

CHEP 2016 - October 13 - San Francisco

# An “artificial retina” processor for track reconstruction at the LHC crossing rate

Simone Stracka  
Universita` and INFN Pisa

F. Bedeschi<sup>\*</sup>, R. Cenci<sup>\*</sup>, P. Marino<sup>\*</sup>, M. J. Morello<sup>\*</sup>, D. Ninci<sup>\*</sup>, A. Piucci<sup>\*</sup>, G. Punzi<sup>\*</sup>, L. Ristori<sup>§</sup>,  
F. Spinella<sup>\*</sup>, S. Stracka<sup>\*</sup>, D. Tonelli<sup>¶</sup>, and J. Walsh<sup>\*</sup>

<sup>\*</sup> Universita`, INFN, Scuola Normale Superiore Pisa

<sup>§</sup> Fermi National Accelerator Laboratory

<sup>¶</sup> INFN Trieste

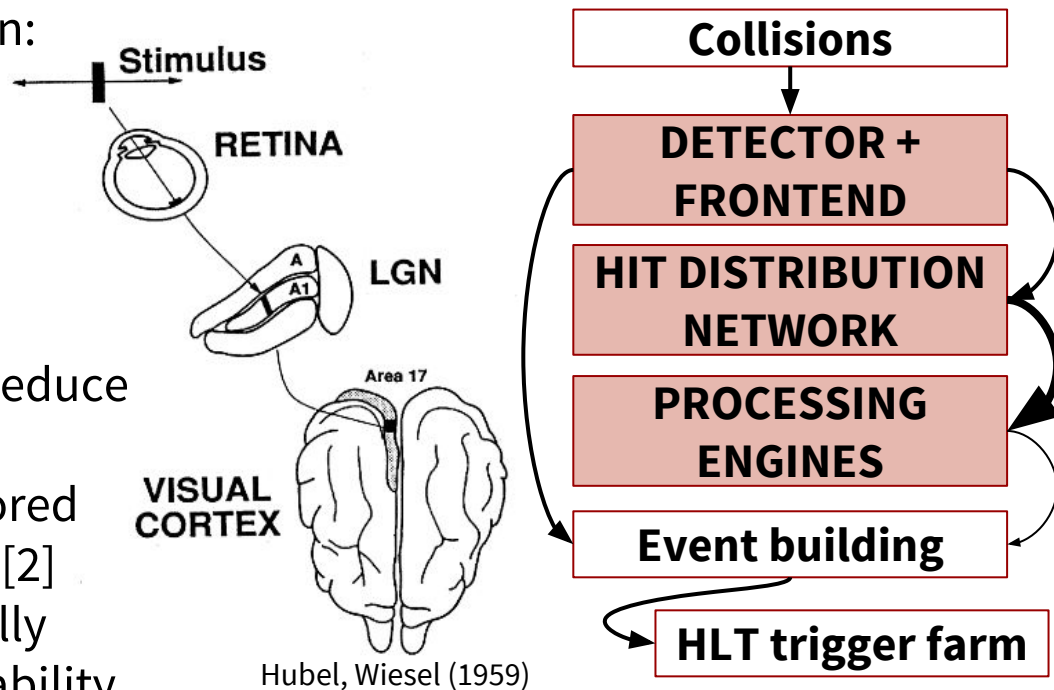
# Artificial retina algorithm

Inspired to early vision [1] and based on:

- Voting scheme for patterns
- Template matching

Features:

- **Analog response** + interpolation reduce number of neurons/patterns
- **Resource optimization**: select stored patterns to maximize information [2]
- **Local connectivity**: exploit spatially local correlations to increase scalability
- **Parallelism**: suitable for FPGA implementation



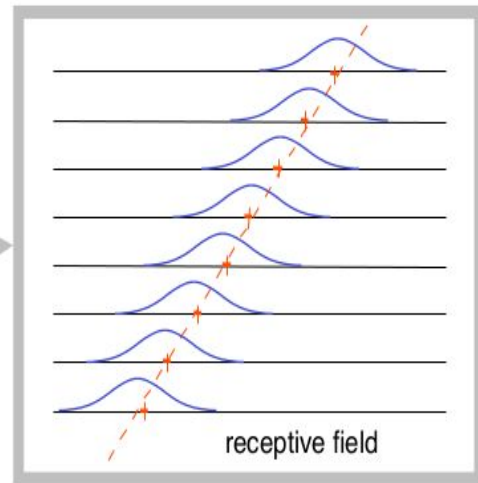
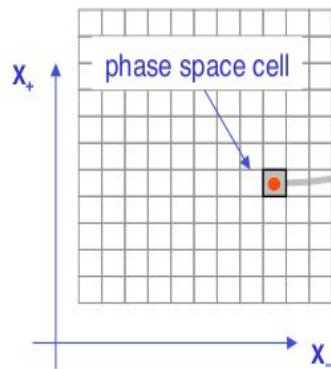
[1] Ristori L, NIM A453, 425 (2000)

