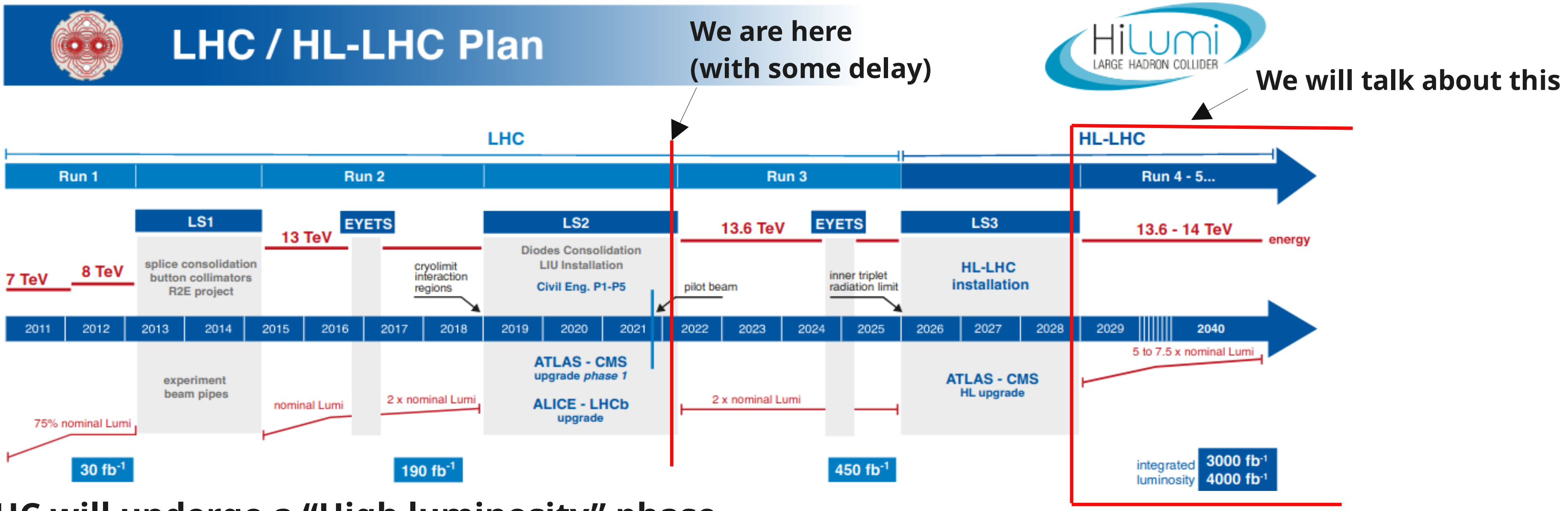
L1 triggering on High-Granularity information at the HL-LHC

Louis Portales (LLR, CNRS) on behalf of the CMS Collaboration

CALOR 2022, Brighton – 19/05/2022

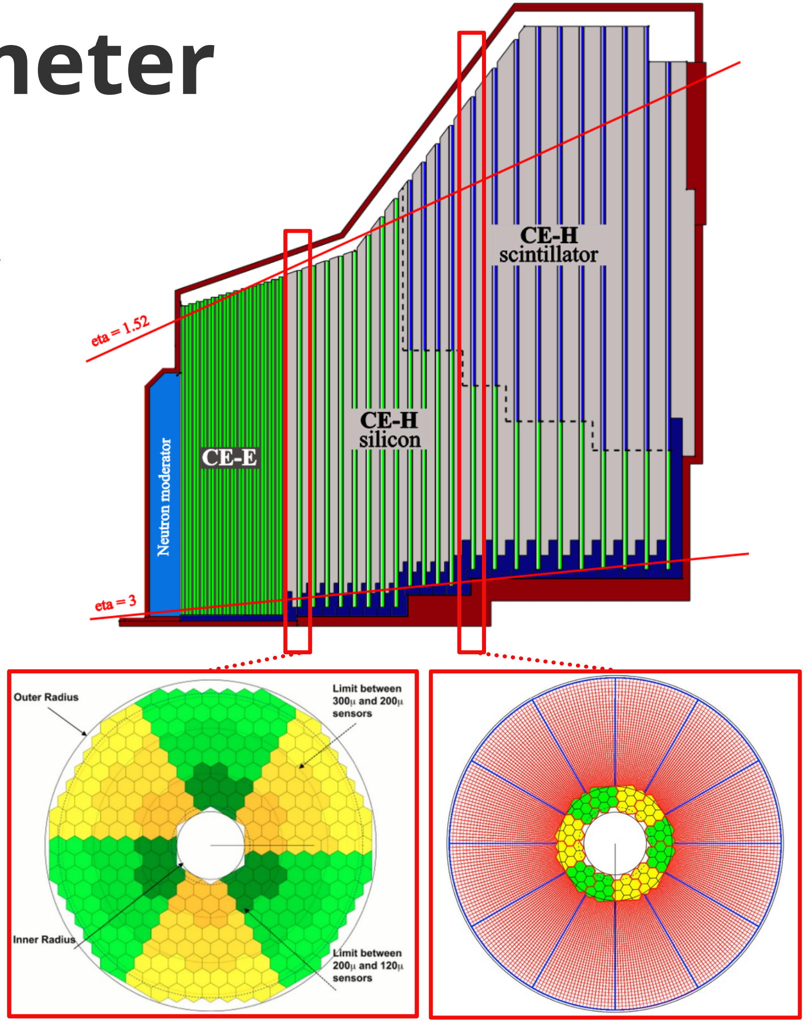
HL-LHC



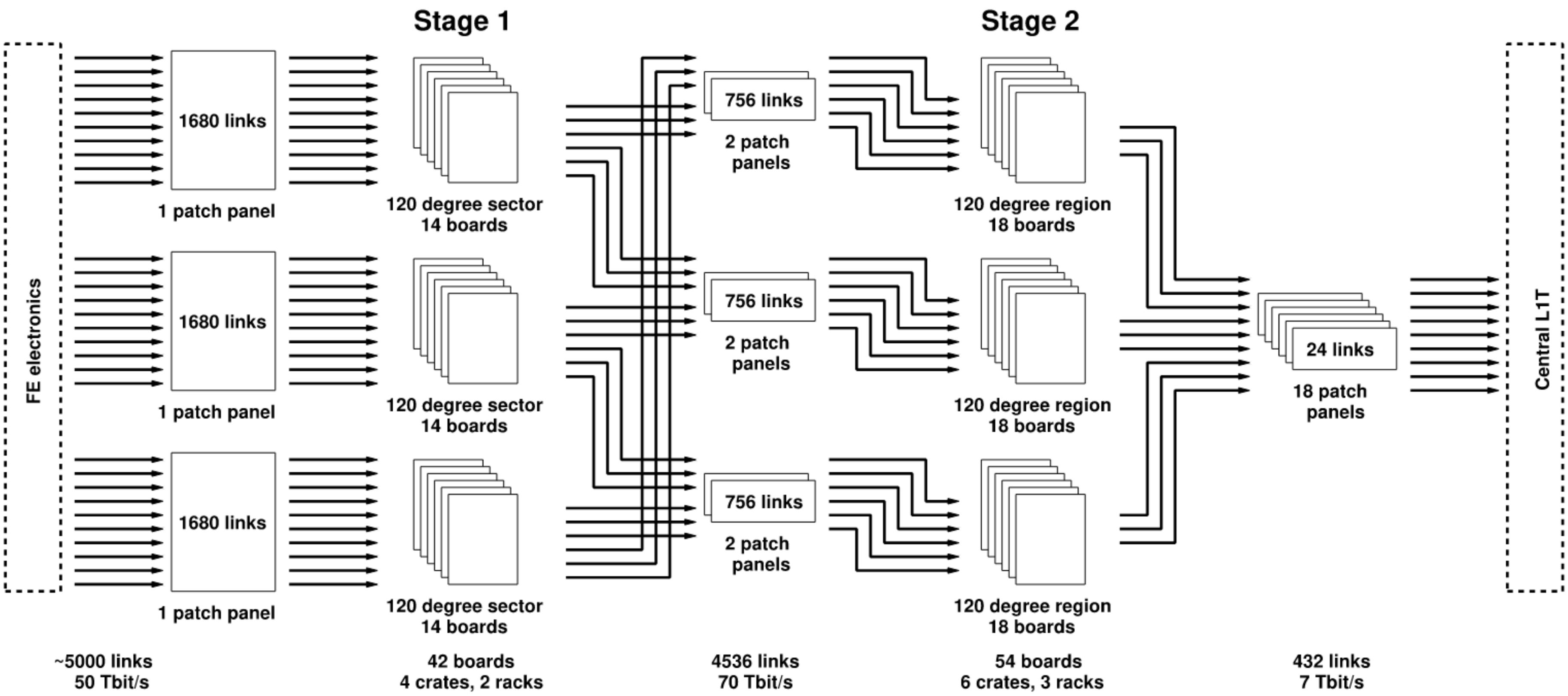
- **LHC will undergo a “High luminosity” phase**
  - **$L = 5 - 7.5 \times 10^{34}$ cm^{-2}s^{-1} and pileup of up to <μ>=200**
    - → About x4 LHC !
  - **Extreme radiation level, detector occupancy and rates are expected**
    - → Challenge for L1 trigger
    - → Must allow to pursue LHC physics program → maintain (at least) Phase 1 trigger thresholds

