GOBIERNO DE ESPAÑA
MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES
Financiado por la Unión Europea
NextGenerationEU
Plan de Recuperación, Transformación y Resiliencia
AGENCIA ESTATAL DE INVESTIGACIÓN

"This work is supported by Ministerio de Ciencia, Innovación y Universidad con fondos Next Generation y del Plan de Recuperation, Transformacionales y Resiliencia (project -TED2021-130852B-100)"

Porting MADGRAPH to FPGA

Héctor Gutiérrez [1], Luca Fiorini, Alberto Valero, Arantza Oyanguren, Francisco Hervas, Carlos Vico, Javier Fernandez, Santiago Folgueras, Pelayo Leguina

[1] Instituto de Física Corpuscular (CSIC-UV)

2nd Computing Challenges workshop (COMCHA)
October 2nd, 2024

IFIC INSTITUT DE FÍSICA CORPUSCULAR
CSIC CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS
VNIVERSITAT DE VALÈNCIA
HL

Contact:
Hector.Gutierrez@ific.uv.es

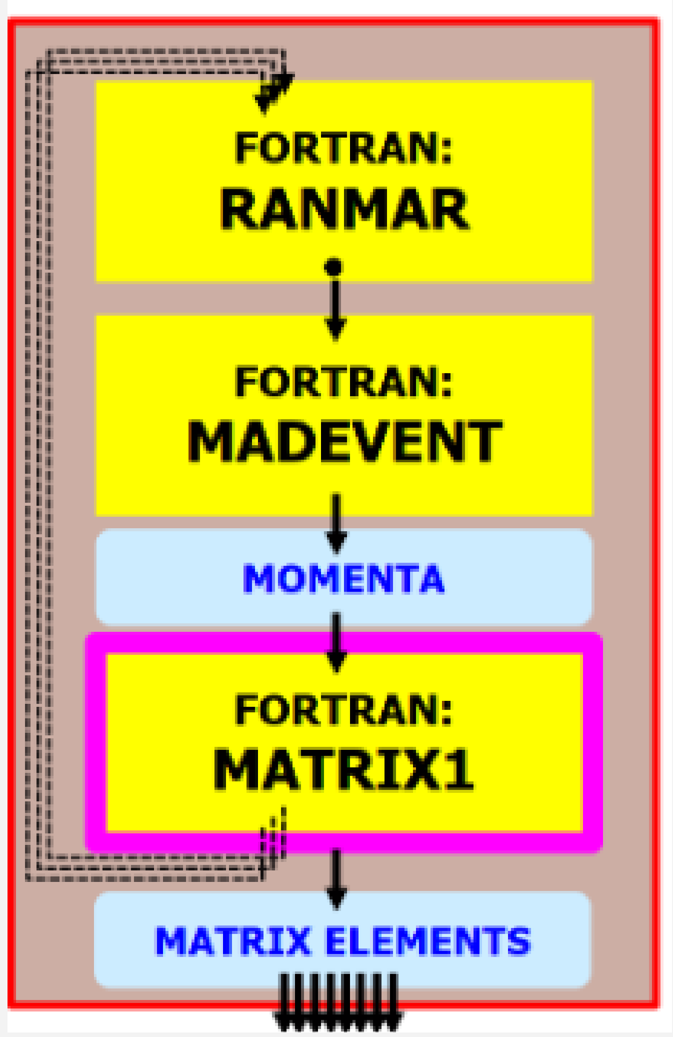
Index

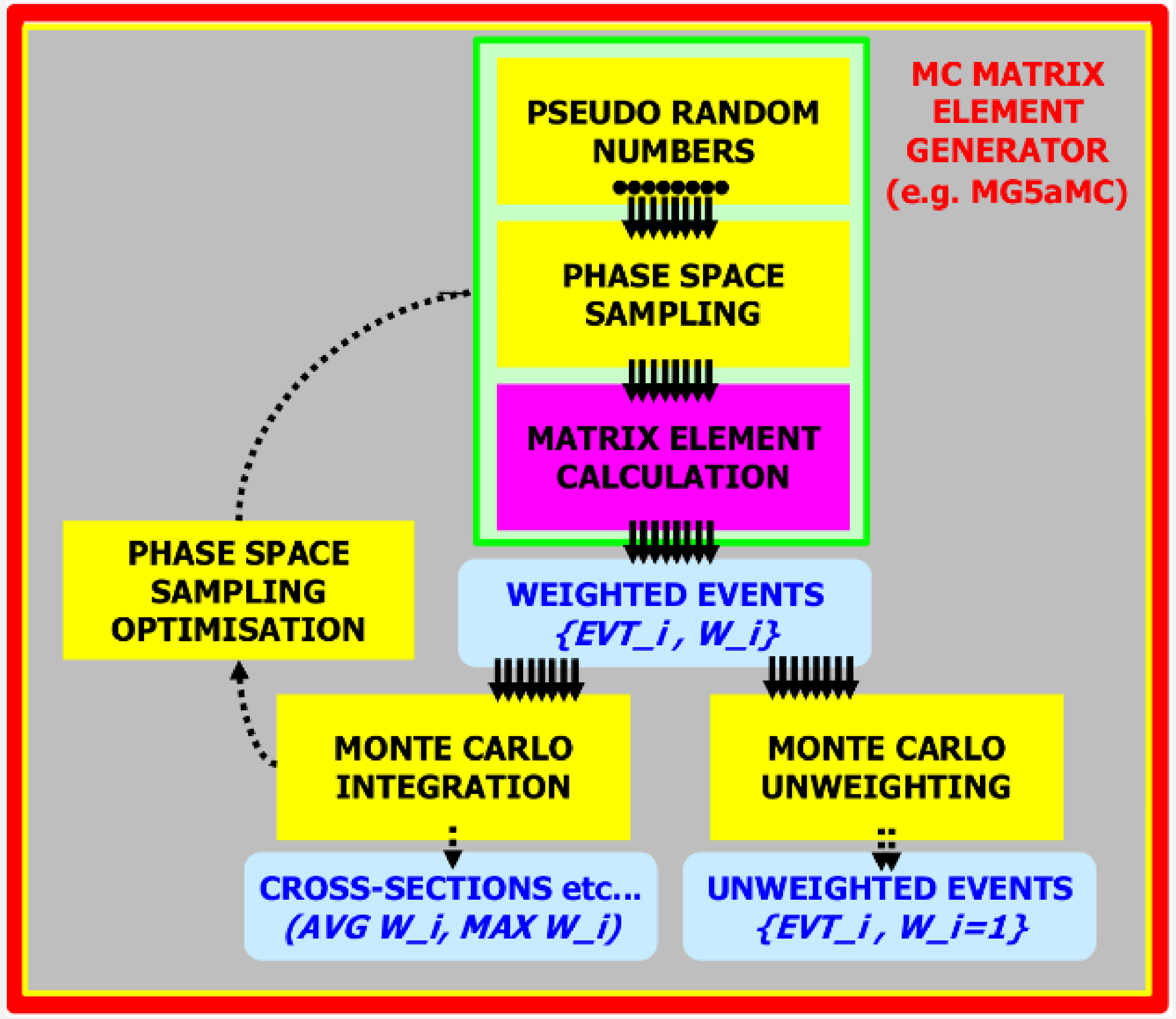
1	2	3	4
MADGRAPH CPU	MADGRAPH GPU	MADGRAPH FPGA	FUTURE IMPLEMENTATIONS

