

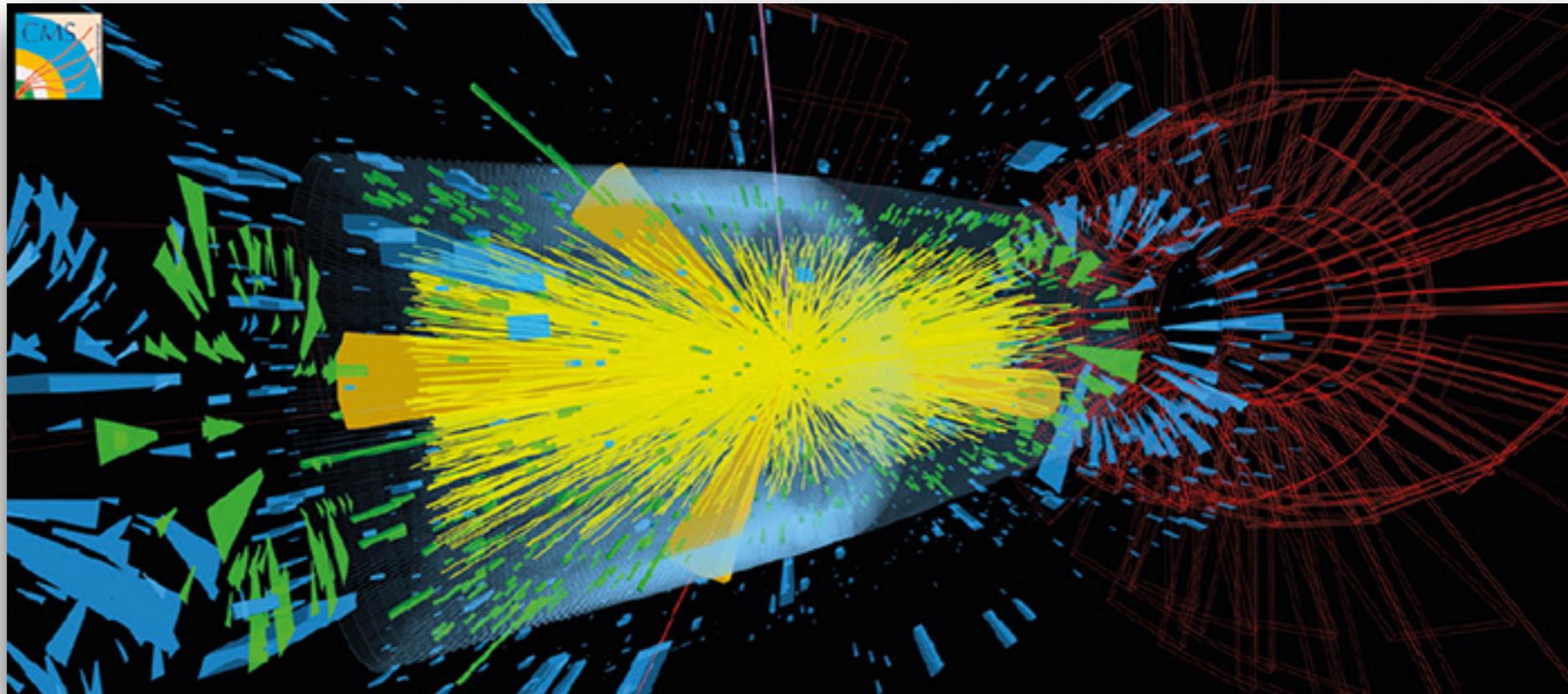
How to do real-time Deep Learning on FPGAs

Introduction

Universität Zürich - Physik Institut, 6 February 2019



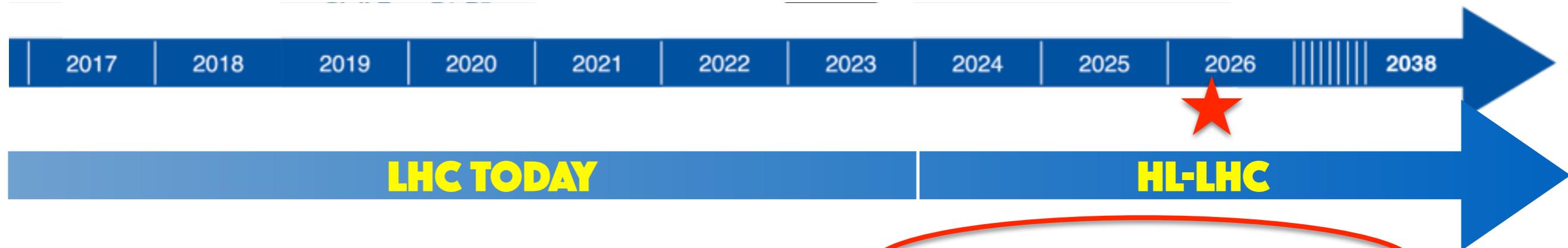
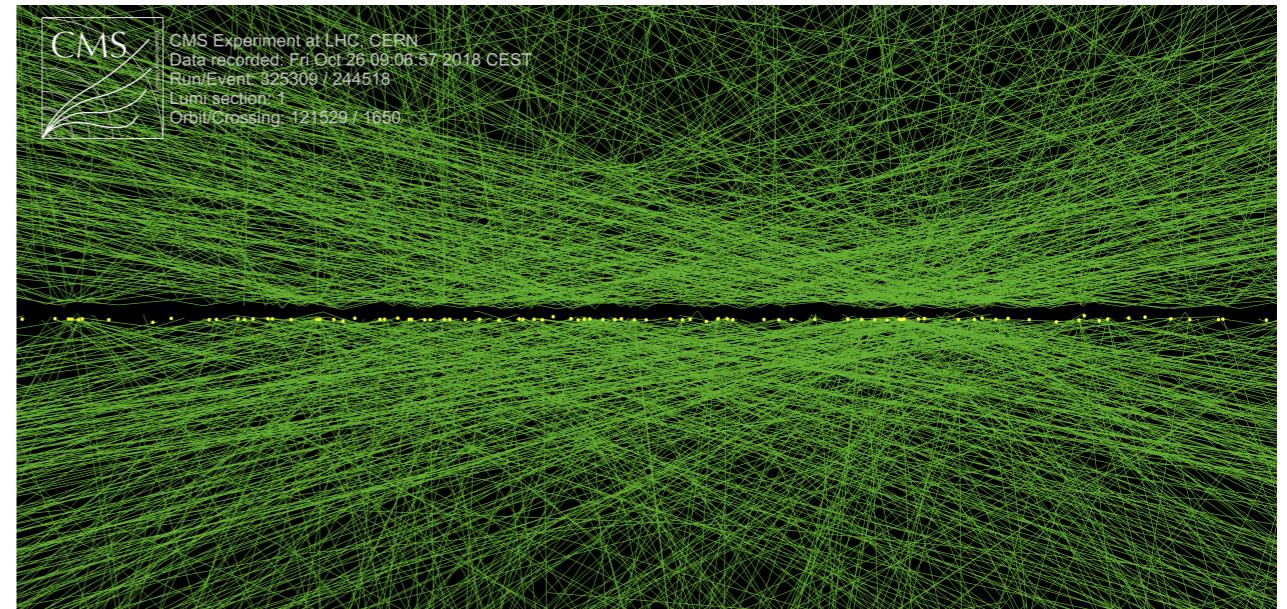
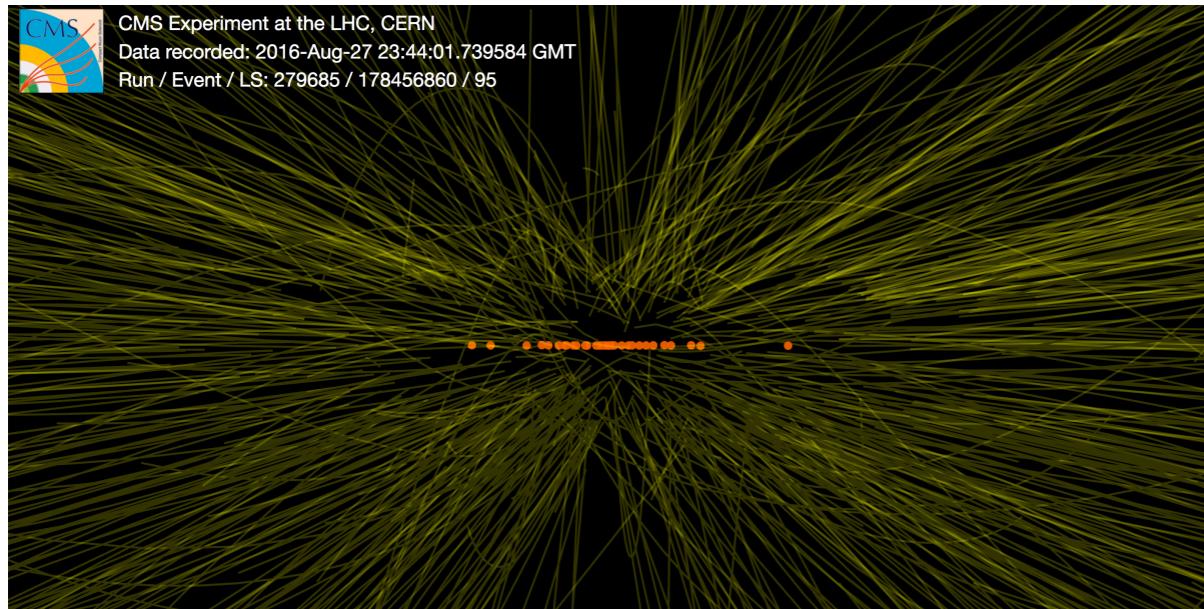
Universität
Zürich^{UZH}



