ATL-DAQ-PROC-2015-061
21 November 2015

An evaluation of GPUs for use in an upgraded ATLAS High Level Trigger

A. T. Delgado, P. Conde Muíño, J. Augusto Soares, R. Gonçalo, J. Baines, T. Bold, D. Emeliyanov, S. Kama, M. Bauce, A. Messina, M. Negrini, A. Sidoti, L. Rinaldi, S. Tupputi, Z. D. Greenwood, A. Elliott, S. Laosooksathit
On behalf of the ATLAS Collaboration

***Abstract*—ATLAS is a general purpose particle physics experiment located on the LHC collider at CERN. The ATLAS Trigger system consists of two levels, the first level (L1) implemented in hardware and the High Level Trigger (HLT) implemented in software running on a computing cluster of commodity CPUs. The HLT reduces the trigger rate from the 100 kHz L1 accept rate to 1 kHz for recording, requiring an average per-event processing time of ∼300 ms for this task. The HLT selection is based on reconstructing tracks in the Inner Detector and Muon Spectrometer and clusters of energy deposited in the calorimeters (electromagnetic and hadronic). Performing this reconstruction within the available HLT computing cluster resources presents a significant challenge. Future HLT upgrades will result in higher detector occupancies and, consequently, will harden the reconstruction constraints. General purpose Graphics Processor Units (GPGPU) are being evaluated for possible future inclusion in an upgraded HLT computing cluster. We report on a demonstrator that has been developed consisting of GPGPU implementations of the calorimeters clustering and Inner Detector and Muon tracking algorithms integrated within the HLT software framework. We give a brief overview of the algorithm implementation and present preliminary measurements comparing the performance of the GPGPU algorithms with the current CPU versions.**

I. Introduction

THE CERN Large Hadron Collider (LHC) [1] was build to explore the fundamental constituents of nature and the forces between them, at unprecedented energies. It is a circular accelerator with a perimeter of 27 km where two proton beams cross 40 million times per second. Each beam crossing is usually referred to as an event. The second data taking period, Run 2, started this year and will last until 2018. During Run 2 the LHC is operating at a centre-of-mass energy of 13 TeV to 14 TeV, almost two times higher than in Run 1, and an average of ∼27 pp collisions per bunch crossing (known as pile-up).

The third data taking period, known as Run 3, is scheduled to start in 2019, after a two years shutdown for upgrade of the accelerator and the detectors, as shown on the activity schedule Table I. In Run 3 the LHC will work with a two times higher luminosity than in Run 2, yielding in an increased mean number of pp collisions per event and resulting in higher number of particles hitting the detectors per event. As this results in events that are more complex, the trigger reconstruction software will demand more computing power. Therefore it will be essential to reduce the processing time of the reconstruction algorithms, to keep them within the time constraints of the online system while maintain the same physics performance. The General Purpose Graphical Processing Units (GPGPUs) can provide better computing performance to power ratio than Central Processing Units (CPUs), and are thus good candidates to maximize the computing cluster power, as the computing cluster is limited by the rack-space and cooling power.

A. The ATLAS experiment

The ATLAS detector is one of the two LHC multi purpose experiments [4].

It is a cylindrical shape detector with 46 m length and 25 m height. The detection elements are arranged in layers around the beam pipe. The inner part is the Inner Detector tracker (ID), immersed in a magnetic field generated by a superconductor solenoid. The ID allows the detection of charged particles trajectories and is made of three different technologies: pixel detectors, in the inner most layers; Semiconductor Tracker (SCT) on the middle layers and Transition Radiation Tracker (TRT) in the outer most layers.

The ID is surrounded by the calorimeter systems, composed by the electromagnetic calorimeter, based on Liquid Argon (LAr) technology, and the hadronic calorimeters, made of LAr and scintillator tile technologies.

The muon spectrometer is the outermost sub-detector, immersed in a second magnetic field generated by superconducting toroids.

B. The ATLAS Trigger and Data Acquisition systems

In total the ATLAS detector has around 10^8 electronic channels, resulting in events with an average size of 1.7 MB.

