Title

ATLAS DataFlow: The read-out subsystem, results from trigger and data-acquisition system testbed studies and from modeling

Journal
2005 14TH IEEE-NPSS Real Time Conference, 2005

ISBN
9780780391833

Authors
Vermeulen, J
Abolins, M
Alexandrov, I
et al.

Publication Date
2005

DOI
10.1109/RTC.2005.1547446

ATLAS DataFlow: the Read-Out Subsystem, Results from Trigger and Data-Acquisition System Testbed Studies and from Modeling

J. Vermeulen[1], M. Abolins[2], I. Alexandrov[3], A. Amorim[4], A. Dos Anjos[5], E. Badescu[6], N. Barros[4], H. P. Beck[7], R. Blair[8], D. Burckhart-Chromek[9], M. Caprini[9,6], M. Ciobotaru[9,10], A. Corso-Radu[9], R. Cranfield[11], G. Crone[11], J. Dawson[8], R. Dobinson[9†], M. Dobson[9], G. Drake[8], Y. Ermoline[2], R. Ferrari[12], M.L. Ferrer[13], D. Francis[9], S. Gadomski[7,20], S. Gameiro[9], B. Gorini[9], B. Green[14], M. Gruwé[9], S. Haas[9], W. Haberichter[8], C. Haeberli[7], Y. Hasegawa[15], R. Hauser[2], C. Hinkelbein[16], R. Hughes-Jones[17], M. Joos[9], A. Kazarov[9,18], G. Kieft[1], D. Klose[4], S. Kolos[19,18], K. Korcyl[20], K. Kordas[13], V. Kotov[3], A. Kugel[16], A. Lankford[19], G. Lehmann[9], M. J. LeVine[21], L. Mapelli[9], B. Martin[9], R. McLaren[9], C. Meirosu[9,10], M. Mineev[3], A. Misiejuk[14], G. Mornacchi[9], M. Müller[16], R. Murillo[9], Y. Nagasaka[22], J. Petersen[9], B. Pope[2], D. Prigent[9], Y. Ryabov[18], J. Schlereth[8], J. E. Sloper[9], I. Soloviev[9,18], R. Spiwoks[9], S. Stancu[9,19], J. Strong[14], L. Tremblet[9], G. Ünel[9,19], W. Vandelli[12], P. Werner[9], F. Wickens[23], M. Wiesmann[9], M. Wu[16] and Y. Yasu[24]

[1]NIKHEF, Amsterdam, The Netherlands, [2]Michigan State University, Department of Physics and Astronomy, East Lansing, Michigan, USA, [3]JINR, Dubna, Russia, [4]CFNUL/FCUL, Universidade de Lisboa, Portugal, [5]University of Wisconsin, Madison, Wisconsin, USA, [6]Institute of Atomic Physics, National Institute of Physics and Nuclear Engineering Bucharest, Romania, [7]LHEP, University of Bern, Bern, Switzerland, [8]Argonne National Laboratory, Argonne, Illinois, USA, [9]CERN, Geneva, Switzerland, [10]Politehnica, Bucuresti, Romania, [11]UCL, London, UK, [12]Dept. Fisica Nucleare e Teorica dell'Università di Pavia e INFN, Pavia, Italy, [13]Laboratori Nazionali di Frascati dell'INFN, Frascati, Italy, [14]Royal Holloway, University of London, Egham, UK, [15]Department of Physics, Faculty of Science, Shinshu University, Matsumoto, Japan, [16]Lehrstuhl für Informatik V, Universität Mannheim, Mannheim, Germany, [17]The University of Manchester, Manchester, UK, [18]PNPI, St. Petersburg, Russia, [19]University of California at Irvine, Irvine, California, USA, [20]IFJ-PAN, Krakow, Poland, [21]Brookhaven National Laboratory (BNL), Upton, New York, USA, [22]Hiroshima Institute of Technology, Hiroshima-shi, Hiroshima, Japan, [23]Rutherford Appleton Laboratory, Oxon, UK, [24]KEK, High Energy Accelerator Research Organisation, Tsukuba-shi, Ibaraki, Japan

