Status update Goethe Universität

Themenbereich I

FIDIUM Meeting

20.10.2022

V. Lindenstruth, A. Redelbach

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Overview

Efficient usage of GPU acceleration is crucial for many (upcoming) experiments in high energy and nuclear physics.

Some further development and generalisations of existing solutions to be integrated in the context of the FIDIUM project.

Subsequent steps:

- **Local reconstruction**: finding patterns of clusters and hits on the basis of digital measurements in a single detector
- **Track finding** includes estimation of track positions and track candidates
- **Track fitting** determines the best track candidates
- **Determination of vertices** finally allows to disentangle decay topologies and ideally also particle decay chains

First focus on milestone: Performance-Vergleichsmessungen von Laufzeiten repräsentativer Rekonstruktionsalgorithmen bei Prozessierung auf CPUs und GPUs

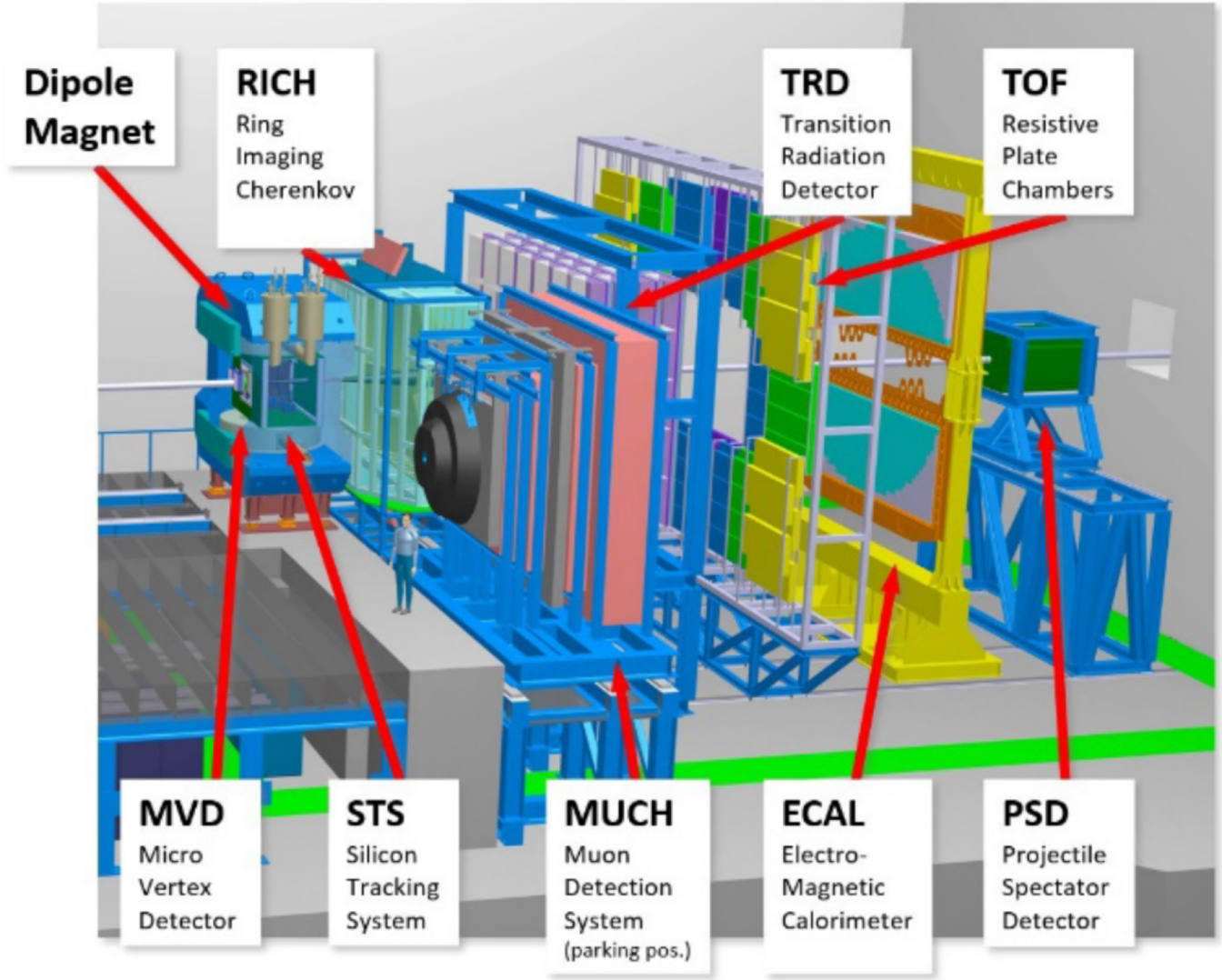
CBM detector for benchmarks

No hardware triggers and high interaction rates

Efficient software-based triggering required

Tracking mainly by STS stations in magnetic field

→ Computations for tracking in STS system consist of **generic reconstruction tasks**



Hardware for benchmarks

CPU (x2):

- Intel Xeon Gold 6130 (Skylake, server);
- Total cores (HT threads): 16 (32);
- Base (max) frequency: 2.10 (3.70) GHz;
- Cache: 22 MB L3 cache;
- Instruction set extension: SSE 4.2, AVX, AVX2, AVX-512.

AMD GPU:

- AMD Radeon VII (Vega20);
- Compute Units (threads per CU): 60 (64);
- Clock rate: 1450 (1800) MHz;
- Memory: 16 GB @ 1000 MHz.

NVIDIA GPU:

- NVIDIA GeForce RTX 2080 Ti;
- Compute Units (threads per CU): 68 (32);
- Clock rate: 1350 (1545) MHz;
- Memory: 11 GB @ 1750 MHz.

Concept of Kalman Filter Track Fitting

Track fitting with KF:

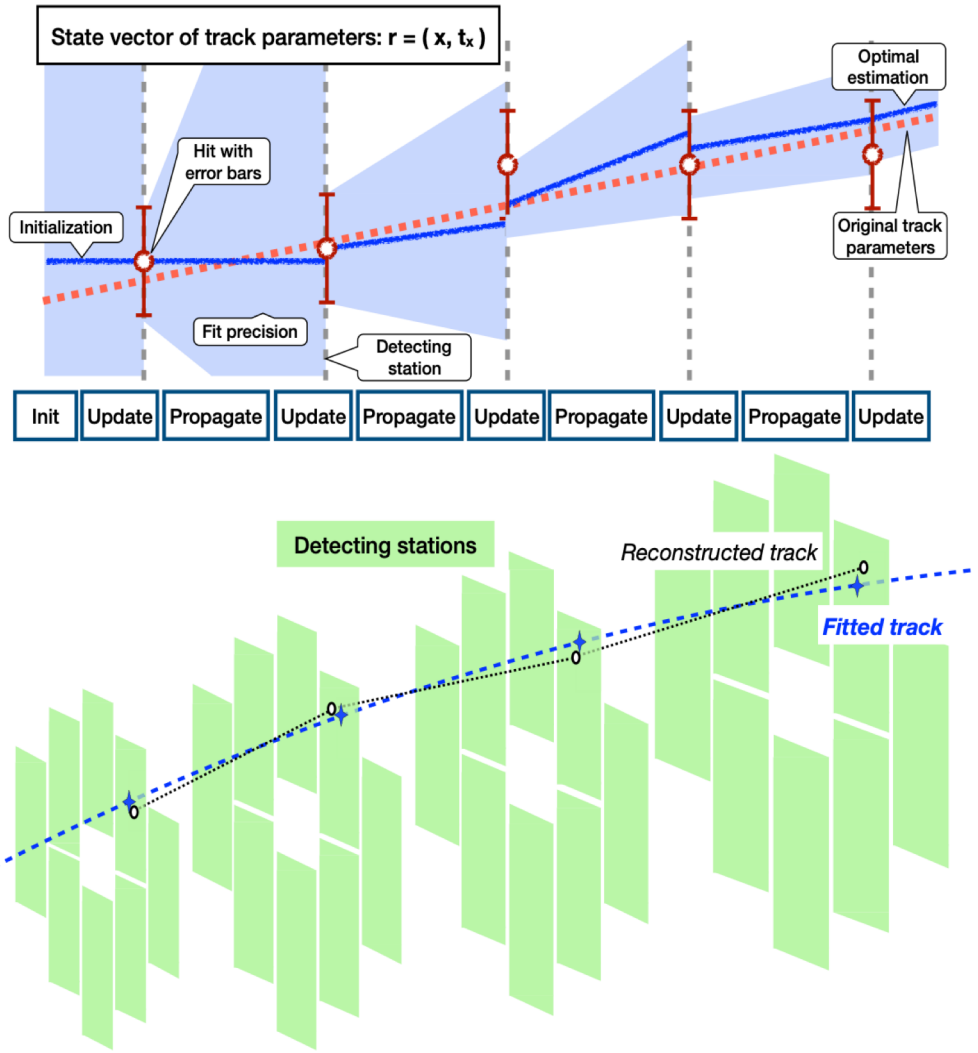
- determines the **track parameters** by sequentially adding hits with updating the **state vector** and **covariance matrix**;
- independently fits each individual track;
- floating point calculations;
- has great potential for **parallelization at both thread and data levels**.

