UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

ATLAS
EXPERIMENT

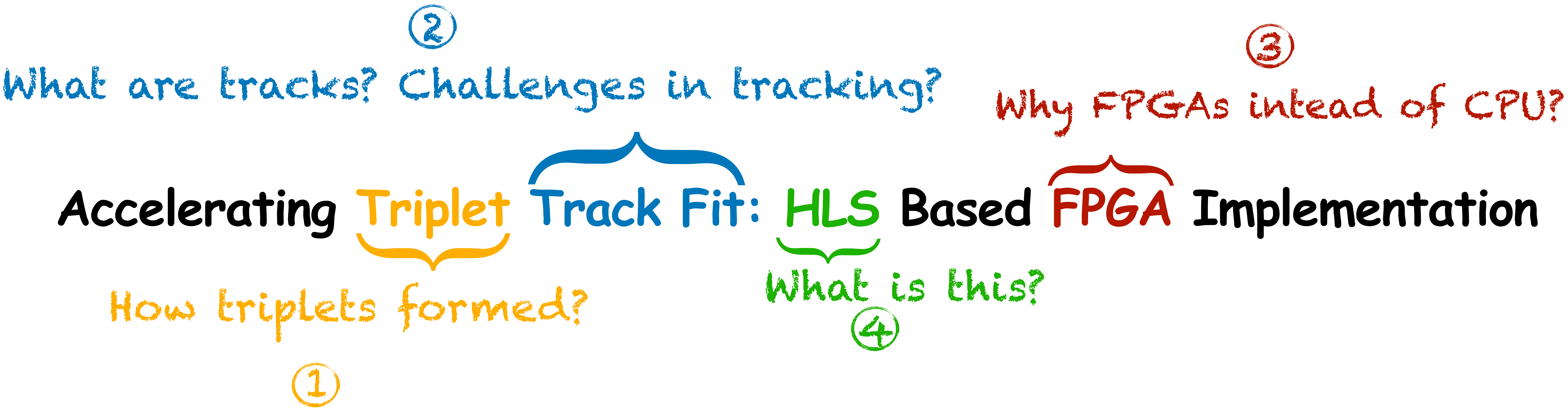
Accelerating Triplet Track Fit: HLS Based FPGA Implementation

ATLAS Heidelberg Meeting, Trifels

19.07.2024

Kadir Tastepe
(PI)

Outline of Today's Talk:



② What are tracks? Challenges in tracking?

③ Why FPGAs intead of CPU?

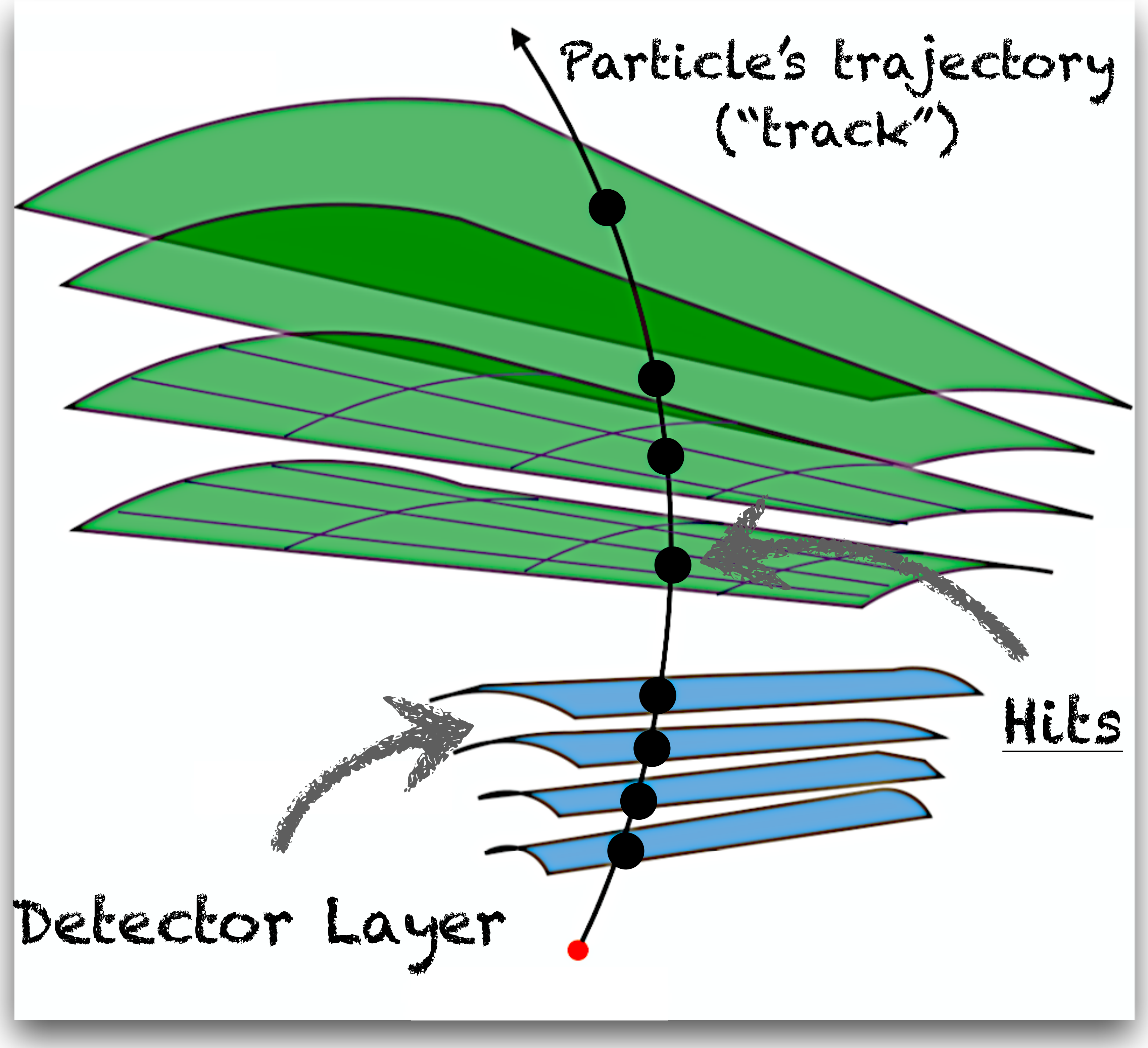
Accelerating Triplet Track Fit: HLS Based FPGA Implementation

How triplets formed? ①

What is this? ④

Tracking Particles

- Particles are measured by their interaction with matter
- Position (x, y, z) is measured in the Tracking Detector
- Momentum measurement in the magnetic field B



Note: Current inner detector (ID); will be inner Tracker (ITk) after Phase-II upgrade

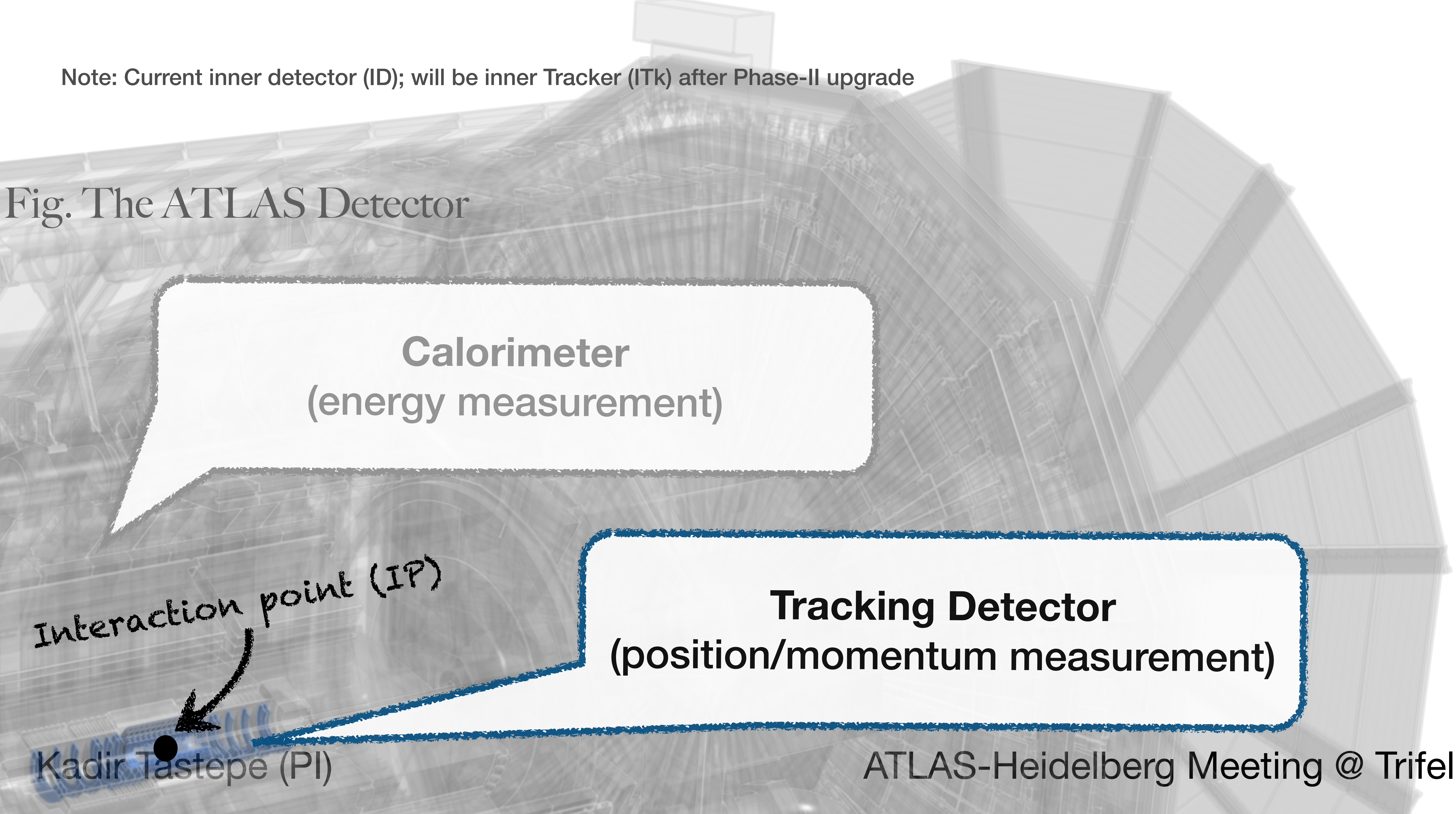


Fig. The ATLAS Detector

Calorimeter
(energy measurement)

Tracking Detector
(position/momentum measurement)

- Signals (~Hits) generated as particles pass through detector layers.

Objective: Reconstruct particle's trajectory from measured hits

Tracking Challenges

Tracking gets more and more challenging with increasing number of hits…

Shared Hits

Also a triplet

Real (HEP) World!

Tracking Challenges in HL-LHC

Reconstructed Tracks
Pixel Clusters
Strip Clusters

Many “Footprints” (hits) …

Tracking Challenges in HighLuminosity-LHC

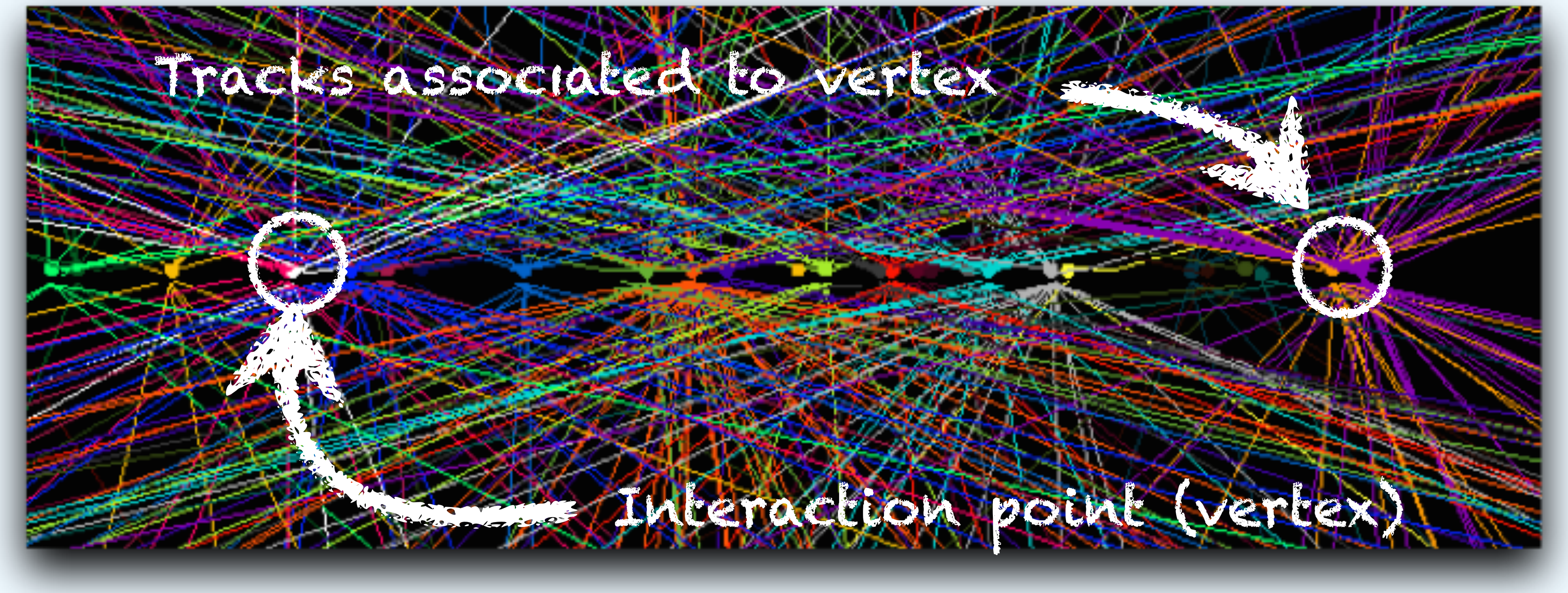
"~Number of interactions"