Abstract— **In the ATLAS experiment at the LHC, the output of readout hardware specific to each subdetector will be transmitted to buffers, located on custom made PCI cards ("ROBINs"). The data consist of fragments of events accepted by the first-level trigger at a maximum rate of 100 kHz. Groups of four ROBINs will be hosted in about 150 Read-Out Subsystem (ROS) PCs. Event data are forwarded on request via Gigabit Ethernet links and switches to the second-level trigger or to the Event builder. In this paper a discussion of the functionality and real-time properties of the ROS is combined with a presentation of measurement and modelling results for a testbed with a size of about 20% of the final DAQ system. Experimental results on strategies for optimizing the system performance, such as utilization of different network architectures and network transfer protocols, are presented for the testbed, together with extrapolations to the full system.**

Index Terms— **ATLAS, LHC, Triggering, Data acquisition, Real time systems.**

I. Introduction

THE ATLAS experiment is one of the four large experiments aimed at studying high-energy particle interactions at the Large Hadron Collider (LHC). The LHC is under construction at the European Laboratory for Particle Physics CERN in Geneva and is scheduled to come into operation in the year 2007. At present the ATLAS experiment [1] is being installed in its underground cavern and the commissioning process has started. The design of the High-Level Trigger and Data-Acquisition (TDAQ) system has been documented in [2]. This paper concentrates on the flow of data in the system [3] and reports on the implementation of the Read-Out Subsystem (ROS) and on results of recent testbed and modeling studies of the TDAQ system.

† deceased.

0-7803-9183-7/05/$20.00 ©2005 IEEE.

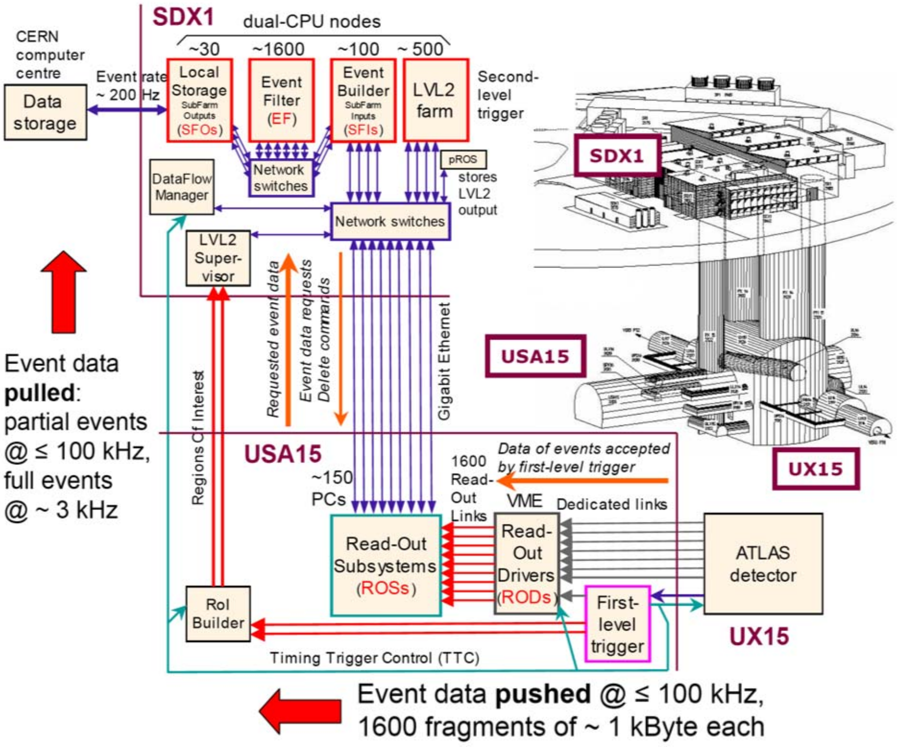

Fig. 1. ATLAS Trigger/DAQ System Overview

II. DataFlow in the ATLAS TDAQ System

In Fig. 1 an overview of the ATLAS Trigger/DAQ system is presented, together with an indication of where the various parts of the system are located.

