# The Dataflow System of the ATLAS DAQ/EF ”-1” Prototype Project

*G. Lehmann*[1,3] , *G. Ambrosini*[3,9], *E. Arik*[2], *H.P. Beck*[1], *S. Cetin*[2], *T. Conka*[2], *A. Fernandes*[3], *D. Francis*[3], *Y. Hasegawa*[4], *M. Joos*[3], *J. Lopez*[3,10], *A. Mailov*[2], *L. Mapelli*[3], *G. Mornacchi*[3], *Y. Nagasaka*[7], *M. Niculescu*[3,5], *K. Nurdan*[2], *J. Petersen*[3], *D. Prigent*[3], *J. Rochez*[3], *L. Tremblet*[3], *G. Unel*[3] , *S. Veneziano*[3,6], *Y. Yasu*[8]

[1] Laboratory for High Energy Physics, University of Bern, Switzerland
[2] Department of Physics, Bogazici University, Istanbul, Turkey
[3] CERN, Geneva, Switzerland
[4] ICEPP, University of Tokio, Tokio, Japan
[5] Institute of Atomic Physics, Bucharest, Romania
[6] I.N.F.N. Sezione di Roma, Roma, Italy
[7] Nagasaki Institute for Applied Science, Nagasaki, Japan
[8] High Energy Accelerator Research Organization (KEK), Japan
[9] Now at Lightning Instrumentation S.A., Lausanne, Switzerland
[10] Now at EDF, Grenoble, France

**Abstract**

In 1996 the Data Acquisition (DAQ) group of the ATLAS Collaboration started a project for the design and implementation of a full DAQ/Event Filter (EF) prototype, based on the Trigger/DAQ architecture described in the ATLAS Technical Proposal. The aim of this prototype was to allow for hardware and software technology investigations as well as their integration aspects in order to reach maturity for the final ATLAS DAQ system design. Being a pre-design prototype it is referred to as ATLAS DAQ Prototype ”-1”. It consists of a ”vertical” slice of the ATLAS DAQ/EF architecture, including all the hardware and software elements of the data flow, its control and monitoring as well as all the elements of a complete DAQ system, from the detectors Read Out Driver to data recording.

This paper describes the dataflow component of the prototype, its design, implementation and performance. More emphasis is given to the description of the Event Builder, the sub-system which merges data fragments coming from different detector parts into full events, since this is the critical element for the scalability to ATLAS sizes of the proposed architecture. Results of modelling studies based and tuned on the present implementation will be shown.

Keywords: ATLAS, DAQ, event builder, LHC, switching network, VME bus

## 1 Introduction

The data flow component of the ATLAS [1] DAQ-1 system [2] is responsible for moving the event data from the detector read-out links to the final mass storage, interacting with the three ATLAS trigger levels: level one (LVL1), level two (LVL2) and Event Filter (EF) [3]. It also provides event data for monitoring purposes and implements local control for the various data flow elements. A global view is shown in figure 1.

Three main functions are provided by the data flow, namely:

- the Front End DAQ which buffers the event data coming from the detector, possibly performs some data collection with the associated event formatting, and interfaces with the second level trigger
- the Event Builder (EB), which merges data fragments coming from the different detector parts into full, formatted events,
- the Farm DAQ which provides data flow support to the processor farms of the event filter and interfaces with the mass storage system.

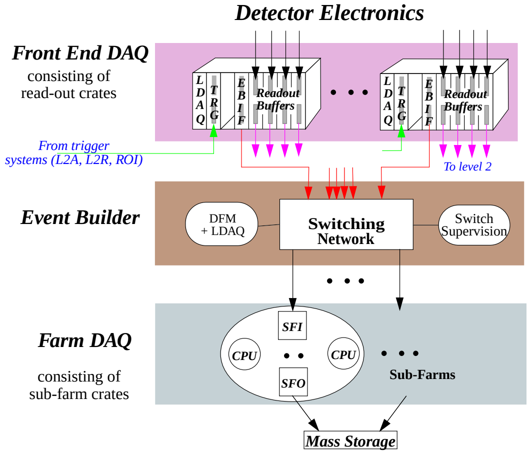

**Figure 1:** The Dataflow system.

Each of these functional elements is factorised in a local DAQ (LDAQ) component, which implements local control for the various data flow elements, interfaces to the Back End[1] and provides data samples for monitoring purposes, and in DAQ Units which strictly deal with the receiving, storing, moving and deleting of data fragments.

