Programa de Doctorado en Investigación Espacial y Astrobiología

Reconfigurable platform for concurrent multichannel pulse analysis in particle detectors and its application to a muon telescope

Tesis Doctoral presentada por

Juan Ignacio García Tejedor

Directores:
D. Juan José Blanco Ávalos
D. Sebastián Sánchez Prieto

Alcalá de Henares, 2021

Abstract

Given the diversity of observed phenomena in the field of experimental nuclear physics, a large variety of scientific instrumentation is constantly being developed. One of the key elements in these instruments is the data acquisition and processing system, which is normally an acquisition chain either custom built for the specific instrument and/or experiment, or built upon generic electronic modules such those using the NIM, CAMAC or VXI standards. Both these approaches have important drawbacks: custom data acquisition systems take time to develop and are not cost-effective, given their specific and inflexible nature; and standard instrumentation modules are flexible in their configuration, but limited in functionality, expensive, and often impractical for instruments where handling a large amount of signals is required.

This work presents a new concept of integrated data acquisition, analysis and processing architecture for use in nuclear electronics instrumentation that tackles on these limitations. A common hardware platform design is proposed that is focused on adaptability to a wide variety of experiment configurations, but capable of providing functionality and performance similar to a custom-built data acquisition system while also staying compact, low power and cost-effective. This is achieved by leveraging technologies such as multi-channel high-speed ADCs, reconfigurable logic FPGA devices, and modern, compact and SoC based Single-Board Computers, which allow the implementation of the acquisition chain through a combination of electronics, modular synthesizable hardware and Linux-based software.

Finally, a prototype of the proposed data acquisition system concept has been built, and its successful use in the Muon Impact Tracer and Observer instrument, a muon telescope composed of two $1\,\mathrm{m}^2$ scintillators and eight Photomultiplier Tubes, which is part of the ORCA cosmic ray observatory currently installed in the Juan Carlos I Antarctic Base, is described.

Resumen

Dada la diversidad de fenómenos que se observan en el campo de la física nuclear experimental, se desarrolla constantemente gran variedad de instrumentación científica. Uno de los elementos clave de estos instrumentos es el sistema de adquisición y procesamiento de datos, que normalmente es una cadena de adquisición construida a medida para el instrumento y/o experimento específico, o bien construida a partir de módulos electrónicos genéricos como los que utilizan los estándares NIM, CAMAC o VXI. Ambos enfoques presentan importantes inconvenientes: los sistemas de adquisición de datos a medida conllevan tiempo de desarrollo y no son económicamente rentables, dada su naturaleza específica e inflexible; y los módulos de instrumentación estándar son flexibles en su configuración, pero limitados en su funcionalidad, caros y a menudo poco prácticos para instrumentos en los que se requiere el manejo de una gran cantidad de señales.

Este trabajo presenta un nuevo concepto de arquitectura integrada de adquisición, análisis y procesamiento de datos para su uso en electrónica de instrumentación nuclear que aborda estas limitaciones. Se propone una plataforma hardware común que se centra en la adaptabilidad a una amplia variedad de configuraciones de experimentos, pero que es capaz de proporcionar una funcionalidad y un rendimiento similares a los de un sistema de adquisición de datos hecho a medida, a la vez que se mantiene compacto, de bajo consumo y con un coste reducido. Esto se consigue aprovechando tecnologías como los ADCs multicanal de alta velocidad, los dispositivos FPGA de lógica reconfigurable y los ordenadores compactos monoplaca basados en SoC, que permiten la implementación de la cadena de adquisición mediante una combinación de electrónica, hardware modular sintetizable y software basado en Linux.

Finalmente, se ha construido un prototipo del concepto de sistema de adquisición propuesto y se describe su uso con éxito en el instrumento Muon Impact Tracer and Observer, un telescopio de muones compuesto por dos centelleadores de $1\,\mathrm{m}^2$ y ocho tubos fotomultiplicadores que forma parte del observatorio de rayos cósmicos ORCA, actualmente instalado en la Base Antártica Española Juan Carlos I.

A mis padres.

Agradecimientos

"I'm sorry, but the princess is in another castle."

— *Braid*

Llegar a este momento es la culminación de un largo camino en el que han influido muchas personas, queriendo o sin querer, y por lo tanto hay mucho que agradecer. Algunas siguen estando cerca; otras se han quedado por el camino. Este trabajo lleva un poco de todas ellas.

Quiero empezar dando las gracias a Daniel Meziat Luna y a Sebastián Sánchez Prieto, a quienes considero mis padres espirituales en la Universidad de Alcalá; primero como alumno y como proyectando de fin de carrera, después como profesor e investigador, y en el caso de Sebastián, también como doctorando. Siempre he sentido su apoyo y amistad, y han sido para mi un ejemplo tanto en lo personal como en lo profesional. Seguramente no estaría donde estoy ni haciendo lo que hago de no ser por ellos.

En segundo lugar quiero expresar mi agradecimiento a Juan José Blanco Ávalos, no solo por ser mi codirector de tesis y quien ha hecho posible el proyecto ORCA, sin el cual este trabajo no hubiera podido existir; sino por su cercanía, su amistad, su paciencia y su calidad humana, de la que tanto me queda por aprender. Poco podía saber yo el día que le conocí, tomando un café en la cafetería del

Politécnico, que iba a jugar un papel tan importante en mi vida.

Quiero también dar las gracias al resto de compañeros de proyecto, Manuel Prieto Mateo y Raúl Gómez Herrero, pero sobre todo a mis otros compañeros de viaje antártico sin quienes no hubiera sido posible llevar el proyecto ORCA a buen puerto, Sindulfo Ayuso de Gregorio y, muy especialmente, Óscar García Población; compañero no solo de trabajo sino de múltiples aventuras, amigo y apoyo en los tiempos difíciles.

Además me gustaría dar las gracias al resto de mis compañeros del Grupo de Investigación Espacial de la Universidad de Alcalá, sobre todo a aquellos con los que he compartido el día a día estos años y me han dado su apoyo, consejo y ánimo constante en los momentos de frustración; Óscar Rodriguez Polo, Agustín Martinez Hellín, Antonio Da Silva Fariña y particularmente, Pablo Parra Espada; primero alumno, después compañero, siempre amigo.

Finalmente, esta tesis ha sido posible gracias a la financiación del Ministerio de Economía y Competitividad en el marco del proyecto CTM2016-77325-C2-1-P, y del Ministerio de Ciencia e Innovación en el del proyecto PID2019-107806GB-I00.

Pero no he terminado aún. Me falta expresar mi agradecimiento a las demás personas que, fuera del ámbito académico, han vivido tantos años mi viaje desde otra perspectiva, y sin cuyo apoyo y comprensión no sé si hubiera podido llegar hasta el final conservando al menos un poco de cordura.

Gracias a mis padres, por darme todas las oportunidades para llegar hasta aquí. Por su amor, por su paciencia y por ser mi ejemplo en la vida. Ellos son los responsables de lo mejor que hay en mi y a ellos les dedico este trabajo.

Gracias a María, mi compañera en los peores años de este periplo y principal damnificada de esta tesis, por su infinita paciencia y por tener casi siempre más fe en mí que yo mismo.

Y para terminar, gracias a todos los amigos que han estado ahí con su apoyo y sus ánimos a pesar de que les he cuidado menos de lo que debería en estos años tan complicados. Gracias Juanan, Txarly, Agus, María, Javi, Alicia, Jana y muchos otros a los que apenas veo si no es a través de un chat o una red social, pero que seguís estando cerca. Gracias en especial a Jorge y Sara por las charlas de madrugada en los momentos difíciles. Y gracias a todos los que me habéis transmitido vuestro cariño aun habiéndonos visto en la vida apenas nada. No puedo nombraros a todos, pero sabéis quienes sois.

Gracias a todos por traerme hasta aquí.

Parece que esta vez sí que hay princesa en el castillo.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> "In a hole in the ground there lived a hobbit."
>
> — J.R.R. Tolkien, *The hobbit, or There and Back Again*

The work presented in this book describes the research performed for the proposal, design and construction of an adaptable hardware and software platform for its use in nuclear instrumentation systems. The goal of this platform, aptly named Adaptable and Reconfigurable Acquisition Concept for Nuclear Electronics (ARACNE), is to provide a complete electronics front end system that is flexible enough to be used with different detector types in a variety of configurations and is capable of performing many signal processing functions common in nuclear physics research, without the need to substantially modify anything else than software and programmable logic blocks, also known as Intellectual Property Cores (IPCores). Also, the platform has been envisaged to be self-contained, meaning that it can operate autonomously without the need of another computer and can be connected to a standard network to publish or store data and to be remotely controlled and updated if necessary.

ARACNE was conceived when the construction of a new muon telescope consisting of eight scintillators and eight Photomultiplier Tubes (PMTs) in two layers, was being planned as part of a project of the Space Research Group (SRG) of the Universidad de Alcalá. The telescope would be capable of determining muon incident directions and thus study anisotropies, and the two detector layers would be placed below and above Castilla-La Mancha Neutron Monitor (CaLMa), another instrument developed and maintained by the SRG, with two purposes in mind; first, to use its lead shielding as a particle filter, and second, to combine the measurements of both neutrons and muons in a novel experiment. Figure 1.1

illustrates the concept.

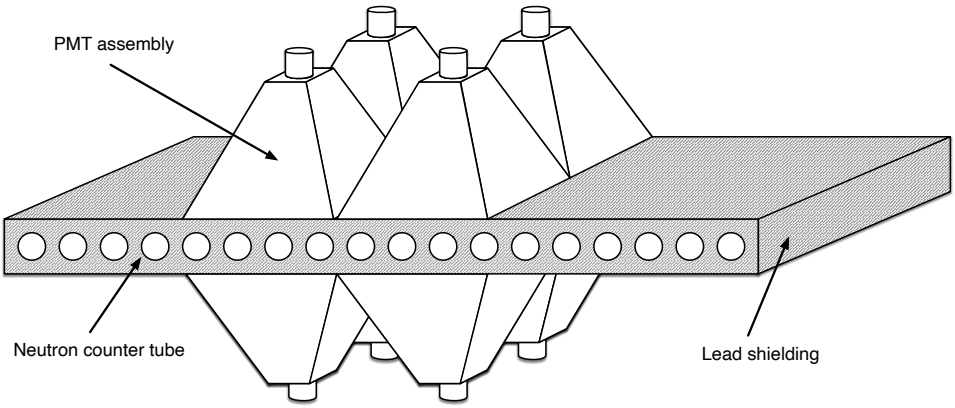


Figure 1.1: Muon telescope early project.

The telescope needed a data acquisition system capable of capturing signals from eight channels and perform several different coincidence tests to determine the particle arrival directions, so it became obvious at a very early stage that a solution based on instrumentation modules of some standard such as Nuclear Instrumentation Module (NIM) was unfeasible, due to the number of channels involved which would mean a high cost and great interconnection complexity. At the same time, using a previously designed acquisition system such as the one built for CaLMa was not possible, because it was an ad-hoc design not adequate for this experiment. So, a new data acquisition system needed to be designed again, and it was decided that it would be a great opportunity to use the knowledge acquired in previous designs to work on a proposal for an acquisition system design philosophy that did not have this problems, and was flexible enough to be used not only in this experiment, but also in future projects. As a result, the development of ARACNE as a design concept started.

Finally, to test the validity of the idea, and even though the design concept can be implemented differently depending on the desired capabilities of a specific scenario (i.e. number of channels, signal bandwidth or processing complexity), a prototype to demonstrate the feasibility of the concept has been designed and built, in a configuration that covers a wide range of applications while preserving a low power consumption and small footprint. Its development is described, and the result of its first application in a novel design for muon telescope is shown.

This book is divided in four parts, corresponding to chapters 2 to 5.

Chapter 2 provides a brief introduction to radiation detectors, the signals they produce and the elements of the detection chain that transform and condition this

signal so that the radiation embedded in the signal can be measured, processed and stored. Finally, the most relevant nuclear instrumentation module standards that serve this purpose are outlined.

Chapter 3 describes the design philosophy of ARACNE, the requirements of the platform and the resulting high level design.

Chapter 4 goes in detail in the design and production of the ARACNE prototype, divided in three main parts: the hardware platform, the IPCore and the software support libraries.

Finally, chapter 5 describes the Cosmic Ray Antarctic Observatory (ORCA) project, the instruments it's composed of and the application of the ARACNE prototype to the Muon Impact Tracer and Observer (MITO) telescope. Results from this project are shown, and an example of additional research that tackles the capabilities of the prototype and its performance in ORCA is described.

Chapter 2

Radiation and nuclear instrumentation

"And AC said, 'LET THERE BE LIGHT!'
And there was light–"

— Isaac Asimov, *The last question*

2.1 Introduction

Since the discovery of radiation in its many forms in the 19th and early 20th centuries, the creation of systems for its detection and measurement has been a field of research of great importance; not only because of its role in the study of radiation itself and the processes where it is produced, but also because of the potential practical applications of the various types of radiation. Nowadays, radiation detection and measurement systems are used in a wide range of applications, from medical and industrial imaging to nuclear physics research.

The field of electronic systems designed for nuclear and elementary particle physics research, and consequently for the study of radiation, is commonly known as nuclear electronics. These systems, usually referred to as front-end electronics, are composed of a chain of subsystems known as detection chain, as shown in Figure 2.1. Signals generated by a radiation detector are transformed stage by stage and ultimately turned into useful information about the phenomena under study. Some of these elements are preamplifiers, amplifiers, pulse shapers, discriminators, coincidence and anticoincidence logic, counters, pulse height analyzers, etc.

Historically, the elements in the detection chain have all been based on analog technology. However, the rapid development of digital electronics, the appearance of programmable logic devices such as Field-Programmable Gate Arrays (FPGAs), the development of digital signal processing techniques and the appearance of Digital Signal Processors (DSPs) has enabled most of the analog front-end electronics components to be replaced by their digital counterparts. This brings many advantages not only in terms of performance and capabilities, but also in cost, complexity and stability due to the digital nature of the signals used in such systems. Finally, the fast evolution of computing in the past two decades in terms of computing power, reduction in size and energy consumption and increase in memory and storage capacity, further boosts the convenience of digital processing in nuclear electronics, culminating in the appearance of System on-Chip (SoC) platforms that combine small form factor and power consumption with more than capable computing capabilities.

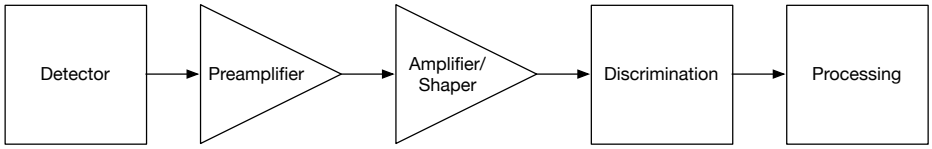


Figure 2.1: Example of a detection chain.

To build the front-end electronics of a detector, the elements of the detection chain were originally developed and built in the laboratories or sold by specialized companies as electronic modules. These modules were later standardized, offering a great advantage in flexibility, interchangeability and reduced system design effort. It also made maintenance and updating of detection systems easier. Some of these standards are Nuclear Instrumentation Module (NIM) and Computer Automated Measurement and Control (CAMAC), or the less nuclear instrumentation specific Versa Module Euro bus (VMEbus). However, these standardized modules can pose a challenge in terms of cost and size or sometimes lack some capabilities specific to a detector system which make them not a feasible alternative, thus forcing the development of custom ad-hoc solutions which are better tailored to the experiment but take much more time and effort to develop.

The following sections will briefly describe the characteristics common to many radiation detectors, the elements of a typical detection chain and the most important nuclear instrumentation module standards.

2.2 Radiation and particle detectors

The radiation types of interest in nuclear and particle physics, such as those produced by cosmic radiation, nuclear decay or reactions in a particle accelerator can be roughly classified in four main types. If the radiation consists of charged particles, they can be either heavy charged particles, with one atomic mass or greater (alpha particles, protons and other ions), or fast electrons, such as beta particles (positive or negative) and energetic electrons. On the other hand, radiation can be uncharged, and in this case electromagnetic radiation (such as X-rays or gamma rays) and neutrons are considered. The energy range of the considered radiation type can vary widely and the generation process has been thoroughly studied [1], so in this work we will only describe briefly the basics of this generation process and then focus on the signals generated by radiation detectors.

A radiation detector is a device used to detect just the presence of such radiation, but more sophisticated ones can also measure radiation attributes such as energy spectrum, timing events or the position of the radiation where it interacts with the detector. Detectors that require an energy source to produce a signal with information about the radiation are called active detectors, and they can be built using a variety of detection materials such as gas, liquid, semiconductors or scintillators. As it will be explained later, these detectors generate a current pulse when ionizing radiation interacts with their sensitive region which carries information about the nature of the radiation, its energy and the way it interacted with the detector. This pulse can be generated directly by the detector because the interaction induces electric current pulses in the detector electrodes, or indirectly, as is the case of scintillators, where radiation interaction produces photons that are then converted to an electric pulse by other device such as a Photomultiplier Tube (PMT). How this pulse is generated depends on the way the aforementioned radiation types interact with the detector materials as they pass through it (or, possibly, until the material stops or absorbs the radiation in question).

2.2.1 Radiation types

Heavy charged particles (e.g., an alpha particle) interact with matter through electromagnetic forces. As they traverse the detector, their positive charge affects electrons from the medium, which can be excited to a higher energy level in its atom or completely removed from it, thus generating a free electron and an ion. With each interaction the charged particle loses some of its energy, and since it interacts with many atoms at the same time from all directions, its trajectory in

the material is mostly a straight line until it stops or passes through it. In the former case, the higher the particle energy, the more atoms will be affected and the longer will be the trajectory of the particle in the material until it stops. In the latter, high energy particles lose the same amount of energy while traversing the detector. In both cases, the interaction time is typically in the few picoseconds range for solid and liquid detectors and a few nanoseconds in gases [1, p. 36].

Fast electrons (such as $\beta+$, $\beta-$ and energetic electrons) in the range of interest, with typical energies less than a few MeV, cause mostly a similar ionization effect [1, p. 43]; but since their mass is a lot smaller and equal to the electrons with whom they interact, they lose energy at a lower rate and their trajectory is not straight.

When the radiation passing through the detector is charged, as in the previous cases, it interacts continuously through the coulomb force with the electrons present in the material. But when the radiation is uncharged, it will pass through the detector without being noticed unless it interacts with the detector material in a way that changes both the material and the radiation. **Electromagnetic radiation** (such as gamma rays) can transfer part or all of its energy to electrons in the detector material through various mechanisms [1, p. 47-50], resulting in secondary electrons similar to fast electrons; and **neutrons** can only interact with nuclei of the material, mostly producing heavy charged ions through neutron-induced nuclear reactions.

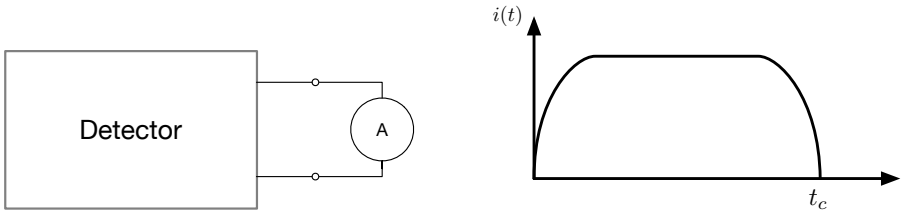
In-depth details of the interactions of the different types of radiation and the ionization they produce, which is the primary mode of interaction, is well documented [1, ch. 2]. But in general, the radiation will create ion pairs along its trajectory, each of which will imply the transfer of a certain amount of energy until the particle is stopped. Thus, the number of ion pairs produced in an interaction is related to the original particle energy and subsequently if the charge that appears in the detector is measured, this energy can be inferred [2, p. 12][1, ch. 5]. As mentioned before, the interaction time is small (in the ps to ns range), so it can be considered instantaneous in most practical situations.

2.2.2 Detector model

If the charge induced by the radiation interaction is left uncollected, it will typically recombine and cause no measurable effect, but recombination can be minimized by creating a strong electric field in the detector with an external power supply. This causes the positive and negative charge carriers created to drift in opposite directions; negative toward the anode and positive towards the cathode,

creating an induced charge in the electrodes. If the circuit between anode and cathode is closed, there will be a current between the electrodes for the duration of the collection time, that is, the time it takes for the charges to reach the electrode.

In the most simple case, if a single charge particle was generated in the detector it would immediately start to move towards the corresponding electrode and an ammeter connected to the detector terminals would read a current during the collection time, as illustrated in figure 2.2.



(a) Ideal detector and ampmeter. **(b)** Current representation.

Figure 2.2: Hypothetical signal from a single charged particle in an ideal scenario.

The exact shape of the current signal will depend on the value of the electrical field (which affects the speed of the particle), the charged particle type and the detector geometry. But, as it will be later explained, the time integral of $i(t)$ over the collection time will always be the generated charge Q.

$$\int_0^{t_c} i(t)dt = Q \tag{2.1}$$

Since a radiation interaction will cause many charged particles to be generated, the integral of the current generated in such situation will be the sum of all charges, and the collection time the is time it takes for all charges to reach their corresponding electrodes. Depending on the mobility of charge carriers (electron, alpha particles, holes, etc.) and the detector volume and shape, the characteristics of this pulse and the collection time varies from nanoseconds to a few microseconds [3].

The detector can thus be treated as a current generator, and in a more realistic scenario it will present an impedance and a capacitance as will the readout electronics (e.g a preamplifier), forming an RC circuit with a time constant $\tau = RC$ as shown in figure 2.3.

The current generated will appear as a voltage at the input of the preamplifier, and the signal shape will depend on the value of the RC product. Intuitively, if

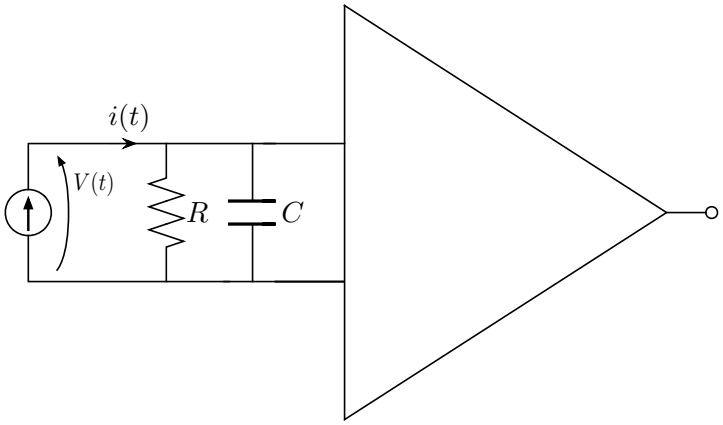


Figure 2.3: Detector and preamplifier circuit.

RC is small compared to the duration of the current pulse, the current will flow through the resistor and the shape of the voltage pulse will be the same as the current pulse. If on the contrary RC is very large, $i(t)$ will be integrated by the capacitor and the voltage signal will represent the charge collected along time. Afterwards, the charge stored in the capacitor will flow through the resistor and the voltage will decrease exponentially depending on the value of RC.

$$V(t) = \frac{Q}{C} e^{\frac{-t}{RC}} \tag{2.2}$$

In summary, a voltage pulse will be produced by the interaction in whose amplitude there will be information about the deposited charge. This will be later studied more thoroughly in section 2.2.4 with an example.

2.2.3 Modes of operation

Depending on the frequency of interactions there are two possible modes of operation: when interactions are very frequent and pulses are generated so fast that they overlap, the system can't measure them individually (i.e. it's not possible to infer the energy associated to each interaction); instead, a current whose average intensity is related to the energy and frequency of interactions at the detector is constantly produced and can be measured, which is enough for some applications. In this case, the detector is said to be working in *current mode*. On the other hand, if interactions are more spaced out and the current generated by a single interaction can be measured individually, the detector is said to be working in

pulse mode, which is important for spectroscopy applications. In this case, a current pulse is generated as described above, and the detector returns to the initial steady state until a new interaction occurs.

2.2.4 Ion chambers

To illustrate the process of pulse generation and understand the basics of a collection circuit, we will use an ion chamber as the simplest example. An ion chamber is a volume of gas (e.g. air) between two electrodes that are at a high potential. In order to provide a simplified explanation of the signal shape produced in this system, we will consider the electrodes are two parallel plates separated by a distance d at a V_0 potential provided by a high voltage source, which generates an electric field $\mathscr{E}$ across the ion chamber, sufficient to make ion recombinations negligible. This illustrated in figure 2.4.

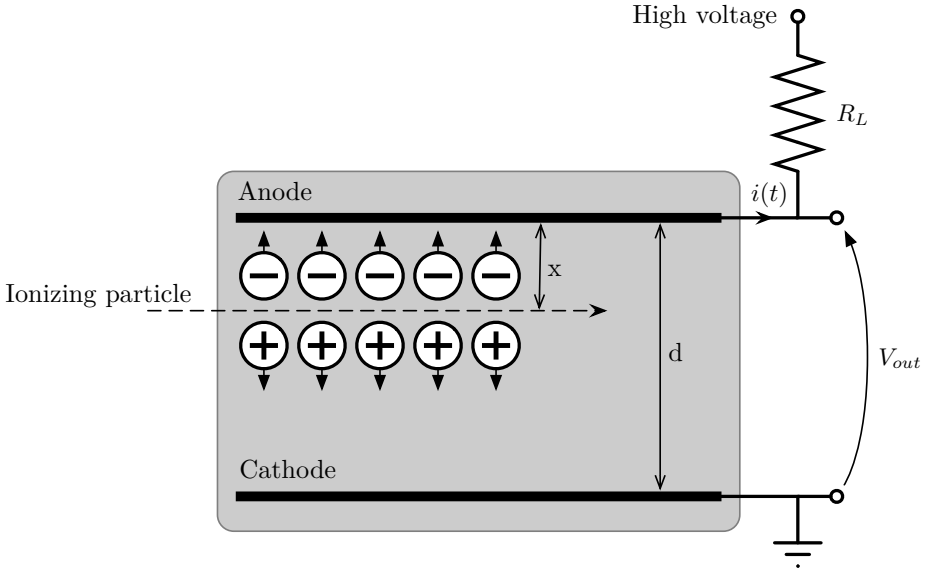


Figure 2.4: Ion chamber diagram.

The chamber will present a capacitance and a resistance that will appear in parallel to R_L, and the cabling and measurement circuit (e.g. a preamplifier) will also present an input impedance and a capacitance. A simplified equivalent circuit is shown in figure 2.5, where R represents all the mentioned impedances in parallel and C the sum of all capacitances. V_R is the signal to be measured and since in a steady state there's no current passing through R, initially it equals 0.

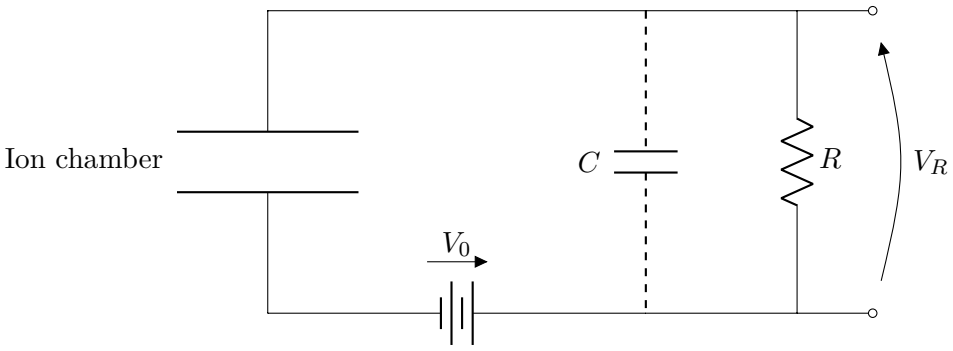


Figure 2.5: Equivalent of an ion chamber, from [1, p. 149]

When ion pairs are created in the chamber as a result of an interaction, the charged particles begin to move in opposite directions as a result of the applied electric field, at a speed v^- for electrons and v^+ for the positively charged ions, where v^- is much higher since the negatively charged particles have a much smaller mass and thus higher mobility. This induces a charge in the anode and cathode of the ion chamber that modifies its original voltage, and since V_0 is fixed, the change appears with the opposite sign in R. Assuming the ion pairs are all created at a distance x of the positive electrode, it can be demonstrated [1, p. 152] that V_R is given by:

$$V_R = \frac{n_0 e}{dC}(v^+ + v^-)t \tag{2.3}$$

Where n_0 is the number of ion pairs created and e is the charge of the electron. It can be observed that V_R increases linearly depending only on the amount of ionization, and does not depend on x. Also, since v^- is much higher than v^+, the slope of the rising edge of the signal depends mainly on the former.

This equation holds true until the electrons reach the anode at $t^- = x/v^-$, where they are collected. From that moment, their contribution to V_R will be constant and equal to $n_0 e v^- t^-/dC = n_0 e x/dC$. V_R will continue increasing due to the drift of the positive ions until they reach the cathode:

$$V_R = \frac{n_0 e}{dC}(v^+ t + x) \tag{2.4}$$

This will happen at $t^+ = (d - x)/v^+$. This is the maximum value of V_R, which becomes

$$V_R = \frac{n_0 e}{C} = \frac{Q}{C} \tag{2.5}$$

This is a very relevant conclusion, because it establishes that the maximum value of the signal is directly proportional to the charge generated in the detector and consequently to the deposited energy.

At this point, the circuit will start returning to equilibrium conditioned by the time constant determined by the RC product, which is assumed to be much longer than the collection time. Consequently, V_R is similar to a step signal, with a very short rise time, an amplitude proportional to the charge Q caused by the ionization, and a long falling time that depends on RC. Figure 2.6 shows the shape of the pulse.

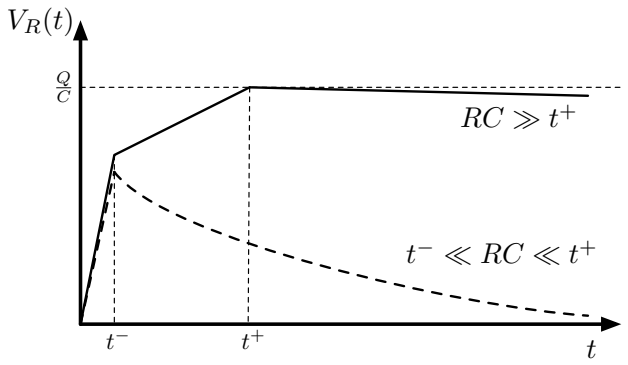


Figure 2.6: Pulse shape in an ion chamber.

It must be noted that t^- is much shorter than t^+ and v^- much higher than v^+, so the first part of the rising edge of the pulse is almost instantaneous compared to the second, where only positive ions are still drifting. And both are much shorter than the time constant $\tau = RC$ of the falling tail of the pulse, so the pulse is often treated as a unit step signal when circuits for the processing of this signal are being designed.

The problem with this approach is that RC must be longer than t^+ if all charges are to be collected, and the collection time for positive ions is already in the milliseconds range. This limits the pulse rate that can be achieved by the system to avoid the pile-up of consecutive events, a necessity if pulse mode operation is desired, so ion chambers are often operated in *electron sensitive* mode, where RC is higher than t^- but lower than t^+, as shown also in figure 2.6. In this case, the contribution of positive ions is lost and the maximum value of V_R is given by

$$V_R = \frac{n_0 e}{C} \cdot \frac{x}{d} \tag{2.6}$$

The pulse amplitude is still proportional to n_0, but now has an undesired dependence on the place ionization takes place, since the voltage induced is proportional to the potential traversed by the particle until it reaches the electrode. This is one of the reasons why other detector geometries such as cylindrical or grided are often used, in order for the majority of electrons to traverse the same potential and thus generate the same voltage [2, p. 16]. In a real situation where ion pairs will be produced at different x values, the result will be a pulse with a rising slope that is not completely linear (the slope will decrease as negative charges are collected at different points in time), and the peak pulse value will depend on the average of x.

This example with an ion chamber gives an idea of the expected shape of the signal produced in a radiation detector, and the basic concepts behind it. For detectors with a more complex geometry using the Shockley-Ramo theorem is more convenient [3, p. 6]. According to this theorem, the current and charge induced at an electrode by a moving charge is

$$i = -q\vec{v} \cdot \vec{E_0}(x) \tag{2.7}$$

$$Q = -q[\varphi_0(x_f) - \varphi_0(x_i)] \tag{2.8}$$

where $\vec{E_0}$ and φ_0 are the weighing field and weighing potential, a measure of electrostatic coupling between the moving charge and the sensing electrode that depend on the detector geometry; and $\vec{v}$ implicitly includes the effect of the electric field, since the charge drifting speed depends on its value.

2.2.5 Indirect charge production: scintillators and photomultiplier tubes

In scintillation detectors, charge production involves two steps. First, a detection material (i.e. a scintillator) generates photons when radiation impacts the detector; and second, a light-sensing device is used to generate an electric signal when these photons are collected.

Scintillators are a very popular type of detectors because their high efficiency and the fact that they can be made in different shapes and large sizes. Their basic principle of operation is that when radiation impacts the detection material,

molecules are excited and some of their electrons reach a higher energy state; and when they de-excite (usually after other radiationless transitions to intermediate excitation states), they emit photons characteristic of the used material. They can be gaseous, liquid or solid, and made from organic or inorganic materials depending on the radiation to be detected. In all cases, the decay of the luminescence must be short and the wavelength of the emitted radiation compatible with the response of the light-sensing element. A detailed account of the different types of scintillators and their characteristics can be found in [2, p. 27-31] and [1, Ch. 8].

The photons produced by a scintillator are in turn detected by very sensitive light-sensing devices, such as Photomultiplier Tubes (PMTs). PMTs are capable of converting extremely weak light signals from scintillation pulses in the corresponding electrical signals, sometimes not even requiring the use of amplification. They consist of a photocathode, a series of dynodes each at a higher potential, and a collecting electrode. This is built inside an enclosure (typically a tube made of glass) where vacuum has been made to allow electrons to flow easily, and the photocathode is placed close to a window in the enclosure or directly deposited as a semitransparent layer on it. Figure 2.7 shows a diagram of the device.

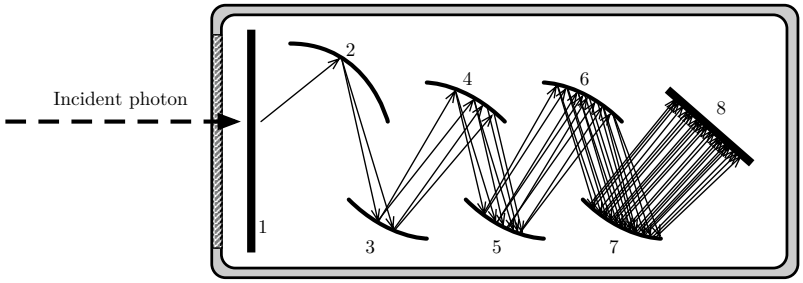


Figure 2.7: Photomultiplier tube diagram. (1) Photocathode, (2-7) dynodes, (8) charge collector.

When photons impact at the photocathode they generate a few electrons due to the photoelectric effect, which are accelerated towards the first dynode where they generate secondary electrons. These are then accelerated towards the second dynode and the process repeats producing a multiplicative effect that results in an avalanche of electrons, which are finally collected at the anode producing an electrical pulse which is proportional to the amount of light gathered, hence providing a measurement of the amount of energy deposited in the scintillator.

2.3 The detection chain

2.3.1 Preamplification

The output of a radiation detector is a charge pulse generated by the incident radiation. Sometimes the amount of charge is sufficient to generate a voltage pulse large enough simply by integrating the charge in the capacity of the detector itself, the wiring and the next stage of the chain, as explained before; but in most cases, mainly in detectors based on ionization caused by incident radiation, the charge is so small that an amplification stage called preamplifier (placed as close as possible to the detector in order to minimize noise) should be used. The basic function of a preamplifier is to extract the output signal from the detector with minimal degradation of the information to be measured, which may be contained in the amplitude or shape of the signal pulses, and provide a sufficiently large output to transmit the signal through the cables connecting it to the next stage, which usually have a high capacitance.

But this is not the only reason. If the maximum voltage at the output of the detector directly depends on the capacity of the detector itself (plus wiring, etc.) according to the equation $V_{max} = Q/C_d$, and this voltage is handled by a regular voltage amplifier, since the intrinsic capacity of the detector can vary over time for many reasons, the system is sensitive to these variations losing the linear relationship between the charge generated and the maximum voltage produced. For this reason it is important to use preamplifiers that are not sensitive to these changes in capacity, the most common of which are the so-called charge-sensitive preamplifiers, which also provide good linearity and high sensitivity, reflecting at their output the charge pulses produced in the detector. We will focus here on this type of preamplifiers and study the type of signal they generate.

The operational amplifier

The basic building block of many analog circuits including preamps is the operational amplifier (figure 2.8). It is a device with two inputs (inverter and non-inverter) that, ideally, outputs a signal inversely proportional to the difference between its inputs with a very high gain A, presenting also (in the ideal case) an infinite input impedance and zero output impedance.

Equation 2.9 describes the ideal behaviour of an operational amplifier:

$$V_{out} = -A(v_+ - v_-) = -AV_{in} \tag{2.9}$$

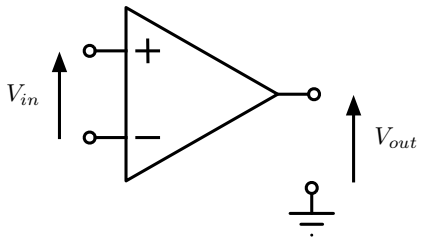


Figure 2.8: Operational amplifier.

Also, because the gain is very high, the voltage difference between the input terminals has to be very small for the amplifier to be working within its output limits. This is an extremely important factor since it allows simplifying many calculations by considering that $V_{in} \approx 0$ (although this can't really be true), and when one of the terminals is connected to ground, the other can be considered at *virtual ground*, which is very useful to design circuits with operational amplifiers based on negative feedback, like those shown in figure 2.9.

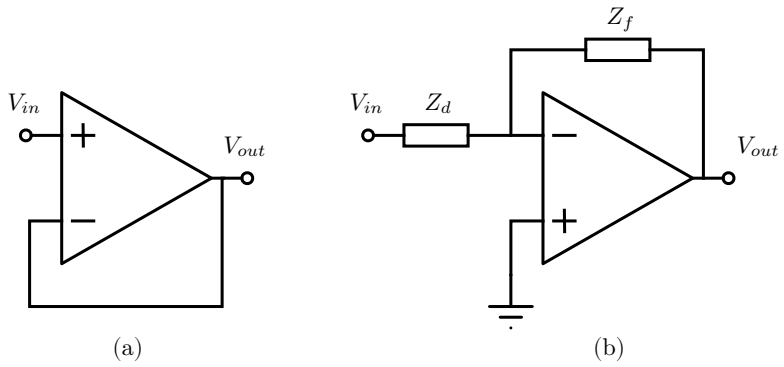


Figure 2.9: Basic circuits with an operational amplifier. (a) Voltage follower. (b) Negative feedback amplifier.

The first example shows a basic circuit built on these premises called voltage follower. Applying the equation 2.9 it can be inferred that $V_{out} \approx V_{in}$, and at the same time the circuit provides a very high input impedance and a very low output impedance, providing isolation between stages in a very simple way.

The second example is a basic amplifier circuit with negative feedback in which it can be easily calculated that

$$V_{out} = V_{in} \frac{Z_f}{\frac{Z_f + Z_d}{A} - Z_d} \tag{2.10}$$

and, if A is very high, the expression can be simplified as

$$V_{out} \approx -V_{in}\frac{Z_f}{Z_d} \tag{2.11}$$

It is possible to arrive at the same conclusion directly by using the aforementioned virtual ground effect in terminal V_- because V_+ is grounded. Since the current passing through both resistors is $I = V_{in}/Z_d$, $V_{out} = -IZ_f = -V_{in}Z_f/Z_d$.

The Miller effect

A relevant effect to study is how a feedback impedance modifies the apparent input impedance of an operational amplifier, according to what is called the Miller effect. According to figure 2.10, and given that $V_{out} = -AV_{in}$, it follows:

$$I_{in} = \frac{V_{in} - V_{out}}{Z_f} = \frac{(1+A)V_{in}}{Z_f} \tag{2.12}$$

And therefore the apparent impedance at the input of the operational amplifier generated by Z_f is

$$Z_{in} = \frac{V_{in}}{I_{in}} = \frac{Z_f}{1+A} \tag{2.13}$$

That is, the feedback impedance is reflected at the input divided approximately by the gain of the operational amplifier, as shown in figure 2.10. In the case of a pure resistor, its value will be divided by the gain; while in the case of a capacitor, whose impedance in the Laplace domain is $Z_c = 1/sC$, its capacity will appear multiplied.

This will be very important in the design of the charge-sensitive preamplifier.

The charge-sensitive preamplifier

A charge-sensitive preamplifier is composed of an operational amplifier, a capacitor and a circuit to discharge the latter which will be explained later. In figure 2.11, the detector is ideally represented by a current generator and a capacitor C_d representing the intrinsic capacity of the detector, the wiring and other parasitic capacities. The simplest charge amplifier uses just a C_f capacitor as feedback between the output and the input.

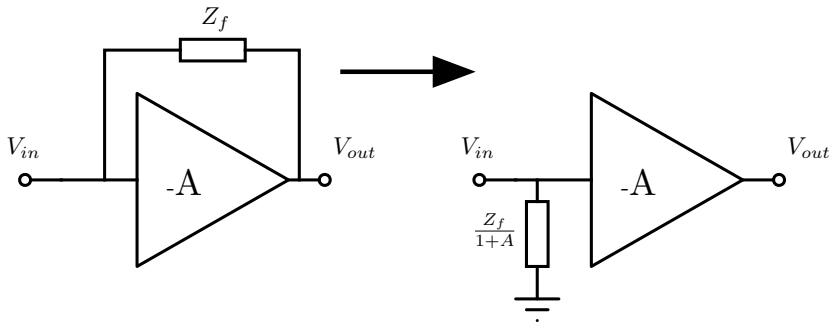


Figure 2.10: Illustration of the Miller effect.

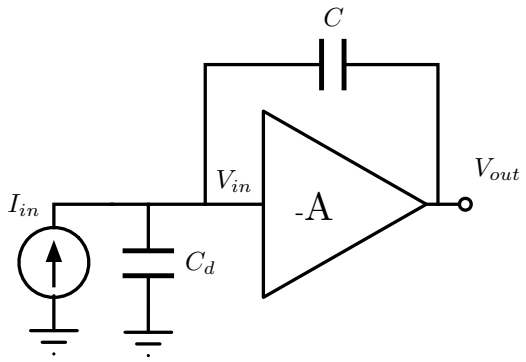


Figure 2.11: Basic charge amplifier.

In this configuration, as explained in section 2.3.1, the capacitor C_f appears reflected at the input of the amplifier multiplied by $(A + 1)$. Since the gain of the amplifier is very high, the effect is that the detector sees at the input of the amplifier a very large capacitor that appears in parallel with C_d, and since the voltage at the detector output depends on the total capacitance it sees, the effect of variations in the detector characteristics, cabling etc. (and consequently, the value of C_d) can be minimized, keeping the output mostly independent of these variations.

Therefore, the detector will generate a maximum voltage at the input of the operational amplifier

$$V_{in} = \frac{Q}{C_d + (A+1)C_f} \tag{2.14}$$

And on the output we will have

$$V_{out} = -AV_{in} = -\frac{AQ}{C_d + (A+1)C_f} \tag{2.15}$$

which can be simplified, if $A \gg 1$ and $(A+1)C_f \gg C_d$

$$V_{out} \approx -\frac{Q}{C_f} \tag{2.16}$$

In conclusion, the amplifier generates a voltage that depends on a component that is under control, C_f, and is virtually independent of variations in other capacities.

To analyze the shape of the output signal, we can calculate the transfer function of the system $H(s)$ as the relationship between the output voltage $V_{out}(s)$ and the input current $I_{in}(s)$.

$$V_{out}(s) = -AV_{in}(s) = -\frac{I_{in}(s)}{sC_f} \tag{2.17}$$

$$H(s) = -\frac{1}{sC_f} \tag{2.18}$$

Thus, the circuit integrates the signal $I_{in}(t)$, and as shown before, $\int_0^\infty I(t) = Q$. But also, since the generation of the charge occurs in a very short time, it can be considered that

$$I_{in}(t) = Q \cdot \delta(t) \tag{2.19}$$

$$V_{out}(s) = -\frac{Q}{C_f}\frac{1}{s} \tag{2.20}$$

Consequentially, the response $V_{out}(t)$ to the charge generated by the detector is a negative voltage step of amplitude Q/C_f which remains constant over time. The amplitude of this signal is shown on figure 2.12.

Under these conditions C_f would store the charge and keep its voltage constant, and successive events would increase the voltage level until reaching the power supply limit of the operational amplifier. That is why some mechanism to discharge the capacitor, such as a R_f resistor with a high value as shown in figure 2.13, is necessary.

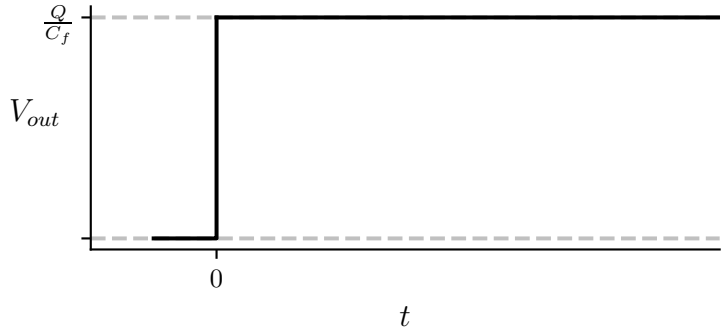


Figure 2.12: Output of a charge amplifier with no reset mechanism.

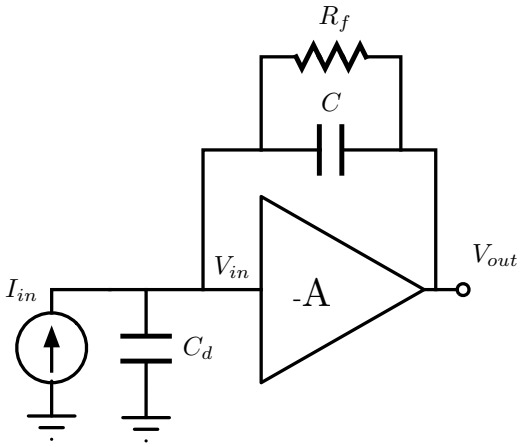


Figure 2.13: Charge amplifier with discharge resistor.

This resistor would be reflected at the input of the amplifier with a value $R_f/(A+1)$ in parallel with the also reflected $C_f(A+1)$, and the transfer function $H(s)$ can be calculated as

$$H(s) = -\frac{A}{A+1} \cdot \frac{1}{C_f} \cdot \frac{1}{s + \frac{1}{C_f R_f}} \tag{2.21}$$

If $A \gg 1$ and $\tau = R_f C_f$,

$$H(s) = -\frac{1}{C_f} \frac{1}{s + \frac{1}{\tau}} \tag{2.22}$$

And therefore, in the event of a charge pulse generated by the detector, the output of the preamplifier would be

$$V_{out}(s) = -\frac{Q}{C_f}\frac{1}{s+\frac{1}{\tau}} \tag{2.23}$$

That is, in the time domain the output signal would be a negative exponential of amplitude $V_{max} = Q/C$ decaying with a time constant τ, as shown in figure 2.14.

$$V_{out}(t) = V_{max}e^{-\frac{t}{\tau}} \tag{2.24}$$

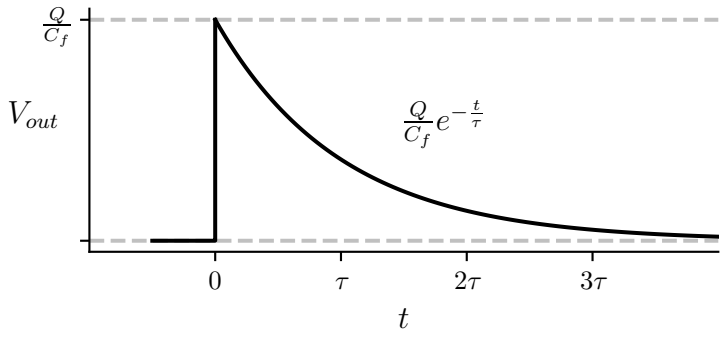


Figure 2.14: Charge amplifier output.

The example shows intuitively that the tail pulse generated retains the maximum amplitude value Q/C_f and has a fall time that depends on the choice of components (mainly C_f and R_f) with a time constant τ, while the rise time is instantaneous since collection of charge is modeled as a $\delta(t)$. Naturally, real charge collection does not occur instantaneously and components are not ideal, so the rise time of the signal, although short, is not zero; and this makes the choice of τ non-trivial. If instead of an instantaneous surge of current $I_{in}(t) = Q \cdot \delta(t)$ the charge collection is modeled as a square pulse with a duration equal to the collection time, T_c, the current generated can be represented as

$$I_{in}(t) = \frac{Q}{T_c}(u(t) - u(t - T_c)) \tag{2.25}$$

where $u(t)$ is the unit step function and again, $\int_0^\infty I(t) = Q$. Consequently,

$$I_{in}(s) = \frac{Q}{T_c}\left(\frac{1}{s} - \frac{1}{s}e^{-T_c}\right) \tag{2.26}$$

$$V_{out}(s) = I_{in}(s)H(S) = \frac{Q}{C}\cdot\frac{1}{T_c}\cdot\frac{1}{s+\frac{1}{\tau}}\left(\frac{1}{s} - \frac{1}{s}e^{-T_c}\right) \tag{2.27}$$

which results in

$$V_{out}(t) = \frac{Q}{C}\cdot\frac{\tau}{T_c}\left(\left(1-e^{\frac{-t}{\tau}}\right)u(t) - \left(1-e^{\frac{-(t-T_c)}{\tau}}\right)u(t-T_c)\right) \tag{2.28}$$

Representing the output amplitude for different values of T_c in relation to the time constant of the circuit τ (see figure 2.15) it can be inferred that the τ value not only impacts the fall time of the tail pulse, but also the amplitude of the pulse. If the time constant is too low, part of the pulse amplitude is lost; an effect known as *ballistic deficit*. Otherwise, if the time constant is high, the pulse generated is similar to a voltage step of amplitude $V_{max} = Q/C_f$ shown in figure 2.12.

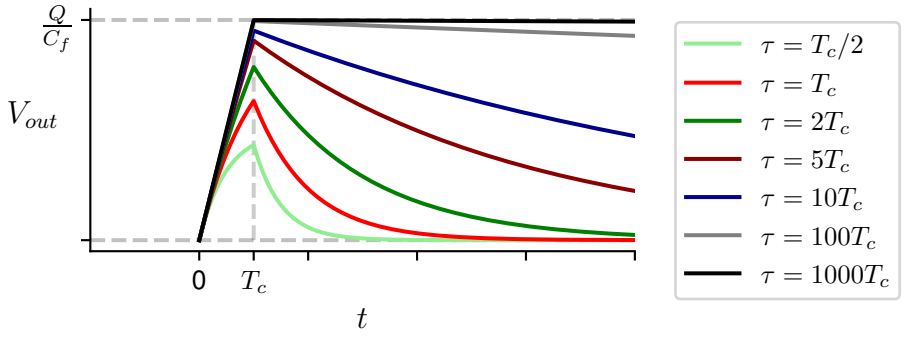


Figure 2.15: Output of a charge amplifier with a time constant τ for a charge pulse of duration T_c.

It is common to make the decay time quite large (between 50 µs and 100 µs) to allow full collection of the charge before the pulse starts its decay ([1, p. 610]), so the pulse generated by the charge preamplifier is usually regarded as a step function for the next stages of the chain.

2.3.2 Amplification and shaping

The amplitude of the tail pulse provided by the preamplifier is usually in the tens to hundreds of mV, which is too small to use directly and also very vulnerable to noise if the signal is to be transmitted through cables to the rest of the electronics. However, there are other problems not related to the pulse amplitude, but its shape, that cause a loss in resolution.

First, since the pulse decay is relatively slow (to the point that it is usually considered a step signal for the following stage), pile-up of pulses at higher rates is very likely. Second, for pulse amplitude applications the pointy shape of the tail pulse is difficult to work with, given the short time the pulse is at its maximum value. Finally, the short rise time means the pulse has high frequency components prone to be affected by the capacity of the cabling and certain types of noise.

For this reasons, it is usually necessary to use an amplification and shaping stage that provides at its output a shaped linear pulse of much larger voltage, matching the input range of the next element in the detection chain (e.g in the 0 to 10 V range, which is the standard for NIM) and a shape that retains all the important information but is easier to transmit and work with, e.g.: a pulse of shorter length and easier to measure peak value, that retains the proportionality between the charge deposited in the detector and the pulse amplitude. The element that provides this functionality, which typically defines the performance of the system, is a linear amplifier, which is characterized by a shaping time constant that defines the shape and length of the output pulse.

Choices must be made, however, in the design of the shaping circuitry of the linear amplifier depending on the application. When dealing with low pulse rates, the system can be usually designed to meet all the aforementioned requirements using a long time constant; but trade offs are often needed when dealing with high pulse rates that require the use of smaller time constants to minimize pile-up, which usually works against achieving a good signal to noise ratio or reducing the ballistic deficit of the amplifier.

The pulse pile-up problem is the easiest to understand. As mentioned before, pulses generated by the preamplifier have a relatively long decay time, which means a pulse can occur when the previous one has still not returned to the signal baseline value. This can happen also after the shaping stage, and means that the peak value of the overlapping pulse does not represent the charge deposited in the detector, since its value is offset a certain amount by the tail of the previous pulse as shown on figure 2.16. The solution for this is to reduce the tails making the pulse shorter, quickly returning to the baseline; for example using

a high-pass filter or differentiator with a time constant small enough to remove the tail.

