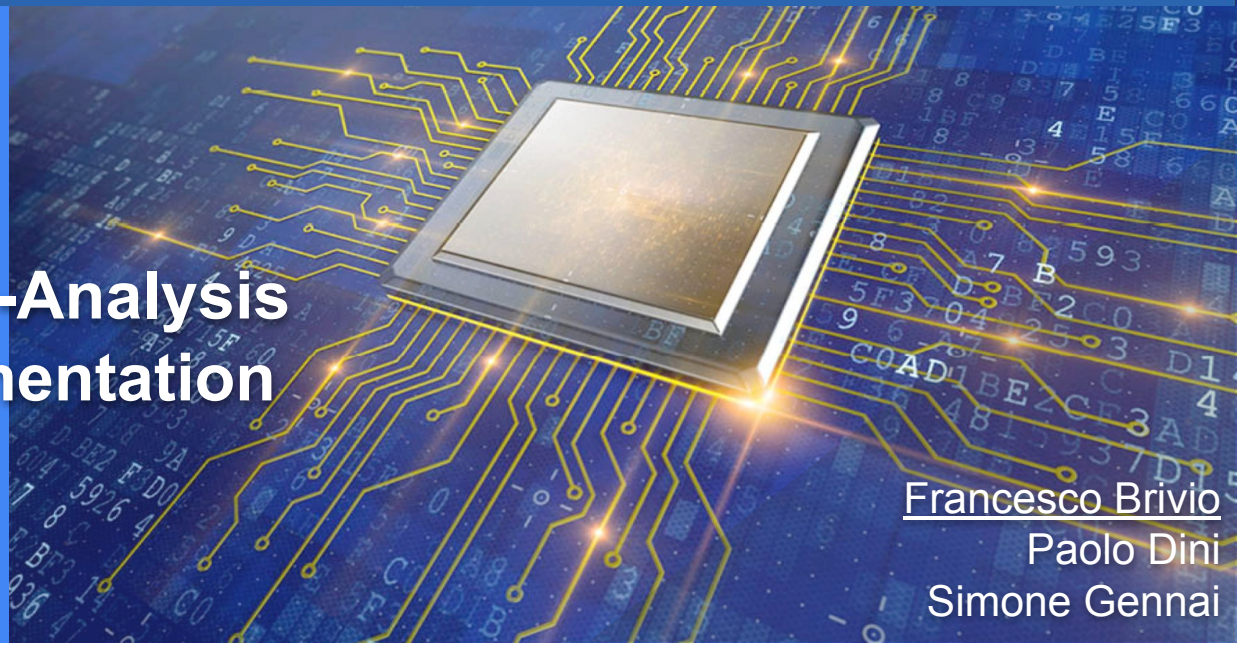
Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

$W \rightarrow 3\pi$ Scouting-Analysis Firmware Implementation



Francesco Brivio
Paolo Dini
Simone Gennai

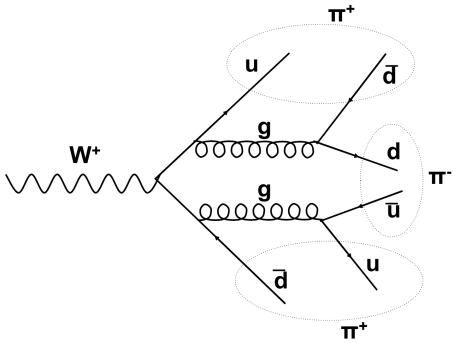
Bi-Weekly WP2 Meeting

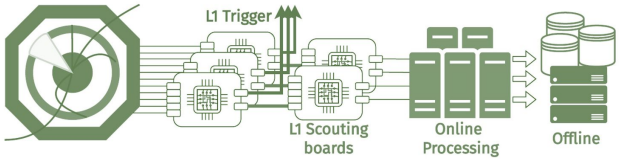
24 September 2024

UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA
INFN MILANO BICOCCA

Outline

- Spoke-2 HPC FPGA Bubbles
  - The Milano-Bicocca Cluster
  - FPGA and FPGA clusters use cases
- The $W \to 3\pi$ decay use case
  - Analysis flow
  - Firmware implementation
  - Preliminary estimates on S/B
  - Next steps
- Talks and other linked activities
- Summary



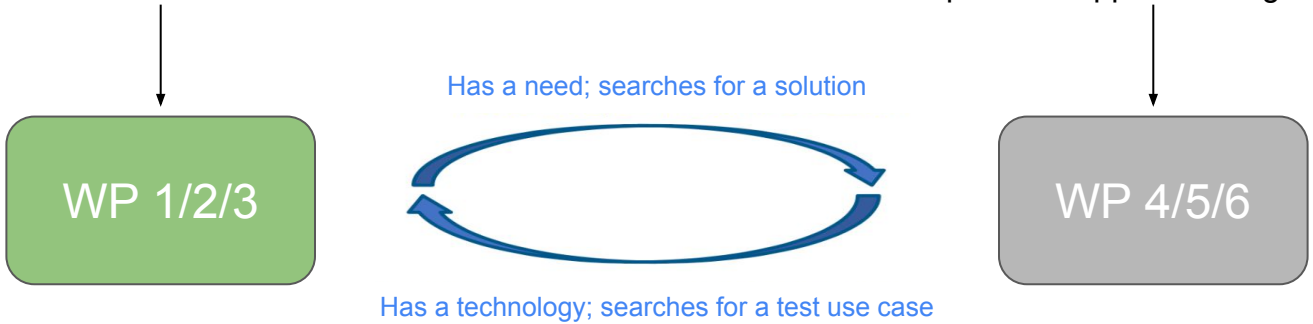


Spoke-2 WP2 ↔ WP4 Synergy

Project placed between the Working Packages 2 and 4

“Scientific” WPs: they analyze the needs of the (sub-)domain, and pose open problems for which advanced computing solutions are needed

“Technological” WPs: they harvest/investigate technological solutions in computing, on the infrastructure of the ICSC and beyond, and provide support/training for these.



