

High-speed data processing in particle physics experiments

Alex Tapper

Slides: <https://www.hep.ph.ic.ac.uk/~tapper/talks/Altera.pdf>

Altera, 15th December 2025

About me ..

- Particle physicist
- Oxford → Imperial → Hamburg → Geneva ↔ Chicago
- Build, operate and analyse data from large experiments in huge collaborations
- Data acquisition and realtime data filtering/analysis

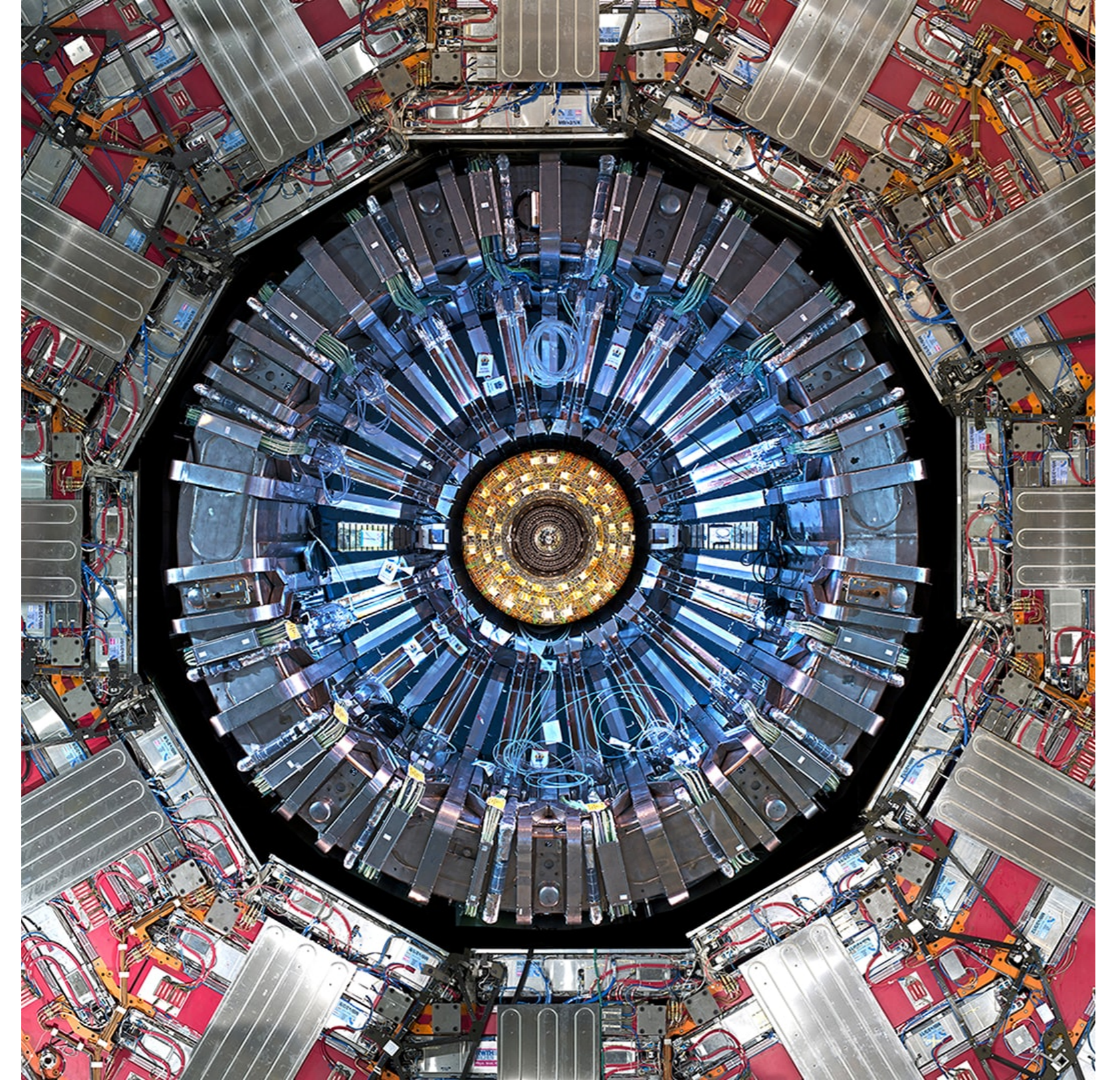
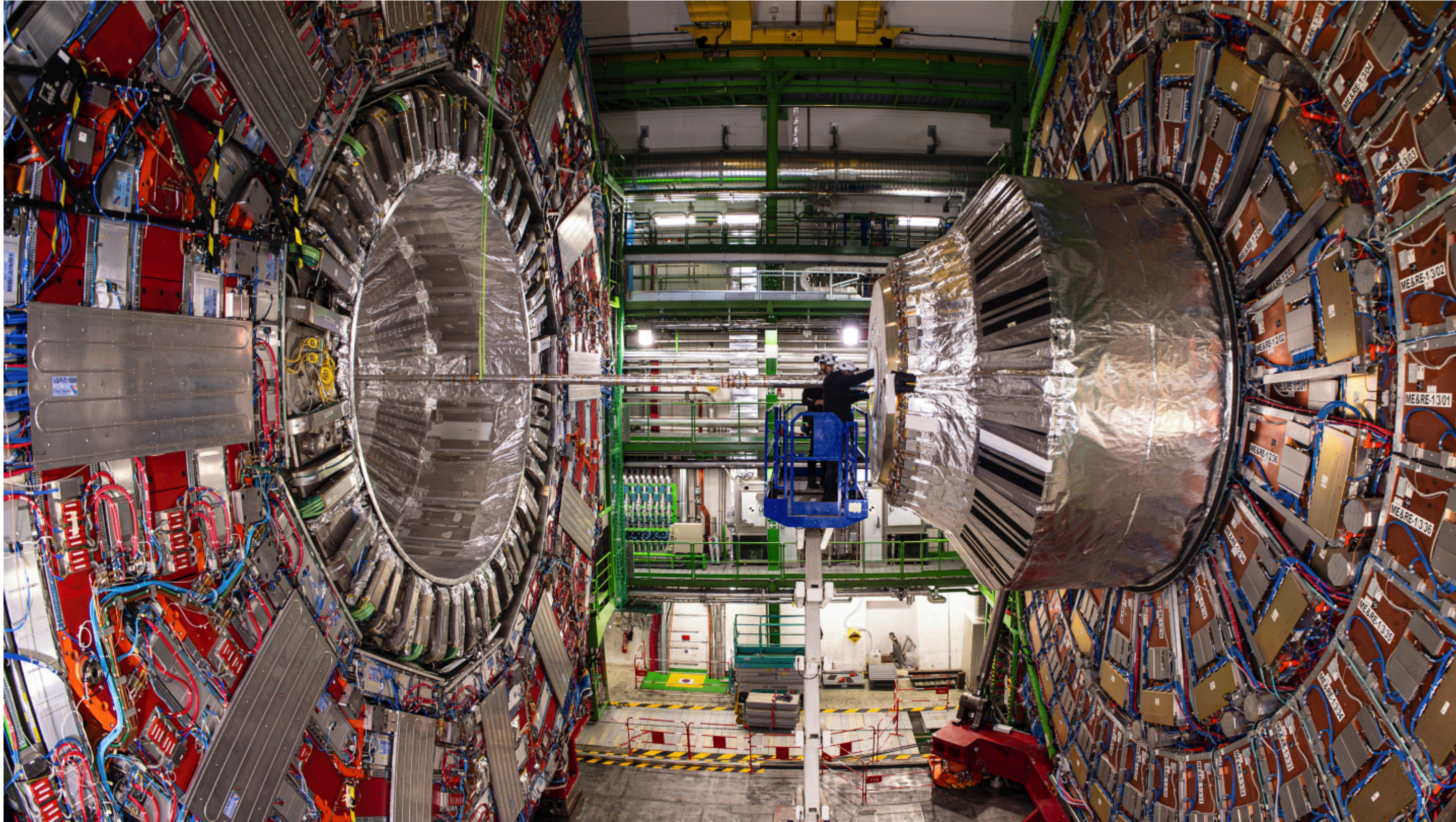


<https://www.hep.ph.ic.ac.uk/~tapper/>

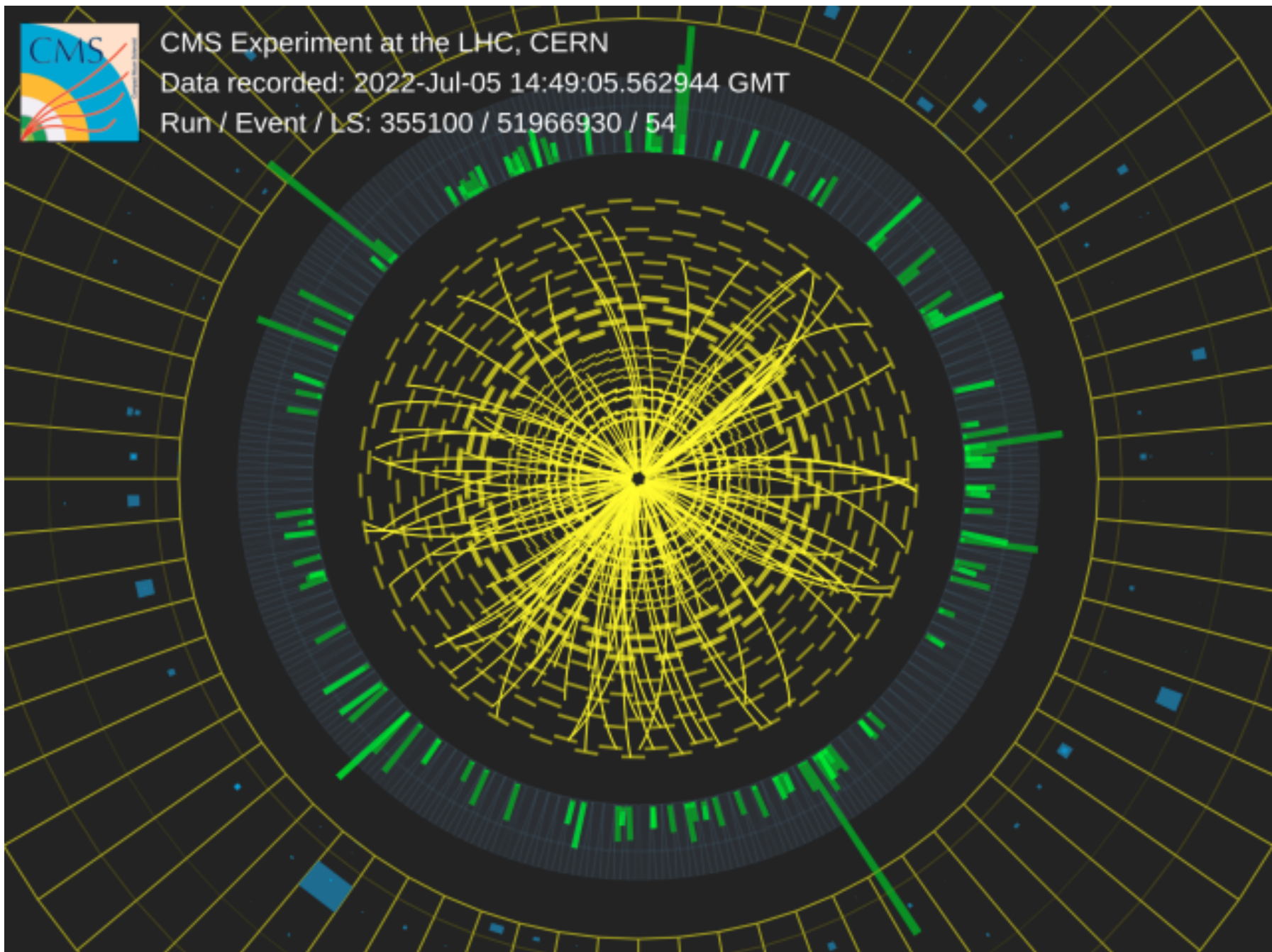
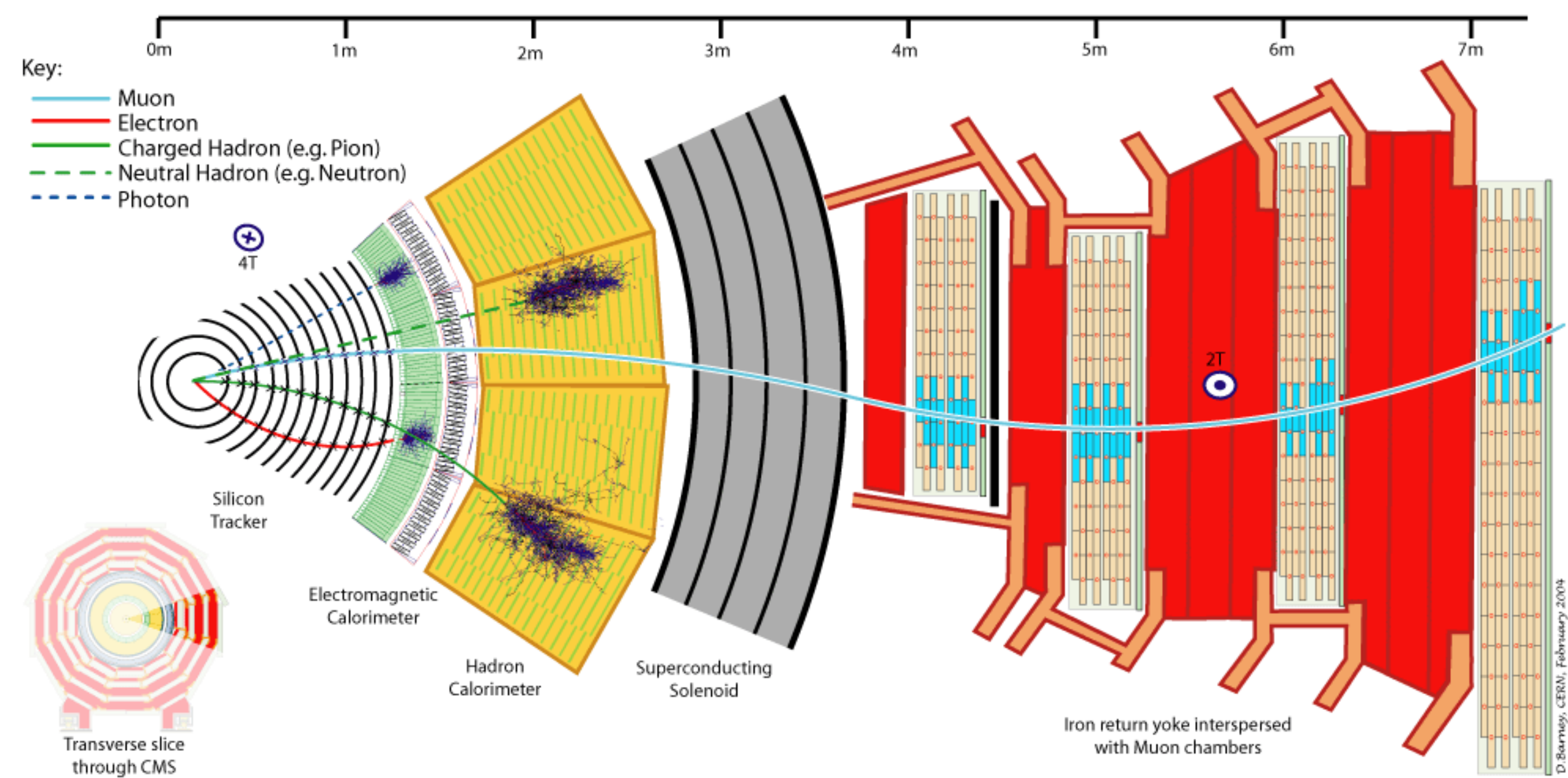
CERN & Large Hadron Collider



