FPGA-Based Cherenkov Ring Recognition in Nuclear and Particle Physics Experiments

Ming Liu[1,2], Zhonghai Lu[2], Wolfgang Kuehn[1], and Axel Jantsch[2]

[1] II. Physikalisches Institut, Justus-Liebig-Universität Giessen, Germany
[2] Department of Electronic Systems, Royal Institute of Technology, Sweden
{ming.liu,wolfgang.kuehn}@physik.uni-giessen.de
{mingliu,zhonghai,axel}@kth.se

Abstract. Cherenkov ring is often adopted to identify particles flying through the detector systems in nuclear and particle physics experiments. In this paper, we introduce an improved ring recognition algorithm and present its FPGA implementation. Compared to the previous implementation based on VMEBus and FPGAs, our design is evaluated to outperform by several tens up to hundred times with acceptable resource utilizations on a Xilinx Virtex-4 FX60 FPGA. The design module will reside in the online data acquisition (DAQ) and trigger facilities, and contribute to significantly reduce the data rate of storage for offline analysis by retaining only interesting events and dropping the noise. Our customized FPGA cluster in one ATCA [1] shelf is foreseen to achieve an equivalent computation capability up to thousands of commodity PCs for particle recognition.

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 [2] and PANDA [3] at GSI Germany, BESIII [4] at IHEP China, ATLAS, CMS, LHCb, ALICE at the LHC [5] 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 recognize emitted particles after the collision and analyze their characteristics. Figure 1 shows the exploded view of the HADES detector system as an example.

The Cherenkov effect was discovered by the Russian experimentalist P. A. Cherenkov in 1934, and explained by the Russian theorists I. Y. Tamm and I. M. Frank. All three scientists recieved the Nobel prize in physics in 1958 because of this discovery. It describes the emission of light (Cherenkov radiation) that occurs in a transparent substance, when a charged particle travels through the

A. Koch et al. (Eds.): ARC 2011, LNCS 6578, pp. 169–180, 2011.

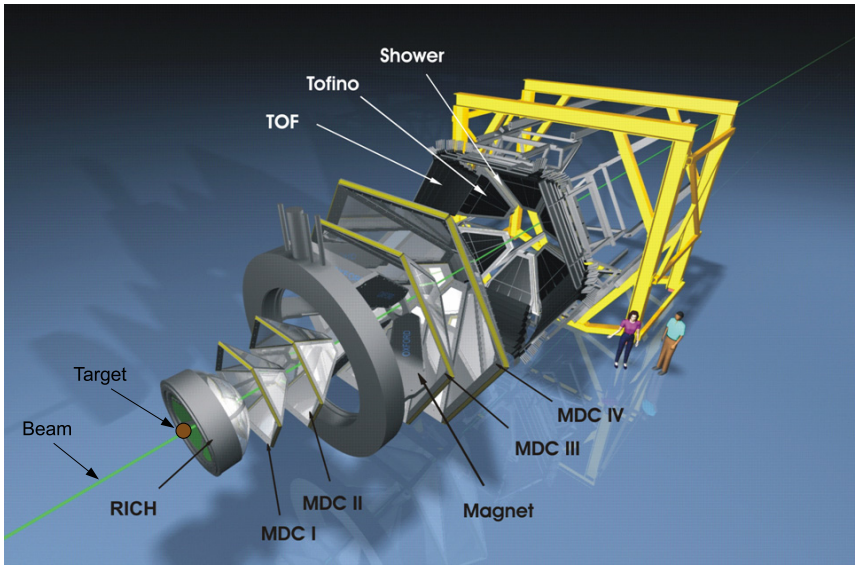

Fig. 1. Exploded view of the HADES detector system (RICH detector for Cherenkov ring recognition)

material with a speed faster than the speed of light in that material [6]. This process is analygous to the shock wave of sound generated by a jet flying faster than the speed of sound in air. Based on the Cherenkov effect, the Ring Imaging CHerenkov (RICH) detector is widely adopted in nuclear and particle physics experiments, to determine the velocity of charged particles and identify particle types. For instance in the HADES experiment, RICH detector together with the Cherenkov ring recognition algorithm is used to identify dilepton pairs in hadron and heavy ion induced nuclear reactions.

