FPGA-based Particle Recognition in the HADES Experiment

Abstract—Modern FPGA technologies are often employed in nuclear and particle physics experimental facilities to accelerate application-specific computation. We present the particle recognition computation for the HADES experiment in this article. The algorithms of particle track reconstruction and Cherenkov ring recognition are introduced in the context of concrete particle detectors. Implementation results of the hardware processing engines reveal the feasibility to realize the pattern recognition computation on the FPGA. Performance estimation indicates that our FPGA cluster in one ATCA shelf can achieve an equivalent computation capability to thousands of commodity PCs for the HADES experiment.

Index Terms—reconfigurable computing, FPGA accelerator, pattern recognition, nuclear and particle physics

1 Introduction

Nuclear and particle physics is a branch of physics that studies the elementary constituents of matter and the interactions between them. It is also called high energy physics because many elementary particles do not occur under normal circumstances in nature, but can be created and detected during energetic collisions of other particles, as is done in particle colliders. Modern nuclear and particle physics experiments, for example HADES [1] and PANDA [2] at GSI Germany, ATLAS, CMS, LHCb, ALICE at the LHC [3] at CERN Switzerland, achieve their goals by studying the emission direction, the energy, and the mass of the produced particles when the accelerated beam hits the target. In the experimental facilities, different kinds of detectors are adopted to generate raw data which are used to calculate and analyze the characteristics of emitted particles from the collision.

Due to high reaction rates, modern high-energy physics experiments commonly deliver high data rates (e.g. PADNA, up to 200 GBytes/s). Among the huge amounts of reaction events[1], only a rare proportion is of interest to the physicists due to its particular physics contents, and should be selected for in-depth analysis. Besides, all the data throughout the experiment time lasting for months cannot be entirely stored because of the storage limitation. Therefore it is essential to realize an efficient on-line data acquisition (DAQ) and trigger system which processes the event data coming from detectors and reduces the data rate by several orders of magnitude by means of rejecting the background and retaining only the interesting events. In the contemporary facilities, pattern recognition algorithms (trigger algorithms) [4], [9] such as *Cherenkov ring recognition*, *particle track reconstruction*, *Time-Of-Flight (TOF) processing*, etc., are implemented as sophisticated criteria according to detector categories. Only the sub-events which possess expected patterns generated by certain types of particles and could be successfully correlated among various detectors, receive a positive decision and are encapsulated in a pre-defined event structure for mass storage and further offline analysis. Others will be discarded on the fly.

Motivated by multiple ongoing projects including the HADES upgrade, we have designed a high-end reconfigurable and scalable computation platform as a general solution. Cutting-edge FPGA technologies as well as high-speed communications are adopted to guarantee high processing capability and channel bandwidth. Easy scalability is an important feature of the platform. To unify the application development on the platform, we employ a hardware/software co-design approach, with which functional tasks are partitioned between embedded microprocessors and customized algorithm cores in hardware. Hence the system design can be largely reutilized for various experiments with little performance penalty or modification effort. Systematic discussion on our platform design can be referred to in [5] for details. In this article, we put our focus on implementing pattern recognition algorithms based on this FPGA platform.

2 Related Work

Trigger algorithms in high-energy physics identify specific patterns out of massive background data to recognize interesting particle reaction events. Traditionally they are preferably implemented in offline software running on PC farms [6] [7], taking advantage of the ease of high-level programming. However offline processing requires all the data to be stored and does not contribute to reduce the data rate before storage. As the great development of FPGA technologies on programmable resource capacity and communication bandwidth, it is feasible now to migrate some software algorithm implementations into the FPGA fabric for online triggering.

1. In high-energy physics, one "event" corresponds to a single interaction of a beam particle with a target particle. It consists of sub-events which typically represent the information from individual detector sub-systems.

