FPGA Coprocessor for High-Level Trigger Applications

T. Alt[1,2], K. Aurbakken[1], G. Grastveit[1], H. Helstrup[3], V. Lindenstruth[2], C. Loizides[1,4], J. Nystrand[1], D. Roehrich[1], B. Skaali[5], T. Steinbeck[2], H. Tilsner[2], K. Ullaland[1], A. Vestbø[1]

[1]Department of Physics, University of Bergen, Norway
[2]Kirchhoff Institute for Physics, University of Heidelberg, Germany
[3]Bergen College, Norway
[4]Institute of Nuclear Physics, University of Frankfurt, Germany
[5]Department of Physics, University of Oslo, Norway

Abstract

High-level trigger systems (HLT) are necessary in order to cope with the anticipated data rate of future High-Energy experiments. The planned HLT for the ALICE experiment consists of a cluster of off-the-shelf PCs connected with a high bandwidth network. One of its elements is a custom-made PCI card, which receives the detector data, the so called *Read-Out Receiver Card* (RORC). It is the input port for detector data into the HLT computing farm and is mounted in some of its PCs. The main device of the RORC is an FPGA, which is necessary to implement the PCI-bus protocol. On the other hand, it can also be used to assist the host processor with first-level processing of the incoming raw data like cluster finding or Hough Transform for track reconstruction.

I. The High-Level Trigger of the ALICE Experiment

The main tracking detector of the ALICE experiment at the upcoming LHC will be a large Time Projection Chamber (TPC), which will produce a data volume of up to 75 MByte per event in central Pb-Pb collisions. With an anticipated event rate of about 200 Hz the data rate is approximately 15 GByte/s. In contrast, the maximum affordable bandwidth to the mass storage system is only about 1.2 GByte/s, which is one order of magnitude less than the data rate of the TPC only. Taking into account the rate of interesting physics triggers (jets, open charm, e^+e^-) and the physics information contained in a raw event, the set limit of 1.2 GByte/s is more than adequate in order to record all events. However, the key is in the selection of the relevant information. As a consequence, a system is needed which is able to reduce the date volume online by processing the events doing pattern recognition and simple event reconstruction in order to select interesting (sub) events or to compress data efficiently by modelling techniques [1], [2]. This is the task of the High-Level Trigger (HLT), a massive parallel computing system. This system will consist of a farm of clustered SMP-nodes based on off-the-shelf PCs, which are connected with a high bandwidth low latency network.

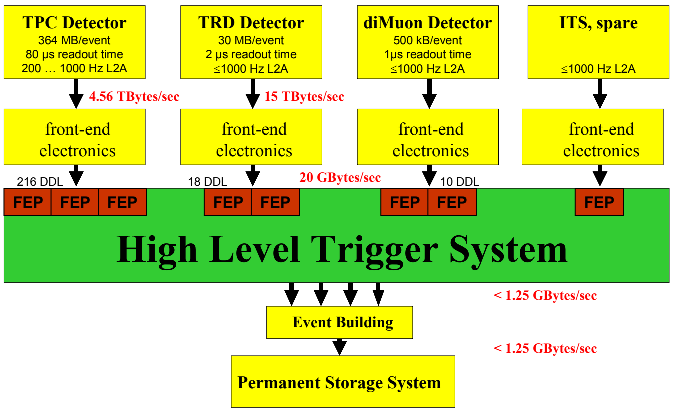

Figure 1: The overall HLT architecture.

The HLT architecture shown in Figure 1 is mainly driven by the ALICE detector hierarchy. Basically, all detectors operate independently and are synchronized by the trigger system. All detectors ship their data via an optical data link (***D****etector* ***D****ata* ***L****ink*, DLL) to the so called ***F****ront* ***E****nd* ***P****rocessors* (FEP), a commercial off-the-shelf-computer equipped with the ***R****ead*-***O****ut*-***R****eceiver*-***C****ard* (RORC). The most obvious interface to be used here is PCI. Therefore, the DDL is mounted on a PCI card which subsequently can be mounted in any computer supporting PCI. The main memory of the computer functions as the event buffer, implementing the most likely cost effective memory. An overview of the architecture of the HLT Front End Processor is shown in Figure 2.