Taking into account the experiment time lasting for months and huge data rates ranging from 10^7 to 10^{11} Bytes/s generated by detectors, it is neither realistic nor necessary to entirely record the experimental data for offline analysis. Therefore it is essential to realize an efficient online data acquisition (DAQ) and trigger system which process the events[1] and reduce the data rate by several orders of magnitude by means of rejecting background noise. In the contemporary facilities, pattern recognition algorithms [7] [8] [9] including Cherenkov ring recognition, are implemented as sophisticated criteria to select interesting events which possess expected patterns generated by certain types of particles. Uninteresting background data will be discarded on the fly.

The remainder of the paper is organized as follows: The previous implementation of Cherenkov ring recognition is addressed in Section 2. The HADES

[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.

detector system and the algorithm principle of Cherenkov ring recognition are explained in Section 3. In Section 4, on-FPGA hardware design is presented. Implementation and performance results will be discussed in Section 5. Finally we conclude the paper and propose our future work in Section 6.

2 Previous Work

The Cherenkov ring recognition computation was previously implemented on FPGAs in [10] and [11], and the design is running in the current HADES DAQ and trigger system. The designers accommodate the computation on 12 dedicated FPGA-based VMEBus cards to search for valid rings appearing on the total 6 trapezoidal sectors of the RICH detector. In their original design, ring recognition for the RICH detector works as separate modules and does not correlate with identified particle tracks from the Mini Drift Chamber (MDC) detectors: The RICH sub-event data are introduced from detectors into the VME cards. Afterwards the complete hit information of the RICH plane is reconstructed in a memory device, distinguishing hit and non-hit pixels. Next ring patterns are searched with all the pixels on the RICH plane regarded as possible ring centers. In order to obtain hardware acceleration over software, the total 96 columns are processed in parallel on 12 Xilinx FPGAs. Despite the parallel structure of the original design, we still believe it computation-inefficient due to the following reasons: Firstly, the hit information of the RICH plane has to be reconstructed before ring search, by filling all the pixels with "hit" or "non-hit"; Secondly, "blind" search considering all the pixels as possible ring centers loses computation efficiency, taking into account the fact that the interesting ring patterns do not occur very frequently; Finally, it results in an inadequate data reduction rate in heavy ion reactions for mass storage and offline analysis, due to the lack of correlation with MDC particle tracks. In the recent HADES upgrade project, we will take the advantage of the MDC track reconstruction results [9] and improve the Cherenkov ring recognition algorithm as well as its FPGA implementation for higher performance. The principle difference of the improved algorithm will be clarified in detail in Section 3.

3 Algorithm Description

3.1 RICH and MDC Detectors in HADES

As an experiment example using the RICH detector to identify particles, Figure 1 shows the exploded view of the HADES detector system and Figure 2 shows the lateral cut-away view. In the experiment, accelerated energetic beam particles collide with target particles. New types of particles are therefore emitted from the collision and fly through the detector system. With the help of the strong magnetic field generated by the superconducting coil, inner (I and II) and outer (III and IV) MDC detectors are employed to reconstruct the flying tracks of particles and further investigate their momentum [9]. When light and

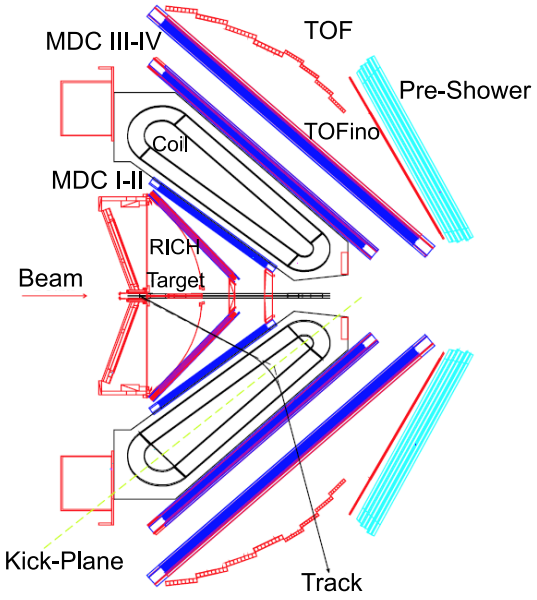

Fig. 2. Lateral cut-away view of the HADES detector system

fast particles (for instance dileptons in HADES) fly through the inner MDCs from the target, the generated Cherenkov light cone is reflected by the mirror and displayed on the RICH detector in the shape of a ring.