Parallel calculations:

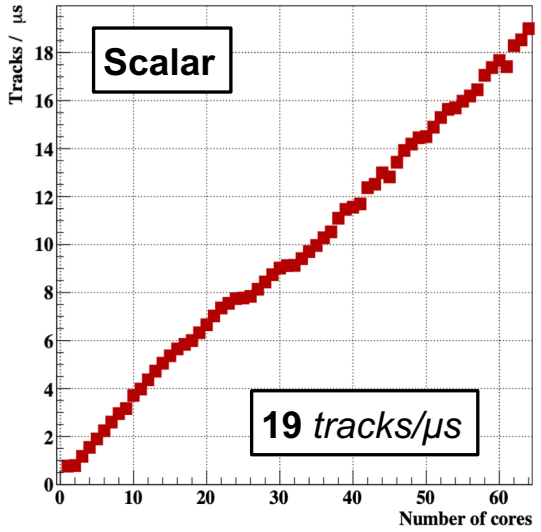
- data level (SIMD): scalar, SSE, AVX2, AVX-512 (headers and Vc);
- task level CPU: OpenMP (CPU affinity);
- task level GPU: OpenCL.

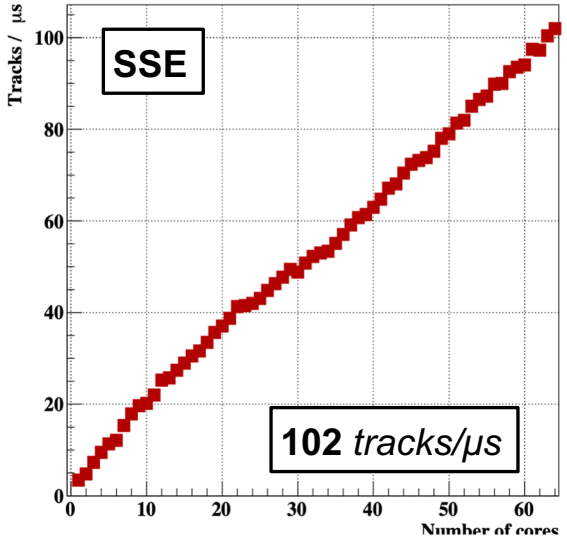
Input data:

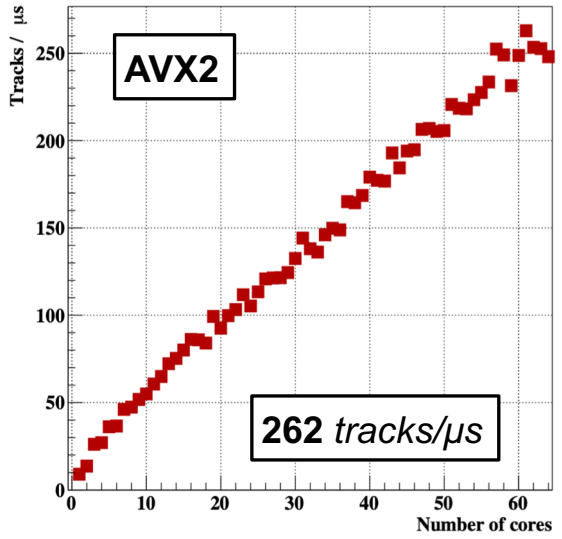
- CBM STS simulated reconstructable tracks;
- 8 hits long tracks only;
- flexible dataset size to evenly fill all threads.

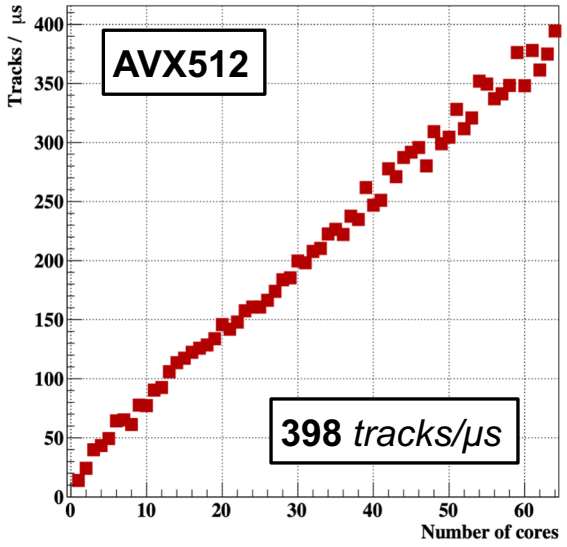


KF Track Fitter | CPU scalability





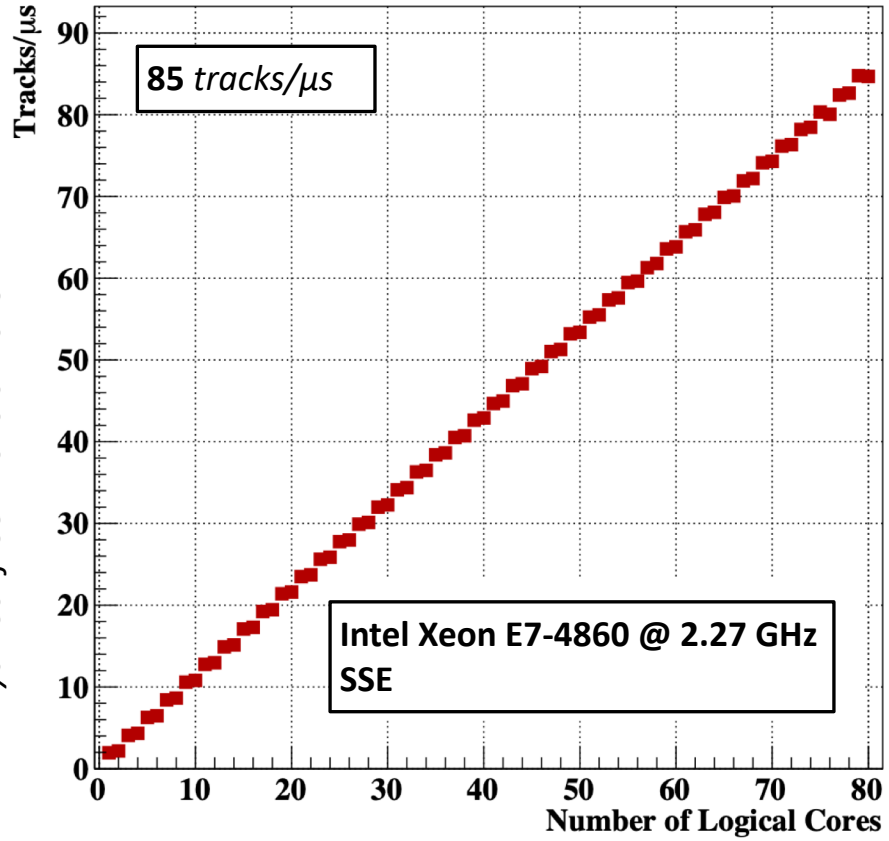


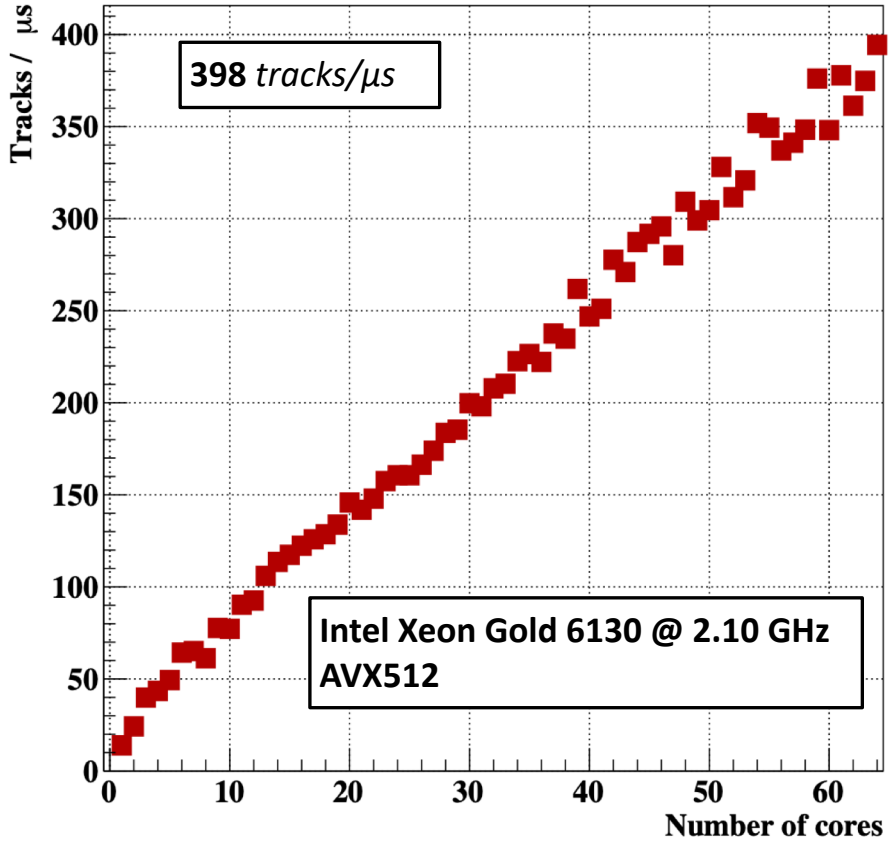


- Shape of the graphs is close to linear. Result of optimized algorithm and mutual independence of individual tasks
- Speed of SIMD calculations is higher than expected due to better utilization of the cache.
- Execution time of AVX instructions is less stable than SSE or scalar ones, especially when using a large number of threads.
- Maximum speed up factor relative to scalar calculations is 21.

