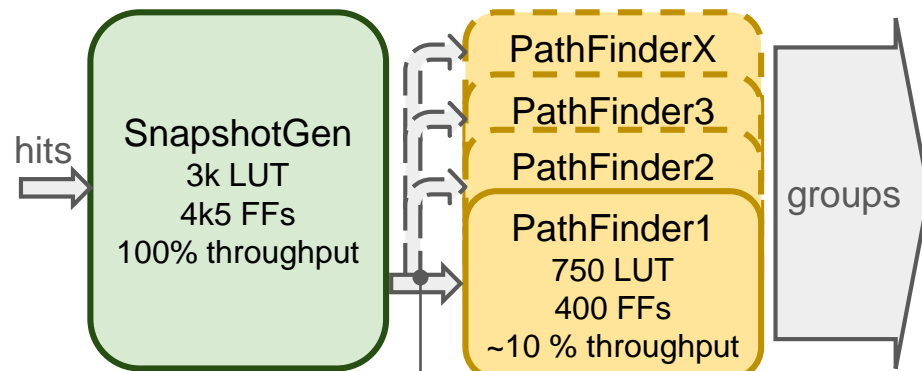
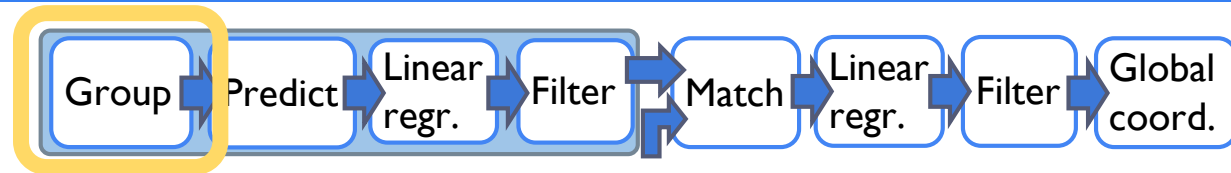
Data goes through different processing stages, but when it goes from one to the next, it doesn't mean that the first pauses and the second starts working; instead, the first one starts working with new data (**pipeline**)



Constraints

- In each FPGA (VU13P)
 - Infrastructure (gigabit links take up much space)
 - 8 chambers (algorithm in previous slide)
 - Other algorithms to come
 - Showers (U. Oviedo)
 - Theta and theta matching
 - RPC matching
 - Maximum latency $\sim 1 \mu\text{s}$ (40 BXs)
 - We're very tight in resources, we're very tight in latency
- ⇒ We must maximize operation frequency to increase computing power. We aim for $\sim 2 \text{ ns}$ clock period ($480 \text{ MHz} = 12\text{x LHC bunch frequency}$)

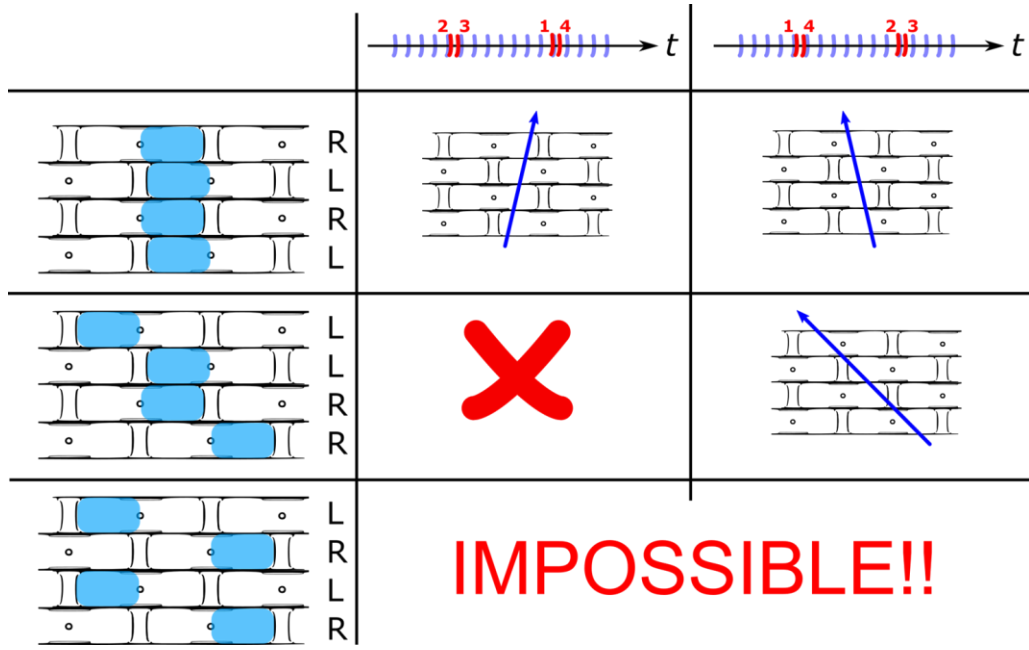
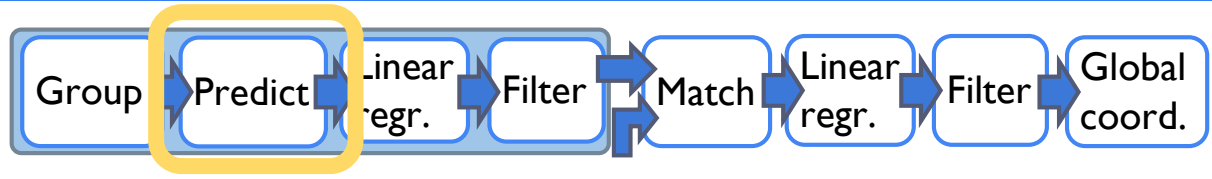
Grouping



- Receives 1 hit/2ns, keeps history, delivers 1 group/2ns, grouped by time and spatial proximity
- Divide and conquer:
 - SnapshotGen: for each hit, delivers a snapshot, a collection of “photos” of the hits received in its vicinity in the past 16 BXs. Keeps up the pace with the input
 - PathFinder: for each snapshot, delivers all possible combinations of past hits with the new hit. Can take many clks to process one snapshot
- PathFinder doesn't keep up with the input rate, several can be instantiated in parallel
- Processing stops when newly-generated groups would be out of maximum latency

Prediction

J. León

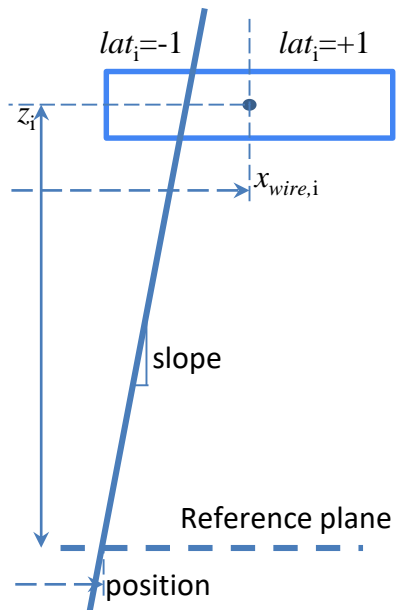


IMPOSSIBLE!!

- Hit groups do not have laterality assignments → new explosion of workload
- Linear regression is expensive. We don't want to waste it with unviable candidates
- Pre-calculated Look-Up Table gives likely acceptable laterality combinations for a group of hits
- Input to the LUT is cell layout (geometry of cells involved) plus *coarsified* time (9 → 2 bits) of each hit
 - *Coarsification* is not just truncation or rounding: optimal thresholds result of bayesian optimizer
- Takes 600 LUT + 500 FFs
- Reduces average number of laterality combinations per group from 2.78 to 2.16

Linear regression

$$y \equiv \begin{bmatrix} x_{wire,1} + lat_1 \cdot t_1 \\ x_{wire,2} + lat_2 \cdot t_2 \\ \dots \\ x_{wire,8} + lat_8 \cdot t_8 \end{bmatrix}, X \equiv \begin{bmatrix} lat_1 & 1 & z_1 \\ lat_2 & 1 & z_2 \\ \dots & \dots & \dots \\ lat_8 & 1 & z_8 \end{bmatrix}, b \equiv \begin{bmatrix} t_0 \\ pos \\ slope \end{bmatrix}$$



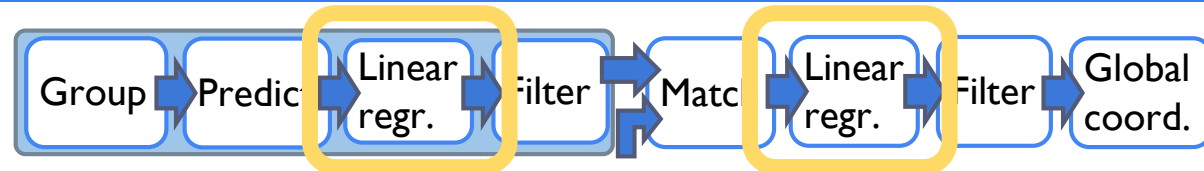
$$y = X \cdot b + \epsilon$$

$$b = (X^T \cdot X)^{-1} \cdot X^T \cdot y$$

$$C \equiv (X^T \cdot X)^{-1} \cdot X^T$$

$$b = C \cdot y$$

ROM (blue arrow), DSP (red arrow), LUT (green arrow)