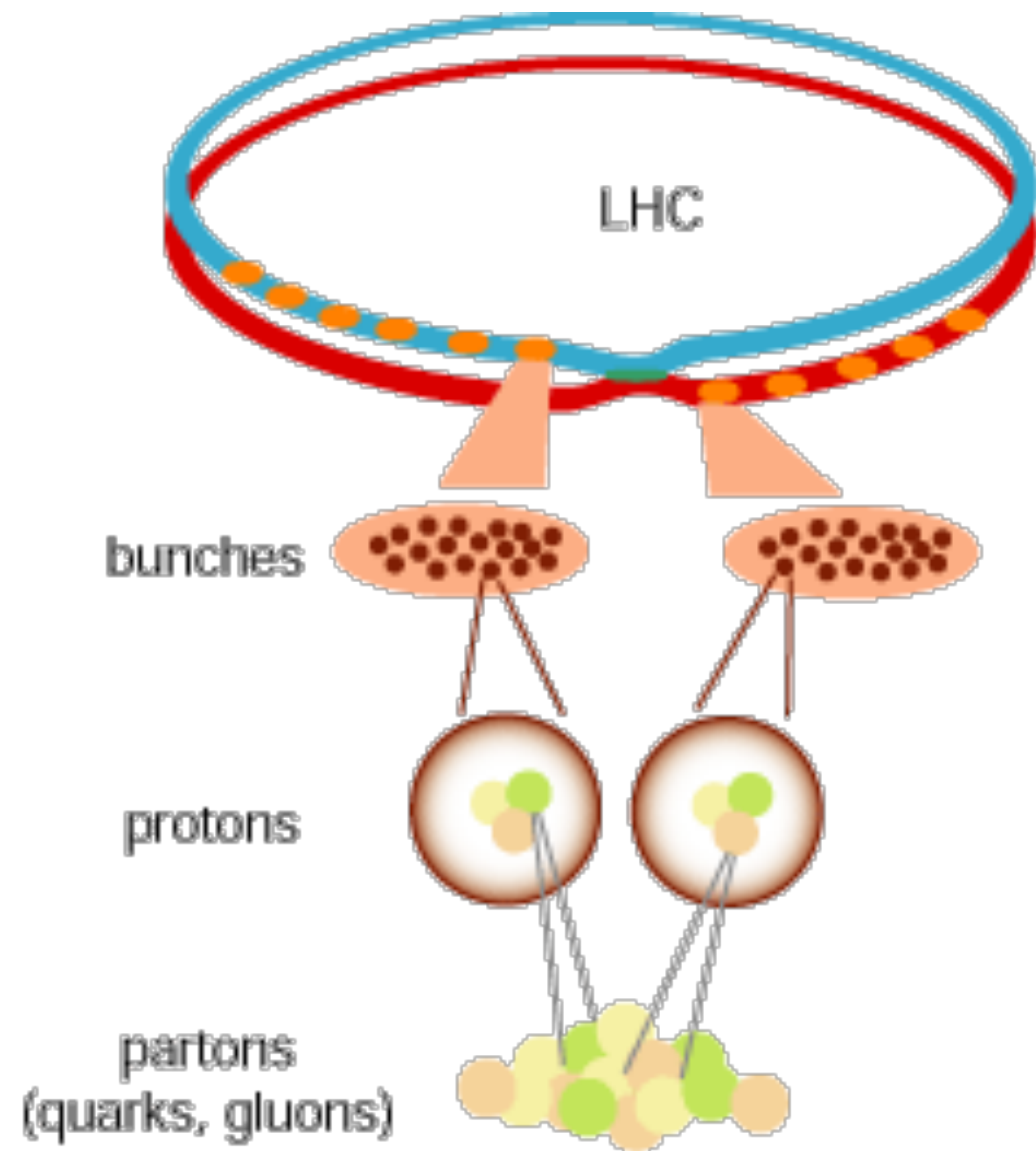
Detectors



Detectors

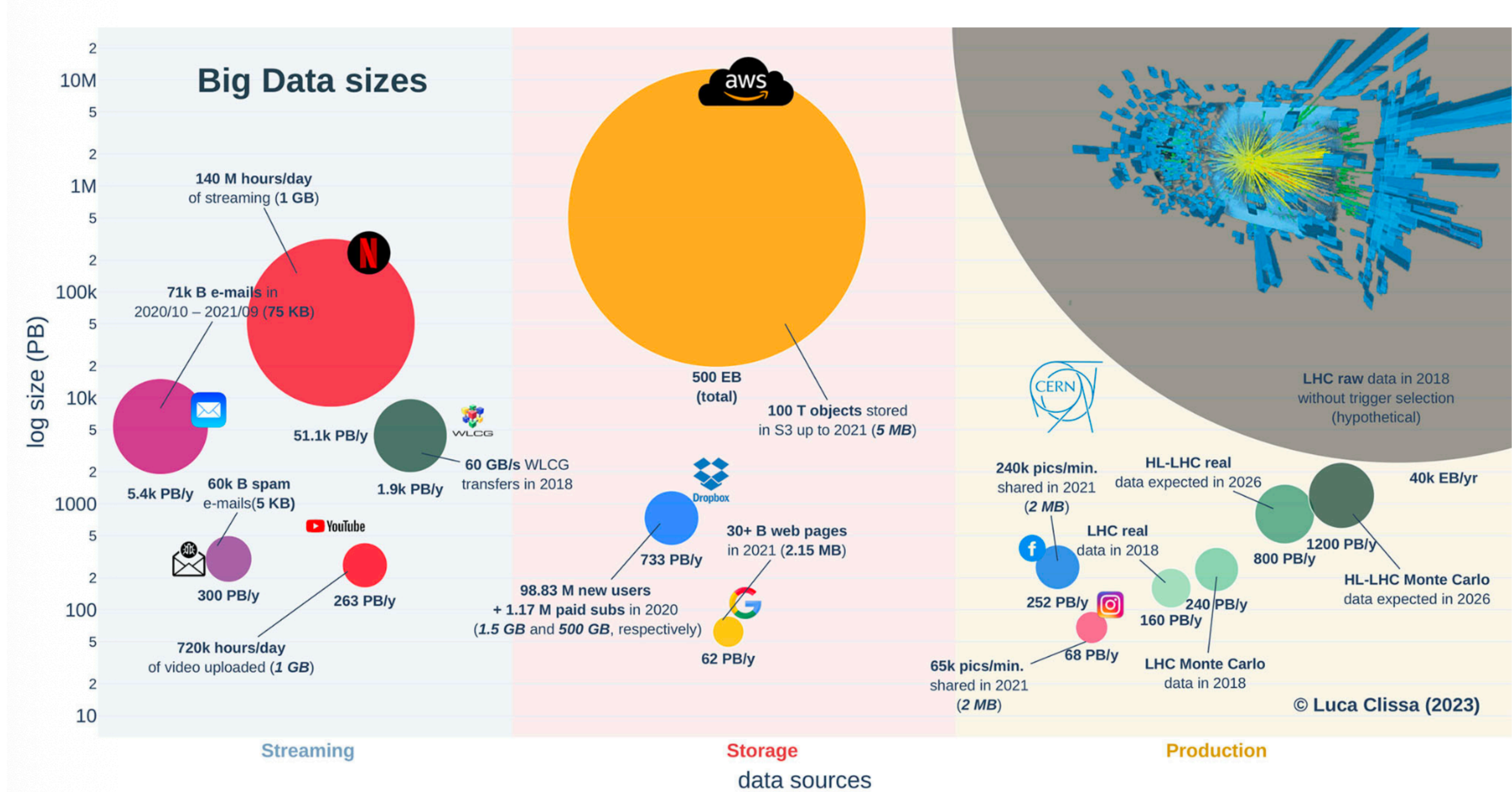


LHC in numbers

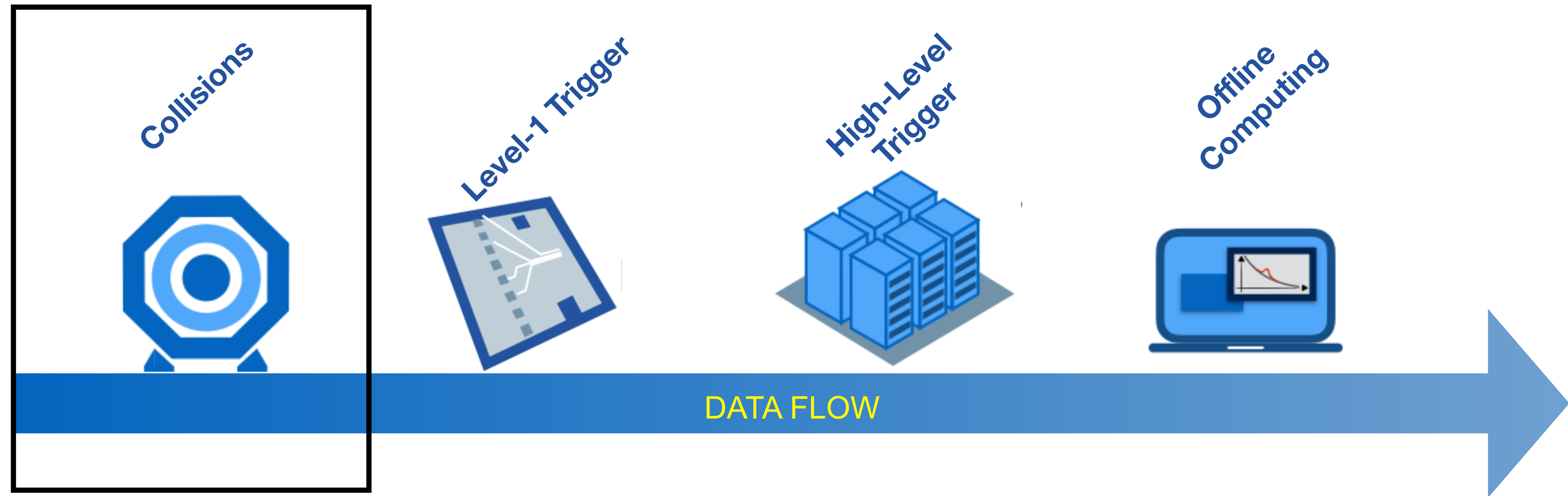


- 10 000 magnets to control beams
- Temperature -271.3 C
- Vacuum 10^{-13} atmospheres
- Beams made up of *bunches* of particles
- Around 3000 bunches, each 10^{11} particles
- Complete the 27km orbit 11000 times a second
- Collisions at 40 MHz

LHC in numbers

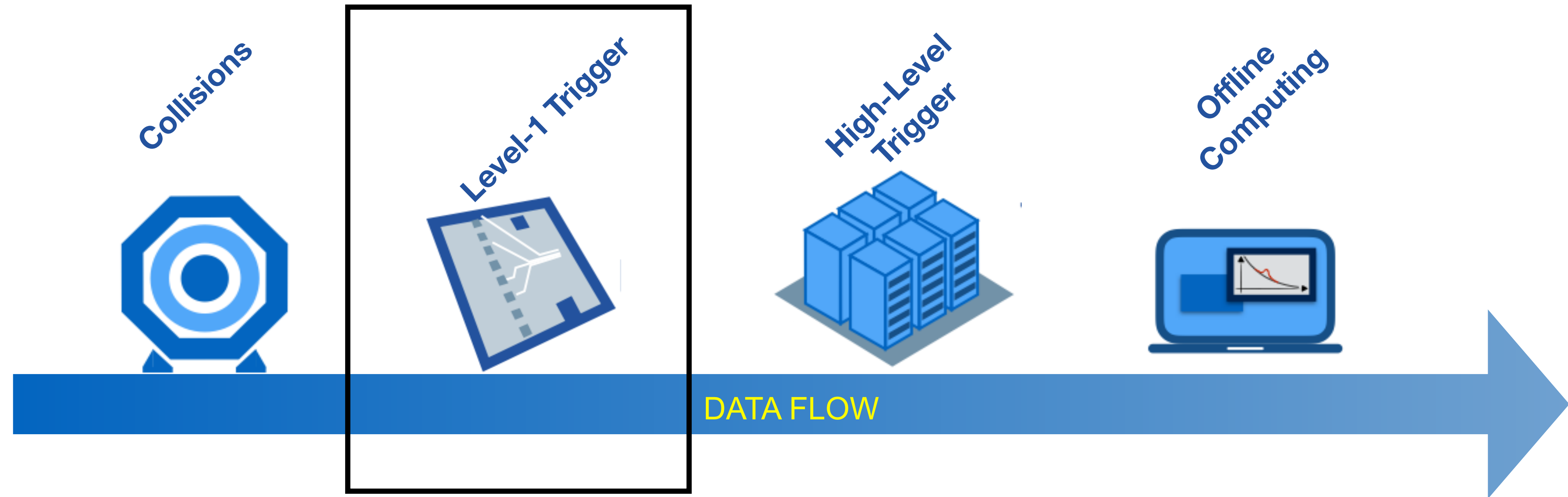


Detector data flow & processing



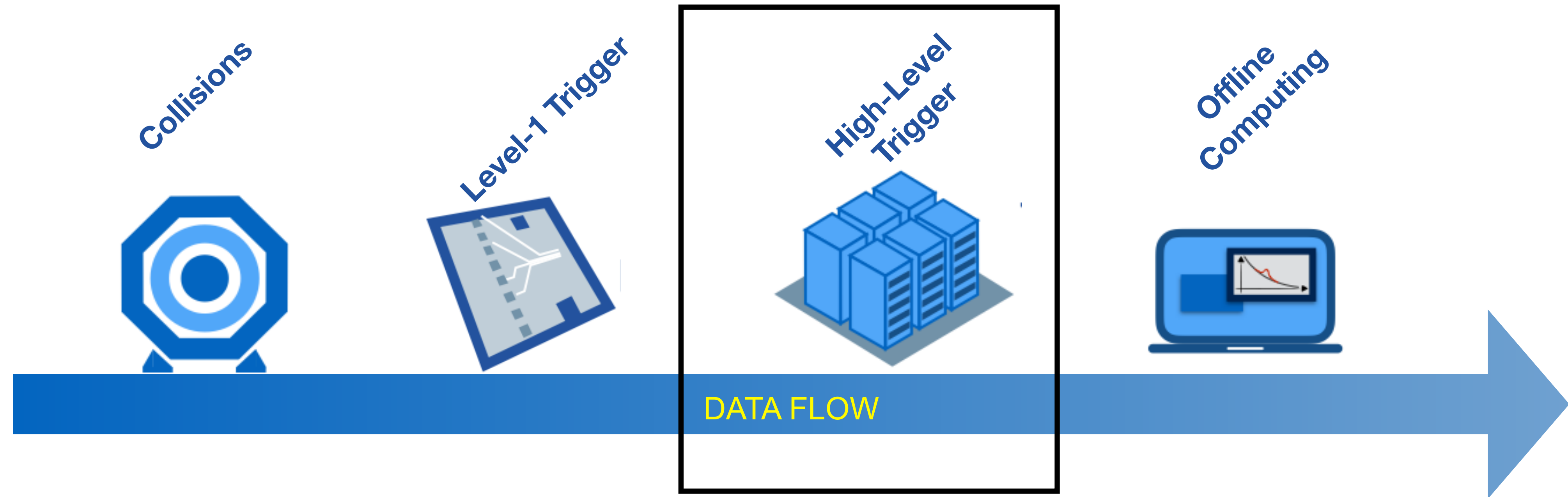
- 40 MHz collision rate x 1-8 MB per event = O (100 TB/s)
- On detector ASICs — noise reduction, pedestal subtraction, store in FIFO
- Impossible to store all data → required to filter data → buffer on detector O(10μs)

Detector data flow & processing



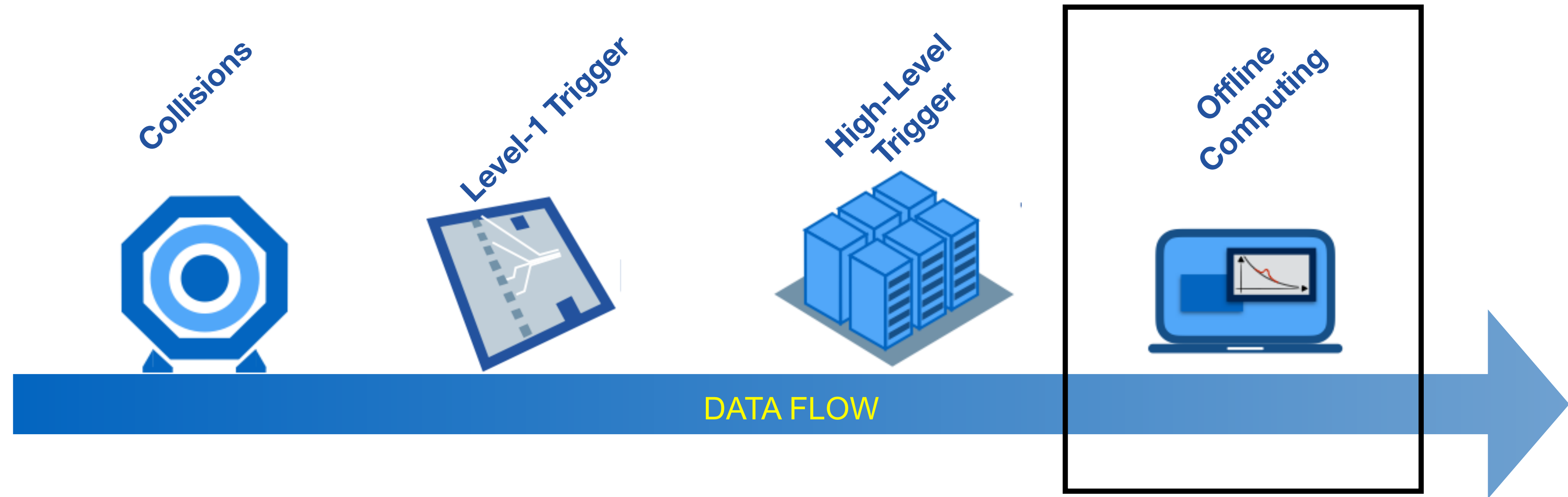
- 40 MHz input rate — 100 kHz output rate
- Process in $O(10\mu\text{s})$ **fixed** latency $\rightarrow$ algorithms $O(1\mu\text{s})$
- Coarse/approximate detector information — currently classical algorithms

Detector data flow & processing



- 100 kHz input rate — O(10 KHz) output rate — 40 GB/s
- Software system — 30 000 x86 CPU cores + GPU accelerators (Alpaka lib)
- GPU offload 40% of processing — leads to 70% improvement in throughput

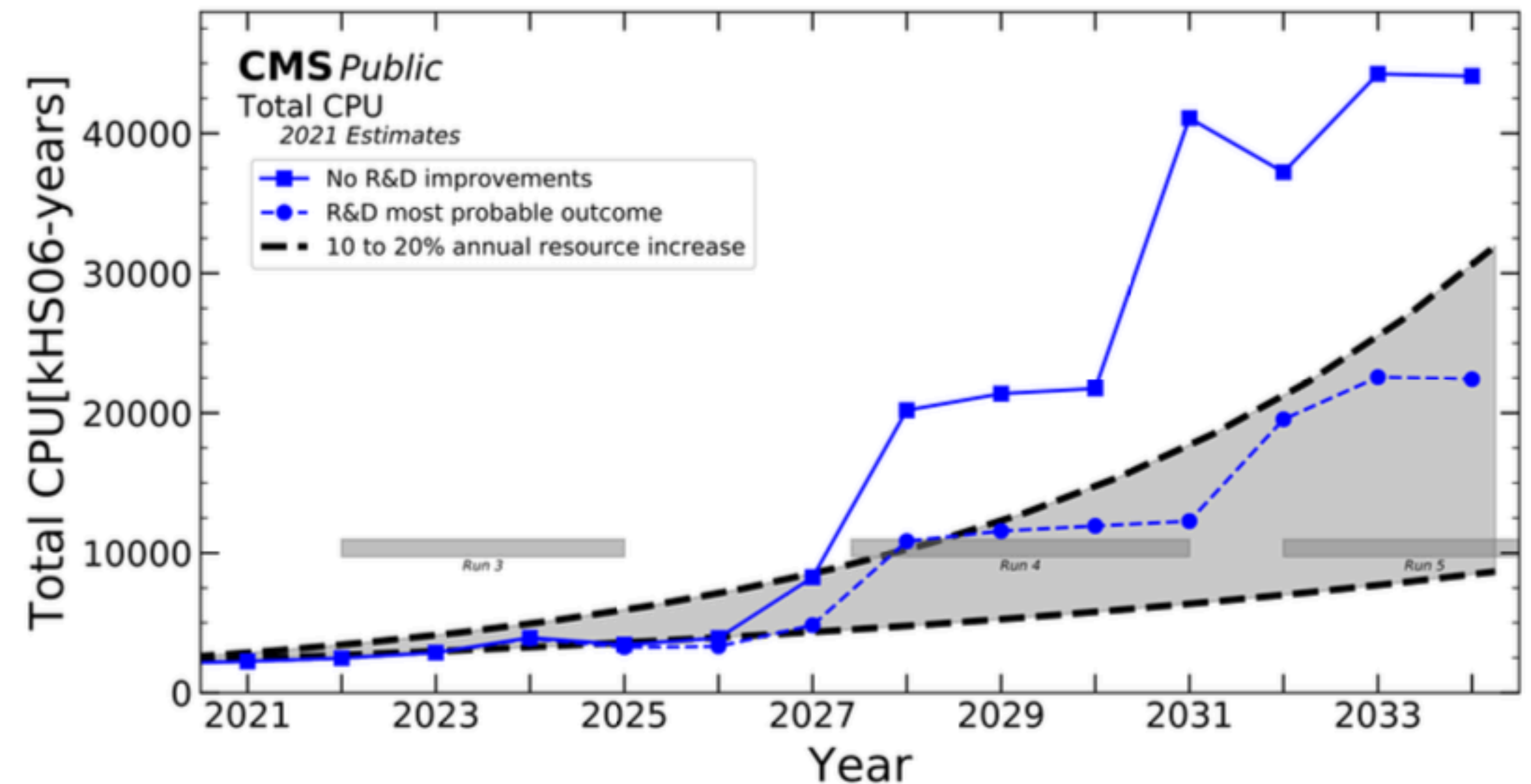
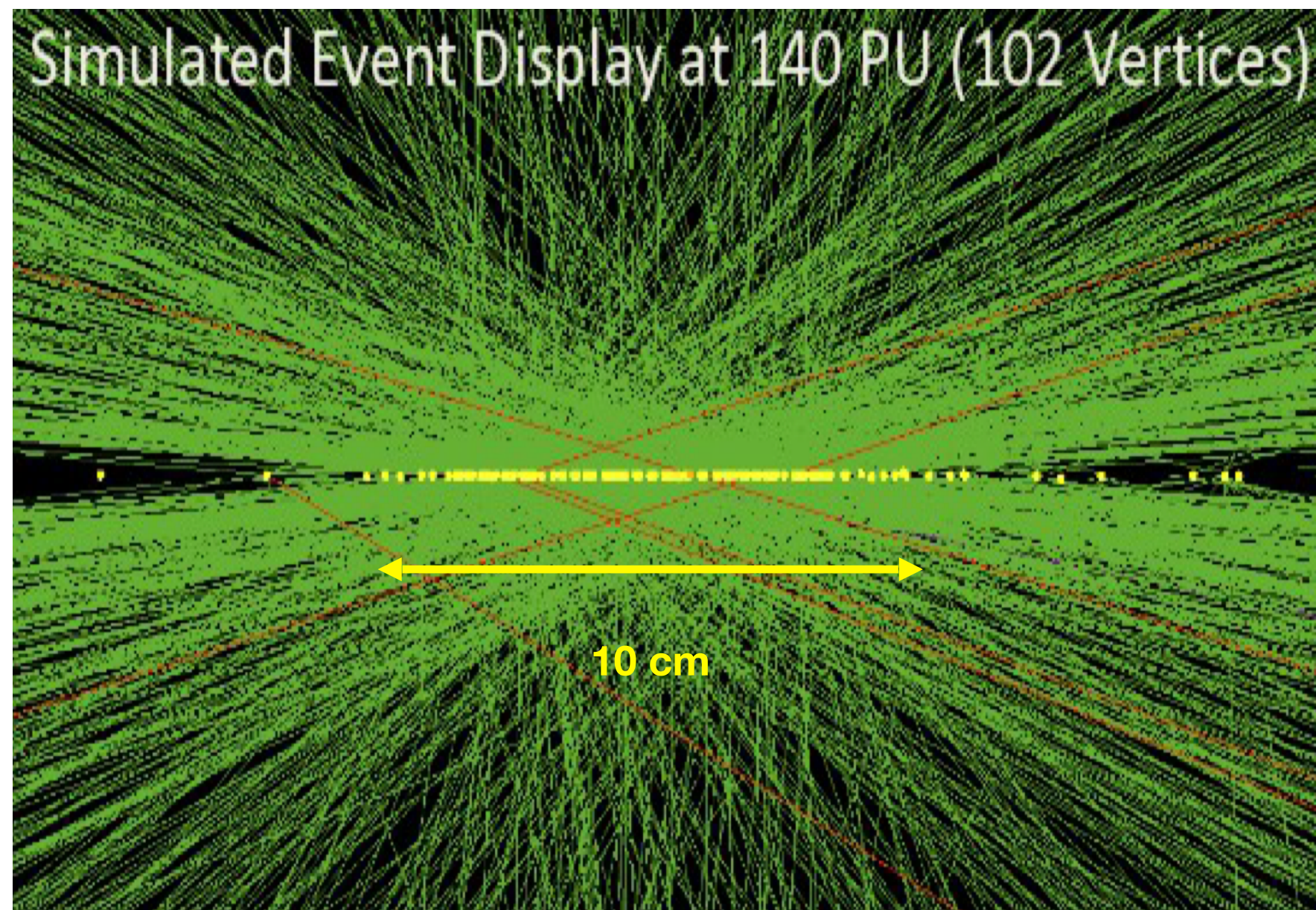
Detector data flow & processing



