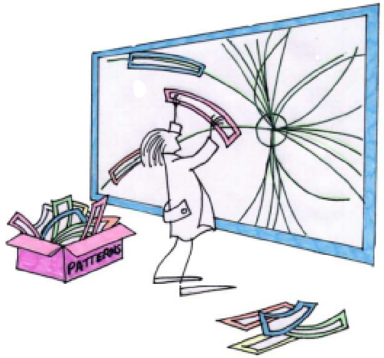


RUPRECHT-KARLS-
UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

The Fast (but not furious) Tracker for ATLAS: a track trigger based on FPGAs and Associative Memory chips

Misha Lisovyi
with many inputs and help from FTK team
LAL Orsay seminar
6.12.2016

ATLAS-TDR-021

Outline

Outline

- Outline

Outline

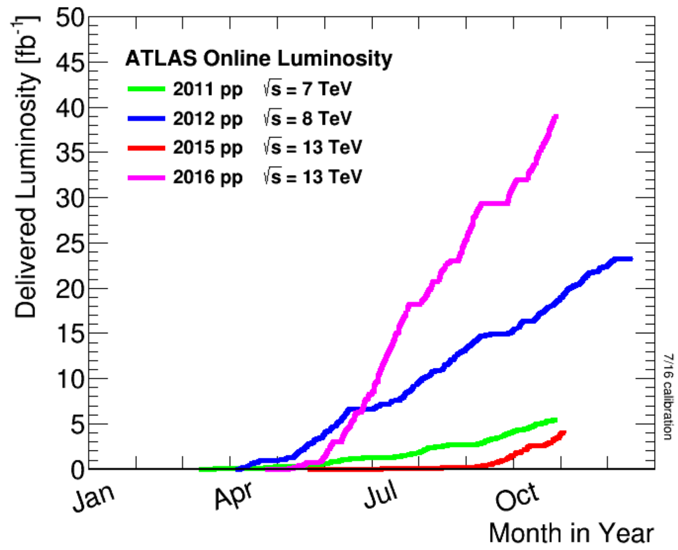
- Outline
- The rest of the slides…

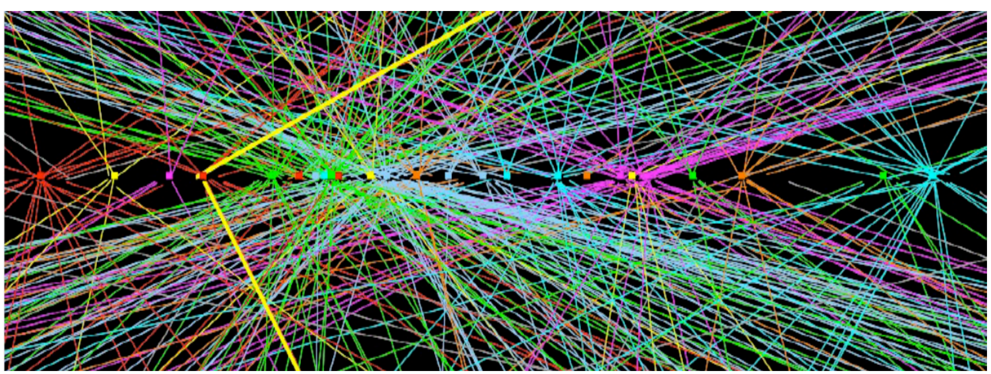
Outline

- Outline
- The rest of the slides...
- Summary (50 minutes later)

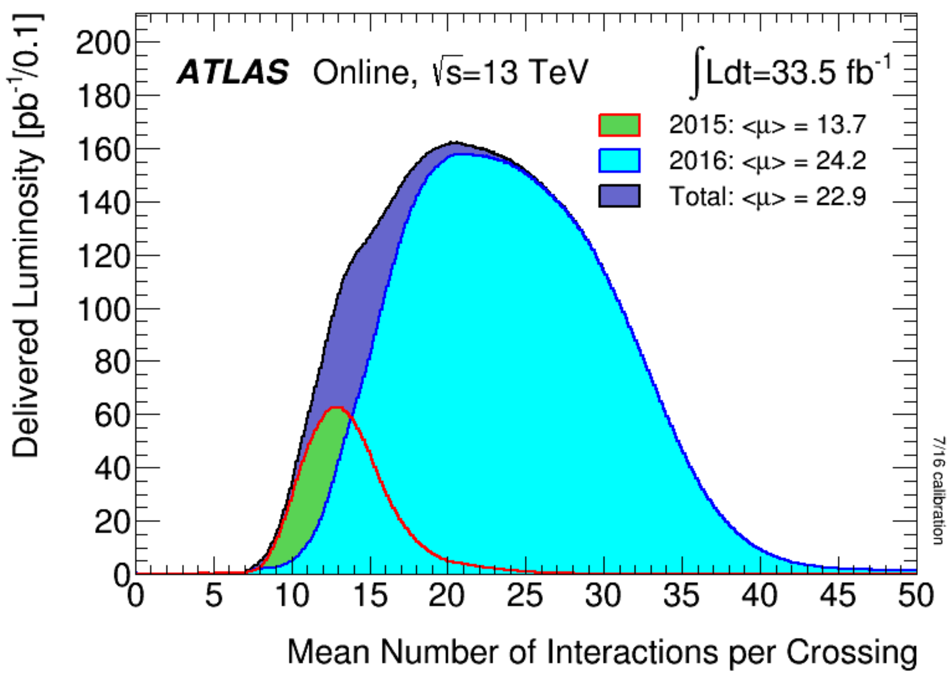
Data taking in ATLAS

- Smooth data taking @LHC, steady luminosity increase (peak instanteneous luminosity ~$1.4*10^{34}$ cm^{-2} s^{-1}).
- Huge data set!
- On average 24 pp interactions per bunch crossing (pileup) in 2016 data.
- Challenging for physics analyses and triggers

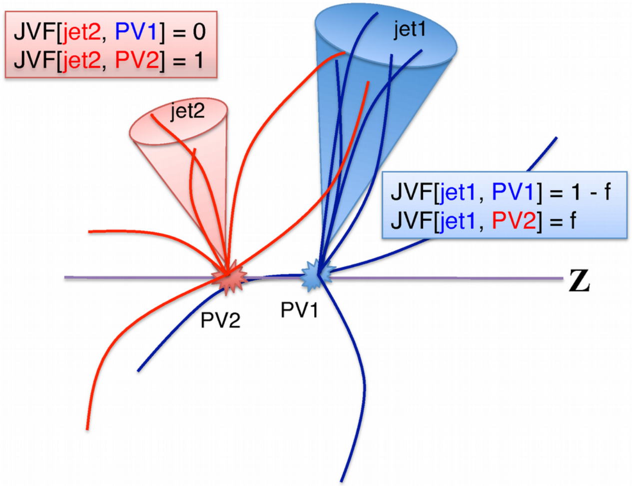


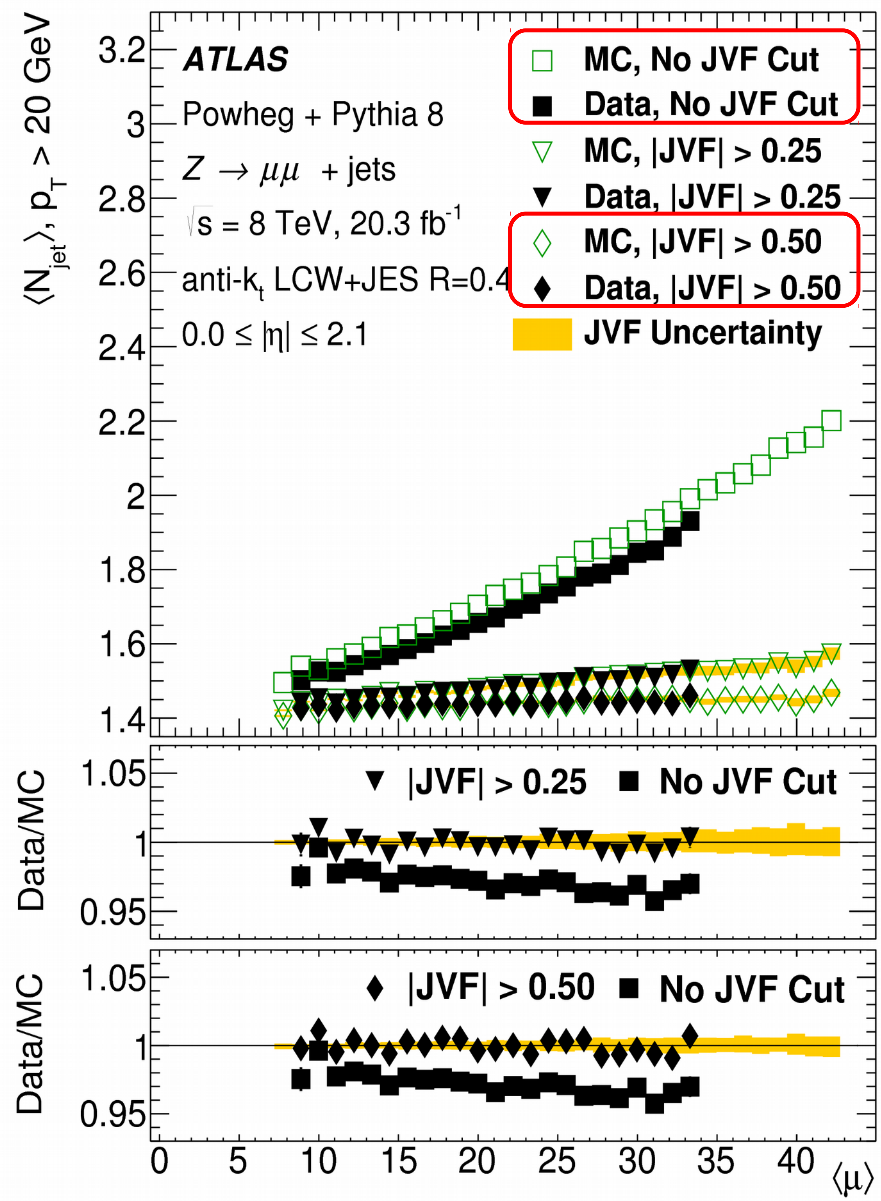


Z →μμ event with 25 reconstructed vetrices @8 TeV in ATLAS



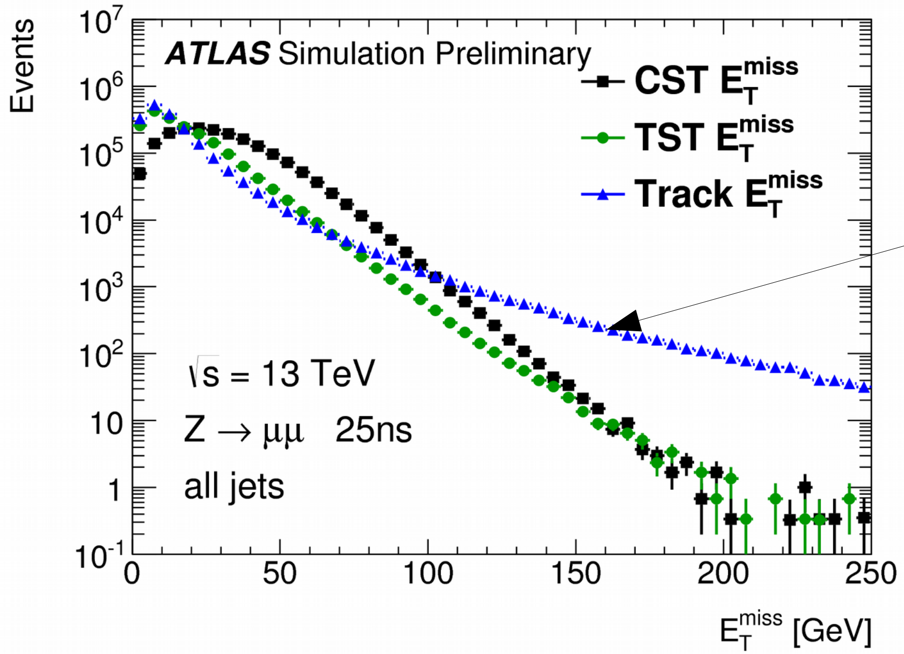
Tracks for jets

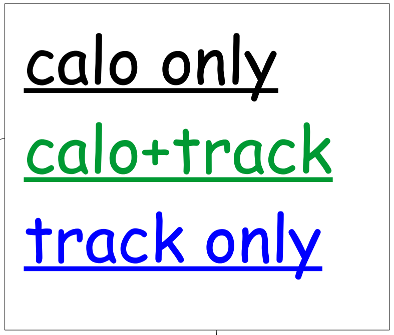


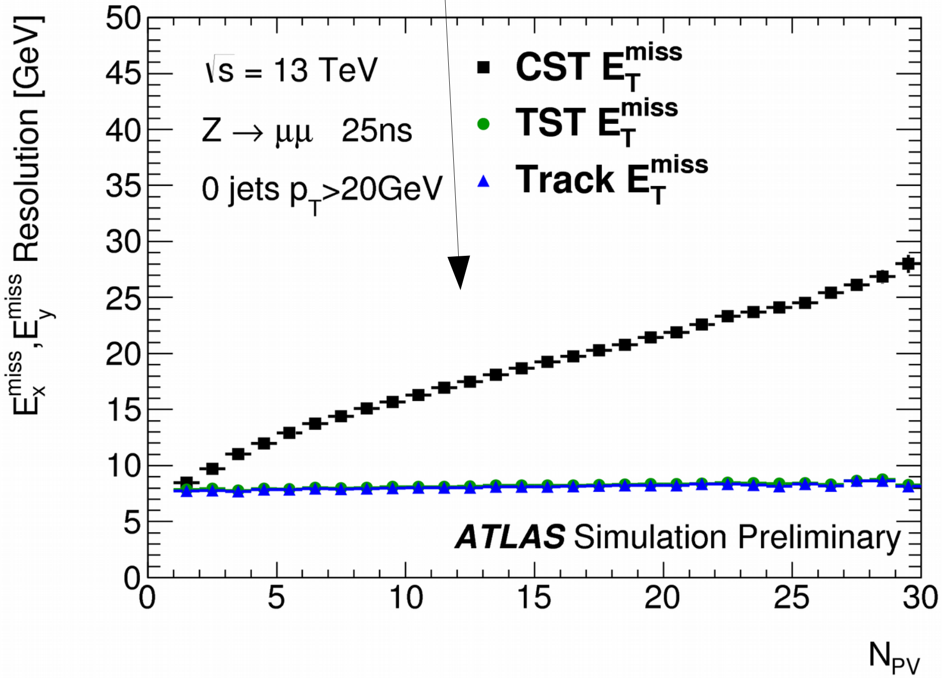


- Use tracks to identify jets that come from the primary vertex.
- Jet Vertex Fraction (JVF, improved and advanced for Run2): a fraction of p_T originating from tracks associated to the primary vertex.
- Improved stability against pileup (μ), when cutting on JVF.
- As input, JVF needs all tracks in the event and vertices reconstructed from those

