AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY

FACULTY OF COMPUTER SCIENCE, ELECTRONICS AND TELECOMMUNICATIONS

A NOVEL DATA ACQUISITION SYSTEM BASED ON FAST OPTICAL LINKS AND UNIVERSAL READOUT BOARDS

Ph.D. Dissertation

Author:

Mgr Grzegorz Korcyl

Supervisor:

Prof. Dr Hab. Piotr Salabura

Kraków, 2015

AKADEMIA GÓRNICZO-HUTNICZA

WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI

NOWOCZESNY SYSTEM AKWIZYCJI DANYCH OPARTY NA SZYBKICH POŁĄCZENIACH OPTYCZNYCH I UNIWERSALNYCH PŁYTACH ODCZYTU

Rozprawa Doktorska

Autor:

Mgr Grzegorz Korcyl

Promotor:

Prof. Dr Hab. Piotr Salabura

Kraków, 2015

ACKNOWLEDGMENTS

Foremost I want to thank my supervisor Prof. Piotr Salabura. His great support and guidance allowed me to explore the field of technology in large physics experiments.

I also want to thank my foreign supervisors: Dr. Michael Traxler from GSI Darmstadt and Prof. Wolfgang Kuehn from Justus-Liebig Universitaet in Giessen.

Being a member of HADES and PANDA Collaborations, I want to thank all the members for giving me the opportunity to be a part of those projects and sharing their experience with me.

I gratefully acknowledge the founding support from the Foundation of Polish Science and from the Jagiellonian University in Cracow.

This work would not be possible without my family and friends, therefore I want to express my gratitude for their support.

ABSTRACT

Various scale measurement systems are composed of the sensors providing data through the data acquisition system to the archiving facility. The scale of such systems is determined by the number of sensors that require processing and can vary from few up to hundreds of thousands. The number and the type of sensors impose several requirements on the data acquisition system like readout frequency, measurement precision and online analysis algorithms. The most challenging applications are the large scale experiments in nuclear and particle physics.

This thesis presents a concept, construction and tests of a modular and scalable, tree-structured architecture of a data acquisition system. The system is composed out of two logical elements: endpoints which are the modules providing data and hubs that concentrate the data streams from the endpoints and provide connectivity with the rest of the system. Those two logical functions are realised by the base modules called Trigger Readout Board (abbr. TRB) which feature basic functionality: digitization of the signals, communication with other modules and external networks, control and monitoring mechanisms. This set of functions can be extended on the modules via a system of Add-on boards that introduce new features and allows to adapt the platform for various applications.

The key characteristics of TRB based system are: scalable, flexible, extensible and reconfigurable. The scalability of the platform is realized by the hub components, which allow to create tree structures with many layers, each opening new ports for additional endpoints, without reducing the performance of the entire system. The TRB boards are based on FPGAs, which are reconfigurable, programmable logic devices. This approach results in a possible use of the same hardware module for different functions with just a change of the firmware. It also allows to introduce new functionalities over time. Together with the Add-on system, the platform can be relatively easily adapted to various applications and extended with new elements.

The platform was developed inside the HADES Collaboration with significant contribution from the author. The HADES detector was also the largest target application and was used for extensive tests of the system. Several conducted experiments and laboratory tests described in this thesis confirm the design and allow to evaluate the system performance. The platform has also found application in various other systems, one of them being the J-PET medical imaging project also described in this thesis.

STRESZCZENIE

Systemy pomiarowe różnej skali składają się z czujników dostarczających dane poprzez system akwizycji danych do infrastruktury archiwizującej. Skala takich systemów jest określana przez liczbę czujników wymagających procesowania i może się wahać od kilku do kilkuset tysięcy. Liczba i typ czujników nakładają szereg wymagań dotyczących systemu akwizycji danych, takich jak częstotliwość odczytu, precyzja pomiaru czy algorytmy analizy w czasie rzeczywistym. Najbardziej wymagającym z tego punktu widzenia zadaniem są eksperymenty fizyczne dużej skali.

W tej pracy zaprezentowana jest koncepcja modularnej i skalowanej architektury w postaci drzewa systemu akwizycji danych. System składa się z dwóch logicznych elementów: elementy końcowych, które dostarczają dane oraz koncentratorów, które odbierają strumienie danych i zapewniają komunikację z całością systemu. Te dwie funkcjonalności logiczne są realizowane przez moduły bazowy Trigger Readout Board (TRB), który zapewnia podstawową funkcjonalność: digitalizacji sygnałów, komunikacji z pozostałymi modułami oraz zewnętrznymi sieciami oraz mechanizmy kontrolno-sterujące. Ten zbiór funkcji może zostać powiększony poprzez system płyt Add-on, które pozwalają wprowadzić nowe funkcjonalności i przystosować platformę do różnego rodzaju zastosowań.

Kluczowymi cechami systemu opartego na modułach TRB są: skalowalność, elastyczność, rozszerzalność oraz możliwość rekonfiguracji. Skalowalność platformy została osiągnięta poprzez zastosowanie modułów koncentratorów, które pozwalają tworzyć drzewiaste struktury z wieloma warstwami, z których każda otwiera nowe porty dla elementów końcowych, bez pogarszania parametrów wydajnościowych systemu. Płyty TRB są oparte na układach FPGA, które są rekonfigurowalnymi, programowalnymi układami logicznymi. Dzięki takiemu podejściu, można używać modułu bazowego do realizowania różnych funkcji, jedynie poprzez zmianę oprogramowania wbudowanego. To również pozwala na wprowadzanie nowych funkcjonalności z biegiem czasu. W połączeniu z systemem płyt Add-on, takie rozwiązanie pozawala na stosunkowo łatwe zaadaptowanie platformy TRB do różnych zastosowań.

Platforma została rozwinięta wewnątrz Kolaboracji HADES przy znaczącym wkładzie autora. Detektor HADES był również docelowym zastosowaniem systemu i został użyty do przeprowadzenia rozległych testów systemu. Szereg przeprowadzonych eksperymentów oraz testy laboratoryjne potwierdzają projekt architektury oraz pozwalają na ewaluację wydajności systemu. Platforma znalazła również zastosowanie w wielu innych aplikacjach. Jedną z nich jest projekt J-PET, tomografu do obrazowania medycznego, który również został ujęty w tej pracy.

TABLE OF CONTENTS

1 Introduction

Modern, large scale measurement systems and real time data processing facilities require the design and development of efficient platforms, which aim is to process multiple data streams in parallel and deliver the product to a final archiving destination. Such platforms are being widely used in the industry sectors like automotive, aerospace, energetics and consumer electronics but also in science fields like medical imaging and experimental physics. Taking for example the ATLAS experiment (CERN, Switzerland) (1), commonly known as the largest detector system ever built on Earth, it features more than one hundred million analog channels that have to be precisely measured, in a synchronized way. The data has to be processed, analysed in the real time and reduced by filtering the unimportant data, treated as noise, before reaching final location.

The scale of those projects impose the development of dedicated platforms, fine-tuned for fulfilling the very high requirements in terms of the precision, readout speed and channel density. The commercially available standards like VME (2) or CAMAC (3) are difficult to scale and their costs are significant. This thesis presents a hierarchical system based on a universal base module called Trigger Readout Board (abbr. TRB) (4), originally designed for the High Acceptance DiElectron Spectrometer (abbr. HADES) at GSI, Germany (5). The base module features connectivity and data processing functionality as well as measurement mechanisms provided by dedicated extension modules. Such high flexibility could be achieved by the use of custom electronics equipped with Field-Programmable Gate Array (abbr. FPGA) reconfigurable, programmable logic devices (6). The base modules can be connected in a tree, hierarchical architecture, which provides high scalability, which is crucial for various applications.

The intention of this work is to present a complete, modular and scalable system for a streamlined data processing in the real time regime. The development consists of many aspects, starting with definition of the system architecture, specification of the electronic modules, together with the base module TRB, implementation of the firmware for FPGAs, definition of custom communication protocols and data structures, up to the implementation of the software needed to control and to monitor the acquisition process. The system is an alternative for standardized many years ago architectures, which in today's applications present low scalability and functionality, thus not applicable for modern measurement systems, also because of the still remaining high costs. The proposition of a high performance platform, compliant with today's standards in terms of data processing and transmission, universal and flexible enough to be applicable for a wide range of experiments and detection systems is a response for the legacy platforms. The success of the developed platform is proven by the significant interest in the community and the number of various scale applications, to which the system was adapted, both in experimental physics and medical imaging device prototypes. The author of this thesis participated in this challenging task providing important contributions to the design of the system architecture - implementing various elements of the system, adapting the solutions for diverse applications and performing evaluation tests. Some of the developed solutions were enclosed in the international patent application (7) and published in international journals (8), (4), (9), (10).

Prior to present the system under discussion, a general introduction to Data Acquisition Systems is conducted. The Chapter 2 opens by covering the basic elements which are building blocks of a measurement system. It is followed by an overview of the concept of triggering and data discrimination. Together with the slow control system, presented in the next section, they cover the basic ingredients of the Data Acquisition System (abbr. DAQ). The chapter is concluded with comparison of different standard DAQ platforms commercially available and a discussion about the challenges and requirements facing modern data acquisition systems.

The Field Programmable Gate Arrays are devices originating the 90's but nowadays experiencing dynamic development. This technology is widely used in modern measurement systems, hence the entire Chapter 3 is dedicated to presentation of the device structure and methodology of firmware development. A brief introduction of VHDL (11), as the hardware description language used for implementing the logic, is included as well as a section where the FPGAs are compared to other computing platforms in order to visualize the differences in their architectures which are essential for DAQ applications.

The body of the thesis is enclosed in Chapters 4 and 5. The first one describes in details the proposed architecture of the data acquisition system, which is under discussion. The focus is on the network communication and data transmission realized by the implementation of Gigabit Ethernet Module (Section 4.5) and on the new TRB3 platform (Section 4.4) which was the original contribution of the author of this thesis. The Chapter 5 covers main applications, where the elements of the system are used under various forms. The description of HADES experiment is a showcase of the entire, large-scale system setup. In contrary, the J-PET application shows how the components can be applied to smaller setups with different requirements.

Chapter 6 encloses results of measurements of the platform performance. Both a laboratory setup and the real applications are tested in terms of scalability issues, data throughput and readout rate, which are the key characteristics of the system architecture.

The work is concluded with a chapter summarizing achieved results and opening a discussion about possible improvement of specific components and mechanisms.

2 OVERVIEW OF DATA ACQUISITION SYSTEMS

Automation of many processes requires the use of various sensors. One of such applications are advanced, industrial production lines. They operate autonomously by processing feedback from devices that monitor a number of characteristics, describing the conditions and manufacturing parameters. It is used for both: quality assessment and real-time adjustments of production processes. Automation of the monitoring can be found in almost all aspects of our lives: from monitoring the condition of power plants and delivery lines through massive transportation systems, weather, up to something very personal like miniature health monitoring devices, that we can carry in our pockets. All those applications complete a standard scheme that consists of sensors, measurement devices and electronics with software that process gathered data. The sensor detects some physics phenomena and converts it into an electrical impulse, which is then digitized by dedicated electronics. The result is processed for extracting some interesting feature, presenting the result in some form and archiving. The set of electronics, firmware and software needed to process the data from the sensors is called Data Acquisition System.

One can distinguish three levels while trying to categorize the DAQ systems. First level, would be very heavy duty applications, like systems used in mining, petrol facilities or transportation. They are designed to operate at very harsh conditions, thus robustness is their key feature. Measurement precision, number of sensors and readout frequency is relatively low. The second level is dedicated to applications which require moderate precision and readout frequency, while keeping limited number of input channels. The most advanced systems, like the ones used in physics experiments, impose the highest requirements. The third level is reserved to applications which operate on hundred thousand or even hundred millions of sensors that measure with resolutions of single nanoseconds at megahertz frequencies.

All three levels, however, share some common functions. They all need a real-time data path from the sensors, through the digitizing device and some sort of processing unit. In the units, algorithms used to process data are designed to extract features from the incoming data stream and generate triggers that will have immediate result on the operation of the system. In applications where information loss due to high readout rate is significant, this critical path and processing time have to be minimized. The time needed to process a given portion of data, during which the system cannot accept any input is called the dead time.

Each of the levels presents a set of different requirements, which impose the use of technologies developed for those dedicated applications. Heavy duty systems are often based on Programmable Logic Controllers (abbr. PLC), which are modular computers typically used for industrial processes. They can implement real-time algorithms, driven by the input ports. More advanced features are presented by systems like LabView from National Instruments. Those are complex solutions developed for measurement and monitoring systems, instrument control and validation systems. They are successfully used for many laboratory applications, as small scale, off-the-shelf measurement stations. Although they present versatile functionality, their scalability is limited and costs per channel are significant. Requirements imposed by applications from the third category, force the development of dedicated, custom solutions, fine-tuned for achieving the peak performance.

A particular example and undoubtedly the most advanced in terms of technology and demands are systems used in particle physics experiments. Although each experiment faces a different aspect, the main structure of their construction is common. Usually experiments are located at particle accelerators which boost projectiles to a specified energy and then hit either stationary target or other beam. The reaction products are measured by a dedicated detector system. The response of the detectors is then registered by specialized readout electronics (analog and digital) and transmitted to some storage devices for further analysis. Properly designed data acquisition system working together with the trigger system (a system which makes a decision to store or abandon given reaction event) are key elements for the efficient collection of data.

The system described in this work was designed to collect and process data from detectors composing a system for experiments in particle physics, though general enough to be applied to any system demanding processing data from some electronic sensors. The following description will focus on the overview of components required to read out the sensors (hereinafter called detectors) which are the state of art devices, developed to measure and register the smallest and the most elusive phenomena humanity could witness so far. This is the reason why the DAQ system for such application must follow the especially high requirements.

Regarding a system architecture, the simplest case scenario is a single channel which generally consists of analog (shaper), digital processing units and transmission path to the storage unit. Such chain can be extended to larger amount of independent channels, by the introduction of additional elements that collect data from several sources. The main challenge is however to avoid in the same time degradation of the performance in terms of rate capabilities of the system. The aim of reading out the sensors is to transform system response to a physical event, classified by the time when it happened, into a set of digital values that represent parameters of the response signal that is interesting from the analysis point of view. The parameters can be the amplitude of the analog signal, the integrated charge of the signal or the time when the signal was generated. The electronic modules that create such information are called digitizers e.g. Analog to Digital Converter abbr. ADC or Time to Digital Converter abbr. TDC. For example signal coming from a typical detector (Figure 1) (e.g. current or voltage pulses) have to be prepared for the digitizer by dedicated analog electronics consisting of several steps like signal amplification, signal shaping and in case of TDC comparison with a predefined discrimination level (fast discriminator).

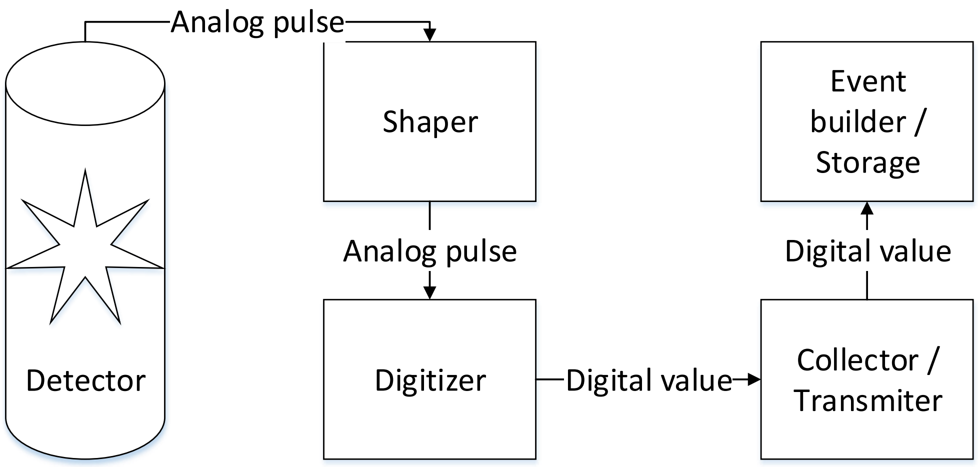


Figure 1: Building blocks of a single channel readout chain. A physical event excites the detector to generate an analog signal that is processed by shapers, digitizers, data collectors and transmitters. The output is saved by Event Builders for offline analysis.

Having digital values is already half of the success. Those values now have to be transported to some permanent storage for later analysis, which in case of a single channel is not a challenge, but in case of complex setups with thousands or millions of sensors, the networking becomes state of art.

In order to fully understand the requirements facing modern data acquisition systems in physics experiments, it's important to understand properties and demands addressed to all elements of DAQ. The first part of the next section is covering all the stages of DAQ chain, describing step by step all its elements, starting with different detectors types and the origin of analog signals. Then the concept of triggering is introduced as a way of selection and preliminary data reduction. Overview of standard, commercially available DAQ platforms is presented in a section followed by description of Slow Control. The chapter is closed by a section that aggregates the challenges facing the design and development of a typical acquisition system.

2.1 Data Acquisition Systems

Large scale physics experiments use various types of detectors to gather the maximum amount of possible interesting information from a single physical event. Each type of the detector represents a specific reaction to a given radiation type (e.g. charge particles, neutrals (photons, neutrons)) and its conversion into output signal, which in general is an electrical impulse. Those responses depend on the type of detector and reactions, thus each detector has to be equipped with a specifically designed readout chain. All such subsystems have to be combined at some point into one unified system. The entire chain can be divided into functional parts, which will be described below, starting from the detector itself up to the final storage device.

2.1.1 Detectors

There are few main characteristics (12) which describe general characteristics and capabilities of a particle detector:

- Sensitivity
  Detector materials and its construction is selected to be sensitive to a given type of radiation at a specified energy range. As it's the first consideration while selecting a detector type, it influences most of the following points.

- Type of detector response
  Usually the information that a reaction happened in a detector or not is not enough. It's also important to know the energy deposited in the detector material and/or the time of arrival by the hitting particle. In terms of electrical response, this is reflected as the integrated charge of the output signal or as its amplitude and/or time when the signal crosses predefined discrimination level.

- Energy, time resolution
  This parameter defines the capability of the detector to distinguish different energies deposed by the particles and the time of arrival. The smaller is the measured difference in case of two identical signals, the more useful is that information.

- Response function
  Response function define the shape of the output signal that is generated by the given particle (e.g. electron, muon, pion etc.). Good knowledge of the response function is essential in order to distinguish between particle species and defines conditions on quality of detector electronics.

- Response time
  This factor is very important and strongly connected with the next point, the dead time of DAQ. Under a strong irradiation, the rate of physical event which have to be properly handled by the detector is very high. The response time is the time that the detector materials and analog electronics spend on constructing the output signal. The more time it takes, the higher is the probability that it will register another reaction, leading to the mix-up of both, which is called a pile-up effect.

- Dead time
  This parameter describes the time needed by each part of the detection system to properly process the event. It's strongly related to the response time and influences detector efficiency.

- Detector efficiency
  The most relevant measure of the detector quality and its suitability for a given reaction to register. Taking into account all the previous parameters, this one is the number of properly registered events compared to the number of emitted events by the source.

2.1.2 Front-End Electronics

The parameters described above have to be taken into account while designing the Front-End Electronics (abbr. FEE). The shape of the output signal is driven by each element of the detector such as used materials, gas mixture or distribution of high voltage. The detectors are adjusted to generate a proper output signal in case of a given type of reaction in order to eliminate noise. Usually the signals are very fast (width of order of several to tens of nano seconds) and can be small (amplitudes of millivolts on 50 Ω). Therefore it is required to design the Front-End Electronics, which will prepare those signals for digitization process. The nature of the signals force the use of very fast processing systems, that's why the fast, analog components are used at the stage of amplification and shaping.

The Front-End Electronics act as the interface between the response signal from the detector and the characteristics of the proper input for the digitizers. The transfer function (transformation from the input to the output signal) of the FEE has to be defined depending on the type of the measurement (whether it is time, charge or the amplitude):

- Amplitude or charge measurement
  Applications where the shape of the signal is analyzed, require very well defined output signals. That is why the electronics are equipped with amplifiers and shapers. First ones adjust the amplitude of the signal to the acceptable amplitudes range of the digitizer. Shapers are used in order to emphasize some characteristics of the signal (e.g. long rising edge of the signal required for taking several samples).

- Time measurement
  A very fast discrimination technique is required in order to separate the signals from noise. Discriminators compare the amplitude of the input signal to the applied threshold level and in case the signal is large enough, an impulse is created on the output of the device. The time difference between this impulse and some reference signal is the exact result of the measurement. The quality of the discriminator defines the time jitter of the output as a response for the same input signal as well as the time needed for generating the response (longer time leads to larger dead time).

2.1.3 Digitizers

When designing the entire data acquisition system, the first thing to start from is the nature and the characteristics of the response signal from all the detectors that need to be read out. Next step is the selection of the digitizing device suitable for the wanted kind of measurement. Such devices work properly when the input signal parameters fit into some range of values. For this reason the raw, output signal from the detector is passed through analog electronics, as described in the previous section. Usually, the larger, longer and better shaped signals are easier to digitize thus generating a more accurate result. On the other hand, preparing a longer signal takes time, which increases dead time of the readout, which in turn might reduce the efficiency in case of high reaction rates. That's why, each scenario has to be analyzed individually and the best balance between wanted result accuracy, overall efficiency and usually the cost per channel has to be worked out. It is preferable to perform digitalization as soon as possible (closest to the detector) as the digital data is less affected by noise interference. The measurement devices can be grouped by the aspect of the analog signal that they observe.

- Analog to Digital Converters (abbr. ADC)
  ADC are devices (13) that generate a series of digital values that approximate the input, analog signal. There are different ways of converting the input voltage into a digital value. The basic idea consists of a chain of comparators, where each one has a different threshold applied, distributed with an even step. Applying an input voltage which is inside the range defined by the thresholds of the comparators, some of them will respond with an active signal, meaning that the input voltage was higher than the applied threshold. Such a measurement is called quantization, repeated at high frequency is called sampling process and results in digital representation of the analog signal in a function of time (example of sampling ADC output on Figure 2). The above described solution forms a device called sampling ADC. Another approach to that matter is the integrating ADC, which instead of comparing the input analog signal, first passes it through a capacitor that collects carried charge and measures its value.

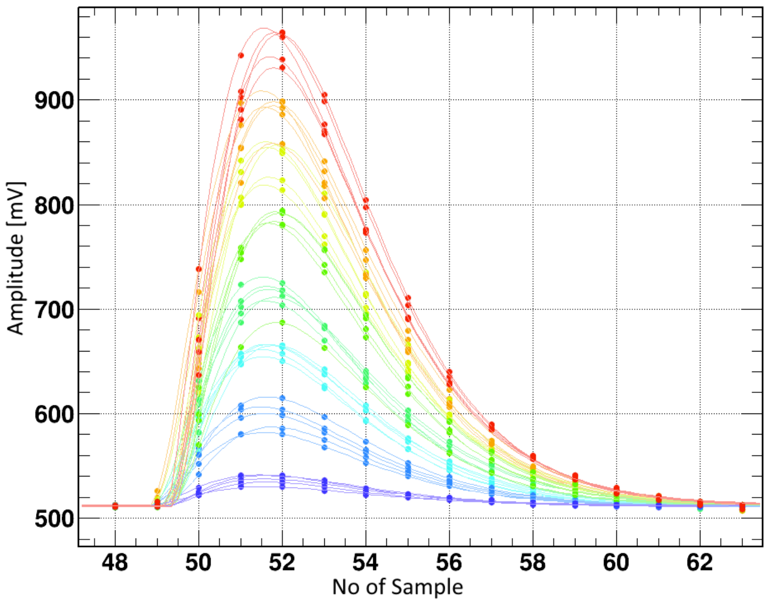


Figure 2: Reconstruction of analog signals from Electromagnetic Calorimeter for HADES experiment of different amplitudes (different colours), measured with sampling ADC. The X axis represents number of sample, which is also marked by dots on the plots, the Y axis represents mV of measured signal at the sampling point. (49)

There are several parameters that define the quality of a single measurement performed by the ADC. Of course the resolution and the sampling rate are the most relevant ones. Resolution is defined by the amount of comparison points of the analog value (e.g. number of comparators to which input voltage is applied). The higher resolution, the more accurately digital value will represent measured voltage. Time distance between two consecutives samples is called sampling rate. Large number of samples collected in a short period of time, gives a possibility to recover the shape of the analog signal. The error in measurement is introduced by several elements in ADC devices. The most important is obviously its resolution but also the nonlinearity of consisting elements and general noise toleration are strong factors.

- Time to Digital Converters (abbr. TDC)
  Applications like time of flight measurement in tracking detectors require devices that can precisely measure the time elapsed between two events. Technology of time measurement has made a huge step forward last years, improving the resolution to single picoseconds using digital TDCs (14) implemented in Field Programmable Gate Arrays (6). Time to Digital Converters make use of the fact that signal propagation through electronic elements requires some time. By creating a chain of such elements and injecting a pulse inside, one can estimate how wide was that pulse or what was the time distance between two signals (Figure 3). The smaller is the delay introduced by a single element the better is overall time resolution. There are of course several obstacles that need to be considered. All elements composing a single delay chain should have the same signal propagation time, any deviation of the mean value is called Differential Non Linearity (abbr. DNL) and is a main factor lowering the resolution of the measurement. What is worse is that those parameters can change due to temperature or input voltage fluctuations during the run time. Some techniques exist that can help reducing this problem, one of those is the wave-union method, which involves performing several measurements on one delay chain, using one input signal.

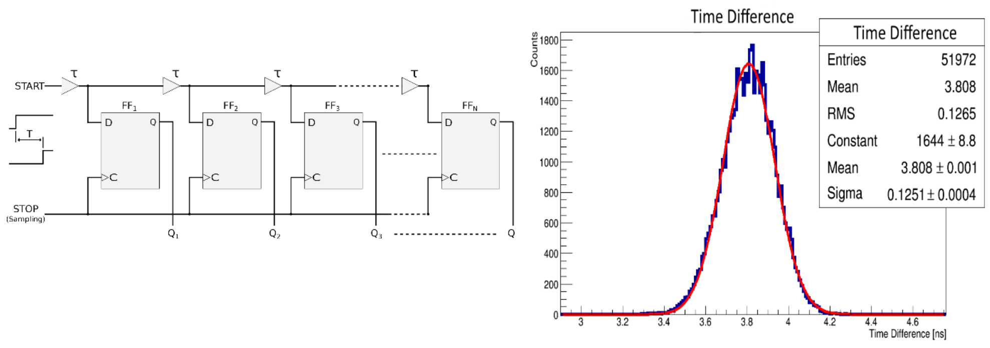


Figure 3: Structure of a single channel of TDC implemented in FPGA logic. Amount of delay elements traversed by the input signal is translated into period of time. Histogram on the right represents a time difference measurement of two photomultipliers used for JPET tomograph prototype. The achieved time resolution is 125ps which include the detector response fluctuations and front-end module jitter.

2.1.4 Data Concentrators, Networking and Event Building

The readout of a single detector channel can be successfully realized with a digital oscilloscope which contains the above mentioned measurement devices. When it comes to high energy physics experiments, their scale can be overwhelming. Taking HADES as an example (which is a medium sized experiment), it consists of 7 detector subsystems, summing up to total of 80,000 analog channels. It's a true challenge for the DAQ and network designers to assure that the data transport will be reliable and fast enough. Each channel has to be shaped, measured, marked with its coordinates in the system and then transported to event building computers. Data collection takes place on several stages, as well on hardware level (ADCs and TDCs usually have several input channels) as on logic (data from a number of digitizers forms larger blocks, which are treated as entities). The data concentration in the DAQ readout chain is performed up to a stage when the data can exit the digital electronics and enter a standardized network via a network gateway, leading to the event builders. The number of concentration steps depends on the type of hardware used in the system and the systems architecture. Analysis algorithms implemented in hardware might require data from a selected sectors or detector subsets. The data from single channels must be therefore properly grouped, marked, packed and delivered to the component running analysis. In case of systems without online hardware analysis features it is preferable to forward the data to the network gateways as soon as possible, as it reduces the costs, complexity level of the system and the dead time.

Concentrators are also used in the opposite direction, not only gathering data coming from the detectors but also distributing the trigger and slow control information. Trigger system (see next section) is required to control the process of the entire readout. The trigger information, under different forms (in some systems it is just an analog impulse and in the others it is a data packet) must be delivered to all the endpoints of the DAQ, including Front-End boards. The concentrator facility can be used to transport that information in the downstream direction. It is the same situation when it comes to slow control. Slow control gives access to the settings of the components in the system from a central point and to monitor state of its components. It is crucial to have a possibility of configuring the system and adjusting its operation according to given experiment conditions. The same infrastructure can be used in order to transmit, broadcast and gather configuration settings from the system components.

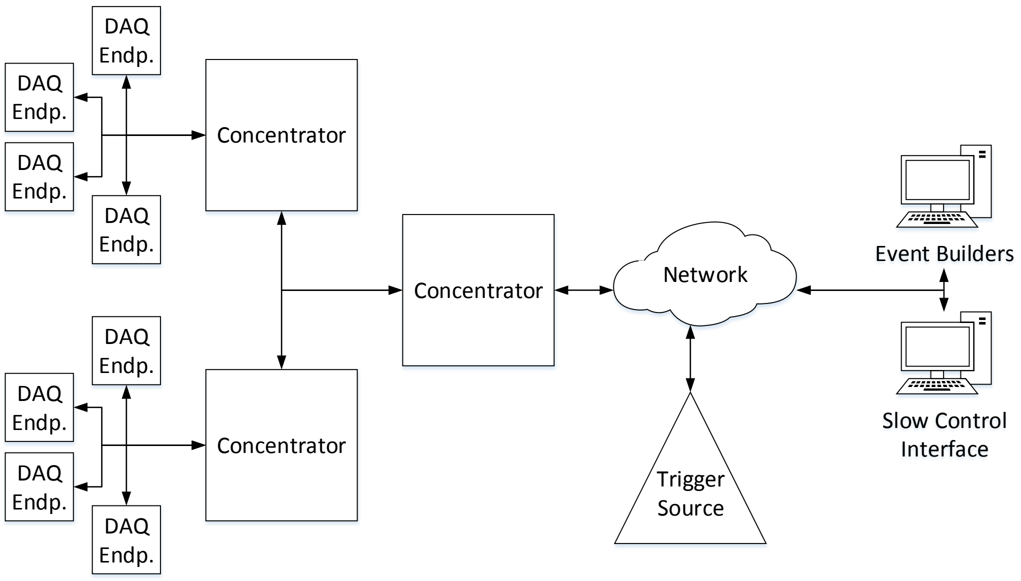


Figure 4: Scheme of the DAQ components interconnections with subsystems. DAQ Endpoints (digitizers, active FEEs, etc.) communicate through several layers of concentrators and network with Event Building machines, Slow Control computers and a Trigger System.

All those three functionalities (data readout, trigger and slow control distribution) require data to cross between two or more different subsystems (Figure 4). The DAQ electronics need to transfer readout data to the Event Building machines, the Slow Control commands generated on a supervisor PC need to be sent to certain DAQ Endpoints and the Trigger information has to be delivered to all the components. Having many interconnected subsystems that exchange data between modules requires a unified network infrastructure. As the industrial telecommunication sector is expanding, there are many out of the box solutions available at the market. The only requirement is that all the subsystems have to be equipped with network gateways applying a chosen standard. Nowadays, the Gigabit Ethernet (15) is a commonly found solution in many existing experiments. It is a well-known, easy to implement and verified standard with affordable equipment.

Event building is a task of reconstructing an entire event from all the small pieces of data, coming from different sources in the whole system, it's the last step of concentration. Depending on the system architecture and the online analysis algorithms, the reassembly can be realized either in the hardware or by the event building computers. In cases where the data from the entire system is needed for an online algorithm, the concentration has to merge all the parts and deliver an entire event to that stage. In other cases, algorithms are performed locally, on a subset of data coming from a given subsystem, hence the event building combines those local parts together into a single unit. Moreover, the raw detector data is often extended by the results of the online algorithms, which have to be properly attached to a correct event data.

Hardware event building is expensive and difficult to implement. The data has to be reassembled, stored in buffers and transmitted as a whole unit to the storage devices. The main problem is the buffering and memory needed for such online processing. It is usually avoided in applications, where there is no requirement of analyzing entire events on the hardware level. Preferably the small data parts are being sent from the DAQ electronics, through the network to the event building machines. Although, high fragmentation can lead to overloading the receiving computers.

Event building machines come in form of powerful, often server class computers with multi-processor support, high speed network interfaces and large disk space. Those requirements are crucial in order to receive and store data streams from the DAQ system. The data is stored temporarily on local hard drives before it is transferred to some permanent storage. As all the data from experiments has to be accessible at any time by physicists, it is required to use technologies that are reliable and robust. One of such is the storage on magnetic tapes which is an expensive and slow medium but gives the best results as permanent storage. It should be noted that, the data volume from a single run of a medium-sized experiment can reach up to dozen petabytes, which have to be stored for unlimited amount of time.