High Granularity Calorimeter

- **CMS Endcap calorimeters replaced by HGCal at HL-LHC**
  - **“5D” detector**
    - → Position + energy + timing
  - **Radiation hard & Highly granular:**
    - → Hexagonal Silicon (Si) modules in high-radiation region
    - → Scintillating tiles (+ SiPM readout) in lower-radiation region
    - → ~ 6M channels in total (~1M used for trigger)
  - **Thorough description in talk by Z.Gecse**
- ***HGCal will be part of L1 triggering system***
  - **Will be used to generate trigger primitives**
    - → 3D clusters & trigger towers used as input to Level 1 trigger



HGCal Trigger Primitive Generation (TPG)

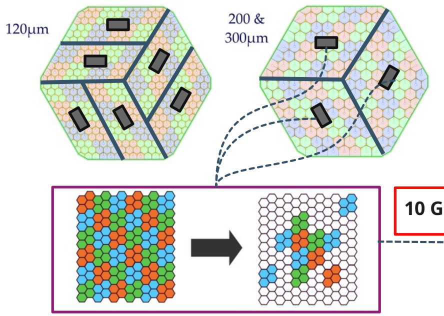


*One endcap

HGCal TPG – Electronics

Stage 1 | Stage 2

Front-end



10 Gbps lpGBT

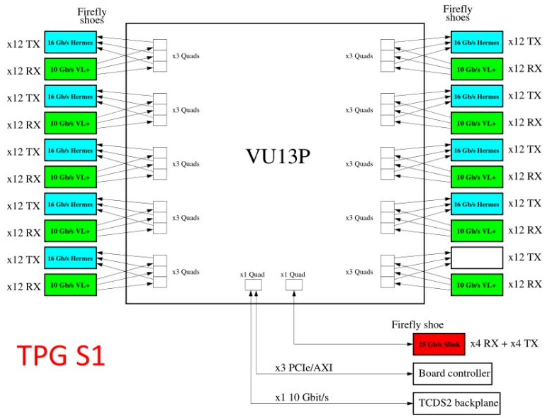
HGCROC

→ Measure cell charges & ToA (@ 40 MHz)
→ Groups cells by 4 (or 9) into Trigger Cells (48 TCs / module)
→ Sums charges per TC

ECON-T

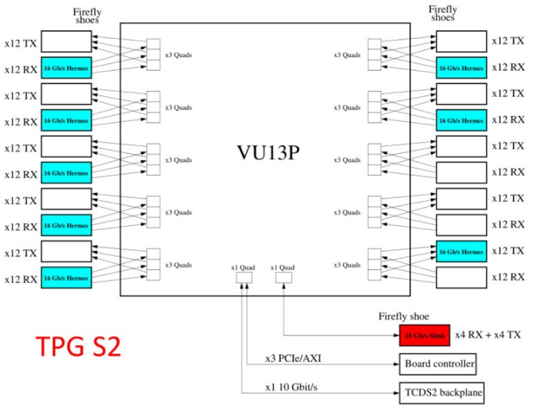
→ Selects + compresses TCs to be sent to backend

Back-end



TPG S1

→ 14 boards per 120° sector (42 total)
→ Unpacks data from ECON-T
→ Builds partial Trigger Towers
→ Sorts & truncates TCs
→ Time multiplex (TMUX) x18 for Stage 2



TPG S2

→ 54 boards in total
→ Builds 3D Clusters from TCs
→ "Finalizes" Trigger Towers
→ Packs data to be sent to L1T

Central L1T

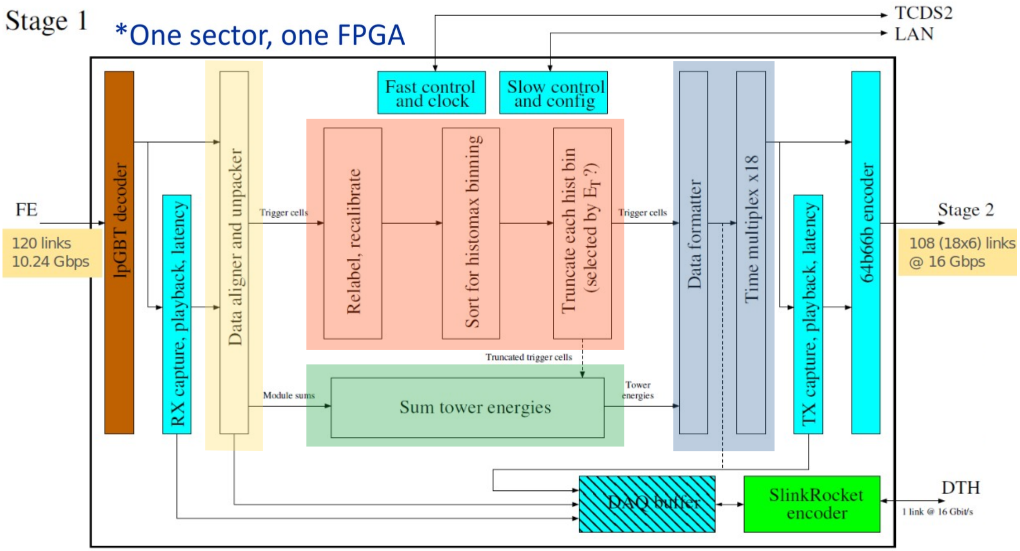
→ Dedicated Serenity ATCA boards for S1 & S2
→ Single FPGA type: Xilinx VU13P, with up-to 120 links