Tracks for E_T^{miss}

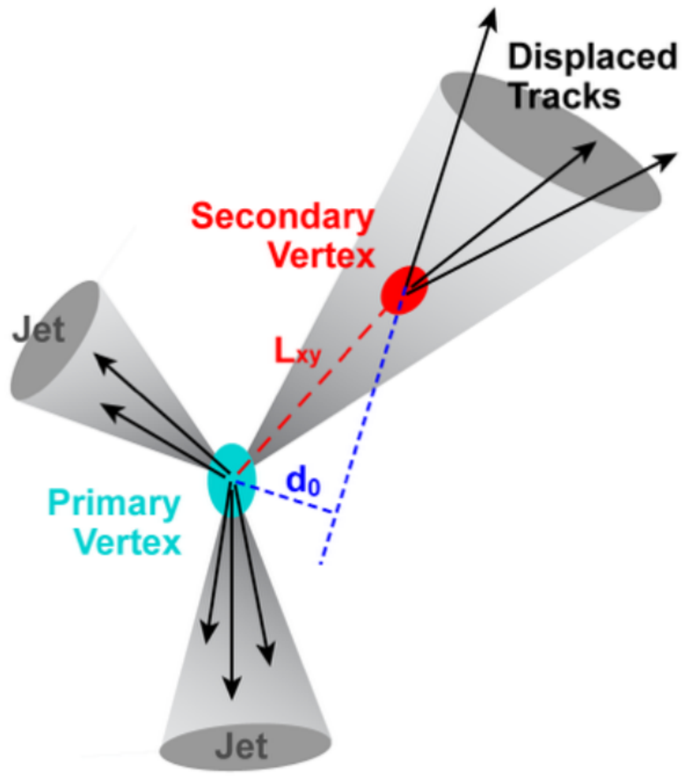


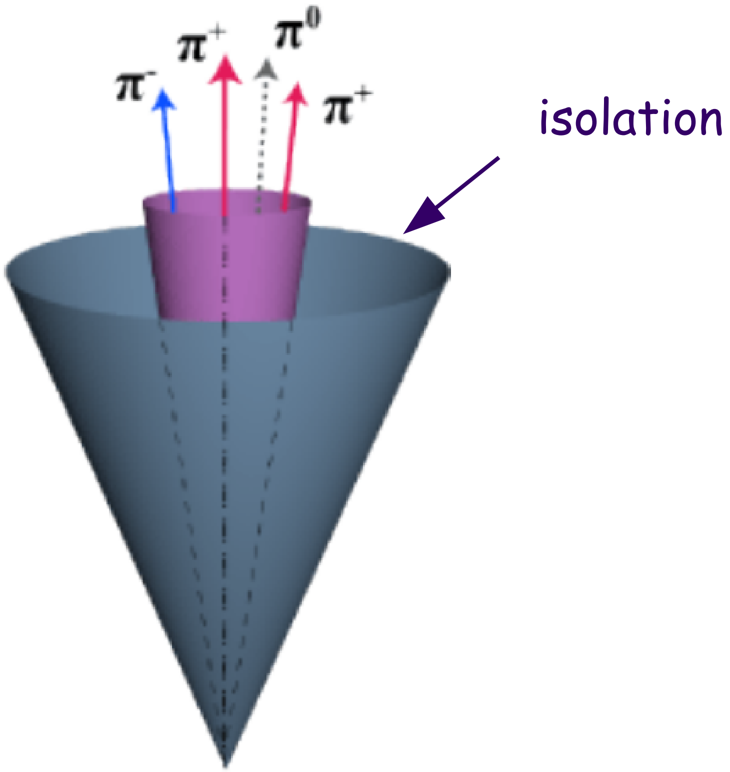




- Tracking information allows to reconstruct E_T^{miss} more precisely.
- Combination of calo+track is more stable vs pileup (number of primary vertices, N_{PV}) than calo only, and removes a bias introduced from neglecting neutral particles in track only
- Calo+track requires reconstruction of tracks from the primary and non-primary vertices

Tracks for flavour tagging





- Reconstruction of displaced secondary vertices: b-tagging
- Calculate isolation requirement using tracks inside a jet: τ-tagging

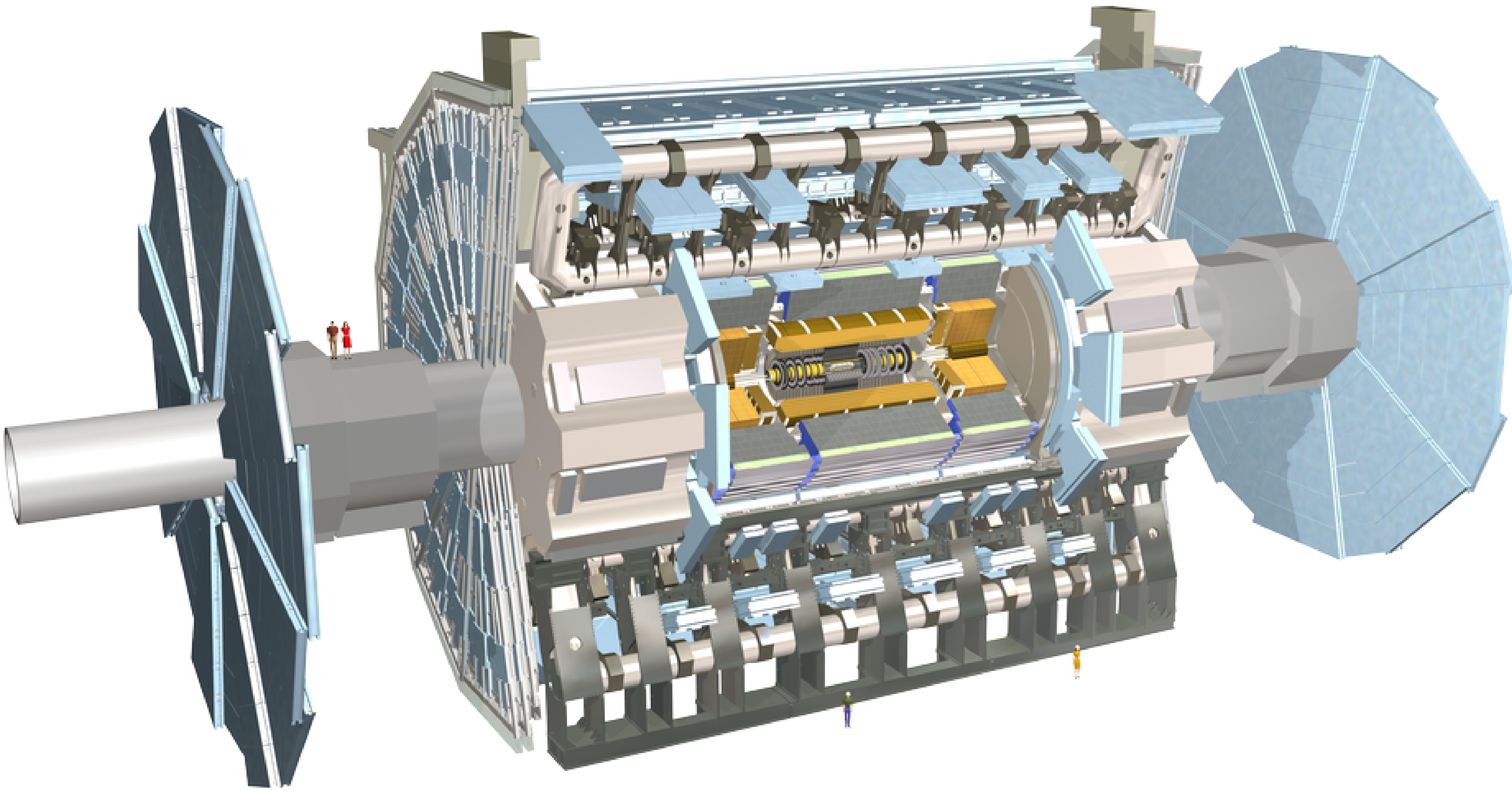
Tracks for physics

- Tracking is extensively used in physics analyses (among other purposes):
  - to mitigate pileup effects for jets and E_t^{miss}
    - key signatures of SM and BSM processes
    - increased importance with higher pileup in Run2-3 and HL-LHC
  - to reconstruct advanced event topologies in b and τ decays
- However, these benefits are available only in offline reconstruction
  - need to have access to tracks in trigger to effectively record events
  - otherwise one needs to apply prescales or increase thresholds in triggers
  - => interesting events might not end up on tape, thus, benefits are useless

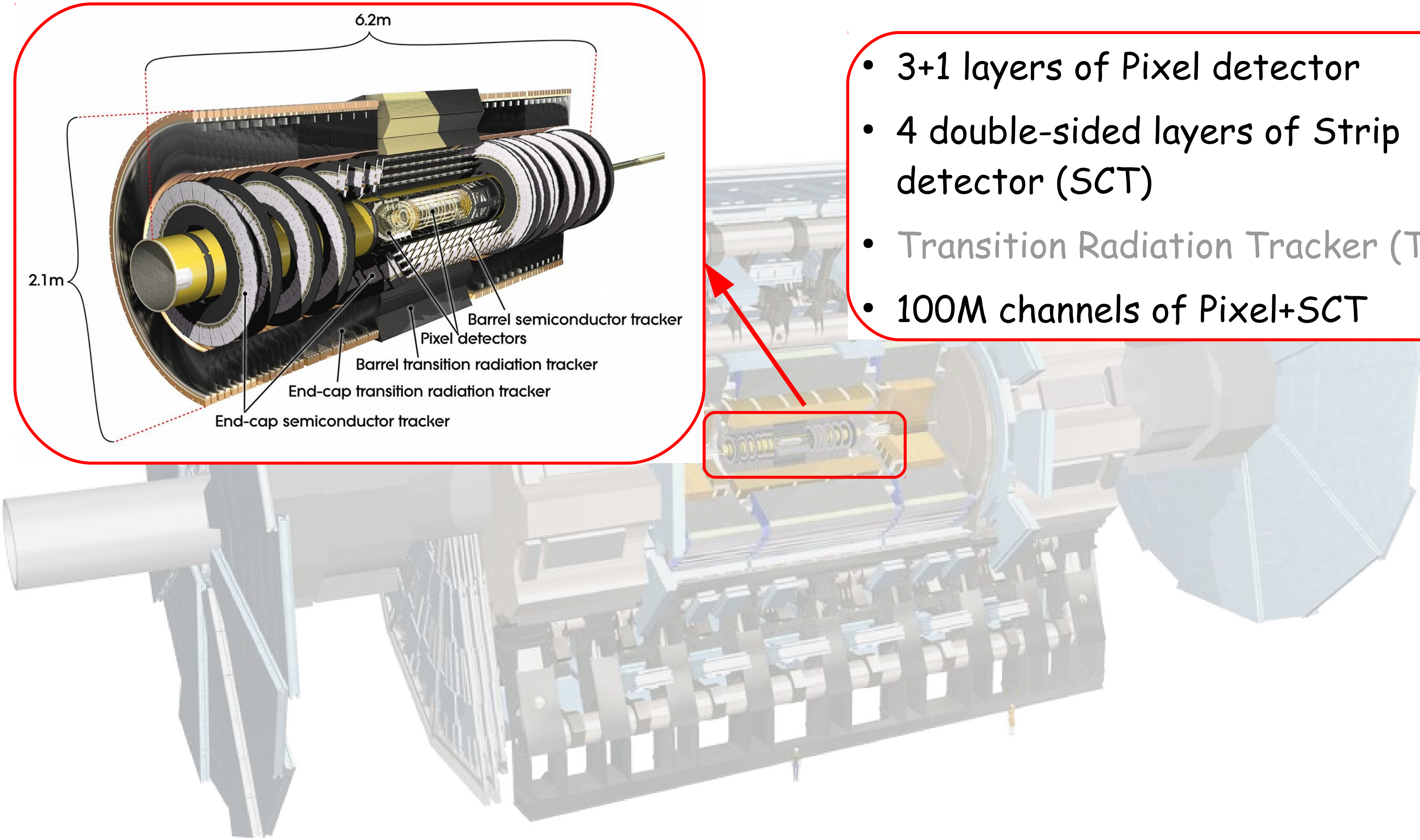
What did we learn (so far)

- Tracking is extremely valuable and used extensively in physics analysis, but it is also needed in trigger to be able to record interesting events.

ATLAS detector



ATLAS tracking system

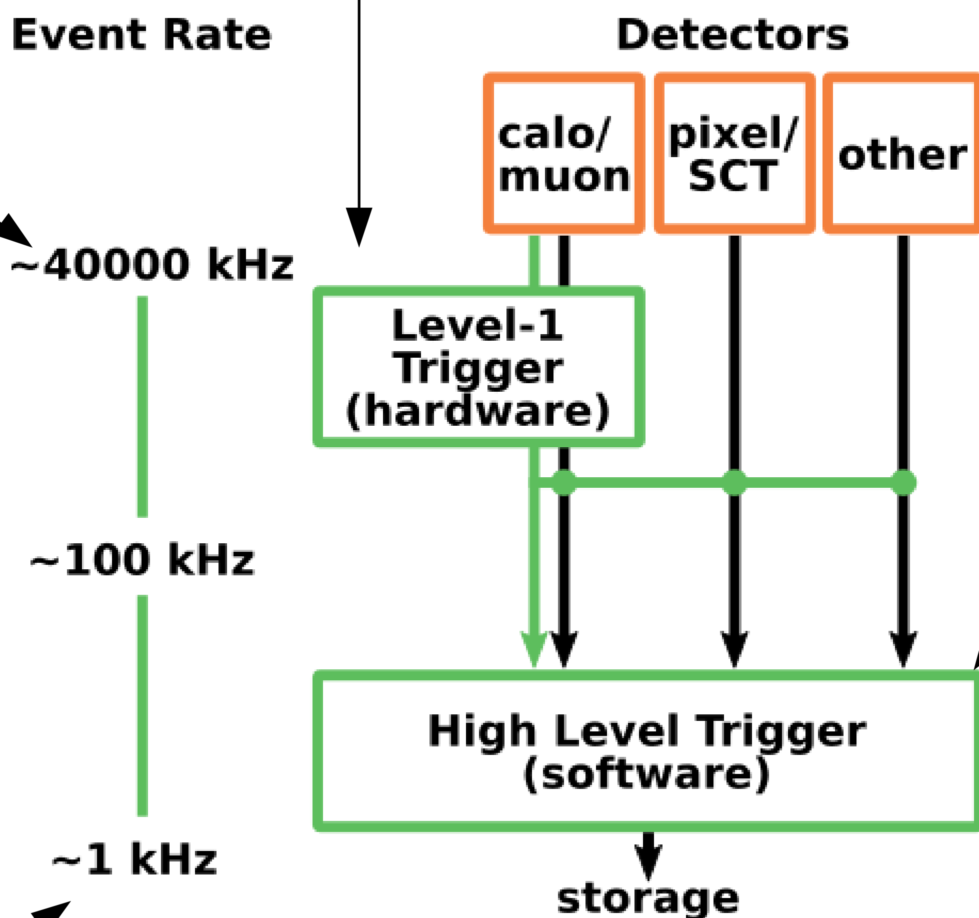


- 3+1 layers of Pixel detector
- 4 double-sided layers of Strip detector (SCT)
- Transition Radiation Tracker (TRT)
- 100M channels of Pixel+SCT