3.2 MDC Track Reconstruction

MDC detectors are employed to reconstruct the particle tracks entering and leaving the magnetic field, for further deriving the deflection angle inside it. The magnetic field does not penetrate into the inner or outer MDCs and exists only within the coil space. Thus particle tracks only bend in the magnetic field, and the segments before or behind the coil could be approximately described by straight lines. Detailed discussion on MDC tracking is out of the range of this paper. What we need only to understand, is that the identified particle tracks will point to the Cherenkov ring centers if they are also reflected by the mirror onto the RICH plane, because light cones are simply generated along the particle flying tracks.

3.3 Cherenkov Ring Recognition

As a demonstration example, Figure 3 displays the captured image frame of one sub-event on a RICH sector whose resolution is 96 × 96 pixels (pads). 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.

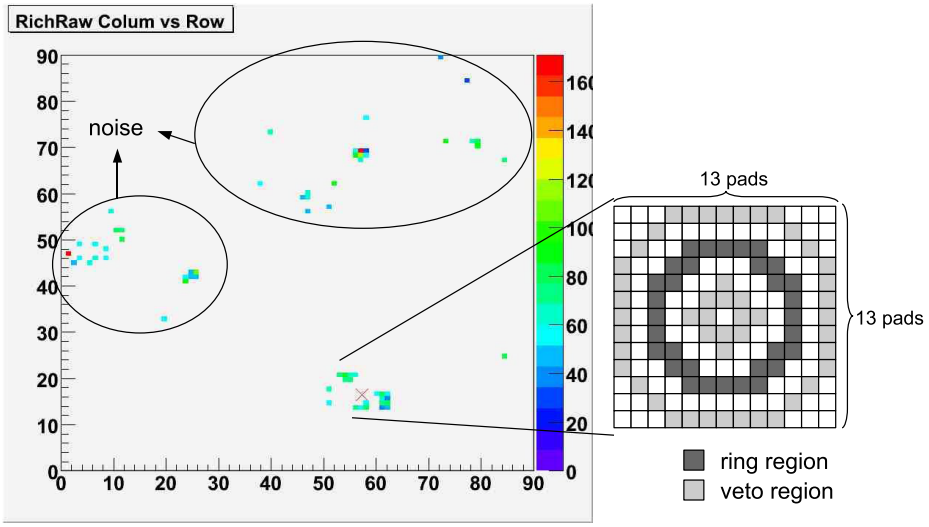

Fig. 3. Ring recognition on the RICH detector

For each potential ring center, the pattern search is conducted within a fixed mask region of 13 × 13 pixels. The hit number on the ring region with a radius of 4 pixels are counted with the value *ring_region*. In order to shape the ring pattern rather than a large piece of solid region full of hits, two veto regions inside and outside the ring region are defined with their hit pixel counts also accumulated. Therefore 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 derived through physics analysis and they are expected to be programmable during the experiment.

