Electrical Engineering 129C
Senior Design Project

University of California, Santa Cruz
Spring 2018

Senior Design Final Report

Jose Fuentes
Melvin Abzun
Joseph Adamson

June 14, 2018

Abstract: With this project we aim to deliver a light weight, cost-effective, and portable diagnostic system for use at a point-of-care setting. The spectrometer-based system provides a means of pathogen detection by spectral analysis of a sample. The system in intended to be simple to use, in order that medical or research personnel can utilize it without worrying about size or complicated interfaces. Existing equipment utilizing this method is cumbersome and highly expensive; we are attempting to reduce these factors by delivering a device approximately the size of a textbook, which can be manufactured for a much lower cost than the full-sized equipment and that can deliver accurate results.

Part 1: Introduction

Our project functions on the principle that light emitted from an illuminated biological sample can provide information regarding the contents of the sample, including such pathogens as viruses or cancer. By utilizing an optical system, spectrometer, and Raspberry Pi, the system is able to detect these pathogens with a device no larger than a textbook.

During winter quarter, we focused on developing and testing each individual portion of the system. Working prototypes of the software, optical system, and mechanical design were developed and tested in the lab, and found to be functioning according to our requirements. Our progress was aided and monitored by our client, who has found our progress to be satisfactory.

During spring quarter, we have gone above and beyond our goals; in addition to producing a fully integrated system, we have also improved upon our initial enclosure designs, further minimized the light source, spectrometer, constructed our own PCB power distribution/daughterboard, constructed a custom microfluidic pump, and added various other polish.

As the final quarter of this project draws to a close, we are happy to document that we have achieved our goals regarding form factor, cost constraint, and methods of control. This report will document each portion of the system integration, beginning with the optical and mechanical designs, moving on to the software design, and finishing with implementation details regarding the full system fabrication/integration.

Part 2: Design

I. Design Summary

The design work for this project breaks down into three major areas, consisting of Optical, Mechanical, and Software design.

The focus of our work during this quarter was system integration; we began with each subsystem functioning separately and proceeded to physically and electrically incorporate them together. The final implementation and design details are described in detail below.

II. System Integration Diagram

The diagram in figure 1 below demonstrates each portion of the final implementation of our project, as well as the nature of the connections between them. Specific details for each subsystem as they relate to overall functionality are provided later in the report.

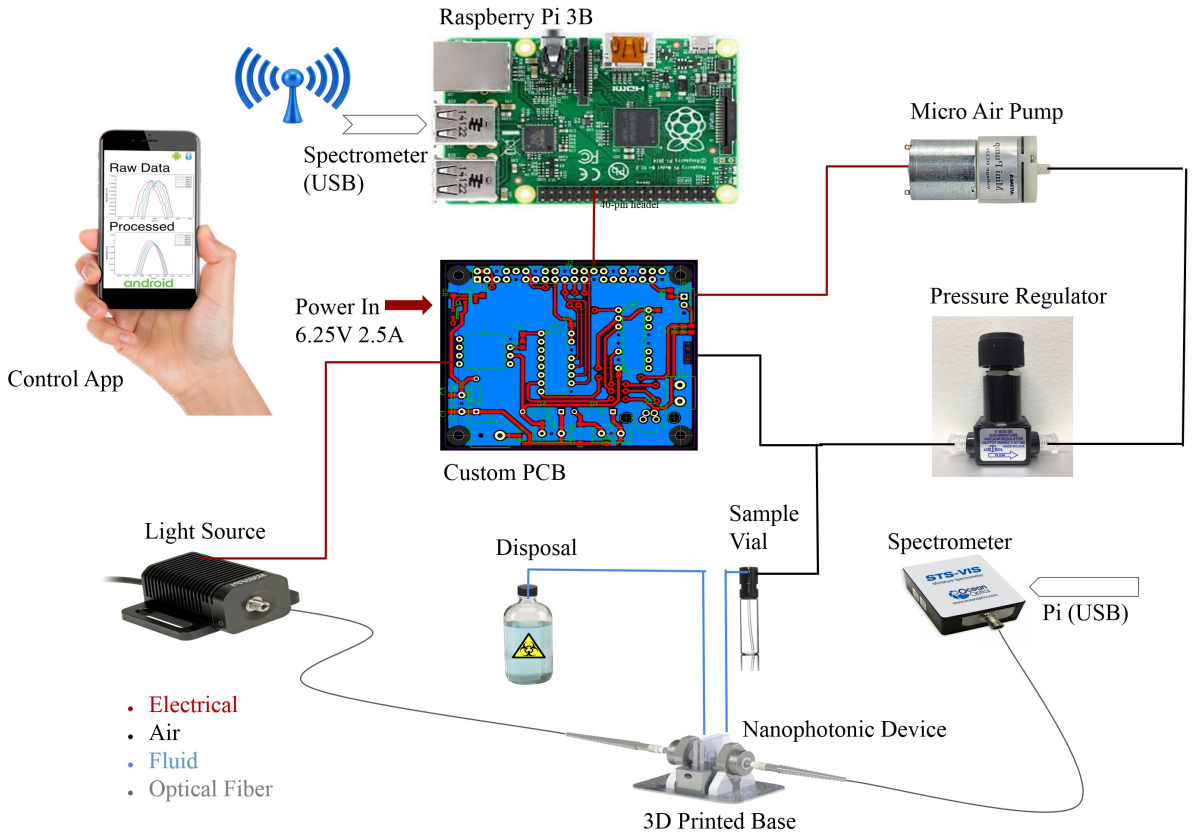


Figure 1: System Integration Diagram

III. Optical System Design

III.A Current Optical System Design Owned by Client

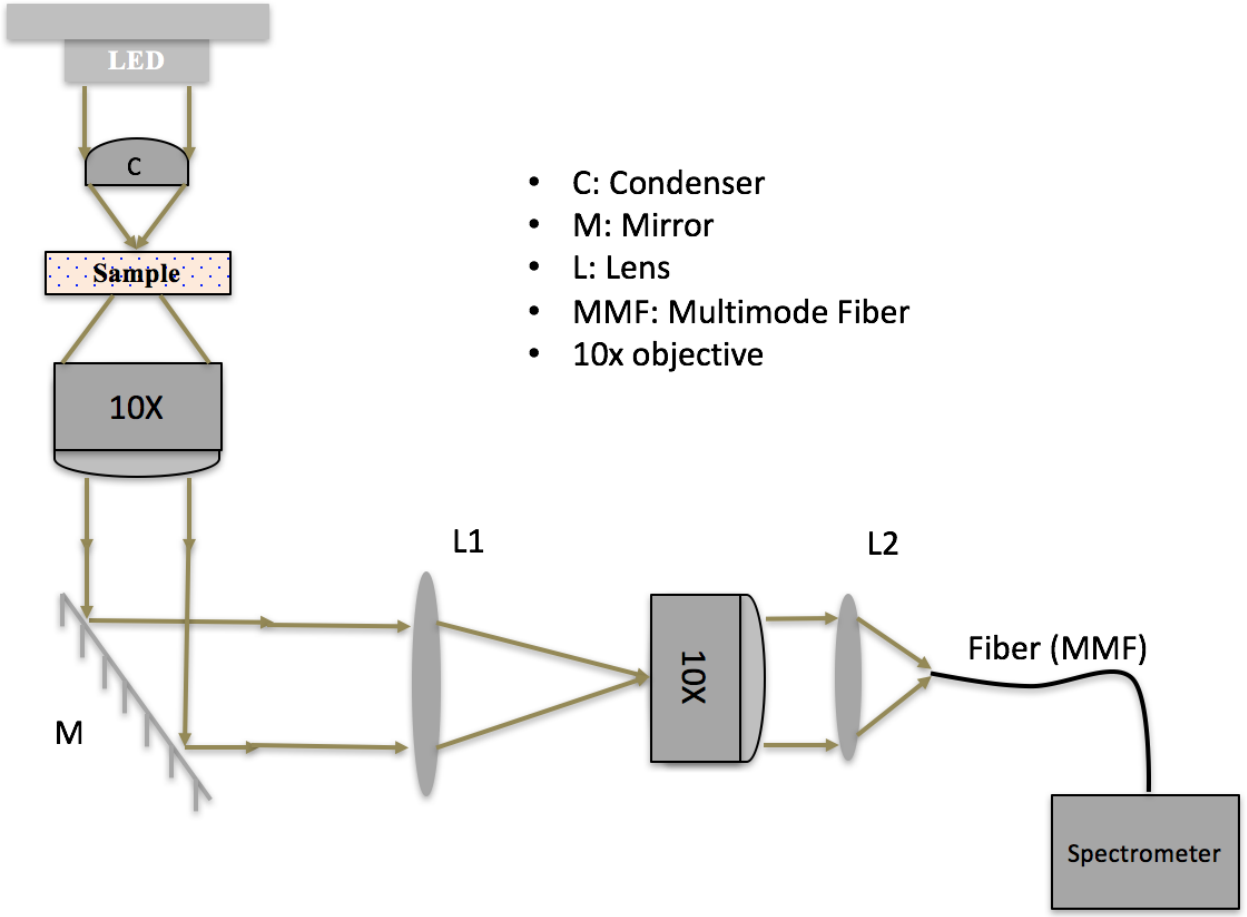


Figure 2: Client's Bulky Optical System Design

A bulky optical system diagram shown in figure 2 above was developed by the optical engineer from typical systems used for Spectral Measurements by the client. An example of the setup can be seen in figure 5. By thoroughly understanding the optical system, the optical engineer created many prototypes over the 2017 winter break.

