The Future Evolution of the Fast TracKer Processing Unit

Christos Gentsos, Francesco Crescioli, Federico Bertolucci, Daniel Magalotti, Saverio Citraro, Kostas Kordas, and Spiridon Nikolaidis, *Senior Member, IEEE*

Abstract—**Real time tracking is a key ingredient for online event selection at hadron colliders. The Silicon Vertex Tracker at the CDF experiment and the Fast Tracker at ATLAS are two successful examples of the importance of dedicated hardware to reconstruct full events at hadron colliders. We present the future evolution of this technology, for applications to the High Luminosity runs at the Large Hadron Collider where Data processing speed will be achieved with custom VLSI pattern recognition and linearized track fitting executed inside modern FPGAs, exploiting deep pipelining, extensive parallelism, and efficient use of available resources. In the current system, one large FPGA executes track fitting in full resolution inside low resolution candidate tracks found by a set of custom ASIC devices, called Associative Memories. The FTK dual structure, based on the cooperation of VLSI AM and programmable FPGAs, will remain, but we plan to increase the FPGA parallelism by associating one FPGA to each AM chip. Implementing the two devices in a single package would achieve further performance improvements, plus miniaturization and integration of the state of the art prototypes. We present the new architecture, the design of the FPGA logic performing all the complementary functions of the pattern matching inside the AM, the tests performed on hardware**

I. Introduction

The Fast TracKer (FTK) processor [1] is an example of dedicated hardware being used to reconstruct full events at hadron colliders. The online event selection in hadron colliders is split into steps, called trigger levels. The first step is called Level-1 and it is hardware-based, while the second step is called High Level Trigger (HLT), and it is implemented on a PC farm. The FTK processing pipeline has been successfully applied in ATLAS [2] simulated event reconstruction at the HLT level, and parasitic installation is scheduled to evaluate its processing capabilities in real time. There is currently strong interest in developing an even faster FTK system, to be eventually included in Level-1 tracking applications for high occupancy environments [3].

To improve the timing performance, the Associative Memory chips (or AMChips, dedicated chips performing the pattern matching) and the FPGA devices that implement the rest of the algorithm, are to be integrated more tightly. Instead of having a ratio of one FPGA device per 16 AMChips [4], the goal is to assemble one AMChip with one FPGA and a standard memory chip die in the same package. This will ensure not only minimum communications overhead between them, which will have a significant impact on the total latency, but also much more available processing power, allowing consequently higher selection efficiency and background rejection even with high detector occupancy.

In applications with stringent timing requirements, like in Level-1 trigger systems with latency of the order of $2-10\,\mu$s, a massive parallelism is crucial. In addition to that, every optimization on the performance of the FTK processor implementation can add significant advantages. Thus, the main building blocks that make up the FTK processor have been redesigned, with the goal of minimizing the total latency, while maximizing processing capacity.

II. Implementation

In FTK the tracking process is split in two steps. By matching hits against stored sets of pre-calculated hit combinations, called patterns, the AMChip first finds low resolution track candidates called "roads". Then, within each road, all real track candidates are examined using a localized linear fit and approved or rejected according to their computed χ^2 value. This step-based approach mitigates the combinatorics problem by a huge factor.

It might be useful to mention the components that perform those functions, before describing them in detail. Playing a core part in the process, the AM Chip (a custom ASIC) performs the low-resolution pattern matching. Subsequent steps are performed in an FPGA. A major component of the FPGA firmware is the Data Organizer. This logic block stores the full resolution hits according to their low-resolution representation. Then the Combiner units compute all track-forming combinations of hits scattered over different layers. At the end, the Track Fitter units exploit the hard DSP blocks existing inside modern FPGAs to perform very fast linear fits. It might be worth noting that everything except the pattern matching process is done using one Xilinx FPGA device [5].