- Software system — ~500 000 x86 CPU cores distributed throughout world
- Also able to use ARM based resources, GPU and HPC resources where available
- Custom data and job management systems — Exa Byte dataset

Future challenges

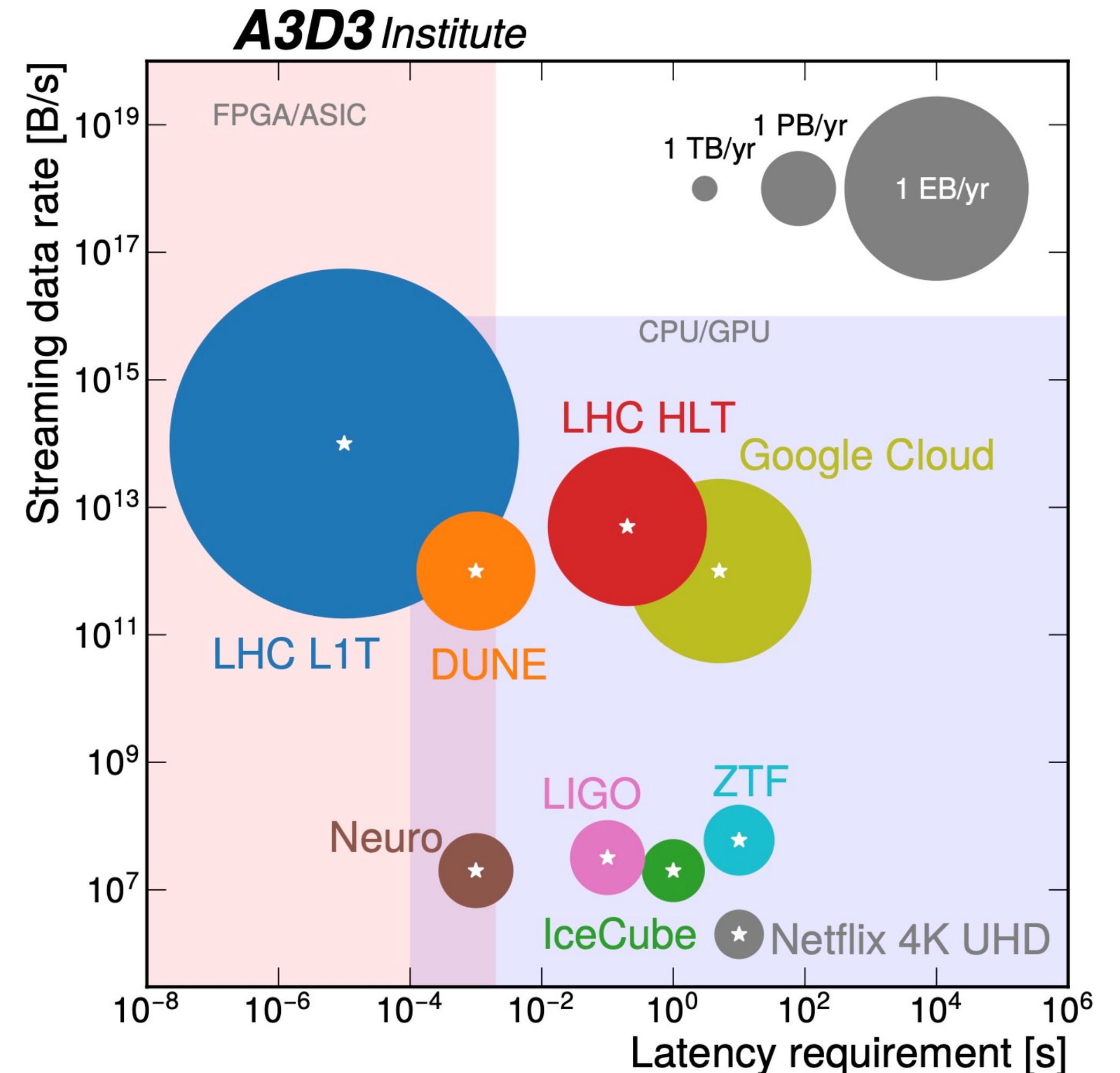
<https://cds.cern.ch/record/2815292>



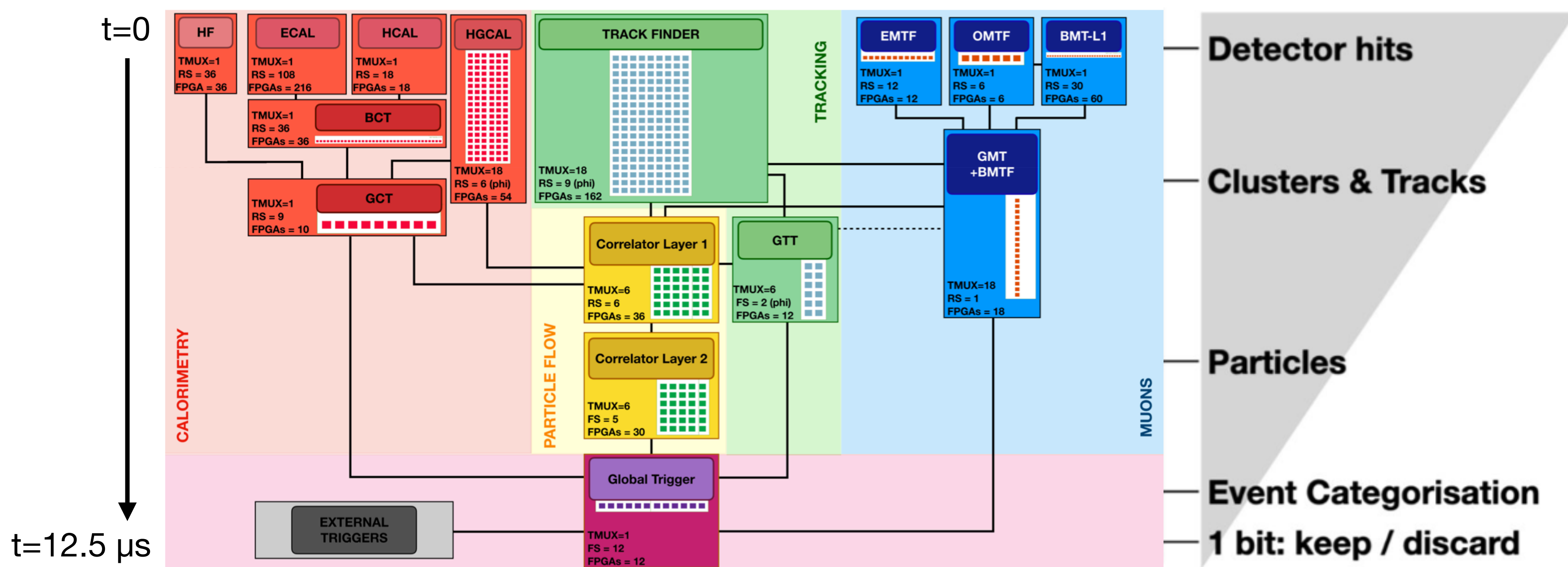
- Upgrade to be completed by 2030 — continue running until ~2040
- Higher intensity collisions → new detectors → larger data sizes

Solutions

- Hardware filtering — upgrade **FPGA-based** processors — ML algorithms
- Stream processing — **hybrid hardware/software** — new ideas
- Software filtering — **heterogeneous computing** — co-processor — IaaS
- Offline processing — **cloud, HPC, heterogeneous computing** ...



Detector data flow & processing



- Small boxes represent individual FPGAs
- Data processing partitioned in space and time
- Optical links with custom protocol (no forward error correction)
- Overall **~700** FPGAs in backend/filtering system

Detector data flow & processing

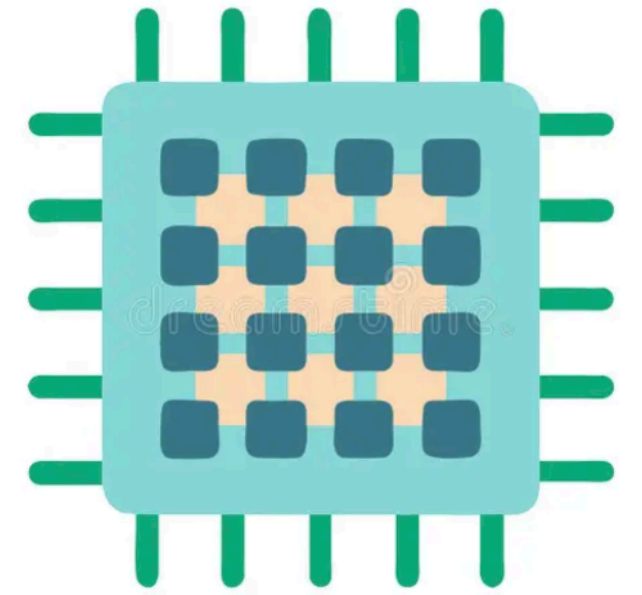


<https://serenity.web.cern.ch/serenity/>

- Generic data processing engine
- Advanced Telecommunications standard (ATCA)
- Samtec Firefly optical Rx/Tx — 124 optical links @ 28 Gb/s
- Single large FPGA — typically AMD Virtex UltraScale+ VU13P (4 SLRs) — also other variants
- Kria onboard control: Zync FPGA+ARM SoC
- Up to 7 Tb/s bandwidth



AI on FPGA: challenges & advantages



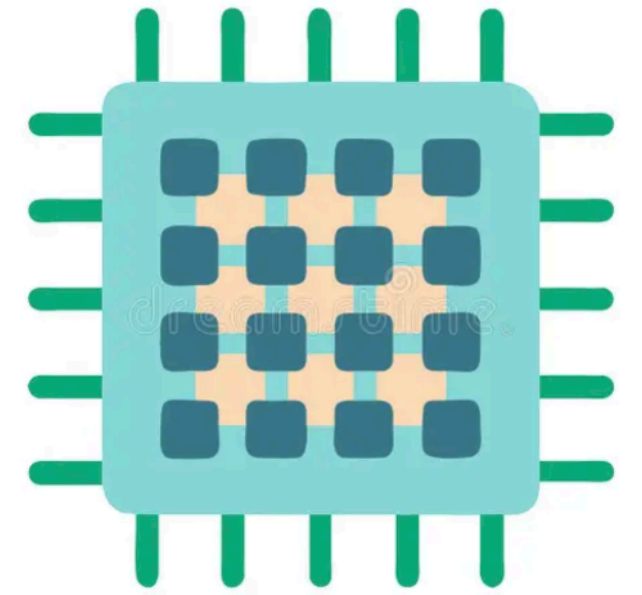
- **Challenges**

- Need to operate with strict, fixed, latency constraints
- Limited FPGA resources — model size critical

- **Advantages**

- Fine-grained/resource parallelism — allows for low latency
- Pipelined — allows for high throughput
- Low power — compared to CPU/GPU
- Custom data types — tailor at low-level to model

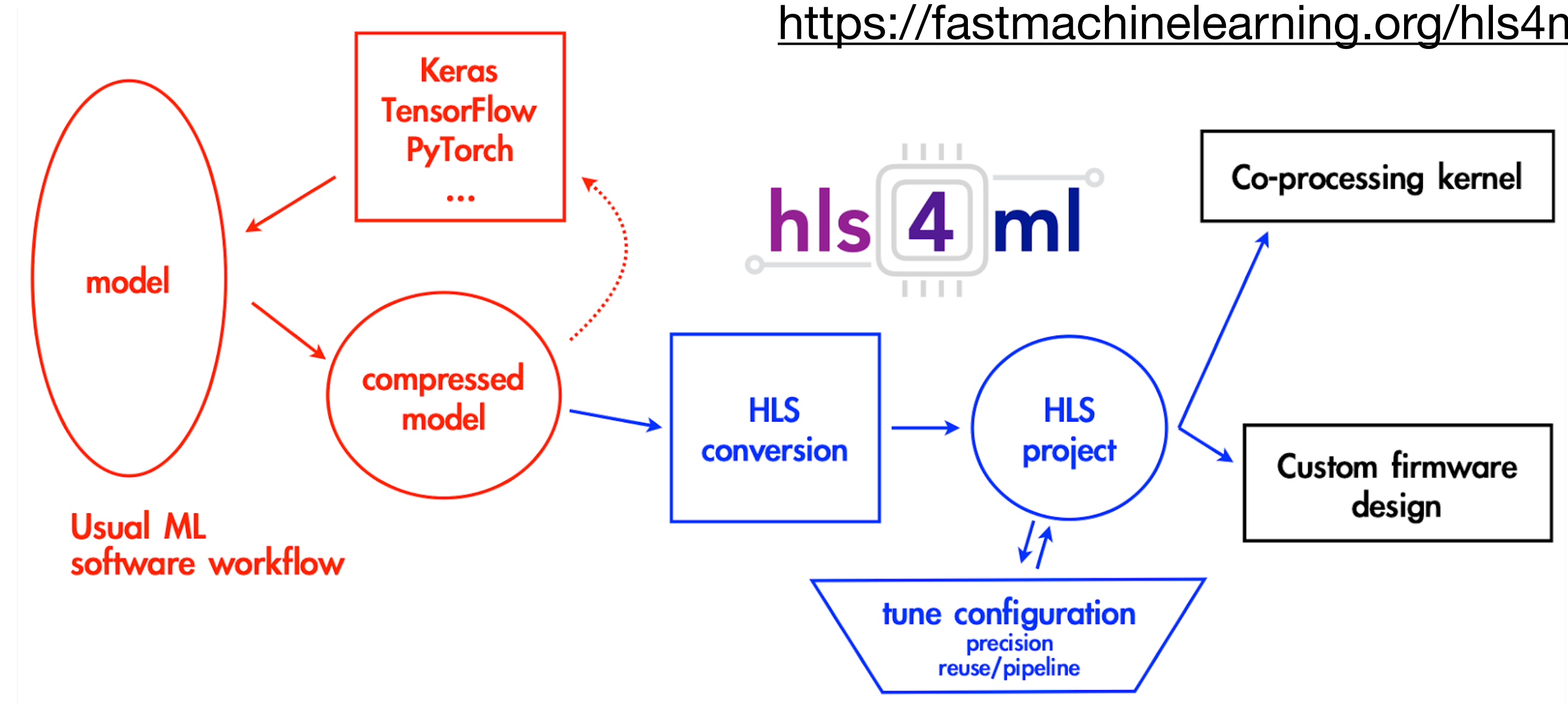
AI on FPGA: skillset & toolchains



- **Engineers**, typically also designers of custom electronics systems
 - VHDL framework firmware: data handling (links), clocking etc.
- **Physicists** (including students)
 - HLS (C++) for non-ML algorithms e.g. clustering, Kalman filter ...
 - ML algorithms using high-level tools
 - VHDL *glue* together e.g. framework → hand crafted variables → ML model
- **Toolchain**
 - Currently main expertise AMD (Vivado/Vitis) — Stratix before ~2015

FastML tools: hls4ml

<https://fastmachinelearning.org/hls4ml/>

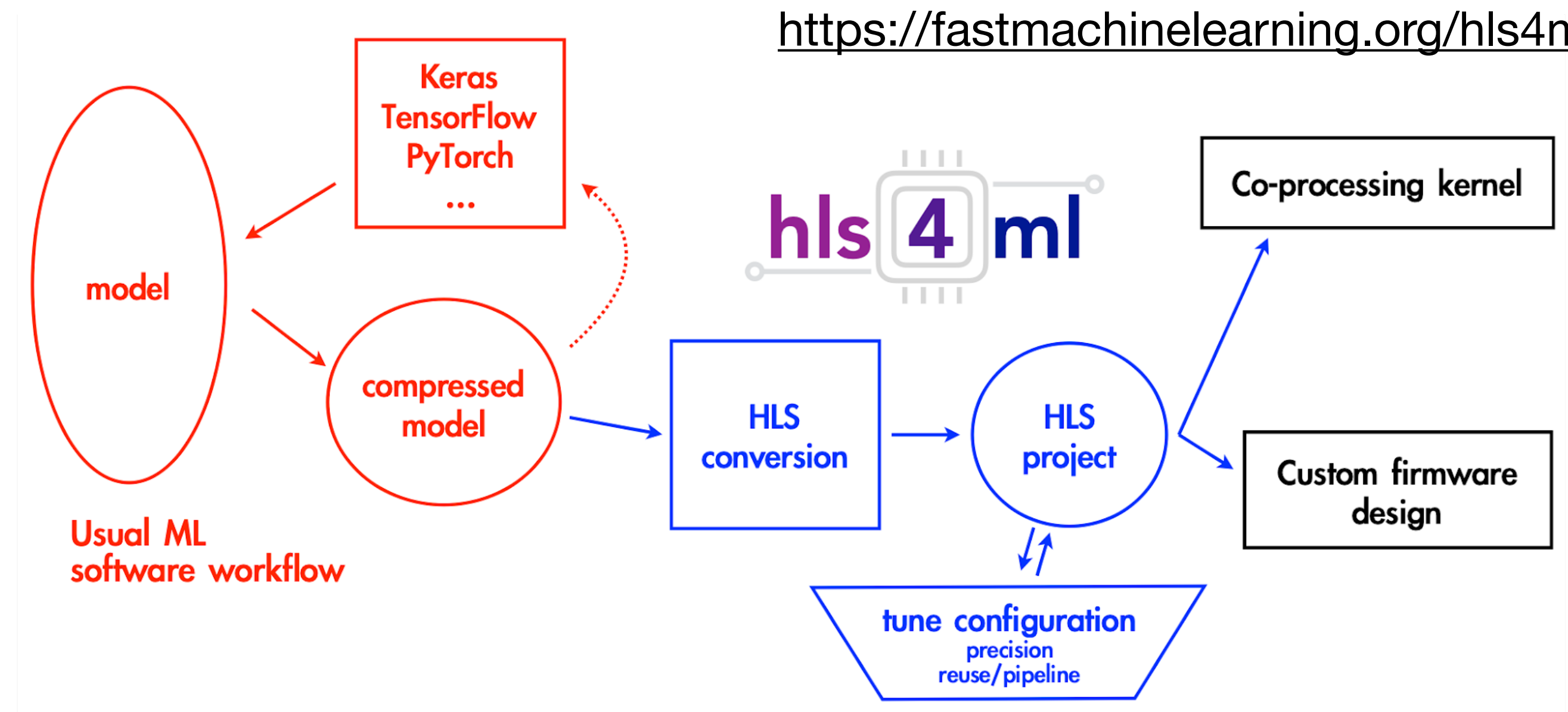


- Take ML models from your favourite framework
- Convert to HLS C++ → different targets (backends)
- Build IP core → integrate into board firmware

