Portable Microscope

Group 27

Austin Bryant - Computer Scientist

Adam Bush - Computer Engineer

Cayla Gill - Electrical Engineer

Hannah Pierson - Photonic Engineer

2019-04-22

Table of Contents

List of Figures

List of Tables

1.0 Executive Summary

Traditional microscopy has been around for a few hundred years, and it has progressed significantly in terms of imaging. However, in terms of user friendliness and portability the microscope is severely lacking, it is usually quite bulky and full of complex optical components that are difficult to navigate for the beginner. In most primary schools, they have microscopes for kids to use but they are all stationary and fragile. The problem with being so fragile is that they are prone to breaking if you pick them up the wrong way, this is why they must remain stationary. They are bulky and difficult to focus for each child, and most do not have a camera. Their bulkiness comes from the plethora of optical components inside such as a condenser, many lens, mirrors, a light source, the eyepiece, the translation stage, and several objective lenses. All of these components do improve the resolution of the microscope, but it is not necessary for the typical things studied in a classroom setting. Our microscope will be much simpler and still be able to image things as small as an onion cell which is a very common thing to look at under a microscope in a classroom setting. Our microscope also solves the issue of being difficult to focus for individual users, it will only need to be focused one time for each specimen as it will have a camera instead of an eyepiece. Because our microscope will be smaller and have less components it will be easier for a user to navigate and be portable.

Our original design was a multi-spectra portable microscope that was able to image in visible and infrared spectra. We built and use a custom PCB that had a Wi-Fi module, an image sensor, and a microprocessor that controlled these things. The aforementioned PCB was scrapped due to extenuating circumstances regarding the embedded systems, had to be scrapped and replaced with a Raspberry Pi. This was our alternative plan, and the plan we have executed at this time. Due to the replacement, we were unable to accommodate multi-spectra imaging, as our sensor in the alternative plan did not allow it. However, it is still a fully functioning wireless, portable microscope that interfaces with an android device.

The app-enabled microscope is be a fully functional microscope that is more portable than traditional microscopes. It has a sleek visual appeal that immediately draws attention to it. Our microscope will has two different levels of magnification so that a viewer can see a wide range of samples. The smallest detail our microscope can resolve is a human blood cell, which is approximately 8.2 um in diameter. This is attainable with a DIN standard objective lens of 10 x magnification with a .17 numerical aperture. This microscope is limited to the visible spectrum of 400-700 nm, we hope to add the infrared range of 850 nm in the future. Our microscope utilizes two light sources, a top light for thick samples and a base light for thin samples. A Raspberry Pi transmits images using a built in Wi-Fi chip as well as an image sensor. All of the electrical components in this microscope are battery operated so as to keep the microscope portable. These batteries are lithium ion because they are rechargeable and last a long time, this will also keep the weight of the microscope down.

The microscope utilizes a system of one objective lens, a 150 mm tube, and a 3 mm aperture that projects an image of the specimen onto a 5 mega pixel image sensor. The image sensor then sends the data to an android device via WAP. Once the image has made it to the android device, it can be viewed in a specially made application. The application allows the user to take pictures or live video. Within the application, the user is able to manipulate the images and/or video in a variety of ways. Some ways to manipulate the image include blurring filters, edge detection filters, and tagging of the photo. The user can also take single frames from the recorded video. The user has the option of naming and saving all of the images either before or after the images have been taken. This will be useful for collecting data and sharing images.

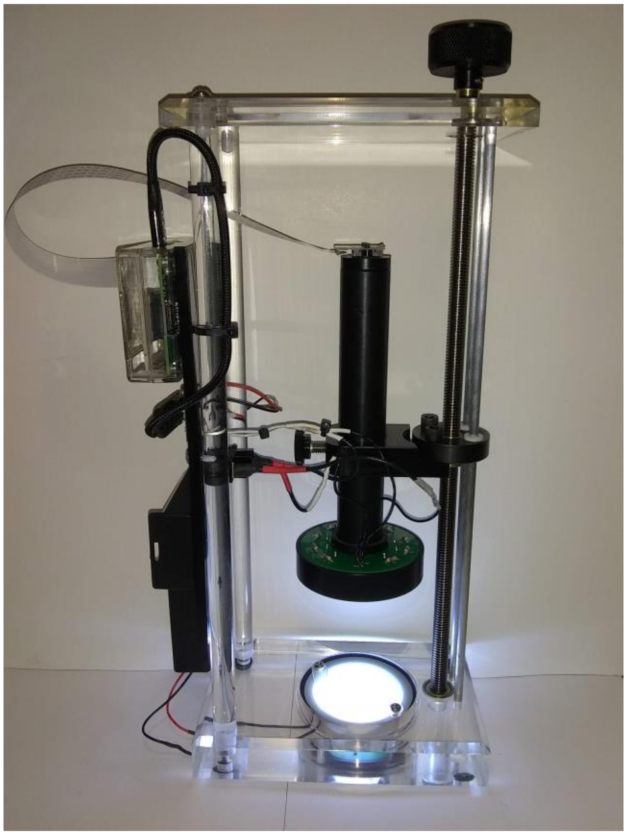


Figure 1: Alternative Design:
The alternative plan that was executed

2.0 Project Description

Our project is a scientific learning tool. It can be used in a classroom setting to introduce children to science, technology, engineering and mathematical (STEM) fields as well as in a tutoring environment to assist in explaining conceptual material. It will help display another perspective of our world that is not normally seen and will expand the knowledge of future scientists' and engineers' knowledge of the building blocks of our world. It is a tool to aid in providing a new perspective as well as increasing an excitement for the scientific world and the true wonders that are presented around us. Without this tool and the technology involved, these goals would not be achieved.

Our microscope has removable features. The base light and the top light are easy to remove from the system. Our microscope has interchangeable Deutsche Industrial Normen standard lenses, which means that any objective lens that is DIN will fit into our system. It is also battery powered which makes it wireless. The sensor that was chosen can also be replaced very easily if the user needed to, as it is mounted into our tube using a standard C mount. The raspberry pi is mounted in a clear case that comes apart so it is easy to remove and replace the pi if necessary.

Our microscope features a modern design of clear acrylic, this enhances its educational purposes. All of the features are visible including all wiring and circuitry. This allows people to readily see where the power is being dispersed.

2.1 Motivation

In this section of the paper, we go in detail of why we were interested in building this particular project. Each student has described what makes this an interesting project for them to work on and how it will benefit them. The first motivation is the optical engineering student, then the electrical engineering student, followed by the computer science student, and then the computer engineering student.

Optics The idea for a portable microscope came from my fascination with the unseen world. Our eyes are capable of observing a small fraction of the world that we live in. We are limited by the cones that can only sense red, blue, and green as well as the rods that can only pick up light and dark. The world around us has a much larger spectrum of color, while we cannot imagine these colors we have created ways to detect them. Then there is the microscopic unseen world. This world is vast and thriving but, due to the lack of resolving power in our eyes, we cannot see it. The microscope is a small window into that world. It allows us to see the building blocks of everything around us and tiny organisms that we did not know existed. This unseen world is what I want to share with others and inspire them to explore.

Aside from wanting to share the unseen world with people, I feel that this project is ideal for me to apply the geometric optics I have learned to a real situation. Be able to use what I have learned from the past few years of studying optics and ray tracing is an exciting venture for me, and something that I can be proud to put on my resume. This also allows me to study microscopy at in depth level, that was unachievable in normal course work.

Electrical My motivation for this project came from the desire to perpetuate learning. As a child, I was fascinated with microbiology, and often considered going into the medical field. I decided I wanted to be on the engineering side of the technological fields, but I have maintained my interest in the medical and microbiological fields. I have many friends in medical school, one of which is studying to be a pathologist. These types of analysts rely on microscopy in order to find problems in human cells and diagnose illnesses. Though this project will only be a prototype for a portable microscope, with more research and development, this device could help many people analyze microscopic worlds. This device could also assist in motivating younger age children to be more interested in the science, technology, engineering and math (STEM) fields. I remember being so interested in science and biology as a child. This is another tool that could be used easily to teach children about the world that is unseen to the naked eye. Seeing these "invisible" things is fascinating, and I want to share that fascination with others. This project was ideal for me in that I can help others with medical sciences while also helping teach children the wonders of the technological world.

As the electrical engineering field, this project stood out to me because of the necessary electrical components presented. Working with a diverse team, I would be able to learn communication with other majors in order to produce a working product. This project also meant working with components I have never used before, like image sensors and Wi-Fi chips. Powering circuits has not been a familiar aspect of electrical engineering for me, but this project poses new challenges that I am prepared to face. This project also presents the opportunity for me to learn new programs and aspects of circuit design that will advance my knowledge in the electronics field.

Computer Science I'm personally motivated to work on this project because I believe that a portable microscope is a fantastic tool that can be used in classrooms, workplaces, or even in someone's home for personal use. As a kid, I loved taking biology classes and getting to look at different organisms and objects underneath a microscope. That being said, I didn't always enjoy having to wait my turn to look through the eyepiece and then only getting about twenty seconds to look at whatever was there. This project completely eliminates that issue and provides a solution that takes away the need for a tethered base and allows sharing to your mobile phone. In addition to that, possessing the ability to take pictures of the things you see under the microscope and share them with your friends could encourage kids and even adults to learn more about the world around them and become interested in science.

Computer Engineering I came into the course hoping to design an accessible, people-focused device, with initial thought toward hearing aids. A learning tool fits that description just as well, and the handheld, small-scale nature of our project meshes perfectly with a need I know firsthand exists. I tutored in and helped manage the science lab at my hometown's community college, FSCJ, with the responsibility to select to-be-acquired learning aids and novelties from educational science manufacturer catalogues. These models, whether it was the replaceable-organ skeleton, the 3D VESPR modeling kit, or the potential energy physics kit were truly helpful. A lot of the people who came in for help didn't have much of a background in the topics, and science is widely considered foreign and challenging. Providing engaging, interesting, approachable learning tools to students of any age is seriously valuable, and in our group's experience, walking around (or sitting down with!) a portable microscope is a very stimulating and thought-provoking experience.

From a technical standpoint, I've taken two courses now on computer networking, and I had large interest in personally connecting two devices by Wi-Fi; I’ve found image processing very interesting, and my only true exposure was during an introductory MATLAB class; and I’ve been constantly excited to learn more about image sensors at the device level.

2.2 Goals and Objectives

This portion of the paper describes what we are going to accomplish and how we are going to do it. It will give the reader a detailed description of what our project will be capable of. It also tells the reader our stretch goals, essentially what we want to accomplish if we have enough time to do so.

- The system functions well as a microscope relative to the market, and provides a clear magnified microscopic image.
- The system is lightweight and portable. Having a lightweight and portable system helps with the user interaction. Bulky and heavy materials would have limited the system in its capabilities for the user. Having this as a goal made it easier for the user to manipulate the system to view certain objects or specimens.
- The system uses batteries as a power source. Since the system is a low power system due to the components on the board, using batteries as a power source is a viable option for this system. Having batteries as a power source also contributes to the goal for the system to be lightweight and portable.
- The system is able to display the output of the image sensor to a separate device in real time and at a visually appealing speed. We wanted the user to be able to receive real time video streaming as well as image acquisition.
- The system is able to transfer all data between devices wirelessly. This adds to the ability of keeping the system portable. The fewer wired connections, the easier it is to maintain a user-friendly system.
- The system is application enabled. Having this system paired with an app makes the interaction of the system easier for the user. The user can use their phone to obtain the real time video and image acquisition wirelessly.
- The system is able to view the object or specimen in the visible spectrum. Having this increases the user’s ability to see changes to objects and specimens that cannot be seen with the naked eye.

We made this project to be a variable mode microscope that has the advantage of being portable, without a stand for stability, as some stabilization is a necessity in current imaging technologies. We made this device to be able to interface with other devices, such as smartphones. Though there are other similar devices which provide magnification as well as lighting, ours will differ because of our better resolution and higher optical magnification.

2.3 Requirements and Specifications

This section is dedicated the design requirements and the specifications to make a successful microscope. There is a list of the design requirements that describe what our project must do as a minimum. It will give the reader a solid idea of what they can expect from our project. It will also compare and contrast the requirements and the goals, so that the reader may understand that they are two separate things, the goals are what we want and the requirements are what we will make.

- The system is able to provide the user with a clear image of a cluster of onion cells and is able to distinguish the walls of the onion cells, approximately 120µm.
- The system has multiple magnifications available to the user.
- The system is able to image in the visible spectrum.
- The system maintains less than one second of delay from the capture of the image to the presentation of it.
- The system has an average frame rate of forty frames per second (FPS) for real time use for the user.
- The system is able to transmit all visual data to a modern smartphone.
- The system maintains a manageable weight of less than ten pounds.
- The system should be one completed structure with no loose pieces.
- The system can be used with slides or backlighting to observe the object or specimen.
- The system will use direct light from the body of the design as lighting for the object or specimen being observed.

These requirements stem from goals 1, 3, 4, 5, 6, 7, and 8. The onion cell is chosen as a representative marker of magnification quality and a real-world standard of introductory laboratory and explorative functionality. This is accomplished with an objective lens, which is the target used in the below house of quality (Figure 1). Multiple resolutions and is demanded because our device must be capable of imaging distinctly different things. The weight requirement was chosen through consultation with customers and the understanding that microscopes require a very stable base to provide meaningful images, and 10 pounds was the safest design limit that is also acceptable for end users. Display rate and delay stem from current expectations of technology, and each is within the limits of the technical constraints (spoken of below), but each is technically demanding. Smartphone integration has proven itself as a means of setting any product above the rest of the field, and their ubiquitous adoption provides us with consistent, high-quality access to a display, offloading that multi-faceted cost to standard user inventory, both of which encourage serious prioritization.

2.4 Anticipated Obstacles

This is an in-depth discussion of all the constraints imposed on each field. The constraints come from physical limitations, as well as industry standards. Each field of study working on this project discusses in detail the constraints that have been imposed on that field of study. They discuss what the constraints are, as well as what is causing the constraint and how to find a way to do what we want in spite of the constraints. This discussion starts with the optical constraints, then the embedded system, then the computer science, and concludes with the electrical constraints.

Optics On the optics side we were able to manage light saturation, if the light is too much we may have a saturated image that is difficult to see. We corrected this with a 3 mm aperture placed 6 mm in front of the sensor. Another constraint for the optics is how expensive the equipment is, we were able to find a set of DIN objective lenses that were only $46. These lenses did not require additional components and therefore helped keep the costs down. The mechanical design of the microscope itself is optics directly and was a challenge to design and build. The threaded rods used for the main moving element needed to have very sturdy threading that would not bend or break under a lot of use. We were able to use an ACME threaded rod which is well known for its durability.

Another constraint is the resolution and dynamic range, we want to see an onion cell clearly, this means we need a very high resolution in the camera and the optics. The objective lenses that we chose we perfect for this, they had a specific tube length of 150 mm. Once we got this built, the lens did its job perfectly.

There are many other constraints as well. Positioning the light in a way that will give us a clear focused image was a challenge because we put the light above the specimen instead of behind the specimen. We encountered many problems with this design, such as the placement, brightness, and contrast of the final image. The end result is we kept the lights because they work very well for thick specimen, but we also incorporated a back light to view thin samples.

Embedded System The primary design constraints will be the delay requirements on the image display, the limitations of embedded microcontrollers (MCUs), the wireless transmission, and the battery-provided power, in that order of significance, though each negatively interacts with the others. The delay requirements demand a more capable processor, which hurts the power consumption and requirement and puts stronger demands upon the wireless connection. The limited higher-rate storage in these devices (which is true of all realistic choices of microcontroller) require the entire system's software architecture to be designed around limited storage capacity. The extensive electromagnetic spectrum regulation will be a large-scale constraint, though it will mostly come into play through a chosen wireless specification (such as IEEE 802.11) and the shared partitioning/crowding of a chosen EM band. The capabilities of the device we pair our system with will be a consideration, as will the timing demands of the image sensor. There must be extreme attention paid to the bandwidth of the connection between the host microcontroller and the hosted Wi-Fi chip, as UART communications will fall short of the ~1MB/s transfer rate, as would single-SPI and similar. There will be some need to manage video and image data formatting or compression, which must be suitable for the paired phone and the limitations of the wireless format of choice, so additional computational power and memory will be needed. The size of the microscope will limit the size of the PCB, which will limit the space allotment for this system. Heat dissipation must also be considered under space- and airflow-constrained conditions.

The limited interaction and especially the mode of interaction with the embedded software, namely, it will all happen through an established wireless link, dictate a very robust, error-correcting control flow. Many users can be expected to have zero technological skills, so it must also perform regulatory functions autonomously, and its controls will likely be limited to an on/off switch, with similar implications. As the device will have no display of its own, it should have an LED placed next to or integrated into its power button to indicate it is on. Finally, there may be legal considerations on the video/image codec (or transcoding) front. Free formats should be chosen when possible to prevent complications.

Computer Science The constraints for the software side primarily dealt with transmission rate of the media files. Initially, our plan was to use Wi-Fi direct for the networking side of this project. However, the transmission rate of the media files was a meager 20 frames per second. To circumvent this, we changed our networking setup such that the Raspberry Pi would serve as a wireless access point. By doing this, we were able to improve the transmission rate to a consistent rate of 40 frames per second. A custom protocol was implemented that allowed the mobile application to determine the size of the byte array that was sent over the socket connection. This involved sending the size of the data as a byte array followed by a special character delimiter that was checked for by the mobile application. Once that special character was read, those bytes would be converted into its corresponding decimal value. The mobile application would then take that integer value and read that number of bytes, convert the data to byte array, and then a valid drawable object was able to be created from the byte array.

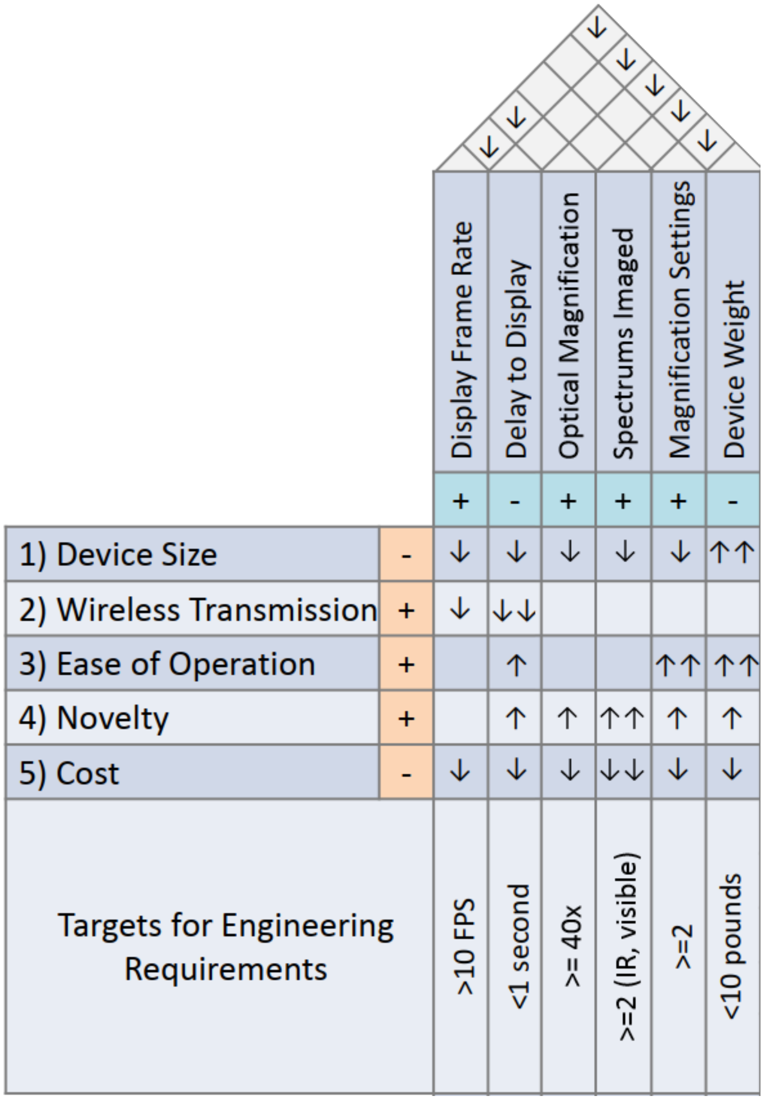
In addition to the data transmission process, the mobile app needed be intuitive with a strong emphasis on aesthetic appearance and functionality. Once the data was received, it it's able to be streamed seamlessly on the phone, which involved extensive research on the best practices of handling a continuous influx of data while displaying the frames on the screen. We also wanted to ensure that the user would be able to save the recorded videos/still images on their phone, which required a naming format that always supplied a unique filename. Additionally, we wanted to allow the user to perform simple image processing techniques on the captured media files. To achieve this, we used the popular image processing library, OpenCV.

We hosted the code used for this project on the senior design GitHub repositories that we were allocated for this class, meaning that no hosting costs were incurred.

Electrical The constraints for the electronic portion dealt with the size of the prototype. Since it was planned to be a handheld, portable device with a weight of less than ten pounds, there were constraints on both the circuitry as well as the power. The circuitry had to be constructed in a way that suited the overall body of the device. The batteries chosen also had to be formed in a way that fit the overall construction of the prototype. The batteries could not take up too much space, as there needed to be room for other things as lenses and PCBs. There was a constraint with power, as the power needed to work with all the different portions of this design. The designer had to take into account the safety measures involved with batteries and circuitry, make sure there were safety features as to not destroy circuitry or harm the user. The components used in this project also had their own set of constraints as to how much current or voltage a component may have been able to receive.

2.5 House of Quality

The house of quality, as shown in Figure 2, is a tool that assisted us in creating relationships between marketing and engineering requirements. Both marketing and engineering requirements are important to keep in mind when presenting the project as a product. Market qualities in particular are properties of a project that would be an interest to a potential user or consumer. Things that might interest a consumer are aspects like device size and portability, wireless transmission, ease of operation and cost.



		Display Frame Rate	Delay to Display	Optical Magnification	Spectrums Imaged	Magnification Settings	Device Weight
		+	-	+	+	+	-
1) Device Size	-	↓	↓	↓	↓	↓	↑↑
2) Wireless Transmission	+	↓	↓↓				
3) Ease of Operation	+		↑			↑↑	↑↑
4) Novelty	+		↑	↑	↑↑	↑	↑
5) Cost	-	↓	↓	↓	↓↓	↓	↓
Targets for Engineering Requirements		>10 FPS	<1 second	>= 40x	>=2 (IR, visible)	>=2	<10 pounds

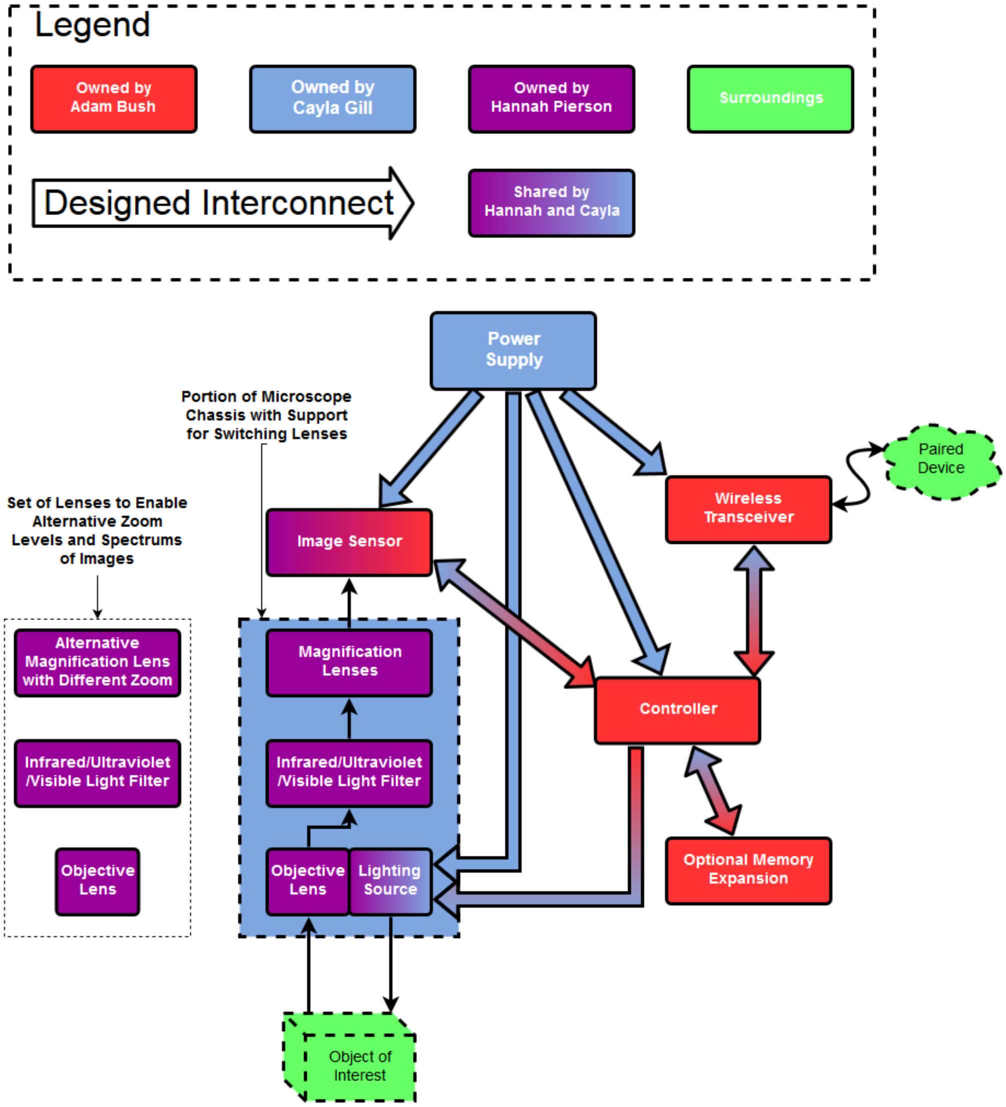
Figure 2 – House of Quality

2.6 System Block Diagrams

Block Diagrams are shown below for both the original and alternate hardware (Figure 3) and software (Figure 4) as a simple introduction to our overall design. A current outline of total and projected component costs and overall project financing is included in a later section. The diagram itself requires little explanation, but the component status is reported in tabulated form below each diagram. These early project outlining efforts have stayed mostly consistent throughout design due to the solved nature of what an imaging system is and a strong commitment to getting off on the right foot – few high-level changes have been forced, with the only real loss being the ultraviolet spectrum, which is a loss to realistic resign, and economic constraints. One high-level change, the removal of filters from our design, was chosen freely out of strong preference and for economical reasons.

2.6.1 Original Hardware and Software Diagrams

This section details the original design in the form of a hardware block diagram. It shows who in our team was responsible for each section. There is also a detailed explanation following the diagram



Legend
Owned by Adam Bush
Owned by Cayla Gill
Owned by Hannah Pierson
Surroundings
Designed Interconnect
Shared by Hannah and Cayla
Power Supply
Portion of Microscope Chassis with Support for Switching Lenses
Paired Device
Wireless Transceiver
Set of Lenses to Enable Alternative Zoom Levels and Spectrums of Images
Image Sensor
Alternative Magnification Lens with Different Zoom
Magnification Lenses
Controller
Infrared/Ultraviolet /Visible Light Filter
Infrared/Ultraviolet /Visible Light Filter
Objective Lens
Objective Lens
Lighting Source
Optional Memory Expansion
Object of Interest

Figure 3 – Hardware Block Diagram Original Design

This figure encapsulates the first imagining of the architectural overview of the microscope in its final form. Extra memory was decided against even though it could extend the capabilities of full-resolution uncompressed image capture, discussed in later sections. The filters were always envisioned as being functional together; that is, we would have three filters, one a band-stop for infrared, one a band-stop for ultraviolet, and one a band-stop for visible light, with the option to use zero, one, or two of them, to capture and combination of those spectrums. Light outside these wavelengths is for all purposes invisible to silicon-based image sensors. The lenses and alternative lenses are self-explanatory in purpose, as are the sensor, controller, transceiver, and power supply. It's notable and accurate that there is no "slide" holding the object of interest; it will be viewed undisturbed by holding the microscope up to it. Component status is shown below in Table 1.

Color in Diagram	Group	Nodes	Status
Red	N/A	Controller	Acquired
		Memory	Obsoleted
		Wireless Transceiver	Acquired
Red + Purple		Image Sensor	Acquired
Purple	Lenses	Objective Lenses	Acquired
		Magnification Lenses	Acquired
		Light Filters	Obsoleted
Purple + Blue	Lighting	Lighting Source	Design / prototyping, with parts acquired
Blue	Power Supply	Power Supply	Design
	Device Chassis	Microscope Chassis	In iterative prototyping

Table 1 – Hardware Block Diagram Component List

The memory was found to be unnecessary after the calculation of timing demands and available speeds of microcontrollers was better understood. The filters have been bypassed thanks to the IR-accepting image sensor, reducing one of our largest costs, which would not have been greatly reduced in a final-version, large-order scheme to the same degree as electrical parts would. In researching the chassis of the microscope, it was determined that the simplest method to bring together multiple lenses, a custom lighting array, and an embedded PCB was custom construction.

The software design presented below in Figure 4 and description thereof in the table below have similarly remained constant.

The high-level software flow starts from the reception of the data at the microcontroller, which must then store and organize that data. Before it's transmitted, the limitations of wireless communications dictate it must be compressed, which is a few-step process, abstracted below into one cohesive unit of operation.

The compressed image is then sent between the wireless link established before any capture takes place. This is a many-step process, as wireless formats typically offer 2 kilobytes of transmission per packet at the top end.

Upon receiving the data, the mobile application will convert the compressed image back to easily manipulated RGB pixel format, performing any necessary or desired processing steps or transformations on the image.

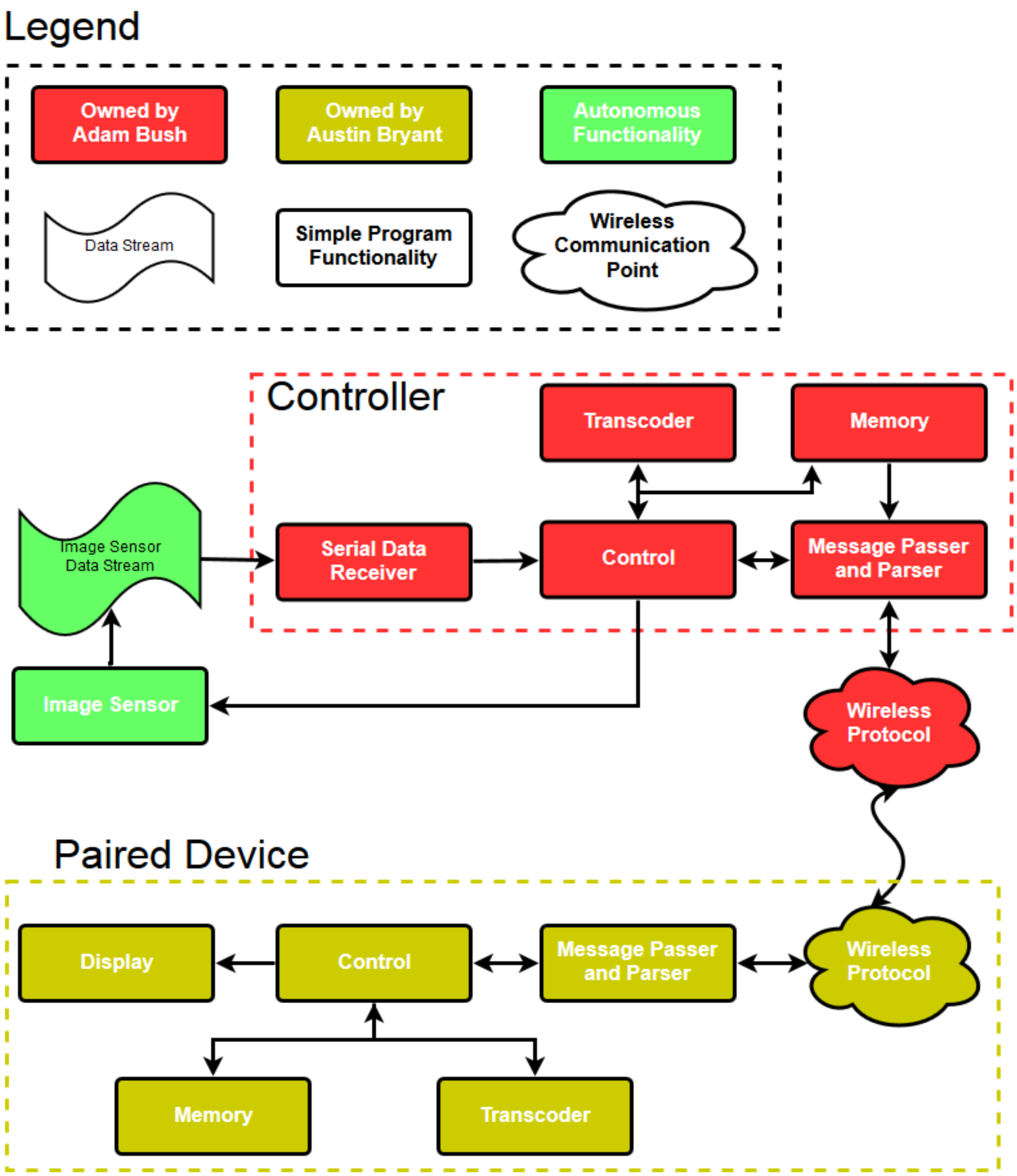


Figure 4 – Software Block Diagram Original Design

Notably, the memory mentioned in the paired device's portion of the block diagram refers to both persistent memory and run-time memory. There will be storage options, both for photos and videos, allowing users to capture and record all they can see with the microscope.

Further, the control node in the microcontroller's portion of the diagram is entirely a slave device of the mobile platform's control node; it will be controlling the operation of the wireless chip, the image sensor, and the first-level image reception and processing, not the overall system flow. Table 2 serves as a summary of the source of these elements.

Color in Diagram	Group	Nodes	Status
Red	Controller	Memory	Obsoleted
		All Others	Acquired
Red	Wireless Chip	Wireless Protocol	Acquired
Gold	Paired Device	Wireless Protocol	Part of device
		Memory, Display	API of device
		All Others	Design

Table 2 – Software Block Diagram Component List

These block diagrams will aid us through the development of this project and serve as a reference for our individual research topics. In addition, these diagrams illustrate how each of our individual components will come together as one finished product.

2.6.2 Alternative Hardware and Software Diagrams

This section covers the alternative hardware and software diagrams. The hardware diagram shows exactly what we used and who was responsible for each section. It begins with a flip of a switch. The system powers up using 7.4 V supplied by the batteries. This is then stepped down by a 5 V regulator to power the Raspberry Pi and the top LEDs. These LEDs were important to provide enough light for the samples the user will be observing. The Raspberry Pi was transmitted power through a USB to micro USB cable. No data was sent through this cable, so it was easy to make a printed circuit board that only needed a power connection and a ground connection. The Raspberry Pi is also connected to an OV5647 sensor that is connected to a 150 mm tube. This tube is connected to a DIN standard objective lens that is situated directly above a base light that illuminates a sample. The image that is on the sensor is then transmitted to an Android device using the built in Wi-Fi module on the Raspberry Pi. This is all shown in Figure 5 below. Each portion is in a different color, as shown in the legend provided. The colors represent the team member who was in charge of each segment, as each of us belong to different colleges at the University of Central Florida. Our team consisted of a Photonics engineer, an Electrical engineer and a Computer Scientist. These individual majors are shown in the diagram below as the different colors and portions of the diagram.

It was important to show the individual portions that each member was responsible for. Luckily, the work for this project was very evenly distributed between the multiple disciplines that were involved. This block diagram also shows how the multiple disciplines were able to work together in order to create a functioning product. This product's different aspects were all able to communicate together and had no problems through the power and the wireless communication that needed to be performed.

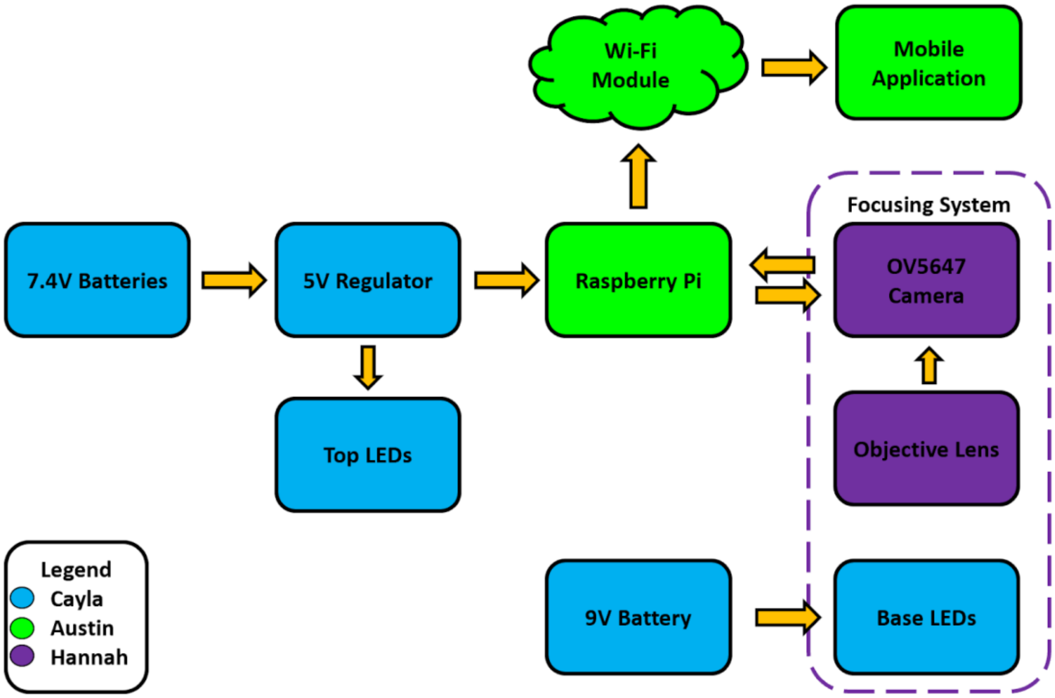


Figure 5 Alternative Hardware Diagram

The software diagram is shown below in Figure 6 and illustrates the control flow for all the code that was used in this project.

When the Raspberry Pi is first turned on, a Python script that is configured to run as a service is started. This Python script interfaces directly with the OV5647 camera module in a continuous loop by using the PiCamera Python module. The PiCamera Python module is a library that allows developers to programmatically interface with any Raspberry Pi camera module and configure the capture process with many customizable parameters, such as the resolution size, camera orientation, and shutter speed.

In addition to interfacing with the camera module, the Python script is also responsible for establishing a socket connection with a client that is currently connected to the wireless access point on the Raspberry Pi. Creating this socket connection only requires an IP address that is obtained from a DHCP server running on the Raspberry Pi, and an arbitrary port number that was selected. From there, the Python script captures an image from the camera module, obtains the size of the image, and sends the size in the form of a byte array, followed by a special character delimiter over the socket connection. Immediately following the transmission of the file size, the script will send the image data in the form of a byte array over the same socket connection.

The mobile application has the integral role of parsing the byte stream that is sent from the Raspberry Pi. To do this, the application will read in data until the special character delimiter is reached and convert the data that was read into its corresponding decimal value. Once number is obtained, it is used to tell the mobile application how many bytes to read following the delimiter. Those bytes are then read and then converted into a drawable image that can be displayed on the phone’s screen.

The secondary role of the mobile application is to allow the user to capture medial files in the form of screenshots or videos and save them to their device’s storage. If the user wishes to have a more structured form of organization when it comes to saving media files to the device, SQLite is there to support them with that. SQLite serves as a self-contained relational database management system in this project that allows users to create custom directories and save file properties to those directories, such as the path of the file. In essence, this allows the user to easily access their captured media files without having to traverse through all the other images on their device. In addition to that, the mobile application also allows users to perform a variety of image processing techniques, such as a gaussian blur, edge detection, and RGB thresholding on the stream of images, which is all done exclusively by the OpenCV SDK.

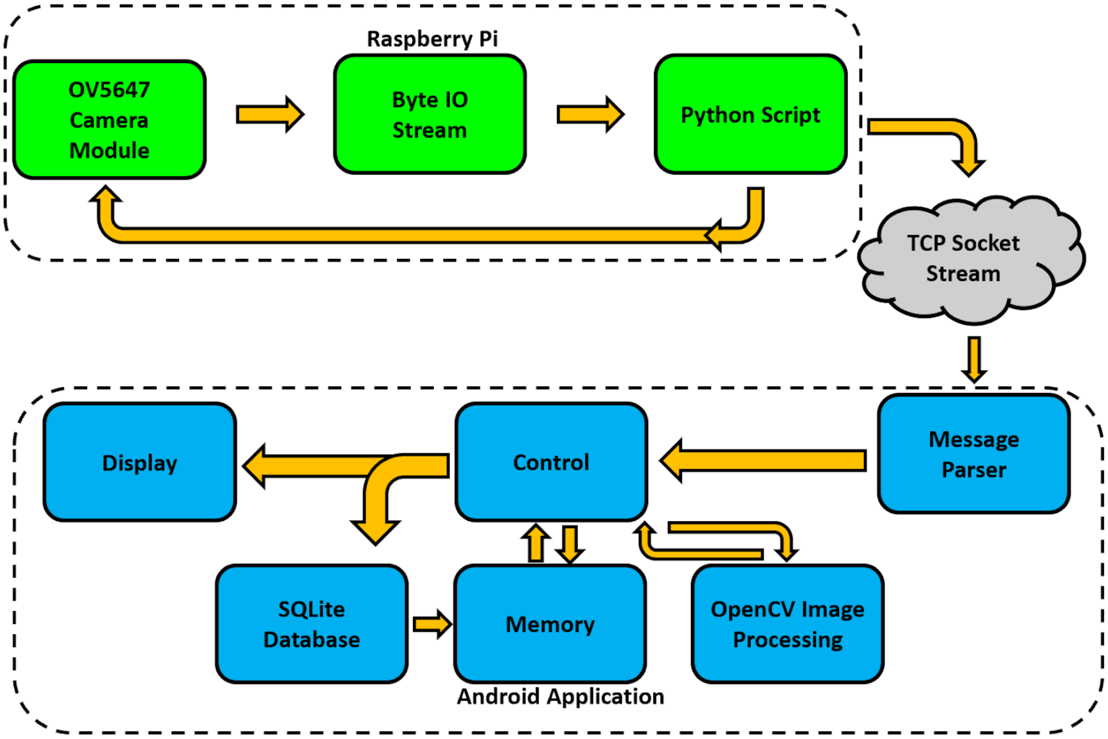


Figure 6 Alternative Software Diagram

3.0 Research

This section is dedicated to the research we did to settle on the idea of building a microscope, and to design the microscope itself. The research done in the section helped us to define the resolution and magnification. It also assisted us in determining how to build the microscope. The first section of this portion of the paper discusses existing microscope and then continues on to explain the difference between magnification and resolution, it explains why it is important to our project to understand the difference between the two. The next section focuses on the actual microscope we purchased and studied, and what we learned from each one. These microscopes helped us to design the body of our microscope.

3.1 Existing Microscopes

To build a microscope, we first had to take a look at what was already on the market. We needed to understand how microscopy works, and how our microscope would be different. We wanted to build a microscope that would be better than the commonly used microscope and address the problems that came with it, such as only one user at a time and refocusing between users. We searched for microscopes that already solved these problems and were able to find a couple of options for fairly cheap. The following section is about how these worked, what we tested them with, and finally what we learned from them.

3.1.1 Resolution vs. Magnification

Before discussing microscopes, it is important to understand the difference between resolution and magnification. Magnification is simply how much bigger something appears where the resolution is the ability to distinguish between two points. The resolution is determined by the numerical aperture of the system, and we will discuss this at length in the resolution section. Magnification is akin to zoom, you can zoom in on something a lot but that does not mean you will be able to resolve the resulting image. Microscopes are built to not only zoom in, but to resolve in great detail so as to study microorganisms and other miniscule objects. We will go further into detail in the following microscopy section and the resolution section.

3.1.2 Microscopes Used for Research

There are just as many types of microscopes as there are types of microscopy. Microscopes can be pocket sized or take up a rather large table depending on how many elements are in the microscope. The elements in these microscopes also are different and can range from small inexpensive lenses, to lenses that can cost many thousands of dollars. The ones that we decided to purchase and research were a pocket microscope and a digital USB microscope.

3.1.2.1 Pocket Microscope

A cheap pocket microscope will only cost about twelve dollars on Amazon, we bought one to test it out and take it apart. It did not have very high resolution, but it still had a system of lenses, variable focus, and a light source. The objective lens was not a system of lenses, so the magnification was very low, and we definitely could not see an object that is 120 µm long and 75 µm wide. The package itself claimed to be sixty to a hundred- and twenty-times magnification. This was definitely not the case after viewing objects during testing. Upon receiving the product, the specifications were discovered in the manual. The real magnification or optical magnification is actually only two. This microscope could not be used for any type of real research, it was made more for a small child playing in the backyard looking at bugs. It was not much better than a magnifying glass.

3.1.2.2 Digital USB Microscope

The next microscope we looked at was a digital microscope that claimed to have two-hundred and fifty-times magnification. It was portable, and it plugged into a computer using the USB drive. We rather liked this microscope because of its lightweight frame and ease of use. Being tethered to a computer was a little annoying because we had to pick up and move one of our laptops around to image things that were not directly in front of us. This was found to not be something we would want in our product. Having a tethered microscope would limit movement and portability. Having these limiting factors would take away from the overall user experience and the vison of this project. The images produced were of good quality and high resolution, but they were not an optical magnification of two hundred and fifty. This microscope had a stand, a work board, a variable light source, a CMOS sensor, and a focusing element. Certain elements of the body of this microscope will be further researched and tested during the development stages of this project. More information is located in section 3.10.

While it left a lot to be desired, it was a very promising demonstration of what we planned to build, and it was a fairly compelling experience to find that a realization of our device with multiple flaws we already planned to address was so usable.

3.1.3 Results of Research

The design of this microscope gave us a few ideas about our own design. We had wanted to create a microscope that is portable and does not need a stand. This microscope came with a stand, but after experimenting with it, we found that the stand was not necessary, due to a guard that was placed on the end. This guard allowed us to push down on the microscope to keep it steady enough to get an image into focus.

Because we want to use more than one objective lens, we were originally going to build a guard that is the same length as the focal length of each lens. These would have been attached to the objective lenses, so a user would be able to swap between them easily, without worrying about the varying focal lengths between the different objective lenses. We settled on utilizing a focusing tube where the stage is stationary but the objective lens moves up and down unlike a traditional microscope. The purpose of this is to move the body to the object at the proper focal length for focus. This will be discussed in the body design of this paper.

This microscope also gave us the idea of battery operated because being tethered to a computer was not convenient. A traditional microscope also plugs into a wall and cannot be moved easily. The components in a traditional microscope are much too sensitive and thus require great care when moving. Our microscope is more robust due to the simple nature of the optics, there are only a few components in it. With our microscope being battery operated and rechargeable, it is much more convenient to the end user. This will be discussed in detail in the power section of this paper.

The last thing that came from studying this microscope, was to make ours wireless. There will still be a program for the images to be viewed but it will not be necessary to have a USB connection. Instead our microscope will be app-enabled so that it can wirelessly connect to an android device. This will be discussed further in the computer science portion of this paper.

3.2 Microscopy

This in an in-depth look at the various facets of microscopy. We will begin by explaining necessary background of the field of study known as microscopy. We will explain how the optics work in a microscope, then we will discuss the aberrations that occur within the optical components of a microscope. Following the aberration discussion, in which the reader will learn what causes the aberration and how to fix it, we will discuss a few different types of microscopy. We will discuss why we settled on the type of microscopy that we will use in our microscope, analyzing Figure 7.

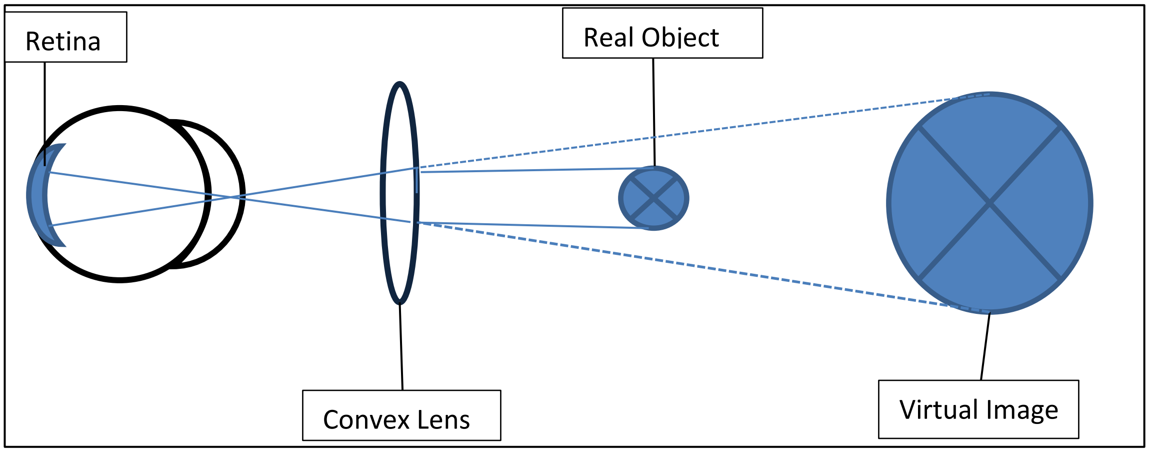


Figure 7 - Ray Diagram
Ray diagram of a specimen being magnified into a virtual image.

Microscopy is the study of things too small to see with the naked eye using optical tools. There are many different types of microscopy that are used depending on what it is you are trying to look at. Microscopes actually create a virtual image that appears bigger than the specimen being studied as shown in Figure 7 above. Virtual images are not tangible and cannot be picked up by a sensor or projected onto a screen.

Since our microscope is not a traditional microscope it does not have an eyepiece but an array of red, blue, and green sensors. In order for the sensors to detect the image, it is necessary to place the camera in the intermediate plane. To get higher magnification, more lenses are required. This technique was developed around the beginning of the 1600s by Galileo and the Janssen brothers. It is the foundation to modern microscopy.