- Ultra-fast ML inference
- High throughput analysis
- …

- FPGA boards and Clusters
- GPU Clusters
- …

HPC “*Bubbles*” & The Milano-Bicocca FPGA Cluster

- **HPC “Bubbles” Model**
  - Several distributed local clusters seen as opportunistic resources to be accessed on demand via the INFN Cloud infrastructure
  - Accessed through
    - PCI board via Virtual Machine
    - Direct access as local user to access the “full mesh” configuration of the boards (optical fiber cabling)
  - Milano-Bicocca Cluster
    - 16 FPGA boards (8 Xilinx + 8 Intel)
    - High-speed internal connection thanks to dedicated QSFP ports and breakout optical links

Possibility to realize a “full mesh” network following the LHCb RETINA project

Sito	Nodi CPU	Nodi GPU	Nodi FPGA	Nodi Storage
CNAF	26	30	4	52
BA	24	6	0	32
MI-BI	0	0	4	0
PI	8	0	0	0
TO	6	6	0	0
LNGS	0	6	0	12
NA	18	1	2	8
RM1	12	0	0	0
PD/LNL	10	6	0	0
LNF	20	6	0	6
CT	12	0	0	8
MI	4	0	0	0
TOTALE	**160**	**61**	**10**	**118**

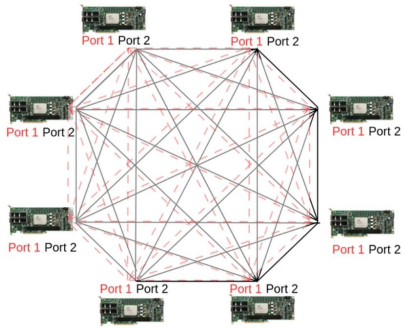


Figura 1. Esempio di rete full-mesh per le board Terasic

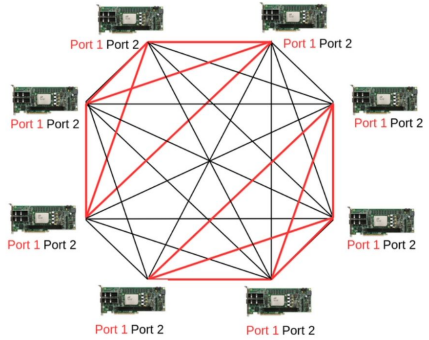
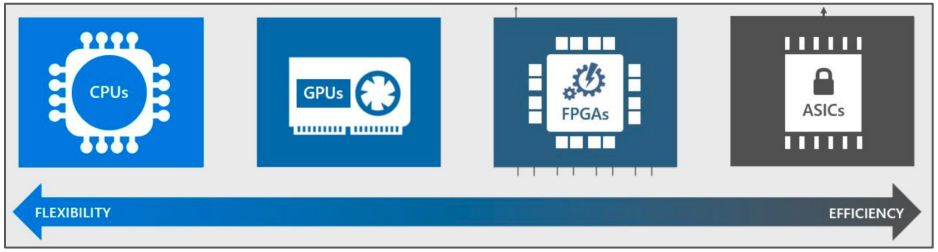


Figura 2. Esempio di rete full-mesh per board Xilinx U55C

Why FPGAs

- **F**ield **P**rogrammable **G**ate **A**rray (FPGA)
  - Re-configurable integrated circuits
  - High efficiency for a fairly good flexibility



- Advantages of FPGA
  - Easily reconfigurable
  - Deterministic latency: repeatable and predictable
  - High throughput (CMS Phase-2 L1 Trigger expected to processes 5% of total internet traffic)

  thanks to:
  - Resource parallelism: can run several algorithms in parallel
  - Pipeline parallelism: can accept new data at ~each clock

- Additionally
  - Commonly used in High Energy Physics experiments
  - Growing interest in private/industrial sector
    - Especially linked to ultra-fast inference for ML
  - Less power consumption with respect to GPUs

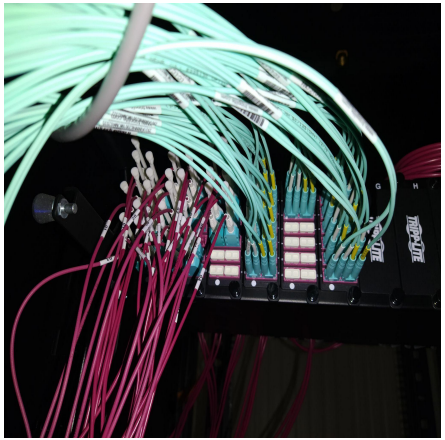
Accelerated ML for HLT Trigger

Type	Hardware	Accuracy	Inference Time	Max Throughput	Power	Throughput/W
CPU	2 x Xeon 2.1Ghz, 8 core	0.91	88 ms	11.36 img/s	~30W	0.38
FPGA	Arria 10 PAC - FP16	0.91	14 ms	84.44 img/s	~31W	2.72
	Arria 10 PAC - FP11	**0.88**	**7.5 ms**	**187.21 img/s**	**~31W**	**6.03**
GPU	GTX 1080	0.90	7.5 ms	192 img/s	~180W	1.06