The segmentation of the detector read-out ($\sim$ 1600 read out links) suggests to organise the Front End DAQ into a number of modular, independent elements, the read-out crates (ROCs), each supporting the read-out from one or more detector segments and having one or more connections to the Event Builder. Similarly, the Farm DAQ is seen as a set of separate modules, the sub farm crates (SFCs), each corresponding to an Event Builder output.

The EB, contrary to the Front End and the Farm DAQ, cannot be further split into independent elements, since it merges all the data fragments. As such it is monolithic, hence a satisfactory performance of the EB prototype alone is not sufficient to assess the value of its design and implementation. Simulation studies are needed in order to extrapolate the performance to the case where there are hundreds of fragments being merged into one complete event and where an aggregate bandwidth of several Gbytes/s is required.

## 2 High Level Design of the Dataflow

The high level design of the dataflow components has been extensively documented in [3] [4] [5]. Here we only recall the flow of a typical event through the dataflow system. When an event is accepted by the first trigger level, data are read out and stored in Read Out Buffers (ROBs) which reside in the different ROCs. Here they are made available to the second level trigger and kept until the LVL2 decision is made. If the event is discarded by LVL2, data are removed from the ROBs, otherwise they are forwarded to the Event Builder Interface (EBIF). If more than one ROB is associated to a single EBIF, ROB-fragments are collected and formatted before being accessed by the Event Builder. The event ID of a LVL2 accepted event is sent to the Data Flow Manager (DFM) of the EB which assigns an SFC for the construction of the full event. Event ID and SFC ID are broadcast to the EBIFs, which send the corresponding data fragments via a network to the appropriate Sub Farm Input (SFI). The SFI counts the arrived fragments, builds a full, formatted event, notifies the DFM when the event is completed and forwards the data to the event filter farm.

[1] The Back End is the system responsible for the global configuration and run control of the DAQ

If the event passes the selection criteria of this last trigger stage, data are sent, via the Sub Farm Output (SFO) to mass storage.

## 3 Implementation of the Dataflow

The dataflow prototype has been implemented on VME bus [8] based single board computers (SBCs) running LynxOS as operating system. As an alternative, PCs running Linux have been introduced at the SFC and LDAQ level. Communication protocols have been defined and developed to support both the intracrate [6] and the event building traffic [7]. These protocols hide the technology specific implementation aspects of the communications allowing to change bus types, switching networks, etc. Furthermore, the implementation of DAQ applications has been organized such that the data flow tasks are decoupled from the infrastructure (e.g. task scheduling and buffer managment) of a process. Such a factorization gives the possibility to move tasks between processors and find the optimal CPU and buses utilization.

## 4 Performance Measurements

Performance measurements have been carried out on a 2 ROCs by 2 SFCs prototype. The results which will be presented correspond to a homogeneous SBC environment in which all processors are RIO II $8062$ ($200$ $MHz$) and RIO II $8061$ ($100$ $MHz$) from CES [9]. The VME bus is used for intracrate communications and ATM [10] as event building technology. Each ROC houses two ROBs and performs local data collection to a single EBIF. The EB consists therefore of five nodes (2 EBIFs + 2 SFIs + 1 DFM). Data are generated locally in the ROBs and a trigger module distributes emulated LVL2 decisions to the two ROCs and to the DFM via a PVIC bus [11]. While for local ROC measurements [4] the critical parameter is the rate at which fragments can be input to the ROBs, which has to be greater than $75$ $kHz$ [3], the interesting performance figure for the global dataflow system is the rate at which events can be pushed through the whole DAQ chain. This is required to be in the order of $1-2$ $kHz$, the expected LVL2 trigger accept rate [3]. Results are shown in figure 2.

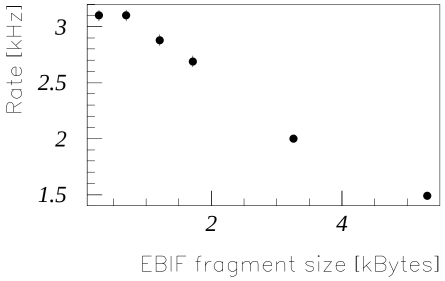

**Figure 2:** The Dataflow performance. The sustained L2 accept rate is plotted as a function of the EBIF fragment size.

In the present implementation the rate is determined by the performance of the EBIF processor which has to carry out data collection over VME bus, receive control information from the data flow manager of the EB and send out fragments over the ATM network. The EBIF performance could be improved with the introduction of a secondary intelligent element (e.g. an intelligent PMC) which could take care of the networking operations, hence offloading the main CPU.