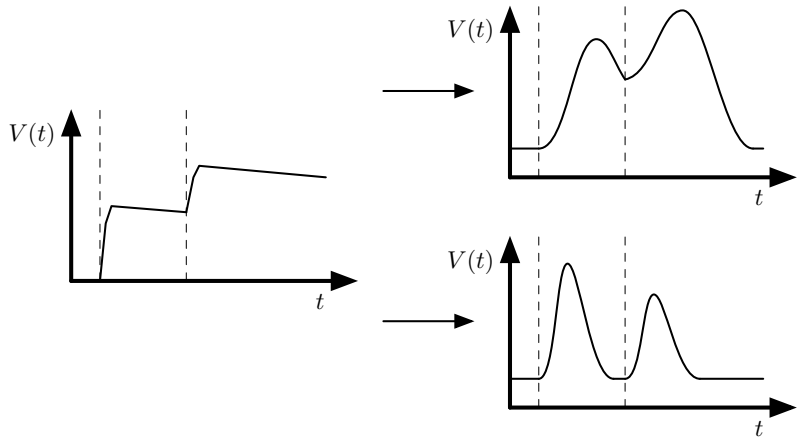


Figure 2.16: Example of pulse pile-up both at the preamplifier (left) and after the shaping stage (top). It can be avoided using a smaller shaping time constant (bottom).

The ballistic deficit, on the other hand, surfaces when the time constant τ of the shaping circuit is not large enough compared to the collection time T_c, and the full amplitude of the preamplifier pulse is not preserved in the resulting linear pulse, as shown in figure 2.15 for the preamplifier stage. The problem, however, can be solved easily if the charge collection times are always the same, because in this case the percentage loss of the amplitude is constant, and thus, can be compensated. But if the charge collection time T_c is variable, events that cause the same ionization and consequently generate the same charge in the detector are registered as pulses of different amplitude as shown in figure 2.17. In this case, the ballistic deficit cannot be corrected and the result is a measure resolution degradation.

Finally, **the noise to signal ratio** is not changed by the amplification gain, since it would affect signal and noise equally. But it is affected by the shaping. The noise in the signal is mainly introduced in the preamplifier stage, and has been thoroughly studied in other literature ([4], [2, Ch.5]); but in general, the frequency spectrum of all the types of noise combined is very broad, and for analytical purposes it is often assumed to be white and of uniform distribution ([1, p. 630-631]). Since the significant signal components are in a certain band of interest, the goal of the shaping circuitry is to eliminate as much broad spectrum as possible in order to obtain a better signal to noise ratio without affecting the useful signal components. This is can be achieved by a combination of high-pass

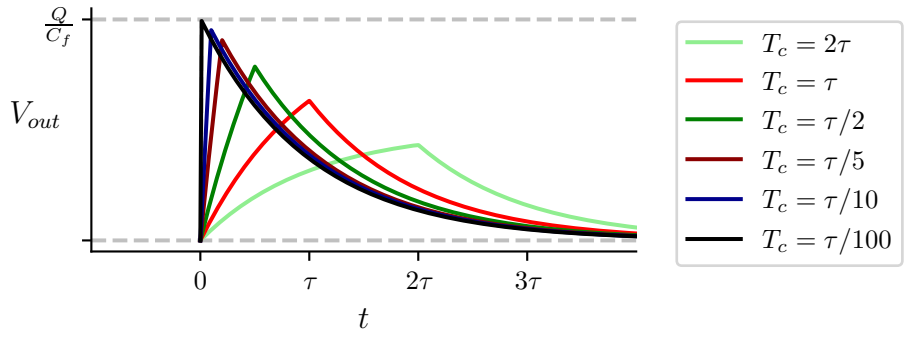


Figure 2.17: Output of a charge amplifier with a time constant τ for different collection times, T_c. Even though Q does not change, the pulse amplitude varies with different values of T_c.

and low-pass filters (i.e. differentiating and integrating circuits), also with their defining time constants.

In summary, apart from signal amplification:

- The ballistic deficit must be minimized through long enough shaping times.
- Pile-up must be minimized through the use of short enough shaping times.
- Signal to noise ratio must be as good as possible, through the use of an appropriate shaping method and filtering [2, Ch. 4].

Pulse shaping techniques have been thoroughly studied ([2, Ch. 3], and simple methods such as CR-RC shaping and pole-zero cancellation have been widely used in practical projects [5], traditionally using analog circuits. However, with the appearance of adequate Analog to Digital Converters (ADCs) and the fast development of digital technology, shaping can be implemented also in the digital domain, which provides significant advantages and allows the application of more advanced shaping algorithms that are difficult to achieve using purely analog circuits [4].

CR-RC shaping

The most basic shaping circuits are built using a capacitor and a resistor, and are usually referred to as *RC circuits*. Just like in the preamplifier circuit, their

behavior depends on a time constant $\tau = RC$, and depending on the position of the R and C elements, the circuits are called CR or RC.

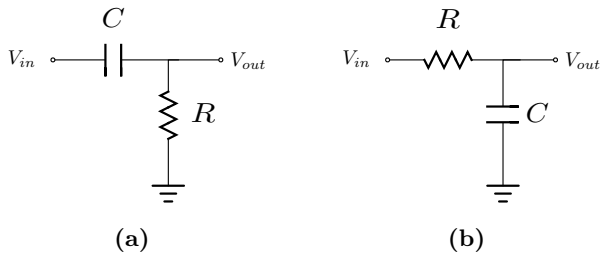


Figure 2.18: CR and RC circuits.

Circuit in figure 2.18a is a differentiator and high-pass filter, whose transfer function is

$$H(s) = \frac{s}{s + \frac{1}{\tau}} \tag{2.29}$$

Since in the frequency domain, with $s = j\omega$, $|H(s)|$ equals 0 for $\omega = 0$ and 1 for $\omega = \infty$, the circuit is a high-pass filter where $\omega = 1/\tau$ is the cutoff frequency. Also, if τ is very small, $H(s) \approx s$ and the circuit would be effectively a differentiator. The unit step response of the circuit, represented in figure 2.19a, is

$$V_{out}(t) = e^{\frac{-t}{\tau}} \tag{2.30}$$

In conclusion, the response of this circuit to a voltage step is the same as that of the charge amplifier on figure 2.14 to a current pulse. On the other hand, the circuit in figure 2.18b is an integrator and low-pass filter, whose transfer function is

$$H(s) = \frac{1}{\tau}\frac{1}{s + \frac{1}{\tau}} \tag{2.31}$$

Conversely to equation 2.29, in the frequency domain $|H(s)|$ equals 1 for $\omega = 0$ and 0 for $\omega = \infty$, so the circuit is a low-pass filter where $\omega = 1/\tau$ is again the cutoff frequency. Also, if τ is very large, $H(s)$ tends to an expression of the form $1/s$, and the circuit would be effectively an integrator. The unit step response of the circuit, represented in figure 2.19b, is

$$V_{out}(t) = 1 - e^{\frac{-t}{\tau}} \tag{2.32}$$

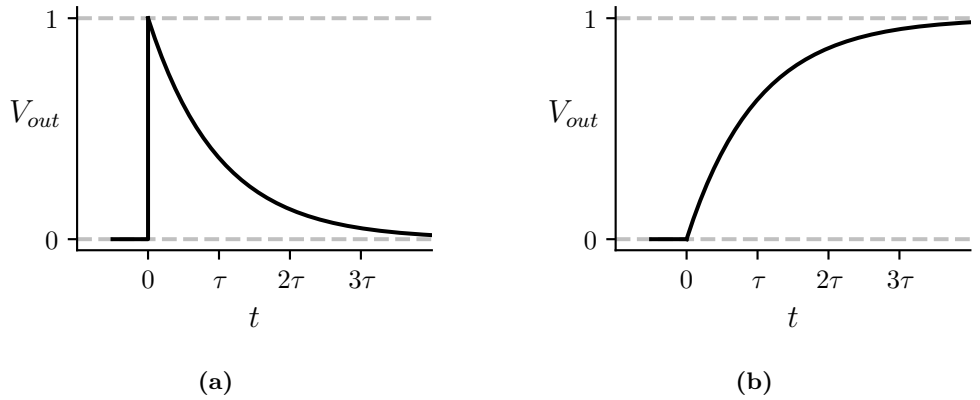


Figure 2.19: Step response of CR and RC circuits.

In conclusion, a CR network with a τ considerably shorter than the time constant of the preamplifier circuit could be used to shorten the length of the pulse produced by it and prevent pile-up while preserving the pulse amplitude. But the pulse shape would still be difficult to work with and would have high frequency components and well as noise, mixed with the signal. For this reason, a RC network is added to filter out the high frequencies and produce a smoother pulse, forming the basic CR-RC shaping network. Figure 2.20 shows the detector with the CR-RC shaping circuit, where there is a voltage follower between stages so that each stage impedance doesn't affect the others.

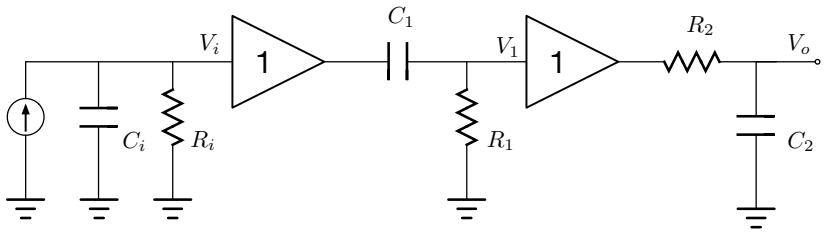


Figure 2.20: CR-RC shaping circuit diagram.

In the aforementioned diagram, the leftmost side represents the detector and the charge preamplifier equivalent circuit, as discussed in section 2.3.1. If R_i is large enough, the produced signal can be considered a step and the CR and RC circuits would generate a shaped pulse whose characteristics would depend on the value of τ_1 and τ_2. Figure 2.21 shows the response to a voltage step of the CR-RC network for various values of the time constants.

It is considered that keeping $\tau_1 = \tau_2$ is the best combination of time constants [2, p. 100] to obtain a flat pulse top, reduced ballistic deficit, quick return to 0

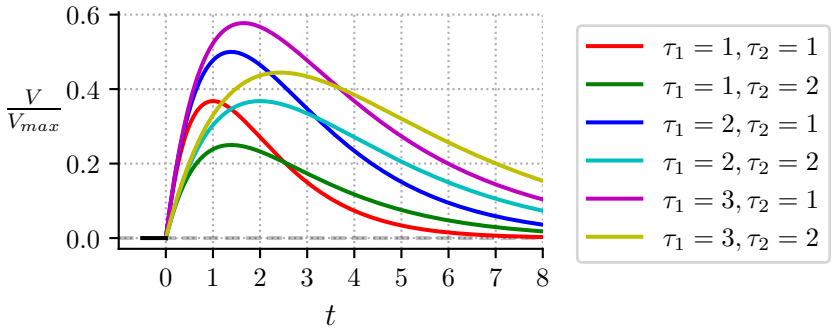


Figure 2.21: CR-RC circuit response to a V_{max} voltage step for different time constants.

volts and best signal to noise ratio.

A problem arises, however, because the real input is not a step but a slowly decaying exponential as described in equation 2.24. In this case, the output of the CR-RC network presents an *undershoot*, e.g. the signal takes negative values for some time as shown in figure 2.22. The undershoot is larger as the time constant of the RC-CR network is larger compared to that of the preamplifier, and if the negative tail is long enough, it can affect the amplitude of the next shaped pulse and consequently the resolution can be affected. A solution is to keep time constants low, but that can also be a problem because if they are comparable to the collection time, the result is a ballistic deficit. So a balance must be found, but it is also possible to use alternative shaping methods like the pole-zero compensation network presented in section 2.3.2.

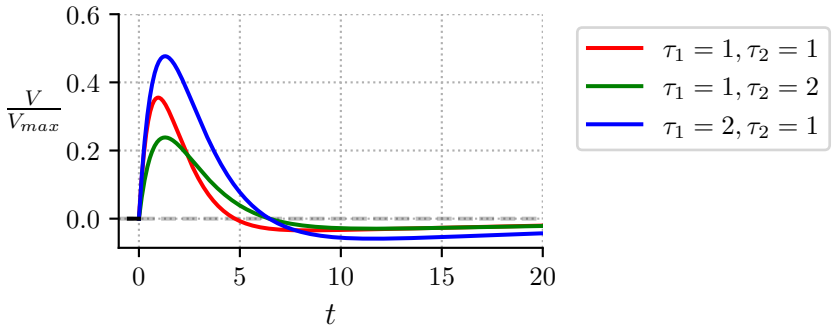


Figure 2.22: CR-RC circuit response to the tail pulse generated by the charge preamplifier for various τ_1 and τ_2 values. In this case, $\tau_i = 20$.

CR-(RC)^n shaping

In order to obtain a more symmetrical pulse shape with faster return to zero, several RC stages can be used after the CR stage, which makes the shaped pulse resemble a Gaussian. With this $\mathrm{RC}-(\mathrm{CR})^n$ shaping circuit, the peak is reduced and occurs at $n\tau$ for n RC stages, but this can be corrected adjusting the time constant and gain. It is considered that for $n = 4$ pulse shape is almost exactly Gaussian, so this is a very frequent configuration. Figure 2.23 illustrates the results obtained with a $\mathrm{CR}-(\mathrm{RC})^n$ shaping circuit for $n = 1$ to $n = 4$, and the shape of a pulse obtained with a $\mathrm{CR}-(\mathrm{RC})^4$ circuit after adjusting τ and gain to match the peaking time and amplitude of the basic CR-RC circuit.

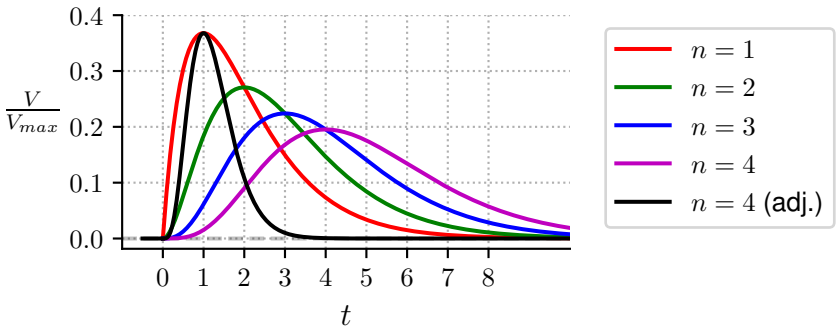


Figure 2.23: CR-(RC)^n circuit response to a unit step signal for various n, and adjusted pulse for n=4.

Pole-zero cancellation shaping

The undershoot that appears in the CR-RC shaping circuit shown in figure 2.22 is a consequence of the charge amplifier not producing a voltage step, but a negative exponential with a large $\tau = \tau_i$. This is a consequence of its transfer function (equation 2.22). Combining the pulse amplifier with the CR-RC stage results in a transfer function of the form

$$H(s) = \frac{1}{C_f}\frac{1}{\tau_2}\frac{s}{\left(s+\frac{1}{\tau_i}\right)\left(s+\frac{1}{\tau_1}\right)\left(s+\frac{1}{\tau_2}\right)} \tag{2.33}$$

where it's evident that if $\tau_i = 0$ (e.g. the output of the preamplifier was a voltage step), the pole due to the preamplifier would be at $s = 0$ and it would cancel out with the zero introduced by the CR stage at $s = 0$. The purpose of using

a pole-zero cancellation stage instead of the CR stage is precisely to introduce a zero at the same place as the preamplifier, thus eliminating the undershoot. This is achieved for example with the circuit shown in figure 2.24.

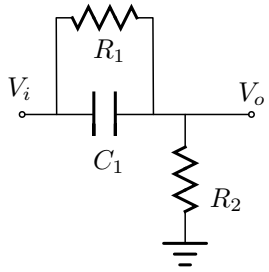


Figure 2.24: Pole-zero circuit.

The transfer function of this circuit is

$$H(s) = \frac{s+a}{s+b} \quad \text{where} \quad a = \frac{1}{R_1 C_1} \quad b = \frac{1}{C1}\left(\frac{1}{R_1} + \frac{1}{R_2}\right) \tag{2.34}$$

R_1 and C_1 can be chosen to match τ_i, and then R_2 can be chosen to obtain the desired τ_1, possibly matching τ_2, so that the transfer function of the system is

$$H(s) = \frac{1}{C_f}\frac{1}{\tau_2}\frac{1}{\left(s+\frac{1}{\tau_1}\right)\left(s+\frac{1}{\tau_2}\right)} \tag{2.35}$$

Figure 2.25 shows the effect of replacing the CR stage with a zero-pole cancellation stage, effectively removing the undershoot. In this example, $\tau_1 = \tau_2 = 1$ and $\tau_i = 20$.

2.3.3 Cabling and impedance matching

Interconnection between components of the signal chain is achieved using cables, with the coaxial type being the most used. This type of cable is composed of a central wire which carries the signal and an outer braided shield that provides a return channel, while minimizing the effect of noise produced by external electromagnetic fields. Between the two conductors a dielectric material is used, and an insulator sleeve protects the assembly.

The signal propagates through the cable at a speed mostly determined by the dielectric material. Where polyethylene is used as dielectric, which is the most

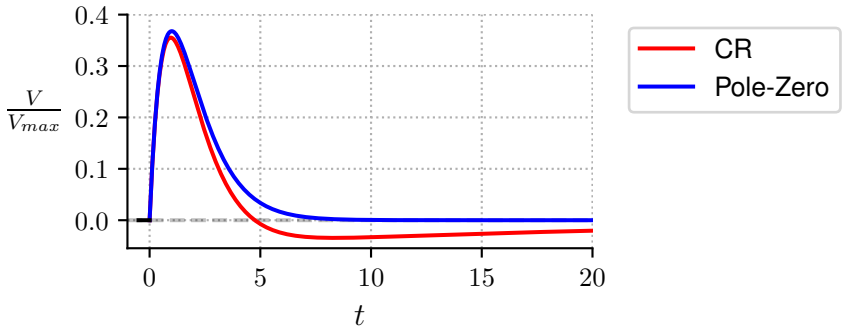


Figure 2.25: Response to an ideal preamplifier tail pulse output using a CR stage and a pole-zero compensation stage.

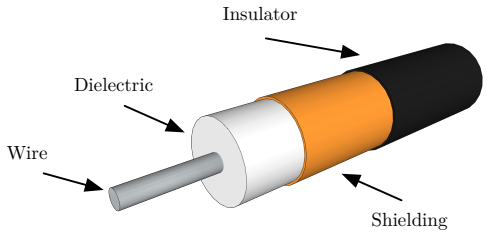


Figure 2.26: Structure of a coaxial cable.

common case, this speed is approximately 2/3 the speed of light in a vacuum. The cable also presents an attenuation and a capacity characteristic that increases linearly with cable length. All this need to be taken into account when fast pulses or high signal frequencies are used, more so with long cable lengths. A pulse is considered fast or slow depending on the comparison between its fastest frequency component (usually related to the rise time of the pulse) and the time it takes for the pulse to propagate through the cable, which for cables with polyethylene dielectric as mentioned before is around 5 ns/m. If the rise time is comparable or shorter than the propagation time, then it's considered a fast pulse.

When dealing with slow pulses, the main concern will be the cable resistance and capacitance to ground. Resistance will produce a signal attenuation which will be low if the cable is not very long, but capacitance is more important at some stages, such at the interconnection between the detector and the preamplifier where noise is affected by this capacity. Thus, in this interconnection, it is desirable that cable length is kept to a minimum.

But for fast pulses the characteristic impedance of the cable (usually referred

to as Z_0) will be more important. This impedance does not depend on cable length, but on dielectric material and diameter of the conductors, and it can be interpreted as the impedance that a voltage generator *sees* when a voltage step is applied to the cable. As current is drawn while the pulse travels through the cable as shown in figure 2.27, an impedance Z_0 can be calculated as the ratio between the voltage applied and the current drawn, which is the characteristic impedance of the cable. Cables used for nuclear electronics are designed to have a specific Z_0, and values of $75\,\Omega$, $93\,\Omega$ or most frequently $50\,\Omega$ are typical.

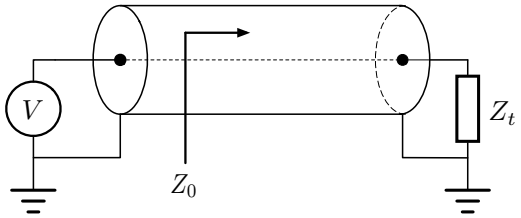


Figure 2.27: Terminated coaxial cable.

Termination

Now, depending on how the end of cable is *terminated*, several things can happen when the pulse arrives at it. If the cable is terminated with a load (or a device with an input impedance) of Z_0 value connecting the wire and the shield, the same current that was traveling through the cable will flow through the load, as impedances match. The impedances are then said to be *adapted*. But if the cable is terminated otherwise, the relationship between the current and voltage traveling through the cable cannot hold at the termination point, and a signal reflection occurs. For example, if the cable is terminated with a short, voltage in the cable must end up being zero, so a negative voltage step of the same amplitude travels back through the cable to the source. On the other hand, if the cable ends in an open circuit current must be zero; so a pulse of the same amplitude and sign travels back through the cable. For termination impedance values in between, but different to Z_0, part of the signal is reflected. And since the same effect happens when the signal reaches the source, more reflections can happen causing problems with the integrity of the signal transmitted if we're dealing with pulses with lengths comparable to the propagation time on the cable.

In conclusion, impedance matching is a very relevant issue when dealing with certain types of signals, and connections must be properly terminated to avoid signal reflections in these cases. If, for example, a $Z_0 = 50\,\Omega$ cable is used to connect an amplifier to the input of another element of the chain that presents

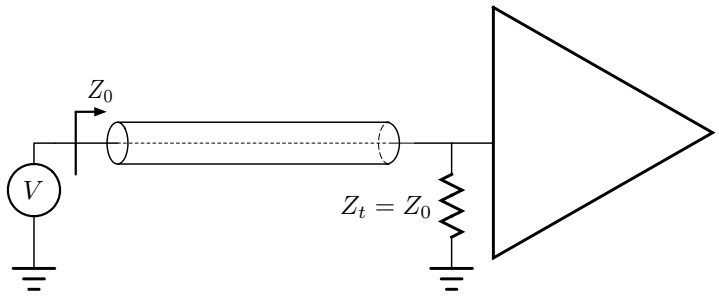


Figure 2.28: Use of a termination resistor to achieve impedance matching.

a high input impedance, a termination resistor can be installed as a shunt at the input of the element as shown in figure 2.28 to provide a matching input impedance. In cases like this, $50\,\Omega$ termination resistors are often used since the resulting of this resistor in parallel with a high impedance input is also close to $50\,\Omega$.

2.3.4 Other common analog elements

Apart from the charge preamplifier and linear shaping amplifier, there are a few other elements of the detection chain that are usually analog in nature, such as attenuators and signal splitters. This section gives a brief insight into these devices.

Attenuators

On occasions it's necessary to reduce the signal amplitude to levels acceptable for a processing module, despite the disadvantages in terms of noise. The simplest method is to use a classic voltage divider, but this does not provide symmetric impedance matching, which is important in fast pulse systems as explained in section 2.3.3. In this case, a T-section attenuator, represented in figure 2.29, can be used.

With this circuit it's possible to simultaneously achieve the desired voltage attenuation and keep the input and output R_0 impedances, as long as

$$R_1 = R_0 \frac{\alpha - 1}{\alpha + 1} \qquad R_2 = R_0 \frac{2\alpha}{\alpha^2 - 1} \tag{2.36}$$

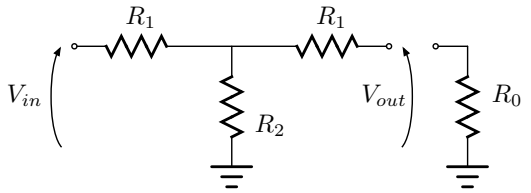


Figure 2.29: T-section attenuator circuit.

where α is the attenuation factor V_{in}/V_{out}. For example, if $R_0 = 50\,\Omega$ and we need to divide the voltage by 2 ($\alpha = 2$), the circuit with $R_1 = 16.66\,\Omega$ and $R_2 = 66.66\,\Omega$ would preserve impedance matching and provide the desired attenuation.

Signal splitting

Similarly, sometimes it's necessary to carry the same signal to more than one place. If signal reflection can be an issue, using a simple T divider isn't enough since impedances would be unmatched; for example, if a $50\,\Omega$ coaxial cable is being used to transport the signal to two modules with $50\,\Omega$ input impedance each using and a T, the two systems would be seen in parallel, their combined inputs would be seen as a $25\,\Omega$ input impedance and a reflection would occur. In this case, a splitter circuit that keeps the impedance matching should be used to avoid problems.

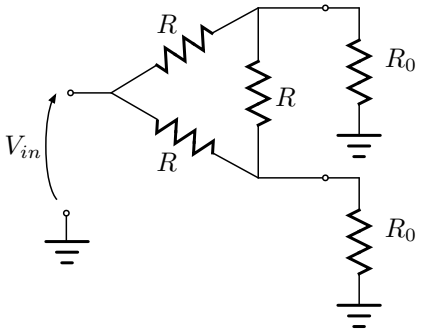


Figure 2.30: 1 input, 2 outputs triangle splitter circuit.

Splitter circuits have one input and two or more outputs. The circuit represented in figure 2.30 splits the signal between its two outputs keeping impedances matched if adequate R values are chosen. For example, if $R_0 = 50\,\Omega$ a value of $R = 50\,\Omega$ must be used. Naturally, the signal at the two outputs is half the signal at the input, since power is divided and also used by the splitter.

2.3.5 Pulse characterization

Previous sections were centered in the process of radiation detection, charge production and collection, and the generation of a signal containing information about the energy deposited with the appropriate characteristics for its transmission and further processing, registration and analysis, as shown in figure 2.31. However, depending on the application and the information of interest (deposited energy per event, event rate or timing, for example) the necessary pulse type and characteristics can vary. This section summarizes the types of pulses usually handled in a detection chain and their characteristics, in order to later present the different types of modules that comprise the rest of the system depending on this signal and the type of results to be generated.

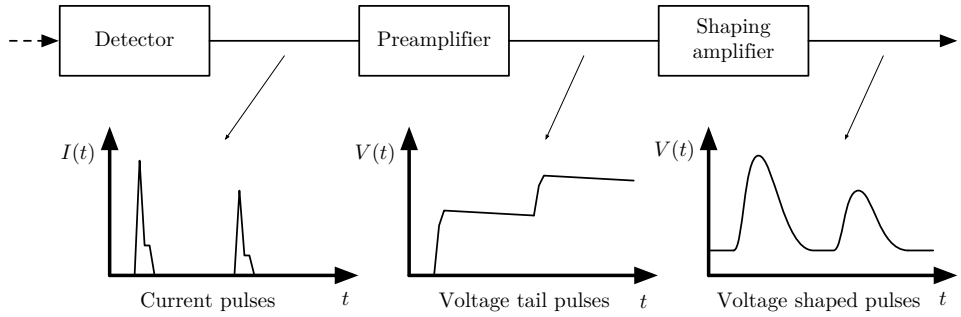


Figure 2.31: Main linear pulse types produced in the detection chain.

Two main types of pulses are distinguished, linear and logic, which can be positive or negative. Linear pulses, as seen in previous sections, are pulses that carry information in their amplitude and sometimes in their shape, and are used when parameters such as particle energy, type, charge deposit process and collection time, are under study. Logic pulses on the other hand are of fixed amplitude, and are used when the measurement is focused in the presence or absence of an event, when counting events, or when precise timing is involved. Logic pulses are usually derived from linear pulses (for example, a logic pulse may be generated when the amplitude of a linear pulse exceeds a certain threshold).

For pulse characterization, some important parameters are:

- Rise time. The time elapsed between the pulse reaching 10% and 90% of its maximum value.
- Fall time. Just like the rise time, time elapsed between the 90% and 10% of the maximum pulse amplitude.

- Full Width at Half Maximum (FWHM). This is the time elapsed between the points when the pulse is at 50% of its maximum value during the rising and falling slopes, and is a measure of overall pulse width.

In summary:

Fast linear pulses are those produced by charge collection circuits with a time constant similar to the charge collection time of the detector. The shape of the rise slope of these pulses is related to the charge generation process, as explained in section 2.2.2, and their exponential fall is very fast, so these pulses are very short, usually in the µs range or less, and usually measured using their FWHM. Because of this, the pulse has a broader frequency spectrum, is more affected by noise and consequently has lower energy resolution than tail pulses; but this same characteristics make it more suitable when timing or achievable counting rate is more important.

Linear tail pulses on the other hand are produced when the charge collection circuit, such as the charge amplifier studied in section 2.3.1, has a time constant much larger than the charge collection time to ensure all charge is collected. In this case, the rise time of the pulse is very fast, determined by the collection time (see figure 2.15) and the fall time is much longer (in the 50 to 100 µs range), hence the *tail* denomination. Amplitude will be much higher than in a fast pulse, and its value proportional to the total charge deposited in the detector.

Shaped linear pulses are tail pulses shaped to shorten their duration, adapt their amplitude to the following stages, reduce their bandwidth in order to obtain a better signal to noise ratio and make the maximum amplitude easier to measure, for example making them Gaussian shaped. The rise time is longer (and consequently they peak later) and the fall time is much shorter than in a tail pulse, depending on the shaping stages. The FWHM is a frequently used parameter, specially when pulses are Gaussian shaped, and their duration is usually of a few µs.

Logic pulses, finally, are most used when high counting rate or precise timing is needed. In this applications the pulse duration, shape or amplitude are unimportant, and having a short rise time is the key since it marks the moment an event has happened. For this reason logic pulses are usually square, with a very fast rise time (a few nanoseconds), and their width is usually not very important. When this type of pulses is used, reflection in the cabling is an important issue as explained in section 2.3.3, so impedance matching must be observed between the cabling and the extremes.

2.3.6 Pulse measurement

We've established the types of pulses produced in detection chains when the detector operates in pulse mode, and their main characteristics. This section provides an overview of the typical applications using the information provided by these pulses. It can be just counting of radiation events that fulfill some specific criteria, the analysis of the distribution of energies deposited in the detector by each event through Pulse Height Analysis (PHA), the study of timing on the arrival of nuclear events, or even deeper study on the way energy is deposited on the detector or the type of radiation involved through Pulse Shape Analysis (PSA).

Pulse counting

The simplest application of a detection chain is counting events that fulfill certain criteria, i.e. particles detected with a minimum level of energy. The criteria can be simple, as in this example, or more complex, in which case it could involve more elaborate system components or even several signals; but in most cases it involves at least the generation of a logic pulse derived from a linear pulse, and a counting module. Also, depending on the criteria, the shaping of the linear pulse needs to be more or less precise.

The fundamental module of counting systems, apart from counters themselves, are discriminator modules. An *integral discriminator* is a module that generates a pulse whenever a linear pulse of a minimum height is detected, and ignores the rest. This is often used as a *trigger* circuit, so that events under study are only those over a certain energy threshold and low level spurious signals and noise are filtered out.

On the other hand, a *differential discriminator* selects pulses that fall between two voltage values, that is, events with a specific range of energies. In spectroscopy, a range of energies is usually called a *channel*, so this circuit in combination with a counter is typically used to build so-called Single Channel Analyzers (SCAs), which count events within a specific energy range defined by a Lower-Level Discriminator (LLD) and a Upper-Level Discriminator (ULD). This, for example, allows the counting of a specific types of radiation, or events where all the energy of an incident radiation has been deposited. Figure 2.32 illustrates the diagram of an event counter and an SCA.

These discrimination modules, however, can have other uses; for example, they can be used to generate *gating* signals to control a *linear gate*, a device that works as an on/off switch for linear pulses, that opens and closes depending on a gate

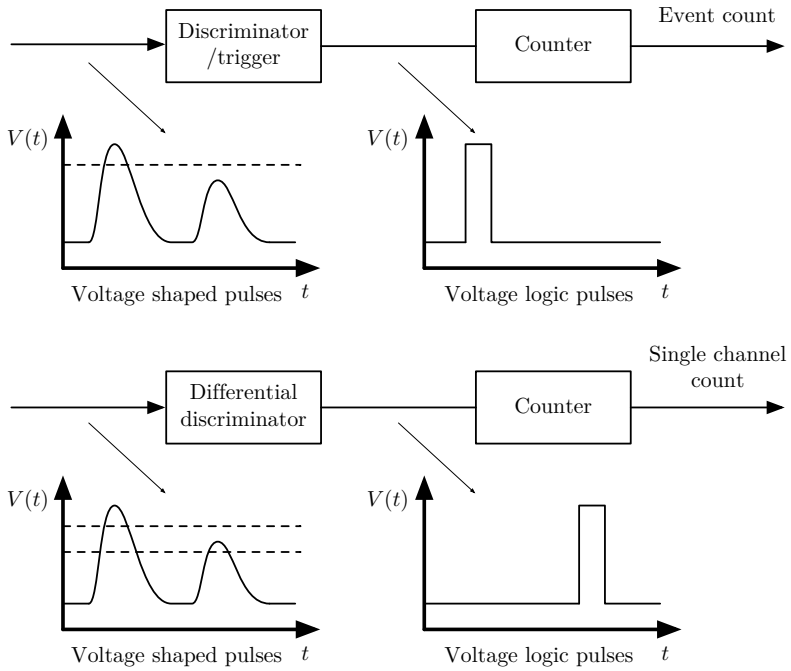


Figure 2.32: Diagram of an event counter (top) and a Single Channel Analyzer (bottom).

control input, allowing that a module connected at its output only processes certain input signals. Also, discriminators can be combined, for example to generate a pulse when a condition is met by more than one input signal (possibly, from different detectors), forming a *coincidence* module. Of course, counting of these coincidences can also be performed.

Pulse height analysis

In nuclear spectroscopy, the goal is obtaining and studying the *energy spectrum* of radiation, which corresponds to the distribution of amplitudes of the linear pulses generated by the detection chain. PHA is used to obtain this information.

To achieve this, the detection system must be able to discriminate the events corresponding to several channels (e.g. ranges of energy) and count how many events occur at each of them, thus obtaining a Multi Channel Analyzer (MCA). This information can then be used to represent an histogram of the energy distribution. Naturally, in this type of systems the role of the linear amplifier is

key and the shaping mechanism must present high resolution and high linearity between the charge deposited in the detector and the pulse amplitude, taking into account problems that can arise that are detrimental on the resolution, as pointed out in section 2.3.2.

To build a MCA, one possibility is to use several discriminators and counters, one for each monitored channel, and feed the detector signal to all of them in a similar fashion to the SCA mentioned in the previous section. This approach however is valid only if the number of channels is not too high, since it means to multiply the electronics by the number of channels to be monitored, and signal distribution and gathering of channel counts can also be complicated. A better choice may be to use a *peak detector* circuit and an ADC to sample the maximum value of each pulse and store it in an intermediate memory. After that, the values sampled over a period of time can be used directly or organized into value ranges (e.g. channels) to obtain an energy histogram. Even though this approach poses other challenges, it provides much more flexibility and allows for a much simpler electronics, while giving the user the possibility of using a high number of channels, potentially limited only by the ADC resolution.

Pulse timing

In timing applications, the measure of interest is the precise instant radiation interacts with the detector, and the electronics module that carries out this function is called a *trigger*. The accuracy of this timing will depend greatly on several factors. To begin with, statistical variations in the deposited charge will cause fluctuations in the signal produced by the detector, leading to different charge collection profiles. Second, variations can arise from the characteristics of the pulse generated by the electronics. For example, if the trigger electronics detects when a certain voltage level is reached (*leading edge triggering*), noise superimposed in the pulse signal can cause a *time jitter*. Also, if the pulses generated can be of different amplitudes, the tripping point will be reached at different moments in time since the beginning of the pulse causing a *time slewing* or *amplitude walk*. Both effects will result in a loss of resolution of the timing measure. Many approaches to the implementation of the trigger circuitry have been explored to minimize these effects, and are explained in other works ([1, p. 659-664]).

Timing applications frequently make use of some kind of clock circuitry in order to store a *time stamp* of the precise moment an event occurs. Together with the advent of digital processing and the capability to store large amounts of data about a pulse (or even the pulse shape), easier timing analysis can be done offline instead of in real time, and in some cases allow the correction of some of

the aforementioned issues, like amplitude walk. Furthermore, data from multiple detectors can be easily correlated and coincidence and timing analysis can be performed.

Pulse shape analysis

Even though the information of interest in a pulse is mainly its amplitude and time of arrival, sometimes part of the shape of the pulse can be of interest.

Differences in the charge deposit and collection process of nuclear events can lead to differences in shape of the current pulse produced in the detector. This can lead to variations in the shape of the leading edge of the tail pulse generated in the preamplifier, which in turn can be reflected in the leading edge of a fast linear pulse if the shaping time constant is short. The techniques that allow the distinction and classification of pulses according to the shape of the pulse (or, very frequently, its leading edge) is what we call Pulse Shape Analysis (PSA).

Discrimination based on pulse shape has many applications [1, p. 679], like discrimination of different particle or radiation types, and several electronic methods can be used to provide this functionality based on different criteria. It is, however, the development of high-speed ADCs and digital pulse processing techniques that opened new possibilities to PSA, since they allow much finer analysis of captured pulses either in real time or offline and the application of advanced computer-based discrimination algorithms, like those based on neural networks or machine learning.

2.3.7 Pulse processing units

After describing the types of pulse signals that are produced and can be found in a detection chain in section 2.3.5 and briefly introducing some of the pulse analysis possibilities in section 2.3.6, it is apparent that a detection and processing chain can be built around a few common and well defined pulse processing building blocks that, in smart combination, can enable the user to perform nuclear radiation analysis from the most simple to the most sophisticated nature.

Some of these blocks have been explained in some detail, like the preamplifier in section 2.3.1 or the linear pulse shaping and amplification blocks in section 2.3.2, and others have been briefly described in section 2.3.6. Table 2.1 summarizes the processing blocks that are commonly found in typical detection chains, and the type of signals they usually accept and generate (based on [1, p. 609]).

Module name	Input	Output
Preamplifier	Charge/current	Linear tail pulse
Linear shaping amplifier	Linear tail pulse	Shaped linear pulse
Stretcher	Fast linear pulse	Conventional linear pulse
Sum/difference	Shaped linear pulses	Sum/difference shaped linear pulse
Delay	Fast/shaped linear pulse	Same pulse, delayed by a fixed time
Linear gate	Shaped linear pulse + logic gating pulse	Same pulse if gating pulse is active for its duration
Integral discriminator/Amplitude trigger	Shaped linear pulse	Logic pulse if above threshold
Differential discriminator (SCA)	Shaped linear pulse	Logic pulse if maximum amplitude between lower and upper limits
Time Pick-Off/Trigger	Fast/shaped linear pulse	Logic pulse synchronized with trigger condition
Time to amplitude converter	Start and stop logic pulses	Shaped pulse with amplitude proportional to time difference
Coincidence	Two or more logic pulses	Logic pulse if all input pulses active during an interval
Anticoincidence	Two logic pulses	Logic pulse if one input is active and not the other during an interval
Scaler	Logic pulses	Logic pulse every N input pulses

Table 2.1: Common pulse processing blocks.

2.3.8 Digital detection chains

Some parts of the detection chain such as the preamplifier must necessarily be based on analog circuits and discrete components, and for decades analog technology was the only choice for all other functional blocks except for some logic, counters, registers and such. However, development of computing, high speed and resolution ADCs and DSPs opened the possibility of performing much of the pulse processing in the digital realm offering a multitude of advantages over traditional analog processing, with only a few drawbacks. Depending on the speed and accuracy of the ADC, the pulse can be digitized at different stages of the detection chain; for example, a fast ADC can sample the tail pulse generated by a preamplifier and preserve the shape of the leading edge, effectively registering

information about the charge generation and collection process and allowing PSA.

Pulse shaping can be performed digitally through digital signal processing. This not only allows the implementation of shaping methods equivalent to those used in analog shaping, but also other methods that are difficult or impossible to achieve using analog circuits. The digital implementation also provides the ability to adjust gain and shaping parameters without the need to make any physical changes to the electronics, which provides high flexibility. Moreover, since digital data is infinitely reusable, different parameters or more than one shaping method can be used and compared, or selected depending on the operating conditions.

Pulse processing once the pulse has been digitized is also considerably easier. Counting, pulse height determination and analysis or even shape analysis if the ADC is fast enough as mentioned above can all be performed digitally, not only in real time but also off-line. Operations such as pile-up rejection or baseline restoration are easier to achieve, too. **Processing units** just like those described along section 2.3.6 can be implemented digitally, and often in different ways depending on the application and the processing requisites (real time or offline, complexity, etc.). This can be achieved in a variety of ways, from software run in a microprocessor or a DSP, to a programmable logic device like an FPGA, or a combination of both.

Finally, digital processing is intrinsically stable, since once the pulse has been digitized the data and the processing algorithms don't change with environmental variations or component aging, just like their analog counterparts.

Limitations of digital pulse processing do exist, however, in part due to the discrete nature of analog to digital conversion, and in part because digital processing frequently involves a series of steps (e.g. arithmetic operations) that introduce delays in the generation of results.

Given that the process of digitizing the input signal involves two steps, sampling and quantization, the ADC is characterized by a sampling rate or frequency, f_s, and a bit depth, or amplitude resolution. Sampling involves observing the value of input signal $x(t)$ every T_s seconds, where $T_s = 1/f_s$ is the sampling period, resulting in a sequence of numbers $x[n]$ where n is an index in this sequence. These numbers can still take any value, so another step is required to represent the signal values digitally with a finite number of bits, which is called quantization. The final result is a sequence of binary numbers of a fixed length defined by the ADC's bit depth, at a rate defined by the sample rate, representing the input signal. It may seem that to preserve the input signal characteristics the highest possible sample rate and the largest possible bit depth should always be used, but that is actually not true, and is an approach that would generate vast amounts of

mostly useless data.

It is obvious that the sampling rate must be high enough so that rapid variations of the signal are not missed; but according to Nyquist's sampling theorem, an analog signal composed of frequencies up to a maximum frequency f_{max} can be completely reconstructed by sampling it at $f_s = 2 \cdot f_{max}$, so higher sampling frequencies are simply not necessary. It must be noted, however, that if the input signal has frequency components that don't comply with the $f_{max} \leq 2 \cdot f_s$ condition, a phenomenon called aliasing occurs that generates an error in the signal reconstruction due to the appearance of anomalous lower frequency signal components. Because of this, it is often necessary to place a low-pass antialiasing filter before the ADC that filters out the unnecessary frequencies. Finally, sampling rate not only impacts the fidelity with which the signal is registered but also limits the resolution of a system for timing applications, since changes in the signal between two samples go unnoticed until the second sample is taken and it's not possible to directly determine the precise moment of variation between these two samples.

Bit depth on the other hand determines the quantization levels of the signal, the minimum voltage differences in the signal that the system can handle, which in an analog system is infinitely small. This introduces a *quantization error* because all signal values in a specific range get converted to the same digital value. The quantization error (i.e. difference between the real signal value and the encoded value) is random, but is always between -0.5 and 0.5 times the voltage quantum (thus, smaller as bit depth gets larger), and its mean value with respect to the quantum value can be calculated.

$$\langle \epsilon^2 \rangle = \int_{-0.5}^{0.5} x^2 dx = \frac{1}{12} \qquad \rightarrow \qquad \epsilon = \frac{1}{\sqrt{12}} \tag{2.37}$$

For example, a 10-bit ADC divides the full range of input amplitude it handles in 2^{10} (1024) levels, that is, it can handle variations of the signal of roughly 0.1% of the full scale (i.e. $\approx 1mV$ for a 10 V input range). It must be taken into account however that using a greater bit depth to reduce this quantization error doesn't always make sense, since the signal to noise ratio in the signal limits the significance of the Least Significant Bits (LSBs) of the samples, and once noise is larger than the quantization error larger bit depths provide no benefit. The concept of Equivalent Number of Bits (ENOB) in analog to digital conversion determines the number of bits in the sample that *really make sense*, taking into account that variations in the signal below a certain threshold cannot be distinguished from noise.

$$ENOB = N - \log_2\left(\frac{RMS_{noise}}{\epsilon}\right) \tag{2.38}$$

Where N is the bit depth. Other imperfections in the conversion arise as non-linearities, and are described in detail in other works, such as in [1, p. 649].

2.4 Nuclear instrumentation module standards

Radiation detection and measurement of most kinds usually involves the generation and processing of pulses in the ways described in this chapter, and almost any task is ultimately performed by combination of functional processing modules such as those described in section 2.3.6. It is for this reason that modules providing each specific functionality started being manufactured and sold, so that assembling an instrument for a certain type of application could be achieved by combining and adjusting a bunch of these modules connected to each other, typically mounted in some type of housing, instead of having to design and produce specific ad-hoc electronic circuits for every application.

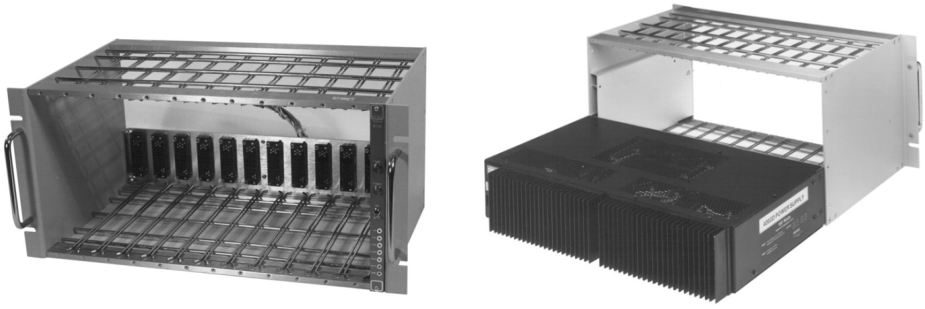
In order to allow the interconnection between modules of different manufacturers, instrument standards were proposed, so that users could reliably build systems based upon these modules and swap modules if needed. These instrument standards define at minimum:

- The basic dimensions for the housing and the modules.
- The way power is delivered to the modules. Typically the housing includes the necessary power supplies, and a standardized connector delivers the required voltage to each module apart from providing mechanical harnessing.
- The polarity and amplitude of the input and output signals, linear and logic.

This section provides an overview of the most important instrument standards.

2.4.1 NIM

The Nuclear Instrumentation Module (NIM) standard was first defined by the U.S. Atomic Energy Commission in 1968-1969, and was revised overtime until the currently published revision was released in 1990 [6]. The standard define a



(a) (b)

Figure 2.33: ORTEC NIM bin with attached power supply. Picture from [7].

system consisting of standard modular elements, a housing (referred to as *bin*) and the associated power supplies.

The housing, as the one depicted in figure 2.33, is designed to fit in a 19-inch rack and is divided in 12 module positions with a width of 34.4 mm, although modules of multiples of this size are allowed. There are two sizes for the height of the bin and the modules, 133 mm and 222 mm, the latter being the more common. Each module location has a 42-pin connector where each module is plugged in, intended mainly for mechanical aligning and power delivery. $\pm 12V$ and $\pm 24V$ are provided by the bin, although some also provide $\pm 6V$. These and other functions are illustrated in figure 2.34c.

It can be seen in this figure that the connector specification does not define the function of many of the pins with the notable exception of power supplies. Some logic can be performed through this connector pins, but in the NIM standard linear and logic signals are supposed to be transmitted among modules by means of coaxial cables to jacks either in the front or the back of each module, using BNC or SHV connectors for regular and high voltage signals respectively.

Regarding this matter, the standard defines three voltage ranges for shaped linear pulses; 0-1V, 0-10V and 0-100V, and care must be taken to interconnect only modules that support the same signal ranges.

For logic pulses, there is a distinction between standard and fast logic pulses. Normal pulses use voltage-based logic, with levels as indicated in table 2.2, while fast logic pulses use current-based logic as indicated in 2.3 and require $50\,\Omega$ impedances.

It is pretty obvious that this standard was not created with digital systems and computers in mind, since there's no specification for some kind of digital data

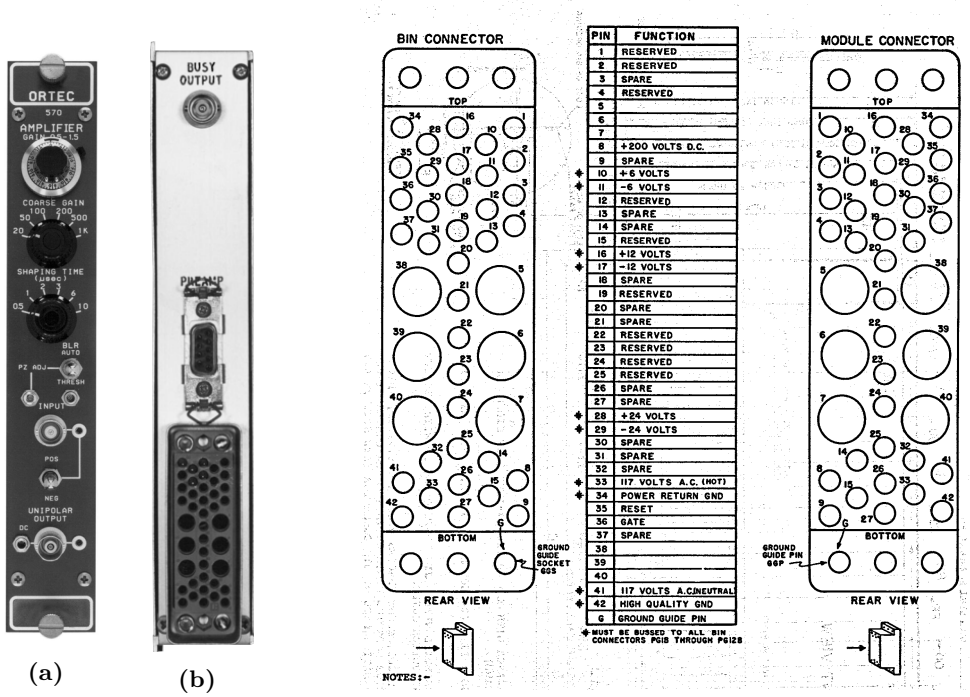


Figure 2.34: ORTEC NIM module and NIM connector diagram. Picture from [7], connector diagram from [6].

Logic Level	Input	Output
1	+4 to +12 V	+3 to +12V
0	+1 to -2 V	+1.5 to -2V

Table 2.2: NIM normal pulse levels.

bus signals between modules, neither any kind of data backplane. This led to the development of the CAMAC standard.

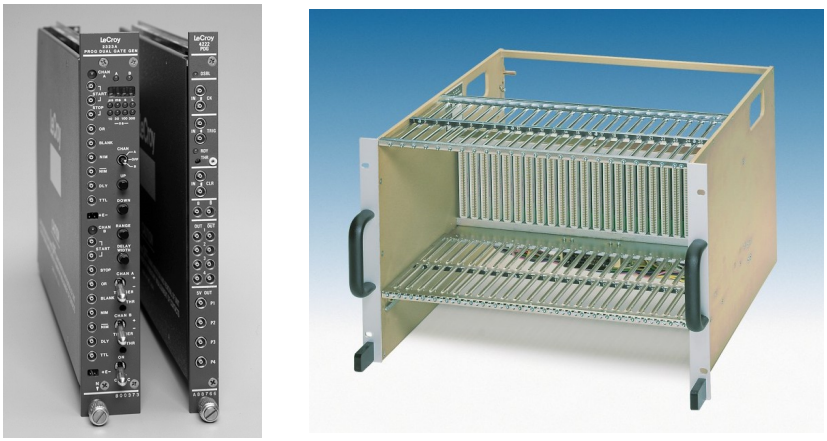
2.4.2 CAMAC

The Computer Automated Measurement and Control (CAMAC) standard IEEE 583-1975 was defined by the European Standards On Nuclear Electronics (ES-

Logic Level	Input	Output
1	-14 to -18 mA	-12 to -36 mA
0	-1 to +1 mA	-4 to 20 mA

Table 2.3: NIM fast pulse levels.

ONE) Committee and revised and reaffirmed in later versions of the specification [8]. Unlike NIM, it is geared towards digital systems and interfacing to computers, featuring a data bus referred to as *dataway* in the module connector that allows modules to exchange digital data.



(a) **(b)**

Figure 2.35: CAMAC module and crate examples. Pictures from [9] and [10].

The standard defines a housing referred to as *crate*, that just like NIM's *bin* fits in a 19-inch rack (figure 2.35b). It is subdivided in 25 17.2 mm wide slots (vs. 12 in the NIM standard), although modules using two or three times the width, as the one shown in figure 2.35a, are supported. Connection of modules to the crate is achieved through a PCB edge connector with 86 contacts as shown in figure 2.36 instead of the bulkier NIM connector. This connector provides power to the module but also the control and data lines dedicated to the aforementioned dataway.

