A Scheme of Read-Out Organization for the ATLAS High-Level Triggers and DAQ based on ROB Complexes

D. Calvet, O. Gachelin, M. Huet, I. Mandjavidze

CEA Saclay DAPNIA, 91191 Gif-sur-Yvette CEDEX, France

ATL-DAQ-2000-014
16/03/2000

Abstract

This paper describes a possible organization of the ATLAS High-Level Triggers and DAQ read-out system downstream the Read-Out Drivers. It is based on the ROB Complex concept which assumes that each read-out unit is formed by several input buffer modules sharing a network interface to a common Trigger/DAQ data collection network. An implementation of such ROB Complex based on PCI bus to connect read-out buffers, a control processor and a network interface card is presented. The total number of ROB Complexes required for ATLAS, as well as the number of CompactPCI crates housing them are estimated. The results obtained from measurements on a ROB Complex prototype integrated in the ATLAS Level 2 Trigger ATM Testbed are given. The feasibility of some data preprocessing within a ROB Complex is shown.

I. INTRODUCTION

Currently several R&D activities are under way to investigate different options for the read-out system of the ATLAS High-Level Triggers and DAQ (referred to as T/DAQ read-out system in this document). Their goal is to provide an input to the ATLAS T/DAQ technical proposal under preparation. Within the Saclay group a particular scheme is being studied which is well adapted to the sequential event selection strategy favoured by the Level 2 Trigger community.

This paper is organized as follows. In section II the concept of the ROB Complex is recalled. Then the required performance for the ATLAS T/DAQ read-out system is evaluated with the help of paper model studies. This is followed by an implementation of a ROB Complex on commercial components. At the end of the section a possible organization for the ATLAS T/DAQ read-out system is presented. Section III outlines the work that has been done in order to validate the feasibility of the ROB Complex principles. It describes the prototype hardware and software developed. Then some results of performance measurements performed on the ROB Complex prototype and its components are presented.

II. COMPACTPCI ROB COMPLEX SCENARIO

A. Concept

The input data rate of the ATLAS T/DAQ read-out system is determined by the Level 1 accept rate of 75 (100) kHz. The input bandwidth of individual read-out buffers can be as high as 160 Mbyte/s. Due to the RoI based event selection, each buffer delivers data to the Level 2 trigger at lower rates, up to ~10 kHz. Full event data collection is performed at most at the Level 2 accept rate which is estimated to a few kilo-Hertz. As a result, the output bandwidth of each buffer is much lower than its input bandwidth. Based on these considerations, it has been proposed in [1] to group several read-out buffers (ROB$_{IN}$-s) in clusters, sharing an interface to a common data collection network for High-Level Triggers and DAQ systems (Figure 1). Operation of such cluster, called a ROB Complex, is controlled by a general purpose processor – the ROB Controller. Within each ROB Complex, a bus of adequate bandwidth connects the ROB$_{IN}$-s, the network interface card (NIC) and the ROB Controller. Depending on the run control and monitoring schemes adopted, yet another processor module might be used to address the necessary functionality.

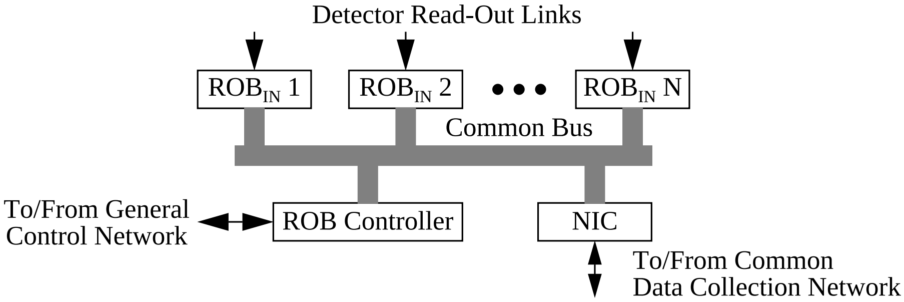


Figure 1. ROB Complex Organization

Two possible modes of operation can be used concurrently for data collection within a ROB Complex. In the “Mapped Memory Mode”, the event memory of each $\mathrm{ROB_{IN}}$ and the memory of the ROB Controller are directly accessible by the network interface card connected to the data collection network. Therefore, the transmission of event data fragments and associated control structures, can be performed by the NIC using a chained data transfer mechanism. This mode of operation reduces data movement over the common bus, as well as the task of the $\mathrm{ROB_{IN}}$-s and the ROB Controller. Preprocessing of event fragments in $\mathrm{ROB_{IN}}$-s is not excluded. In the “Copy Mode”, data is first moved from the $\mathrm{ROB_{IN}}$-s event memories to the ROB Controller’s memory, either by the $\mathrm{ROB_{IN}}$-s themselves or by a DMA mechanism of the ROB Controller. When this operation is completed, data is sent from the ROB Controller’s memory to the data collection network by the NIC. In addition to a local preprocessing within the $\mathrm{ROB_{IN}}$-s, this mode of operation allows preprocessing of event fragments at the ROB Complex level, prior to their transfer to the High-Level Triggers and DAQ systems.

B. Design Considerations

For a given throughput of a network link, the number of $\mathrm{ROB_{IN}}$-s that can be grouped within a ROB Complex depends not only on the $\mathrm{ROB_{IN}}$-s output bandwidth, but also on the rate at which the $\mathrm{ROB_{IN}}$-s have to supply data to the Level 2 trigger and the event filter process. In general, both parameters, bandwidth and rate, vary depending on the detector considered and within each detector as well. For obvious reasons, the sum of the $\mathrm{ROB_{IN}}$-s average output bandwidth within a ROB Complex cannot exceed the bandwidth of its network link. The constraints on rate are imposed by the following considerations. For each data collection request, the ROB Controller should notify the $\mathrm{ROB_{IN}}$-s concerned and collect their response messages, even if this does not imply an actual movement of event data. For RoI data requests, only a fraction of the $\mathrm{ROB_{IN}}$-s within each ROB Complex may be concerned. However, for some request of Level 2 (e.g. TRT Full Scan, Missing E_T calculation) and for full event building, all the $\mathrm{ROB_{IN}}$-s have to deliver their data. In these cases, the ROB Controller should be able to operate at a frequency equal to the product of the request rate by the number of $\mathrm{ROB_{IN}}$-s within a ROB Complex. In the absence of a multicast mechanism on the common bus, the same consideration applies to event data clear requests which have to be delivered to all $\mathrm{ROB_{IN}}$-s.

