2018-05-01

# Design of a Low-Cost Capillary Electrophoresis Laser-Induced Fluorescence System: Lessons Learned When Trying to Build the Lowest Possible Cost System

Steven James Perry
*Brigham Young University*

# Design of a Low-Cost Capillary Electrophoresis Laser-Induced Fluorescence System:

# Lessons Learned When Trying to Build the Lowest Possible Cost System

Steven James Perry

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Gregory Nordin, Chair
Adam Woolley
Neal Bangerter

Department of Electrical and Computer Engineering

Brigham Young University

ABSTRACT

Design of a Low-Cost Capillary Electrophoresis Laser-Induced Fluorescence System:
Lessons Learned When Trying to Build the Lowest Possible Cost System

Steven James Perry
Department of Electrical and Computer Engineering, BYU
Master of Science

Capillary electrophoresis laser-induced fluorescence (CE-LIF) is widely used to detect both the presence and concentration of fluorescently labeled b iomolecules. In CE-LIF, a plug of sample fluid is electrophoretically driven down a microchannel using a high voltage applied between the opposite ends of the microchannel. Molecules of different sizes and charge states travel at different velocities down the channel. Laser light with a wavelength in the excitation band of the fluorophores is focused near the end of the c hannel. As each species of molecule passes through the laser spot, the fluorophores emit a fluorescence signal which is measured with an optical detector. Commercial CE-LIF systems are available as a complete, expensive package. Custom CE-LIF systems are a collection of commercially available components that meet the specific needs of the end user. Using the custom system in Dr. Woolley's lab as the standard, we hypothesized that 3D printed parts in conjunction with low-cost components could be used to significantly reduce costs and simplify the system, which in turn would make such systems more widely available with a lower barrier to entry.

Testing this hypothesis began with five semesters of small teams of senior undergraduate students trying to design and assemble a low-cost CE-LIF system as part of their mandatory one-semester senior project. I was one of the seniors who worked on the system. Although none of the senior project teams were successful, a partially functioning system was ultimately produced. I reference this system as the starting point system throughout this thesis, which is focused on identifying and solving the system's obstacles in order to reach a working state. I re-designed and re-built each sub-system of the starting point system as needed if within the available budget to create a system that was functional. Budgetary constraints were included in evaluating potential improvements. The end goal was to compare the improved system's performance with that of an expensive conventional system (hereinafter referred to as the standard system) available in Dr. Adam Woolley's laboratory on the Brigham Young University campus.

The ultimate conclusion of my masters' thesis work is that a low-cost CE-LIF system based on 3D printed and low-cost components results in a system that does not offer repeatable performance. In the course of my work, many lessons were learned as to what would reduce overall system costs while maintaining a user-friendly experience. My analysis is given on a subsystem basis to explain what limited the ability of the system to run consistently or what caused it to fail altogether. Details and methodology of my contributions including circuits designed, code written, components used, and 3D models printed in order to test the hypothesis are documented. Attribution of the work prior to mine is laid out when each subsystem is broken down in detail for the failure modes that prevented consistent operation. Future work is suggested to correct the problems encountered and provide a path forward to implement a next-generation system that can be achieved at a lower cost compared to a conventional system, and yet which does not suffer from

the performance problems associated with the version explored in this thesis in which maximum cost reduction was aggressively pursued.

Keywords: CE-LIF, detection, biomarkers, laser, optics, microcontroller, high voltage

ACKNOWLEDGMENTS

I want to thank Dr. Nordin, Dr. Woolley, Anna Nielson, Mike Beauchamp, Dr. Comer, and Joseph Bussio who provided constructive feedback and help with this project. I also thank the senior project students who worked on the first versions of the system and the lessons that they learned. Tuition support and teaching assistantship from the Electrical and Computer Engineering Department to be able to complete the degree is graciously acknowledged. I want to thank my wife and children, Michelle, Zachary and Kimberly. Their constant support for the necessary time and work made the completion of this work possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# CHAPTER 1. INTRODUCTION

Capillary electrophoresis laser-induced fluorescence (CE-LIF) is widely used to detect both the presence and concentration of fluorescently labeled biomolecules. In CE-LIF, a plug of sample fluid is electrophoretically driven through a microchannel using high voltage applied between the opposite ends of the microchannel. One of the ways to form this plug is to have two intersecting channels in a t-shaped geometry, and inject the fluorescently labeled biomolecules through the shorter cross-channel and then separate down the longer channel. The biomolecules travel at different velocities down the channel based on differences in size and charge state. Laser light having a wavelength in the excitation band of the fluorophores is focused near the end of the channel. As each species of molecule passes through the laser spot, the fluorophores with an excitation spectrum matching the incident laser light emit a fluorescence signal which is measured with an optical detector. Each CE-LIF system tends to be one of two kinds, a custom collection of commercial components that are assembled or a commercial CE-LIF system can be bought. Both options are expensive and are highly tailored to the chemistries they are to be used with.

## 1.1 Purpose

We hypothesized that 3D printed parts in conjunction with low-cost components could be used to significantly reduce costs and simplify the system. In turn, the results could make such systems more widely available and have a lower barrier to entry. This thesis describes a CE-LIF system that was designed and implemented to test that hypothesis. Five small groups of students created a first-order system during a sequential series of senior projects. The resulting system was still non-functional and needed dedicated work and deeper understanding of operation in order to fully vet the hypothesis. The purpose of my thesis research was to take that non-functioning system and refine it to a working state. Although my research shows that the combination of all the design decision made didn't work, some decisions are shown to have an impact on reducing

the cost of the standard system through particular subsystems. The working subsystems can easily be incorporated into the standard system by using the results herein.

## 1.2 Justification

As technology progresses, more options become available to accomplish the same task. With the widespread acceptance of 3D printing, parts can be made that are both inexpensive and have a feature set similar to what is available using conventional fabrication methods. By paring down the feature set of the standard system and using lower cost components, the price can be brought down significantly. In addition, I implemented several new methods that drastically changed the way the system was controlled, most notably in terms of the alignment procedure. In a standard system, the microscope is the central element of the system and the most expensive part. Replacing it was, therefore, the first step to greatly reduce the cost of the entire system. The other subsystems incurred similar overhauls to cut costs while striving to create the same subsystem functionality.

The various cost cuts were made to try to prove that a system could be made at a lower cost. Lowering the cost would decrease the entry barrier for new researchers in the field, and for medical tests to be run in areas of the world where money is an overwhelming factor.

## 1.3 Scope (Summary of Contributions)

In order to lower costs, each subsystem needed to be evaluated to discover the minimum viable features in order for it to function. From there a design was implemented for each subsystem. Although each of the five groups of senior project students assembled a number of prototypes for each subsystem, I went through each prototype and found any detrimental flaws that needed to be fixed as well as the features that made it excel. If possible, the necessary changes were made to create a functional version. After each subsystem was finished it was checked in the system to verify that it did not have any conflicts when interacting with the other subsystems. If there were conflicts, each was resolved until the budget was the limiting factor impeding further improvements. All of my work on subsystem development is documented in Chapter 3. Although,

as shown in Chapter 4, the designs implemented allowed the system to function, it did not do so with the consistency of the standard system it was intended to replace.

## 1.4 Outline

The remainder of this thesis is organized as follows.

*Background*. Chapter 2 is a brief survey of the chemistry and history of CE-LIF with descriptions of the function of each subsystem and its use. The cost of each component of the standard subsystems is presented to provide a baseline for our cost reduction efforts.

*System Design and Implementation*. In Chapter 3 I discuss the subsystem designs, the problems I encountered, and how I addressed them while maintaining the lowest cost functional system. Each subsystem, whether it went through one or many iterations, is described in a summary of its functions, lessons learned, what modifications I made and their effect, and what further modifications could or should be made to enhance or achieve performance.

*Experimental Results*. Chapter 4 presents the data obtained from running the system using Alexaflor 532 (AF532) as a standard fluorophore for calibration purposes. A concentration of $1\mu$M was used to evaluate system performance. Performance problems are investigated in terms of subsystem performance as well as the system as a whole.

*Conclusions and Future Work*. Chapter 5 presents the conclusions drawn from this work about which subsystems could replace their equivalents in a standard system, which need more work, and what further improvements I recommend based on the experiences with this work. These changes are presented for both the system and subsystem levels. These improvements include alignment, PMT automation, combining the python scripts, and alternatives to the current XYZ stages. Also laid out are the changes to a standard system that could improve its sensitivity.

# CHAPTER 2. BACKGROUND

Capillary electrophoresis (CE) and laser induced florescence (LIF) have been used for many years in the bio-research community. [9], [10] This chapter discusses what CE and LIF are, how they are used, what a standard CE-LIF system design includes, and how much it costs to build an entire standard system using the system in Dr. Woolleys lab as a model. This background is integral to understanding why developing a lower cost alternative system is imperative to enable a broadening of the field of study by those interested in doing research but not having the means/funds to build a system and dive into the research.

## 2.1 Capillary Electrophoresis

CE is an analytical technique that involves using an applied voltage to separate ions in an aqueous solution based on their electrophoretic mobility. Many factors account for how mobile the molecule is that is being measured. These include charge, size of the molecule, viscosity of the solution and the strength of the applied electric field. As the applied electric field strength increases so does the velocity of the molecules. For a given size molecule, the larger the charge the faster it moves. If two molecules have the same charge, the smaller molecule will generally move faster. These charges are controlled for a given biomolecule separation by the pH of the buffer solution used which has a large effect on the mobility of the molecules. The voltage determines the strength of the electric field and controls the velocity of the labeled material down the channel. [9]

By employing capillaries for use in electrophoresis, some of the problems that traditional electrophoresis had were solved. Capillaries increased the surface to volume ratio which eliminated overheating caused by high voltages being applied to the channel which acts as a resistor. The capillaries also add favorably by reducing lateral diffusion and evening out temperature gradients within the capillary.

### 2.1.1 Capillary Electrophoresis Applications

CE is a widely used technique in bioscience across many facets of applications. Some of the applications include DNA fingerprinting, drug analysis, and the characterization of proteins. An advantage is the resolution for these applications. For example, DNA can be separated down to one base pair resolution allowing for high resolution mapping. The high selectivity is also great for chiral separations to only have the effective drug within the compound in the delivery mechanism. Due to the separation mechanism employed, voltage gradients, proteins and other amphoteric molecules can be separate easily by adjusting the pH to increase or decrease mobility of the target or selected against molecules [9], [11].

## 2.2 Laser Induced Fluorescence

LIF, also known as laser-stimulated fluorescence, is a method in which a laser is used to excite a molecule. Following absorption of the light that molecule will go through spontaneous emission to produce photons of a different, longer wavelength. Although the topic of this thesis covers the use of LIF for selective species detection the method is also used to study molecular structure and facilitates the visualization of flow.

### 2.2.1 Laser Induced Fluorescence Applications

LIF has many applications for use including molecule detection. Laser induced fluorescence allows a labeled molecule to be excited with a laser and the emissions collected and quantified. The use cases for this method include presence detection, concentration quantification, as well as size and charge estimation. The ability to detect the presence of a molecule is used in medical applications in order to see if specific molecules are being produced in the body which identify a specific condition of the patient. [12] Concentration quantification is useful if checking for the byproduct of a reaction or if a protein is denatured or not by allowing the percentage of completion the reaction went to or how much of a protein survived an environmental condition [13]. The size and charge estimation is useful in identifying the labeled molecules from each other if for example you have a label that attaches to a common group such as an amino acid. [14] Another use case is to detect how many different charge states are among a single molecule.

### 2.2.2 Label Attachment Method Overview

In the experiments the fluorescent label Alexa Fluor 532 (AF532) was used to detect the presence of the labeled proteins. Dr. Woolley's lab prepared the samples for testing. The samples were prepared in a 10mM concentration and then diluted down from there. The method for attachment of the AF532 label is described in detail by Jena Bioscience. [15] In order to label the proteins, the optimal protein concentration should be 10mg/ml with a minimum of 2mg/ml. Amine-free buffers commonly used for labeling are PBS, MES, or HEPES. The following summary is a simplified version of the procedure.
Experimental protocol

- Create a 1M sodium bicarbonate solution with ultra-pure water.
- Use the 1M sodium bicarbonate solution to dilute the protein solution to a final concentration of 100 mM.
- Prepare 10mg/ml of dye solution by adding DMF. When handling the fluorophore, work under low light conditions!
- Mix the protein solution (10mg/ml) and the dye solution (10mg/ml) in an appropriate vial to achieve 10mM labeled protein.
- Incubate for one hour in a shaker at room temperature. Protect from light!

## 2.3 Microchip Capillary Electrophoresis

Microchip CE, meaning using a microchip device to perform a CE experiment, became popular in the 1990s as an approach to separate biological compounds in a shorter period of time. The microchip devices used can be a variety of materials but PMMA is a common choice do to its ease of fabrication and low cost of materials. [16] Another advantage of the devices is the ability to incorporate multiple tests or procedures into the same chip. This allows for minimal transfer between devices and much smaller sample volumes. Because of this less sample and reagents are used which also saves time and money. Microchips have another feature of allowing creation of a sample plug in the intersection of two channels. To generate that plug the shorter

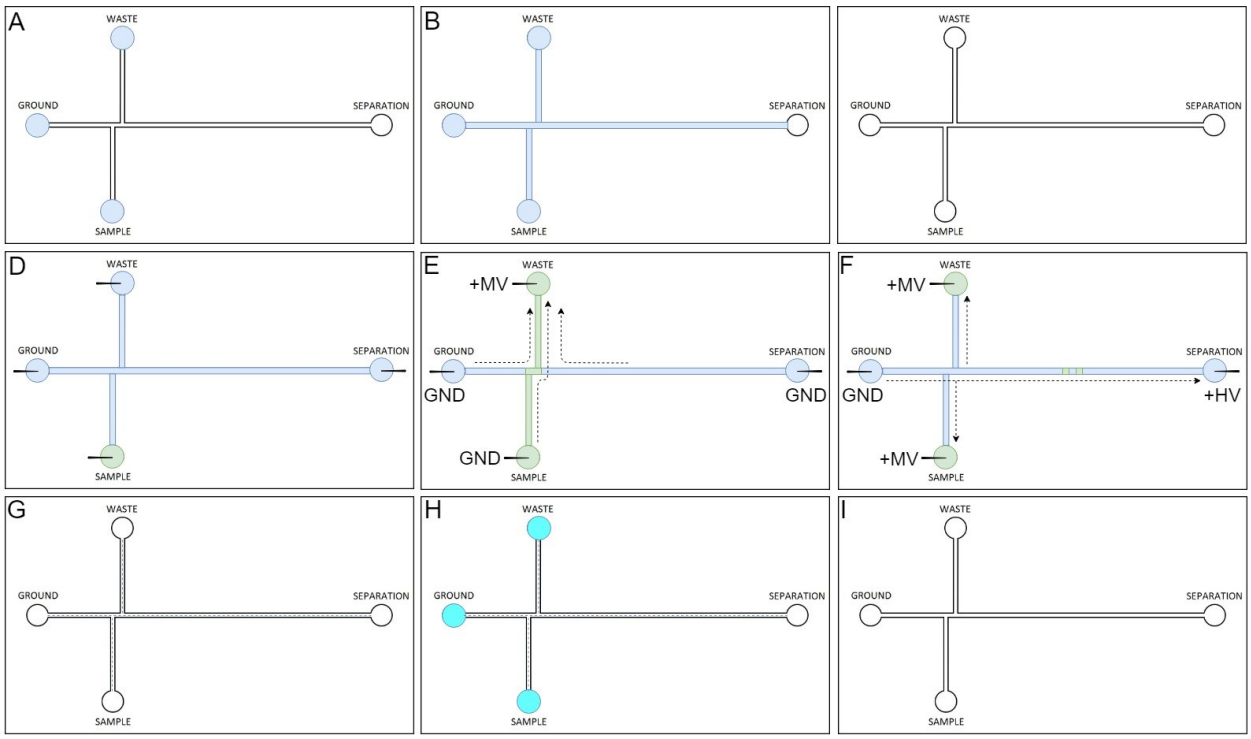



Figure 2.1: The steps taken to run a microchip CE-LIF experiment. A) Fill three of the wells with buffer. B) Use vacuum to pull the buffer into all of the channels and ensure no bubbles are present. C) Empty all wells with a pipet and add equal amounts of buffer or sample to the appropriate wells. D) Attach device to the system and insert the electrodes. E) Run an injection with the voltages configured as shown. F) Run a separation with the voltages configured as shown. G) Unattach the electrodes, remove device from system and vacuum out all liquid. If running again repeat steps A-G. H) Fill three of the wells with DI water. I) Vacuum out the device completely.

cross channel has sample flowed through it using voltages in an injection configuration. Next a separation configuration is used to pull back the sample in the short channels and only allow the plug to flow down the separation channel.

### 2.3.1 Microchip Usage Procedure

When loading and running the microchip in the CE-LIF system the following procedure is the method used for the standard system and as such was adopted for use in our system to eliminate a variable in the operation procedure. Figure 2.1 covers in detail the steps taken to load a device with sample, run an experiment, and then clean the device.

Figure 2.1 shows the procedure in 9 steps that go step by step through the process. Step A is to fill three of the four wells with buffer (blue) in order to coat the channels and reduce the

amount of labeled molecules that stick to the channel walls. Step B shows what the device looks like after using a vacuum attached to the injection well pulls the buffer into all of the channels. Step C illustrates what is placed into each well with three wells getting buffer and the fourth well being filled with the sample (green). Step D is representative of where to place the electrodes in the well when placed within the system setup. Step E shows what voltages are used as well as the direction of flow and where the sample is when the step is completed. Step F shows an example of a separation with the flow in the device that diverges into two plugs as it travels down the channel toward the laser light in the detection window. Step G is figurative of what you should see after vacuuming out the device, small droplets of residual buffer and sample mixture lining the walls of the channels. Step H is very similar to step A in that 3 wells are filled with deionized (DI) water in order to rinse the device to prevent contamination from compounds sticking to the wall when the droplets evaporate. Step I wraps up the process when the DI water used to rinse the leftover buffer and sample is vacuumed out of the device.

These steps of the process allow the device to be used for a few days in a row before the voltages start to deteriorate the channel walls and affect device performance. If these steps are not followed, buffer and sample can dry into the channel walls and create plugs that cannot be cleaned out with DI water and vacuum rendering the device useless.

## 2.4 Standard System Design

A standard CE-LIF system, as found in Dr. Woolley's Lab, contains the major components that are illustrated in Figure 2.2. Those major components are the laser and its driver, the PMT and its controller, the signal path and data collection platform, the optical bench, the device, and the high voltage system. Each component shows what inputs the system needs and what it in turn outputs using arrows to show flow as well as what those inputs and outputs are.

### 2.4.1 Laser and Driver

The laser and its corresponding driver are the start of the signal path. This is the excitation source for the fluorophore in the channels of the device. The driver provides the power and a modulation source that drives the laser diode. The output power stability of the laser is $<1\%$

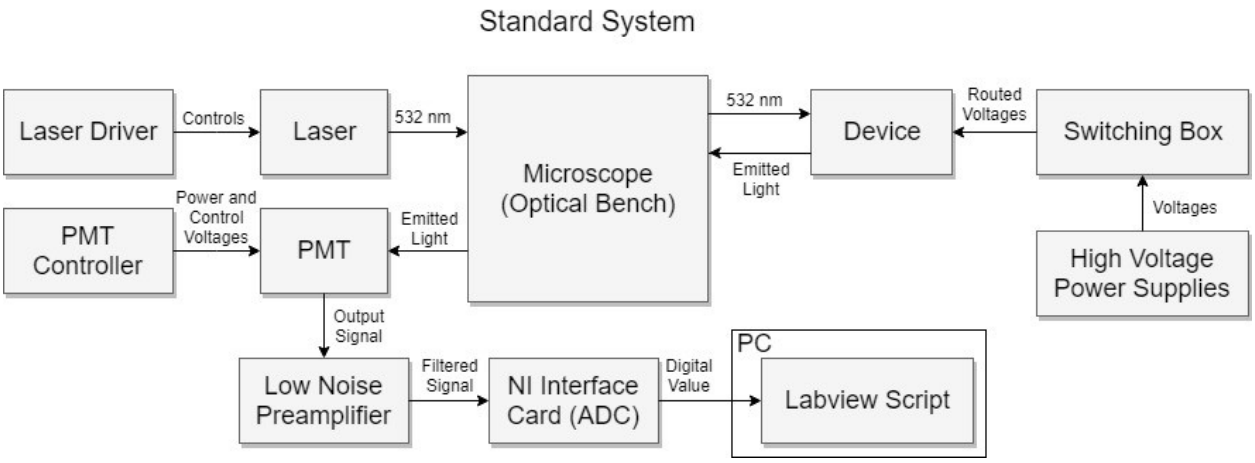



Figure 2.2: Standard system block diagram

variability with a 10 minute warm up time. The laser is in a large metal housing which minimizes variations due to temperature. The laser light is within +/-1nm of the 531.65nm center wavelength. The full width half max of the laser is less than 0.2nm which means the light is spectrally pure.

### 2.4.2 Microscope

The microscope acts as the optical bench of the system. What that means is that the light is filtered and routed to the right place by the mirrors, lenses, and filters within the housing. The microscope is also where the device and PMT are mounted in their respective points in the optical path. Manual controls on the microscope allow for precise alignment of the channel at the focal point of the laser for optimum signal using direct visual observation.

### 2.4.3 Device

The devices used with the standard system described here are made of the materials poly methyl methacrylate (PMMA) or cyclic olefin copolymer (COC). They accept the incoming light and allow the emitted light to propagate with little attenuation to the PMT optical detector. Devices have channels for fluid flow as well as wells for the sample and buffer, and as a connection interface for the electrodes from the high voltage system.

### 2.4.4 High Voltage System

The high voltage system includes three parts: two different high voltage power supplies and a switching box. The high voltage power supplies have a maximum output of $1250V_{DC}$ and $5000V_{DC}$, respectively. These are set individually in order to get the proper ratio for the electric fields in each channel for the correct flow to occur. Each supply is manually set, fed into the switching box, and then routed to the correct location by one of two control switches on the switching box. The first switch controls whether the voltage makes it from the supplies to the second switch. The second control determines whether the box is outputting the voltages in a configuration corresponding to an injection or a separation. From there those voltages are routed to platinum electrodes that are taped in their corresponding wells.

### 2.4.5 PMT and Controller

The PMT converts the collected light emitted from excited fluorescent molecules into an electrical signal (voltage). This signal is sent to a low noise amplifier with built in filtering in order to attenuate unwanted frequencies and amplify the desired ones. The PMT has its own gain control from the controller that supplies the necessary power and control voltages. The gain varies logarithmically as voltage changes linearly.

### 2.4.6 Low Noise Amplifier

The low noise amplifier (LNA) takes the signal from the PMT and processes it. It has selectable filters in order to remove noise from the signal and then a selectable gain in order to increase the signal. This amplifier provides the necessary input impedance and output impedance to minimize loss in the system.

### 2.4.7 LabVIEW Script and NI DAQ

The NI Data Acquisition (DAQ) device is how the output from the LNA is linked into the computer. The NI DAQ has an analog to digital converter (ADC) inside that samples the output from the LNA and converts it into a digital value that is sent over USB to the LabVIEW script.

The script then interprets the value and then both plots it real time and stores the data in a file for the given run. The stored data can be reviewed and plotted using Excel.

## 2.5 Standard System Cost

Each of the components described in the previous section had an acquisition cost which adds up to the large overall cost of the standard system. This thesis hypothesis is to determine if a system can be built for a much lower cost with the same performance. In order to know that the system is a lower cost a baseline cost of the standard system is laid out in Table 2.1. These prices are for the components used in the system located in Dr. Adam Woolleys lab. The prices were looked up in October 2017 and were significantly lower than when the system was originally bought.

These costs are meant to be representative of a typical system. One of the main things that came as a surprise was the estimate given on the system cost and the actual cost were vastly different. Table 2.1 was compiled after the system was tested resulting in a total cost of approximately $40,000 including shipping. The prices above are only for the components and equipment and do not include the shipping costs.

## 2.6 Conclusion

With this system as a basis, the next chapter will discuss the designs and problems encountered as a system of lower cost was developed. Although some subsystems were broken up and/or combined to form new subsystems, the resulting overall system was designed with the intent to be able to produce results of the same quality and have similar functionality to that of standard system.

Table 2.1: Cost of Standard CE-LIF System

| | | | |
|---|---|---|---|
| | Laser | | |
| 1 | Laser + Driver | $1,004.00 | Laserglow Technology |
| 2 | Laser Height Adjustment | $500 | Estimate |
| | Total | $1,504.00 | |
| | | | |
| | Optical Bench | | |
| 3 | Optical Table 4’ x 8’ x 8.3” | $7,870.00 | ThorLabs |
| 4 | Ziess Axio Observer A1 | $16,526.00 | Bartels & Stout |
| 5 | 100um Pinhole | $66.00 | ThorLabs |
| 6 | PMT - Hamatsu H10722-20 | $1,114.00 | Hamamatsu |
| 7 | PMT Power Supply C10709 | $1,510.00 | Hamamatsu |
| | Total | $27,086.00 | |
| | | | |
| | High Voltage | | |
| 8 | PS310 Power Supply | $1,495.00 | Stanford Research Systems |
| 9 | PS350 Power Supply | $1,495.00 | Stanford Research Systems |
| 10 | Custom Control Box | $100.00 | Chemistry Support Shop |
| | Total | $3,090.00 | |
| | | | |
| | Signal Path and Data Acquisition | | |
| 11 | SR560 - LNA | $2,595.00 | Stanford Research Systems |
| 12 | NI USB-6212 | $1,257.00 | National Instruments |
| | Total | $3,852.00 | |
| | | | |
| | Software | | |
| 13 | LabView Full License | $2,999.00 | National Instruments |
| | Total | $2,999.00 | |
| | | | |
| | System Total | $38,531.00 | |

# CHAPTER 3. SYSTEM DESIGN AND IMPLEMENTATION

In order to build a comparable system to the standard one in Chapter 2, there are some essential pieces that had to be included and some non-essential pieces that could be modified or removed. Each of the subsequent sections of this chapter discusses the decision making and design process used to create the system in its present state. In Figure 3.1 below, the subsystems are laid out as well as how they interface with the other parts of the system.

Notice that the heart of the standard system, the microscope, is missing from the figure. In order to keep the costs low, the microscope had to be replaced. The microscope portion of the standard system was replaced by the following subsystems: Optical Bench including optics, sample holder, PMT holder, and two sets of XYZ stages for the PMT and sample holder.

The arrows in Figure 3.1 show the flow of light, voltages, and data throughout the system. The following sections discuss each subsystem's functions, problems encountered, design changes made to overcome problems, and the subsystem design that became part of the final implementation used in the system. The sections in order are power supplies, high voltage system, devices, 3D printed components, laser and driver, optical components, XYZ stage, photosensor module, Arduino shield, microcontrollers, Python keylogger, Python sampling and graphing, and system cost.

## 3.1 Power Supplies

In order to reduce the cost of the system, power supplies were one of the first things targeted. The reasoning for this is that the power supplies are pervasive throughout the system in their effect. They drive the high voltage system, PMT, motors, a motor driver circuit, microcontrollers and periphery circuitry. The power supplies in this system are working to lower the cost of lines 1, 4, 7-9, 11 and 12 in Table 2.1 Even though the amount of cost reduction is a small percentage of the whole it allows for the customization of the system while not being limited to fixed voltage

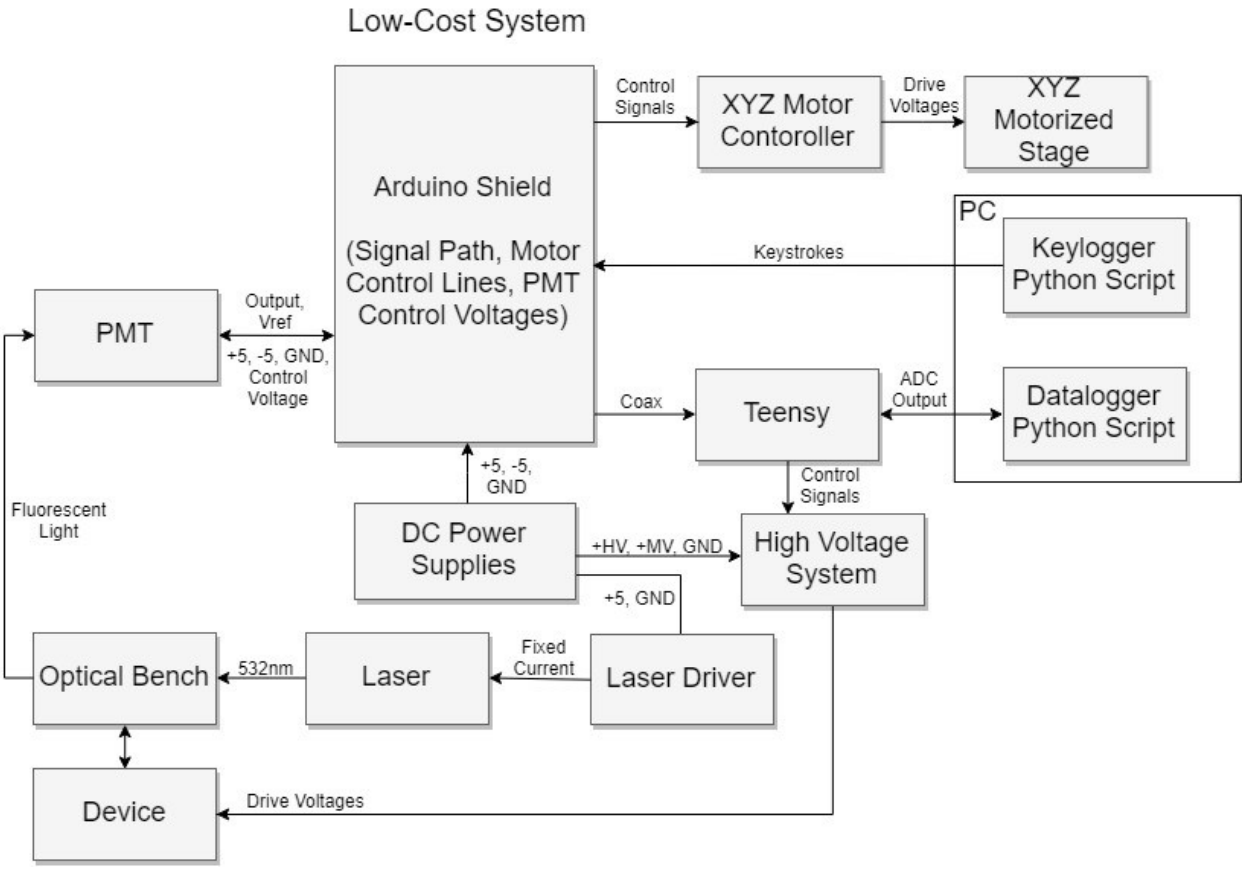



Figure 3.1: System block diagram

supplies. There are only two issues uncovered when using multiple power supplies that will be addressed in Section 3.1.1. Section 3.1.2 discusses why each power supply was selected. Section 3.1.3 covers alternative power supplies that could be used as replacements to bring the cost down further.

