Hardware Based Tracker Track Finders for Triggering at HL-LHC

Mark Pesaresi *on behalf of the CMS and ATLAS collaborations*

with thanks to Nikos Konstantinidis, Alberto Annovi, Peter Wittich, Kristian Hahn, Fabrizio Palla, Tom James, Giacomo Fedi

ACES 2018 - Sixth Common ATLAS CMS Electronics Workshop for LHC Upgrades

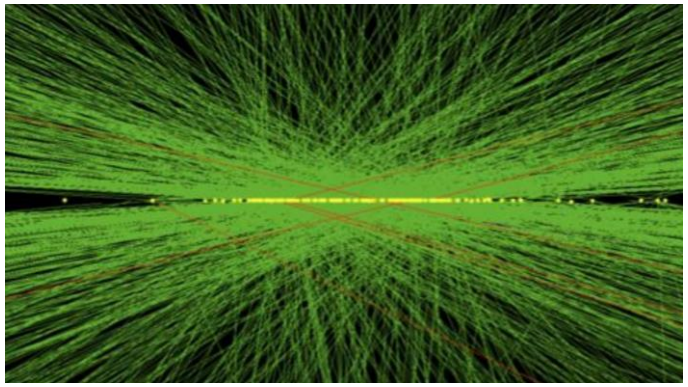
25th April 2018

TRIGGERING USING THE TRACKER

HL-LHC will deliver higher luminosity than ever

- Ultimate luminosity of 7.5×10^{34} Hz/cm^2

$\Rightarrow$ **200 pileup** (PU) interactions per 25ns crossing



All new silicon trackers for ATLAS & CMS at HL-LHC

- Higher granularities to cope with increased fluence

[e.g. CMS ~75 million channels -> ~2 billion channels]

Isn't tracker data already use in the trigger?

- CMS & ATLAS : In the CPU-based HLT processing farm (100 kHz)
- ATLAS : Also, Fast TracKer (FTK) hardware tracking pre-processor now coming on-line (100 kHz rate) -> frees significant CPU time

Why might tracker data needed earlier in the trigger?

- Detectors will be **triggered O(10) more often** than present

[e.g. ATLAS 100 kHz -> 1 MHz]

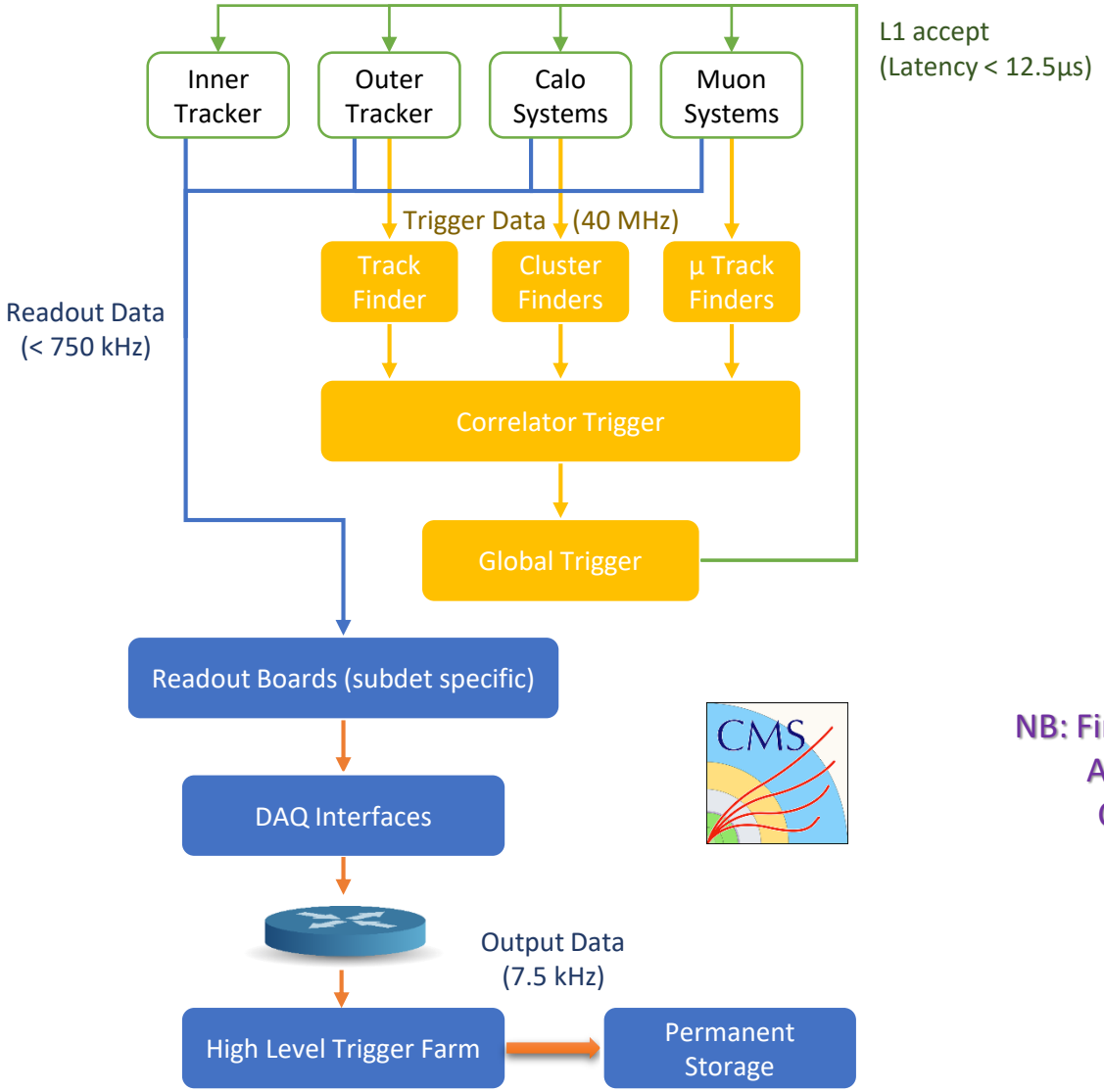
- **Huge amounts of data** will be produced

[e.g. CMS 2 Tb/s -> 50 Tb/s]

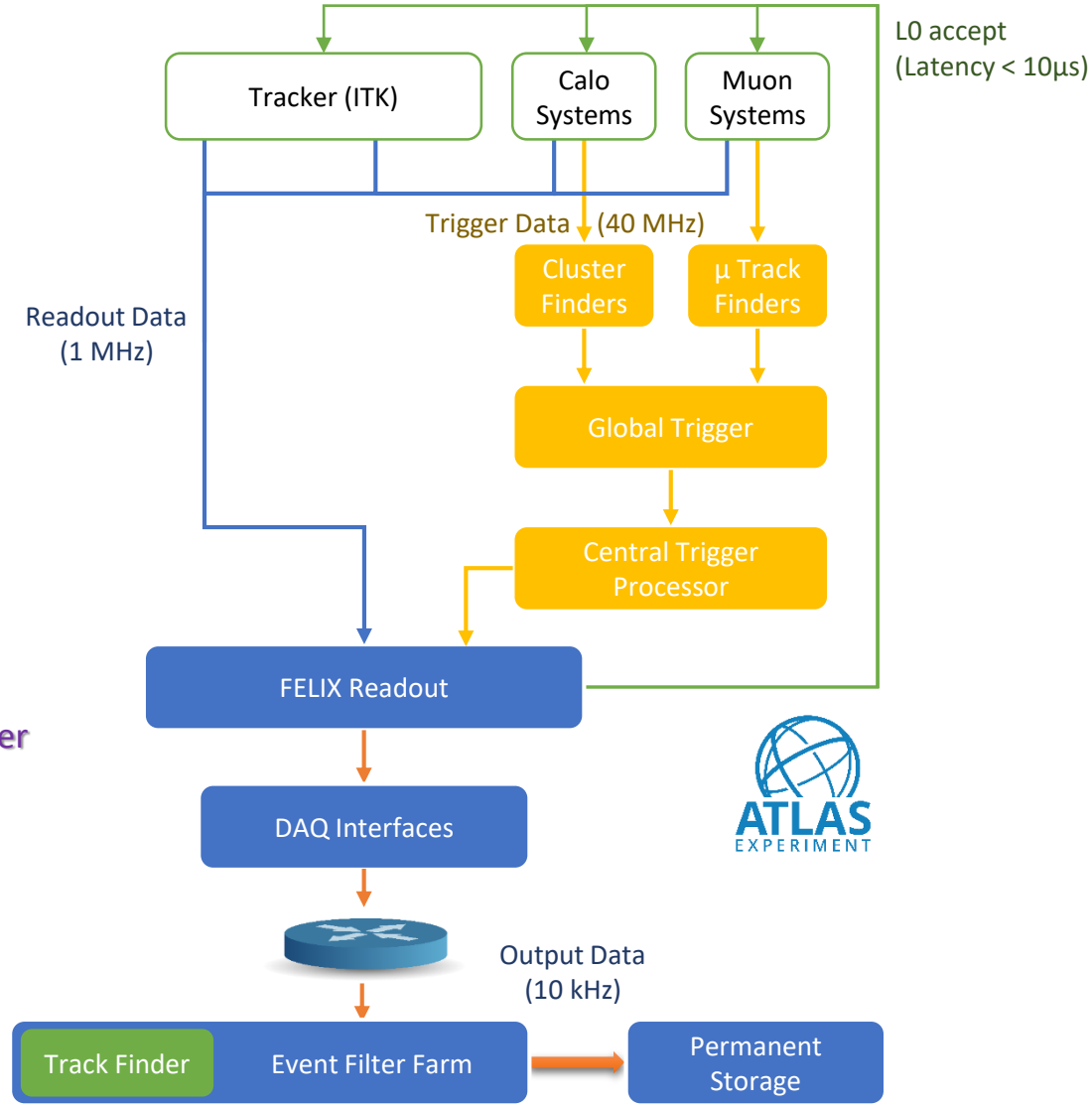
Extended performance at high pileup is important

- But just *surviving* will be will be difficult enough!

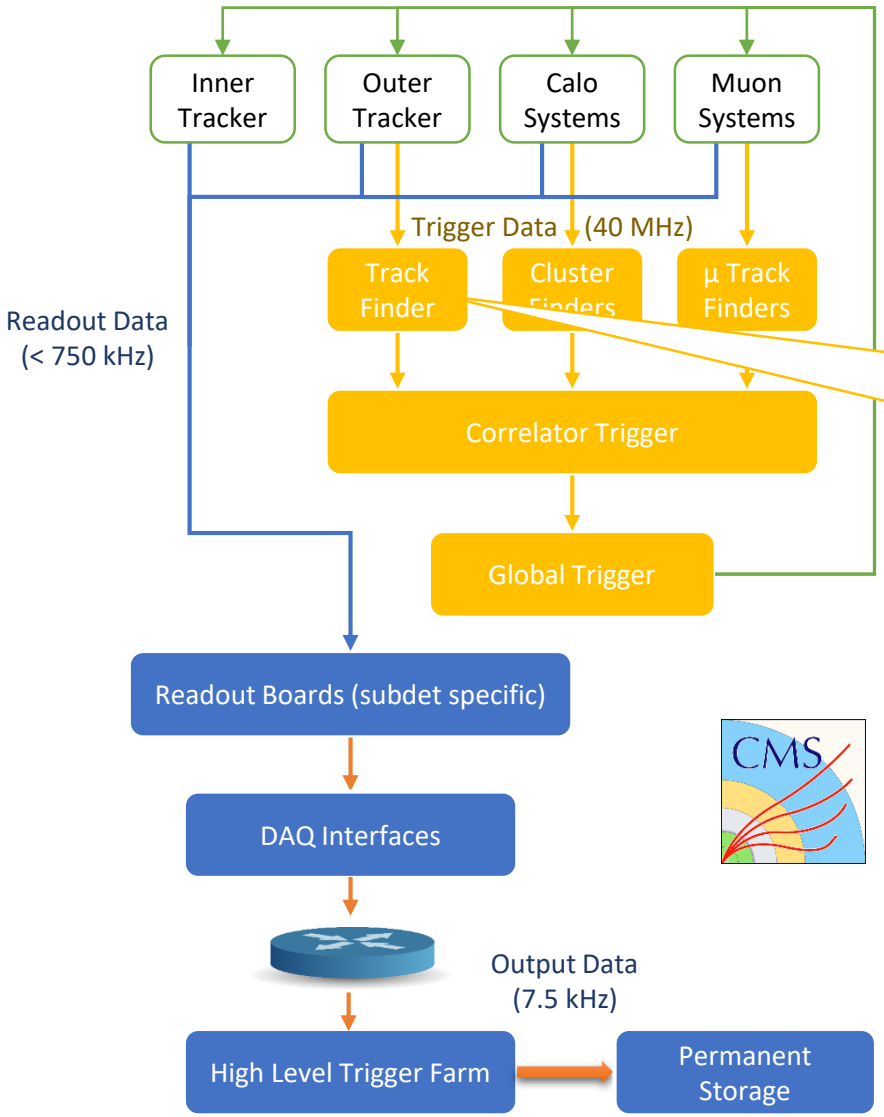
[SIMPLIFIED] HL-LHC TRIGGER ARCHITECTURES



NB: First Level Trigger
ATLAS => L0
CMS => L1



[SIMPLIFIED] HL-LHC TRIGGER ARCHITECTURES



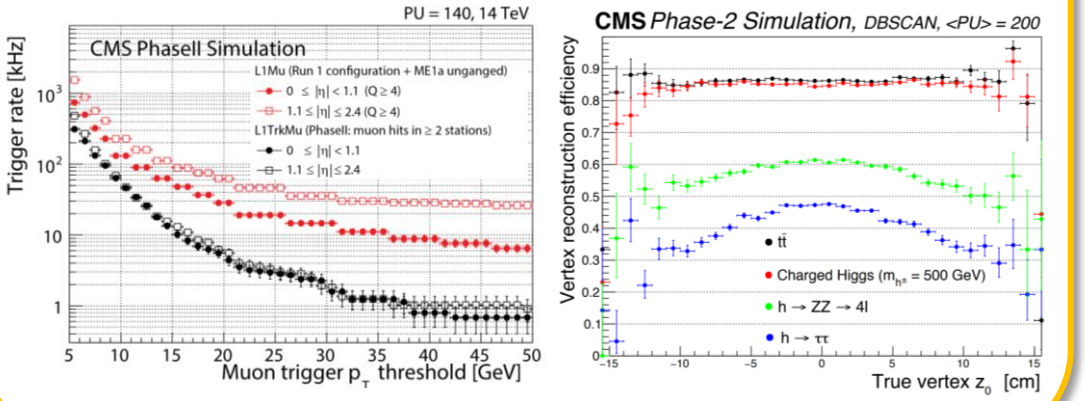
NB: First Le
ATLAS
CMS =

Necessary to include tracking information at **first level of triggering**

Rate w/o tracking approaches **4 MHz** at 200PU!

Many trigger thresholds can be either maintained or improved – typically ~ x10 reduction

Vertexing a new handle at L1



Output Data
(10 kHz)

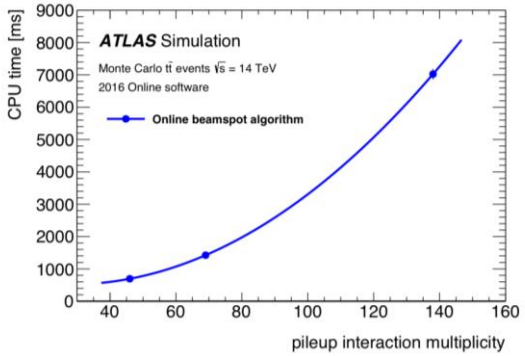
Track Finder | Event Filter Farm → Permanent Storage

[SIMPLIFIED] HL-LHC TRIGGER ARCHITECTURES

FTK would not scale at HL-LHC
- Factor of 20 input bandwidth increase
- Maximum processing rate would fall from 100kHz to 5kHz
- Alternatively reconstruction p_T threshold must be raised from 2 to 10 GeV/c

CPU time for software based track reconstruction **scales geometrically** with PU

HLT needs to be able to maintain trigger thresholds/rejection



Read
(< 7

Output Data
(7.5 kHz)

High Level Trigger Farm

Permanent Storage

t Level Trigger
S => L0
MS L1

L0 accept
(Latency < 10µs)

Tracker (ITK)

Calo Systems

Muon Systems

Trigger Data (40 MHz)

Readout Data
(1 MHz)

Cluster Finders

µ Track Finders

Global Trigger

Central Trigger Processor

FELIX Readout

DAQ Interfaces

Output Data
(10 kHz)

Track Finder

Event Filter Farm

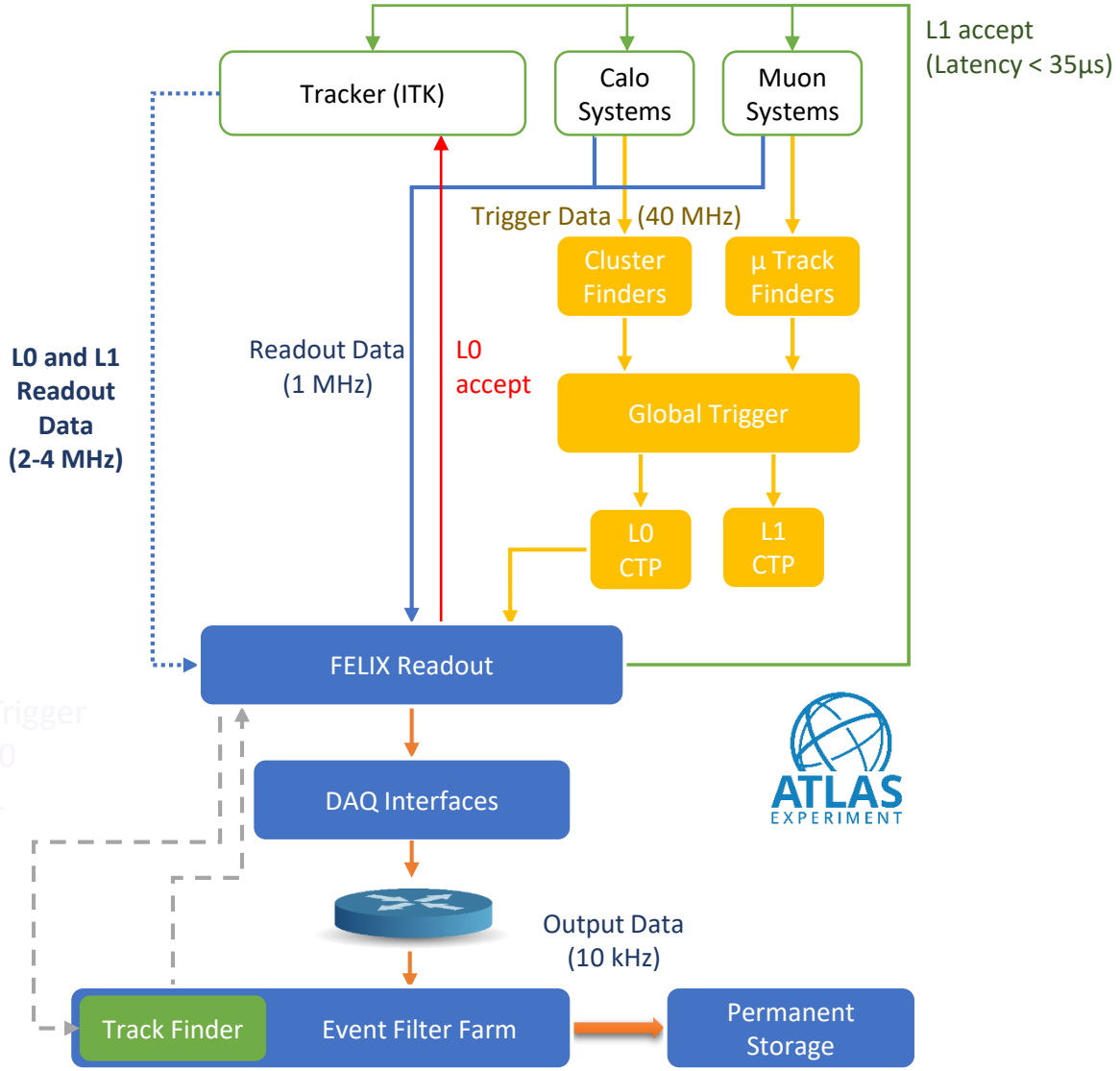
Permanent Storage

[SIMPLIFIED] HL-LHC TRIGGER ARCHITECTURES

These describe **baseline designs** for ATLAS and CMS

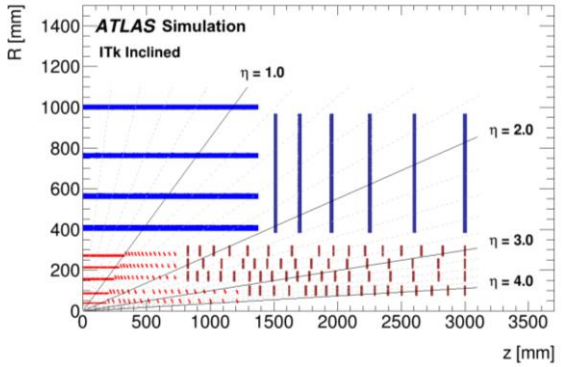
ATLAS also considers an ‘evolved’ design
- Two stage trigger (L0 & L1) before full data readout
- Based on Regions of Interest
- Regional Tracking at 4 or 2 MHz, vs 1 MHz

