Development of a Modular, Multi-Channel, Cost-Efficient Scintillation Detector for Use in Automated Synthesis of Positron Emission Tomography Radiopharmaceuticals

Final Project Report

Authors: Chris Hatoum, Habib Jarweh, Philippe Olivier Fils, Tofic Esses
ECSE458 - Capstone Design Project
Group 16

McGill University

Advisors: Dr. Shirin A. Enger, Liam Carroll, Víctor Daniel Díaz Martínez

December, 2024

Abstract

This project focuses on the development of a modular, multi-channel, cost-efficient scintillation detector for validating radiopharmaceutical production in positron emission tomography (PET). While PET imaging is essential in diagnosing and monitoring various diseases, current detector systems face significant limitations in accessibility, efficiency, and adaptability. The aim of this project is to address these challenges by designing a scalable and affordable detector system utilizing silicon photomultipliers (SiPMs) and ESP32 microcontrollers for wireless communication, with a target cost of under 200 CAD per detector.

The project's objectives include creating a system that supports at least four independent detection channels, ensures real-time data acquisition with specific detection criteria (tolerance of ±10 mCi and detection range of 50-4000 mCi), and offers ease of scalability and maintenance. The design incorporates innovative features such as a dual spiral fiber holder and enhanced ADC sampling for improved detection accuracy. During this semester, we made significant progress on both hardware and firmware development, building upon previous work from Dr. Enger's research group.

The firmware, responsible for signal processing, peak detection, and real-time communication, was developed using the ESP-NOW protocol, with particular attention to enabling off-site detection to reduce radiation exposure.

Contents

List of Abbreviations

- **ADC** Analog to Digital Converter: Electronic module often found in microcontroller devices. Converts an analog signal into a digital signal for computational interpretation.
- **MNI** Montreal Neurological Institute
- **Op-amp** Operational Amplifier: Electronic component used for amplifying electronic signals.
- **PET** Positron emission tomography
- **PMT** Photomultiplier tube
- **RXTX** Receiver-Transmitter: This is a term often given to systems that can receive and transmit data over a medium, either a physical (using electric signals) or non-physical (using Electromagnetic Radiation).
- **SiPM** Silicon Photomultiplier: This is an electronic component that can detect photon incidents on the surface of the SiPM. By means of the avalanche effect, a single photon can excite multiple electrons, producing an electrical peak signal that can be processed using other electronic components.

1 Motivation

The primary goal of this project is to develop a modular, multi-channel, cost-efficient scintillation detector specifically for use in automated synthesis of positron emission tomography (PET). The motivation for this endeavor stems from the significant challenges posed by the high costs and complex designs associated with current PET systems. These challenges often restrict the availability of advanced medical imaging technologies, particularly in under-resourced or remote settings.

By creating a scintillation detector that is both affordable and adaptable, we aim to transform the accessibility of PET technology across a diverse range of medical environments. This approach will enable more healthcare facilities, especially those with limited budgets, to employ PET imaging. As PET is crucial for accurate diagnosis and treatment planning in serious conditions such as cancer and various neurological disorders, enhancing its accessibility could have a substantial impact on public health. Our project intends to leverage recent advances in material science and modular design principles to construct a detector that not only costs less but also offers the flexibility to be customized for different diagnostic needs. This flexibility is key for adapting the technology to different types of medical facilities and patient care protocols. Furthermore, reducing the complexity and cost of these detectors could lead to broader adoption in clinical practice, facilitating earlier detection of diseases and more precise monitoring of treatment responses. In turn, this could lead to improved patient outcomes and more efficient use of medical resources.

In summary, our project seeks to make a pivotal technology in medical imaging more accessible and practical for a wider range of healthcare providers. By doing so, we aim to enhance the quality and reach of medical diagnostics and treatment, which could ultimately contribute to better health outcomes globally.

2 Background

Scintillation detection is essential in nuclear medicine for detecting and measuring ionizing radiation, particularly in the validation phase between production and consumption of radiopharmaceuticals. The process involves scintillators, materials that emit light upon radiation exposure, with the emitted light converted into electrical signals by photodetectors such as silicon photomultipliers (SiPMs). These signals are crucial for ensuring accurate, sensitive, and non-invasive radiation detection in radiopharmaceutical production.

Current research emphasizes the benefits of modular and scalable detector designs, which allow for cost-effective customization and scalability across various applications. Building on foundational work from Dr. Enger's research group, this project aims to overcome existing limitations in detector systems. Our approach incorporates specific technical innovations, including a dual spiral fiber holder design that calculates the mean of two inputs ($s = \sqrt{s_1 s_2}$) and faster ADC sampling, to enhance detection accuracy while maintaining a practical detection range of 50-4000 mCi with a tolerance of ±10 mCi. These specifications are particularly important for enabling off-site detection capabilities, which help reduce radiation exposure during the validation process.

This project builds on the foundational work of researchers like Hailey Ahn [1], who demonstrated the feasibility of using cost-effective materials in effective scintillation detectors. Building on these innovations, this project aims to develop a modular, multichannel scintillation detector system that incorporates advanced materials and photodetection technologies to enhance performance, reduce costs, and adapt to the growing needs of nuclear medicine and radiopharmaceutical production.

3 Requirements and Problem Description

The goal of this project is to design and develop a modular, multi-channel, cost-efficient scintillation detector specifically for use in the automated synthesis of positron emission tomography (PET) radiopharmaceuticals. To achieve this, the system must meet several key requirements and overcome specific challenges related to both hardware and software.

3.1 Project Requirements

1. **Modular Design:**
    - The system should consist of multiple independent modules, each capable of functioning autonomously or together in a multi-channel configuration. This allows for easy scalability, repair, and replacement without requiring a complete system overhaul.
    - Modules must be stackable and compact to fit into limited workspaces, with a simple interface to connect power and communication between them.
2. **Multi-Channel Capability:**
    - The detector system should support simultaneous data collection from at least four scintillation detector channels. These channels must be able to process incoming signals concurrently and independently, allowing for multiple PET radiopharmaceutical production tasks or signal processing tasks to occur in parallel.
    - Synchronization between channels must be ensured to guarantee accurate measurements and avoid data loss.
3. **Cost-Efficiency:**
    - The system should be designed using affordable and readily available materials, especially for key components such as silicon photomultipliers (SiPMs) and the electronic components used for signal processing.
    - The system must achieve its cost goals without sacrificing performance, reliability, or scalability.
4. **High Sensitivity and Signal Processing:**
    - The detector should be capable of detecting and processing low-amplitude signals from SiPMs, which are often small in magnitude but carry critical information about photon events.
    - The signal processing circuitry must amplify, filter, and perform peak detection with minimal signal distortion, allowing for accurate and reliable measurements.

5. **Wireless Data Transmission:**
    - The system must include a wireless communication feature to transmit data from the detector to a host computer, minimizing physical wiring and increasing the system's flexibility. ESP-NOW was selected for this purpose, enabling reliable wireless communication between multiple ESP32 modules.
6. **Real-Time Data Acquisition and Processing:**
    - The firmware must be capable of reading, processing, and transmitting data in real-time. This includes tasks such as analog-to-digital conversion (ADC), signal integration, and peak detection within a defined time window to avoid data lag or loss.
    - Each module should perform summation of signals, time-stamping of events, and peak detection within specified integration periods.
7. **Software Interface:**
    - A user-friendly software application must be developed to display real-time data from the detector system. The application must provide options for configuring channel parameters, visualizing radiation data, and exporting results for further analysis.
    - The software should support communication with multiple channels, allowing users to control integration periods and start/stop the system remotely.
8. **Environmental Constraints:**
    - The system must operate efficiently under typical laboratory conditions and be robust enough to handle environmental factors such as temperature variation and minor physical disturbances.
    - Power consumption should be minimized, especially for long-term or continuous monitoring tasks, to ensure cost-effectiveness and sustainability.
9. **Safety Considerations:**
    - The system must adhere to electrical safety standards, particularly when handling higher voltages for the SiPM biasing circuits. Additionally, user interaction with radioactive materials must be minimized through wireless data transmission and remote control of the system.

3.2 Problem Description

Current PET radiopharmaceutical production systems face several challenges, primarily due to the high cost and complexity of scintillation detectors used in these systems. Existing detectors, which are often based on photomultiplier tubes (PMTs), are expensive, bulky, and fragile. Moreover, their designs are not modular, making it difficult to scale

systems according to specific production needs. This presents a barrier for smaller or underfunded facilities looking to implement PET technology.

By addressing these challenges, this project seeks to offer a solution in the form of a cost-effective, flexible, and scalable scintillation detector system. The modular design allows for easier maintenance, upgrade, and scalability, while the integration of SiPM technology and wireless communication reduces costs and increases the portability of the system.

In conclusion, the success of this project hinges on meeting these technical, operational, and cost-related requirements, ensuring that the final system can be deployed in real-world PET radiopharmaceutical production environments, offering improved accessibility to advanced medical imaging technologies.

4 Design

4.1 Firmware Solution

4.1.1 Background

The firmware is the code that is written into the memory of various microcontrollers and embedded systems. It is used as the first software layer in any system containing hardware components that need to be controlled. In the case of the detector, the firmware is responsible for receiving analog signals from the processed SiPM signals, processing them, and communicating with other microcontrollers for multi-channel capabilities.

4.1.2 Requirements

The requirements for the firmware are:

- Read two analog signals concurrently
- Integrate each of the subsequent reads for a specified duration (integration period)
- Perform the square-root-product mean on the two integrated values

$$s = \sqrt{s_1 s_2}$$

- Capture the time (timestamp) of the processed data point, s
- Send the captured data to the computer in real-time
- Have multiple channels performing these tasks

Although the requirements for the firmware are short, achieving these can become quite complex depending on the setup.

4.1.3 Design Choices

After carefully considering other microcontrollers, namely the Arduino Nano and the Arduino Uno, the ESP32 has been chosen. Below is a table that summarizes the key features of the three microcontrollers.

The main reasons for choosing the ESP32 were the superior clock speed, memory, wireless capabilities, size, and cost. The capabilities and limitations of the ESP32 heavily influenced the firmware's design choices and overall system functionality.

ESP32 Capabilities
The wireless capabilities, especially ESP-NOW (a wireless point-to-point connection between ESP32 devices) make this microcontroller an excellent choice. Controlling the detector wirelessly will give the user more protection against radiation, since they would be further away. Furthermore, the superior clock speed of up to 240 MHz ensures that there is no significant delay in processing the data (summation/integration of the ADC

Feature	ESP32	Arduino Nano	Arduino Uno
Processor	Dual-core Tensilica Xtensa LX6	ATmega328P (8-bit AVR)	ATmega328P (8-bit AVR)
Clock Speed	160/240 MHz	16 MHz	16 MHz
Flash Memory	4 MB to 16 MB	32 KB	32 KB
SRAM	520 KB	2 KB	2 KB
GPIO Pins	30-36	14 (6 PWM)	14 (6 PWM)
Analog Inputs	16 ADC channels	8	6
Digital I/O Pins	34	22	14
Communication Interfaces	UART, SPI, I2C, CAN, I2S, Ethernet	UART, SPI, I2C	UART, SPI, I2C
Wireless Connectivity	Wi-Fi (802.11 b/g/n), Bluetooth 4.2, ESP-NOW	None	None
Operating Voltage	3.3V	5V	5V
Input Voltage (VIN)	5V to 12V	6-20V	7-12V
Power Consumption	Medium	Low	Low
USB Interface	USB Micro-B	Mini-USB	USB Type-B
Dimensions	25.5x51 mm	18x45 mm	68.6x53.4 mm
Programming Environment	Arduino IDE, ESP-IDF	Arduino IDE	Arduino IDE
Price Range	CAD 13	CAD 10	CAD 20

Table 1: Comparison of Features between ESP32, Arduino Nano, and Arduino Uno

readings) and performing other tasks.

ESP32 I2S (Inter-Integrated Sound) Peripheral
When testing the sampling frequency of the ADC using the analogRead() wrapper function, the maximum sampling frequency obtained was only 56kHz. A low sampling frequency is not desirable because a lot of the SiPM signals can be missed. A workaround is to use the I2S hardware peripheral inside the ESP32. Although designed for sound processing, it can be used to read voltage values from any analog input. The I2S is combined with the DMA and buffers to enable faster sampling, achieving up to 2 MHz sampling.

ESP32 Limitations
A major limitation, but not a dealbreaker, is that the ESP32, despite having dual processing cores, cannot sample from two ADCs simultaneously. To perform the square-root-product mean from two SiPM signals, one channel will require two ESP32s reading signals and processing them concurrently.

4.1.4 Firmware Overview

To ensure synchronization between two ESP32s for one channel, ESP-NOW can be utilized to transmit and receive messages between them. These messages should contain synchronization information and data. One of these ESP32s would have to perform the square-root-product mean and transmit the information to the host computer. This will be called the Leader. The other ESP32 will transmit integrated values to the leader. This one is called the Follower. The Leader will maintain control over the Follower by sending requests. The Follower will respond to these requests when it has finished its task.

4.1.5 Sampling and Summation

A large component of the firmware is reading and processing the data. The I2S is used to sample the data and write each sample into a buffer. This is done at 1215000 SPS (samples-

per-second). The software will then take over to read the buffer and sum all values stored in the buffer. During summation, the hardware is not active, and so there will be a time when the analog signal is not read. This is called the summation duration. During summation, the software will actively check if new peaks were detected. When a new peak is detected, a flag is set, and a GPIO is scheduled for a HIGH voltage to discharge the capacitor (responsible for holding the peak) after a set duration, peakHoldTime. After the peakHoldTime has elapsed, the GPIO will become HIGH for a dischargeDuration of 10 microseconds.