KF Track Fitter | CPU | comparison with old results

Ivan Kisel and for CBM Collaboration 2018
J. Phys.: Conf. Ser. **1070** 012015

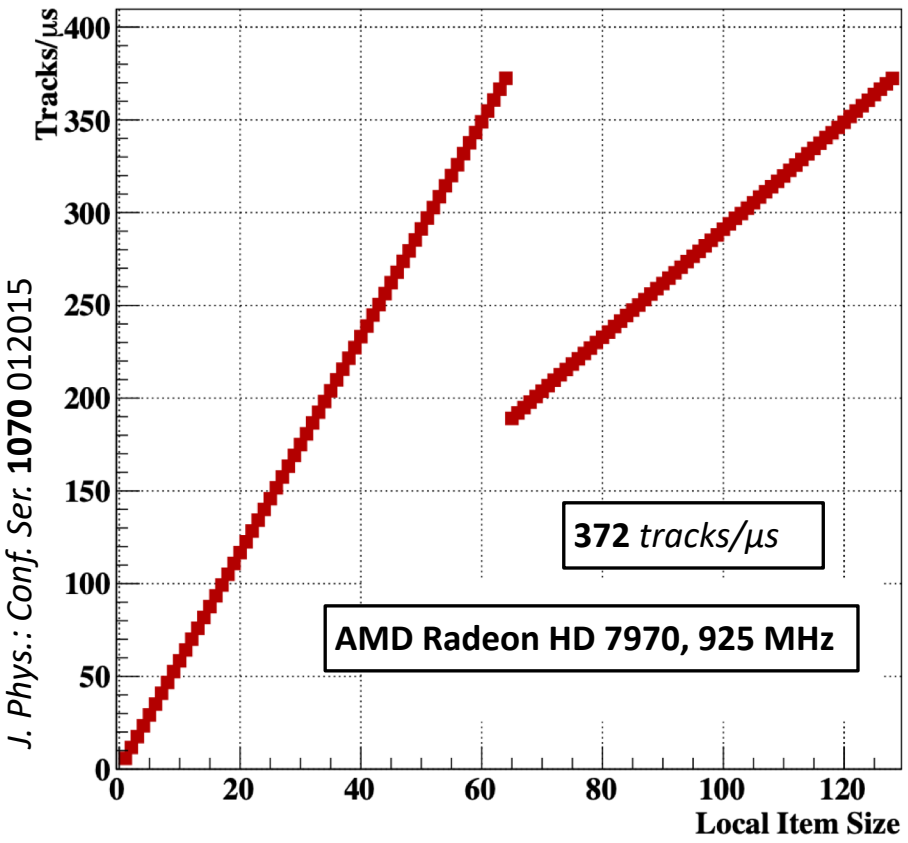


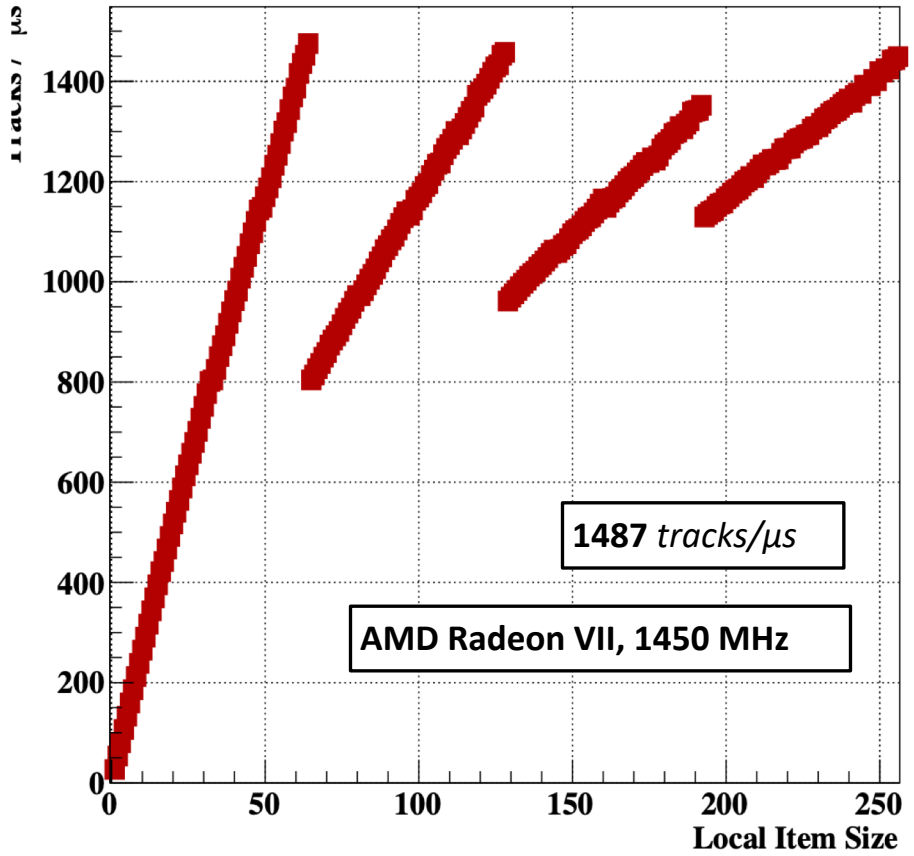


- New benchmarks calculation speed advantage reaches 4.5 at its peak or exceeds 5.8 if the number of computational threads is equal.
- The new AVX-512 graph is less stable than the old SSE, but is still close to linear.

KF Track Fitter | AMD GPU | comparison with old results

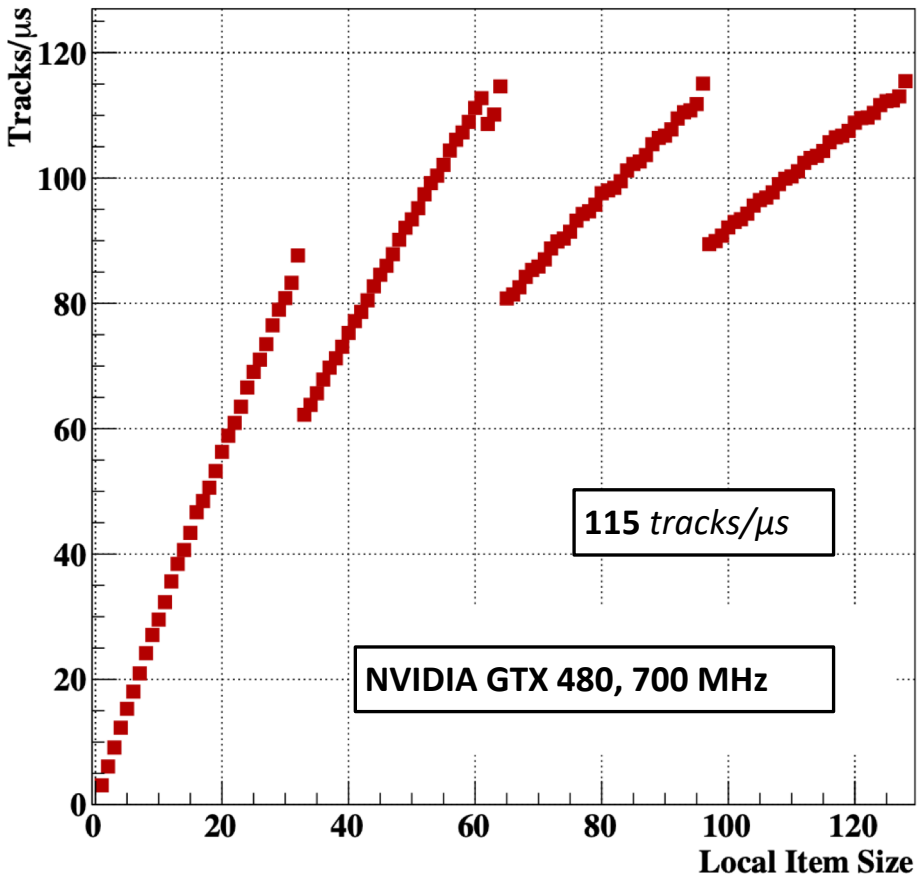
Ivan Kisel and for CBM Collaboration 2018
J. Phys.: Conf. Ser. **1070** 012015