Not discussed further
- Track Finder implications minimal
- More details in backup slides




ATLAS
EXPERIMENT

TRACK FINDING FOR TRIGGERING



Data arrives at Level0 rate of 1 MHz
-> online Storage volume

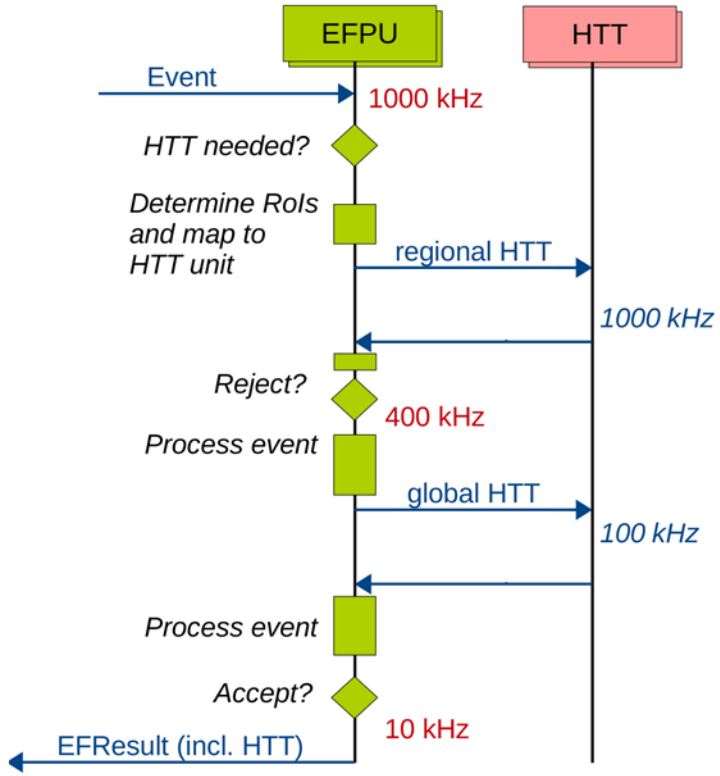
Event Filter (EF) computing farm consists of Processing Units

- Must reduce event rate from **1 MHz to 10 kHz** for offline storage
- Needs to **reconstruct tracks from Tracker**

Baseline solution -> **custom hardware-based Track Finding co-processor, known as the Hardware Track Trigger (HTT)**

- x10 reduction in CPU power required (30,000 -> 3,000 dual-socket servers)

TWO-STEP TRIGGERING WITH THE HTT



Step 1

Level0 information provides trigger path & **Regions of Interest** (RoIs)

Partial Tracker event data pulled from Storage (**10%** on average)
- Hits from **8 outermost layers** only

Regional tracking **(p_T > 2 GeV/c)** performed by HTT -> **rHTT**

EFPU makes use of rHTT tracks, in combination with L0 information
-> helps reduce rate to **400 kHz**

Step 2

Full hit data (strip + pixel) pulled from Storage if required

Full tracking **(p_T > 1 GeV/c)** performed by HTT -> **gHTT**

EFPU uses gHTT tracks, in combination with 'offline-like' analyses
-> helps reduce rate to requisite **10 kHz**

THE HTT SYSTEM

rHTT & gHTT implemented by the **same ATCA-based hardware system**

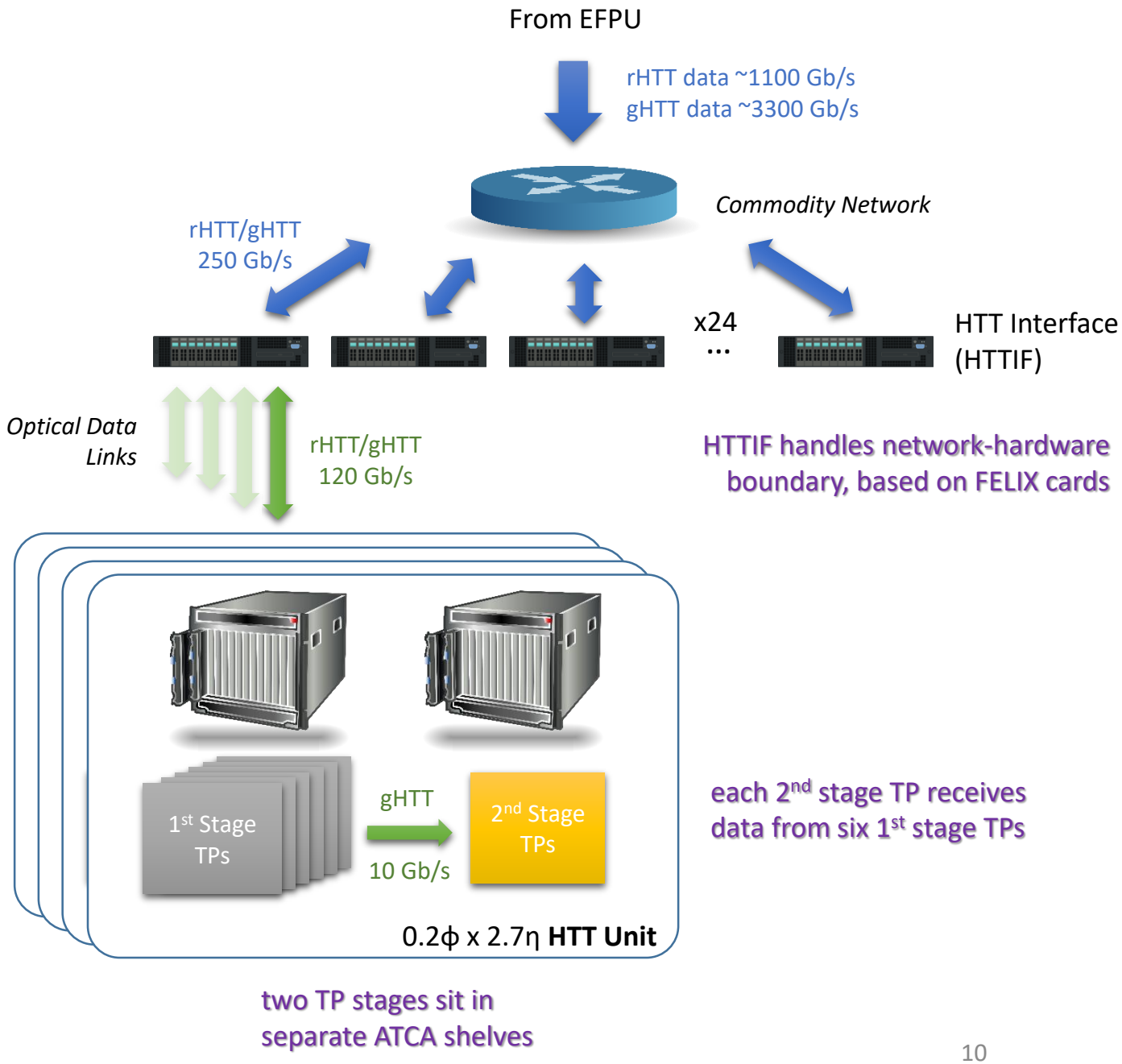
- Builds on the present FTK experience

672 Tracking Processor (TP) blades

- 56 ATCA shelves, full-mesh backplane

Each TP handles small region in ϕ - η space

- Pre-duplication of hit data performed mainly by Event Filter network via HTT Interface (HTTIF)

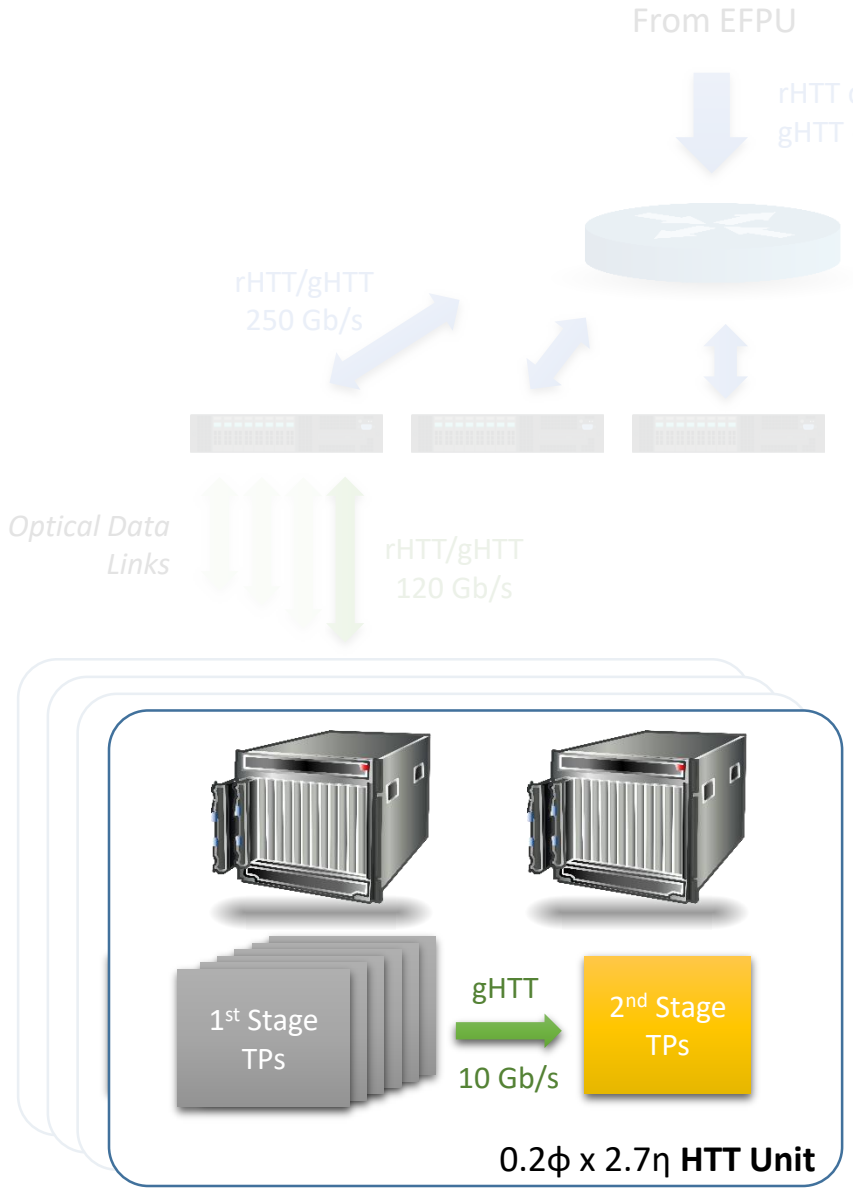


THE HTT SYSTEM

Two TP stages depending on trigger path:

rHTT -> 1st Stage TP only
- Tracks (p_T > 2GeV/c) passed back to EF

gHTT -> both 1st and 2nd Stage TPs
- Tracks (p_T > 1 GeV/c) passed 1st -> 2nd stage
- Tracks are refined with full TK hit data



THE HTT SYSTEM

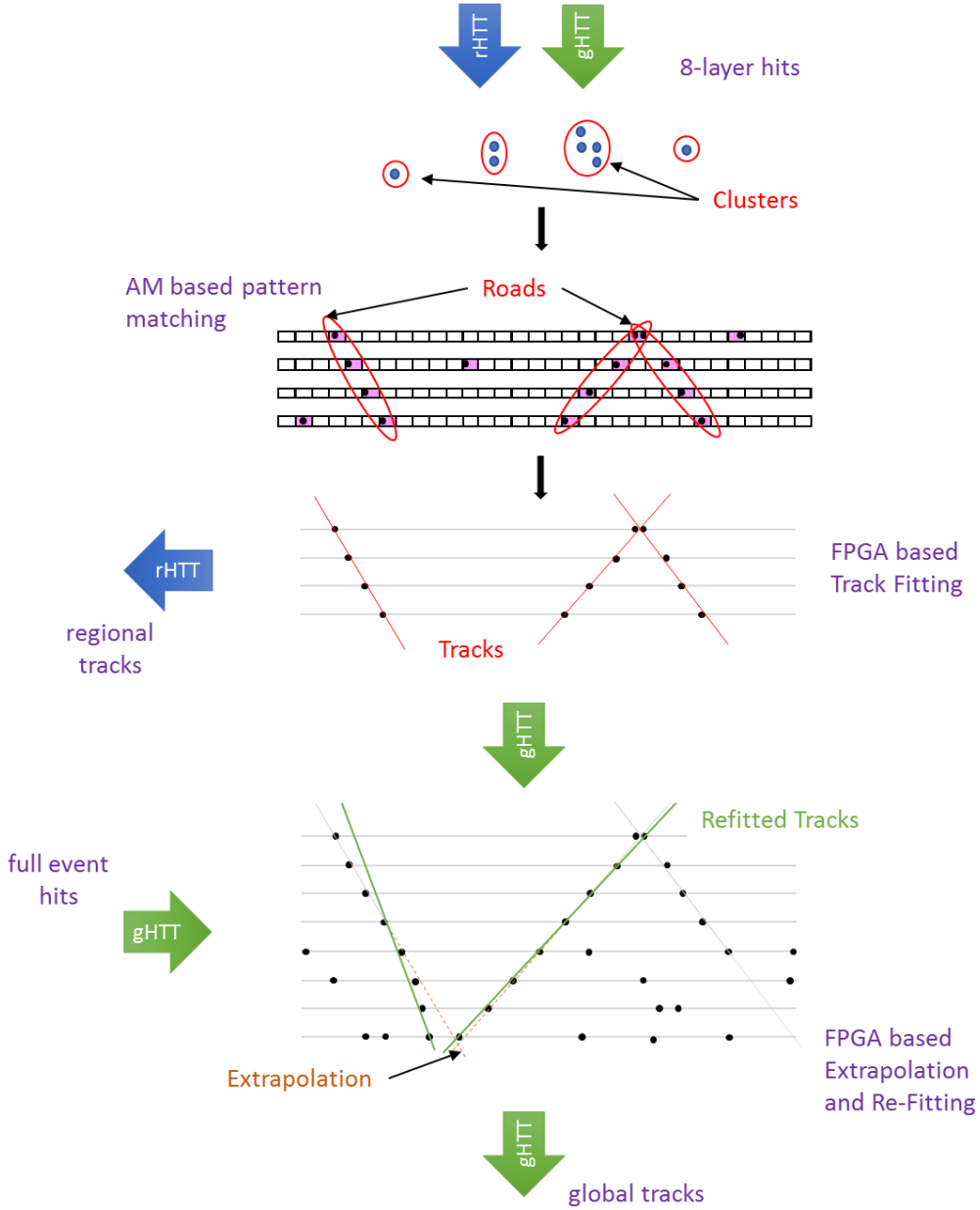
Single base card, differentiated by **Mezzanines**, implement 1st and 2nd stage TP types

1st Stage - **AMTP**

- Hit clustering and data organisation
- Uses **Associative Memories (AMs)** to match roads from clusters in up to 8 layers
- Track Fitting on roads performed in an FPGA
- **Pattern Recognition Mezzanines (PRM)**

2nd Stage - **SSTP**

- Second Stage (SS) TP receives 1st stage tracks/ clusters from AMTPs, plus full event data from HTTIF
- Hit clustering in new layers
- Track Extrapolation & Re-Fitting performed in FPGAs
- **Track Fitting Mezzanines (TFM)**

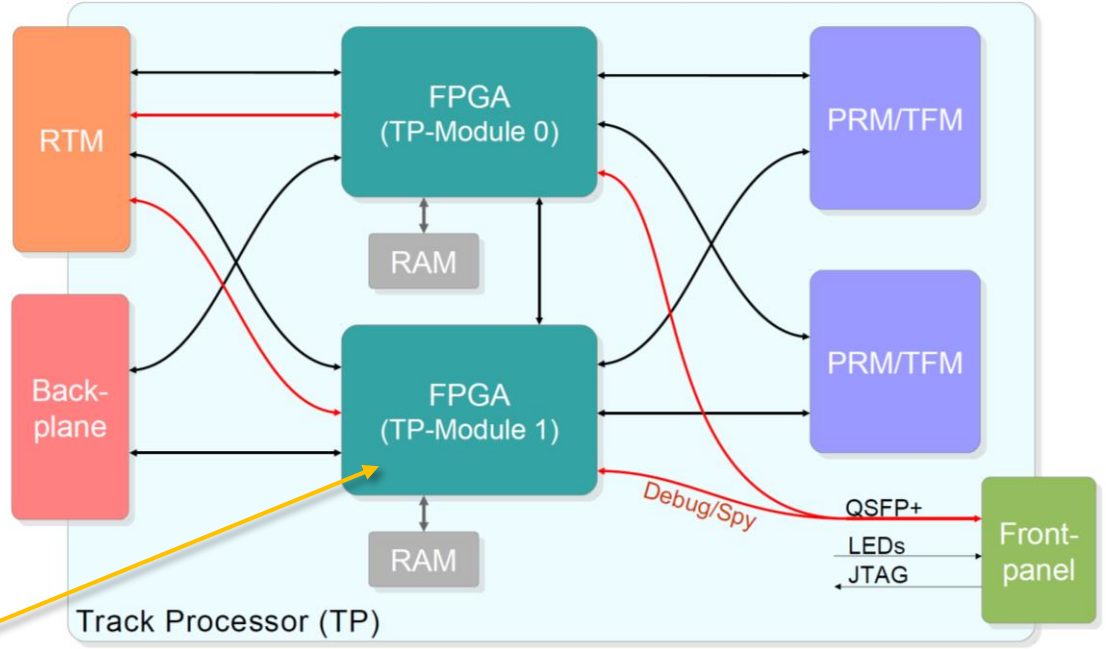


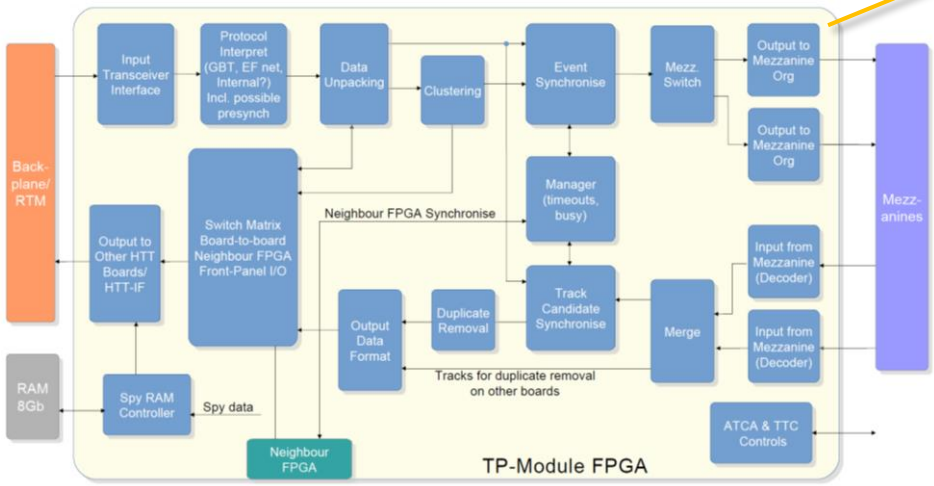
TP HARDWARE

TP ATCA blade includes:
- **2 FPGAs** (targeting Xilinx Kintex Ultrascale)
- 8Gb external RAM per FPGA
- **2 slots** for PRM or TFM mezzanines