[arXiv:2512.01463](https://arxiv.org/abs/2512.01463)

FastML tools: hls4ml

<https://fastmachinelearning.org/hls4ml/>



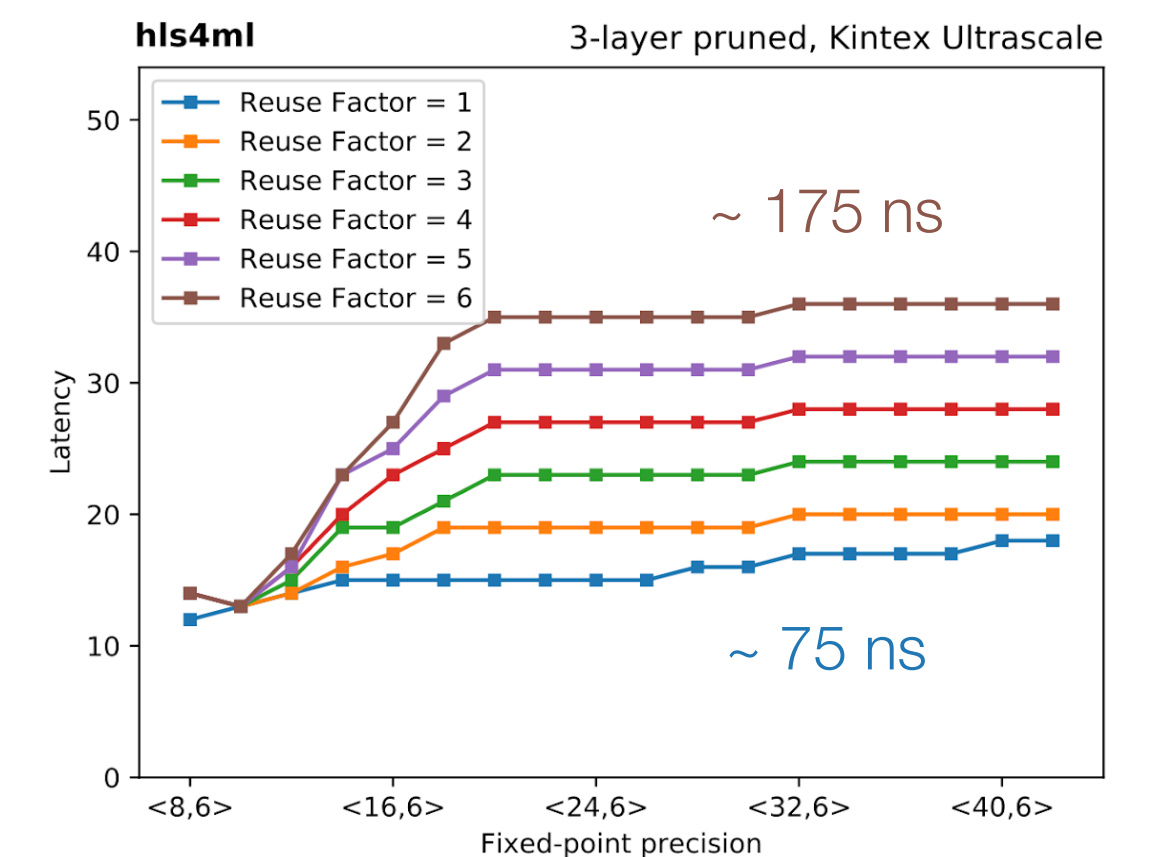
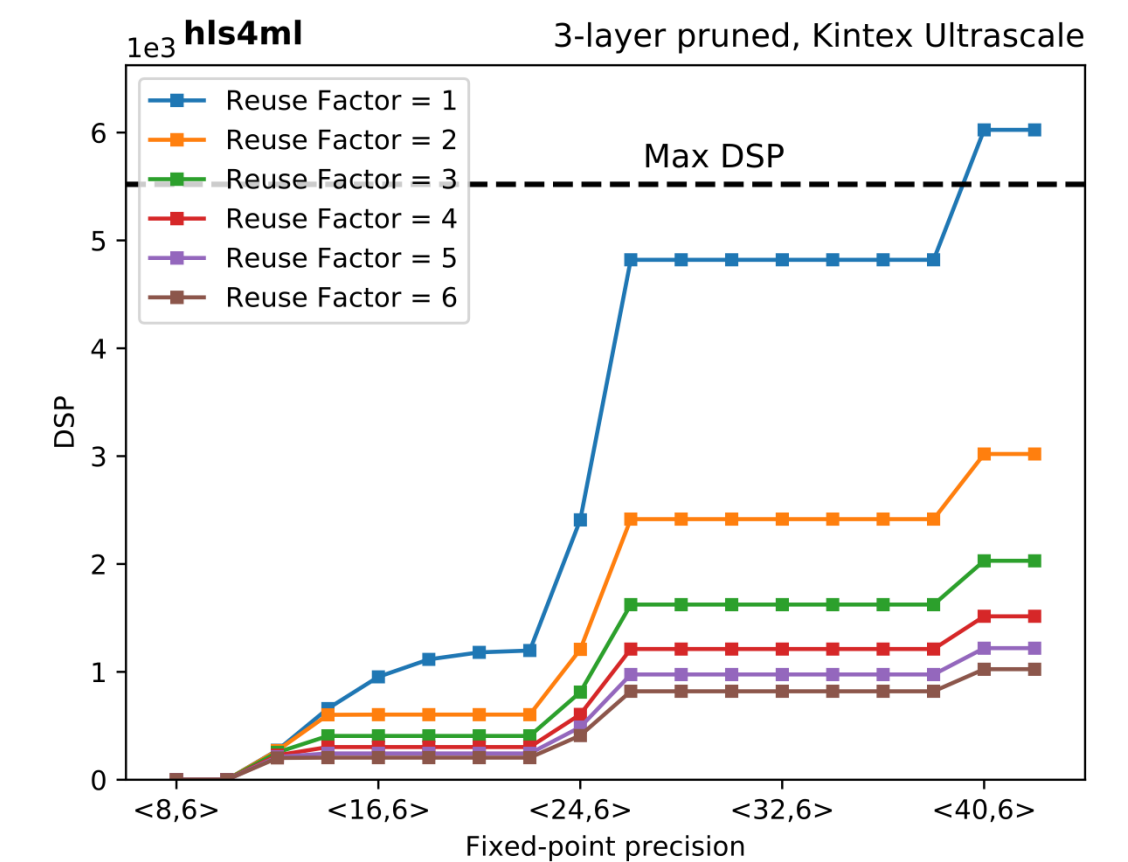
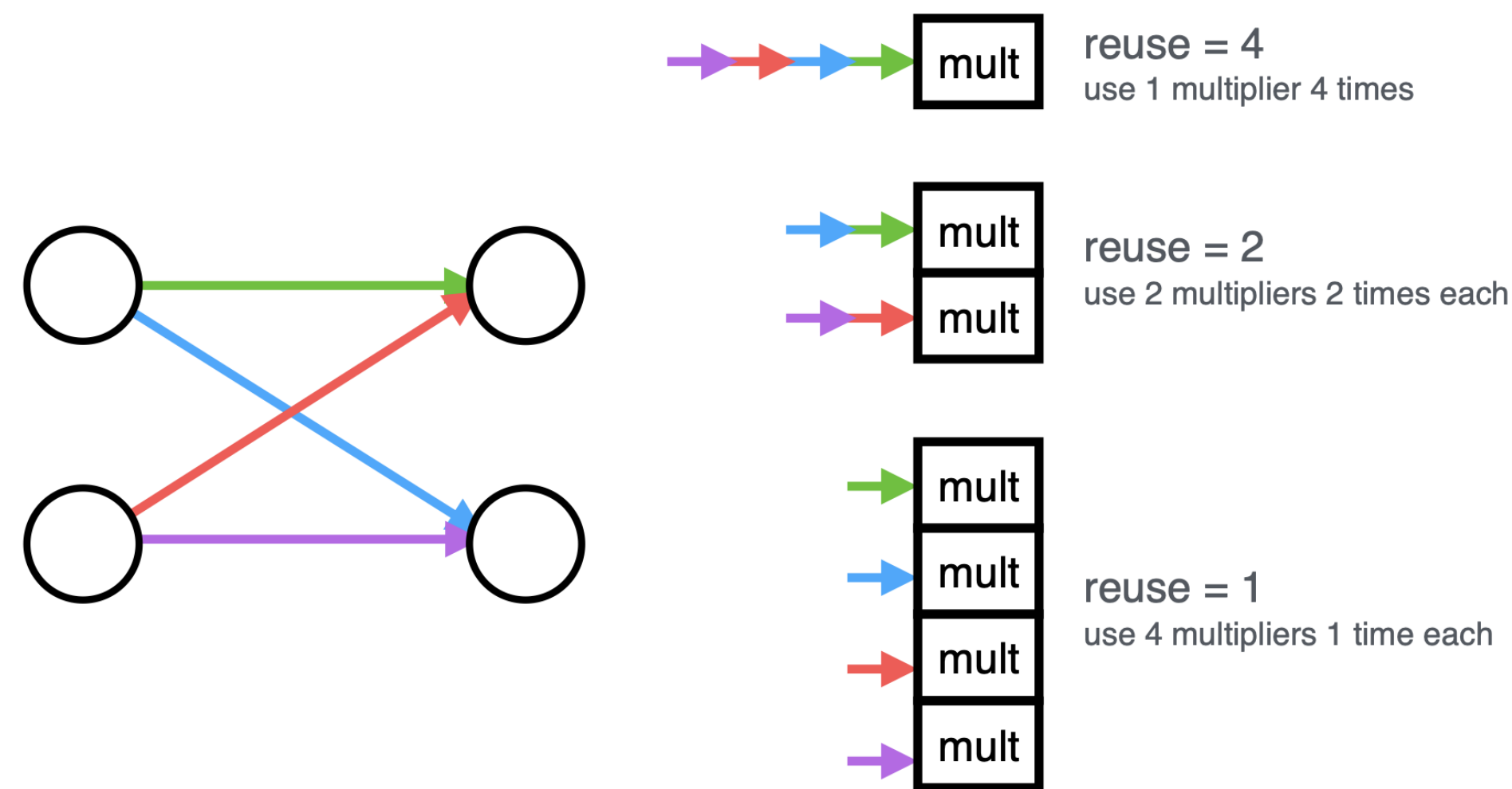
- New features — depth-wise convolutions, high granularity quantisation support, working on transformer support
- Code optimisation based on feedback from Altera of the **OneAPI** backend — efficient placement of variables
- Feedback to Altera on single bit fixed point support

[arXiv:2512.01463](https://arxiv.org/abs/2512.01463)

FastML tools: parallelism

Control over resources → necessary to optimise firmware core

Reuse factor — unrolling controls parallelism



<https://github.com/fastmachinelearning/hls4ml-tutorial>

FastML tools: precision

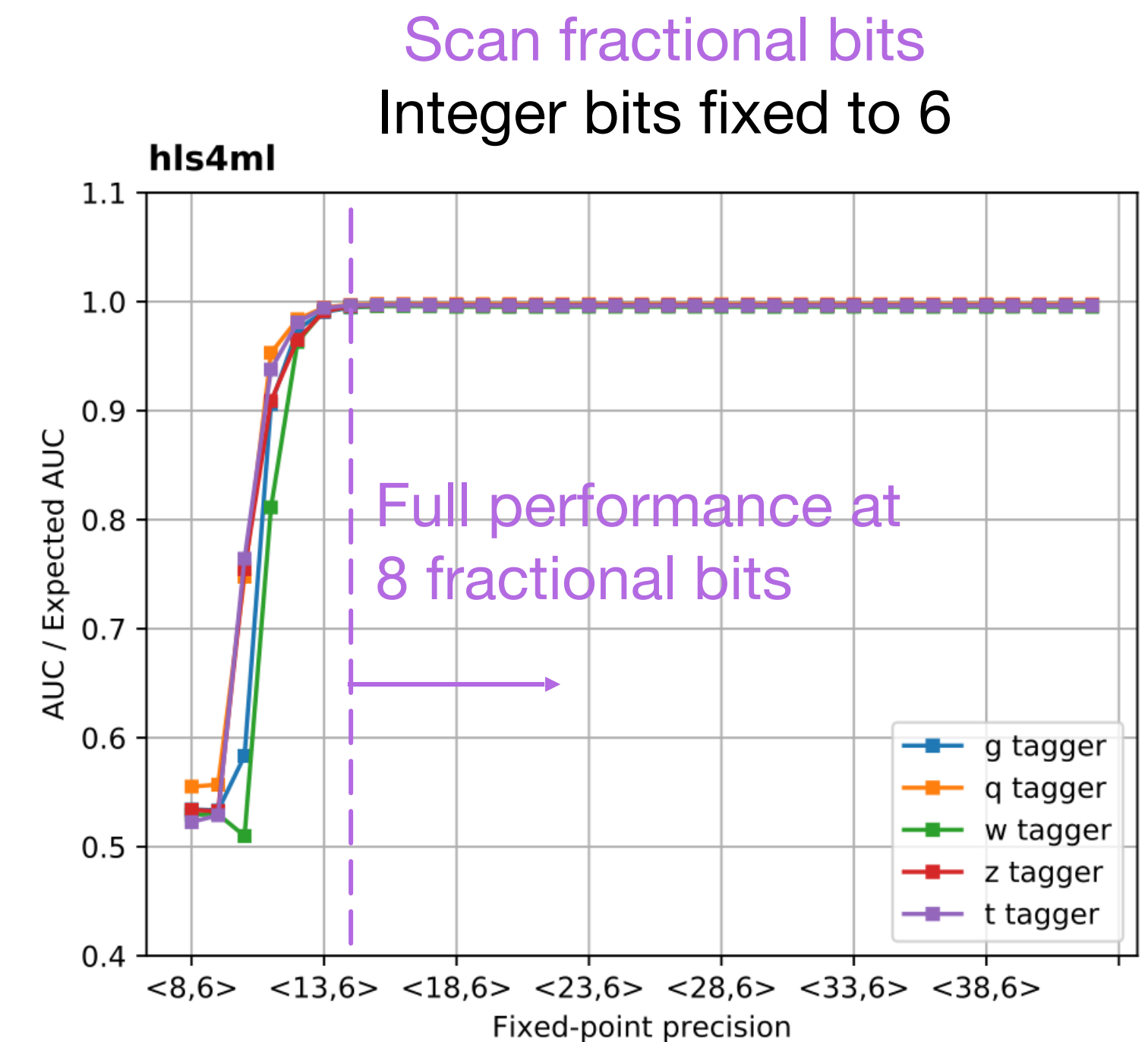
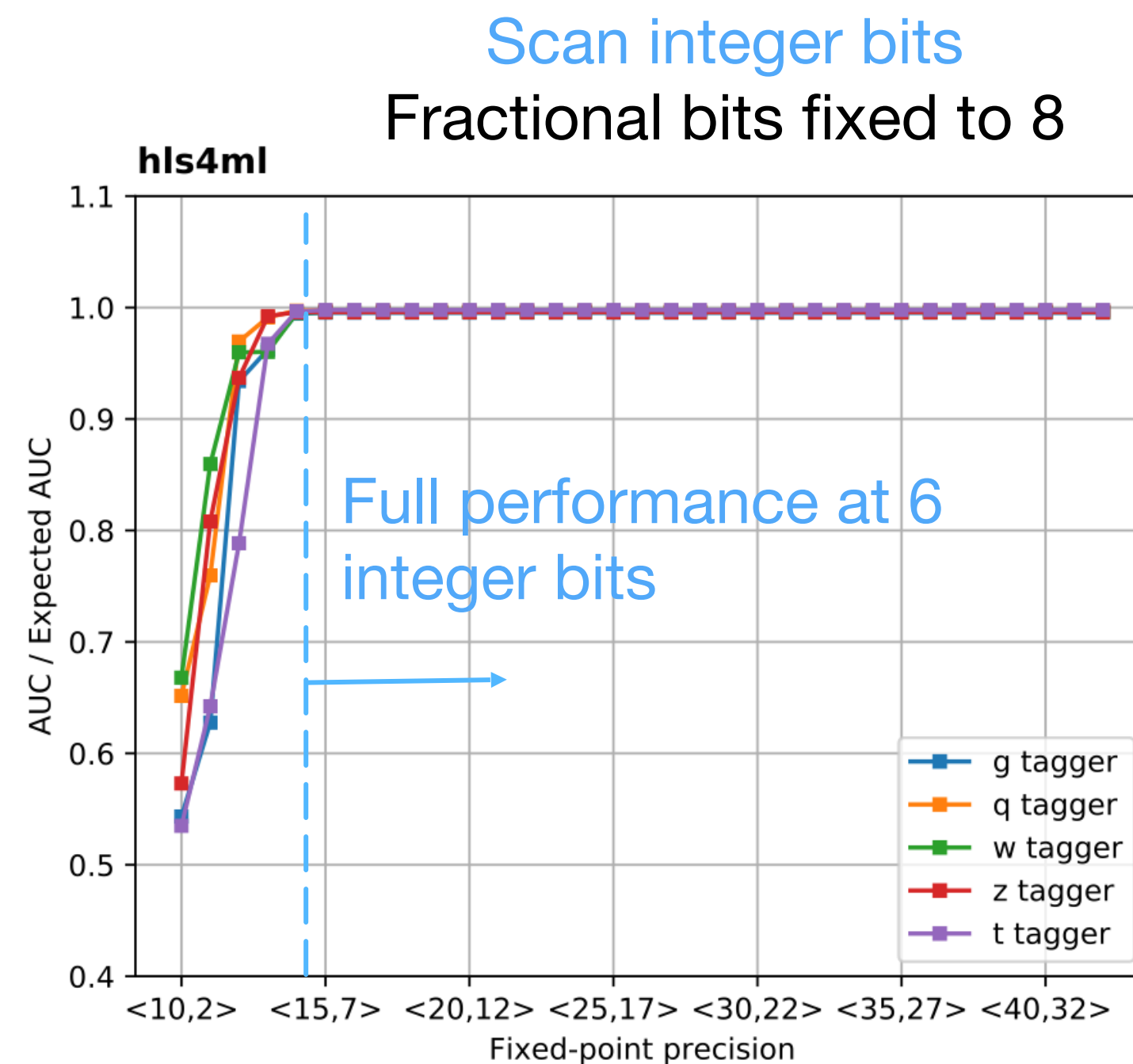
Control over resources → necessary to optimise firmware core

