CosmicWatch:

The Desktop Muon Detector

June 12, 2018

The CosmicWatch Desktop Muon Detector is an MIT based undergraduate-level physics project that incorporates various aspects of electronics-shop technical development. The desktop muon detector is a self-contained apparatus that employs plastic scintillator as a detection medium and a silicon photomultiplier for light collection. These detectors can be battery powered and used in conjunction with the provided software to make interesting physics measurements. This paper descirbes an updated version of the detector, previously introduced in *https://arxiv.org/pdf/1606.01196.pdf*.

Authors: Spencer N. Axani (saxani@mit.edu) and Katarzyna Frankiewicz (katarzyna.frankiewicz@ncbj.gov.pl)

Contents

1 Document Overview

CosmicWatch is an outreach program that is being designed to provide students the ability to build their own in-expensive muon detector. A single detector costs approximately 100$. Each detector takes a novice high-school student approximately 4-hours to build for the first time and a second detector can be built in about an hour. Details regarding our project can be found on our website:

`www.cosmicwatch.lns.mit.edu`

All suplementary marerials are available in the GitHub repository:

`https://github.com/spenceraxani/CosmicWatch-Desktop-Muon-Detector`

The CosmicWatch Desktop Muon Detector consists of a 5 cm$\times$5 cm$\times$1 cm slab of solid plastic scintillator instrumented with a silicon photomultiplier (SiPM) to detect scintillation light emitted from charged particles as they pass through the scintillator. The signal from the SiPM is sent through a custom designed printed circuit board (PCB) which shapes the signal such that a micro-controller can measure the time and amplitude of the SiPM signal. We use an Arduino Nano to measure the pulse amplitude and record the count number, time of the event, pulse amplitude, and detector dead time. The threshold for a signal from the SiPM to trigger the data acquisition can be tuned in the provided Arduino software. The detector can be powered by a mini-USB to USB connector.

The information from the detector can be readout to a computer via a USB connection using the provided python software or plotted in real-time on our website: www.cosmicwatch.lns.mit.edu. The OLED screen on the front of the detector updates every second with the count number, count rate, and a bar indicating the amplitude of the most recent event. An LED flashes every time an event is registered. The detector can also be connected to an oscilloscope through the BNC header on the back of the detector to view the raw SiPM pulse.

This document serves as an instruction manual for the Desktop Muon Detector. The first part of the document is used to go through the construction while outlining an potential dificulties students may encouter. We then elaboarate on how the electronics in the detector were designed, the software for operating the detector and recording data, and provide a few example measurements. The final section can be used for troubleshooting.

Figure 1: An array of CosmicWatch Desktop Muon Detectors.

2 Condensed instructions

This section provides a concise list of recommended steps to follow. A detailed description of each step is provided in the subsequent sections.

1. Purchase all components listed in the "Purchasing_List.xl".
2. Upload the code to the Arduino Nano.
3. Populate all components on Main PCB.
4. Test HV pin on Main PCB to ensure it is delivering roughly +29.5 V.
5. Populate SiPM PCB.
6. Machine scintillator to 5×5×1 cm. Drill 4 mounting holes for the SiPM PCB and heat polish machined surfaces.
7. Wrap plastic scintillator in reflective foil.
8. Put optical gel on SiPM and screw SiPM PCB into the plastic scintillator.
9. Wrap the plastic scintillator and SiPM PCB with black electrical tape.
10. Plug the SiPM PCB into the Main PCB. The OLED screen will read out the rate of triggers and the LED will flash every time the detector triggers.
11. If there is a problem, see troubleshooting section (Section 7) of this document.

3 Instructions for building your first detector

This section contains a detailed description of the construction of one of the CosmicWatch Desktop Muon Detectors. Here, we try to elaborate on potential issues, and provide as much information as possible without going into the physics of the device. For a more in-depth description of the components and physics, see our pre-print with previous version of the detector:

`https://arxiv.org/pdf/1606.01196.pdf`

.

3.1 Purchasing the components

First step is to purchase the required equipment. In the GitHub repository we've provided an excel spreadsheet which contains a list of the components used in the project. If you are building a single detector, we still recommend purchasing extras of the small surface mount resistors and capacitors.

1. Send the PCB Gerber files, contained in the zipped PCB file, to a PCB manufacturer. We use Elecrow.com [8], from which you can order a minimum of 5 PCBs for approximately 5 USD. We found that expedite shipping takes roughly a week. If you do not choose expedite shipping, it takes about a month.

2. Elecrow.com can also laser cut acrylic. We use this service for the faceplates of the case. Optionally, one could 3d print the faceplates using the Endplates files in the "Enclosure_Files" folder. In order to laser cutting the faceplates, simply upload the "EndPlates_for_laser_cutting.dxf.zip" file to the website and choose the dimensions to be $10\times15\times2.5$ mm.

3. The aluminium enclosure (2506-2.9 inch anoodized aluminium, without faceplates) can be purchased from enclosuresandcasesinc.com [9] by providing them with the top and bottom silkscreens found in the "Enclosure_Files" folder.

4. The electronic components found in the excel file can all typically be found at either Digikey.com [6], Ebay.com [7], or Amazon.com [4]. The spreadsheet provides a link to where we purchase our components from.

5. Arduino Nanos can be found at many local electronic shops but we prefer to purchase large orders from Ebay where they can be found for under 2.5 USD.

6. The SiPM can be purchased from SensL [10]. We use the SensL 60035 SMT C-series. It has a photocathode area of 6 mm$\times$6 mm. The documentation can be found in Ref. [2]. The price per SiPM drops significantly once you place an order of x100 or more. It is recommended to look for other groups who are making large SiPM orders and work

with them. We've found many physics professors and research scientists that work with photomultipliers and they may be able to help you.

7. There are several companies that sell plastic scintillator, but we found that physics departments often have left over scintillator from previous experiments. Ebay and other online shops often sell used scintillator as well. The SiPM we use (the SensL) is most sensitive to 480 nm light, therefore the scintillator should be chosen to emit near that frequency with the highest photon yield per MeV. We've found all of our scintillator in various physics departments.

3.2 Uploading code to Arduino Nano

Before soldering the header pins onto the Arduino Nano we should upload the code to the Arduino to make sure it works. We first must install the Arduino Integrated Development Environment (IDE), this will allow us to modify and upload the code to run the Arduino. The Arduino IDE can be found in Ref. [5].

If you are using a Mac operating system, you must also install the CH340g driver. This driver can be installed from 'brew' or from downloading the package from Ref. [3].