Low demands in terms of I/O
- ~16 x 10 Gb/s optical link pairs
- RTM for optical interface to HTTIF, AMTP->SSTP link
- Backplane interface to other boards (full-mesh)

Power per blade estimated at ~300W





Complexity hidden in TP firmware
- Hit data clustering and organisation
- Event synchronisation and management
- Switch matrix for board-to-board communication
- Mezzanine readout/control & management
- Duplicate Track Removal

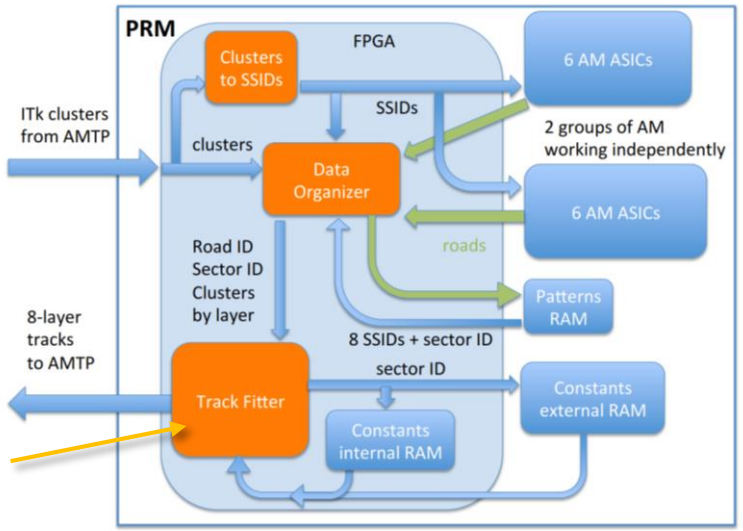
AMTP – PATTERN RECOGNITION MEZZANINE

1152 Production PRMs required for final system

- Each with **12 AM ASICs** (6 x 2)
- Processing FPGA; likely Kintex Ultrascale
- External RAM for storing patterns and constants

FPGA responsible for

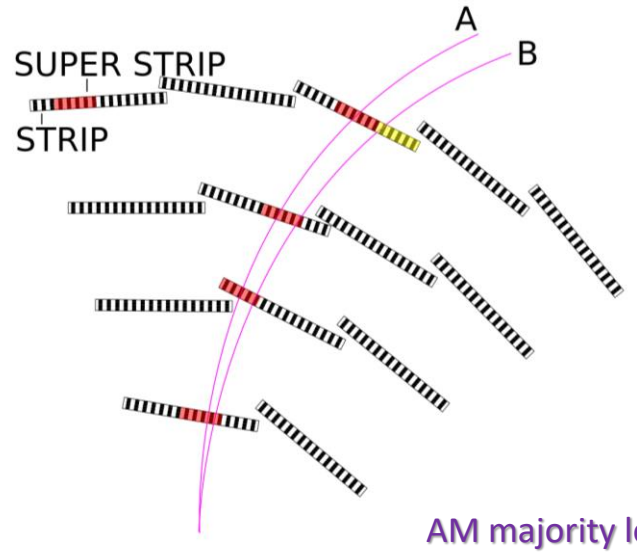
- Superstrip conversion
- Data Organisation to handle hits & roads
- Track Fitting (see TFM)
- AM configuration & monitoring



Linear Track Fitter based on TFM

pixels/strips binned into Super Strips

better performance & fewer patterns via variable resolution -> use of Don't Care (DC) bits in AM



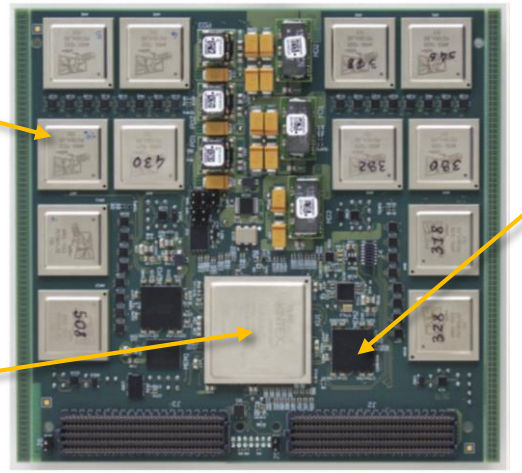
AM majority logic is configurable e.g. 7/8 or 6/8 matches to fire road

Production PRM target: 1.5M -> 4M patterns for p_T > 1 GeV/c

AMTP – PATTERN RECOGNITION MEZZANINE

Pre-prototype PRMs in hand [INFN]

- Based on AM06
- Otherwise meets expected requirements for production PRM



Demonstration of PRM06 on a Pulsar IIb [FTK Data Formatter board]

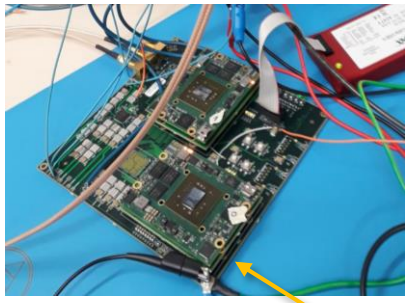
- As part of a CMS test stand
- Results documented here: https://cds.cern.ch/record/2272264/files/CMS-TDR-014.pdf

AMTP – PATTERN RECOGNITION MEZZANINE

AM07 evaluation ongoing - first results encouraging

- Verified lower power consumption per comparison **matches simulation**

Details on first results in dedicated ACES poster by G. Fedi

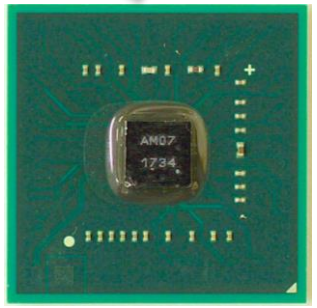


Area: 10 mm²
Memory depth: 16 kpatterns
400 bumps
4 independent cores
LVDS or LVCMOS interface
Working frequency: 200 MHz

AM06

65nm
128k patterns
2015
Used in ATLAS FTK

reduced power consumption (factor 1.7)
increased density (factor 2.9)



AM07

28nm
16k patterns
2016
Evaluation ASIC

AM08

28nm
16k patterns
Q4 2018
Final Technology Selections

AM09 requirements vs AM06

- Higher pattern density [384k patterns]
- Faster clock speed/bandwidth [250 MHz core/200 MHz readout]
- Low power [4W @60 MHz cluster rate]
- Better decoupling (vs. AM06)

AM09 for HTT

13,824 AMs in HTT!

28nm
384k patterns
12 AMs -> 4M patterns/PRM
2019/20
Production ASIC

SSTP – TRACK FINDING MEZZANINE

TFM must refit tracks from AMTP using full ITK hit information
- Uses a **linearised track fitter** (same as in PRM -> based on FTK)
- O(100) million pre-calculated coefficients per mezzanine

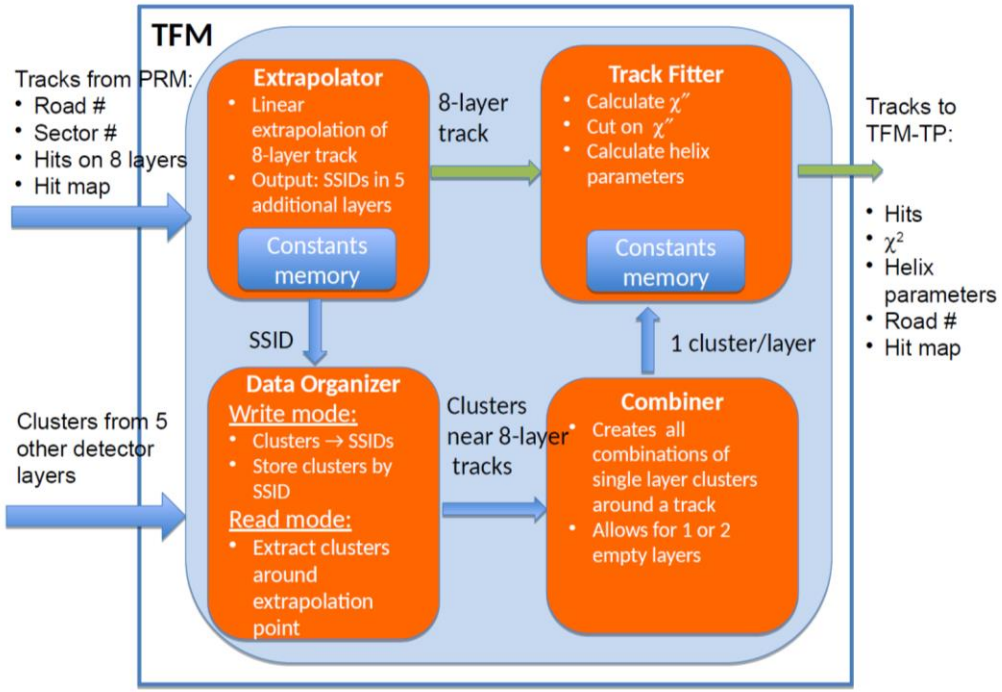
Likely design specifications:
- 2 Kintex Ultrascale FPGAs, 11 x 10 Gb/s optical inputs
- DDR3 for storing constants
- 192 needed in final system

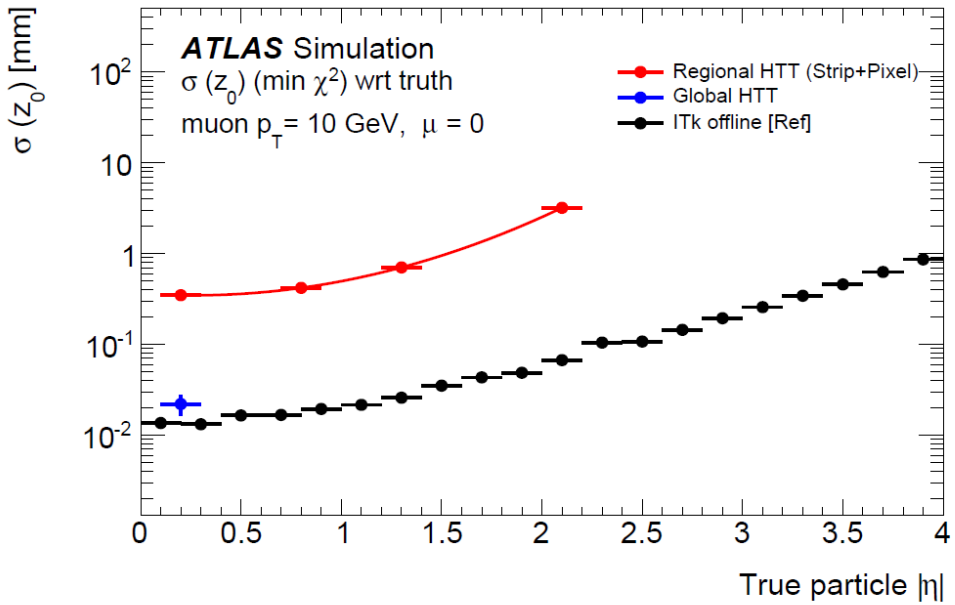
linear constants

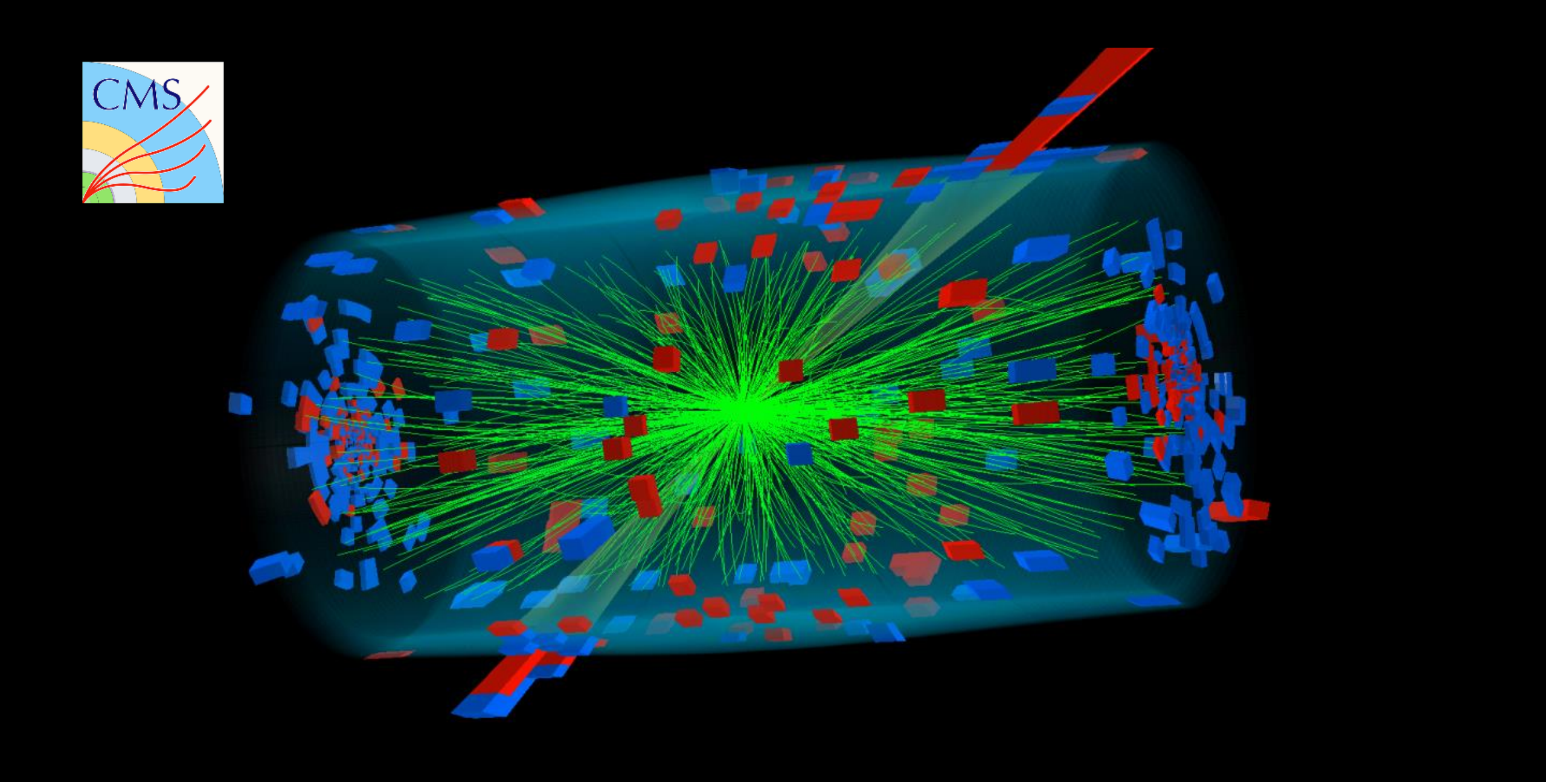
$$p_i = \sum_{j=1\ldots N} C_{ij}\, x_j + q_i$$

5 helix parameters (p_T, η,ϕ,z_0,d_0)

full resolution hit positions



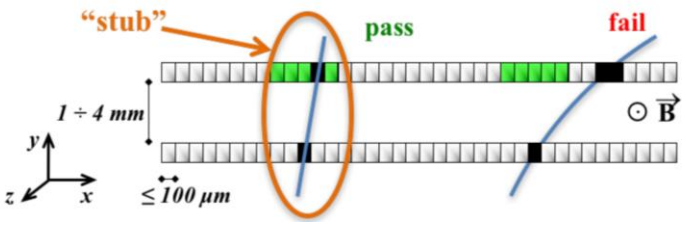



CMS

A TRACKER DESIGNED FOR TRIGGERING

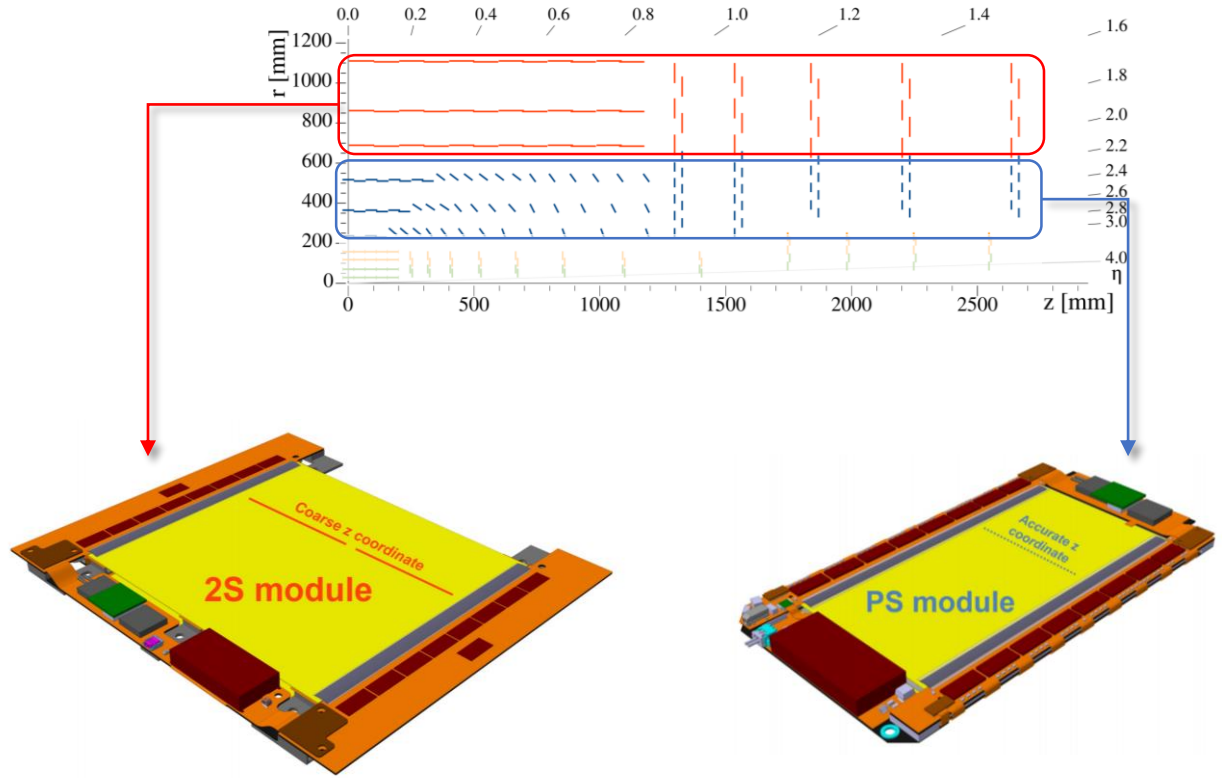
Tracker provides <u>limited</u> hit data at **full 40 MHz crossing rate**