ATLAS trigger system

Dedicated hardware that utilises Calorimeter and Muon systems only

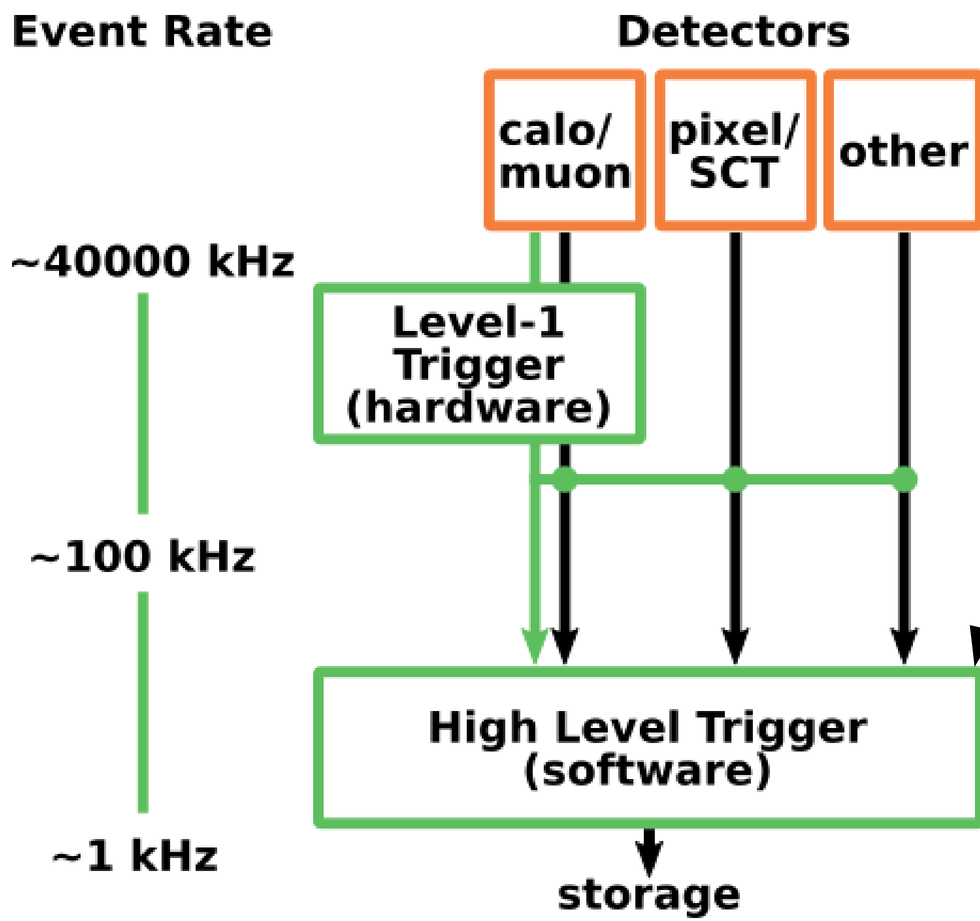
LHC bunch crossing rate



- Computer farm of ~40k CPUs
- Objects are reconstructed within Regions of Interest (RoI), that are seeded by items in Level1 trigger decision

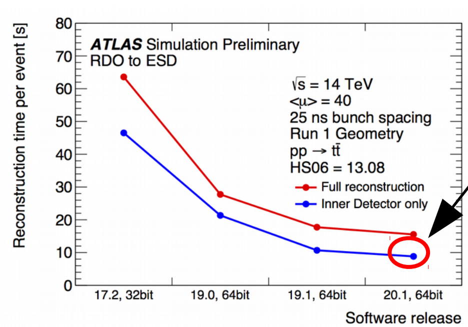
ATLAS recording rate

ATLAS trigger system

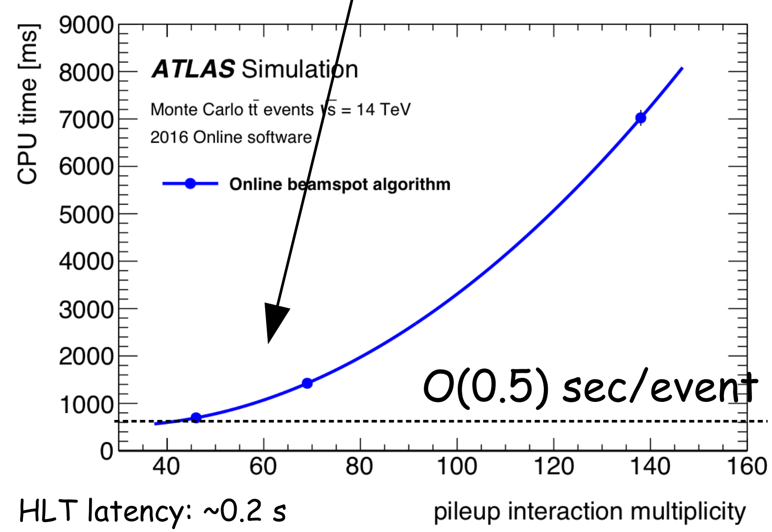


- Use Calorimeters and Muon systems predominantly
- Tracking done for specific signatures (e.g. τ tagging) in specific RoIs only
- Only a couple of triggers use full track reconstruction:
    - low rate (large prescale)

Tracking on CPUs is very expensive in CPU time & non-linear in μ



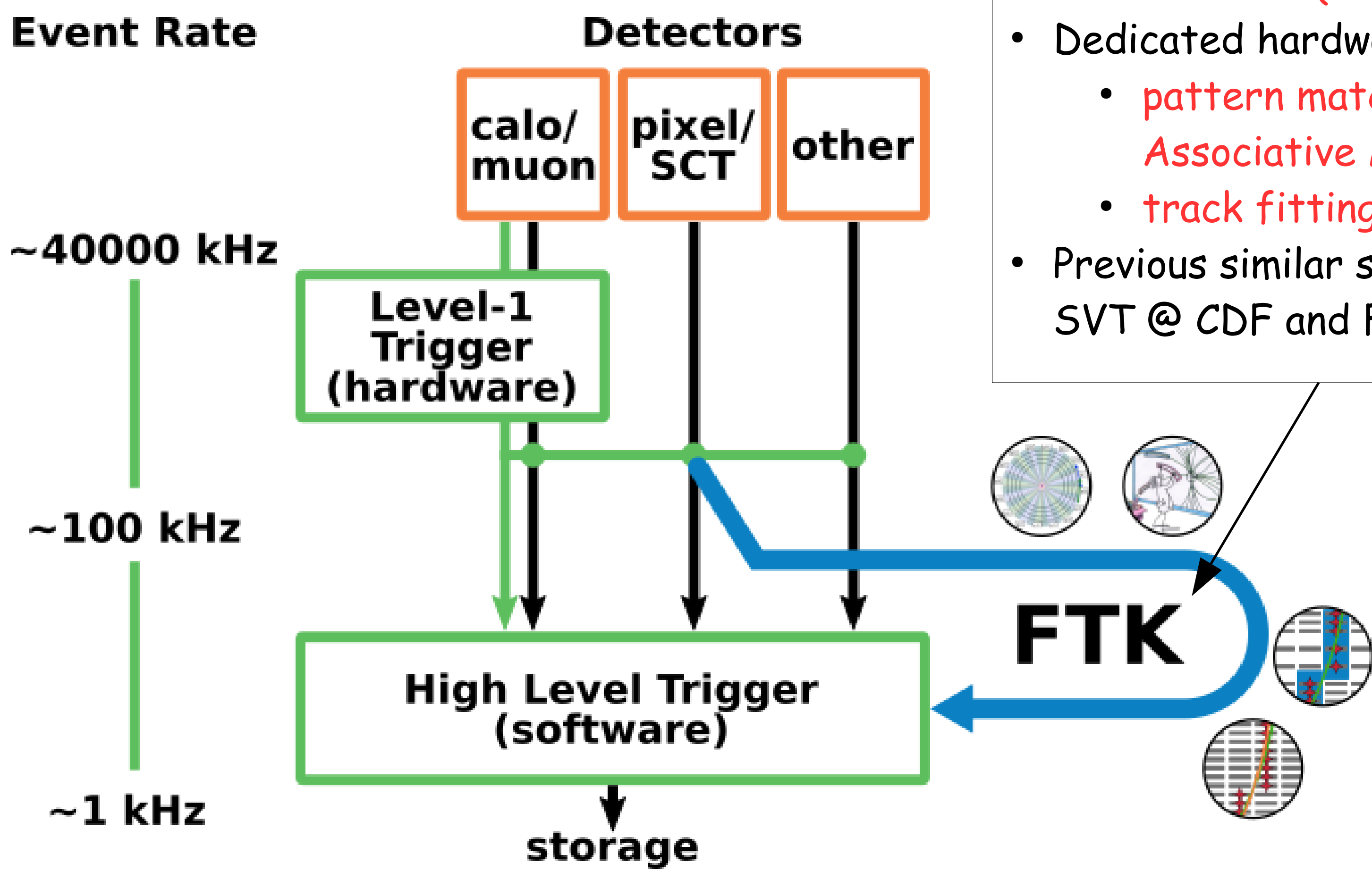
Full offline tracking is even 10x slower...



HLT latency: ~0.2 s

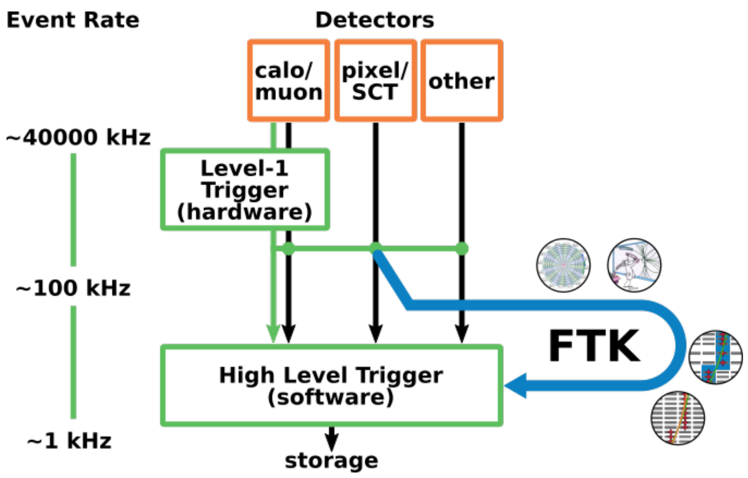
ATLAS trigger system + FTK

- Fast TracKer (FTK) system
- Dedicated hardware:
  - pattern matching in Associative Memory chips,
  - track fitting in FPGAs
- Previous similar systems: SVT @ CDF and FTT @ H1



FTK design goals

- Full track reconstruction for p_T > 1 GeV as input to HLT:
  - processing events @ 100 kHz
    - on each Level1 accept decision
  - full pixel and strip readout for each event
    - 380 optical input links
  - latency up to 0.1 ms
    - significant improvement in processing speed over CPU-based tracking (~500 ms)
  - highly parallel system
    - improved processing speed, expandable in a staged way
  - designed and evaluated at ~60 pileup interactions per event, works up to ~80 pileup
    - Covers Run2 and Run3 conditions until HL-LHC



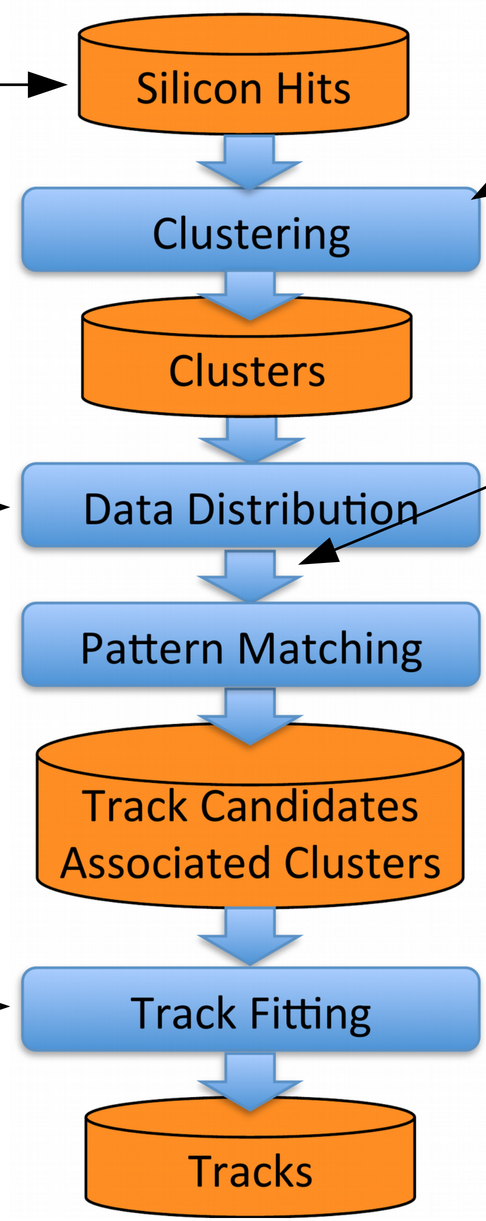
What did we learn (so far)

- Tracking is extremely valuable and used extensively in physics analysis, but it is also needed in trigger to be able to record interesting events.
- At the moment, ATLAS trigger system makes only very limited use of tracking.
- Fast TracKer trigger will do full tracking and provide inputs to High Level Trigger.

FTK processing logic

Pixel and strip hits from the detector

Pixel and strip hit clustering similar to offline



Clusters of coarse granularity to reduce data size, but still meet performance goals

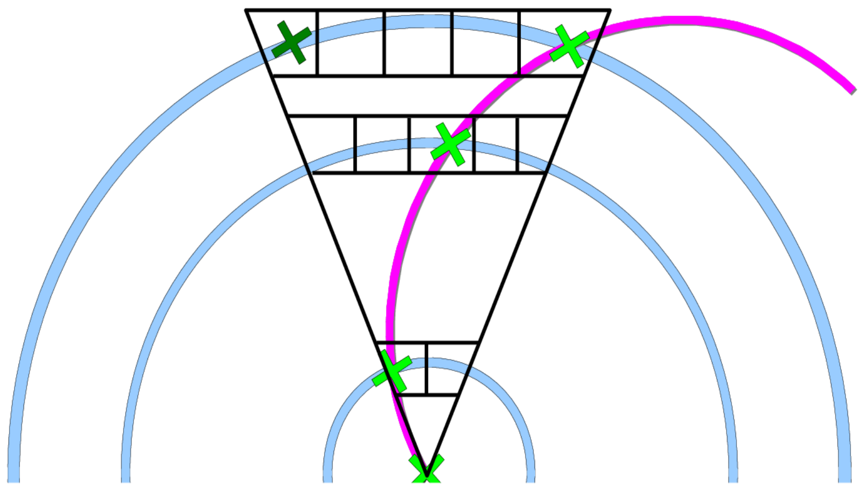
Distribute clusters within 64 (η,φ) regions to parallelize the tracking task

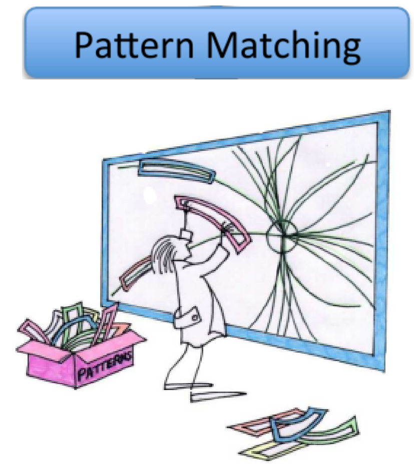
Find preliminary particle trajectories using coarse clusters. Most demanding step on CPUs.