Hamza Javed, Maurizio Perini,
Jennifer Ngadiuba, Vladimir Loncar

FPGA Cluster Use Cases

- **Several use cases planned for the MIB FPGA cluster**
  - Spread among different INFN sites and different experiments
  - **LHCb Experiment (MIB and Pisa)**
    - Mainly focussed on the RETINA project for online tracking
    - Comparison of performances in the cluster wit respect tp what currently installed in the LHCb VELO system
    - Development of the “full mesh” configuration to study performances in the second tracking detector (SciFi)
    - System expected to run in production already in Run4 with same FPGA boards
  - **INFN Perugia**
    - Development and testing of the BondMachine “distributed architecture”
  - **CMS (mostly MIB)**
    - Development of a Transformer for Tau reconstruction in the Level-1 trigger, including model distillation and porting on FPGA
    - CMS Phase-2 Scouting system

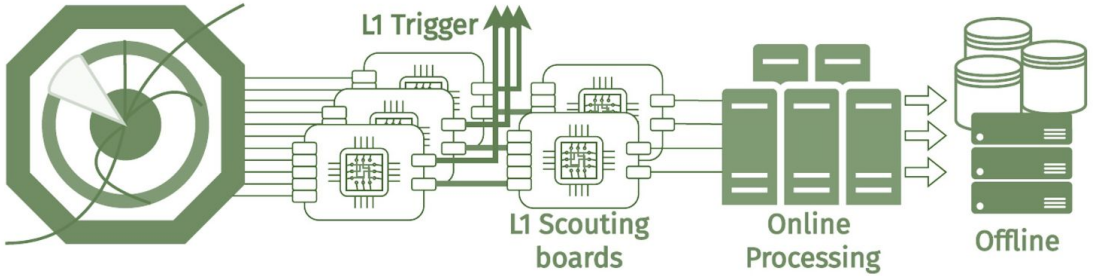


Example of “full mesh” configuration with optical links among different FPGA boards currently installed in Point 8 (LHCb)

The CMS Phase-2 Scouting System

The High-Luminosity Phase of the Large Hadron Collider will deliver data to the experiment with higher energy and especially at a **much higher luminosity**, with between 5 and 7.5 times the number of collisions with respect to the nominal LHC design

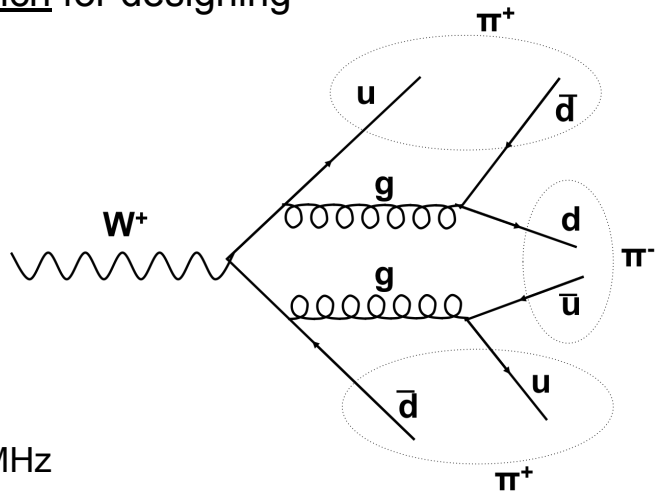
- All the experiments must exploit this huge amount of data as best as possible
- The **CMS experiment**, among the many upgrades, is developing an innovative trigger program based on the *Scouting System*
- ***40 MHz Scouting*** core idea
  - Acquire & analyze the L1 Trigger information for <u>all collisions</u>, happening at a rate of 40 MHz



  - **Target:** Look for signatures identifiable with just L1T info, that would evade the standard CMS *L1T* → *HLT* → *Offline* chain:
    - Too large “irreducible” backgrounds, e.g. narrow resonances of unknown mass
    - Signal identification requires algorithm that can’t fit the L1T constraints (combinatorics, complex NN…)
    - Signal identification requires time-correlation across several *Bunch-Crossings* (e.g. long-lived BSM)

The $W \to 3\pi$ Analysis

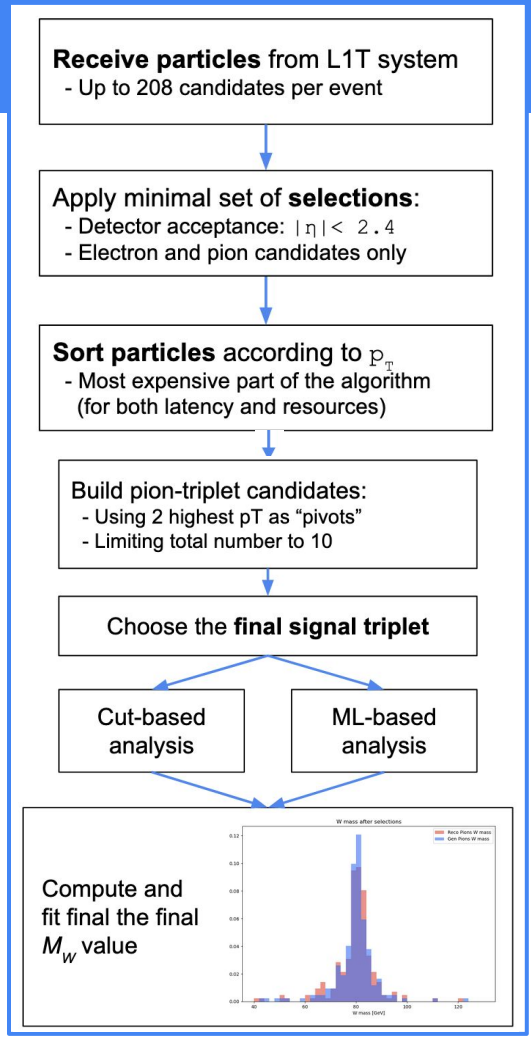
The search for the $W \to 3\pi$ decay represents an excellent test-bench for designing future Phase-2 scouting analyses



