ACAT 2021

IBS Science Culture Center, Daejeon, South Korea

CMS High Level Trigger performance comparison on CPUs and GPUs

Andrea Bocci

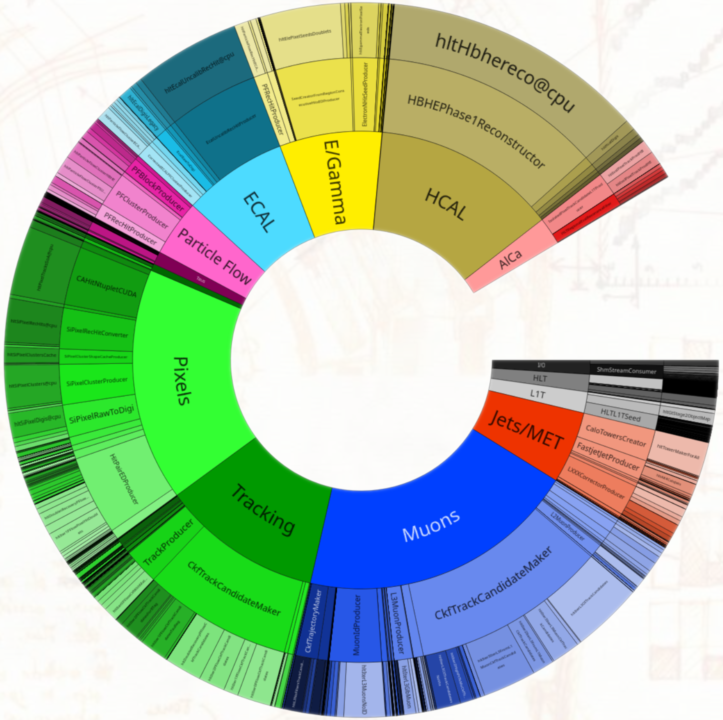
on behalf of the CMS Collaboration

HLT at the end of Run-2

- CMS and LHC scenario at the end of Run-2
  - peak average instantaneous luminosity of $2 \times 10^{34}\ cm^{-2}s^{-1}$
  - about 50 proton-proton collisions per bunch crossing
  - 100 kHz input rate (from the Level 1 Trigger rate)

- a traditional CPU farm
  - over 1000 machines
    - from three different years and generations
  - 716 kHS06
  - 30500 physical CPU cores / 61000 logical cores
    - HLT running with multithreading
    - 15k jobs with 4 threads

331 ms/ev

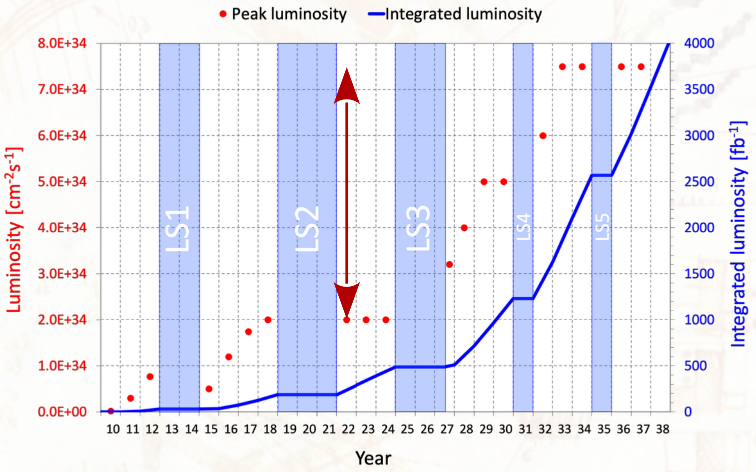


pie chart link – snapshot of the HLT menu under development for Run-3

looking forward

- from The Phase-2 Upgrade of the CMS Data Acquisition and High Level Trigger TDR

	Run-2	Run-3	Run-4	Run-5
peak luminosity	2×10^{34} cm^{-2}s^{-1}	2×10^{34} cm^{-2}s^{-1}	5×10^{34} cm^{-2}s^{-1}	7.5×10^{34} cm^{-2}s^{-1}
pileup	50	50	140	200
HLT input rate	100 kHz	100 kHz	500 kHz	750 kHz
HLT output rate	1 kHz	< 2 kHz	5 kHz	7.5 kHz
HLT farm size	0.7 MHS06	0.8 MHS06	16 MHS06	37 MHS06



- the increase in luminosity, pileup and input rate require a very large increase in the computing power for the HLT farm
- these estimates already take into account
  - +20%/y improvement in performance/cost
  - a further 1.6× (2.5×) software improvements for Run-4 (Run-5)
- however
  - a traditional computer farm of this size would not fit in the CMS HLT data center
  - still missing at least a further 2× improvement

