The Read-Out Crate in ATLAS DAQ/EF prototype -1

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]*, G. Lehmann*[1,3] *, 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*[3,6,11]*, 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] National Laboratory for High Energy Physics (KEK), Japan
[9] Now at Lightning Instrumentation S.A., Lausanne, Switzerland
[10] Now at EDF, Grenoble, France
[11] *Corresponding author*

Abstract

A prototyping project has been undertaken by the ATLAS DAQ and Event Filter group. The aim was to design and implement a fully functional vertical slice of the ATLAS DAQ and Event Filter with maximum use of commercial components.

The Read-Out Crate is a component within the vertical slice whose principle functionality is to receive, buffer and forward detector data to the Level 2 Trigger and Event Builder systems.

The high-level design and initial implementation of the Read-Out Crate will be presented, putting emphasis on the concept of a ROBIN. The latter implements the functionality of receiving and buffering incoming detector data at up to 160 Mbyte/s.

The initial implementation is based on commercial components: VMEbus PowerPC and Pentium based single board computers; one or more ROBIN PMCs; an additional intra-crate bus supporting broadcast; the LynxOS real-time operating systems.

Results will be presented which summarise the performance of a commercially available PowerPC based PMC. In addition, results on the performance of the Read-Out Crate, measured in the context of the ongoing DAQ/EF -1 project will be presented.

Keywords: ATLAS, DAQ, ROB, dataflow, PMC, FPGA

1 Introduction

The ATLAS [1] Data Acquisition and Event Filter system (DAQ/EF) has been described elsewhere [2] and the Dataflow subsystem of DAQ/EF is described in these proceedings [3]. In the following sections emphasis will be put on the DAQ component of the Dataflow which receives data from the detector systems, the Read-Out Crate (ROC), and its implementation within the DAQ/EF-1 prototype [4].

1.1 The Read-Out Crate

Each ROC is responsible for: collecting data from various Read-Out Links (ROLs) which are connected to Level-1 (LVL1) derandomizers; buffering events during the Level-2 (LVL2) latency and distributing event fragments to the Event Builder and to the LVL2 system. Major design components of the ROC are the Local DAQ (LDAQ), which is responsible for local control and monitoring, and the DAQ-unit, which implements the flow of data within the ROC. A DAQ-Unit

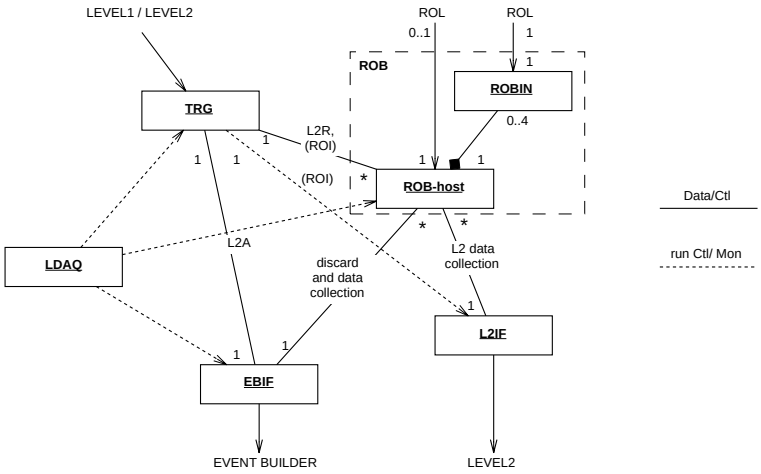

Figure 1: The ROC logical model

is composed of a number of I/O modules (IOM), which come in several flavours, depending on the type of external I/O channel to which they are connected: the TRG receives LVL1 and LVL2 signals to collect (respectively Region-Of-Interest, i.e. ROI and L2A signals) or reject (L2R) events; the ROB collects event fragments from ROLs; the EBIF sends event fragments to the Event Builder; the L2IF sends data to LVL2. The ROC logical module schema is shown in Fig.1, where the inter-connections between the various components are also shown.