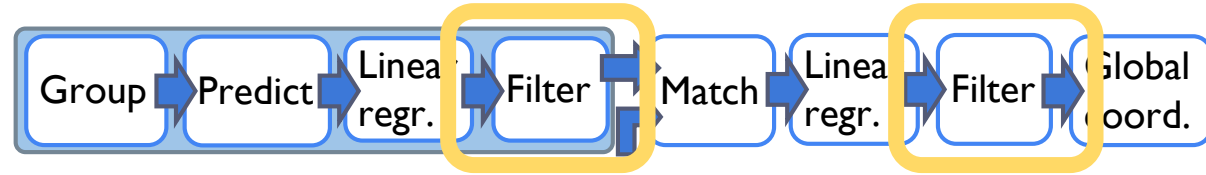
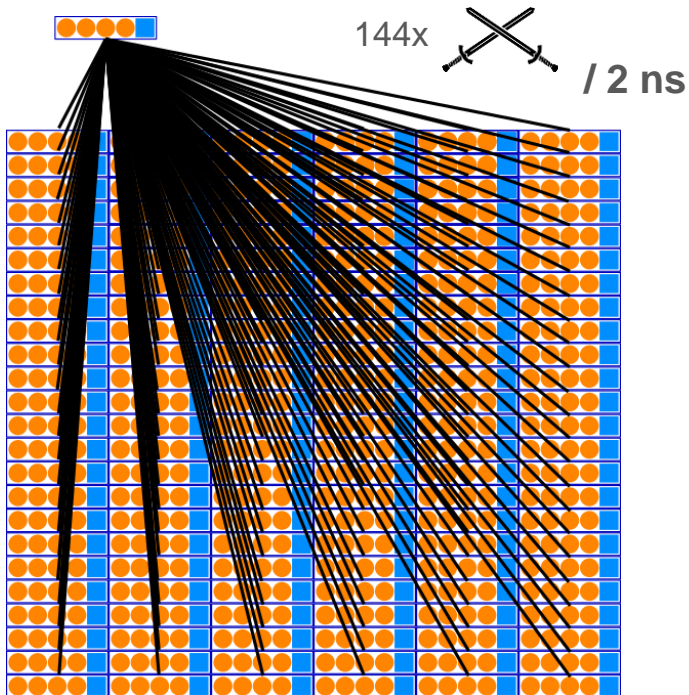


- Inputs: horizontal position of the wires, TDC value, laterality hypothesis
- Outputs: track parameters (T0, position, slope) and chi squared
- Ordinary Least Squares, matrix solution:
 - C matrix super expensive computationally
 - y matrix: much simpler, unavoidable (hit timestamps)
- We compute C matrix offline, store it in ROM
 - For 1 SL, ~50 rows, 200 bits each → distributed RAM
 - For 2 SL, ~1500 rows, 360 bits each → 20 Block RAM (UltraRAM doesn't allow ROM initialization)
- Computational load reduced to bare minimum
- Latency: 15 clock cycles
- 8 layers: 3k LUT, 4.5k FFs, 20 BRAM, 25 DSP

Filtering



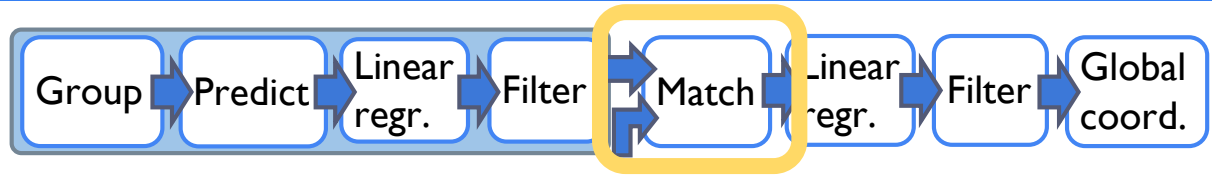
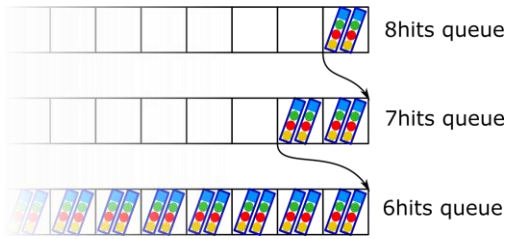
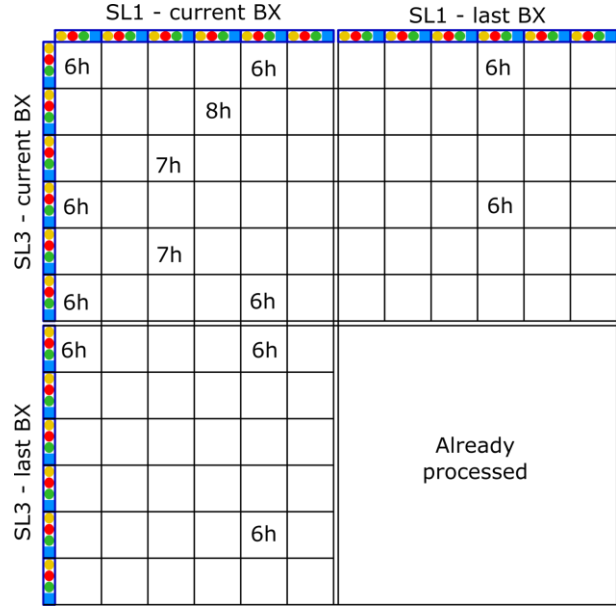
1 segment
= 35 bits (*)



- Hits must be used once, in the best-quality segment
- Arriving segments are compared with 144 previous segments (24-BX deep history x 6 segments width):
 - May duel any (only if they share any hit)
 - May be killed by any
 - If it survives, it may kill any
- All must happen before the next segment arrives (2 ns!)
- Pipeline the input stage: before being written, each incoming segment spends 4 cycles calculating its “duel” result with all 144 pre-stored segments
- **But...** when a hit in the pipeline reaches the table, the table may have been changed already... Incoming hits “shoot bullets” at each other, these “bullets” also traverse the pipeline and reach its target in the required moment
- 6k LUT, 7k FFs, 4 BRAM

(*) Actually close to 300 bits, but unused are stored to RAM

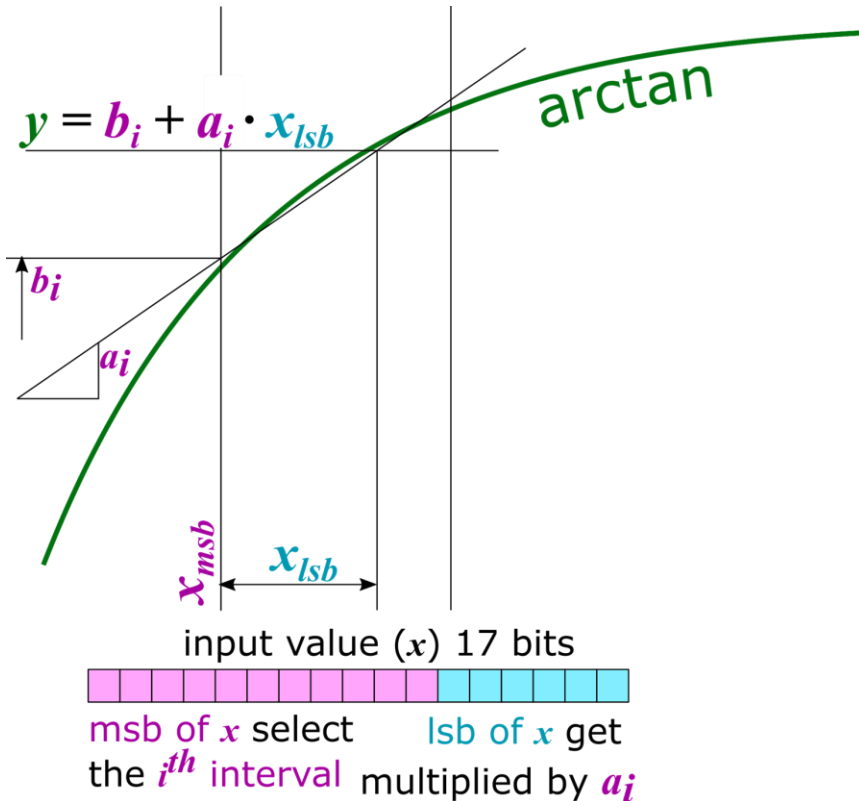
Matching



1 SuperLayer segment = 20 bits
 (Actually close to 300 bits, but unused bits are stored to RAM)

- For ϕ view, pairs are made with segments from SL 1 and 3, from present and last BX
- 108 possible combinations: assess compatibility in t0, position, slope
- Only the best (linear regr. is expensive!) are selected for re-fitting
- Perfectly sorting data within each quality would be too expensive (and seldom make a difference)
- We classify the candidates in N queues according to predefined criteria (categories)
- Candidates are delivered starting from the highest-priority queue
- Currently 3 queues based only on number of hits of the combined segment
- Resources: 1k8 LUT, 1k2 FF (pairings), ~500 LUT/FF (each queue)

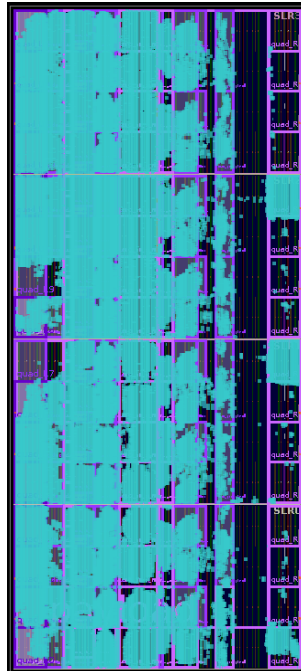
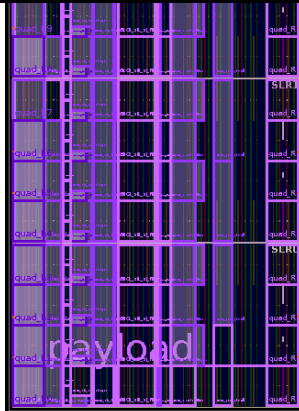
Global coordinates



