PCI-based Readout Receiver Card in the ALICE DAQ System

W. Carena[(1)], F. Carena[(1)], P. Csató[(2)], E. Dénes[(1)], R. Divià[(1)], T. Kiss[(2)], J.C. Marin[(1)], K. Schossmaier[(1)], Cs. Soós[(1)], J. Sulyán[(2)], A. Vascotto[(1)], P. Vande Vyvre[(1)]
(for the ALICE collaboration)

[(1)] CERN, 1211 Geneva 23, Switzerland
[(2)] KFKI-RMKI, Budapest, Hungary

Abstract

The PCI-based Readout Receiver Card (PRORC) is the primary interface between the detector data link (an optical device called DDL) and the front-end computers (PC running Linux) of the ALICE data acquisition system. This document describes the prototype architecture of the PRORC hardware and firmware, and of the PC software. The board contains a PCI interface circuit and an FPGA. The firmware in the FPGA is responsible for all the concurrent activities of the board, such as reading the DDL and controlling the DMA. The co-operation between the firmware and the PC software allows autonomous data transfer into the PC memory with little CPU assistance. The system achieves a sustained transfer rate of 100 MB/s, meeting the design specifications and the ALICE requirements.

I. Introduction

The Detector Data Link (DDL) is an integral part of the ALICE [1] data acquisition system (see Figure 1). The DDL will be installed between the detectors and the front-end computers called Local Data Collectors (LDC). According to the experiment requirements, the DDL provides 100 MB/s sustained bandwidth. The DDL interfaces are specified in [2].

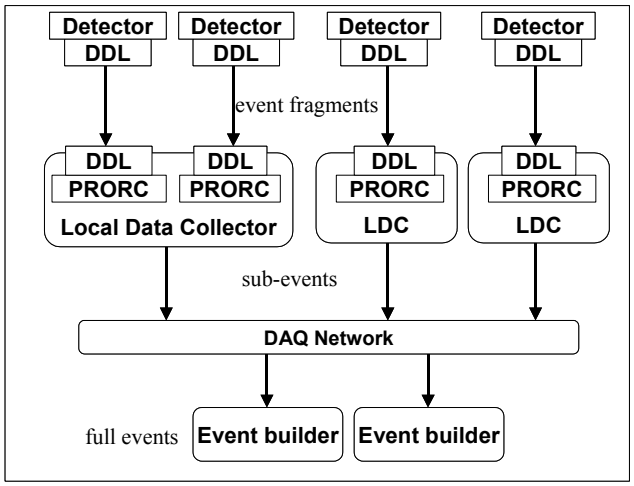

Figure 1: PRORC in the ALICE data acquisition system

The Readout Receiver Card (RORC) is the interface between the DDL and the local I/O bus of the front-end computers. At present, the PCI local bus exists in most computers and provides sufficient bandwidth for the DDL. The theoretical maximum speed of the 32 bit/33 MHz version is 132 MB/s, while using the 64 bit/66 MHz version one could in principle achieve 528 MB/s. Therefore, we have decided to implement a PCI-based RORC, called PRORC.

II. Hardware and Firmware

The first version of the PRORC was designed as a 32 bit/33 MHz PCI master card (see Figure 2). The PCI interface was realized by a commercial ASIC of type AMCC S5935 [3], which provides the necessary functions to perform PCI bus mastering. Using the master mode operation, the card is able to transfer data from the DDL directly into the PC memory. Consequently, there is no need for on-board memory, thus reducing complexity of the board, and lowering cost. The custom logic functions are implemented in a programmable logic device of type ALTERA EP20K200.