<width bits, integer bits>
0101.1011101010
integer fractional
width

Precision — tune to
give optimum data type
for model

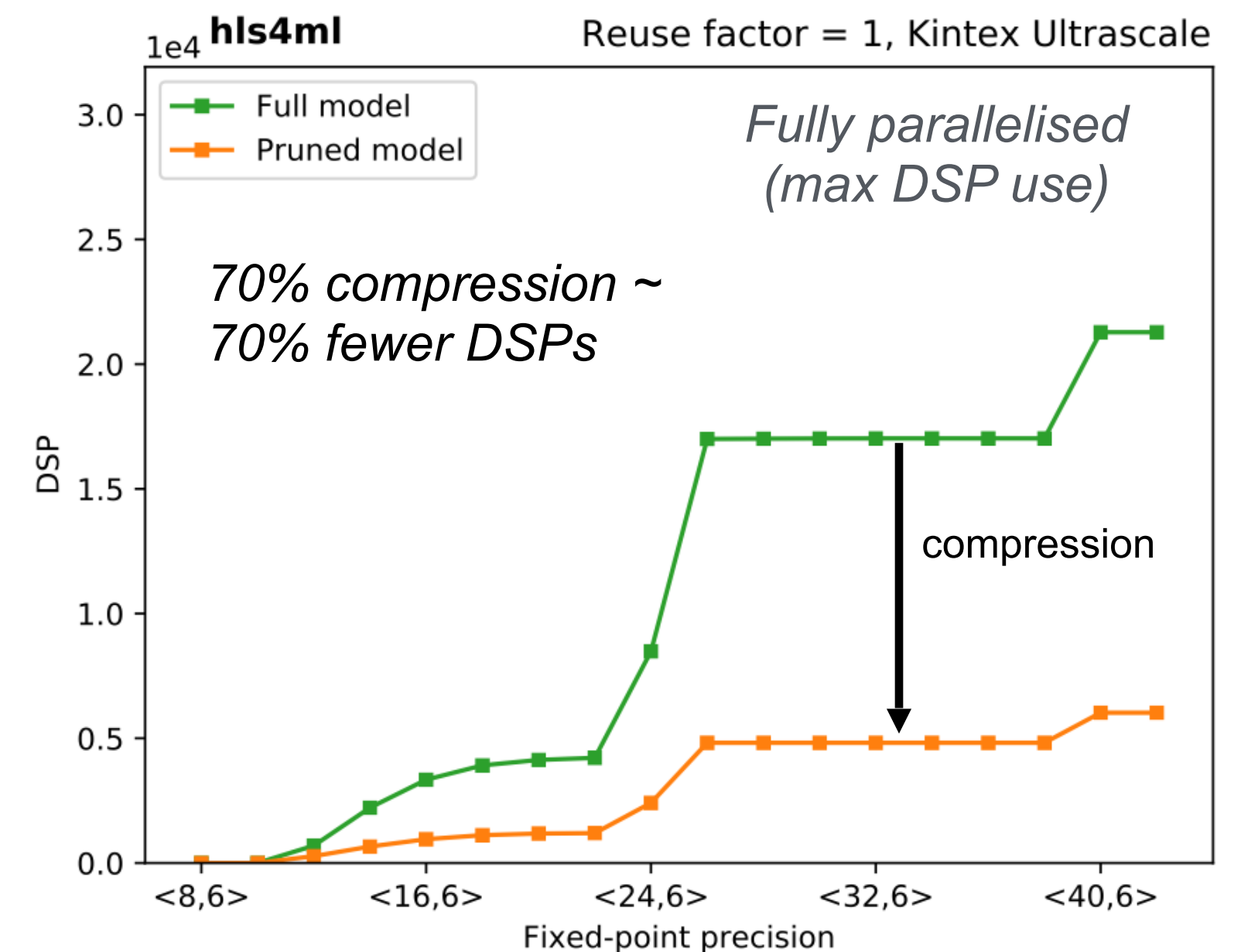
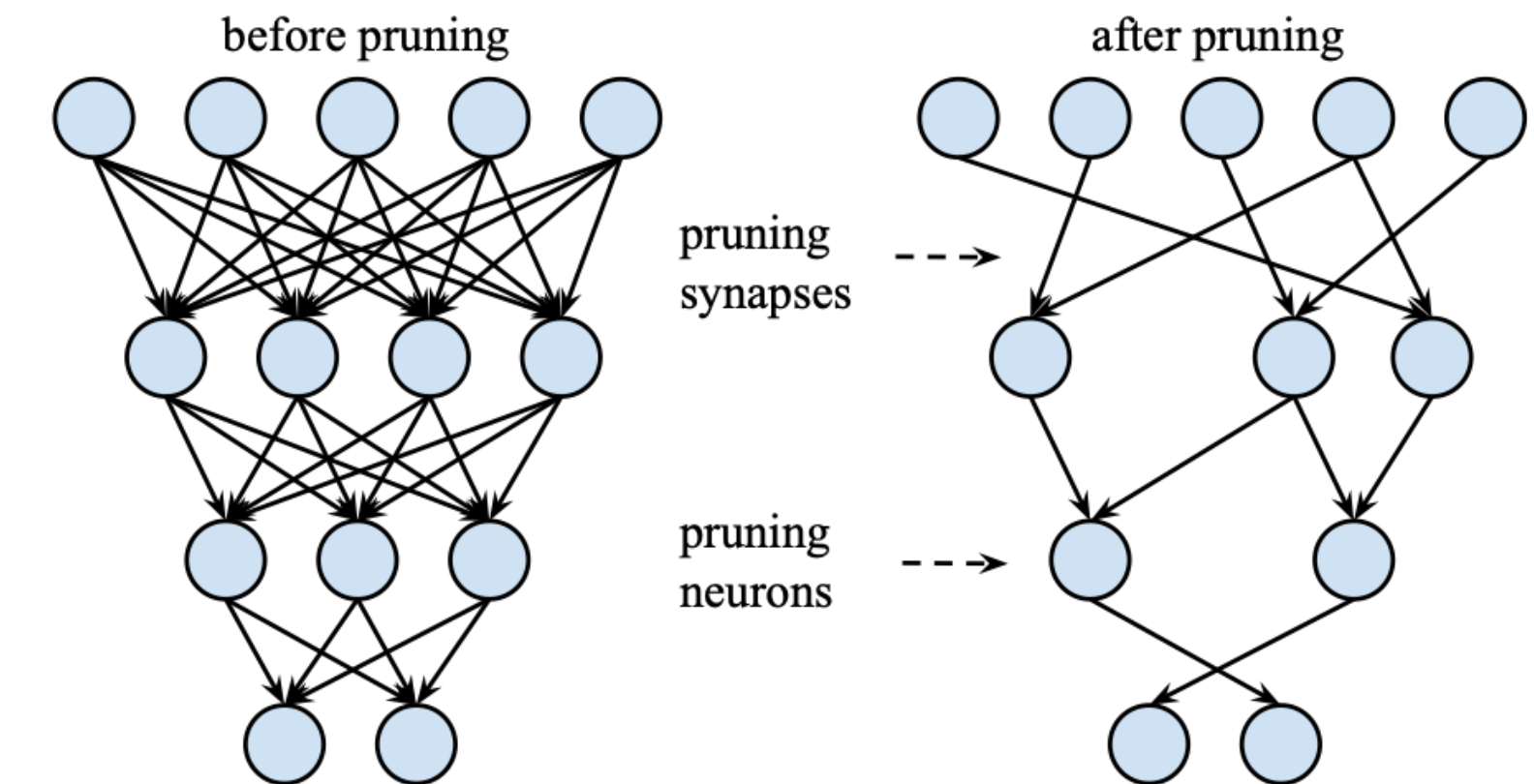
Weights and activations

<https://github.com/fastmachinelearning/hls4ml-tutorial>



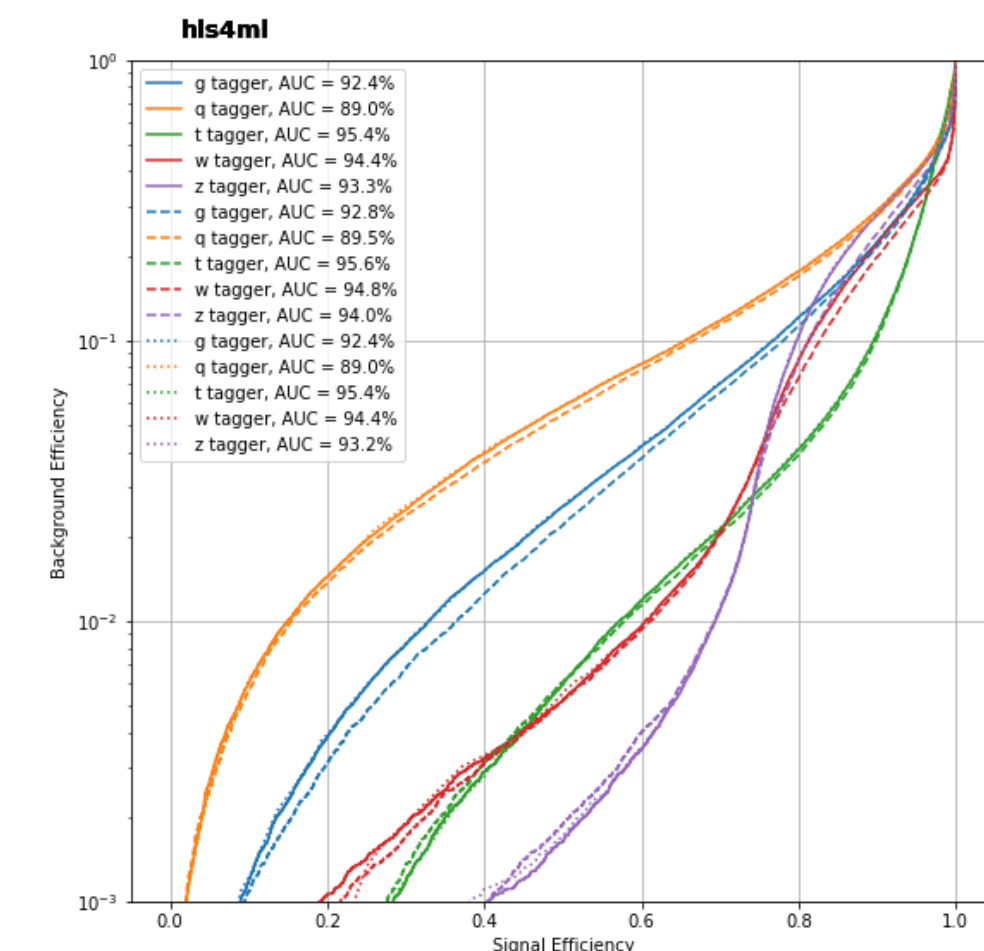
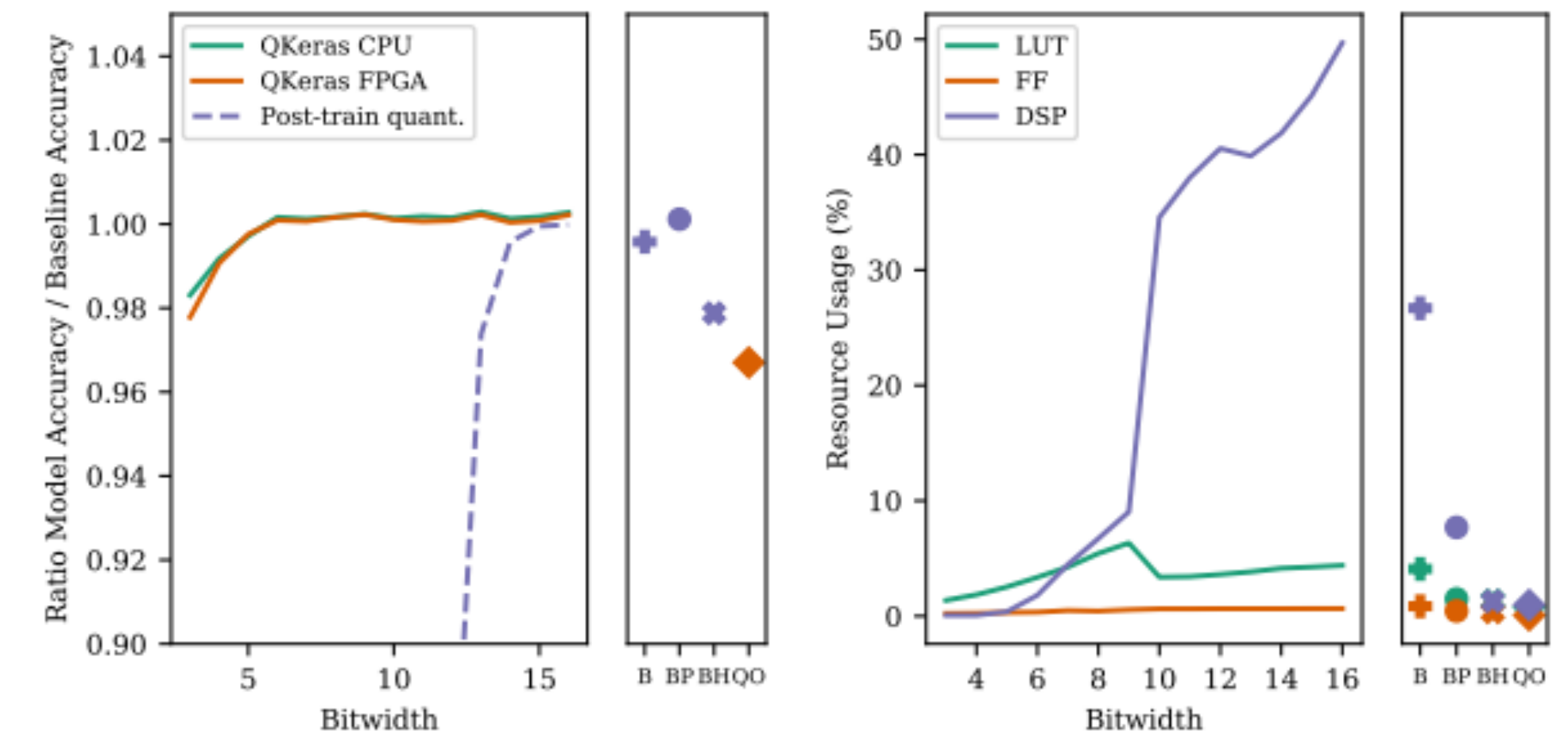
FastML tools: pruning

- Standard technique in ML ops
 - Remove weights, nodes, layers with low weights
- Check accuracy
- Repeat while accuracy loss is acceptable
- Large reduction in resource usage possible while retaining high accuracy



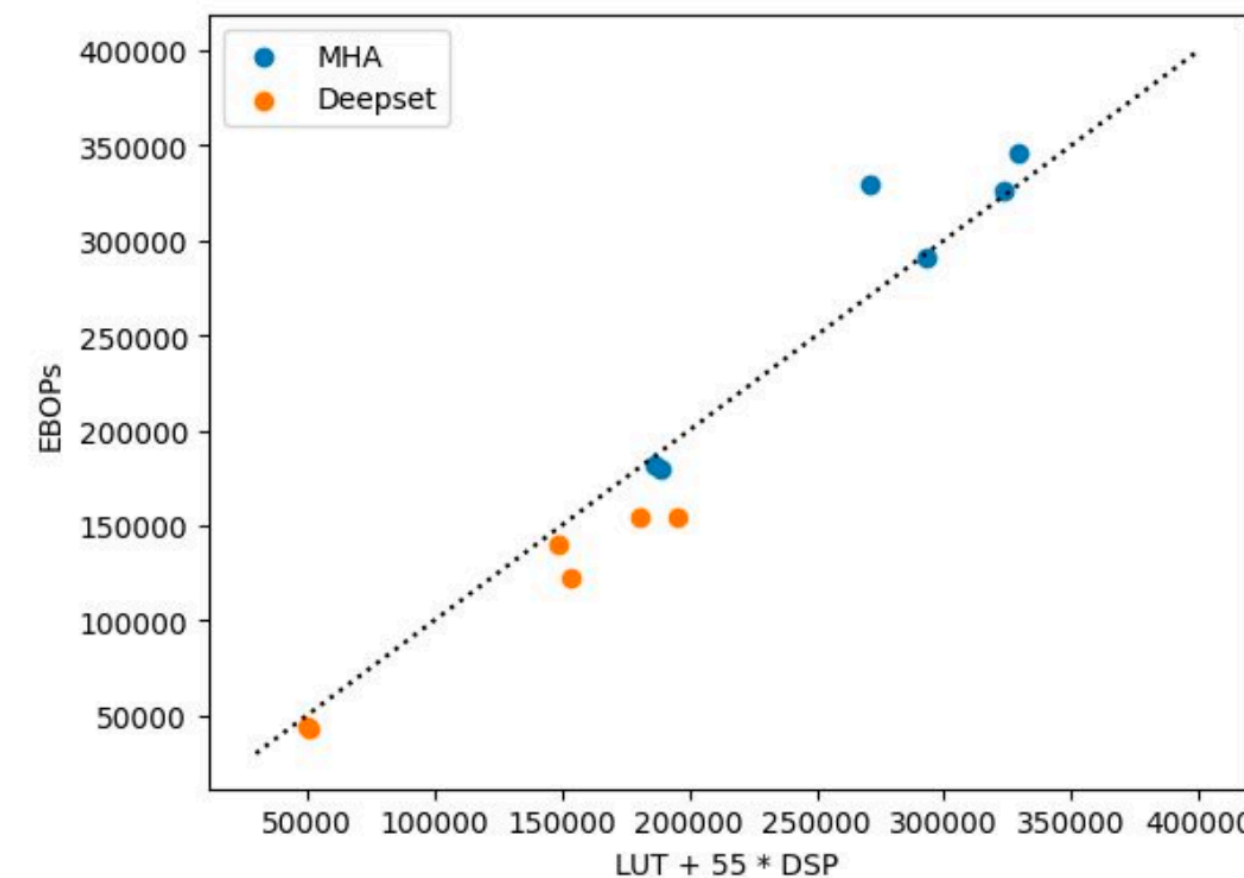
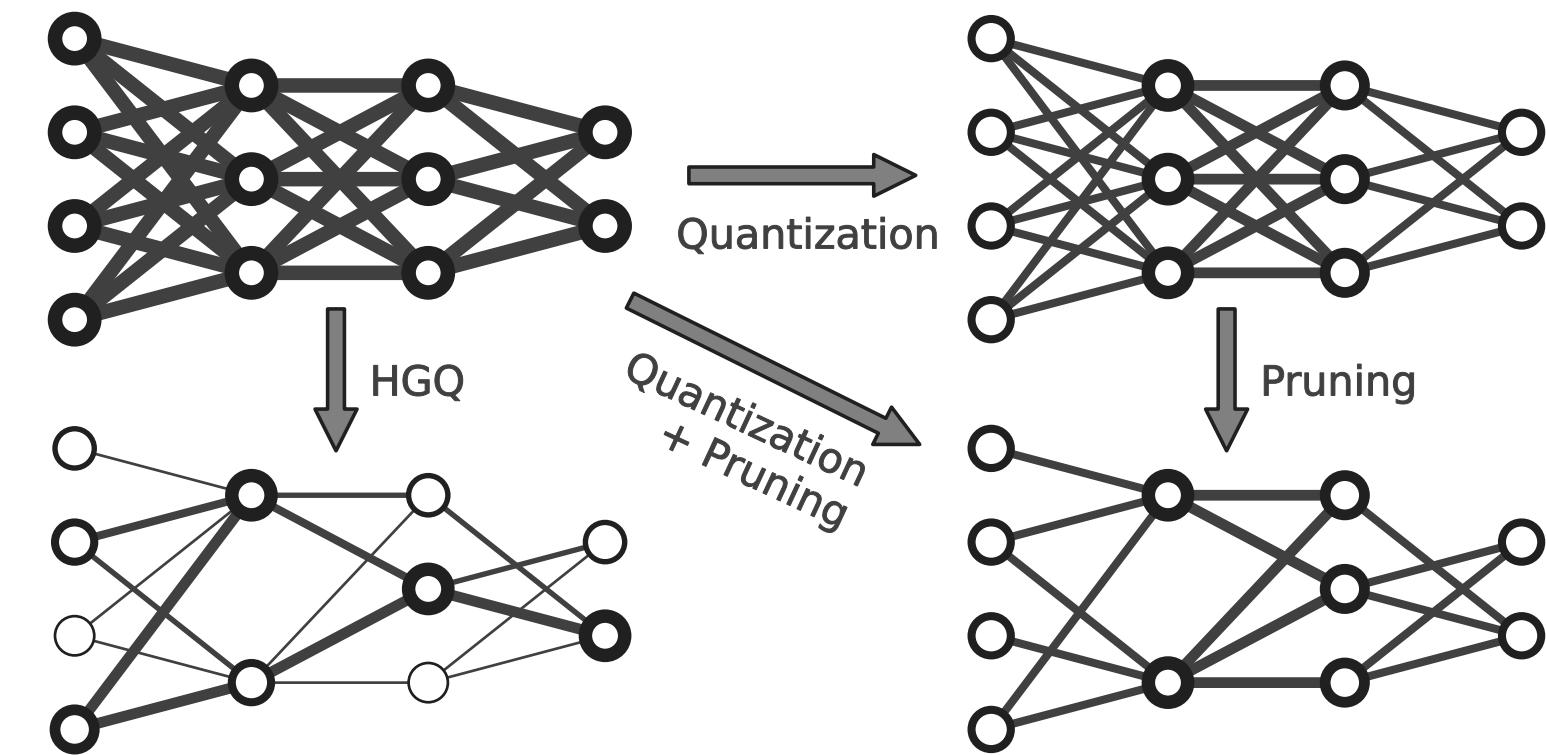
FastML tools: QAT

