Portable Microscopic Fluorescence Detection

By

Jacob Reger, Mikhail Burlachenko

Graham Donaldson, Ben Adams

Prepared for:

Sanjida Yeasmin

Cheng Research Group

1. Overview

1.1. Executive Summary

The goal of our team with the Portable Microscopic Fluorescence Detection project is to create a low cost wireless portable microscopic fluorescence detector for use in medical analysis and materials research. Microscopic fluorescence detection can be used for the sensing of various chemical compounds, and the field of biochemical sensing has been particularly impacted by recent innovations in fluorescent detection. Our platform will utilize thin-film polymers on carbon quantum dots tuned to specific biomarker emissions fabricated by the Cheng research group at Oregon State University.

The project detailed here will consist of inheriting an abandoned partially-developed platform, completing a thorough system analysis on the platform, and then redesigning the system. This redesign will include an optimization for unit cost while ensuring satisfactory detection accuracy as well as a redesign of the wireless communication method and user facing mobile application. This process will begin with a comprehensive analysis on the current state of the project. Following this, embedded code will be refactored, a power supply will be constructed, and the communication method will be ported from WiFi to Bluetooth. Once that is complete, an enclosure will be constructed and a mobile application will be developed which will interface with the platform.

1.2. Team Communication Protocols and Standards

Our team's current primary method of direct communication is a shared Discord server. We use a Trello board to keep track of the project's progress and immediate interim goals. A shared cloud file on Box holds the inherited project documentation, and a shared cloud file on Google Drive holds ongoing collaborative documents.

Individual submissions to collaborative group work are expected to be done before set pre-deadlines so that the entire team has time to review and make any changes before a final submission. If a teammate expects to miss a deadline, they are expected to notify other teammates at least 24 hours before the deadline. Before starting the work on any assignment, requirements will be set and each team member will be assigned tasks to be completed. When assigning tasks, the team will discuss expected task goals and determine ways to verify whether the task has been completed sufficiently. A concise table of standards can be seen above.

Protocol	Assessment Parameters
Participate in weekly meeting to deliver and discuss progress	Group members must attend weekly Discord meetings.
Absences	A person who missed a meeting is responsible for catching up on what the team has done via reading over the meeting notes and any relevant documentation.
Timesheet	Every group member is responsible to fill out their tab in Timesheet weekly
Weekly presentations	Group members are responsible to complete and record their assigned part in a presentation
Weekly tasks	Group members are responsible to complete their assigned weekly tasks. The status of these tasks is tracked in Trello board
Work completed on time	Every group member is responsible to complete their work on time. In case they can not complete it by the deadline, they need to notify the team in 24 or more hours.

Table 1: Communication Protocols

Team Member	Role	Email	Phone
Ben Adams	Hardware	adamsb2@oregonstate.edu	(503)-798-1081
Jacob Reger	Software	regerj@oregonstate.edu	(541)-556-4816
Graham Donaldson	Software	donaldgr@oregonstate.edu	(253)-444-7267
Mikhail Burlachenko	Hardware	burlachm@oregonstate.edu	(541)-286-0609

Table 2: Contact Information

1.3. Gap Analysis

This project is the instantiation of an existing laboratory-exclusive technology onto a low-cost, mobile platform inherited from Oregon State University's Cheng Research Group. Fluorescence microscopy is a well established technology used to identify the presence of specific compounds in an organic or inorganic sample [2]. This can include identifying certain molecules within a cell as well as identifying certain cells within tissues. Current fluorescence microscopy technology is expensive, with models ranging from $2,000- $20,000 USD and requiring main power and the permanent lab installation of heavy equipment [3]. This makes the use of these current fluorescence microscopes in the field and in developing areas difficult or impossible due to lack of electrical infrastructure and prohibitive unit cost [4]. An affordable mobile platform would bring this technology to developing countries where it is not readily available.

A mobile platform would also be more convenient for field testing because it doesn't require sending the samples to a lab and awaiting results which can sometimes take multiple days or longer. Instead, for example, a performance enhancing drug test is often usually collected and tested in a lab. This is not an issue for high budget operations, but for lower budgets this can be difficult and inconvenient to do. A mobile platform would enable technicians to test on site with very little delay. Dr. Cheng's research group should be considered a primary internal stakeholder. External stakeholders include optics researchers, healthcare providers, and biology researchers without access to advanced diagnostic laboratory equipment.

1.4. Proposed Timeline

	Week 1	Week 2	Week 3	Week 4	Week 5
Presentation Video				Presentation Video ("E")	Presentation Video (Thur)
Project Partner Updates				Project Partner Update ("E")	
Project Documentation				Draft Project Document Section 1 ("E")	Draft Project Document Section 2 (Fri)
Research					
Prototyping / Testing					Test previous "inherited" project, determine downsides/what should be improved ("E")
Prototyping / Testing, cont.					
Evaluation					Review Proj Doc + inherited project: Redesign or not? (E)
Other		Meet up and decide meeting times		Exec Summary ("B")	Requirements ("J")
Other Cont.				Gap Analysis ("J")	Design Impact Statement ("B")
Other Cont.				Team Com. Protocols/Standards ("M")	Risks ("M")
Other Cont.				Timeline ("G")	References and File Links ("E)
Other Cont.					Revision Table ("E")

Table 3: Fall Term Timeline 1/2

	Week 6	Week 7	Week 8	Week 9	Week 10
Presentation Video	Presentation Video ("E")	Presentation Video ("E")	Presentation Video ("E")		Presentation Video ("E")
Project Partner Updates		Project Partner Update ("E")			Project Partner Update ("E")
Project Documentation	Peer Review Draft Project Document: Section 1 and 2 ("E")	Project Document: Section 1 and 2 ("E")	Draft Project Documentation Sections 3 ("E")	Peer Review Draft Project Document: Sections 1-3 ("E")	Project Document: Sections 1-3 ("E")
Research	Lenses to add to camera to provide a "better" image ("G")	Bluetooth Image/Data transmission Research ("B")	Bluetooth Image/Data transmission Research ("G")		
Prototyping / Testing	Get working prototype to take images, and turn on/off provided LEDs ("J")	Prototype takes pictures on press of button ("J")	Prototype sends some data through bluetooth (to computer or phone, whichever is easier) ("J")	Prototype sends tweaked image/4 images from week 8 to some device (PC or mobile, whichever easier) ("E")	
Prototyping / Testing, cont.		Prototype takes 4 pictures on press of button: 1 w/ no LED's, 1 w/ LED A, 1 LED w/ B, 1 LED w/ C ("J"/"B")			
Other			Block Diagram ("J")		
Other Cont.			Block Descriptions ("B")		
Other Cont.			Interface Definitions ("E")		
Other Cont.			References and File Links ("E")		
Other Cont.			Revision Table ("E")		

Table 4: Fall Term Timeline 2/2

Beyond the overall timeline, we have broken our development up into a three phase model. This model can be seen below. It contains three distinct development gates. This means that development on the next phase will only really begin once the previous phase has been completed and the associated project requirements have been met. Our first phase, the primary functionality, is meeting the project requirements related to the direct analysis of samples. This means that by completion of phase one, the project should be capable of analyzing a sample.

Phase two, secondary functionality, is the hardware and software requirements that are related to usability and reliability. These requirements are not critical to the core functionality, but are still firm requirements for the project to be considered a success. Finally, there is phase three, or auxiliary functionality. These are our stretch goals, or development goals which are not required for the project to be considered successful and will only be pursued in the interest of creating a final platform which is as professional as it can be given the time and resources we have available.