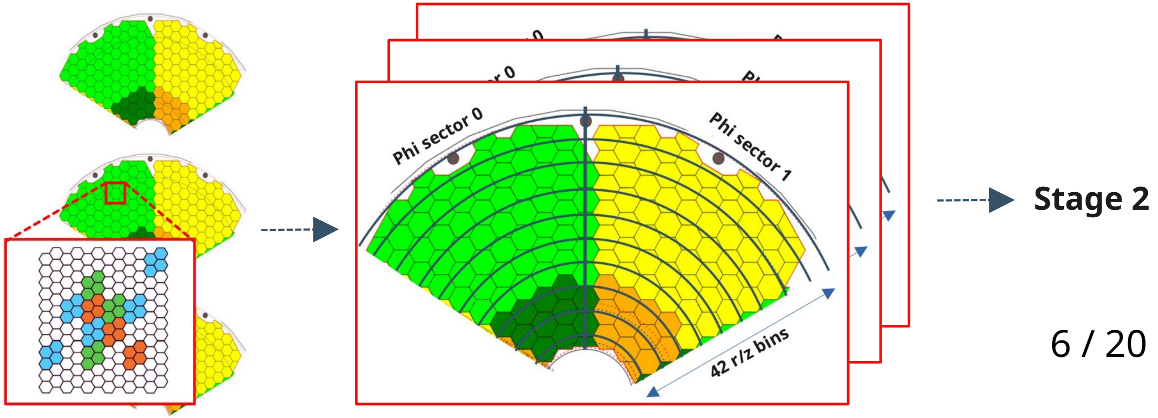
See also talk by N.Strobbe

*One endcap

HGCal TPG – Stage 1 algorithmic blocks

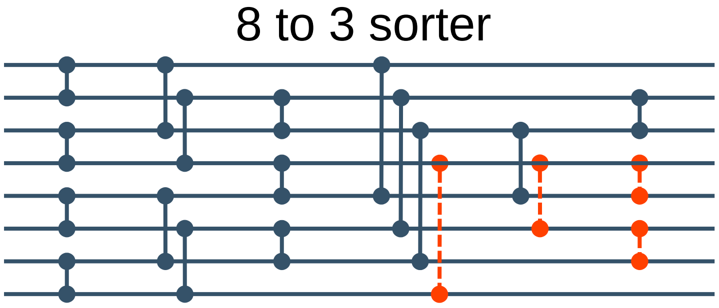
- **Data aligner / unpacker**
  - **Groups inputs per bunch crossing (BX)**
  - **Uniformizes (unpacks) data from FE**
    → Variable-size inputs
- **Partial tower sums**
  - **Sums module energies into towers**
    → CE-E / CE-H summed tower energies (pTT)
- **TC sorting and truncation**
  - **Sorting/truncation in (r/z,Φ) bin**
    → Returns N most energetic TCs per bin
- **Data formater / packer**
  - **Reformates data to be sent to stage 2**
    → + 18 BX time multiplexing

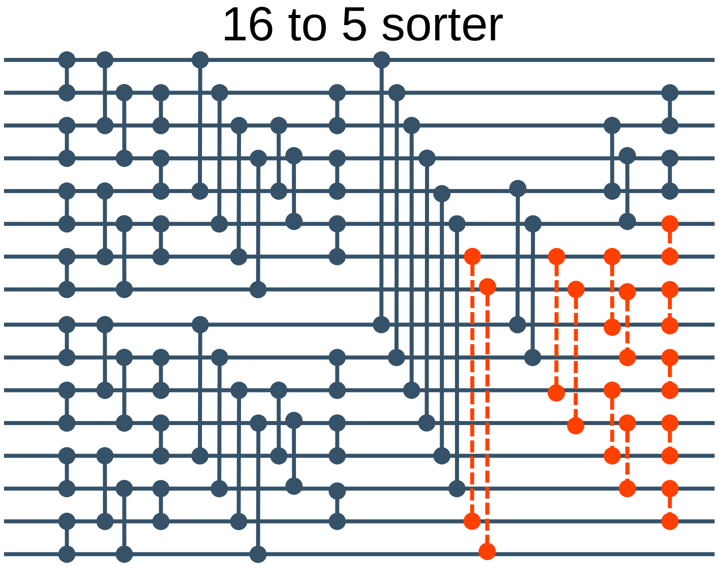




HGCal TPG – Stage 1 Sorting & truncation

- **TCs regrouped in 3D regions of the detector**
  - **84 “2D” bins per 120° sector**
    → (2 phi regions)x(42 r/z bins)
  - **Dedicated sorting and truncation per bin**
    → Optimized regarding both resource requirements and physics impact
- **Need efficient sorting algorithm on FPGA**
  - **Batcher odd-even mergesort algorithm**
    → Succession of small sorter and merger blocks
    → Adapted to FPGA logic & resource-efficient
    → Adaptable to arbitrary numbers of inputs
- **On-the-fly truncation while sorting**
  - **Fixed output size per bin**
    → Can remove comparators pointing to truncated outputs



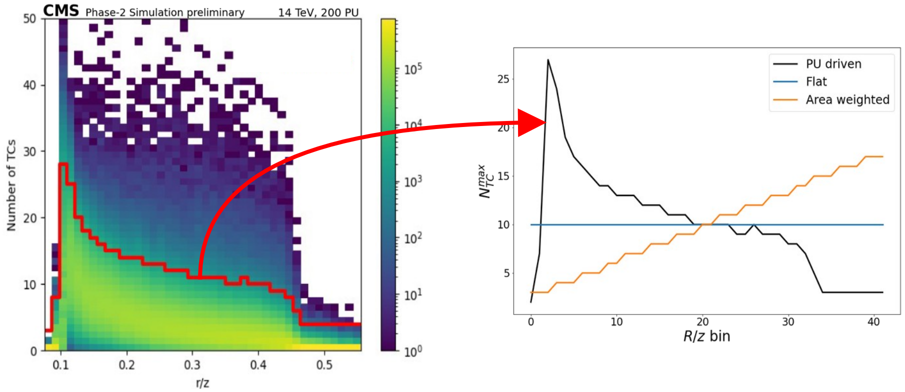


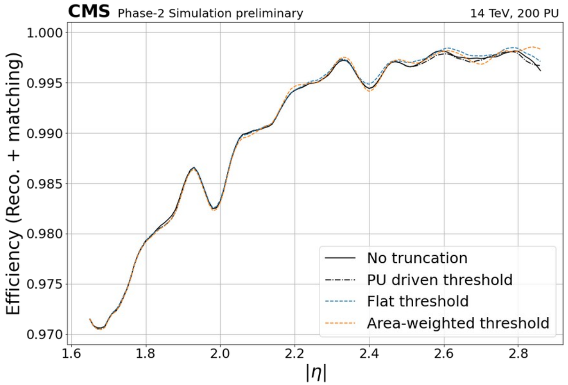
HGCal TPG – Stage 1 optimization

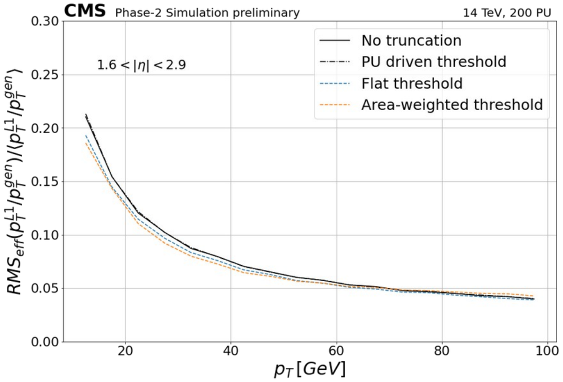
- **Tuned FE → Stage 1 link mapping**
  - **Aiming at balancing load per FPGA**
    → Achieved equivalent total N_{TC} per FPGA
- **Truncation parameters investigated**
  - **Output N_{TC} limited by Stage 1 → Stage 2 bandwidth**
    → Max 420 TC per link (2 links/FPGA) with VU13P
  - **Several truncation profiles investigated**
    → **Default**: “PU-driven” truncation
      - Aimed at preserving constant efficiency, truncating as little TCs as possible

    → **Flat**: equal N_{TC} per bin
      - Better distribution of hardware ressources

    → **Area-weighted**: N_{TC} dependant on bin size
      - Heavy truncation in denser (low r/z) regions
  - **Area-weighted truncation yields best results**
    → Better overall resolution