Now, lauch the Arduino IDE and install the required libraries for the code. Libraries can be installed directly in the IDE by clicking Sketch→Include Library→Manage Libraries. From here, install the following libraries:

1. Adafruit_SSD1306
2. Adafruit GFX
3. TimerOne

Finally, we need to select that we are using an Arduino Nano, and tell the IDE which USB port it is plugged into. Click Tool→Board→Arduino Nano and also select the USB port from Tool→Port.

We can now upload the Arduino code. See the Section 7 if you are having problems.

3.3 Populating the main PCB

There are two PCBs which must be populated.The smaller PCB holds the SiPM and some filtering electronics for biasing the SiPM. The larger of the two is the main PCB (shown in Fig. 2) and contains the data acquisition electronics. It was designed so that the surface mount devices (SMD), like resistors and capacitors, are on the bottom of the board and all the large components are on the top (except for the 4-pin connector for the OLED screen, which is on the bottom). The bottom components perform all the signal processing and power regulation. If you are unfamiliar with using surface mount technology (SMT), check out an online video on YouTube. We use 0805 SMT components, meaning that the resistors and capacitors measure 0.08 inches by 0.05 inches. These are small, but easily manipulated with a good pair of tweezers.

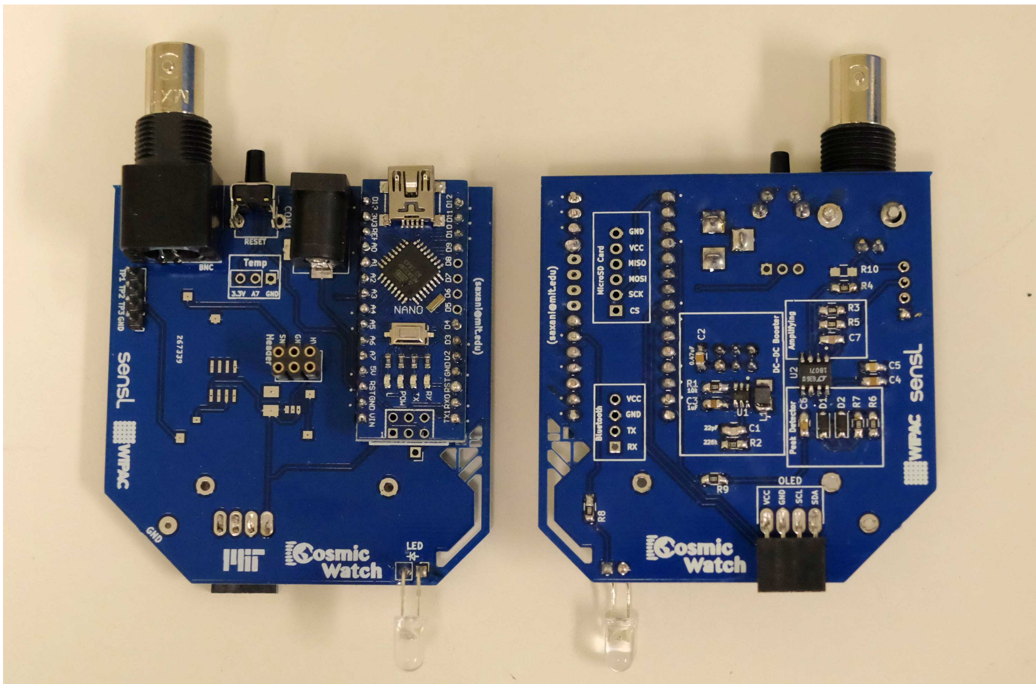


Figure 2: The top and bottom of populated main PCB.

The top side of the board contains the larger components. The alignment and position of each component is shown on the silkscreen. The 6-pin header is used to connect the SiPM PCB to the main PCB. The reset connection is used to mount the reset button. When pushed, this will reload the Arduino Code. There is also a DC barrel jack that that you could power the device through a 5 V wall adapter. This connection is optional, since the detector is typically powered through a USB connection. The two LED pins are used to mount the LED, however it should be noted that this connection has a direction, as indicated on the silkscreen. The OLED screen 4-pin connector should be mounted on the bottom of the board, as shown in the image. The Arduino Nano header pins should be inserted into the PCB before soldering them to the Arduino, since this will help ensure that they are properly aligned.

The final connection on the top of the main PCB is the BNC connector. It is connected directly to the SiPM output (pin 1). Therefore, can be pluged into an oscilloscope to see the raw SiPM pulses, or used as a trigger for other devices. This also serves another purpose: if the SiPM PCB is not plugged into the main PCB, we can inject pulses into the BNC connection. This is useful for calibrating the circuit. A further description of how we calibrate the electronics can be found in Section 4.2.

After the main PCB has been populated, we can test the supply voltage to SiPM. Check the voltage between the HV pin and the GND pin on the 6-pin header. It should read +29.5V, if it doesn't, check the DC-DC booster circuit on the PCB to make sure you have the correct components. We can also plug the OLED screen into the 4-pin connector. If you were to plug the detector in at this point, you would see the CosmicWatch splash screen and the OLED will begin reading out a negative count rate. This is because the SiPM PCB is not connected. The LED will also be on continuously. If not, check the trouble-shooting section of this document.

3.4 Populating the SiPM PCB

After the main PCB is populated, we can move to the SiPM PCB. The resistors and capacitors on this PCB are used to filter the DC biasing voltage for the SiPM, except for R3, which is simply used to hold the line at ground when there is no signal from the SiPM (called a pull-down resistor). The same side of the PCB that has the SMT components also has a 6-pin connector and two aluminum stand-offs that are used to mount the detector assembly to the main PCB. The top side of the Main PCB is used strictly for the SiPM. We've made a video describing this part, that you can view at this link: www.youtube.com. Fig. 3 shows the fully populated PCBs.

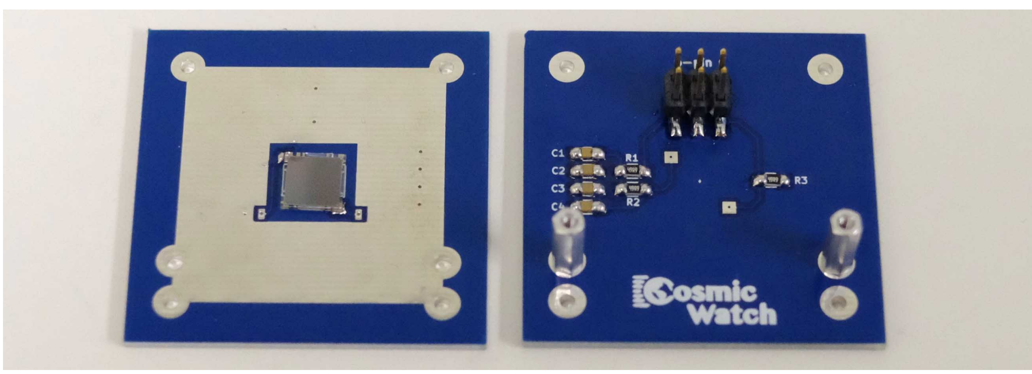


Figure 3: The top (left) and bottom (right) of the SiPM PCB. On the top we can see the SiPM in the middle surrounded by a reflective grounding plane. On the bottom we see the 6-pin connector and two stand-offs used to connect to the main PCB.