MADGRAPH_aMC@NLO CPU

What is MADGRAPH?:

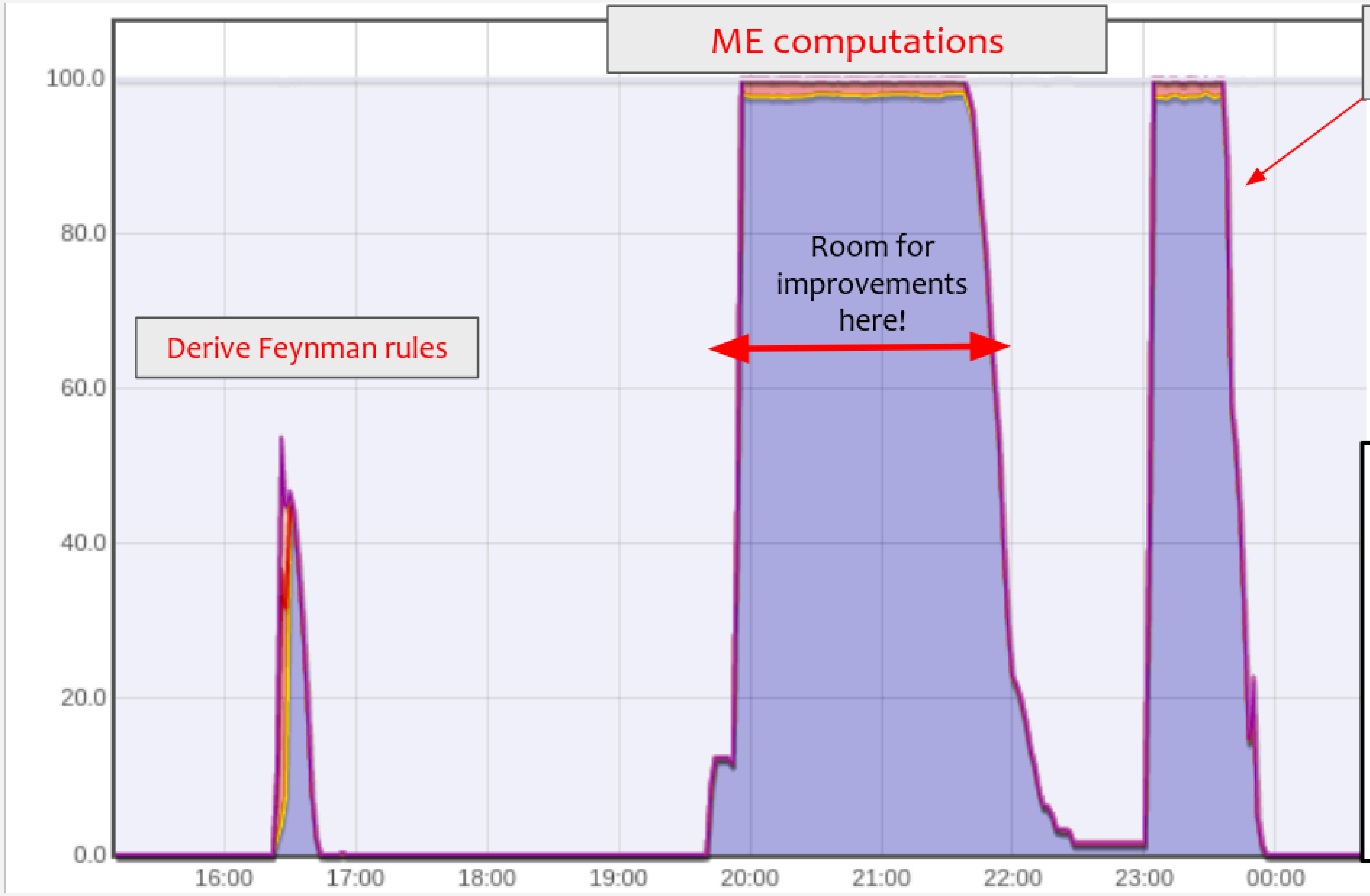
“MadGraph5_aMC@NLO is a framework that aims at providing all the elements necessary for SM and BSM phenomenology, such as the computations of cross sections, the generation of hard events and their matching with event generators. Processes can be simulated to LO accuracy for any user-defined Lagrangian, and the NLO accuracy in the case of QCD (Quantum Chromo Dynamics) corrections to SM processes. Matrix elements at the tree- and one-loop-level can also be obtained. “





PSEUDO RANDOM NUMBERS
MC MATRIX ELEMENT GENERATOR (e.g. MG5aMC)
PHASE SPACE SAMPLING
MATRIX ELEMENT CALCULATION
PHASE SPACE SAMPLING OPTIMISATION
WEIGHTED EVENTS {EVT_i, W_i}
MONTE CARLO INTEGRATION
MONTE CARLO UNWEIGHTING
CROSS-SECTIONS etc... (AVG W_i, MAX W_i)
UNWEIGHTED EVENTS {EVT_i, W_i=1}

MADGRAPH_aMC@NLO CPU



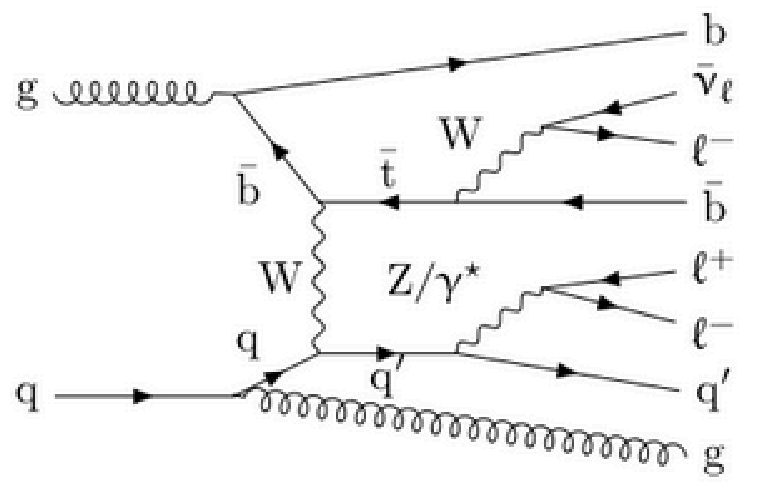
ME computations

Decays, reweightings, ...