Because of the constant diameter of ring patterns, the computation challenge falls on the position identification of ring centers. In the original design of [10] and [11], the designers treat all the 96 × 96 pixels on the RICH plane as potential ring centers: The system first receives RICH sub-event raw data containing the position of all hit pixels from the detector readout circuits. Then 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 regarded as possible ring centers, in parallel for all 96 columns on 12 Xilinx FPGAs [10] [11]. 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 further 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. The particle tracks are to be reflected by the mirror onto the RICH plane. Hence the coordinate of the track penetration points on the MDC projection plane is converted into the one of potential ring centers on the

```
-- (x1, y1) : position coordinate of the hit pixel
-- (x0, y0) : position coordinate of the ring center candidate
-- x_distance = abs(x1 – x0); : distance on x axis from the hit pixel to the center candidate
-- y_distance = abs(y1 – y0); : distance on y axis from the hit pixel to the center candidate
-- hop =  x_distance + y_distance; : hop distance from the hit pixel to the center candidate

position_result <= FARAWAY_NOISE_REGION; -- by default the hit pixel is deemed as noise

if (hop <=2) then
      position_result <= INNER_VETO_REGION; -- the hit pixel falls into inner_veto_region
end if;

if (((hop = 5 or hop = 6) and x_distance <= 4 and y_distance <= 4)
  or (y_distance = 0 and x_distance = 4)
  or (y_distance = 4 and x_distance = 0)) then
      position_result <= RING_REGION;            -- the hit pixel falls into ring_region
end if;

if ((hop = 9 and x_distance <= 6 and y_distance <= 6)
  or (y_distance = 6 and x_distance <= 2)
  or (x_distance = 6 and y_distance <= 2)) then
      position_result <= OUTER_VETO_REGION; -- the hit pixel falls into outer_veto_region
end if;
```

Fig. 4. Position judgment of a hit pixel to a ring center candidate

RICH plane. To avoid complex online geometrical calculation, the coordinate conversion task is done offline to arrive at a Look-Up Table (LUT). Taking into account the coordinate conversion error due to the resolution difference from MDC to RICH, normally we map a single particle track to multiple neighboured pixels in a search window (e.g. 5×5) on the RICH plane. Hence interesting ring patterns can be prevented from being ignored due to the slight coordinate conversion error.

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 belonging to the same RICH sub-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 to be discarded. The position judgment is realized by geometrical calculation on the distance between the hit pixel and the ring center, as demonstrated by the VHDL-syntax code in Figure 4.

After the position judgment of all the hit pixels corresponding to all the potential ring centers, the hit counts in the ring region as well as in the inner/outer veto regions can be accumulated. They will be compared to the thresholds for determining whether a ring pattern is successfully identified or not. Only the data with identified patterns are to be retained. The noise will be discarded on the fly for reducing the data rate before storage.

4 Implementation

4.1 Ring Recognition Unit Design

The Cherenkov Ring Recognition Unit (RRU) is described in VHDL and implemented on our customized Compute Node (CN) based on the Advanced Telecommunications Computing Architecture (ATCA) backplane and Xilinx Virtex-4 FX60 FPGAs [12]. Figure 5 illustrates its design structure in block diagram, and Figure 6 shows the overall system design on the FPGA with RRU incorporated. The incoming event data via optical links are buffered in on-board DDR2 memory. Afterwards, RICH sub-events are supplied into the input FIFO by DMA transfers via the system PLB bus. The coordinate information of hit pixels belonging to the same event are extracted and further buffered in *single_event_buffer* in the RRU module. Meanwhile, ring center candidates are imported from the MDC tracking results [9]. The coordinates of track points are reflected onto the RICH plane, converting them into ring center candidates with an offline-built LUT. Then potential ring centers are loaded into *ring_search_units*. The number of *ring_search_unit* is configurable according to the available resources on the FPGA. Loading the ring center candidates is also in the unit of a same event: If the number of configured *ring_search_unit* is larger than or equal to the center candidate count of an event (i.e. the number of recognized particle tracks from MDCs), the hit pixel data in *single_event_buffer* can be simply read out and traversed for deciding their positions and identifying rings from the accumulated values of *ring_region*, *inner_veto_region* and *outer_veto_region*. Otherwise ring centers have to be loaded into *ring_search_unit* in multiple computation batches, and accordingly all the hit pixels belonging to this same event must be looped back and reiterated in *single_event_buffer* until all the center candidates are done. After each round computation, the coordinates of center candidates with recognized rings are shifted out and collected in the output FIFO, to be exported by Gigabit Ethernet for result mass storage in the PC farm.

In each *ring_search_unit* shown in Figure 5, not only the directly derived RICH plane pixel from an MDC particle track is loaded as a potential center candidate for ring search, but also its neighbours in a region (e.g. 24 neighbours in a search window of 5×5) considering the coordinate conversion error from MDC to RICH. We name the computation cores for the neighbour pixels *shadow cores*. Thus for each *ring_search_unit*, there exist in fact 25 cores (1 core for the track derived pixel and 24 shadow cores for its neighbours) working in parallel. The parallel infrastructure and the pipelined design guarantee RRU a high performance in the ring recognition computation.