Fit tracks using pixel and strip clusters that correspond to the coarse clusters on the matched pattern

FTK magic

- Associative Memory (AM) ternary content-addressable memory

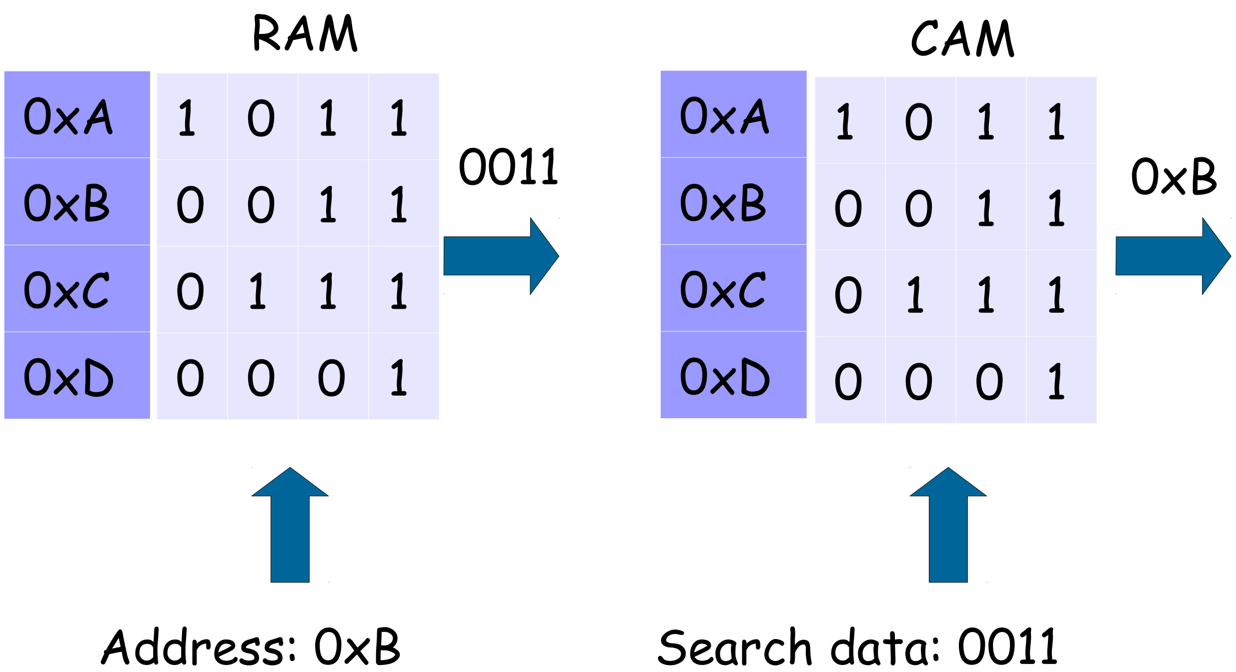




- Pattern matching boils down to a check if a combination of hits can lie on a particle trajectory:
  - pre-compute "valid" hit combinations (= simulate all possible particles traversing the tracking detector)
  - implement a fast search of those pre-computed patterns in an event

FTK magic

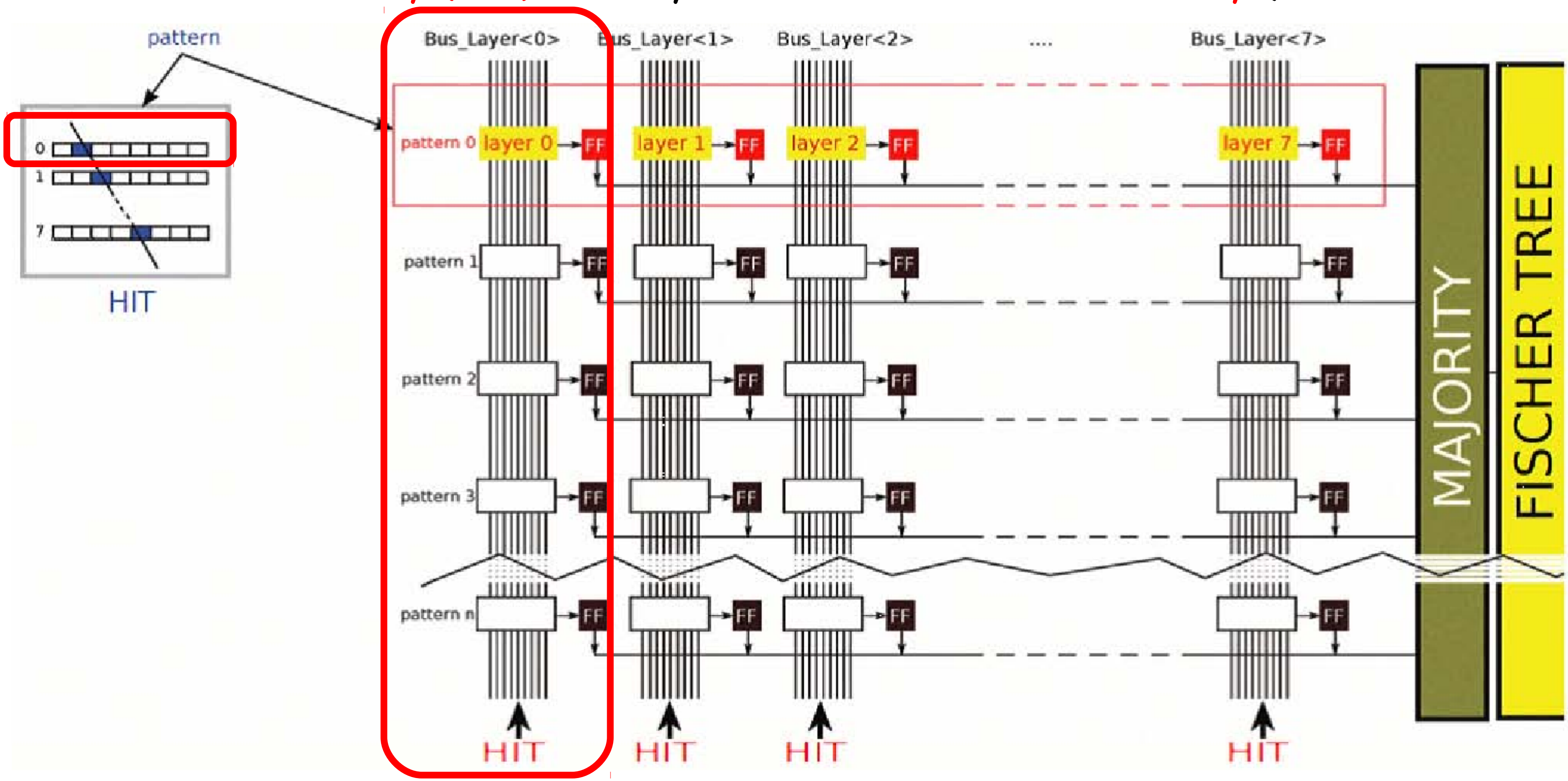
- Associative Memory (AM) ternary content-addressable memory



- Content-addressable memory (CAM) allows a very fast search of data matching.
- Many commertial solutions (e.g. network hardware).

FTK magic

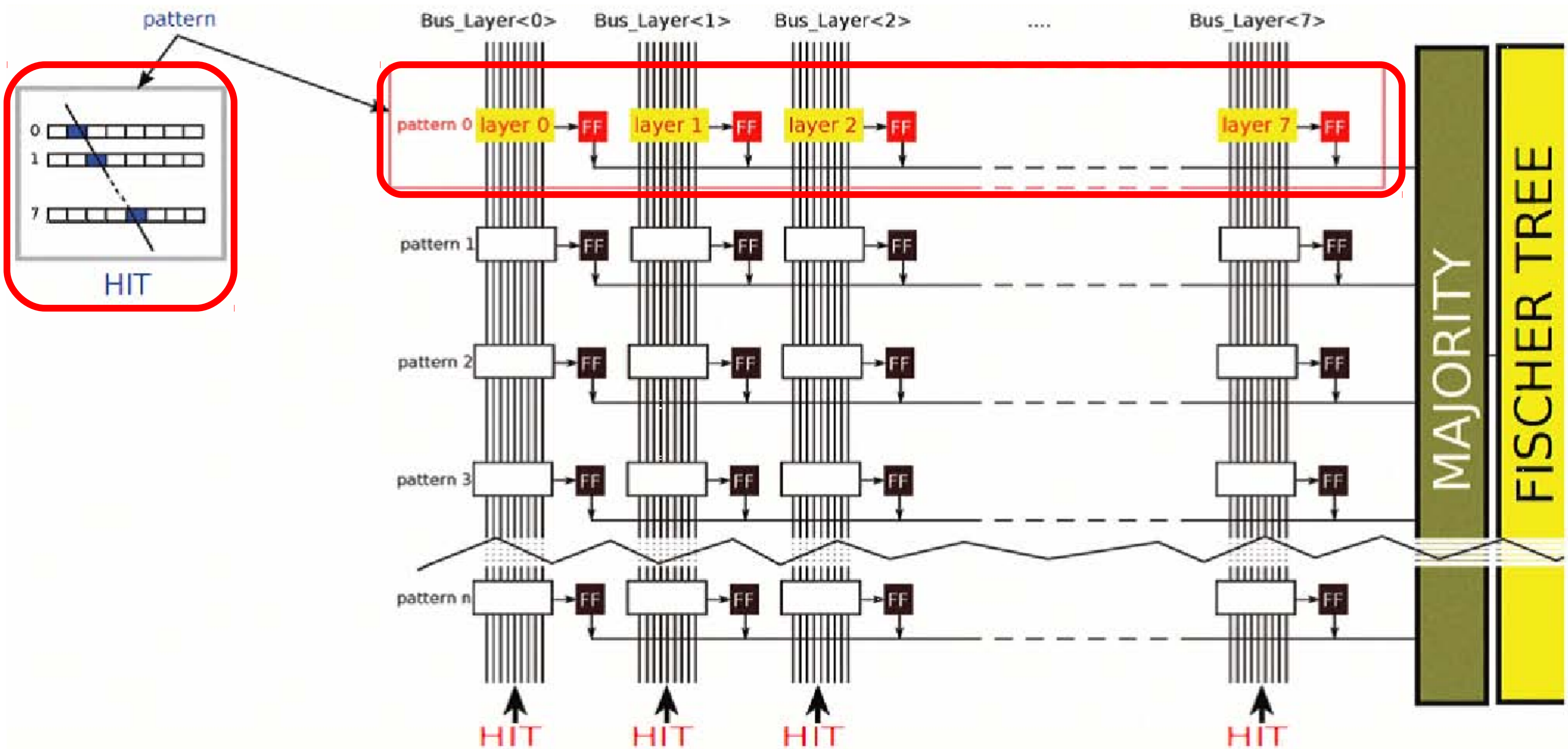
- *Associative Memory (AM)* ternary *content-addressable memory* (custom solution)



- Encoded clusters from 8 tracking layers are input via dedicated 15-bit bus lanes.
- Address space: up to 2^{15} = 32k cluster addresses can be encoded. Further extension due to splitting into (η,φ) regions.

FTK magic

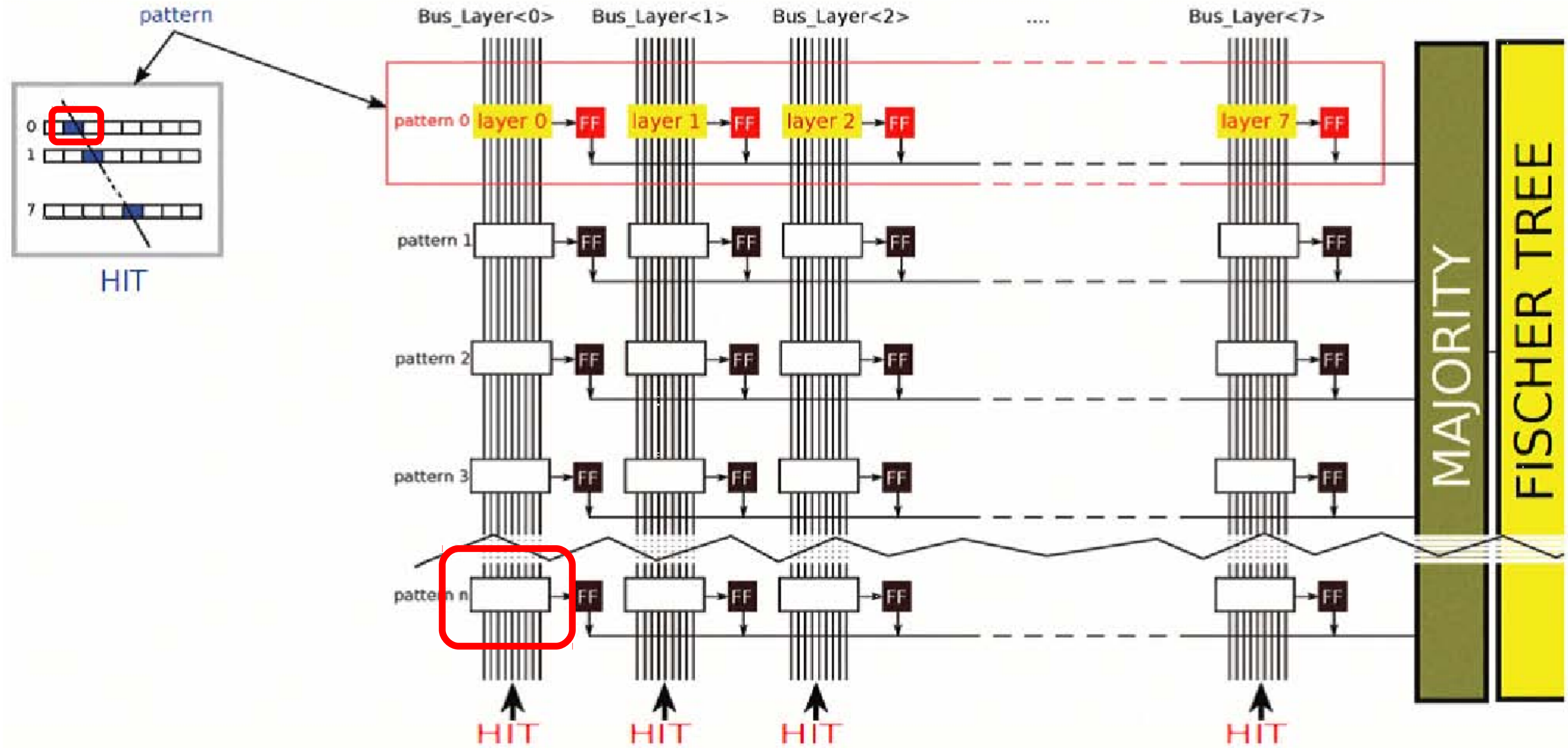
- Associative Memory (AM) ternary content-addressable memory (custom solution)