A. Data Organizer

The functions of the Data Organizer (DO) architecture include the following. Firstly, it stores full resolution hits

This work receives support from Istituto Nazionale di Fisica Nucleare; Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science and MEXT, Japan; and the European community FP7 People grants ARTLHCFE 254410 FP7-PEOPLE-2009-IOF and

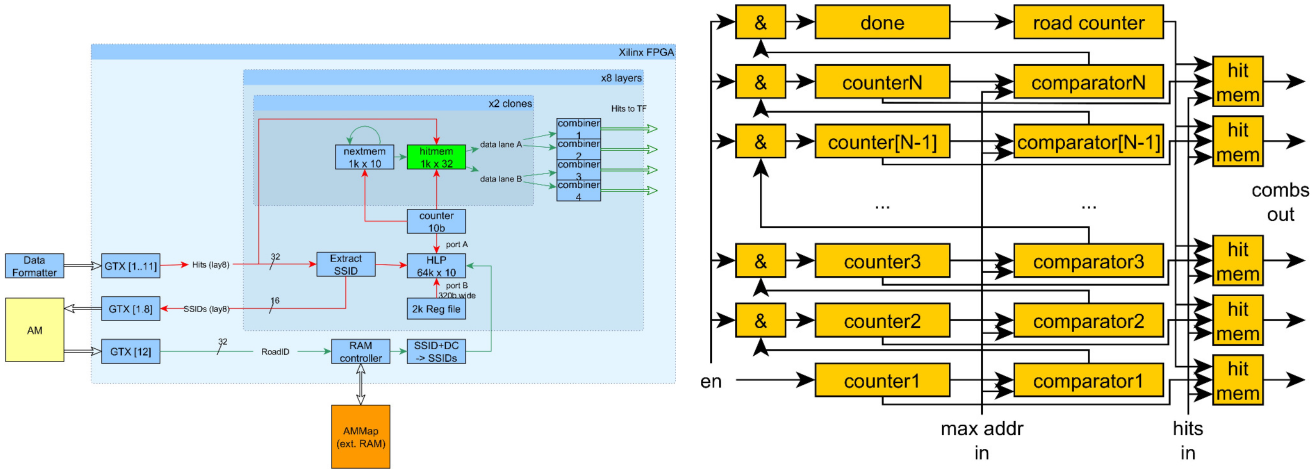
C. Gentsos is with the Aristotle University of Thessaloniki, Greece, seconded to Prisma Electronics S.A., e-mail: cgentsos@physics.auth.gr.

F. Crescioli is with the LPNHE CNRS, Paris, France, e-mail: francesco.crescioli@lpnhe.in2p3.fr.

F. Bertolucci and S. Citraro are with the University of Pisa, Italy, e-mails: federico.bertolucci@cern.ch, saverio.citraro@for.unipi.it.

D. Magalotti is with the University of Modena and Reggio Emilia, Italy, e-mail: daniel.magalotti@pg.infn.it.

K. Kordas and S. Nikolaidis are with the Aristotle University of Thessaloniki, Greece, e-mails: kostas.kordas@cern.ch, snikolaid@physics.auth.gr.



(a)

(b)

Fig. 1: (a) System block diagram highlighting Data Organizer structure, (b) Combiner core architecture block diagram

according to 16-bit SSIDs (effectively coarse resolution hits, called Super-Strips, on a detector layer). Secondly, the AM Chip output has to be decoded in SSIDs and Don't Care bits (the function of which is explained below). Finally, the full resolution hits must be retrieved based on this information.

It also has to satisfy a number of requirements. First of all, conforming to the AM Chip Don't Care capability [3], a pattern can be matched to up to 8 SSIDs/layer. This function improves the overall pattern recognition efficiency of the pattern bank, effectively creating patterns of variable shape. Also, the number of hits that can be grouped within an SSID must not be limited, as is the case in previous implementations. Then, the performance of the Data Organizer and its surrounding modules has to be as deterministic as possible and finally, especially on the reading stage, it has to maintain a very high processing bandwidth.

The DO functionality for one detector layer is equivalent to a somewhat complicated linked-list: its simplified block diagram and role in the system can be seen at Fig. 1a. The full resolution hits are written in a memory as they arrive. Then the Hit List Pointer (HLP) memory, whose every location corresponds to an SSID, is updated with the address of the last hit to arrive for that specific SSID. At the same time, another memory is updated with the invalidated information the HLP held, so the address of the last hit stored can point to the previous hit of the same SSID (if there is one).