In the practical experiment, the occupancy of hit pixels (the proportion of hits out of the total 96×96 pixels) on the RICH plane is normally rather small. Among them, valid ring patterns are even rare. Therefore in contrast to blind search scanning all the pixels on the plane, our new design traverses only the hits in a pipelined fashion. With respect to various ring centers specified by identified particle tracks, the processing is conducted in parallel *ring_search_units*. This approach will lead to significantly higher performance, as we can observe in Section 5.

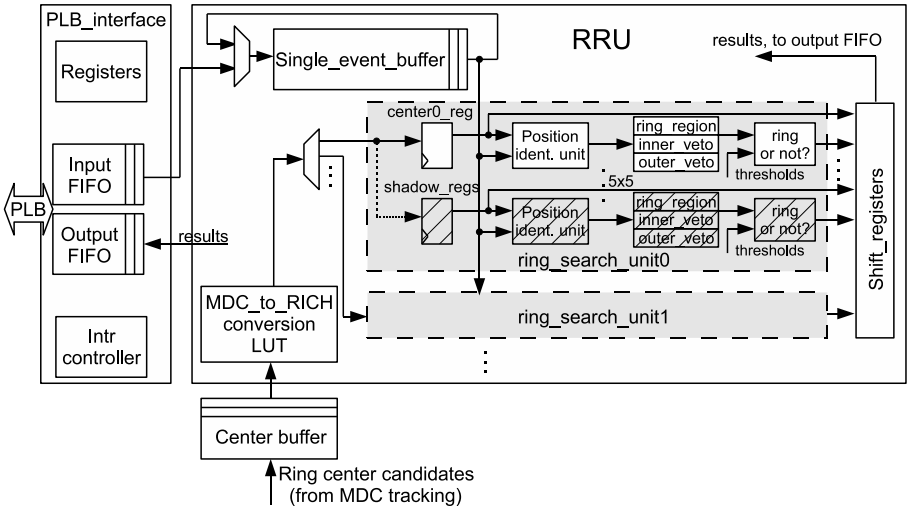

Fig. 5. Design structure of RRU

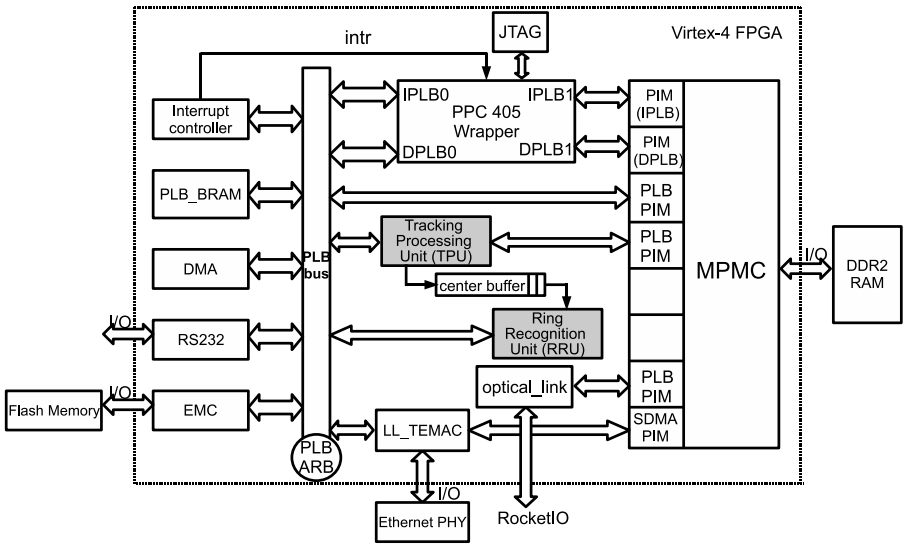

Fig. 6. System design with algorithm engines

4.2 System Integration

Together with the MDC Tracking Processing Unit (TPU) [9], RRU will be integrated in the system-on-an-FPGA as an algorithm processing engine (see Figure 6). Incoming event data are received via the optical link from detector readout circuits, and buffered in the DDR2 memory under the control of the Multi-Port Memory Controller (MPMC). Afterwards MDC and RICH data are