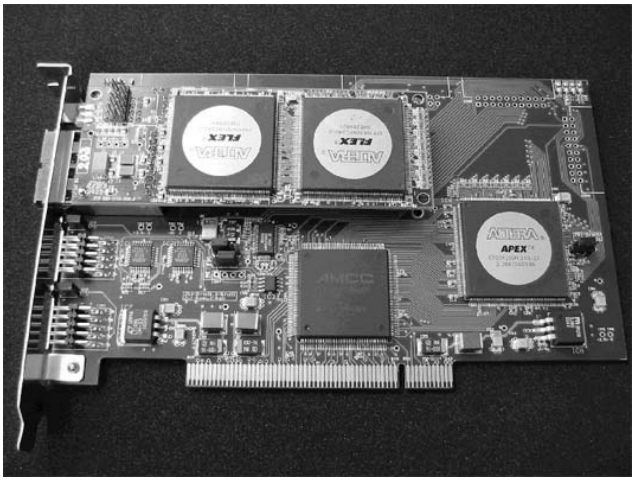

Figure 2: The PRORC card with the DDL piggyback board

The key feature of the AMCC PCI interface is to connect the devices on the AMCC local bus to the PCI bus. The integrated circuit consists of the interfaces to the buses, several control registers, mailboxes, and small FIFO buffers (see Figure 3). There are three sets of registers to control the read and write DMA operations on the PCI bus and the general behaviour of the interface. The mailboxes are special registers accessible either from the PCI bus or from the AMCC bus.

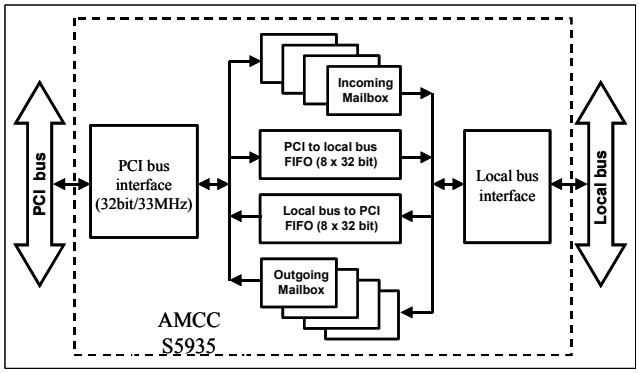

Figure 3: PCI interface circuit

The firmware in the ALTERA can be divided into three parts according to the functions they provide (see Figure 4).

- The AMCC interface handles the signals between the different firmware parts and the AMCC chip, and manages the mailboxes available in the AMCC PCI interface. The PRORC can be controlled by the PC software using these mailboxes. Commands are passed through the incoming mailboxes and the firmware is notified by means of a hardware interrupt. When the commands are executed, their result (e.g. the status of the PRORC) is placed in the outgoing mailboxes.
- The control part receives the commands and interprets them in order to carry out the required actions. Memory management and DMA engines are also implemented in this block. There are two different channels for reading and writing the link, namely the forward channel, which conveys data from the detectors to the LDC computers, and the backward channel, which uploads data into the detector interface.
- The DDL interface performs the link operations by connecting the firmware to the DDL through FIFO memories. These small buffers ensure the synchronization between the different clock domains of the PCI and the link. This interface contains a pattern generator as well, which can be used for testing purposes.

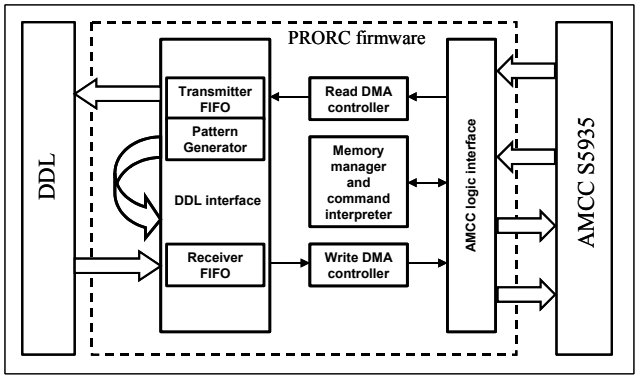

Figure 4: Firmware architecture

III. OPERATION

The card has been designed to be compatible with the ALICE data acquisition software (DATE) [4] running under Linux. Furthermore, the way it works makes it possible to use it with other kinds of data acquisition software. There is one important requirement, however: the software must be able to allocate memory regions in physical space and to provide their physical address. The current DATE version includes a package, called *physmem*, which fulfils these requirements for Linux.