A final miniaturized optical system diagram, shown in in figure 3, was developed and found to be working correctly with data collected with a spectrometer owned by our client. Our project was very dependent on this optical system in order to proceed at the beginning of this quarter. The completion of the optical system let the mechanical and software engineers start brainstorming their ideas and deliverables for the rest of the academic year to complete the project.

III.B Our Current Simplified Optical System Design

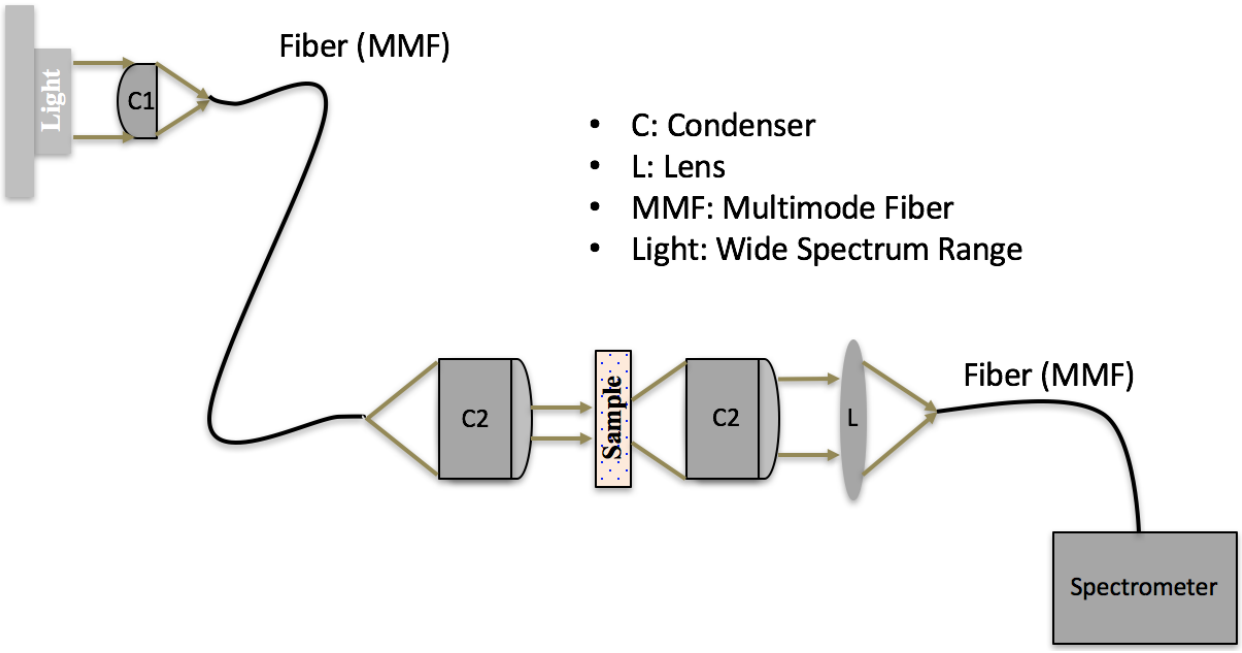


Figure 3: Our Optical System Design

A close collaboration between the mechanical and optical engineers was established to implement our working simplified optical system manufacturing and fabrication. The collaboration resulted in the fabrication of our own mechanical parts as discussed under the fabrication section below, and as seen in figure 4. Emphasis was placed on the middle optical parts between fibers. A comparison between our optical system design and the implementation currently owned by the client is shown in the following section.

Figure 4: Initial Miniaturized Optical System 3D Design

IV. Optical Design Implementation

Figure 5: Typical Systems used for Spectra Measurements

In figure 5 we can see a typical microscope similar to the one that our client uses now. The client's main focus was a proof of concept, with size being initially irrelevant. Now that the client has a working system, size is a priority for portability purposes. After we simplified the bulky system shown in figure 5, we went ahead and completed a working system without pump which may be observed in figures 6 and 7. The pump was considered a stretch goal for spring quarter. A fully working system at this point was the main priority in order to move on with stretch goals and further alternative size reduction components.

