Towards embedded data recostruction in HEP: the RETINA approach

Giovanni Punzi
University & INFN-Pisa

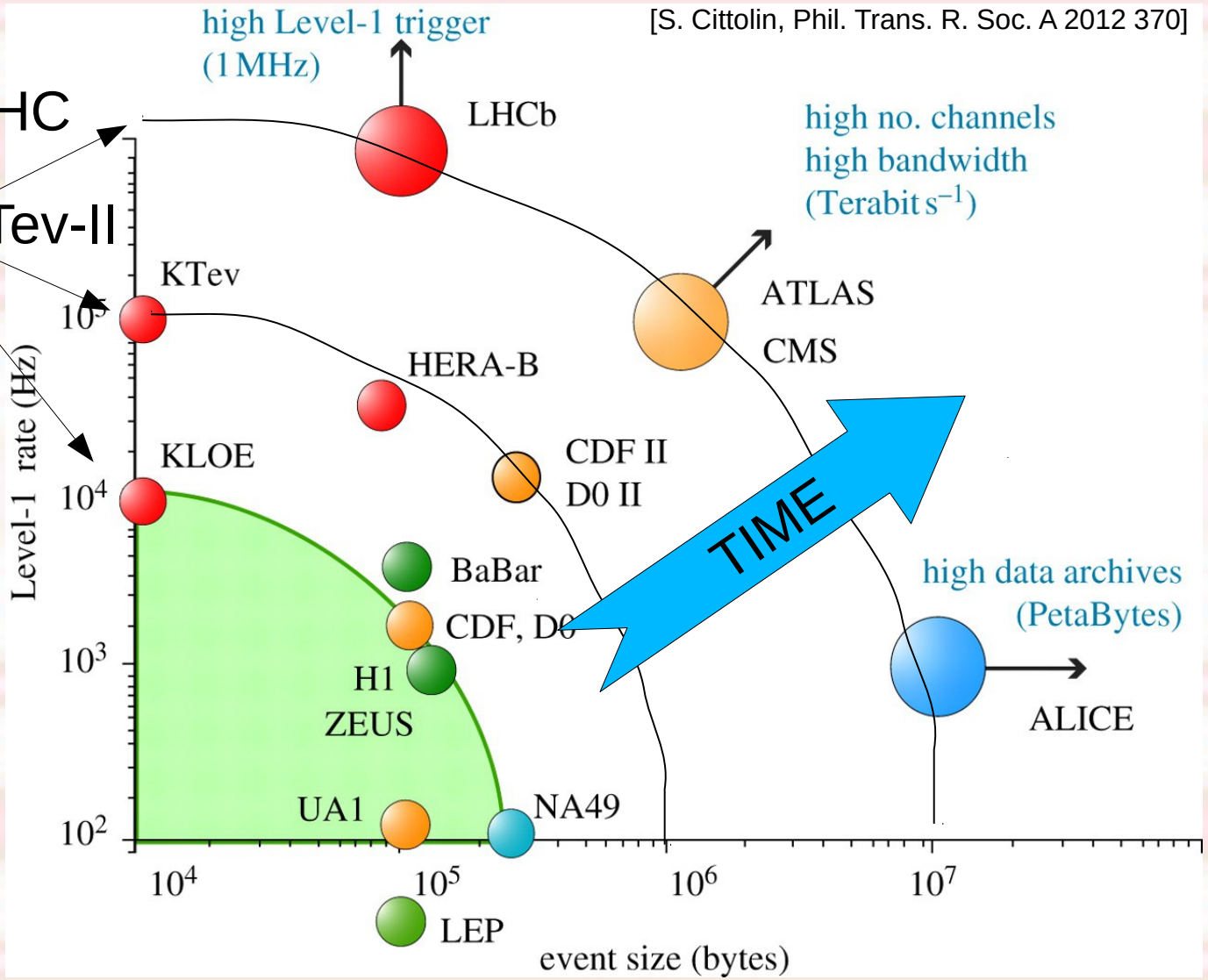
Giulia Vita Finzi Symposium
Rome, July 5th, 2016

Introduction

- LHC has opened a new era in HEP - also for data processing
- Exploitation of the upcoming High-Luminosity LHC phase will pose even greater challenges
- Data reconstruction and storage will be ***really tough*** issues
- Trigger, DAQ , Computation, Storage... have been part of HEP since its earliest days - complexity and computational load in-creases while electronics was having huge price/performance drops

But there is evidence that further progress will require bigger steps forward

Evolution of Data Processing in HEP



Some problems NOT getting easier with time

- In spite of increasing DAQ bandwidth and storage availability, the need for large data reduction factors to permanent storage keeps getting stronger.
- Evolution of computing not necessarily going as fast as in the past

Problem compounded by ***physics needs***:

- Precision measurements becoming more important
- Event structure more complex ("pile-up") even at constant rate.

→ Need **more computing** power to take the same decisions

e.g: CMS need to reduce data from the tracker to read it out...
LHCb has "signals" in every collisions...

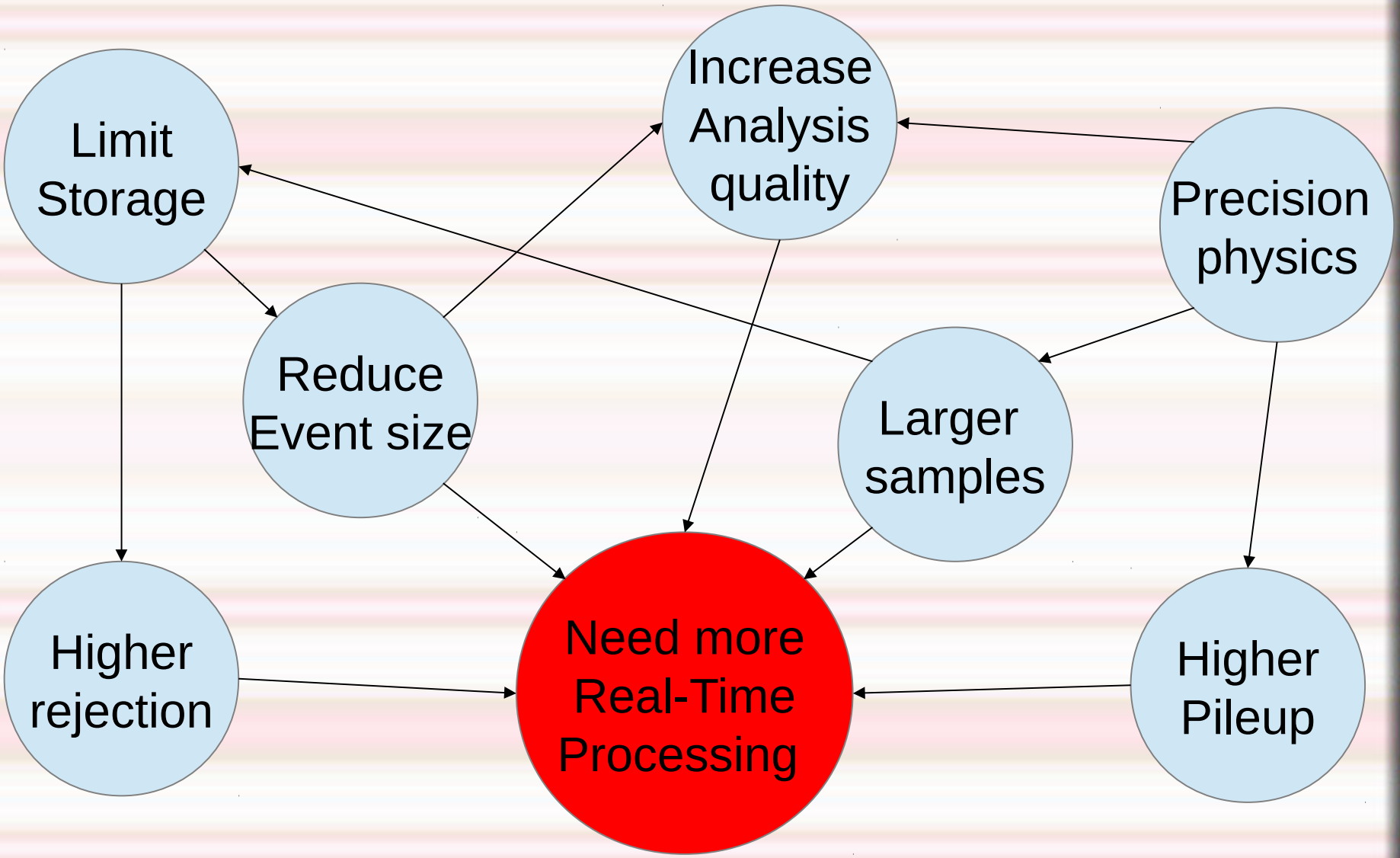
- In future, all SM physics will be "low-Pt physics"

At the FCC, the rate of top events will be 3kHz...