3.5 Machining the scintillator

The scintillator slabs may have to be machined to size on a mill and then polished in order to make the faces optically transparent. Polishing the scintillator improves the photon collection efficiency of the SiPM by increasing the optical transparency at the interface between the SiPM and the plastic scintillator. It also promotes total internal reflection on the walls of the scintillator, thus increasing the overall number of photons incident on the SiPM. We also wrap the plastic scintillator in reflective foil (just store-bought tin foil, preferably something a little thicker). We use optical gel to match the refractive indices of the plastic scintillator and the protective layer on the SiPM photocathode to increase efficiency.

We've included a CAD drawing of the machined scintillator in the pictures folder of the supplementary material. You can cut the scintillator to the rough size on a band saw then machine it to specification on a mill using a 1/2 inch end mill running at about 1300 rpm. We've found that we can safely remove 2.5 mm for a standard pass on the mill, and about 1 mm on a climbing pass. The milled surface is relatively optically transparent, but we will improve it with heat polishing. Prior to heat polishing though, we need to drill four holes to mount the SiPM PCB onto the plastic scintillator. Drilling must be performed at very low rpms. On the mill set the spindle to roughly 60 rpm and use a pecking motion to drill 4 holes in a square, 3×3 cm. We use a number 54 drill bit (1.397 mm) which makes a large enough hole that a 5/16" Number 0 screw will self tap into the scintillator. We've found that if we generate too much heat while drilling, the scintillator will crack.

After the scintillator has been fully machined, we can heat polish it to improve the surface transparency. We only heat polish the surfaces that were machined, since the others are already sufficiently transparent. We use a hot air gun from a soldering station, with a low air flow setting at roughly 450 °C. By slowly passing the hot air over the machined surface, the scintillator will melt and re-solidify into an optically transparent surface. Overheating will cause the scintillator to deform or burn. Heat polishing must be the final step. Any machining preformed after the heat polishing will cause the scintillator to crack.

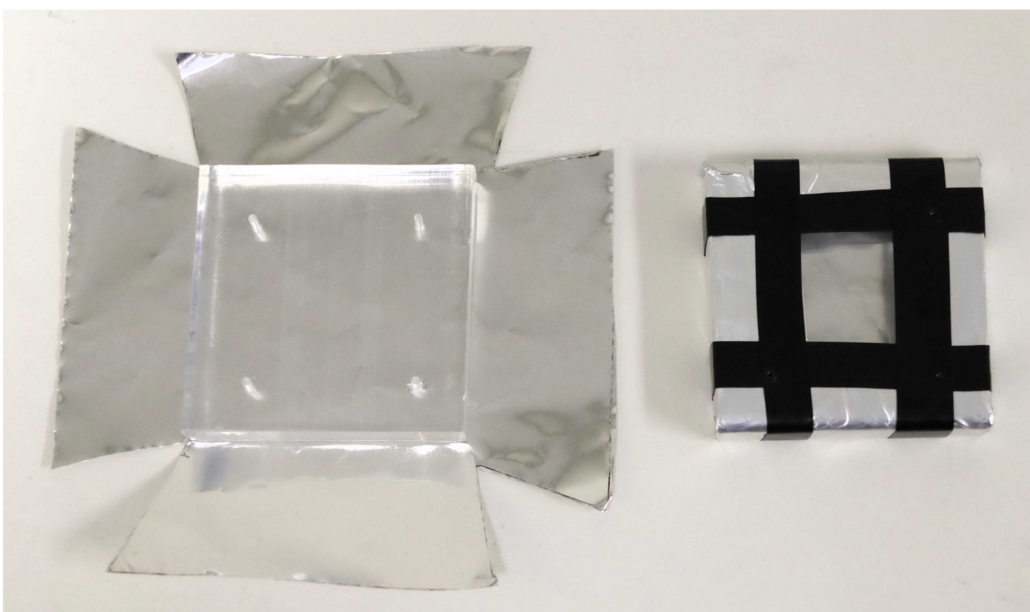


Figure 4: Left: The machined scintillator sitting on top of the reflective foil. Right: the covered scintillator with a 2 ×2 cm opening for the SiPM.

3.6 Mounting the SiPM PCB to the scintillator

The scintillator should be covered in reflective foil to increase the number of photons observed by the SiPM. Leave a 2 × 2 cm open area on the face of the scintillator for the SiPM as shown in the left side of Fig. 4. We hold the reflective foil in place using black electrical tape. The electrical tape should not be stretched, since over time it will lose it's elasticity and peal off the surface. We poke holes through the electrical tape to provide a guide hole for the Number 0 screws. A small amount of optical gel should be placed on the surface of the SiPM before attaching the SiPM PCB to the scintillator. If you don't have optical gel, that's alright, you can use the detector without it, but the photon efficiency will just be slightly lower.

The SiPM PCB is mounted to the plastic scintillator using four 5/16" Number 0 screws. We do not tighten the screws all the way since this would put too much pressure onto the SiPM face. Instead, we leave approximately 1 mm of space between the SiPM PCB and the plastic scintillator. Once the PCB is secured in place, we can wrap it in black electrical tape to make the entire thing light-tight. Do not stretch the electrical tape when wrapping it. The wrapped component can be seen plugged in to he main PCB on the right side of Fig. 5.

3.7 Assembling the Desktop Muon Detector

The two main components (main PCB and the SIPM PCB + plastic scintillator) have now been made and we can begin assembling the detector. Prior to plugging the 6 pin connector from the SiPM PCB into the main PCB, we should check again to ensure that the voltage is appropriate. The 6-pin connector is used to supply the biasing volate from the main PCB to the SiPM, as well as transmit the signal from the SiPM back to the main PCB. The pins on the 6-pin connector are redundent, meaning that the top two are used for the voltage, the middle two are used for ground, and the bottom two are used for the signal. To check the biasing voltage, connect a multi-meter between the HV pin and ground. It should read approximately +29.5 V. The breakdown voltage for these SiPMs is +24.7 V and we supply an overvoltage of roughly 4.7 V. Anything over 30.0 V may cause damage.