[2] Del Viva MM et al, PLoS ONE 8(7): e69154 (2013)

# The track reconstruction algorithm

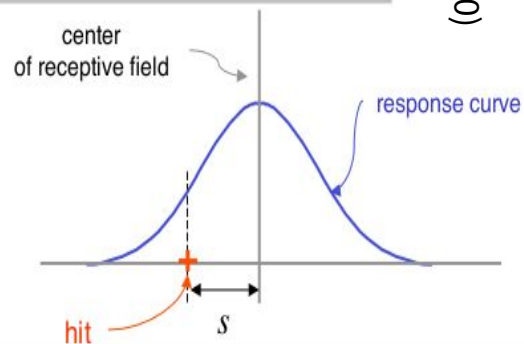
1. The space of relevant **template patterns** / tracks is **encoded into the device**
2. Template-space cells are **routed only to the relevant detector elements**
3. An **analog voting scheme** is executed in **parallel** for each cell in processing engines
4. Tracks are identified as **local maxima**, using interpolation for increased precision

Center of receptive field corresponds to center of phase space cell



Response of each cell is summed over all hits

$$R = \sum_{\text{all hits}} e^{-\frac{s_i^2}{2\sigma^2}}$$



# Motivation

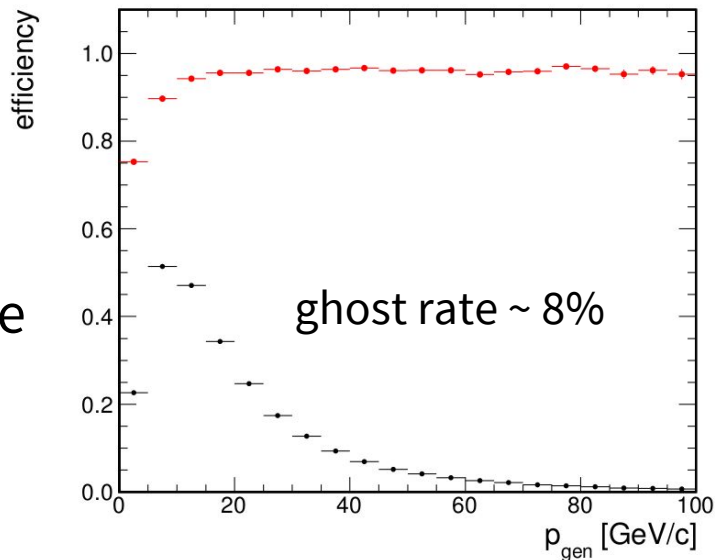
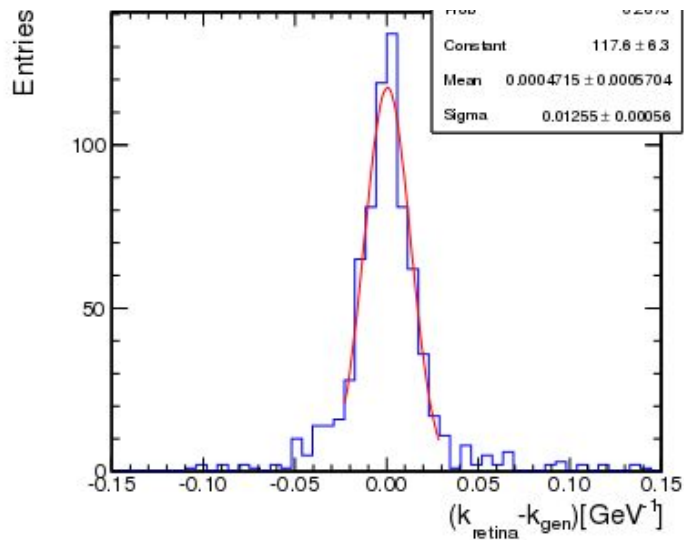
Computing and storage demands of future (HL-)LHC experiments represent a challenge: need a more efficient and scalable usage of available hardware

- More processing will have to be performed only once (“online”)
- Performing a repetitive task on FPGA frees CPU resources for higher-level tasks
- FPGAs allow for low latency (sub- $\mu$ s): acts as a “track-detector”. The event primitives are immediately available to Event-Building and High-Level-Trigger farms

# High level simulation (1)

The retina track reconstruction was simulated for two case studies:

1. 3D track reconstruction using the LHCb vertex locator + 2 tracking stations in the fringe field of the magnet