- **Extremely rare process** predicted by the Standard Model
  - Most recent (CMS) results can only set an upper limit on the branching ratio of BR < ~10^{-6} (PhysRevLett.122.151802)
  - The large statistics provided by scouting-data is well suited for studying such a rare process
- Idea is to design an **entirely FPGA-based analysis**
  - Full event processing from raw data up to the reconstructed M_W
  - Algorithm with ultra-low latencies to accommodate a rate of 40 MHz
- Practically
  - Exploit the ML-based approaches to improve selection efficiency
  - Translate analysis into firmware using state of the art libraries: *Vitis-HLS*, *hls4ml*, *Conifer*…
- Strong synergy with the PNRR ICSC Spoke-2 project on HPC-Bubbles with FPGA clusters
  - Clusters with both Xilinx and Intel boards distributed in several INFN sites and shared via Cloud

Analysis Flow

- Design is strictly linked to firmware implementation
  - Not all possible algorithms can be ported to firmware (without paying a price in latency or resources)
  - L1 scouting system offers a good opportunity to have dedicated boards and more relaxed timing constraints
    - The MIB FPGA cluster is a perfect system to benchmark and study feasibility of such analyses
- Analysis following the “standard” flow, but with some needed simplifications, e.g.
  - Looser selections
  - Avoiding combinatorial
  - Smart choice of input features for ML
  - …



Firmware Overview

Module	Latency Clocks	Latency Timing
masker	0	0 ns
slimmer2	0	0 ns
orderer7f	119	0.595 us
merger7f	17	85.000 ns
get_triplet_inputs	17	85.000 ns
decision_function_1	8	40.000 ns

Module	BRAM_18K	DSP	FF	LUT
masker	0	0	0	13104
slimmer2	0	0	0	10400
orderer7f	0	0	1774	46784
merger7f	0	0	44576	98207
get_triplet_inputs	2	9	508	2168
decision_function_1	0	0	19447	56669
Utilization (%)	~0	~0	4	27

- Overall view of the firmware implementation
  - For the (almost) full chain: from inputs to selecting the triplet with highest BDT score
  - Total Timing: 303 clocks (~1.5 μs)
    Total Resources: 27% of LUTs and 4% of FFs
- Firmware split in “kernels”
  - Following the analysis flow shown before
  - Each analysis step is run sequentially, but highly parallelized within the kernel itself
- Two approaches being followed
  - Latency-optimized → more resources
  - Resource-optimized → more latency

  (here only showing the second one for sake of time)

Firmware Overview: Pion Selections

Module	Latency Clocks	Latency Timing
masker	0	0 ns
slimmer2	0	0 ns
orderer7f	119	0.595 us
merger7f	17	85.000 ns
get_triplet_inputs	17	85.000 ns
decision_function_1	8	40.000 ns

Module	BRAM_18K	DSP	FF	LUT
masker	0	0	0	13104
slimmer2	0	0	0	10400
orderer7f	0	0	1774	46784
merger7f	0	0	44576	98207
get_triplet_inputs	2	9	508	2168
decision_function_1	0	0	19447	56669
Utilization (%)	~0	~0	4	27

Find the optimal selections allowed by the hardware, while maintaining highest possible gen-matching efficiency.

- Selections applied to pions are “minimal”
  - Detector acceptance: $|\eta| < 2.4$
  - ParticleID: only pions and electrons
  - No cut on p_T
- Selecting only the 10 highest p_T candidates

  (instead of all 208 possible) yields nonetheless to an **efficiency ~ 91%**
  - For a latency which is still below 1 clock

Gen-match efficiency of selecting # highest p_T reco pions

# Pions	Numerator	Denominator	Eff
3	6148	8910	**0.690**
4	7308	8910	**0.820**
5	7675	8910	**0.861**
6	7827	8910	**0.878**
7	7954	8910	**0.893**
8	8016	8910	**0.900**
9	8064	8910	**0.905**
10	8107	8910	**0.910**

Firmware Overview: Sorting

Module	Latency Clocks	Latency Timing
masker	0	0 ns
slimmer2	0	0 ns
orderer7f	119	0.595 us
merger7f	17	85.000 ns
get_triplet_inputs	17	85.000 ns
decision_function_1	8	40.000 ns

Module	BRAM_18K	DSP	FF	LUT
masker	0	0	0	13104
slimmer2	0	0	0	10400
orderer7f	0	0	1774	46784
merger7f	0	0	44576	98207
get_triplet_inputs	2	9	508	2168
decision_function_1	0	0	19447	56669
Utilization (%)	~0	~0	4	27

- Sorting all 208 possible candidates is the most expensive kernel
  - Several approaches tested (bubble-sort, iterative sorting…)
  - Chosen **bitonic sorting and merging**
    - Split candidates into 8 sub-arrays
    - Sort individually
    - Merge them maintaining order
- Nevertheless most of the latency and resources are consumed in this step
  - Taking up ~17% of LUTs tables and 2% of FFs
  - Most of the clocks

→ Further optimization ongoing!

Firmware Overview: Triplet Combinatorial

Module	Latency Clocks	Latency Timing
masker	0	0 ns
slimmer2	0	0 ns
orderer7f	119	0.595 us
merger7f	17	85.000 ns
get_triplet_inputs	17	85.000 ns
decision_function_1	8	40.000 ns

Module	BRAM_18K	DSP	FF	LUT
masker	0	0	0	13104
slimmer2	0	0	0	10400
orderer7f	0	0	1774	46784
merger7f	0	0	44576	98207
get_triplet_inputs	2	9	508	2168
decision_function_1	0	0	19447	56669
Utilization (%)	~0	~0	4	27

- To find the final “pions-triplet” candidate one should build all possible combinations of 208 pions… extremely inefficient (even in standard c++ code)!
  - Tested different options:
    - Reducing number of inputs
    - Fixing highest p_T candidate as “pivot”
    - Using 2 pivots
  - Optimal solution found:
    - Fix two candidates and consider only the 8 highest p_T triplets
    - Completely avoiding any combinatorial!