The data reduction rate is therefore increased for more efficient storage, and the pattern recognition processing can be accelerated by using customized hardware designs. For example in [8], the authors utilize FPGAs to implement the Compact Muon Solenoid (CMS) trigger of the Large Hadron Collider (LHC) particle accelerator at CERN. Specifically in the original HADES DAQ and trigger system, online Level-2 trigger algorithms were implemented as embedded designs on FPGAs and DSPs, except the MDC tracking processing in software programs due to the algorithm complexity for hardware implementation [4]. In the HADES upgrade project targeting heavier ion reactions, we will try to implement complete trigger algorithms for particle identification in the detectors.

3 Algorithm Description

3.1 The Detector System in HADES

The HADES (High Acceptance Di-Electron Spectrometer) experiment was constructed at GSI, Darmstadt, Germany to investigate hadron properties inside nuclear matter [1]. Figure 1 demonstrates the exploded view of the HADES detector system. By energetic collisions between the accelerated beam and the target, product particles are generated and fly through the detector system. Various detectors are employed to investigate the characteristics of the generated particles, such as mass, energy, charge, etc. In its upgrade project, the DAQ and trigger system is to be enhanced for heavier ion reactions and higher processing capability.

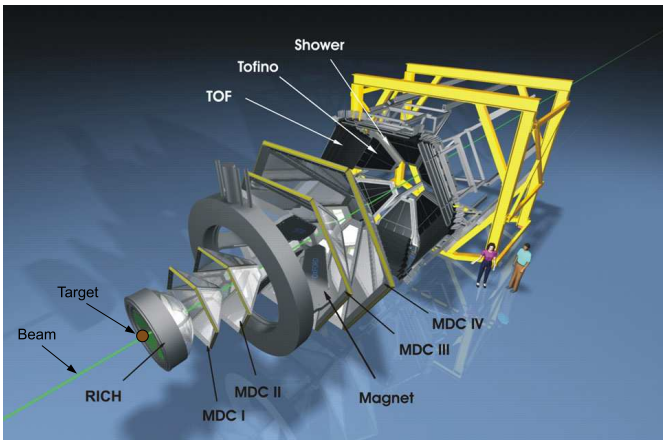
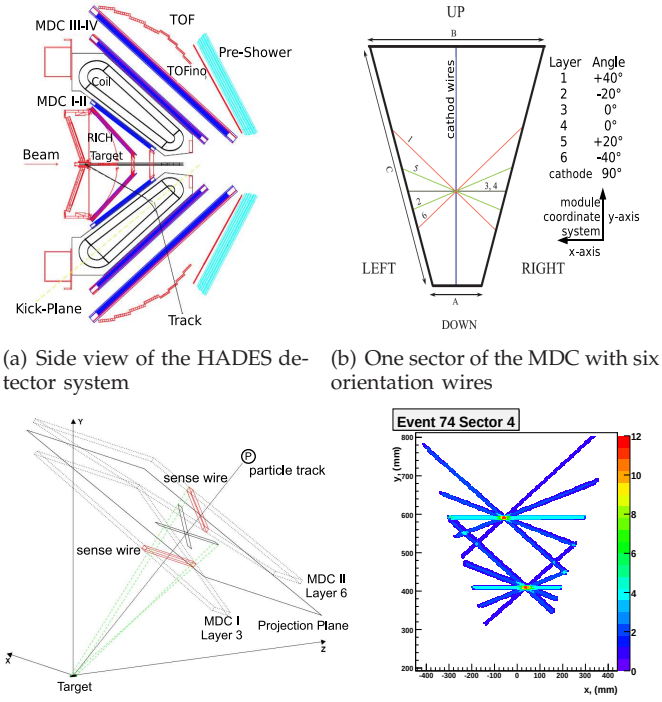


Fig. 1. The HADES detector system

3.2 Track Reconstruction in MDCs

In high-energy physics experiments, the momenta of charged particles are studied by observing their deflection in magnetic field. The so-called Mini Drift Chamber (MDC) detectors are used to reconstruct the particle tracks entering and leaving the magnetic field, for further deriving the deflection angle inside it. The HADES tracking system consists of four MDC modules which have six identical trapezoidal sectors (see Figure 1). Two MDC layers are located before and two behind the toroidal magnetic field which is produced by 6 superconducting coils, as illustrated in Figure 2(a). In first approximation the magnetic field does not penetrate into the MDCs. Thus particle tracks only bend in the magnetic field and the segments before or behind the coil could be approximately described by straight lines. The two segments can be reconstructed separately with the inner (I - II) and the outer (III - IV) MDC information. The basic principle is similar and hence in this article we focus only on the inner part for explanation.



