

OPEN ACCESS

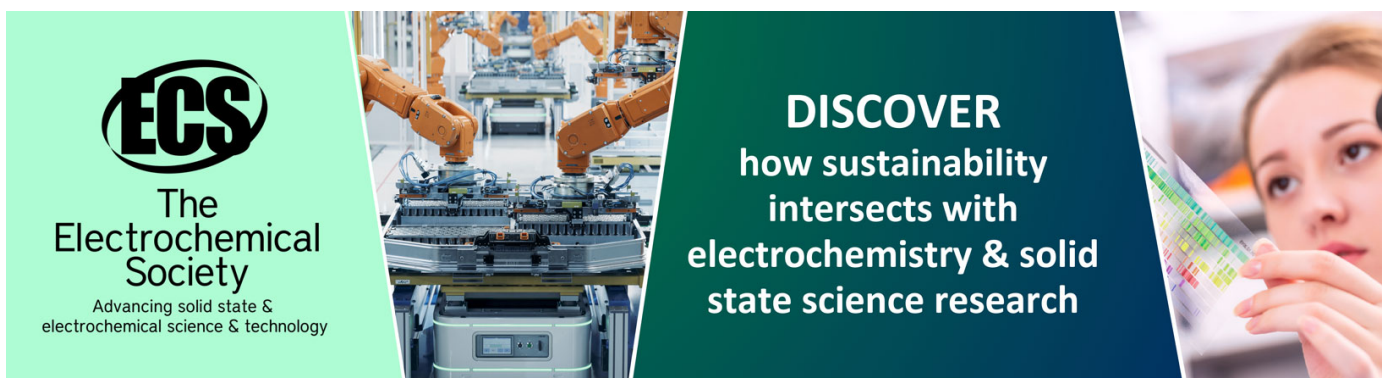
The ALICE high-level trigger read-out upgrade for LHC Run 2

To cite this article: H. Engel *et al* 2016 *JINST* **11** C01041

View the [article online](#) for updates and enhancements.

You may also like

- [Balancing the Resources of the High Level Trigger Farm of the ATLAS Experiment](#)
N Garelli, M T Morar and W Vandelli
- [Operational experience with the ALICE High Level Trigger](#)
Artur Szostak
- [The CMS High Level Trigger System: Experience and Future Development](#)
G Bauer, U Behrens, M Bowen et al.



ECS
The
Electrochemical
Society
Advancing solid state &
electrochemical science & technology

DISCOVER
how sustainability
intersects with
electrochemistry & solid
state science research

TOPICAL WORKSHOP ON ELECTRONICS FOR PARTICLE PHYSICS 2015,
SEPTEMBER 28TH – OCTOBER 2ND, 2015
LISBON, PORTUGAL

The ALICE high-level trigger read-out upgrade for LHC Run 2

H. Engel,^{a,1} T. Alt,^b T. Breitner,^a A. Gomez Ramirez,^a T. Kollegger,^c M. Krzewicki,^b
J. Lehrbach,^b D. Rohr^b and U. Kebschull^a on behalf of the ALICE collaboration.

^a*Institut für Informatik, Johann Wolfgang Goethe-Universität Frankfurt,
Robert-Mayer-Straße 11-15, 60629 Frankfurt, Germany*

^b*Frankfurt Institute for Advances Studies, Johann Wolfgang Goethe-Universität Frankfurt,
Ruth-Moufang-Straße 1, 60483 Frankfurt, Germany*

^c*GSI — Helmholtzzentrum für Schwerionenforschung GmbH,
Planckstraße 1, 64291 Darmstadt, Germany*

E-mail: hengel@cern.ch

ABSTRACT: The ALICE experiment uses an optical read-out protocol called Detector Data Link (DDL) to connect the detectors with the computing clusters of Data Acquisition (DAQ) and High-Level Trigger (HLT). The interfaces of the clusters to these optical links are realized with FPGA-based PCI-Express boards. The High-Level Trigger is a computing cluster dedicated to the online reconstruction and compression of experimental data. It uses a combination of CPU, GPU and FPGA processing. For Run 2, the HLT has replaced all of its previous interface boards with the Common Read-Out Receiver Card (C-RORC) to enable read-out of detectors at high link rates and to extend the pre-processing capabilities of the cluster. The new hardware also comes with an increased link density that reduces the number of boards required. A modular firmware approach allows different processing and transport tasks to be built from the same source tree. A hardware pre-processing core includes cluster finding already in the C-RORC firmware. State of the art interfaces and memory allocation schemes enable a transparent integration of the C-RORC into the existing HLT software infrastructure. Common cluster management and monitoring frameworks are used to also handle C-RORC metrics. The C-RORC is in use in the clusters of ALICE DAQ and HLT since the start of LHC Run 2.