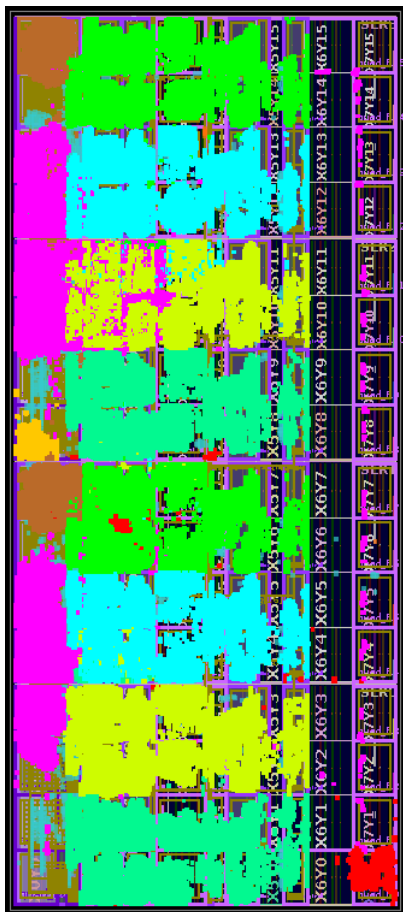
- Essentially 2 arctangent operations: one for position, one for slope
- High throughput, low latency, low resources...
- Piecewise linear approximation
- a and b coefficients stored in RAM (loaded at configuration, different for each chamber)
- ≤ 1 LSB approximation error
- Latency 4 clock cycles
- 500 LUT, 500 FF, 3 BRAM, 3 DSP

Timing closure

```
#####
pathfinder
SSSDSSRSSDSSSSS  USSDSSSS  SSSSSSSDSSRSSDS
Slice shape: 6x30, boundaries=(48,55,0,1)
Resource utilization:
  dsp      0 /      12 (0.0%)
  ff       700 /    2880 (24.3%)
  lut      950 /    1440 (66.0%)
  bram     0 /      0 (0.0%)
  ultraram 4 /      8 (50.0%)
#####
mortadelos_input_sorter
SDSSSSSUSSDSSSS  SSSSSSSDSSRSSDSSR  SSSSSSSSSSSSSSD
Slice shape: 13x15, boundaries=(56,72,0,0)
Resource utilization:
  dsp      0 /      12 (0.0%)
  ff      1000 /    3120 (32.1%)
  lut      700 /    1560 (44.9%)
  bram     8 /      6 (133.3%)
  ultraram 0 /      0 (0.0%)
#####
```



- Big design, high clk freq → challenge
- Initial naive approach, out-of-context each module with big margin (1.7 ns), failed:
 - OOC hides interconnection issues
 - Vivado placer does poor job on big designs (more random choices more likely to make bad ones)
- Regular placement constraining is cumbersome with our design → developed python library and scripts to auto-generate pblocks
 - More user-friendly, way easier to maintain, with the design still in development
- Helped identifying the netlist problems between modules (high-fanout, insufficient piping...) and gave vivado the boost it needed to get me those last 150 ps
- Presented at 1st FPGA Developers Forum (June'24)
 - <https://indico.cern.ch/event/1381060/contributions/5923235>
- Highlighted at [TWEPP 2024 Optics and FPGA users forum](#)
- Available at gitlab.cern.ch
 - https://gitlab.cern.ch/anavarro/ant_placer



FPGA Resource utilization

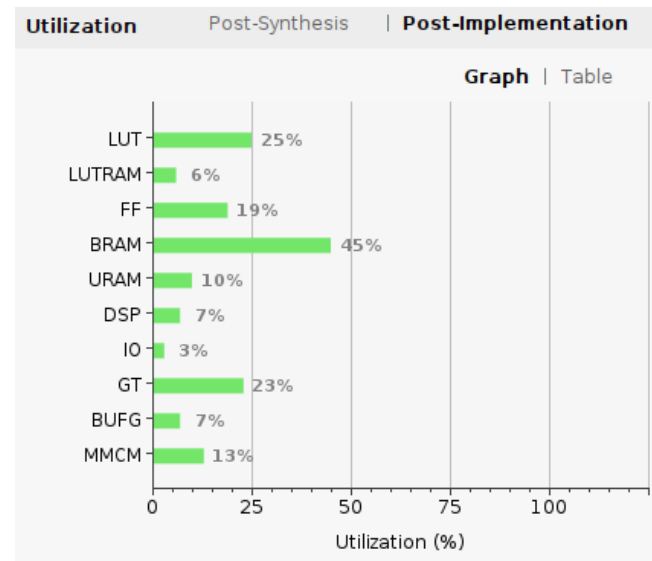
- Image of the firmware implemented in the VCU13P with 8 chambers
- Fits in latency: 20 BX (of which, 12 BX processing, 8 BX wait times)
 - Phi TP generation+infrastructure and first BF FW prototypes
 - Missing theta Superlayers & RPCs.
- ~35-40% of effective occupation

Infrastructure:

- datapath (links)
 - inputs from OBDT
 - outputs to Barrel Filter
- ttc
- ipbus

Payload

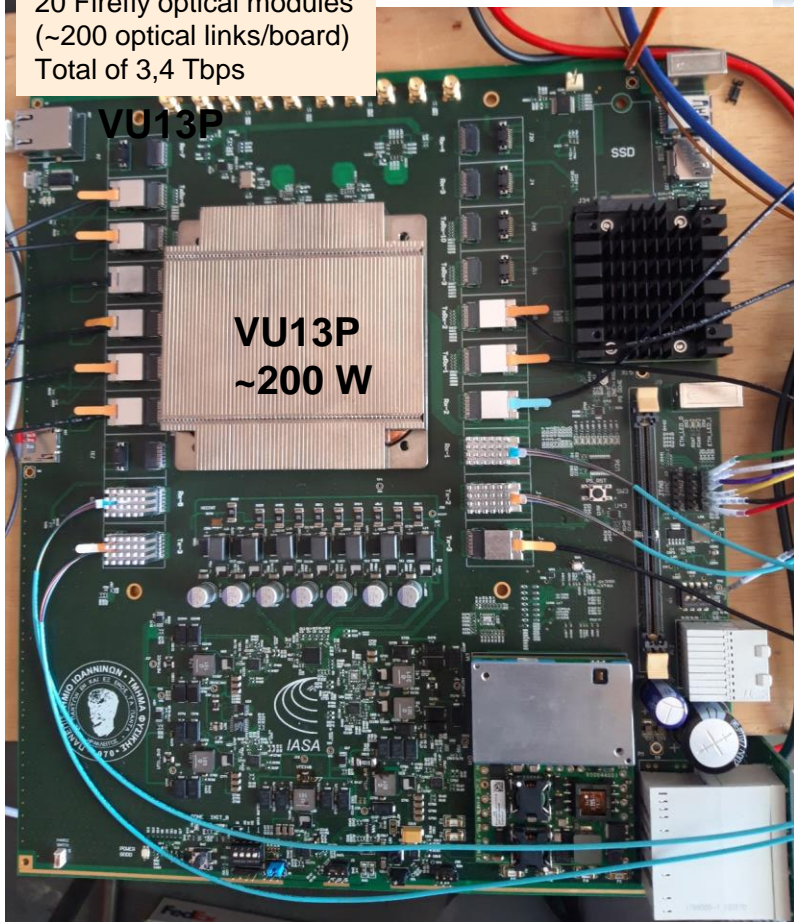
- S0MB1 and S1MB1
- S0MB2 and S1MB2
- S0MB3 and S1MB3
- S0MB4 and S1MB4
- Barrel Filter