For reference

LHC (design) Luminosity $\sim 10^{34}\ cm^{-2}/s$

- Upcoming upgrade of LHC provides challenging environment for tracking
- Many simultaneous interactions (high pile-up)
- Track reconstruction is computationally intensive

LHC

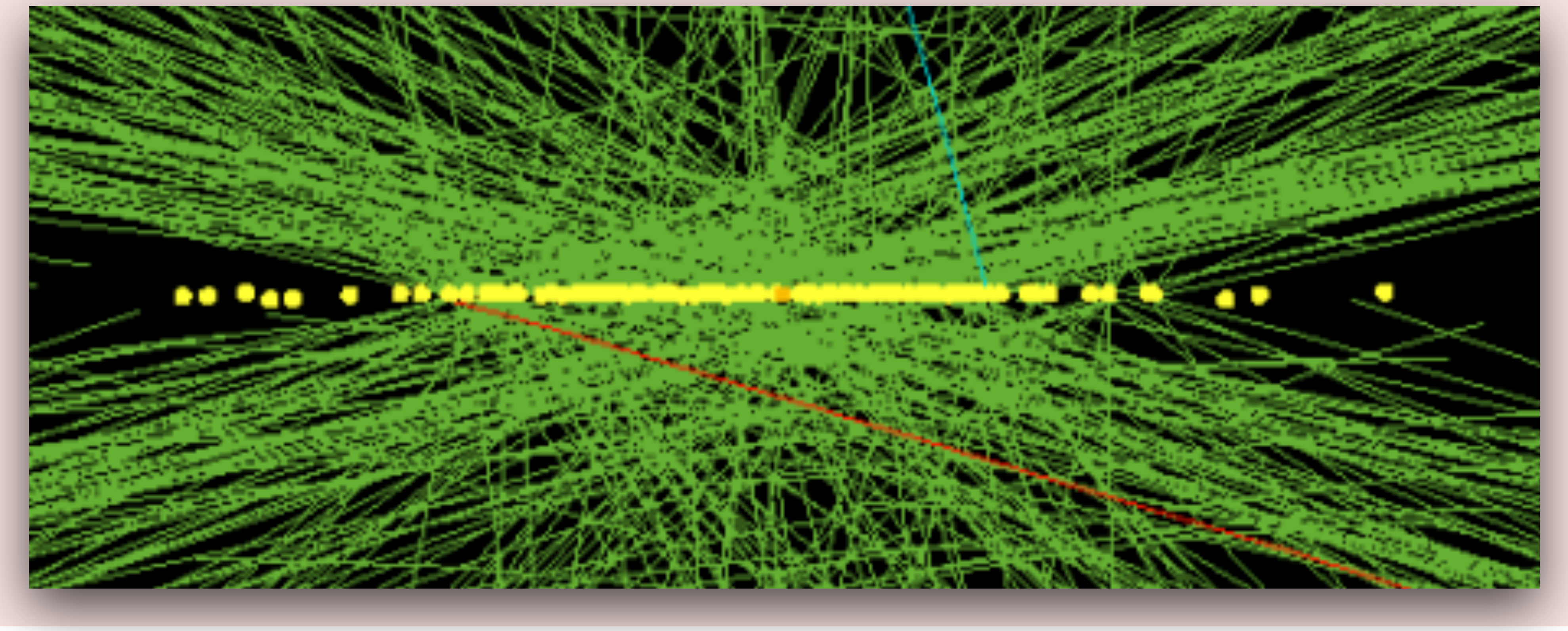


Luminosity: $2 \times$ LHC

Pile-up: ~ 60

NOW

HL-LHC



Luminosity: 5-7.5 $\times$ LHC

Pile-up: ~ 200

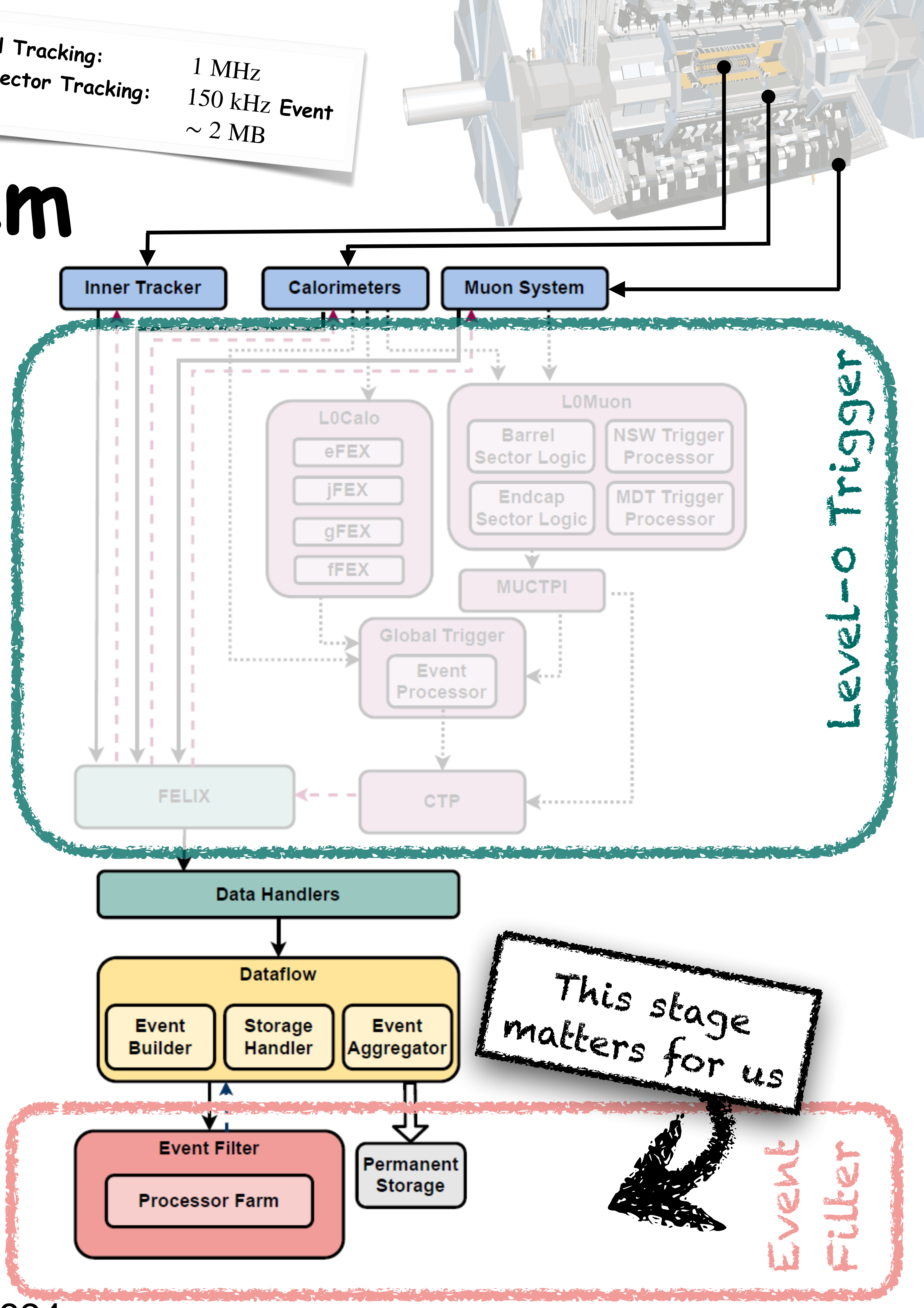
FUTURE

The ATLAS Trigger & Data AcQuisition (TDAQ) System

Regional Tracking: 1 MHz
Full Detector Tracking: 150 kHz Event
Size: ~ 2 MB

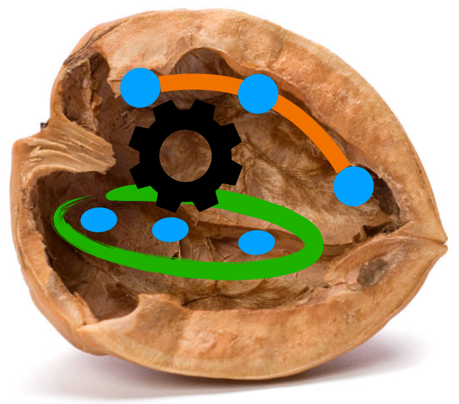
- 2 Level Trigger System Level 0 and Event Filter to select interesting events
- Event Filter will be based on Heterogeneous computing farm
- Current ongoing competition of ideas for future tracking concepts

(General) Triplet Track Fit

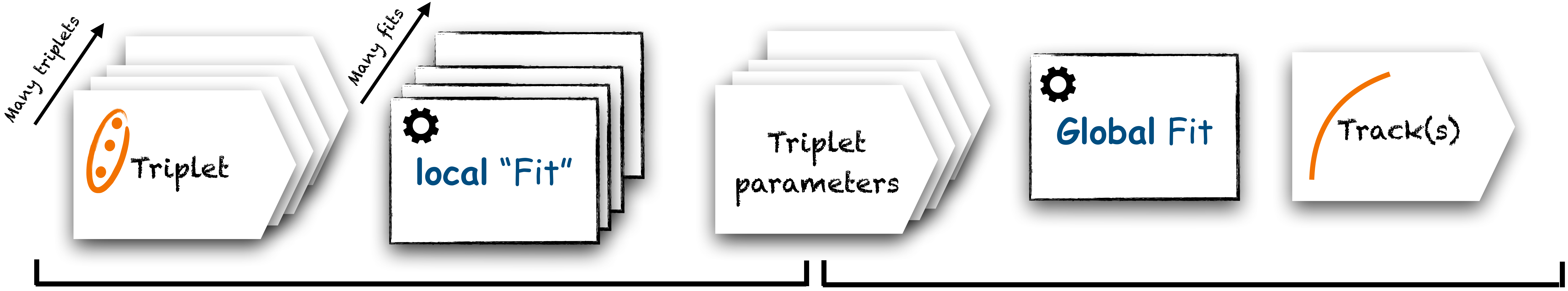


This stage matters for us

General Triplet Track Fit – In a Nutshell!



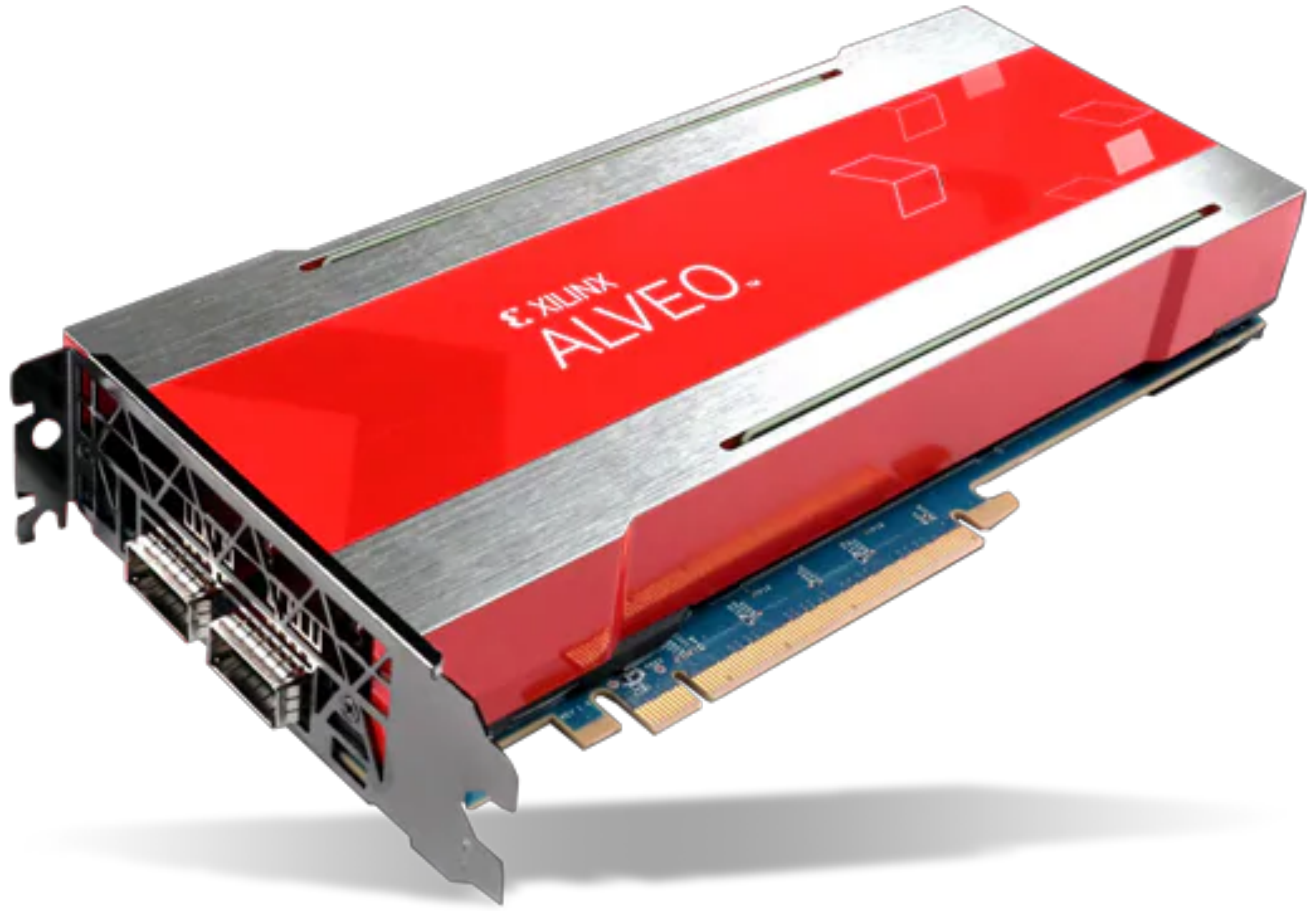
Triplet track fit based on two-step procedure that can be factorized:



① Accounts for all detector-specific information, e.g., B field, material budget; **highly parallelizable** (GPUs, FPGAs!)