→ Need to feed ***more data*** into each decision

Implications for DAQ: much larger B/W into the trigger

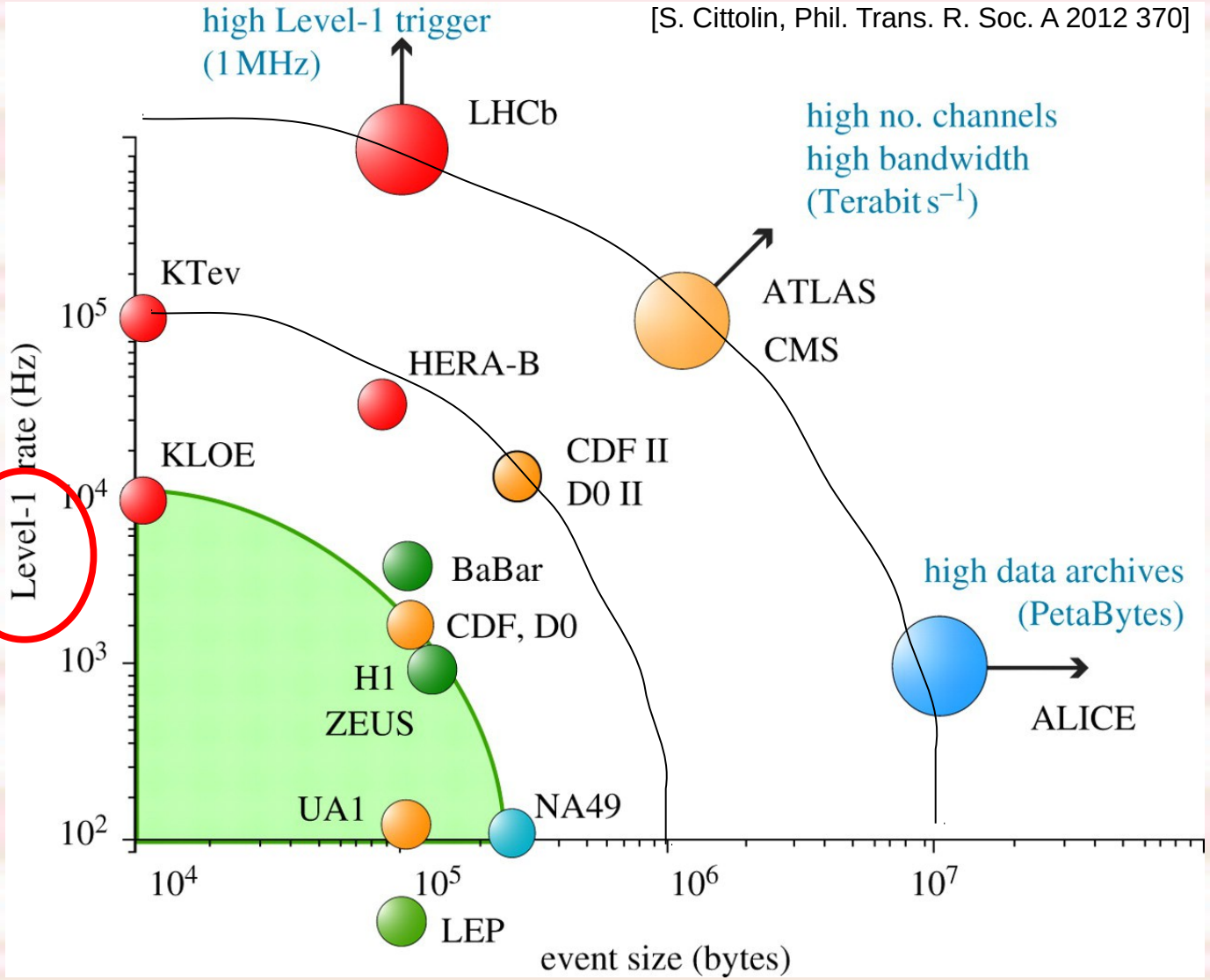
Pressures on Real-Time computing



Evolution of Data Processing in HEP

This is NOT
The full rate !
>10^2 reduction
by Level-1

Greater
challenge
in future



[S. Cittolin, Phil. Trans. R. Soc. A 2012 370]

The issue with the first level of processing

- It is “true real-time”: latency and local data availability requirements carry a weight
- Greater specificity, tighter optimization
- Less amenable to “plug-and-play” commercial solutions.
  - Requires larger development time, specialists
  - Less commonality with other solutions
  - Traditionally implemented in “hardware”
  - Now the distinction between hardware and software is much more blurred... electronics boards are typically completely programmable in software, although the software may be more application-specific
  - More than anything else, ***architecture matters***. Design is not made of procedures, but of structures (happening to general-purpose software as well, where increasing parallelization requires the programmer to think in terms of actual execution)

Personal view of future evolution of HEP

- Experiments will be limited by computing
- Large “commodity” computing will be used
- Physics reconstruction will happen mostly on-line
- Only a small fraction of events, and of data within an event, will be saved
- Calibration will need to be completely done on-line

First-level processing will evolve into “detector-embedded” re-construction of complex primitives - that will make the rest of computation manageable.

A tracking detector will need to produce TRACKS, not HITS. This will al-low the large online CPU farms to use their computing power to do more intelligent things, running more sophisticated algorithms.

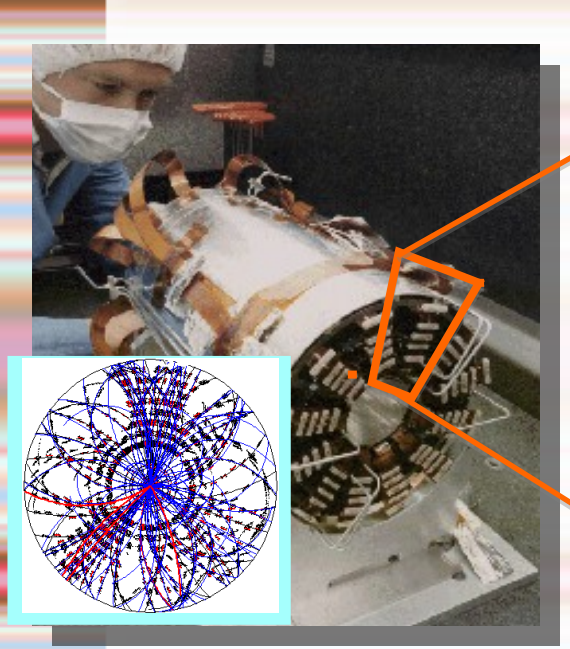
This will be the focus of the rest of my talk.

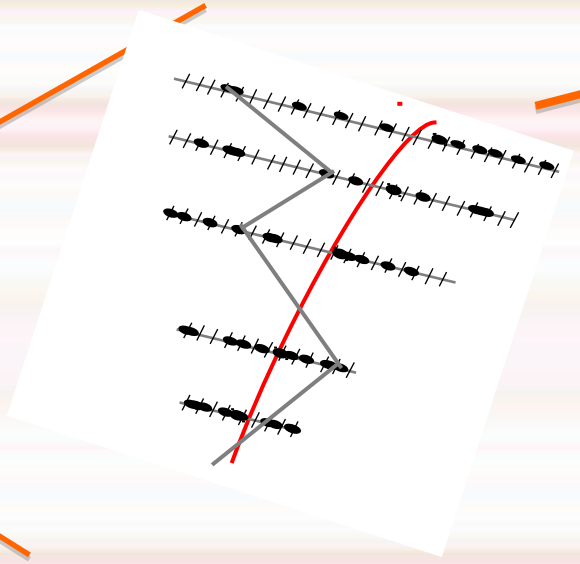
Tracking by pattern-matching

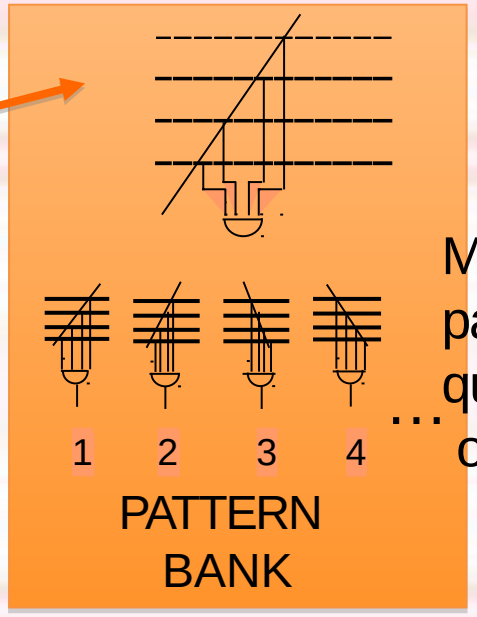
- The fastest approach to tracking that has been used up to now is direct matching to a bank of stored templates
- First large system to use this method has been CDF, at the Tevatron, where a real-time processor named SVT was capable of reconstructing quality tracks in ~10μs.
- Based on custom ASICs implementing content-addressable memory (Associative Memory [NIM A278, (1989), 436-440])
- It actually worked ! Allowed CDF to discover Bs oscillations (amongst other things)
- This same approach is continuing in FTK for ATLAS and in the planned Phase 2 upgrade for CMS

Track reconstruction by pattern-matching using "Associative Memory"

A *pattern* is a sequence of hits in the different layers, represented by coordinates. A particle trajectory is a specific sequence of hits. Hit are read out sequentially, and compared **in parallel** to a set of pre-calculated "track patterns" - ***NO combinatorics***.

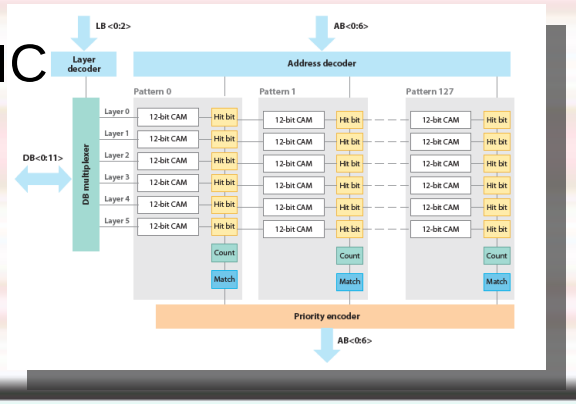






Matched patterns queued to output.

Based on custom ASIC



Track parameters found in a 2nd step

(more sequential, but fast if you used enough AM cells in the first stage)

Successful past examples of real-time tracking by pattern-matching

Name	Tech.	Exp.	Year	Event rate	clock	cycles/event	latency
XFT	FPGA	CDF-L0	2000	2.5 MHz	200 MHz	80	<4µs
SVT	AM	CDF-L2	2000	0.03 MHz	40 MHz	~1600	<20µs
FTK	AM	ATLAS-L2	2015	0.1 MHz	~200 MHz	~2000	O(10µs)

Compare with the requirements of a L0@LHC:

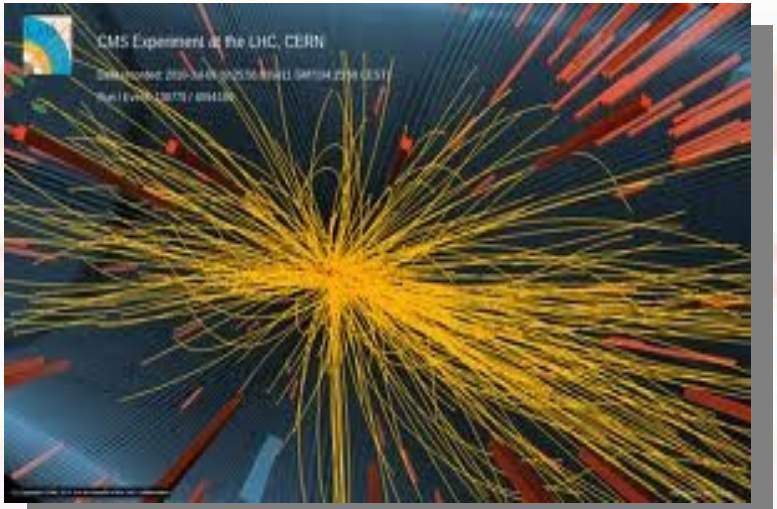
?	***?***	***LHC-L0***	***~2020***	***40MHz***	***~1GHz***	***~25***	***few µs***

- The task of L0 tracking at LHC appears daunting despite the progress of electronics.
- Any complex tracking calls for O(10^3) clock cycles/event in latency and throughput (still much faster than CPUs)
- No known example of a system making non-trivial pattern reconstruction in O(25) time units

Maybe just an impossible task ?