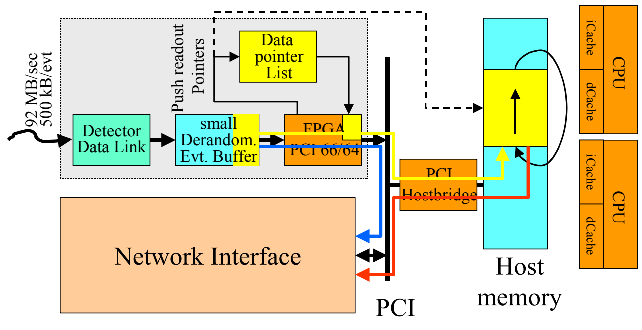

Figure 2: Overview of the HLT Front End Processor (FEP) architecture.

Overall, the HLT receives zero suppressed raw data from about 250 detector links, aggregating 20 GByte/s. The processing rate varies as a function of trigger type and running scenario. In case of heavy ion running, the TPC trigger rate is limited to about 200 Hz in contrast to 1 kHz in case of proton-proton running.

II. THE READ-OUT RECEIVER CARD

A key component of the system is the ***R**ead-**O**ut-**R**eceiver-**C**ard* (RORC), which interfaces the front-end electronics on the detectors with the HLT computing cluster. These custom PCI cards receive the detector raw data via an optical fibre (the so-called Detector Data Link, DDL) and forward the data to the main memory of the host computer. This dataflow is easily implemented by first reserving a large memory block in the host and then filling it with direct memory access (DMA) transfers directly by a DMA engine, which is implemented, on the RORC. The availability of free buffer space can be implemented by pointer lists stored in small FIFO memories on the PCI card, see Figure 2.

Despite its functionality as a PCI device, the FPGA is used to assist the host processor with first level processing while the data is being transferred to the memory of the corresponding node. A photograph of the HLT RORC is shown in 3.

Figure 3: The prototype of the HLT-RORC.

Initialisation of the RORC, or any other PCI device, and all subsequent access to the card is done via bus cycles on the PCI bus. For the host computer running the Linux operating system, an appropriate device driver was developed, the so called ***PCI S**hared Memory **I**nterface* (PSI). The design of this driver is based on a very flexible approach allowing it to be used as a generic driver for a large class of PCI devices. In addition, the driver supports named shared memory blocks, which are used for DMA operations into and out of the computer. Access to different devices or memory blocks is done via a virtual directory structure, which reflects the operating systems' organization of the PCI bus. Another feature of the PSI driver is its hot swap functionality, which allows a reconfiguration of the devices on the PCI bus without rebooting the host computer.

III. ONLINE EVENT RECONSTRUCTION

For the functionality of the HLT fast pattern recognition is crucial [1]. Depending on the multiplicity of the events, which is expected to be in the range 1000 to 8000, this is realized in two different ways. For low multiplicity events (dN/dy < 4000) a sequential feature extraction working on space points is foreseen; cluster finding followed by a track finder. For high multiplicity events there is a large fraction of overlapping clusters, which leads to a reduced tracking efficiency if one uses the same strategy as for low-multiplicity events. In that regime it is foreseen to use an iterative feature extraction scheme, which directly works on the raw data. This includes a Hough Transform as a tracklet finder followed by a detailed analysis and deconvolution of the charge clusters using the found tracklets as starting points.

In both cases the FPGA will be used for the initial data-processing. The incoming raw data enters the circuit through the DDL. The data is organized as a back linked list of sequences as defined by the ALTRO chip [3]. Clusters or track candidates are computed by the Cluster Finder or Hough Transform respectively. The results are transferred via the PCI-bus to the HLT CPUs.

A. Cluster Finder

1) Algorithm

The input data to the cluster finder is a zero suppressed, time ordered list of 10-bit ADC values on successive pads and padrows representing the charge values. Clusters are regions of contiguous ADC-values in the pad-time plane of TPC-data (see Figure 4). Cluster finding consists of determining the regions and calculating the centre of gravity for each region/cluster.

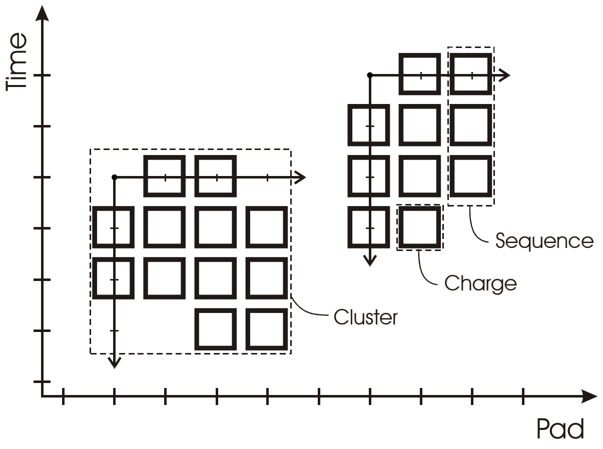

Figure 4: Terms used in the discussion of the Cluster Finder.