The first-level trigger, implemented in dedicated hardware, decides for each crossing of beam bunches (one per 25 ns) whether data from the calorimeters and / or muon trigger detectors satisfy the trigger criteria. Accept signals (maximum rate of 100 kHz) are sent to the front-end electronics via the optical fibers of the Timing, Trigger and Control (TTC) system. As a result data pertaining to the trigger are transmitted to the Read-Out Drivers (RODs). RODs are subdetector-specific, their task is to collect and process data (e.g. zero-suppression, but no event selection). The event data are passed on as fast as possible via Read-Out Links (ROL, 160 MByte/s optical fiber) to the Read-Out Subsystem (ROS). The ROS is built from about 150 PCs. Each PC hosts groups, of in most cases, four custom made PCI cards ("ROBINs"). Each ROBIN has three inputs for Read-Out Links and three associated event data buffers, i.e. a single ROS PC receives data from up to 12 ROLs. Each ROS PC is a 4U high rack-mountable PC and is connected to a Gigabit Ethernet network. The same type of ROL, ROBIN and ROS PC is used for all sub-detectors.

The Region of Interest (RoI) Builder [4] receives, via dedicated links, information from the first-level trigger for each first-level accept, formats it and passes the result, again via a dedicated link, to one of the second-level trigger (LVL2) supervisors. This decides which of the processors in the second-level trigger processor farm has to handle the event and sends this processor the RoI information via a network link. The processor requests data from the ROS PCs as needed (possibly in several steps), produces an accept or reject and informs the LVL2 supervisor. For accepts, results produced by the second-level trigger algorithms are sent to a PC functioning as ROS PC, referred to as pseudo-ROS (“pROS”). The LVL2 supervisors pass the decisions to the DataFlow Manager. Accepts irrespective of the outcome of the second-level trigger processing (“forced accepts”) are possible.

For each accepted event the DataFlow Manager chooses an Event Builder processor, referred to as SFI (from “Sub-Farm Input”), and sends it a request to take care of the building of a complete event. The SFI sends requests to all ROS PCs for data of the event to be built. Completion of building is reported to the DataFlow Manager. For rejected events and for events for which event building has completed the DataFlow Manager sends "clears" to the ROS PCs for 100 - 300 events together. On request the event data are passed from SFI to an Event Filter processor, where the final event selection is performed.

ROS PC, second-level trigger, Event Builder and Event Filter applications are all implemented as multi-threaded C++ programs.

III. Requirements

Rates of RoI requests received by the ROS PCs have been estimated with a "paper model", where "paper" refers to "back-of-the-envelope" calculations. In practice, the required calculations are done with a C++ program. The calculations are based on the assumption that the RoI rate does not depend on the η and ϕ of the centre of the RoI, but only on the area in

η- ϕ space associated with the RoI. The RoI rates for each possible RoI location and type (electromagnetic shower, jet, single hadron, muon) are obtained with a straightforward calculation. Inputs for it are: the LVL1 accept rate, exclusive fractional rates for the various LVL1 trigger menu items, the number and type of RoIs associated with each trigger item and the η-ϕ area associated with the RoI location and type. The rates of requests received by each ROS PC and the request rates for each ROL are then obtained using the mapping of the detector onto the ROLs, the acceptance factors of the various LVL2 trigger steps, and the RoI rates for the RoI locations associated with the η-ϕ areas from which data are requested (RoI type and detector dependent).