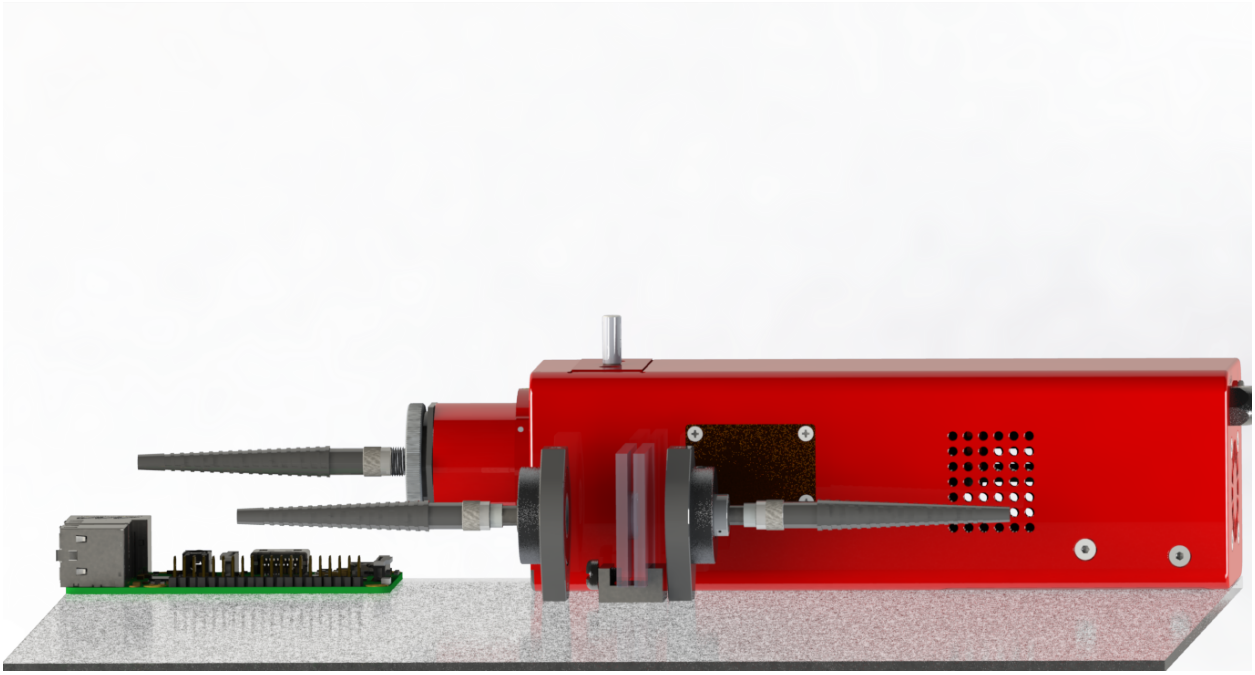


Figure 6: Side View Placement of Initial Prototype Parts

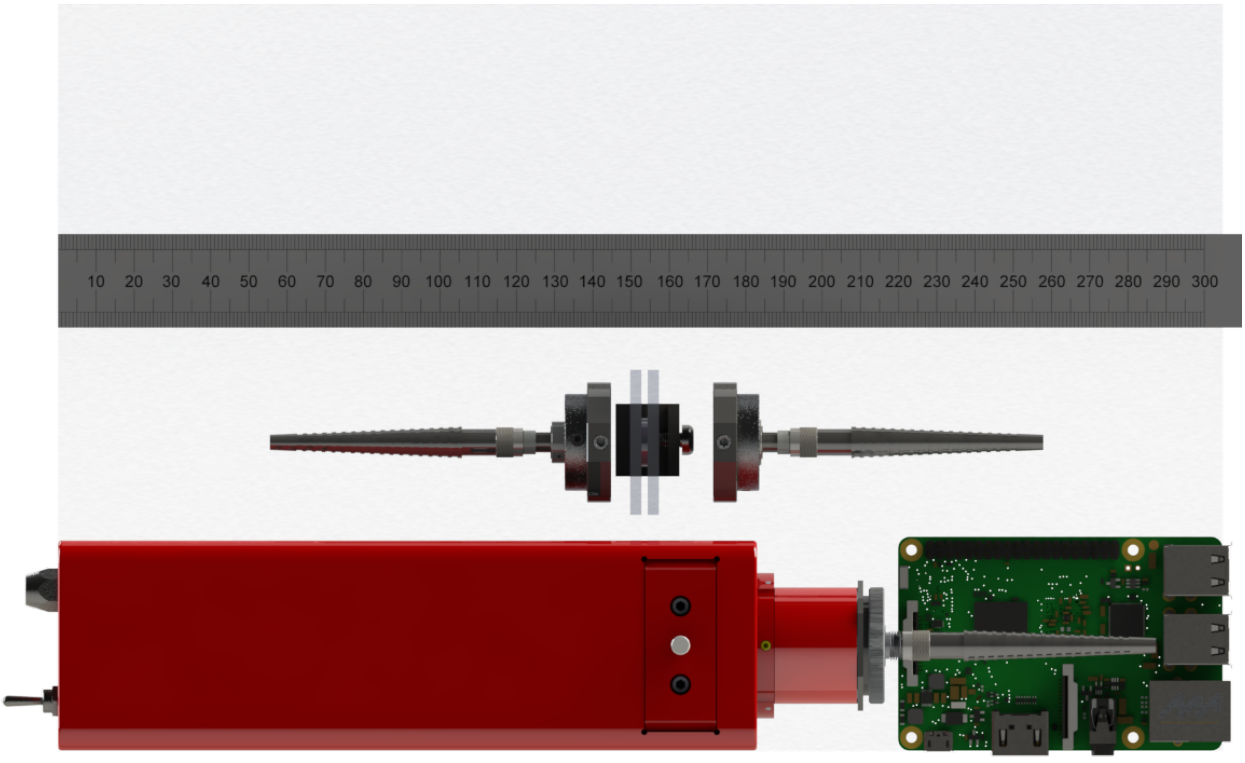


Figure 7: Top View Placement of Initial Prototype Parts

V. Size Reduction of Mechanical/Optical Design Implementation

The working system was initially tested with an external syringe pump and spectrometer owned by the client. The form factor constraints desired by the client were still not met, which led to finding alternative methods of size reduction at this stage of our project. The main bulky components of concern were the SLS201L broadband light source (the red block observed earlier in figures 6 and 7) and the larger spectrometer owned by client.

V.A Light Source Replacement

Following a number of technical discussions during client meetings, the exact spectrum range we need became clear; the spectrum of interest from the client was approximately 650 - 750 nm. At this time the optical engineer acquired valuable information regarding a smaller light source, the MBB1F1. As demonstrated in figure 8, the MBB1F1 outperformed the older, larger light source within our spectrum of interest.

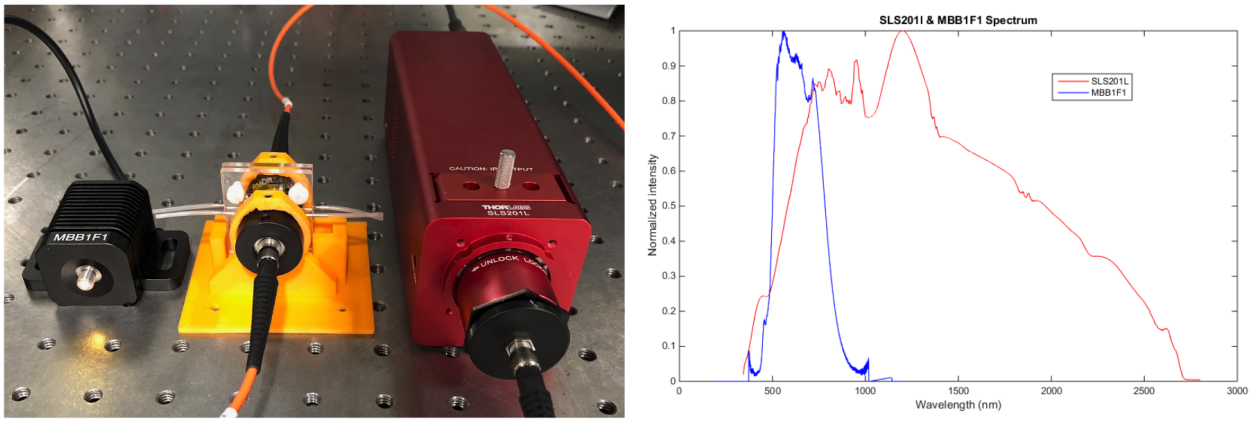


Figure 8: Light Source Spectrum Comparison

The second main reason for changing the light source other than how much space it took up was the heat dissipation. The original light source, SL201L, gave off a lot of heat after being turned on for about thirty minutes. This was due to the amount of power it output over the excessive range of wavelengths. Upon discovery of the new light source, it was clearly the optimal design choice given the size reduction, heat reduction, and high efficiency. With this new light source the mechanical engineer was able to further minimize the enclosure design to meet the client's request.

V.B Spectrometer Replacement

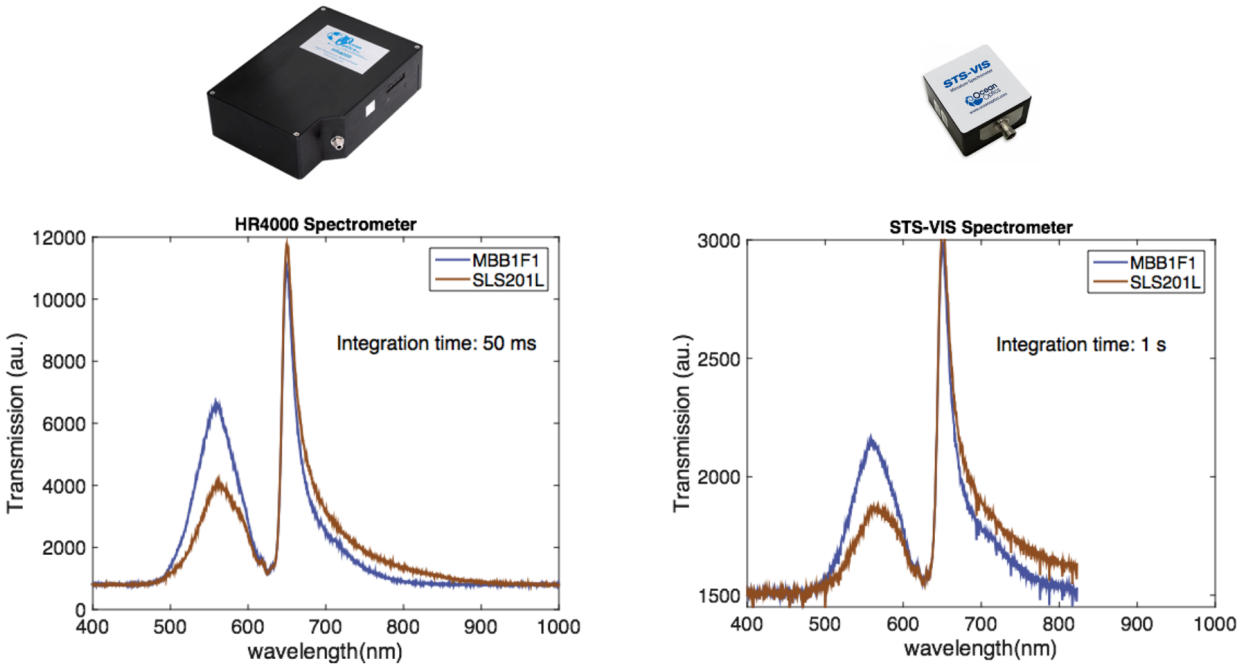


Figure 9: Spectrometer Light Intensity Captured Comparison

With new LED MBB1F1 in place, the Spectrometer was the next component to replace. Utilizing the STS-Vis spectrometer, with an area of about 1 cubic inch, provided a large improvement in size. In the process of testing it, we observed the data found in figure 9. The number of photons captured by the STS-VIS were four times smaller but well above noise ratio to tell any spectrum shifts. The STS-VIS worked great as seen in the plots above with a slightly longer integration time, which was not a problem for the client. The STS-VIS offered 95% size reduction in comparison to the HR4000, which itself is about the size of our final system. This can be seen in images of the fully integrated system shown under the results section towards the end on our report.

VI. Software Design

Software development during this quarter saw a significant overhaul in functionality. Initially, the control software was essentially a series of testbed functions called sequentially to verify performance with something of a "mock functionality". This served quite well for testing purposes, but required a monitor/keyboard and user intervention. The completed version has assumed a much more autonomous and transparent form. The system operates on a client-server model between the Raspberry Pi and an Android application, the plans for which were laid down during winter quarter and implemented now. In short, the raspberry pi awaits commands from a connected cell phone and responds by activating corresponding functions in the hardware through a custom hardware driver.

VI.A Raspberry Pi:

The raspberry pi, within the device, functions as a server and mediator between the user and the hardware. When connected to the control application by bluetooth, the user is able to interact with each hardware component of the device, as well as apply usage-specific settings and obtain data about the operation. The server code is written in C.

After a steep learning curve and lots of example code, we were able to interact programmatically at a low level with the bluetooth hardware onboard the raspberry pi. The system uses BlueZ, the official linux bluetooth stack/API, sitting atop the standard Sockets library, to access the transceiver and host a server socket. When the android code attempts to find and connect to this socket, a serial-link relationship is established, allowing the two to share bytes of data in half-duplex mode.