Motivation: *use cases at the LHC and beyond*

Future challenges @ LHC

Extreme bunch crossing frequency of 40 MHz → extreme data rates O(100 TB/s)



- ▶ ~ 40 collisions/event
- ▶ ~ 10 sec/event processing time

- ▶ ~ 200 collisions/event
- ▶ more granular detector
- ▶ ~ minutes/event processing time
- ▶ flat budget for computing resources

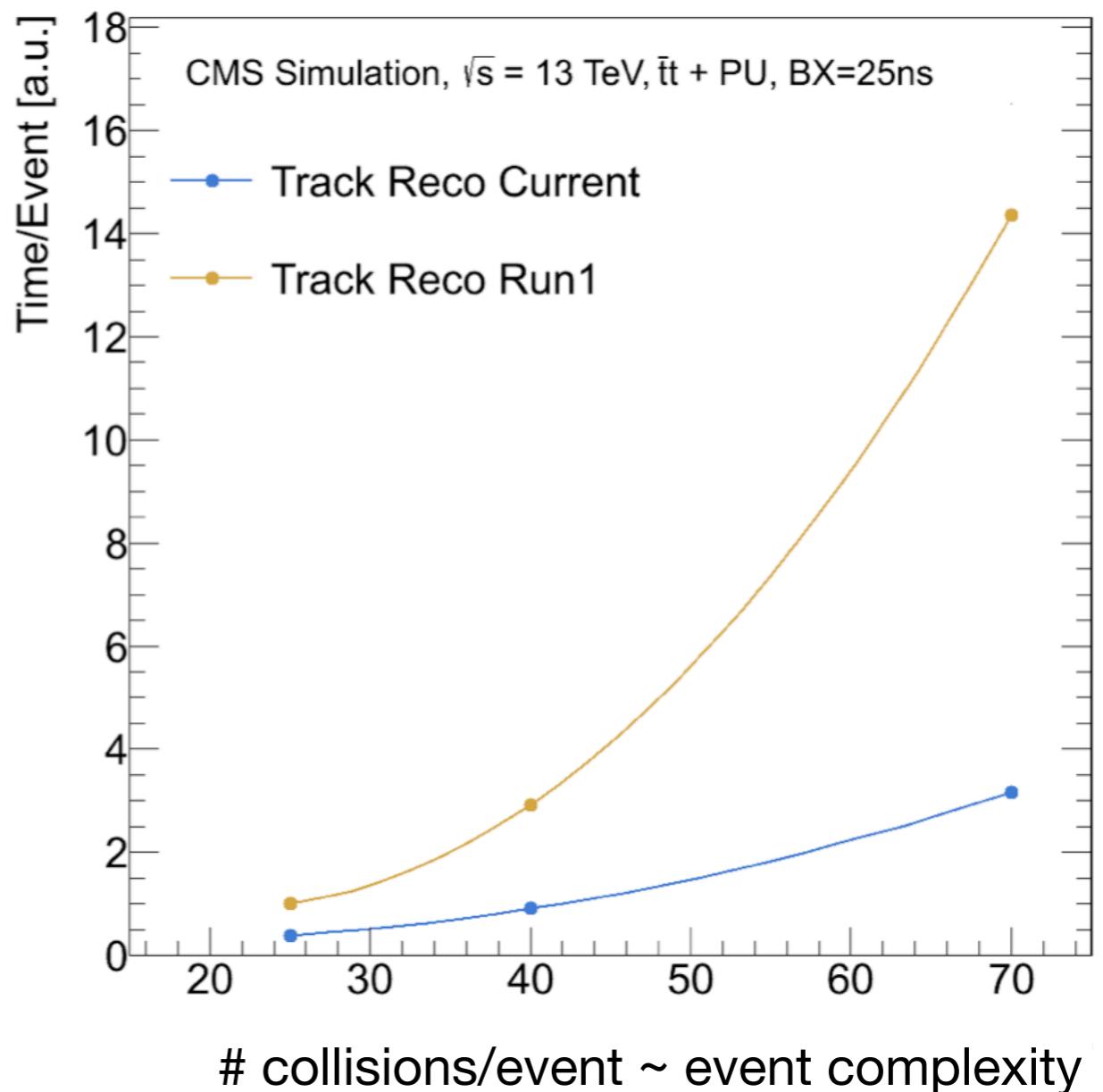
Future challenges @ HL-LHC

Modern machine learning methods might be the way out!

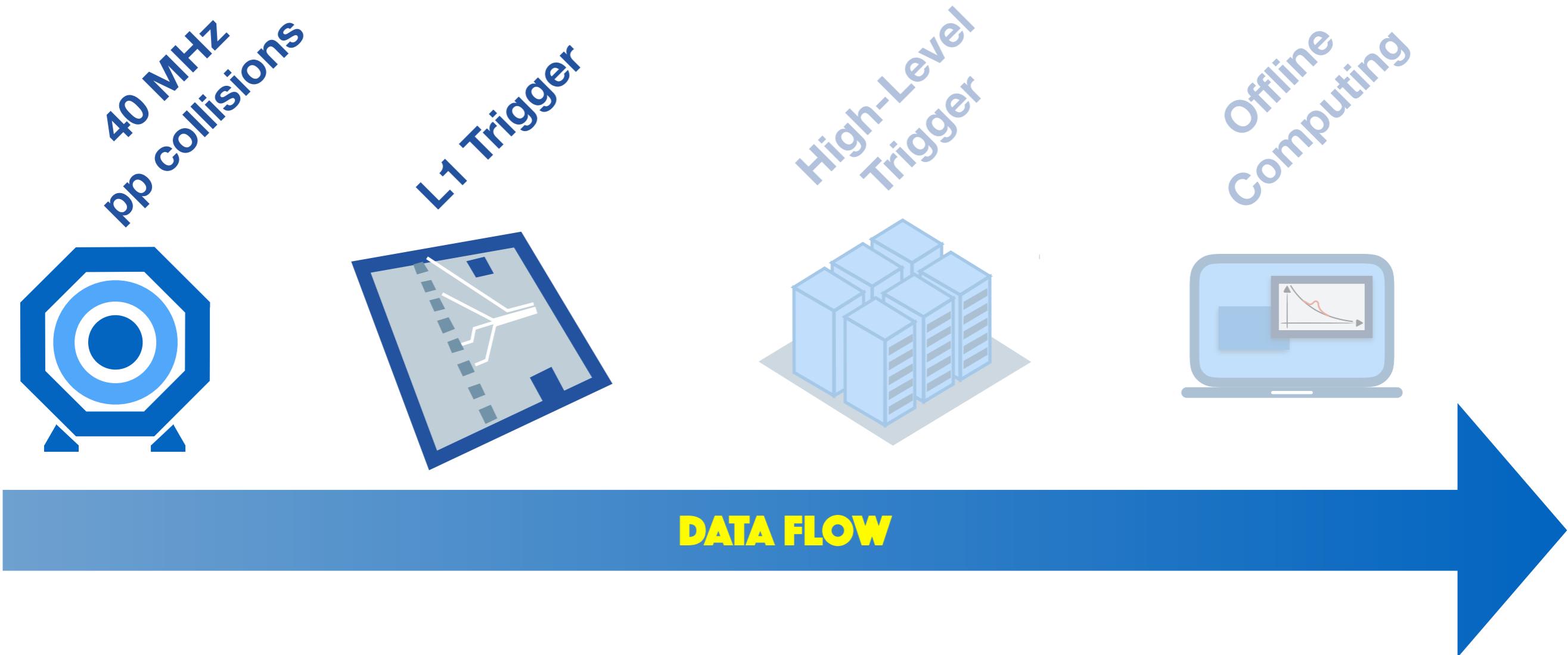
Current event reconstruction
algorithms will not be sustainable

**Recast instead the problem as
a machine learning problem**

- ▶ Excellent physics performance
- ▶ Intrinsically parallelizable → high speed
- ▶ Follow industry trends in developing new devices optimized for ML and speed the up the inference

