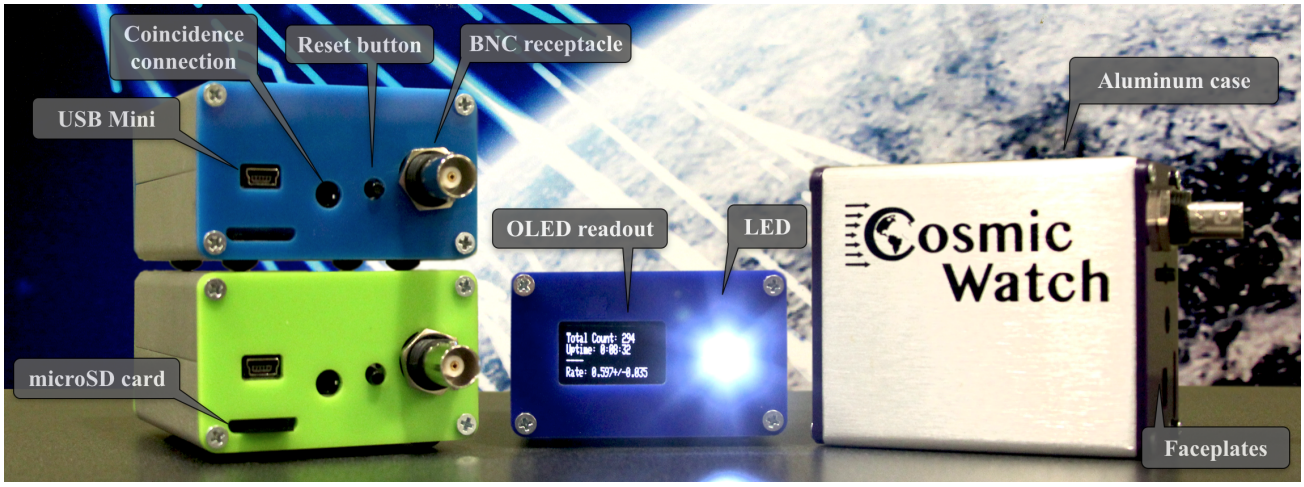
CosmicWatch: The Desktop Muon Detector

Instruction Manual



February 21, 2018

The CosmicWatch Desktop Muon Detector is a Massachusetts Institute of Technology (MIT) and National Center for Nuclear Research (NCBJ) 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. Data can be recorded directly to a microSD card, to a computer, or plotted in real time on the CosmicWatch website. This document describes in detail how to build, test, and troubleshoot a detector.

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

Contents

1 Document Overview

CosmicWatch is an outreach program developed by MIT and NCBJ that enables students to build their own in-expensive cosmic ray muon detector. A single detector costs approximately 100$ and takes a novice high-school student approximately 4-hours to build for the first time and a second detector can be built in under two hours. Details regarding our project can be found on the CosmicWatch website:

www.cosmicwatch.lns.mit.edu

This version of the detector includes several improvements over the previous itteration. In particular, this version includes a microSD card reader/writer, updated electronics, and a connection which can be used to connect multiple detectors together to make coincidence measurements. Further, we reduce the noise, improve the electronics circuit design, and update the software necessary to use the detector. All suplementary marerials pertaining to this project is available in the GitHub repository:

https://github.com/spenceraxani/CosmicWatch-Desktop-Muon-Detector-v2

The CosmicWatch Desktop Muon Detector consists of a 5 cm×5 cm×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 (an Arduino Nano) 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, temperature, and detector dead time. The threshold for a signal from the SiPM to trigger the data acquisition can be adjusted in the provided Arduino software. The detector can be powered by a USB Mini to USB connector.

There are multiple ways to read out data from the detector. The OLED screen on the front of the detector updates every second with the count number, count rate, and a bar indicating the SiPM pulse amplitude of the most recent event. An LED flashes every time an event is registered. The detector can be connected to an oscilloscope through the BNC header on the back of the detector to view the raw SiPM pulse. We include a python based program (supplementary material: Recording_data/import_data.py) that allows the user to record data directly to the computer via a USB port, or connect directly to the webstie to plot the data in real-time. Finally, we also provide code for recording data directly to a build-in microSD card reader/writer.

This document serves as an instruction manual for the desktop muon detector. We begin by outlining simplified, short instructions for users who are already familiar with the basic concepts of the detector. Then, we break it down each step into a more detailed description in which we 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.

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 section.

1. Purchase all components listed in the "Purchasing_List.xl." This includes purchasing the printed circuit boards.
2. Upload the coincidence.ino code to the Arduino Nano. This requires several libraries to be installed which are listed in the header and the README file in the Arduino directory.
3. Populate all the components using the SMT reference (SMT_reference.pdf) on main PCB and SDcard PCB.
4. Power the detector through the Arduino and test HV pin on Main PCB to ensure it is delivering roughly +29.5 V.
5. Populate the SiPM PCB.
6. Machine the scintillator to 5×5×1 cm. Drill 4 mounting holes for the SiPM PCB then heat polish the machined surfaces.
7. Wrap the plastic scintillator in a 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 to make it light-tight.
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. Name the detector by uploading the naming.ino Arduino code.
12. Upload the OLED.ino or the SDCard.ino code to the Arduino depending on if you want to use the OLED screen or the microSD card respectively.
13. If there is a problem, see troubleshooting section (Section 9) 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.

3.1 Purchasing the components

The first step is to purchase the required equipment. In the GitHub repository we've provided a list of the components used in the project. It should be noted that it is advantageous to build multiple detectors, either at your institution, or in a group in order to reduce cost of purchasing single components. We've also included the price at which we purchase these items for the project.

1. Send the PCB Gerber files (`PCB_files/PCB_Gerber_files.zip`), contained in the zipped PCB file, to a PCB manufacturer. The three PCBs are manufactured together and fit inside 10 cm×10 cm square. They are all two layer, and the PCB thickness should be 1.6 mm in order to properly fit inside the aluminum case. A rendering of the PCBs are shown in Fig. 1. For manufacturing, we use Elecrow.com [7], from which you can order a minimum of 5 PCBs for approximately 5 USD. We found that expedite shipping takes roughly 10 days. If you do not choose expedite shipping, manufacturing and shipping takes approximately a month.

2. Elecrow.com can also laser cut acrylic. We use this service for the colorful faceplates (see Fig. 2) on the front and back of the case. In order to laser cut the faceplates, simply upload the "EndPlates_for_laser_cutting.dxf.zip" file to the Elecrow.com website, select the dimensions to be 10×15×2.5 mm, and choose your desired color. Each 10×15×2.5 mm acrylic plate holds two front plates and two backplates.