2.2 Trigger Systems and Data Discrimination

The rate of physical events taking place in a detector system is very high in comparison to the rate of events really interesting for the physicists. An example that is usually brought up is the Large Hadron Collider (abbr. LHC) constructed at CERN facility. The event rate (16) expected for the operational energy settles at level of 10^9 events per second, while the Higgs Boson is due to appear only once per second. That means that for such a case, all the other events are not valuable from the physics point of view and are treated as noise. The mechanism of selecting events that are supposed to contain important information is called triggering and is crucial for an efficient operation of the entire experiment. Trigger systems are hierarchical mechanisms that preform data filtering on several stages, passing to the next stage only data which was proven positive for passing the tests. Each trigger stage introduces more advanced method or algorithm to select only valuable events. Those systems are very efficient in reducing the amount of data that needs to be processed by the entire system, but also help physicists by providing them cleaner data, with less unimportant, noise events. On the other hand there is a risk of rejecting valuable data. The trigger mechanism introduces a latency which can result in a dead time and lower overall rate of detectors readout, thus resulting in data loss. The online analysis algorithms require time to perform which enforces the buffering of the data at the early stages. Taking high rates into account, those processes always result in lowering the overall efficiency of the system, but deliver more valuable events.

There is also an alternative approach, which doesn't make use of trigger system, instead it uses a more advanced discrimination mechanisms. An example of such system is the design of PANDA data acquisition system (17), where data is taken continuously and is buffered in the DAQ electronics, waiting until the readout signal arrives, which happens with a fixed frequency. The data organized in time epoques is then transferred to powerful computer farm for the event processing with several alternative trigger algorithms. Such approach is called trigger-less system and depends on discrimination power of data at the earliest stage possible and powerful network capabilities allowing the transport of data representing the detectors state over an entire period of time.

As the detectors and front-end electronics work continuously processing analog signals into digital data, the trigger signal can be treated as the signal that selects which data should be passed to the next stage of the readout chain. Usually some crucial parts of detectors are being analyzed in terms of response existence and put through some Boolean function. An example of such approach is the multiplicity or coincidence triggers, which activate the readout only in case the amount of detector channels that have produced a proper response is higher than some limit. Of course there exist solutions combining different approaches together, creating a tailored solution for a given experiment requirements.

Higher trigger levels often include pattern recognition algorithms, which can be implemented in FPGA devices or on GPUs. This assures the minimum dead time and allows highly parallel solutions. An example of such algorithms is the recognition of detector channels that fired forming some kind of geometry figures like straight lines or circles as it is the case in Ring Imaging Cherenkov Detector (abbr. RICH) detectors type (Figure 5).

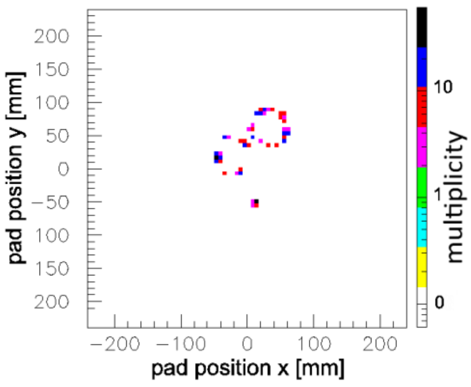


Figure 5: An example of an event collected with RICH (37) detector in HADES experiment. Colored pixels represent cells in the detector which fired, surpassing thresholds. One can notice ring shaped clusters that represent photons.

2.3 Slow Control

Another key element of the DAQ system is the ability to control its elements. Such subsystem, called Slow Control System (abbr. SCS), requires hardware facility, firmware and software allowing to configure parameters of the readout process and permanently monitor its behavior. Hardware needs to be equipped with some sort of network interface. Hence the firmware on the electronics also has to support the selected protocols and communication standard. In modern systems, the basic channel for Slow Control is the Gigabit Ethernet connection. It features individual, broadcast or multicast addressing and routing capabilities which allow to communicate with selected elements of the system. An important aspect is also the fact that basic networking knowledge is enough to write software for sending and receiving Slow Control commands from the PC level.

There are systems which mix the communication standards in order to reduce required hardware and use efficiently the existing components. An example of such system is the HADES DAQ, described in more details in section 5.1. Slow Control commands are being sent from the controlling PC over Gigabit Ethernet links to the first GbE gateway of the system. It is usually the first concentrator board. At that stage the Ethernet packet is transcoded into custom, inter-electronics communication protocol TrbNet, originally developed for the system presented in this thesis, and is transmitted further to the endpoint with the use of optical TrbNet connection. The protocol offers three logical channels with following functionalities: readout request distribution, data readout and Slow Control.

Most of the programmable electronics components of the system, present a set of registers which can be grouped into two main groups: status and control. The status registers allow to check the state of different components in the module (e.g. transmitted bytes counter, received trigger signals counters etc.). This is used mainly for the monitoring process. The control registers are used in order to alter the default configuration of the modules. Through those registers, one can for example enable or disable data channels, set the thresholds on inputs etc.

The ability of remote control of the elements as well as wide spectrum of available functionalities through the registers is crucial for efficient use of the modules. The Slow Control System has to be designed in a modular way that facilitates the process of including new components and functionalities. This has to be assured by both the firmware, which exposes multiple registers and software that allows the configuration of those registers in a human friendly manner. As the DAQ systems are getting more complex, most of the monitoring, online statistics and configuration processes are being executed automatically by the Slow Control software, but has to give the ability for the operator to access those values at any time.

2.4 Off-shelf DAQ and Trigger Platforms

There are several commercial standards which were developed throughout the years, which unify some of the basic building blocks of DAQ and trigger systems. Such platforms provide standardized and modular solutions for building complex systems. They are based on a concept of a crate, which provides power, cooling and interconnection for a number of modules that can be inserted inside. Modules are designed to perform the basic functions of DAQ systems, like discriminators, digitizers or controllers. By connecting them together, one can build a complete data acquisition chain. The main advantage of such approach is that there are ready, out of the box modules, which are reliable and supported by the manufacturer. The drawback is that the modules are designed to work in a wide spectrum of applications and are not tailored for the exact type of detector or signal and have limited scalability possibilities.

The first and the simplest standard was introduced in 1968 by the U.S. Atomic Energy Commission. Nuclear Instrumentation Module (abbr. NIM) (18) standardizes the size, cabling, power supply and the backplane pinout. The modules perform very simple tasks like signal discrimination, coincidence, logic functions etc. Even though, they are not interconnected and cannot be controlled programmatically, they are widely used in some parts of modern experiments, which do not require the processing of digital data. For instance NIM modules can be successfully used for generation of low level trigger based on coincidence, thanks to its robustness and fast analog signal processing.

Low level functionality and manual operation of NIM modules led to development of a new solution which could be connected to computers. Computer Automated Measurement and Control (abbr. CAMAC) (3) was introduced in 1972 and was the first standard of readout electronics that could be controlled by computer, providing automation of the entire data collection process. Like NIM, standard describes the mechanical characteristics of modules, electrical standards and backplane pinout. CAMAC extends NIM functionality by communication features, which allows the transfer of digital data. The modules are working in a slave-master mode. Each crate should be equipped with a Crate Controller, a module that act as an interface with controlling computer and a bridge to access each module individually. The Crate Controller is called master while all the other modules are slaves. The system can be easily scaled for higher number of creates by the use of module called Branch Highway, which allows to connect several creates together.

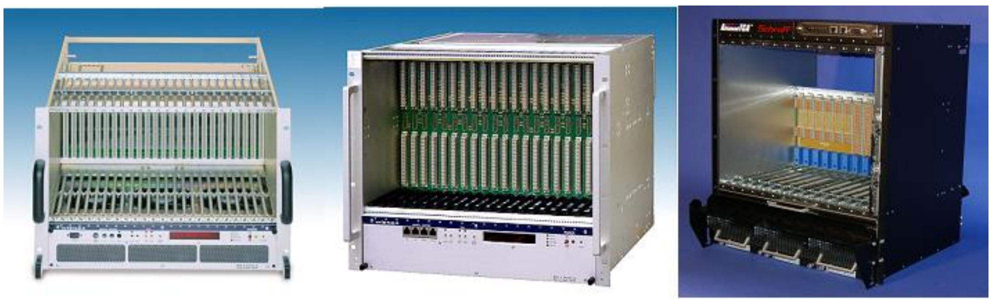


Figure 6: Three different create standards. From the left side: CAMAC, VME and ATCA. [Pictures from www.wiener-d.com, www.pentairprotect.biz/en/emea]

Fast growth of scales of physic experiments and of computing power, quickly exceeded the bandwidth offered by the CAMAC standard, which became obsolete. Its place was taken by developed in 1981 Versa Module Europa (abbr. VME) bus (2). This standard has also found its place in industry which resulted in development of a high number of modules implementing versatile functionalities that could also be used in science. Faster bus allows transmission of larger amount of data (40 MBps), thus increasing channel density that could be processed by a single module. In addition, in order to assure backward compatibility, a module that interfaces to CAMAC system was developed, which helped in reuse of legacy modules (Figure 6).

Even though all of the mentioned standards are still being used in most of the experiments as some parts of DAQ systems, the scale of detector systems and the complexity of required computing induce the development of new solutions. Up to now there is no modern standard platform on which the entire readout chain could be built. One platform that found its way from the communication industry into physic experiments is the Advanced Telecommunications Computing Architecture (abbr. ATCA) (19). It offers very high backplane connection speed in full mesh architecture, which means that each module in crate can directly communicate with any other modules. The main drawback is that as the main purpose of the standard was different, there are not many modules with functionality needed for DAQ systems available on market. The ATCA standard is used more as platform for custom built electronics, providing mechanical support, power supply, cooling systems and interconnectivity of modules.

2.5 Models, Requirements and Challenges

In an ideal case, data acquisition system in conjunction with trigger system is supposed to process all the events happening in detectors. The reality imposes a number of conditions which limit the amount of data that can be processed and stored for later analysis. All factors, starting from detector response shaping time, time needed for digitization, network throughput and buffering capabilities define time needed to process a single event, which in turn determines dead time of the entire system. On the other hand, only events considered as candidates for containing interesting data should arrive to the end of the readout chain. That is important in order to minimize amount of the storage needed and also to facilitate the work for physicists who will analyze the data by online filtering, thus reducing data volume.

2.5.1 Models

The way the systems operate can be categorized as pull or push type of systems. In system operating in pull mode, the data is requested to be read out by the trigger system. The data is buffered at an early stage and some parts of it are being analyzed by trigger algorithms. Positive trigger signal is propagated back and the event parts are retrieved from buffers. This architecture is based on very fast trigger mechanism and is useful in conditions where most of the events can be easily rejected as noise. In opposition, in push architecture, collected data is directly or on fixed rate, transferred between data acquisition stages. Each stage introduces higher level of filtering mechanism, reducing amount of data that arrives at the end of the chain. This solution is efficient in case the decision about the quality of an event is more complex, requiring complicated analysis or operating on larger detector areas. Depending on the data qualification algorithms, mixed systems also exist, where initial phase of acquisition is accomplished in one type of architecture and further processing is realized by the other type.

Designing architecture of the data acquisition systems is a complex balance between the capabilities of readout electronics, online data analysis computational complexity, possible network infrastructure and the requirements imposed by physicists on wanted data quality. DAQ system concepts should be simulated prior to taking final decision about their architecture. Models describing data flow in a system are based on mechanisms specified by queueing theory (20). Single fragments of the entire readout chain can be treated as queue instances represented in Kendall notation as:

$$1 / 2 / 3 / 4$$

Where [1] stands for the time distribution of incoming elements into the queue, [2] is the time distribution of processing an element by a single service, which amount is represented by [3]. The way elements from the queue are selected to be processed [4] can be either one of: FCFS (First Come First Served), LCFS (Last Come First Served) or randomly selected. In case it is not specified in the formula, FCFS is assumed by default. For instance a primary task of a collector board is to collect data from several links, encapsulate with some headers and forward to further stages on a single link. For a single event, the data fragments are arriving on the input links with different time offsets [1], forming a queue. The process which takes that data and forms an outgoing packet is a single service [3]. As all data fragments can have various sizes, the time needed for its processing is also a variable [2]. The process forms a packet by taking input fragments as soon as they arrive, thus the queue type is FCFS [4].

2.5.2 Requirements and Challenges

As most of the electronics are nowadays equipped with programmable devices, the main weight on the capabilities and efficiency of the system is imposed on the firmware and software development. The detectors are able to generate enormous amount of data, thus designing a DAQ system is always about finding a proper balance between the achievable throughput, the measurement resolutions and the final amount of data with reasonable signal to noise ratio.

Designing a data acquisition system is extremely difficult. The experiments take sometimes dozens of years to develop. Taking into account the pace of evolution of technology, some functionalities that are not available at the beginning of the R&D phase can become common throughout the years. That is the reason why, the programmable devices are widely used in the DAQ electronics. It is easier and cheaper to develop new firmware than to produce new electronic components.

It is also almost impossible to predict how the entire project will develop over the time. Thus the architecture of the DAQ system has to be extensible and flexible enough in order to include new modules and functionalities introduced during the operation time. To get most of the detector systems, they have a diverse physic program foreseen to perform over the years. It is important to have a DAQ system that can be adjusted for efficient data collection under different conditions imposed by the specific experiment.

The first constraints about the data quality are submitted by physicists and their physic goal. This depends on the type of collision they want to register and the type of the detectors used in the system. Different intensities derive directly the hit rate on the detectors. The wanted type of collision can be selected with a trigger system and reduce the rate of the events that are processed by the system. The range of the accepted events rate is the first aspect that has to be taken into account. The second aspect is the desired digitalization resolution and channel density. More precise measurement results in larger amount of digital data that has to be processed by the system.

In order to achieve the maximum available data throughput in the system, many of its components have to be properly designed and implemented. At first the digitized data is captured in buffers and waits for the readout request. The efficient use of available memory resources is crucial for minimizing data loss and so called event mixing when it comes to high rates. Data loss can happen when the defined buffers overflow, which can be a cause of backpressure generated by busy subsequent components in the system. Thus, real time processing with minimum latency is required to be implemented on critical paths. To avoid event mixing, a situation when parts of different events are combined together and memory is organized in a form of queues, usually FIFOs. Synchronized around the entire system write and read operations on such memory blocks allow to keep collection and reassembly process in order.

3 PROGRAMMABLE LOGIC DEVICES

The main difference between standard processor and a programmable logic device (abbr. PLD) is that their internal structure is not fixed. Taking standard CPUs as an example, one can run a program which will be executed on the internal infrastructure of logic components. Programming logic devices means describing and defining this infrastructure. Instead of fixed structure, PLDs consist of arrays of general logic blocks, which can be configured individually. Using a Hardware Description Language (abbr. HDL) and a dedicated compiler, the abstract logic functions described by the developer are translated into series of logic blocks configured and interconnected accordingly.

There is a lot of various programmable logic devices available nowadays on market. They can be classified under different aspects like reprogramming, available resources, configuration holding etc. We will focus only on two main families of devices: Complex Programmable Logic Devices (abbr. CPLD) and Field Programmable Logic Devices (abbr. FPGA), which are most commonly used PLDs categorized as High Capacity Programmable Logic Devices (abbr. HCPLD). The aspect that distinguish those two is the non-volatile memory of CPLDs and the amount of resources. The internal structure of the chip is constructed differently, using the so called "sea of gates" instead of configurable logic blocks found on FPGAs, allowing once loaded configuration to remain after power cycle. All logic blocks building an FPGA lose their configuration and need to be programmed after powering up. On the other hand, FPGAs deliver much more complex hardware features inside the chip and have the amount of resources higher by several orders of magnitude. The programming of FPGAs can be automated by installing a dedicated RAM memory holding configuration, which is loaded on startup. Those two aspects separate the use cases for both types. CPLDs are mostly used for simple tasks with fixed functionality like interfaces between devices or implementing glue logic. Huge amount of resources on FPGAs gives the possibility of implementing complex functions and algorithms, hence they are used as system controllers and data processors.

Miniaturization and introduction of 20 nm technology process strongly increased the amount of logic resources that could be packed into a single device, while keeping reasonable size, price and power consumption. Large jump in logic capability attracted a lot of customers from fields like networking technology, data processing, military and of course high energy physics. They no longer offer logic gates only but also complex hardware elements like high speed transceivers, memory blocks, Digital Signal Processing (abbr. DSP) blocks and even built in microprocessors platforms like PowerPC or ARM, transforming FPGAs into System On Chip (abbr. SoC) solutions.

High performance capabilities, reconfiguration and relatively low cost are the key reasons why FPGAs are often chosen to equip electronics in data acquisition systems. The chapter starts by covering the nature and structure of FPGA devices. Then the basics of programming language are presented, followed by the methodology of working with this kind of programmable devices. The chapter is closed by an example of comparison between different computational platforms.

3.1 FPGA Device Structure

The key building blocks of FPGA devices are Slices, grouped by two or more, depending on a specific device model. Grouped Slices are forming Configurable Logic Blocks (abbr. CLB), which in turn are arranged into large arrays. Each Slice can be configured to realize a given, basic logic function like AND, OR etc. of its inputs and present the result on its output. Depending on the complexity of the FPGA, the construction and the components included in a single Slice differ. They all share some basic features (Figure 7) though, which are Lookup Tables (abbr. LUT), multiplexers and Flip Flops. Lookup Tables, also called Function Generators store logic functions which are selected through configuration process, Flip Flops realize synchronization with the clock and the multiplexers select appropriate outputs. This set of essential components can be extended by additional adders, RAM blocks or carry logic, which can be shared between Slices. Each CLB is interconnected with its neighbors, which gives the possibility of implementing any kind of logic functions by configuring each element accordingly.

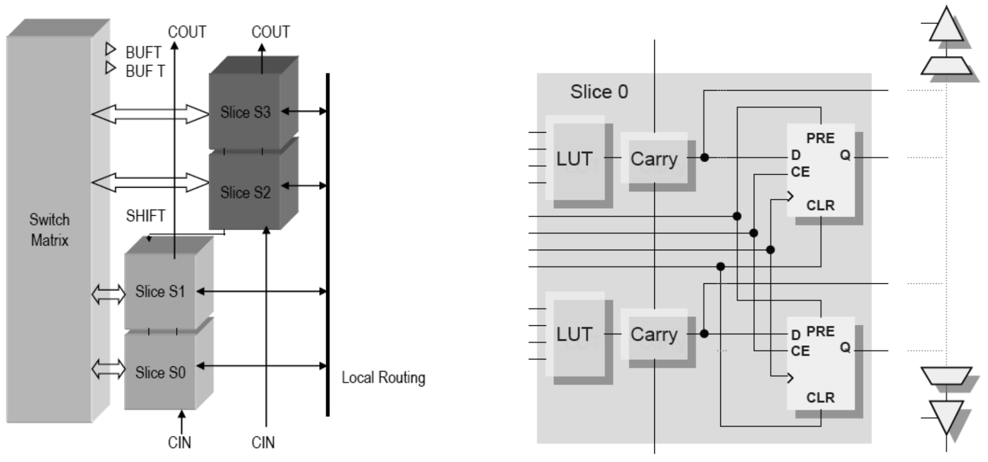


Figure 7: CLB (left) of Virtex II FPGA containing four Slices, each Slice (right) is built out of two 4-input LUTs, carry lines and flip-flops. [xilinx.com]

CLB can process several input signals, through selected basic logic function and deliver the outcome signal on its outputs. In order to supply input signals from FPGA pins to its interior, there are special IO Blocks. Each general pin can be configured as input, output or tristate port. In order to enable such functionality, pins are surrounded with clocked registers, which can be adjusted to the selected type of electrical standard. A pair of pins can run in differential standard like LVDS, which is used in cases where high speed and noise secure transmission is required. Configured as input, registers act as buffers registering the state of input signal at the clock ticks. In case of output ports, the registers drive the pin with a clocked signal provided as its input. More advanced FPGAs present more complex IO blocks supplying additional functionality like Serializer-Deserializer (abbr. SERDES) modules or delay blocks. SERDES facilitates transmission of entire data words over single pins by dividing the word into a sequence of bits. In the other direction, it gathers several bits together and presents a recovered word to FPGA internal logic. Those modules can run in Single Data Rate (abbr. SDR) or Dual Data Rate (abbr. DDR). The difference is in the way that single bits are presented at the output (or registered on the input). In case of single rate, the bits change at the rising edge of the clock and in case of dual rate, the change occurs both at the rising and at the falling edge of the clock signal. The delay block helps adjusting the input data signal to the clock signal edges, in order to register its state exactly at the moment of clocks state transition.

Most applications require logic to run synchronously in respect to some clock signal. The signal can come from various sources like external oscillator connected to an input pin or clock recovered from incoming data stream. The clock distribution to all the Flip Flops in Slices is realized by special routing nets. In order to allow the different parts of logic to run at different clock frequencies and to optimize the usage of resources, there exists many kinds of clock routing net types in a single FPGA chips. Inside an average device, user can find several nets call Global Clocks, which should be used by general clocks needed to be distributed all over the device. High frequency applications can suffer from delay introduced by long distribution paths. Regional Clocks are routed only within some defined regions, usually a single bank. They are efficient for logic interfacing with external devices, connected to pins located at a given bank. Single design often contains several parts of logic, running with different clock frequencies. Having a single clock signal, it is possible to generate a number of clock signals with various frequencies by the use of Digital Clock Managers (abbr. DCM). It is very important for a developer to properly combine those parts together and secure all the clock domains crossings.

Data processed by an FPGA needs to be buffered for transmissions or stored for further manipulations. There exist several solutions which can be applied for that purpose. All FPGAs provide some amount of internal memory in form of Memory Blocks placed in specified locations between CLBs. Very fast to access (one clock cycle to retrieve an entire word) and easiest to use is the main source of memory, but it’s capacity is limited to few or dozen MB per average device. Another way to store data is the use of Slices in CLBs as memory cells. One can consume general resources and convert it into memory as a tradeoff between the amount of resources available for implementing logic and the amount converted to memory. In case the capacity is the key aspect, there is no other way, than to access an external memory chip or card. It’s the most difficult to design and implement solution, also much slower than the use of internal memory. The internal memory can be configured in various ways. Access type defines if the memory block will be used as standard memory (write a data word under a specified address cell) or as a queue (written elements are added at the end of a queue). The access operations can be realized using one or two ports. In single port memory, write and read operations are executed synchronously to one, main clock, while in dual port, each operation can be clocked with a different frequency. This is especially useful for passing data through different clock domains.

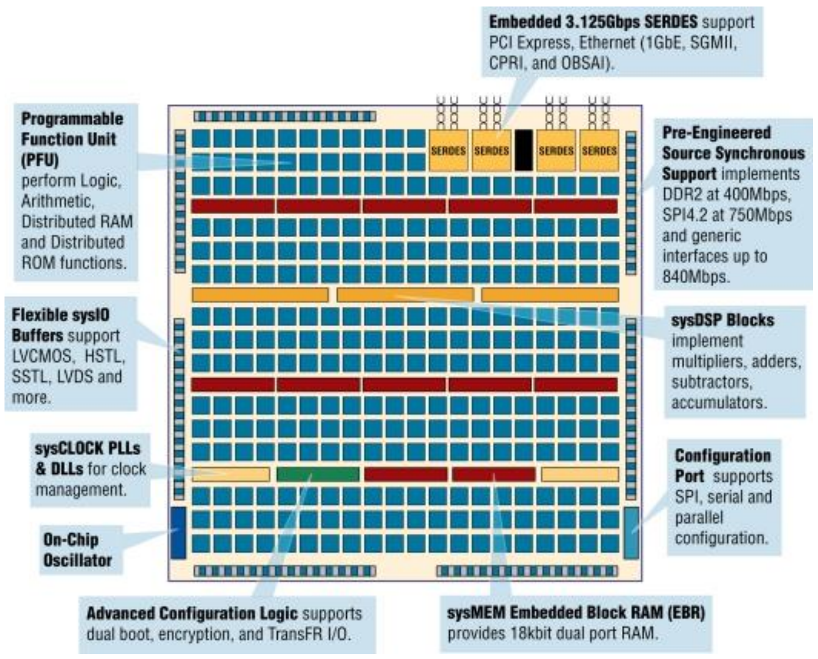

Embedded 3.125Gbps SERDES support PCI Express, Ethernet (1GbE, SGMII, CPRI, and OBSAI).
Programmable Function Unit (PFU) perform Logic, Arithmetic, Distributed RAM and Distributed ROM functions.
Pre-Engineered Source Synchronous Support implements DDR2 at 400Mbps, SPI4.2 at 750Mbps and generic interfaces up to 840Mbps.
Flexible sysIO Buffers support LVCMOS, HSTL, SSTL, LVDS and more.
sysDSP Blocks implement multipliers, adders, subtractors, accumulators.
sysCLOCK PLLs & DLLs for clock management.
Configuration Port supports SPI, serial and parallel configuration.
On-Chip Oscillator
Advanced Configuration Logic supports dual boot, encryption, and TransFR I/O.
sysMEM Embedded Block RAM (EBR) provides 18kbit dual port RAM.
SERDES
SERDES
SERDES
SERDES

Figure 8: Example of Lattice ECP2M FPGA internals, IO blocks can be found on the edges, blue blocks are CLBs and red elements are memory blocks [latticesemi.com]

Presented above components (Figure 8) are essential for all applications, but modern FPGAs feature many additional hardware elements that extend their functionality. One of such elements commonly found on nowadays devices is the Digital Signal Processing (abbr. DSP) block. Similarly to memory blocks, the DSPs are distributed inside an FPGA and can be used to perform intensive computationally operations. In order to provide data to or transfer processed data out of a FPGA, it is equipped with many Gigabit Transceivers. Standard FPGA pins are capable of driving signals up to few Gbps. Gigabit Transceivers are prepared to handle data transmission in current communication standards, reaching up to 32 Gbps for most advanced devices. Those transceivers enable full duplex communication in standards like 1/10 Gigabit Ethernet, high speed PCIExpress and others. Those link layer protocols can also be found as hardware elements, built in FPGA fabric. Another interesting device found inside chips are microprocessors. Full featured PowerPC or ARM cores can be accessed from internals of FPGA and can be used to perform high level calculations on data received and pre-processed by standard logic Slices.

3.2 Programing Language – VHDL

It is crucial to understand the difference between writing executable code for standard CPUs and writing a code that represents physical architecture of a system. FPGAs do not execute any commands but gets configured to process data from input pins and present the result on output pins. This is the reason why the family of languages used for developing logic is called Hardware Description Language (abbr. HDL). Initially they were used for describing entire printed circuit boards, with all equipped electronic components and interconnections. First languages were used to automate the process of developing new electronic systems. Introduction of ASIC and FPGA devices required a way of configuring them by the developers. At the beginning they had to be selected and connected by hand logic gates, creating schematics which were converted into stream of bits representing the configuration of CLBs inside FPGA. In order to elevate the abstraction level and automate this process, some of HDL languages were adapted, from which two most popular are still in use: Verilog and Very High Speed Integrated Circuits Hardware Description Language (abbr. VHDL). As the language used for implementing solutions, which are subject of this thesis, we will focus on the second one.

VHDL focuses on description of logic circuits in a form single *entities* that realize logic functions of signals on its input ports and provide the result on the output ports. The set of ports is called *an interface*. Such entity can be easily pictured as an electronic component with its pins as interface and some internal logic, with the difference that in case of FPGA this component is just a logical module instantiated in an array of CLBs. Module called *top entity* is the main component that represents the entire FPGA (Figure 9). Each port on its interface is mapped into a physical pin of the chip. All other entities can be instantiated inside top entity, what makes the hierarchical structure of the VHDL code.

A single entity describes the relation between its input signals and output ports. Such relation can be basic (e.g. logical conjunction of several inputs presented on one output port) or complex, where input signals pass through logic functions, embedded hardware FPGA components or instances of other entities. Apart from interface ports, each entity can define a number of internal signals, local to its instance.

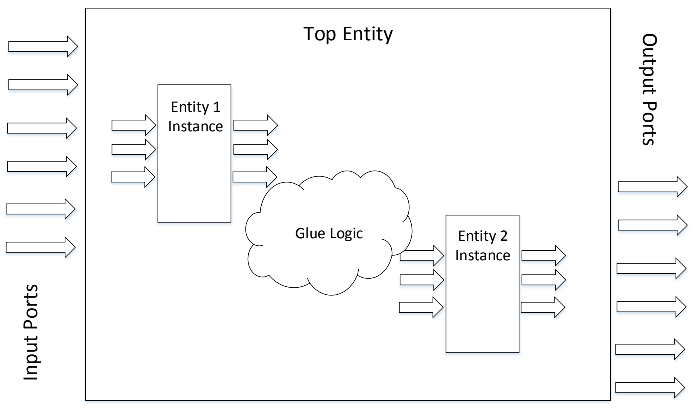


Figure 9: Example of structure of a top entity with ports and instances of other components in the interior interconnected by some logic.

Relations between signals are written as statements in a form of assignments or processes. Assignments are asynchronous basic logic functions and connections between signals. Asynchronous logic is realized without the use of Flip Flops, which means that in case when one signal drives another, the second one will change its state after a short delay, which is equal to the sum of delays introduced by electronics components on the path between CLBs in which the signals are stored. It's important to notice that such assignment is not a command that is executed once, it rather describes a hardware path which will exist in the FPGA and the dataflow. Behavioral logic requires a code structure called a *process*. It's a structure which helps encapsulating several assignments under complex conditions and introducing helper local variables. Similarly to an assignment, logic included in several processes is performed in parallel and describes relations, so it's not executed once.

Each statement describing logic of an entity, whether it is an assignment or a process is working in parallel in respect to other statements in the entire project. Such logic is a stateless automata. In order to implement advanced processing algorithms, there is a need to introduce Moore or Mealy Finite State Machine (abbr. FSM) mechanism. In a single FPGA device the logic can be a mix of synchronous/asynchronous parts and direct or controlled by FSM logic. Finite State Machines are essential elements of controlling the dataflow and the processing procedures (Figure 10). They define a set of states in which the logic can operate. In each state the logic is realizing a specified task. It is a way of dividing and sequencing the whole procedure into a number of steps that need to be taken. Each state has well defined transitions to other states. Starting from the first state, the transition conditions are checked at each clock cycle. If any condition is proven positive, the actual state is changed, otherwise the actual state remains the same and the procedure continues to perform the task described by it. VHDL provides conditional statements like *if* or *case*. Those statements are translated by the compiler into series of comparators and multiplexers configured and connected in CLBs. The nature of FPGA devices allow to instantiate a number of FSMs inside a single design, all of them will run in parallel, steering the work of some parts of the logic. This aspect can be visualized by the example of the design of a network switch. Each network link should be handled by the logic independently. Hence in the design, a module for controlling a link, having an FSM which defines the states of link, can be instantiated as many times as there are links. All instances will run in parallel allowing the processing of data on all the links at the same time.

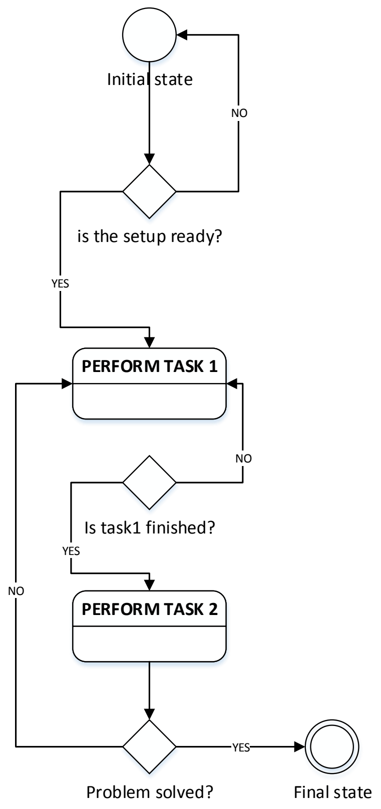


Figure 10: Example of a state machine composed of 4 states including initial and final states. The transitions between states happen in case defined conditions are met.

There are a lot of ready to use components, developed by the FPGA manufacturers, third party companies and open source community, which can be instantiated in a design. Intellectual Property Cores (abbr. IP Cores) are building blocks from which an entire design can be built with addition of user logic. All FPGA hardware elements like memory or DSP blocks, gigabit transceivers etc. are accessible to the developer via instantiation of appropriate core. Most standard functions can be found as IP Cores, especially when it comes to communication protocols. FPGAs need external devices in order to receive data to process or to transfer results out. That's why the knowledge of modern data transmission standards and protocols is required in order to build an operational design.

3.3 Design Flow and Methodology

Developing FPGA logic is a different process than writing high level procedures that are executed sequentially on a CPU. The fact that the written code is translated not into a sequence of low level assembler commands but into a representation of basic electronic components included in an FPGA, requires a totally different approach. The developer needs to know the internal structure of the device for which the logic is being written as well as how the compiler will translate the code into configuration of Slices in logic blocks. The compilation is a very sophisticated set of processes, executed one by one in order to create a final bit file containing configuration of each CLB in a target FPGA device.

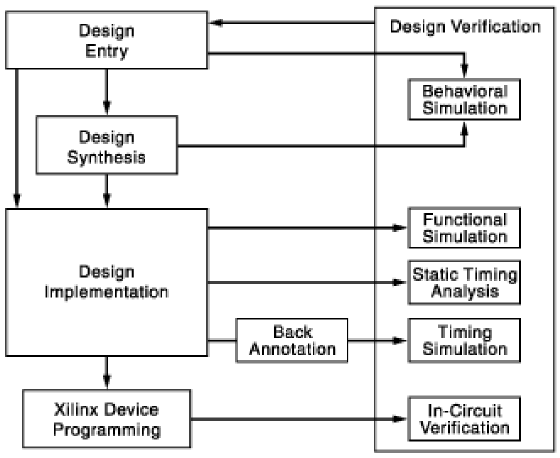


Figure 11: Detailed design flow. Firmware development requires multiple iterations of the cycle consisting of several levels of compilations (left column) and verification of the results using different tools (right column). [xilinx.com]

Efficient FPGA design process flow presented on Figure 11, requires several steps to be performed before configuring the device. The entire code in a design contains parts which are platform independent and parts which are strongly tied with an actual target FPGA model. That's why, as the starting point, the developer needs to get to know the exact FPGA structure as well as all the external devices mounted on the PCB which will be used in the design. Knowing these details, one can define the interface of top entity that is mapped into FPGA pins connected to peripherals. The mapping is realized by including a constraints file, which pairs the port of the interface with a pin of the FPGA.