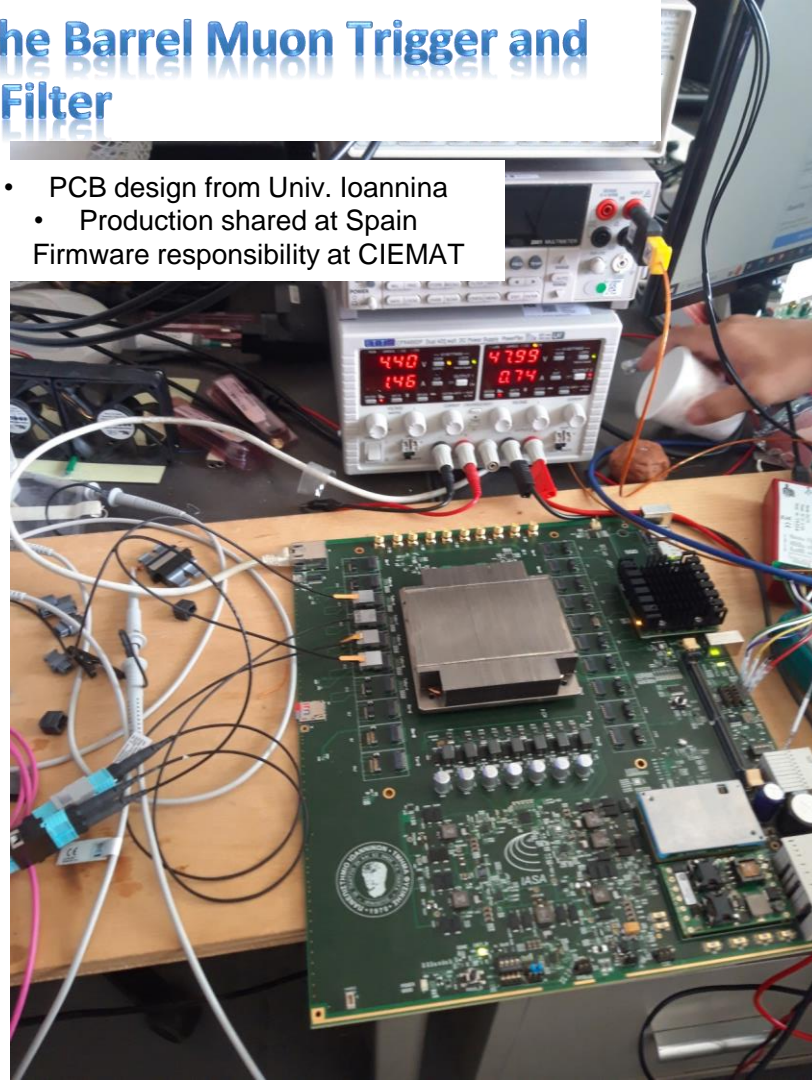
DT Trigger Algorithm in actual hardware

BMTL1: ATCA processor for the Barrel Muon Trigger and Barrel Filter

20 Firefly optical modules
(~200 optical links/board)
Total of 3,4 Tbps

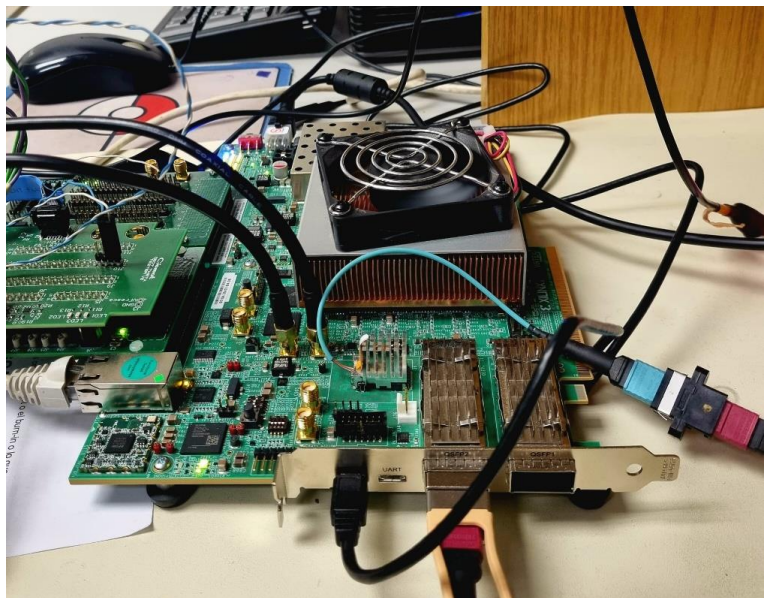


- PCB design from Univ. Ioannina
- Production shared at Spain
- Firmware responsibility at CIEMAT



Firmware implementation in HW demonstrators

- AM algorithm developed in VHDL firmware for various platforms: Virtex 7, Virtex Ultrascale +



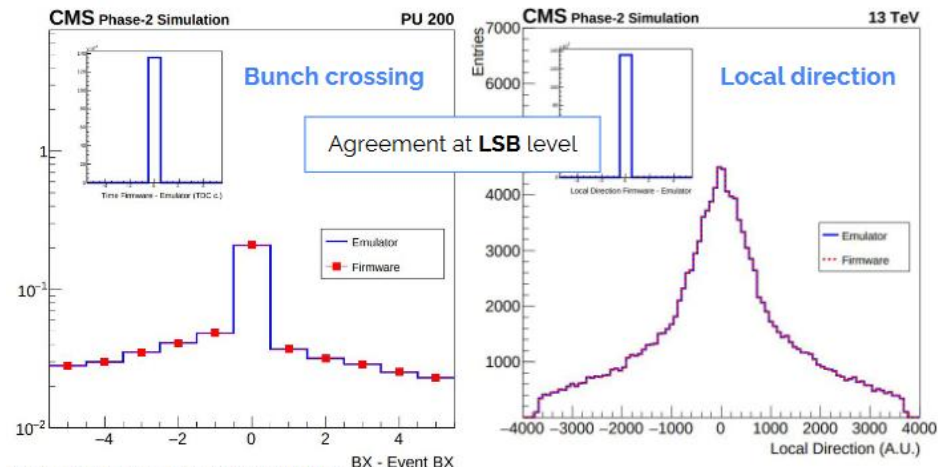
VCU118 evaluation board VU9P proxy of the VU13P of the BMTL1

- Firmware includes functionalities for control and operation
- Extensive firmware-emulator comparisons show excellent agreement

Matching efficiency

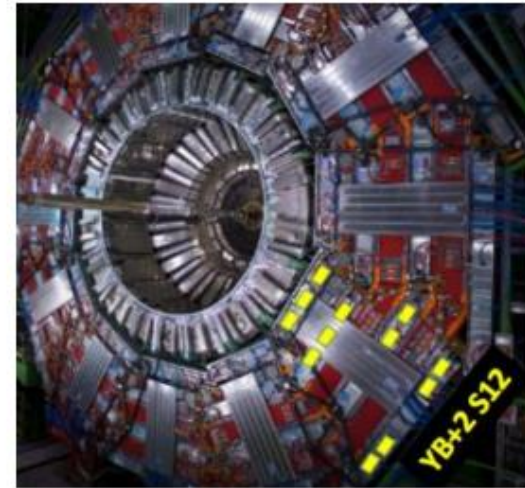
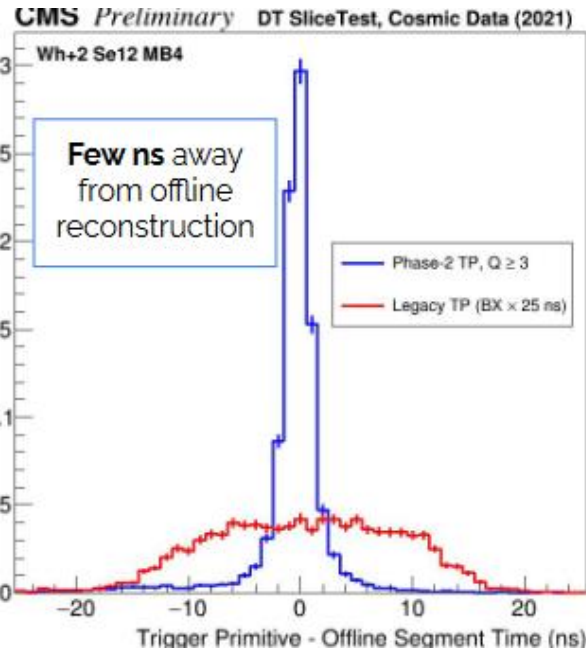
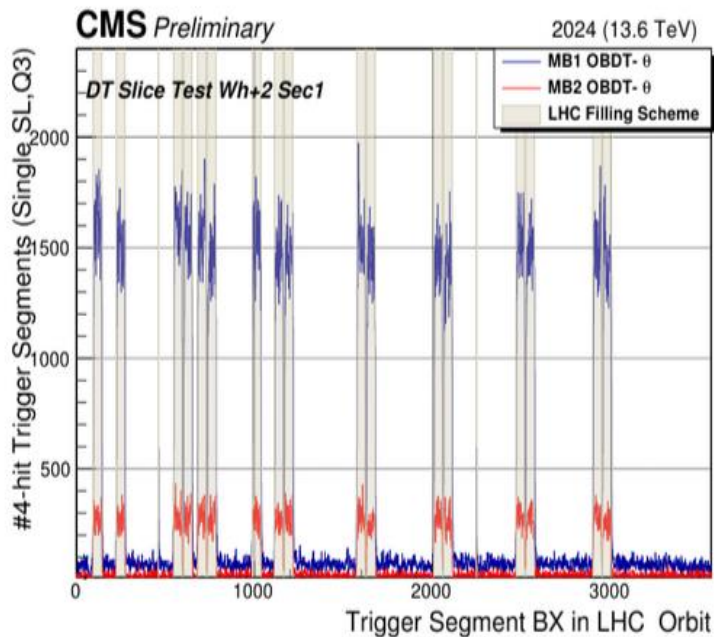
Quality	Matching w.r.t. emulator (%)
1	98.17 ± 0.05
3	99.70 ± 0.03
6	94.22 ± 1.19
7	97.63 ± 0.21
8	99.99 ± 0.01
Average	98.76 ± 0.03

Quality	Matching w.r.t. firmware (%)
1	99.46 ± 0.03
3	99.93 ± 0.02
6	93.52 ± 1.24
7	97.61 ± 0.21
8	99.99 ± 0.01
Average	99.56 ± 0.02



CMS DT Slice Test

- Since LS2 two CMS Sectors (Wh+2 S12 & S1) have been instrumented with Phase-2 front-end and back-end prototypes, in parallel to the current Phase-1 system.
- Campaigns of data taking with cosmics and collision data demonstrate offline-like performance.



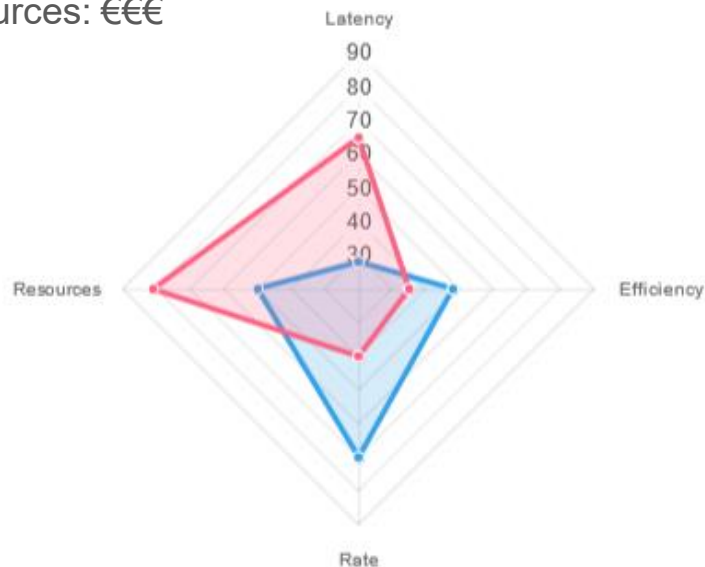
IEEE (NSS), Tweep, iWorld, etc

Close-up

Why FPGA? HEP Trigger Algorithms

Trigger is a trade-off between

- Latency: small and deterministic
- Efficiency: do not miss an interesting event
- Rate: do not record many uninteresting events
- Resources: €€€



Traditionally there was no choice: hardware (ASIC, FPGA)

