FPGA based data acquisition system for COMPASS experiment

M Bodlak¹, V Frolov³, V Jary¹, S Huber², I Konorov², D Levit², J Novy¹, R Salac¹ and M Virius¹

¹Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University, Brehova 7, 115 19 Prague 1, Czech Republic
²Physik-Department E18, Technische Universität München, James-Franck-Strasse, 85748 Garching, Germany
³Joint Institute for Nuclear Research, Joliot-Curie 6, 141980 Dubna, Moscow region, Russia
⁴European Organization for Nuclear Research - CERN, CH-1211, Geneve 23, Switzerland
E-mail: josef.novy@cern.ch

Abstract. This paper presents the current data acquisition system (DAQ) of the COMPASS experiment at CERN and discusses development of a new DAQ. The aim of the new DAQ is to substitute software event building by structure composed of special FPGA cards that will do the event building. The software part of the new DAQ is robust multinode system with high emphasis on reliability. It is based on state machines and implemented in C++ with usage of the QT framework, the DIM library, and the IPBus technology. A prototype of the system has been developed and tested. The new DAQ software fulfills given requirements.

1. Introduction

This paper presents design, implementation, and preliminary performance results of a new data acquisition system (DAQ), based on the Field Programmable Gate Array (FPGA) technology, for the COMPASS (Common Muon Proton Apparatus for Structure and Spectroscopy) experiment. COMPASS [3] is a fixed target experiment at CERN’s SPS that studies hadron structure and hadron spectroscopy with high intensity muon and hadron beams. Typical data rate of this experiment is approximately 1600 MB/s during 10 second on-spill, the off-spill time varies between 30 and 50 seconds depending on SPS super cycle. Its present DAQ system was built during years 1999-2001 [6], since then many technologies such as multiple PCI bus interfaces became obsolete and required system upgrade. The FPGA technology and the PCI-Express bus have been chosen for modernization of the DAQ for future periods of datataking. As the FPGA technology started to be cost-effective and developed enough in recent years, the usage for event building purposes is now available at reasonable cost, and thus the new FPGA based DAQ can be designed, analysed, implemented, and deployed.

The Data Acquisition and Test Environment (DATE) [10] software package and usage of FPGA-based cards have been widely studied [2], [7], [8], [9]. The study allowed to evaluate complexity of the new system and select existing libraries and software packages for the new development. The aim of the development is to build a DAQ based on state-of-the-art software and hardware designs, thus improve performance and reliability of the COMPASS DAQ. This
will be achieved by implementing event building functions in hardware and more extensive usage of Object-oriented programming in software design.

2. Technologies of the new DAQ design
The development of the new DAQ started by studying the present one, which is based on the DATE software package originally developed for the ALICE experiment at CERN. It is written in the C programming language and it uses the DIM library for communication. The Distributed Information Management System (DIM) is a multi-platform library that serves for an asynchronous one to many communication through Ethernet. Extensive state machines implemented in State Management Interface (SMI++) are part of this system. Both DIM and SMI++ libraries were originally developed for the DELPHI experiment at CERN. In the last years a growing number of tools, which use DIM, were made for the COMPASS experiment, and as a consequence usage of the DIM package for communication was recommended, thus the DIM is used in unchanged form. State machines in the present DAQ were taken as a basis for the new DAQ state machines but were significantly reworked to take into account the new DAQ architecture and implemented in the C++ programming language with QT framework. SMI++ is not used because implementation of the new state machines in C++ with QT framework proved to be more flexible and time efficient.

The Qt framework is a cross-platform application used for the development of graphical user interfaces (GUI) and for non-GUI programs. The Qt framework was chosen originally as a tool to simplify GUI implementation, but its role in the design has changed soon after start of the development and now it is used in all parts of the new DAQ designs. It was chosen due to its extensive support for multi-threading, data structures, databases (SQL), XML processing, file access, multimedia; which allows faster software development. The standard Qt SDK includes also a development environment called Qt Creator. The Qt Creator consists of code editor, distributed revision control, and performance testing tools like Valgrind and Cachegrind. All those functions make the development more efficient. Subversion is used as a version control tool.

In the new DAQ an event building functionality is implemented in custom FPGA modules. The authors believe FPGAs become the most cost efficient solution for simple data processing operations such as data checking, data sorting, data formatting, and data transmission. The FPGAs are aggressively extending their market to many digital applications by including hard cores for high speed Serial links, big memory blocks and support for SDRAM memory interfaces. The FPGA architecture has been developed for high speed parallel data processing which makes them, together with its flexibility, ideal devices for DAQs in high energy physics experiments.