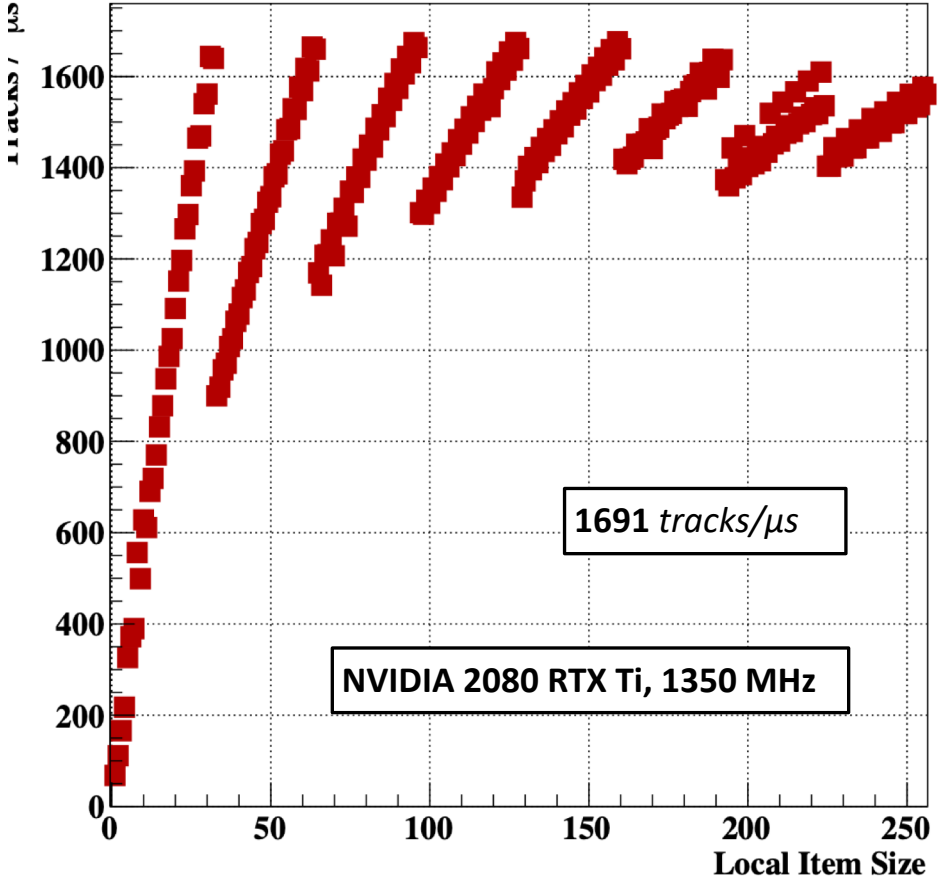




- The stair-like structure of the scalability graph with a step of 64 corresponds to the number of threads per AMD GPU Compute Unit.
- maximum and optimal work- group size is fixed at 256
- The increase in performance when using the new GPU corresponds to the difference in device characteristics: the number of Compute Units and clock speed.

KF Track Fitter | NVIDIA GPU | comparison with old results



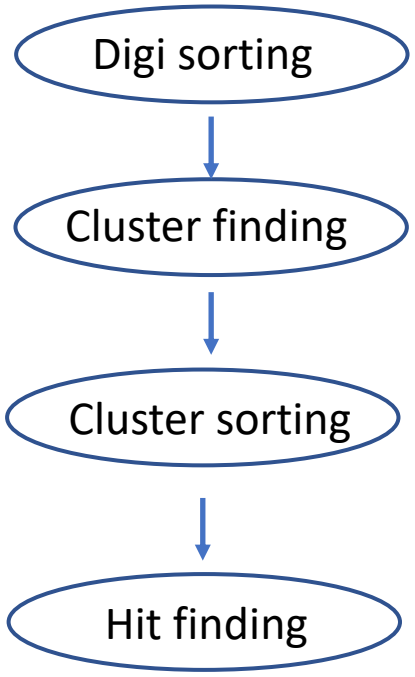


- Stair-like structure of the scalability graph with a step of 32 corresponds to the number of threads per NVIDIA GPU Compute Unit.
- Increase in performance when using the new GPU corresponds to the difference in device characteristics: the number of Compute Units, clock speed, and memory speed.

Concept of hit finding in CBM STS

CBM STS detector:

- 8 stations (fixed target scheme);
- geometry v21e with 876 micro-strip sensors.



CBM STS hit finder:

- digi sorting - by channel and by time;
- cluster finder - merges digi in adjacent channels if the time difference is small;
- cluster sorting - by time;
- hit finder - intersection of clusters with similar timestamp.

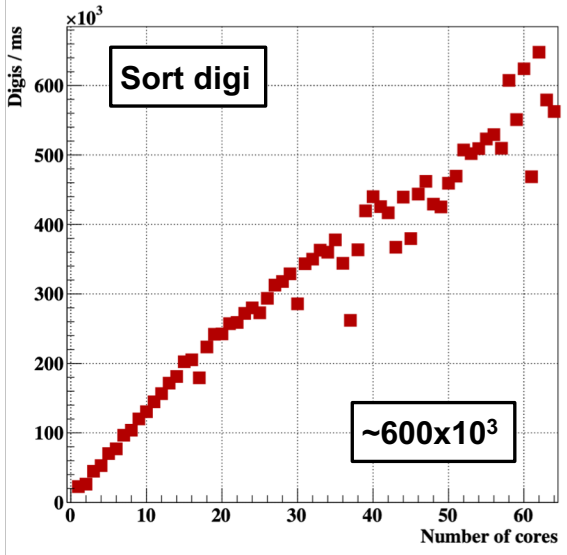
CBM STS hit finder on CPU:

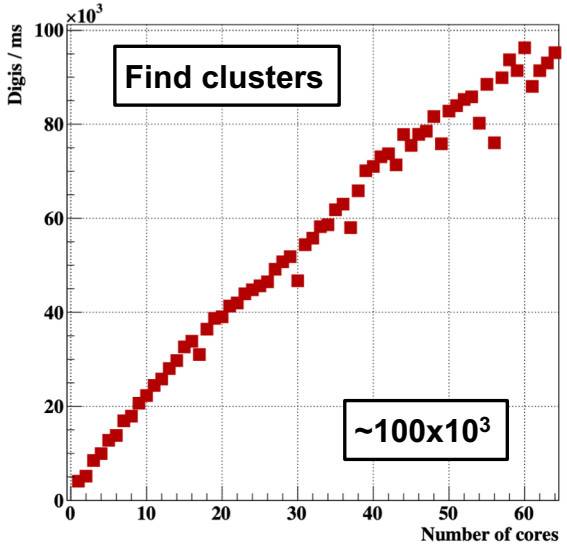
- OpenMP (CPU affinity);
- global parallelization - processing single sensor per thread;
- sorting with std::sort and data movement.

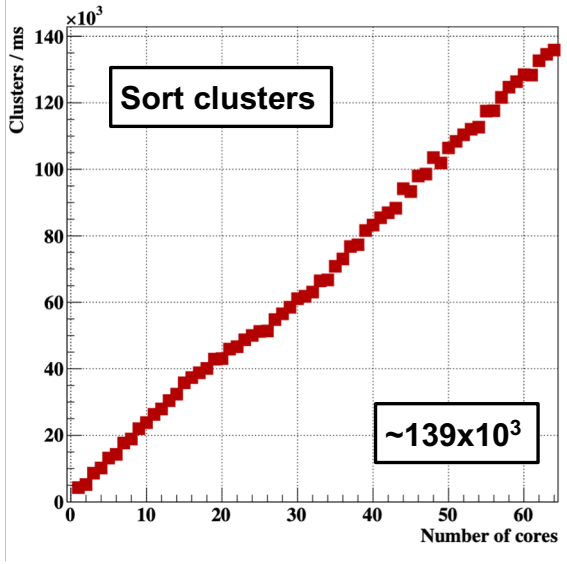
CBM STS hit finder on GPU:

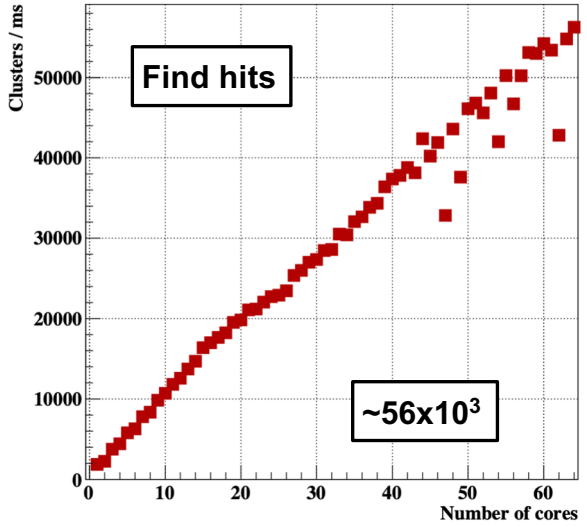
- XPU framework;
- intra-sensor parallelization - multiple threads operate with the same sensor to ensure optimal utilization of the GPU capabilities;
- sorting by GPU-optimized parallel Mergesort algorithm without actual data movement.