Manuscript received November 23, 2015. A. T. Delgado acknowledges the support of the IDPASC doctorate network and Fundação para a Ciência e Tecnologia, Portugal, through the grant SFRH/BD/51792/2011 and the FEDER/COMPETE-QREN.

A. T. Delgado, P. Conde Muíño, J. Augusto Soares and R. Gonçalo are with the Laboratório de Instrumentação e Física Experimental de Partículas (LIP), Lisbon, Portugal

J. Augusto Soares is also with INESC-ID and Faculdade de Ciências, Universidade de Lisboa, Lisboa, Portugal

J. Baines and D. Emeliyanov are with STFC and Rutherford Appleton Lab, GB

T. Bold is with AGH Univ. of Science and Technology, Krakow, Poland

S. Kama is with Southern Methodist University, Dallas, TX, USA

M. Bauce and A. Messina are with Sapienza Università di Roma and INFN, Italy

M. Negrini and A. Sidoti are with Istituto Nazionale di Fisica (INFN), Italy

L. Rinaldi is with Università di Bologna and INFN, Italy

S. Tupputi is with INFN-CNAF, Italy

Z. D. Greenwood, A. Elliott and S. Laosooksathit are with Louisiana Tech University, Ruston LA, USA

(e-mail: tavares@cern.ch).

Table I: Table with LHC upgrade phases and nominal parameters [2][3].

	Run 1		LS1 - Phase-0	Run 2	LS2 - Phase-I	Run 3	LS3 - Phase-II	Run 4
	2011	2012		2015-17		2019-21		2023-
Center of mass Energy $\sqrt{s}$ (TeV)	7	8		13-14		14		14
Luminosity ($\mathrm{cm}^{-2}\mathrm{s}^{-1}$)	8×10^{33}			1×10^{34}		2×10^{34}		5×10^{34}
Bunch spacing (ns)	50			25		25		25
Number of interaction/event, $< \mu >$	10	20		~ 27		$\sim 55 - 80$		~ 140
Total Integrated luminosity (fb^{-1})	25			~ 100		~ 300		~ 3000

As they are read for every proton bunch crossing, every 25 ns, the total data volume is closer to ~ 64 TB/s, unfeasibly large to be recorded or processed with full precision in real-time with LHC. Furthermore, only a small fraction of these events contain a significant physics interest. The selection of which events should be kept for later analysis is made by the ATLAS trigger system [5], which probes the event data against a menu of desirable physical characteristics, typically the presence of a certain physical object (e.g. a highly energetic electron), that leaves a distinctive pattern in the detector. This menu contains a few thousands of different possibilities. The event selection is hampered by background patterns which mimic the desired objects. Each event has to be processed in real-time and the data volume reduced by a factor of 10^4 [6], [7] for offline storage.

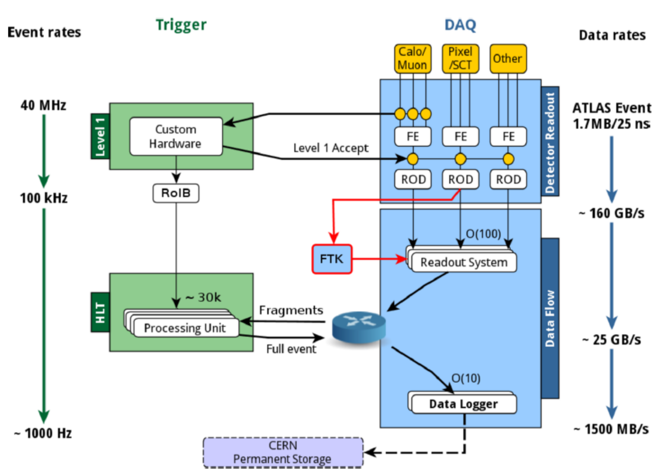

Figure 1: Schematic diagram of the ATLAS Trigger system showing the input and output event rates and the expected data rates at the different trigger levels.

The trigger system is divided in two levels, as shown in Figure 1, the Level-1 (L1) and the High Level Trigger (HLT). L1 is based on custom hardware and is located near the detector. It uses a simple and fast reconstruction, over a coarse granularity readout of the calorimeter and muon spectrometer, to find *Regions of Interest (RoI)*, where high transverse energy (E_{T}) objects like electrons (e), photons (γ), muons (μ) or jets are found. L1 takes a decision within a latency of 2.5 µs and selects at most one out of 400 events, thus reducing the 40 MHz crossing rate to a maximum of 100 kHz for input to the HLT. The detector data of these events is then read from the front-end electronics and stored in buffers in the Readout System (ROS), waiting for HLT requests and decisions.