Derive Feynman rules

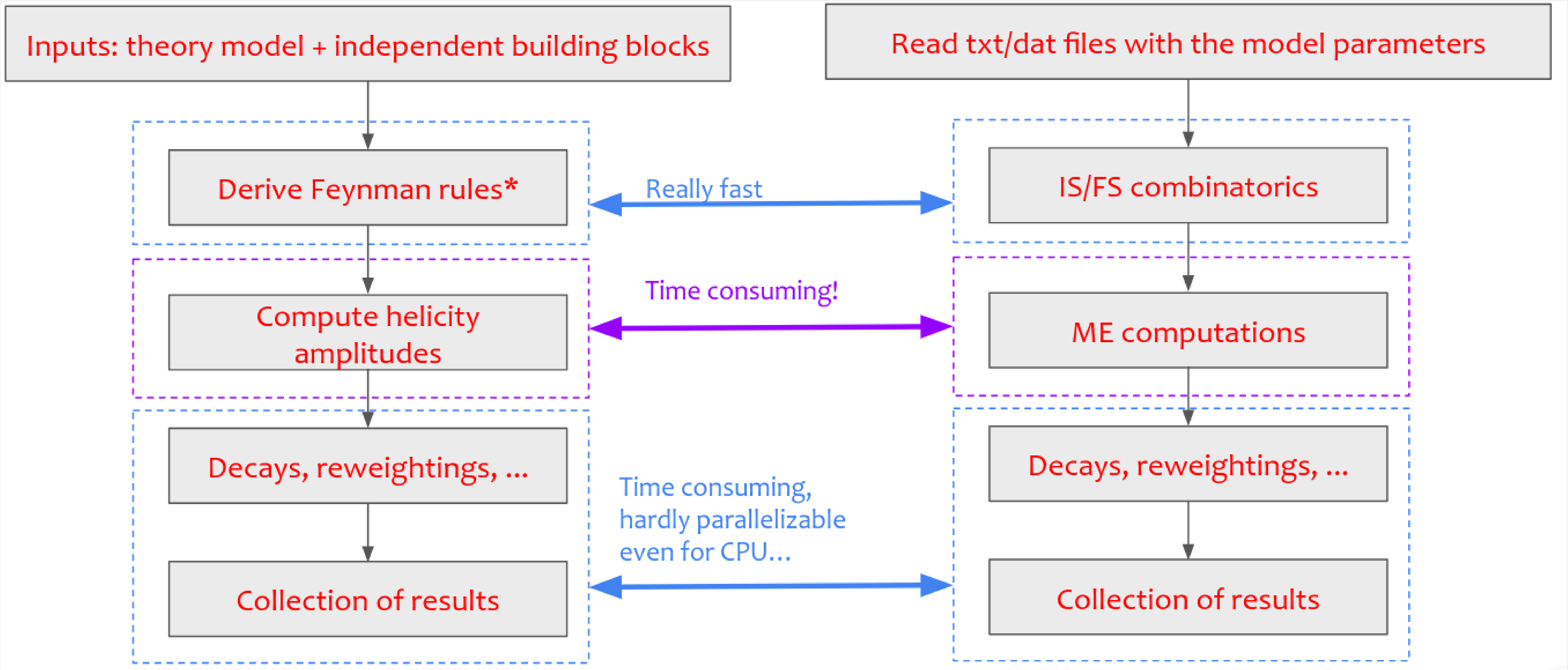
Room for improvements here!

tZq



- “Simple” process
- ~2hrs in 64 cores (normally we don’t use such big machines for processes like this one)
- Integrating an accessible corner of phase space (mll > 30 GeV)
- Below 30 GeV, process is undoable because computations do not converge quickly and gets kicked out due to walltime errors.

MADGRAPH_aMC@NLO CPU



Inputs: theory model + independent building blocks

Derive Feynman rules*

Compute helicity amplitudes

Decays, reweightings, ...

Collection of results

Read txt/dat files with the model parameters

IS/FS combinatorics

ME computations

Decays, reweightings, ...

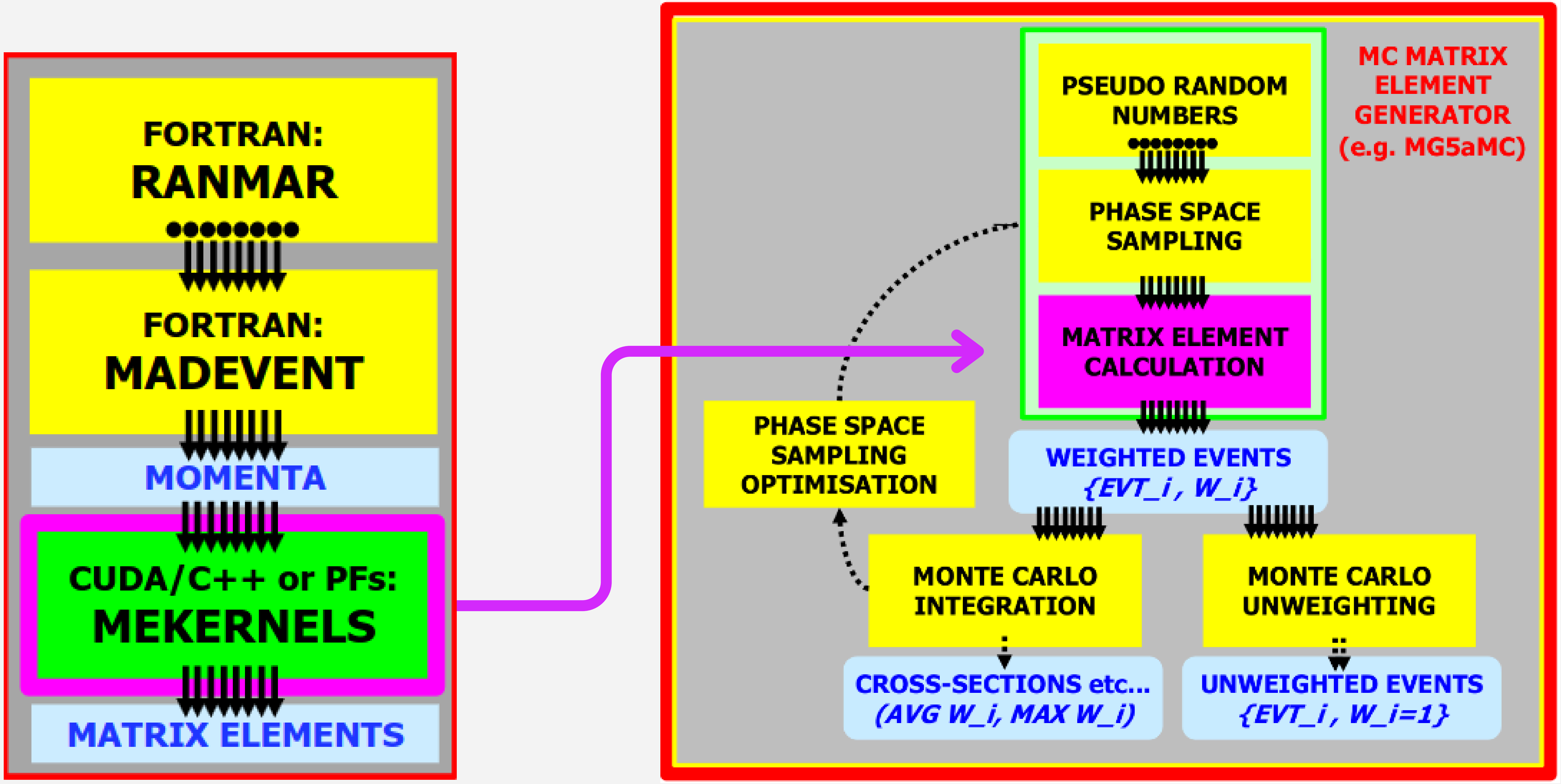
Collection of results

Really fast

Time consuming!

Time consuming, hardly parallelizable even for CPU...