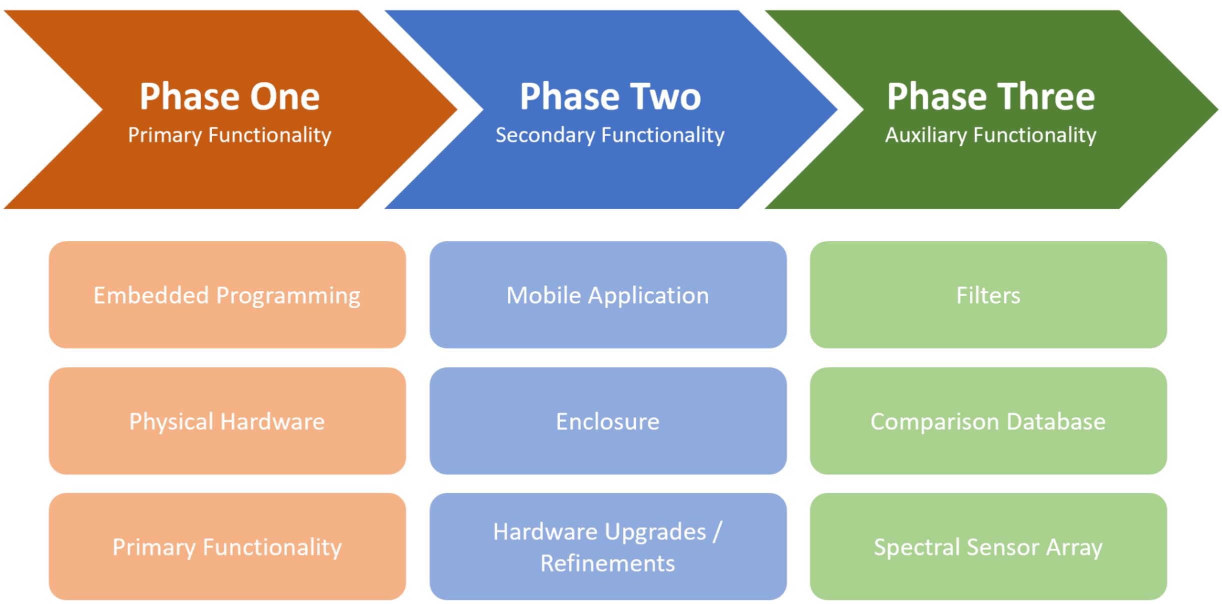


Figure 1: Development Workflow Breakdown

1.5. References and File Links

References

[1] Young-Ho Shin *et al* "Review—Recent Progress in Portable Fluorescence Sensors" *J. Electrochem. Soc.* Vol. 168 No.1 Jan. 2021. Doi: 10.1149/1945-7111/abd494 [Online]. Available: https://iopscience.iop.org/article/10.1149/1945-7111/abd494

[2] ONI "Fluorescence Microscopy". ONI.com. https://oni.bio/nanoimager/super-resolution-microscopy/fluorescence-microscopy-overview/ (Accessed Oct. 20, 2021).

[3] New York Microscope Company, "Fluorescence Microscopes". MicroscopeInternational.com https://microscopeinternational.com/fluorescence-microscopes/ (Accessed Dec. 1, 2021)

[4] United Nations Conference on Trade and Development. "People in Least Dev. Countries Lack Electricity". UNCTAD. https://unctad.org/topic/least-developed-countries/chart-july-2021 (Accessed Dec. 1, 2021)

File Links

[1] Timeline Link
https://docs.google.com/spreadsheets/d/1ouE_E5FX80h85E5u2vF_gLWPRAbcxqr9WxoZP73m2go/edit#gid=0

1.6. Revision Table

10/20/2021	Jacob: Initial Document Creation
10/20/2021	Adam: Executive Summary Added
10/21/2021	Jacob: Gap Analysis Added
10/22/2021	Michael: Team Communication Standards Added
10/22/2021	Graham: Timeline Added
10/22/2021	Jacob: References and Revision Table Added,

	Document Formatting Done
11/3/2021	Graham: Timeline changed to reflect individual roles for some “jobs” + added timeline link
11/6/2021	Ben: Revised/ reworded Sections 1.1, 1.2, and 1.3
11/12/2021	Michael: Team protocols added. Revised section 1.2
11/12/2021	Jacob: Included phase breakdown in timeline
12/1/2021	Jacob: Reformatted text, font, and spacing
12/2/2021	Jacob: Revised all of Section 1 based on feedback from instructors on draft submissions
4/14/2022	Ben: Reworded Project and document name

2. Requirements, Impacts, and Risks

2.1. Requirements

Here a total of eight engineering requirements and their verification methods will be detailed.

Requirement 1 - Portable

Project Partner Requirement:

“The portable device shall be able to take a microscopic image with excitation lights. The device needs excitation lights of 3 different documented peak wavelengths. These lights must have peak color spectrums that are less than 50 nm wide measured at Full Width Half Maximum (FWHM).”

Engineering Requirement 1:

The portable device sub-system needs a secure enclosure, with an opening to house a sample slide. Internal components (list is provided below) shall be secured to the enclosure and shall comply with the following verification method.

Verification Method:

Step 1: Visually inspect the enclosure for loose/unsecured components.

Step 2: Note the positions of specified enclosure components (see list below).

Step 3: Rotate the enclosure 180° to be upside down, then another 180° to return to its original position

Step 4: Check all sub-components for movement. This verification will fail if any sub-components move more than 2mm.

List:

1. Power supply PCB
2. Debugging LEDs PCB
3. Raspberry Pi Zero
4. Camera
5. Battery

Requirement 2 - Excitation Bandwidth

Project Partner Requirement 2:

The portable device shall be able to take a microscopic image with excitation lights. The device needs excitation lights of 3 different documented peak wavelengths. These lights must have peak color spectrums that are less than 50 nm wide measured at Full Width Half Maximum (FWHM).

Engineering Requirement 2:

The system shall use at least three LED excitation sources with a light spectrum of less than 50 nm as measured at full width half maximum (FWHM). Note: this is a project constraint.

Verification Method:

Step 1: An excitation LED will be powered on by the portable subsystem

Step 2: The subsystem will be placed into the FluoroMax-4 spectrofluorometer in the Cheng Research Lab located in Room 302 of Oregon State University's Dearborn Hall.

Step 3: The spectrofluorometer will measure the emission spectrum of the LED

Step 4: The data gathered by the spectrofluorometer shall be analyzed to determine the LED's FWHM in nanometers.

Step 5: Steps 1-4 will be repeated for each LED excitation source.

Requirement 3 – Excitation and Camera Control

Project Partner Requirement 3:

The device will work with an Android Application via a Bluetooth interface.

Engineering Requirement 3:

The device will change camera settings and excitation LEDs from an Android device via a bluetooth app interface.

Verification Method:

Step 1: The Android based mobile application subsystem will be demonstrated to wirelessly connect with the portable subsystem.
Step 2: A specific excitation parameter will be selected
Step 3: A user will demonstrate initiating the image capture by only using the mobile application subsystem.
Step 4: Steps 2-3 will be repeated using different excitation parameters.
Step 5: Image data displayed on the mobile application subsystem will be demonstrated to be different for each excitation parameter captured.

Requirement 4 – Wireless Communication Resilience

Project Partner Requirement 4:

The device will work with an Android Application via a Bluetooth interface.

Engineering Requirement 4:

The portable device subsystem will wirelessly send images to the mobile platform from at least twelve feet away.

Verification Method:

Step 1: The portable device subsystem will be placed at least twelve feet from the mobile device running the mobile application subsystem as measured by a tape.
Step 2: A distinct image will be initiated by the mobile device.
Step 3: The system will pass the test when the image is transmitted successfully from the portable enclosure to the mobile device

Requirement 5 - Fluorescence Spectrum Analysis

Project Partner Requirement 5:

The application needs to be able to process and analyze the images taken.

Engineering Requirement 5:

The application will generate a fluorescence emission spectrum graph.