To account for the case of a missing layer and support the Don't Care function, while storing a hit matched to an SSID, the system takes into account whether any other hits have already been written during the same event. That is made possible with the use of a register file. To keep the register file small (2k instead of 64k), and at the same time to boost read performance, the HLP memory has been widened so that each location covers 32 SSIDs. This has a big impact on performance: it allows to read all the possible hit locations of 8 SSIDs in one clock cycle, releasing the HLP to fetch the next road locations. Moreover, the dual-port hit memory is duplicated, so there are four channels reading at the same time, providing the necessary memory bandwidth to support the fast parallelized HLP output.

The operating frequency of the Data Organizer is $400\,\mathrm{MHz}$. When writing it can accept hits every 2 clock cycles, thus the maximum input rate is $200\,\mathrm{MHits/layer/s}$, and when reading data the maximum output rate is $1600\,\mathrm{MHits/layer/s}$.

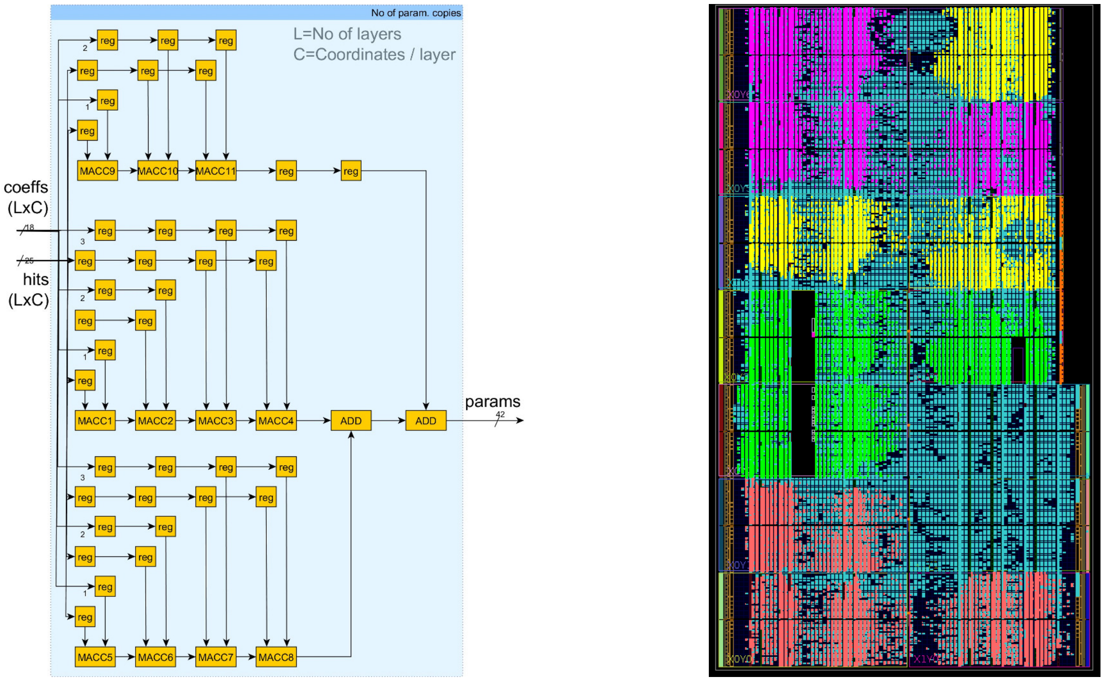
B. Combiner module

Between the Data Organizer module and each Track Fitter unit, it is necessary for a Combiner module to compute all the possible combinations that form valid tracks. The Combiner function seems simple but the realization in hardware has a long critical path, which keeps the frequency low. To speed up this step, two combiner units are assigned to each Track Fitter, each working at exactly half the frequency ($250\,\mathrm{MHz}$). At each clock cycle, a Combiner unit produces exactly one combination. Between successive roads, there is no downtime, making it possible to process a road of one track in just one clock cycle.