② Completely independent of detector; just gets triplet parameters and calculates tracks

Nice feature: each triplet comes with a χ^2_k that can be used for filtering



FPGA Implementation

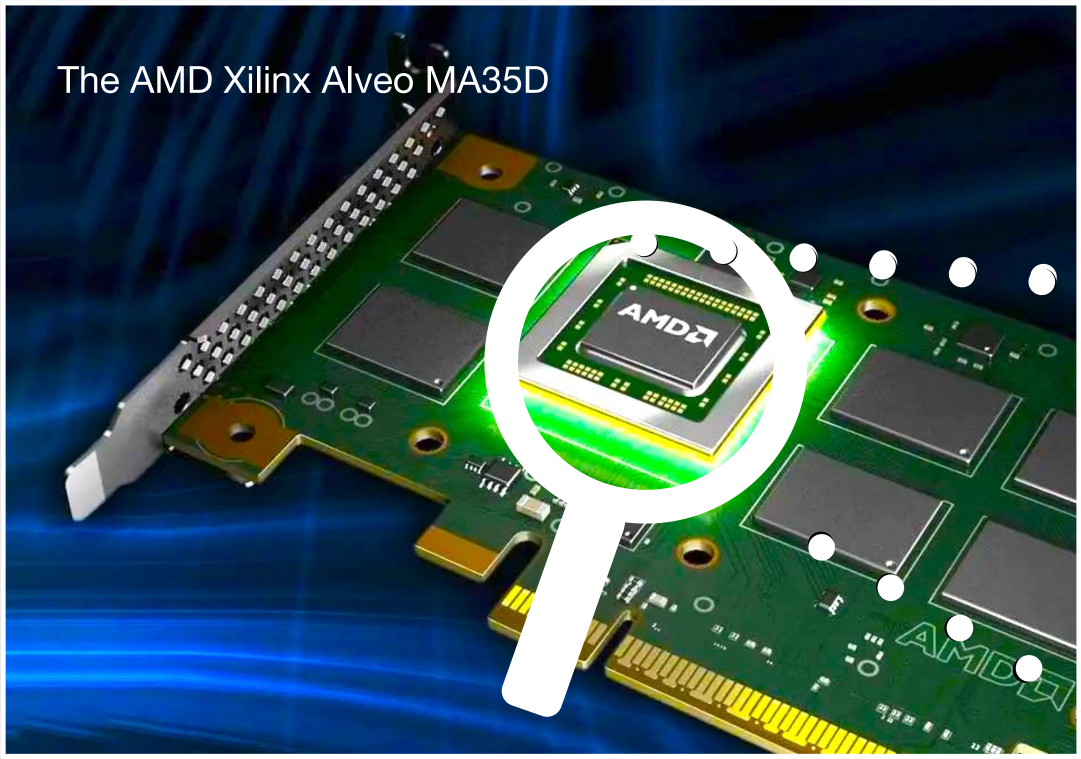
Field Programmable Gate Arrays (FPGAs)

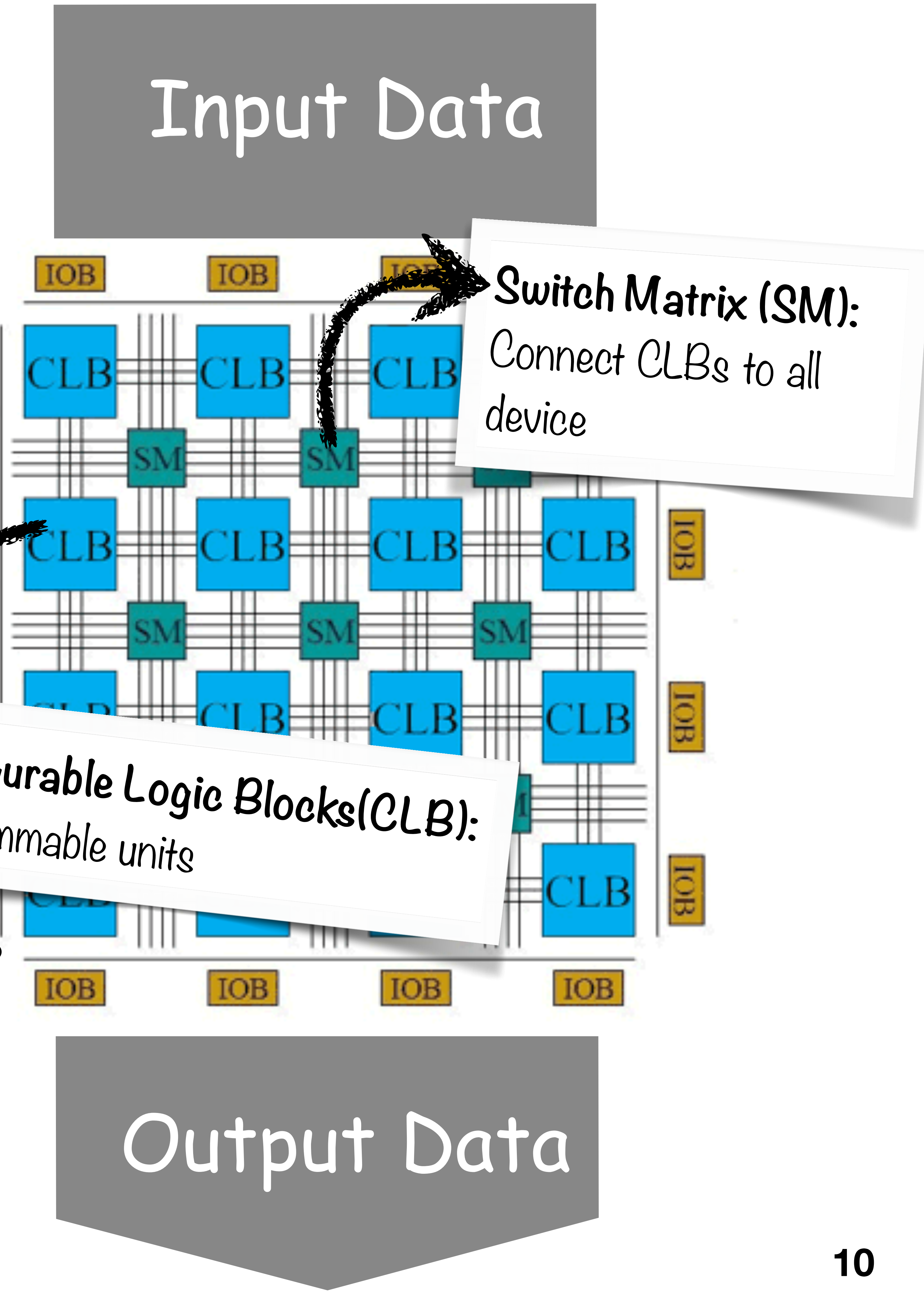
HDL — Hardware Developing Language

Programmable chip (reconfigurable hardware) enabling custom digital circuit creation for diverse applications

Why use FPGAs?

- High-degree of parallelism
- High throughputs
- Low latency
- Power efficiency
- Reconfigurability


The AMD Xilinx Alveo MA35D



Input / Output Block (IOB): Communicates with external devices

Switch Matrix (SM): Connect CLBs to all device

Configurable Logic Blocks(CLB): Programmable units

Why NOT use FPGAs?

Among others, HDLs (e.g. Verilog) are “not” widely spread.

But: Many people know C++: Would be nice to <u>synthesize</u> (“translate”) C++ code to FPGA

→ **High Level Synthesis (HLS)**

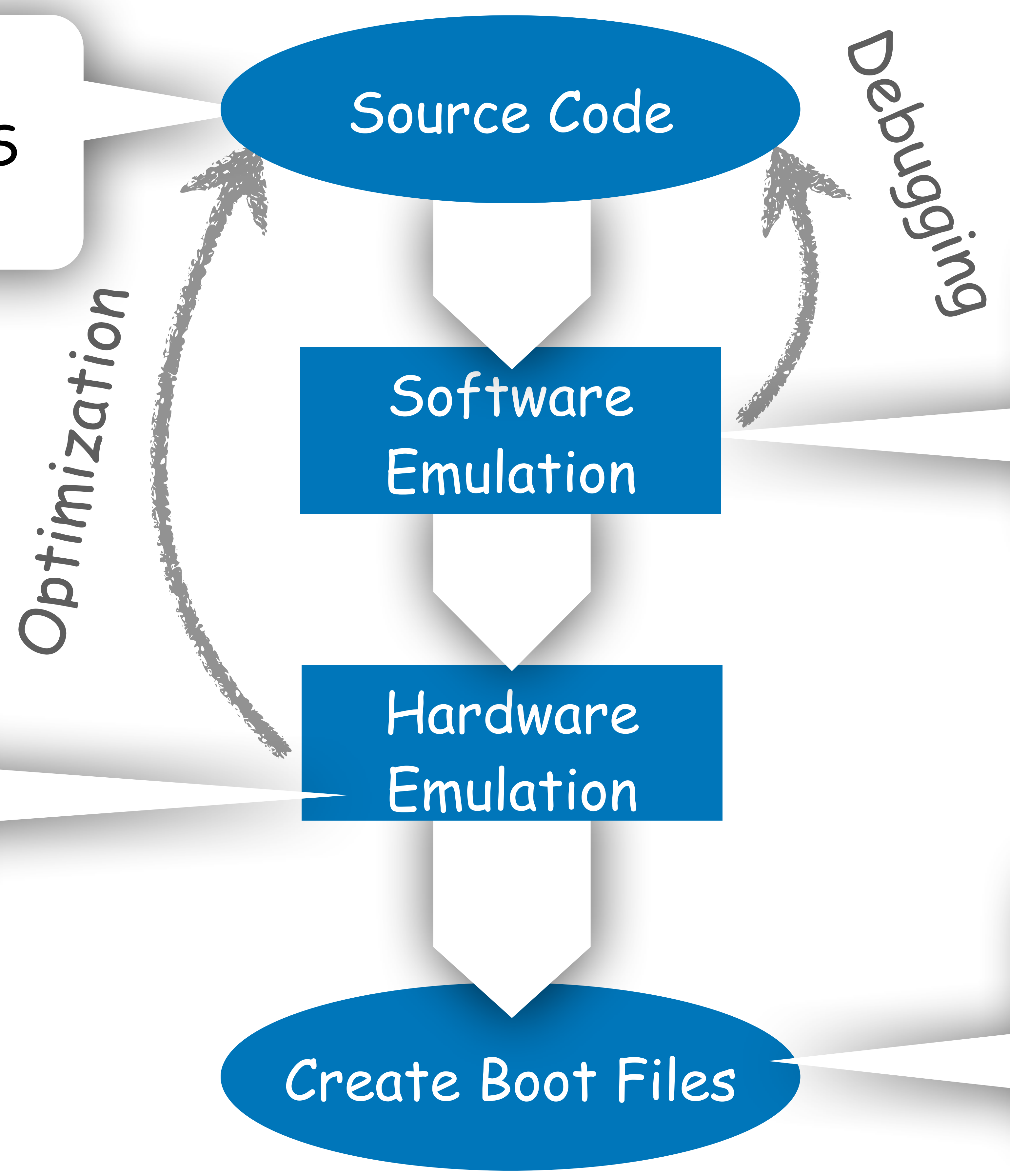
HDL — Hardware Developing Language

What is High Level Synthesis?

- From this C++ to this HDL (Verilog) to run on this .