Once the voltage has been verified we can plug the SiPM PCB into the main PCB. Depending on many factors, you should be able to see a count rate of about 0.5 - 0.8 cps. Many of these are cosmic ray events, however, part of counts are actually gamma rays from radiogenic backgrounds. In the measurement section below, we describe some characteristics of the detector, as well as how to take a coincidence measurement to reduce backgrounds.

If you would like an electronics enclosure, we have desinged the detector to just fit inside the 2506-2.9 inch aluminium case from Ref. [9]. We do not use the faceplates provided from this company. Instead, we either 3d print the faceplates (printing files found in the supplementary material), or have acrylic pieces laser cut from Elecrow . There are two reasons to use plastic faceplates rather than aluminium ones. First, they are cheaper than aluminium face plates and can be manufactured in bulk with different colors. And secondly, the OLED screen has a four pin connector that perturdes out the front of the screen and can come in contact with the face plate. The files for laser cutting are also provided in the supplementary material.

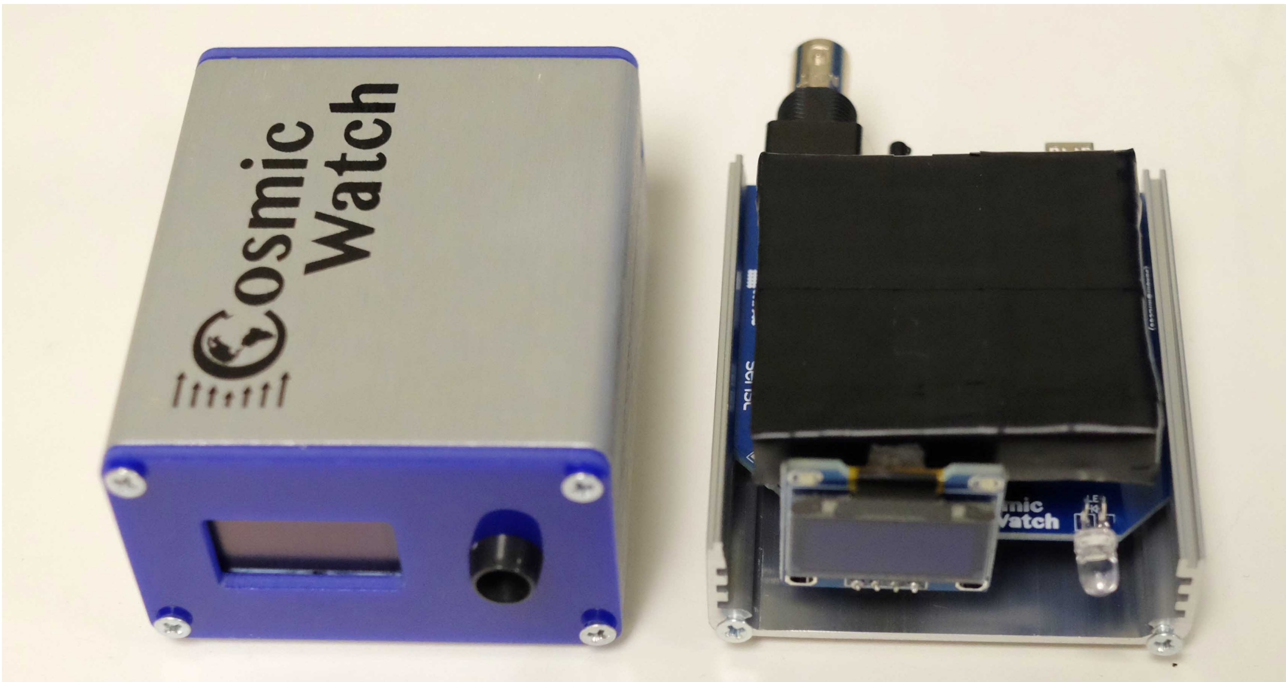


Figure 5: A picture of the complete detector.

4 Electronics Description

This section provides a detailed description of the circuitry and illustrates how we convert between the measured pulse amplitude on the ADC and the SiPM pulse amplitude.

4.1 Description of the circuit

The SiPM PCB is mounted directly onto the plastic scintillator by means of four No. 0 screws to maintain pressure on the SiPM face thus ensuring good optical contact between the photocathode area and the plastic scintillator. The main PCB contains electronics used to amplify and shape the signal from the SiPM such that it can be measured by the microcontroller. It also filters and regulates the voltages used in the detector. The amplification and shaping of the waveform is accomplished using dual rail-to-rail input and output operational amplifier (op amp), whose functions we will describe below. We use an inexpensive 16 MHz Arduino Nano ATmega328 as a microcontroller to read the data out to a 0.96-inch OLED screen and through a mini-USB cable to a computer. The code necessary to run the Arduino as well as a list of the required libraries (which all can be installed in the Arduino IDE) are provided in the supplementary material and described in the subsequent section. We also provide a Python script to log the data to computer, and a program that will connect the desktop muon counter to our website to record and plot data in real time.