A. Data acquisition

The DDL transfers the event fragments from a specific part of the detector as blocks separated by a DDL-generated status word called Data Transmission Status Word (DTSTW). The fragments are temporarily stored in the Receiver FIFO, before being sent via a DMA write into the PC memory.

There are two activities involved in the data acquisition (see Figure 5). One is the allocation of memory pages made by the readout software running on the front-end computer. The other is the data transfer controlled by the DMA engine, implemented in the firmware of the PRORC. The communication between the software and the firmware is based on two FIFOs, respectively called Free FIFO and Ready FIFO. It is worth noting that the Free FIFO is located in the firmware, whilst the ready FIFO resides in the PC memory.

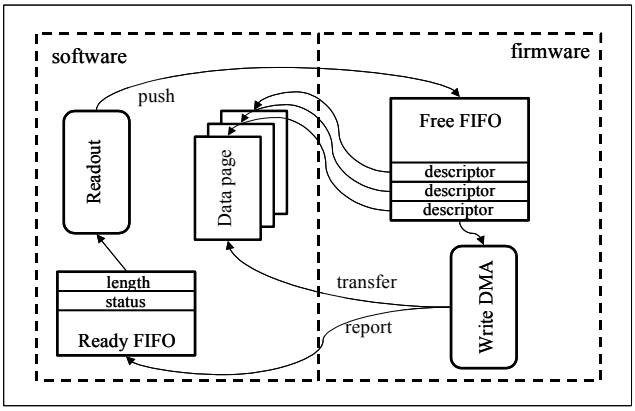

Figure 5: Software and firmware operation

In order to carry out the DMA, the PC software has to allocate a memory page of a given length, where the data will be transferred. In addition, the PC software must reserve an entry in the Ready FIFO. The physical base address of the memory page, the page size and the index of the Ready FIFO entry must be passed through the PCI into the Free FIFO. In this way, the firmware can maintain a list of the free pages in the Free FIFO.

When the transfer is enabled and the Receiver FIFO is not empty, the DMA engine takes one entry from the Free FIFO and starts the transfer. The target address in the PCI interface circuit is set to the base address of the next available free page. The data is transferred to the target memory location until the memory page is full or the fragment is finished. If

one of these conditions is met, the firmware terminates the DMA and the engine transfers the actual data length and the status into the corresponding Ready FIFO entry.

The status field of the Ready FIFO entry has an additional role. Since the incoming fragment can be larger than the allocated page size, the fragment must be split into two or more pages. In this case, the PRORC uses the status field to indicate the continuation by setting it to zero. The last page of a fragment is marked by a DTSTW.

There are several advantages in using a readout based on this approach. Since the firmware can store many page descriptors in the Free FIFO, the DMA engine can proceed autonomously, without assistance from the PC software. It is also possible for several PRORCs to work in parallel. Since the CPU is not directly involved in the data transfer, other processes can take over and continue the data processing. Moreover, a good fraction of the PCI bandwidth can be saved by avoiding polls on the PCI. Using the Ready FIFO, the software can poll the memory instead of the bus.

B. Data download

Some detectors may require configuration data to be downloaded via the DDL. For this purpose the PRORC supports block mode download.

The transfer is started by setting the base address and the length of the block to be downloaded, and the address of the memory location where the status will be put. Like the DMA write, the DMA read is also autonomous. While the PRORC is managing the transfer, the host CPU is free for other activities. When the PRORC finishes, it will copy the status to the given memory address where the software can read it. In case of a successful download, the status will contain the size of the block.