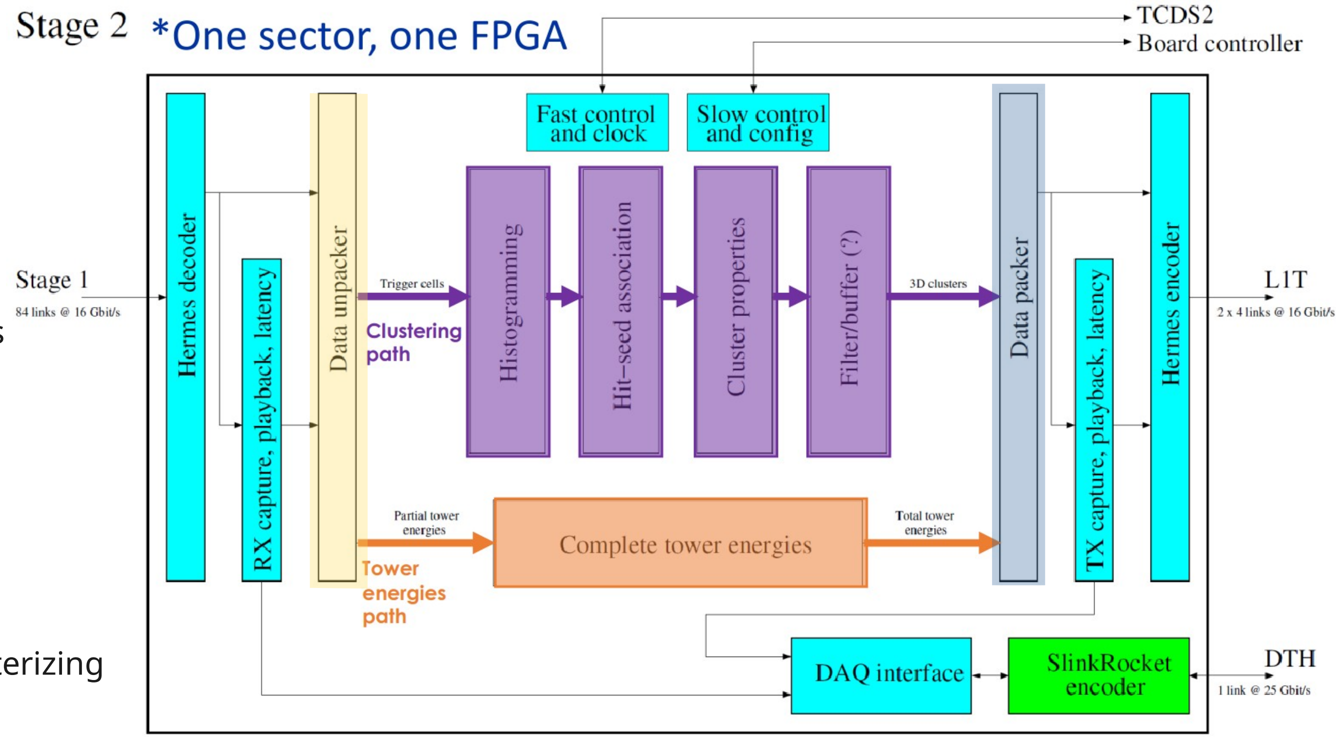




e/γ clusters

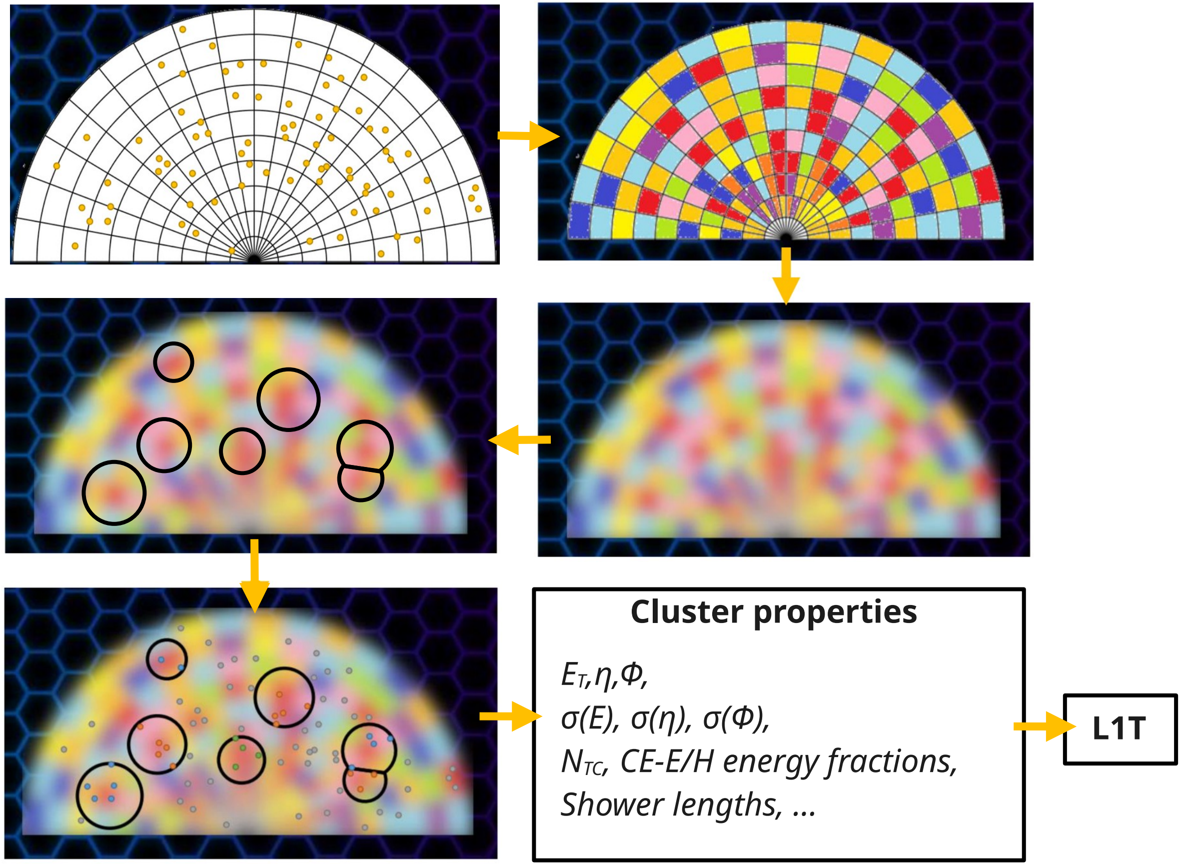
HGCal TPG – Stage 2 algorithmic blocks

- **Tower energies**
  - **Sums Towers from Stage 1**
  - → Into map of 20x24 (η,Φ) bins
- **TC clustering**
  - **Histogramming**
  - → Forms 2D energy map in (r/z, Φ) bins
  - **Hit-seed association**
    → Looks for local maxima in histogram
    → Clusters TCs around maxima
  - **Cluster properties**
    → Computes various quantities characterizing the clusters
  - **Filter/buffer (if required)**
    → Bandwidth control: selects clusters to be sent to L1T