The model predicts that for the design luminosity trigger menu, per first-level accept on average buffered event fragments from 16.2 ROLs connected to 8.4 ROS PCs will be requested, i.e. from on average 2 ROLs per PC for these ROS PCs. This illustrates that RoI-driven processing is a key property of the ATLAS LVL2 system. With 1 - 1.5 kByte per fragment a network bandwidth of ~ 2 GByte/s is needed at 100 kHz first-level trigger accept rate, instead of ~ 150 GByte/s for full read-out at 100 kHz. Furthermore the maximum rate of requests per ROL by the second-level trigger is about 5 – 8 kHz, instead of 100 kHz that would be required for full read-out. The ROS PC takes care of distributing requests to the ROBINs (for each ROL individually) and of partial event building. The maximum request rate and output event fragment rate per ROS PC, for second-level triggering only, are about 20 – 25 kHz for a few ROS PCs; for the remaining PCs they are less than 15 kHz. The maximum data volume to be transferred is predicted to be less than 35 MByte/s. Note that, if needed, there is some room for minimizing request rates and data volumes per ROS PC by interchanging ROL connections.

Event Building is anticipated to take place at about 3–3.5 kHz at maximum with an associated total bandwidth requirement of 3–5 GByte/s.

In view of possible non-standard triggers the requirement for the request rate per ROL (by LVL2 and Event Builder) that the ROBIN should be able to handle has been set to 21 kHz, i.e. much higher than the requirements resulting from the paper model.

IV. ROBINs

The ROBIN [5], [6] is a 64-bit 66 MHz PCI card. The final version has three inputs for Read-Out Links and one Gigabit Ethernet interface (copper). The latter is not used in the baseline system, but provides an upgrade path. The design is based on a prototype version with two inputs for ROLs and a Gigabit Ethernet interface for optical as well as copper media.

The ROBIN is built around a Xilinx XC2V2000 FPGA [7], a PowerPC PPC440 processor [8] with a clock frequency of 466 MHz and 128 MByte of external memory and a PLX PCI9656 PCI interface [9]. The FPGA receives event fragments from the ROLs and stores them in 64 MByte buffer memories, one per ROL. Event fragments are stored in memory pages, which have a programmable size (between 1 and 128 kByte, a typical value is 2 kByte). The CPU provides the available page locations to the FPGA via a Free Page FIFO that can accommodate up to 1024 entries. For every incoming fragment a new page entry is taken from the FIFO. If the size of the fragment is larger than one page, subsequent entries are taken from the Free Page FIFO. There is no limit to the fragment size at this stage. When a page is full or the fragment is complete, the page information (starting address, length and status) is written to the Used Page FIFO that can buffer 512 page entries. There are three words per entry: page information, event number and status, which includes error information related to fragment transmission quality and format. The Free and Used Page FIFOs for each ROL are separate and independent. The CPU keeps track of used pages and the event numbers associated with these using the information read from the Used Page FIFOs. It handles requests for event fragments input via the ROL specified in the request, which is communicated to the ROBIN via the PCI bus. After looking up the page descriptor (or descriptors in case more than one page is occupied by the event data) of the data in the buffer memory the CPU causes the FPGA to transfer an appropriate header and the data via the PCI bus to the memory of the ROS PC. Requests arriving via the Gigabit Ethernet interface (if used) are handled in a similar way. Delete messages result in descriptors of memory pages, associated with the event data to be discarded, being returned to the Free Page FIFO.

To determine the performance of the ROBIN, measurements have been done with a ROS PC generating requests and deletes as fast as possible. Emulated data were input via the ROL interfaces and generated with the help of a DOLAR card residing in a second PC. This is a FILAR card [10] (a custom PCI card with 4 S-link interfaces) with modified firmware, which takes care of generating the event data without transporting data across the PCI bus. The DOLAR card can fully saturate the output S-links with the event rate throttled by the S-link XOFF signals generated by the ROBIN on buffer memory full conditions. With this setup a linear relation between request rate and incoming event rate (which is equal to the delete rate) was found [11]. For the final version ROBIN it was shown that for 1 kByte fragments a request rate of 27 kHz per ROL (80 kHz in total) can be sustained for an incoming event rate of 100 kHz, i.e. more than the 21 kHz requirement specified. For bookkeeping, the CPU needs per incoming event fragment and per ROL 1.8 μs, and per request per ROL 5.4 μs.