2) VHDL Implementation

The FPGA implementation of the cluster finder comprises of four components as shown in Figure 5. The *Decoder* decodes and interprets the incoming raw data, which consists of the ADC sequences at dataflow rate. Since the data are zero-suppressed every sequence of charges is tagged with the time of the first charge. The *Decoder* accumulates the charges that make a sequence, thereby calculating the total charge of the sequence and the centre of gravity in time direction. The individual ADC values are not needed after the calculations and are therefore no longer stored. After the information about a sequence is assembled it is sent to the second part, the *Merger*. A FIFO is used to buffer the sequence data to decouple both processing units. This is needed because the *Decoder* converts the narrow fixed rate data stream into wide data words of varying rate, and furthermore the processing speed of the Merger is dependent on cluster-geometry.

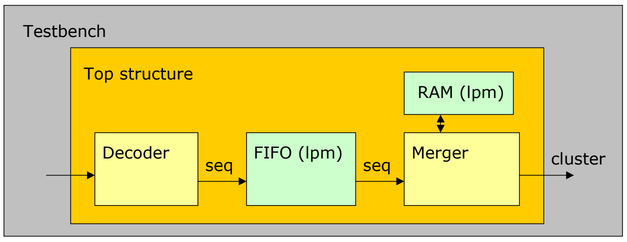

Figure 5: Block diagram of the FPGA cluster finder.

The *Merger* reads the sequences from the FIFO and merges sequences on adjacent pads. The output of the *Merger* is the coordinates of the clusters. In order to determine if an incoming sequences matches a sequences on an adjacent pad some of the previous sequences must be remembered. Since by definition a cluster does not contain holes only the immediately preceding pad needs to be searched. Therefore, only two lists are needed. One list contains the started clusters of the previous pad while the other list contains started clusters already processed at the current pad. The list of clusters on the current pad will be searched when a sequence on the next pad arrives. If that happens, there will be no matches for the remaining clusters on the preceding pad. Hence, everything in the old search range is sent to the output and the lists are exchanged. The old list becomes free memory, the current list becomes the old, and a new current list with no entries is created. The manipulation of the lists is implemented with a ring buffer in RAM [4].

To keep processing fast, relative scales are used for each individual cluster (see Figure 4). This reduces the range of one factor in all multiplications thus speeding up arithmetic. The method limits the maximum size of the clusters. To compensate for this, and to increase the efficiency of the Cluster Finder, a simple deconvolution has been added. The clusters are split at local minima in the time and the pad direction. Hence Centres of Gravity that correspond to closely spaced tracks can be calculated even if some clusters originally overlapped.

3) *Verification and Timing Results*

For verification purposes the system was simulated by a test bench. The test bench reads in an ASCII file containing simulated AliROOT raw data in an ALTRO like back-linked list, which is sent to the *Decoder*. Found clusters of the *Merger* circuit are directed back to the test bench which writes the results to a file. That file is then compared to a result file created by C++ code running nearly the same cluster finder algorithm on the same data set. The found clusters show a very good agreement as shown in Figure 6.

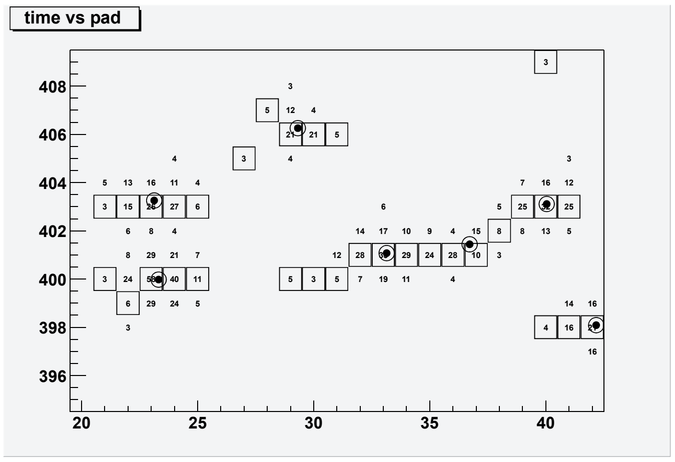

Figure 6: Verification of the cluster finder circuit. Numbers represent charges on a given pad row. Squares are mean values of the sequences that were used for matching. The circle represents the space point determined by the C++ code while the dot is the space point determined by the VHDL model.