- A pattern corresponds to a row of connected CAM cells in all layers

FTK magic

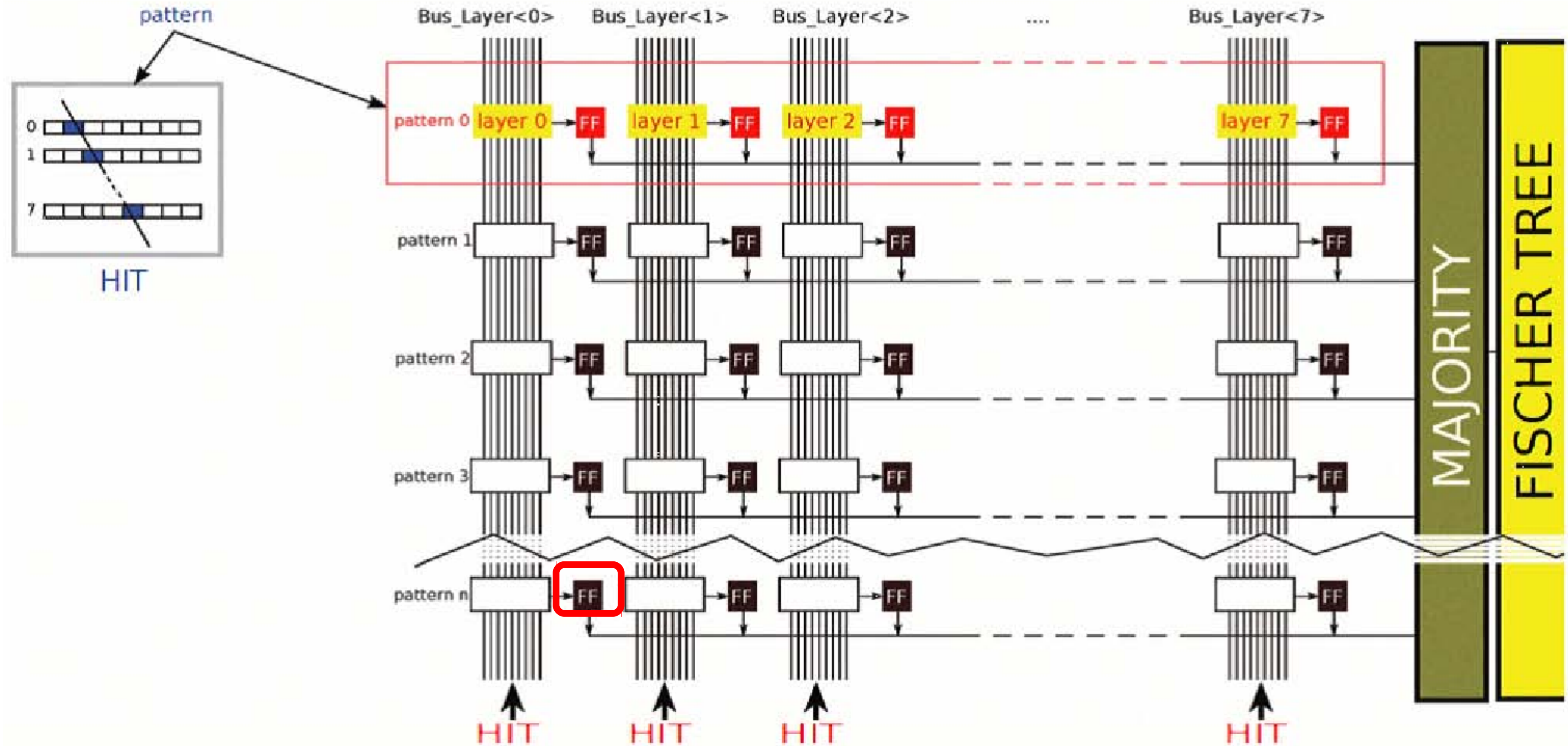
- Associative Memory (AM) ternary content-addressable memory (custom solution)



- Each cluster of each pre-computed pattern is stored in a dedicated CAM cell
- All cells in a layer are compared to an input cluster in parallel on a single clock cycle (@ 100 MHz)

FTK magic

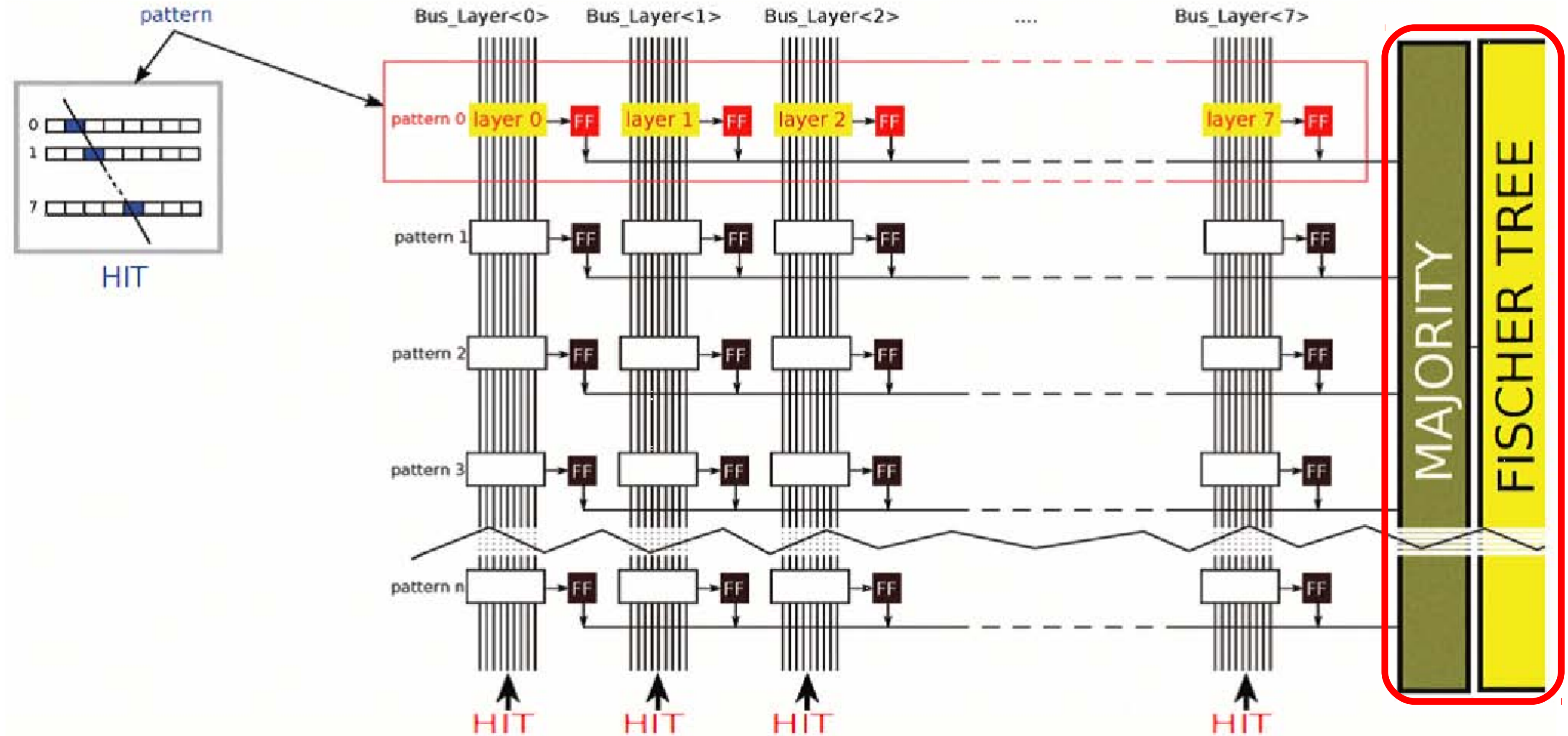
- Associative Memory (AM) ternary content-addressable memory (custom solution)



- A dedicated memory cell ("flip-flop" comparator, FF) to store that a hit was found in an event.
- FF memory is reset at the end of each event

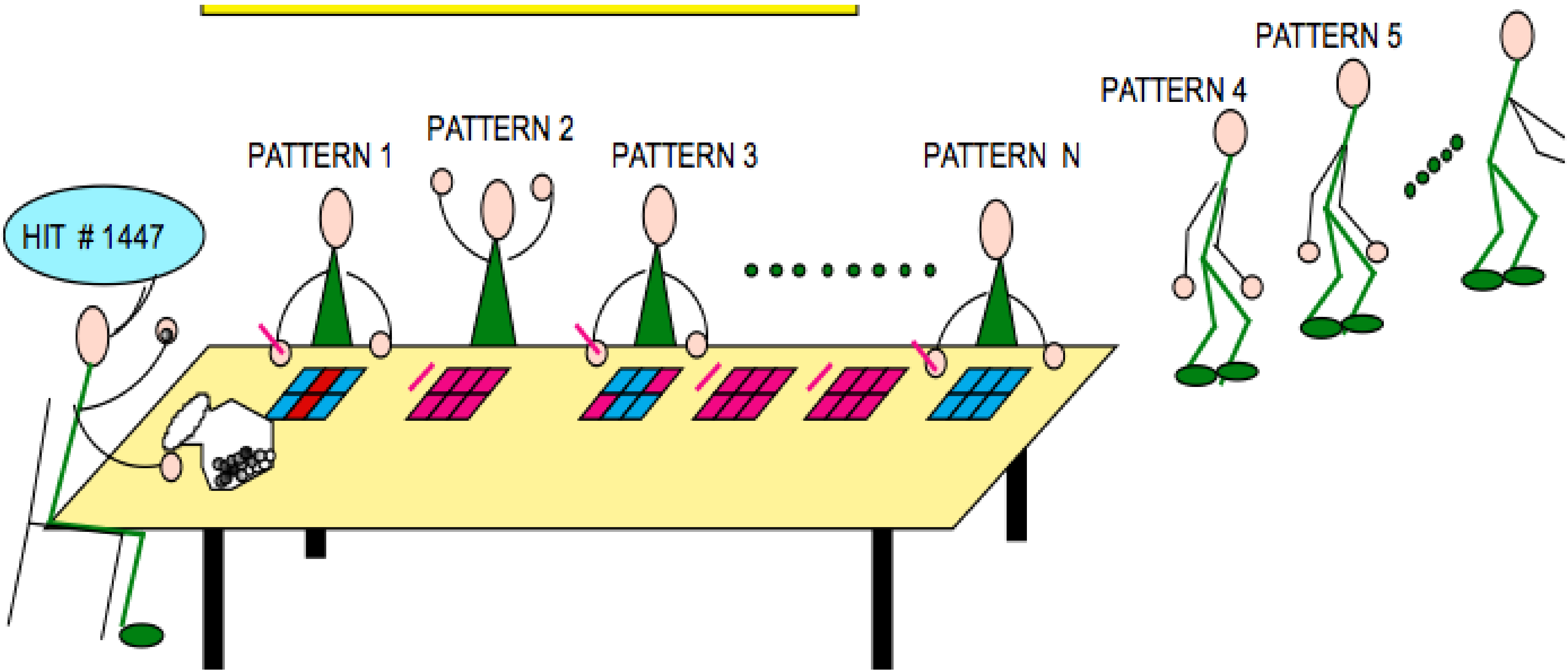
FTK magic

- Associative Memory (AM) ternary content-addressable memory (custom solution)



- Majority logic: check if all or all but one layeers are matched (FF in fired)
- Increases efficiency
- Fischer tree: Output the addresses of matched patterns

Let's play ~~bingo~~ FTK!



FTK magic

- Associative Memory (AM) ternary content-addressable memory

RAM

0xA	1	0	1	1
0xB	0	0	1	1
0xC	0	1	1	1
0xD	0	0	0	1

Output: 0011

Address: 0xB

CAM

0xA	1	0	1	1
0xB	0	0	1	1
0xC	0	1	1	1
0xD	0	0	0	1

Output: 0xB

Search data: 0011

Ternary CAM

0xA	1	0	1	1
0xB	0	0	X	1
0xC	0	1	1	1
0xD	0	0	0	1

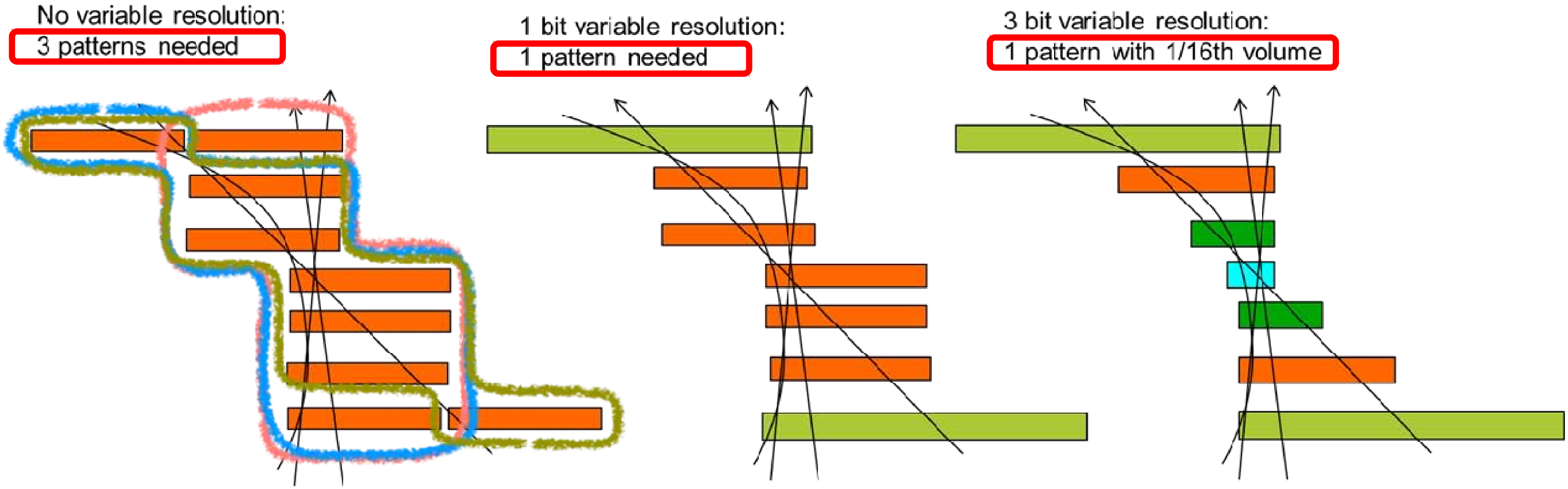
Output: 0xB

Search data: 0011

- Ternary CAMs introduce “Don’t Care” bit, X.
- Allow to search for a match regardless of the specific bit value.
- The longest explicit match is output.