### 3.1.1 Problems Encountered with Initial Power Supplies

The first of the two issues uncovered was the stability of the power supplies. The stepper motors need around one ampere each to run but have a high initial current spike when motion is initiated. The power supply that was powering these motors was an ATX computer power supply. This high spike and sometimes the subsequent inductive spikes while operating would cause the power supply's resettable fuse to trip and stop operation. This would cause the Arduino to have to be reprogrammed and the Python script to be restarted in order to reestablish the connection.

The second issue was ground not being at the same potential for the microcontrollers and power supplies. The microcontrollers are both connected via USB to the computer and its power supply grounding system. When connecting the power supplies' grounds together with the computer they were not at the same level and caused a significant current draw. Even though they were all plugged into the same wall circuit this difference in voltage caused the motor drive circuit and the microcontrollers to have trouble communicating.

### 3.1.2 Solutions to the Problems

In order to correct the power supply shutoff problem the supply needed a current limiting circuit instead of a resettable fuse. When looking for alternatives to the ATX power supply system, the HP 6237BA Triple Output Power Supply was selected for its individually controlled channels and sufficient current output. The overcurrent issue that was occurring with the ATX power supply no longer occurred with the linear power supply used to replace it.

The issue of different ground potentials was solved by adding a ground lift into the circuit. A ground lift is a circuit or component that removes the common earth ground reference. The component used was a three to two prong adapter. Removing the reference to Earth ground from the power supply providing power for the motors, allowed the Arduino and computer to be able to interface and send commands to the Big Easy Driver boards. With the linear power supply disconnected from Earth ground and the computer power supply referencing Earth ground, the circuit was able to function as normal.

### 3.1.3 Future Improvements

The only issue with this method we later found out is that it does not abide by the university's code for electrical wiring. By removing earth ground the power supply case may end up becoming charged and run the risk of electrical shock and/or fire. These potential hazards are the reason for the electrical code. If your location has similar regulations then a custom solution would need to be developed for either the data connections to the computer or for the connection to the offending power supply. An example would be a wall plug that uses an isolation transformer to

provide the necessary electrical protection as well as the separation from earth ground preventing a differential between grounds.

## 3.2 High Voltage System

The high voltage system drives electro-osmotic flow and electrophoretic motion within the microfluidic device. In the standard system, they have off the shelf power supplies on lines 8-10 of Table 2.1 that fulfill this function. In order to reduce the cost of this subsystem, a single PCB version was developed to fulfill the same role. The system's role is to have two stable but adjustable high voltage outputs to tune the electric fields in the device to the necessary ratio and strength in order to facilitate a separation in the channel. The voltage stability and ratio determine the fidelity and integrity of the resulting output signal from the fluid sample plug which travels down the separation channel. If these two things are working it will keep the separated peaks sharp and prevent both plug pull-back (diminishing/shrinking) and weak pull-back (injection channel leakage) which convoluted the separation by leading to smaller, broader, and/or extra peaks (aka noise). These voltages need to be routed to the different wells at specific times in order to achieve the desired electric fields within the device for successful injection and separation. The routing and timing is all part of the high voltage system.

The high voltage system revolves around the EMCO F40 unit as the high voltage source. The EMCO F40 is a proportional DC-DC converter which takes an input voltage of 0-15V and using a frequency-based boost topology converts it to 0-4000V. This range was selected in order to cover the operating range of voltages the device can handle. Although other companies make converters and there are other models this model was chosen because it was inexpensive and available on eBay. Afterwards, it was noted that its documentation is much more thorough. In order to understand the changes made to create the final high voltage system, section 3.2.1 discusses the problems encountered with the original two designs and why a redesign was necessary.

### 3.2.1 Problems with the Starting Point System

When initially looking at the two versions of the high voltage system created by the senior projects, four issues surfaced that needed to be addressed in order to create one implementation

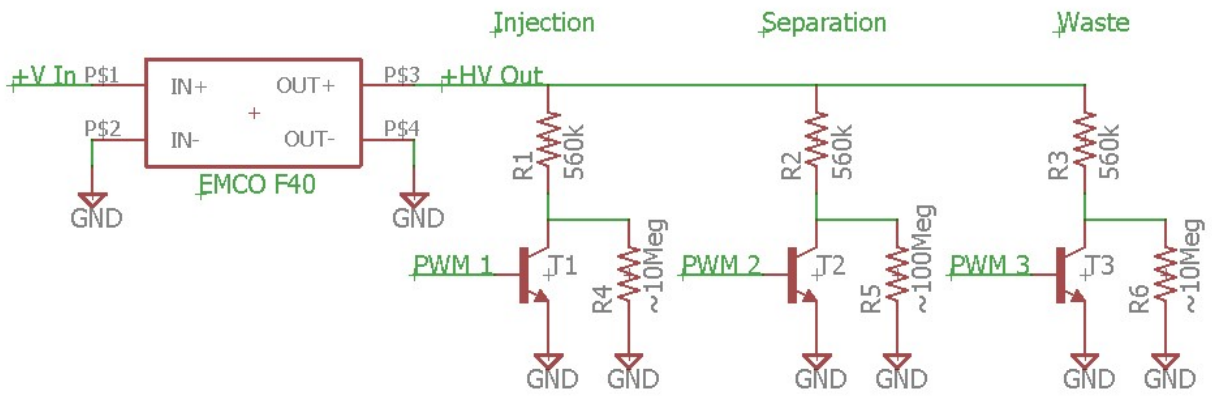



Figure 3.2: Initial high voltage circuitry

that could be usable as a stable high voltage subsystem. Each of these issues affected the output voltage stability, both in value and ratio, which led to issues when trying to run a separation. The issues that needed to be addressed were the two different negative effects of using a PWM control, switching issues that arose when using transistors or resistors, and issues that occurred based on how you power the system.

The first issue encountered involved using PWM to reduce the EMCOs output voltage. The PWM control was implemented on a fixed DC supply to turn on and off the EMCO unit. The goal was to create an effective voltage on the EMCO's input to generate a constant voltage on the EMCO's output. The issue with this is the typical 260ms response time of the circuitry inside the EMCO. This is equivalent to how long it takes to get up to its output voltage for a given input voltage. When trying to use PWM the EMCO is starting that time all over every time the pulse goes high which leads to an unstable and inconsistent output voltage.

The second issue encountered involved using PWM to lower the voltage being applied to the channel. Figure 3.2 shows how the common emitter stage was connected to the channel. The resistors R4, R5, and R6 represent the channels that have the voltage applied to them. Although it did provide a reasonably stable output voltage the settling time was the major issue with this implementation. The output voltage took over two seconds to be within 50V of the final value of the channels equating to approximately five time constants. When you estimate the capacitance of the transistor, the only unknown in the equation for tau you get a value of approximately 715nF. The only factor not taken into account in these calculations is the effect of PWM on the transistors

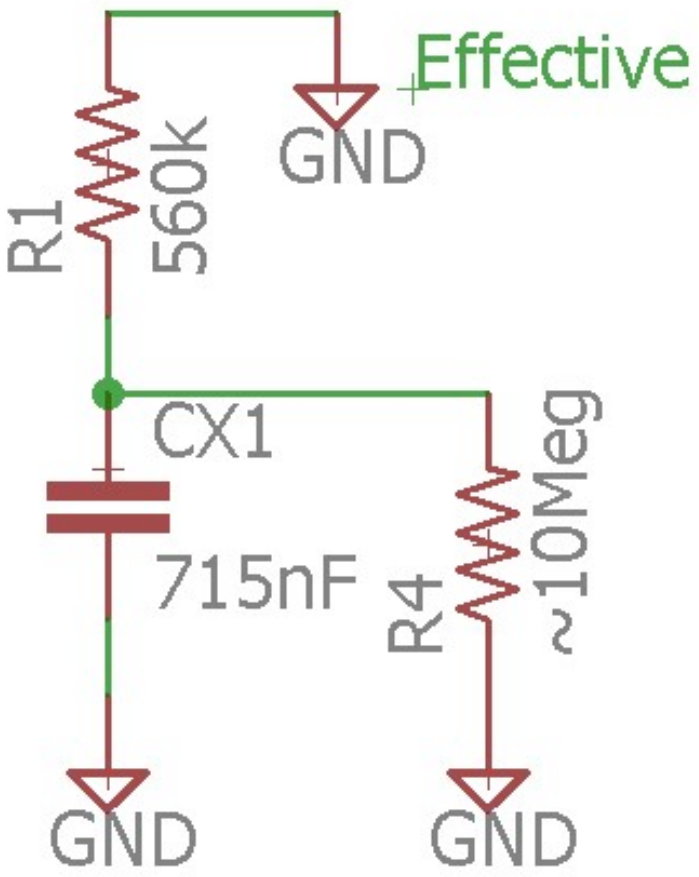



Figure 3.3: Equivalent injection channel circuit

capacitance as modeled in the equivalent circuit in Figure 3.3 as the capacitance of the device was unknown as well. The equation below shows the settling time equation as represented by tau.

$$\tau = R * C \tag{3.1}$$

The third issue comes up when trying to obtain the same voltage on different channels from the same EMCO output. Figure 3.2 shows how the device is hooked up to the EMCO and each channel. With each line switching at slightly different times the amount of total resistance the EMCO sees as a load varies. This varies the output voltage reaching each of the channels. In order to obtain the same voltage on two equivalent channels, you would need different pulse widths for each of those channels. The reason for this is every time the PWM signal turns on the resistance of that branch drops which draws more current. There was no load other than the small

current through the channels on the output of the EMCO so when you short the 560kOhm resistor to ground the current draw from the EMCOs output is significantly higher. This is due to the lower half of the voltage divider having a much larger resistance than 560kOhms.

The fourth issue also concerns PWM but it is what physically manifests itself on the device. When using the EMCO output into the device, the constant switching exhibited as bubbles forming on the platinum electrodes that are inserted in the wells. The interface between these electrodes and the buffer and sample in the wells in which they are placed acts as an electrolysis chamber generating gas bubbles of the byproducts of the fluid. These gas bubbles were pulled into the channel more often than not which led to a blocked channel. If the voltage is left on for too long after a bubble stops flow it leads to destroyed channels either by burning or some other failure mechanism.

### 3.2.2 How Problems Were Overcome With the New Design

In order to overcome the problems discussed in the previous section, there were a few major design changes that needed to be made. The changes were selected in order to have the greatest effect on the four issues discussed in the previous section. The changes made were to remove PWM entirely, use two EMCO power supplies and to use relays instead of transistors for switching.

The first change made was getting rid of the PWM signal. By removing the PWM controls and replacing the controls with DC voltages to run the system the bubbles discussed in the fourth issue that used to form at the platinum wire and fluid interface inside the well no longer showed up. This change did induce a redesign of the circuit in order to get two consistent and different voltage values by adding a second EMCO F40 unit.

By switching to two EMCO F40 units the issue of using PWM to lower the output to two different voltages was removed. Each EMCO was set to the desired output value by supplying the required input voltage. This resolved the issues caused by using PWM to drive the EMCOs input. The other change was switching to relays which removed extra resistance and capacitance in the circuit required when using transistors to switch the high voltages. Removing excess components eliminated both the settling problem and differences in channel voltages that were supposed to have the same potential. Although this corrected the voltages, it introduced a stability problem on

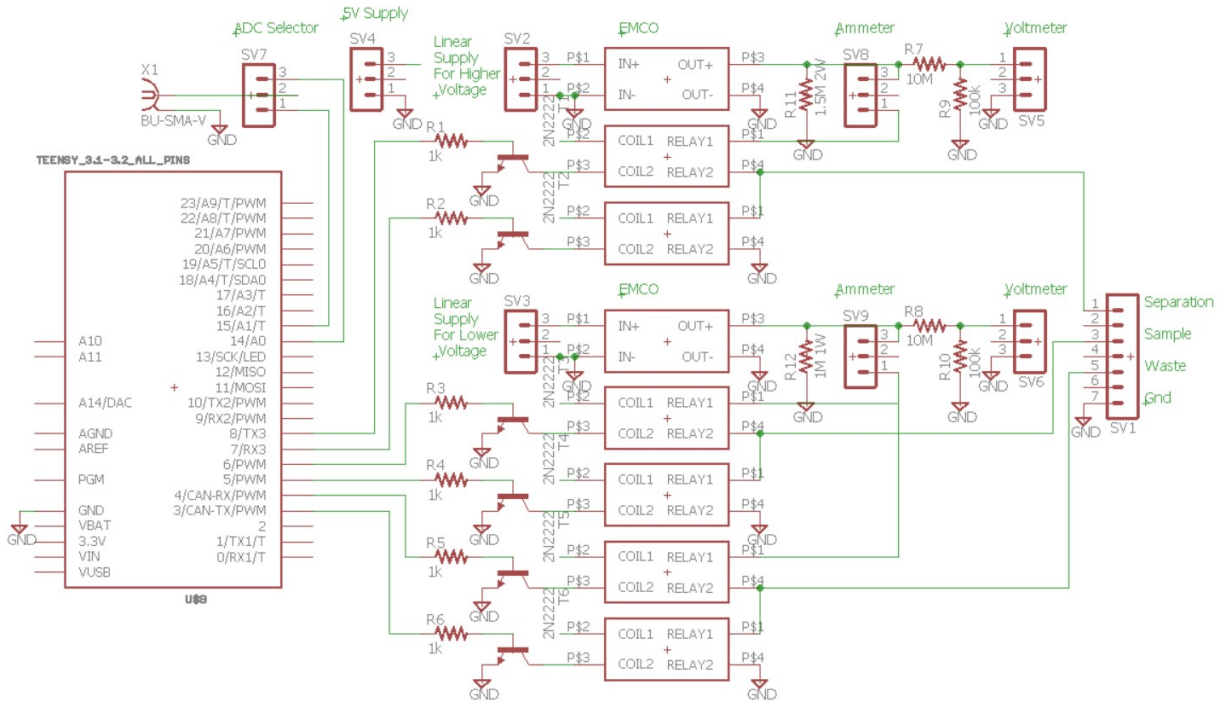



Figure 3.4: Schematic of high voltage system

the output. In order to correct this new issue that arose, a load was added to the EMCO to transition from the no-load condition to a more stable condition while under load.

### 3.2.3 Final Design

The final design used is diagrammed in Figure 3.4 and includes a dual variable DC power supply connected to SV2 and SV3, two EMCO power supplies, six high voltage reed relays, a Teensy 3.2 , and supporting circuitry. The dual variable DC power supply was selected over a fixed power supply to give the adjustability to the output voltages. A dual supply allowed the higher and lower voltages to be set separately. The output of each EMCO unit was then connected to a resistive load, a voltage divider and through a relay to the channel(s). The resistive load was added to provide output stability as seen in the Load vs No-load graph on the datasheet. These loads and the multimeter connection points are not shown in Figure 3.6 but can be inferred when comparing with the schematic in Figure 3.4. In practice, the voltage only drops a few volts when the channels get connected. The load added to the higher voltage EMCO was 1.5Meg 2W (R11) and the load added to the lower voltage EMCO was 1Meg 1W (R12). The resistive divider was

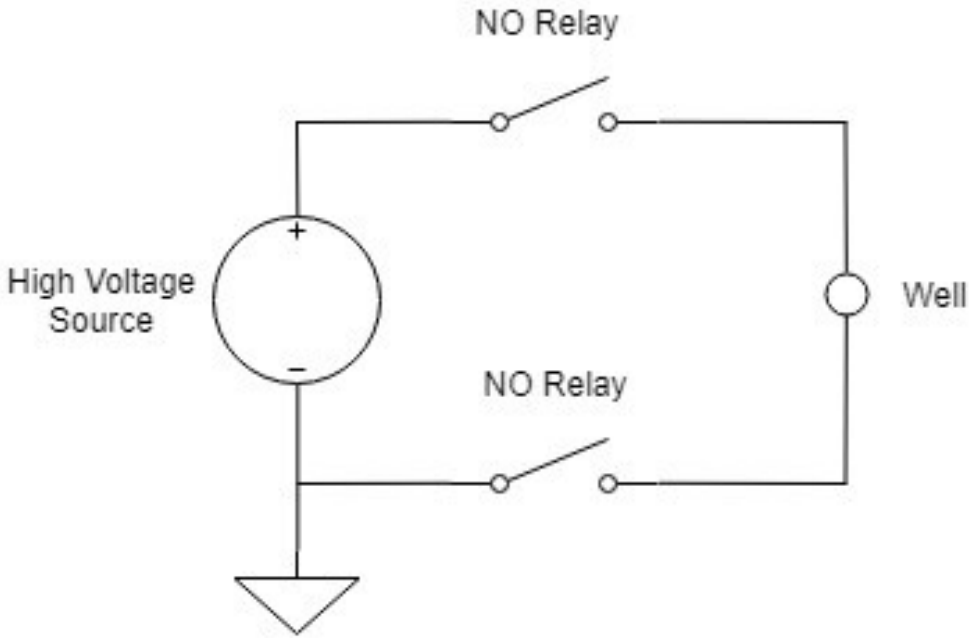



Figure 3.5: Well voltage potential switching schematic

added to have a place to tap off a lower voltage for the real-time monitoring of the voltage by a digital multi-meter (Connected to SV5 and SV6). The voltage divider brings the voltage down by a factor of 1000 (R7-R10). Digital multi-meters (Connected to SV8 and SV9) were also placed in series between the output and channel(s) to monitor current as a device safety precaution and a form of bubble detection within the channels.

The high voltage reed relays (COTO Technology 5501-05-1) are controlled by the Teensy 3.2 through the +5V supply connected to SV4 and a BJT transistor (T1-T6 2N2222A) acting as a switch. This provides the 125mA the relays need while only using 2.6mA from the microcontrollers digital control pin. A limiting resistor is placed in the base in order to provide protection for the microcontroller in case of transistor failure. There are two single-pole-single-throw (SPST) relays per well. Using the relays, three wells can be individually switched between their high voltage setting and ground for the right configuration for the three settings of system off, injection and separation. This switching mechanism is detailed in Figure 3.5 below for a single well.

From the datasheet for the EMCO F40, it talks about a capacitor on the input to provide voltage stability. A 10$\mu$F cap across the input to each EMCO added a preventative measure against input noise, stabilizing the output voltage significantly.

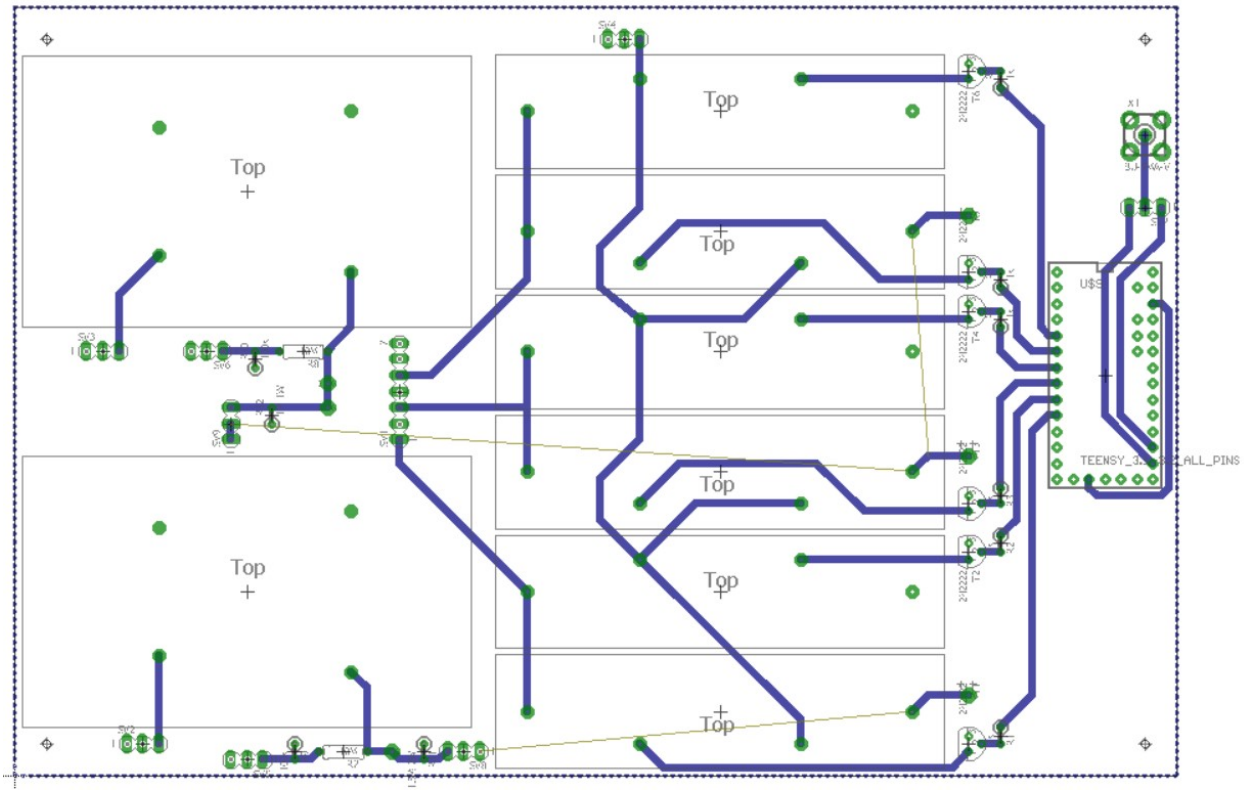



Figure 3.6: Layout of the high voltage system

The Teensy 3.2 provides the drive signals for the relays according to input signals received from the controlling PC.

### 3.2.4 Recommended HV System Improvements

The high voltage subsystem has a few areas that can be improved. Real-time feedback to control the output voltages would provide the most stable output voltages. Adding that control into the Python GUI would allow for easier voltage changes that could be entered into the console and not have to calibrate with a knob on the power supply when the voltages need adjustment. If you do not need to go up to 4kV for your system then switching out the EMCO to a lower maximum voltage model would allow for stable voltages lower than 300V. The EMCO has a minimum output voltage that is approximately 7.5% of the max rated output voltage according to the datasheet in Appendix F.

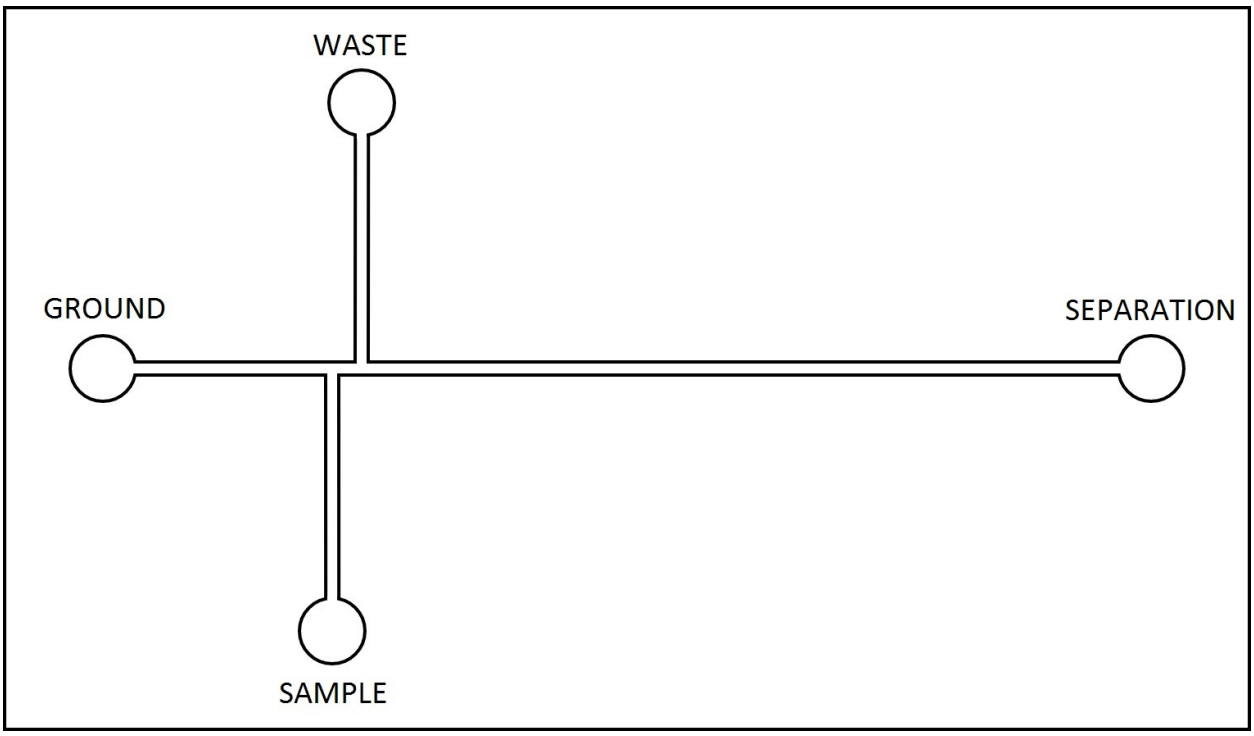



Figure 3.7: Device design - top view

## 3.3 Devices

The microfluidic devices used have the pattern shown in Figure 3.7. This channel pattern allows for a larger plug of the sample to be separated as it goes down the separation channel as described in Chapter 2. The fabrication method for the devices used is called hot embossing and is described in section 3.3.1. The main issue with the devices was delamination where the two pieces of PMMA would separate enough to render them useless.

### 3.3.1 Fabrication Method

The process for creating a microfluidic device is depicted in Figure 3.8. Using two brass pieces and two glass slides to make up the two sides two pieces of PMMA are clamped with six c-clamps to provide pressure. One of the pieces is clamped with a silicon template of the channel. After being heated in the oven to 135 °C and given time to flatten and imprint they are allowed to cool. After removal from the clamps and glass slides the imprinted channel side of the one PMMA piece is clamped against the non-embossed PMMA piece using the same method above.

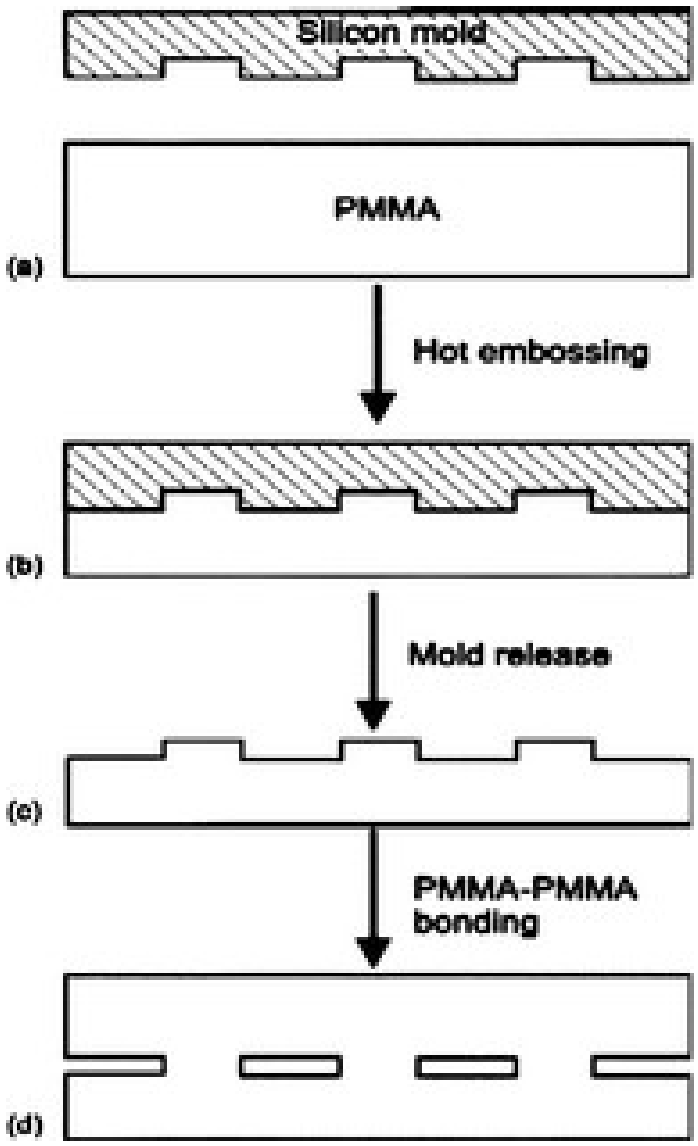



Figure 3.8: Hot embossing steps. (a) Align silicon template with the PMMA to be patterned. (b) Hot emboss the PMMA to create channels. (c) Release the silicon mould from the PMMA. (d) Bond the PMMA embossed piece to the other PMMA half of the device. [17]

This fixture is then placed in the oven at 110 °C. After heating the fixture is then pulled out and allowed to cool before unclamping. The device is then sealed with acetonitrile along the seam between the two pieces and checked for bonding of the two PMMA halves around the channels under a microscope.

### 3.3.2 Fabrication Issues

Due to fabricating the devices by hand mistakes could be made. The mistakes that were noted in the devices fabricated during this project were unsealed devices, cracked glass that caused roughness, and sealed channels within the device. All of these failures could have been minimized with more training or experience in device fabrication.

## 3.4 3D Printed Componenets

In order to keep the costs as low as possible, 3D printing was employed for the parts that needed to be lightweight, required extensive machining, or had interior features that could not be machined using conventional methods. The parts that were required to be lightweight were the sample holder and the PMT holder. The linear stages had difficulty supporting machined versions and provide consistent and repeatable positioning. The part that required extensive machining and had interior features unable to be machined was the optical bench with its optical dumps.

### 3.4.1 Summary of Components' Function

The sample holder shown in Figure 3.9 mounted to the motorized XYZ stage is where the device was taped into place. It is positioned above the optical bench to align the focal point of the laser to the desired detection point in the channel. It needed to be able to have a minimal separation between the bottom of the device and the optical bench to have the laser focus within the channel. The sample holder is mounted to the motorized XYZ stage and has to be rigid enough to not bend while the wires are connected to the wells of the device.

The PMT holder shown in Figure 3.10 has the PMT mounted into its channel. The PMT holder was held in place during operation by the manual XYZ micrometer stages behind it. This structure had to be rigid so that the PMT would not shift out of alignment after calibration.

The optical bench is where the laser, optical filters, lenses, and light dumps are mounted. This part is securely fixed to the optical breadboard to prevent movement. The optical bench needed to be rigid to prevent bending, have ways to contain reflected light from corrupting the sample signal and have slots sufficient to hold the components in place. This printed part is depicted in further detail in Section 3.4.4.