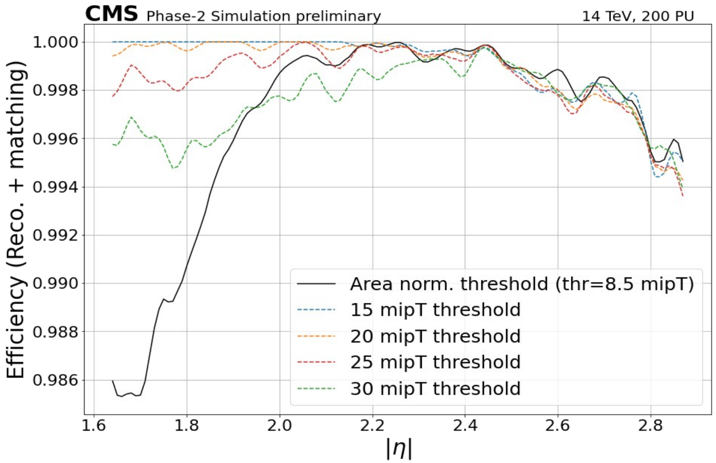
HGCal TPG – Cluster building

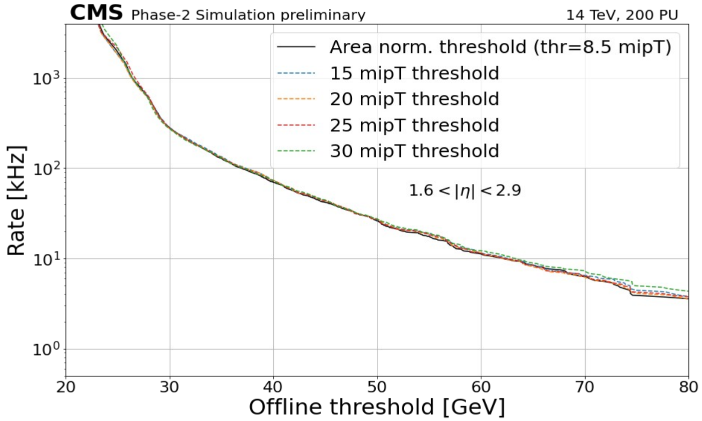
- **Histogramming**
  - **Finer binning in Φ than in Stage 1**
  - → 42x108 (r/z, Φ) bins, in wider Φ range (180°)
  - **Histogram smeared for improved peak finding**
- **Hit-seed association**
  - **Local maxima finder + theshold**
    - → Bins above threshold become seeds
    - → Evaluate seeds position from TC average
  - **Clustering**
    - → TCs within $\Delta R^2=1$ of seed are clustered with R in the (x/z, y/z) plane
- **Cluster properties**
    - → Build Kinematic & quality information to be sent to L1T
    - → Selected and optimized accounting for hardware constraints



HGCal TPG – Cluster building optimization

- **Clustering parameters tuned**
  - **Several seeding thresholds considered**
    - → **Default**: Area-weighted threshold
      - Area normalization applied in histogram smearing step
      - Yields η-dependent threshold
      - Allowed for ~ constant PU density vs η
      - But found to bring efficiency loss
    - → Constant thresholds (**15**,**20**,**25**,**30** **mipT**):
      - Removing area normalization from smearing
      - Restores efficiency at lower η
      - Yield similar rates for e/γ object
  - **Further tuning performed/ongoing, e.g.:**
    - → r/z bin edge definition
      - To reduce artificial shower-splitting effects at high |η|
    - → Cluster variables encoding
      - Multi-objective optimization (physics impact + resources)

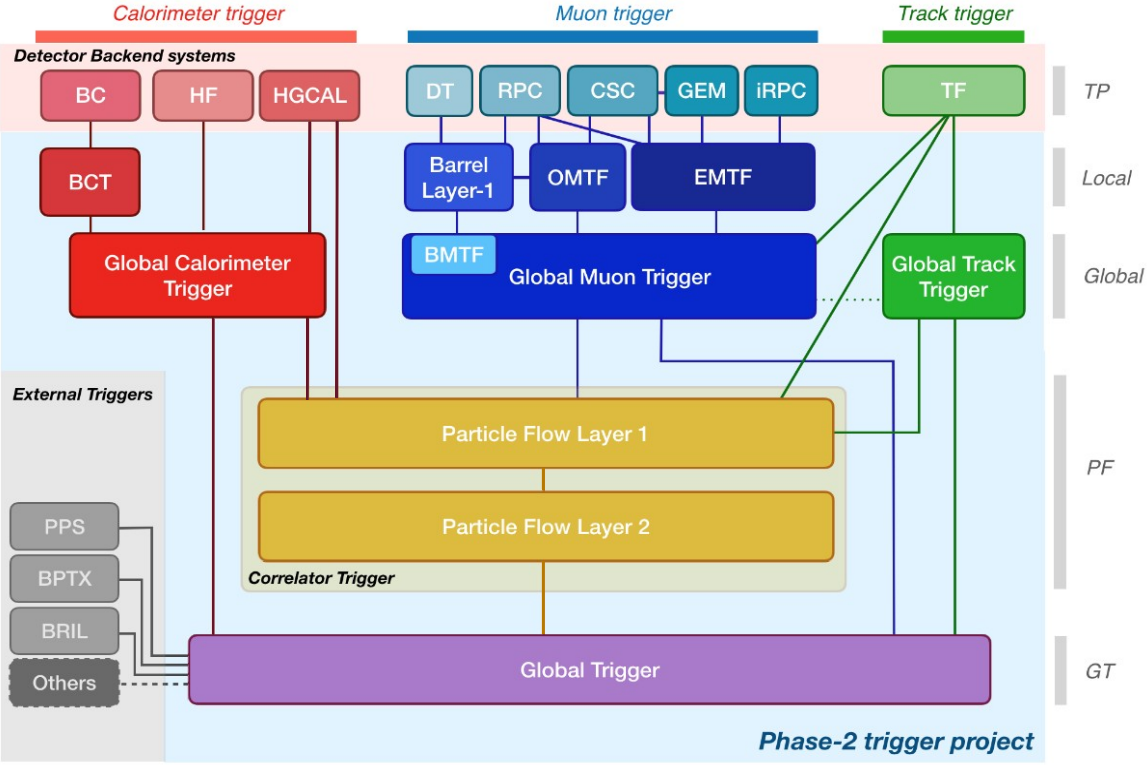




e/γ clusters

HGCal → L1 Trigger

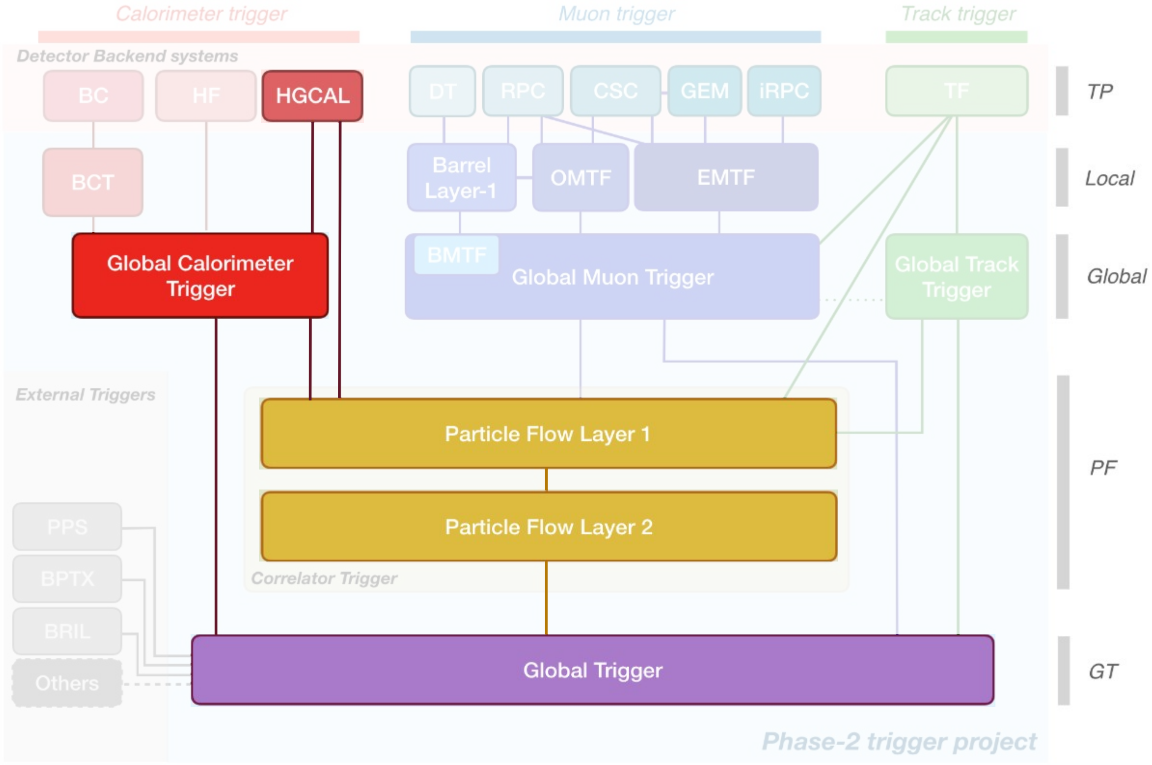
- **HGCal TPG part of much larger-scale upgrade**
  - **Calorimeter Trigger**
    → *7-500x (!) improved granularity w.r.t. Phase 1*
  - **Muon Trigger**
    → *extended to $|\eta| = 2.8$*
  - ***Track Trigger***
    → *up-to $|\eta| = 2.4$*
- **HGCal TPG contributes to**
  - **Global Calorimeter Trigger**
    → Builds standalone e/γ objects, Calo jets, τ_h and H_T *up-to $|\eta| = 3$*
  - **Correlator Trigger**
    → Combines Calo, muon and track triggers
    → Implements *advanced reconstruction techniques* (e.g. Pflow, PUPPI) so far limited to offline reconstruction



