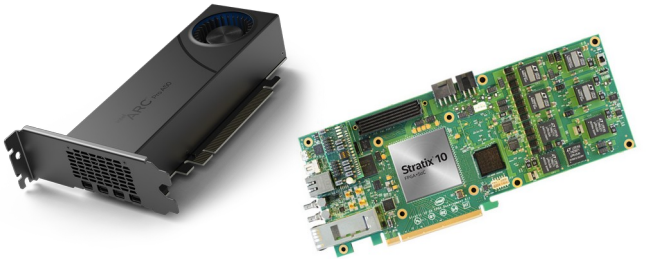
Scientific Computing on Heterogeneous Architectures

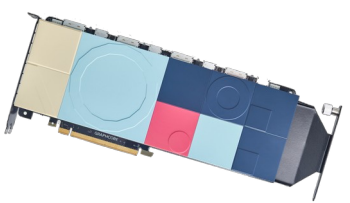
Dorothea vom Bruch
Aix Marseille Univ, CNRS/IN2P3, CPPM, Marseille, France
Email: dorothea.vom.bruch@cern.ch

Thematic CERN School of Computing
June 2023
Split, Croatia

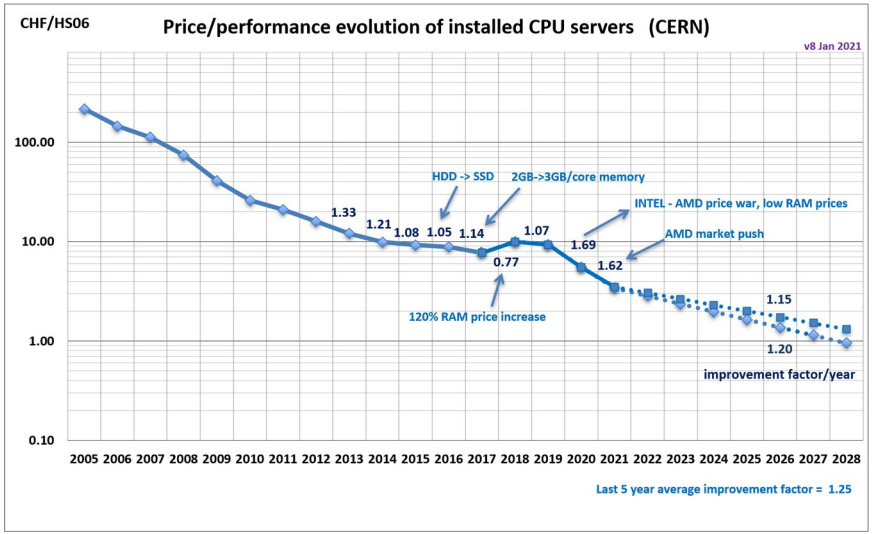
Outline

- Heterogeneous computing
- Trade-offs between multi-core and many-core architectures
- From general to specialized: Hardware accelerators and applications
- Type of workloads ideal for different accelerators
- Implications of heterogeneous hardware on the design and architecture of scientific software
- Embarrassingly parallel scientific applications in High Energy Physics
    - Processed on Graphics Processing Units (GPUs)

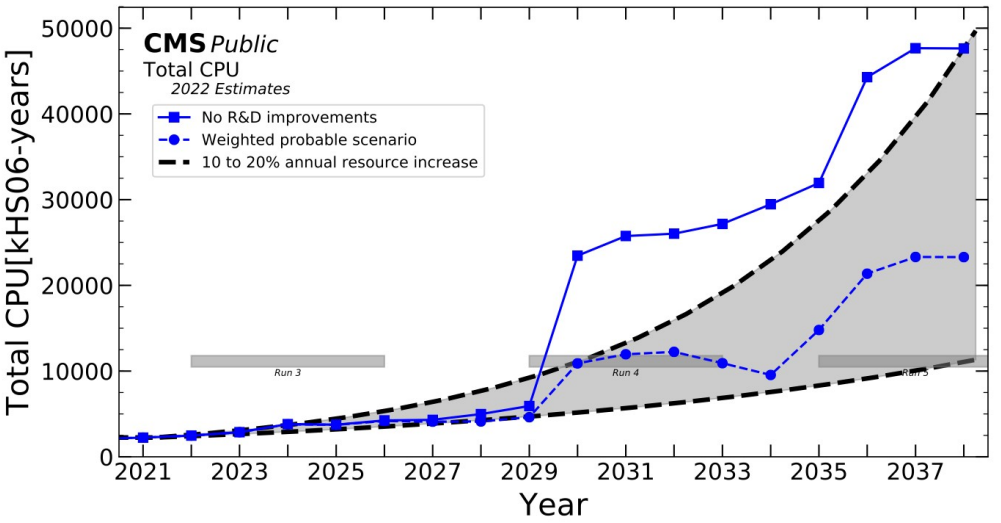




Computing performance challenge @ CERN



Courtesy Dr. Bernd Panzer-Steindel
(CERN/IT, CTO)



CMS Offline and Computing Public Results

- In high energy physics, usually assume flat budget for computing cost estimation
- Can no longer count on a stable increase of CPU processor performance / dollar
- Energy efficiency increasingly important
- Need to exploit heterogeneous systems in scientific applications – following High Performance Computing (HPC)

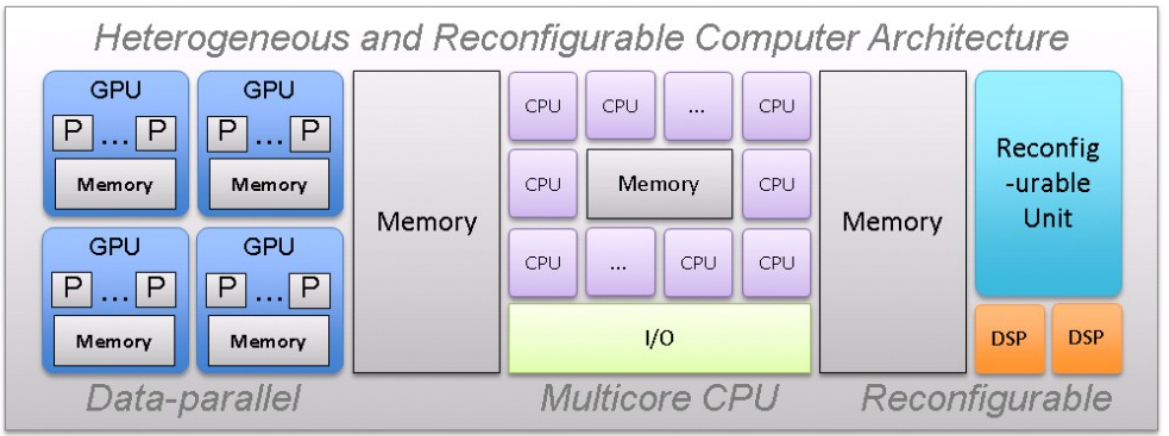
What does “heterogeneous” mean?

- System uses multiple types of computing cores or processors based on different computer architectures
  - Central Processing Units (CPUs)
  - Graphics Processing Units (GPUs)
  - Application-Specific Integrated Circuits (ASICs)
  - Field Programmable Gate Arrays (FPGAs)
  - Neural Processing Units (NPUs)
  - Tensor Processing Units (TPUs)
- Processors are designed for specific purposes or specialized processing

  → Assign workloads according to matching characteristics
- Optimize performance and energy efficiency
- “Accelerators” and “co-processors” both describe processors providing computing power in addition to a general-purpose processors (typcally a CPU)

Heterogeneous computing

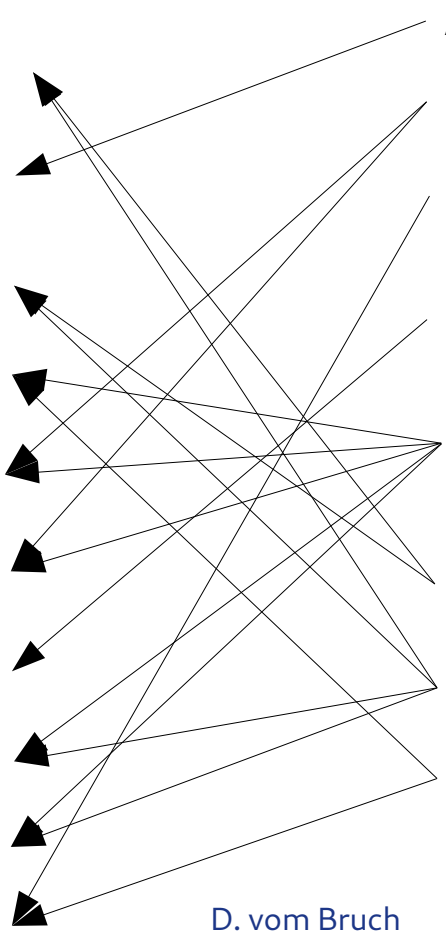
- Part of our everyday life: (de)compression, encryption, video stream decoding, 3D graphics acceleration, pattern / object recognition, automatic vehicles
- Accelerator technology often scaled to become a discrete device
    - Plug-and-play several components into a heterogeneous architecture



Source: https://www.iti.uni-stuttgart.de/en/chairs/ca/projects/oldprojects/simtech/

Trend towards heterogeneous solutions: TOP500

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,730,112	1,102.00	1,685.65	21,100
2	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442.01	537.21	29,899
3	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	2,220,288	309.10	428.70	6,016
4	**Leonardo** - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, Atos EuroHPC/CINECA Italy	1,463,616	174.70	255.75	5,610
5	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148.60	200.79	10,096
6	**Sierra** - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM / NVIDIA / Mellanox DOE/NNSA/LLNL United States	1,572,480	94.64	125.71	7,438
7	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, NRCPC National Supercomputing Center in Wuxi China	10,649,600	93.01	125.44	15,371
8	**Perlmutter** - HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10, HPE DOE/SC/LBNL/NERSC United States	761,856	70.87	93.75	2,589
9	**Selene** - NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100, Mellanox HDR Infiniband, Nvidia NVIDIA Corporation United States	555,520	63.46	79.22	2,646
10	**Tianhe-2A** - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000, NUDT National Super Computer Center in Guangzhou China	4,981,760	61.44	100.68	18,482



ARM processors

IBM Power9

Matrix-2000 NUDT many-core processors

Sunway: combination of Compute Processing Elements and Management Processing Elements