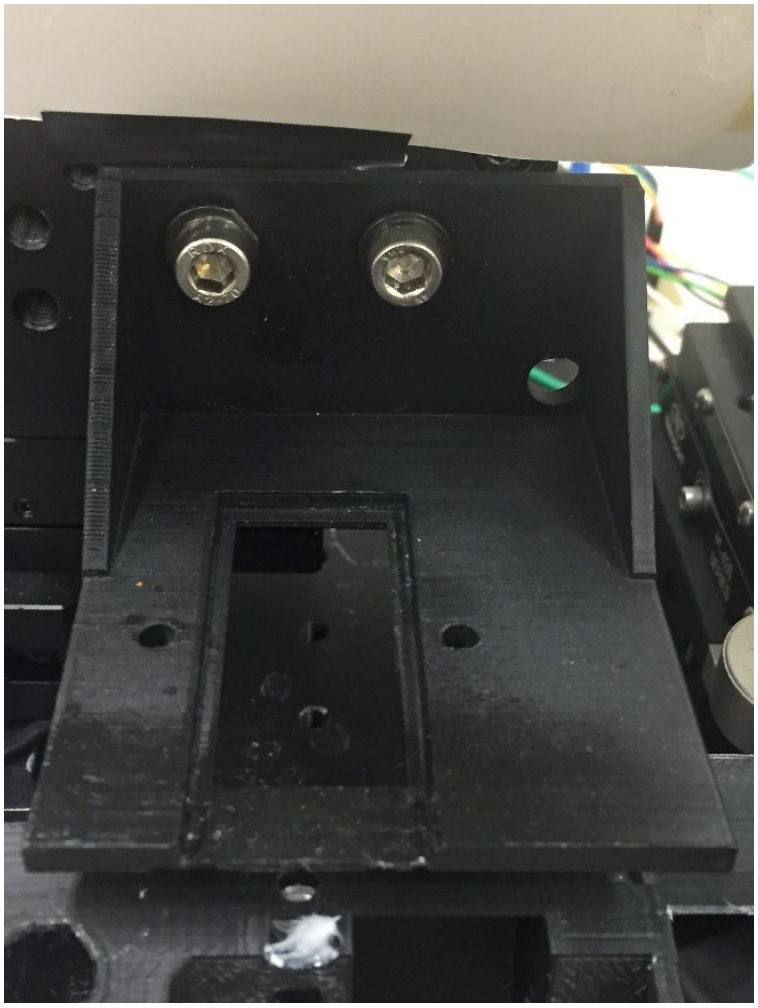


Figure 3.9: Sample holder in the completed system

### 3.4.2 Sample Holder Design Implemented

Figure 3.11 shows the final design that was implemented for the sample holder. It has holes in the thick back plate that are sized and spaced to mount to the XYZ motorized stage. The recessed slot is where the device is slid in and taped in place. There is enough space on either side of the device to tape the leads to so they stay securely in the wells. The only issue incurred during building was getting the base that the device rests in to be flat after 3D printing due to warpage in some prints.

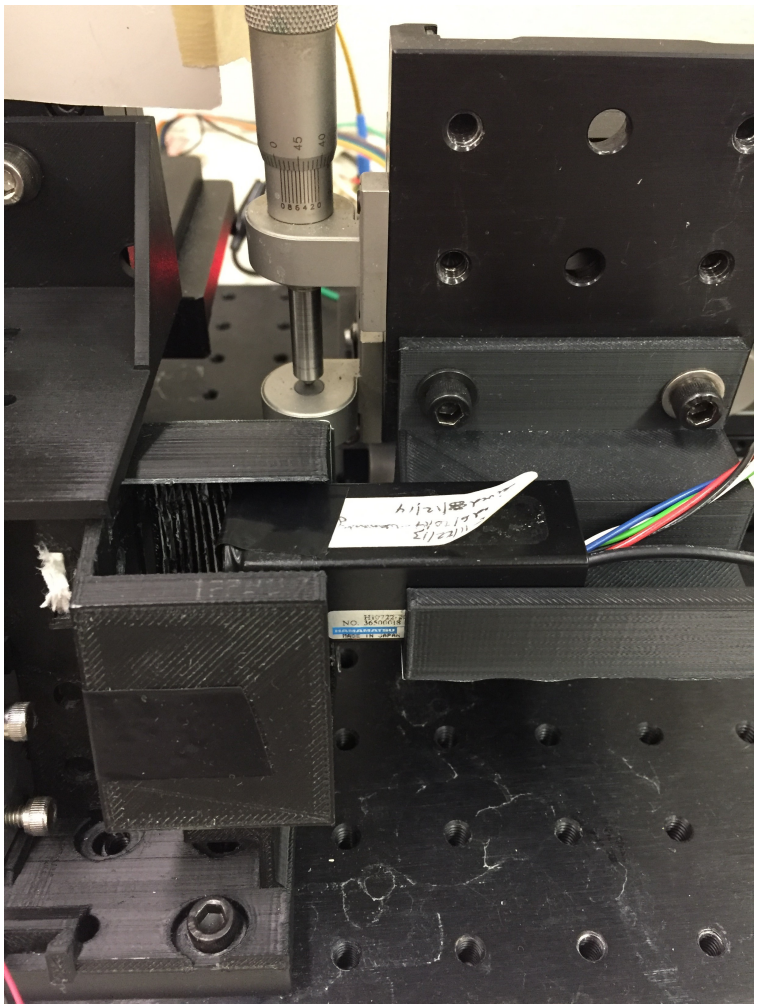


Figure 3.10: PMT, PMT holder and manual micrometer stages

### 3.4.3 PMT Holder Design Implemented

Figure 3.12 shows the final design implemented for the PMT holder. The holes in this part were spaced to be able to mount to the micrometer XYZ stage. The part was rigid and held up well to the weight of the PMT. The only real issue was that in order to interface with the optical bench the PMT had to be halfway out of the channel as shown in Figure 3.10.

### 3.4.4 Optical Bench Design Implemented

Figure 3.13 shows the 3D printed version of the optical bench with each piece labeled with a number. Each of the components mentioned in this section will be discussed in greater detail in section 3.6. Slot 1 is for the dichroic mirror. Slot 2 is for a neutral density filter. Slot 3 and 8 are

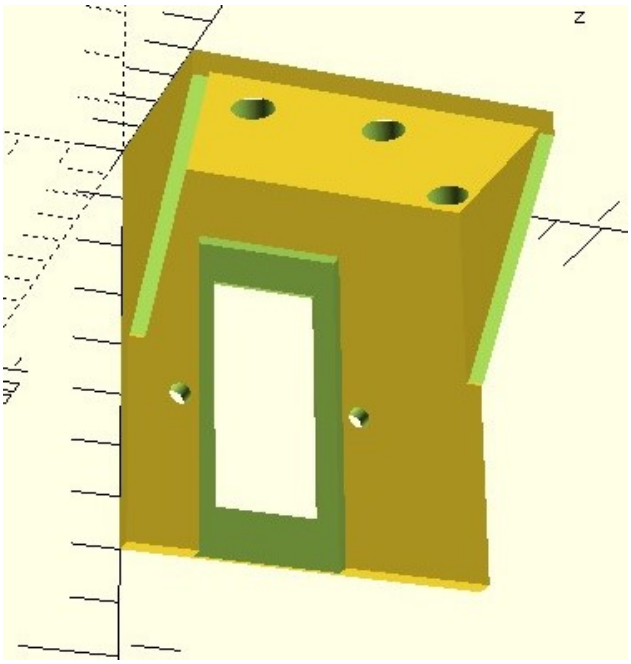



Figure 3.11: Sample holder OpenSCAD render

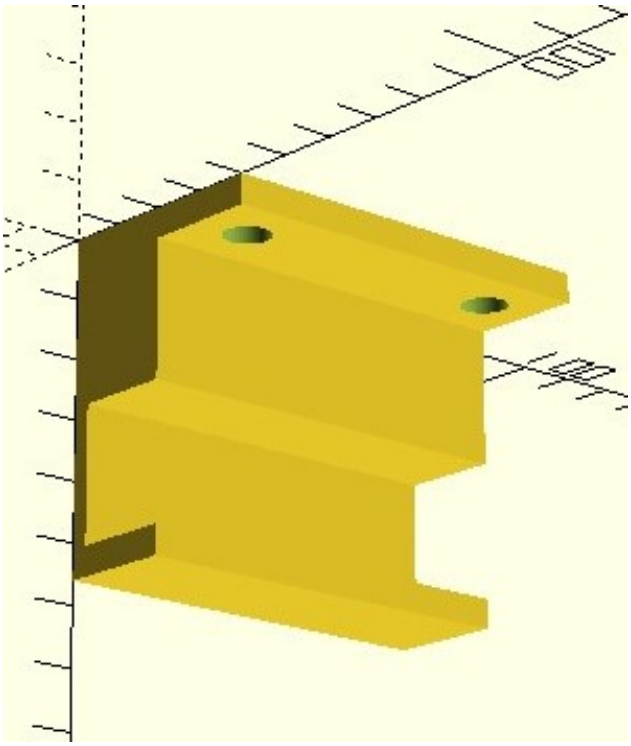


Figure 3.12: PMT holder OpenSCAD render

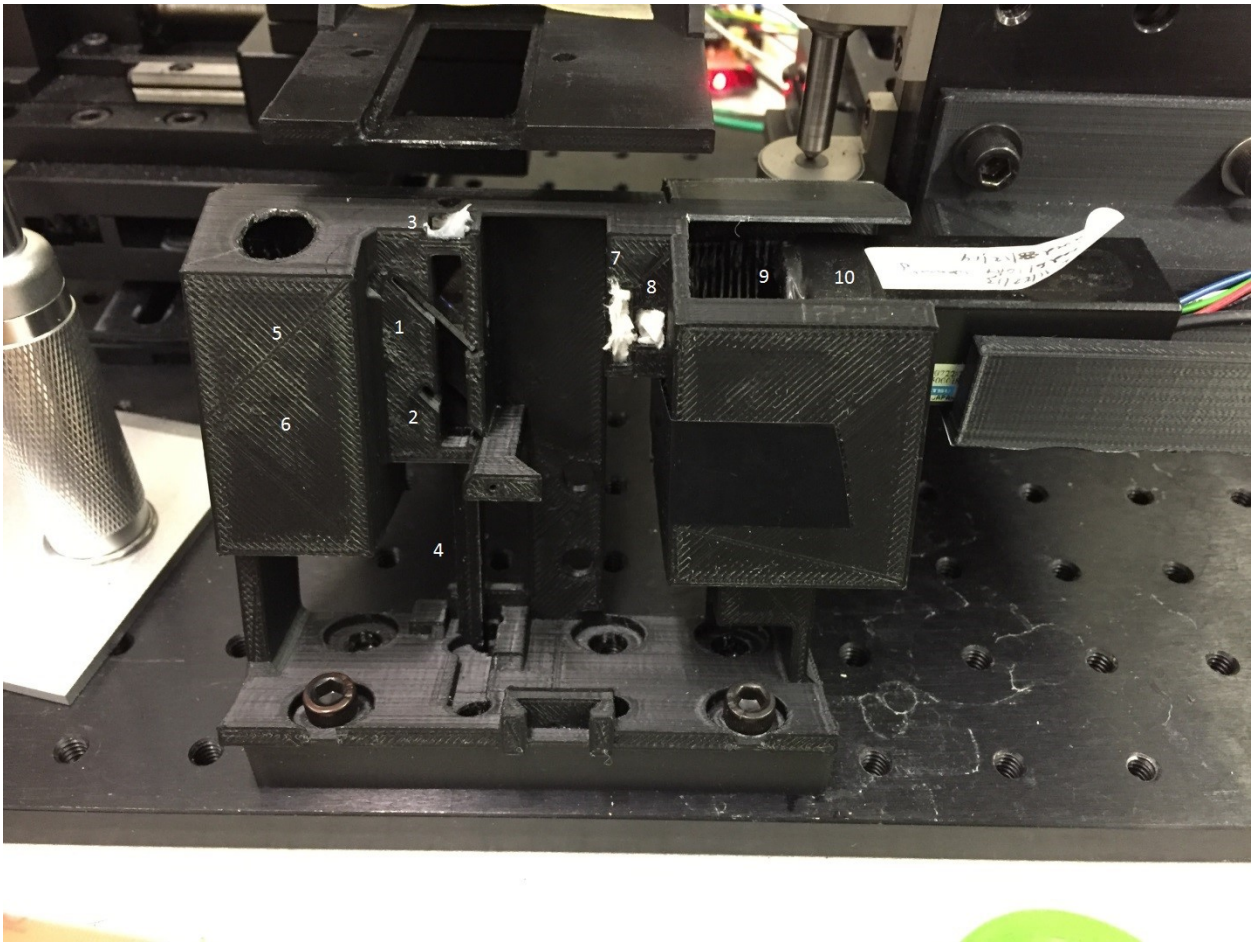



Figure 3.13: Optical bench printed part

for lenses. The trough labeled 4 is where the laser and milled aluminum heat sink mount to align with the optical path. In the part labeled 5 and 6 are two optical dumps for the reflected light off of the components in slots 1 and 2. Slot 7 is for the longpass filter. Component 9 is a pinhole and is affixed to the PMT, component 10. The white silicone glue affixed parts 3, 7, and 8 in place to keep the beam path in focus and on course. The lens in slot 3 both focuses the laser light onto and collects the emitted light from the channel. The lens in slot 8 focuses the light through the pinhole and onto the detector (PMT). The whole print is tilted back at a 23.6-degree angle from vertical in order to prevent direct reflection of any laser light off of the device and back into the optical path. This is a key innovation to reduce optical noise.

The major issue with this implementation was all the openings in the optical bench. These openings let stray light into the optical path which corrupted the PMT signal with noise. Minor

issues showed up when trying to calibrate the system. Alignment of lenses and filters proved difficult. In the end, the lenses were glued in place with silicone because aligning and then packing with filler would cause the lens to shift while filling. Aligning the PMT was another issue because the PMT had to be halfway out of its holder to reach the focal point of the optical path.

### 3.4.5 Recommended Design Improvements

In order to correct the issues presented in the previous subsections, here are the suggestions that proved to be the best based on initial proof of concept tests. By adding coverings to the optical bench and enclosing the optical path more the stray light getting into the PMT decreased significantly. In a redesign of the PMT holder and the optical bench receptacle that the PMT slides into a more stable system for the PMT alignment can be realized. The next recommended change is to make a better method for mounting the optical components within the bench. Having an easy way to secure the small components or upgrading to larger and easy to mount components will allow for a much quicker calibration and swap out the procedure in event of a failure or becoming misaligned.

## 3.5 Laser and Driver

The laser is one of the most important pieces of the whole system. The laser provides the excitation wavelength required with enough power to excite the fluorophore in the beam path. Without a quality laser, beam integrity cannot be ensured as it travels through the optical path, laser intensity may be unsteady, and the green laser's spectrum could be dual frequency if not properly filtered. All of these issues were present so redesigning by going back to the fundamentals of lasers was the approach taken. The initial design uses an Aixis 532nm 5mW laser module which was kept for all iterations due to its low cost ($42 + shipping).

### 3.5.1 Problems Encountered with the Initial Design

The initial design used Bi-Polar Junction Transistors (BJT) which are current controlled current sources. The first issue with the method implemented is that it does not have temperature compensation. The second issue is any voltage fluctuation in drive voltage causes fluctuations in

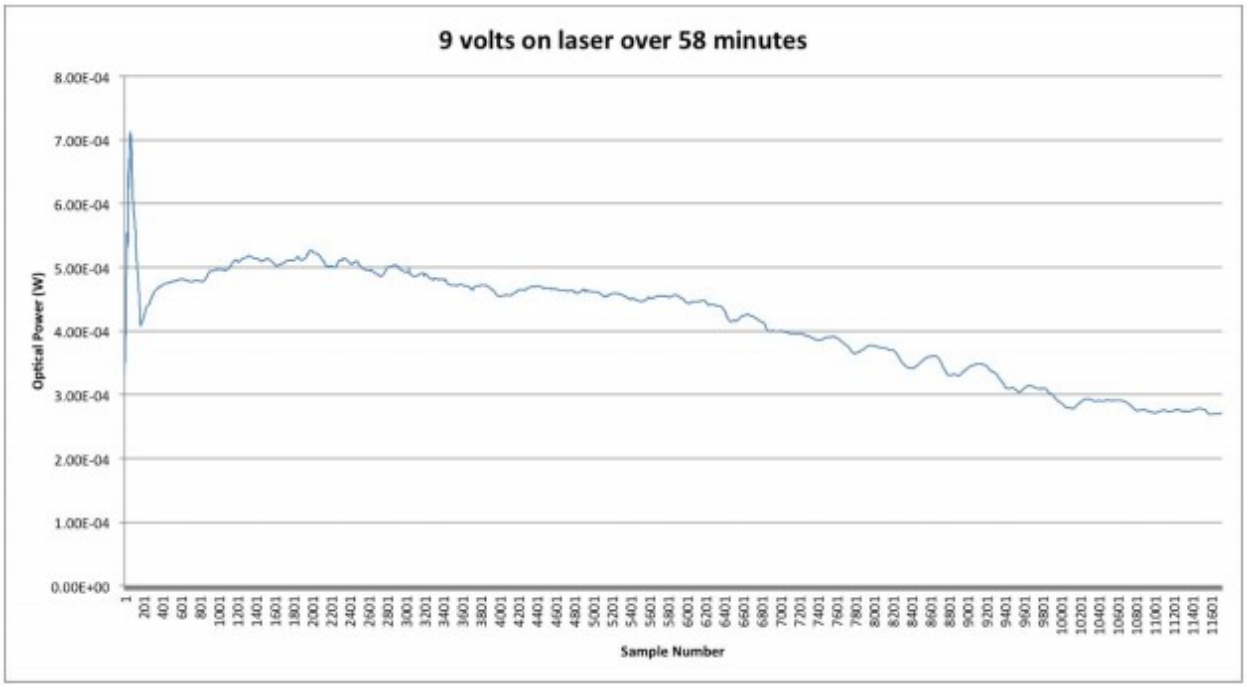



Figure 3.14: Laser power sampled over 58 minutes of samples

the input current which directly affects the output power swing by up to 50% of the rated laser output power. This voltage controlled system's output is illustrated in Figure 3.14 which shows the power output when controlled by a voltage source over the course of 58 minutes. Using a voltage provided inconsistent current due to both temperature and resistance variations and thus led to inconsistent output power.

### 3.5.2 Solutions to Problems Encountered

Lasers are devices that output optical power linearly based on input current after their threshold point, Ith, as shown in Figure 3.15. Using a voltage would provide inconsistent current due to both threshold voltage and resistance variations due to temperature and thus lead to inconsistent output power. A constant output power coming out of the laser minimizes the drift seen in both the background signal and the peak amplitude as seen during a calibration, injection, or separation test. In order to achieve this state, a constant current source needed to be designed to drive the laser. A constant current driver would compensate for any variations due to temperature

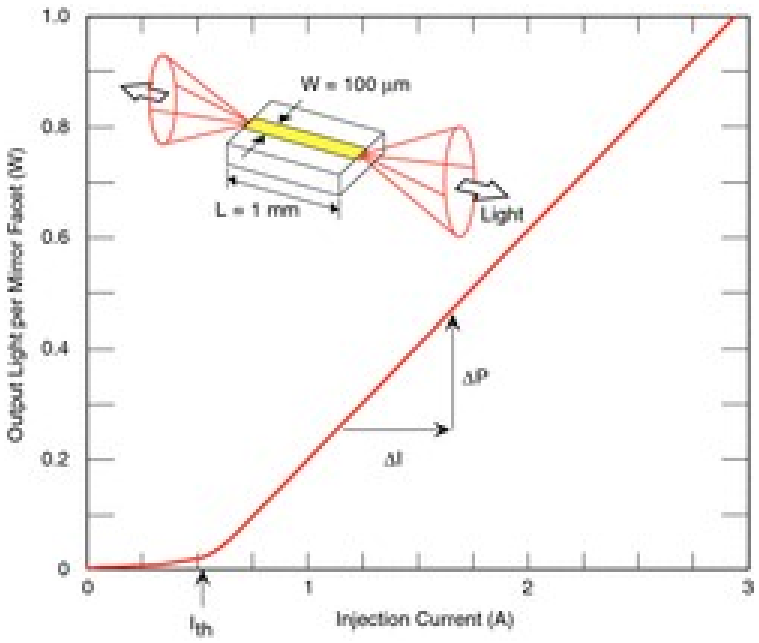



Figure 3.15: Laser optical output power as a function of input current [1]

by allowing for fluctuations in voltage across the laser to not cause a change in the current flowing through it.

### 3.5.3 Final Design Implemented

In the search for a constant current source, the IC selected was the LT3092; using just the ratio of two resistors you can set a constant current output value. The IC and current control resistors are placed in series between the power supply source and the load as shown in Figure 3.16. The circuit board for the laser has three switches that select which resistor you are connected to. Each resistor produces different output intensity. The large set resistor (R1 through R4) has $10\mu$A running through it and the small load resistor (R5) will match the voltage drop across the large set resistor giving a constant current through it. This yields much better stability after allowing both the laser and IC to come to equilibrium. The circuit board layout is shown in Figure 3.17.

Figure 3.18 shows the average optical power out while revealing another issue; the laser has an internal drive circuit that is in conflict with the external circuit that was designed. Although much more stable than the prior circuit design, the mystery circuitry inside the laser proved to be problematic in being able to get readings unless the laser was allowed to warm up for 30 minutes

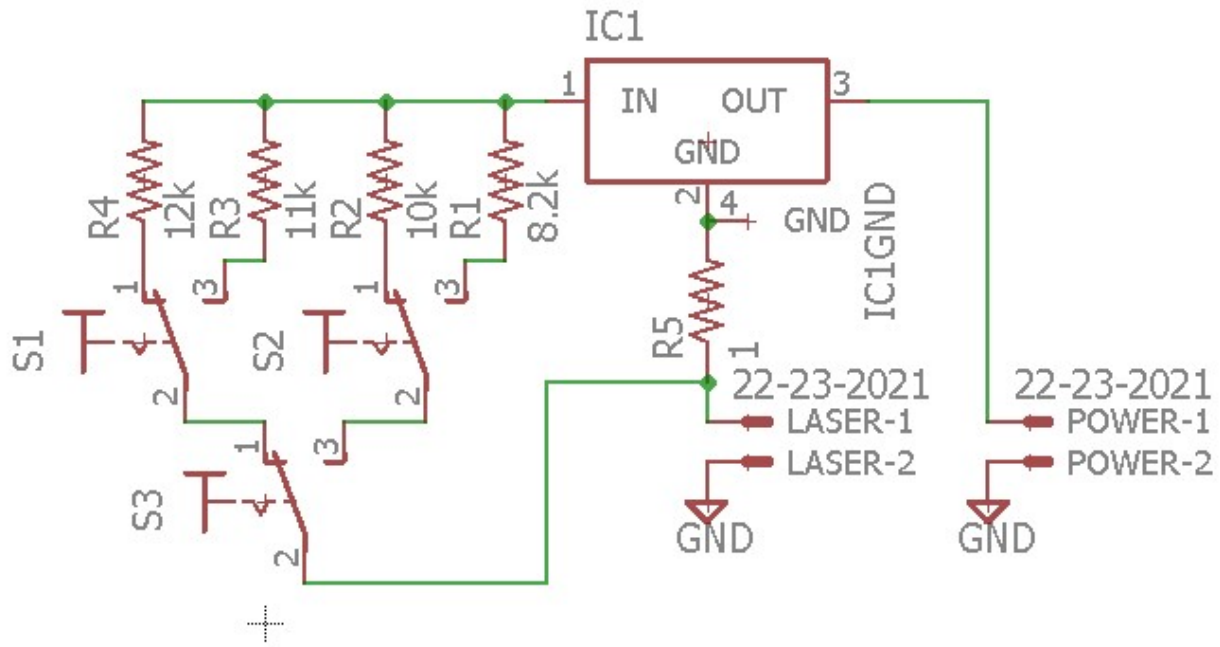



Figure 3.16: Laser driver schematic

prior to running the system. This is the main reason that we were getting significant data differences and is discussed in further detail in Chapter 4. The circuit has to restart the stabilization time for every change made to the input current setting of the laser driver.

### 3.5.4 Recommended System Improvements

It is highly recommended to spend more of the budget in the laser category. Having a high-quality laser will provide a more stable output that reaches equilibrium faster. Adding in TTL control will allow frequency modulation that allows the output signal to move away from base band where flicker noise is a large factor in the noise floor seen in the PMT signal. With a new laser, a redesign of the optical bench 3D printed part would be necessary.

## 3.6 Optical Components

The optical components control the flow of light through the optical bench. The laser light is routed to the device but rejected from going toward the PMT. The emitted light from the excited fluorophore is collected and directed to the PMT. The components that make this happen are a

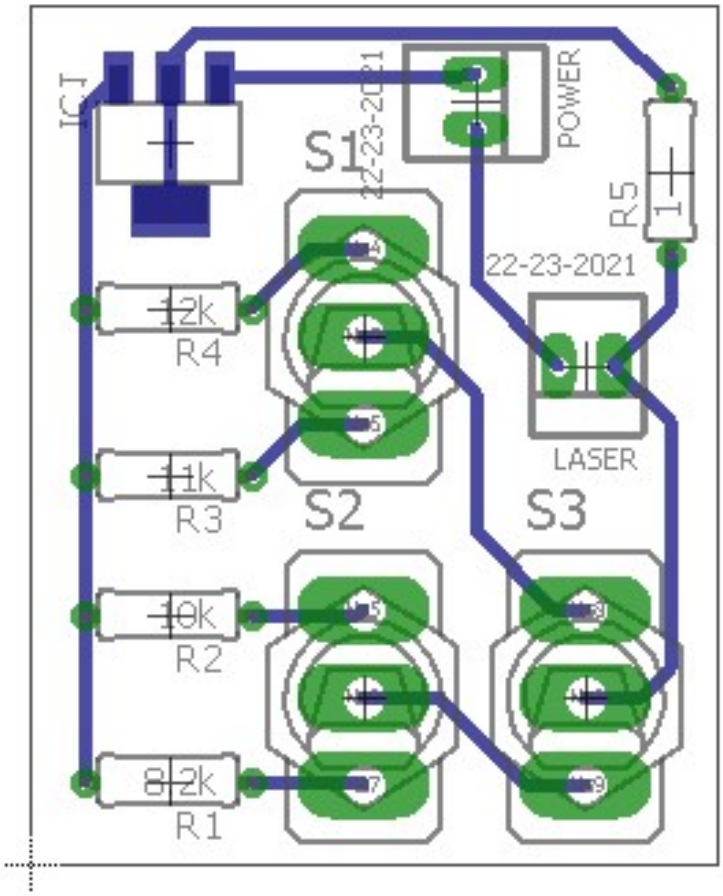



Figure 3.17: Laser driver layout

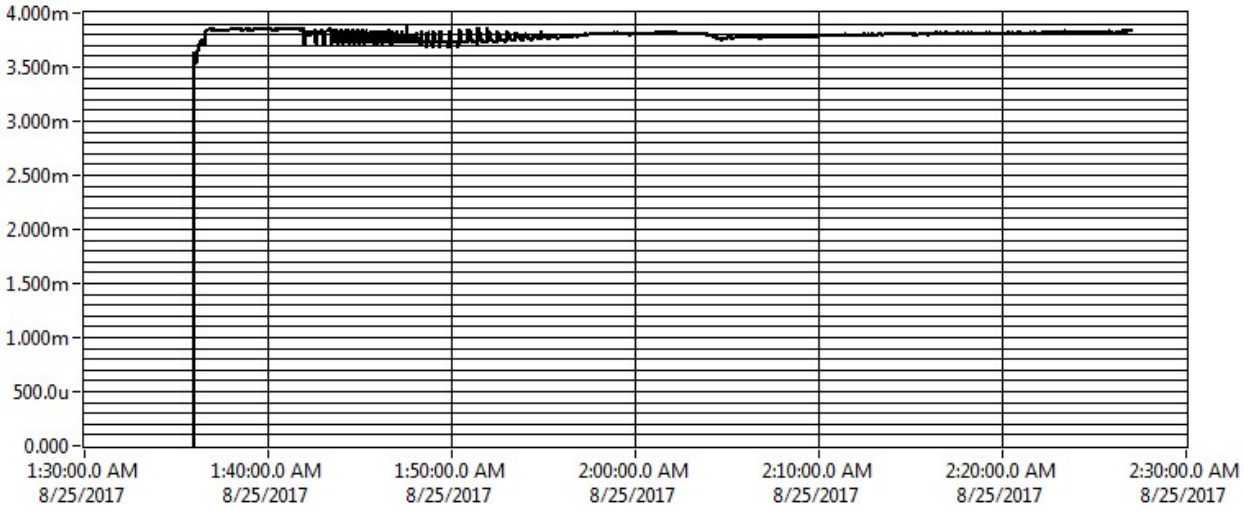



Figure 3.18: Laser output over 51 minutes

neutral density (ND) filter, lenses, a dichroic beamsplitter, a long pass filter, and a pinhole. Each of these components will be described in function and attributes of the specific one used in the implementation.

### 3.6.1 Neutral Density Filter

The laser power needed to excite the sample in the channel is much smaller than the output power of the laser. To reduce the laser's effective output power, an ND filter is employed. An ND filter allows only a certain fraction of light to pass through which is determined by the filter's optical density. Optical density (OD) is on a logarithmic scale and follows equation 3.2.

$$\text{Faction of Light Let Through} = 1/10^{OD} \tag{3.2}$$

With a 5mW laser feeding the system a 1.3 OD ND filter was selected, which results in 250$\mu$W passing through the neutral density filter.

### 3.6.2 Lenses

The laser emits a collimated beam of light. To focus the light down to the size of the channel, a lens is used. The lenses interface the optical bench with the device and the PMT. On the interface between the device and the optical bench, the lens has two purposes. The first is to take the collimated laser beam and focus it down to the channel for excitation of the label. The second is to collect fluorescent light from the device, collimate it and route the signal to the PMT. The focal lengths of the two lenses are 10.92mm for the device and 18.15mm for the PMT.

### 3.6.3 Dichroic Beamsplitter

The dichroic beamsplitter (Semrock part number FF535-SDi01) is used to pass the 532nm laser light and to reflect light with a wavelength longer than 535nm. Figure 3.19 shows that if the wavelength of light is longer than 535nm then the dichroic beamsplitter reflects at least 98% of the light; if shorter than that wavelength it varies between 5 and 10% reflected. The beamsplitter is designed to work at a 45-degree incidence angle for the light.