3. The aluminium enclosure (2506H-2.9 inch anoodized aluminium, without faceplates) can be purchased from enclosuresandcasesinc.com [8] by providing them with the top and bottom silkscreens found in the "Enclosure_Files" folder. The number 2506 represents the split case design of a certain size, the 'H' is not used since we do not want the metal faceplates, and the '2.9 inch' represents the lenght of the case.

4. The electronic components found in the excel file can all typically be found at either Digikey.com [5], Ebay.com [6], or Amazon.com [4]. We also use a few components from Linera Technology and Texas Instruments. The spreadsheet provides a link to where we purchase our components from, but most components are generic.

5. The 16 MHz Arduino Nano 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 [9]. We use the SensL 60035 SMT C-series. It has a photocathode area of 6 mm×6 mm (36 mm^2 active area). 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 may be able to help you. Currently, SensL offers a special price (60 USD) for students here, [10]. When purchasing in quantities greater than x100, the price currently drops to 48 USD per SiPM.

7. There are several companies that sell plastic scintillator, but we found that physics departments often have left over scintillator from previous experiments. We use 1 cm thick scintillator since it appears to be the most common size available and will produce a sufficient number of photons to help distinguish a muon event from a background event. Ebay and other online shops often sell used scintillator as well. The SiPM photon detection efficiency peaks at 420 nm , 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 labs and physics departments, but have talked with people who aquire scintillator from companies such as Eljen [**?**] or Saint-Gobain [**?**]. We've also seen scintillator of the correct size available on Ebay.

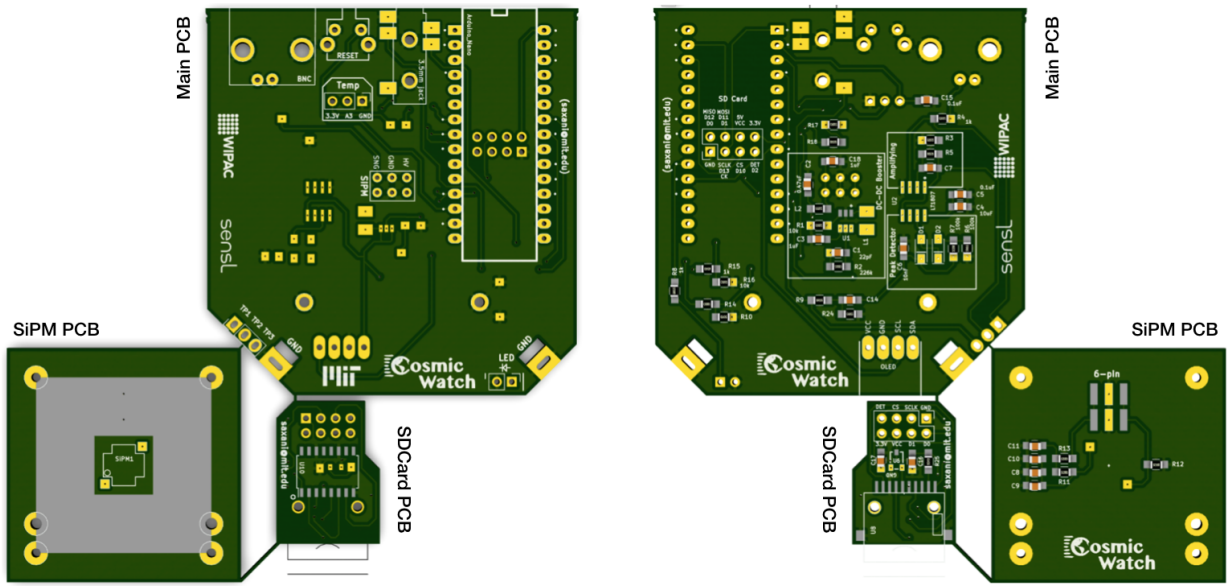


Figure 1: A rendering of the three PCBs.

3.2 Uploading code to Arduino Nano

Before soldering the header pins onto the Arduino Nano, the OLED.ino (Arduino_code/OLED.ino) Arduino code should be uploaded to make sure the Arduino works. First, we must install the

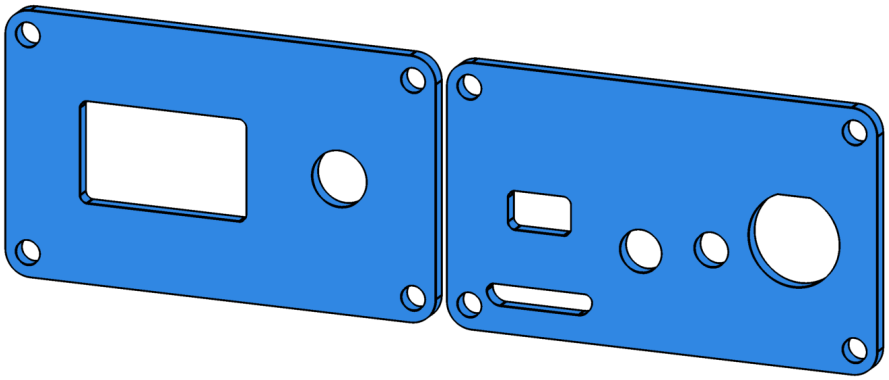


Figure 2: A rendering faceplates for the aluminum enclosure.

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 here: https://www.arduin

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].

It also looks like Windows users will need to install the CH340g driver.

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 – by Adafruit Version 1.0.1
2. Adafruit_GFX – by Adafruit Version 1.0.2
3. TimerOne – by Jesse Tane et al. Version 1.1.0
4. EEPROM
5. SD
6. Wire
7. SPI

the EEPROM, SD, Wire, and SPI are probably built-in.

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

We should now be able to upload the Arduino code. Simply click the check mark at the top left of the IDE to make sure the code compiles without errors, then click the arrow button to upload the code to the Arduino. The full sketch utilizes approximately 70% of the Arduino storage space, and 60% of the available SRAM. It will take about 40 seconds to upload to the Arduino. See the Sec. 9 if you are having problems. If it uploads without any errors, we'll assume the Arduino is working properly.

4 Populating the PCBs

There are three PCBs which must be populated. Each PCB was designed using 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. If you are unfamiliar with using surface mount technology (SMT), check out an online video on YouTube. We've made some videos going through the construction of the detector[11]. These can be found here:

www.cosmicwatch.lns.mit.edu

The reference list for populating the PCB can be found in the supplementary materila (SMT_reference). The color of the reference list determines which PCB the component is associated with.

4.1 The SDcard PCB

The smallest PCB is for the microSD card reader/writer. This PCB contains the microSD card socket, a 3.3V regulator, and a logic level translator for communicating with the microSD card at 3.3V. We do not use the onboard 3.3V from the Arduino Nano, since it has a maximum current rating of 150 mA and microSD cards are known to draw upwards of 200mA at peak load. The logic level translator has a direction, and pin 1 is indicated by a small circle on the silkcreen.

