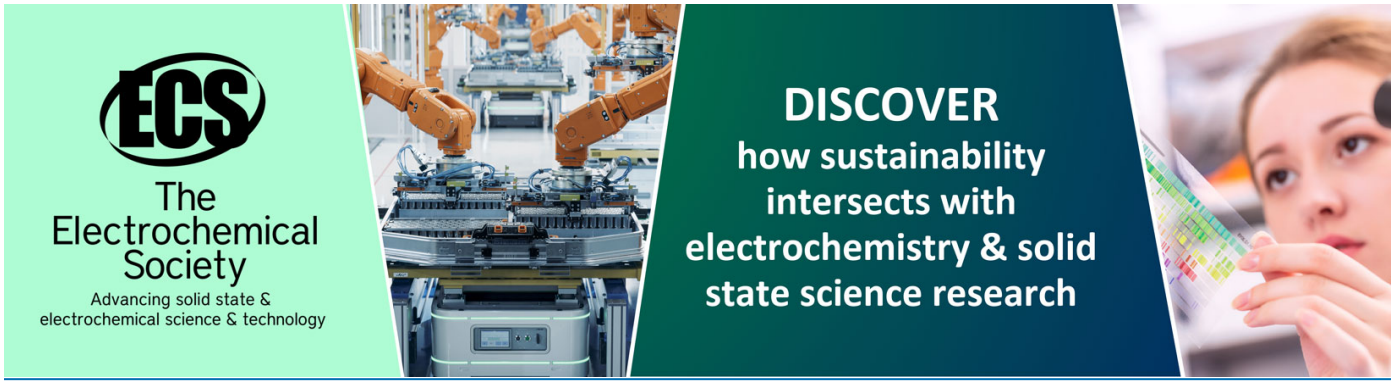
An artificial retina processor for track reconstruction at the LHC crossing rate



IOP Conf. Series: Journal of Physics: Conf. Series **898** (2017) 032038 doi:10.1088/1742-6596/898/3/032038

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

F Bedeschi[1], R Cenci[1,3], P Marino[1,3], M J Morello[1,3], D Ninci[1,2], A Piucci[1,2], G Punzi[1,2], L Ristori[4], F Spinella[1], S Stracka[1,2], D Tonelli[5] and J Walsh[1]

[1] Istituto Nazionale di Fisica Nucleare, Sezione di Pisa, I-56127 Pisa, Italy
[2] Università di Pisa, I-56127 Pisa, Italy
[3] Scuola Normale Superiore, I-56127 Pisa, Italy
[4] Fermi National Accelerator Laboratory, Batavia, IL 60510-5011, USA
[5] Istituto Nazionale di Fisica Nucleare, Sezione di Trieste, I-34149 Trieste, Italy

E-mail: simone.stracka@pi.infn.it

Abstract. The goal of the INFN-RETINA R&D project is to develop and implement a computational methodology that allows to reconstruct events with a large number (> 100) of charged-particle tracks in pixel and silicon strip detectors at 40 MHz, thus matching the requirements for processing LHC events at the full bunch-crossing frequency. Our approach relies on a parallel pattern-recognition algorithm, dubbed artificial retina, inspired by the early stages of image processing by the brain. In order to demonstrate that a track-processing system based on this algorithm is feasible, we built a sizable prototype of a tracking processor tuned to 3 000 patterns, based on already existing readout boards equipped with Altera Stratix III FPGAs. The detailed geometry and charged-particle activity of a large tracking detector currently in operation are used to assess its performances. We report on the test results with such a prototype.

1. Introduction

Computing and storage demands of future LHC experiments at very high luminosity represent a challenge for HEP data processing, which calls for an efficient and scalable usage of the hardware. The increasing input rates and growing complexity of physics events, along with the finite bandwidth for writing to long term storage, call for sophisticated and computing intensive trigger algorithms. The available CPU time and I/O bandwidth, to and from storage, limit the amount of offline data reprocessing that can be performed.