MADGRAPH4GPU



FORTRAN:
RANMAR

FORTRAN:
MADEVENT

MOMENTA

CUDA/C++ or PFs:
MEKERNELS

MATRIX ELEMENTS

MC MATRIX ELEMENT GENERATOR (e.g. MG5aMC)

PSEUDO RANDOM NUMBERS

PHASE SPACE SAMPLING

MATRIX ELEMENT CALCULATION

PHASE SPACE SAMPLING OPTIMISATION

WEIGHTED EVENTS
{EVT_i , W_i}

MONTE CARLO INTEGRATION

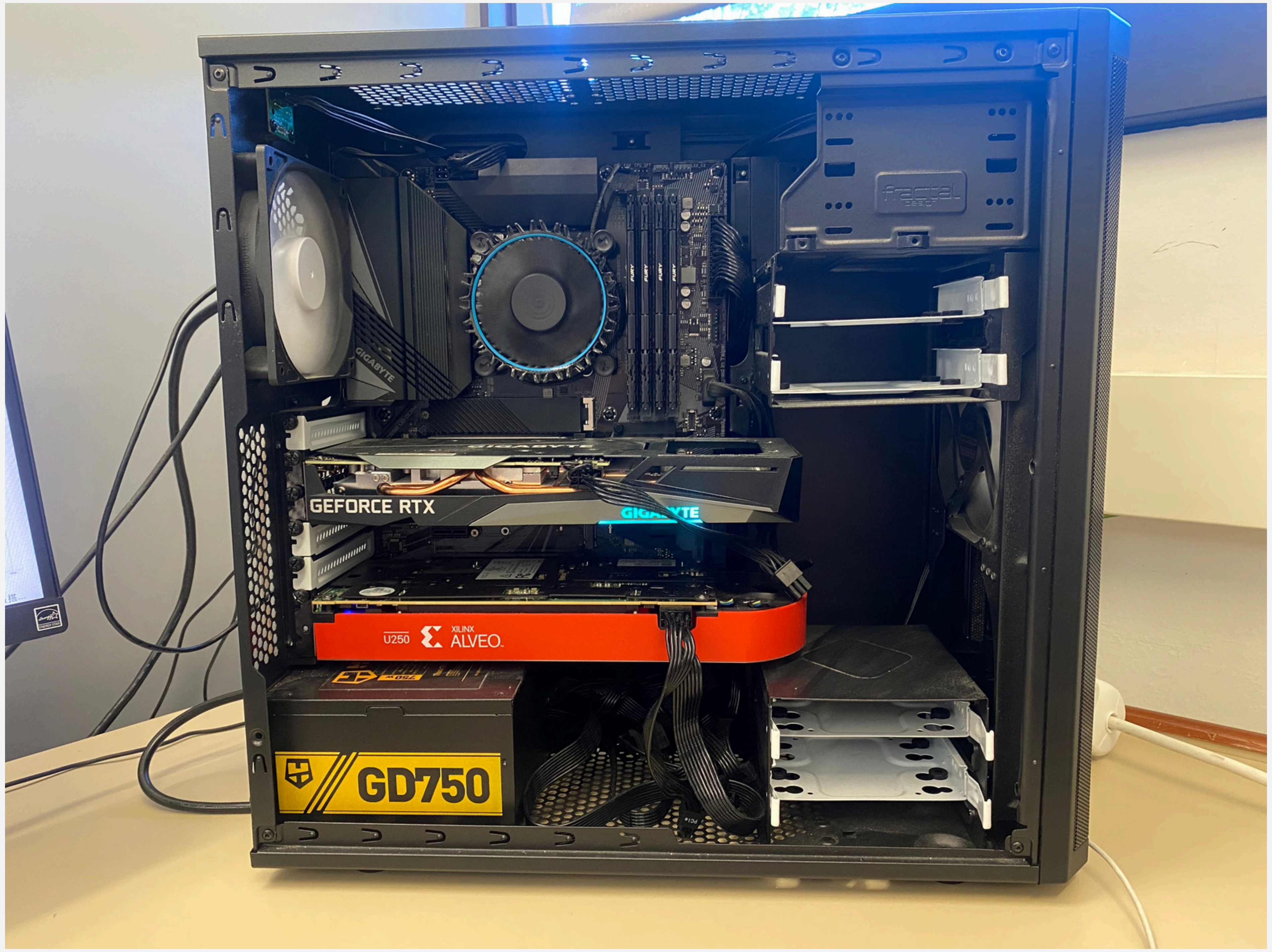
MONTE CARLO UNWEIGHTING

CROSS-SECTIONS etc...
(AVG W_i, MAX W_i)

UNWEIGHTED EVENTS
{EVT_i , W_i=1}

Credits to: Madgraph5_aMC@NLO for GPUs group

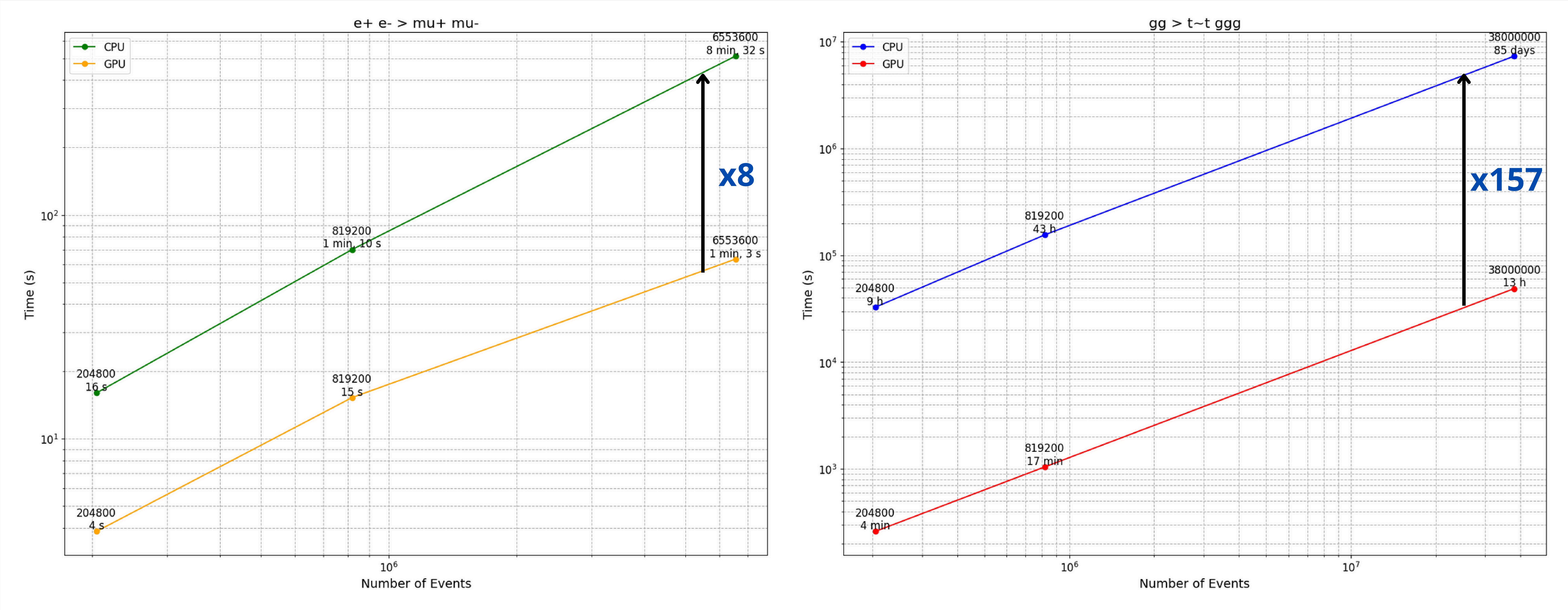
Setup (CPU + GPU)