Inspiration from "Natural computing": comparing natural vision with HEP

VISION

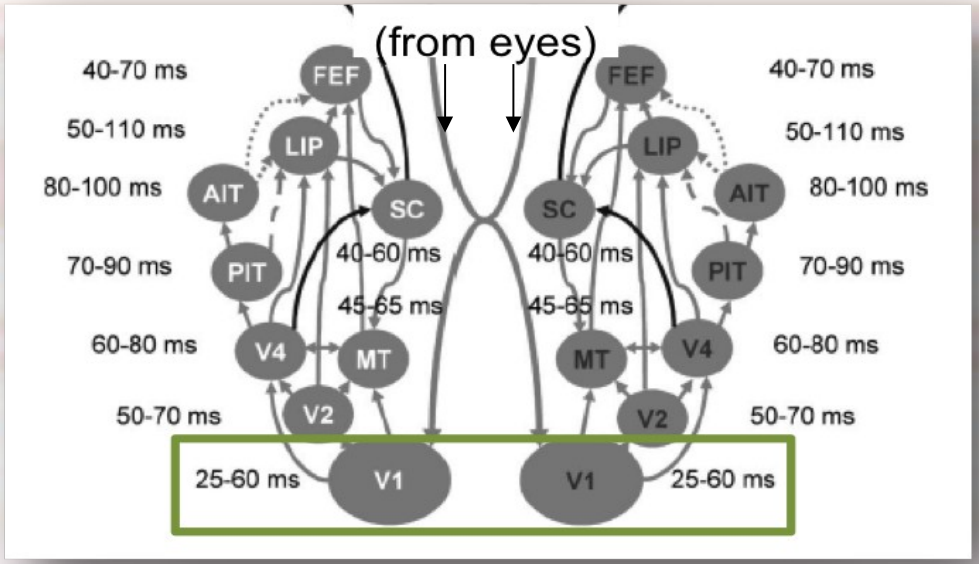


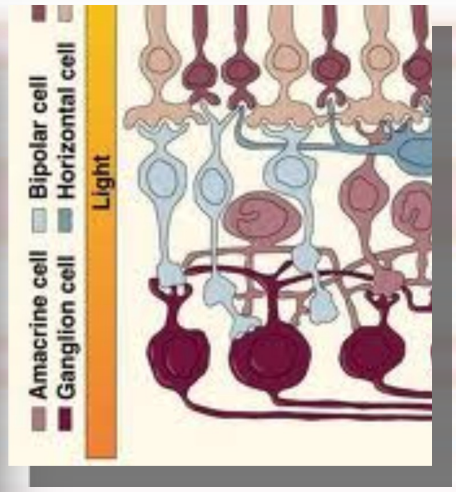
HEP

Many similarities:

- Lots of complex data/combinatorics
- Little time available
- Pressure to make accurate decisions
- Strongly constrained computing resources

A look at size and timing of the natural vision system





$9.2X10^7$ Rods +
$4.6X10^6$ Cones

20 Gb/s

$1X10^6$ Optic Nerve fibers

0.8 Gb/s + 4 Gb/s

TOTAL ~5Gb/s

- The early visual areas in human brain produce a recognizable sketch of the image at 30-40Hz, with latencies <100ms

$>10^9$ neurons for vision, typical switching time ~1ms.

Performance: Natural vs Man-made

Name	Tech.	Exp.	Year	Event rate	clock	cycles/event	latency
SVT	AM	CDF-L2	2000	0.03 MHz	40 MHz	~1600	<20µs
FTK	AM	ATLAS-L2	2014	0.1 MHz	~200 MHz	~2000	O(10µs)
Vision	*(neural)*	*(Brain)*	*old*	*~40 Hz*	*~1kHz*	*~25*	*<100ms*

- Complex tracking calls for O(10^3) clock cycles/event (both in latency and throughput) – Vision works within just ~25
- If we could do the same in an electronics device, we could easily do **real-time tracking of every LHC collision**: 25 cycles@1GHz → 25ns : 40MHz
- The scaled flow of data would be **5 Pb/s –** enough for a huge detector

Brain outperforms HEP triggers greatly - WHY ?

What is so special about the "brain algorithm" ?

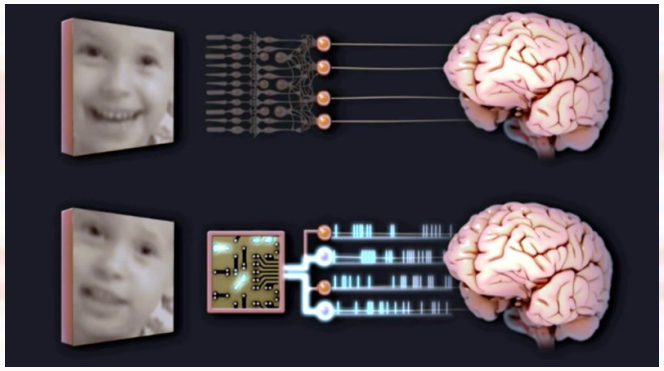
- Parallelism, of course - but Associative Memories are very parallel devices as well...
- Some important differences, though:
  - Hit processing in AM cells still happens serially, while in the visual system only relevant data reaches a cell. This is faster, and allows processing power to be **spread over a network**.
  - The AM has "rigid templates" with yes/no response, while the brain works by **interpolation of analog responses**. This saves internal storage and makes it easier to deal with "missing information".

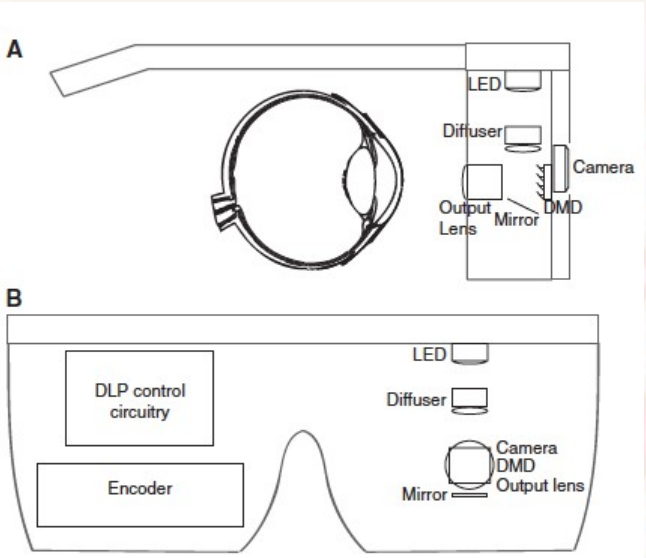
Could these features be implemented in a viable artificial device ?

Investigating these questions is the goal of the "RETINA project"

One-slide digression: Sheila Nirenberg's retina encoder

www.pnas.org/cgi/doi/10.1073/pnas.1207035109





OMAP35x Processors

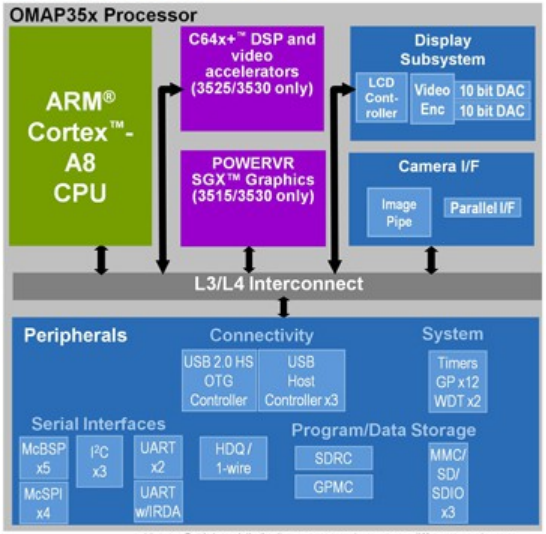
Laptop like performance at handheld power level

Performance

- High-performance Superscalar ARM® Cortex™-A8 featuring NEON co-processor with immersive 2D/3D Graphics accelerator
- HD video decode utilizing TMS320C64x+ DSP and video hardware accelerators
- Low power utilizing TI's SmartReflex™ technology with option for integrated and discrete Power Management ICs

Features

- Cores
  - Cortex A-8 with NEON™ SIMD Coprocessor / DSP-based TMS320C64x+ DSP and video accelerators (max performance only)
    - 720 MHz / 520 MHz @ 1.35V
    - 600 MHz / 430 MHz @ 1.35V
    - 550 MHz / 400 MHz @ 1.27V
  - 2D/3D Graphics Engine - Up to 10M polygons per second
- Memory
  - ARM:
    - 16 kB I-Cache; 16 kB D-Cache; 256kB L2
  - TMS320C64x+ DSP and video accelerators
    - L1 32kB Program Cache/32kB Data Cache + 48kB SRAM
    - L2 64kB Program / Data Cache + 32 kB SRAM; 16 kB ROM
  - On Chip: 64kB SRAM; 112kB ROM
- Peripheral Highlights
  - Support for LPDDR
  - Support for NOR, NAND, SRAM, Pseudo SRAM
  - USB 2.0 HS Compliant OTG Controller w/ 2 additional USB Host Controllers
  - Display subsystem with LCD and TV interface. Supports PIP, color space conversion, resize and rotation.
  - Camera I/F with CCD controller and Image-pipe (Preview, Resize, Statistics
- Package 1 (CBB): 12x12 mm, 0.4mm pitch, Package On Package (POP); 515 pin PBGA; production now; can be used with discrete memory
- Package 2 (CUS): 16x16 mm 0.65 mm pitch. 423 pin PBGA; production now. Utilizes Via Channel™ Array Technology with 0.8mm pitch plus design rules.
- Package 3 (CBC): 14x14 mm, 0.5 mm pitch POP; 515 pin PBGA; production now; must use POP memory



Notes: Peripheral limitations may apply among different packages

POWERVR SGX™ 3D engine is licensed from Imagination Tech. Ltd.