- No inner pixel tracker hits (Inner Tracker)
- Hits from outer 6 tracking 'layers' only (Outer Tracker)
- Only hits compatible with tracks p_T >~ 2 GeV/c read out



p_T discrimination provided by use of special modules

- Pairs of closely spaced silicon sensors, separated 1-4 mm
- Signals from each sensor are correlated
- Only hit pairs compatible with p_T threshold **("stubs") are forwarded off-detector**
- Factor ~10 data reduction



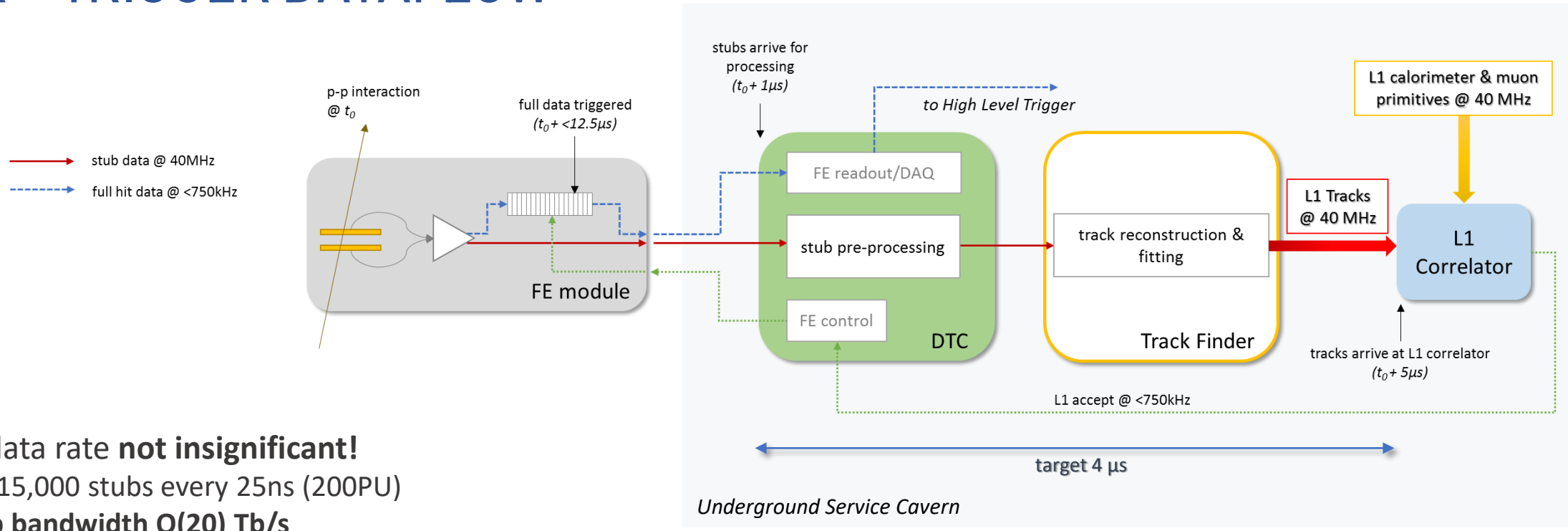
<u>Strip-Strip (2S) modules</u>

- Both sensors: 2×1016 strips
  5 cm long, 90 µm

<u>Pixel-Strip (PS) modules</u>

- Top sensor: 2×960 strips
  2.4 cm long, 100 µm
- Bottom sensor: 32×960 pixels
  1.5 mm × 100 µm

TRACKER→TRIGGER DATAFLOW



Even still, data rate **not insignificant!**

- Average 15,000 stubs every 25ns (200PU)
  -> **Stub bandwidth O(20) Tb/s**

L1 hardware trigger reduces event rate from **40 MHz to <750 kHz** using calorimeter, muon and tracker primitives

- **TK primitives are all tracks (p_T > 2-3 GeV/c), from Outer Tracker**
- L1 accept triggers all front end buffers to read out to DAQ -> HLT farm

FE L1 latency buffers (including TK) limited to 12.5 μs

Transmission of stubs to BE electronics	1 μs
Correlation of trigger primitives (inc. tracks)	3.5 μs
Broadcast of L1 accept to FE buffers	1 μs
Safety Margin	3 μs

-> Track finding from stubs must be performed in 4 μs

TRACK FINDER ARCHITECTURE

Two stages of data processing
- **DAQ, Trigger and Control (DTC) layer**
- **Track Finding Processing (TFP) layer**
- **All-FPGA** processing system
- **ATCA**; CMS standard backplane (dual-star)

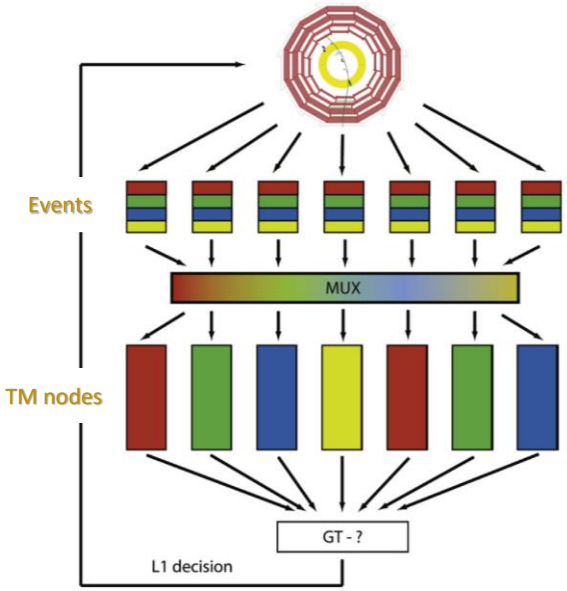
Outer Tracker cabled into nonants
- Use of **time-multiplexing** to increase parallelisation

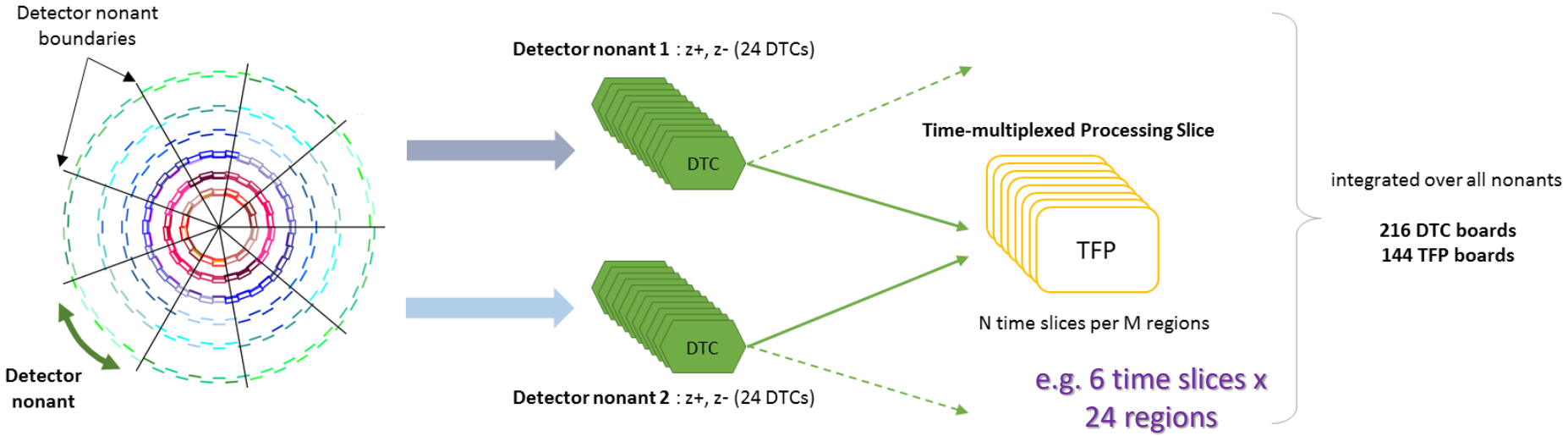
Time-multiplexing directs data from multiple sources to a single processing node
- Employed in CMS L1 calorimeter trigger
- 1 event per processing node
- Fixed passive optical patch panel as MUX

Helps minimise complexities due to physical segmentation and boundary handling

Processors are independent entities – simplifies commissioning and operation

Spare nodes available for redundancy





TRACK FINDER ARCHITECTURE – DTC

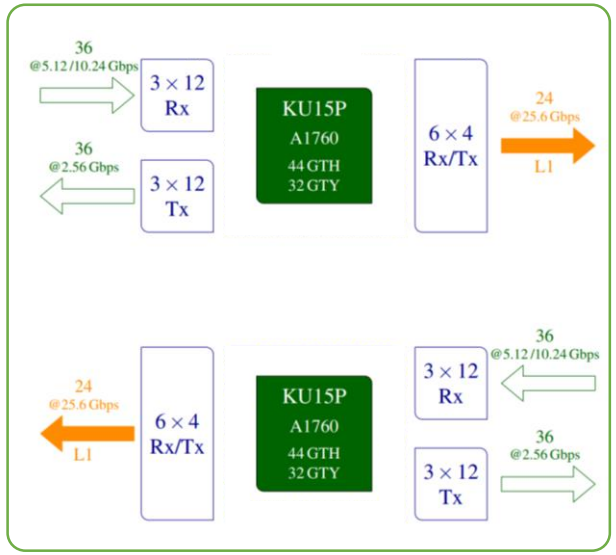
Two stages of data processing
- **DAQ, Trigger and Control (DTC) layer**
- **Track Finding Processing (TFP) layer**
- **All-FPGA** processing system
- **ATCA**; CMS standard backplane (dual-star)

Outer Tracker cabled into nonants
- Use of **time-multiplexing** to increase parallelisation

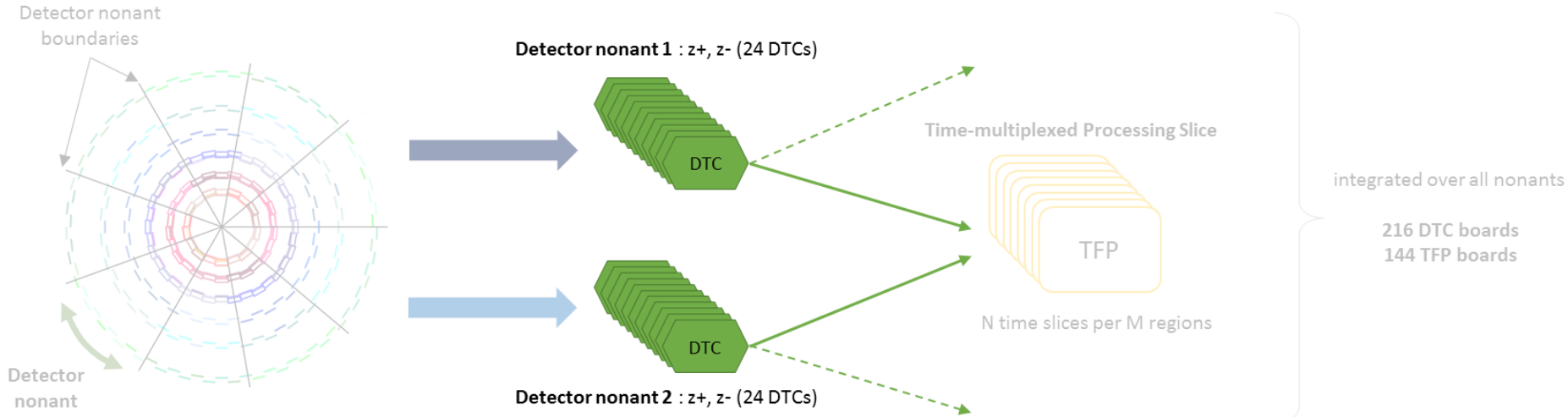
DTC card must handle
- **<=72 modules (5G/10G lpGBT opto-links)**
- Control/Readout for each module
- Direct L1 stream to central DAQ (16G/25G)
- Direct stub stream to TFPs (16G/25G)

Stub pre-processing includes:
- Local->Global look up, position calibration
- Sorting and pre-duplication
- Time-multiplexing

-> 216 DTC boards, 18 shelves, 1 rack/nonant



~600 Gb/s processing card



TRACK FINDER ARCHITECTURE – TFP

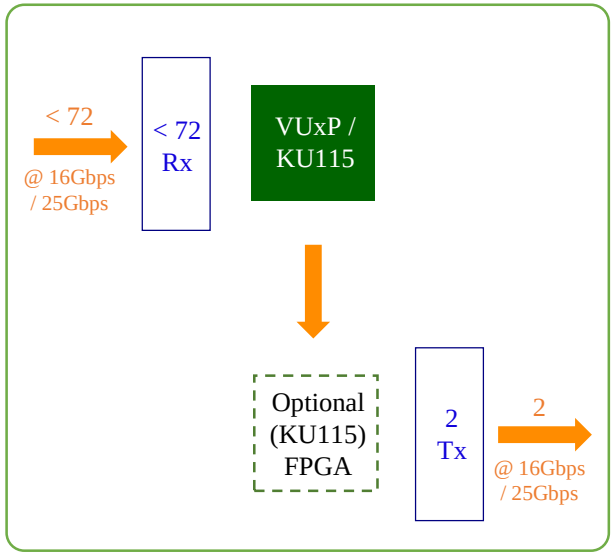
Two stages of data processing
- **DAQ, Trigger and Control (DTC) layer**
- **Track Finding Processing (TFP) layer**
- **All-FPGA** processing system
- **ATCA**; CMS standard backplane (dual-star)

Outer Tracker cabled into nonants
- Use of **time-multiplexing** to increase parallelisation

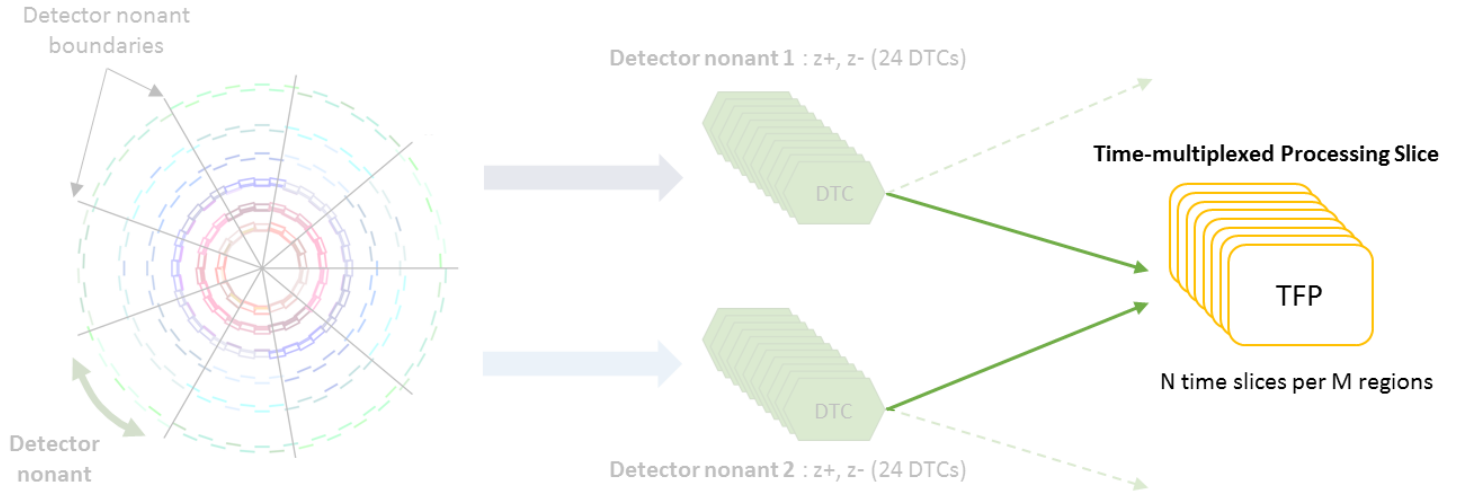
TFP card must handle
- **Up to 72 DTCs (16G/25G optical links)**
- Track Finding from stubs
- Track Fitting
- Transmission to L1 Correlator Trigger

High bandwidth processing card
- Rate to L1 Correlator much lower < 30Gb/s

-> 144 TF boards, 12-18 shelves



~1 Tb/s processing card

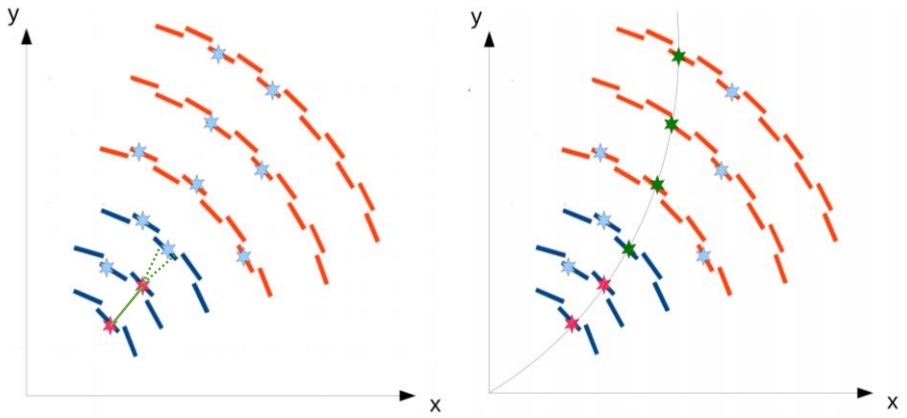


overall compact system
(30-36 shelves)
integrating readout &
trigger

TRACK FINDING ALGORITHMS

Two principle algorithms for reconstructing tracks
- Plus a number of hybrids, variations and options

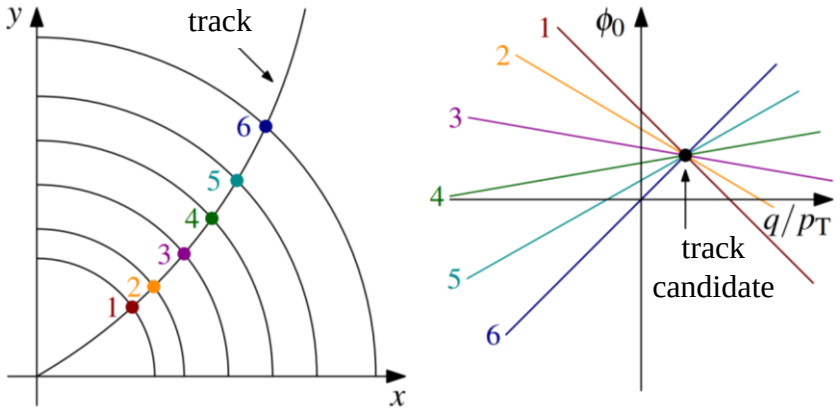
Tracklet Approach



- Combinatorial approach using **pairs of stubs as seeds**
- **Extrapolation** to other layers -> hit matching
- **Linearised χ^2 fit** on candidates
- Uses **full resolution stubs** at earliest stage of processing

- N time-slices x M regions -> 6 x 24 , 9 x 18

Hough Transform + Kalman Filter Approach



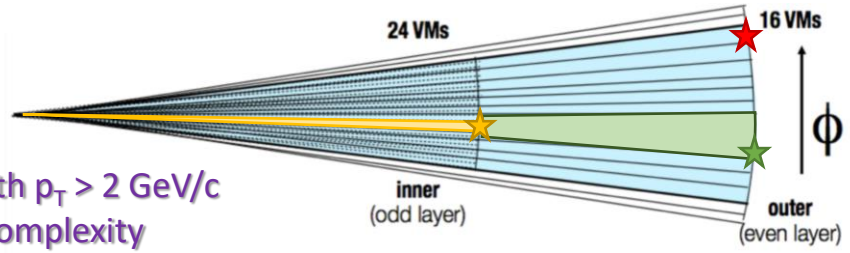
- Uses a **Hough Transform** to detect coarse candidates
- Candidates are filtered and fitted in a single subsequent step using a **Kalman Filter**
- Combinatorial problem pushed to latter stages of processing