Nvidia GPUs

AMD GPUs

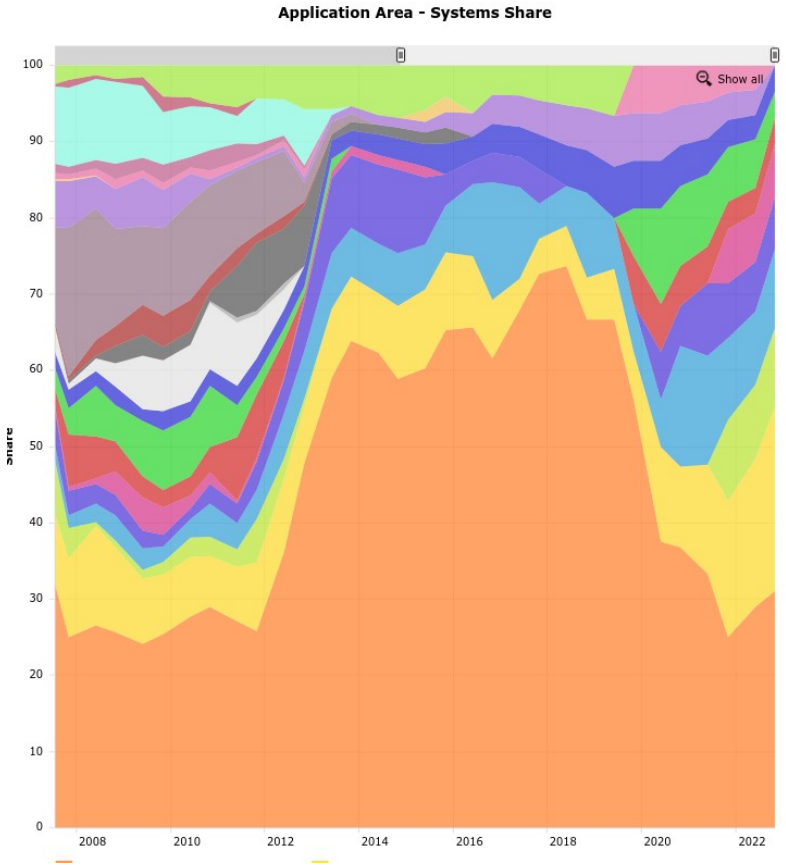
AMD CPUs

Intel CPUs

9/10 systems from the Top500 list use accelerators

https://www.top500.org/lists/top500/2022/11/

Application areas of TOP500 data centers

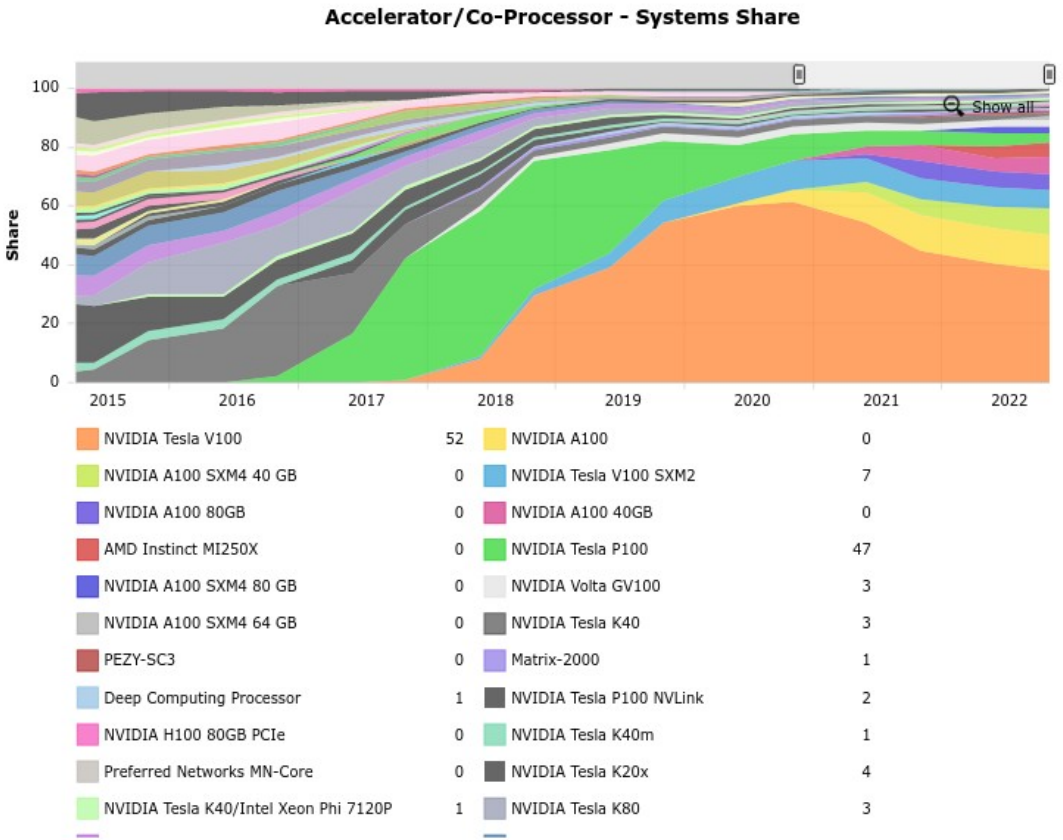


Research
IT Services
Weather and Climate Research
Services
Aerospace
Electronics
Information Processing Service
Semiconductor
Automotive
Geophysics
Defense
Others
Energy
Software
Information Service
Logistic Services
Web Services
Finance
Chemistry
Telecommunication
Database

- Share of data centers (not performance)
- Largest fraction is “Research”
- Last years increase in “Automotive”, “Information Service”, “Energy” and “IT Services”

https://www.top500.org/statistics/overtime/

Accelerators in TOP500 data centers



- Mainly Nvidia GPUs
- Some systems with AMD GPUs (increasing in last years)
- Some with processors dedicated to deep learning applications

https://www.top500.org/statistics/overtime/

Heterogeneous solutions & sustainability: Green500

Green500 Data

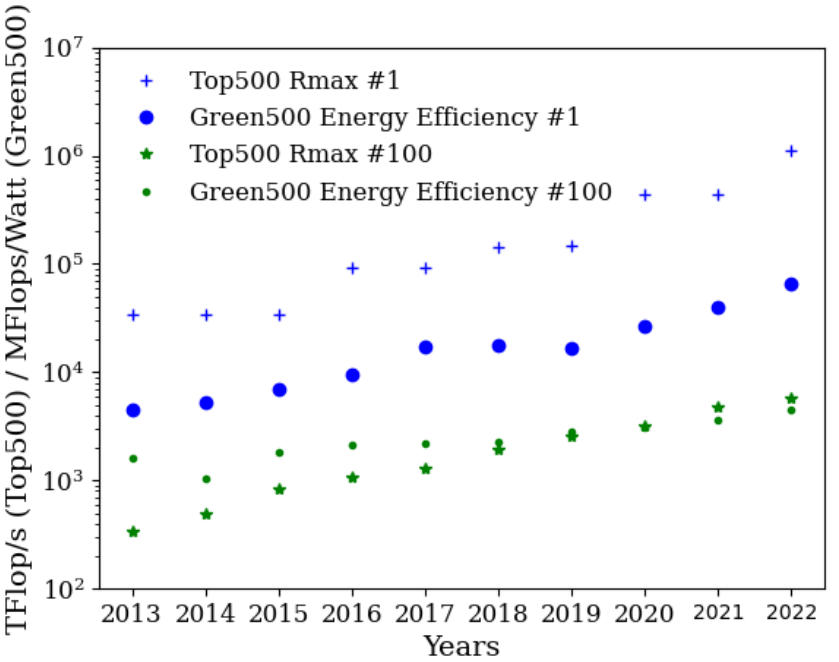
Rank	TOP500 Rank	System	Cores	Rmax (PFlop/s)	Power (kW)	Energy Efficiency (GFlops/watts)
1	405	**Henri** - Lenovo ThinkSystem SR670 V2, Intel Xeon Platinum 8362 2800Mhz (32C), NVIDIA H100 80GB PCIe, Infiniband HDR, Lenovo Flatiron Institute United States	5,920	2.04	31	65.091
2	32	**Frontier TDS** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	120,832	19.20	309	62.684
3	11	**Adastra** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Grand Equipement National de Calcul Intensif - Centre Informatique National de l'Enseignement Suprieur (GENCI-CINES) France	319,072	46.10	921	58.021
4	15	**Setonix – GPU** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Pawsey Supercomputing Centre, Kensington, Western Australia Australia	181,248	27.16	477	56.983
5	68	**Dardel GPU** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE KTH - Royal Institute of Technology Sweden	52,864	8.26	146	56.491
6	1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,730,112	1,102.00	21,100	52.227
7	3	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	2,220,288	309.10	6,016	51.382
8	159	**ATOS THX.A.B** - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, Atos Atos France	25,056	3.50	86	41.411
9	359	**MN-3** - MN-Core Server, Xeon Platinum 8260M 24C 2.4GHz, Preferred Networks MN-Core, MN-Core DirectConnect, Preferred Networks Preferred Networks Japan	1,664	2.18	53	40.901
10	331	**Champollion** - Apollo 6500, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 80 GB, Mellanox HDR Infiniband, HPE Hewlett Packard Enterprise France	19,840	2.32	60	38.555

- All top 10 systems from the Green500 list use accelerators
- 9/10 are accelerated with Nvidia or AMD GPUs
- MN-3 uses an accelerator optimized for matrix arithmetic, targeting deep learning applications

https://www.top500.org/lists/green500/2022/11/

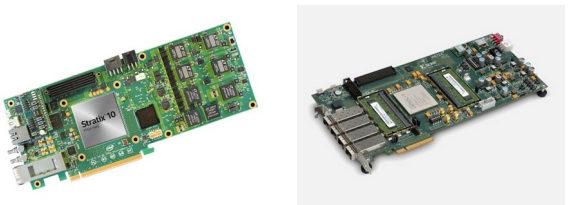
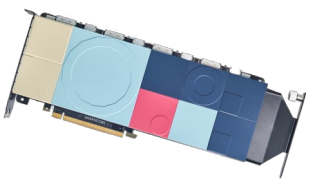
Energy efficiency

- Energy efficiency increasinly important
  - Electricity prices
  - Environmental impact
- Powering processors often costs more than buying them
- Definition of power consumption not uniform:
  - Only power delivered to machine
  - Power for machine, cooling and monitoring systems
  - Average versus peak power consumption
- Energy efficiency alone can hide increased absolute power demands → also consider absolute power
- Energy efficiency has increased less than processing power over last decade



Data taken from https://www.top500.org/lists/
Rmax: Maximal LINPACK performance achieved

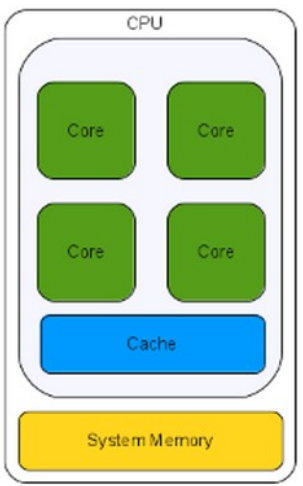
Types of hardware accelerators used in HEP

General purpose processors	**Graphics Processing Units (GPUs)** Vendors: AMD, Nvidia, Intel 	**Field Programmable Gate Arrays (FPGAs)** Vendors: Xilinx, Altera 
Dedicated accelerators	**Tensor Processing Units (TPUs)** Vendor: Google Specialized for machine learning 	**Intelligent Processing Units (IPUs)** Vendor: Graphcore Specialized for machine learning 

Multi-core versus many-core architecture

Multi-core

- O(10) cores
- Flexible: designed for both serial and parallel code
- Larger caches
- Emphasis on single thread performance



Many-core

- O(100-1000) cores
- Designed for parallel code
- Small caches
- Simpler cores

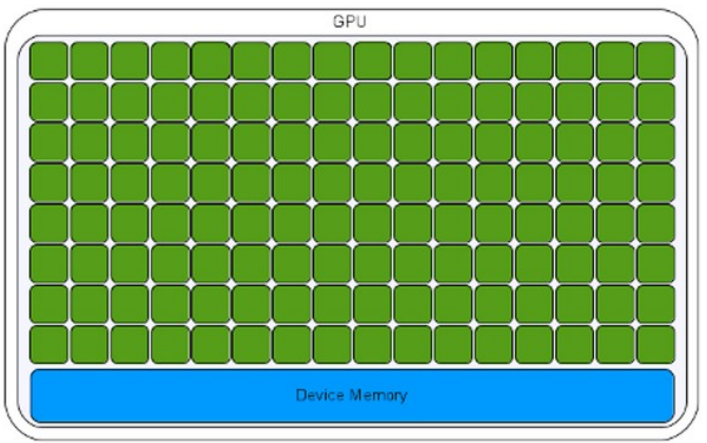


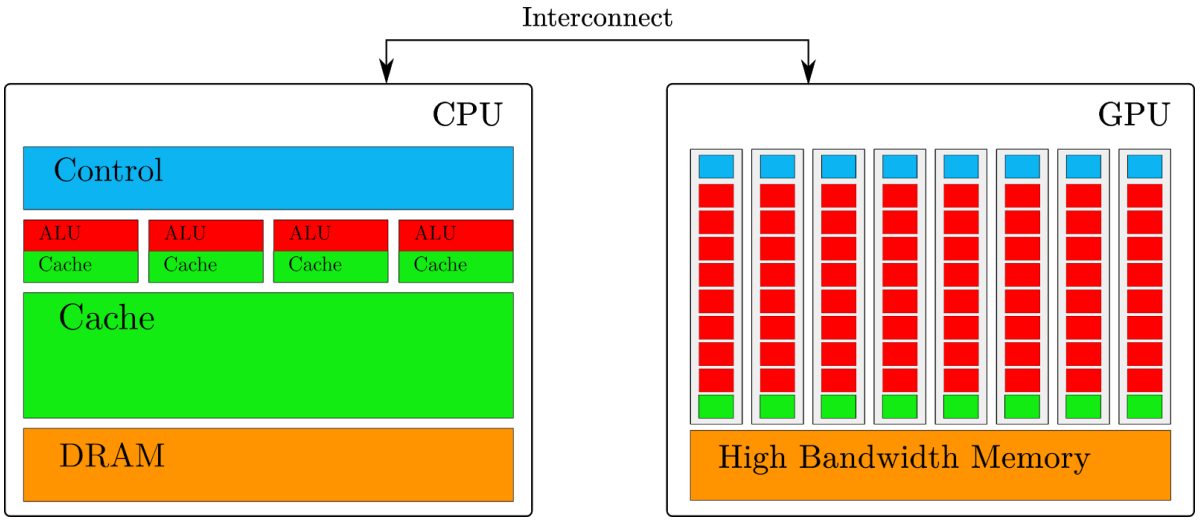
Image source

Types of workload for multi/many core architectures

- Typically, the main processor is multi-core and paired with a many-core accelerator
- Ensures that both serial and parallel code can be run efficiently
- Multi-core processors are often CPUs
  - Legacy code can run on them (albeit with low performance if not optimized for multi-threading)
  - They provide good serial performance
- Many-core processors are typically specialized accelerators
  - Individual algorithms / chains of algorithms are developed specifically for the accelerator
  - Only highly parallelizable problems are efficiently processed by them
  - The most widely used accelerators in science are many-core architectures