While the very early microscopes were quite impressive, they still encounter fairly low resolution by today's standards. The reason for this was chromatic and spherical aberrations that occurred when the light passed through the lenses.

3.2.1 Chromatic Aberration

Chromatic aberration occurs because the of the refractive indices of the lenses. Due to the wave nature of light, shorter wavelengths are refracted more than longer wavelengths. This means that blue light will be bent more than red light. The reason this bending occurs is directly related to the speed of light. In a vacuum the speed of light is $3 * 10^8 \frac{m}{s}$ or c. The speed of light is related to wavelength by $c = \lambda f$, where f is frequency and λ is wavelength. The velocity of light through a medium is $\frac{c}{n}$ where n is the refractive index. When we put this all together we get $v = \frac{\lambda f}{n}$ and can see that the velocity of light through a medium is directly proportional to the wavelength and inversely proportional to the refractive index. This is important because the higher the refractive index and the smaller the wavelength the more the velocity will be decreased. This means that when blue light passes from air (n≈1) through glass (n≈1.5) it will suddenly slow down causing the light to bend. The shorter wavelengths bend more because the medium they pass through slows them down more. This causes a separation of the image, which in turn creates blurring.

3.2.1.1 Correcting Chromatic Aberration

In the 1700's a lens maker named Chester Moore discovered that when he combined one concave lens made of flint glass and one convex lens made of crown glass that he could reduce the chromatic aberrations. All objective lenses today are corrected for at least two colors. Some objective lenses can correct for up to five colors, however these are highly specialized lenses made of very specific materials. This means they are quite expensive, anywhere from a few hundred dollars to thousands of dollars. Because of their cost our team is using simple achromatic lenses which only correct for blue and red wavelengths. If it had been within our price range, we would have gotten a plan fluorite objective lens so that we could image ultraviolet, since we could not we are limited to the visible spectrum and infrared. All curved lenses have spherical aberration, it is unavoidable.

3.2.2 Spherical Aberration

The spherical aberration occurs because the rays traveling through the center of the lens travel faster and are bent less than the rays at the edge of the lens. This can cause either the image to become highly focused at the center and hazy at the edges. This type of aberration can be reduced by utilizing an aperture to limit the number of rays that come through the outer edges of the lens. This method also limits the amount of light within the system, so it is not always effective. Manufacturers typically solve the problem of spherical aberration by using glass elements that have been cemented together. This system of lenses with varying convexity and concavity ensure that all of the rays that pass through the system come to the same focal point.

3.2.3 Types of Microscopy

There is a plethora of types of microscopy such as: darkfield microscopy, differential interference contrast (DIC) microscopy, phase contrast microscopy, fluorescent microscopy etc. Brightfield is the most common and well-known type of microscopy, it illuminates the whole specimen and the rays that are reflected back into the microscope are gathered and focused into an image by the objective lens. There is also darkfield microscopy that works a lot like brightfield except that instead of having an unobstructed light source, it obstructs most of the light using a small ring that is dark in the center. It only allows a fraction of the light to get in. DIC microscopy is fascinating because it uses a Nomarski prism to create an optical path length difference. The result is that the optical path length difference is transformed into an amplitude difference in the waves. This creates a very high contrast image where the smaller amplitude differences are brighter, and the larger amplitudes are darker. Our microscope utilizes brightfield microscopy and is optically reminiscent of the early microscopes that used multiple lenses in conjunction to produce a clear magnified image. The reason for choosing brightfield is because it is a simpler design with only a few lenses that lends itself well to being portable and smaller. It is also much cheaper because there are not as many optical components, such as prisms, polarizers, and filters.

3.2.4 Conjugate Planes

Conjugate planes are the basis of forming an image, and it is crucial to understand how these work in order to build a microscope. This section will go in detail to why they are so imperative, and how they work. We will then explain how this applies to our project and show the calculations for how we decided on the lenses to form our conjugate planes within the microscope.

Conjugate planes use a system of lenses that are place at an exact focal plane to create magnification in perfect focus. The first lens is placed in the system, with an object placed at the front focal plane. Another lens is placed behind that lens with its front focal plane at the back focal point of the first lens. This will put the real image created by the first lens at the exact focal point of the back lens which will create an image of the object. This will create conjugate planes, and create an image that is magnified based on the total power of both lenses instead of just one lens. The magnification can be greater than one which creates a larger image, or it can be less than one with will create a smaller image. This is based on the lenses and if they are convex or concave. This system of lenses is what is used in created all types of optical systems.

A microscope uses this concept of conjugate planes to magnify an object at a high resolution. The objective lens itself is one set of lenses, however we will consider it just one lens for the sake of simplicity. The lens that comes after the microscope needs to be placed at the conjugate plane, i.e. the back focal point of the objective lens, so that the image can be magnified. The back focal plane of the objective lens is generally the tube length, so we need to place our second lens at this point. This should create a magnified image for our sensor to detect. If everything is perfectly set up we will actually increase the magnification of the objective lens, the total system will have a total magnification that will combine the magnification of the objective lens and the additional lens.

3.3 Image Sensors

At a very low level, modern image sensors operate using a Bayer filter, a repeating pattern of red-, green-, and blue-sensitive photosensors. Each pixel is formed of analog intensity readings of the photosensors.

The moment of sampling varies between devices, with each using either a rolling or global shutter. The rolling shutter samples from one area at a time, while the global shutter exposes all areas of the sensor simultaneously. The former can lead to issues with moving elements in a photo, while the latter is more prone to blur. The circuitry required to read global shutter device is also more involved, as far more analog readings must be made, and often stored, at once, increasing costs.

3.3.1 CMOS Image Sensors

CMOS image sensors are widely used due to their ideal sensitivity to the visible spectrum, their closeness to existing CMOS fabrication, and their energy efficiency; the second allowing a lower price point, which is crucial to our project, and the third supporting our target of a battery-powered microscope.

Their sensitivity extends beyond the visible spectrum, however, and require the use of bandpass filter or a pair of band-stop filters to isolate the visible spectrum from any ultraviolet and infrared light. For our purposes, this correction leads to the simple subversion of utilizing CMOS sensors with an infrared bandpass filter to capture that spectrum of light without abandoning the advantages of CMOS-based designs.

Existing sensor designs support this to a very fine level of detail: each pixel, formerly composed of many samples of red, green, and blue, exchanges some of each sample with an infrared bandpass filtered photosensor. This allows imaging in low-light settings and general infrared imaging.

3.3.2 Charge-Coupled Devices

Charge-coupled devices use the global shutter method and require a relatively larger size to sense infrared light due to their physical mode of operation. These in addition to the method's power draw make it an unappealing choice for our microscope.

3.3.3 Image Sensing Schemes

CMOS was the early and obvious choice, so product selection began with attention focused toward infrared sensitivity, pixel size, resolution, output format, power draw, and cost.

Initial plans involved utilizing a sensor which didn't filter out infrared light at the photosensor level, then physically managing filters in the microscope body to either produce an infrared bandpass or a visible light bandpass effect. These relied on an approximately 20% CMOS efficiency at detecting near-infrared light and would require a considerable source of light.

Due to the cost of external filters, the required physical manipulation, and the suboptimal sampling efficiency, use of these sensors was not desired. Then, two families of image sensors, an Olympus and an ON Semiconductor, were found to natively support infrared sensing in some capacity.

ON Semiconductor's AR023x line of RGB-IR image sensors was chosen as it supports all primary use cases sought by the device requirements, as well as a significant complementary use cases: simultaneous RGB plus infrared imaging.

3.4 Image Processing Approaches

At the receiving (phone application) end, there is ample processing power, memory space, and software support to run image processing algorithms to improve the perceived quality or detail of an image without adding disruptive delay.

3.4.1 Image Processing Techniques

There are thousands of possible image processing techniques that can be performed on various types of media. For size constraint reasons as well as feasibility, we did not include every possible image processing technique in this project. Instead of trying to include all the different possible image processing techniques, we decided to incorporate algorithms that touch on each of the main categories of image processing: edge detection, sharpening, and smoothing. Listed below in Table 3 are some of the image processing techniques that we have chosen for this project.

Image Processing Techniques	Description of Techniques
Contrast adjustment	Adjusting the amount of sharp differences there are between black and white in an image
Gaussian Blur	A blurring effect that is often used to smooth images and remove noise
RGB to Binary (B&W)	Converts the RGB image to a B&W one
Edge Detection	Accentuates the edges in a given image to make them more visible
Cropping	Cutting a section out of an image to produce an entirely new image
Identification Labels	Ability to label certain features in a given image
Sharpening	Improves quality by smoothing edges and then enhancing them
Color Balancing	Altering the color composition of an image to improve its presentation and pull out additional aspects or levels of detail

Table 3 – Image Processing Techniques

Additionally, we also wanted to make sure that we could record clips from the stream and take screenshots when desired. The final mobile application has buttons that allows the user to start and stop recording video, start a rapid image capture, and take a single still image.

All these image processing techniques are currently available in the OpenCV SDK and have ample documentation and references that can be used during the development process. Most of these techniques are considered “nice-to-haves” and are completely optional and will only be used when the user elects to use them. The idea behind this was for all these image processing techniques to be stackable. However, an issue with this is that many of these algorithms can be very computationally expensive to run, so stacking them on top of each other could potentially slow down the overall performance of the application and decrease the framerate. The decision to keep this in was heavily considered but we decided to leave it in the application as the decrease in framerate turned out to be negligible.

For techniques such as the Gaussian Blur, it's also possible to fine-tune some of the parameters, such as the sigma value, kernel size, and pixel interpolation method. Because of this, some of these techniques allow the user to select relevant parameters, adding more granularity to the image modification process.

To record video, a sequence of screenshots is taken and are stitched together using an MP4 encoder. While the recording is happening, a red line is shown around the perimeter of the stream so that the user is aware that the recording is happening. Upon completion, the red border will disappear, and the resulting video is stored to the user's local storage on the phone. Taking a still image of the stream requires the same process, but instead of recording multiple frames, we only grab a single frame.

3.5 Microcontrollers

The microcontroller will, with the aid of an embedded Wi-Fi chip, receive at a minimum rate of 20MHz (for video, at VGA resolutions, due to memory constraints - 640x480, 10 frames per seconds, though the still image size is more flexible), it will convert the raw RGB image to the JPEG format to allow Wi-Fi to provide a sufficient transmission bandwidth for live video, and it will manage control signals to and from the phone application and to the image sensor.

The most computationally-difficult aspect is the JPEG encoding, which requires an initial color transform, multiple multiplication and additions, averaging, a discrete cosine transform, and quantization. The latter portion is eased with the storage of results in a lookup table. Altogether, the former requires a considerable amount of CPU time.

Memory management is another significant issue, with pixel data coming in within two orders of magnitude of a standard embedded processor's clock. In addition, the converted image must be forwarded to the Wi-Fi chip.

General requirements from the above can be summarized as: Capable of efficient JPEG encoding, capable of managing memory in fault-resistant manner, and capable of managing the outgoing high bitrate data transfer to the Wi-Fi chip.

3.5.1 A Dual-Core Approach

A master-slave architecture, with one core dedicated to management of I/O, control signals, and directing the other core, which will in turn perform all mathematical operations, was strongly considered. Multiple Arm embedded processors would be suited to such a task, and the programming would follow easily, with likely support of C libraries for JPEG conversion.

The issues which prevented this approach from progressing are threefold. The system would have a relatively high power draw, using two general-purpose processors to perform very repetitive, computationally-demanding and memory-intensive tasks, which would reduce battery life substantially; the architecture puts a high demand upon perfectly precise programming, which is a difficult drawback to accept in an embedded device tied to receiving all control and debugging through an external phone application; and the avenues of failure increase dramatically. The two cores would require tightly-coupled shared memory, which is a figurative minefield of conflicts.

3.5.2 An Embedded JPEG Codec

An extremely task-appropriate microprocessor was found – namely, the STMicroelectronics STM32H7xx family. This chip features direct memory access (DMA) with support for queued (first in first out) and DMA output to a Wi-Fi chip through a generic (USB, UART, etc.) serial interface. It also supports 12-bit parallel camera formatted data through DMA. Most significantly, it has an on-chip processor-independent JPEC codec and sufficient tightly-coupled memory. It is also based on the Arm family of microcontrollers, namely the lower-power Cortex-M7, but the independent, ASIC JPEG handling is its primary indicator for our project.

3.6 Database Management on Android

There are many options to choose from when selecting a database for a software application. For this application, we required a relational database management system that did not have much overhead and had a familiar query syntax. The database in question did not need to be remotely hosted and could instead be run locally on the mobile device, which is how we ended up deciding to move forward with SQLite.

3.6.1 SQLite

For the database for this app, we have decided on SQLite. The reason for this is that SQLite runs off a local device, which is preferred because a remote database is not needed for this application since there will only be a one-to-one connection between the microscope and the mobile device. If in the future the design changes to support a one-to-many type of connection, then a remote database would be needed for storing the metadata of the media files. By keeping the database local to the device, we do not have to worry about maintaining an active internet connection and the additional overhead that comes with querying a remote database.

SQLite is also generally the database of choice when SharedPreferences are not an option and persistent storage is needed for Android applications. Databases are also preferred when you must store a lot of similar information that would otherwise quickly exceed the limitations of SharedPreferences. SQLite is also incredibly lightweight, supports all the relational database features and has much better performance than traditional read/write strategies. In addition to that, Android comes with a built-in SQLite database implementation, which maked integrating a database into the app a seamless task.

To implement a SQLite database into an Android application, all that is required is that the developer must include the *android.database.sqlite* package in the source code and then some minor modifications need to be made to the Android manifest file.

For this project, since we will used Android API level 14, we ended up using SQLite version 3.7. As anticipated, there weren’t any limitations imposed by the SQLite version as the mobile application just be executes basic create, read, update, and delete queries and didn’t require any special queries that were introduced in later versions of SQLite.

SQLite is also incredibly popular and is widely used across the world. As such, there are numerous resources online that can be referenced throughout the duration of this project in case any issues arise getting it properly set up. Additionally, the app developer for this project has a lot of experience with relational database management systems, so implementing this step of the project was relatively painless.

3.7 Communications Methods

Interfacing the mobile application and the embedded system will require significant bandwidth, and both the medium and quality of service assurances require consideration. The target is roughly 1.5 megabytes/s.

3.7.1 Choosing Wi-Fi over NFC and Bluetooth

Near-Field Communications and Bluetooth both present as potentially-distance-appropriate solutions to our communications needs, but neither offer the bandwidth necessary for a video application.

NFC targets an upper limit of 400 kilobits per second, which is woefully insufficient, and the range would not support broadcasting, which is a desired feature expansion for a production-ready version of our device.

Bluetooth falls just short of 1 megabit per second, which would support a roughly one frame per second rate under ideal conditions.

3.7.2 Transmission Control Protocol

The transmission control protocol is a connection-oriented protocol, meaning that a connection is maintained until both parties are finished with the process of exchanging messages. As such, the transmission control protocol is used to guarantee the transmission of messages. This is done by using a process known as a three-way handshake. The way that a three-way handshake works is as follows:

- A client node sends a SYN packet over a network to a server either on the same or external network. This process is used to ask if the server is open for new connections.
- The server must have open ports before it can accept and initiate a new connection. When the server receives the SYN packet from the client, it will respond and return a confirmation known as a SYN-ACK message.
- Once the client receives the SYN-ACK message from the server, the client will then send an ACK message over to the server and then a TCP socket connection is established.

The transmission control protocol is known as a reliable protocol for this reason. This three-way handshake process occurs during the establishment and destruction of a socket connection and is a way for users to ensure that their data is transmitted without the loss of any packets. A diagram of the three-way handshake process can be seen in Figure 8 below.

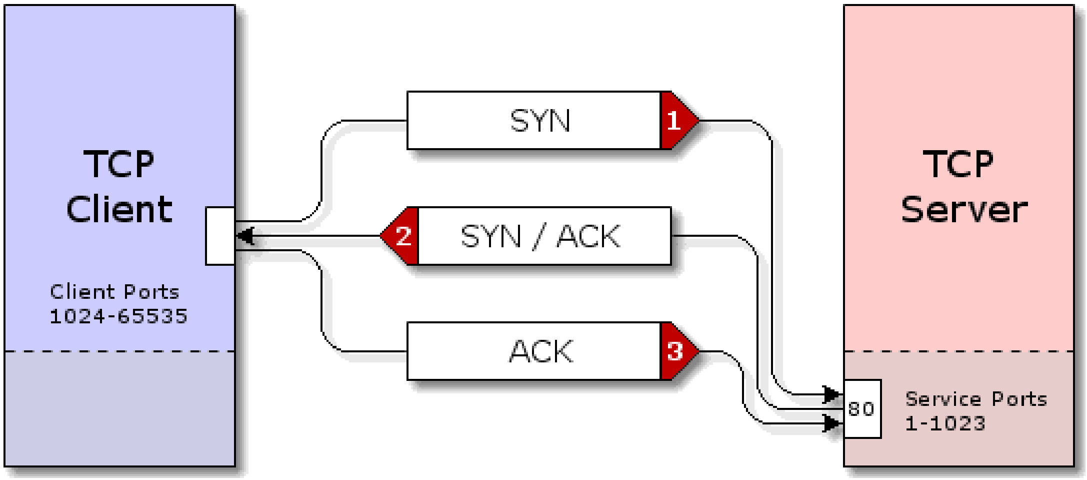


Figure 8 - Three-Way Handshake

Since this connection is considered reliable, it would seem like the transmission control protocol should always be used as the data transmission protocol of choice. However, since the transmission of packets is guaranteed, this means that TCP will ensure that everything is delivered, regardless of whether every individual packet is truly needed. This is cumbersome during video transmission, which is why TCP’s counterpart, UDP is generally used for handling video and voice communication. If TCP was used for voice communication, this means that every packet must be delivered to each end user, regardless of what happens. This is problematic during video or voice communication because if a packet fails to be delivered a few times in a row, TCP will continue to try and transmit the packet until it is successfully sent. This sounds like a good thing, but it will inevitably cause a delay (commonly referred to as lag) between the client and the server. This is something that can be incredibly annoying for an end user to deal with, so while TCP is great for ensuring that data will get to the end user, it’s not the best protocol choice for every type of communication.

Initially, we were planning on using UDP as opposed to TCP due to the acclaimed difference in speed, however we found the difference in speed to be negligible. As such, we decided to go with the more reliable network protocol for this project, TCP.

3.7.3 User Datagram Protocol

The transmission control protocol’s counterpart, the user datagram protocol, is commonly referred to as a connectionless, or unreliable protocol. The term unreliable is a bit deceptive though and leads people to believe that it could never be used as an effective communication protocol, which certainly isn’t the case.

The primary difference between TCP and UDP is that UDP doesn’t use a three-way handshake like TCP does. Instead of focusing on the handshake, UDP skips that step and focuses purely on the transmission of data. As a result, UDP has lower overhead and is significantly faster than TCP, making it the ideal choice for video and voice communication. The increased speed does come with a drawback though. Since UDP doesn’t focus on ensuring that the data is properly sent or received, if something goes wrong during the transmission process then the packet will be dropped and won’t be recovered.

While this might sound bad, the UDP protocol is the ideal choice for voice and video communication because of this feature. For example, during a video call, so many frames are being sent at a given time that the user will likely not even notice if a couple frames get dropped. Even if the user does notice the drop in these frames, it’s been shown that end users much prefer frames to get dropped then for their streams or conversations to be delayed by a few seconds because they were waiting for the TCP handshake process to complete.

As we can see, the choice in protocol is incredibly important when transmitting data. If there was vital information contained in any transmitted packets that the end user needed to see, and the packets were just dropped for some reason, the users would not be happy. For that reason, if transmission needs to be guaranteed, TCP is a good choice for a protocol. Otherwise, UDP is generally the better option.

3.8 Printed Circuit Board

Printed circuit boards are the most common assembly method of electronics today. PCBs give the designer the ability to connect various components internally in the board instead of connecting individual pins with wires. Paths in the conductive portion of the board act as physical wires would. By designing these boards and having them as one solid and durable structure, these boards eliminate the fear of easily destroying circuits, due to their strong structural nature, as well as short circuits and incorrect wiring.

3.8.1 Chip Package Designs

When researching the image sensor, microcontroller, Wi-Fi chip, we ran into the decision between a type of Ball Grid Array (BGA) connection for the printed circuit board, or a Quad Flat Pack (QFP) connection. These two connections are very different, both in connection, soldering technique, and design. When constructing the PCB, all of these factors needed to be taken into account. Since this is a prototype project and the concept of PCB design and construction is a newer concept for us, the BGA, also known as a ball grid array, seemed to pose problems for our project.

Ball grid arrays are beneficial for microcontrollers and other integrated circuits that require more than one hundred pins. These types of integrated circuits utilize the back of the outer casing to make the connections to the circuit board. There are no exterior pins, only small circular connections onto the board. These connections permanently mount the integrated circuit to the printed circuit board. A problem with this array of connections is, since they are on the back of the casing, it is very difficult to test the connections to be sure they are connected properly. If they are not connected properly and the chip malfunctions, the only proper way to check it is with more advanced technology that may need to use X-rays in order to properly inspect them. This posed a problem for us, as we did not have access to technology for this. Soldering also was a problem since each pin needed soldering and we would need to take the board to a company that specializes in soldering these types of chips, or else solder it blindly. Though these are good for many projects as far as pin number and efficiency may be concerned, this type of layout was not good for what we needed the chip to do.

For the QFP, Quad Flat Pack, layout, there are perimeter pins, which are connected to the printed circuit board. This layout is beneficial for integrated circuits that do not need more than one hundred and fifty pins. Most integrated circuits have between one hundred and one hundred and forty-four pins around the perimeter. These chips are soldered to the surface of the printed circuit board with the pins that surround the casing. Using this method, the pins are much more easily tested to ensure the connections to the circuit board are correct and secure. The connections in this layout are not hidden or hard to obtain like those in the ball grid arrays.

Low-profile Quad Flat Packs, or LQFPs, are beneficial for their small size and ability to be surface mounted to the printed circuit board. These have exterior pins as other QFPs have and therefore will be easier for our particular project. With this research in mind, the pins would be easier for us to test and able to design. This type of chip layout will be consistent with our image sensor, microcontroller and Wi-Fi chip.

When researching our specific chips for our circuit board, we were able to obtain two the four chips in a surface mount form. These chips, for the most part, can be mounted on the board without the assistance of a soldering company. However, the amount of pins on the microcontroller will make it difficult for an amateur solder technician to connect properly to the printed circuit board. As far as our preferred image sensor and the Wi-Fi chip selection, the package only came in the ball grid array form. Although this was a setback for our project, we were able to take it to a local company that was able to help us properly solder each of these components' pins in the correct location on the printed circuit board.

3.8.2 Layers of Printed Circuit Board

When first designing the printed circuit board, it was important to determine the thickness that would be appropriate for our board. The PCB can be made with many layers of thickness. This thickness depends on what the board needs, the components on the board and what the purpose of the board is. Before determining what the layers we needed were, we first needed to understand the materials and purposes for each layer.

We started by researching the purpose of the inside of the board, which consists of a base material, usually fiberglass. This layer is essential, for it gives the circuit board a thick nature and provides rigidity. Though there are flexible PCBs, they are not as common as the harder plastic structures. There are cheaper boards available made with different plastics and the thicknesses can vary. These plastics make it difficult when soldering because of the fumes that are given off. The more durable fiberglass and other thermoplastics are able to withstand the heat of the soldering iron. Other plastics, such as phenolics, may burn, smoke and char when the soldering iron is left on it for too long.

After the plastic layer comes the copper foil, the conductive portion of the board. Either one or both sides of the plastic layer can be coated, depending on the layers the board has. A two-layer board is also known as a double-sided board. These boards have two layers of copper, one on each side of the board. The amount of copper on the board can vary, but the majority of PCBs have one ounce of copper per square foot, which means about thirty-five micrometers of copper thickness.

After the copper layers lays a soldermask layer. This soldermask is the green tint, which is commonly seen with printed circuit boards. This layer is used to insulate the copper pathways so that there are no incorrect or accidental connections. Metal portions are visible over the insulation so connections can be made through soldering components to the board.

Finally, a silkscreen is applied on the surface of the soldermask layer. Silkscreens add labels such as letters, symbols, and numbers for easier assembly by the designer. These symbols are necessary for the construction of the PCB. If certain components have specific polarities or pin locations, this silkscreen indicates each components individual needs. The designer can assemble the components in the proper way, so no unnecessary problems arise because of soldering mistakes.

3.8.3 Circuitry Design Process

When first considering how the design would come to fruition, it was important to determine the design process that would need to take place. In order to have a successful printed circuit board design, the first process that needed to be done was the circuitry design of the system outside of a program. By hand designing, we saw how the proper layout would have to be. From here, we needed to see how the layout design performed by creating breadboard tests. These breadboard tests were imperative to be sure the voltage and current to each section of the circuit is correct. If these values are not correct, the paper design can be altered and the design can be tested again before being solidified as the permanent design.

After creating a board layout and testing on a breadboard, we would need to create a schematic design, decide upon the layout of the board, and then create the printed circuit board design. Using this process, the creation of the board will be successful. If this was done in a different order than this, the circuit may end up being incorrect and previous work would need to be discarded. This would lead to wasted work, wasted time, frustration, and maybe even wasted parts, if components were destroyed in the process.

In this paper, each of these sections will be present for each system that needs designing. For this specific project, there will be the designs of the voltage regulator, the Wi-Fi chip, the image sensor, the microcontroller as well as the LED lighting system. Each of these systems will be designed and tested as well as having printed circuit board designs present. These will be seen in later sections.

3.8.4 Eagle Program

When creating the necessary printed circuit boards, Eagle showed to be extremely helpful in designing the necessary chip designs that were required. However, the printed circuit board design was not consistent and posed many problems when working. Most of the controls were not natural for the user and had to take extra steps to group parts and move connections. Instead, a browser-based program was used. This program was called Easy EDA. This program was very user friendly, included a connection check complete with DRC check that ensured no traces exceeded the necessary spacing and that all connections were correct. The Gerber files were easy to create and the website linked to JLC PCB, which was the company we chose to order our completed printed circuit boards from. This company was very efficient with production and within a week we received our designs. These designs were then taken to Quality Manufacturing Services, which soldered the proper components to our board. This company was very helpful in soldering components we were unable to connect to our board and gave us the designs in a very timely manner.

3.9 Power Supply Options

During the conceptual phase of this design, we considered the power source of the device. It was determined that the microscope needed to be both portable and wireless. Knowing this, we eliminated the option of a wall socket. This type of design would have a wall plug with either two or three prongs, known as Type A and Type B, respectively. Type A would be an ungrounded plug and Type B would be a grounded plug, with three prongs. Both types are used commonly in the United States and give 15 amps of alternating current, as well as provide voltage ranging from 100 to 127 volts. This alternating current would not necessarily be a problem in creating the device and the safety requirements needed to protect the circuits within, however, the cable is highly inconvenient for the purposes of this project.

The main concern with this device was its portability. The wires and cables that would need to be inserted into a wall socket make the device less capable of being moved around easily and caused problems that can be solved easily by powering with a battery source. This power option was feasible, but not convenient. A main focus for this project was its convenience and portability, therefore we moved onto a more suitable power element.

The next logical step was to search for a capable battery for this device. However, before determining which battery would be the most effective, we first had to determine what kind of device our system belonged, in terms of power consumption.

There are various classifications for these devices and they are categorized by how much energy the system drains from the battery. This helped in determining the correct battery to use depending on how long the battery life would have to be. High-drain devices, such as digital cameras or global positioning systems (GPS), have prolonged and continuous usage. Low-drain devices, such as LED flashlights or remote controls, use much less power instantaneously and slowly drain power over time. There are moderate-drain devices that work between the two draining points. Knowing the drainage of the system helped in determining the type of battery the system will need.

3.9.1 Batteries

A focus for this project was for the device to be portable and handheld. In order for this to be possible, we needed to have a battery-powered system. Batteries are best when powering low-power systems, which is what we designed. However, there are many different types of batteries,2 and each is beneficial in their own way. It is important to recognize the types of batteries and their different qualities in order to find the best option for our power system.

3.9.1.1 Single-Use Alkaline Batteries

The difference in the system drainage will affect the type of battery chosen. When first considering batteries, alkaline batteries were one of the initial ideas. These are readily available and are very common in many electrical systems, especially those around the house. Alkaline batteries are actually the most commonly used batteries. The name "alkaline" alludes to the fact that these batteries contain an alkaline electrolyte. The most common electrolyte used is potassium hydroxide, which is commonly referred to as "battery acid." These batteries are best used for low-drain devices. They can be used for high-drain devices; however, the life of the battery is significantly shortened. Even though alkaline have a high initial engird capacity, the excessive quick and endless drain from high-drain systems significantly drain the energy of the batteries.

As with all things, there are positives and negatives about alkaline batteries. A few positives is that they are not very expensive and are readily available almost anywhere. A negative is that they can sometimes be recycled, but instead are often disposed of in landfills. These batteries have a nominal voltage of 1.5 volts, though the voltage discharge decreases over time to 1 volt. Though the voltage drops over time, these batteries have a constant capacity over a wide range of drain currents.

Alkaline batteries have a good shelf life and have better temperature and leakage performance when compared to commuting cells. They are suitable for many consumer applications as they are offered in many different sizes for varying purposes. One downside is, though they are decently priced, there are other cells that are cheaper.

3.9.1.2 Rechargeable Alkaline Batteries

Through the design process, we discussed the single-use and rechargeable versions of alkaline batteries. Rechargeable alkaline batteries are a positive choice. Even though they may cost more in the initial purchase, they can be used over and over again. An alkaline battery, that is rechargeable, is the Nickel-Metal Hydride battery. This battery provides a better long-term value as opposed to single use batteries.

For a standard Nickel-Metal Hydride battery, nickel hydroxide is used as the positive electrode. An alloy, a mixture of metals, or a mixture of metals and other elements, is used as the negative portion of the battery. Potassium hydroxide is commonly used as an electrolyte. These types of batteries are one of the most commonly used, when recharging is necessary. They are beneficial because they offer high-energy capacity, and are efficient for high-energy devices like digital cameras. However, they do not work well for low drain devices, like some detectors and other items not frequently used, or those that are stored over long periods of time.

Nickel-Metal Hydride batteries are known for their capability for delivering energy at a constant rate, which is referred to as a flat discharge rate. While the single-use alkaline batteries will decrease voltage output over time, Nickel-Metal Hydride batteries will not dim as the energy discharge lessens. They are also known to deliver substantially more current than normal alkaline batteries, which makes them ideal for high-drain devices. The performance is more ideal in low temperatures, has better long term values, and are recyclable.

However, these batteries also have negative aspects associated with them. Though they replaced the Nickel-Cadmium battery as the most popular rechargeable battery, it is not as durable. They have a twenty percent discharge in the first twenty-four hours after charge and a ten percent discharge every month thereafter. By modifying the hydride in the battery, the discharge can be lowered and reduces corrosion, but this also decreases energy.

These batteries need to be charged before use and should be charged every month. Unfortunately, they are delicate as well, as performance may be affected if roughly handled or dropped. This was not a bad option for the device's power source, but other options were pursued as well.

3.9.1.3 Single-Use Lithium Batteries

From here, we discussed the option of the ever-popular lithium battery, which has taken the industry by storm. Both single-use lithium batteries and rechargeable lithium batteries are available to us. These batteries are very popular due to their high power outputs and versatility. They can work both for low power systems, when regulated correctly, or high power systems.

First, we sought out the single-use lithium batteries. Lithium batteries are known to give the highest energy density of any battery cell and can store more energy when compared to its alkaline counterpart. Lithium batteries are good in extreme temperatures as well. The temperature portion of this product does not affect our choice of the battery, however. We did not plan on placing our product in extreme temperatures. The microscope we created would be mostly used indoors and at room temperature.

Lithium batteries can be dangerous to circuitry because of the higher voltage capacity. These are best for moderate and high drain devices and have the best lifespan by far in the single-use category. Lithium batteries also have a very long shelf life and are lightweight. Though they are more expensive, they make up for their cost when compared to their lifespan, which can reach up to five times that of single-use alkaline batteries. Lithium batteries contain a high current output as well, which is also beneficial for many types of systems.

When regulated correctly, a large current output can extend the power dissipated over time. Alkaline batteries do not contain as much current output as Lithium batteries, and therefore the power output and lifespan in a system is shorter. More comparisons can be made between Lithium and Alkaline batteries, and a few of those are shown below in Table 4.

	Alkaline	Lithium
Nominal Voltage	1.5	3.0
Estimated Shelf Life	5 – 10 Years	10 – 15 Years
Distinctive Characteristics	- most commonly used and widely available battery	- Superior Performance in extreme temperatures - Lighter weight than alkaline batteries - Could be too powerful for some devices

Table 4 - Comparison of Single-Use Batteries

3.9.1.4 Rechargeable Lithium Batteries

Since Lithium-ion is such a reliable and powerful battery, it only made sense for us to use this for our project. The fact that there are rechargeable lithium ion batteries were even better. This rechargeable aspect benefited the project even more. Some notes about these batteries are that they are more expensive than the single-use batteries, but since they are rechargeable they make up for the price with their lifespan. They are mainly used in notebook computers, smartphones, portable power devices and even bike lights. These devices prove even more that Lithium Ion's lightweight nature is beneficial for the portability of our device.

The voltage is usually set at 3.6 and has an estimated 500 to 1000 recharging cycles. The self-discharge rate is very low, at less than 2 percent every month. This discharge rate is at the lowest of current rechargeable technology.

These batteries are recommended to charge regularly, as using the battery to exhaustion may cause harm to the battery. The battery is also recommended to charge more frequently in order to increase the lifespan of the battery. More frequent charges for a battery with more stored power (at least 50 percent) can help with increasing the charging cycles for the battery.

To show how much these batteries are more suited for this project, a chart has been supplied, comparing the standard Nickel Metal Hydride (NiMH), the pre-charged NiMH batteries with the Lithium Ion batteries. Table 5, shown below, demonstrates this.

The Lithium-ion battery outperforms the others across the board, and the few downsides are accepted for the greatly increased efficiency and reduction in leakage current, given the device is likely to be left unattended for long periods of time in the hands of a typical user, especially in a classroom environment.

	Standard NiMH	Pre-charged NiMH	Lithium-Ion
Nominal Voltage	1.2	1.2	3.6
Recharging Cycles	150 – 500	150 – 500+	300 – 500+
Average Rate of Self-Discharge	1% per day (poor)	20% per 6 months (very good)	Less than 2% per month (Excellent)
Characteristics	- Must be charged before use - When used intensely used over time, it tops the pre-charged due to higher energy capacity	- Ready to use from package - Trumps the standard when used over prolonged time due to slow self-discharge rate	- Must be charged before use - High performance - Limited to specific products -Diminished by age apart from use

Table 5 - Comparing Lithium-Ion to Other Rechargeable Batteries

3.9.1.5 Battery Design Choice

An important aspect that we discussed is the ease of the consumer and the simplest way the user could handle this device. We discussed the battery types that were researched and decided that the best option would be to have a rechargeable battery, also known as a secondary battery. Users are now very accustomed to the idea of rechargeable systems due to the popularity of smart devices. Laptops, tablets and phones, as well as smaller devices are known to need recharging. This would make the device portable, have a prolonged battery life, and provide less work for the user.

Ideally, we did not want the users having access to the internal workings of the device. With its delicate interior full of electronics and fragile lenses, there would pose many problems that may occur with access to the internal components. However, after looking into this more in depth, we found that creating a rechargeable battery pack within the system would pose many problems for the electrical portion of this project. The biggest concern was having the time to perform these duties.

After further research on the battery that is needed for our project, we decided it would be best to recharge the batteries outside of the system. We wanted to focus on rechargeable lithium ion batteries, which will still be present as the power supply to this project. These batteries are easy to use as a power supply and have a long lifetime that makes them ideal for this project. However, designing a circuit in order to keep track of the sensitive nature of the Li-Ion batteries would be too much to take on this semester. Lithium Ion are know for their sensitivity in regards to over-charging and over-use. Over-using or overcharging these cells may pose problems in the long run and affect the efficiency and lifetime of the cells. Creating a circuit for determining these sensitive amounts while still considering the AC/DC conversion that would be needed for a recharging system is unattainable at this time. Because this was a stretch goal for our project, there are no requirements that will be broken if this portion is not completed.

When considering the alternative plan for this project, the power for the Raspberry Pi and additional light sources needed to be taken into account. All of these systems, when researched, were low power systems. The Raspberry Pi could be powered given 5V with 2.5A input. This will be discussed more in the voltage regulator section regarding the alternative design. The top lighting system was designed so that it could be powered with the same 5V input as the Raspberry Pi system. These proved to both be able to be powered by the same two 3.7V Lithium Ion batteries as were chosen for the first design. This would not require a change in the battery choice. There was an additional battery chosen to power the base light, which was a 9V Lithium Ion battery. This battery powered a base light of seven diffused LEDs, which would allow the sensor used enough light to view individual sample slides. This 9V battery was chosen to be Lithium Ion because of their greater current capacity, which aided in increasing the battery life of the system.

When taking these batteries into consideration, it was important to calculate the battery life of the system. Using a battery life equation, which compared the current consumption of the system to the current capacity of the batteries, the two 3.7V batteries were calculated to have a lifespan of 1.15 hours. The total current capacities of the two 3.7V batteries equaled 3000mAh due to their series configuration. This 3000mAh divided by the 2.6A estimated current consumption gave us the calculated battery life. However, when testing this system, the current consumption was seen to be lower than what was estimated. This is due to certain components on the Raspberry Pi being used at different times and therefore affecting the overall current consumption. In reality, the batteries were able to last closer to three hours, showing that the overall current consumption is just over 1A. For the 9V battery, the battery life was very long due to the LEDs low power system. The battery life was calculated to be ten hours. This has proved to be longer, as the system is not always operating and after hours of testing, the battery power was strong throughout.

3.9.2 Voltage Regulator

Voltage regulators are important for regulating the voltage output from a battery source. This device is essential for this project because many components need a voltage to maintain a certain range. Certain components require certain voltage ranges and if the voltage is not contained within this range, the components could burn, and the design would need to be replaced. If the voltage goes unchecked and a component burns, the voltage from that burned component would proceed to destroy other parts on the board and cause the entire design to be destroyed. In order to not have the components destroyed and keep the voltages in a desired range, we need to include a voltage regulator.

In electronic devices, the voltage regulators are kept within range by utilizing semiconductor devices to smooth out discrepancies in the current flow. Voltage regulators behave much like a variable resistor that changes with the load of the circuit. When the load is high, the resistance decreases and when the load is low, the resistance increases.

There are many different types of voltage regulators, and many range in price, components and efficiency. Some of these include regulators like the Zenner controlled transistor voltage regulator, discrete transistor series voltage regulator, automatic voltage regulator and constant voltage regulator. These types of voltage regulators can be used from larger systems and generators to small circuit voltage regulators.

The two main types of voltage regulators are linear voltage regulators and switching voltage regulators. Linear voltage regulators are usually compact and efficient for low power systems. Switching regulators are better with efficiency with a wider range of voltages present.

3.9.2.1 Types of Regulators

Linear regulators are some of the most basic voltage regulators to regulate voltage. These devices work by adjusting series resistance based on feedback voltage. By testing the feedback, the regulator is able to adjust the resistance accordingly. This adjustment of resistance helps with maintaining a constant linear voltage. These linear regulators are low power, low cost solution to regulating the voltage in an electrical circuit. Because of their simplicity and low cost, they are the most popular of regulators.

There are downsides to this type of regulator. One of these is that linear regulators need a large input voltage in order to maintain regulation. For example, common regulators often require at least two volts to maintain the voltage correctly. This means if the voltage that needs to be regulated is 4 volts, then there needs to be a 6 volt input in order for the voltage desired to be maintained. As the voltage difference between the input and output increases, the power dissipation becomes less and less efficient. As this happens, the efficiency of the regulator decreases. The linear regulators work by taking the difference of input and output voltage and burn off the waste voltage as heat. The larger the difference, the greater the heat produced, which would not fare well for our design.

These linear regulators work at very low efficiencies, usually around 40%. Many times these regulators can produce so much heat there needs to be heat sinks in order to cool the circuits. Using these heat sinks can make the circuit bulky and increases battery usage. These are all negatives for our circuit. We don't want to heat up the circuits in this project, especially if this will be handheld. If the heat increases to a certain temperature, it could damage our designs and, to a greater extent, harm the user with the increasing temperatures within the metal encasing. After finding out these things about the linear regulator, this seems like not a viable option for our voltage regulation. We moved on to investigating the viability of a switching regulator.

After discovering the disappointing efficiency of the linear voltage regulator, we moved onto searching for information on the switching regulator. This type of regulator is used when high efficiency is needed, or there is a large difference in voltage that is expected. Switching regulators work by taking smaller amounts of energy from the input voltage and move them to the output. This is done by using an electrical switch and a controller that monitors the movement of energy from the input to the output. Using this method, the efficiency raises from 40% with the linear regulator to 85% efficiency. Because of this small movement of parts of energy, this regulator can work much easier with a larger difference in voltages. Though our circuit will not have such a large difference in input and output voltages, it is an advantage for other situations.

These types of regulators usually require more components in order to be successful and the values of these components have a higher impact on the performance of the regulator. These values can pose design problems that can compromise the effectiveness of the regulator and the other circuit components. Because of the complexity of these circuits, they are not as popular as the linear voltage regulators. Even though these circuits are more complex and may require more time and design work, it is worth the increased efficiency. The more efficient the regulator, the more efficient and protected the designed circuit will be.

3.9.2.2 Voltage Regulator Design

When looking over the options of voltage regulators, we decided on designing a switching voltage regulator. Linear regulators may be popular, inexpensive and simple, but switching regulators are more efficient, better for battery life and will provide the circuit design with more protection. The protection of the rest of the circuit is a great concern, as we will be working with important components that will be essential for imaging and communication between devices. These components have particular voltages that we need to stay in range of for their own protection.

Using a linear regulator may not have the efficiency that is needed for these particular components and may cause other damage to them. Having an inefficient voltage regulator could cause too much voltage or current to move through these devices, causing damage to them. Too much of either of these could cause the components to be made unusable because the internals would be completely destroyed. There was some concern with the noise that a switching regulator would create, especially with a Wi-Fi chip. However, after continuing the research process, it was found that a small amount of noise, in the kilohertz range, does not disrupt a Wi-Fi signal being processed. A Wi-Fi signal is set at a standard of 2.4GHz, so a few kilohertz signal will not affect a gigahertz signal because of the vast difference in frequency range. Therefore, we still moved forward with the designing of a switching regulator. Knowing to keep the noise of the circuit low will be essential knowledge for when we design the proper regulator for the project.

Switching regulators would guarantee protection because of the way they operate. As voltage is input into the regulator, small bits of energy are released as output. This switching regulates the voltage that is needed for the circuit. These regulators may be a little pricier as well, but the efficiency makes up for this. The more efficient the device is, the more accurate the regulation will be, and the more protection will be provided for the circuit.

3.9.2.3 Voltage Regulator Design Comparisons

After finding the appropriate regulator, we moved to the design stage. Since we knew the minimum and maximum voltages for each of the main components in the circuit, we began the layout portion for the switching regulator.

For the design portion of the voltage regulator, we found that a switching regulator would be the best option, both for efficiency, accuracy, and battery life. It was also best for the constant output that this regulator would provide for the circuit. A problem that occurred was the switching frequency that would occur in these regulators. When the capacitors switch during the regulation process, noise is produced. This is important to note because we have a Wi-Fi chip that will be present on our circuit. This Wi-Fi chip sends and receives signals that are very important for our design, and adding a large amount of noise to the system may be detrimental. Keeping the noise as low as possible will help our Wi-Fi chip function properly and make it easier for it to process properly.

Also when researching this, it was important to choose which switching option to use: integrated or external switching. External switching is when the switching mechanism is outside of the integrated circuit (IC) and integrated switching is when the switching happens inside the IC. After looking in depth into both of these options, we found that the integrated switching would be better overall for us.

When comparing integrated and nonintegrated switching, the integrated circuit designs were more compact and saved space while also saving power consumption. They were found to have simpler designs as well. All of these positive factors helped us focus our attention on the integrated switching voltage regulator. After knowing this information, along with the general guidelines for the voltage rail system for our printed circuit board, we moved on to find designs that could help us create the ideal voltage regulator.

After referencing the TI Webench for different design options, we selected the following components for comparison: TPS561208, TPS561201, and TPS563210. These chips and their respective designs fit into the previous design specifications that were mentioned earlier. All of them were able to fulfill the low frequency, integrated switching and were able to work with our desired input and output voltages that are necessary. The comparisons for these chips and the designs for the proper regulator circuit are shown in the table below, Table 6.

	TPS561208	TPS561201	TPS563210
Chip Design Characteristics	- 4.5V – 17V input - 1A output - Syncronous Step-Down Convertor	- 4.5V – 17V input - 1A output - Syncronous Step-Down Convertor in 6-pin SOT-23	- 17V input - 3A output -8-pin Syncronous Buck Convertor
Design Footprint	42mm^2	42mm^2	57mm^2
Chip Cost	$0.25	$0.25	$0.50
Components in Design	7	7	9
Total Design Cost	$0.98	$0.98	$1.29
Efficiency	84%	89.4%	92.8%
Frequency	542.82 kHz	100.97 kHz	139.33 kHz
Topology	Buck	Buck	Buck

Table 6 - Voltage Regulator Design Comparisons

When comparing these chips and their designs, it was important to remember two important things for our circuit: frequency and efficiency. Efficiency is an important factor because the chips that we will be working with are voltage sensitive. The higher the efficiency of the design, the better regulated the voltage will be. The other important factor, as discussed earlier, is the frequency emitted from the design. This frequency can affect the signal received by the Wi-Fi chip.

When looking at these factors in Table 6, the two best options are TPS561201 and TPS563210. These are better performing in efficiency and frequency than the TPS561208. Therefore, this option is eliminated. When comparing the final two, there is a difference in efficiency and frequency, but not by much. When comparing the other portions of these designs, the TPS563210 requires more space on the printed circuit board, costs more and has more components required. It is more efficient than the other option, but has a higher frequency. The efficiency is 3% higher and the frequency is 39 kHz more than the TPS561201. With all of these comparisons taken into consideration, the TPS561201 was the chosen design. This design has a slightly lower efficiency, but a lower frequency, lower cost and fewer parts needed. The design for this circuit is shown below in Figure 9.

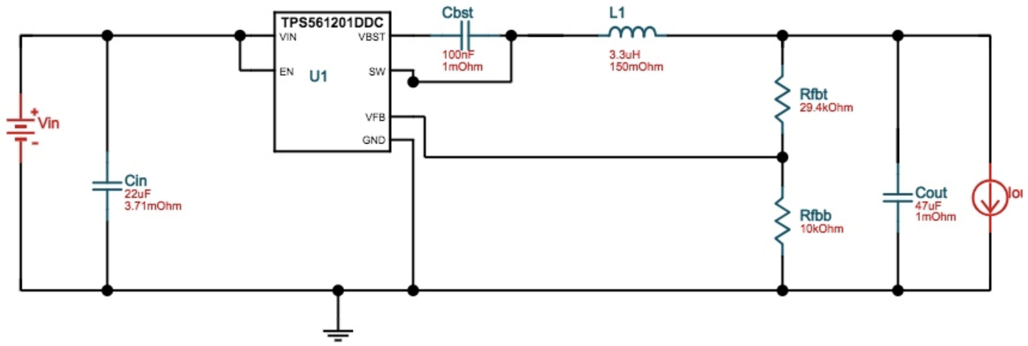


Figure 9 – Voltage Regulator Design Schematic

3.9.2.4 Voltage Regulator Design Alterations

When reviewing the data sheets of this project, it was realized that the regulator selected would not fit the voltage needs of the design. Instead, the three main chips required three different voltages in order to function. The regulators that were required were a 1.8V, a 2.8V and a 5V regulator. These were chosen to be switching in order to increase battery life and accuracy of the voltage output from each regulator. The regulator chosen for the 1.8V output is shown in Figure 10, the 2.8V output regulator in Figure 11, and the 5V regulator in Figure 12.