- N time-slices x M regions -> 18 x 9

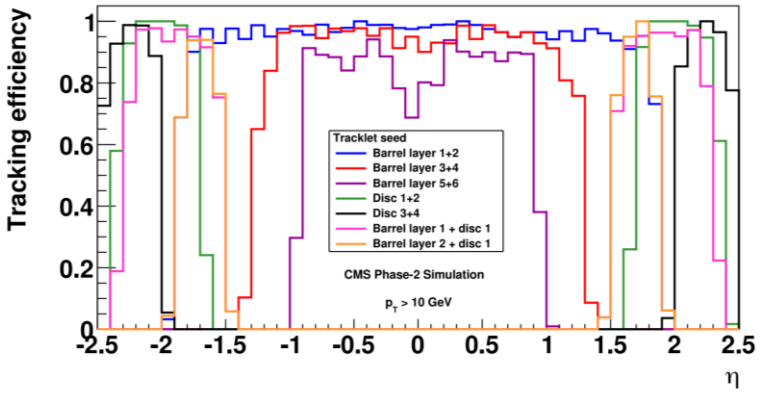
TRACKLET ALGORITHM

Seeding Step

- Seeds are generated from **pairs of layers**
- Layer pairs selected to give full coverage in η, including **redundancy**
- Seeding step **massively parallelised** by internal geometrical partitioning & stub organisation (Virtual Modules - VMs)

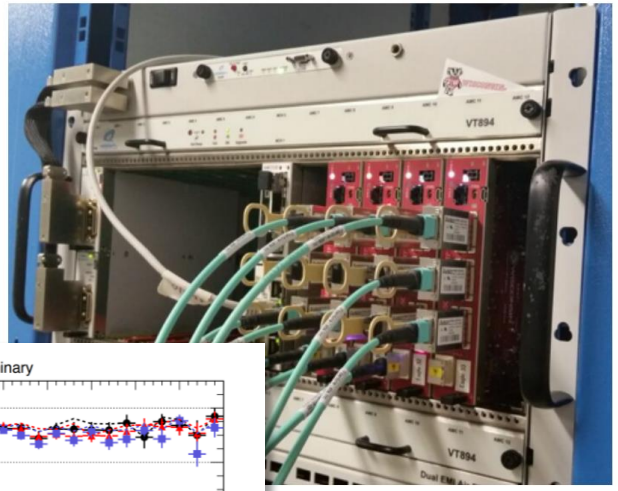


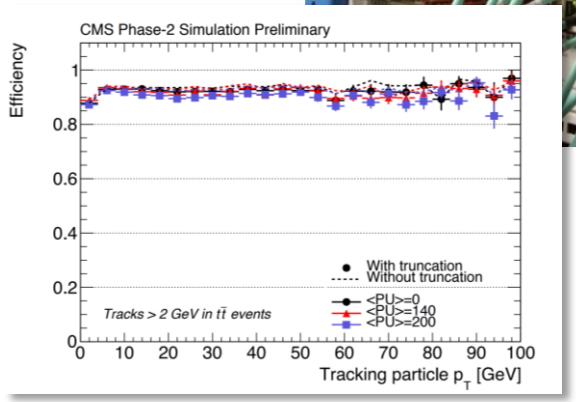
only ~30% of VM pairs compatible with p_T > 2 GeV/c – reduces combinatorics & f/w complexity



Tracklet Projection & Fitting Steps

- Tracklet seeds are projected to other layers -> matched with stubs in VMs
- **Residuals calculated** -> update track parameters
- Track re-fitted at each step using linearised χ^2 fit -> constants tabulated in LUTs
- Duplicate removal step needed due to multiple seeding layer pairs





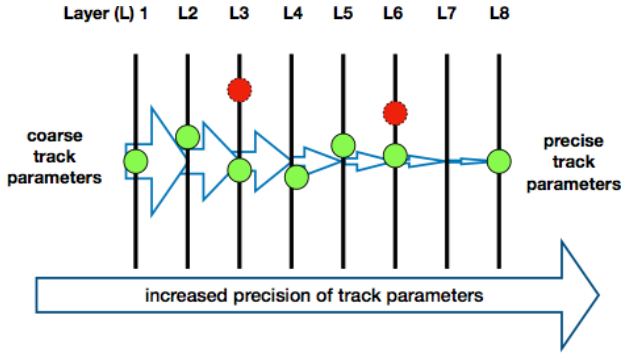
Demonstration in hardware, verified using emulation software

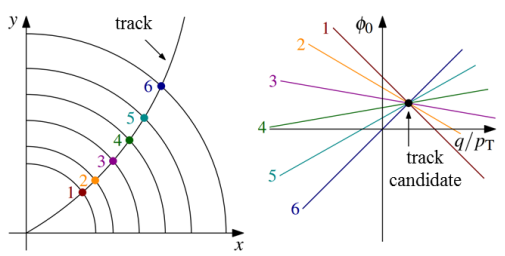
- MC stubs from PU 0->200 samples passed through slice demonstrator
- Latency verified to be **3.3 µs**, agrees with latency model

HOUGH TRANSFORM+KALMAN FILTER ALGORITHM

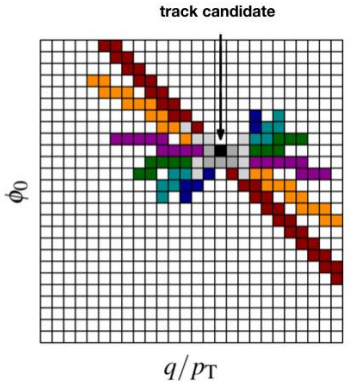
Hough Transform (HT)

- Search for primary tracks in r-ϕ using **parameterisation (q/p_T, ϕ_0)**
- Stub positions correspond to **straight lines** in Hough Space
- Where **4+ lines intersect** -> track candidate
- **Internally parallelised** into 36 independent η-ϕ regions (HTP), 1 HT array per region





each HT is discretised as a 32 x 64 cell array

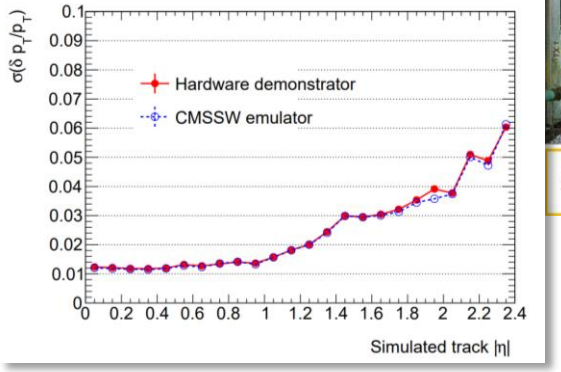


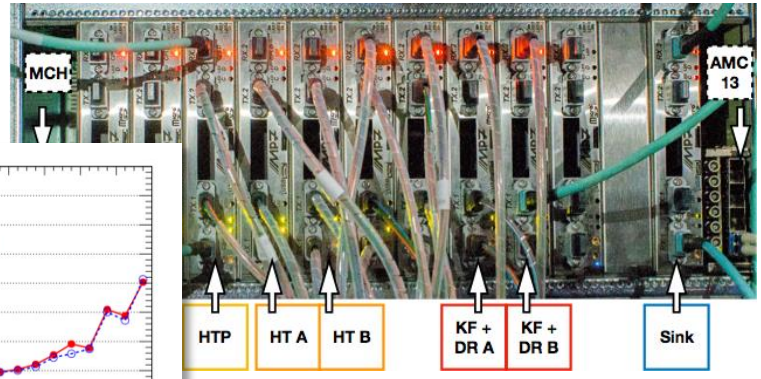
Combinatorial Kalman Filter (KF) & Duplicate Removal (DR)

- Iterative algorithm as used in offline reconstruction, seeded by HT
- Able to fit and select different combinations of stubs on track candidates -> **best combination selected on χ^2**
- Only constants are hit errors (and scattering angle vs p_T if required)
- Simple DR block removes HT candidates with similar helix parameters after fit

Demonstration in hardware and emulator

- Many samples from PU 0->200 passed through MP7-based slice demonstrator
- Latency verified to be **3.5 µs**



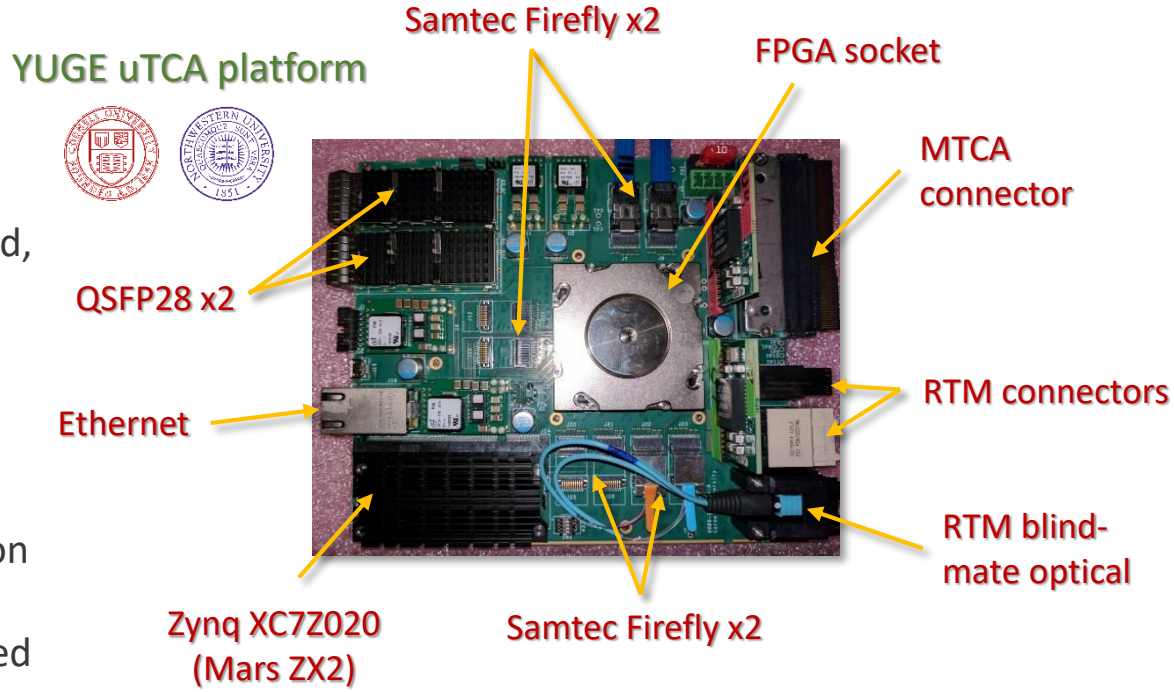


R&D ACTIVITIES

- **Evaluation of Xilinx Ultrascale/Ultrascale+ FPGAs**

- **Testing and evaluation of data transmission at 16->25Gb/s**
  - Trialling different optical transceivers and modules (mid-board, edge-mount..), attempting to follow industry
  - PCB signal integrity & design for high speed
  - Protocols & latency (important for L1)

- **Evaluation of embedded CPU options, kick-start s/w projects**
  - Required for slow control plus on-board processing/calibration
  - Zynq SoC; integrated FPGA/ARM control processor
  - Computer On Module (COM) express; pluggable module based on quad Intel Atom



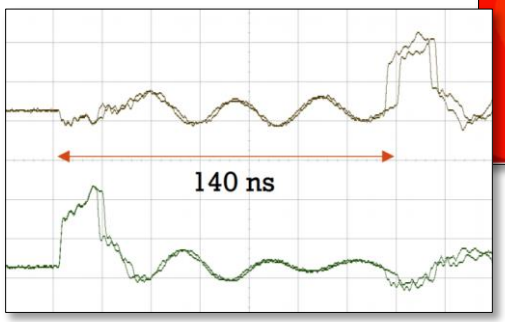
Mars ZX2

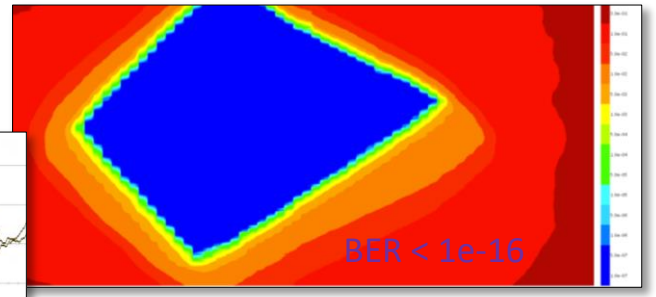
84mm x 55mm

COM express T10 mini

[Dedicated ACES talk by R. Spiwoks]

Aurora 64b/66b demonstration





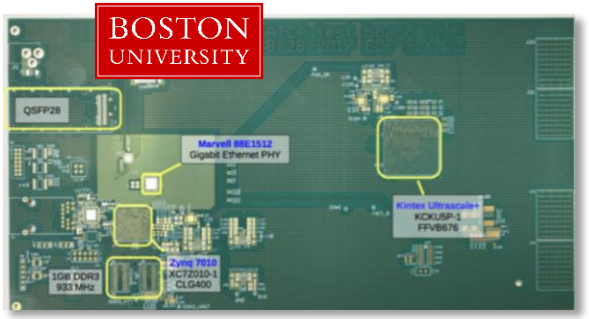
electrical loopback & board to board via Firefly twinax @ 28Gb/s
optical loopback at 14Gb/s verifed

R&D ACTIVITIES

- **ATCA infrastructure**
  - Systematic thermal studies - significant concerns about air x-section and impact on opto-lifetime
  - Backplane signal integrity -> important for DAQ/timing

- **Use of interposer technology**
  - Flexibility (e.g. FPGA)
  - Mitigate losses/costs due to yield issues
  - Modularity; separate complex and simpler parts of board design

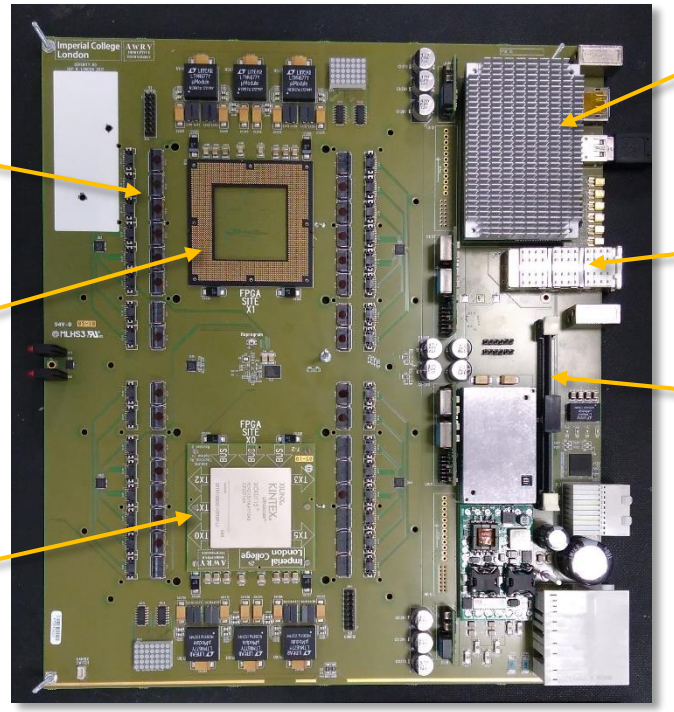
- **PCB design practices, stackup and material**
  - Build up relationship with manufacturers



ATCA Test Blade

[ATCA developments discussed in dedicated ACES talk by E. Hazen]

Serenity ATCA Platform



Samtec Firefly x16 RX/TX pairs

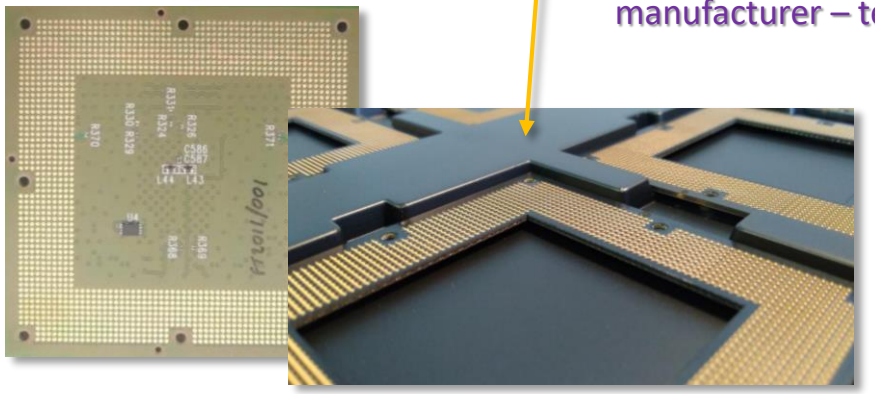
Samtec Z-RAY interposer

COM express

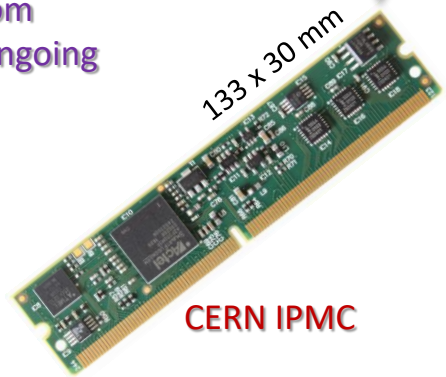
QSFP28

CERN IPMC

FPGA KU115 daughtercard on interposer



recently returned from manufacturer – testing ongoing



CERN IPMC

SUMMARY & OUTLOOK

Both ATLAS and CMS must include tracking information in their trigger decision logic at HL-LHC
- CMS at L1 (first level)
- ATLAS at the EF (high level)