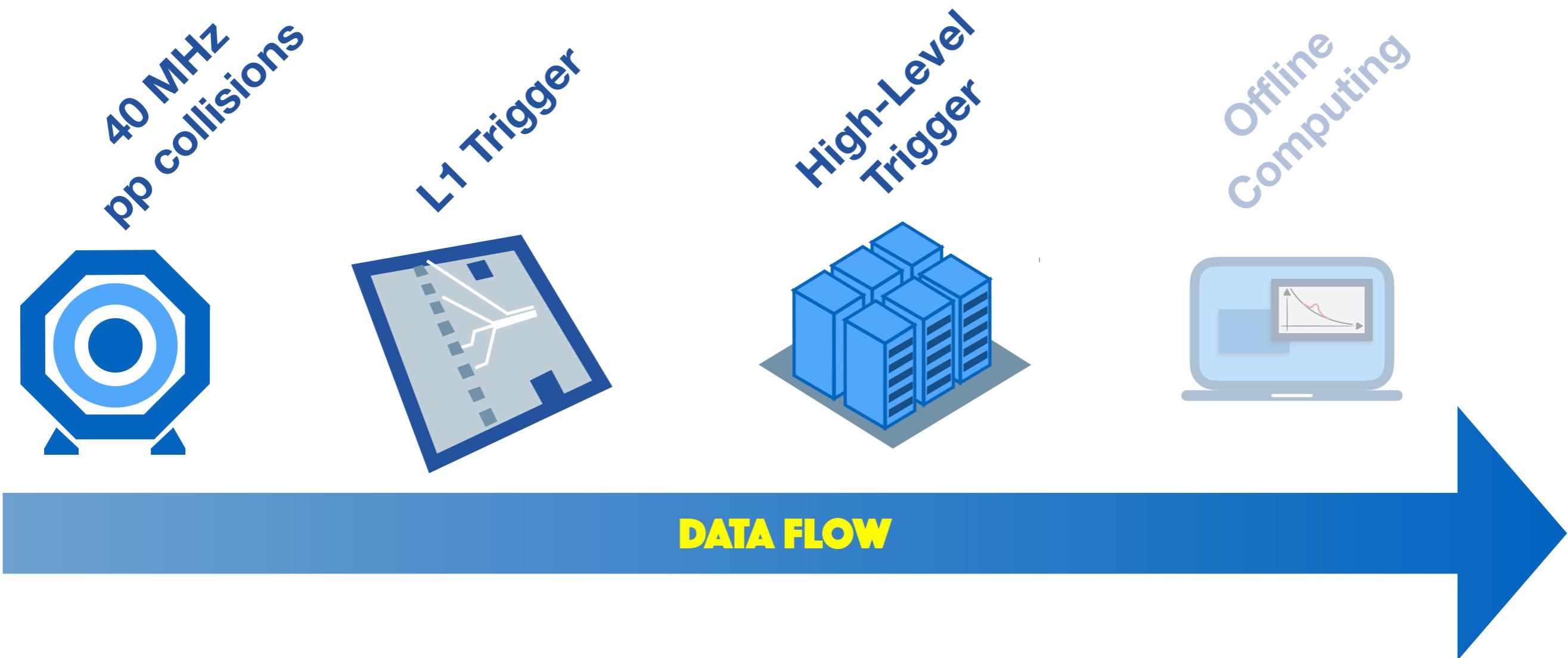


The LHC big data problem



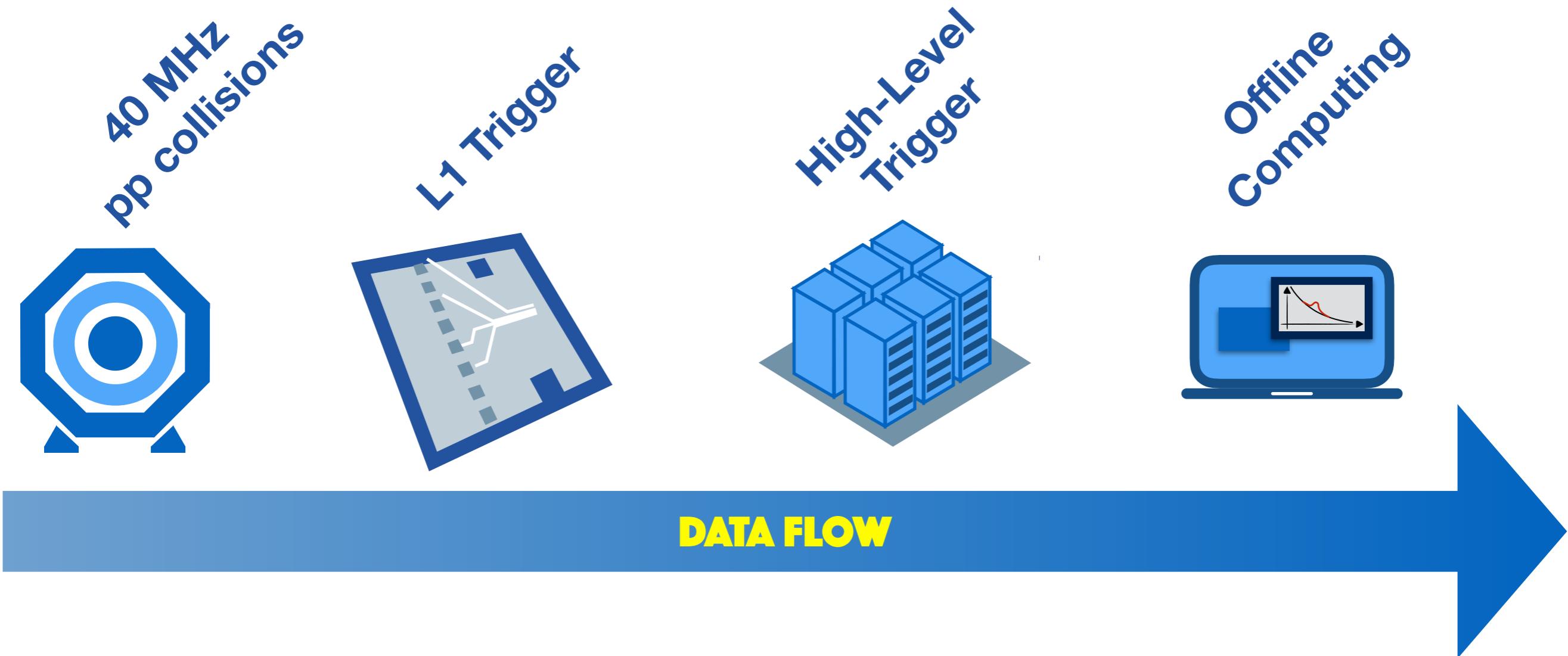
- 40 MHz in / 100 kHz out
- Absorbs 100s TB/s
- Trigger decision to be made in **~ 10 μ s**
- Coarse local reconstruction
- FPGAs / Hardware implemented

The LHC big data problem



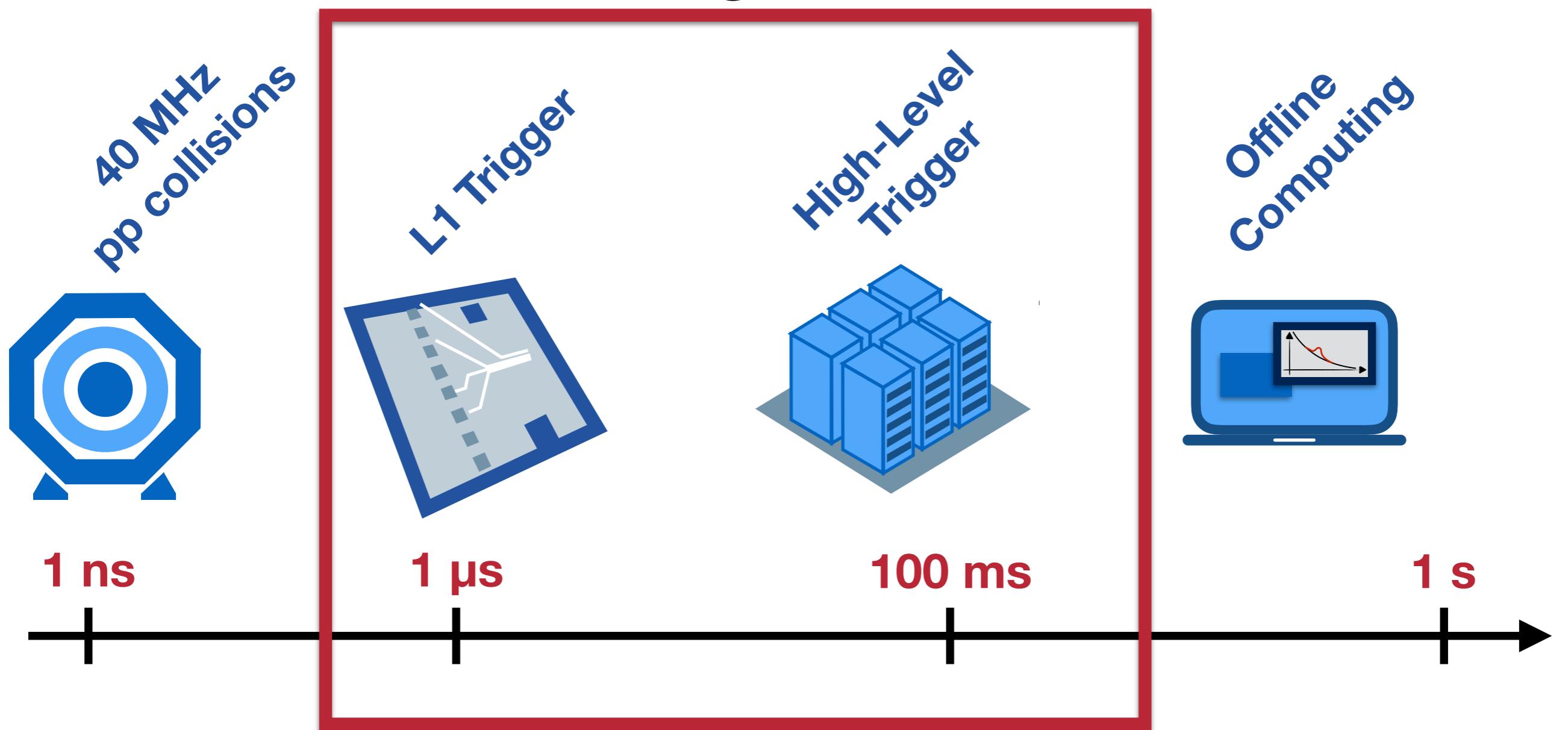
- 100 KHz in / 1 KHz out
- Output: ~ 500 KB/event
- Processing time **~ 300 ms**
- Simplified global reconstruction
- Software implemented on CPUs