Collaborations are actively experimenting with the reduction of the event size, which can be accomplished, e.g., by persisting only a limited set of quantities calculated at trigger level instead of the complete event [1]. By taking into account the typical duty cycle of an accelerator complex, the size of the disk buffer in the online farm may be increased to maximize CPU utilization [2]. As a byproduct, a significant fraction of the first (and sometimes only) processing pass may now take place before the data are sent to tape. Part of the collected data are nevertheless temporarily set aside without being processed, waiting for computing resources to become available.

The trend of migrating the event reconstruction to the online stage is likely to continue. Tracking represents a large fraction of the event reconstruction, and is performed for almost

every event. FPGAs are particularly suited to perform repetitive tasks, such as tracking, freeing CPU resources for higher-level operations. FPGA systems also allow for low and fixed latencies: an FPGA-based tracking unit could be integrated in the DAQ architecture and act as a "track-detector", thus making event reconstruction primitives immediately available to event-building and high-level-trigger farms.

2. The track reconstruction algorithm

The *artificial retina* [3] is a highly-parallel pattern-matching algorithm, whose architectural choices, inspired to the early stages of image processing in mammals, make it particularly suitable for implementing a track-finding system in present-day FPGAs. The processing steps are illustrated in figure 1 and figure 2.

The track parameter space is divided into *cells*. For a given input, only a few units are active. The portion of the input space where a cell has non-zero response is called its *receptive field*. This local representation mirrors the locally sensitive, orientation-selective neurons in the cat's visual system [4]: cells in the visual cortex are tuned to recognize a specific pattern and to respond maximally when the retinal stimulus and the pattern are closest. Local connectivity allows us to exploit spatially local correlations to increase scalability, and is particularly important when dealing with high-dimensional inputs, as it would be impractical to connect all cells to all regions of the input space. In order to implement this feature we introduce a distance function between a given input hit and a track hypothesis associated to cell h. In what follows we use the Euclidean distance between the position $x_\ell^{(i)}$ of a hit i in detector layer ℓ and the position $m_\ell^{(h)}$ of the intersection of the track with that layer.

For each incoming hit i in the receptive field of cell h, the cell's weight is increased by the value of a response function, which is here chosen to be Gaussian:

$$w_i = \exp\left[-\frac{\left|x_\ell^{(i)} - m_\ell^{(h)}\right|^2}{2\sigma_\ell^2}\right].$$

Since contiguous cells in parameter space usually map to overlapping receptive fields, the same hit may contribute, with different weights, to the response of all cells in the neighborhood of h. After all hits from the same event have been processed, the accumulated weight of a cell can be compared to those of neighboring cells to look for local maxima, which are then identified as tracks. By calculating the centroid of the weights in the neighborhood, it is possible to improve the precision on track parameters achieved with a fixed number of stored patterns. The continuous level of matching featured in the artificial retina represents therefore an advantage over algorithms relying on a binary response, such as the Hough transform [5] or the Associative Memory [6]. The usage of resources can be further optimized by choosing the set of stored patterns that maximizes information.

3. Tracking performances

Previous studies on simulation showed that a good tracking quality (similar to the one achievable offline) can be obtained for large detectors using the artificial retina. In Ref. [7], we characterized the performances of three-dimensional track reconstruction with this algorithm, under the assumption that the input data originated from a detector system consisting of the LHCb pixel vertex locator and two silicon-strip stations in the fringe field of a dipole magnet. The logic resources required to implement the artificial retina trained to solve this task on FPGA were estimated to correspond to 50 Stratix V FPGA chips. A 97% track finding efficiency and an 8% fake track rate were measured with the artificial retina algorithm. These values are similar to the ones measured for the offline algorithms [8]. The momentum resolution was found to be 25% broader for the retina algorithm with respect to the offline algorithm.