Once communication is established, the server sits in a main loop and awaits command bytes. The commands are enumerated on both sides of the code, ensuring that each device is using the same set of commands. Upon receiving bytes, the server parses the commands and executes the corresponding code if a valid code is received. This functionality is summed up in figure 10 below. For commands which also include a data payload (setting parameters, for example) the packet is parsed and handled accordingly.

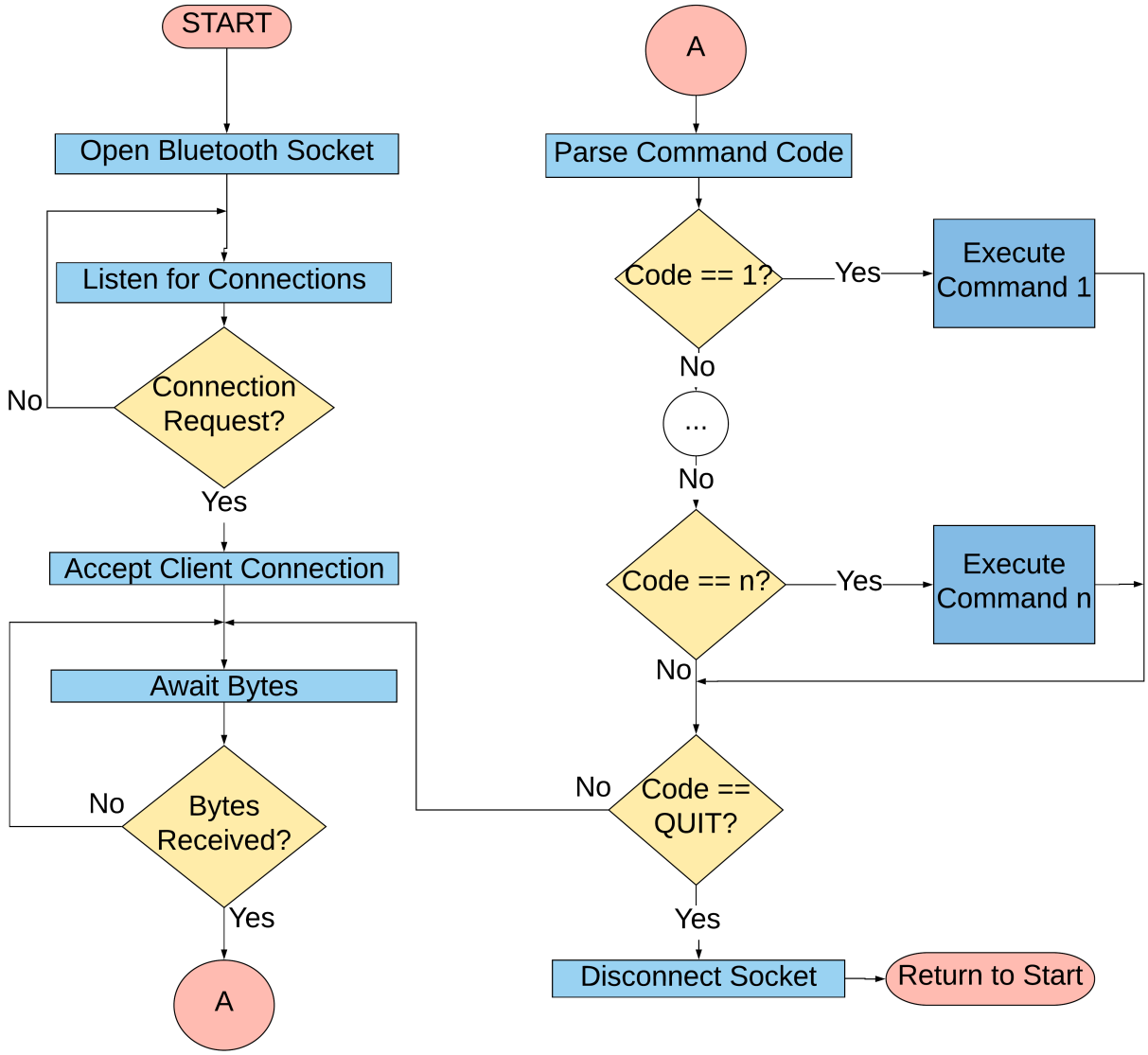


Figure 10: Server-side Behavior

The commands currently available within the system are:
MOTOR_ON;
MOTOR_OFF;
LIGHT_ON;
LIGHT_OFF;
REQUEST_PRESSURE;
REQUEST_SPECTRA;
INTEGRATION_TIME;
QUIT;
with the capability of adding arbitrary functions as needed in the future.

The raspberry pi is connected to each hardware component via GPIO or USB, including:

Pressure sensor: read from ADC chip using the SPI interface;
Micro air pump: controlled with PWM through an H-Bridge chip;
LED: switched via our LED driver circuit;
Spectrometer: connected via USB

In order to facilitate the server interaction with hardware, a standalone driver module for the hardware was developed. The module provides convenient access to each component of the device, including the pressure sensor, LED, motor, and spectrometer unit. The module also handles hardware initialization procedures, during which the raspberry pi GPIO is setup, the spectrometer unit activated, SPI communication initiated, settings applied, and various other aspects of hardware setup.

The pressure reading command is unique in that it starts a data stream since the user needs continuous access to the pump readings. For this functionality, a server thread is started which sends a pressure reading once per second until commanded to stop.

The server code has been set up to run as soon as the Raspberry Pi boots, so that the application can connect to it without the need of an external screen. With this design, the unit is completely standalone.

VI.B Application:

Following the plans designed during winter quarter, the control application was developed and tested. It was developed in Java using Android Studio and its associated developer packages.

The application serves as the connection between the user and the server. In this sense, it is the face of the device. The main screen provides the user with a number of buttons related to the device functionality, including hardware control and readouts of the pressure reading and connection status.

Before doing anything else, the user must connect the bluetooth device. None of the other buttons will respond until this is done. After a connection is established, the status message is updated, and the buttons become active. In the connected state, each button has an associated onClick() function, meaning that designated code will execute upon a detected click. In short, a button press corresponds to a function call within the code, which then sends the associated command to the server; the server responds accordingly as described above.

The settings screen comes loaded with defaults, but allows the user to alter the settings as their needs may dictate. The available settings include integration time (a critical spectrometer parameter), how many scans to take, and a number of data filtering options.

The Main and Settings screens can be seen in further detail in figure 11.