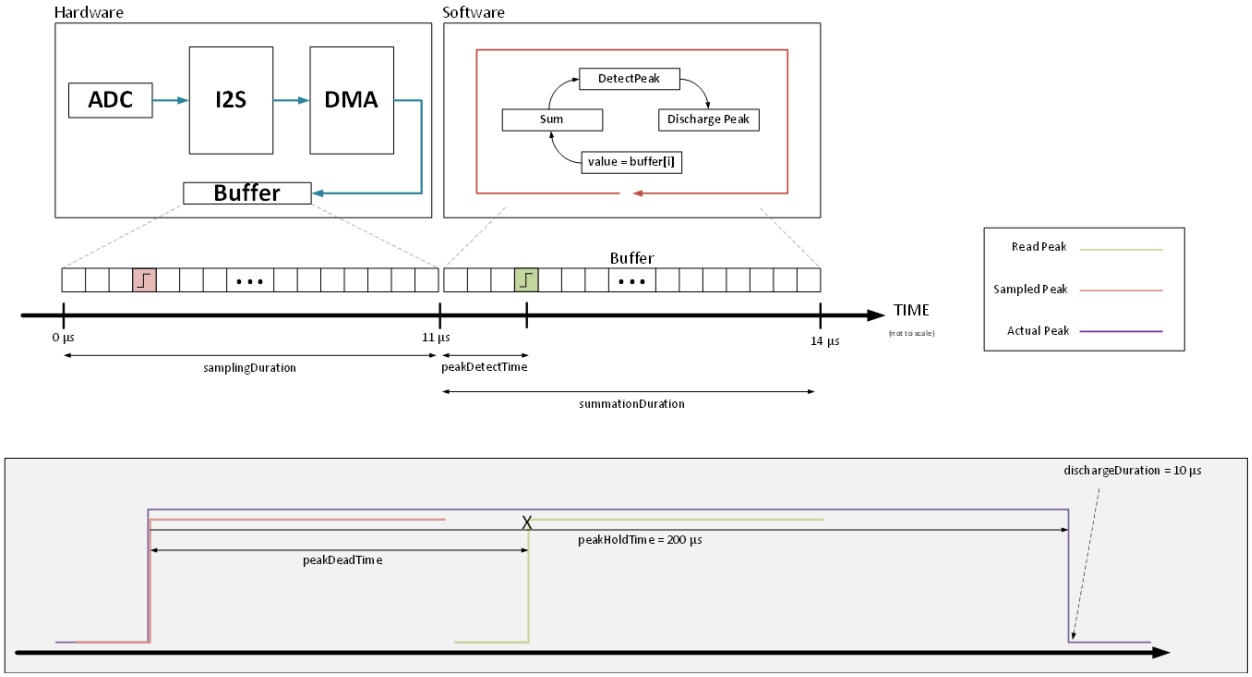

Figure 1: Peak Hold and Discharge, and Summation

Notice that summation only starts after the buffer has been fully filled, called the samplingDuration. Then it takes an extra amount of time to detect a new peak, called the peakDetectTime. The time between the actual peak and the peakDetectTime is called the peakDeadTime. This must be calculated to subtract from the peakHoldTime, to ensure that peaks are consistently held at the desired peakHoldTime. i is the index in the buffer where the peak has been detected.

$$\text{peakDeadTime} = \text{samplingDuration}\left(\frac{i}{16}\right) + \text{peakDetectTime}$$

Finally, to limit the amount of dead time during summation, the buffer must not be too big, nor too small. If the buffer is too big, the software will spend more time summing the values. If it's too small, there would be too many context switches in the software, increasing the overhead and deadtime. After carefully testing all the cases, the ideal buffer size is 16. This resulted in a samplingDuration of 11 microsecond and summationDuration of 3 microseconds. The dead time is therefore about 20%. With these parameters, the effective samples per second after a full integration period of 1 second is 1 MSPS, even

though the I2S is configured at 1215000 SPS. This is because of the dead time during the many summations. This is a sampling loss of 17%, a good amount considering the effective sampling frequency is 1 MSPS. The previous design that used the Arduino Nano achieved a sampling rate of 115 kSPS. This is an 8.7X increase is sampling rate. Thanks to this increase, more individual peaks caused by SiPM spikes be detected in the same integration period window.

4.1.6 System Overview

Please note that everything explained here is high-level. For more details on the actual implementation, have a look at the firmware code.

The Leader (M) and Follower (S) are both ESP32 devices communicating which each other using ESP-NOW. This means there is no need for a physical communication medium to share data. Thereby, reducing the complexity of the hardware. Communication between the Leader and Follower are essential for maintaining synchronization of the concurrent reads. The RXTX receives the data from each channel's Leader, also using ESP-NOW. The RXTX packages the data and transmits the data to the application via a UART (USB) connection. The application transmits commands to the RXTX, and the RXTX transmits the individual commands to each ESP32 Leader.

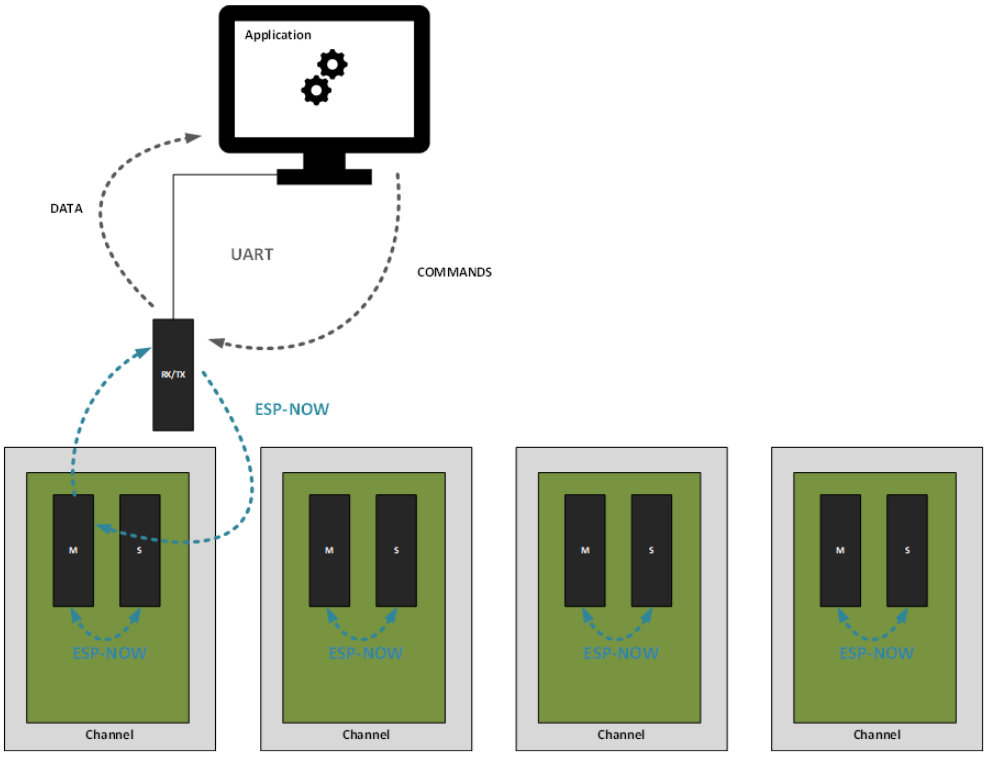

Figure 2: Firmware System

4.1.7 Follower Component

Ensuring each ESP32 behaves appropriately, it is important to model the behavior using state machines. See below the state diagram of the Follower device. This device is mainly responsible for sampling and integrating from its own ADC (called the INTEGRATE state), and then transmitting the sum value to the Leader (called the TRANSMIT state). The WAIT state waits for the Leader to begin a new integration window to capture a new data point.

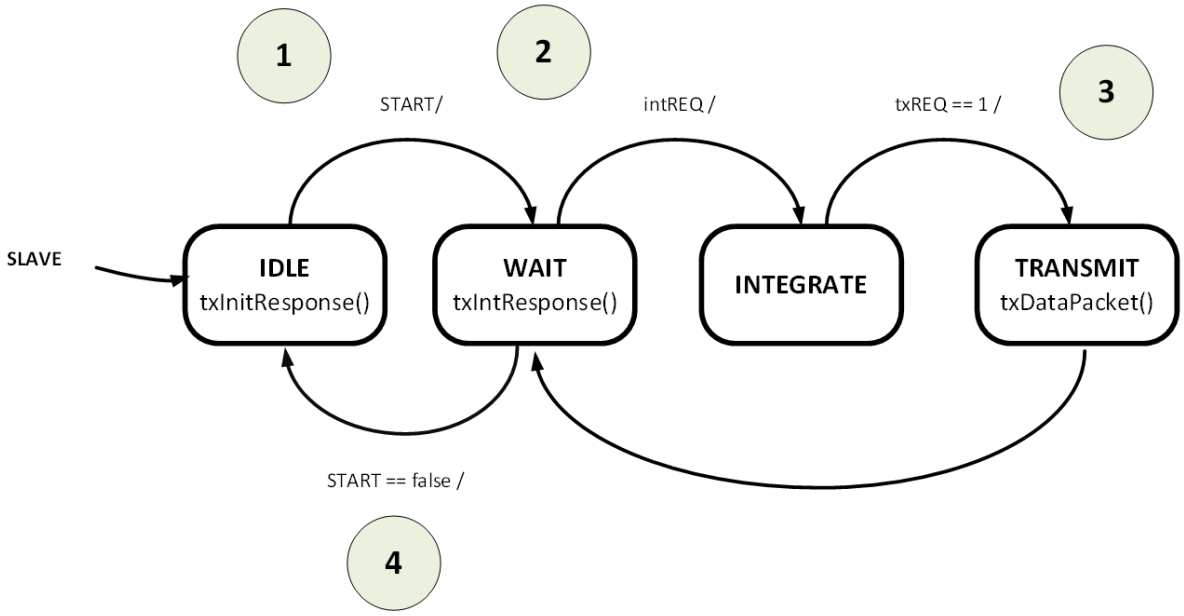

Figure 3: Follower State Machine

4.1.8 Leader Component

See below the state diagram of the Leader device. This device is the leader of the communication. It dictates when the follower must enter the INTEGRATE state and when it must transmit the integrated data (by entering the TRANSMIT state). The leader also has its own INTEGRATE state because it must sample and integrate from its own ADC too. After the INTEGRATe state the Leader automatically transitions to the RECEIVE state and transmits a Data Request message to the Follower, thereby asking it to send its sum. After receiving (NEW == 1), the Leader enters the MEAN state and computes the square-root-product mean of the two summations. After computing the mean, it automatically transitions to the TX_RXTX state. In this state the Leader transmits the square-root-product mean, timestamp, and other information (more on this later). The Leader then enters the WAIT state and transmits an Int(egration) Request to the Follower, asking it if it's ready to start integrating for a new data point. Both the Leader and Follower state diagrams have the IDLE state. This is the initial entry state of Leader/Follower and when it is not running. In the IDLE state the leader transmits an InitRequest to the Follower, essentially asking it if it is ready to start running. If it never received a response from the follower

(ACK), the state transition to the WAIT state will never occur.

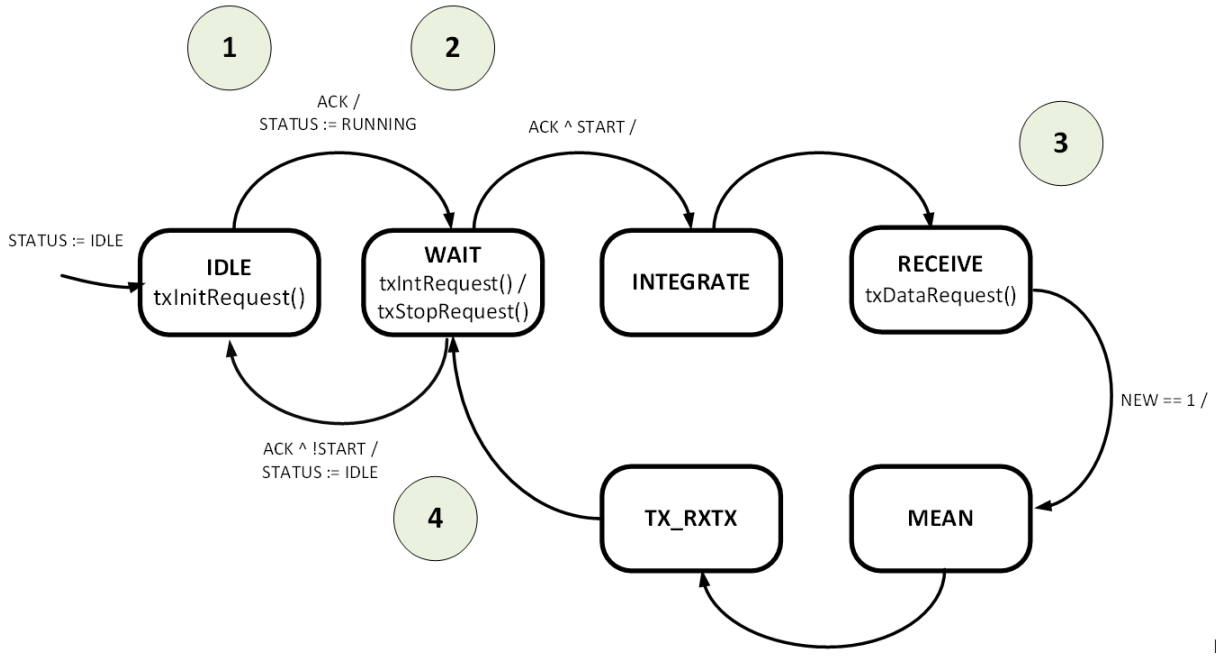

Figure 4: Leader State Machine

4.1.9 Leader-Follower Communication

Below is the communication diagram between the Leader and Follower. It shows how the two devices communicate with each other, and the type of data they transmit and receive. They are number coded so that it is straightforward to follow along in the state diagram above. Number 4 shows how the devices communicate to stop running.

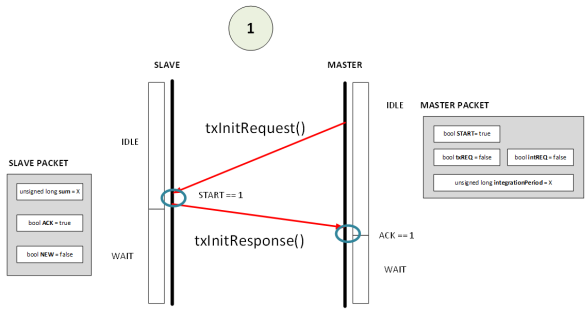

Figure 5: Initialization

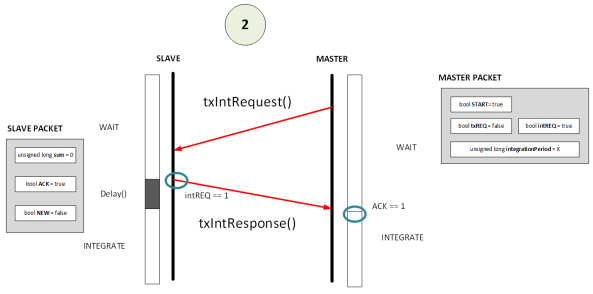

Figure 6: Integration

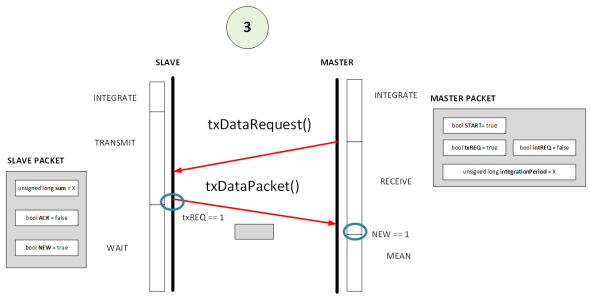

Figure 7: Transmission

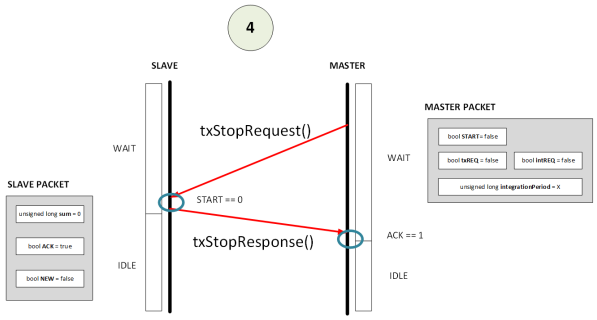

Figure 8: Termination

4.1.10 RXTX-Leader Communication

The Leader will send the DATA PACKET to the RXTX using ESP-NOW. This packet includes:

- **MAC Address (`mac[6]`):** This is the MAC address of the Leader device and is also used as its unique identifier.
- **TYPE:** The type specifies what type of packet is being transmitted. It's either a DATA packet or a PRESENCE packet. DATA packet means the fields `timestamp`, `mean`, and `packetID` are important. `PRESENCE_packet` is only used to inform the RXTX that the Leader is still connected. If neither DATA or PRESENCE packets are being received, then the device is considered to have lost connection, and a timeout will occur. The RXTX will then erase the entry in the DATA TABLE (more on this later).
- **STATUS:** The status specifies if the device is running or idle.
- **`newData`:** This specifies if new data has been received by the Leader. This is important for the DATA TABLE (more on this later).
- **timestamp:** This is the time of when the data point (`mean`) was processed, at the end of its integration period.
- **mean:** This is the value computed by the square-root-product mean of the sums of the Leader and Follower.
- **packetID:** This is an integer value that keeps score of the packet number received since it started running. It starts at 1 when the device initializes and increments every time a new data packet is sent.