Customers considering the CBC package should secure POP memory supply before designing with this solution

- Functionality of retinal circuitry was measured and replicated in standard digital devices
- Application to vision prostethic being developed
- *Different from our purpose* – but suggestive

The RETINA project

https://web2.infn.it/RETINA

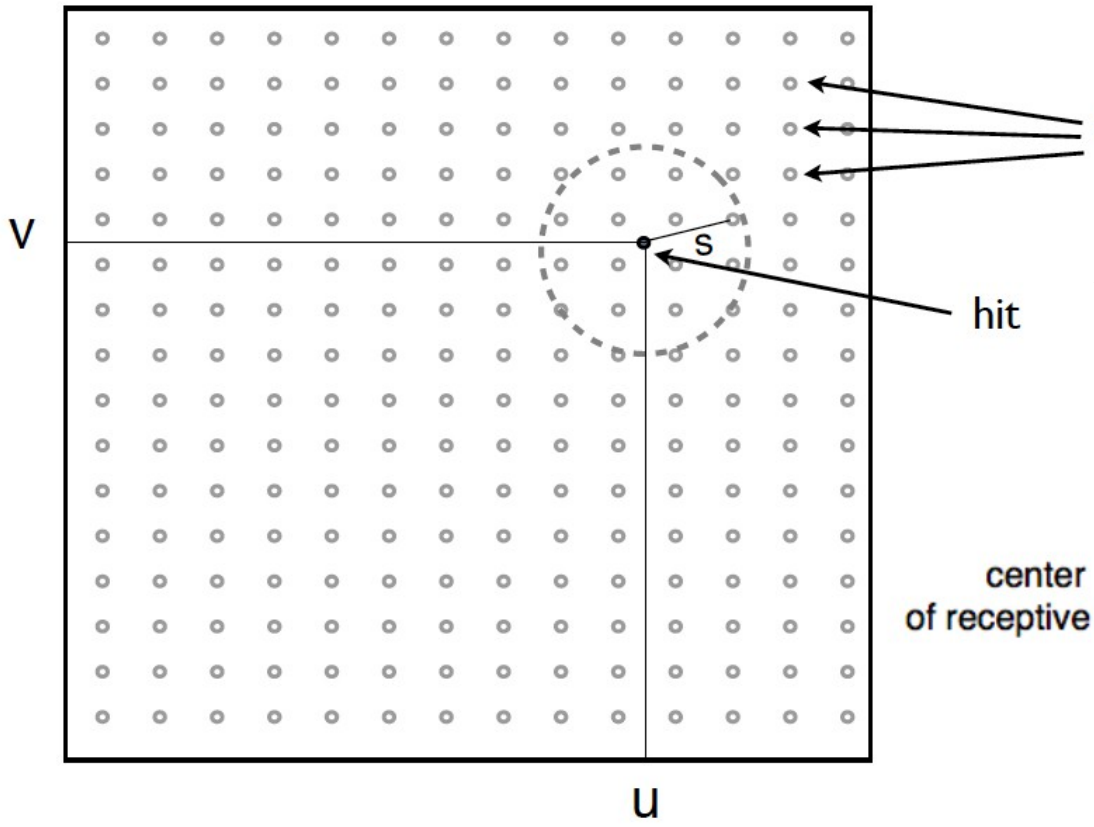
- R&D program supported by INFN CSN5 (Technological research division)

- Goal: study the possibility to build a specialized track processor based on a vision-like architecture and evaluate its performance for tracking in LHC environment

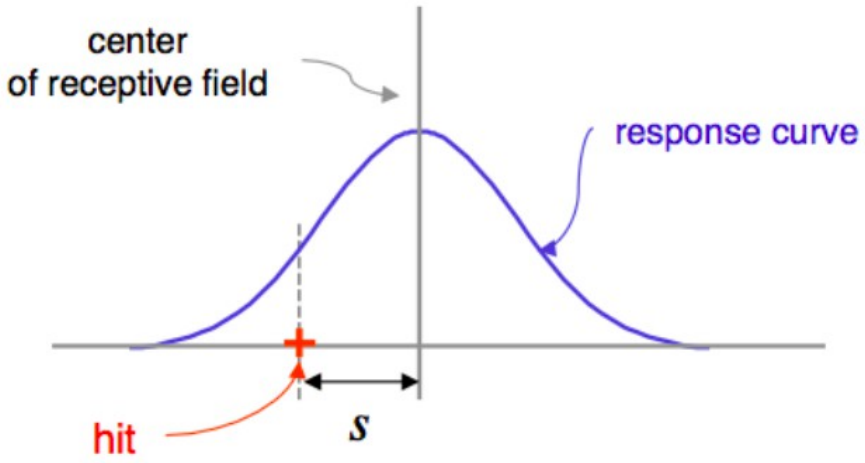
- Specialization is important: the success of GPUs stems from specialization for a narrow purpose. Our aim is to build something that does for Tracking what the GPU did for Graphics (just with a smaller market...) (a “TPU”).

- Not intended to replicate vision in detail - just exploit similar design principles.

Implementing a "neural-like" tracking algorithm



Each cell performs a weighted sum of hits "in the vicinity" ("graduated response").

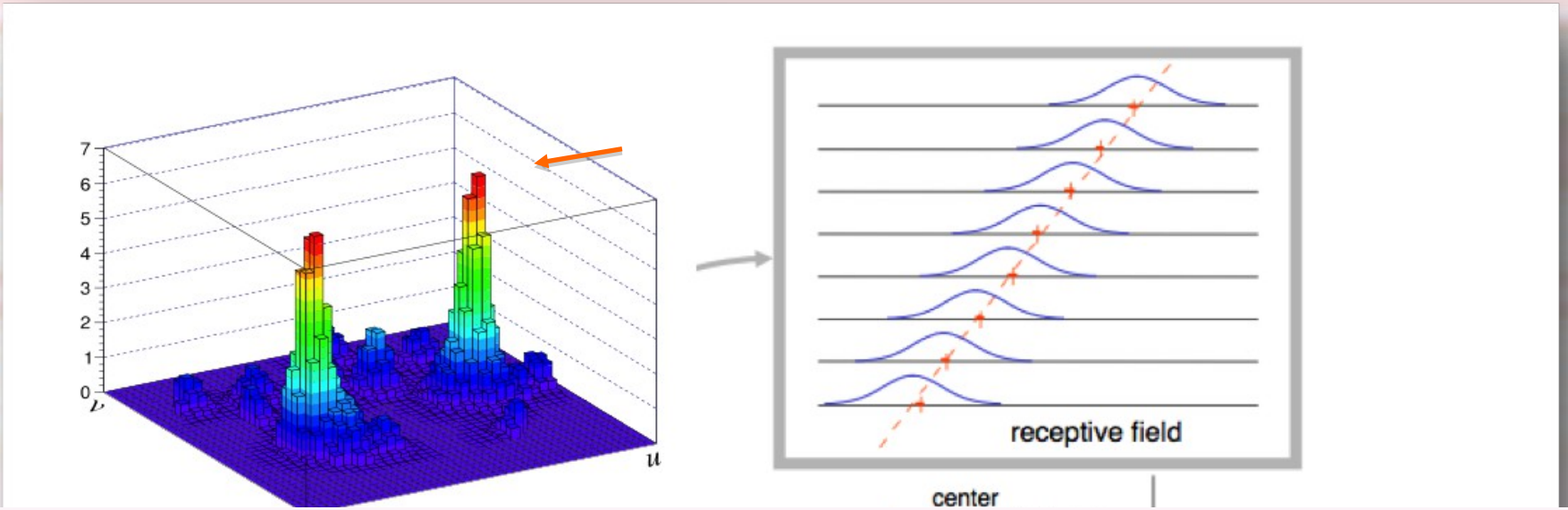


Response of each cell is summed over all hits

$$R = \sum_{all\ hits} e^{-\frac{s_i^2}{2\sigma^2}}$$

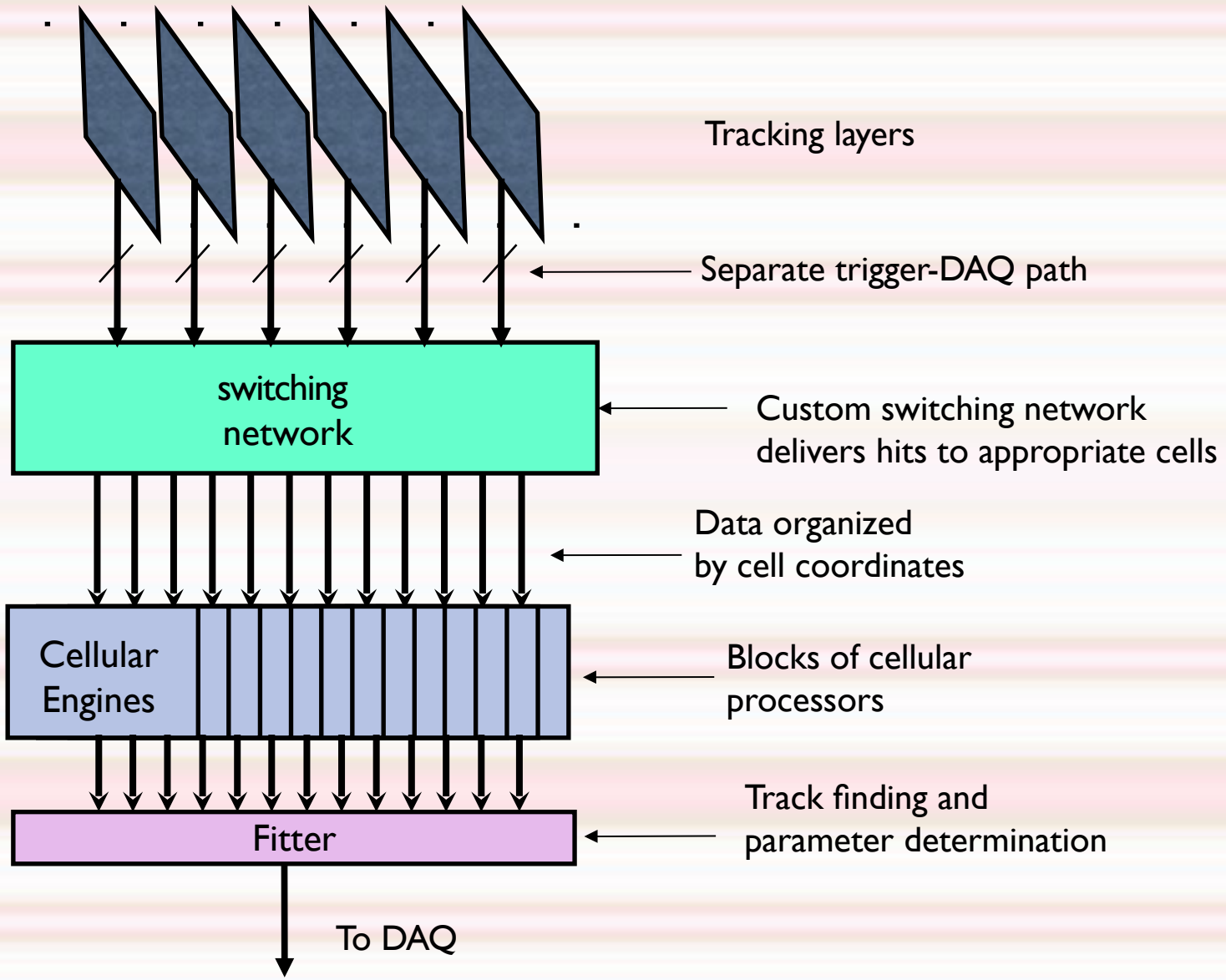
Moving beyond AM's yes/no response allows using fewer cells, and yields immediate parameter estimates