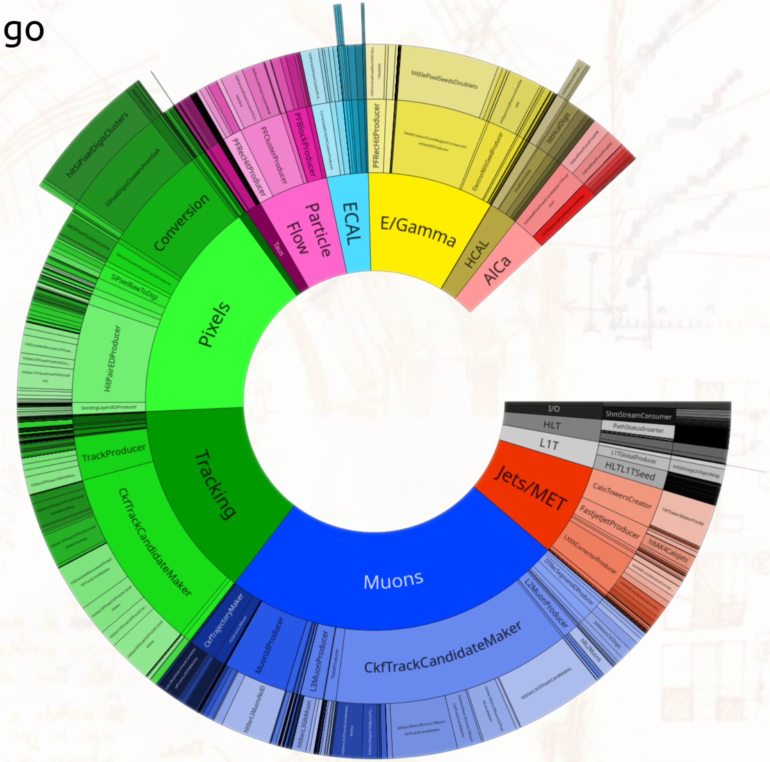
can we use *accelerators* to close the gap ?

HLT on GPUs

- the latest effort in CMS to use GPUs for reconstruction started 5 years ago
  - 2016
    - first attempts of porting the pixel-only track reconstruction at *EuroHack 2016* in Lugano
  - 2018 – 2019
    - *Patatrack* "demonstrator" for using GPUs at HLT
  - 2020 – 2021
    - integration in the experiment's software
    - official adoption for HLT in Run-3
    - work on *performance portability*
- Run-3 is the ideal scenario for testing new tecnologies !
  - no external pressure from LHC conditions
  - gain experience
    - using a heterogeneous software in a production environment
    - in procuring, commissioning and running a GPU-equipped data centre
    - reduce number of racks and power consumption
  - take advantage of the extra computing capacity
    - high rate real time *data scouting* at HLT
    - spare capacity on the GPUs for porting more algorithms

253 ms/ev

24% faster !



pie chart link – snapshot of the HLT menu under development for Run-3

performance comparisons

- compare the performance of the HLT …
- … running on different kind of hardware …
  - CPU-only
  - with offloading to local GPUs
- … with two complementary approaches
  - measure the performance of the full HLT menu
    - overall system performance
    - most of the times will be constraned by one of the two resources
      - either CPU-limited or GPU-limited
  - measure the performance of the offloadable part of the HLT
    - runs completely on CPU or (almost) completely on GPU
      - CPU still used for overall orchestration, scheduling, I/O, etc.
    - allows a more direct comparison of the CPU and GPU performance
      - of course, affected by the *software implementation* as well as the hardware
      - just a snapshot of the "state of the art" in CMS

how we run these measurements:

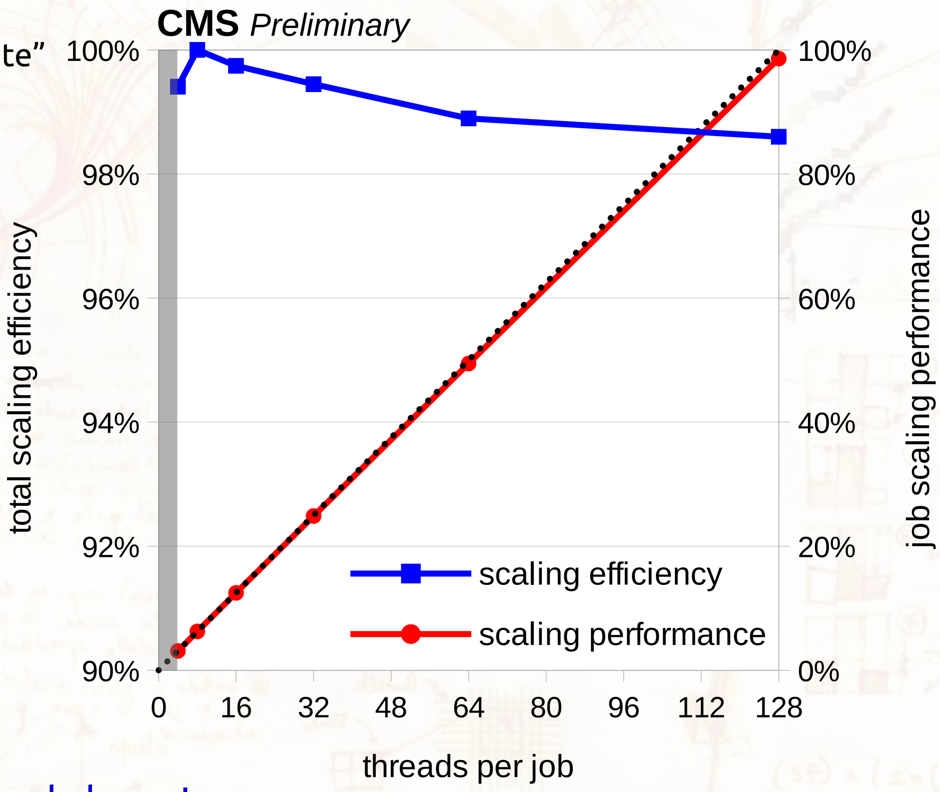
- run over real data collected in 2018, with a pileup of 50 proton-proton collisions
- unbiased by the HLT selection: reproduce the population of events seen in input by the HLT
- data use the uncompressed "FED raw data" binary format, to reproduce the HLT behaviour and minimise the impact of I/O
- when benchmarking the full HLT menu, or runnig the offloadable part of the HLT only on CPUs, use many multithreaded jobs in parallel to completely saturate the processors on the machine
- when benchmarking the offloadable part of the HLT on GPUs, use as many jobs and threads to fully utilise the GPU
- code compiled for the "common denominator" architecture (currently `-msse3`)

full HLT on CPUs

multithreading vs multiprocess

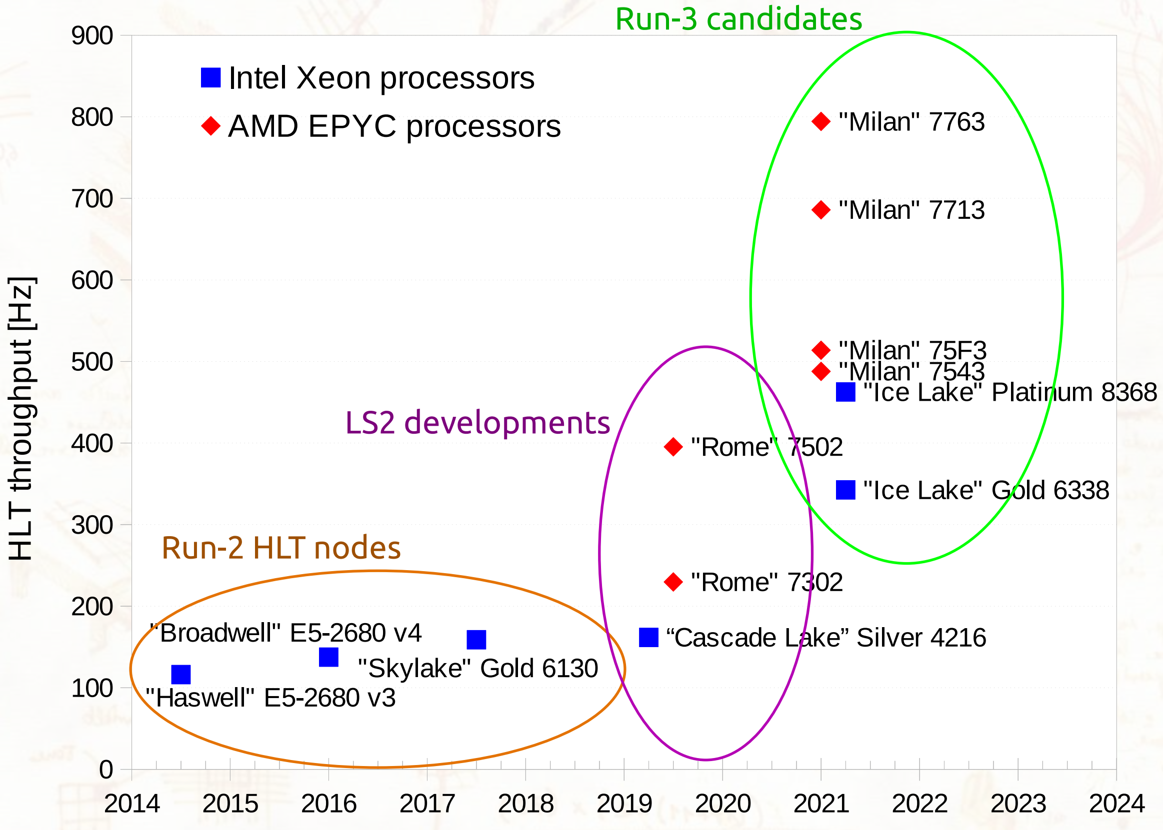
- multithreading is essential to reduce the memory usage
  - in Run-1, HLT used multiple single-threaded jobs with “copy-on-write”
  - in Run-2, switched to multithreaded jobs (with 4 threads each)
- multithreading can allow efficient sharing of resources
  - in Run-3, adapt the number of threads to optimise GPU usage
- requires efficient scaling vs the number of threads