IV. PERFORMANCE TESTS

In order to measure the performance of the PRORC, we built a test bed providing realistic test conditions. Since we could not connect the DDL to the real detectors, we used an emulator made in-house, called front-end emulator, which is able to generate formatted event fragments complying with the DDL protocol. On the receiving side, we used an Intel™ 800 MHz PC with two PRORC cards on one PCI bus and one Gigabit Ethernet card on another PCI bus. In these tests, we used the Linux operating system and the full version of the DATE software. In each test, we measured the transfer speed and the event rate using different fragment sizes.

A. Single PRORC without event builder

In the first test, we were looking for the raw performance of the system, including the PRORC. The event fragments were produced by the front-end emulators and sent across the DDL. In addition, some consistency check had been applied on the fragments, before they were discarded. The results are shown in Figure 6.

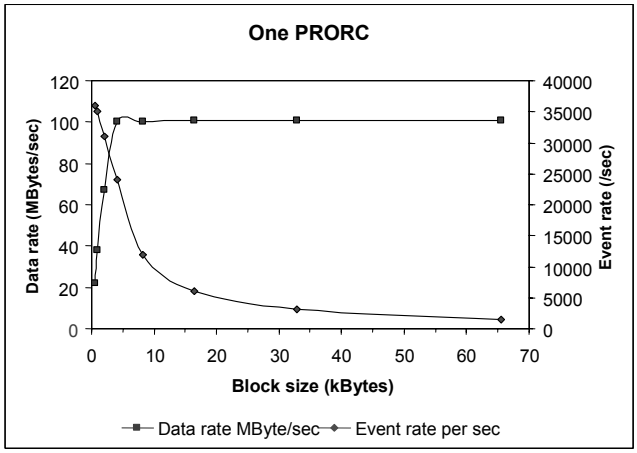

Figure 6: Performance results of one PRORC card

Using zero length blocks, we could measure the software overhead, which was 28 µs. This value corresponds to 35 000 events/s. By increasing the fragment size, the transfer rate was increasing steadily, until it reached the 100 MB/s maximum rate. The measurements show that we reach the full DDL bandwidth for fragments as low as 5 kB.

B. Two PRORC cards without event builder

The goal of the second test was to see how the PRORC cards share the PCI bus. Each PRORC card was connected to a DDL, which was fed by its own front-end emulator. In order to keep the front-end cards synchronized, each event contained an identification number.

The shape of the bandwidth vs. block size curve is similar to the one measured in the single card configuration (see Figure 7).

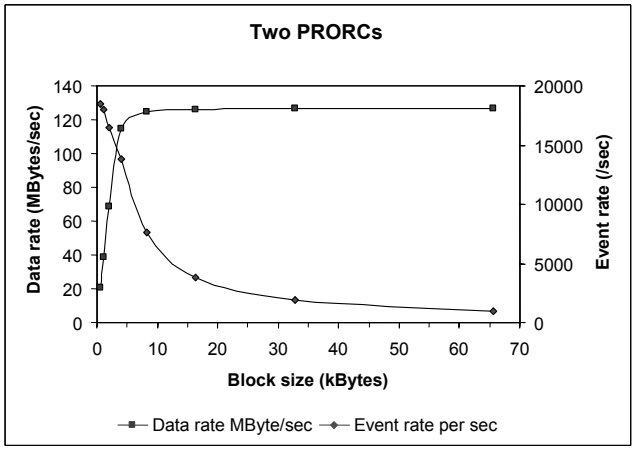

Figure 7: Performance results of two PRORC cards

By using two cards, however, the overall bandwidth is higher. When large fragments are sent, the aggregate transfer speed is saturated at 127 MB/s, near the theoretical speed of the 32 bit/33 MHz PCI local bus.

C. Single PRORC with event builder

The next test was made to measure the performance and the long-term stability of the complete system including the event builder. In this case, the LDC performed some data formatting, and then it sent the sub-events to the event builder through the Gigabit Ethernet interface. To reduce the competition on the PCI bus, the network card and the PRORC were placed on separate PCI buses.