The HLT is software based, implemented mainly in C/C++, and runs on a CPU computing cluster, under a a component framework named Athena [8]. The system was designed for multi-process event processing, running one HLT Processing Unit (HLTPU) in each CPU core. HLT executes chains of reconstruction *(feature extraction)* algorithms followed by hypothesis testing (hypothesis) algorithms. Chains are seeded by the L1 RoIs. Each algorithm in a chain runs over the output of the previous one. If the same algorithm is scheduled for execution in different chains over the same data, then the first execution output is cached and used on the remaining chains. In this way repeated calculations are avoided. The hypothesis algorithm's job is to compare the features produced against some configured hypothesis and accept or reject the events. The system was designed for early event rejection. It also allows chains than run over partial data, requiring typically 2 % to 6 % of the full event data, to be processed in order to reject the events. For the rejected events the data is flushed from ROS system and the information of this collision is not retained. This is the case for 99% of events. HLT has an average event processing time budget of 300 ms. In this time it selects at most one out of 100 events thus reducing the event rate for permanent storage to about 1 kHz, translating to a data rate of about 1.5 GB/s.

II. Trigger on GPUs

ATLAS is investigating the execution of trigger reconstruction algorithms on GPUs, as a potential solution to meet Run 3 challenges of event reconstruction within the trigger computing cluster constraints, mainly the available rack-size and the thermal extraction capability. As GPUs are autonomous high performance computing devices, data has to be transferred between the CPU host and the GPU device, imposing a data transfer overhead. The performance achieved, along with the code porting effort required, in terms of manpower, will contribute to the architecture choices of the trigger computing cluster upgrade. To aid the architectural choice a demonstrator implementation of several algorithms, most time consuming or demonstrating the worst luminosity scalability, was launched. It is supposed to conclude with the indication of the cost/benefit estimates for various hardware compositions of the future HLT computing cluster.