Implementing a "neural-like" tracking algorithm



- A valid track appears as a cluster of cell responses – parameters can then be determined by interpolation of nearby cells.
- First work in this direction in year 2000 [L. Ristori, "An Artificial retina for Fast Track Finding" NIM A453 (2000) 425-429] (historical reason for the name, although today we believe most of this processing actually happens in the primary visual cortex areas)
- Mathematically related to "Hough transform" [P.V.C. Hough, Conf.Proc. C590914 (1959) 554] – but the actual issue is *architectural implementation*

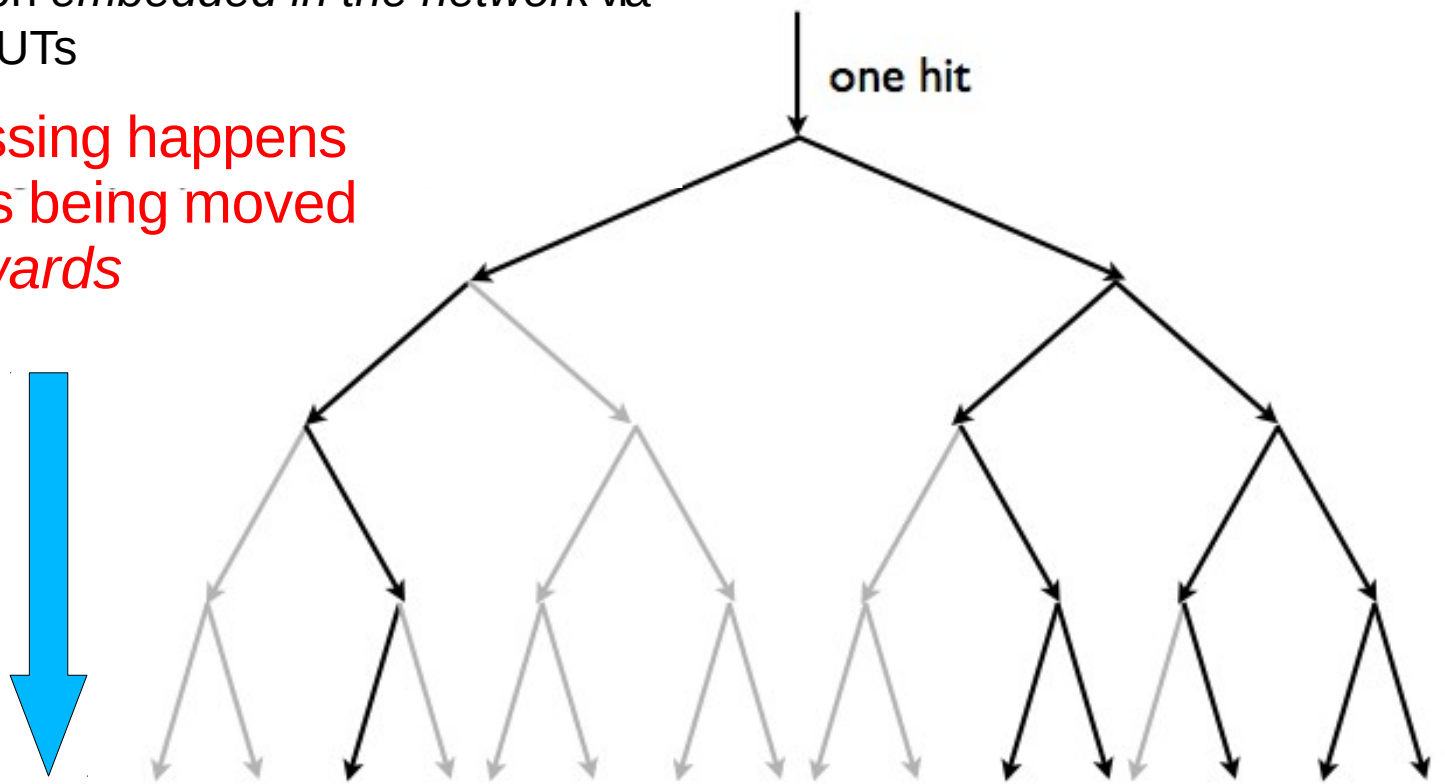
System Architecture is crucial



Hit delivery via programmable switch logic

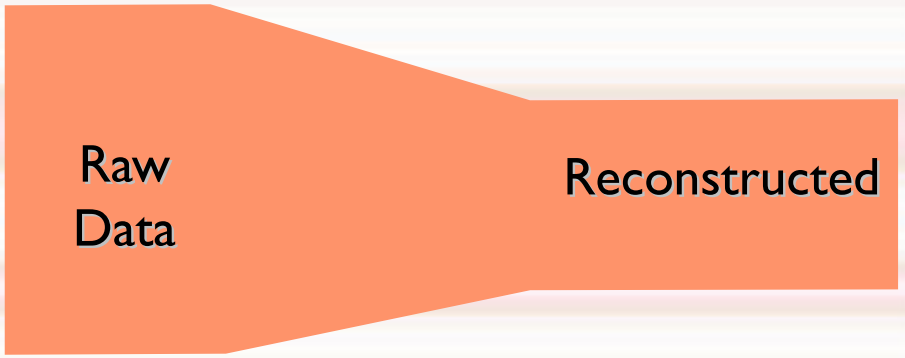
- Hits must be delivered only to the cells that need them (there can be more than one)
- Switch network “knows” where to deliver hits
- All information *embedded in the network* via distributed LUTs

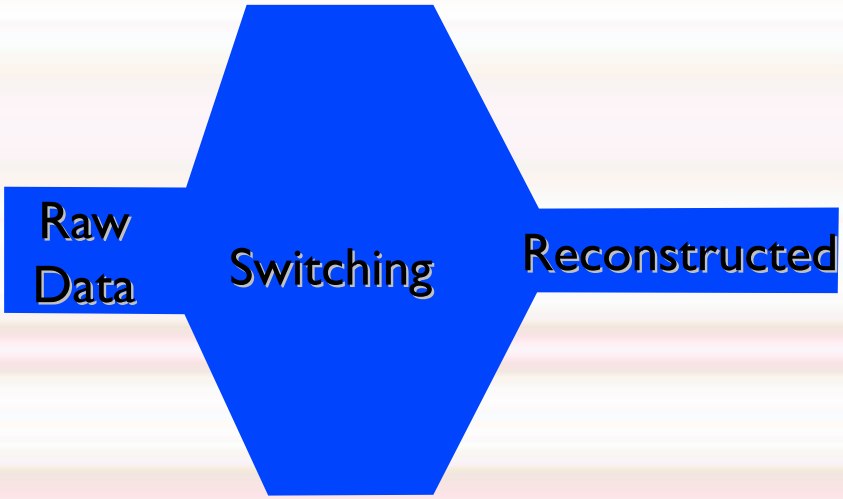
Data processing happens *while* data is being moved - not *afterwards*