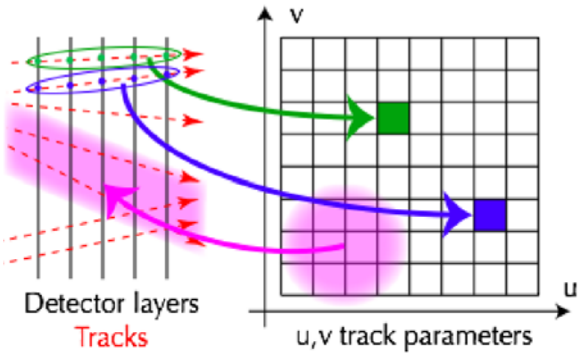
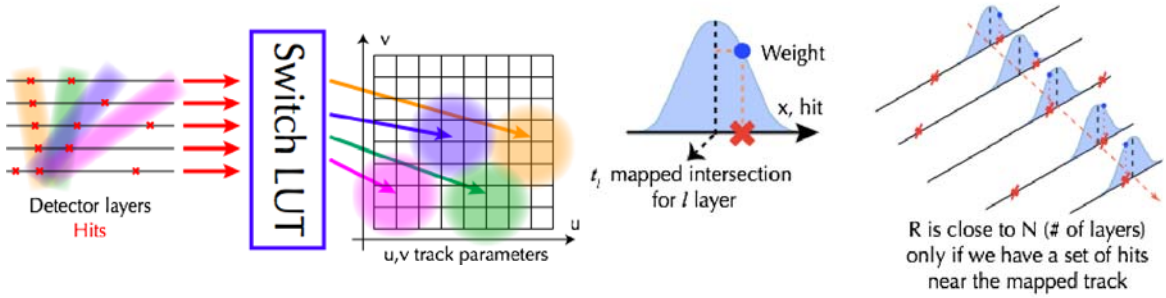


Figure 1. Retina mappings for tracks on a plane without magnetic field, where tracks can be described by two parameters u and v.

1) Template-space cells are routed only to the relevant detector elements.

2) An analog voting scheme is executed in parallel for each cell in processing engines.



3) Tracks are identified as local maxima, using interpolation for increased precision.

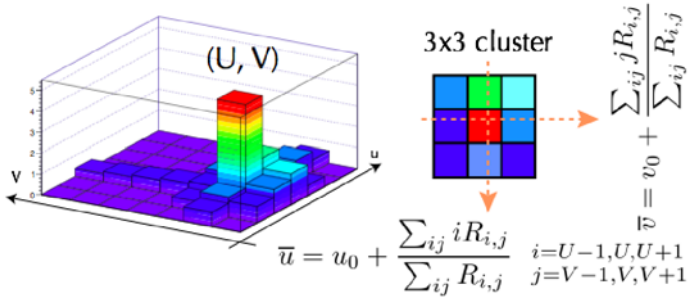


Figure 2. Processing steps of track reconstruction with the artificial retina algorithm.

4. Prototype for a tracking device

To explore the feasibility of a track processing device based on the artificial retina algorithm, we have then focused on a simpler pattern recognition problem, for which the trained retina algorithm could be fully implemented in a small prototype relying on existing FPGA boards, originally designed for the NA62 DAQ system. This prototype has been subjected to various functionality tests.

4.1. Geometry and simulation studies

The application case for this prototype consists of a six-layers silicon strip telescope, designed to reconstruct two-dimensional tracks in the $x - z$ plane (figure 3). The distance between the first and last layer is 70 cm along the z direction. Each layer is 53 cm wide and 22 cm high, with strips oriented along the y axis and separated by a 200 μm pitch, and is divided into 7 independent readout modules. To simplify pattern recognition, the layers are arranged into three pairs, where the layers in each pair are separated by 5 cm. Each event consists, on average, of 10 signal tracks. Background tracks and noise hits are also present in the event, for an average occupancy of 140 hits per event ($<$ 250 hits at 90% probability). The retina algorithm has been populated with 3000 patterns (cells), identified by the values of the x coordinates of the intersections of the track with the first and last layer (u and v, respectively).