Verification Method:

Step 1: The mobile application subsystem will be demonstrated with a user selecting arbitrary image capture parameters while a given sample is loaded into the portable device subsystem.
Step 2: The button to take an RGB analysis of the sample will be clicked.
Step 3: The resulting RGB spectrum will be observed when it is saved to the mobile device.

Requirement 6 - System Effective Power

Project Partner Requirement 6:

The system shall be fit for field use.

Engineering Requirement 6:

The portable device shall capture at least 30 images with one battery life.

Verification Method:

Step 1: Ensure that the device is not plugged into mains (120 VAC) power.
Step 2: The portable device subsystem will be demonstrated connecting to the mobile phone application subsystem
Step 3: The mobile phone application will initiate a picture from the portable subsystem
Step 4: Step three will be repeated until thirty image captures are initiated. This test is meant to measure the portable battery life, therefore resultant image quality will not be a factor.

Requirement 7 - Battery Life

Project Partner Requirement 7:

The system shall be fit for field use.

Engineering Requirement 7:

The portable device will have a battery life of over 24 hours of continuous "on" operation without data transmission.

Verification Method:

Step 1: Ensure that the device is *not* plugged into mains (120 VAC) power.
Step 2: The portable device will be demonstrated to be "on" by flipping a switch on the side to the "on" position and noting the presence of an indicator LED.
Step 3: Immediately after Step 2 a timer will be started.
Step 4: The portable device will be checked at two specified intervals during a 24 hour period measured by the timer in step 3. At each check time, one time stamped photo will be taken such that an indicator LED is clearly visible. These intervals are listed below.
Step 5: When at least 24 hours has elapsed since starting the timer in step 3, One final check will be performed. One final time stamped photo will be taken such that an indicator LED is clearly visible.

Step Number	**Interval Timing (Hours: Minutes)**
4.A.	06:00- 10:00
4.B.	14:00- 18:00
5.	>24:00

Requirement 8 - Status Detection

Project Partner Requirement 8:

The system shall be fit for field use.

Engineering Requirement 8:

The mobile application subsystem will have an array of 5 status LEDs corresponding to the status of the portable device subsystem.

Verification Method:

Step 1: Request an image to be taken with the BLUE excitation LED on, and when the camera goes to take the picture look at the status LEDs, and make sure that both the BLUE excitation LED is on, as well as the corresponding status LED (far left).

Step 2: Request an image to be taken with the GREEN excitation LED on, and when the camera goes to take the picture look at the status LEDs, and make sure that both the GREEN excitation LED is on, as well as the corresponding status LED (second from far left).

Step 3: Request an image to be taken with the AMBER excitation LED on, and when the camera goes to take the picture look at the status LEDs, and make sure that both the AMBER excitation LED is on, as well as the corresponding status LED (third from far left).

Step 4: Request to run the error code example from the testbench script (execute cmd), and make sure the 1st error code LED turns on and off (third from far right).

Step 5: Request to run the error code example from the testbench script (execute cmd), and make sure the 2nd error code LED turns on and off (second from far right).

Step 6: Start with the system off, and make sure that the power status LED is off (far right). Then turn it on with the power button, and make sure that the power status LED is on.

2.2. Design Impact Statement

A portable microscopic fluorescence detection platform, though an unwieldy title, is comprehensively descriptive. The scientific method of fluorescence spectroscopy utilizes a light source to excite a sample and detect the presence and concentration of a compound [1]. This project will port this technology to a mobile platform, one which will be capable of analyzing these samples in the field and communicating its findings wirelessly to a mobile application on an Android device. This kind of technology has applications most saliently in healthcare. Use of this technology can aid in diagnosis of patients, as well as identification of certain drugs in the body. Bringing this to a mobile platform would extend its availability greatly.

When a new project is funded and pursued, as a modern engineer it is critical that thorough research and consideration into what the potential impacts, both positive and negative, of the project will be. This is important in order to avoid situations like the racial bias exposed in facial recognition technologies. It has been found that "current implementation of these technologies [involve] significant racial bias, particularly against Black Americans" [2]. However, this is just one example of a complicated issue. Along with this there are very relevant concerns about the ethicality of AI development and where that may take humanity [3]. In this section, we will explore the potential impacts of elements of our project including the use of Lithium Ion batteries, electronics recyclability, economic availability, and more.

First we will explore impacts that a developing technology can have on health, safety, and welfare. One potential impact of our project would be inadvertently pricing out a certain segment of the population. One of the goals of our project is to create an affordable (relative to the current alternative) platform for fluorescent microscopy. Current implementations of the fluorescence microscopy are often priced at least $2,000 USD and up to $20,000 USD and higher [4]. Despite being confident we will be under the $2,000 USD mark, with the price of hardware we risk still pushing portions of people out of the window of affordability. This is especially important because access to this technology is not a luxury, it is an important part of healthcare.

Strong relationships between socioeconomic status and availability to healthcare have been found which indicate that the poor are not receiving adequate levels of healthcare relative to the wealthy [5]. See Figure 1 below for a view of the disparity. One of the goals of this project is to achieve a level of affordability as to avoid continuing this trend of wealth skewed healthcare. Careful consideration must be given to the parts selected for use in the product in order to ensure we yield a final product that is affordable to a wide range of economic abilities. Balancing this with our goal of creating an accurate and usable platform will be critical.

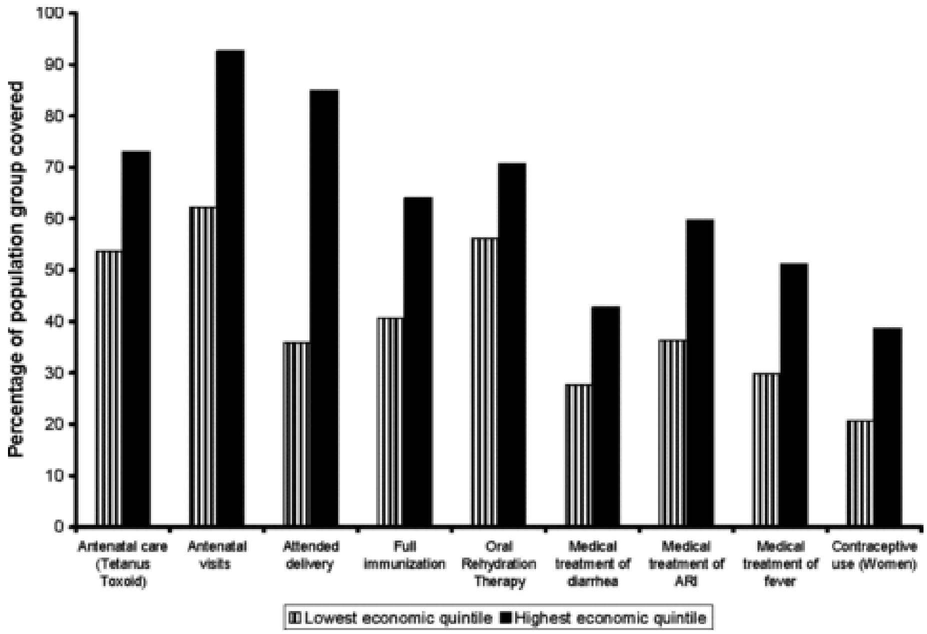


Figure 1: Use of Health Services by Lowest and Highest Economic Quintiles in Low and Middle Income Countries [6]

Because of the goal of mobility for this platform, the use of a Lithium ion Battery (henceforth referred to as an LiB) was decided on. An LiB is necessary to provide power to the platform when no main power is available, furthering our goal of availability, especially to developing countries that, according to the United Nations Conference on Trade and Development (henceforth referred to as UNCTD), still don't have electrical availability for over half of their populations and what access there is is often unreliable [7]. But the choice to use LiBs is not one we take lightly. One part of an LiB is known as the cathode (See Figure 2). This cathode contains two specific elements which are cause for concern. Firstly it contains lithium ions which are used as the transmitter for electricity in the battery [8]. However, lithium is unstable on its own so the second element, cobalt, is used to stabilize it along with other elements not relevant to this assessment [9].