The code in the source files needs to be compiled into representation of basic logic components like gates, multiplexers, adders etc. This process is called *synthesis* and is independent from the target device. The result of synthesis is a *netlist* which can be displayed graphically as a schematic of the translated logic. Compiling the code into a netlist is a straight forward procedure of converting VHDL statements into logic functions and connecting registers. This process introduces different levels of optimization in order to remove redundant or unused parts of the logic. At first all the registers which outputs are not connected in the netlist are being removed. Second step is the minimization and simplification of generated logic functions, a process similar to Karnaugh method (21). The last step is the merging of redundant logic parts.

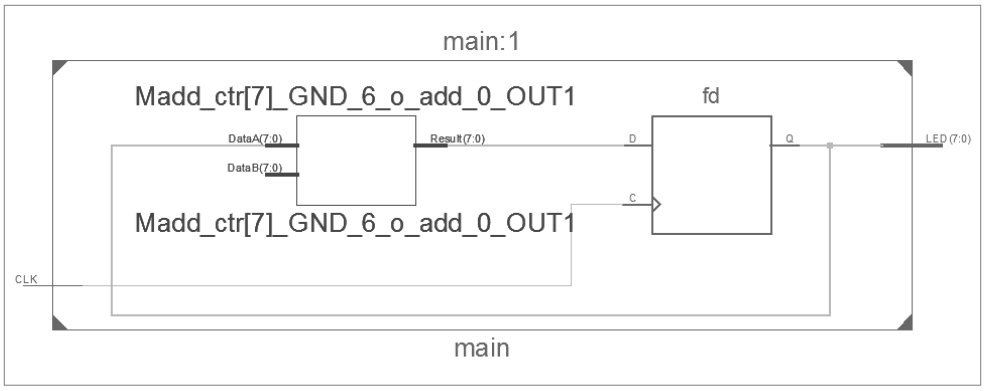

main:1
Madd_ctr[7]_GND_6_o_add_0_OUT1
fd
DataA(7:0)
DataB(7:0)
Result(7:0)
D
Q
LED(7:0)
C
Madd_ctr[7]_GND_6_o_add_0_OUT1
CLK
main

Figure 12: Netlist of a counter running on input clock CLK and presenting the output on LEDs, the code has been synthesized into an adder module and a flip flop with feedback connection.

The generated netlist (example presented on Figure 12) is a starting point for further compilation steps but also for the first stage of debugging. The correct design flow suggests running behavioral simulation prior to implementing design. Behavioral simulation helps finding bugs in the logic functions and verify if the translation done by the synthesizer is correct. Simulation (sample output presented on Figure 13) can be run for selected component only or for the top entity, which means the whole logic in the design will be processed. As the VHDL components describe the manipulation of input signals, the simulation needs some kind of description, how those input signals work. Such description is called a *stimulus vector* and is included in a *testbench*. It is also a VHDL component, which has an instance of a component under test and some processes that describe each input port of that component. In a testbench, one can use VHDL statements, which are not synthesizable by the compilation process. Those commands are useful in description of the signal in function of time. Commands like *wait for [time unit]* will be compiled by the simulation tool as a pause between executing the next statement. The use of *wait* commands changes the standard VHDL process into a set of commands executed sequentially. What is important to mention is that the testbench process will be only executed by the simulation tool, hence on the standard CPU. The operating system running on a CPU has the means of measuring time, that's why the wait statements are possible to use. There is no such features while implementing a design on an FPGA, which is the reason why such statements are not synthesizable. The simulation will feed the described input signals to the component under test ports and calculate its response during a specified period of time. The calculation occurs with a time step (usually 1 ps), evaluating the state of all components, logic gates, flip flops etc. included in the design. In order to verify the behavior of the component during 1 ns, simulation tool will recalculate the state of all elements 1000 times, which is a large computational effort. The key to a successful simulation is the preparation of stimulus vector. In order to find potential bugs, a well-designed vector should contain some randomized parts, which would cover all the possible states of input ports. Only when the response of the component under test is correct for all those states, the component can be qualified for further implementation steps.

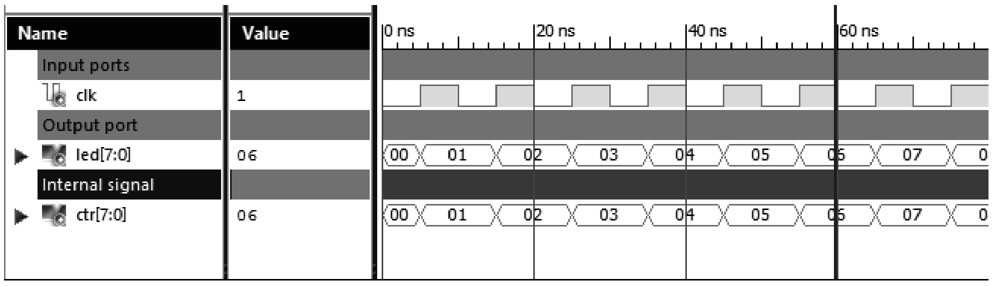


Figure 13: Example of 80ns simulation output result. Stimulus vector describes the clock ticks on the input port, for each tick an internal signal is incremented and its value is presented on the output port.

When the results of simulation prove the correctness of designed logic, the developer can proceed with design implementation. Because the implementation is a process of fitting and placing the netlist in an FPGA, this process is strongly dependent on the target device. That means that the result can be used only on a device type specified in the project settings. First step is the translation, which converts the netlist generated by synthesis into set of elements included in the actual device. This stage is needed as the synthesis, being platform independent process, can be realized by software from different providers, while implementation software can only be provided by the manufacturer of the FPGA. When the translator converts the general netlist into a set of components available in a target device, the processes that will place those elements around the chip are *mapper* and *place and route* (abbr. PAR). At first the mapper is recognizing all the components included in the design as hardware elements in the FPGA and tries to locate them. The placing can be forced by the developer using statements in the constraint file, which locks components to specified location in the FPGA array. Not specified elements are distributed all over the device by some placing algorithm. The algorithm tries to create the shortest paths possible between all the configured CLBs. Once placed, logic is then process by PAR. Its task is to adjust the placement of components in such a way that all timing constraints are met. The timing constraints are essential in order to properly communicate with external peripherals (typical situation presented on Figure 14). The user defines the frequency of the clocks used in the design as well as setup and hold times of certain signals. The delays introduced by the routing of elements inside the FPGA can violate the timing specifications of protocols used between the chip and the peripherals. Those specifications need to be well defined by the developer in the constraint file, then the PAR algorithm can place the logic in such a way that the calculated delays are within some declared margins. The operation of PAR algorithm can be adjusted in terms of area covered by the logic or most timing efficient placement.

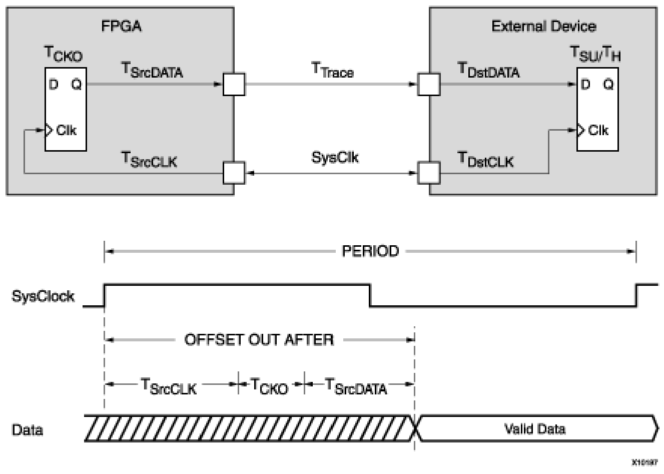


Figure 14: Visualization of timing requirements for output signals. Valid data offsets in respect of clock signal provided to an external device has to be constrained for proper operaton of PAR algorithm and production of design with secured timing. [xilinx.com]

The output of the entire implementation process is a detailed description of each CLB and any other hardware component configuration in the target FPGA structure. Successful PAR run assures there are no violations of timing on essential data paths and all the design components are placed in valid locations. The verification of timing aspects of the design can be done by checking the generated reports describing the longest paths and potential problematic connections. This information can also be used for timing simulation. The simulation performed after synthesis does not take into account any timing issues of the design (assumes zero delays), it works only on Register Transfer Level (abbr. RTL) level, which is useful for validating correctness of logic functions. Timing simulation process, extends the behavioral simulation by adding all the calculated during PAR timing information.

Iterations between simulation and implementation processes are needed in order to achieve timing closure of the design. Once the logic is verified and correctly implemented, one can generate a *bit file,* which is a binary file containing configurations for each element in target FPGA. It is a translation of PAR results into binary format. A bit file can be then loaded into an FPGA. Once the device is programmed, the developer has to verify by his means if the wanted functionality is properly executed. Last years, major FPGA manufacturers developed a system of in-circuit verification, which allows to check the operation of the logic running on a programmed FPGA device. The integrated logic analyzer can be used to display the selected internal signals in a function of time, triggered by some defined condition or manually at any time. This is the last step in debugging the design. In case a wrong way of operation was found which needs a change in the logic, the entire design flow cycle has to be repeated from the first step. As the place and route algorithms are not deterministic, often guided by pseudo random number generators, each run and more importantly each change in the entry point of the design can result in unpredictable implementation results. That's why it is so important to properly constrain the design and verify each step of the procedure with the simulations and reports lecture.

3.4 Comparison to CPUs and GPUs

High-performance computing is a fast evolving branch of hardware development imposed by demands coming from various markets like science, banking, consumer electronics etc. Technology and physics limitations reached in transistor miniaturization in production process, forced the questioning if the Moore's law (22) is still applicable. While single cored CPUs reached their peak performance achieved through core complexity and clocking frequency, products combining multiple cores in a single chip emerged. Multicore approach introduced new possibilities in data processing but also new software and firmware constructs that could efficiently make use of such architectures. In response to specific demands, three main families of specialized computing units are now existing on the market. Multicore CPUs are general purpose processing devices, with several complex cores. GPUs feature few orders of magnitude more cores, which are optimized for complex arithmetic operations. PLDs are "empty" devices, which allow the developer to design its internal architecture according to specific requirement.

3.4.1 Architecture

Multicore CPUs (Figure 15) are encapsulating several processors in one chip together with additional memory and controlling mechanisms. In consumer products it is often to find an additional graphic processor. A single core is an independent processing unit with its own scheduler, registers, Arithmetic and Logical Unit (abbr. ALU) and cache. The communication and synchronization between cores is achieved by shared memory and the CPU controller. Each core is sequentially processing instructions from Complex or Reduced Instruction Set (abbr. CISC and RISC) defined by the chip producer. The bottleneck of fast clocked cores is the memory access. That's why the dependences between threads run in parallel on separate cores should be minimized in order to achieve maximum computing efficiency.

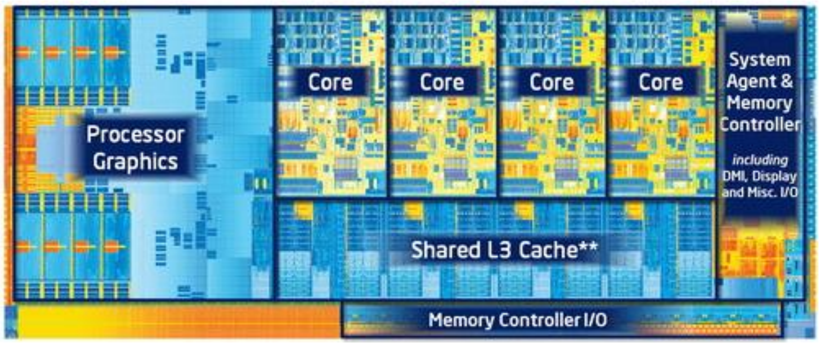


Figure 15: Internal die architecture of Intel Core i5 3570K processor. There are four visible cores, a separated graphic processor unit, cache memory shared between all components and various IO controllers. [intel.com]

Different architecture is presented by GPUs (Figure 16). High number of smaller, specialized cores is targeted to perform similar operations multiple times. Single Instruction Multiple Data (abbr. SIMD) approach is based on delivering different data to a set of threads, each one executing same kind of instructions. In case of GPUs, a single core consists of many ALU units, equipped with local cache. Those ALU units are grouped into so called *warps,* and they work on single instruction with separate data sets. It is an advantage for processing fine grained independent problems. In order to assure best performance of computation, the amount of operations distributed for each thread in a single warp should be even, which is a programmatic challenge. The software is divided into parts performed by GPUs and a part executed on CPU, which prepares and delivers data to the GPU platform.

Programmable Logic Devices (Figure 17), as described at the beginning of this chapter, are left for the developer to design their internal architecture. Large arrays of configurable components, together with additional hardware infrastructure can be configured to perform specified tasks. The amount of resources available in current devices, allows parallelized and pipelined data processing in real time. Lack of predefined architecture and set of instructions allows to design highly optimized solutions. Low level, basic, bit-wise operations provided by logic blocks are inefficient for floating point operations, which is the limiting factor in use cases of PLDs in computing.

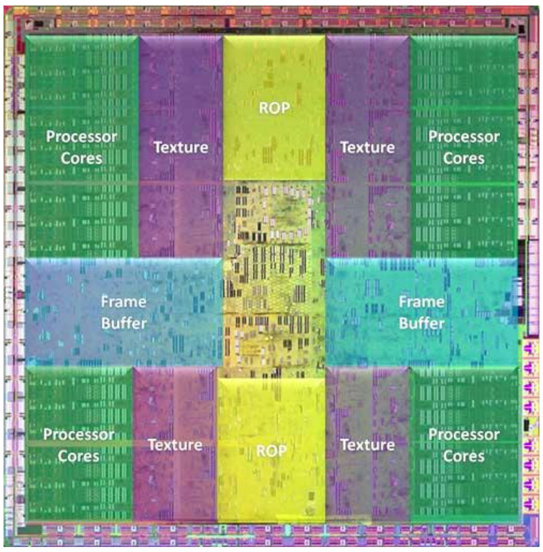


Figure 16: Internal die architecture of Nvidia GeForce GTX 280 graphic processor unit. There are four visible arrays of processor cores as well as buffers and memory components. [Nvidia.com]

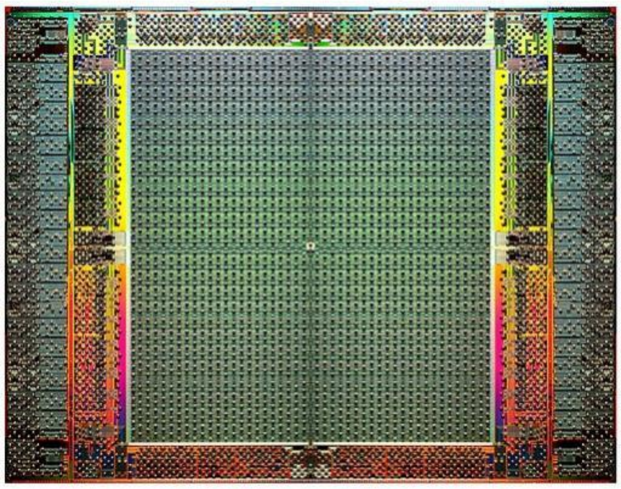
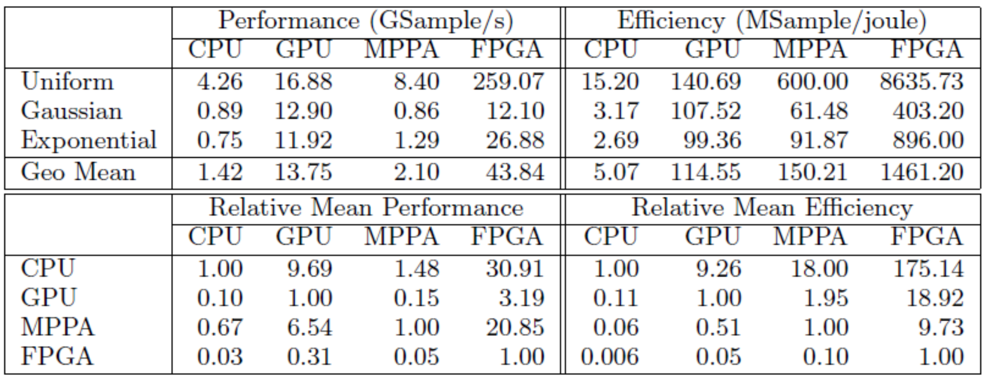


Figure 17: Internal structure of Altera Stratix FPGA. Most of the device is occupied by array of Slices. Additional resources like RAM blocks, DSP blocks are distributed around the device while the high speed transceivers are located at the edges. [altera.com]

Fundamental differences in the architecture of those three types of computing devices categorize them for being applicable for specific kind of tasks. Multicore CPUs are efficient for solving complex problems which can be divided into few functional threads with mixed instruction sequences. The GPUs best perform calculating very high number of same operations with different datasets on each thread. The structure of PLDs offers unique performance for real time applications and online, parallel data processing with limited arithmetic operation set.

3.4.2 Example Application - Random Number Generator

In depth performance comparison was studied in many available publications. Important aspect is the selection of an algorithm that is possible to be implemented on all kind of devices. In High Energy Physics, Monte Carlo simulations are the basic tool for preparation for experiments. Those simulations are well parallelizable problems which are strongly based on Random Number Generators (abbr. RNG). The quality of those generators affects the simulation results. The publication (23) is covering different RNG algorithms and their implementations on CPUs, GPUs and FPGAs. Several algorithms for uniform, Gaussian and exponential generators were implemented and compared in terms of performance, which is number of samples delivered per second using all resources possible on selected devices. FPGA architecture presents an advantage of performing bit-wise operations simultaneously on all Slices available. This feature is especially useful for uniform algorithms where no high level arithmetic operations are required, hence the entire logic can be implemented on Slices structure. For other algorithms, the limiting factor is the need of DSP and RAM blocks, which amount is limited and data access introduces additional latency. Implementations on GPUs require wise selection of algorithm in order to take advantage of SIMD and warp thread structure. To efficiently distribute the work over the available threads, it is important to take care of equalizing algorithm paths in such a way that the amount of warp threads waiting for the slowest one is minimized. CPUs take advantage of well defined, developed over long period of time and verified algorithms, which are executed on devices that are being run at very high clock frequencies. The very low number of parallel threads in comparison to other platforms, definitively makes them the slowest solution. It is important to notice, that in those tests the time needed for data transfer and the communication latency are not taken into account, just the raw number of generated numbers per second.



	Performance (GSample/s)				Efficiency (MSample/joule)			
	CPU	GPU	MPPA	FPGA	CPU	GPU	MPPA	FPGA
Uniform	4.26	16.88	8.40	259.07	15.20	140.69	600.00	8635.73
Gaussian	0.89	12.90	0.86	12.10	3.17	107.52	61.48	403.20
Exponential	0.75	11.92	1.29	26.88	2.69	99.36	91.87	896.00
Geo Mean	1.42	13.75	2.10	43.84	5.07	114.55	150.21	1461.20

	Relative Mean Performance				Relative Mean Efficiency			
	CPU	GPU	MPPA	FPGA	CPU	GPU	MPPA	FPGA
CPU	1.00	9.69	1.48	30.91	1.00	9.26	18.00	175.14
GPU	0.10	1.00	0.15	3.19	0.11	1.00	1.95	18.92
MPPA	0.67	6.54	1.00	20.85	0.06	0.51	1.00	9.73
FPGA	0.03	0.31	0.05	1.00	0.006	0.05	0.10	1.00

Figure 18: Comparison of peak performance of random number generators on different platforms. (23)

The table above (Figure 18) shows that the FPGA implementations outperformed the other platforms, especially in case of uniform distribution. Similar performance to GPU is only in case of Gaussian random generator. GPU implementations present an order of magnitude faster sample delivery than an ordinary CPU. Also in terms of power consumption, the FPGA devices are the most efficient solution. The platforms used for the purpose of this measurement were: CPU – Intel Core2 QX9650, GPU – NVidia GTX 280 and FPGA – Xilinx xc5vlx330.

4 Data Acquisition System Architecture Based on Universal Readout Boards

The subject of this thesis is the entire concept of the trigger and data acquisition system based on universal readout boards Trigger Readout Board v2 (abbr. TRB2) and its successor Trigger Readout Board v3 (abbr. TRB3). Both solutions were developed in cooperation of many institutions, with key developers from GSI Helmholtzzentrum Darmstadt and Jagiellonian Univeristy in Cracow. The architecture of the TRB boards and details of their construction are given in Sections 4.3 and 4.4.

The main motivation was to design a platform consisting of hardware, firmware and software out of which readout systems of various scale requesting distributed data processing could be constructed. Thus, the key features of the TRB platform are: scalability, extensibility, flexibility and reconfiguration. Those points were achieved by the design of a system based on interconnected in a tree architecture, identical base modules with extension cards (Add-on boards) for various measurements (for example ADC). Such modules provide vital functionality like signal digitization with feature extraction, data transmission, control mechanisms and hub features, which allow to introduce further modules and expand the system. However one should keep in mind that extension should preserve the performance characteristics, while the number of modules increases. The additional modules are needed in order to introduce a higher number of input channels, a new type of measurement or data processing that is not provided by the base module. Such approach led to creation of a fully universal platform for a broad range of measurement systems, which is reflected by the amount of various users and high demand for this solution.

Detailed description of the system is presented in the next sections. By showcasing its architecture and main components, the concept of TRB platform is introduced. Both, the hardware and firmware are described with a strong emphasis on communication features, which were implemented by the author of this thesis and are of key importance for the system architecture. The in-depth analysis of the system performance can be found in Chapter 6, where two setups are evaluated.

4.1 System Architecture

The system is composed out of two main logical elements: endpoints and hubs. They both can exist as hardware components or as firmware modules. The endpoints are the elements in the system that perform signal digitization and some kind of processing, which results in generated data. Each endpoint has to be connected to one port of the hub, through which it can receive readout requests and control commands from the central controller. Through the hub connection, the endpoint can also transmit its data portion. Each module connected to a hub acts as an endpoint to that hub. Such approach masks the complexity of the system and returns it as a really scalable platform. One can always connect a new hub to an existing and therefore, open several additional slots for new endpoints (which, in turn could also be hubs etc.).

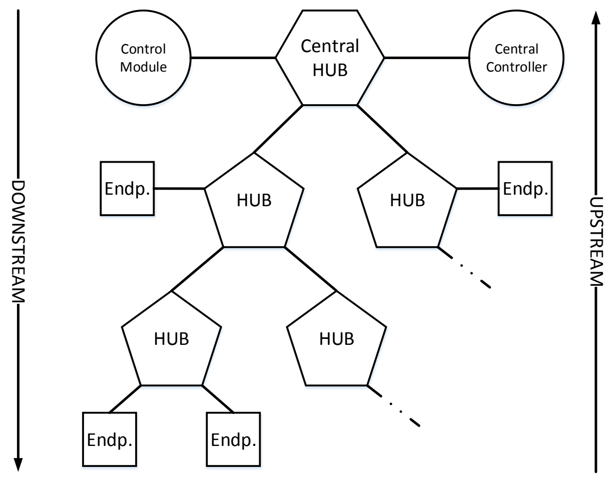


Figure 19: A schematic view at the tree architecture of the system. The central hub is a root of the tree and branches consisting of another layers of hubs and endpoints are derived. The central controller and control modules communicate with the rest of the system through the central hub. The data flow directions are marked by the arrows: messages sent from the central point to the endpoints flow downstream, while responses from the endpoints flow upstream.

The elements of the systems are composed in a tree hierarchical architecture (Figure 19). There is one root module which is the central hub. From this point several branches can be driven, each consisting of hubs and endpoints, which are leafs. The central hub distributes the readout requests coming from the central controller and control messages from the control module. Those messages flow in downstream direction, from the source to the endpoints. The responses generated by the endpoints flow in upstream direction. As the endpoints are treated as independent from each other, there is no communication channel between them in a horizontal direction. The hubs can distribute messages from its root to its endpoints or collect messages from the endpoints and forward them to the root.

Such architecture is required in order to perform synchronized measurements. While each endpoint is processing data independently from the others, a readout request from the central controller forces all the endpoints to tag the data with the current, centrally generated readout number marker (trigger tag) and transmit the data out of the system. Data fragments tagged with the same number can be then combined into structures representing the state of the processing in the whole system in a precise moment of time.

The TRB platform provides a base module, which can act as a hub and/or as an endpoint. Depending on the loaded firmware, the board can realize all the above described functions: contain endpoint modules that provide data, an instance of the hub, a control and a central controller modules. It means that the smallest system can be built out of a single board, which is convenient for small setups. This high flexibility in the usage of the base module could be possible thanks the FPGA devices. While the hardware remains the same and provides the infrastructure, the exact function of a board is defined by the loaded firmware.

The base module also provides a support for Add-on boards, which can extend its measurement and processing functionality. Those extension boards require only the connection to the hub instance in the base module in order to become an endpoint of the system. This concept results in a highly extensible system. The development of new elements, compatible with the system requires only the implementation of a proper endpoint interface.

It is worth to remind that the terms endpoint and hub exist as both: dedicated hardware modules and also as firmware components. This allows for development of structured firmware with well-defined functionality distribution and common interfaces. It applies also to the protocol used for inter-electronics data exchange as well as for the inter-components communication in the firmware.

Data processing can be executed on each layer of tree structure. The endpoints are data sources, with some logic resources for basic feature extraction and which can transmit the collected data only in the upstream direction. They are also equipped with readout buffers which can store data until arrival of the of the readout request.

The data path can go through the layers of hubs or alternatively exit the system at any point that features external network gateway. Hubs closer to the root of the system deal with larger subsets of data, collected from underlying layers. It is both: an advantage because more complex algorithms can be executed there but also possible bottlenecks can occur, which have to be addressed by the proper system configuration, because increasing data subsets can emerge. The algorithms that introduce some additional latency to the real time data path increase the dead time of the system. The base modules are equipped with gateways to the external network, through which the collected data can exit the system. Therefore, the systems has to be properly designed for each application, depending on the imposed processing requirements.

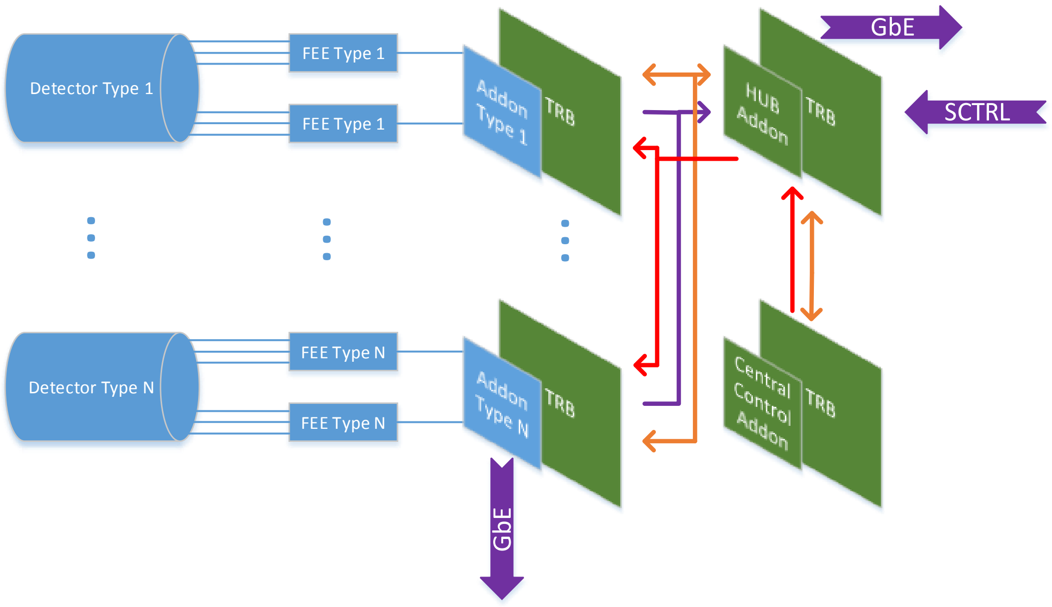


Figure 20: Schematic view of the system architecture applied for detector readout. Signals from different detectors are connected to the appropriate front-end electronics, which outputs are digitized and processed by the endpoints. Inter-TRB communication shares three logic channels: data readout (purple), slow control (orange) and trigger distribution (red). The blue elements combine detector specific hardware, while the green modules are common platforms. The violet arrows show the gateways for external networks.

An example of TRB platform application (Figure 20) is a data acquisition for a general DAQ system consisting of N detectors. The system can consist of various detector types that need to be simultaneously digitized and the results need to be stored in output buffers. The signals coming from different detectors require different handling, therefore the system has to feature various endpoints providing the requested measurement and processing capabilities. The concept of Add-on modules in TRB platform realizes this point. The core of the system remains the same, while only the Add-on modules need to be developed or adapted for specific applications.

The process of collecting data from detectors is controlled by the central controller of the system, connected to the central hub. The endpoints digitize and process the data in the real time according to their local algorithms and store the data in readout buffers. At some point, the central controlled issues a readout request command, which is distributed through all the hubs in the system to the endpoints, directing them to forward buffered data to the external network gateways. After sending this request, the system enters busy state. It is the time when no further readout requests can be issued, until all the endpoints report their readiness, which usually happens after freeing up the buffers. During this period of time, the system is not capable of recording new events. Proper implementation of buffering mechanism can reduce the dead time, but it is not possible to avoid it completely. The time since issuing the readout request and receiving all the busy release messages is the dead time of the system, that's why it is important to keep it as low as possible.

4.2 Communication Protocols

A crucial part of the presented architecture are the communication protocols which were applied to provide data transmission capabilities between the system components (Figure 21). Two main functionalities were distinguished and appropriate protocols were implemented basing on specific requirements and available resources. The first protocol is used for controlling the readout process by distributing the readout request messages and control commands. It requires the lowest latency possible, needs to be implemented between the system components and does not require advanced routing mechanisms. The original TrbNet (24) was invented and developed as a system specific protocol fulfilling the above requirements. Different requirements are imposed by the handling of the data gathered by the endpoints. This data has to exit the system at some point and be delivered to the servers for further, offline analysis. As they are PC class computers, a natural choice was to use a well-established standard like Gigabit Ethernet.

4.2.1 TrbNet

Indispensable element of the TRB platform is the TrbNet protocol (25). A dedicated network protocol, developed especially for TRB platform is used for three main purposes: distribution of readout requests, readout data transport and exchanges of the slow control messages. The protocol is independent from media interface, which means it can run on lines between devices on a PCB, optical fibers, copper cables and any other communication media. It requires two lines per connection as it can run only in full duplex mode needed by the handshake mechanism.

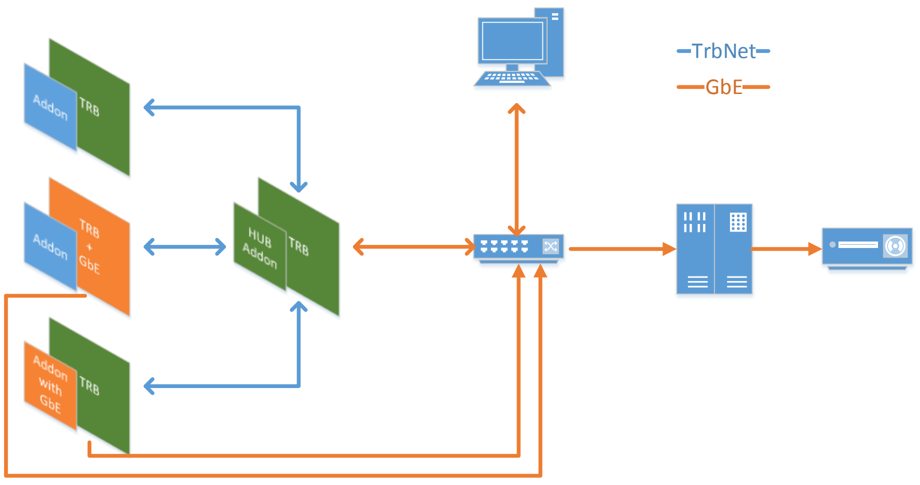


Figure 21: Schematic view of the networking in the DAQ system. The TRB boards communicate with each other through optical connections running TrbNet protocol (blue). The readout of the collected data, as well as slow control from PCs is performed on Gigabit Ethernet links (orange). Some Add-on boards are capable of transmitting GbE directly. The Gigabit Ethernet connections from the electronics, enter the infrastructure, composed of network switches, which distributes readout data to the event builders and then to permanent storage as well as assures the slow control interface.

One link is shared over all three functionalities. It means that one physical link contains three logical channels: one for trigger, one for data and one for slow control. It is very important that the protocol assures lowest latency possible on readout request distribution channel, which is crucial for proper synchronization and operation of the system. It is realized by prioritizing the types of messages. Readout requests has the highest priority, then comes the data readout and slow control at the end which is the least timing dependent channel.

Transmission is realized in a form of fixed-length, 80 bit packets of different types like header, data, end of block, termination and acknowledge. Each transmission consists of exchanging at minimum one header, one data and one termination packet. If there is much more data to transmit, it can be divided into several blocks and then end of block packet is used in order to separate them. In order to keep the latency low, it is important to avoid large data transfers and divide large portions of data into several parts. In such way, the packets containing readout requests can be injected in the middle of the ongoing transmission. After each block, the receiver sends an acknowledgment packet in order to assure the reliability of the connection. Packets contain a calculated CRC which is checked by the receiver. Depending on situation the packet can be marked as containing error or retransmission is requested. Such mechanism is especially needed in highly radiated locations, which can lead to Single Event Upset (abbr. SEU). Those can often happen, when the electronics are mounted close to the detectors and can even lead to complete de-synchronization of the link.