The LHC big data problem

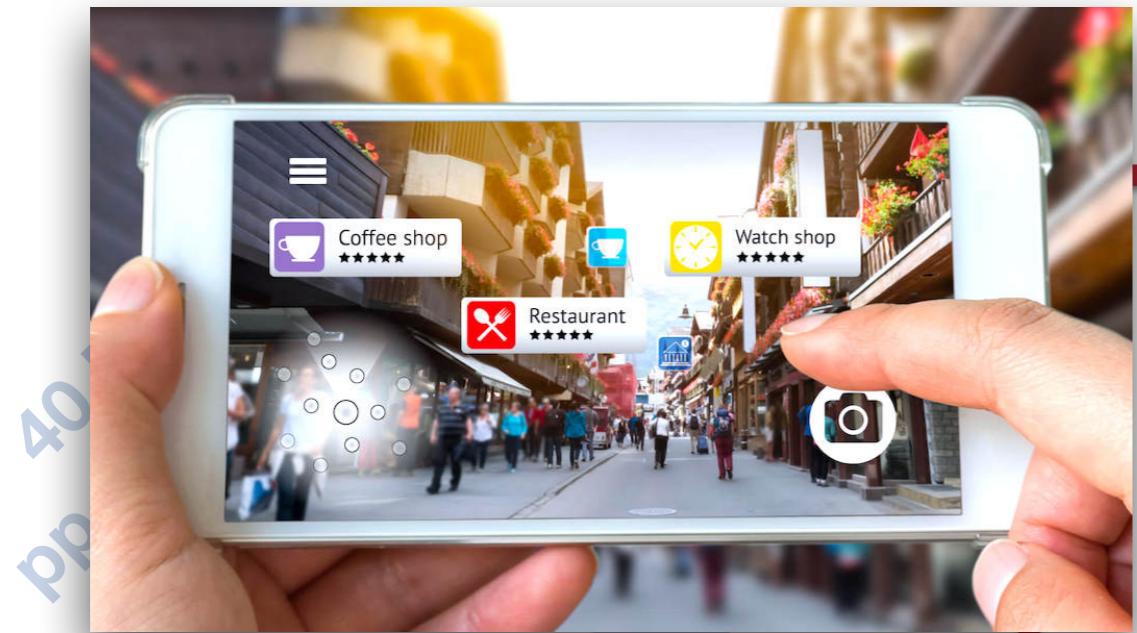


- Output: max. 1 MB/event
- Processing time $\sim \textbf{20 s}$
- Accurate global reconstruction
- Software implemented on CPUs

The LHC big data problem



Deploy ML algorithms very early in the game
Challenge: strict latency constraints!



High-Level
Trigger



100 ms

1 s



Beyond LHC

Ex: self-driving cars

A single self-driving test vehicle can produce ~ **30 TB/day**



There are over **250 million cars** on the road in the US alone

If **< 1% replaced by autonomous vehicles** by 2020
→ Huge amount of data generated, not manageable by central servers!

Beyond LHC

Ex: self-driving cars

A single self-driving test vehicle can produce ~ **30 TB/day**



There are over **250 million cars** on the road in the US alone

If < 1% replaced by autonomous vehicles by 2020
→ HUGE amount of data generated, not manageable by central servers!

Need **edge computing architectures**, low power and small in size to run powerful data analytics programs onboard

NB: latency matters! even a few milliseconds of delay can result in an accident!

The stakes are too high to wait the answer from a distant cloud server.

FGPA Vs GPU: Autonomous Car Makers In Silicon Valley Have Definitely Chosen A Side When It Comes To AI Chips



RICHA BHATIA · MAY 9, 2018



PEOPLE MIGHT HAVE DIFFERENT OPINION... BUT TODAY WE LEARN ABOUT FPGA & MACHINE LEARNING!

FPGA vs GPU for Machine Learning Applications: Which one is better? Can FPGAs beat GPUs? Farhad Fallahlalehzari, Applications Engineer Like (1) Comments (0)

Synced Follow Mar 30, 2018 · 3 min read

NVIDIA CEO Says “FPGA is Not the Right Answer” for Accelerating AI

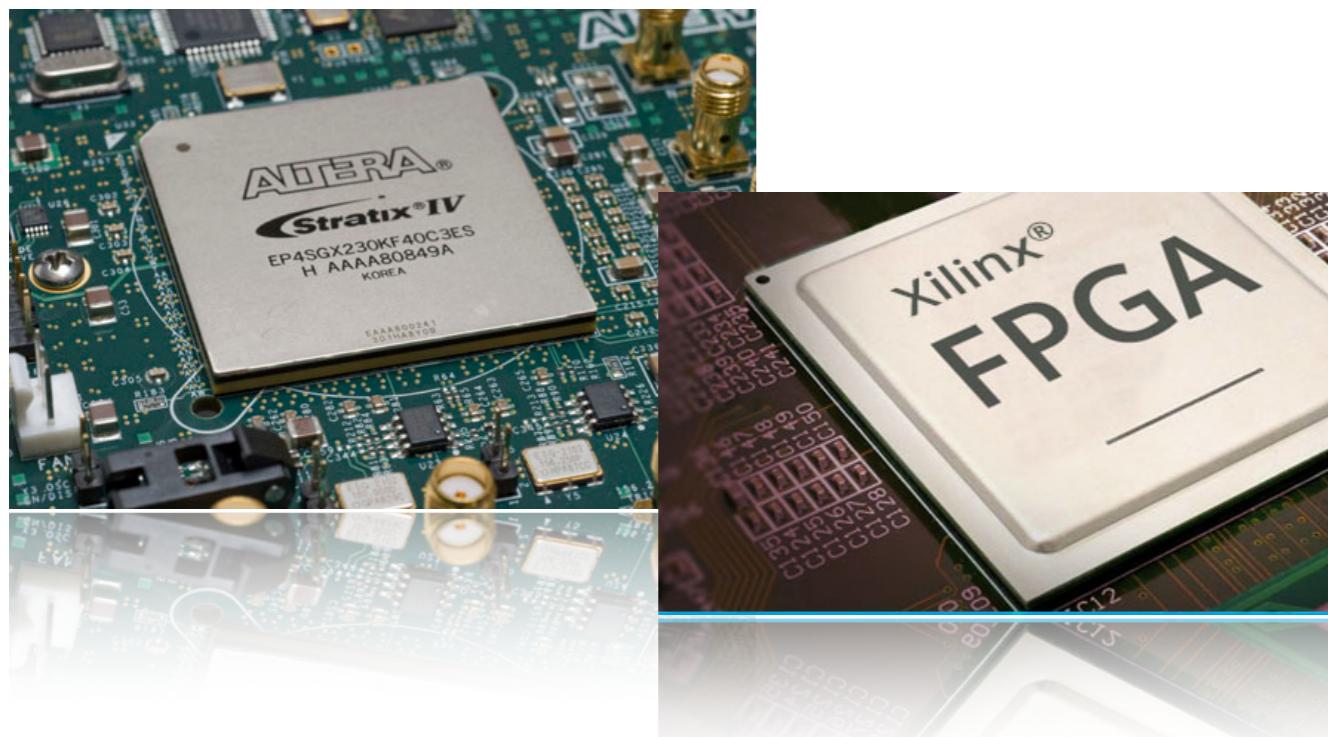
Since the popularity of using machine learning algorithms to extract and process the information from raw data, it has been a race between FPGA and GPU vendors to offer a HW that runs computationally intensive machine learning algorithms fast and efficiently. As has driven most of the advanced machine learning applications, it is regarded as a comparison point.