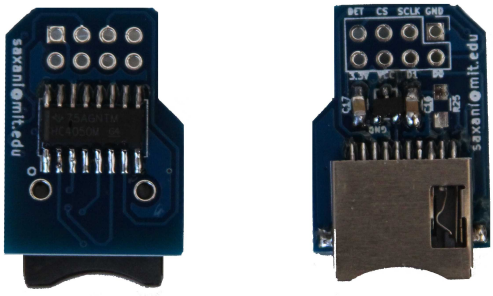


Figure 3: The top (left) and bottom (right) of the SDcard PCB.

4.2 The main PCB

The largest PCB, the Main PCB, (shown in Fig. 4) 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.

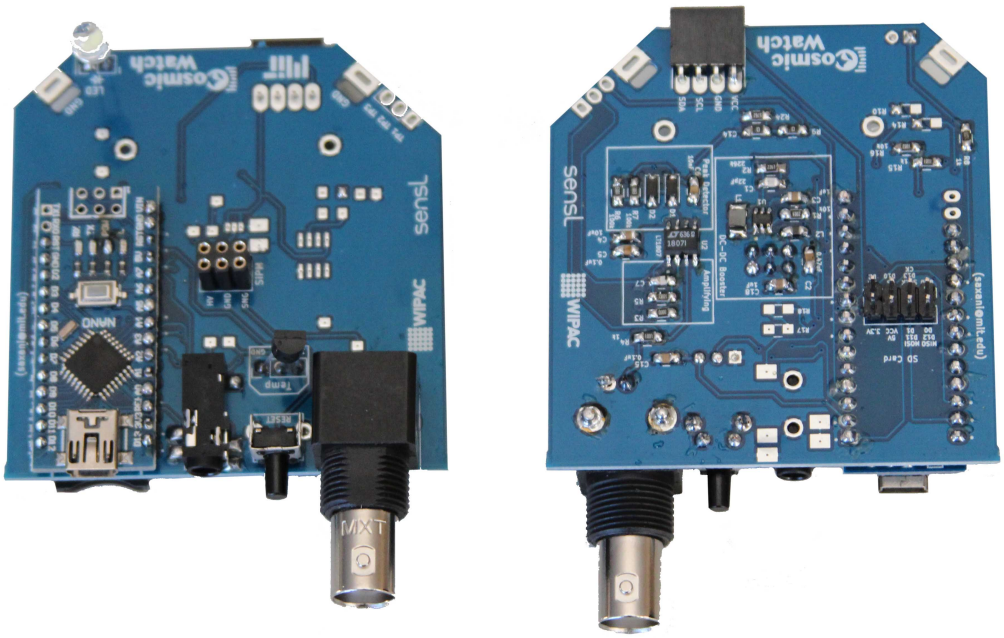


Figure 4: The top and bottom of populated Main PCB.

On the bottom of the Main PCB, inbetween the two headers for the Arduino Nano there is an 8-pin (2x4) connection. This is used for mounting the SDcard PCB after it is populated. Rather than purchase an 8-pin header for this connection, we use the 6-pin (2x3) header that came shipped with the Arduino Nano and two extra pins from the Arduino Nano header. You can see, next to some of the header connections for the Arduino Nano, there are small silkscreen dots next to them. These are the pins that are used for the detector. We solder the 6-pin and 2-pin header into the 8-pin connection before mounting the Arduino Nano (short legs on the header should go into the 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 reset the Arduino Code. The LED connection towards the front of the Main PCB is used to mount an LED. It should be noted that this connection has a direction, as indicated on the silkscreen (short leg of the LED should go to the ground connection). The OLED screen 4-pin connector should be mounted on the bottom of the board, as shown in the image. There is also

a 4-pin 3.5 mm female audio connection, which is used for coincidence measurements. We use a 4-pin connector, but the coincidence code only actually requires 3-pins.

The final connection on the top of the Main PCB is the BNC connector. It is connected directly to the SiPM output (anode or pin 1 of the SiPM). 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 circuit through the BNC connection. This is useful for calibrating the circuit (i.e. determining the relationship between SiPM pulse amplitude and measured ADC value). A further description of how we calibrate the electronics can be found in Section 5.2.

The final component to mount is the Arduino Nano. We should first insert the 1x16 and 1x14 pin header into the main PCB (recall we removed two pins from one of the headers for the SDCard PCB), then solder them in place. The headers should have the long legs going through the main PCB. Make sure you mount the header onto the correct side of the main PCB. Before soldering the Arduino in place, we should also ensure that the 8-pin (2x4) connection for the SDcard PCB already has 8-pins in place. Again, note which side of the PCB these should be inserted. Now, we can mount the Arduino Nano and then, finally, solder the SDcard PCB in place.

After the main PCB has been populated, we can test the supply voltage to SiPM. Power the Main PCB using the Mini USB connection on the Arduino and check that the voltage between the HV pin and the GND pin on the 6-pin header reads aboute +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 then probably see some nonsense count rate. This is because the SiPM PCB is not connected and the Arduino still reports the voltage on analog pin0 (A0) of the Arduino. Without the SiPM PCB plugged in, A0 is reading static.

4.3 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 the resistor on the anode side of the SiPM, 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 SiPM PCB is used strictly for the SiPM and a reflective ground plate. Fig. 5 shows the fully populated PCBs.

PCBs aren't typically light-tight, therefore we have attempted to plate either side of this PCB with a metal filling in order to reduce the posibility of a photon passing through the PCB. The large metal filling around the SiPM simply helps with reflecting photons.

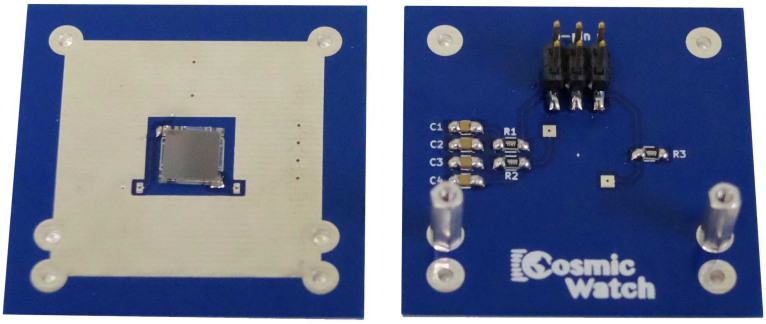


Figure 5: The top (left) and bottom (right) of the SiPM PCB. On the top we can see the SiPM in the middle surrounded by a reflective grounding plane. On the bottom we see the 6-pin connector and two stand-offs used to connect to the main PCB.