Each physical TrbNet channel is composed of several elements, symmetric for the receiver and the transceiver. Media interface is the lowest, physical layer. It implements the link access, low level data encoding like 8b/10b and the mechanism of bit transport and reception over the link. It's a composition of hardware transceiver and firmware that controls it. The next element, which is included in each TrbNet node is the endpoint. The firmware that handles and decodes the data is static and common to all implementations. The data received on the media interface is passed to the multiplexer which recognizes the type of the incoming packet and redirects it to the appropriate I/O buffer. There are three I/O buffers, one for each logical channel. What's important to notice is that the buffers are located after the multiplexer. This is requested in order to assure the lowest latency and bottleneck–free way of readout request packets delivery. After each buffer, there is a handler designated for specific type of packet, which decodes the payload and delivers the data on the output interface for the user to process it further. The mechanism for transmission works in the exactly same way and the data passes through all those elements in the reversed order.

4.2.2 Gigabit Ethernet

While TrbNet is a reliable and fast protocol, its drawback is that it can only be implemented on custom electronics and that it features functionality which is not needed for the basic data transmission. That is why Gigabit Ethernet (15) has been chosen as the protocol used for transmitting the collected data out of the system. It is a well-established on the market standard in modern telecommunication systems. The hardware needed for constructing a network is cheap, easy to access and standard PCs are usually equipped with needed Network Interface Cards (abbr. NICs) together with well-verified drivers and software, supported by the large manufacturers.

The network environment of the DAQ (Figure 4) systems in most of the applications can be treated as Local Area Network (abbr. LAN). It is characterized by the fact that those systems consists of many interconnected devices, exchanging large amount of data between each other, with limited communication outside defined sub-networks and located in a specified area. They are a mix of passive nodes, that only transmits gathered data out to a defined destination, and nodes that require communication in both directions as well as a mix of custom electronics and standard, market-available devices. Those are the reasons of choosing the TCP/IP protocol suite as a set of protocols running over Gigabit Ethernet networks. For the Transport Layer, two most common protocols are used: TCP and UDP. Depending on the setup and needed functionality, those two are complemented by a set of network discovery and other helper protocols.

In the described TRB platform, the TrbNet protocol is used only for inter-electronics communication. When it comes to transmitting data out of the system, the Gigabit Ethernet is introduced at all the hardware modules equipped with appropriate devices. It is important to enable the data exit the system at any point in order reduce bottlenecks and to keep the TrbNet links available for readout requests and control messages. In case it is not possible, that readout data is passed via TrbNet to the first module that has a gateway to GbE network.

The GbE gateways gather data coming from connected endpoints and compose UDP packets that are transmitted to the offline analysis servers. Although the protocol is not reliable and does not guarantees the delivery of uncorrupted data it is much more suitable for implementation as FPGA logic and features lower overhead due to reduced headers set than TCP. Some sort of reliability can be restored by a proper design of the network infrastructure, which has been proven to be sufficient, even for large scale setups.

The Gigabit Ethernet Module is described in detail in Section 4.5.

4.3 System Components

The entire system is composed of many modules, specialized to perform a given task or type of measurement. The main element is the TRB board which acts as a support for the extension modules and provides basic functionality. Let's first take a closer look at TRB2 (Figure 22) (26), which is still widely used in many setups.

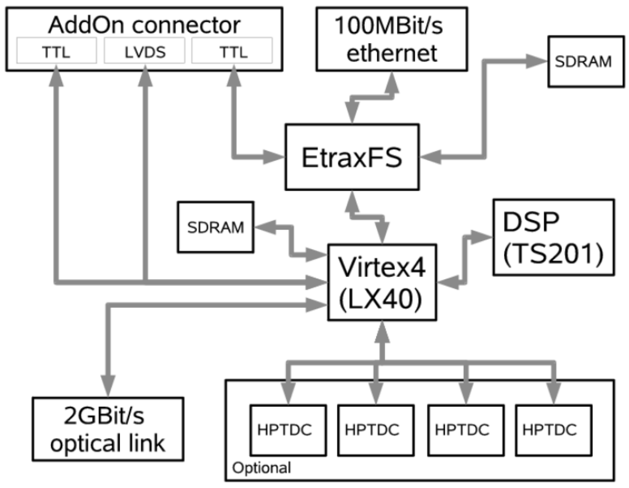


Figure 22: Schematic view of the TRB2 platform. Two main processing elements are the Virtex4 FPGA and the EtraxFS processor. The first one is connected to all the peripherals, including HPTDCs, while the Etrax assures the Ethernet gateway to the system network (26).

The central point of the board is a Xilinx Virtex 4 FPGA, which is the controller of all the other devices mounted on the PCB. The gateway to board is assured by the AXIS ETRAX processor, which runs a lightweight version of a Linux operating system with the entire TCP/IP stack, using a 10/100 Mbps Ethernet RJ45 copper cable media interface. Thanks to this processor, one can remotely log in to the TRB from a standard PC and execute slow control commands. The processor is connected to the FPGA, which allows loading firmware, setting its control and reading out status registers. Additionally, there are four HPTDCs (27) for precise time measurement. Their highest resolution can be set to 25 ps, limiting the amount of input channels per device to 8. There are 32 channels in standard resolution mode of 100 ps. The TDCs are connected and controlled by the FPGA, which also gathers the generated data. There are two ways of transmitting collected data out of the board. One goes through ETRAX processor and then further using standard Fast Ethernet network and UDP protocol. This mode provides 100 MBps, it is a slow connection and thus another option is the use of the optical transmitter SFP connected to the FPGA through a TLK SERDES (Serializer / Deserializer) chip has been foreseen. On this 2 Gbps link a custom protocol TrbNet is used. In order to reach a PC through a 1 Gb Ethernet network an additional module is needed, so called HUB. HUB Add-on supplies many input optical links, and one Gigabit Ethernet optical fiber output, which can be connected to a standard network infrastructure. Additionally, TRB2 supplies a TigerShark DSP device from Analog Devices. It's a dedicated processor for fast signal processing, which can be used optionally. As it was mentioned earlier, an important element of TRB platform is the Add-on connector. The connector provides 32 LVDS and 47 LVTTL lines reaching the controlling FPGA and additionally +5 V power and grounding. There are also 18 LVTTL lines connected to the ETRAX processor, in order to directly reach the Add-on boards from the Linux level.

The power of modern FPGA devices, which grow at a very high pace, was a response for a demand of higher requirements imposed by now-a-day experiments. In TRB3, which replaces TRB2, one can find only FPGA devices, which can be configured with a firmware appropriate for a specified task. This is an example of the shift from designing dedicated electronics into developing highly efficient firmware that replaces the functionality and can be uploaded into configurable devices at any time. Hence, there are five Lattice ECP3 FPGAs mounted on TRB3, four of them are located at the edges of the board and serve as processing devices, while the fifth one is central and acts as the controller of the entire board. For example it can be programmed as TDC with variable time precision and number of channels. Each edge FPGA has also a large connector capable of supporting an additional Add-on card. Thanks to such construction, it's possible to adapt the TRB3 to many different applications. As well as the previous version, the TRB3 board has an Add-on connector on the back. It means, that the base board can be extended by additional five different modules, all powered and interconnected through the central FPGA.

Eight 3.2 Gbps optical transceivers, all connected to the central FPGA provide sufficient bandwidth to transmit collected data to further steps of data acquisition system and maintain the TrbNet connectivity. The TRB3 board, as one of the key developments from the author is described in detail in Section 4.4.

There are multiple Add-on boards designed to extend the functionality of the base board. These include:

- HUB Add-on
  Board that provides 24 high-speed optical transceivers and 2 interconnected FPGAs for data processing. The module is used as a concentrator board and as a gateway to Gigabit Ethernet network.

- Shower Add-on
  Board dedicated for analog signal sampling using ADCs, was designed for the Shower sub detector of HADES system. There are six ADCs from Analog Devices, featuring 10 bit resolution, 20 MHz sampling rate and 8 input channels each. The ADCs are read out by the three Lattice ECP2M FPGAs and collected data can be transmitted directly to the Gigabit Ethernet network.

- Central Trigger System Add-on
  Board is used for a precise generation of trigger signal, which includes many inputs from various subsystems, an FPGA that is processing those signals and many signal outputs for the generated trigger signal. Additionally there are two SFP connectors for communication with the rest of the readout system. The board issues the readout requests and gathers the busy release messages in order to control the readout flow.

- MDC Add-on
  Large concentrator board, providing 32 optical transceivers for POF (abbr. Plastic Optical Fiber) type of fibers, each reaching 250 Mbps in full duplex mode. They are used to collect data from and control the motherboards on which Optical EndPoints (abbr. OEPs) are mounted. There are two additional, standard SFPs, one for communication in TrbNet protocol and the second one the Ethernet gateway. As processing units, there are five Lattice ECP2M FPGAs. One, central acts as a controller, while the four remaining control subsets of POF transceivers.

- TOF Add-on
  This Add-on can be considered as a front-end module as it contains discriminators only and no digitizing nor processing units. Having 128 input channels, discriminated with fast NINO (28) chips, the module fits directly to the HPTDCs on the base TRB2 board.

- General Purpose Add-on
  A board design for test purposes only. It features many input and output connectors, operated at different standards (e.g. LVDS, ECL etc.) providing a platform for firmware and hardware evaluation.

4.4 Trigger Readout Board version 3

The group of basic electronic boards building the HADES DAQ system has been extended two years ago by a new member, the new version of TRB board. It is a result of the experience gained over the years with TRB2 and of the fast development of technologies like the FPGA devices. It is also a consequence of growing demand for higher precision, faster and flexible solutions designed for modern experiments. Built as a replacement for TRB2, it has been assured that the new version is backward compatible. The board has been also designed keeping in mind that it would serve as a standalone measurement station, without the need of constructing the setup composed of several boards.

4.4.1 Hardware

In terms of hardware, the TRB3 (4), (29) (Figure 23) presents a different approach than its predecessor. All specialized devices, have been replaced by the FPGAs running firmware with dedicated functionality. Having FPGAs as main processing devices allows configuring them with custom firmware adapted for a specialized application. The board is equipped with five Lattice ECP3 FPGAs, connected in a star-like architecture with one central device and four satellites called edge FPGAs.

The central FPGA device is foreseen to act as a controller of the entire platform. Its communication is supplied by eight 3.2 Gbps SFP transceivers. Each one can run as Gigabit Ethernet or TrbNet link, depending on the functionality implemented as FPGA logic. The SFPs are grouped by four and connected to hardware SERDES blocks, included in the FPGA fabric. Therefore, two communication standards can run at the same time on the optical connections. Lattice ECP3 has 16 SERDES channels, the rest of them is used for inter-FPGA communication. The TRB2 had ETRAX processor with a lightweight version of Linux running on it. The processor assured the communication with a controlling PC via Fast Ethernet. In case of TRB3, this functionality comes in a form of the optical fiber connector and FPGA firmware implementing the Gigabit Ethernet Module. As the backward compatibility was mentioned, the board is equipped with the same Add-on connector on the back as TRB2, to host large extension modules like Shower or HUB Add-ons. The connection is supplied to the central FPGA, which hosts the communication features with Add-ons. Additionally, one can find two RJ45 connectors on the PCB, which are supposed to deliver digital trigger signals to the central, control device. Each connector can receive or transmit up to four differential LVDS signals to the external devices.

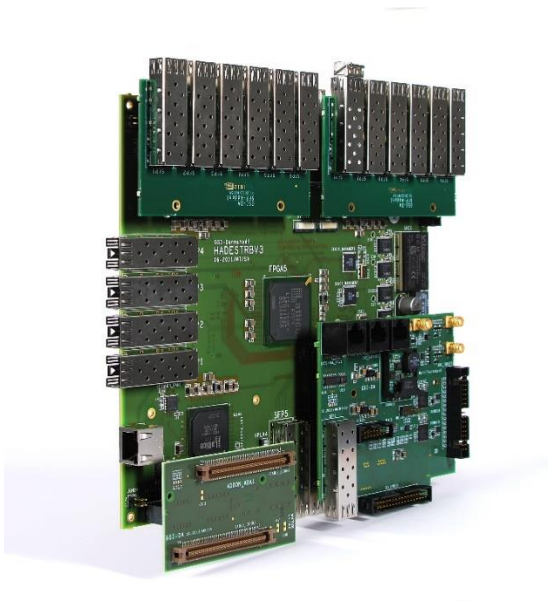


Figure 23: Photo of TRB3 equipped with different extension modules. On top there are two SFP mezzanine Add-ons plugged, down left is an ADA mezzanine Add-on and on the down right Multi-test mezzanine Add-on. [trb.gsi.de]

The HPTDCs mounted on TRB2, have been replaced by four Lattice ECP3 FPGAs, soldered on four edges of TRB3. Each of those has a dedicated 208-pin Samtec QMS connectors. Those advanced components provide the connectivity, power supply and mechanical support for small extension boards, called mezzanine cards. Those cards are the interfaces for the edge FPGAs with the other elements of the system. Thus the firmware loaded to the FPGA fabric has to be adapted for a given mezzanine card type that is connected. Several of those small extension modules have been developed and are already in use:

- **ADA Mezzanine Add-On**
  It is a passive board that contains two 80-pin KEL connectors – the same that can be found on TRB2, providing timing signals to the TDC. This Add-on assures the backward compatibility with all the front-end modules that up now were readout by TRB2. For this application, the FPGAs are programmed as TDC units with up to 64 channels each. There exist also two other versions, with different channel density of KEL connectors.

- **SFP Mezzanine Add-on**
  The mezzanine provides additional 6 SFP slots for multi-gigabit transmission. It can be used as a hub for TrbNet protocol, allowing the connection of slave boards and extending the setup or as a multilink Ethernet gateway.

- **CTS Mezzanine Add-on**
  The board is equipped with additional I/O for the use with CTS firmware.

- **ADC1 Mezzanine Add-on**
  A card with 12 ADC devices mounted. Summing up to total of 48 channels with sampling rate of 65 MHz and 10 bit resolution, stands as an efficient ADC platform.

- **PADIWA**
  The board features high speed comparators with configurable thresholds that are used for signal discrimination. The output in form of digital impulses is delivered to the TDC firmware running on the supporting FPGA.

- **PET-ADC Mezzanine Add-on**
  A mezzanine card developed for a novel ADC implementation with the use of fast comparators and high-precision time measurement.

- **Multi-Test Mezzanine Add-on**
  The general purpose board, equipped with a number of connectors and devices.

The variety of above described mezzanine cards, shows how the TRB3 can be used in many different applications, whether there is a need for ADC or TDC measurement but also for other purposes. The TRB3 itself does not feature any functionality except the processing power of the FPGA devices, those are the extension boards and the adapted firmware which defines platform functionality.

As the FPGA have volatile configuration, they need to be programmed after each reboot or power cycle. It can be done via JTAG chain (abbr. Joint Test Action Group) and an external programming device. This process has to be done manually which is a problem in large systems. In order to avoid this inconvenience, each of the FPGAs on TRB3 has a Flash memory chip connected. The memory can be loaded with a FPGA design file which will be loaded automatically after the boot-up. The interface to the Flash memory is provided through the FPGA, hence it first needs to be programmed via JTAG with a working design containing TrbNet and then it opens the way to upload the bit file to the memory.

4.4.2 Firmware

As the TRB3 does not contain any other processing devices except FPGAs, it is in the firmware that all the functionality has to be implemented. Some modules like the inter-FPGA communication is common to all the applications and stand as the framework which is extended by the custom modules, each implementing a dedicated function. The following is the list of most important firmware components developed so far:

- **TDC**
  Very high precision time measurement channel (30), (31). Instead of using the dedicated TDC chips, the functionality has been implemented as FPGA logic. Very specific, non-standard usage of the FPGA internal structure has been applied for measuring time difference between the trigger signal and an input signal from the external device. The special connections between internal FPGA elements, called carry-chains introduce a specified delay on carried signal. Injecting a signal into such a chain and measuring how many of the chain elements have been "hit" provides an estimate about the length of the signal. This method results in time resolutions of less than 20 ps, depending on applied calibration and optimization algorithms. The great advantage of such solution in comparison to standard dedicated TDC devices, is that it is a highly configurable implementation. The amount of channels can be traded for more optimization modules, resulting in higher precision. In case the application requires higher channel density, it can be reconfigured at the expense of the time resolution. It all depends on the amount of available resources in the FPGA. Up to now, a stable release of the firmware, implements 64 channels with 10 ps time resolution and double edge detection.

- **CTS**
  The Central Trigger System functionality has been implemented as a module included as a part of the central FPGA (8). The module takes LVDS input signals from RJ45 connectors as external trigger sources but can also generate the signal internally. A number of random and periodical internal pulsers can be configured as well as trigger generations by coincidences between predefined groups of signals and/or special signal patterns. It is one of the vital modules that allow to use the TRB3 as independent measurement station.

- **TrbNet**
  All components needed for implementation of full-featured TrbNet protocol, have been ported to Lattice ECP3 devices. Usually the edge devices have TrbNet Endpoint modules and the central FPGA contains TrbNet HUB acting as a concentrator. The protocols has been covered in details in section 4.2.

- **Gigabit Ethernet Module**
  The module that allows the communication in both directions with other system components via Gigabit Ethernet networks. Required in case of

single-board setups as it is the only interface between the PC and the board. The module is described in the next Section.

4.4.3 Gigabit Ethernet Module

The lack of a dedicated processor that could handle the communication with standard computers, forced the development of the full-duplex GbE Module, together with the implementation of additional protocols. The module stands as the only interface between the board and the controlling PC or event builders. On the boards implementing GbE as a gateway for the readout data, there was no need for the receive channel to be active. The TRB3 needs communication in both directions. This means not only low level reception of incoming bytes but also implementation of the protocols needed for network discovery like DHCP and ARP. The Gigabit Ethernet Module for TRB3 is used for two purposes. The first one is the standard GbE gateway for the collected data as the board acts as a concentrator (Figure 24). The second one is the Slow Control interface (Figure 25). As the board can run as standalone measurement station, it is essential that communication is kept with a PC giving the operator a way of controlling the operation of the device.

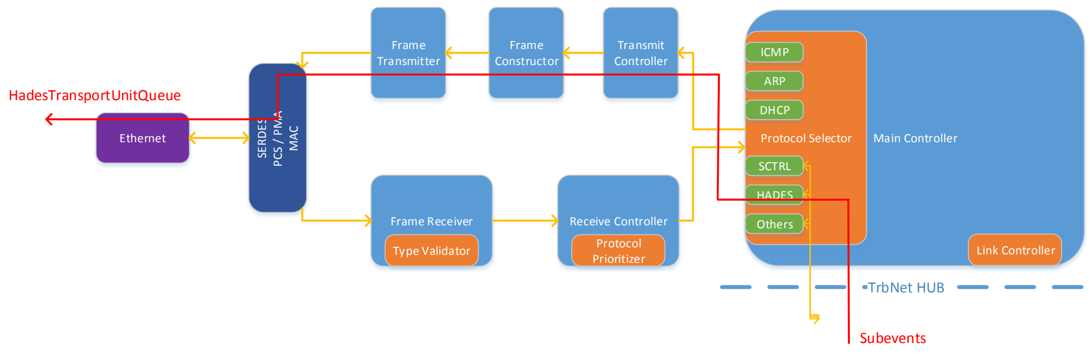


Figure 24: Schematic view of the Gigabit Ethernet Module implementation, adjusted to TRB3 board. The data readout path is highlighted by red arrow. The data flows only in outside direction, passing through transmitting and constructing components.

The Slow Control Response Constructor replaces the ETRAX processor located on TRB2. The processor kept communication with a PC in a form of server-client communication. Each TRB2 has one instance of TrbNet Daemon which receives and processes the requests from TrbNet Clients running on computers in the network. The processing consists of decoding the TrbNet message carried by the received network packet and its injection into the TrbNet Slow Control channel via the interface between the ETRAX processor and the FPGA on TRB2. In case of TRB3 the similar functionality is implemented by the Gigabit Ethernet Module extended by the Slow Control Response Constructor. Packet reception and decomposition is realized by the network stack implementation of the GbE Module. The Response Constructor receives the payload of the UDP packet sent from PC running TrbNet Client and proceeds with insertion of the TrbNet Message content into the Slow Control channel of the network. Each TrbNet request results with a response message sent from the electronics back to the PC. This message can take two forms. One is an acknowledge message which is sent in case there is no data to be transmitted back to the Client. The second one is the message containing additional payload, as a result for example of a message requesting the content of some registers in the system. Messages containing payload can be as large as single UDP packet, up to 64 kB. This requires a large buffer to be instantiated in the Response Constructor which will receive the entire data from the Slow Control channel. The content of the buffer is then formed into UDP packets and fragmented by the components of the GbE Module and transmitted back to the Client PC.

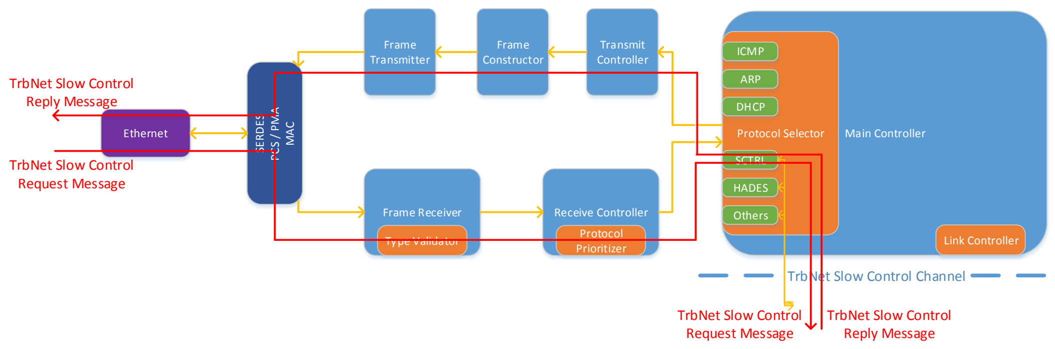
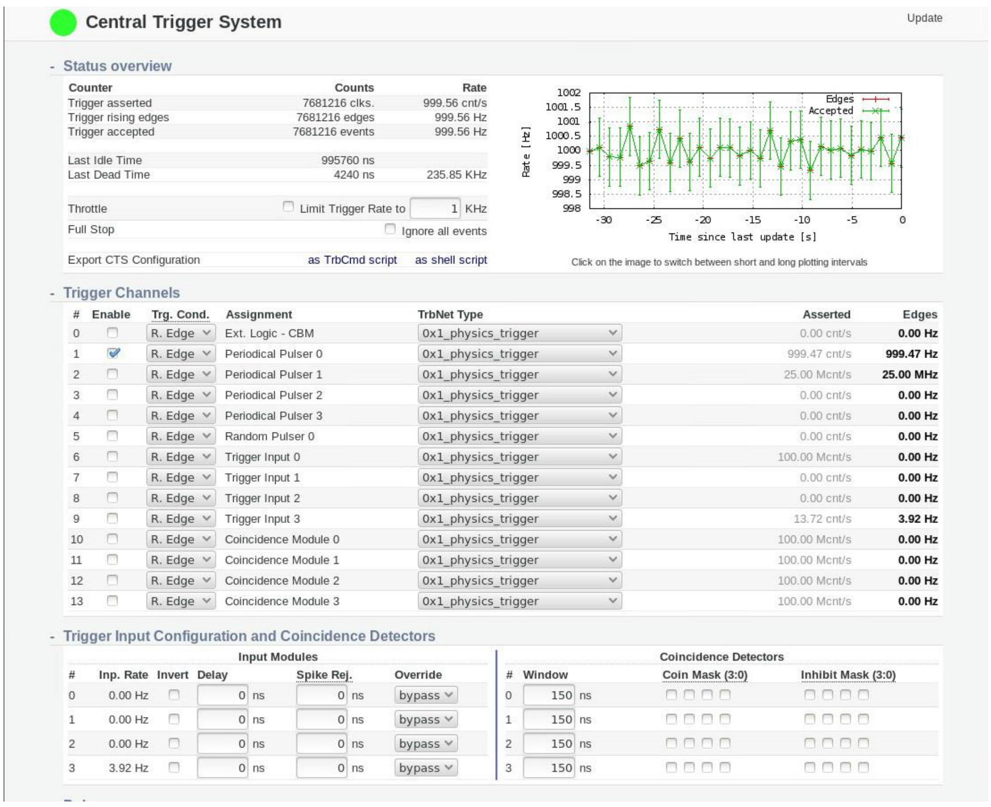


Figure 25: Schematic of Gigabit Ethernet Module implementation, adjusted for TRB3 board. Slow Control channel for TrbNet is highlighted by red arrows. The protocol require duplex communication. The Requests are processed by the receiving part and delivered to TrbNet Slow Control port on the TrbNet HUB. The Response messages supplied by the HUB are constructed into UDP packets by construction and transmit path.

4.4.4 Software

As TRB3 platform is foreseen to be used in many applications, it comes with a packet of software for the Slow Control and Event Building which forms a complete environment for measurement. Most of the software is based on the solutions developed in HADES experiment.

The TrbNet Slow Control Client has been updated to handle proper communication with the Gigabit Ethernet Module instead of ETRAX processor. The data format has been extended by additional headers and retransmission mechanism in case of missing reply message has been implemented. Having Slow Control channel active, there is a number of monitoring tools that has been built on top of the client including web interface which can be accessed through any web browser in the network. Those tools provide very useful information about the general condition of the setup, data rates, hit counts on TDC channels, etc. All the registers implemented in the FPGAs in the system can be accessed and their values can be presented under some form, through the web interface. The information on the interface is updated with a configurable interval, giving real-time access to the values. Apart presenting register values, the interface also gathers input from the operator. It is the way of configuring the modules included in the FPGA designs, such as Central Trigger System (Figure 26). The interface is written in a modular way, opening the possibility of extending it by additional components, as the FPGA firmware introduces new features.



Central Trigger System

Status overview

Counter	Counts	Rate
Trigger asserted	7681216 clks.	999.56 cnt/s
Trigger rising edges	7681216 edges	999.56 Hz
Trigger accepted	7681216 events	999.56 Hz
Last Idle Time	995760 ns	
Last Dead Time	4240 ns	235.85 KHz

Throttle ☐ Limit Trigger Rate to 1 KHz

Full Stop ☐ Ignore all events

Export CTS Configuration as TrbCmd script as shell script

Click on the image to switch between short and long plotting intervals

Trigger Channels

#	Enable	Trg. Cond.	Assignment	TrbNet Type	Asserted	Edges
0	☐	R. Edge	Ext. Logic - CBM	0x1_physics_trigger	0.00 cnt/s	**0.00 Hz**
1	✓	R. Edge	Periodical Pulser 0	0x1_physics_trigger	999.47 cnt/s	**999.47 Hz**
2	☐	R. Edge	Periodical Pulser 1	0x1_physics_trigger	25.00 Mcnt/s	**25.00 MHz**
3	☐	R. Edge	Periodical Pulser 2	0x1_physics_trigger	0.00 cnt/s	**0.00 Hz**
4	☐	R. Edge	Periodical Pulser 3	0x1_physics_trigger	0.00 cnt/s	**0.00 Hz**
5	☐	R. Edge	Random Pulser 0	0x1_physics_trigger	0.00 cnt/s	**0.00 Hz**
6	☐	R. Edge	Trigger Input 0	0x1_physics_trigger	100.00 Mcnt/s	**0.00 Hz**
7	☐	R. Edge	Trigger Input 1	0x1_physics_trigger	0.00 cnt/s	**0.00 Hz**
8	☐	R. Edge	Trigger Input 2	0x1_physics_trigger	0.00 cnt/s	**0.00 Hz**
9	☐	R. Edge	Trigger Input 3	0x1_physics_trigger	13.72 cnt/s	**3.92 Hz**
10	☐	R. Edge	Coincidence Module 0	0x1_physics_trigger	100.00 Mcnt/s	**0.00 Hz**
11	☐	R. Edge	Coincidence Module 1	0x1_physics_trigger	100.00 Mcnt/s	**0.00 Hz**
12	☐	R. Edge	Coincidence Module 2	0x1_physics_trigger	100.00 Mcnt/s	**0.00 Hz**
13	☐	R. Edge	Coincidence Module 3	0x1_physics_trigger	100.00 Mcnt/s	**0.00 Hz**

Trigger Input Configuration and Coincidence Detectors

Input Modules						Coincidence Detectors			
#	Inp. Rate	Invert	Delay	Spike Rej.	Override	#	Window	Coin Mask (3:0)	Inhibit Mask (3:0)
0	0.00 Hz	☐	0 ns	0 ns	bypass	0	150 ns	☐ ☐ ☐ ☐	☐ ☐ ☐ ☐
1	0.00 Hz	☐	0 ns	0 ns	bypass	1	150 ns	☐ ☐ ☐ ☐	☐ ☐ ☐ ☐
2	0.00 Hz	☐	0 ns	0 ns	bypass	2	150 ns	☐ ☐ ☐ ☐	☐ ☐ ☐ ☐
3	3.92 Hz	☐	0 ns	0 ns	bypass	3	150 ns	☐ ☐ ☐ ☐	☐ ☐ ☐ ☐

Figure 26: A view at the Central Trigger System control and monitoring panel as one example of the DAQ related software. It is a convenient interface for operators to access most of the settings and status registers. Applying to trends, the interface is implemented as a web page, depending on networking setup of the host PC it can be accessible remotely.

The Event Building software is exactly the same as the version running in HADES experiment. There are two reasons for that: the data structure coming from the electronics is the same and the software is already verified through a series of performed experiments. It is easy scalable for the amount of TRB3s in the system and for the number of computers running the software. The reassembled and saved events are in the binary format and need to be unpacked into digital values representing the results of the measurements. For this task several groups have developed different programs tailored for their own purposes. One of such has been developed by the author of this thesis as one of the crucial components of many TRB based DAQ systems, providing straight forward access to collected data. The Unpacker is a standalone library running under the ROOT framework. It iterates through the events, decoding the structure of individual sub-events. As the system can consist of different modules, each having its own internal data format, the Unpacker can be extended by modules deriving from one main class the basic functionality and interface. The developer has only to focus on implementing the data structure of its sub-event.

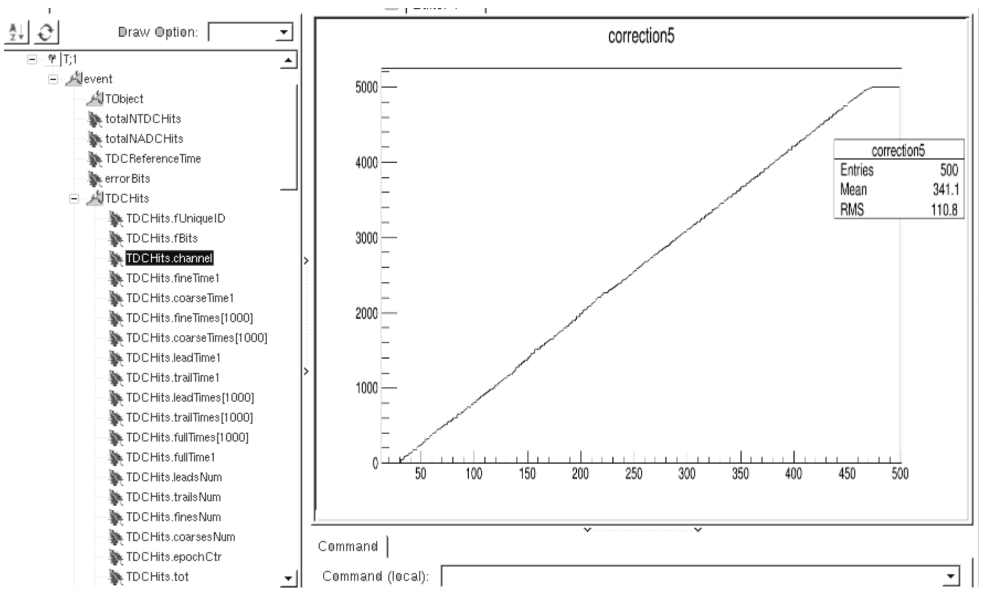


Figure 27: Example of data decoded with the Unpacker and opened in ROOT framework. The structure is represented as a tree structure on the left. Tree T consists of events and each has an array of TDCHits. Each hit is described by many fields like fine time, coarse time, channel number etc. On the right side a histogram of delays introduced by 500 delay elements in a TDC chain. One can notice that the line is not perfectly straight which is a result of DNL – single delay elements introduce slightly different delay times.

The Unpacker comes together with various ROOT macros that can accelerate the process of analyzing collected data. Easily adaptable for different hardware setups, they implement base operations on data and generate histograms, which help evaluating the measurement quality. A particular example of software is the TDC DNL corrections calculation (Figure 27). It is a set of tools used to calculate corrections for the individual errors introduced by the non-linearity of single delay elements in the TDC chain.