It is with the extraction of cobalt, that we find a harrowing social impact of LiB use. Cobalt is mostly sourced from the Democratic Republic of the Congo (henceforth referred to as the DRC). More than 60% of the world's cobalt comes from the DRC [10]. However, this extraction process is largely unregulated or ineffectively regulated, especially in what are known as Artisanal and Small-Scale Mines (henceforth referred to as ASMs). Reuters reports that “150,000-200,000 artisanal miners [are] currently working [in] cobalt deposits in Congo with more than a million others [are] directly economically dependent on their activity” [11]. The lack of regulation has led to dire impacts to the Congolese people, many of whom entirely rely on these mines to feed themselves and their families. This includes but is not limited to: child labour, exploitation of labour, and increased rates of birth defects on account of cobalt mining induced toxicity on account of both ASMs as well as large scale industrial mining operations [12][13].

We are unfortunately at the mercy of the current world-wide reliance on LiBs. However, there is still hope. According to the Office of Energy Efficiency and Renewable Energy (US), “industry has recognized the risks of [cobalt] dependency, and many battery manufacturers and end users have established ambitious goals to move to low- or no [cobalt] containing cathodes” [14]. While this move to low cobalt or cobalt free batteries is likely more economically driven than human rights driven, we should accept progress where we can get it. In the future when low-cobalt or cobalt free batteries are more available, a port of our project to one such battery would be prudent.

We will explore two prominent possible environmental issues relevant to this project. First, recall the elements used in LiBs discussed earlier, cobalt and lithium. According to the Institute for Energy Research (henceforth referred to as IER), lithium extraction uses an approximate 500,000 gallons of water per metric ton of lithium extracted [15]. IER continues to describe the process as drilling holes into salt flats and extracting a mineral dense brine. This brine is then repeatedly evaporated in evaporation pools to purify for lithium carbonate. IER states that “there is the potential for toxic chemicals to leak from the evaporation pools into the water supply” and “in Nevada, researchers found impacts [from lithium mining] on fish as far as 150 miles downstream.” Beyond just water depletion and soil / water contamination, the UNCTD reported that cobalt mines in the DRC may also contain sulphur [16]. They continue to state that this sulphur can create sulfuric acid through what is called acid mine drainage, whereby distrubed sulphur minerals can mix with air and water and create the sulfuric acid. This acid is detrimental to nearby waterways which are relied on not just by aquatic life, but also by the people and animals nearby to it.

The second concern is especially relevant in recent years with the increasing discourse around climate change and the environmental impact of technologies [17]. The concern being, what happens to the product after its life cycle is complete? E-waste and the rise in end-of-life electronics has been a large issue recently [18]. This is in part due to inefficient or nonexistent methods for recycling some key components of electronics [19]. Since our device is not specifically designed to last for a very long time, it is especially important to develop sustainably. This includes using easily reusable components like the Raspberry Pi and the camera. Until a time when cleaner battery alternatives or better and more available LiB recycling methods have been developed, that part of our e-waste is unfortunately unavoidable.

Finally, perhaps the most obvious impact that comes to mind is the economic impacts. The share of GDP that health expenditures make up has been steadily growing from around 5% in 1962, to around 17% in 2018 and it's only expected to increase in the coming years [20]. While our product is specifically designed to provide portable and, more importantly, cheaper alternatives to the current fluorescence microscopy devices on the market right now, it is still important to consider how our platform could be used in the future for economic exploitation. Examples of companies in the public health sector drastically inflating the prices of products suddenly and without proper justification have occurred [21]. Having a comprehensive idea of where the product is going and who it will be sold to or where it will be used is critical so that the ethical implications can be considered. New technologies can easily be price inflated because there is often little competition to keep the prices in check. Because the price that companies inflate medical equipment and fees to is often largely agnostic of the actual cost to produce and use, mitigation of this impact comes in careful evaluation of who we develop the project for, their intentions, and their values.

Countering these potential impacts comes down largely to mitigation rather than response. Mitigating the health impacts can be done through conscientious component selection to optimize for end sale price. Addressing cultural and social impacts is less clear. Due to the current world-wide reliance on LiBs in technology with few readily available and reasonable alternatives on the market, we unfortunately have little in the way of autonomy over our use of them. The inclusion of a LiB in our design is thus a hard constraint. With recent pushes for increased human rights protections both within the DRC and from lawsuits to major companies, change in the social and cultural impacts seems imminent [11][12]. When we consider our possible environmental impacts, we will select parts with reusability in mind. Choices to include general purpose parts like a

Raspberry Pi and a standard camera allow for recyclability into other projects. Once more efficient, cleaner batteries are available or once LiB battery recycling is more developed, the environmental impact of LiBs will be partially or fully mitigated. Finally, a portable and affordable platform for fluorescence microscopy has the potential to increase availability of this technology to the less affluent globally, but also has the potential to be price-inflated unfairly. In order to avoid this, careful consideration is given to who the project is being developed for and their intentions with it. There are many pitfalls when developing new technologies. There is serious potential for harm, but there is also serious potential for good. Only with careful consideration can we maximize the good while minimizing the harm.

2.3. Risks

Risk ID	**Risk Description**	**Risk Category**	**Risk Probability**	**Risk Impact**	**Performance Indicator**	**Responsible Party**	**Action Plan Synopsis** (A full risk action plan document can be found at [this link])
R1	Code being deleted/ misplaced	Technical	25%	M	Do we lose code?	Jacob	Avoid: Keep remote repository updated for backup and version control
R2	Damaged parts	Technical/ Cost	50%	M	Did we break something/does it not work due to hardware	Ben	Transfer: Re-create/fix said parts
R3	Faulty part	Cost/ Timeline	25%	M	Is the part not meeting it’s design specifications	Michael	Transfer: RMA the part / get replacement
R4	Lack of communicati on	Communic ation	80%	H	Someone stops talking for a week	Jacob	Reduce: Contact instructors and project partner and divide workload
R5	Miscommuni cation	Communic ation	60%	M/L	Work done in area not intended or needed	Graham	Reduce: Constant and clear communication to ensure point is clear, if it happens redo
R6	Ordering wrong parts	Technical/ Cost	20%	M/H	Did we order a part, we didn't mean to?	Ben	Reduce:Ensure peer review before ordering is placed
R7	Hardware shipping delays	Timeline	30%	M/H	Not meeting shipping dates	Michael	Transfer: Update group and look at potential alternatives
R8	Parts not being	Timeline	40%	H	Expected parts unavailable	Michael	Reduce: Check early and often to ensure redesign is

	available						not necessary
R9	Parts being too expensive	Timeline/ Cost	30%	M	Expected parts costing MORE than expected	Michael	Avoid: Hard budgets in place and careful planning so that feature creep and cost creep do not occur
R10	Not meeting design/ development timelines	Timeline	55%	L/M/H	Failed to meet timeline deadline	Graham	Reduce: Constant communication and request help from other group members.