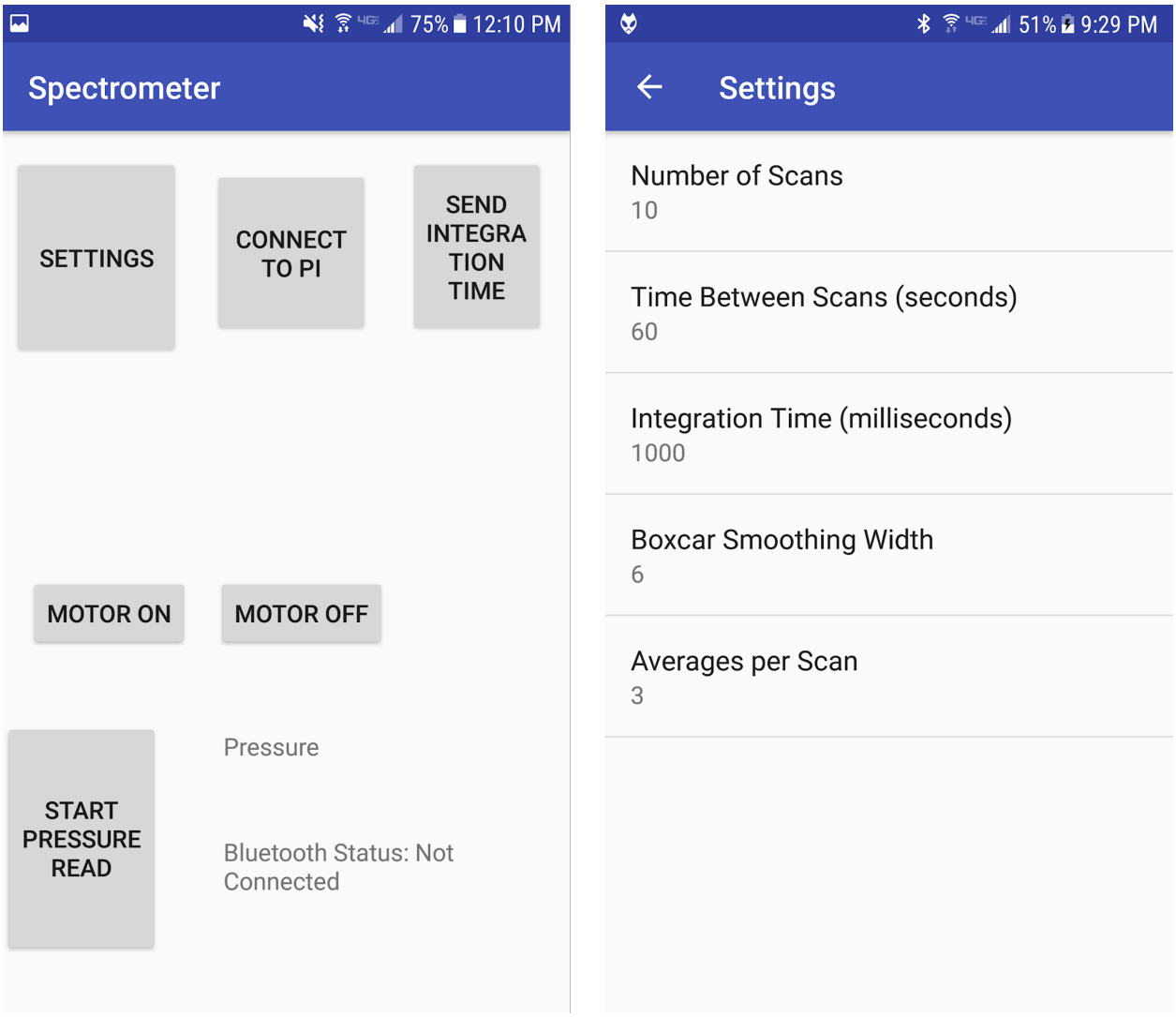


Figure 11: Left: Main Screen; Right: Settings Menu

Implementing bluetooth for this project was not an easy task; much effort was placed into the learning curve associated with getting communication up and running between the two devices. It was further complicated by the fact that each platform uses a different operating system and coding language! Still, studying a number of examples and open-source applications allowed us to reach a point of proper connectivity.

As described above, the server (raspberry pi) hosts a socket and listens for incoming connections. The client (our application) must connect to this socket without blocking/binding up the phone. For this reason, a bluetooth service module was developed within the application utilizing threading techniques. This service used official Android sample code as a basis, but was heavily modified to our needs.

The bluetooth service instantiates a thread which searches and attempts to connect to a socket. Once detected, the thread returns a socket object which is connected to the device it

found. The service then uses this socket to instantiate another thread, which this time never returns. The connected thread provides the service with read() and write() functions (which interface to the underlying I/O streams.) With this connection, the main() function of the application can now interact with the service object through and inter-object messaging system used in Android development; in short, the system is now connected and will activate the buttons. A simplified UML diagram of this service is provided below:

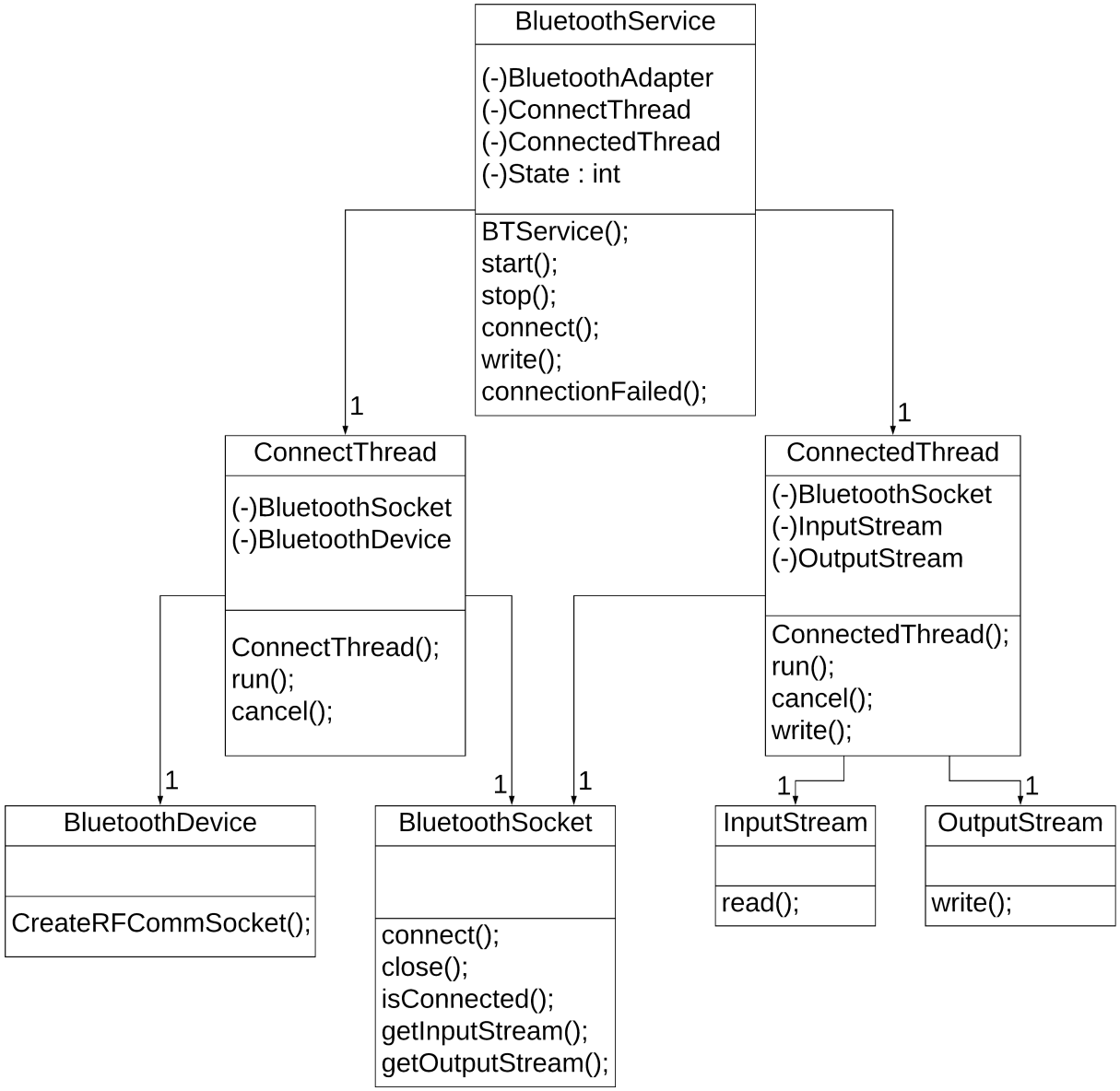


Figure 12: Client-side Bluetooth Service

Part 3: Fabrication

I. Microfluidic Pump

The next steps were focused on manufacturing a pumping system to incorporate with the optical system that we have in place. After our preliminary research on pumping techniques,we were able to find a suitable solution to pump fluid at the low flow rates required.

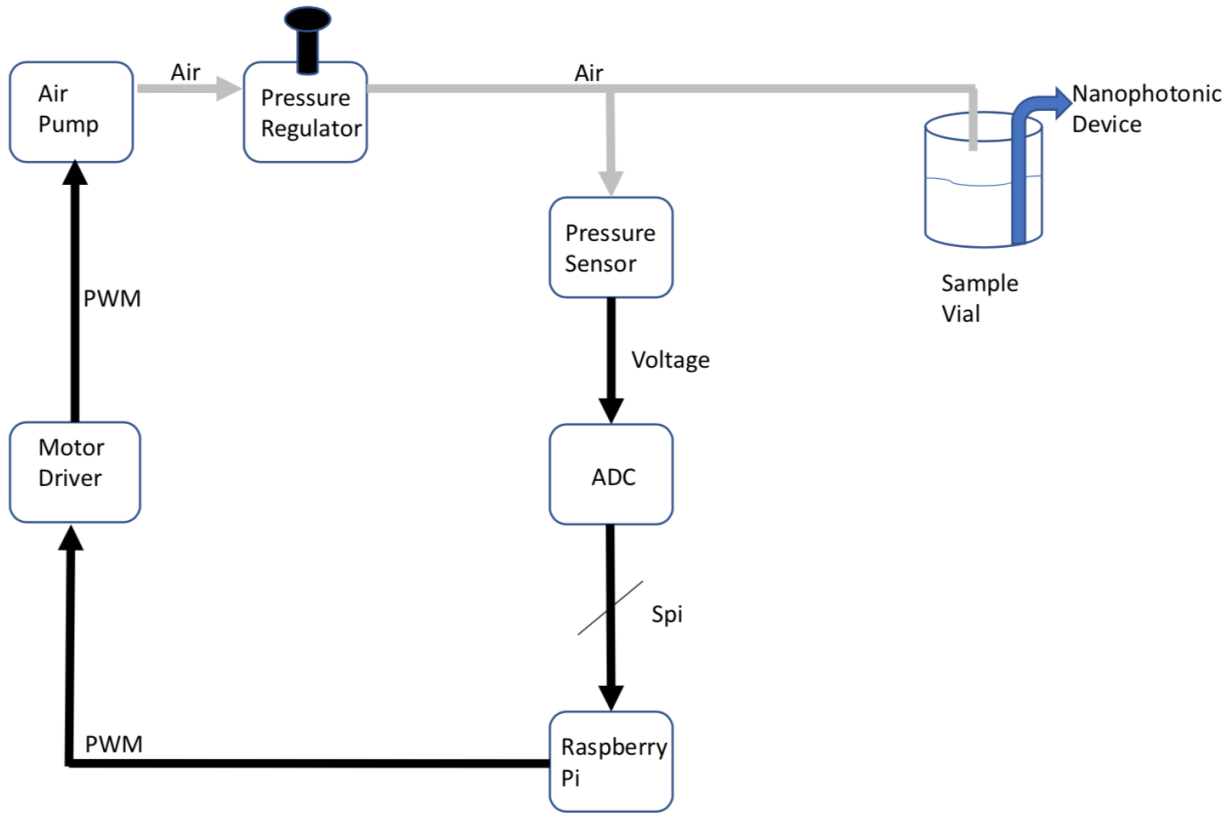


Figure 13: Microfluidic Pump Block Diagram

As can be seen in Figure 13, the pumping system consisted of an air pump, a pressure regulator, and a pressure sensor. Those were the main components needed to characterize the flow rate generated by the pumping system. The other components seen in the block diagram were used to communicate with the Raspberry Pi. Given that the Raspberry Pi lacks analog inputs, we used a common ADC to send the reading from the analog pressure sensor to the Raspberry Pi. The chip communicates with the Pi via the SPI protocol.