## 5 Scalability studies

The performance of the dataflow at the final ATLAS size is strongly dependent on the behavior of the Event Builder, as the number of nodes attached to the network increases. In the absence of a large scale prototype, modelling has to be used to evaluate the design and implementation of the Event Builder. A standalone EB has been extracted from the dataflow prototype: data are generated in the EBIFs and events are deleted at the SFI after having been built and formatted. Its performance in different configurations (see table I) is shown in figure 3.

**Table I:** Configuration for the two event building prototypes

| # EBIFs | # SFIs | Processors | Op. system | Network technology | Protocol |
|---|---|---|---|---|---|
| 4 | 4 | RIOII 8062 200 $Mhz$ | LynxOS | ATM 155 $Mbit/s$ | AAL5 |
| 7 | 7 | Pentium 450 $MHz$ | Linux | Gigabit Ethernet | TCP/IP |

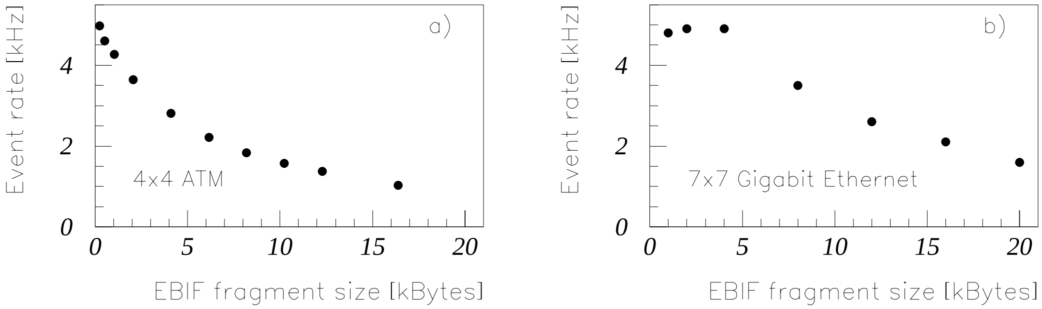

**Figure 3:** The EB performance for a) ATM, $155\ Mbit/s$ and b) Gigabit Ethernet. The sustained event rate is plotted as a function of the EBIF fragment size.

While for Gigabit Ethernet a detailed model of the switching network is needed to take into account its dynamic behavior, for ATM, where the traffic congestion avoidance can be handled at the individual nodes via QoS techniques, the switching network can be modelled as an ideal routing element which only introduces a constant delay between input and output. The prototype based on ATM was therefore used to extract all relevant parameters for the simulation and to perform several tests in order to understand the scalability of the system: the time to build a single event as a function of the number of fragments composing it has been proven to increase fairly linearly (figure 4). All the processing times have been time stamped showing approximately a constant behaviour independently on the number of nodes involved in event building.

A simulation program using PTOLEMY [12] has been developed according to the design and implementation of the EB prototype and tuned with the measurements carried out using the ATM ($155\ Mbit/s$) technology. The latest estimated event size for the ATLAS experiment corresponds to $\sim 2.2\ MBytes$ [13]. The required event building rate is in the order of $1 - 2\ kHz$, resulting in an aggregate bandwidth of $4.5\ GBytes/s$. The minimal network configuration, if one imposes that single links shall not be utilized to more than 70% of their capability, has to forsee $400$ EBIFs and $400$ SFIs for ATM $155\ Mbit/s$, or, alternatively, $100$ EBIFs and $100$ SFIs for ATM $622\ Mbit/s$. Figure 5 shows the simulation results for a) ATM155 and b) ATM622 using the AAL5 protocol. The extrapolated rate is strongly dependent on the assumptions made on the evolution of processing times as a function of the number of nodes. The rate upper limit is the

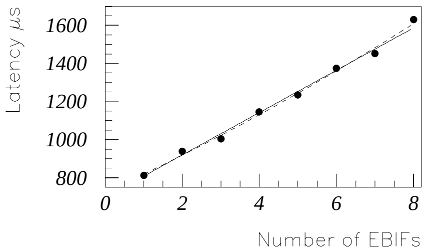

**Figure 4:** The time to build a single event. Data can be fitted with a linear function $f(x) = 697 + 111x$ (full line) or with a quadratic function $f(x) = 723 + 93x + 2.2x^2$ (dashed line). A quadratic dependency of the event building time as a function of the number of EBIFs is expected if there are one or more functions called for every fragment which have themselves a linear dependancy of their processing time on the number of nodes in the EB.

case in which the processing time per fragment stays constant as the EB system grows[2] while the lower limit is the worst case scenario which assumes a linear dependancy of the processing time per fragment on the number of EBIFs, thus a quadratic increase of the event building time (see figure 4).