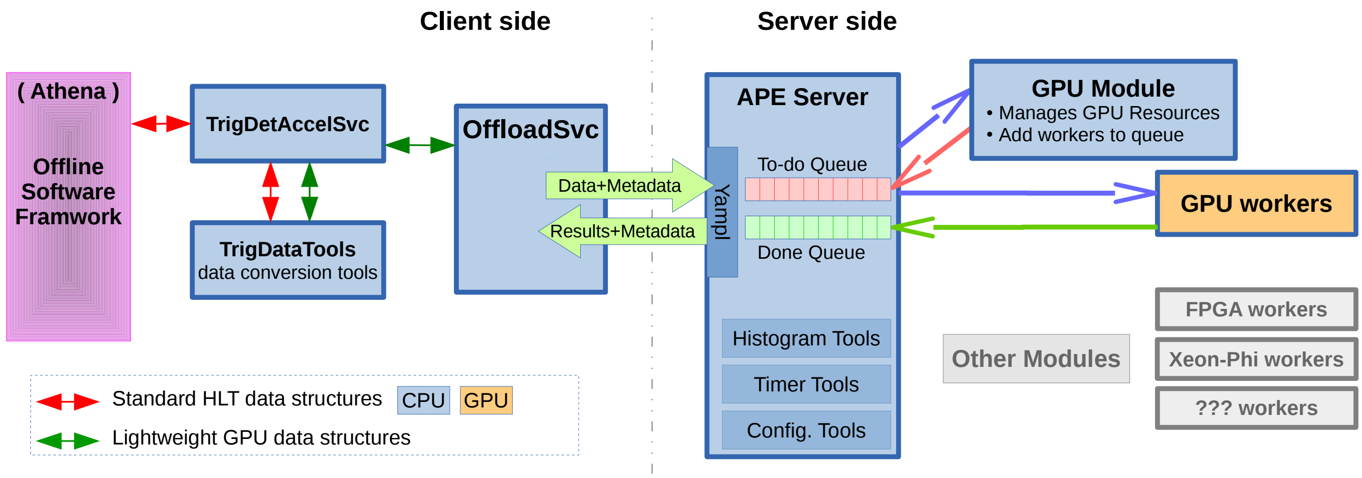

Figure 2: Trigger on GPU framework schematic. ATLAS framework (Athena) clients request offloading to the *TrigDetAccelSvc*. *TrigDetAccelSvc* uses *TrigDataTools* for data conversions between the client and server structure. Data and meta-data is then sent through the *offloading* service. The server reads the meta-data and hands the request to the proper *module*. The module appends the data to a unique data space and gives it to a new *worker*. The *worker* is appended to a to-do queue that is managed by the server. After execution, the *worker* goes to a done queue and the server hands the result back to the client.

The trigger GPU project demonstrator comprises ID, calorimeter, muon and jet reconstruction algorithms. From an initial analysis, taking into account expected speedup due to Amdahl law, the following set of algorithms was selected to be ported to a GPU architecture:

- Inner Detector data preparation, seed making and track-following algorithms.
- Calorimeter cell clustering algorithm.
- Muon tracking algorithms, based on Hough transform.
- Jet finding *Anti-*k_t algorithm [9], [10].

Nvidia cards with the CUDA [11] framework were chosen for the demonstrator, based on the hardware quality, maturity of the technology and framework, the good framework support and lower porting effort.

A. *GPU Acceleration framework*

The demonstrator implements a client-server architecture, based on the Accelerator Process Extension (APE) frame-work [12], to offload and process HLT data, as shown in Figure 2. This allows a reduction of the resources needed as the services of one server are available to many clients as well as separation of concerns, where the APE server is only responsible for computing on GPU while HLT only for processing on CPU.

1) Client Side: The client side is implemented in the HLT. For the reconstruction of data from each ATLAS sub-detector an algorithm requesting the GPU-accelatared processing is developed. These algorithms extract the input data from the detector, request GPU-acceleration process of the data through an acceleration service (*TrigDetAccelSvc*), and inject the result back to the HLT.

The *TrigDetAccelSvc* uses *TrigDataTools* to convert back and forth the sophisticated Athena data structures to ones suitable for GPU implementation. Each sub-detector uses its own Acceleration service and data export tool.

The data to be processed are then sent to the GPU through the offloading service (*OffloadSvc*) to the APE server. After processing, the offloading service sends the result back to the detector specific acceleration service, which in turn sends it back to the HLT algorithm, after converting back to the Athena data structures.

2) Server side: APE implements a plug-in mechanism and is composed by the manager, the modules and the workers. The manager deals with the offloading requests from the HLT processes and demands the workers execution.

The module manages resources and the processing requests by creating work items. When initialized, the module creates a set of data contexts that are stored in a context queue, managed by the module. Each data context is a unique space, containing all the necessary memory blocks and configurations needed to process the input data. After initialized, the module stays as a service waiting for acceleration request. Upon new request it picks the next free context data and pairs it with the received data to crate the work item.

Work items, being GPU ported versions of Athena algo-rithms, perform the computations requested by the clients. They are usually composed by CUDA kernels and each work runs on its own independent stream. When the worker finishes it moves itself to the work done queue. The worker returns the data context back to the context queue after it receives the request for results and before its own destruction.

It is the module that is responsible for executing the work items by placing them in to-do queue. It is also responsible for sending the result of the computation back to the HLT. Modules and workers are specific of each detector.

By using the accelerator abstraction and the modular struc-ture, APE can exploit any kind of computing resource, such as GPUs, FPGAs or Xeon-Phi, as long as modules and workers are provided for such technologies.

B. *Trigger modules implementation*

The GPU demonstrator project started with an evaluation of the HLT algorithms. It highlights the most interesting algorithms to port to GPUs by each sub-system: the inner detector tracking, the calorimeter cell clustering, the muon tracking and the jet finding algorithms.

The ID tracking is the most time consuming part of the system, followed by the calorimeter clustering, muon tracking and jet maker.