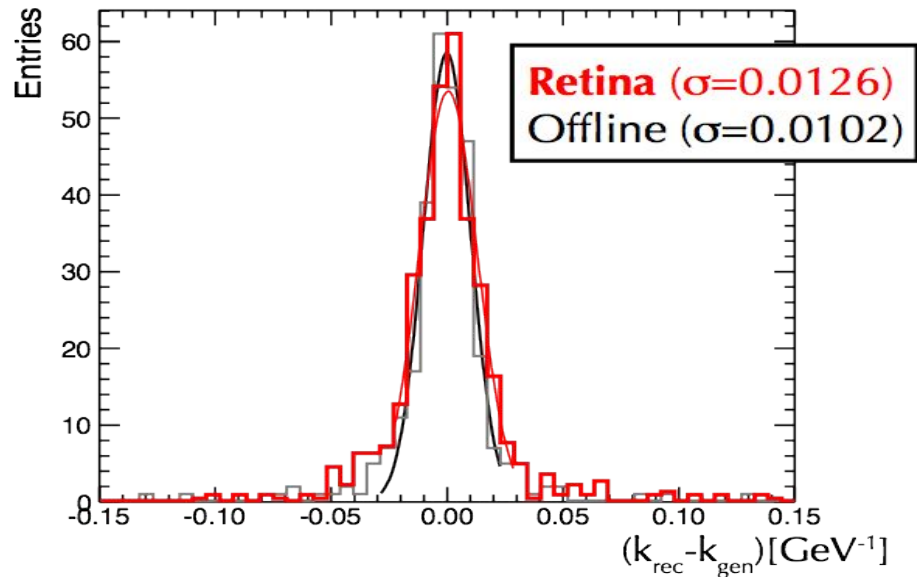
With a system of 50000 cells (50 Stratix V FPGA), one could achieve O(100) MHz retina tracks / FPGA at a **reasonable cost**

# High level simulation (2)

2. Reconstruction of 2D track segments in a small 6-layer silicon strip telescope, located after a bending magnet

Curvature can be measured with O(%) precision with 3000 cells

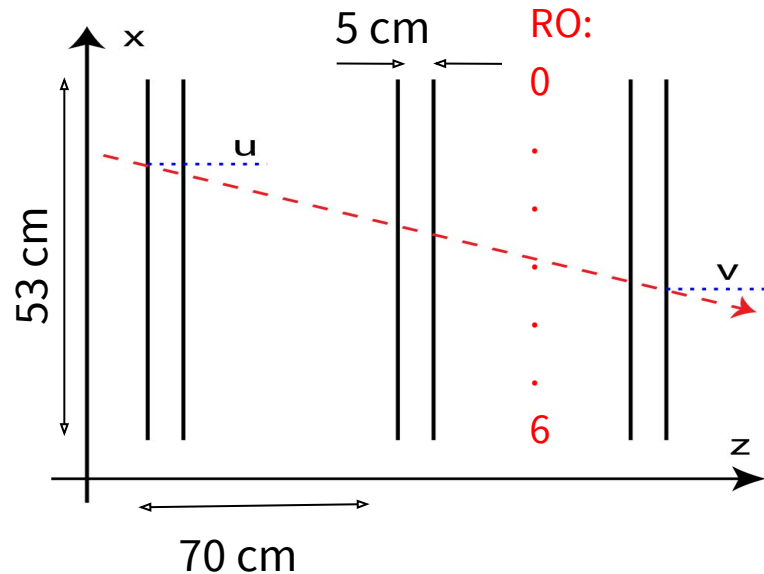
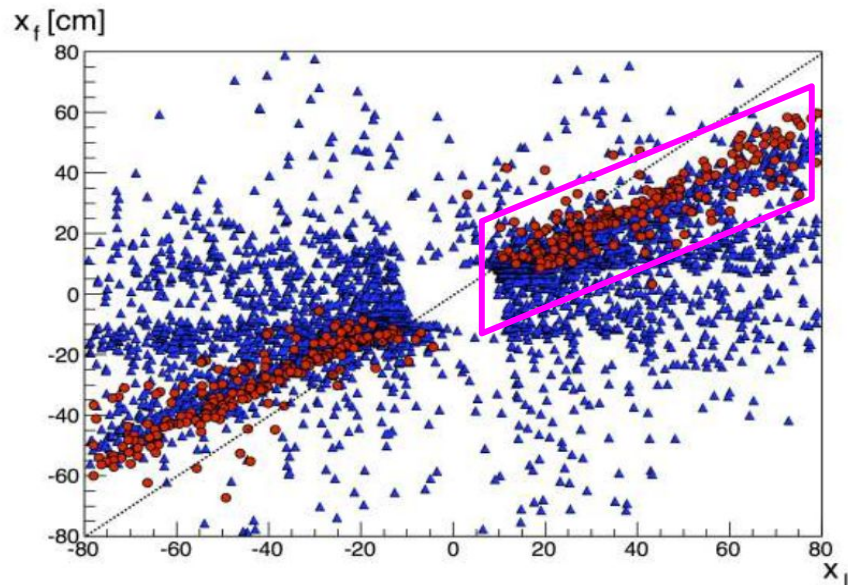
Smaller-sized system for functionality tests using existing FPGA boards