Again, the same fragment size dependency was observed during the test (see Figure 8) with similar behaviour of the system. The transfer speed dropped to 70 MB/s, which was determined by the Gigabit Ethernet interface card. During the longest run lasting almost one week (stopped by the operator), the system demonstrated good stability.

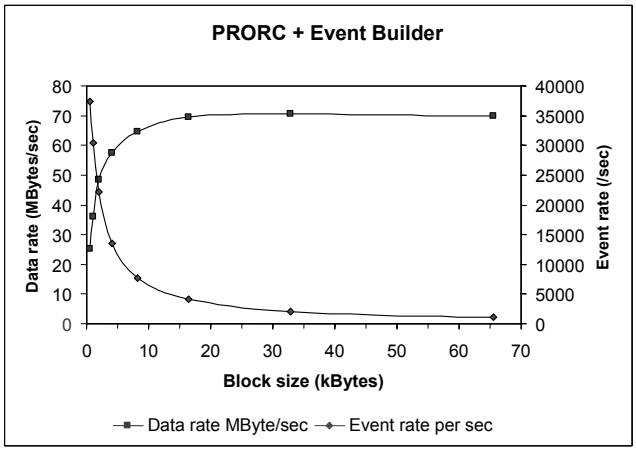

Figure 8: Performance results of the PRORC and the event builder

V. Applications

The PRORC card is already being used by some detector groups that are advanced enough to test their electronics. The DDL and the PRORC provide the required functions for testing the readout of all the ALICE detectors. For a quick start, there is a software package available [5], which covers the basic operations.

A sector of the ALICE Time Projection Chamber (TPC) [6] is going to be tested using the DDL. The data from the front-end electronics will be collected by a module called Readout Control Unit (RCU) and then will be sent to the PC through the DDL and the PRORC. The integration of the DDL and PRORC with the first version of the RCU prototype has already been done.

The Silicon Drift (SD) detector of the ALICE Inner Tracking System (ITS) [7] uses a special ASIC, called CARLOS, to process the signals of the detector. It converts the analog signals to digital signals and performs data compression before sending the events through the dedicated optical links (GOL). The CARLOS-RX module interfaces the DDL to the GOL. During the tests, the events were read out and written to disk using the PRORC and DATE.

VI. Summary

The prototype PCI-based Readout Receiver Card has been developed to interface the Detector Data Link and the front-end computers. The card is able to transfer the event fragments from the link directly into the PC memory without on-board buffering at 100 MB/s, which fulfils the original bandwidth requirement of the ALICE data acquisition system. The direct memory transfer is carried out by the firmware in co-operation with the readout software running on the front-end processor. In this closely coupled operation, the role of the software is limited to the bookkeeping of the page descriptors. This approach allows sustained autonomous DMA with little CPU assistance and minimal software overhead. Performance measurements and long-term tests show that the system is fully exploiting the PCI bandwidth and that the transfer rate is largely independent from the block size. The PRORC has been integrated and tested with the ALICE data acquisition software. Moreover, the card is already in use by different sub-detector groups for testing the readout of their front-end electronics.

VII. References

[1] ALICE Collaboration, “ALICE – Technical Proposal for A Large Ion Collider Experiment at the CERN LHC”, CERN/LHCC 1995-71, December 1995

[2] G. Rubin, P.V. Vyvre, “ALICE Detector Data Link (DDL) – Interface Control Document”, ALICE Internal Note, INT-96-43, December 1996

[3] Applied Micro Circuit Corporation (AMCC), “S5935 PCI Product Data Book”, http://www.amcc.com

[4] CERN ALICE DAQ Group, “DATE V3.7 User’s Guide”, ALICE Internal Note, INT-2000-31, February 2001

[5] E. Dénes, “pRORC Library Routines, Version 2.2”, ALICE Internal Note, INT-2002-09, July 2002

[6] ALICE Technical Design Report of the Time Projection Chamber, CERN/LHCC 2000–001, January 2000

[7] ITS Technical Design Report, CERN/LHCC 1999-12, 1999