Table 1 summarizes some results from modelling [2] relevant to the ATLAS detector read-out organization (the Level 1 rate is 75 kHz, the Level 2 accept rate is 2 kHz, the ‘Scan’ column corresponds to low luminosity operation with B-physics triggers, the ‘Low’ column corresponds to low luminosity operation without the B-physics triggers, the ‘High’ column corresponds to high luminosity operation). The Full Scan triggers generate high, 10-12 kHz, data request rate in the inner detector. Taking into account a full event building rate of about 2 kHz, the rate constraint suggests that a grouping factor of 4 for the ROB Complexes of the inner detector. The electromagnetic and hadronic calorimeters require high, 5-7 MByte/s output bandwidth. This is partly due to full event building which is performed for every event accepted by the Level 2 trigger. Because the data collection rate for the ROB_{IN}-s in the calorimeters is sufficiently low, the grouping factor is determined by the throughput of the network link chosen. A ROB Complex with one ROB_{IN} would generate 70% and 40% load on a 100 Mbit/s Ethernet and 155 Mbit/s ATM link respectively. This option results in a fairly large data collection network where about 1000 ports are used just to connect all ROB_{IN}-s. Therefore, links with higher throughput could be more appropriate for the calorimeter ROB Complexes. Possible solutions are 622 Mbit/s ATM or Gigabit Ethernet links which allow grouping factors of 4 to 6 ROB_{IN}-s. The muon sub-detector does not seem to impose severe constraints, neither for the data collection request rate nor for the output bandwidth of a ROB Complex. The number of ROB_{IN}-s per ROB complex might be dictated by detector read-out segmentation considerations and full event building rate. Grouping factors of 12 to 16 seems possible.

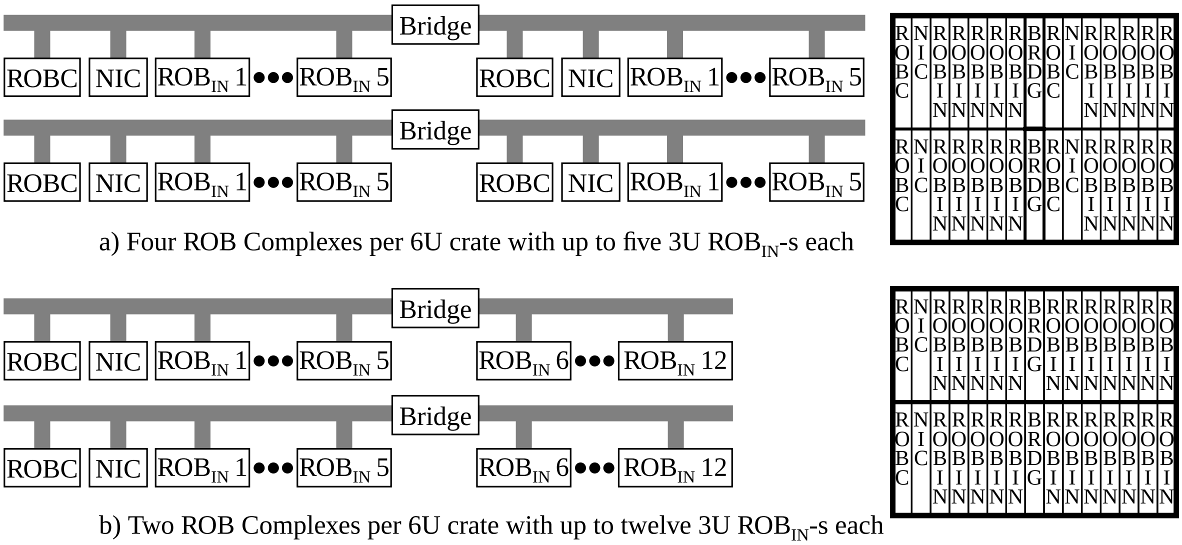
Table 1: Average data request rate and output bandwidth for ROB_{IN}-s in different subsystems

Detector Subsystem	Number of ROB_{IN}-s	Level 2 Data Request Rate (kHz)			Level 2 and EB Bandwidth (MB/s)		
		Scan	Low	High	Scan	Low	High
Muon Precision	192	1.0	0.6	0.3	2.4	2.1	1.9
Muon Trigger	48	2.0	1.2	0.6	1.5	1.2	1.0
EMC	760	1.5	1.1	1.5	5.1	4.7	5.1
HAC	98	3.0	2.0	1.6	6.7	5.6	5.2
TRT	256	11.0	0.9	0.6	4.8	1.8	2.6
SCT	92	11.0	1.4	1.0	3.3	0.9	4.8
Pixel	84	12.0	2.2	1.5	1.1	0.3	2.8
Total	1530				~6.7 GB/s	~5.2 GB/s	~6.1 GB/s

C. ROB Complex Implementation

The long lifetime of the ATLAS experiment, system development and maintenance considerations suggest to find solutions based on the well proven industry standards. The modular structure of the ROB Complex allows to use widespread technologies in many places. This is true for networking where ATM and Ethernet are good candidates, for the ROB Controller with an implementation based on a general purpose single board computer running a real-time operating system, and to some extent for the ROB_{IN}-s as well (several vendors produce I/O boards based on fast DSP-s or CPU-s). For the common interconnection bus within a ROB Complex, the CompactPCI standard [3] seems a good choice today because of its high bandwidth and performance, robustness, industry support and price. The 8-slot limit of a single CompactPCI bus can be overcome by using transparent PCI-to-PCI bridges that link independent bus segments into the desired hierarchy. Each segment contains the arbitration logic to build a multiprocessor system. For example, some vendors propose 3U crates with 14-slots or 6U crates with 27 slots ([4], [5]). The diversity of mechanical support gives the possibility to implement different organizations of ROB Complexes within a CompactPCI chassis. Figure 2 illustrates some configurations that might be used within the ATLAS T/DAQ read-out system.