4.5 Gigabit Ethernet Module

As the DAQ system concept was evolving, especially with introduction of TRB3, the requirements for GbE Module evolved through the time. At first, the GbE functionality was limited to transmission of readout data only from the main concentrator boards like HUB, Shower and MDC Add-ons. Its task was to construct for each received readout request an UDP packet and transmit to a given destination. For this purpose no receive path was needed to implement as UDP is connectionless protocol. It also did not require any network discovery mechanisms, as all the source and destination addresses were supplied through slow control channel. Such implementation, called lightweight, was a framework, on which the extended one was built for TRB3. The new version of the main board in the system, is designed to work as a part of a larger system but also as a standalone measurement device. It is equipped with FPGAs only, so the only interface, through which the user can communicate with the board is an optical transceiver connected directly to the FPGA logic. That's why, the entire slow control, together with data readout had to be implemented as Gigabit Ethernet Module, this time, operating in both directions transmitting packets out as well as receiving them from the controlling computers.

4.5.1 Network Model

Implementations of a network structure are often represented in correspondence to Open Systems Interconnections Model (abbr. OSI) (32). The model represents the functions of a communication system, grouped into 7 layered stack (Figure 28). Each layer is communicating with its adjacent neighbors and the data flows in only one direction depending if it is reception (RX) or transmission (TX) channel. The TCP/IP suite introduces a slightly different model (33) that is also based on OSI but consists of four layers only, with the functionality grouped differently. Such approach is very convenient for describing building blocks of the GbE Module for a TRB-based system.

The lowest TCP/IP Link Layer (OSI Layer 1 and 2) is responsible for maintaining the local communication between two directly connected, point to point, in the Ethernet standard nodes and reception of basic data structures – Ethernet Frames. The most basic function is to maintain the media interface running (optical fiber or copper cable), agree on the link speed and synchronize the receiver with the transmitter to the same clock frequency and phase for correct single bit recognition and then find the transmitted word boundaries. The data formats that operates at this level are: single bits received on the link, reconstructed 8 bit words and then those words composing a single Ethernet Frame. This layer is responsible for the first stage of data decoding by recovering 8 bit words from 10 bit words received in the 8b/10b encoding. This type of encoding adds two additional parity bits at the end of transmitted word for a basic consistency check and DC balance (same amount of bits set to 1 and 0). In GbE Module, the functionality of this Layer is performed by hardware and software cores: Serializer-Deserializer (abbr. SERDES), Physical Coding Sublayer – Physical Medium Attachment (abbr. PCS-PMA) and Medium Access Controller (abbr. MAC) cores. The functionality of the Link Layer is enough to build a single local network. The Ethernet nodes are being addressed with the MAC address, which is a part of the Ethernet Frame header and allows for frame exchange between multiple nodes in the same sub-network.

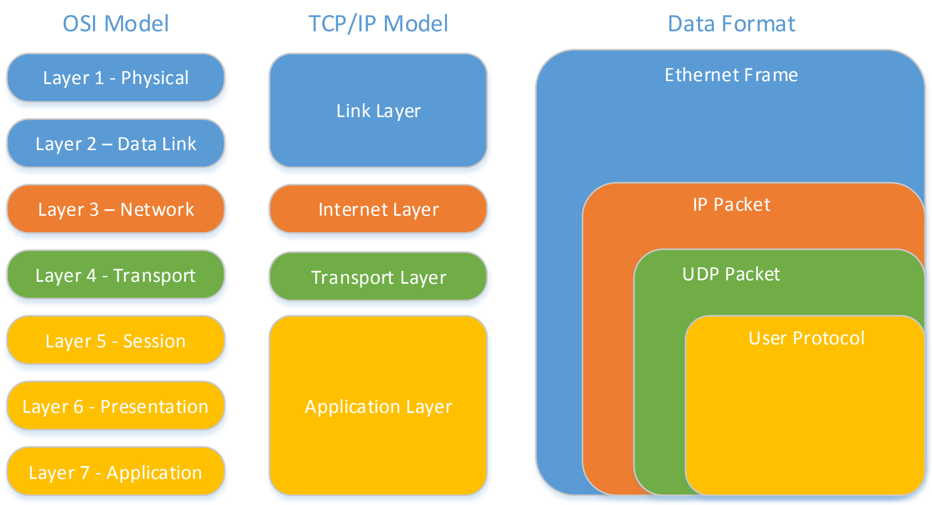


Figure 28: Different network stack models are presented. Similar functionalities are grouped by colours. On the left, the basic OSI model, in the centre the TCP/IP model and on the right side, the encapsulation of successive protocols.

The Internet Layer (OSI Layer 3) assures the communication on a higher level, allowing the inter-network exchange of packets. That means that another level of addressing and mechanisms are introduced in order to send and receive packets between multiple, different, connected together networks. At this stage the Internet Protocol (abbr. IP) is encapsulated into Ethernet Frames, changing them into IP packets. Such packet contains IP source and destination addresses, which define the networks of the two nodes. Then the mechanism of routing finds a way through the elements of the network in order to deliver the packet. The construction and decoding of IP packets in GbE Module is partly done by the Frame Constructor and Frame Receiver. At this point, the entire functionality, needed to send packets around the Globe is implemented.

Next comes the Transport Layer (OSI Layer 4) which by adding another level of addressing, defines the connectivity between two processes running on different nodes. This is where the type of the connection is defined as connection-oriented TCP or connectionless UDP. The GbE module does not implement (at least in its basic development) TCP but only UDP packets construction and reception. One UDP is a basic transport unit (maximum 64 kB), into which the readout data can be packed.

By specifying the destination IP address and the UDP port number, such packet will be delivered to the place, which is realized by the lower layers (Figure 29).

The last, Application Layer (OSI Layer 5, 6 and 7) defines the protocol that is transported by the functionality of the lower Layers. In this case that is inside the UDP packets, fragmented into IP packets which are encapsulated into Ethernet Frames. At this place the user defines its own protocol, what is transported on the network and how it should be decoded. In case of GbE Module, there are two specific protocols that are introduced at this layer: Hades Data and Slow Control over GbE.

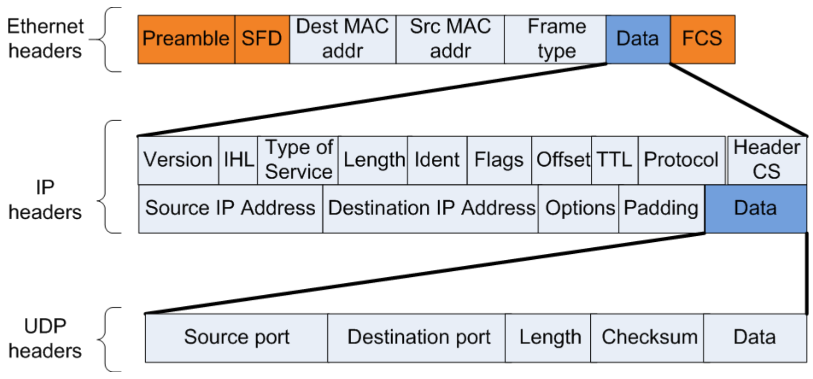


Figure 29: Detailed view of packet structure and protocol encapsulation. Orange headers are automatically added by MAC layer, while the "data" parts marked in blue are containers for the payload consisting of higher level protocol frame. Those are three levels of TCP/IP model successive layers: link (Ethernet), internet (IP) and transport (UDP).

4.5.2 Gigabit Ethernet Module Internal Structure

The Gigabit Ethernet Module is developed as a single VHDL module with a reduced to minimum interface consisting of ports needed to interface the link itself and ports for the User Logic to deliver and receive transmitted data (Figure 30). It is constructed in such a way that someone who needs to use the GbE Module in his project, could do it with minimum effort and knowledge about the how it works in detail. As VHDL is a platform independent language, all the components specific to a given platform were separated. This allow an easy migration to a different FPGA chip only by exchanging this one module or adjusting to the new architecture. The rest of the logic remains the same with all the functionality preserved.

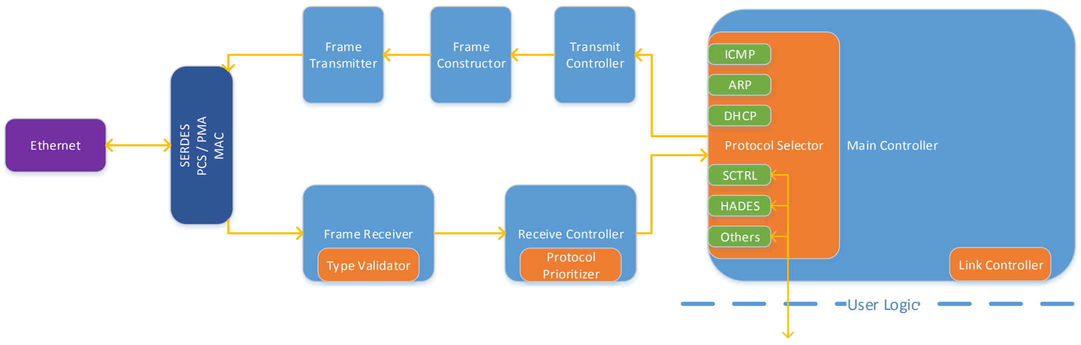


Figure 30: Schematic view of the Gigabit Ethernet Module. The network is interfaced via SERDES, PCS/PMA and MAC cores. The light blue components are platform independent modules implementing receiving and transmitting parts. All implemented protocols are housed within Protocol Selector, where data can be delivered to and from the rest of the user logic.

4.5.2.1 Media Interface Module

The only component that is fully platform dependent is the module that connects directly to the transceiver. The Media Interface module has different forms and is adjusted to the connected hardware on the PCB, whether it is an SFP transceiver directly connected to the FPGA fabric or the RJ45 connector connected through an external PCS (e.g. Marvell Alaska chip). It has a well-defined interface in direction of the rest of the logic, which is the Frame Receiver and the Frame Transceiver, which assures that the rest of the logic remains unchanged. Depending on the platform on which the GbE Module is implemented, the internal structure of the Media Interface varies and so far it has been developed for the following combinations of FPGA chips and transceivers:

- **Lattice ECP2M on Shower and HUB Add-ons**
  The FPGA is connected directly to the SPF transceivers. The ECP2M device has a hardware SERDES core and software PCS/PMA and MAC cores. Those components have well defined interfaces and can be easily connected together, performing the tasks of the lowest Layer.

- **Lattice ECP3 on TRB3**
  The FPGA is connected in a similar way to the SFP transceivers and also the cores are provided. It only had to be adjusted to a slightly different version of the SERDES hardware core.

- **Xilinx Virtex4 on Compute Node v2**
  Two SFP transceivers are connected directly to the FPGA fabric, which in contrary to the previous solutions, contains not only hardware SERDES core but also hardware PCS and MAC cores.

One RJ45 connector is connected via Marvell chip to the FPGA logic. It requires the use of one MAC hardware or software core.

- **Xilinx Virtex5 on Compute Node v3**
  Four of the SFP transceivers are connected directly to the FPGA fabric, which similarly to Virtex4 contains hardware and software cores that fulfill the functionality of the lowest Layer.
  One RJ45 connected via Marvell chip, requires only internal MAC to be used from the FPGA.

- **Other platforms**
  The GbE Module has been implemented on various other boards and FPGA chips, like different versions of Xilinx Virtexes 4 and 5 and the latest Xilinx Series 7 members like Kintex and Zynq.

All described solutions use three common elements: SERDES, PCS/PMA and MAC under different forms. As it was explained above, those elements are responsible for the functionality that is grouped under OSI Layer 1 and 2. SERDES together with PCS/PMA cores share the tasks assigned to Layer 1. They are responsible of encoding the received 8 bit word into 10 bit using the 8b/10b encoding, serialize such word and transmit individual bits over the link. The reverse procedure is applied in case of received traffic, the decoding of 10 bits into 8 bit words is performed. They also perform auto-negotiation, which is the procedure for the two link endpoints to agree on transmission speed and connection features. As the endpoints operate on separate clocks, they are not aligned in phase. That's why the two stage synchronization is needed. First stage assures that the receiver clock edge arrives in the middle of the validity of the received bit. The second stage, finds where the first bit of the transmitted word is in order to correctly decode single words. Usually this is realized by sending known comma characters which have to be properly received and decoded by the receiver.

The Media Access Controller operates on a higher level (OSI Layer 2). After the single bits are received and formed into words, the MAC reassembles them into single Ethernet Frames, which are the basic units of data transmission in the Ethernet. It checks its validity by calculating the size and the checksum. The analogue functionality is present while transferring a frame out. Additionally, it is a component which introduces the first level of addressing. The Ethernet Frame header consists apart of other fields of source and destination MAC addresses, which are supposed to be unique addresses of each device.

In most modern FPGAs the SERDES core is an ASIC which is a part of the FPGA fabric, a hardware core. In more advanced devices or in the FPGA families designed for high speed communication, the other two cores, PCS/PMA and MAC happen to be hardware cores but can also appear as software cores that are delivered by the manufacturer but have to be generated as implementations using standard FPGA logic and resources.

Let's now take a tour over the building blocks, presented on Figure 30, of the GbE Module platform independent logic, starting from the receiving path, going through main components with implementations of protocols and ending up with the transmission channel.

4.5.2.2 Frame Receiver

Received bits, reconstructed into 8 bit words and then reassembled into Ethernet Frames by the Media Interface Module are transferred into Frame Receiver Module. This module contains the mechanisms for the first stage frame decomposition, filtering and buffering. When a frame arrives, its destination MAC address is being checked at the first place. The module accepts frames that are addressed to this particular receiver or are broadcasts. The frame decomposition is a process of reading out the header fields of the protocols included in a frame and decoding individual fields like addresses, types of protocols, payload lengths etc. This is fulfilled by a finite state machine, which iterates through the frame bytes and depending on the recognized protocol, proceeds with appropriate header decoding. The module is prepared to handle IP and ARP packets on Internet Layer, UDP and ICMP packets on Transport Layer and multiple other Application Layer protocols, transported by the previous Layers. After all the headers are decoded, their values are passed to the Type Validator component, which contains configurable values of accepted protocols and decides if the received frame is valid to be passed to further stages of processing. In case the frame type and protocols included are not recognized, the frame is addressed to a different destination or the validator response is negative, the received frame is dropped. Otherwise, decomposed headers and payload are put into buffers in order to wait to be read out by the next stage. The buffers serve two purposes: received frame has to wait until its protocol implementation takes it out for processing but also they separate two clock domains. As the receiver clock is synchronized (phase-aligned) with transmitting endpoint clock and runs at a given frequency (125 MHz for Gigabit Ethernet), the received data has to be adjusted to the system clock running the processing logic. The dual port FIFO buffers allow to separate the read and write operations and synchronize them with the processing logic by supplying two independent clocks.

4.5.2.3 Receive Controller

The next module which is the Receive Controller, keeps track of the content buffered by the Frame Receiver and communicates with the Main Controller to find out whether the logic is ready to process the buffered frames. Depending on the type of the waiting frames, they are ordered by the Protocol Prioritizer module.

4.5.2.4 Main Controller

The Main Controller is the central point of the entire GbE Module. Its first task is to control the link status. After booting up, the controller configures MAC, waits for the positive outcome of the auto-negotiation process and then proceeds in two possible ways. If the GbE Module is configured to use IP protocols and has a static, hardcoded IP address, then the link is up and ready to use by the protocols implementations. There is also another way of acquiring the IP address. It can be done by the Dynamic Host Configuration Protocol (abbr. DHCP) which is a standard way of getting an IP address after booting up by network devices. The protocol client is implemented and in case it’s active, after the auto-negotiation is finished, the Main Controller enables the client to acquire the address. In the same time, all the other protocols are blocked from transmitting and receiving frames, other traffic than the DHCP messages is being dropped. The exact operation of the DHCP implementation is described in the following Section 4.5.2.5. After the module has an IP address, the link is ready to be used.

As the Gigabit Ethernet link works in full-duplex mode, the Main Controller takes care of the received and transmitted frames flow in parallel. Its key component is the Protocol selector which contains modules that implement selected higher-level protocols.

4.5.2.5 Protocol Selector and Response Constructors

When a frame is received and passes through the filtering, signalizing it is ready to be processed, the Protocol Selector activates the given protocol implementation called Response Constructor. The module can be treated as a large bus to which all the implemented Response Constructors are connected at specified slots. The signal from the Main Controller informing that a frame of a given protocol type is buffered in the Frame Receiver and waiting for processing, checks if the Response Constructor responsible for that protocol is ready to accept a new frame and if so, redirects all the decomposed headers and the payload into that module.

The state machine in Protocol Selector, constantly iterates through all the Response Constructors, checking their state and looking for a signal that one of them is ready to transmit a frame back. In that case the module asks the Main Controller if the link is available for transmission and if so, the Transmit Controller is activated and all the data from Response Constructor is redirected for transmission.

Both processes are executed in parallel, which is a great advantage of the FPGA device architecture. One Response Constructor can work on a received frame in the same time while another one is being transmitted. More precisely, all implemented Response Constructors work in parallel, whether it is processing an received frame, building up a new packet to be transmitted or simply waiting for any action to perform. As it is an advantage, it is a challenge as well in synchronizing the work of so many elements. That is why a bus architecture has been selected together with activity flags like response ready, busy and activate, which are being monitored and controlled by the Protocol Selector state machine.

The architecture is open for implementing new Response Constructors taking care of additional protocols. All protocol implementations share a common interface, which connects directly to the bus and to the mechanisms of receiving frame bytes as well as transmitting. The only difference between them is in the content of the packets and the interaction with the user logic. The following is the list with brief description of the main Response Constructors, which come as a part of the GbE Module:

- **ARP**
  Address Resolution Protocol (abbr. ARP) (34) is a basic protocol used in Ethernet networks to map the OSI Layer 3 IP addresses into physical MAC addresses of the OSI Layer 2. In case of the GbE Module it is used by other network devices to find out where the packet should be sent in order to arrive to the electronics running the GbE implementation. The Response Constructor works in a non-buffered, request-reply mode. For each ARP request frame that arrives to the module, it replies automatically with a response frame to the sender in case the link is available. There is no other processing needed, except checking the consistency of the request message.

- **DHCP**
  Dynamic Host Configuration Protocol (abbr. DHCP) (35) is a BootStrap (abbr. BOOTP) based protocol used by the client to acquire the IP configuration from the server running DHCP daemon. While the client can receive a lot of different settings, the current implementation is used only in order to get the IP address. The successful communication with the server is a four-step, handshake-like process of exchanging specified types of messages. It starts with the client sending a broadcast DHCP Discover message, which role is to find a running DHCP daemon in the network. When such message arrives to the daemon, it checks its configuration to find an available IP address and if the result is positive, it replies back to the client with DHCP Offer message. The offer contains an IP address, which the client can assign to itself and send a DHCP Request message. If the process is successful, the daemon replies with a DHCP Acknowledge message and that terminates the communication, leaving the client with a valid IP address. In case something is going wrong, the client is reset and starts sending DHCP Discover each second until it receives a proper response.
  The above described procedure is implemented in a form of three state machines. First one keeps track of the communication stage with the DHCP daemon. It either wait for a proper message type to arrive or launches the construction of an outgoing message. The second FSM starts running when a

frame is received. It parses the DHCP headers in other to gather out the message type, proposed IP address and overall validity. The third FSM steers the construction of DHCP Discover and Request outgoing frames.
The Response Constructor works in a non-buffered way, parsing the received frames from the Frame Receiver in the real time and constructing the responses on the fly. This approach helps reducing the FPGA resources needed by the implementation.

- **Ping**
  Ping is a basic tool, used to check if a network device is active. It comes in a form of an Echo Request message sent as Internet Control Message Protocol (abbr. ICMP) (36) and waiting for an Echo Reply. The current implementation serves only as a client which waits for a request and builds a reply message back. Its construction is similar to the ARP Response Constructor with a change in the parsed headers and a form of replied frame. Additionally, the module has a built-in CRC calculator, which is needed to properly construct the reply message. As it was the case in previous two protocols implementations, no buffers were used in this one as well.

- **Statistics**
  This is a custom Response Constructor built mainly for the debugging purposes. Its task is to send with a fixed frequency a message to a given by configuration destination, containing the statistics generated by all the protocols included in the GbE Module. It's is composed of a memory with predefined memory space for each Response Constructor in the system. This space is filled with statistics like sent, received or dropped frames, transmitted bytes etc. When it comes to the construction of a packet, the memory content is packed into an UDP packet and sent into predefined location, where a process should listen on an open port, acquire the incoming message and decode the bytes into the statistics. It is an example of an active Response Constructor, which does not require an incoming request from the network in order to build up a response.

The Response Constructors developed especially for a given application and setup of the DAQ system are described in the sections of Chapter 5. Additional protocols are also introduced in the Section 4.4.3 about TRB3.

4.5.2.6 Transmit Controller

When a Response Constructor signalizes with a response ready flag that it is ready to transmit an outgoing frame and is selected by the Protocol Selector state machine for transmission, it gains an access to the Transmit Controller. The module is responsible for interfacing between OSI Layer 4 Response Constructor and the OSI Layer 3 Frame Constructor. As the Layer 3 works on single Ethernet Frames, which are usually up to 1.5 kB and Response Constructor Layer 4 can build UDP packets up to 64 kB, the Transmit Controller needs to fragment the prepared payload into a series of smaller frames. Each single fragment of the payload needs to be encapsulated with the appropriate protocols headers and transmitted before the construction of next fragment begins.

The Transmit Controller consists of a state machine which communicates between the Frame Constructor and the Response Constructor. When a packet is ready to be sent, the FSM starts by asking the Frame Constructor to prepare headers based on the protocol and addressing information provided by the Response Constructor. When the headers are ready, the specified amount of payload is being transferred between the two modules. In case there is more data to transmit than allowed 1.5 kB, the FSM instructs the Frame Constructor to close the frame and send it out. During its transmission, a following frame construction is started and so on, until there is no more payload waiting in the Response Constructor. The Transmit Controller has to keep track of the entire transmission process, in order to correctly prepare header values in various protocols.

It is worth to mention that the single frame size of 1.5 kB, which is called Maximum Transmission Unit (abbr. MTU) is a value that can be configured. Not only the Gigabit Ethernet Module has to be adjusted, but all the devices in the network need to be configured accordingly. This value can reach up to 9 kB, the frames larger than 1.5 kB are called Jumbo Frames. It’s a mechanism used to balance and reduce the overhead introduced by headers due to high fragmentation. The configuration is strongly dependent on the traffic characteristic and the network architecture.

4.5.2.7 Frame Constructor and Frame Transmitter

Basing on the parameters provided by the Transmit Controller, the Frame Constructor builds a single Ethernet Frame together with the headers of higher level protocols, up to OSI Layer 4. It means that it can build packets of IP, ARP, ICMP and UDP types. The higher level protocols headers need to be prepared by Response Constructor and delivered here as payload. It consists of one large state machine, which task is to combine the provided values in a specified order, building up the frame headers and complement with payload data. The entire prepared frame is saved in a dual port FIFO buffer. The buffer is used here for the same reasons as it was used in the receiving path. It is the last place before the crossing of the domains of the system and the transmission clocks, which require proper synchronization.

When the frame is ready, the Frame Transmitter receives a signal to begin the transmission of the prepared data. This module interfaces with the MAC through the Media Interface Module, which takes care of transmitting process.

5 Applications

The systems based on TRB platform are commonly used in physic experiments in various applications, whether those are tests of detectors, test of other DAQ components like front-end electronics, small setups or as elements of existing architectures. Each of those setups require assistance of TRB experts during several phases of the development. At the concept stage, decisions about the system components have to be made depending on the requirements. As much as TRB platform is flexible, there are always adjustments needed in order to perform measurements in a most efficient way. After the setup is constructed and running, the maintenance, monitoring and evaluation is required.

All of those aspects are extremely important as they always provide a feedback about the operation of the system and possible improvements, which result in overall progress and development of the platform. It is also a significant part of work conducted by the author of this thesis. Not only the firmware development but also extensive tests and the hands-on experience is necessary in order to deeply understand the nature of data acquisition systems.

Apart many others, the main impact of this work was imposed on the HADES experiment as the application for which the TRB system was originally designed. The J-PET scanner is a great example of how the system originated from large scale, high-energy physics can be applied in smaller applications. The design, development and realization of this system is a substantial contribution of the author, resulting in a solution which became a subject of patent submissions (7). Those two experiments are described in detail in the following sections. The evaluation of those systems has been conducted and covered in Chapter 6.

5.1 HADES Experiment

High Acceptance Di-Electron Spectrometer (abbr. HADES) (5) (25) is a large detector system constructed at GSI Helmolthzzentrum für Schwerionenforschung GmbH facility in Darmstadt, Germany. The system consists of seven different detectors, each composed of six sectors that forms a ring over the reaction area. The beam line from the UNILAC linear accelerator passes through SIS18 synchrotron and arrives to the HADES cave where the particles hit the target, leading to the reaction which products are measured by the surrounding detector system. Apart of strong physics motivation, the experiment stands as a challenge for the data acquisition system design. Seven different detectors subsystems, sums up to 80,000 individual channels that hypothetically need to be processed and stored at rates between 20 and 100 kHz depending on the type of collision. Situation when all channels fire never happens, as during an event, the particles hit only some regions of the detectors, firing just a subset of channels. Simulations show that only up to 20% of the entire detector gets excited per event, depending on the reaction type. Ideally the DAQ system should be designed to readout the entire detector at a highest rate possible but the costs in both money and human work impose a compromise that has to be achieved between the system designers and physicists who require certain measurement coverage and accuracy.

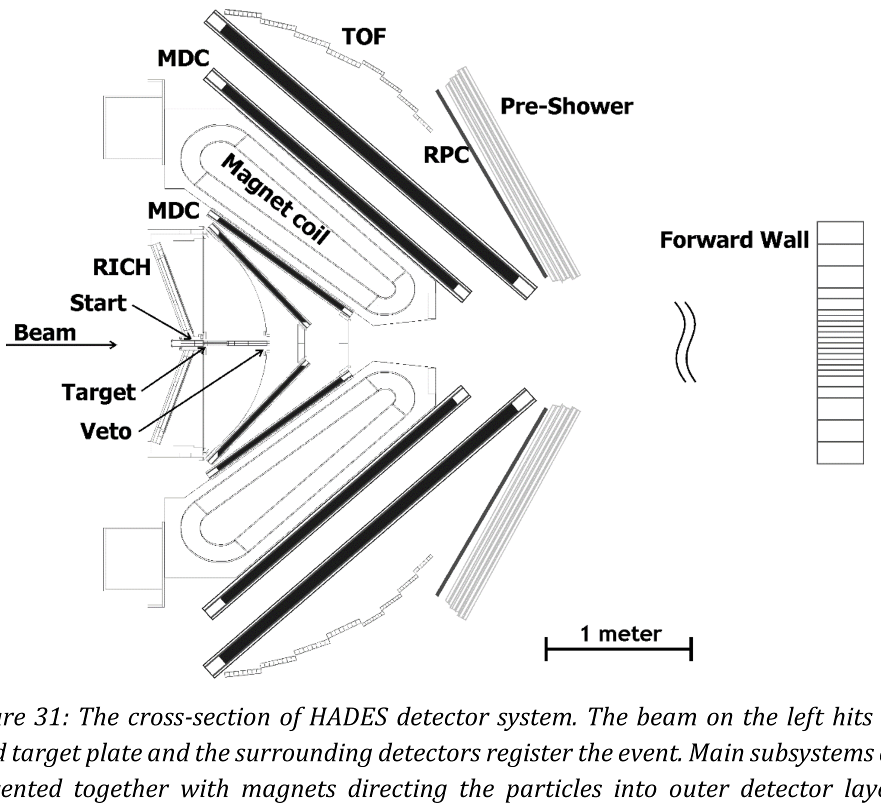


Figure 31: The cross-section of HADES detector system. The beam on the left hits the fixed target plate and the surrounding detectors register the event. Main subsystems are presented together with magnets directing the particles into outer detector layers. [www-hades.gsi.de]

5.1.1 Subsystems

Each detector subsystem produces different output signals which have to be treated separately and then introduced to the TRB boards that unifies all the modules in the system. Hence different front-end electronics, whether it were separate modules or in a form of TRB Add-ons were designed for individual subsystems. Let's now take a closer look at each detector (Figure 31) and its readout scheme:

- **Start and Veto**
  Those are the first and the smallest detectors in HADES, located close to the target. Each consists of a diamond plate divided into 8 channels which require amplitude discrimination and then time measurement. As the analog signals from the detector are very fast and small, the NINO (28) chip based front-end has been chosen. Its digital output is provided to the HPTDC on TRB2 running in very high resolution mode (25 ps). As this thesis is being written, the TRB2 has been exchanged into TRB3, which delivers higher measurement resolution of 10 ps.

- **RICH (Ring Imaging Cherenkov) Detector**
  The detector (37) has a high channel density of 28,500 channels read out by front-end electronics (AVP) containing amplifier modules. As the detector is located very close to the reaction area, it was important to convert the analog signals to their digital representation at the earliest stage. That's why the front-end electronics are connected directly to ADC Modules mounted on the detector sectors. No need for long lines decreases a risk of deformation of transported signals. Each ADC Module collects data from up to 16 front-end boards and has a direct connection to the HUB Add-on collector through an SFP optical fiber. A Lattice FPGA controls the readout and implements the TrbNet protocol in order to communicate with the rest of the system.

- **MDC (Multi-wire Drift Chambers) Detector**
  Detector consisting of four layers of drift chambers, positioned along the beam line axis, each divided into six sectors forming a cone. More than 27,000 channels in total are being readout by front-end electronics equipped with ASD8 chip that amplifies and discriminates the analog signals. Those are next digitized by TDCs, which output is processed by small Lattice ECP2 FPGAs and transmitted over POF (abbr. Plastic Optical Fiber) fiber links. All these modules are located on the detector. The fiber links are connected to the MDC HUB Add-on concentrator board, which is an extension board for TRB. The MDC HUB contains a direct Gigabit Ethernet gateway link for direct transmission of collected data to the event builders.

- **RPC (Resistive Plate Chamber) Detector**
  The main elements of the readout of this subsystem are the TRB2 boards equipped with HPTDCs for time measurement. The front-ends mounted on the detector contain amplifiers, shapers and programmable discriminators, which produce a digital signal out of analog ones that surpassed the imposed threshold voltage. The differential LVDS lines carries the digital output signals to remotely located TRB2 which performs the measurement of signals coming from several front-ends and then transmit the packed data via TrbNet link to the HUB Add-on module which converts it into UDP packets and sends further via Gigabit Ethernet link.

- **TOF Detector**
  The detector consisting of photomultipliers is readout by the same chip as Start and Veto detectors but organized in a different way. Sixteen NINO chips are mounted on a TOF Add-on which is an extension module for TRB. A pair of such add-on together with a TRB2 board forms the readout of one TOF sector. The analog signals and amplified and discriminated by the NINO chip which output in a form of LVDS digital signals is measured by the HPTDS on TRB2. The data collected on TRB is then transferred to the event builders over the HUB Add-on

- **Shower Detector**
  The detector is composed out of 6 independent sectors and each sector has its own individual readout path. A sector consists of 3 layers of matrixes 32x32 pixels. This sums up to 3072 pixels per sector and a total of 18432 pixels per the entire subsystem. Signals from 32 pixels (one row) are processed by a dedicated Front-end Board that shapes the analog signals and presents them on the output, one by one via a multiplexer. Those output signals are then digitized by the ADC module mounted no Shower Add-on boards. As significant part of the Shower detector (38), (39) readout firmware was developed by the author, a more detailed description of this subsystem is presented in Section 5.1.5.

- **Forward Wall Detector**
  The readout of this detector is realized in the exactly same way as TOF.

5.1.2 The DAQ System Structure

All the above mentioned subsystems have TRB modules in common. Those are base elements of HADES DAQ system as they provide remote control of all connected devices and provide essential TrbNet connectivity. The setup (Figure 32) is built in a hierarchical way with one central HUB which is connected to the HUBs in the subsystems that concentrate several boards in the next layer. One central point of the system is needed in order to distribute the readout requests messages to all the electronics as well as serve as an interface between TrbNet and a PC for the Slow Control. A message sent from a PC can be addressed to any TrbNet endpoint in the system but also can be sent as a broadcast to a subgroup of endpoints located in various detector subsystems. Those messages enter the central HUB and then find the way to their destinations through a series of concentrator modules. The TrbNet network is also used as a channel for the detector data readout in the parts of the system which does not have a direct Gigabit Ethernet interface (the components with GbE gateways are denoted by orange color). In such case the first HUB to which the endpoints are connected serves as an Ethernet gateway. While the MDC and Shower are examples of the systems which introduce Gigabit Ethernet at the first stage of concentration, right after the digitizers, the other subsystems, especially those using TRB2 need a HUB to transmit the collected data out to the event builders. Thanks to the structure based on HUBs connected in a hierarchical way, the setup is very flexible and scalable. When needed, another HUB can be inserted at any place, opening several slots for additional electronics in the system.

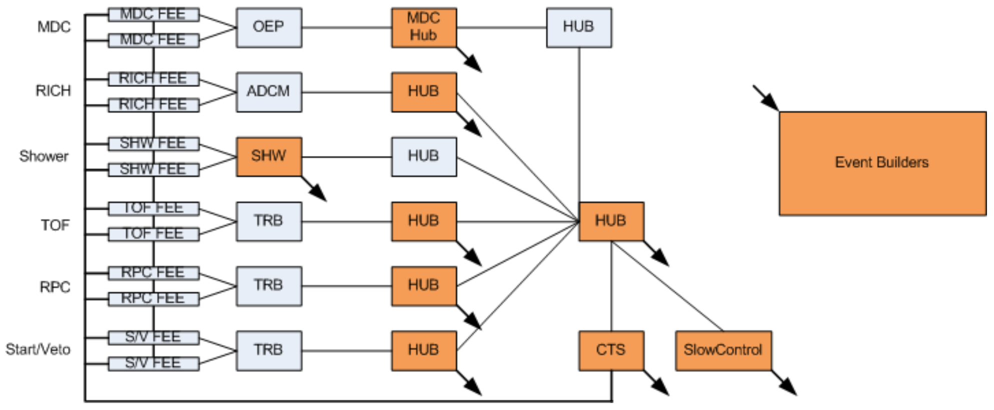


Figure 32: Schematic view of the HADES experiment DAQ System. Different detector subsystems have individual readout paths, sharing the same type of components when possible. The hierarchical structure is shown by the connections from the main hub, through several layers of concentrators, reaching the endpoints. The components, marked in orange, are the ones capable of transmitting readout data directly to Event Builders through Ethernet network.