GPUs

- Developed for graphics pipeline
- General purpose computations possible
- Increasingly used for AI applications
- Hardware specialized in this direction since few years
- Programmed with high-level language



Low core count / powerful ALU

Complex control unit

Large chaches

→ **Latency optimized**

High core count

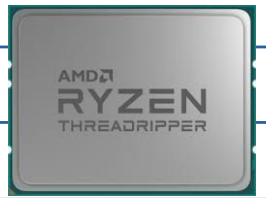
No complex control unit

Small chaches

→ **Throughput optimized**

GPU vs. CPU: Specifications

	AMD Ryzen Threadripper 3990X	**Nvidia A100**
Core count	64 cores / 128 threads	6912 cores
Frequency	2.9 GHz	1.41 GHz
Peak Compute Performance	3.7 TFLOPs	19.5 TFLOPs (single precision)
Memory bandwidth	Max. 95 GB/s	1.6 TB/s
Memory capacity	Max O(1) TB	40/80 GB
Technology	7 nm	7 nm
Die size	717 mm^2	826 mm^2
Transistor count	3.8 billion	54.2 billion
Model	Minimize latency	Hide latency through parallelism



Connectivity with GPU: PCIe connection

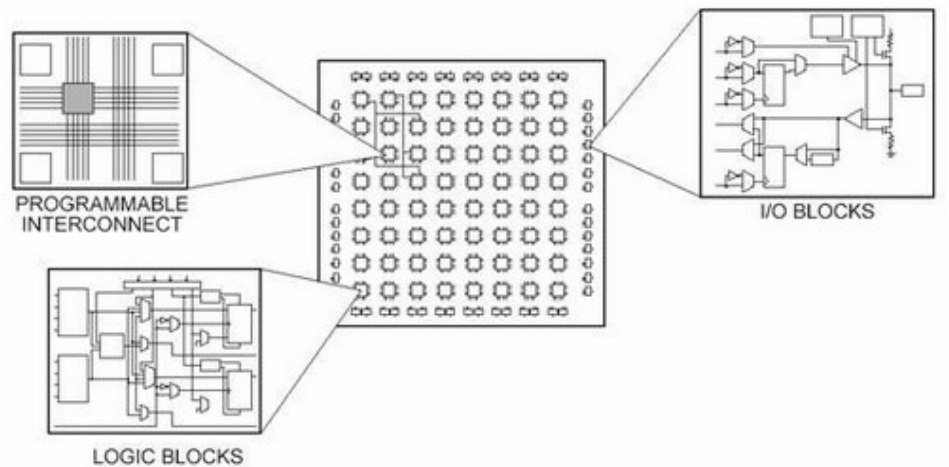
PCIe generation	1 lane	16 lanes	Year of announcement
2.0	500 MB/s	8 GB/s	2007
3.0	985 MB/s	15.75 GB/s	2010
4.0	1.97 GB/s	31.5 GB/s	2011
5.0	3.94 GB/s	63 GB/s	2017
6.0	7.56 GB/s	121 GB/s	2019
7.0	15.13 GB/s	242 GB/s	2022

https://en.wikipedia.org/wiki/PCI_Express

FPGAs

- Thousands of logic blocks
- Input/Output blocks
- Connected via programmable interconnect

- Configure a circuit to do the task it is programmed for
  → Hardware implementation of an algorithm
- Fixed latency
- Very good at integer computations
- Does not require a computer to run (has its own I/O)
- Traditionally, programmed with hardware description languages (Verilog, VHDL) → long development time
- Increasingly more high-level languages developed



Source: National Instruments

xkcd

GPU vs. FPGA

GPUs

- Higher latency
- Connection via PCIe (or NVLink)
- Bandwidth limited by PCIe

- Very good floating point operation performance
- Lower engineering cost
- Backward / forward compatibility

FPGAs

- Low & deterministic latency
- Connectivity to any data source
- High bandwidth

- Intermediate floating point performance
- High engineering cost
- Not so easy backward compatibility

CPU – GPU - FPGA

	Latency	Connection	Engineering cost	FP performance	Serial / parallel	Memory	Backward compatibility
CPU	O(10) µs	Ethernet, USB, PCIe	Low entry level: Programmable with C++, pthon, etc.	O(1-10) TFLOPs	Optimized for serial, increasingly vector processing	O(100) GB RAM	Compatible, except for vector instruction sets
GPU	O(100) µs	PCIe, Nvlink	Low to medium entry level: Programmable with CUDA, OpenCL, etc.	O(10) TFLOPs	Optimized for parallel performance	O(10) GB	Compatible, exept for specific features
FPGA	Fixed O(100) ns	Any connection via PCB	High entry level: traditionally hardware description languages, Some high-level syntax available	Optimized for fixed point performance	Optimized for parallel performance	O(10) MB on the FPGA itself	Not easily backward compatible

https://arxiv.org/pdf/2003.11491.pdf

Types of workloads for different accelerators

GPUs:

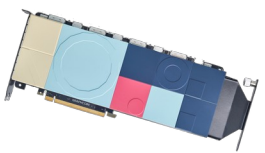
- Relaxed latency requirements
- High FLOPs need
- I/O via PCIe no bottleneck
- Highly parallelizable problem
- Fits within GPU memory

FPGAs:

- Strict latency requirements
- High I/O needs
- Highly parallelizable problem
- Fits within FPGA resources (logic elements and memory blocks)

TPUs / IPUs etc.:

- Machine learning training or inference
- TPUs: Use as a service in the cloud
- IPUs: MIMD compatible problem
- Fit within memory



Challenges in heterogeneous computing

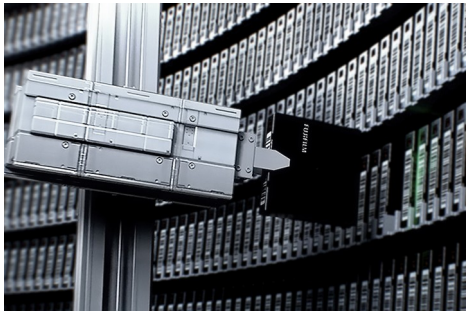
	Challenge	**Approach**
Different architectures	Different instruction sets can produce results that are not bit-wise reproducible	Check requirements of problem at hand: What is the minimum required resolution?
Data transmission between devices	• Interconnect can cause bandwidth bottleneck • Data layout: one might not be suitable for all device architectures and memory structures	• Minimize copies between devices • Minimize transformations between data layouts
Programming environments	• Different compilers • Different APIs	• Use programming environments designed for heterogeneous computing → lecture by D. Campora