Table 5: Risk Table

2.4. References and File Links

[1] P. M. V. Raja, A. R. Barron. Mar. 21, 2021 "Fluorescence Spectroscopy" Chemistry LibreTexts. https://chem.libretexts.org/Bookshelves/Analytical_Chemistry/Physical_Methods_in_Chemistry_and_Nano_Science_(Barron)/01%3A_Elemental_Analysis/1.11%3A_Fluorescence_Spectroscopy (Accessed December 1, 2021)

[2] A. Najibi. Oct. 24, 2020. "Racial Discrimination in Face Recognition Technology". Harvard, Science in the News. https://sitn.hms.harvard.edu/flash/2020/racial-discrimination-in-face-recognition-technology/ (Accessed October 25th, 2021)

[3] C. Pazzanese. October 26, 2020. "Ethical Concerns Mount as AI Takes Bigger Decision-Making Role in More Industries". The Harvard Gazette. https://news.harvard.edu/gazette/story/2020/10/ethical-concerns-mount-as-ai-takes-bigger-decision-making-role/ (Accessed October 25th, 2021)

[4] New York Microscope Company. "Fluorescence Microscopes". MicroscopeInternational.com https://microscopeinternational.com/fluorescence-microscopes/ (Accessed November 30, 2021)

[5] D.J. McMaughan, O. Oloruntoba, M.L. Smith. "Socioeconomic Status and Access to Healthcare: Interrelated Drivers for Healthy Aging". *Front Public Health.* Published June 18, 2020.

[6] D. H. Peters, et al. “Poverty and Access to Health Care in Developing Countries”. *The New York Academy of Sciences.* Published July 25th, 2008.

[7] United Nations Conference on Trade and Development. “Over Half of People in Least Dev. Countries Lack Access to Electricity”. UNCTAD.org. Published July 1st, 2021. https://unctad.org/topic/least-developed-countries/chart-july-2021 (Accessed Dec. 1st, 2021)

[8] Samsung. “The Four Components of a Li-ion Battery”. SamsungSDI.com. https://www.samsungsdi.com/column/technology/detail/55272.html?listType=gallery (Accessed Dec. 1st, 2021)

[9] M. Li, J. Lu. :Cobalt in Lithium-Ion Batteries”. *Science.* Vol. 367, no. 6481. Pp. 979-980. Published Feb. 28, 2020.

[10] Statista. (Feb. 2021) Distributed by Statista. https://www.statista.com/statistics/339834/mine-production-of-cobalt-in-dr-congo/

[11] A. Home. “Cobalt, Congo, and a Mass Artisanal Mining Experiment”. Reuters.com. Published May 13th, 2021. https://www.reuters.com/business/energy/cobalt-congo-mass-artisanal-mining-experiment-andy-home-2021-05-13/ (Accessed Dec. 1st, 2021)

[12] A. Kelley. “Apple and Google Named in US Lawsuit over Congolese Child Cobalt Mining Deaths.” The Guardian. Published Dec. 16th, 2020. https://www.theguardian.com/global-development/2019/dec/16/apple-and-google-named-in-us-lawsuit-over-congolese-child-cobalt-mining-deaths (Accessed Dec. 1st, 2021)

[13] Amnesty International. “DRC: Alarming Research Shows Long Lasting Harm from Cobalt Mine Abuses". Published May 6th, 2020. https://www.amnesty.org/en/latest/news/2020/05/drc-alarming-research-harm-from-cobalt-mine-abuses/ (Accessed Dec. 1st, 2021)

[14] Office of Energy Efficiency and Renewable Energy. “Reducing Reliance on Cobalt for Lithium-ion Batteries”. Energy.gov. Published April 6th, 2021. https://www.energy.gov/eere/vehicles/articles/reducing-reliance-cobalt-lithium-ion-batteries (Accessed Dec. 2nd, 2021)

[15] Institute for Energy Research. "The Environmental Impact of Lithium Batteries." Published Nov. 12th, 2020. https://www.instituteforenergyresearch.org/renewable/the-environmental-impact-of-lithium-batteries/ (Accessed Dec. 2nd, 2021)

[16] United Nations Conference on Trade and Development. "Developing Countries Pay the Environmental Cost of Electric Car Batteries". UNCTAD.org. Published July 22nd, 2020. https://unctad.org/news/developing-countries-pay-environmental-cost-electric-car-batteries (Accessed Dec. 2nd, 2021)

[17] C. Davenport. October 28, 2021. "Climate Change Became the Largest Part of Biden Spending Bill". The New York Times. (Accessed October 25, 2021)

[18] EPA. "Cleaning Up Electronic Waste". United States Environmental Protection Agency. https://www.epa.gov/international-cooperation/cleaning-electronic-waste-e-waste (Accessed Oct. 25th, 2021)

[19] H.Y. Kang, J.M. Schoenung. Electronic Waste Recycling: A Review of U.S. Infrastructure and Technology Options. *Science Direct.* (Published December, 2005)

[20] PGP Foundation. April 20, 2020. "Why Are Americans Paying More for Healthcare". Peter G. Peterson Foundation. https://www.pgpf.org/blog/2020/04/why-are-americans-paying-more-for-healthcare (Accessed Oct. 26th, 2021)

[21] X. Hua, N. Carvalho, M. Tew. Expenditures and Prices of Antihyperglycemic Medications in the United States: 2002-2013. *Jama Network.* (Published April 5, 2016)

2.5. Revision Table

10/20/2021	Jacob: Initial Document Creation.
10/27/2021	Ben, Jacob, Michael, Graham: 2.1 Requirements and 2.3 Risks added.

10/29/2021	Ben: Formatting to 2.1 and 2.3. Revision Table added.
11/12/2021	Ben: Section 2.1 reformatted to match rubric. Verification methods added for each requirement.
11/12/2021	Jacob: General formatting revisions.
12/1/2021	Graham: Changed certain Requirements to fit feedback given in "Project Document: Section 1 and 2".
12/1/2021	Jacob: Heavily reformatted this section's text, font, and spacing.
12/2/2021	Jacob: Revised risk table action plans to include keywords
12/3/2021	Ben: Included link to Risk Action plan description sheet to be more informative without ruining formatting.
3/6/2022	Team: Changed 2.1.8 to re-address status LEDs, and revised previous requirements to fit current project standards
5/6/2022	Ben: Changed Verification methods in requirement 4 (Section 2.1.4) and requirement 6 (section 2.1.6) to match finalized methods in section 5.
5/6/2022	Jacob: Ported in my design impact assessment to section 2.2 and general formatting.