There is a special crate position at the rightmost end, called the *control station*, where a special module named *crate controller* must be installed. This module provides the control and arbitration logic that allows all modules to exchange

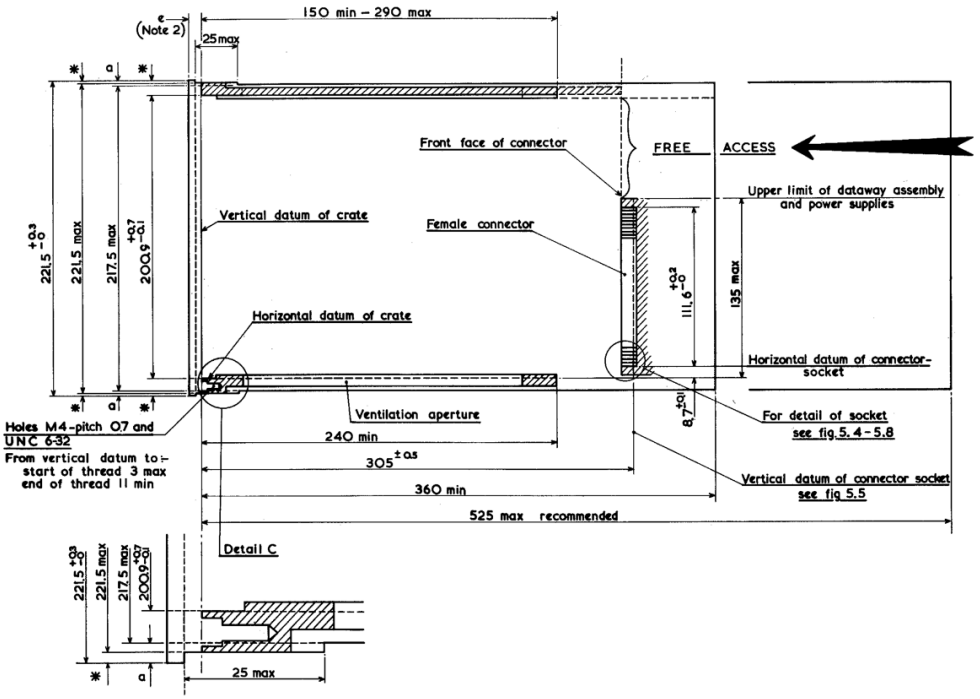


Figure 2.36: Diagram of a CAMAC module, from [8].

data, and the interface between the crate and other equipment (computers, other crates, etc.), so it is mandatory unless there's only one instrumentation module.

The signals on the backplane are different for the crate controller and the other 24 stations. The stations are interconnected by a write and a read bus of 24 lines each, and each also has a selection (N) and *look-at-me* (L) line. The control station on the other hand has lines for the 24 N and 24 L signals, instead of the read and write buses. All the data transfers among modules are arbitrated by the crate controller, using these lines and the rest of the common module subadressing, strobe, function, control flow and status signals, and speeds of up to 1 million 24-bit words per second are achievable [11]. Figure 2.37 illustrates both the dataway wiring and control station position. Naturally, if the crate controller needs to read or write data, possibly because it is connected to another crate or an external computer it must access the read and write buses, so typically the crate controller takes not one but the two rightmost slots of the crate. Additionally, the backplane provides $\pm 6V$ and $\pm 24V$ power supplies and optionally, $\pm 12V$.

Linear signals on the other hand are still transmitted using coaxial cables, connected to the front or back of each module just like in NIM modules. For this,

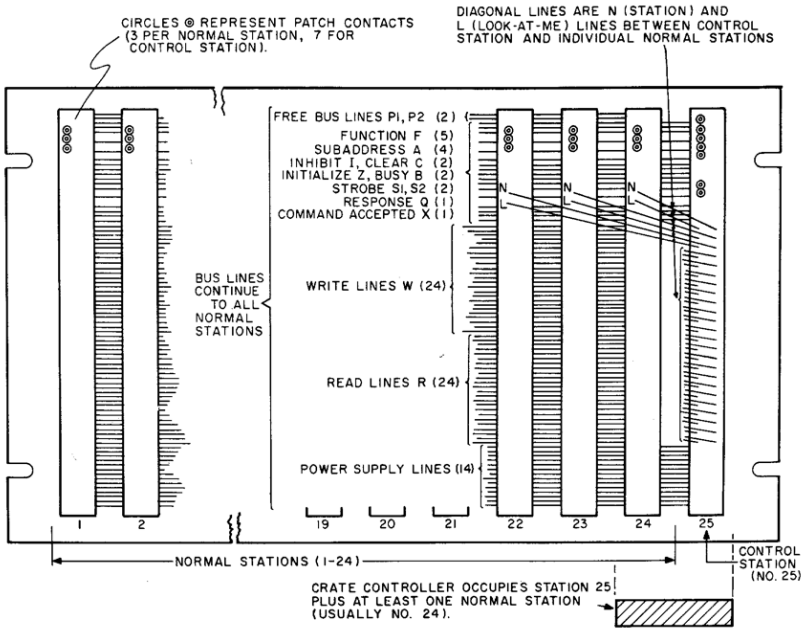


Figure 2.37: CAMAC backplane wiring, from [8].

a 50CM type coaxial connector [8, fig. 13] is recommended.

Finally, for applications that require more than one crate (because the number of modules is high or other logistic reasons), CAMAC specifies crate interconnection methods, as shown in figure 2.38. Crate controllers can be connected to each other either in parallel or serially, using the *branch highway* or *serial highway* specifications. The former requires sharing a large number of electrical lines and is faster, but costs are higher and is typically limited to 7 crates. The latter uses a serial chain with two wire pairs, so is slower but allows for up to 62 crates and at a much lower cost.

2.4.3 VMEbus and VXI

Versa Module Euro bus (VMEbus) is not an instrumentation specific standard as NIM or CAMAC, but a computer bus developed by Motorola for the 68000 microprocessors in the early 80's, and later standardized by the International Electrotechnical Comission (IEC) as ANSI/IEEE 1014-1987. Its purpose was to establish an industry-standard system for computer modules for telecommunications, industry and military applications that could be installed in a rack and

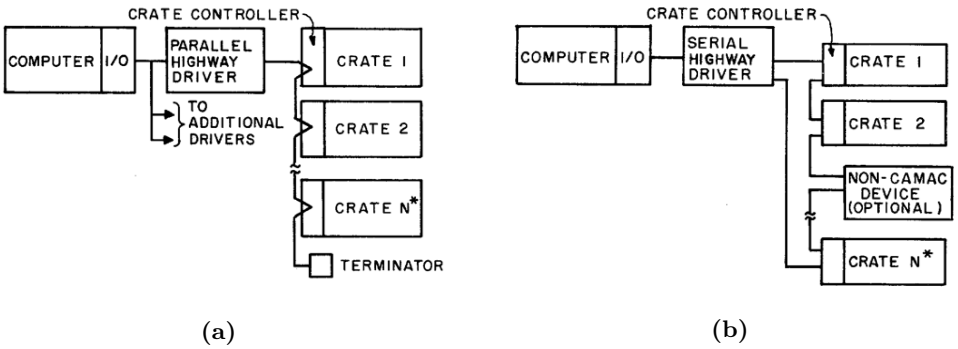


Figure 2.38: CAMAC crate interconnection models, parallel (a) and serial (b). Diagrams from [8].

interconnected using a high-speed backplane. Originally it provided a 16-bit data bus for data transmission, that later evolved to 32 and 64-bit versions, and is capable of providing high speeds up to 40 MB/s. However, it lacked instrumentation features present in other digital systems such as CAMAC, so the VME eXtensions for Instrumentation (VXI) standard emerged from VMEbus in 1987, keeping its capabilities and expanding them by defining among other things additional bus lines for timing and triggering as well as requirements for message-based communication and chassis expansion.

The VMEbus standard defines *subracks* that fit in standard 19-inch racks, holding up to 21 VMEbus modules. Modules are Eurocard-sized, and can be single (3U) or double height (6U), although 9U are sometimes used too. They are plugged into a backplane in the subrack, which has two 96-pin connectors named J1 and J2. 3U boards use only the J1 connector, while 6U boards can use both, as illustrated in figures 2.39 and 2.40. The backplane provides $+5V$ and $\pm 12V$, and a $+5V$ standby supply. Other more advanced specifications such as VME64x define extra pins and additional connectors, as well as additional voltage supplies.

The backplane is composed of four buses: the data bus, the priority interrupt bus, the arbitration bus and the utility bus. Using just the J1 connector allows for 16-bit transfers over a 24-bit address space, while J2 adds extra lines to allow 32-bit data width and 32-bit addressing. VMEbus modules transfer data using the backplane through Direct Memory Access (DMA), with the modules taking the role of master or slave, and interrupts over the backplane are supported; but the backplane is controlled by a special *arbiter module* that must be installed in slot 1, the leftmost of the subrack, in a similar fashion to the CAMAC control station. This arbiter module provides bus arbitration, management of timeout errors and other utility functions such as system clock or activity monitor, as

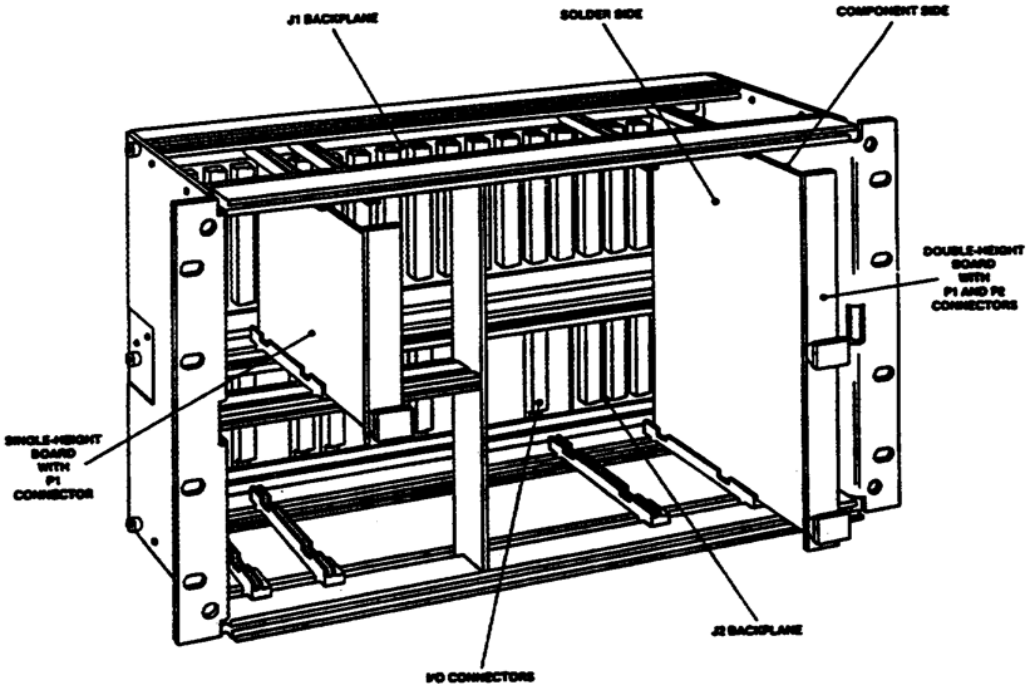
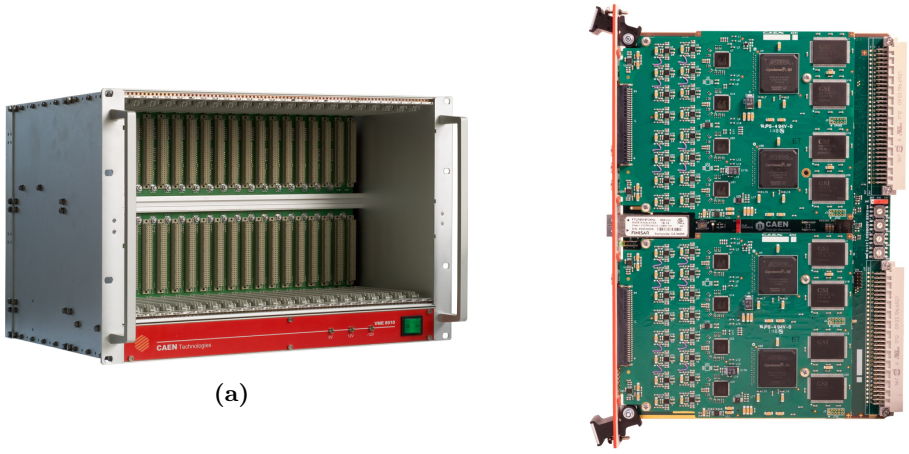


Figure 2.39: VME subrack and boards, from [12].

well as communication to external systems such a computer. It is quite common, however, to find embedded computers inside an VMEbus rack as a module for convenience and faster access to data.

VXI builds on the technology and features provided by VMEbus and adds instrumentation specific resources. A VXI system is structured in three levels; the single device (usually contained in a single module), the VXI subsystem (composed by a subrack and its modules) and the VXI system, composed of a number of interconnected subsystems. It must be noted though that a module can contain more than one device, each assigned an individual VXI address, and that VXI subracks (usually referred to as *mainframes* or crates) are limited to 13 modules. Up to 256 VXI modules are supported.

Module dimensions are the same Eurocard size, and each subsystem must include a control module, sometimes referred to as *slot 0 controller* since it must be installed in slot 0 of the chassis. The VXI specification requires automatic system configuration performed by the controller, requiring all devices to store their configuration data in four registers, specifying their type, status, memory requirements and control procedures.



(a) (b)

Figure 2.40: VME subrack and 6U card examples, from [13].

With the VMEbus bus at its core, the VXI bus adds instrumentation-specific buses and signals:

- A *Global bus*, that includes the VMEbus bus, a *power* bus, a *trigger* bus and an *analog sum* bus, dedicated to analog signals.
- A *Unique* bus that includes a *time and synchronization* bus, a *module identification* bus and a *star* bus.
- A *Private* bus, consisting of a local bus for modules inside chassis.

At the core of the global bus is the VMEbus described before. In addition, the trigger bus supports up to fourteen trigger signals, eight TTL and six ECL; the analog sum bus is used to generate analog current signals, for example for simulations; and the power bus provides power supplies for a variety of voltages; $+5V$, $-5.2V$, $-2V$, $\pm 12V$ and $\pm 24V$.

At the unique bus, the time synchronization bus provides two 10 MHz and 100 MHz clocks and a synchronization line. The star bus, which as its name implies implements a star topology interconnection between modules with the system controller at the center, minimizes the communication time among any two modules without affecting the rest of the system. Finally, a 12 line identification bus allows each module to connect directly to the system controller, thus reducing the interruption handling time.

Finally, the hierarchical VXI system is usually controlled by a PC, either connected to a VXI module or directly implemented as an embedded computer and installed in the chassis as a module, which allows much higher data rates.

2.4.4 PXI

Just like VXI appears as an instrumentation extension for VMEbus, PCI eXtensions for Instrumentation (PXI) is an instrumentation extension of Peripheral Component Interconnect (PCI). PCI was created in 1992 as a high-speed local bus for attaching hardware devices to a computer. It supports the functions of a processor bus, but is standardized and independent of the processor. It was implemented first in Intel x86 PC compatibles where it was a great success with many peripheral cards and chips being produced supporting it, and was later adopted in other architectures, such as PowerPC. The initial version of PCI was capable of transfer speeds up to 132 MB/s, with later versions doubling or even quadrupling this figure in the case of Peripheral Component Interconnect eXtended (PCI-X).

PCI started being used in industrial computers in 1995, in the form of Compact PCI (cPCI). This was an implementation of PCI signaling that used rugged Eurocard mechanical packaging and connectors and a passive backplane where cards could be plugged, typically mounted in a rack. Up to 8 devices can share a common backplane (while only up to 4 devices are supported in a Personal Computer (PC) implementation), and systems with more elements can be built using standard PCI-to-PCI bridges. 3U and 6U Eurocard sized modules are supported, and the backplane is composed of two 108-pin connectors, J1 and J2. J1 implements the PCI bus, and J2 can carry other user-defined signals or the upper 32 bits of a 64-bit PCI implementation.

Since cPCI is, therefore, a computer bus for industrial environments, it can be thought of as an alternative to VMEbus that offers lower costs because of high component availability, a consequence of its popularity in the consumer computer market, and also high flexibility in the development of software and instrumentation systems, given its compatibility with PC computers, drivers, tools and operating systems such as Microsoft Windows or Linux.

However, also like VMEbus, cPCI lacks essential instrumentation features, which is the reason PXI was introduced in 1997. Using the cPCI form factor, it adds the needed resources to build instrumentation systems while offering interoperability with existing cPCI products. PXI modules can be used in a chassis, just like CAMAC or VXI modules, but also directly attached to an embedded computer to act as a stand-alone instrument. And just like cPCI, it benefits from

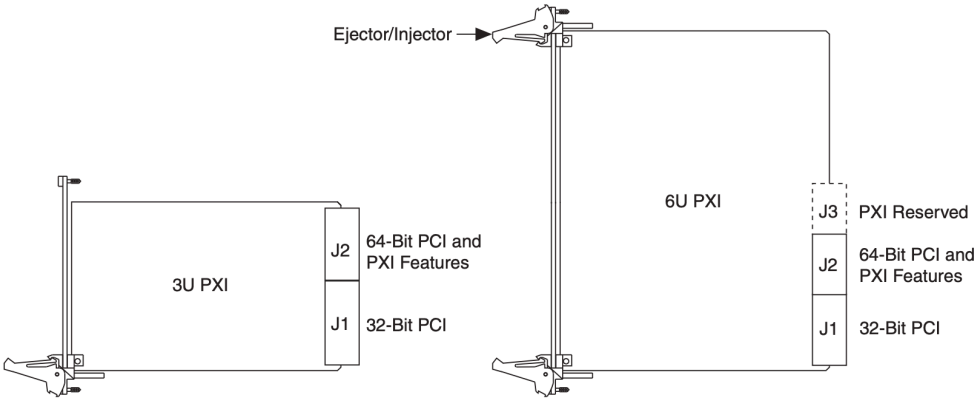


Figure 2.41: PXI boards diagram for 3U and 6U, from [14].

the wide support of PCI technology in the PC world.

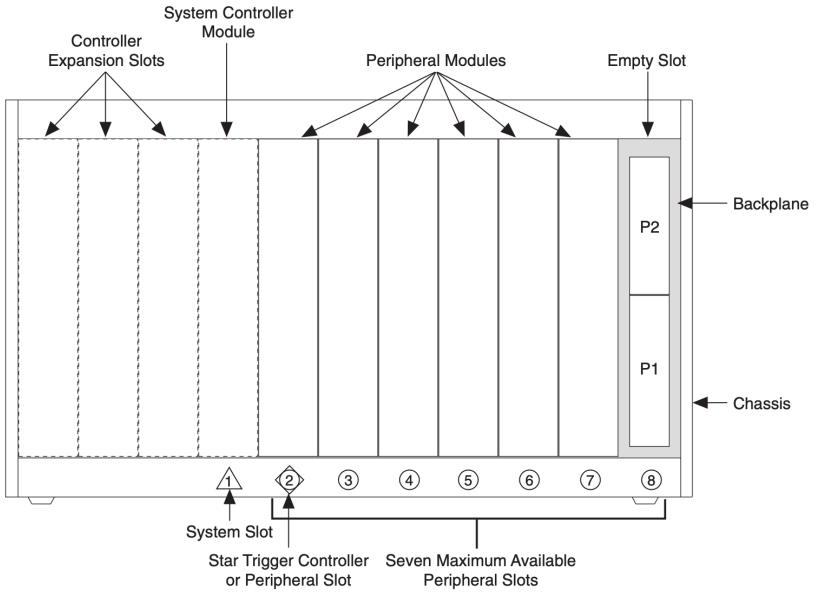


Figure 2.42: PXI 3U chassis diagram, from [14].

Mechanically, PXI adds to the cPCI form factor one more optional J3 connector in the backplane, as represented in figure 2.41. It requires the chassis to have one *system slot* where the *system controller* must be installed, thus reducing the number of possible *peripheral modules* to 7 when working at 33 MHz or

4 at 66 MHz (although PCI-to-PCI bridges can also be used for expansion). It also defines a *star trigger controller* module that can be optionally installed right next to the system controller, and several *controller expansion slots* that allow the installation of larger system controller modules or embedded computers that include the system controller functions. Figure 2.42 shows the slot distribution and connector positions in a 3U chassis.

Electrically, the core of the PXI bus is a PCI bus, with the addition of the following resources:

- Local buses for communication between modules. Each bus is composed of 13 lines, carrying both analog and digital signals.
- A 10 MHz clock line for synchronization.
- A trigger bus with 8 trigger lines.
- An optional star trigger bus for high-precision synchronization between modules without affecting the performance of the rest of the system. This feature requires the star trigger controller module.

These bus resources are implemented in the J2 connector, while J3 is reserved for future PXI features.

Figure 2.43: PXI system in a fully populated chassis, including a system controller using several slots. Image from [15].

Finally, just like in VXI, the system is usually controlled by a PC, either connected to a module or directly implemented as an embedded computer and installed in the chassis as a module, as depicted in figure 2.43.

Chapter 3

ARACNE description

> "Begin at the beginning," the King said, very gravely,
> "and go on till you come to the end: then stop."
>
> — Lewis Carroll, *Alice in Wonderland*

3.1 Building a detection chain

As outlined in chapter 2, a nuclear instrumentation system is composed of a set of well defined general elements organized in what is known as the *detection chain*, given the stage-by-stage design that transforms the signal generated by a detector into information suitable for analysis and study. Depending on the application, the chain can be more or less sophisticated, include more than one detector or use a combination of signals at some point; but in general, the chain construction is similar and the function of the involved elements is more or less standardized.

Some elements of this chain necessarily need to be built using analog electronics, while many others can be implemented digitally, an option that is preferred in most modern systems. But in all cases, the user is presented with two main options to build a system; design and produce an ad-hoc, application specific electronic system, or build a system by putting together Commercial Off-The-Shelf (COTS) modules and housings, possibly compliant with some standard, such as those described in section 2.4.

If the detection chain is **built in-house**, the user typically needs to design, manufacture and assemble all of the electronics corresponding to each of the el-

ements of the chain, except maybe a few that may be readily available and not worth the effort; for example, the specific detector electronics and probably the preamplifier and amplifier.

This approach includes development both in the hardware and software fronts; circuit design and simulation, electronics schematic capture, Printed Circuit Board (PCB) development, production and testing, low and high level software design, and so forth. The result is a highly specialized system, optimally tailored to the application and quite possibly very resource efficient in terms of power and footprint. However, it requires good knowledge in a wide array of technologies, a lot of work and quite possibly a lot of time to finally deliver a successful prototype, which ultimately leads to a high development cost. Moreover, this system will be hardly reusable for a different application since hardware is static by nature, and any change in the application will probably mean to go back to the drawing board.

A lot of effort and time can be saved and a lot of risk avoided if the system is built **using standard instrumentation modules**. The user just needs to determine what modules, cabling and housings are needed, purchase them and do the appropriate interconnections and configuration, without any knowledge on electronic design or manufacturing. Modules can be reconfigured, reorganized and replaced. And they are inherently reusable for other applications, so they are ideal for lab work. But this approach has, of course, a few drawbacks.

The first of them is prize. Modular systems are typically very expensive, with modules frequently costing thousands of euros and systems that easily exceed tens of thousands, which may not be feasible for many institutions and small laboratories. Also, depending on the application and the capabilities of the modules more than one of each may be needed, making the problem worse (i.e. in multi-detector experiments). Second, even though modules are typically designed with flexibility in mind, their features are fixed in hardware (for example, number of inputs or possible positions in a dial); they will never be as tailored to an application as a custom solution. And at the very same time, they are usually complex and overengineered to provide a flexibility that is not always taken advantage of. This means a modular system, at the end of the day, is probably very inefficient in terms of power and footprint compared to a custom design if it's always going to be doing the same job.

With this scenario in mind, this work proposes an alternative concept for the implementation of the detection chain called Adaptable and Reconfigurable Acquisition Concept for Nuclear Electronics (ARACNE) that complements the two aforementioned approaches, tackling on the rigidness and development difficulty of fully custom electronic systems and the cost and inefficiency of module-based

systems. It is a platform comprised of hardware, IPCores, a tightly coupled embedded computer and software libraries, that is adaptable at several levels to different applications, reconfigurable, reusable and efficient.

3.2 Concept

The fundamental premises on which the ARACNE idea is based are essentially three:

1. The signals generated by the detector, preamplifier and amplifier of a detection chain can typically be classified into a few types, as described in section 2.3.5, and a common hardware for basic analog signal conditioning and digitalization that contemplates many scenarios is feasible.
2. If signals are digital, standard instrumentation modules can be implemented as synthesizable hardware, or a combination of this with software running on a Single-Board Computer (SBC) tightly coupled to it.
3. Modern ADCs, programmable logic devices such as FPGAs, and SoC based SBCs make possible the implementation of complex, multi-channel detection chains in a very resource-efficient and cost-effective package.

As such, ARACNE is proposed as a hybrid platform composed of hardware, synthesizable hardware, an embedded SBC and software libraries, that can be used with a variety of detector types and configurations to implement different applications, and whose functionality can be radically changed without making any modifications to the physical hardware.

There are several parts of this approach that are new. In the first place, the usage of programmable logic in data acquisition systems already became popular some time ago, but its usage doesn't automatically make these systems adaptable or reusable. In many cases an FPGA is used in a design that is tailored specifically to a detector or scenario, and neither the front-end electronics or the IPCore are built with reusability in mind. And also, in many cases the FPGA capabilities for signal processing are left unused, since the detector signals are transformed into logic pulses that the FPGA, in turn, deals with. This is the case of [16] or [17], that present an FPGA based system specific for neutron monitors, or [18] and [19] that present a proposal for a muon telescope. Another interesting case is [20][21], where a COTS development board is used and the IPCore is outsourced to an external company to build a replacement for an older module [22], or increase its

functionality [23]. In this sense, the philosophy of the ARACNE concept is to provide a hardware platform with a front end electronics as generic as possible to be used with different detectors and configurations, and digitize the analog signals so that processing is performed on the FPGA, both leveraging its capabilities and making the use of external analog or digital processing modules unnecessary.

There are other cases, however, where the FPGA has been truly used to process the digitized signal. Some of these send the raw digitized data to a PC for processing, and others are capable of data reduction by detecting and processing pulses and sending to the PC just the captured pulses or results from its processing. These are, however, high-performance systems that require the use of high performance FPGA devices, very high speed data links such as PCI Express (PCIe) and a dedicated PC. This is the case of [24]/[25], which uses for its instrument a commercial solution consisting of a digitizer board with a reprogrammable Virtex5 FPGA and a custom IPCore, connected via PCIe to a PC running a Matlab application. Or [26], which uses state-of-the-art hardware and a x4 PCIe connection to a PC. These are systems where compactness or energy consumption are not important, and cost is not an issue; in the ARACNE concept, the usage of a tightly coupled SBC that makes the use of an external PC unnecessary pursuits precisely these goals. And there exist projects such as [27] that do use an embedded processor, but only to send data to an external computer. The platform in this case is based on a development board and commercial ADC modules.

Finally, even though some of these developments could be modified and somewhat adapted, none is is aimed at producing a reusable platform. The hardware is either designed for a specific experiment, or COTS hardware such as development boards or ADC modules are used. The IPCore are specifically designed for each application, with no declared intention for reusability. And software is either commercial or specifically designed for the corresponding IPCore. There are some efforts in this direction regarding synthesizable logic, such as the one described in [28], where the IPCore for a specific platform is considered to be mostly fixed, but a block interface for algorithm developers is proposed so that the user doesn't need to know all the hardware details. In the ARACNE proposal, on the other hand, the goal is to structure the design of the whole IPCore identifying hardware-related and algorithm-related blocks, and at the same time outline the structure of the software libraries that should accompany the IPCores.

3.3 High level design

An ARACNE system is composed of three main elements:

- A *hardware platform* containing an ADC, a programmable logic device such as an FPGA and an embedded computer connected to both. It must be able to accept analog input signals from detectors, convert them to digital samples and feed them to the FPGA, which in turn offers processed data to the computer for further processing.
- A set of *IPCores* that can be programmed into the FPGA to perform the digital processing functionality of the detection chain required for the application.
- *System libraries* that allow control of the hardware platform and the IPCores, in order for the user to be able to write software that makes use of the resources they provide and to implement other functionality of the detection chain not provided by the IPCores.

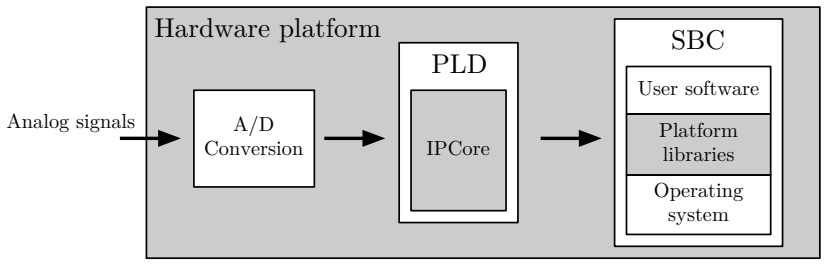


Figure 3.1: ARACNE concept diagram. Elements in grey are provided by the platform.

Figure 3.1 illustrates this approach, and an overview of the characteristics of each of these elements will be provided in the next sections, followed by a brief review of their requisites.

3.3.1 Hardware platform

The hardware platform provides the core electronics for the system, implemented in a PCB. It provides all the electronics needed for the system except for the SBC, which in principle will be a COTS device suited for the application.

As shown in figure 3.2, it is composed of the following parts:

- **An analog signal input and conditioning stage**. Signals coming from the detectors will be accepted complying with the impedance requisites of the cabling and previous devices, and will be adapted as necessary (shaped, filtered, amplified, etc.) to comply with the requisites of the digital conversion stage. Electronics for each input channel will be provided, and a certain degree of control of this stage by the SBC (i.e. enabling and disabling inputs) will exist.

- **A multichannel analog to digital conversion stage**. Analog signals will be converted to digital samples by a multichannel ADC and fed to the Programmable Logic Device (PLD) device, e.g. the FPGA. This ADC is controlled by the SBC.

- **The high-speed digital processing stage**. The digital signals are concurrently accepted and processed in real-time in the FPGA, and resulting data is made available to the SBC. In this stage, the functionality of the acquisition chain is implemented in the IPCore loaded into the FPGA. Also, circuitry necessary for the device programming is provided.

- **The SBC connection**. An appropriate connector for the SBC is provided where all the necessary signals for communication with the FPGA, ADC and other electronics are implemented, as well as power and other additional expansion signals.

- **Power supplies** for every on-board component.

- **Miscellaneous I/O**; for example, switches, jumpers and LEDs for user interaction.

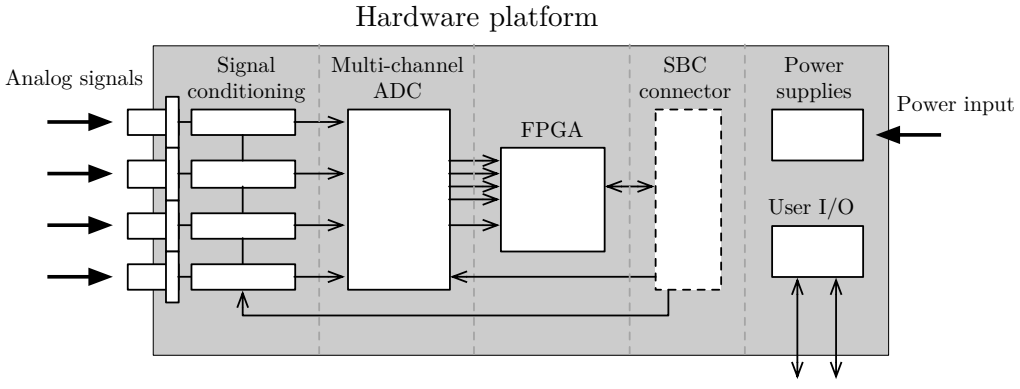


Figure 3.2: ARACNE hardware platform diagram.

The platform can also include other additional features, such as test points, test signal generation facilities or miscellaneous I/O expansion; but these are outside the scope of this specification.

The hardware platform is the only part of the proposal that is immutable once built, and thus must be designed more carefully to accommodate for the range of applications where the ARACNE system is to be used with a specific platform implementation. This does not mean, however, that there's only one possible implementation of this platform; based upon the same concept and structure, different solutions can be produced for different performance or functionality requisites, adapting each of the crucial components to the required capabilities; for example, an ADC with more channels or higher sampling rate, a higher speed FPGA or a faster SBC connected via a more capable bus can be used without changing the general design philosophy.

3.3.2 Synthesizable hardware

The IPCore loaded into the FPGA provides the functionality of the acquisition chain, and as such will be modified and reconfigured to meet the desired functionality for each application. For this reason, a complete, static design cannot be specified; but a general modular structure is proposed that should ease the process of creating new application-specific IPCores through the combination of these modules.

Two types of modules can be distinguished in the proposal:

- *Modules that depend on the hardware platform.* These modules deal with signals related to the FPGA I/O, and consequently depend on the characteristics of the hardware system components they are connected to, such as the ADC and SBC. Because of this, these modules are tailored to a specific hardware platform and must be produced in a case-by-case basis. However, once these modules are available they can be reused in all the applications where the specific hardware platform is going to be used, without modifications. These modules will produce signals that are platform-agnostic according to a predefined set of logical interfaces.

- *Generic modules that do not depend on the hardware platform.* These modules do not have any contact with the outside world and deal purely with internally defined signals and logical interfaces. As such, they can be infinitely reused and replicated in different hardware platforms as long as their functionality and interfaces are well defined and are written in plain syn-

thesizable hardware language such as Very High Speed Integrated Circuit Hardware Description Language (VHDL).

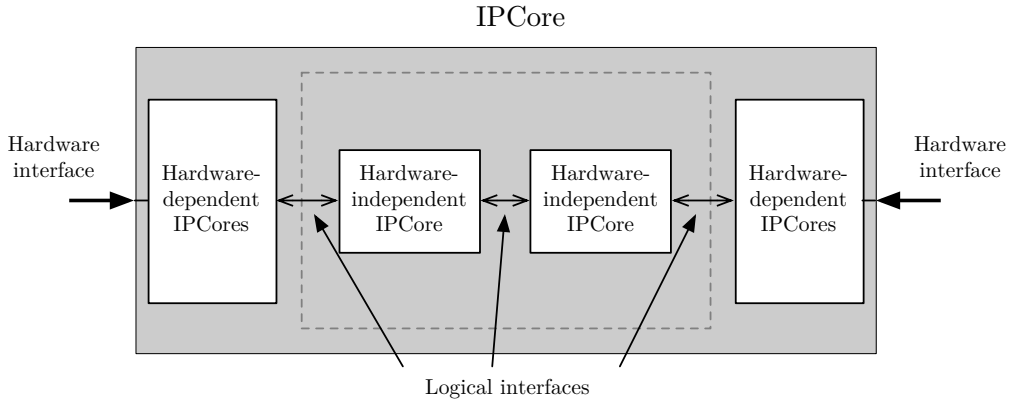


Figure 3.3: ARACNE IPCore design philosophy. Elements outside the dotted line are hardware platform dependant, while those inside it are not.

Figure 3.3 illustrates the concept and the relationship between both types of modules. It is often convenient, though, to write versions of both types of modules that are tied to a specific FPGA model, family or manufacturer to take advantage of specific resources of a device. For example, if an FPGA provides dedicated hardware Serialize/Deserialize (SERDES) modules it makes sense to create input modules based on them, instead of implementing the SERDES functionality using FPGA logic. The same principle applies to other often used elements, such as First-In First-Out (FIFO) or Random Access Memory (RAM) modules. This approach renders better efficiency and speed at the cost of portability.

In the ARACNE IPCore design, modules of both types are identified; as such, some parts of this design will need to be partially adapted to the specific devices chosen in the implementation, and the rest can be programmed once and used in any design. Most prominently, the elements that provide the functions of the digital acquisition chain are of the last type. As such, they can be reused an interconnected at will to create new applications just like standard instrumentation modules can be interconnected and configured in a modular instrumentation system, while the hardware-dependent modules stay the same. Figure 3.4 illustrates this concept.

In figure 3.4, modules outside the dotted line receive signals from the external hardware elements and transform them into generic, hardware independent signals grouped into logical interfaces, which are then used by modules that are

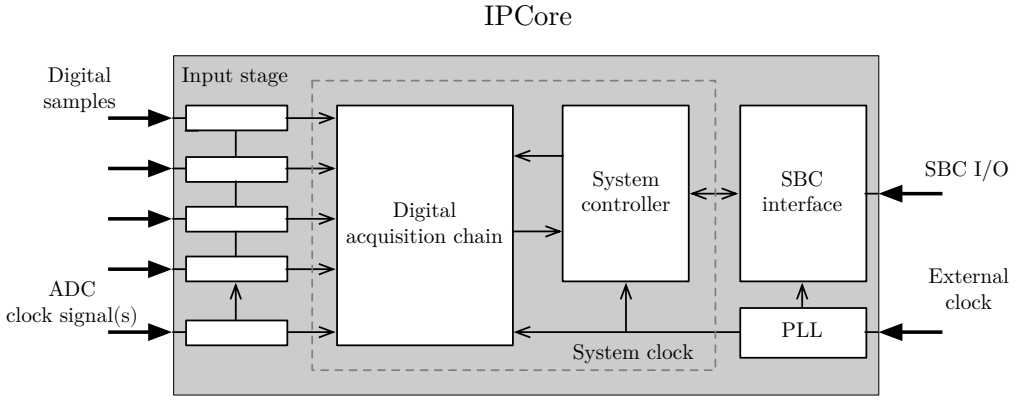


Figure 3.4: ARACNE general IPCore design diagram.

programmed based on them. For example, there are diverse ways for an ADC to output the sampled data, such as parallel or serial interfaces as will be explained later; but from a logic point of view, all of them provide a full sample in sync with an additional sample clock signal. The modules in the *input stage* in the aforementioned figure would transform the hardware ADC signals into the logic sample data interface, which in this example would be composed of data signals representing the bits of each sample, and a sample clock signal.

The user is free to include additional functionality to the IPCore, but the fundamental modules in the general ARACNE IPCore design proposal are the following:

- *Input stage.* This stage, as explained in the example above, is hardware platform dependant and transforms the specific ADC sample output interface into a simple parallel data interface composed of the sample data and a clock signal. In some cases this stage will be almost non-existent since some ADCs already output data in this or a very similar format, but in others (such as single or multiple serial link interfaces) it will be a necessity.

- *Digital acquisition chain.* This module is platform independent and implements the functionality of the acquisition chain. It will be composed of a variety of interconnected IPCore modules, just like standard modules in a modular instrumentation system, that will implement a variety of functionalities such as triggers, coincidence detectors, peak detectors, counters, filters, stretchers, FIFO buffers, etc. The signals and parameters that define the behavior of each module will be mapped to registers that the system controller will be able to read and write, and data results produced by these

modules may be read either serially or through memory buffers.

- *System controller.* This is a platform independent module that on one side will receive commands and report status and data results to the system's SBC through a generic logical interface, and on the other will be able to configure, parametrize and control the digital acquisition chain through a register file where signals and parameters of all modules will be mapped.
- *SBC interface.* This stage is hardware platform dependent and will implement the communications interface employed by the SBC, translating it into a logical data stream that is understandable to the system controller and vice-versa. For example, the SBC may communicate in a variety of ways such as through serial SPI or UART interfaces, or an ad-hoc solution; and the SBC interface would adapt the communications type to a common logical interface.

3.3.3 Support libraries

In a way similar to a modular instrumentation system, the acquisition chain is configured and monitored via software running in the system's computer, in this case the SBC hardware module. Also, results and processed data are read gathered from the acquisition chain with this software, so that they can be further processed, stored, transmitted or displayed.

Since every application is different and the digital acquisition chain implemented in the IPCore varies, different software will need to be produced for each case. Also, each synthesizable instrumentation module will require different configuration and programming, requiring a separate library or set of functions to deal with its peculiarities in a similar way a device driver deals with specific hardware in a computer system.

However, as explained in sections 3.3.1 and 3.3.2, there are common elements in the hardware platform and the IPCore structure that will always require a certain treatment from the programmer's point of view:

- There will always be an ADC, and regardless of its specifications there will always be a need to communicate with it and establish its parameters, typically through registers.
- The SBC will be connected to the FPGA (and consequentially, to certain IPCores) through General-Purpose Input/Output (GPIO) pins and other communications channels, such as Serial Peripheral Interface (SPI).

- The software will always need to interact with the IPCore system controller using some communications protocol, regardless of the specific communications channel used for this purpose both in the SBC and the corresponding IPCore module.
- Also, the interaction between the software and the IPCore system controller will be similar in all implementations (i.e. read or write registers or read data), regardless of the effect or ultimate recipient of this interaction.

Consequentially, software libraries to deal with the above common functionality can be provided to hide many of the implementation details to the programmer and making it easier both to produce high level software and expand the IPCore *driver* library. To this matter, a low-level software structure as outlined in figure 3.5 is proposed, trying to account for the necessities both of the final user and the IPCore designer.

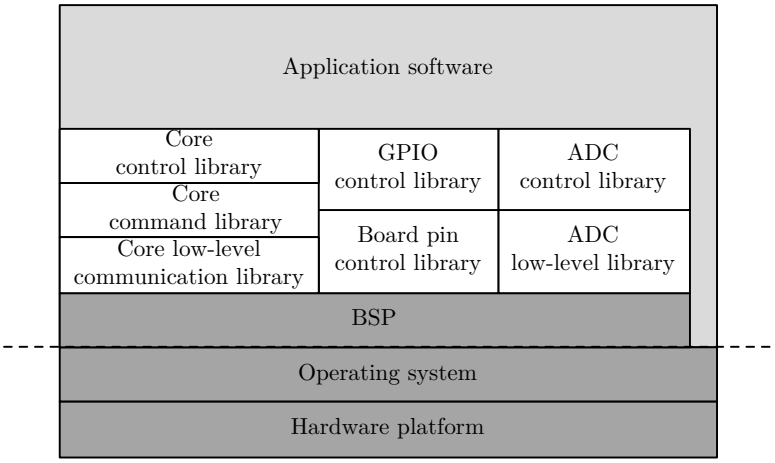


Figure 3.5: Proposed ARACNE software structure diagram.

- The SBC will run an **operating system**, and a third-party I/O library or **Board Support Package (BSP)** will be provided in order for the software to be able to access the hardware resources like I/O pins, communications devices such as SPI, Universal Asynchronous Receiver-Transmitter (UART) controllers and other miscellaneous hardware elements.
- An **ADC low-level library** will deal with the communication with each specific ADC device and the procedure to read and write its registers, offering to the upper layer a generic, high-level interface for reading and writing

the device registers. This interface will in turn be used by the **ADC control library** to implement software that leverages the features of the specific ADC device, offering higher level commands for the ADC configuration and operation.

- Similarly, a **board pin control library** will deal with the GPIO functionality provided by the specific BSP, offering the upper layer a generic pin control interface for setting the pins mode (i.e. input or output), setting or reading its value, etc. This interface will in turn be used by the **GPIO control library** that will implement the control software for the operation and default configuration of this type of resources.

- Regarding the control of the IPCores, a more complex library structure is proposed, since two things must be dealt with; interaction with the system controller IPCore and with specific IPCores of the detection chain. The **core low-level communications library** deals with the hardware communication interface chosen to interact with the system controller IPCore (for example, an SPI link), using the specific BSP capabilities. This will offer a simple send and receive high level interface to the upper layer. With this, the **core command control library** defines the communications protocol with the system controller in terms of message format and a set of generic commands (for example, read or write a register from the system controller register bank) that allow the interaction with the rest of the IPCores of the system and the transmission of experimental data. Finally, a **core control library** module implements the software that, using the commands provided by the previous layer, is able configure and monitor each specific IPCore of the system so that it provides the high-level functionality needed for each application.

- With all this, the user will be able to write its **application software** using the functionality provided by the aforementioned libraries, that provide high-level interfaces to manage the ADC, GPIO interfaces and IPCores without dealing with many of the details of the hardware platform or the IPCores.

Some of these libraries must be written for each specific ARACNE implementation, since hardware elements and communication details can vary, and others can be kept mostly the same. Ideally, a core library module should be mostly implementation-independent, just like the synthesizable IPCore modules of the detection chain; and a library module should always be provided for each IPCore.

3.4 Platform requirements overview

The design of the hardware platform is probably the most critical when producing an ARACNE prototype, since once produced hardware is hard or impossible to change to account for an unforeseen application scenario. For this reason, an analysis of the main requisites that must be taken into account when planning for a new design has been done at several levels. This section sets out these requirements and outlines some of the design decisions of the proposed ARACNE prototype detailed in section 4.

3.4.1 Analog signal input and conditioning

The first decision should be what is the first stage of the detection chain that ARACNE should implement, which will define the type of signals that the system will be expected to deal with.

Since there's a wide variety of detector types with different electrical characteristics, and preamplifiers are usually very detector-specific, it becomes apparent that it's not worth it to try defining a common electronics platform at this level. Also, the preamplifier must usually reside physically very close to the detector itself, which is incompatible with a system that's supposed to be able to work with multiple detectors simultaneously and possibly be rack mounted just like standard instrumentation modules.

Similarly, the linear shaping amplifier, which is the next stage in the chain, needs to be close to the preamplifier except in some cases where the signal generated by it is large enough to be transmitted through a distance without problems, and its first shaping and amplification sections are commonly detector specific.

For this reason, a choice has been made to design a system intended to accept amplified and at least partially shaped signals. These signals will usually be shaped pulses with a baseline around 0 V, that can be positive, negative or bipolar and with amplitudes in the range of ± 1 V. The bandwidth of these pulses will depend on the shaping, and their duration will usually range from a few hundreds of nanoseconds to tens of microseconds. Finally, the system will preferably present an input impedance of 50 Ω, which is a widely used value for the characteristic impedance of coaxial cabling, although other impedance values should be easily supported recalculating some component values.

However, the platform should provide some reasonable flexibility to support other pulse types (such as logical pulses, fast pulses or signals directly generated

by some preamplifiers) or pulses with characteristics that are not ideal. For this reason, some optional analog conditioning circuits will be included in the design. As such, the input stage will feature the following optional elements:

- Decoupling circuitry, to support AC and DC coupled inputs.
- Attenuator circuit such as the one described in section 2.3.4, to support input signals of higher amplitude without affecting the input impedance.
- Basic RC shaping circuitry that allows the user to implement a simple RC or pole-zero cancellation shaping circuit similar to the ones described in section 2.3.2.

Finally, ARACNE has been conceived from the beginning as a platform supporting operations involving multiple signals simultaneously (such as coincidence or anticoincidence), so multiple analog input channels are supported and each of them will individually include the aforementioned optional functionality.

3.4.2 Analog to digital conversion

The analog to digital conversion must be performed by an ADC chip, whose basic parameters as explained in section 2.3.8 are sampling frequency and bit depth. However, when selecting an ADC there are other technical and practical decisions that need to be addressed, and in our case where covering a wide range of scenarios is key, these are of great importance. As such, we need to consider these and other aspects:

- **Multichannel capability**. Our goal is to design a system capable of working simultaneously with several analog input signals that in most cases will have some kind of correlation with each other, i.e. signals from two scintillators working in coincidence. Because of this correlation, it is of great interest not only to be able to convert several signals simultaneously, but to make sure that the signals are sampled at exactly the same time. It is possible to use several separate ADC chips in the same circuit, but synchronization of the data samples of all the channels is much easier if the ADC is of the multi-channel type to begin with, and it samples and outputs synchronized data from all channels simultaneously. To this matter, there are ADCs in the market that support 2, 4, 8 or more channels that provide this capability.

- **Input type and voltage range**. ADCs are designed to accept either single-ended or differential inputs, being the latter option preferred to achieve higher noise immunity. Depending on the choice of ADC, the analog input stage must account not only for adaptation between the input signal voltage range and the ADC input range, but also of any single-ended to differential signal conversion that may be needed.

- **Output type**. Once the analog signal is sampled, the resulting binary data must be output through some kind of wiring to the next stage in the chain. Depending on the sampling rate, the bit depth and the physical pins available on the chips to implement this interface, the method to output the data can vary.

  The most simple way to output sampled data is a parallel interface, where there's a wire for each bit in the sampled word, e.g. 12 pins for a 12-bit ADC, and a clock pin with a frequency equal to the sampling rate to indicate that a new sample is present in the output pins. This option provides the fastest achievable data rate at the cost of needing a lot of signals, which means larger chips and more complicated routing, since signals must travel through several circuit board traces that must be kept of equal length to avoid synchronization problems. Naturally, the problem gets worse in ADCs with large bit depths or supporting multiple simultaneous channels, i.e. an 8-channel, 14-bit ADC would need 112 pins dedicated just for data output, and 112 traces should be carefully routed on the PCB.

  Another way to transmit sampled data is through a serial interface such as SPI or LVDS. In this case, samples are transmitted bit by bit through just one signal pair, in some cases using another clock line for synchronization. This method uses the least number of pins both in the transmitting and receiving devices and greatly simplifies the circuit layout. However, since the bit rate of this signal is the product of sample rate by bit depth, the frequency of this signal can be very high and problematic to transmit properly through a circuit trace; and also, it can exceed the processing speed achievable by the receiver. For example a 10 MS/s, 12-bit ADC would require a 120Mbps serial link and a device capable to process data at this rate. This often limits both the sample rate and bit depth. Higher speed signaling protocols such as Low Voltage Differential Signaling (LVDS) are often used in these cases, and sometimes samples are transmitted using more than one serial link to reduce signal frequency, at the cost of transmitter and receiver complexity.

  Since ARACNE is conceived as a multichannel platform, serial links will be preferred; but depending on the channel number, sampling speed and bit

depth required, the choice of ADC can change without affecting the general proposition.

- **Sampling rate**

  As mentioned in section 2.3.8, the sampling rate of the ADC must be just *good enough* for each particular case. Since ARACNE is just an approach to building digital detection chains, the choice of ADC is independent of the concept, and could be chosen, together with the electronics in charge of using the data generated by it, on an ad-hoc basis.

  However, a good first approximation would be to consider shaped pulses of the characteristics described in previous sections, where the higher frequency components of the signal depend on the rise time of the pulse. Every case will be different, but it is reasonable to consider that the duration of these pulses will be of a few microseconds, and the rise time will be between a few tens to a few hundreds of nanoseconds. Using Nyquist's sampling theorem, an ADC capable of sampling frequencies between 10 MHz and 100 MHz is a good target for a prototype.

- **Bit depth**

  Again, as explained in section 2.3.8, bit depth should be high enough to minimize the quantization error and its subsequent effect in the processing precision (i.e. limiting the number of channels that a MCA can provide), but not so high that the ENOB is limited by the noise in the signal. It is difficult, however, to foresee the precision necessity of a generic platform or the noise resulting of a specific configuration, so oversizing the bit depth seems like a reasonable approach since LSBs can be easily dropped by the digital processing electronics if deemed unnecessary.

  Since the input voltage range proposed is $2V_{pp}$ (from $-1\,\mathrm{V}$ to $1\,\mathrm{V}$), a minimum target bit depth of 12 bits seems reasonable and provides a quantization of $\approx 0.5\,\mathrm{mV}$.

- **Other considerations**

  Apart from the main parameters summarized above that determine the signal processing capabilities of the system, there are other considerations when choosing an adequate ADC that are important for other factors such as usability, flexibility or manufacturing feasibility.

  - *Power consumption.* The power draw of an ADC is typically determined by the sampling frequency, the data rate of the output signals and the number of channels. Since ARACNE is aimed at providing

a resource-efficient platform that should not require complex power supply circuits, this is a characteristic that should be observed.

- *Control interface.* Most modern ADCs allow a high degree of flexibility and their working parameters can be modified dynamically during operation, apart from offering a lot of advanced features. They are usually managed by a dedicated microcontroller or a computer through a more or less standardized interface, such as SPI or UART. Choosing an control interface that is simple to implement and easy to program is of great importance.

- *Packaging.* As technology in the field of electronics has progressed during decades, component miniaturization has improved greatly allowing for smaller and cheaper PCBs and higher performing devices. However, old packaging technologies that could be easily handled, soldered and tested by a human operator have given way to packagings that are next to impossible to handle except by a robotic manufacturing facility or very expert professionals using specific equipment. So, even though this is not really a key factor if an implementation of ARACNE is going to be manufactured in a high technology facility or by a third party, the choice of a package that's reasonably manageable such as a variant of Quad-Flat Package (QFP) is deemed most convenient in a prototyping stage.

- *Additional digital processing and advanced features.* Some modern ADCs offer an additional digital processing block after the analog to digital conversion that can significantly increase the system flexibility, offloading tasks that otherwise should be performed by other parts of the system. For example, some of this digital processing functions include:

  - Configurable Infinite Impulse Response (IIR) or Finite Impulse Response (FIR) digital filters.
  - Configurable digital gain.
  - Multi-channel input to output routing, so that any analog input channel can be output to any digital output.
  - Channel averaging.

  Finally, some ADCs also offer the automatic generation of test patterns so that the digital chain can be tested independently of the analog inputs.

3.4.3 Programmable logic hardware and IPCore

Once the analog input signals have been converted into digital signals, they can be processed, used and analyzed to provide the scientific results required by the application. However, the amount of data produced by a high-speed ADC and the real-time constraints inherent to the continuous analog to digital conversion are impossible to deal with by a conventional microprocessor system, even less so in a multichannel application. For example, a four-channel, 20 MS/s, 12-bit data acquisition system would generate four new 12-bit samples every 50 ns, for a total 120 MB of raw data every second that should be analyzed, filtered and possibly stored in real time.

This problem led to the appearance of specialized processors, specifically designed to deal with real-time data produced by an ADC and run digital processing algorithms on them, known as Digital Signal Processors (DSPs). However, just like in a conventional processor, all the stages of the signal processing are carried off by software tasks (or *threads*) that share common processor resources and can interact in unexpected ways even with careful programming, potentially causing system failures due to these threads not being able to comply with the real-time constraints. The problem is made worse as data rates are higher, multiple signals must be dealt with at once, and more complex algorithms are needed. Also, highly capable DSPs are usually very power-hungry and run very hot.

This can be solved by using specific purpose hardware instead of software running in a microprocessor or DSP, partitioned effectively to avoid resource sharing conflicts. Each digital signal is assigned its own logic resources and each processing stage is implemented separately, effectively achieving true parallelism and providing real-time processing capability. This can be achieved by designing an application specific logic circuit with such partitioning, which can be implemented straight into silicon, in a device known as Application Specific Integrated Circuit (ASIC), or in a PLD, such as an FPGA. The latter, which can be reprogrammed to change the function and capabilities of the implemented circuit, is of course the most flexible solution; and advances in speed and logic capacity that allow the implementation of very complex and high performance designs on FPGAs have made them ideal to use in signal processing applications.