The bandwidth profile issue

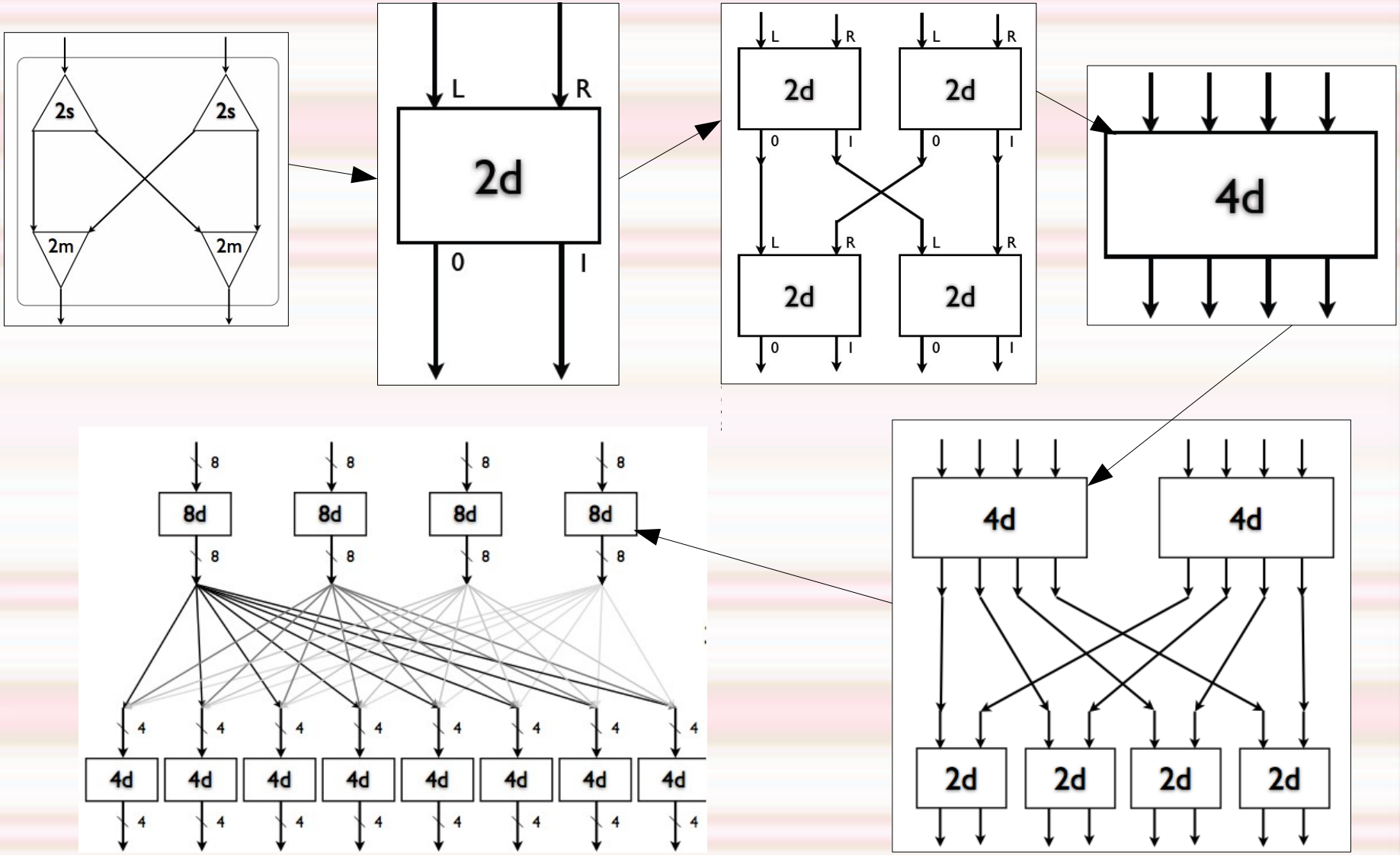
- HEP DAQ typically works by progressively reducing the data bandwidth (funnel-like)
- The RETINA approach needs to *increase* the data flow in the initial stage, by making multiple data copies, and then the bandwidth is shrunk back to lower values when the maxima location is found.
- Curiously enough: evidence of similar process in the brain visual path.
- The process is dependent on the **geometry** of the tracking detector
  - Correlated information between layers helps a lot
  - e.g. CMS's double-layers
  - Possible future time-tagged hits



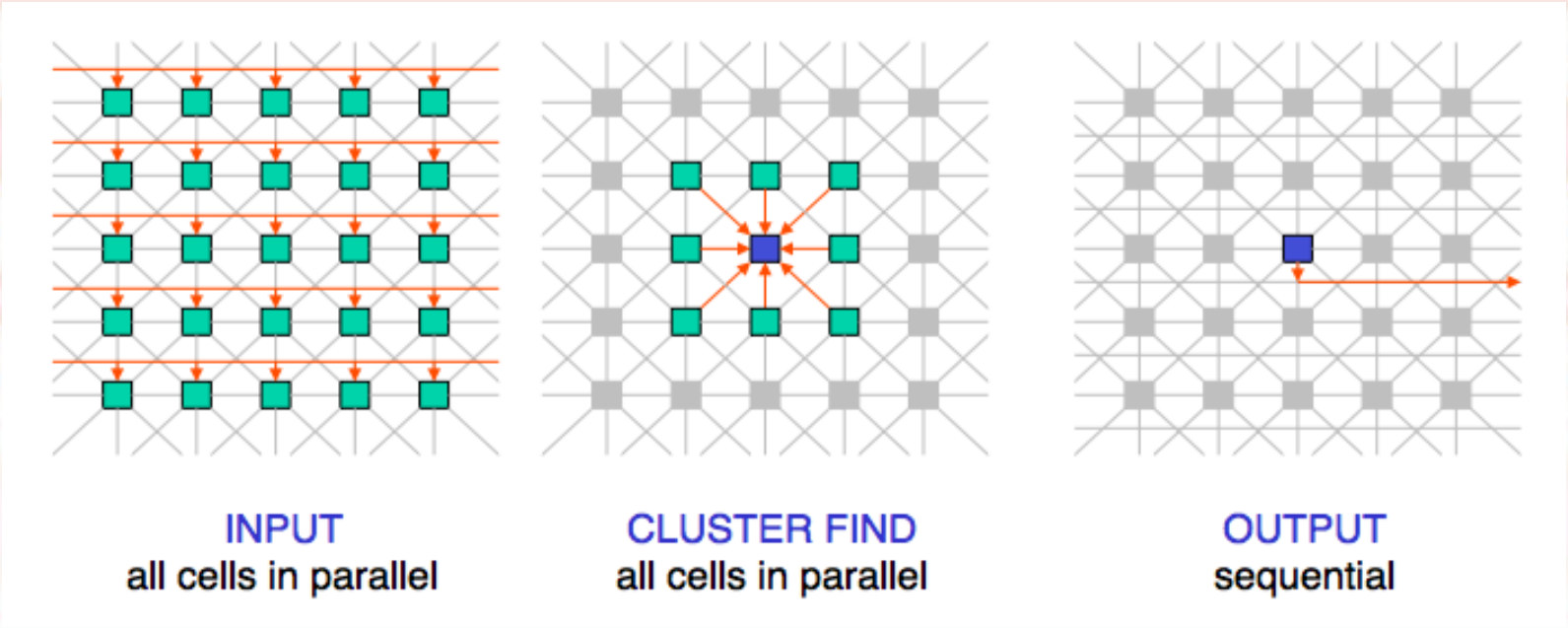


Best to build detector with Data-Processing in mind

Building a large custom switching network from uniform elementary blocks



Cellular computing engine working principle

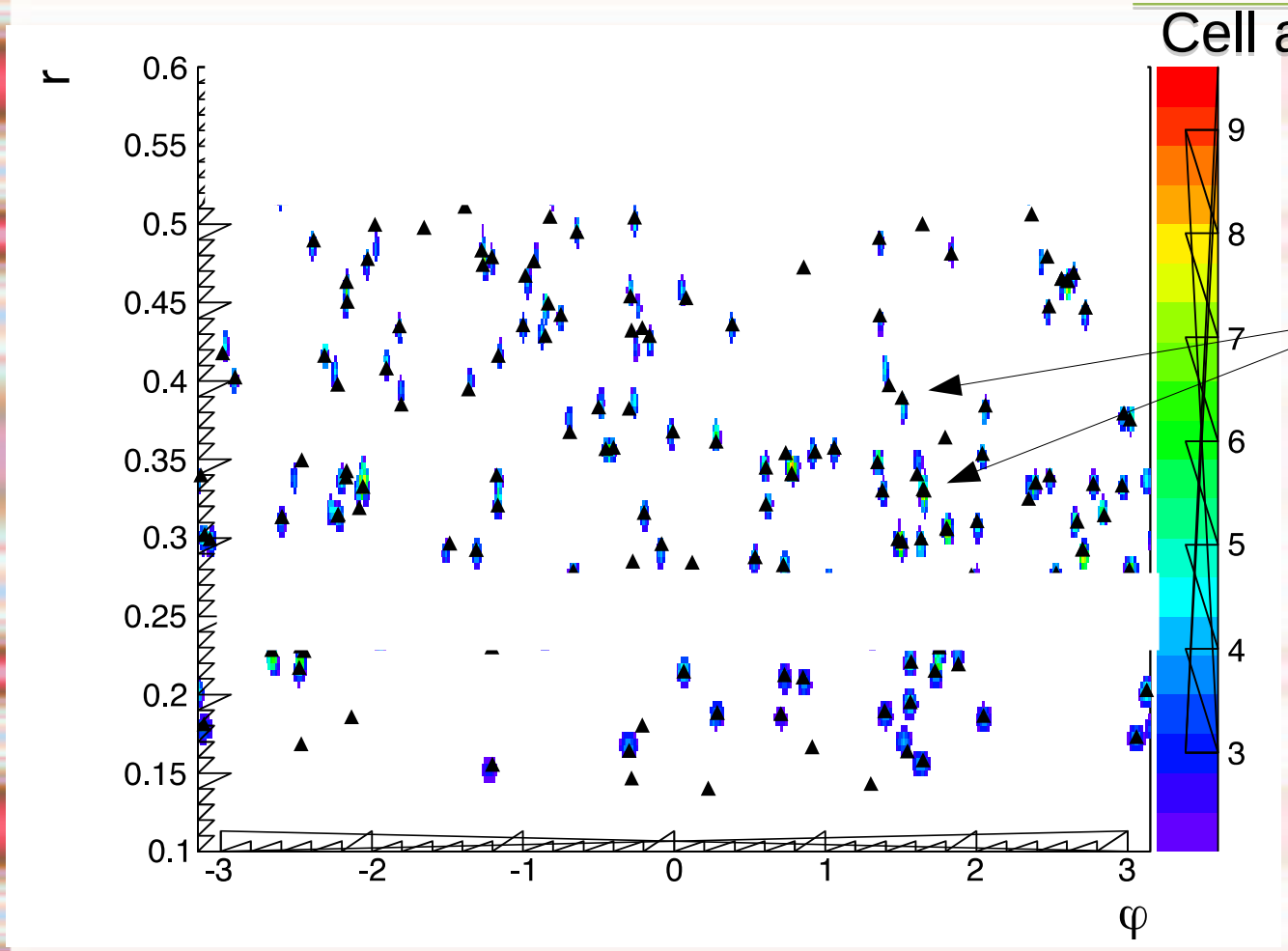


Each node:
- Performs calculation of weights for a hit into a cell
- Handles time-skew between events

In second stage:
- Deals with surrounding cells → local clustering
- Queues results to output