See also poster by P.Kumar

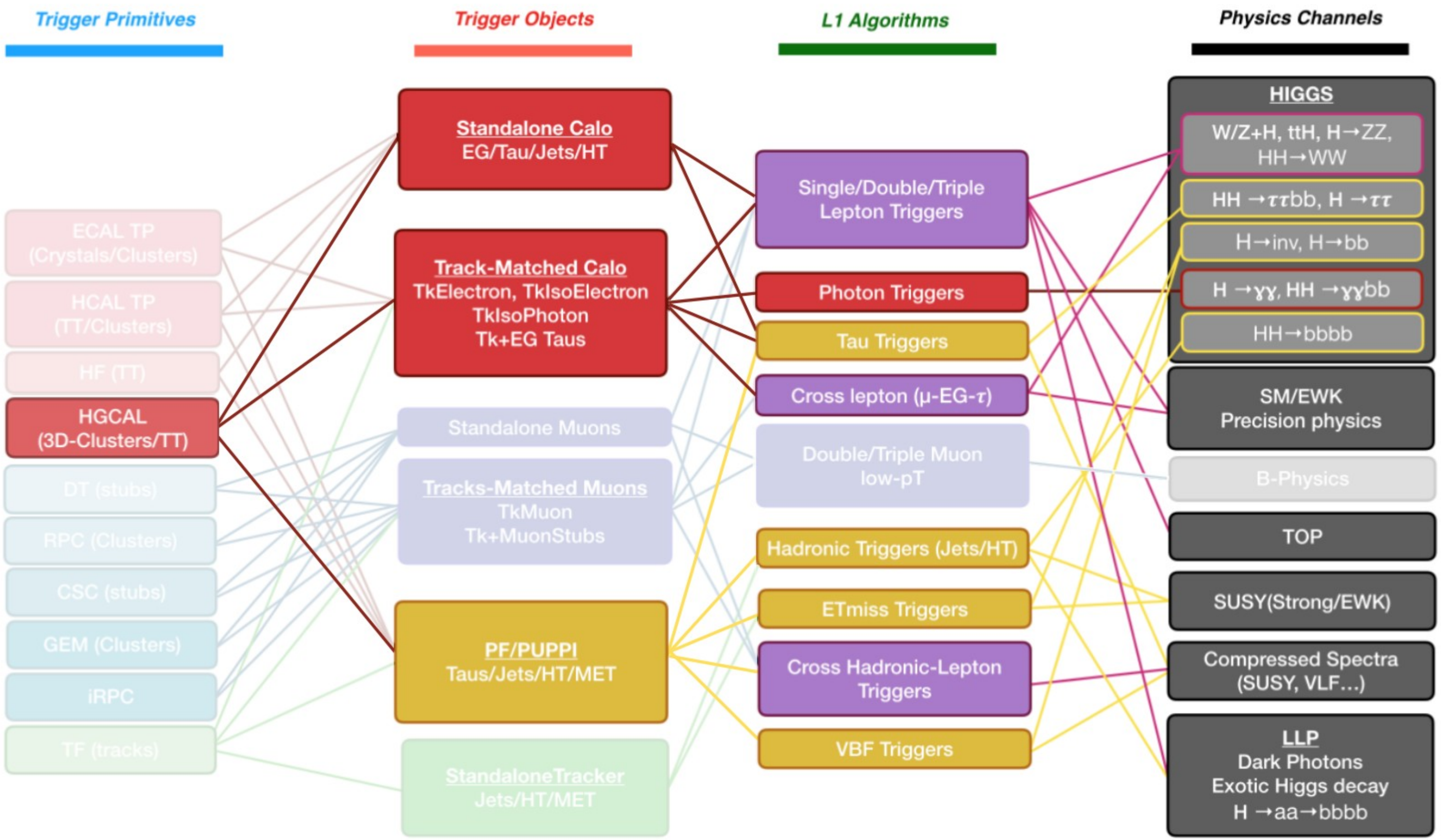
HGCal → L1 Trigger

- **HGCal TPG part of much larger-scale upgrade**
  - **Calorimeter Trigger**
    → *7-500x (!) improved granularity w.r.t. Phase 1*
  - **Muon Trigger**
    → *extended to $|\eta| = 2.8$*
  - ***Track Trigger***
    → *up-to $|\eta| = 2.4$*

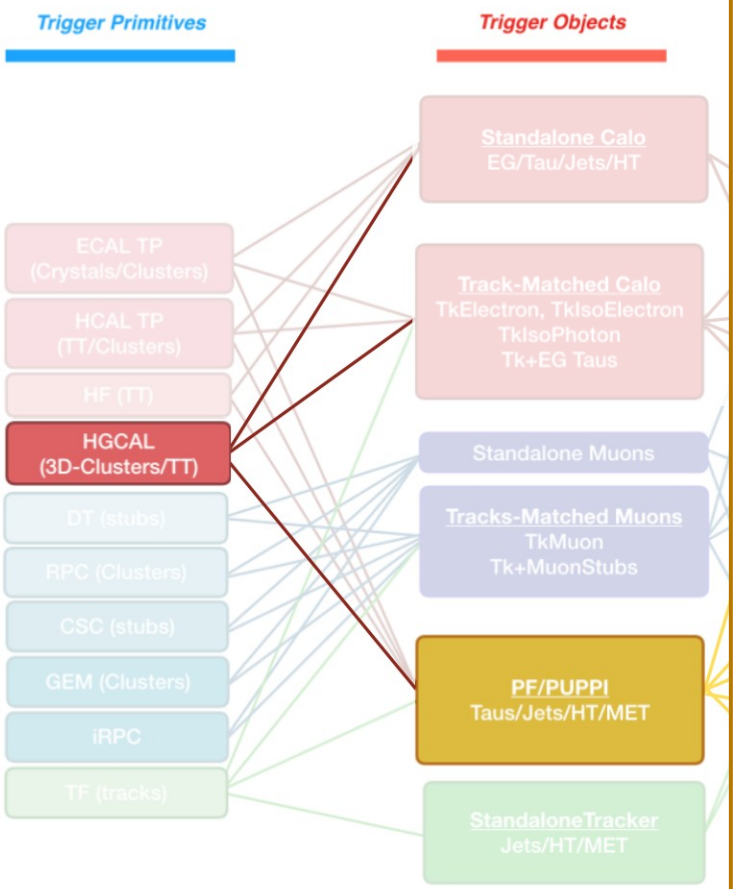


- **HGCal TPG contributes to**
  - **Global Calorimeter Trigger**
    → Builds standalone e/γ objects, Calo jets, τ_h and H_T *up-to $|\eta| = 3$*
  - **Correlator Trigger**
    → Combines Calo, muon and track triggers
    → Implements *advanced reconstruction techniques* (e.g. Pflow, PUPPI) so far limited to offline reconstruction