4.4 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 see 10mL bottles of optical gel on Ebay.

We've included a CAD drawing of our machined scintillator in the pictures folder of the supplementary material. The filet edges shown in the CAD drawing are simply due to the fact that our scintillator was extruded rather than cast. 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. 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. Ensure that drilling is performed at low speeds and try to remove chips from the hole as they are produced.

After the scintillator has been fully machined (milled to size and 4-holes drilled), we can heat polish the machined edges 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.

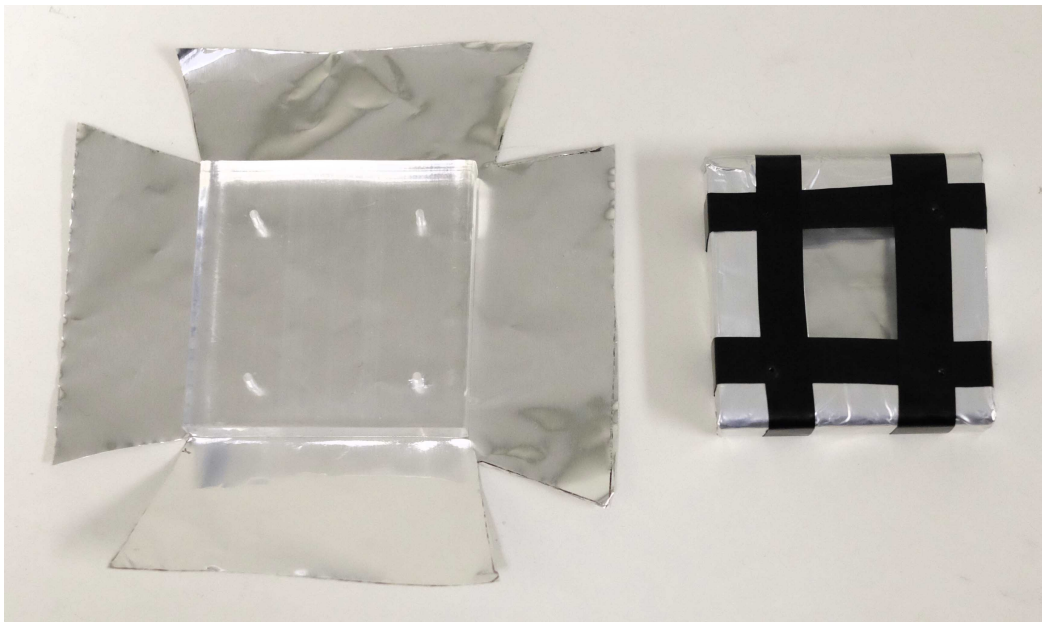


Figure 6: Left: The machined scintillator sitting on top of the reflective foil. Right: the covered scintillator with a 2 $\times$2 cm opening for the SiPM.

4.5 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 $\times$ 2 cm open area on the face of the scintillator for the SiPM as shown in the right side of Fig. 6. 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 four No. 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 detection efficiency will just be slightly lower.

The SiPM PCB is mounted to the plastic scintillator using four 5/16" No. 0 screws. Do not tighten the screws all the way since this would put too much pressure onto the SiPM face. Instead, we leave approximately 0.5-1 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 the Main PCB on the right side of Fig. 7.

4.6 Assembling the Desktop Muon 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 (labelled HV), the middle two are used for ground (GND), and the bottom two are used for the signal (SNG). 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 (setting of the threshold in the Arduino code, version of the detector, gamma ray background, altitude ...), you should be able to see a count rate of about 0.5 - 1.5 cps at sea level. A good guess is that roughly 0.5 Hz of these are cosmic ray events, however, part of the 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 the background components.

If you would like to insert the detector into an electronics enclosure, we have desinged the detector to fit inside the 2506H-2.9 inch aluminium case from Ref. [8]. We do not however use the faceplates provided from this company. Instead, we 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 (Enclosure_Files/EndPlates_laser_cut_files.zip).

5 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.

5.1 Description of the circuit

The main PCB contains electronics used to amplify and shape the signal from the SiPM such that it can be measured by the Arduino Nano. 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

Figure 7: A picture of the complete detector.

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 (/Recording_data/import_data.py) to log the data to computer or connect the desktop muon counter to our website to record and plot data in real time.

5.1.1 DC-DC Booster

The DC-DC Booster is used to take the 4.6 V DC supplied by the USB connection and boost it to +29.5 V in order to bias the SiPM. If we look at the bottom-side of the main PCB (see Fig. 4), 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).