CBM STS Hit Finder | CPU scalability



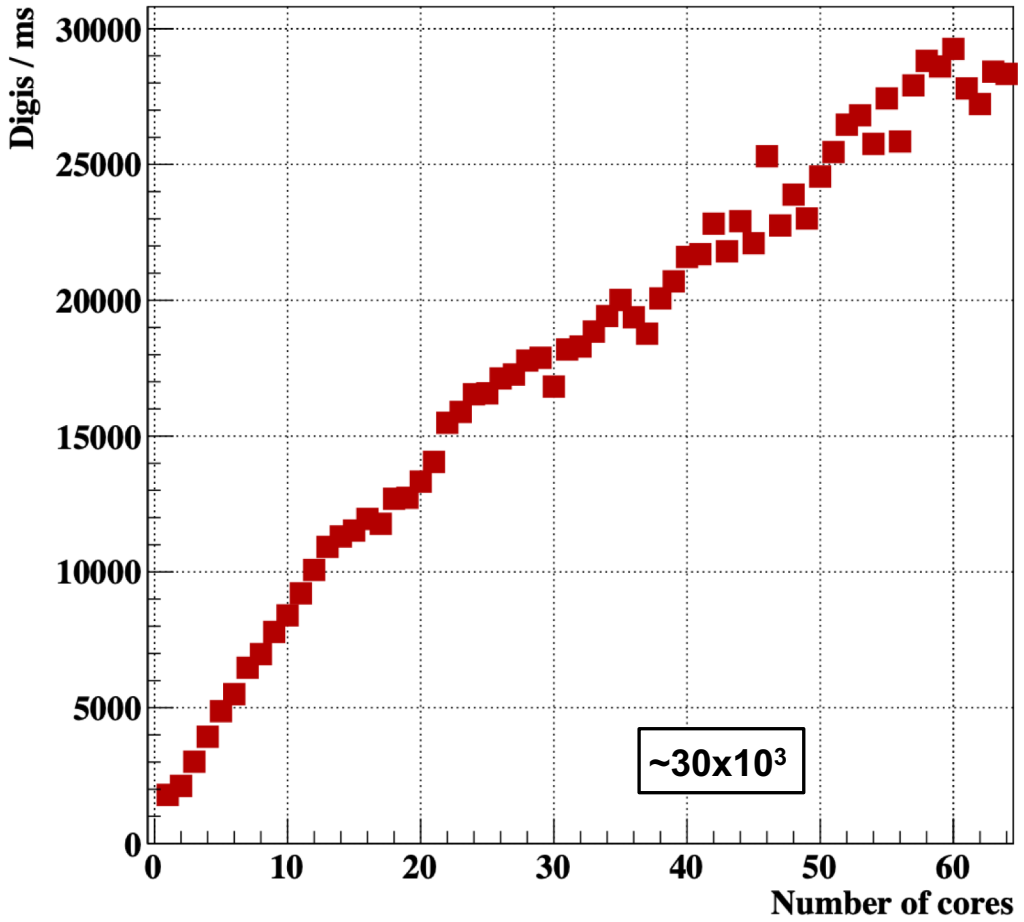






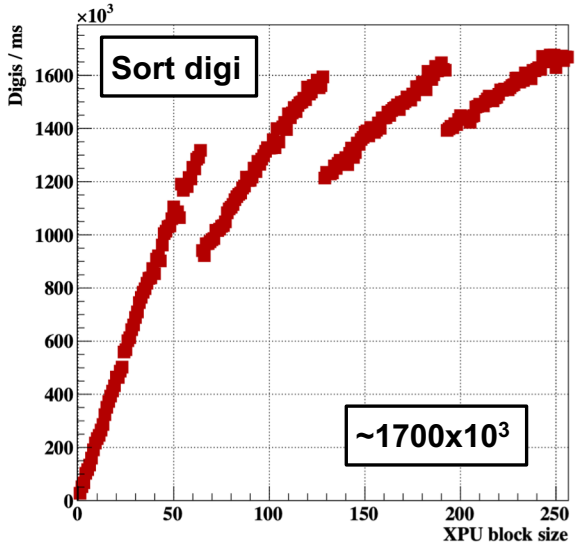
- Test dataset: 1 time-slice, 25 million digis.
- Average computation time for threads.
- Linear increase in performance depending on the number of threads.
- There are effects from third-party minor background processes.
- Cluster sort is significantly slower than digi sort due to the difference in object sizes.

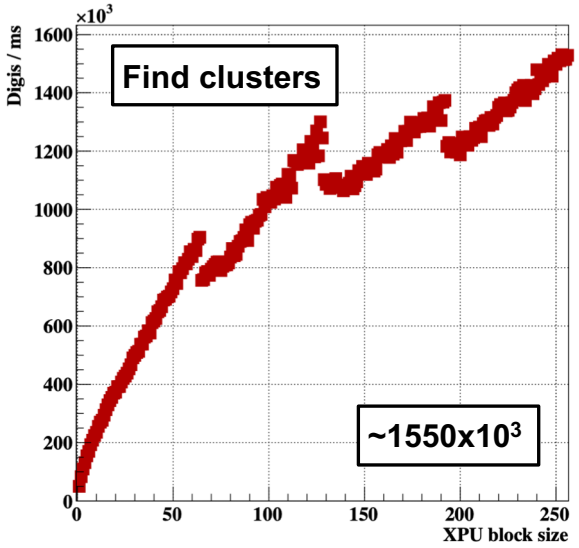
CBM STS Hit Finder | CPU scalability | Total time

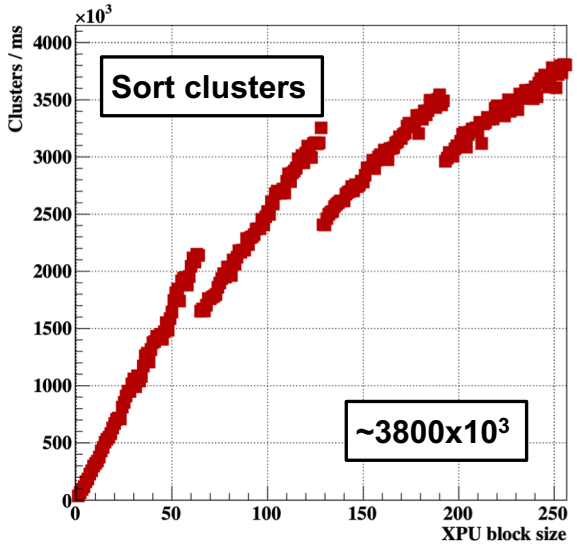


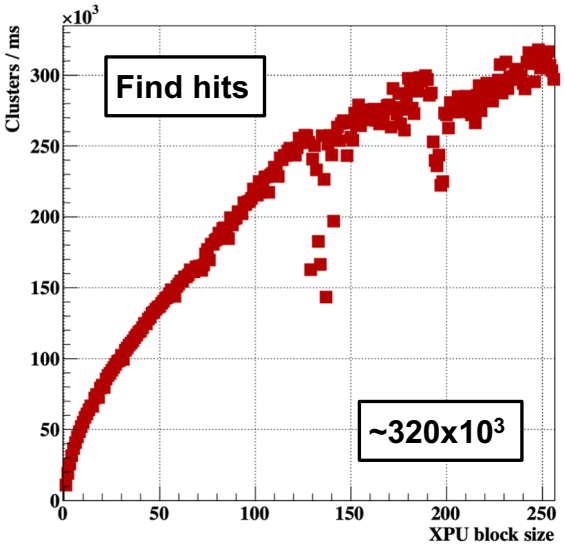
- Close to linear histogram structure with a some amount of distortion.
- Total computation time i. e. the difference between the start and end timestamps of the parallel region – determined by the work of the slowest thread.