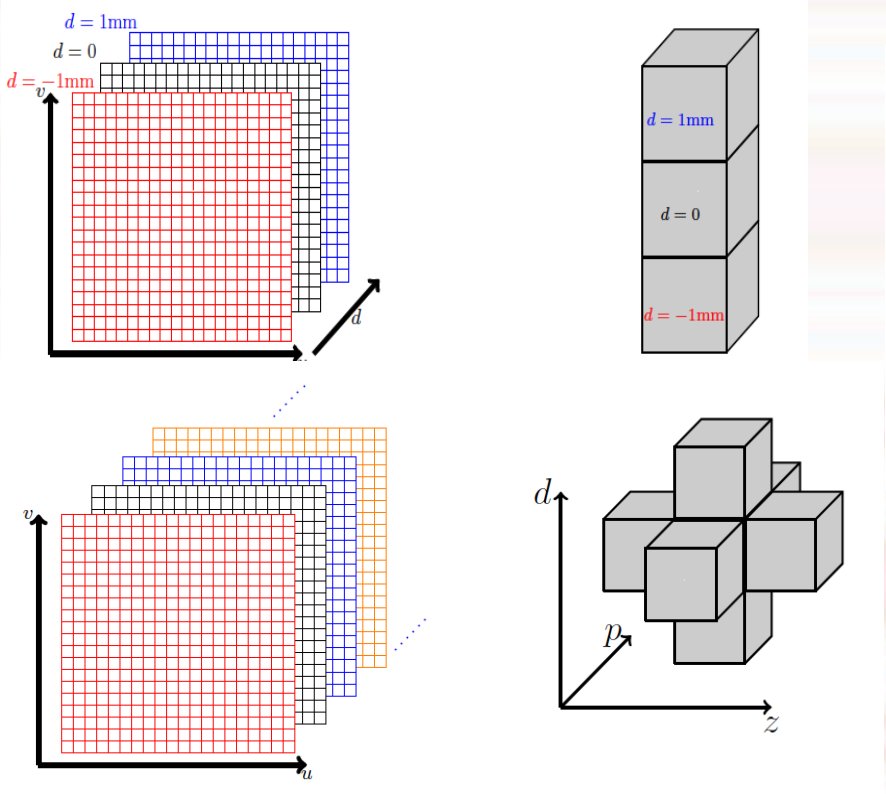
All the above happens in pipeline without stops (data-flow)

High-level simulation in C++: Cell activation map of typical multi-track event

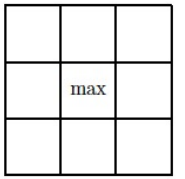


Final stage: Parameter extraction

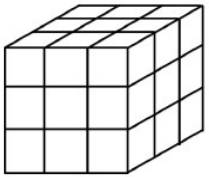
- Two (or 3!) parameters can be extracted directly from cluster centroid in 2D array of cells.
  How about other 2 or 3 parameters ?
  - Add “lateral cells” and interpolate their response
    (Enough when parameter spread is limited)
  - Perform *local* linearized fit (easy with hardware DSPs)



(u, v)
$3 \times 3 = 9$ weights

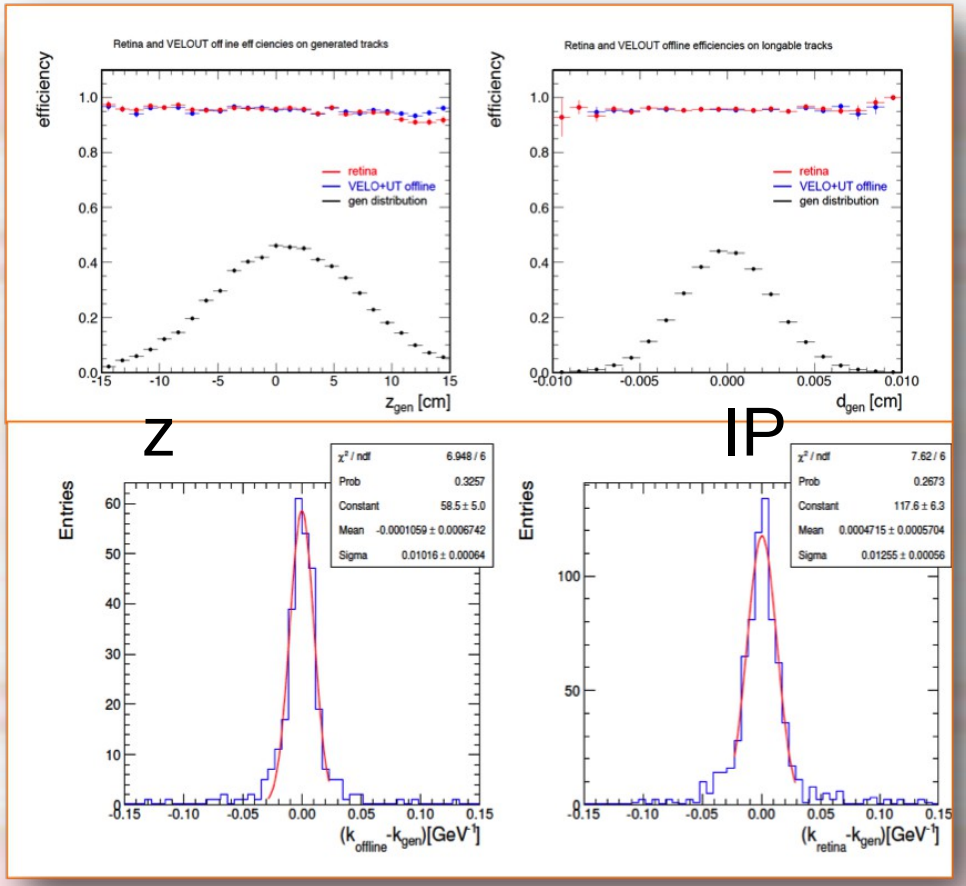


(u, v, d)
$3 \times 3 \times 3 = 27$ weights



Tested with up to 3^5=243 cells
(full 5-parameters tracks)

Tracking performance checks



EFFICIENCY/UNIFORMITY
Equivalent to offline reconstruction
(fake track rates equivalent as well)

MOMENTUM RESOLUTION
Very close to offline.

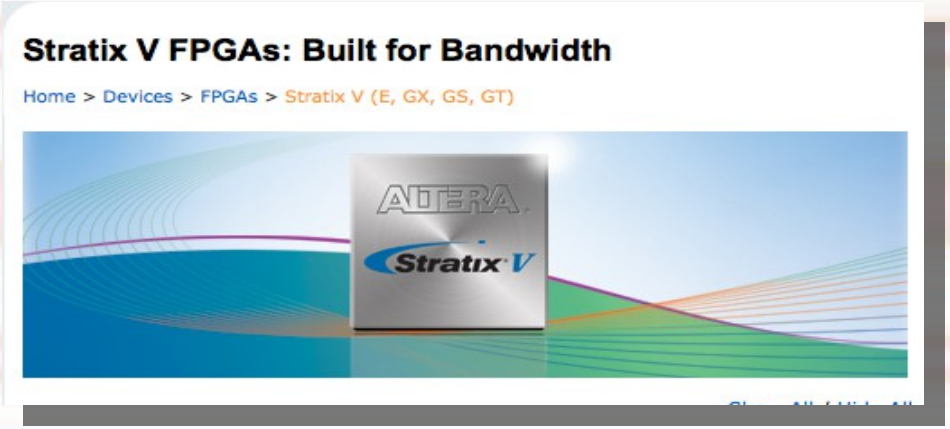
Promise of quality reconstruction at LHC crossing frequency

Implementation Considerations

Most promising and accessible medium: large state-of-the-art FPGA devices.

- Large I/O capabilities: now O(Tb/s) with optical links !
- Large internal bandwidth (a must !)
- Distributed computing resources: DSP slices, SoC...
- *Low power consumption* → critical in the current computing era
- *Fully flexible*, easy (!) to program and simulate in software
- Steep Moore's slope, easily upgradable
- Highly reliable, easy to maintain and update

→ Industry's method of choice for complex projects for small productions (CT scanners, high-end radars...), low-latency (finance, military)



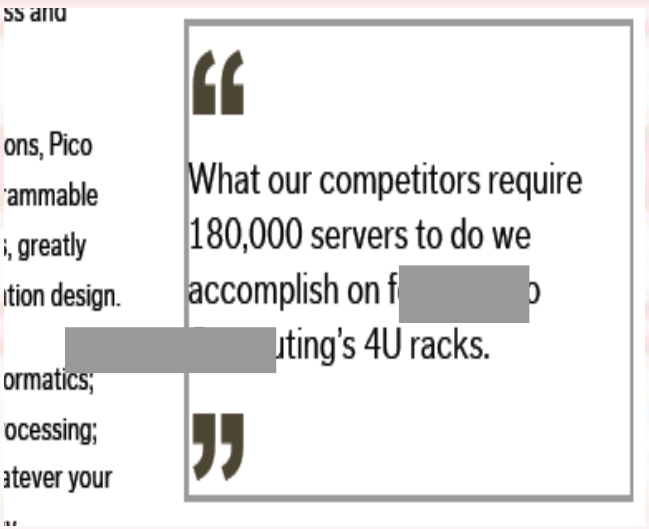
Reality check: other experiences with custom-designed processing in FPGAs

PMC full text: Sensors (Basel). 2013 Jul; 13(7): 9223–9247.
Published online 2013 Jul 17. doi: 10.3390/s130709223
Copyright/License ► Request permission to reuse

Table 3.

Calculation time comparison.