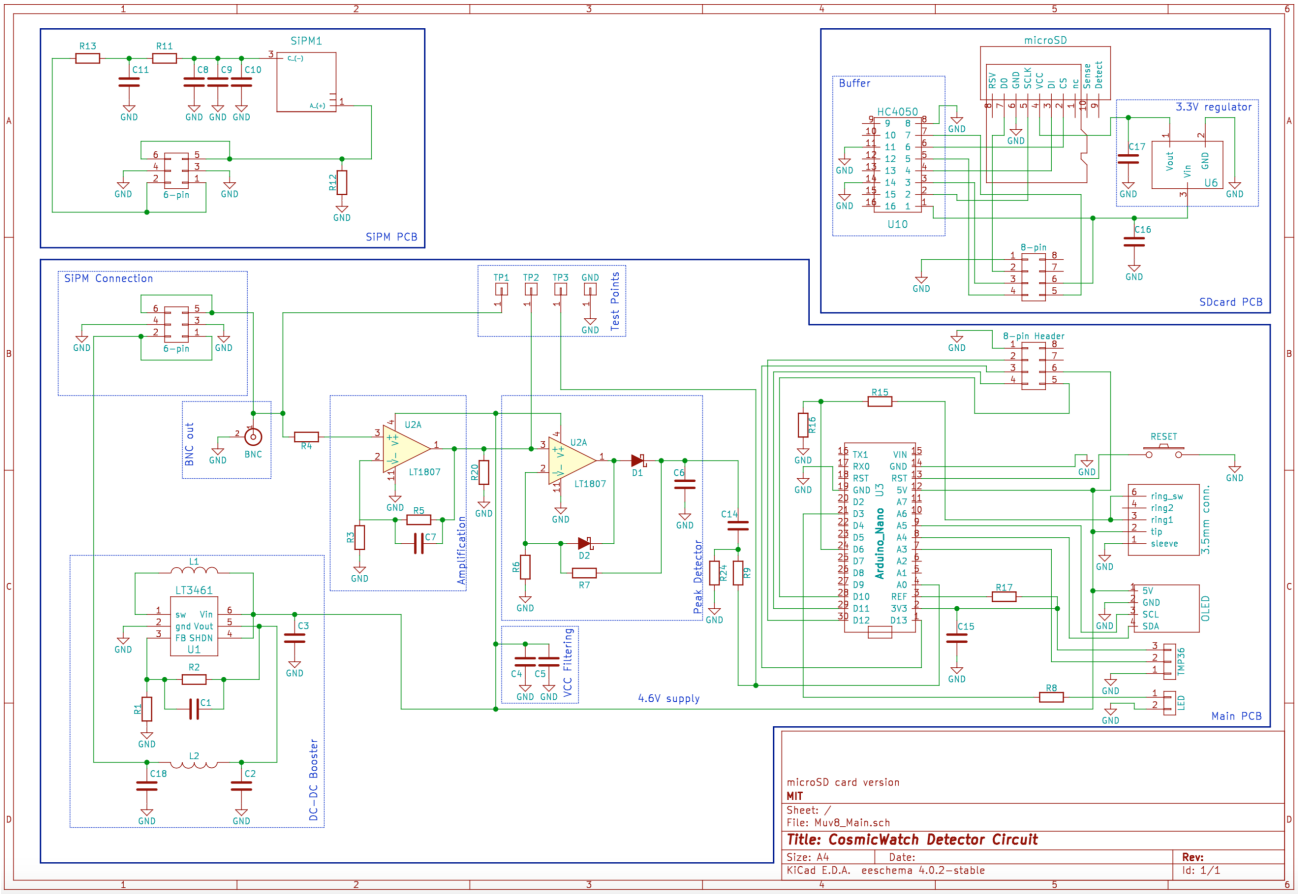
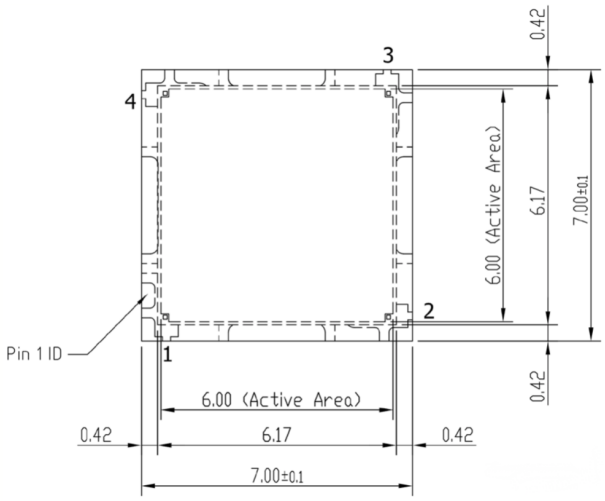


Figure 8: The circuit diagram for the detector. We have broken the circuit into the individual PCBs, outlined in solid blue. The dashed blue lines separate the individual parts of the circuit and are described in the text.

5.1.2 SiPM circuit

The SiPM circuit on the top left of Fig. 8 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 low-pass filter). The SiPM pins can bee seen below in Fig. 9. 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. 8, 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. 8, 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 9: 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.

5.1.3 Amplifying circuit

This circuit takes the positive signal from the SiPM and amplifies it by about a factor of 24. We limit the high-frequency amplification using a 10 pF capacitor (C7) to improve ADC response. The waveform of the amplified signal can be seen by connecting a scope probe to test point 2 (TP2). The amplified signal is then sent to the peak detector circuit.

5.1.4 Peak detector

The purpose of the peak detector circuit is to latch onto the peak of the amplified pulse and hold the voltage at that level for a sufficient time 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 during charging. When the pulse from the amplifying circuit subsides, D1 becomes back-biased and forces C6 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 0.5 ms.

Since the output of the op amp can only be driven to 4.6 V (since the opamp is connected to the +5 V rail on the Arduino – which actually only outputs 4.6 V when connected to a USB port) and the voltage drop across D1 is approximately 0.4 V, the maximum output voltage now becomes approximately 4.2 V. We have specifically chosen the diodes to minimize the forward voltage drop, thus allowing us to measure a higher possible voltage.

5.1.5 microSD card circuit

The microSD card circuit is shown in the top right of Fig. 8. This circuit contains a 3.3 V regulator, which takes a 4.6 V input from the Arduino Nano, and converts it to a 3.3 V line for powering the microSD card. The capacitors on either side of the 3.3V regulator act as filters to reduce noise and store energy. Also, included in this circuit is a non-inverting buffer which is used as a logic level translator that converts the 4.6 V communication to 3.3 V (the microSD card requires 3.3V communication).

5.1.6 Arduino

The Arduino clock is 16 MHz, but a single ADC measurement takes approximately 90 clock cycles, therefore we are only able to sample the waveform at roughly 178 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 (order 0.5 ms), the first few 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.
7. Record data to the microSD card.

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 5-10 ms due to the limited speed of the serial communication (at a baud rate of 9600 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.

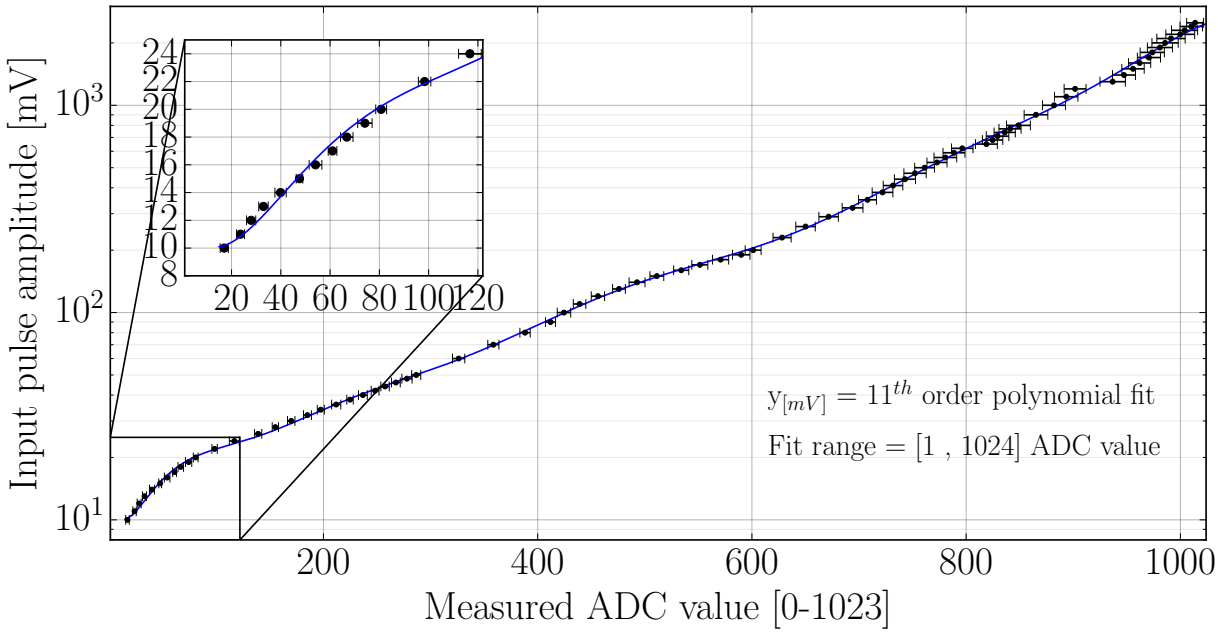
The Arduino waits for a signal greater than the trigger threshold on the analog pin A0, then begins data acquisition. We do not record the triggering measurement since it may have been made during the rise time of the pulse. Instead, we take the subsequent four measurements and average over them. This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