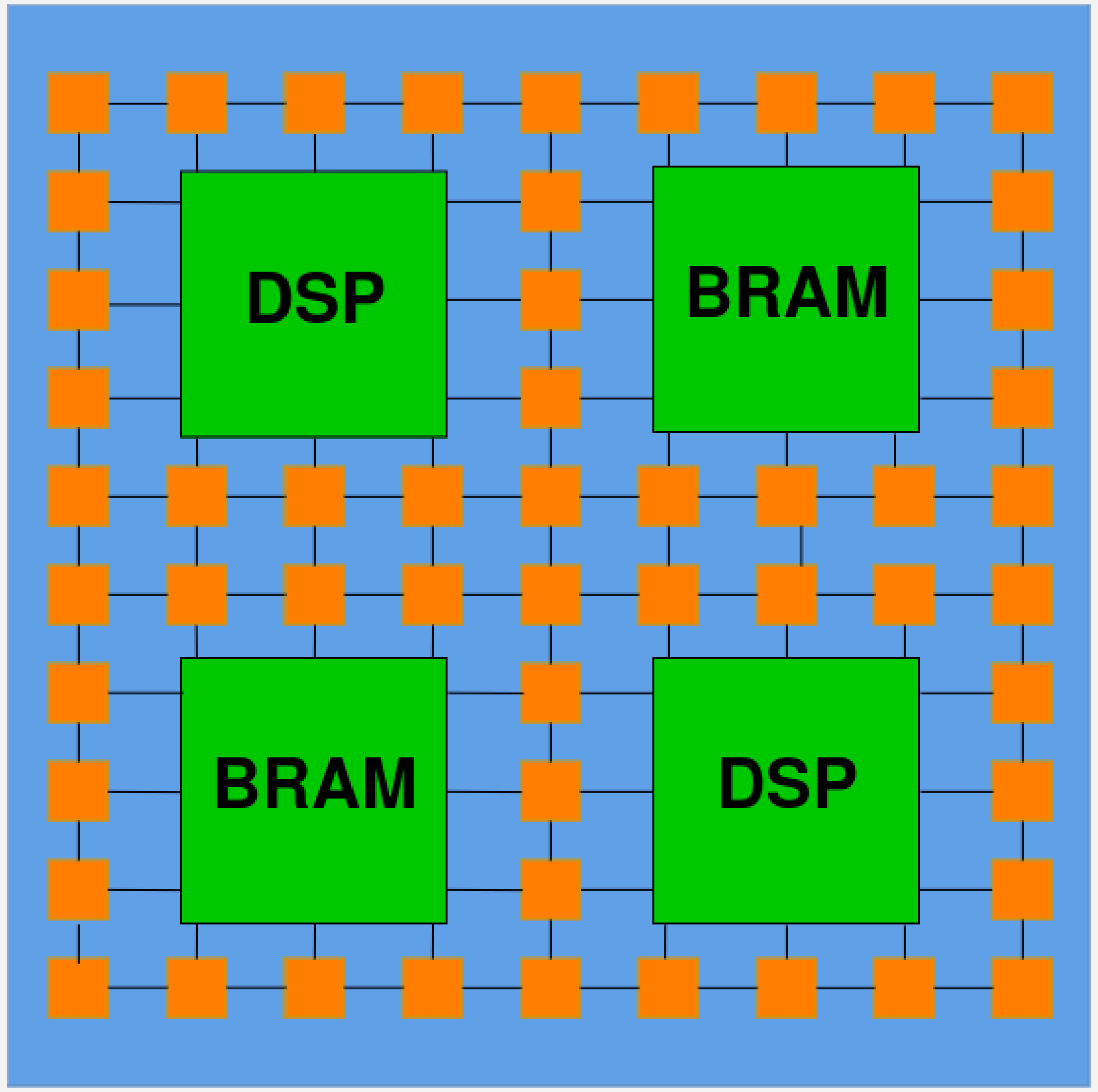
13TH GEN INTEL(R) CORE(TM) I7-13700H 2.40 GHZ