KEYWORDS: Data processing methods; Online farms and online filtering; Pattern recognition, cluster finding, calibration and fitting methods; Data acquisition concepts

¹Corresponding author.



Contents

1	Introduction	1
1.1	ALICE online systems	1
1.2	ALICE High-Level Trigger	3
2	High-Level Trigger C-RORC firmware	3
3	FPGA hardware cluster finding	5
4	Integration into the HLT data transport framework	6
5	C-RORC maintenance and monitoring	7
6	Summary and Run 2 status	7

1 Introduction

1.1 ALICE online systems

ALICE [1] is one of the four major experiments at the Large Hadron Collider at CERN. It is dedicated to the study of the physics of strongly interacting matter in central heavy-ion collisions and is designed to cope with high particle densities in central Pb-Pb collisions. ALICE consists of 19 subdetectors that are read out with a custom optical link protocol called Detector Data Link (DDL) [2]. The DDL connects the front end read-out electronics in the experiment cavern with the computing clusters of Data Acquisition (DAQ) and High-Level Trigger (HLT). The DDL protocol is available in two versions running on a number of different hardware devices and can be operated at link rates between 2 and 6 Gbps [3].

The data from the detectors is received in custom FPGA based Read-Out Receiver Cards (RORC) in the Data Acquisition system, transferred into the host PC and handed over to the DAQ software framework. Additionally, an exact copy of the data is sent towards the High-Level Trigger already inside the DAQ firmware. The High-Level Trigger also uses FPGA based read-out boards to receive detector data from Data Acquisition, process it and send it data back to Data Acquisition via separate RORCs.

The Common Read-Out Receiver Card (C-RORC) [5] was developed as a joint effort of ALICE DAQ and HLT to enable the read-out of detectors at higher link rates and to extend the online pre-processing capabilities of the HLT. Higher link rates are especially anticipated by the Time Projection Chamber (TPC) [6] and the Transition Radiation Detector (TRD) [7]. Additionally, the interfaces of the previous generation of HLT read-out boards are not available anymore in recent server PCs. An overview of the read-out architecture with focus on the Read-Out Receiver Cards is shown in figure 1. The DDL links are drawn as blue arrows. The orange boxes indicate the