Since the popularity of using machine learning algorithms to extract and process the information from raw data, it has been a race between FPGA and GPU vendors to offer a HW that runs computationally intensive machine learning algorithms fast and efficiently. As has driven most of the advanced machine learning applications, it is regarded as a comparison point.

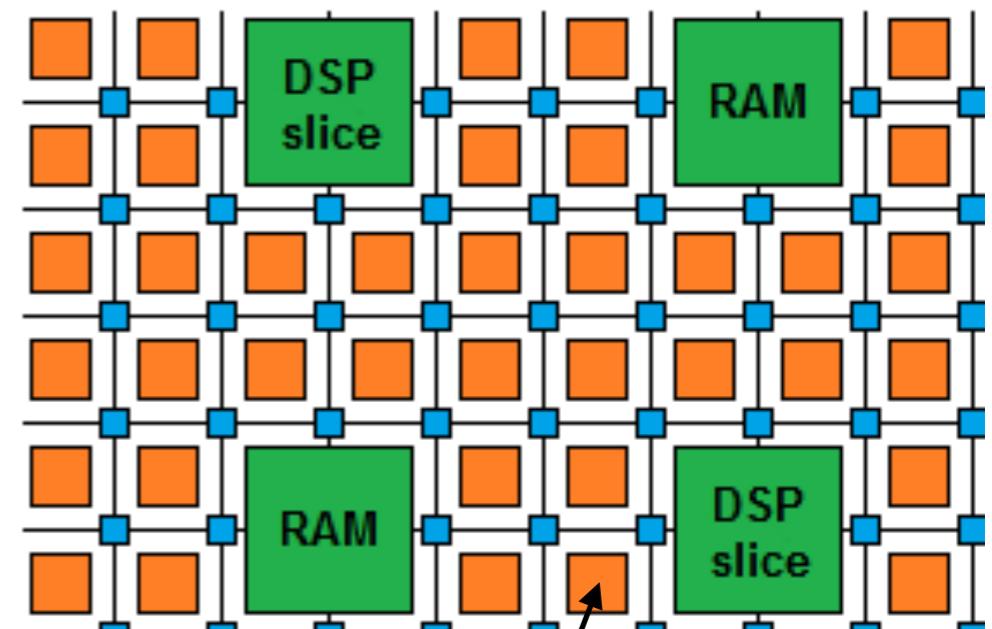
What are FPGAs?

Field Programmable Gate Arrays are reprogrammable integrated circuits

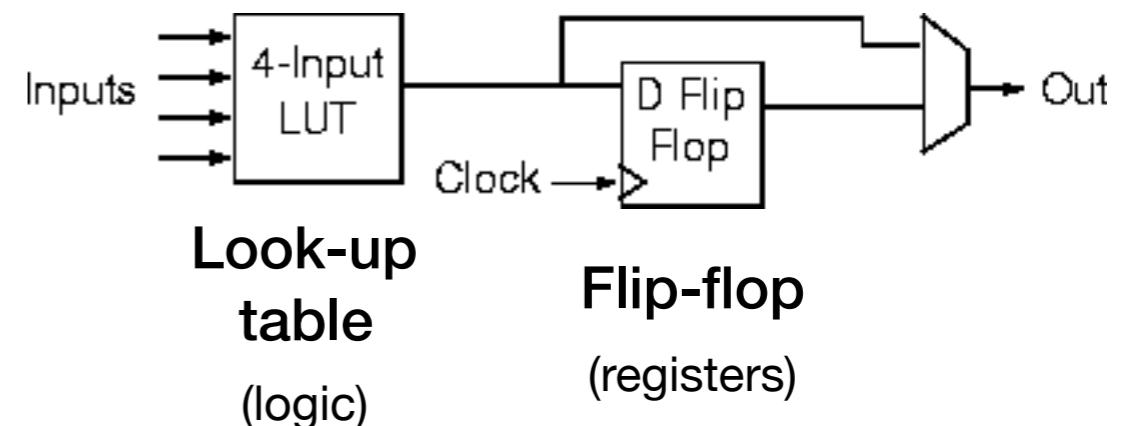
Contain array of **logic cells** used to configure low level operations (bit masking, shifting, addition)



FPGA diagram



Logic cell



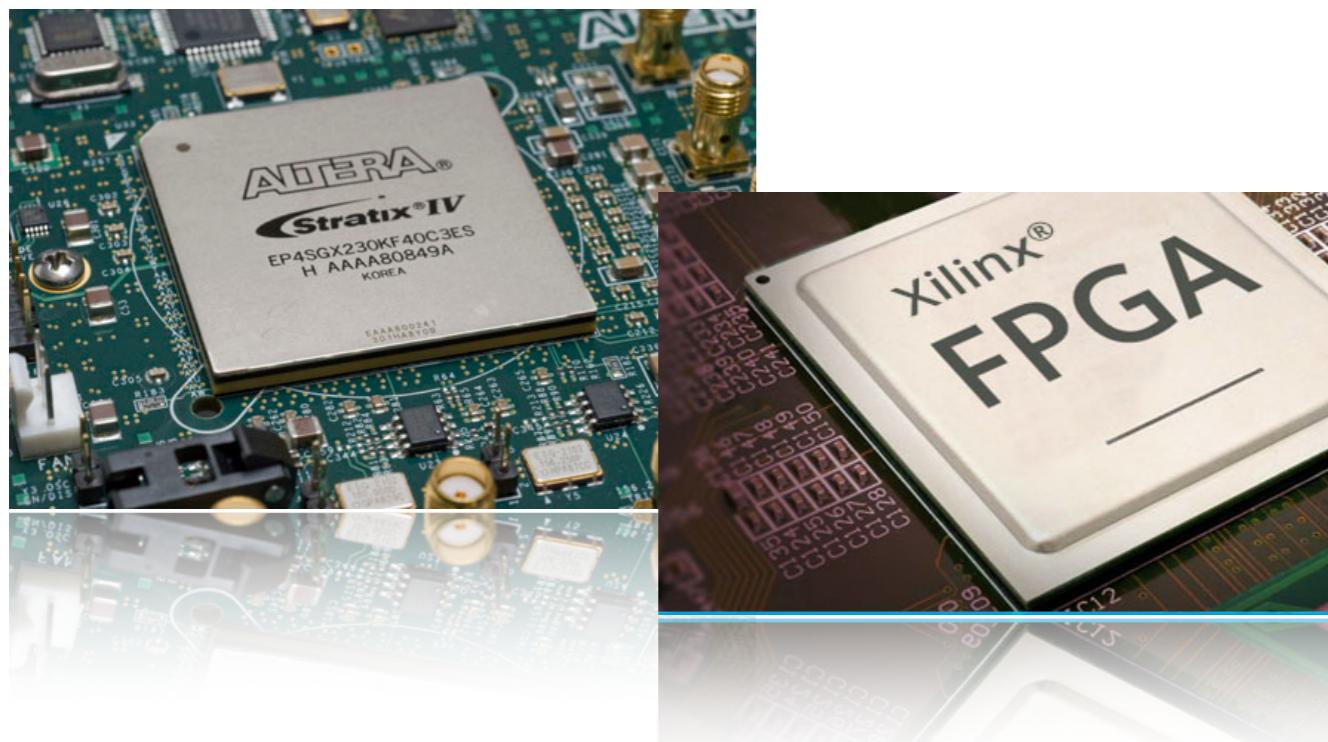
Look-up
table
(logic)

Flip-flop
(registers)

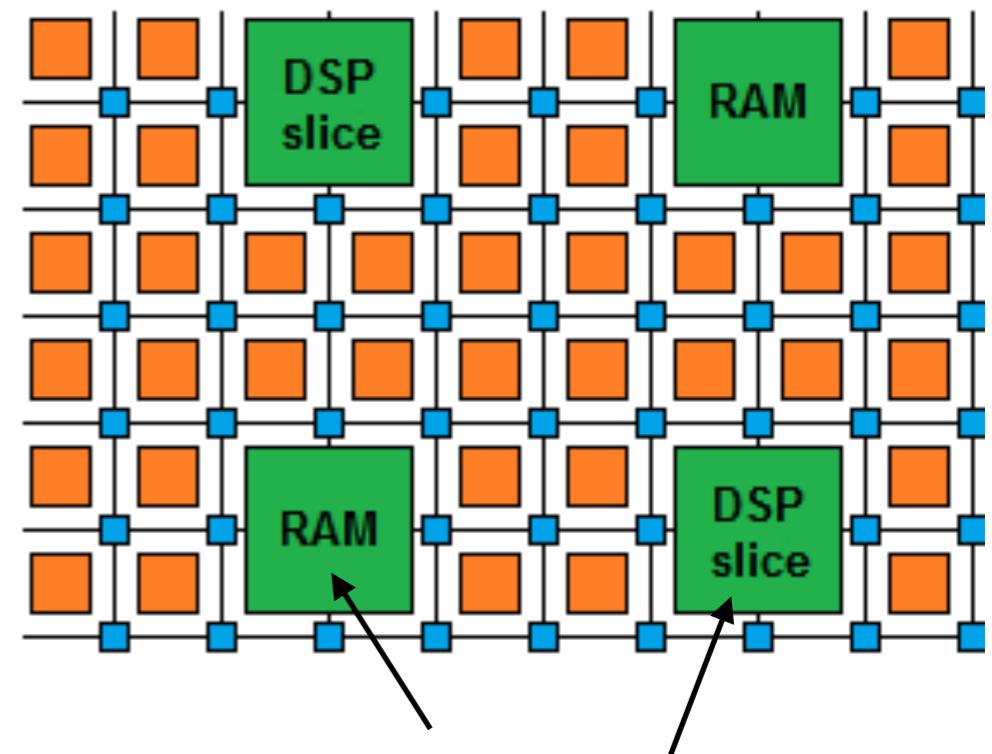
What are FPGAs?