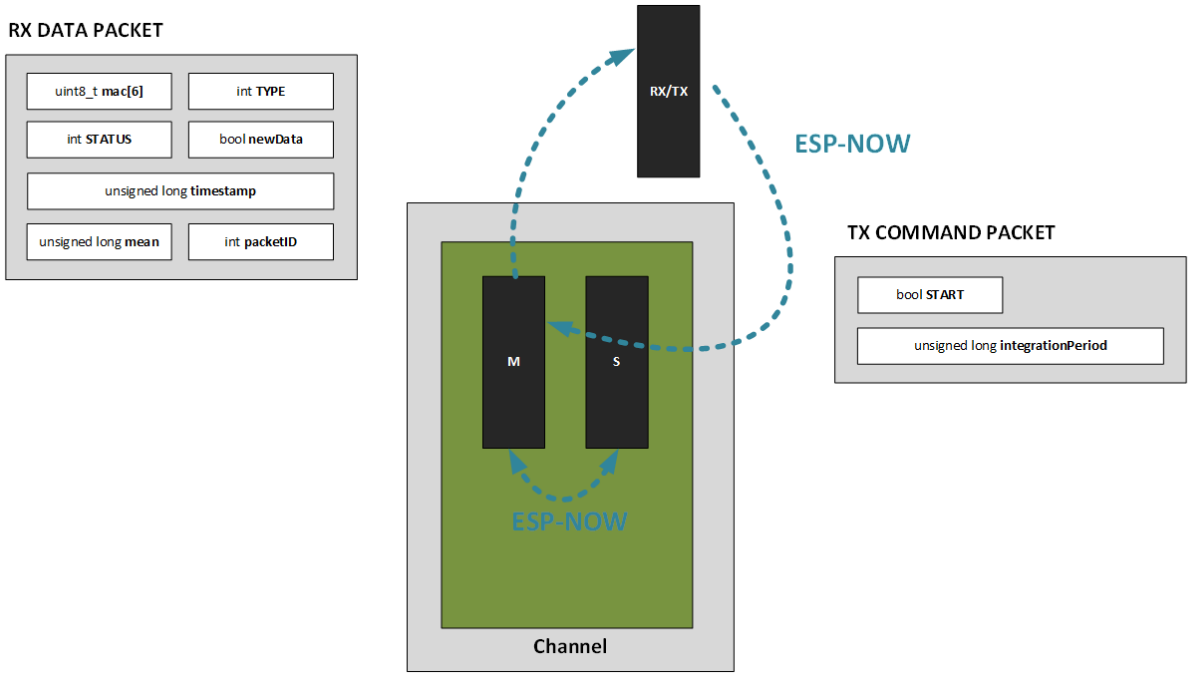

Figure 9: Leader-RXTX Packet Exchange

The COMMAND PACKET is sent from the RXTX to the Leader. It includes:

- **START:** This field tells the leader when to start running (`START = 1`) or stop running and become idle (`START = 0`).
- **integrationPeriod:** This field tells the leader what integration period needs to be set for the entire channel (Leader and Follower).

4.1.11 RXTX Component

The RXTX component is responsible for packaging the data received from each channel (Leader) into a DATA TABLE. This DATA TABLE is then sent to the application on the computer. The DATA TABLE is sent periodically to the application, and the application will interpret what data is important and what is not based on the fields for each entry in the table (more on this later). The RXTX also periodically checks the last time it received a packet from a specific Leader. If it exceeds a specific amount of time. The device is said to have lost connection and a timeout occurs. The RXTX effectively removes the entry associated with the timed-out device by setting the isValid flag to false (more on this later). The RXTX also receives the COMMANDS TABLE from the application (more on this later). The RXTX interprets the COMMANDS TABLE and transmits the commands to each leader.

4.1.12 RXTX-Computer Communication

Communication between the RXTX and the computer is done through USB port using the UART (Universal Asynchronous Receiver / Transmitter) communication protocol. This means that the RXTX would have to be connected to the computer using a cable. Via the COM port, the computer receives all Serial.print statements executed by the RXTX. This includes the actual DATA TABLE as well as debugging information. For the application on the computer to determine what's important and what's not, a start marker and an end marker are used, sandwiching the data table that is being sent.

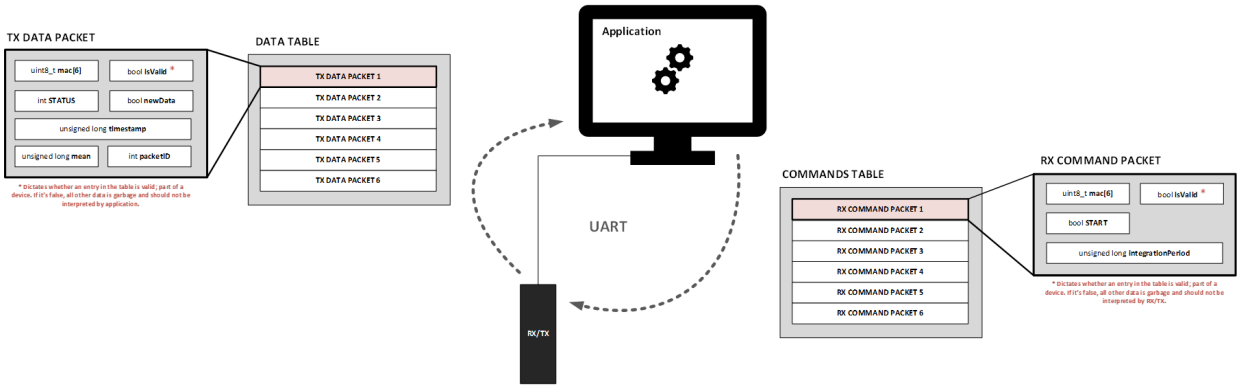

Figure 10: RXTX-Computer Table Exchange

In the diagram above, the RXTX transmits the DATA TABLE which includes entries of DATA PACKETS from each Leader. The table is a fixed size. What determines if a DATA PACKET is valid is the isValid flag. Before any device connects, all isValid flags are false. This means no devices are connected. When a device establishes a connection, the isValid flag turns true. Similarly, when a device times-out, the isValid flag would become false, freeing up the table for other connections.

When the isValid flag is true, the application assumes all other fields in the DATA PACKET are valid, meaning they belong to an actual Leader that is connected. The other fields are:

- **MAC Address (`mac[6]`):** This is the address of the Leader device and is used as the unique identifier of the channel.
- **STATUS:** This takes on two states, RUNNING (1) and IDLE (0). The application can confirm if the channels have started or stopped.
- **`newData`:** This flag means that there is new data to be read (`newData = 1`) by the application (mean, timestamp, and `packetID`). Otherwise, when `newData = 0`, ignore the data fields.
- **timestamp:** This is the time captured at the end of the integration for that channel.
- **mean:** This is the mean value computed after the square-root-product mean of the two integrations.

- **packetID:** This is the counter of the data packets being sent. It starts at 1 when the channel starts running and increments every time a new DATA PACKET is sent. This can be used by the application to determine if any packets have gone missing, implying there is a problem with the connection, and should be notified to the user.

The computer application transmits the COMMANDS TABLE to the RXTX. The fields in this table are:

- **MAC Address (`mac[6]`):** This is the address of the Leader device and is used as the unique identifier of the channel.
- **START:** This tells the leader when it should start (1) / stop (0) running.
- **isValid:** When isValid is true, it means the command should be transmitted to the designated leader, otherwise ignore it.
- **integrationPeriod:** This sets the integration period (in ms) for the designated channel (leader) by the user.

4.1.13 Firmware Conclusion

The firmware is responsible for a great deal of the processing. Maintaining synchronicity and data integrity is paramount. This document is merely to understand the high-level nature of the firmware developed. Please review the code for a thorough understanding of the lower-level parts and the actual implementation. The firmware is broken down into 3 arduino files: rxtx.ino, follower.ino, and leader.ino. Below is a diagram that shows the full system overview of the firmware.

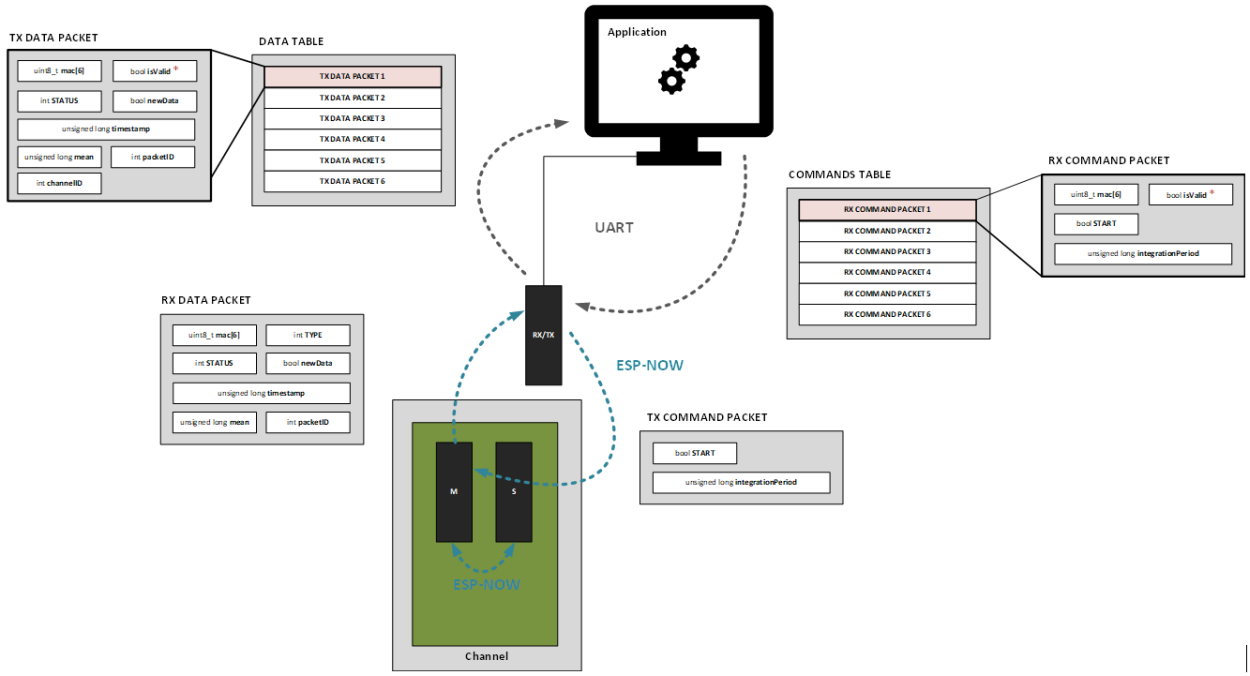

Figure 11: Full Firmware System Overview

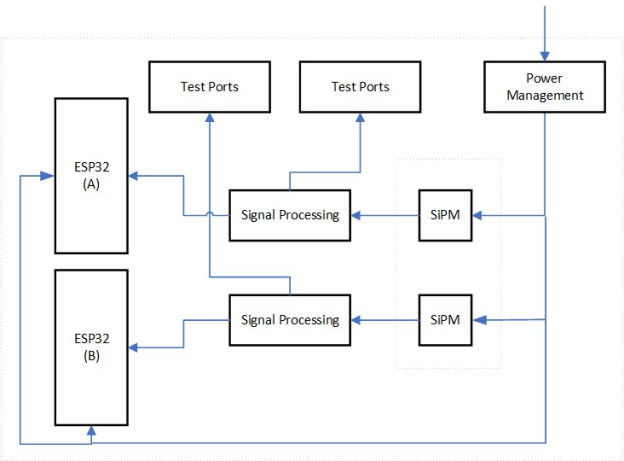

Figure 12: System Diagram of a detector module

4.2 Hardware Solution

4.2.1 System Overview

The detector system consists of 4 channels, each channel being handled by a single detector module. With a single channel per module, it allows the detector system to be easily scalable to accommodate more channel in the future. It also increases maintainability of the overall system, since if module becomes defective it can be easily remove and replaced by another modules while it is being repaired or troubleshooted.Indeed, the detector modules are to be connected and stacked vertically to be able to fit on our user's workspace that is already crowded with various devices.

Each detector module features multiple parts: the main power entry, the module-to-module power lines, the signal processing circuity and the SiPM power and signal interface. The detector system will connect and communicate with the user's computer via the RXTX module connected to the computer. This enables wireless communication between the detector system and the capture software. Wireless communication allows the user to perform a scintillation capture while staying at their desk which is a lot more convenient for their workspace configuration.

4.2.2 Signal Processing Simulation

It is paramount to simulate the behavior of the electronic circuitry before prototyping the signal processing circuitry and creating a PCB realization. LTSpice is the software used for these simulations.

Firstly, the signal processing circuitry had to be modeled on LTSpice. The circuitry is powered by the 3.3V supply of the ESP32, coupled with VCC filtering to remove any fluctuations in power output.

It is also important to model the raw SiPM signals. This was achieved using the EXP functionality in LTSpice. Three peaks of varying amplitudes were modeled: 6mv, 32mv, 100mv. This validates that the signal processing circuitry has a large dynamic range and can detect peaks of various amplitudes.

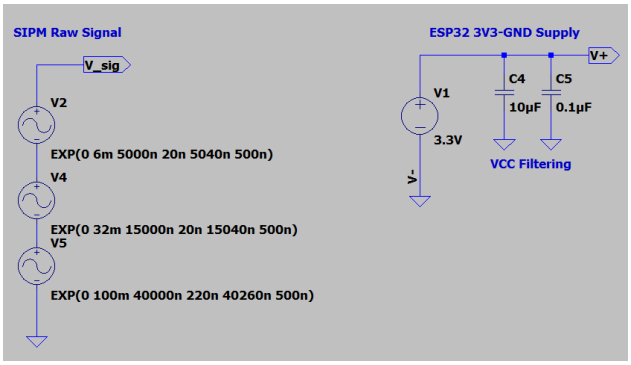

Figure 13: ESP32 Power Supply and raw SiPM signals

The signal processing circuitry itself was modified in comparison with Hailey's prototype. The amplifying circuit's gain was changed from 24V/V to 30V/V. The peak detector circuit contains two OPA354, a high-speed operational amplifier (op-amp) from Texas Instruments, designed for applications requiring high bandwidth, low noise, and fast settling time. It is particularly suited for precision signal processing, data acquisition, and buffering tasks. A peak detector implementation was found at Physics Open Lab [2]. The circuit was copied onto LTSpice and modeled to validate the behavior.