3. Top Level Architecture

3.1. Block Diagrams

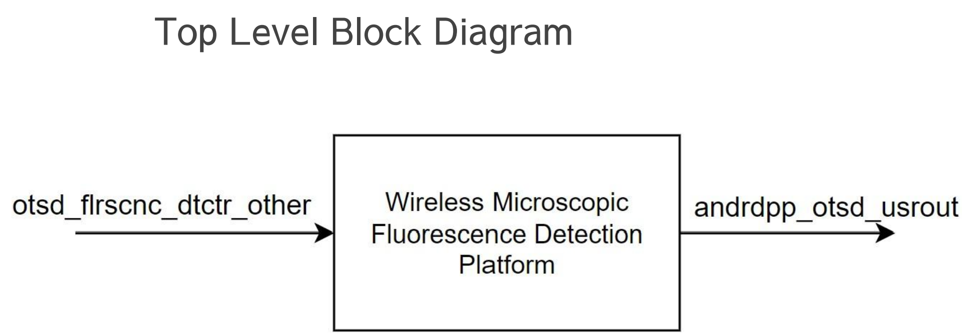


Figure 2: Black Box Diagram

System Block Diagram

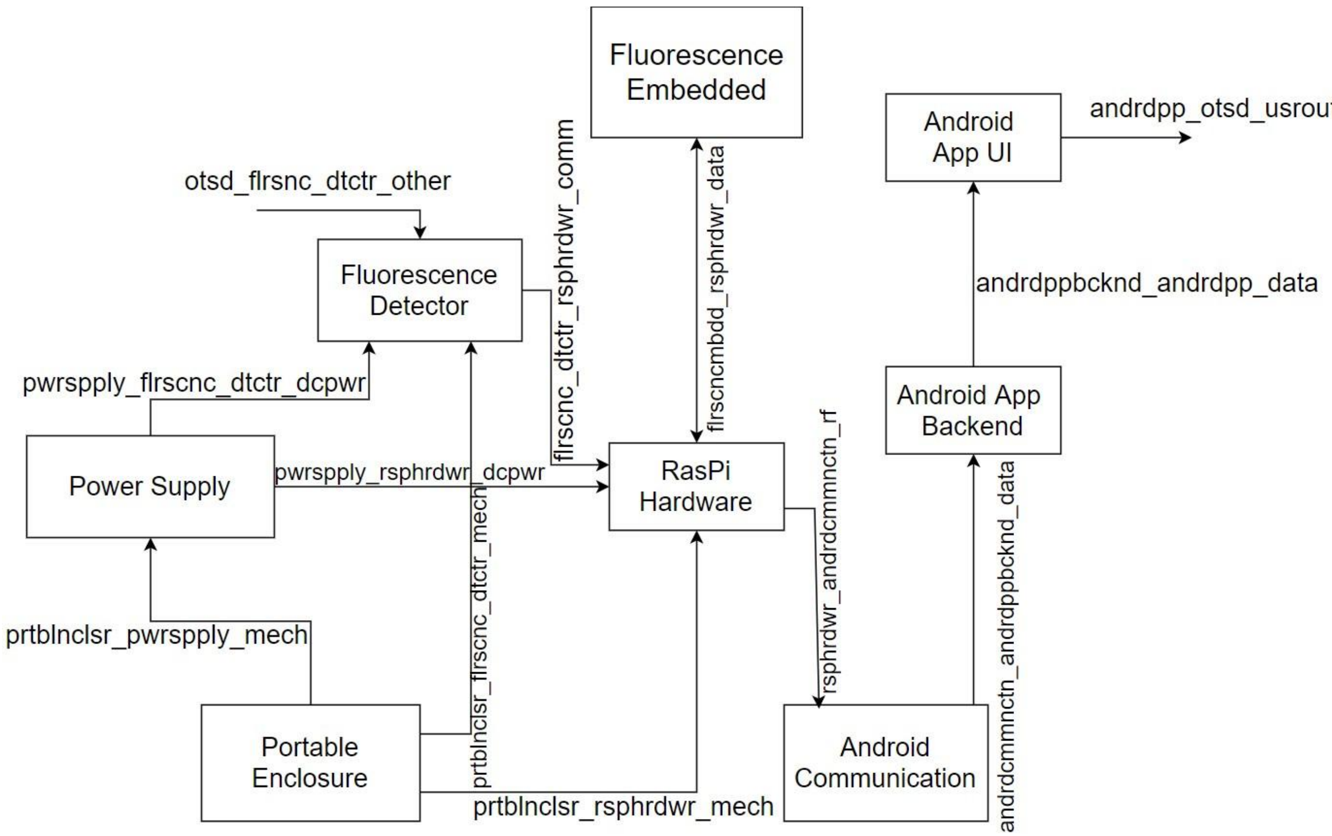


Figure 3: Block Diagram

3.2. Block Descriptions

The Eight functional blocks of the system introduced above are described in detail in this section. The ownership of these blocks is divided according to the following chart.

Block Name	Block Owner	Block Type
Fluorescence Detector	Ben Adams	Sensing
PowerSupply	Ben Adams	Power Supply
PortableEnclosure	Michael Burlachenko	Enclosure
RasPiHardware	Michael Burlachenko	PCB
AndroidAppUI	Graham Donaldson	User Interface
AndroidCommunication	Graham Donaldson	Translation
AndroidAppBackend	Jacob Reger	Processing
FluorescenceEmbedded	Jacob Reger	Code

Table 6: Block Champion Table

Fluorescence Detector

The fluorescence detector is the entire system's primary sensor input. This electromechanical system will house a sample slide, optically excite that slide when supplied with signals, and then digitally encode an image of the sample before sending it to the Fluorescence Embedded code to be processed and communicated.

Fluorescence Embedded

This embedded code block will control the general data inputs and outputs for the portable system. This includes the camera, LEDs, and Android Communication code. This code must interpret signals and decode these signals into parameterized inputs and outputs. It will be coded in Python to make interfacing with the Python coded Raspberry Pi sided communication code easier. The runtime overhead of Python should not prove to be a problem for this use case.

Power Supply

This block will charge a battery from the power of a regulated DC supply. This battery power will then be converted into usable power for the rest of the portable system. The battery is necessary to achieve portability of the detector. Battery capacity will depend on power consumption of devices such as sensors, regulators, raspberry pi.

Portable Enclosure

This enclosure will house and protect all of the components of the portable system. Designing for portability also necessitates secure fastening of all electrical and mechanical components. This enclosure will also have a loading compartment to allow users access to the fluorescence detector. Samples will be secured to the portable enclosure. We are planning to 3D print the enclosure with PLA plastic but we might use composites as well.

RasPi Hat

This hardware block encompasses all of the components that directly drive the Raspberry Pi based microcontroller. A custom PCB will be designed and built. The PCB is planned to be a hat for Raspberry Pi which will connect all devices and components of the portable detector. The hat will provide a secure and easy to access connection for components.

Android Communication

This block acts as sort of the translator between the Android App Backend, and the actual raspberry pi. This is the part where Bluetooth is used to transmit the image from the Raspberry Pi to the mobile device, and send various commands from the mobile device to the Raspberry Pi. This block includes code present on both the raspberry pi as well as the Android mobile app and will interface with both sides via a function call.

Android App Backend

All data received from the Android Communication block sent to the Android App will be processed for fluorescence detection. The transmitted images will be saved to the phone running the app. Settings for the detection platform will be controlled using the GUI block, but will be interpreted and readied for transmission here in this block before being sent to the Android Communication block. This will be coded in Kotlin.

Android App UI

This UI will act as the main interface between the user and the rest of the portable system. It will let the user control various aspects of the camera, and the lights inside. It will communicate these values with the Android App Backend block, which will do most of the heavy lifting in receiving the data/processing it before sending it off to the Android Communication block where it will be sent to the device. The software will be developed with Kotlin.

3.3. Interface Definitions

Name	From	To	Properties
otsd_flrscnc_dtctr_other	Outside	Fluorescence Detector	1. Sample Slide Size = 25mm (±2mm) x 25mm (±2mm) x 1mm (±0.2mm) 2. Sample Temperature = -20°C-60°C
pwrspply_flrscnc_dtctr_dcpwr	Power Supply	Fluorescence Detector	1. Vmax = 5.2 V 2. Vmin = 4.8 V 3. Ipeak = 100 mA 4. Inominal = 45 mA
pwrspply_rsphrdwr_dcpwr	Power Supply	RasPi Hardware	1. Vmax = 5.2 V 2. Vmin = 4.8 V 3. Ipeak = 3.4 A 4. Inominal = 3 A
flrscnc_dtctr_rsphrdwr_comm	Fluorescence Detector	RasPi Hardware	1. Vmax = 3.6 V 2. Vmin = 2.9 V 3. Ipeak = 350 mA 4. Inominal = 250 mA (Active) 5. Protocol: 15-pin MIPI Camera Serial Interface (CSI-2)
andrdpp_otsd_usrout	Android App UI	Outside	1. Image Resolution = 1280 x 720 pixels 2. Visual fluorescent excitation response graph
prtblnclsr_pwrspply_mech	Portable Enclosure	Power Supply	1. Secures Component adhering to the verification method detailed in Section 2.1.1. 2. Component must be able to be removed/replaced with hand tools.
prtblnclsr_flrscnc_dtctr_mech	Portable Enclosure	Fluorescence Detector	1. Secures Component adhering to the verification method detailed in Section 2.1.1. 2. Component must be able to be removed/replaced with hand tools.
prtblnclsr_rsphrdwr_mech	Portable Enclosure	RasPi Hardware	1. Secures Component adhering to the verification method detailed in Section 2.1.1. 2. Component must be able to be removed/replaced with hand tools.
andrdcmmnctn_andrdppbcknd_data	Android Communication	Android Backend	1. Needs to set a flag when the Bluetooth device is not connected. 2. Needs to store image data as .IMG files
andrdppbcknd_andrdpp_data	Android Backend	Android App UI	1. Must retreive stored data in less than 1 second. 2. Must store an array for user-input image parameters.
flrscncmbddd_rsphrdwr_data	Fluorescence Embedded	RasPi Hardware	1. Needs to configure Camera Serial Interface (CSI-2) of the hardware block. 2. Needs to configure SPI of the hardware block. 3. Needs to configure I/O of the hardware block.
rsphrdwr_andrdcmmnctn_rf	Raspi Hardware	Android Communication	1. Needs to compress image data with a Compression Ratio >5 2. F = 2.4GHz 3. Minimum Range = 10m