- **Quantisation Aware Training**
 - More sophisticated optimisation of precision
 - Model quantisation is included in training forward pass → training can adapt robustly to quantised data types
 - Provides better optimisation than post-training quantisation
 - Example using QKeras (Google)



FastML tools: HGQ

- **High Granularity Quantisation**
 - Combination of pruning and quantisation
 - Quantisation per weight, bias and activation — high granularity
 - Bit widths optimised during training — pruning occurs automatically when width = 0
 - Incorporates estimate of resource usage in training (EBOPS)

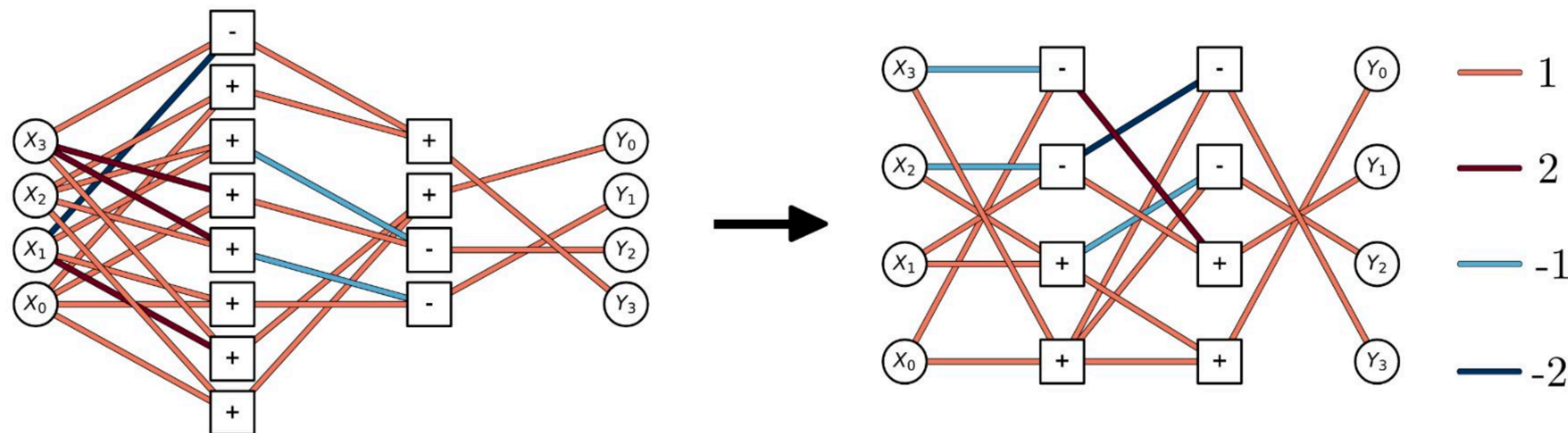


$$\mathcal{L} = \mathcal{L}_{\text{base}} + \beta \cdot \overline{\text{EBOPS}} + \gamma \cdot \text{L1}_{\text{norm}}$$

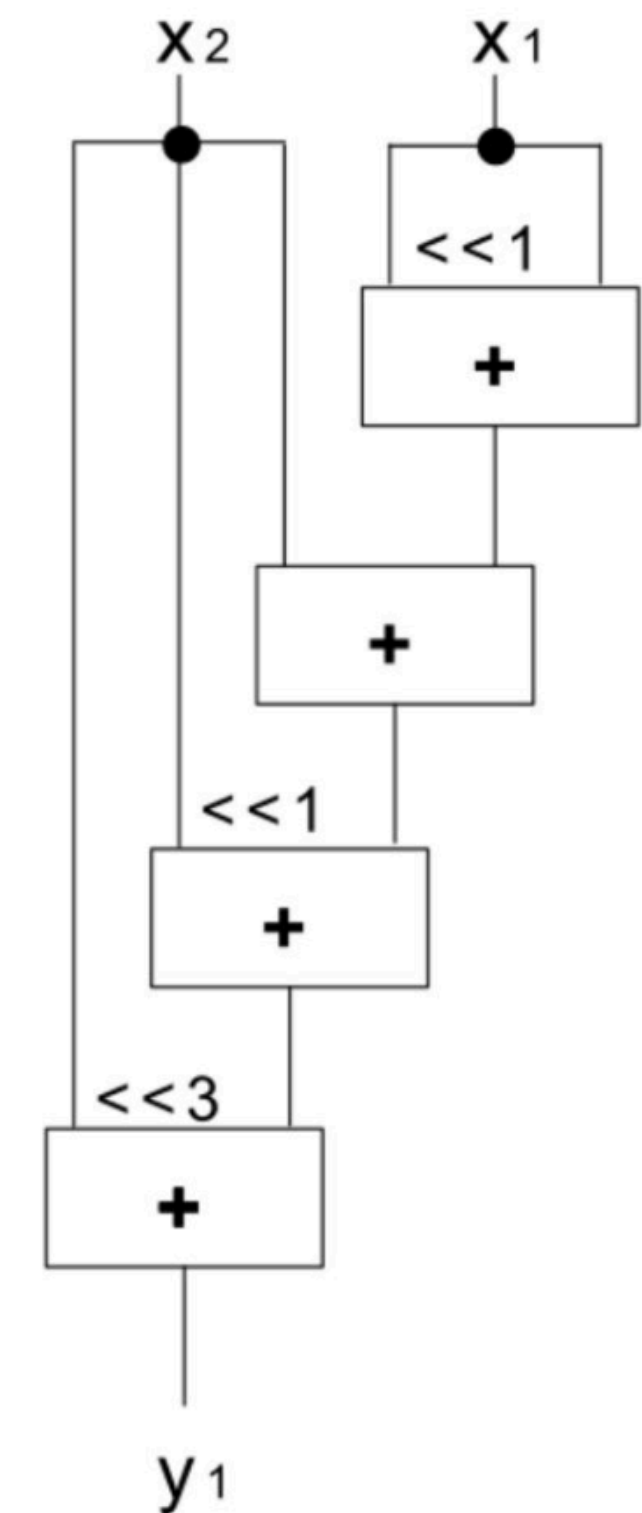
[arXiv:2405.00645](https://arxiv.org/abs/2405.00645)

FastML tools: DA

- Constant Matrix Vector Multiplication (CMVM) operations optimised using **Distributed Arithmetic**
- Example: splits multiplications into additions and bit-shifts to further reduce the FPGA resource usage after quantisation
- Also other more sophisticated optimisations



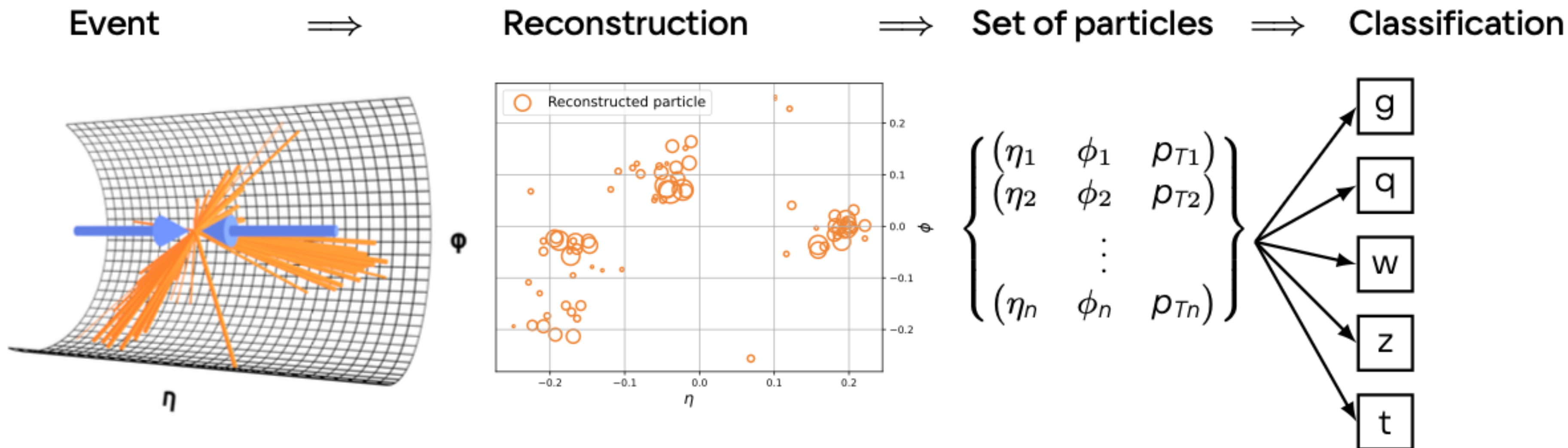
$$y_1 = 3x_1 + 11x_2$$



[arXiv:2507.04535](https://arxiv.org/abs/2507.04535)

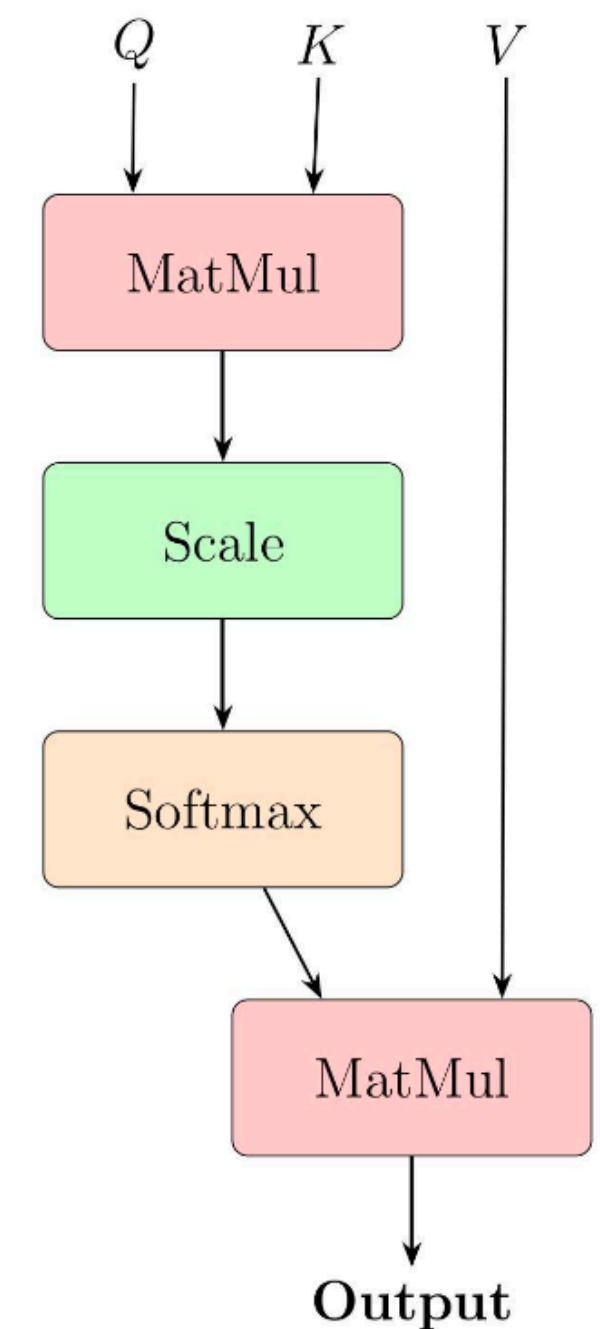
Future algorithms: transformer

- Example: Jet tagging $\rightarrow$ examine attention between vectors of particles to classify combination