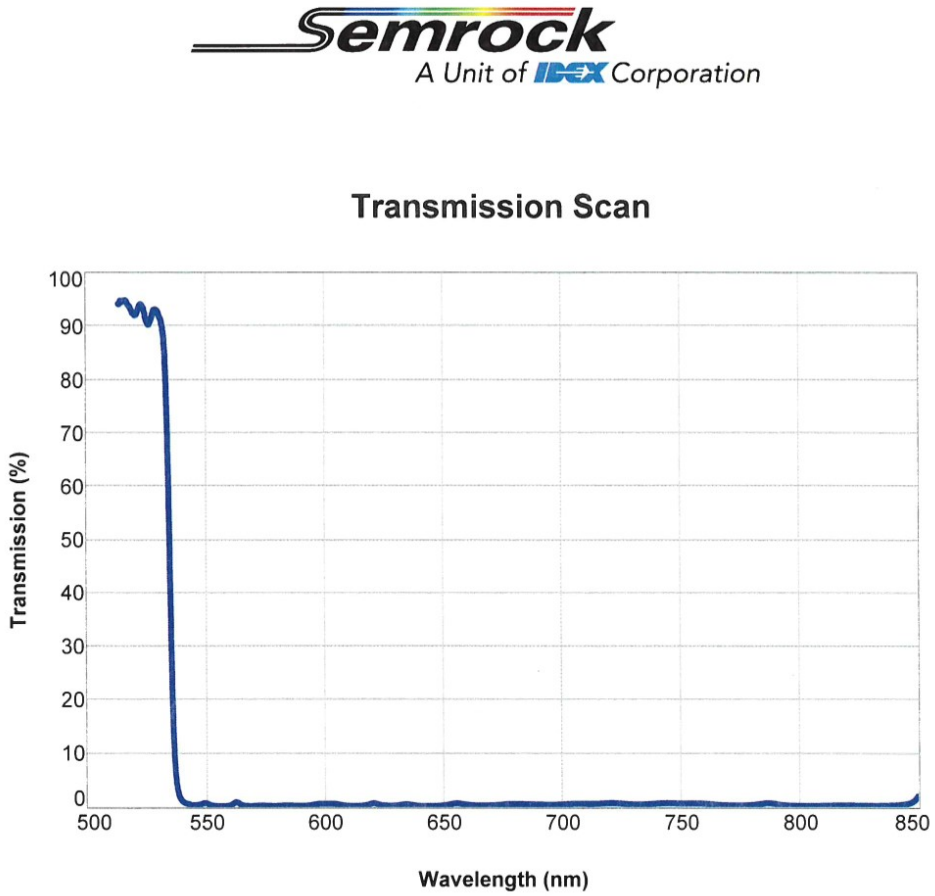



Figure 3.19: Transmission spectrum of the dichroic beamsplitter [2]

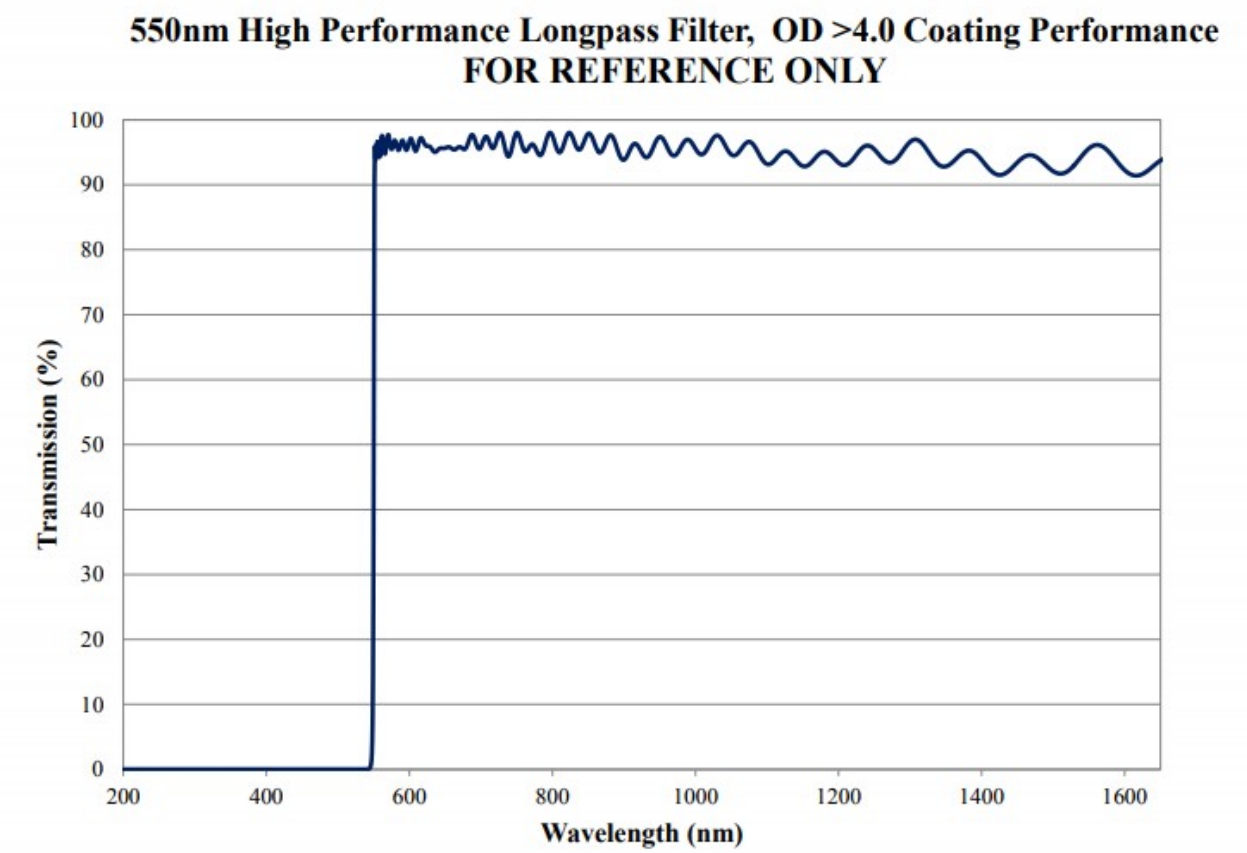



Figure 3.20: Transmission spectrum of the longpass filter [3]

### 3.6.4 Long Pass Filter

The long pass filter allows the longer wavelengths (550nm+) to pass while attenuating wavelengths below 550nm. This is ideal because the laser light that we do not want getting to the detector is 532nm and the emitted light from the fluorophore is above 550nm. The selected filter (Edmund Optics 550nm 12.5mm Dia., High-Performance Longpass Filter) passes less than 1% in its stop band and greater than 90% through the majority of its passband as shown in Figure 3.20. The passband for the filter includes the wavelengths longer than 550nm while stopping over 99% of wavelengths between about 420 and 550nm.

## 3.7 XYZ Stage

Alignment is the ultimate pitfall of the overall system and the two sets of XYZ stages are the way that the system is designed to overcome that obstacle. The stages have to meet stringent

tolerance and step size requirements in order to be able to line up the small channels and pinhole with the focal points of the optical path created by the lenses. With the channel having a cross-sectional area of 50$\mu$m by 50$\mu$m and the pinhole being 100$\mu$m in diameter, that provides the parameters needed to calculate the minimum resolution that we need for the XYZ stages. The limiting factor in the design is the channel size for the minimum step resolution needed to place the focal point of the laser inside the channel. In order to focus the laser beam to be wholly within the channel, the resolution must be at minimum the width of the channel. This resolution would guarantee and least the center of the beam is at least in the channel during one step. In order to maximize the power seen by the channel, a resolution smaller than a channel width is needed for the stages. The resolution needed is based on the beamwidth at the laser's focal point and the variability in the channel width across devices. The following sections will cover the motor controller design for the motorized XYZ stage, the issues with the current stages, and the changes that need to be made in order to have better performance.

### 3.7.1 Problems Encountered With the Initial Design

In order to be able to make measurements, the microfluidic channel and PMT must be aligned to the optical path. This allows the PMT and the sample holder to be positioned at the points that give the highest signal. To do that the XYZ stage is made up of three motorized linear stages, three Sparkfun Big Easy Drivers, and the signals delivered by the Arduino based on the input from the computer keyboard. Each stepper motor is connected to a driver. The driver was selected for being capable of micro-stepping which is a movement of only a fraction of a step, up to 1/32 of a step. The stepper motor works by energizing the motors coils to generate a magnetic field which is used to align a spoked driver gear inducing a rotation of a fixed number of degrees. The motor shaft is connected to the lead screw which moves the linear stage.

### 3.7.2 Solutions to Problems Encountered

The only issue encountered when building out the system was that of a common ground. When trying to implement initially, power was being delivered to the Big Easy Driver for both the logic level and motor power. After digging into the documentation, the 5V regulator on the Big

Easy Driver was discovered to be converting the 12-volt motor power to the 5-volt logic level. Removing the redundant 5-volt power supply connection did not eliminate the source of the horrible motor grinding sound when running. By unplugging the ground connection to the Arduino that was supplying the five volts the sound stopped uncovering the actual issue, differing ground potentials causing signals to be in the digital gray region where circuits have a hard time determining if the signal is a 1 or 0. When connecting the ground through the Big Easy Driver between the Arduino and the Triple Power Supply the board would not accept every command and draw a more current than needed to run the steppers. After disconnecting the Arduino it would function properly again.

The next test was disconnecting the Arduino's power from the computer and powering it from the same Triple Power Supply as the Big Easy Drivers which worked. From these two tests, I found that it was an inconsistent ground between the computer and the Triple Power Supply. By adding in a ground lift on the cord connecting the Triple Power Supply to the wall (removing the 3rd prong from the circuit) the whole subsystem functioned as intended and no issues were located elsewhere in the entire system.

### 3.7.3 Final Design Implemented

In order to get the resolution that was required for the system, the motors and drivers were the two main concerns for the design. The three stepper motors each have 128 steps per revolution and each is attached to a lead screw with a pitch of 1mm. This is equivalent to 7.8125um (pitch divided by number of steps) per step which is a factor of approximately six smaller than the width of the microfluidic channel. Each of the stepper motors draws less than 1A which allowed for the use of a single triple power supply wired in parallel illustrated in Figure 3.21. This internal system configuration allows a max of 4A out to the X5 and X6 connectors on the PCB in Figure 3.22. The connections to these motors come through DB9 connectors on the XYZ Motor Drivers PCB labeled as X1, X2, and X3 in the schematic.

With the motors and power supply OK, that left the drivers. The main specification of concern was the voltage and current requirements for the stepper motors that the drivers needed to interface with. The stepper motor voltage and current required are 12V and 1A maximum. The Big Easy Driver board from Sparkfun was chosen because it can handle up to 1.4A without a heat sink and the 12V fell in the middle of the motor voltage range for the board. A secondary

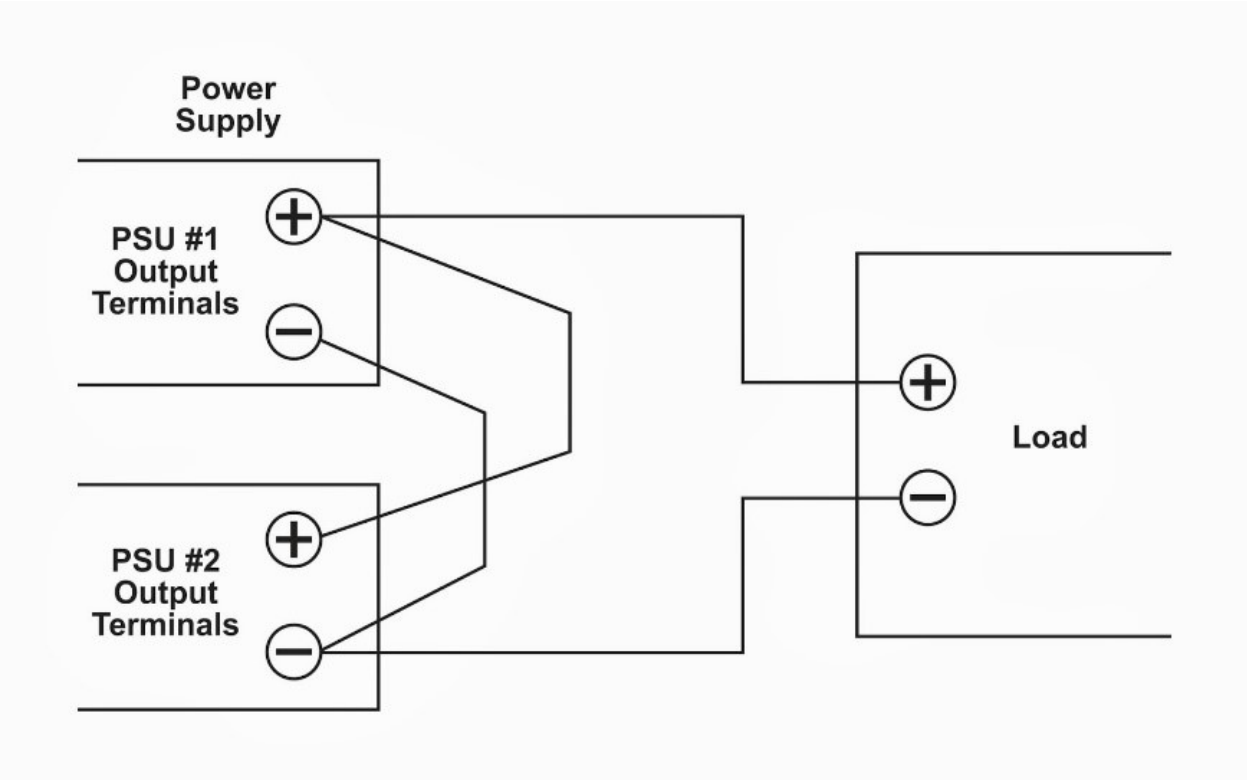



Figure 3.21: Parallel power supply configuration [4]

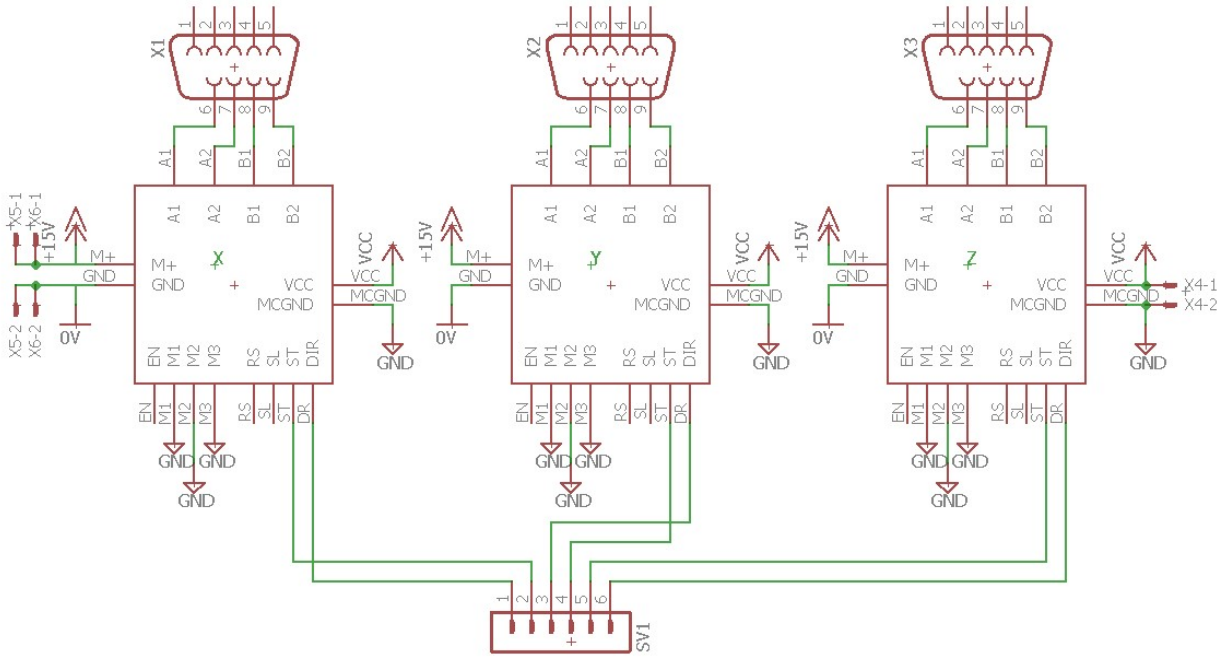



Figure 3.22: XYZ Motor drivers PCB schematic

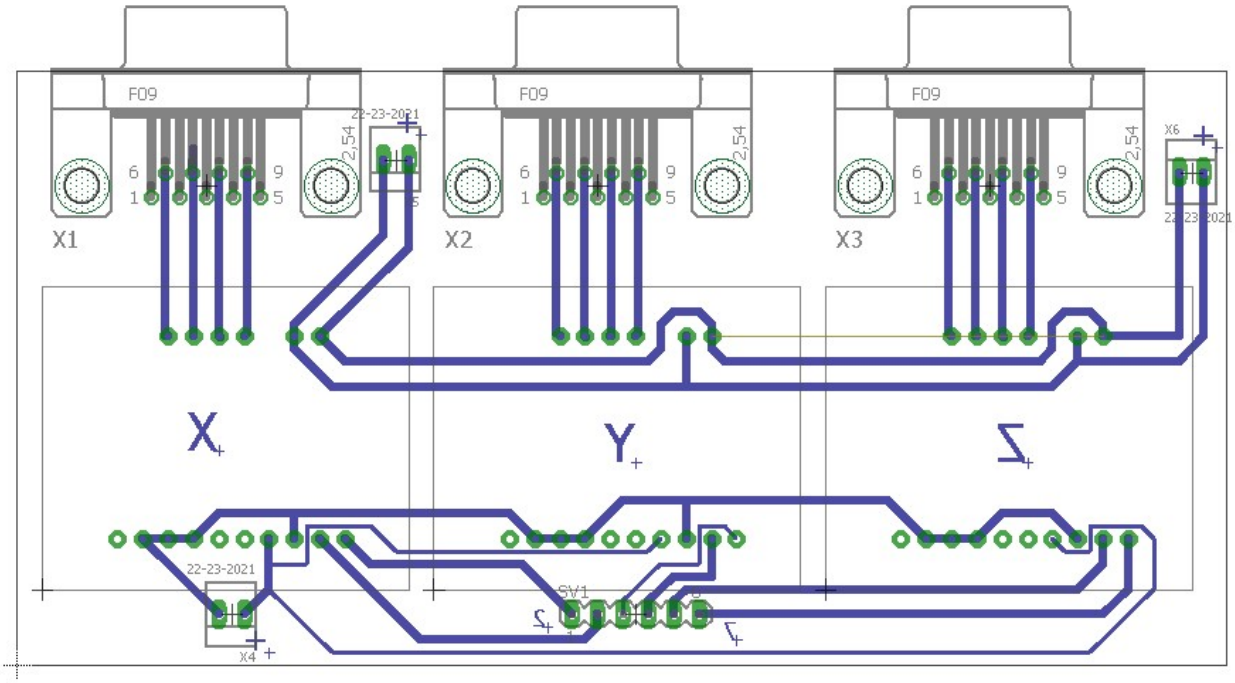



Figure 3.23: Layout of the XYZ motor driver PCB

reason the Big Easy Driver was chosen was for the Arduino code library, AccelStepper, which made implementing the design fast and simple.

The library sends serial commands from the Arduino microcontroller to the Big Easy Driver board through the SV1 connector. The Big Easy Driver interprets the signals to drive the motors. The Python script on the connected PC interfaces with the Arduino UNO that is listening to the sent commands. The layout in Figure 3.23 shows the location of each driver as well as what axis connects to which driver on the board.

### 3.7.4 Recommended System Improvements

The one required change to the subsystem that needs to happen to be functional is to purchase stages that have a higher of tolerance for the limit of uncontrolled motion. The motorized y stage has 1.5mm of movement in the connection between the lead screw and the motor. All of the motorized stages have minor movements due to gaps between the lead screw and the shuttle plate but this range of motion is significantly larger than their manual counterparts. The main issue on the manual stages is that they are spring loaded. Spring-loaded has the attribute that if the stage

catches with a non-zero settling distance and the stage vibrates releasing the catch, the spring will resettle the stage in a different position and move out of alignment. This happened a few times before the problem was identified. If motors were the choice to move forward with it would be ideal to select a finer pitch motorized stage of higher quality for both XYZ stages.

The one adjustment to the system that could be helpful is adding micro-stepping to get finer than 7.6125$\mu$m resolution in any of the three directions. This would allow for greater precision in focusing in the best spot to obtain the greatest signal. The Big Easy Driver has the capabilities to do full, half, quarter, eighth, and sixteenth steps. The truth table for how to set the three micro-stepping pins to select each mode is included in the datasheet on Sparkfun [18].

## 3.8 Photo-Multiplier Tube

The photo-multiplier tube (PMT) acts as the detector for the system. The same PMT was used in both systems which eliminated a variable that could have led to performance differences between the two systems. This section will discuss reasons for choosing the module, problems that were present in the initial design, solutions developed for those problems and suggested changes to increase performance moving forward.

### 3.8.1 PMT Specification

The PMT used in the system is a Hamamatsu H10722-20. It was chosen primarily for its sensitivity response curve (Figure 3.24) at the wavelength of interest 554nm. The -20 curve (red) has its highest sensitivity between 500nm and 680nm which includes the wavelengths we are both excited and fluorescing at. The one downside to this is it also is very sensitive to the 532nm excitation wavelength that is produced by the laser so scattering and reflections need to be minimized in the system.

Another major reason to select this PMT is the integrated amplifier inside the unit as shown in Figure 3.25. The amp shown boosts the signal allowing sensitivity down to much lower concentrations. Also in the figure, you see the inputs and outputs laid out with two voltages, Vref and Vcont. Vref is a 1.2V reference voltage output from which you can derive the voltage necessary to send back into Vcont. Vcont is the control voltage for the gain of the PMT. Figure 3.26 shows

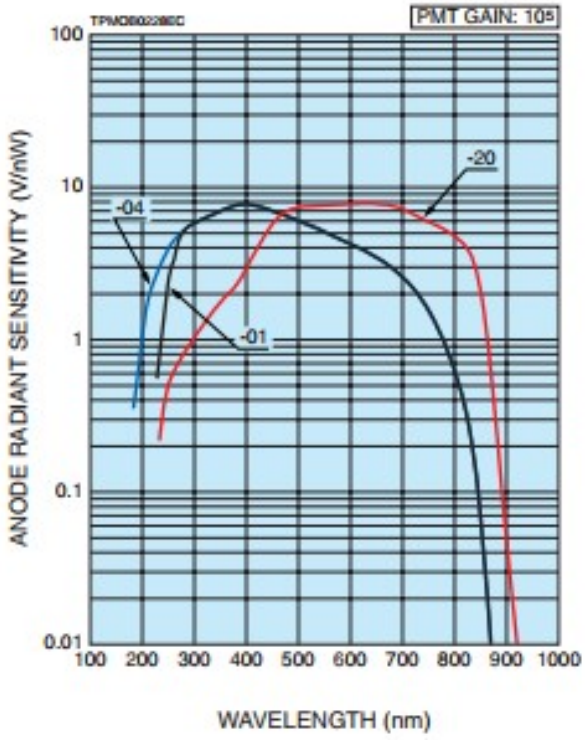



Figure 3.24: Typical spectral response of H10722 [5]

what the corresponding PMT gain is based on the input voltage applied to $V_{cont}$. Now that the reasons for selection are clear, the next section covers the issues preventing the PMT's optimal performance.

### 3.8.2 Problem with Initial Design

The initial design had three problems that inhibited optimum functionality from the PMT. The first problem was the breadboard making insufficient and unsecured connections. Second, the controls for the gain of the PMT were ineffective at creating a stable and repeatable gain. Third, the supply voltage fluctuations greatly affected the sensitivity.

The initial design included everything hooked into a breadboard. From this stemmed all of the connection problems that were encountered with the PMT. Instead of a connector, bare wires were plugged into breadboard slots. Due to the 26 gauge wire size being much smaller than the 22 gauge sizing on the breadboard holes, the wires are not held very well. This caused the power wires to either fall out or lose connection with the rest or the breadboard row. The other issue with

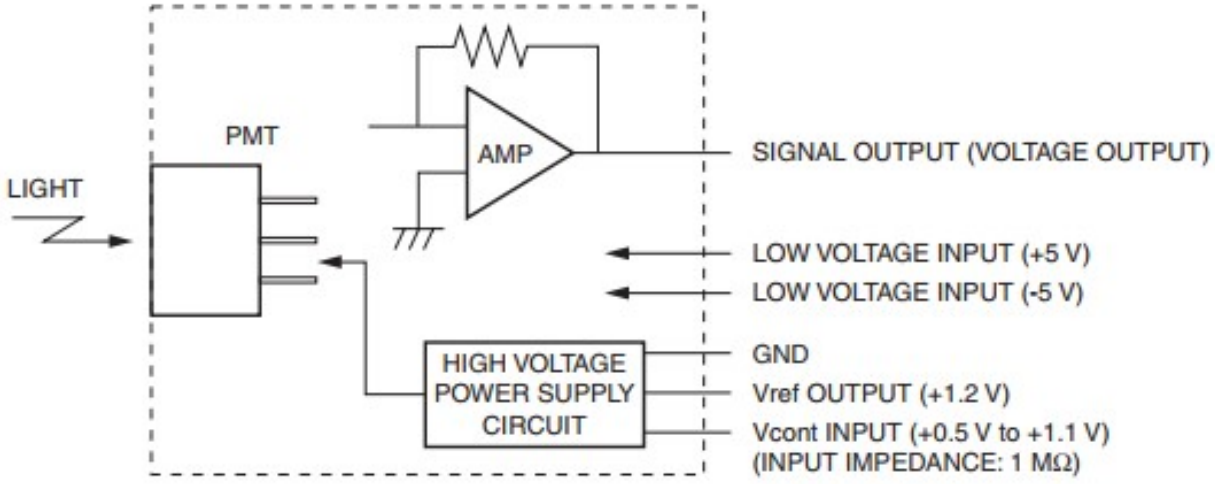



Figure 3.25: Schematic diagram of Hamamatsu H10722-20 [6]

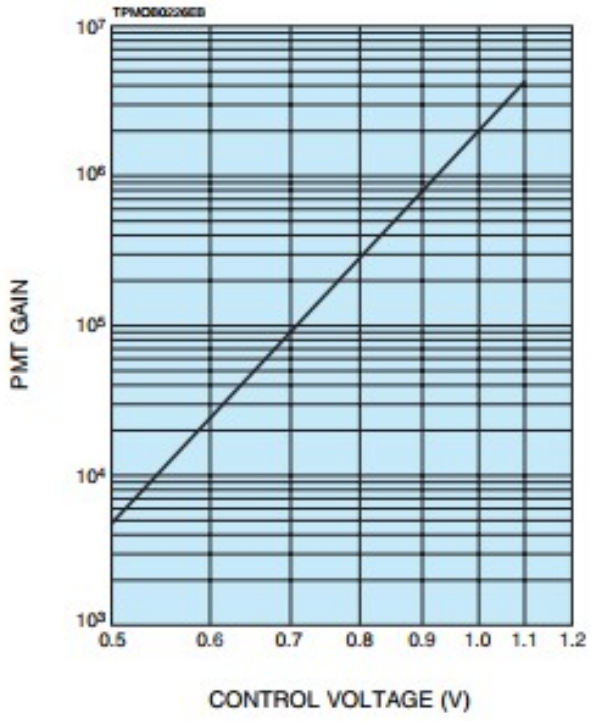



Figure 3.26: PMT gain vs control voltage [7]

the breadboard was the signal output on the RG-174/U coaxial wire was connected to the board using alligator clips on the stripped end. This introduces noise into the signal because the ground no longer shields over half of the distance between the PMT and the op-amp.

After the issues with the breadboard, the next major problem arose from the use of a manual single turn potentiometer to set the optical gain of the PMT. It was designed with sharpie marks to align to for the two gain settings. The problem with this is that the resistance when moving back and forth between different settings is not guaranteed to be the same. Due to the linear input voltage invoking a logarithmic change in the gain, a small difference makes a larger impact on the overall output of the PMT.

Third, power supply voltage fluctuations is the final issue that was having a significant effect on the PMT output. Commonly referred to as voltage ripple, these fluctuations manifest themselves in the form of PMT gain instability. The voltage fluctuations can either push the PMT out of the operating voltage range if large enough or can affect the ground relative to the whole system. These fluctuations affect both the amplifier and the high voltage source within in the PMT housing.

### 3.8.3 Solution to Problems

The first step taken was to remove the breadboard and replace it with a PCB. This removed any disconnection issues that were occurring and cleaned up the overall set-up. The output signal from the PMT was routed into an SMA connector on the PCB and a cable mount was attached to the RG-174/U coming from the PMT to eliminate noise getting into the wiring. After the breadboard was removed the manual potentiometer was replaced with an electronically controlled digital potentiometer for repeatability of the gain setting. The power supply was ultimately still an issue and will be discussed further in Chapter 4.

### 3.8.4 Recommended Changes to PMT Setup

The recommended change to the system is to obtain the corresponding power supply for the PMT module. This provides two main advantages. The first is removing the dependency on Vref to supply Vcont. Removing this dependency provides a stable voltage keeping internal gain

constant throughout a run when it sags as it goes above 75% output signal. The second advantage it provides is electrical isolation for the PMT while removing the need for a precision triple power supply and separate control circuitry using the common ground of the whole system. The main advantage this has is to separate the PMT from the sources of noise such as the computer and linear power supplies on the common ground. This switch to the corresponding high precision power supply isolates the PMT inputs from noise due to the circuitry on the Arduino Shield and on the motor driver board.

## 3.9 Arduino Shield

The Arduino shield performs the following critical functions for the entire system: the signal path is filtered and limited, the 3-axis sample holder stages are controlled, and the PMT receives all necessary voltages from this PCB. Each of these functions plays a crucial role in maximizing the amount of signal that comes out of the PMT. The signal path circuit was initially just an operational-amplifier (op-amp) and passive second order filter on a breadboard. The issues and design discussed in this section will explain why the features were added as well as what could be improved if it does not meet requirements for a particular application.

### 3.9.1 Problems Encountered with the Initial Implementation

The initial design had the coax output of the PMT connected into the input of an op-amp. The output of the op-amp was run through a second order low pass filter and then into the analog to digital converter (ADC) on the Teensy 3.2. Although a simple implementation, the op-amp design worked well as a limiter in order to ensure that the signal would not go above 3.3VDC and damage the Teensy ADC. The first issue arose by placing the filter after the op-amp which increased the noise floor of the signal path. The second issue was the manual potentiometer that was being used to set the gain lacked true repeatability in duplicating the same gain after changing it. The third issue was the type of wire being used to transmit the signal from one point to another.

The first issue created an interesting result. By using the op-amp to clip the signal to 3.3VDC you introduce a lot of frequencies similar in composition to a square wave. When low pass filtering after the clipping you attenuate the higher frequencies that were added in to cause

the clipping and lose signal integrity by distorting the signal. Due to the signal, we are measuring is very close to DC the amount of noise that the filter added to the signal was detrimental to being able to measure small concentrations.

The second issue of being able to repeatedly set the potentiometer was near impossible to accomplish. The procedure either required a lot of fine tuning with a screwdriver and multimeter or just tuning to a line drawn on the enclosure of the potentiometer. The first was time-consuming but more accurate while the second option was quick but had a much larger margin of error.

The third real issue with the initial design was the wire being used to transmit the signal from the breadboarded circuit to the Teensy's ADC. The wire being used was solid core non twisted pair wiring. The main issue with this is that the wires, when touched directly or moved indirectly, would have an effect on the measured signal. The effect could be a small change in the DC level to losing connection to the breadboard leading to the op-amp acting like a comparator by going high or low. All three of these issues were addressed in the final design of the Arduino Shield.

### 3.9.2 Solutions to Problems Encountered

The main change that all the rest of the solutions in this section revolve around is the design and building of an Arduino Shield PCB. The reason for this is to remove the faulty connections, improve the signal integrity, and consolidate the required circuitry into a much smaller footprint. The solutions to issues below are all incorporated into this PCB.

The first issue tackled was the op-amp before the analog filter as it added the most noise into the signal. By filtering the signal after it came out of the PMT and then feeding it into the op-amp for limiting, the signal could then be seen real-time on the computer without the computer doing any digital filtering. Although this was sufficient the analog filter was replaced with an IC filter (LTC1069) that had a tunable corner frequency in order to increase the attenuation in the stop band and achieve lower attenuation in the pass band. Figure 3.27 shows the frequency response when a 300KHz clock frequency is provided to the chip.