GEFORCE RTX 3050 GIGABYTE

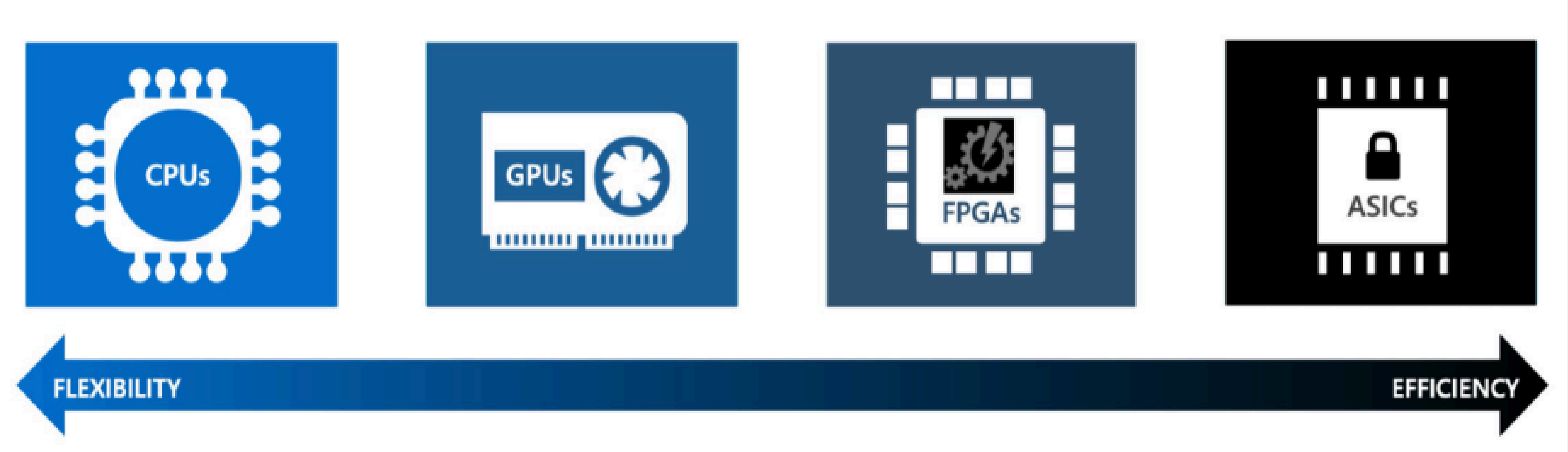
Results CPU vs GPU



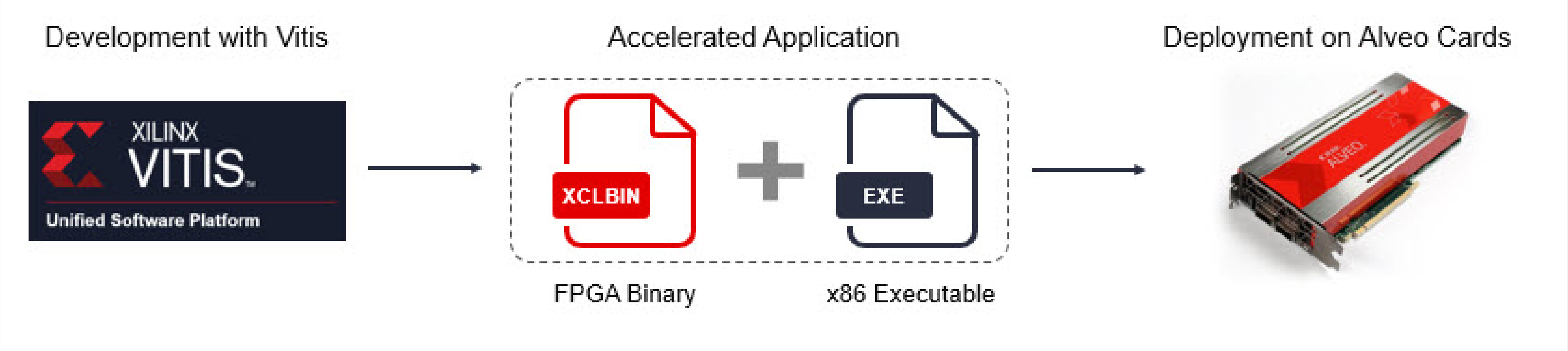
What is an FPGA?

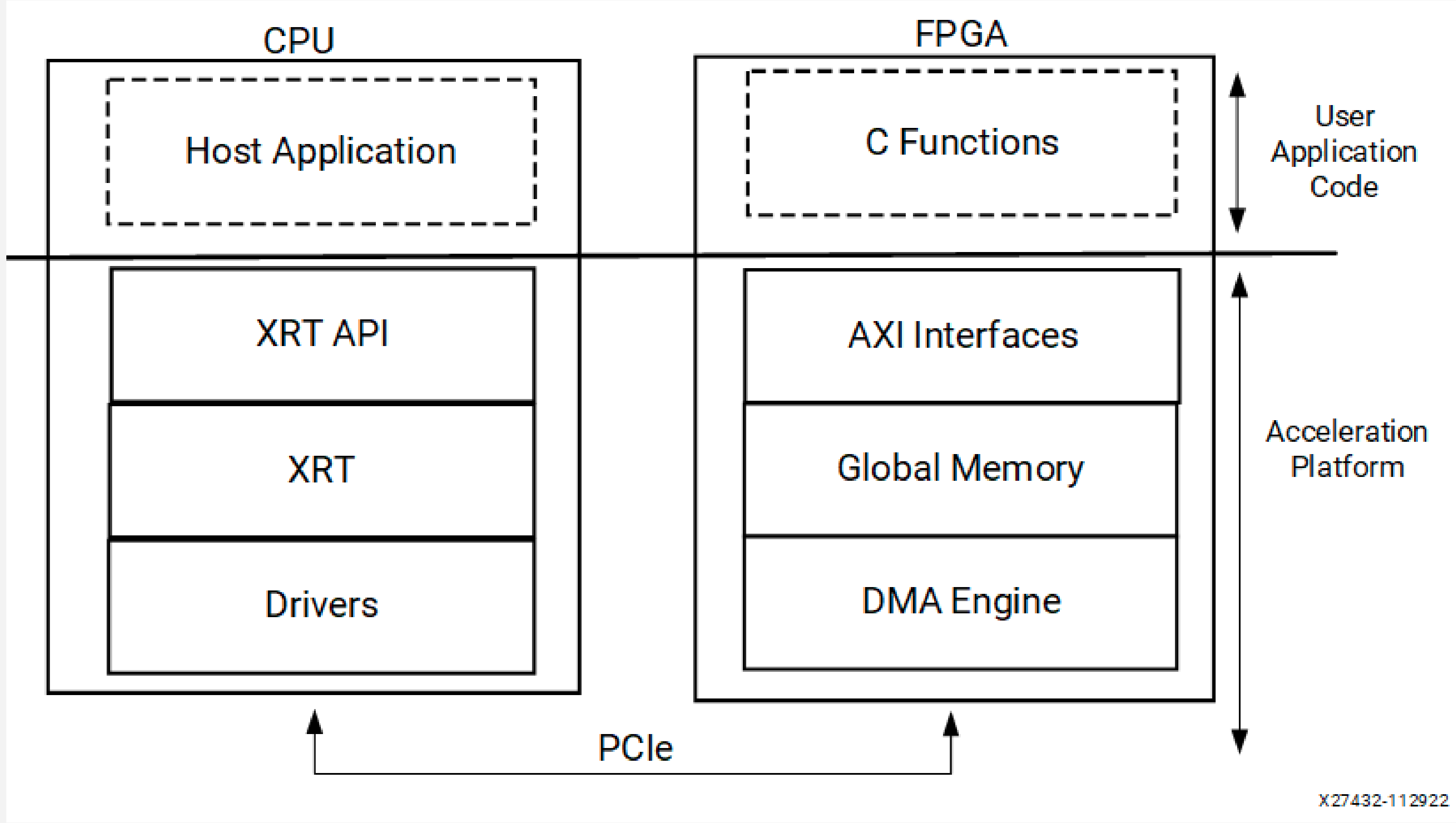


FPGA (Field Programmable Gate Array)
DSP (Digital Signal Processing)
LUT (LookUp Table)
FF (Flip-Flop)
BRAM (Block RAM)