- Very fast
- Simple algorithms, limited performance
- Example DT: ASICs (on-detector), FPGAs

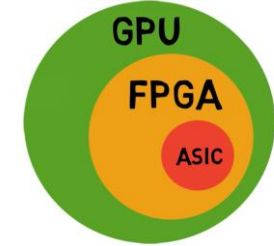
Present – future

- more workload: High-Luminosity
- more latency budget
- more computing power available, bigger, faster FPGAs, but also GPUs
- More advanced algorithms

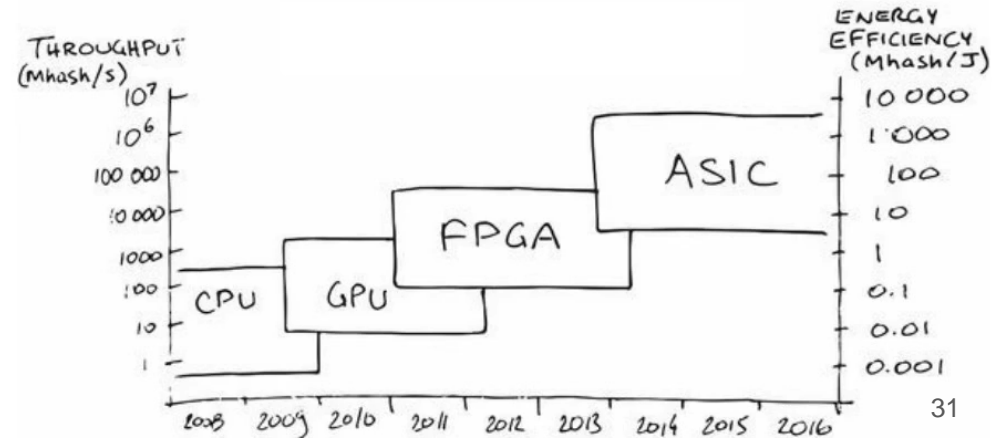
Why FPGA? Computing (crypto mining example)

- Interested in profits. Costs:
 - Developer: one-time, doesn't scale with load
 - Silicon: one-time, scales with load
 - Energy: recurrent, scales with load
- Highest energy efficiency comes from a silicon device that is perfectly tuned and suited for your problem → ASIC
 - But #1. When an ASIC is perfectly suited for a task, it might not be very useful for a just slightly different task. Also bugs, **initial investment**.
 - But #2. FPGAs, GPUs and CPUs are also ASICs. They're just tuned to general tasks.
- So each of your problems may be more efficient (profitable?) in a different platform

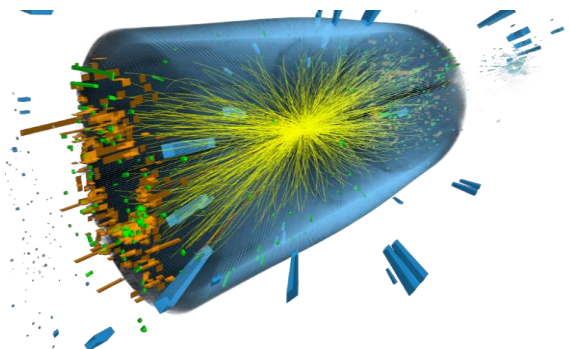
ALGORITHM COMPATIBILITY



PERFORMANCE VS TIME



Future of FPGAs in HEP



- Our core activity as of today is focused on the CMS Muon Trigger
- Challenging project which required very high expertise in FPGA design
- We are working in collaboration with international developers for sharing code and developments . We are collaborating in DRD 7.5
- Useful tools are being disseminated through Workshops and Forums. Also schools.
- Use of AI on FPGAs is happening also in the background
- Developments ongoing will be the base of architectures for future colliders:
 - ultra fast reconstruction with offline performance
 - Use of FPGAs will likely mark the tendency for computing acceleration
 - Intelligence on the detector will also be prosecuted with expected low radiation at FCC e^+e^-

Work in on going on **cutting-edge programmable logic for implementing intelligent algorithms** directly at the detector level offering real-time data analysis capability.

Thanks!

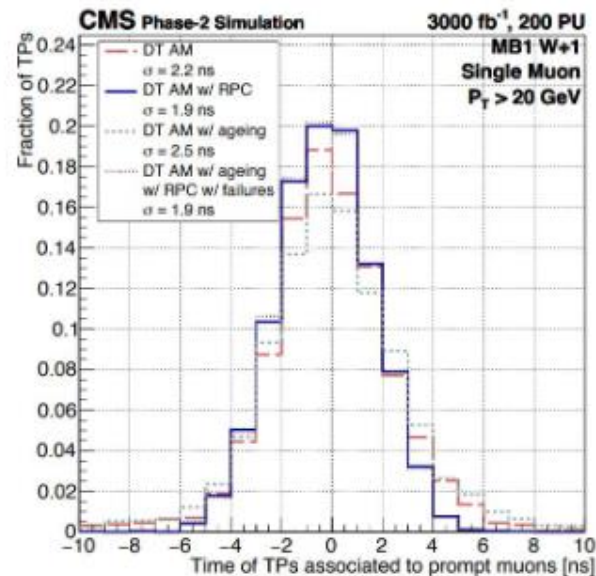
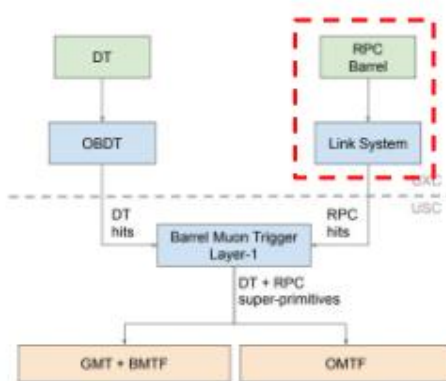
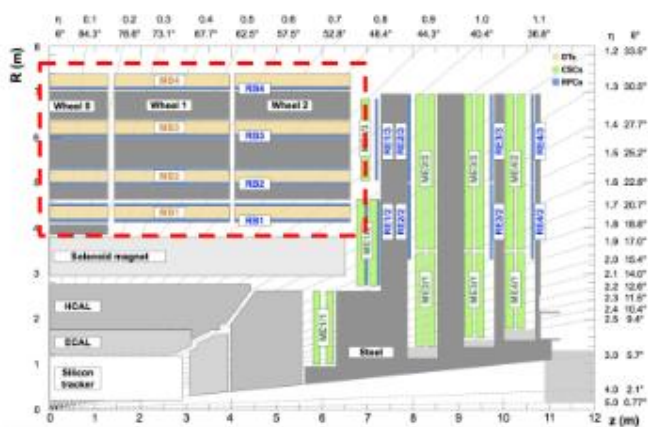
Future Plans in CMS

Several extensions of the AM algorithm are being developed:

- Phi and theta segments matching algorithm being put in place (Oviedo)
- Coincidence algorithm to reduce the rate of “ghosts” (Dermott Moran)
- Identification of showers (Oviedo)

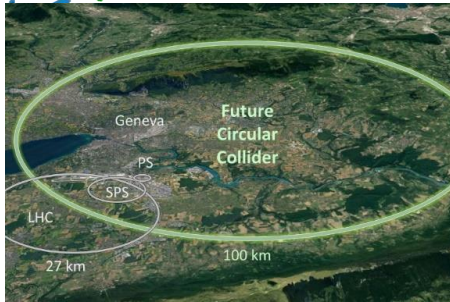
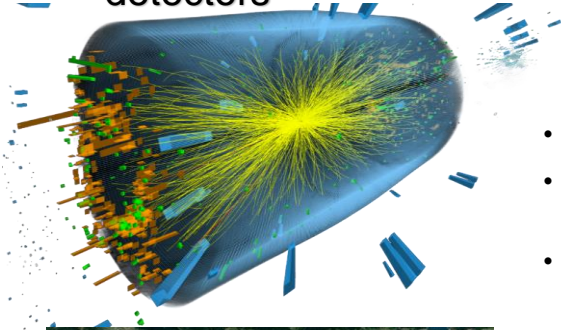
Inclusion of DT Trigger Primitives for Luminosity measurement with bunch crossing granularity

- Ongoing work to extend the Analytical Method to include the RPCs in the barrel
- Combined “Superprimitives” exploiting DT space resolution and RPC time resolution
- Exploring the use of neural networks profiting from the increased flexibility and computational power of the FPGAs



Intelligence on detector

Particle collider detectors



- Technology trend is to bring larger amount of **intelligence near the detector**, allowing intelligent techniques:
 - Discriminating signal/noise, signal shape, geometry, PID,
 - Time measurement, switching matrixes, handling of large throughput
- **Programmable logic** allows implementation of diverse algorithms and also, modification
- In general FPGA performance is worse than ASIC's regarding radiation and power but it has **big advantages in terms of developing time and reusability**
- In LHC detectors, trade off typically restricted FPGAs to the outside areas of the detector, typically Muons.
- In detectors for e+e- colliders the **boundary of usability due to radiation moves significantly towards the IP** (no hadrons, only TID).
 - For instance, Alice inner Tracker CMOS MAPs, a technology considered for FCC-ee, spec is [700krad, 10krad]
- This may open the door to using **commercial FPGAs in larger areas of the detectors**
- Microchip has in the market the **PolarFire technology of Flash based FPGAs** adequate for mild levels (<3000Gy). **We have designed a board around it.**
- **Some of the planned actions include developments of AI in this platform**