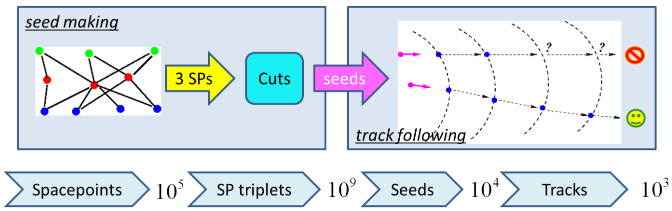

Figure 3: Inner detector seed making and track-following algorithms schematics. Compatible clusters in inner layer are paired to form seeds. Seeds are then paired with outer layer clusters to form triplets. Triplets are then followed through the full detector to form track candidates. A decision algorithm then selects the final tracks.

1) Inner detector: The inner detector reconstruction starts with the decoding of the raw data [13]. It then clusters neighbouring activated sensors (hits), using a cellular automaton algorithm [14]. Compatible clusters in the two inner most layers are paired to form objects known as seeds, left side of Figure 3. Seeds are paired with the clusters in the outer layer to form triplets of space-points. Then track-following algorithm starts and extrapolates the triples of space-points (SPs) through the full detector, to form track candidates, as shown in right side of Figure 3. After the track-following a large number of tracks candidates is formed due to the significant detector occupancy. Therefore, at the final stage, an hypothesis algorithm selected best tracks from all candidates.

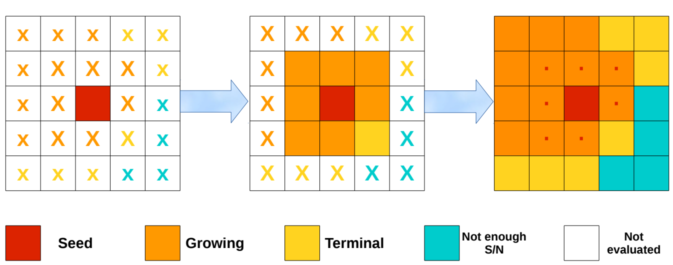

Figure 4: Calorimeter cell clustering algorithm schematic. The algorithm starts by classifying the cells in 3 groups according to S/N value: SEEDS > GROWING > TERMINAL. SEED cells initiate clusters, with a defined unique cluster tag. SEEDS and GROWING cells tag is passed to their neighboring cells, if they are tagged with higher S/N ratio. The algorithm stops when no cell tag is modified.

2) Calorimeter: The ATLAS Topological Cluster (TC) [15] algorithm joins the calorimeter detection units, known as cells, to form three-dimensional energy deposition clusters whilst suppressing the noise contribution. The noise suppression is achieved by making the cell clustering dependent on the neighbouring cells energy significance (S/N), the latter given by the ratio of the energy deposition with respect to the average electronics and pile-up noise in that cell. However, this kind of clustering requires more computation than what is required by simpler algorithms. Thus, the Topological Cluster algorithm is only used in the latest stage of the original ATLAS trigger and in the offline reprocessing of the accepted events.

The calorimeter cell clustering classifies the detector cells in three groups, according to the cells signal-to-noise ratio. Cells with higher ratio, usually above four, are called SEED cells. Beside those, cells are classified as GROWING, usually if the energy is two times higher than the noise, or TERMINAL, which are remaining cells with absolute energy above zero. Each SEED cell starts a cluster formation, as shown in Figure 4. The clusters grow by iteratively including the neighbours of SEED or GROWING cells. TERMINAL cells are added to clusters to form the outer layer. Two different clusters are merged if they share a SEED or GROWING cell.

The GPU implementation of this algorithm, the Topological Automaton Clustering (TAC), is a parallel oriented implementation of the TC algorithm. It has to keep the TC properties and produce the same results. The algorithm starts with the cells classification. At this stage, the work space is simplified into pairs of cell and a neighbour. This abstraction assures an evenly distribution of workload across all GPU cores. Then the SEED cells are ordered so that each cluster will have a unique tag, the position of the SEED cell in the ordered list. The clustering starts after that. The clustering is based on a cellular automaton algorithm. Each thread evaluates a pair of cells and makes the cluster tag propagate according with the rules specified before. This process continues till the iteration cells do not change their tag. The set of cells in each cluster is the result shipped to the HLT client.