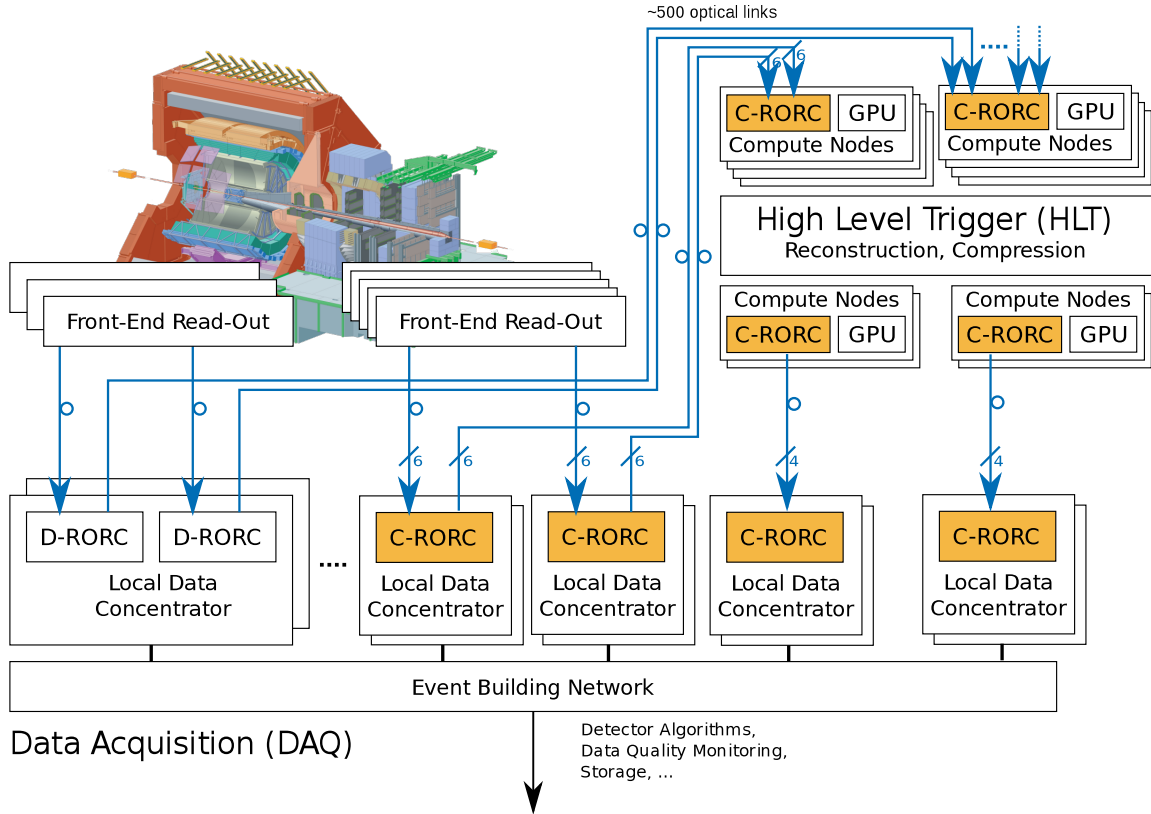


Figure 1. Overview of the ALICE online read-out system with focus on the FPGA based Common Read-Out Receiver Cards (C-RORC) highlighted in orange. ALICE schematics from [4].

installation of C-RORCs in the systems of DAQ and HLT. Several detectors continue to use the first generation of DDL during Run 2 so some of the links are still read out from DAQ using the previous generation of read-out boards, D-RORCs, as shown in the leftmost read-out path. The C-RORC is backward compatible with these boards, which allows DAQ to install a mixture of C-RORCs and D-RORCs and enables HLT to connect to the DAQ D-RORCs.

The C-RORC is a XILINX Virtex[®]-6 based FPGA board with an eight lane PCI-Express Generation 2 interface to the host PC and 12 optical links realized with three QSFP sockets. The optical links can be used up to around 6 Gbps limited by the capabilities of the FPGA. The new hardware provides a factor of six increased link density compared to the previous boards which in return significantly reduces the number of FPGA boards and server nodes required to read out the detectors. Two optional pluggable DDR3 SO-DIMM modules allow to integrate large off-chip memories into the firmware. An on-board flash storage for multiple FPGA bitfiles and a configuration controller provide a fast and reliable FPGA configuration with failsafe fallback image and control path to the board even if the PCI-Express link is down.

The C-RORC boards required for Run 2 were produced together with the ATLAS Experiment [8], which is using the same hardware for the Run 2 upgrade of their Trigger and Data Acquisition ReadOut System (TDAQ ROS) [9]. Around 370 boards were produced and installed into the Run 2 production systems of ALICE and ATLAS in Q3 2014 and are in use since the start of Run 2 in June 2015.

1.2 ALICE High-Level Trigger

The High-Level Trigger (HLT) is responsible for the online reconstruction and compression of detector data. The HLT selects interesting events (triggering) or instructs the Data Acquisition system to discard the detector raw data and to replace it with the results of the online reconstruction. Raw data replacement and HLT data compression significantly reduce the data volume towards permanent storage in the DAQ system and are especially important to handle the data rates in heavy-ion runs. Online reconstruction is required for high-level trigger algorithms and can be used for more elaborate data compression. The reconstructed HLT tracks are used as seeds for the offline reconstruction to save memory and CPU resources there. Online calibration of the TPC is being prepared, working on reconstructed tracks to compensate for temperature and pressure changes inside the detector. This frees up offline compute resources and makes HLT reconstruction more precise. Additionally, reconstruction allows much more detailed quality assurance and online conditions monitoring than what is possible with the raw data.

The HLT is a computing cluster consisting of 180 server nodes, each with a Graphics Processing Unit (GPU) and 74 of the servers with a C-RORC installed. The machines are interconnected with a 56 Gbps Infiniband FDR network for data transfer and an Ethernet network for management tasks. The reconstruction is realized with the help of FPGA-based cluster finding already in the C-RORC firmware and tracking on GPU accelerators [10] in addition to CPU based data processing and compression. The experimental data is received via C-RORCs from the Data Acquisition system, processed inside the HLT and sent back to DAQ via C-RORCs. In this sense, the HLT appears like another detector to DAQ.