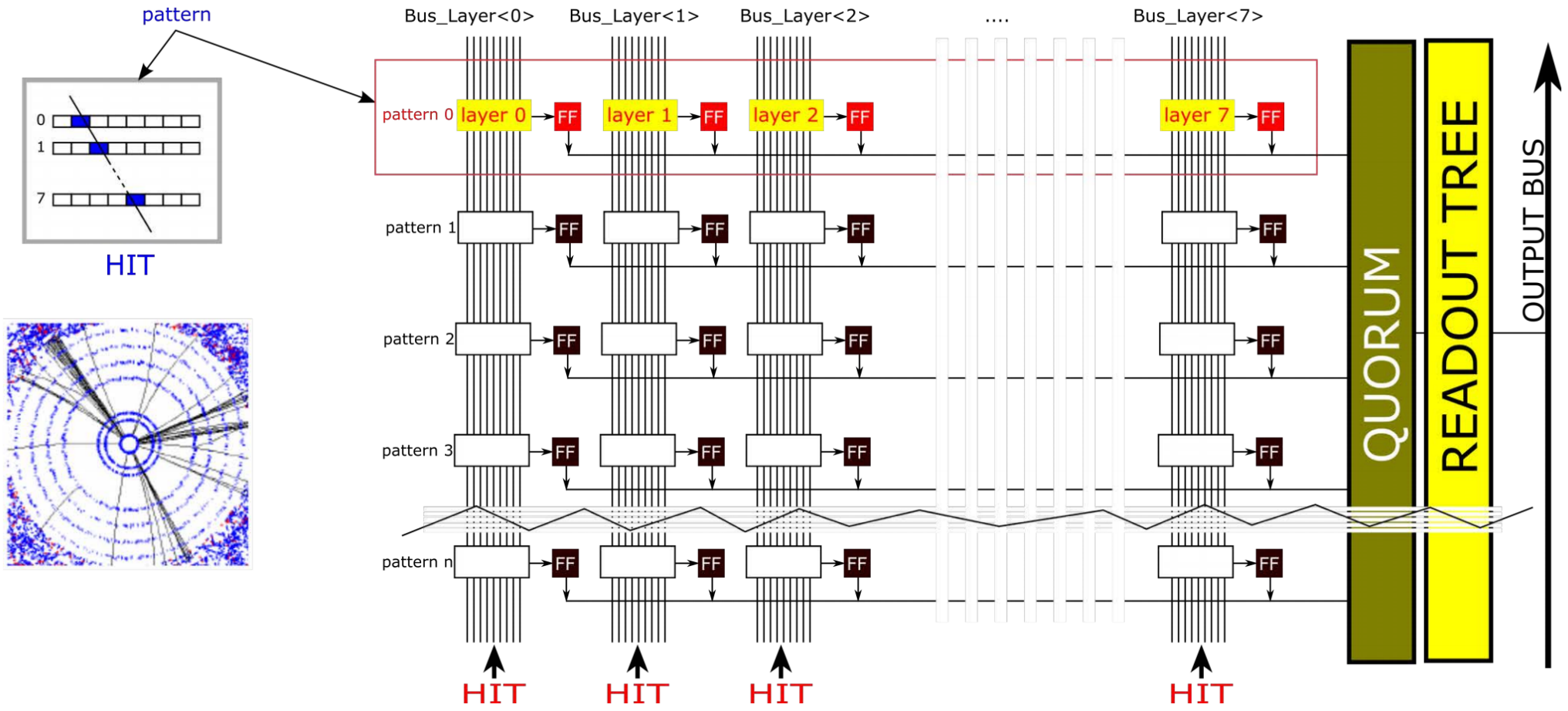
ATLAS track finder an evolution of the FTK
- More powerful Associative Memory (AM09), key technologies currently under test
- Acts as a co-processor farm, independent nodes receiving data from the network
- More homogeneous hardware solution, common components, interfaces and technology
- Baseline configuration defined Q2 ‘19, prototype demonstrators in 2021

CMS track finder is a new experience
- Stringent requirements on latency and performance to keep L1A rate below 750 kHz
- Leveraging advances in FPGA technology, and industry where possible
- Pre-prototype slice demonstrators in 2016 -> prototype demonstrators in 2019
- Final system specification Q4 ‘19

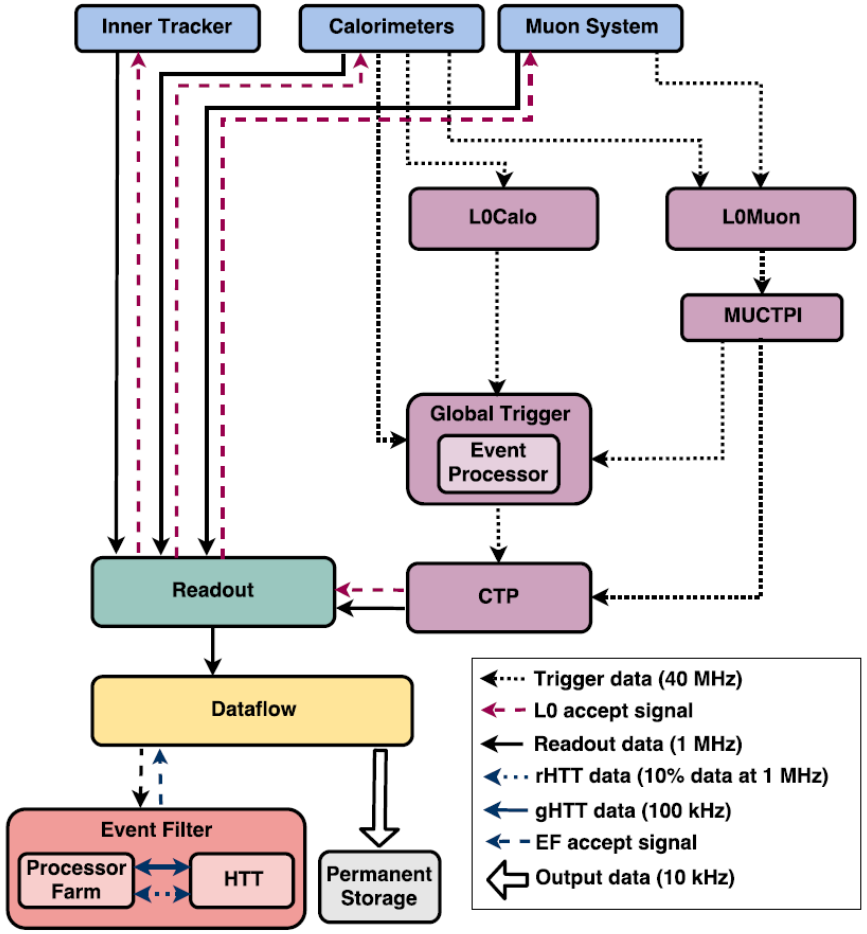
Both systems based on ATCA
- Blades will dissipate significant heat
- Requires careful board design to minimise hotspots and protect optics

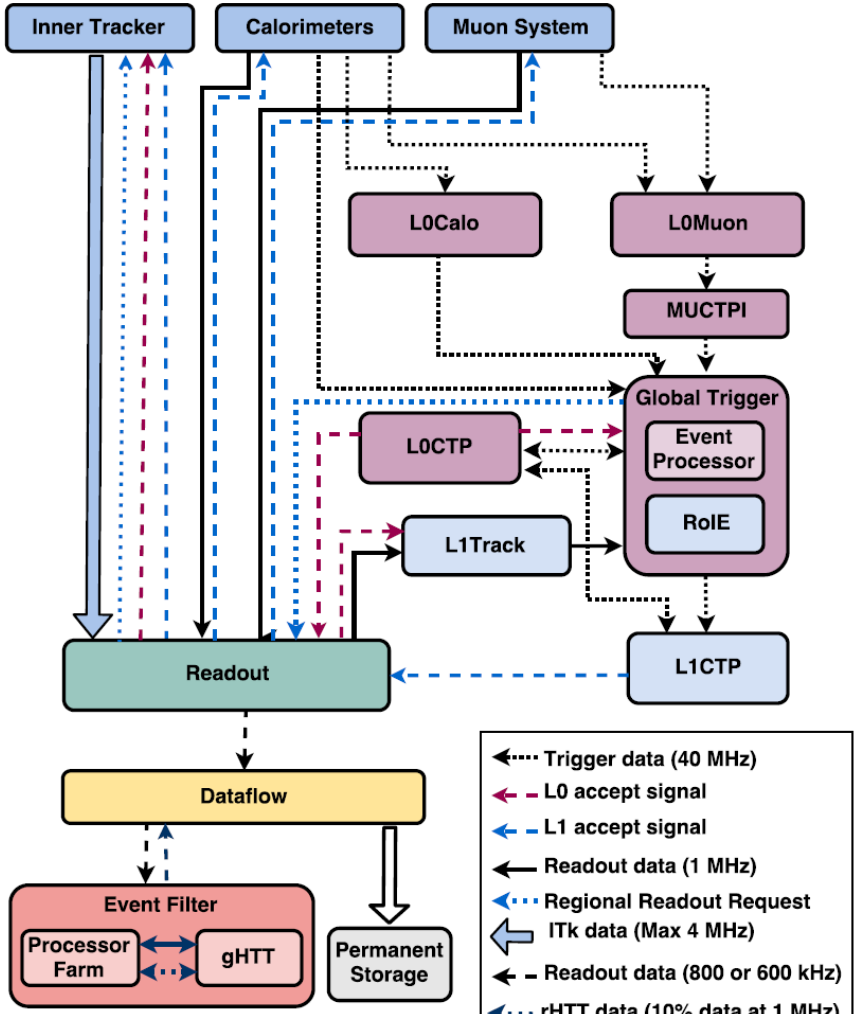
lpGBT/VL+ common to detector readout.
Other common projects? IPMC, embedded systems?

AM PRINCIPLE



EVOLVED OPTION

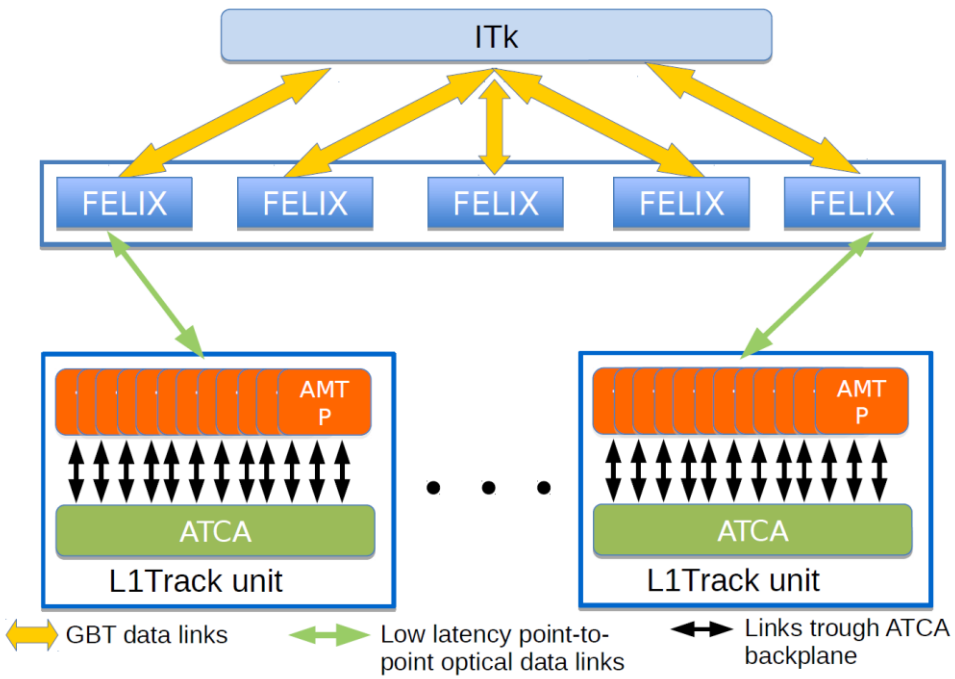


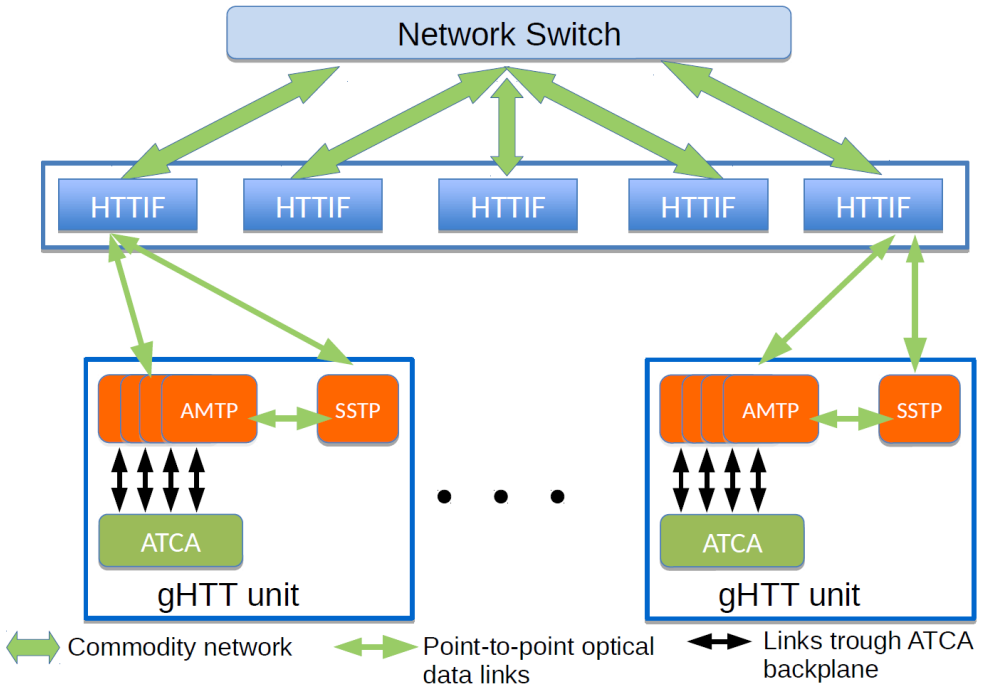


	4 MHz L0A Scheme	2 MHz L0A Scheme
Maximum Level-1 Latency	30 μs	35 μs
Fraction of Regional Data	10%	10%
Effective Regional Readout Rate	400 kHz	200 kHz
Full ITk Detector Readout Rate	600 kHz	800 kHz
Total Data Rate	1 MHz	1 MHz

EVOLVED OPTION

Trigger	Latency requirement	Level-0 rate [MHz]	Trigger threshold [GeV]
rHTT	No	1	2
L1Track	6.0 μs	2–4	4





FTK COMPARISON

FTK in Run II / Run III

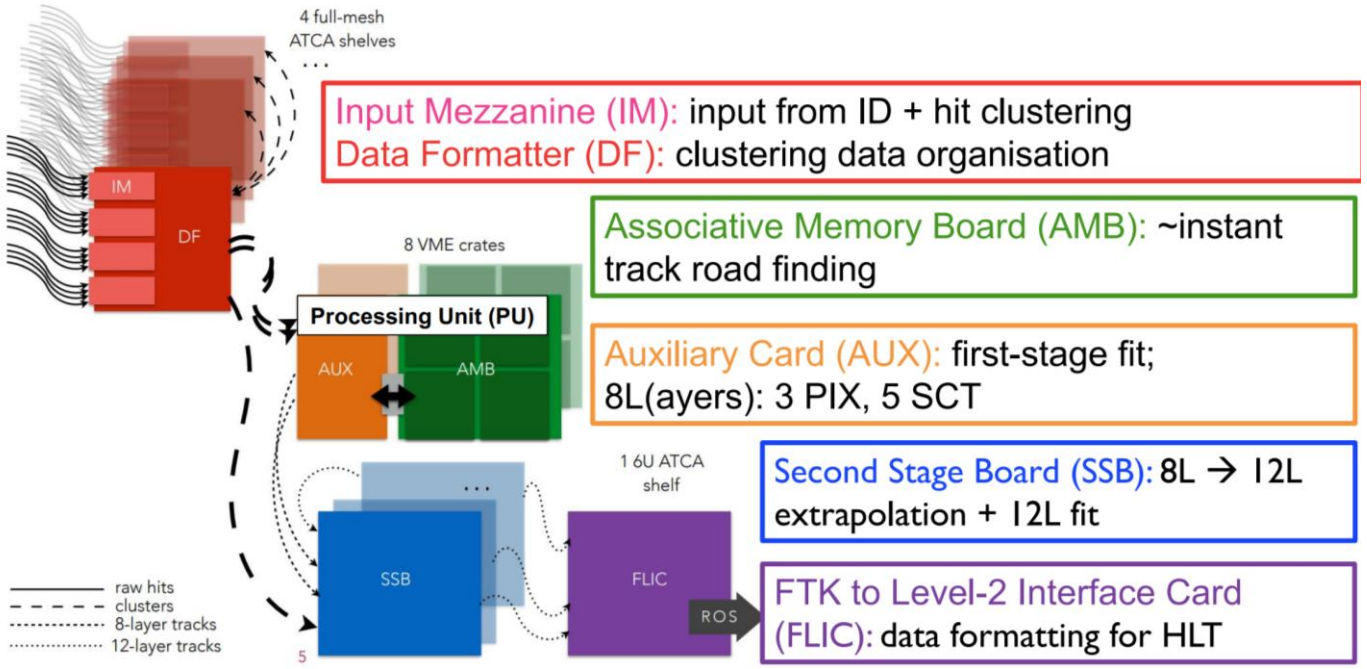
Provides full 100kHz tracking as an input to the HLT

- Tracking in $|\eta| < 2.5$
- pT > 1GeV
- Tracking volume is divided in 64 $\eta - \phi$ towers
- All events selected by Level-1 are processed

Very complex hardware system

- About 150 boards of 6 different kinds, each with specialized functions
- Heterogeneous connections between boards (fiber, backplane, connectors, and with different bandwidth / width)

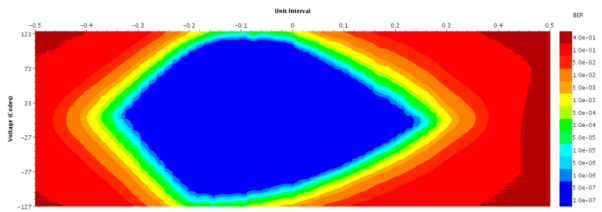
Regional Tracking is done also in CPU software



25Gb/s TESTING

Optimization

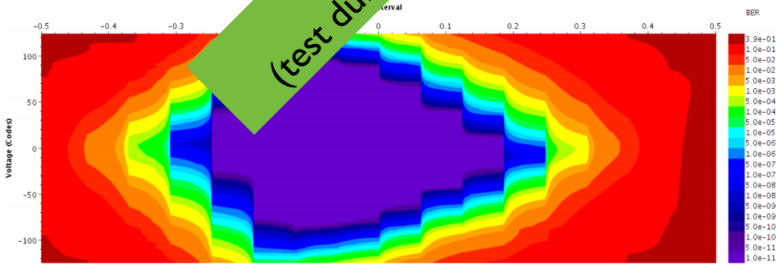
- Found a range of optimal settings
  - Within the range open areas vary by +/- 5% (tolerance)
- Settings closer to IBERT default are chosen
  - RX Termination = 400 mV, TX Pre/Post Cursor Emphases = 2.21/0.00 (dB), TX Driver Swing Control = 924 mV
- Eye diagram reproduced with maximum amount of HORZ/ VERT_OFFSET increments
- Wide open area observed



Results

- BER range = [0,1e-12]
- Wide open area observed
- Expected (20%) shrinking in ea
  - N.B: chosen a medium amount of _OFFSET, otherwise ~20 days needed

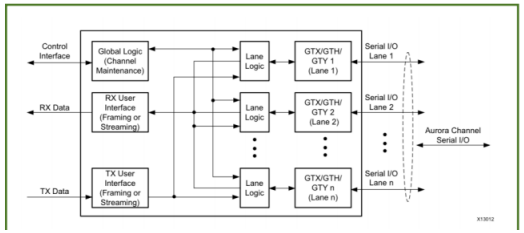
BER < 1e-16
(test duration ~ 4 days)

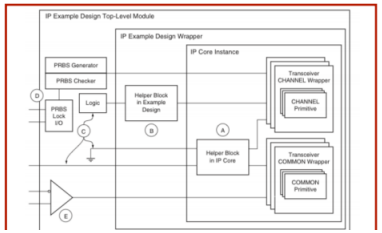


25Gb/s TESTING

How to start

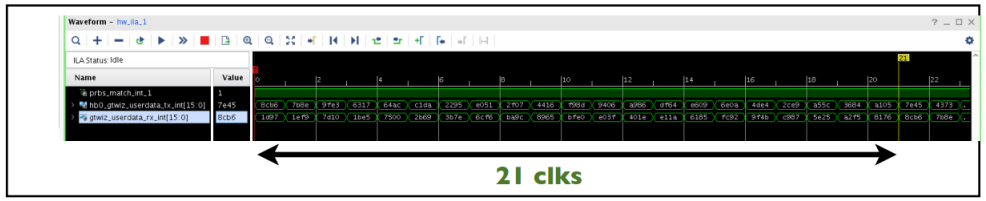
1. Aurora Example Design (link)
2. Ultrascale FPGA Transceiver Wizard Example Design (link)





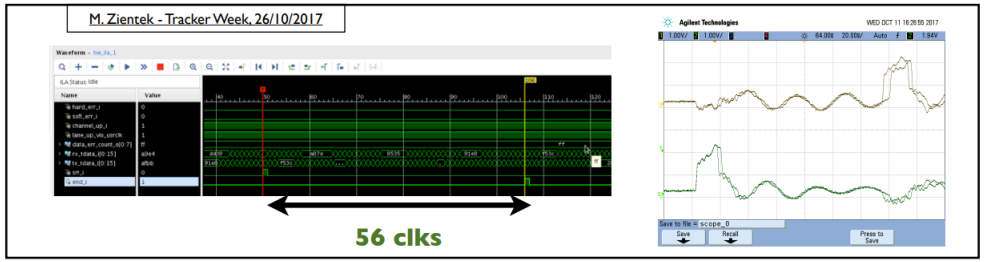
Latency Measurement (via 2.)