- Large numbers of particles — complexity grows $O(n^2)$

5

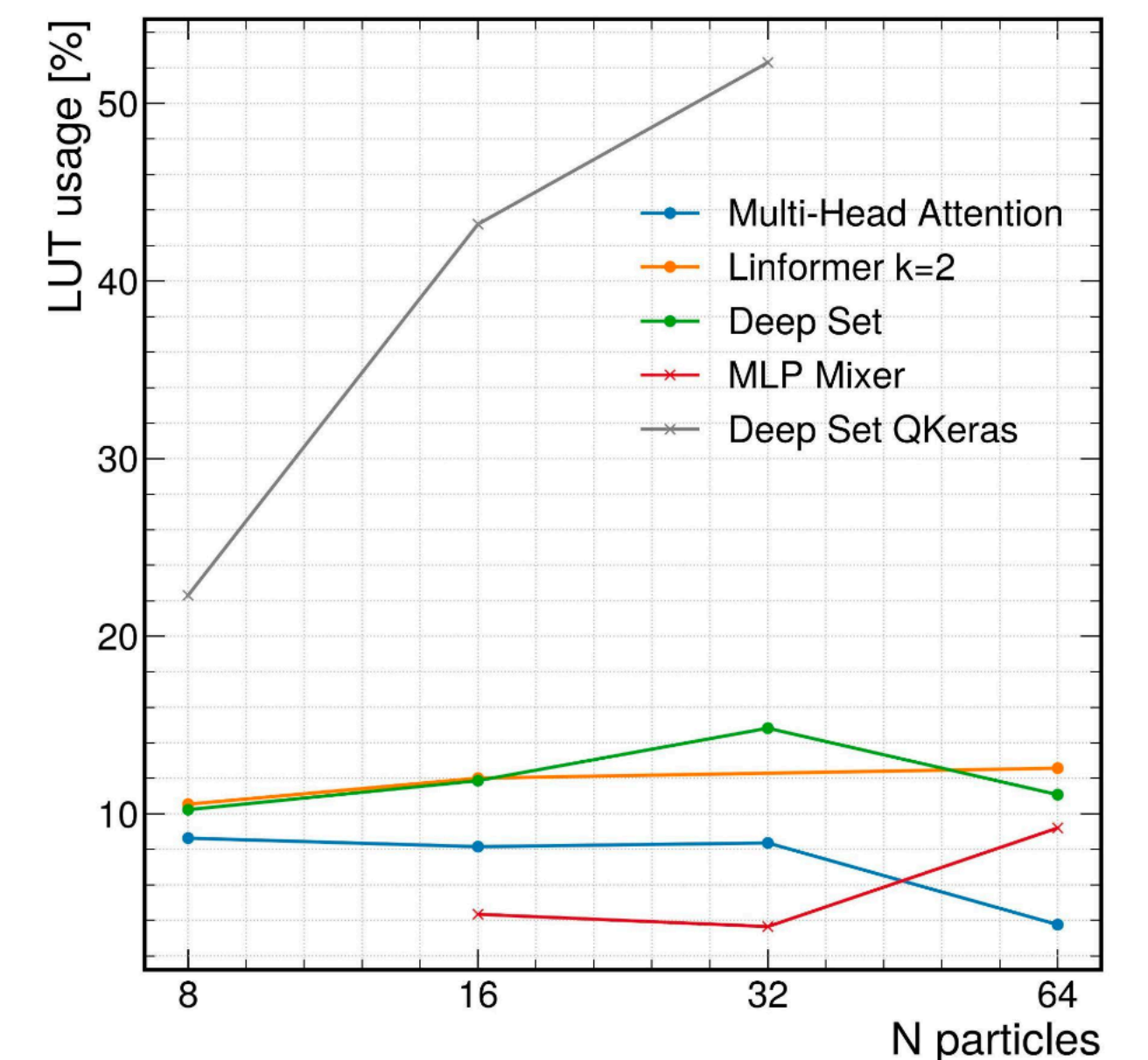
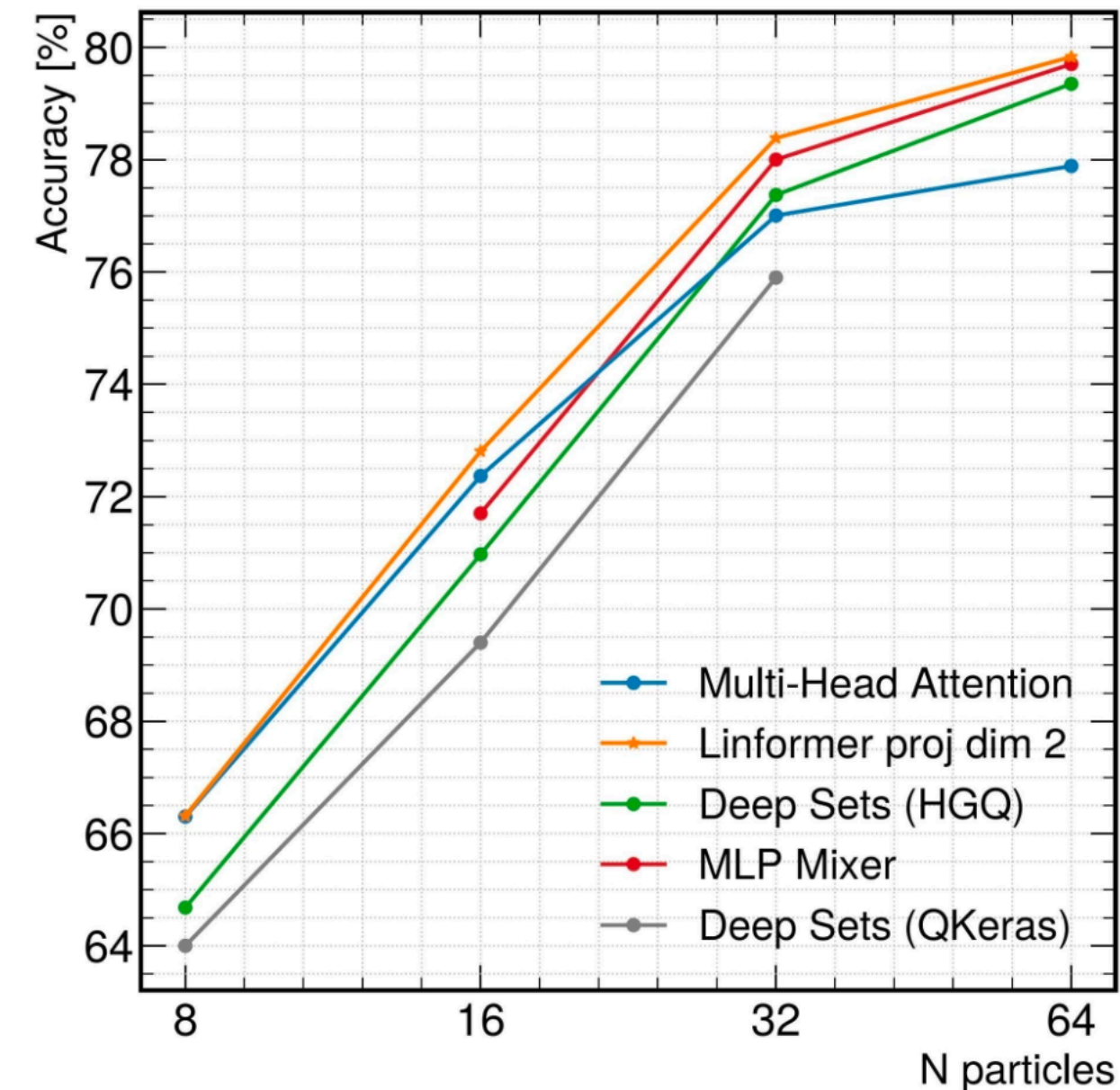
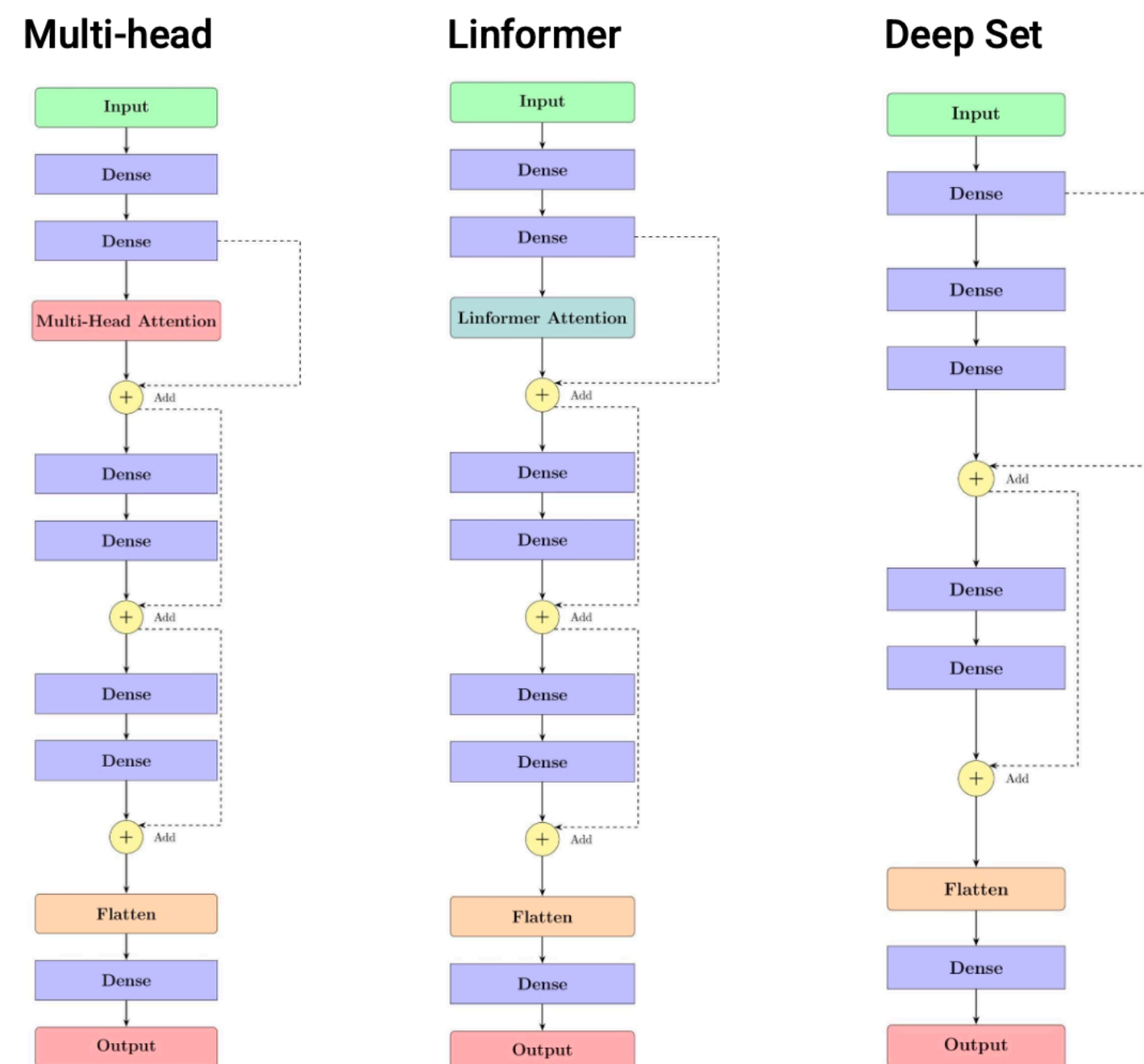


[arXiv:2510.24784](https://arxiv.org/abs/2510.24784)

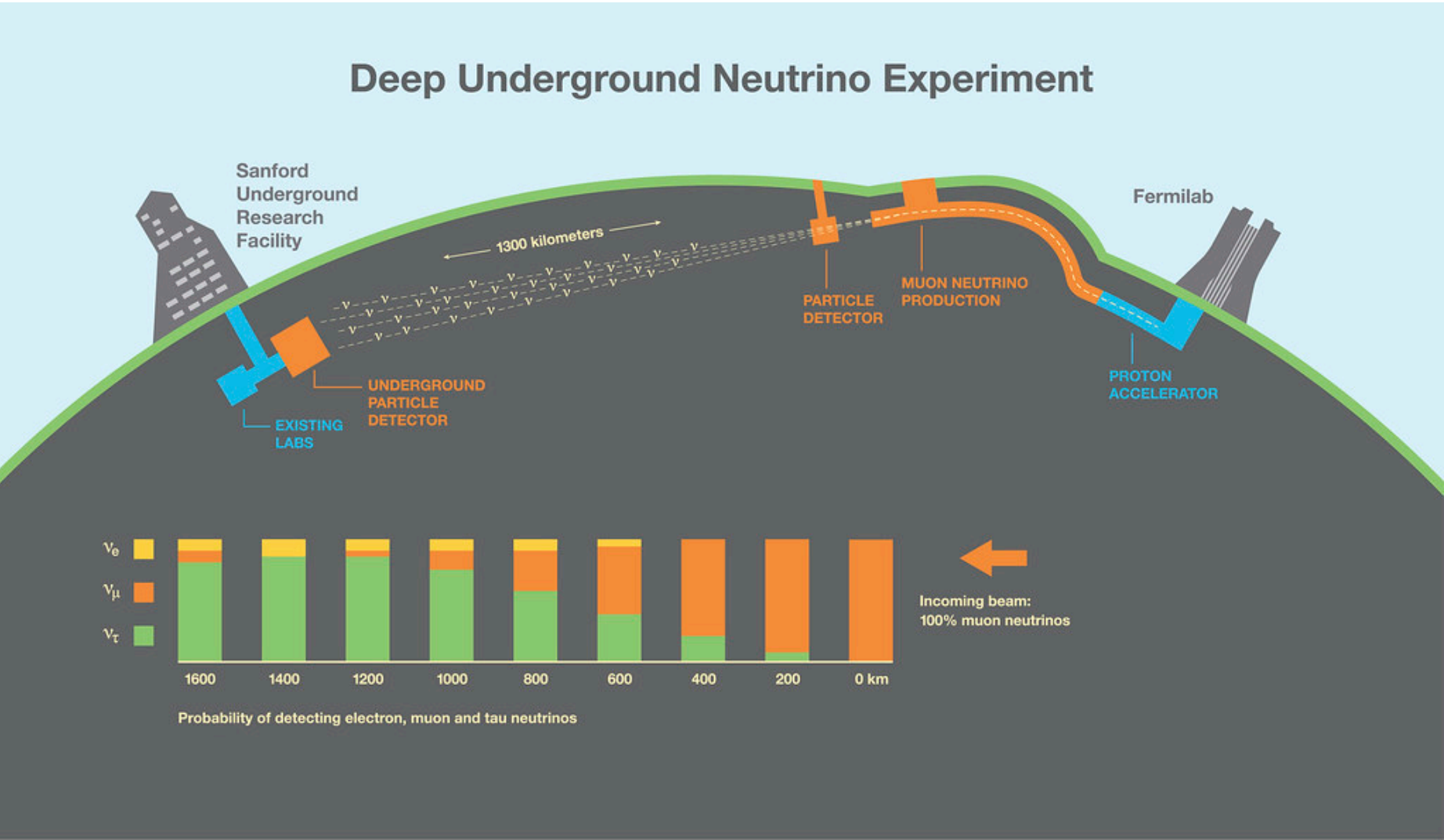
Future algorithms: transformer

- Using the tools discussed:
 - HGQ & DA — fully parallel — ~100 ns latency — 350K EBOPs
- Three low-level input features — five classes output

[arXiv:2510.24784](https://arxiv.org/abs/2510.24784)