5.2 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 an arbitrary 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 high-order polynomial. The result of the fit and the data is shown in Fig. 10. We use these results in the function "Calibration_fit" in the Arduino code.

6 Recording data

There are multiple methods to readout data from the detector.



s

Figure 10: This plot shows the average measured ADC value for an injected pulse of known amplitude. The blue line is the best fit according to a nth order polynomial fit (the degree of the fit is shown in the plot).

1. The OLED screen reads out the number of observed triggers, the detector up-time, and the count rate (with the dead time taken into account).

2. Data can be recorded to the computer by running the import_data.py python program. The program will scan through the USB ports and prompt the user to identify which of the in-use USB ports the detector is plugged into. Once the user identifies the port, the program will ask the user what they want to do:

   (a) record the data to the computer,

   (b) save the data from the microSD card,

   (c) delete all data on the microSD card,

   (d) connect the detector to the website

   If the first option is selected, the program will ask the user where they would like to save the data. Either the OLED.ino or the SDcard.ino code can be uploaded to the Arduino Nano.

3. Data can also be plotted in real-time on our website: www.cosmicwatch.lns.mit.edu. This simply requires the user to select the last option in the list above and access the 'Start Measurement' section on our website.

4. The data can also be saved directly to a microSD card. This requires the user to upload the SDCard.ino Arduino code to the Arduino. Each time the detector is plugged-in or reset, a new file is created on the microSD card. Selecting the second option above will create a folder in the current directory called SDFiles, and will begin saving all the files on the SD card to that folder. The files on the microSD card can be deleted by selecting the third option.

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*`).

An example data file is shown in Fig. 11. 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. 11 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 (as shown in Fig. 10). The sixth column is the calculated SiPM pulse amplitude – this is calculated through the calibration described in Section 5.2. The second to last column is total dead time measured . Dead time must be taken into account for all rate measurements. Finally, the last column is the measured temperature.[1].

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

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

```
################################################################################
### CosmicWatch: The Desktop Muon Detector
### Questions? saxani@mit.edu
### Comp_date Comp_time Event Ardn_time[ms] ADC[0-1023] SiPM[mV] Deadtime[ms] Temp[C]
################################################################################
Device ID: Destroyer
2018-01-02 12:51:10.746574 1 32 7.00 13.07 33 24.12
2018-01-02 12:51:13.184446 2 2495 146.25 48.73 105 23.39
2018-01-02 12:51:20.463850 3 9781 17.00 14.50 349 23.39
2018-01-02 12:51:20.695046 4 10033 191.00 54.53 390 23.39
2018-01-02 12:51:22.790896 5 12130 196.50 54.84 464 23.39
2018-01-02 12:51:25.053360 6 14394 167.00 51.98 538 23.39
2018-01-02 12:51:26.389299 7 15731 294.00 76.51 578 23.39
2018-01-02 12:51:26.422539 8 15741 121.00 42.58 584 23.39
2018-01-02 12:51:27.778981 9 17101 19.75 15.43 659 23.39
2018-01-02 12:51:28.400843 10 17722 148.00 49.04 665 23.39
```

Figure 11: An example of the format of saved data from the detector. At the top we have six lines for the header and the show the data for the first ten measurements in a run. The 7th row is the device ID ("Destroyer" in this case) that was assigned to the detector after uploading the Naming.ino code to the detector. If using the SDCard.ino code, the first two columns are exempt.

7 Setting the detectors up in coincidence

The 3.5 mm female audio jack located on the back of the detector is used to connect multiple detectors together such that they can operate in coincidence mode (*master* or *slave* mode). The "slave" detector is defined as the detector which records the event only if the *mater* detector also triggerd within $30\mu s$ window. Once two detectors are connected via a 3.5 mm male-to-male audio cable, the mode is selected by resetting both detectors using the reset button on the back. The first detector to be reset becomes the *master*, while the second detector (if reset anytime between 10 ms and 2000 ms after the *master*) becomes the *slave*. The tip conductor of the audio cable provides power the the other connected detector, which means only one detector needs to be connected to a USB power source. We use the hardware coincidence in order to make several measurements described in Sec. **??**.

8 Example Measurements

This section describes various measurements that have been performed using the Desktop Muon Detectors.

8.1 Beamline measurement at FermiLab

The top-left measurement in Fig. 12 shows the count rate as a function of time of a single detector located in the Fermilab testbeam. This measurement illustrates the usefulness of the BNC receptacle on the main PCB. The detector was connected to a 10,000 mAh USB power pack

and used as a portable trigger for another experiment we were running downstream. During the two second long beamilne spill, GeV scale pions and electrons pass through the detector. The raw SiPM pulse is sent through the BNC receptacle, through a roughly 30 m long BNC cable to a seperate data aquisition system.

8.2 Rate measurement 1km underground at Super Kamiokande

Two detectors were brought to the Kamioka Mine in Japan to perform a rate measurement near the Super-Kamiokande **??** detector. Super-Kamiokande was built in the Kamioka mine and has roughly 2100 m.w.e. of overburden. For this measurement, the detectors were left in their aluminum enclosure and placed one-on-top of each other. They were connected together using a 6-inch 3.5 mm audio cable and the *slave* data was recorded directly to a laptop.

Using the same detectors and setup, a rate measurement was also performed outside the Kamioka mine and in the airplane at 36,000 ft when travelling from Warsaw to Tokyo. Fig. **??** shows the rate for these three measurements as a function of calculated SiPM voltage.

In the Kamioka mine, near the Super-Kamiokande detector, the downgoing cosmic ray muon rate is expected to be attenuated by a factor of $10^{-}5$. During the 8 hour data taking period in the mine, we observed 101 events. This represents a decrease in rate from ground level of approximately 5×10^{-5}.