Each IOM is also associated to one or more I/O channels internal to the ROC and each I/O channel has an associated Task. In order to minimize overheads due to the operating system, all the Tasks belonging to an IOM are scheduled within a single process and activated by polling conditions. All components within the ROC communicate via a well defined message protocol and exchange data via links, depending on their deployment [5]: VMEbus and PVIC for intra-crate communications between different Single Board Computers (SBC); PCI and the CPU bus when the modules are implemented into the same SBC or distributed on the SBC and PMCs; TCP/IP based message passing in inter-crate communication.

2 The Read-out Buffer (ROB)

The ROB has three main tasks: it receives and buffers Read-Out Driver (ROD) fragments from a ROL; it receives and processes the L2R and ROI messages from the TRG; it performs the data collection, based on request from the EBIF. The ROB must be able to receive and buffer event fragments of $\sim 1kB$ at a frequency of up to $100kHz$. The system bus (PCI) of todays commercial VMEbus Single Board Computers (SBC) does not provide the bandwidth required by many ROLs. Therefore, to avoid moving the event fragments over the system bus, the idea of using a specialized board, called ROBIN, has been developed. The ROBIN is an add-on PMC card which, by a combination of HW and SW, receives and buffers event fragments locally, thus avoiding moving the data over the PCI bus at $100kHz$. A ROB may host one or more ROBINs and use an Event Manager API to locate, delete and retrieve events from the ROBINs. The Event Manager is deployed over the ROB-host (on a SBC) and the ROBIN and implies simple message passing over the system bus. The complete data collection mechanism works as follows. When a L2A is received the EBIF requests from each ROB the event fragment parameters (address, lenght) relative to a particular event ID (L1ID). The ROB header is now built The ROB then returns pointers and fragment lenghts to the EBIF for every fragment coming from the ROBINs. Then event fragments

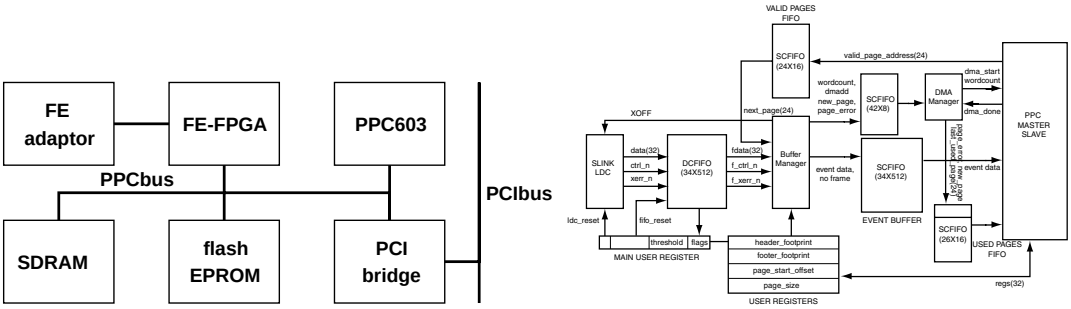

Figure 2: The MFCC block diagram and the logic implemented in the FE-FPGA.

and headers are read (via a DMA chained transfer) on VMEbus by the EBIF. The transferred event data are then removed from the internal buffers of the ROBIN. When a L2R is received, the events pointed by the received message are removed from the ROBIN's buffer.

3 The MFCC based ROBIN

The PMC card used in these studies is a commercial intelligent I/O processor from CES, the MFCC8441 [7]. It contains a PCI interface, a computing core based on a PowerPC processor, a Front-End application specific Field Programmable Gate Array (FE-FPGA) and up to 64 MB of global memory. The FE-FPGA handles the receiving and buffering of data from the ROL (SLINK LDC [8]), while the processor handles the Event Management and FPGA control. The buffering memory is organized in fixed size pages, and events can span one or more pages. The SLINK LDC is a mezzanine board, mounted on a VME carrier (S2P2), connected to the MFCC via a backplane connector.