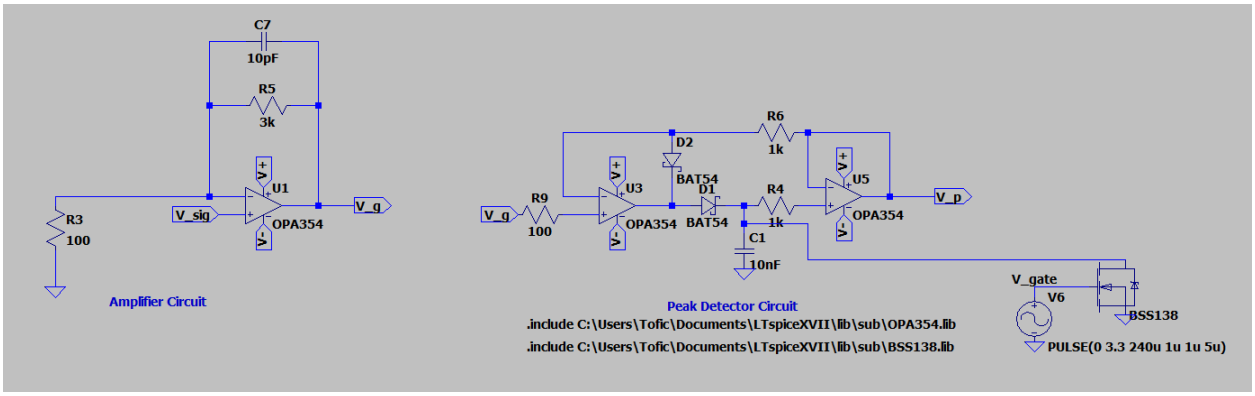

Figure 14: Signal Processing Circuitry: Amplifier and Peak Detector

The capacitor, C1, is charged when a peak is detected. Normally, if it were connected in parallel with a resistor, the capacitor would slowly discharge at the time constant proportional to RC.

$$\tau = R \cdot C$$

However, with the introduction a MOSFET transistor, BSS138, the capacitor can be rapidly discharge when the gate voltage is set to high. This allows the peak detection circuit to detect peaks more frequently. The peak is held for 200 µs. Control of the peak hold time and discharge is entirely handled by the firmware.

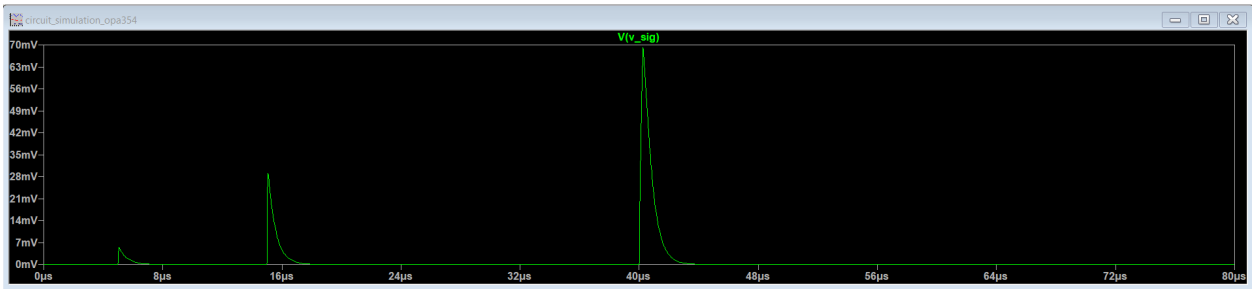

Figure 15: SiPM Raw Signals

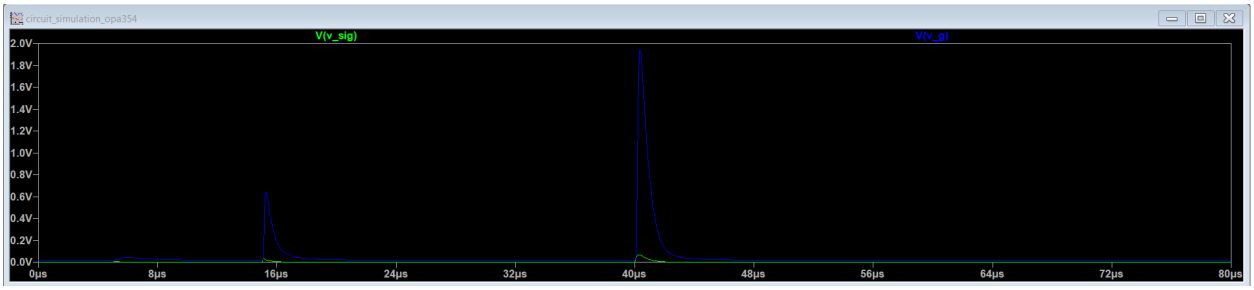

Figure 16: Amplified SiPM Signals at 30V/V

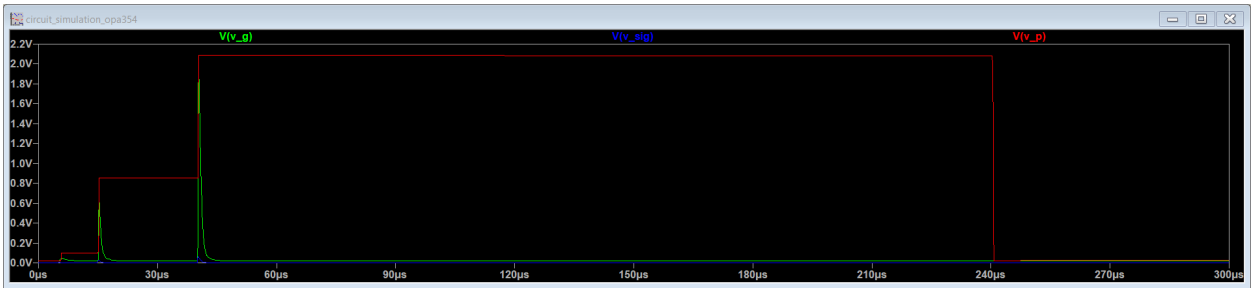

Figure 17: Peak Detected Amplified SiPM Signals

4.2.3 DC-DC Booster Simulation

The DC-DC booster is required to supply a bias voltage to the SiPM of about 29.5V. Otherwise the SiPM would not produce pulses when detecting photons. Similarly to the signal processing circuitry, the DC-DC Booster circuit was also simulated. The idea is to supply 5V directly from an external power source, to the DC-DC Booster. This circuit was taken from the Cosmic Watch project [3].

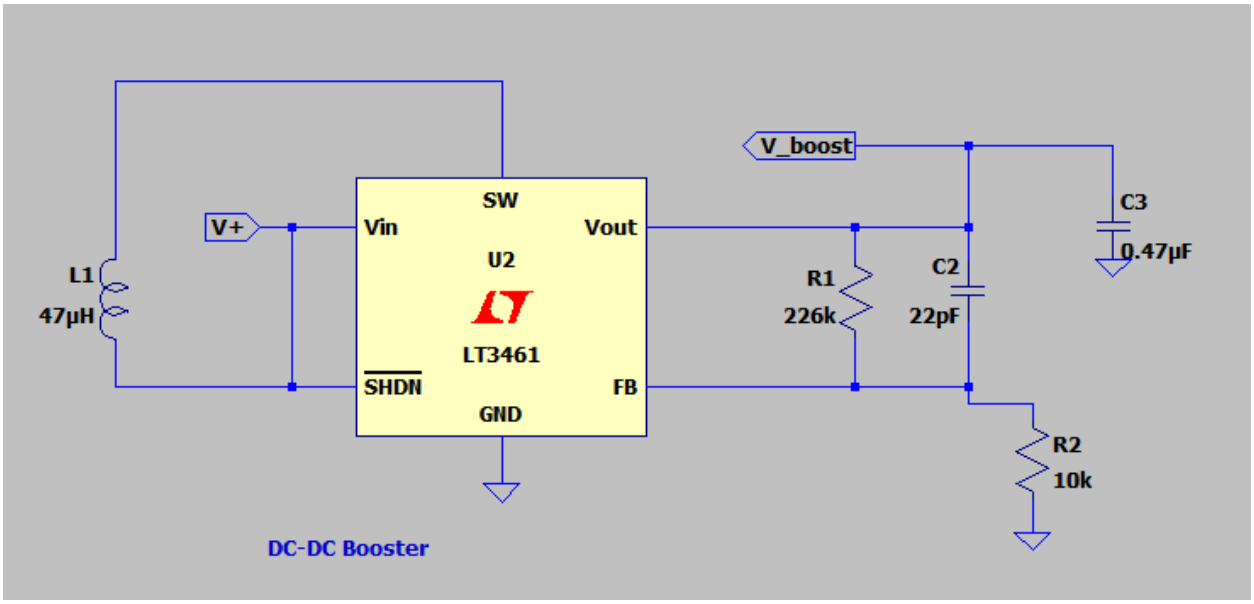

Figure 18: DC-DC Booster Circuit

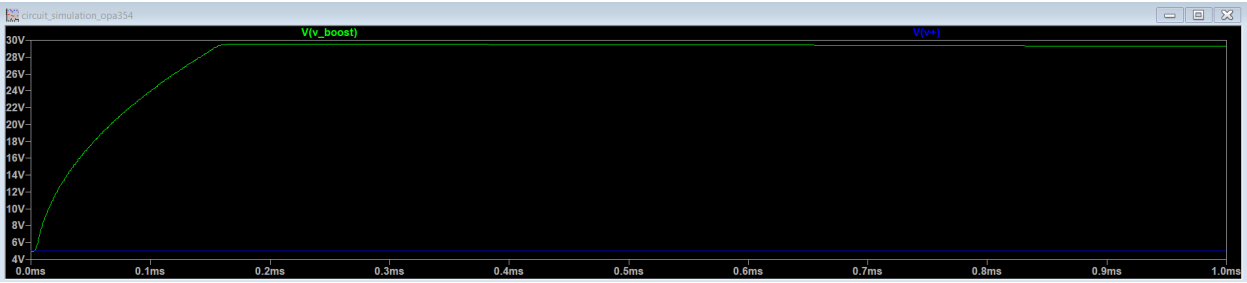

Figure 19: Voltage Boosted Supply

4.2.4 Power Circuitry

The design of the power circuitry is aimed at providing power to the micro-controller units, the signal processing circuitry, the SiPMs and power the other modules. There is a main power entry is the DC barrel plug of the module. The plug can be connected to a wall outlet using a standard 3A 5VDC AC-DC converter cable similar to laptop chargers. That feeds the system a constant 5VDC. Each module has a power switch that turns on or off all power to the board and to the power line. The power is filtered and routed to the different parts of the system to ensure optimal functionality. Once the modules are connected together, they can all be powered via the power line and independently turned on or off with their switch. The reason for a shared power line and independent power switches was to have a highly modular and flexible system where each module is identical and interchangeable. Therefore, the user does not need to worry about following complex instructions in order to connect and power the detector system. Simply connect a module to a wall outlet and connect the other modules to the first one and the user can power on the system and the number of desired modules effortlessly. The power consumption of

Components	Supply Voltage (V)	Power Consumption (mA)
ESP32	3.3(5)	240
Peak detector op-amp (AD8602)	2.7(5.5)	1.3
Voltage amplifier op-amp (LT1807)	2.5(12.6)	1.3
DC-DC Booster op-amp (LT3461)	3.3(5)	In: 2.8(3.6) Out: 250
SiPM	29.5	20
Total SiPM Power Consumption	Single module: 40 mA	4 channel detector: 160 mA
Total Board Power Consumption	Single module: 508.8 mA	4 channel detector: 2035.2 mA

Table 2: Power Consumption of the components

all the main components of a module was researched in order to determine and estimate the power consumption of a module and of a 4 channel detector system. These values dictate the current rating of the power connectors and the power cable.

4.2.5 RXTX PCB Realization

The schematic of the RXTX PCB is straightforward. It connects the digital pins of the ESP32 to the digital pins of a small OLED display. The display rests on the bottom side of the PCB and the ESP32 rests on the top. The pins are soldered to the pads to ensure a stable electrical connection.

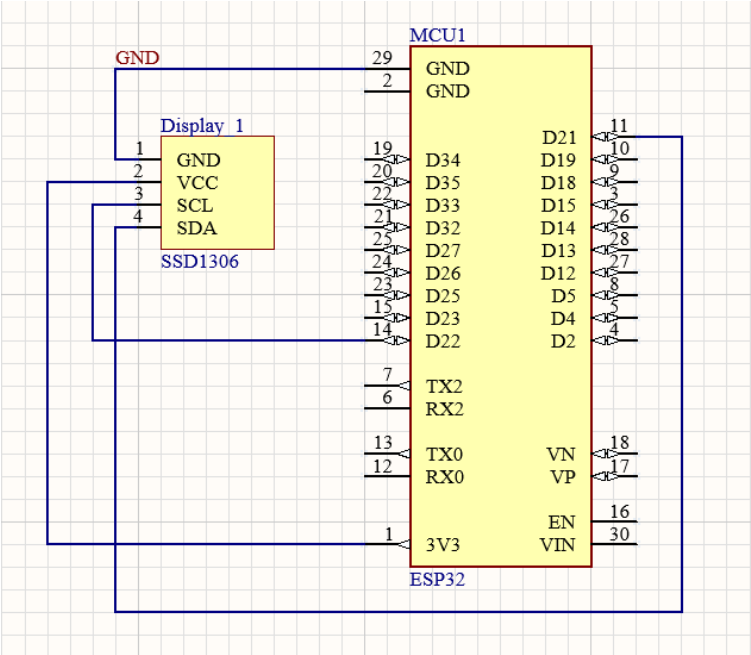

Figure 20: RXTX Circuit

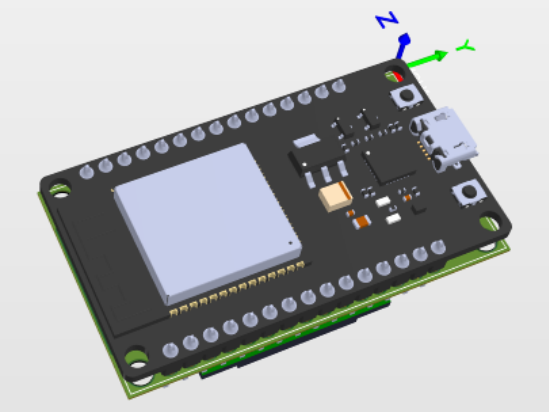

Figure 21: RXTX PCB Top Perspective

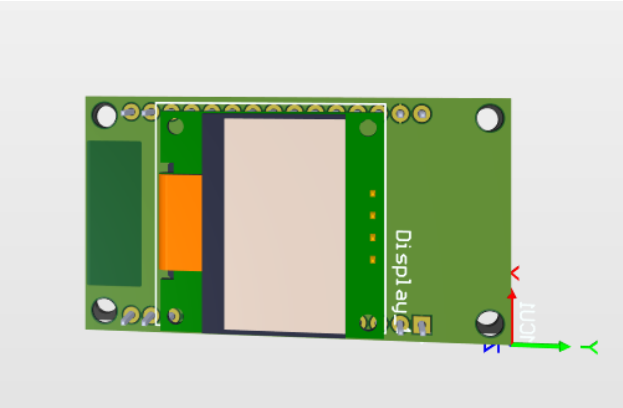

Figure 22: RXTX PCB Bottom Perspective

4.2.6 SiPM PCB Realization

The schematic of the SiPM PCB follows precisely the one designed by the Cosmic Watch Project. The PCB itself was designed using Altium Designer. The layout has been re-designed to reduce size.