- system hardware
  - dual AMD EPYC Milan 7763, with SMT, and 256 GB of RAM
  - 2 sockets × 64 cores × 2 threads = 256 threads
- from 4 to 128 threads per job
  - 1 and 2 threads per job not feasible due to memory usage
  - 256 threads would incur in NUMA and synchronisation effects
- as many jobs as necessary to fill the whole machine



- no significant performance loss is observed from 4 threads and above !

HLT performance



snapshot of the HLT menu under development for Run-3

HLT performance for x86 CPUs from multiple generations and vendors

- HLT nodes from Run-2
- nodes for HLT development in LS2
- candidate HLT nodes for Run-3

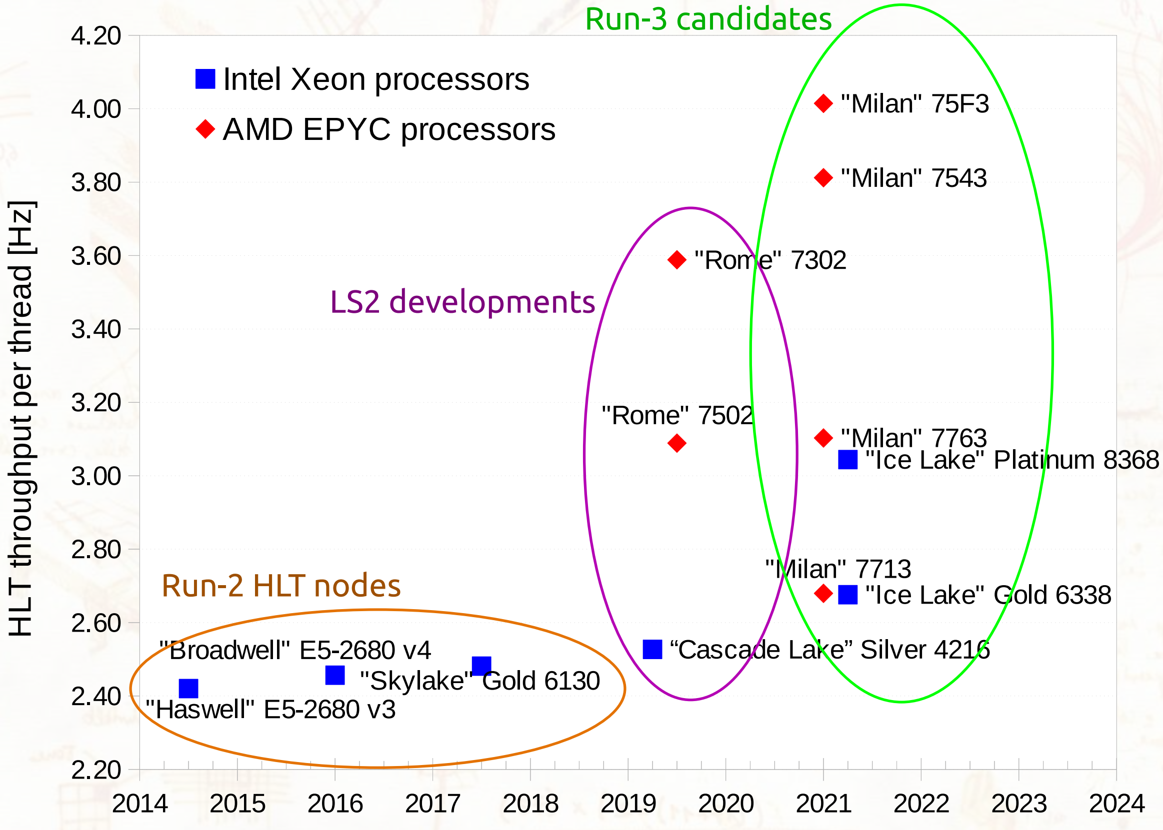
measurements

- on fully loaded dual socket machines
- with a snapshot of the HLT menu under development for Run-3