2 High-Level Trigger C-RORC firmware

The C-RORC has two functionalities in the HLT cluster: it is used as a receiving entity for experimental data coming from the Data Acquisition system (HLT-IN) and as a sending entity to provide the processed results back to DAQ (HLT-OUT). Data transfer from and to the C-RORC is realized with Direct Memory Access (DMA) to maximize the data throughput of the interface while saving CPU resources. Firmware support for DMA data transfer via PCI-Express is required in both directions, DMA-to-host and DMA-to-device, to serve both use cases. For the same reason two different modules are available as interface to DDL via the optical links: in the HLT-IN case the Destination Interface Unit (DIU) is implemented to receive data and in the HLT-OUT case the Source Interface Unit (SIU) is integrated to send data. Unfortunately, not all HLT-IN units run at the same link speed because different detectors use different versions of the DDL protocol. An optional hardware pre-processing core is integrated into the read-out path on a per-detector level. The firmware is designed in a modular approach so that the interfaces between the different building blocks are well defined and the blocks are easily interchangeable. This also allows the integration of the optional FastClusterFinder core, an online pre-processing component to perform hardware cluster finding on TPC data. The cluster finding is described in detail in section 3. The data direction, the different link speeds and also performance reasons for the hardware pre-processing require a set of different firmware images to be deployed in the HLT cluster for different detectors. Not all features make sense to be integrated into the same firmware image, but are enabled or

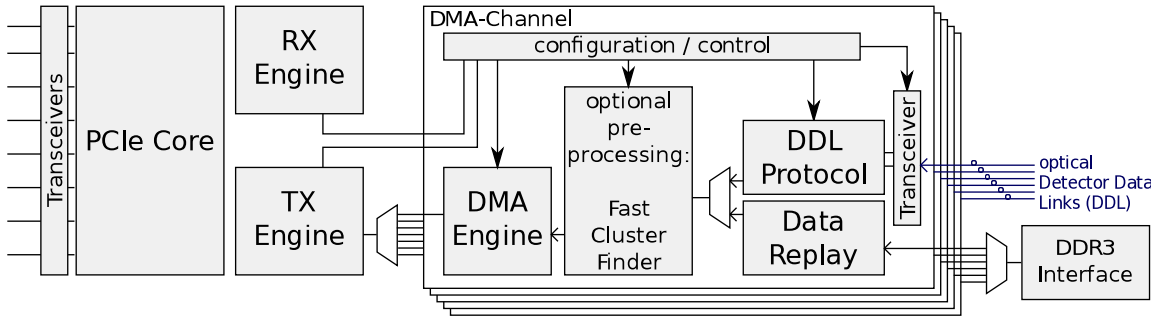


Figure 2. Simplified overview of the firmware building blocks for the HLT-IN case.

disabled with compile time switches from the same source tree. A simplified overview of the HLT-IN firmware building blocks is shown in figure 2.

The PCI-Express interface in the firmware is implemented with a custom DMA engine. With up to 12 data channels (one per optical link) and the requirement of independent operation of all channels and the possibility of attaching them to the existing HLT software infrastructure, none of the commercially available DMA cores could provide the required functionality. The DMA engine supports scatter-gather list DMA and operates on ring buffers. The scatter-gather list support of the DMA engine allows the engine to be used with almost any memory allocation scheme in the host machine. Two ring buffers are used per data channel, one as event buffer and one as report buffer. In the DMA-to-host case, experimental data is continuously written into the event buffer. After each event an entry is written into the report buffer containing offset, size and status flags of the corresponding event data block in the event buffer. The software notifies the firmware of processed events in the event buffers and thus makes the buffer space available for DMA again. In the case of DMA-to-device, the software pushes a list of descriptors into the DMA engine. The engine fetches the data according to these descriptors and writes a report buffer entry when done.

The DDL protocol block is an IP core provided by the Data Acquisition group. It connects directly to the FPGA transceivers and provides a simple bus interface with flow control. Two DDR3 memory controllers are implemented in the HLT-IN firmware images to allow the replay of generated or previously recorded detector data into the system at its very first entry point into the HLT. The HLT output firmware can be configured in a way that it discards event data right before it would be sent back to DAQ. This allows to do standalone tests of the full HLT cluster with configurable data, event rates or sizes. The replay system proved to be a valuable tool to test the limits of different HLT configurations during the initial commissioning of the HLT system, but also during the integration of new or updated processing components.