III. RESULTS

Preliminary results of the trigger GPU demonstrator are presented below. The gross figure of merit for the demonstrator is the throughput expressed as the number of events processed per second using the specific combination of hardware and software. The benefits of faster execution on the PC with GPU have to be compared against performance of same power or same cost machine with only CPUs. Fair comparisons have to assume comparable performance of the algorithms of which an example is presented below. In addition to the throughput the scalability of the implementations has to be assessed. This is achieved by measuring the algorithm processing time as a function of input data size.

A. *Inner detector*

The per-event execution time of the track seeding algorithm, as a function of the number of space points, is shown in Figure 5. The plot compares the standard CPU serial implementation against the parallel version ported to GPU. This test was performed on a machine with an Intel^{TM} Xeon E5-2695@2.3GHz and a Nvidia^{TM} Tesla K80. The data set used

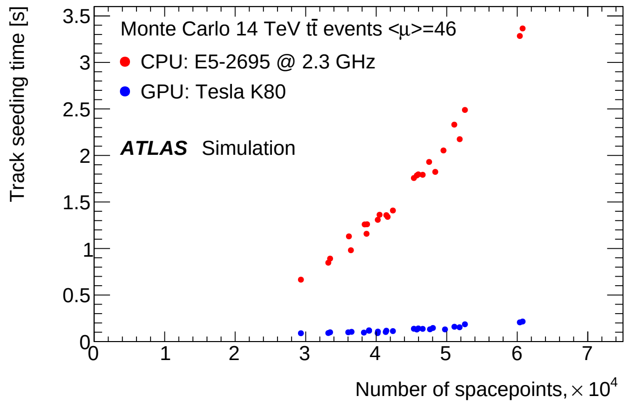

Figure 5: Timing of the Inner Detector (ID) track seeding algorithm for the full detector. The red dots represent the standard HLT ID algorithm, running on a single CPU core, the blue dots show the algorithm ported to GPU. The timing is shown as a function of input data size, a number of space points from which the seeds are formed [16].

consisted of Monte Carlo simulated $t\bar{t}$ events, for a scenario of 14 TeV collisions and a mean value of 46 proton interactions per bunch crossing, representing a typical scenario for Run 3.

This plot shows that the GPU implementation of the ID tracking algorithm is already up to 17 times faster than the CPU version and its performance scales linearly in the region of interest.

B. *Calorimeter cell clustering*

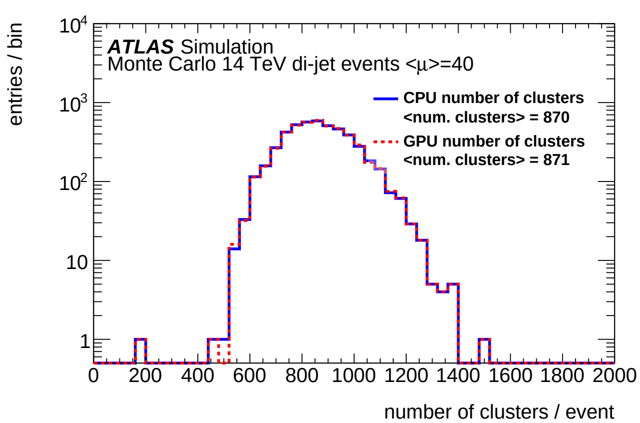

Figure 6: Number of calorimeter clusters reconstructed using the standard CPU cell clustering algorithms and the algorithm ported to GPU. The blue line represents the CPU standard algorithm. The red dashed line represents the GPU cell clustering based on a Cellular Automaton algorithm [17].

The number of clusters reconstructed per-event is shown in Figure 6. The histogram compares the standard CPU serial implementation against the ported GPU parallel version of the algorithm, for a data sample of QCD di-jet events, simulated using Monte Carlo simulated events for a scenario of 14 TeV collisions, with leading-jet transverse momentum above 20 GeV and a fixed number of 40 simultaneous interactions per bunch-crossing. The results presented are obtained after the complete Trigger Clustering execution. The histogram shows that both distributions are in very good agreement, with the mean number of cluster agreeing within 0.1% for both implementations.