L1 Trigger algorithms & HGCal



L1 Trigger algorithms & HGCal



PFlow & PUPPI

”Particle Flow”:

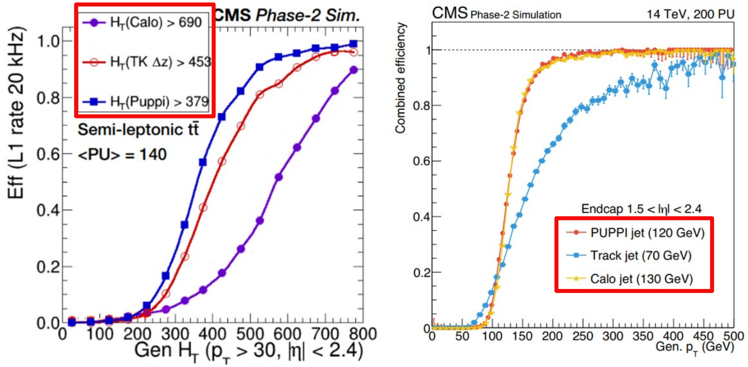
- → **Offline/HLT reconstruction algorithm in CMS**
- → **Adapted to L1 reconstruction**

“Pile-Up Per Particle Identification”:

- → **Originally Offline PU mitigation algorithm**
- → **Using vertexing to remove contributions from charged PU**
- → **Remove/weight-down contribution from neutral PU using *local* information only (event-wide for offline version)**

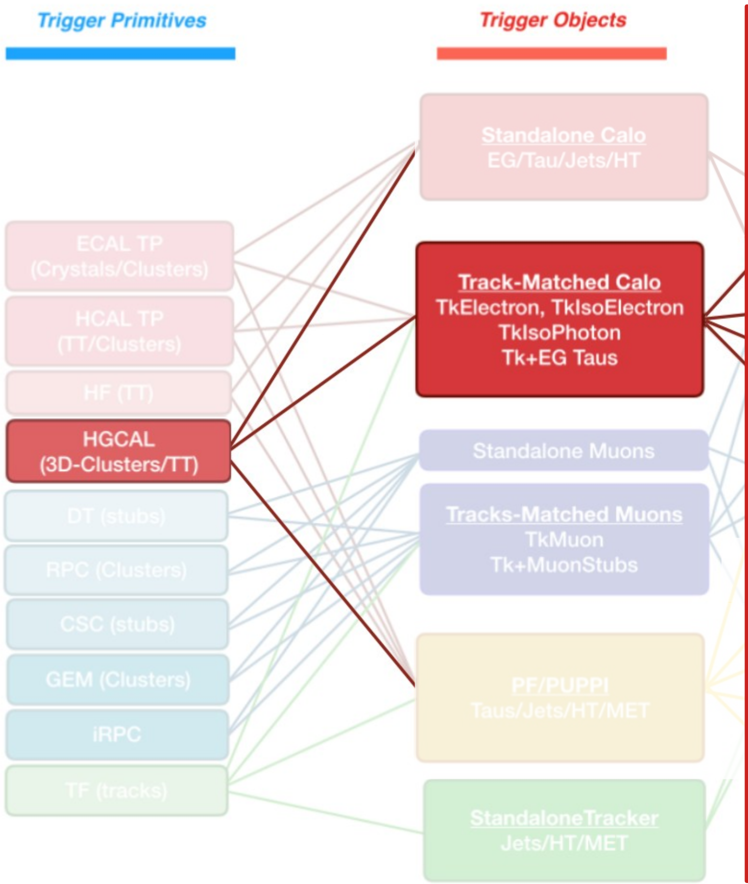
Both implementable at L1T

- → **Thanks to tracking & vertexing capability, and upgraded L1T hardware and optimized firmware**
- → **Allow to reconstruct clean, PU-ridden trigger objects**
- → **Up to |η| = 2.4 (Track trigger + HGCal acceptance)**

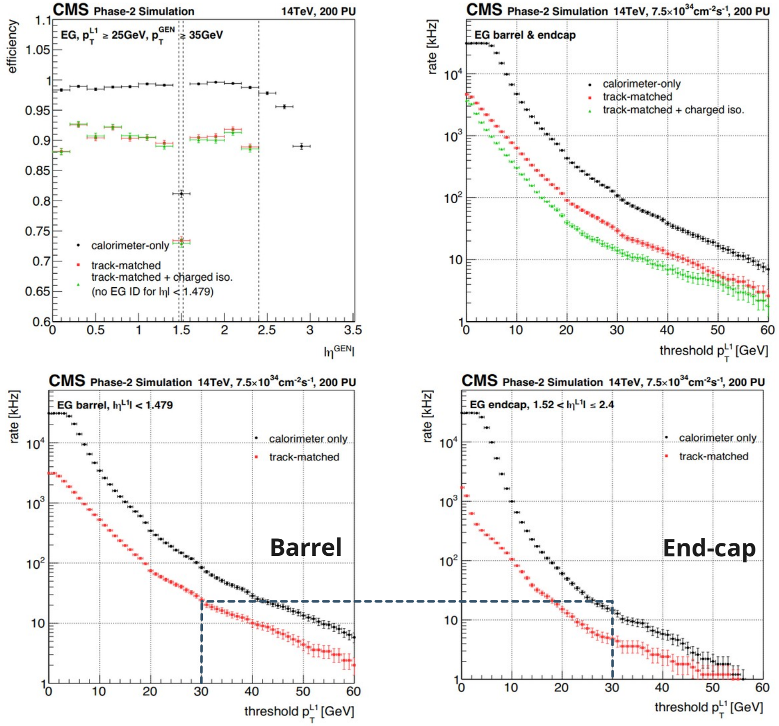


Efficiencies at fixed rate

L1 Trigger algorithms & HGCal

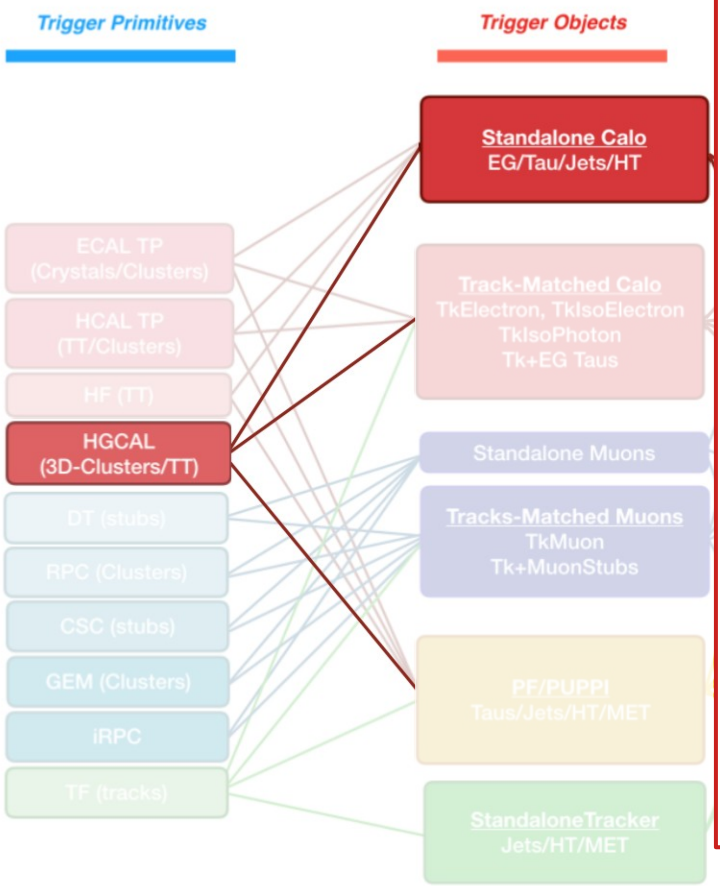


Track-matched e/γ



→ Constructed above |η| = 1.5 from HGCal clusters
→ Similar reconstruction efficiency as in Barrel
→ Significantly smaller rates