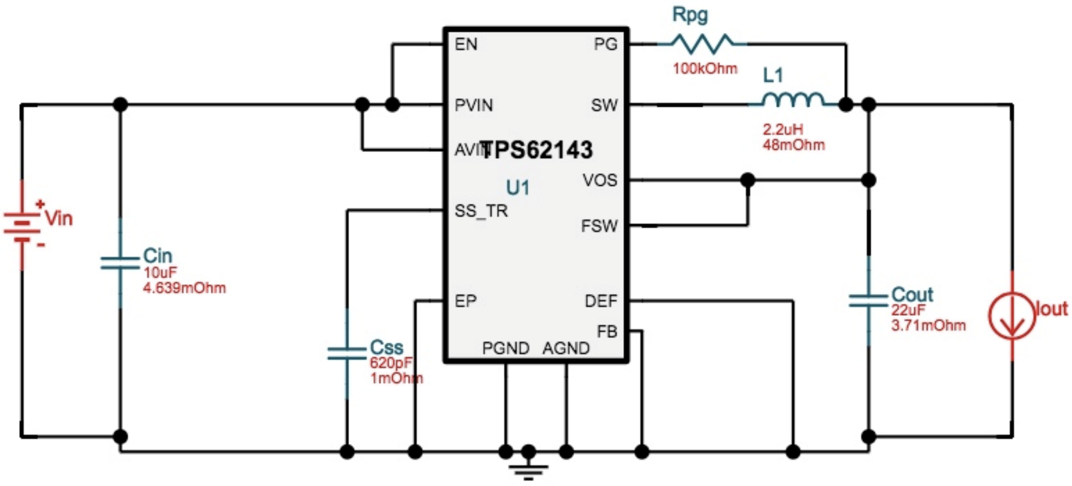


Figure 10: 5V Regulator Schematic

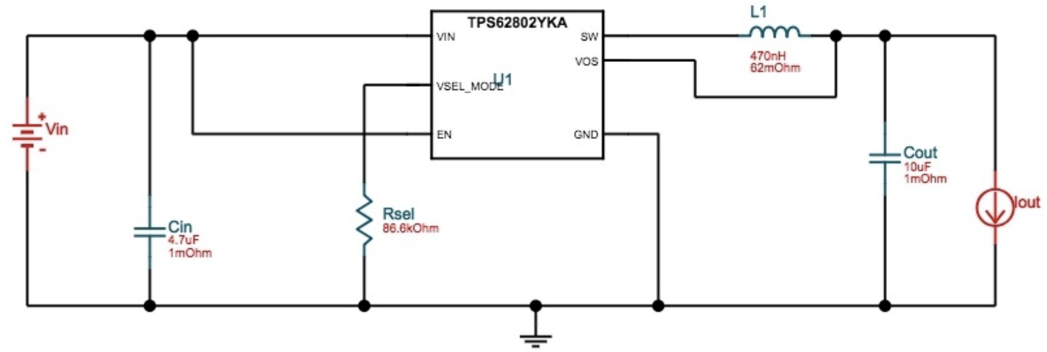


Figure 11: 2.8V regulator schematic

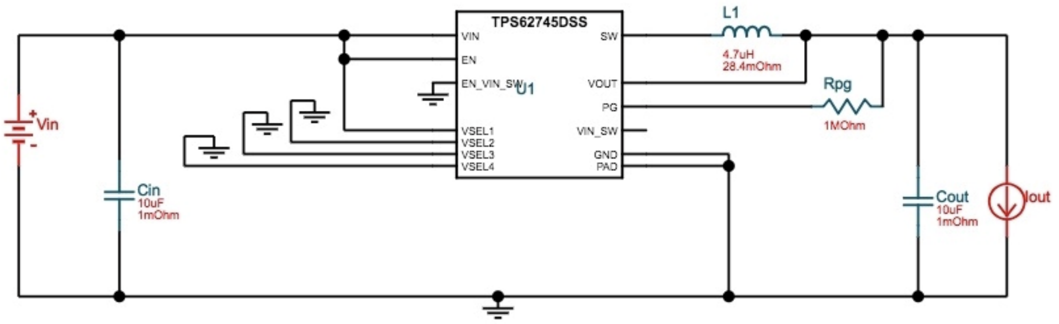


Figure 12: 1.8V Regulator Schematic

When considering the power needs for the alternative plan, the regulator that was required needed to be altered. This is because the current output needed to be greater than what was designed before, as well as the voltage output. The required power input for the Raspberry Pi, and therefore the minimum output from the regulator, was 5V with 2.5A. This also needed to include the powering of the top light LEDs that was about .1A. Using these two values, the total voltage output needed to be 5 volts while the current output needed to be 2.6A. TI Webench was used to create a vast amount of designs that would fit these criteria.

We chose a regulator that would use the TPS564201DDC as the regulator along with other resistors, capacitors and inductors that would regulate the voltage properly. The design chosen was a switching regulator. Though those types of regulators usually require more space, this regulator was very high in efficiency, which would assist the batteries with extending their lifetime. The schematic of this circuit is shown below in Figure 13.

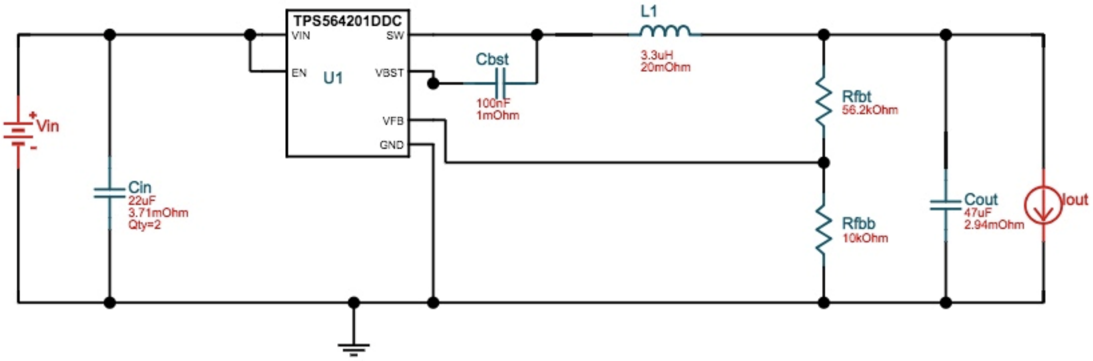


Figure 13: 5V Alternate Regulator

3.10 Body

The body of this project is a major factor for our design. It is the structure that contains all the components necessary and what the user will interact with the most. It is necessary for the body to have a durable design for its portable nature. This section of this paper is dedicated to discussing the construction and design of the body for this project.

3.10.1 Project Body Concepts

To begin this project, we discussed the layout of the project's body. A major requirement for this project is that it needs to be portable. When first envisioning the design of this project, we pictured it to be about the size of a flashlight, since there would need to be lenses inside to perform the magnification. The sizing was not as small as other products we have seen because of the lenses characteristics and the spacing needed for each individual focal length. Because we were not just using an image sensor alone or some sort of camera, the device needed to be more lengthy and thicker in order to accommodate such internal devices as the printed circuit board (PCB), battery pack and internal lenses.

We searched the Internet for design ideas and to see if there were products that were similar to the structure we planned to build. We did find a few, but they dealt with the zooming of a camera instead of actual magnification, as a traditional microscope uses. These examples, such as that shown in Figure 14, were beneficial in seeing how the outside structure would be laid out. The focal adjusting with a mechanical component, the guard for the objective lens and the diffusion of the LED light were all ideas that were included in this project.

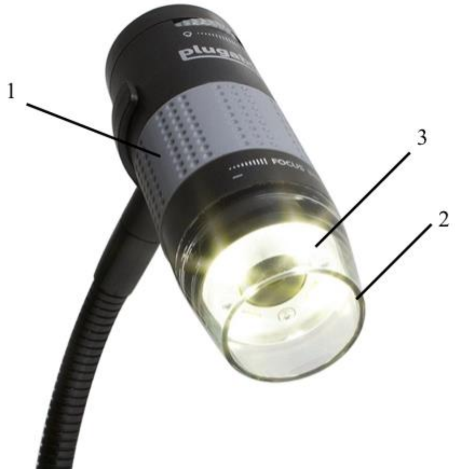


Figure 14 : Example of Existing Portable Microscope

Ideas from other products helped form the idea we began to develop for our own body structure. The lighting of this structure was good to see, even though it only had white light LEDs. This showed how a plastic casing around the lights diffused light well and that just a few LEDs worked well with this particular design. This was essential when considering the lighting we would need for our own project. It gave us an idea of how feasible LEDs would be to use as a low power and cost effective light source. This design also gave us the idea of the adjusting the focal length of the lenses. The guard would also be useful in keeping the object steady when looking at high magnitudes. Slight movements when so magnified can make the visuals blurry and lose focus.

This guard also protected the objective lens from any of the specimens that are being observed. The guard also helped the user see where the microscope is pointed and would make it easier to focus on the desired object. These were all important ideas to help move the exterior design forward and consider problems that may arise later. Looking at existing products solved all concerns with focusing, stability and protection.

After extensive research for the design of the body, we decided to make a moving optomechanical focusing system. The design consists of an ACME threaded rod with matching flange nut. The flange nut is attached to a guide rail system of two steel rods, this keeps the flange nut from turning as the threaded rod is being turned. The rod is attached to ball bearings and a knob to allow it to turn in place, as it turns the flange nut will move up and down.

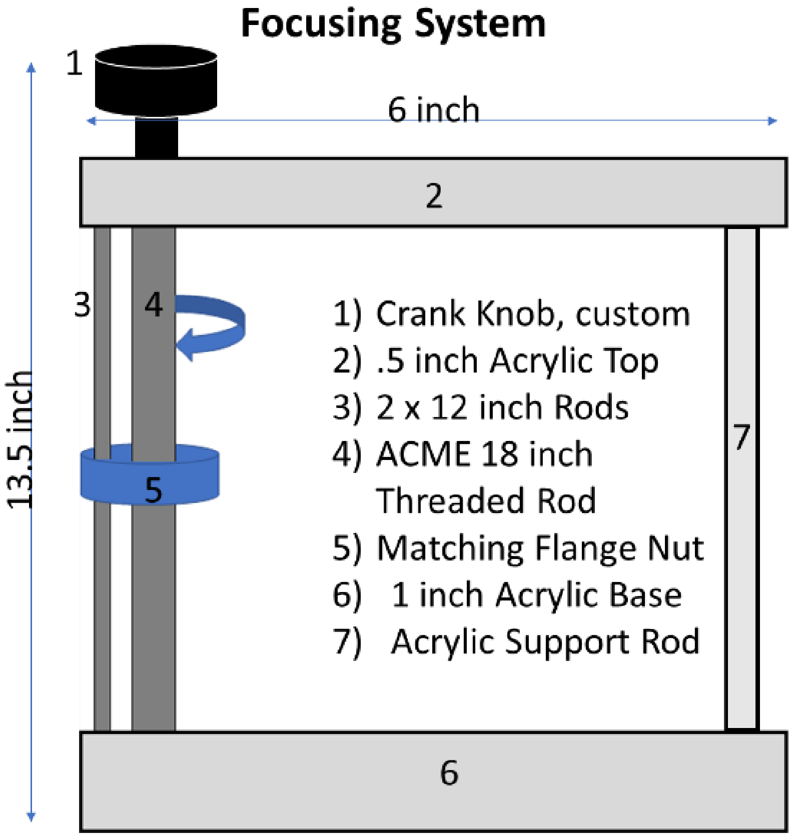


Figure 15: Focusing System-

This is the full design of the vertical translating focusing system. This allows us to move the optical elements of the tube up and down with ease. This allows us to focus onto our specimen.

The base of the system is a 4” x 6” one inch thick piece of clear acrylic. It has a hole in the base to allow the base LED system, and a removeable diffusion filter also fits in it. The Top of the system is also 4” x 6” clear acrylic, but it is half an inch thick. We also included clear acrylic rods to support the system. On the focusing side of the body a black acrylic panel has been installed to allow the electronics to be mounted. The figures below illustrates the design in Figure 15. The actual fully built system is shown in Figure 16 and Figure 17.

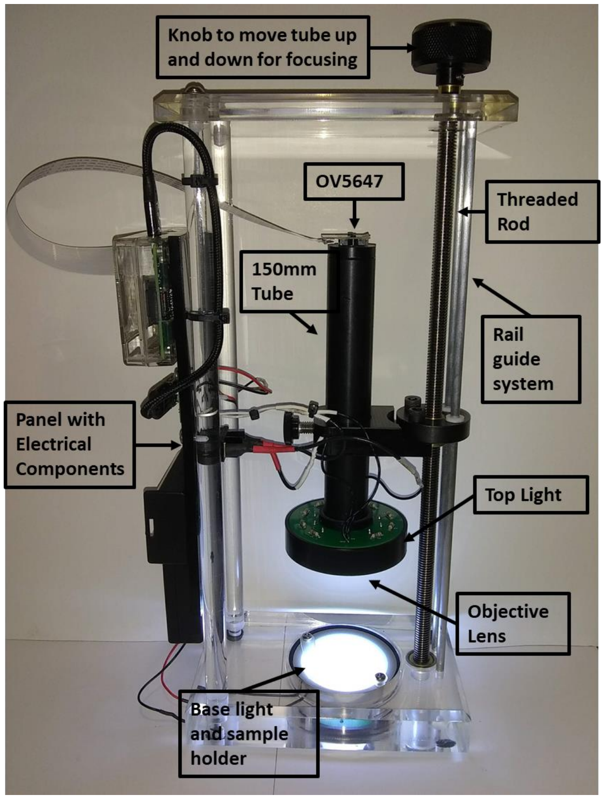

Knob to move tube up and down for focusing
OV5647
Threaded Rod
150mm Tube
Rail guide system
Panel with Electrical Components
Top Light
Objective Lens
Base light and sample holder

Figure 16 Body implementation

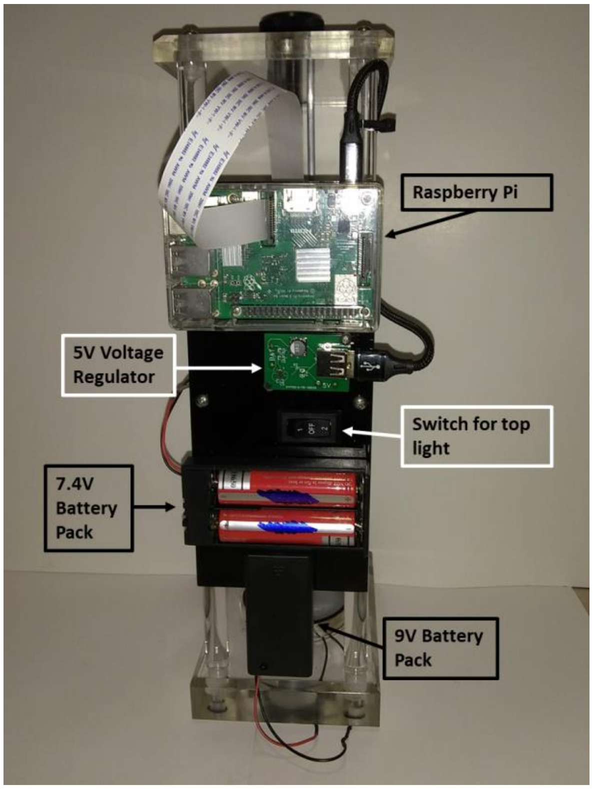


Figure 17: Panel for Electronics

3.10.1.1 Lighting Design

Lighting was an issue since the light would need to hit the object and illuminate the subject enough that the light can be focused and seen by the image sensor. Light emitting diodes, or LEDs, were our first thought. These devices are low power and have a high light output, which would be perfect for our handheld, battery powered design. Our focus was for there to be white as well as infrared light to show the different ways specimens react to the different light spectrums. There were many ideas, which were thought of for the arrangement of the LEDs around the objective lens, but we settled on a ring of alternating types of LEDs, both white light and infrared light. This design setup would give us a proper amount of lighting for the specimen and would provide an even distribution of the types of light shown on the desired object. This layout is shown in the diagram below, Figure 18.

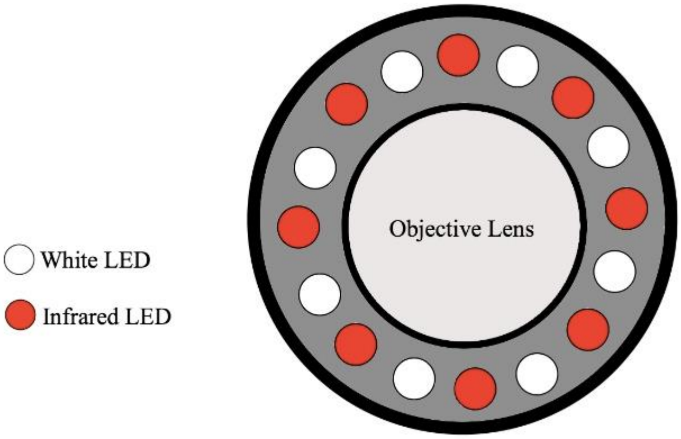


Figure 18: LED Layout Around Objective Lens

It was also discussed that there would be a transparent plastic enclosure for the lights to help disperse the light to cover all aspects of the desired focus. Another problem that came about was the thought of the light sensor picking up visible light instead of only infrared. This was solved by placing a guard along the objective lens of the system, as seen in Figure 9. However, on the example, the guard was transparent. The guard on our project was designed to be opaque, so that there would be a more concentrated amount of the specific desired light that would be captured by our image sensor within the body of the microscope. The guard would then not only protect the objective lens and provide stability, but also produce a purer sample with the lighting that is wanted.

For our final design, we were limited to using only white light diffused LEDs due to the limitations from the final image sensor. In the end, we were only able to view images in white light. The design remained the same circular design and this is discussed later in this paper. We also included a base light with a diffusion filter, pictures of this system will be in section 5.3.1.2 light sources. The design will also be discussed further.

The opaque nature of this guard would be able to keep out visible light when infrared is the desired input, therefore eliminating distortion on the processed image. The infrared image would not be affected by visible light seeping through, as there was if the guard was transparent. The diagram for the overall design of the system is shown below in Figure 19.

This diagram shows the overall design shape of the end project, the objective lens surrounded by LEDs, the adjustable focal length and the opaque guard.

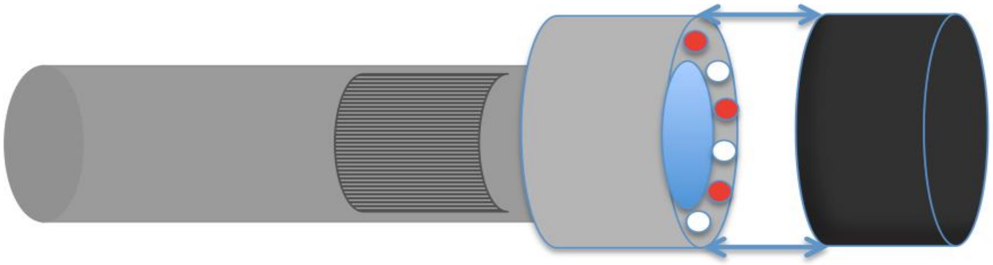


Figure 19: Overall Design of Body

3.10.2 Plastic Enclosure

A plastic enclosure will be used to encircle the inner tube containing the optical lenses. This outer tube will be useful for the outer grip of the project as well as containing any wiring connecting the LEDs to the controller on the printed circuit board, since they will need connections that can reach the length of the project. If lens focus is also a problem later in the project, the inner tube may be adjusted by a physical turning mechanism that will move the lens closer or farther from the object that is being observed.

3.10.3 Body Construction

After discussing the problems that may arise with the lighting, guard and adjustment parts of this project, we moved on to how the exterior would be built. There are two major options that we have for the construction of our exterior parts: 3D printing and machining. These are the two most open options for us. However, there are positive and negatives of each.

3.10.3.1 3D Printing

3D printing has become a very popular method of construction due to its additive nature and comparatively fast output rate. 3D printers are able to use a variety of different materials to give the user the finished product they desire. The material cost is also very low, and printers are available for a lower price than machining devices. 3D printing allows the user to create single, connected parts, something that machining would need welding to achieve. This printing technique is beneficial to many small companies and product developers because of its fast printing rate. A design can be made and printed, then analyzed, corrected and reprinted. This ability to alter designs so quickly and easily is a huge advantage.

3D printers are more portable than other machining devices, but this factor doesn’t play much into what we need the printing for. A more advantageous aspect is the 3D printer’s ability to be additive instead of using subtractive techniques of machining. Machining takes a piece of material and grinds away at it to make a product, while 3D printing begins with material and builds upon itself. However, these 3D printers are usually restricted to a few materials, mainly types of thermoplastics.

We will have a 3D printer available to us for all of our 3D printing needs, so this is not a problem for us. However, the exterior of our project will pose many problems if it was 3D printed. The lenses we are using will need specific threading that will need to be calculated and replicated for the 3D printed design. This takes more time learning the program that is needed to design this as well as creating the exact threading that would be needed to perfectly fit each individual lens. 3D printing would be better suited to help us with custom battery cell holders and LED covers that would help with protection and dispersing of light.

In our final design, 3D printing was used in order to create an aperture to limit light on the sensor. This is described in more detail later in this paper.

3.10.3.2 Machining

On the other hand, machining can work with a more diverse set of materials. These materials include soft materials, such as thermoplastics and modeling foams, as well as hard materials, like wood or metal alloys. Machining requires training to learn the different machines available for what is needed for the chassis of out project. After taking a machining class, we found this would be the best option for creating the case for our microscope. This class taught us important machines like the lathe, drill press, band saw, and the milling machine, along with all the tools associated with each. These tools will be beneficial for many parts of our project.

After taking the machining class, we quickly discovered that threading using a lathe would be much easier than programming threads and certain thread counts using a 3D printing software. The individual thread counts can be calculated and replicated using threading tools in the machine shop. These threads can be made within non-reflective plastics or harder materials like steel. These can be custom made for our chosen lenses so we won’t have to worry about the threads being compatible. Making the threads through machining, we will know if the threading lines up or not almost immediately.

The process of machining takes less time than 3D printing as well. Though the blueprint and setup is tedious and drawn out, the finished product is done quicker and easier than the 3D printing would take. 3D printing can take up to a day for the object to print. If the 3D printer has a large amount of projects to print, our individual project could take weeks waiting to be printed. This process would take too long for us to fix any mistakes that were made in the design process. Machining is an almost instantaneous process that can be fixed as soon as a problem arises. The blueprints can be customized with the shape, texture, and material. A machine shop is available to us, so that is also not a problem.

After reviewing all of these points, the machine shop seems to be the most versatile and best option for the exterior shell of our project. 3D printing is beneficial for less complicated shapes and pieces for this. More complex maneuvers such as threading and drilling are easiest to do using the machine shop.

3.10.3.3 Diffusion Filter Construction

In further planning for the design, it was found that the construction of the diffusion filter would be created through using recycled, scrap plastic. This diffusion filter will need to be a frosted transparent plastic that will evenly disperse the light given off by the LED high sources. The machine lab will be used to create the tubes with threading needed to hold the proper lenses in place. Machining these pieces was quick and easy. Machining this type of diffusion filter may be difficult due to the shape of the material and the material used to create it.

The diffusion filter was created by using a solid 3 inch diameter piece of Delrin. This was then hollowed out to create a ring of Delrin. We then cut a piece of plastic from a milk gallon jug to a slightly larger circle. We fastened this piece of opaque plastic using three screws. The diffusion filter fits securely into the base of our microscope without the need for adhesive or other fasteners. This is useful so that it can be removed to fit specimen that are too large normally.

On top of this piece, we decided it would be useful to add clips to the filter. With the clips, we can now hold the sample we are studying in place, it also allows the sample to stay flat so that we have an even surface to examine. These clips were taken from a children's microscope. A depiction of all the pieces together is shown below in Figure 20.

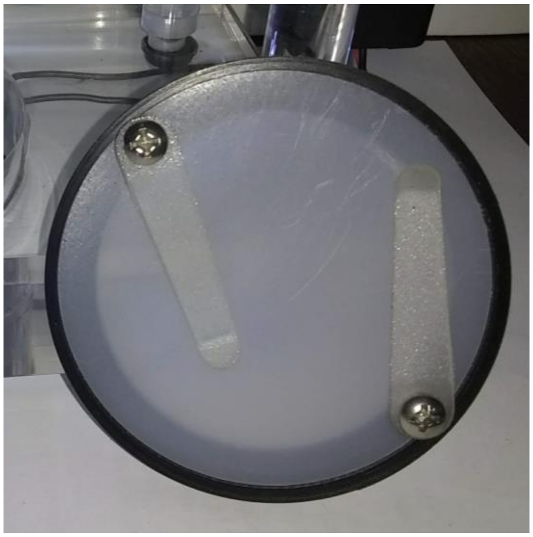


Figure 20: Diffusion Filter with Clips

3.11 Mobile App Cross-Platform Considerations

The initial plan for the mobile application was to allow cross-platform support so that iOS users could also experience the app. To do this, the plan was to write all the code using React Native, which is a framework used for building mobile applications using JavaScript and code native to the operating system that you're writing the application for.

React Native was initially considered for this project because of how simple it is to create cross-platform applications. React and React Native are both known for having a steep learning curve, but they're also known for being incredibly flexible and powerful once a developer has the basics down. Another benefit of using React Native is that since its library abstracts over the operating system SDK API, the codebase that a developer is writing suddenly no longer depends directly on that SDK directly. This is a major benefit because when iOS or Android pushes out new changes to their SDK, fewer things break. This allows a developer to detach themselves from these potential concerns and focus on the code that they're writing and not stress about potential application-breaking dependencies that could become deprecated during a major code release by Google or Apple.

While our app developer has some experience with React, they do not own a physical computer with OS X on it, which is required to develop iOS applications. Additionally, while there are ways to obtain access to OS X, such as through virtualization, we decided that since the app developer has prior experience with building Android applications and that there is only one person working on the app that they should just focus on writing an Android application and leave an iOS counterpart as a goal for a future team that might work on this project.

Android's native Wi-Fi direct support and iOS's choice to use and develop only a proprietary nearby peer-to-peer communications format contributed to our decision, but Wi-Fi direct is ultimately a convenience that this project could have gone without, though it adds weight to our preference of Android.

3.12 Parts and Components Selection

After reviewing the research, the parts and components were chosen for the final design of this project. These components were ordered, and most have been received by the finalization of this paper. Those that were chosen and received are shown below, in Figures 21, 22, and 23 with numbers indicating the individual parts and components. The numbering on these images is explained in Table 7. Any parts that are not pictured are ordered and on the way, but are not in our possession at this time.

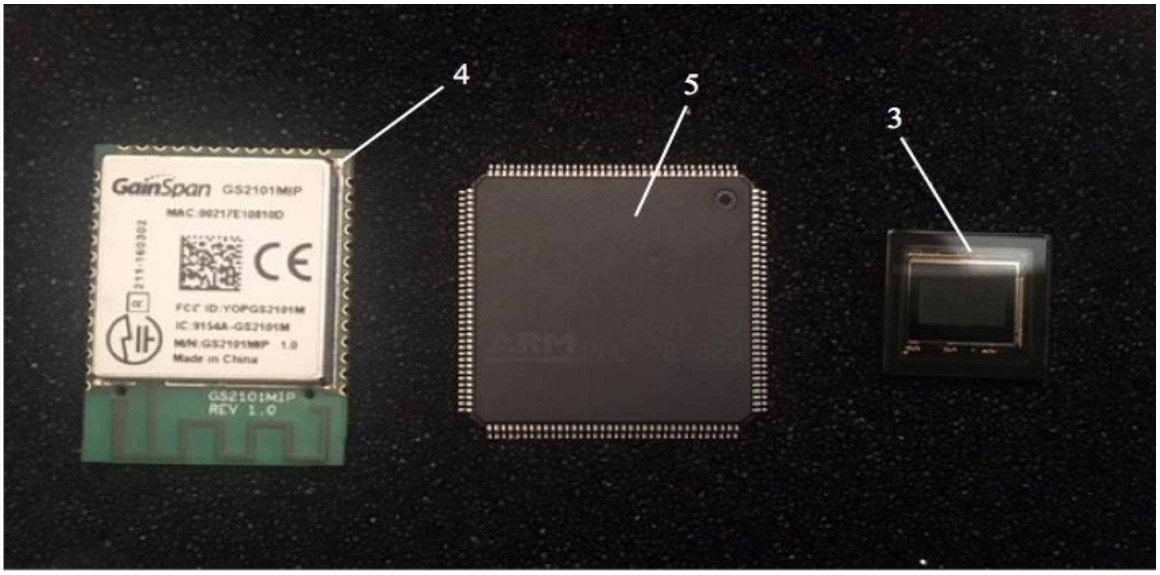


Figure 21 – Final Chip Selections

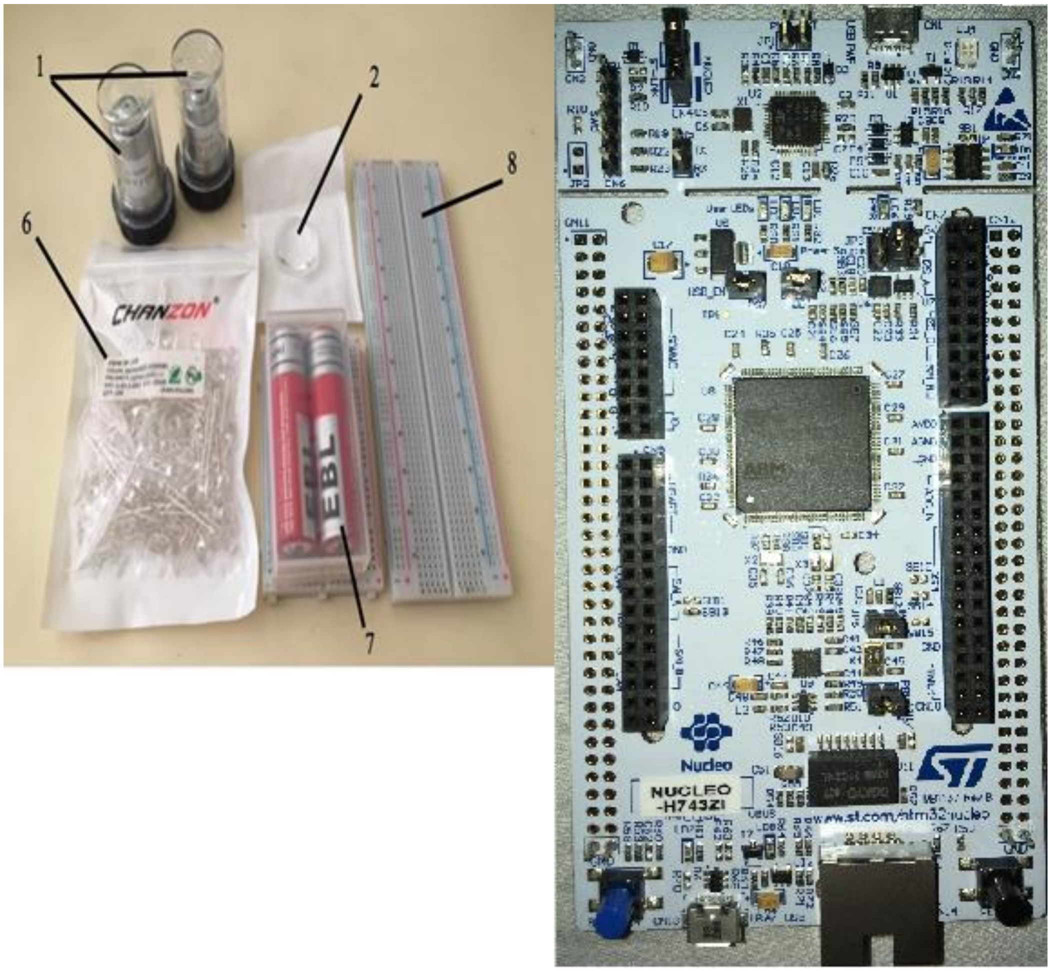


Figure 22: Parts and Components Selection, STM32 Development Board

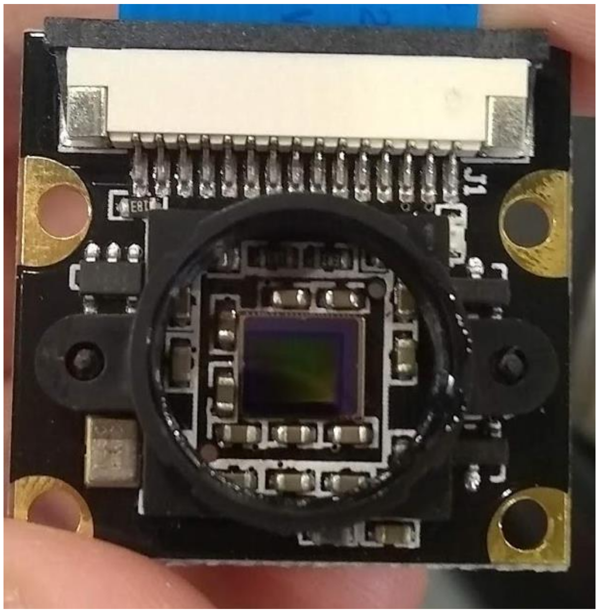


Figure 23: OV5647 Image Sensor

	Description	Manufacturer	Part Number	Qty	Unit Cost	Total Cost
1	Objective Lenses	Amscope	N/A	1	$47.60	$47.60
2	Lens	Newport	KPX585	1	$21.40	$21.40
3	Image Sensor	ON Semiconductor	AR0237IRSH12SHRA0-DR	1	$11.15	$11.15
4	Wi-Fi Chip	Telit	GS2101MIP	1	$8.41	$8.41
5	Microcontroller	STMicroelectronics	STM32H753ZIT6	1	$23.52	$23.52
6	IR LEDs	Chanzon	100F5T-IR-FS-850NM	Pack of 100	$7.75	$7.75
7	Battery	EBL	EBL C839 charger with 18650 Li-Ion Rechargeable	1	$13.99	$13.99
8	Breadboards	Aspen Burg	N/A	1	$6.50	$6.50
9	MCU Development Board	STMicroelectronics	NUCLEO H743ZI	1	$22.01	$22.01
10	Wi-Fi Development Board	Telit	GS2101MIP-EVB3-S2W	1	$179.0	$179.0
11	Image Sensor	OmniVision	OV5647	1	$16.99	$16.99

Table 7 – Parts and Components List

4.0 Standards and Constraints

For this project, we encountered many aspects of the design that had constraints and standards that came along with them. The realistic design constraints as well as the technological standards required are included in this section of the paper.

4.1 Constraints

The Portable Microscope project includes many hardware and software design constraints that need to be addressed. These constraints are important to address when discussing the desired outcomes of the project and shy certain aspects could not be fulfilled. These are discussed in the section below.

4.1.1 Economic Constraints

The cost was the most limiting constraint for us in this project. Unlike some of the other teams in Senior Design this year, the members of our group are the only funding for this project. There are no outside sponsorships or funding that are contributing to the expenses of this project. This has been a very limiting factor because we do not have the funds for some of the more advanced image sensors or other optical components. The combination of optical image sensors, lenses, filters and other components can reach into the thousands, even tens of thousands of dollars. We could not afford these types of components, so we were limited in this way.

The microprocessor, Wi-Fi chip and image sensor are all important components in the electrical and computer portions of this project. These components are very sensitive to current and voltage. Since we were so limited by our budget, we were not able to try multiple expensive designs and make errors time and time again. Our group resorted to researching the best designs and what would work best for our project, as opposed to performing trial and error testing. We were not able to order several different components to test which ones would work the best. We were only able to choose one component that we thought would work the best, and from there we made the design work together.

Economic constraints would also include the market price. If we were to market this to consumers, the price would have to be reasonable for the product to be successful. The optical components were the most expensive in this project. Most companies go around this by using zoom as a replacement for optical magnification. Because we are using physical optical magnification, this product will naturally cost more. This limits our consumer market to higher paying customers. In an ideal situation, we would use the best products at the lowest prices, but this does not exist for this project. Because of this, we were limited and had to find a balance between price, functionality and quality.

4.1.2 Environmental Constraints

It was found there were very few which would affect the system in a considerable way. The only environmental constraint that would apply to the device in isolation is that the ambient light that could affect the infrared and visible light that is observed by the image sensor. Other than that, the issue of energy usage is not a concern. If we discuss the environmental concerns about energy, the batteries that were used are rechargeable and have a long lifetime. This was beneficial because used alkaline batteries will not be needlessly thrown into landfills, where they usually go. It is better for both the user and the environment that the batteries have this capability.

The environmental constraint with the largest likelihood of damaging the device or its operation is the abundant utilization of the Wi-Fi spectrum, which can bottleneck the physical capacity of the medium. This is especially concerning in a city setting, let alone a metropolitan setting, and we consider it essential for our device to operate under the circumstantial difficulties brought on by its location and event of presentation; failure during a showcase is the single highest-risk moment for failure, and a busy event on a university campus is among the least likely places to achieve adequate wireless bandwidth. The device's functional setting was also challenged by this aspect, as we designed this to be primarily used by educators in classrooms.

One method we've planned, long-term, to help mitigate the load on the electromagnetic spectrum is the optional support of broadcast from the embedded system itself, allowing an instructor and any number of nearby students to tune in to the Wi-Fi signal without additional adverse effects on the medium.

4.1.3 Social Constraints

A major goal for this project was to make the portable microscope user friendly and have very few problems that could occur on the user's end. To make this project user friendly, everything needs to be constrained to one solid object and controlled by an outside source, like a phone or tablet. Having this be the case, the mobile application that was created has a very easily controlled user interface and is visually appealing. These are all important aspects because the less the user interacts with the internals of the system, the better it is for the designers. We want this product to be available to everyone, be that adults or children, for educational and entertainment purposes.

Though there are options available to the user in terms of light spectra and magnification, there are also control systems available in the app. These control systems can control the LEDs illuminating the project and have the user determine the amount of lighting on the object or specimen being observed. The user is also able to control the video streaming and image capturing of the object or specimen. We tried to make these controls as easy as possible for the user to make the user's experience with our project more enjoyable.

4.1.4 Political Constraints

There are multiple FCC regulations concerning Wi-Fi use; all of ours exist in the 2.4GHz band. The output power of our device is limited to prevent signal interference and maintain an enforced maximum range of interference, to allow the narrow Wi-Fi band to be more broadly utilized by neighboring users.

The bandwidth of our signal is also limited: pre-defined channels exist for different sets of users. Standard limitations for civilians preclude the use of channels 12-14, though lower-power signals can be sent in channels 12 and 13. This wasn't advisable for our uses, as the risk of signal leakage into channel 14 carries severe weight.

While they did not apply to our project, a large set of regulations extends across the usable electromagnetic spectrum, restricting the device technology we have access to both directly, as some Japanese Wi-Fi chips support channels 12 and 13 for high transmission power use, and indirectly, as the narrow and crowded region we operate under is tightly controlled by the implications of these regulations.

As we consumed low-level hardware, it sufficed to check each piece of wirelessly-interfacing hardware and verify it operates under the specified channels (1-13), the power requirements, and the bandwidth pollution guidelines. Android phones may be safely assumed to operate within all codes and regulations, but many Wi-Fi chips on the market come ready for operation in other countries and on other continents, where the extended channels are valuable space for design. These had to be vetted for control options that allow us to stay within all boundaries.

4.1.5 Ethical Constraints

This project's focus is to provide entertainment and education to the user through the creation of a simpler way to image objects microscopically. In no way were any corners be cut in this project that would endanger our team members working on this project or the end users. No toxic chemicals or materials were used that could damage the product or harm the user. The components chosen for this device were also not altered with the intent to shorten the lifetime of the device to obtain lower cost and higher profit. No features were discarded solely on the base of funding or other finances.

Extensive research was done to be sure our portable microscope project did not infringe on any patents that currently exist. Other previous designs or concepts were not be used without giving proper credit when it was due.

4.1.6 Health and Safety Constraints

When doing research into this project, there were not many health and safety concerns that could occur because of this project. This is a handheld lightweight device that is for educational purposes. The only safety concern would be the voltages short-circuiting and causing problems with the printed circuit board within the body of the project. The worst situation is that the pieces would be burned within and a small fire may occur. However, we went through all the necessary safety protocols with the board layout and printed circuit board structure to avoid such accidents. Other than the electrical components, there should be no other parts that could cause harm to the designers or the users.

4.1.7 Manufacturability Constraints

When creating the portable microscope, it was important to consider the individual components for the system and the availability for the chosen materials. Some chips that we were researching required many weeks for manufacturing and we were limited to options that had no lead times and were available immediately for purchase. We were also limited to how many of a certain part were in stock from the manufacturer. Manufacturing would be less concerning if there were many thousands of each piece were available. However, when only a few hundred remain in stock at a company, this could cause problems for mass production.

The amount of stock also was concerning for the prototype stages of this project. Some products have lead times or other obstacles that would inhibit the prototype construction. These components and parts that have excessive lead times were not accessible for this project.

4.1.8 Time Constraints

The time constraint of this project is the most important constraint of all. The goal by the end of Senior Design II was to have a magnified image of an object or specimen that is processed by an image sensor and sent from the sensor to an external device such as a phone. The process time from initial conception to final product was roughly eight months. The time constraint is important to remember through the concept and design stage of this project. We had several problems that increased our time constraints, meaning that we had to come up with a viable project that was still a microscope in roughly 3 months' time. We were successful in this.

All determined components for use in the system needed to be decided on in a timely manner to allot enough time for testing. There were many different capabilities that could be included in the portable microscope project, but there is limited time to continuously add certain features. For the team to create a successful project, certain key features were obtained and worked on from there. Any additional features were added if the proper time is allotted. To help combat the time constraint, a schedule was devised and follow to the best of the team's ability. This schedule is written in the Administration chapter of this paper under the Milestones section.

The printed circuit board was a major factor to consider in this time constraint as well. Individual parts testing needed to occur to be sure the design of the PCB is correct. The design took time to conceptualize and create. There was time set aside for individual parts testing as well as testing the full PCB design, as well as time allotment for any PCB changes that will need to take place. Any changes that were made to the PCB resulted in a redesigning of the board, placing an order for a new board, getting the components soldered correctly, and further testing of the new design board. This meant the sooner testing occurred and the sooner the design for the PCB was perfected, the more time the team had for other fixes later.

4.1.9 Testing and Presentation Constraints

The testing section of this project took time due to the individual components of this project. Some of the components required a special type of testing board which had to be created to test communication systems. These boards were individually designed in order to test these systems and their pin connections.

Due to a few of the packaging layouts, the BGA designed systems needed to be soldered by another source, namely a local company that soldered surface mount components. Testing these components after this process wasn't a problem. The PCB design was confirmed by these tests and ordered. From there, the PCB was tested and verified that the design is correct.

Optical and light source testing was to be performed as the parts arrive. Once these tests are completed, the designs of these systems can move forward. The body of the project was able to come to fruition and the lighting PCB was designed and ordered for continual testing. When testing the systems for the control PCB, the optical system was used to verify the functionality of the image sensor and other components necessary for this project. Continual testing was able to be performed with the test boards for the components where coding and other controls will be necessary. Application testing was needed for user interface design and communication with the microcontroller. These tests were all performed in the beginning of Senior Design II.

While some of the optical components, primarily the lens systems, were testable through outside means – the use of a camera, visual inspection, etc., the image sensor required electronic control and electronic reading at a very low level, so it was tied to the speed at which the microcontroller's development board comes online. This was one of many reasons we chose to go with the microcontroller we did, as it has a development board which comes with support for reading image sensor output.

As far as presentation, due to its handheld and portable nature, there were no problems showcasing this product. Initial testing for the PCB and control system design were the limiting constraint. Once these were conquered, the presentation of this product was a success.

4.2 Standards

Standards are a set of criteria and requirements that all sections of an industry must abide by. Standards exist for programming languages, operating systems, communication protocols and electrical interfaces. Standards are important because they are used as an overall base for understanding and for the safety of team members and for the end user. The use of standards also ensures that products and systems will be able to communicate properly and that they are compatible. This is especially important for our portable microscope system since many components will be intertwined and the communication systems need to be compatible in order for us to obtain the proper end result.

Our project cuts through optical, electrical, and the extremely broad range of computer harder and software standards, providing us a lot of guidance and pre-determined good design options. These will be detailed below.

4.2.1 Power Supply Standards

Power supply safety standards for this project are provided by CUI and these standards can be applied to many parts of our project. From the circuitry and its components to the construction of the battery and ground for the design. The standards required for the electrical circuit classifications are shown below in Table 8.

The circuitry in the portable microscope project will be an Extra-Low Voltage system, since there will only be a max voltage of 7.4 Volts at the battery terminal. Insulation will be included on the external portions of the wiring from the batteries. The batteries will be encapsulated with a non-flammable plastic insulation, and the printed circuit board will contain plastic shielding as well. This plastic shielding will help with prevention of short circuits that may occur as well as protection against any other metal that may come into contact with the system.

These standards are very important to uphold, especially when considering the team and the future users. If this product goes to market, the safety risks without these standards being upheld would be astronomical. Not abiding by these standards could cause serious harm to the user. It is important to remember these standards and abide by them in order to create a well-engineered system that will not pose a risk to anyone during operation.

Circuit Type	Type Definition
Hazardous Voltage	Any voltage exceeding 42.2 VAC peak or 60 VDC without a limited current circuit.
Extra Low Voltage (ELV)	A voltage in a secondary circuit not exceeding 42.4 VAC peak or 60 VDC, the circuit being separated from hazardous voltage by at least basic insulation.
Safety Extra-Low Voltage (SELV) Circuit	A secondary circuit that cannot reach a hazardous voltage between any two accessible parts or an accessible part and protective earth under normal operation or while experiencing a single fault. In the event of a single fault condition (insulation or component failure) the voltage in accessible parts of SELV circuits shall not exceed 42.4 VAC peak or 60 VDC for longer than 200 ms. An absolute limit of 71 VAC peak or 120 VDC must not be exceeded. SELV circuits must be separated from hazardous voltages, e.g. primary circuits, by two levels of protection, which may be double insulation, or basic insulation combined with an earthed conductive barrier.
Limited Power Source (LPS)	These power sources are designed with prescribed output voltage, current, power and short circuit current limits. Specific methods can limit the capacity of the power source. The requirements for wiring and loads supplied by LPS power supplies are relaxed due to the reduced hazard of electric shock or fire caused by an LPS power supply.

Table 8: Circuit Classification Standards of CUI.

4.2.2 IEEE 802.11

IEEE’s 802.11 WLAN standards affect our design at every level. The extreme utility it provides in the accessible and widely supported range of Wi-Fi options and modalities it enables is the reason we are able to pursue wireless communications. It also guides our design choices at the implementation level: Wi-Fi chips that meet different or multiple flavors of the overarching standard, 802.11a, 802.11b, etc., offer differing feature sets and levels of support at a range of consideration.

Throughput is capped based on the physical associations of the standard. Crowding is heavily dependent upon the wireless band chosen, with differing amounts of traffic at the sub-GHz level, the 2.4 GHz band, and the 5 GHz band. Compatibility between our device and target devices for communication is a notable aspect, though it’s rendered irrelevant by the very well-adopted set of standards and the propagation of their use throughout modern smartphones, tablets, laptops, and wireless access points in general.

These bandwidth requirements are directly significant for our video transfer’s bandwidth needs, though most Wi-Fi formats, especially the current-day relevant formats, support sufficient maximum theoretical bandwidth for our purpose. The specific added demand of the necessary conditional to the prior sentence, “under permissive network traffic conditions”, is addressed in more detail in the above Environmental Constraints section, as it is a primary point of failure.

4.2.3 Other Wireless Communications Standards

Other wireless formats exist: Zigbee, Bluetooth, and near-field communication see common use in nearby, paired wireless devices like those in our project, but only Wi-Fi supports a bandwidth sufficient for our transmissions, which is predictable based on the tight management of the electromagnetic spectrum. Each was investigated, but none proved useful to our purpose.

4.2.4 Standard for Tests for Flammability of Plastic Materials for Parts in Devices and Appliances

Though our project will be a low voltage system, the printed circuit board that will be constructed is mainly a plastic material. Because of this, there will be material standards that will be necessary to follow in order to ensure safety. Due to this, the PCBs used in this project will be designed in accordance with the UL 94 standard. This standard is a type of plastic standard created by Underwriters Laboratories that provides a method for rating the ignition characteristics of different types of plastics. The plastic component in printed circuit boards, known as and FR4 substrate, meets the requirements of the UL 94 standard. By using this material in our PCBs, as well as abiding by the requirements for our power supply and overall low power system, the electrical component portion of this project will ensure safety for both the team and future users.

4.2.5 IPC PCB Standards

IPC, the Association Connecting Electronics Industries, is a trade association that creates many useful standards for electronic components, equipment, and assembly systems. There are many standards that can be applied to printed circuit boards in order to extend the lifetime of the PCB as well as maintain its reliability. The standards for these circuit boards include all layer types, from the smaller two-layer, to the thicker double-sided and many multilayer types. There are many different types of documentation available for researches as well as material and design specifications. There are layout standards that discuss the component mounting requirements as well as generic requirements for the software design used to create the printed circuit boards, as can be seen in IPC-221B. In this standard there are also requirements written for thickness, tolerance, trace spacing and design, as well as spacing between components and their connections.

It is essential for us to follow these standards because they help to achieve a successful design. Following any steps in order to obtain safety, longevity for the project and proper technique for creating a printed circuit will be beneficial for our project. By following IPC standards, we will better be able to communicate with others, especially in the business and manufacturing side, in creating a product that will ensure quality for the end user. These standards will also ensure that the design will create a product that will be able to fulfill any tests successfully in the future and therefore lessen costs, and shorten time taken to fix problems and alter designs.

4.2.6 Software Project Management

Software engineering has developed its own sets of common practices, which, while many are adopted or abandoned based on developer preference, have proved highly valuable throughout both industry and academia. The Software Engineering Institute (SEI) is perhaps the most well-known publisher of these standards, models, and practices, but the knowledgebase common to studied programmers extends back to the first true programming languages.

Clean code, good naming practices, version control, use of common libraries, best practices surrounding the use of common libraries and API, development frameworks, testing procedures and methodologies, system design principles, system design modeling and structuring, the software development lifecycle, and many more all influenced our efforts at both the embedded and phone levels of programming and system design. Further elaboration is unproductive, but mindfulness of the above concepts is essential, and their incorporation will be apparent in Sections 5 and 6.

4.2.7 Physical and Inter-Chip Communication Formats

The transmission of electrical signals between otherwise unrelated components will govern many of our hardware design choices. These allow our design to function without defect while utilizing hardware from multiple manufacturers which need not have any knowledge of each other or each other’s designs.

Their importance is apparent in their broad array of points of failure. Data can be corrupted, lost, misinterpreted, duplicated, interfered with; enough to solidify its importance, all without mentioning that data has a physical manifestation which is more than capable of and routinely does destroy hardware: voltage ratings, current ratings, and temperature ratings must all be followed, or parts of if not the entire set of hardware can be lost to thermal runaway, LEDs, being diodes, can enter breakdown, batteries can very literally explode, and transistors can fail, which is itself a best-case scenario in the face of their potential to contribute to overheating and runaway.

4.2.7.1 Universal Asynchronous Receiver/Transmitter

UART’s truly universal adoption and common mode of operation make it a reasonable introductory format for the serial communications modality. Omitting the history and some of the extended functionality implemented by UARTs, their basic use and designated purpose will de detailed as an example case that provides familiarity with other serial communications.

The basic operation of a UART is as follows: Two paired devices with matching functionality are paired together and interfaced with by independent parties for the purpose of communication. Each device has a Transmit (Tx) port and Receive (Rx) port. The Tx port of one device is wired to the Rx port of the other device, and vice-versa. Either device, to send a segment of data (which is provided through the interface with a host), can pull its Tx port to a pre-defined level (in the case of UART, low) to alert the other device of its intent to send data. The other device, noticing this change, sets up its receiving clock at this point. After this first “start bit”, the sending device begins sending its data one bit at a time. If the devices share the same configuration settings, these rates will be equal, and the receiving device will sample and store each incoming bit in a register.

There are additional control layers to this operation. Options for Parity Bits exist, stop bits – the terminating pieces of each byte-wise transmission – are mandatory, and may be of length one or two, the length of the data between start and stop bit itself is configurable, and the receiving device must implement some method of maintaining the validity of its data, as often one UART communications port is comprised of two lone single-byte registers, each connected to either the Tx or Rx. These are written to circularly, and if one byte is transmitted, and then another byte is transmitted before the prior byte is read, that data will be lost, and the sender will have no way of knowing. This is one method of maintaining the validity of the data – abandoning any thought of guarantee and allowing each device to send data with the knowledge it may be lost.