4) *Synthesis result*

The Cluster Finder circuit has been synthesized for the ALTERA APEX20KE-400-1x and simulated with back-annotated delays. The Circuit uses 1750 (11%) Logic Elements and 8448 bits (4%) of memory and runs at 50 MHz. Reading 10 bit every clock cycle this yields a maximum data-rate into the Merger of 62.5MB/s. As mentioned one full event is expected to be 75MB. The Cluster Finder is only applicable in low multiplicity events. Thus assuming half of this data volume is distributed to the 216 RORCs, each Cluster Finder must process 170k of data, giving a latency of 2.8 ms. To match the bandwidth of the DDL (150MB/s) the possibility of running two or three Cluster Finders in parallel on each FPGA is being researched.

B. *Hough Transform*

Hough Transform is a standard tool in image analysis, which allows recognition of global patterns in an image space by recognition of local patterns in a transformed parameter space. The basic idea is to find curves that can be parameterised in a suitable parameter space. In its original form one determines a curve in parameter space for a given signal corresponding to all possible tracks with a given parametric form it could possibly belong to [5]. All such curves belonging to the different signals are drawn in parameter space, which is discretised, and the entries are stored in a histogram. In this histogram one track is represented by a peak at the corresponding parameters.

In ALICE, the local track model is a helix. In order to simplify the transformation, the detector volume is divided into sub-volumes in pseudo-rapidity. If one restricts the analysis to tracks starting from a common interaction point, then the circular track in the η sub-volume is characterized by only two parameters: the emission angle with the beam axis, and the curvature κ [6]. The transformation is performed from (R,ϕ)-space to (ϕ_0,κ)-space using the equations:

$$R = \sqrt{x^2 + y^2}$$

$$\phi = \arctan\left(\frac{y}{x}\right)$$

$$\kappa = \frac{2}{R}\sin\left(\phi - \phi_0\right).$$

Each active pixel (ADC-value above threshold) then transforms into a sinusoidal line extending over the whole -range in the parameter space. All the corresponding bins in the histogram are incremented with the ADC-value of the transformed pixel. The superposition of all of these point transformations produces a maximum at the circle parameters of the track. Examples are shown in Figure 7 and Figure 8 for a single pion with transverse momentum of 400 MeV. The track recognition is now done by searching for local maxima in the parameter space. Once the track parameters are known, cluster finding on the raw data can be performed by a straightforward unfolding of the clusters [7].

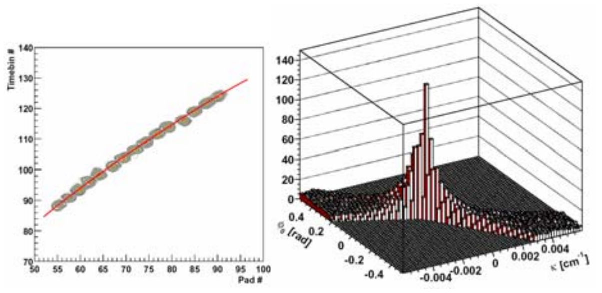

Figure 7: One track segment with clusters on 15 successive pad rows (left) Parameter space representation for a pion with transverse momentum of 400 MeV (right).

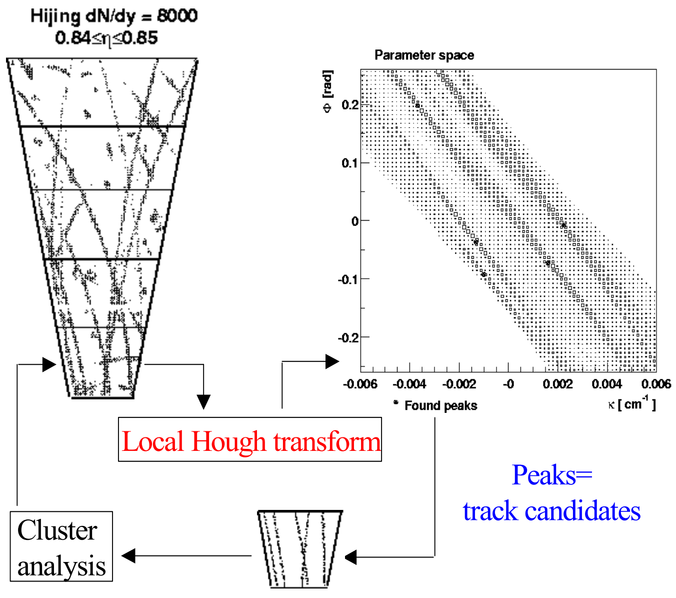

Figure 8: Online data analysis for high multiplicity events.