The second issue of setting the gain on the op-amp was overcome by replacing the manual potentiometer with a digital potentiometer. Digital potentiometers are analogous to having a number of resistors in series that make up a potentiometer. The circuitry inside connects the wiper to one of the tap points between resistors. This means that the resistance at that point is always going

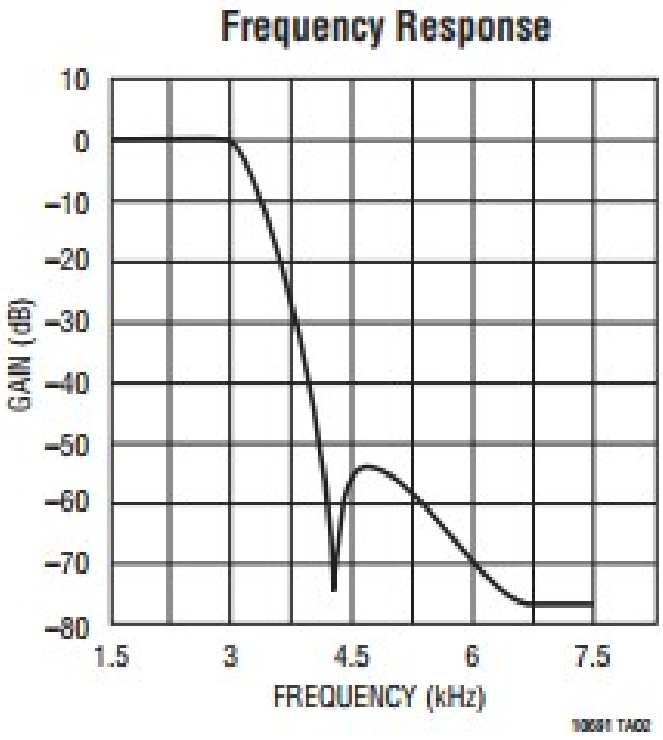



Figure 3.27: Example frequency response of the LTC1069 with a 300 KHz clock [8]

to be the same. Although there are fewer possible values for the potentiometer, the repeatability of the circuit was the goal of the design.

The third issue and cause of the noise was not using a shielded wire to run the distance between the breadboard and the Teensy ADC pin. The signal coming out of the PMT comes out on an RG-174/U coax cable that was plugged into the breadboard. In order to maintain that signal over the wired connections between the PMT, PCB, and the Teensy, the wires were replaced with shielded coax and SMA PCB mount connectors.

### 3.9.3 Final Design Implemented

The shield designed provides the interface for all the necessary controls of the photomultiplier tube (PMT). The PMT needs seven connections, three for the 5V dual power supply, a Vref output, a feedback voltage for gain control, and a two connection coax for the signal output and ground. The five different power and control voltages for the PMT route through the PMT-CONN in Figure 3.28. The SV3 connector is where an external 5V dual power supply is connected in.

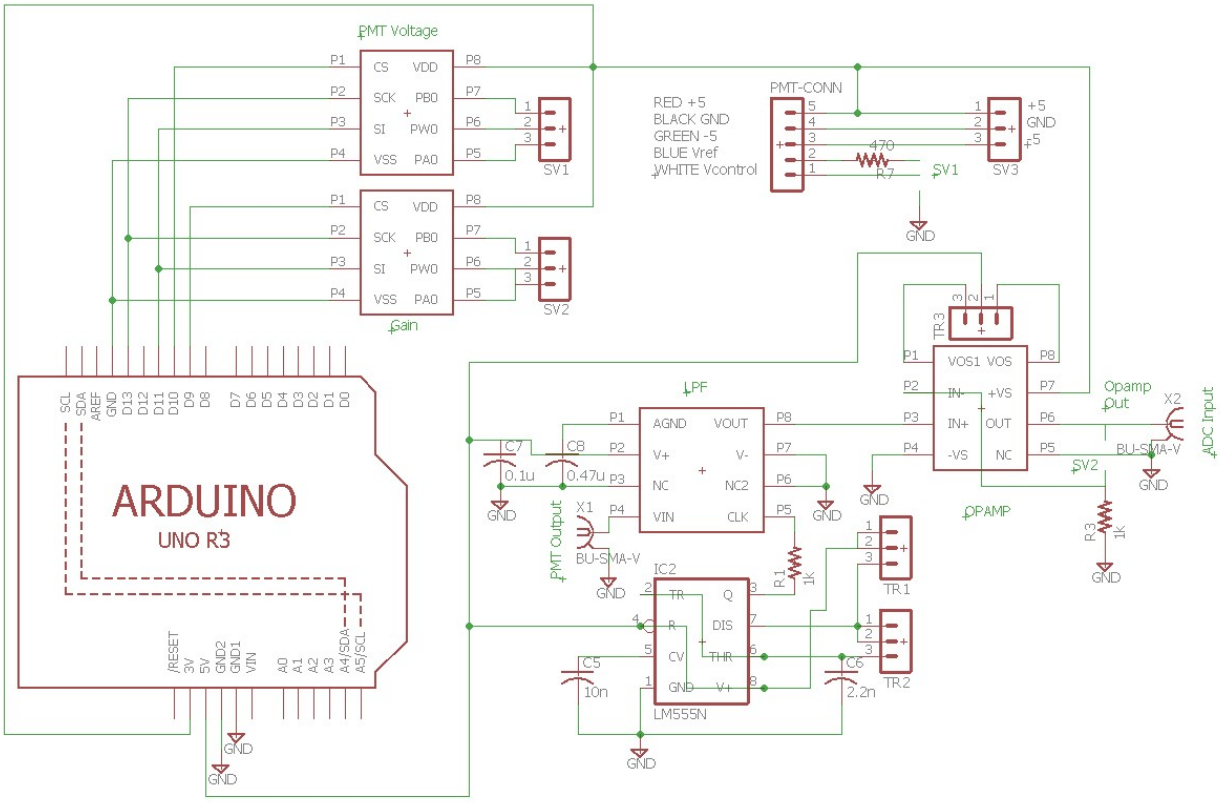



Figure 3.28: Schematic of the Arduino shield

The feedback voltage is generated by taking the $V_{ref}$ output of the PMT and be divided by a voltage divider, the potentiometer connected to SV3, to provide the voltage to the pin that sets the gain of the PMT. This potentiometer is sent I2C commands based on the input from the python interface. The value can vary between 0 and 1V but we stay in the range from 0.5V to 1V, which is typical for PMT usage.

The signal from the PMT (X1) first goes into a low pass filter (LPF) IC. This IC (LTC1069) is an 8th order progressive elliptic LPF. The corner frequency is set by the output of the 555 timer, IC2. This output from IC2 is fed into the $f_{Clock}$ pin on the LPF chip in Figure 3.28 to set the corner frequency for the internal filter. The output frequency of IC2 is set to one hundred times the desired corner frequency as noted in the example in Figure 3.27. Both ICs are supplied with 5V from the Arduinos 5V pin.

Next, we send the signal through the limiting amplifier. The gain of the LT1677 op-amp is controlled by R3 and the other digital potentiometer. This varies between a gain of +1V/V and

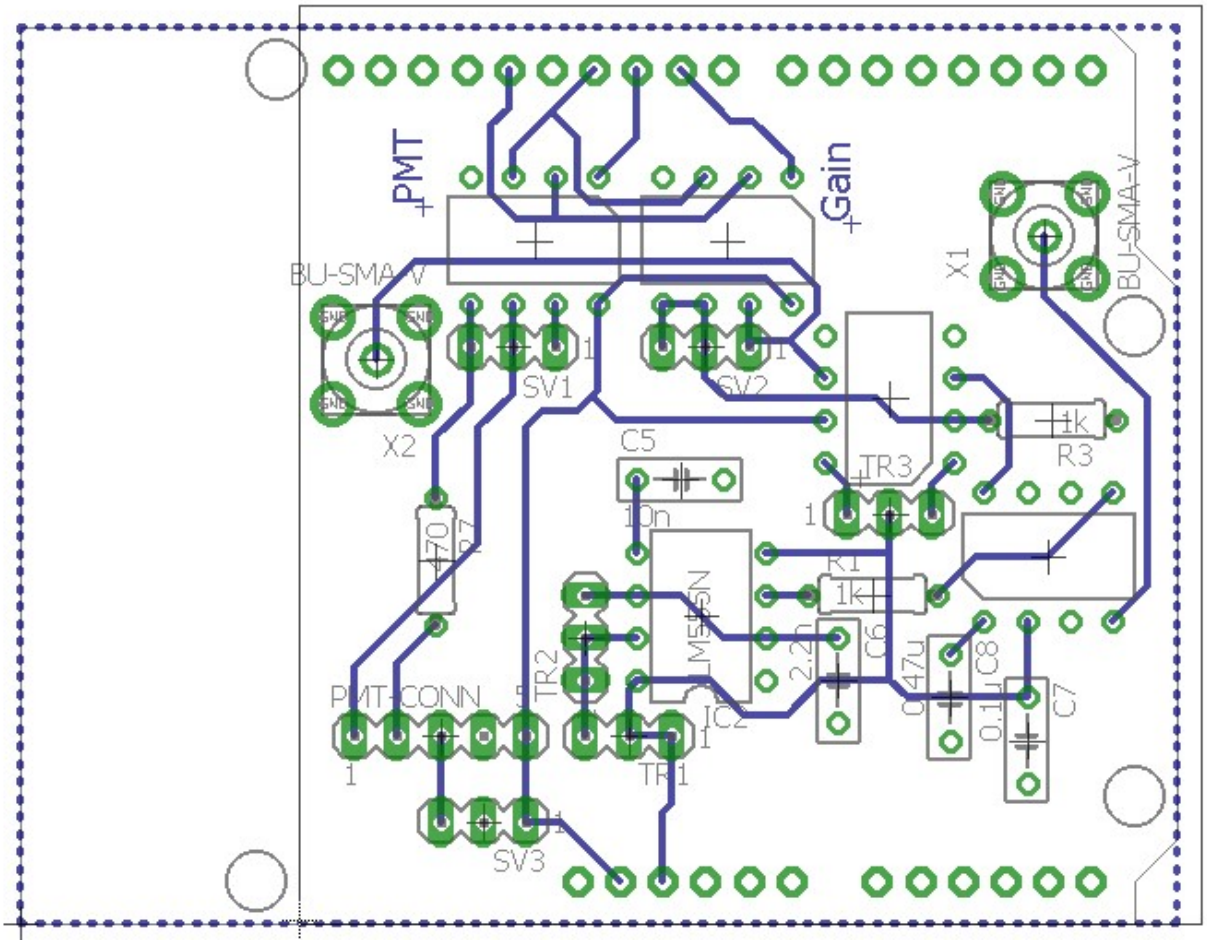



Figure 3.29: Arduino shield PCB layout

+11V/V. This op-amp also limits the output voltage from a 0-4V input possible to the 3.3V max of the Teensy ADC input. The signal is then sent via coax and SMA connectors (X2) to the Teensy on the high voltage board for sampling. The final layout of the entire shield is shown in Figure 3.29.

### 3.9.4 Recommended Arduino Shield Improvements

The first recommended change would be to change the limiting circuit to a 3.3V Zener diode. This will remove the DC bias added by the op-amp not operating rail to rail and provide a simpler, cleaner circuit. I would also look into only using the limiter circuit as an alternative. By design, ADCs have high input impedance and adding in a 3.3V limiter would greatly simplify the circuit and add the protection the buffer provided without the high DC offset. The issue with this

is losing your top approximately eighteen percent of the signal coming out of the PMT. Another option would be to incorporate an ADC onto the shield in order to be able to remove the limiter altogether and increase voltage range past the Teensy 3.2's capabilities.

## 3.10 Microcontrollers

The microcontrollers are the interface between the computer and the hardware of the system. Each microcontroller was chosen for the specific tasks it was assigned to control. A description of the tasks and the abilities of the microcontrollers are laid out in this section.

### 3.10.1 Arduino

The Arduino is the workhorse in the system. The Arduino interfaces with the computer through the Keylogger.py script in order to receive commands to set the two digital pots and control the motors. The potentiometer commands are whether to increment or decrement the count or position of the wiper. The motor commands include what direction and how many steps to move the stepper motors for the XYZ motorized stage. Due to the limited speed of the Arduinos processor, it was only designed for tasks that did not have timing concerns, which are the parts that the user interfaces with. This allowed it to control many features without concern for signal integrity being compromised. The Arduino has the ability to control two sets of XYZ stages which are recommended in order to increase the reliability of the system.

### 3.10.2 Teensy

The Teensy is basically a much faster version of the Arduino. It was chosen over an Arduino for data collection because of its capabilities in being able to sample at a much higher rate with the integrated ADC. The Teensy interfaces with the computer through the teensyData.py script to receive commands to control the HV system and when to sample the data. The data was able to be sampled at most around 10 kHz which will allow for modulation in the future in order to reduce the noise floor further.

### 3.11 Python Keylogger

The Python Keylogger.py script (Appendix B) is the interface between the Arduino microcontroller and the PC. Keystrokes are sent across the COM port of the computer that the Arduino is connected to and then decoded in order to execute the corresponding command. The full code is in Appendix D.

#### 3.11.1 Keylogger User Interface

The Keylogger script acts as both a front-end and back-end user interface (UI). This means that when you interface with the UI the same script is what is sending the commands across the COM port to the Arduino. The UI has three menus and two sections as shown in the screen capture of Figure 3.30. The three menus are Ports, Positions, and Set Position. Under the Ports menu, you can select what COM port the Arduino is connected to. Under the Positions menu, you have the option to move the XYZ stage to a position that has been saved for injection, separation, or crosshairs. Under the Set Position menu you can save the current positions under injection, separation, crosshairs, or home.

The upper section is where feedback for the last action is written. This includes movements in the XYZ stage, changes in step size of those movements, and changes to the digital potentiometers. The lower section is where the input is entered. Each keystroke is shown in this section to provide a record of the previous keystrokes entered as well as what COM port the script is connected to.

#### 3.11.2 Recommended Improvements to Keylogger.py

The recommended improvements to this interface are two fold. The first would be to integrate this with the teensyData.py interface. This would allow for easier use for both of the interfaces. The second is to add feedback of what the current states of the variables are and update those values in the upper section to give real-time feedback on what the variables are at.


Figure 3.30: Interface for Keylogger.py

## 3.12 Python Sampling and Graphing

The teensyData.py script (Appendix B) is used to do real-time plotting of the PMT filtered output and control the High Voltage System. The features of this python script show up in three different windows that will be discussed in the following sections. The first section discusses the main window that is the interface for running the system and the second window that plots the data after it has been saved to a CSV file. The second section covers how the data is saved and what qualifiers can be recorded in order to make the data searchable by the third python script.

### 3.12.1 Functionality of the Three Windows

When the program is run, two windows are produced. The first contains an empty graph and a few control buttons (Figure 3.31), and the second contains an empty graph and a single button labeled quit (Figure 3.32). The second window is hidden behind the first window and is not immediately important. The first window is where the data collection happens. On the right side, there are buttons labeled Insertion (meant to be Injection), Separation, HV On, HV Off, and Calibration. When these buttons are pressed, they send commands to the Teensy, which then sends the appropriate commands to the High Voltage System. On the left side, there is a button labeled Start. When this button is pressed, the program will begin receiving and processing the serial data from the Teensy. The Start button then becomes a Stop button, which instructs the Teensy to stop the data collection. The Quit button closes the first window.

After the first window closes, the second window (Figure 3.32) will display the recorded data on the graph if the checkbox Save data was checked. If the checkbox was not selected then nothing will be plotted. The Quit button will close the second window. When the second window is closed and the Save data checkbox on the first window was selected, a third window appears.

The third window (Figure 3.33) prompts the user to record some notes about the test data. There are text boxes for description, device number, a number of times device has been used, laser position, analog gain, laser voltage, PMT control voltage, high voltage settings, the buffer used, fluorophore used, and concentration of fluorophore. There are also check boxes for if there was equipment failure, unknown failure, or the experiment was successful. All of this data is stored


Figure 3.31: Main interface and the first window for teensyData.py

in a log file. When the Submit button is pressed the window is closed and the program is exited entirely.

### 3.12.2 Features of the Script

The teensyData.py script has a few modifiers with which the program can run. Depending on which modifiers are used to launch the script a number of different functions can happen. All of these can be read about if run using the following command, python teensyData.py h. Additional information about the data is saved in a .log file along with all of the input from the final windows form.

### 3.12.3 tagSearcher.py Script

The tagSearcher.py script (Appendix E) is used to search through the notes users have made about the tests run. It is able to select a specific set of runs based on the data in order to find tests

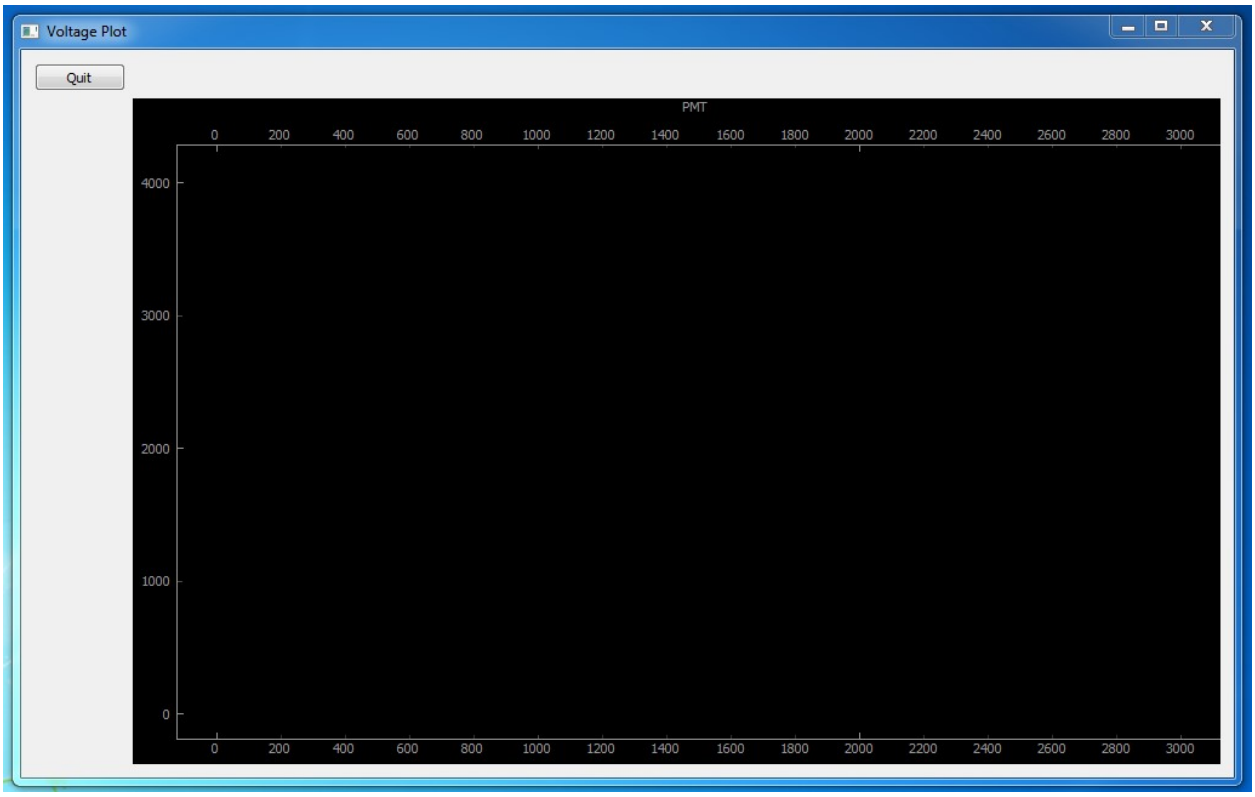



Figure 3.32: Second window for teensyData.py

containing the desired data. When this program is run, it will create a window with many fields that match the ones in the notes from the third window that the teensyData.py script creates. When the submit button is pressed, a new window appears showing the filenames of all files that have tags matching those that the user entered. If the verbose checkbox was checked, this window will show all the notes about those files along with the filenames.

### 3.12.4 Recommended Changes

There are a few improvements that could be made. Incorporating both the teensyData.py and Keylogger.py into the same script but threading the functions separately would allow for an increase in operating speed while still keeping the same functionality. Having one window where everything is taking place is also easier to manage rather than the window alternating that is currently done in order to be able to manage everything on the computer monitor.

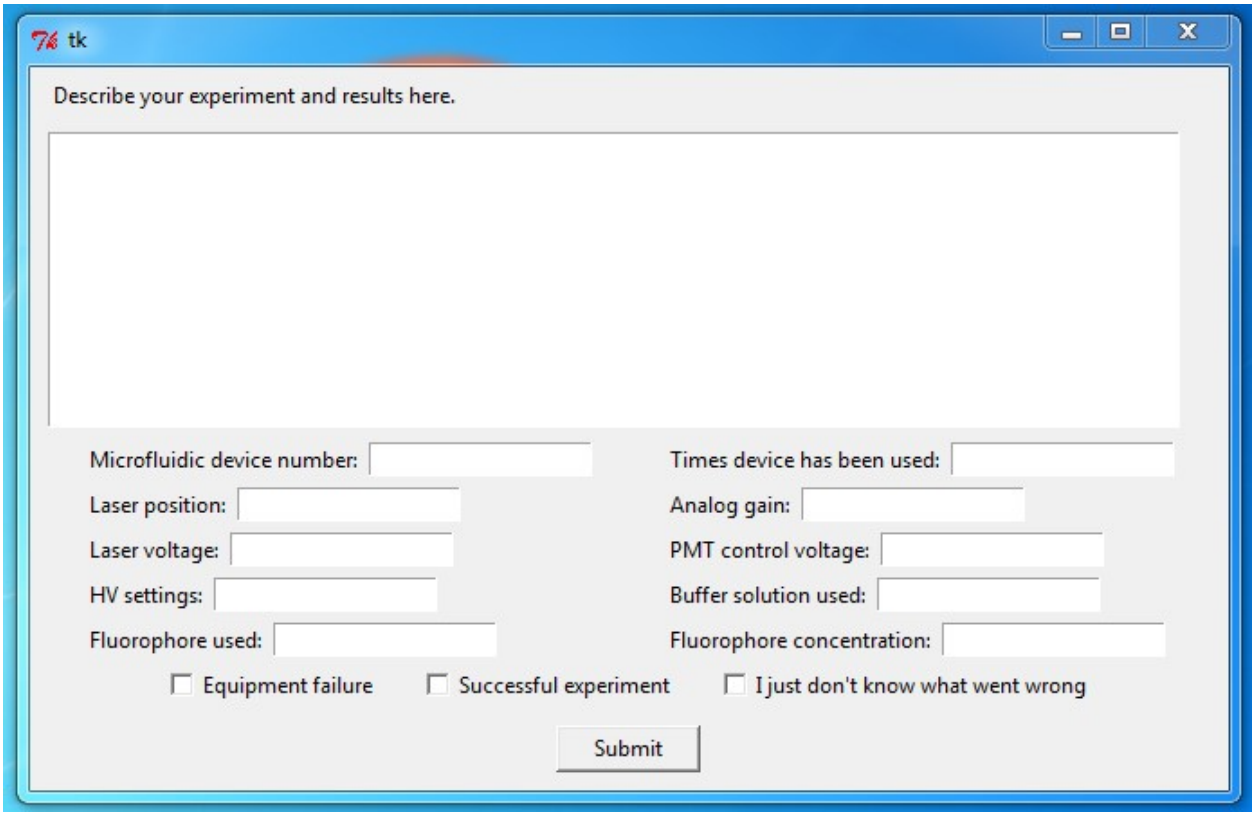



Figure 3.33: Third window for teensyData.py

## 3.13 System Cost

The main hypothesis that this thesis is addressing is "Can we build a CE-LIF system at a much lower cost and with comparable performance to the standard system?" The following section shows the itemized breakdown of the system by subsystem, parts, cost and procurement location. Each table will correspond to one of the five sections in Table 2.1. The first section will talk about each of the five sections which are a laser, optical path, high voltage, signal path, and software. The second section will cover the pricing compared to the standard system as well as the percentage reduction in each system and as a whole.

### 3.13.1 Subsystem Pricing

The laser system had a significant cost reduction due to trying to use a laser pointer style diode. The final design included the laser, current source IC, peripheral components, power supply, and PCB. The individual pricing for the components as well as the source is shown in Table 3.1.

Table 3.1: Laser subsystem pricing

| | Laser Driver | | |
|---|---|---|---|
| 1 | Laser | $42.00 | Aixis |
| 1 | PCB | $33.00 | 4PCB.com |
| 1 | LT3092 - Current Source IC | $2.50 | Linear |
| 5 | 1/4 watt resistors | $0.50 | Mouser |
| 3 | PCB SPDT | $9.81 | Mouser |
| 2 | M Header 0.1” | $0.20 | Shop |
| 2 | F Connector 0.1” | $0.20 | Shop |
| 1 | Dr Meter Single-Output Power Supply | $64.99 | Amazon |
| | Total | $153.20 | |

The largest changes in the system occur in the optical path. The low cost version makes a few significant changes that are priced out in Table 3.2 under three different parts. They are the motor driver PCB, the 3D printed components and the section titled Optical Bench. That section includes the XYZ stages, optics, PMT, power supply and optical breadboard to mount everything to. The optical bench setup replaces the microscope, optical table, and PMT controls with a custom system. The microscope was replaced with linear stages and 3D printed parts.

For the high voltage system, the components listed in Table 3.3 replace the two power supplies and the switching box used in the standard system.

The PMT controller was replaced with the Dr Meter power supply and one of the digital pots on the signal path PCB in Table 3.4. The optical table was replaced with an optical breadboard. The PMT is the same model in both systems. The motor driver board is used to drive the Jiangxi XYZ motorized linear stages.

The Arduino shield priced in Table 3.4 replaces the LNA and the NI data acquisition unit in the standard system. The one note here is that the Teensy 3.2 has an ADC that is used in place of the NI USB-6212 in the standard system to acquire the data. The Teensy is included in Table 3.3 as it controls the high voltage system as well.

The last piece of the system is the software that was written for the low-cost system. This was not given a value along with all of the other hours that were put into the system.

Table 3.2: Optical path pricing

| | Optical Bench | | |
|---|---|---|---|
| 1 | 10.92mm FL - Lens | $20.00 | ThorLabs |
| 1 | 535nm - Dichroic Beamsplitter | $249.00 | Semrock |
| 1 | 550nm - Longpass Filter | $150.00 | Edmund Optics |
| 1 | 18.15mm FL - Lens | $20.00 | ThorLabs |
| 1 | 1.3 OD - ND Filter | $25.00 | ThorLabs |
| 1 | 100um Pinhole | $66.00 | ThorLabs |
| 3 | Micrometer linear stages | $2,310.00 | ThorLabs |
| 1 | XYZ Motorized linear stages | $900.00 | Jiangxi |
| 1 | Optical Breadboard | $265.00 | ThorLabs |
| 1 | PMT - Hamatsu H10721-01 | $1,114.00 | Hamamatsu |
| 1 | Dr Meter Triple Power Supply | $195.99 | Amazon |
| | Total | $5,314.99 | |
| | 3D Printing | | |
| 1 | Sample Holder | $23.70 | Shapeways |
| 1 | Optical Bench (estimate as not compatible file type) | $75.00 | Shapeways |
| 1 | PMT Holder | $39.87 | Shapeways |
| | Total | $138.57 | |
| | Motor Driver | | |
| 1 | PCB | $33.00 | 4PCB.com |
| 3 | Big Easy Driver | $59.85 | Sparkfun |
| 3 | Female DB9 PCB Connectors | $3.72 | Shop |
| 10 | F Header 0.1” | $1.00 | Shop |
| 3 | M-M DB9 Cables | $10.59 | Shop |
| 1 | Dr Meter Single-Output Power Supply | $64.99 | Amazon |
| | Total | $173.15 | |

### 3.13.2 Total System Pricing and Comparison

With all of the components of the individual subsystems laid out the overall cost can be compared. Table 3.5 is broken down into each of the five segments of the system for the standard system, the low-cost system, and the percentage of the standard system. The total row is the overall cost of each system and then the percentage of the standard system cost.

Although the low-cost system reduced the price to 17.41% of the standard system, Chapter 4 goes into depth as to why the overall system was insufficient.

Table 3.3: High voltage pricing

| | High Voltage | | |
|---|---|---|---|
| 1 | PCB (7.25”x4.8”) | $33.00 | 4PCB.com |
| 2 | EMCO F40 | $300.00 | XP Power |
| 6 | 2N2222 | $12.18 | Mouser |
| 6 | 1k 1/4 watt | $0.60 | Mouser |
| 6 | 1M 1/4 watt (2 for VD 4 for load) | $0.60 | Mouser |
| 2 | 100k 1/4 watt | $0.20 | Mouser |
| 4 | 1.5M 1/2 watt | $0.40 | Mouser |
| 2 | 100uF 25VDC caps | $0.18 | Mouser |
| 6 | Coto Relays | $239.28 | Mouser |
| 1 | SMA-F PCM Mount | $2.69 | Mouser |
| 1 | PCB SPDT | $3.27 | Mouser |
| 1 | Teensy 3.2 | $24.94 | Mouser |
| 33 | Female Header Pins 0.1” spacing | $3.30 | Shop |
| 5 | Digital Multimeters (PMT + Voltage & Current for each EMCO) | $20.00 | Shop |
| 1 | Dr Meter Triple Power Supply | $195.99 | Amazon |
| | Total | $836.63 | |

Table 3.4: Filter and data acquisition pricing

| | Arduino Shield | | |
|---|---|---|---|
| 1 | Arduino UNO | $5.00 | Shop |
| 1 | PCB | $33.00 | 4PCB.com |
| 1 | SMA-M Terminator Crimp | $4.33 | Mouser |
| 2 | SMA-F PCB Mount | $5.38 | Mouser |
| 3 | 1/4 W resistors | $0.30 | Mouser |
| 4 | Ceramic Caps | $0.36 | Mouser |
| 2 | MCP41010 - Digital Pot IC | $2.84 | Mouser |
| 1 | 555 timer | $0.46 | Mouser |
| 1 | LTC1069-1 - Low-pass Filter IC | $5.20 | Linear |
| 1 | LTC1100 - Op-amp IC | $7.50 | Linear |
| 5 | 8 pin dip sockets | $4.65 | Mouser |
| 30 | Headers for shield | $3.00 | Shop |
| 23 | headers for pots, PMT, power, and motor controller | $2.30 | Shop |
| 1 | M-M SMA cable 3ft | $16.93 | Mouser |
| | Total | $91.25 | |

Table 3.5: System pricing comparison

| System | Standard | Low-Cost | Percent |
|---|---|---|---|
| Laser | $1,504.00 | $153.20 | 10.19% |
| Optical Path | $27,086.00 | $5,626.71 | 20.77% |
| High Voltage | $3,090.00 | $836.63 | 27.08% |
| Filter and Data Acquisition | $3,852.00 | $91.25 | 2.37% |
| Software | $2,999.00 | $0.00 | 0.00% |
| Total | $38,531.00 | $6,707.79 | 17.41% |

# CHAPTER 4. CHARACTERIZATION OF THE SYSTEM

In this chapter we first review the procedure used to collect data with the system described in Chapter 3 and then discuss the analysis used after the data was collected, the results obtained, why the system as a whole is insufficient, and what changes need to be made to get it working.

## 4.1 System Operating Procedure

The following procedure is what was used to test the system.

HOW TO RUN THE SYSTEM:

1. Locate a microfluidic device.
2. Verify that the microfluidic device (MFD) is functional
   (a) Use the microscope to make sure that the MFD channels are clear
   (b) Fill the source (or insertion) well with deionized (DI) water
   (c) Watch the channels under the microscope to ensure flow in all channels
   (d) If the liquid is not flowing down one or more channels, use the vacuum to assist the flow
   (e) Recheck the MFD under the microscope to make sure there are no bubbles or leaks in the channels
      i. If there are bubbles, use the vacuum to suck them out
   (f) Once you are confident the MFD is functioning correctly, use the vacuum to suck out all of the DI water and use the microscope to verify
3. Insert buffer solution into the MFD
   (a) Start with the source well

(b) Use the microscope to verify proper flow.