More common is the strategy of flags. The Tx and Rx ports above are paired with Clear to Send (CTS) and Request to Send (RTS), which are in turn interconnected (RTS to CTS) between the devices. CTS is an input, and whenever the transmitting device would send a byte, it must first check its CTS, the other's RTS, to see if the other device has raised what amounts to a "ready" flag. It is important to note that no request is send; the CTS signal is statically maintained by each device at all times.

This window allows the receiving device time to clear its register, either by waiting for the managing application to come by and consume that data, by transferring it to a First In, First Out (FIFO) queue, or by sending that data to an extended set of memory. In all cases the transmission medium can fail. The managing application can fail to return to the new data, the FIFO itself can fill up in the same manner as the register itself, or the memory may have its own version of RTS/CTS and prevent any new writes. Ultimately, precise (and cooperatively symmetric) management is required at the device level to support this behavior.

The RTS/CTS behavior above is one form of what is known as hardware flow control, and it will be a common theme in this system as Direct Memory Access and other communications formats come into play.

4.2.7.2 Other Relevant Serial Interfaces

While UART is common and will be used in our project, the same is true of other formats, which merits discussion. Many of the other formats are explicitly clocked, as opposed to the method of implicitly clocking by utilizing internal settings, as was the case with the UART example above. One (or both) devices will tell the other device when data is being sent with the rising or falling edge of a clock signal to ensure sampling is always synchronous. This is more useful in many cases, triggered design is widely used, so it is available, and the inherently synchronous format (provided each device is capable of supporting the chosen clock rate) means the device controlling the signaling can increase or decrease the clock (implying transmission) frequency without a handshaking protocol or other methods of initialization.

The Serial Peripheral Interface (SPI) uses a master-slave architecture in which one device maintains an activity/selection signal and controls the clock rate while each device is capable of sending data to the other through dedicated, interconnected input/output lines. This format achieves full duplex communications at the cost of dedicated transmission ports, a slave select port, and the complete loss of slave hardware control – it cannot notify the paired device of a full buffer, so it must choose to either drop the old data or refuse the new data.

I^{2}C uses a paired clock and data line with one master device. The clock is used for rate control and indication of new messages, and the messages themselves include commands such as "read" and "write", which tell the slave device what to do. This greatly reduces port costs at the expense of transmissions overhead, half duplex operation, and loss of explicit slave control options, but its minimal cost for the clocked serial domain and wide support have maintained its presence in hardware.

Many others exist, but those can have their details omitted as their inclusion is usually hardware-supported and more concerned with their Direct Memory Access capabilities than anything else.

4.2.7.3 Common Parallel Interfaces

Where in the above section there were clock signals and hardware flow control systems in place to manage single data lines, many interfaces instead utilize a number of data lines from a single (a-) synchronous control system. These will be critical for our project, as even an 8-bit standard VGA signal at 10 frames per second would require a 25 Mbit/s pure data rate, and all communications incur at least some overhead in practice.

The practicality of parallel data signals is such that many formerly-serial formats have adapted with 2-, 4-, and 8-bit options. SPI, described above, has Dual and Quad SPI modes, which, respectively, selectively repurpose existing wires and use a combination of the former and an additional two I/O wires.

Significant to this design, the Secure Digital Input Output (SDIO) interface uses a selectable number of data lines to interface with other devices. This is traditionally used for memory card connections, but the protocol has been repurposed as a more general form of I/O by some developers, very notably Arm, whose microcontrollers serve as the processors on both the host MCU and Wi-Fi chip in our design. Its details are similarly left to other sections, as needed.

4.2.7.4 Camera Parallel Interface

The CPI will be described in terms of its application in our project, though it has multiple modes.

The CPI is a 15-bit interface with 12 data lines, a Horizontal Sync line, a Vertical Sync line, and a Pixel Clock (HSYNC, VSYNC, PIXCLK). The data lines transfer image data in a format which must be known by other means which is live on each tick of PIXCLK. The HSYNC line signifies that a new line is about to begin, while the VSYNC line signifies that a new frame (image) is about to begin, or, in either case, that a line/frame has just ended.

The extremely parallel data transmission lines (each carrying and entire pixel worth of data) help manage the high transmission rates demanded by high-speed imaging. While the HSYNC and VSYNC lines have traditionally existed to serve physical ends on cathode ray tube monitors, they serve an extremely useful end in our device and many others: they provide signals to Direct Memory Access controllers that can help a system (and its designer) manage memory allocation and the pointer arithmetic and management involved. As an example, if a device had a set of n x 640 memory spaces dedicated to VGA transmission, the pointer could be treated with a simple logical flow, upon receiving a new PIXCLK, of: if (HSYNC) then write_address <= next_chip; else write_address += 1;

The CPI is capable of being reversed, allowing the image sensor to be told when to move from one pixel to the next, one line to the next, and one frame to the next, which has its own obvious utility.

5.0 System Design

Under the constraints and standards documented above, and with the devices specified, their organization and programming into a cohesive unit will follow the protocols developed and described below.

Our project intersects four disciplines, so understanding of the requirements flow of our now compiled pieces, which were chosen to be able to meet the project's complete requirements, will first be established.

Our system is solely concerned with imaging, so the primary requirements of capturing images, displaying images, managing the data format of the image, and getting the image from the capture device to the display are determined. Capturing images expands to (1) centering the light of interest on the image sensor, (2) sensing images, and (3) converting them to digital signals, both of the latter being met in full by the image sensor chosen; displaying images expands to (1) acquired and positioning a display, both relieved by our use of a smartphone common to the users, during production, and common to the developers as needed; managing the data format expands to (1) compression, to permit realistic transfer, (2) decompression, to allow the display to perform its function, and (3) image processing, to shape the data generated by the sensor into the output desired by the project; the transfer of the images expands to (1) initial capture of the image data stream, (2) transitional storage to allow for compression, (3) transmission through the wireless medium, (4) transitional storage to allow for decompression and image processing, and (5) routing the final image to the display. Many of the above dictate a common requirement that can be generalized as power and routing.

The portions requiring extensive attention, beyond that which a single or single set of methods can meet, are centering the light, transmission through the wireless medium, and the power and routing concerns. Our design will attempt to resolve or answer these complex issues and issues related to large points of failure with its earliest architectural choices, while flexibility remains highest.

5.1 System Architecture

Due to the complex nature of the design, the number of domains it crosses, and the intricacy of the algorithms, memory management, and detailed data flow, a simplified data flow diagram, Figure 25, will be the initial point of reference. The diagram's legend, Figure 24, is immediately below.

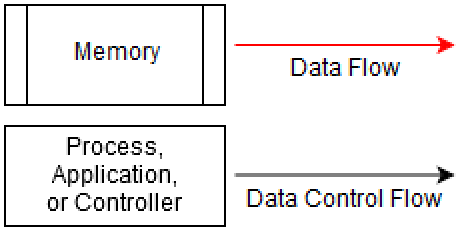


Figure 24 - Data Flow Diagram Legend

The diagram itself elides only one significant step along the data path, the raw 12-bit RGB to YCBCR conversion, which is handled through a simple and highly efficient loop-up table, which requires only a small portion of the available memory at 12 bits = 4096 = 4k memory locations, and the device's included libraries natively support the conversion. Note that YCBCR requires the same two bytes per pixel as the original 12-bit input data, which will be padded and formed into a [0000 | 12'b pixel | 0000 | 12'b pixel] by the DCMI. Additionally, any potential need for in-RAM storage, should performance tests show the need, will be almost free, given that the STM32H753 MCU used in our design has a substantial amount of tightly-coupled memory to ensure swift access for any color translations.

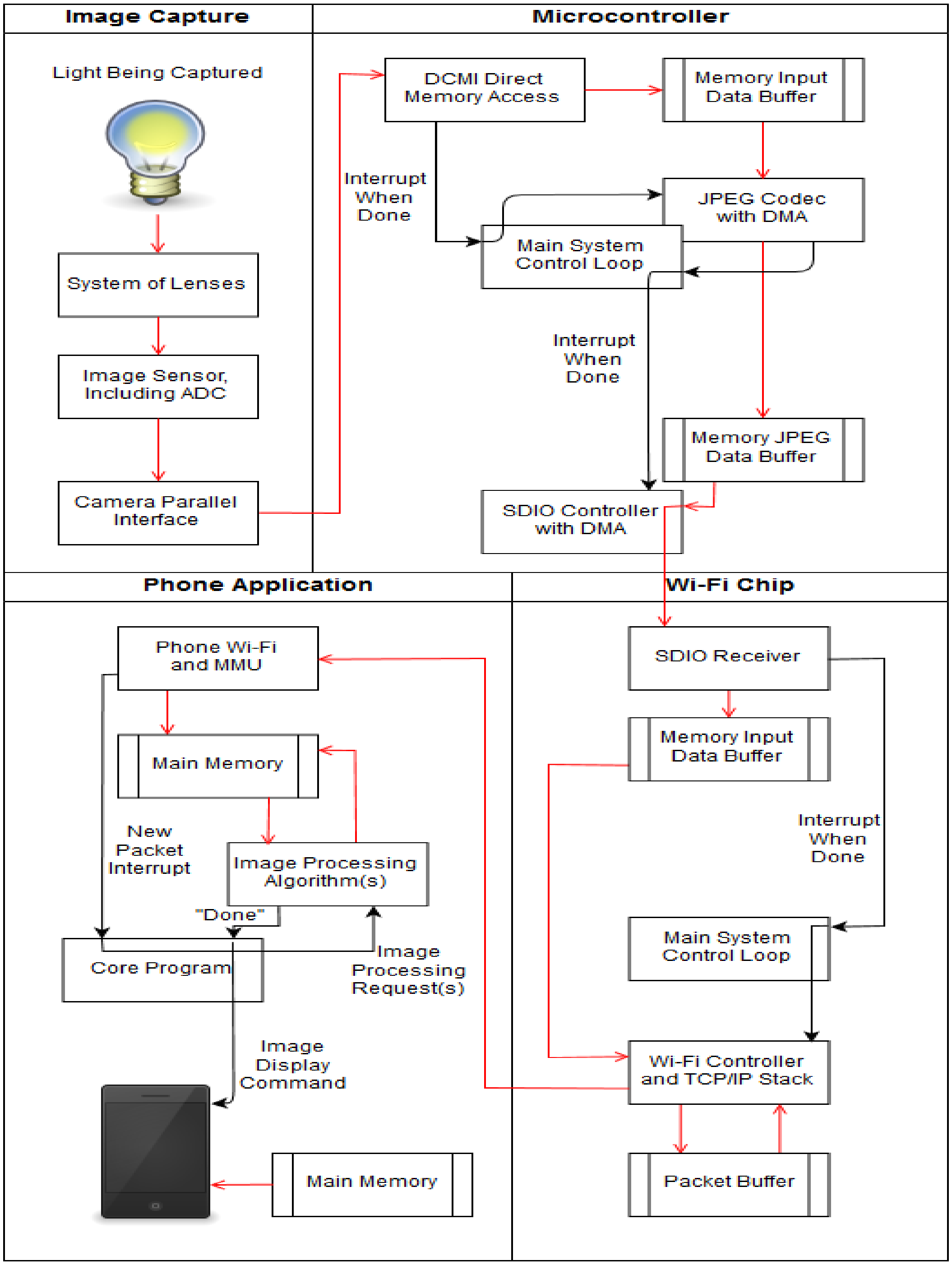


Figure 25 – Data Flow Diagram

The interfaces deserve immediate attention, as bandwidth bottlenecks will drive many design decisions. The image sensor, CPI, and DCMI will all outperform both the MCU and the embedded Wi-Fi chip, so focus can shift to the embedded Arm processor and its integrated JPEG Codec unit. The JPEG Codec requires YCBCR color input, which will not be natively produced by the image sensor. The STM32 documentation contains a software kit that converts the data, however, greatly easing the development process. The following is all speaking to VGA-sized (640x480) images, which are relatedly at the limit of what 1MB of RAM is capable of supporting, so trying to up-scale more than a small amount is unlikely to show meaningful results.

The conversion software, running off of external SRAM, required 58ms to convert RGB to YCBCR, so there is room for improvement following an already-adequate performance. The hardware encoding of the YCbCr to JPEG process consumes an additional 4ms of time. Extremely noteworthy is the capacity of the color conversion to begin working on incomplete images: once any 8x8 block is completely read in, the utility can be called to perform the conversion mid-read; this will effectively merge the image sensor read time and the color conversion window, with the exception of the first eight lines of data, which must be complete before the conversion can start. The time needed for this to occur is minimal, as the image sensor supports windowing, delivering an effective framerate of $\left(\frac{1088\ vertical\ pixels}{480\ vertical\ pixels}\right) * 60\ FPS = 136\ FPS$, yielding an expected delay for seven lines and eight pixels of $\left(\frac{7\ lines + \frac{8\ horizontal\ pixels}{640\ horizontal\ pixels\ per\ line}}{480\ lines}\right) frames * \frac{1}{136FPS} = 107.4$ microseconds. Thus, at a target minimum rate of 10 frames per second, the added delay can be rather precisely estimated at a rounded-up 1.1 milliseconds.

To use the baseline estimates provided above, this process will consume 621ms per each second's window of ten frames. This looks, at first glance, like cutting it close. Re-examining the data flow diagram shows an inherent parallelism between in the operations performed by the MCU: the transfer of data from the JPEG Codec to the SDIO Wi-Fi transfer buffer is DMA-controlled, as is the transfer from the SDIO buffer to the Wi-Fi chip itself.

The SDIO interface itself is of the 4-bit parallel variety, supporting a maximum 40MHz clock rate on the slave-only receiving end, with a 512-byte receiving buffer. The clock rate will be reduced to 30MHz for reasons of overprovisioning and convenience of the STM MCU's clock rate generation. Early manipulations of the STM32Cube's configurator show a very clean, powers-of-two divided path to a 30MHz SDIO clock signal, which is ideal.

At present time, runtime tests have yet to be performed on the transmission rate supported by the Wi-Fi chip, but the device goes well out of its way to support high-frequency (by embedded standards) 4-bit SDIO as a primary data transmission interface over methods like SPI and, less significantly, UART. Even if the link is only available at a presumable underestimate of 10% of the time, $\frac{.5bytes}{transfer} * 30M\,\frac{transfers}{s} * \frac{1}{10}$ yields a healthy 1.5 MB/s, which looks suitable for 10 FPS VGA resolution JPEG, which has shown an average filesize of 90-120 KB in our own testing, though worst-case attempts such as noisey fractals have produced files in the 230KB range. Testing of microscope images is planned upon completion of our test rig.

The chip is also known to support Wi-Fi transmission rates exceeding this speed. This pends testing in high-spectrum-utilization environments, but unrelated experience as a Wi-Fi user lends itself to optimism in this regard.

Once the packet makes it to the receiving app, the multiple-gigahertz domain is trusted to bring it the rest of the way to the screen in a timely manner.

5.2 STM32H753ZI Microcontroller Operations and Details

The STM32 will serve as the master of the image sensor, Wi-Fi chip, and integrated JPEG Codec, which is itself a significant device and effectively a peripheral with direct memory access. This device will be in charge of initialization of these devices and more internal elements.

For a brief overview of the relevant, used features of the chip, see Table 9 below.

Name	Amount	Speed	Other Notes
Arm Cortex-M7	1	400MHz	Direct JTAG access ports
SRAM	1MB	192KB of 0-wait tightly-coupled memory	Very well-organized along multiple buses and perfect sizing for required memory zones
Flash Memory	2MB	100-200MHz	Full Read-Write, with optional protected regions for bootloader code and critical sections
DMA Controllers	4	200MHz multi-channel	Features one Master DMA with link-list support
DCMI	1	Max of 80MHz	Supports 12-bit parallel data, and integrates superb DMA
JPEG Codec	1	4ms per VGA frame encoded	Fast enough, and with DMA, to never be a bottleneck

Table 9 – Microcontroller Feature Set Overview

5.2.1 Development Environment

The STM32H series has broad support: a low- and high-end development kit, an extensive, well-documented Hardware Abstraction Library, an SDK with a configuration wizard for every port, included peripheral, and interface currently being investigated for incorporation in this project, including a full clock tree setting GUI, an Arm MDK, and a host of software examples and pre-sets.

5.2.1.1 STMCube32

The STM-developed Cube package contains dedicated HALs and example software for three development kits, every feature planned for use in this project, including (M)DMA, DCMI, JPEG Codec, and pre-made software to perform functions related to device features, like the RGB to YCbCr conversion software described in Section 5.1.

This project will be developed using all tools named above. The base code will be generated through the Cube platform, the provided color conversion algorithms will be used barring unforeseen timing conflicts, and supported configurations will be made through the graphical wizard for its user-friendly and less-error-prone benefits.

The HAL code will also be referenced and studied to further device architecture familiarity throughout development.

5.2.1.2 The NUCLEO-H743ZI Development Kit

The H753ZI and H743ZI, which our development kit is built around, differ only minorly across all features. As they also differ minorly in pricing, the 753 was chosen for inclusion in our device, mainly to maintain the option of including the security features included natively in the 753. The 743 will serve as a functionally-equivalent centerpiece of a development kit, and this board was chosen as the other is not affordable within the context of a student-funded senior design project.

This kit benefits from a supplied set of premade examples, and the Wi-Fi chip chosen is commonly paired with this family of MCUs to the point that there is dedicated cross-use reference material between its development kit and this development kit, which will aid our development by enabling an early interfacing between the Android application and the MCU code in a real, live test environment.

While this kit does not support the existing firmware example for JPEG encoding, it does support a limited set of essential DMA and MDMA example programs, the two of which will be the real workhorse of the MCU's data routing.

5.2.1.3 Arm's Keil MDK

Keil's uVision IDE will be used to bring together setup code generated by the STM Cube setup manager, the example program for JPEG's first step of conversion, and the code to be developed in the IDE for this project. Interfacing with the device will be handled by the ST-LINK/V2 programmer/debugger, which has strong support by Keil (and Arm in general).

The NUCLEO development kit also has native support within this MDK, which will form the test environment for in-development code.

5.2.2 Memory Management

The program's memory management will primarily involve dealing with the SRAM, bouncing the DCMI between empty buffers, and caching RGB code before overwriting it with freshly-calculated YCbCr values.

The DMA, which more than merits its own subsection, will primarily be in charge of (1) transferring image date from the sensor to the memory buffers used for RGB and YCbCr data, (2) supplying the JPEG Codec with fresh data as needed, (3) transferring the converted JPEG file from the encoder's output to the pre-SDMI buffer, (4) receiving messages from the Wi-Fi chip, including any control transmissions from the paired application, and (5) sending necessary interrupts to the MCU at times when, for example, the DCMI has read a frame and should be disabled, or the JPEG Codec is finished, and the input buffer is ready again to read an image.

5.2.2.1 Direct Memory Access

The 7x3 family of microcontrollers has a wide array of DMA options, including multiple channels of priority queues, separate DMA channels, and four DMA managers, one of which can reach and modify the same-cycle tightly-coupled memory of the main microprocessor. These are configurable through the Cube development framework, and they will be used to route the vast majority of the I/O and between-memory data movement, as the images (both input and output) will require more space than a single chip of the MCU's memory can provide.

5.2.2.2 Managing Uncompressed Single-Frame Transfers

Among the most demanding features we're attempting to support is a raw image transfer from the embedded system to the phone application. The specifics, at least, are clearly defined, though the process will almost entirely distinct from the video streaming mode, which necessitates the development of entire algorithms and protocols to support it.

That said, still imaging is fundamentally the deepest feature we can offer at the imaging level. Video streaming is our project, but to have a ~1920x1080 image sensor under 40x optical magnification and limited the output to down-sampled or windowed 640x480, lossily-compressed images would be leaving a large opportunity on the table, which is not our intent.

The limitations are rather severe. A raw 1920x1080 feed implies a roughly 2M pixel count, which places us beyond capacity in RAM, and at 2/3rd capacity raw storage on the entire chip, and that's under the incorrect assumption of one pixel being one or less than one byte. In actuality, the raw size of one full frame approximately fills our entire storage capacity, and many potential solutions can be dismissed outright.

The Wi-Fi link won't be able to help at the necessary rate, given we are receiving a 3-megabyte file in 16.67 milliseconds. Its own memory is limited to roughly 256 kilobytes per core – one application core and one networking core, the latter of which must dedicate a large, static portion of its memory to the TCP/IP stack and physical layer drivers. Whether or not transmitting 100 kilobytes of data during the frame read process is viable will be treated as unreliable and ultimately inconsequential.

Down-sampling the 12 bits to 8 bits is entirely unacceptable, as it defeats the entire purpose of this design space.

A memory expansion port would enable this very well, but it would add overhead to product cost, complexity, implementation difficulty, timing requirements, and synchronization which are not desired unless absolutely necessary.

Partial encoding sounds intriguing – putting the frame of the picture through a JPEG conversion to save some memory – but the process is too slow, and the memory saved is not quite sufficient to enable a frame to fit naturally, even ignoring the partial breakdown of our objective, though the impact of JPEG compression on natural images is not too large. The format mainly earned its infamy for its extremely poor quality reproduction of text, especially text on a static color, due to the way it pulls apart sets of pixels to form coded units.

One very promising method exists to compress the file size without losing any data or incurring relevant performance overheads, but it is insufficient to meet the need. Recalling that the raw RGB data is stored in 32-bit words of the following format: [0000 | b12,b11,…,b0 | 0000 | b12,b11,…,b0], it would be a simple task to shift the data to this format: [b11,b10,…,b0 | b11,b10,…,b0 | b11,b10,…,b4] + [b3,b2,b1,b0 | b11-b0 | b11-b0 | b11-b8] + [b7-b0 | b11-b0 | b11-b0], repeating every three 32-bit words. As each word of memory would go from storing two pixels to storing two and two thirds, our memory cost goes from $(1920 * 1080)\ pixels * \frac{4\ bytes}{2\ pixels} = 4\ MB$ down to $4\ MB * \frac{2\ pixels\ per\ word}{8\ pixels\ per\ 3\ words} = 3\ MB$. This is almost manageable, as there is a total of 3MB of combined flash plus SRAM on the MCU, but some sacrifice must be made as our total memory naturally cannot be committed to a single file. The cost of fixing the data on the receiving end would be low, but it is worth keeping in mind as a downside to this approach. The process would take more effort for the microcontroller, of course, but its clock cycles would be free in terms of overhead, as they would operate at the same time as the reading process, with the assistance of the MDMA controller feeding in new pixel data directly to the tightly-coupled memory.

Our most realistic solution so far is a compromise. It's believed that all standard-operation flash memory usage will be able to fit in a hard-limited 500 kilobyte space, including boot memory, as the needs of the system are rather narrow in terms of programmatic memory cost. Further, more than 500 kilobytes of SRAM are made available to encoding during the video capture mode, which is free to be utilized for the storage of the uncompressed image during single capture mode.

Combining the above two options, if fully 2 megabytes worth of spaces are capable of being committed to a single application of runtime use, then two thirds of the raw image capabilities of the image sensor can be captured using this shift-and-store compression of the 12-bit image data. Because vertical pixels are naturally more limited, because, relatedly, shaving horizontal pixel resolution lowers the data density fastest, and because the expected viewing room on a modern Android mobile device can fit 1080 pixels in either direction, the choice naturally favors imaging a horizontally-narrowed portion of the image.

Tests will be performed to determine whether more than 500 kilobytes of SRAM can be committed to this process. Leaving one 125 kilobyte buffer as the output stream to the Wi-Fi chip and committing one 125 kilobyte portion of tightly-coupled memory to microprocessor operations, with an additional, nearly-negligible kilobyte of space to inbound transmissions from the mobile application should be viable, leaving slightly more than two thirds of the full image.

5.2.3 Included Libraries and Initializations

Some components of the device come with pre-developed, pre-tested solutions for peripheral operations, most notably the JPEG Codec, the RGB to YCbCr transformative step necessary for the JPEG Codec, and the DCMI.

The first and last of these have been explored and seem nearly-optimal for our purposes, so their integration is desired. The JPEG Codec supports VGA resolutions, which also fits a memory allocation well-suited to the microcontroller – roughly 300 kilobytes of raw image data, approximately 100-150 kilobytes of converted JPEG data, and enough room to buffer for additional parallelizable operation.

The RBG to YCbCr library was optimized for breadth of operation rather than efficiency, so it will be used only for initial testing and validation purposes.

5.2.3.1 DCMI Setup

The MCU's Digital Camera Interface (DCMI) allows autonomous reception of the image sensor output, natively controlled by the provided Line Valid (or Hsync), Frame Valid (or Vsync), and Pixel Clock of the sensor.

This interface also has direct interfacing with the MDMA controller, which allows for fully-automated, programmable reading of each image without processor intervention. This will be setup to allow the processor to commit all of its time during image reading to the start of the RGB to YCbCr conversation process, which has been determined to be the critical component of time cost in this unit's operation.

5.2.3.2 RGB to YCbCr Conversion

The JPEG format takes advantage of the relatively higher sensitivity to luma values and the relatively lower sensitivity to chrominance values of an image, compressing the color components more heavily than the brightness component. Thus, the first step of converting the image to JPEG is transforming the raw data to YCbCr.

This is performed most often mathematically, as the time cost is relatively low – a few multiplications, bit shifts, and additions per sub-sampled value per pixel, but the requirements of this embedded image processing environment make that too slow. Instead, a tabulated form of all possible computation results will be stored in tightly-coupled memory, as the required memory space is small at only 2^12 = 2048 input combinations. This should achieve significant performance improvements over the included library meant for YCbCr conversion, which utilized off-chip ROM.

5.2.3.3 JPEG Codec Setup

The JPEG Codec has a well-documented HAL which will be set up and used without need of custom modification.

5.2.4 Parallelizable Data Flows

As the microcontroller and generally the embedded system are the bottlenecks of the system, optimizations must be made where possible. A visualization of which parts of the MCU's operations and behavior conflict with each other, Figure 26, is presented below.

Some considerations remain outside the scope of this perspective; these pend implementation testing. The achievable rate of RGB to YCbCr conversion, if high enough, may allow for buffering of the next image in on-chip flash memory, waiting for the Wi-Fi chip to transfer the former image.

Alternatively, if the transmission speed of the Wi-Fi chip is outperforming the conversation rate of the rest of the system, which will be variable based on Wi-Fi congestion and channel availability, the conversion may be allowed to start while the data is nearly sent in its entirety.

Furthermore, if the JPEG images have been consistently sized under the 128 kilobytes of allocated output buffer space, the next conversion may be started early, providing a backup allocation of memory in another location, likely either TCM or flash, to prevent buffer overflow and data loss.

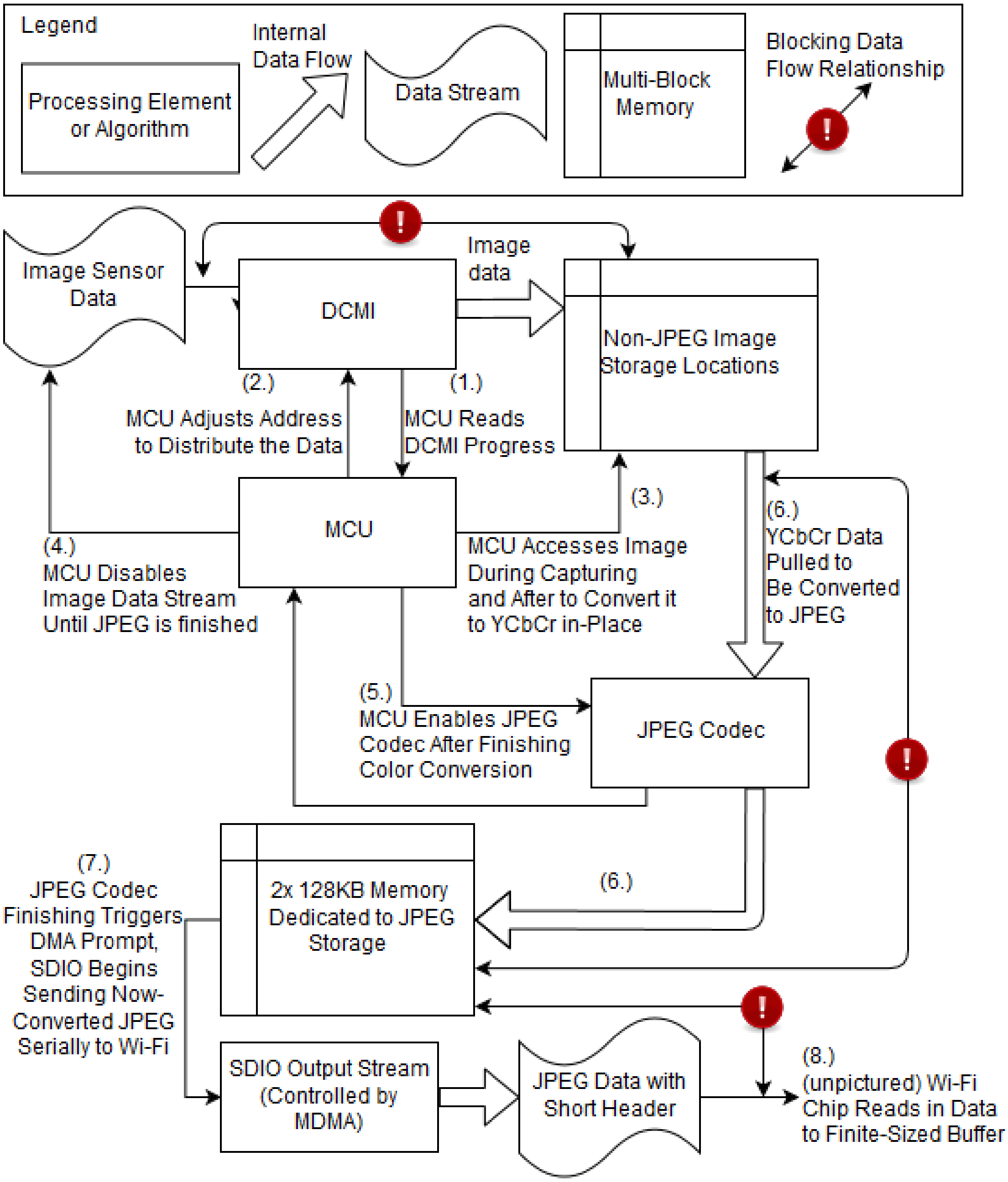

Legend
Processing Element or Algorithm
Internal Data Flow
Data Stream
Multi-Block Memory
Blocking Data Flow Relationship
!
Image Sensor Data
DCMI
Image data
Non-JPEG Image Storage Locations
(2.)
MCU Adjusts Address to Distribute the Data
(1.)
MCU Reads DCMI Progress
MCU
(3.)
MCU Accesses Image During Capturing and After to Convert it to YCbCr in-Place
(4.)
MCU Disables Image Data Stream Until JPEG is finished
(6.)
YCbCr Data Pulled to Be Converted to JPEG
(5.)
MCU Enables JPEG Codec After Finishing Color Conversion
JPEG Codec
(7.)
JPEG Codec Finishing Triggers DMA Prompt, SDIO Begins Sending Now-Converted JPEG Serially to Wi-Fi
2x 128KB Memory Dedicated to JPEG Storage
(6.)
SDIO Output Stream (Controlled by MDMA)
JPEG Data with Short Header
(8.)
(unpictured) Wi-Fi Chip Reads in Data to Finite-Sized Buffer

Figure 26: Parallelizable Video Data Flow

Very significantly, the most time-consuming processes at the MCU side, the color conversion and the SDIO transfer, can be handled simultaneously with the aid of a thoroughly-designed MDMA protocol. One important element left out of the diagram for brevity is a re-routing scheme for tail pieces of any JPEG frames which would exceed the memory assigned to them. There is ample space for this, however, and unless timing needs dictate otherwise for a reason that is beyond present understanding of the hardware, the fallback storage location will be a dedicated portion of the on-chip flash memory, as it would otherwise be assigned to be used for full-resolution image transfer, which is necessarily not happening under video transfer conditions.

The presumed standard working condition of the MCU, given the above architecture and previously-elaborated timing needs of JPEG encoding and variable-rate SDIO transfer, is a reading-and-color-converting process in the upper portion of the diagram with an active DMA thread routing JPEG data to the hosted Wi-Fi chip.

Optimizations, therefore, should target these stages exclusively. An attempt will be made to optimize the color conversion step through the following means:

First, the RGB to YCbCr conversion tables will be stored in flash memory. When operating in video mode, the MDMA will load in the tables to the processor’s Tightly-Coupled Memory. This must necessarily be the master DMA thread, as no others have access to that memory space. As data arrives in sufficient number for processing, MDMA will bring the first 8x8 array of pixels to the TCM for optimally-efficient access.

As the 8x8x2 (x2 because each pixel occupies a two bytes space) = 512B data is being expanded to the Y, Cb, and Cr arrays, the MDMA will bring the next 8x8 array to the other memory space in the TCM, as follows: Working Space 1 ([8x8x2 RGB] [Y output space] [Cb output space] [Cr output space]) will be utilized initially, while a mirror image of memory space, Working Space 2 ([8x8x2 RGB] [Y output space] [Cb output space] [Cr output space]) will be populated. As soon as the MCU finishes in Working Space 1, it will raise a flag, alerting the MDMA to its status.

The MDMA will naturally move the computed block to its designated memory location in the uncompressed image memory space, while the MCU begins processing Working Space 2. As the MDMA is simply moving data while the MCU is processing complex data, the MDMA will be able to fetch the third 8x8 array, again to Working Space 1. Through these means, the processor will spend the entire portion of color conversion time at full throughput. This will tie the processor up for potentially tens milliseconds, after which it will be free to check for any new incoming messages, interrupt flags, etc., and there are no outside processes which require a faster than 20Hz response. The full 50ms unoptimized wait time incurred by the HAL conversion method would not adversely affect user experience.

5.3 Microscope Design

The microscope design section will discuss in detail the design of our microscope. It will explain the necessary background for each design, and how we decided on the design. This section starts with the microscope optical design where each optical design element will be explained in great detail.

Following the optical design section, there is a microscope body design section. This section is dedicated to how we will build the body of the microscope, the reasons behind why we designed it the way we did, and the material used. It will also discuss how all of the individual components will fit together. This portion of the paper ends with the image sensor section, in which we discuss the necessary details of how the sensor will work in conjunction with the other components, as well as why we chose that specific sensor.

5.3.1 Microscope Optical Design

This section covers all of the optical design elements of the portable microscope. It will cover in detail the illumination system, light sources, apertures, the objective lenses, the overall resolution of the system, cover slips, and tube lenses. The illumination system is vital to our project because it determines the amount of light allowed into our system. This can saturate the sensor or not have enough light to get an image at all if it is not designed correctly.

The light source section will discuss the options available to use as a light source. This will determine what our sensor will pick up. The objective lenses are the base of the entire design. These need to be able to magnify a specimen and project it onto a sensor in such a way that the sensor will read a high-resolution image.

The overall resolution will discuss what the theoretical resolution of what our system is capable of. Cover slips are a key component of any microscope because all current market objective lenses assume that they will be used. The final discussion is in regard to tube lenses because almost all microscopes today use them.

5.3.1.1 Illumination System

As stated previously, the illumination system is a key component to any microscope. It determines the type of microscopy that will be used as well as the light that will go through the entire system. The illumination system much be such that the specimen is not overly saturated in light as it can cause too much reflection and in turn cause the sensor to be saturated.

The end result of a saturated sensor is a giant bright blur spot and no resolvable features. To avoid this, our illumination system has been designed with careful attention to detail. This section is going to go into detail about what is currently used in the field of microscopy and what we actually utilize in our system.

5.3.1.1.1 Our Illumination System vs. Traditional

Brightfield microscopy typically utilized Koehler illumination, however due to the design of our microscope this is not an option. As shown in Figure 27 below, our illumination system includes both a back light and a top light.

Typically, a condenser is used to gather all of the rays emanating from the light source and condense them into one bundle to evenly illuminate the specimen. This helps to get a clear image and because our design does not include a condenser, Zeiss microscopy says that our resolution will be halved. However, we have discussed even illumination and to make it happen.

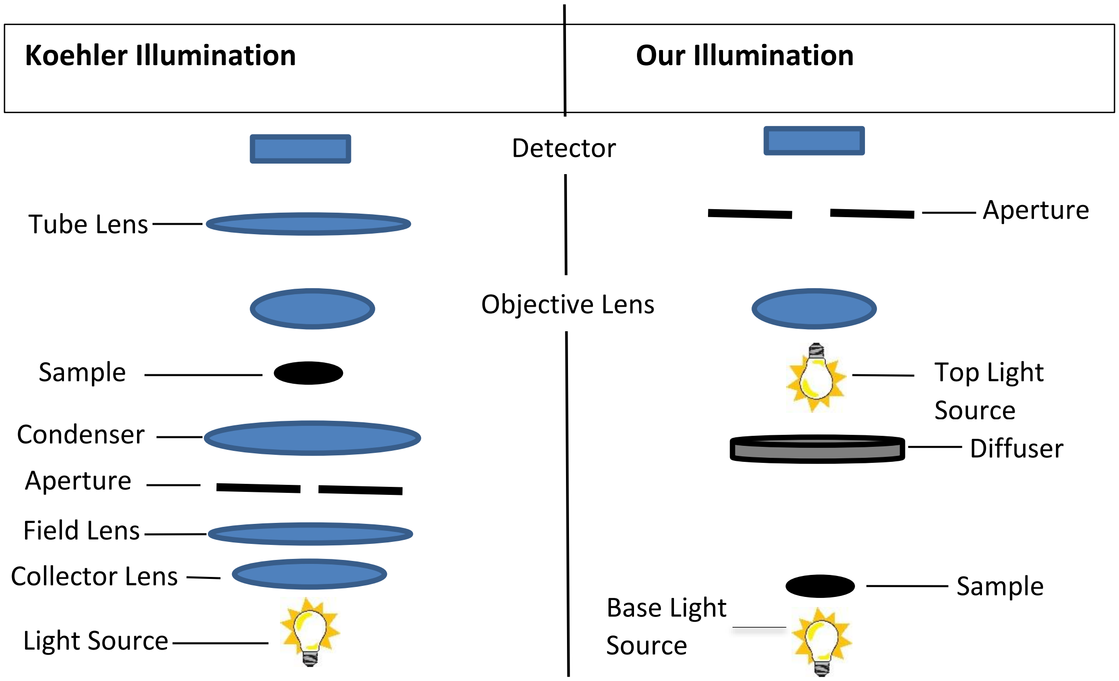


Figure 27 – Illumination Comparison
A comparison of Koehler illumination to our illumination.

5.3.1.1.2 Ideas for Even Distribution of Light

We thought about using prisms to redirect the light to one focal point, but we have multiple LED's. Each LED would require its own prism which is impractical for two reasons. The first is that it would cost too much. The second problem is that it would take up too much space and make the tip of our microscope quite bulky. In essence prisms are just too impractical. We then discussed lenses to focus the light, but a large lens would be in the way of the objective lens and obstruct the specimen. If we tried to use multiple small lenses at an angle, it might do what we want however the cost of that is simple too much. Also creating enclosures for each lens would create a lot of unnecessary work. The next option was to use lasers because of their inherently narrow beams, however the issue with lasers is the power consumption and scintillation. We would have too much reflectivity with a laser, and we would have needed multiple lasers. Having multiple lasers would have made the microscope bulky and use a lot of power. The other problem with lasers is the narrow spectrum that is produced, to get a full color image would be impossible because there are not any white lasers. In the end we have decided on using something akin to a ground glass diffuser so as to evenly distribute the light onto our specimen.

5.3.1.1.3 Diffusion Filters

The diffusion process eliminates bright spots and distributes the illumination in an even pattern so that our whole specimen gets an equal amount of light. This works by scattering the light in an even distribution as show in the Figure 28 below.

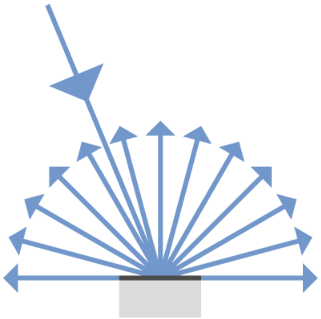


Figure 28: Lambertian Distribution

This demonstrates the behavior of the light on a Lambertian surface.

This is known as Lambertian scattering, which is from Lambertian reflection. Lambertian reflection is a property of the material that creates the same apparent brightness regardless of the viewing angle. A perfect diffuser will create this Lambertian scattering effect so that the illumination is completely independent of the angle. There are many different optical diffusers that utilize this effect. The best one is a holographic diffuser which is made by holographically etching on a polycarbonate material. The pattern used for etching is completely random. This will give the best Lambertian distribution, shown in Figure 16. However it is very expensive to buy this type of diffuser. It is also not readily available in the shape and size that we need, nor do we have the means to make our own. Ground glass is another option, it is a diffuser that is made by sandblasting a piece of glass to create a very rough textured surface. We are not able to make our own ground diffuser as we cannot cut glass to the size and shape we need, nor can we manufacturer or own glass. Due to these problems we will be making our own diffuser out of a semi-opaque plastic. The plastic is readily available, cheap, and easily customizable. As shown in Figure 29 below, our diffusion filter evenly distributes the light.

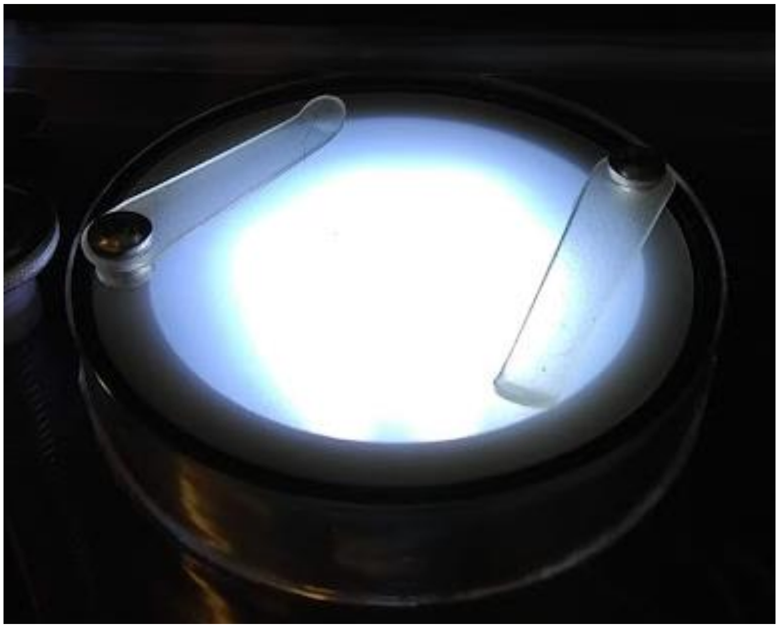


Figure 29 Even Distribution of Light using Diffusion Filter

5.3.1.1.4 Light Position

Our microscope also illuminates the specimen from in above instead of underneath as with a typical microscope. Most microscopes employ the use of a back light, but our design is for portability and ease of use, so we have decided to illuminate from above the specimen. The drawback of this design means that the light will not fully illuminate the sample, so samples with very low contrast will have even less contrast. For our purposes of using this microscope as an introductory teaching device, it will be sufficient. We will still be able to image an onion cell that is approximately 75 um wide and 120 um long, however we would not be able to image a human cell.

5.3.1.1.5 Controlling Amount of Light

Most microscopes also have an aperture in place that limits the amount of light that ravels into the system. These apertures can sharpen the edges of what is being imaged by reducing the number of rays that pass through the periphery of the lens. This can also reduce the amount of spherical aberration. As of right now we are implementing a single aperture. We are also using a switch to control our LED’s. There is a setting that is bright, and brightest. This will be accomplished by having three to six LED’s of each type. We have also implemented a base light that utilizes seven LED’s in a circular arrangement, covered with a plastic diffusion filter. We will be tested this method and it does produce enough resolution to image an object of 120 um long and 75 um wide. The reason for having multiple settings is so our image is not saturated with light. Some objects will reflect more light than others which can cause saturation of the sensors, so the multiple light settings allow us to get around this problem. We found that simply controlling the brightness was not enough to keep our sensor from being saturated, so we will introduced an aperture to our system to further reduce the illumination.

5.3.1.2 Light Sources

In a brightfield microscope, the light source is usually tungsten and halogen lamps or mercury and xenon arc lamps. The tungsten halogen lamps have a very broad spectrum, are relatively cheap, and are very reliable. These lamps are very good for brightfield, photomicrography, and digital imaging of stained cells, when used in conjunction of a ground glass diffusion filter. They generate a continuous distribution of light across the visible spectrum and can be used for extended periods of time with minor fluctuations. The drawback of this type of light source is the majority of light is lost in the form of infrared, so most of the energy created is wasted. The mercury and xenon arc lamps are more specialized due to being 10 to 100 times brighter than the tungsten and halogen lamps. They are very intense light sources but at very specific wavelengths and are excellent for fluorescent imaging. The main drawbacks of this type of lamp are alignment is critical, they have a much shorter lifespan, they have a risk of explosion, and they cost more.

For our microscope we decided on using LED’s due to their cost and ease of use. We have chosen 5 mm clear white LED’s. The white LED’s use very little energy, have a very broad spectrum, and a very low cost. Using these in conjunction with our faux ground glass diffusion filter will provide us with a reliable light source across a broad spectrum that will allow us to see the specimen in full color.

Our LEDs emit light at a solid angle of 20 to 30 degrees, in order to get the most light out of our LEDs we left them exposed and put a skirt around them for the top light. The idea behind this is to shine the light as directly as we can onto the sample. For the base light we used a diffusion filter to evenly distribute the light to effectively illuminate our sample. All of this is shown in the body design section.

5.3.1.3 Apertures

We designed our microscope to have an aperture 6 mm in front of our sensor. This was to make our tube 1 mm longer and also to mount the sensor to the tube. The aperture was added to our system to help limit light so that our sensor was not saturated and could clearly resolve the specimen that we are studying. The reason that the resolution became clearer is because by limiting the amount of light that reached the sensor, it improved the contrast, which in turn made objects easier to see. The cad design of the aperture is shown below in Figure 30.

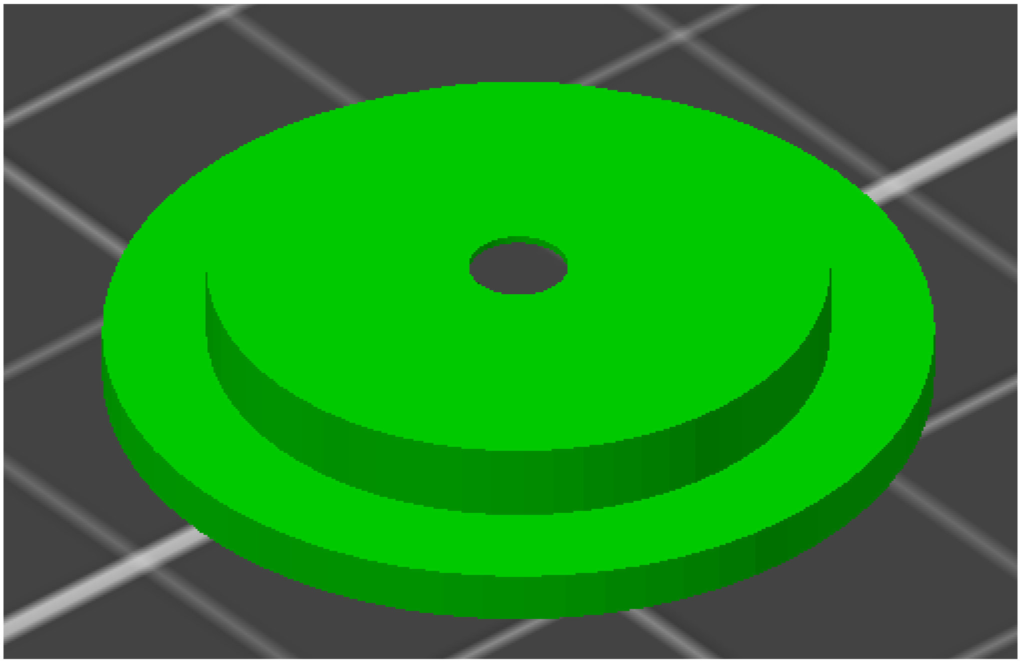


Figure 30: Aperture- CAD design of Aperture. This design measures 4 mm from top to bottom and has a 1 inch diameter. The aperture itself is 3 mm in diameter.

5.3.1.4 Objective Lens

Objective lenses are arguably the most important part of a microscope, they limit the field of view, they determine the resolution, and they determine the spectrum that can be detected. The field of view and resolution will be discussed in a later section. The spectrum that can be detected depends on what wavelengths that the objective lens has been corrected for. Objective lens is an interesting term because it implies one lens, however an objective lens is actually a complex system of lens that work together to magnify a specimen as shown in Figure 31.

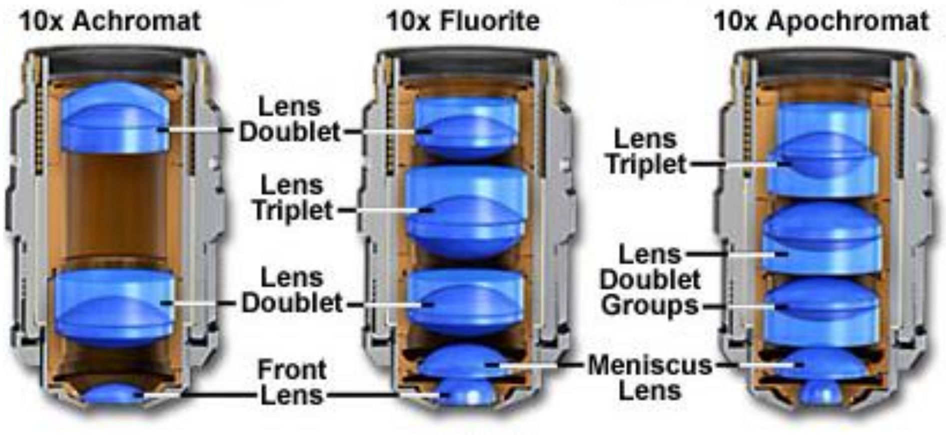


Figure 31 – Objective Lenses
The most common types of objective lenses and their lens layout.