Field Programmable Gate Arrays are reprogrammable integrated circuits

Contain array of **logic cells** used to configure low level operations (bit masking, shifting, addition)



FPGA diagram



Also contain embedded components:

Digital Signal Processors (DSPs):
logic units used for multiplications

Random-access memories (RAMs):
embedded memory elements

What are FPGAs?

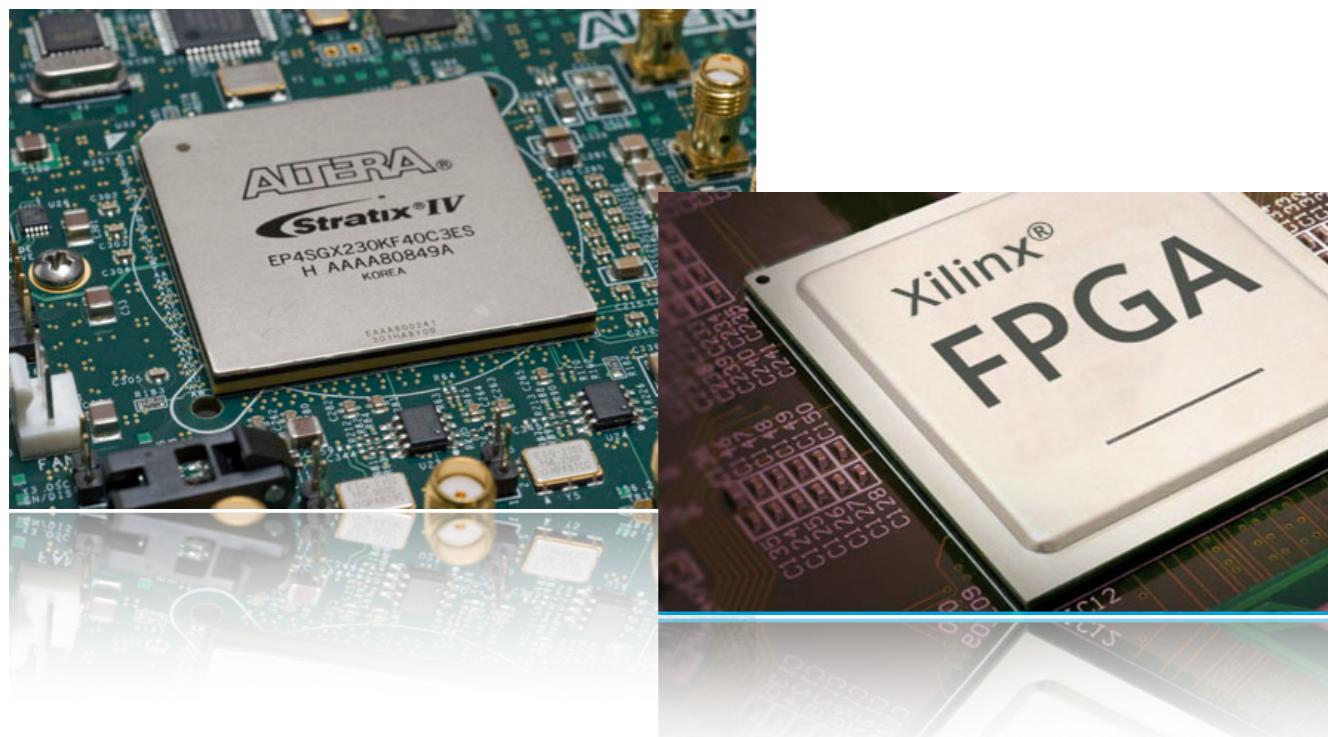
Field Programmable Gate Arrays are reprogrammable integrated circuits

Contain array of **logic cells** embedded with **DSPs**, **BRAMs**, etc.

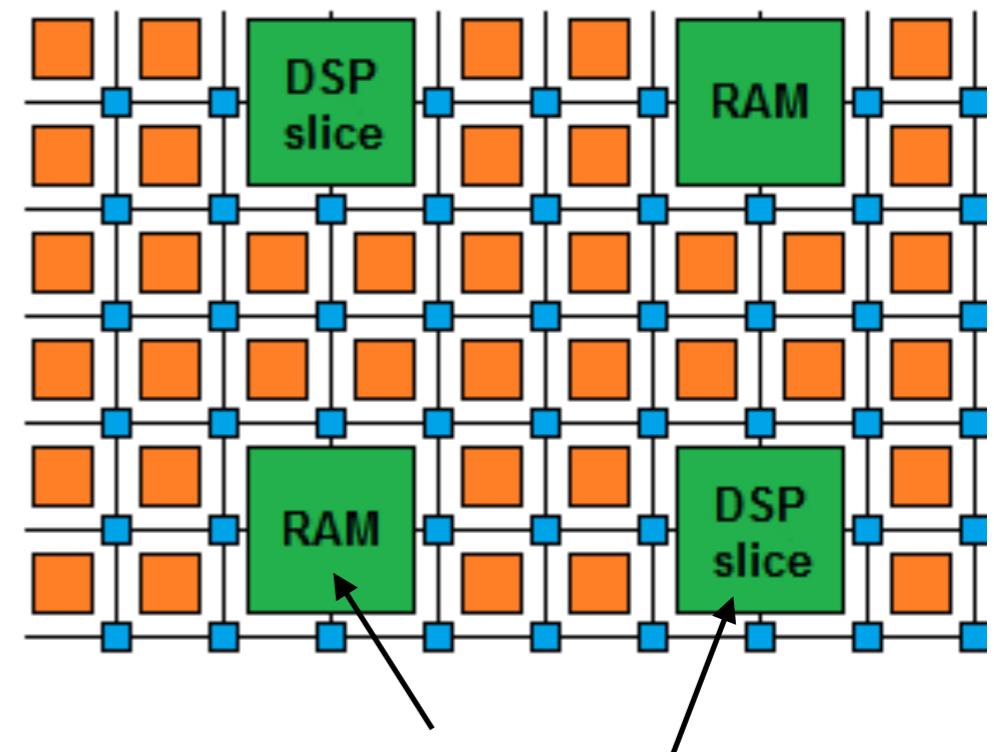
High speed input/output to handle the large bandwidth

Support **highly parallel** algorithm implementations

Low power (relative to CPU/GPU)



FPGA diagram



Digital Signal Processors (DSPs):
logic units used for multiplications

Random-access memories (RAMs):
embedded memory elements

Flip-flops (FF) and look up tables (LUTs) for additions

How are FPGAs programmed?