3.1 The ROBIN implementation

The standard DAQ software, single process and based on a task scheduler, is running on the MFCC on top of the Lynx OS operating system, without interrupts service, without using dedicated device drivers, thus minimizing operating system calls. Event fragments are written into the ROBIN by the LDC synchronously, at a maximum bandwidth of $32bit \times 40MHz$. They are synchronized and buffered in a 256 words deep dual-clocked FIFO (DCFIFO). Flow control is applied to the LDC by checking the fill level of this DCFIFO. In Fig.2 the logic diagram of the ROBIN and FE-FPGA logic is shown. Communication between the FPGA and the processor is done via two FIFOs: the $ValidPagesFIFO$ (16 deep) which contains short page addresses of on-board SDRAM memory to which the event fragments have to be written by the FE-FPGA; the $UsedPagesFIFO$ (16 deep), which contains addresses, lenghts and status flags of event fragments which the FE-FPGA has already trasferred to global memory. The $BufferManager$ state machine is responsible for reading the DCFIFO, measure the fragment size and format data into an output FIFO (SCFIFO), which decouples the $BufferManager$ from the $DmaManager$. The $DmaManager$ is a second state machine which implements a PPC bus master, it writes bursts of event fragments into valid memory pages and fills the Used Pages FIFO. Communication between the two state machines is done via a $DmaFifo$, 8 events deep, which stores extended page addresses, DMA transfer sizes and status flags. User registers, accessible via the PPC bus, contain various parameters useful to tune the ROBIN to the subdetector to which it is dedicated, like page size, maximum number of pages per event, expected SLINK packet header and footer footprints. The input of the DCFIFO works at the SLINK frequency ($\leq 40MHz$), while the rest of the logic works at the PPC

bus frequency of $66 MHz$.

4 Measurements

The ROC baseline implementation [9] has been done deploying the LDAQ and each IOM on VME Single Board Computers (SBC), more specifically the CES RIO8062. A single ROB was used during the following ROC performance measurements.

4.1 ROB performance

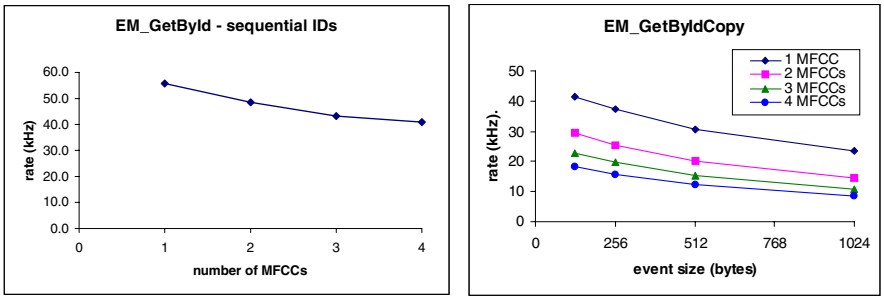

Figure 3: Event location (first plot) and event retrieval (second plot) rates vs number of MFCCs and event size (one page per event is chosen)

In Fig.3 the rate at which event fragments can be located and retrieved by the ROB-host is shown. In the first plot a dependancy on the number of MFCCs is shown, because no broadcast mechanism, as favoured by the message passing system, exists on PCI. On the second plot the event retrieval rate versus the event fragment size is shown decreasing as expected with large fragment sizes. A transfer bandwidth of 50 MB/s is obtained between the ROB and the ROBINs, mainly limited by the CPU/RAM subsystem of the RIO and MFCC. The MFCC system bus in the best condition has a sustained bandwidth of 145 MB/s of data move from SLINK to MFCC memory. In the following figure (Fig.4), the event deletion rate is shown without and with SLINK

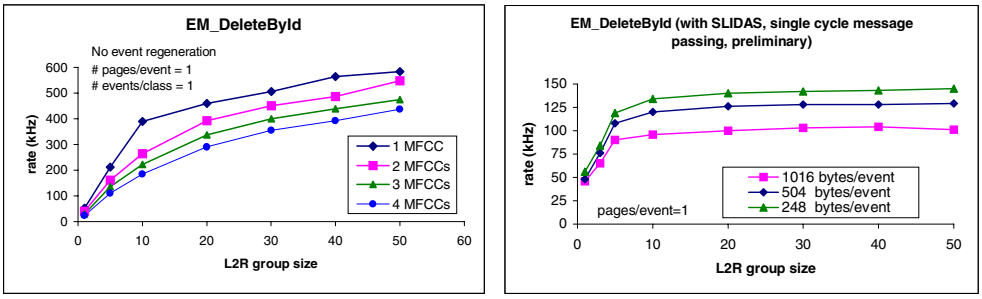

Figure 4: Event deletion vs L2R group size without concurrent data input (first plot) and during data buffering (second plot), for different event fragment sizes