In other words: HLS translates high-level languages (C, C++) to FPGA

This is C/C++ code with some compiler statements to tell HLS what to do



Source Code

Software Emulation

Hardware Emulation

Create Boot Files

Optimization

Debugging

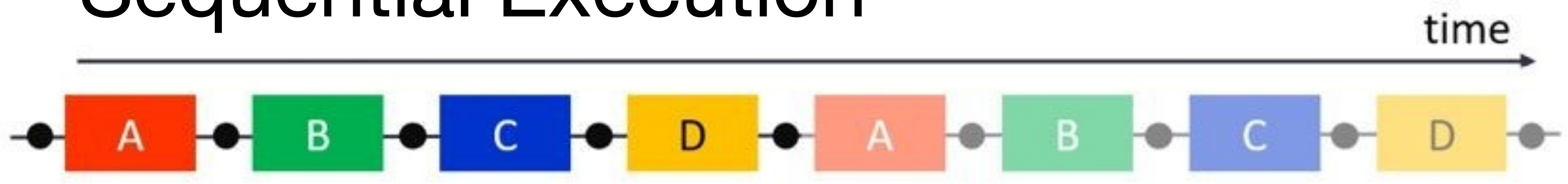
This is just for debugging the C/C++ code using HLS libraries. Compiling: **FAST ~ O(seconds)**

This is for emulating the synthesised code on target hardware. Compiling: **NOT SO FAST ~ O(minutes)**

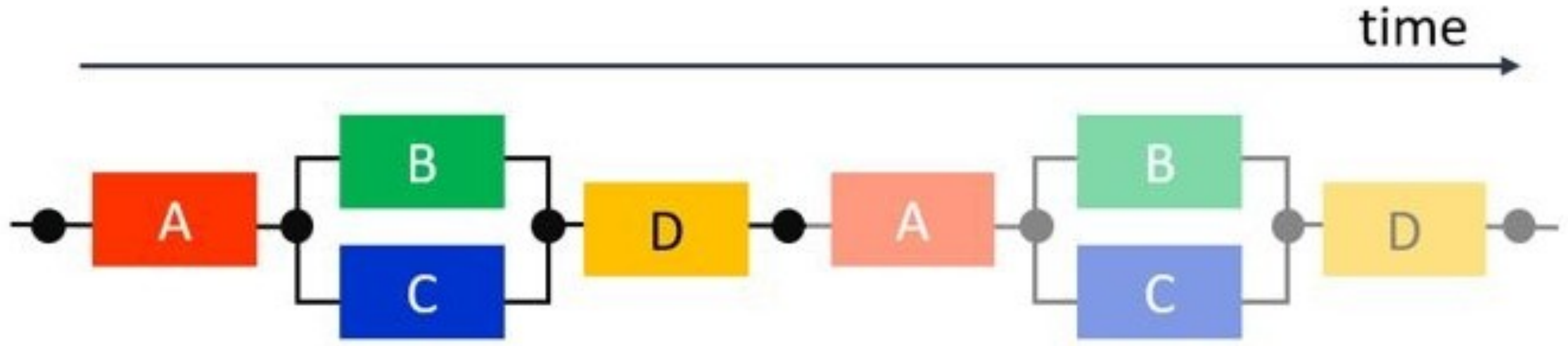
If everything works, build for actual hardware. Compiling: **VERY SLOW ~ O(hours)**

Optimization Example

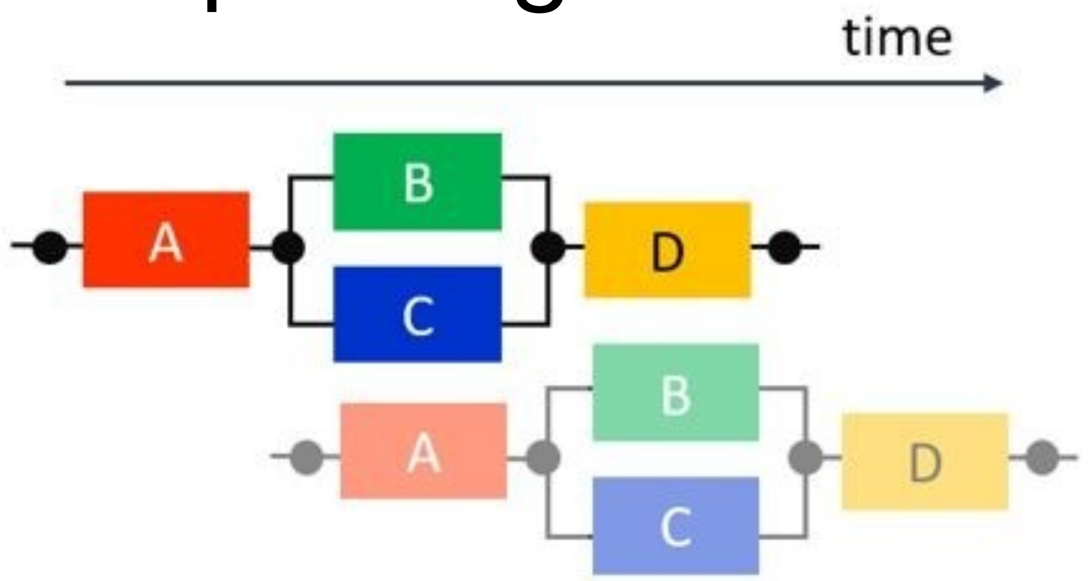
Sequential Execution



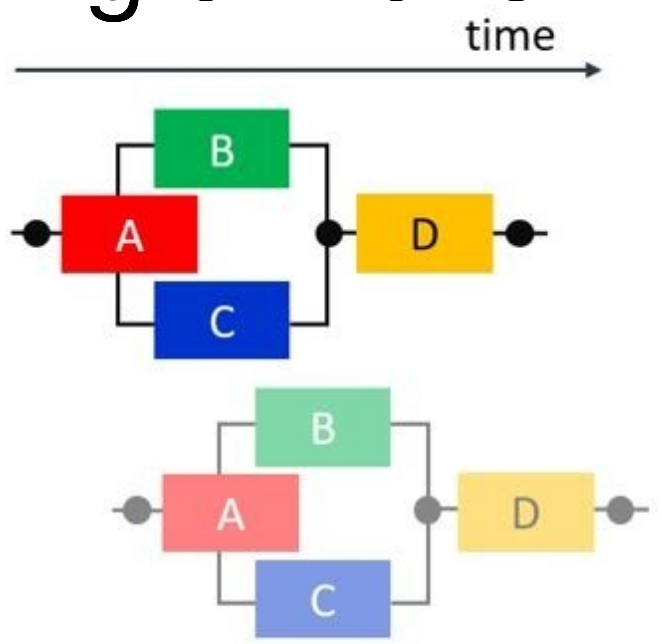
Task Parallelism within a Run



Task Parallelism with Pipelining



Task Parallelism and Pipelining of Runs



A → provides data

B
C } Independent Calculations

D → output

● → synchronization

Some performance results for vector multiplication

Base design throughput 1.2 Gb/s

Pipelined throughput 4.7 Gb/s

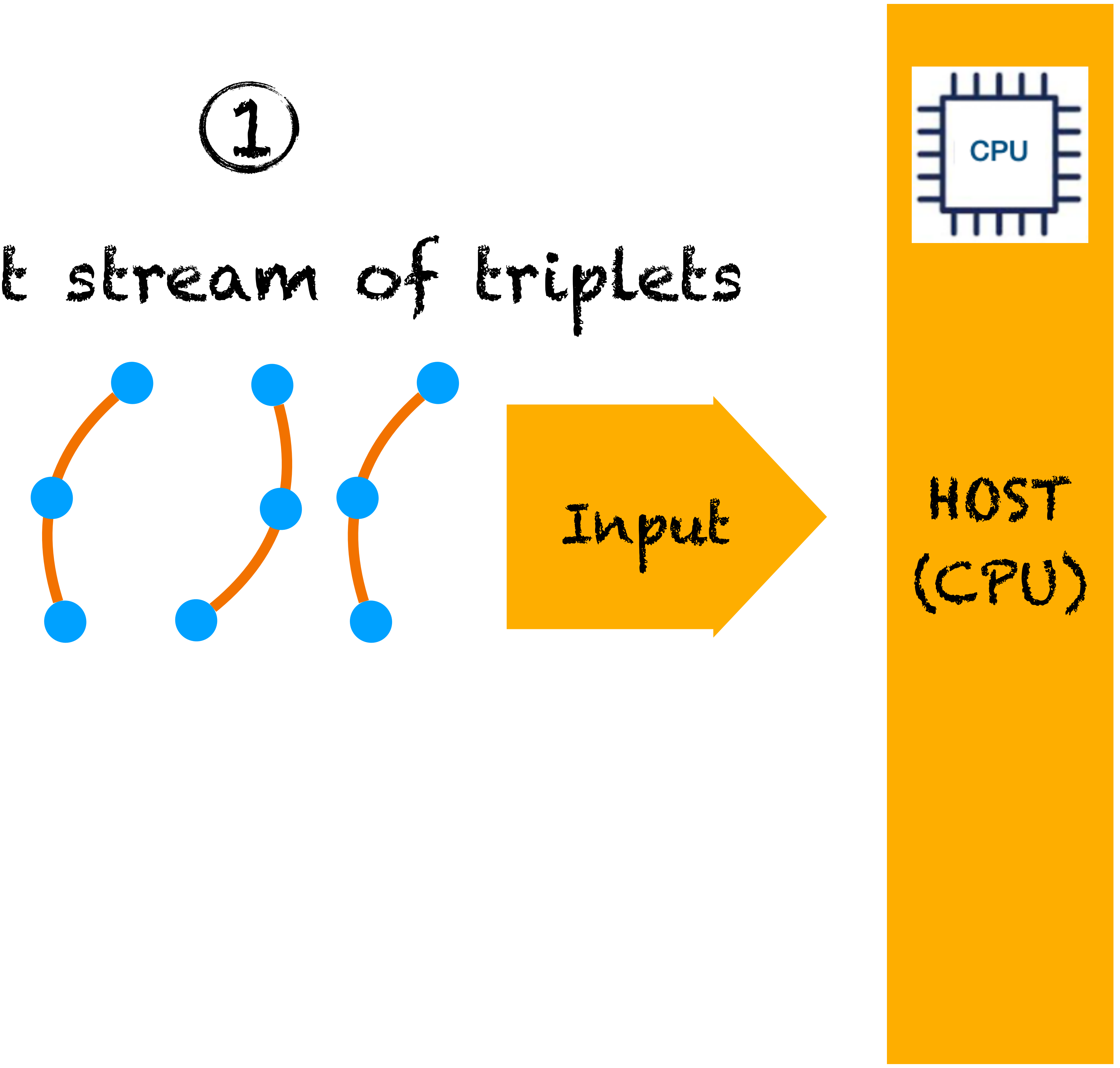
Unrolled throughput 120 Gb/s

Fully pipelined throughput 960 Gb/s

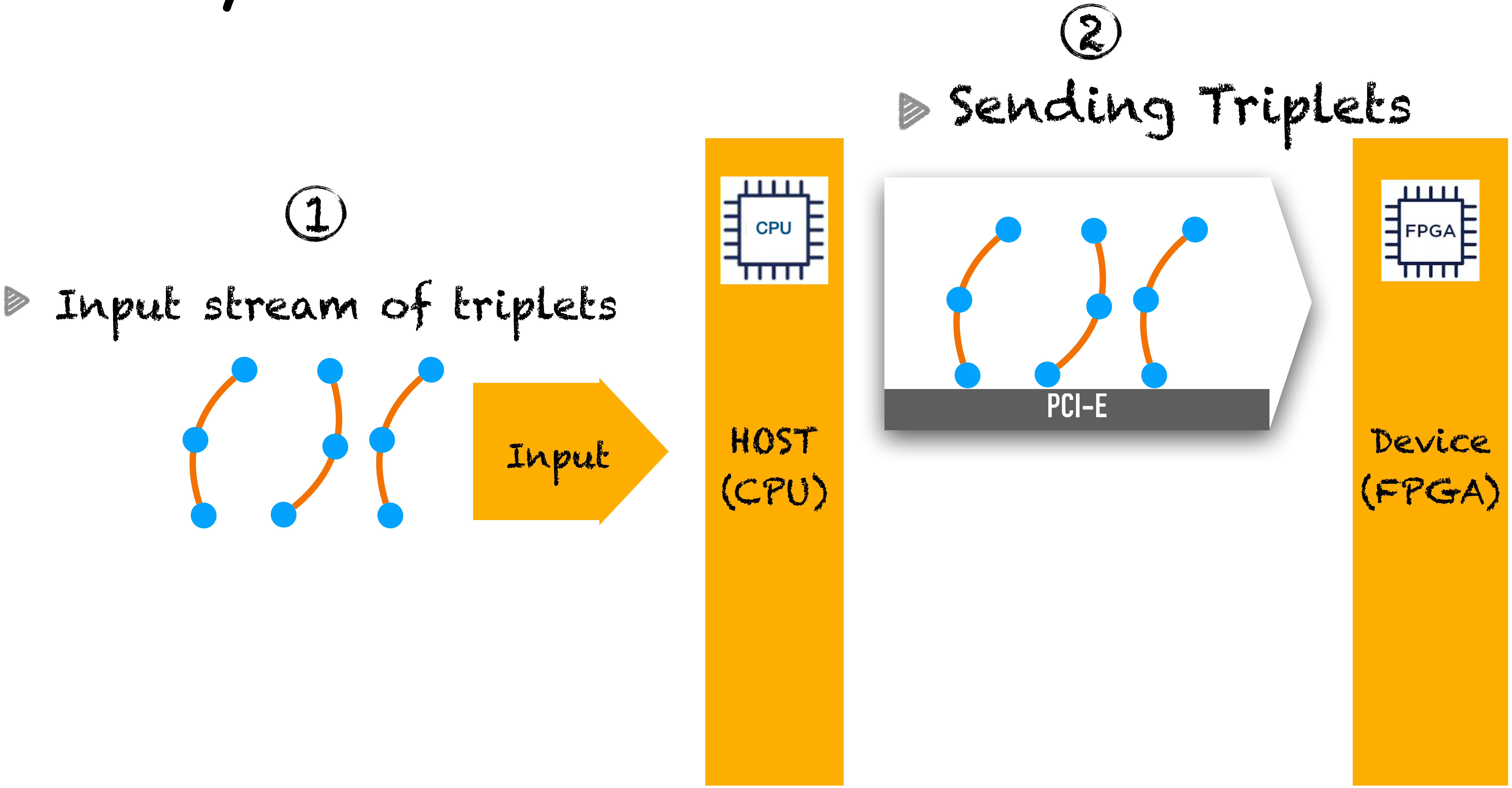
Implementation of Triplet Track Fit
– Only Local Part so far