(c) If needed, use the vacuum to assist flow or remove bubbles

(d) Once the channels are all full of buffer solution, fill all wells with an equal amount of buffer

(e) Use the microscope to verify that there are no bubbles

4. Insert fluorophore into the insertion well

   (a) First, remove most the buffer solution from the insertion well by holding the tip opposite the channel in the well reducing the chance of introducing bubbles

   (b) Then fill the well with an amount of fluorophore solution equal to the buffer volume into the insertion well

   (c) Use the microscope to verify no bubbles were introduced when filling the well

5. Secure MFD on the sample holder

   (a) Place MFD on the sample holder

   (b) Use tape to make sure the MFD will not slide around during operation

6. Turn on optical bench systems

   (a) Turn on power to XYZ stage

   (b) Start XYZ control python script.

   (c) Turn on laser

      i. The laser should cast a scatter pattern onto the white sheet of paper that is taped to the top of the XYZ stage

      ii. Let the laser warm up for five minutes before running any samples through the systems

   (d) Turn on the PMT power supply (BUT DO NOT UNCOVER UNLESS THE LIGHTS ARE OFF)

7. Turn off room lights

(a) Monitor lights are okay, as are phone screens and other small light sources as long as they aren't directed at the PMT

(b) Uncover the PMT

8. Position the laser so it goes through the source well

9. Align the PMT (iterative)

   (a) Use the X positioning screws on the PMT stage to pull it away from the pinhole.

   (b) Adjust the Z and Y position to maximize the PMT output signal level

   (c) Move the X position closer to the pinhole

   (d) If upper voltage limit (3.3V or 1024 on the ADC) is reached lower the PMT control voltage

   (e) Repeat steps b through d until moving the PMT closer to the pinhole doesn't increase signal level. At this point, you should have found the focal point

10. Move laser to source channel just left of where the channels cross

11. Put high voltage leads into the wells.

    (a) The cables are labeled GND, 1, 2, and 3. Put the labeled leads into the corresponding wells as described in Chapter 2

12. Turn on high voltage system power supply

13. Turn on data collection program

    (a) Click start to begin data collection

    (b) Click HV on

14. Run Insertion

    (a) Click the insertion button on the data collection program. This will automatically set the high voltage system to generate flow from the source well to the waste well.

(b) You should see a rise in the oscilloscope signal level as the fluorescence flows into the channel

15. Realign laser to the separation channel (long channel) just beyond the crosshairs

16. Run separation

    (a) Click the separation button on the data collection program

    (b) You should see a brief spike of the signal as the fluorescence in the crosshair passes down the channel. This close to the crosshair it shouldnt take more than a few seconds to see the spike.

17. If you are able to see the spike close to the crosshair, repeat step 14 and then align the laser farther along the separation channel

    (a) If you can see a spike farther along the channel then the experiment worked as intended.

18. Once you have finished your test, whether successful or unsuccessful, shut down the system.

    (a) Properly close the data collection program using the stop and quit buttons on any windows

    (b) Close the XYZ program

    (c) Turn off the high voltage, laser, XYZ power, and PMT

    (d) Cover the PMT (at this stage you may turn the lights on)

    (e) Remove the HV leads from the MFD

19. Remove the MFD from the system and clean it out

    (a) Use the vacuum to remove all of the buffer and fluorophore from the channels

    (b) Fill the channels with DI water to clean out any residue left by the buffer and fluorophore

    (c) Use the vacuum to remove ALL of the DI water and then use the microscope to verify

20. Put everything away

(a) Put the buffer and fluorophore back in the refrigerator

(b) Put working MFDs back in the case and the nonworking ones into their container

(c) Throw away the used tips from the mechanical pipettes

(d) Tidy up the optical bench and surrounding area. Shorts can damage the system.

COMMON ROADBLOCKS:

- If you don't see a rise in signal level during step 14, you may not be getting flow. This can be caused by bubbles in the channel. This holds true for steps 16 and 17. Bubbles are your worst enemy.

- Lack of flow can also happen if the high voltage leads are placed incorrectly or incompletely. Make sure all leads are in the correct well and are all the way in the well.

- The data collection program can create prohibitively large files if left to run for long periods of time. Only turn on the data collection program right before you plan to run Insertion or Separation and make sure you turn it off as soon as you have finished with it. Even if you plan on running Insertion and Separation more than once, reset the data collection between runs to create a new file for each run.

## 4.2 Data Analysis Process

After the data is collected following the steps in Section 4.1, the next step is to run the signal processing on the data in order to eliminate unwanted frequencies and get the signal to noise ratio (SNR) of the run. In order to accomplish this, the MATLAB script in Appendix E was written in order to low pass filter the signal, find the standard deviation of the noise, find the value of the peaks and find the average DC voltage level. Each of these components is critical in order to get the parameters so the SNR can be calculated using Equation 4.1.

$$SNR = \frac{V_{PEAK} - V_{DC}}{\sigma} \tag{4.1}$$

The equation uses the height of the peak above DC divided by the standard deviation of the noise in order to calculate the SNR for that peak. All of these values are outputs of the script which

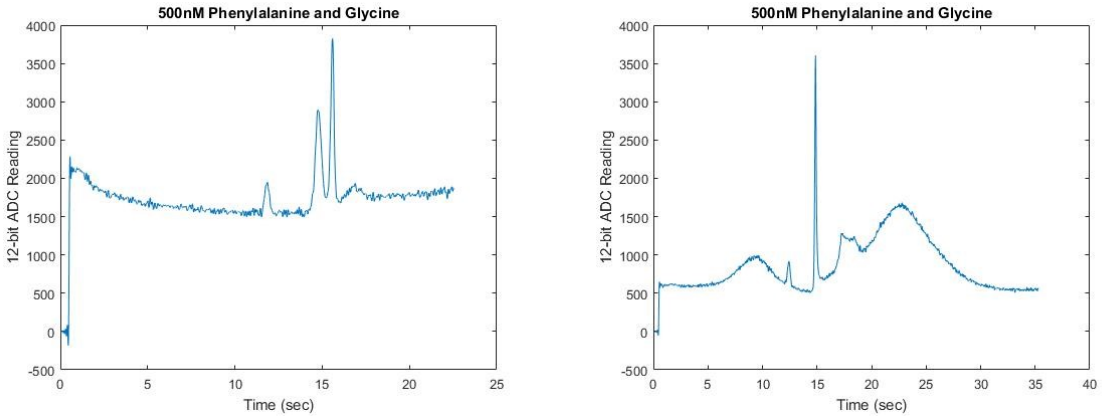



Figure 4.1: Two runs with 500nM Phenylalanine and Glycine

makes the calculation simple to perform after running the script. With the procedure and the data analysis in place, the system was ready to be characterized.

## 4.3 System Results

In order to characterize the system a fluorophore needed to be chosen that matched both the laser spectrum and had more than one peak. The fluorophore selected for use was AF532. This compound was selected for two reasons. First, it exhibits two peaks for its two forms, binding and non-binding. The non-binding form is a byproduct when the binding form of AF532 breaks down over time. The two forms have different electrophoretic mobilities, allowing them to separate with the applied voltage on the channel. Second, the excitation spectrum of AF532 overlaps very well with the emission spectrum of the laser. The tests were originally run with AF532 being bound to two amino acids, phenylalanine and glycine. These two amino acids have different molecular weights allowing them to be easily separated and detected resulting in three total peaks seen in some of the following graphs.

Testing on the system used concentrations ranging from a 1$\mu$M down to 10nM from which consecutive runs data was pulled to show the inconsistencies that inhibited obtaining repeatable results from the system. Figure 4.1 shows two runs that exhibited signal using a concentration of 500nM. There are two things that should be immediately apparent. First is the DC level of the graphs are shifting and at significantly different levels between the two runs. The one on the left is sitting around an average of 1600 when the peaks come out which equates to about 1.3V out of

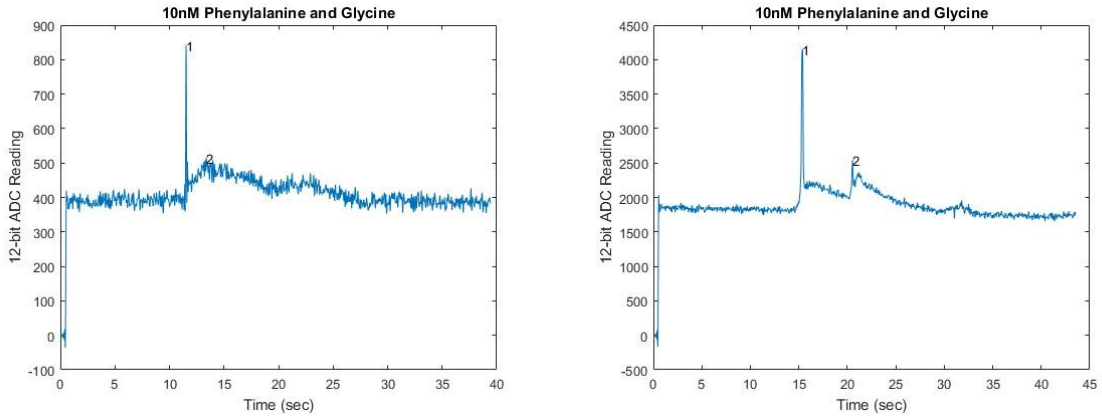



Figure 4.2: Two runs with 10nM Phenylalanine and Glycine

the 3.3V reference used to denote the full range at 4096. The graph on the right has a starting DC level of about 600 or about 0.5V. The DC levels are also fluctuating. The cause of this fluctuation is the laser which is when the black box problem of the laser became clearer. The laser was not just a lone diode inside the housing but has some internal circuitry as well. This circuitry was not adjustable and was negatively interacting with the current source used to drive the laser. The second issue with these plots is the difference in both peak sharpness and height. The height of the tallest peak in the left plot was about 2000 above the noise floor while the height of the tallest peak in the right plot was about 3000. The taller peak of the right plot was also much sharper than the left plot. The second plot only shows one peak which is odd because there should be a second peak for the labeled phenylalanine.

The two graphs in Figure 4.2 illustrate a different phenomenon experienced when using the laser, that of changing brightness. Sometimes when running the laser would seemingly at random intervals change intensity similar to switching the high-beams on in a car. This would exhibit a change in magnitude of both the DC and peak height but they would follow the same ratio. To illustrate Figure 4.2 has the first graph on the left with a DC level of about 400 and a peak height of about 850. The graph on the right shows a DC level of about 1900 and a peak height of about 4050. Both have a ratio of peak height to DC level of about 2.13. The effective output power of the laser jumped almost fivefold between runs.

Although these runs show some results most of the runs completed came up with nothing despite having visibly identical focus and alignment within a channel even with higher concentra-

tions. This was verified by checking the separation well and seeing fluorophore within the reservoir after the run was completed. These runs depicted and the ones not shown but run over the course of the work put in led to the discovery of all of the issues discussed in the next section.

## 4.4 System Issues

Over the course of the research for this system design and implementation, many issues arose. Those issues include the XYZ problems, the 3D printed part problems, the laser issues, the PMT voltage issues, and non-ideal optics placement capabilities.

The motorized XYZ stage exhibited a noticeable shift in the sample y stage during one instance of collecting data. When looking into the cause of this the stage had 1.5mm of uncontrollable travel that could take place as described in Section 3.7 of the XYZ Stage. Given that the channels are only 50$\mu$m or 1/30 of the 1.5mm of uncontrollable travel, this means that even a minor slip in the stage could cause a large enough shift of the focal point of the laser out of the channel. This is a huge inconsistency that inhibits repeatability in the alignment of the channel. When checking the PMT side of things they can only travel in 1 direction due to a spring being the only thing holding the movable slide in place. When trying to verify how much each stage could shift side to side, which was controlled by the tolerance of the slide mechanism, the results only showed significant deviation in the y stage. This was verified using the laser on a device to see if the laser lost alignment with the channel.

The variations in the spring loaded manual stages for the PMT holder may have also caused the pinhole to move off of the maximum point and lose signal. This loss of alignment was most noticeable due to the weight on the z stage. Because the manual stages are spring loaded, the weight of the PMT in the PMT holder limited the effectiveness of the spring. The stage could stick until friction was overcome or the spring stretched far enough to start it moving again. This has two root causes, lack of sufficient lubrication and diminished spring function. These errors led to a range of distances moved without actually adjusting the stages adjustment controls.

After checking the XYZ stages for motion, the 3D printed parts were checked for deviations away from the designed shape. Of the three parts, the PMT holder was the best suited for the task as it only moved a few mm at most with a significant force beyond normal conditions. The optical bench could deviate from the rotational point formed at the optical breadboard and base plate.

This would cause the laser beam when initially focused on the center of a device to move beyond the edges of the device with minimal force. The sample holder had the most deflection in the Z direction which not only would move the device in and out of focus but rotate the device out of the horizontal plane changing the lasers angle of the instance.

The laser issues were demonstrated with data in Section 4.3. What was learned from this is that the cost savings gained from an undocumented laser are not worth the price and headaches caused.

The last issue was the PMT voltages. When the output voltage on the PMT would rise above about 2.8V the Vref output that was used to drive the control voltage would drop and decrease the gain. This was an issue until the ground lift circuit was implemented as discussed previously.

Aside from the aforementioned issues, the optics could have been another source of loss and noise. The optical components were all placed inside of receptacles that were 3D printed into the optical bench. Each hole has some surface roughness due to the 3D printer tolerances. The lenses had to be glued into place as the holes for them were too large to hold them where they needed to be to capture the laser light. The insufficient coverings built into the 3D printed optical bench caused visible noise as seen in the real-time data. This has two causes including the not fully opaque printed material and the minimalist design to the print. When these open regions were covered the noise floor was reduced.

## 4.5 Necessary Changes

Based on the issues presented in Section 4.4 there are a number of key components that need to be changed in order for the system to function properly. These include the laser, PMT controller, optical path subsystem, and XYZ stages.

The changes suggested for the laser will improve both the usability and the repeatability of performance. The suggested paths that can be taken to improve performance include selecting a laser diode with no circuitry and using a driver designed to work with it or by buying a laser diode with a corresponding built-in driver to ensure compatibility. Adequate documentation for the laser would also allow the user to understand what is happening and potentially eliminate the manual variation in switching output power levels of the laser through the current source designed for the system.

By switching to a dedicated PMT controller designed by the same company specifically for this model, the currents necessary and the tolerances on the voltages will be as intended by the manufacturer. This will eliminate the Vref and PMT gain sag that occurred when the output was being driven toward its upper limits. This change would also isolate the voltages from having the same reference ground as the signal effectively isolating the gain controls, and giving tighter control and stability on PMT's voltages.

The optical path in the way it was designed is insufficient for the job it needed to accomplish. The change to this subsystem and the added time it takes to use the setup is necessary in order to be comparable to the standard system. The microscope is almost a necessary part of the system for the many added benefits that it provides. These benefits include precise alignment of optics, enclosed optical path, magnification for alignment, and only one set of manual controls for precise movement of the device in the optical path.

The XYZ stages made it much more difficult to use the system both in running and calibration. The time it took to move the same distance as the standard system was an order of magnitude longer. Between the time and the deficiencies in the stages, these should be replaced with the microscope without a second thought.

Chapter 5 concludes this thesis by proposing the best option based on the data collected and analysis performed over the course of the research.

# CHAPTER 5. CONCLUSION

The hypothesis this thesis set out to investigate was "Can the cost of the system be reduced without losing the performance of the system?" The conclusion reached is a conditional yes. As shown in Chapter 4 the system was able to obtain data, but with degredation of performance. This goes to show that the implementation of the system was not of a high enough caliber in order to provide both consistency of results and maintain the performance of the standard system specifications.

In order to bring the low-cost system up to the original system's specifications some systems can be kept while others need to be modified. The first section will summarize why the conclusion to the hypothesis was a conditional yes by covering the main changes introduced into the system's runtime procedure that were problematic. The second section will discuss the systems that were functional and can be kept unmodified in a future implementation. The third section will discuss the systems that should be changed, why they should be changed, and what the recommended change is based on the experience with the iterations used. The fourth section will cover what the recommended system would cost and provide places that could receive further cost reduction if the engineering resources are available. The last section summarizes the thesis.

## 5.1 Problems Introduced With Implementation Chosen

As demonstrated in Chapter 4, the system did not meet the specifications of being comparable in operation to the standard system. There were reason for the failure which mainly included the mechanical issues of the 3D printed parts, the tolerances of the motorized linear stages, the laser's output power instability, and the light penetration into the optical path corrupting the PMT signal. All of these issues, except the laser, stem from the first step taken in creating the system, that of removing the microscope.

When the microscope did not meet the inexpensive criteria for the system, a substitute for all of the functions it performed needed to be implemented. This included alignment of optical components, the device, and the PMT. 3D printing replaced the mechanical aspects of the microscope and XYZ stages acted in place of the microscope's alignment mechanisms. The mechanical aspects of the 3D prints can be further modified to decrease unwanted motion and stray light corrupting the system for improvements. The largest change caused by removing the microscope was the two XYZ stages that replaced the alignment mechanisms in the microscope. One XYZ stage held the PMT and PMT holder with the three axes needed to align the PMT to the fluoresced light coming out of the optical bench. The other XYZ stage aligns the device in the sample holder to the laser coming from the optical bench as well as the fluoresced light coming from the device.

Although the stages would work with a few upgrades the flaws in the new implementation made the system hard to get and to maintain a functioning state. The main issue stemmed from maintaining alignment of the optical path. Each component had to be meticulously aligned in order to keep the laser or fluoresced light going in the right direction. The nature of the laser beam quality and size of lenses used made it difficult to create a precise alignment. There is also the aspect of having three separate parts to align that are not mechanical secured like in the standard system's microscope. These features made it much more time consuming to align the low cost system and maintain it for more than a few consecutive runs. Other than the microscope and the changes it caused, the laser's output power stability was the other major issue that prevented consistency in the system's performance. The laser's output exhibited drastic deltas in output power at a given current setting. For example, the output would be one brightness to the eye and then all of a sudden it would switch to be approximately four times as bright or almost non-existent without changing a setting. The inconsistencies in operational interactions between the integrated internal driver that had no available controls and the external driver with the necessary controls caused difficulty even within a run to obtain consistent results.

Besides the output power issue of the laser the laser was also spotty; as the beam was not a Gaussian beam shape as one would expect from a laser. The beam's pattern when examined on a piece of paper coming from the optical bench exhibited what could be described a bunch of dots with intensities that look Gaussian. The lack of uniformity led to the conclusion that the laser purchased for the experiments was not sufficient to meet the performance desired for the system.

## 5.2 Functional Systems

When evaluating the designed subsystems after not obtaining consistent results some met the required specifications in order to work if used to replace the corresponding piece of the standard system. The three systems that could be used were the high voltage system, the data acquisition and the corresponding software.

The high voltage system is not only easy to implement but the corresponding software was extremely useful for collecting the data and storing the run variables. The overall system functioned while only costing 27% of the power supplies and switching box used for the standard system. The voltages and currents had real-time feedback for detecting if a bubble occurred in the channel or if a voltage was at the wrong level similar to the standard system. The teensy used to control the high voltage system is also used to acquire the data.

The data acquisition is sufficient although it needed to be limited to 3.3V from the 4V maximum output of the PMT in order to stay within operating limits.

The software written has no issues and works well when communicating with both the Teensy and the Arduino. In order to keep the two python scripts on the computer functioning without problems each microcontroller has a single python script that it connects to through a unique COM port on the computer.

## 5.3 Scope (Summary of Contributions)

The systems not mentioned in the previous section are in need of some work in order to bring the system into alignment with the specifications. The subsystems and the recommended work needed will be discussed in this section. The subsystems in need of further work are the optical path, laser, and voltage limiting circuit protecting the Teensy's ADC.

The optical path is really the critical piece to the whole system that needs the most reworking. The system not being able to obtain consistent results on the same concentration all stem from this subsystem. The microscope in the standard system is not only significantly easier to use it provides many benefits. Those benefits are ease of PMT calibration, rigid calibrated optical path, minimizes light corrupting the optical path, and much simpler to operate. Due to the user having to use a microscope to check for bubbles when filling the device it makes sense to build that into the

system instead of having the user add on the cost of a microscope for that purpose. The low-cost system has to be calibrated to the individual device as well as to the specific detection spot in that device due to the inconsistencies in the planarity of the device holder.

If everything was working great in the optical path the laser would still be causing issues for the run. As shown in the hour test in Chapter 3, the laser is not stable until it has been on for over 30 minutes. This causes the two issues of diminished useable laser lifetime and questionable stability when changing current settings on the driver. The most obvious solution would be to spend the money on the nicer laser used in the standard system. It has an included controller and power supply which make it much easier to implement into the system. The issue with this in the current low-cost system is that the whole system would have needed a redesign due to the laser's size. This is due to the slot that was built into the optical bench for a laser with a specific diameter. If the newly selected laser was not the same diameter then the part would need a redesign.

If the laser was not an issue then the other problem would be the limiting circuit. This circuit adds in a DC offset which could not be removed without dropping the gain from unity. Although a limiter is needed with the Teensy as the ADC there are two options for moving forward. They are replacing the limiter with a Zener diode and resistor combination, using an external ADC that has the necessary range with the Teensy, or replace the Teensy with a 5V microcontroller that can handle the necessary sampling rate.

## 5.4 Recommended System Pricing

Using the recommendations in the previous sections of this chapter, Table 5.1 was compiled to give an estimate of the cost of a functional system by taking the working parts of each system and combining them. The laser is direct from the standard system. The only change to the standard system optical path is the table was replaced with a breadboard as the space needed is much smaller than the table and vibrations were never an issue for the microscope used in our lab. The high voltage system is the one designed and discussed in this thesis. Using the suggestion of just adding a Zener diode and resistor as the limiter on the output of the PMT is the only added cost and design change to the PCB of the high voltage system. This is estimated as one dollar as the two parts are inexpensive. The only python script needed at this point would be the python data logger code in

Appendix B. All of this comes out to the total of $21,322.63 which is 56% of the standard systems cost.

Table 5.1: Recommended system cost

| System | Recommended |
|---|---|
| Laser | $1,004.00 |
| Optical Path | $19,481.00 |
| High Voltage | $836.63 |
| Filter and Data Acquisition | $1.00 |
| Software | $0.00 |
| Total | $21,322.63 |

## 5.5 Conclusion

The results presented throughout this thesis show which subsystems of a CE-LIF system have a working low-cost alternative. The subsystems, which did not have working low-cost versions, present why they did not work and if based on the results further developed should be pursued or if they should revert to the standard system configuration. The microscope is the main part of the standard system that is highly recommended to keep in the system due to making it very straightforward to use. This ease of use and consistency in the systems performance provides the ability to garner repeatable and consistent results. While a full open-source system is not available from this thesis, the documentation is here for the reader to use as a part of their system or develop further if they desire to do so. The other aspects needed to operate this system including device fabrication and sample preparation will need to be provided for in order to obtain results. Using this basis for a low-cost system the desire is that this research will provide the means to break into the field or improve in another way upon the standard CE-LIF system to further advance the capabilities of the field through the results it produces and conclusion drawn.

## REFERENCES


[1] FOSCO., "Laser diode characteristics," Online, March 2016. ix, 32

[2] Semrock, "535 nm edge brightline©single-edge epi-fluorescence short-pass dichroic beamsplitter," Online, 2018. ix, 36

[3] E. Optics, "550nm 12.5mm dia., high performance longpass filter," Online, 2018, curv_62977.pdf. ix, 37

[4] P. Guy, ""brute force" parallel of power supplies," Online, September 2013, curv_62977.pdf. ix, 40

[5] Hamamatsu, "Photosensor modules h10722 series," Online, 2018, h10722 datasheet. ix, 43

[6] ——, "Photosensor modules h10722 series," Online, 2018, h10722 datasheet. ix, 44

[7] ——, "Photosensor modules h10722 series," Online, 2018, h10722 datasheet. ix, 44

[8] ——, "Low power, 8th order progressive elliptic, lowpass filter," Online, 1996, lTC1069-1 datasheet. ix, 48

[9] P. Camilleri, *Capillary Electrophoresis: Theory and Practice*. Boca Raton, Florida: CRC Press LLC, 1998, vol. 2. 4, 5

[10] P. M. Parker, *Laser-induced Fluorescence: Webster's Timeline History, 1973 - 2007*. Boca Raton, Florida: ICON Group International, Inc., 1998, vol. 2. 4

[11] C. Today, "Advantages and disadvantages of capillary electrophoresis," Online, November 2014. 5

[12] M. Panjehpour, C. E. Julius, M. N. Phan, T. VoDinh, and S. Overholt, "Laserinduced fluorescence spectroscopy for in vivo diagnosis of nonmelanoma skin cancers," *Lasers in Surgery and Medicine*, vol. 31, pp. 367–373, 2002. 5

[13] P. M., S. R., F. D., B. M., O. S., H. W., and O. B., "Quantification of phthalocyanine concentration in rat tissue using laser-induced fluorescence spectroscopy," *Lasers Surg Med*, vol. 13, pp. 23–30, 1993. 5

[14] I. van Cruyningen, A. Lozano, , and R. Hanson, "Quantitative imaging of concentration by planar laser-induced fluorescence," *Experiments in Fluids*, vol. 10, pp. 41–49, 1990. 5

[15] J. Bioscience, "Atto 532 protein labeling kit," Online, 2018. 6

[16] C. Y., D. H., Z. L., and C. G., "Fabrication of pmma ce microchips by infrared-assisted polymerization," *Electrophoresis*, vol. 29, pp. 4922–4927, 2008. 6

[17] P. Abgrall, L. Low, and N. Nguyen, "Fabrication of planar nanofluidic channels in a thermoplastic by hot-embossing and thermal bonding," *Lab on a Chip*, vol. 7, pp. 520–522, 2007. 24

[18] A. MicroSystems, "Dmos microstepping driver with translator and overcurrent protection," Online, 2013, a4988 datasheet. 42

## APPENDIX A. ARDUINO UNO CODE

```
#include <AccelStepper.h>
#include <SPI.h>

// Digital Pot Initialization
const int CS1 = 10;
const int CS2 = 9;
int PMTVoltage = 1;
int GAINvalue = 0;
int PMTvalue = 128;

// Motor Parameters
int motorSpeed = 1200;  // maximum steps per second
                        // about 3rps at 16 microsteps
int motorAccel = 80000;

// X Motor Setup
int motor_x_DirPin = 2;  // digital pin 2
int motor_x_StepPin = 3; // digital pin 3
int motor_x_dir = -1;    // reverse direction if need be

// Y Motor Setup
int motor_y_DirPin = 4;  // digital pin 2
int motor_y_StepPin = 5; // digital pin 3
int motor_y_dir = -1;    // reverse direction if need be
```

```
// Z Motor Setup
int motor_z_DirPin = 6;  // digital pin 2
int motor_z_StepPin = 7; // digital pin 3
int motor_z_dir = -1;    // reverse direction if need be

// Motor Initialization
AccelStepper motor_x(1, motor_x_StepPin, motor_x_DirPin);
AccelStepper motor_y(1, motor_y_StepPin, motor_y_DirPin);
AccelStepper motor_z(1, motor_z_StepPin, motor_z_DirPin);

String incommingCommand;
String commandString;
int distance;

void processCommand(String in) {
        commandString = in.substring(0,6);
        distance = in.substring(6).toInt();

        if(commandString == "1:x:+:") {         // X positive
                motor_x.move( distance * motor_x_dir);
        } else if(commandString == "1:x:-:") { // X Negitive
                motor_x.move( distance * -1 * motor_x_dir);
        } else if(commandString == "1:y:+:") { // Y Positive
                motor_y.move( distance * motor_y_dir);
        } else if(commandString == "1:y:-:") { // Y Negitive
                motor_y.move( distance * -1 * motor_y_dir);
        } else if(commandString == "1:z:+:") { // Z Positive
                motor_z.move( distance * motor_z_dir);
                motor_y.move( distance * motor_z_dir * .37);
```

```
            // Y factors in laser angle offset
        } else if(commandString == "1:z:-:") { // Z Negitive
            motor_z.move( distance * -1 * motor_z_dir);
            motor_y.move( distance * -1 * motor_z_dir * .37);
            // Y factors in laser angle offset
        } else if(commandString == "1:g:+") { // Gain up
            MCP41010Write(CS1, ++GAINvalue);
        } else if(commandString == "1:g:-") { // Gain down
            MCP41010Write(CS1, --GAINvalue);
        } else if(commandString == "1:p:+") { // PMT Voltage up
            MCP41010Write(CS2, ++PMTvalue);
        } else if(commandString == "1:p:-") { // PMT Voltage down
            MCP41010Write(CS2, --PMTvalue);
        }
}

void checkSerialCommand() {
        if (Serial.available() > 0) {
            // get incoming byte:
            int inByte = Serial.read();

            if( (char)inByte == '\n' ) {
                processCommand(incommingCommand);
                incommingCommand = "";
            } else {
                incommingCommand += (char)inByte;
            }
        }
}
```

```
void MCP41010Write(int CS, byte value) {
        // Note that the integer vale passed to this subroutine
        // is cast to a byte

        digitalWrite(CS,LOW);
        SPI.transfer(B00010001); // Tells IC to set pot
        SPI.transfer(value);     // Tell IC pot's value
        digitalWrite(CS,HIGH);
}

void setup() {
        motor_x.setMaxSpeed(motorSpeed);
        motor_x.setSpeed(motorSpeed);
        motor_x.setAcceleration(motorAccel);
        // motor_x.moveTo(1000);

        motor_y.setMaxSpeed(motorSpeed);
        motor_y.setSpeed(motorSpeed);
        motor_y.setAcceleration(motorAccel);

        motor_z.setMaxSpeed(motorSpeed);
        motor_z.setSpeed(motorSpeed);
        motor_z.setAcceleration(motorAccel);

        pinMode (CS1, OUTPUT);
        pinMode (CS2, OUTPUT);
        SPI.begin();

        Serial.begin(9600);
        Serial.print("begin\n");
```

```
}

void loop() {

        motor_x.run();
        motor_y.run();
        motor_z.run();

        MCP41010Write(CS1, PMTvalue);
        // Serial.println("PMT Voltage: "+analogRead(PMTVoltage));
        MCP41010Write(CS2, GAINvalue);
        // Serial.println("Gain: ~" + GAINvalue*10/256);

        checkSerialCommand();

}
```