(a) Side view of the HADES detector system

(b) One sector of the MDC with six orientation wires

(c) Track recognition and reconstruction in inner MDCs

(d) Projection plane with two passed tracks

Fig. 2. MDCs in HADES detector system

In the two inner MDC modules, a total number of 12660 sense wires (6 sectors) are arranged in 12 layers and 6 orientations: +40°, -20°, 0°, 0°, +20°, -40°, shown in Figure 2(b) for one trapezoidal sector. When the beam hit the target, charged particles are emitted from the target position and go forward through different wire layers in straight pathes. Along their flying ways, pulse signals are generated on sense wires close to the tracks with high probability (>95%). We also say that the sense wires are "fired" by flying particles. Shown in the coordinate system in Figure 2(c), if the sensitive volumn of each wire is projected from the boundary of the target onto a plane located between two inner chambers, apparently the particle passed through the projection plane at the point where all projections of fired wires from different layers overlap. To search for such regions, the projection plane is treated as a two dimensional histogram with the projection area as bins (pixels). For each fired sense

wire, all the pixels covered by its projection are increased by one. By finding the locally maximum pixels whose values are also above a given threshold, track points can be recognized and the tracks are reconstructed as straight lines from the point-like target to those peak pixels. Figure 2(d) demonstrates the 2D projection plane for one sector with two penetrating particles. We observe that the projection of fired wires from the total 12 layers overlap in two dots, which feature the amplitude peaks and represent two particle tracks.

An offline built Look-Up Table (LUT) is employed to tell which pixels on the projection plane are touched by the projection of every fired sense wire. Thus realtime coordinate calculation can be avoided considering its geometrical complexity for FPGA implementation. In practice, we choose the resolution of 128 $\times$ 256 pixels for each sector. The *projection_LUT* is about 1.5 MBytes per sector and is feasible to be initialized in the DDR2 memory.

3.3 Ring Recognition in RICH

The HADES Ring Image CHerenkov (RICH) detector is used to identify dilepton pairs based on the Nobel Prize winning discovery of Cherenkov effect. As explained in [10], a charged particle emits light cone (Cherenkov radiation) when it travels through a transparent substance with a speed faster than the speed of light in that material. Specifically in the HADES experiment, dileptons are emitted from the collision. They fly at a high velocity through the inner MDC detectors from the target. Therefore the generated Cherenkov light cone can be reflected by the mirror and displayed on the RICH detector in the shape of a ring. The ring pattern is searched on the RICH plane with a resolution of 96 $\times$ 96 pixels per sector. According to the physics principle, the Cherenkov ring from the dilepton pair features a constant diameter equivalent to the distance of 8 pixels on the RICH plane. As shown in Figure 3, the ring pattern search is conducted within a fixed mask region of 13 $\times$ 13 pixels for each potential ring center. The hits on a ring with a radius of 4 pixels are added to the value *ring_region*. There are two veto regions inside or outside the ring region, where hit pixel counts are also accummulated. The ring pattern can be identified only if both the *ring_region* sum is above and the *veto_region* sums are below their respective thresholds. The thresholds are programmable during the experiment.