5.1.3 Trigger System

The characteristics of the HADES experiment such as the reaction type, expected trigger rates (up to 100 kHz) and data volume (up to 700MBps) were the reasons of deciding that the readout will work in a single-level, triggered mode. Three subsystems: Start and Veto, RPC and TOF were chosen as the detectors that provide the multiplicity information, derived from number of fired channels, about the event type. This information is delivered to the CTS (abbr. Central Trigger System), which decides if the readout procedure should be initiated. This whole procedure has to be performed within a strictly defined period of time as the information has be delivered to the endpoints before they lose track of the current event. This information is delivered to the CTS, which based on pre-defined conditions, decides if the event is worth further processing. By accepting a trigger, the system is blocked for upcoming events until all the endpoints report that they properly finished the readout. This mechanism is realized by asserting a busy flag while the electronics process an event and when they are finished, each endpoint sends a release message to the CTS. When release messages from all subsystems are collected, the CTS is ready to accept a next trigger. The time which takes for all subsystems to reply is called the dead time and defines the maximum acceptable trigger rate. It consists of the dead time of the FEE modules, readout and the data transmission and is different for various subsystems components. This will be discussed in details in the next Chapter 6 where the performance of the system is evaluated.

The triggering procedure takes several steps to accomplish. At first the current event has to fire enough channels in selected subsystem to surpass the multiplicity thresholds. If so, the CTS checks if the system is ready to process data and sends a fast, hardware signal to the endpoints called reference time and the corresponding trigger number carried in a readout request message, over dedicated TrbNet channel, needed for the event building. The hardware signal is adjusted in a way that it arrives in the same time to all the receivers. It is needed for synchronization of subsystems and further reassembly of different detector parts and event reconstruction. Then the CTS blocks incoming triggers and sends a LVL1 (readout request) message over the TrbNet links to all the system endpoints. Each module that receives a LVL1 packet, packs the collected data and transmits it to the event builders via the nearest Gigabit Ethernet gateway. After sending data out, the endpoints report the job done by sending back to the CTS a release message, which opens the system for next events.

The CTS is a fast processing and highly configurable TRB Add-on board that gives the operator a way of controlling the triggering procedure. It has many inputs to which the preliminary, hardware based trigger signals can be connected and many outputs to distribute the reference time signals. The main FPGA samples the inputs with 800 MHz and processes them with implemented trigger logic. The algorithm can be optimized by the operator according to the current reaction type, prioritizing given trigger patterns over the others. The board is also equipped with three optical links, one is used to receive Slow Control commands, the second one is the LVL1 trigger output link, which is connected to the central HUB for further distribution to all subsystems and the third one is available to send statistics over Gigabit Ethernet link.

5.1.4 Data Transmission and Event Building

The DAQ system consisting of 80,000 channels is aimed to run at the trigger rate between 20 and 100 kHz. Assuming that each analog channel is converted into a 4 bytes digit, the entire detector could generate as much as 16.5 GBps of data. That is a hypothetical situation, just a small subset of channels fires during an event and the rate varies over time as the beam structure is not constant. The HADES DAQ readout network and event building infrastructure allows to transmit and store up to ~700 MBps. Currently performed, high-intensity experiments, shows about 70% of that network bandwidth being used in most critical situations. System performance, as measured in experiment, is discussed in details in Section 6.2.

Each system endpoint generates a subevent as a response to LVL1 message. It contains data collected from some region of the detector. Those subevents are transmitted to machines called Event Builders, which collect individual parts of the entire events, match them together and save on storage. As Event Builders are server class PCs the natural way of transporting data is the 1/10 Gigabit Ethernet. Where it was possible the Gigabit Ethernet Modules were implemented closest to the first data collectors layers of the subsystems. Otherwise, the readout data has to be transmitted over TrbNet link to the nearest HUB, which results in longer event processing time, thus increasing the dead time, limiting the acceptable event rate.

5.1.4.1 HADES Data Format

Before taking a closer look at the mechanism of constructing packets, let's present the data formats for a better understanding of manipulations which take place in the process. The measurement data received from several endpoints by the concentrator is packed into a single unit called a subevent. One concentrator produces one subevent per trigger. Subevents collected from the collectors represent the entire event registered by the detectors in all subsystems. The measurement data in each subevent is encapsulated with Subevent Headers. They carry information about the subsystem where the data came from, trigger number needed to match with parts from other concentrators and some other fields like the amount of transported data. As a subevent contains data from several endpoints, those smaller parts can be called sub-subevents. The architecture of the presented DAQ system is hierarchical which means there can be several levels of concentrator boards on the way. Each board will encapsulate the received subevents with another layer of similar headers.

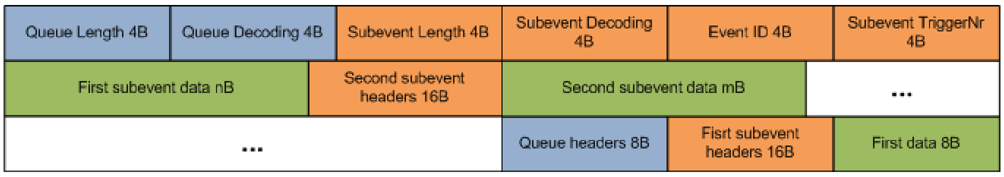


Figure 33: Internal data format of a Hades Transport Unit Queue. The Queue headers (blue) encapsulate several subevents (orange) which carry the payload (green). The structure can be extended to unlimited amount of subevents, but summing up to total 64 kB UDP packets. The last three elements are called a trailer and are repeated bytes from the packet beginning, appended for the consistency check.

The Gigabit Ethernet Module is a place where the subevent has to be packed into an UDP packet and all the protocols of lower layers have to be applied. The header format of the protocols stack was presented in the Section 4.5.1 about network model. In order to be properly recognized by the Event Builder, the subevent header has to be reformatted and formed into a unit called Hades Transport Unit Queue (abbr. HTUQ), which is another level of encapsulating (Figure 33). It is needed because one HTUQ can transport more than one subevent. Such functionality is introduced in order to reduce the fragmentation. In response to LVL1 message all subsystems have to respond with a generated subevent. In case there is no data to transport an empty subevent is produced anyway. In such case, the overhead coming from headers added at each step of encapsulation is significant. Consolidation of such empty or small subevents optimizes the bandwidth usage, lowers the load on Event Builders CPUs and reduces dead time. The GbE Module can be configured to run in a single or multi-event mode.

5.1.4.2 Gigabit Ethernet Module

In the HADES DAQ system, the Gigabit Ethernet Module is designed only to transmit the readout data collected by the electronics to the Event Builders. Hence, the lightweight module, not including the receive channel or any additional protocols was implemented on the Gigabit Ethernet gateway electronics (Figure 34). As it was already stated, the GbE Module is included in all the boards of the system that were equipped with an available and properly connected SFP fiber link transceiver to an FPGA. Those are: HUB Add-on, Shower Add-on and MDC Add-on. The base board TRB2 itself, features only one optical fiber connector, which has to be used with TrbNet protocol, thus one has to rely on the Fast Ethernet connection through Etrax processor.

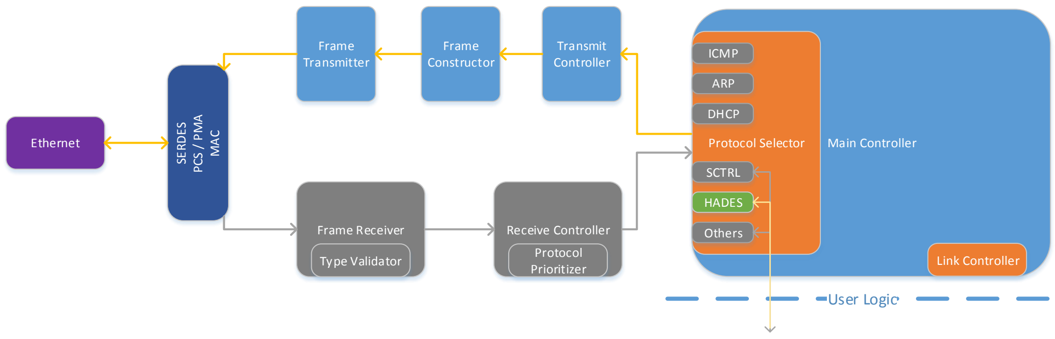


Figure 34: Schematic view of the Gigabit Ethernet Module adjusted for basic data readout on HADES DAQ network gateway electronics. The elements marked in grey are removed from the base, duplex design. The readout data is collected from the User Logic part and passed through dedicated HADES TrbNet Data Response Constructor protocol implementation.

As the GbE Module is implemented always on some stage of collector boards, the collected subevents come through TrbNet links, physical or logical. The logic infrastructure of TrbNet protocol introduces a TrbNet HUB component (Figure 35), which is the central point where the data from endpoints resides. It features a streaming interface to which GbE Module is connected in order to capture the processed subevents stream. Like other parts of TrbNet protocol, the communication through this interface works in a handshake way, the GbE Module as a client asserts a flag whether it is able to accept incoming data, then the hub pushes 16 bit words at 100 MHz. De-asserting this flag holds the hub, at the same time extending the event processing time, which affects the dead time of the entire system. That's why it was important to prepare the GbE Module together with the Response Constructor for the data readout, in a way that it does not introduce any additional latency.

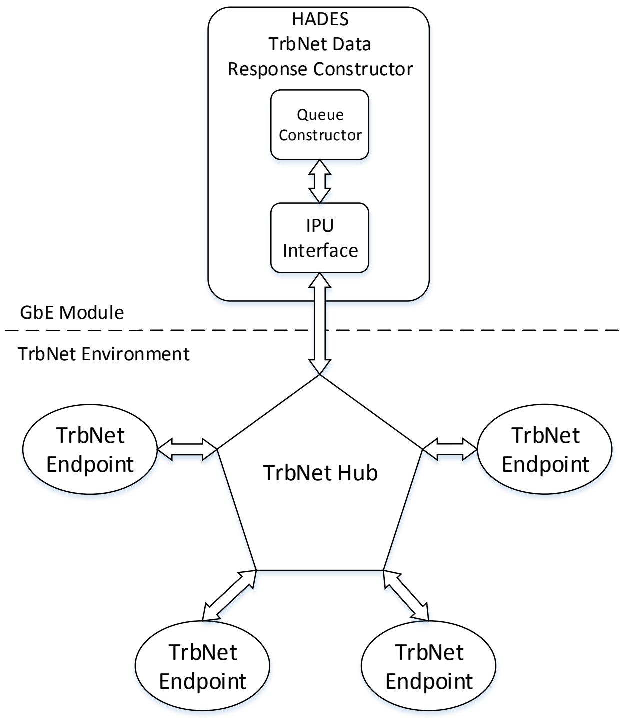


Figure 35: The interface between TrbNet logic and the Gigabit Ethernet Module. A TrbNet HUB instance collects data from several endpoints in the design and passes through IPU Interface to the HADES TrbNet Data Response Constructor protocol implementation in the GbE Module.

The Response Constructor developed for handling the readout data is responsible for converting the incoming subevents stream into properly formatted payload of UDP packets. The IPU Interface component keeps the communication with TrbNet Hub and stores the incoming subevents in a FIFO buffer. As TrbNet logic can run at different clock frequency than GbE Module, the buffer is needed not only to store the data but also in order to separate those two clock domains. The operation of the component is divided into two state machines. One saves the incoming subevents while keeping a proper communication with TrbNet component. The second one starts work at the moment there are any subevents waiting in the buffer. Its responsibility is to read the subevents from the buffer and forwarding them to the Queue Constructor. This process can be executed in two modes: single or multi-event. In the first mode, just one subevent is taken from the buffer and passed to the Queue Constructor. The constructor prepares all the needed headers and stores the ready HTUQ in an output buffer. As the Response Constructor interfaces with the Protocol Selector bus in the same way as all the other protocols implementations do, it signalizes that the payload is ready and the packet construction starts as it was described in Section 4.5.2. When the multi-event mode is enabled, the subevents are readout from the first buffer as long as one of the following condition happens:

- The HTUQ reaches 64 kB limit which is the maximum UDP packet size
- The trigger type changes, one queue can transport only subevents collected for one type of trigger
- Destination Event Builder address changes

The entire process of reading the subevent out from the TrbNet Hub and constructing the HTUQ works in pipelined way. From one side the subevents are saved into input buffer, while a separate state machine takes those subevents and constructs final units storing them in an output buffer. Those two processes are connected only by the fill level of the buffer which signifies the availability of subevents to process. The HTUQ construction is a data driven process, the subevent has to be first readout and its headers are required to be analyzed in order to take decision about extending the currently built queue by its payload. The order of construction, together with data units is presented on Figure 36.

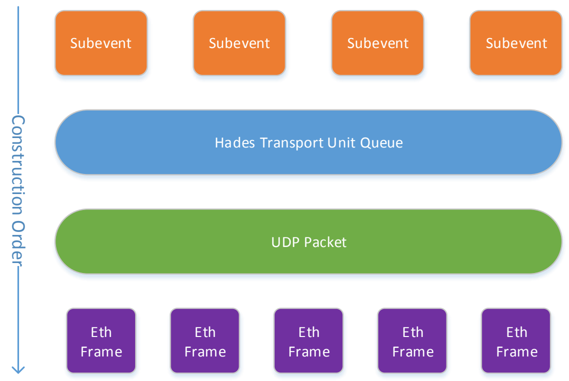


Figure 36: Overview of data units in the readout chain. Single subevents (orange) are combined into Hades Transport Unit Queue (blue), which is then converted into an UDP packet (green). Large packets of up to 64 kB are fragmented into Ethernet Frames (purple) of 1.4 kB.

5.1.4.3 Event Building

The high amount of data generated by the electronics, reaching 700 MBps at its peak, require an efficient network infrastructure (Figure 37) as well as processing power. The event fragments sent over 1/10 Gigabit Ethernet have to be received in one place and reassembled into a unit representing an entire events. Therefore, eight server class machines work as Event Building machines (40). Each one is equipped with two six-core CPUs, 64 GB of RAM, 44 TB of disk space and two 10 Gb network interfaces.

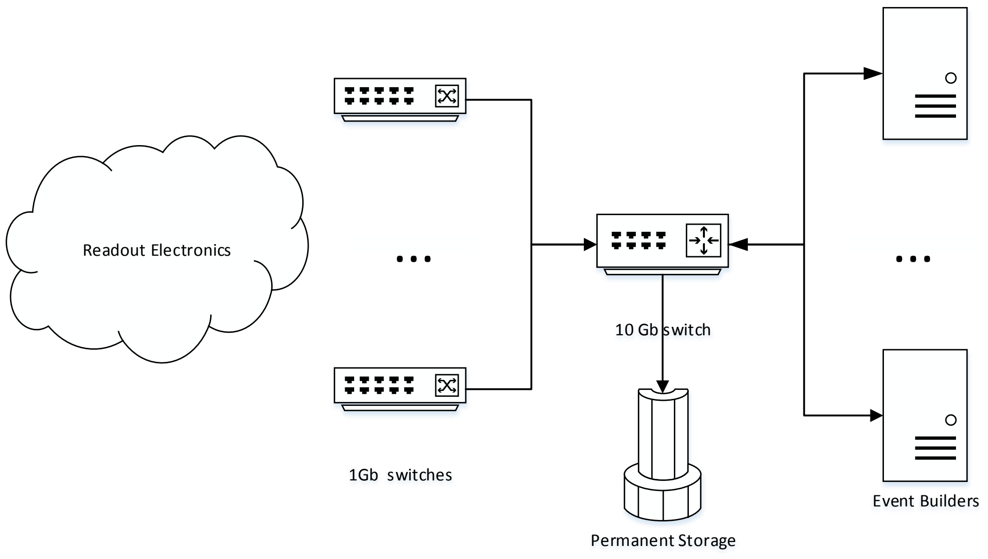


Figure 37: Schematic view of the HADES Event Building facility. Data from readout electronics is transmitted through 1 Gb Ethernet network switches and collected with a master 10 Gb switch. From there, the data is distributed to a farm of Event Building computers. Collected data on Event Builders is then transported to permanent storage.

Two data concentrators containing Gigabit Ethernet Module are connected to one 1 Gb network switch, those switches are in turn connected to one 10 Gb switch which distributes the received packets to the Event Builders. As the experiments can run and collect data over months, there is a need for some kind of permanent storage. The reassembled events are temporarily being saved on the Event Builders hard drives. Next the data is moved over the 10 Gb switch to the permanent storage, which in case of HADES is twofold: a tape storage and a computing cluster.

The Event Building machines run the event building software. The software listens on a specified ports for incoming UDP packets and combines subevents into entire events. The constructed events are being saved on local hard drives in a form of binary files. In order to properly process all the incoming data flow, each EB machine can run several instances of EB software in parallel. This means there are several destination for the packets that are being built in the data collectors by the GbE Modules. In order to distribute the load on different EBs, the round-robin mechanism has been implemented in the system. Each GbE Module is configured with the destination addresses of all the EB machines and processes. The Central Trigger System which generates the LVL1 messages, includes in its headers an address to which EB process the currently collected event should be sent. This information arrives to all GbE Modules which acquire the proper addresses from their memory and construct the network headers of the outgoing packets accordingly. The CTS can be configured for how many successive events should be sent to one EB process before changing the address to the next one. In this way, the entire load is evenly distributed to the number of running EB processes. Another way of optimizing the network and CPU load which was already discussed in a section above, is the construction of multi-event queues. Small, highly fragmented packets affects in a significant way the performance of the Event Building machines, which under high load start losing events. Configurable, multi-event packets help reducing the los by a better balance of the network traffic characteristics.

5.1.5 Digital Signal Processing Platform

An interesting example of data processing on FPGA is presented by the ADC platform developed for the readout of Shower subsystem of HADES detector, the Shower Add-on (39), (38). The subsystem is equipped with a readout that has been updated for the needs of modern experiments. Some of its parts, like the front-end electronics remained the same, but the digitizers and data collectors had to be redesigned in order to meet the higher trigger rates, measurement precision and comply with the introduced TrbNet system. A subsystem based on Shower Add-on readout is evaluated in Section 6.2.

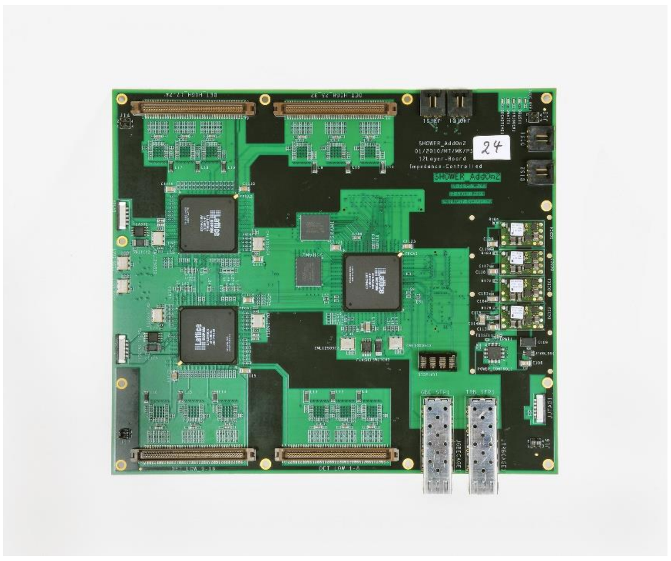


Figure 38: Photo of Shower Add-on board. Four brown connectors deliver analog signals to ADCs which are controlled and read out by the FPGAs. One central FPGA is a controller and an interface to both TrbNet and Gigabit Ethernet network, through the optical connectors.

5.1.5.1 Electronics

The front-end electronics for the Shower detector consist of FEBs (abbr. Front-End Boards) equipped with dedicated ASIC. Its function is to convert the received analog impulse charge into an output signal with a proportional amplitude and width. One ASIC has 32 inputs which after conversion are transferred in a serial manner by means of multiplexer on one output to the digitizer. The front-end boards contain also the current source which is used to provide calibration impulses to the inputs of the ASIC. By injecting impulses with a known charge one can calculate the individual differences and offsets introduced by each channel.

As digitizers, the 8 channel ADCs are used with 10 bit resolution and sampling at a rate of 40 MHz. Twelve of such devices are mounted on Shower Add-on board (Figure 38). They are grouped by six and connected to two Lattice ECP2 FPGAs, which control their operation and gather the measured digital data. The board is designed in a symmetric way, thus the same firmware can run on both FPGAs. The third FPGA is used as a controller and a hub for the other two as well as an interface for the TrbNet network and a Gigabit Ethernet gateway. In order to handle both network connections, two SFPs for optical fibers are installed on the board. The readout chain is presented on Figure 39.

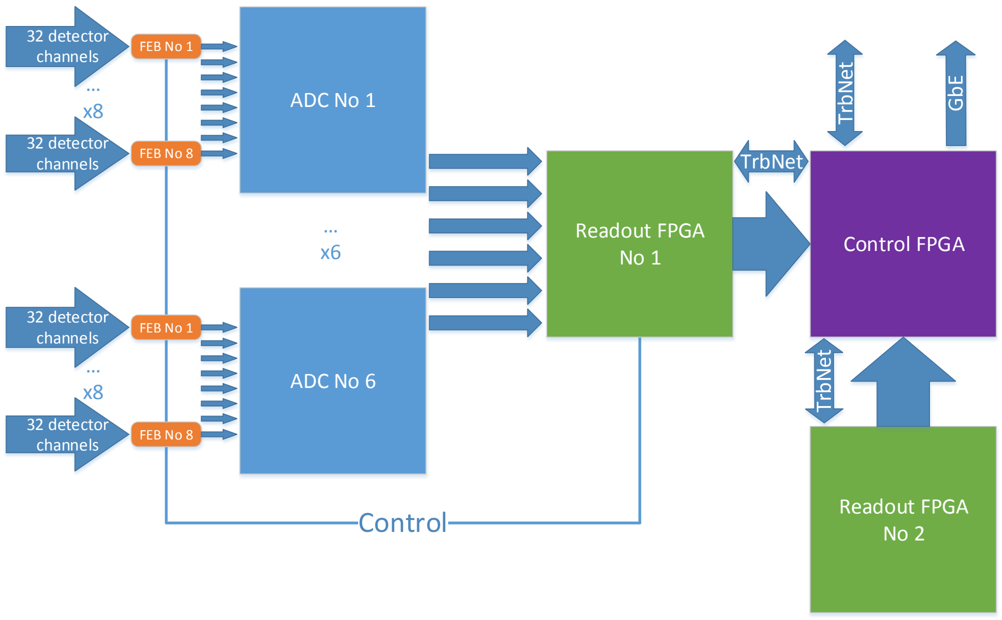


Figure 39: Schematic view of the Shower readout chain. 32 analog outputs from the detector sector are amplified and shaped by the Front-End board, which outputs are digitized by ADC. The data stream with samples is readout by the Readout FPGA, which acts as a TrbNet Endpoint. The Control FPGA has an instance of TrbNet HUB implemented that gathers the data from Readout devices and transmits it over Gigabit Ethernet directly to the Event Builders.

5.1.5.2 Firmware

Important part of the Shower readout system is the firmware that runs on the FPGAs and implements the mechanism of communication with digitizers. Each FPGA has six ADCs connected to its inputs. Eight ADC input channels are digitized and the results are transmitted to the FPGA as eight serial streams of bits in a Dual Data Rate (abbr. DDR) mode, each representing a digital value of measured amplitude. The main component of the FPGA consists of six instances of the modules controlling the ADCs, the FEB Controller module and the TrbNet endpoint infrastructure. Each ADC

Controller module contains eight DDR buffers, one per ADC channel. They de-serialize the incoming bit stream independently, only sharing the same clock signal. Each decoded value is compared to a threshold value that is kept in a memory block to provide zero-suppression. If the value is higher than the threshold, it is tagged with the module and channel number and stored in a buffer for the readout, otherwise no data is generated.

Each incoming hardware trigger signal that arrives to the Shower Add-on starts the work of FEB Controller module. It implements a precisely timed state machine that sends a hold command to the ASICs ordering them to lock the converted amplitudes and then to iterate over all channels via multiplexer presenting its signal at the output one after the other. The iteration advances after receiving a signal from the FEB Controller. There is a fixed delay between the updated ASIC output and its digitized value by the ADCs. This delay is taken into account while de-serializing out the ADC samples by the FPGA logic.

There are three different modes of operation of the Shower readout and related algorithms that were implemented. The selection of the mode happens by sending a specified trigger type from the Central Trigger System:

- **Standard data collection**
  The inputs from the detector are converted and locked by the ASICs, which are then digitized by the ADCs and analyzed by hit detection algorithm (comparison to pre-defined threshold level). In order to acquire samples representing the proper input signal amplitude, several samples are collected and the mean value is calculated. Such functionality is important for timing adjustment between the real signal appearance and the moment of the readout.

- **Calibration trigger**
  As a response to the calibration trigger, the test impulse is injected at the ASICs inputs instead of the signal coming from the detector. The impulse is first applied to the even channels and then switched to the odd channels. Such procedure is very helpful for detector calibration and diagnostics of the readout system.

- **Pedestal trigger**
  As a response for the pedestal trigger, a procedure of collecting 64 successive samples for each channel is launched. Calculating the mean value of those 64 measurements provides the information about the level of noise on each channel and thus defines the threshold levels.

The data collected in the ADC Controller modules buffers is read out after the LVL1 trigger messages arrives. As the FPGAs act as TrbNet endpoints, they implement standard modules for interfacing the data readout channel of TrbNet. Response construction consists of reading out one buffer after the other and pushing the data into the TrbNet interface as one block. Two readout FPGAs are connected to the central FPGA on which the TrbNet Hub module is implemented together with GbE Module which receives the content of the buffers and directly constructs UDP packets, acting as a first and the last stage concentrator.

5.1.5.3 Other Applications

The Shower Add-on stands as a flexible platform for any kind of measurement requiring multi-channel ADC sampling. One of such applications is the RPC PET (abbr. Resistive Plate Chamber based Positron Emission Tomography) prototype developed by the Laboratorio de Instrumentacao e Fisica Experimental de Particulas at Coimbra, Portugal. It is an advanced project, already showing very promising results (41), which is a great example of how technologies developed for physic experiments can be applied in other science fields like medical imaging.

5.1.5.3.1 RPC-PET

The modern, commercially available PET scanners are in most cases built as matrixes of small crystals with connected Photomultipliers (abbr. PMTs). The two co-liner photons emitted from the Beta+ source implemented in human body, hit the crystals which, with the help of PMTs generate electrical pulses. Those signals are captured by the DAQ system, processed and stored. The analysis process applies several algorithms, which basing on the position of crystals which were hit, allows to calculate the spatial distribution of source of emitted photons, which in turn leads to reconstruction of a body, where existing anomalies can be found by professional doctors. The approach introduced by the RPC PET project, is based on the use of a different detector type (42). The Resistive Plate Chambers are composed of several layers of glass plates with gas in between them. The plates are covered by electrodes, organized in strips or pads, which collect induced current produced by moving electrons generated by the passing particles and therefore producing an electric impulse. The RPC detectors are relatively cheap in comparison to crystals and photomultipliers.

The prototype setup built and tested in LIP laboratory consists of four chambers, each having sixteen layers of plates. It is a small scale detector designed for small animal scanning and as a proof of technology. In total 192 detector channels are processed by two readout chains. First one analyzes the signal shape. Front-end boards, equipped with amplifiers and shapers are connected to ADCs on Shower Add-on boards. The second chain is used for the Time-Of-Flight analysis. Passing through different front-end boards, the signals are delivered to TDCs mounted on TRB boards.

While the TDCs operate in a standard way measuring time used for offline corrections, the firmware running on Shower Add-on FPGAs had to be adjusted for this particular application. Instead of collecting one sample for the amplitude of the input signal, several samples are transmitted, from which the entire analog signal can be reconstructed by offline analysis. The ADC Controller Module is equipped with additional buffers, one per input channel. The samples produced by the ADC at 10 MHz rate are constantly being written to the buffer, which work in ring buffer mode. That means it collects a number of samples until it get fill up and then each next sample is written while the space for it is made by removing one sample from the front. When the trigger signal arrives, the ring buffer is closed and another number of samples is collected. The result of such operation is a defined number of samples acquired before the trigger and after (Figure 40). Both amounts of samples are configurable via Slow Control. Such mode of operation is close to mode of digital scope. Collected samples are the input for Digital Signal Processing algorithms that can be either implemented as the FPGA logic or as offline procedures.

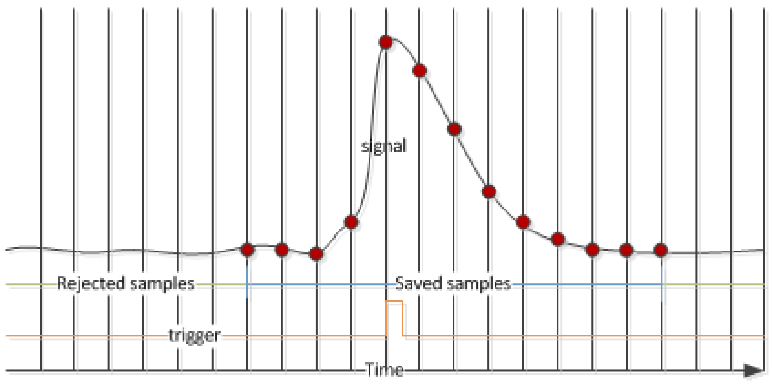


Figure 40: Sampling mechanism implemented in Shower Add-on for RPC-PET application. Configurable amount of samples on the input signal are collected before and after the trigger signal arrives.

The entire DAQ system for the RPC PET prototype is based on the electronics developed for HADES with an exception for the front-end electronics. The main components such as TRB boards, HUB and CTS Add-on are used together with the TrbNet protocol are used as a backbone for the system. The collected data is readout by the Gigabit Ethernet Modules and Event Building software is running on a PC for data storage.

The first measurements performed with RPC PET for small animals show sub-millimeter spatial resolutions of 0.55 mm achieved, which compared to known commercial solutions that present close to or more than 1mm is a world class result. Another key aspect is the cost of the detector which is several orders of magnitude smaller than in case of crystal based solutions.

5.1.5.3.2 Electromagnetic Calorimeter for HADES

Another application of the Shower Add-on is the prototype of the readout for the Electromagnetic Calorimeter (43) built as a replacement detector for Shower in future HADES experiment. As Shower detector reaches its lifetime, it has been decided to replace it with a detector composed of a matrix of large lead-glass modules to measure electron or photon energy by means of total charged deposition of an electromagnetic shower. Similarly to the PET project, analog signal from a single module has to be sampled in oscilloscope mode by the ADC and additionally the TDC measures the precise time when the analog signal crosses the applied threshold. A special front-end board has been designed, with the functionality of two measurement chains. One chain amplifies and shapes the analog signal for a proper ADC sampling, while the second chain includes a comparator which is used as discriminator and generates a digital impulse for the TDC. One output of the front-end is connected to the Shower Add-on and the second output leads to the TRB2 board for time measurement.

The firmware running on the ADC Controlling FPGAs of the Shower Add-on contains the same functionality as presented in the RPC PET section above. It is additionally extended by an experimental module that implements Digital Signal Processing algorithms that are executed in real time on collected samples for each trigger. The algorithms provide additional information like: total area under the pulse, maximum of the pulse and time measurement via Constant Fraction technique, about acquired measurement, which is transmitted together with the samples values to the Event Builder.

When it comes to measuring time when analog impulses start their rising edge, the effect of so called time-walk plays an important role (Figure 41 - left). Having two analog signals which start at the same time but have different amplitudes, the time when they cross the same threshold will vary, leading to an imprecise assumption about the exact time of the creation of the signal. It is a well-known effect and several different techniques for correction have been developed. One of such techniques is the offline Constant Fraction algorithm, which can be performed on a number of samples collected on the rising edge of the analog signal (idea of the algorithm is presented on the right side of Figure 41). Simulations show that eight samples are enough to properly approximate a curve similar to the rising edge, thus calculate a correction for its starting point time.