Table 7: Interface Definition Table

3.4. References and File Links

N/A

3.5. Revision Table

11/18/2021	Jacob: Initial create and ported over the work we had done in the portal tool for block creation previously.
11/19/2021	Ben: Populated Sections 3.2 and 3.3. Replaced the system block diagram picture (Section 3.1.2) with one more easily readable.
11/19/2021	Michael: Added interfaces' implementation plans
11/19/2021	Jacob: Added content to sections 3.2.1-2 and 3.2.6-8 and revised.
12/1/2021	Ben: Re-uploaded less blurry block diagram images, fixed the Interface definitions chart.
12/1/2021	Jacob: Heavy reformatted text and titles
12/3/2021	Michael: Added risk 3,7,8,9 full description. Revised RasPi hardware block description. Revised section 2 and 3

4. Block Validations

4.1. Fluorescence Detector

Description

This Fluorescent detector block consists of a mechanical enclosure and simple electronic and optical system to test samples. Samples are deposited onto a square glass slide and placed into the mechanical/ optical enclosure. The fluorescence detector optically stimulates the sample as commanded by the "Fluorescence Embedded" code block through the hardware of the "RasPiHAT" hardware block. Optical stimulation will be achieved through three narrow bandwidth LEDs with different frequencies. The Fluorescence detector houses a camera to detect the fluorescent response of the sample. This camera is considered as part of the "Fluorescence Embedded" design block.

Considered as a black box diagram, the Fluorescence Detector is essentially a passive electro-optical component. Fluorescence samples are loaded through the otsd_flrscnc_dtctr_other interface; LED stimulation is achieved by the rspht_flrscnc_dtctr_dsig interface; The system is secured to the portable enclosure through the prtblnclsr_flrscnc_dtctr_mech interface; Finally, the camera is driven by the flrscncmbddd_flrscnc_dtctr_dsig interface. This design block is championed by Ben Adams.

Design

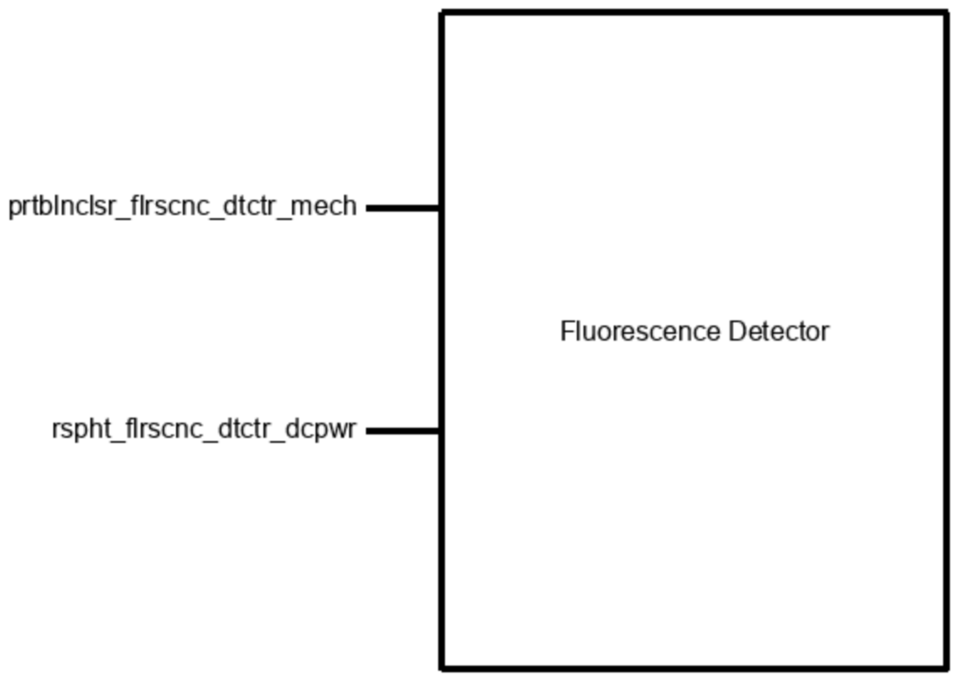


Figure 4: Power Supply Overview Block Diagram

This design block was built around redesigning and updating an existing design project- referred to herein as the "inherited detector." Documentation on the inherited detector was existent but lacking. After a review of the inherited documentation, a redesign was envisioned consisting of flipping the inherited detector upside down and redesigning the mechanical "Lid" component as a new "Base" component. The new base component will house three redesigned LED boards. This design allows for all of the other mechanical components from the inherited project- Front Casing, Back Casing, Shank, Stage, Rod, Tray, Lens, and CMOS Camera- to be reused. This design will depend on the "RasPiHAT" replacing the inherited Electronic Control Unit pictured in Figure 2.

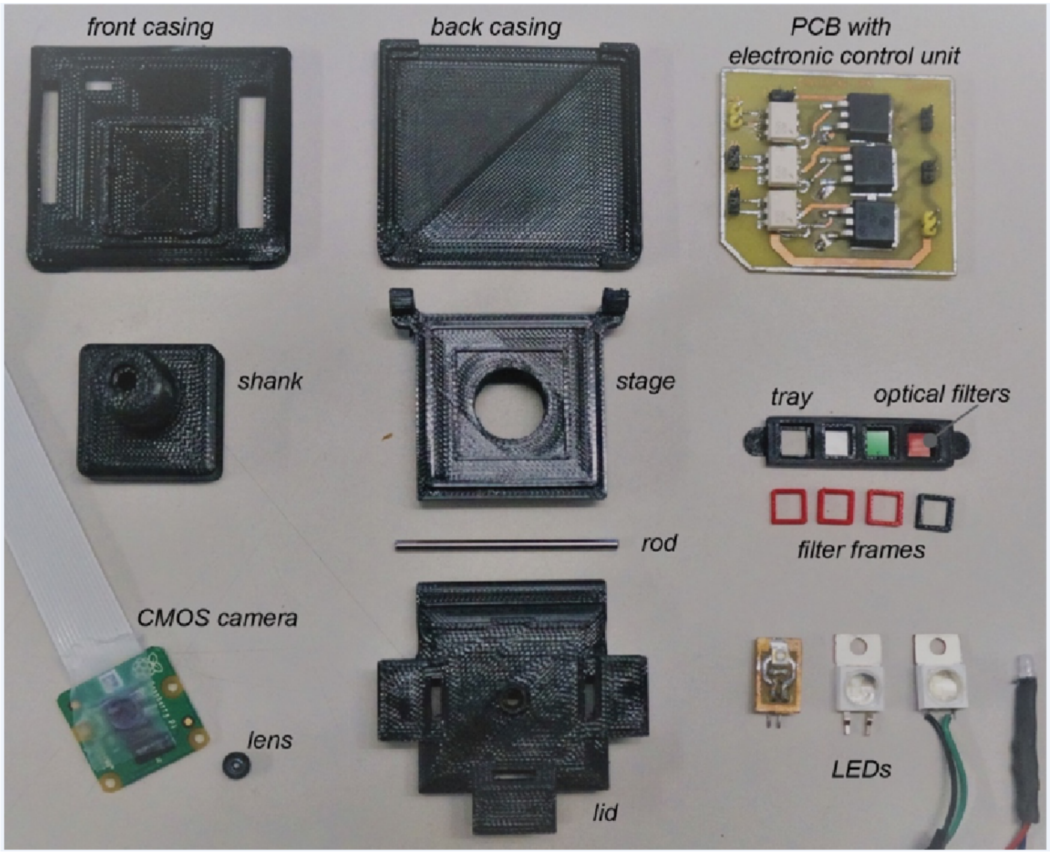


Figure 5: Inherited Detector hardware; See below for a detailed discussion on updating this system

LED Circuit Design

This design was completed by first selecting two families of LEDs with acceptable optical properties, then designing a circuit to work modularly with either LED installed. Note that both LEDs are not expected to be installed at the same time.