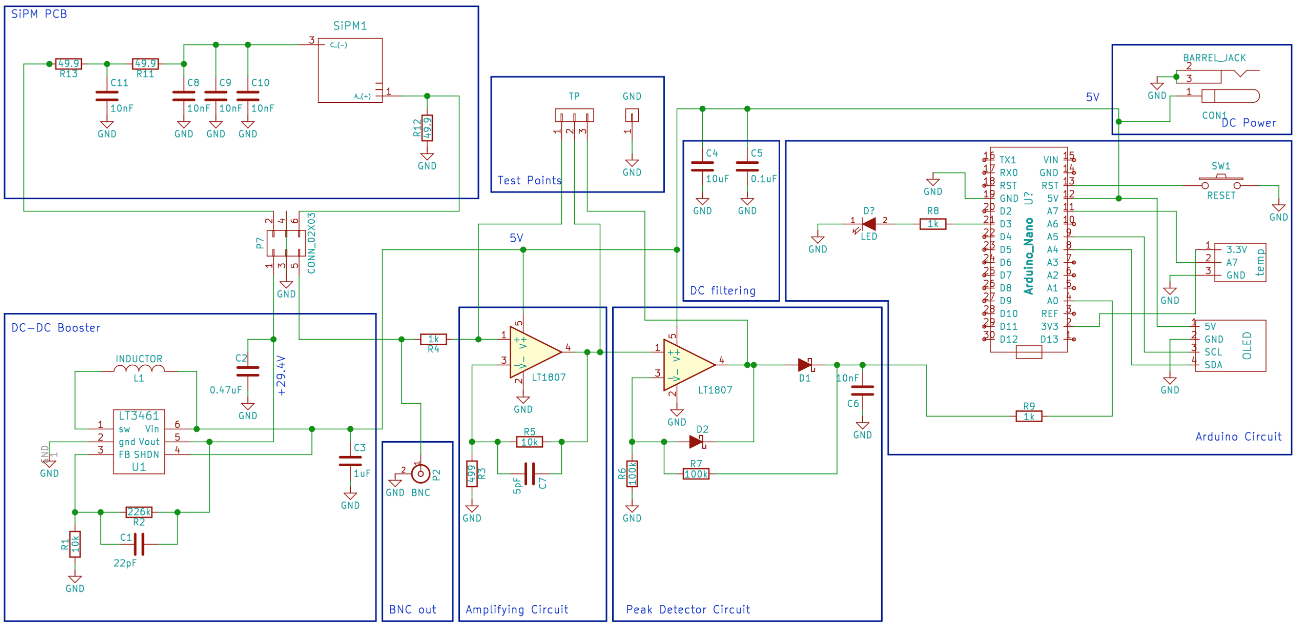


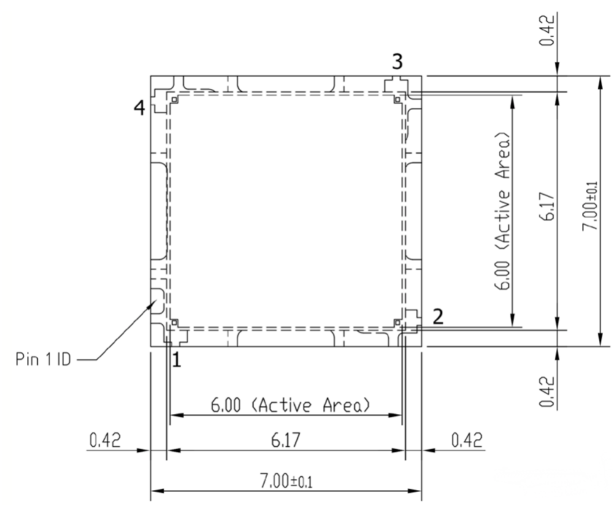
Figure 6: The circuit diagram for the detector. We have broken the circuit into sections, outlined in blue, which are individually described in the text.

4.1.1 DC-DC Booster

The DC-DC Booster is used to take the 5 V DC supplied by the USB connection and transform it up to +29.5 V to bias the SiPM. If we look at the bottom-side of the main PCB (see Fig. 2), we see that the silkscreen divides the circuit into multiple sections. The DC-DC Booster is located near the center. The +29.5 V is labeled on the top side of the main PCB as “HV” on the 6-pin header. Here, you also see the a ground connection “GND”, and the signal connection from the SiPM labelled “SNG”. Each connection uses two pins from the 6-pin connector – they are redundant in case a pin breaks and to provide extra stability. The DC-DC booster was designed using the Linear Technology integrated circuit, LT3461A. The documentation for this device can be found in Ref. [1] or in the datasheets folder in the supplementary material (specifically look at image on Page 10).

4.1.2 SiPM circuit

The SiPM circuit on the top left of Fig. 6 contains several resistors and capacitors connected to pin 3 of the SiPM. These are used to filter out high frequency noise in the supply voltage (know as a high pass filter). The SiPM pins can bee seen below in Fig. 7. The Fast Output pin is not used and left floating, along with pin 4 and 5 (located at the bottom side). Pin 1 is identified by the extra leg sticking out on the side (all corners are symmetric except for pin 1). We also see that there is a 49.9 Ω resistor on the anode (pin 1) of the SiPM. This resistor is known as a pull-down resistor and is used to hold the line at ground when there is no signal from the SiPM. When a micro-cell in the SiPM discharges, a small amount of current flows from pin 3 to pin 1. The voltage associated with a single discharge is going to be on the order of a few mili volts. When a muon passes through the scinitllator, we typically see a few dozen photons. The “BNC out” connection, shown in the middle of Fig. 6, can be connected to an oscilloscope to observe the raw pulse from the SiPM. This is useful if you would like to use the Desktop Muon Detector as a trigger for another experiment. The test point connection, TP1, shown at the top middle of Fig. 6, can be connected to an oscilloscope to see the SiPM pulse after it passes through the 1kΩ resistor, R4.



Pin Assignments	
Pin #	**B-Series**
1	Anode
2	Fast Output (not used)
3	Cathode
4	No Connect
5	No Connect

Figure 7: The description of SiPM pins (taken from `http://sensl.com/downloads/ds/DSMicroCseries.pdf.`). Pin 5 is not visible; it is the center pad on the other side of the SiPM.

4.1.3 Amplifying circuit

This circuit takes the positive signal from the SiPM and amplifies it by about a factor of 20. Hoever, we limit the high-frequency amplification using a 5 pF capacitor (C7) to improve ADC response. The effective amplification is roughly a factor of 10. We can look at the amplified signal by connecting the test point, TP2, to an oscilloscope. The amplified signal is then sent to the peak detector circuit.

4.1.4 Peak detector

The purpose of the peak detector circuit is to detect the amplitude of the amplified pulse and hold the voltage at that level for a sufficient time, such that the Arduino can measure it, then decay and wait for the next pulse. Once a pulse from the amplifying circuit enters the non-inverting input of the op amp (+), the Schottky diode D1 becomes forward-biased and allows the op amp to charge the sampling capacitor C6. While charging, there is an unavoidable leakage current through the resistors R7 and R6 to ground. However, these resistors were chosen to be large enough so that this is negligible. When the pulse from the amplifying circuit subsides, D1 becomes back-biased and forces C1 to discharge through R7. The current will then flow to ground via two different paths depending on the voltage on C6.

If there is a large voltage on C6 (greater than the forward voltage drop on D2), D2 becomes forward-biased and will allow current to flow to the output of the op amp, which is now sitting at the negative rail, in our case it is ground. The decay time associated with this is then R7×C6. If the voltage on C6 is smaller than the forward voltage drop on D2, the diode will be back-biased and current will flow through the series of resistors R6 and R7. The decay constant associated with this is (R6+R7)×C6. This bifurcation was found to greatly improve the response of the circuit to very small and very large incoming pulses. Given that R6 and R7 are 100kΩ and C6 is 10 nF, we expect a decay time of roughly 2 ms.