①

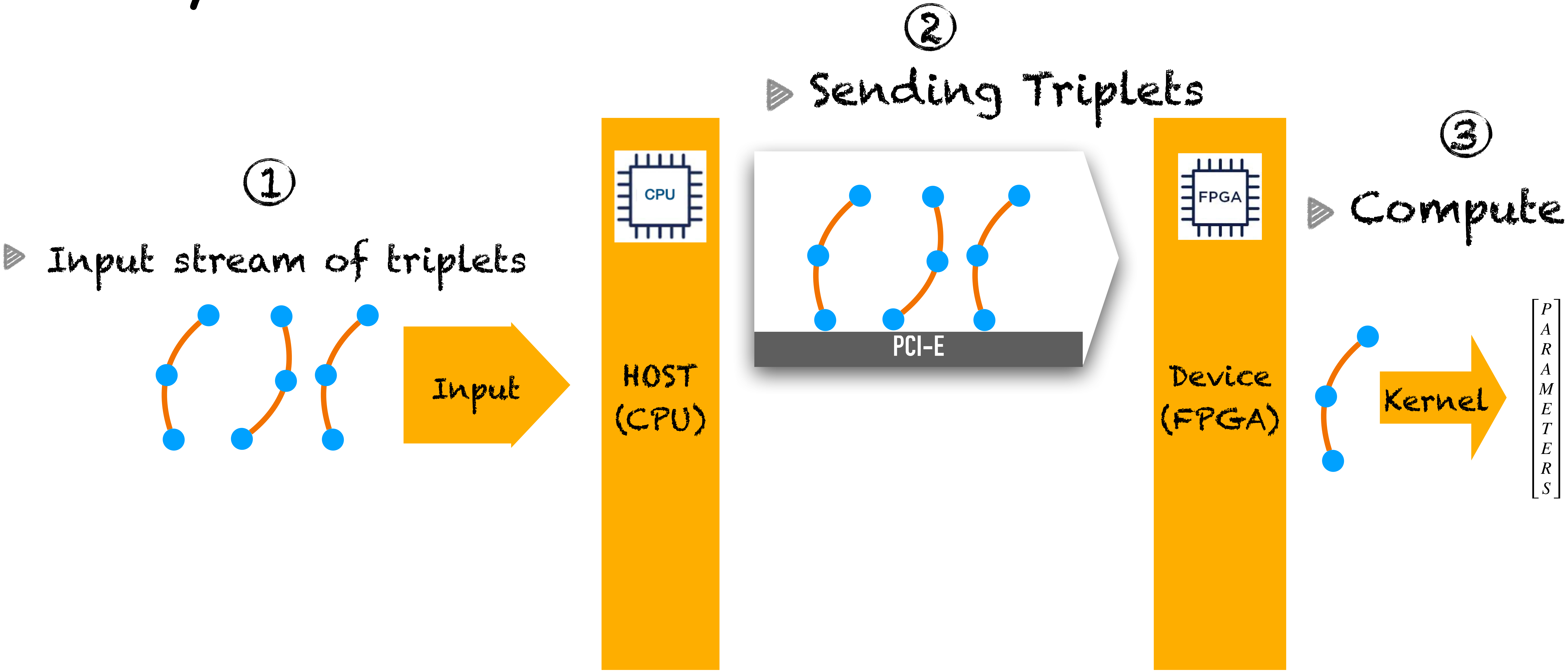
- Input stream of triplets



Implementation of Triplet Track Fit
- Only Local Part so far



Implementation of Triplet Track Fit
- Only Local Part so far



①

- Input stream of triplets

②

- Sending Triplets

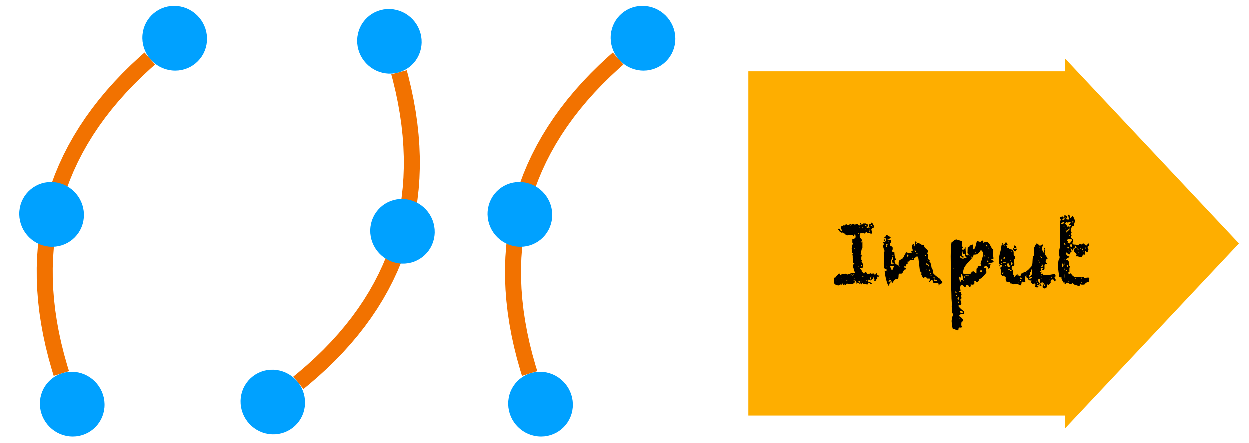
③

- Compute

Implementation of Triplet Track Fit
- Only Local Part so far

①

- Input stream of triplets

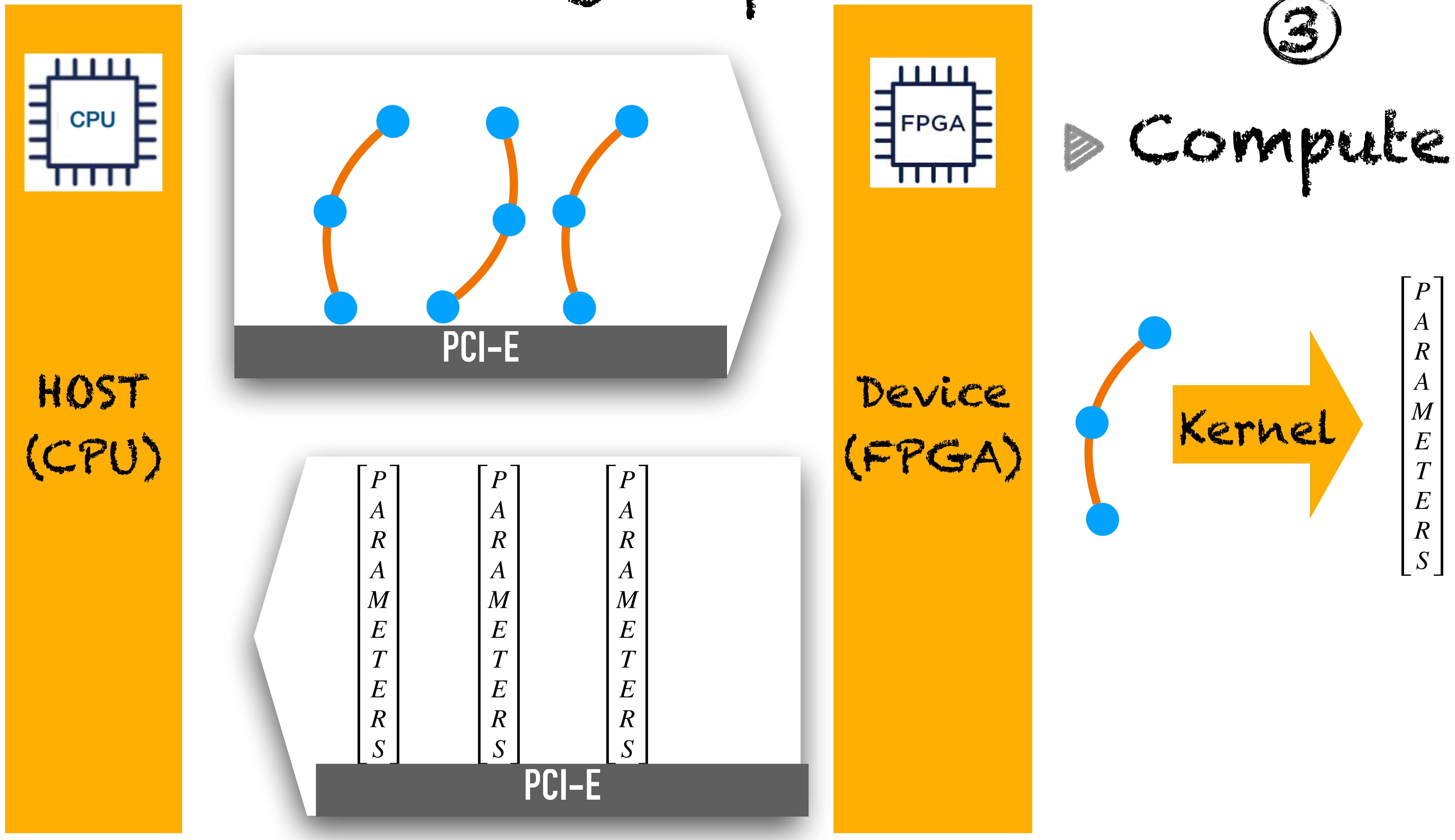


②

- Sending Triplets

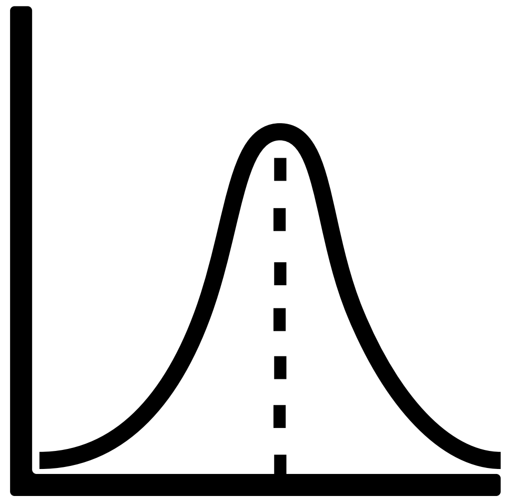
③

- Compute



- Collecting Triplet Parameters

④

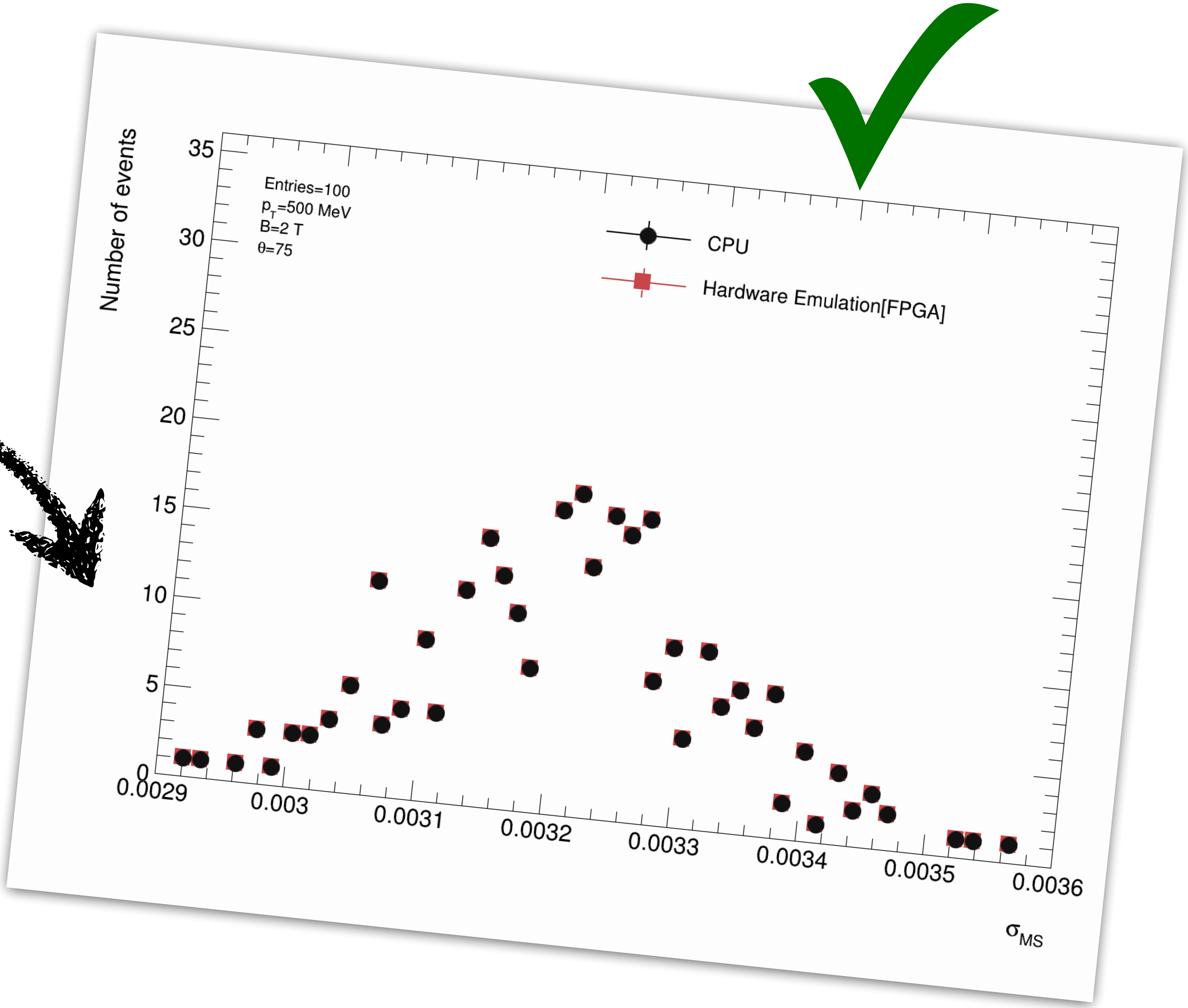


First Results

Consistency Check

Triplet Track Fit is already implemented for CPU

Better make sure that we get the same results!