The foreseen dataflow diagram for the Hough Transform Coprocessor is shown in Figure 9.

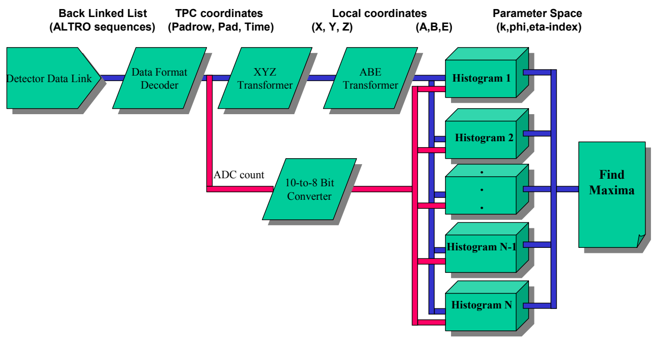

Figure 9: Dataflow diagram of the Hough Transform FPGA coprocessor.

In the *Data Format Decoder* the data will be decoded into a quadruple of padrow, pad, time, and ADC value. Before further processing the ADC values will be converted non-linearly from a 10-bit resolution to 8-bit, resulting in a constant relative accuracy over the whole dynamic range and a reduction of the event size. The conversion is accomplished with a fixed look up table containing 1024 entries.

In the first transformation (XYZ), the padrow, pad, and time coordinates are transformed into local Cartesian coordinates using two additional look up tables. To optimise for hardware implementation the data is transformed into another coordinate system (ABE), reducing the number of trigonometric calculations needed to fill the histograms. Finally, the histograms are searched for peaks above a given threshold. These peaks represent track candidates and for each of these candidates a triplet representing curvature, emission angle, and an index of the pseudo-rapidity interval is send via the PCI bus to the host PC for further processing. At present, a behavioural model of the Hough Transform exists. The model was verified with simulated event data and the results compared to the results of the online C++ code.

IV. SUMMARY

The HLT enables event selection based on physical signatures. This allows for a significant reduction of the data volume, which can be further reduced by storing computed event parameters instead of raw data. A functional small scale HLT running in software exists. FPGA inherent in the readout chain will be utilised for the first stage of online event analysis; Cluster Finding for low multiplicities and the Hough transform for high. The advantage is a significant reduction in the number of CPUs needed in the HLT farm.

A synthesized Cluster Finder model and a behavioural VHDL model of the Hough transform have been simulated and verified by comparing the output to the output of equivalent software.

V. REFERENCES

[1] V. Lindenstruth et. al., *Online Pattern Recognition for the ALICE High Level Trigger*, Proceedings of 13th IEEE-

NPSS Real Time Conference, Montreal, Canada, May 2003.

[2] J. Berger et. al., *TPC Data Compression*, Nucl. Instr. Meth., A 489 (2002) 406.

[3] R. Bosch, A. de Parga, B. Mota, L. Musa, *The ALTRO Chip: A 16-channel A/D Converter and Digital Processor for Gas Detectors*, submitted to the 2002 IEEE NSS/MIC, Nuclear Science Symposium, Norfolk, November 12-14, 2002.

[4] G. Grastveit et. al., *FPGA Co-processor for the ALICE High Level Trigger,* Proceedings of CHEP03 La Jolla, California, March 24-28. 2003.

[5] P. V. C. Hough, *Machine Analysis of Bubble Chamber Pictures*, International Conference on High Energy Accelerators and Instrumentation, CERN, 1959.

[6] D. Brinkmann et. al., *Image data analysis for the NA35 streamer chamber*, Nucl. Instrum. Meth., A354 (1995) 419-436.

[7] U. Frankenfeld, D. Röhrich, B. Skaali, A. Vestbø, *A High-Level Trigger for Heavy Ion Experiments*, Proceedings, 12th IEEE-NPSS Real Time Congress on Nuclear and Plasma Sciences, Valencia, Spain, June 2001.