Thus, an FPGA is at the core of the ARACNE proposal, and synthesizable hardware modules known as IPCores implementing the processing capabilities, that can be combined and implemented in the FPGA fabric, provide the specific functionality of the digital acquisition chain.

In summary, the functionality that must be delivered by this FPGA and the

IPCores implemented in it is:

- Receiving the sampled data produced by the ADC in real time.
- Perform the required digital processing that implements the functionality of the detection chain.
- Deliver the application results to a computer for storage, transmission, display or further processing.
- Provide a reasonable degree of control and configurability of the implemented detection chain via parameters that can be programmed from a computer or via other hardware elements (such as buttons, switches, etc.)

At first sight, the requisites for the device are simple: enough I/O pins to implement the interfaces to the ADC and the computer, and enough programmable logic capacity (in terms of *system gates*) to implement the synthesizable IPCores. However, some other aspects must be considered given the use case:

- Maximum working frequency, meeting at least the expected sample rate from the ADC.
- Availability and amount of native memory blocks, usable either as RAM or FIFO memory.
- Availability and amount of integrated clock generation circuits such as Phase-Locked Loops (PLLs).
- Availability of dedicated SERDES blocks for applications where data is transmitted or received serially at high speeds.
- Supported I/O logic standards, both single-ended and differential, in accordance with the ADC and computer interfaces; for example, LVDS or LVTTL.
- Device technology. There are two main types of FPGA devices, RAM based or flash based. The former is unconfigured at power-on (e.g. without any IPCore loaded) and then loads the device configuration from a dedicated RAM chip, while the latter keeps the configuration when powered off using internal non-volatile flash memory. RAM based FPGAs are usually faster, while flash based FPGAs are instant-on and typically require less power.

There are also other considerations, such as availability of different logic capacities in the same FPGA family while keeping pin compatibility, and availability of a manageable packaging for prototyping as explained in section 3.4.2. These details may seem secondary, but are good to take into account.

In ARACNE the decisions about I/O will depend greatly on the chosen ADC, but given its multi-channel aspirations, support for high-speed LVDS will be practically a must. And the needed logic capacity of the device is not easy to determine in advance unless the IPCores are developed and the digital chain structure has been already laid out and synthesized; but modern FPGAs offering in excess of 1,000,000 equivalent system gates are not rare and should be enough for most applications in a multichannel system, so it seems a reasonable target.

Finally, required operating frequency will depend heavily on the sampling rate and ADC data transmission method, since data transmitted serially will need to be deserialized at a rate that is the byproduct of the sampling rate by the bit depth (although for very high data rates more than one serial link per channel can be used, thus alleviating this requirement). For example, a 20 MS/s, 12-bit ADC transmitting data serially would require the FPGA to operate at least at 240 MHz just to deal with the incoming data.

3.4.4 Embedded computer

In modular instrumentation systems, the detection chain elements are managed by a computer, either mounted in the system's chassis or connected to the controller module (i.e. the crate controller in a CAMAC system, or arbiter module in VXI). This computer is not only in charge of configuring and monitoring the instrumentation modules that make up the detection chain, but usually stores, transmits or displays the resulting data, and can also perform data processing that is not feasible otherwise.

In ARACNE the computer is used for the same purposes, but in this case the data processing aspect is more prominent. Synthesizable hardware running in a FPGA has a great advantage in terms of parallelism and real time operation, as explained in section 3.4.3, and is potentially capable of performing complex data processing if the IPCore programmer is skilled enough. However, very often it is much easier to implement complex processing algorithms in software and not all of the data processing needs to be done in real time. For this reason, the design philosophy of ARACNE is to divide the tasks of the detection chain between the FPGA and software running in the system's computer.

Additionally, the computer must control the ADC, which resides in the same

PCB as the FPGA, and any other additional facility that may be included in the platform, such as indicators or additional I/O.

For this reason, the idea is to connect the computer to the main PCB, directly to the FPGA device, ADC and other electronics, thus resulting in a tightly coupled embedded system. This allows the implementation of a variety of communications interfaces between the IPCores in the FPGA and the embedded computer, which allows higher flexibility and potentially higher communications speed.

On the other hand, one of the premises of ARACNE is to propose a system that is also compact and requires low power for operation. Together with the direct connection requisite, it seems natural to opt for some Single-Board Computer (SBC) solution that provides the necessary FPGA hardware interfaces, such as GPIO pins or serial interfaces such as UART or SPI, and standard networking and I/O interfaces to access the computer itself.

Finally, this computer should be based on a platform capable enough to run an advanced operating system such as Linux, and all its accompanying software, reasonably fast. Using such a UNIX-based environment opens the door to using a wide range of readily accessible tools and languages to perform almost any processing, storage or communication function, and provides the acquisition system other important features such as networking and remote access.

Currently, there's a variety of COTS SBC solutions based on Acorn RISC Machine (ARM) SoCs that fulfill this requisites and provide a very powerful embedded computer supporting Linux, with a variety of interfaces, which are compact, reasonably low power and very affordable.

3.4.5 Power and form factor

The idea behind ARACNE is to propose a system that requires low power and has simple power supply needs for operation, while keeping a small footprint.

The main power-consuming elements in the system are the ADC, the FPGA and the SBC, so the choice of each of them and their operating conditions define the power requirements of the system. Most ARM-based SBCs have a maximum power consumption of between 5 W and 15 W. Power required by an FPGAs depends on their technology (either flash-based or RAM-based), but also on their operating frequency, amount of logic used and the IPCore design. And regarding ADCs, the number of channels and sample rate define their power consumption. Also, each device requires different voltages for operation and most of them require different voltages and power supplies.

For this reason it's difficult to give a target power figure, so the power supply will have to be chosen in a case-by-case basis. However, in order to simplify the system installation and operation, a single power supply will be used and the rest of voltages will be provided by on-board power conversion elements, so that ideally a simple 5 V supply can be used.

Regarding form factor, since the system is self contained there's no requisite to comply with any of the card formats defined in modular instrumentation standards such as those mentioned in section 2.4. However, it may be a good idea to use one of the Eurocard standard dimensions so that the system can be mounted in a standard rack if necessary.

Chapter 4

Prototype

"Are you telling me that you built a time machine... out of a DeLorean?"
— Marty McFly, *Back to the Future*

4.1 Overview

A prototype data acquisition system for nuclear applications following the philosophy outlined in chapter 3 has been built and used successfully in the Cosmic Ray Antarctic Observatory (ORCA). This chapter describes the design and construction of the system, which includes the three aspects introduced in section 3.3:

- **The hardware platform**, which includes the design and manufacturing of the PCB and the choice of the main system components.
- **The IPCore**, which implements the functionality of the digital acquisition chain required for the Muon Impact Tracer and Observer (MITO) instrument of the ORCA project. This includes a set of platform-specific IPCores, a system controller and detection chain modules, plus their combination to provide the desired functionality.
- The software **system libraries** that provide access to the platform resources to software running in the SBC, and the specific acquisition software corresponding to the MITO application.

The resulting prototype is of course not restricted to its use in the MITO instrument, and has been designed with other applications and scenarios in mind, such as the muon telescope described in chapter 1, thus demonstrating the feasibility of the concept. However, some decisions for the hardware platform were made based in the requisites of the data acquisition needs for MITO, which will be thoroughly described in chapter 5. Without entering into much detail, the MITO detector plus the installed amplifiers generate eight simultaneous shaped linear pulses with a duration of a few microseconds, at a rate above a hundred pulses per second. Depending on the specific MITO application, the acquisition system must be able to work in coincidence between channels, count events, register pulse heights for spectrography or impact point determination, and even capture the pulse shapes.

The following sections will discuss the design decisions, components selected and other details.

4.1.1 Miscellaneous additional features

In addition to the main functionality, the prototype has been designed with a few additional features to make the system testing easier and to add expansion possibilities. More specifically:

- Circuitry has been added so that the FPGA can generate test logic pulses in all inputs. This way, the acquisition chain can be tested without the need of a pulse generator or the actual detector signals.
- Several communication interfaces among the SBC and the FPGA have been placed, so that different solutions can be implemented.
- Provisions have been made to be able to exchange any of the on-board power supplies for external ones through a header, in case the power needs of any of the stages change (i.e. the input amplifier needs different power rails in a specific application).
- An external two-wire interface with the FPGA where a serial UART protocol can be implemented has been added. This way, the IPCore may be debugged, monitored or controlled without the need of an SBC.
- Unused digital inputs and outputs from the SBC and FPGA are available to the user in the form of LED indicators, buttons and jumpers so that they can be used as GPIO inputs or user interface. In the case of the FPGA, this allows the input and processing of additional logic pulse signals.

- Unused analog inputs from the SBC are available on a header, so that they can be used for miscellaneous applications, such as high-voltage power supply monitoring.

4.2 Hardware platform

The design of the hardware platform has been done in accordance with the structure proposed in section 3.3.1. As such, its main elements are an analog input stage, an ADC, an FPGA and a SBC, plus power supplies and user I/O. The platform design, component selection and PCB production details of each stage will be discussed next.

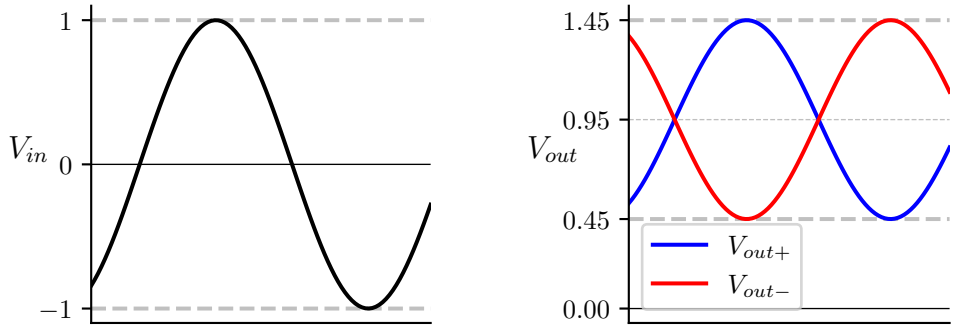
4.2.1 Analog input stage

This section of the hardware platform adapts the analog signals coming from the detector and amplifier to the signal required by the ADC, while providing impedance adaptation, proper termination and optional signal conditioning. Since the prototype must be able to deal with 8 signals generated simultaneously, the hardware dedicated to this task will need to be replicated 8 times in the PCB.

The expected input signals are described in section 3.4.1. Single-ended bipolar signals between $\pm 1\,\mathrm{V}$ (which accounts also for signals between $0\,\mathrm{V}$ and $1\,\mathrm{V}$ with undershoots or baseline offsets) are expected, and an input impedance of $50\,\Omega$ has been selected as the most convenient. However, the input range and impedance may be changed by recalculation of a few system components and usage of the on-board attenuator provided by the optional signal conditioning circuitry. It is worth mentioning that the PCB provides the possibility of installing either BNC or SMA type connectors for signal input.

On the other hand, the signal expected by the ADC depends, of course, on the chosen ADC. In the case of the ADC chosen for this prototype, which is thoroughly described in section 4.2.2, this input is differential with a common voltage of $0.95\,\mathrm{V}$ and a maximum differential excursion of $2\,\mathrm{V_{pp}}$ (which means the differential inputs can have voltage values in the range $0.95 \pm 0.5V$). To this matter, the ADC provides a common voltage reference of $0.95\,\mathrm{V}$. Figure 4.1 illustrates the input and output signals of this stage.

Since single-ended to differential signal conversion needs to be performed and input and output voltage ranges are equal, a differential amplifier circuit with unity gain using a Fully Differential Amplifier (FDA) has been used. Additionally,



(a) **(b)**

Figure 4.1: Expected input and output signals. Note that the differential output $V_{diff} = V_{out+} - V_{out-}$ is also a $\pm 2\,V_{pp}$ signal, like the input.

the aforementioned optional conditioning circuitry has been included both at the input and output of the amplifier and some control and testing signals have been included, as illustrated in figure 4.2.

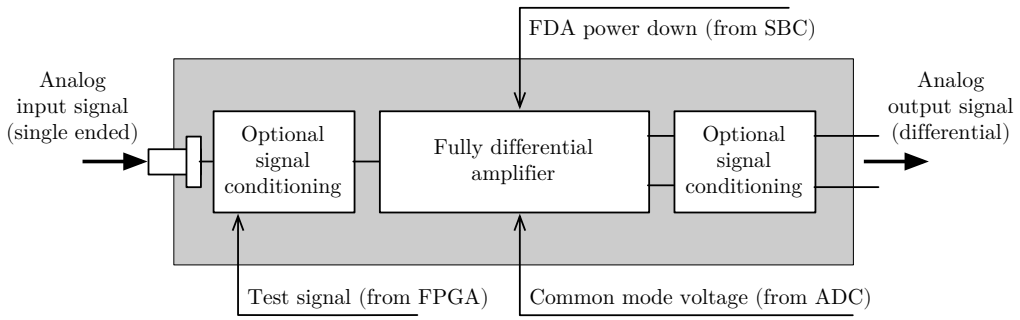


Figure 4.2: Input stage diagram.

Details for both the FDA circuit and optional conditioning circuitry are described in the following sections.

Fully-differential amplifier

A Fully Differential Amplifier, represented in figure 4.3, is similar to a regular operational amplifier, which was described in section 2.3.1. Both have differential inputs, have very high input impedance and very high open loop gain, so the voltage between its input terminals can be considered zero as long as they are in

linear operation. However, the output of an operational amplifier is single-ended, while the output of a FDA is differential.

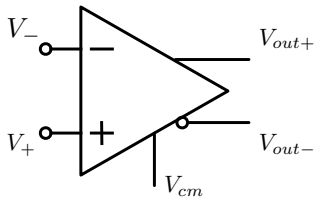


Figure 4.3: Fully differential amplifier.

More specifically, $V_{od} = A \cdot V_{id}$ where A is the amplifier's gain, $V_{id} = V_+ - V_-$ and $V_{od} = V_{out+} - V_{out-}$. The implication of this is that while in an operational amplifier the output common mode voltage and the signal are the same and depend on the input common mode voltage, in an FDA the input common mode voltage is eliminated and the output common mode voltage is controlled independently, typically through a specific V_{cm} input.

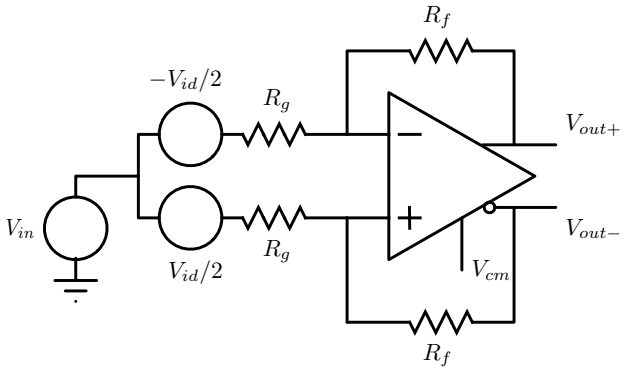


Figure 4.4: FDA basic circuit for analysis.

This is demonstrated through analysis of the most basic FDA configuration, depicted in figure 4.4, where both feedback paths are balanced and forming two inverting amplifiers. In this figure, the differential input is represented by its common mode and differential components defined as $V_{ic} = (V_{in+} + V_{in-})/2$ and $V_{id} = V_{in+} - V_{in-}$. In this conditions, considering only V_{ic} and since $V_- = V_+$:

$$\frac{R_g V_{out+}}{R_f + R_g} + \frac{R_f V_{ic}}{R_f + R_g} = \frac{R_g V_{out-}}{R_f + R_g} + \frac{R_f V_{ic}}{R_f + R_g} \tag{4.1}$$

$$V_{out+} - V_{out+} = V_{od} = 0 \tag{4.2}$$

So the common input voltage V_{ic} has no effect in the differential output V_{od}. And similarly for the differential input component, V_{id},

$$\frac{R_g V_{out+}}{R_f + R_g} + \frac{1}{2}\frac{R_f V_{id}}{R_f + R_g} = \frac{R_g V_{out-}}{R_f + R_g} - \frac{1}{2}\frac{R_f V_{id}}{R_f + R_g} \tag{4.3}$$

$$R_g(V_{out+} - V_{out+}) = R_f V_{id} \qquad \rightarrow \qquad V_{od} = V_{id}\frac{R_f}{R_g} \tag{4.4}$$

So the differential output depends only in the differential input voltage, with R_f/R_g being the closed loop gain of the amplifier, and not at all of the common input voltage. This result has important implications, since noise coupled into the input wires appears as common mode voltage, which is efficiently rejected by a FDA. But also because the conversion between single-ended and differential signals is also very easy with a circuit such as the one in figure 4.5, where it can be easily demonstrated that $V_{od} = A \cdot V_{in}$ where $A = R_f/R_g$.

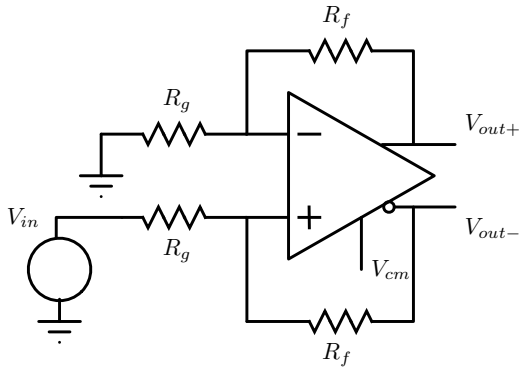


Figure 4.5: Single-ended to differential conversion circuit using an FDA.

Additionally, because of its internal balanced construction, an FDA is more immune to coupled noise at the output and noise coming from power supplies that also typically appears as common mode voltage [29]. Finally, the functionality of an FDA can be achieved using a complex circuit and discrete operational amplifiers and components, but an integrated FDA offers a more elegant, precise and compact solution.

In the case of the circuit for the ARACNE prototype, and taking into account the input and output specifications, a single-ended to differential conversion circuit with a gain equal to 1 and $V_{cm} = 0.95\,\mathrm{V}$ would suit the needs. However, two things remain unsolved; proper termination so that the transmission line finds a $50\,\Omega$ input impedance, and keeping the balance between the two amplification

paths while doing so. To achieve this, the circuit in figure 4.6 is used, where R_s is 50 Ω and R_t and the four main resistors must be calculated.

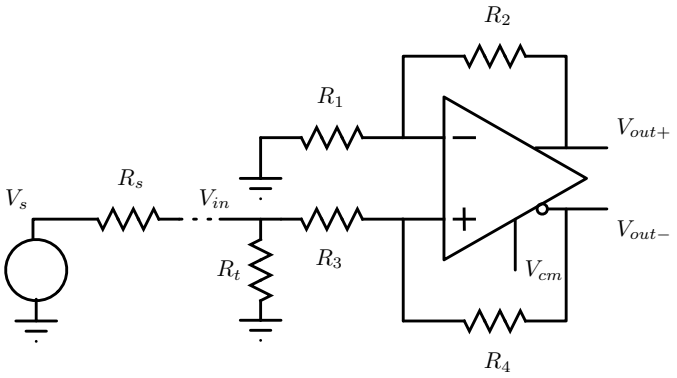


Figure 4.6: Single-ended FDA circuit with input termination.

Assuming linear operation and that the amplifier is balanced, $V_{out+} = 1/2 \cdot AV_{in}$ (half of the differential output) if V_{cm} is zero, where A is the closed loop gain and its value must be $A = R_2/R_1$. In this conditions,

$$V_+ = V_- = \frac{V_{in}}{2}\frac{R_1}{R_1 + R_2}A \qquad \rightarrow \qquad I_{R_3} = \frac{V_{in} - V_+}{R_3} \tag{4.5}$$

And thus the impedance seen at the input in parallel with R_t is

$$R_{in} = \frac{V_{in}}{I_{R_3}} = \frac{R_3}{1 - \frac{1}{2}\frac{R_1}{R_1+R_2}A} \qquad \rightarrow \qquad R_{in} = \frac{R_3}{1 - \frac{1}{2}\frac{A}{1+A}} \tag{4.6}$$

and for input adaptation, $R_s = R_t \parallel R_{in}$, so R_t can be calculated as

$$R_t = \frac{1}{\frac{1}{R_s} - \frac{1}{R_in}} \tag{4.7}$$

And finally, since the Thevenin equivalent of the circuit is as shown in figure 4.7, to keep the balance $R_4 = R_2$ and $R_1 = R_3 + R_s \parallel R_t$ must also hold. From this condition and equations 4.6 and 4.7, all values can be calculated by fixing A and R_s, choosing a starting value and iterating. For example, in our case $R_s = 50\,\Omega$ and $A = 1$, so initially assuming $R_1 = 365\,\Omega$ and $R_t = 50\,\Omega$, $R_3 = 365 - 50 \parallel 50 = 340\,\Omega$. With this result, $R_t = 56.2\,\Omega$, which in turn will require adjusting R_3 and so forth. This can be easily done using a spreadsheet, for which there are some examples that can be found online.

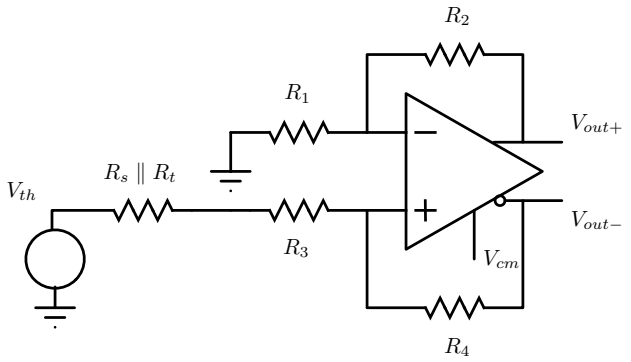


Figure 4.7: Thevenin equivalent of the single-ended FDA circuit.

Another important aspect of the circuit design is choosing the appropriate power supply voltages, not only to comply with the required output voltage ranges but also to ensure that the voltage at the inputs of the FDA does not violate the input common-mode voltage range. Going back to figure 4.5 and assuming linear operation, $V_+ = V_-$ so calculating the common mode voltage range for one input is enough. For the non-inverting input the voltage is

$$V_{ic} = \frac{R_g}{R_f + R_g} V_{out-} + \frac{R_f}{R_f + R_g} V_{in} \tag{4.8}$$

which can be calculated at the extremes of V_{in}. In our case, assuming a gain of 1 implies $R_f = R_g$, and the input and output signal ranges were mentioned before and represented in figure 4.1. When $V_{in} = -1\,\mathrm{V}$, $V_{out-} = 1.45\,\mathrm{V}$ and $V_{ic} = 0.225\,\mathrm{V}$, and when $V_{in} = 1\,\mathrm{V}$, $V_{out-} = 0.45\,\mathrm{V}$ and $V_{ic} = 0.725\,\mathrm{V}$. Such low minimum input and output voltages (0.225 V and 0.45 V respectively) imply that if the FDA negative supply is connected to ground, the device will have to be carefully selected and support rail-to-rail input and output. Otherwise, an adequate negative power rail must be used.

Taking all the above into account, the Texas Instruments THS4541 integrated FDA was chosen as the ideal solution for the ARACNE prototype. Not only it is a precision, low noise and high bandwidth device, but it is the only rail-to-rail input and output device of its characteristics. As such, a single power supply can be used in the circuit simplifying the prototype design, although the possibility of using an external negative power supply has been included in the prototype.

The circuit was simulated to ensure proper operation using the real resistor values and FDA device using the TINA simulator provided by manufacturer Texas Instruments, as shown in figure 4.8. The signal source and ADC input equivalents

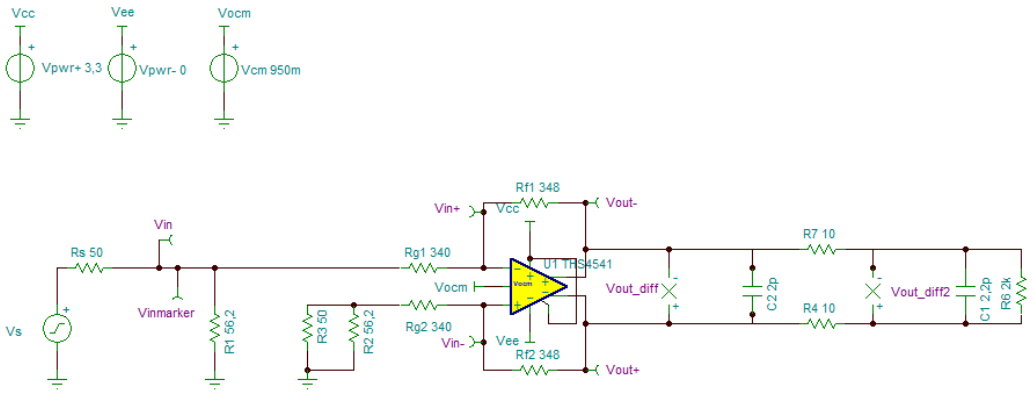


Figure 4.8: Final FDA circuit in the TINA simulation software.

are used, plus a few additional components recommended in the application notes. The input and output of the circuit is shown in figure 4.8, which complies with the expected behaviour. Slight variations in the input and output are due to the fact that standardized resistor values as close as possible to the calculated values must be used, even though measures have been taken to minimize the impact such as using a combination of resistors in the non-inverting input (equivalent to R_1 in figure 4.6) of the same values as in the other side of the FDA, instead of a single resistor.

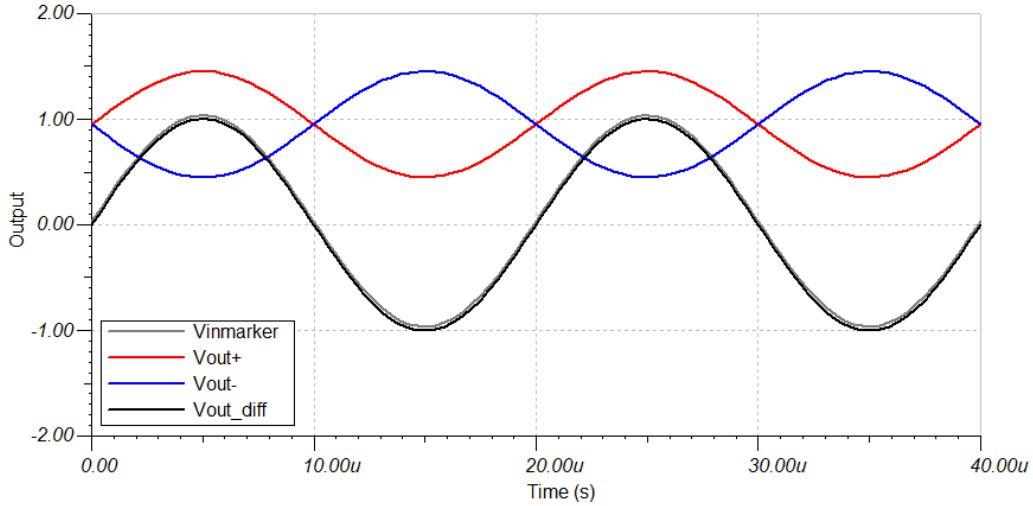


Figure 4.9: TINA simulation results.

Optional signal conditioning

The amplifier stage described in the previous section is enough to adapt the expected single-ended input pulses to the differential input range of the ADC, and the application to the MITO telescope does not require anything else. However, some simple signal conditioning at the input and output of the amplifier stage has been included in the prototype that can otherwise be easily bypassed by soldering a $0\,\Omega$ resistor.

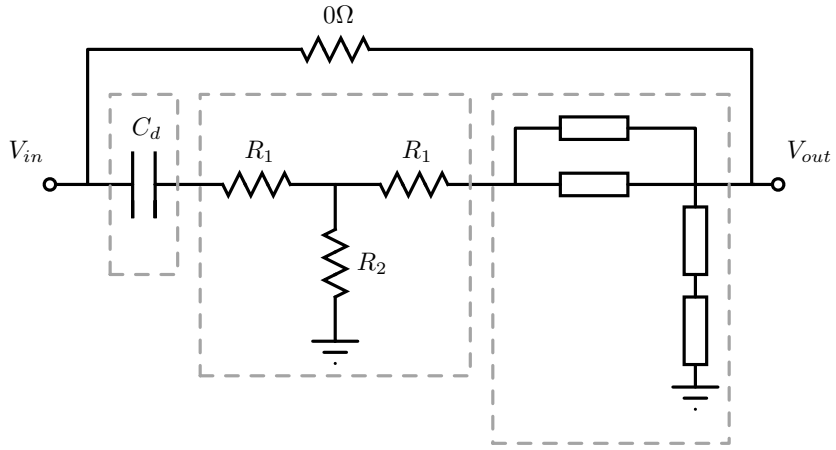


Figure 4.10: Input stage optional input modules.

The input stage is composed of three elements, as shown in figure 4.10:

- A **decoupling capacitor**, for applications where AC operation is needed.
- A **T-pad attenuator** circuit such as the one described in section 2.3.4. By choosing the adequate component values the desired attenuation can be achieved while keeping the same input and output impedances.
- A multipurpose **RC shaping circuit**. Different passive elements such as resistors or capacitors can be installed alternatively to implement several circuit types, such integrators or differentiators as those described in section 2.3.2, pole-zero cancellation circuits such as the one described in section 2.3.2 and others.

All these sections can be easily bypassed by installing $0\,\Omega$ resistors, or all three altogether with a global bypass as mentioned before.

The output stage, depicted in figure 4.11, is composed of just one element that can also be bypassed. The design allows the installation of several passive components that allow the user to implement, among other things, a differential bandpass or low pass filter such as the antialiasing filter required at the input of the ADC in some applications, as discussed in section 2.3.8. Finally, in case the user needs to make use of this stage the output common mode voltage of the FDA may be lost due to the DC decoupling, so additional resistors have been added to regenerate this voltage from the V_{cm} provided by the ADC.

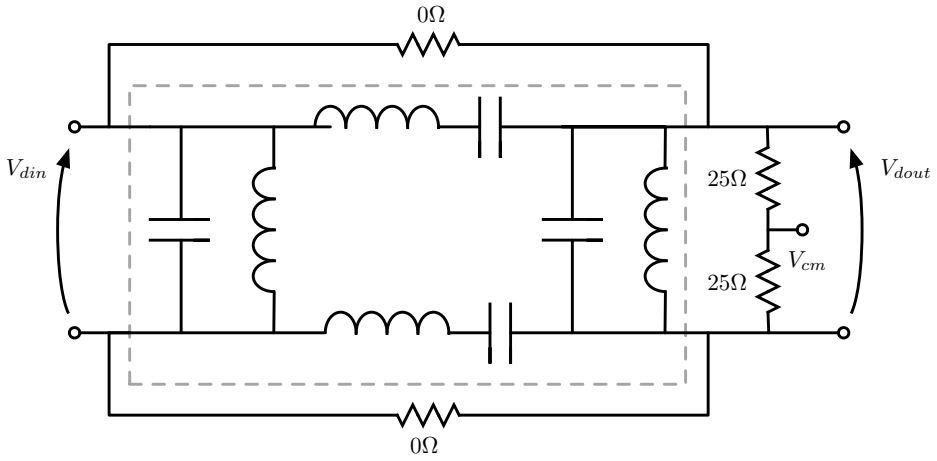


Figure 4.11: Input stage optional output module.

4.2.2 Analog to digital conversion stage

This stage must transform the eight analog input signals generated by the input stage into a stream of synchronized digital samples, and transmit the digital data to the FPGA for processing. The analog to digital conversion must thus be performed by several ADCs, so the main task to undertake for this part of the design is the choice of digital conversion device (or devices), careful treatment of the input and output signals and the appropriate layout of signals on the PCB.

ADC choice

The factors to consider when choosing an ADC and the way this choice conditions other design decisions have been outlined in section 3.4.2. In the case of the ARACNE prototype, one of the most important concerns is that eight signals will need to be sampled simultaneously at high speed and resolution, and tight

synchronization between channels is a must. This means that several ADC devices must be used and controlled, and that a large number of high data rate digital signals will need to be dealt with.

It is quite evident that choosing ADCs with integrated multi-channel capability eases the task considerably, since interchannel synchronization is a given and control for several channels is unified, simplifying the communication among the on-board controlling computer and the analog to digital conversion stage. Also, a multi-channel ADC usually adds features that may be useful, such as channel averaging, arbitrary mapping between input and output signals, etc.

It is possible to find devices with a variable amount of channels on the market that could be combined to obtain the eight channel digitalization capability required, but for the ARACNE prototype a single 8-channel device has been chosen for maximum simplicity, the Octal-channel Texas Instruments ADS5294 [30]. This device is capable of a sample rate of 80 MS/s and has 14-bit resolution, while offering low power consumption (≈ 0.6 W for all 8 channels at full speed). Also, a QFP type packaging that is compact but manageable in lab conditions is available. The input and output characteristics, control interface and other features will be described in the following sections, and are represented in figure 4.12.

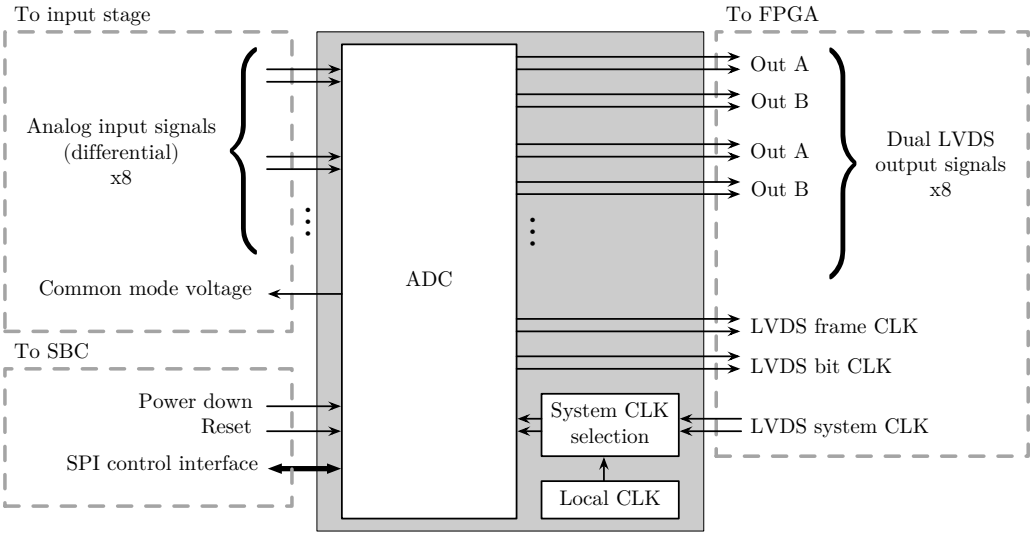


Figure 4.12: ADC stage diagram.

Analog input interface

Signals are input using two pins for each channel, since they are differential to achieve higher noise protection. The maximum differential input range is $2\,V_{pp}$ and common mode voltage must be 0.95 V, for which the ADC provides a 0.95 V reference to facilitate the job to the previous stage. Figure 4.1 of section 4.2.1 illustrates this.

The analog section of the ADC, however, is powered by a 1.8 V single-ended power supply, so input signals must never get above this value or below ground at the risk of damaging the device. To this matter, protection diodes have been installed at the 16 input pins.

Digital output interface

Sample data is output using serial LVDS interface lines to reduce the number of necessary pins and PCB traces. However, at 80 MS/s and 14-bit resolution this means transmitting data at 1.12 Gbit/s per channel, so each channel can use one or two LVDS lines to reduce the data rate per line and allow the FPGA device on the receiving end to operate at lower speeds.

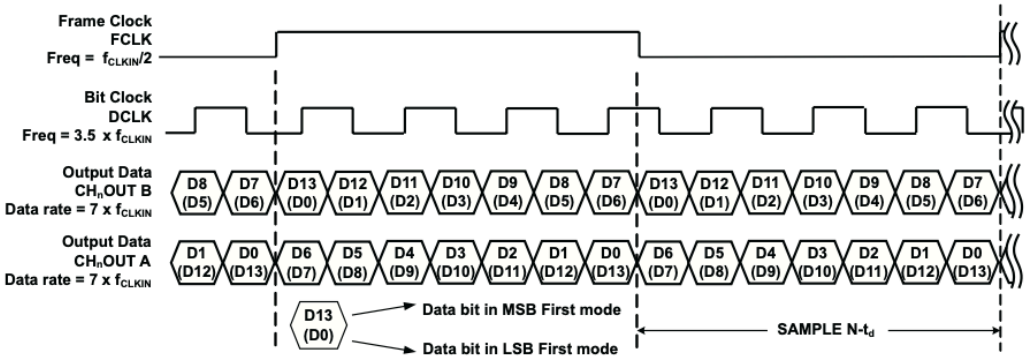


Figure 4.13: LVDS timing diagram for 2-wire operation. Image from [30].

In addition to the data lines the ADC outputs another two global synchronization signals, also through LVDS; a *frame clock* signal , which is synchronized with the start of each new sample, and a *bit clock* signal, synchronized with each bit of the data lines. Figure 4.13 illustrates this in a 2-wire operation example. The ADC, however, supports a variety of synchronization configurations and output modes.

ADC clock

The ADS5294 supports both single-ended and differential clocks up to 80 MHz. With flexibility in mind, in the ARACNE prototype three possibilities are available, with two of them selectable via a jumper: using an on-board discrete clock generator or a differential LVDS clock signal provided by the FPGA, which allows the usage of different sample rates without other hardware modifications. The third option is using an external clock, for which a connector is provided.

ADC control interface

The ADS5294 ADC can be controlled by means of a SPI interface, which requires 4 signals (chip select, two data input and output lines and a synchronization clock). Internally, the ADC exposes a set of 36 16-bit registers that can be written and read back to control and configure every function of the device. The communications protocol is thoroughly explained in the documentation [30, p. 37-39], as well as the register maps and the description of every register. Additionally, the device features two additional lines for resetting and powering down the device when needed.

The hardware prototype has been designed so that the on-board computer can both configure the ADC through the SPI interface and control the additional reset and power down lines. Additionally, headers for manually powering down or resetting the ADC for development purposes have been provided in the PCB.

Power supply requisites

The ADS5294 chip needs two 1.8 V power supplies, one for the analog section and another one for the digital part. As such, the same power supply could be used to drive both sections; but in order to reduce noise in the analog section, two separate supplies have been used.

Layout considerations

As mentioned in the previous sections a large number of differential signals both at the input and output of the ADC are being used in the design, with some of them operating at very high frequencies and requiring tight timing constraints. This poses several challenges when placing components in the PCB and routing signals between them. Differences in trace length between signals of the same

differential pair, or between differential pairs of different channels, can result in time skews that can lead to loss of precision or limit the operating speed.

For analog signals, different trace lengths in a differential pair leads to a time skew that will manifest as an error voltage in the differential signal. And differences between pair lengths of different channels will result on signals that will be sampled at different moments (actually, they will be sampled at the same time but one of the signals will be delayed with respect to the other due to the pair length differences).

With LVDS signals the problem will be the same, but the effects will be different due to its digital nature. In the case of differences in length in signals of the same pair, the eye pattern of the signal [31] will be degraded because of a reduction in the eye width, so that attainable speeds will be slower. And differences in pair length will result in synchronization problems, as bit clock between signals will vary. In our case there's a LVDS signal that acts as bit clock, as shown in figure 4.13, that tells the receiver when to sample the input to determine the bit values of all data LVDS signals. If there's a time skew between differential pairs, the data will not be sampled at the optimum moment (the middle of the bit duration) reducing again the maximum attainable transmission speed.

Signal propagation speed in a PCB depends on a variety of factors, but a figure around 15 cm/ns is a commonly accepted approximation. So, given the small distance between components of the prototype, the problems described above will hardly arise if PCB traces are laid out carefully; on the analog side, trace length differences in the few millimeters range would mean time skews in the range of picoseconds, which is well beyond the projected capabilities of the prototype. And regarding the LVDS signals, the highest data rate is 560 Mbit/s, which means bit durations slightly below 2 ns. Even though trace length equalization is more critical in this case, it seems unlikely that differences of a few millimeters can significantly degrade performance. Some detail of the manual routing of the LVDS signals of the prototype and the termination resistors is shown in figure 4.14.

However, since there can be other sources of signal degradation reducing the operating margin in both cases, great care has been put into trace equalization to guarantee the best signal quality on the analog side and the most precise timing on the digital signals.

Another concern regarding the routing and layout of high-speed signals is differential pair impedance and termination, as was discussed on section 2.3.3. According to the ADC documentation, a termination resistor of 100 Ω across the differential pair close to the FPGA is required, and keeping the characteristic

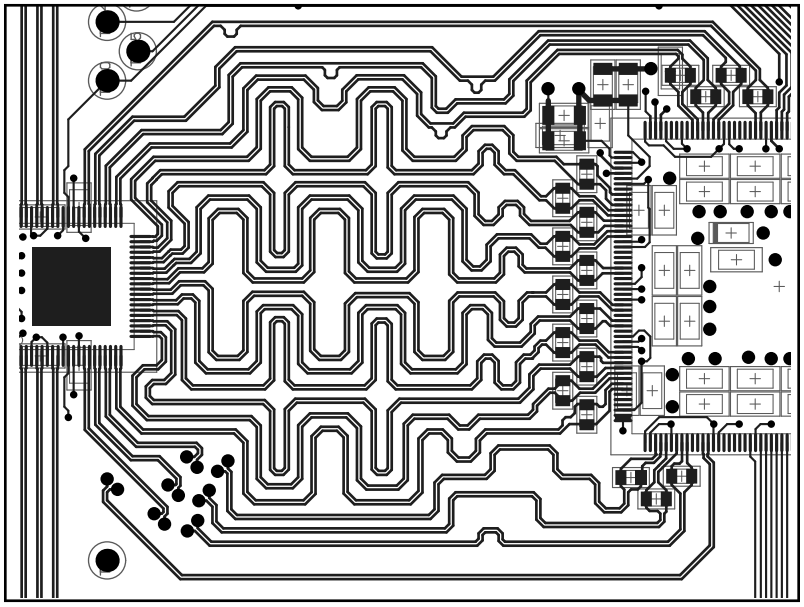


Figure 4.14: LVDS trace equalization and termination from the prototype PCB.

impedance of the transmission line at the same value is necessary to avoid signal degradation. This is a complex calculation that depends on many factors such as the PCB substrate type, layers thickness, trace width, separation between differential pair lines, etc. For this reason, the Saturn PCB Design Toolkit software has been used for the calculation, based on the characteristics of the PCB technology used by manufacturer EuroCircuits, which was used for the prototype production and assembly. The PCB was designed using 4 layers; two for signals and two for ground and power supplies. So, given that a FR4 standard substrate type is used and a separation between the signal and plane layer is 14 mil as shown in figure 4.15, a trace width of 8 mil with a trace separation of 4 mil seems to yield a differential impedance value close enough to the target 100 Ω, according to the software (see figure 4.16).

Other features

Apart from the main features described in previous sections, the ADS5294 ADC is an advanced device that in addition to analog to digital conversion provides several signal processing capabilities that can prove very useful.

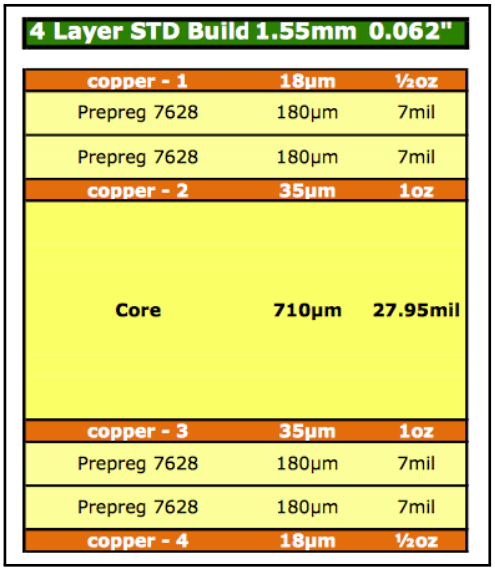


Figure 4.15: Eurocircuits PCB specifications for the ARACNE prototype.

- **Programmable FIR decimation filter**. The device is capable decimating the digital signal using a 23 or 24-tap FIR filter, either built-in (low-pass, high-pass and band-pass filters are provided) or using custom 12-bit coefficients set by the user. This way, the input signal is sampled at the predefined sample rate, but the output data rate can be reduced by a factor of 2, 4 or 8.

- **Programmable IIR high pass filter**. The device provides a first-order digital filter of this type that can be optionally used for example to remove a DC component in the signal.

- **Programmable mapping between inputs and outputs**. Even though not every analog input can be mapped to any LVDS output, there is a high degree of flexibility. Channels are grouped in two sets, from 1 to 4 and from 5 to 8. Any of the inputs of each set can be mapped to each of the outputs of the same set.

- Per-channel **programmable digital gain**. If the analog input is less than the $2\,V_{pp}$, a gain can be set to achieve the full scale output codes. Gains from 0 dB to 12 dB in 1 dB steps are supported.

- **Channel averaging**. The device is capable of automatically averaging two or four channels to improve the signal to noise ratio, and output the result through a predefined LVDS output. Not all channel combinations are

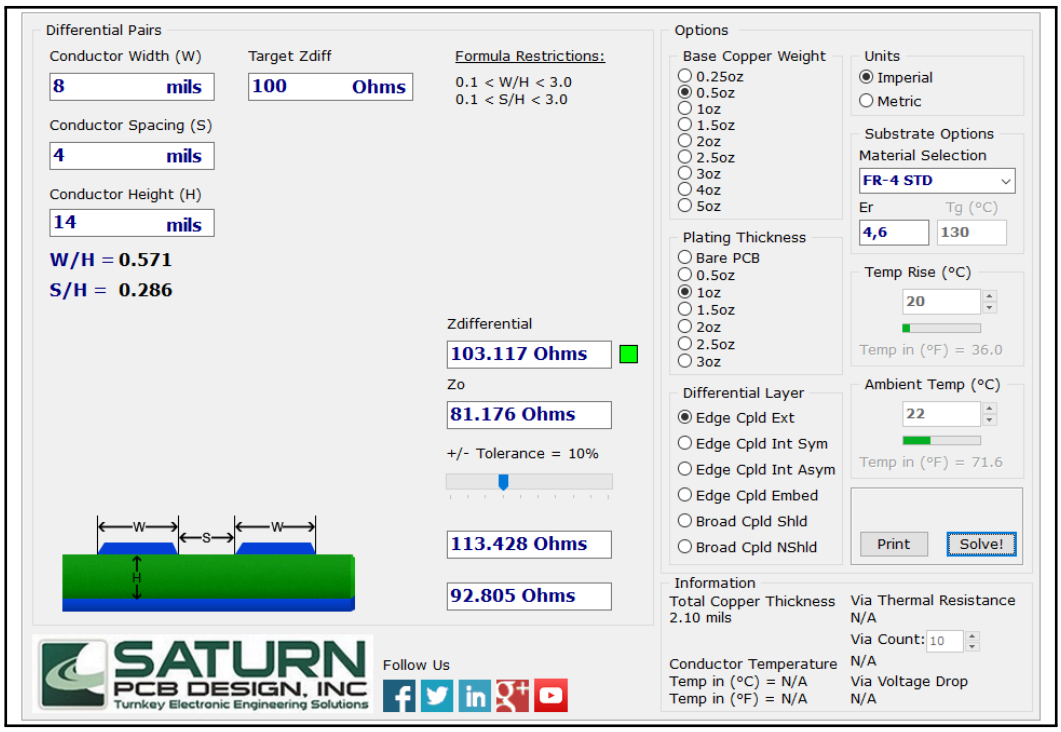


Figure 4.16: Results from the Saturn PCB Design Toolkit for the calculation of trace width and separation of the LVDS lines.

possible though, and the resulting average cannot be sent to any output. For example, for four channel averaging only channels 1-4 or 5-8 can be used, and they can only come out on the LVDS outputs 1 or 4 in the first case, and 5 or 8 in the second. The same way, for two channel averaging only specific adjacent channels and outputs can be used.

- **Test pattern generator**. The ADC is capable to automatically generate different signal patterns and transmit them through LVDS. This way, testing of the PCB, the IPCore or the software can be done without the need of input signals or a functioning input stage, considerably easing the development. Patterns supported include a single value, alternating between two values, a full-scale ramp and three presets for syncing and testing: a deskew pattern (outputs the 01010101010101 word), a sync pattern (outputs the 11111110000000 word) and a Pseudo-Random Binary Sequence (PRBS) pattern.

4.2.3 Programmable logic stage

This part of the hardware platform is where most of the adaptation and reconfiguration capability of the ARACNE proposal resides. Analog signals coming from detectors have been conditioned and digitized at high speed in the previous stages, and must now be used to provide the required experimental data using a digital acquisition chain that is implemented in a reprogrammable logic device, and that can be changed and configured without the need of hardware changes.

At the heart of this stage is an FPGA device, which as discussed in section 3.4.3 must be chosen with several system requisites in mind. But once the device is chosen, thorough study of its architecture and resource usage is needed in order to sensibly assign its inputs and outputs to the many signals present in the hardware platform. Failure to do so would result in a PCB that is unfit for the task after it's been already produced an assembled, or the impossibility to place certain IPCore resources as intended.

A summary of the signals that must be dealt with by the FPGA, the choice of specific device and a discussion regarding its architecture and how the signals have been assigned consequently, is outlined in this section.

Input and output signals

Three general types of signals are managed by the FPGA, apart from the Joint Test Action Group (JTAG) programming interface: data and clock signals to and from the ADC, control and communications signals from the SBC, and miscellaneous I/O signals for the user interface and optional functionality, as shown in figure 4.17.

- The **interface with the ADC** consists of 19 LVDS pairs as described in section 4.2.2:
  - 16 links (two for each channel) at up to 560 Mbit/s each, producing an aggregate data flow of up to 8.96 Gbit/s at full speed that must be dealt with by the FPGA.
  - 2 links carry frame clock and bit clock signals, the first one signaling the beginning of a new sample in all channels, and the second signaling the availability of a new bit in all data lines. These signals are of Double Data Rate (DDR) type (that is, both the rising and the falling edges of the signal are used), so their frequencies are half of the sample rate and bit rate, respectively.

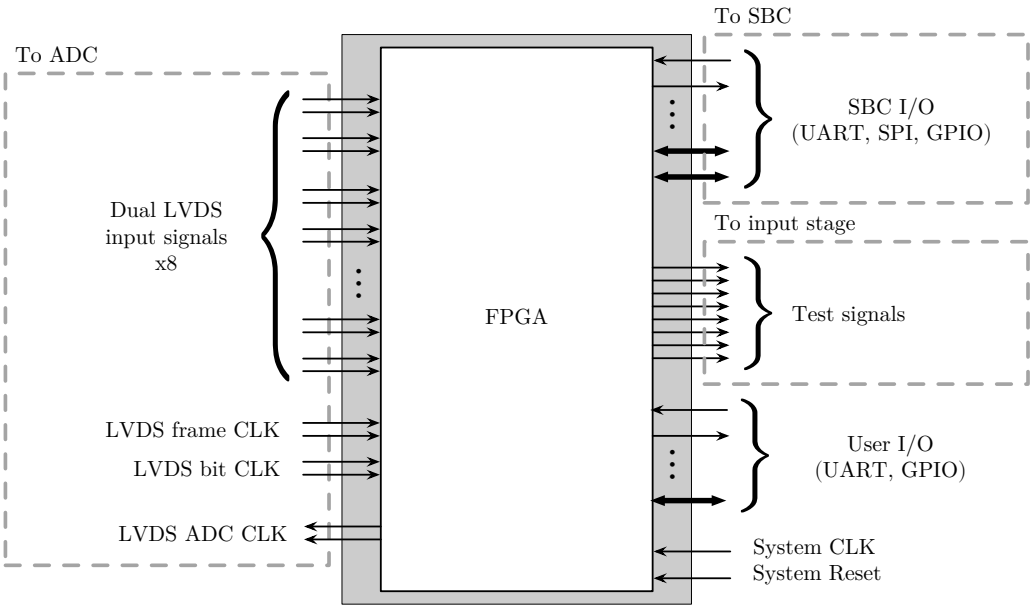


Figure 4.17: FPGA stage diagram.

  - 1 link from the FPGA to the ADC that provides a synthesized system clock, so that the sample rate can be changed depending on the application.

- The **interface with the SBC** consists of a variety of signal types, so that different communications and control options can be implemented on the IPCore depending on the necessity. Naturally, the possible signal types depend on the choice of SBC and will be described in section 4.2.4. In the current prototype, the following interfaces are available among the devices (not all of them simultaneously):

  - 2 bidirectional SPI interfaces (one of them shared with the ADC).
  - 4 bidirectional UART interfaces.
  - 12 I/O signals that are connected directly to one of the Programmable Real-time Units (PRUs) present in the SBC SoC. As will be explained later, these PRUs are 32-bit processors that are independent of the main processor and can be used as co-processors for high-speed communications and other real-time tasks. Additionally, another 5 I/O signals are connected to the second PRU.
  - 10 additional GPIO signals that can be used for a variety of purposes. It should be noted though that many of the I/O signals provided by

the SBC and mentioned in the above interfaces can be reconfigured and used as GPIO, so the total number of GPIO signals possible can be much larger.

- Finally, there's **miscellaneous I/O** as was described in section 4.1.1 that includes signals for the following functionality:
  - System clock and system reset signals.
  - 8 test outputs, so that the FPGA can produce programmable test pulses that can be used at the input stage.
  - User I/O, consisting of 8 inputs wired to a bank of jumpers, 4 inputs wired to as many push switches and 8 I/Os that can be used either as outputs or inputs, and are wired to 8 Light-Emmitting Diodes (LEDs).
  - 1 UART interface for debugging and control purposes.