- *Line rate = 25 Gbps, clk frequency = 390 MHz*
- TX: 64b packets from the wrapper moved to the channel, ready to be sent over GTY transceivers
- packets looped back via FIREFLY connector
- RX: Packets received over GTY transceivers, moved to the user interface
- Watched TX and RX: **latency ~ 50 ns**
  - extrapolated from 21 clks @ 390 MHz
  - smaller latency than 1. because of the reduced overhead logic
  - smaller that what expected (~70-80 ns) from previous studies (link)...investigating



Latency Measurement (via 1.)

- *Line rate = 25 Gbps, clk frequency = 390 MHz*
- TX data: 64b packets from the user interface moved to the channel, ready to be sent over GTY transceivers
- packets looped back via FIREFLY connector
- RX data: Packets received over GTY transceivers, moved to the user interface
- Watched TX and RX: **latency ~ 130 ns**
  - from specs (GTH) expected ~55 clks @ 390 MHz = 140 ns



- Imported **IBERT optimal settings** into the design
- No errors observed for ~ 5 days
- **BER < 1e-16**

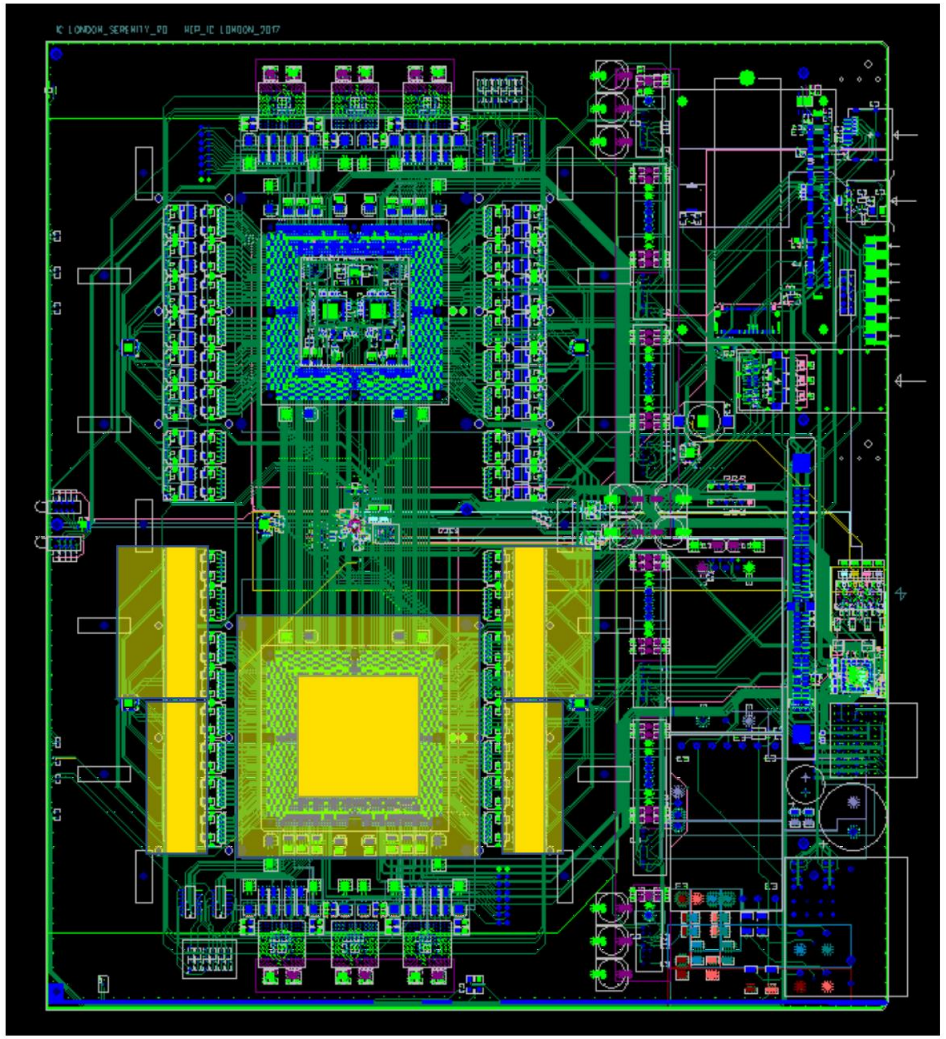
THERMAL STUDIES

Power

- FPGAs
- Dual Kintex KU115: **45W – 68W**
- Single VU9P: **90W – 130W**

- Optics Module Power Dissipation
- 12x 16G, 1.7W nom, 3.6W max
- 4x 28G, 5W nom

- Dual FPGA Tracker Board
  - x6 16G and x6 (perhaps x3) 28G **per FPGA**
  - Assume **2x20W** over 2x6 modules **per FPGA**
- 96 Ch Trigger Board
  - x24 28G for single FPGA
  - Assume **2x60W** over 2x12 modules
- 64 Ch Trigger Board
  - x16 28G for single FPGA
  - Assume **2x40W** over 2x8 modules



THERMAL STUDIES

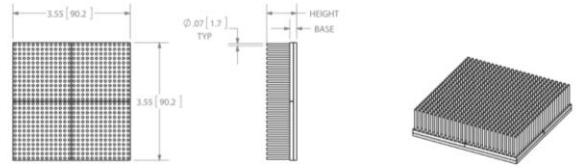
FPGA Heatsink

- Types
  - Die casting, Extrusion, Stacked fin, Folded fin, Skived fin
- Types
  - Moderate, Sparse
  - Cu or Al

Overview
- Provides outstanding cooling power
- Rapid heat spreading
- Composed of four forged heat sinks that are brazed on a copper base
- Recommended airspeed range: 400 to 1,000 LFM (2 to 5 m/s)
- RoHS compliant

Technical
- Material: Pure Copper
- Mfg. process: Cold forging
- Plating options: Electroless nickel, black zinc
- Base finish: Lapped
  Flatness: Better than 0.001 in/in
  Surface roughness: 16 RMS

Flexible Parameters
- Footprint (length and width)
- Height (pin length & base thickness)
- Single or multiple pins can be eliminated
- Comprehensive machining (holes, threads, clearances, etc.)



Thermal Resistance in °C/W

	Part Number	Length	Width	Height	Configuration	Material	0 m/s*	0.5 m/s	1 m/s	2 m/s	3 m/s	4 m/s
1	3-343405M	86.4	86.4	12.7	Sparse	Al	4.7	2.1	1.25	0.73	0.54	0.42
2	4-363605U	90.2	90.2	12.7	Moderate	Cu	-		1	0.57	0.39	0.29
3	3-383805M	95.5	95.5	12.7	Sparse	Al	4	1.85	1.11	0.63	0.46	0.37

○ in ◉ mm

*Air Speed in meters per second 1

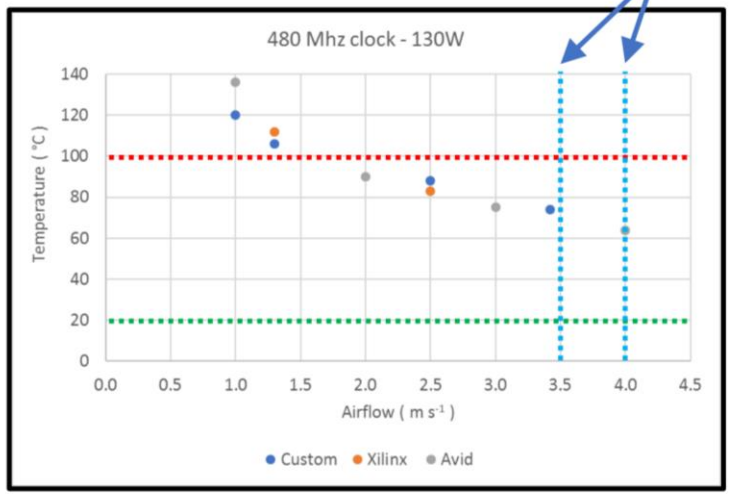
Single Mid-range Ultrascale+

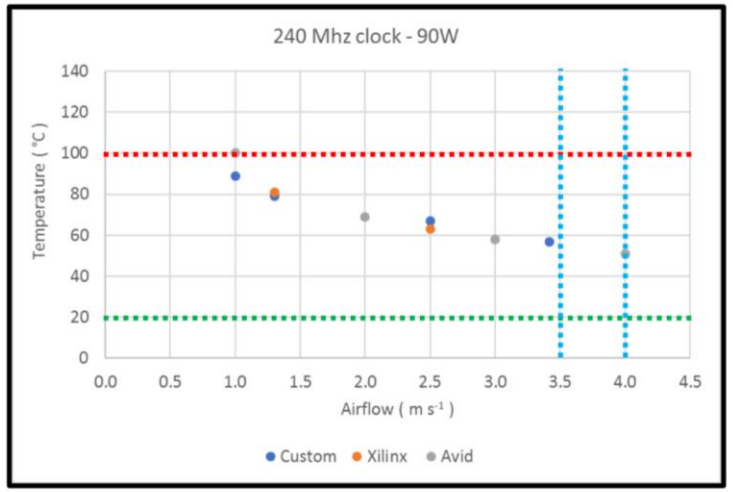
This is best case – Real world performance may be different

Set to high LUT usage, low DSP usage

VU9P, C2104
LUTs & FFs @ 80%
DSPs @ 30%
Clock @ 480 MHz
72 Low Power 10G transceivers
28 DFE 25G transceivers
4 transceivers unused
BRAM & URAM @ 80%
No I/O or external memory

Two estimates of air speed with 96 dBA in USC55





TRACKLET

System Overview

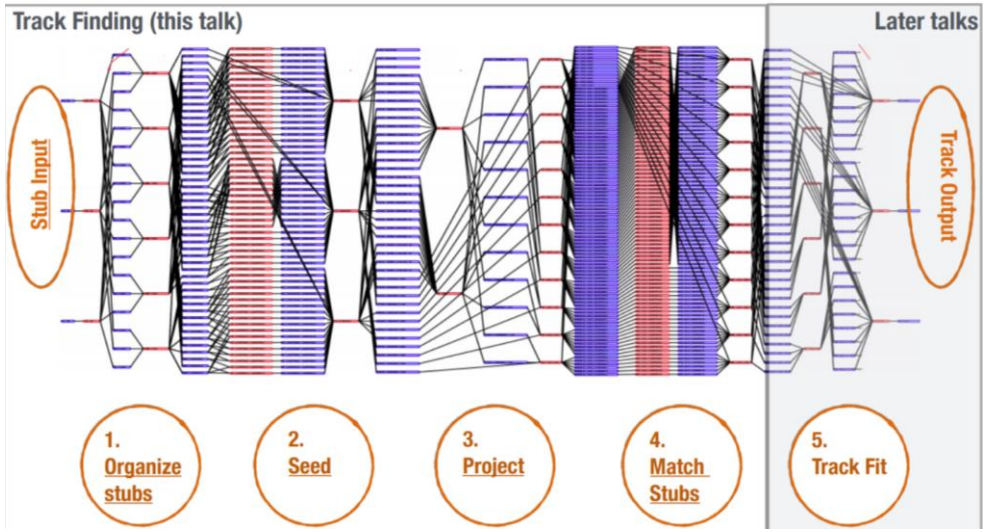
- Algorithm consists of **11 processing steps** (hand-optimized Verilog)

- Each processing step of the algorithm
  - Separated by **memories** (BRAMs):
    read in memory → process → write out memory
  - Processes a **new BX every 25 ns * TMUX** = 25 * 6 = 150 ns currently
    - Has a **fixed time** to produce its first output (latency)
    - **Pipelined design** → producing outputs until new BX (150 ns)

- **Automated wiring** (python script) of the processing modules <—> memories
  - Driven by configuration file
  - Python script wires firmware and integer emulation → identical setup
  - Also generates…

Our Favorite Tracklet Diagram

Memories
Processing modules



Step 1) Stub Organization

- Stubs coming from DTC are split by the **Layer Router** into each layer/disk

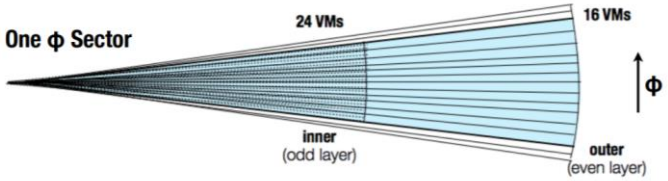
- Stubs in from DTC
  - Header (with BX)
  - Encoded order
  - Each stub
  - Trailer

Quantity	bits
Number of hits in L1	35:30 (6 bits)
Number of hits in L1+L2	29:24 (6 bits)
Number of hits in L1+L2+L3	23:18 (6 bits)
Number of hits in L1+L2+L3+L4	17:12 (6 bits)
Number of hits in L1+L2+L3+L4+L5	11:6 (6 bits)
Number of hits in L1+L2+L3+L4+L5+L6	5:0 (6 bits)

Tracklet 1.0 Scheme — strategy will depend on cabling

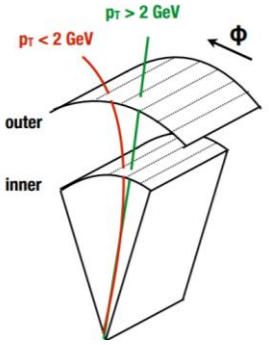
Virtual Modules

Tracklet 2.0 Configuration
Old Config. in Backup



- Split the ϕ sector into smaller ϕ regions (full length in z)
  - Inner (odd) layers (eg. L1) **24 ϕ divisions**
  - Outer (even) layers (eg. L2) **16 ϕ divisions**
    VM memories for even layers split into 8 z bins

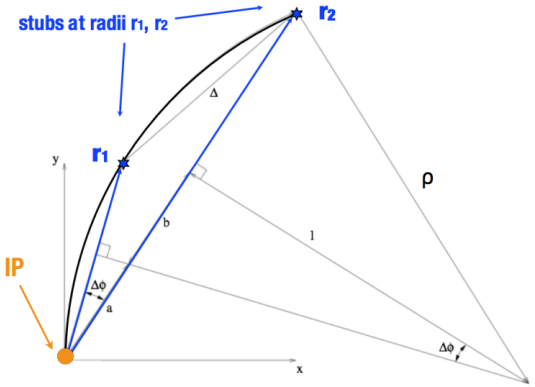
- Total of 24 * 16 = 384 pairs of VMs,
  only **120 pairs** consistent with p_T > 2 GeV

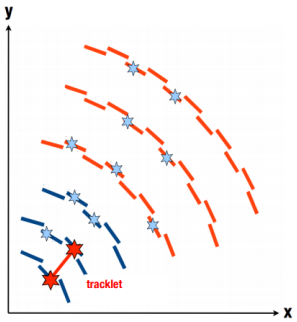


TRACKLET

Tracklet Formation

- Seed by forming a **tracklet** from pairs of **stubs** in adjacent layers (or disks)
  - Initial track parameters from stubs + IP
  - Must be consistent with $p_T > 2$ GeV, $|z_0| < 15$ cm





Inverse of radius of curvature:

$$\rho^{-1} = \frac{2\sin\Delta\phi}{r_2 - r_1}\left(1 + \frac{2r_1 r_2}{(r_2 - r_1)^2}(1 - \cos\Delta\phi)\right)^{-1/2}$$

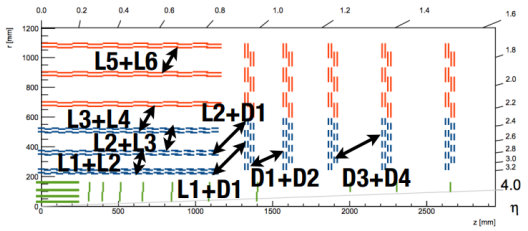
Starting point, Taylor expand to translate to calculation for FPGA

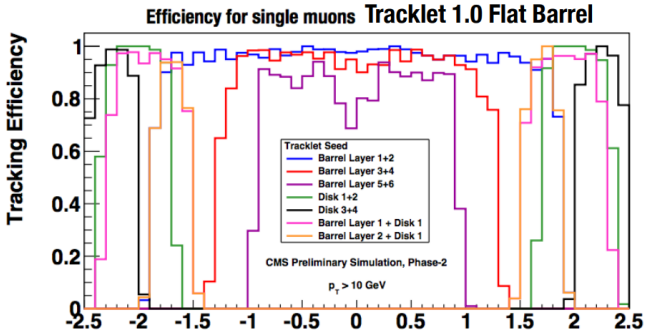
Details on integer calculations in DN

Note: Using full stub resolution!