# 2D tracking case study

Prompt tracks from collision (●) (10 per event)

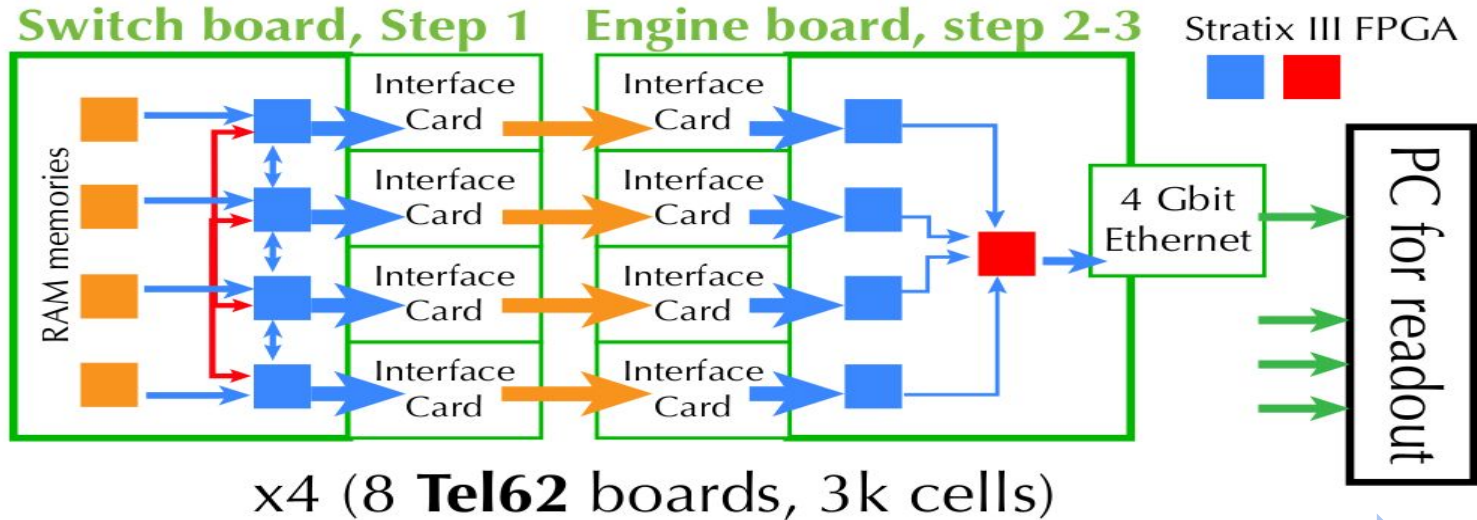
Tracks originating elsewhere (▲)



6 layers:  $53 \times 22 \text{ cm}^2$ ,  $200 \mu\text{m}$  pitch  
140 hits/event on average ( $<250$  @90%)