These objective lenses are all refractive lenses and are the most common type of objective lenses. They all have multiple lens that work in a relay to produce an image, but some of them have more lenses than others. Typically, all of the inner lenses get coated with an antireflective coating. This reduces back reflection and enhances the overall illumination.

The objective lens that we are using are made of several smaller lenses that work in unison to create a magnified image. The lenses are made of convex and concave lenses, as well as having different refractive powers. The convex lenses have are converging lenses that take focus the incoming bundle of light rays into one singular image. They are thinner at the edges and thicker in the middle.

The concave lenses are diverging lenses that take incoming rays and diverge them. They are thicker at the edges and thinner in the middle. The shape and refractive indices of the lenses are how they guide the incoming rays. Having a thicker edge with a higher refractive index in air, means that the light will be refracted more than in the center. This is because the light has further to travel and is slowed down more than at the edges. The convex lens works in the same way, but since the thicker part is in the middle that is where the most refractive power is. A diagram of these types of lenses are shown in Figure 32 so that it is easy to see the refractive powers of each type of lens.

Both types of lenses are within the objective lenses. To make a doublet lens, two lenses are combined by cementing them together. One lens is concave and the other is convex, and depending on the type of effect that is needed, the material will be picked with a refractive index to that effect. This will be discussed more in the chromatic and spherical aberration section.

Some lenses are aspherical, meaning that one side is curved but the other side is flat. This will have the effect of collimating the bundle of rays. This works by placing an object at the front focal point where the rays converge, and then those rays will pass through the lens and travel in parallel to the next lens. The next lens can then take the parallel rays and refocus them as needed.

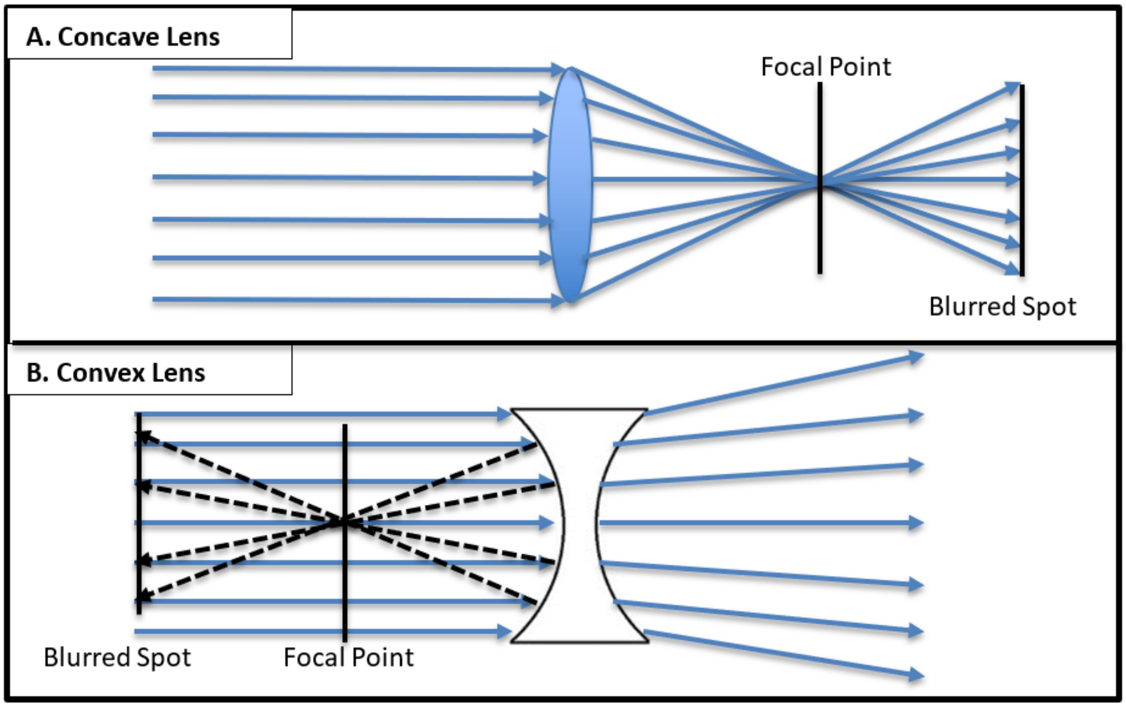


Figure 32: Lens Types-
A diagram of a convex lens and a concave lens. The Concave lens is diagram A. It demonstrates the converging powers of a convex lens. The closer to the center of thickness, the more refractive power the lens has. Diagram B shows a concave lens and how its refractive powers work. It is shown that the closer to the edges, where the glass is thickest, the more the rays are diverged. Both diagrams show where the image is being formed by showing the focal point. The blurred spot is where an image is still being formed, however it is now out of focus.

5.3.1.4.1 Achromat Objective Lenses

The achromat lens as mentioned earlier only has chromatic aberration correction for two colors and it also has some spherical aberration correction. The standard for achromatic lenses requires the middle 60% of an image to be corrected for spherical aberration. It also has been corrected for chromatic aberration for red and blue. It is the simplest type of objective lens and only has three to five lenses in the system depending on the level of magnification. The chromatic aberration for red and blue is typically corrected by using a crown flint combination that is known as a doublet or an achromat. Since the dispersion caused by flint glass is approximately twice that of crown glass, if a negative flint lens is cemented to a positive crown lens the combined dispersions will be almost equal but opposite thus eliminating the chromatic dispersion and increases magnification. It specifically corrects for 486 nm and 656 nm, and if the correct combination of refractive index, curvature, dispersion and thickness are used it will bring the two waves into a common focal plane. The combination of a positive and negative lens also reduces the spherical aberration. These types of objective lenses are the most commonly used because of the cost and good resolution of most things. To get better resolution and less aberration, you need to go to the next type of objective lens.

5.3.1.4.2 Fluorite Objective Lenses

The fluorite objective lens is the next step up and it contains anywhere from five to nine lenses. This type of lens is usually used for low level emission specifically fluorescence. It is made with fluorspar lens combined with another lens to form a doublet that is achromatized at three specific wavelengths. This is the mid-level lens, somewhere between achromat and apochromatic. These still have a considerable price jump from the achromatic lenses, they start at a couple hundred dollars apiece.

5.3.1.4.3 Apochromatic Objective Lenses

The apochromatic objective lens can have anywhere from nine to eighteen lenses and has two doublets, and a triplet lens. These lenses work together and correct for spherical aberration and chromatic aberration for up to four wavelengths. One or more of the lenses within the doublet or triplet contains fluorspar. It can bring red, blue, and green into a single focal plane with only a minute amount of chromatic aberration. These apochromatic objectives are very expensive starting at around one thousand dollars. They are used in very high-end research labs and are not used by the average user.

There are many more types of objective lenses that we did not consider because we are doing brightfield microscopy and the refractive objective lenses are ideal for that purpose.

5.3.1.4.4 Optical Design Choice

There were many lenses to choose from, and the absolute best objective lenses were plan fluorite, however they were far too much for our budget. We purchased some very cheap objective lenses from Amazon for testing with. They were so cheap, that they did not have any specifications or diagrams with them. We discovered that they were able to work within our system, so we elected to not buy the more expensive lenses for two reasons.

First the more expensive lenses were of a higher quality, however they were priced from $100-$250 and we already had lenses that were only about $12.50 apiece. It seemed wasteful to spend an additional $350.00 when the lenses we purchased worked for our purposes.

Second, the more expensive lenses were infinity corrected which meant that they would also require additional lenses. At the minimum the additional lenses would cost $20 each. Again, this seemed unnecessary since we already obtained perfectly good lenses. If we are unable to get a high resolution image with what we have then we will revisit the idea of changing lenses.

For our project, we had really wanted to be able to image in at least three different spectra, so we wanted a fluorite or higher objective lens. Our budget was unable to accommodate that, and so we went with the achromat lens. Unfortunately we were unable to do multi spectra due to our sensor.

Our particular lenses should be more than sufficient to image 120 um and up. We have chosen to go with a set of two achromatic objective lenses made by AmScope. We did attempt to get the lens diagram from them, however we were told they do not have that information. However, from the label imprinted onto the lens we were able to ascertain the numerical aperture, mechanical tube length, cover slip thickness, and magnification levels. Table 10 shows the information.

Selected Objective Lenses		
Magnification	40 times	10 times
Numerical Aperture	.65	.25
Coverslip thickness	.17 mm	.17 mm
Mechanical Tube Length	160 mm	160 mm

Table 10 – Selected Objective Lenses

Based off of all the research we have done, achromat objective lenses are made basically the same on the inside. The differences between the different manufacturers lies in the working distance, mechanical tube length, and if they are infinity corrected or not. The ones that are infinity corrected require a tube lens, but the ones that go by a standard are not. Another thing to note is that our microscope objective lenses are Duetshe Industrie Norm or DIN, this will become important when we get to tube lenses.

There is a coverslip thickness included in the specifications of the objective lens, this is because all objective lenses are created to be used in perfect harmony of a certain thickness of coverslip. We will discuss how this will affect our overall image in the resolution section.

5.3.1.5 Resolution, Field of View, and Focal Length

This section covers the most important parameters of any optical system. We will discuss at length the resolution, field of view, and finally focal length. The resolution will be discussed as it relates to our system and how we calculated the theoretical resolution of our system for each type of light we will be using. The field of view will be discussed as it pertains to our system. We will give a brief description of what it is and why it is important. We will end with the theoretical field of view that should be obtained using the lenses we have obtained. The final part of this section is about the focal length, this is very important to get a good image. The focal length is key, because all lenses have a focal point that the refracted rays of light will converge to. If the length is wrong it will be out of focus and the image will be unresolvable.

5.3.1.5.1 Resolution

Resolution is defined as the smallest resolvable detail. It is the defining feature of any optical system. What this means is that there are two lines, how far away from each other do they need to be in order for them to be seen as two separate lines. It is not unlike aliasing in imaging or overlapping signals in electronics. The two things need to be far enough away from each other to be seen as individual details.

For our optical system, we can see an onion cell. Since it is impossible for us to get a solitary onion cell, our system is able to distinguish detail that is the size of an onion cell. This is approximately 75 um. To determine what the resolution of our system is and if it will in fact be able to resolve that fine of detail, we did many calculations.

The resolution was figured out using the equation

$$d = 1.22\frac{\lambda}{2NA}$$

Given that our lambda can range anywhere from 400 nm to 850 nm, we have created a chart, Figure 33 below, to demonstrate the best to worst resolutions we can achieve with our chosen levels of magnification.

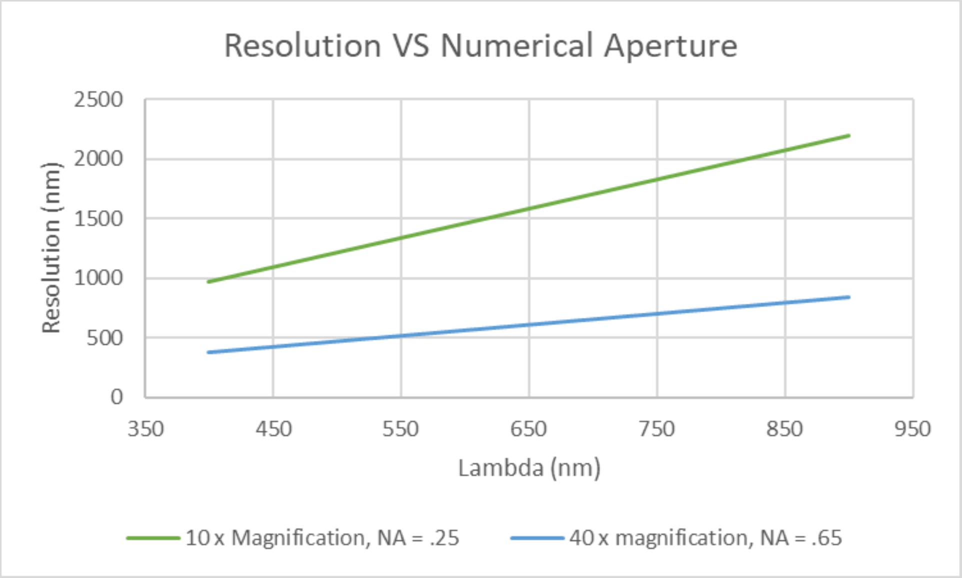


Figure 33 - Resolution vs. Numerical Aperture
A comparison of resolution versus wavelength for 10-x objective lens and a 40 - x objective lens.

As shown in the figure above, the 40x magnification with a numerical aperture .65 has the higher resolution and the resolution in both cases gets better as the wavelength gets smaller. What this means is the smallest detail we can observe in the infrared is 844 nm or .844 um using the 40-x objective lens with a numerical aperture of .65. Considering an onion cell is approximately 75 um wide and 120 um long, we should be able to image it with no issues.

5.3.1.5.2 Field of View

The field of view is another key parameter of optical systems. It is closely related to the resolution, and is defined as the angle that can be seen through the lens. The lens in our system is the aperture and is what defines our field of view. The field of view is the angle at which objects can still be seen through an optical system. In our system the objective lens is what determined the field of view.

With the numerical apertures that we have our field of view can be obtained using the equation:

$$NA = \sin(\theta)$$

For our 10-x objective lens the numerical aperture is .25 therefore

$$.25 = \sin(\theta)$$

$$\sin^{-1}(.25) = 14.48^{\circ}$$

For our 40-x objective lens the numerical aperture is .65 therefore

$$.65 = \sin(\theta)$$

$$\sin^{-1}(.65) = 40.54^{\circ}$$

These equations represent the absolute best numbers that can be achieved, they are the theoretical limit of each objective lens. We will be testing these numbers at a later date and will include the results in this report.

5.3.1.5.3 Focal Length

While both field of views are good, the focal length will also determine what will be viewed under the microscope. The focal lengths are determined by the following equation:

$$M = \frac{Focal\ Length\ of\ Tube\ Lens}{Focal\ Length\ of\ Objective\ Lens}$$

$$40 = \frac{160\ mm}{4\ mm};\ 10 = \frac{160\ mm}{16\ mm}$$

If you are not at the correct focal length the image will be blurry and out of focus, to account for this we will have guards that are the correct length so that when the microscope is pressed onto a sample it will be nearly in focus. The second part of fine tuning the focal point will be to have a mechanical piece inside of the tube that can move the objective lens closer to the specimen as needed. This design will be further discussed in the design portion of this document.

5.3.1.6 Cover Slip or Slide

The objective lens also refers to a cover slip or slide thickness. Ours is .17 mm, the reason for this thickness being indicated is that most samples use a cover slip. This thickness shows that the objective lens is already corrected for that thickness of cover slide. This is important because a cover slip is made out of glass, and as such has a refractive index. Anytime light travels through a medium with a refractive index it will be bent, and that can cause blurry images even though it is only .17 mm thick. Since the majority of microscopy applications use specimens that are on a slide, the manufacturers generally correct for a specific thickness.

Most samples are put are a slide, this is for multiple reasons. The first being that a lot of sample are biological tissue or very fragile, this cover slip ensures that they are protected. The coverslip is also used to preserve a sample for an extended amount of time. The objective lens itself is also protected by the slide, it keeps any wet, dyed, or dirty sample from getting on to the objective lens. These slides also make it easier to move the specimen around under the lens. They also make it easier to find the sample, because samples are generally put in the slide.

Our microscope can certainly make use of samples that under a cover slip, but we want our microscope to see things that are not usually underneath a cover slip such as a table, a wall, or maybe even an orange. We have discussed a few ways to fix this without using glass, one option is to just not see anything that is not under a cover slip. That does not work for us because the whole point of our design is to explore the world without the hassle of a traditional microscope. Another option was to put a transparent lens cover on that is the same refractive index and thickness of a cover slip. This is not feasible for a couple of reasons. The first is that it would have to be custom made and we do not have the means or knowledge to do that. The second is that it still would not work for round objects. There was also the problem of it breaking or getting scratched. For all of these reasons we decided it was a bad idea. The next idea was to make some sort of polymer that mimics the qualities of a slide but it flexible enough to put on a round object. This is a very interesting idea, but we still ran into the problem of not knowing how to go about doing that. So as of now, we will use a coverslip when we have one, and take a lower resolution when it is not possible to use one.

5.3.1.7 Tube Lens

Tube lenses are employed in all infinity conjugate microscopes. Infinity conjugate objective lenses bend the light into straight rays as shown in the ray diagram in Figure 34 below.

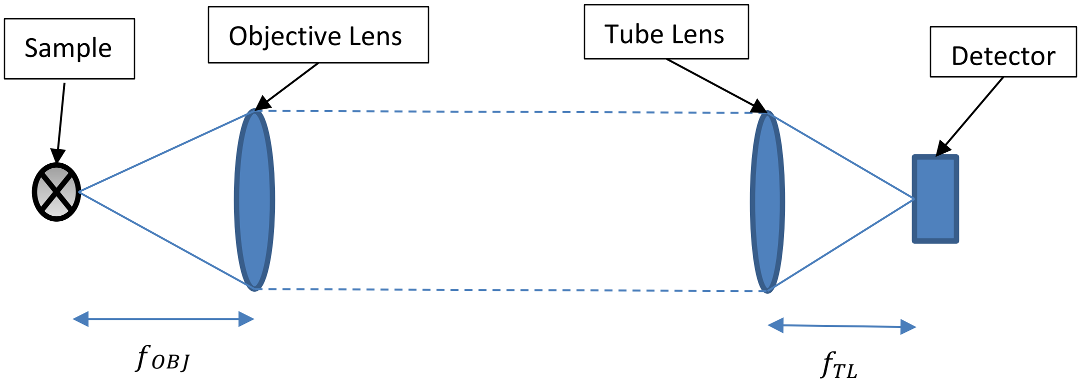


Figure 34: Tube Lens
Ray diagram of infinity corrected objective lens in conjunction with a tube lens.

These extra lenses allow the magnification to be perfectly matched to pixel size using the following set of formulas:

$$resolution = \frac{.61 * \lambda}{2 * NA}; Magnification_{TubeLens} = \frac{Pixel\ size}{resolution}$$

Once the magnification is calculated, you can then determine the needed focal length of the tube lens using:

$$f_{TL} = M * f_{OBJ}$$

For example, to perfectly image an object with a 40-x infinity corrected objective lens with numerical aperture of .65 to a 3-um pixel size detector at 850 nm is as follows:

$$r = \frac{.61 * 850\ nm}{2 * \ .65} = 400nm; M_{TL} = \frac{3\ um}{400\ nm} = 7.5$$

To get a magnification of 7.5 x:

$$f_{TL} = 7.5 * 4\,mm$$

Without this a tube lens you will not be able to focus your image onto an eyepiece or a detector. Without a tube lens of the correct focal length for your detector you will start to digitize your image and lose some resolution. The tube lens also allows for additional elements to be added inside the tube length such as beam splitters and filters.

In addition to being brightfield our microscope is also finite conjugate, which does not utilize tube lenses. We chose this type of microscope because of our objective lenses. Our objective lenses are Duetshe Industrie Norm standard which means that they are perfectly focused for a 160 mm tube length with a Duetshe Industrie Norm eyepiece on the end. Since we will be using a detector and not an eyepiece, our detector is located 150mm away from the objective lens. This is where the intermediate image plane is. The ray diagram is shown in Figure 35 below.

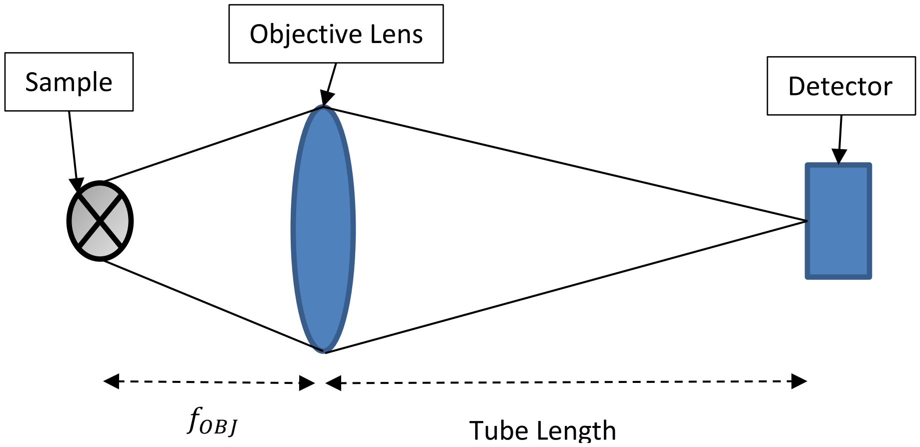


Figure 35: DIN Standard Microscope
Ray diagram of a finite conjugate microscope.

The benefits to using a Duetshe Industrie Norm objective lens, is that the objective lenses are all standardized for a specific tube length. Infinity corrected objective lenses change needed tube length based on the manufacturer. Some require 165 mm, 170 mm, or 180 mm. Another advantage to using this type of lens is that an additional lens is not required to focus the image onto the detector. This type of lens is also much cheaper than their infinity corrected counter parts, making it ideal for the common user. This will also make our design more robust because they are less optical elements along the optical axis that can shift when moving around. This means that it can withstand a little more wear and tear.

The drawbacks to using a Duetshe Industrie Norm standard objective lens, is that without a tube lens, it will not be able to perfectly match the magnification to the pixel size of our detector. We can add an additional lens and refocus the image, however this will reduce resolution and can increase aberration of the lenses. We want to get the highest resolution that we possibly can, but the price of upgrading to infinity corrected objective lenses is around eleven hundred dollars. That is the price for the one 10 x achromatic, one 40 x achromatic lens, and one tube lens. This is unfortunately out of our price range at this moment and would not be a viable option for most consumers anyway.

5.3.2 Image Sensor

This section is dedicated to the optical side of our sensor. It will discuss how the sensor will interact with the other components, it will discuss the way it takes an image, and it will discuss the spectrum that is used to image. This section will not discuss the electrical or computer side of the sensor.

The first thing to note is that the image sensor is not a camera. It is in fact an array of pixels that are capable of detecting light. This light is then converted into an image using the bits from each pixel. There is no lens on this to focus the image directly onto it, so it is essential a blank canvas for an image to be formed pixel by pixel. It is important to realize that because it does not have a lens, the image would not match the pixel size by just placing the camera at the sensor. It will need to have a lens added to the system so that the image can fit perfectly onto each pixel. We have previously discussed this in the microscopy section.

Initially we did think that we could just place the sensor at the image plane, however because it is not a camera it does not have a lens to focus the image. This means that unless we turn the sensor into a full blown camera, we will need to add a lens to it. We have ordered a lens to do just that. Without this lens, the image runs the risk of aliasing. This means that the light reflected off of the sample would not send a clear message but would instead start to overlap, much like a digital signal does without the proper bandwidth. This would cause a blurry pixelated image, and a microscope cannot have a blurry image if it is to be functional.

5.3.2.1 Pixels

For our design, we chose to use an array of sensors to detect the image instead of an eyepiece. This has a few advantages, first it is cheaper. Eyepieces contain multiple lenses, so they cost a lot. Another reason is that with a sensor, the image only has to be focused once instead of each person adjusting the microscope so that they can see. Also, a sensor allows us to transmit the image digitally to another device so that everyone can see the same image.

There are drawbacks to this design, the first one is that we need to perfectly focus the image onto the pixels such that a clearly defined image can be observed. This is easier said that done, because a lot of objects are actually quite large and magnifying them at all can make the image of the said object too large for our sensor.

It is actually easier to create a microscope that has high resolution using an objective lens and an additional lens as an eyepiece, and a camera or just your eyes. This is because a camera and your eyes already employ complex optical systems that readily adjust to focus an image on to the sensor of a camera or your retina.

Foregoing these prebuilt optical systems creates new challenges for us as engineers to solve. Starting with how to match the image to a pixel size of 1.4 um. We could de magnify the image to fit on the pixel but that defeats the purpose of magnifying it in the first place. We could try to get a larger sensor that has larger pixels, but that is also out of the question because they do not make sensors that large. The most viable option is to use specimen that are quite small to begin with, that way when they are magnified they will hit the pixels at a better ratio.

5.4 Software Architecture and Design

The following sections detail the overall software architecture and design for the mobile application component of this project. These sections include, but are not limited to, image processing considerations, software libraries used, and the nuances between the different modes of data storage. In addition to discussing these categories in a very technical manner, justification behind our software design decisions can also be found in the following sections.

Software Requirements

- Mobile application that can stream video from microprocessor
- Ability to perform basic image processing on the video stream
- Allow for local storage of videos and screenshots
- Provide an aesthetically pleasing, yet simple UI
- Provide the ability to tag images and sort them from within the app

Software Stretch Goals

- Add a web interface that has the same functionality as the mobile application
- Allow for user accounts so that users can store and retrieve media in the cloud
- Instead of a one-to-one connection between the microprocessor and the mobile phone, allow one mobile phone to distribute the data to other phones in a master/slave type of fashion (useful for classroom settings)

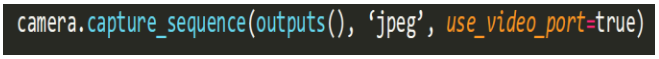
5.4.1 Raspberry Pi

To capture camera input and transmit images to our Android application, we decided to use the Raspberry Pi 3 Model B+. This model of the Raspberry Pi is currently the most powerful Raspberry Pi available on the market, featuring processing speeds of up to 1.4 GHz and 1 GB of on-board memory. Most importantly, this model of the Raspberry Pi includes an on-board Wi-Fi module that achieves average network transfer speeds of around 90 Mbits/sec, which outperforms older models of the Raspberry Pi by a factor of three. Careful consideration of the network speed of our chosen middleman device was critical for the success of this project. For this project to be effective the Wi-Fi module chosen had to be capable of sending images quickly enough so that we could achieve frame rates exceeding thirty frames per second. While it would be possible for this project to work if we were getting less than thirty frames per second on average, it would certainly detract from the user experience.

5.4.2 Interfacing with the OmniVision OV5647 Image Sensor

All images are obtained by interfacing with the Raspberry Pi's camera module using a Python script. The Raspberry Pi camera module comes complete with a set of commands that are used for performing various tasks with the camera, such as taking a single image, recording a video, or taking a continuous stream of images.

Our implementation involves utilizing the PiCamera Python module which allows us to interface directly with the camera without having to open an external shell and hardcode a startup command. The PiCamera module allows us to instantiate a PiCamera object which allows us to easily fine-tune specific parameters such as the image resolution, image orientation, and framerate. For our project, we have chosen an image resolution of 600x400 pixels, and set the desired framerate to be a theoretical maximum of 80 frames-per-second. The command to begin the capture sequence is shown in Figure 36.



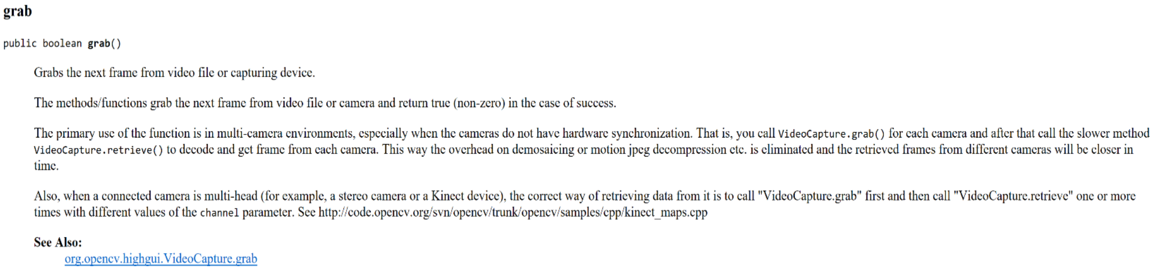
```
camera.capture_sequence(outputs(), 'jpeg', use_video_port=true)
```

Figure 36: Capture Sequence Code Sample

The method outputs() parses the IO buffer and sends the images to the mobile application. The use_video_port parameter is used to simulate the speed of video captures, but in a burst format with a series of still images. The use of this parameter in conjunction with setting the framerate of the camera to its theoretical maximum allows us to obtain images from the camera module at peak efficiency.

5.4.3 Displaying Video using OpenCV

Once the data has been received, depending on the format of the data, it will need to be decoded by OpenCV so that it can be properly be read and displayed. The specific class that we considered using for this was *org.opencv.highgui.VideoCapture*, with the following methods used for receiving and decoding the data Figure 37:



grab

`public boolean grab()`

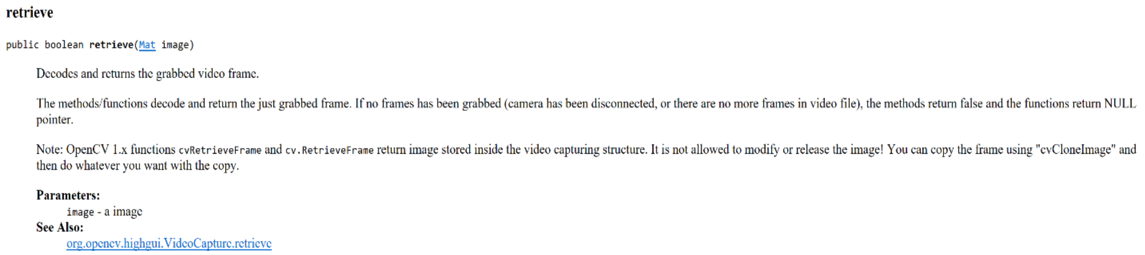
Grabs the next frame from video file or capturing device.

The methods/functions grab the next frame from video file or camera and return true (non-zero) in the case of success.

The primary use of the function is in multi-camera environments, especially when the cameras do not have hardware synchronization. That is, you call `VideoCapture.grab()` for each camera and after that call the slower method `VideoCapture.retrieve()` to decode and get frame from each camera. This way the overhead on demosaicing or motion jpeg decompression etc. is eliminated and the retrieved frames from different cameras will be closer in time.

Also, when a connected camera is multi-head (for example, a stereo camera or a Kinect device), the correct way of retrieving data from it is to call "VideoCapture.grab" first and then call "VideoCapture.retrieve" one or more times with different values of the `channel` parameter. See http://code.opencv.org/svn/opencv/trunk/opencv/samples/cpp/kinect_maps.cpp

See Also:
org.opencv.highgui.VideoCapture.grab



retrieve

`public boolean retrieve(Mat image)`

Decodes and returns the grabbed video frame.

The methods/functions decode and return the just grabbed frame. If no frames has been grabbed (camera has been disconnected, or there are no more frames in video file), the methods return false and the functions return NULL pointer.

Note: OpenCV 1.x functions `cvRetrieveFrame` and `cv.RetrieveFrame` return image stored inside the video capturing structure. It is not allowed to modify or release the image! You can copy the frame using "cvCloneImage" and then do whatever you want with the copy.

Parameters:
`image` - a image
See Also:
org.opencv.highgui.VideoCapture.retrieve

Figure 37: OpenCV Display Methods

Once the video has been properly read, the resolution can be set based on the size of the phone screen and display the video. OpenCV is known for being a great library for anything to do with image and video processing, and the documentation for it is incredibly thorough and is widely used across the world.

Our number one priority when getting the video to display properly was maintaining an appropriate framerate while preventing any tearing or disfiguration in the picture. Through our extensive research, we found that the transmission of video frames of around 30 frames per second is easily achievable, depending on how quickly the frames can be received from the microcontroller.

Originally, our plan was to receive JPEG images from the microcontroller and then pass those along to our camera object within OpenCV and then display those on the canvas that will be presented in the middle of the mobile application.

The canvas where the stream is displayed live on the streaming activity of the application and takes up about 70% of the screen. If the application is properly connected to the microcontroller and is receiving and processing frames as expected, they will show up here. Otherwise, a template image will be displayed if no other images have previously been read.

If previous images have already been read and displayed and an issue occurs, such as a loss of connectivity with the microcontroller, the previous frame that was processed will be displayed. If connectivity is completely lost and frames are no longer being received, a dialog will appear and notify the user about the disruption in the connection and the image will revert back to the default, which will likely be an image with a “Connection Failed” caption.

One option that we set as a stretch goal and successfully implemented is the ability to allow the user to display the video feed in a fullscreen mode. This process was done by simply expanding the canvas to the full dimensions of the phone. In addition to that, there are settings that detect the current orientation of the phone so that we can flip the dimensions of the canvas so that the stream is displayed as intended. This was mostly a matter of an Android API call, which should prove simple and visually clean.

One concern that we had when adjusting the canvas size of the video display was the stretching of the image, which would likely result in a blurrier, more pixelated version of our images. Through our previous experience with OpenCV, there are settings that can be adjusted to alter the type of interpolation that is done when increasing the size of an image. Similarly, if one were to shrink an image using OpenCV, the library can take the average of neighboring pixels to account for one pixel, which allows us to scale images however we would like. The process of allowing fullscreen viewing was a stretch goal for this application and is something that will be researched and tested further if we decide to include it, although early calculations seem to indicate it will be realized without undue trouble, as large-resolution uncompressed images should be capturable from the embedded system, though video transfer at this level of image data size would look more like a progression of still images than a true video. More information on this can be found in Section 5.2.2.2.

5.4.4 Data Storage

One required feature of this mobile application was the ability to store images and video to the phone's internal storage. While the storage is internal in the sense that it is on the phone and not on a remote server, the storage that the data we will be storing to is considered Android's external storage. Listed in Table 11 are the differences between internal and external storage on a device running the Android operating system:

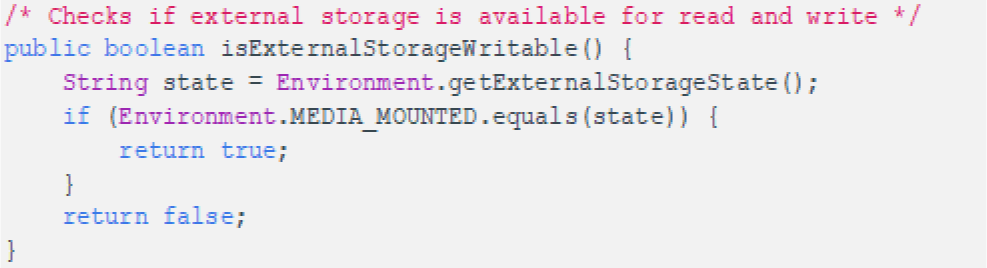
Internal Storage	External Storage
It is always available to the application. The user can't just remove this.	It is not always available, because the user can mount the external storage as USB storage and depending on the type of storage, remove it from the device
Files saved here are only accessible by the app and not by the user	It is world-readable, so the files stored here may be read outside of the app's control
When the user uninstalls the app, the system removes all of the app's files from internal storage.	When the user uninstalls the app, the system will only remove the app's files from this if they are stored in the directory returned from getExternalFilesDirectory().

Table 11 – Storage Differences on Android OS
Key differences between internal and external storage on the Android OS

Since we wanted our users to be able to share the media that they have collected from the application, all media that is gathered will be stored to the phone's external storage. There are some drawbacks to this, such as corrupted or lost data if the user were to remove the external storage device while writing to storage. This was circumvented by caching the media somewhere or temporarily storing it in the phone's internal storage until access to the external storage has been restored. Alternatively, the images could be stored in a database automatically after they're taken as a backup, but there are many reasons why this is a bad idea.

For starters, we would then need a way to uniquely identify images based on users, which would mean that we would need to incorporate support for user accounts (currently a stretch goal for the project). Secondly, storing images in a database is generally not recommended because to reference these images, you'll need to reference the database and download the image, which is going to cause serious performance issues. Finally, this method is almost certainly overkill considering how rare this use case is. Generally, users will not remove their external storage device when using the product, and on the off chance that they do, caching to the internal storage was a much more practical idea than building a database and a having a remote file server that hosts all of the media files.

However, we do need to consider the case where the external storage becomes unavailable. There are several reasons why an external storage device might be unavailable, but the most common occurrence is that the user accidentally removed it or that it was never there in the first place. To prevent any potential errors, it is imperative that we check whether we can access the external storage before trying to write any data to it. In Java, we can do so by executing the following snippet of code, shown below in Figure 38:



```
/* Checks if external storage is available for read and write */
public boolean isExternalStorageWritable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state)) {
        return true;
    }
    return false;
}
```

Figure 38: Storage Code Sample

Code Sample showing the process for checking if a storage device is available

Upon failure, a notification is displayed to the user telling them that their external storage is currently unavailable, and that media will be unable to be stored until the issue is corrected. This check is performed immediately after the user clicks one of the buttons that starts the media capture.

Another vital component of external data storage on Android devices was the request permissions requirement. For an application to access user files with read/write access, the user needed to be prompted with a dialog box asking whether the application is allowed to write to the external file storage. If the user agrees then the app will proceed normally, but if not, a new notification will display informing the user that media cannot be saved to the device due to insufficient permissions. This notification displays and informs the user to restart the application and accept the permissions request. These various hardware permissions are granted in the manifest file of an Android application and the requirements vary based on the API level that is being used. According to the Android developer documentation for data storage, a write permission request is only required on API level 18 and lower, meaning that devices running API level 19 or higher will not require this permission request. To mitigate this, we just adjusted the maxSdkVersion attribute of the request in the manifest file to read 18, so that the application only requests permission on API levels lower than 19.

In addition to these read/write permissions, checking the size of the data was very important and was something that was handled by the software in the mobile application. Generally, the Android operating system will notify a user when their phone's storage capacity reaches about 90%, but this is not something that should be relied on to prevent exceptions.

Thankfully, there were ways to check the current storage capacity of your given environment using the *java.io.File* library, which contains a method for checking the amount of available free space currently left on the device. Since media files are stored to the device after they are finished being captured, we simply compared the amount of free space on the device to the size of the media file that we are trying to save and either allowed or blocked the operation depending on whether there was enough available space.

This approach could potentially be annoying for a user who is recording a lengthy video only to find out after recording for a few minutes that their files were discarded due to insufficient space. To notify the user more quickly, we used some mathematical strategies to estimate the final size of the video while it is being recorded and notify the user in real-time that their video won't be able to be saved and subsequently stop the recording process. This is something that most Android devices support by default when using the pre-loaded camera application, so implementing something like this wasn't cumbersome and will save our users from headaches when they are trying to capture media files from the stream.

5.4.4.1 Naming Convention for Files

To guarantee a unique naming scheme for the files that are stored to the user's local storage, the application named the file either image, video, or time, and appends a timestamp to the name. This ensures that no files are overwritten due to more than one file sharing the same name. For example, a still image might have the filename: image_201810040340151, where the number sequence corresponds to the following: YYYYMMDDHHMMSSX, where X will be equal to one if it is PM or equal to 0 if it is AM.

The method listed above is one way of making a unique identification number to append to the actual name of an image. Having a noun such as image, video, or time is very generic and might make identification of the images difficult for the user, which is not ideal. Instead, we decided to only set this as the default option and then allow the user to organize images in a better way by giving their media files a unique name to use in lieu of a generic name that identifies the file type.

Additionally, we found that when experimenting with other portable microscope products, the software provided did not allow the tagging or ordering of images. This was bothersome while we were testing, and as a result, we felt that incorporating the ability to do this into our project was a good idea and would greatly benefit our users.

5.4.4.2 Organizing Images

Categorization of the media files was a functionality that was integral to the success and utility of this mobile application. As such, we wanted to make sure that our users were able to organize their collected media files in a clear and coherent manner. For example, we wanted a toggle that allowed the user to set a label for the images that are about to be taken.

To do this, a user will be allowed to type in a folder name for the set of images that are about to be taken, then the application will create a subdirectory within the images folder where the new images will be stored. They then have the option to either choose an existing folder on the phone's storage or create a new one. We also offer the ability for users to leave their images as uncategorized. If the user chooses to do this then their saved media files will be saved under the unfiled directory. The use case diagram for this process is shown below, in Figure 39.

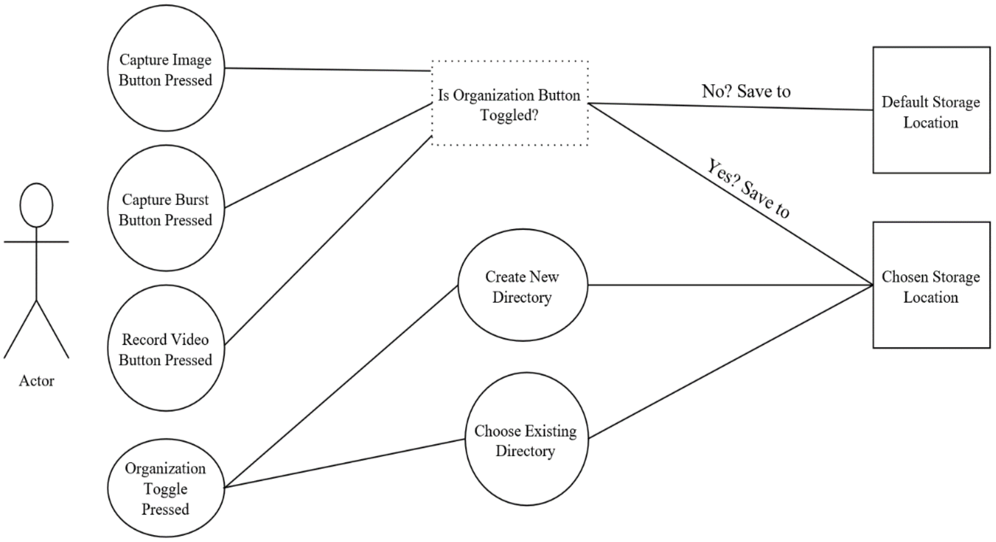

Actor
Capture Image Button Pressed
Capture Burst Button Pressed
Record Video Button Pressed
Organization Toggle Pressed
Is Organization Button Toggled?
No? Save to
Yes? Save to
Create New Directory
Choose Existing Directory
Default Storage Location
Chosen Storage Location

Figure 39: Use Case Diagram
This diagram is showing how images will be organized

This categorization of images was an important aspect of this application and directly impacts the overall user experience. We have imagined our own directory structure, containing a large group of media files that we have obtained and realized how annoying it would be to try and find a media file that you are looking for if it has some generic name, such as "image001". Granted, this might not matter to a lot of people, but we felt like giving the user control of the naming and categorization of their captured media files was something that needed to be included.

By default, we tried to create a naming convention for our media files that maked it easy to identify and select their images while maintaining its uniqueness property, but in reality, this is something that is very difficult to accomplish. The most common ways to generate unique file names are to generate a unique UUID to append to the name of an image, or to use a timestamp. We considered using a unique UUID, but not only would this require an algorithm that never produces the same value twice, but the readability for the user would be likely be terrible. The only way that this UUID wouldn't have poor readability was if it was in some sort of recognizable format, such as a timestamp. To reach a compromise, we ended up using the type of media file that we are saving and appending the current timestamp to the end of it, ensuring that the generate filename will always be unique.

As it turns out, it’s fairly common practice to use a timestamp as a unique identifier, as time is one of the only things in life that is never the same at two given points. In addition to the uniqueness of our filenames, they are also readable if you are familiar with generic time formats. As such, we decided on this for our default naming scheme. Of course, many people might want more specific names for their data, which is why we also allowed for users to customize the organization and naming of their files.

5.4.4.3 Data Recall

One feature that was desired of the application was the ability for the user to view the images and media that they have captured. Ideally, we wanted an option for the user to have instant access to all the media that has been saved by the app instead of having to sift through all of their other media files that might be stored on their phone. A simple way to do this was to somehow store the file paths of all the media files that are saved by the application in some kind of lookup table. One way that we considered tackling this was to use SharedPreferences, which are essentially key-value mappings stored in an XML file that allow data to be stored even when the application is closed, or the phone is turned off. This is great for what we need but there is a size limitation of SharedPreferences of 1.4 megabytes, which really should not be a problem, but it is possible for someone to have that many paths stored, especially if they frequently capture a series of images. Additionally, the SharedPreferences interface was primarily designed to store user preferences consisting of a few variables and not to store copious amounts of repeating data, such as hundreds of file path strings.

Another possible solution that we considered was to have a file explorer that displays all the user’s images on their device and allows them to be viewed within an image viewer from within the application. While this works for what we want, we need to consider the use-case for this project and who will be using this application. This product is intended to be used primarily by educators, which means that these educators are going to want quick, easy access to media files captured by the application so that these educators can show their students what they are looking at under the microscope. Not only would an educator probably not want all the other images on their phone being displayed in the application, but it could result in delays in selecting the appropriate image, which is not advisable in a classroom setting. However, we have also considered the use-case where someone might want to load an image into the application and then perform some image processing on the image, such as sharpening the image. This seems like a valid use of the app, so while we won’t automatically load all the user’s images into the application, we added an option that gives the user the ability to load their own images from their external storage device into the phone and modify them if they wish to do so.

An even better solution, and what we ended up using, was an SQLite database with a table containing all the paths for the different images captured by the application. We wanted the user to be able view the recent media files that they've collected and have the option to rename them, delete them or create copies of them. To do this, all we need is access to the path for the media file and then we can allow the user to make any desired modifications that they want.

This database model is incredibly simple and only requires entries to have two attributes, the primary key, which is the path of the file, and the name of the file, which will be used for the ListView where these entries are displayed on the application. This mediaPath attribute is guaranteed to be unique because only one file can exist within a given location. Entries are added to this table as we capture images and video. This data is pulled from the database immediately when the application opens and then a ListView is populated containing a thumbnail of the image and the name corresponding to the media file. From here, the user can modify the image to their liking. However, since this data is likely to update dynamically as are processed and displayed in the file explorer, we considered up switching out the ListView for a RecyclerView. The RecyclerView is effectively the same thing as a ListView except it can dynamically modify contents of the list without requiring a user gesture such as a swipe or a button press. ListViews, unfortunately, are static elements that cannot be modified at runtime without destroying and recreating the entire list from scratch. Using a ListView can become a problem because destroying and creating a ListView are very costly operations if we are dealing with any reasonably-sized list.

One drawback of using this method was that to maintain the integrity of the file paths of these media files they cannot be modified by the user outside of the application. This really was not a major issue as we just checked for the existence of a file given the existing path in the database and then delete the entry if the file no longer exists, but it was fine considering that this feature of the app should only be used for images that were recently taken. In a perfect world, we would be able to ensure that all media that is captured by the application would remain on the device, but that is out of our control and beyond the scope of this project. If the user chooses to delete an image from their personal file explorer or camera gallery, then we needed a way to handle that, and one simple solution was to compare contents in the database to actual file contents on the phone and remove any file paths from the database that no longer exist on the user's device.

Another consideration for the SQLite database was to have a user's table that contains important information about a given user, including a primary key that represents the user's unique id. This id would also be a foreign key in a corresponding images table which would link a set of images to a specific user. The reason that we decided not to do this is because there really isn't any point in having a user's table at all since each instance of the SQLite database will only contain information relevant to one user. This is one of the good things about using SQLite. Each SQLite database that is created for android application is unique to that application instance, meaning that we do not have to design an intricate database schema or worry about any security vulnerabilities that would exist if we were using a remote database. Unfortunately, if we were to try and support multiple users, our database schema would need to be redesigned and we would also have to abandon SQLite for a different relational database management system, such as MySQL. In addition to that, we would need to find a suitable host for our remote database, such as Digital Ocean, AWS, or Azure. In the future, if we decide to expand and allow for user account support so that images could be stored in a cloud service, it would require an entire backend implementation to be designed and implemented, which would require a great deal of effort.

While multi-user support sounds like a great feature to have, there really did not seem to be much need for it given the scope of this project. The goal of this project was to create tool that can be used to stream media from a microscope and display it on a smartphone. Implementing our own remote data storage seems unnecessary considering that we have access to the mobile devices that have their own form of storage that do not require us to integrate with any third-party remote hosting service. Additionally, having user accounts requires us to create and maintain a registration and login process for our application, which also seemed excessive for this application. We ultimately wanted this application to require minimal setup and use for users, and adding user accounts and all of the supporting features that go along with it would be an additional complication that would take away from the intuitive nature and accessibility of the application.

5.4.5 Data Transmission

The data transmission portion of this project involves the method in which the JPEG image is transferred from the microcontroller to the Android application and how control signals will be transferred from the Android application back to the microcontroller. This data transmission will take place over a Wi-Fi direct connection with socket programming. The complexities of both will be discussed in great detail in the following sections.

5.4.5.1 Sockets

To establish a connection and transmit data between the microcontroller and the mobile device, a socket needed to be created between the two so that both devices could send and receive information.

We decided to go with sockets for this method because we want a one-to-one duplex style connection that will allow for us to transmit data between the devices. There are some other alternatives to this approach that we encountered, but socket programming is something that is ubiquitous, meaning that finding documentation for it and troubleshooting any potential errors would be much simpler than if we chose some other obscure methodology to use. Socket programming is well supported in the Java programming language and creating a stable implementation that connects two clients together was a relatively simple task when using the built-in libraries that Java has to offer.

For a socket connection to be established, the two devices need to know some information about the other. The two devices will need to know the IP address of where the server is running, and the port that the connection is run on. We decided to run the socket server on the microcontroller using the device’s Wi-Fi capabilities and then use the microcontroller’s IP address and designated port number to connect and establish a connection between the two devices.

As seen in Figure 40, the server is responsible for establishing a socket by binding an IP address and a port, essentially opening a connection stream to the outside world. From this point on, the server is now listening for incoming connection and communication requests. The client is then responsible for connecting to the socket that the server created. If the connection is successful, then the two devices will be able to transmit data back and forth between one another.

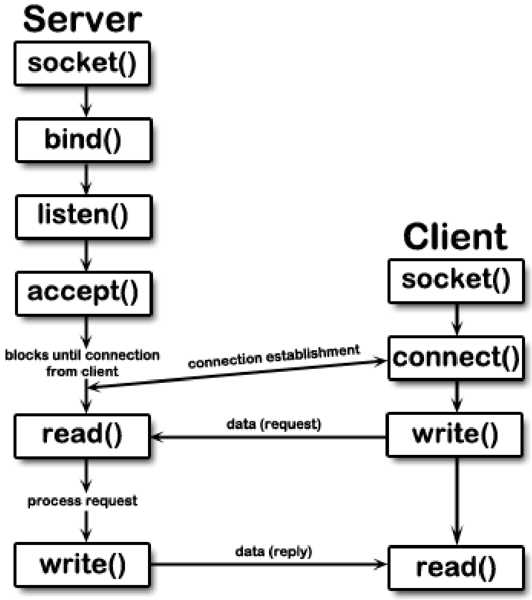


Figure 40: Diagram of Socket Programming Client-Server Model

5.4.5.2 Control Signals

The mobile device also possesses the ability to send control signals to the microcontroller. These control signals are used to control various features of the microscope, such as turning LEDs on and off, requesting either a picture of a video, or changing any other settings that we had in place with the microscope. The control signals will be designed and tested during the development stage of the project, but they are sent as binary strings that will be received, parsed, and decoded by the microcontroller.