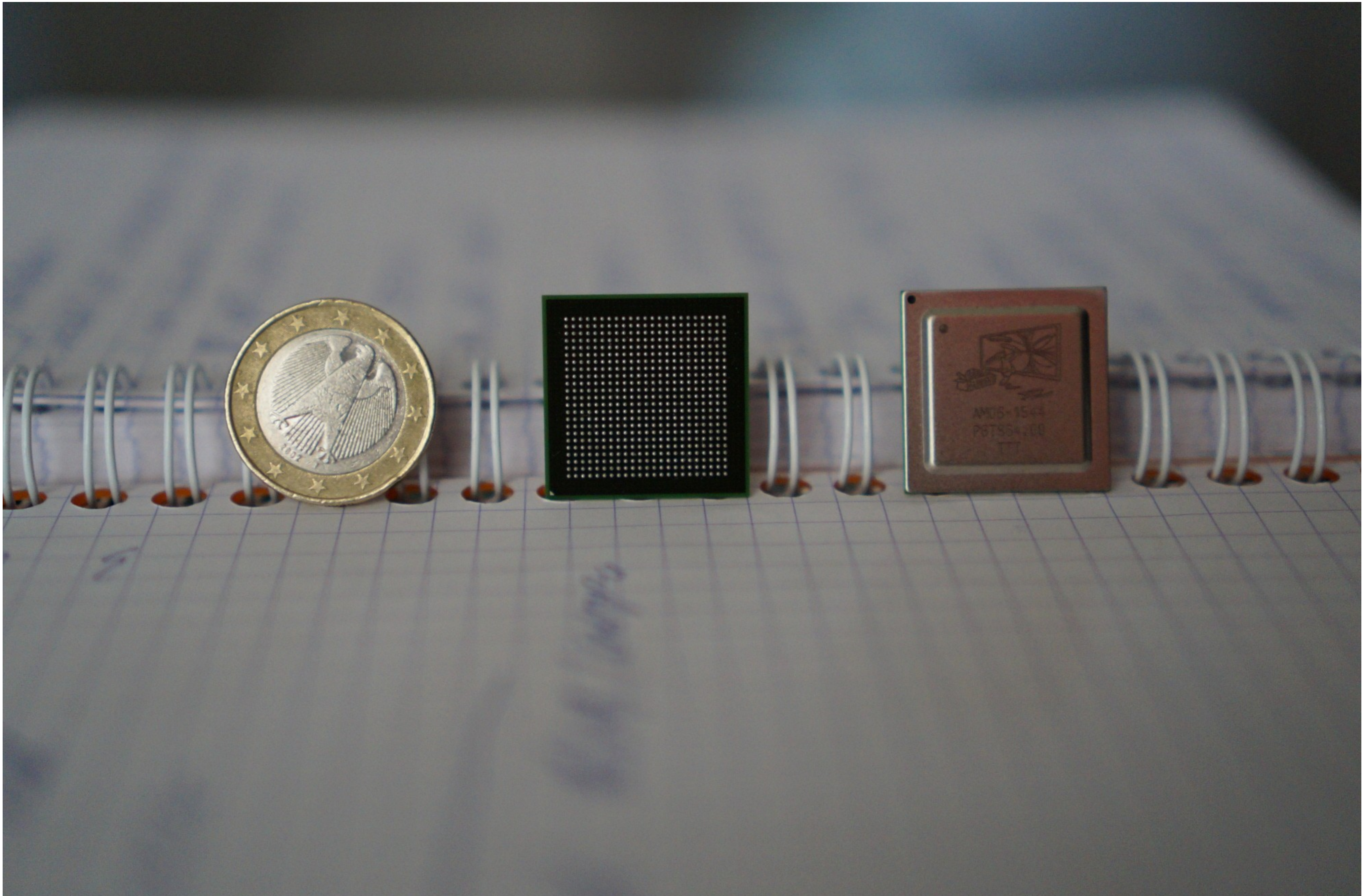
FTK magic

- Associative Memory (AM) ternary content-addressable memory



- DC bits allow to define patterns of variable shape (multiple clusters match a single pattern in a layer)
- Up to 6 DC bits / layer.
- Significantly increases effective pattern-bank size, keeps fake rate low.
- Example: 2 DC bits correspond to 3-5 increase in the effective number of patterns, but increase the size of the chip by 17% only.

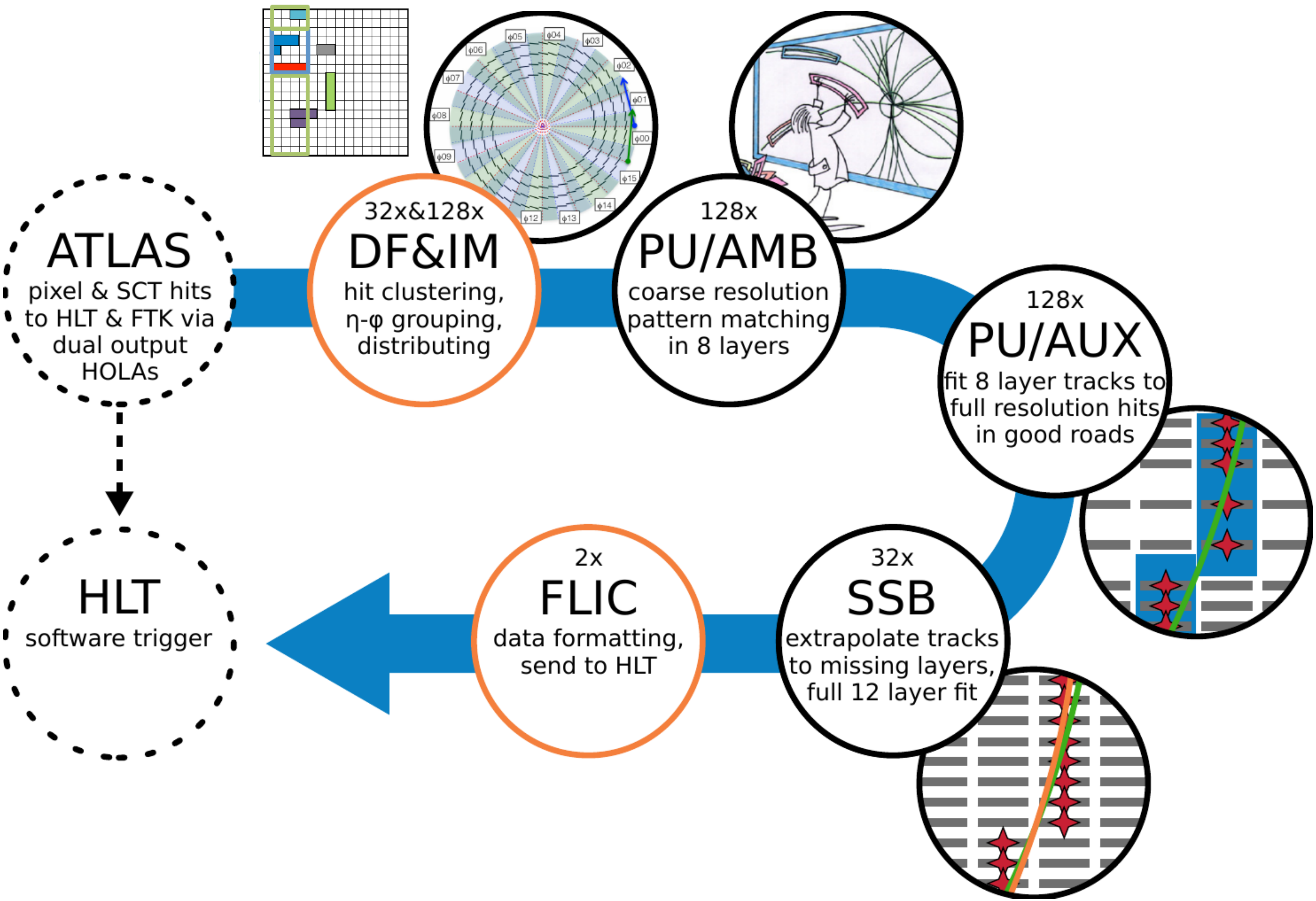
FTK reality

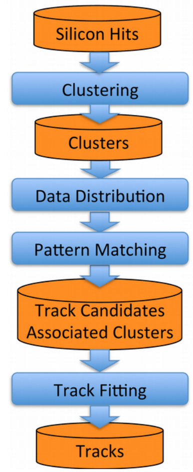


What did we learn (so far)

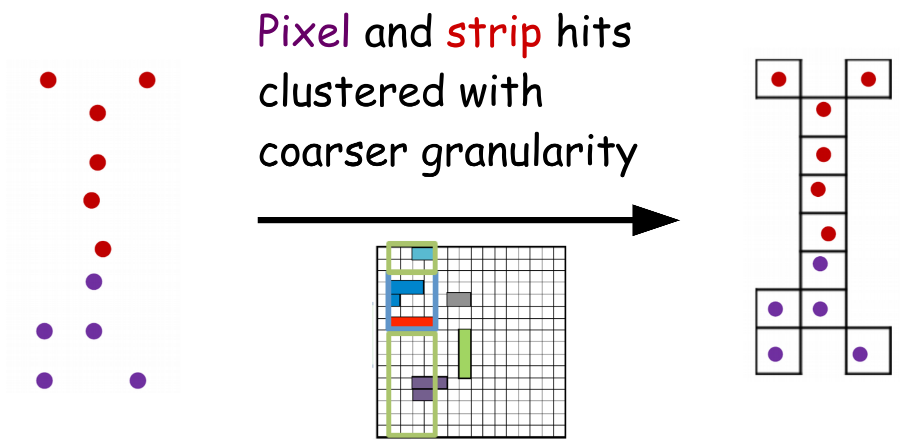
- Tracking is extremely valuable and used extensively in physics analysis, but it is also needed in trigger to be able to record interesting events.
- At the moment, ATLAS trigger system makes only very limited use of tracking.
- Fast TracK trigger will do full tracking and provide inputs to High Level Trigger.
- Use content-addressable memory to do fast track pattern recognition.

FTK processing steps

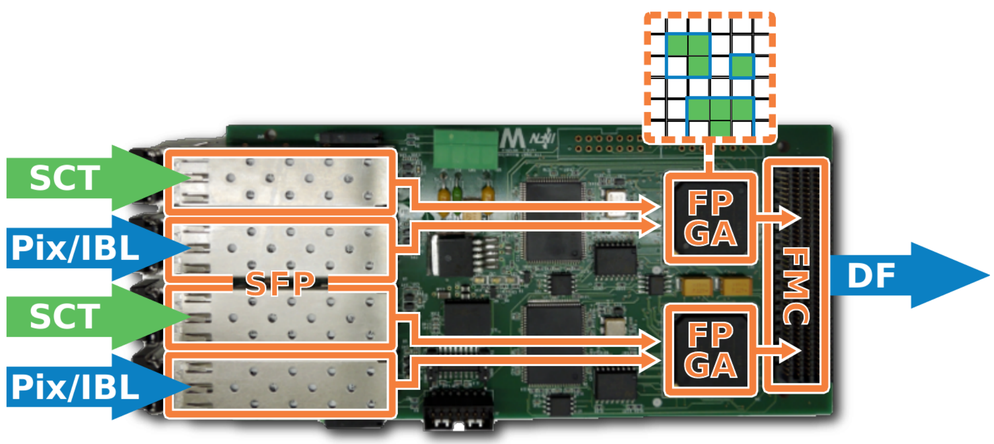




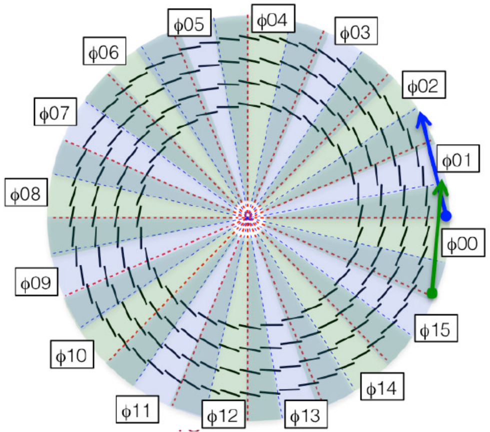
Hit clustering



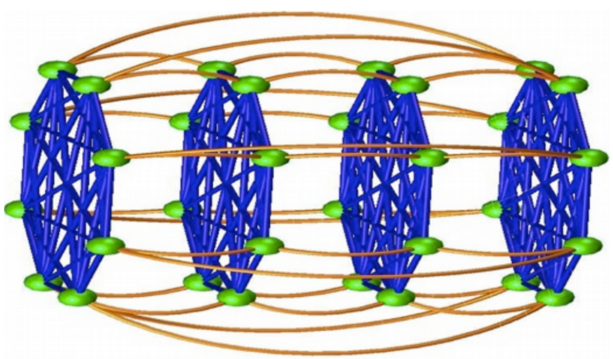
- IM board: 2 FPGAs / board; 1 Strip + 1 Pixel input per FPGA
- 128 IM boards, Spartan6 and Artix7 FPGAs
- 2 Gb/s input
- Sliding-window algorithm to find clusters of neighbouring pixels/strips
- Typical cluster size: 2x3 pixels, 2-3 strips
- 4 parallel clustering threads on each FPGA

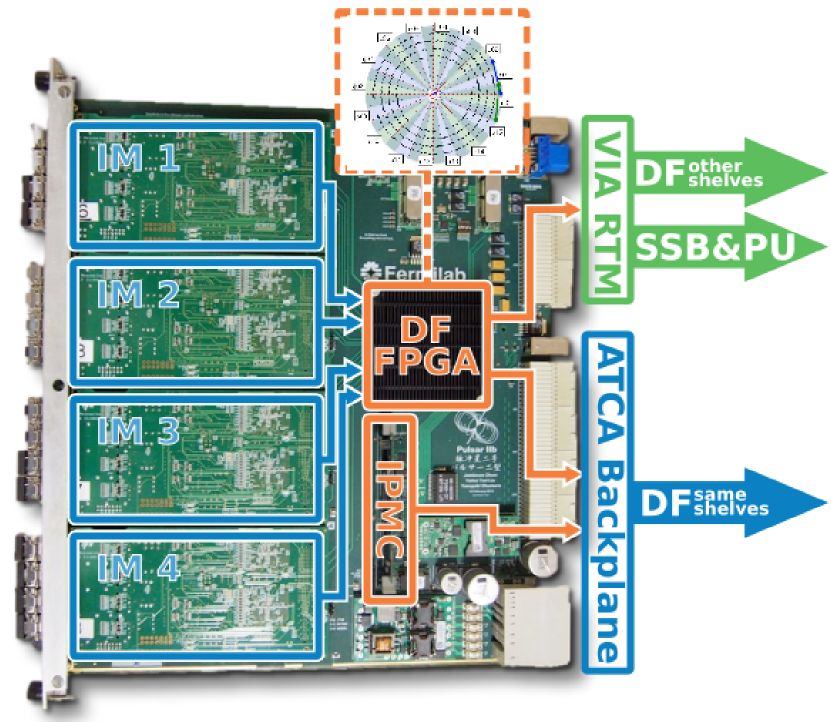


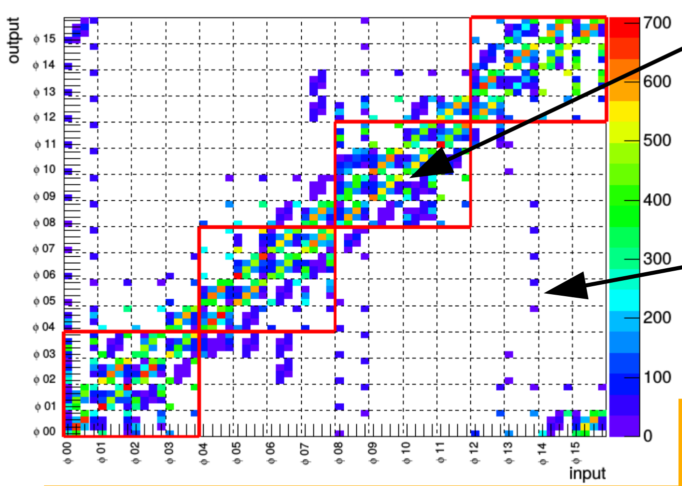
Data distribution



- Data Formater (DF): 1 FPGA(Xilinx Virtex7)/board; 4 IM's/DF
- 32 DF boards in 4 crates
- Parallel system:
  - distribution of hits within 4 η x 16 φ regions
  - massive interconnection between boards