7 independent readout modules/layer  
Two nearby layers constitute a **doublet**

# Implementation

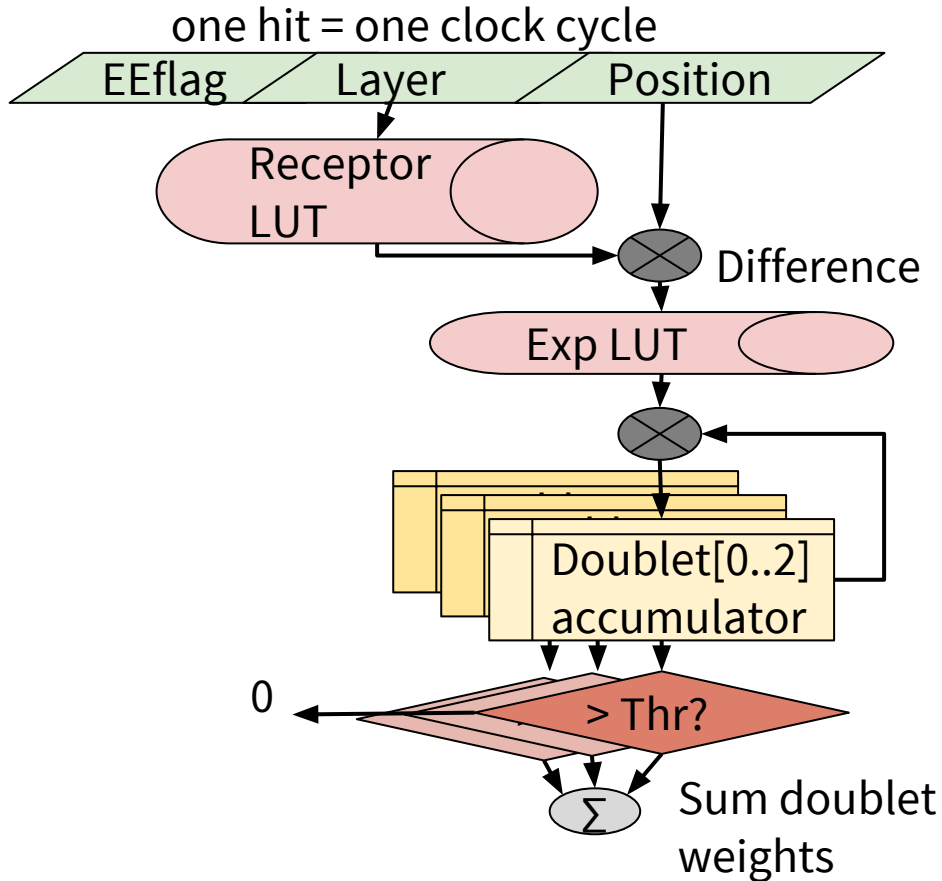


Bandwidth: **X~0(Gbps)** **4X** **X/2**

Paired configuration of TEL 62 boards (Stratix III, 65nm, used by NA62)  
Hit sequences pre-loaded in embedded RAM blocks (x16) and injected in the switch - all input hit sequences terminated by an End-Event



# Pipelined hit processing



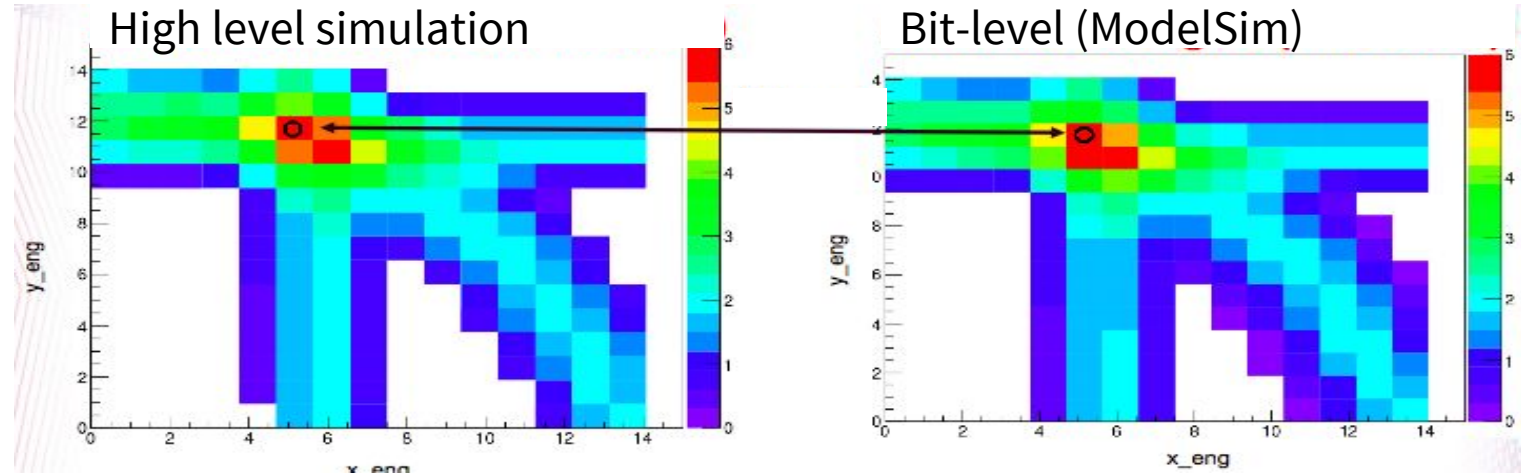
Intersections of templates with detector layers stored in look-up tables: **easy to update**

Upon end-event, sum doublets' weights that are above threshold (THR\_DB), and reset accumulators

Engines include the logic to check for local maxima: if sum is above threshold (THR\_S), compare value with first neighbors

Each FPGA can host 16x15 cells, using 90% of the resources

# Simulation of the device



The latency is  $\sim 120$  clock cycles ( $< 1 \mu s$ ), from ModelSim simulation

Simulated switching network, fed with realistic events (up to 250 hits/event), delivers  $< 90$  hits (@ 90%) to any engine