Hardware Description Languages

HDLs are programming languages which describe electronic circuits

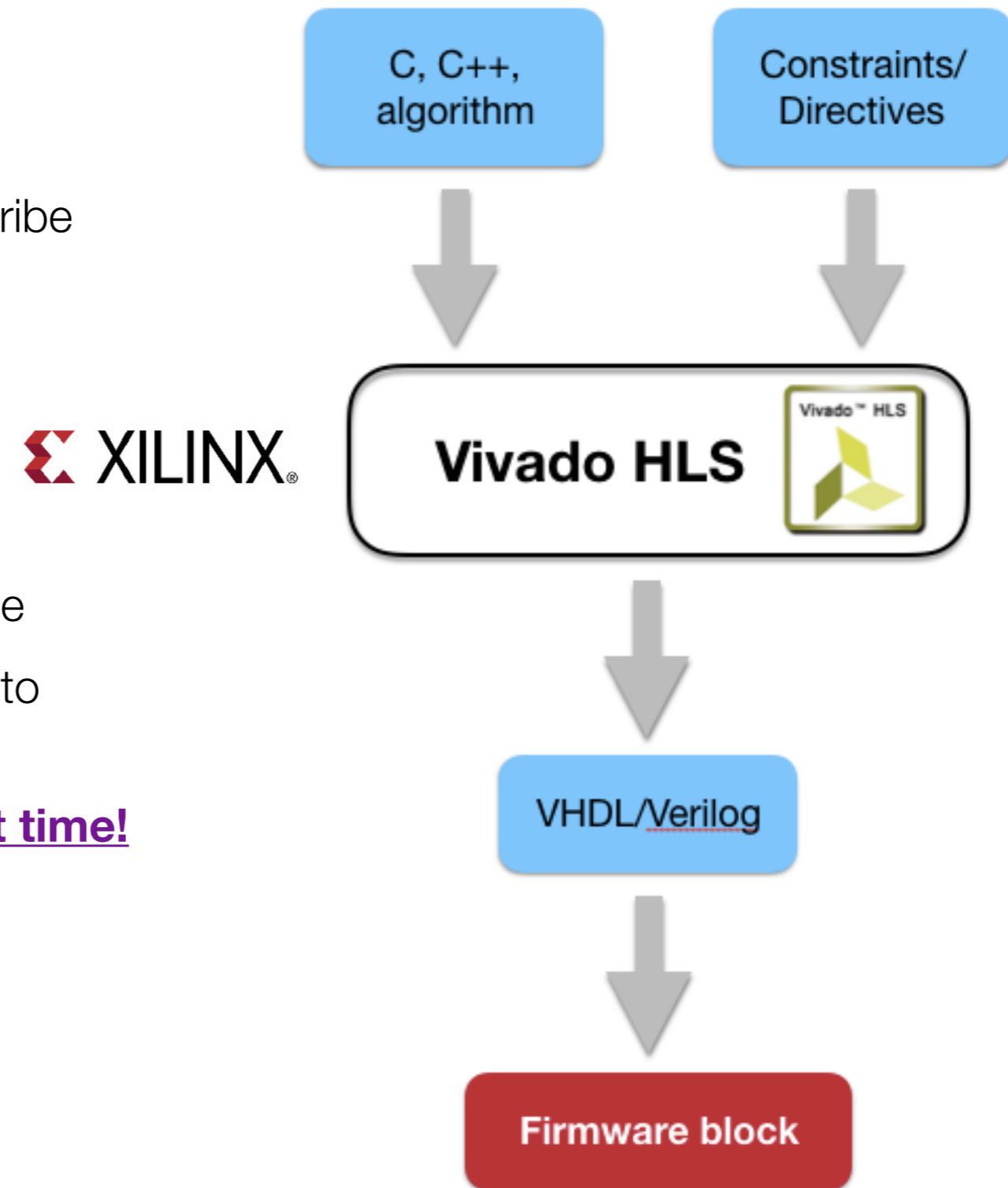
High Level Synthesis

generate HDL from more common C/C++ code

pre-processor directives and constraints used to optimize the timing

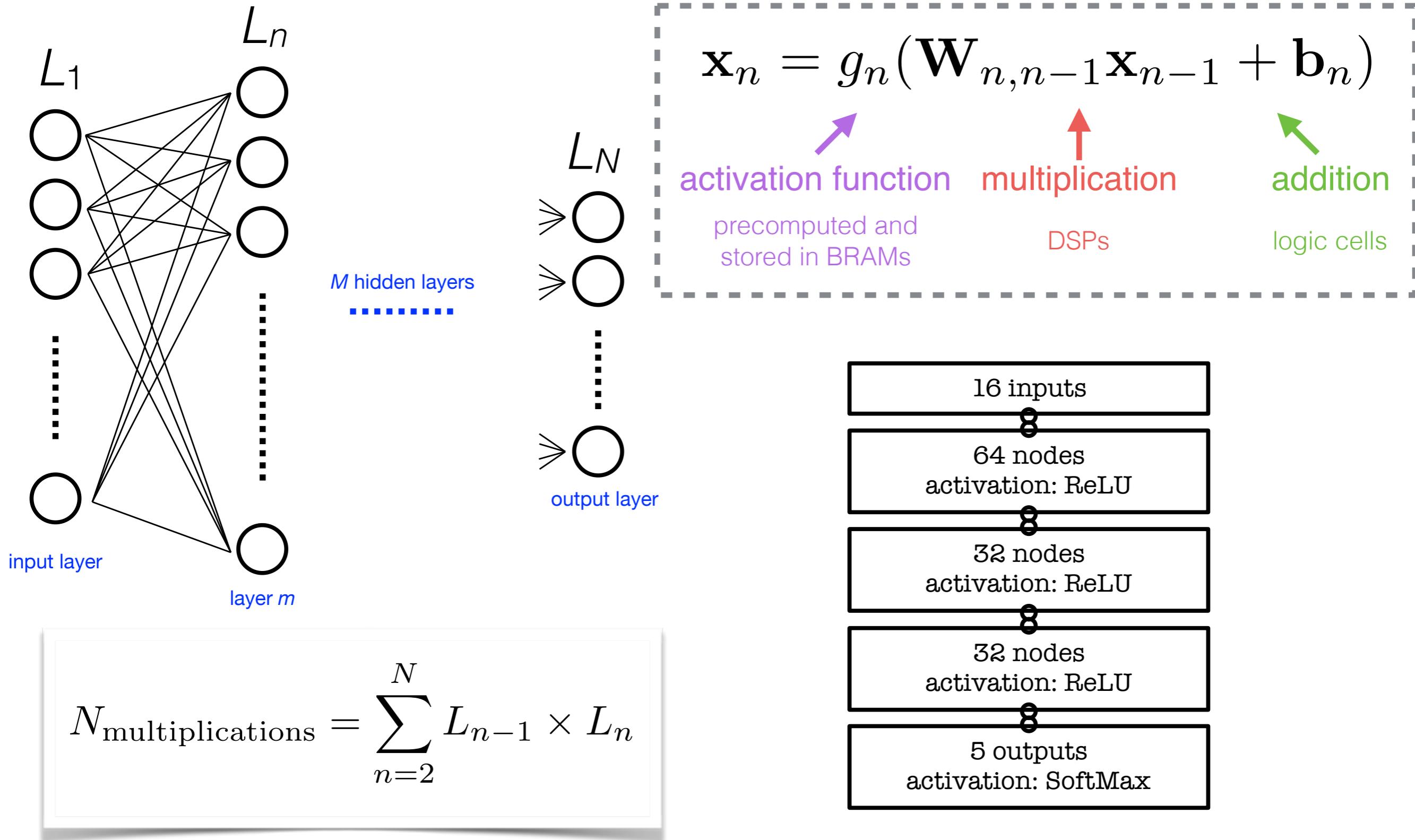
drastic decrease in firmware development time!

We use today **Xilinx Vivado HLS** [*]

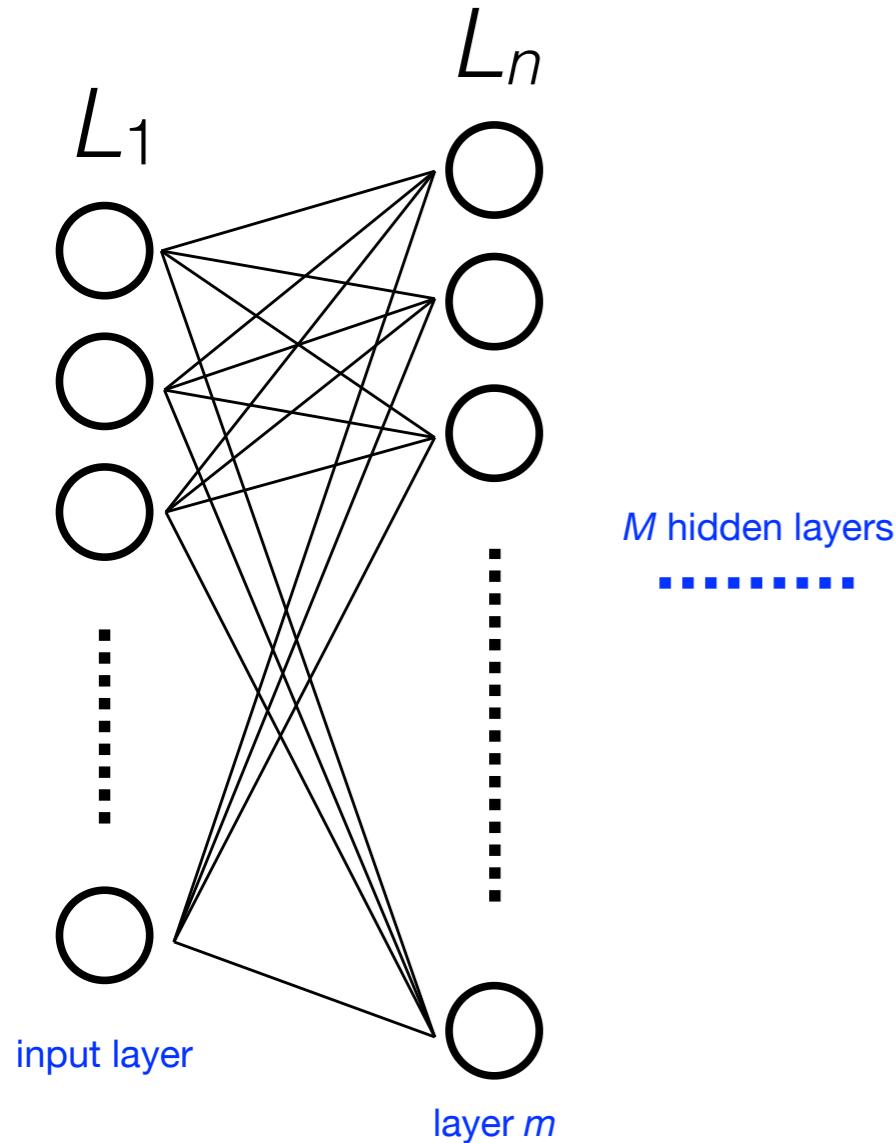


[*] https://www.xilinx.com/support/documentation/sw_manuals/xilinx2014_1/ug902-vivado-high-level-synthesis.pdf