Computing needs in HEP

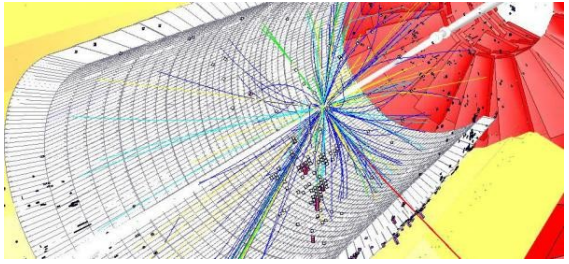
Detector

Trigger / Real-time analysis

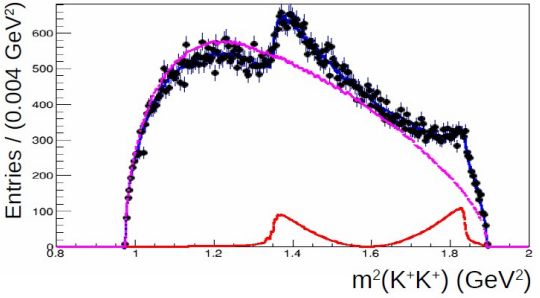
Storage



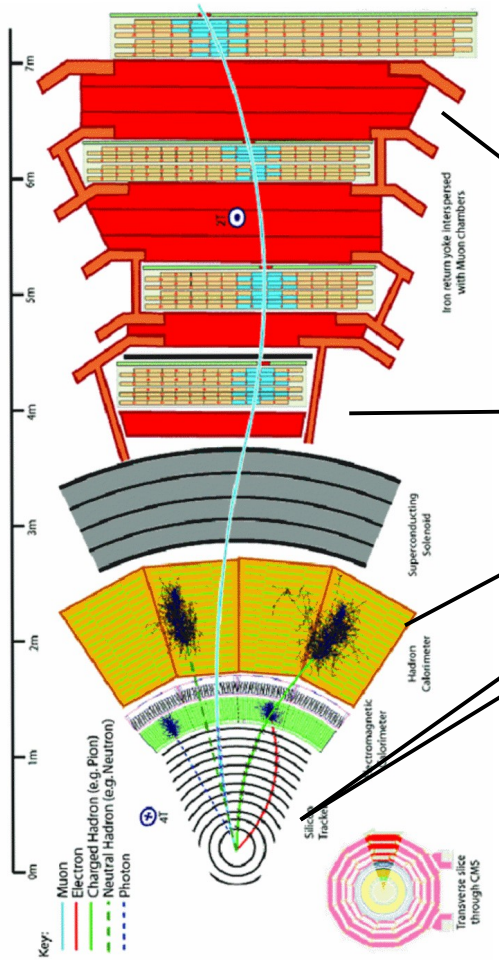
Simulation

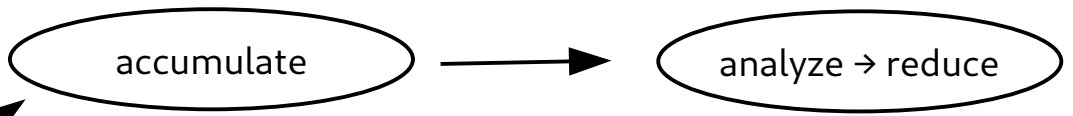


Data analysis

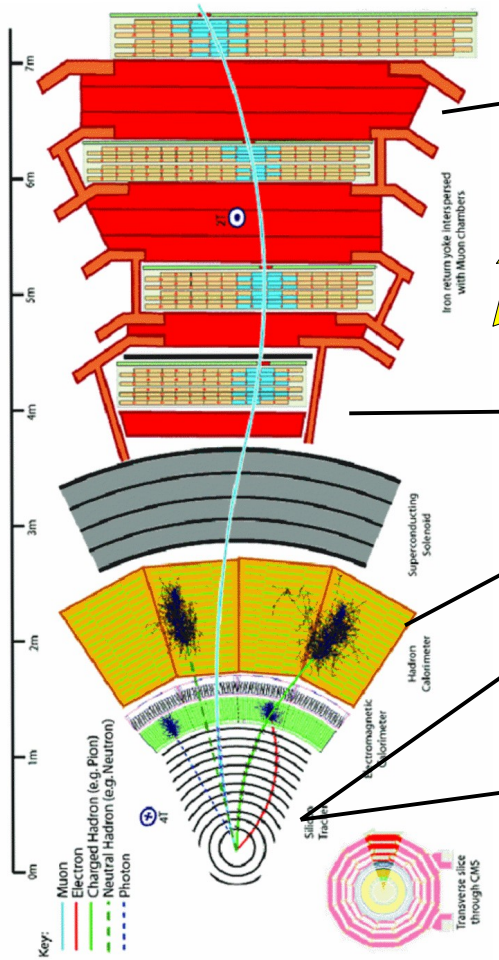


“Trigger”: Real-time data analysis and reduction





“Trigger”: Real-time data analysis and reduction



analyze → reduce

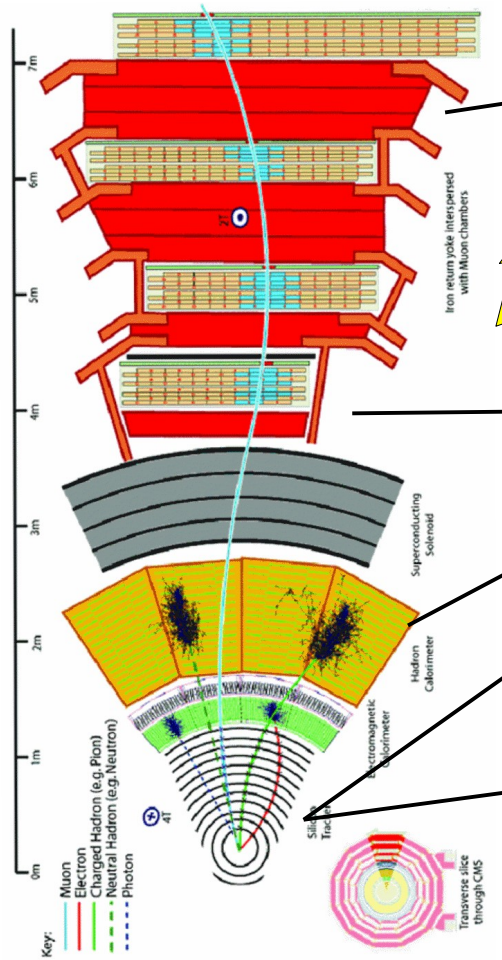
Large data rate

accumulate

analyze → reduce

analyze → reduce

“Trigger”: Real-time data analysis and reduction



analyze → reduce

Large data rate

accumulate

analyze → reduce

analyze → reduce

First: Hardware trigger

- Data obtained directly from detector
- Decision taken in fixed time, low latency
- Based on local information from a subdetector
- Chip constraints → not too complex calculations

Second: Software trigger

- Data already transferred to a server
- Decision taken with medium latency
- Based on information from several subdetectors
- Processor constraints less stringent

Match trigger to hardware

First: Hardware trigger

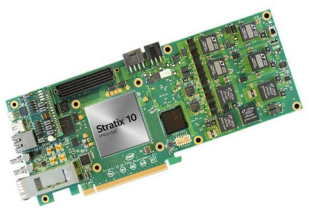
- Data obtained directly from detector
- Decision taken in fixed time, low latency
- Based on local information from a subdetector
- Chip constraints → not too complex calculations

Second: Software trigger

- Data already transferred to a server
- Decision taken with medium latency
- Based on information from several subdetectors
- Processor constraints less stringent

Field Programmable Gate Arrays (FPGAs)

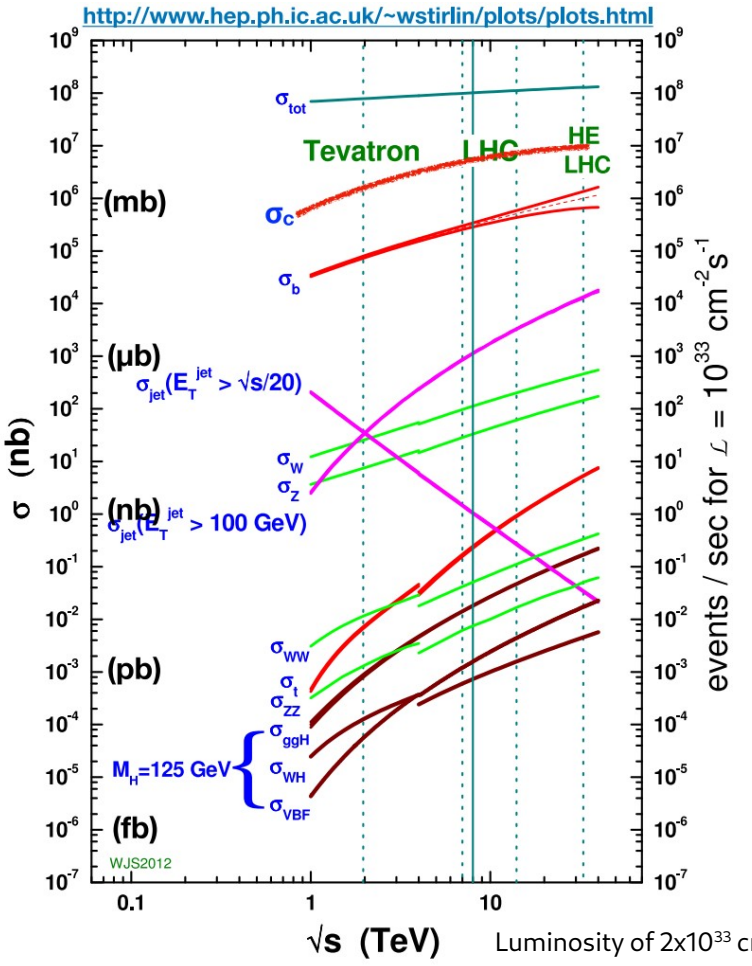
- Low & deterministic latency
- Connectivity to any data source → high bandwidth
- Intermediate floating point performance



CPUs and GPUs

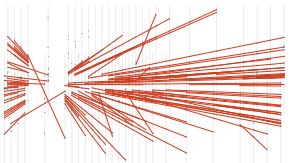
- Higher latency
- Very good floating point performance
- Connected to server (via PCIe connection for GPU)

Efficient signal selection



Luminosity of 2×10^{33} cm^{-2}s^{-1}

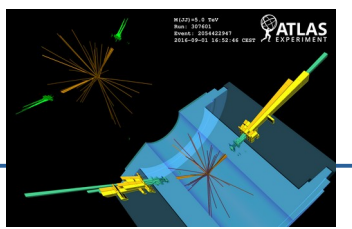
LHCb: Mainly beauty and charm physics



- Signal rates at MHz level
- Signal characteristics: Displaced vertices, momentum, particle type
- → No optimal local criteria for selection

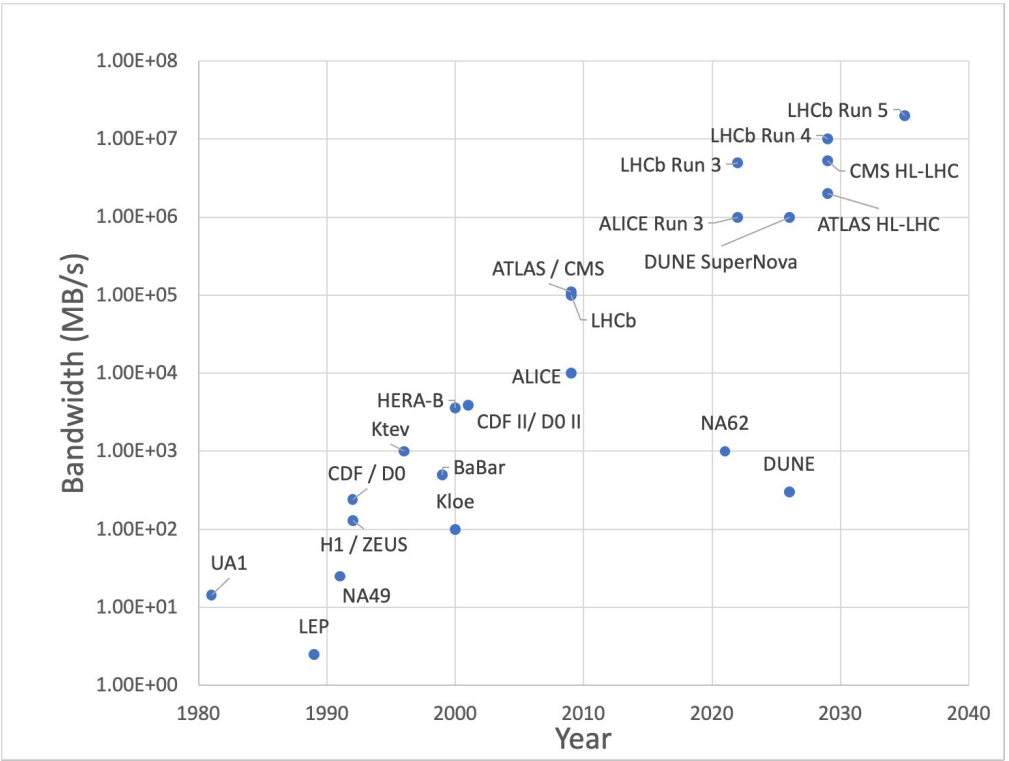
ATLAS & CMS: Mainly Higgs properties, high p_T new phenomena

- Signal rates up to hundreds of kHz
- Signal characteristics: high pT / transverse energy
- → Local criteria for selection possible



Challenge I: Real-time analysis (RTA)

LHC long-term schedule



A. Cerri – University of Sussex

Overview of GPU usage in various HEP experiments

Experiment	Main tasks processed on GPU	Event / data rate	Number of GPUs	Deployment date
CMS	Decoding, clustering, pattern recognition in pixel detector	100 kHz	O(400)	2022
ALICE	Track reconstruction in three sub-detectors	50 kHz Pb-Pb or < 5 MHz p-p / 30 Tbit/s	O(2000)	2022
LHCb	Decoding, clustering, track reconstruction in three sub-detectors, vertex reconstruction, muon ID, selections	30 MHz/ 40 Tbit/s	O(250)	2022

https://arxiv.org/pdf/2003.11491.pdf

Overview of GPU usage in various HEP experiments

Experiment	Main tasks processed on GPU	Event / data rate	Number of GPUs	Deployment date
CMS	Decoding,	100 kHz	O(400)	2022
	vertex reconstruction, muon ID, selections			

All experiment needs and environments are quite different

→ heterogeneous solutions are different

Common points

- Reconstruction algorithms are main candidates for parallelization and off-loading to accelerators
- Scheduling of memory copies, calculations on accelerator, calculations on host server is crucial
- Flexible software frameworks are necessary

https://arxiv.org/pdf/2003.11491.pdf

Recurrent tasks in real-time data analysis

Raw data decoding

- Transform binary payload from subdetector raw banks into collections of hits (x,y,z) in LHCb coordinate system

Track reconstruction

- Consists of two steps:
  - Pattern recognition: Which hits were produced by the same particle? → "Track"

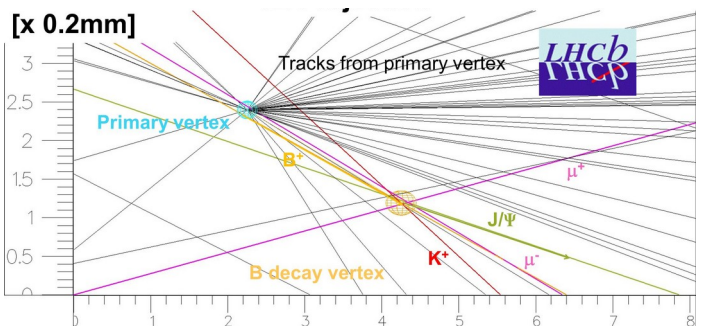
    → Huge combinatorics when testing different combinations of hits
  - Track fitting: Describe track with mathematical model

Vertex finding

- Where did proton-proton collisions take place?
- Where did particles decay within the detector volume?