input of new events. In the first plot the rate increases with the number of event delete requests per message (L2R group size) and the rate is dominated by the message traffic on PCI. The rate shown in the second plot is dominated instead by the input event fragment traffic, and flattens rather soon for group sizes of 10 events. A dependance on other experimental or Event Manager parameters

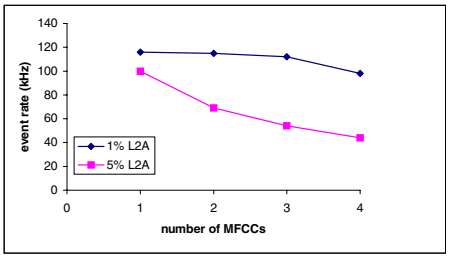

Figure 5: ROC rate vs number of ROBINs, for a L2A/ROI of $1\%/0\%$ and $5\%/0\%$, and internally generated data fragments, for one ROB

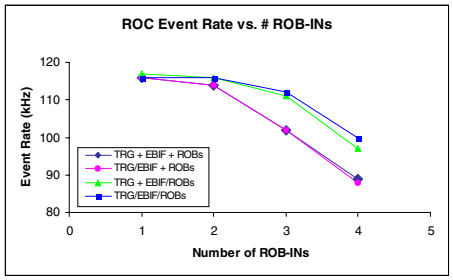

Figure 6: ROC rate vs number of ROBINs and Slink Input, of baseline implementation, with TRG/EBIF collapse, EBIF/ROB collapse and full collapse of DAQ-Unit

like the L2A/ROI to input rate ratios, number of pages per event, fragment size, has also been observed.

4.2 ROC performance

Results shown up to now rely on test programs running on the ROB-host, independently from the TRG and EBIF IOM, to better understand the ROB basic functionality and performances. Other measurements have been done to understand the performance of a ROB within the context of a ROC. In Fig.5 the ROC event rate through the DAQ-Unit is shown. The rate in this case is dominated by the PCI traffic between ROB and ROBINs (currently done in single cycles). The obtained event rate of $\sim 120 KHz$ is limited by the performance of the DAQ software running on the MFCC.

4.3 Collapsing of IO Modules into a Single Board Computer

A further flexibility can be achieved by minimizing the data movement and message passing, through collapsing of ROC components into the same SBC. By collapsing, the gain in ROC bandwidth usage is paid at the cost of increasing the number of Trigger, Event Builder or LVL2 links on the ROB. The ROC performance measurement of different mergings of IOMs, up to the full collapse of the DAQ-Unit, is shown in Fig.6, The biggest gain is obtained when the data collection is internal to a SBC (no use of the VMEbus).

5 Conclusions

The results shown, compared with the expected maximum LVL1 accept rate of $75-100 KHz$ and L2A rate of a few kHz are encouraging. The ROC prototype has been designed with a high level of modularity, adaptable to different sub-detector needs.

If the final choice of ROC components should be done now, most probably commercial versions would be chosen. Software and hardware maintainance and support are relevant in a large system like the ATLAS DAQ/EF.

References

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

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 G.Lehmann et al., “The Dataflow system of the Atlas DAQ/EF-1 prototype project”, these proceedings.

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

5 et al., “Intra- and Inter-IOM communications summary”.http://atddoc.cern.ch/Atlas/Notes/126/Note126-1.html

6 G. Ambrosini et al, ”DAQ-Unit performance summary” http://atddoc.cern.ch/Atlas/TPR.

7 http://www.ces.ch

8 http://www.cern.ch/HSI/s-link/

9 G. Ambrosini et al, ”ROB performance Milestone document”http://atddoc.cern.ch/Atlas/TPR.