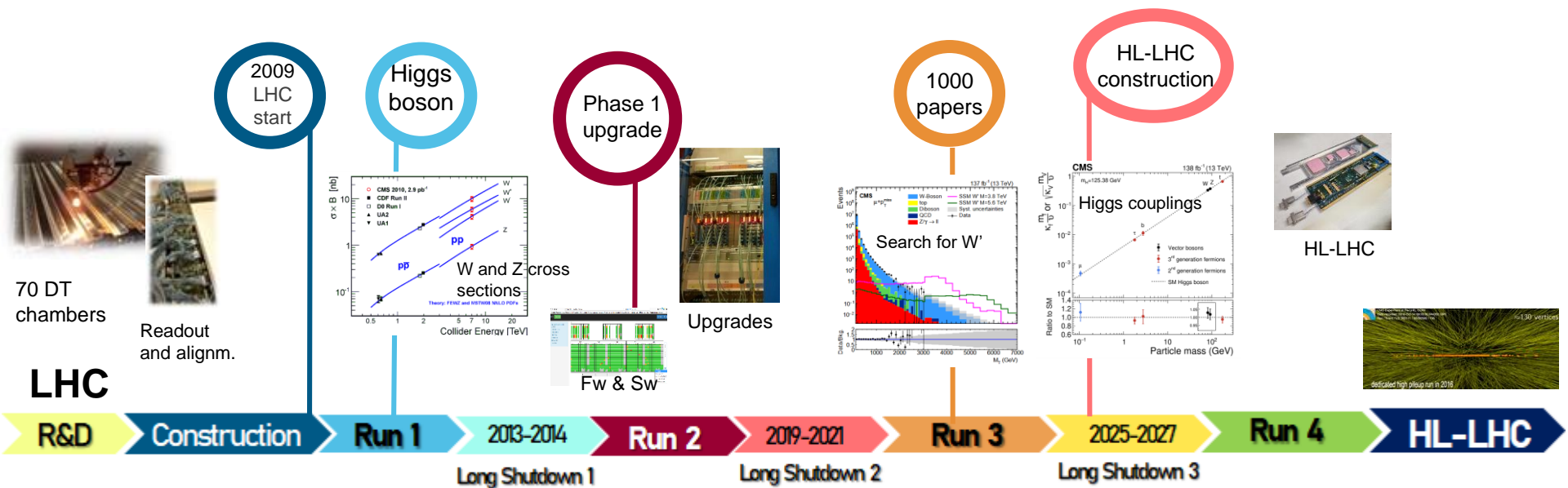


CIEMAT at Colliders



- Continued participation of CIEMAT at highest energy colliders since decades.
- Critical LHC contributions since early days.
- Involvement in future colliders simultaneous to LHC and HL-LHC

CMS
(Compact Muon Solenoid)



FPGA Algorithms take-away message

- Traditional division
 - hardware (ASIC, FPGA) → fast, simple algorithms
 - software (CPU, GPU) → slow, complex algorithm
- Not anymore
 - Some software solutions can run in trigger latencies
 - Hardware computing becoming competitive
- FPGAs have their niche
 - Parallelism, medium data widths, high throughput, low latency
 - Requires overcoming the learning curve

Other activities: DRD 7.5

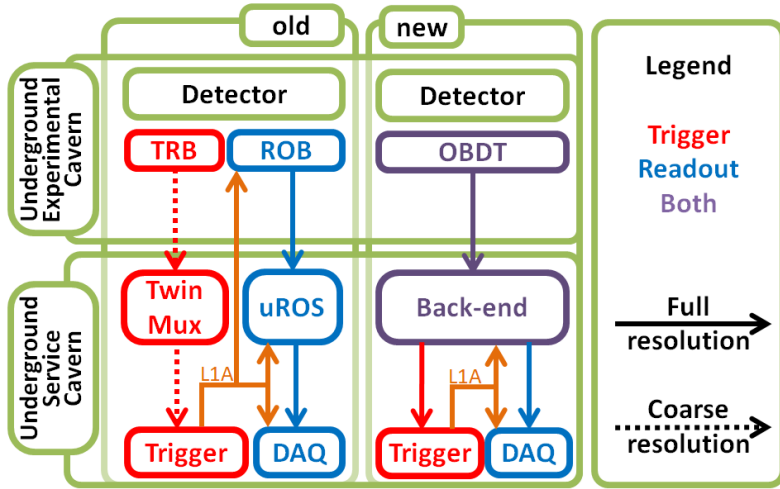
- The evolving COTs technologies and its performance needs to be considered for real-time backend processing
- We are working now in the development of TDAQ tools.
- Throughout the years, we have developed plenty of highly-optimized (**latency, resources, throughput**) modules for FPGA which could be of interest for the community

Some examples:

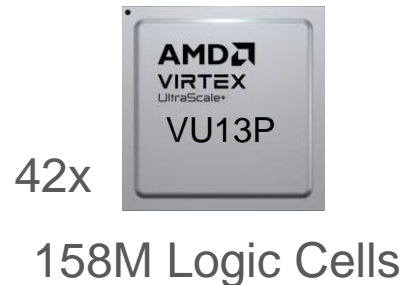
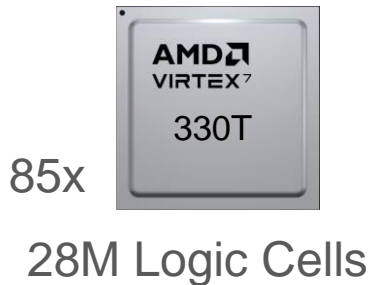
Made available to DRD7.5 collaborators

- **Implementation of a piecewise approximation to an arbitrary function**
 - Automatic input range partitioning and sizing of fixed-point coefficient widths for a target approximation error.
 - HDL implementation uses of DSP and RAM, highly pipelined, high clock frequency.
 - Example: coordinate conversions using complex trigonometric relationships
- **Multivariate linear regression of a set of data points with quantized independent variable(s)**
 - Automatic sizing of fixed-point coefficient widths for a target resolution.
 - HDL implementation uses DSP & RAM, highly pipelined, high clock frequency. Outputs parameters, residuals, sum of squared residuals
 - Example: DT segment fit (z fixed for each layer), detections at fixed time intervals, fixed distances...
- **Sparse input serializer: serialize data from parallel source with sparse occupancy**
 - Automatic selection of optimum architecture depending on parameters (clock frequency, input/output widths...), option for radiation environment (TMR, RAM ECC)
 - Examples: on-detector readout of high-granularity, low-occupancy detector; also useful as submodule in algorithmic block for queuing “hits” from a combinatory stage.
- **Several other smaller modules, some LHC-specific or CMS-specific**

Introduction - Upgrade Phase 2

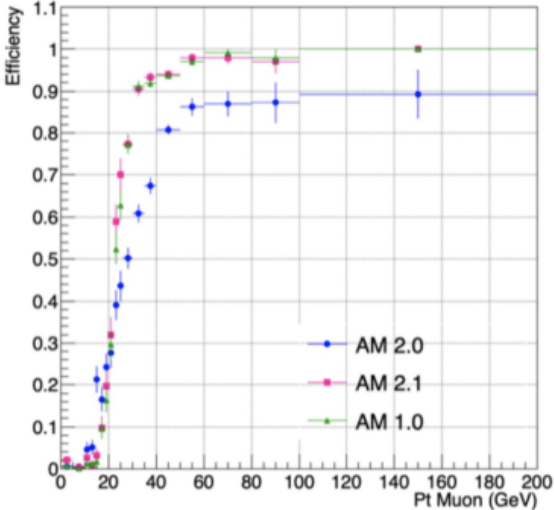
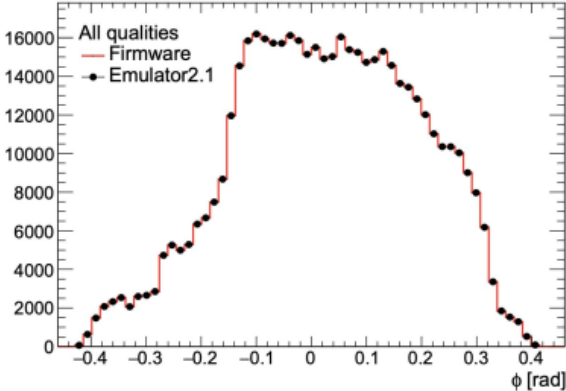
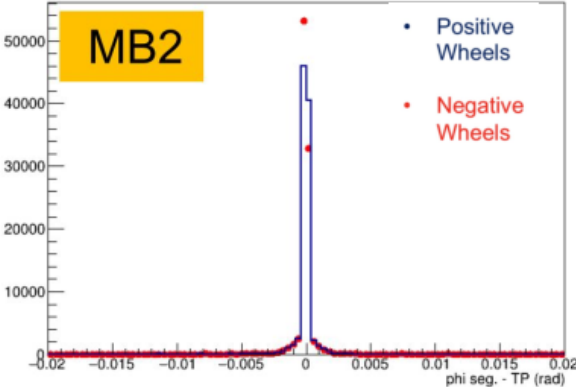


- Trigger/readout architecture changes
- Full data streaming to Underground Service Cavern (USC) of all DT data (no filtering)
- Trigger can done at USC (no radiation): bigger and faster FPGAs (before on-detector ASIC + USC V7)
- Achieve offline-grade (SW) performance at Level-1 Trigger (HW)



[2] Global coordinates (v2.1)

- The **fix in the assignment of global sector coordinates** solved the discrepancies with offline segments and between firmware and emulator, and recovered the L1 trigger efficiency:



Foot note 1: on GPUs

The leap in parallelism between a CPU and a GPU is unfairly represented by the animal metaphor (metaphors are powerful but limited).