The production of 50 ROBIN boards for the “pre-series”, to be used in the preparatory commissioning phase of the Trigger/DAQ system, is under way, while the full production of 680 boards is being prepared.

V. ROS PC

The software running on the ROS PCs is based on the IOManager (IOM) framework, also used for the ROD Crate DAQ (RCD) [12]. It consists of a multi-threaded C++ program using Linux POSIX threads. The configuration and control of the ROS software is similar to that of the ROD Crate DAQ.

[redacted]
[redacted]

Fig. 2 Thread structure of the ROS application.

Here we concentrate on the threads involved in the data flow for which the thread structure is sketched in Fig. 2.

The receipt of a message from one of the second-level trigger processors or Event Building nodes activates the trigger thread (it may also be activated by an internal "event" in case of emulation of the reception of messages). The trigger thread creates a request object, specifying the action to be executed according to the type and contents of the message, and posts it to a queue. The action can consist of requesting the ROBINs to delete event fragments or of the retrieval of one or more event data fragments from the ROBINs for being sent back to the requesting node.

The request objects are retrieved from the queue by request handler threads. Each handler handles one request at a time, but different handlers can work in parallel in order to achieve better CPU utilization. The number of threads is configurable. In case of data requests, the request handler thread builds a larger fragment from the event fragments received from the ROBINs (as described in the previous section, the ROBINs transfer the requested data from their buffer memories to the memory of the PC) and outputs it to the destination specified in the request object.

Measurements have been made using 12 ROLs connected to 6 prototype ROBINs with 2 ROL inputs [5], in the same way as the measurements described in the previous section, using a ROS PC with a SuperMicro X5DL8-GG motherboard [13] and a 3 GHz Xeon CPU [14]. Request and delete messages were internally generated as well as sent by another PC via the network, using the UDP as well as the TCP protocol. For requests and deletes sent via the network using UDP, a maximum request and response rate of 7.5 kHz was found for event data requested from all 12 ROLs, with 1 kByte event fragments arriving via the ROLs at a rate of 100 kHz. The requests corresponded to Event Builder requests, i.e. the responses consisted of single messages containing event fragments built from 12 fragments of 1 kByte. With TCP and under the same conditions the maximum rate was found to be 7.0 kHz. Note that for LVL2 requests on average the data from only two out of 12 ROLS is needed, i.e. it seems likely that the rates predicted by the paper model can be handled, as is indeed shown by results of further measurements. It is foreseen that in the final system ROS PCs with at least two network interfaces will be used.

Due to the limited number of ROBINs available it has not been possible to set up a system with several ROS PCs equipped with these boards. However, it was found that emulation of the behavior of a ROS PC equipped with 6 prototype ROBINs by means of software is possible. The maximum request rates found for the real system and for the emulated system differ by only a few percent. This emulation is based on emulation of the behavior of a single prototype ROBIN, which is also accurate within a few percent. Also the behavior of a final version ROBIN has been emulated again giving rise to maximum rates within a few percent of those measured. An emulated version of a ROS PC equipped with final version ROBINs could therefore be produced, which is believed to be accurate and which has been used in the latest testbed measurements, discussed in the next section.