In both examples the same mechanical and electrical configuration of a 6U crate serves for two different organizations of ROB Complexes with 5 and 12 ROB_{IN}-s respectively. While the former can be used for the inner detector and calorimeter read-out, the latter is more suitable for the muon subsystem. Note that a 6U chassis with four electrically independent busses allows to have four ROB Complexes in it, with up to 6 ROB_{IN}-s each. Very large grouping factors (up to 26) can be achieved by transparently bridging four PCI bus segments. Inversely, connecting bus segments with few PCI slots allows for many ROB Complexes with small grouping factors.



a) Four ROB Complexes per 6U crate with up to five 3U ROB$_{IN}$-s each

b) Two ROB Complexes per 6U crate with up to twelve 3U ROB$_{IN}$-s each

Figure 2. Some possible configurations of ROB Complexes in CompactPCI chassis

D. A Possible Organization of the ATLAS T/DAQ *Read-out system*

These considerations can be used to make some estimates for the organization of the ATLAS T/DAQ read-out system. Table 2 gives the number of ROB Complexes and the number of read-out crates calculated with the following assumptions: the bandwidth of the data collection links is ~80 MByte/s; grouping factors for ROB$_{IN}$-s in the muon, calorimeters and inner detectors are 12, 6 and 4 respectively; ROB Controller, NIC and ROB$_{IN}$-s are implemented in 3U form factor, 6U CompactPCI chassis are used.

Table 2: Estimates for the ATLAS T/DAQ read-out system organisation

Detector Subsystem	Number of ROB$_{IN}$-s	Grouping Factor	Number of ROB Complexes	ROB Complexes per crate	Number of crates
Muon Precision	192	12	16	2	8
Muon Trigger	48	12	4	2	2
EMC	760	6	127	4	32
HAC	98	6	17	4	5
TRT	256	4	64	4	16
SCT	92	4	23	4	6
Pixel	84	4	21	4	6
Total	1530		272		75

According to these estimates, the 1530 read-out buffers of ATLAS could be grouped in 272 ROB Complexes housed in 75 CompactPCI chassis. At most, a ROB Complex should sustain ~14 KHz of data request rate (in the Pixel detector) corresponding to an internal request servicing of ~56 kHz. In the scheme envisaged here, each ROB Complex has a single network link that transports both Level 2 and event building data. About 300 links with a usable bandwidth of 80 Mbyte/s are necessary to connect the ATLAS T/DAQ read-out system to the data collection network. The maximum load on such links does not exceed 50% (for the hadronic calorimeter), therefore, leaving some safety margins for the operation at 100 kHz Level 1 trigger rate.

The study of the network organization and of its behaviour under the estimated data flow patterns are tasks for computer simulation and demonstrator systems. Both the size of the network and the bandwidth requirements can be reduced if event filtering algorithms that operate on partial, rather than full, event data can be used.

III. Prototype Studies

A. Hardware and Software Organization of a ROB Complex Prototype

The validation of the proposed principles for the ROB Complex organization is going on within the ATLAS Level 2 trigger Pilot Project ROB Complex working group. At Saclay a $\mathrm{ROB_{IN}}$ board is under development with 8 MByte of event memory, a FPGA logic that handles input data streams and a 100 MHz I960JT processor responsible for the management of the event memory and the service of event data requests (Figure 3). A first version contains a processor subsystem with a 33 MHz I960JC CPU and 512 kByte of system memory. Three units of the version #1 prototype are currently available.

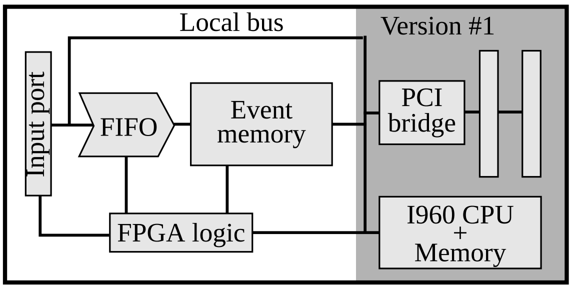


Figure 3. $\mathrm{ROB_{IN}}$ prototype and its first version.

Thanks to the flexibility of the PMC form factor chosen for our first version of the $\mathrm{ROB_{IN}}$, a prototype ROB complex with up to three $\mathrm{ROB_{IN}}$-s has been assembled and tested in three different environments:

- In a CompactPCI crate with a 6U 200 MHz PowerPC CPU board [6] used to perform the ROB Controller functions. Four 3U PMC-to-CompactPCI adapter boards are used to plug the three $\mathrm{ROB_{IN}}$ prototypes and an ATM network interface. When available two more $\mathrm{ROB_{IN}}$-s will be added. The PowerPC CPU board runs LynxOS real-time operating system.
- In a VME crate with a 6U 300 MHz PowerPC CPU board [7] used as the ROB Controller. The two PMC slots of the CPU board are occupied by an ATM network interface and a $\mathrm{ROB_{IN}}$. The PCI bus of the CPU board is extended with a PMC Extender Board (PEB, [8]) which supports two $\mathrm{ROB_{IN}}$-s. By adding yet another PEB, the number of $\mathrm{ROB_{IN}}$-s in this ROB Complex could be increased to five. As for the previous platform, the operating system is LynxOS.
- In a 400 MHz Pentium II PC where the host processor is executing the functions of the ROB Controller. One of the PC's PCI slots is occupied by an ATM network interface; the others are populated by up to three $\mathrm{ROB_{IN}}$-s. On the type of PC used, a free PCI slot remains and one more $\mathrm{ROB_{IN}}$ could be added. For operation on a Pentium, the ROB Complex software has been ported to Linux.

For all these three platforms, the ROB Complex is a multi-PCI bus system that use transparent PCI-to-PCI bridges. The ATM network interface used is based on NicStar segmentation and re-assembly chip from IDT [9] and provides a 155 Mbit/s connection to the common data collection network.

The structure of the ROB Complex software is presented on Figure 4. It is organized in layers, where each layer provides services to the subsequent upper layer.