It should be noted that regardless of the interface type and utility, the voltage standard used in each case is important as it will dictate constraints regarding the signal placement and the power supplies required by the FPGA. In this prototype, it will be necessary to deal with LVDS signals and Low Voltage Transistor to Transistor Logic (LVTTL) 3.3 V logic, which is what the SBC needs.

FPGA family and model

Beyond the basic requisites for the project, such as capability to interface with the required system components, speed and capacity to implement the required synthesizable hardware for the digital acquisition chain, the choice of FPGA family and model for the prototype was determined by a number of additional factors.

To begin with, it was important to make sure the resulting PCB would be as simple as possible, something very difficult with packaging solutions for large FPGAs, that are usually based on gridded pin distributions, such as Ball Grid Array (BGA) or one of its variants. These usually require 6 or more PCB layers for routing and very high precision traces, and are almost impossible to solder without highly specialized equipment. Moreover, laboratory testing and probing is very difficult.

In second place, availability of software tools for IPCore development that are free or low cost is also an issue. It's frequent that FPGA manufacturers offer freely available tools for IPCore development, but not all FPGA devices are supported; normally, more powerful models require the use of the paid versions of these tools.

Finally, low power is one of the goals of this prototype so a flash-based solution was preferred. These solutions also provide instant-on capability and are single chip solutions, whereas SRAM based FPGAs usually require external Read Only Memory (ROM) chips to store the FPGA configuration and some additional electronic components. They also have some configuration delay at power on.

With all these considerations in mind, at the time the prototype was being projected it seemed that the ProASIC3 family of devices from Microsemi was the best option, and more specifically the A3PE1500 model fulfilled all the important requisites:

- It's a flash-based, single chip solution that uses very little power and works at low 1.5 V for the core.
- A model fulfilling the I/O requisites with a design-friendly PQ208 Plastic Quad Flat Pack (PQFP) packaging is available
- It's got 1.5 million equivalent system gates, which allows large designs to be synthesized and loaded. Moreover, there's two more devices that use the same packaging with higher and lower capacities (3 and 0.6 million system gates), and if selection of used I/O pins is done carefully, they can be used in the same PCB.
- Operation speed of up to 350 MHz is enough to support the high data rates expected from the ADC.
- It supports LVDS I/O up to 700 Mbit/s using DDR signaling, so it's compatible with the kind of signals used by the ADC. With the PQ208 packaging, 65 differential pairs are available, more than enough to support the 19 LVDS signals required.
- It supports a variety of other single-ended and differential I/O standards that can be used simultaneously. The device is divided in 8 I/O banks, and a different voltage for each bank (that defines what I/O standards are supported) can be set individually. This, of course, requires careful planning; but even using separate banks for LVDS and LVTTL, there's enough I/O pins to accommodate all the required signals.
- Dedicated RAM is present on the device, with the A3PE1500 model supporting about 270 kbit organized in sixty 4.608 kbit blocks. These can be used to implement several FIFO or SRAM configurations at up to 350 MHz operation.

- It includes 2 PLLs for clock generation, which is enough for an internal system clock and possible use with the ADC bit clock.
- Even though it's a large FPGA, it's supported by the freely available versions of the Libero design software provided by the manufacturer.

Signal assignment considerations

An FPGA device can be deceivingly perceived, at first sight, as a completely flexible and uniform programmable logic pool, in the sense that it doesn't matter to which pins does the PCB designer assign signals, independently of their type and function. It could be assumed that as long as the design fits in the device, the place and route software tools will always be able to configure the logic in such a way that it will be possible to implement any design regardless of the pin assignment choices. This, however, is far from the truth, and correct pin assignment while designing the PCB requires thorough study of the specific FPGA device and its architecture. Failure to do so could result in unimplementable designs, or designs that are limited in speed. As will be explained in the following sections, several factors need to be taken into account:

- I/O pins in an FPGA are usually organized into *banks* that have independent supply voltages, and I/O standards that use very different voltage levels (i.e. LVDS and LVTTL) often can't be used in the same bank.
- Some FPGA resources (such as a specific PLL block) can only be directly accessed from specific I/O pins.
- Global networks, used to distribute clocks, resets or other high fanout nets that require minimum skew, are limited in number. This could be not as obvious as the limitation in other resources, such as the number of RAM blocks or PLLs.
- These global networks can sometimes only be directly accessed from specific I/O pins. Moreover, accessing them from one of the possible pins sometimes rule out others.

As such, an analysis of the signals that are going to be used, their I/O standard and their functionality, like it has been previously done in this section, is a must. And performing a sensible pin assignment depending in the aforementioned factors by studying the specific FPGA device is a necessity. In the specific case of the hardware platform prototype, the following are the most critical aspects to be taken into account:

- Two main logic standards are used, LVDS and LVTTL, so signal distribution between I/O banks will be needed. Additionally, the ADC works using LVCMOS 1.8 V logic, so if interfacing it via the FPGA is considered a possibility, it must be taken into account.

- Two signals from the ADC are critical since they are used for synchronization, and will likely be used by a large amount of logic: the *bit clock* and *frame clock* LVDS signals. These signals should be assigned to pins capable of accessing global networks.

- All FPGA devices need at least one external clock signal, which is used to synthesize the necessary system clocks for the IPCores to work using one of the integrated PLLs. As such, it should be assigned to a pin capable of accessing one PLL block.

- Finally, all IPCore designs typically include a *system reset* signal, so an input in the FPGA will have this function. Since it's a signal that will need to reach all the logic in the device, it should be assigned to a pin with access to a global net.

The rest of I/O signals described in this section don't have, a priori, such constraints. However, pin assignment taking into account the distribution of components in the PCB and their pinout distribution has been done to simplify the trace routing.

FPGA architecture and resources

The A3PE1500 FPGA model is available in several packaging formats from 208 to 676 pins. This means there are slight differences in the available resources and most notably, the available I/O pins. Table 4.1 summarizes the pin functionality of the PQ208 package used in the prototype.

Pin function	Number of pins
Ground, supply voltages and reference voltages	53
JTAG interface and supply voltages	7
Not connected	1
I/O	147
TOTAL	208

Table 4.1: A3PE1500 pin function summary for the PQ208 packaging.

It must be noted that while all I/O pins can be used for single-ended signals, some of them can't be used for differential I/O. Also differential I/O signals must be assigned to specific pin pairs where the positive and negative pin is predefined.

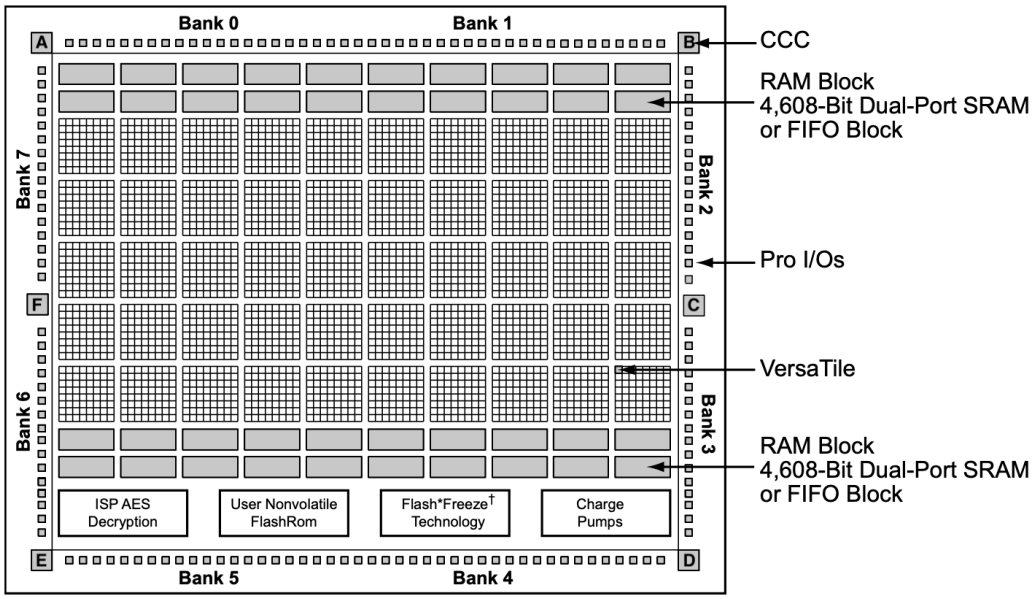


Figure 4.18: ProASIC 3 A3PE1500 architecture diagram, from [32].

Something of great importance is that beyond supporting single-ended or differential signals, *not all I/O pins can access the same FPGA resources directly.* It depends on their position and is a consequence of the FPGA architecture. Without entering into much detail, the architecture and basic resources that conform the A3PE1500 FPGA are represented in figure 4.18, taken from [32]. The basic elements in the diagram relevant for the discussion are:

- The configurable logic fabric, made up of *VersaTiles*, which are the basic configurable logic blocks.
- RAM blocks.
- Clock Conditioning Circuits (CCCs). There are six of them, four on the corners of the device and two on the East and West sides. As will be discussed later, only the last two include a PLL.
- I/O pins, which are divided in 8 banks, 2 per side of the device.

I/O banks distribution

The first important factor to understand the signal assignment restrictions is that I/O pins are divided among 8 banks, numbered 0-7, each of which can be configured to support a different set of I/O standards depending on the supply voltage provided to the bank. Table 4.2 summarizes the number and type of I/O pins per bank. It must be noted that the table indicates the maximum I/O signals of each type, and obviously if two pins are used for a differential pair, it will mean two less available pins for single-ended I/O.

Bank number	Single-ended I/Os	Differential I/Os
0	25	12
1	15	7
2	17	6
3	16	7
4	15	7
5	22	10
6	19	9
7	18	7
TOTAL	147	65

Table 4.2: A3PE1500 pin distribution summary for the PQ208 packaging.

In addition to this, a thorough comparison with the I/O pin capabilities of the other two pin-compatible FPGA models, the A3PE600 and A3PE3000 has been performed. The conclusion of this study has determined that even though all the I/O pins are the same and consequentially all three models are interchangeable for single-ended signals, there are differences in some of the differential pair pin assignments according to the documentation [33, p. 100-106]. In conclusion, if the PCB is designed with model compatibility in mind, these differential pairs should be avoided and the table of compatible I/Os is restricted to the signal summary of table 4.3. As a consequence, at least three I/O banks will need to be configured for LVDS I/O in order to accommodate the 19 signals of this type that are planned.

Clock conditioning resources

The second thing that can be noticed from figure 4.18 is the positions of the six CCC blocks at the corners and center of the East and West sides of the device, labeled A through F. I/O signals requiring access to the input of these clock conditioning blocks, and more specifically to the ones needing access to a

Bank number	Single-ended I/Os	Differential I/Os
0	25	8
1	15	6
2	17	6
3	16	7
4	15	4
5	22	5
6	19	9
7	18	6
TOTAL	147	51

Table 4.3: Compatible signals distribution for the A3PE600/1500/3000 FPGAs with PQ208 packaging.

PLL, will need to be assigned to pins close to the corresponding block as will be described later. This will also restrict signals to specific I/O banks (e.g. the PLLs will only be accessible from banks 2, 3, 6 and 7, which are adjacent to positions C and F).

Global network resources

And finally, not represented in the diagram are the networks and routing elements that interconnect the FPGA resources. Most importantly, the *global networks* that, as mentioned before, provide high-speed, low-skew and high-fanout interconnection. They are scarce and not accessible from all I/O pins, which will further restrict the signal assignment.

The device is divided in four *quadrants* and global nets are distributed around the chip, in the manufacturer's terms, using *ribs* and *spines* (East-West and North-South lines) that are accessible from the VersaTiles. There are two types of global nets; *chip* global networks and *quadrant* global networks.

- There are 6 chip global networks formed by a rib in the center of the device, running from West to East, and spines that reach all the VersaTiles. As such, these networks are only accessible by I/O pins near the C and F positions of the device, 3 from each side.
- Quadrant global networks are formed by a rib starting at each corner of the device, and spines reaching only the VersaTiles in the quadrant of the corresponding corner. There are 3 networks per quadrant (12 in total) and only VersaTiles of the same quadrant can be connected using these networks.

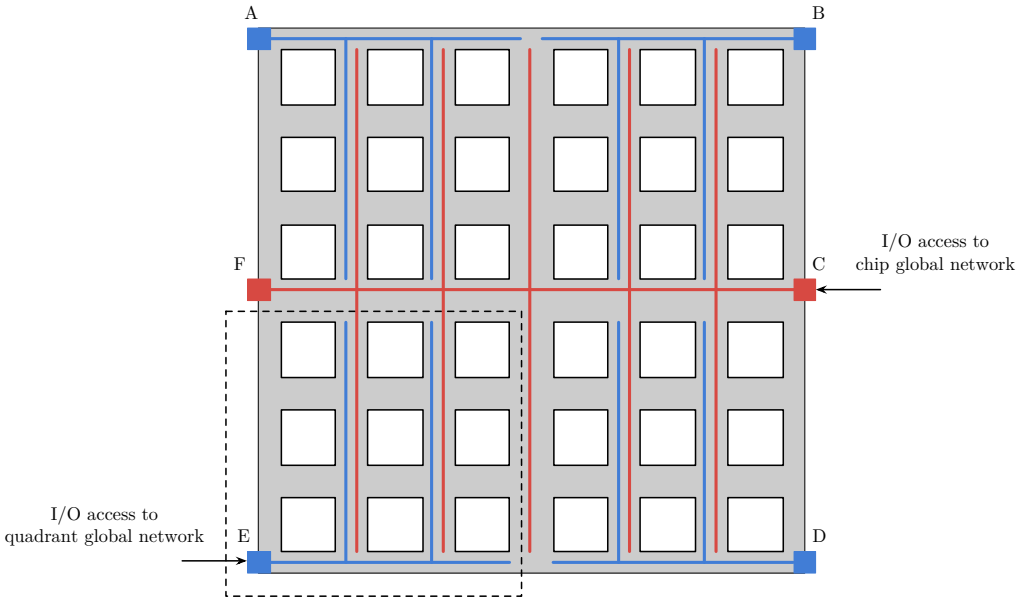


Figure 4.19: ProASIC 3 A3PE1500 simplified global networks diagram. Only one chip global network (red) and four quadrant networks (blue) are represented.

Also, they are accessible only by I/O pins near each corner of the device, at locations A, B, D and E.

As a result, there is a total of 18 global networks in the device but each VersaTile has access to only 9; the 6 chip global networks and the 3 global networks of its quadrant. Figure 4.19 better outlines this distribution and how the global networks can be connected to I/O pins. It can be concluded that external signals that must reach global networks are severely constrained in their I/O pin assignment, even more so if the chip global networks need to be accessed.

In summary, there are 6 locations of the device that are special both for CCC and global network direct access. The four corner locations A, B, D and E allow access to 3 quadrant global networks and a regular CCC each. And the locations at the center of the East and West sides, C and F, each give access to 3 chip global networks and CCC that includes a PLL.

In the prototype, four signals should potentially reach more than one quadrant, if not all the device, or access a PLL. Two of them are LVDS (the bit clock and frame clock from the ADC), and two are LVTTL (the system reset and the input clock). This means 4 out of the 6 available chip global networks will be already used, and since they use different I/O standards, they will need to placed at

locations C and F in different I/O banks.

Global I/Os

The final piece of the puzzle is how to access the global resources of each of the A to F locations from the corresponding I/O pins. To this matter, there's three multiplexer structures (named A, B and C) such as the one represented in figure 4.20 at each location. Each one allows access to the corresponding GLA, GLB or GLC global network of the location, through the CCC. Each multiplexor supports three external global I/O pins, 0, 1 and 2, whose naming convention is Gnmx; n is the location, m the multiplexer and x is the input number. The example in figure 4.20 represents multiplexer A of location B, so the three input pins it supports are named GBA0, GBA1 and GBA2, and will drive the CLKA input of the CCC, which in turn will provide access to the GLA global net.

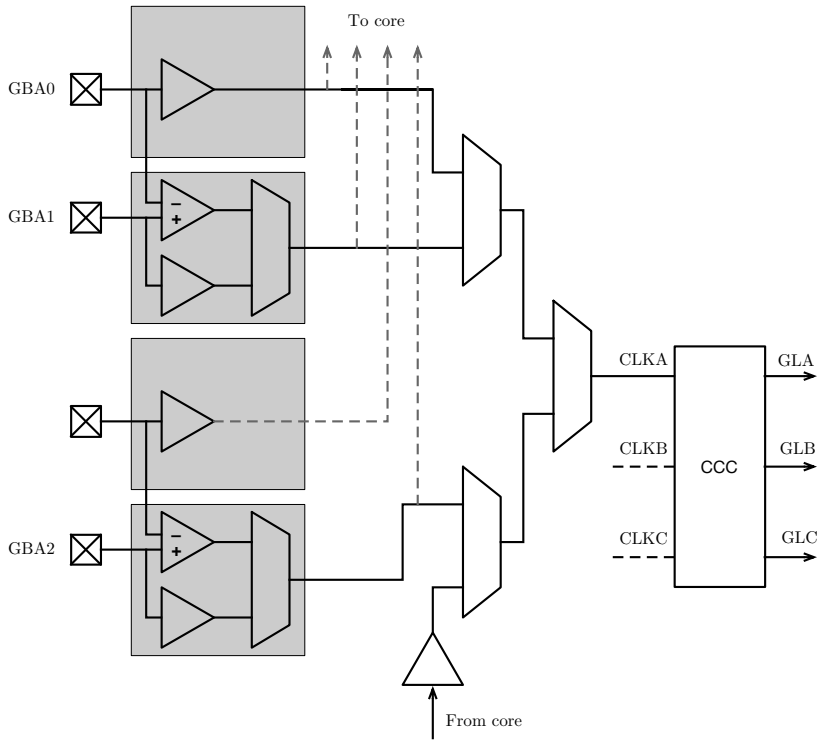


Figure 4.20: ProASIC 3 A3PE1500 global I/O structure.

The global I/O multiplexer structure allows direct access from one of the three global I/O pins to the corresponding global resource in single-ended configuration;

or to the pair formed by global inputs 0 and 1 in differential mode. Global I/O input 2 can also be used as differential input in conjunction with its corresponding regular I/O pin. Additionally, any of the global pins can be used as conventional I/O (routed to the core), and a signal from the core can also be routed from the global resources.

It is important to note, though, that *only one input can have access to the global resource at any given time*. For example, if global network GLA at location B is being accessed through the differential pair formed by inputs GBA0 and GBA1, it cannot be accessed using GBA2. As a result, 54 global input pins are present in the A3PE1500 (except for two of them that are missing in the PQ208 package), but only 18 input signals can access the 18 global networks.

Finally, there's one important detail that can go unnoticed. Each CCC input CLKA, CLKB and CLKC can be accessed though the corresponding GmAx, GmBx and GmCx global I/Os, but if the user needs access to the PLL of a CCC, it must be *necessarily* accessed through the CLKA input, and consequently the global I/O pins GmAx must be used.

Signal assignment in the prototype

Taking all the above into account, the signals of figure 4.17 have been assigned according to its constraints. First of all, the signals needing access to global resources are assigned according to table 4.4.

Signal name	Description	I/O Std.	Global	I/O	Bank
SYSCLK	External clock	LVTTL	Chip (PLL)	GFA0	6
SYSRESET	External reset	LVTTL	Chip	GFB2	6
LCLK	ADC bit clock	LVDS	Chip (PLL)	GCA2	3
ADCLK	ADC frame clock	LVDS	Chip	GCB0	2

Table 4.4: Assignment of critical signals. For differential pairs, only one of the used global I/Os is included since the other member of the pair is mandatory.

This mandates I/O banks 2 and 3 to use a 2.5 V supply, so they will support LVDS signals, and bank 6 to use a 3.3 V supply for LVTTL support. With this in mind, the rest of the LVDS signals can be assigned. The specific I/O pairs they use are not important since they don't need access to global resources, so I/O pairs are chosen as specified in table 4.5 to ease signal routing, keeping signals laid out as they exit the ADS5294 ADC. Since all signals don't fit in banks 2 and 3, banks 1 and 4 must be used.

Signal name	Description	Bank
CLK_P	ADC clock	1
CH1_OUT[1,2]	Channel 1 pairs 1 and 2	1
CH2_OUT1	Channel 2 pair 1	1
CH2_OUT2	Channel 2 pair 2	2
CH3_OUT[1,2]	Channel 3 pairs 1 and 2	2
CH4_OUT[1,2]	Channel 4 pairs 1 and 2	2
CH5_OUT[1,2]	Channel 5 pairs 1 and 2	3
CH6_OUT[1,2]	Channel 6 pairs 1 and 2	3
CH7_OUT[1,2]	Channel 7 pairs 1 and 2	4
CH8_OUT[1,2]	Channel 8 pairs 1 and 2	4

Table 4.5: Assignment of differential signals.

The rest of the signals are assigned to banks 5, 6, 7 and 0 distributed also to ease routing. Table 4.6 summarizes this.

Signal name	Description	Bank
TESTn	Test pulse outputs	5
JMPn, SWn, IOn	FPGA User I/O	5, 0
FPGA[SCLK,PD,CSZ,SDATA,RESET ADC,SDOUT]	FPGA ADC out I/F	5
ADC_[RESET,PD], SPI1*	SBC ADC in I/F, SPI1	6
SPI0*	SBC SPI0	6
UART1_[TXD,RXD]	SBC UART 1	6
UART4_[TXD,RXD]	SBC UART 4	7
UART5_[TXD,RXD]	SBC UART 5	0
FPGA_UART_[TXD,RXD]	FPA UART	0
PRU1n	SBC PRU1 signals	7, 0
All other GPIO signals	SBC Misc. I/O	6, 7, 0

Table 4.6: Assignment of single-ended signals.

As a result of this signal distribution, banks 1, 2, 3 and 4 will require a 2.5 V power supply and banks 5, 6, 7 and 0 will require 3.3 V. Optionally, since the ADC control signals can be controlled by the FPGA and require 1.8 V logic, bank 5 can be configured to use either 3.3 V or 1.8 V.

4.2.4 Embedded computer

The final piece of the hardware platform, as described in section 3.3.1, is the embedded computer. Its mission, preliminary requisites and the rationale behind the decision of directly connecting it to the hardware platform are discussed in

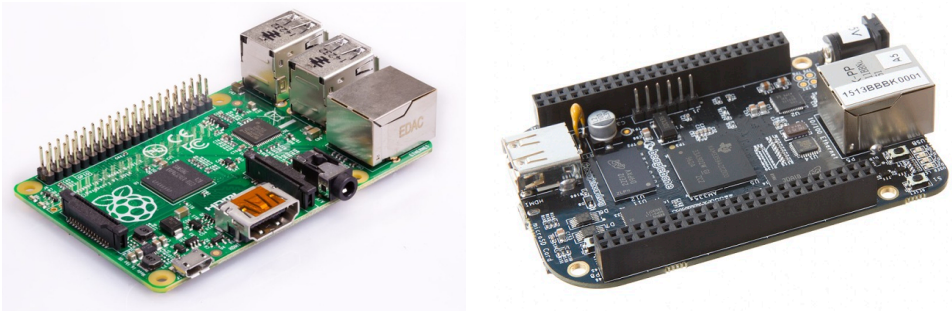
section 3.4.4. In this section, a description of the chosen platform, its interfaces and the requisites for integration with the rest of the system is provided.

SBC choice

One of the circumstances of the current state of technology that the ARACNE proposal leverages is the recent availability of inexpensive, compact size and high performance embedded computers that have a low power consumption and are capable of running an advanced operating system such as Linux, together with virtually any type of software.

This was made possible by two technological factors. First, the development of SoCs that include in one chip not only the CPU, but also a vast array of computer elements and peripherals that used to be discrete, allowing the creation of whole Single-Board Computers in very tiny formats. And second, the consolidation of the GNU/Linux environment as a complete, robust and well supported software platform.

As a consequence, platforms like Raspberry Pi [34] or Beaglebone [35] appeared in the last decade and have been evolving ever since, turning into more and more sophisticated SBCs. Their last versions feature quad-core 64-bit CPUs with clock speeds exceeding 1 GHz, several gigabytes of RAM, multiple wired and wireless connectivity solutions and even special-purpose co-processors. Still, they are credit-card sized and inexpensive so they seem a viable solution for many embedded applications.



(a) **(b)**

Figure 4.21: Raspberry Pi 1 Model B+ (left) and BeagleBone Black (right). Pictures from [34] and [35].

When the ARACNE platform was being devised, these two were the platforms that seemed more adequate and best supported for the task of fitting the hardware

platform with a suitable SBC. The models available at the moment were on one hand the Beaglebone Black (figure 4.21b) and on the other the Raspberry Pi 1 Model B+ (figure 4.21a). Key factors to evaluate were

- Performance to adequately run the necessary software.
- I/O capabilities to interact with the ADC, FPGA and the rest of system elements.
- Software support (e.g. the GNU/Linux environment and the necessary software).

Both platforms seem very similar at first sight; they are both ARM-based, feature similar amounts of storage and RAM, include header connectors for several types of I/O and run GNU/Linux, so both of them would have fitted the bill in terms of performance and software support. However, evident differences in their I/O capabilities show that they are aimed at different applications. It must be noted that it's not only a matter of the SoC featuring a specific I/O resource, but also that this resource is made available to the user through the I/O connectors on the SBC.

The header connector on the RaspBerry Pi offers 1 UART, 2 SPI and 1 Inter-Integrated Circuit (I2C) serial interfaces, plus 28 GPIO pins. These signals are multiplexed, so the pins must alternatively be used for one function or another (i.e., SPI0 channel takes 5 GPIO pins).

On the other hand, the BeagleBone Black features two 46-pin header connectors plus an additional 6-pin serial debug header (which, in the case of the Raspberry Pi, can only be implemented with the only available UART interface). The header connectors include 7 analog inputs (the SoC includes an on-board ADC) and 65 digital I/O pins for which 8 possible modes can be selected, including GPIO. This allows great flexibility and I/O possibilities, including 5 additional UARTs (one of them output only), 2 SPI channels and 2 I2C. Also, the BeagleBone Black SoC includes 2 real-time programmable coprocessors (named PRU) that are independent from the ARM Central Processing Unit (CPU), and that also have access to several I/O pins. This allows the implementation of advanced processing from external peripherals.

As a result, it can be concluded that the Raspberry Pi seemed more convenient for building multimedia systems or small servers, while the Beaglebone seems clearly more focused at embedded systems given the available I/O and the existence of the 2 PRUs and on-board ADC. Additionally, the CPU performance

was higher compared to the Raspberry Pi model available at the time, so the BeagleBone Black was chosen as SBC for the ARACNE prototype, and two 46-pin connectors were included in the prototype so that it could be directly plugged onto the PCB.

Interface with the PCB

The interfaces among the SBC and the rest of the elements of the hardware platform have already been mentioned and are depicted in the diagrams of the input stage (figure 4.2), ADC stage (figure 4.12) and FPGA stage (figure 4.17). The remaining I/O signals available in the SBC header connector have been used to implement some additional features, as described in section 4.1.1. Finally, a few system signals on the SBC connector must be taken into account in order for the interface among the PCB and the SBC to be complete. Figure 4.22 provides a basic diagram of the above.

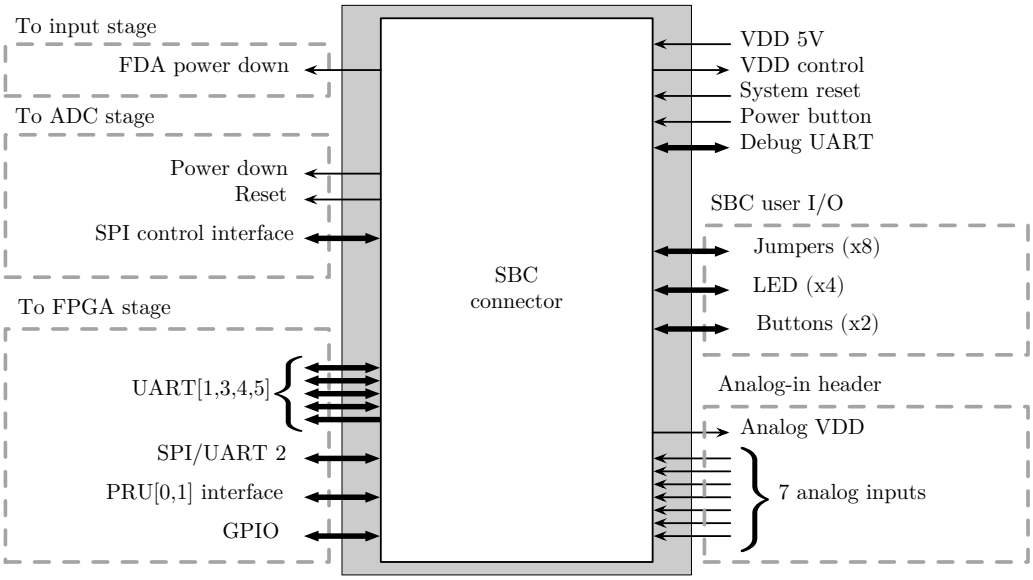


Figure 4.22: SBC I/O signals diagram.

This signal organization allows the SBC and the IPCore implemented in the FPGA to communicate in several ways; via an SPI channel, one or more UARTs, using the PRUs or simply by GPIO. However, the figure deserves a few clarifications:

- **FPGA I/O**. Not all I/O signals between the FPGA and the SBC depicted

in the figure can be used simultaneously, since many pins have multiple functions and must be configured for one function or another. For example, the UART5 signals can be used alternatively for GPIO, or the SPI lines are shared with the UART2 device. Despite this limitations, a large number of GPIO signals can be used simultaneously with the SPI communications interface and several UART devices.

- **ADC control**. The signals that control the ADC are also routed to the FPGA. From here, the signals are output to a jumper bank that allows the user to select if ADC control is to be performed directly from the SBC or through the FPGA. This allows two things: first, the ADC may be controlled by the FPGA. And second, the SPI channel used to control the ADC can also be used to control an additional SPI device implemented in the FPGA, since the channel supports two chip select signals. Figure 4.23 illustrates the concept. Depending on the configuration, the user must also select a different 1.8 V supply for I/O bank 5 of the FPGA and install a resistor, since that is the operating voltage of the ADC.

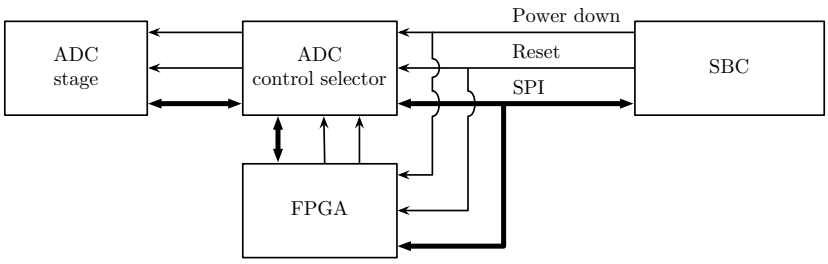


Figure 4.23: ADC control selector diagram.

- **Shared user I/O**. The LED and switch signals for SBC user I/O are also routed to the FPGA, so they can be used by either device alternatively.

- **Unaccessible physical resources**. The system reset and power button signals can be provided or accepted by the SBC, and are activated by on-board buttons that can be seen on the right-hand side of picture 4.21b. However those buttons are unaccessible once the board is plugged onto the main PCB, so they have been implemented in the main board too. The same thing happens with the debug UART header, which has been replicated to allow its usage.

SBC power supply and power-up considerations

There's two important issues worth mentioning regarding system power-up, related to the Beaglebone Black (BBB) SBC.

First, the Beaglebone Black manufacturer is very explicit about the damage that can occur if any signal is input into the board when it's not powered, and more explicitly, before the VDD control pin shown in figure 4.22 is high [36]. For this reason this signal has been used on the PCB design to control the on-board power supplies, so that no component on the PCB is powered on before this condition is met and ensuring the BBB always is powered first. However, the design also contemplates the PCB working without a BBB attached to it for debug purposes, so power supplies can be enabled via a jumper too.

And second, the BBB can be powered independently or by the PCB, via the 5V VDD pin. The design allows both possibilities.

4.2.5 Power supplies

Given the variety of devices used in the hardware platform and their different power supply needs, careful design of a power supplies section is necessary. Two were the main goals when approaching this task; providing a solution that only required a simple 5 V external power supply and offering the possibility to bypass any of the on-board power supplies and replace it with an external one.

Table 4.7 summarizes the needed voltages in the prototype.

Device	Description	Voltage
FDAs	Positive rail	3.3 V
FDAs	Negative rail	0 V
ADC	Digital supply	1.8 V
ADC	Analog supply	1.8 V
FPGA	Core supply	1.5 V
FPGA	2.5 V I/O supply	2.5 V
All PCB	Global supply and FPGA 3.3 V I/O supply	3.3 V

Table 4.7: Summary of power supplies needed on the prototype.

Supplies for the Fully Differential Amplifiers (FDAs) are apart from the rest to avoid noise problems. Also, in the default prototype configuration the negative rail is tied to ground, which is enough to comply with the input signal requisites. However, the possibility to install an external negative supply is provided and

requires only the addition of a few components. Similarly, the supplies for the analog and digital part of the ADC are separate despite they are of the same voltage value. On the other hand, the same 3.3 V power supply is used both for general use on the PCB and the FPGA banks configured for LVTTL I/O.

All supplies are implemented using Low Dropout Regulators (LDOs) for low noise, powered by the 5 V external supply after a protection circuit. All of them support an external enable input that can be controlled from the BBB. A Texas Instruments TPS75215 fixed 1.5 V, 2 A device is used for the FPGA core, and several Linear Technology LT3080 adjustable voltage, 1.1 A LDOs are used for all other supplies, for simplicity. The output of this regulator can be configured with a single external resistor.

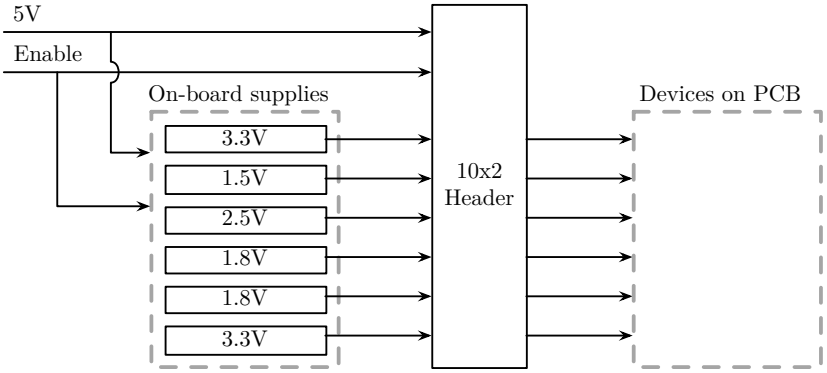


Figure 4.24: Supply header diagram.

As mentioned before, any of these supplies can be bypassed by an external power supply in case the need arises. This is achieved by using a 2 by 10 pin header where a jumper must be installed for a supply to be connected to the rest of the PCB, but can be replaced by an external supply, as shown in figure 4.24. The header also provides the required enable signal and access to the 5 V rail, so a daughtercard with the new supplies can be easily built.

4.2.6 Expansion and user interfaces

Finally, the PCB has been fitted with some additional expansion and user interfaces, both to take advantage of hardware resources that were not used otherwise and to provide some physical interface for the user. This can be useful both when developing and debugging the IPCore and the software, and as a mean of configuration and monitoring.

- **Analog input header**. It allows 7 analog inputs to the BBB, while providing both a 3.3 V supply, a 1.8 V reference and an enable signal to avoid damage to the BBB.
- **Two digital jumper headers connected to FPGA I/O**. 16 FPGA I/O signals are routed to two jumper banks, where they can be treated as inputs or outputs. One of the banks includes LEDs.
- **Four push button switches connected to FPGA I/O**.
- **Two push button switches connected to FPGA and BBB I/O**.
- **Four LEDs connected to FPGA and BBB I/O**. The user should be careful to configure only either device as output to avoid damage.
- **A jumper header connected to BBB I/O**. 8 BBB I/O signals are routed to a jumper bank, where they can be treated as inputs or outputs.
- **A three-pin header connected to FPGA I/O**, to allow the implementation of a LVTTL UART debug interface.

4.2.7 Resulting hardware platform prototype

The resulting prototype is a 160 mm by 238.5 mm, 4 layer PCB with all the features described along this section. The card is slightly higher than a standard 6U EuroCard (160 mm by 233.5 mm and connectors are not on the wide side, but since this was a prototype it was never designed to be rack-mounted but to be installed stand-alone and have accessible user I/O. A future prototype could be designed to be rack-mounted from the start. Figures 4.25 and 4.26 show the top and bottom sides, with the Beaglebone Black (BBB) SBC already mounted and visible in the latter. Bayonet Neill-Concelman (BNC) connectors were used in this unit.

4.3 Synthesizable hardware

The design of the synthesizable hardware loaded in the FPGA, or IPCore, has been done in accordance to the proposal outlined in section 3.3.2, and is composed of the modules shown in figure 3.4. As such, the main elements of the IPCore are a deserializing input stage, a communications interface to the BBB SBC, a set of modules performing the digital data processing functions and a system controller module that allows control and configuration from the SBC. General

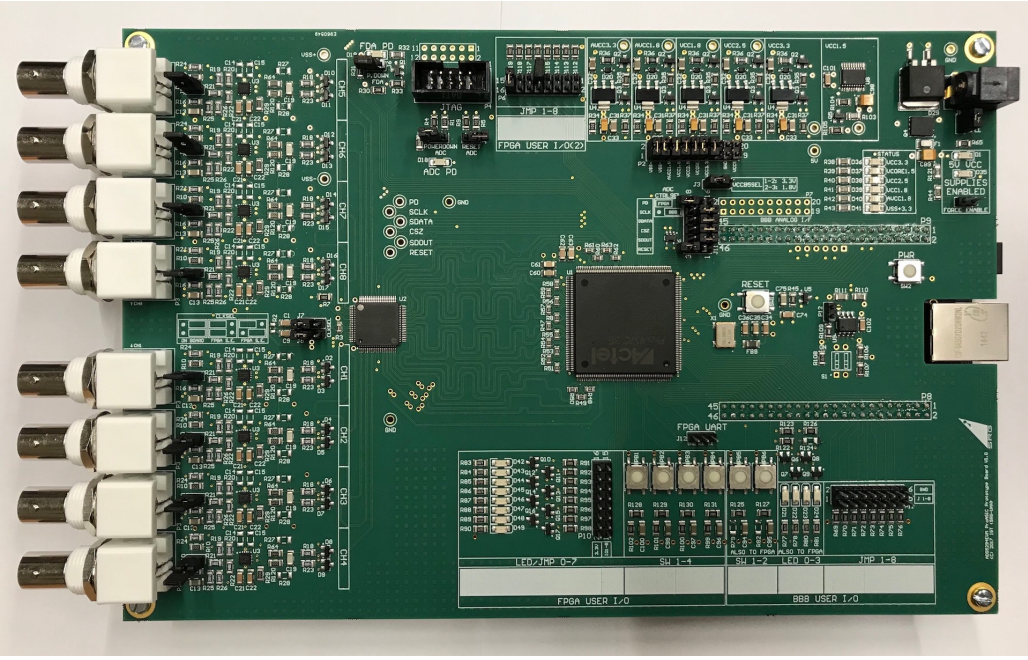


Figure 4.25: ARACNE hardware platform prototype, top side.

design philosophy and details on the implementation of each part is discussed in the following sections.

4.3.1 IPCore design philosophy

Synthesizable hardware modules are usually written in a hardware description language such as Verilog or VHDL, and a whole system can be built combining and interconnecting such modules. However, dependencies on specific vendor characteristics and tools (such as pin to signal assignments or timing constraint definitions) and optimizations to reduce signal delays or fanout, make the organization of code a bit different than on a classic software project.

However, the IPCores of the ARACNE prototype have been written and organized as a proof of concept with reusability in mind where possible. The design of small building blocks with easy to understand functionality that can be combined with each other and used in other FPGAs has been prioritized over more complex elements, and the usage of clear interfaces and behavioural description of hardware modules has been preferred over more compact but more difficult to modify and understand Register Transfer Level (RTL) coding. To this matter,

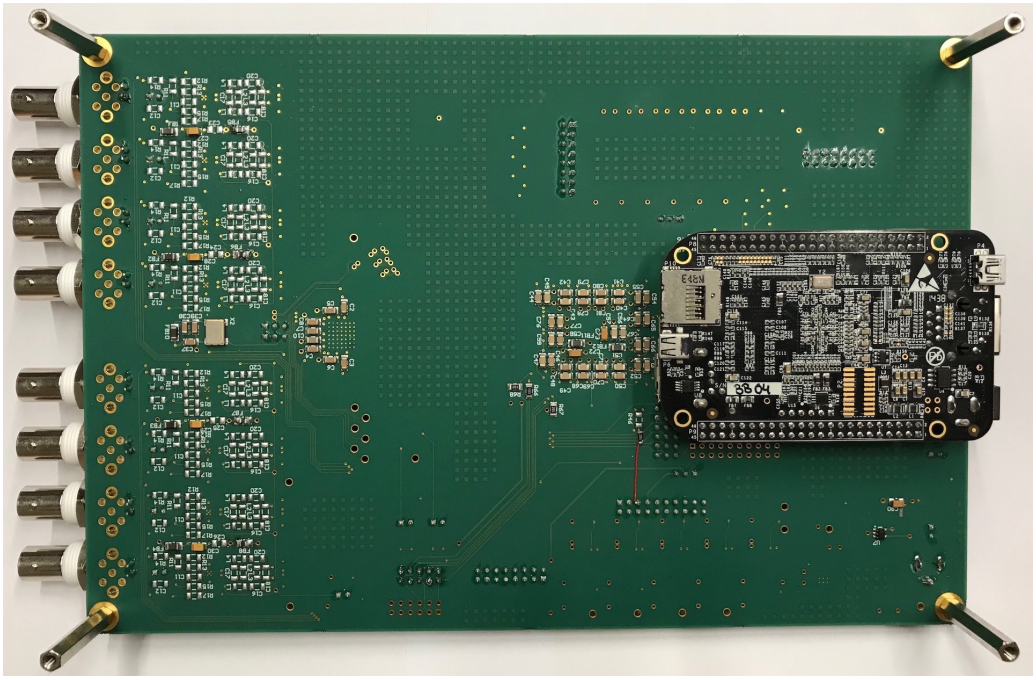


Figure 4.26: ARACNE hardware platform prototype, bottom side.

the methodology described in [37] has been used in most of the code, which is written in VHDL except where vendor-specific modules provided by the Libero IDE tool need to be used.

4.3.2 Prototype functionality general structure

The design of the IPCore for the prototype is conditioned by the target application; the MITO telescope of the ORCA project, which is described in chapter 5. The general design follows the ARACNE IPCore structure proposal outlined in figure 3.4, pt the same time providing the basis to implement more modules and expand the functionality.

In the MITO application, the prototype has to deal with linear pulses coming simultaneously from eight PMTs, in two sets of four, where each set corresponds to a scintillator. Every time a pulse is detected at any of the eight channels, the system must register what happened in all of them; if there was a pulse or not, if there was a coincidence among channels, the height of each pulse, and even store the pulse shape for all eight channels simultaneously. Depending on what is the result of interest (event count, number of events in coincidence, spectrography,

point of impact, particle trajectory, etc.) the required data from the acquisition system can vary.

To achieve this functionality, part of the job is performed by the IPCore and part by the software running on the SBC. Specifically, the IPCore detects particle events and records the full shape of the pulses on all channels, and the software performs further analysis to generate the required data for counts, coincidence, pulse height, etc. The rationale behind this division of work is simple; on one hand, the ADC generates simply too much data in a continuous manner (up to $80\,\mathrm{MS/s} \cdot 14\,\mathrm{bit} \cdot 8 = 8.96\,\mathrm{Gbit/s}$) for it to be transmitted to the SBC in real time, and even in that case it would be impossible to process or even store it at that speed. Thus, the IPCore filters the data of interest, and only that data is transmitted, processed and stored by the SBC. On the other hand, complex data processing is much easier to achieve when performed in software, rather than writing and modifying complex IPCores.

Consequentially, the IPCore has been designed to work the following way:

1. The ADC samples the signals continuously and the FPGA receives data from all eight channels all the time. It deserializes the two LVDS signals per channel, obtaining a flow of 14-bit samples per channel and a data clock signal.
2. The stream of samples for each channel is stored continuously on a FIFO memory and simultaneously fed to a trigger module, which can be configured with a programmable trigger threshold.
3. When the threshold is reached at any of the channels, a trigger signal is sent to each FIFO controller which waits until the pulse shape is completely stored on the FIFO of all channels. The FIFO depth and number of pre-trigger samples stored are programmable.
4. Once full, the FIFO controller freezes the data in the FIFO and generates a signal indicating data for an event is available. This signal triggers a data transfer process in the SBC.
5. The SBC reads the data from all FIFOs, issuing commands to the system controller. Once all data is read, the FIFO controllers continue the data storage on the FIFOs and the trigger module is ready to detect a new pulse and start the process again.
6. Additionally, the system features a timestamp generator that stores the exact time of the last trigger, which can also be read by the SBC, and a few other utilitary modules such as a test pulse generator.

The following sections describe how each part is designed.

4.3.3 Input stage

This stage transforms the digital data and synchronization signals produced by the ADC into a generic parallel data representation that can be processed by the digital chain. In the particular case of the prototype, the ADC transmits the 14-bit samples of each input channel using two serial LVDS links, each of them carrying half (7 bits) of the sample on each data frame. Each bit is synchronized to a bit clock DDR signal (LCLK), and each sample is synchronized to a frame clock DDR signal (ADCLK). Consequently, the data rate of the LVDS links is 7 times the sampling rate. Figure 4.13 represents this. On the other hand, the desired output is a deserialized 14-bit word per channel and a synchronization data clock signal (DCLK), both of them with a data rate equal to the sampling rate. Additionally, since it can take a few cycles for the system to synchronize, a data valid (DVALID) signal is added. The combination of these signals (sample data, DLCK and DVALID) conforms the *input data interface*, and is used by the modules of the digital chain stage. Figure 4.27 represents the functional diagram of the input stage.

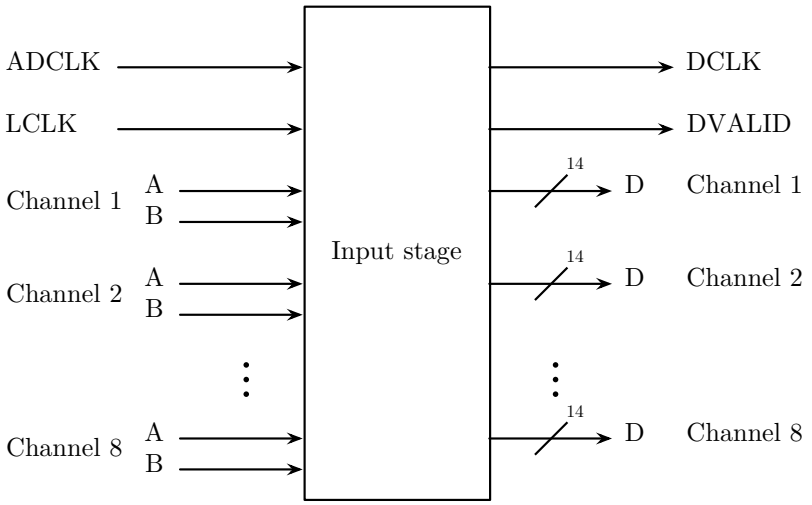


Figure 4.27: IPCore input stage diagram.

Given that the input stage must deseralize data for eight identical channels and sixteen LVDS links the implemented design is highly parallel, instantiating the same deserializer modules eight times; but the data clock and data valid signals can be produced by an independent module, since both can be derived

from ADCLK and LCLK. Figure 4.28 represents the internal structure of the input stage, which is composed of three smaller modules:

- DCLKGEN, which generates the DCLK and DVALID signals.
- DDRDESER7, which deserializes data from each LVDS link using the DDR clock signals LCLK and ADCLK, outputting the resulting half sample. This is a complicated core for several reasons; the high operating speed, the peculiarities of the DDR input cells of the ProASIC A3PE FPGA family, which output 2 bits on the rising edge of the LCLK signal, and the fact that the samples have an odd number of bits.
- DDRCHANNELDESER14, which glues two DDRDESER7 modules together to produce a full sample.

.

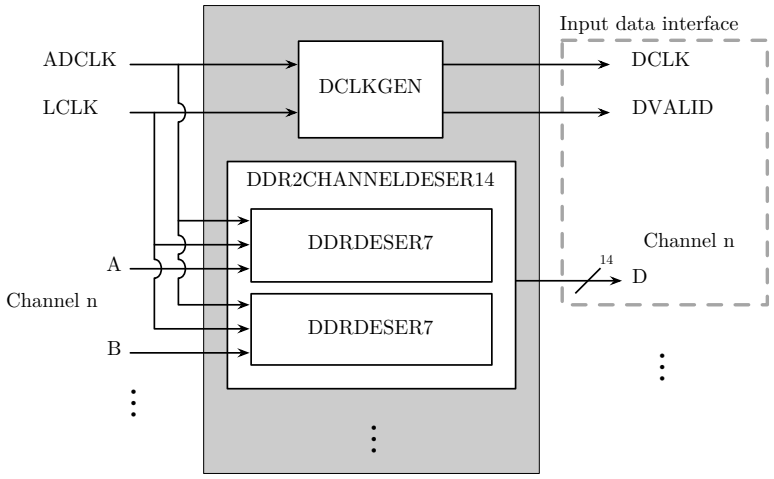


Figure 4.28: IPCore input stage structure.

4.3.4 SBC interface

This stage implements the physical interface for the communication between the SBC and the synthesizable logic implemented in the FPGA, transforming the method chosen by the user (i.e. UART, SPI or parallel interface) into a common channel that can be used to implement a simple communications protocol between the user software and the system controller stage. It must be noted that the

IPCore and the SBC can also "communicate" through the use of other GPIO signals, for example for the SBC to enable or disable certain characteristics of the IPCore or for the latter to provide status or notify that a certain event has happened to the SBC; but these signals are out of the scope of this module.

In the case of the prototype, many possible options for the communications interface have been foreseen, and the hardware platform has been designed so that several options can be implemented (see figure 4.22). The IPCore prototype, however, uses an SPI link, so this stage deals with this type of signals as represented in figure 4.29.

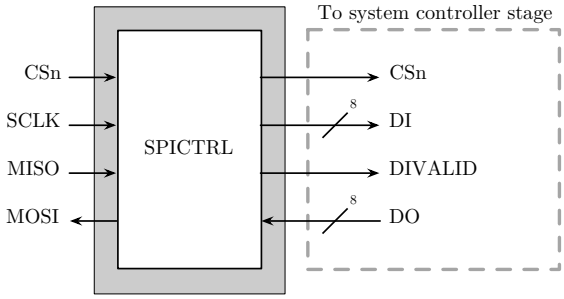


Figure 4.29: IPCore SBC interface stage diagram.

The SPICTRL is the module that implements the typical SPI slave bidirectional interface, and outputs four signals to the system controller:

- CSn is an active low signal that indicates a command from the SBC has started. It returns to a high state once the command has finished and the bidirectional communication is over.

- DI is a parallel 8-bit data bus that contains the last byte received from the SBC in the course of the current communication. Its contents are considered valid when a rising edge of the signal DIVALID occurs.

- DO is a parallel 8-bit data bus with a byte to be sent to the SBC. Data placed in the bus immediately after a DIVALID rising edge will be sent at the same time the next byte is received, and should be stable until the next DIVALID rising edge.

- DIVALID is a synchronization signal both for received and sent data, as described above.

4.3.5 Digital acquisition chain

This stage implements the functionality of the required application and is the core of the ARACNE proposal. Functional signal processing modules are designed and interconnected to perform the operations in a similar fashion to physical instrumentation modules, using essentially three types of resources:

- Detector signals provided by the input stage in the form of the input data interface.
- System registers that allow each module to provide a status and to be controlled and configured. These registers are mapped into a global *register file*, and can be accessed from the SBC through the system controller.
- Specific core interfaces. When interaction through registers is insufficient or inadequate, modules can implement a specific interface that can be managed by the system controller in order for the SBC to interact with the module.

In the case of the ARACNE prototype, which implements the functionality required for the MITO telescope as described in section 4.3.2, the structure of the digital chain is as represented in figure 4.30. This section explains the functionality of each module and the rationale behind the design.

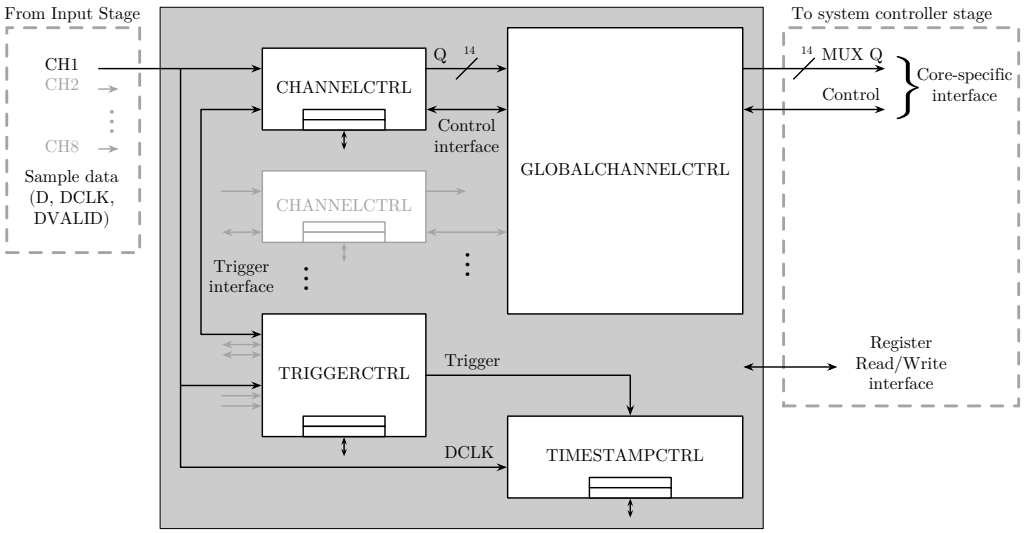


Figure 4.30: IPCore digital chain stage diagram.