VI. Testbed Measurements

A scale model of the final system in the form of a testbed is used for system studies. The testbed is built from up to 24 ROS PCs with two Gigabit Ethernet interfaces, 16 Event Building nodes (SFIs), 15 second-level trigger processors (L2Ps), a PC functioning as DataFlow Manager, 7 PCs functioning as LVL2 Supervisors and one as pseudo ROS, all interconnected via Gigabit Ethernet with the help of three Gigabit Ethernet switches. The ROS PCs run in the emulation mode described in the previous section. The L2Ps do not run algorithms, i.e. they effectively emulate concentrating switches connecting about 10 L2Ps to the central switch or switches in the final system. The testbed therefore represents about 20% of the full system. Apart from the ROS PCs all nodes are dual Xeon nodes, running at 2.4 or 3 GHz and all are running multi-threaded programs written in C++. The UDP protocol is used and delete messages are broadcast by the DataFlow Manager to all ROS PCs.

For Event Building only (i.e. no LVL2 traffic) measurements have shown that the maximum Event Building rate scales with the number of SFIs, with a throughput per SFI determined approximately by the Gigabit Ethernet line speed, if the data volume output per ROS PC (one network interface used) is lower than the maximum. Otherwise the maximum Event Building rate is determined by the number of ROS PCs. In these measurements the data are not output by the SFIs.

Measurements have also been done with LVL2 traffic only (i.e. no Event Building) with per event data requested from random chosen ROS PCs and from a single ROL per ROS PC. The time needed for RoI data collection shows a linear dependence on the number of ROS PCs from which data are requested and is around 0.8 ms for 16 ROS PCs. This is less than 10% of the time budget available per event (for 500 dual CPU machines and 100 kHz event rate this is 10 ms per CPU).

With combined LVL2 and Event Building traffic the LVL2 accept fraction defines the fraction of events that will be built. Typical measurements determine the Event Building rate for varying number of L2Ps and for a given accept fraction, with data from randomly chosen ROLs requested by the L2Ps. For increasing number of L2Ps the rate first increases until it becomes constant, either due to full utilization of the ROS PCs, the SFIs, or the available bandwidth. Measurement results of this type have been reproduced by the "at2sim" discrete event simulation program [15], as illustrated in Fig. 3

[redacted]

Fig. 3. Comparison of testbed results and model predictions (left) and model predictions for the full system (right).

for recent measurement results.

The “full” system has been simulated on the basis of the testbed results. Not all ROS PCs have been taken into account in this simulation, only those from which data can be requested by the LVL2 system, i.e. 127 ROS PCs. The simulated system consisted further of 110 SFIs and 504 dual-CPU L2Ps, with groups of 6 L2Ps connected via concentrating switches to two central switches. Half of the SFIs were connected to one central switch, the other half to the other central switch. ROS PCs had two network interfaces for connecting to both central switches with LVL2 and Event Builder traffic mixed on both network interfaces. The paper model trigger menu and processing sequences were used, but for each subdetector the LVL2 requests were sent to randomly chosen ROS PCs associated with the subdetector, while the algorithm times were set to 0. The results of the simulation (see Fig. 3) show stable operation at 100 kHz first-level accept rate, with about 55 events simultaneously processed by the LVL2 farm (this number is low due to the algorithm times being set to 0), an event building latency of about 15 ms and a largest number of requests queued in the ROS of about 10. Simulation of a configuration where one of the two central switches is used exclusively for Event Builder traffic and the other only for LVL2 traffic also shows stable operation, but there tends to be somewhat more time needed for RoI data collection, while also the queues are somewhat larger. This leads to a preference for “mixed” traffic, which also has the advantage that in case of failure of one central switch still half of the LVL2 and Event Building systems are available.

In the studies described so far the SFIs did not output event data to the Event Filter. In reality Event Filter processors will send requests to the SFIs, which will respond by supplying complete events. A drop of about 15% was observed in the maximum event building rate per SFI in case of input from 12 ROS PCs if the data are output to one or more Event Filter processors. In reality about 150 input messages have to be collected for a single event, i.e. for a single output message. As data are not copied in the SFI a smaller relative performance drop is expected.