improvements over the years come from

- larger number of cores per socket
- better performance per core

HLT performance



snapshot of the HLT menu under development for Run-3

HLT performance for x86 CPUs from multiple generations and vendors

- HLT nodes from Run-2
- nodes for HLT development in LS2
- candidate HLT nodes for Run-3

measurements

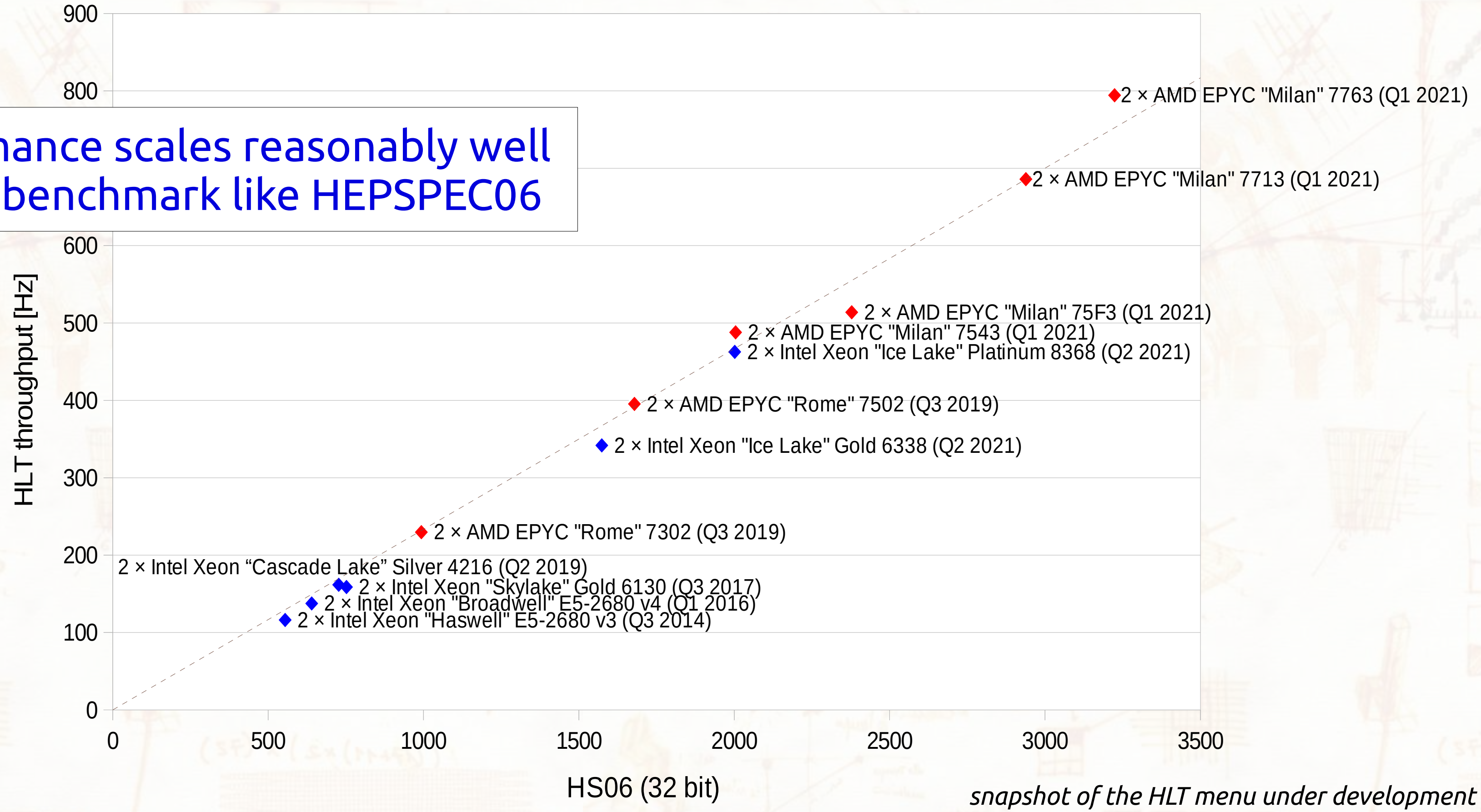
- on fully loaded dual socket machines
- with a snapshot of the HLT menu under development for Run-3