The ROB Complex Application which runs on the ROB Controller's host processor is responsible for start-up, initialization and servicing messages from/to the network interface. Two types of data request messages are distinguished. The "selective data requests" are forwarded only to the ROB$_{IN}$-s that contain the requested data. The “full event data requests” are forwarded to all ROB$_{IN}$-s. When all the ROB$_{IN}$-s concerned respond with event data, a ROB Complex response message is prepared and sent to its destination node. Event clear messages are sent to all ROB$_{IN}$-s. The local data collection latency, the rate at which various types of requests are serviced, as well as many other operational parameters are constantly measured. This statistic is used to respond to monitoring requests.

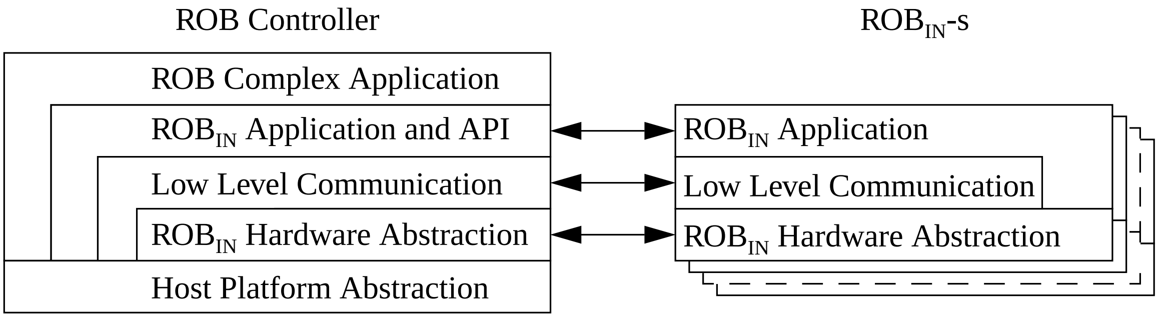

ROB Controller
ROB$_{IN}$-s
ROB Complex Application
ROB$_{IN}$ Application and API
Low Level Communication
ROB$_{IN}$ Hardware Abstraction
Host Platform Abstraction
ROB$_{IN}$ Application
Low Level Communication
ROB$_{IN}$ Hardware Abstraction

Figure 4. ROB Complex software organization.

The ROB$_{IN}$ Application and API layer allows for communication between the ROB Controller and the ROB$_{IN}$-s. The ROB$_{IN}$ Application which runs on the ROB Controller’s host processor establishes a connection with each ROB$_{IN}$. It receives the data collection, event clear and monitoring requests from the ROB Controller and passes them to the ROB$_{IN}$ using the ROB$_{IN}$ API. The API defines and implements the function calls for the ROB$_{IN}$ initialization, data collection, event clearing and monitoring. When a ROB$_{IN}$ finishes to service a data request, the ROB$_{IN}$ Application forms a ROB$_{IN}$ response message by adding a ROB$_{IN}$ header information to the received event data. It then passes it to the ROB Complex Application layer.

The ROB$_{IN}$ Application which runs on the ROB$_{IN}$ local processor performs event management based on a hash table (Figure 5). The event memory of a ROB$_{IN}$ is divided into fixed size pages. The FPGA logic receives event data from the input port and places it into a number of memory pages. When the reception of an event is completed, the FPGA logic passes to the ROB$_{IN}$ Application the identifiers of the memory pages that contain the event data as well as some status information. The ROB$_{IN}$ Application extracts an event descriptor from the Free Event Descriptor Queue. It fills this descriptor with the received event information and places it in the Hash Table, chaining, if necessary, the event descriptors in raws.

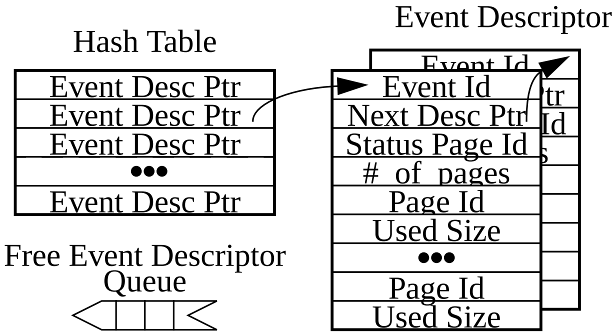


Figure 5. Event management in the ROB$_{IN}$-s.

When a data request is received from the ROB Controller, the ROB$_{IN}$ Application finds in the Hash Table the entry in the corresponding event descriptor chain (event identifier modulo Hash Table size) and then searches in the chain for the event descriptor itself. Finally, the ROB$_{IN}$ Application returns to the ROB Controller the number of pages and their identifiers, the offset and the size of data within each page. Event clear requests contain a list of events to remove from the event memory. For each event in the list of an event clear request, the ROB$_{IN}$ Application finds and removes the corresponding event descriptor from the Hash Table, frees event pages that contain event data and status information and appends the descriptor of the pages freed in the Free Event Descriptor Queue.

The Low Level Communication layer is based on a Client-Server message passing paradigm to exchange information between any two processors. This layer is independent of the ROB Complex application. A client submits a command to the server which processes it and returns a status information. Both the command and the status have variable length. The command contains a type, indicating the action to be performed by the server, a size and the command parameters. The status response contains a pointer to the corresponding command, its size and the status information itself. The Low Level Communication layer supports both synchronous and asynchronous commands. When a synchronous command is issued, the client blocks until the command is processed by the server. In the asynchronous mode of operation, the client can pipeline commands without having to wait for the status responses on the commands posted. Several types of command are pre-defined at the Low Level Communication layer itself and the server executes these commands without passing them to the upper layer. The commands that are not recognized by the Low Level Communication layer are delivered to the upper layer on the server side, as well as the corresponding status responses on the client side.

The ROB$_{IN}$ Hardware Abstraction layer hides implementation details of the ROB$_{IN}$-s. It allows to download the ROB$_{IN}$ software in the system memory of the ROB$_{IN}$ CPU, start and stop its execution, reset and halt the ROB$_{IN}$.

The Host Platform Abstraction layer hides implementation details of the ROB Controller such as operating system, host processor memory and PCI subsystem organization, their mappings, etc.