HOW TO PROGRAM IN HLS





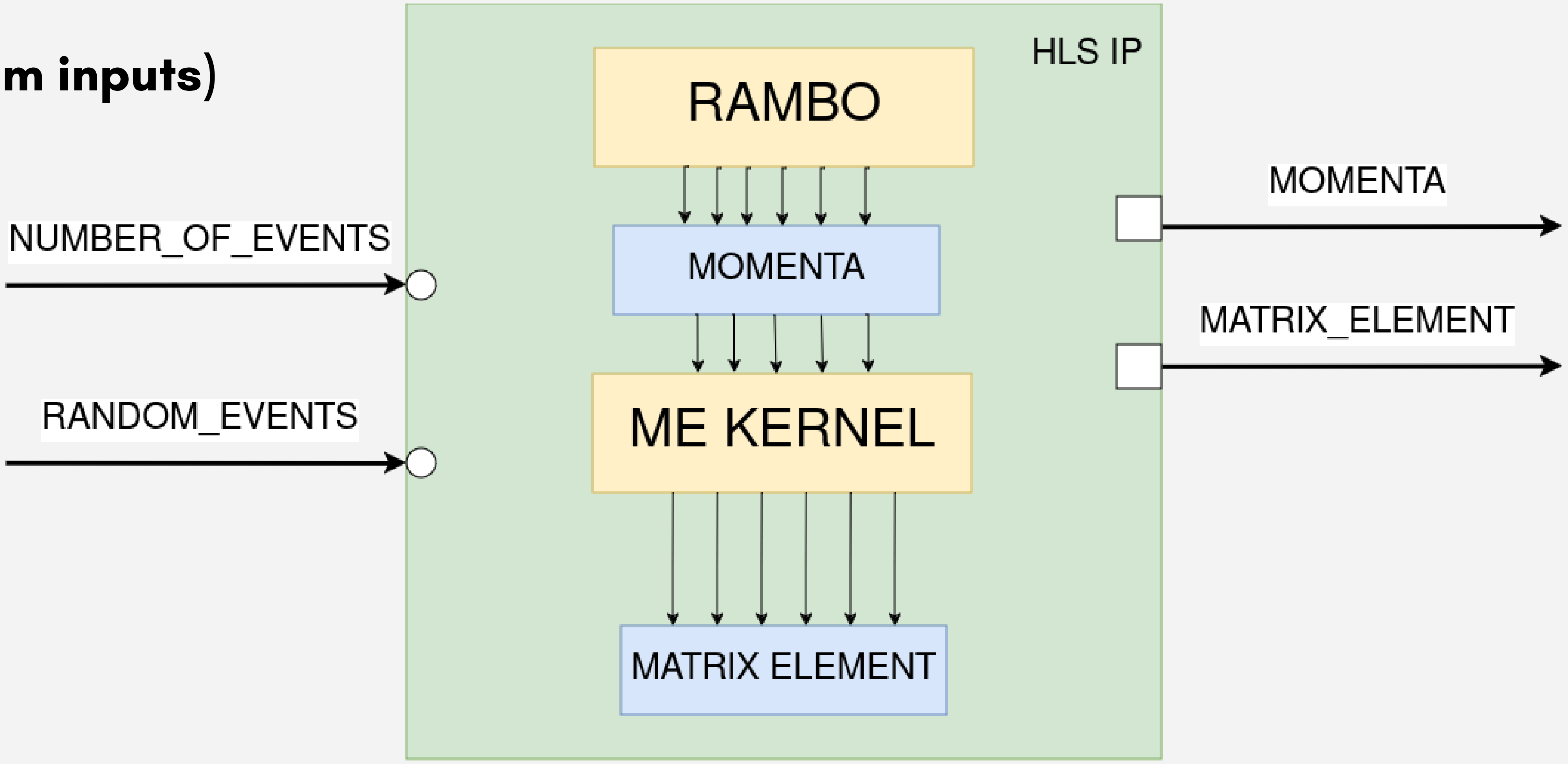
- FPGA application development is more complex, often using low-level languages like Verilog or VHDL.
- The Vitis environment allows using C/C++/OpenCL C to design functions (kernels).
- Kernels are automatically converted into RTL using High-Level Synthesis (HLS).
- Once RTL is generated, Vitis manages:
  - Synthesis
  - Mapping
  - Creation of the bitstream (packaged in an xclbin file) to program the FPGA.

- Developing applications for Alveo involves two parts:
  - Programming the host (runs on x86 processors).
  - Programming the FPGA (accelerates specific functions).
- Host development is similar to regular software development.
- C/C++ and the OpenCL API are used to:
  - Manage tasks on the FPGA
  - Transfer Data
  - PRogram the FPGA in real-time, optimizing its resources

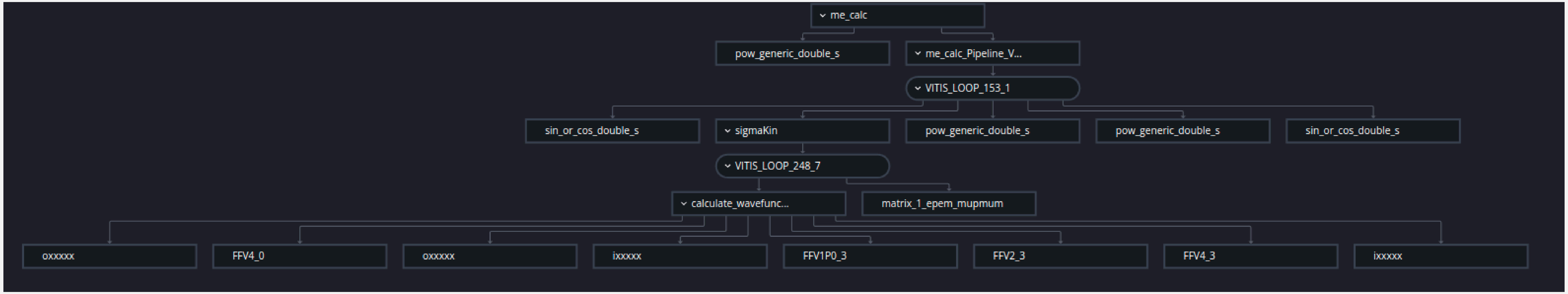
MADGRAPH FPGA

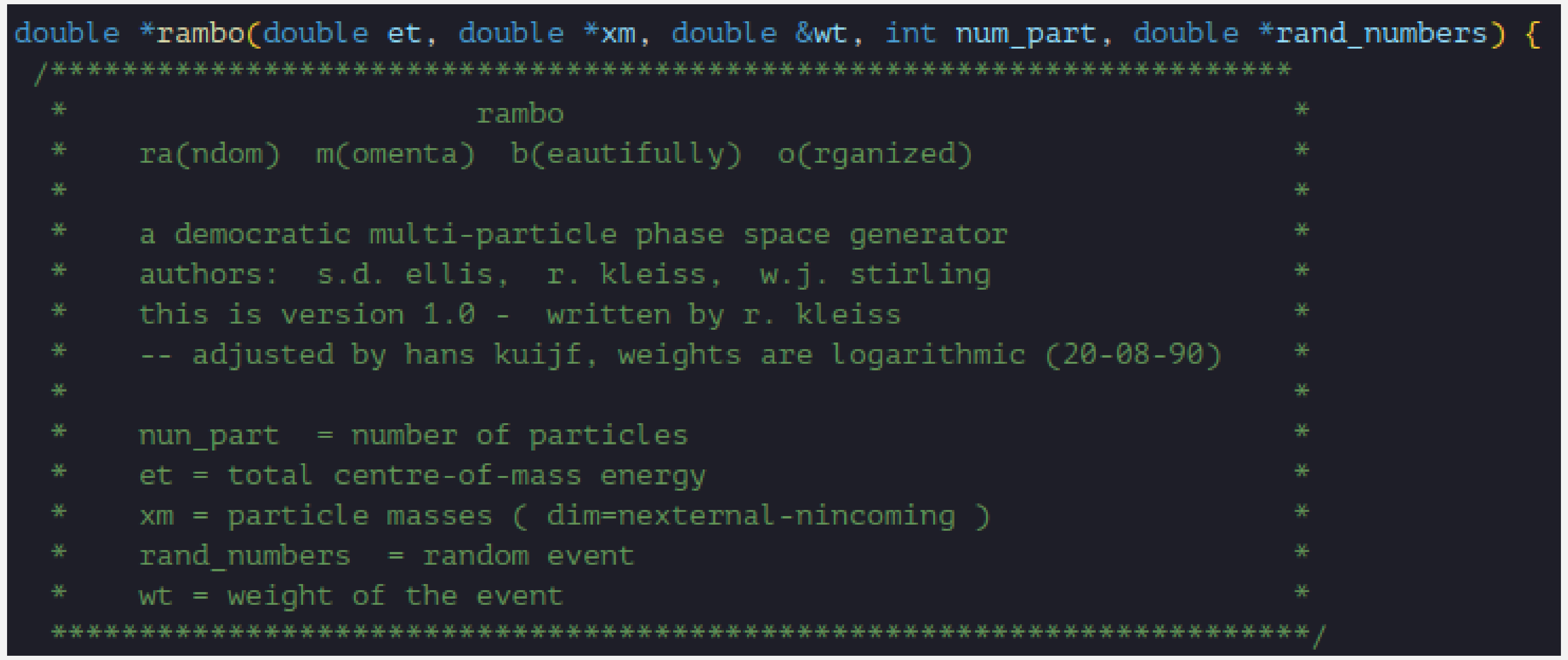
Process: e- e+ > u- u+

- top -> check_sa.cpp (IN:NUMBER_OF_EVENTS, IN:RANDOM_EVENTS, OUT: MOMENTA, OUT:MATRIX_ELEMENT)
  - Rambo -> Momenta
  - SigmaKin -> Matrix Element
- Rambo(Energy, masses, weight, masses_size, random inputs)
    - Obtain Momenta
- SigmaKin
  - InitProc() -> SetIndependentCouplings
  - SetParameters & SetDependentcouplings
  - Calculate waveforms
  - Calculate matrix of the process
    - Obtain Matrix Element