Calorimeter / muon detector reconstruction

- Reconstruct clusters in the calorimeter / muon detectors
- Match tracks to clusters



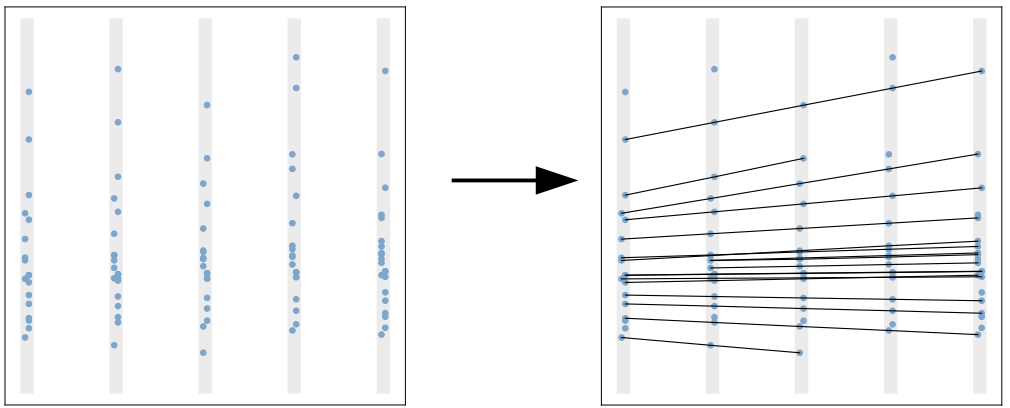
Computational challenge: Track reconstruction

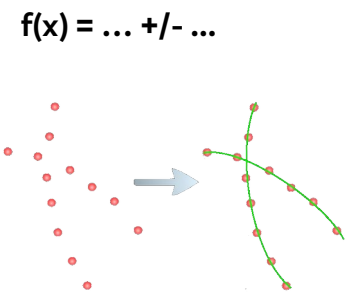
Pattern recognition

- Which measurements originate from the same particle?
  → "Track"
- Huge combinatorics when testing different combinations of measurements

Track fit

- Describe track with mathematical model
- Calculate where it came from and how it continues



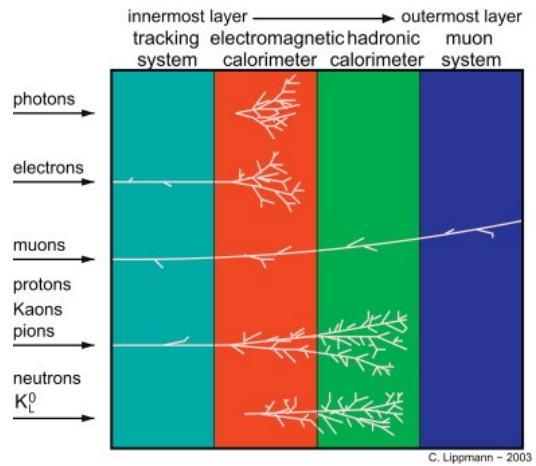


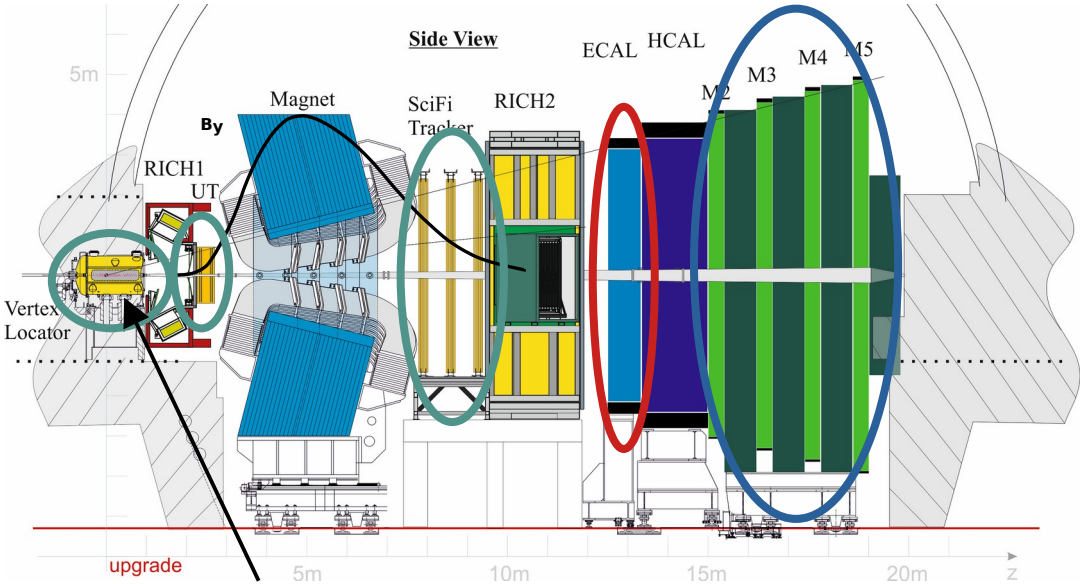
Huge computing challenge for 10^9 – 10^{10} tracks / second

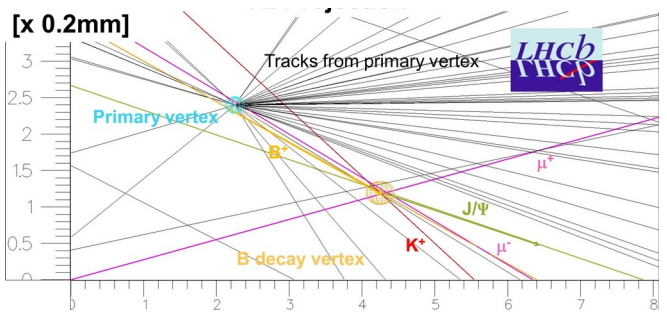
LHCb's first level real-time analysis

High Level Trigger 1 (HLT1) tasks

- Decode binary payload of sub-detectors
- Reconstruct charged particle trajectories
- Identify electron and muon particles
- Reconstruct particle decay vertices
- Select proton-proton bunch collisions to store







LHCb: How does HLT1 map to GPUs?

Characteristics of LHCb HLT1	**Characteristics of GPUs**
Intrinsically parallel problem: - Run events in parallel - Reconstruct tracks in parallel	Good for - Data-intensive parallelizable applications - High throughput applications
Huge compute load	Many TFLOPS
Full data stream from all detectors is read out → no stringent latency requirements	Higher latency than CPUs, not as predictable as FPGAs
Small raw event data (~100 kB)	Connection via PCIe → limited I/O bandwidth
Small event raw data (~100 kB)	Thousands of events fit into O(10) GB of memory

LHCb: The Allen project

- Named after Frances E. Allen

- Fully standalone software project: https://gitlab.cern.ch/lhcb/Allen, documentation
- Framework developed for processing LHCb´s first real-time selection stage (HLT1) on GPUs

- Cross-architecture compatibility via macros & few coding guide lines
  - GPU code written in CUDA, runs on CPUs, Nvidia GPUs (CUDA), AMD GPUs (HIP)
- Algorithm sequences defined in python and generated at run-time
- Multi-event processing with dedicated scheduler
- Memory manager allocates large chunk of GPU memory at start-up
- Reconstruction algorithms re-designed for parallelism and low memory usage: O(MB) per core

- Publications: Comput Softw Big Sci 4, 7 (2020), Technical Design Report (2020), Comput Softw Big Sci 6, 1(2022), EPJ Web of Conferences 251, 04009 (2021)