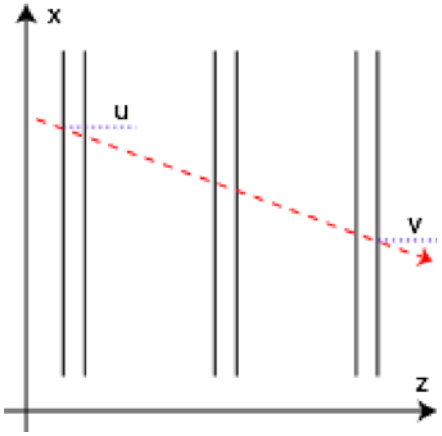


Figure 3. Sketch of the 6-layer telescope used for the prototype. Each track is represented by the u and v parameters, shown in the picture.

4.2. Board details

We chose the TEL 62 board [9] as the building block of the prototype. Each board includes four Stratix III (65 nm) chips for data-processing. The main clock is 40 MHz, while the processing internal to the FPGAs is performed at 160 MHz. Each chip is connected via a high-speed link (10 Gbit/s) to a master Stratix III FPGA, which collects the processed data and controls the other FPGAs, to a 2 GByte DDR2 RAM via a 40 Gbit/s bus, to a mezzanine connector that can host I/O interface cards (5 Gbit/s), and to the neighboring FPGAs (2.5 Gbit/s). The master FPGA is also connected to a mezzanine card with four Ethernet ports (2.5 Gbit/s). A slow-control interface, implemented on a processor embedded on the board, can be used to monitor all devices and run performance tests.

4.3. Implementation

The prototype is implemented using a paired configuration of TEL 62 boards, shown in figure 4 and figure 5. The first board (switch board) hosts the switching network, whose task is to distribute hits from the readout modules to the relevant hit-processing units (engines) of the system. This network is assembled using two-way dispatchers with two inputs and two outputs: data on any input can be redirected to any output, according to a look-up table, using two splitters and two mergers. When fed with realistic events ($<$ 250 hits at 90% probability) the switching network delivers up to 90 hits to any engine.

All engines are hosted in the second board (engine board). Each processing engine may implement the voting scheme for one or more cells in the track parameter space. The intersections of template tracks with the detector layers are stored in look-up tables, and can be updated without recompiling the firmware. Each Stratix III FPGA can host 16×15 processing engines (one per cell), using 90% of the chip resources. Sixteen chips, corresponding to four pairs of TEL 62 boards, are therefore sufficient to implement the whole system. The implementation of processing engines is fully pipelined, and each engine can receive one hit per clock cycle. With 90 hits per event, the maximum event rate sustainable by an engine board with a nominal clock frequency of 160 MHz is therefore 1.8 MHz. Using a VHDL simulation environment [10], we also estimated the latency to be ≈ 120 clock cycles ($< 1\,\mu$s).

All processing engines in a chip receive the same hit sequence and, after a special end-event hit is received, the cells' accumulated weights are copied to the following stage where the search