respectively supplied by DMA transfers to TPU and RRU, which have a buffer in between to deliver identified particle tracks to RRU for pointing out ring center candidates. All the processed results will be again collected by DMA back into DDR2, and exported to the PC farm for mass storage through Gigabit Ethernet. In the system, a hardcore PowerPC 405 processor with embedded Linux OS achieves slow controls and conducts TCP/IP stack processing for Ethernet transmission.

5 Experimental Results

5.1 Implementation Results

Resource utilizations of RRU with 1 or 2 *ring_search_unit* configurations are listed in Table 1. We observe that both designs consume less than one sixth of available resources on the Xilinx Virtex-4 FX60 FPGA. The overall system design (shown in Figure 6) with a single *ring_search_unit* RRU consumes less than half of the Virtex-4 FX60 FPGA. The resource utilization is acceptable and still enables the possibility to extend the system in future designs.

Table 1. Resource consumption

Resources	**RRU** (1 ring_search_unit)	**RRU** (2 ring_search_unit)	**system** (RRU of 1 ring_search_unit)
4-input LUTs	4723 out of 50560 (9.3%)	8186 (16.2%)	21933 (43.4%)
Slice Flip-Flops	3663 out of 50560 (7.2%)	5190 (10.3%)	17185 (34.0%)
Block RAMs	31 out of 232 (13.4%)	31 (13.4%)	104 (44.8%)

The complete RRU design features two clock domains: One is the PLB interface and the other is the RRU core. Two clock domains are coupled by input and output asynchronous FIFOs. To match the system PLB bus speed, the interface runs at 100 MHz. The RRU core can run at a maximum frequency of 160 MHz, according to the timing constraints.

5.2 Performance Measurements

To evaluate the performance of the RRU design, we pick some RICH and MDC sub-events from the HADES experimental data and initialize them in the DDR2 memory. DMA takes charge of supplying the raw data to RRU and TPU, and collecting their results back into DDR2. According to the measurements, we observe that a typical interesting HADES event with 3 MDC particle tracks or Cherenkov rings takes a single *ring_search_unit* RRU about 2000 ns for ring recognition computation. For pure noise events without any interesting pattern, the processing time may further be reduced to hundreds of ns or even less. Pure noise events require little computation effort and result in the best-case processing speed (e.g. $\sim$10 MSub-events/s). In the worst case when all the events

contain 3 ring patterns (i.e. 3 dilepton pairs. This will never be possible in practice, because interesting events occur very rarely in experiments.), the 2000 ns processing time implies a processing speed of about 0.5 MSub-events/s. With more practical estimation of the mixture of rare interesting events and large proportion of uninteresting noise events, the processing speed can be reasonably estimated in the order of magnitude of several MSub-events/s[1]. In [11], the authors reported a speed of 49 KSub-events/s of their system. Thus we conclude that a single *ring_search_unit* in our RRU design can outperform the previous system consisting of 2 VME cards and 12 obsolete FPGA chips for each detector sector, by several tens up to hundred times. The performance speedup mainly comes from the algorithm principle improvement of introducing MDC particle tracks to point out ring centers. In addition, the fully optimized design running at high clock frequencies also contribute to the performance enhancement. Further tests using large amount of experimental data will be scheduled in the final system verification stage.

According to the experimental results reported in [9], each single TPU core can achieve a processing speedup of about 20 times than an Intel Xeon 2.4 GHz CPU core for particle track reconstruction. Together with RRU, the system design on a single Virtex-4 FX60 FPGA is roughly estimated to achieve a computation capability equivalent to several tens up to hundred of commodity PCs for HADES particle recognition. In the customized computation platform described in [12], up to 70 FPGA chips are accommodated within one ATCA shelf. 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 form of Single-Instruction-Multiple-Data (SIMD). Therefore one ATCA shelf full of 70 FPGAs implies an equivalent processing capability of about thousands of commodity PCs for the particle recognition computation in the HADES experiment.