Since the output of the op amp can only be driven to 4.78 V (with 5 V supplied to the positive rail) and the voltage drop across D1 is approximately 0.4 V, the maximum output voltage now becomes approximately 4.28 V. We have specifically chosen the diodes to minimize the forward voltage drop, thus allowing us to measure a higher possible voltage.

4.1.5 Arduino

The Arduino clock is 16 MHz, but a single ADC measurement takes about 90 clock cycles, therefore we are only able to sample the waveform at roughly 172 kHz. This is actually much quicker than the default Arduino analog sampling time. The way we accomplish this is to use what is called a “prescaler”. With this, we are able to make a measurement of the pulse every 5.8 μs. Since the peak dectector pulse lenght is much longer than this, the first three measurements that we use appear roughly constant.

The Arduino Nano code can be found in the Arduino folder in the supplementary material. The Arduino is used to perform several tasks:

1. Set the trigger threshold on the ADC.
2. Measure the pulse amplitude from the peak detector circuit.
3. Convert the pulse amplitude to a SiPM pulse amplitude.
4. Record the time of the event and dead time between events.
5. Control the OLED screen and LED light.
6. Send the data through USB to a computer.

The uncertainty on the time of the trigger pulse is roughly 4 μs due to the limited sampling speed of the Arduino Nano. When you are recording the data on the computer, there is an added uncertainty on the trigger of roughly 0.1 ms due to the limited speed of the serial communication (at a baud rate of 115200 bits per second).

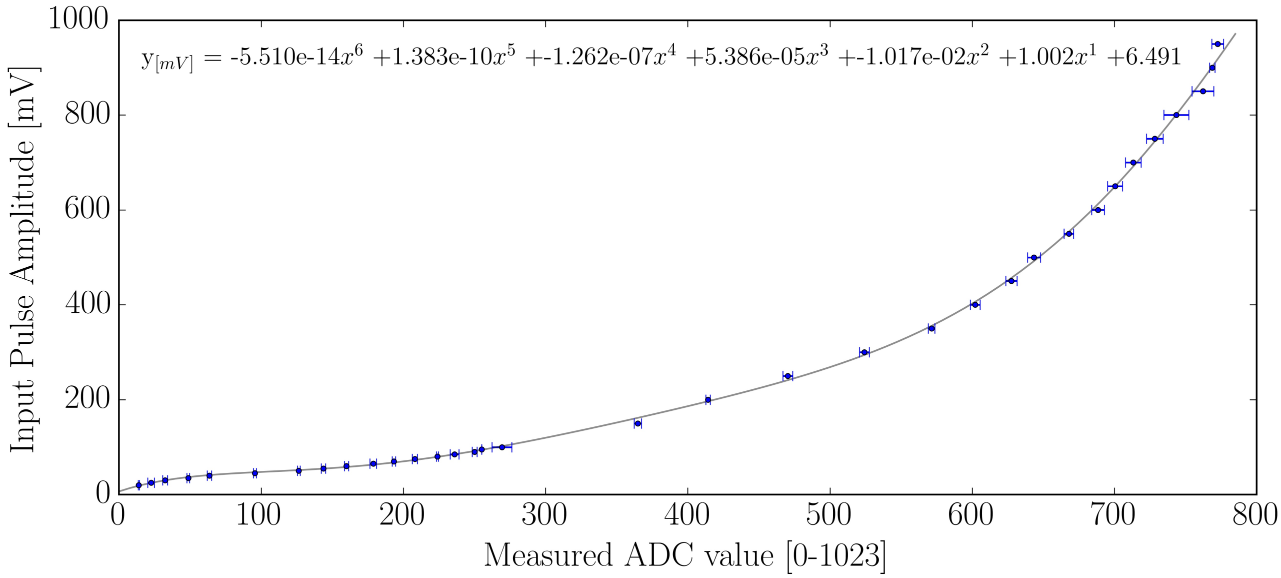
The Arduino is used to update the OLED screen every second. In the code, we see that the program is interrupted every 1,000,000 μs to update the clock using the getTime() function. The screen shows the total running time, the number of counts, and the count rate (with the dead time taken into account). The dead time is calculated by measuring the time it takes the Arduino to perform each operation, then summing that time together and subtracting it from the total time.

The Arduino waits for a signal greater than the trigger threshold on the analog pin A0, then begins data acquisition. We do not record the triggering measurement since it may have been made during the rise time of the pulse. Instead, we take the subsequent three measurements and average over them. This is the value recorded as the pulse amplitude in ADC counts from 0 to 1023 ($2^{10} = 1024$ values or 10-bit measurement).

The current version of the code requires 60% of the storage space and 49% of the dynamic memory. The OLED libraries require the majority of the resources. Adding extra functionality to the code must be such that the storage space and available dynamic memory remains bellow 70%. This means that if there is a particular functionality, like adding an SD card reader, the code might need to either be further optimized, or the OLED code might need to be removed.

4.2 Calibrating the electronics

To determine the correlation between a measured ADC value from the peak detector circuit and the actual SiPM pulse amplitude, we begin with taking a screenshot of the SiPM output waveform, digitize it on a computer, and then program it into a waveform generator. The waveform generator allows us to scale the voltage and inject it into the circuit through the BNC connection (without the SiPM PCB connected). The plot below shows the relationship between the amplitude of the injected pulse and the measured ADC value. In this image, at each point, we inject 200 pulses then calculate the standard deviation of the measured ADC values. We then fit the distribution according to a 6^{th} order polynomial. The result of the fit and the data is shown in Fig. 8. We use these results in the function "Calibration_fit" in the Arduino code.



s

Figure 8: This plot shows the measured ADC value for an injected pulse of known amplitude. The circle data point are the averaged ADC value. The blue line is the best fit according to a $6^t h$ order polynomial. The values of the fit are shown at the top of the plot.

5 Recording data

In order to record data from the detector directly to the computer, you have to plug the detector into a USB port.

- If you are Mac user, you will need to install the CH340g driver to read from serial port (see Ref. [3]).
- If you are Linux user, you may need to change the permissions for the USB port (in terminal, type `> sudo chmod 666 /dev/ttyUSB*`).

Data from the detector can be recorded using python scripts provided in Recording Data folder, in the GitHub repository (scripts are prepared for python 2.7). You can run python script import_data.py or use corresponding executable program provided for Windows, Linux, or Mac OS platform. If you running python script you need to have pyserial module installed, if you are using executable version, you don't need to install any additional software on your computer.