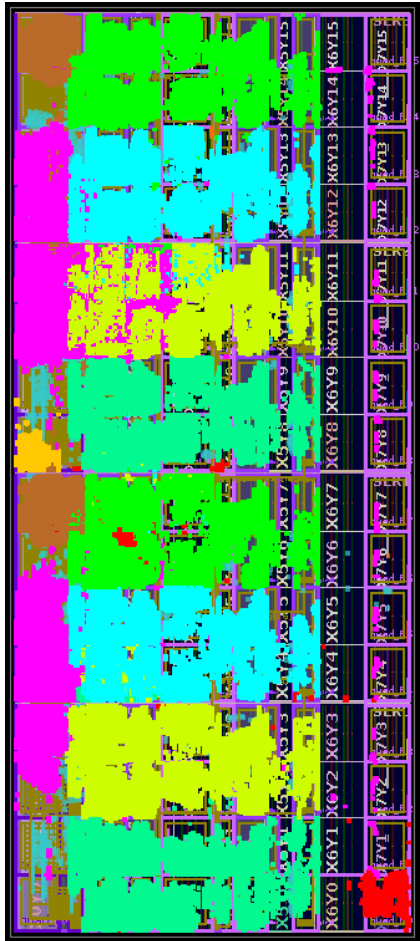
Switching from processor to GPU requires a big mindset change in the same direction as going to FPGA: unfolding sequential loops in multiple processors running in parallel

All have use cases in which they excel

AMD: ryzen, radeon, xilinx

Stay tuned for the GPU course in COMCHA next days!





- Image of the firmware implemented in the VCU13P with 8 chambers
- Only phi Trigger primitive generation+infrastructure, missing theta Superlayers & RPCs
- Occupancy ~35%

Infrastructure:

- datapath (links)
 - inputs from OBDT
 - outputs to Barrel

Filter

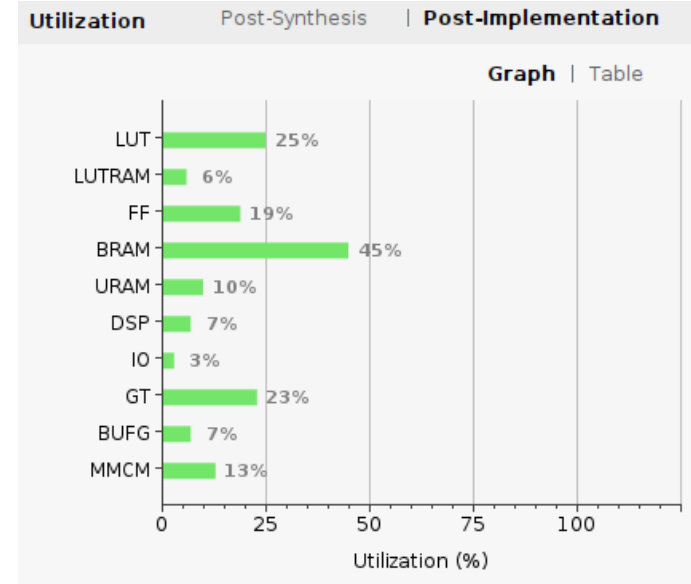
- ttc
- ipbus
- output data formatting and readout

Payload

- SOMB1 and S1MB1
- SOMB2 and S1MB2
- SOMB3 and S1MB3
- SOMB4 and S1MB4

Firmware of the trigger algorithm:

- The Analytical Method has been developed to implement analytical solutions for reconstructing the DT trigger primitives for Phase 2.
- This algorithm exploits the maximum resolution achievable by the DT chambers, bringing the hardware system closer to the offline performance capabilities.

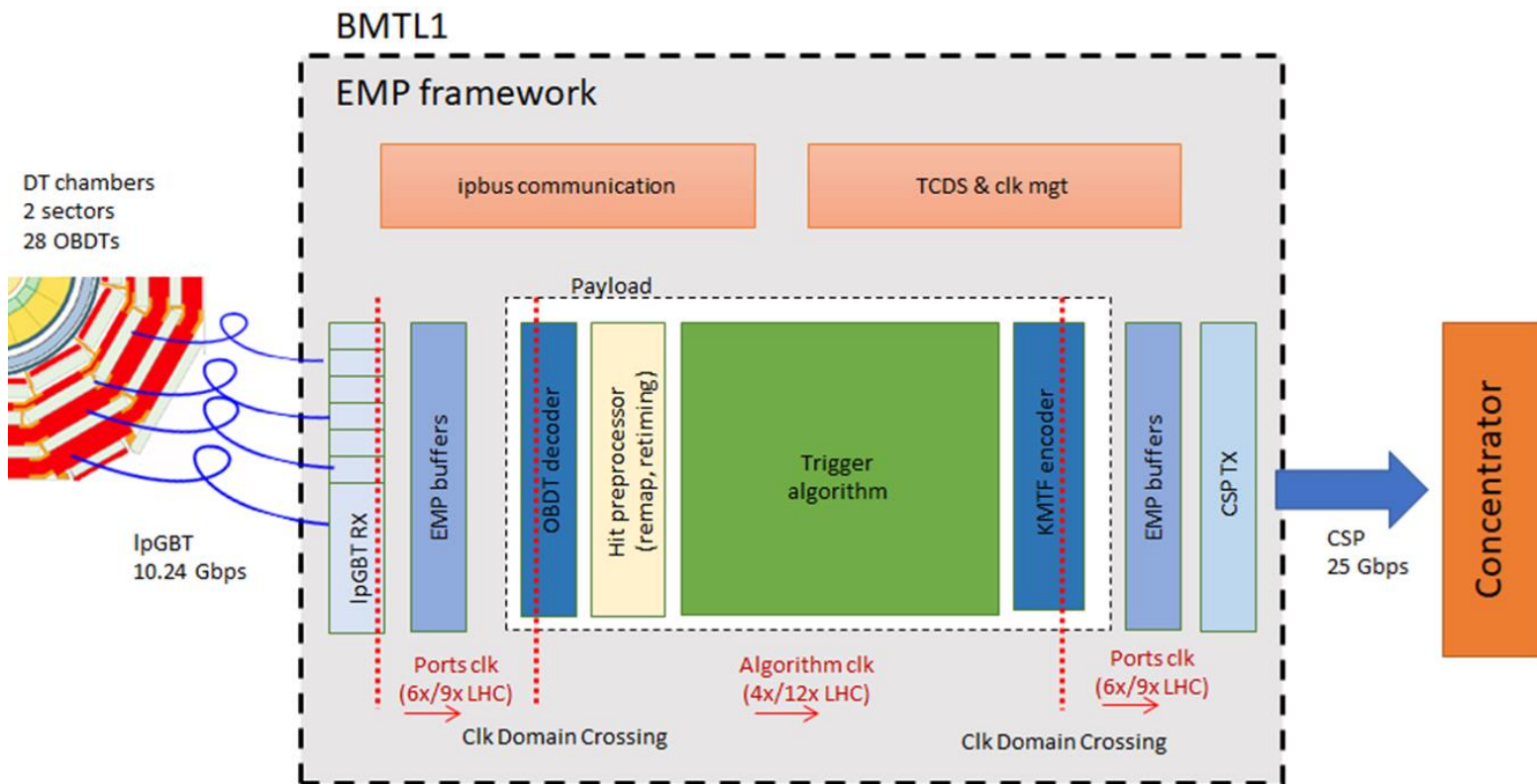


G. Abbiendi et al. 2023 “The Analytical Method algorithm for trigger primitives generation at the LHC Drift Tubes detector”. Nucl.Instrum.Meth.A 1049 168103. DOI: [10.1016/j.nima.2023.168103](https://doi.org/10.1016/j.nima.2023.168103)