We used various soldering techniques to assemble the SiPM PCBs. The SiPMs were solder onto to the board using reflow soldering. This consists of applying solder paste onto the SiPM pads located on the board and then placing the SiPM onto the covered pads. The SiPM PCBs are then placed inside a reflow oven that will follow a reflow profile to heat up the solder and safely cool it down to ensure optimal solder joints. The passive components and the connector were hand soldered using a soldering iron. We also used a hot air gun to re-solder some of the SiPMs to their boards that weren't properly soldered initially.

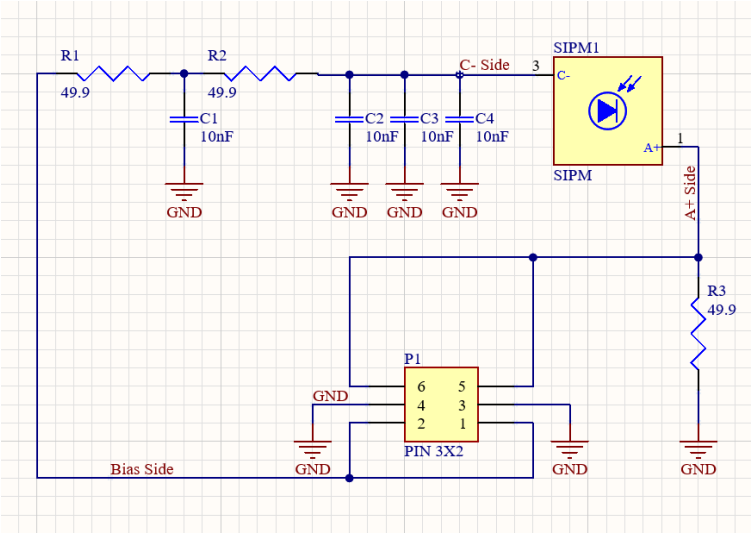

Figure 23: SiPM PCB Circuit

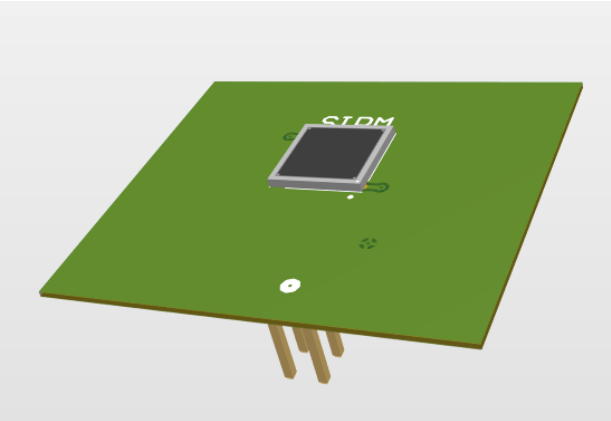

Figure 24: SiPM PCB Angled Perspective

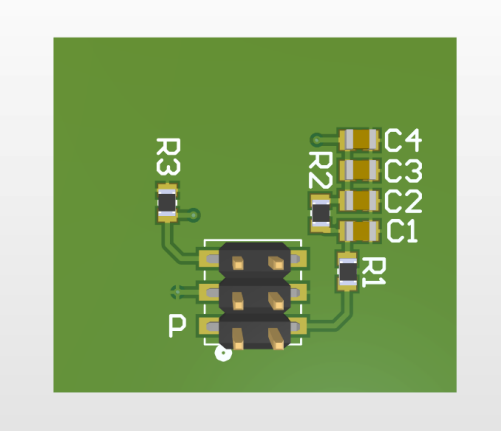

Figure 25: SiPM PCB Bottom Perspective

4.2.7 Main PCB Realization

The signal processing circuitry, the power circuitry, the micro-controller units and the SiPM interface has been put together in for the main system schematic. It basically builds on the circuit simulation and the documentation read to come altogether in a single board.

Multiple inputs and outputs have been added to the design in order to make the system more user-friendly or easier to troubleshoot. The notable inputs and outputs are a LED to indicate when the module is powered, a OLED screen to display useful information about the module, header pins to serve as test points and allow probing of the SiPM signal as it goes through the signal processing stages, a reset button to reset the micro-controller units and BNC connectors to be able to connect an oscilloscope and analyze the raw SiPM signals.

Additionally, the micro-controller units will be connected to the board via header sockets to allow the MCUs to be replaced without having to manufacture another board. Following the design of the Cosmic Watch, the SiPMs will be interfaced to the main system using header pins for the same reason as the MCUs.

To assemble the main PCB, we used reflow soldering and hand soldering. Using a relfow stencil of the main PCB, the solder paste was applied to all the pads of the surface mount technology (SMT) components (resistors, op-amps, capacitors, inductors). The reflow oven was used again to solder those components. The rest of the components were through-hole devices that required to be hand soldered after the PCBs came out the oven.

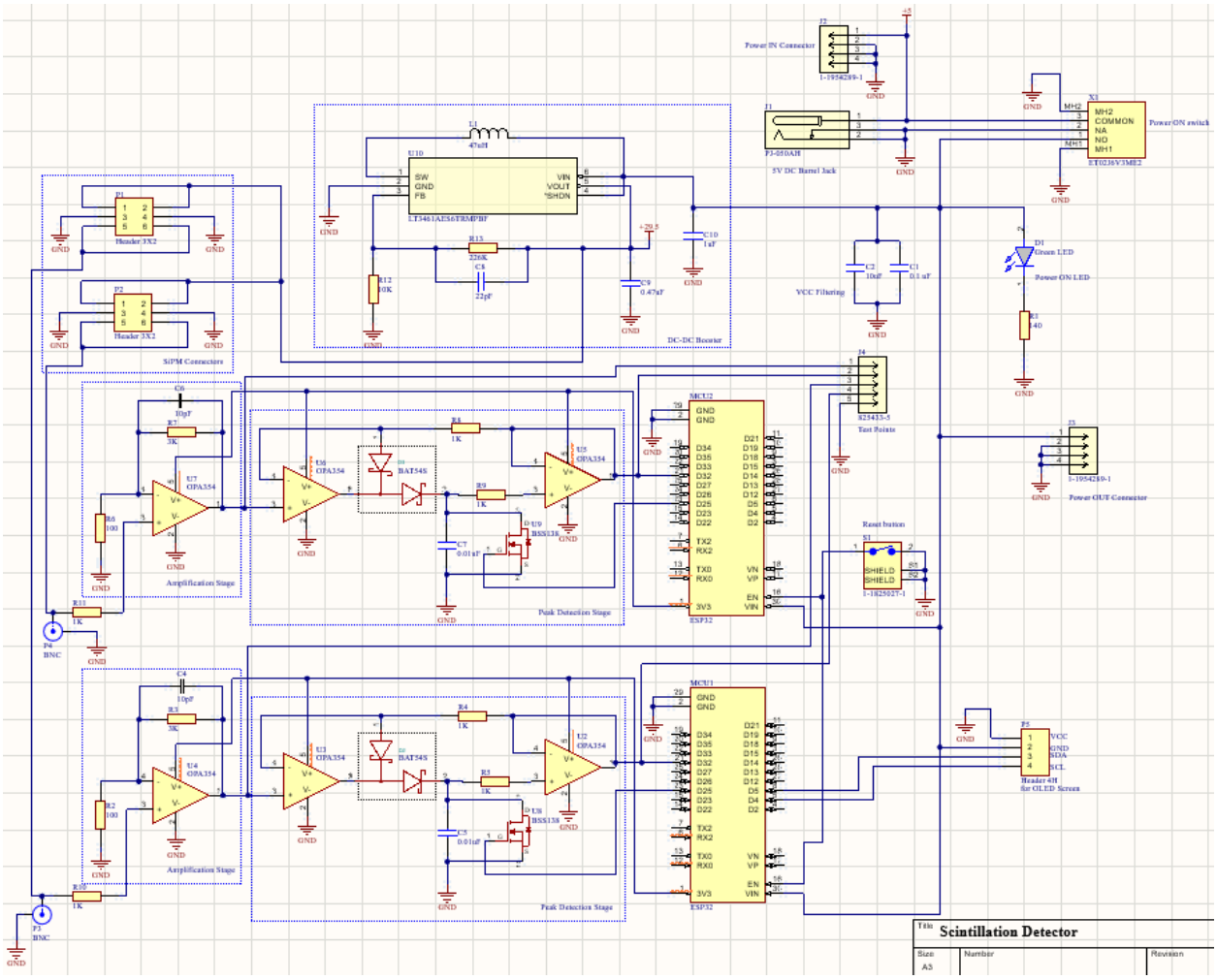

Figure 26: Main PCB Schematic

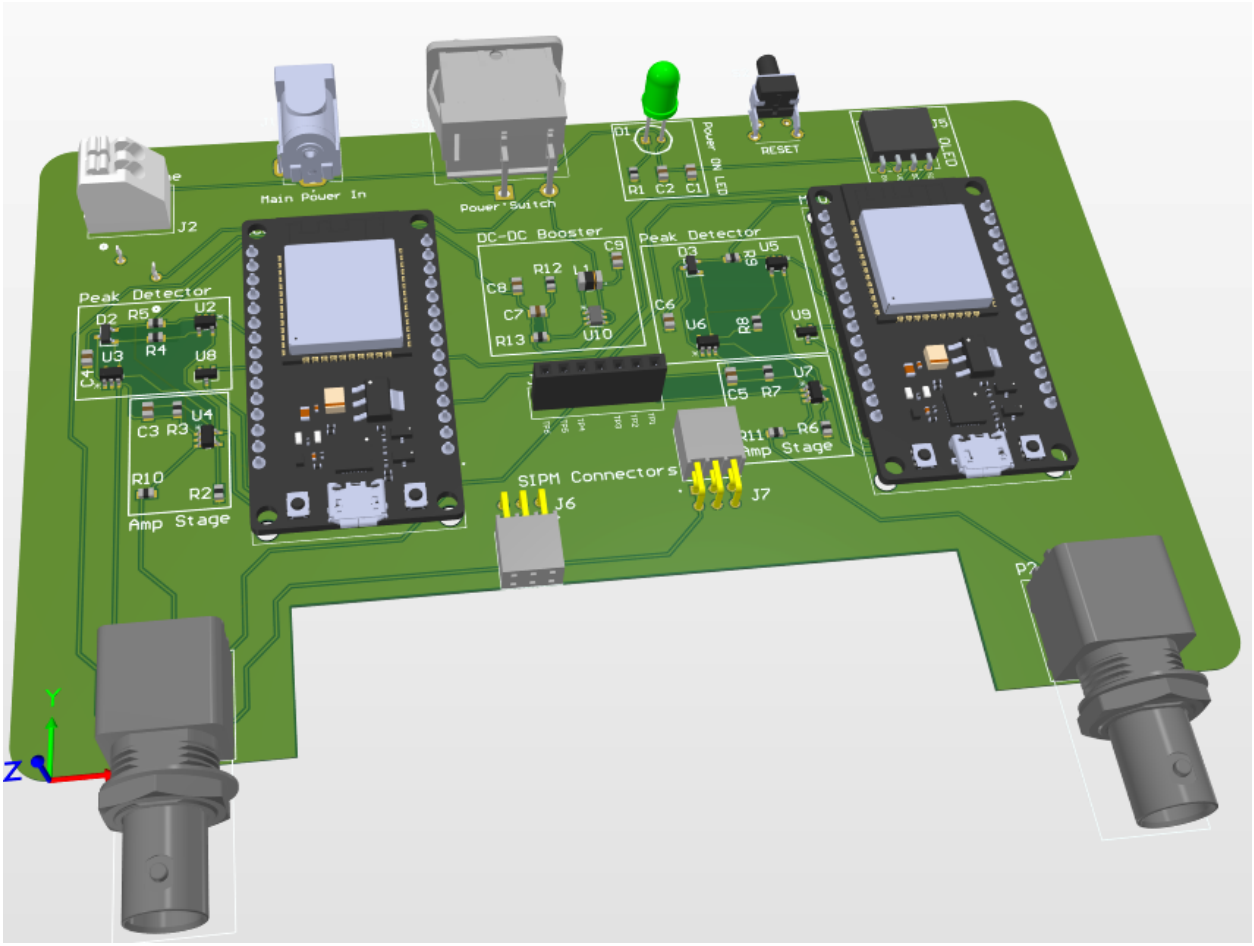

Figure 27: Main PCB 3D Layout

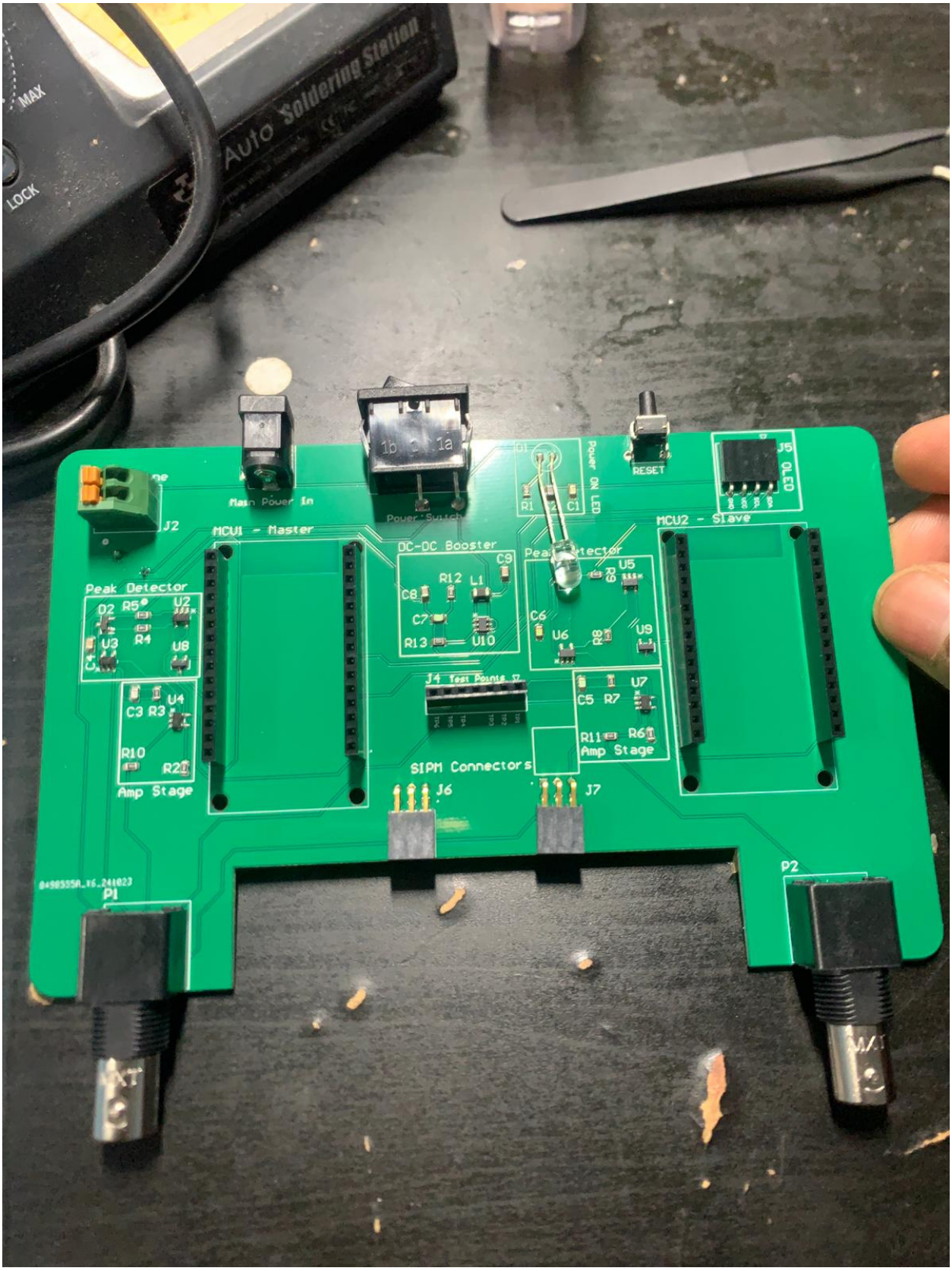

Figure 28: Assembled Main PCB

4.3 Enclosure Solution

All enclosure designs are done in Autodesk Fusion 360 software. This software allows to import 3D step files which allows a direct representation of how the PCBs fit in the enclosures. It is also a relatively easy and simplistic software to use, making it a great choice for quick prototyping under time constraints.