Built in Redundancy

- **Seed multiple times** in parallel → good coverage & redundancy
  - Barrel: **L1+L2**, **L2+L3**, **L3+L4**, **L5+L6**
  - Disk: **D1+D2**, **D3+D4**
  - Overlap: **L1+D1**, **L2+D1**



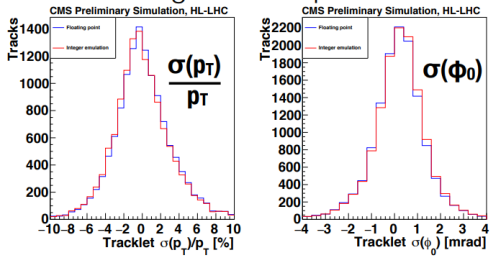


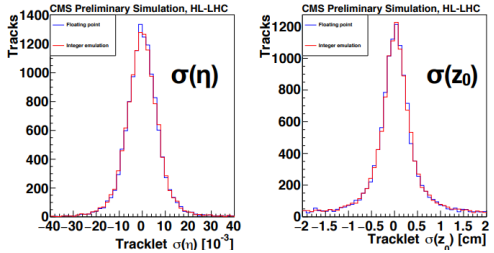
Adaptable

Can add more seeding combinations if needed (and if resources allow)

Tracklet Paramater Resolutions

- Even at this stage, tracklets have good track parameter resolutions





Integer Emulation
Floating-point calculation

Step 3) Projections

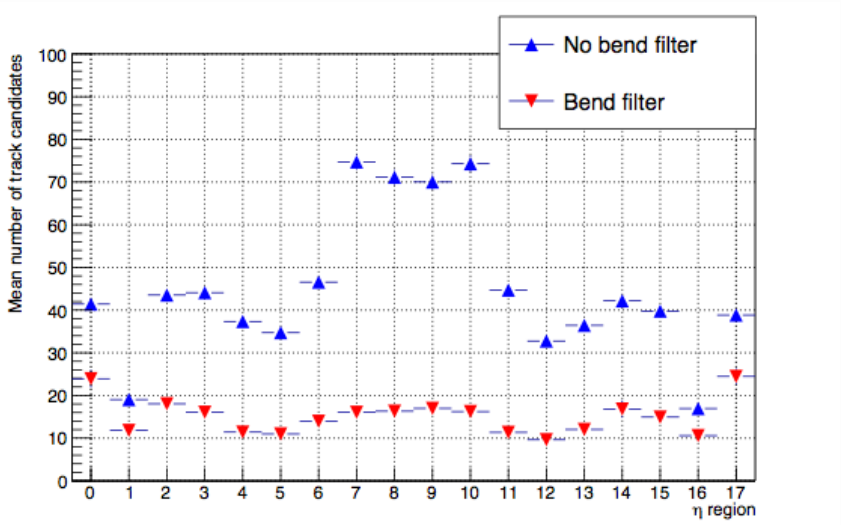
- Use tracklet to **project** track to other layers and disks
  - Project **both inwards and outwards**
  - All projections made in **parallel** in the **Tracklet Calculator**

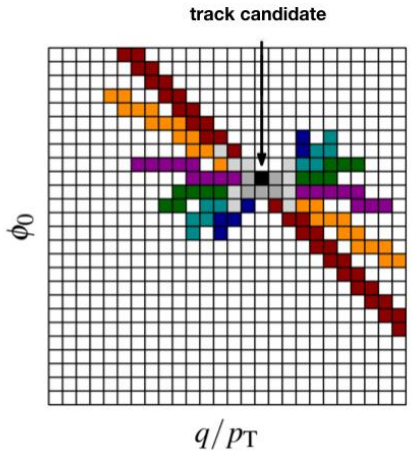
- Projection calculation:
  - Use the average radius (z) for barrel (disk)
  - Correct avg. position by derivatives (LUTs) to get exact projection for actual stubs

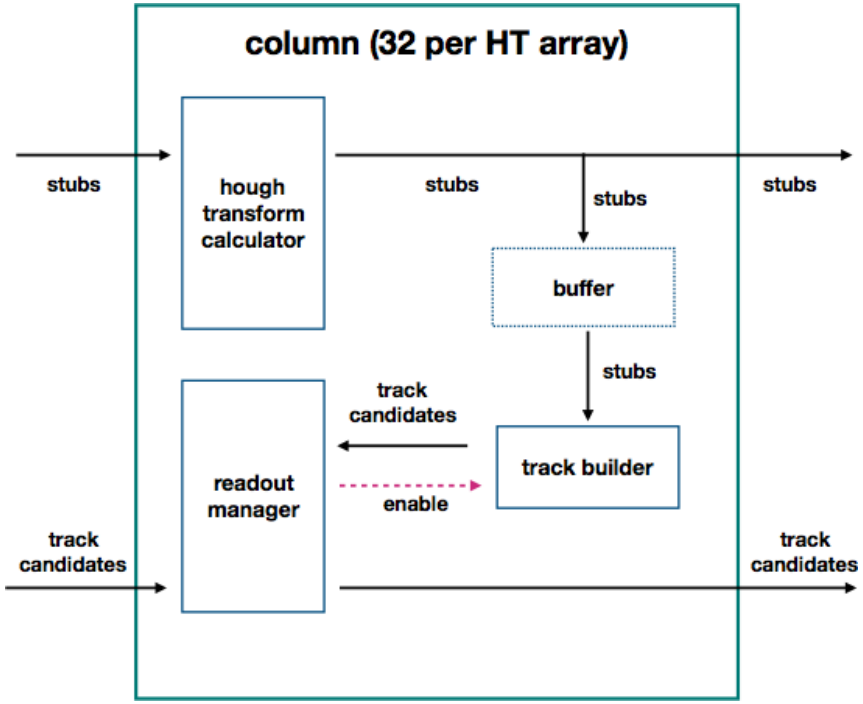
$$\phi_{proj} = \bar{\phi}_{proj} + \Delta r \frac{\partial \phi}{\partial r}$$
$$z_{proj} = \bar{z}_{proj} + \Delta r \frac{\partial z}{\partial r}$$

NB: Keep these derivatives for use in matching

HT

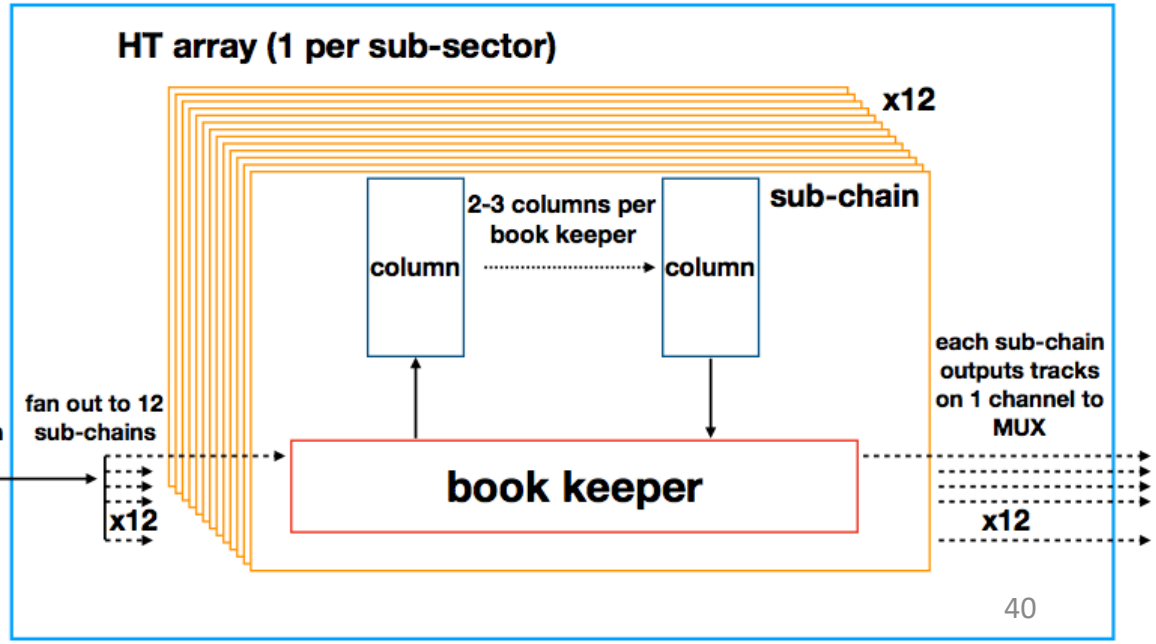




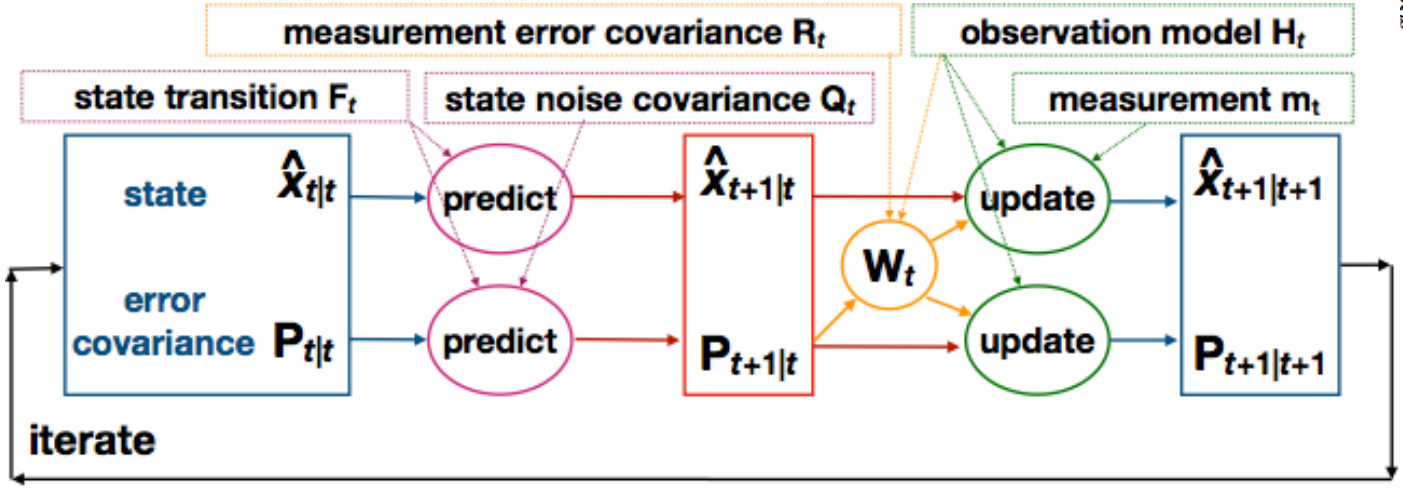


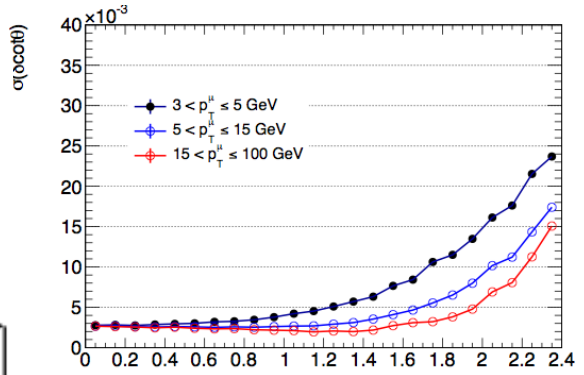
Hough Transform

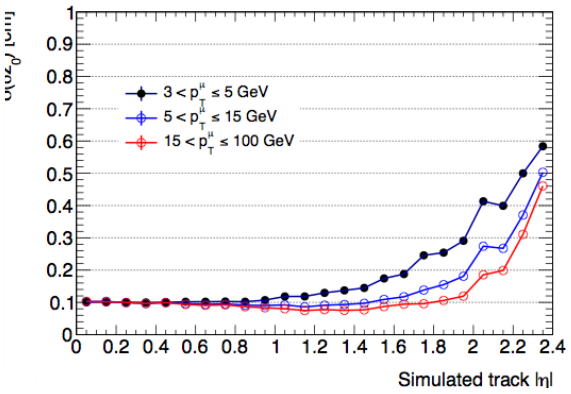
- Pipelined firmware processes one stub per clock cycle Book keeper receives stubs and propagates to each q/p_T bin (represents one array column) in turn.
- The p_T estimate of the stub from stacked modules is used to constrain the required q/p_T space
- Inside the Bin, the corresponding ϕ_0 of the stub for the column is calculated and the appropriate cell(s) are marked
- Candidates marked by stubs from > 4 layers propagate back to the Book Keeper and are read out

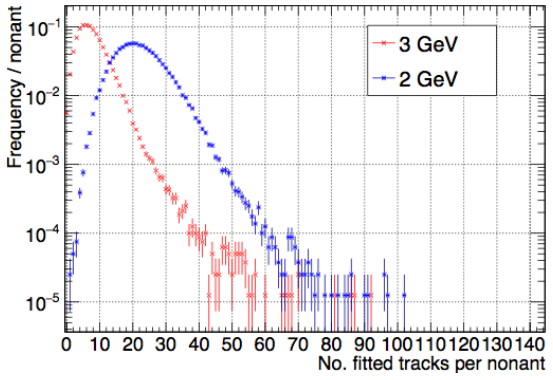


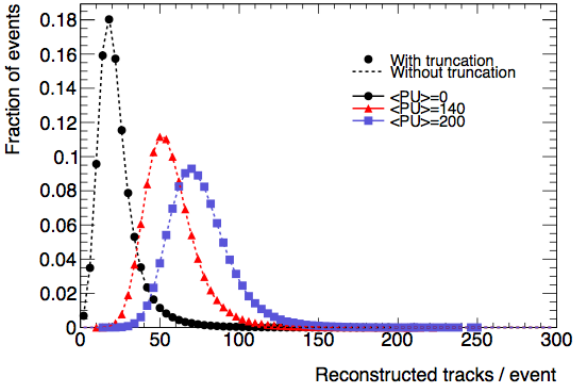
KF











Kalman Filter

-1) Incoming stubs stored in BRAM for later retrieval

-2) Seed state creator outputs initial state (HT seed)

-3) State control multiplexes incoming seeds and partially worked states

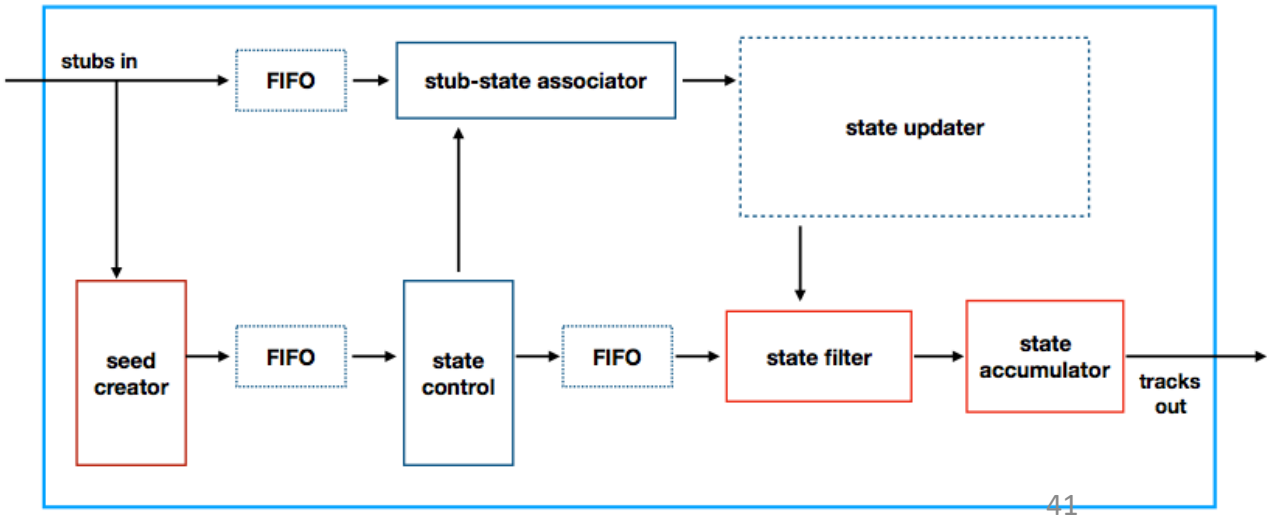
-4) Stub state associator retrieves next stub (in increasing radii) from memory

-5) Kalman filter updates matrices and state with weighted average of previous and new inputs

-6) Repeat for (a configurable) accumulation period (or until 4 stubs are added to all tracks)

-7) State filter selects the best state for each candidate (χ2)

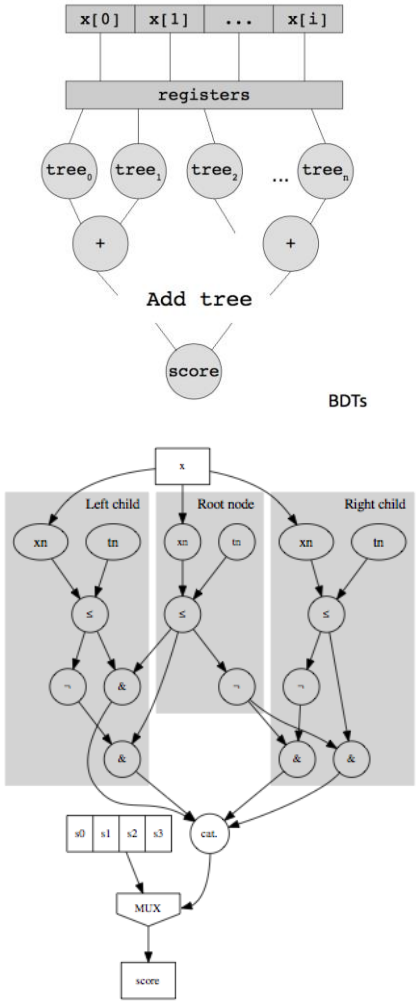
- Processing latency dominated by matrix math update, 230 ns per iteration

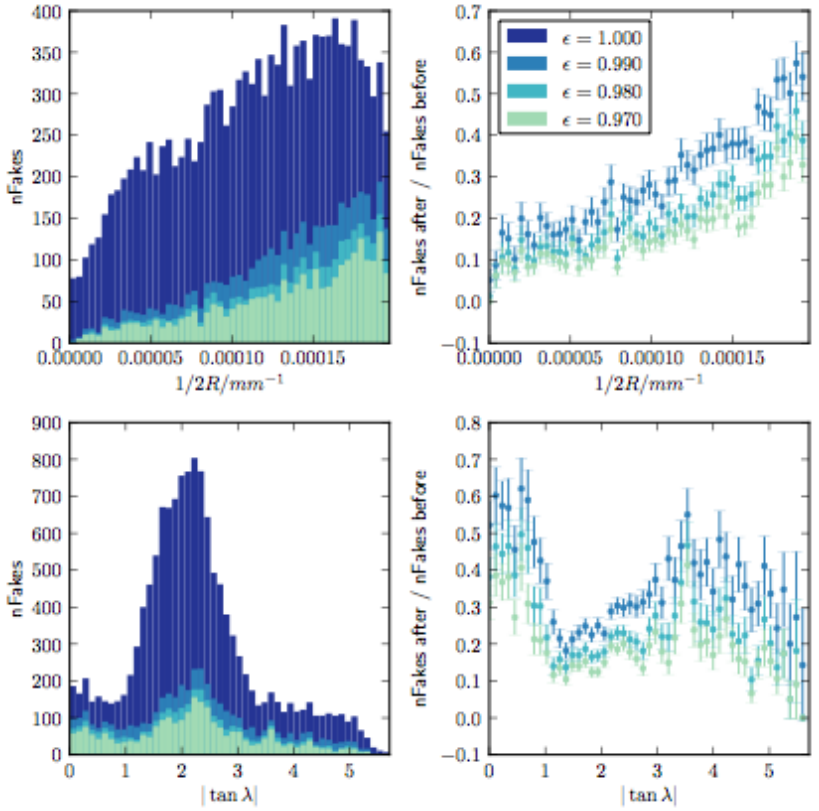


BDT IN FPGA

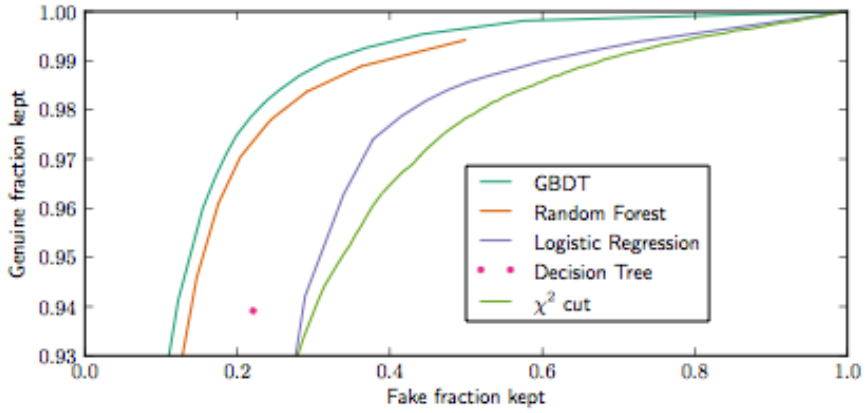
BDT Fake Rejection

- Idea to use a gradient boosted decision tree, implemented in FPGA logic, to select and remove fake tracks after the track fit
- Make a static, fully pipelined implementation of a pre-trained BDT ensemble
- Train ensemble on a CPU (using scikit-learn)
- Export trained ensemble to JSON file
- Read by firmware
- 4 integer features
  - $-\chi^2$, $|\ 1/p_T\ |$, $|\ \tan\lambda\ |$, num. skipped layers
- 100 trees, depth 3
- Tuneable on eff. vs fake rate curve
- Latency 12 clocks @ 400 MHz (Stratix V), 30 ns only!





KF, $t\bar{t}$ at 200 PU



efficiency loss [%]	1.0	0.5	0.1
fake rate reduction [%]	70	50	35

One BDT	V7-690	KU-115	VU-9P
LUTS [%]	2.24	1.46	0.82
FFs [%]	1.14	0.75	0.42

Scaling	n. Trees	max. Depth
Latency	log(n)	n
FPGA Resources	n	2^n