5.4.5.3 Internet Protocol Choice

When dealing with data transmission, arguably the most important part of the decision process was the protocol that was chosen. An internet protocol is just a set of rules that governs that format of data that will be sent over a network. There are numerous internet protocols out there, but a couple of the most common internet protocols are TCP (transmission control protocol) and UDP (user datagram protocol). We considered the protocol that was chosen for this project and discussed in further detail the intricacies of both TCP and UDP in their respective sections in this document.

5.4.6 Google’s Android Design Guidelines

The four major components that go into the core app quality guidelines are visual design and user interaction, functionality, compatibility, performance and stability, and security. A summarized breakdown of what each of these entails is shown in the Table 12 below.

Core App Quality Tenant	Description
Visual design and user interaction	This section includes adhering to standard design protocols, avoiding redefining standard components of buttons (e.g., navigation buttons), notifications do not contain advertising information or irrelevant material, notifications only being used to indicate a change in context personally relating to the user or to expose information about an ongoing event.
Functionality	This section pertains to requesting only the absolute minimum permissions necessary from the user, normal functionality occurs on SD card, support for landscape and portrait mode is included (if possible), services are not left running when the app is closed (unless absolutely necessary), and the app is able to handle rapid transitions in a fluid manner.
Compatibility, performance and stability	This section pertains to ensuring that the app doesn’t crash, force close, freeze, or otherwise function abnormally, loads and shuts down quickly, can run on the latest public version of the Android platform, displays media smoothly, without crackle, stutter, or other artifacts.
Security	This section pertains to safely and properly handling users’ private data. All private data should be stored in the app’s internal storage, all data from external storage is verified before being accessed, all private should be encrypted and network traffic should be sent over SSL.

Table 12 – Google’s Android Design Guidelines

Core App Quality Tenants for Android Mobile Applications

These core app quality guidelines that Google has published must be tended to before the app can be published to the Google Play Store. If any of the quality requirements are not adhered to, the app cannot be published until the issue is corrected. As such, our mobile application closely follows these guidelines to ensure the highest quality possible.

Our mobile application in its most basic form does not be transfer private data across a network, so there are not any issues regarding user privacy. The only potential area of concern is the Wi-Fi connection that the mobile application establishes with the microcontroller. However, this behaves the same way as connecting to any other access point, so the user accepts the same responsibility that they would accept connecting to any other wireless access point.

The user interface for connecting to the access point will be the same as your traditional interface that is used on for connecting to the internet on any mobile device. The mobile application will scan for available networks and select the appropriate access point from there. If the user wishes to, they can select an option that will allow the mobile application to remember the network and automatically connect to the same one the next time the application is loaded.

5.4.7 User Interface Design

Android applications rely heavily on their appearance to attract and keep users. It has been shown time and time again that despite the functional value of the application, if it looks bad then the developer is going to have a difficult time maintaining a userbase. To help developers out, Google has constructed a set of material design guidelines that developers should refer to when creating their apps. If you look at the Android play store’s most popular apps, a common theme with all of them is that they follow these design guidelines. Google has found through extensive research that people like the look of apps that follow these guidelines and thus, will be more inclined to download them.

The material design guidelines are very thorough and were be referred to through the duration of the project. Our app developer will be familiarizing themselves with these guidelines to ensure that our app looks great and functions as intended. This material design guide also includes documentation for the material design support library, which includes new components and styles that can be used in Android applications. Not only do these guidelines touch on ensuring that your application is visually appealing, but they also ensure that your application appears as intended on all devices, including tablets and TVs. There is a surprising amount of work and precision that goes into the design process of a mobile application, so this was one of the more difficult, yet rewarding parts of the development process.

The design of the application required a lot of feedback from the project team and was altered multiple times as we thought of new and creative ways to display information on the device. This was one of the big reasons while we picked the Agile software development methodology, as allowed us to examine the incremental progress of the app and either omit or add unique features as we made progress.

While the design and overall appearance of the mobile application was a critical portion of this project, priority was be given to getting the functionality of the application in working order before we spent too much time on the aesthetics. Since the primary functionality of the application was completed ahead of schedule, the remaining time was spent improving the overall appearance and user experience of the application. The current plan was to focus on making the user interface intuitive, clean, and attractive. One of the main points made during Google's material design document is to refrain from using too many flashy features and listings, as overcomplicating the application with these features might make the overall product look better, but it will almost surely detract from the user experience and result in some mixed emotions about the overall application.

5.4.7.1 Mobile Interface for Connecting Devices

A lot of consideration has gone into the interface that allows the mobile device to connect the server socket on the microcontroller. Like how a user can select the wireless network to connect to on their phone, the mobile application is able to view a list of available wireless networks that it can connect to. One of these connections is the wireless access point that is enabled on the microcontroller. This access point will likely have a predefined SSID, like "*Portable_Microscope*", which the user can choose to connect to. As of right now, we do not plan on having any kind of advanced wireless security to protect this network other than a simple WPA2 authentication with a predetermined password. .

Once the mobile device is connected to the same network as the microcontroller, we also allow the user to be able to connect directly to the socket through the click of a button. The plan is to currently have a predefined static local IP address and port that the microcontroller uses when establishing the server, and then just have the app automatically try to connect using those settings. The goal of this was to make the interface as simple as possible for the user.

One option that we considered was to simply have two text fields, pertaining to the IP address and the port number and allow the user to specify those on their own, but we felt like that wouldn't be a viable option for most end users, as they will likely not know how to or not want to go through the process of typing out an IP address and port number in order to establish a connection.

Seeing as how a mobile device will be the only thing connecting to the microcontroller at a time, there weren't any issues with the socket being unavailable. Additionally, since the two devices will be on the same network and the server socket uses a local IP address, we didn't need to worry about the risk of attackers obataining access to the socket established between the two devices.

To further reduce the likelihood of an attacker gaining access, we had the option to both secure the wireless access point on the microcontroller and encrypt the data that is being sent between the two devices. However, encrypting the data seemed like it would be overkill considering that there isn't any sensitive information being transmitted between the two devices. If a packet was somehow intercepted between the devices, the only thing that they would be able to modify is the image being sent to the mobile device or the control signals being sent to the microcontroller.

While modification of the control signals might be an issue, there were checks on the microcontroller that ensured that the bit string that was received was valid, so that no unexpected behavior occurs and potentially causes damage to the microscope. Furthermore, the actual control power that the mobile application has over the microscope is limited. As of now, the only control that the mobile application has over the microscope is the ability to toggle LEDs and the ability to modify some basic settings, such as the type of data that is received from the microcontroller.

Upon opening the mobile application, the first thing that the user sees is a status bar indicating whether the device is currently connected to the microscope. This status bar says either in green text that the device is connected, or it says disconnected in red text.

To reach the interface where the connection process takes place, the user can simply swipe right on the landing page of the application and then click the *Establish Connection* option in the list view that appears. From this screen, the user will see a couple of status bars, which indicate whether the device is connected to the microcontroller's wireless access point and whether the device is ready to transmit data via the established socket. From this point, the user can select an available Wi-Fi network and choose whether they want to connect to it. Upon success, the status bar will now say that the device is connected and will ask the user if they are ready to transmit data. If the device can successfully connect to the socket then another notification will display, informing the user that they will now be able to begin sending control signals and receiving video from the microcontroller.

Upon opening the mobile application, the first thing that the user will see is a status bar indicating whether the device is currently connected to the microscope. This status bar will either say in green text that the device is connected, or it will say disconnected in red text.

To reach the interface where the connection process takes place, the user can simply swipe right on the landing page of the application and then click the *Connect Devices* option in the list view that appears. From this screen, the user will see a couple of status bars, which indicate whether the device is connected to the microcontroller's wireless access point and whether the device is ready to transmit data via the established socket.

From this point, the user will be able to select an available Wi-Fi network and choose whether they want to connect to it. Upon success, the status bar will now say that the device is connected and will ask the user if they are ready to transmit data. If the device can successfully connect to the socket then another notification will display, informing the user that they will now be able to begin sending control signals and receiving media from the microcontroller.

5.4.8 Software Development Methodology

The software development methodology that we chose for this project is the Agile development method. The Agile methodology revolves around churning out code at a rapid pace and attending to issues quickly as they arise. This software methodology was chosen because of the adaptability of the model so that problems can immediately be addressed as they arise.

Additionally, it is difficult to map out everything before the start of development process, so having a more flexible model is often preferred. We wanted to strongly emphasize this development methodology to prevent regressions and preventable bugs in the development process. Since our development team consists of two members, lacking a proper development methodology could have easily introduced poor coding habits and lead to the cowboy coding style, where each developer has complete autonomy over their respective portion of the project.

While the developers for this project had a bit of autonomy over their respective sections, it was still important that a proper methodology was followed so that there was consistency in terms of coding style, documentation, and design practices. Not only was this beneficial to the developers during the project, but it will make the ramp-up process much easier for any future contributors.

To monitor our issues, we used the GitHub Issues platform to track all the problems that we encountered during the development process. Since the programming for this project was split between the embedded system and the mobile app, the code for each section was hosted in its own repository.

For this project, the Computer Science student focused primarily on the mobile application and its components and the Computer Engineering student focused primarily on the embedded applications. Considering both the developers were responsible for their own respective sections, any issues encountered needed to be personally monitored. To encourage good development habits and to ensure that the developers stayed on top of their work, we conducted meetings and code review sessions in order to hold each other accountable.

5.4.9 Remote Code Repository Usage

The software that we write for this project needed to be hosted somewhere remotely. Since this is a project that will likely have more contributors in the future, we wanted to host our codebase in a location that was accessible to many people instead of just storing everything locally on someone's personal computer. In the following sections, we discuss our rationale behind choose GitHub as our primary code repository and briefly explain the useful features that it comes with.

5.4.9.1 GitHub

For hosting, we used UCF's organizational GitHub account, which allowed us to create private repositories for both the mobile application and all the embedded code that went on the microcontroller. Using the organizational account was beneficial for all parties because it puts all our work in one place where we can directly access and modify it. It also allowed us to use the GitHub Issue Tracker to keep track of bugs and monitor our progress throughout the development cycle. Additionally, the teaching assistants and anyone who has access to our repositories had the ability to view our code and do whatever they want with it once the course ends.

5.4.9.2 GitHub Issue Tracker

As previously stated, we used the GitHub Issue Tracker to track any to-dos or bugs that we encountered during the development process. We considered using a specific project management tool like JIRA for this, but since there were only two people doing any kind of development, using something like JIRA seemed like it would be overkill. Additionally, the GitHub issue tracker is integrated with GitHub, which is where our code repositories were held, saving us from having to use and monitor two different applications during the development process.

GitHub Issue Tracker was also advantageous for us because it allowed for code reviews, and task and assignment tracking. While there were only two people on this project responsible for writing the software, it was beneficial for both of us to each check each other's code during the process and give feedback and suggestions for how the code can be improved. Not only did this help us keep our code up to the highest standard, but it also ensured that we knew what the other person was currently working on and allowed us to offer suggestions or bounce ideas back and forth.

5.5 Power System

Creating a proper power system for each chip is essential for the system to work properly. If too much power or current is supplied to each system, the entire design could be burned up and created useless. This would be a disaster, especially during the printed circuit board stages of designing. An important guide for these power designs is each chip's individual data sheet. These data sheets gave important notes for each pin's power input and limit. From here, voltage regulators could be designed and pin connections could be determined. To begin, we looked at the powering of the Wi-Fi module, which is pictured in Figure 41. In this figure it is shown that the input voltage will need to be 3.3 V and that various other pins will be connected to the ground layer of the printed circuit board.

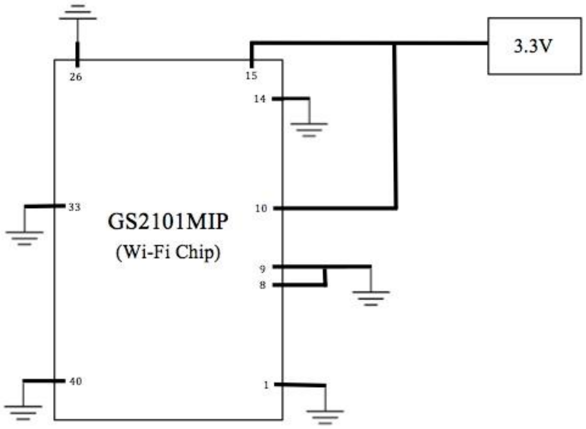


Figure 41: Power Design for Wi-Fi Module

From here, we moved to designing the power layout for the image sensor. For this particular module, it was found that there needed to be two voltage inputs, those being 2.8 V and 1.8 V depending on the pins and their function in the system. It was also required that both input voltages by correctly decoupled by capacitors. The S_{DATA} and S_{CLK} pins were also required to have 1.5 kΩ resistors connecting those pins to the V_{DD_IO} power supply pin. With all of these requirements taken into consideration, the diagram for the power system designed for the image sensor is shown below in Figure 42.

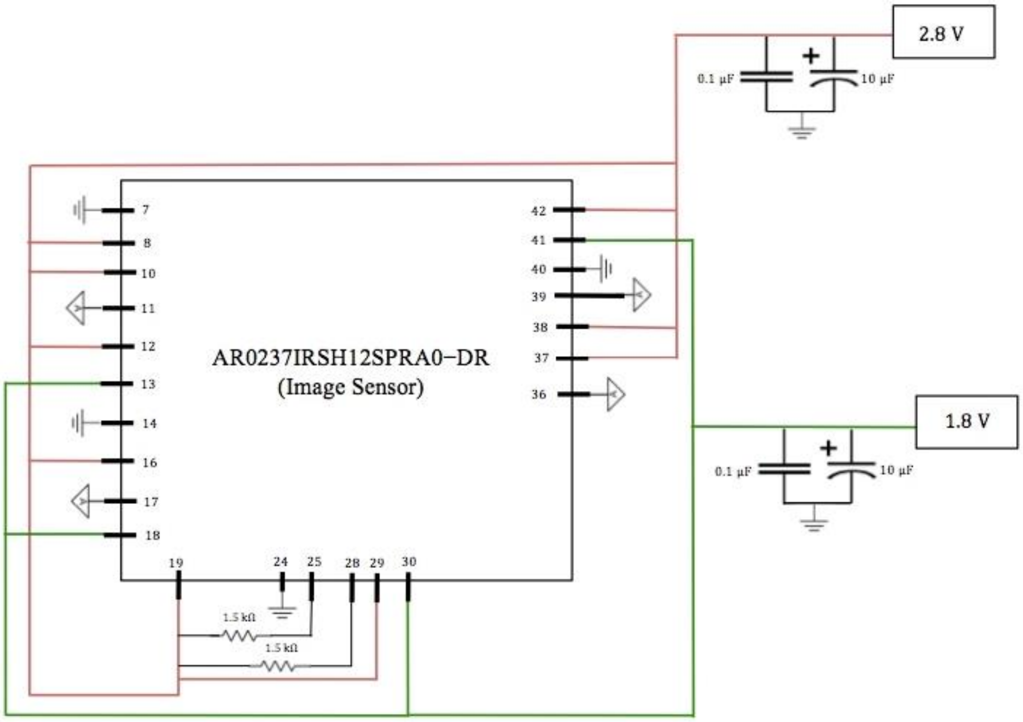


Figure 42: Power Design for Image Sensor

Finally, we were able to design the power system for the microcontroller. Using the datasheet as a guide, coupled capacitors were needed once again in order to regulate the voltage input. Like the Wi-Fi module, the input for this system is 3.3 V.

As with the other inputs, the voltage for this system needed to be decoupled by capacitors. The values of these capacitors, however, were found based on the amount of inputs that voltage would go to. This controller had eleven input voltages of 3.3 V, so the total coupling was a 1.1µF capacitor and a 4.7µF capacitor, which are both shown in the diagram for the microcontroller power inputs.

V_{DDA} also needed to be decoupled with a 1.1µF capacitor. Pin 6 shows the input straight from the voltage regulator that will be outputting 3.3 volts. The pins that require grounding are shown as well. The full diagram showing the pins and their power connections are shown below in Figure 43.

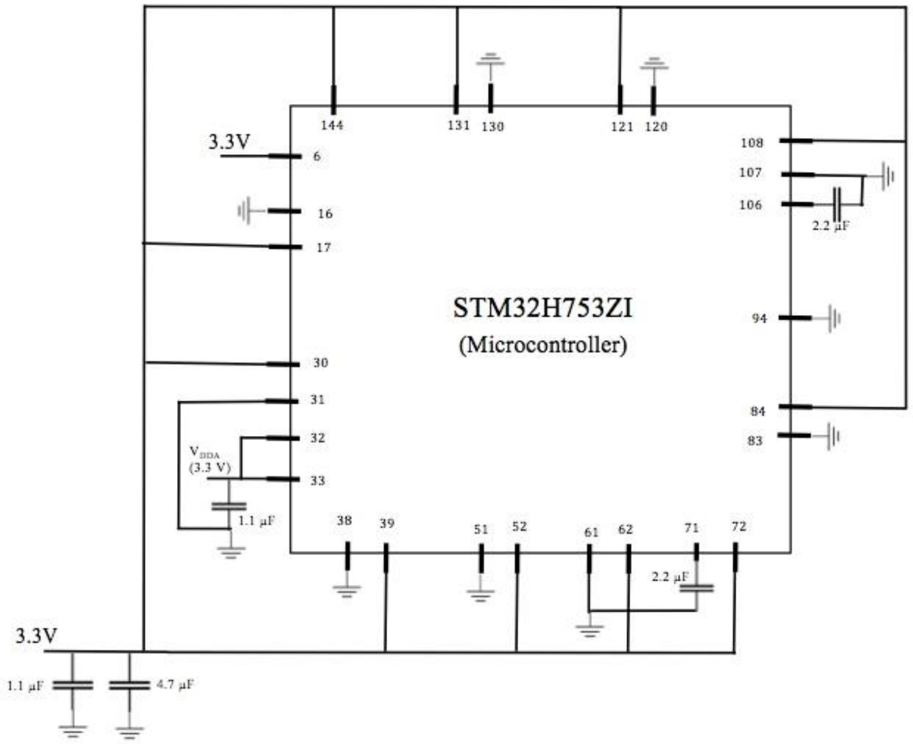


Figure 43: Power Design for Microcontroller

For the alternative plan, the batteries chosen will supply a 7.4V input that will be regulated by a switching regulator that outputs 5V. This 5V output from the regulator will be powering the Raspberry Pi and the upper lighting system. A 9V Li-Ion battery will be powering the base light system, which contains seven LEDs. This power diagram is shown below of how the powering system will operate in Figure 44.

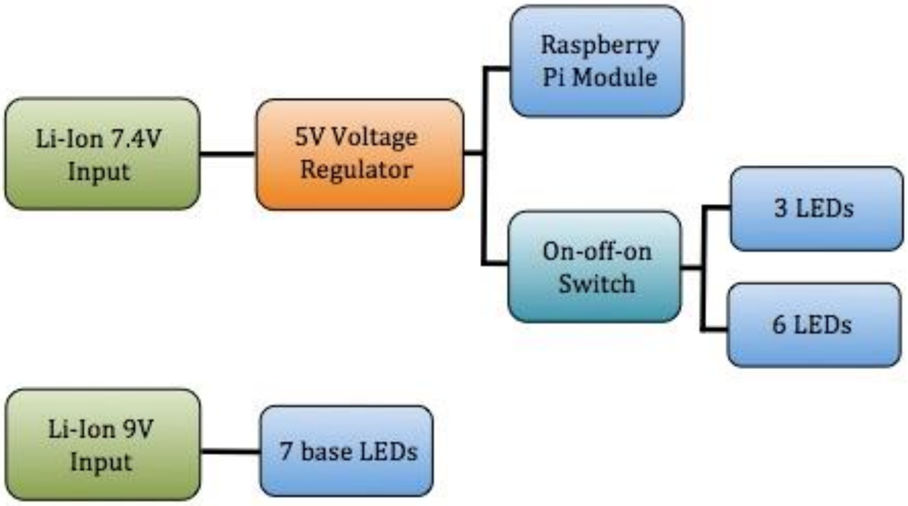


Figure 44: Power Block Diagram

5.6 Hardware Design

After finding the power systems for each of these chips, we moved on to designing the hardware layout of the system. For this process, the pin charts from the data sheet were useful to know which sections of pins would connect to where on each chip. Knowing the function of each pin was essential for this designing process. Any wrong pin connections would mean that the programming might not control the correct input and output.

Throughout the design of these hardware interconnects, STM's Cube software was used to generate the initial pin mapping of the microcontroller, which was then verified against the manual for two-factor validated.

First, the Wi-Fi chip and the microcontroller connections were determined. These connections are important because the microcontroller will need to send information to the Wi-Fi sensor so that the phone or other application-based system will be able to receive it. This function is important for both the computer engineering and computer science sides since these chips will communicate through programming code. Without the correct coding and pin connections, the circuit and the communications for the system will not work properly. Knowing the importance of making these connections, the schematic was created, and is shown below in Figure 45.

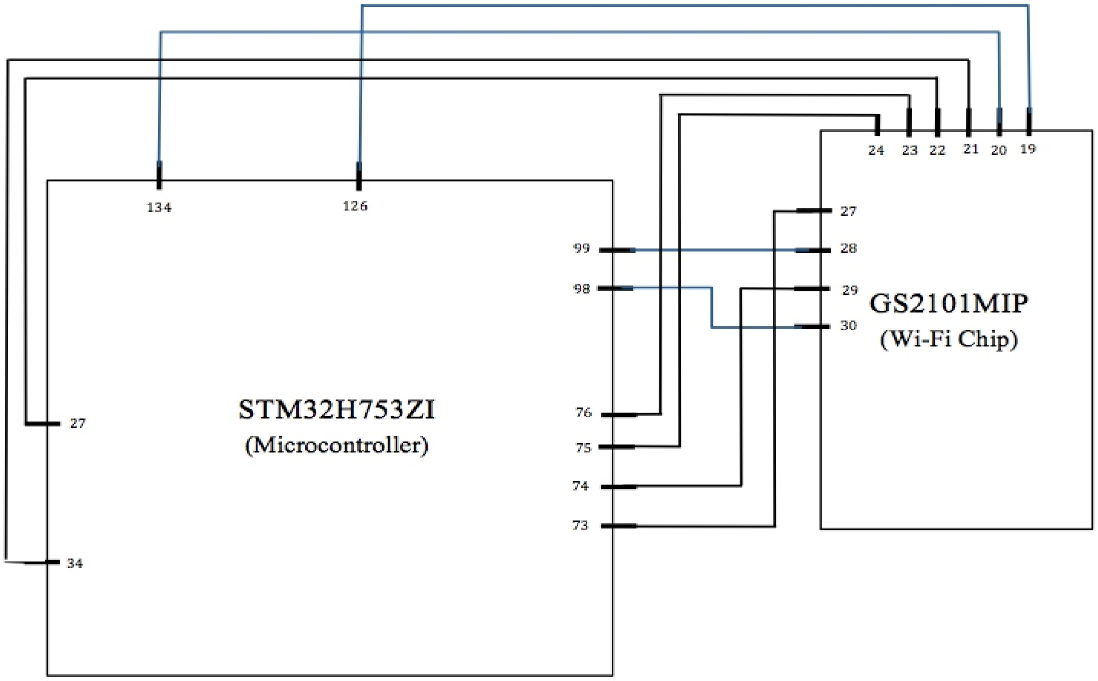


Figure 45 - Schematic for Microcontroller and Wi-Fi Connections

For the next portion, we researched the best way to program and control many LED outputs for the lighting of the system. For this, the microcontroller could be used to turn the lights on and off, but the current of each pin would not be enough to light the system. Knowing this, it was found that using an LED driver chip would be the best option. By connecting the "com" channel chip to the battery, more LEDs would be able to be illuminated. We wanted there to be both infrared and white light LEDs present for lighting. By having multiple pins controlling multiple pins, the user would be able to control how many LEDs would illuminate, as well as the types of lights on at one time. Given these specifications, and after calculating the resistors needed to keep the current at 20mA, the hardware diagram was created and is shown below in Figure 46.

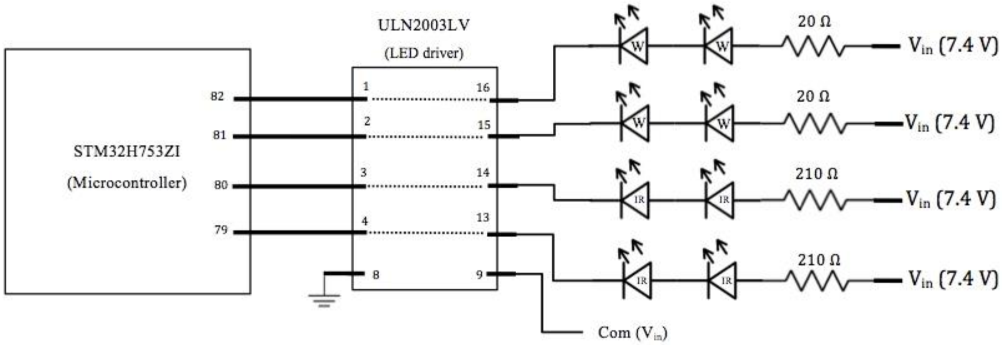


Figure 46: Schematic for LED Connections

After this, we made the connections from the image sensor to the microcontroller. These connections are essential for programming the format of the image. These pins are also for processing the image and transferring it to the Wi-Fi chip that, in turn, sends the image to the application. The pin connection layout for the image sensor and the microcontroller are shown in Figure 48, pictured below.

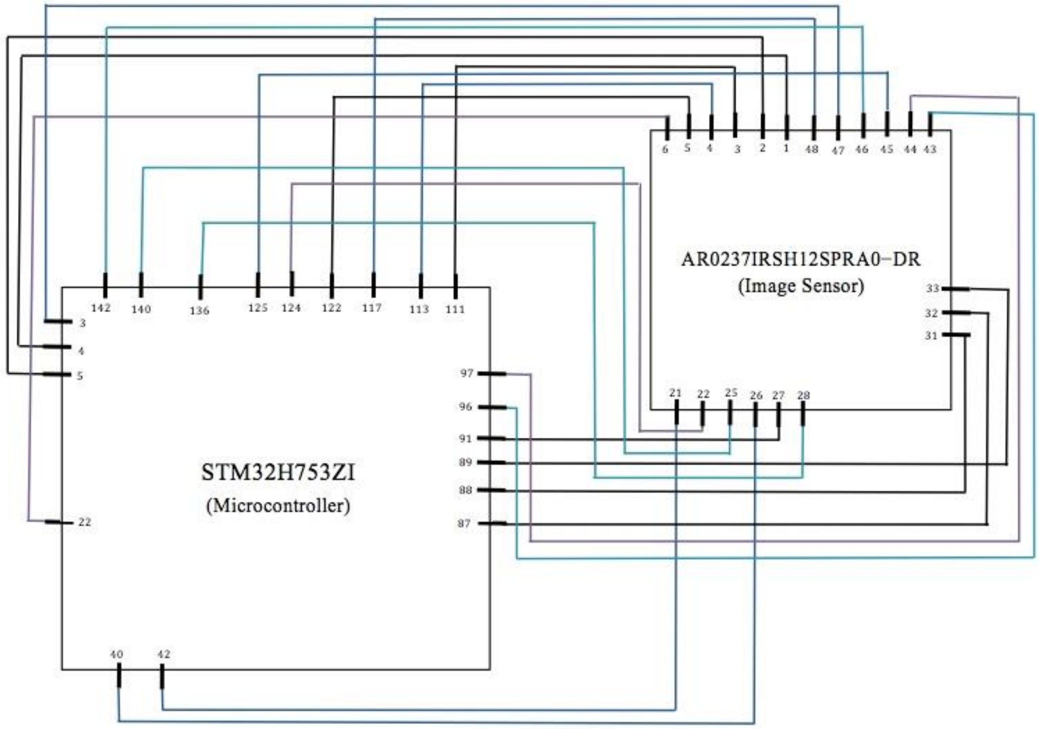


Figure 47 - Schematic for Microcontroller and Image Sensor Connections

5.7 Printed Circuit Board Design

For this project, printed circuit boards were necessary for the construction of the product. As discussed before, the printed circuit boards give structure and combine the components necessary for the design in an orderly and stable system. This section of the paper discusses the completed PCB designs of both the original and the alternate plan for this project.

5.7.1 Completed Main PCB Design

Due to time constraints and problems that were discovered during the coding process, the original design for this project could not be tested. Therefore, no testing results are available to show for the first planned idea. The printed circuit board for this initial plan was designed and created with all of the necessary components. The full design schematic is shown below in Figure 48 along with the printed circuit board layout below in Figure 49 and the Gerber file images in Figure 50 and Figure 51.

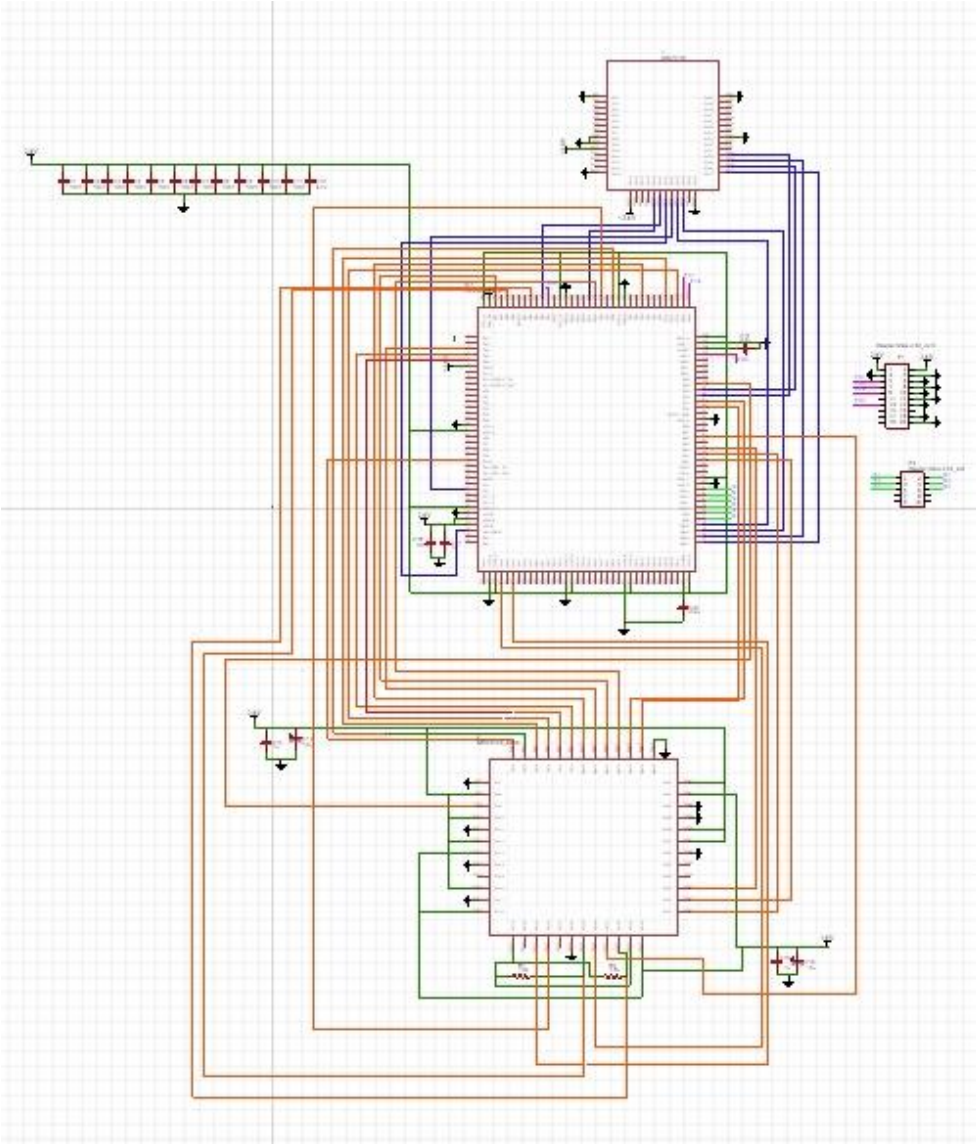


Figure 48: Completed Main PCB Schematic

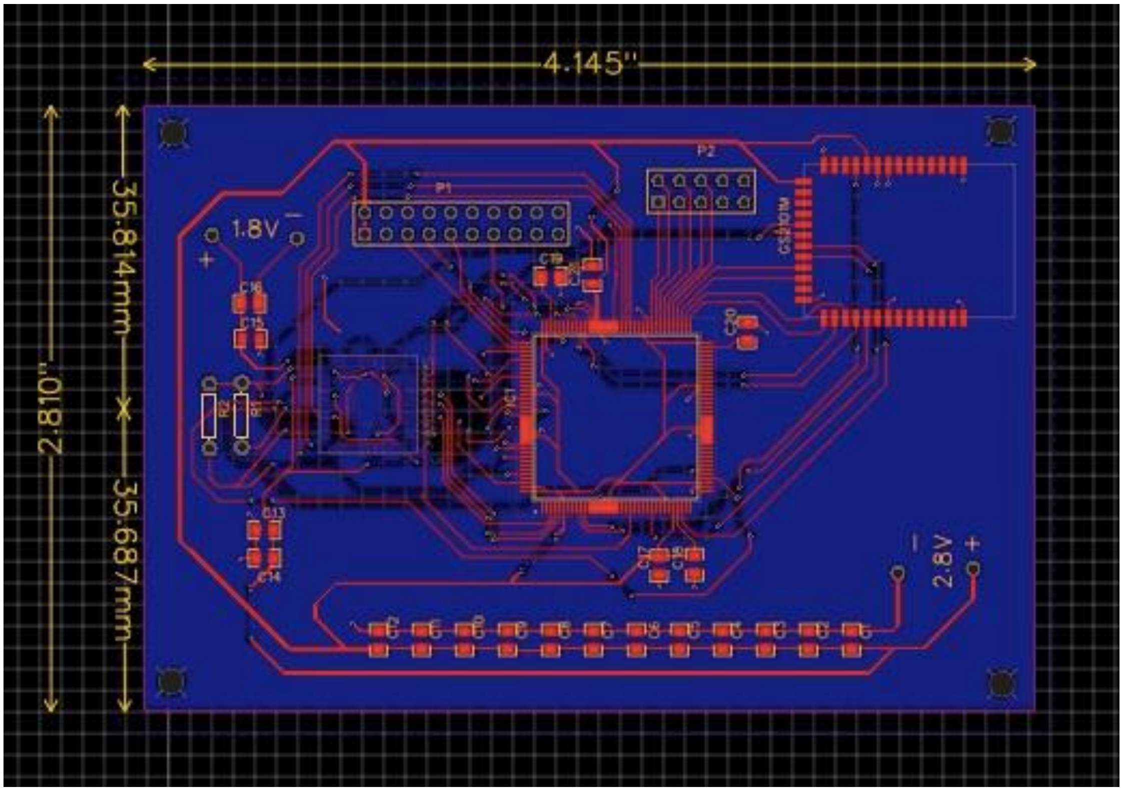


Figure 49: Completed Main PCB Diagram

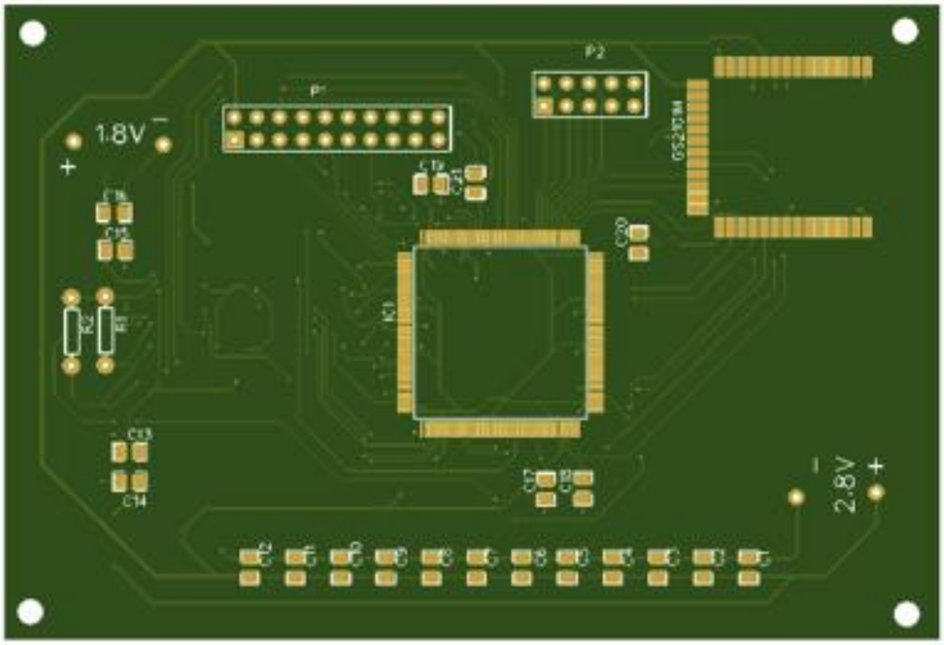


Figure 50: Main PCB Gerber Top View

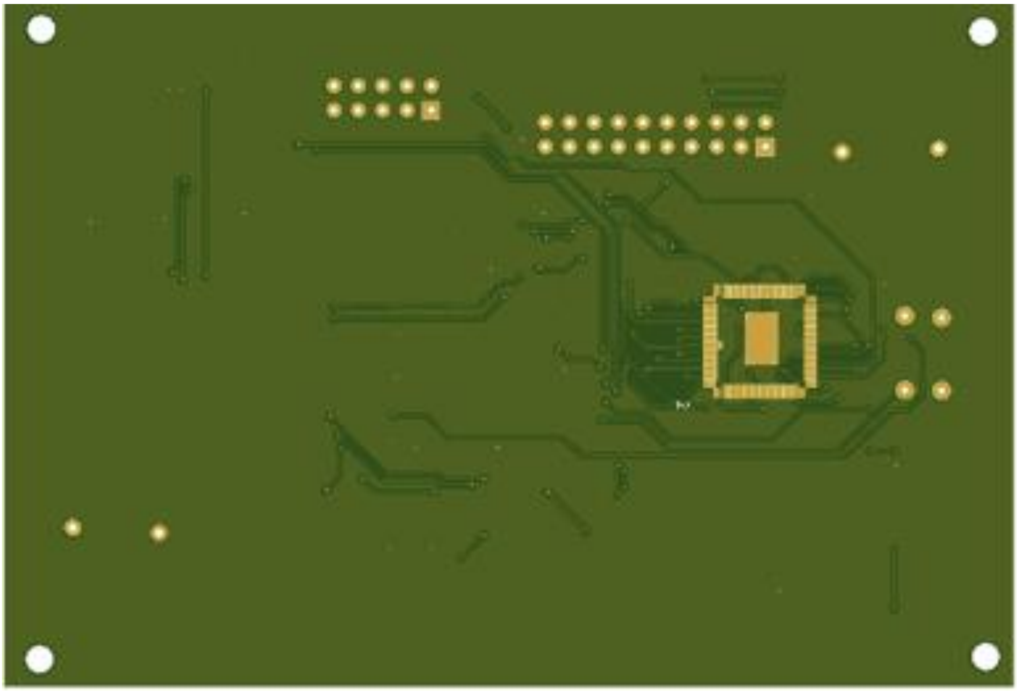


Figure 51: Main PCB Gerber Bottom View

Figure 51 above gives a better idea of the layout of this board. The microcontroller and Wi-Fi chip were located on the top of the board along with any of the surface mount components, and the image sensor was on the bottom of the board, allowing better isolation for placing the tube around the sensor. There was a connection for a JTAG port so that code could be programmed into the microcontroller, as well as a port to connect the top LED board for control of the white and infrared LEDs. Power inputs would come from the respective outputs from the voltage regulator board.

5.7.2 Voltage Regulator PCB Design

The noise from three switching regulators would affect the reception of the Wi-Fi module, so the regulators were put into one printed circuit board. The components used were all surface mount so that the footprint and component packages could remain small and compact. This compact size would vastly increase the portable nature of the device. Soldered wires would connect the voltage input and the voltage output for each of the regulators. Some regulators had more components than others, and one design did not work correctly. However, the designs included in these diagrams worked perfectly for the voltages that needed to be output to the main PCB. These voltages could also be easily tested on this separate board to be sure that the voltage output did not harm the rest of our system.

This board contained the three regulators as discussed earlier and had output voltages of 1.8V, 2.8V and 5V for the various voltage inputs that were needed for the system. This printed circuit board design is shown in Figure 52 as well as its Gerber file image in Figure 53.

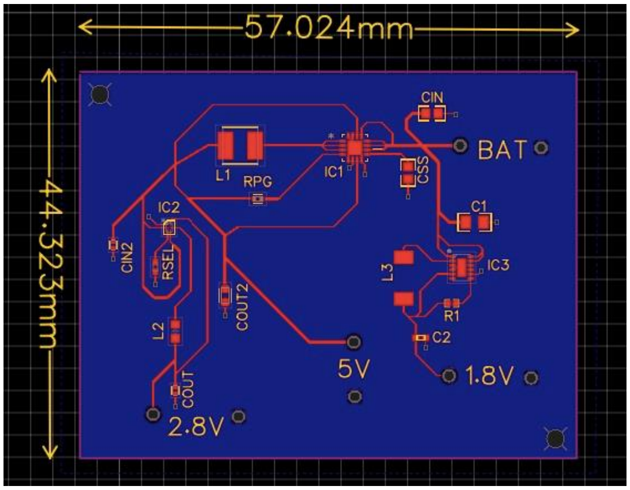


Figure 52: Original Voltage Regulator PCB Design

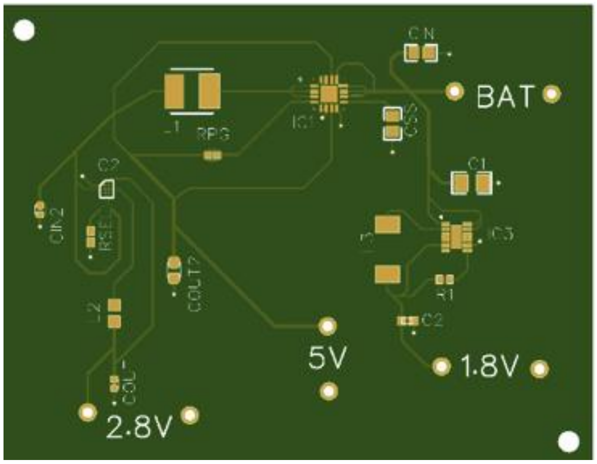


Figure 53: Original Voltage Regulator PCB Gerber View

For the alternative plan, printed circuit board designs were also created. The regulator design followed that of the Webench design previously discussed and the design for the completed board is shown below in Figure 54. This board included a USB port attachment because the power will be sent to the Raspberry Pi through a USB to Micro USB cable. The Gerber file of this board is also included and shown below in Figure 55.

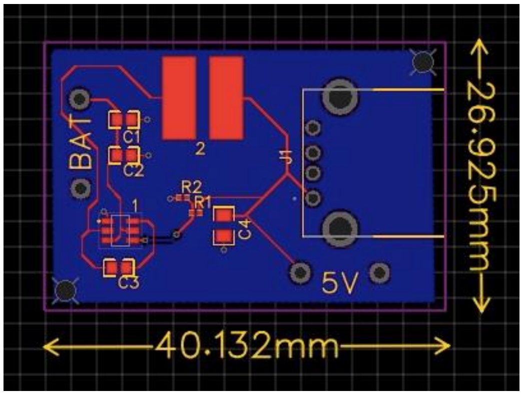


Figure 54: Alternate Voltage Regulator PCB Design

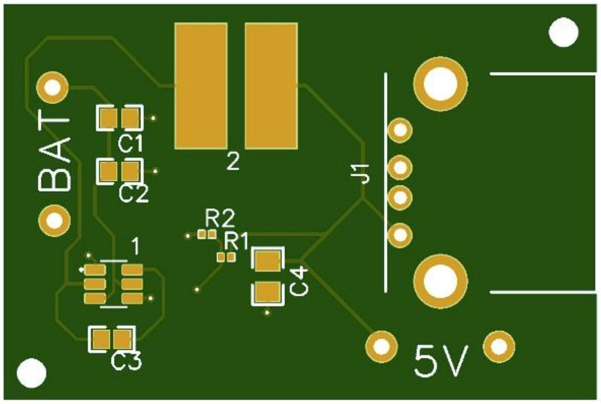


Figure 55: Alternate Voltage Regulator PCB Gerber View

5.7.3 Lighting PCB Design

For the top light system, an LED driver would control the individual branches of white and infrared light. This system was designed in a circular shape in order to fit around the tube which fit the lens for our system. The printed circuit board design shown below in Figure 56.

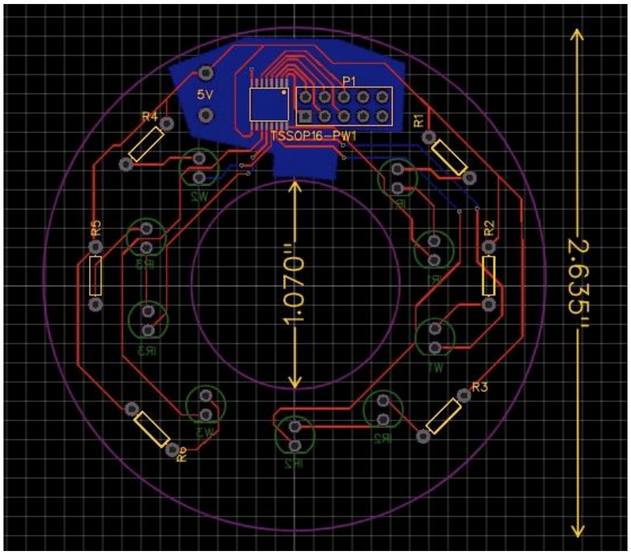


Figure 56: Original Above LED PCB

For the lighting systems, there were also circuit boards designed. The top lighting system included nine LEDs that the user can switch between three and six diffuse LEDs in order to control the amount of lighting that might be needed. An on-off-on switch on the side of the structure would control this option. Voltage went into the switch and depending on either 1 or 2 selected would output voltage to that branch of the system. The top lighting printed circuit board design is shown below in Figure 57.

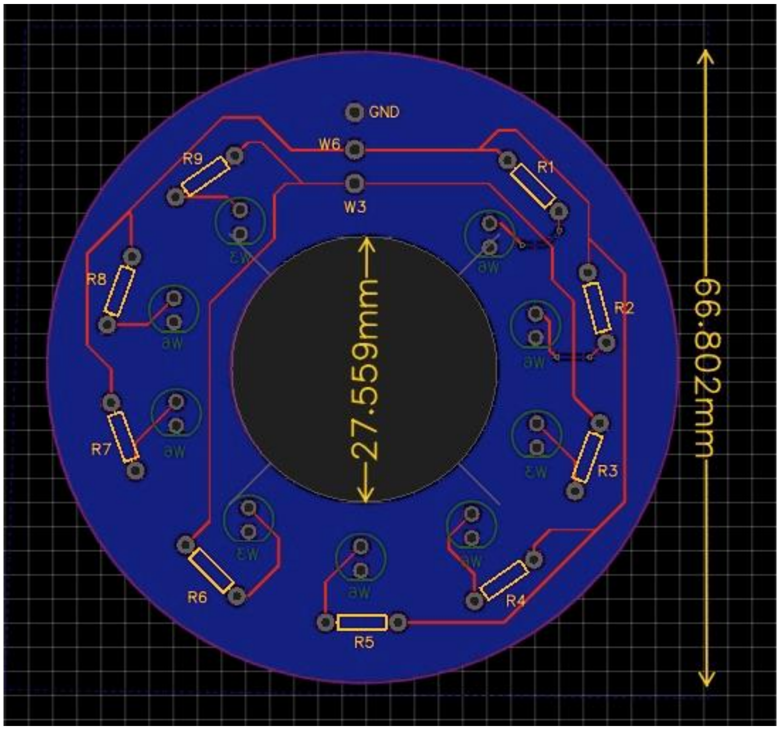


Figure 57: Alternate Above LED PCB Design

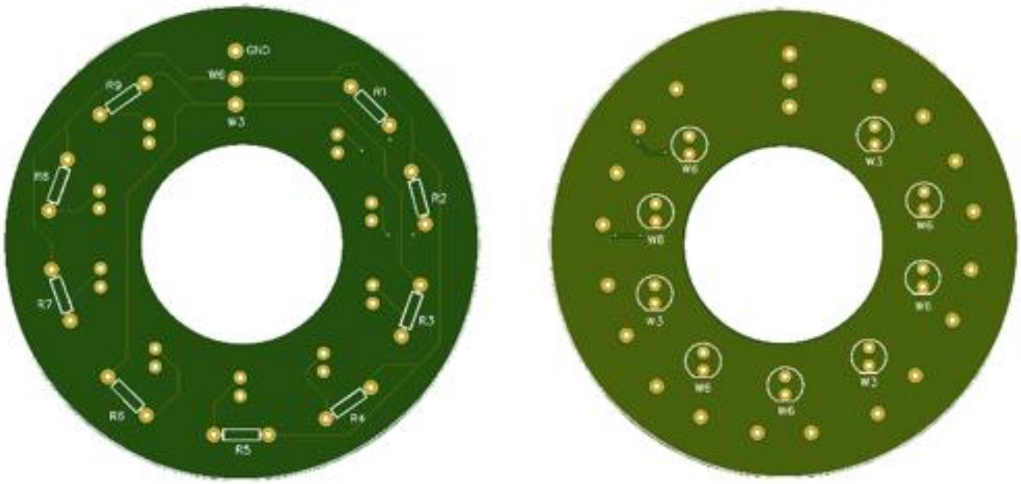


Figure 58: Alternate Above LED PCB Gerber View

For the base lighting system, there were seven LEDs, which would illuminate sample slides. This was a removable disc that had a diffusion filter above it for an even light distribution. This design is shown in the figure below, Figure 59 and Figure 60. This was an essential portion of our design that, without it, we would not be able to view any sort of thin sample. This system was planned to be powered by a Lithium Ion 9V battery. This battery was chosen because of its larger current capacity. This would increase the battery life for this already low power system.