CBM STS Hit Finder | AMD GPU scalability



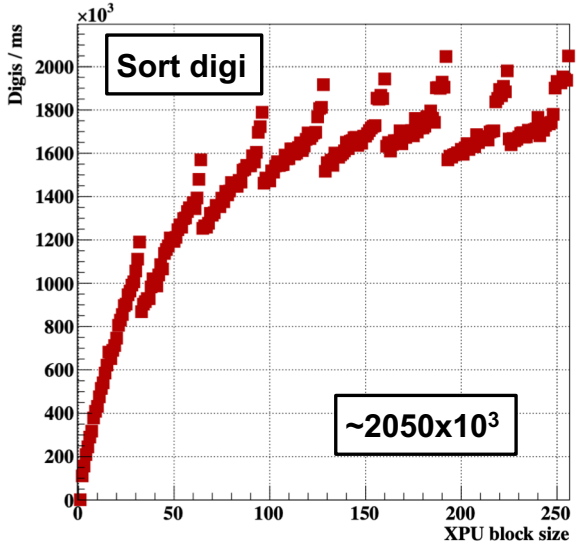


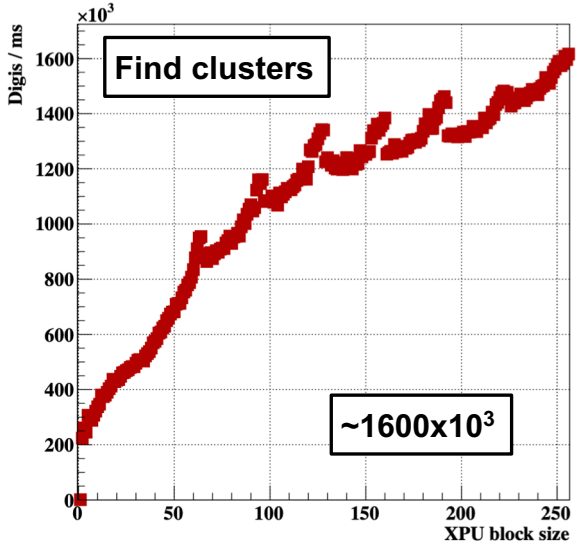


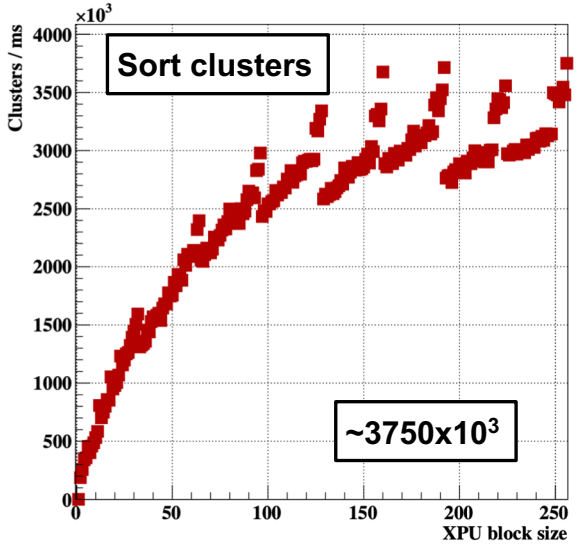


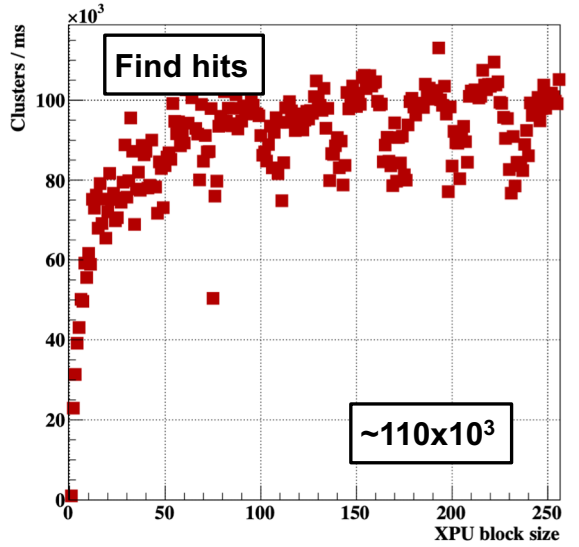
- Stair-like structure of the scalability graph with a step of 64 corresponds to the number of threads per AMD GPU Compute Unit.
- Optimal compute speed at maximum XPU block size.
- Performance can be reduced for insufficient data to fully load large the system (see also backup)
- Cluster sorting (1 key) is twice as fast as digi sorting (2 keys) because it is implemented without moving objects.
- Multiple performance increase relative to CPU results.
- Requires around 5.5 GB of available GPU memory.

CBM STS Hit Finder | NVIDIA GPU scalability



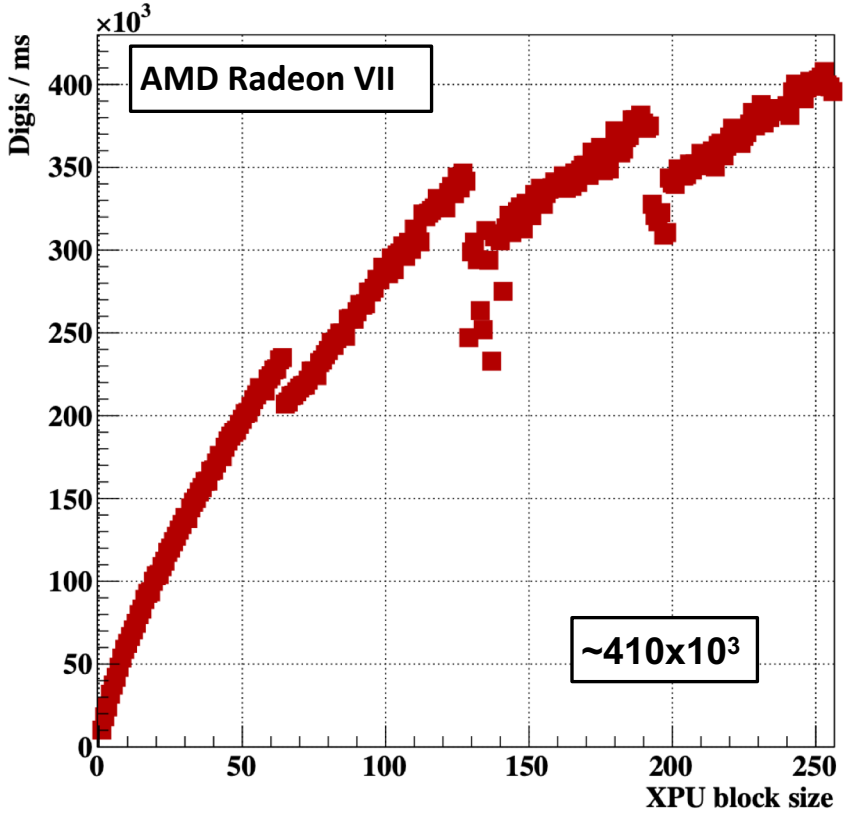


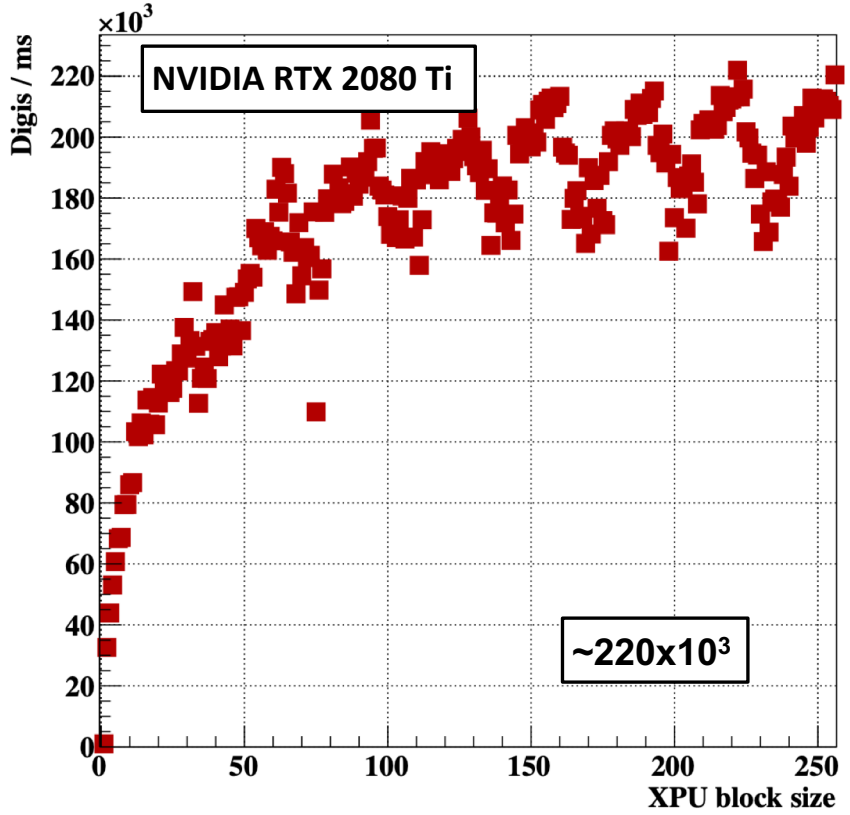




- The stair-like structure of the scalability graph with a step of 32 corresponds to the number of threads per NVIDIA GPU Compute Unit.
- The non-linearity of the segments highlights the importance of choosing a block size that is a multiple of 32.
- Requires around 5.5 GB of available GPU memory.

CBM STS Hit Finder | GPU scalability | Total time





- Most relevant contributions from cluster (~linear) and hit finding (~sawtooth)
- Bus bandwidth (effective memory speed x bus width) can become bottleneck
- Bus bandwidth for AMD 1024 GB/s vs. NVidia 616 GB/s

Summary and next steps

Detailed study of performances of reconstruction codes on CPUs/GPUs:

- Different architectures
- Tasks for local reconstruction
- Track fitting with optimal loading of threads
- Scalability for growing size of input datasets

→performance of each device can vary significantly depending on the settings and the size of the dataset

Focus on the context of first milestone: Set of benchmarks for CPU/GPU performances

Integration in other computing environments to be discussed

Goal: Efficient utilisation of CPU and GPU resources in compute intensive workflows

References and sources

SIMD KF Track Fitter (on Google.Drive):

- OpenMP version for CPU:

https://drive.google.com/file/d/1AM582g4on6AuH6JVJKK1QVhYNvJ6Lyvj/view?usp=sharing

https://goo.su/On5bI5q

- OpenCL version for GPU:

https://drive.google.com/file/d/1fhft9Idc-ixYgQoIcoGlsJa5uuDbxJtm/view?usp=sharing

https://goo.su/JzOY6

CBM STS Hit Finder:

- CbmRoot framework:

https://git.cbm.gsi.de/computing/cbmroot

- STS Hit Finder:

https://git.cbm.gsi.de/computing/cbmroot/-/tree/master/reco/detectors/sts

- XPU framework and examples:

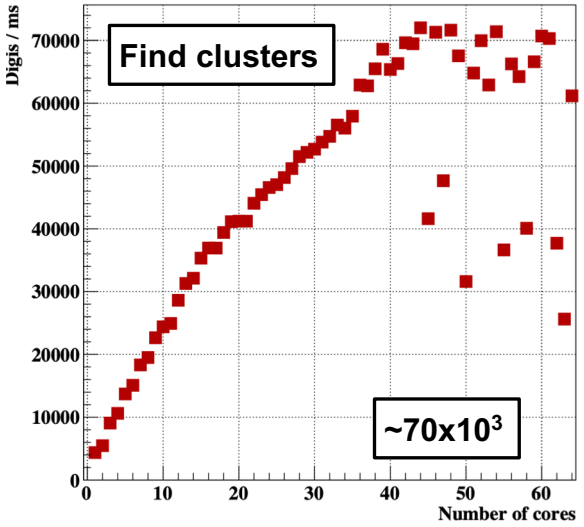
https://github.com/fweig/xpu

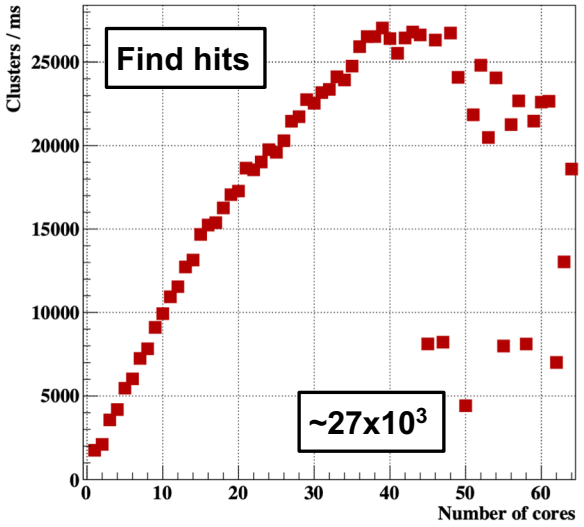
BACKUP

Small versus big time slice comparison

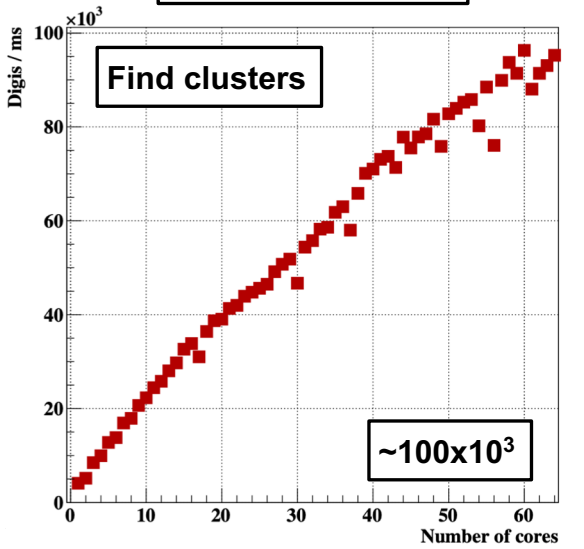
CBM STS Hit Finder | Cluster and hit finding on CPU