Local Fit implemented for FPGA!

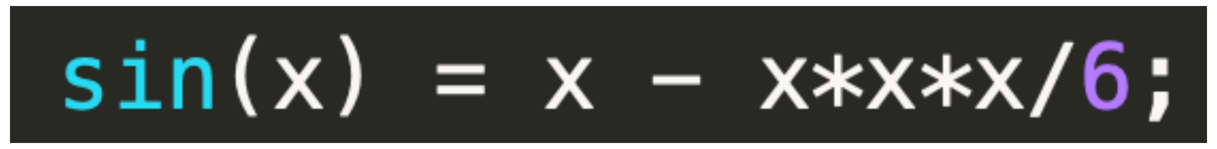
All triplet parameters are consistent with CPU results

Next: Optimization!

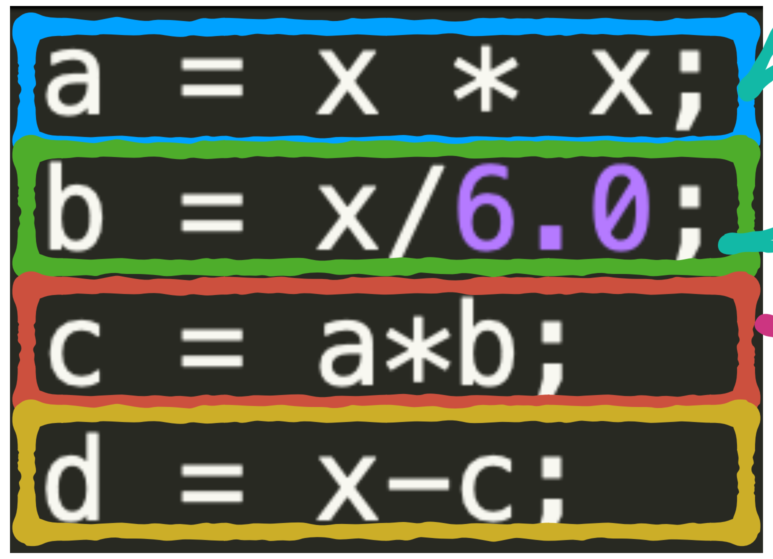
Optimization

- Approximate trigonometric functions with small values.
- Calculate other functions by Cordic.
- Break down the algorithm into intermediate steps.
- Avoid redundancies.

Example:



```
sin(x) = x - x*x*x/6;
```



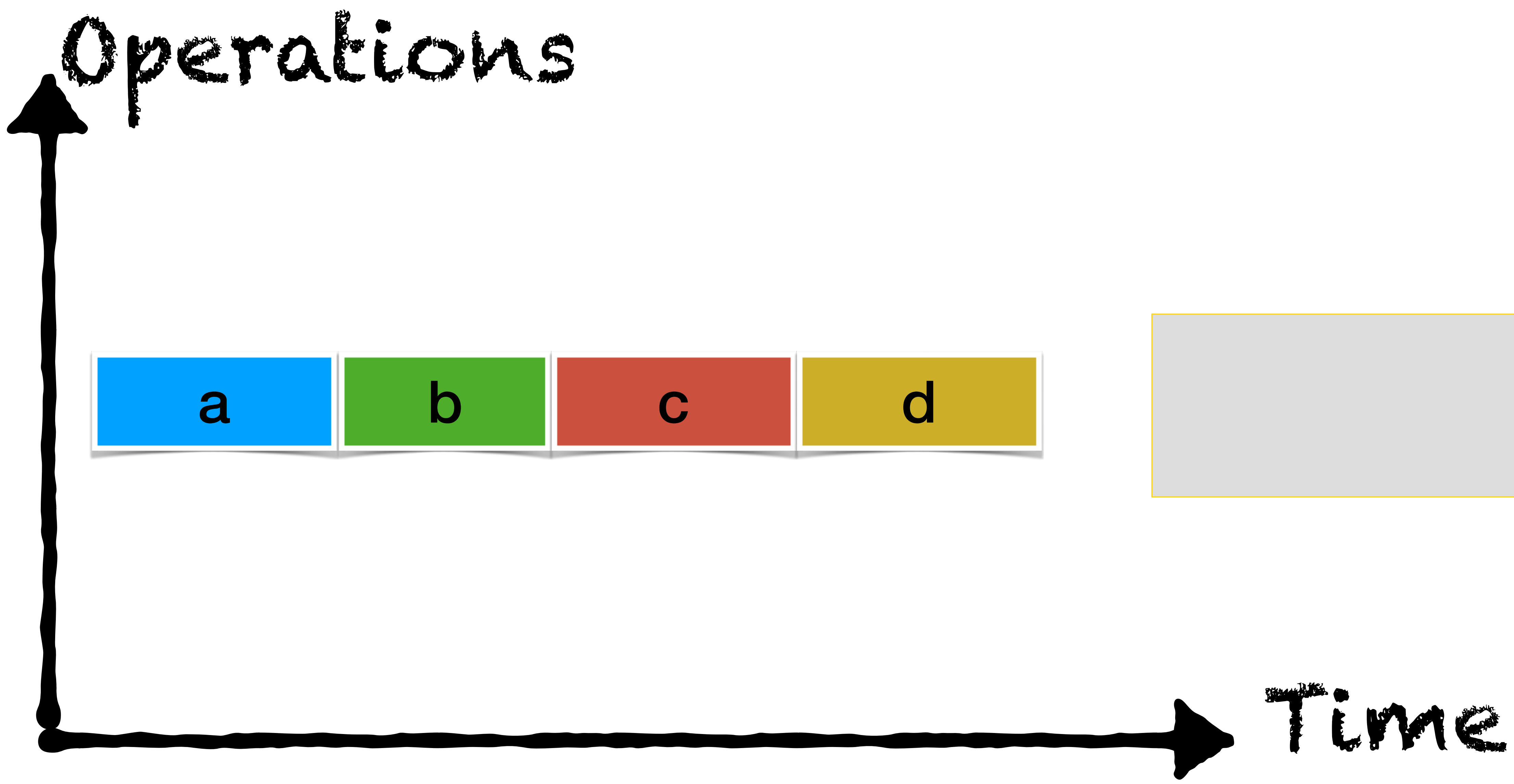
```
a = x * x;
b = x/6.0;
c = a*b;
d = x-c;
```

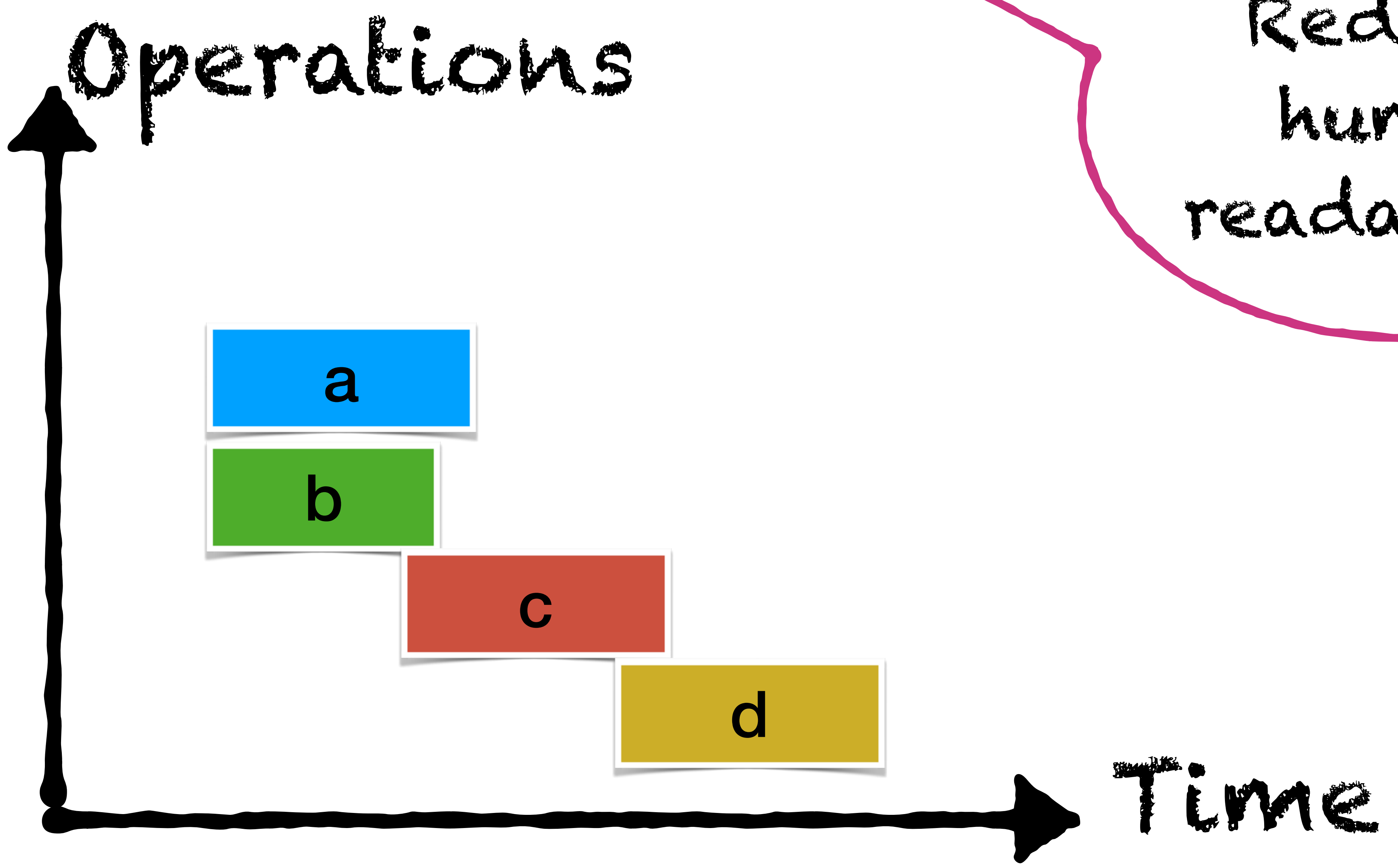
"a" and "b" independent!

TASK LEVEL PARALLELISM

Helps avoid duplication.

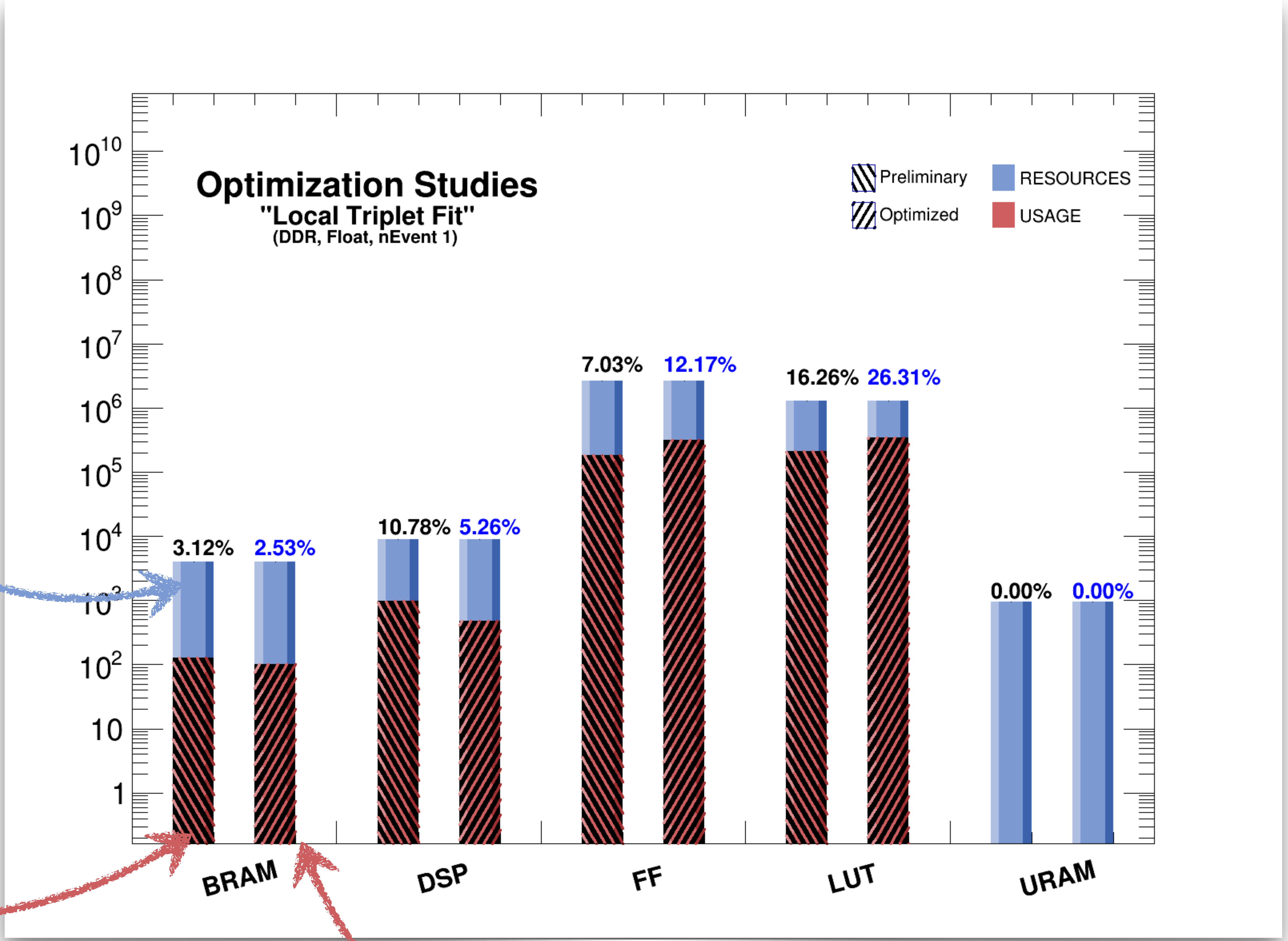
Reduces human readability.