## APPENDIX B. PYTHON DATA INTERFACE CODE

```
#HV Control &
#Read and Plot from the PMT

#This code records the data received into the Teensy's ADC.
#Includes HV controls and replotting the results at the end.

#See CSV Dataplot notebook to plot old experiment data.

from pyqtgraph import QtGui, QtCore #PyQt4's UI design library
import pyqtgraph as pg              #Initiation of plotting code
import serial                       #Serial port communication
from datetime import datetime       #Current date and time
from time import strftime           #For time formatting
import numpy as np
from Tkinter import *               #For the notes prompt
import json                         #For saving tags
import threading                    #For multithreading
import Queue
from collections import deque
import pprint                       #For pretty debug printing
import binascii
import struct
import ufluidics_dsp as udsp        #For DSP processing
import sys
```

```
import getopt

def usage():
    print(These options are available from the command line:)
    print(-h / --help          Displays this usage information.)
    print(-f / --filter        Enables realtime filtering)
    print(                     (disabled by default).)
    print(-n / --nosave        Disables data saving, including)
    print(                     the logfile (enabled by default).)
    print(                     Also turns off the window that)
    print(                     regraphs data after the first one)
    print(                     is closed, since it reads from)
    print(                     the savefile.)

#determines if data will undergo realtime filtering (True).
filterData = False
#determines if data will be saved (True). Also determines if the
#data will be regraphed in the second window after the first one
#is closed, since it reads from the savefile. This might be worth
#changing later.
saveData = True

try:
    opts, args = getopt.getopt(sys.argv[1:], "hfn", ["help",
        "filter", "nosave"])
except getopt.GetoptError as e:
    usage()
    sys.exit()
for opt, arg in opts:
    if opt in ("-h", "--help"):
```

```
            usage()
            sys.exit()
        elif opt in ("-f", "--filter"):
            filterData = True
        elif opt in ("-n", "--nosave"):
            saveData = False

## Initialize the container for our voltage values read in from
## the teensy IMPORTANT NOTE: The com port value needs to be
## updated if the com value changes. It's the same number that
## appears on the bottom right corner of the window containing
## the TeensyDataWrite.ino code If someone forgot to change the
## COM port here, this will catch the error and ask the user for
## the correct port. Then you should change the portName.
portName = "COM5"
portOpen = False
validInput = False
while not portOpen:
    try:
        teensySerialData = serial.Serial(portName, 115200)
        #port successfully opened
        portOpen = True
        if validInput:
            print(Please open the code and edit the COM port.)
    except serial.serialutil.SerialException as e:
    # Port failed to open
        print e
        while not validInput:
            port = raw_input(Try another port? (n/port number): )
            if port[0] == 'n':
```

```
            #check if the user wants to exit
            sys.exit()
        elif port[:4].upper() == "COM ":
            # check if the user entered "COM #"
            portName = port[:3].upper() + port[4:]
            validInput = True
        elif port[:3].upper() == "COM":
            # check if the user entered "COM#"
            portName = port.upper()
            validInput = True
        else:
            # The user entered a number or an invalid input
            try:
                # check if the input was a number
                portNum = int(port)
            except ValueError:
                # bad input, try again.
                print("Invalid input.")
            else:
                # loop back and try valid new port
                portName = "COM" + str(portNum)
                validInput = True

## Always start by initializing Qt (only once per application)
app = QtGui.QApplication([])

## Define a top-level widget to hold everything (a window)
w = QtGui.QWidget()
w.resize(1000, 600)
w.setWindowTitle('Voltage Plots')
```

```
startBtnClicked = False
calBtnClicked = False

csvFile = None
logFile = None

## This function is the response when the start button is clicked
def startButtonClicked():
    global startBtnClicked
    global startBtn
    global saveData
    global csvFile
    global logFile
    if (startBtnClicked == False):
        teensySerialData.flushInput() #empty buffer for input
        startBtnClicked = True
        startBtn.setText('Stop')

        ## Save file to Documents/IPython Notebooks/RecordedData
        i = datetime.now()
        timeString = str(i.year) + "-" + str(i.month) + "-" + \
            str(i.day) + " at " + str(i.hour) + ":" + \
            str(i.minute) + ":" + str(i.second)
        if saveData:
            csvFile = open(folderName + fileTime + '.csv', 'a')
            csvFile.write("# Data from " + timeString + '\n')
            csvFile.write("Timestamp,PMT\n")

            ## File for logging
```

```
            logFile = open(folderName + fileTime + '.log', 'a')
            logFile.write('# Log from ' + timeString + '\n\n')
            logFile.write('# DSP Processing Report:\n')
            logFile.write(udsp.PROCESSING_TYPE)
            logFile.write('\n# Event: timestamp (from teensy)\n')

        #start threads
        serial_read_thread = serialReadThread()
        serial_read_thread.daemon = True
        serial_read_thread.start()
        time_data_thread = timeDataThread()
        time_data_thread.daemon = True
        time_data_thread.start()
        if (filterData):
            pmt_data_thread = pmtDataThread()
            pmt_data_thread.daemon = True
            pmt_data_thread.start()
        else:
            pmt_data_thread = pmtDataThread_LogOnly()
            pmt_data_thread.daemon = True
            pmt_data_thread.start()
        if saveData:
            data_write_thread = dataWriteThread()
            data_write_thread.daemon = True
            data_write_thread.start()

        saveDataChk.setDisabled(True)

    elif (startBtnClicked == True):
        startBtnClicked = False
```

```
        startBtn.setText('Start')
        startBtn.setDisabled(True)

## At the end of the update function, check if quitBtnClicked
def quitButtonClicked():
    ## Close the file and close the window.
    if (startBtnClicked == False):  ##don't quit while running.
        w.close()

        if saveData:
            csvFile.close()
            #showNow = True
            data = np.loadtxt(open(folderName + fileTime + \
                '.csv' ,"rb"),delimiter=",",skiprows=2)
            numSamples2 = data.shape[0]
            if(len(data) > 0):
                pmtCurve2.setData(data[:,1])

def quitButtonClicked2():
    w2.close()

## Buttons to control the High Voltage
def HVoffButtonClicked():
    teensySerialData.write('0')
    if saveData:
        logFile.write('High Voltage Off: '+str(timeElapsed)+'\n')
    print("HV Off")

def HVonButtonClicked():
    teensySerialData.write('1')
```

```
    if saveData:
        logFile.write('High Voltage On: '+str(timeElapsed)+'\n')
    print("HV On")

def insertionButtonClicked():
    teensySerialData.write('3')
    if saveData:
        logFile.write('Sample Insertion: '+str(timeElapsed)+'\n')
    print("Insertion")

def separationButtonClicked():
    teensySerialData.write('2')
    if saveData:
        logFile.write('Sample Separation: '+str(timeElapsed)+'\n')
    print("Separation")

## Button for Calibration
def calibrateButtonClicked():
    global calBtnClicked
    if saveData:
        logFile.write('Signal Calibration: ' + str(timeElapsed)\
                + '\n')
    calBtnClicked = True

def saveDataCheckboxClicked():
    global saveData
    saveData = saveDataChk.isChecked()

# Start Recording in Widget
## Create widgets to be placed inside
```

```
startBtn = QtGui.QPushButton('Start')
startBtn.setToolTip('Click to begin graphing')
# This message appears while hovering mouse over button

quitBtn = QtGui.QPushButton('Quit')
quitBtn.setToolTip('Click to quit program')

quitBtn2 = QtGui.QPushButton('Quit')
quitBtn2.setToolTip('Click to quit program')

HVonBtn = QtGui.QPushButton("HV on")
HVonBtn.setToolTip('Click to turn the high voltage on')

HVoffBtn = QtGui.QPushButton("HV off")
HVoffBtn.setToolTip('Click to turn the high voltage off')

insBtn = QtGui.QPushButton("Insertion")
insBtn.setToolTip('Click to start insertion (#3)')

sepBtn = QtGui.QPushButton("Separation")
sepBtn.setToolTip('Click to start separation (#2)')

calBtn = QtGui.QPushButton("Calibrate")
calBtn.setToolTip('Clock to callibrate signal sample')

# Checkbox to determine if incoming data should be saved
saveDataChk = QtGui.QCheckBox("Save data")
saveDataChk.setChecked(saveData)
saveDataChk.stateChanged.connect(saveDataCheckboxClicked)
```

```
## Functions in parantheses are called when buttons are clicked
startBtn.clicked.connect(startButtonClicked)
quitBtn.clicked.connect(quitButtonClicked)
quitBtn2.clicked.connect(quitButtonClicked2)
HVonBtn.clicked.connect(HVonButtonClicked)
HVoffBtn.clicked.connect(HVoffButtonClicked)
insBtn.clicked.connect(insertionButtonClicked)
sepBtn.clicked.connect(separationButtonClicked)
calBtn.clicked.connect(calibrateButtonClicked)

## xSamples is the maximum amount of samples graphed at a time
xSamples = 3000

if saveData:
    ## To plot the entire csv file after the experiment.
    ## Plots at the click of the quit button

    ## Define a top-level widget to hold everything
    w2 = QtGui.QWidget()
    w2.resize(1000,600)
    w2.setWindowTitle('Voltage Plot')

    #Create Plot Widgets
    pmtPlotWidget2 = pg.PlotWidget()
    pmtPlotWidget2.setSizePolicy(QtGui.QSizePolicy.Expanding, \
        QtGui.QSizePolicy.Expanding)
    pmtPlotWidget2.setYRange(0, 4096)
    pmtPlotWidget2.setLabel('top', text = "PMT")
    pmtCurve2 = pmtPlotWidget2.plot()
```

```
    lowerBoundValue2 = 0
    upperBoundValue2 = xSamples
    pmtPlotWidget2.setXRange(0, upperBoundValue2)

    ## Create grid layout to manage the widgets size and position
    layout2 = QtGui.QGridLayout()
    w2.setLayout(layout2)

    ## Add widgets to the layout in their proper positions
    layout2.addWidget(quitBtn2, 0, 0)
    layout2.addWidget(pmtPlotWidget2, 1, 2, 1, 1)
        # wGL goes on right side, spanning 3 rows

    ## Display the widget as a new window
    w2.show()

## Create plot widget for PMT signal input
## For real time plotting
pmtPlotWidget = pg.PlotWidget()
pmtPlotWidget.setYRange(0, 4096)
pmtPlotWidget.setXRange(0, xSamples)
pmtPlotWidget.setLabel('top', text = "PMT")
        #Title to appear at top of widget

if (filterData):
    # Create and initialize plot widget for processed signals
    PROCPLOT_TITLE = "Filtered Processed PMT Data"
    procplot_widget = pg.PlotWidget()
    procplot_widget.setYRange(0, 1)
```

```
    procplot_widget.setXRange(0, xSamples)
    procplot_widget.setLabel('top', text=PROCPLOT_TITLE)
    #procplot_curve = procplot_widget.plot()
        # Check with this line it may be done later in code

## Create a grid layout to manage the widgets size and position
## Grid layout allows a widget to be located in a column and row
layout = QtGui.QGridLayout()
w.setLayout(layout)

## Add widgets to the layout in their proper positions
## The format is (button, row, column)
layout.addWidget(quitBtn, 0, 0)
layout.addWidget(startBtn, 3, 0)
layout.addWidget(HVonBtn, 0, 2)
layout.addWidget(insBtn, 3, 2)
layout.addWidget(sepBtn, 4, 2)
layout.addWidget(HVoffBtn, 5, 2)
layout.addWidget(calBtn, 6, 2)
layout.addWidget(saveDataChk, 3, 1)

layout.addWidget(pmtPlotWidget, 1, 1)
if (filterData):
    layout.addWidget(procplot_widget, 2, 1)

## Display the widget as a new window
w.show()

## Initialize all global variables
```

```
## Whenever we plot a range of samples, xLeftIndex is the x value
## on the PlotWidget where we start plotting the samples,
## xRightIndex is where we stop These values will reset when
## they reach the value of xSamples
xRightIndex = 0
xLeftIndex = 0

## Used to determine how often we plot a range of values
graphCount = 0

## Time values in microseconds read from the teensy are stored in
## these variables. Before timeElapsed is updated, we store its
## old value in timeElapsedPrev
timeElapsed = 0L
timeElapsedPrev = 0L

## Determines if this is the first run through the update loop
firstRun = True

## Create a new file, with the name being today's date and
## current time and write headings to file in CSV format
fileTime = strftime("%Y%m%d_%H%M%S")
folderName = 'RecordedData\\'

# Change this to match the value in the Teensy's
# "timer0.begin(SampleVoltage, 110);" line
usecBetweenPackets = 1000.0

packetsRecieved = 0L
```

```
# Data structures for data from Teensy
recieved_data = deque()
time_data = deque()
pmt_data = deque()
time_write = deque()
pmt_write = deque()
graph_sema = threading.Semaphore()

filter_block = []
time_block = []
pmt_filtered = []
pmt_thresholded = []
pmt_calibrate = []

pmt_baseline = 0.0
pmt_threshold = 100.0 #default threshold value
startTime = 0L;
endTime = 0L;

sample_detected = False
high_start = -1

class serialReadThread (threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
    def run(self):
        ## Set global precedence to previously defined values
        global firstRun
        global startBtnClicked
        global time_data
```

```
global pmt_data
#inputBytes = []
while (startBtnClicked):
    #Reads the values from the serial port into
    #time_value, adc_value, and trash. In the ">LHH"
    #argument, the ">" tells it the data is big-endian,
    #the "L" tells it that time_value is an unsigned
    #long, and the "H"s tell it that adc_value and trash
    #are unsigned shorts. Trash is only there because
    #things break if 6 bytes are sent at a time.
    #It is then ignored.
    time_value, adc_value, trash = struct.unpack(">LHH",\
        teensySerialData.read(8))
    if (firstRun == True):
        ## Only run once to ensure buffer is flushed
        firstRun = False
        teensySerialData.flushInput()
        continue
    #Bytes read in and stored in a char array of size 6
    time_data.append(time_value)
        #Append time value to time_data deque
    time_block.append(time_value)
        #Append time value to time_block deque
    if saveData:
        pmt_write.append(str(adc_value))
        #Append string adc value to pmt_write deque
    pmt_data.append(adc_value)
        #Append adc value to pmt_data deque
    ##print out the recieved data in hex format
    # out = ""
```

```
            # for d in struct.unpack(">LHH", \
                #teensySerialData.read(8)):
                # out += ('%08X' % d) + " "
            # print(out)

class timeDataThread (threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
    def run(self):
        global timeElapsed
        global timeElapsedPrev
        global startBtnClicked
        global startTime
        global time_data
        global packetsRecieved
        while (startBtnClicked):
            while(not time_data):
                pass
            timeElapsedPrev = timeElapsed
            timeElapsed = time_data.popleft()
            if (timeElapsedPrev == 0):
                startTime = timeElapsed
                timeElapsedPrev = timeElapsed

            # We'll add all our values to this string until we're
            # ready to exit the loop, at which point it will be
            # written to a file
            if saveData:
                time_write.append(str(timeElapsed))
```

```
            ## This difference calucalted in the if statement is
            ## the amount of time in microseconds since the last
            ## value we read in and wrote to a file. If this
            ## value is significantly greater than 100, we know
            ## we have missed some values, probably as a result
            ## of the buffer filling up and scrapping old values
            ## to make room for new values. The number we print
            ## out will be the approximate number of values we
            ## failed to read in. This is useful to determine if
            ## your code is running too slow
            if (timeElapsed - timeElapsedPrev > \
                (usecBetweenPackets*1.5)):
                print("missed time: " + str((timeElapsed-\
                        timeElapsedPrev)/usecBetweenPackets))
                if saveData:
                    logFile.write('Missed Sample: ' +\
                        str(timeElapsed) + '\n')

            #Counts packets recieved to track missed packets
            packetsRecieved += 1

##This thread takes the pmt data from the pmt_data deque, filters
##it (to be implemented) and adds the filtered data to pmt_graph.
class pmtDataThread_LogOnly (threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
    def run(self):
        global startBtnClicked
        global pmt_data
        global pmt_filtered
```

```
        global xRightIndex
        while (startBtnClicked):
            while(not len(pmt_data) >= 2):
                pass
            numData = pmt_data.popleft()
            #numData = numData*3.3/1024
            #numDataRounded = numData #- numData%.001
                #Round voltage value to 3 decimal points
            graph_sema.acquire()
            pmt_filtered.append(numData)
            graph_sema.release()


class pmtDataThread (threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
    def run(self):
        global startBtnClicked
        global calBtnClicked
        global pmt_data
        global filter_block
        global time_block
        global pmt_filtered
        global pmt_calibrate
        global pmt_baseline
        global pmt_threshold
        global sample_detected
        global high_start
        while (startBtnClicked):
            while(not pmt_data):
                pass
```

```
filter_block.append(pmt_data.popleft())

if len(filter_block) >= udsp.FILTBLK_SIZE:
    graph_sema.acquire()
    filter_block = udsp.filter_signal(filter_block)
    if calBtnClicked:
        pmt_calibrate.extend(filter_block)
    filter_block[:] = [x - pmt_baseline for x in\
            filter_block]
    pmt_filtered.extend(filter_block)
    filter_block[:] = [x > pmt_threshold for x in\
            filter_block]
    filter_block = udsp.filter_threshold(filter_block)
    pmt_thresholded.extend(filter_block)
    detections, high_start = udsp.detect_sample(\
            filter_block, time_block, high_start)
    filter_block = []
    time_block = []
    graph_sema.release()

    while detections:
        peak, width = detections.pop()
        if saveData:
            logFile.write("Sample Detected: " +\
            str(peak))
        print("Sample Detected: " + str(peak))
        if saveData:
            logFile.write("Sample Width: " +\
            str(width))
```

```
                print("Sample Width: " + str(width))

            if calBtnClicked and pmt_calibrate:
                if len(pmt_calibrate) >= udsp.CALIBRATION_WIDTH:
                    pmt_baseline, pmt_threshold = udsp.calibrate(\
                        pmt_calibrate)
                    pmt_calibrate = []
                    calBtnClicked = False
                    print "PMT Baseline:", pmt_baseline, "PMT \
        Threshold:", pmt_threshold

class dataWriteThread (threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
    def run(self):
        global startBtnClicked

        while (startBtnClicked):
            while(not time_write or not pmt_write):
                pass
            localTimeElapsed = time_write.popleft()
            pmtNumDataRounded = pmt_write.popleft()

            stringToWrite = localTimeElapsed + "," + \
                pmtNumDataRounded + '\n'
            csvFile.write(stringToWrite)

def update():
    global xLeftIndex
    global xRightIndex
```

```
global pmt_filtered
global pmt_thresholded
global xSamples

if (len(pmt_filtered) > 0):
    #Plot new values once we have new values to plot
    if (xLeftIndex == 0):
        ## Remove all PlotDataItems from the PlotWidgets.
        ## This resets the graphs (approx. every 30k samples)
        pmtPlotWidget.clear()
        if (filterData):
            procplot_widget.clear()

    ## pmtCurve are of the PlotDataItem type and are added to
    ## the PlotWidget. Documentation for these types can
    ## be found on pyqtgraph's website

    graph_sema.acquire()

    xRightIndex += len(pmt_filtered)

    pmtCurve = pmtPlotWidget.plot()
    xRange = range(xLeftIndex, xRightIndex)
    pmtCurve.setData(xRange, pmt_filtered)

    if (filterData):
        procplot_curve = procplot_widget.plot()
        procplot_curve.setData(xRange, pmt_thresholded)
        pmt_thresholded = []
```

```
            ## Now that we've plotted the values, we no longer need
            ## these arrays to store them
            pmt_filtered = []

            xLeftIndex = xRightIndex
            graphCount = 0
            if(xRightIndex >= xSamples):
                xRightIndex = 0
                xLeftIndex = 0

            graph_sema.release()

## Run update function in response to a timer
timer = QtCore.QTimer()
timer.timeout.connect(update)
timer.start(0)

## Start the Qt event loop
app.exec_()

#check how many packets were missed
endTime = timeElapsedPrev
packetsSent = (endTime - startTime) / usecBetweenPackets
packetsMissed = packetsSent - packetsRecieved
if packetsSent > 0:
    percentageMissed = (float(packetsMissed) / \
        float(packetsSent)) * 100.0
else:
    percentageMissed = 0
```

```
print("start: " + str(startTime) + "  endTime: " + str(endTime)\
        + "  received: " + str(packetsRecieved) + '\n')
print("You missed " + str(packetsMissed) + " out of " + \
        str(packetsSent) + " packets sent. (" + \
        str(percentageMissed) + "%)")

if saveData:
    logFile.write('Start Time: ' + str(startTime) + '\n')
    logFile.write('End Time: ' + str(endTime) + '\n')
    logFile.write('Packets Received: ' + str(packetsRecieved) \
        + '\n')
    logFile.write('Packets Sent: ' + str(packetsSent) + '\n')
    logFile.write('Packets Missed: ' + str(packetsMissed) +\
        ' (' + str(percentageMissed) + '%)' + '\n')
    logFile.close()

    # Prompts the user to add a description for the test data and
    # saves it in RecordedData\TestNotes.txt
    ## Tkinter is used to create this window
    testNotes = Tk()

    ## The label telling the user what to do
    l = Label(testNotes, text = "Describe your experiment and \
        results here.")
    l.pack(padx = 10, pady = 5, anchor = W)

    ## The textbox where the notes are written
    textBox = Text(testNotes, height = 10, width = 78)
    textBox.pack(padx = 10, pady = 5, anchor = W)
    textBox.focus_set()
```

```
## textbox for microfluidic device number
inputBoxFrame = Frame(testNotes)
inputBoxFrame.pack(padx = 10, anchor = W)
inputBoxLeftFrame = Frame(testNotes)
inputBoxLeftFrame.pack(in_ = inputBoxFrame, padx = 10, \
    side = LEFT, anchor = W)
inputBoxRightFrame = Frame(testNotes)
inputBoxRightFrame.pack(in_ = inputBoxFrame, padx = 10, \
    side = LEFT, anchor = W)
inputBoxFrameL0 = Frame(testNotes, pady = 2)
inputBoxFrameL0.pack(in_ = inputBoxLeftFrame, padx = 10, \
    anchor = W)
inputBoxFrameR0 = Frame(testNotes, pady = 2)
inputBoxFrameR0.pack(in_ = inputBoxRightFrame, padx = 10, \
    anchor = W)
inputBoxFrameL1 = Frame(testNotes, pady = 2)
inputBoxFrameL1.pack(in_ = inputBoxLeftFrame, padx = 10, \
    anchor = W)
inputBoxFrameR1 = Frame(testNotes, pady = 2)
inputBoxFrameR1.pack(in_ = inputBoxRightFrame, padx = 10, \
    anchor = W)
inputBoxFrameL2 = Frame(testNotes, pady = 2)
inputBoxFrameL2.pack(in_ = inputBoxLeftFrame, padx = 10, \
    anchor = W)
inputBoxFrameR2 = Frame(testNotes, pady = 2)
inputBoxFrameR2.pack(in_ = inputBoxRightFrame, padx = 10, \
    anchor = W)
inputBoxFrameL3 = Frame(testNotes, pady = 2)
inputBoxFrameL3.pack(in_ = inputBoxLeftFrame, padx = 10, \
```

```
    anchor = W)
inputBoxFrameR3 = Frame(testNotes, pady = 2)
inputBoxFrameR3.pack(in_ = inputBoxRightFrame, padx = 10, \
    anchor = W)
inputBoxFrameL4 = Frame(testNotes, pady = 2)
inputBoxFrameL4.pack(in_ = inputBoxLeftFrame, padx = 10, \
    anchor = W)
inputBoxFrameR4 = Frame(testNotes, pady = 2)
inputBoxFrameR4.pack(in_ = inputBoxRightFrame, padx = 10, \
    anchor = W)

deviceNumLabel = Label(testNotes, text = \
    "Microfluidic device number: ")
deviceNumEntry = Entry(testNotes)
deviceNumUsedLabel = Label(testNotes, text = \
    "Times device has been used: ")
deviceNumUsedEntry = Entry(testNotes)
laserPosLabel = Label(testNotes, text = "Laser position: ")
laserPosEntry = Entry(testNotes)
analogGainLabel = Label(testNotes, text = "Analog gain: ")
analogGainEntry = Entry(testNotes)
laserVoltLabel = Label(testNotes, text = "Laser voltage: ")
laserVoltEntry = Entry(testNotes)
pmtVoltLabel = Label(testNotes, text = \
    "PMT control voltage: ")
pmtVoltEntry = Entry(testNotes)
hvSettingsLabel = Label(testNotes, text = "HV settings: ")
hvSettingsEntry = Entry(testNotes)
buffSolLabel = Label(testNotes, text = \
    "Buffer solution used: ")
```

```
buffSolEntry = Entry(testNotes)
fluorophoreLabel = Label(testNotes, text = \
    "Fluorophore used: ")
fluorophoreEntry = Entry(testNotes)
fluorophoreConLabel = Label(testNotes, text = \
    "Fluorophore concentration: ")
fluorophoreConEntry = Entry(testNotes)

deviceNumLabel.pack(in_ = inputBoxFrameL0, side = LEFT)
deviceNumEntry.pack(in_ = inputBoxFrameL0, side = LEFT)
deviceNumUsedLabel.pack(in_ = inputBoxFrameR0, side = LEFT)
deviceNumUsedEntry.pack(in_ = inputBoxFrameR0, side = LEFT)
laserPosLabel.pack(in_ = inputBoxFrameL1, side = LEFT)
laserPosEntry.pack(in_ = inputBoxFrameL1, side = LEFT)
analogGainLabel.pack(in_ = inputBoxFrameR1, side = LEFT)
analogGainEntry.pack(in_ = inputBoxFrameR1, side = LEFT)
laserVoltLabel.pack(in_ = inputBoxFrameL2, side = LEFT)
laserVoltEntry.pack(in_ = inputBoxFrameL2, side = LEFT)
pmtVoltLabel.pack(in_ = inputBoxFrameR2, side = LEFT)
pmtVoltEntry.pack(in_ = inputBoxFrameR2, side = LEFT)
hvSettingsLabel.pack(in_ = inputBoxFrameL3, side = LEFT)
hvSettingsEntry.pack(in_ = inputBoxFrameL3, side = LEFT)
buffSolLabel.pack(in_ = inputBoxFrameR3, side = LEFT)
buffSolEntry.pack(in_ = inputBoxFrameR3, side = LEFT)
fluorophoreLabel.pack(in_ = inputBoxFrameL4, side = LEFT)
fluorophoreEntry.pack(in_ = inputBoxFrameL4, side = LEFT)
fluorophoreConLabel.pack(in_ = inputBoxFrameR4, side = LEFT)
fluorophoreConEntry.pack(in_ = inputBoxFrameR4, side = LEFT)

## Checkboxes are used to add some tags to the data
```

```
checks = Frame(testNotes)
checks.pack()

## Variables showing if the boxes are checked (1) or not (0)
broken = IntVar()
success = IntVar()
wrong = IntVar()

## Create the checkboxes
brokenCheck = Checkbutton(testNotes, \
    text="Equipment failure", variable=broken, \
    onvalue = 1, offvalue = 0)
successCheck = Checkbutton(testNotes, \
    text="Successful experiment", variable=success, \
    onvalue = 1, offvalue = 0)
wrongCheck = Checkbutton(testNotes, \
    text="I just don't know what went wrong", \
    variable=wrong, onvalue = 1, offvalue = 0)
brokenCheck.pack(in_ = checks, side = LEFT, padx = 10)
successCheck.pack(in_ = checks, side = LEFT, padx = 10)
wrongCheck.pack(in_ = checks, side = LEFT, padx = 10)

## Dictionary for keeping track of tags
with open(folderName + 'tags.json', 'r') as fp:
    checkDict = json.load(fp)

## This function runs when the submit button is pressed
def submit():
    ## save the text in the box
    text = textBox.get("1.0",'end-1c').strip()
```

```
deviceNum = deviceNumEntry.get()
#deviceNum = filter(str.isdigit, deviceNumEntry.get()).\
        #rjust(3, '0')
# Uncomment the above line if you don't trust the user to
# give a number, and change the if statement below.
# WARNING: Works in Python 2.7, but not Python 3
## If the user didn't write anything, note that
if text == "":
    text = "[No notes were included for this test.]"
## create new dictionary entry
checkDict[fileTime] = {"deviceNum": deviceNum, \
        "deviceUsed": deviceNumUsedEntry.get(), \
        "laserPos": laserPosEntry.get(), \
        "analogGain": analogGainEntry.get(),
    "laserVolt": laserVoltEntry.get(), "pmtVolt": \
        pmtVoltEntry.get(), "hvSettings": \
        hvSettingsEntry.get(), "buffSol": \
        buffSolEntry.get(),
    "fluorophore": fluorophoreEntry.get(), \
        "fluorophoreCon": fluorophoreConEntry.get(), \
        "success": success.get(), "broken": \
        broken.get(), "wrong": wrong.get(), "text": text}
## add other data to the front of the text for printing
if deviceNum != "":
        # If the line above was uncommented, change
        # this to 'if deviceNum != "000":'
    text = "[Microfluidic device #" + deviceNum + "]\n" \
        + text
if wrong.get() == 1:
    text = "[I just don't know what went wrong]\n" + text
```

```
    if broken.get() == 1:
        text = "[Equipment failure]\n" + text
    if success.get() == 1:
        text = "[Successful experiment]\n" + text
    ## Write to the file
    fNote = open(folderName + 'TestNotes.txt','a')
    fNote.write("\n\n***" + fileTime + "***\n" + text + "\n")
    fNote.close()
    ## Write tags to .json file
    with open(folderName + 'tags.json', 'w') as fp:
        json.dump(checkDict, fp)
    ## Close the window
    testNotes.destroy()

## The submit button
submitButton = Button(testNotes, text = "Submit", \
    width = 10, command = submit)
submitButton.pack(pady = 10)

## This actually runs the prompt
testNotes.mainloop()
```

## APPENDIX C. ARDUINO UNO CODE

```
/*
When we run this program we will read in voltage values
from pins A0(PMT) from a Teensy and write these values,
along with the time in microseconds from the start of
the program, to the serial port for retrieval by our
python program, Teensy Data.
*/

// new board
int relayPin1 = 8;  // HV Separation Well
int relayPin2 = 7;  // GND Separation Well
int relayPin3 = 6;  // HV Source Well
int relayPin4 = 5;  // GND Source Well
int relayPin5 = 4;  // HV Waste Well
int relayPin6 = 3;  // GND Waste Well


// Variables
int incomingByte=0;
int pmt = 0;
unsigned long time;
IntervalTimer timer0;

// Function we will run in response to a 100 us timer
```

```
void SampleVoltage() {
        analogReadResolution(12); // Max 12 bit
        pmt = analogRead(0); // Voltage read from Teensy pin A0
        time = micros(); // Microseconds since start of program

        // We will write our values as an array of bytes for easy
        // transmission and retrieval
        unsigned char serialBytes[8];
        serialBytes[0] = (time >> 24) & 0xff;
        serialBytes[1] = (time >> 16) & 0xff;
        serialBytes[2] = (time >> 8) & 0xff;
        serialBytes[3] = time & 0xff;
        serialBytes[4] = (pmt >> 8) & 0xff;
        serialBytes[5] = pmt & 0xff;

        // These two bytes are here because the serial
        // communication only works when 8 bytes are sent.
        serialBytes[6] = 0x00;
        serialBytes[7] = 0x00;

        Serial.write(serialBytes,8);
}

void timer_setup() { // run SampleVoltage every 1ms
        timer0.begin(SampleVoltage, 1000);
}

void timer_stop() {
        timer0.end();
}
```

```
void setup() {
        // Set the pins as outputs
        pinMode(relayPin1, OUTPUT);
        pinMode(relayPin2, OUTPUT);
        pinMode(relayPin3, OUTPUT);
        pinMode(relayPin4, OUTPUT);
        pinMode(relayPin5, OUTPUT);
        pinMode(relayPin6, OUTPUT);

        digitalWrite(relayPin1, LOW);
        digitalWrite(relayPin2, LOW);
        digitalWrite(relayPin3, LOW);
        digitalWrite(relayPin4, LOW);
        digitalWrite(relayPin5, LOW);
        digitalWrite(relayPin6, LOW);

        Serial.begin(9600);
        delay(2000); //To allow time for serial port to begin
        timer_setup(); //Run timer
}

void loop() {
        //-------------------------------------------------------
        //-------------------FOR HIGH VOLTAGE--------------------
        //-------------------------------------------------------
        while(Serial.available()) {

                incomingByte = Serial.read();
```

```
// Change LOW/HIGH to route the correct voltage
// Change the pins (1/2, 3/4, 5/6) for each well
switch (incomingByte) { // Case format is DEC
case 49:
// HV on
digitalWrite(relayPin1, LOW);
digitalWrite(relayPin2, LOW);
digitalWrite(relayPin3, LOW);
digitalWrite(relayPin4, LOW);
digitalWrite(relayPin5, LOW);
digitalWrite(relayPin6, LOW);
break;
case 50:// Number 2 in 8-bit decimal
// Separation
digitalWrite(relayPin1, LOW);
digitalWrite(relayPin2, LOW);
digitalWrite(relayPin3, LOW);
digitalWrite(relayPin4, LOW);
digitalWrite(relayPin5, LOW);
digitalWrite(relayPin6, LOW);

digitalWrite(relayPin1, HIGH);
digitalWrite(relayPin3, HIGH);
digitalWrite(relayPin5, HIGH);
break;
case 51:
// Injection
digitalWrite(relayPin1, LOW);
digitalWrite(relayPin2, LOW);
digitalWrite(relayPin3, LOW);
```

```
                digitalWrite(relayPin4, LOW);
                digitalWrite(relayPin5, LOW);
                digitalWrite(relayPin6, LOW);

                digitalWrite(relayPin2, HIGH);
                digitalWrite(relayPin4, HIGH);
                digitalWrite(relayPin5, HIGH);
                break;
                default:
                // HV off
                digitalWrite(relayPin1, LOW);
                digitalWrite(relayPin2, LOW);
                digitalWrite(relayPin3, LOW);
                digitalWrite(relayPin4, LOW);
                digitalWrite(relayPin5, LOW);
                digitalWrite(relayPin6, LOW);
                break;
        }
}
}
```