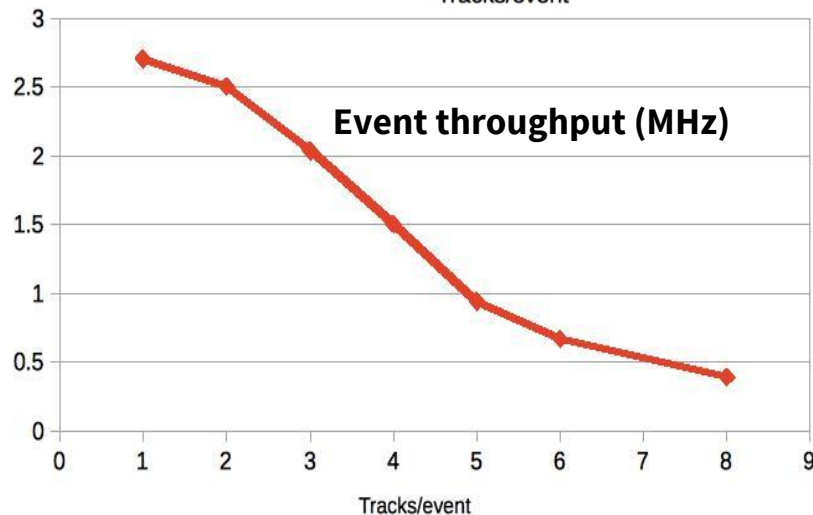
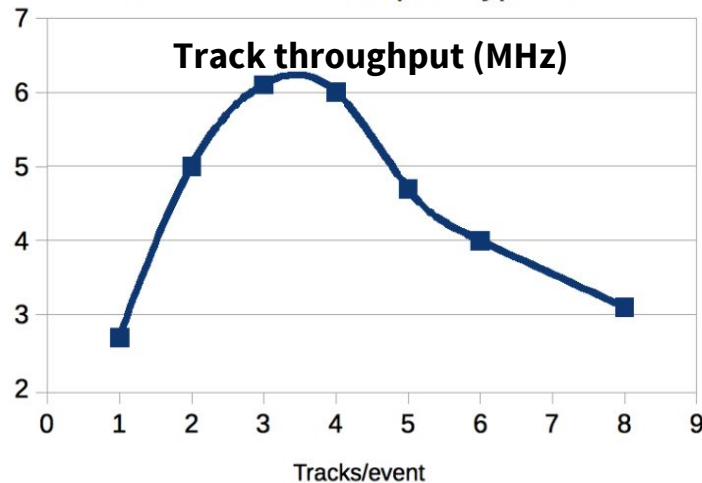
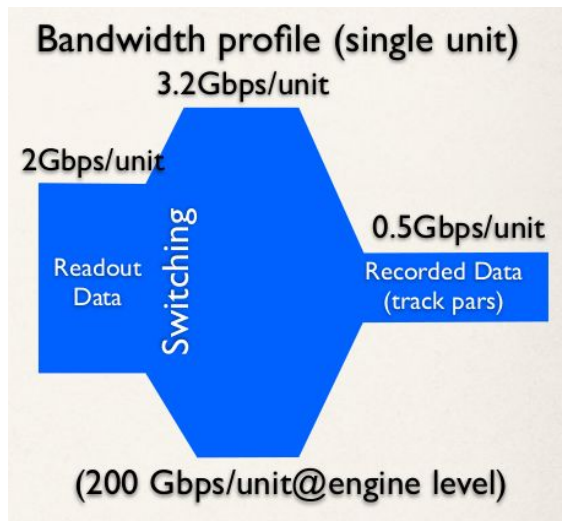
Fully pipelined  $\rightarrow$  One hit/clock cycle  $\rightarrow \sim 1.8$  MHz event-processing rate

# Results of functionality test

Test with simple events (6 hits per track, without noise hits)

Pair of boards running continuously at nominal clock speed (160 MHz)

Events processed at the same rate (sustained) as a DAQ system built on the same hardware



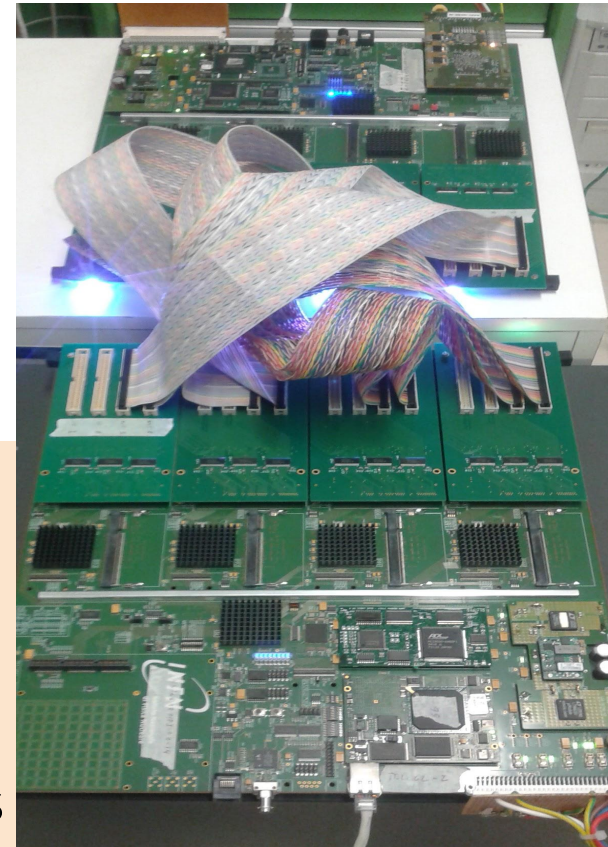
# Conclusions

Work in progress:

- test at 40 MHz event input rate (w/ commercial boards with bigger / faster FPGAs)
- test of lateral communication among modules: full mesh switching network (FPGA + optical fibers)

Summary:

- ✓ Developed **firmware** for existing boards
- ✓ Prototype unit (1/4 of modular system) designed, simulated and **run at nominal clock speed**
- ✓ **Competitive track rates** w.r.t. existing methods
- ✓ **Cost effective** scaling to larger area detector





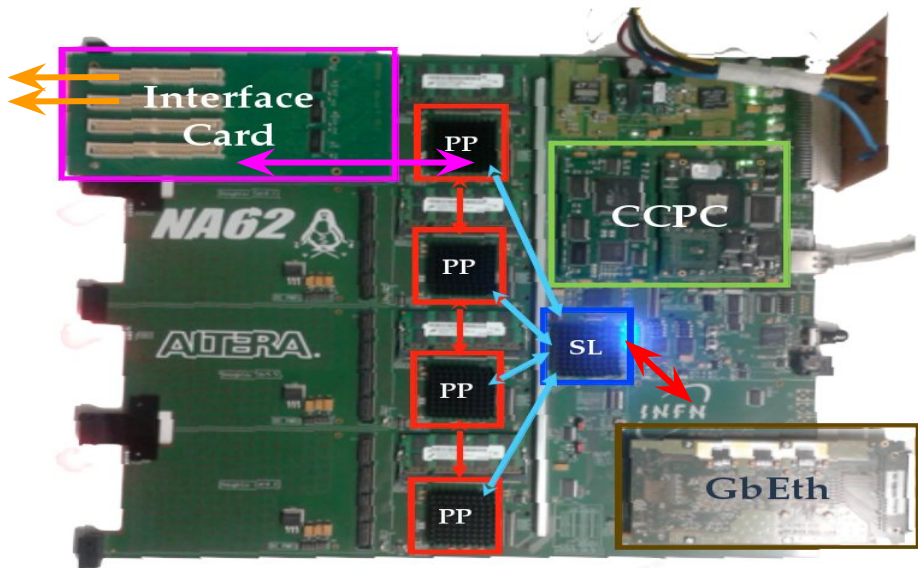
# Implementation

A standard readout board used in the NA62 experiment is the building block of our test setup

Stratix III FPGAs (2006, 65 nm): 4 data processing chips (PP) + 1 master chip (SL)

Main clock: 40 MHz

Data processing/transfer: 160 MHz



10 Gb/s links

2.5 Gb/s links

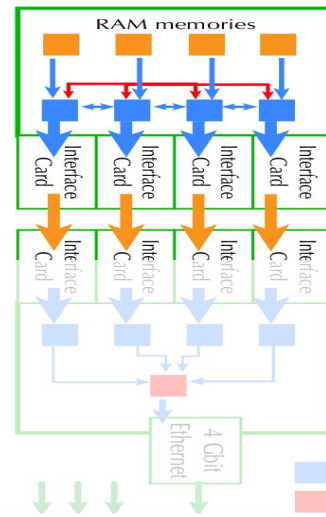
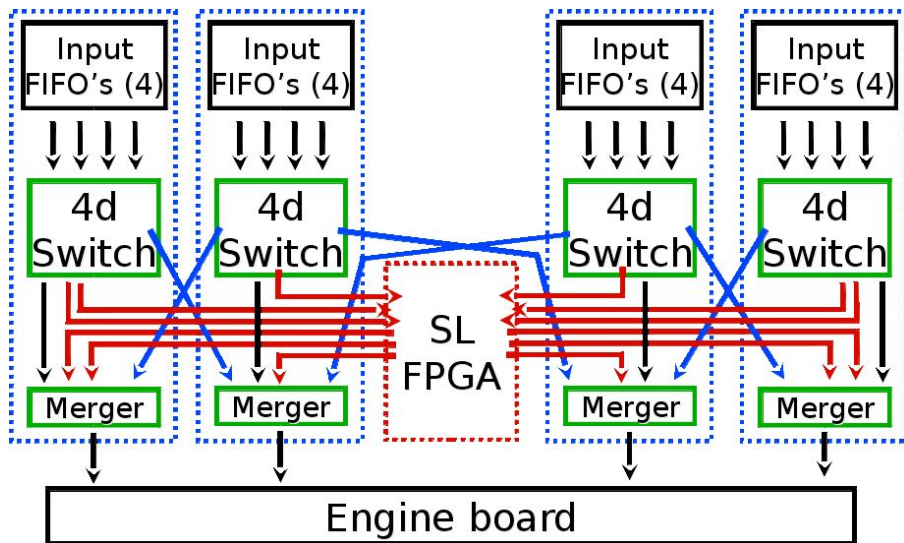
5 Gb/s links