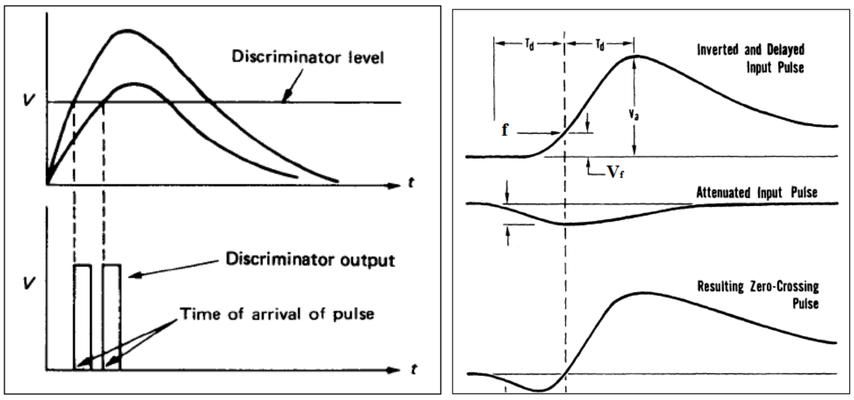


Figure 41: Presentation of time-walk effect. On the left side, a constant threshold level discriminator operation is show. For two signals with the same starting point but different amplitudes, the time of threshold level crossing is different. On the right side, the analog process of building constant fraction discriminator output is presented. The input signal is inverted, attenuated, shifted in time and then combined with the original signal. The device generates an output pulse at the zero-crossing point, independent of the input signal amplitude. (50)

5.2 J-PET Scanner

One example of PET scanner prototype has already been presented in the section about the Shower Add-on applications. The one described below is a large project realized by the Jagiellonian University in cooperation with partners from Warsaw and entities from commercial sector. The project introduces the use of plastic scintillators instead of expensive crystals used in commercial solutions. The use of precise time measurement for calculating the TOF (abbr. Time Of Flight) of gamma quanta helps achieving comparable space resolutions at much lower cost of the device (44). The entire project consists of developing new kinds of scintillators by the chemistry department, building up the prototype detector, composing an efficient DAQ system and implementing software for reconstruction and visualization of collected data (10).

Described below data acquisition concepts for PET scanners are a subject of both international and Polish patent applications which were stated and edited by the author of this thesis (7).

5.2.1 Setup

The first prototype detector is composed of long scintillator strips of rectangular shape with photomultipliers attached at its ends. There are 24 of such modules arranged in a ring around the source of radiation. Those modules sum up to 48 individual PMT channels that need to be processed by the TDC measurement. The second prototype is under construction and will contain up 400 photomultipliers. A schematic view of the readout chain is presented on the Figure 42. Prior to digitizing, the signals have to be discriminated in order to get rid of the background noise. There are two solutions under investigation: multi-level and constant-fraction discriminator front-ends. Both are needed for very high precision time measurement as countermeasures for the time-walk effect which appears in single-level threshold discriminators. The multi-level solution splits the input signal from the PMT into several paths, where each one applies a different threshold. Their properly adjusted values, allow to predict the initial shape of the input signal and thus apply corrections to the results. As the time-walk effect is strongly driven by the amplitude of the input signal, one solution is to discriminate the signal on fraction of its amplitude instead of a constant level.

Three different front-end boards were developed for the purpose of this research. The first one implements the constant-fraction approach. It has 8 input channels where each one is split into four different, adjustable fraction levels, summing up to 32 outputs to the TDC. This board is developed as a proof of concept and is under investigation. The second board comes in a form factor of VME 6U and fits into a standard crate with a custom-made backplane. One front-end board features 3 input channels, each put through multi-threshold circuits and conversion from charge to width. Four independent and programmable thresholds are used for gathering four points on the leading edge of the input signal, which is enough to extrapolate it in order to find the correction for the time-walk. There is an additional path for each channel which performs the conversion from the charge carried by the analog signal into the width of the output digital signal. The conversion in the selected charge range is linear, thus the value can be used for calculating additional correction values. Four of such front-end boards can be connected to a single TDC on TRB3, through which they can also be programmed with threshold values. The third front-end solution is an experimental method of using the LVDS buffers which exist on the input pins of the FPGA devices as discriminators. The board is a mezzanine add-on for TRB3 and includes passive splitters and programmable DACs. The analog signal is connected to one input of the LVDS buffer and the programmed threshold is applied to the second input. The LVDS buffer generates a digital pulse in case the crossing of those two signals happens. This pulse is then measured by the TDC channel implemented in the same FPGA. This front-end also implements multi-threshold solution by splitting each input channel into four separate paths with individual thresholds.

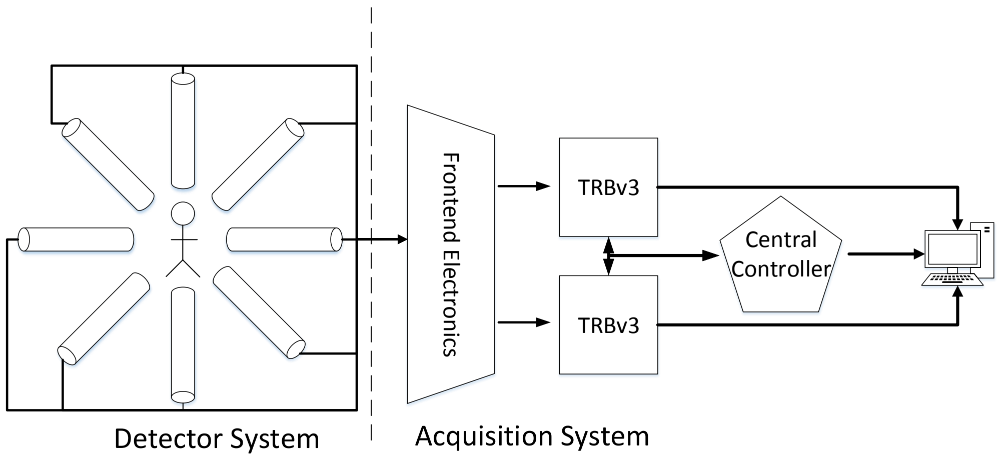


Figure 42: Schematic view of the J-PET prototype DAQ system. The analog signals from the detectors are discriminated by the Front-End electronics which are digitized by TRB3 platform. The readout process is controlled by Central Controller board, which is also receiving a copy of the collected data for the online analysis features.

5.2.2 The Readout

All three solutions share one thing in common, they are connected directly to TRB3 board with TDC firmware running on the edge FPGAs. In order to readout the entire prototype, the setup must be composed of a number of TRB3s which need to be controlled and synchronized. It has been decided to build a trigger-less system with one common clock distributed to all the TRB3s as a reference signal. In this mode the TDCs open their measurement window for 20 us repeatedly at 50 kHz which is the frequency of the reference signal. Taking the negligible dead time between two consecutive measurements into account, the TDC operates in this mode capturing detector hits during the full period of measurement. This mode is especially efficient when the electronics produce reasonable data volume. Triggered systems produce much less data at the end but can filter some events which could be helpful during offline analysis. Processing all the hits and events can result in better spatial resolution of scanner within shorter time of the measurement. In order to help dealing with that amount of data a Central Controller board is under design. Two main tasks assigned to this board are distribution of reference signal and of the trigger tag to the TRBs and collection of the readout data. The Controller will act as a source of the common clock as well as additional packets distributed to all the boards in the system with the trigger number and control values needed for the synchronization of the system. The heart of the board is a Xilinx Zynq System-On-Chip FPGA. This is a new development which combines the standard FPGA logic resources together with a Cortex ARM processor. It is a powerful solution giving the ability to implement data analysis in high-level programming languages with an underlying data collection and processing mechanism realized in programmable logic. The incoming data is gathered through 16 optical links which are handled by the FPGA logic. The logic can act as the event builder by combining the subevents coming from connected TRBs into single units. Such unit can be sent directly further to the permanent storage or a PC but its copy can also be delivered to the embedded processor via a high-speed interface. The processor can perform feature extraction algorithms providing additional information for the offline analysis. The processor can also act as a filter deciding if the currently collected unit contains valuable information and if it's worth to processed forward or just drop in order to save bandwidth and disk space.

An evaluation setup has been constructed, consisting of four TRB boards and its performance was measured through a series of tests. The procedure and the results are enclosed in Section 6.1.

5.2.3 Central Controller and Online Event Building

The backbone of the DAQ system for this PET prototype is based on main components described in this work. The TRB3 platform is used together with the firmware and software which comes from other experiments and is adjusted to this particular system. This shows a great flexibility of the platform and synergy between different applications. The new element is the Central Controller, which is similar to the HUB2 Add-on board of the HADES system, featuring many optical connections, but the difference is in the FPGA on the controller which provides much more processing power than only multiple link handling. The next-gen family of FPGAs with the embedded ARM processor opens a new way of processing data and interfacing with the system. One application which can take advantage of such component is the online event building and data unpacking. For the presented PET DAQ system, it is crucial to optimize those processes as the amount of data from a single measurement can be overwhelming due to the trigger-less mode. Standard scenario would apply several PCs as event builders and distribution of collected data evenly over those computers. After the measurement is finished, the process of unpacking can be started in parallel on those machines in order to convert the binary data into high-level objects. First drawback of such solution is the lack of the online monitoring possibility as the data is reassembled and stored on PCs. Any feature extraction requires the unpacking preformed first which takes excessive amount of time making it unusable for any kind of online analysis. The second aspect is the amount of storage needed to save non-filtered data.

A solution to that is the Central Controller which has enough input links to collect data coming from all the TRBs in the system. The characteristics of the FPGA devices provide the facility to decode each stream individually in the real-time. Network interfaces present received data words at a constant transmission clock frequency. The format of the data coming from TRB3s is optimized for word-by-word processing. That means there is no need for additional buffering of the entire portion of the data prior to its decoding. The decoding process recognizes the beginning of a packet and knowing the structure of the incoming data can easily gather the words presenting the input channel number and the value of measured time. Fetching the size fields of the header, the process can dynamically adapt to the content of the packet without introducing any additional latency.

Decoded values are then stored in a memory block shared with the ARM processor. The memory block can be organized in segments representing events and arrays with input channel number and its measured time. On System-on-Chip devices with such architecture a Linux kernel can be launched with the framework needed for production of the high-level data structures from the values saved in the shared memory. Having decoded parts from all the TRBs in one place is a starting point for any kind of processing, filtering and feature extraction algorithms.

6 System Performance Measurements

Originally the system was designed to fulfill the requirements imposed by HADES experiment. Several conducted in-beam measurements have proven that the concept is correct and that the proposed architecture together with the implemented mechanisms allows to efficiently collect data from a complex measurement system at high rates. The significant number of other applications where the TRB platform is being used, confirms the assumptions about the scalability and the flexibility of the system.

Below, two setups are described, used for tests of the performance of the system. The first one is a laboratory setup, where characteristic of the TRB platform could be measured, without the influence of the factors connected to specific detector readout chain, i.e. without front-end electronics. The data was generated internally by the TRB boards using the built-in functions implemented in FPGAs, as described in details below. The next test was performed on the full HADES detector system. The results give a valuable point of reference, while comparing to the one taken in clean, laboratory conditions.

6.1 Laboratory Measurements

A setup consisting out of several TRB3 boards has been constructed in order to perform performance measurements. Being a versatile platform, TRB3 could be used as both hubs and endpoints, depending on the firmware configurations loaded to appropriate FPGA devices. In total 4 TRB3 boards were composed into a setup presented in Figure 43. One board, called Master contains firmware modules performing functions of the Control Module and of the Central Controller. Both modules, together with a hub logic module were instantiated inside the central FPGA of the Master board. One edge FPGA is configured as a hub and communicates with slave TRBs through optical connections provided by SFP Mezzanine Add-on. Each slave TRB has a hub instance in the central FPGA and TDC modules loaded into the four edge FPGAs. It is worth mentioning that, all four boards are exactly the same, only the firmware and connections assigns them given functions in the system. Figure 45 presents a photo of the constructed test setup.

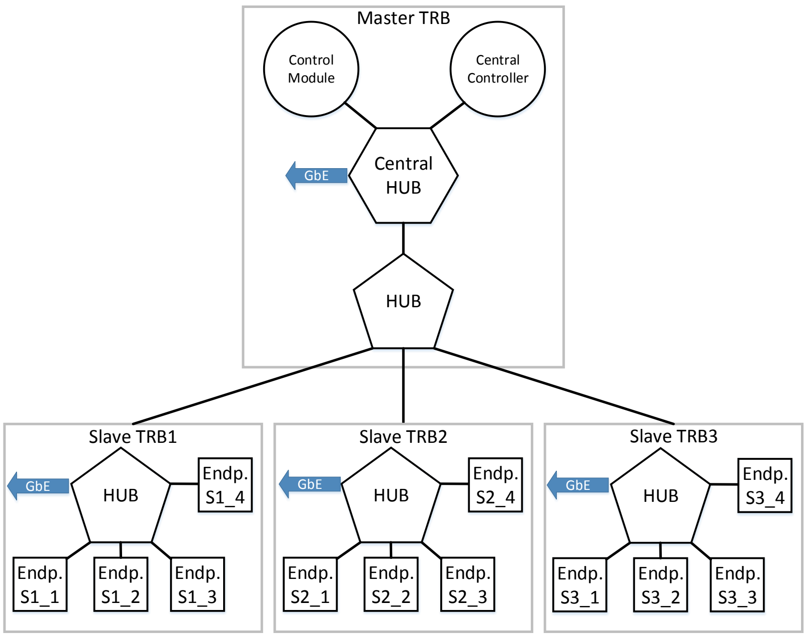


Figure 43: Logical view of the TRB platform test setup. One TRB3 board was used as a Master board, while three others act as Slaves. The Master board contains firmware modules that provide features of Control Module, Central Controller and hub. Each Slave board has a hub, to which 4 TDC endpoints are connected.

Such setup presents all the main features of the full-scale system. The Master TRB acts as a central hub, Central Controller module and Control Module. The central hub maintains the TrbNet communication with all the connected endpoints (that is local devices as well as Slave boards) and provides data transmission path via GbE. The Central Controller module provides a readout control mechanism. The operator can configure how the readout requests are generated. They can be triggered by some external event, as well as generated internally. The Control Module is a gateway for monitoring and control channel between the operators' computer and all the components in the test setup. The Slave TRBs are equipped with the hub instance, which acts as a data concentrator of the four TDC endpoints that are the data sources in the system. Data collected from those endpoints exits the system through GbE gateway provided by the local hub instance.

6.1.1 Measurements Methodology

In order to perform performance measurements of the system, Central Controller has been configured to send readout requests periodically with 50 kHz frequency. The TDC firmware features a calibration functionality, which generates internal signals at the inputs of the TDC. Those signals are then digitized by the module. Together with the configurable buffer size (number of recorded, measured signals per channel) it gives the operator a way to control the size of the outgoing data stream. In order to enable the calibration, special type of readout request has to be sent from the Central Controller, hence it has to be configured accordingly.

A series of tests has been performed in order to visualize the performance of the system. Each test consisted on scanning the accepted readout request rate as a function of the number of channels delivering data. The difference between individual tests was the number of Slave TRBs included in the setup as well as the number of enabled endpoint. At first, only the Master and one Slave were measured, then the remaining Slaves were enabled. Also various configurations of enabled/disabled endpoints was evaluated. All measurements configurations are presented in Figure 44.

Test No	S1_1	S1_2	S1_3	S1_4		S2_1	S2_2	S2_3	S2_4		S3_1	S3_2	S3_3	S3_4
1	X													
2	X					X								
3	X					X					X			
4	X	X												
5	X	X				X	X							
6	X	X				X	X				X	X		
7	X	X	X											
8	X	X	X			X	X	X						
9	X	X	X			X	X	X			X	X	X	
10	X	X	X	X										
11	X	X	X	X		X	X	X	X					
12	X	X	X	X		X	X	X	X		X	X	X	X

Figure 44: Table visualizing performed measurement configurations. First column states the test number, while the rows show enabled endpoints (marked with X). The following columns represent Slave boards (S1, S2 and S3) with their corresponding endpoints.

Each test in a series consisted on configuring the setup (enabling/disabling endpoints and hubs) and then starting the readout at 50 kHz frequency with all channels on the endpoints disabled. That means that at the initial moment, “empty” subevents were constructed, consisting of headers and control data only. With the step of 5 seconds, successive channels were being enabled, on all active endpoints at the same moment.

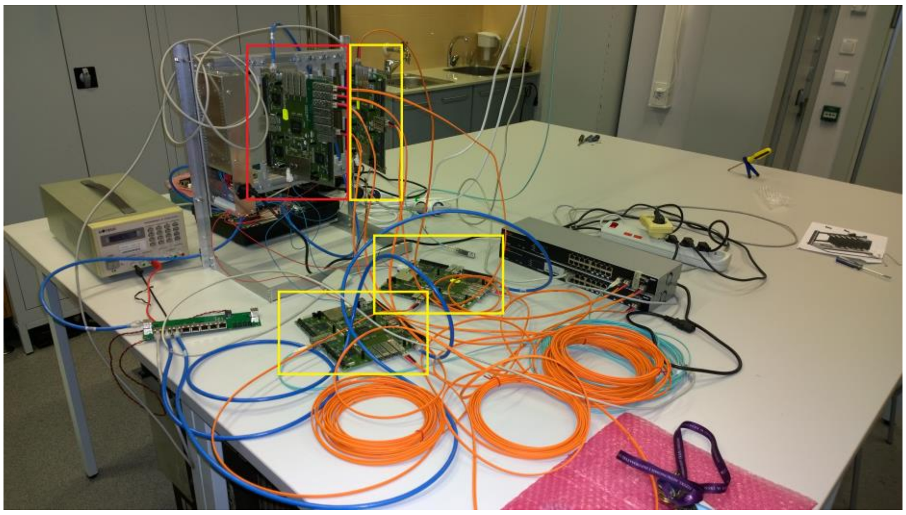


Figure 45: Photo of the test setup. Four TRB boards were used, from which one (red) as a Master and three others (yellow) as slaves. The boards are interconnected with optical fibres for both the TrbNet connections and for the Gigabit Ethernet. One can notice a SFP Mezzanine Add-on on the Master board, which concentrates the fibres from the Slaves to the central hub.

6.1.2 Estimations of the System Performance

One can estimate the amount of data being generated and therefore the time needed for the processing, resulting in the dead time of the system and the corresponding accepted readout rates. As there are TDC-based endpoints only, according to its data format description (29) a single time measurement (called *a hit*) on a channel is represented by 4B word. The hits are complemented with one 4B long word representing so called epoch counter, one such word exists per active channel. The readout buffers were configured to contain up to 32 hits per channel (the TDC firmware features 49 channels) between two consecutive readout requests. In calibration mode, the buffers are filled to their limits with generated hits. Additionally each TDC inserts at the beginning a 4B TDC header word and debug word at the end, also 4B long. The data coming from the endpoints (for detailed data format description see Section 5.1.4.1) is encapsulated into sub-subevent entities (additional 4B) and then transformed into subevents (additional 16B header and 8B trailer) and then into HTUQ unit (additional 8B of header and 32B of trailer). As each TRB in the system has its own network gateway for the data transmission, processes the data by building HTUQ units from local endpoints only.

The hub component, access the data using internal TrbNet interface, which has 16b wide data port and the logic is clocked with 100 MHz system clock. Data from the hub is then forwarded to the Gigabit Ethernet Module, which runs at 125 MHz frequency, processing 8b wide words through GMII interface (15). This point is at the moment the bottleneck of the system, because the data from the hub can be delivered much faster (200 MBps) than it can be transmitted out from the system (125 MBps). The Gigabit Ethernet Module converts the HTUQ unit into an UDP packet, fragmented into Ethernet frames with MTU of 1400B, therefore all underlying protocols layers have to be applied. This includes Ethernet, IP and UDP, which sums up to additional 46 B per frame and 8 B per whole packet. Figure 46 presents the calculations concerning amount of generated data and achievable readout rate for a single board with one TDC endpoint enabled and with all channels active.

(1)
$$\begin{aligned} DataFromHUB &= QueueHeaders \\ &+ \Bigg(SubeventHeaders \\ &+ \Bigg(ActiveEndpoints \\ &* \Big(SubSubHeader + EndpointsHeader \\ &+ \big(ActiveChannels \\ &* \ (EpochCtr + BufferSize * HitWord)\big)\Big)\Bigg)\Bigg) \end{aligned}$$

(2)
$$DataFromHUB = 40 + 24 + \Big(1 * \Big(4 + 8 + \big(49 * (4 + 32 * 4)\big)\Big)\Big) = 6\,544\ B$$

(3)
$$DataForGbE = \left\lceil \frac{6\,544}{1\,400} \right\rceil * 46 + 8 + DataFromHUB = 6\,782\ B$$

(4)
$$6\,782\ B * 8\ ns = 54{,}3\ \mu s$$

(5)
$$\frac{1\ s}{54{,}3\ us} = 18\,416$$

Figure 46: Equation (1) represents the amount of data generated by a single hub per readout request as a function of the number of connected endpoints, active channels, various headers and buffer sizes. Calculated size (2) of an HTUQ for the situation when only one of four endpoints is enabled and it has all channels active. Dividing that amount of data by MTU of an Ethernet frame, results in 5 Ethernet frames, that have to be constructed by adding several layers of encapsulation, summing up to a total of 6788B (3). The time needed for processing an entire portion of data generated by a hub as a response for one readout request (4). Assuming the GbE readout rate, theoretically, possible to achieve readout rate would be 18,42 kHz (5).

6.1.3 Measurement Results

Measurements were performed by configuring the setup according to the table presented in Figure 44 and scanning the possible data sizes from the endpoints by increasing the amount of active channels. The configuration as well as the readout of performance parameters was possible through the registers on FPGA devices, accessible via the Slow Control over GbE channel. For each number of active channels five snapshots of registers were recorded with one second break in between. The parameters taken into analysis were: accepted readout rate from the Central Controller and amount of the data transferred through the Gigabit Ethernet Gateway on each of the TRBs in the system.

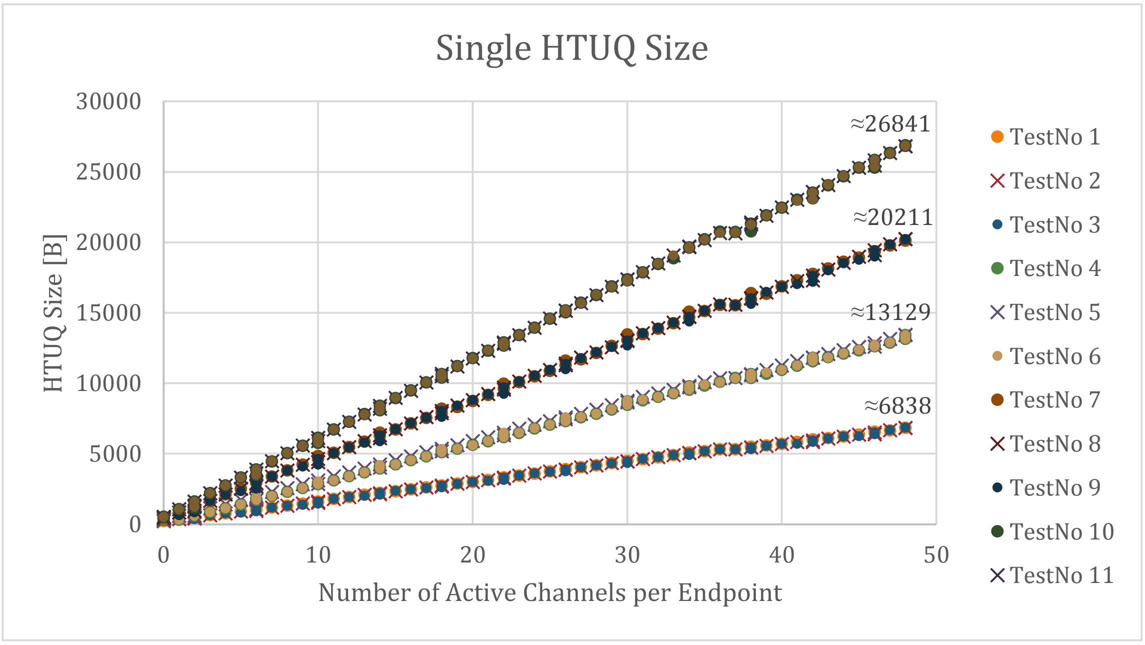


Figure 47: Chart presents the size of generated HTUQ entities as a function of active channels per enabled endpoint and number of boards in the system

Figure 47 presents the average size of the generated HTUQ units in each of the performed tests. One can distinguish four separated groups, each one corresponds to the amount of enabled endpoints in a particular test. For example the first line with maximum HTUQ size (value 6838, which stays within error margin comparing to the value calculated in Figure 46) corresponds to tests number 1, 2, 3. One can see that introduction of additional Slave boards does not influence the size of constructed units by individual Slaves. Enabling a second endpoint (tests number 4, 5, 6) results in a unit two times larger, as both endpoints provide the same amount of data and so on. It is important to notice that the endpoints are being readout by the hub in a sequential manner, therefore the time needed for processing is proportionally longer. It can be seen in the next Figure 48, which shows the readout rate dependence of the number of active channels per enabled endpoint.

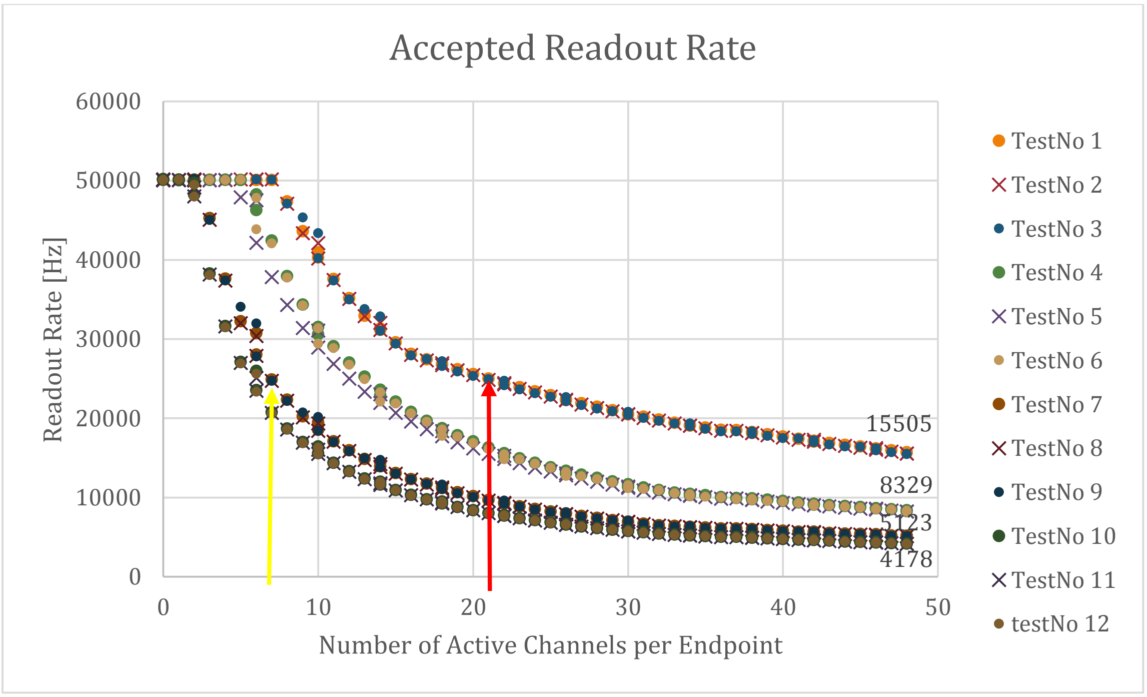


Figure 48: Chart presents the achieved readout rate as a function of active channels per enabled endpoints and number of boards in the system

The more input channels are enabled, the more time it takes to digitize the hits and construct network packets containing received data. For one Slave board with one endpoint enabled and all channels active (test 1) it is at level of 15,505 Hz. The same rate is achieved even though additional Slave boards are introduced (tests 2 and 3) as they work in parallel, delivering the same amount of data. The achieved readout rate is lower than the one calculated in Figure 46. The reason for that is that the calculations did not include the busy-release message generation and delivery time to the Central Controller module, which also contributes into the dead-time of the system as well as constant latencies introduced by the endpoints multiplexing and HTUQ construction process.

One can also distinguish four groups of test results. Each group corresponds to the amount of endpoints enabled on a single TRB board. As four endpoints are connected to one hub entity, which has one Gigabit Ethernet Module gateway, the data from four endpoints has to be sequentially multiplexed into one output interface (presented by tests 10, 11, 12). That is why increasing the number of endpoints connected to one gateway, increases the dead time of the system.

The initial readout rate (configured to 50 kHz) is maintained as long as the readout module manages to work in a streamlined way by transmitting the previously generated data, stored in buffers while constructing a new units. In case the transmitter does not clear the buffers in time, the whole system has to wait until it is ready to process new portion of data. This dead time causes the system to hold the readout requests, which results in lowering the readout rate. The internal buffers are designed to support the data stream for constant work of Gigabit Ethernet Module. Therefore, when the amount of data saturates the link (reaching about 100 MBps, see Figure 49), the system has to hold the readout requests until the buffers get cleared. This reflects on reducing the overall readout rate.

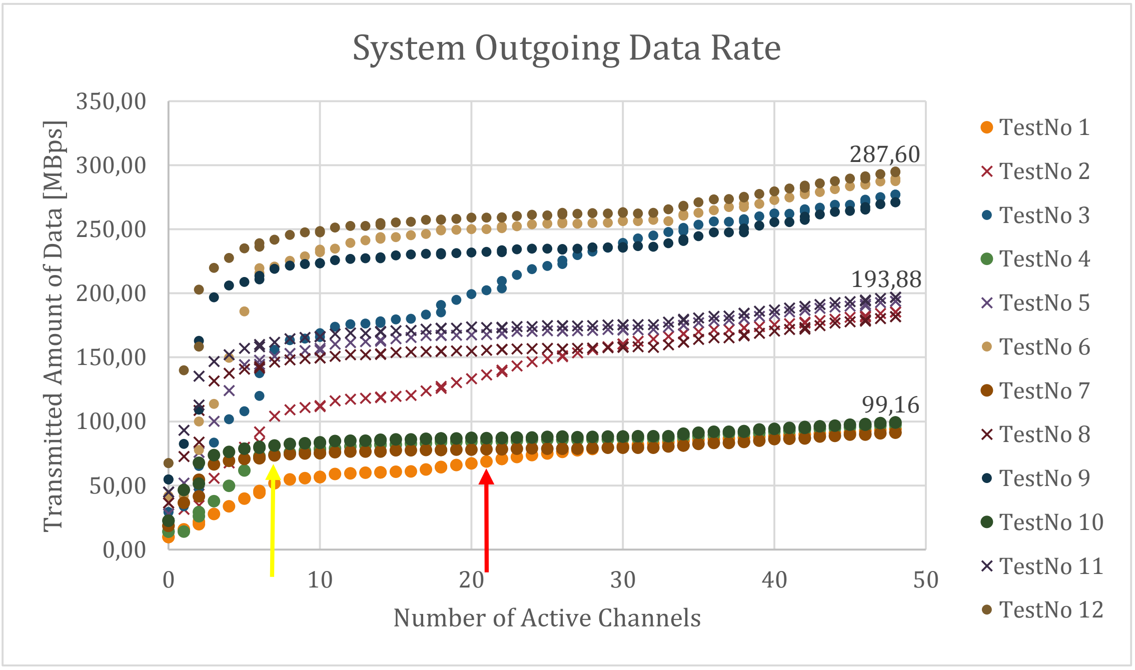


Figure 49: Chart presents the outgoing amount of data, aggregated from all the boards in the system as a function of active channels per enabled endpoints and number of boards in the system.

Figure 49 presents the total amount of data generated and transmitted by the system per second. On this chart, in contrary to the previous ones, one can distinguish only three groups of test results. Those groups represent the number of enabled Slave boards in the system. For example the lowest group (tests 1, 4, 7, 10) corresponds to the situation when 1, 2, 3 and 4 endpoints send data to the same hub instance. One can notice that the saturation level is reached faster in case four endpoints are enabled per hub.

An example illustrating the flexibility of the system configurations can be derived from Figure 48 and Figure 49. Marked by two lines, are the two configurations: yellow line points to the test number 7, during which only one Slave was enabled with 3 endpoints active, each having 7 channels providing data. The red line indicates the test number 1 that is one Slave with only one endpoint active but three times as much enabled channels. During those two measurements, the overall data volume outgoing from the system was comparable (Figure 49) (slightly larger in case of test 7, because of the additional endpoints headers) and according to Figure 48, the accepted readout rate was also at the same level in both cases. This indicates, that as long as outgoing links on the individual components are not saturated, the system can be constructed in various ways.

6.1.4 Conclusions

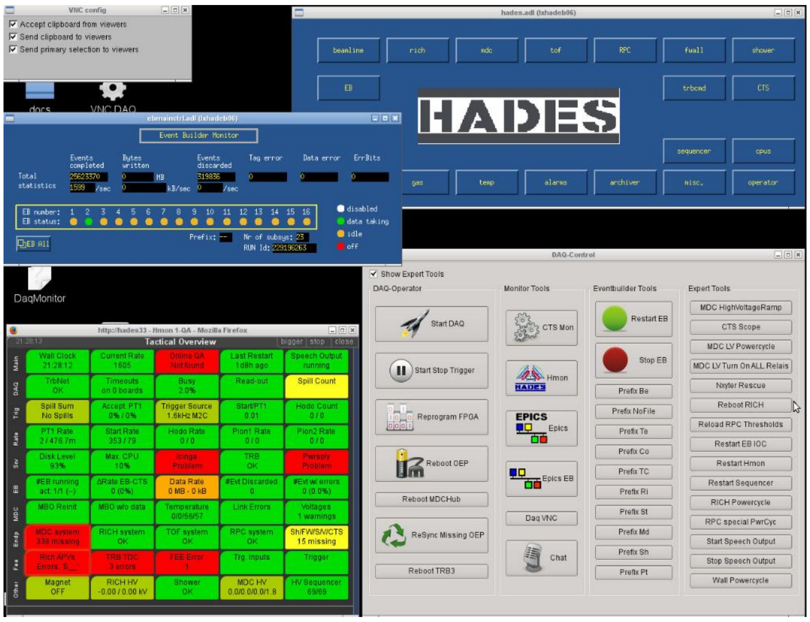
The measurements performed on a laboratory test setup provide valuable insight at the possibilities of the system architecture. By testing various system configurations, it can be clearly stated that the accepted readout rate of the system depends only on the processing time of its endpoints and not the number of enabled components. The Figure 48 presents that the readout rate is constant while the number of enabled Slave boards increases, as long as they process the same amount of data. Therefore another statement can be derived: the readout rate of the system depends only on the processing time of its slowest (or most occupied) endpoint. The Figure 49 gives a solution about how to increase the readout rate of the system when needed. By introduction of additional hub components, equipped with GbE gateways and properly segmenting the system, assuring that the processing time of generated data by each endpoints stays within the transmission time, one can create optimal solutions applicable for high-rate and large-scale measurement systems.