→ Reached **88.5% gen-matching efficiency**

  - Run ML inference only on the selected triplets to find the signal candidate

Firmware Overview: DNN for $W \rightarrow 3\pi$ identification

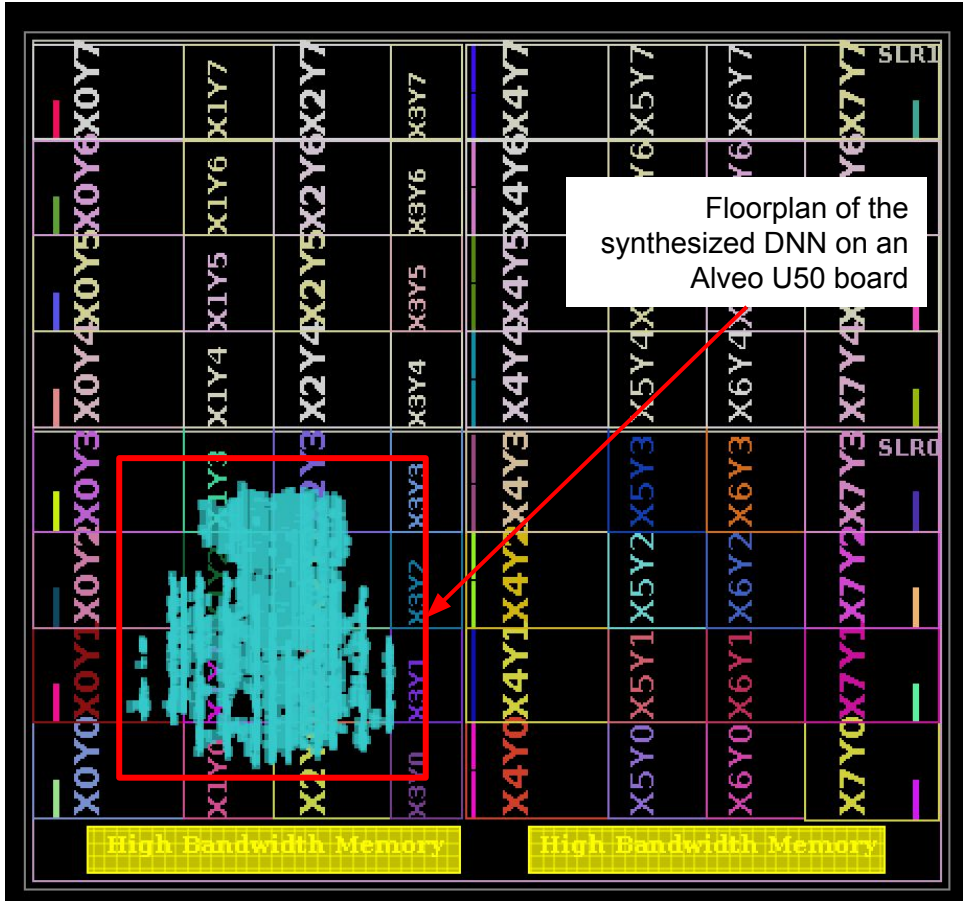
Different ML approaches being studied to identify the signal $W \rightarrow 3\pi$ candidate: DNN, BDT, multiclassification…

Example of a simple fully-connected DNN with:

- 5 layers with `[35-20-20-25-35]` neurons
- ReLu activation functions, sigmoid for output
- 23 input features
- Trained on $W \rightarrow 3\pi$ and MinBias events
- Model trained with TensorFlow/Keras

Firmware implementation:

- Quantization, pruning and synthesis performed with [hls4ml](hls4ml) library



Floorplan of the synthesized DNN on an Alveo U50 board

Signal & Background Selection Efficiencies

Comparison of different approaches used to select the events and the $W \to 3\pi$ candidate

- Cut-based: tight cuts to select only 1 triplet in the event → reject all others
- ML-based: run DNN/BDT inference on few (O(10)) triplet per event → keep only if `score > threshold`

	W->3p	N evts	%	N_S	SingleNu	N evts	%	N_B	N_S/N_B
	File evts	50400	-	-	File evts	1941240	-	-	-
Analysis									
Cut-based	-	3852	0,076	626,7	-	1	5,2E-07	8,7E+07	7,24E-06
Pruned DNN	DNN > 0.987	5303	0,105	862,8	DNN > 0.987	1	5,2E-07	8,7E+07	9,97E-06
BDT	BDT > 0.964	7320	0,145	1191,0	BDT > 0.964	1	5,2E-07	8,7E+07	1,38E-05

Expected number of events obtained assuming 400 fb^{-1} of data collected in ~1 year in Phase-2 by CMS

ML approach, if optimized, shows best performance in terms of expected S/B ratio by a factor 2!