1.6 Gb/s links

# Switch board

We rely on master FPGA to implement interconnection among non-adjacent FPGAs → full mesh

Engines in the same chip receive the same hit sequence



# Switch implementation

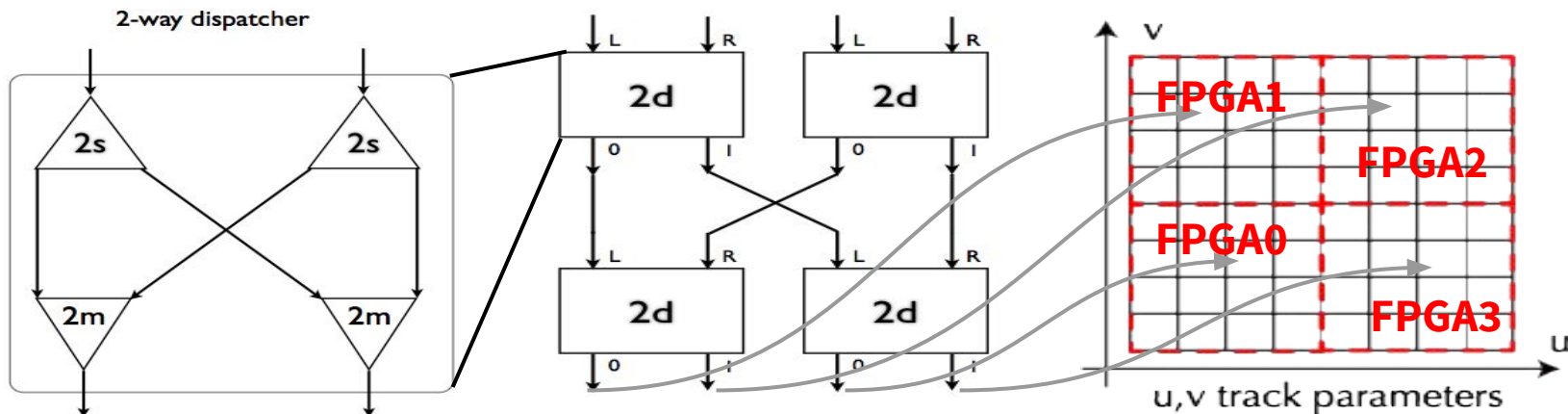
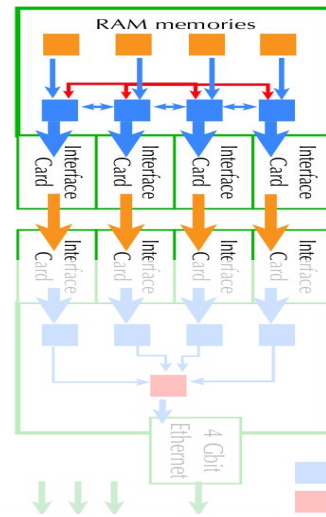
Pipelined to increase throughput, and modular

Latency proportional to  $\log_2[\max(\#\text{inputs},\#\text{outputs})]$

Splitter (2s): Latch, **LUT** (always copy EE), control FSM

Merger (2m): Latches (x2), MUX, control FSM

EE hits, after being received on both inputs, are copied to output





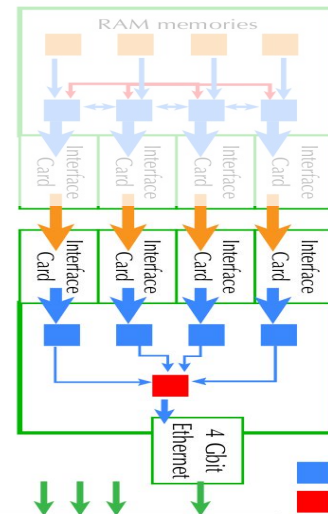
# Engine board

Engines include the logic to check for local maxima

→ Lots of interconnections, limiting number of engines

Each PP FPGA can fit an array of 16x15 engines (cells), using 90% of the FPGA resources → can fit 3000 engines in 4 TEL 62 boards

The master (SL) chip collects maxima from all the processing chip and it sends them out through the Ethernet connectors



# Logic Analyzer output for engines

Sequences of hits from 100 single-track events are loaded into engines and processed at the nominal TEL 62 speed (160 MHz)

✓ Fully pipelined → one hit per clock cycle

Bottleneck is the switch: maximum hit input rate = 10 MHz / line