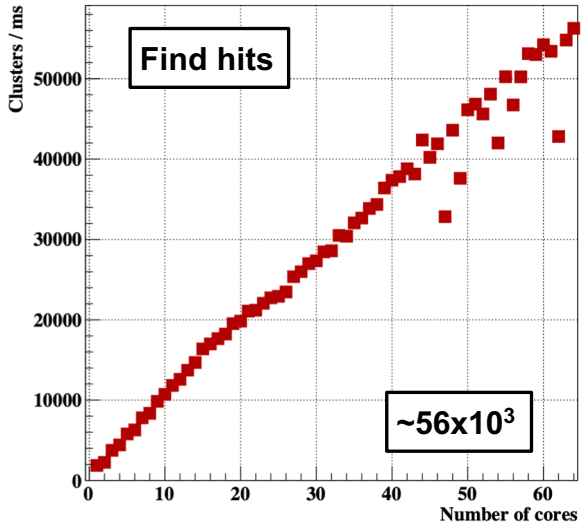
2.5 million digi



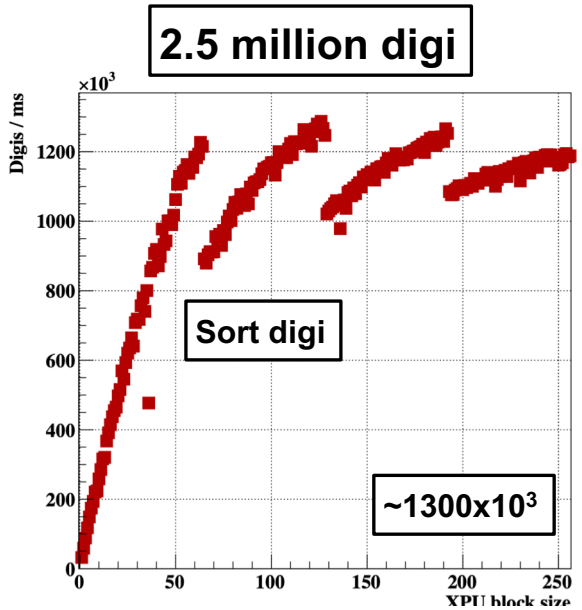


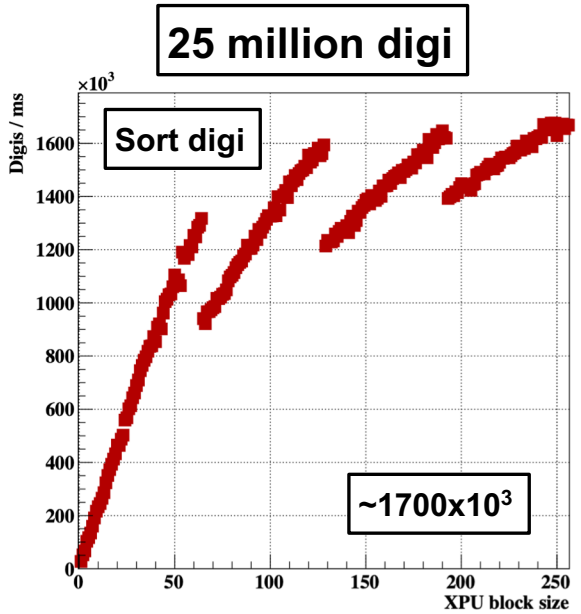
25 million digi

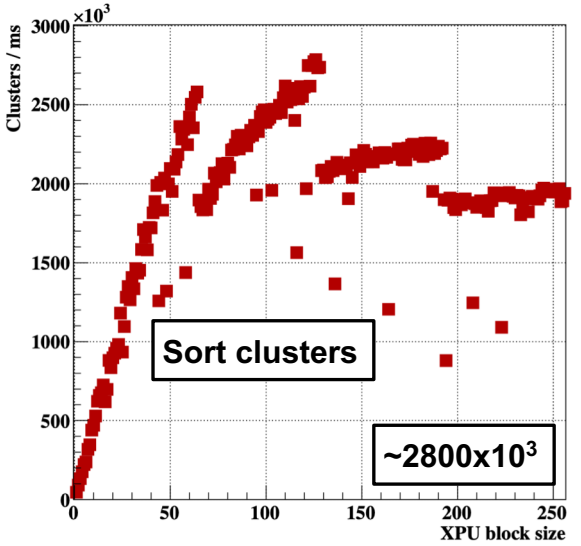


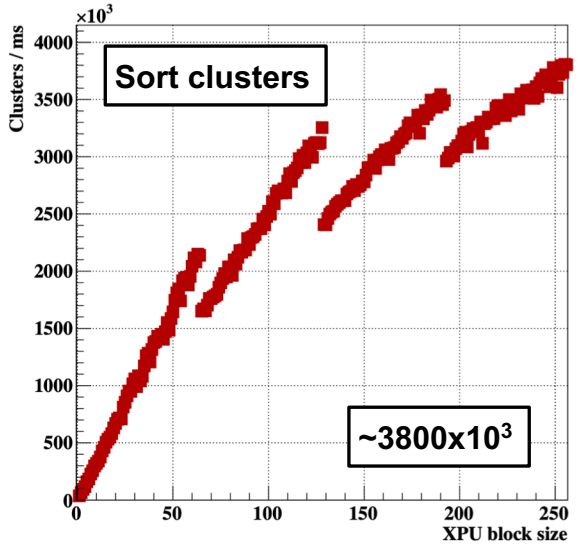


CBM STS Hit Finder | Cluster and hit sorting on AMD GPU

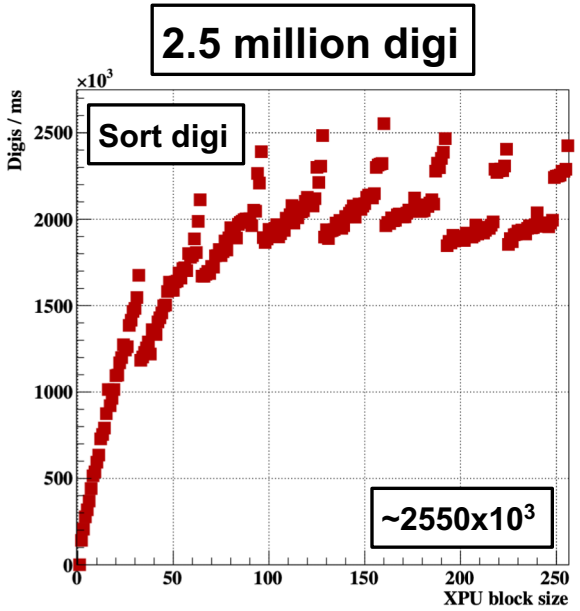


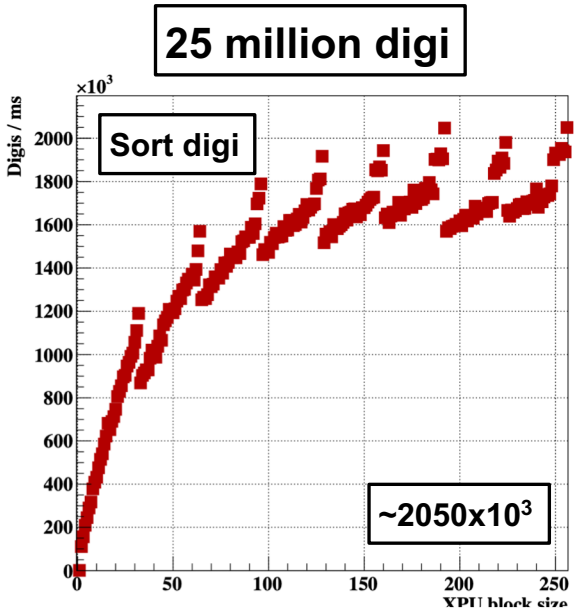


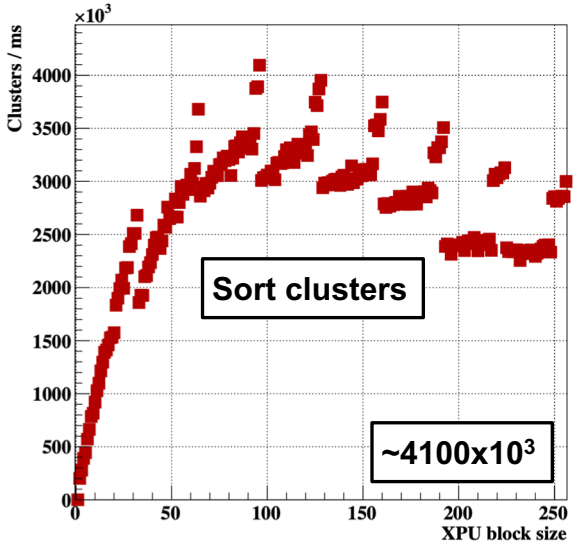


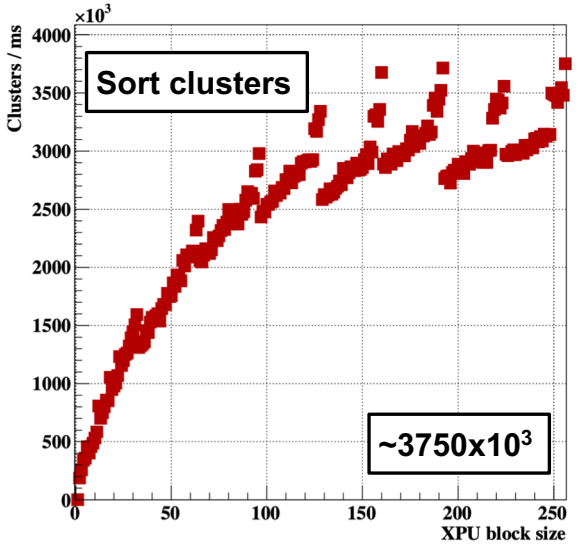


CBM STS Hit Finder | Cluster and hit sorting on NVIDIA GPU

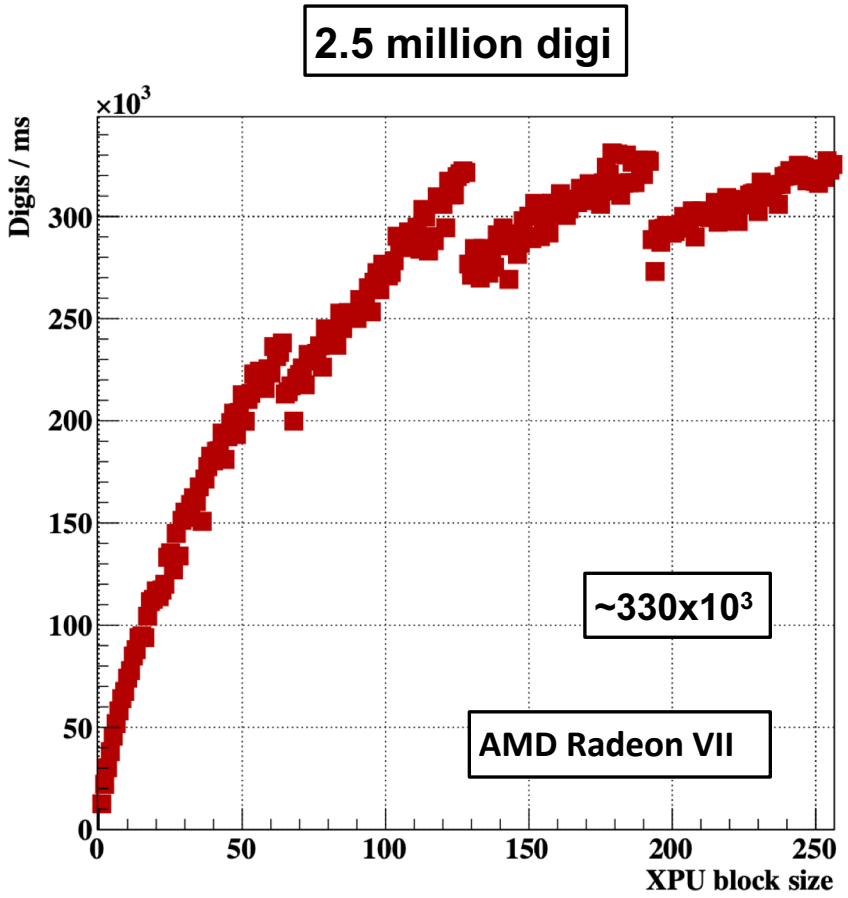


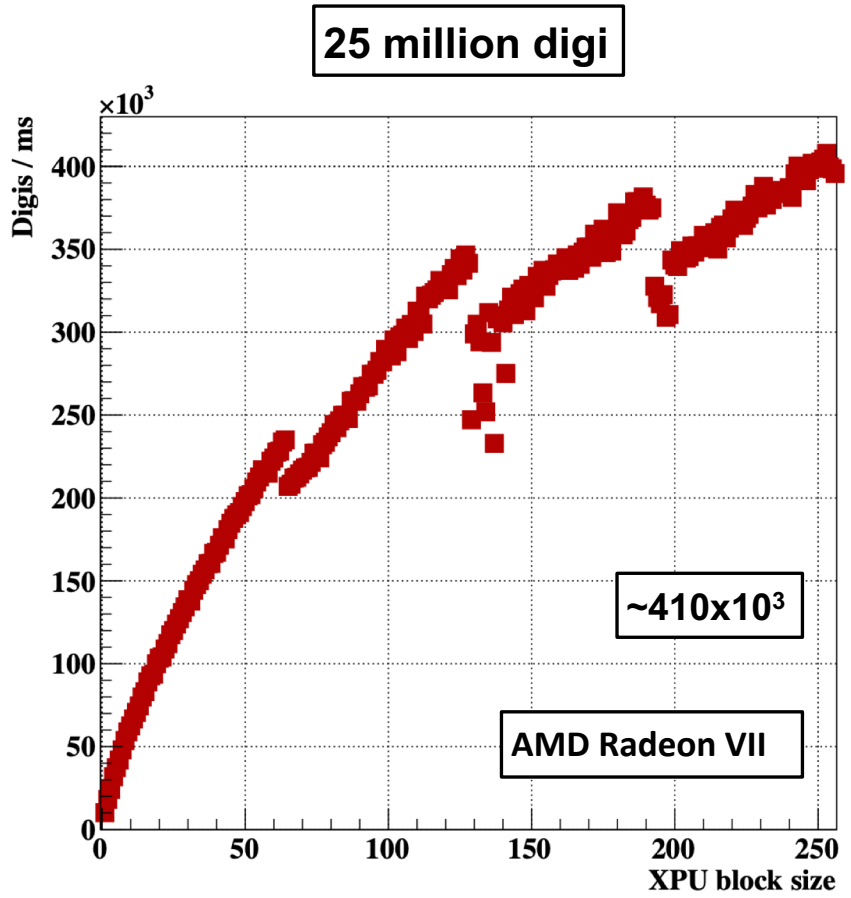






CBM STS Hit Finder | AMD GPU scalability | Total time





CBM STS Hit Finder | NVIDIA GPU scalability | Total time