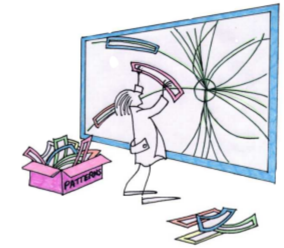


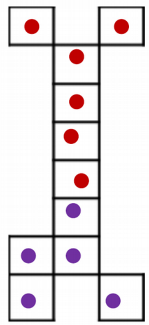


Data distribution within a crate

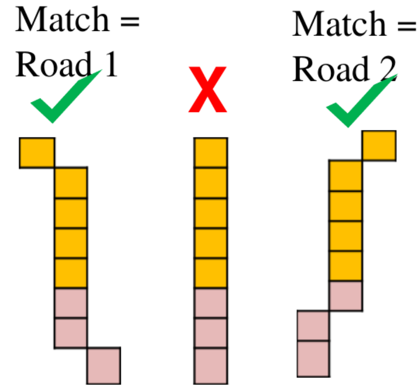
Data distribution between crates

Pattern recognition



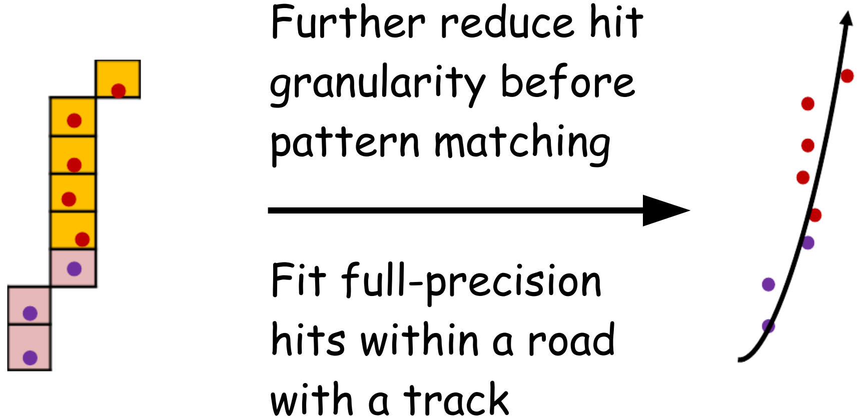


Matched patterns are called "roads"



- AM board: 2 FPGAs / board + 64 AM chips
- 128 AM boards, 8000 AM chips
- 128k pattern/chip => 1 billion patterns in total
  - (CDF: 5k patterns/chip => 8M patterns in total)
- Pattern matching @ 100 MHz
- 2 AM boards / 1 (η,φ) region = 16 M patterns / region
- 8 layers are used as input (3 Pixel + 5 Strips)
- 3.6 W / chip => 29 kW for the whole system
  - needs advanced air cooling

8-layer track fit

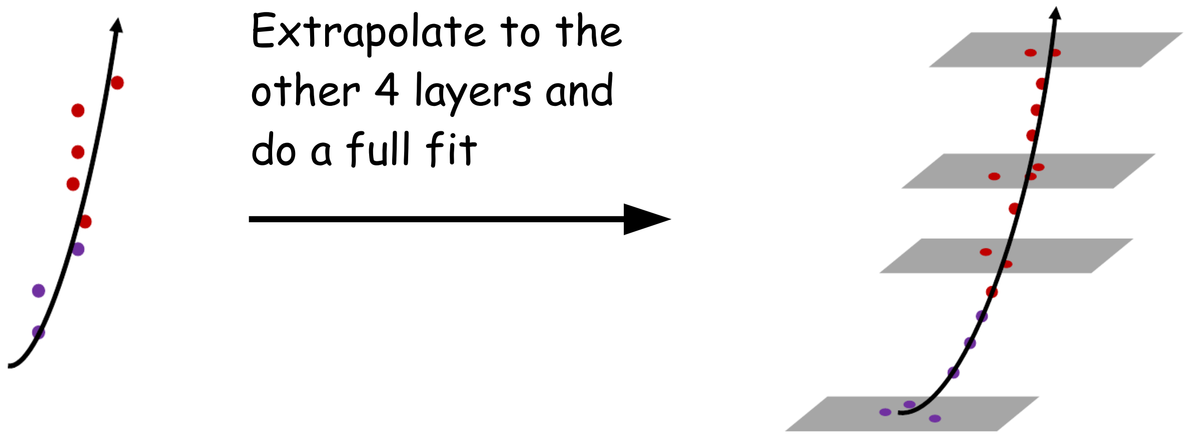


- <u>AUX</u> board: 4+2 FPGAs (Arria V)
- 128 AUX boards, each doing
  - data preparation for AM board
  - fit "<u>good</u>" tracks using roads + associated hits
  - removal of track duplicates (locally,shared hits)
- Fast track fit is achieved linearising the fit
- 512 FPGAs doing 5 fits / clock cycle @ 200 MHz
  - => 2.5 trillion fits / s
- ~1% roads end up in tracks

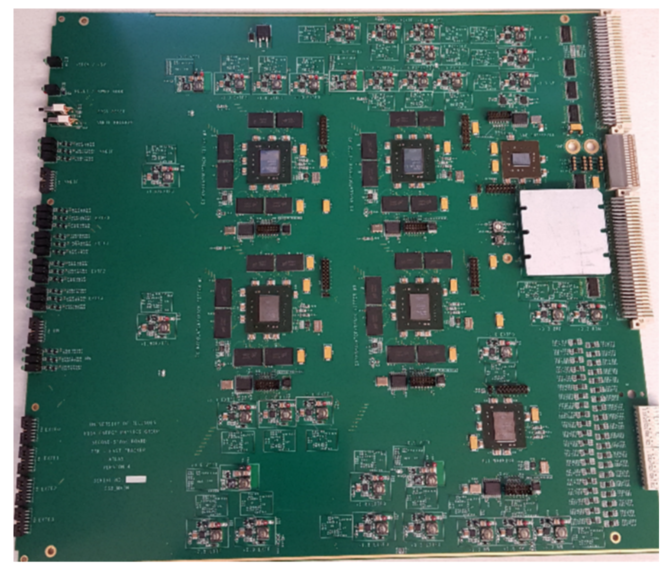
<u>Track parameters</u>, hit positions, pre-computed constants

$$p_i = \sum_j C_{ij} \cdot x_j + q_i$$

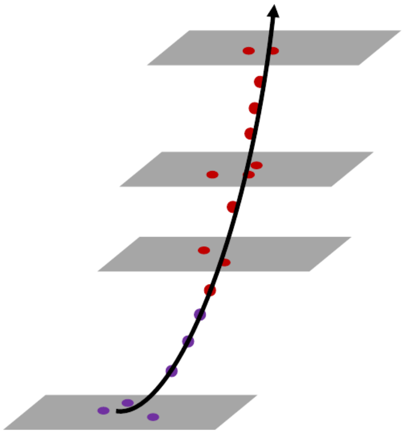
12-layer track fit



- Second Stage board: 4+2 FPGAs (Kintex 7)
- 32 SSB boards, interconnected with each other:
  - similar to AUX functionality
  - seeded by 8-layer tracks
  - fast track fit is achieved linearising the fit
  - overlap removal globally between (η,φ) regions
- Output tracks with final improved precision



Output collection and formatting



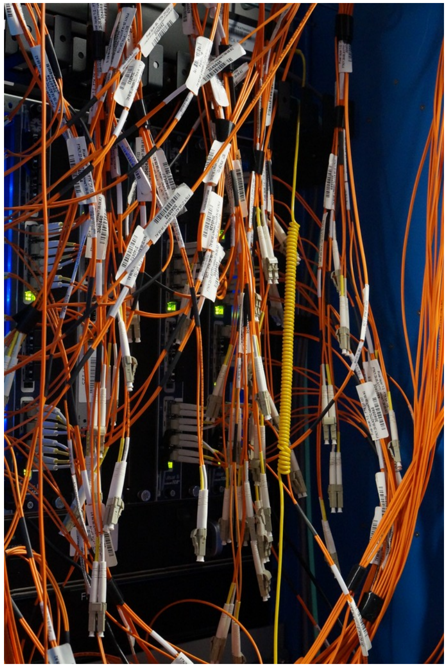
- FLIC board: 4 FPGAs (Virtex 7)
- 2 FLIC boards.
- Collect data from all SSBs and merge together
- Convert into the common ATLAS data format
- Strip/add monitoring information to the output packets
- 3 Gb/s output

What did we learn (so far)

- Tracking is extremely valuable and used extensively in physics analysis, but it is also needed in trigger to be able to record interesting events.
- At the moment, ATLAS trigger system makes only very limited use of tracking.
- Fast TracK trigger will do full tracking and provide inputs to High Level Trigger.
- Use content-addressable memory to do fast track pattern recognition.
- Fast tracking, latency of ~0.1 ms, in FTK is achived by:
    - several layers of dedicated hardware;
    - massively parallel structure of the system;
    - Associative Memory chips for pattern recognition + fast track fits in FPGAs;
    - simplifications in the fit procedure (linear approximation)

FTK commissioning status

- 2 FTK slices are installed in ATLAS cavern:
  - A: IM+DF+AUX (-> ATLAS output)
  - D: IM+DF+AUX+AMB+SSB+FLIC -> ATLAS output
- Slices are integrated with the common ATLAS TDAQ system. Slice A was regularly running during ATLAS data taking in fall 2016.
- Firmware is complete and being debugged with simulated events and real data.
- Board inter-communication was established and work ongoing on robustness.
- Comparison of FTK hardware output to FTK simulation ran on hits from ID.



12-layer FTK tracks

FTK commissioning status

- Commission individual boards and the full FTK chain.
- Get first FTK tracks written to the ATLAS system
- Install HW to cover full barrel region (16 / 128 AMB+AUX)

- Commission full barrel system.
- Commission HLT triggers using FTK tracks
- Install hardware to cover full detector ~40 pileup (64 / 128 AMB+AUX)

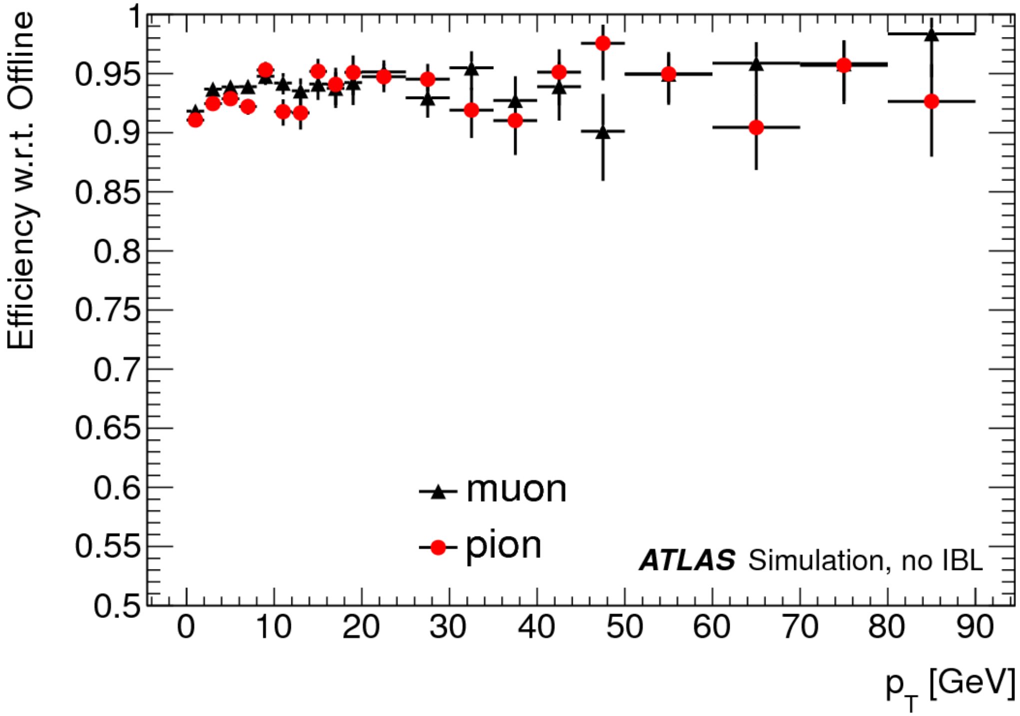
- Commission full ID detector
- HLT triggers based on FTK in data taking
- Full HW installed

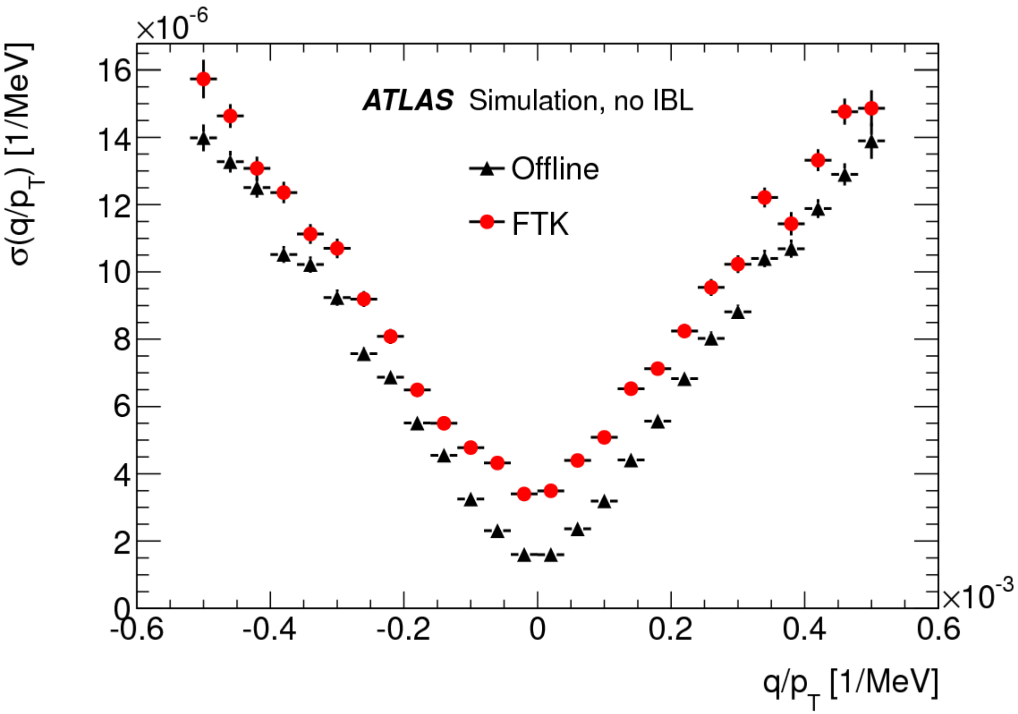
2016 → 2017 → 2018+

What did we learn (so far)

- Tracking is extremely valuable and used extensively in physics analysis, but it is also needed in trigger to be able to record interesting events.
- At the moment, ATLAS trigger system makes only very limited use of tracking.
- Fast TracK trigger will do full tracking and provide inputs to High Level Trigger.
- Use content-addressable memory to do fast track pattern recognition.
- Fast tracking, latency of ~0.1 ms, in FTK is achived by:
  - several layers of dedicated hardware;
  - massively parallel structure of the system;
  - Associative Memory chips for pattern recognition + fast track fits in FPGAs;
  - simplifications in the fit procedure (linear approximation)
- FTK is in active integration and commissioning. First inputs into HLT in 2017