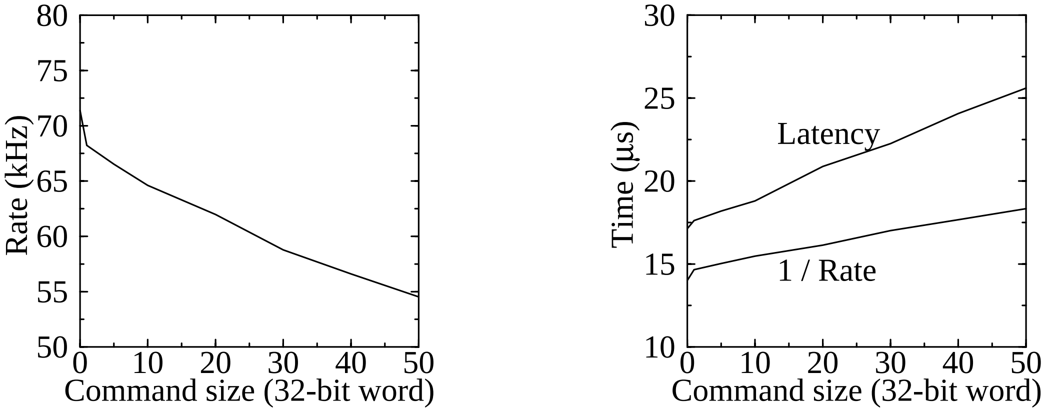
The ROB Complex software is implemented in C programming language. Apart from the three platforms described above and the Saclay ROB$_{IN}$ prototype, this software is also ported on a ROB Complex composed of a 400 MHz Pentium II PC running WindowsNT (ROB Controller) and a commercial PMC from TransTech equipped with a 40 MHz Sharc DSP (to act as a ROB$_{IN}$) [10].

B. ROB$_{IN}$ Performance Measurements

A stand-alone test program has been developed to evaluate the performance of the first version of the Saclay ROB$_{IN}$ prototype. The test program runs on the ROB Controller host processor and operates at the level of the ROB$_{IN}$ Application and API layers. With pre-defined rates it generates selective data requests and full event data requests, as well as event clear messages, places them in asynchronous Low Level Communication commands and posts the commands to the ROB$_{IN}$. When the ROB$_{IN}$ finishes the execution of a command it returns the corresponding status information to the test program that measures the service latency and rate.

B.1 Performance at the Low Level Communication level

In order to separate the Low Level Communication part from the rest of the overall ROB$_{IN}$ performance, the test program is able to generate variable size “dummy” commands that are recognized by the Low Level Communication layer and are immediately acknowledged by the ROB$_{IN}$ without passing them to the ROB$_{IN}$ Application layer. Figure 6 shows the “dummy” command service rate and latency as a function of command size. The selective data and the full event data requests are transported within four 32-bit word commands. At the Low Level Communication layer commands of that size can be exchanged at ~67 kHz rate with a service latency of about 17 µs. While the ROB$_{IN}$ CPU processes a command the ROB Controller can start the submission of the next one. At the Low Level Communication layer, for a four 32-bit word command, this parallelism gives a gain of ~2 µs (difference between the values of Latency≈17 µs and 1/Rate≈15 µs on Figure 6.b).



a) Command servicing rate

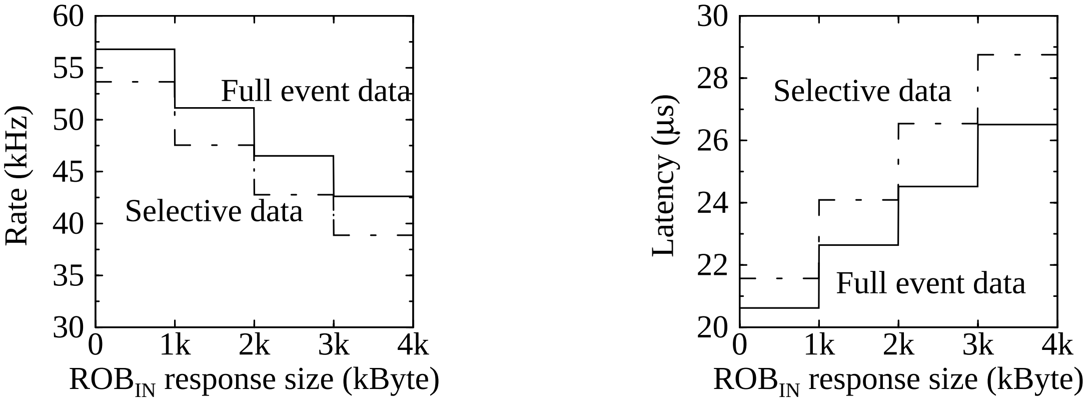
b) Command servicing time

Figure 6. ROB_{IN} performance at the Low Level Communication layer.

For event clear requests containing a list of 48 events, the command fits in fifty 32-bit words. At the Low Level Communication layer, such commands can be exchanged at a rate of ~55 kHz and their service latency is ~26 µs. A gain of about 9 µs is obtained by pipelining the commands by the parallelism of operation of the ROB Controller and the ROB_{IN} processors.

B.2 Data Request Servicing

In the current implementation of the ROB_{IN} Application, it is assumed that event data contains a fixed size Level 2 block. This might be the case for the electromagnetic calorimeter ROB_{IN}-s that represent nearly a half of all ROB_{IN}-s. As previously mentioned, the ROB_{IN} Application distinguishes two types of data requests. For full event data requests, a ROB_{IN} responds with all event data. For selective data requests only the Level 2 block of event data is returned. Figure 7 shows data request service rate and latency as a function of data response size.