8.3 Muon rate measurement at 33,000ft

A rate measurement was performed during a flight from the Boston International Logan Airport (BOS) to the Chicago O'Hare Airport (ORD) using a single detector. The data was recorded to the microSD cards with a single detector plugged into a 10,000 mAh USB battery pack. The altitude of the airplane was collected from the flight records found in Ref. [**?**]. The altitude data was scaled by an exponential and plotted to compare to the measured trigger rate data (shown on the middle-left of Fig. 12).

The middle-right of Fig. 12 shows the altitude as a function of measured count rate. Here we show the exponential fit extended beyond the measured value. The count rate errors were calculated by taking the square root of the sum all the events measured at a particular altitude. The uncertainty in the altitude was taken as the .

8.4 Master/Slave mode comparison at sea level

The first measurement illustrates the ability of the detector to distinguish between radiogenic backgrounds and cosmic ray muons at sea level. For this measurement, two detectors were removed from the aluminum enclosure and put one-on-top of the other (as shown on the right side plot of Fig. **??**). The detectors were linked together with a 6-inch 3.5mm male-to-male cable and setup in coincidence mode.

The data was recorded directly to a computer via the mini-USB cable. The measurement was made over the period of three days. The left side of Fig. **??** shows the data from the *slave* detector superimposed on the data from the *master* detector. Above 130 ADC counts, 90% of the triggered events of the *master* also triggered the *slave* detector. Similarly, above 40 ADC counts, 50% of the triggered events of the *master* also triggered the *slave* detector

8.5 Cosmic ray muon angular distribution measurement

The final measurement illustrates the cosmic ray angular muon dependence. For this measurement, two detectors were set to coincidence mode and placed back-to-back, 52 mm apart, inside their aluminum enclosure. Placing the detectors back-to-back rather one-on-top of the other reduces the angular uncertainty. The angle was determined by tapping the detectors in place on a 100 cm long rectangular bar and then positioning the bar against a wall at a known height. We also positioned the detectors such that incoming trajectory had a minimum amount of attentuation from the building. Each data point represents approximately 10-12 hours of data. The measurement at $\theta = \pi/2$ is divided by a factor of 2, since at this angle it accepts cosmic-ray muons from both directions, whereas all the other angles only accept down going muons.

The PDG indicates that the angular dependence at sea level should follow a cosine squared dependance. Fig. **??** shows the measured rate as a function of angle and a cosine squared distribution.

The angular uncertainty was calculated by looking the the maximum and minium angle of a trajectory that could trigger both detectors. This does not represent the uncertainty in the measurement, rather the uncertainty in the individual trajectory of the triggering muon. The uncertainty in the count rate was taken as the square root of the number of counts.

9 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 (these can be seen in the center of Fig. 8). TP1 is connected to the SiPM anode, 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. 13.

Incorrect bias voltage for the SiPM. Prior to plugging the SiPM PCB into the main PCB, make sure if you are supplying the correct voltage. Using a multi-meter (see Fig. 14), check to see if that you receive roughly +29.5 V between the HV pin on the main PCB 6-pin header and ground (GND). 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 roughly 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 18 (which is the ADC count number to trigger on).

If the detector is continuously counting, that might indicate means that the main PCB does not see the SiPM PCB. The reason for this is that the pull down resistor (R12 on the SiPM PCB) holds the line at ground when there is no pulses. If the main PCB does not see this resistor the line will be picking up stray signals and accidentally be triggering the detector.

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. 8). This pulse should look very similar 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 a factor of 20 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. 13).
- 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. 13).

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 supplementary material 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.

Error when uploading Arduino Sketch. This could be one of many potential problems. The first thing to check is to make sure you have selected the Arduino Nano as the device from the pull-down menu Tool→Board→Arduino Nano and also select the USB port from Tool→Port. If you are instead recieving a warning saying that you are running out of flash storage space or SRAM, this could be an indication that the libraries that you are using are a different

version. We use nearly all the resources available on the Arduino, so changes to the libraries can potentially break the our code.

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.digikey.com/. Digikey, July 2017.

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

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

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

[9] SensL. Microcseries, July 2017.

[10] SensL. Sensl, July 2017.

[11] Youtube. https://www.youtube.com/watch?v=e4IXzNiNxgU, accessed: Jan 2018.