## APPENDIX D. PYTHON KEYLOGGER INTERFACE CODE

```
import Tkinter as tk
from Tkinter import *
import sys
import glob
import serial
import time

#Added to check the GIT weirdness

def serial_ports():
        if sys.platform.startswith('win'):
                ports = ['COM' + str(i + 1) for i in range(256)]
        elif sys.platform.startswith('linux') or \
                sys.platform.startswith('cygwin'):
                ports = glob.glob('/dev/tty[A-Za-z]*')
        elif sys.platform.startswith('darwin'):
                ports = glob.glob('/dev/tty.*')
        else:
                raise EnvironmentError('Unsupported platform')

        result = []
        for port in ports:
                try:
                        s = serial.Serial(port)
```

```
                        s.close()
                        result.append(port)
                except (OSError, serial.SerialException):
                        pass
        return result

current_ports = serial_ports()
#current_ports = [1,2,3,4,5]
ser = serial.Serial(current_ports[0], 9600, timeout=1)
print 'Running on serial port (to be configured)'
distance_text = 0
global x_pos
global y_pos
global z_pos
global gain
global voltage

def setTextBox(start, end, text):
        textBox.delete(start, end)
        textBox.insert(start, text)

def key(event):
        if event.keysym == 'Escape':
                writeFile()
                root.destroy()
        if event.char == event.keysym:
                global distance_text
                global command_count
                global x_pos
                global y_pos
```

```
global z_pos
global gain
global voltage
message = "0"
#This is a normal number or letter char
#insert what you want it to do here
print( 'Normal Key %r' % event.char )
if event.char == '4':
        message = "1500"
        distance_text = 192
        print( 'Distance set to ' + message)
        setTextBox("1.0", "1.50", \
        "Distance changed to " + message)
elif event.char == '3':
        message = "125"
        print( 'Distance set to ' + message )
        distance_text = 16
        setTextBox("1.0", "1.50", \
        "Distance changed to " + message)
elif event.char == '2':
        message = "46.875"
        print( 'Distance set to ' + message )
        distance_text = 6
        setTextBox("1.0", "1.50", \
        "Distance changed to " + message)
elif event.char == '1':
        message = "7.8"
        print( 'Distance set to ' + message )
        distance_text = 1
        setTextBox("1.0", "1.50", \
```

```
            "Distance changed to " + message)
    elif event.char == 'w':
            print( 'Moving up' )
            send_movement_command(1,'y','+',\
            distance_text)
            y_pos += distance_text
    elif event.char == 'a':
            print( 'Moving left' )
            send_movement_command(1,'x','-',\
            distance_text)
            x_pos -= distance_text
    elif event.char == 's':
            print( 'Moving down' )
            send_movement_command(1,'y','-',\
            distance_text)
            y_pos -= distance_text
    elif event.char == 'd':
            print( 'Moving right' )
            send_movement_command(1,'x','+',\
            distance_text)
            x_pos += distance_text
            #print('Command count ' + \
            # str(command_count))
    elif event.char == 'r':
            print( 'Moving zup' )
            send_movement_command(1,'z','+',\
            distance_text)
            z_pos += distance_text
    elif event.char == 'f':
            print( 'Moving zdown' )
```

```
            send_movement_command(1,'z','-',\
            distance_text)
            z_pos -= distance_text
        elif event.char == 'y':
            print( 'Gain up' )
            send_potentiometer_command(1,'g','+')
            z_pos += distance_text
        elif event.char == 'h':
            print( 'Gain down' )
            send_potentiometer_command(1,'g','-')
            z_pos -= distance_text
        elif event.char == 'i':
            print( 'PMT Voltage up' )
            send_potentiometer_command(1,'p','+')
            z_pos += distance_text
        elif event.char == 'k':
            print( 'PMT Voltage down' )
            send_potentiometer_command(1,'p','-')
            z_pos -= distance_text
        elif event.char == 'o':
            setZero()
        elif event.char == 't':
            for x in range(200):
                    send_movement_command(1,'x','+',1)
        elif event.char == 'p':
            setTextBox("1.0", "1.100", \
            "Current position: " + str(x_pos) + ":" \
            + str(y_pos) + ":" + str(z_pos) + \
            " PMT:" + str(1.2*voltage/256) + \
            "Gain" + str(10*gain/256))
```

```
        else:
            print('Doing nothing...')

def change_port(value):
        ser.port = value
        print "Serial port changed to " + ser.port
#       current_ports[0] = value
#       print "Port changed to " + str(value)
        portBox.delete("1.0", END)
        portBox.insert(INSERT, "Serial port changed to " + \
                ser.port)

position = []

def readFile():
        print("Reading file...")
        file = open('position.txt', 'r')
        f = file.readlines()
        global position
        global x_pos
        global y_pos
        global z_pos
        global gain
        global voltage
        pos = f[0].split(':')
        x_pos = int(pos[0])
        y_pos = int(pos[1])
        z_pos = int(pos[2])
        gain  = int(pos[3])
        voltage = int(pos[4])
```

```
        position = f[1:]
        print("Done.")

def writeFile():
        global x_pos
        global y_pos
        global z_pos
        global gain
        global voltage
        global position
        print("Writing to file...")
        file = open('position.txt','w')
        file.writelines(str(x_pos) + ":" + str(y_pos) + ":" \
                + str(z_pos) + "\n")
        file.writelines(position)

def send_movement_command(XYZ_set, axis, direction, distance):
        to_write = str(XYZ_set)+':'+str(axis)+':'+str(direction)\
                +':'+str(distance)+"\n"
        ser.write(to_write)
        print "Sent: " + to_write
        setTextBox("1.0", "1.100", "Sent: " + to_write)

def send_potentiometer_command(XYZ_set, axis, direction):
        to_write = str(XYZ_set)+':'+str(axis)+':'+str(direction)\
                +"\n"
        ser.write(to_write)
        print "Sent: " + to_write
        setTextBox("1.0", "1.100", "Sent: " + to_write)
```

```
def send_position(command):
        global position
        com = command.split(':')
        t = 0
        pos_vector = []
        for x in position:
                n = x.split(':')
                print(n[0] + " " + com[0])
                if (n[0] == com[0]):
                        pos_vector = x
                t += 1
        global x_pos
        global y_pos
        global z_pos
        pos = pos_vector.split(':')
        x_move = int(pos[1])
        y_move = int(pos[2])
        z_move = int(pos[3])
        x_final = abs(x_pos - x_move)
        y_final = abs(y_pos - y_move)
        z_final = abs(z_pos - z_move)
        if x_move > x_pos:
                setDir = '+'
        else:
                setDir = '-'
        send_movement_command(1,'x',setDir,x_final)
        if y_move > y_pos:
                setDir = '+'
        else:
                setDir = '-'
```

```
        send_movement_command(1,'y',setDir,y_final)
        if z_move > z_pos:
                setDir = '+'
        else:
                setDir = '-'
        send_movement_command(1,'z',setDir,z_final)
        print "Sent position: " + command
        x_pos = x_move
        y_pos = y_move
        z_pos = z_move

def setZero():
        global x_pos
        global y_pos
        global z_pos
        x_pos = 0
        y_pos = 0
        z_pos = 0
        print("Set zero")

def setPosition(name):
        global x_pos
        global y_pos
        global z_pos
        global position
        q = name.split(':')
        setString = q[0]
        t = 0
        for x in position:
                n = x.split(':')
```

```
                print(n[0] + " " + q[0])
                if (n[0] == q[0]):
                        setString += ":" + str(x_pos) + ":" + \
                        str(y_pos) + ":" + str(z_pos)
                        position[t] = setString
                t += 1
        #updateMenu()
        print("Position set." + str(position))



def updateMenu():
        t = 0
        for x in position:
                name = x.split(':')
                set_position_menu.entryconfig(t, label=\
                        str(name[0]), command=lambda x=x: \
                        setPosition(x))
                print(name)
                t += 1


root = tk.Tk()
readFile()
menubar = Menu(root)
portmenu = Menu(menubar, tearoff=0)
positionmenu = Menu(menubar, tearoff=0)
set_position_menu = Menu(menubar, tearoff=0)
textBox = Text(root)
portBox = Text(root)
textBox.insert(INSERT, "Distance changed to " + \
```

```
        str(distance_text))
portBox.insert(INSERT, "Port is set to " + str(current_ports[0]))
textBox.pack()
portBox.pack()

for x in current_ports:
        portmenu.add_command(label=str(x), command=lambda x=x:\
                change_port(x))

for x in position:
        name = x.split(':')
        positionmenu.add_command(label=str(name[0]), \
                command=lambda x=x: send_position(x))

for x in position:
        name = x.split(':')
        set_position_menu.add_command(label=str(name[0]), \
                command=lambda x=x: setPosition(x))
        print(name[0])

menubar.add_cascade(label="Ports...", menu=portmenu)
menubar.add_cascade(label="Positions...", menu=positionmenu)
menubar.add_cascade(label="Set Position", menu=set_position_menu)
#x_pos = 0
#y_pos = 0
#z_pos = 0
print( "Press a key. Let's see if this works...")
root.bind_all('<Key>', key)
#this hides the GUI window. We can replace this soon
#root.withdraw()
```

```
root.config(menu=menubar)
root.mainloop()
Appendix E      Matlab Post Processing Script
fc = 10;
Fs = 1000;
Wn = (2/Fs)*fc;
b = fir1(1000,Wn,'low',kaiser(1001,3));

%fvtool(b,1,'Fs',Fs);
%d = decimate(PMT2,8)
y = filter(b,1,PMT7);
figure();
x = [0.001:0.001:(length(y)/1000)]';
plot(x,y);
title('1uM Phenylalanine and Glycine');
ylabel('12-bit ADC Reading');
%ylim([600 inf]);
xlabel('Time (sec)');

begining = input('Starting index of the noise in seconds');
ending = input('Ending index of the noise in seconds');
noise = y(begining*1000:ending*1000);
ave = mean(noise);
sigma = std(noise);
[psor,lsor] = findpeaks(y,1000,'SortStr','descend',\
        'MinPeakDistance',0.5);

index = 2;
text(lsor(1:index),psor(1:index),\
        num2str((1:numel(psor(1:index)))'))
```

```
Peak1 = (y(Isor(1)*1000)-ave);
Peak2 = (y(Isor(2)*1000)-ave);
```

## APPENDIX E. MATLAB SCRIPT

```
fc = 10;
Fs = 1000;
Wn = (2/Fs)*fc;
b = fir1(1000,Wn,'low',kaiser(1001,3));

%fvtool(b,1,'Fs',Fs);
%d = decimate(PMT2,8)
y = filter(b,1,PMT6);
figure();
x = [0.001:0.001:(length(y)/1000)]';
plot(x,y);
title('1uM Phenylalanine and Glycine');
ylabel('12-bit ADC Reading');
%ylim([600 inf]);
xlabel('Time (sec)');

begining = input('Starting index of the noise in seconds');
ending = input('Ending index of the noise in seconds');
noise = y(begining*1000:ending*1000);
ave = mean(noise);
sigma = std(noise);
[psor,lsor] = findpeaks(y,1000,'SortStr','descend',
        'MinPeakDistance',0.5);
```

```
index = 2;
text(lsor(1:index),psor(1:index),
        num2str((1:numel(psor(1:index)))'));
Peak1 = (y(lsor(1)*1000)-ave);
Peak2 = (y(lsor(2)*1000)-ave);
```

# APPENDIX F. EMCO DATASHEET

Proven Reliability

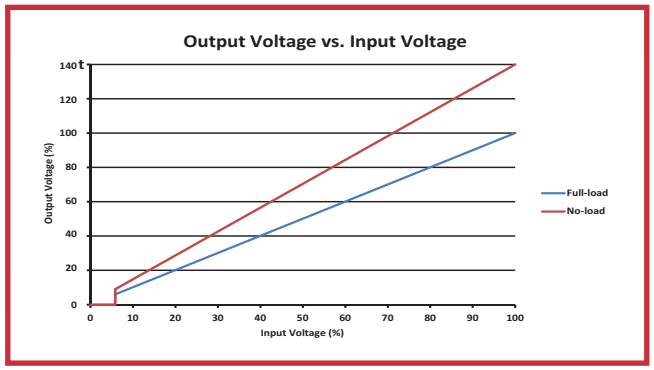
# F SERIES

## ISOLATED, PROPORTIONAL DC TO HV DC CONVERTERS

**200V to 8kV at 10W**

## PRODUCT DESCRIPTION

The F Series is a broad line of robust, field-proven DC to HV DC converters which deliver 10 Watts continuous output power. Outputs range from 200V to 8kV. The input and output are galvanically isolated so a unit can be used to deliver a positive or negative high-voltage. The output voltage is proportional to input voltage with a low 0.7V typical turn-on voltage. The F Series employs EMCO's quasi-sinewave oscillator, a fully enclosed transformer, input and output filtering, and a 5-sided metal enclosure. As a result, these modules exhibit very low EMI/RFI, noise and ripple. A dual output option with center-tap pin which, when grounded provides both positive and negative outputs from one compact module.




## APPLICATIONS

- Electrophoresis
- Capacitor Charging
- Piezo Devices
- Field Generation
- Grid Bias
- Mass Spectrometry
- Electrostatic Chuck
- Igniter / Spark Module
- Sustaining Ion Pumps

## OPTIONS

- Dual output (Center Tap) Models Available (CT Suffix)
- Mounting Holes (F02 thru F60 only)(H Suffix)

## PRODUCT SELECTION TABLE

| MODEL | OUTPUT VOLTAGE*2 | OUTPUT CURRENT*1 |
|---|---|---|
| F02 | 0 to 200V | 50 mA |
| F03 | 0 to 300V | 33.3 mA |
| F04 | 0 to 400V | 25 mA |
| F05 | 0 to 500V | 20 mA |
| F06 | 0 to 600V | 16 mA |
| F08 | 0 to 800V | 12.5 mA |
| F10 | 0 to 1kV | 10 mA |
| F12 | 0 to 1.2kV | 8.3 mA |
| F15 | 0 to 1.5kV | 6.6 mA |
| F20 | 0 to 2kV | 5 mA |
| F30 | 0 to 3kV | 3.3 mA |
| F40 | 0 to 4kV | 2.5 mA |
| F50 | 0 to 5kV | 2 mA |
| F60 | 0 to 6kV | 1.66 mA |
| F70 | 0 to 7kV | 1.5 mA |
| F80 | 0 to 8kV | 1.25 mA |

## FEATURES

- Proportional Input/Output
- Compact, PCB Mount Package
- Metal Case / Shielded Transformer
- Short Circuit Protection
- Low Ripple, Low EMI / RFI
- Proven Reliability
- Input/Output Isolation
- Low Leakage Current
- Low Input/Output Coupling Capacitance
- No External Components Required
- No Minimum Load Required
- MTBF: >810K hrs per Bellcore TR-332
- RoHS Compliant

ISO 9001:2008 CERTIFIED ISO 14001:2004 CERTIFIED RoHS COMPLIANT IPC Certified J-STD-001 Application Specialist

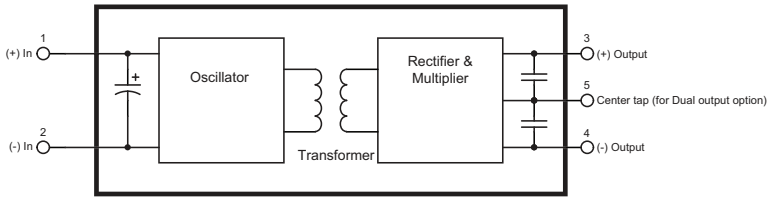
## ELECTRICAL SPECIFICATIONS*2

| MODELS | INPUT VOLTAGE | OUTPUT CURRENT | RIPPLE*4 | REVERSIBLE MODEL | OUTPUT VOLTAGE | DUAL OUTPUT (CENTER TAP) MODEL | OUTPUT VOLTAGE |
|---|---|---|---|---|---|---|---|
| F02/F02CT | 0 to 12V | 50mA | <1.0% | F02 | 0 to 200V | F02CT | 0 to +/- 100V |
| F03/F03CT | 0 to 12V | 33.3mA | <1.0% | F03 | 0 to 300V | F03CT | 0 to +/- 150V |
| F04/F04CT | 0 to 12V | 25mA | <1.0% | F04 | 0 to 400V | F04CT | 0 to +/- 200V |
| F05/F05CT | 0 to 12V | 20mA | <0.1% | F05 | 0 to 500V | F05CT | 0 to +/- 250V |
| F06/F06CT | 0 to 12V | 16mA | <0.1% | F06 | 0 to 600V | F06CT | 0 to +/- 300V |
| F08/F08CT | 0 to 12V | 12.5mA | <0.1% | F08 | 0 to 800V | F08CT | 0 to +/- 400V |
| F10/F10CT | 0 to 12V | 10mA | <0.1% | F10 | 0 to 1kV | F10CT | 0 to +/- 500V |
| F12/F12CT | 0 to 12V | 8.3mA | <0.1% | F12 | 0 to 1.2kV | F12CT | 0 to +/- 600V |
| F15/F15CT | 0 to 12V | 6.6mA | <0.1% | F15 | 0 to 1.5kV | F15CT | 0 to +/- 750V |
| F20/F20CT | 0 to 12V | 5mA | <1.0% | F20 | 0 to 2kV | F20CT | 0 to +/- 1kV |
| F30/F30CT | 0 to 15V | 3.3mA | <1.0% | F30 | 0 to 3kV | F30CT | 0 to +/- 1.5kV |
| F40/F40CT | 0 to 15V | 2.5mA | <1.0% | F40 | 0 to 4kV | F40CT | 0 to +/- 2kV |
| F50/F50CT | 0 to 15V | 2mA | <1.0% | F50 | 0 to 5kV | F50CT | 0 to +/- 2.5kV |
| F60/F60CT | 0 to 15V | 1.66mA | <1.0% | F60 | 0 to 6kV | F60CT | 0 to +/- 3kV |
| F70/F70CT | 0 to 15V | 1.5mA | <2.5% | F70 | 0 to 7kV | F70CT | 0 to +/- 3.5kV |
| F80/F80CT | 0 to 15V | 1.25mA | <2.5% | F80 | 0 to 8kV | F80CT | 0 to +/- 4kV |

## F-SERIES ELECTRICAL SPECIFICATIONS*3

| PARAMETER | VALUE |
|---|---|
| INPUT VOLTAGE | 0 TO 12V (F02 TO F20) |
| | 0 TO 15V (F30 TO F80) |
| TURN-ON VOLTAGE | <0.7V |
| INPUT CURRENT | <500MA, NO LOAD |
| | <1.5A, FULL LOAD |
| ISOLATION | < +/-3.5kV BIAS (F02 TO F60) |
| | < +/-500V BIAS (F70 TO F80) |
| INPUT CAPACITANCE | ~240uF |
| RESPONSE TIME | 260 ms (typical) |
| OUTPUT VOLTAGE TOLERANCE | +/-5% (Full Load, 100% output voltage) |
| FREQUENCY | 25 kHz TO 125 kHz |
| OPERATING TEMPERATURE*5 | -10C to +70C (Case) |
| STORAGE TEMPERATURE | -25C to +90C |

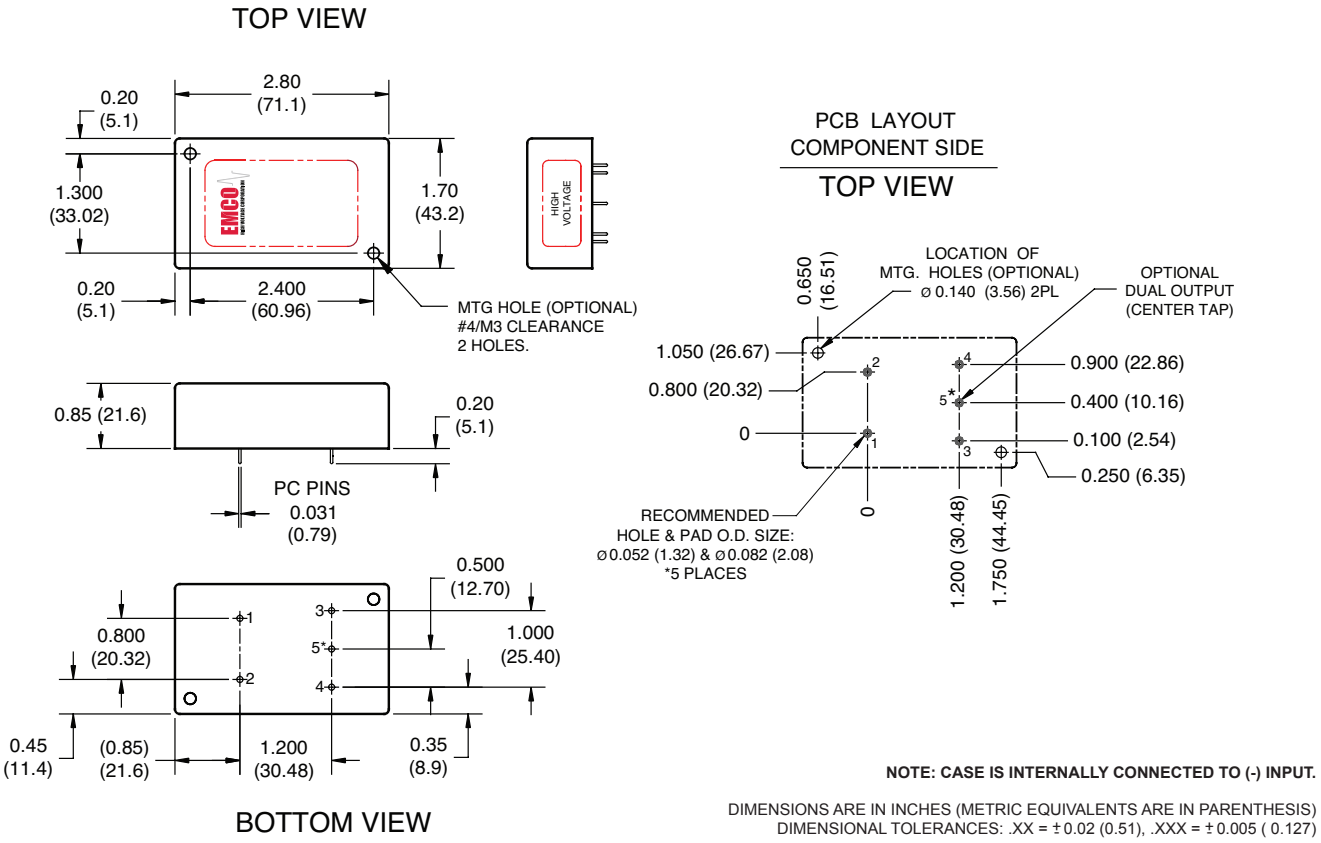
## BLOCK DIAGRAM




## HOT SPOT CASE TEMPERATURE POINT


DIMENSIONS ARE IN INCHES (METRIC EQUIVALENTS ARE IN PARENTHESIS)
DIMENSIONAL TOLERANCES: .XX = ± 0.03 (0.76), .XXX = ± 0.005 ( 0.127)

## MECHANICAL SPECIFICATIONS F02-F60



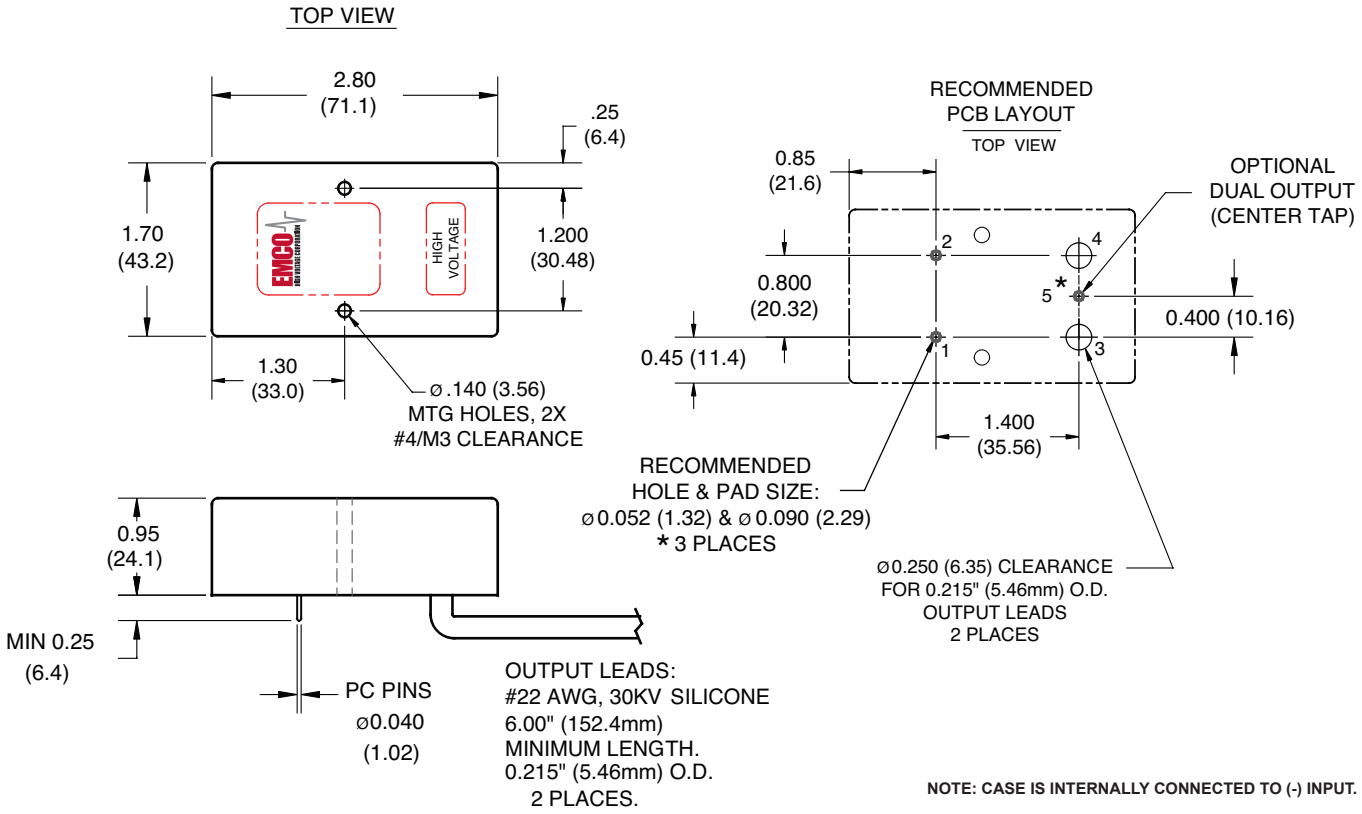

**NOTE: CASE IS INTERNALLY CONNECTED TO (-) INPUT.**

DIMENSIONS ARE IN INCHES (METRIC EQUIVALENTS ARE IN PARENTHESIS)
DIMENSIONAL TOLERANCES: .XX = ± 0.02 (0.51), .XXX = ± 0.005 ( 0.127)

| PARAMETER | VALUE |
|---|---|
| WEIGHT | <5 OZ (142 GRAMS) |
| VOLUME | 4.0 cu. In. (66.3 cc) |
| DIMENSIONS | 2.80L (71.1L) x 1.70W (43.2W) x 0.85H (21.6H) |
| CASE MATERIAL | Black Anodized Aluminum |

| PIN # | FUNCTION |
|---|---|
| 1 | (+) Input |
| 2 | (-) Input |
| 3 | (+) Output |
| 4 | (-) Output |
| 5 | Dual Output/Center Tap (optional) |

## MECHANICAL SPECIFICATIONS F70-F80




**NOTE: CASE IS INTERNALLY CONNECTED TO (-) INPUT.**

DIMENSIONS ARE IN INCHES (METRIC EQUIVALENTS ARE IN PARENTHESIS)
DIMENSIONAL TOLERANCES: .XX = ± 0.02 (0.51), .XXX = ± 0.005 ( 0.127)

| PARAMETER | VALUE |
|---|---|
| WEIGHT | <5 OZ (142 GRAMS) |
| VOLUME | 4.5 cu. In. (74.1cc) |
| DIMENSIONS | 2.80L (71.12L) x 1.70W (43.18W) x 0.95H (24.13H) |
| CASE MATERIAL | Black Anodized Aluminum |

| PIN # | FUNCTION |
|---|---|
| 1 | (+) Input |
| 2 | (-) Input |
| 3 | (+) Output |
| 4 | (-) Output |
| 5 | Dual Output/Center Tap (optional) |

## OPTION CODE TABLE

| OPTION | | ORDER CODE | MODELS |
|---|---|---|---|
| POLARITY | Positive / Negative reversible | Blank | ALL |
| | Dual Output (center tap) option | CT | ALL |
| OPTIONS | Mounting Holes | H | UP TO 6KV |

## HOW TO ORDER

### PART NUMBER SELECTOR:

Model Number:

**F30CTH**

Model

Output Voltage (See table)

Mounting Holes (optional)

Center Tap (for dual output option)

EXAMPLE: **F30CTH** (**F**-Model, **30**-Output Voltage, **CT**-Center Tap, **H**-Holes)

* Notes:

1. At maximum rated output voltage.
2. Output voltage is load dependent. Under light or no load conditions, reduce input voltage so maximum rated output voltage is not exceeded.
3. Specifications after 1 hour warm up, full load, at 25C unless otherwise indicated.
4. Ripple specification for dual output units applies to the voltage between the positive and negative output terminals.
5. Proper thermal management techniques are required to maintain safe case temperature at maximum power output