- Crucial to correctly implement in firmware (quantization of weights, input variables, scores…)

Firmware Developments: Next Steps

- Next steps in the firmware development:
  - Finish optimization of algorithms and kernels
  - Test final implementation (including P&R) on different boards
  - Currently collaborating with:
    - CERN CMS L1 group → mainly for the “scouting” part
    - University of Colorado Boulder → mainly for algorithms and optimization
      - They are developing a similar approach (for the L1T system) based on the CMS Phase-2 Track Trigger

- Firmware testing
  - Organize a test within the CMS Phase-2 Demonstrator system
  - Test on the MIB FPGA cluster using a “multi-board approach”:
    - Generate a stream of $W \rightarrow 3\pi$ events on one board
      - MC code for generation already existing, to be adapted for our system
    - Transmit data to a different dedicated board
    - Decode and analyze events

Talks and Linked Activities

- **Talks on the Topic**
  - Within CMS
    - "W→3pions studies" - L1 Scouting Weekly meeting (Oct. 2023)
    - "Phase 2 Scouting W3Pi performance" - Level-1 DPG meeting (Jan. 2024)
    - "Phase-II W->3Pi scouting analysis" - Level-1 DPG meeting (Apr. 2024)
  - Within Spoke-2
    - "The Milano-Bicocca FPGA Cluster" - Spoke-2 Annual meeting (Dec. 2023)
  - Other
    - "Analysis of the $W \to 3\pi$ decay as a use case for scouting--analyses on FPGA" - SIF 2024 (Sept. 2024)

- **Other Activities**
  - Development and improvements of one of the "*c++ → firmware*" backends of the Conifer package
    - Included in the official release since version v1.5

Conifer version	ScorePrecision	vsynth LUT	vsynth FF	Latency
`master`	`ap_fixed<11,4,AP_RND_CONV,AP_SAT>`	51936	5032	133
This PR	`ap_fixed<11,4,AP_RND_CONV,AP_SAT>`	51163	4489	9
--------	--------	--------	--------	--------
`master`	`ap_fixed<11,4,AP_RND_CONV>`	43329	3542	4
This PR	`ap_fixed<11,4,AP_RND_CONV>`	42988	3800	4

Courses in cooperation with Working Package 5

One of the Working Package 5 milestones is the “Educational KPI”:

> *“Organizing courses about FPGA programming on low and high level”*

Two courses organized so far:

- **Introductory course to HLS FPGA programming (Agenda)**
  - 27-30 November 2023
  - Introduction to high-level FPGA programming (HLS + Vivado) and ML on FPGA
  - Hands-on exercises on Virtual Machines provided by CNAF
- **Introductory course to VHDL (Agenda)**
  - 4-6 March 2024
  - Introduction to low-level FPGA programming (VHDL)
  - Hands-on exercises on FPGA boards provided by INFN & University of Padova (A. Triossi)

Coming Next:

- **New courses on high-level FPGA programming are being organized/planned**
  - Dates not yet scheduled

Summary

- The Spoke-2 HPC FPGA Bubbles project provides an optimal system to test and develop new algorithms and technologies for the HL-LHC phase
  - The MIB Cluster will consist of 8 Xilinx and 8 Intel FPGA boards, also in “full-mesh” configuration
    - Will be available via INFN Cloud and “direct connection”
  - Involvement from different INFN sites and different LHC experiment
  - Quite large delays in orders and delivery
    - Hopefully we should receive the hardware before the end of this year

- The analysis of the $W \rightarrow 3\pi$ decay represents a perfect use case for designing and studying how to better exploit HPC Clusters
  - A preliminary analysis has been designed to run fully on FPGA, optimizing selections for hardware limitations and exploiting ML approach to maximise sensitivity
  - More optimization of algorithms and firmware implementation is ongoing
  - Next step is to test in different conditions (CMS Demonstrator, MIB FPGA CLuster)

Backup

The FPGA Cluster in Milano-Bicocca

- **Server:**
  4 server 4U format, 4 slot pci (pci4 o pci5):
  - 2 server with 4 board Xilinx U55C each
  - 2 server with 4 board Terasic DE10-Agilex Dev. Board each

→ For a total of 16 FPGAs

- **Storage:**
  - Uno of the U55C server equipped with 10 TB in Raid5 (SSD NVMe)
  - Uno of the Terasic server equipped with 10 TB in Raid5 (SSD NVMe)
- **Network:**
  - Two 10 Gbps ethernet ports for each server
- **Connections and "topology"**
  - Board Xilinx → 2 ports at 100G (QSFP28)
  - Board Terasic → 2 ports at 200G (QSFP-DD)
- **Network Hardware:**
  - Breakout cables (MPO-LC)
  - LC-LC connectors
  - Patch panel for LC-LC connectors

Possibility to realize a "full mesh" network using breakout cables following the LHCb RETINA project