The IPBus package is used for communication with FPGAs. It was developed for the level one trigger upgrade of the CMS experiment. This package consists of firmware part and software part. The firmware part mediates access to registers and memory of an FPGA card through Ethernet. The software part is implemented in C++ and contains all classes needed for a connection to the interface of the firmware part.

3. COMPASS DAQ architecture
3.1. Old DAQ architecture
The basic structure of the old DAQ architecture is shown in Figure 1. It can be divided in three parts according to their functionality: data taking, event building, and control. The data taking part incorporates frontend electronics, and data concentrators. The event building part consists of thirty readout buffer computers (ROBs) and twenty computers for the event building (EB) all of which are interconnected through Ethernet switch. The control part is represented by ordinary PCs running runcontrol and monitoring utilities (COOOL, MurphyTV, etc.) [3].
Figure 1. The old DAQ architecture

Figure 2. The new DAQ architecture
3.2. New DAQ architecture

The new DAQ replaces the event building network composed of computers with special FPGA DHC (Data Handling Card) cards as shown in Figure 2. The DHC card shown in Figure 3 has been designed as a compact AMC card and features 2 GByte of DDR3 memory, 16 high speed serial links with configurable bandwidth from 1 Gbps up to 6.25 Gbps, Gigabit Ethernet, and COMPASS Trigger Control System receiver.

![Figure 3. New DHC-Switch/Multiplexer FPGA card](image)

They will be used with two different functionalities according to two stages of the event building process. The first stage of FPGAs will multiplex up to 120 incoming links to 8 outgoing ones, providing one more data concentration level. The second stage consists of only one DHC card, which performs the event building and the event distribution functions. Events are then sent to readout computers, where they are received by Spillbuffer cards, copied via DMA to RAM, converted to the DATE format, and stored temporarily on local disks before being transferred to the CERN Advanced STORage manager (CASTOR). The Spillbuffer card is a commercially available FPGA based PCI-Express card developed by Inrevium company with SLink input interface that buffers a data stream from data producers before copying them to a computer memory.

A bandwidth of the DHC DDR3 memory is 6 GB/s which exceeds combined maximum data rate of incoming links for both DHC-Multiplexer and DHC-Switch architectures and allows to collect data without throttling data transmission even at maximum rate. The COMPASS typical data rate is 1600 MB/s during spill which is collected from more than 100 front-end modules. The multiplexer will combine data from up to 15 front-end modules providing about 240 MB/s at the outgoing link in average. The links between the multiplexers and the switch have a bandwidth of 300 MB/s each allowing transmit data without storing them in local memory. A maximum aggregated throughput of the DHC-switch is 3 GB/s. Currently the system bandwidth slightly exceeds the typical data rate and will provide data to online computers during spill. Taking into account accelerator duty cycle and significant local memory resources the system has a safety margin of 200-300%.

Newly designed event building part allows usage of more compact control system. The hardware event builder performs online data consistency check and includes programmable error recovery algorithm with configurable error tolerance level. These DAQ design features will make the system more reliable.
3.3. The new DAQ software architecture

The software part of the new DAQ consists of several processes: Master, Slave-control, Slave-readout, Runcontrol GUI, MessageLogger, and MessageBrowser. The Master process acts as the middleman between GUI and Slave processes. The Master process monitors states of Slave processes. State monitoring messages are sent at 100 Hz rate. Furthermore, the Master process performs runcontrol functionality, error handling, and data exchange with a configuration database. Slave processes access the database by sending a request to the Master process. All software parts and connections between them are shown in Figure 4.

Each Slave-control process supervises one FPGA card by accessing registers via IPBus. More precisely it configures the FPGA, monitors FPGAs resources, and informs the Master process about the state of the connected FPGAs. The full scale system will contain 17 Slave-control processes which will be distributed over the readout computers with access to two separated networks: COMPASS main network for Master-Slaves message exchange and network with FPGA cards for IPBus traffic.

The Slave-readout process is the most complex and demands most of CPU resources in the new DAQ. It is a multithreaded process that monitors readout activities. The data sources used in this process are derived from a common abstract class; thus it is easy to add a new type of device to the system. This construction significantly increases flexibility of the system. A Spillbuffer card is used as the only data source in both the old and in the new DAQ. Data consistency is verified during readout. The data are transferred between threads via signal-slot connections mechanism of Qt by blocks of about 1000 events. The last task is to distribute data to outputs according to the set filter parameters. All outputs have their own threads.