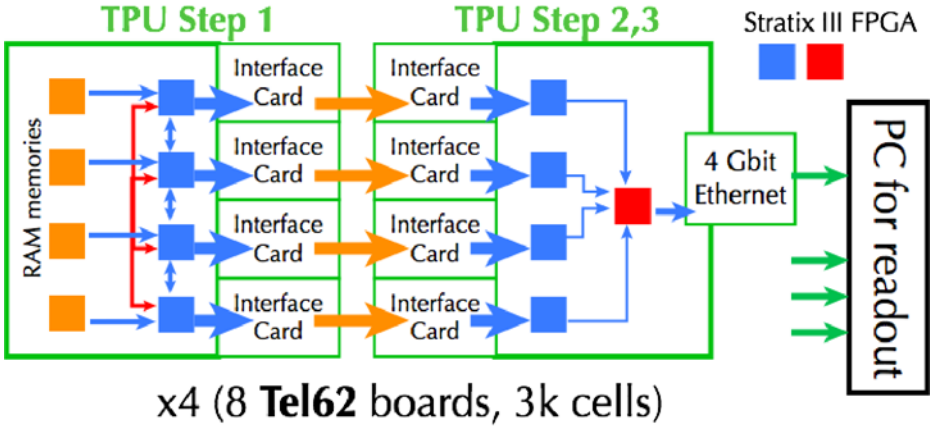


Figure 4. Architecture of the prototype using TEL 62 boards.

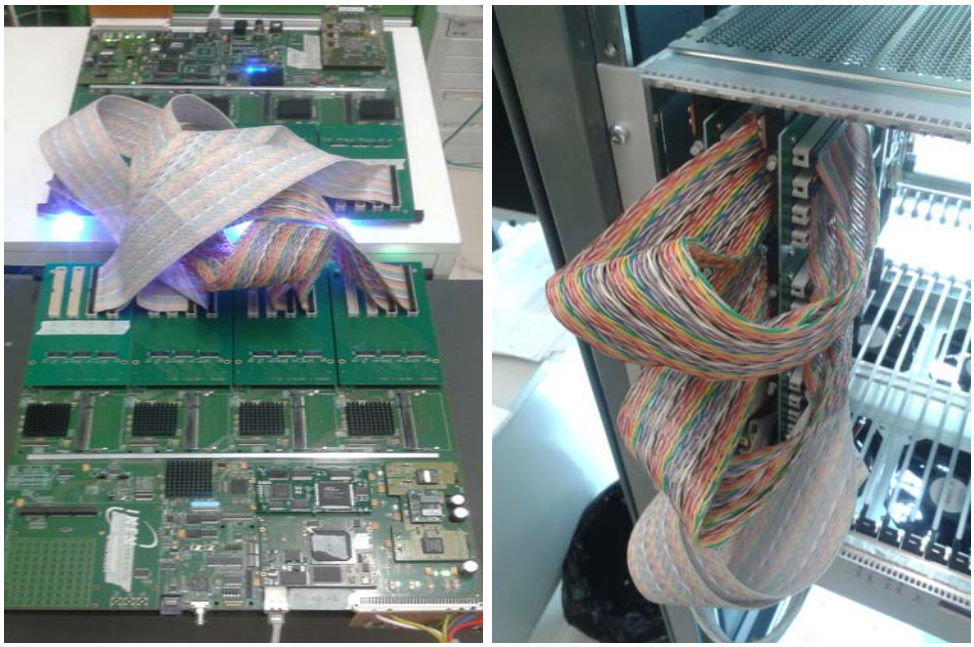


Figure 5. Prototype of the track-reconstruction system, based on the retina algorithm, in the laboratory.

for local maxima happens, while accumulators are reset and the next hit sequence is loaded. The engines include the logic to check for local maxima: if the accumulated weight is above a given threshold, it is compared with first neighbors. This requires a large number of interconnections between engines, which ultimately limits the number of engines that can be fit in a chip.

The bandwidth increases significantly during the distribution of hits, as multiple copies of the same hit are produced, but shrinks down in the last step, when only the information about local maxima is kept. All maxima from the processing chips can therefore be collected to the master chip and transfered to a PC via Ethernet.

5. Results

All the processing steps, like dispatching hit data and finding local maxima in the track parameters space, have been successfully implemented in the board and run at the nominal clock frequency (160 MHz). During the functionality tests, data transmission between the switch and engine boards was running at 80 MHz, using two data channels for each interface card pair.