4.3.1 RXTX Module

The RXTX module contains 5 components:

1. **ESP32:** With rxtx.ino uploaded to it, this acts like the RXTX.
2. **SSD1306:** This is a small electronic display, 64 x 128 pixels, used to give information to the user; the number of channels connected.
3. **RXTX PCB:** This PCB has 4 traces that connect the digital pins of the ESP32 to the digital pins of the SSD1306.
4. **Top Cover:** This cover is 3D printed, and is the top part of the RXTX Module enclosure. It includes an opening for the SSD1306.
5. **Bottom Cover:** This cover is 3D printed, and is the bottom part of the RXTX Module enclosure. It includes air vent to dissipate the heat generated from the ESP32 and a slit to fit a micro USB cable to power the ESP32 and receive/transmit data between the RXTX and computer.

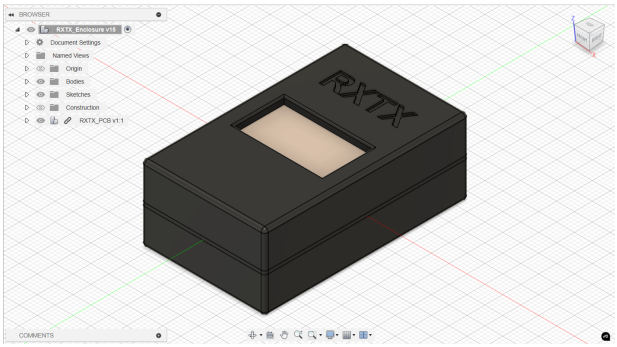

Figure 29: RXTX Top Cover View

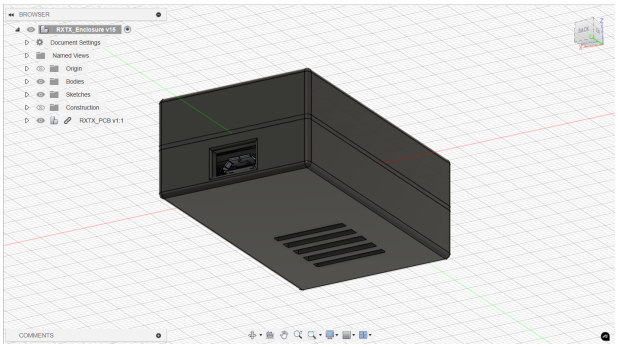

Figure 30: RXTX Bottom Cover View

The top and bottom covers were 3D printed. See below for the realization of the actual RXTX.

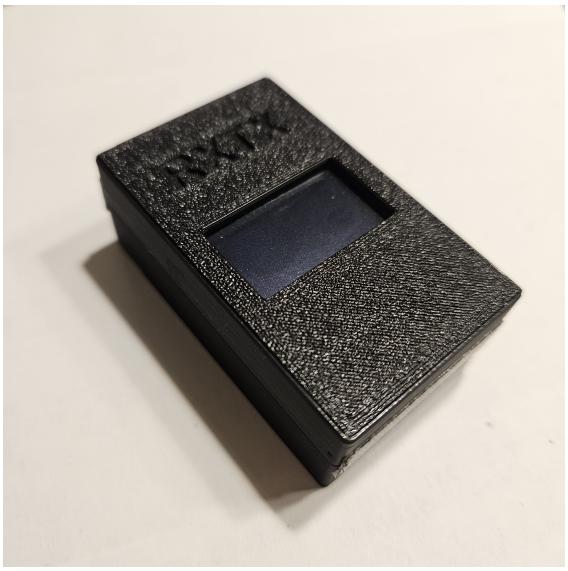

Figure 31: RXTX Powered Off

Figure 32: RXTX Powered On

4.3.2 Dual SiPM Light-Tight Enclosure

The Dual SiPM Light-Tight Enclosure contains 8 distinct components:

1. **2 x 'SiPM PCB':** First stage of signal acquisition.
2. **Bottom Layer:** This layer is 3D printed, and and is the bottom part of the Dual SiPM Light-Tight Enclosure. It contains space for 2 SiPM PCBs to rest inside while ensuring light-tightness.
3. **Middle Layer:** This layer is 3D printed, and and is the middle part of the Dual SiPM Light-Tight Enclosure. It contains two tunnels where the light guide will be placed inside and a cutout for the SiPM to be surrounded by, providing optimal light-tightness.
4. **Top Layer:** This layer is 3D printed, and and is the top part of the Dual SiPM Light-Tight Enclosure. It contains spacing for the ADASMA adapter to be placed inside.
5. **2 x 'ADASMA':** This is the adapter where the optical fiber cable will be attached. This component leads to the light guide.
6. **2 x 'FEC5 - Fiber End Cap':** This is a crystalline structure that allows photons to pass through, behaving as a light guide.
7. **4 x 'M5-0.8 Brass Inserts':** These metal inserts are place inside the bottom layer.
8. **4 x 'M5-12mm Strainless Steel Screws':** These screws are secure all components together, by creating a tight seal between the layers and ensuring light-tightness.

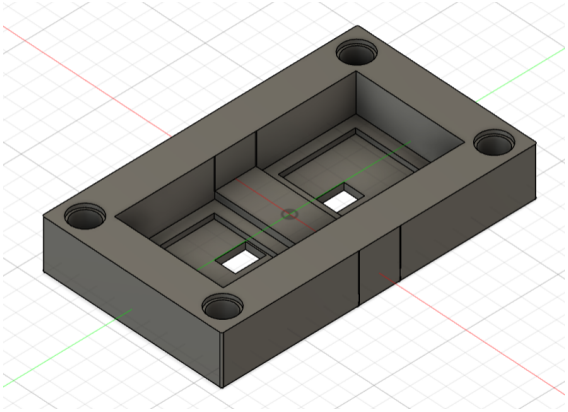

Figure 33: SiPM Enclosure Bottom Layer

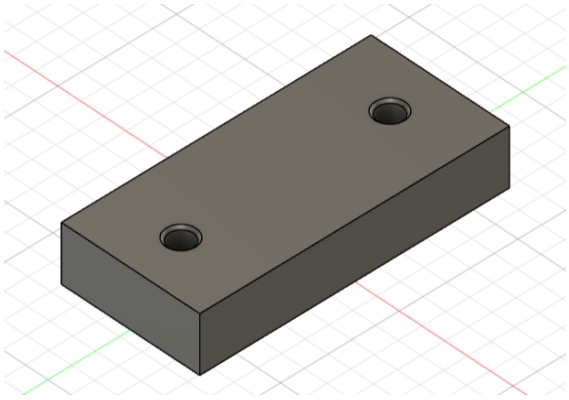

Figure 34: SiPM Enclosure Middle Layer

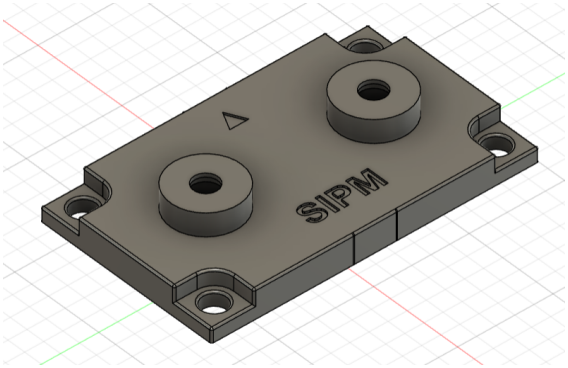

Figure 35: SiPM Enclosure Top Layer

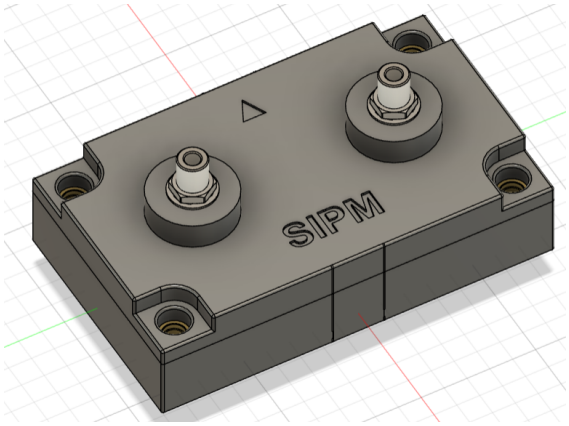

Figure 36: SiPM Enclosure Complete

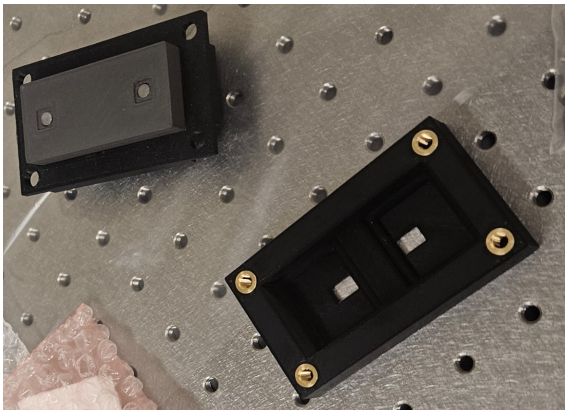

Figure 37: SiPM Enclosure Assembly

Figure 38: SiPM Enclosure Top View

4.3.3 Channel Module

The Channel Module Enclosure contains x distinct components:

1. **SiPM Enclosure:** Defined previously.
2. **Main PCB:** Defined previously.
3. **Box:** The 3D printed box contains railing for the main PCB to slide through, and 4 circular openings for the ADASMA adapter and coax outputs. It also contains vents for air circulation to prevent heating of stagnant air.
4. **Lid:** The 3D-printed lid contains openings for all the user-interfacing components. It is used to close the opening of the box.
5. **2 x 'M5-0.8 Brass Inserts':** These metal inserts are at the edges of the box.
6. **2 x 'M5-12mm Strainless Steel Screws':** These screws are secure the box and lid together.
7. **XT-60 Power Connectors:** These power connectors are placed vertically to ensure power distribution when multiple detectors are used.

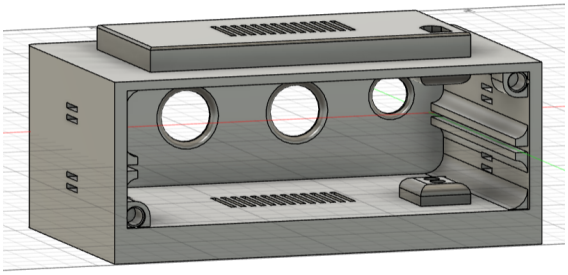

Figure 39: Box

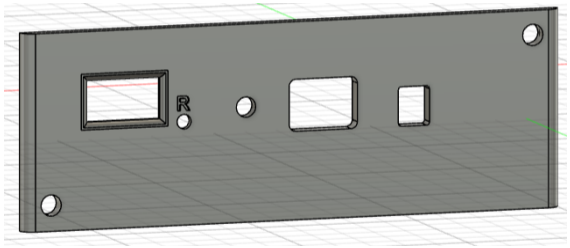

Figure 40: Lid

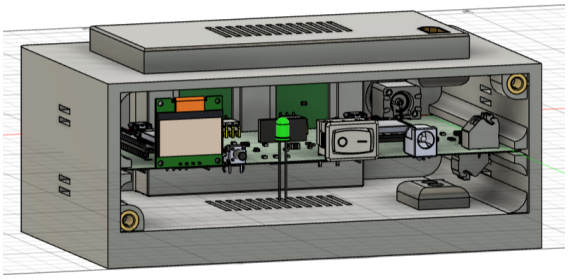

Figure 41: Channel Module Opened

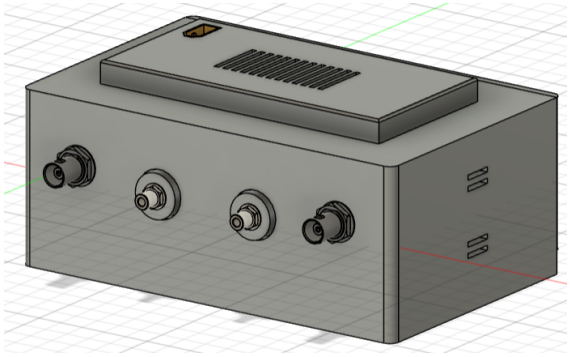

Figure 42: Complete, Front View

Figure 43: Channel Module Back View

Figure 44: Channel Module Front View

4.4 Modularity

The modular design of the detectors for multi-detector capabilities is supported by the firmware, application, hardware, and enclosure design.

For additional detectors, subsequent detector modules can be stacked on top of each other. This would ensure power distribution to all detectors.

The cross-section view shows how the vertical power connectors connect.