When you run either the executable program or the python script, you will see the list the available ports on your machine. You need to select the port where the detector is plugged into. Then you enter a file name, where the data will be saved (Mac users may need to also enter the directory location). Finally, the program asks for a "Device ID." This is simply for you to keep track of which detector was used to collect the data. Whatever you enter here will be added to the header of the saved file.

Example data file is shown in Fig. 9. The header describes the content of each column. The first and second column is a data and time stamp provided by the computer. This is accurate to a few miliseconds due to the limited transfer speed of the serial port. Column four is also a time stamp, given in miliseconds, by the Arduino Nano. We save two versions of the time stamps because the Arduino has a clock that is only accurate to roughly ± 50 ppm, meaning that over the course of a day, we would expect to see a discrepancy between two detectors of a few seconds. The time stamp from the computer is much more accurate however less precise. This is due to the transfer speed of the serial port. To make a software coincidence measurement, we suggest using the computer time with a time window of about 10 ms. The third column in Fig. 9 represents the count rate. When data from a detector is being recorded and the reset button is pushed, the count number resets to one. The fifth column is the measured amplitude of the ADC. This is a value from 0 to 1023, although in practice, due to how the electronics were designed, the effective range is more-or-less from 0 to 800. The sixth column is the calculated SiPM pulse amplitude – this is calculated through the calibration described in Section 4.2. And finally, the last column is the dead time measured since the previous pulse. Dead time must be taken into account for all rate measurements.[1].

```
######################################################################
### Desktop Muon Detector
### Questions? saxani@mit.edu
### Comp_time Counts Ardn_time[ms] Amplitude[mV] SiPM[mV] Deadtime[ms]
### Device ID: 1
######################################################################
2017-07-14 14:01:16.657004 1 112 35.00 31.25 19
2017-07-14 14:01:23.202956 2 6653 96.67 47.14 45
2017-07-14 14:01:31.789663 3 15233 79.00 44.28 145
2017-07-14 14:01:32.171374 4 15613 103.00 48.06 90
2017-07-14 14:01:46.866385 5 30300 353.33 153.88 81
2017-07-14 14:01:56.141732 6 39572 167.00 59.77 11
2017-07-14 14:01:57.930102 7 41358 137.33 53.38 248
2017-07-14 14:02:17.210412 8 60630 255.00 94.21 10
2017-07-14 14:02:27.674790 9 71089 56.00 39.03 216
2017-07-14 14:02:29.776414 10 73188 159.00 57.82 10
```

Figure 9: An example of the format of saved data. At the top we have six lines for the header and the show the data for the first ten measurements in a run.

We also prepared CosmicWatch web application which allows to monitor data taking process in real time. You can visit *cosmicwatch.lns.mit.edu*→Start measurement section, and use provided detector_server program (python script or executable version) to connect your device to the web application. Then you have to click on "Start data collection" on the left hand side. If you

[1]To reduce the detector dead time, you can turn off the screen by modifying the Arduino code (set OLED and LED Booleans set to "0")

stop data collection, you can save this results to text file ("Save" button) or load the previously recorded measurements ("Open" button).

6 Example measurement

We were given permission to make a measurement during a domestic flight from Boston to Chicago. As the airplane climbs to cruising altitude (roughly 40,000 ft), we observe a significant increase in the rate of the detector compare to the ground-level measurement. This is near the peak in the cosmic ray muon production region at approximately 45,00060,000 ft. This result shows that we can easily identify the altitude of the airplane and correlate it to the cosmic ray muon flux. More example measurement is available on our web page.

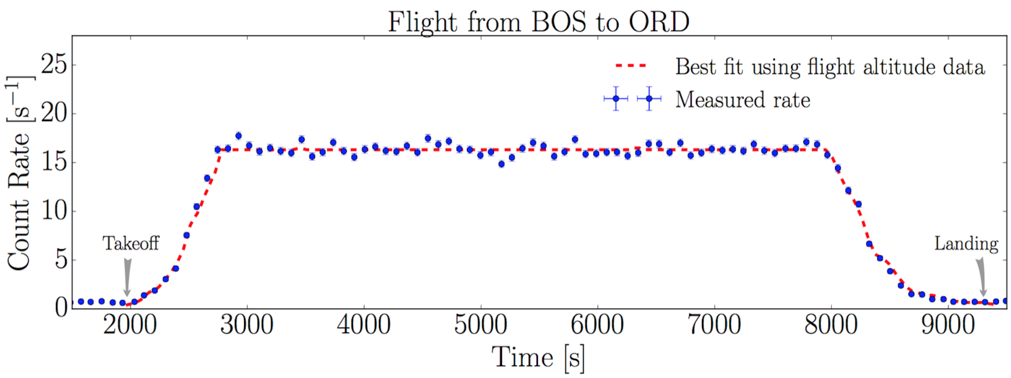


Figure 10: Measured event rate during short domestic flight. The website `https://flightaware.com/` provides the flight details (eg. time and altitude of the aircraft).

7 Troubleshooting

The most common mistakes we see is either using the wrong component or insufficient solder on one of the connections. The missed connections are typically found on the small legs of the DC-DC booster (LT3461A) or the op amp (LT1807).

The main PCB has been designed with several test point connections for trouble shooting. The test point connections are labeled TP1, TP2, and TP3. TP1 is connected to the SiPM directly after the 1k resistor, TP2 is connected to the output of the amplifying circuit, and TP3 is connected to the output of the peak detector circuit. Using these, one can identify which part of the detector is experiencing problems. Three example traces from these test points should look similar to the three traces shown in Fig. 11.

Incorrect bias voltage for the SiPM. Prior to plugging the SiPM PCB into the main PCB,