improvements over the years come from

- larger number of cores per socket
- better performance per core

HLT throughput vs HS06

HLT performance scales reasonably well with an old benchmark like HEPSPEC06



snapshot of the HLT menu under development for Run-3

full HLT on GPUs

full HLT with offloading to GPUs

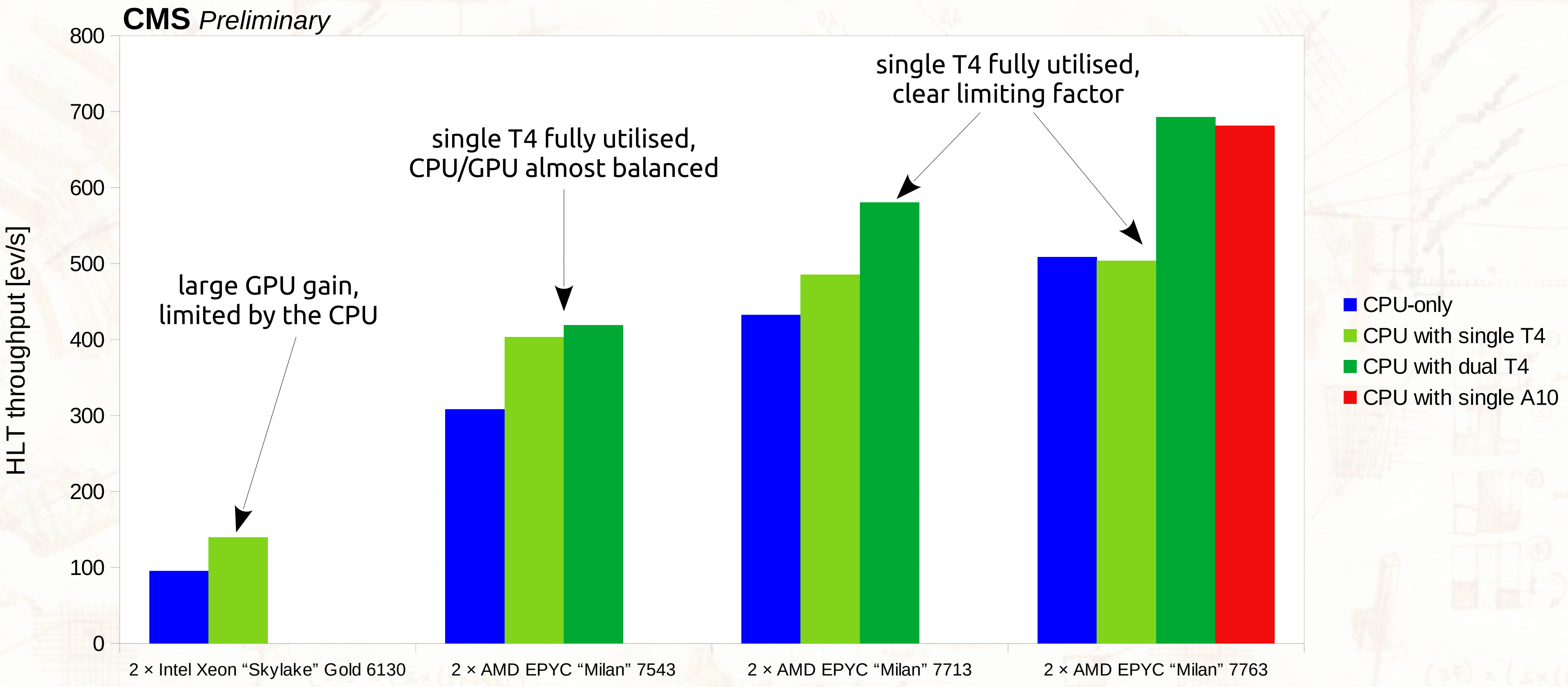
- CMS HLT can offload four main components to GPUs
  - pixel tracker local reconstruction
  - pixel-only track and vertex reconstruction
  - electromagnetic and hadronic calorimeter local reconstruction

- the impact of using GPUs depends
  - on the fraction of time used by those components
    - and for converting the results of the GPU-based reconstruction to the legacy data formats
  - on the combination of CPU and GPU being used

- coupling different CPUs and GPUs, either of the component can be the limiting factor
  - depending on the number and complexities of the algorithms running being offloaded
  - and how frequently they are used

