Proposed FPGA based tracking for a Level-1 track trigger at CMS for the HL-LHC

Nicola Pozzobon for the CMS Collaboration

Abstract

The High Luminosity LHC (HL-LHC) is expected to deliver a luminosity in excess of $5 \times 10^{34} \text{cm}^{-2}/\text{s}$. The high event rate places stringent requirements on the trigger system. A key component of the CMS upgrade for the HL-LHC is a track trigger system which will identify tracks with transverse momenta above 2 GeV already at the first-level trigger within 5 $\mu$s. This presentation will discuss a proposed track finding and fitting based on the tracklet based approach implemented on FPGAs. Tracklets are formed from pairs of hits in nearby layers in the detector and used in a road search. Summary Fast pattern recognition in Silicon trackers for triggering has often made use of Associative Memories for the pattern recognition step. We propose an alternative approach to solving the pattern recognition and track fitting problem for the upgraded CMS tracker for the HL-LHC operation. We make use of the trigger primitives, stubs, from the tracker. The stubs are formed from pairs of hits in sensors separated radially by a few millimeters. This allows us to place a $p_T$ cut on the stub and reject the vast majority of the hits from low $p_T$ tracks. In a typical bunch crossing at the HL-LHC we will have approximately 140 proton-proton interactions, producing about 10,000 stubs. The proposed pattern recognition algorithm forms tracklets, seeds for the pattern recognition, by combining pairs of stubs in neighboring layers that are consistent with $p_T$ greater than 2 GeV and the tracklet originating from the interaction region. These tracklets are used to define roads where stubs in other layers are added to form the complete track. A linearised $\chi^2$ fit is used to obtain the final track parameters. The implementation of this algorithm on an FPGA is done in sectors, where each sector is processed by one FPGA. In the first implementation of this algorithm we assume a time multiplexing of a factor of 4. In this presentation we will discuss the performance of the track finding algorithm and the resources used in the implementation.

Presented at TWEPP 2014 - Topical Workshop on Electronics for Particle Physics
Proposed FPGA Based Tracking for a Level-1 Track Trigger at CMS for the HL-LHC

N. Pozzobon*

Università degli Studi di Padova e INFN - Sezione di Padova, via F. Marzolo 8, 35131, Padova, Italy
*on behalf of the CMS Collaboration
E-mail: nicola.pozzobon@pd.infn.it

ABSTRACT: The High Luminosity LHC (HL-LHC) is expected to deliver a luminosity in excess of $5 \times 10^{34} \text{cm}^{-2}\text{s}^{-1}$. The high event rate places stringent requirements on the trigger. A key component of the CMS upgrade for the HL-LHC is a track trigger to identify tracks with transverse momenta above 2 GeV already at the first-level trigger within 5 $\mu$s. This presentation will discuss a proposed track finding and fitting based on the “tracklet-based approach” implemented on FPGAs. Tracklets are formed from pairs of hits in nearby layers in the detector and used in a road search.

KEYWORDS: CMS Upgrade; HL-LHC; Tracking; Level-1 Trigger.
1. Introduction

The CERN Large Hadron Collider is expected to be upgraded in the forthcoming decade to provide collisions at 14 TeV at extremely high luminosities (HL-LHC). The HL-LHC is being designed to collect 3000 fb$^{-1}$ over 10 years (2025-2035), at peak luminosities larger than $5 \times 10^{34}$ cm$^{-2}$s$^{-1}$. At such luminosities, at a collision rate of 40 MHz, hundreds of superimposed collisions (pile-up, PU) are expected to be recorded. The current reference value chosen for performance studies is 140 PU per bunch crossing, on average. In such an environment, trigger rates cannot be sustained with the resolution of (even upgraded) calorimeters and muon system, while maintaining performance as in 2012. For this reason, a tracker-like resolution must be put into early trigger stages, making the need for a novel tracker arise. Such a new tracker must provide tracks at Level-1 Trigger [1].

Given the high granularity needed to resolve ambiguous hits in a densely populated environment, the expected data rate is too high to be processed by the Level-1 Trigger. Tracker-Trigger primitives are built at the tracker front-end in order to reduce the data volume. This is done through pattern hit correlation between two parallel and closely spaced sensors sharing the same read-out. Such a pair of sensors with a common read-out performing local reconstruction of the track $p_T$ are the core of what is called “$p_T$-module”. Separations of $O(1)$ mm and sensor pitch of $O(100)$ µm let the resolution to be good enough to put a threshold of few GeV (current threshold used for performance studies is 2 GeV) and reject 90% to 95% of hits produced by primary tracks from minimum-bias background. Only pairs of hits that are aligned within pre-defined matching windows, and are then consistent with track $p_T$ larger than a chosen threshold, are accepted and delivered to the next stage of track trigger. Accepted pairs of hits are called “track stubs”. Two different types of $p_T$-modules are currently being designed and prototyped. The first kind, called “2S” (strip-strip), is based on two identical sensors featuring half-strips $\approx$5 cm long and 90 µm wide, to be used at radii larger than 60 cm where the occupancy is lower. The second kind, called “PS” (pixel-strip), one is based on smaller sensors, one featuring half strips $\approx$2.4 cm long and 100 µm wide, one featuring
2. Concept of an algorithmic track finding at Level-1 Trigger

In this contribution, a proposed track finding and fitting, based on the “tracklet based approach” to be implemented on FPGAs is presented. This approach is algorithmic and designed around track seeds made with pairs of stubs and called “tracklets”.