In order to validate the hardware pre-processing core the TPC read-out firmware is equipped with a secondary read-out path: the TPC raw data is additionally available via separate DMA channels and can optionally be read out to compare the hardware pre-processing results with a software reference operating on raw data. The firmware allows reading out only a configurable selection of the raw data in order to enable a validation also in a high load scenario or as a continuous parallel service.

Several error counters and status flags throughout the whole data path allow to gather statistics at several stages and help to detect error conditions. Status and control signals of the transceivers, the protocol and all firmware components are available from an on-chip slow control bus via software.

Status information from the QSFP modules is available via an I2C master core also connected to the slow control bus and allows diagnosis of the fiber connection network.

3 FPGA hardware cluster finding

The FastClusterFinder is an optional pre-processing component in the HLT C-RORC firmware for the read-out of the Time Projection Chamber (TPC) data. The FastClusterFinder was already developed for and used in Run 1 and is still an essential component of the system that significantly reduced the amount of CPU computing power required in the HLT [11]. It is operating on TPC raw data and is performing cluster finding in time direction and one space direction along a row of read-out pads. Clusters of neighboring pads are merged and overlapping clusters are separated. The correlation of data from adjacent pads and sample times allows clusters to be localized below the pad dimension and time stamp resolution. The FastClusterFinder is designed in a way that it can handle the full bandwidth of the DDL link so it does not throttle the event rate and introduces only a marginal additional latency in the read-out path.

The algorithm consists of four major processing steps. In a first stage the detector protocol is decoded. The hardware address of the input data block allows a look-up of the physical location of the data origin as row and pad. The look-up additionally provides a per-pad gain correction factor which is applied on the input data stream. A sliding window peak finding algorithm in a second stage analyzes the time based sequence of data samples and cuts out a region of interest (ROI) around the peaks. The ROI is characterized with weighted sums in time and space direction, as well as the peak value and the accumulated charge. The resulting clusters are merged with clusters from neighboring pads in a third step. The last stage is a single floating point division with data formatting and handover to the DMA engine interface. In all except the final step the data processing operations are based on integer or fixed point arithmetic, which makes it well suited for the C-RORC FPGA.

The Readout Control Unit (RCU), the device responsible to ship data from the TPC front-end electronics to the Data Acquisition system, is being upgraded to a new revision, the RCU2 [12]. The RCU2 is doubling the read-out link speed to 4.25 Gbps which also influences the FastClusterFinder in the HLT C-RORC firmware: the core has to run with twice the clock rate to maintain its ability to handle the full DDL bandwidth. Some data reordering tasks are moved into the RCU2 firmware which reduces the resource usage of the HLT FastClusterFinder core. In order to achieve the required clock rates some of the processing components were rewritten and register stages were integrated into existing parts where possible.

The FastClusterFinder core for the RCU2 is now running at 318.75 MHz in the C-RORC FPGA and can handle the full input bandwidth of the DDL2 link at 4.25 Gbps. Six FastClusterFinder instances are implemented in each of the 36 TPC C-RORCs in the HLT. The TPC end-plates are segmented into 2×18 trapezoidal sectors and the DDL links from each sector are read out from the same C-RORC. This allows local processing of TPC data already on the input node without having to move data around. The FastClusterFinder enables cluster finding at the full data rate requiring zero additional CPU resources.

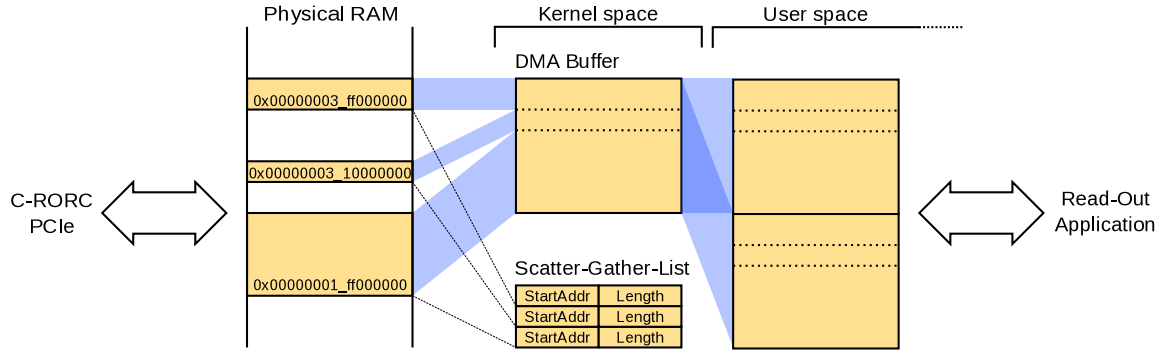


Figure 3. Mapping of physically scattered DMA memory into a virtually continuous user space buffer.