a) Data request service rate

b) Data request service latency

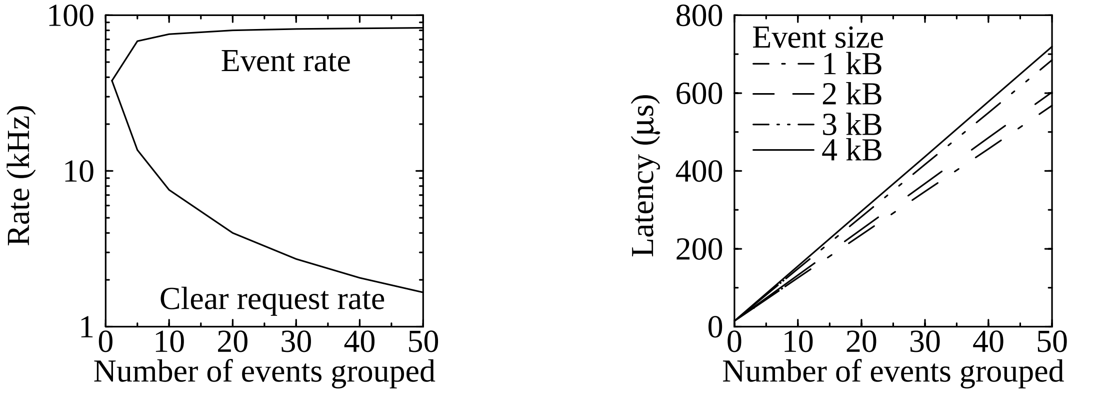
Figure 7. ROB_{IN} data request service performance.

For data requests no actual data movement from the ROB_{IN} memory to the ROB Controller memory is made. Instead only a list of page identifiers, which contain the response data, is returned. The ROB_{IN}'s event memory is organized in pages of 1 kByte. Depending on the Level 2 block size and total event size 1, 2, 3 or 4 page identifiers are returned to the ROB Controller for selective data and full event data requests. This results in the stair functions observed for both the service rate and the latency. The processing of selective data requests is somewhat longer than that of full event data requests because the boundaries of Level 2 blocks in a variable number of pages have to be determined. However, the difference of performance is marginal.

In the case of electromagnetic calorimeter the event size is ~1.8 kByte. The Level 2 block is of the order of 1 kByte. The full event data request service rate and latency are 51-52 kHz and ~23 µs respectively. Comparing these figures with the Low Level Communication performance, the overhead of the ROB$_{IN}$ Application for a data request service is evaluated to ~5 µs. The ROB$_{IN}$ is capable of servicing selective data requests at a rate of 53 kHz with a latency of ~22 µs. These measurements show that, for the electromagnetic calorimeter, a very simple data preprocessing can be done by the ROB$_{IN}$ CPU with no significant performance degradation.

B.3 Event Clear Request Servicing

Because there is no external input of event data in the first version of the Saclay ROB$_{IN}$, when the ROB$_{IN}$ Application processes event clear requests, for each deleted event a new one is generated and placed in the Hash Table. Figure 8 shows the service rate and latency for event clear requests versus the number of events grouped per request message.



a) Clear request service rate

b) Clear request service latency

Figure 8. ROB$_{IN}$ event clear request service performance.

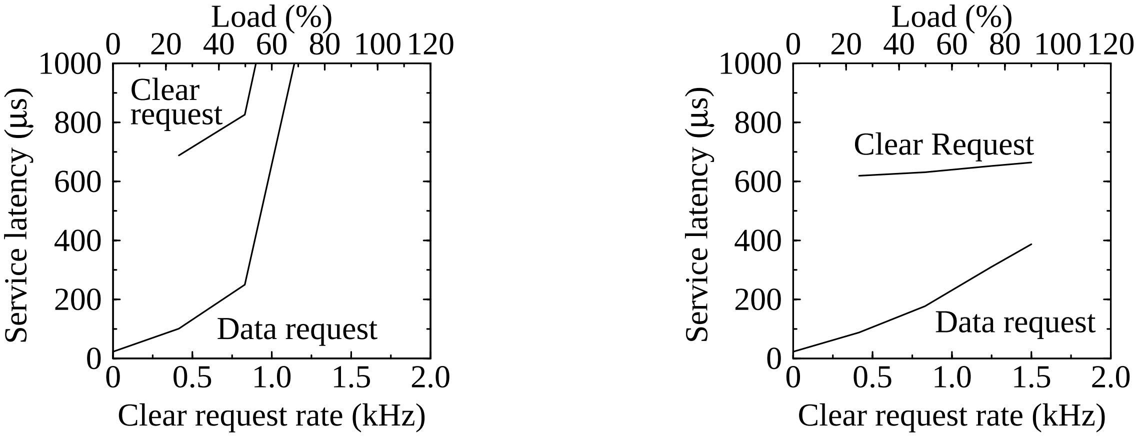
As before, the event memory page size is 1 kByte. The Clear request rate curve on Figure 8.a is plotted for 2 kByte events (roughly corresponding to the electromagnetic calorimeter ROB$_{IN}$-s). The Event rate curve is a product of the Clear request rate and the number of events grouped. It corresponds to a sustainable Level 1 trigger rate. For grouping factors above twenty, the Event Rate is almost constant and approaches 83 kHz for 50 events in the event clear request.

The event clear request service latency is a linear function of the event grouping factor. Its slope depends on event size, or more precisely, on the number of pages that events occupy in the event memory of the ROB$_{IN}$-s. Taking into account the Low Level Communication performance, one can estimate the time spent in the ROB$_{IN}$ Application to remove an event from the Hash Table, free event memory pages, than generate a new event and append it to the Hash Table. In the case of 2 kByte events this time amounts to ~12 µs.

B.4 Overall ROB$_{IN}$ Performance

In the mixed mode of operation, when the ROB$_{IN}$ services both data and event clear requests, its performance has been measured for run parameters close to those of the electromagnetic calorimeter ROB$_{IN}$-s, i.e. event size of 2 kBytes, selective data and full event data request rates of 3 and 2 kHz respectively. The grouping factor for event clear requests is 50 events.

Figure 9 shows the average service latency of requests as a function of event clear request rate. Two cases are considered. In the first case, the ROB Controller posts all received event clear requests to the ROB_{IN}-s without waiting for the previous clear request to be serviced. For 50 events grouped in an event clear request, the service latency is of the order of 600 µs. If a data request is placed behind several event clear requests, its processing will be delayed until the ROB_{IN} finishes with all event clear requests in front of it. Indeed, Figure 9.a shows that the event clear requests strongly influence data request service latency which grew from a modest ~20 µs to a value well beyond 1 ms.



a) No flow control on clear requests

b) One clear request in the pipeline

Figure 9. ROB_{IN} performance in mixed data and event clear request mode of operation

In the second case, the ROB Controller posts only one event clear request at a time to the ROB_{IN}-s. Therefore, in front of any data request there cannot be more than one event clear request that delays the data request by at most ~600 µs (Figure 9.b). These measurements show that if the data request service latency is an issue (e.g. for Level 2 trigger data collection), flow control could be applied to the event clear request messages. This can be done either by the ROB Controller that enqueues the event clear messages while the ROB_{IN}-s process the previous ones, or by the ROB_{IN}-s assigning a higher priority to service data request messages than requests to clear some events.