6 Conclusion and Future Work

In this paper, we have presented a new Cherenkov ring recognition algorithm and its FPGA implementation. Correlated with the particle track reconstruction algorithm, the RRU module identifies flying particles with ring patterns and significantly removes the noise in the online DAQ and trigger systems of nuclear and particle physics experiments. With acceptable resource utilizations, a single RRU module can outperform the previous system design by several tens up to hundred times. The complete ATCA platform consisting of 70 Xilinx

[1] The exact event processing speed is heavily dependent on the data stream constitution consisting of interesting events with ring patterns and noise data. Therefore from the experimental statistics in the long run, the effective processing speed is between the best-case and the worse-case performance, but very close to the best case due to the appearance rarity of interesting events. Hence it is only feasible but meaningful to estimate the system performance within the accuracy of one order of magnitude.

Virtex-4 FX60 FPGAs, is estimated to achieve an equivalent processing capability of thousands of commodity PCs for particle recognition computation. The improved computation efficiency will largely reduce the hardware costs, increase the online data reduction rate, and focus the offline analysis on the data which most probably interest the physicists.

In our future work, various algorithm engines specifically the RRU module for Cherenkov ring recognition and the TPU module for MDC tracking, are to be dynamically reconfigured during the system run-time. They will be chosen as a real application to verify the design framework of self-aware adaptive computing [13]. Through dynamically loading or unloading different modules into or out of the same reconfigurable region, more efficient resource utilization is foreseen and higher performance/area ratio may be achieved in comparison with the conventional static design approach on FPGAs.

Acknowledgment

This work was supported in part by BMBF under contract Nos. 06GI9107I and 06GI9108I, FZ-Juelich under contract No. COSY-099 41821475, HIC for FAIR, and WTZ: CHN 06/20.

References

1. PCI Industrial Computers Manufactures Group (PICMG), PICMG 3.0 Advanced Telecommunications Computing Architecture (ATCA) specification (December 2002)
2. High Acceptance Di-Electron Spectrometer (HADES) @ GSI, Darmstadt, Germany (2008), `http://www-hades.gsi.de`
3. antiProton ANnihilations at DArmstadt (PANDA) @ GSI, Darmstadt, Germany (2008), `http://www.gsi.de/panda`
4. BEijing Spectrometer (BES) @ IHEP, Beijing, China (2008), `http://bes.ihep.ac.cn/bes3/index.html`
5. The Large Hadron Collider (LHC) @ CERN, Geneva, Switzerland (2008), `http://lhc.web.cern.ch/lhc/`
6. Cherenkov, P.A.: Visible Radiation Produced by Electrons Moving in a Medium with Velocities Exceeding that of Light. Physics Review 52, 378–379 (1937)
7. Froehlich, I., et al.: 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 (December 2004)
8. Hinkelbein, C., Kugel, A., Manner, R., Muller, M., Sessler, M., Simmler, H., Singpiel, H.: Pattern Recognition Algorithms on FPGAs and CPUs for the ATLAS LVL2 Trigger. IEEE Transactions on Nuclear Science 48(3) Part 1, 296–301 (2001)
9. Liu, M., Kuehn, W., Lu, Z., Jantsch, A.: System-on-an-FPGA Design for Real-time Particle Track Recognition and Reconstruction in Physics Experiments. In: Proc. of the EUROMICRO Conference on Digital System Design (September 2008)
10. Lehnert, J., et al.: Ring Recognition in the HADES Second-level Trigger. Nuclear Instruments and Methods in Physics Research A 433, 268–273 (1999)

11. Lehnert, J., et al.: Performance of the HADES Ring Recognition Hardware. Nuclear Instruments and Methods in Physics Research A 502, 261–265 (2003)
12. Liu, M., Johannes, L., 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 (September 2008)
13. Liu, M., Lu, Z., Kuehn, W., Yang, S., Jantsch, A.: A Reconfigurable Design Framework for FPGA Adaptive Computing. In: Proc. of the International Conference on ReConFigurable Computing and FPGAs (December 2009)