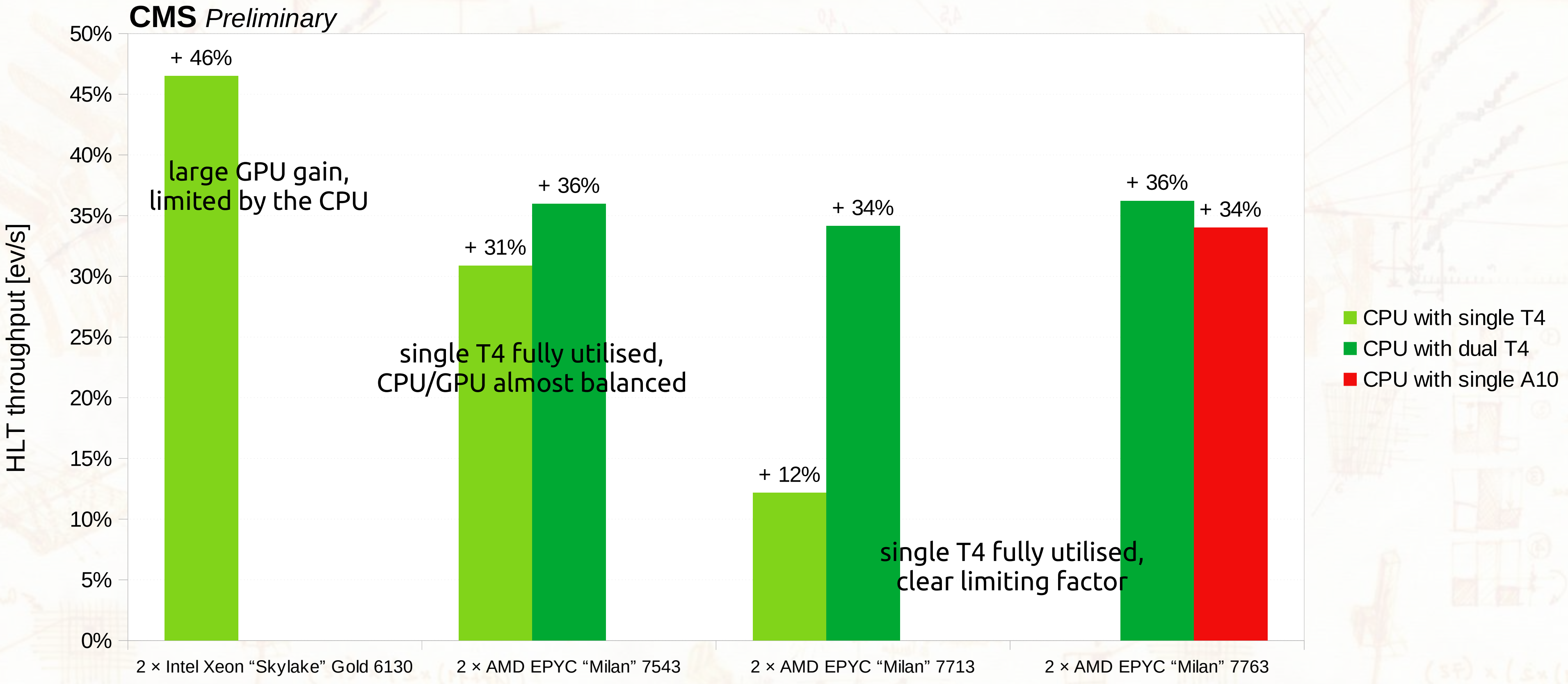
- we illustrate this comparing the gain from using GPUs for different hardware combinations