To measure the throughput, simple input hit sequences were used, terminated by an end-event flag. Each hit sequence corresponds to a fixed number of tracks, with six hits per track and no noise hits. Because the board arrangement used for the prototype does not allow data transfer from an external input unit to the switch board, hit sequences were pre-loaded in embedded RAM blocks ($\times 16$) and from there injected into the switching network. We compared the values of the weights accumulated in each cell with high-level simulation, and verified that correct parameters for reconstructed tracks were received on the output. The achieved track- and event-throughputs of a TEL 62 board-pair (1/4 of the designed system) are shown in figure 6 and figure 7, as a function of the number of tracks in each input event.

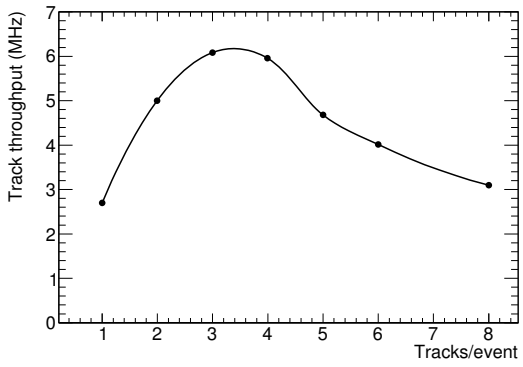


Figure 6. Track-throughput of a TEL 62 board-pair as a function of the number of tracks in each event provided in input to the switch board.

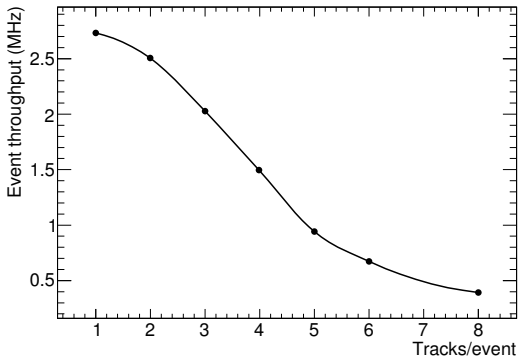


Figure 7. Event-throughput of a TEL 62 board-pair as a function of the number of tracks in each event provided in input to the switch board.

6. Conclusions

A prototype track-processing unit based on the artificial retina algorithm has been designed, simulated, and built using existing boards originally intended for a readout-only functionality at 1 MHz, to demonstrate the feasibility of fast track-finding with an FPGA-based system. The system has been run at the nominal clock speed and track rates in the MHz range have been achieved. The design can be scaled to larger area detectors and higher input rates in a cost-effective way, and a second prototype, based on more recent FPGAs, is under development to test the functionality at higher event rates. Such specialized processors may be used to rapidly

find the reconstruction primitives (e.g., tracks), and provide these primitives to the high level trigger, in parallel with the raw detector information in the event.

References

[1] CMS Collaboration 2012 *Data Parking and Data Scouting at the CMS Experiment* CMS-DP-2012-022 (Geneva: CERN).
[2] Frank M *et al* 2014 *J. Phys. Conf. Ser.* **513** 012006.
[3] Ristori L 2000 *Nucl. Instrum. and Meth.* A **453** 425.
[4] Hubel D H and Wiesel T N 1959 *J. Physiol.* **148** 574.
[5] Hough P V C 1959 *Proc. 2nd Int. Conf. on High-Energy Accelerators and Instrumentation, HEACC 1959 (CERN)* C59-09-14 (Geneva: CERN) p 554.
[6] Shochet M *et al* 2013 *Fast TracKer (FTK) Technical Design Report* ATLAS-TDR-021 (Geneva: CERN).
[7] Abba A *et al* 2015 *JINST* **10** C03008.
[8] Bediaga I *et al* 2014 *LHCb Tracker Upgrade Technical Design Report* LHCB-TDR-015 (Geneva: CERN).
[9] Angelucci B, Pedreschi E, Sozzi M and Spinella F 2012 *JINST* **7** C02046.
[10] ModelSim. Mentor Graphics, Wilsonville OR, USA; software available at http://www.mentor.com.