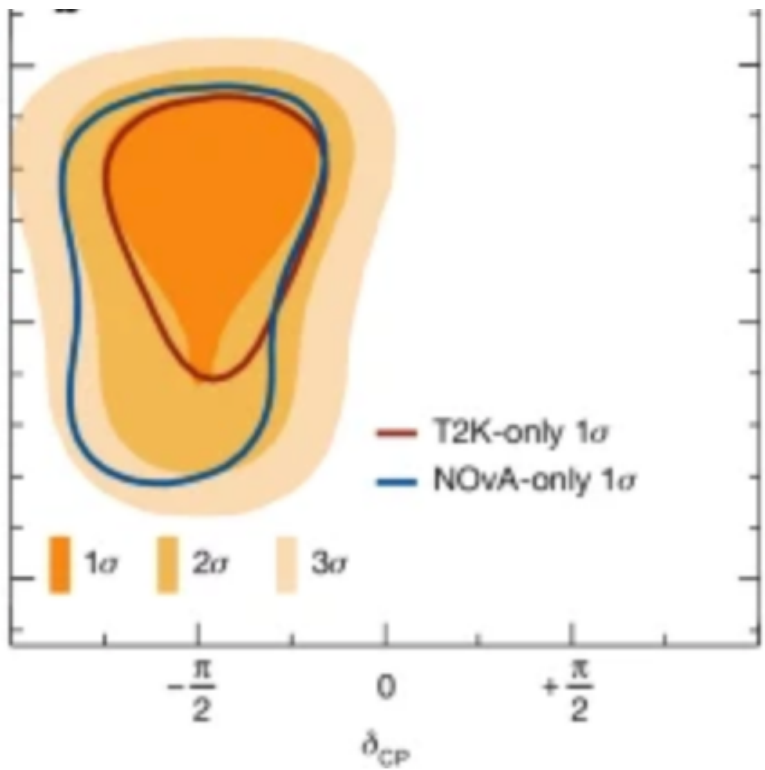


Future experiments

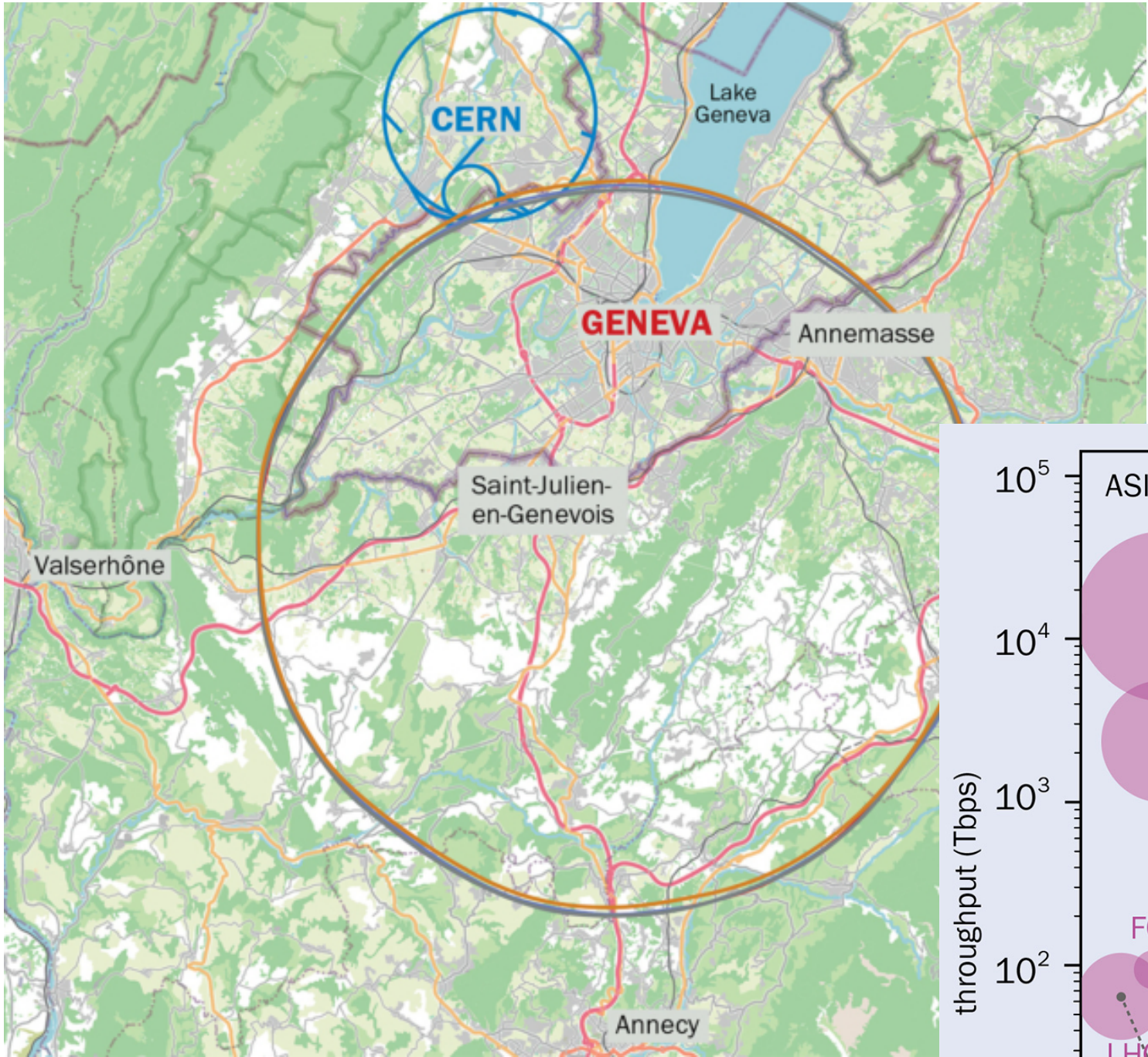


Altera

Flagship US neutrino experiment — 2030+

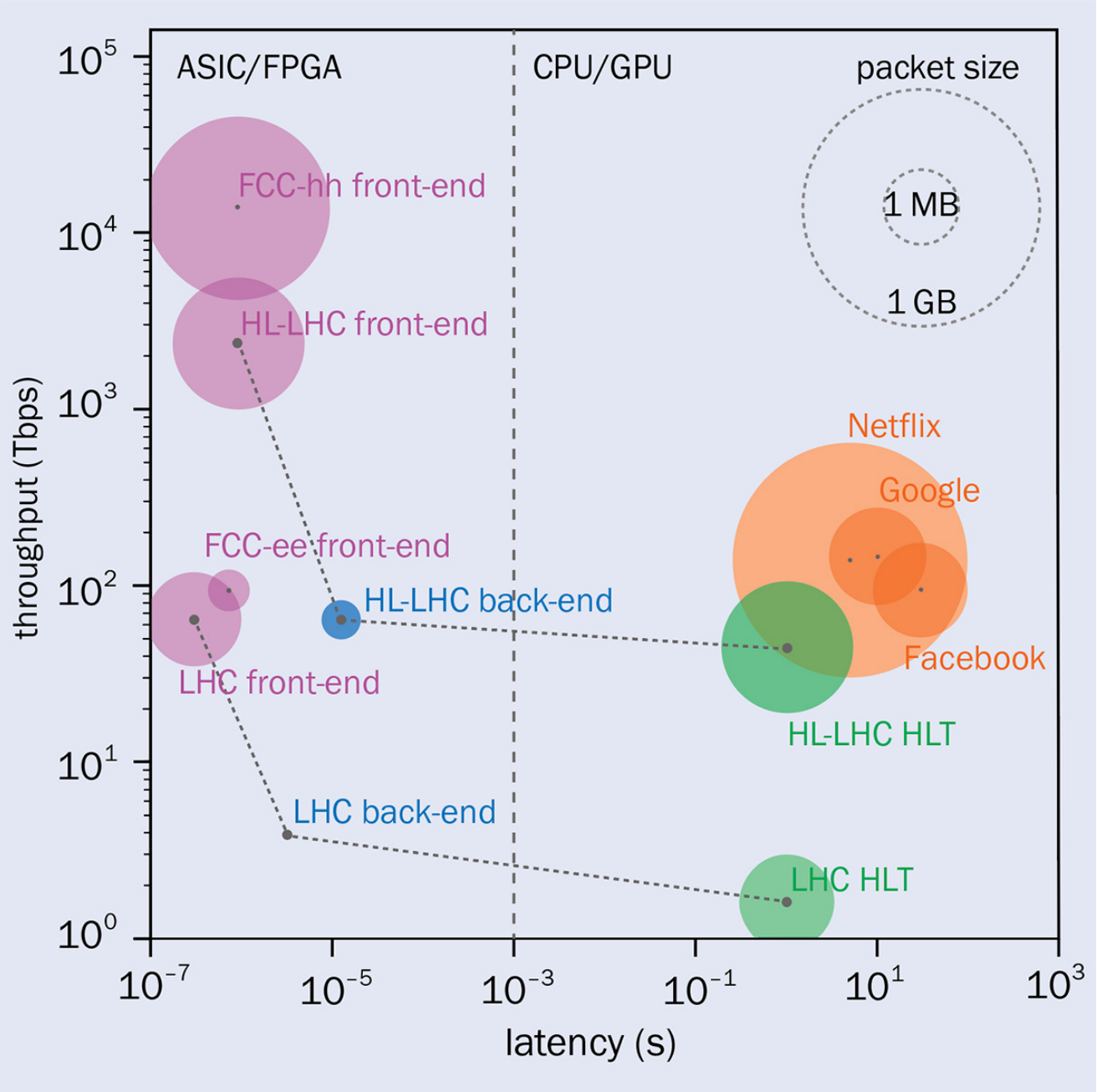


<https://lbnf-dune.fnal.gov/>



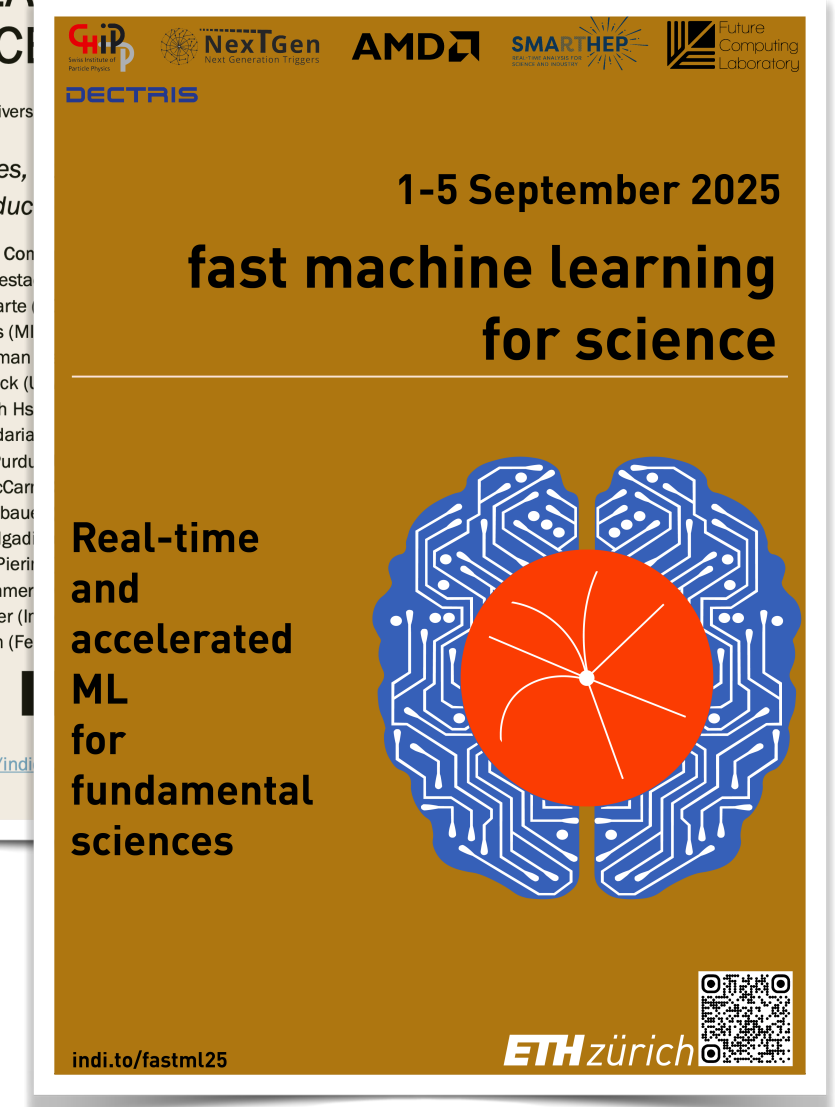
<https://fcc.web.cern.ch/>

Altera



Flagship European proposal — 2040+

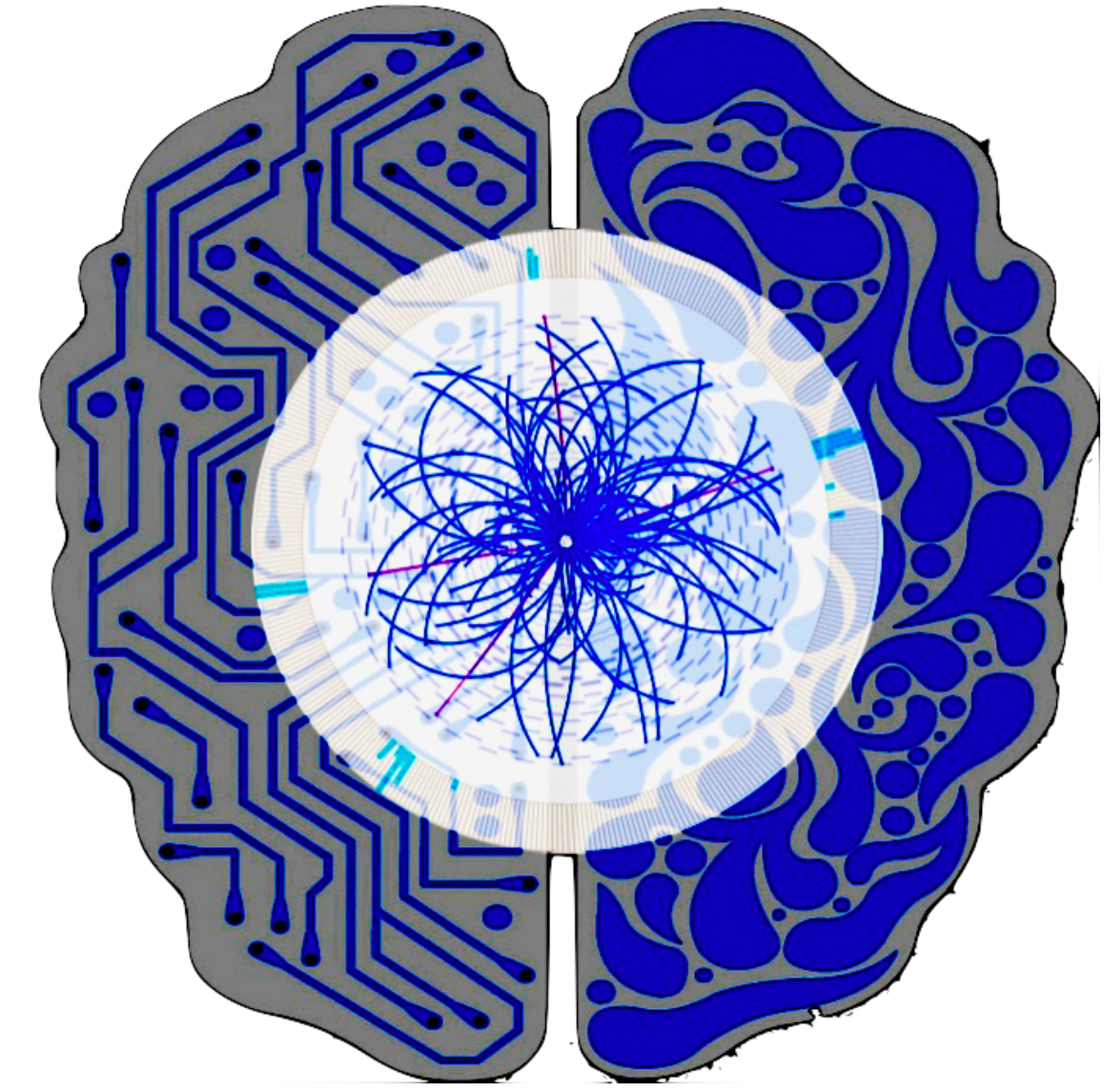
FastML community



- Growing community interested in using fast machine learning for science
- Annual workshop established with strong participation
- Particle physics, neuroscience, biology, fusion, astronomy... **Altera**, AMD, Nvidia, Groq, Graphcore ...

FastML foundation

- Working towards a non-profit foundation to support FastML tools, training and outreach
- Advisory Board with representatives from all academic and industry members
- Currently 30+ members
- Membership through attendance at annual workshop

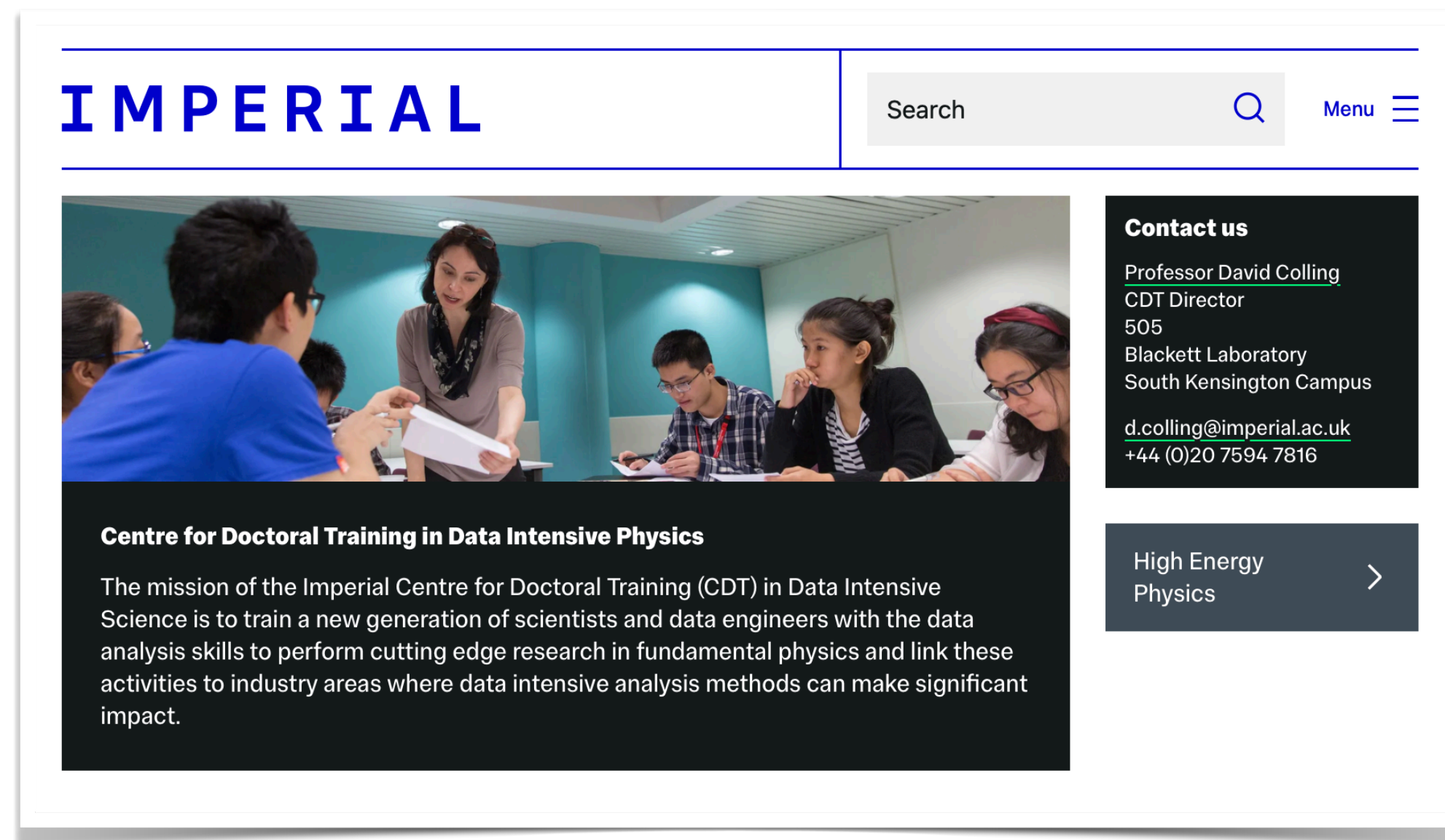
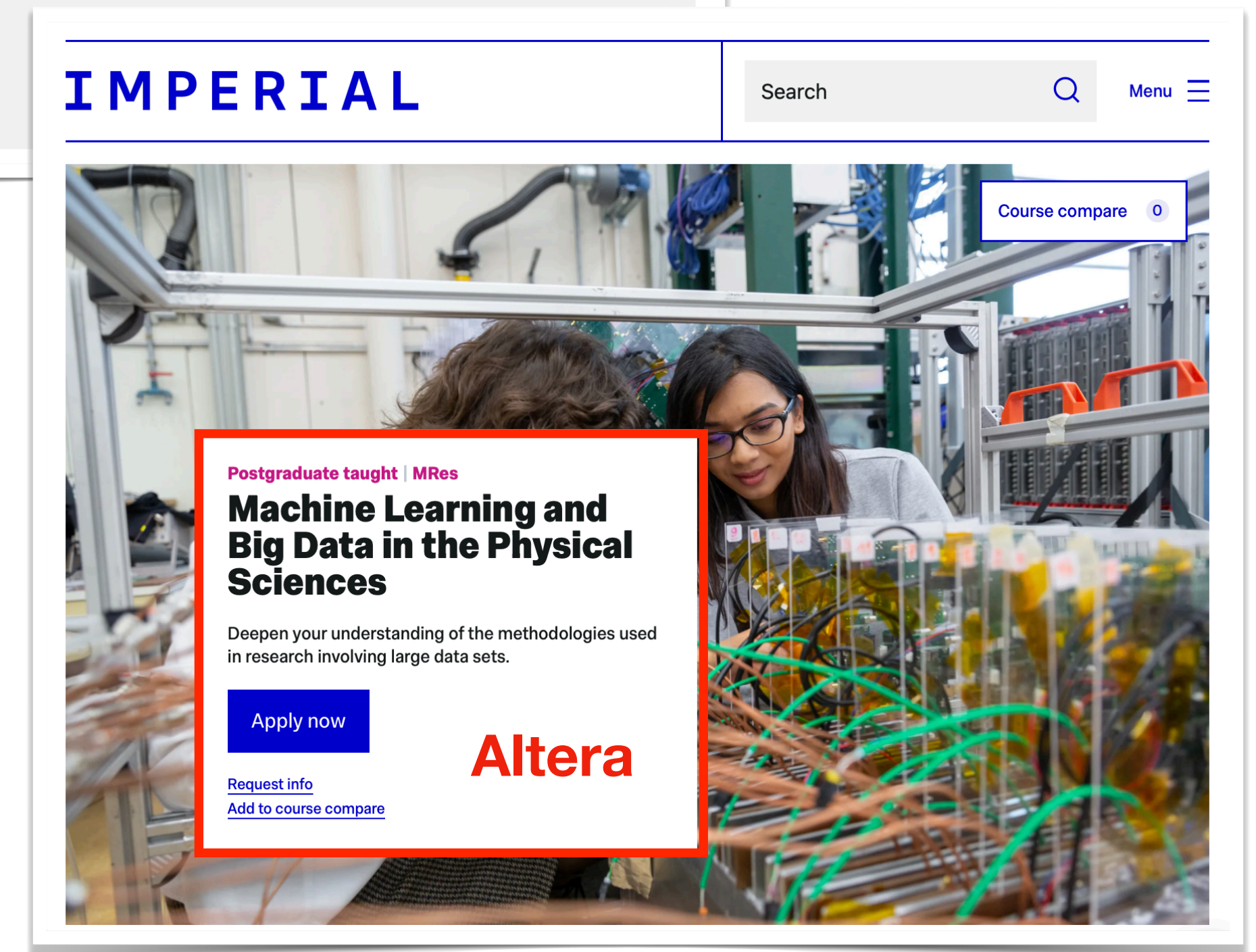
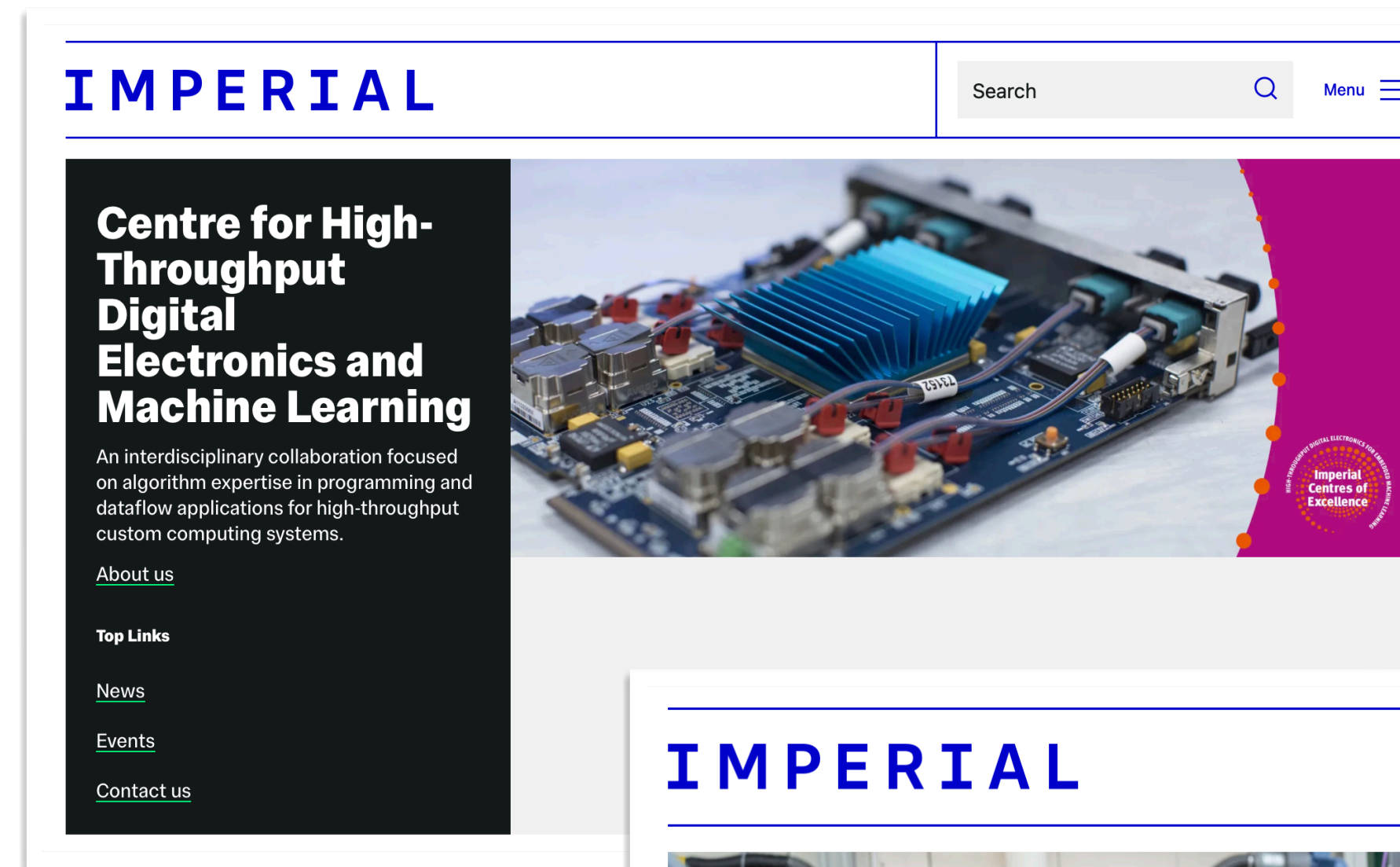


<https://fastmachinelearning.org/>

(to be updated soon ...)

Opportunities

- Masters & PhD student projects
- Input into courses and materials
- Collaborate on 6 or 9 month projects with partners



- Also directly with CERN and other labs e.g. OpenLab

Summary/conclusions

- Broad approach to future challenges: FPGA, GPU, HPC, AI, IaaS ...
 - Machine learning set to be key — low latency niche is a focus
 - Making exciting progress towards goals ~5 years away
- Looking further ahead starting to plan for future experiments > 10 years away
 - Intelligent detectors — processing on detector
 - High-speed machine learning — in network processing ...
 - Quantum algorithm development
- Wide range of opportunities to collaborate with industry partners