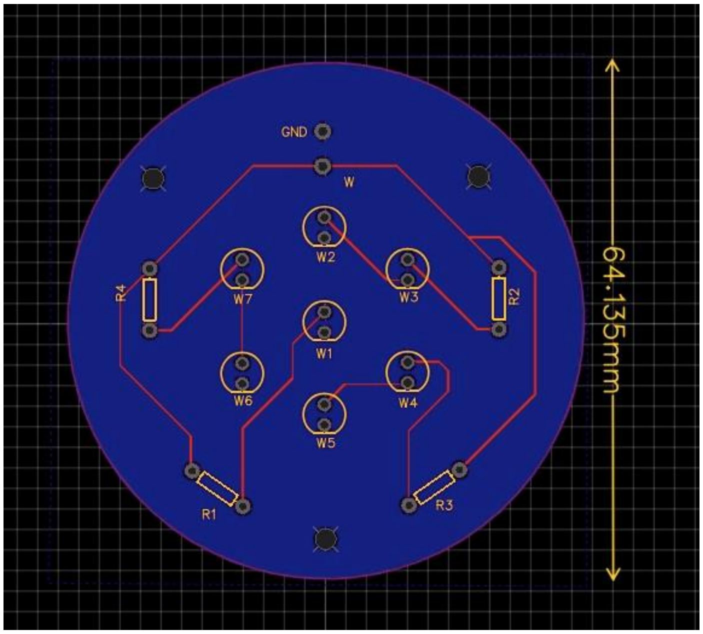


Figure 59: Base LED PCB Design

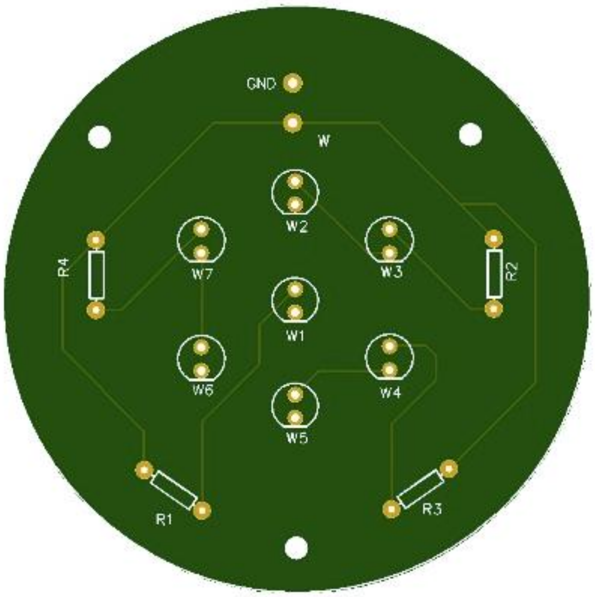


Figure 60: Base LED PCB Gerber View

5.8 The GS2101 Wi-Fi Chip

The GS2101 will be responsible for high-frequency, high-throughput transmissions from the MCU to the mobile platform and low-frequency, near-zero-throughput transmissions from the mobile application to the MCU. Its embedded message-passing will be through a UART connection, with a four-bit SDIO interface for inbound data flow.

This device will have no processing responsibilities, and its application processor will be dedicated to constant availability for the bidirectional microcontroller-to-mobile link it is designed to provide. Inter-processor communications are programmatically automated at the device firmware level, leaving only the communications interfaces between this module and the two devices it connects to be designed.

5.8.1 Summarized Feature Set

The utilized features of the device are shown in Table 13 below, and their complete design explained in subsections:

Feature	Description	Use
Two Paired Processors	One dedicated application processor and one dedicated wireless control processor.	Self-explanatory
UART Interface	Hardware flow-controlled I/O Interface	Sending commands to and receiving control signals from this module
SDIO Interface	1- or 4-channel SDIO link	Data transfer to Wi-Fi module (4-channel)
AT Command Interface	ASCII-based, interpreted application command library	Readable abstraction layer
Wi-Fi Direct	Simplified Peer-to-Peer Wi-Fi Connection Protocol	Establishing a link between the embedded and mobile systems.

Table 13 – GS2101 Feature Set Overview

Note that all PCB-level I/O interfaces connect to only the application processor through an I/O reader and mid-sized buffer.

5.8.2 Interfacing

All connections and related details are detailed below. It’s notable that this module supports other control configurations; bi-directional SDIO for both control and data, single-channel SPI exclusive mode, and UART exclusive mode, but the throughput requirements of our application demanded more, and the UART plus 4-bit SDIO connection modality is the reason this Wi-Fi chip was selected.

5.8.2.1 UART and SDIO Connections

To handle all logic-level behaviors, including control signals sent from the mobile application to the MCU, the UART connection will be used. These messages are all formatted per the device-specific AT Commands which introduce a small, negligible overhead during operation, and require interpretation code on the side of both the MCU and the mobile application. These commands will be detailed in the section below. A baud rate of 921600 is supported by both the Wi-Fi module and the

This module’s SDIO channel will be set up to operate as a four-bit parallel, slave-mode interface, effectively spoofing an SD card. By specification, it supports a maximum 40

MHz host clock, though there is a notice that it has only been verified up to 33 MHz, which is shown to be more than sufficient by the following calculations:

$$33\ MHz * \frac{.5\ bytes}{transfer} = \frac{16.5MB}{second}, \qquad \frac{1\ second}{16.5MB} * \frac{2304\ bytes}{WiFi\ MTU} = 139.6\ \mu s\ (period)$$

For a 100kB JPEG, encoding takes 4ms, per timings in documentation:

$$\frac{1s}{16.5MB} * 100kB = 6.06ms\ \approx 4ms$$

The transfer rate of the SDIO interface can almost keep up with the JPEG codec itself, which is by far the fastest step of the conversion process, being rate limited by the RGB to YCbCr prerequisite conversion.

5.8.2.2 AT Commands

The application processor has a HAL language which supports and greatly extends the old Hayes command set, now termed AT commands, and it’s through this command format that all instructions will be passed by the MCU and mobile application. The command format follows a readable syntax, details can be found in Appendix A.

5.8.3 Wi-Fi Initialization

The Wi-Fi initialization was handled entirely by the Raspberry Pi, with all parameters, up to and including the DHCP server and wireless access point. These section includes many features, including Wi-Fi direct that were initially considered for this project but were later abandoned. Information about the final networking procedures used are explained in section 5.8.3.2 below.

5.8.3.1 Wi-Fi Direct

The original chosen method for establishing a line of communication between the microcontroller and the mobile device was to use Wi-Fi direct. Wi-Fi direct is a technology that allows two devices that support Wi-Fi direct to be able to sync together and do things like print, share, sync, and display, all without having join a wireless network. This decentralization makes connecting devices and sharing data much simpler than the traditional means of connecting to an intermediate network or by manually wiring a connection between the two devices. Wi-Fi direct was initially introduced because the required faster data transfer speeds couldn’t be obtained by existing wireless standards, such as Bluetooth. In our project, this played a large factor in our initial decision to use Wi-Fi direct as opposed to Bluetooth, as the bit rate for transferring data over Bluetooth was unacceptable for our needs.

As mentioned before, Wi-Fi direct is very simple to use and can be found in most electronic devices today, including all devices that run the Android operating system since Android 4.0 (Ice Cream Sandwich), all Apple devices since iOS 7, and even various hardware devices, including portable media players, headphones, and printers.

For a P2P connection to be established, a discovery process must first occur. This discovery process is simply a means of showing a list of potential Wi-Fi direct peers that your device can connect to. This process can be done on most computers and almost all cellular devices by going to the advanced Wi-Fi settings on the device and clicking the Wi-Fi direct option. From this point, a list of discovered peers should appear on the screen and the user can then click on one of the available connections. The next step sends a connection request to chosen device and if the request is accepted, a negotiation process begins. This negotiation process involves determining which device is going to be the group owner and which device is going to be the client. Once those roles have been determined, a P2P group has been formed and data can now be transferred back and forth between the two devices, as well as between any other clients that join the group.

5.8.3.1.1 Group Formation and Election Process

To determine which device becomes the group owner and which device becomes the client, an intent value is sent to the opposing device in a manner that is like the three-way handshake of the transmission control protocol model. The intent value is a numeric value ranging from zero to fifteen that both P2P devices will set for themselves during the negotiation process. The larger the value, the higher the probability that the device associated with that value will become the group owner. This intent value is typically set in a configuration file, commonly referred to as *wpa_supplicant.conf*.

If the two intent values are the same, a tie-breaker bit is included in the negotiation request and is randomly set. In other words, one device will have this bit set to one and one device will have this set to zero. Once the negotiation process begins, a control structure is then used to determine who becomes the group owner and who becomes the client. The diagram shown below Figure 61 details control flow for this election process.

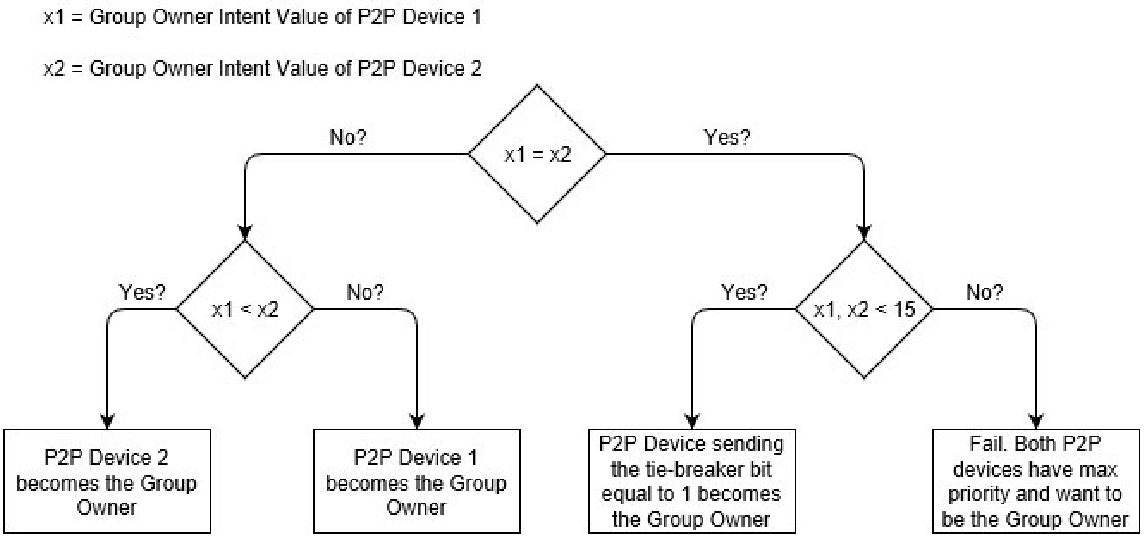

x1 = Group Owner Intent Value of P2P Device 1
x2 = Group Owner Intent Value of P2P Device 2
x1 = x2
No?
Yes?
x1 < x2
Yes?
No?
x1, x2 < 15
Yes?
No?
P2P Device 2 becomes the Group Owner
P2P Device 1 becomes the Group Owner
P2P Device sending the tie-breaker bit equal to 1 becomes the Group Owner
Fail. Both P2P devices have max priority and want to be the Group Owner

Figure 61: Group Owner Election Process

With this diagram in mind, it is important to note that some Wi-Fi direct groups might be impossible to form depending on the established configuration settings. If the two devices have intent values set to fifteen, then the connection will not be formed because both devices require they become the group owner.

This election process is the de facto standard for creating Wi-Fi direct connections and is what you will see between most devices. As mentioned previously, it’s important that intent values are set carefully to ensure that a connection can be properly established. When using Wi-Fi direct programmatically, such as in our Android application, it was possible to set the intent value by using the *Wi-FiP2pConfig* object and by setting the member variable, *groupOwnerIntent*, to a value between zero and fifteen.

5.8.3.1.2 Group Owner vs. Client

Wi-Fi direct devices communicate with one another by establishing a P2P group. This P2P group consists of exactly one group owner and one or more clients. Like a standard wireless network, an access point is used to allow incoming clients to gain access to the P2P group. In a Wi-Fi direct P2P group, the access point isn’t a physical networking hardware device like in traditional networks, but instead it is one of the P2P devices that elects to serve as the access point and perform the same functionality as traditional access points. In addition to that, the device that acts as an access point will also occasionally act as a P2P client. The device that is designated as the access point is referred to as the group owner, whereas all other devices in the group are referred to as P2P clients. Like a traditional access point, the P2P group owner will announce itself and broadcast a signal that will allow other clients to then connect to the group.

There are a few caveats to consider when using this group owner/client relationship in Wi-Fi direct. One of which being that Wi-Fi direct doesn’t allow transferring the role of the P2P group owner role within the group. This means that no matter what might occur, the group owner that was determined during the election process will remain the group owner until the group is dismantled, and at no point can that role be transferred to another device. In addition to that, if the P2P group owner leaves the group then the group will automatically be dismantled and will need to be re-established. Because of this, it’s very important that the group owner is carefully chosen and is known to be a stable device that will ideally be kept in range of all the other devices.

The intent values can end up playing a very critical role in the group formation process. The probabilistic nature of the election process makes it impossible to ensure that the device you want to be the group owner will be the group owner while guaranteeing that a group will be formed. As we’ve mentioned in the election process section, if two devices have their intent values set to fifteen and try to form a P2P group, the group will not be able to form because both devices will be competing to be the group owner.

5.8.3.1.3 Android Implementation

The process for joining a Wi-Fi direct group on Android is the same exact process for joining a Wi-Fi direct group on any other device. When we tested our Wi-Fi direct in our mobile application, the code that we used to do this involved the use of broadcast receivers, peer-to-peer managers, and various other P2P configurations that are built into native Android.

The first step of this process was to discover all of the available Wi-Fi direct peers that are available to us. We initially needed to create two variables of type *Wi-FiP2PManager* and *Wi-FiP2PManager.Channel*. The first object, the manager object, was used to grab the native WI-FI_P2P service that can be found on all Android devices running Android 4.0 (Ice Cream Sandwich). This service was then initialized using the *initialize(Context, Looper, Wi-FiP2pManager.ChannelListener)* method. This method registered the application with the Wi-Fi framework and had to be called before any P2P operations could be performed. The return value of this method is of the type *Wi-FiP2pManager.Channel*, so we set that equal to our previously defined variable.

Once the application was registered and ready to go, the *discoverPeers(Wi-FiP2pManager.Channel, Wi-FiP2pManager.ActionListener)* method could be called from our *Wi-FiP2pManager* variable which initiated the peer discovery phase of the process. This process involved scanning for available Wi-Fi peers for the purpose of establishing a connection with them. The call to this function returns immediately after sending a discovery request to the Wi-Fi framework. The application is then notified of a success or failure to initiate the discovery process through the ActionListener callbacks, *onSuccess()* and *onFailure()*.

Upon success, the list of peers that were discovered can be obtained by calling the *requestPeers(Wi-FiP2pManager.Channel, Wi-FiP2pManager.PeerListListener)* method on our P2P manager object. Similar to the *discoverPeers* method, this method has a callback called *onPeersAvailable(Wi-FiP2pDeviceList)*, where when activated, an argument is passed of the type *Wi-FiP2PDeviceList* that contained a list of all of the available P2P devices that were found during the discovery process. The *Wi-FiP2pDevice* object has some important fields and methods that we will utilize during the connection process. The class diagram shown in Figure 62 illustrates what the various methods and fields are for this object.

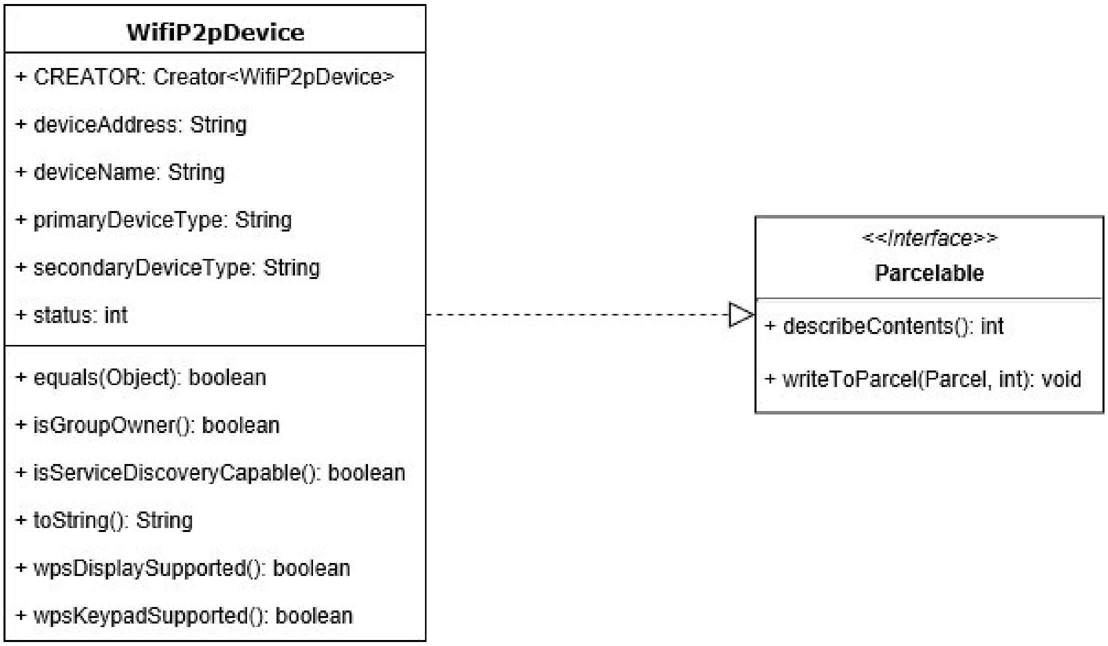
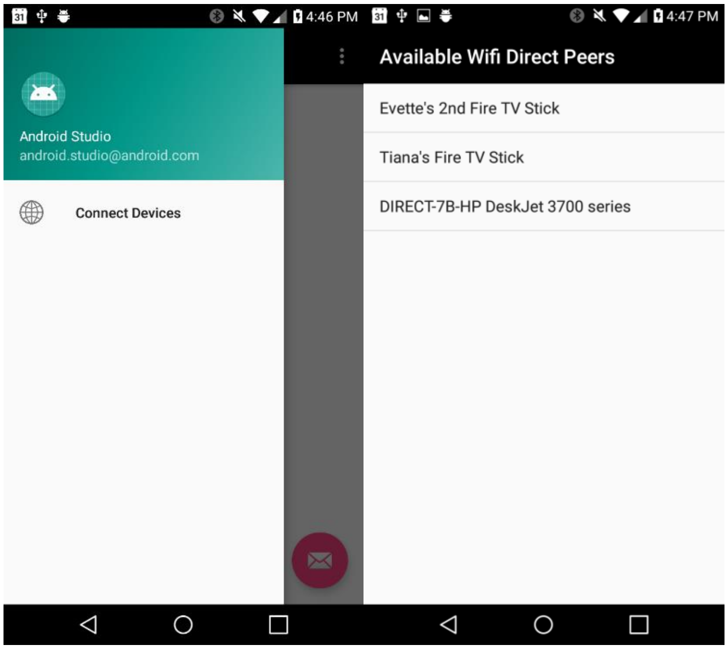

WifiP2pDevice
+ CREATOR: Creator<WifiP2pDevice>
+ deviceAddress: String
+ deviceName: String
+ primaryDeviceType: String
+ secondaryDeviceType: String
+ status: int
+ equals(Object): boolean
+ isGroupOwner(): boolean
+ isServiceDiscoveryCapable(): boolean
+ toString(): String
+ wpsDisplaySupported(): boolean
+ wpsKeypadSupported(): boolean
<<Interface>>
Parcelable
+ describeContents(): int
+ writeToParcel(Parcel, int): void

Figure 62: Class Diagram of WifiP2pDevice

Most of these fields and methods are self-explanatory, but the relationship between *Parcelable* and *WifiP2pDevice* can be slightly confusing. The *Parcelable* interface is an interface for classes whose objects can be written to and restored from a parcel. The parcel object is a generic object that is used for data serialization. In other words, the data contained in the *WifiP2pDevice* object can be flattened and sent out to recipients in a way that allows for maximum performance, which is necessary when dealing with Wi-Fi packets. The *Parcleable* interface requires that classes that implement it also include the non-null static field called CREATOR, which consists of a type that implements that Parcelable.Creator interface. In our situation, the *WifiP2pDevice* is the object the implements the *Parcelable* interface, so it's required to have that field.

Once the list of these *WifiP2pDevice* direct objects has been obtained, we can iterate over the list and store the objects in our own ArrayList. From this point, we now have access to all the discovered peers and can present them to the user so that they can choose to connect to any of them from a list.

A simple example of the discovery process was implemented during the research stage of this project and these were the results obtained from our app developer's apartment shown in Figure 63.



Available Wifi Direct Peers

- Evette's 2nd Fire TV Stick
- Tiana's Fire TV Stick
- DIRECT-7B-HP DeskJet 3700 series

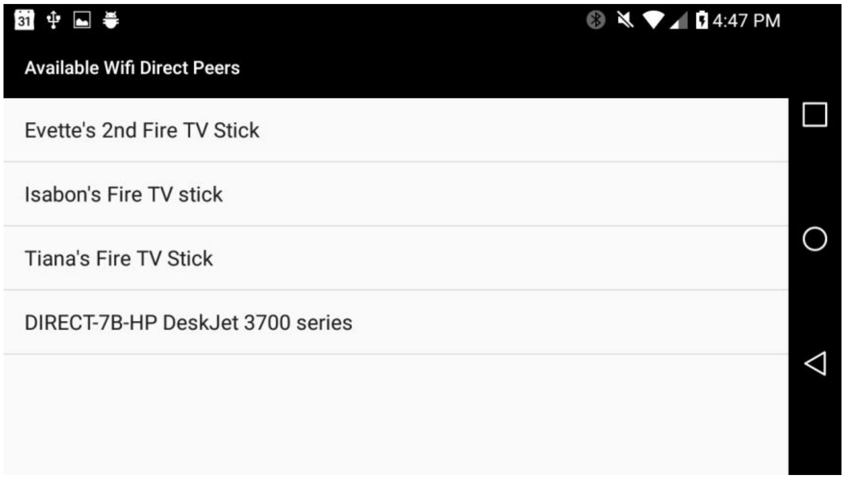
Figure 63: Wi-Fi Direct Discovery Example
Left is Navigation Bar Example Right is Available Wi-Fi Direct Peers

When the application first opens, a landing page will be displayed with some information about the current connection status of the microscope. If the user wants to connect to the microcontroller, they just simply needed to swipe right to open up the navigation drawer and select *Connect Devices* from the list. Once they clicked that button, they were taken to the *ConnectionActivity* which performed the discovery process that was just previously discussed. Once the list of peers was acquired, a simple list view was displayed showing all the different available options that the user could connect to. Once they selected an option, either a confirmation or an error dialog will pop up and informed the user of the result.

The final user interface for this these pages will look significantly better and contain more options, but these mock-ups were created to to illustrate the anticipated flow of the application.

Originally when we were using Wi-Fi direct, the plan was for this *ConnectionActivity* to handle the discovery process in an asynchronous manner, similar to how most devices handle searching for available Wi-Fi networks. What this means is that the device will be able to update the list in real-time so that the information displayed to the user is as up to date as possible. For example, if a potential peer goes offline or disables its Wi-Fi direct capabilities for some reason, we would like for the application to be able to immediately remove that option from the list without requiring any input from the user. This is a relatively simple mechanism to implement and only requires the use of the ReyclerView, which is essentially a dynamic version of a ListView, allowing for objects to be added and removed from the list without the user needing to click a refresh button. With that in mind, we also planned to have a refresh button situated at the top right corner of the screen in case the user decides that they want to perform a manual refresh and check to see if any new peers are available.

An example of this and the usefulness of using flexible layouts is shown in Figure 64 below.



Available Wifi Direct Peers

Evette's 2nd Fire TV Stick

Isabon's Fire TV stick

Tiana's Fire TV Stick

DIRECT-7B-HP DeskJet 3700 series

Figure 64: Flexible Layouts and Asynchronous Processing

As we can see, the data in the list updated to include a new Wi-Fi direct peer that it found during the discovery process. Since this process works automatically, no input is required from the user to get the most current information available.

Like we mentioned in the section on UI and maintaining a respectable appearance within mobile applications, our goal was to ensure that the content that the app is providing was visible in a clear and organized manner, regardless of both the orientation of the device and also the type of device that we are using.

5.8.3.1.4 Authorization

As with any application that alters user data or modifies network connection, our android application required some permissions to be defined in the *AndroidManifest* file, which is an xml file that serves as a configuration file for the entire application. There are four specifics permissions required for using Android direct and they are as follows: ACCESS_COARSE_LOCATION, ACCESS_WI-FI_STATE, CHANGE_WI-FI_STATE, and INTERNET.

The location permission is required because Wi-Fi direct only works within 200 meters, so by utilizing the device’s location, the application can determine which peers are available within that 200-meter area. The rest of the permissions are self-explanatory as Wi-Fi direct interacts directly with the Wi-Fi framework, and modifying any existing state within that framework will require the application to request permission from the user.

To request permission from the user, the application opens a dialog at runtime specifying that the application in question needs access to certain parts of the device, such as location services, Wi-Fi settings, etc. The user can then choose to either allow the application to have access to these features or block them. In the scenario that the user blocks the application from having access to these features, the application will notify the user that application will not be able to run without access and will promptly close out of the application.

5.8.3.2 Wireless Access Point

The following sections detail our final decision to use a hosted WAP (wireless access point) for this project. In addition to discussing the fundamentals of hosting a wireless access point on the Raspberry Pi, justification for our decision to move away from Wi-Fi direct is also included.

5.8.3.2.1 Why WAP over Wi-Fi Direct?

Initially, our plan was to use Wi-Fi direct as the Wi-Fi standard for this project. According to our initial research, Wi-Fi direct met our requirements, specifically transfer speeds up to 250Mbps. However, once we began testing using Wi-Fi direct, we quickly realized that Wi-Fi direct worked fine, but not as well as we wanted it to. The first issue was that the process of connecting devices via Wi-Fi direct was flaky at best. During our testing process, we found that occasionally the WPA command-line interface would experience a strange bug where it would not accept a peer-to-peer negotiation request from the mobile application and would only accept the request if the Raspberry Pi was rebooted or if the WPA Supplicant service was restarted. Additionally, programmatically starting a peer-to-peer connection using Wi-Fi direct was not directly supported. This caused multiple issues because although we were able to initiate the connection process there was not a feasible way to detect when the two devices were connected. This was problematic because the Python script that was configured to connect the devices would just have to continuously try to connect and then attempt to ping the other device after an allotted amount of time. This occasionally worked but was far from a viable solution. In addition to the connectivity issues that we faced while using Wi-Fi direct, we also weren’t consistently reaching the framerates that we desired for this project. During our testing process, we would consistently achieve around twenty frames-per-second, but this would occasionally dip down under ten. These inconsistent framerates ended up making the testing process quite difficult as we could not reliably view the stream in real-time.

During the testing process, it occurred to us that it would probably be simpler and more reliable to host a wireless access point on the Raspberry Pi and just connect to that like you would any other wireless network. This not only made the setup process much simpler for us and future users, but we also immediately saw an improvement in data transfer speeds, consistently reaching around forty frames-per-second. Not only were the transfer speeds now noticeably and consistently better, but it was also much easier to programmatically create a socket connection and transmit data over that to a pre-determined IP address that our DHCP server would lease out to clients.

5.8.3.2.2 WAP Hosting on Raspberry Pi

To utilize the abilities of WAP over Wi-Fi direct, it was necessary to configure the Raspberry Pi to server as a wireless hotspot so that other devices could connect to it as you would any other wireless network. The process of configuring a wireless hotspot on the Raspberry Pi was a relatively painless process and involved using two important software libraries: Hostapd and Dnsmasq.

5.8.3.2.3 Hostapd

Hostapd (Host access point daemon) is a user space software access point that is capable of turning normal network interface cards into access points and authentication servers. For this project, we wanted to convert our Raspberry Pi's network interface card into a wireless access point, so this software library was perfect for our needs. Hostapd is designed to work on Linux-based machines, and will work on the Linux-variant Raspbian Jessie, which is the operating system that the Raspberry Pi runs.

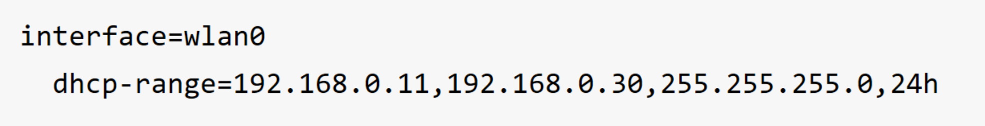
To properly configure Hostapd, a configuration file located in */etc/hostapd/hostapd.conf* was modified to contain important information for our new network. This configuration file contains information such as the appropriate interface to use, the channel, the SSID, and the desired passphrase, if the user chooses to add one. From this point, the only step left was to configure Hostapd to reference that configuration as a daemon so that the access point would become active as soon as the Raspberry Pi powers on. To do this, we just needed to modify the */etc/default/hostapd* file and set the variable *DAEMON_CONF* equal to the path of the configuration file that we just edited.

At this point, the wireless access point is now active and visible to other devices but will not accept new connections. This is because there is no DHCP server set up that will handle the process of leasing out IP addresses to clients. To resolve this, we configured Dnsmasq to act as our DHCP server.

5.8.3.2.4 Dnsmasq

Dnsmasq is a software library that provides network infrastructure for small networks, specifically DNS, DHCP, router advertisement, and network boot. For this project, we just needed a lightweight DHCP server that was simple to configure, reliable, and interacted well with Hostapd. Dnsmasq does all of these things and is often used in conjunction with Hostapd. Similar to Hostapd, setting up our DHCP server using Dnsmasq just required modifying a configuration file and a reboot of the network services on the Raspberry Pi.

The file that was edited was */etc/dnsmasq.conf*. Within this file, all that was required of us was to find the same interface that we used for the wireless access point and add the following command, as seen in Figure 65.



```
interface=wlan0
  dhcp-range=192.168.0.11,192.168.0.30,255.255.255.0,24h
```

Figure 65: Dnsmasq.conf Configuration Example

These lines that were added mean that Dnsmasq is going to supply the IP addresses between 192.168.0.11 and 192.168.0.30 for the wlan0 interface. In other words, when a client attempts to connect to the Raspberry Pi, the client will be given an available IP address within this range.

The “24h” keyword at the end of the string specifies the lease time of each IP address. In this example given, the lease time is twenty-four hours. All this means is that a specific IP address will be leased to a client for twenty-four hours. Once this lease time expires, the client will have to refresh their connection and obtain a new IP address. According to the Dnsmasq documentation, the minimum lease time that is allowed for the DHCP server is two minutes. This means that it is possible for an IP address assignment error to occur when a device tries to connect if there is already a device connected. In our project specification section, it was stated that this project is meant to only be a one-to-one connection, meaning that only one client can connect at a given time. As such, this current configuration will not cause any problems, but it is possible that the DHCP server will not be able to lease an IP address to a client because server has exhausted all the available IP addresses in its pool.

One minor pitfall that we have encountered with Dnsmasq is that when a device disconnects, the DHCP server does not always disassociate with the connected client. This occasionally causes problems when another device tries to connect to the Raspberry Pi’s wireless access point because there are no available IP addresses to lease out. The issue is still being investigated and is likely to be due to a configuration error, but the root cause remains unknown.

This issue can be fixed by restarting the Dnsmasq service or by removing the lease file that is generated by Dnsmasq when a new client connects to the Raspberry Pi. All things considered, Dnsmasq has worked incredibly well for our needs and has allowed us to effectively use the Raspberry Pi as a wireless access point, which has proven to be a much better networking option than Wi-Fi direct.

5.8.4 Development Environment

The design of this device’s software will be aided and verified through the corresponding development board, which uses a command-line interface in the default configuration, and the IAR Embedded Workbench is its preferred IDE and programming suite. The Wi-Fi functionality will be developed hand-in-hand with the mobile application, details of which can be found in Section 5.6.3.1 above.

5.8.5 Support for TCP

The dual-core, dedicated network processor design and split RAM of the device means a TCP connection to the mobile application should be easily achieved. All necessary code for TCP communication is included in the GS2101's software package, and it will be used for the sole purpose of control signals if and only if testing determines it doesn't interrupt the chip's throughput. Lacking this, we will implement an extremely simple, UDP-based acknowledgement protocol for any control packets which would cause issue if lost; "this application is powering down" serves as a reasonable bidirectional example.

6.0 Integration and Testing

The following sections discuss our testing methodology in detail and include information about our unit and integration testing strategies. In addition to those strategies, we provide many graphics demonstrating how we tested various optical and electrical components, including the LEDs and the objective lens.

6.1 Standalone Testing

Some systems, largely the software, are capable of internal verification methods. The design of these will be detailed below, turning early objectives, expectations, and concerns into a test suite.

6.1.1 Mobile Application

One of the biggest concerns that arises when creating Android mobile applications is ensuring that the app will display and function properly on all devices. As of 2015, there are over 25,000 unique android devices on the market, meaning that testing on all these different devices is a borderline impossible feat, especially given the time constraints of this project. Thankfully, all that developers really need to worry about is the resolution and the current Android API level that is supported. We ended up using Android API Version 15 (4.0.3 IceCreamSandwich), which allowed our application to run on all devices that run the Android operating system. Since there is such a wide breadth of unique Android devices, it can be difficult to ensure that the mobile application will look the same across each user's appliance. However, because we know that the API level confirms that the software will at least run on the mobile device, we can then narrow down the testing scope further by testing on the most common resolution sizes that are available on the market. The graphic shown in Figure 66 below illustrates the resolution sizes of the most commonly purchased 87 Android devices on the market. This data was compiled by collecting the data from a website, putting it into an Excel spreadsheet and then a short Python script was written that calculated the frequency of the each of the resolution sizes that were in the sheet, ordered them, and then wrote them to a new spreadsheet.

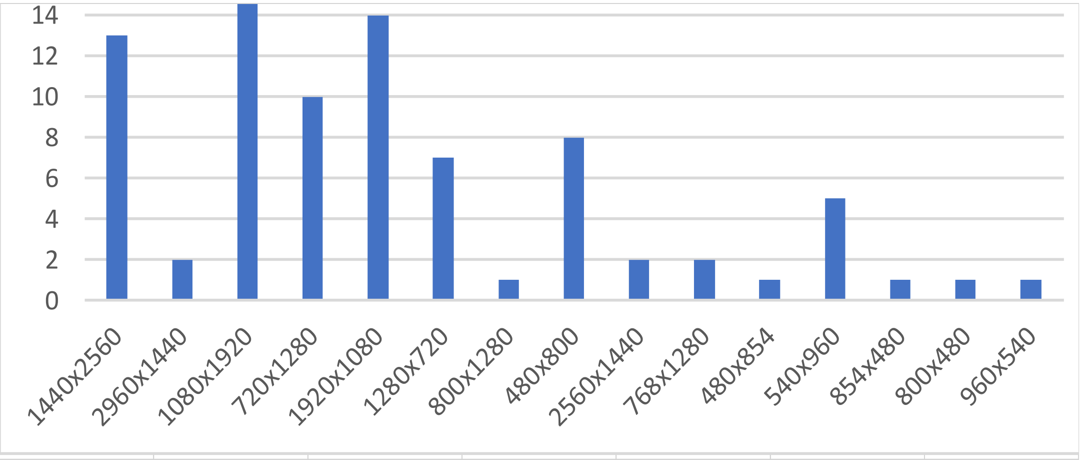


Figure 66: Popular Android Resolution Sizes
Resolution sizes from 87 most popular Android devices as of September 2018

With this information, we can target the most commonly used resolution sizes during the testing phase of development and ensure that the app performs properly on a wide array of devices.

Since we didn't have this many physical devices available at our disposal, we used the AVD (Android Virtual Device) tool within Android Studio. The Android Virtual Device tool is a hypervisor that can emulate the kernel and driver code required to run different models of Android phones. This hypervisor essentially allowed us to run a virtual copy of a specific model of an Android phone or tablet within Android Studio and conduct our testing of the mobile application. This allowed us to emulate each of these device resolution sizes and build and run the created APK on the emulated device. From there, we were able to perform unit tests on the application to ensure that everything works as intended and make sure that none of the visual features are messed up on specific devices.

It is important to note that running an emulated version is often a very slow and arduous process, so most of the primary development will be done using the developer's physical phone. This is done by simply connecting your Android mobile phone or tablet to your computer and then setting it as the primary testing environment within Android Studio. The primary benefit of this is that the application is compiled and stored on your physical device as opposed to a virtual environment. Not only is this convenient in the sense that your application is like any other application on your phone in that you can unplug your device from your computer and your application will still be on your phone, but the compile and APK installation process is significantly faster when using a physical device. Also, when using a hypervisor, the computer that is being used for development is put under a decent amount of additional stress because it must allocate some its resources to run the emulated device. This starvation of resources causes Android Studio to run slower, thus resulting in slowing compilation times, and it makes the entire development process take longer than it needs to.

However, we were of course limited by the number of physical Android devices that we have. We still needed to test the mobile application on multiple devices, so we ran custom unit tests for the software during different testing stages at the end of each coding sprint. These unit tests were then evaluated on devices matching the different resolution sizes in the chart above and any bugs or abnormalities encountered were added as an issue to the GitHub issue tracker and taken care of during the next sprint cycle.

When creating Android mobile applications, it is critical to make sure that all the views and components that are used are created in a dynamic fashion and avoid using static size constraints. We decided to make use of the various tools for creating flexible applications, such as the ConstraintLayout layout. The ConstraintLayout was used as the base layout for this mobile application because of its dynamic properties. ConstraintLayout works in a way that sizes views in a relative way, with special anchoring properties to other views that you might have. This allows the components of the application to grow and shrink with one another accordingly, regardless of the resolution of the screen. Using this layout allowed us to effectively develop an application that can work on large phones, small phones, tablets, and even some of the larger applications, including an Android Smart TV.

6.1.1.1 Unit Testing

To test key components of the mobile application, unit tests were developed to test the core functionality of the application. Listed below in Table 14 are some of the tests that were created and run against each of our builds.

Unit Tests
Application is capable of discovering Wi-Fi direct groups
Video stream appears as intended on screens of various sizes
Storing images works as intended
Image processing is uniform across all devices
User Interface isn't buggy/deformed on multiple devices
Stitching together JPEG frames works as intended

Table 14 – Unit Tests

The primary focus of having these unit tests was to ensure that basic components of the app perform well on their own. These tests were performed apart from the integration testing that was done because we wanted to ensure that each component functioned as intended and was ready for integration testing. That way, if a bug arose during the integration testing phase, pinpointing the source of the error was much simpler and we were able to quickly remedy the issue.

While many of the tests could have been written by our developer without a helper framework, Google recommends using their testing framework, Espresso, for any UI tests that you need to be completed. Espresso is extremely handy because it provides APIs for writing UI tests to simulate user interactions within a single target app. In addition to that, Espresso can do things such as detect when the main thread of your application is idle, so that it is able to run tests at the appropriate time, which greatly improves the reliability of your tests. Since the framework does this on its own, it saves the developer from having to go in and add manual sleep statements, which could cause unwanted errors in the application and could also decrease the reliability of the tests that are being run. The Espresso framework runs in conjunction with the JUnit test framework, and the AndroidJUnitTestRunner test runner, which provided us with a simplistic way to automate all the tests found in the table above. To create these tests, we used the ActivityClassRule rule to test the functionality of a given activity. This allowed us to segment our tests by activity, while still being able to automatically perform user interface and programmatic unit tests.

6.1.1.2 Embedded-Facing Components

In addition to constructing unit tests for the mobile application, there will be specific integration tests that will constructed to ensure that the various components of the mobile application work together, and that the mobile application functions well with the microcontroller used in this project. The primary things that we want to test between the mobile application and the microcontroller are the Wi-Fi direct connection, the socket connection, and the duplex data transmission. To test the Wi-Fi direct connection and the socket connection, we just need to navigate through the user interface to the network settings activity and attempt to connect to the microcontroller. If the connection is successful then there a dialog box will appear, informing the user that the connection attempt was a success. If the connection attempt fails, a similar dialog box will appear telling the user that the application failed to connect.

In addition to testing connectivity between the mobile application and the microcontroller, we also need to check the ongoing connection that the mobile application will have with the SQLite database stored on the app. To test this, we can write some test code that will attempt to connect to a database and perform create, read, update, and delete functions on the database. Testing this part of the application is critical because if the database stops working for some reason then the user will no longer be able to quickly access their files that they've taken directly from the application. As for automating integration testing, there are some options that we've considered, but since there will be one primary developer for the mobile application, we won't need to worry too much about integration testing. The only true integration testing that will need to be performed will be testing network connectivity with the microcontroller. After doing some research, we found that there are many ways to automate integration testing, and in fact, it's fairly popular to use JUnit for automating these tests. The only difference between using JUnit for unit tests versus using it for integration testing is that your actual test method will perform a test such as attempting to connect to a third-party device instead of texting a specific segment of code within the application.

6.1.1.3 Espresso Testing Framework

Since the Espresso framework works alongside with the JUnit library, the process for writing test methods works the same way as writing tests as when using JUnit. The JUnit library includes a set of JUnit rules that allow you to run JUnit style test classes on various Android devices, including those that use the Espresso testing framework. The rule that we used for this project was the ActivityClassRule. The JUnit library requires us to use a set of annotations before each method in our testing class in order to denote what each part of class represents. Below in Table 15 are some examples of annotations used in the JUnit test library and what they correspond to.

Annotation	Purpose
@BeforeClass	Runs once before any of the test methods in the class are executed.
@AfterClass	Runs once after all the tests in the class have been run
@After	Methods denoted with this annotation will be run after the method with the @Test annotation.
@Before	Methods denoted with this annotation will be run before the method with the @Test annotation.
@Test	Methods denoted with this annotation indicate the unit test and what will be run

Table 15 - Espresso Method Annotations

For each class of unit tests that are created for this project, these annotations were used to tell the JUnitTestRunner in what order to run everything. We considered writing our own series of unit tests for the application without the use of an external framework but doing so would have made the overall testing process significantly more tedious than it needed to be. With the combination of JUnitTest and Espresso, we were able to automate the entire process of testing various aspects of the application. If we were not using this framework, we would either have to go in and write our own unit test runner to handle the creation and destruction of the unit tests and then somehow handle the output of these tests in a way that allowed us to test the validity of our code.

6.1.2 Optical Component Testing

This portion of the paper will be discussing the optical components and how we tested them. It will describe the optical layout of the microscope as well as successful and unsuccessful testing and what was learned from each test. We will describe what specifically we were testing for as well as the testing method used.

6.1.2.1 Objective Lens Testing

As discussed in previous sections of this paper, our objective lenses are Deutsche Industrie Norm so they have a specified tube length. This tube length is 160 mm, and the intermediate image plane is at 150 mm. Using this knowledge and an objective lens with 4 times magnification and a numerical aperture of .1, we designed a test to see what the camera actually sees. The test consisted of a 150 mm threaded tube made of delrim, a moto e phone, and a resister. We attached the objective lens to the threaded end of the tube and then mounted it on a stand. We then used the flashlight of an iPhone to illuminate our specimen which in this case was a resistor. This set up is shown below in Figure 67.

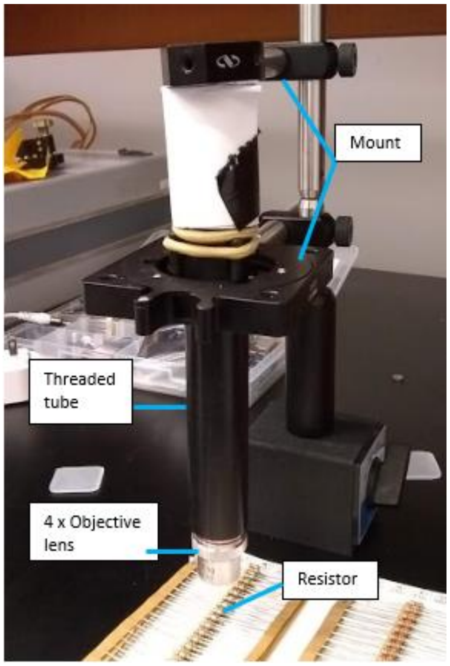


Figure 67: Optical Test Set Up

The above picture is our set up for the initial testing of our objective lens. This is to test the objective lens to camera ratio and make sure we get a clear image.

Once the set up was complete we used a Moto E android device to take pictures and see how well it could image the specimen with the objective lens in use.

6.1.2.2 Results – Objective Lens

The results were not what we were expecting. The image showed up very clearly at the 160 mm, not the 150 mm that we were expecting. The image was also too small, when viewed through the camera of the Moto E device the image was 90% tube and only 10 % image. We tested different distances between the image sensor and the objective lens to try to enlarge the image, but it was unsuccessful.

This means that there is a problem with the information we have on the objective lenses, because by all accounts this should have been a success. This problem encountered sent us to go do some more research, and what we have concluded is that either these objective lenses are actually infinity corrected at 160 mm tube length or that they require an additional lens at the 160 mm tube length.

6.1.2.3 Additional Testing and Results – Objective Lens

Following our unsuccessful testing, we decided to use the same setup but with an additional lens. This was a one-inch lens with a focal point of 3.5 mm. When this lens was used in conjunction with our objective lens, the image was magnified and easy to see. We initially placed this lens at 150 mm from the back of the objective lens, but the image that formed was unfocused. Even though it was unfocused it did improve our image to tube lens ratio.

To try and focus the image, we adjusted the lens numerous times until we had a clear, focused, and magnified image of a resistor. Once the image was all of these things we measured the distance from the end of the objective lens to the additional lens and found that it was 160 mm. From this observation we have learned to get a larger resolvable image, an additional lens must be used. A comparison of the previous resulting image and the resulting image after the lens was added is shown in Figure 68.

Having learned this, we have done some calculations using the information provided by the objective lens. The equations used are shown below and a table following that shows the focal lengths of the lenses needed.

$$r = \frac{.61\lambda}{2 * NA}; f_{TL} = M * f_{Obj.Lens}; M = \frac{Pixel\ Size}{r}$$

Where *r* is the resolution, *f* is the focal length, and *M* is the magnification. The resolution was calculated for the 4 x objective lens with numerical aperture (NA) of .1, the 10 x objective lens with numerical aperture of .25, and the 40 x objective lens with numerical aperture .65. Following the resolution of each of these, the magnification needed was calculated for each objective lens so that the focal length of the necessary lens could be calculated. The results of this are shown in the Table 16 below.

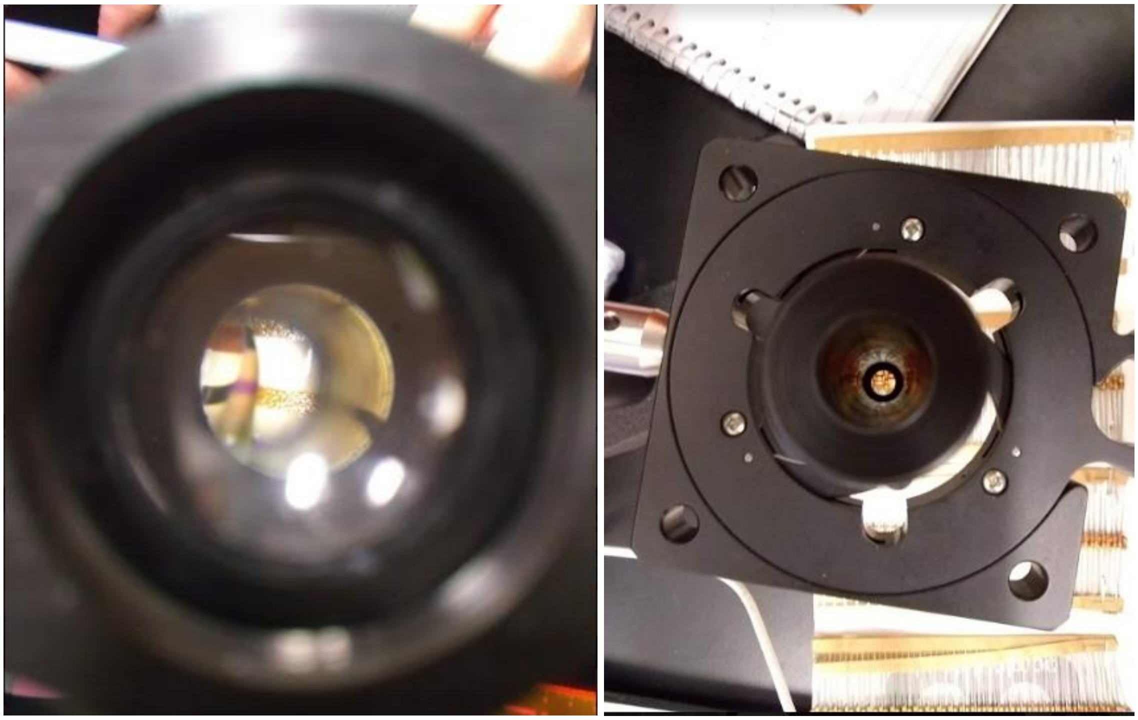


Figure 68: Optical Test Results

A side by side comparison of before an additional lens was added and after the additional lens was added. The one on the right has only an objective lens and a camera, the one of the left has an additional lens, an objective lens and a camera. This shows that an additional lens is necessary for our system.

Objective Lens	Resolution (nm)	Magnification	Needed Focal Length (mm)
4x	2592	1.57	62.8
10x	1037	2.89	46.24
40x	400	7.5	30

Table 16 - Objective Lens Characteristics

Table showing results of calculations used to identify the focal lengths of the lenses needed.

Using the results of these tests and the calculations above, we have settled on a one-inch diameter lens that has a focal point of 62.9 mm.

This test has also shown that our Deutsche Industrie Norm objective lens are finite conjugate lenses, which means that they do in fact need the additional lens. Because the lenses are finite conjugate, there is a possibility that we will actually need more that one additional lens.

We may need to have an additional lens for each objective lens, and if that is the case our body design will be altered. Due to the nature of the electronics within the body of our microscope it is not feasible to have interchangeable lenses within the tube, so we may have to omit having multiple levels of magnification. However, we are going to test the new lens when it arrives with all three of our lenses and because the focal length required is a minimum, and not exact distance we chose the largest focal length required and ordered a lens based on that.