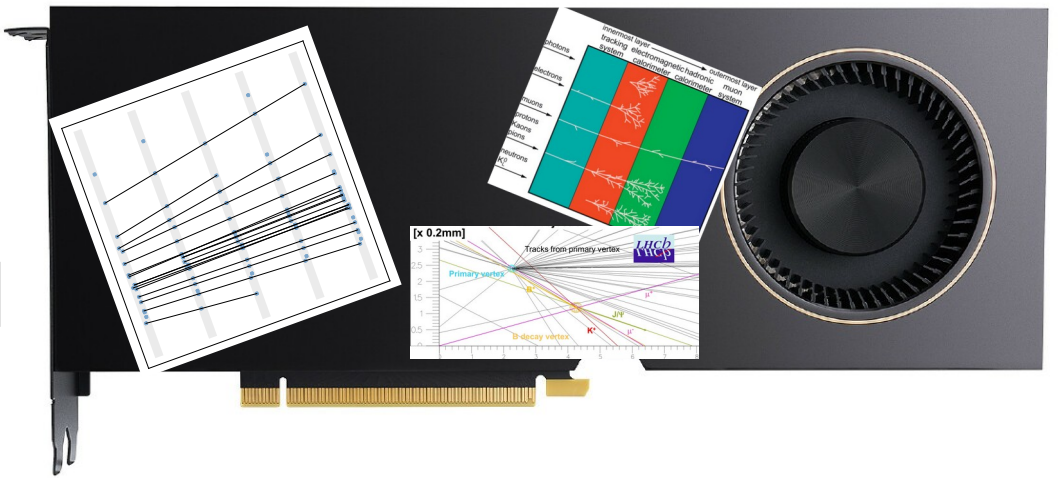
LHCb: Minimize copies to / from GPU

Server

Raw data

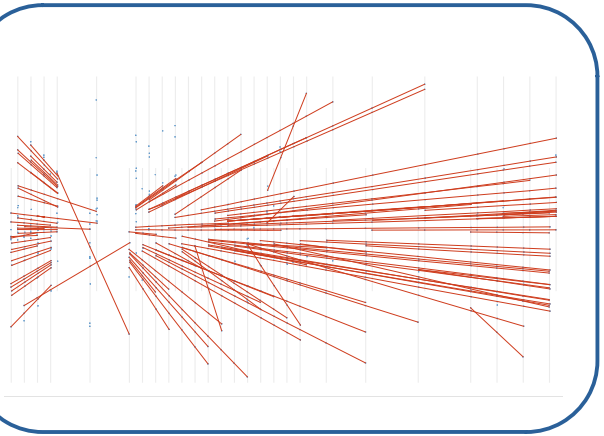
Selection decisions

GPU

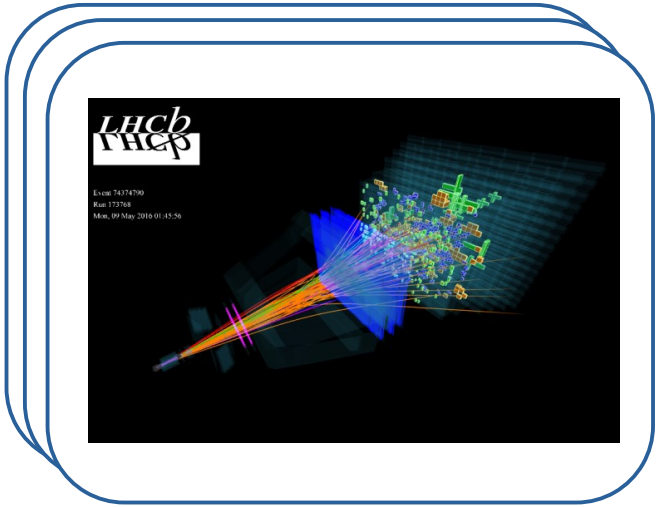


LHCb: Three levels of parallelization

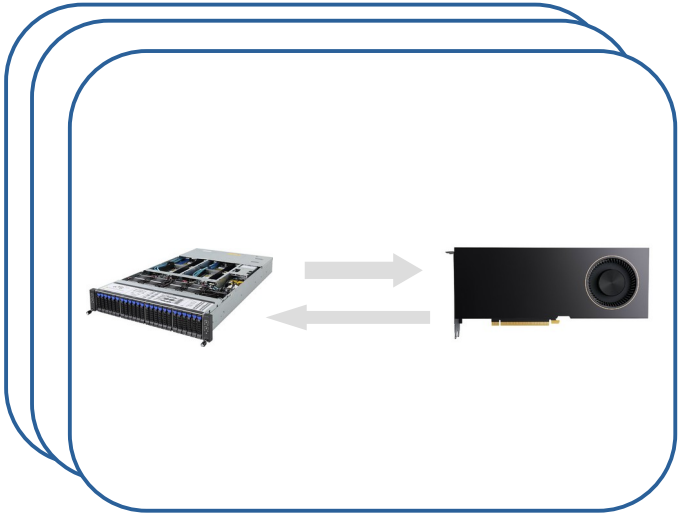
Intra-collision: Tracks, vertices, …



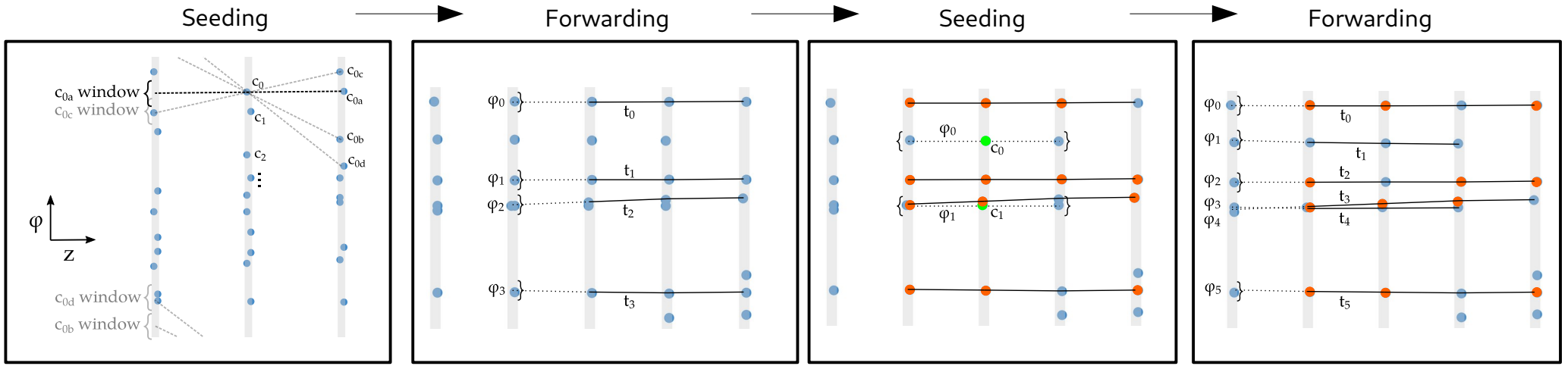
Proton collisions



Collision batches



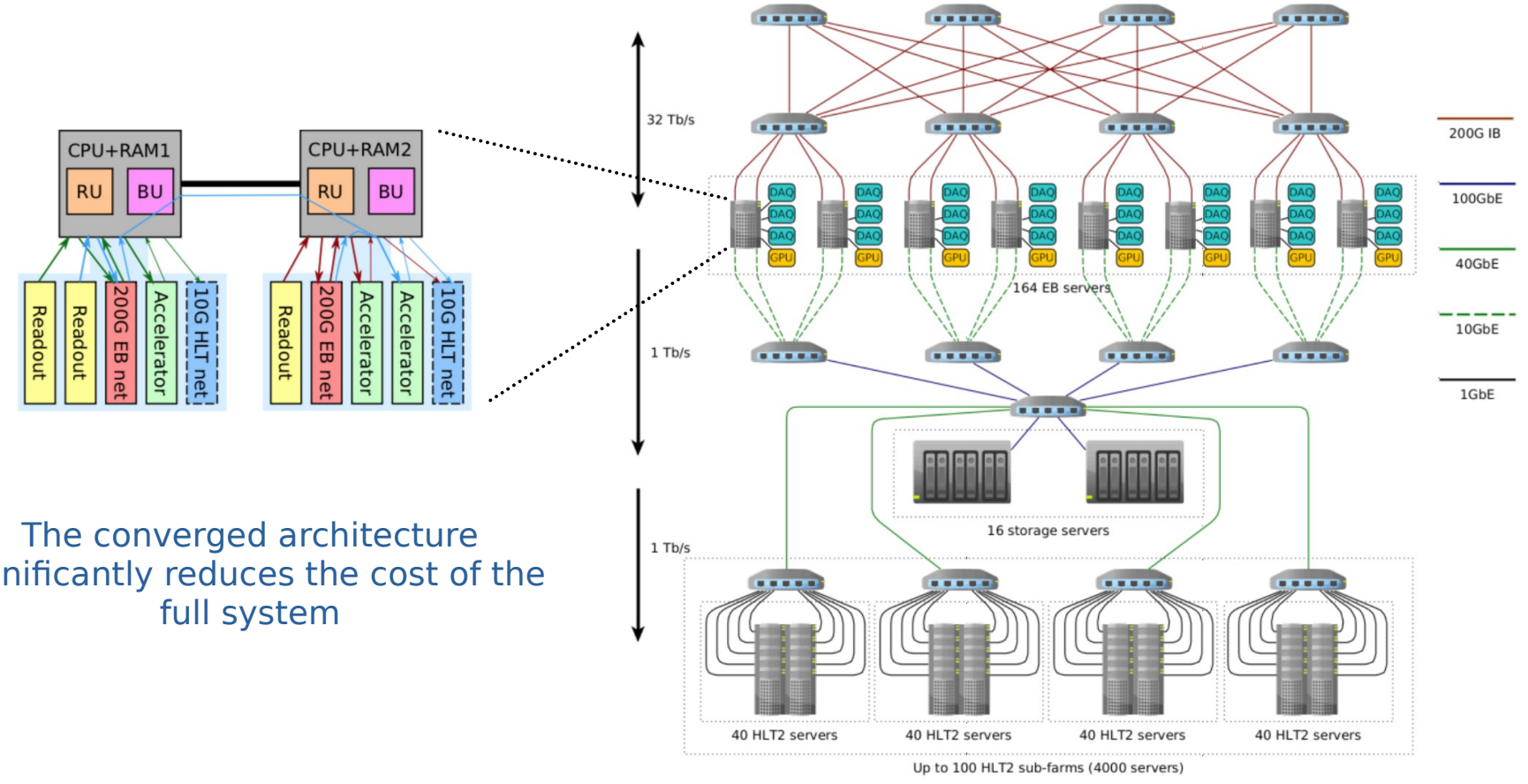
LHCb: Example algorithm: “Triplet” finder



D. Campora et al, “Search by triplet: An efficient local track reconstruction algorithm on parallel architectures”, Journal of Computational Science 54, 101422 (2021)

- Build “triplets” of three hits on consecutive layers → parallelization
- Choose them based on alignment in phi
- Hits sorted by phi → memory accesses as contiguous as possible: data locality
- Extend triplets to next layer → parallelization

LHCb: GPU HLT1 within data acquisition system



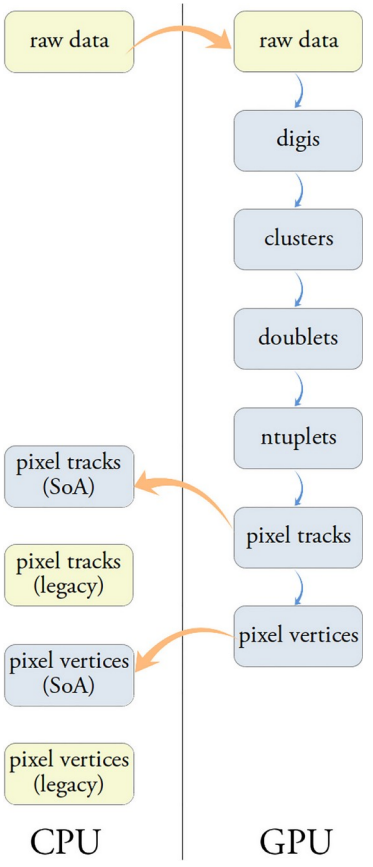
The converged architecture significantly reduces the cost of the full system

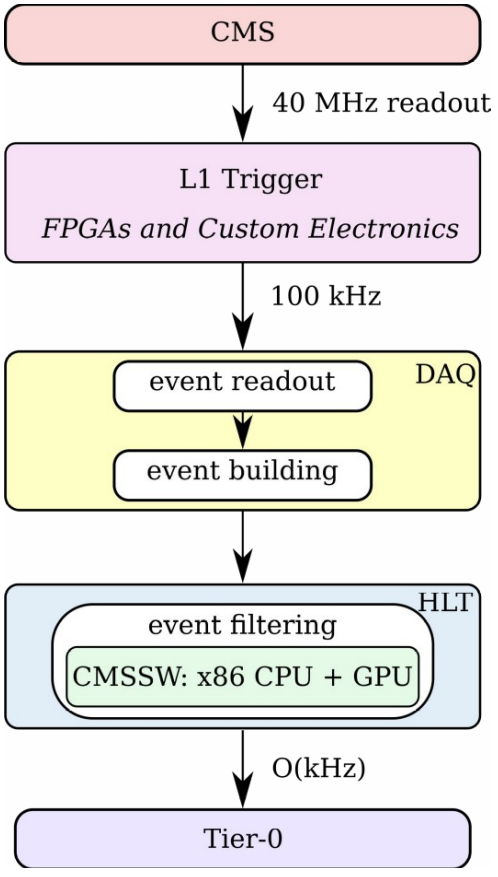
CMS reconstruction on GPUs

- Several algorithms ported to GPUs for Run 3:
  - Track reconstruction in pixel detector
  - Primary vertex reconstruction from those tracks
  - Calorimeter local reconstruction of ECal and HCal
- Crucial to allow close interlinking of CPU and GPU software

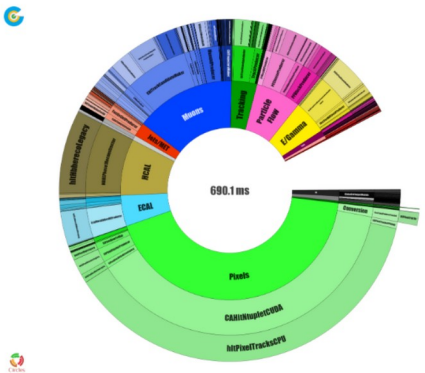
  → integrated into CMSSW (arXiv2004.04334)
- Work ongoing for other reconstruction algorithms

Front. Big Data 3 (2020) 601728

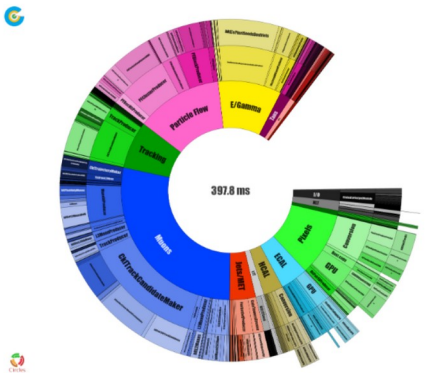




CMS HLT performance with GPUs



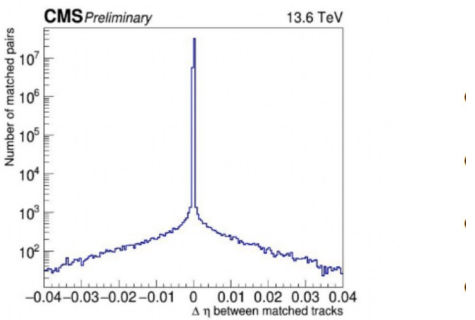
Average time per event on CPU



Average time per event on GPU

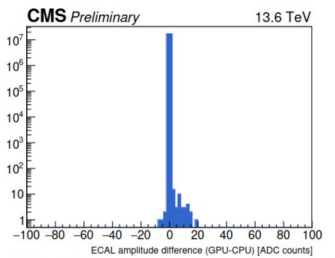
- GPU offload increases HLT throughput by factor 1.7
- 400 Nvidia Tesla T4 cards in HLT farm

- Event-by-event comparison between CPU and GPU results
- Double precision on CPU, single precision on GPU



Difference in η of a track reconstructed on CPU with the track reconstructed on GPU, matched within a geometrical acceptance of ΔR < 0.2

ECAL



ECAL barrel: difference of amplitude of same pulse when the fit is run on GPU and on CPU

ACAT poster by M. Huwiler

Common characteristics of software frameworks

- Same code base compiled for various computing architectures: GPUs, x86,…
- Memory management system for GPU memory: avoid dynamic memory allocation
- Schedule pipelines of GPU (and CPU) algorithms → hide memory copies
- Integration into experiments' main software frameworks