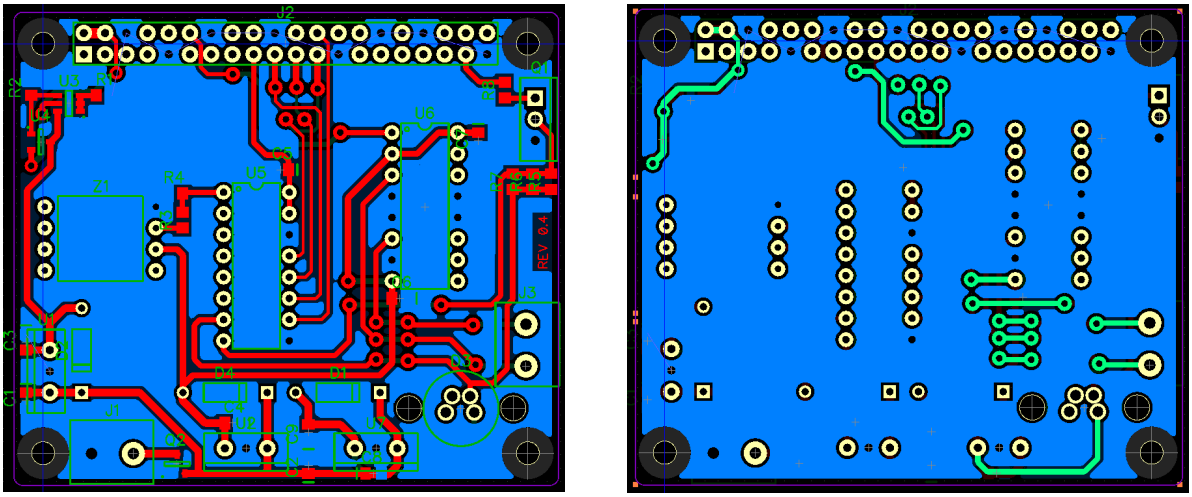
The motor driver was needed to power the air pump, as it requires high current. The Pi was used as a control input to the driver, which consisted of a simple H-Bridge. This setup allowed switching control of the air pressure generator.

These components work together to pressurize a sample vial with very low pressure, in order to pump the sample out.

Once the pumping system was developed, we ran experiments to characterize the flow rate. Through these experiments we were able to reduce the flow rate from 8 mL per minute without a regulator to 100μL per minute when we used the regulator. This flow rate is still too fast for the clients nano-plasmonic chip, so we still need to improve in this area. Due to the fact that we took a modular approach we can easily replace the current regulator and put in it's place a more appropriate regulator. With a more precise part in place, we can regulate down to the required flow rate of 20μL per minute.

II. PCB/power distribution

As planned during winter quarter, we found ourselves in need of a centralized board to distribute power, hold chips, and route signals to/from the Raspberry Pi. These plans were formalized and completed during this quarter, resulting in the following design.



(a) Top side of the PCB

(b) Bottom side of the PCB

Figure 14: Daughter board PCB layout

With the daughter board in Figure 14 we were able to achieve both power distribution and as a custom LED driver for the light source previously mentioned. We decided that it would be best to forgo powering the Pi through the micro USB port and to power it directly through the daughter board. We were able to achieve this by using a 5 Volt 2.1 Amp linear voltage regulator. In order to ensure the Pi didn't get damaged we went ahead and designed our own reverse polarity protection, since powering it through the GPIO bypasses the internal protection. The Pi gets power from the 5 Volt rail that is on pins 2 and 4. In this manner, we reduced the system down to just one single plug.

The LED driver was one of the most important components powered by the board, and thus we needed to ensure that we didn't blow it up. The LED wanted a forward voltage of 3.6V and a max current of 500 mA. The electrical and systems integration engineers got together to design a simple LED driver that would not surpass this current. In order to do

that we used a Darlington with three current limiting resistors. The reason we used three resistors instead of one was to evenly dissipate the power loss generated across the resistors.

Following a good deal of theoretical calculations and discussion with management, we knew we needed a 2.1 ohm resistor to draw about 475 mA of current. After we laid out the board and milled it we looked at the actual current draw across the LED and realized it was around 480 mA; a successful design. We designed for a current that was less than the max since we knew that there are component variations and wanted to err on the side of caution in order to not burn the expensive LED out.

In the Figure below, Figure 15, we see the full schematic of the daughter board including the pressure sensor, ADC, and motor driver IC's. These components were interfaced with the Pi primarily and received power from one of the 5 Volt regulators.By having the power distribution board also be a Raspberry Pi daughter board we were able to minimize the space we used inside the enclosure. As a result we further improved the overall device form factor.

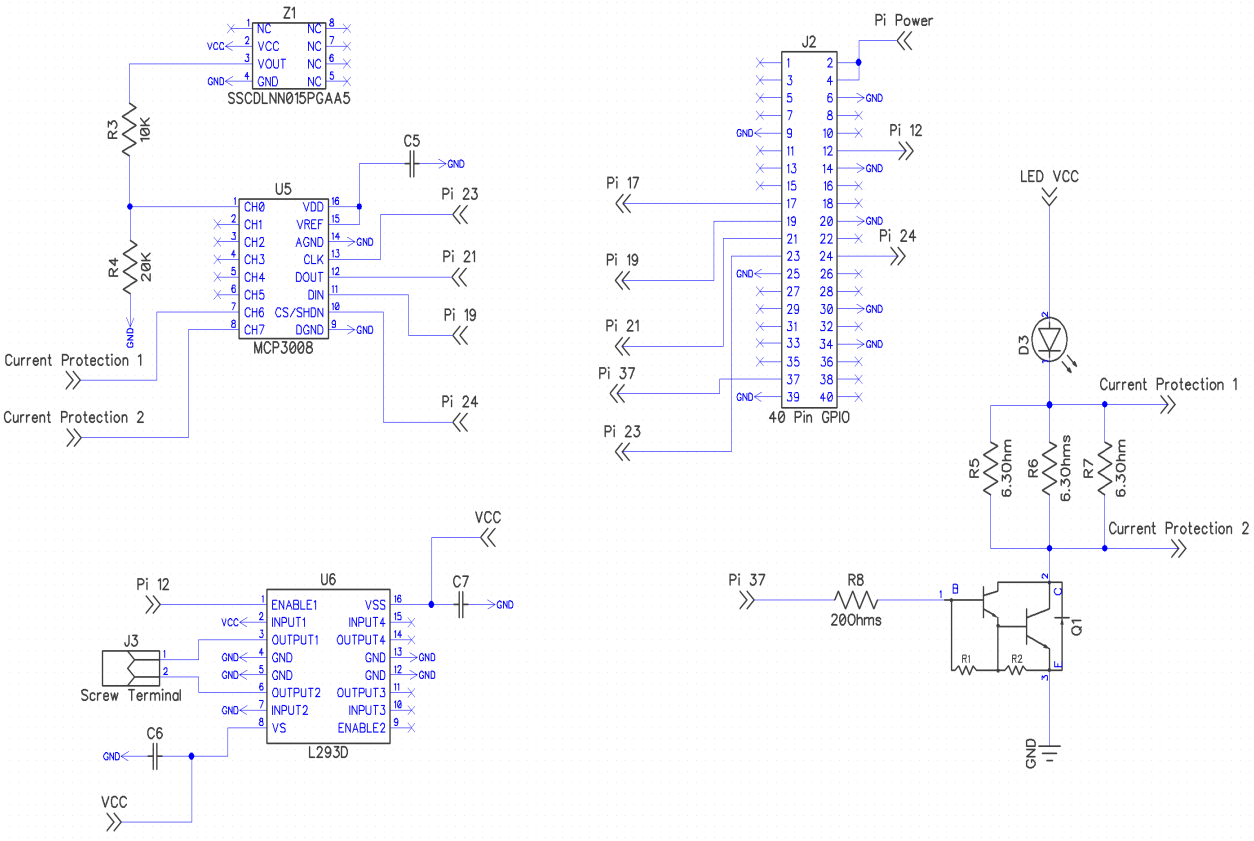


Figure 15: Daughter board schematic

III. Enclosure

The final prototype enclosure can be seen in Figures 16 and 17. This design was made using Acrylic and put together using the tab and slot technique. The material was chosen due to the fact that we had experience with it, and knew that it is light and strong enough to support the internal hardware without bending.