4 Integration into the HLT data transport framework

The HLT data processing is realized with an object oriented data transport framework based on the publisher-subscriber model [13]. A chain of framework components allows detector data to be processed in a pipelined way. Gatherer components collect event fragments from the different input sources, bridges provide a transparent connection between different processing nodes and scatterer components allow processing tasks to be distributed among parallel processes or nodes. A framework interface to AliRoot, the ALICE Offline framework for simulation, reconstruction and analysis [14], allows AliRoot tasks to be instantiated as the actual processing entities. This enables the same code to be used online in the HLT and in the Offline environment. The required processing steps for each run depend on the list of detectors being used and the read-out configuration. An automated HLT tool prepares this chain, the underlying publisher-subscriber data graph and the distribution of these processes across the worker nodes at the start of each run from a set of configuration files and the information received from the Experiment Control System (ECS).

Data exchange between C-RORC and software framework is implemented with the Portable Driver Architecture (PDA) [15] and device access is realized fully from user space. DMA memory for the ring buffers is allocated with the help of a minimal kernel module using standard Linux allocation methods. The allocated memory is not necessarily physically continuous but may be scattered into several chunks. The physically scattered memory fragments are mapped twice and consecutively into a virtually continuous buffer in user space so the data processing software does not even notice the scattering or the ring buffer structure. This allows a transparent handling of a ring buffer wrap-around: events always appear in a single, virtually continuous memory region. An illustration of the buffer mapping is shown in figure 3. The ring buffers for C-RORC read-out are allocated once at the start of run so they do not increase memory fragmentation over acquisition time.

The data source components in the publisher-subscriber chain initialize the C-RORC DMA engine, allocate buffers and poll the report buffer for new events. Each DDL link is handled by a separate read-out process, so up to 12 processes are accessing the same C-RORC. Whenever an event is received, its descriptor is published into the processing chain. Follow-up subscriber components directly access the C-RORC DMA buffer based on the information in the event descriptor, process the data and put the result into their own output buffers. Whenever an event is processed completely, the corresponding buffer memory is made available for DMA again. The data sink components connect to the HLT-OUT C-RORCs in the same way: the processed data is placed into DMA buffers

by the last publisher component in the chain. The data output process sends an event descriptor to the C-RORC. The DMA engine fetches the event from the DMA buffer and writes a report buffer entry when done. The firmware takes care about transferring the event to the Data Acquisition via DDL. In parallel to data handling, the data source and sink components also monitor the DDL link and the corresponding DMA engine: link errors or stall conditions are detected and reported immediately. Per-event conditions are handed over to software synchronously with the event stream using predefined flags in the report buffer entry.

5 C-RORC maintenance and monitoring

A total of 74 C-RORCs are installed in the HLT. Depending on data direction and connected DDLs, they have to have different firmware images loaded. In order to keep track on the firmware of each board an automated firmware management system is deployed. The software installation, configuration and maintenance of the HLT machines is done with Puppet [16] and Foreman [17]. The existing system is configured to also watch and adjust the C-RORC firmware. Each machine with a C-RORC installed is assigned to a host group defining the C-RORC function. A central configuration file defines firmware revision and type for each host group. The currently active firmware revision and type is available via *Facter*, a Puppet tool to gather facts about the client system. A Puppet run on the client compares the current firmware with the target version of the host group. On mismatch, the correct FPGA bitfile is fetched and written into the C-RORC on-board flash via PCI-Express. A reconfiguration of the FPGA is triggered via the on-board configuration controller. After a rescan of the PCI-Express bus the C-RORC is available again with the new firmware. A firmware update of the whole cluster is as easy as uploading the new firmware image to the server and adjusting one configuration file. The next Puppet run takes care of everything else without even rebooting the machine.

In addition to the information about the firmware image itself, each board provides status and health readings from on-board sensors. These per-board readings are queried by Zabbix [18], which is used as monitoring solution for the host metrics of all HLT machines. This allows to monitor FPGA temperature and PCI-Express link state of all boards in the cluster and makes it possible to alert the administrator in case of error conditions. Monitoring of per-link conditions is integrated into the data transport framework as described in section 4.