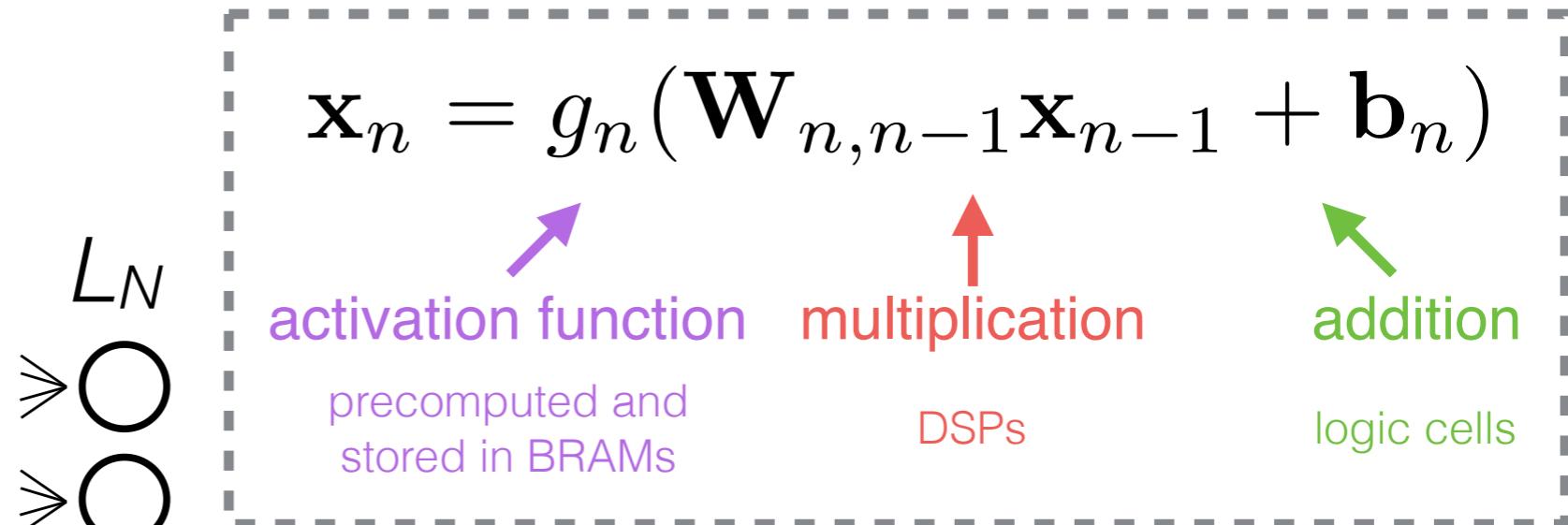
Neural network inference



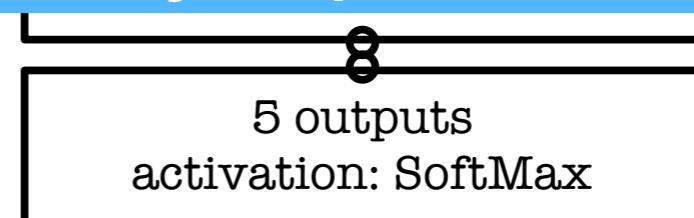
Neural network inference



$$N_{\text{multiplications}} = \sum_{n=2}^N L_{n-1} \times L_n$$



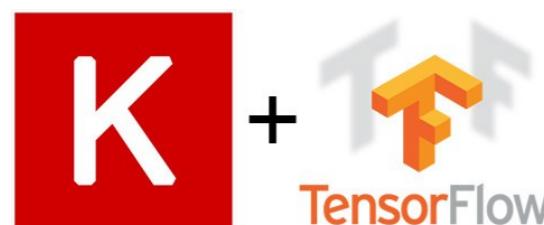
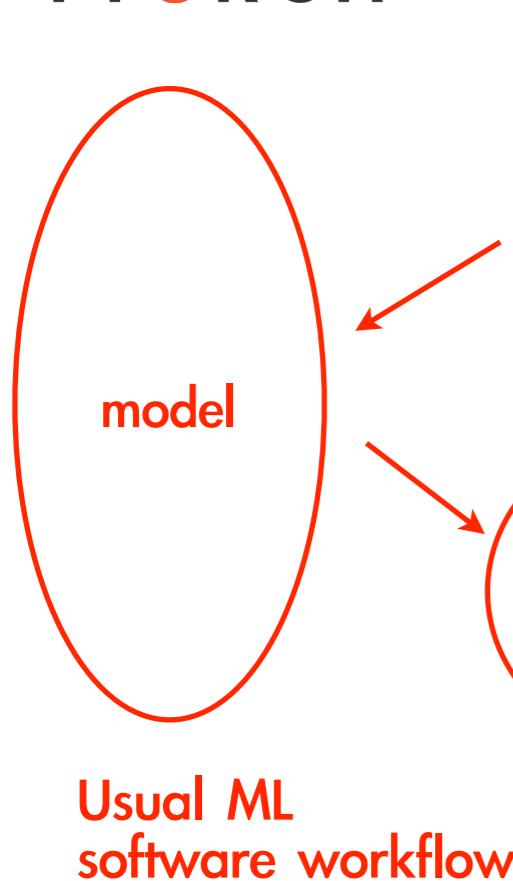
How many resources?
DSPs, LUTs, FFs?
Does the model fit in the latency requirements?



Today you are going to implement a NN on FPGA with this package:

high level synthesis for machine learning

PYTORCH



<https://arxiv.org/abs/1804.06913>



HLS conversion

HLS project

Co-processing kernel

Custom firmware design

tune configuration
precision
reuse/pipeline

<https://hls-fpga-machine-learning.github.io/hls4ml/>

Efficient NN design for FPGAs

FPGAs provide huge flexibility

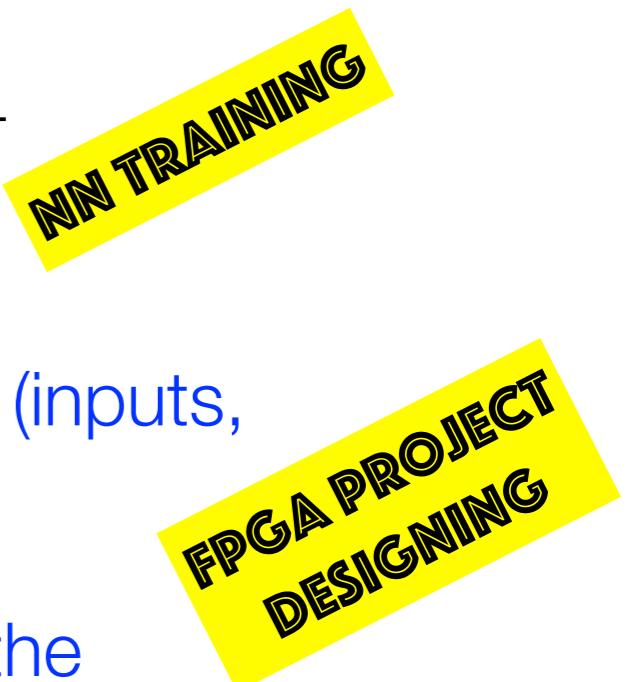
Performance depends on how well you take advantage of this

Constraints:

Input bandwidth
FPGA resources
Latency

Today you will learn how to optimize your project through:

- **compression:** reduce number of synapses or neurons
- **quantization:** reduces the precision of the calculations (inputs, weights, biases)
- **parallelization:** tune how much to parallelize to make the inference faster/slower versus FPGA resources



Today's **hls4ml** hands on

- First part:
 - take confidence with the package, its functionalities and design synthesis by running with one of the provided trained NN
 - learn how to read out an estimate of FPGA resources and latency for a NN after synthesis
 - learn how to optimize the design with quantization and parallelization
- Second part:
 - learn how to export the HLS design to firmware with SDAccel
- Third part:
 - learn how to do model compression and its effect on the FPGA resources/latency
- Fourth part:
 - learn how to accelerate NN inference firmware on a real FPGA (provided on Amazon cloud) with SDAccel
 - timing and resources studies after running on real FPGA