Figure 16 shows the components from the side. In this rendering, the part which appear to be floating are in reality side-mounted. By doing this we were able to open up more space in the base platform for the Raspberry Pi and the pumping system. The side view also demonstrates the separate layer that we had on this enclosure. We decided on constructing a lower level where we could route all the cables and the tubing for the pumping system. We came to this approach after trying to fit everything on a single layer and failing due to the large amount of wiring and tubing. The addition of this bottom layer allows the parts to fit comfortably with the cabling routed underneath. The final result was a clean and open design.

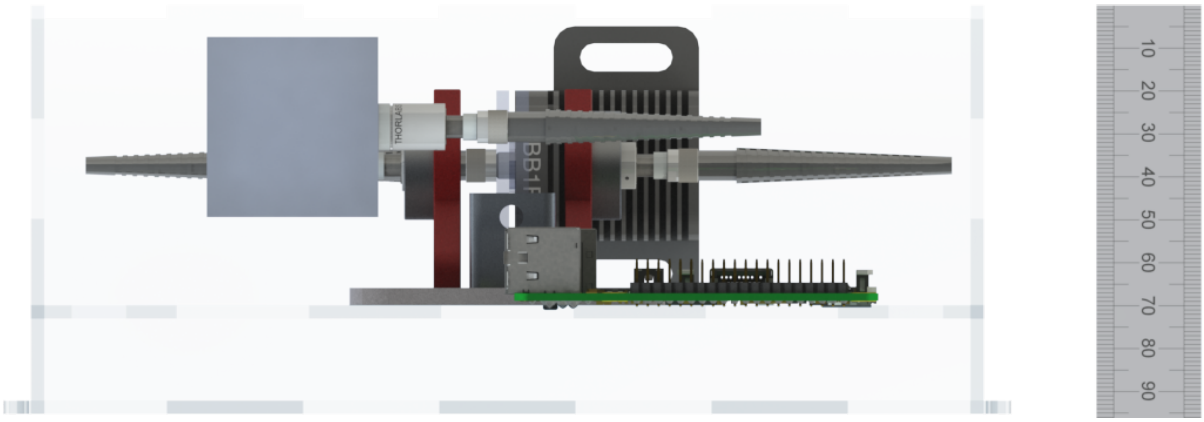


Figure 16: Side View Placement of Current Parts

In Figure 17 we can see how much space we have with the two layer design. The bottom later is accessible through the two long vertical slits that run near the sides of the enclosure.

Following these steps, we reached the current version of the enclosure, which we are quite happy with.

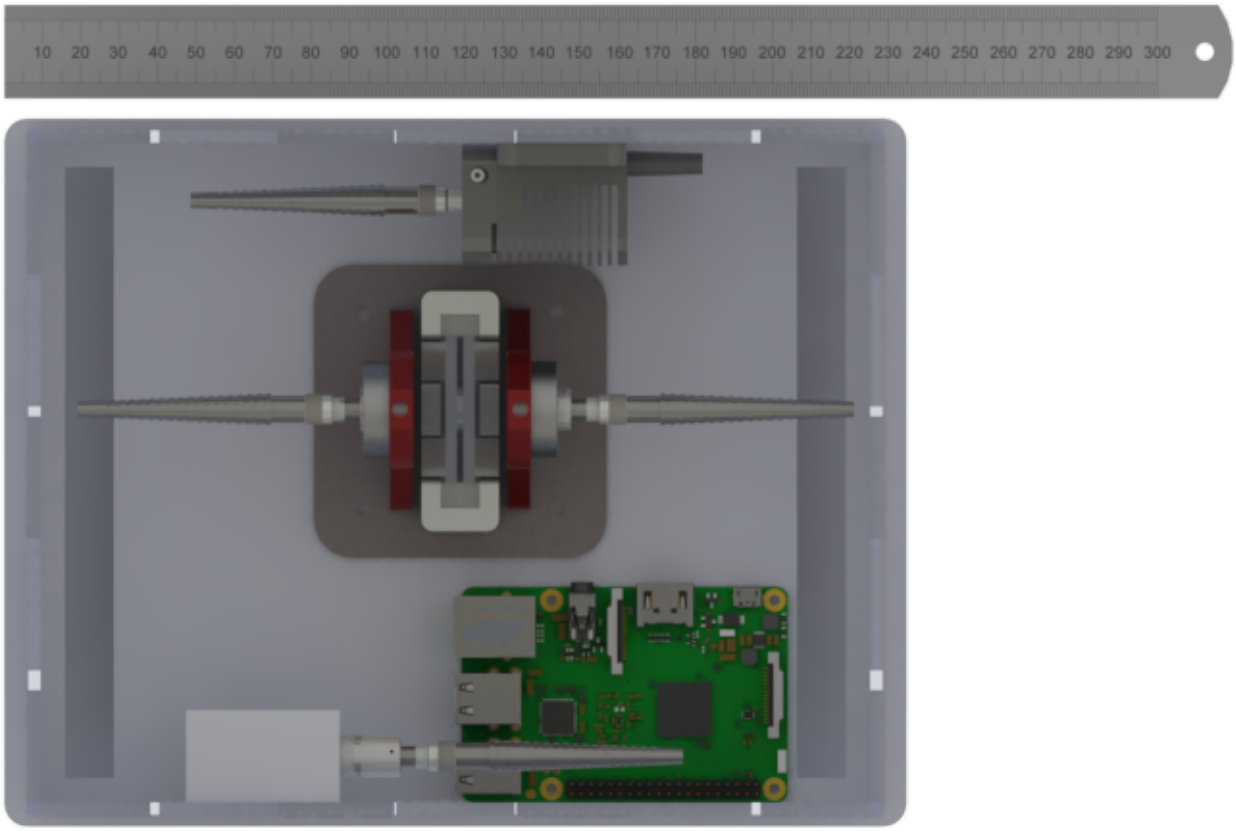


Figure 17: Top View Placement of Current Parts

IV. 3D Printed Base

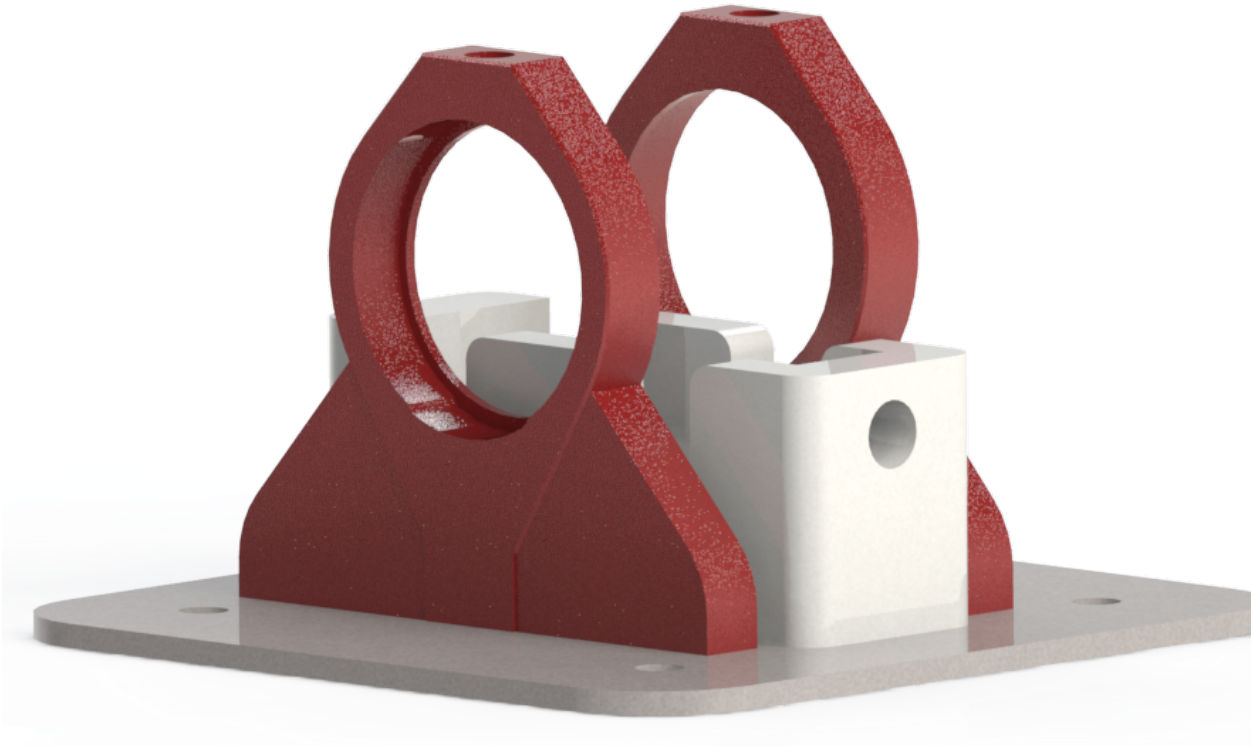


Figure 18: Assembled Optical Mounting Components

As can be seen in figure 18 we have assembled all the components required to mount the optical system in conjunction with the the sample. These parts were originally designed and tested individually for quality purposes. We first started with the sample holder; the design began with general sample dimensions as a rough starting point. The job of the Hardware Engineer was to figure out a way to securely mount the sample in way that was accessible to the user. The sample holder had to be positioned in a way that the tubing will not interfere with the rest of the optical system therefore various iterations were manufactured until the optimal design was achieved. The sample holder was then designed to be slightly wider than the sample for easy placement, yet not too wide that the sample couldn't lean on the edges for support when unsecured. The sample holder is the center part of the assembly in figure 18.