The GUI runcontrol process is the most complicated one from the design point of view because it provides visual information and receives commands from expert and nonexpert users as well. Its structure is designed and developed with emphasis on ergonomy and flexibility. It provides different sets of DAQ status information for expert and nonexpert users. It is designed as fully

![Figure 4. Communication diagram](image-url)
customizable. It runs in one of two modes: primary and secondary. There is only one primary GUI runcontrol allowed in the system; it controls and monitors state of system. The number of running secondary GUIs is not limited, as they are used only for monitoring.

The last two programs to be discussed are the MessageBrowser and MessageLogger [8]. The MessageLogger receives messages from all parts of the system and stores them in the database. The MessageBrowser is a visualization tool for browsing through these messages.

4. Tests of prototype
Several tests have been done to validate system components. The performance of the DIM server has been measured. The graph in Figure 5 shows that for messages sizes up to 2 kB the DIM server limits maximum number of processed messages while with bigger message sizes a traffic reaches almost maximum network bandwidth of 100 MB/s. The maximum message frequency reached during tests exceeded 100 kHz. The system consists of 8 Slave-control processes for MUX, 1 Slave-control process for SWITCH, 8 Slave-control processes for Spillbuffer, and 8 Slave-readout processes (total 25 slaves processes); it will generate 2000 messages per second (80 messages per slave per second) during normal operation leaving enough room to handle additional error and debug information.

![Figure 5. DIM server speed performance](image.png)

The second set of tests measured combined performance of Spill Buffer PCI-Express interface, PCI-Express driver and readout program. Tests were carried out on one server with quadcore 3 GHz CPU, 4 GB RAM, 1 TB HDD, Scientific Linux CERN 5 OS 32-bit and two spillbuffer cards. The graph in Figure 6 shows maximum speed as a function of the event size. The red vertical line in the graph shows the expected size of events in the next run of the COMPASS experiment. The maximum speed was limited by the bandwidth of Slink interfaces which were used for these tests. If one scales the system up to 8 online computers each equipped with one spillbuffer card, the total data rate will reach 1.1 GB/s. This performance supported by buffering capabilities will be sufficient to process typical COMPASS data rate and thus the full scale system will fulfill the requirements.
5. Conclusion

Demands on the data acquisition system were extracted from the initial studies of the present data acquisition software of the COMPASS experiment at CERN. The new DAQ design is based on these demands and restrictions. Prototype versions of all processes were implemented in the C++ language using the QT library. The performed tests proved the viability of the designed solution. The COMPASS Technical Board has approved the system deployment for the next running period starting from 2014.

Acknowledgments

The work on this project has been supported in part by the following grants: MSMT LA08015 and SGS 11/167. It is also being supported by the Maier-Leibnitz-Labor, Garching and the DFG cluster of excellence Origin and Structure of the Universe.

References

[1] M. Bodlak, et al. Developing Control and Monitoring Software for the Data Acquisition System of the COMPASS Experiment at CERN. Acta Polytechnica: Scientific Journal of the Czech Technical University in Prague. Prague, CTU, 2013, issue 4. Available at: http://ctn.cvut.cz/ap/
[2] M. Bodlak, et al. New data acquisition system for the COMPASS experiment. Journal of Instrumentation. 2013-02-01, vol. 8, issue 02, C02009-C02009. DOI: 10.1088/1748-0221/8/02/C02009. Available at: http://stacks.iop.org/1748-0221/8/i=02/a=C02009?key=crossref.a76044facdf29d0f21f9e63305aa5
[3] P. Abbon, et al.(the COMPASS collaboration): The COMPASS experiment at CERN. In: Nucl. Instrum. Methods Phys. Res., A 577, 3 (2007) pp. 455-458
[4] T. Anticic, et al. (the ALICE collaboration): ALICE DAQ and ECS Users Guide. CERN, ALICE internal note, ALICE-INT-2005-015, 2005.
[5] M. Bodlak, V. Jary, J. Novy: Software for the new COMPASS data acquisition system. In: COMPASS collaboration meeting, Geneva, Switzerland, 18 November 2011
[6] L. Schmitt, et al.: The DAQ of the COMPASS experiment. In: 13th IEEE-NPSS Real Time Conference 2003, Montreal, Canada, 1823 May 2003, pp. 439-444
[7] V. Jary: Analysis and proposal of the new architecture of the selected parts of the software support of the COMPASS experiment Prague, 2012, Doctoral thesis, Czech Technical University in Prague
[8] M. Bodlak: COMPASS DAQ Database Architecture and Support Utilities Prague, 2012, Master thesis, Czech Technical University in Prague
[9] J. Novy: COMPASS DAQ - Basic Control System Prague, 2012, Master thesis, Czech Technical University in Prague
[10] T. Anticic, et al. (ALICE DAQ Project): ALICE DAQ and ECS User’s Guide CERN, EDMS 616039, January 2006.