6.2 HADES DAQ System Measurements

The HADES data acquisition system (25) is the largest setup built entirely on TRB platform. It consists of total 80,000 analog channels which are processed by 450 endpoints (not including front-end modules) at readout rates up to 100 kHz and aggregated into 30 data streams by the hub components with Gigabit Ethernet Modules. The schematic view of the detector system and data acquisition system were presented on Figure 31 and Figure 32 in Section 5.1 about system applications.

The HADES detector system consists of 7 individual subsystems. Each subsystem features a specific detector type and its readout mechanisms, up to a stage when a DAQ system endpoint collects or digitizes the data from the front-end modules. At this point the data enters the TRB system architecture composed of various types of endpoints and hubs. It is important to notice that this system is a combination legacy modules like TRB2 and recent developments. For example Shower detector readout has hub functionality under a form of firmware module instantiated in the FPGA device on a digitizer board. Some of other subsystems require a dedicated hardware HUB Add-on board in order to concentrate data and provide the GbE gateway. This imposes several constraints on possible readout schemes and segmentation.

The system is controlled by the user friendly control panel (Figure 50) which provides monitoring of most important parameters of the system. Tactical Overview panel is located at lower left. It presents at a glance all the vital parameters of the system like readout rate, data throughput and indicates the health of the subsystems as well as the dead time they introduce. Located on the right side is the DAQ-Control panel, which gives the operator an interface to most common operations like starting and restarting the system or reconfiguring its components. It is also a place from which Central Trigger System can operated. On top is located an EPICS-based interface for operating the detector subsystems. On the top left side the Event Builder Monitor is presented, showing the status of the machines collecting data from the DAQ network.



Events completed | Bytes written | Events discarded | Tag error | Data error | ErrBits

Total statistics 25623370 | 0 MB | 319836 | 0 | 0 | 0

1599 /sec | 0 kB/sec | 0 /sec

EB number: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Prefix: -- Nr of subsys: 23 RUN Id: 229196263

21:28:13

Main	Wall Clock 21:28:12	Current Rate 1605	Online QA Not found	Last Restart 1d8h ago	Speech Output running
DAQ	TrbNet OK	Timeouts on 0 boards	Busy 2.0%	Read-out	Spill Count
Trig	Spill Sum No Spills	Accept PT1 0% / 0%	Trigger Source 1.6kHz M2C	Start/PT1 0.01	Hodo Count 0 / 0
Rate	PT1 Rate 2 / 476.7m	Start Rate 353 / 79	Hodo Rate 0 / 0	Pion1 Rate 0 / 0	Pion2 Rate 0 / 0
Srv	Disk Level 93%	Max CPU 10%	Icinga Problem	TRB OK	Pwrsply Problem
EB	#EB running act 1/1 (--)	ΔRate EB-CTS 0 (0%)	Data Rate 0 MB - 0 kB	#Evt Discarded 0	#Evt w/ errors 0 (0.0%)
MDC	MBO Reinit	MBO w/o data	Temperature 0/0/56/57	Link Errors	Voltages 1 warnings
Endp	MDC system 338 missing	RICH system OK	TOF system OK	RPC system OK	Sh/FW/S/V/CTS 15 missing
Fee	Rich APVs Errors: S__	TRB TDC 3 errors	FEE Error 1	Trg. Inputs	Trigger
Other	Magnet OFF	RICH HV -0.00 / 0.00 kV	Shower OK	MDC HV 0.0/0.0/0.0/1.8	HV Sequencer 69/69

Figure 50: A screenshot of the HADES DAQ control panel. Several control panels provide access top most commonly used operator functions as well as overview of the subsystems status and operation parameters.

System	Channels	kB/Readout	Estimated Occupancy	Estimated kB/Readout	Estimated Data Volume [MBps]		Number of Endpoints	Number of Hubs	Available Output Bandwidth [MBps]*		Estimated Bandwidth Usage
MDC	25964	105.882	17 %	18	360		372	12	600		60 %
RICH	28272	114.666	1.5 %	1.7	34		30	3	150		23 %
TOF	768	6	15 %	0.6	12		7	1	50		24 %
RPC	2232	29.142	7 %	2.2	44		24	2	100		44 %
Shower	18432	73.728	2.5 %	2.0	40		12	6	300		13 %

Figure 51: Table presents various subsystems, their scale in a form of channels number, theoretical maximum data rates as well as the ones after noise suppression for Au+Au experiment (25) for the 20 kHz readout rate. On the right side, two columns indicate the amount of hubs and endpoints used for data transmission in the particular readout chains. The last column shows the percentage of the available bandwidth, used during beam time.

6.2.1 In-Beam System Performance

In spite of the complexity of the HADES DAQ system, the possible data rates generated by the subsystems are possible to estimate, taking into account the amount of channels, digitizing procedures and data units construction processes (45).

Five major subsystems are enclosed in Figure 51. Estimated occupancy and data volumes have been derived from simulations for a specific experiment with Au+Au beam type (25), which than has been conducted in year 2011 (9). This experiment represents the most challenging case for the system because of the largest data throughput.The operation of the entire detector system, therefore the generated data rates, strongly depend on the beam type and its structure. This particular beam type produced an average of 20 kHz readout request rate, but it should to be noted that the beam structure was not uniform over time. As presented in Figure 52, the beam comes in a form of spills lasting around 0.5 ms. During such a spill, the intensity drastically fluctuates, resulting in proportional readout requests rates. Hence the system architecture must be able to process much higher rates during short periods of time, than the expected, average 20 kHz rate.

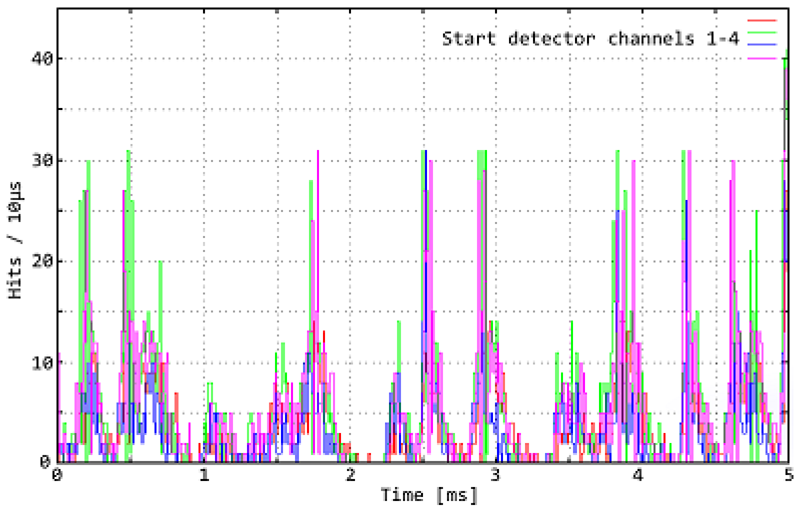


Figure 52: Beam spill structure, presented as a hit rate on Start detector planes. The single beam spills come at an average frequency of 20 kHz. (25)

During an experiment just a small subset of channels provided important data as only parts of the detectors are irradiated and the signals must surpass applied thresholds, depending on the subsystem readout scheme. The occupancy in the analysed experiment varied from 17 % in case of the MDC detector, down to 1.5 % for the RICH detector. Taking into account the amount of channels and the occupancy, one can conclude the data rates generated by each subsystem during experiment runtime (shown in the table in fifth column).

It should be noted (*) that the available output bandwidth presented in Figure 51 is only 50 MBps per hub instead of full Gigabit Ethernet bandwidth. This is caused by an early (year 2011) implementation of the Gigabit Ethernet Module gateway, in which the internal buffers have to be completely cleared in order to accept the next readout request. As it was shown in Figure 49, the recent developments allow to transmit up to 100 MBps.

Even though, the hubs presented at that time reduced available bandwidth smaller than the full speed Gigabit Ethernet, the architecture of the system allowed for the efficient data taking. It was achieved by the proper segmentation assured the balance between available bandwidth served by hubs and the number of connected endpoints. The extra safety factor has also been taken into account in order to properly handle the presented in-beam spill fluctuations. Taking for example the MDC subsystem, which produces the largest data volume, it uses 60% of the available bandwidth. The other subsystems utilize an average of 25% of available bandwidth. According to the laboratory measurements, described in previous Section 6.1, the performance of the system is limited by the data output capabilities of the hubs. In case the output links get saturated, the accepted readout requests rate is reduced, which in case of a real experiment, results in data loss. The example under discussion, of Au+Au beam type, shows that the constructed DAQ system uses less than 50% of available bandwidth, therefore the outgoing data construction and transport processes are not the limiting factors for the overall system readout performance.

6.2.2 Performance Reducing Factors

As it was stated in the previous section, in case of DAQ system in HADES, the data transport is not the limiting factor. In order to investigate the factors, that introduce substantial dead time, an example of the readout chain of the Shower detector subsystem (38), (39), previously mentioned in Sections 5.1.1 and 5.1.5 will be analyzed.

This readout chain is especially interesting because it has fixed readout time of the front-end modules. Each of the FEE modules multiplexes 32 input signals to one ADC channel, as described in Section 5.1.5. The process of switching the multiplexer input takes 300 ns, multiplied by the number of inputs, gives 9.6 us in total for reading out a Front-End Board. Outputs from the front-ends are digitized in parallel by the ADC modules on the Shower Add-on boards, therefore it is the total time of collecting data from the front-end stage into the endpoints for the entire subsystem. The digital data is passed through zero-suppression algorithm in order to filter out the noise and reduce the outgoing data stream. Such situation corresponds to the normal data taking mode. However, configuring the threshold levels by setting the highest value “empty” events can be generated, containing headers only. In contrary, setting the thresholds to 0 allows all data to pass through, and generate maximum-sized data units.

Two FPGA devices on Shower Add-on board are endpoints, collecting data from the ADCs, while the third one acts as a hub and a Gigabit Ethernet gateway. In correspondence to the laboratory measurements (previous Section 6.1), this configuration is reflected by tests number 4, 5, and 6 (two endpoints enabled on each Slave board).

During those 9.6 us, the Shower subsystem holds the busy-release message, preventing the Central Controller (in HADES it is the Central Trigger System module, see Section 5.1.3) from generating a new readout request. This derives directly the dead-time of the entire system and therefore maximum accepted readout rate. However, in order to calculate the entire time needed by Shower subsystem since the reception of a readout request until a busy-release message is delivered back to the CTS, one has to take into account two additional factors: time needed for generating and transmitting the busy-release message and the time needed to construct HTUQ units out of digital data coming from endpoints. The latency caused by the transmission of busy-release packet was measured in (25) and takes 3 us. In case of high thresholds (rejected data from ADC) it is possible to estimate the size of outgoing data units which would only consist of headers based on the Equations (1) and (3) from the Figure 46. The changes are in the number of endpoints (which are 2 in this case) and in the fact that they do not provide any data except headers.

$$DataFromHUB = 40 + 24 + (2 * 4) = 72\,B \tag{6}$$

$$DataForGbE = \left\lceil \frac{72}{1\,400} \right\rceil * 46 + 8 + DataFromHUB = 126\,B \tag{7}$$

$$126 * 8\,ns = 1\,008\,ns \tag{8}$$

$$9{,}6\,us + 3\,us + 1\,008\,ns = 13{,}6\,us \tag{9}$$

$$\frac{1s}{13{,}6us} = 73\,529 \tag{10}$$

Figure 53: Calculations of the dead-time introduced by the Shower subsystem. Equation (6) gives the size of HTUQ in case when all the data from ADCs is filtered. Such an "empty" event is then encapsulated into a network packet (Equation (7)) and the time needed for construction is presented on Equation (8). All three factors: front-end readout, packet construction and busy-release message transmission are summarized on Equation (9), hence achievable readout rate is concluded (10).

Calculated maximum readout rate, for a system that includes only the Shower subsystem for the "empty" event is at the level of 73 kHz. Similar calculations are shown in Figure 54, for a case in which the filtering was disabled, allowing the construction of maximum-sized packets. In that case, the highest achievable rate settles at 8.7 kHz.

$$(11)\quad DataFromHUB = 40 + 24 + (2 * (8 + 48 * 128)) = 12\,368\ B$$

$$(12)\quad DataForGbE = \left\lceil \frac{12368}{1400} \right\rceil * 46 + 8 + DataFromHUB = 12\,790\ B$$

$$(13)\quad 12790 * 8\ ns = 102{,}2\ us$$

$$(14)\quad 9{,}6\ us + 3\ us + 102{,}2\ us = 114{,}8\ us$$

$$(15)\quad \frac{1s}{114{,}8us} = 8\,710$$

Figure 54: Calculations of the dead-time introduced by the Shower subsystem in case all the data is accepted and forwarded into HTUQ construction module.

The size of a single, maximum data unit generated by Shower Add-on is at the level of 12.5 kB. This can be compared to the laboratory measurements and the case when 47 channels were enabled on two activated endpoints per hub (point 47 on X axis of the Figures Figure 47Figure 48 and Figure 49). According to the Figure 48, achievable readout rate for such configuration is around 8.6 kHz, which validates the calculated value (Equation 15) in Figure 54, within the measurement error margin.

The calculated values can be now compared to a series of tests performed with HADES DAQ system, during a beam-off, idle state. During such state, the detectors are switched off as well as parts of front-end electronics. Figure 55 presents the results of the peak readout rate measurements, depending on the configuration of the subsystems included in the readout system.

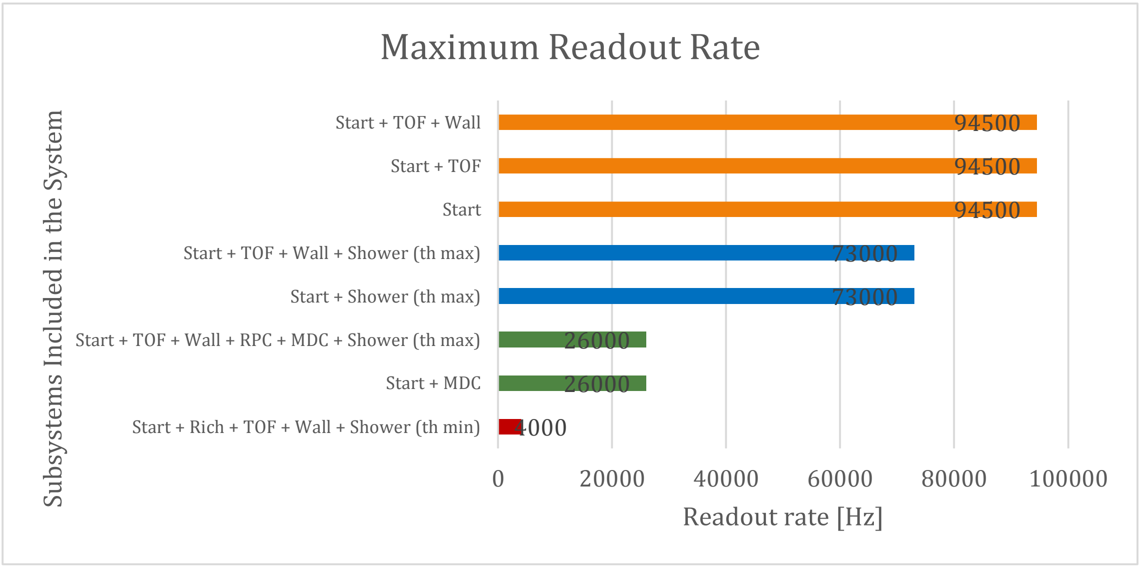


Figure 55: Chart presents maximum achieved readout rates as a function of the type and the number of subsystems included in the readout.

The Start detector features very fast readout scheme of only few channels, hence it has been chosen as a reference for the other subsystems. Similar capabilities are presented by subsystems TOF and Wall (orange bars), which introductions does not influence the maximum accepted readout rates, settled at the level of 94.5 kHz. All those subsystems contain TRB boards as the endpoints and are connected on the same level in the tree system architecture. It means that number of hubs, which introduce latency, on the route to the CTS is equal, hence the dead-times introduced by those subsystems are comparable. In these cases the endpoints are in an idle state and readout rates depend only on the amount of generated data.

The first reduction of the readout rate in the whole system is caused by the Shower subsystem (blue bars). According to the previous description, this subsystem needs a constant amount of time for reading out the front-end modules. As calculated in Figure 53, in case of maximum thresholds settings, the subsystem introduces 13.6 us dead-time, which results in a possible, maximum readout rate of 73.5 kHz, value comparable to the measured one.

Introduction of the MDC subsystem, reduces the achievable accepted readout requests rate to the level of 26 kHz. Such long dead time is cause mainly by the front-end modules readout (25), which use different type of optical transceivers (as described in Section 5.1.1) that offers only 250 Mbps throughput. The procedure of reading out the entire detector sectors to the endpoints is therefore highly time consuming and takes in a worst case up to 38.4 us and corresponds to the accepted readout rate 26 kHz, what is presented by the measurement on Figure 55 by green bars.

Critical case is represented by the red bar in Figure 55. Shower subsystem with thresholds turned off, processes data from all the input channels, which results in generating maximum-sized data units, which require to be transmitted by the hub through the Gigabit Ethernet gateway. According to calculations in Figure 54, under such conditions, the electronics deliver 12.8 kB of data per single readout request, which, at the theoretical rate of 8.7 kHz, results in 111.3 MBps and exceeds the 50 MBps capabilities of the legacy GbE Module implementation on the Shower Add-on at the time of experiment (see Figure 51 in the previous section). In this situation, the data transport performance, reduces the accepted readout rate by half, to the level of 4 kHz, as confirmed by the measurement. It should be noted, that this is an artificially generated situation as the occupancy of the subsystem during normal operation is only 2.5 % and for such case data transport is not the limiting factor.

6.2.3 Conclusions

In-beam experiments conducted with HADES detector system have validated the concept of the TRB based, large scale data acquisition system. Properly designed architecture, with carefully balanced load from the endpoints on the hub components allows to assure that the data transport is not an issue for the system performance, which is reflected mostly by the achieved readout rate. Figure 51 illustrates with numbers, the typical load on main HADES subsystems as well as their processing capabilities. The constructed architecture has a safety margin of over 50% in order to handle fluctuating data volumes, caused by beam characteristics. Therefore a statement can be derived that given the scalable architecture of the TRB platform, it is applicable for large scale measurement systems.

HADES DAQ measurement in an idle state, provides an insight at what are the limiting factors in the described system. Various configurations of subsystems have been evaluated in order to retrieve the achievable maximum accepted readout requests rate. One significant observation, which also confirms the laboratory tests conducted in the previous section, should be noted: the system performance does not dependent on the amount of elements included in the readout process but on the dead time related to its slowest element.

7 Conclusions and Outlook

Measurement systems, used in various real time applications, require system architectures for the effective data acquisition and processing. Described in this thesis architecture based on universal TRB modules targets the most challenging application – the readout of detectors in physics experiments. Those measurement systems vary in scale: from hundreds up to hundreds of thousands single sensors, which require parallel and synchronized processing in the real-time regime. Therefore the main goal was to create a scalable architecture which allows to cover the large scale of applications without loses of the performance in terms of data processing. The other goal was to develop effective data protocols which allow for the effective data transmission, system control and monitoring processes, as well as distribution of time-critical readout control messages. For this purpose original TrbNet protocol was developed. On the other hand, the system should be easily interfaces with commercially available network infrastructures. This second demand was achieved by application of the Gigabit Ethernet standard.

Proposed tree-structured architecture of the platform (described in detail in Chapter 4) allows to construct data acquisition systems consisting of a high number of components. Based on two logical elements: endpoints and hubs, the composed platform can be extended by the introduction of hub modules with opens several slots for additional endpoints. The endpoints are the elements of the system which digitize, process and deliver data to the system. Both of those logical functions are realized by the same base module TRB board (4), (29), thanks to application of FPGA devices. Therefore the entire system can be constructed out of the identical units what ease the system configuration and maintenance. The individual function of a particular module is assigned by its firmware configuration, implemented features and applied extension modules.

The scalability of the platform has been extensively tested and proven by the realization of two setups. The first one is the HADES data acquisition system (24), (9), (45), for which the TRB platform was designed, the second one is the J-PET tomograph (10), (7). In case of HADES, the system consists of 450 endpoints and 30 hubs, which can be treated as an application of large scale. The system designed for J-PET uses the same base modules, configured as 40 endpoints and 10 hubs. The detailed overview of both architectures was presented in Chapter 5. The systems have been evaluated in terms for performance and the results of measurements are enclosed in Chapter 6, providing an insight at the capabilities of the system.

Several statements are concluded from the performed evaluation tests. By measuring the readout rate of the system in various configurations of the hubs and endpoints, it has been proven that the achieved rate is dependent only on the processing time of its slowest element (i.e. data production process) and not the total number of elements included in the system. Once the data is produced and stored in the readout buffers, system effectively transport them to the final destination without additional losses. It is an important point, which has been verified by the laboratory setup measurements as well as confirmed by the analysis of the performance of HADES detector system during operation. Another observation focuses on data transport capabilities of the platform. The bandwidth provided by a single hub module is limited to 100 MBps (or 50 MBps in legacy systems). In case the endpoints produce amount of data that saturates the link, the overall readout rate is reduced due to the time needed to clear internal buffers. However, the easy way out from this problem is to introduce additional hub modules and perform segmentation of the system, assuring the right balance between the data volume provided by the endpoints and the hub networking capabilities.

Both applications are also a showcase of the flexibility of the system. High-energy physics experiment and the medical imaging device share some common concepts, therefore, the TRB platform could be applied to both, thanks to its configurable nature and the system of Add-on boards, which extend its base functionality and adapt the module for a specific processing type.

7.1 Outlook

The presented TRB platform has found many applications in various systems. Significant part of the work is dedicated to adaptation of the platform for new users and the setups maintenance. The main systems constructed out of the presented solutions are for PANDA and CBM prototypes, which are future experiments at FAIR facility in Germany, NA61 and some applications at CERN in Switzerland, as well as already mentioned PET prototypes in Cracow and in Coimbra. Such wide collaboration provides valuable feedback about possible improvements and further development directions.

An interesting example, which is under development, is provided by the Anti-Proton Annihilation at Darmstadt (abbr. PANDA) (46), which will be a part of the FAIR. The physics that stand behind the experiment impose several requirements which have to be taken into account while designing even more challenging data acquisition system. One of challenge is very high interaction rate, set at level of up to 30 MHz. The detector is composed of several subsystems like trackers, electromagnetic calorimeters, drift chambers, DIRC and RICH subsystems. The single event size is simulated to be in the range of 4 to 20 kB, which combined with very high rate, gives as much as 200 GBps of raw data in the stage after the front-ends.

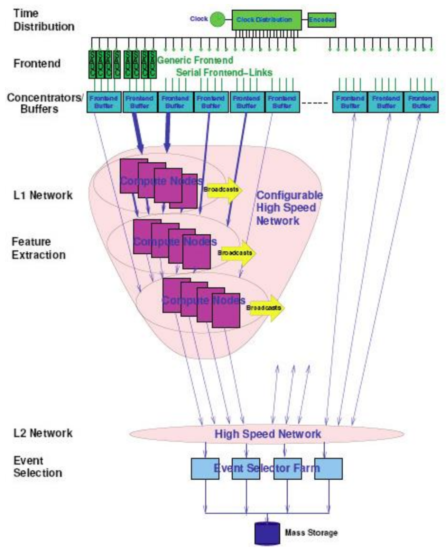


Figure 56: Schematic view of the PANDA DAQ system concept. TRB platform could be used as both front-ends and concentrators in several subsystems readout chains. The data from concentrators is forwarded to computing farm composed of Compute Node modules (47). The readout of the system work in trigger-less mode, with push architecture. That means that the front-ends generated a constant, synchronized stream of data that has to be analysed in the real-time.

Several concepts of the DAQ and Trigger systems architectures have been investigated and simulated (47). The result proposal describes a trigger-less system with one common time-stamp distribution, precise information used for synchronization of subsystems (Figure 56). The subsystems though, are independent DAQ platforms designed separately for each of the detector subsystems. As the proof of concept, TRB platform is used for several of those subsystems. Ongoing research shows that the next version of TRB board, equipped with recent FPGA device Xilinx Kintex7 and adapted to MicroTCA standard, would be the right way of development. The MicroTCA standard has been already chosen for the Compute Node modules (48), with which, the TRB boards would interact, hence a common standard providing connectivity and housing seems as a reasonable choice.

Upgrading the FPGA device, would also allow to introduce higher speed communication standards like 10 Gigabit Ethernet. The module already developed by the author, could be easily migrated to such a new platform, with only low layer components exchange. Increasing the output bandwidth of the hub gateway would allow to generate much more data by its internal endpoints, therefore the input channel density or the data volume generated by processing algorithms per module could be increased. Hence systems consisting of smaller number of boards, could be built.

Another interesting and ongoing development is conducted for J-PET project and involves the Central Controller board (described in Section 5.2.3). The introduction of Xilinx Zynq System-on-Chip devices allows to combine functions that, up to then were distributed over different system components. Integration of an ARM processor with the FPGA logic opens new ways of interacting with FPGA based electronics and creates new possibilities of implementing more complex algorithms for the online data processing mechanisms.

8 References

1. **The ATLAS Collaboration.** *Atlas detector and Physics Performance, Technical Design Report.* : CERN, 1999. ATLAS TDR 14, CERN/LHC 99-14.

2. **IEEE P1014 Standard Committee.** *VMEbus Specification Manual.* 1985.

3. **Esone Committee.** *CAMAC Updated Specifications.* : Commission of the European Communities, 1983. 9282542629, 9789282542620.

4. **Korcyl, G., et al.** A compact system for high precision time measurements (<14 ps RMS) and integrated acquisition for a large number of channels. *JINST.* 6, 2011, 10.1088/1748-0221/6/12/C12004.

5. **Agakchiev, G., et al.** The high-acceptance dielectron spectrometer HADES. *The European Physical Journal A.* 41.2, 2009, 243-277.

6. **William H., Ebeling C. i Borriello G.** Field Programmable Gate Array. *US Patent and Trademark Office.* 1993, US5208491 A.

7. **Korcyl, G., et al.** A system for acquisition of tomographic measurement data. *World Intellectual Property Organization.* 2015, WO/2015/028594.

8. **Neiser, A., et al.** TRB3: a 264 channel high precision TDC platform and its applications. *Journal of Instrumentation.* 8.12, 2013, C12043.

9. **Michel, J., et al.** The Upgraded HADES trigger and data acquisition system. *Journal of Instrumentation.* 6, 2011, Vol. 12, C12056.

10. **Korcyl, G., et al.** Trigger-less and reconfigurable data acquisition system for positron emission tomography. *Bio-Algorithms and Med-Systems.* 2014, Vol. 10, 1895-9091.

11. **Ashenden, P. J.** *The Designer's Guide to VHDL.* : Morgan Kaufmann, 2010. 0080568858, 9780080568850.

12. **Leo, W. R.** *Techniques for Nuclear nad Particle Physics Experiments: A How-to Approach.* : Springer Science & Business Media, 2012. 3642579205, 9783642579202.

13. **Corcoran, J. J.** *Electronics Instrument Handbook: Chapter 6 Analog-to-Digital Converters.* : The McGraw-Hill Companies, 2004. 9780070126183.

14. **West, B. G.** Time-to-digital converter. *US Patent and Trademark Office.* 2002, 6,501,706.

15. **Frazier, H.** The 802.3z Gigabit Ethernet Standard. *IEEE Network.* 3, 1998, Vol. 13, 10.1109/65.690946.

16. **Buttinger, W.** The ATLAS Level-1 Trigger System. *IOP Publishing.* 396, 2012, 012010.

17. **Xu, H., et al.** Introduction to PANDA Data Acquisition System. *Physics Procedia.* 37, 2012, 1855-1860.

18. **U.S. Department of Energy.** *Standard NIM Instrumentation System.* 1990.

19. **PICMG.** *AdvancedTCA Base Specification.* : PICMG, 2008. R3.0.

20. **Cooper, R. B.** *Introduction to Queueing Theory.* : Washington DC: CEEPress, 1981.

21. **Karnaugh, M.** The Map Method for Synthesis of Combinational Logic Circuits. *American Institute of Electrical Engineers.* 1953, 10.1109/TCE.1953.6371932.

22. **Moore, G. E.** *Cramming More Components onto Integrted Circuits.* : McGraw-Hill, 1965.

23. *A comparison of CPUs, GPUs, FPGAs and massively parallel preocessor arrays for random number generation.* **Thomas, D. B., Howes, L. and Luk, W.** ACM : International Symposium on Field Programmable Gate Arrays, 2009.

24. **Michel, J., et al.** The HADES DAQ System: Trigger and Readout Board Network. *IEEE Transactions on Nuclear Science.* 58, 2011, Vol. 4, 1745-1750.

25. **Michel, Jan.** *Development and Implementation of a New Trigger and Data Acquisition System for the HADES Detector.* Fankfurt am Main : Johann Wolfgang Giethe-Universitaet, 2012.

26. **Froelich, I., et al.** A General Purpose Trigger and Readout Board for HADES and FAIR-Experiments. *IEEE Transactions on Nuclear Science.* 55.1, 2008.

27. **Christiansen, J.** *HPTDC - High Performance Time to Digital Converter.* : CERN, 2004.

28. **Anghinolfi, F., et al.** NINO: an ultra-fast and low-power front-end amplifier/discriminator ASIC designed for the multigap resisitve plate chamber.

Nuclear Instruments and Methods in Physics Research Section A. 533.1, 2004, 183-187.

29. **Korcyl, G., et al.** *A Users Guide to the TRB3 and FPGA-TDC based Platforms.* , 2015.

30. **Ugur, C., et al.** A 16 channel high resolution (<11ps RMS) Time-to-Digital Converter in a Field Programmable Gate Array. *JINST.* 7, 2012, 10.1088/1748-0221/7/02/C02004.

31. **Ugur, C., et al.** 264 Channel TDC Platform Applying 65 Channel High Precision (7.2 ps RMS) FPGA Based TDCs. *IEEE Mediteranean Workshop on Time to Digital Converters.* 2013.

32. **Zimmermann, H.** OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications.* 28.4, 1980, 425-432.

33. **Comer, D. E.** *Internetworking with TCP/IP Vol. I: Principles, Protocols and Architecture.* : Prentice Hall, 1995. 978-0132169875.

34. **Plummer, D. C.** *An Ethernet Address Resolution Protocol or Converting Network Protocol Addresses to 48.bit Ethernet Address fro Transmission on Ethernet Hardware.* : RFC Standard, 1982. 826.

35. **Droms, R.** *Dynamic Host Configuration Protocol.* : RFC Standard, 1997. 2131.

36. **Postel, J.** *Internet Control Message Protocol.* : RFC Standard, 1981. 777.

37. **Zeitelhack, K., et al.** The HADES RICH detector. *Nuclear Instruments and Methods in Physics Research Section A.* 433.1, 1999, 201-206.

38. **Balanda, A., et al.** Development of a fast pad readout system for the HADES shower detector. *Nuclear Instruments and Methods in Physics Research Section A.* 2-3, 1998, Vol. 417, 0168-9002.

39. **Balanda, A., et al.** The HADES Pre-Shower detector. *Nuclear Instruments and Methods in Physics Research Section A.* 3, 2004, Vol. 531, 0168-9002.

40. **Yurevich, S., et al.** *The HADES Event Building System.* : GSI Scientific Report, 2010.

41. **Martins, P., et al.** Towards Very High Resolution RPC-PET for Small Animals. *Journal of Instrumentation.* 9.10, 2014, C10012.

42. **Blanco, A., et al.** TOFtracker: gaseous detector with bidimensional tracking and time-of-flight capabilities. *Journal of Instrumentation.* 11, 2012, Vol. 7, P11012.

43. **Svoboda, O., et al.** Electromagnetic calorimeter for HADES@FAIR experiment. *Journal of Instrumentation.* 9, 2014, 10.1088/1748-0221/9/05/C05002.

44. **Moskal, P., et al.** Strip-PET: a novel detector concept for the TOF-PET scanner. *Nuclear Medicine Review.* 15, 2012, C68-C69.

45. **Michel, J., et al.** In-beam experience with a highly granular DAQ and control network: TrbNet. *Journal of Instrumentation.* 2, 2013, Vol. 8, C02034.

46. **Johansson, T.** PANDA at FAIR. *Acta Physica Polonica B.* 4, 2012, Vol. 5, 5.1197.

47. **Korcyl, K., et al.** Modeling of the Architectural Studies for the PANDA DAT System. *IEEE Transactions on Nuclear Science.* 55.1, 2008, 429-434.

48. **Liu, M.** *A High-End Reconfigurable Computation Platform for Particle Physics Experiments.* PhD Thesis : Royal Institute of Technology, Sweden, 2008.

49. *Performance of the Krakow Frontend Electronics in Beam.* **Kardan, B.** : HADES Collaboration Meeting XXVII, 2014.

50. **Celiktas, C. and Ermis, E. E.** Timing Application to Improve the Energy Resolution of NaI(Tl) Scintillation Detectors. *International Journal of Instrumentation Science.* 1.5, 2012, 54-62.