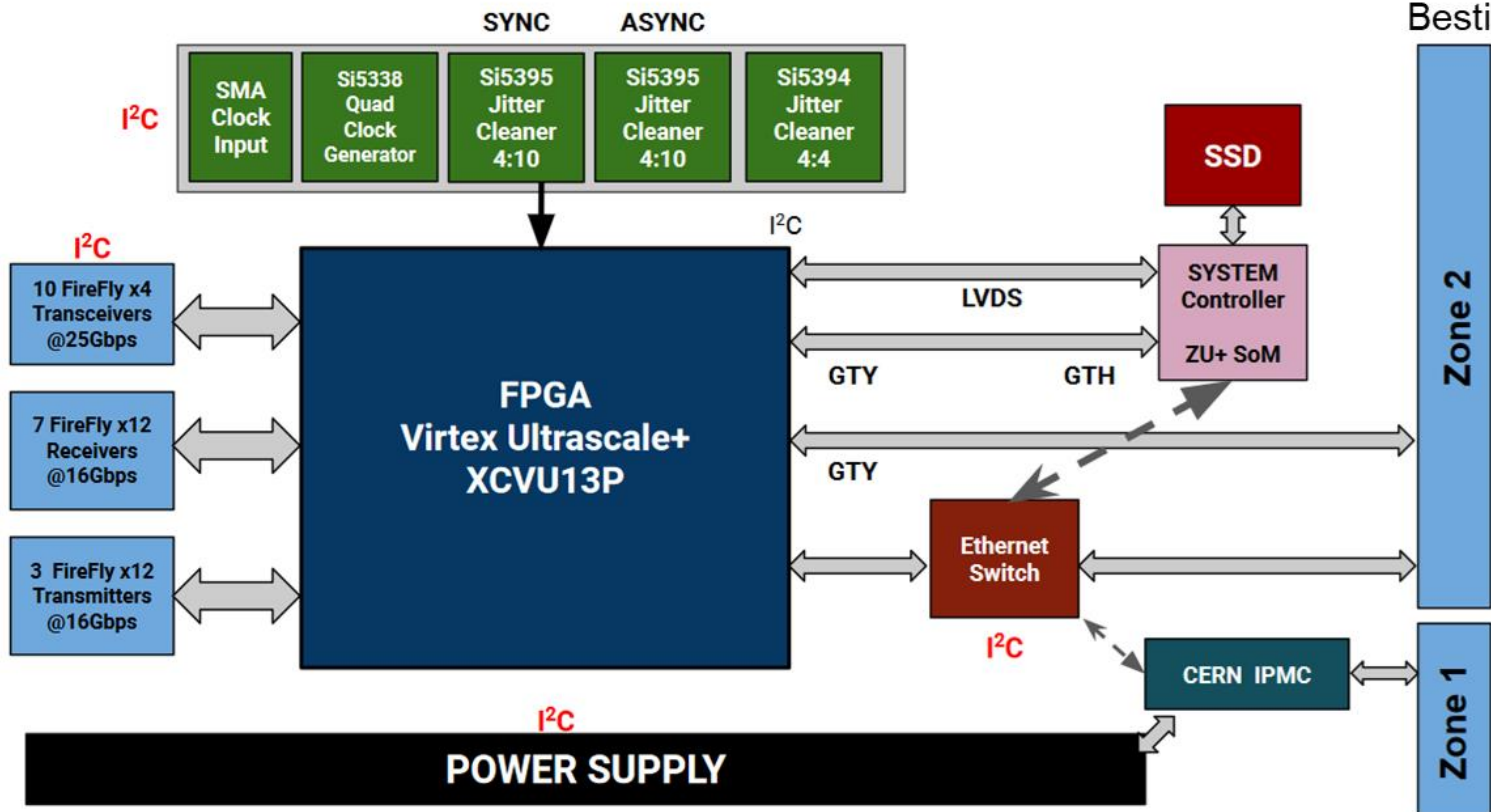
Cell Layouts



BMTL1 board fw modules



From Ioannis Bestintzanos



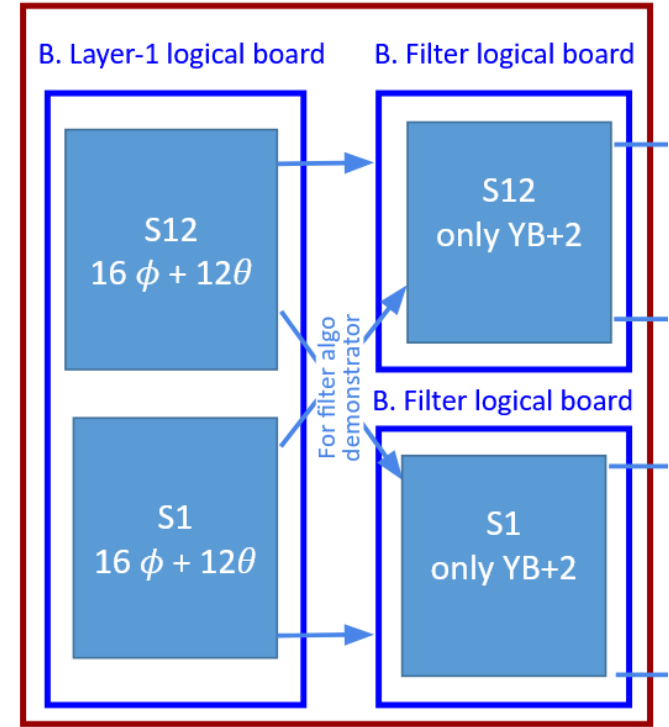
Infrastructure/integration

- EMP, IPbus integration, TCDS, gigabit links (gbtx, lpgbt, CSP), linux infrastructure, under control of Ioannina group
- Readout: readout path from BMTL1 to AB7 (to AMC13 to DAQ) on 12xGBTX links, joint effort Ciemat-Ioannina
- Taking cosmics at SXA5, producing primitives with the new algorithm (reduced clock frequency), planning on moving to UXC before collisions end
- Communication with Ocean established at SXA5, successfully sending trigger primitives with the agreed format

Barrel Filter development

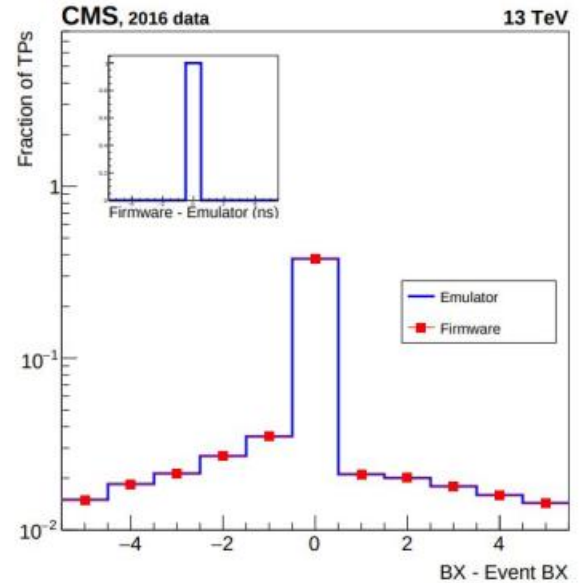
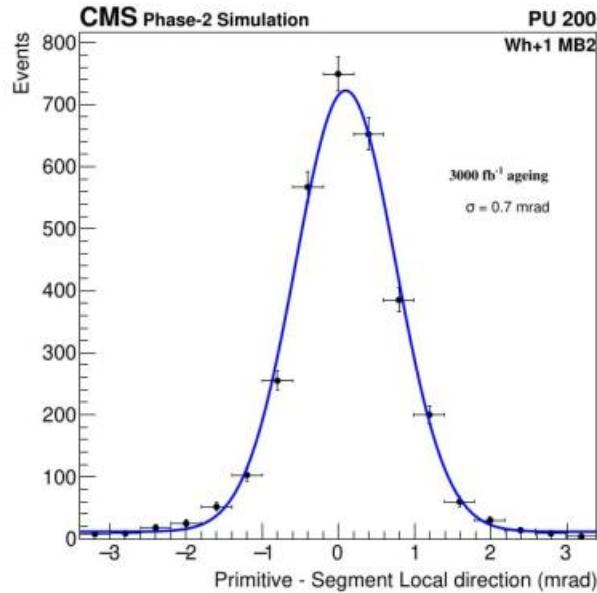
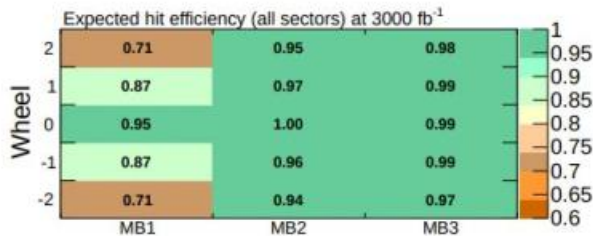
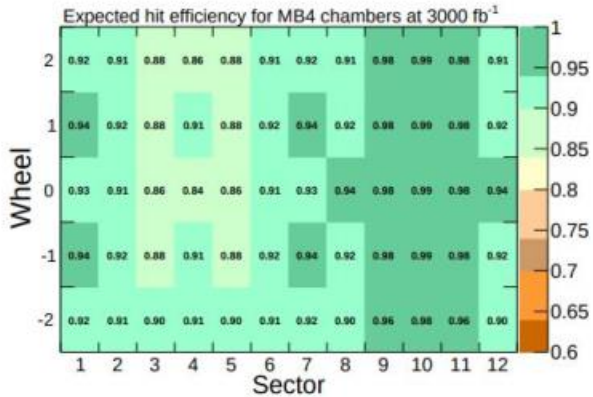
- Since for the moment there is room in the BMTL1 board for more than just the layer one algorithms, we're also using it to start integration of the barrel filter firmware
- Barrel filter is reduced for the slice test (only one wheel per sector/board)

BMTL1 physical board

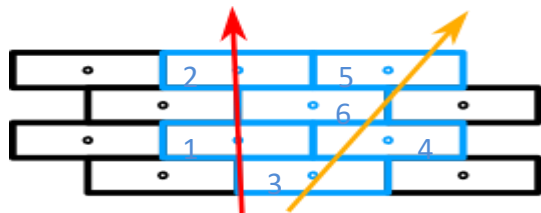


Efficiency and resolution (v2.0)

- Very good **performance** with worst-scenario aging conditions:



Perfect cleaning @ superlayer level requires 16 BX of additional latency



3h = 1,2,3

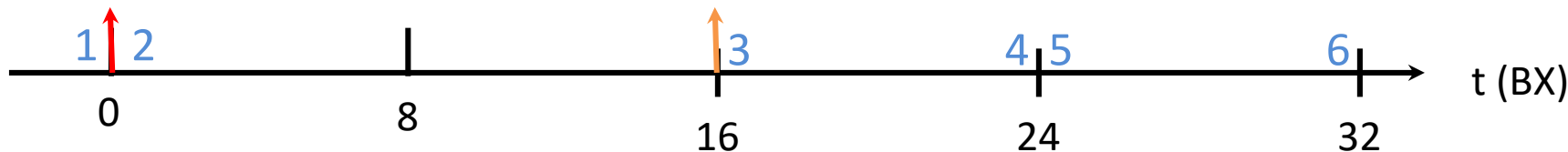
$t_0 = 0$ BX

gen @BX16 (hit3)

4h = 3,4,5,6

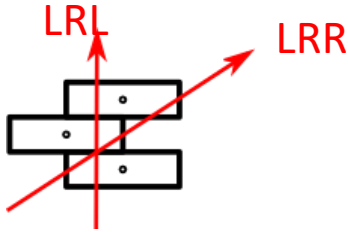
$t_0 = 16$ BX

gen @BX32 (hit6)



- A hit might produce TPGs with $t_0 = t_i - 16\text{BX}$
- A hit might group with hits arriving at $t_j = t_i + 16\text{BX}$ and still produce stronger segments
- For perfect cleaning, 32 BX are needed (16 BX of drift time + 16 BX of artificial latency)
- This is true only for backward filtering (killer segment comes later)
- Forward filtering (killer segment comes first) can be performed without added latency

Perfect cleaning @ matcher level requires 8 BX of additional latency



$$t_1 = t_2 = t_3$$

$$t_0 \text{ (LRL)} = t_1 - 8 \text{ BX}$$

$$t_0 \text{ (LRR)} = t_1 - 16 \text{ BX}$$

- A given group of 3 hits might produce ghosts separated by 8 BX
- If the first one can only be released after the later one hasn't made a correlation, it has to be held for 8 additional BX
- (Again,) this is true only for backward filtering (killer segment comes later)
- (Again,) forward filtering (killer segment comes first) can be performed without added latency