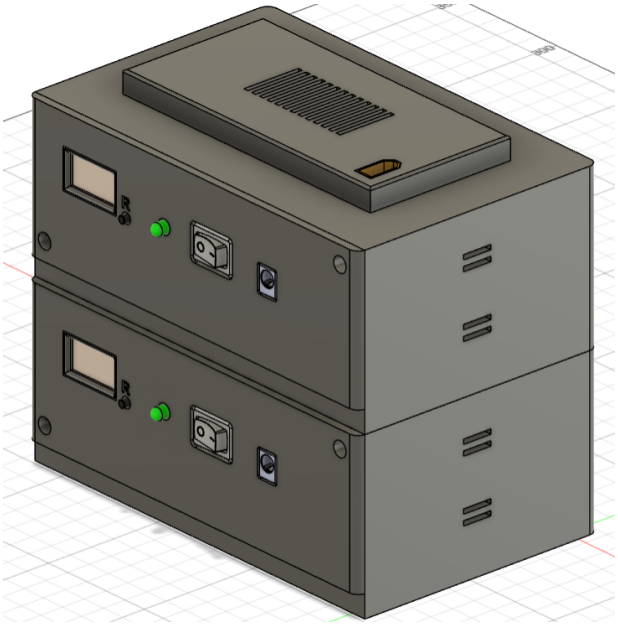

Figure 45: Modular View

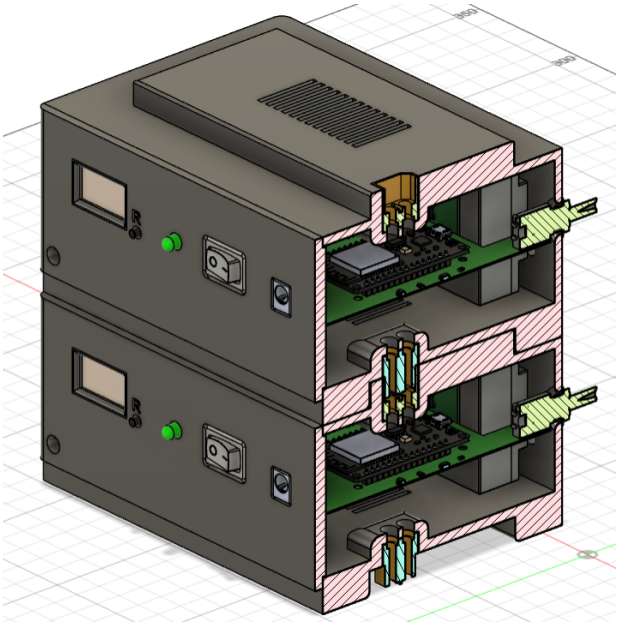

Figure 46: Modular Cross-Section View

4.5 Software Application Solution

4.5.1 Software Overview

The software application is a Python, GUI-based application that monitors, processes, and plots data received from a serial port. It primarily relies on the PyQt5 library for the graphical user interface (GUI) and matplotlib for plotting data. Additionally, it uses

pySerial to interface with serial devices, which is essential for reading binary data from external hardware (e.g., sensors).

The application detects channels through MAC addresses received from the serial port, assigns them to virtual channels, collects their data, and visualizes the ”mean” value over time. It also allows users to configure and control each channel’s settings, such as integration periods. Once data collection stops, the user can export the data as CSV or PDF files.

4.5.2 Main Features

Serial Port Detection: The software automatically detects active serial ports that are sending data. This ensures the correct port is used for communication without manual configuration.

Binary Data Handling: It reads binary-encoded data packets over the serial connection, parses them, and extracts relevant information such as MAC addresses, validity flags, status, mean values, timestamps, and packet IDs.

Dynamic Channel Management: When valid data is received from a new MAC address, the program dynamically assigns it to an available channel. The user can view, modify, and control each channel’s integration period (sampling period) and activation status.

Real-Time Data Plotting: The application continuously updates a real-time plot of ”mean” values for different MAC addresses (channels). This visualization is performed using matplotlib and is rendered in the PyQt window via a FigureCanvas.

Data Export: The user can export the collected data into CSV files or PDF format for further analysis. Export functionality is provided only after data collection is stopped.

Threaded Operation: The serial port data collection and real-time plotting are handled in separate threads. This design ensures the GUI remains responsive while data is processed and visualized in the background.

4.5.3 Design Choices

Modular Structure: The application is divided into several components to separate concerns: Serial data reading and processing happen in a dedicated thread (SerialThread). Real-time plotting occurs in the PlottingThread, ensuring continuous updates to the plot while data is received. The main GUI and control logic are handled by the MainWindow class, which manages user interaction and rendering.

Threaded Design: By using threads for serial communication and plotting, the software avoids blocking the main event loop, allowing smooth user interactions. This ensures that even if serial data is being processed, the user can still interact with the UI.

Control Panel Integration: A form-based control panel is dynamically generated to allow interaction with the various channels. For each MAC address detected, a new row is added to the panel, allowing the user to configure its integration period and activation status.

Real-Time Plotting: The design of the PlottingThread separates data plotting from data processing, meaning the graph is continuously updated as data streams in. Data for

each channel (MAC address) is stored and displayed with proper labels to distinguish them on the graph.

Input Validation: The program ensures that the data entered by the user for integration periods is valid, providing error messages when invalid data is entered.

4.5.4 Data Handling

Serial Data Input: There is a function that continuously scans available serial ports until one with data is found. Once a valid port is detected, it's opened, and the program listens for incoming data packets. Binary data packets are processed and read by a function, which reads data byte by byte and buffers it. When a complete packet is detected (between start and end markers), it is passed to a function that processes the binary message.

Binary Data Parsing: Data packets consist of several fields, including a MAC address (6 bytes), validity flag (1 byte), status (4 bytes), new data flag (1 byte), mean (4 bytes), timestamp (4 bytes), and packet ID (4 bytes). This data is unpacked using the struct module and then emitted to the main application via a signal, where the GUI can process it further.

Data Queue: The application uses a queue.Queue to store incoming data. The PlottingThread retrieves data from this queue for real-time plotting, ensuring data is processed in a thread-safe manner.

Data Storage: Data is stored in a dictionary (or map) where the MAC address is the key, and the values are lists of timestamp and mean value pairs. This allows the application to keep a history of data for each channel, which is visualized and can later be exported.

Exporting Data: Data can be exported to a CSV file for each MAC address. The CSV contains the timestamp and mean value recorded for that particular channel. The plot can be exported to a PDF file using matplotlib's PDF backend, providing a snapshot of the data visualization.

4.5.5 GUI Description

A. Control Panel (Right section)

This panel provides options for configuring the system, managing channels, and interacting with the data stream. The control panel dynamically updates based on detected MAC addresses and allows control over the system's behavior.

- Serial Port Display: At the top of the control panel, the detected and active serial port is displayed once it is detected and opened. The user can see which port is currently in use.
- Start/Stop Button: This button allows the user to initiate or stop the data collection process. When pressed: Start: Opens the serial port, starts receiving and plotting data in real time. Stop: Halts data collection, allowing the user to export or save the data collected.

- Dynamic Channel Management Table: A dynamically updated table that lists all detected MAC addresses and provides controls for each channel:
- Channel: Displays the MAC address of the connected devices.
- Integration Period Input: A text input field where users can set the integration (sampling) period for each channel.
- Activation Checkbox: Allows users to enable or disable specific channels.
- Save Settings: Applies any changes made to the integration period and activation status.
- Reset: Resets the settings of a channel to the default.
- Export Options: Buttons at the bottom of the control panel allow the user to:
- Export to CSV: Save all collected data as CSV files, with each MAC address having its own CSV file containing timestamped mean values.
- Export to PDF: Export the real-time plot to a PDF, providing a visual snapshot of the data at the time of export.

B. Data Plotting Area (Left Section)
This section of the GUI is dedicated to visualizing the incoming data in real time.

- Real-Time Plot: A matplotlib-generated plot that shows the data stream from the detected channels (MAC addresses). The plot automatically updates as new data comes in. Each MAC address (channel) has a separate line, clearly labeled, showing the mean value over time. The plot auto-scales to fit the data as it comes in, ensuring visibility of trends across all channels. The x-axis represents time, and the y-axis represents the ”mean” value received from the data packet.
- Plotting Updates: The plot refreshes regularly as new data is received from the serial port, ensuring a near real-time display of the data. The threading mechanism ensures that the GUI remains responsive during this process.

4.5.6 Firmware-Software Integration

The firmware on the ESP32 or rx/tx communicates with external software through UART. The firmware uses ESP-NOW for wireless communication, exchanging data packets such as sensor readings or status updates with other ESP32 devices. For external communication with the software application specifically, the firmware uses UART (via a USB connection to the laptop/device running the software application) to send and receive data in binary format. This integration allows the software to control the firmware's behavior, request data, and receive real-time updates, while the firmware handles data collection, processing, and communication tasks.

5 Results

5.1 Firmware Tests

Extensive tests were carried out during the development of the firmware:

- ADC absolute sampling frequency
- ADC summation delays
- State transitions of Follower and Leader
- ESP-NOW communication between Follower and Leader
- ESP-NOW communication between Leader and RXTX

To monitor and verify successful state transitions, a test setup was built. The displays show important information, such as:

- The current state of the Follower/Leader
- Summation after integration
- The computed mean by the Leader

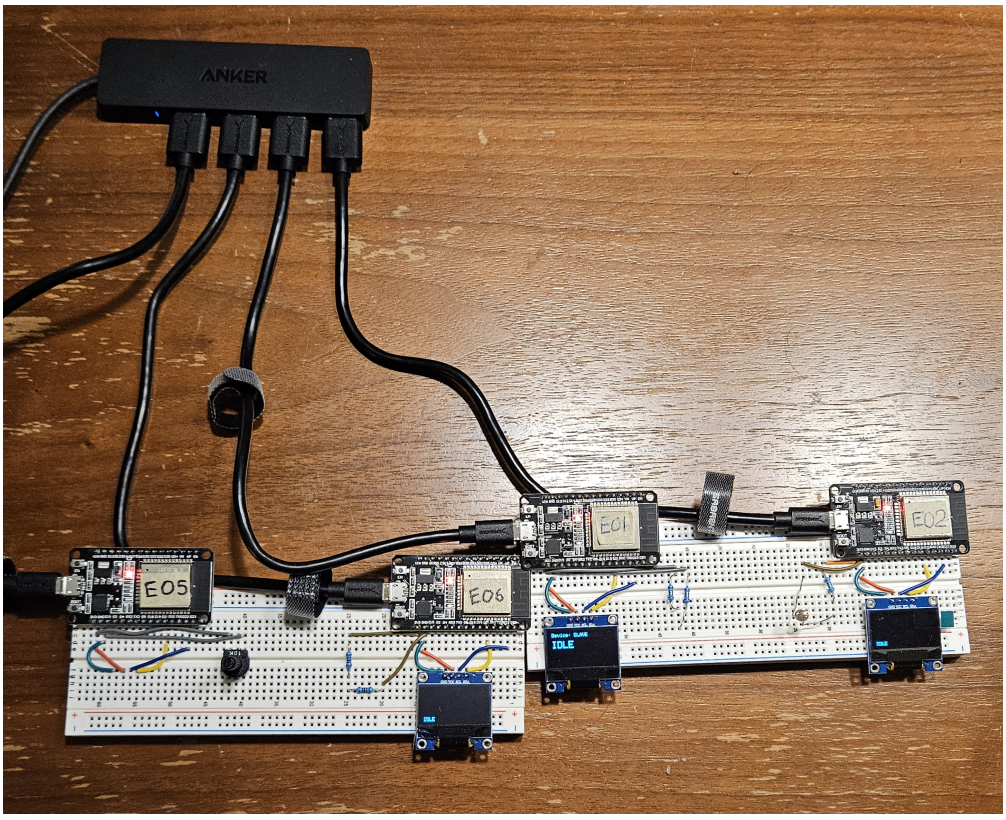

Figure 47: Firmware Test Setup

5.2 Software Tests

A number of tests were conducted throughout the development of the software to validate its functionality.

- UI/UX Testing: To verify that the user interface (UI) is intuitive, functional, and free of errors, we tested each feature, button, and input field in the software, and checked the layout and interaction flows across different devices and screen sizes.
- Input Validation Testing: To ensure that user inputs are correctly validated, wecentered invalid data in input fields (e.g., incorrect MAC addresses, out-of-range values) and observe the software's response.
- Data Handling Testing: To ensure that the software handles data properly without real-time data from the firmware, we simulated data inputs or use mock data (e.g., dummy ESP-NOW data) to test how the software processes and displays the data.
- Data Transmission Simulation Testing: To simulate the software's ability to send commands and receive data without firmware, we used mock data to simulate sending/receiving data, testing how different responses are handled.
- Performance Testing: To assess the software's responsiveness and performance under different loads, we simulated multiple connected devices or large data inputs and monitored the software's performance.

In addition, after the detector was assembled and the subsystems linked together, we conducted integration tests. Namely, we conducted a test with a light source that we were turning on and off, which can be seen in figure 36. The graph shows peaks, which depict the moments the light sources was turned on.

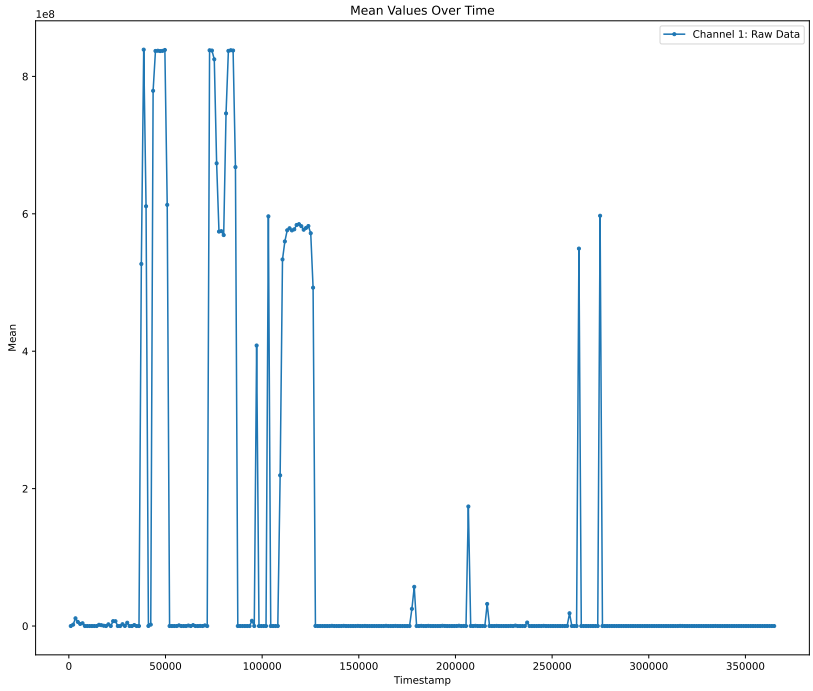

Figure 48: Test 5 conducted, measuring radioactivity with a light source

Another notable test we conducted was with a real radiation source. Below in figure 37, we can see that we can detect a high and relatively consistent radioactivity (minus some errors shown by peaks or drops).

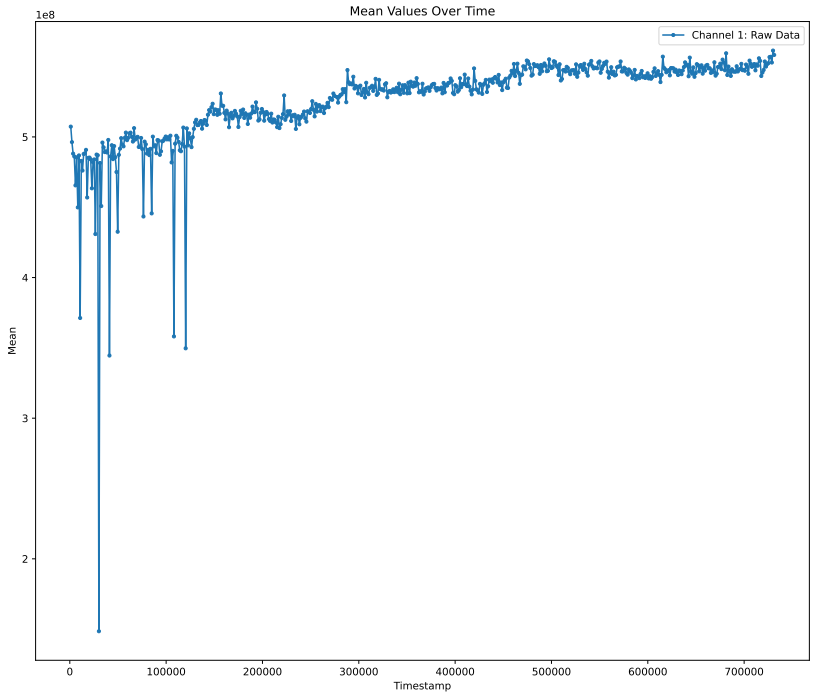

Figure 49: Test 6 conducted, measuring radioactivity with a real radioactive source

5.3 Hardware Tests

In order to validate the hardware design as well as the printed circuit board manufacturing and assembly, each portion of the hardware were tested individually.