Allen framework at LHCb

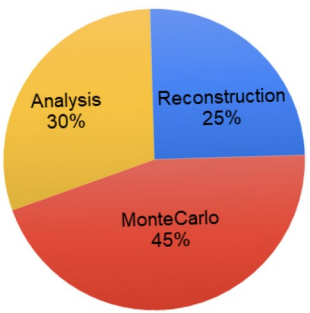
Patatrack at CMS

O2 at ALICE

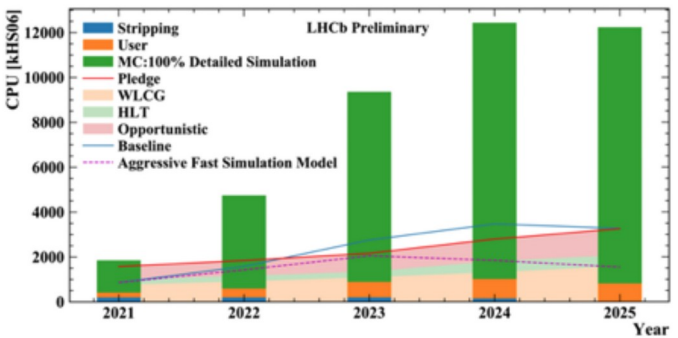
Challenge II: Simulation

- Running experiments at higher luminosity leads to large increase in simulation demands
- Projected between 45 and 90 % of CPU usage for simulation
- Large effort ongoing to process simulation on GPUs
- Partially driven by hardware available in HPC centers

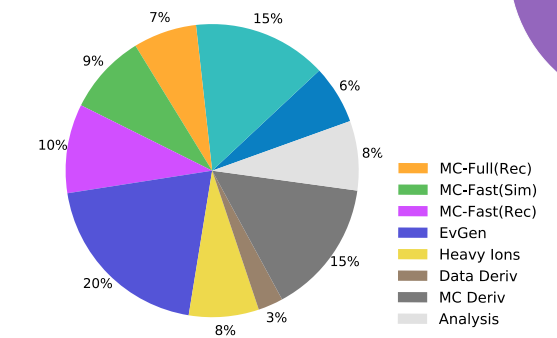
ALICE

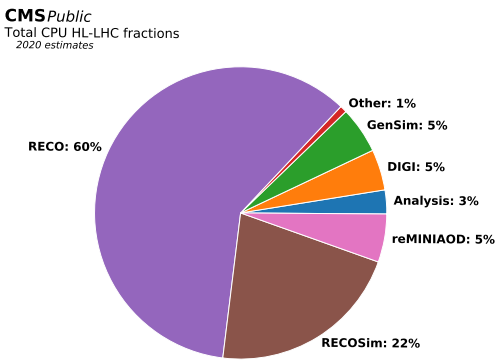


Projected needs for Run 3



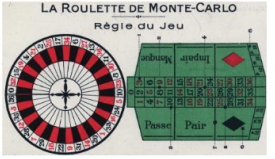
See H. Gray's opening vCHEP talk





Simulation: Where to use accelerators?

Lockstep? MC generators (*lucky!*) vs MC detector simulation (unlucky)



- *Monte Carlo methods are based on drawing (pseudo-)random numbers*: a dice throw
- From a software workflow point of view, these are used in *two rather different cases*:

Data parallelism (NB: MULTI-EVENT API !)

MC SAMPLING

INPUT

ME event generators* (*before* ME calculation):
- MC integration (cross sections)
- MC generation (event samples)

SAME CALCULATION ON DIFFERENT DATA!

OUTPUT

Lockstep processing Good for SIMT/SIMD

MC DECISIONS* (*after* ME calculation):
- MC unweighting (keep/reject)
- Parton showers (PS)
- Fragmentation and decays

INPUT

DECISION

OUTPUT

Stochastic branching Bad for SIMT/SIMD

Detector simulation (Geant4)
- Particle/matter interaction (when? how?)
- Particle decays (when?)

DIFFERENT CALCULATIONS ON DIFFERENT DATA!

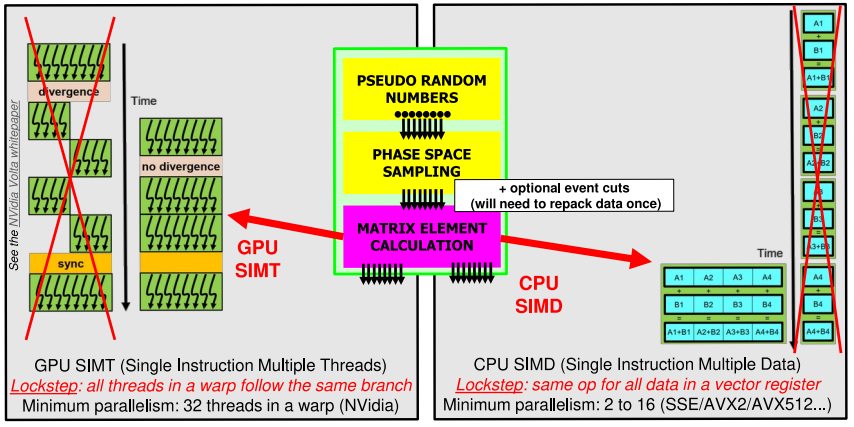
**NB: the CPU-intensive ME calculation comes before PS, fragmentation, detector simulation*

Event generators on GPUs

- Madgraph4gpu project: started in 2020 within HSF Generator WG
- Port MC event generators, in particular matrix element calculation (current bottleneck), to GPUs
- Make use of CUDA's random number generator: cuRAND

Main design idea: event-level data parallelism (lockstep)

- In MC generators, all events *in one channel* initially go through the same calculations
  - *Computing MEs involves the calculation of the exact same function on different data points*
  - This is what *makes event generators a good fit for GPUs (SIMT) and vector CPUs (SIMD)*

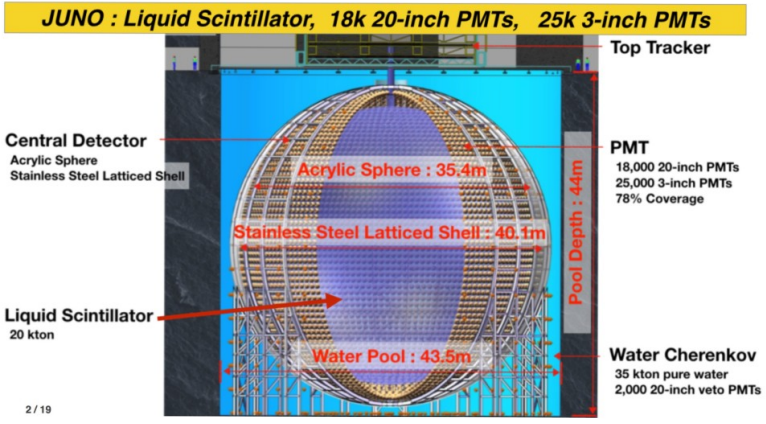


~~Executive summary for the impatient~~ Conclusions!

- The Matrix Element calculation in any ME generator can be efficiently parallelized using SIMD or GPUs
- Our reengineering of MG5aMC is close to a first fully functional alpha release for LO QCD processes
  - *The new ME calculation is integrated in MadEvent* – we get the same cross section *and LHE files* as in Fortran!
- On CPUs, in vectorized C++ we *reach the maximum x8/x16 (double/float) SIMD speedup for MEs alone*
  - The speedups achieved for the overall workflow are slightly lower due to *Amdahl's law*, but not much
  - Example: our *current overall speedup is x6/x10 (double/float)* for $gg \to t\bar{t}gg$ (on one CPU core)
- On GPUs, using CUDA we *achieve O(100-1000) speedups for MEs alone over one no-SIMD CPU core*
  - The speedups may be much lower due to *Amdahl's law*, but we are improving on that
  - Example: our *current overall speedup is x60/x100 (double/float)* for $gg \to t\bar{t}ggg$ on an NVidia V100
- Floats are x2 faster than doubles in SIMD and NVidia GPUs – we also added 'mixed' precision modes
- In SYCL we get ~similar performances to CUDA on NVidia and we may run also on AMD or Intel GPUs
- Future challenges include optimizing heterogenous processing on one GPU and multiple CPU cores

Photon simulation with Nvidia OptiX

- Photon simulation is similar to ray tracing problem
  → ideally suited workload for GPU
- Opticks framework developed for photon simulation, e.g. in a LAr TPC
- Uses Nvidia's OptiX ray tracing engine and integrated with Geant4



See S. Blyth's vCHEP talk

Ray tracing

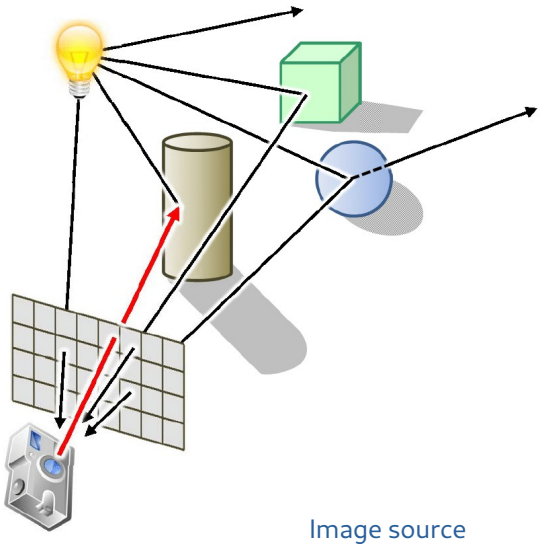
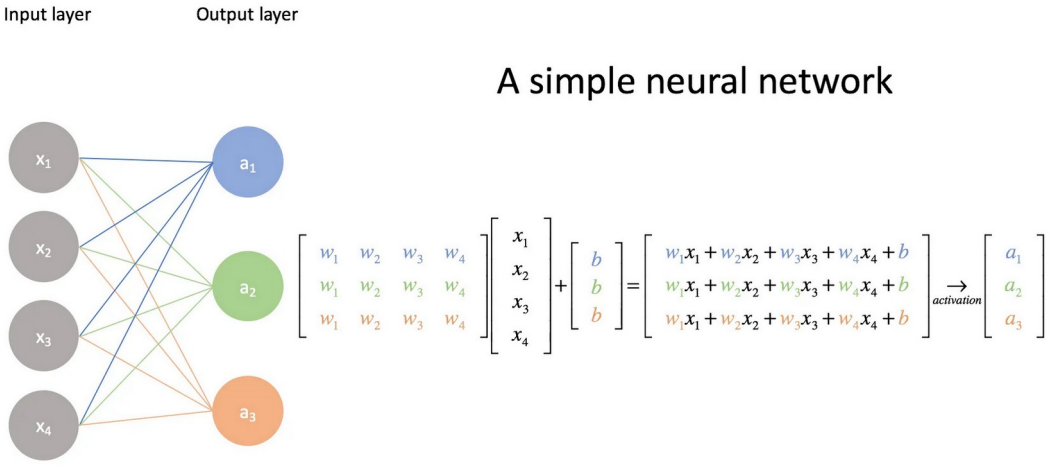


Image source

Also IceCube are working on using ray tracing for their photon simulation, see this vCHEP talk

Machine learning: Training



- Large amount of data to handle: high memory bandwidth on GPUs
- Neural networks are embarassingly parallel problems: matrix multiplication
- Many networks can be trained with reduced precision
- Applications in HEP: Pattern recognition, categorization, fast simulation, …
- Libraries used: Tensorflow, Keras, PyTorch, …
- HSF tutorial on machine learning with GPUs

Summary

- We are facing a huge computing challenge in HEP, mainly in real-time reconstruction and simulation
- Cannot be solved solely by using CPU processors
- Trend in HPC is towards heterogeneous architectures
- Heterogeneous architectures are crucial for energy efficient systems
- Make use of many-core accelerators for embarrassingly parallel problems within HEP
- Most popular accelerator: GPUs
- Various experiments have developed and commissioned heterogeneous real-time analysis systems with GPUs
- Extensive R&D also ongoing to use them for simulation
- Frameworks for heterogeneous software are being developed

- Note: Compute Accelerator Forum organized by HEP Software Foundation, Openlab, SIDIS
  Presentations roughly once per month on accelerator topics