Resource Usage

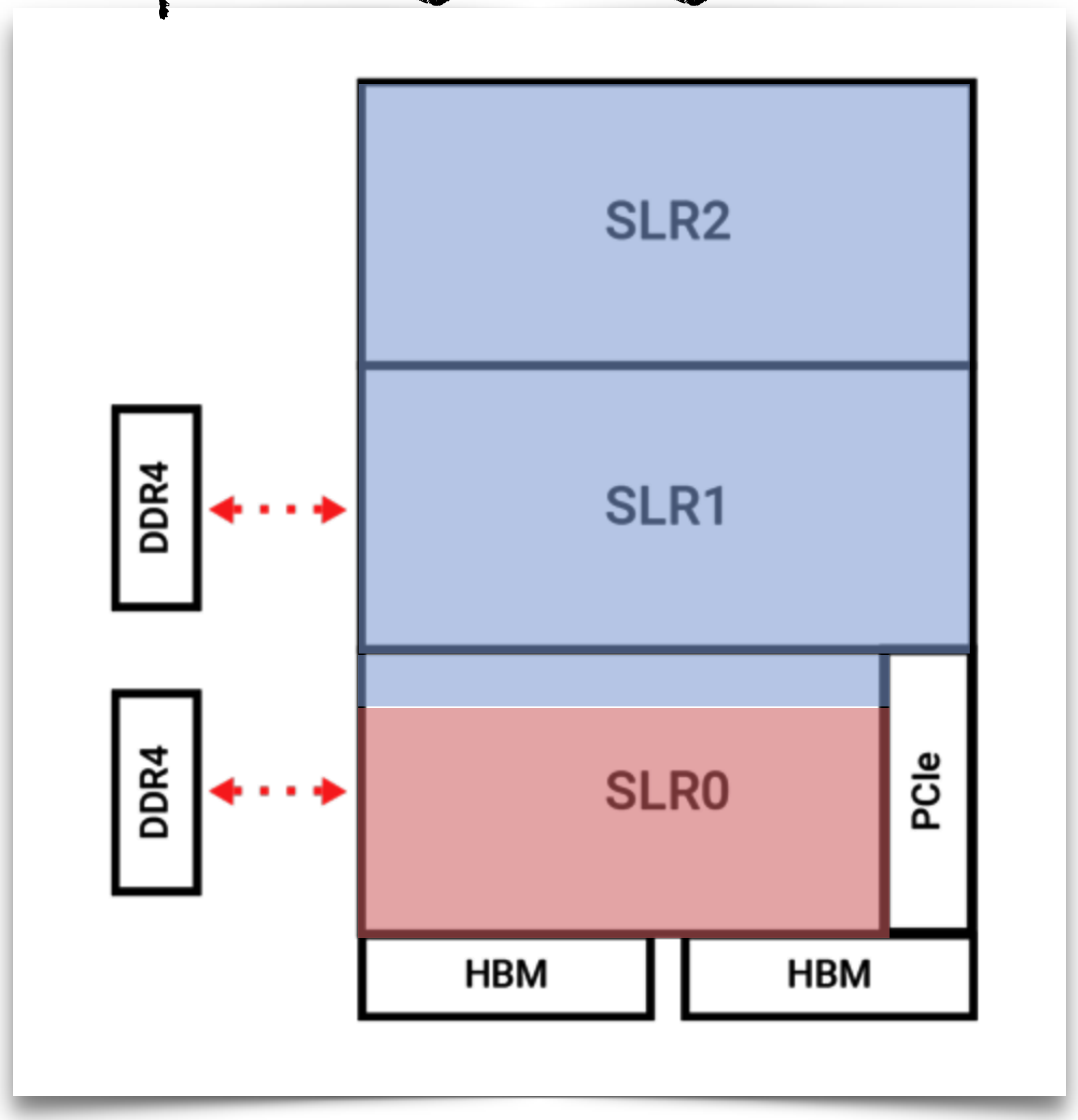
Available limited resources on the FPGA

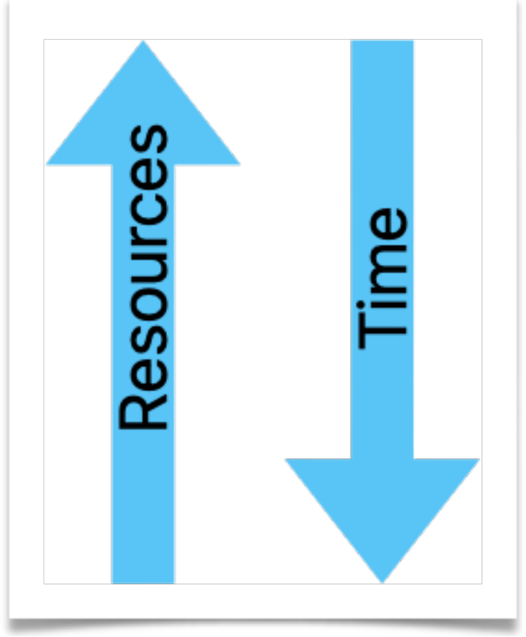


\\ Resource usage without any optimization

// Resource usage with optimization

Super Logic Region(SLR)



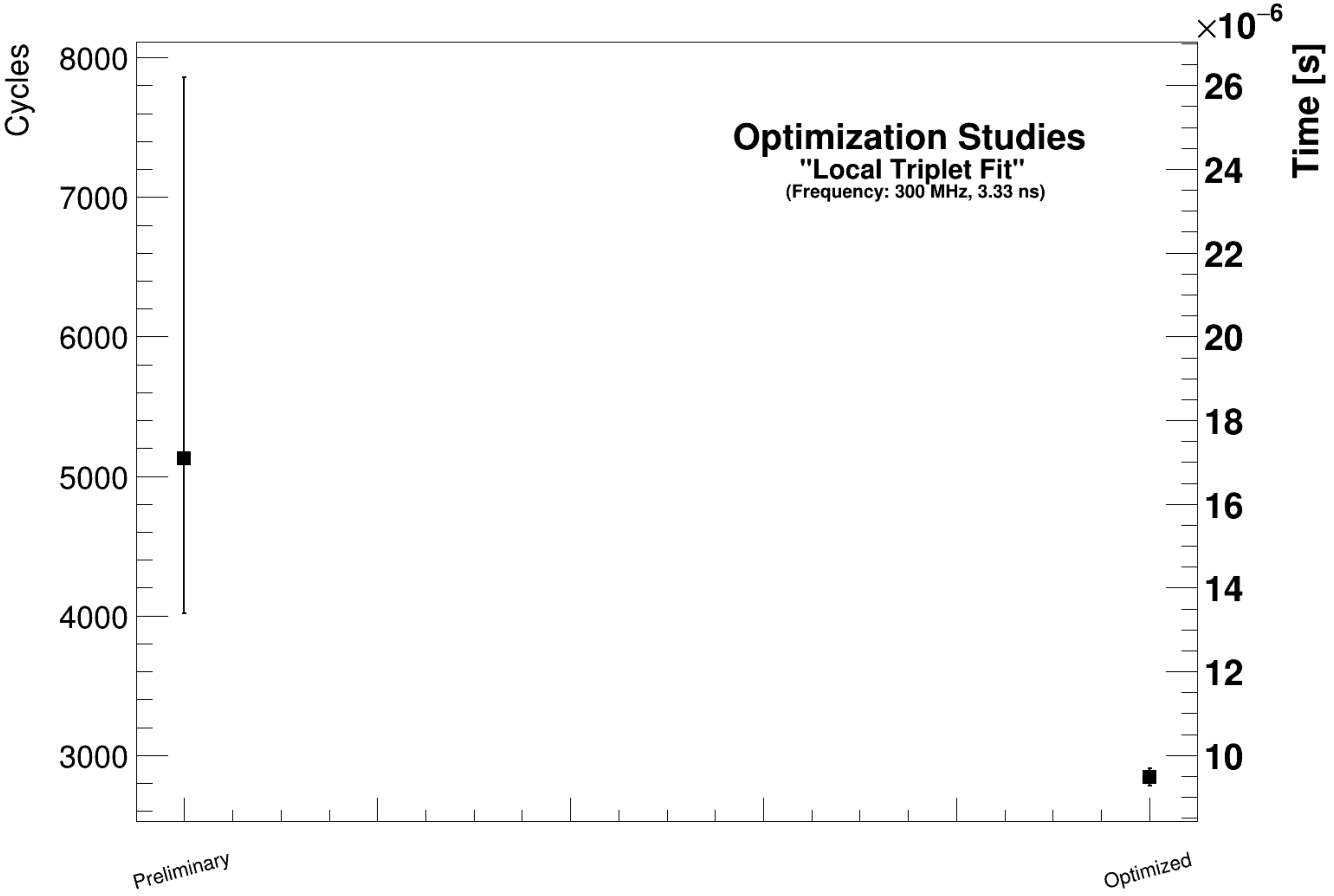


Resources vs
Latency trade-off

Kernel Latency

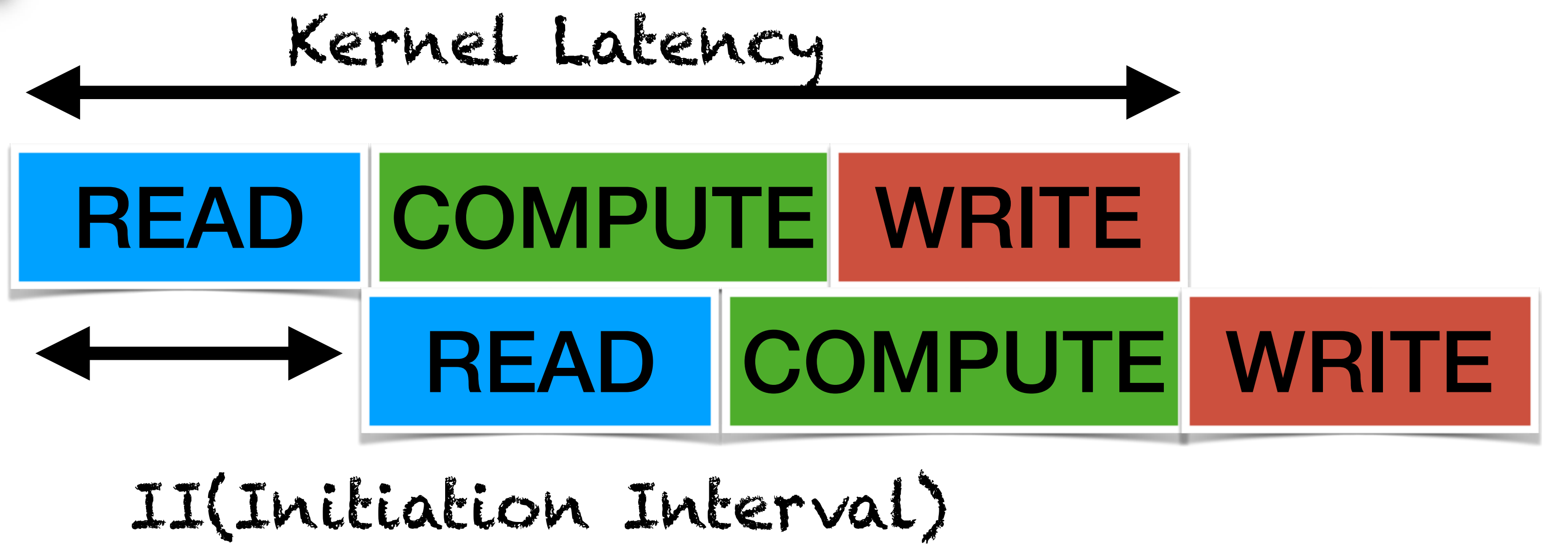
Using more resources to reduce latentcy!