- Continuity Testing: This is the first test performed once the main PCB is assembled. It consists of using a multimeter's continuity tester to check all sensitive nodes on the board and ensure the solder joints were done properly. It allows us to validate that all the components are soldered properly and that there is no open circuit in the signal flow. While doing the continuity checks, we are also probing to detect possible short circuits across the board. By example, we probed the input power lines and the ground lines to make sure they are not shorted together.

- SiPM Assembly Testing: This test is to ensure the SiPMs were soldered properly to the SiPM PCB. Using a multimeter's ohmmeter feature, we measure the resistance across the input bias pin and the output signal pin of the SiPM connector. The resistance between the input bias pin and the output signal pin should be finite (in the range of mega ohms) while the resistance across any other SiPM pins should be near infinite (open circuit). While performing this test, the team actually found out that one of the SiPMs was soldered onto the board with the wrong orientation and

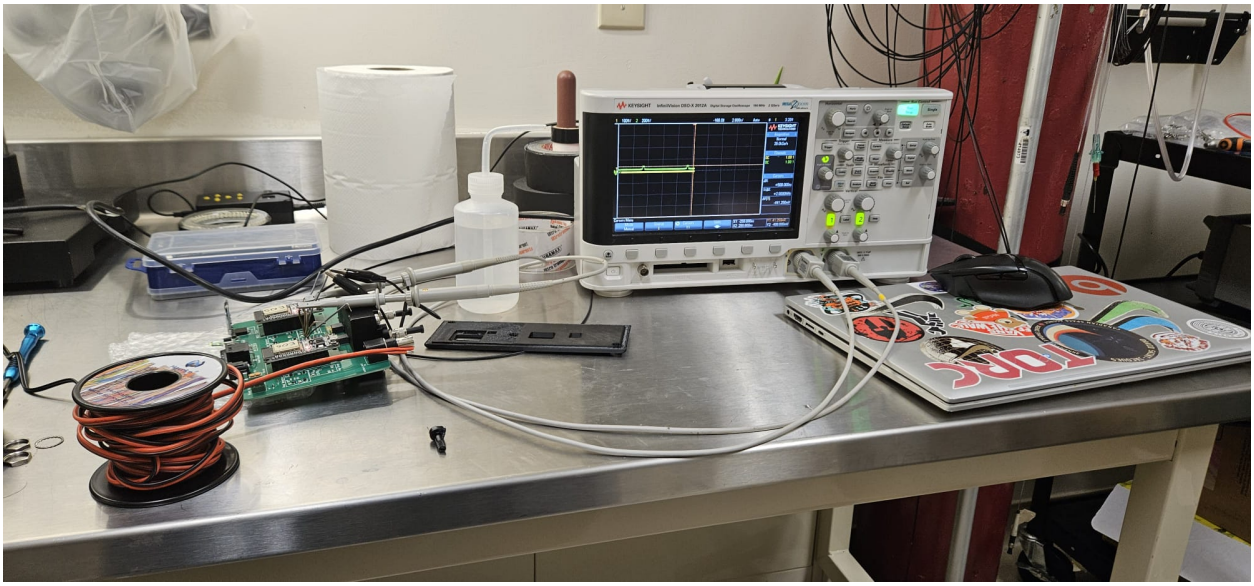

Figure 50: Test Setup for Amplifier Stage and Peak Detection Stage Validation Testing

the pins were flipped. We were then able to correct the board by re-soldering the SiPM in its correct orientation.

- Power Management Testing: Once the manufacturing and the assembly of the board has been verified and validated, we power on the board and probe the main power line, the shared power lines and the different input power pins of the components to ensure they are at the right voltage level.
- Amplifier Stage Testing: The main PCB was designed for testing by having test points at the different stages of the signal processing circuitry. When connecting these test points to an oscilloscope, we can see the signal at that stage. In order to test the proper functioning of the amplifier, a known signal was injected into the board's input using a signal generator and the output of the amplifier stage was connected to an oscilloscope. The test setup enabled us to validate the gain and the noise sensitivity of the amplifier circuit. The amplifier stage design was proofed.
- Peak Detection Stage Testing: The peak detection stage was validated using a similar test setup as the amplifier stage testing described above. The output of the peak detection stage was connected to the oscilloscope and the hold period, the signal fall time and gain of the peak detector signal was observed. The peak detection stage design was proofed and the board was ready for basic functionality testing.

5.4 Integration Tests

A number of tests were conducted to validate the communication and the integration of the software and firmware subsystems.

- Data Transmission Testing: To verify that the firmware correctly sends data to the software, we ran the firmware on the ESP32, ensured the device is connected to the software, and observed whether data (e.g., presence and data packets) is received and processed correctly.
- Command Transmission Testing: To ensure that commands from the software are correctly received and executed by the firmware, we sent commands (e.g., start/stop, integration period adjustments) from the software and monitored the ESP32's behavior and response to ensure the commands are correctly implemented.
- Data Integrity Testing: To ensure data integrity between firmware and software, we sent known data from the firmware to the software and verified that the received data (e.g., timestamps, MAC addresses, mean values) matches what was sent without corruption or loss.
- Multiple Channels Testing: To verify that the system can handle multiple devices simultaneously, we connected two channels to the software and ensured that all devices are able to communicate, send data, and receive commands without conflicts or performance degradation.
- Basic functionality testing using an LED: This tests consists of testing the hardware, firmware and software of the system simultaneously by shining light pulse coming from an LED onto the SiPM interface. Since the detector is dual readout with a leader input and a follower input for the light, both readouts were tested individually by disabling the dual readout feature in the firmware. This was also to test the SiPM board individually when a light is shined on them. The detector is placed in a blacked out box with a LED placed in front of the SiPM interface light input port. This was done so no external or ambient light could interfere with test results. The test revealed initial solder joint issue with the SiPM board, but worked fine once that issue was fixed and the SiPM was re-soldered. The team also noticed that the detector was sensitive to vibration when the main PCB was operating outside of its enclosure. The sensitivity to vibration was almost null once the board was placed inside its enclosure, so no further investigation was done.
- Full functionality Testing using an LED: The full functionality test is essentially the same as the basic one but now with the dual readout enabled. This allows us to verify that the firmware for dual readout works properly and that both the leader and follower readouts interact as expected.
- Full functionality testing using a radioactive source: This test now uses an actual source of radiation and the Enger Lab's scintillating fiber device to simulate more realistic scenarios of user cases. The test was not conclusive because the software app was not able to display any activity. The detector currently works with high light pulse like the ones from an LED, but it might not be able to detector weaker light pulse from a low activity radiation source. More testing with a more active radiation source will need to be done to reach a conclusion about the possible issue.

Figure 51: Test Setup for Functionality Testing Inside the Black Box

6 Next Steps

6.1 Firmware

The firmware is mostly complete, but may undergo a few minor changes in the integration algorithm if improvements in accuracy and dynamic range follow.

6.2 Software

Calibration Function

The scintillation detector system requires precise calibration to ensure accurate measurements of different isotopes. The calibration process consists of two essential phases: comparative testing and calibration function development.

The comparative testing phase will involve operating our detector alongside a pre-calibrated reference detector. Both detectors will measure multiple isotope samples simultaneously to gather comparative data. Measurements will be taken across various integration periods to ensure consistency in the detector's response. Through statistical analysis of these parallel measurements, we will establish the correlation between our detector's output and the reference measurements.

Following the comparative analysis, we will develop specific calibration functions for each isotope (1-4). These functions will transform the raw detector output into standardized units that match established measurements. The current placeholder functions in the code, which simply multiply the output by 5 and add 4, will be replaced with empirically derived transformations. These calibration curves will account for several factors: the detector's specific response characteristics, compensation for background radiation, temperature sensitivity adjustments, and energy-dependent detection efficiency.

Data Smoothing

After completing the calibration phase, the software will incorporate signal smoothing techniques to improve data quality and readability. The signal processing improvements will focus on implementing sophisticated data smoothing algorithms while maintaining the system's real-time processing capabilities.

The software will employ either moving average or Savitzky-Golay filtering techniques for data smoothing. These methods will be implemented to reduce noise during data acquisition while preserving significant peak characteristics in the measurements. The processing parameters will be carefully tuned to maintain an optimal balance between noise reduction and signal preservation.

The smoothing algorithm will feature adjustable parameters based on the integration period and specific isotope characteristics. This flexibility ensures that the processing remains effective across different measurement scenarios while maintaining system responsiveness. The implementation will allow for real-time processing of incoming data, ensuring that the system remains practical for continuous monitoring applications.

6.3 Hardware

The detector currently doesn't work when using the radioactive source available at the EngerLab. The first hypothesis is that the pulses generated by those sources are low and undetectable by the detector. The next steps consists of studying the behaviour of the signal processing circuitry when the amplitude of the input pulses are low (less than 10mV). Both readouts would need to be tested individually. This can be done using the same test setup used for validating the amplifying and peak detection stages. Depending on the results of that testing, we can explore adjusting the input voltage threshold of the detector and/or redesigning the amplifying and peak detection stage to allow great input voltage range.

7 Impact on Society and the Environment

This project's development of a modular, multi-channel, cost-efficient scintillation detector for positron emission tomography (PET) scans carries profound implications for both society and the environment. By examining both the positive and negative impacts, this analysis aims to provide a comprehensive understanding of the broader consequences of implementing this technology.

7.1 Environmental Impacts

- **Use of Non-Renewable Resources:** The manufacturing of scintillation detectors requires metals and rare earth elements, many of which are non-renewable. Although the modular design promotes reusability and potentially extends the lifespan of the detectors, the dependence on such materials necessitates robust recycling programs and waste minimization strategies to mitigate environmental impact.

- **Comparative Environmental Benefits:** These detectors are designed to be more compact and energy-efficient than traditional PET system technologies, which typically consume more materials and energy. This efficiency translates into a reduced carbon footprint during both production and operational phases. Additionally, the modular nature of the design means reduced electronic waste, as components can be individually replaced or upgraded without discarding the entire system.

7.2 Societal Impacts

- **Safety and Risks:** Safety for both developers and users is a critical consideration. The low-voltage design minimizes electrical hazards, enhancing user safety. Nevertheless, the handling, usage, and disposal of the chemical and electronic components used in these detectors must strictly adhere to health and safety regulations to prevent exposure to hazardous materials.

- **Societal Benefits:** The primary advantage of this technology is the enhanced accessibility of PET scans, which are crucial for early diagnosis and treatment of many diseases. By reducing the cost and increasing the adaptability of PET technology, diagnostic services become accessible to a broader population, improving public health outcomes. Economically, the reduced cost of technology can lower healthcare costs significantly, enabling better allocation of medical resources.

This project, while presenting certain environmental challenges, primarily related to the consumption of non-renewable resources, offers significant societal benefits that include improved healthcare access and economic growth, thereby potentially enhancing the quality of life and contributing to sustainable development.

8 Report on Teamwork

8.1 Roles and Responsibilities

- **Chris Hatoum:** Developer, Software-Firmware Integration Tester, Integration Tester
- **Habib Jarweh:** Software Developer, Software-Firmware Integration Tester, Integration Tester
- **Philippe Olivier Fils:** Circuit Simulator, Main PCB Designer, Assembler, Hardware Tester, Integration Tester
- **Tofic Esses:** Firmware Developer, Circuit Simulator, SiPM PCB Designer, Enclosure Designer, Hardware Tester, Integration Tester

8.2 Group Collaboration

The group mainly communicated using Whatsapp. External collaboration with advisors was done through Outlook. Each member of the team became experts in their roles, but still required to understand other systems at a higher level for integrations. Each member would take on as many responsibilities as possible. Available time and semester workload were the limiting factors.

8.3 Challenges and Overcomings

The main challenge our team faced was handling the dependencies of the assigned tasks. For example, the firmware and software application require careful planning and integration. It was said at the beginning of the design phase that RXTX-Computer communication would be handled using Wi-Fi.

Tofic Esses was developing the firmware, and Habib Jarweh was developing the software application, both having Wi-Fi in mind. However, when it came time to do the integration, many technical challenges were faced. As a result, Habib Jarweh lost a significant amount of development time by developing a solution that would work with Wi-Fi.

Reflecting on this, it would have been better to test ESP32-Computer Wi-Fi communication before writing a lot of code.

9 Conclusion

Throughout this semester, significant progress has been made in the development of a modular, multi-channel, cost-efficient scintillation detector for use in the automated synthesis of PET radiopharmaceuticals. Key accomplishments include the completion of the firmware, the finalization of the signal processing and power circuitry designs, the development of the RXTX PCB, and the initiation of software and hardware integration tests. These achievements have laid a solid foundation for the continued advancement of the project.

The firmware development, while complex, has reached a level of maturity where only minor adjustments are anticipated moving forward. Our use of the ESP32 and ESP-NOW protocol has been validated through extensive tests, demonstrating the feasibility of real-time data collection and communication across multiple channels. Additionally, the signal processing simulations conducted in LTSpice have proven that the designed circuits meet the required specifications for peak detection and amplification, ensuring the integrity of the SiPM signals.

On the hardware side, the designs for the main PCB and SiPM boards are near completion. The enclosure designs have also begun, ensuring that the final product will not only perform efficiently but also maintain a user-friendly form factor. The software application has successfully demonstrated its ability to interface with the firmware, providing real-time data visualization and control over multiple channels.

Moving forward, the focus will shift towards finalizing hardware manufacturing, completing enclosure designs, and conducting full system integration tests. This includes ensuring the smooth operation of the entire system, from SiPM signal detection to data visualization on the software application. Additionally, efforts will be made to implement features such as calibration and curve fitting in the software, enabling precise radioactivity measurements.

The insights gained during this semester highlight the importance of early and continuous integration testing to mitigate potential communication challenges between subsystems. Furthermore, the need for a modular approach has proven beneficial in both system design and troubleshooting, allowing us to address issues at the component level without compromising the entire system.

As we move into the final stages of development, we are confident that the project is on track to achieve its primary goals of reducing costs, improving accessibility to PET technology, and providing a scalable solution for various medical imaging applications.

References

[1] H. S. H. Ahn, "Development of a small, cost-efficient scintillation detector for use in automated synthesis of PET radiopharmaceuticals," Master's thesis, McGill University, Medical Physics Unit, Montréal, Québec, Canada, August 2023, a thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Medical Radiation Physics.

[2] Physics Open Lab, "Front-end electronics for sipm," https://physicsopenlab.org/2017/11/28/front-end-electronics-for-sipm/, 2017, accessed: 2024-09.

[3] Cosmic Watch, "Cosmic watch: The desktop muon detector," http://www.cosmicwatch.lns.mit.edu/, 2024, accessed: 2024-01 - 2024-09.