Because of the constant diameter of ring patterns, the computation challenge falls on position identification of ring centers. In the original design of [11] [12] from J. Lehnert et al., they treat all the 96 $\times$ 96 pixels on the RICH plane as potential ring centers: With the received RICH sub-events containing the position of all hit pixels from the detector readout circuits, the complete hit information of the RICH plane is reconstructed in a memory device. Afterwards ring patterns are searched within respective mask regions of all the pixels as ring centers, in

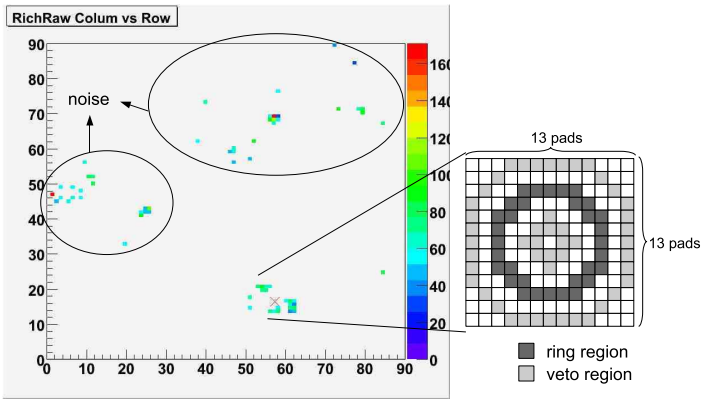


Fig. 3. Fixed-diameter ring recognition on the RICH detector

parallel for 96 columns on 12 Xilinx FPGAs [11] [12]. To treat all the pixels as ring centers is not only computation inefficient, but also resouce consuming on FPGAs. In addition, it requires extra work to correlate the RICH results with the rest detector system (especially inner MDCs) in the offline analysis. In order to simplify ring recognition and correlate the RICH pattern with the inner MDC tracking information, identified particle tracks in inner MDCs are introduced to point out potential ring centers. A LUT is employed to convert the coordinate of track points on the MDC projection plane into the one of potential ring centers on the RICH plane. To take into account the coordinate conversion error due to the resolution difference from MDC to RICH, we normally select all neighboured pixels (e.g. in a search window of 5 $\times$ 5) as center candidates to search for rings, rather than considering only the single derived center pixel from MDC.

With the small number of specified ring center candidates, we do not have to reconstruct the complete hit information for all the pixels on the RICH plane, but need only traverse all the hit pixels in a single event to judge their positions. If they fall into the ring region of a center candidate, they may come from the valid Cherenkov light generated by flying dileptons; Otherwise they are probably the noise. The position judgment is realized by geometrical calculation on the distance between the hit pixel and the ring center.

4 Implementation

Both algorithms are implemented in the FPGA fabric as hardware processing engines. Figure 4 demonstrates the design structure of the Tracking Processing Unit (TPU) and the Ring Recognition Unit (RRU). In the system design, event data are imported from the detector front-end circuits into the FPGA via optical links [5]. They are continuously supplied to the input FIFO residing in the slave interface design of each algorithm engine. The TPU core extracts the fired wire serial numbers and derives the address information for each wire using

an *address_LUT*. With the storage address of the *projection_LUT* in the DDR2 memory and the position address on the projection plane specified, a master device reads out the *projection_LUT* data for each fired wire, and feeds them to the *accummulate_unit* in which the projection overlap histogram is accummulated. Afterwards the *peak_finder* module figures out the peaks by comparing each pixel with all its eight neighbours. The comparison is conducted simultaneously on all the 128 pixels in a row for parallel processing. The complete projection plane is scaned row by row to identify peaks. Identified peak pixels represent the positions in which particle tracks penetrate on the projection plane. These results are both collected in the output FIFO for being recorded, and induced into the *ring_center_buffer* to specify potential ring centers for RICH ring recognition.