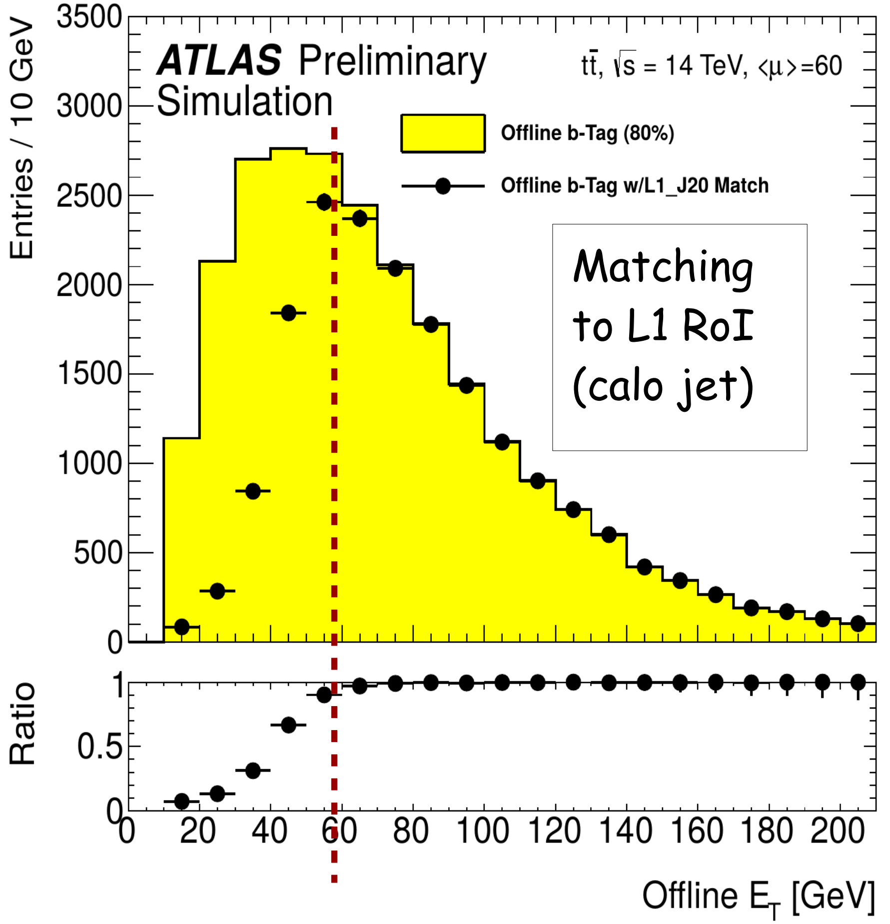
Expected FTK performance

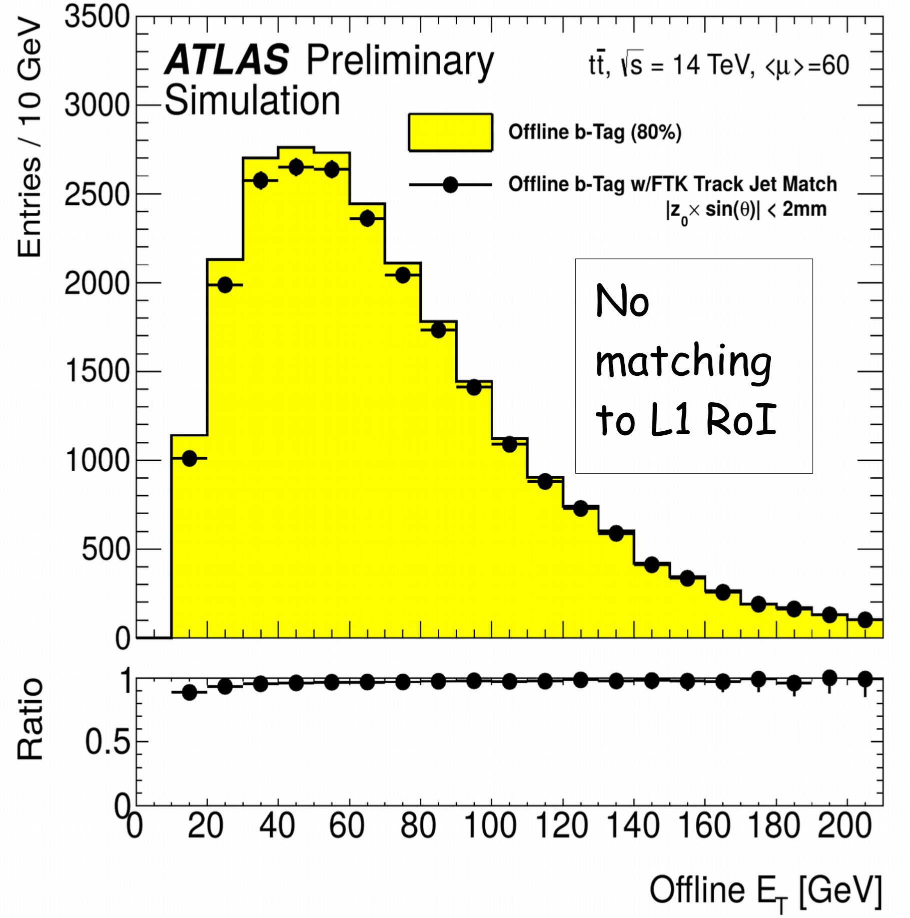




- Efficiency wrt offline tracking is higher than 90%
- Resolution in p_T is similar to offline
- Small difference are due to Pixel + Strip systems only (+TRT in Offline) and simplified clustering and track fitting

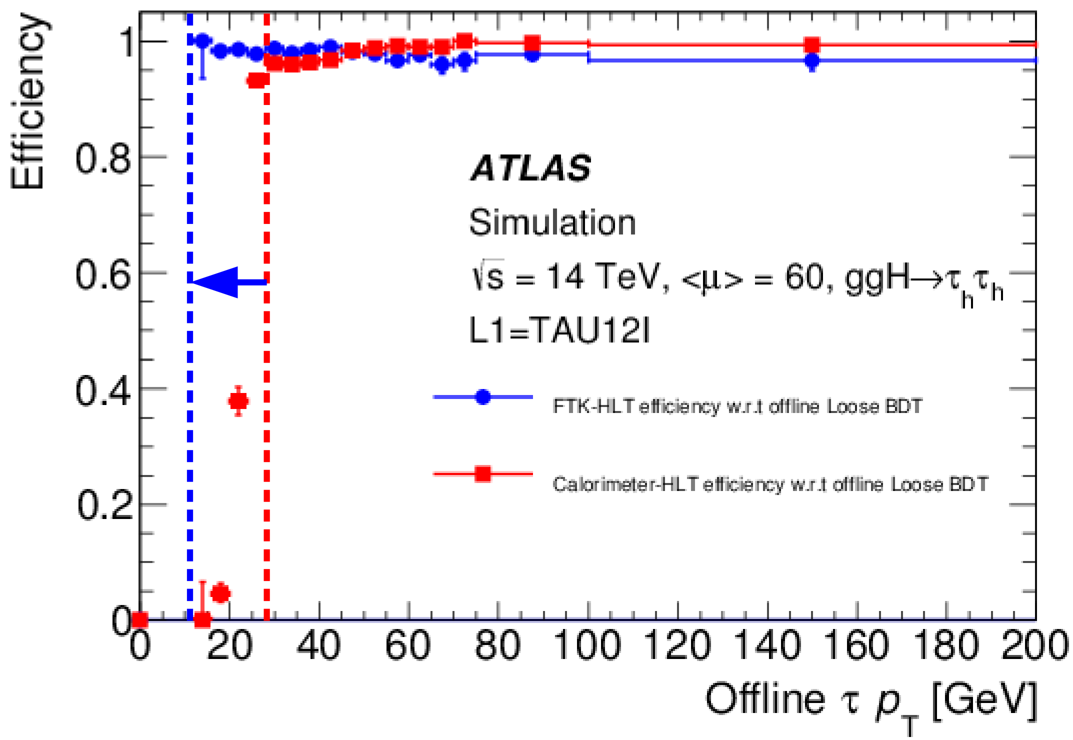
Expected FTK performance





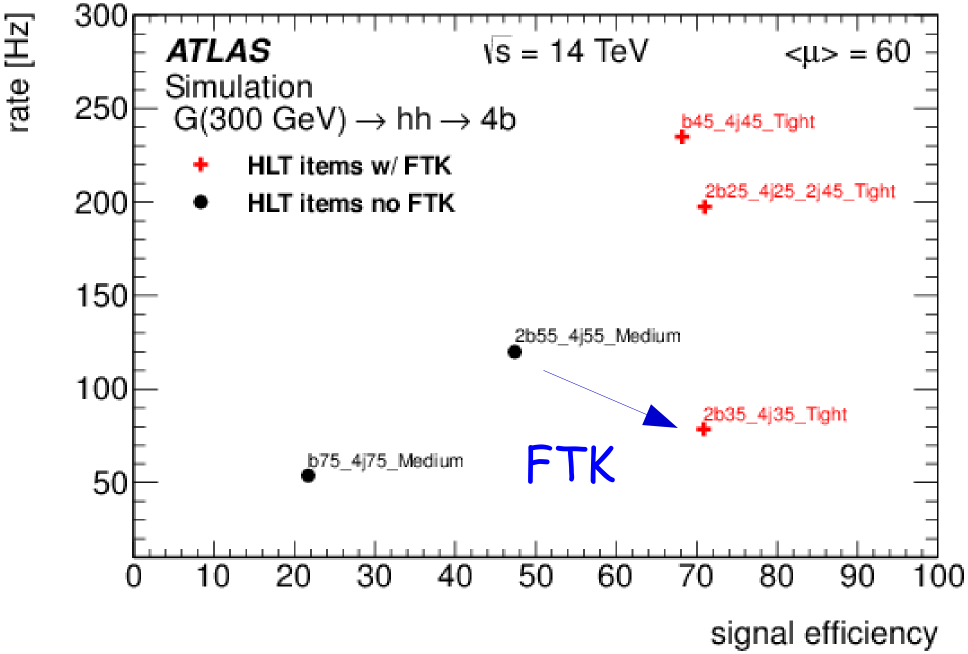
- Improved efficiency of b-tagging in HLT due to seedless jet finding

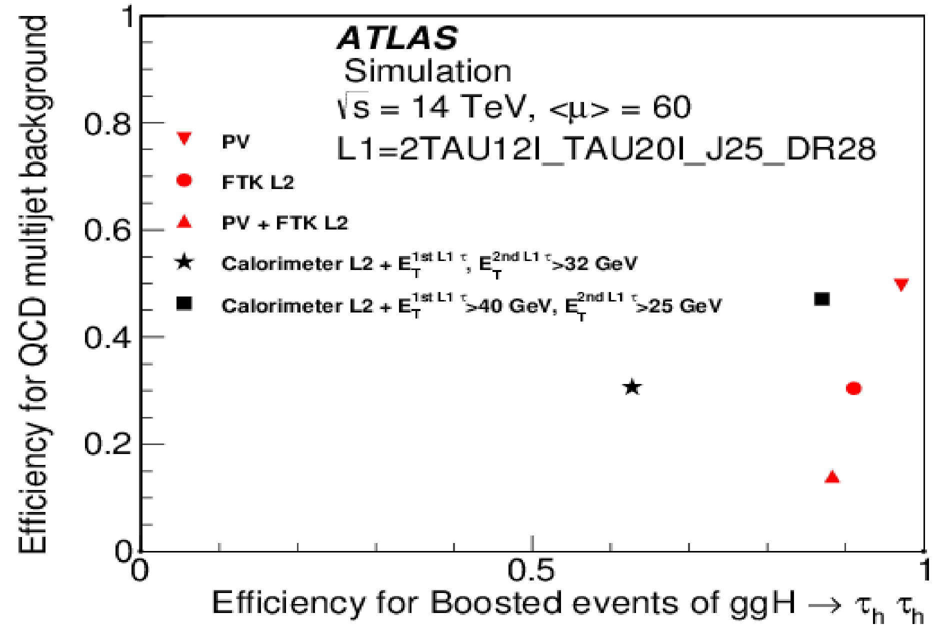
Expected FTK performance



- Reduced kinematic threshold due to trigger requirements for τ-tagging

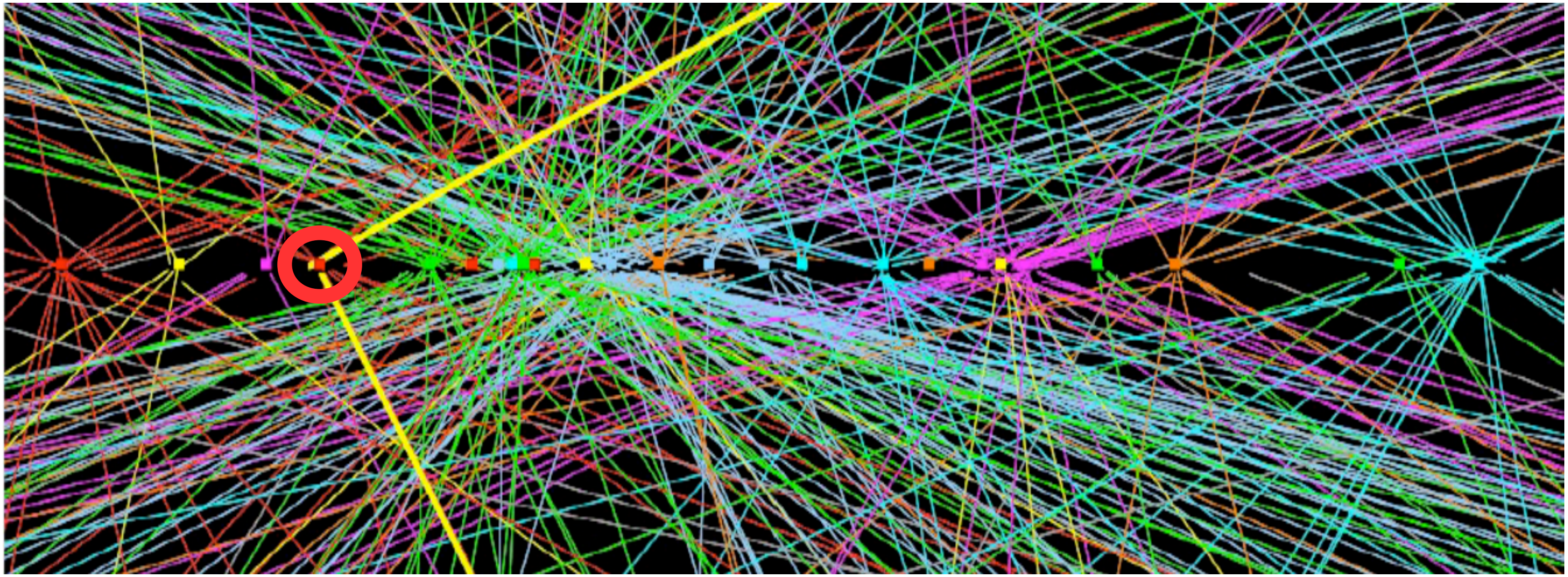
Expected FTK performance





- Significantly improved signal efficiency for triggers with multiple b- or τ-tags

Expected FTK performance



- Identify and veto the vertex that caused the L1 accept
- Other ~80 vertices are reconstructed by FTK without any L1 bias
- Unbiased physics in pileup .collisions
    - with effective 1/400 lumi due to L1 rate reduction
- Might be useful if the signature is hard to trigger on L1, have distinct tracking activity.

Summary

- Tracking is extremely valuable and used extensively in physics analysis, but it is also needed in trigger to be able to record interesting events.
- At the moment, ATLAS trigger system makes only very limited use of tracking.
- Fast TracK trigger will do full tracking and provide inputs to High Level Trigger.
- Use content-addressable memory to do fast track pattern recognition.
- Fast tracking, latency of ~0.1 ms, in FTK is achived by:
  - several layers of dedicated hardware;
  - massively parallel structure of the system;
  - Associative Memory chips for pattern recognition + fast track fits in FPGAs;
  - simplifications in the fit procedure (linear approximation)
- FTK is in active integration and commissioning. First inputs into HLT in 2017
- Simulated FTK shows comparable to offline performance and significant improvements in various signatures in triggers.
- 2017: "FTK: let's make ATLAS great again!"