Per-channel sample storage

The most important module of the design is the **channel controller** (CHANNELCTRL) module. This module continuously stores the samples received from an input channel, and once it receives a trigger signal and finishes storing the samples of interest, freezes the contents of the storage buffer and notifies that a frame of data corresponding to an event is available. Then, through a control interface, it outputs the stored data serially and once finished it starts storing the digital signal again, ready for a new trigger.

To perform this task, the module is composed of a native FIFO element and a FIFO controller IPCore that implements the required behaviour, as represented in figure 4.31.

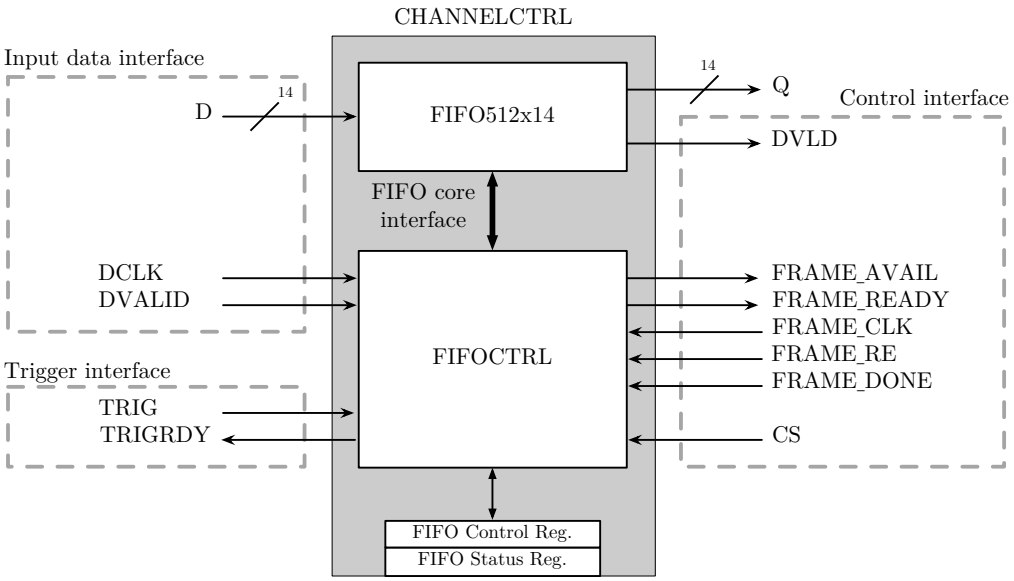


Figure 4.31: CHANNELCTRL IPCore diagram.

The module can be configured with two parameters: the depth of the FIFO to use (e.g. the number of samples to be stored for each event) and the number of samples before the trigger to include in the data set. This way, the full pulse plus the moments before the event occurs can be stored.

Apart from the data input interface, the module implements a *trigger interface* consisting of an input trigger signal (TRIG) with the function explained above, and a trigger ready signal (TRIGRDY) that indicates the trigger module that enough data has been collected in the FIFO, and a trigger is possible. It also

implements a *control interface* used to indicate that an event has been captured and allowing the captured data to be read through the Q port. This control interface consists of the signals summarized in table 4.8.

Signal name	Type	Description
FRAME_READY	Output	An event has been captured and the module is ready to transfer the data
FRAME_AVAIL	Output	There's still data available in the FIFO
CS	Input	The module is selected for reading data
FRAME_CLK	Input	Clock for the reading operation
FRAME_RE	Input	Read enable
DVLD	Output	Data at the Q port is valid after FRAME_RE has been asserted
FRAME_DONE	Input	Transfer is over and the module must restart capturing data

Table 4.8: CHANNELCTRL module control interface.

The module makes use of two global registers. A control register is used to define the depth and pretrigger parameters described above, and the signals FRAME_AVAIL, FRAME_READY and TRIGRDY are reflected in a status register.

Triggering

The next module in this stage is the **trigger controller** (TRIGGERCTRL), which defines the triggering behaviour of the system. It interacts with the other modules via the trigger interface (defined above) and continuously analyzes the input signals, for which it uses the data input interface of all channels. Each observed channel provides a trigger ready (TRIGRDY) signal and the module generates a trigger pulse when all channels are ready and the input signals fulfill a triggering criteria (i.e. a threshold level has been exceeded at a specific input channel). Figure 4.32 represents the module diagram.

The module can be implemented as the user desires and the criteria for triggering can be arbitrary, as long as the interfaces and general behaviour are observed. In the ARACNE prototype the module is composed by eight maskable, single-channel, level trigger modules with hysteresis. They generate an internal trigger pulse when the signal at the specific channel exceeds a certain threshold, and reset when the signal falls below another threshold. If any of the unmasked channels generates a trigger and all channels are ready, the trigger module generates a global trigger signal. Both the channel mask and threshold levels are configurable

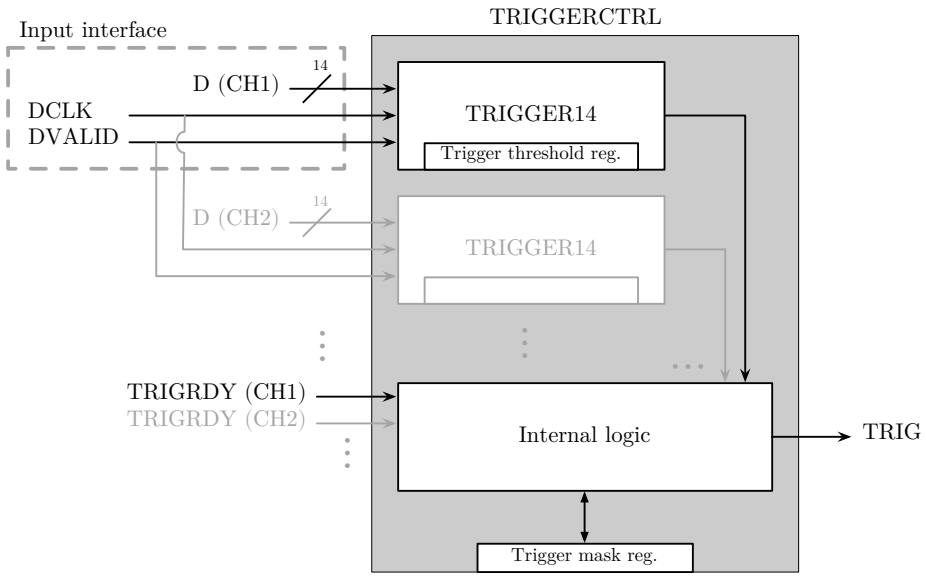


Figure 4.32: TRIGGERCTRL IPCore diagram.

via global registers.

Per-event time-stamp

The next module in the digital chain stage is a **time-stamp generator module** (TIMESTAMPCTRL), which allows the system to store the exact moment an event occurs. The module uses the DCLK signal from the data input interface to count the samples, and internally stores the day, hour, minute, second, millisecond and sample number, elapsed from a preset starting time. When a trigger pulse occurs, it separately stores the time stamp at the moment the trigger happened, which can be later read through global registers. Figure 4.33 represents the module diagram.

As will be explained later, this an example of a module completely accessed through registers by the system controller. It can be enabled or disabled and the time can be set by operating through a control register, and both the current time and timestamp of the last trigger can be read through two 32-bit registers each; one for low resolution time (from days down to seconds) and the other for high resolution time (milliseconds and sample number in that millisecond).

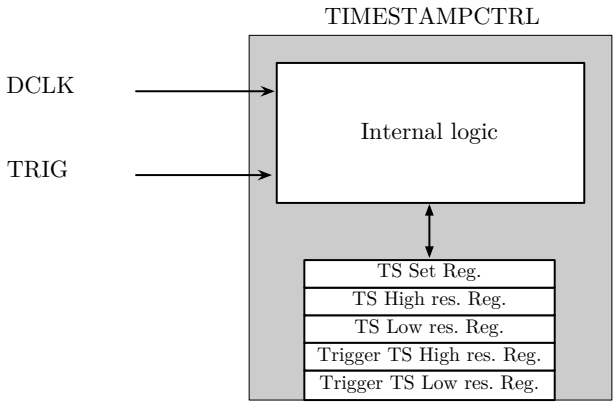


Figure 4.33: TIMESTAMPCTRL IPCore diagram.

Global channel management

The last module of this stage is the **global channel controller**, which allows the system controller stage to read data from all channel FIFOs by multiplexing the Q output data signals and control interfaces (described in figure 4.31 and table 4.8) of all the channel controller modules, as shown in figure 4.34.

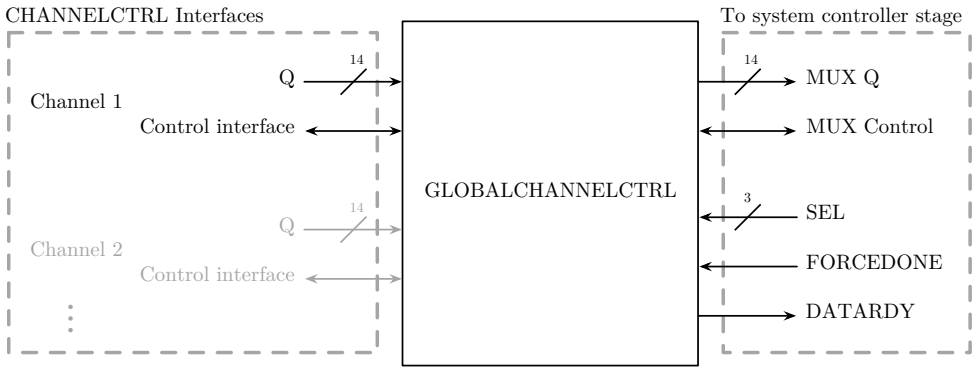


Figure 4.34: GLOBALCHANNELCTRL IPCore diagram.

In addition to the multiplexed signals mentioned before, the module implements a 3-bit channel selection (SEL) signal to determine the channel that is available through the multiplexor, a DATAREADY signal that indicates that there's at least one of the channels in the ready state (e.g. it's not in data capture mode because it still has unread data), and a FORCEDONE signal that can be used to force all channels to discard their stored data and start capturing samples again.

4.3.6 System controller

Finally, the last element of the ARACNE IPCore structure proposal is the system controller. This stage leverages the SBC communications interface provided by the SBC interface stage, as described in section 4.3.4, to implement a command/response protocol to communicate with the software running in the SBC, allowing the user to configure, monitor and, in general, interact with the synthesized hardware modules implemented in the FPGA. To achieve this interaction with the system's modules, there are two basic methods, which were briefly outlined in section 4.3.5; through registers or using core-specific interfaces.

The first method relies in reading and writing registers implemented globally, which on one hand can be read and written by the system controller using a specific register managing module and are, in turn, used by the different modules implemented to define their behaviour or provide status information. Using this method, the user can interact with the desired IPCores by using only two read register and write register commands implemented by the system controller. This method is simple and core independent (e.g. the system controller doesn't need to know anything about the cores). Examples of cores using this method are the timestamp controller and the trigger controller mentioned in section 4.3.5.

However, sometimes interaction only through registers is not feasible or adequate and a different interaction method is needed; for example when large amounts of data must be transferred to or from an IPCore. In this case, the system controller can provide special purpose commands, specific to a module, and leverage the core-specific interface to provide the interaction. An example of a core of this characteristics are the multiple channel controller modules described also in section 4.3.5.

With this functionality in mind, this stage is structured as depicted in figure 4.35. It consists of two modules; the **command server** and the **register file controller**.

Command server

The first module, named SPICMDSERVER in this particular case, implements the communications protocol with the SBC and performs the aforementioned methods of interaction either through each particular core-specific interface or through the register management module (REGFILE) interface. The communications protocol defines a behaviour, a message format and a set of supported commands and their corresponding message payloads.

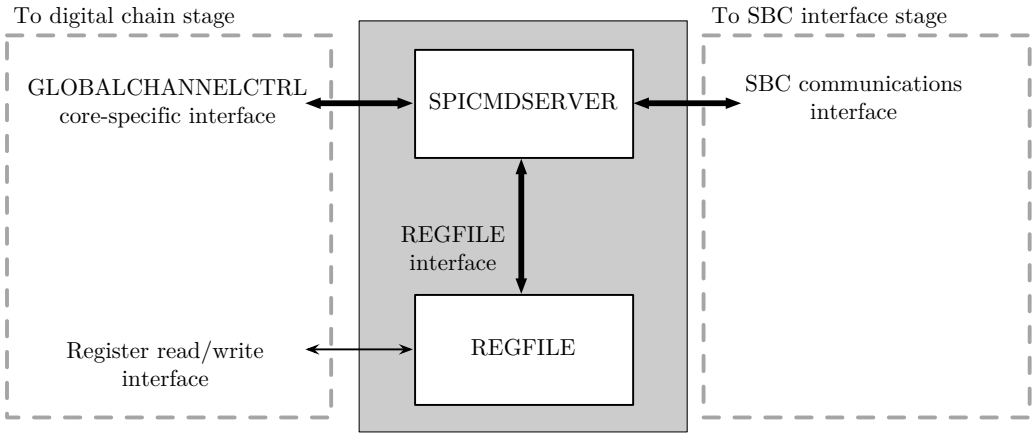


Figure 4.35: IPCore system controller stage diagram.

Command number (hex)	Command name	Description
1	Register write	Writes a control register
2	Register read	Reads a status or control register
3	Channel data read	Reads data from specified channel FIFO.
4	Channels reset	Forces a restart on all channels
AA	Ping-pong	Sanity probe, a preset response is expected

Table 4.9: IPCore commands summary table.

The message format is very simple. The first byte is the command to be executed, and the rest of the message as well as the length of the payload depends on the command. The behaviour of the system is such that a response to a command (if any) is sent back in the same transaction, that is; while the CSn signal is still asserted (see figure 4.29). In this version of the system this behaviour is very tailored to that of a SPI interface (hence the module name), although other communications interfaces such as UART could be used without modifications. The behaviour, however, could be changed in the future to make it more generic. The supported commands are summarized in table 4.9 and their functionality and format are briefly described next:

- **Register write** command (figure 4.36). Writes one of the 16 control registers, specified as a parameter, with the contents of the payload.
- **Register read** command (figure 4.37). Reads the contents of one of the 32 registers, specified as a parameter.
- **Channel data read** command (figure 4.38). Reads data from one of the 8 channels, specified as a parameter. The full contents or any number of bytes

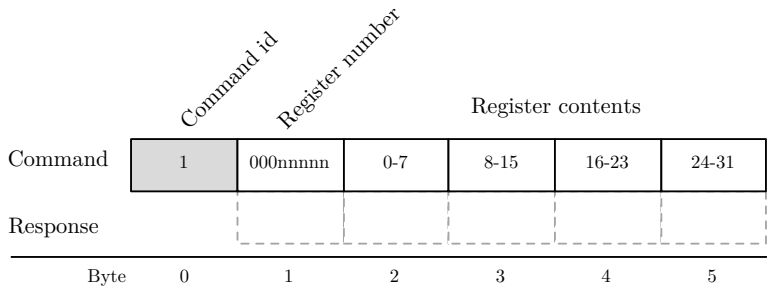


Figure 4.36: Register write command format.

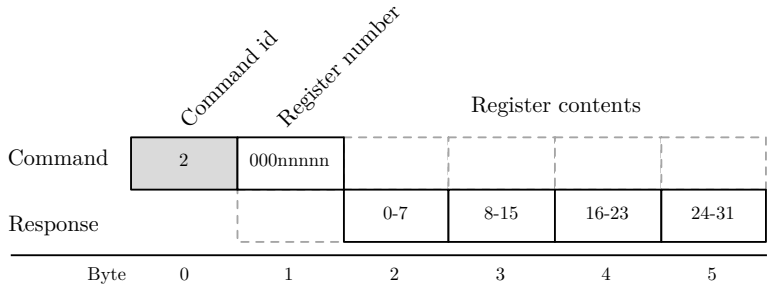


Figure 4.37: Register read command format.

may be read, and they will be consumed from the corresponding FIFO. The remaining contents of the FIFO can be read in subsequent commands, and if data is read beyond the limit, zeroes will be used as padding. It must be noted that the FIFO length is set by the user and thus is known in advance. Also, the status of the channel (i.e. if data is available) can be read from the corresponding status register.

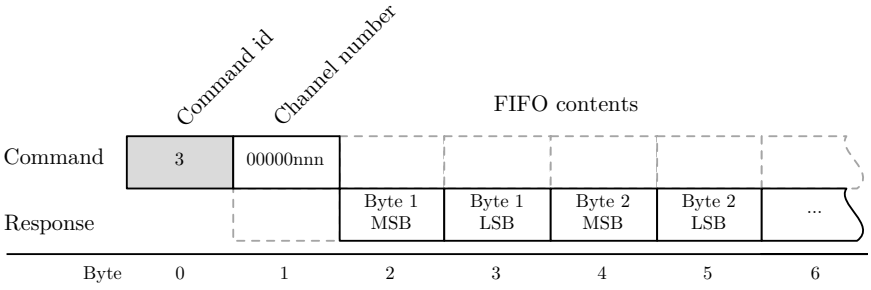


Figure 4.38: Channel data read command format.

- **Channels reset** command (figure 4.39). Commands all channel controllers

to discard the data on the FIFOs and restart data capture immediately. This command has no payload.

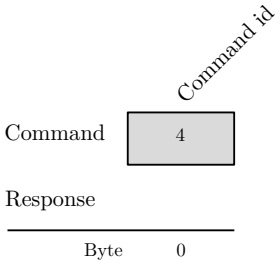


Figure 4.39: Channels reset command format.

- **Ping-pong** command (figure 4.40). Utility command to check that the system controller is working. A preset response consisting of the 32-bit word `1A2B3C4Dh` is expected.

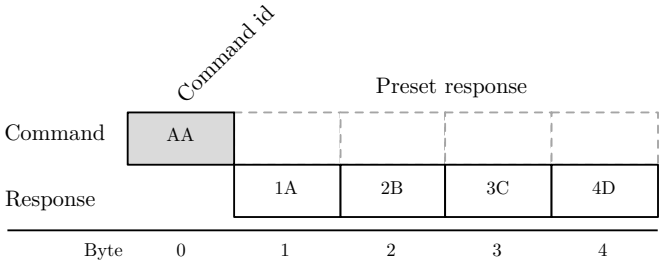


Figure 4.40: Ping-pong command format.

Register management module

The second module of this stage is the aforementioned register management module, represented in figure 4.41, which implements the logic to read and write the global control and status registers. In the current implementation it supports two banks of sixteen 32-bit registers, although only a few of them are used in the prototype. Table 4.10 provides a summary of the used registers.

4.3.7 Resulting IPCore prototype

The design has been implemented successfully using the Libero IDE from MicroSemi Semiconductor Corp. using the hardware platform ProASIC A3PE1500

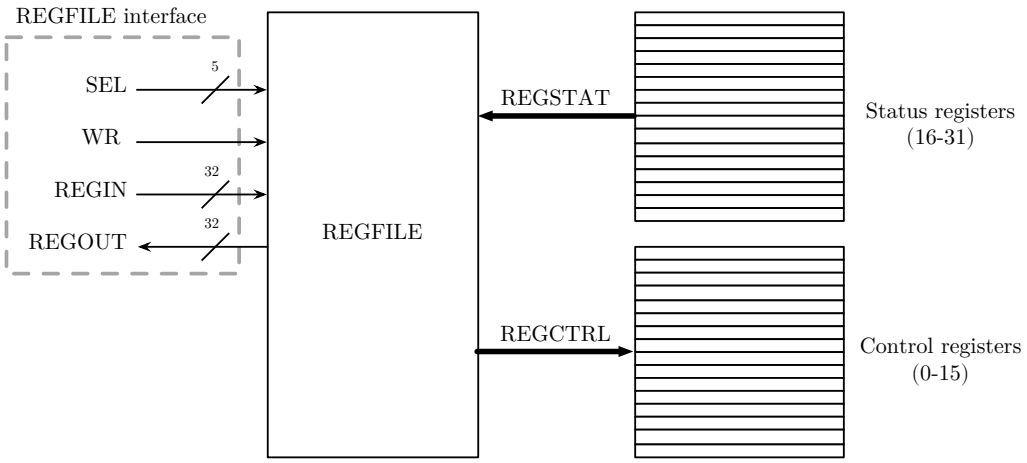


Figure 4.41: REGFILE IPCore diagram.

FPGA as target, adequately configuring all the I/O banks and performing the necessary pin to signal assignments. The current version of the prototype uses less than 18% of the logic resources of the FPGA and less than a 27% of the RAM/FIFO resources, while using 72% of the I/O available.

The internal system clock driving the logic circuits is set at 100 MHz, and the first versions are stably providing the expected functionality with the ADC configured at 40 MS/s and the SPI link to the BBB SBC configured at 3 Mbit/s. In this conditions, the system is capable of capturing in excess of 9.000 events per minute with the current software, depending on the configuration. As expected, the depth of the FIFO buffers affects the performance, since deeper buffers mean longer transfer times to the SBC and longer dead times, and full pulse storage in the SBC is a lot more demanding than just counting events or storing peak values.

An extensive report of the results provided by the prototype in the MITO instrument of the ORCA project is provided in chapter 5. However an illustrative example of these results is provided in figure 4.42. It represents the data gathered during one event from the eight input channels of MITO by the IPCore, using the real hardware platform. A pulse is observed in all channels, with different peak values. In this case, 200 samples per channel, which is enough to capture the full pulse shape, were stored at 40 MS/s. A pretrigger value of 40 samples was selected, which allows not only to fully capture the start of the leading edge of the pulse, but also a significant amount of samples of the baseline value. As a consequence a baseline offset can be observed, which can be later corrected in

Reg. num.	Register name	Description
0	FIFO control	Bits 25-16: FIFO delay. Samples to store pre-trigger. Bits 9-0: FIFO depth.
1	Trigger mask	Bits 7-0: Enable (1) or disable (0) trigger for channels 8 to 1.
2	Trigger thresholds	Bits 29-16: Trigger threshold level. Bits 13-0: Trigger reset level.
3	Timestamp control	Bit 31: Set time using value field. Bit 30: Enable timestamp counter. Bits 25-0: Time value to set, where 25-19: Day 16-12: Hour 11-6: Minute 5-0: Second
16	FIFO status	Bits 23-16: Ready for trigger, channels 8 to 1. Bits 15-8: Data available, channels 8 to 1. Bits 7-0: Channel ready, channels 8 to 1.
17	Timestamp low	Bits 25-0: Current time, low resolution; where 25-19: Day 16-12: Hour 11-6: Minute 5-0: Second
18	Timestamp high	Bits 25-0: Current time, high resolution; where 26-17: Millisecond 16-0: Sample number
19	Trigger timestamp low	Bits 25-0: Last trigger time, low resolution; where 25-19: Day 16-12: Hour 11-6: Minute 5-0: Second
20	Trigger timestamp high	Bits 25-0: Last trigger time, high resolution; where 26-17: Millisecond 16-0: Sample number
29	IPCore version	Bits 31-16: Major version. Bits 15-0: Minor version.

Table 4.10: IPCore register table.

post-processing.

4.4 Embedded software support

The final piece of the ARACNE proposal is low-level software that provides the end-user a layer of abstraction that helps in the development of control and monitoring software to implement the desired instrumentation application, as described in section 3.3.3. The goal of these libraries is two-fold; on one hand, to offer an Application Programming Interface (API) that is more user-friendly than the one provided by the specific SBC, which is usually defined in terms of low-level hard-

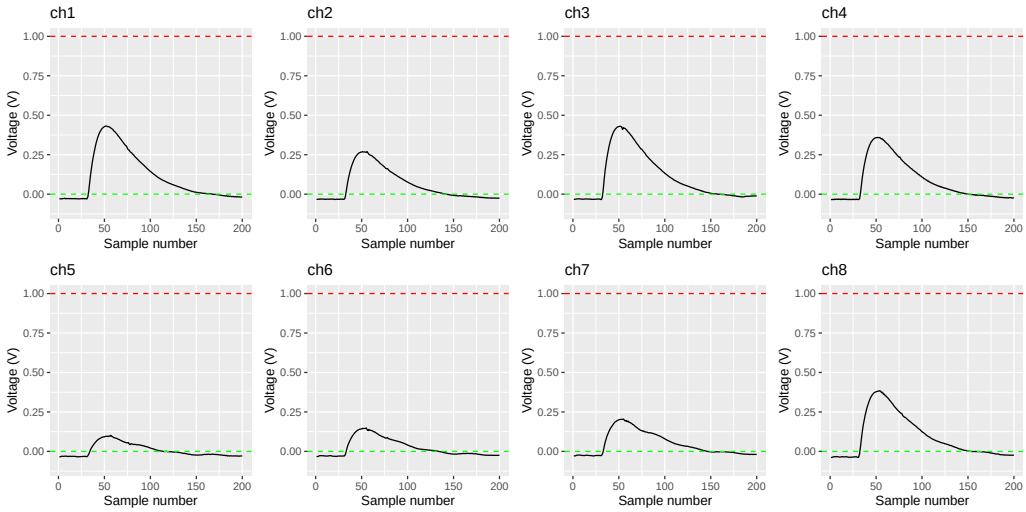


Figure 4.42: Example of results obtained with the IPCore.

ware interfaces. And on the other, to provide an homogeneous API regardless of the particular SBC.

This section starts describing the interfaces provided by an I/O API (similar to a BSP) for the chosen SBC, and then describes how the libraries provided leverage them to provide a higher level interface for ADC, GPIO and core control. With ease of implementation and wide availability in mind, the Python language has been chosen for these libraries.

4.4.1 The I/O API

The Adafruit BeagleBone I/O Python API (BBIO) Python API [38] enables applications written in Python to access the I/O hardware of the BBB, such as GPIO pins, SPI channels, UARTs, etc. Regardless of its limitations, it provides the necessary resources to control the hardware platform ADC and miscellaneous control signals, as well as the interface between the SBC and the FPGA. For the prototype, the GPIO and SPI parts of the API have been used, and will be described next.

GPIO low-level API

The low-level GPIO API provided by BBIO consists of three basic elements; pin naming definitions, utility macros, and pin control functions.

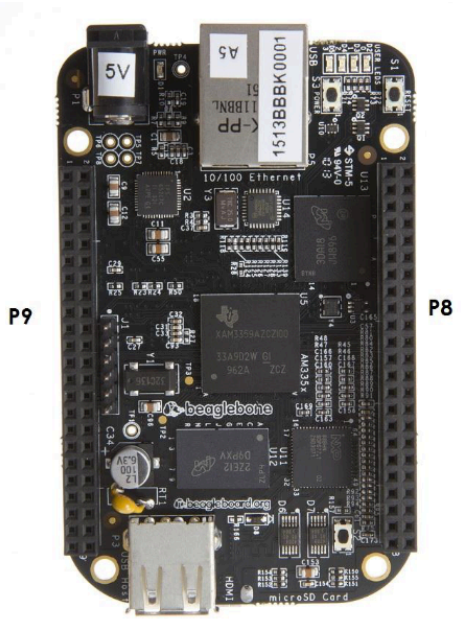


Figure 4.43: BBB user I/O connector naming.

Pins are named depending on the I/O header and pin number. Headers are named P8 and P9 (see figure 4.43), and each has 46 pins, thus for example pin 15 of header P8 is named `P8_15`. There is, however, an alternative way for pin naming that depends on the SoC resources; but it's a lot less intuitive. Utility macros on the other hand are used to specify a variety of standard parameters for functions of the API, such as state, pin mode and signal transitions. Finally, pin control functions allow the user to configure a GPIO pin as input or output, set or read its state, wait for a transition to occur, etc. Macros along with the API function reference and several functional examples can be found at the documentation [39], so they will not be reproduced here.

It must be noted, though, that this low-level API, although simple, can change depending on the SBC. The goal of the system libraries provided in the proposal is to provide similar functionality while homogenizing the API, so that the same software written for a platform can be used (or require minimal changes) at another.

SPI low-level API

The SPI section of the BBIO API provides access to the two SPI channels on the Beaglebone Black. In the prototype, one of them is used to control the ADC and the other to communicate with the system controller on the IPCore. The API is object oriented. As such an `SPI` object is instantiated on each corresponding bus and device, configured as required and then used invoking the methods provided by the class which allow writing, reading and simultaneously writing and reading.

As with the GPIO section of the API, the function reference can be found in the documentation [39] so it will not be reproduced here.

4.4.2 Support libraries

The mission of the provided support libraries is to encapsulate the BBIO API and the configuration details pertaining to the interaction between the hardware platform and the IPCore with the SBC. An explanation of the functionality of each module of the libraries was outlined in section 3.3.3 and a diagram of their location in the software system was provided in figure 3.5. This section describes the API provided by each of these modules.

ADC low-level library

Most ADCs are configured through writing and reading registers, regardless of the method of communication used between the device and the programming platform, and the ADS5294 used in the hardware platform is no exception. But this apparently simple operations may not be *that simple*, depending on how these registers are accessed at the hardware level. Not only the hardware to communicate with the ADC must be configured and managed, but in many cases also a communications protocol must be implemented (i.e. when the communications are implemented through a serial link). Moreover, in some occasions a simple operation can be composed of more than one command for the ADC. For this reason, the ADC low-level library layer provides an API for reading and writing registers, hiding the way this is done on a specific platform.

In the prototype, this functionality is provided by the `ADS529x` class, implemented in the library file `ads529x.py`. This library performs all the required initialization and implements the communications protocol described in the documentation [30], providing a simple, object oriented user interface. When an object of this class is instantiated, the corresponding SPI connected to the ADS5294

channel is initialized, and the object can be used to interact with the ADC using the methods summarized in table 4.11.

`ADC_reg_write( regnum, regvalue)` `ADC_reg_write_check( regnum, regvalue)`	Writes the 16-bit register specified by `regnum` with the value `regvalue`. The `check` version reads it back to check for validity and returns `True` or `False` depending on success or failure of this check.
`ADC_reg_read( regnum)`	Reads the 16-bit register specified by `regnum`, returning its 16-bit value.

Table 4.11: ADS529x API summary.

This very simple API makes low-level interaction with the ADC extremely easy and straightforward. For example, the python code of listing 4.1 sets up the ADC and reads register number 2.

```
1 ADC = ADS529x()
2 result = ADC.ADC_reg_read( 0x02)
```

Listing 4.1

Otherwise, the ADC SPI channel should be initialized and configured manually. Then, three messages in a specific format should be sent to the ADC as per the documented communications protocol; a register write to set the ADC in read mode, the register read per se and another register write to return the ADC to the initial state. And the code would be only valid for this specific ADC. The sample code shown before, on the contrary, is independent of these details.

The functionality provided by this class makes the implementation of a higher level ADC control library described in the following section much easier.

ADC control library

If the user is left with a low-level library that just allows easy reading and writing of registers, everything should be done based on the documentation and reading and writing the appropriate registers to perform every operation. For example, to enable testing mode and force the ADC to output a single `val` value continuously, the user should use the code of listing 4.2.

```
1 ADC.ADC_reg_write( 0x25, 0x0010 | ( (val >> 12) & 0x3))
2 ADC.ADC_reg_write( 0x26, (val << 4) & 0xFFF0)
```

Listing 4.2

This does not seem very convenient and is very error-prone. It would be much easier to just write code similar to that of listing 4.3.

```
1 ADC.testing_singleval( val)
```

Listing 4.3

For this reason, an ADC control library to make working with the ADC safer and simpler is proposed. It must be noted that a library of this type will always be specific to the device (the ADS5294 in this case), because although most ADCs are configured in a similar way as explained in the previous section, their features and characteristics vary considerably and the function of registers is different.

In the prototype the library provides the `ADS5294` class, which is implemented in the `ADS5294.py` library file and makes use, in turn, of a low-level ADC control object of the class `ADS529x` described in the previous section. It performs the configuration of the ADC for the prototype, and is able to interact with the appropriate device registers to leverage its features, providing a simple, object oriented user interface for them as in the example above. The result is a user program can initialize and setup the ADC in the ARACNE system with the code shown in listing 4.4.

```
1 from pyADS529x import ADS5294
2
3 ADC = ADS5294()
```

Listing 4.4

Apart from correctly configuring the ADC of the prototype when an object of this class is instantiated, the methods summarized in table 4.12 are available. It must be noted that the API offered by the current ADC control library does not leverage all the features of the device, since they were not used in the application. But the library code is very easy to read and extend, and the ADC low-level library presented in the previous section provides an easy way to manage the affected registers, so completing the functionality should be a simple task.

Testing mode methods	

`testing_singleval( val)`
`testing_dualval(val1, val2)`
`testing_ramp( resetval)`
	Enable ADC single value, dual value or ramp testing mode.

`testing_disable()`	Disables testing mode.
Input related methods	
`inputs_polarity(mode)`	Tells the ADC to invert the polarity of the input signals. `mode` can be `normal` or `inverted`.
`DDR_phase( phase)`	Phase of LCLK DDR signal with respect to the data signals . Possible values are `00`, `01`, `10` and `11` (see [30, p. 56]).
Output related methods	
`datarate( rate)`	Sets the ADC output data rate with respect to the sample rate; the same (`full`), 1/2 (`1_2`), 1/4 (`1_4`), or 1/8 (`1_8`).

`output_config(`
` LVDS_out=<mode>,`
` LSB_First=boolean)`
	Output mode of the ADC (one or two LVDS links per channel) and bit order of samples, LSB first or MSB first. The only currently supported `mode` is `2WIRE`

Other methods	
`reg_dump()`	Prints the contents of the whole ADC register bank to the console. This can be used for development purposes.

Table 4.12: ADS5294 API summary.

Board pin control library

Even though the GPIO control interface provided by the BBIO API is relatively simple, the board pin control library provides an additional abstraction layer with two goals in mind: simpler pin initialization and manipulation, and a platform-agnostic interface for the GPIO control library to use. At the same time, it provides an object-oriented approach that makes it easier to use.

The board pin control library defines the `BRDpin` class, which can be instantiated for each GPIO pin that is going to be used. Listing 4.5 provides a usage example.

```
1 from pyBRDpin import BRDpin
2
3 PinADCReset = BRDPin( "P9_23", GPIO.OUT)
```

Listing 4.5

In addition to the pin mode, other parameters can be set when declaring a pin, such as initial value and resistor pull-up/pull-down for output pins or

edge sensitivity for input pins. If these parameters are not specified, the class provides reasonable defaults. Once instantiated, the library provides the methods summarized in table 4.13 for objects of this class.

`read()`	Returns the state of an input pin.
`sethigh()`	Sets an output pin to high (1).
`setlow()`	Sets an output pin to low (0).
`edge_waifor([blocking=boolean])`	Waits until the configured edge transition occurs. If the `blocking` parameter is set to false, the method returns false if a call would block. In all other cases, it returns true.

Table 4.13: BRDpin API summary.

GPIO control library

Even though the board pin control library could be used by the programmer directly, the GPIO control library adds an abstraction layer that makes working with an specific platform implementation easier. It defines a class named `BRDgpio` that provides higher level pin naming, specific to an IPCore implementation, so that the user doesn't have to think in terms of the SBC I/O. Also, the class instantiates all the GPIO pins with their default configuration, and then allows its manipulation through a very simple interface, which is an extension of the `BRDpin` class. In the case of the prototype, all the GPIO pins are initialized and ready to be used just by using the code shown in listing 4.6.

```
1 from pyBRD import BRDgpio
2
3 board = BRDgpio()
```

Listing 4.6

Then, GPIO operation is as simple as invoking the appropriate method for a GPIO pin defined in the class itself, as shown in listing 4.7.

```
1 board.pin_pulse_high(board.PIN_ADC_RESET, pulselensec=0.050)
```

Listing 4.7

The GPIO pins defined in the GPIO control library version of the prototype are summarized in table 4.15, and the methods provided by the `BRDgpio` class are summarized in table 4.14.

Obviously, the `BRDgpio` class must be tailored in each case to the hardware

`pin_pulse_high( pinname, [pulselensec=0])` `pin_pulse_low( pinname, [pulselensec=0])`	Generates a positive (high) or negative (low) pulse at the specified output pin, with a duration defined by `pulselensec`.
`pin_set_high( pinname)` `pin_set_low( pinname)`	Sets the specified output pin to a high (1) or low (0) state.
`pin_read( pinname)`	Returns the state of the specified input pin.
`pin_edge_waitfor( pinname)`	Waits until the specified input pin transitions, unless a `blocking=false` parameter is passed, in which case it returns immediately and provides a return value of false if the call would have blocked or true otherwise.

Table 4.14: BRDgpio API summary.

Pin name	Description
PIN_ADC_RESET	ADC reset.
PIN_ADC_PD	ADC power down.
PIN_FDA_PD	Fully-differential amplifiers power down.
PIN_CORE_RESETN	IPCore global reset.
PIN_TRIG_FORCE	Force a trigger from the SBC.
PIN_TRIG_IN	Trigger signal from the IPCore. Only generated if all channels are ready.
PIN_TRIG_RDYALL	All channels are ready for trigger in the IPCore.
PIN_TRIG_ALL	Trigger signal from the IPCore regardless the channels are ready or not.
PIN_DATARDY	Data is available at the input channels.

Table 4.15: GPIO pins defined in the prototype.

platform and the IPCore implemented in the FPGA since it is necessary to define the pin naming and their configuration, but this is a simple task that must only performed once. For example, in the case of the ARACNE prototype for the MITO application, the code shown in listing 4.8 is used (not all the actual code is shown):

```
PIN_CORE_RESETN     = "P9_14"
PIN_ADC_RESET       = "P9_23"
PIN_ADC_PD          = "P9_15"
PIN_FDA_PD          = "P9_16"
PIN_TRIG_IN         = "P9_12"
PIN_DATARDY         = "P8_10"
[...]

self.BRDpins = {
[...]
 self.PIN_ADC_PD:    BRDpin( self.PIN_ADC_PD, GPIO.OUT, defval=GPIO.LOW),
 self.PIN_FDA_PD:    BRDpin( self.PIN_FDA_PD, GPIO.OUT, defval=GPIO.LOW),
 self.PIN_TRIG_IN:   BRDpin( self.PIN_TRIG_IN, GPIO.IN),
 self.PIN_DATARDY:   BRDpin( self.PIN_DATARDY, GPIO.IN, GPIO.RISING)
[...]
}
```

Listing 4.8

Core low-level communication library

This library encapsulates the physical means of communication between the SBC and the system controller IPCore implemented in the FPGA. It automatically sets up and configures the necessary hardware and provides a simple API to send and receive bytes, that in turn will be used by the core command library to implement the communications protocol.

In the case of the prototype, the physical communications channel is a SPI link, and the low-level communication library implements the `COREcommspi` class, which provides this functionality. As such, it is possible to set up the channel by instantiating an object of this class as shown in listing 4.9.

```
from pyCOREcommspi import COREcommspi as COREcomm

corecomm = COREcomm( datarate)
```

Listing 4.9

The optional `datarate` parameter is specific to a SPI link and indicates the link speed in bits per second. Then, the methods summarized in table 4.16 are available for the `corecomm` object.

With just these two commands, the upper layer can define a message format and exchange messages with the system controller IPCore to manage the digital acquisition chain.

`writebytes( databytes)`	Sends a sequence of bytes to the IPCore. `databytes` is a list of bytes.
`txferbytes( databytes, len=0)`	Simultaneously sends and receives a sequence of bytes of the same length to/from the IPCore. If `len` exceeds the length of `databytes`, padding 0x00 values are added to it. Returns the received sequence.

Table 4.16: COREcommspi API summary.

Core command library

This library leverages the commands to send and receive data provided by the core low-level communication library to implement the communications protocol with the command server IPCore and the commands it supports, as defined in section 4.3.6. It defines the `COREctl` class, which in turn initializes the communications channel and provides the commands API. As such, interaction with the IPCore is possible with the code shown in listing 4.10.

```
1 from pyCOREctl import COREctl as COREctl
2
3 corectl = COREctl( datarate)
```

Listing 4.10

The optional parameter `datarate` has the same meaning as in the previous section, and is directly passed to the low-level communication library. Then, the `corectl` object can be used through the methods summarized in table 4.17.

This API provides a high level of abstraction and allows the programmer to think just in terms of IPCore registers and data in FIFOs of input channels. This way, writing software that controls a new user-designed IPCore can be done without knowledge of the underlying details of the communication between the SBC and the FPGA if the IPCore just uses system registers; and at the same time, adding new commands to the system controller and using them can be done just by expanding the core command library.

Core control library

Similarly to what happened with the ADC, the API provided by the core command library allows the user to interact easily with the IPCore by reading or writing registers or reading data from FIFO channels, but this is not very convenient. For example, setting the FIFO depth and pretrigger delay parameters

Register related methods	
`reg_write( regnum, regvalue)` `reg_write_check( regnum, regvalue)`	Writes the 32-bit `regvalue` value to the specified IPCore register. The `check` version reads it back, checking if it was correctly written, and True or False is returned.
`reg_read( regnum)`	Reads and returns the 32-bit IPCore register specified.
FIFO related methods	
`FIFO_read( fifonum, fifolen)`	Reads `fifolen` samples from the channel specified in `fifonum`. It returns a list of integers with the sample values. If the FIFO does not contain enough data, extra samples will be 0.
`FIFO_forcedone()`	Commands the IPCore to discard the data in all the FIFO and restart data capture in all channels immediately.
Other methods	
`pingpong()`	Simple *ping* command that checks if the command server is working. The response is returned by the method.

Table 4.17: COREctl API summary.

would require the user to know that the implied register number is 0x02 and the format of this register. Then, the parameters could be set as shown in listing 4.11.

```
1 regval = (delay & 0x3FF) << 16 | ( depth & 0x3FF)
2 corectl.reg_write( 0x02, regval)
```

Listing 4.11

Again, not very convenient and error-prone, apart from the fact that the code is not easy to understand. For this reason, a core control library should be written to implement a high-level user interface that helps writing user programs and hides the hardware details. In the case of the prototype, a library implementing a `COREobj` class that provides a rich and easy to use API to leverage the functionality of the implemented IPCore is provided. A summary of the methods available can be found in table 4.18.

It is easy to see that, by using a library with this level of abstraction, programming an application for an ARACNE-based acquisition and processing system should require little knowledge of the hardware platform or the IPCore layout and design. All the methods in the API are defined in terms of channel numbers, threshold levels, timestamps or easily understandable concepts like if a channel's

FIFO-related methods	
`FIFO_set_params( delay, depth)` `FIFO_get_params()`	Set or get the FIFO depth and pretrigger delay parameters.

`FIFO_ch_read( fifonum, fifolen)`
`FIFO_ch_flush( fifonum)`
`FIFO_all_flush()`
	Read `fifolen` samples from the specified FIFO, returned in a list of integers. The `flush` versions read all the contents, as defined by the `depth` parameter in the IPCore. The `all` version reads all the FIFOs and returns a list of lists.

`FIFO_all_ignore()`	Discards the contents of all FIFOs and starts capturing data again.

`FIFO_ch_istrigready( channel)`
`FIFO_ch_isready( channel)`
`FIFO_ch_dataavail( channel)`
	Checks if a channel is ready for triggering, is ready to transfer the data and has data still available in the FIFO, respectively. True or False is returned.

Trigger-related methods	

`trig_all_enable()`
`trig_all_disable()`
`trig_ch_enable( channel)`
`trig_ch_disable( channel)`
	Enables or disables triggering for all channels or a specific channel.

`trig_ch_isenabled( channel)`
`trig_allchannels_mask()`
`trig_allchannels_status()`
	Checks if a channel is enabled for trigger, returning True or False, or checks all of them returning either a 8-bit mask or a 8 position list of 0 and 1 values.

`trig_set_thresh( th_hi, th_lo)` `trig_get_thresh()`	Sets and gets the threshold values for triggering and rearming.
Timestamp-related methods	
`ts_enable()` `ts_disable()`	Enable or disable (stop) the timestamp generator.

`ts_set( ts_day, ts_hour,`
`ts_min, ts_sec)`
`ts_get()`
`ts_trig_get()`
	Set the timestamp to a specified value in day, hour, minute and second, or get the current or last trigger timestamp value.

Other methods	
`pingpong()`	Send the ping-pong command and return the 32-bit value returned by the IPCore.

Table 4.18: COREobj API.

trigger is active or not. The user doesn't need to think in terms of register numbers or formats, how is the FPGA being interfaced or, mostly, how things work. The example at the beginning of this section would be reduced to the code shown in listing 4.12.

```
1 ipcore = CORElib( )
2
3 ipcore.FIFO_set_params( delay, depth)
```

Listing 4.12

A library (or set of libraries) like this should be written each time a new acquisition module is implemented in order for its functionality to be made available to the application programmer, in a similar way as a computer needs a new device driver when new hardware is installed. As a reminder, the goal of the ARACNE concept is to provide the user with a hybrid platform composed of fixed hardware, a reconfigurable hardware and system libraries that allow the creation of new acquisition chains without hardware modifications. As such, new synthesizable hardware elements require supporting software to be easily used.

4.4.3 Resulting system libraries prototype

Once all the libraries mentioned in this section are deployed, writing a Python application for data capturing, processing and storage, and in general to control the ARACNE prototype, is simple.

```
1 from pyBRD import BRDgpio
2 from pyADS529x import ADS5294
3 from pyCORE import COREctl
4
5 board = BRDgpio()
6 ipcore = COREdev( CORE_SPI_DATARATE )
7 ADC = ADS5294()
```

Listing 4.13

A piece of code just like the one shown in listing 4.13 would be enough to initialize the system and provide control over all the implemented hardware and IPCore, allowing the programmer to focus on the functionality he needs to implement for the specific study or experiment.

Chapter 5

Application to ORCA

> "Some men see things as they are and say 'why?'. I dream of things that never were and say 'Why not?' "
>
> — Robert F. Kennedy

During the development and construction of the ORCA project by the SRG of the Universidad de Alcalá, a nuclear instrumentation system was needed for the MITO telescope [40], capable of simultaneously and adequately register eight signals from two scintillation detectors, with the purpose of muon flux and anisotropy studies. To accomplish the construction of this system, the design principles of the ARACNE proposal were used to implement its digital acquisition chain section, which resulted in the multipurpose prototype described in chapter 4. This prototype has successfully fulfilled the application necessities of the MITO telescope, and was installed in Antarctica in January 2019 integrated in the ORCA station (Figure 5.1), where it's been gathering data ever since and has successfully completed its first year-long autonomous and uninterrupted operation period from January 2020 to January 2021.

This section provides a summary of the ORCA project and a description of the MITO instrument, as well as its signal acquisition and analysis requisites. Then, the functionality of the software created to leverage the ARACNE prototype in the context of the MITO instrument is described, and finally an overview of the results obtained is provided.

Figure 5.1: ORCA (the black container on the right) installed at its final location, at Juan Carlos I Antarctic Base.

5.1 ORCA, the Antarctic Cosmic Ray Observatory

One of the radiation types Earth is subject to are Galactic Cosmic Rays (GCRs), charged particles accelerated in galactic and extragalactic sources. When these particles collide with the atmosphere at high altitude they produce a continuous shower of secondary particles, such as neutrons and muons, part of which can reach the Earth surface. Information about the energy of the colliding particles is reflected in the amount of secondary particles generated.

At the same time, GCRs in the hundreds of MeV to tens of GeV energy range are affected by solar activity, which modulates their intensity both by long term changes, such as the solar cycle or the star's evolution, and short term causes, such as interplanetary Coronal Mass Ejections (CMEs), Magnetic Clouds (MCs), shock waves, flares and other interactions. Consequently, solar activity can be studied by observing the GCR flux variations indirectly, through the measurement of the flux of secondary particles reaching the Earth's surface. In addition, Solar Energetic Particles (SEPs) produced by the Sun, which can have energies similar to those of some GCRs, can be studied by similar instruments. For example, the arrival of solar wind structures as interplanetary shock waves and magnetic clouds cause a decrease in the particle flux known as Forbrush Decreases (FDs) [41], as shown in figure 5.2.

Apart of the scientific value that instruments capable of measuring such flux variations can provide, these measurements are of particular relevance for the

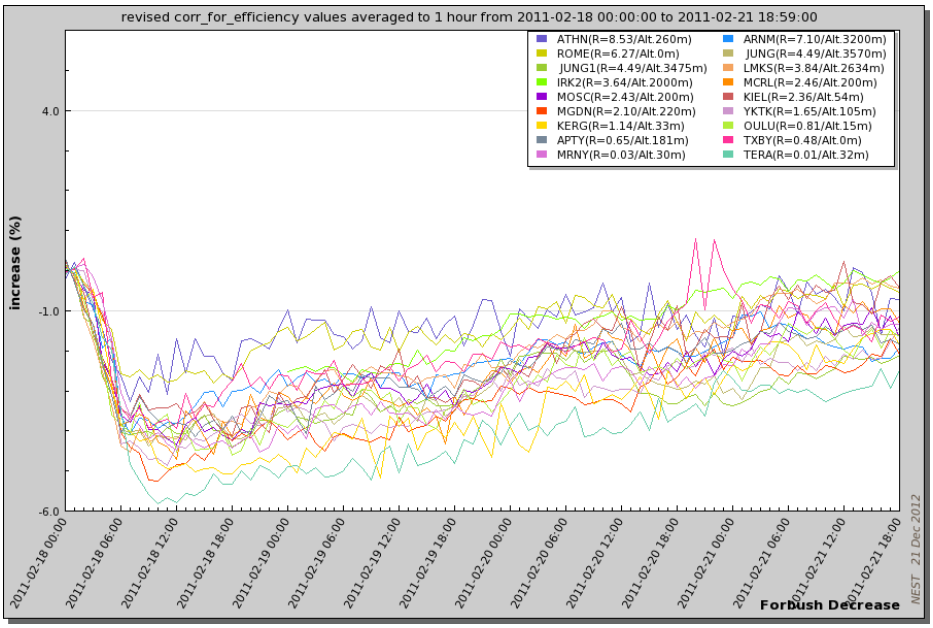


Figure 5.2: FD example, registered by neutron monitors of the Neutron Monitor Database (NMDB). Pictures from [42].

space weather field if they are delivered in real time since they can bring the capability to forecast solar events that are potentially dangerous for humans and equipment. For example, explosive events such as CMEs and solar flares directed towards Earth produce SEPs that can be dangerous, but the most energetic of these particles arrive at Earth several minutes in advance of the bulk of SEPs and cause sudden and relatively short duration increases in particle flux detected by these instruments, as shown in figure 5.3, commonly known as Ground-Level Enhancements (GLEs). Thus, detection of a GLE allows for a time window of up to tens of minutes before the arrival of the bulk of SEPs [43] that can be used to take protection measures against their effects.

Neutron monitors and muon telescopes are the main instruments used to measure cosmic rays arriving at the Earth's atmosphere.

Neutron monitors measure the integrated flux of secondary neutrons produced by cosmic rays in the stratosphere. They produce stable data, and long historical records are available, but their response is heavily influenced by their location. First, the energy threshold of GCRs that are able to produce secondary cosmic rays the monitor can observe is determined by the atmospheric cutoff (which is around 1 GV) and the geomagnetic location of the monitor, which de-

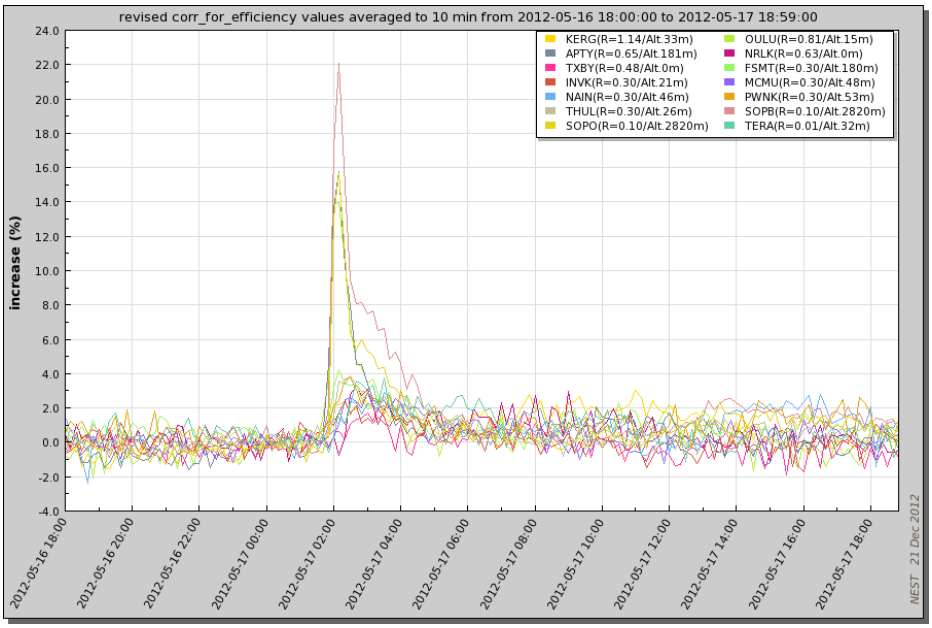


Figure 5.3: GLE example, registered by neutron monitors of the NMDB. Pictures from [42].

termines the vertical cutoff rigidity, ranging from 0 GV at the South Pole to 20 GV in South-East Asia, as shown in figure 5.4. And second, the Earth's magnetic field determines the incident direction of arrival of GCRs with energies below some hundreds of GeV, so the monitor's field of view cone is determined by the asymptotic cosmic ray directions at its location, which are far off the vertical. Consequentially, location determines the observable GCRs energy range and a single neutron monitor cannot provide incident direction information with adequate accuracy.