Another interesting result of the second experiment, is that the focal length of the random additional lens is only 3.5 mm. This is not at all what calculations have shown, so this is greatly perplexing. Unfortunately, we could not obtain a lens diagram of the inside of the objective lens, and due to this we cannot do any computer ray tracing to find out why this result was produced.

6.1.2.4 Light Source Testing

We needed to test the light source to be sure that we did not saturate the image on the camera. The other reason for light source testing was to find the position that would be best for the light source. The last thing we were testing for with the light source testing was to see if we could see in the 850 nm range.

To set up the test, we used the same procedure outlined in the objective lens testing section, with only minor modifications. One modification was to move the added lens, when we moved the added lens, we actually made a conjugate plane. This meant that we achieved the proper ratio of specimen to background in the image. This was interesting to note because the focal length of the lens was 3.5 mm, and we had it at almost double. Not only that, but when the lens was removed from the system, the image was no longer in focus with just the objective lens.

We used a piece of Styrofoam and cut out the center so that the tube could pass through it. We then soldered black wires to the cathodes of each LED and a red wire to the anode of each LED. The purpose of these wires was to make sure they could be plugged into the breadboard where the power supply was. Following that, we made eight holes in the Styrofoam, four for the white LEDs and four for the 850 nm infrared LEDs. We then pointed the LED’s straight down at our table, as we could not aim them directly at the sample.

6.1.2.5 LED Results

We found that the white LED’s were perfect, one LED was enough to light a sample, which in this test was a piece of notebook paper. The resulting image was very clear, and a great amount of detail could easily be seen. The result of this is shown below in Figure 69.

We were pleased with the results given by the white LEDs; however. we did notice that the camera was saturated very easily, so we adjusted our original design that included eight of each type of LED to include only four white LEDs. These LEDs will not be dimmable, but instead we will turn on either zero, two, or four of the LEDs at a time.

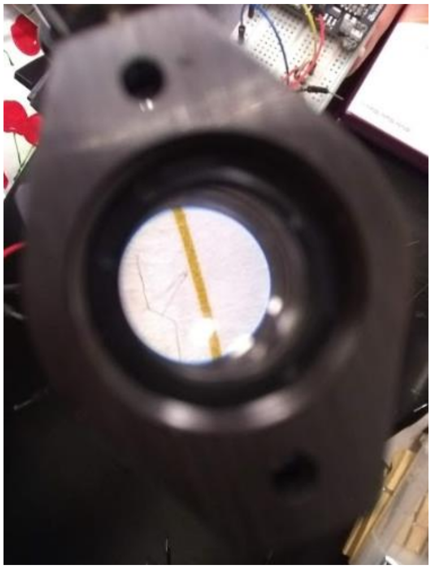


Figure 69: LED Test Results
Image captured using just one LED as a light source.

We were pleased with the results given by the white LEDs, however we did notice that the camera was saturated very easily. So we adjusted our original design that included eight of each type of LED to include only four white LEDs. These LEDs will not be dimmable, but instead will turn on one, two, or four LEDs at a time.

6.1.2.6 Additional Testing and Results - LED

Following the success of the white LED's we then started testing the 850 nm LEDs. The first part of this was to find a camera that detected infrared in the first place, as our eyes are unable to. We would have liked to use our sensor, however we only just received our sensor and it is not yet hooked up and functional. This meant that we needed a suitable replacement. After testing with an I phone and a Moto E, we discovered that the Moto E is the most suitable for this. The IPhone has a filter in place that blocks any infrared to reduce glare in the camera. Using the Moto E, we first checked to make sure the 850 nm LED's were on and hooked up properly in the circuit. We started with only one infrared LED. Once we had checked that everything was on and working, we attempted to get an image using the Moto E, but were unsuccessful. The next step was to get rid of all the ambient light, so we turned off the lights to the lab. Again we tried to see anything through the microscope using the Moto E, but we were still unsuccessful. The last thing we tried, was to hook up all four infrared LEDs at one time and turn off all ambient light. We were still faced with no viable results.

We then analyzed the results, we knew that the Moto E did sense infrared light, which meant that was not the problem. We had eliminated the ambient light, so that was not the problem either. We think that the problem is one of two things, either the sample does not reflect infrared light or the infrared light was too dim. We are currently in the process of researching the first idea, but in the meantime we have ordered larger, brighter infrared LED's to test that idea.

6.1.2.7 Optical Testing – Camera

For the optical testing that was performed we needed a camera that was able to view in the infrared as well as visible. Due to unforeseen obstacles, we are unable to use the sensor we purchased so instead we actually used an android phone in its place. This is sufficient to get an image, however the image itself will be misleading. The camera within the android device has a lens on the front of it, and our eyes also have a lens in them. This means that without our sensor we will not have the correct results, as our sensor is incapable of seeing what our eye or our android device sees due to the lack of lens on it. Due to the circumstances we will continue to test the optical components using this camera, however it is important to keep in mind that some results might not be accurate. Once our sensor is up and running we should be able to test the calculated results.

Imaging test #1

After the success of our first test that used an android camera, we now needed to test using a sensor that was comparable to what we will be using in the final design. To do this we used a three foot tall half inch steel metal rod as an optical rail. We mounted this using a magnetic clamp, and then mounted our 4 x objective lens into the tube we built. We then placed a 62.9 mm focal length lens above that and proceeded to take apart a 2.1 megapixel Nerf digital camera. We removed the casing to locate the optical components of the camera. We were able to find a small lens with an antireflective coating that simply unscrewed to remove. Once we had everything set up we tried to get an image with the 2.1 megapixel sensor from a digital camera.

The test was a failure, all the sensor was able to detect was a blur spot. There was no image whatsoever, just a giant blur. We moved the image up and down the optical rail to try to get it in focus to no avail. We thought that maybe the sensor was saturated, so we turned off all of the lights except for the source light. We still only got a blur spot. It was a complete failure and we had to go back to square one to see where the optics had failed.

The first problem with our set up, is that it was based off of the success from our previous test. The previous test was indeed successful in getting a clear and magnified image, however we forgot to take into account the optical components within the cell phone camera that we were using. The reason that we thought the objective lens was not Duetshe Industrie Norm standard was because of the tiny image that was produced in the first tests. It turns out that the optical components in the camera were the reason for this misleading result. Upon realizing the error of our first tests, we tried to correct it by simply removing the optical components in front of the sensor. This would not work, because now the image was too magnified, and could not fit on the sensor. It was only seeing a spot.

Having identified the problem we removed the extra lens from the experimental set up. We then used a screen to find the real image, so that we could place a sensor in that spot. This time, it was a success. The result is shown below in Figure 70. We were able to find the image of a double slit at 160 mm from the colored band of the objective lens. So we were able to correctly find the working distance. We placed the sensor at this distance and were able to capture an image. The image is not in high resolution due to the image being too magnified for the size of the pixels within the sensor.

We intend of solving this problem by using smaller specimens, however the test is still considered a success because we were in fact able to magnify and project an image onto a sensor which proves that our project will work. Another option is to use a smaller magnification level, so the image is not so big for the sensor.

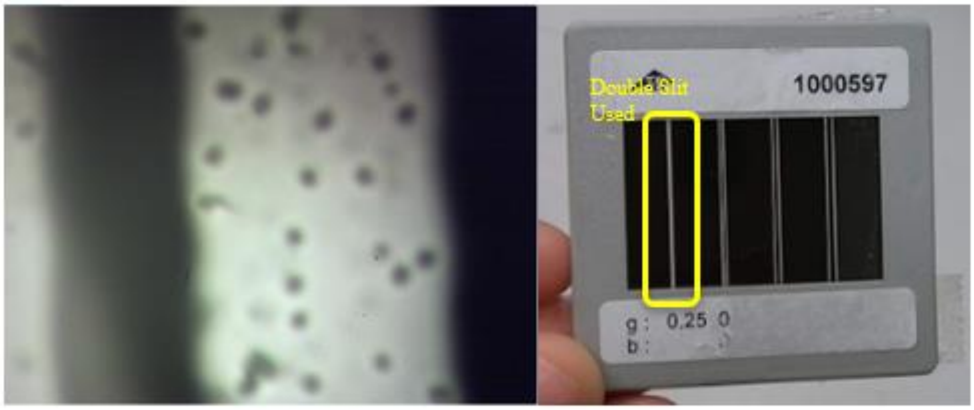


Figure 70: Imaging Results
Side by side comparison of the original object(right) and 10 times magnified(left).

6.2 Breadboard Testing

Breadboard testing is an important part of any electrical testing. Preforming breadboard testing gives researchers the ability to test their chips and designs to make sure they are properly working. If they are not, breadboard testing provides the ability to troubleshoot the design and find any incorrect connections, voltages, currents or short circuits that could damage the circuit if implemented into the final printed circuit board. Though not all components can be breadboard tested due to their chip package, certain aspects can be tested on the breadboard, like LED lighting systems and other breadboard-pin friendly packaging.

6.2.1 LED Testing

While testing for image acquisition, we also tested for the LED lighting systems. These were simple circuits, consisting of a power source, a resistor and four white LEDs in parallel. These LEDs needed to be in parallel so that each cell could receive the same amount of power, which would maintain each LED at its maximum brightness. In this testing environment, an AC-DC converter was used from a wall connection and provided an output of 5 volts. With this amount of voltage, a 1 kΩ resistor was placed before the LEDs were connected in parallel. This resistor lowered the voltage to a safe amount. We plan to have a voltage regulator that will output about 3V to the printed circuit board. With this low of a voltage, the LEDs would not require a resistor to protect from excess voltage to the components; this was only used for these testing purposes. Examples of the white and infrared LEDs are shown in Figures 71 and 72 below. Figure 71 shows the white light circuit containing eight LEDs, and Figure 72 shows the infrared light circuit containing four LEDs.

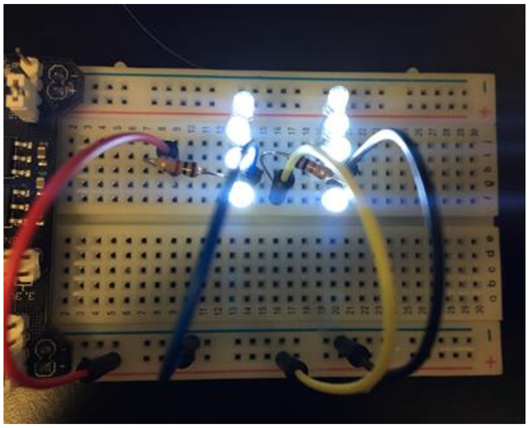


Figure 71 - White Light LED Breadboard Testing

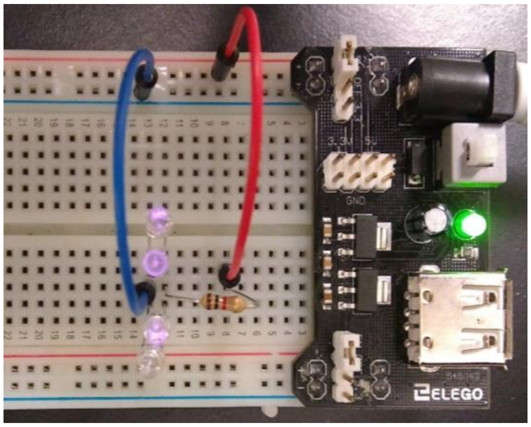


Figure 72 - IR Light LED Breadboard Testing

When we began testing these circuits, we tested the white LEDs in a group of four and a group of eight. We used these combinations because they would both be even spacing for the white and infrared LEDs around the lens of the microscope. When looking with the eye, the set of eight was brighter and seemed to be the proper amount in order to light the system, especially with the use of a diffusion filter. However, when testing the image acquisition, ambient light in the room was enough to illuminate the magnified sample, as shown in the imaging portion of testing. After seeing these results, we decided to test these LEDs again with a more stable tube structure and with another type of image sensor to see if the images from that will be saturated. The human eye has an easier time differentiating light than sensors do. Though our image sensor for our final project cannot be tested during this portion of the design project, another sensor will be used in its place for simple image testing. LEDs will be properly tested in this section as well to be sure the proper amount of illumination for the sensor without over saturating the image.

6.2.2 Breadboard Testing Plan

It is essential for this project to have a test plan due to its limited capability for breadboard testing at this time. Due to the individual layout and packaging of some of our components, most of them could not be tested during this semester. However, there are plans for testing in the near future. Because two of our chips are a BGA format of pin layout, there will need to be individual chips created in order for each chip to be tested. Individual boards will be created and a local electronics company will solder each chip to their respective boards. From here, we will test power application to be sure the schematics are correct.

The microcontroller and Wi-Fi chip that were selected for this project are available on development boards. These development boards have been ordered and are in the process of being shipped to us. This will enable us to program the chips while the power supplies and connections are being properly tested. This process will be beneficial to us because by the time the power supply is properly designed and PCB designs are tested and completed, the programming will be ready for the system. When the individual breadboard tests are done for this system, and all connections and power supply are verified to be correct, the PCB design can be completed and submitted at the beginning of Senior Design 2.

6.3 Integration Testing

Bringing the components together will be a task all its own, as there are four fully-independent, intricate electronic components. This task will be supported by the development kits, and the first priority of the development stage will be establishing and testing the connections between all components, slightly de-emphasizing the image sensor, as its operations are the least complex by an extreme margin. All testing will be done between and written at the evaluation kit level, as the correspondence of evaluation kit to component is fully one to one following well-documented the steps of device programming and PCB interconnection.

6.3.1 Microcontroller and Image Sensor Interoperation

Using the provided HAL libraries of the NUCLEO board, images will be captured and saved to a testing computer’s memory to validate the sensor’s functionality.

After the sensor is known to be operational, developed microcontroller code can be coherently tested against the image sensor. DCMI operations for selecting only a specific portion of the image output, YCbCr conversion algorithms and their runtimes, and JPEG Codec conversion will be tested and validated, in that order, verifying each is fully functional against expected software output.

For the testing of uncompressed image capture, full data coherency tests can be performed at this level. An image is read in, and then the image is read out by USB. If it is complete, both pixel by pixel and in terms of size, then the input can be known to work.

6.3.2 Microcontroller and Wi-Fi Chip Communications

As this will be performed through communication at the development board to development board level, the planned testing is straightforward. For each case of microcontroller to Wi-Fi module transfer, give the microcontroller a set of data and verify it is correctly received. For each case of Wi-Fi to microcontroller operation, give the Wi-Fi module a set of data and verify it is correctly received.

The most significant aspect of this testing pair is a throughput test on the SDIO 4-bit interface; this will identify whether or not there is a bottleneck between them, its severity if it exists, and what could be done to remediate it.

6.3.3 Mobile and Embedded Communications

The Wi-Fi board has an included, full-fledged command line interface to send and display bidirectional communication and connecting handshakes, which will help develop procedures on both the mobile and embedded side.

The Wi-Fi throughput will also be tested at this level under isolated, household, business, and university levels of network load, and protocols will be developed to achieve high-quality user experiences under each load level.

6.3.3.1 Accessing Data Stream

To display the video on the mobile application, we needed to establish a one-to-one ad hoc connection between the mobile phone and the microprocessor. The mobile application is be able to scan for available network connections and the user can select the appropriate network to connect to. Once a connection has been established between the mobile phone and the microprocessor, data will be able to be transferred across that connection, including video. We were able to successfully implement streaming from the microprocessor to the mobile application over a socket, which allowed us to transfer the frames to the mobile device without having an active connection to the internet.

For Java-based applications, such as this mobile app, dealing with sockets is a surprisingly simple task and just involves using the Java library *java.net.Socket*. This allowed us to a establish a connection with our server hosted on the microcontroller by supplying the assigned IP address and the port number associated with the socket to the object's constructor.

Currently, the data that will be received from the microcontroller are segments of a JPEG image that were captured from the image sensor. Due to memory constraints on the microcontroller side, the JPEG images sent over the socket will not be able to contain the entire image. As such, when we receive these image segments, and there needed to be some way for the mobile application to process these image segments and stitch them together into one frame in the background. In addition to doing this, we needed to make sure that we adhered to the delay time requirement that was imposed by our design requirements, meaning that this process needs to be nearly instantaneous.

We ended up using methods contained within the *java.io.File* library to read bytes from the given JPEG and store them in a byte array. As we receive the JPEG files, we needed to add them to an array buffer and then convert that buffer back to a JPEG image once all the segments have been received for the given image. To do this, we needed a way to either identify the total size of the image or determine when we needed to convert the byte array to a JPEG and display the image to the user. The way we decided to implement this was to attach the total size of the JPEG image to the data packet that gets sent from the microcontroller to the mobile application. This solution works and is preferable because it allows the mobile application to determine the size of the byte array as soon as the first segment of the image is received. This is beneficial for multiple reasons.

For starters, if we don't know the total number of bytes in the image then we'll either need to define an array of a static size that is larger enough to hold any reasonably sized image that we could receive from the microcontroller. This static size is about ten megabytes, which while being reasonably sized, it still introduces a lot of unnecessary overhead as we will not be processing images anywhere near this size.

For example, the maximum heap size for any application on most devices running Android 2.3 or later is going to be around 24 megabytes or higher. This number varies greatly from device to device and is entirely dependent on the hardware capabilities of the device. As an example, the maximum heap size on a Galaxy S3 is 64 megabytes, whereas the Nexus 5 has a maximum heap size of 192 megabytes. As such, it's very important that we consider the worst-case scenario of around 24 megabytes. If we always have this ten-megabyte array allocated, our remaining heap space is effectively halved, which is dangerous territory to be in.

It was very important for us to closely monitor the heap usage of the application during the development and testing stages of this project, because if we got too close to this maximum heap size then an OutOfMemory exception will be thrown and the application will terminate. Ideally, we want to avoid having the mobile application unexpectedly close at all costs. As such, we decided that it would be better for us to know the total size of the image as soon as each of its segments are received.

An alternative solution to having large a statically defined array was to read in each image segment into its on individual byte array and then merge all those arrays together once all of the segments have been received. While this would have saved us from always having to allocate ten megabytes of heap space for our byte array, it introduced the performance overhead of having to read these bytes into an array twice. The first time will be the initial byte arrays that store each of the byte segments of the image, and then the second time will be adding all these values into one large array whose size is the sum of the sizes of the individual arrays. This added performance overhead will likely be negligible in the end, but since we have a strict delay requirement, we would like to minimize the performance overhead from the application side and instead have a larger buffer to allow for any potential slowdowns during the data transfer from the microcontroller to the mobile application.

One final issue that needed to be considered when dealing with receiving these JPEG images in segments is knowing where the end of the image is. It isn’t guaranteed that each image will arrive in segments of three, four, or some other arbitrary number. Instead, this number will vary depending on the size of the image, the pixel intensities included in the image, and various other factors. To handle this, we needed a way to indicate the end of a file. To solve this, the Python script that we wrote would simply send over the total size of the image first, allowing the mobile application to parse that size and determine how many bytes of data were being sent afterwards.

6.4 Second Semester Optical Testing

This section will cover all the optical testing that resulted in design changes over the course of the second semester. We will discuss in detail overall imaging testing as a whole and not singular components.

From last semester, we learned that our optical system needed to move in order to easily focus, that design is discussed at length in the body design portion. We also learned that we needed a base light from thin samples, so we implemented one of those to be able to get images if things such as an onion cell. We also had to test the image sensor to make sure we were in fact at the intermediate place and the correct focal length to get a resolved image. The last thing we tested was to see if an aperture would increase contrast.

We tested the above lighting source with the 4 and 10 x magnification objective lenses. We found that the above light was not suitable for the 10 x objective lens, because the lens was too long and created a shadow directly where the sample was. The 4 x objective lens worked very well for thick samples. The results are shown in Figure 73 below.

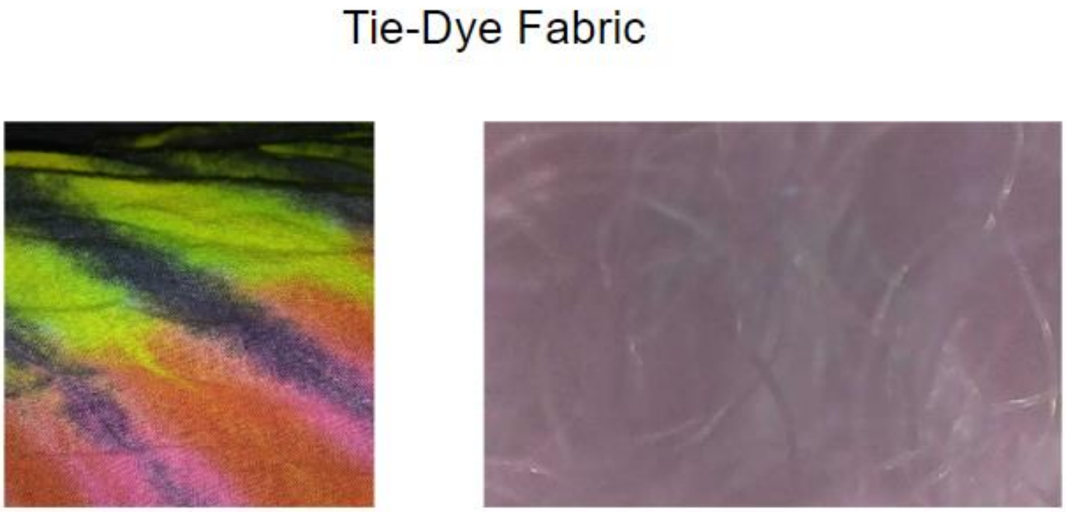


Figure 73: Tye-Dye Fabric- under a 4x objective using the top light source.

Following these results, we started testing with a base light. Our first iteration of the base light was a single white LED phone a phone camera, with a piece of white computer paper over it. The results were quite promising so it was decided to design and build a base light using the design described in the PCB and lighting source sections. This base light produced very good images as shown below in Figure 74.

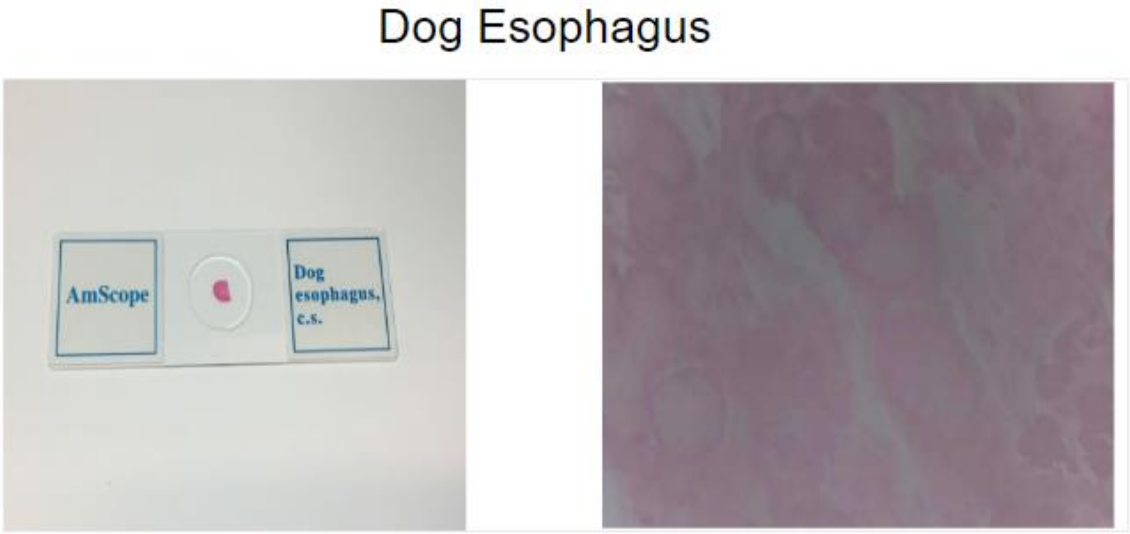


Figure 74: Dog Esophagus- using 10 x objective lens and base light

This was a great result, however we notice the contrast was not what we wanted in to be. At first we tried to fix this with software, but there was only much that could be done digitally. The problem was in the optics. Our sensor was getting saturated. There was enough saturation to lose detail, but not so much that we could not still see an image. This prompted us to introduce the aforementioned aperture. This introduction greatly increased the contrast and was part of the final design. The results of the aperture testing are shown below in figure 75.

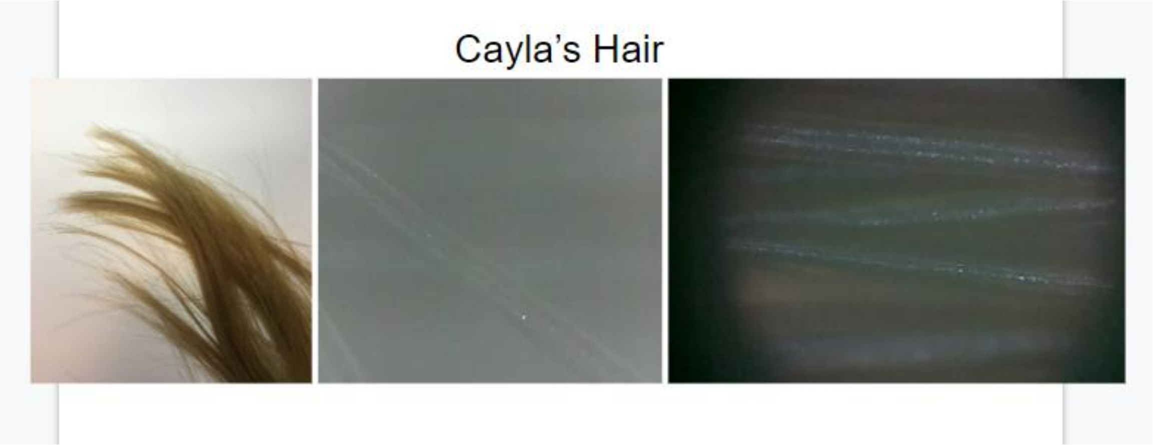


Figure 75: Cayla’s Hair Aperture Testing- Shows the increase in both resolution and contrast.

7.0 Administrative

This section of the paper contains the budget and management portions of the project. It will discuss the plans for the estimated prices and payment for the project. In the Milestones section, we will be discussing the overall schedule for this project and when major milestones will be completed. These milestones are presented for both the first and second semesters that the project will be worked on.

7.1 Estimated Budget & Financing

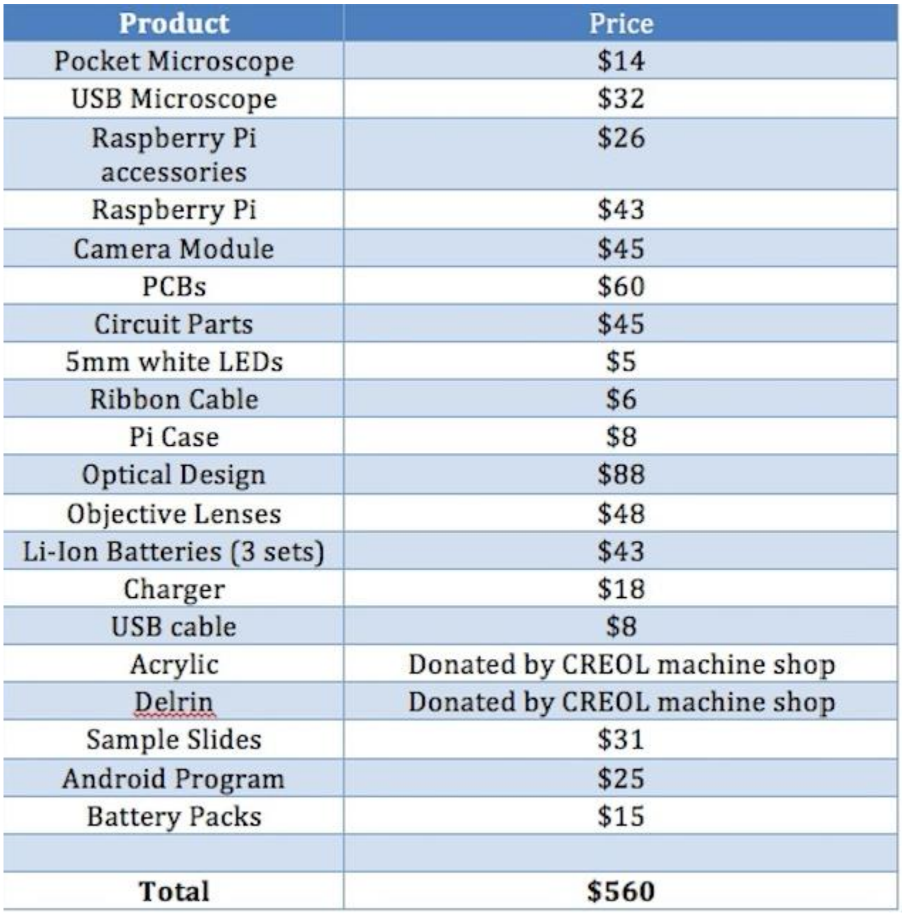
Table 17 below shows the estimated cost overall for this project. Since this is just an estimate at the beginning of the project process, the true values will be calculated and added above in section 3 when they are obtained.

Lights	$5
Microprocessor	$30
Camera	$50
Lenses	$550
Filters	$315
Battery	$30
PCB	$20
Enclosure	$20
Android Developer fee	$25
Microscope stand	$50
Estimated Total Budget	$1095

Table 17 – Estimated Project Budget and Financing

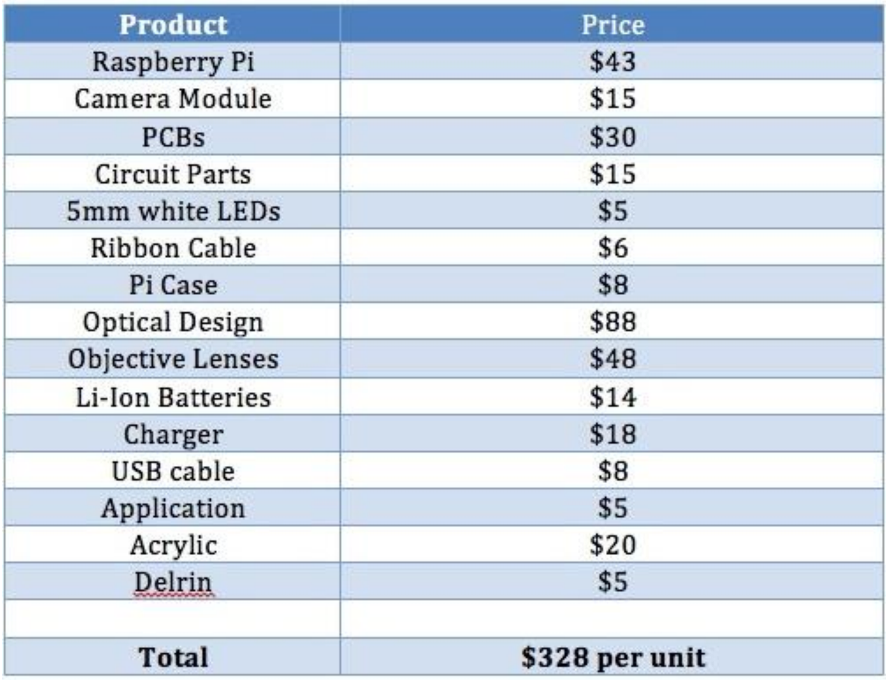
Given this total budget, it is also important to note that this project will be self- financed and therefore paid for by each of this team's members.

When calculating the budget and production costs for the alternate project design, we came to a total research cost of $560 and a total production cost for each unit at $328. These costs included redundant components that were purchased. These also included any products that we researched and observed for additional ideas for our own designs. Compared to the original price this was almost cut in half, which was good news for the members of this team. This was all funded by the members of this group, and we are very proud to have made this product. The layout of costs are shown in Tables 18 and 19 shown below.



Product	Price
Pocket Microscope	$14
USB Microscope	$32
Raspberry Pi accessories	$26
Raspberry Pi	$43
Camera Module	$45
PCBs	$60
Circuit Parts	$45
5mm white LEDs	$5
Ribbon Cable	$6
Pi Case	$8
Optical Design	$88
Objective Lenses	$48
Li-Ion Batteries (3 sets)	$43
Charger	$18
USB cable	$8
Acrylic	Donated by CREOL machine shop
Delrin	Donated by CREOL machine shop
Sample Slides	$31
Android Program	$25
Battery Packs	$15
Total	**$560**

Table 18 - Research Cost



Product	Price
Raspberry Pi	$43
Camera Module	$15
PCBs	$30
Circuit Parts	$15
5mm white LEDs	$5
Ribbon Cable	$6
Pi Case	$8
Optical Design	$88
Objective Lenses	$48
Li-Ion Batteries	$14
Charger	$18
USB cable	$8
Application	$5
Acrylic	$20
Delrin	$5
Total	**$328 per unit**

Table 19 - Production Cost

7.2 Milestones

The milestones were set based on individual tasks for each of the team members along with any due dates set for the semester. These times were also based on team members' schedules and individual responsibilities. Tables 20, 21, and 22 show our expected and to-date completion of these objectives.

Milestone	Predicted Date	Actual Date
Project Idea Finalized	9/14/2018	9/10/2018
Initial Research	9/28/2018	9/14/2018
Parts and Components Research	9/30/2018	10/14/2018
Report (60 page document)	11/2/2018	11/1/2018
Components Chosen	11/9/2018	11/13/2018
Components Ordered	11/16/2018	11/16/2018
Report (100 page document)	11/16/2018	11/16/2018
PCB design	11/20/2018	In Progress
Components Checked	On Arrival	In Progress
Basic Image Acquisition	11/23/2018	11/26/2018
Report Document Completed	12/3/2018	

Table 20 – Fall 2018 Project Milestones

Milestone	Date
Initial Testing (Lenses)	1/21/2019
3D Printing Design Done	1/21/2019
Machining Design Done	1/21/2019
Initial Testing (Microcontroller)	1/21/2019
Initial Testing (Electrical)	2/4/2019
Initial Testing (Power System)	2/18/2019
Initial Testing (App Development)	2/25/2019
Final Testing	3/15/2019
Final Prototype	4/5/2019
Project Presentation	4/19/2019

Table 21 – Spring 2019 Project Milestones

Milestone	Date
UI for Connecting to Microprocessor Complete	1/20/2019
Can Establish Connection/Share Data Between Devices	2/15/2019
Generic Image Processing Software Complete	2/30/2019
Able to Store Videos/Images to Local Storage	3/7/2019
Finalize UI	3/20/2019
Integration with Microscope	4/1/2019
Initial Testing	4/5/2019
Final Testing	4/10/2019

Table 22 – Spring 2019 Software Milestones

8.0 Project Summary and Conclusions

In summary this was a fulfilling project where each group member got to really research and learn more about their respective fields. There were many issues that plagued this project. From the optical perspective, it seems deceptively simple with a sensor and a lens, but the reality was that it was a bit more complicated than that. We had to design a focusing system, which was difficult as none of us have any mechanical background. We had to find the perfect tube length, this took a lot of measuring and testing. Then we ran into issues with contrast, so more research was done and we found a solution in the form of an aperture.

The electrical engineering was also a challenge. There were problems faced with finding a cable to transmit the proper amount of current, problems with voltage regulator designs and with soldered wiring connections. Flickering occurred because of loose wire connections, low power indicators flashed, and holes were soldered closed, which left boards unable to be corrected. New board designs were created and ordered, circuits tested and verified, and power eventually was transmitted correctly. After much trial and error and many months of research, we were able to solve all electrical problems in every way.

The computer science portion that was challenging was developing an effective way to transmit images over a socket connection while still being able to obtain a consistent transmission rate of 40 FPS. This involved many hours of testing and tweaking, but eventually a solution was reached.

In the end each challenge made us learn more and forced us to engineer a solution. We were able to come together and engineer a product we can be proud of and one that we can demonstrate in our own communities.

Appendices

Appendix A – References

“3D Printing vs CNC: Explained and Compared.” *All3DP*, 28 Apr. 2016, all3dp.com/3d-printing-vs-cnc-milling/

“A Beginner's Guide to Switching Regulators.” Dimension Engineering, www.dimensionengineering.com/info/switching-regulators

Abramowitz , Mortimer, et al. “Lenses and Geometric Optics.” *Molecular Expressions Microscopy Primer: Specialized Microscopy Techniques - Fluorescence Digital Image Gallery - Normal African Green Monkey Kidney Epithelial Cells (Vero)*, Florida State University, 13 Nov. 2015, 1:18pm, micro.magnet.fsu.edu/primer/lightandcolor/lenseshome.html

“Alkaline Batteries.” Electrical Power Generation from Hydrogen Fuels, www.mpoweruk.com/alkaline.htm

AT Commands Reference Guide. Telit, 7 Sept. 2016, www.telit.com/wp-content/uploads/2017/09/Telit_AT_Commands_Reference_Guide_r24_B.pdf

Britannica, The Editors of Encyclopaedia. “Voltage Regulator.” *Encyclopædia Britannica*, Encyclopædia Britannica, Inc., 20 July 1998, www.britannica.com/technology/voltage-regulator

“BU-203: Nickel-Based Batteries.” *Lithium-Based Batteries Information – Battery University*, batteryuniversity.com/learn/article/nickel_based_batteries

Burris, Matthew. “Learn More About Three Different Types of Voltage Regulators.” *Lifewire*, Lifewire, www.lifewire.com/types-of-voltage-regulators-818851

“Core App Quality | Android Developers.” Android Developers, Google, 25 Apr. 2018, developer.android.com/docs/quality-guidelines/core-app-quality

“Data and File Storage Overview | Android Developers.” Android Developers, Google, 10 Sept. 2018, developer.android.com/guide/topics/data/data-storage

Davidson, Michael W. “Microscope Objective Specifications.” *Nikon's MicroscopyU*, Nikon, 2018, www.microscopyu.com/microscopy-basics/microscope-objective-specifications

“Diffuser Selection Guide | Edmund Optics.” *Edmund Optics Worldwide*, Edmund Optics, 2018, www.edmundoptics.com/resources/application-notes/optics/diffuser-selection-guide/

“How to Choose Batteries | REI Expert Advice.” *REI*, REI, www.rei.com/learn/expert-advice/batteries.html

Karwin, Bill. “Antipattern: Always Depend on One’s Parent.” SQL Antipatterns - Avoiding the Pitafalls of Database Programming, edited by Jacquelyn Carter, The Pragmatic Bookshelf, 2010, pp. 5–10.

Keller, H. Ernst, et al. “Optical Aberrations Interactive Tutorials.” *Molecular Expressions Microscopy Primer: Specialized Microscopy Techniques - Fluorescence Digital Image Gallery - Normal African Green Monkey Kidney Epithelial Cells (Vero),* Florida State University, 18 Nov. 2018, 8:36am, micro.magnet.fsu.edu/primer/java/aberrations/chromatic/index.html

Keller, H. Ernst, et al. “Spherical Aberrations.” *Olympus Microscopy Resource Center | Lenses and Geometrical Optics,* Olympus America, 2012, olympus.magnet.fsu.edu/primer/java/aberrations/spherical/index.html

Mitchell, Bradley. “802.11 WiFi Standards Explained.” Lifewire, Lifewire, 3 Oct. 2018, www.lifewire.com/wireless-standards-802-11a-802-11b-g-n-and-802-11ac-816553

Mitchell Electronics. “Is There An Actual Difference Between Batteries?” *Mitchell Electronics,* 20 Feb. 2018, www.yesmec.com/is-there-an-actual-difference-between-batteries/

“Plug & Socket Types.” *World Standards*, 30 July 2018, www.worldstandards.eu/electricity/plugs-and-sockets/

Power Designer, webench.ti.com/power-designer/

“Power Supply Safety Standards, Agencies, and Marks.” *CUI.com*, www.cui.com/catalog/resource/power-supply-safety-standards-agencies-and-marks.pdf

Rottenfusser, Rudi, et al. “Education in Microscopy and Digital Imaging.” *ZEISS Microscopy Online Campus | Tungsten-Halogen Lamps*, 2018, zeiss-campus.magnet.fsu.edu/tutorials/basics/microscopeconjugateplanes/indexflash.html

Rottenfusser, Rudi, et al. “Historical Perspective of Optical Microscopy.” *ZEISS Microscopy Online Campus | Tungsten-Halogen Lamps*, 2018, zeiss-campus.magnet.fsu.edu/articles/basics/historical.html

Rottenfusser, Rudi, et al. “Introduction to Microscopy and the Concept of Magnification.” *ZEISS Microscopy Online Campus | Tungsten-Halogen Lamps,* Zeiss, 2018, zeiss-campus.magnet.fsu.edu/articles/basics/introduction.html

“Screen Sizes | Viewport Sizes and Pixel Densities for Popular Devices.” Screen Sizes | Viewport Sizes and Pixel Densities for Popular Devices, 2018, screensiz.es/

“SQLite As An Application File Format.” SQLite Home Page, SQLite, 2015, www.sqlite.org/aff_short.html

"Standard for Tests for Flammability of Plastic Materials for Parts in Devices and Appliances." *Standards Catalog*, standardscatalog.ul.com/standards/en/standard_94_6

"Standards." *Current Industry Trends | IPC*, www.ipc.org/ContentPage.aspx?pageid=Standards

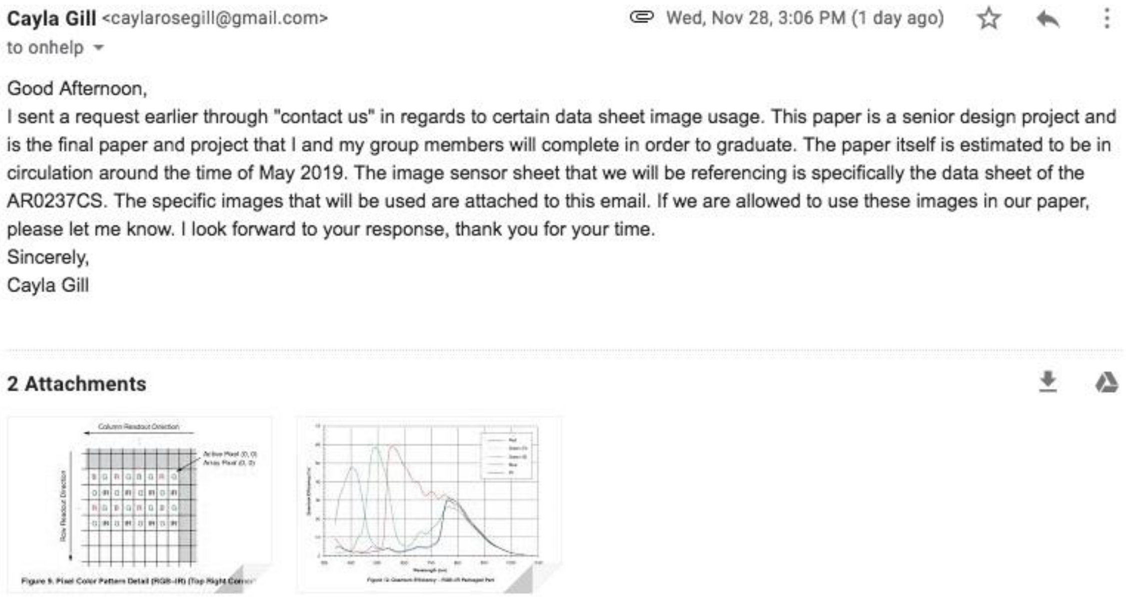
"Support Different Screen Sizes | Android Developers." Android Developers, Google, 24 Sept.

"The Growing Popularity and Applications of 3D Printing." *Money Inc*, Money Inc, 11 May 2017, moneyinc.com/growing-popularity-applications-3d-printing/

"Understanding Microscopes and Objectives | Edmund Optics." *Edmund Optics Worldwide*, Edmund Optics, 2018, www.edmundoptics.com/resources/application-notes/microscopy/understanding-microscopes-and-objectives/

"Voltage Regulators,Circuits,Types,Working Principle, Design, Applications." *Electronic Circuits and Diagrams-Electronic Projects and Design*, 9 Aug. 2018, www.circuitstoday.com/voltage-regulators

Appendix B - Copyright Permissions



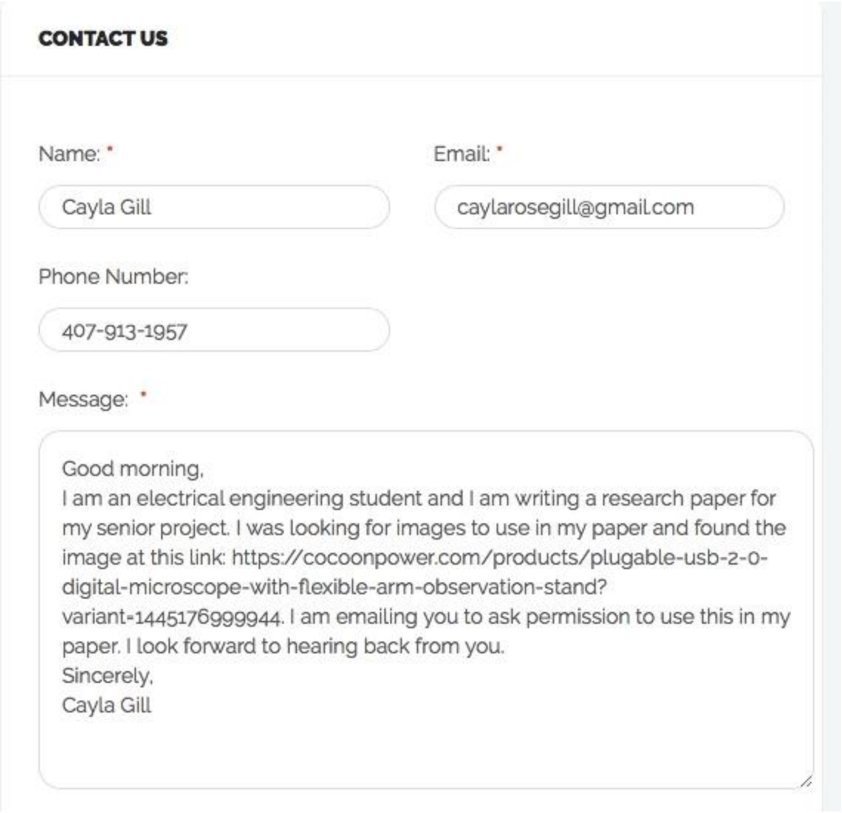
Cayla Gill <caylarosegill@gmail.com>

Wed, Nov 28, 3:06 PM (1 day ago)

to onhelp

Good Afternoon,
I sent a request earlier through "contact us" in regards to certain data sheet image usage. This paper is a senior design project and is the final paper and project that I and my group members will complete in order to graduate. The paper itself is estimated to be in circulation around the time of May 2019. The image sensor sheet that we will be referencing is specifically the data sheet of the AR0237CS. The specific images that will be used are attached to this email. If we are allowed to use these images in our paper, please let me know. I look forward to your response, thank you for your time.
Sincerely,
Cayla Gill

2 Attachments



CONTACT US

Name: *

Cayla Gill

Email: *

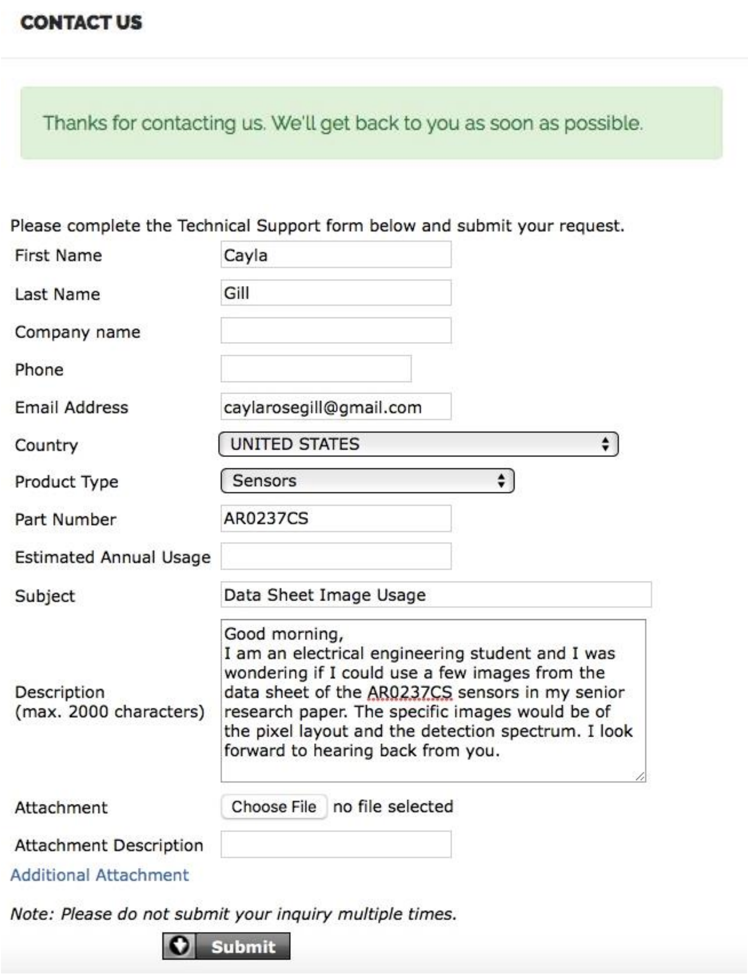
caylarosegill@gmail.com

Phone Number:

407-913-1957

Message: *

Good morning,
I am an electrical engineering student and I am writing a research paper for my senior project. I was looking for images to use in my paper and found the image at this link: https://cocoonpower.com/products/plugable-usb-2-0-digital-microscope-with-flexible-arm-observation-stand?variant=1445176999944. I am emailing you to ask permission to use this in my paper. I look forward to hearing back from you.
Sincerely,
Cayla Gill



CONTACT US

Thanks for contacting us. We'll get back to you as soon as possible.

Please complete the Technical Support form below and submit your request.

First Name	Cayla
Last Name	Gill
Company name	
Phone	
Email Address	caylarosegill@gmail.com
Country	UNITED STATES
Product Type	Sensors
Part Number	AR0237CS
Estimated Annual Usage	
Subject	Data Sheet Image Usage
Description (max. 2000 characters)	Good morning, I am an electrical engineering student and I was wondering if I could use a few images from the data sheet of the AR0237CS sensors in my senior research paper. The specific images would be of the pixel layout and the detection spectrum. I look forward to hearing back from you.
Attachment	Choose File no file selected
Attachment Description	

Additional Attachment

Note: Please do not submit your inquiry multiple times.