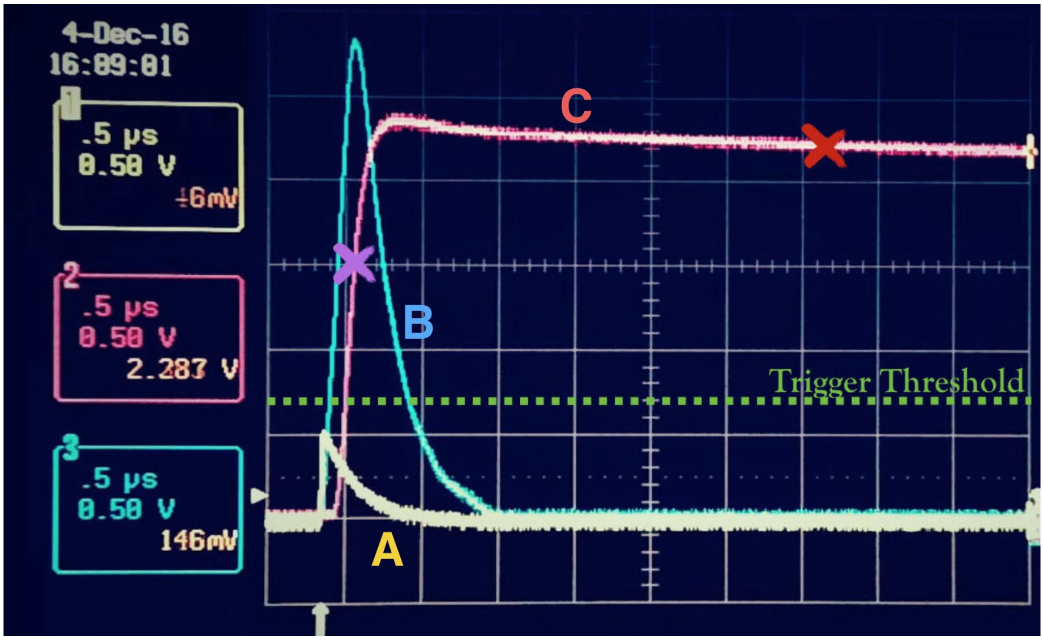


Figure 11: The three waveforms from the test point connections. Waveform A represents the raw SiPM pulse after it passes through a 1kΩ resistor. B is the amplified pulse, which is see is roughly a factor of 10. Waveform C is from the peak detector circuit.

make sure if you are supplying the correct voltage. Using a multi-meter, check to see if that you receive roughly +29.5 V between the HV pin on the main PCB 6-pin header and ground. If you power the detector without the SiPM PCB connected, you will see continuous flow of counts (LED will be continuously on). If the voltage is not +29.5 V, then there is a problem with the DC-DC booster circuit. Double check the components and connections. The legs on the LT3461A are often the culprit.

High count rates. If you plug the SiPM PCB into the main PCB and the count rate is above say 2 Hz (for a 5x5 cm piece of scintillator), you most likely have a problem (unless you are near some gamma-ray source, or above 10,000 ft). A high count rate may be due to the black tape not properly making the plastic scintillator light-tight. The most likely area that we've found light to enter is around the corners of the plastic scintillator. If you still see a high count rate, you can raise the trigger threshold in the Arduino code. In the Arduino code, there is a variable called SIGNAL_THRESHOLD, we run it around 25 (which is the ADC count number to trigger on). Try raising this to 30 or higher.

If the detector is continuously counting, that means that the main PCB does not see the SiPM PCB. The reason for this is that the pull down resistor (R3 on the SiPM PCB) holds the line at ground when there is no pulses. If the main PCB does not see this resistor, the wiring acts a an antenna and begins triggering on the noise in the line.

Low count rates. If the detector has power and we do not see any signals, we need to check the circuit. There are several test points on the circuit board, but let's start with the SiPM signal. With the SiPM PCB plugged into the main PCB, power the detector using the mini-USB connection. Using a BNC cable, connect the BNC output on the main PCB to an oscilloscope. If your SiPM PCB was constructed properly, you will see positive pulses that are roughly 500 ns in width and 10-100 mV in amplitude whenever a muon passes through the plastic scintillator (roughly at 0.5 Hz on average). These are the raw pulses from the SiPM. If you don't see any pulses, you have something wrong with the SiPM PCB. If you do see the pulses, we need to begin troubleshooting the main PCB.

You should be able to narrow down which part of the circuit has the problem by checking the test points TP1, TP2, and TP3. A problem here will probably indicate a misplace surface mount component.

- The TP1 connection is connected directly after the 1k resistor from the SiPM (as shown in the top middle of Fig. 6). This pulse should look nearly identical to the raw SiPM pulse from the BNC connection.
- TP2 is connected to the output of the amplification circuit. The amplification should appear to be roughly 10 compared to the raw SiPM pulses. We include a capacitor on the amplifying circuit to smooth the pulse and eliminate some of the high-frequency components (see waveform B in Fig. 11).
- The TP3 waveform should be a steeply rising pulses, roughly the same amplitude as the amplified pulse, but with a much longer width in time (see waveform C in Fig. 11).

If all the test point connections look good, and the SiPM appears to be working properly, the final area we could have a problem in is the Arduino Nano. Either there is the wrong code compiled on the Arduino or the Arduino is broken. The most up-to-date code for this version of the detector can be found in the Arduino folder.

No readout on the OLED screen. If you do not see anything on the OLED screen when you plug it in, there are two possible problems. Either the Arduino code has the OLED Boolean set to "0" (off), or there is a problem with the screen. We can turn the screen on by setting the OLED Boolean in the Arduino code to "1", then verifying it and uploading to the Arduino. We've found two potential problems with the Arduino OLED screens. First, occasionally we receive screens that are cracked and do not work. The second problem we've run into is that one manufacturer of the screens reversed the VCC and GND connection, even though their picture on their website illustrates the correct orrientation.

Problems recording data to the computer. If you are trying to read the data through a USB port and do not see anything, there is probably a problem with the Arduino being recognized by the computer. We've compiled executable programs for Windows, MAC, and Linux. These programs are simple python based serial readers. If, for some reason, they aren't working with your OS, we can use the python version instead, but you will have to install pyserial on your OS. The python code, import_data.py simply loops through all connected ports on the computer and asks you which one of these ports has the CosmicWatch detector plugged into. MAC requires you to have the CH340g driver installed in order for your computer to recognize the Arduino Nano. Linux may require you to add permission to read from the USB port.

If you are trying to run the device on the website, you should only have to run the executable program detector_server for the appropriate operating system.

References

[1] http://cds.linear.com/docs/en/datasheet/3461Afa.pdf. Linear technology, July 2017.

[2] http://sensl.com/downloads/ds/DS MicroCseries.pdf. C-series low noise, blue-sensitive silicon photomultipliers datasheet, July 2017.

[3] https://github.com/adrianmihalko/ch340g-ch34g-ch34x-mac-os-x driver. Ch340g driver, July 2017.

[4] https://www.amazon.com/. Amazon, July 2017.

[5] https://www.arduino.cc/en/Main/Software. Arduino, July 2017.

[6] https://www.digikey.com/. Digikey, July 2017.

[7] https://www.ebay.com/. Ebay, July 2017.

[8] https://www.elecrow.com/5pcs-2-layer pcb.html. 5pcs- 2 layer pcb, July 2017.

[9] http://www.enclosuresandcasesinc.com/. Enclosures and cases, July 2017.

[10] SensL. Microcseries, July 2017.