IV. Conclusions

The LHC instantaneous luminosity for Run 3 will double compared to Run 2. For the ATLAS trigger system, higher luminosity will require more computation power to exploit the full potential of the LHC. GPUs are massive parallel architectures with high computing throughput and efficiency, in terms of operations per watt, making them interesting solutions for the trigger computing cluster upgrade.

A GPU trigger demonstrator prototype is being implemented to assess the potential of such a system. For it, a server-client system was chosen to handle the trigger requests for GPU data processing. The demonstrator covers a set of algorithms from all sub-detectors. The ID tracking algorithm has already demonstrated a very significant speed-up of 17 times. For the calorimeter cell clustering, the results showed an almost perfect agreement between the CPU and the GPU versions of the algorithm. Muon and jet algorithms are in the final implementation stage. The final stage of integration is ready and further tests are going to be performed to include detailed measurements of the throughput per unit cost for various architectural choices.

References

[1] L. Evans and P. Bryant, *LHC Machine*, JINST **3** (2008) S08001.

[2] ATLAS Collaboration, *Letter of Intent for the Phase-I Upgrade of the ATLAS Experiment*, Tech. Rep. CERN-LHCC-2011-012. LHCC-I-020, CERN, Geneva, Nov, 2011. https://cds.cern.ch/record/1402470.

[3] ATLAS Collaboration, *Letter of Intent for the Phase-II Upgrade of the ATLAS Experiment*, Tech. Rep. CERN-LHCC-2012-022. LHCC-I-023, CERN, Geneva, Dec, 2012. https://cds.cern.ch/record/1502664.

[4] ATLAS Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, JINST **3** (2008) S08003.

[5] ATLAS Collaboration, *Performance of the ATLAS Trigger System in 2010*, Eur. Phys. J. C **72** no. 1, (2012). http://dx.doi.org/10.1140/epjc/s10052-011-1849-1.

[6] ATLAS Collaboration, *Technical Design Report for the Phase-I Upgrade of the ATLAS TDAQ System*, Tech. Rep. CERN-LHCC-2013-018. ATLAS-TDR-023, CERN, Geneva, Sep, 2013.

[7] ATLAS Collaboration, *Performance assumptions for an upgraded ATLAS detector at a High-Luminosity LHC*, Tech. Rep. ATL-PHYS-PUB-2013-004, CERN, Geneva, Mar, 2013. http://cds.cern.ch/record/1527529.

[8] ATLAS Collaboration, *ATLAS computing: Technical design report*,.

[9] M. Cacciari, G. P. Salam, and G. Soyez, *The anti-k_t jet clustering algorithm*, Journal of High Energy Physics **2008** no. 04, (2008) 063–063. http://dx.doi.org/10.1088/1126-6708/2008/04/063.

[10] ATLAS Collaboration, *Run II Jet Physics: Proceedings of the Run II QCD and Weak Boson Physics*. 2007.

[11] J. Nickolls, I. Buck, M. Garland, and K. Skadron, *Scalable parallel programming with CUDA*, Queue **6** no. 2, (2008) 40. http://dx.doi.org/10.1145/1365490.1365500.

[12] S. Kama, *Triggering events with GPUs at ATLAS*, Tech. Rep. ATL-DAQ-PROC-2015-013, CERN, Geneva, May, 2015. https://cds.cern.ch/record/2016471.

[13] ATLAS Collaboration, *Algorithms for the ATLAS high-level trigger*, **51** no. 3, (2004) 367–374. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1310527.

[14] D. Emeliyanov and J. Howard, *GPU-Based Tracking Algorithms for the ATLAS High-Level Trigger*, Tech. Rep. ATL-DAQ-PROC-2012-006, CERN, Geneva, May, 2012. https://cds.cern.ch/record/1450130.

[15] W. Lampl et al., *Calorimeter Clustering Algorithms: Description and Performance*, Tech. Rep. ATL-LARG-PUB-2008-002. ATL-COM-LARG-2008-003, CERN, Geneva, Apr, 2008. https://cds.cern.ch/record/1099735.

[16] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/HLTTrackingPublicResults#Phase_I_Upgrade_public_plots.

[17] https://twiki.cern.ch/twiki/bin/view/AtlasPublic/HLTCaloPublicResults#HLT_Calo_performance.