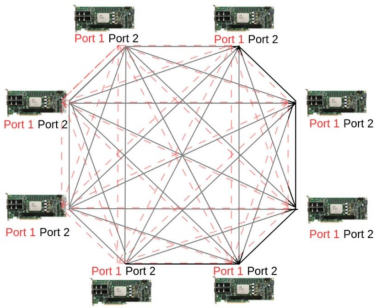


Figura 1. Esempio di rete full-mesh per le board Terasic

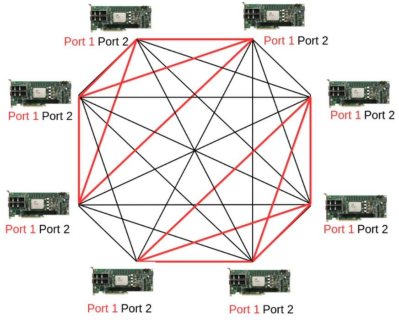


Figura 2. Esempio di rete full-mesh per board Xilinx U55C

MIB FPGA Cluster Details

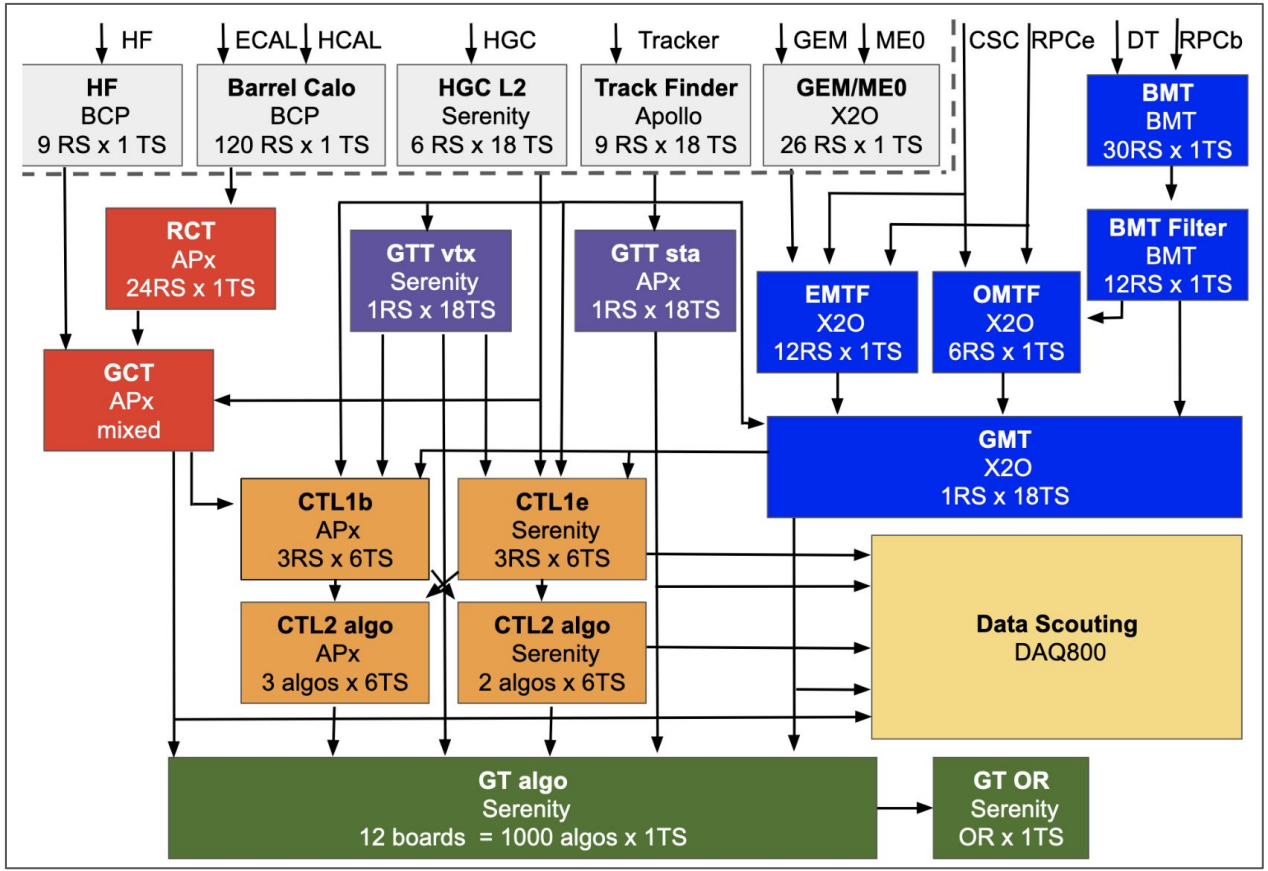
Appalto specifico Terabit per MIB

	MIB				
	oggetto	**quantita'**	**fondi**	**totale**	**note**
	L1FPGA_S	2	terabit	62,038.00 €	
Lotto1	L1FPGA_X	8	terabit	31,992.00 €	
FPGA	L1QSFP28	16	terabit	3,296.00 €	
xilinx	L1CBLMPO12	16	terabit	1,472.00 €	
e cavi	L1LCLC	2	terabit	177.00 €	Basta un solo pannello, ne inseriamo uno in piu
	L1ETHCBL2	4	terabit	28.00 €	per connettivita' eth dei server
	L1FPGA_S	2	terabit	62,038.00 €	
Lotto1	L1FPGA_T	8	terabit	73,608.00 €	
FPGA	L1QSFPDD	16	terabit	25,872.00 €	
terasic	L1CBLMPO24	16	terabit	1,472.00 €	
e cavi	L1LCLC	2	terabit	177.00 €	
	L1ETHCBL2	4	terabit	28.00 €	per connettivita' eth dei server
				262,198.00 €	**iva esclusa**
				319,881.56 €	**iva inclusa**
				319,881.56 €	**terabit iva inclusa**
				- €	**dare iva inclusa**
				319,881.56 €	**Totale acquisti su lotto 1 iva inclusa**
				- €	**Totale acquisti su lotto 2 iva inclusa**

P. Dini

Similar model has been “re-used” for the other Spoke-2 HPC FPGA Bubbles

The CMS Phase-2 L1 Trigger System



https://cds.cern.ch/record/2714892

Evaluation of Expected Number of Events

To evaluate the number of expected signal ($W \to 3\pi$) and background ("MinBias") events we use:

- **Signal** $\rightarrow$ $N_{Sign} = \sigma_{W\ prod} \cdot BR_{W \to 3\pi} \cdot IntLum \cdot \epsilon$
  - where:
    - $\sigma_{W\ prod} = 2.05 \cdot 10^{8}$ fb
    - $BR_{W \to 3\pi} = 10^{-7}$
    - IntLum = 400 fb^{-1}
    - ϵ = signal selection efficiency
- **Background** $\rightarrow$ $N_{Bkg} = R_{MB} \cdot T \cdot F$
  - where:
    - R_{MB} = 31.5 MHz (MinBias rate)
    - T = 400 fb-1 / 7.5x10^{34} = 5.33x10^{6} sec
      = 61 days (data-taking period)
    - F = background selection efficiency

Triplet Efficiencies Comparison

Gen-match efficiency of selecting # reco triplets:

- 1st+2nd pivots → 8 triplets
- 1st+3rd pivots → 7 triplets
- Total: 10 candidates → 15 triplets total

# Triplets	Numerator	Denominator	Ratio
1	6148	8910	**0.690**
2	6978	8910	**0.783**
3	7255	8910	**0.814**
4	7381	8910	**0.828**
5	7489	8910	**0.841**
6	7538	8910	**0.846**
7	7575	8910	**0.850**
8	7608	8910	**0.854**
9	7856	8910	**0.882**
10	7912	8910	**0.888**
11	7920	8910	**0.889**
12	7931	8910	**0.890**
13	7936	8910	**0.891**
14	7940	8910	**0.891**
15	7945	8910	**0.892**

# Triplets	Numerator	Denominator	Ratio
1	6148	8910	**0.690**
2	6978	8910	**0.783**
3	7255	8910	**0.814**
4	7381	8910	**0.828**
5	7489	8910	**0.841**
6	7737	8910	**0.868**
7	7793	8910	**0.875**
8	7857	8910	**0.884**

Optimal Triplets chosen
- 5 triplets from 1st+2nd pivots
- 2 triplets from 1st+3rd pivots
- 1 triplet from 2nd+3rd pivots

Firmware Synthesis: Timing Estimates

+ Timing:
  * Summary:

Clock	Target	Estimated	Uncertainty
ap_clk	5.00 ns	3.926 ns	1.35 ns

+ Latency:
  * Summary:

Latency (cycles)		Latency (absolute)		Interval		Pipeline
min	max	min	max	min	max	Type
303	303	1.515 us	1.515 us	304	304	no

  + Detail:
    * Instance:

Instance	Module	Latency (cycles)		Latency (absolute)		Interval		Pipeline
		min	max	min	max	min	max	Type
masked_masker_fu_2305	masker	0	0	0 ns	0 ns	0	0	no
call_ret_slimmer2_fu_2517	slimmer2	0	0	0 ns	0 ns	0	0	no
grp_orderer7f_fu_2939	orderer7f	119	119	0.595 us	0.595 us	119	119	no
grp_merger7f_fu_4243	merger7f	17	17	85.000 ns	85.000 ns	17	17	no
grp_get_triplet_inputs_fu_4377	get_triplet_inputs	17	17	85.000 ns	85.000 ns	17	17	no
grp_decision_function_1_fu_4422	decision_function_1	8	8	40.000 ns	40.000 ns	8	8	no

Firmware Synthesis: Resource Occupancy

* Summary:

Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	605	-
FIFO	-	-	-	-	-
Instance	2	9	66305	227332	-
Memory	0	-	1768	1898	0
Multiplexer	-	-	-	9923	-
Register	-	-	11041	-	-
Total	2	9	79114	239758	0
Available SLR	1344	2976	871680	435840	320
Utilization SLR (%)	~0	~0	9	55	0
Available	2688	5952	1743360	871680	640
Utilization (%)	~0	~0	4	27	0

+ Detail:

* Instance:

Instance	Module	BRAM_18K	DSP	FF	LUT	URAM
grp_decision_function_1_fu_4422	decision_function_1	0	0	19447	56669	0
grp_get_triplet_inputs_fu_4377	get_triplet_inputs	2	9	508	2168	0
masked_masker_fu_2305	masker	0	0	0	13104	0
grp_merger7f_fu_4243	merger7f	0	0	44576	98207	0
grp_orderer7f_fu_2939	orderer7f	0	0	1774	46784	0
call_ret_slimmer2_fu_2517	slimmer2	0	0	0	10400	0
Total		2	9	66305	227332	0

Firmware Implementation: Latency Optimized

Preliminary example of latency-optimized firmware implementation

Latency-optimized

Fastest possible processing,
large usage of resources

Latency (cycles)		Latency (absolute)	
min	max	min	max
104	104	0.520 us	0.520 us

Module	BRAM18K	DSP	FF	LUT	URAM
masker	0	0	0	13104	0
slimmer	0	0	0	10400	0
orderer	0	0	31616	157981	0
merger	0	0	24715	167638	0
get_triplet_inputs	0	6	302	1514	0
decision_function_1	0	0	19067	56669	0
Total	0	6	75700	407306	0
Utilization (%)	0	~0	5	46	0

- Extremely reduced latency:
  - Around 0.5 microseconds for the full chain
    - from receiving inputs
    - to running inference on the triplet candidates
- Firmware (pre-implementation step) takes up almost 50% of the resources of the full board (Alveo U50 board)
  - Sorting kernel is the most expensive:
    - Alone occupying ~37% of available LUTs
    - Based on bitonic sorting and merging

Firmware Implementation: Resources Optimized

Preliminary example of resources-optimized firmware implementation

Resources-optimized

Reduce resource consumption to the detriment of total latency

Latency (cycles)		Latency (absolute)	
min	max	min	max
228	228	1.140 us	1.140 us

Module	BRAM18K	DSP	FF	LUT	URAM
masker	0	0	0	13104	0
slimmer	0	0	0	10400	0
orderer	0	0	1774	46784	0
merger	0	0	45234	99666	0
get_triplet_inputs	0	6	326	1642	0
decision_function_1	0	0	19067	56669	0
Total	0	6	66401	228265	0
Utilization (%)	0	~0	4	27	0

- By removing some of the parallelization, one can reduce largely the resource consumption
  - *E.g.* by removing the `#pragma inline` directive from some of the recursive methods used in the sorting process
  - Forcing some kernels to run in sequence rather than in parallel
- Resource consumption greatly reduced
  - LUTs down from 46% to 27%
  - For a price in overall latency: now around 1.1 μs

→ Note that this implementation is still preliminary and more optimization is certainly possible! (e.g. with proper function pipelining, etc…)