However, directionality can be obtained using a network of neutron monitors around the globe, combined with numerical models and tools. This is the goal of the Neutron Monitor Database (NMDB) [43], which combining the measurements of such a network results in a single energetic particle detector capable of measuring the energy spectra as well as the incident direction distribution of GCRs with high precision. The more rigidity is covered, the more sensitive the neutron monitor network is to cosmic ray energy, which, together with the necessities for directionality, requires a distribution of neutron monitors around the globe. Their current distribution, however, is not uniform and stations are concentrated in the northern hemisphere and Antarctica, with some important gaps due to areas not covered by any station.

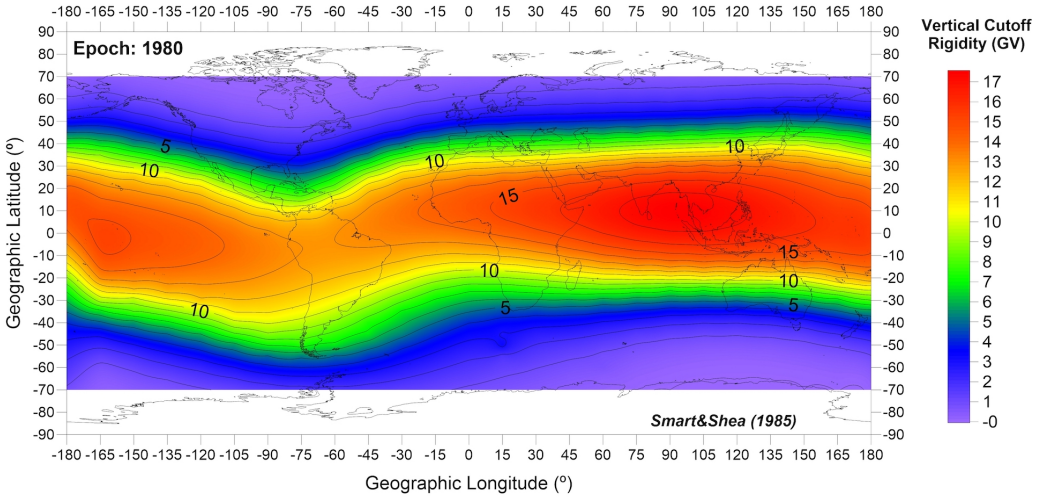


Figure 5.4: Map of vertical cutoff rigidities, from [44].

Regarding **muon telescopes**, muons produced by cosmic rays with energies from about 10 GV and above can be detected, and solar activity has a clear effect in their count rates. However, although FDs can be observed, GLEs can rarely be measured because of their higher energy threshold. What muon telescopes can measure, though, is incident direction of muons; and the Earth's magnetosphere has little or no effect in their propagation. The usual approach for this is to use two layers of individual, large surface scintillators arranged in a matrix separated by a distance and with a lead layer in between to act as a filter. This arrangement allows the calculation of a rough trajectory when a coincidence is detected in two scintillators, one of each layer, as shown in figure 5.5. The main issue with this approach is, of course, the large size of the instrument and the high cost if high directionality resolution is desired.

Just like with neutron monitors, muon telescopes are also very useful in space weather forecasting, either by analysis of flux or by studying anisotropies obtained with directional muon telescopes. In a similar manner to the NMDB, there is also a muon detector network, the Global Muon Detector Network (GMDN) that is helpful in space weather forecasting [46].

ORCA is a project to build a cosmic ray multi-detector facility and install it in the Juan Carlos I Antarctic Base (JCI) in Livingston Island (62°39′46″S, 60°23′20″W, 12 m asl), making it the first Spanish cosmic ray observatory in Antarctica. The project has two main scientific goals: to analyze the solar cycle and activity by means of cosmic rays observation at ground level, and to be a space weather instrument by providing measurements in real time. Given its location

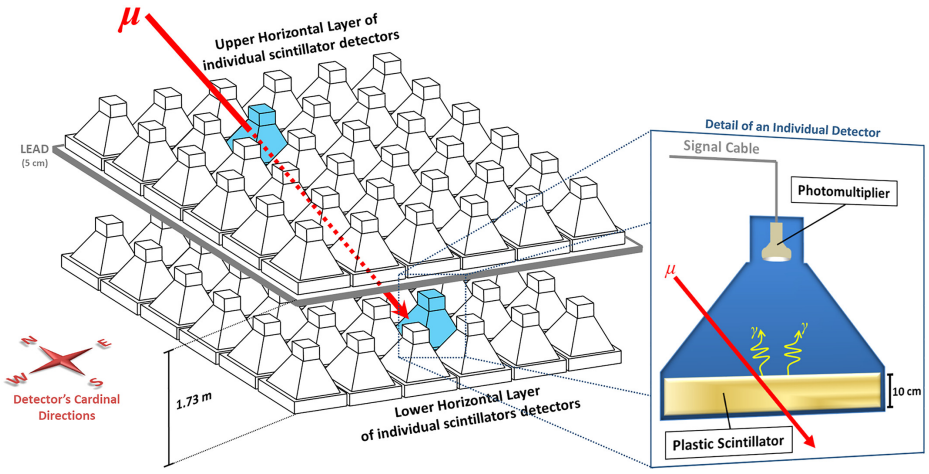


Figure 5.5: Sketch of the Nagoya muon detector, from [45].

and detector composition, ORCA would provide counting rates for primary cosmic rays above 3 GeV, and direct measurement of incident directions for cosmic rays above some tens of GeV, making it capable to observe GLEs, FDs and also GCR anisotropies. These observations, in turn, enables the evaluation of propagation models for cosmic rays through the atmosphere and magnetosphere.

As mentioned, real time data uploading capability would make it a space weather instrument; and more specifically, regarding neutron measurements this detector would cover a geographical gap in the NMDB network. Also, it would work in a region of low rigidity, around 3.52 GV, which apart of making it capable of observing GLEs and FDs, is complementary to that of the CaLMa facility [47], a neutron monitor already in service in Guadalajara (40°38′33″N, 3°9′45″W, 708 m asl) also built by the SRG of the Universidad de Alcalá, which is in a region with a rigidity of around 7 GV and is already serving data to the NMDB in real time.

The following sections provide a description of the project design and construction, focusing on the parts developed by the SRG of the Universidad de Alcalá and more specifically, in those where the ARACNE prototype was used as a fundamental part of the instrumentation system.

5.1.1 Project overview

ORCA is a multi-detector facility designed to be installed at the Juan Carlos I Antarctic Base and, additionally, perform a latitudinal cosmic ray flux study along the Atlantic Ocean during its transition from Spain to its final location. It is

composed of two main detector subsystems: CALMA-A, which is an arrangement composed of a neutron monitor (NEMO) and large area scintillator-based muon telescope (MITO); and TRISTAN, a tracking detector sensitive to muons and electromagnetic showers based on Resistive Plate Chambers (RPCs). Figure 5.6 shows a preliminary diagram of the arrangement.

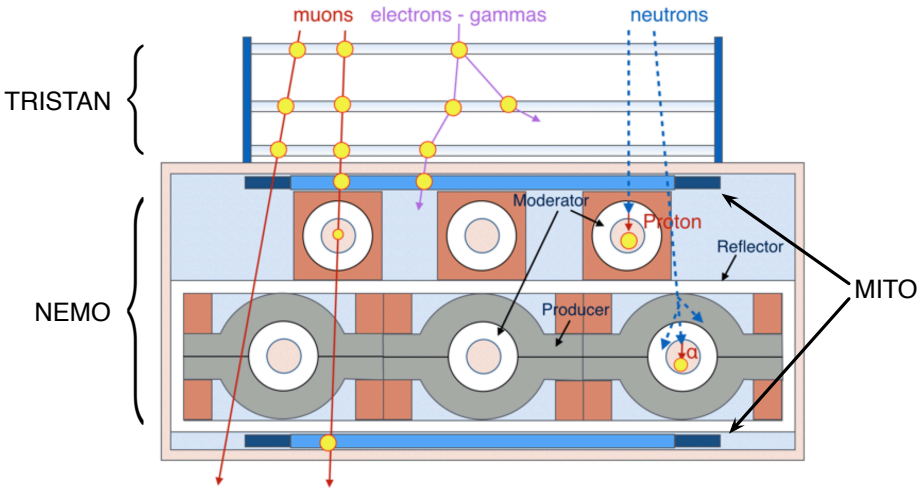


Figure 5.6: Diagram of the ORCA detectors arrangements, from [48].

The detectors and their instrumentation systems, as well as all the ancillary systems that provide other basic functionality, are housed in a 20ft maritime container, as depicted in figure 5.7. This container is thermally isolated and features a heat pump and air conditioning unit, and also provides energy for all systems either from external sources or through a built-in autonomous energy system based on solar panels and batteries. Additionally, an Ethernet connection to an external network is also provided, to allow remote management and data transmission.

Both CALMA-A and the container facilities are responsibility of the SRG team of the Universidad de Alcalá, while TRISTAN is provided by the Universidad de Santiago de Compostela. It must be noted that, although in the original project all detectors were arranged inside the ORCA container, practical reasons required the installation of TRISTAN in a nearby but separate laboratory both during transit and in their final location at the base.

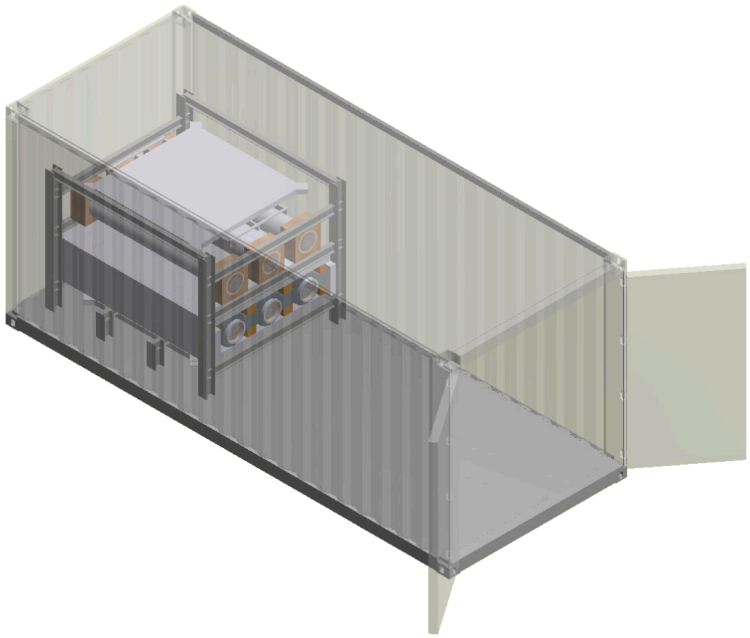


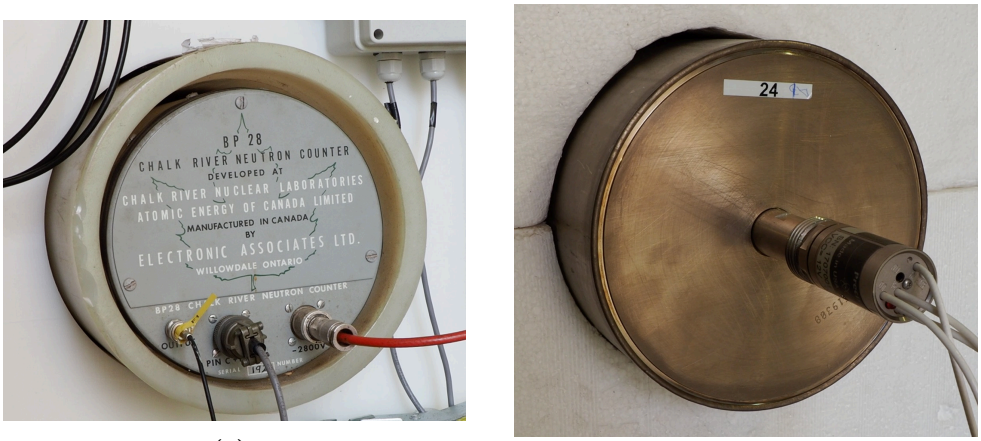
Figure 5.7: Preliminary diagram of the ORCA container and the disposition of the detector arrangement inside it, from [48].

5.1.2 NEMO

The NEutron MOnitor (NEMO) section of CALMA-A is composed of two sets of three counter tubes, one above the other, as depicted in figure 5.6. The first set is a direct heritage of the CaLMa neutron monitor and is composed of three BP28 counter tubes with their corresponding lead and polyethylene pieces that act as producer and moderator elements, so that they meet the NM64 neutron counter tube standard. Considered individually, this set forms a 3NM64 neutron monitor. The second set, that we call 3BNM, is formed by three LND2061 counter tubes that are bare, in the sense that they lack any lead or polyethylene surrounding them. Once these two counter tube sets are integrated in the NMDB, their short station names will be ORCA and ORCB respectively.

Counter tubes require a high voltage power supply for biasing, and generate a charge pulse when an event occurs that must be dealt with by a preamplifier. For this reason, a biasing and preamplifier module installed on the tube header is needed. The BP28 tubes include such a device, requiring a $-2800\,\mathrm{V}$ HV supply for biasing and $13\,\mathrm{V}$ supply for the preamplifier (figure 5.8a), but the LND2061 tubes require a separate module so each of them is fitted with a Precision Data

Technologies PDT10A-HN-12V-BF3 preamplifier. These preamplifiers require a 12 V power supply, and a biasing voltage of 1800 V (figure 5.8b).



(a) **(b)**

Figure 5.8: NEMO counter tube headers, BP28 (left) and LND2061 (right).

All preamplifiers generate a TTL pulse that is transported by coaxial cables to an ad-hoc data acquisition system, which is capable of counting the events for all channels and produce the appropriate science data. This data can be stored locally or transmitted through the network, ideally to the NMDB in real time. In-depth description of the data acquisition system of NEMO is out of the scope of this work, but a study on the construction and performance of this counter tubes and the CaLMa neutron monitor can be found in [47] and [49].

5.1.3 MITO

As explained in a previous section, unlike neutrons, muons can not only be measured in terms of flux; but their incident direction can also be determined, typically using telescopes composed of a large number of individual detectors. For this reason, this type of muon telescopes is usually very large and expensive. For example, facilities such as the Nagoya Multi-Directional Muon Telescope, in Japan (depicted in figure 5.5) uses seventy-two 1 m^2 scintillators with their corresponding PMTs, in two 6x6 layers separated 1.73 m. Other facilities use a different number of elements, such as Hobart (Australia) (2 layers, 4x4), Sao Martinho da Serra (Brazil) (2 layers, 4x9) or a different but equivalently large and expensive approach, such as the Kuwait telescope, which uses 186 proportional counter tubes in four layers.

The Muon Impact Tracer and Observer (MITO) telescope concept is a novel design that tackles on the size and cost of directional muon telescopes, providing directionality information with only two large size scintillators and eight Photomultiplier Tubes (PMTs) in a much more compact size. The full details on MITO, its motivations and characteristics, simulation results and a comparison to the aforementioned facilities can be found in [40], but an outline of its design and the features relevant to the data acquisition system based in the ARACNE concept that was designed for it are detailed in the following sections.

Design and operating principle

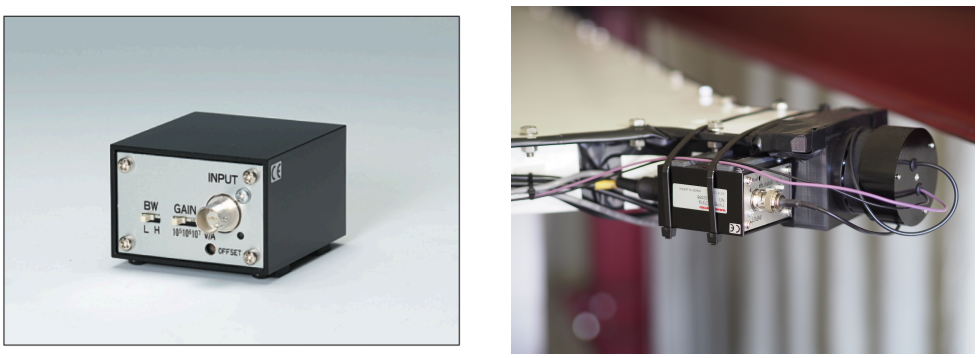
The detector section of MITO consists of two identical devices, stacked one above the other at a distance of 137.5 cm, each of which is essentially composed of a Saint-Gobain BC-400, $100 \times 100 \times 5$ cm plastic scintillator and four Hamamatsu R2154-02 Photomultiplier Tubes (PMTs). The PMTs are arranged around the scintillator as depicted in figure 5.9a at a distance of 50 cm from each side, and an aluminum housing is used to achieve the described arrangement and prevent the passage of light from the outside. This housing provides four triangular prism-shaped light guides and is painted in matte white, as shown in figure 5.9b.

(a) (b)

Figure 5.9: Left: sketch of one of the MITO devices. The scintillator dimensions are $100 \times 100 \times 5$ cm and the PMTs are located at 100 cm from its center. **Right**: open MITO housing and scintillator arrangement, during assembly. The PMTs are not yet installed at this stage.

To complete the construction of each MITO device, the PMTs are fitted with

appropriate sockets (model E1198-07) and 3D-printed flexible fixtures are used to safely attach them to the corners of the aluminum housing. Then, each photomultiplier is fitted with a Hamamatsu C7319 amplifier unit [50], which is installed close to the PMT socket to minimize signal noise as shown in figure 5.10, and its output is taken to the data acquisition equipment with coaxial cabling. This amplifier requires a $\pm 15\,\mathrm{V}$ power supply, whereas the PMT itself requires a high voltage power supply of up to $1750\,\mathrm{V}$.



(a) **(b)**

Figure 5.10: Left: Hamamatsu C7319 amplifier. **Right**: detail of the PMT assembly.

The arrangement described is a consequence of the working principle of MITO. When a charged particle passes through the scintillator, a certain amount of light proportional to the energy deposited by the particle is generated. Photons propagate in all directions, eventually coming out through the lateral sides of the prism, where they are led by a light guide to the corresponding PMT, as shown in figure 5.11a. There, light is collected and a charge proportional to the amount of photons gathered is generated by the PMT.

This charge is collected by an amplifier, generating a linear voltage pulse of proportional amplitude. As a result, the pulse amplitude carries information both about the deposited energy and the distance between the impact point and the corresponding PMT. Since the deposited energy in an event is the same for all four PMTs, differences in the pulse amplitude must thus correspond to differences in distance from the impact point to each PMT (see figure 5.11b), and consequently, by using these amplitudes an algorithm could be envisaged to reconstruct the impact point at the scintillator.

Finally, since MITO is composed of two of these devices separated by a distance as shown in 5.6, by calculating the impact points of a particle that traverses both devices, its trajectory can be calculated, as depicted in figure 5.12. Optionally,

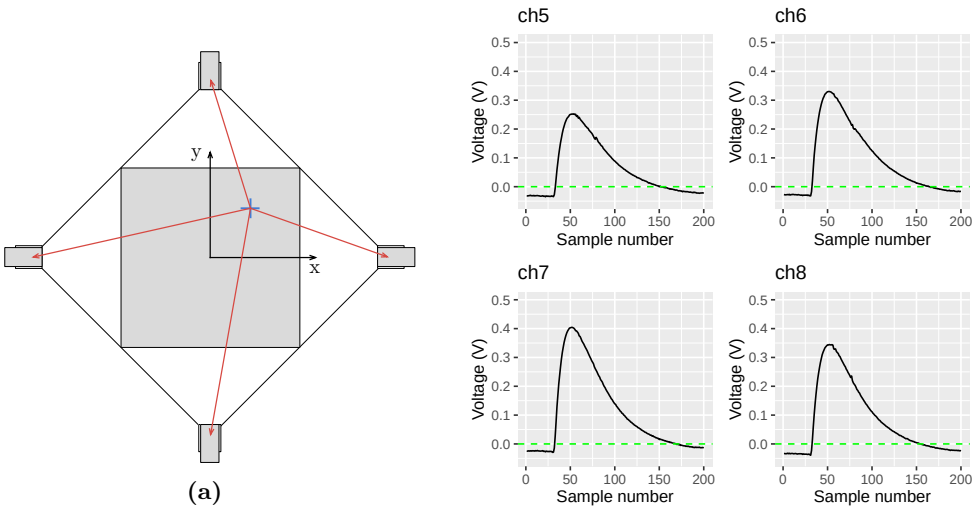


Figure 5.11: Left: Diagram of the MITO working principle for one of the layers. **Right**: signals obtained from the four PMTs when a particle impacts the scintillator.

a lead layer between the two devices can be installed to work like a filter for unwanted particles, depending on the application. In the case of ORCA, this explains the arrangement of detectors of figure 5.6, where NEMO is between the two MITO devices precisely for this reason; the lead of the 3NM64 section of NEMO serves a dual purpose, being used also as the filtering lead layer of MITO.

Naturally, given the square shape of the scintillators the Field of View (FOV) is not the same in all directions: it is wider along the diagonals, and narrower along the X and Y axis. And it also depends on the distance between the MITO planes, so for a distance of 1.365 m and given the dimensions of the scintillators, this FOV ranges from 72.4° to 92°, and a total solid angle of 1.42 sr [40].

5.1.4 ORCA construction

The main detector elements of NEMO and MITO described in the previous sections are mounted on a structure made up of steel beams inside the ORCA container (figure 5.13). It is divided in four levels, as illustrated in figure 5.14.

From bottom to top, levels 1 and 4 are dedicated to securely install and align the two Bottom and Top MITO devices over two rails (figure 5.15).

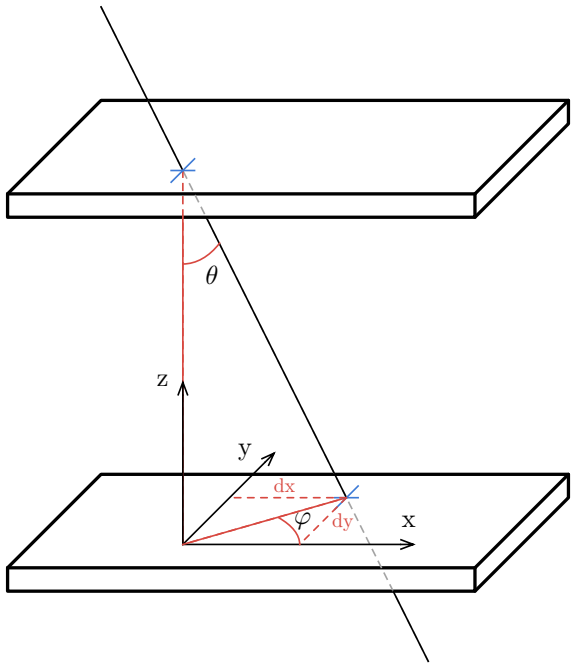


Figure 5.12: Sketch of the operating principle of MITO. Given the impact points on the two planes, azimuth and elevation angle of the particle can be calculated.

Level 2 is designed to install the 3NM64 section of NEMO, and as such is heavily reinforced to withstand the weight, which is almost 6 t, not only statically but during the trip to Antarctica. Figure 5.16 shows the internal disposition of the tubes, lead pieces and polyethylene sarcophagus during its construction. Finally, level 3 is designed to house the bare 3BNM section of NEMO. Figure 5.17 shows the completed NEMO arrangement.

Besides the steel structure, a rack is used to house all the supporting electronics, which includes all the power supplies, the data acquisition systems for MITO and NEMO and the rest of ancillary systems that provide the remaining services of ORCA.

Data acquisition systems

Three data acquisition systems are present in ORCA, one of them for NEMO and two for MITO, each of them mounted in a standard 19” box that is then attached to the instrumentation rack.

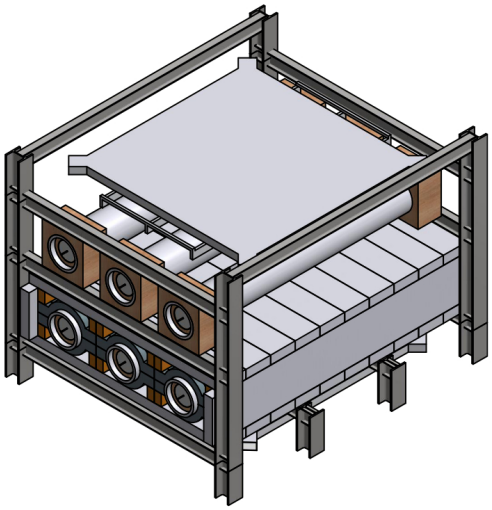


Figure 5.13: ORCA structure render.

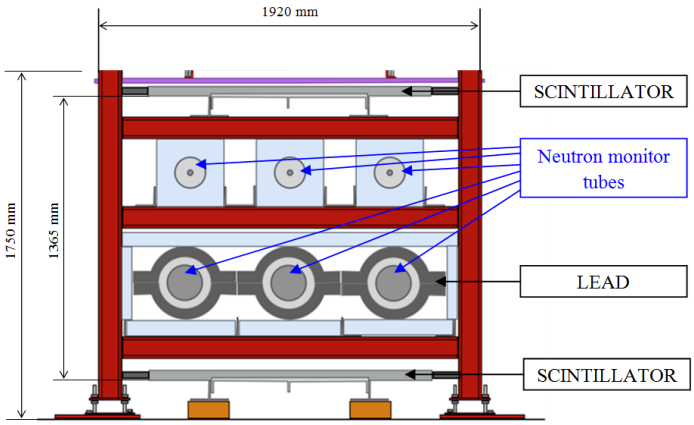
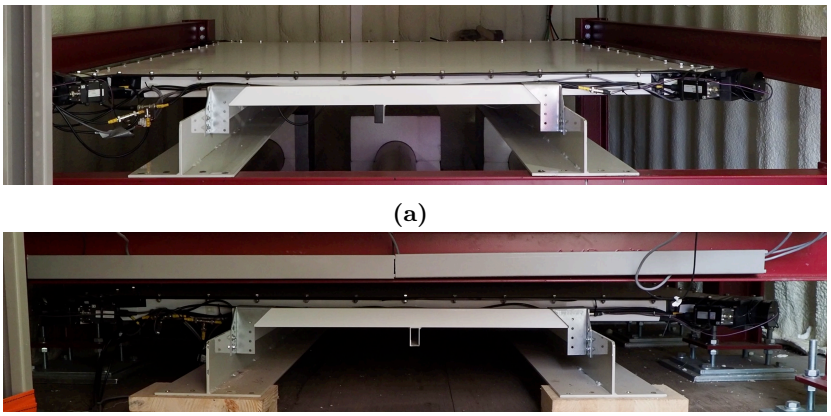


Figure 5.14: ORCA Detector distribution.

The acquisition system for NEMO is an in-house design named NOAS. It will not be studied in-depth here, but it consists of a digital input adapter board and an FPGA-based capture board that is able to count neutron events on all channels simultaneously. This board is controlled by an SBC, and also implements interfaces for other ancillary systems of ORCA, such as the pressure and temperature sensor system. This board features its own low voltage power supply, and in general, is a downsized version of the next generation data acquisition system of CaLMa, so it is capable of sending real time data to the NMDB depending on the communications capability of the base.



(a)

(b)

Figure 5.15: MITO devices installed in ORCA.

Figure 5.16: NEMO 3NM64 section during construction. 240 lead pieces weighing around 21.4 kg are used, and the total weight is almost 6 t.

For MITO, two data acquisition systems are used in parallel, with the eight signals produced by the amplifiers fed to both of them simultaneously through a splitter. This allows, among other things, the cross-validation against each other. One of the systems is the ARACNE prototype that has been thoroughly described in chapter 4. The other is named SAS, and in contrast to the ARACNE system it processes the linear pulses produced by the detector analogically. It is focused on pulse counting and coincidence, since it does not determine pulse

Figure 5.17: Finished NEMO detector.

height or shape, and consists of an input amplifier, a pulse discriminator for which it includes an LLD and a ULD, and coincidence logic to detect coincidences between channels. An SBC counts and registers these coincidences, and also provides remote monitoring and control.

Power supplies

Several power supplies, both of high and low voltage are required in ORCA, even though some systems have their power supplies built-in. As such, several modules have been built to provide these supplies:

- A dual High-Voltage (HV) power supply for the counter tubes in NEMO, which provides −2800 V for the three BP28 tubes and 1800 V for the three LND2061 tubes.
- A quad HV power supply for the PMTs of MITO. This means each supply must be used for two PMTs. These supplies are adjustable.
- A Low-Voltage (LV) power supplies module. This module includes several individual power supplies dedicated to different parts of ORCA, such as SAS, the MITO amplifiers, the BP28 and LND2061 amplifiers or the MITO high voltage power supplies.

A summary of the ORCA subsystems and their voltage requisites and supplies is displayed in table 5.1.

Module/Element	Required voltages	Supplied by
MITO PMTs	8x Variable HV	MITO HV supplies module (4 supplies)
MITO amplifiers	8x ±15 V	LV supplies module (2 ±15 V supplies)
MITO HV supplies module	±12 V	LV supplies module
SAS module	±5 V	LV supplies module
ARACNE prototype	5 V	Built-in
NEMO BP28 tubes	−2800 V	NEMO HV supplies module
NEMO LND2061 tubes	1800 V	NEMO HV supplies module
NEMO BP28 amplifiers	13 V	LV supplies module
NEMO LND2061 amplifiers	12 V	LV supplies module
NOAS module	±5 V, 12 V	Built-in

Table 5.1: ORCA voltage requirements and supplies.

Ancillary systems

Apart from the acquisition systems themselves, additional equipment is necessary for ORCA to function. Pressure and temperature, for example, affect neutron and muon measurements. The date and time of registrations is important, and a reference is not available if internet connectivity is not guaranteed. The storage capacity of SBCs is usually limited, and data integrity is an important issue. And finally, networking for the system components must be supported by adequate hardware. For this reason, additional systems are installed in the ORCA rack:

- A Vaisala PTU300 combined pressure, humidity and temperature transmitter.
- A GPS time server providing time synchronization through the Network Time Protocol (NTP) protocol in the local network.
- Networking equipment composed of a low power Ethernet switch and a router, both for systems interconnection and communications with the outside facilities (i.e. the JCI network).
- A low power and high capacity SSD-based Network Attached Storage (NAS) server, where the acquisition systems can safely store data.

All these systems are commercial, and have their own power supplies.

Figure 5.18: Complete ORCA arrangement inside the container, after installation at the base.

Final ORCA arrangement

The elements described in the previous sections have been installed inside the ORCA container, with the instrumentation rack besides the structure that houses the detectors. AC power distribution has been arranged inside the container so that it is easily accessible, and lightning has been installed. Finally, networking as well as external power is brought from the outside through sealed access ports in the back of the container.

Figure 5.18 shows the completed arrangement with all the detectors assembled and the electronics rack in its final position, inside the container (Figure 5.19). With this configuration, ORCA has been continuously gathering data until the base closed in 2019 and energy run out, and continuously from January 2020 to January 2021 thanks to the renewable energy systems of the base. Also, it's been remotely accessible from Spain while the base has been open during both campaigns, for data downloading, maintenance and monitoring. After the base closed in 2020, ORCA has been sending measurement data to the JCI registering facilities, which has been gathered remotely and relayed to us periodically.

Figure 5.19: View of the ORCA arrangement from outside the container, in Antarctica.

5.2 ARACNE as data acquisition system for MITO

Given the described working principle and the type of signals produced by the MITO detector outlined in section 5.1.3, the requisites for the instrumentation system needed for the application can be identified: Eight signals from the eight PMTs must be dealt with simultaneously, and both counting and pulse amplitude must be registered, since Pulse Height Analysis (PHA) must be performed on them for particle trajectory determination. Coincidence filtering (e.g. preserving only data when there's a pulse at certain channel combinations) must be supported, for example to select only events where a signal is present in all eight channels for trajectory calculations. And additionally, the capability to perform Pulse Shape Analysis (PSA) in post-processing is desirable for several reasons; full count and pulse amplitude reconstruction, baseline correction, rejection of pulses that may present pile-up, or even assess the possibility of identifying particle characteristics depending on the pulse shape.

This is an ideal scenario to use a digital instrumentation chain based in the ARACNE concept; a multichannel system that fully and synchronously digitizes analog input signals, capable of performing complex processing digitally and even fully, but selectively, record pulse shapes for post processing. And, at the same

time, with a very low energy consumption suitable for usage in the harsh conditions of the Antarctic base. As such, the first ARACNE-based prototype was designed and built with the requisites of MITO in mind, as described in chapter 4, but suited to be used in other detectors or applications applications as well.

Its usage in MITO is described in this section, and the obtained results are presented in section 5.3.

5.2.1 Application software description and capabilities

As it has been presented in detail in chapter 4, the ARACNE prototype provides an adequate platform to implement the MITO acquisition chain:

- A hardware platform with an input stage capable of dealing with the signals produced by the C7319 amplifiers of MITO [50]. These amplifiers were recommended by Hamamatsu for use with the R2154-02 PMTs, have an output impedance of 50 Ω and a maximum output voltage of 2 V in these conditions, which is reduced to 1 V if used with an impedance matching splitter. Otherwise, an external attenuator can be used to adapt the input to the prototype input range. Following the input stage, the digital conversion stage far exceeds the necessities of these signals, rated at 200 kHz maximum bandwidth.

- A configurable IPCore capable among other things of detecting pulses in all eight channels, auto-triggering their capture simultaneously together with a precise time stamp, and notifying the SBC of their availability. The pulses produced by the amplifier have been measured to have a duration of around 4 µs so for a sample period of 25 ns (40 MS/s), more than enough for PSA, a FIFO size of 512 samples per channel has been configured to allow more than twice the storage of one pulse.

- A set of software libraries to ease writing an application that performs the desired measurements.

As such, an application software has been written leveraging the system software libraries and the capabilities of the IPCore to perform the desired measurements of the MITO telescope. The software is a Python program that can be parametrized and run unattended, and that is automatically launched by the SBC running Linux. It is capable, among other things, of:

- Registering per-minute event counts, both in several coincidence modes and individually, per channel.
- Registering per-event pulse amplitude values of all channels.
- Registering per-event full pulse sample values of all channels.

This allows several types of flux measurements, PHA and PSA as specified. Details on how each type of measurement is performed and stored is provided in the following sections. Additionally, the software supports several configuration parameters to alter the behaviour and experimental data gathered:

- Registering of pulse amplitudes or pulse shapes can be enabled or disabled, and pulse shape registering can be restricted to those events when there's coincidence among all 8 channels. Event count registering is always enabled.
- Event count data can be optionally sent every minute to a UDP server belonging to the data infrastructure system of the base, for remote gathering when the base is closed during the winter.
- The FIFO parameters (depth and pre-trigger delay) can be configured.
- The trigger threshold levels can be configured. It is also possible to arbitrarily define which channels to use for triggering.
- It is possible to reject events where pulse clipping occurs (i.e. pulse amplitude is abnormally outside the input range of the ADC).
- It is possible to define how long the program will run before restarting. The default is 24 hours.

On a side note, since each input channel corresponds to a PMT (where channel 1 corresponds to PMT 1, etc.) and their position determines both impact coordinates and particle trajectories, it is important to mention the way channels are numbered and their position with respect to the ORCA container. Channels 1 to 4 correspond to the lower MITO device (also known as "Bottom") and channels 5 to 8 correspond to the upper level (also known as "Top"). Figure 5.20 illustrates the distribution of channels 1 to 4 with respect to the lower MITO device and the container: channel 1 is closest to the instrumentation rack, nearest to the container door, and the numbering continues clockwise. Channels 5 to 8 are arranged in an identical way on the upper level.

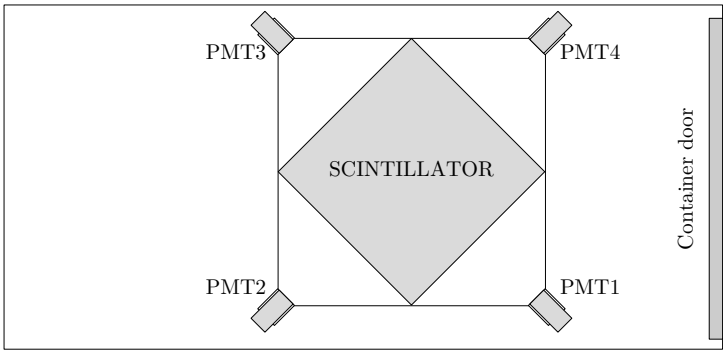


Figure 5.20: PMT numbering diagram with respect to MITO and the container. The Bottom MITO device is shown and the Top device follows the same numbering scheme for PMTs 5 through 8.

Program flow

The data capture and processing software is a mostly cyclical program, given the nature of the task at hand. It is composed of two main parts, an initialization stage and a capture stage. In the latter, the program waits for the IPCore to notify through a GPIO signal that an event has occurred and there's data available to read from the capture FIFOs. Once this happens, the program reads all data and performs an analysis on every channel to determine if a pulse exists and its amplitude, and if there's been any kind of inter-channel coincidences. Then it updates the appropriate counters and if the current minute is finished, records the event count data. Finally, if it is configured to register pulse amplitudes and/or pulse shapes it does so, and if the capture period has not expired, goes back to waiting for the next event. A more detailed diagram of the program flow is represented in figure 5.21.

5.2.2 Event count registering

The first type of results generated by the MITO acquisition chain is per-minute event counts. These counts are stored in a Comma Separated Values (CSV) file which includes the following information in each line:

- Column 1: Time stamp of the minute corresponding to the counts in the line.
- Column 2: Total number of events detected (regardless of coincidences, e.g. a pulse was detected at any channel).

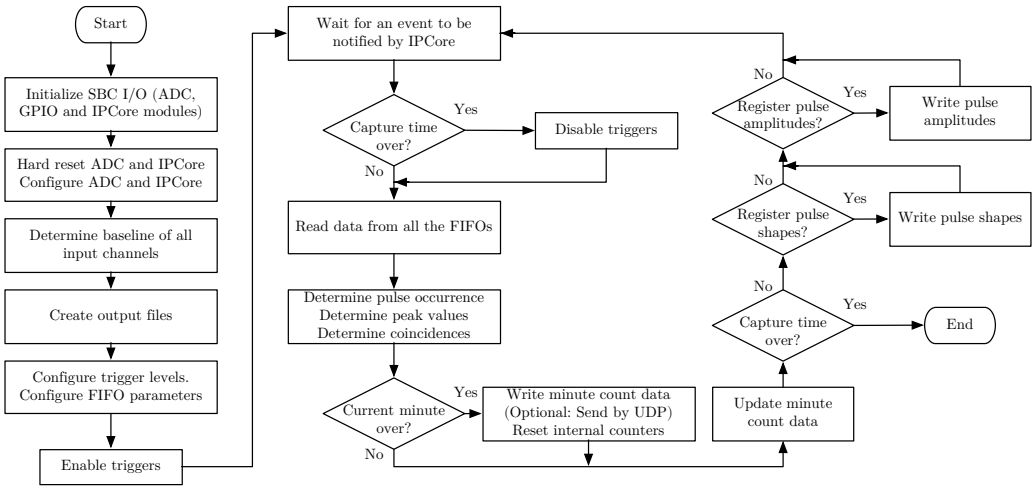


Figure 5.21: Program flow of the MITO software.

- Column 3: Number of events detected where there is an 8-way coincidence (e.g. a pulse was detected in all channels).
- Column 4: Number of events detected where there is a 4-way coincidence in the bottom level.
- Column 5: Number of events detected where there is a 4-way coincidence in the top level.
- Columns 6-13: Number of events detected at each of the channels, from 1 to 8.

This data is key to perform particle flux studies. A file is generated for each day, and the following is a portion of an event count file from February 23, 2020.

```
1 2019-02-23 00:00:00.012389,6556,648,2518,4492,2778,2665,2607,2683,4577,4649,4637,4618
2 2019-02-23 00:01:00.016109,6549,633,2448,4541,2757,2609,2565,2627,4613,4700,4693,4663
3 2019-02-23 00:02:00.021460,6557,642,2467,4525,2760,2627,2586,2651,4581,4673,4668,4650
4 2019-02-23 00:03:00.011135,6630,683,2452,4640,2761,2619,2550,2628,4712,4805,4800,4769
5 2019-02-23 00:04:00.012768,6605,650,2472,4591,2745,2623,2580,2640,4669,4749,4737,4708
```

In this particular example it can be seen that the top MITO device registers more events, both in coincidence and per individual channel, which is due to the effect of the lead layer. Events in coincidence at the top and bottom are slightly lower than on individual channels at both levels, which is normal since an impact may not be ”seen“ by all PMTs on a level. Finally, events in coincidence of all eight channels are lower than on top and bottom, since many particles, depending on

their trajectory, cross one of the levels but not the other depending on trajectory and also because of the effect of the lead layer.

5.2.3 Pulse amplitude registering

The second type of data registered is per-event pulse amplitude values for all channels. For this task, the software samples the signal level at all channels every time it is launched to establish the baseline signal level. Then, on each event, it determines the maximum sample value at each channel and calculates the amplitude with respect to the baseline of the channel. If the amplitude is above a certain threshold, it considers that a pulse has been registered; otherwise, it considers the signal is absent or at most, noise. Finally, it determines if some type of coincidence has occurred between channels. The results are stored in a CSV file which includes the following information in each line.

- Column 1: Time stamp of the event corresponding to the line.
- Columns 2-9: Peak value relative to the baseline level at each of the channels. Each column corresponds to a channel, from 1 to 8. A 14-bit value is given, where 8192 corresponds to 1 V (half of the ADC input range).
- Columns 10-17: Sample number from the beginning of the capture window where the peak value has occurred. Each column corresponds to a channel, from 1 to 8.
- Columns 18-20: Three flags with values 0 or 1 that indicate that a specific type of coincidence was detected (1) or not (0). The first flag indicates an 8 channel coincidence, the second a coincidence in the 4 Bottom channels and the third a coincidence in the 4 Top channels.

This data is key to perform Point of Impact (POI) and trajectory calculations for anisotropies measurement. Given the amount of data generated, a file is generated every hour. The following is a portion of a pulse amplitude values file from February 17 2020.

```
1 2020-02-17 12:00:00.382158,2659,1203,1216,2265,37,25,16,53,21,22,24,21,46,49,47,45,0,1,0
2 2020-02-17 12:00:00.389438,1620,1414,1668,2307,654,1022,1256,1339,20,22,23,20,28,24,24,20,1,1,1
3 2020-02-17 12:00:00.396031,1598,2451,1703,2793,35,34,37,41,25,20,24,26,2,3,1,4,0,1,0
4 2020-02-17 12:00:00.401363,34,30,44,42,1166,1161,1441,2414,42,27,48,47,20,20,22,21,0,0,1
```

In this example it is possible to see an event that produces pulses in all channels, and thus is considered as an 8-channel coincidence. The rest are events that

only produce pulses at the Top or Bottom levels, since the signal levels are only significant at the four channels corresponding to that level.

5.2.4 Pulse shape registering

Finally, the last type of data that the acquisition system can record is pulse shape at all channels for an event. In this case, data is also stored in a CSV file, but eight lines are added to the file on each event, one per channel, which includes the following information:

- Column 1: Time stamp of the event, which will be the same for the eight lines corresponding to each channel.
- Column 2: The channel number.
- Columns 3 and 4 : The configuration of the FIFO when the pulse samples were captured. This includes the pre-trigger delay and FIFO depth, which in this case will indicate the sample number where the trigger occurred (not to be mistaken with the sample number when the peak occurred) and the number of samples that are coming next.
- Columns 5 and beyond: A series of sample values that represent the pulse shape. These are raw values, as generated by the ADC, without baseline correction as in the case of pulse amplitudes. As a reminder, the ADC generates values from 0 to 16383 for the $\pm 1\,\mathrm{V}$ range, so 8192 would be $0\,\mathrm{V}$.

This data allows performing PSA if desired, but also to do any kind of signal reprocessing such as rejection of undesired pulses, recalculation of the baseline and pulse amplitudes, digital noise reduction, etc. The only disadvantage is the large amount of data produced and large processing overhead. When this mode is enabled a file is generated every hour, which is already very large. The following is a portion of a pulse data file from January 5th 2020, edited for clarity, where only the data corresponding to an event for channels 3 and 4 is represented. In this example, 200 samples for each pulse are registered, which at 25 ns per sample amount for 5 µs per pulse. For clarity, the part of each row corresponding to the samples has been put in separate lines of 15 samples, starting at lines 2 and 18 of the listing for each channel. In this case, the trigger point is at sample number 35, and some negative bias can be observed on the baseline (e.g. samples before the start of the pulse should be around the 8192 value, not 7900).

```
1  2020-01-05 04:00:01.469880,3,35,200,
2  7951,7946,7947,7946,7946,7946,7948,7947,7941,7942,7947,7939,7945,7942,7944,
3  7943,7932,7941,7939,7942,7946,7947,7942,7933,7934,7945,7937,7941,7935,7939,
4  7924,7931,8259,8878,9656,10294,10843,11314,11751,12139,12500,12792,13059,13276,13454,
5  13597,13705,13789,13837,13888,13927,13924,13911,13913,13829,13837,13706,13718,13651,13596,
6  13478,13428,13340,13241,13151,13056,12954,12872,12775,12692,12600,12502,12413,12237,12265,
7  12117,12015,11932,11862,11700,11658,11554,11472,11381,11282,11204,11117,11032,10946,10865,
8  10794,10712,10641,10570,10500,10438,10361,10314,10226,10170,10102,10048,9995,9940,9887,
9  9838,9787,9747,9692,9661,9619,9577,9532,9501,9452,9424,9375,9344,9302,9268,
10 9236,9200,9165,9132,9099,9071,9044,9008,8980,8963,8935,8902,8884,8866,8835,
11 8813,8797,8782,8750,8732,8720,8694,8683,8656,8641,8621,8598,8594,8564,8551,
12 8532,8515,8488,8480,8465,8448,8429,8417,8388,8381,8370,8352,8342,8324,8311,
13 8306,8283,8283,8266,8259,8255,8243,8236,8233,8221,8219,8205,8204,8187,8183,
14 8188,8177,8182,8166,8160,8154,8146,8145,8133,8131,8133,8121,8111,8106,8106,
15 8102,8090,8088,8079,8077
16
17 2020-01-05 04:00:01.469880,4,35,200,
18 7931,7927,7934,7931,7938,7934,7925,7920,7933,7922,7934,7931,7930,7920,7939,
19 7918,7941,7920,7927,7924,7927,7920,7926,7920,7921,7922,7922,7921,7925,7921,
20 7929,7885,8119,8659,9384,10028,10597,11070,11519,11933,12308,12635,12924,13154,13345,
21 13506,13650,13763,13832,13903,13958,13974,14004,14048,14019,13995,14010,13915,13851,13785,
22 13738,13650,13606,13504,13421,13329,13230,13120,13022,12922,12826,12726,12635,12548,12446,
23 12335,12237,12149,12032,11954,11837,11730,11652,11562,11477,11379,11294,11198,11119,11027,
24 10951,10870,10795,10718,10649,10572,10513,10426,10374,10292,10229,10175,10113,10044,10005,
25 9942,9886,9824,9773,9715,9674,9614,9578,9539,9486,9438,9405,9345,9317,9265,
26 9230,9189,9156,9116,9095,9052,9032,8991,8963,8935,8917,8878,8866,8833,8824,
27 8786,8772,8746,8731,8717,8690,8668,8653,8619,8627,8586,8577,8549,8548,8521,
28 8516,8484,8465,8450,8440,8419,8413,8382,8375,8358,8359,8332,8324,8302,8305,
29 8273,8274,8258,8256,8243,8244,8231,8217,8214,8213,8199,8204,8198,8188,8175,
30 8169,8165,8163,8151,8157,8140,8145,8126,8120,8117,8126,8101,8103,8090,8093,
31 8086,8086,8068,8076,8065
```

By looking at the data, it is possible to distinguish the rising edge of the pulse, happening around sample 35 as expected (lines 4 and 20 of the listing for channels 3 and 4 respectively). A graphical representation of pulses similar to these has already been provided on section 4.3.7, figure 4.42.

5.3 Results

As shown in the previous section, the ARACNE prototype is capable of registering particle counts in several coincidence and non-coincidence modes and simultaneous per-event pulse heights and pulse shapes. As such, it's been installed at the Juan Carlos I Antarctic Base for more than two years now, the last one registering data without interruption except in the short periods of time when the renewable energy systems of the base were depleted and shut off temporarily. Even in those cases, the prototype has restarted correctly and started registering data again. This is a proof of its robustness and the feasibility of the concept as a platform for nuclear instrumentation systems, specially those where resilience, low power consumption and remote management capabilities are very valuable.

This period of time working at the base has already produced a large amount of useful data that allows the study of particle flux as well as to corroborate the feasibility of the MITO directional muon telescope concept. The following sections will show the results obtained from of the analysis of data gathered by

the ARACNE prototype.

5.3.1 Preliminary results and software gain correction

Photomultiplier tubes are devices difficult to manufacture, and units are typically different in their response even for the same model; their sensitivity varies, which means that they have different gains and different optimal HV bias values. This could be a problem for the MITO principle of operation, since determining the POI at each plane requires either an identical response in all PMTs of a plane or the appropriate corrections to the signals they produce in post processing. For this reason, a calibration or a response analysis must be performed for each device. In our case, an experimental bias adjustment has been done, and then statistical software corrections have been applied. Details about the bias adjustment can be found in [40], which resulted in voltages from 1200 V to 1400 V to achieve a similar number of counts per minute on all channels.

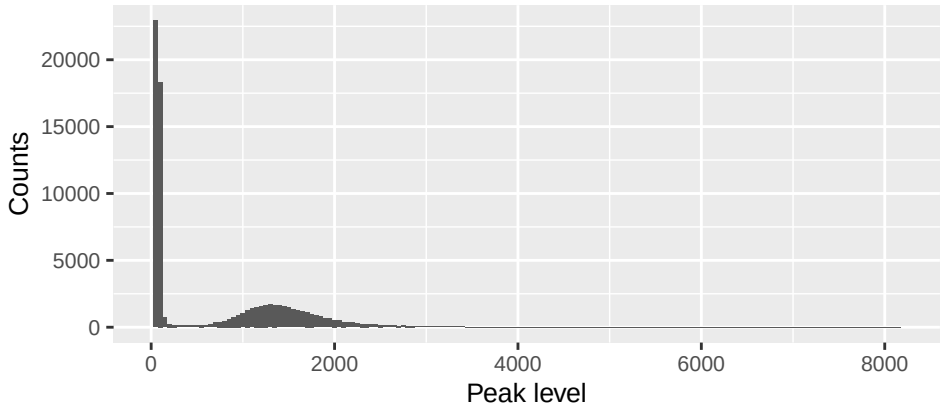


Figure 5.22: Histogram of pulse amplitudes at one of the MITO channels.

The first preliminary test that should be performed is to check the response of individual PMTs is as expected. To achieve this, an energy spectrum of the detected particles must be generated, a function typically performed by a Multi Channel Analyzer; in our case, the data produced by the acquisition chain is processed in software using a program written in R providing this functionality.

Figure 5.22 shows the histogram of the pulse heights registered at one of the eight channels of MITO. This result is the expected response, with a high amount of low level pulses produced by background noise or undesired particles, then a concentration of samples around a single value representing the energy

deposited by the particle the detector is sensitive to. The low level pulses are usually registered by one PMT, so they can be filtered out by considering only the events in coincidence with other PMTs. Figure 5.23 shows the histogram obtained from the same channel after keeping only the events in coincidence.

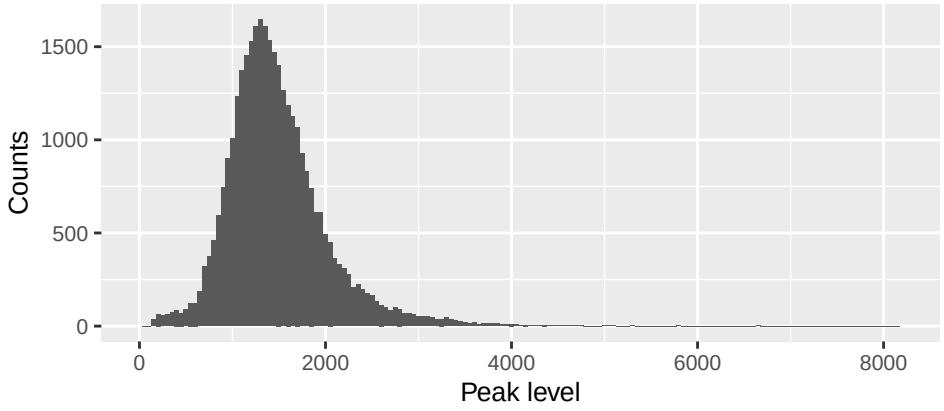


Figure 5.23: Histogram of pulse amplitudes at one of the MITO channels in coincidence.

Once it has been established that the response of each channel looks correct from a pulse amplitude distribution standpoint, it is necessary to verify that the response is the same for all channels, and if it is not, apply the appropriate corrections. Given the symmetry of the detector, the amplitude distribution of the pulses on all channels should be the same.

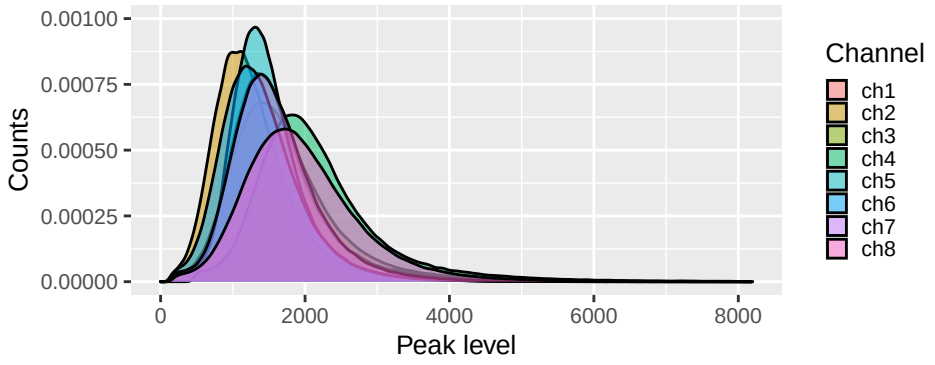


Figure 5.24: Density function of the pulse amplitudes at all of the MITO channels in coincidence.