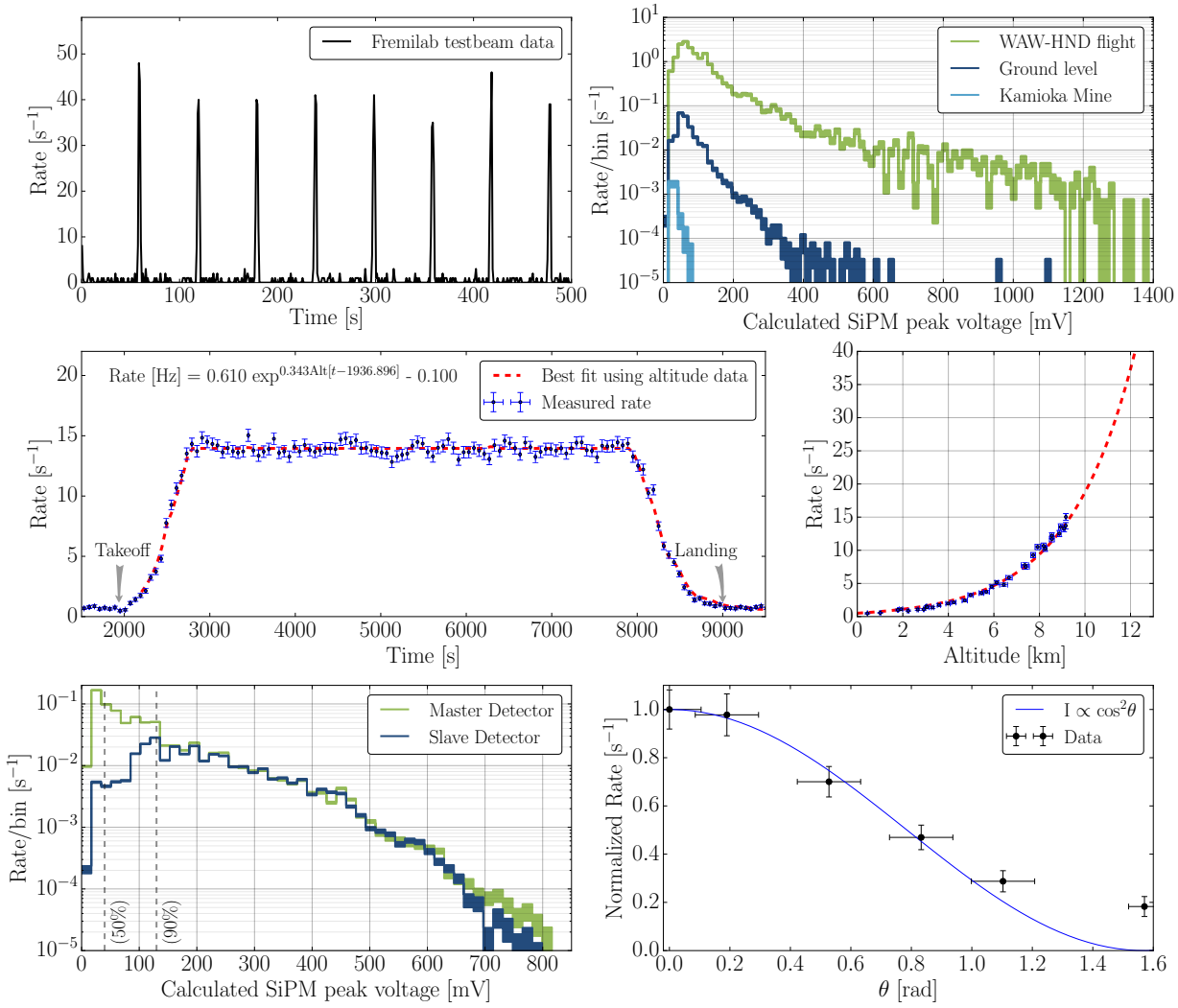


Figure 12: Measurements made using the Desktop Muon Detectors. **Top Left**: The count rate as a function of time of a single detector sitting inside Fermilab testbeam. **Top Right**: The calculated SiPM pulse amplitude for three different measurements: at 36,000ft, ground level, and 1km underground near the SuperKamiokande detector. **Middle Left**: A measurement made at 30,000ft while flying from Chicago to Boston. The dashed-red line indicates the actual altitude of the airplane scaled by the exponential in the top-left of the plot. **Middle right**: The measured count rate versus altitude of the airplane. **Bottom left**: a measurement made by two detectors, setup in coincidence, removed from their aluminium enclosures and placed one-on-top of the other. The verticle dashed lines indicate above which the count rate of the *slave* is 50% and 90% that of the master. **Bottom right**: and agular measurement of the cosmic ray muon flux. The horizontal error bars indicate the angular acceptance of a muon trajectory. The blue line indicates the prediction.

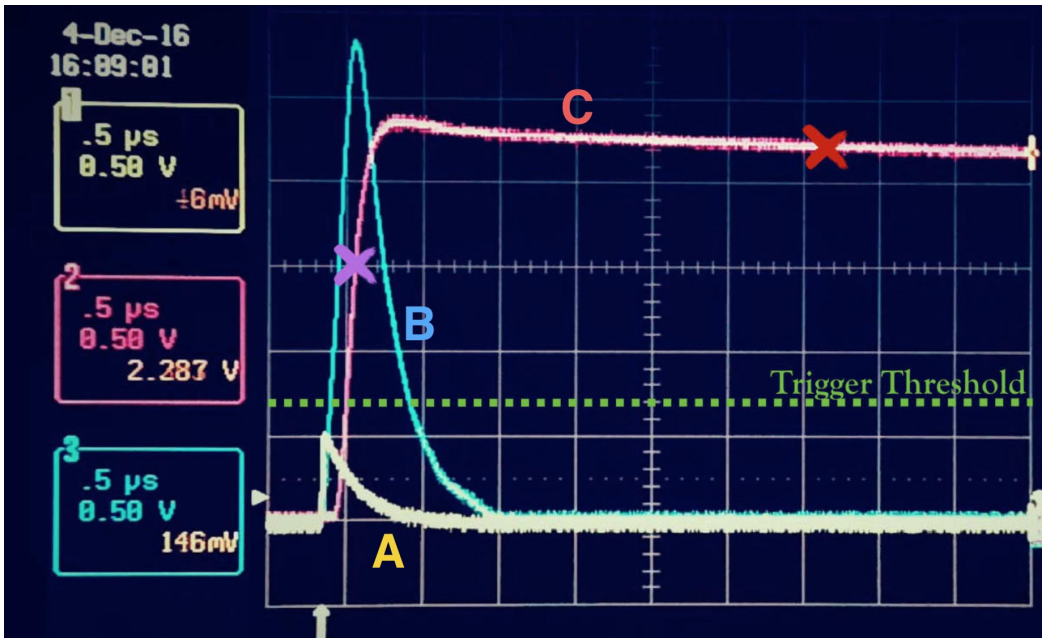


Figure 13: The three waveforms from the test point connections. Waveform A represents the raw SiPM pulse. B is the amplified pulse, which is amplified by a factor of roughly 25. Waveform C is from the peak detector circuit.

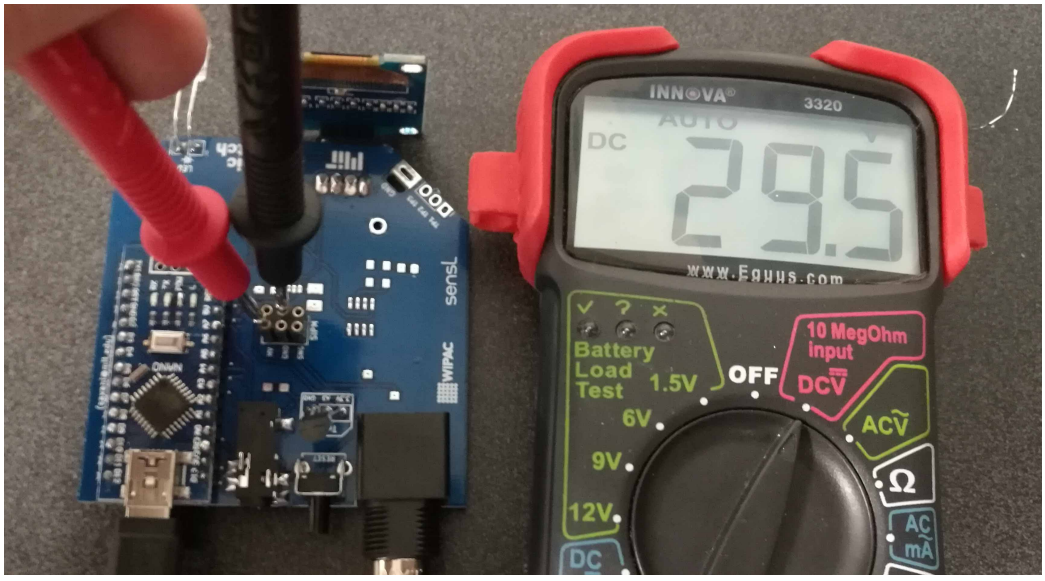


Figure 14: Checking the voltage on the DC-DC booster.