L1 Trigger algorithms & HGCal

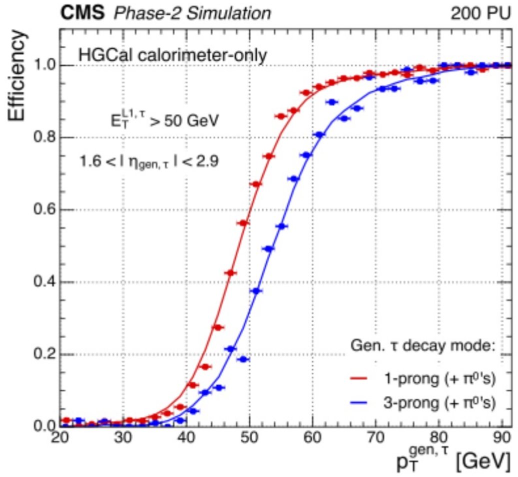


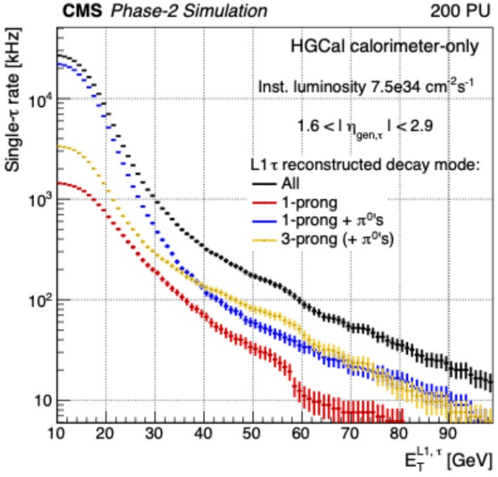
Calorimeter Tau with HGCal

Example of ML-based trigger algorithm

- → **Complements PUPPI Tau algorithm for higher p_T objects**
- → **Takes 3D Calorimeter clusters as inputs**
- → **Passed through dedicated BDTs for discrimination against PU (using cluster-shape variables)**
- → **Calibrates clusters (PU substraction + energy correction)**
- → **Identification Tau decay modes using a Random Forest Classifier**

→ ***Showing promising results already, with sensible improvements considered***

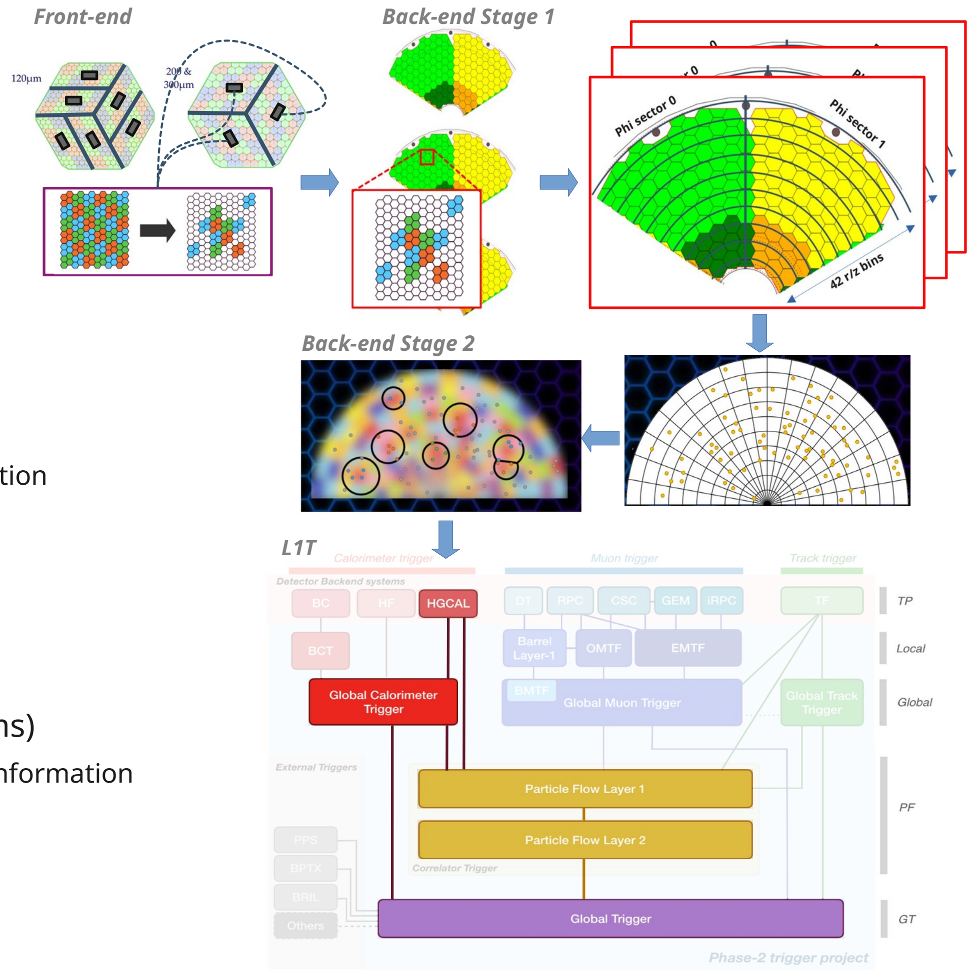




H →aa→bbbb

Summary

- **HGCal will be part of the CMS L1 trigger at HL-LHC**
  - Sending Trigger Primitives (Clusters and Towers) to L1T
- **The different steps of TPG were improved since the L1T TDR**
  - Sorting & truncation of the TCs sent from the Front-end
    - Implementing optimal sorting algorithm in FPGA
    - Optimized TC truncation parameters yielding improved resolution
  - Cluster building from TCs
    - Robust chain in place
    - Fine-tuning of parameters, improved efficiency
- **Wide influence of HGCal on L1 trigger performance**
  - Both for e/γ and hadronic objects (jets, MET & energy sums)
    - Excellent performances at high |η| with the help of tracking information
    - Complementary to PFlow/PUPPI objects at high p_T
- **Firmware implementation and hardware tests are ongoing**
  - Both for the HGCal TPG and L1T algorithms
    - A large part already implemented/tested
    - More optimizations & new ideas still to come



Back up

ML @ L1 Trigger

- **Many ML-based improvements**
  - **Constraint by FPGA implementation**
    → Dedicated workflow in place
    → Already allowed inclusion & test in FW of many algorithms
  - **Dedicated workflow in place**
    → hls4ml
    → ML model (python) → HLS conversion
    → Allows for model testing directly in firwmare (or firmware emulator)
  - **Already allowed inclusion & test in FW of many algorithms**
    → Some examples following