C. ROB Complex Performance Measurements

The ROB Complex prototypes previously described have been integrated in the ATLAS Level 2 Trigger ATM Testbed [11]. The configuration of the testbed is depicted on Figure 10. An ATM switch connects a monitor, 2 supervisors, 14 destination processors and a ROB Complex. The size of the testbed system and the working parameters of the nodes are chosen so that the ROB Complex can be saturated.

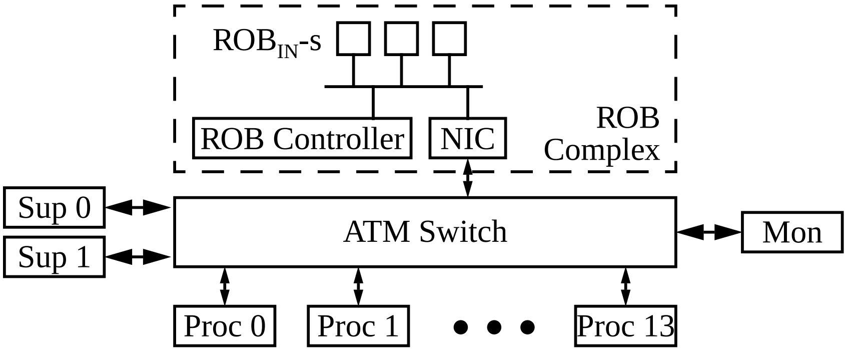


Figure 10. ATM Testbed configuration for ROB Complex performance measurements

For each event generated, the supervisor selects a processor within the farm and communicates an allocation message to it. The destination processors run a very simple single step algorithm that consists in requesting RoI data or full event data from the ROB Complex. Neither reformatting of received data, nor an emulation of some selection algorithm is performed. Instead, as soon as the requested data arrives, a processor sends a random trigger decision to the corresponding supervisor. The latter accumulates a block of 32 decisions before sending them to the ROB Complex.

The ROB Complex task consists in forwarding data requests to a selected group of $\mathrm{ROB_{IN}}$-s, forming event fragments from data collected from relevant $\mathrm{ROB_{IN}}$-s and sending them to their destination processor, receiving decision messages from supervisors and broadcasting them to all $\mathrm{ROB_{IN}}$-s, measuring its own performance and periodically communicating it to the monitor node. The following statistic is gathered on-line: selective data request service latency and rate, full event data request service latency and rate, decision request rate.

The $\mathrm{ROB_{IN}}$ Application previously described, which performs event management using paged event memory and a hash table, has been developed recently. Not all three platforms could be tested with this version of the $\mathrm{ROB_{IN}}$ software because the testbed has been moved from Saclay to CERN. Unless otherwise stated, in what follows, the ROB Complex performance measurements correspond to a simplified event management algorithm in $\mathrm{ROB_{IN}}$-s, which is based on a large table with entries pointing to contiguous memory locations of events. The handling of event clear requests is simplified as well: the $\mathrm{ROB_{IN}}$ just acknowledges them.

The results of performance measurements for ROB Complex on PC/Linux platform is shown on Figure 11. It presents the maximum rate of data requests that can be serviced by the ROB Complex versus the response size which can be composed of data from 1, 2 or 3 $\mathrm{ROB_{IN}}$-s.

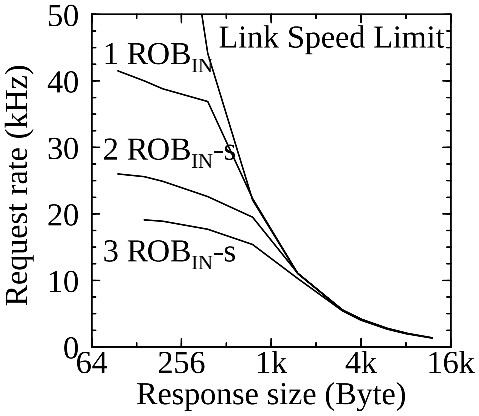


Figure 11. Data request rate for PC/Linux ROB Complex.

The limitation due to the saturation of the 155 Mbit/s output link is reached for response messages larger than 1.5 kByte independently on the number of $\mathrm{ROB_{IN}}$-s in the ROB Complex. For small response messages where the performance is determined by the ROB Complex software overhead, the following formula can be derived (for 400 MHz Pentium II):

$$\text{Request rate (kHz)} = 1000 / (11.5\ \mu\text{s} + 13.5\ \mu\text{s x number\_of\_ROB}_{\text{IN}}\text{-s}) \qquad (1)$$

Drawing final conclusions would require measurements with more $\mathrm{ROB_{IN}}$-s to be done, but according to equation (1), the rate requirements estimated in Table 1 seems within reach. However, the data request rate which can be sustained by a ROB Complex depends on the complexity of the ROB Controller and the $\mathrm{ROB_{IN}}$ applications. Table 3 compares the performance of a ROB Complex for three cases: when the $\mathrm{ROB_{IN}}$ functionality is emulated by software, when the simplified event management scheme based on a large table is used in the $\mathrm{ROB_{IN}}$, and when a more realistic paged event memory management based on a hashing mechanism is used. In all cases the ROB Complex reply data size is 128 Bytes.

Table 3. ROB Complex performance as a function of ROB_{IN} application complexity

ROB_{IN} type	Max data request rate (kHz)		
	1 ROB_{IN}	2 ROB_{IN}-s	3 ROB_{IN}-s
Software emulation of ROB_{IN}	62	50	42
ROB_{IN} with a large table	40	26	19
ROB_{IN} with a hash table	28	25	19

ROB Complex performance comparison for CompactPCI, VME and Linux platforms is presented in Table 4. As before the ROB Complex reply data size is 128 Bytes. The request service latency values are given for a very low load on the ROB Complex, when data request and supervisor decision messages arrive respectively at ~1 kHz and ~31 Hz rate.