Algorithm and Platform	Execution Time	Processing Image Resolution
LSM of Ji *et al.* [3] on FPGA	15.57 ms	1,024 × 768
Chen *et al.* [40] on FPGA	2.07–3.61ms	512 × 512
Proposed Method on FPGA	15.59 ms	1,024 × 768
Direct HT Computation on PC	(a-1) 0.93 s	1,024 × 768
	(a-2) 1.26 s	1,024 × 768
	(a-3) 1.62 s s	1,024 × 768
	(a-4) 1.45	1,024 × 768



Speedup factors of 70÷500 regularly obtained in vision, military, finance applications

Table II
COMPUTING TIME OF THE HOUGH TRANSFORM

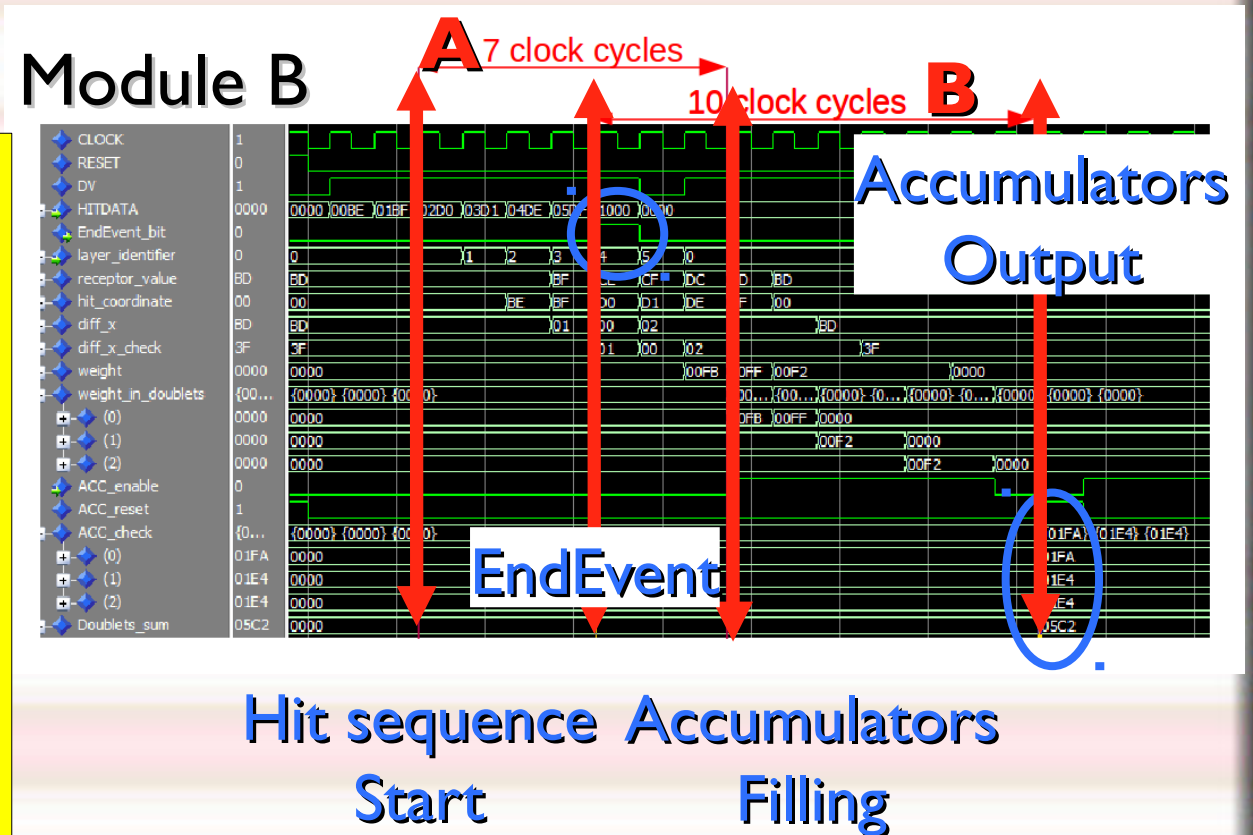
Image	Size	# edge points	Time (FPGA)	Time (CPU)	Speed-up
Figure 1(b)	512×512	33232	135.75μs	37.10ms	273.3
Figure 8(a)	1024×1024	23293	95.27μs	27.47ms	288.3
Figure 9(a)	4096×4096	80092	326.61μs	121.64ms	372.4

FPGA implementation, Timing simulation

- Processing time depends only on # of hits in the event - Results always available after fixed number of cycles

A Time between hit delivery and accumulator update

B Time between end sequence and accumulator output

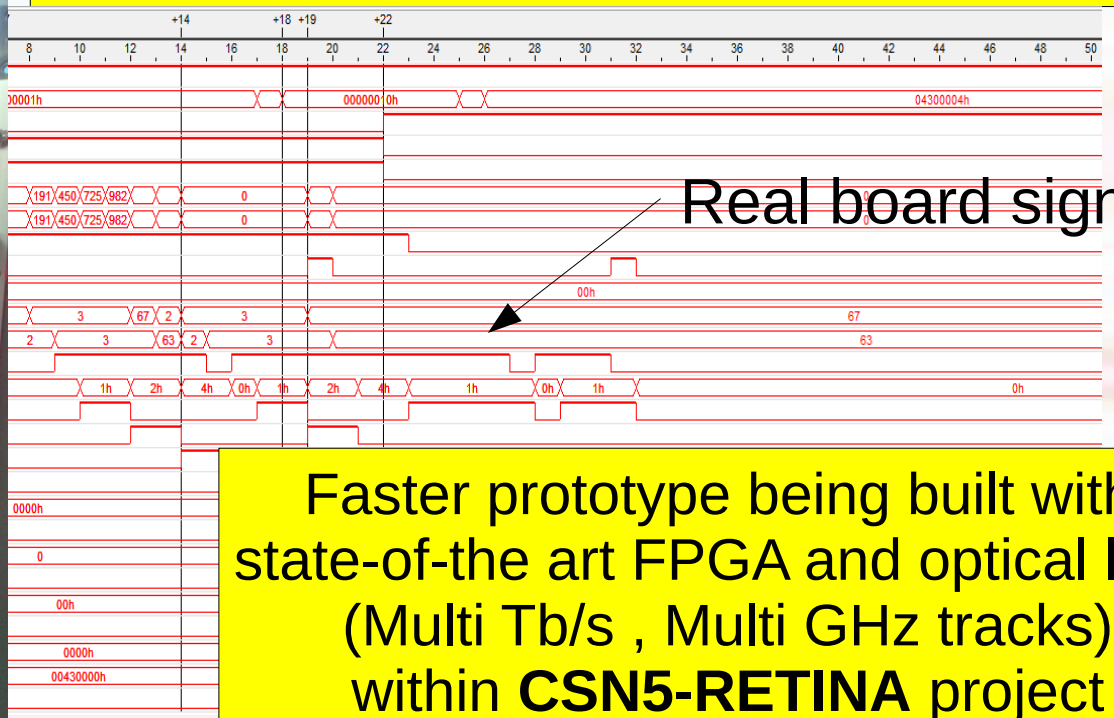


- Latency **~20 cycles. Shortest ever achieved**
- Require **1 – 5 kLE** of logic → O(10^3) cells/average FPGA
- *Can build tracker with **O(100)** medium-size FPGAs*

Reality Check: Lab Test with NA62 DAQ boards (TEL62)

- Boards based on 4 Stratix-IV FPGAs @160MHz (not optimized for this job !)
- Events processed in boards and bit-level checked with C++ simulation.
- Reconstruction rate: 6 MHz/(board pair) (latency **few µs** → will improve)
  Compare normal readout-only operation @1MHz →
  Track reconstruction doable "on the fly" while reading detector

→ Step towards "embedded tracking"



Faster prototype being built with state-of-the art FPGA and optical links (Multi Tb/s , Multi GHz tracks) within **CSN5-RETINA** project

Summary

- Future HEP experiments will increasingly depend on large computing power
- A key to progress will be the capability of real-time reconstruction by special-purpose processors.
- RETINA project aimed at designing better real-time tracking processors using architectures inspired by natural vision
- Encouraging preliminary results may lead to a HEP future with *detector-embedded data reconstruction*

Bibliography

G.Punzi and L.Ristori, Triggering on heavy flavors at hadron colliders, Ann.Rev.Nucl.Part.Sci. 60 (2010) 595-614

L. Ristori, An artificial retina for fast track finding, NIM A 453 (425-429), http:// inspirehep.net/record/539203

M. M. Del Viva, G. Punzi, The brain as a trigger system, http://arxiv.org/abs/ 1410.5123

M. M. Del Viva, G. Punzi, D. Benedetti, Information and Perception of Meaningful Patterns [PDF]

N. Neri et al., First results of the silicon telescope using an 'artificial retina' for fast track finding, Talk at ANIMMA15 conference, http://www.ipfn.ist.utl.pt/ANIMMA2015/

S. Stracka, A specialized processor for track reconstruction at the LHC crossing rate, Talk at Connecting The Dots Workshop 2015, https://indico.physics.lbl.gov/ indico/getFile.py/access? contribId=14&sessionId=2&resId=0&materialId=slides&confId=149

R. Cenci, Artificial retina processor for track reconstruction, Talk at Connecting The Dots Workshop 2015, https://indico.physics.lbl.gov/indico/getFile.py/access? contribId=2&sessionId=9&resId=0&materialId=slides&confId=149

A. Abba et al., Progress Towards the First Prototype of a Silicon Tracker Using an 'Artificial Retina' for Fast Track Finding, Poster at TWEPP14, https://indico.cern.ch/ event/299180/session/7/contribution/64

A. Piucci, Reconstruction of tracks in real time at high luminosity environment at LHC, Master thesis, https://etd.adm.unipi.it/theses/available/etd-06242014-055001/.

D. Ninci, Real-time track reconstruction with FPGA at LHC, https://etd.adm.unipi.it/ theses/available/etd-11302014-212637/.

F. Spinella et al., The TEL62: A real-time board for the NA62 Trigger and Data Acquisition. Data flow and firmware design, IEEE Nucl. Sci. Symp. Conf. Rec., 1 (2014).

A. Abba et al., The artificial retina for track reconstruction at the LHC crossing rate, arXiv:1411.1281 [ICHEP 2014], https://inspirehep.net/record/1326137.

N. Neri, First prototype of a silicon tracker using an 'artificial retina' for fast track finding , PoS TIPP2014 (2014) 199 [TIPP2014], https://inspirehep.net/record/ 1315951.

A. Abba, The artificial retina processor for track reconstruction at the LHC crossing rate, JINST 10 (2015) 03, C03018 [WIT2014], https://inspirehep.net/record/1315154.

A. Abba et al, Simulation and performance of an artificial retina for 40 MHz track reconstruction, JINST 03-10 (C03008) [WIT2014], https://inspirehep.net/record/ 1314984.

A. Abba et al., A Specialized Processor for Track Reconstruction at the LHC Crossing Rate, JINST 9 (C09001) 2014 [INSTR14], https://inspirehep.net/record/ 1303542.

A. Abba et al., *The Readout Architecture for the Retina-Based Cosmic Ray Telescope*, Real Time Conference (RT), 2014 19th IEEE-NPSS, [IEEE-RT 2014] http://dx.doi.org/10.1109/RTC.2014.7097516.

A. Abba, et al., A retina-based cosmic rays telescope, Real Time Conference (RT), 2014 19th IEEE-NPSS, [IEEE-RT2014], http://dx.doi.org/10.1109/RTC. 2014.7097515, https://inspirehep.net/record/1367442.

A. Abba et al., A specialized track processor for the LHCb upgrade, CERNa-LHCb- PUB-2014-026 https://cds.cern.ch/record/1667587.

BACKUP