MADGRAPH FPGA





```
double *rambo(double et, double *xm, double &wt, int num_part, double *rand_numbers) {
 /***********************************************************************
  *                        rambo                                         *
  *    ra(ndom)  m(omenta)  b(eautifully)  o(rganized)                   *
  *                                                                      *
  *    a democratic multi-particle phase space generator                 *
  *    authors:  s.d. ellis,  r. kleiss,  w.j. stirling                  *
  *    this is version 1.0 -  written by r. kleiss                       *
  *    -- adjusted by hans kuijf, weights are logarithmic (20-08-90)     *
  *                                                                      *
  *    nun_part  = number of particles                                   *
  *    et = total centre-of-mass energy                                  *
  *    xm = particle masses ( dim=nexternal-nincoming )                  *
  *    rand_numbers  = random event                                      *
  *    wt = weight of the event                                          *
  ***********************************************************************/
```

```
double CPPProcess::sigmaKin(double p16[16]) {
```

FPGA Resources

Resource	Utilization	Avalible	Utilization(%)
LUT	404715	1759631	23
FF	549021	3660140	15
DSP	5218	12424	42
BRAM	22	3280	0.6

Frecuency : 121.95 MHz

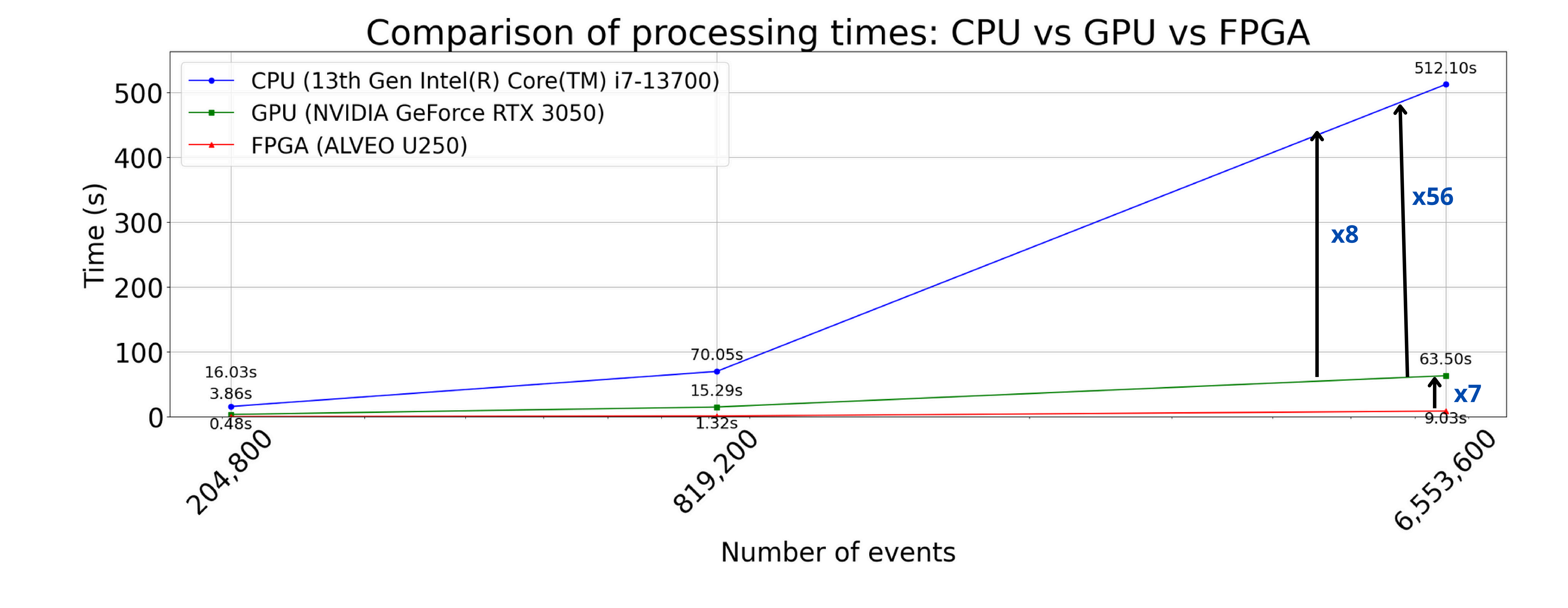
Setup (CPU + FPGA)

13TH GEN INTEL(R) CORE(TM) I7-13700H 2.40 GHZ

ALVEO U250

Results CPU vs GPU vs FPGA

Process: e- e+ > u- u+



Advantages and disadvantages of HLS for this application

- **Advantages:**
  - Increased Efficiency and Productivity
  - Better for prototyping
  - Reduce the development time
  - Enables developers to program FPGA in high-level languages
- **Disadvantages:**
  - Additional Translation Time
  - Slower Than coding RTL
  - Less control over memory resources

FUTURE IMPLEMENTATIONS

- Write the code in VHDL for the simple process
- Create the code for a complex process
- Create the new version for all LO processes
- Study the implementation of BSM processes and NLO
- Create a version that combines all three implementations (CPU + GPU + FPGA)

GOBIERNO
DE ESPAÑA

MINISTERIO
DE CIENCIA, INNOVACIÓN
Y UNIVERSIDADES

Financiado por
la Unión Europea
NextGenerationEU

Plan de Recuperación,
Transformación y
Resiliencia

AGENCIA
ESTATAL DE
INVESTIGACIÓN

"This work is supported by Ministerio de Ciencia, Innovación y Universidad con fondos Next Generation y del Plan de Recuperation, Transformacionales y Resiliencia (project -TED2021-130852B-100)"

IFIC
INSTITUT DE FÍSICA
CORPUSCULAR

CSIC
CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS

VNIVERSITAT
DE VALÈNCIA

HL

Porting MADGRAPH to FPGA

Héctor Gutiérrez [1], Luca Fiorini, Alberto Valero, Arantza Oyanguren, Francisco Hervas, Carlos Vico, Javier Fernandez, Santiago Folgueras, Pelayo Leguina

[1] Instituto de Física Corpuscular (CSIC-UV)

2nd Computing Challenges workshop (COMCHA)
October 2nd, 2024

Contact:
Hector.Gutierrez@ific.uv.es