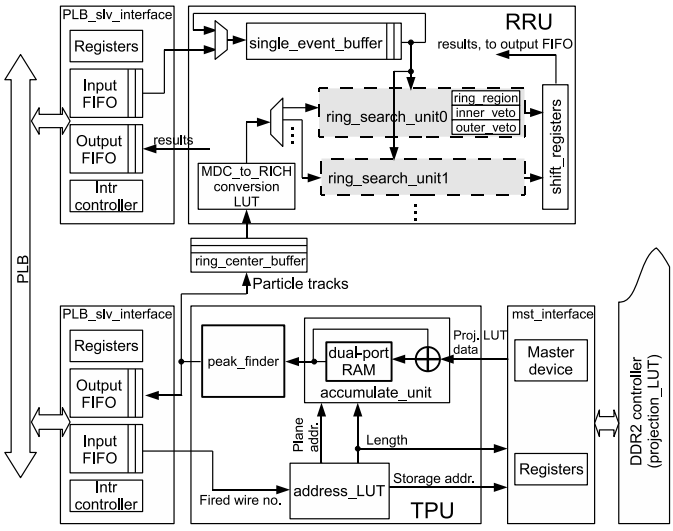


Fig. 4. Hardware design of the algorithm engines

The incoming RICH sub-events specify hit pixels on the RICH plane. All the hit pixels belonging to the same event are to be extracted and buffered in the *single_event_buffer* in RRU. Meanwhile, ring center candidates are derived from particle tracks, converting their position coordinates in the MDC projection plane into the RICH plane with a LUT. Therefore ring pattern search is conducted in *ring_search_units* within the mask regions of the specified ring center candidates. The number of parallel *ring_search_units* is configurable according to the available resources on the FPGA. Each *ring_search_unit* takes charge of one ring center pixel as well as its neighboured pixels in a search window, which are also deemed as center candidates for avoiding the coordinate conversion error from MDC to RICH. Specifically for a search window of 5×5, there exist in fact 25 processing cores in each *ring_search_unit* corresponding to the single derived center candidate and its 24 neighbours. Loading the ring center candidates in *ring_search_units* is also in the unit of a same event. If the number of configured *ring_search_units* is larger than or equal to the center candidate count of an event (i.e. the number of found particle tracks from MDCs), the hit pixels in *single_event_buffer* can be simply read out and traversed for deciding their positions and identifying rings from the accummulated values of *ring_region*, *inner_veto_region* and *outer_veto_region*. Otherwise ring centers have to be loaded in *ring_search_units* for multiple computation rounds, and accordingly all the hit pixels belonging to this event must be reiterated in the *single_event_buffer* until all the centers are done. After each round computation, the center candidates with recognized ring patterns are shifted out and collected in the output FIFO for result archiving.

Both TPU and RRU feature a parallel and pipelined design structure. With many processing units instantiated for histogram accummulation and peak finding in TPU, and ring pattern search in RRU, the hardware engines accelerate the application-specific computation even though they work at lower clock frequencies than General-Purpose microprocessors (GPCPU). Moreover, fine-grained memory inside the FPGA using Block RAMs is rather efficient to implement fast memory accesses such as LUTs. This also contributes to the hardware acceleration on pattern recognition algorithms.

5 Results

5.1 Implementation Results

Resource utilizations of RRU with 1 or 2 *ring_search_unit* configurations are listed in Table 1, as well as the TPU design. Interface blocks are included in the reported results. We observe that both TPU and RRU consume a reasonable fraction of available resources on a Xilinx Virtex-4 FX60 FPGA. Taking into account the system design which consists of embedded processor, Multi-Port Memory Controller (MPMC), peripherals plus the algorithm engines, it is feasible to implement the complete system design on a single Virtex-4 FX60 FPGA for particle recognition computation. The resource utilization is acceptable and still enables the possibility to upgrade the system in future designs.

Resources	RRU (1 ring_search_unit)	RRU (2 ring_search_units)	TPU
4-input LUTs	4533 out of 50560 (9.0%)	8186 (16.2%)	6072 (12.0%)
Slice Flip-Flops	3033 out of 50560 (6.0%)	5190 (10.3%)	3315 (6.6%)
Block RAMs	31 out of 232 (13.4%)	31 (13.4%)	48 (20.7%)

TABLE 1
Resource utilization

The timing reports reveal that the TPU module can run at 125 MHz maximumly. To match the speed of the PLB bus as well as the interface design, its clock frequency is chosen as 100 MHz in practice. The RRU design features two clock domains: The PLB interface runs at 100 MHz, and the RRU core can run at a frequency up to 160 MHz.

These two clock domains are coupled by asynchronous input and output FIFOs.