The two optical fiber mounts were designed to hold two optical fibers of different sizes, with each being manually adjusted to the right height for optimum transfer of light. In conjunction with the optical engineer there were various test that were done to measure the dispersion of light for each iteration of the fiber holder.The two fiber holders were iteratively printed until we were able to make sure they were concentric. We faced difficulties due to the fact that we do not have full control of the precision of the 3D printer we use. The printer tends to print the parts larger than expected, and was found to have an inconsistent margin of error. This part was originally modeled after a optical holder purchased from Thorlabs. In order to maintain structural integrity, some modifications were made during the design

process. Upon first printing, the parts broke at the base from the stress; this was addressed in later iterations.

Finally the base platform was designed to hold the optical components at an optimum distance from each other. That means that each optical fiber mount is placed at a precise distance from the sample holder and each other to achieve optimum light collimation. This whole piece was then placed on the devise enclosure which includes the light source, raspberry pi, spectrometer, and the integrated pump.

Part 4: Results

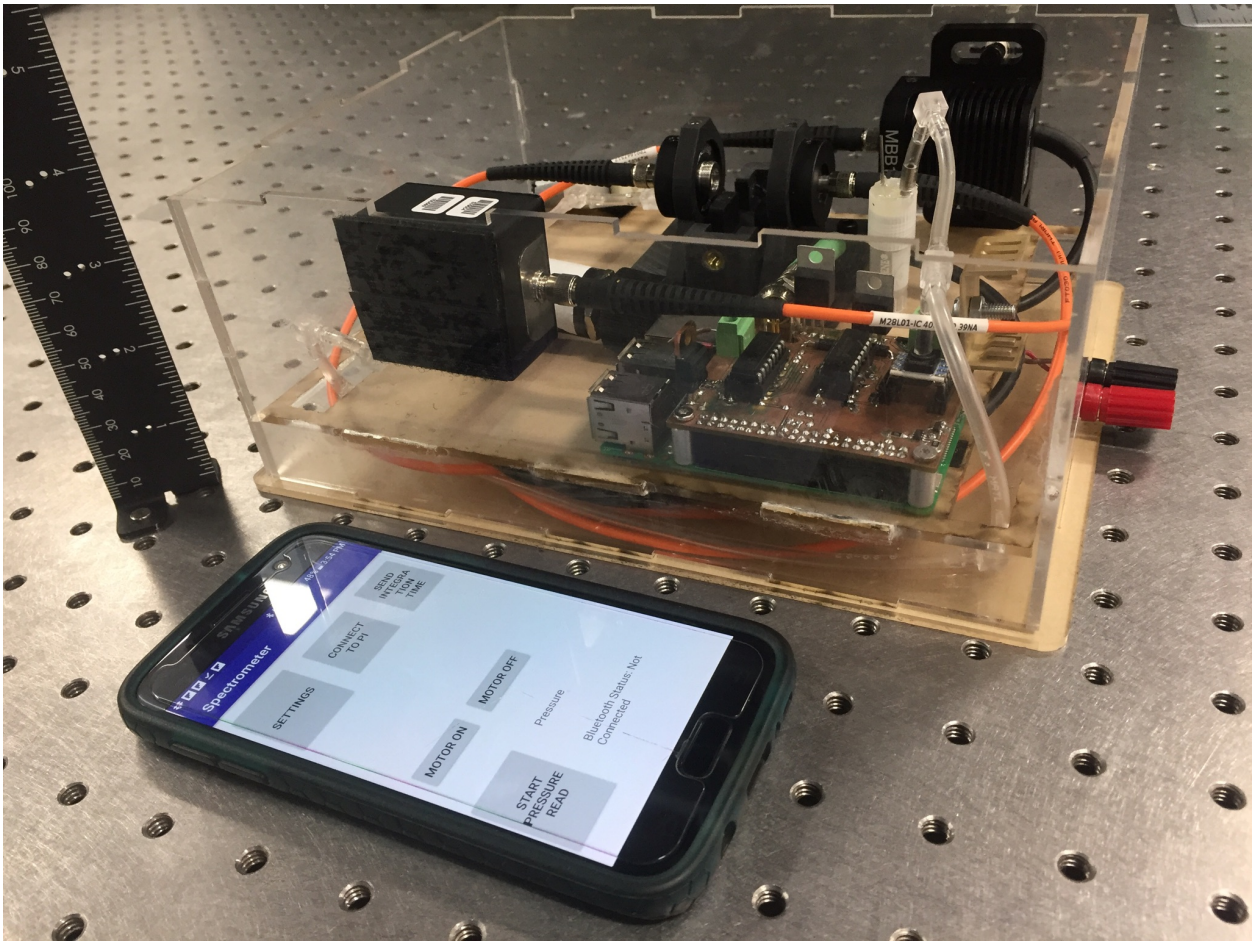


Figure 19: Device with App

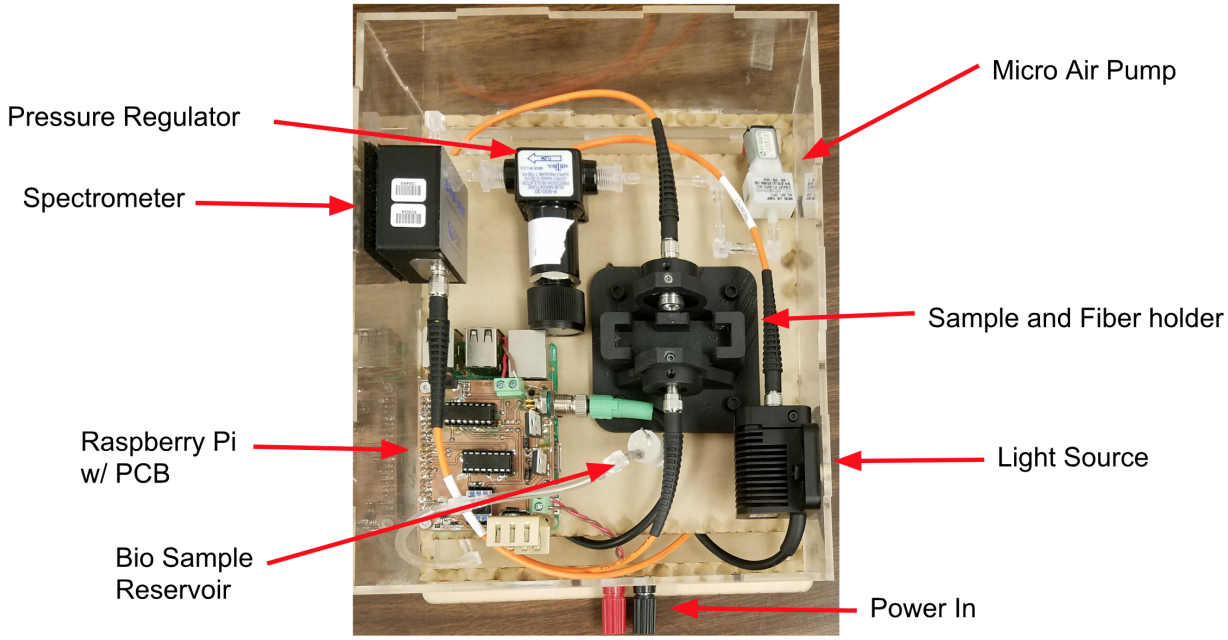


Figure 20: Final System Integration

Conclusion

Following many hours of work, we are happy to have produced the final design detailed above. While it has certainly been a difficult road, it was all worth it just to see the look on the client's face as we revealed the product. We have each learned many new things and explored topics which would have been otherwise unfamiliar to us. This project, which has incorporated topics such as optics, fluidics, programming, electrical/mechanical/software engineering, and true teamwork, is not one we will soon forget. Furthermore, the experience in dealing with management, handling client meetings, and otherwise learning how to interact professionally with other humans, will prove invaluable in the decades to come as we attempt to venture out into the real world.

Thanks everyone!