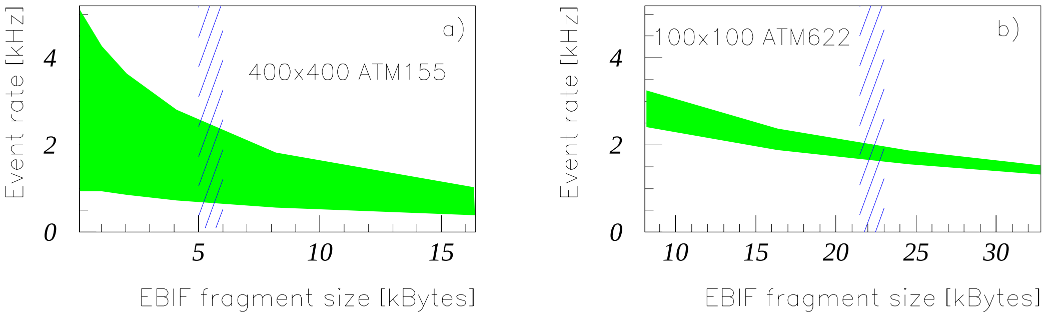

**Figure 5:** Simulation results for a) a $400$ EBIFs by $400$ SFIs Event Builder with ATM155 and b) a $100$ EBIFs by $100$ SFIs Event Builder with ATM622. The shaded area shows the expected event rate as a function of the fragment size. The dashed area shows expected fragment size for ATLAS.

## 6 Conclusions

A full dataflow system has been designed and implemented showing that, at least for small scale prototypes, the required performance of $75\ kHz$ LVL1 accept rate [4] and $1-2\ kHz$ LVL2 accept rate can be achieved. Areas of further possible improvement have been identified concerning both hardware and software. The simulation of the event building sub-system has demonstrated that the dataflow system design is scalable and that the target performance for the overall ATLAS DAQ is in reach.

The DAQ/EF “-1” prototype is now ready to be integrated with the detector read out links and introduced as data acquisition system for the ATLAS test beams.

[2] there is a perfect scaling behavior and the time to build an event increases linearly with the number of fragments it is composed of

## References

1 The ATLAS Collaboration, “Technical Proposal for a General Purpose pp Experiment at the Large Hadron Collider at CERN”, CERN/LHCC/94-43

2 G. Ambrosini et al., “The ATLAS DAQ and Event Filter Prototype ”-1” Project”, presented at Computing in High Energy Physics 1997, Berlin, Germany. `http://atddoc.cern.ch/Atlas/Conferences/CHEP/ID388/ID388.ps`

3 The ATLAS Collaboration, “ATLAS DAQ, EF, LVL2 and DCS Technical Progress Report”, p. 136 ff, CERN/LHCC/98-16

4 S. Veneziano et al., “The Read-Out Crate in ATLAS DAQ/EF prototype -1”, presented at Computing in High Energy Physics 2000, Padova, Italy

5 G. Ambrosini et al., “Event Building in the ATLAS DAQ/EF Prototype ”-1” ”, presented at Computing in High Energy Physics 1998, Chicago, USA. `http://atddoc.cern.ch/Atlas/Conferences/CHEP98/Welcome.html`

6 G. Crone et al., “The DAQ-unit Message Passing API”. `http://atddoc.cern.ch/Atlas/Notes/091/Note091-1.html`

7 G. Ambrosini et al., “Event Builder Network I/O Library”. `http://atddoc.cern.ch/Atlas/Notes/067/Note067-1.html`

8 `http://www.vita.com`

9 `http://www.ces.ch`

10 `http://www.atmforum.com`

11 `http://www.ces.ch/Products/Connexions/PVICFamily/PVIC.html`

12 `http://ptolemy.eecs.berkeley.edu/index.html`

13 P. Clarke et al., “Detector and Read-Out Specification, and Buffer-ROI Relations, for Level-2 Studies”, ATL-DAQ-99-014