1. A tracklet is built by matching together stubs in consecutive layers/disks of the tracker, provided they meet specific criteria to ensure compatibility of the corresponding track with a \( p_T \).
Figure 3. Block diagram of the data flow in current firmware implementation (sector processor). Communication between neighbour sectors is included in the diagram, as well as the final stage, i.e. the merger/output processor.

threshold and with being originated within the luminous region. Track parameters are computed using beam position as trajectory constraint, hence drawing the projection of the helicoidal trajectory onto the transverse plane as a circumference passing through three points. The creation of multiple seeds is allowed, including letting the same track to be seeded in different regions of the tracker.

2. The coarse tracklet momentum is used to extrapolate the trajectory and search for additional stubs in other layers/disks of the tracker, within pre-computed windows that depend on the layers/disks where the seed is built, and the extrapolation distance. Up to one stub per layer/disk can be attached to the seed. Both inwards and outwards projections of each seed are allowed in order to enhance the efficiency.

3. Once the extrapolation is completed, the track is fit using a linearised-$\chi^2$ method, searching for a minimum close to the momentum of the seed. Two options are being evaluated for being implemented at Level-1, one including the transverse impact parameter of the track $d_0$, one fixing it to 0.

4. Duplicates and ghosts built from multiple seeding or overlapping modules are then removed picking the one with the best $\chi^2$.

3. The tracklet-based Level-1 track finding in FPGAs

The algorithmic track finding described in section must meet strict requirements. It must work with an input rate of one event every 25 ns, same as the HL-LHC bunch crossing rate. It must be fast enough to cope with the Level-1 decision latency. It must process large amounts of data from high PU, which is expected to be $O(10^4)$ stubs per bunch crossing at reference luminosities and track trigger thresholds. It must also be efficient when implemented using only integer computation that can fit into commercial hardware at a reasonable cost [4].

The current implementation of the Verilog emulation of the tracklet-based track finding, as pictured in figure [3], needs:
• for the coarse $p_T$ and $z_0$ requirements at seeding, 1 LUT;
• for the seed momentum evaluation, 4 steps in a pipeline, 8 sums, 12 multiplications, 1 LUT;
• for the projection of the expected position along the trajectory after extrapolation, 6 steps in a pipeline, 3 sums, 5 multiplications.

In order to reduce the data volume to be processed, the tracker is divided into azimuthal sectors. A bending track can cross sectors, and the maximum number of sectors the tracker can be divided into is 28 in order to have communication only between nearest neighbours. The expected position along the trajectory is sent to the processors corresponding to the modules close to the position. Stubs are accepted if they fall within pre-computed $\phi$ and $z$ windows, and their residuals with respect to the predicted position are then sent back to the processor that built the seed for the final track fit. The linearised-$\chi^2$ final fit is based on pre-computed track derivatives stored in LUTs. The addresses of the accepted stubs are stored together with the corrected track parameters. In order to allow for a higher spacing between events, the system is time-multiplexed by a factor 4. The whole algorithm can be processed in 10 steps within 1 $\mu$s. The maximum number of stubs that can be processed is determined by the FPGA clock.

4. Expected performance

The performance of the tracklet-based track finding has been evaluated with both floating-point simulations and Verilog emulations of the data flow in FPGA boards. In particular, floating point simulations have been embedded within the framework of the software used also for the CMS offline event reconstruction and data analysis, and it has been run on top of full Geant4 simulations of the CMS detector in HL-LHC environment. Verilog emulation has been used to improve the algorithm itself, in particular for what concerns the performance with respect to combinatorial seeds. For example, $\approx 3600$ seeds per bunch crossing per sector can arise from combinatorics in the two innermost barrel layers, but only $\approx 5$ true tracks are expected with $p_T > 2$ GeV. A clever organisation of data into virtual modules and $z$-regions, combined with a preselection of the tracklets, can lower down the number of possible seeds to be evaluated to 50 per bunch crossing per sector. Such values can be managed with $O(100)$ boards equipped with next-generation FPGAs, such as Virtex7, that are expected to be a widely available and consolidated technology in time for construction and assembly of the upgrades for the HL-LHC era. A comparison between floating point simulations and Verilog emulations can be found in figure 4 and 5.

5. Current status and concluding remarks

The described algorithm has already been translated into firmware in Verilog. The full track finding and fitting chain emulation is in the barrel solid, and is currently being completed with the duplicate removal stage. Next steps towards the completion of this design will include the scaling-up of the emulation to cover a full sector, the extension of the firmware implementation to endcap disks.
Most of the tools already developed are expected to work straightforwardly also in endcaps.

The proposed pattern recognition algorithm forms tracklets, seeds for the pattern recognition, by combining pairs of stubs in neighboring layers that are consistent with $p_T > 2 \text{ GeV}$ and the tracklet originating from the interaction region. The algorithm is well understood and undergoes constant review to meet the requirement for its implementation in FPGAs. Its expected performance in HL-LHC collisions currently meets all the requirements in terms of efficiency and reso-
solution. Simulations have shown that the proposed method is viable for integer implementations in commercial hardware.

References