offloading to different GPUs



snapshot of the HLT menu under development for Run-3

offloading to different GPUs



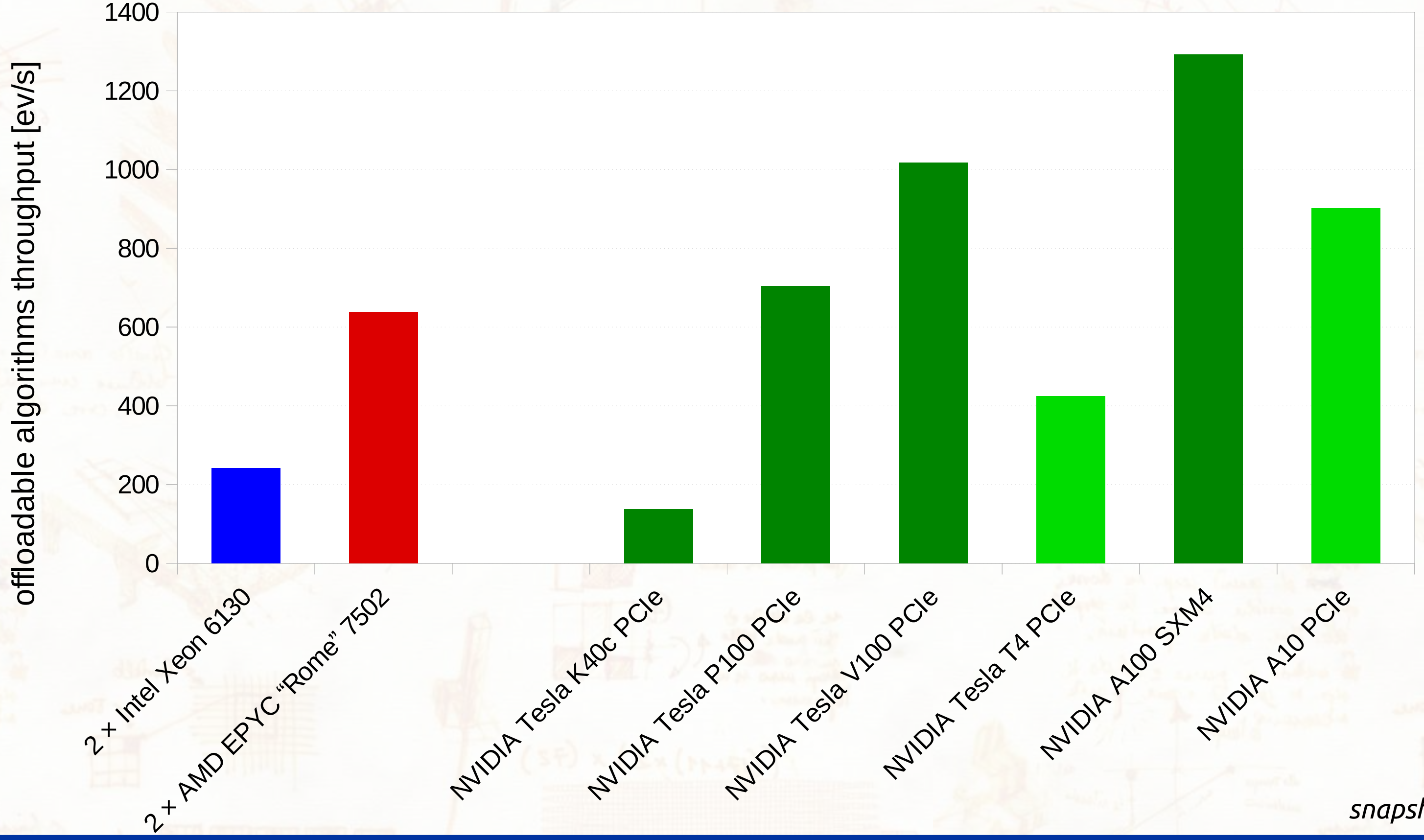
snapshot of the HLT menu under development for Run-3

offloadable algorithms

offloadable algos on CPUs and GPUs

- CMS HLT can offload four main components to GPUs
  - pixel tracker local reconstruction
  - pixel-only track and vertex reconstruction
  - electromagnetic and hadronic calorimeter local reconstruction
- the impact of using GPUs depends
  - on the fraction of time used by those components
    - and for converting the results of the GPU-based reconstruction to the legacy data formats
  - on the combination of CPU and GPU being used
- run only these algorithms to benchmark different GPUs (almost) independently from the host CPU

benchmarking different compute



running on CPUs

- fully loaded CPUs
- multiple jobs

running on GPUs

- fully loaded GPU
- single job, 8-10 threads
  - except V100 with 4 jobs
  - < 10% load on CPUs

snapshot of the HLT menu under development for Run-3

conclusions

conclusions

- over the past 5 years CMS has brought the use of GPUs for physics reconstruction from the R&D to the production stage
  - for deployment on a fully heterogeneous HLT farm
  - for opportunistic use of HPC and other grid resources

- this introduces new challenges related to benchmarking, procurement, allocations, …
  - different workflows will have different benefits from offloading
  - many hardware combinations will leave either the CPUs or GPUs underutilised
  - with some hardware and software combinations, offloading may actually harm performance !

- R&D activities are always ongoing !
  - rewrite more algorithms using parallel implementations suiatable for offloading to GPUs
  - the next goal is *performance portability* (with Alpaka – see the poster by W. Redjeb)

thank you for your attention

any questions ?