Estimated number of cycles required to execute kernel.



Latency without any optimization

Latency with optimizations.



Kernel Latency

READ COMPUTE WRITE

READ COMPUTE WRITE

II(Initiation Interval)

Reducing latency improves the throughput!

Summary

- Ongoing studies to implement Triplet Track Fit on FPGAs for the ATLAS Event Filter
- Hardware emulation of the local triplet fit produces consistent results
- Preliminary optimization improves latency and resource usage
- Next Steps:
    - □ Global Track Fit will be implemented
    - □ Running on Hardware

Thank you for your attention! Questions?

References

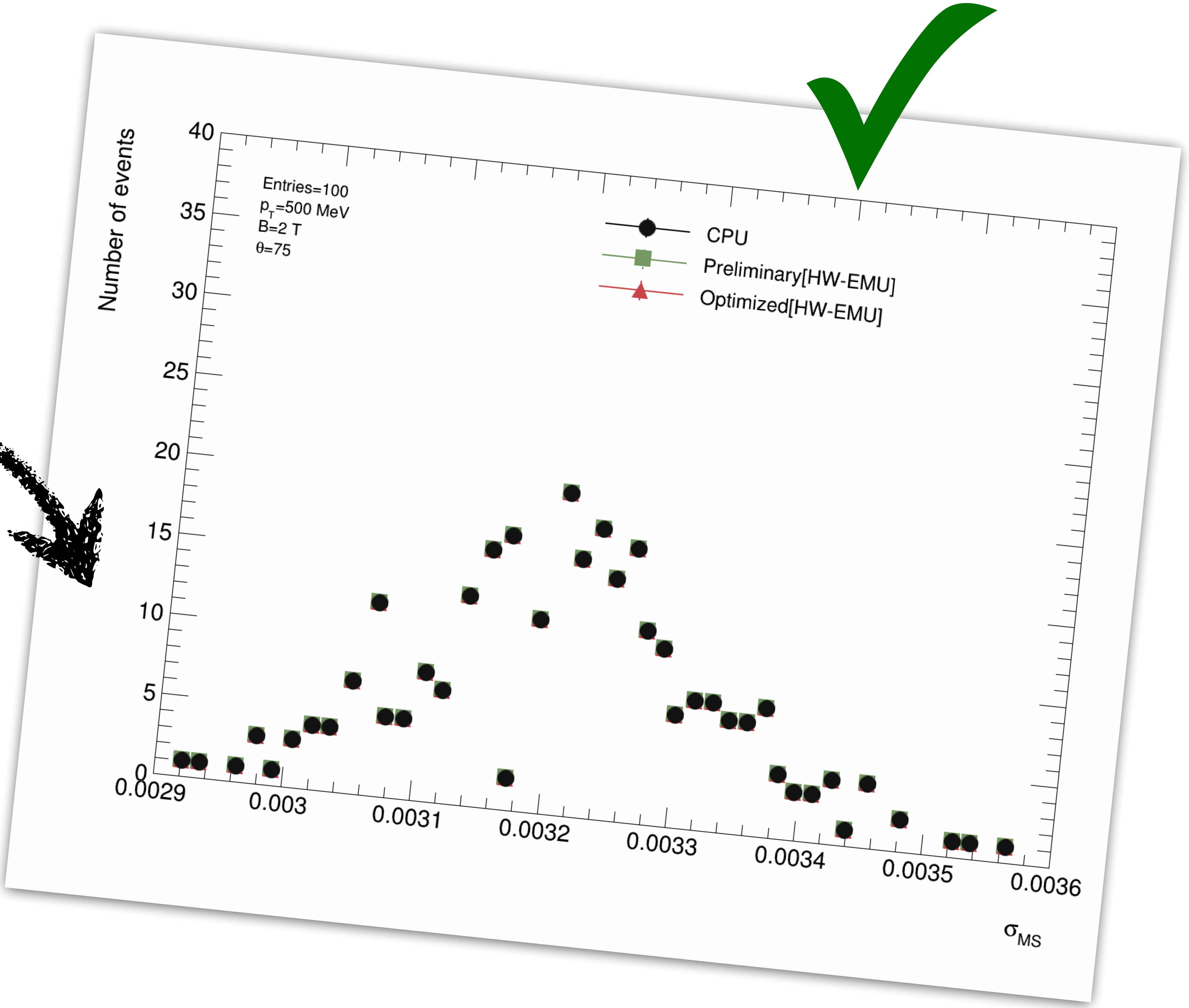
N. Berger, A. Kozlinskiy, M. Kiehn, A. Schöning, A new three-dimensional track fit with multiple scattering, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. 844 (2017) 135–140.

https://docs.amd.com/r/2023.1-English/ug1393-vitis-application-acceleration

Backup

After Optimization

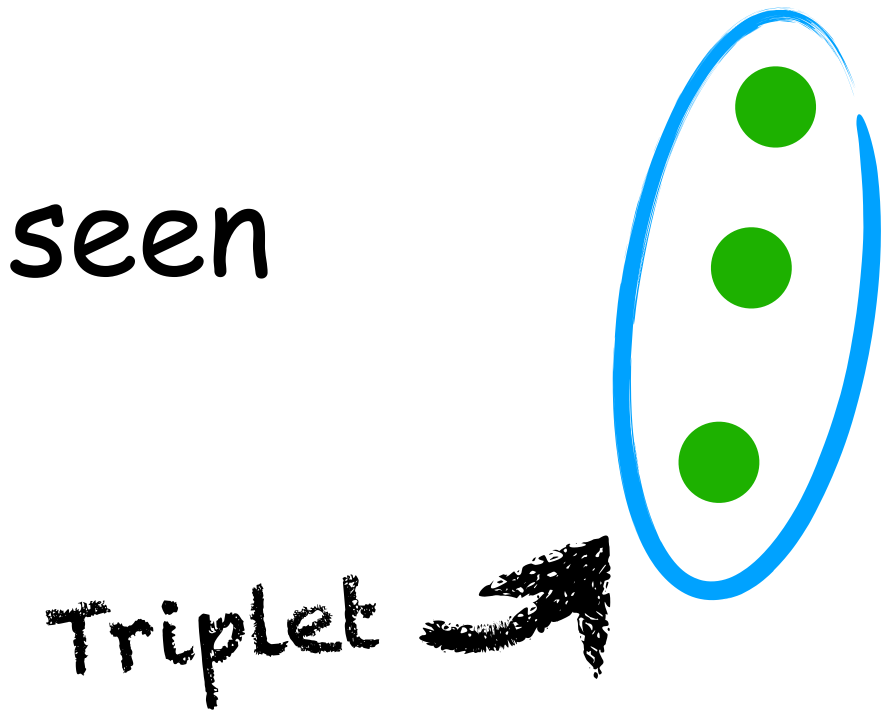
Better make sure that we get the same results after optimizations!



All triplet parameters are still consistent with CPU results.

What is a Triplet?

A Triplet is a collection of three consecutive hits and can be seen as the smallest building block of tracking.

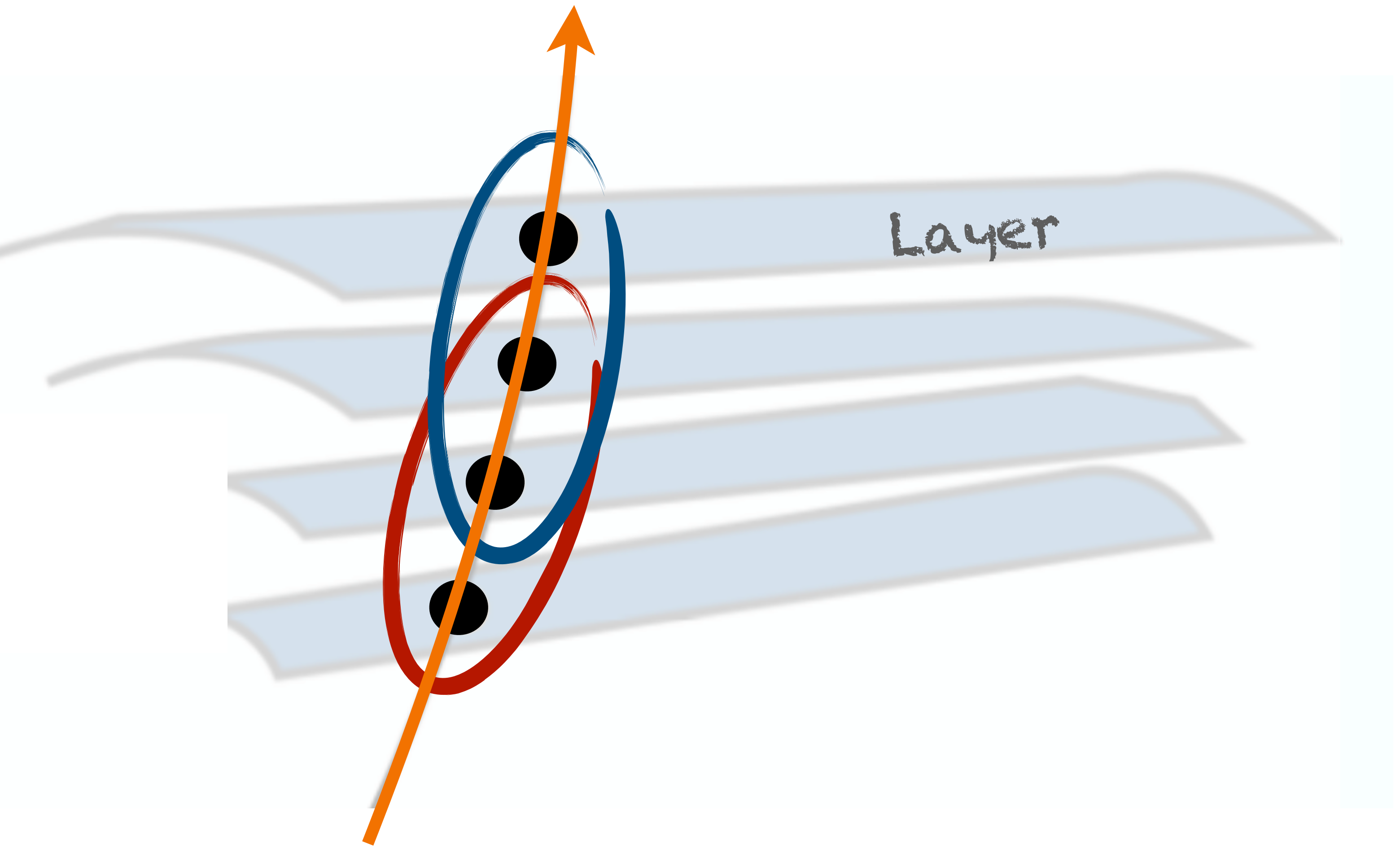


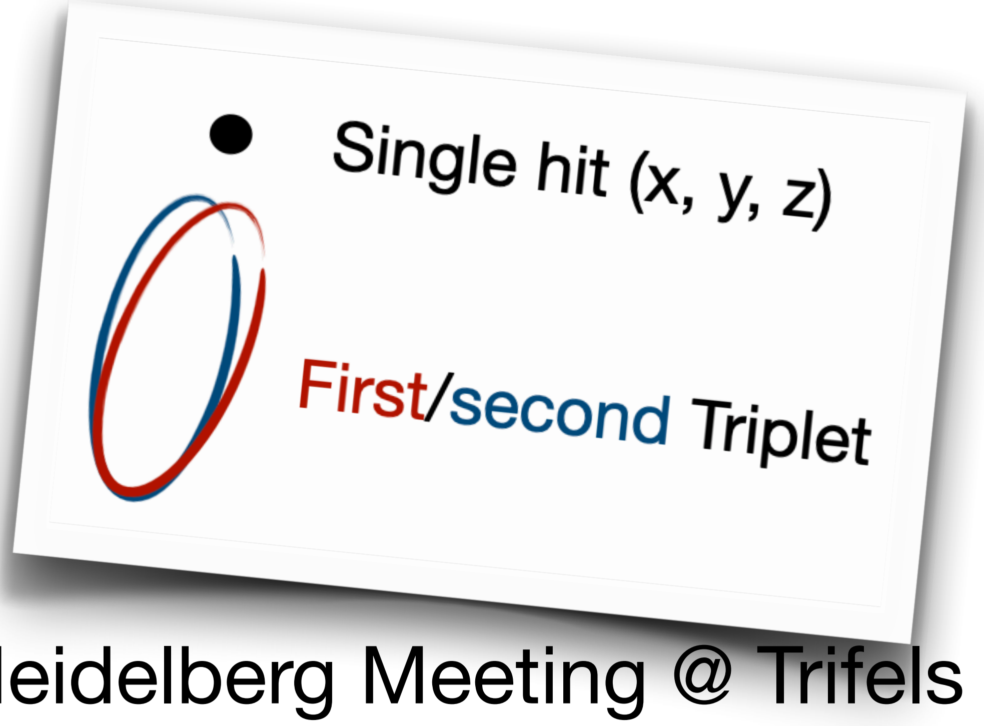
Why Triplets?

- Minimum required number of hits to calculate track parameters (e.g.curvature)
- Consecutive triplets share two hits.

Getting Tracks from Triplets

Tracks are combinations of triplets consisting of hits.





Info:

The dataset we're using consists of triplets