Backup

Types of GPUs

	Scientific GPUs	**Gaming GPUs**
Precision	~3 times more single precision TFLOPS than double precision → suited for double precision	~40 times more single precision TFLOPS than double precision → not well suited for double precision
Error correction	Available	Not available
Connection	NVLink & PCIe	Only PCIe

R&D to use Graphcore's IPUs

Simulation

- Study usage of IPUs for event generation with fast simulation technique
- Particularly suited for machine learning techniques
- Tested event generation with generative networks (GAN)

Track reconstruction

- Also implemented Kalman filter for track fitting on the IPU
- Multiple Instruction Multiple Data (MIMD) architecture
- → Higher performance observed for conditional control-flow programs
- No direct comparison to GPU implementation

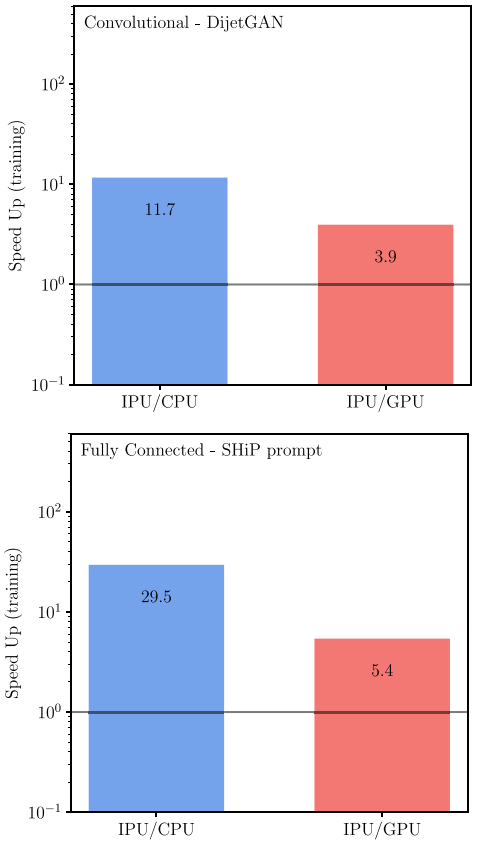
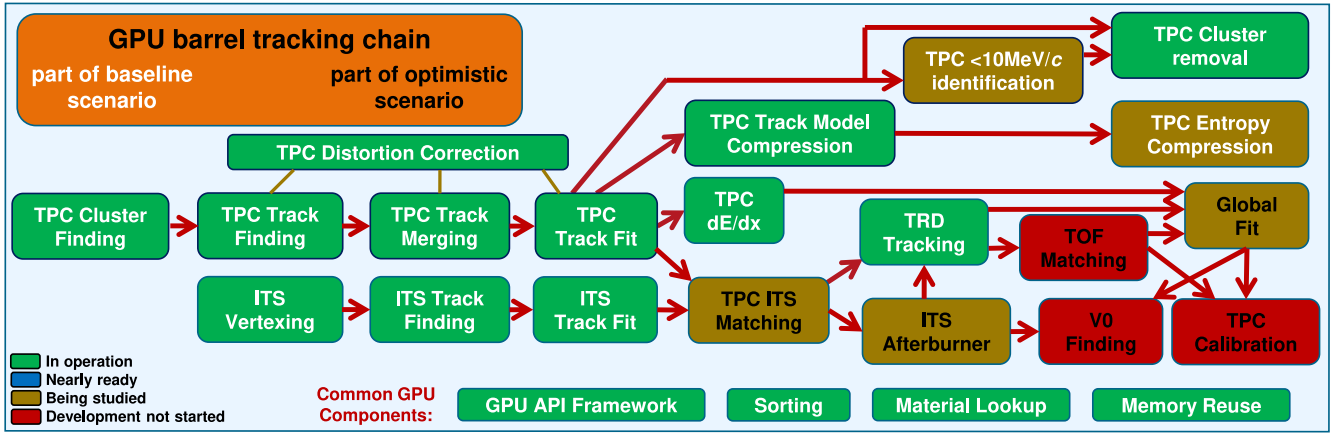


Fig. 3 Comparison of the time to train the IPU relative to the CPU or GPU of Table 1

Computing and Software for Big Science 5, 8 (2021)

ALICE: Reconstruction on GPUs

- Process 10 ms timeframes, O(10 GB) size
- One detector dominates computing needs: Time Projection Chamber (TPC)
- TPC reconstructed in real time on GPUs for compression and calibration since Run 1
- Also adding reconstruction of other detectors to the GPU workflow
- Aiming to process full barrel reconstruction on GPUs
- New facility for data processing and compression – 1500 CPU/GPU nodes, 60 PB storage

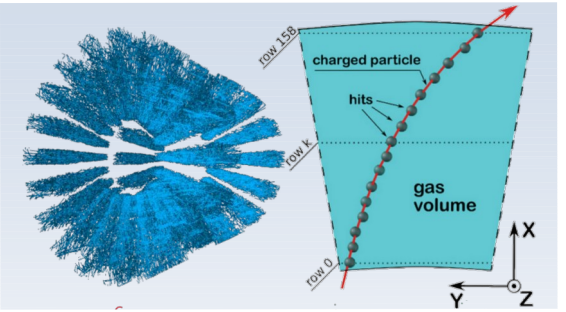


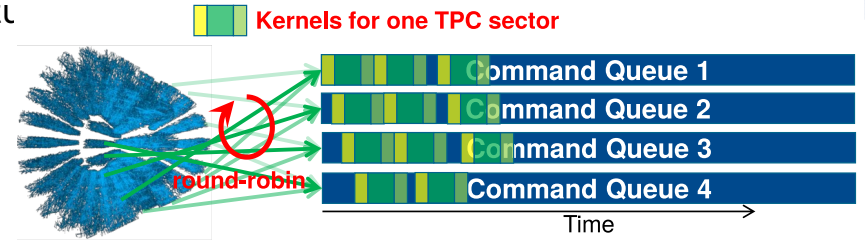
See D. Rohr's vCHEP talk

ALICE TPC reconstruction on GPUs

- Run several events in parallel
- The event size is large, so not too many events fit into GPU memory at once
- Process the sectors of the TPC in parallel
- Same code base for CPU and GPU code

  → can run on either architecture

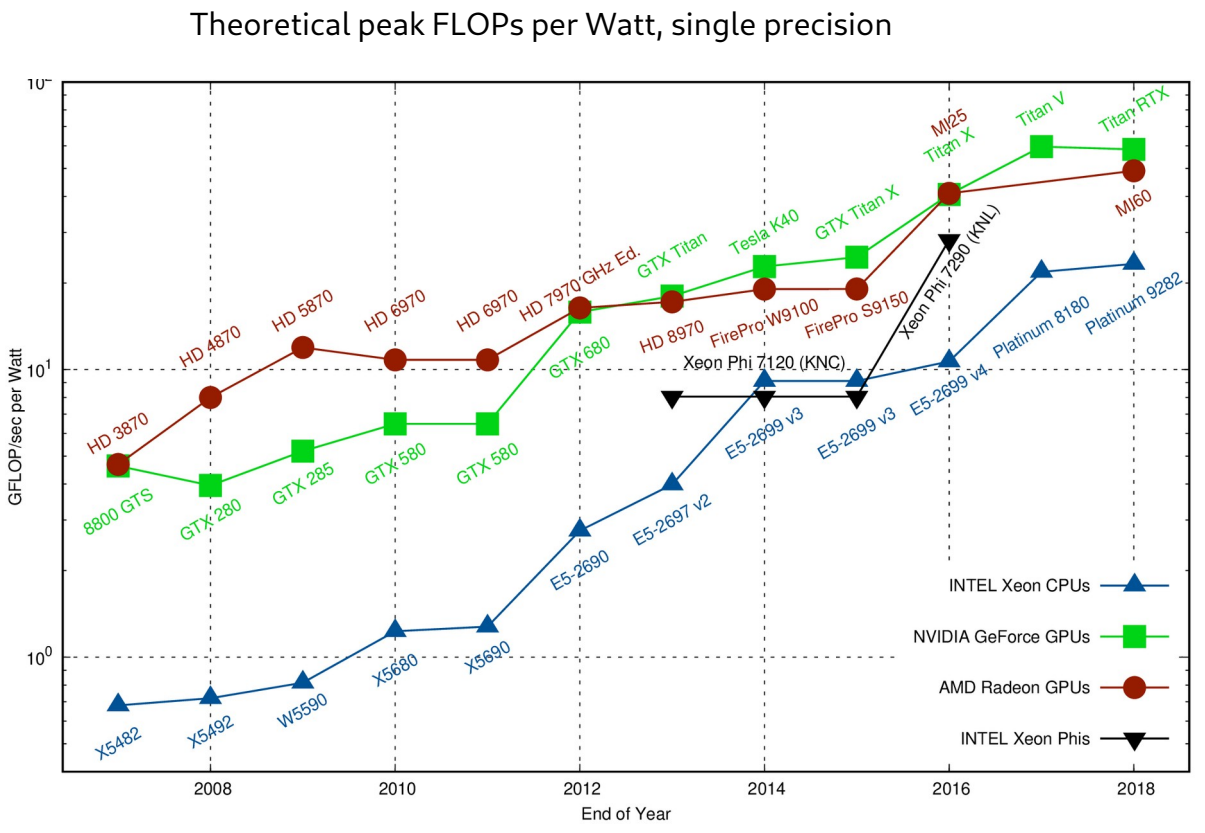




#	Phase	Task	Method	Locality	Time	Device
1	I	Seeding	Cellular Automaton	Very local	30 %	CPU & GPU
2		Track following	Simple Kalman filter	Sector-local	60 %	CPU & GPU
3	II	Track Merging	Matching Covariance	Global	2 %	CPU
4		Final Fit	Kalman filter	Global	8 %	CPU (or GPU)

arxiv 1712.09430

GPU power efficiency

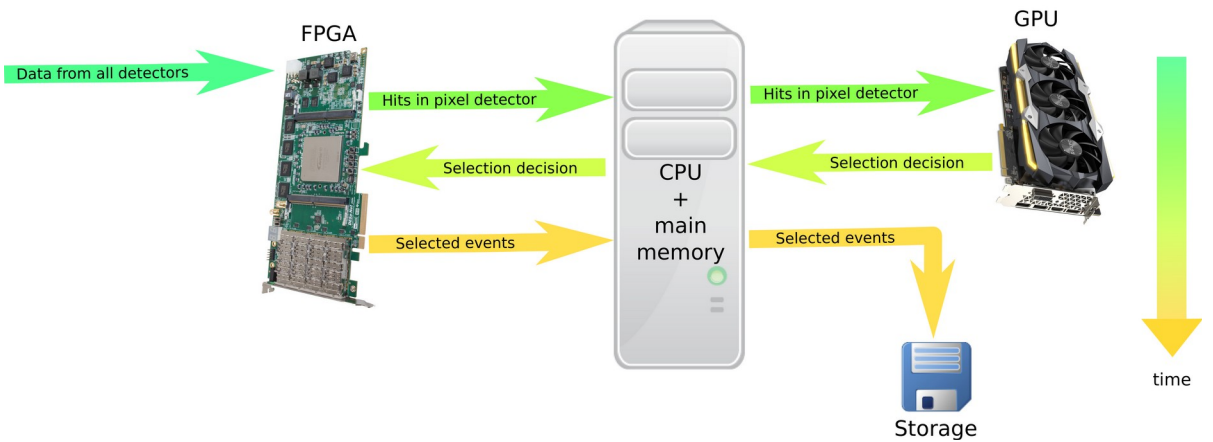


Mu3e experiment

- Fixed target experiment at the Paul Scherrer Institute in Switzerland
- Study lepton flavor violating decay $\mu^+ \rightarrow e^+e^-e^+$
- Triggerless readout @ 10 GB/s, reduce to 100 MB/s with GPU filter farm
- Process 50 ns time slices of data
- Linear track fit for low-momentum particles for real-time data selection implemented on GPUs
- Measured $2 \cdot 10^6$ time slices / s on one Nvidia GTX 1080

  → Can do full event selection with 12 GPUs

- Planned to start data-taking in 2023



EPJ Web of conferences, 2017

Mu3e Technical Design Report: arXiv2009:11690

GPU power efficiency