As figure 5.24 shows that is not the case, as pulse distribution looks correct

for all channels but the pulse amplitudes are slightly different in each case. The difference must be attributed to biasing adjustments and differences in the gain of each PMT, which as mentioned before is not unexpected and can be easily corrected by applying a constant gain to all pulse amplitudes from given channel. Figure 5.25 shows the pulse amplitude distribution once an adequate gain has been applied to each of each channel to achieve this. In this case, the mean pulse amplitude of each channel has been used.

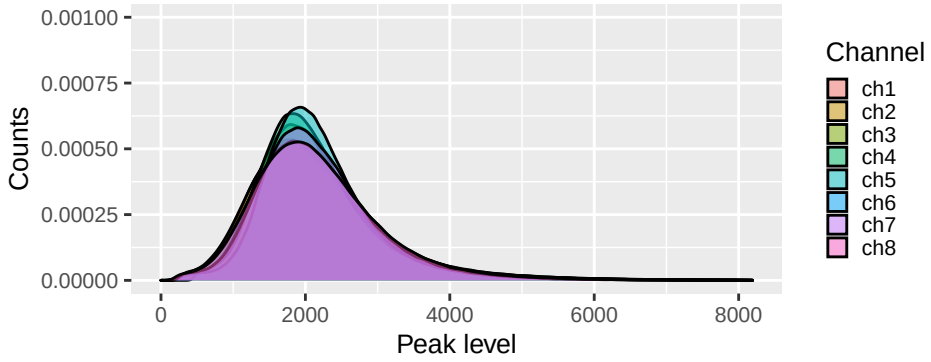


Figure 5.25: Density function of the pulse amplitudes at all of the MITO channels in coincidence, once corrected.

Once it's been checked that the detector is providing the desired measurements, particle flux and incident particle trajectories can be studied, as the following sections will outline.

5.3.2 Particle flux

One of the main goals of a muon telescope is the measurement of particle flux, which has several applications as described in section 5.1. As already explained in section 5.2.2, the acquisition chain is capable of counting and registering several types of counts: events detected by each PMT, in coincidence at the top and bottom scintillators, or in full 8-channel coincidence. This raw count data must then be pressure corrected, since these variables affect measurements. The method for this corrections is thoroughly explained in [40].

Particle count for seven months, between April 1st and October 31st 2020 while the base was already closed and ORCA was operating autonomously is represented in figure 5.26, together with count data from NEMO gathered by its acquisition system. The gaps in the data are caused by power outages at the base, since energy is generated by a renewable energy system based on solar panels, wind

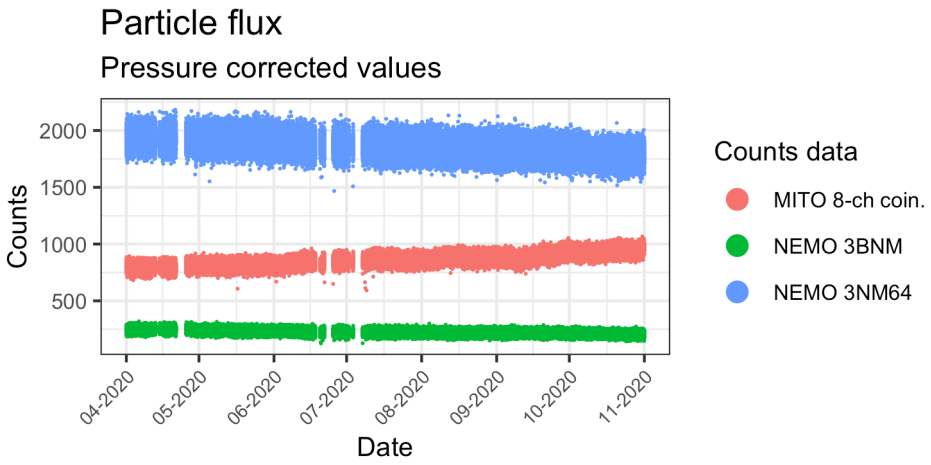


Figure 5.26: Flux data from MITO gathered by the ARACNE prototype, pressure corrected, together with data from the NEMO tubes.

mills and batteries that can run out if there's no wind or solar exposure. The acquisition systems of ORCA, however, resumed operation successfully without supervision once power was back on.

5.3.3 Particle trajectories

The second main goal of the MITO telescope is to be able to measure anisotropies, that is, variations in the distribution of the incoming directions of particles. As explained in section 5.1.3, to accomplish this it must calculate the impact point of particles in two scintillators by using the pulse amplitudes generated by an arrangement of four PMTs around each; then, a trajectory can be obtained. The following sections describe a simple algorithm both for impact point and trajectory calculation and show the results obtained when such algorithms are applied to the experimental data.

Impact point calculation

Even though the prototype MITO telescope installed at the Juan Carlos I Antarctic Base has already demonstrated promising results in its ability to determine impact points and particle trajectories, it is still in a perfecting stage and thus a definitive algorithm to reconstruct this information based on PHA has not been fully developed: simulations can still be perfected and some enhancements to

the construction are still being studied. However, several approaches to the reconstruction algorithms have been developed and confirm the feasibility of the concept, such as the one explained in [40]. Another approach, based on simple geometrical calculations will be detailed here.

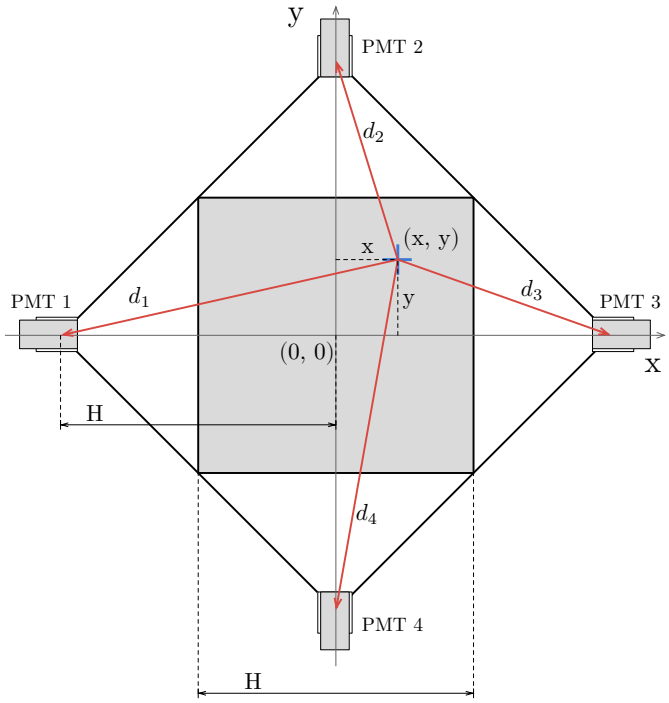


Figure 5.27: Diagram of the geometry of MITO for POI calculation.

The POI calculation algorithm, which is based in the geometry of the MITO devices represented in figure 5.27, makes a couple of assumptions and approximations. First, it assumes spherical propagation, whereas it is likely that each PMT gathers more light that it would in open air since the interior of the MITO housing is painted white. Also, the fact that the scintillator has a thickness along the Z-axis that is not zero is also not considered. Consequently, the light generated at a point in the scintillator will propagate in all directions, and the decrease in intensity will be inversely proportional to the distance squared. Since the PMT generates a pulse whose amplitude is proportional to the light it receives, it can be concluded that

$$p_n = \frac{K \cdot E}{{d_n}^2} \tag{5.1}$$

Where p_n is the pulse amplitude generated by PMT number n, d_n is the distance from the POI to the PMT, E is the energy deposited by the particle and K a proportionality factor that expresses that the intensity of light generated at the POI is proportional to E. Of course, linearity between deposited energy and light generated, as well as between light gathered and pulse amplitude is also assumed.

Also, an approximation is made regarding the position of the PMTs with respect to the center of the scintillator, which is used as the origin of the coordinate system. Given the geometry of figure 5.27, it's easy to see that the distance between the corner where each PMT is installed and the origin of the coordinate system equals the lateral size of the scintillator H (in the prototype, $H = 100\,\text{cm}$). However, even though it has been considered that the PMT is exactly in the corner, it has a light gathering window of a certain size, so the distance from this window and the photocathode to the origin is not exactly H.

In this conditions, and if the POI coordinates are (x, y), four simple equations can be derived:

$$(x + H)^2 + y^2 = {d_1}^2 \tag{5.2}$$

$$x^2 + (y - H)^2 = {d_2}^2 \tag{5.3}$$

$$(x - H)^2 + y^2 = {d_3}^2 \tag{5.4}$$

$$x^2 + (y + H)^2 = {d_4}^2 \tag{5.5}$$

where d_n can be substituted in virtue of equation 5.1 by $K \cdot E/p_n$. Now, with some manipulation of the resulting equations, x and y can be obtained from the pulse heights. We start by dividing equation 5.2 by 5.4:

$$\frac{(x + H)^2 + y^2}{(x - H)^2 + y^2} = \frac{p_3}{p_1} \tag{5.6}$$

It can be observed that this expression is independent of K and E. This leads to

$$x^2 + y^2 + H^2 = -2H\frac{p_1 + p_3}{p_1 - p_3}x \tag{5.7}$$

Similarly, from equations 5.5 and 5.3 we arrive at:

$$x^2 + y^2 + H^2 = -2H\frac{p_4 + p_2}{p_4 - p_2}x \tag{5.8}$$

Now, we define two factors r_x and r_y that depend on the pulse amplitudes of the PMTs of the x and y axis respectively, as follows:

$$r_x = \frac{p_1 - p_3}{p_1 + p_3} \qquad r_y = \frac{p_4 - p_2}{p_4 + p_2} \tag{5.9}$$

Subtracting 5.8 from 5.7 and making the substitutions of equations 5.9 we obtain

$$y = \frac{r_y}{r_x}x \tag{5.10}$$

Now we can use this equation in 5.7, and that leaves an expression that only depends on x. Manipulating the expression and grouping the polynomial terms, this results in

$$(r_y^2 + r_x^2)x^2 + 2Hr_x x + H^2 r_x^2 = 0 \tag{5.11}$$

Solving for x,

$$x = \frac{-Hr_x\left(1 \pm \sqrt{1 - (r_x^2 + r_y^2)}\right)}{r_x^2 + r_y^2} \tag{5.12}$$

where the negative square root must be used, since it's the solution that makes $x = 0$ when $r_x = 0$ and $r_y = 0$, which according to equations 5.9 only happens when pulses from complementary PMTs are of equal amplitude, and thus the x must be 0. Finally, using equation 5.10

$$y = \frac{-Hr_y\left(1 - \sqrt{1 - (r_x^2 + r_y^2)}\right)}{r_x^2 + r_y^2} \tag{5.13}$$

In conclusion, POI coordinates could be easily calculated by computing the r_x and r_y factors for each event and then using equations 5.12 and 5.13, where H in the case of MITO is 100 cm. This has been performed using a program written in the R language and data in the format described in section 5.2.3 gathered by

the ARACNE prototype in MITO, at the Juan Carlos I Antarctic Base, over five hours on February 14th, 2020 to generate figures 5.28 and 5.29.

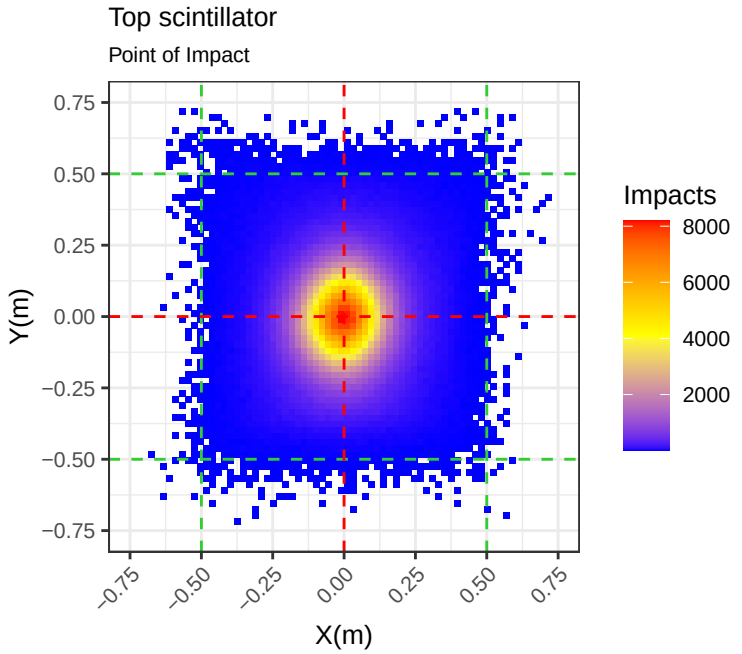


Figure 5.28: POI histogram for individual Top coincidence in MITO. Impacts that have produced a coincidence in the four Top PMTs have been considered, regardless of coincidence in the other level.

Figures 5.28 and 5.29 show the 2D histograms of the point of impact calculated over the 2.4 million events contained in the aforementioned data. Of these, about 1.3 million generated a four channel coincidence on the top level of MITO and are represented in figure 5.28, and about 900.000 generated a coincidence in the bottom level and are represented in figure 5.29. No corrections have been made over the presented POI calculation method, and yet the impact point distribution presents in both cases a square shape of approximately the dimensions of the scintillator, which are outlined by the green dashed lines. These are very promising results, and the impact points falling outside the scintillator limits can be attributed to the simplicity of the algorithm and the assumptions made about the light propagation and the PMT positions.

On the other hand, figure 5.30 shows the histograms for events that generated a coincidence in both the top and bottom levels simultaneously, and thus can be used also for trajectory calculation. In this case, ten hours of data was taken from a different day, February 27th 2020, which contained 500.000 events with

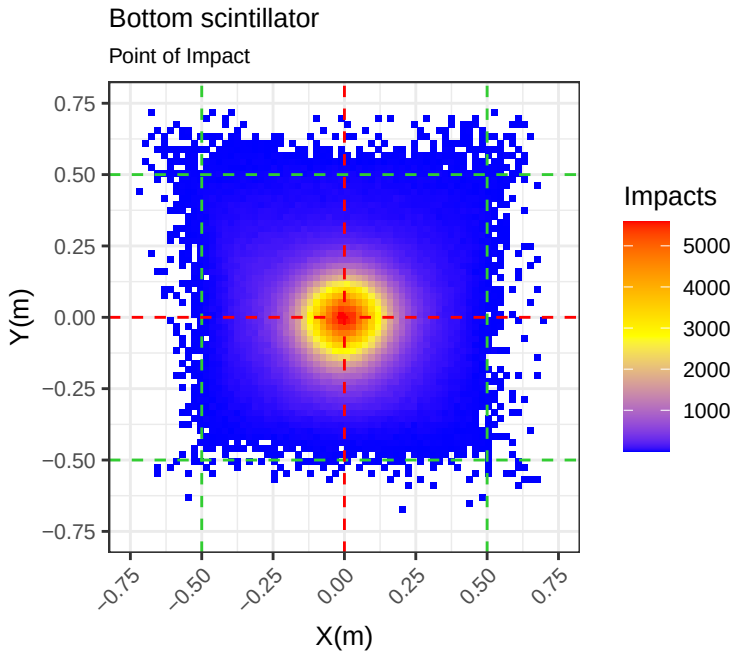


Figure 5.29: POI histogram for individual Bottom coincidence in MITO. Impacts that have produced a coincidence in the four Bottom PMTs have been considered, regardless of coincidence in the other level.

this type of coincidence. It can be seen that while most impacts occur close to the center of the scintillator, impact points near the edges are filtered out. This is not unexpected, because impacts near the edges on one level are more likely to follow a trajectory that falls out of the scintillator on the other level.

Trajectory calculation

With the POI on both levels, incoming directions can be easily computed. From figure 5.12, if dx and dy are the differences in the POI coordinates of the two impacts and the separation between the MITO planes is D, the spherical coordinates of the trajectory can be calculated as

$$\theta = \arctan \frac{\sqrt{dx^2 + dy^2}}{D} \qquad \varphi = \arctan \frac{dy}{dx} \tag{5.14}$$

After performing these calculations for the same dataset used in figure 5.30,

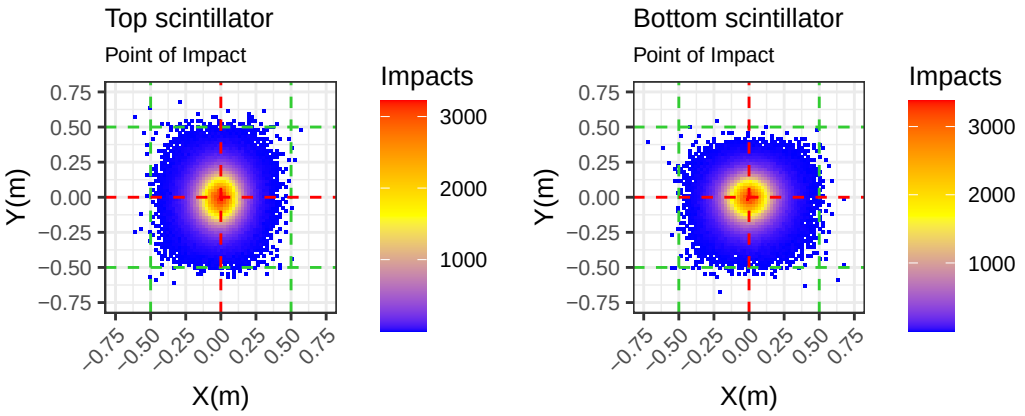
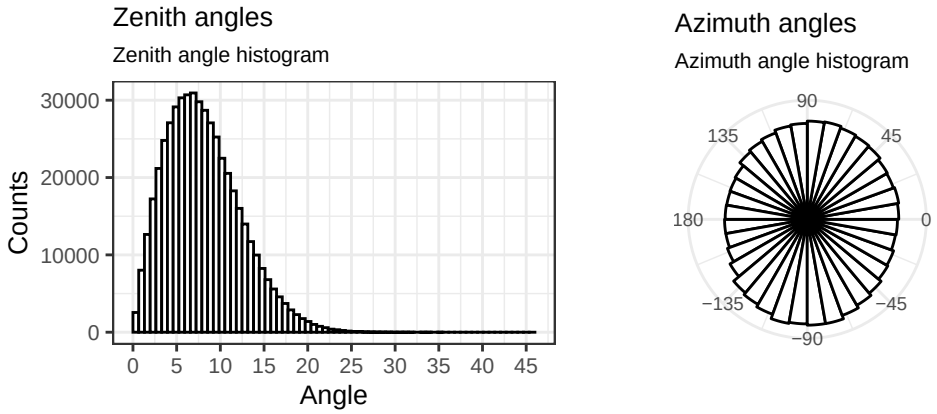


Figure 5.30: POI histogram for simultaneous coincidences in both levels of MITO.

the distribution of azimuth and zenith angles for particle trajectories shown in figure 5.31 can be obtained.



(a) **(b)**

Figure 5.31: Trajectory angle distribution for zenith angle (left) and azimuth (right).

It can be observed that the azimuth angles distribution (figure 5.31b) indicates a certain preference for directions along the y axis. Also, the zenith angle distribution indicates that incident particles are more frequently detected with an incidence angle of around 7°. This is because this histogram doesn't take into account that certain angle sections represent a larger portion of the solid angle

covered by the instrument's FOV. Correcting the histogram by setting breaks that represent the same amount of solid angle, a result closer to expectations is obtained (see figure 5.32i).

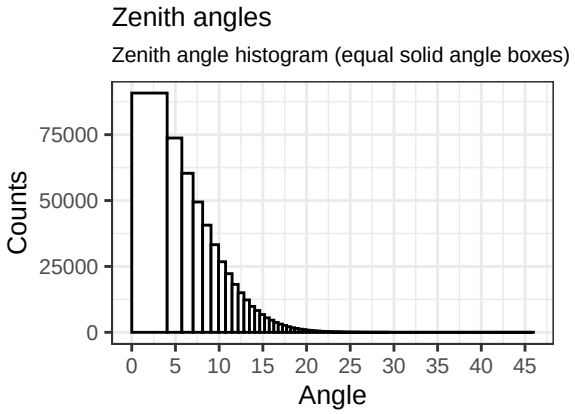
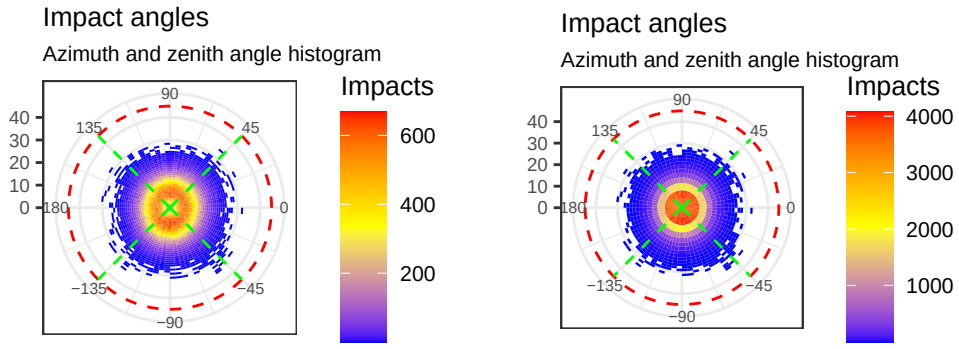


Figure 5.32: Zenith angle distribution for equal solid angles of the FOV.

Both distributions can be better visualized when combined into a 2D polar histogram, as shown in figure 5.33



(a) **(b)**

Figure 5.33: Polar histogram of trajectory angles, linear distribution (left) and constant solid angle distribution (right).

In conclusion, beyond the exactitude that can be obtained from the current MITO implementation, it seems clear that the preliminary results are promising and seem to confirm the validity of the concept; and on the other hand, that the implementation of a large part of the nuclear instrumentation chain using a

prototype based in the ARACNE concept has been successful, validating also the proposal.

5.4 New techniques: Self-Equalizing Maps for POI estimation

The usage of the ARACNE prototype in MITO described in section 5.2 has provided large amounts of data for direct application to the MITO operating principle as described in section 5.1.3, yielding promising results. But it's also been used to envisage other innovative methods to tackle on the most problematic aspects of the MITO concept, as mentioned in section 5.3.1: the difficulty to determine the POI position using analytic methods, and the differences in response of the PMTs.

As a result, a neural network approach based on a novel concept called Self-Equalizing Map (SEM) has been developed and published, using the data gathered by the ARACNE prototype. The details can be found in the aforementioned publication [51], but this section briefly outlines its working principle and the obtained results.

Motivations and principle of operation

Developing a reliable reconstruction algorithm for POI on each plane based on the measured pulse heights is difficult, but the fact that the measurements it is based upon depends on multiple factors, such as response linearity of the PMTs and the associated electronics or environmental factors such as temperature, make it even more difficult. And careful calibration can be performed or corrections made, but conditions can change overtime for different reasons, such as component aging or changes in the environment conditions.

The idea behind this method for determination of the POI is to avoid the need of manual calibration by using a mapping algorithm that transforms the distribution obtained by some simple reconstruction algorithm (such as the ones described in [40], [51] or section 5.3.3) to the expected distribution of impact points, obtained by simulation. This mapping algorithm, called Self-Equalizing Map, thus accounts for all the non-linearities and external factors and can be easily re-run periodically to generate a new map when conditions change, eliminating the need for manual recalibration.

5.4.1 Development

Three things are necessary to develop the algorithm: a real POI distribution obtained from experimental MITO data, a valid target distribution produced by simulation, and an algorithm to produce a map that translates the real data distribution to the target distribution.

The real distribution used as starting point can be the one produced by any of the algorithms enumerated in the previous section; for example the distribution shown in figure 5.30, or the one described in [51]. The important thing in these distributions is that they are ordered, that is; given two impacts, the differences in their x and y coordinates represent a difference in the pulse height registered by complementary PMTs, even though if it is not an exact representation of the POI.

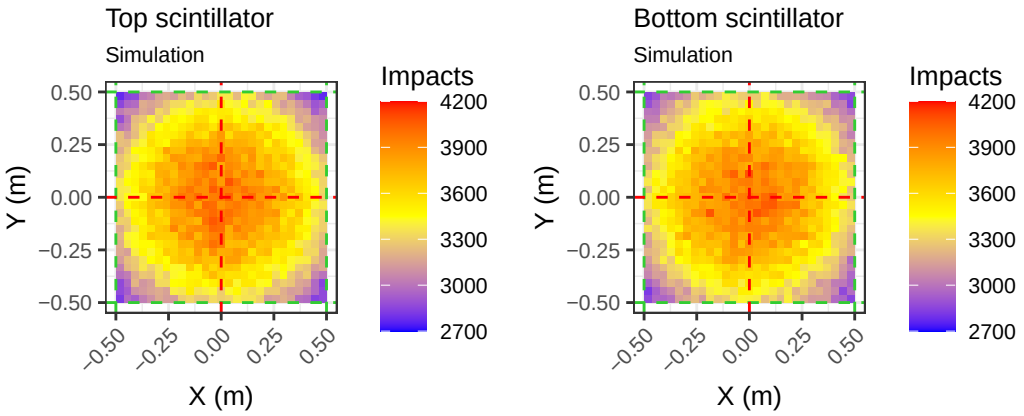


Figure 5.34: Histogram for simulated POIs in MITO.

On the other hand, the target distribution can be generated by simulation, assuming muon impact points and incidence directions of a model such as the one presented in [52], that is, uniform distribution of φ and $cos^2\theta$. Such a simulation performed with a program in R over 20,000,000 impacts taking into account the dimensions of the detector and counting only the particles that cross both scintillators yields the distribution shown in figure 5.34.

Finally, to produce a map between both distributions, a type of artificial neural network called Self-Equalizing Map (SEM) is used. This neural network is trained used unsupervised learning to produce a discrete distribution function that preserves the topological properties of the input, that is, it reorganizes the input data without changing its order. The details on how this map is calculated,

its performance and its limitations can be found in [51].

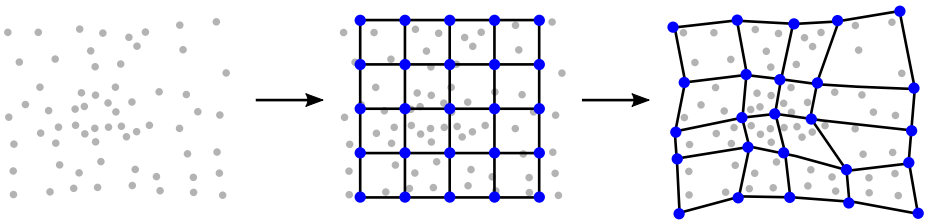
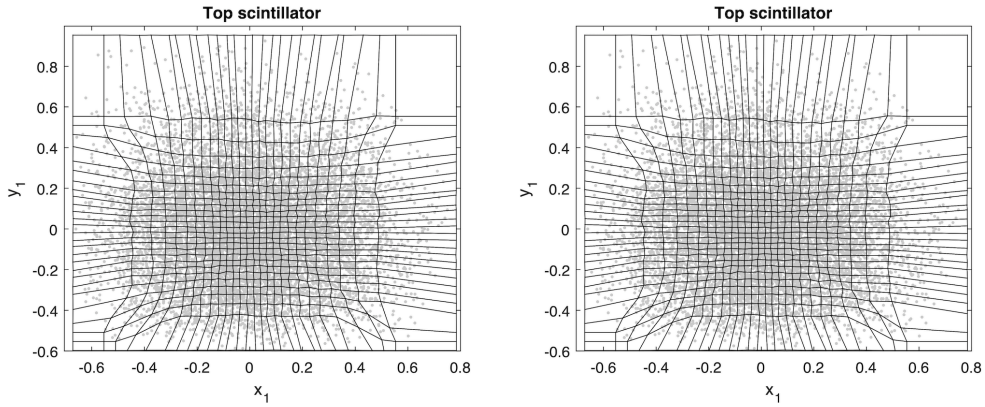


Figure 5.35: Learning process for the SEM algorithm. Figure from [51]

Roughly, the algorithm works by starting with a regular grid of neurons that in turn define a grid of cells covering the area of the scintillator. Then, it gradually moves the neurons with the goal of making the cells contain the same number of impact points that they would in their original positions if the target distribution was used. Figure 5.35 illustrates this concept. In the first frame, a certain distribution of impacts is presented. In the second, a 5x5 neuron grid defining 16 cells is superimposed and it's easy to see each cell has a different number of impacts. Now, if a homogeneous distribution of impacts is theoretically expected, each cell should contain four impacts. In the last frame, the neurons have adapted the grid so that each cell contains 4 impacts.



(a) **(b)**

Figure 5.36: Grids created by the neurons after the training process. Figure from [51]

After an iterative process that stops when it is as close as possible to the desired result, the algorithm produces a grid such as those shown in figure 5.36. These maps can then be used to estimate the correct approximate POI as follows: first, an impact point estimation using the simple method based on the pulse

amplitudes of the original dataset is used, yielding the initial coordinates. These coordinates are then applied to the calculated grid, where they fall in a specific cell. Finally, the POI is estimated using the center of the original position of the cell at the beginning of the training process. It is important to note that this results in a quantization of the POI at each plane, since the resulting coordinates will always be the center of a cell, and not arbitrary. This will, in turn, cause a quantization of the possible angles of incidence as will be shown in the results.

5.4.2 Results

The described method has been used to create a SEM that maps data obtained on MITO from January 17th to 21st of 2019 at the Juan Carlos I Antarctic Base. A random subset of data from these days has been used to train a network of 28x28 neurons (27x27 cells), and then it's been applied to the full dataset as reconstruction algorithm to obtain POI histograms for every day. The results are shown in figure 5.37.

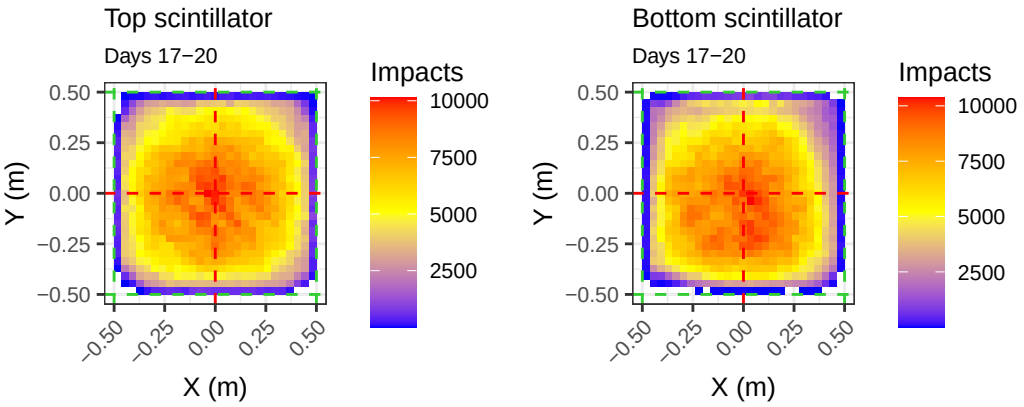
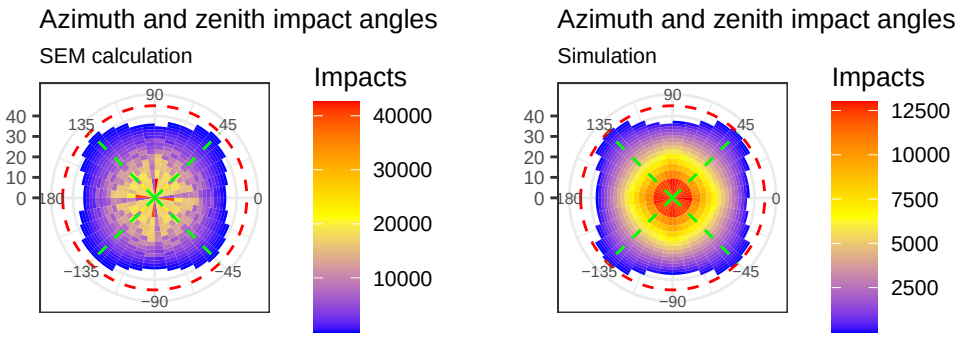


Figure 5.37: POI histograms resulting of applying the SEM to real data

It can be seen that the result is a distribution similar to the simulation results of figure 5.34, except at the edges of the scintillator where the decay in impact frequency is more pronounced because the algorithm struggles to adapt empty areas to the edges of the simulated impact distribution. From these results, particle trajectories can be calculated using the same equations of section 5.3.3.

The results from both the simulation (which is the target distribution) and the real data processed by the SEM algorithm are shown in figure 5.38. It can be observed that there are discontinuities in the histogram of figure 5.38a, but this



(a) **(b)**

Figure 5.38: Polar histogram of trajectory angles generated by the SEM algorithm (left) and simulation (right). A constant solid angle distribution was used in both cases.

is expected since the SEM introduces a quantization in the possible POIs that propagates to the possible angles that can be calculated. This is specially prominent in angles close to the vertical trajectories, where there's a high concentration in certain azimuth orientations while there's very few in others adjacent to them. This problem can be mitigated with grids with smaller cells at the cost of higher computational resources.

Chapter 6

Conclusions

> "You know? It's very strange. I have been in the revenge business so long, now that it's over, I don't know what to do with the rest of my life."
>
> — Iñigo Montoya, *The Princess Bride*

In this work, a design methodology has been developed for building the digital acquisition chain in nuclear instrumentation systems. It focuses in reusability and adaptability to different experimental scenarios without making changes in the hardware, while keeping a small form factor and power consumption. The Adaptable and Reconfigurable Acquisition Concept for Nuclear Electronics platform concept is based in the digitalization of pulses generated by radiation detectors and the use of synthesizable hardware plus software running in a tightly coupled Single-Board Computer (SBC) to perform the functionality of classic instrumentation modules. As a proof of concept, a prototype following this philosophy has been produced, and it has been used as digital acquisition chain for the Muon Impact Tracer and Observer telescope in Antarctica to study muon flux, and anisotropies in their incoming directions.

The following are the most important conclusions that we have obtained in the development of the project:

- A new concept for the design of the digital chain in a nuclear instrumentation system has been proposed. It is composed of a hardware platform with specific features and elements, a modular distribution of IPCore blocks with specific roles and a layered set of system libraries to leverage the hardware and IPCores from the user software perspective.

- The design philosophy has been used to develop a prototype, where it has demonstrated to have helped at several levels. For example, the design of the prototype's IPCore once the hardware-dependant modules were available, could be done mostly independent of the hardware details and focusing in the processing of the digitized signal. Also, the parts of the acquisition software for the MITO telescope that involved the digital acquisition chain were written just by using a set of system libraries created beforehand, requiring little knowledge of the hardware or IPCore details.

- The prototype developed successfully serves its purpose on the MITO telescope in Antarctica, where it provides data for pulse count, pulse amplitude and pulse shape since January 2019. In January 2021, it has completed its first year-round campaign of continuous measurements, surviving the Antarctic winter.

- The prototype developed can be interactively controlled remotely through the Internet while functioning at the Juan Carlos I Antarctic Base, when the base provides satellite connectivity. Also, it provides real-time data to the local servers at the base, so that it can be downloaded periodically when the base is closed.

- The results produced by the ARACNE prototype have been used to characterize the response of the MITO telescope and have been fundamental to corroborate the validity of the MITO concept. Together with results from simulations they will be invaluable to perfect the impact point reconstruction algorithms and in turn, of anisotropy studies.

- The data from the MITO telescope has also been used to develop a novel impact point determination SEM algorithm based in neural networks.

- Finally, as intended by using the ARACNE design philosophy, this prototype can be used for other applications, either with MITO or with other detectors. For example, for MITO there are plans for a calibration campaign that involves using four channels for one level of the telescope and another two for a mobile two-scintillator system that acts as a trigger. This experiment will be performed simply by changing the software application, with no changes to the hardware or IPCore. In other cases, and following the ARACNE philosophy, the hardware platform would remain valid as well as the fundamental IPCores and software libraries, and it would only be necessary to create new processing IPCores and their associated libraries, and then write the new application software.

This work, however, can be continued and enhanced on several fronts. For example:

- An evolution of the design philosophy can be proposed driven by the technological advances of the last decade, such as the appearance of FPGA devices with integrated ARM cores, a major advancement over synthesized CPUs that take large portions of the FPGA fabric and usually have mediocre performance and also over independent SBCs, which require the use of some kind of communications channel. This tighter coupling between the IPCore and main computer of the platform would allow not only a higher level of integration but faster access to acquisition data from the software, and the possibility to perform in real time more complex operations that are difficult to implement in an IPCore

- Producing a standard set of IPCore and system libraries supporting all the classic processing blocks of nuclear instrumentation systems. This work provides some of these, and a basis to develop new ones; but a more complete library of IPCores would considerably push the application of the design philosophy in the future.

- Exploring the possibility of a similar proposal for a distributed digital acquisition chain. In many scenarios, signals generated by detectors cannot be practically carried to a single data acquisition system, and it would be very interesting, from a technical standpoint, to perform the digitalization or even some processing directly at the output of the detector. Thus, a proposal for a system where part of the digital chain is distributed through several physically separate electronics systems, that dealt with the associated problems of this topology, would be very interesting.

- Working on a low-level driver and userspace library specification that allows better performance and control of the SBC I/O portion of the hardware platform, and implement it for a prototype system such as the one described in this work. It's clear from the experience developing the current prototype that the third-party libraries used are not problem-free, their performance is questionable and in general, are not designed with low-level access in mind.

- Finally, a second prototype with much better performance and reconfiguration capabilities can be produced, based on the lessons learned in this work.

Acronyms

BBIO	Adafruit BeagleBone I/O Python API
AC	Alternating Current
ADC	Analog to Digital Converter
ARACNE	Adaptable and Reconfigurable Acquisition Concept for Nuclear Electronics
API	Application Programming Interface
ARM	Acorn RISC Machine
ASIC	Application Specific Integrated Circuit
BAEJCI	Juan Carlos I Spanish Antarctic Base
BBB	Beaglebone Black
BGA	Ball Grid Array
BNC	Bayonet Neill-Concelman
BSP	Board Support Package
CaLMa	Castilla-La Mancha Neutron Monitor
CAMAC	Computer Automated Measurement and Control
CCC	Clock Conditioning Circuit
CME	Coronal Mass Ejection
COTS	Commercial Off-The-Shelf
cPCI	Compact PCI

CPU Central Processing Unit

CSV Comma Separated Values

DDR Double Data Rate

DMA Direct Memory Access

DSP Digital Signal Processor

ECL Emmiter Coupled Logic

ENOB Equivalent Number of Bits

FD Forbrush Decrease

FDA Fully Differential Amplifier

FIR Finite Impulse Response

FIFO First-In First-Out

FOV Field of View

FPGA Field-Programmable Gate Array

FWHM Full Width at Half Maximum

GCR Galactic Cosmic Ray

GLE Ground-Level Enhancement

GMDN Global Muon Detector Network

GPIO General-Purpose Input/Output

GPS Global Positioning System

HV High-Voltage

I2C Inter-Integrated Circuit

IEC International Electrotechnical Comission

IIR Infinite Impulse Response

IPCore Intellectual Property Core

JCI Juan Carlos I Antarctic Base

JTAG Joint Test Action Group

LDO Low Dropout Regulator

LED Light-Emmitting Diode

LLD Lower-Level Discriminator

LSB Least Significant Bit

LV Low-Voltage

LVDS Low Voltage Differential Signaling

LVTTL Low Voltage Transistor to Transistor Logic

LVCMOS Low Voltage Complementary Metal Oxide Semiconductor

MCA Multi Channel Analyzer

MC Magnetic Cloud

MSB Most Significant Bit

MITO Muon Impact Tracer and Observer

NAS Network Attached Storage

NEMO NEutron MOnitor

NIM Nuclear Instrumentation Module

NMDB Neutron Monitor Database

NTP Network Time Protocol

ORCA Cosmic Ray Antarctic Observatory

PC Personal Computer

PRBS Pseudo-Random Binary Sequence

PCB Printed Circuit Board

PCI Peripheral Component Interconnect

PCIe PCI Express

PCI-X Peripheral Component Interconnect eXtended

PHA Pulse Height Analysis

PMT Photomultiplier Tube

POI Point of Impact

PQFP Plastic Quad Flat Pack

PRU Programmable Real-time Unit

PSA Pulse Shape Analysis

PXI PCI eXtensions for Instrumentation

QFP Quad-Flat Package

RAM Random Access Memory

ROM Read Only Memory

RPC Resistive Plate Chamber

RTL Register Transfer Level

SBC Single-Board Computer

SCA Single Channel Analyzer

SEP Solar Energetic Particle

SEM Self-Equalizing Map

SERDES Serialize/Deserialize

SMA SubMiniature version A

SoC System on-Chip

SOM Self-Organizing Map

SPI Serial Peripheral Interface

SRAM Static Random Access Memory

SRG Space Research Group

SSD Solid-State Drive

PLD Programmable Logic Device

PLL Phase-Locked Loop

TRISTAN TRasgo para InveSTigaciones ANtárticas

TTL Transistor to Transistor logic

UAH Universidad de Alcalá

UART Universal Asynchronous Receiver-Transmitter

UDP User Data Protocol

ULD Upper-Level Discriminator

VHDL Very High Speed Integrated Circuit Hardware Description Language

VMEbus Versa Module Euro bus

VXI VME eXtensions for Instrumentation

Bibliography

[1] Glenn G. Knoll. *Radiation Detection and Measurement (4th Edition)*. 2010.

[2] Paul W. Nicholson. *Nuclear electronics*. London ; New York : Wiley, 1974. "A Wiley-Interscience publication.".

[3] Mohammad Nakhostin. *Signal processing for radiation detectors*. 2017.

[4] Alberto Regadío Carretero. *Procesamiento digital de señal aplicado a la detección de partículas energéticas*. PhD thesis, Universidad de Alcalá, 2014.

[5] Sebastián Sánchez Prieto. *Sistema de control de un telescopio detector de iones cósmicos*. PhD thesis, 1998.

[6] Louis Costrell, Frank R. Lenkszus, Stanley J. Rudnick, Eric Davey, John Gould, Seymour Rankowitz, William P. Sims, R. Roy Whitney, Robert W. Dobinson, Henk Verweij, Norman Latner, Vincent C. Negro, Edward J. Barsotti, Thomas E. Droege, Cordon Kerns, Kathleen J. Turner, Robert W. Downing, Frederick A. Kirsten, A. E. Larsh, Stewart C. Loken, Dick A. Mack, Lee J. Wagner, Robert C. Lucena, Dennis W. O'Brien, Allan Gjovig, Frank Naivar, Ronald O. Nelson, D. Hywell White, Carl Akerlof, Donald E. Stilwell, James H. Trainor, Bruno Gobbi, John A. Biggerstaff, Nat W. Hill, Gerald K. Schulze, David B. Gustavson, Dale Horelick, Paul F. Kunz, Leo Paffrath, Helmut V. Walz, W. Kenneth Dawson, John Cresswell, Satish Dhawan, and Charles E. L. Gingell. Standard NIM instrumentation system. 5 1990.

[7] ORTEC. ORTEC website. `https://www.ortec-online.com`, 2020. [Online; accessed 27-October-2020].

[8] IEEE standard modular instrumentation and digital interface system (CAMAC) (Computer Automated Measurement and Control). *ANSI/IEEE Std 583-1982*, pages 1–84, 1982.

[9] Wikipedia contributors. Computer Automated Measurement and Control (CAMAC) — Wikipedia, the free encyclopedia. [Online; accessed 27-October-2020].

[10] WIENER. Wiener website. `https://www.wiener-d.com`, 2020. [Online; accessed 27-October-2020].

[11] Waldemar Nawrocki. *Measurement systems and sensors*. Artech House, 2005.

[12] IEEE standard for a versatile backplane bus: VMEbus. *ANSI/IEEE Std 1014-1987*, pages 1–231, 1987.

[13] CAEN. CAEN Group website. `https://www.caen.it`, 2020. [Online; accessed 28-October-2020].

[14] PXI hardware specification rev. 2.3. pages 0–58, 2018.

[15] Wikipedia contributors. PCI eXtensions for Instrumentation — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Computer_Automated_Measurement_and_Control&oldid=918188744`, 2020. [Online; accessed 29-October-2020].

[16] O.G. Población, J.J. Blanco, R. Gómez-Herrero, C.T. Steigies, J. Medina, I.G. Tejedor, and S. Sánchez. Embedded data acquisition system for neutron monitors. *Journal of Instrumentation*, 2014.

[17] Rocío García Ginez. Development of a custom-made DAQ system for Neutron Monitor and Muon Telescope at Mexico City's Cosmic Ray Observatory. In *33rd International Cosmic Ray Conference*, page 0943, 2013.

[18] S.-I Yasue, K. Munakata, Chihiro Kato, Takao Kuwabara, S. Akahane, M. Koyama, Z. Fujii, Paul Evenson, and J.W. Bieber. Design of a recording system for a muon telescope using FPGA and VHDL. 6:3461, 06 2003.

[19] C R Braga, A Campos, N J Schuch, and A Dal Lago. A proposal of a counting and recording system for cosmic ray muon detectors. *Journal of Physics: Conference Series*, 409:012137, feb 2013.

[20] T. Steele, L. Mo, L. Bignell, M. Smith, and D. Alexiev. FASEA: A FPGA acquisition system and software event analysis for liquid scintillation counting. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 609(2):217 – 220, 2009.

[21] C. Bobin, J. Bouchard, and B. Censier. First results in the development of an on-line digital counting platform dedicated to primary measurements. *Applied Radiation and Isotopes*, 68(7):1519 – 1522, 2010. Proceedings of the 17th International Conference on Radionuclide Metrology and its Applications (ICRM 2009).

[22] J. Bouchard and P. Cassette. MAC3: an electronic module for the processing of pulses delivered by a three photomultiplier liquid scintillation counting system. *Applied Radiation and Isotopes*, 52(3):669 – 672, 2000.

[23] C. Bobin, J. Bouchard, S. Pierre, and C. Thiam. Overview of a FPGA-based nuclear instrumentation dedicated to primary activity measurements. *Applied Radiation and Isotopes*, 70(9):2012 – 2017, 2012. Proceedings of the 18th International Conference on Radionuclide Metrology and its Applications.

[24] M. Faisal, R. T. Schiffer, M. J. Haling, M. Flaska, S. A. Pozzi, and D. D. Wentzloff. A data processing system for real-time pulse processing and timing enhancement for nuclear particle detection systems. *IEEE Transactions on Nuclear Science*, 60(2):619–623, 2013.

[25] Randolph T. Schiffer, Marek Flaska, Sara A. Pozzi, Sean Carney, and David D. Wentzloff. A scalable FPGA-based digitizing platform for radiation data acquisition. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 652(1):491 – 493, 2011. Symposium on Radiation Measurements and Applications (SORMA) XII 2010.

[26] R.C. Pereira, J. Sousa, A.M. Fernandes, F. Patrício, B. Carvalho, A. Neto, C.A.F. Varandas, G. Gorini, M. Tardocchi, D. Gin, and A. Shevelev. ATCA data acquisition system for gamma-ray spectrometry. *Fusion Engineering and Design*, 83(2):341 – 345, 2008. Proceedings of the 6th IAEA Technical Meeting on Control, Data Acquisition, and Remote Participation for Fusion Research.

[27] E Fysikopoulos, G Loudos, M Georgiou, S David, and G Matsopoulos. A Spartan 6 FPGA-based data acquisition system for dedicated imagers in nuclear medicine. *Measurement Science and Technology*, 23(12):125403, nov 2012.

[28] A. M. Fernandes, R. C. Pereira, J. Sousa, A. J. N. Batista, A. Combo, B. B. Carvalho, C. M. B. A. Correia, and C. A. F. Varandas. HDL based FPGA interface library for data acquisition and multipurpose real time algorithm processing. In *2010 17th IEEE-NPSS Real Time Conference*, pages 1–5, 2010.

[29] James Karki. Fully-Differential Amplifiers. Technical report, Texas Instruments, 2016.

[30] Texas Instruments. *Eight-Channel, 14-Bit, 80-MSPS Analog-to-Digital Converter (ADC)*, 2018. Rev. E.

[31] On Semiconductors. *Understanding Data Eye Diagram Methodology for Analyzing High Speed Digital Signals*.

[32] Microsemi Corporation. *ProASIC3 FPGA Fabric User's Guide*, 2012. Rev. 4.

[33] Microsemi Corporation. *ProASIC3E Flash Family FPGAs*, 2012. Rev. 15.

[34] Raspberry Pi Foundation. Raspberry Pi Foundation website. `https://www.raspberrypi.org`, 2020. [Online; accessed 16-December-2020].

[35] BeagleBoard.org Foundation. BeagleBoard.org Foundation website. `https://www.beaglebone.org`, 2020. [Online; accessed 16-December-2020].

[36] BeagleBoard.org Foundation. BeagleBone Black system reference manual. `https://github.com/beagleboard/beaglebone-black/wiki/System-Reference-Manual`, 2020. [Online; accessed 17-December-2020].

[37] Jiri Gaisler. A structured VHDL design method. *Fault-tolerant microprocessors for space applications*, pages 41–50, 2011.

[38] Cooper, Justin. Adafruit BeagleBone I/O Python API project. `https://pypi.org/project/Adafruit-BBIO/`, 2020. [Online; accessed 05-January-2021].

[39] Adafruit Industries and contributors. Adafruit Beaglebone I/O Python API reference. `https://adafruit-beaglebone-io-python.readthedocs.io/en/latest/index.html`, 2020. [Online; accessed 05-January-2021].

[40] Ayuso, Sindulfo, Blanco, Juan José, García Tejedor, Juan Ignacio, Gómez Herrero, Raúl, Vrublevskyy, Iván, García Población, Óscar, and Medina, José. MITO: a new directional muon telescope. *J. Space Weather Space Clim.*, 11:13, 2021.

[41] J. J. Blanco, E. Catalán, M. A. Hidalgo, J. Medina, O. García, and J. Rodríguez-Pacheco. Observable Effects of Interplanetary Coronal Mass Ejections on Ground Level Neutron Monitor Count Rates. *Solar Physics*, 2013.

[42] NMDB Consortium. NMDB: Real-Time Database for high-resolution Neutron Monitor measurements. `http://www01.nmdb.eu`, 2021. [Online; accessed 14-January-2021].

[43] H. Mavromichalaki, A. Papaioannou, C. Plainaki, C. Sarlanis, G. Souvatzoglou, M. Gerontidou, M. Papailiou, E. Eroshenko, A. Belov, V. Yanke, E. O. Flückiger, R. Bütikofer, M. Parisi, M. Storini, K. L. Klein, N. Fuller, C. T. Steigies, O. M. Rother, B. Heber, R. F. Wimmer-Schweingruber, K. Kudela, I. Strharsky, R. Langer, I. Usoskin, A. Ibragimov, A. Chilingaryan, G. Hovsepyan, A. Reymers, A. Yeghikyan, O. Kryakunova, E. Dryn, N. Nikolayevskiy, L. Dorman, and L. Pustil'Nik. Applications and usage of the real-time Neutron Monitor Database. *Advances in Space Research*, 2011.

[44] D. F. Smart and M. A. Shea. World grid of calculated cosmic ray vertical cutoff rigidities for epoch 2000.0. In *Proceedings of the 30th International Cosmic Ray Conference, ICRC 2007*, 2007.

[45] R. R.S. Mendonça, C. Wang, C. R. Braga, E. Echer, A. Dal Lago, J. E.R. Costa, K. Munakata, H. Li, Z. Liu, J. P. Raulin, T. Kuwabara, M. Kozai, C. Kato, M. Rockenbach, N. J. Schuch, H. K. Al Jassar, M. M. Sharma, M. Tokumaru, M. L. Duldig, J. E. Humble, P. Evenson, and I. Sabbah. Analysis of Cosmic Rays' Atmospheric Effects and Their Relationships to Cutoff Rigidity and Zenith Angle Using Global Muon Detector Network Data. *Journal of Geophysical Research: Space Physics*, 2019.

[46] M Rockenbach, A Dal Lago, NJ Schuch, K Munakata, T Kuwabara, AG Oliveira, E Echer, CR Braga, RRS Mendonça, C Kato, et al. Global muon detector network used for space weather applications. *Space Science Reviews*, 182(1-4):1–18, 2014.

[47] José Medina, Juan J. Blanco, Oscar García, Raúl Gómez-Herrero, Edwin J. Catalán, Ignacio García, Miguel A. Hidalgo, Daniel Meziat, Manuel Prieto, Javier Rodríguez-Pacheco, and Sebastián Sánchez. Castilla-La Mancha neutron monitor. *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 2013.

[48] Gonzalo Díaz-Romeral Marcos. Módulo de Alojamiento para Observatorio de Rayos Cósmicos Antártico (MAzORCA), 2018.

[49] Óscar García Población. CaLMa Station and a new data acquisition system for Neutron Monitors, 2020.

[50] Hamamatsu Photonics K.K. *C7319 Amplifier unit*, 2011.

[51] Alberto Regadío, J. Ignacio García Tejedor, Sindulfo Ayuso, Óscar García Población, Juan José Blanco, Sebastián Sánchez-Prieto, and Óscar Rodríguez Polo. Trajectory determination of muons using scintillators and a novel self-organizative map. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 973:164166, 2020.

[52] J.D. Sullivan. Geometric factor and directional response of single and multi-element particle telescopes. *Nuclear Instruments and Methods*, 95(1):5 – 11, 1971.