Table 4. ROB Complex performance comparison for the three platforms

Platform	Max request service rate (kHz)			Request service latency (µs)		
	1 ROB_{IN}	2 ROB_{IN}-s	3 ROB_{IN}-s	1 ROB_{IN}	2 ROB_{IN}-s	3 ROB_{IN}-s
CPCI/Lynx (200 MHz PPC)	20	16	13	46	52	61
VME/Lynx (300 MHz PPC)	35	25	19	37	43	50
PC/Linux (400 MHz P‖)	40	26	19	35	50	58

It is expected (and observed) that for the same type of the ROB Controller's host processor and operating system the performance increases with the clock speed of the host. This gives good hope that the use of faster processors will counterbalance the complication of the ROB Complex software as its functionality closer matches the requirements for ATLAS. The tail distribution of the request service latency when 128 bytes of data are requested from one ROB_{IN} is shown on Figure 12.

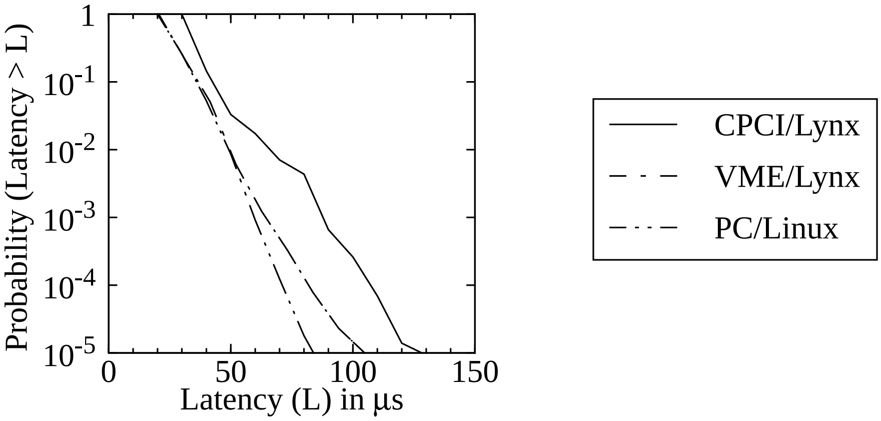


Figure 12. Data request service latency for the three platforms.

The ROB Complex operates at 50% load i.e. the data requests arrive at 10, 17 and 20 kHz for CompactPCI, VME and PC platforms respectively. The average data request latency for the three platforms equals 46, 38 and 38 µs and none of them exhibit long tails.

More results on the ROB Complex performance measurements can be found on the Saclay ATLAS Trigger Web page [12].

IV. Summary

The proposed organization of the ROB Complex allows to achieve a high degree of modularity: input of detector data streams and event memory management, data collection protocol, run control and monitoring are logically separated functions that can be separated physically as well. The ROB Complex architecture suggests the use of commercially available components based on widespread industrial standards. At present CompactPCI, with the diversity of its mechanical and electrical support, seems to be a good candidate technology to build various configurations of ROB Complexes able to satisfy the different needs of ATLAS detector subsystems in terms of data rates and bandwidth. ROB Complexes composed of one to ~20 read-out buffers can be implemented using the same CompactPCI crate mechanics. The actual grouping factors for each of the detector subsystems have been estimated with paper modelling. In the particular organization of the ATLAS T/DAQ read-out system studied, about 80 CompactPCI crates could house the totality of the read-out buffers grouped into ~300 ROB Complexes that are connected to a common network for Level 2 and event building with 80 Mbyte/s links.

Prototyping work is ongoing to validate the concept of the ROB Complex. A ROB Complex with three $\mathrm{ROB_{IN}}$-s has been integrated in the ATLAS Level 2 Trigger ATM testbed. Though the first version of the $\mathrm{ROB_{IN}}$ card has reduced functionality, it remains sufficient to study communication aspects within the ROB Complex and with the rest of the High-Level Triggers and DAQ system. The performance measurements give encouraging results. The possibility of some preprocessing for electromagnetic calorimeter data by the $\mathrm{ROB_{IN}}$ has been shown. It is likely that data request rate requirements can be met by the $\mathrm{ROB_{IN}}$-s and the ROB Controller. The next version of the $\mathrm{ROB_{IN}}$ with its input data port and the associated memory management logic is now under test. This will allow building and testing larger ROB Complexes with functionality much closer to the needs of a final design for ATLAS.

V. References

[1] J. Bystricky et al., “A Sequential Processing Strategy for the ATLAS Event Selection”, IEEE Transactions on Nuclear Science, vol. 44, 3 June 1997

[2] J. Bystricky, J. Vermeulen, “Paper Modelling of the ATLAS LVL2 Trigger System”, ATL-COM-DAQ-2000-022, March 2000

[3] PICMG, CompactPCI Specification, Version 2.1, September 2, 1997

[4] Gespac Innovative Embedded Solutions, http://www.gespac.com

[5] TreNew Electronic, http://www.trenew.com

[6] RIOC 4063/4064, PowerPC based RISC I/O Board, User’s Manual, version 0.1, Creative Electronic System S.A., October 1997

[7] RIO2 8062, PowerPC based RISC I/O Board, User’s Manual, version 1.0, Creative Electronic System S.A., October 1997

[8] PEB 6406/6407, VME PMC Extension Board, User’s Manual, version 1.0, Creative Electronic System S.A., December 1997

[9] IDT77211 NICStAR, User’s Manual, Version 1.0, Integrated Device Technology INC., February 26, 1997

[10] ASP-C2, SHARC PMC with DRAM, http://www.transtech-dsp.co.uk

[11] D. Calvet et al., “The ATLAS High Level Trigger ATM Testbed”, In Proc. of the 11th IEEE Real Time conference, Santa Fe, New Mexico, USA, 14-18 June 1999

[12] The Saclay ATLAS L2 Trigger group, http://www-dapnia.cea.fr/Phys/Sei/exp/ATLAS/pres/pres.html