6 Summary and Run 2 status

The Common Read-Out Receiver Card (C-RORC) is a common hardware platform used in the production systems of ALICE DAQ, ALICE HLT and ATLAS TDAQ ROS. The modular structure of the HLT C-RORC firmware allows several firmware variants to be created from the same source tree, including variants with online data pre-processing already in the FPGA. The FastClusterFinder is an online pre-processing component in the HLT C-RORC firmware that performs cluster finding on TPC raw data. It complements the CPU and GPU processing power of the HLT cluster by an FPGA contribution. The C-RORC is integrated into the existing HLT data transport framework with data source or sink components using a user space device driver. Standard open source cluster management tools are used to assure correct firmware images and to monitor all boards in the cluster.

The C-RORC is in use since the start of LHC Run 2 and all ALICE data from and to HLT as well as all data from the TPC and the TRD is handled by C-RORCs. The HLT in Run 2 is connected

to 437 DDLs from DAQ of which 216 are coming from TPC. The DDL link rates vary between 4.0 Gbps for TRD, 4.25 Gbps for TPC RCU2 and 2.125 Gbps for everything else. The data path from HLT to DAQ is done with 28 DDLs running at 5.3125 Gbps. The raw TPC data is discarded by the Data Acquisition system and only the compressed TPC data from HLT containing the clusters from the FastClusterFinder is sent to permanent storage. This reduces the TPC data volume by a factor of around three to four.

Acknowledgments

Supported by the German Federal Ministry of Education and Research BMBF 05P15RFCA1.

References

- [1] ALICE collaboration, *The ALICE experiment at the CERN LHC*, 2008 *JINST* **3** S08002.
- [2] ALICE collaboration, H. de Groot, *ALICE trigger data-acquisition high-level trigger and control system: technical design report*, CERN-LHCC-2003-062, CERN, Geneva Switzerland (2003).
- [3] F. Carena et al., *DDL, the ALICE data transmission protocol and its evolution from 2 to 6 Gb/s*, 2015 *JINST* **10** C04008.
- [4] J. Thaeder, *3D ALICE schematic*, August 2012.
- [5] H. Engel and U. Kebschull, *Common read-out receiver card for ALICE Run 2*, 2013 *JINST* **8** C12016.
- [6] J. Alme et al., *The ALICE TPC, a large 3-dimensional tracking device with fast readout for ultra-high multiplicity events*, *Nucl. Instrum. Meth. A* **622** (2010) 316 [[arXiv:1001.1950](#)].
- [7] ALICE collaboration, *ALICE transition-radiation detector: technical design report*, CERN-LHCC-2001-021, CERN, Geneva Switzerland (2001).
- [8] ATLAS collaboration, *The ATLAS experiment at the CERN Large Hadron Collider*, 2008 *JINST* **3** S08003.
- [9] W. Vandelli et al., *Evolution of the ReadOut System of the ATLAS experiment*, *PoS(TIPP2014)* 205.
- [10] ALICE collaboration, *ALICE HLT high speed tracking on GPU*, *IEEE Trans. Nucl. Sci.* **58** (2011) 1845.
- [11] T. Alt, *High-speed algorithms for event reconstruction in FPGAs*, presentation at *IEEE Real-Time*, (2010).
- [12] J. Alme et al., *RCU2 — the ALICE TPC readout electronics consolidation for Run 2*, 2013 *JINST* **8** C12032.
- [13] T.M. Steinbeck, V. Lindenstruth and M.W. Schulz, *An object-oriented network-transparent data transportation framework*, *IEEE Trans. Nucl. Sci.* **49** (2002) 455.
- [14] *ALICE off-line project webpage*, <http://aliweb.cern.ch/Offline/>, accessed October 2015.
- [15] D. Eschweiler and V. Lindenstruth, *The portable driver architecture*, in *Proceedings of the 16th Real-Time Linux Workshop*, Open Source Automation Development Lab (OSADL), Germany October 2014.
- [16] *Puppet, Open Source Configuration Management webpage*, <https://puppetlabs.com>, accessed October 2015.
- [17] *Foreman webpage*, <http://theforeman.org>, accessed October 2015.
- [18] *Zabbix, Enterprise Open Source Monitoring for Networks and Applications webpage*, <http://www.zabbix.com>, accessed October 2015.