Several studies on possible optimizations have been undertaken. One of them is on the choice of the C++ compiler. It has been found that the icc (version 8.0) [16] and gcc3.2.3 compilers give similar results, the maximum event building rate of an SFI was found to be 6% higher in case of using gcc with Pentium 4 optimizations instead of PentiumPro optimizations. Another study concerned the use of a dual-processor PC as ROS PC. This has been studied only in emulation mode so far. The maximum rate that can be handled increases 5-10% if “affinity” instead of “auto-affinity” for assigning threads to the CPUs is used. For “auto-affinity” no difference with the performance of a single-processor ROS PC is seen. Concerning the choice of network protocol it has been found that using UDP exclusively or UDP only for event data transfers and TCP for control messages results in the same performance of the testbed. However using TCP only has a negative effect on the performance. The possible use of 10 Gbit Ethernet in part of the system was also studied [17]. For realizing a larger scale testbed we plan to use the network testers described in [18]. Finally, discrete event simulation studies have shown that improper settings of the configuration parameters may lead to a significant reduction in the maximum first-level accept rate that can be handled or even to system instability.

VII. Conclusions and Outlook

The results obtained from measurements and modeling indicate that there are no “show-stoppers” with respect to obtaining the required performance for the flow of data in the ATLAS Trigger/DAQ system. Installation and commissioning of the “preseries”, a precursor to the final system, has been started at “Point 1” (the site of the experiment), to be followed by stepwise building up of the full system in 2005 and 2006, taking into account the experience gained with the “preseries”.

References

[1] *ATLAS Detector and physics performance technical design report.* CERN-LHCC 99-13, The ATLAS Collaboration (1999), http://atlasinfo.cern.ch/Atlas/GROUPS/PHYSICS/TDR/access.html.

[2] *ATLAS High-Level Trigger, Data-Acquisition and Controls Technical Design Report*, ATLAS TDR 016, CERN/LHCC/2003-022, June 2003, http://atlas-proj-hltdaqdcs-tdr.web.cern.ch/atlas-proj-hltdaqdcs-tdr/.

[3] H.P. Beck et al., “The Base-Line DataFlow System of the ATLAS Trigger and DAQ”, IEEE Trans. Nucl. Sci., Vol. 51, 2004, pp. 470-475.

[4] R. Blair et al., “RoI Builder Manual”, https://edms.cern.ch/document/566646/3.4

[5] B. Green et al., “ATLAS Trigger/DAQ RobIn Prototype”, IEEE Trans. Nucl. Sci., Vol. 51, 2004, pp. 465-469.

[6] B. Green et al., “ATLAS TDAQ/DCS ROS ROBIN Production Readiness Review: Design Document”, 2005, https://edms.cern.ch/document/555100/1

[7] http://www.xilinx.com/products/tables/fpga.htm.

[8] http://www-03.ibm.com/chips/products/powerpc/cores/.

[9] http://www.plxtech.com/products/io_accelerators/PCI9656/.

[10] http://hsi.web.cern.ch/HSI/s-link/devices/filar/.

[11] B. Gorini et al., “ATLAS TDAQ/DCS ROS ROBIN Production Readiness Review: Performance Measurements”, 2005, https://edms.cern.ch/document/555103/1.

[12] S. Gameiro et al., “The ROD Crate DAQ of the ATLAS Data Acquisition System”, these proceedings.

[13] http://www.supermicro.com/products/motherboard/Xeon/.

[14] http://www.intel.com/products/processor/xeon/.

[15] R. Cranfield et al., “Computer modeling the ATLAS Trigger/DAQ System Performance”, IEEE Trans. Nucl. Sci., Vol. 51, 2004, pp. 532-538.

[16] http://www.intel.com/software/products/compilers/clin/.

[17] S. Stancu et al., “ATLAS TDAQ DataFlow Network Architecture Analysis and Upgrade Proposal”, these proceedings.

[18] M. Ciobotaru et al., “GETB, a Gigabit Ethernet Application Platform: its Use in the ATLAS TDAQ Network”, these proceedings.