The block diagram of the Combiner unit core can be seen in Fig. 1b. Each hit memory is split into sectors, each one filled with the hits from a specific road. The sector data are written asynchronously during the read operation in a cyclic way, similarly to a FIFO memory operating on its locations, with FIFO-like flags being generated to protect the hit data from being overwritten. Due to its simple structure, the architecture is easy to be adapted to an arbitrary number of layers. Since increasing the number of layers will bring the operating frequency down, one can exploit higher level parallelism by changing the number of Combiners/Track Fitter.

C. Track Fitter

The Track Fitter units extract the helix parameters from the local hit coordinates employing linear functions and using a χ^2 cut as a goodness of fit indicator. The resolution of the



(a)

(b)

Fig. 2: (a) Block diagram of the Track Fitter architecture (b) FPGA implementation of the Data Organizer and 4 Track Fitter units, each unit is highlighted in color

resulting fit approaches that of a full helical fit, as long as the fit coefficients are generated for a narrow sector of the detector [6].

Making extensive use of dedicated DSP units in modern FPGA devices, one can realize a very fast implementation. The block diagram of the Track Fitter architecture can be seen in Fig. 2a. The architecture tries to strike a balance between low latency and resource usage (both DSP and register), while maintaining a high operating frequency and exploiting the parallelism capabilities of FPGAs. Each Track Fitter unit operating at $500\,\mathrm{MHz}$ can perform a fit every $2\,\mathrm{ns}$, after an initial latency of $\sim 90\,\mathrm{ns}$. Four units can fit in a modern mid-grade device (for a floorplan of an implementation see Fig. 2b), making the maximum fitting performance of the FPGA $2\,\mathrm{GFits/s}$. Such high performance can help in maintaining the trigger efficiency in a high occupancy environment.

III. Results

A firmware has been designed, that makes efficient usage of the device resources to allow for a future-proof implementation, maintaining a significant performance overhead to allow for AMChip upgrades. Track fitting performance reaches 2GFits/s, and the rest of the architecture (namely the Data Organizer and Combiner modules) can sustain that performance. The total latency of the system (here defined as the time interval between the first pattern matching result coming from the AMChip and the first track candidate having been fully processed, as the first track that actually passes the χ^2 cut can't be predicted) is $\sim 300\,\mathrm{ns}$. The design has already been implemented on a Xilinx 7-series Kintex FPGA device and is currently being integrated as a full system on a mezzanine PCB, to be seated on a PULSAR board. After the integration process, it is planned to test the actual performance on the board using simulated events on real-time, and do the necessary simulation studies to evaluate the impact the increased performance will have on the achievable efficiency of the tracking system.

References

[1] ATLAS Collaboration, "Fast TracKer (FTK) Technical Design Report," CERN, Geneva, Tech. Rep. CERN-LHCC-2013-007. ATLAS-TDR-021, Jun 2013, ATLAS Fast Tracker Technical Design Report. [Online]. Available: http://cdsweb.cern.ch/record/1552953

[2] ATLAS Collaboration, "The ATLAS Experiment at the CERN Large Hadron Collider," *J. Instrum.*, vol. 3, p. S08003. 437 p, 2008, also published by CERN Geneva in 2010. [Online]. Available: https://cdsweb.cern.ch/record/1129811

[3] A. Annovi *et al.*, "Associative Memory for L1 Track Triggering in LHC Environment," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 5, pp. 3627–3632, Oct 2013.

[4] P. Luciano *et al.*, "The Serial Link Processor for the Fast TracKer (FTK) at ATLAS," in *Proceedings, 3rd International Conference on Technology and Instrumentation in Particle Physics (TIPP 2014)*, vol. TIPP2014, SISSA. SISSA, 2014. [Online]. Available: http://pos.sissa.it/cgi-bin/reader/conf.cgi?confid=213

[5] *7 Series FPGAs Overview*, Xilinx, 5 2015, v1.17. [Online]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf

[6] S. Amerio *et al.*, "The GigaFitter: A next generation track fitter to enhance online tracking performances at CDF," in *Nuclear Science Symposium Conference Record (NSS/MIC), 2009 IEEE*, R. C. Lanza, Ed., IEEE. New York, USA: IEEE, Oct 2009, pp. 1143–1146. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5384532