5.2 Performance Estimation

Experimental measurements have been done on the TPU design with various wire multiplicities, meaning that different numbers of wires in different positions are fired for each event. A single TPU core is observed to achieve 10.8 to 24.3 times speedup than an Intel Xeon 2.4 GHz CPU core in the particle track recognition processing. With respect to the RRU design, the improved approach with specified ring centers from particle tracks can outperform the previous system described in [11] and [12] by about one hundred times. Roughly estimating, one TPU/RRU pair achieves a computation capability equivalent to several tens up to hundred of commodity PCs for HADES particle recognition. In our customized computation platform [5], up to 70 Xilinx Virtex-4 FX60 FPGAs are accommodated within one ATCA shelf. Therefore many TPU and RRU cores can be instantiated and distributed on the FPGA cluster. They will work together to cope with the enormous raw data rate from the particle detectors, in a fashion of Single-Instruction-Multiple-Data (SIMD). One ATCA shelf full of 70 FPGAs implies an equivalent processing capability of thousands of commodity PCs for the particle recognition computation in the HADES experiment.

6 Conclusion

We have presented the particle recognition computation for the HADES experiment, specifically particle track reconstruction and Cherenkov ring recognition. The algorithms are introduced in the context of concrete particle detectors. Hardware processing engines are implemented in the FPGA fabric. Implementation results reveal the feasibility to integrate the processing engines in the system design on FPGAs. Performance estimation has been conducted with experiments. It is foreseen to achieve a computation capability equivalent to thousands of commodity PCs with the FPGA cluster in one ATCA shelf for the particle recognition computation in the HADES experiment.

References

[1] High Acceptance Di-Electron Spectrometer (HADES) @ GSI 2008. Darmstadt, Germany, “www-hades.gsi.de”.

[2] antiProton ANnihilations at DArmstadt (PANDA) @ GSI 2008. Darmstadt, Germany, “www.gsi.de/panda”.

[3] The Large Hadron Collider (LHC) @ CERN 2008. Geneva, Switzerland, “http://lhc.web.cern.ch/lhc/”.

[4] Froehlich, I., et al. 2004. Pattern recognition in the HADES spectrometer: an application of FPGA technology in nuclear and particle physics, *In Proc. of the IEEE International Conference on Field-Programmable Technology*, pp. 443-444.

[5] M. Liu, Johannes Lang, et al., “ATCA-based Computation Platform for Data Acquisition and Triggering in Particle Physics Experiments”, *In Proc. of the International Conference on Field Programmable Logic and Applications*, Sep. 2008.

[6] K. Anikeev, et al., “Event-Building and PC Farm based Level-3 Trigger at the CDF Experiment”, *In Proc. of the IEEE NPSS Real Time Conference*, Jun. 1999.

[7] A. Gellrich, “A Large Linux-PC Farm for Online Event Reconstruction and Data Management at HERA-B”, *In Proc. of the International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, Oct. 2000.

[8] A. Gregerson, “FPGA Design Analysis of the Clustering Algorithm for the CERN Large Hadron Collider”, *In Proc. of the IEEE Symposium on Field Programmable Custom Computing Machines*, Apr. 2009.

[9] Hinkelbein, C., Kugel, A., Manner, R., Muller, M., Sessler, M., Simmler, H., and Singpiel, H. 2001. Pattern Recognition Algorithms on FPGAs and CPUs for the ATLAS LVL2 Trigger, *IEEE Transactions on Nuclear Science*, Volume 48, Issue 3, Part 1, pp. 296-301.

[10] P. A. Cherenkov, “Visible Radiation Produced by Electrons Moving in a Medium with Velocities Exceeding that of Light”, *Physics Review*, volume 52, pp. 378 - 379, Aug. 1937.

[11] J. Lehnert, et al., Ring Recognition in the HADES Second-level Trigger, *Nuclear Instruments and Methods in Physics Research A 433*, pp. 268 - 273, 1999.

[12] J. Lehnert, et al., Performance of the HADES Ring Recognition Hardware, *Nuclear Instruments and Methods in Physics Research A 502*, pp. 261 - 265, 2003.