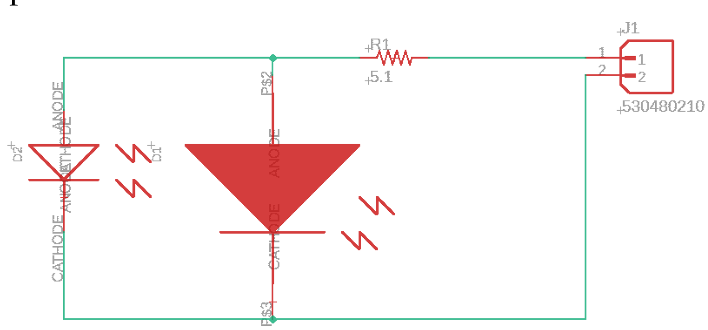


Figure 6: LED Circuit Schematic

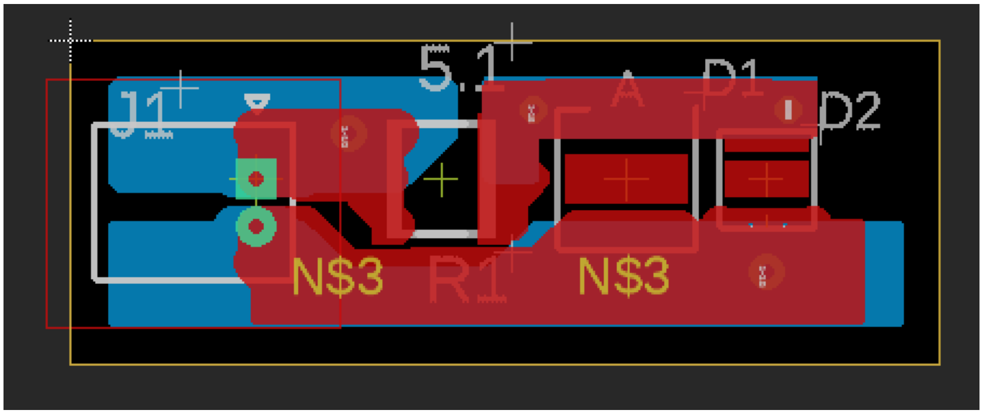


Figure 7: LED Circuit PCB Design

Mechanical Base Design

The mechanical base design block was limited by the constraint of mating with the mechanical components of the inherited design. Inherited design mechanical components were further limited to one set of physical components and one set of .STL files. Physical measurements were taken from the actual components because these should include any manufacturing defects. A new mechanical design was generated from these measurements with cutouts to house the revised modular LED board design. Dimensions and tolerances for this design are given in Subsection 6.2.

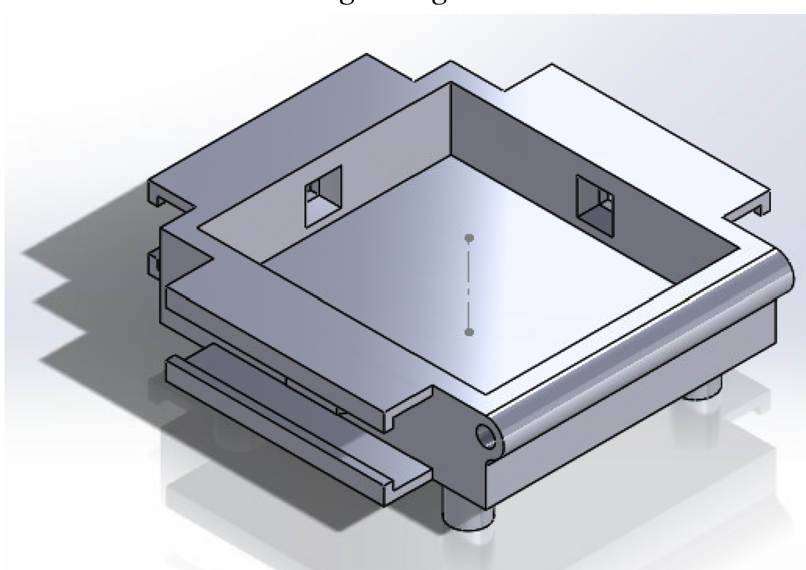


Figure 7: Mechanical Base Design Rendering

General Validation

The Fluorescent Detector design block is both relatively passive and the heart of the entire Portable Microscopic Fluorescence Detector

(PMFD) System. Our overall project benefited from an inherited project, and the fluorescence detector design was able to take advantage of inherited design's considerations for optical considerations like focal lengths, lens detection, optical isolation, etc. The mechanical base design will leverage and improve upon these inherited design considerations. Documentation from the inherited detector project did not include any mention of fluorescent detection accuracy, data on the fluorescent excitation LEDs used, or data from samples collected, which was interpreted to mean that the system accuracy was lacking. Care was taken to only make design choices that improved system performance, and the re-selection of excitation LEDs was identified by our project partner as a primary system concern.

Mechanical Base Validation

The mechanical base will be made from black 3d printed PLA material following the design considerations from the inherited project. The material must be black to minimize optical interference from reflected light. The mechanical base must also mate with the inherited mechanical components such that 1.) The sample housing space is optically isolated from outside light and 2.) The sample remains at an acceptable focal length despite the entire mechanism being flipped upside down. Both of these parameters were ensured by the careful measurement of the existing components with mechanical calipers.

The mechanical base should also optically mate with the excitation LED boards to both ensure efficient fluorescence excitation for later calculations and avoid interference with other light sources. This will be achieved by the use of 1.) A clear adhesive to mount the board to the mechanical base with minimal optical loss and 2.) Application of optically absorptive (black, opaque) adhesive over the PCB to negate interference. This absorptive media will also be used to pad the slide bed to ensure that slide vibration tolerance is adhered to.

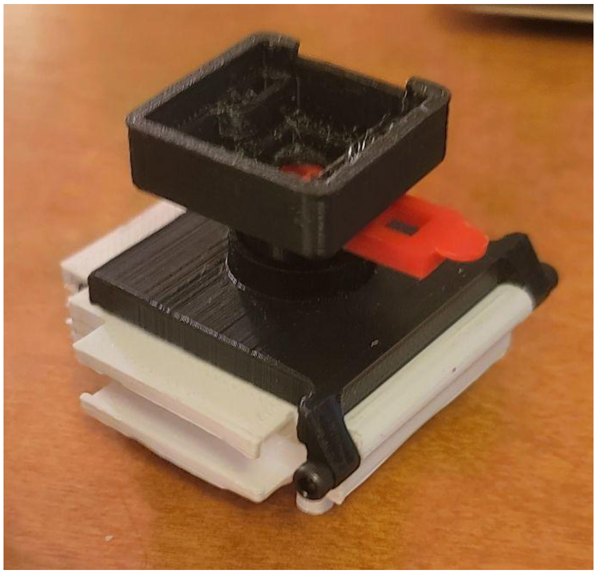


Figure 8: The new base (bottom, white for the prototype) mated with the inherited lid

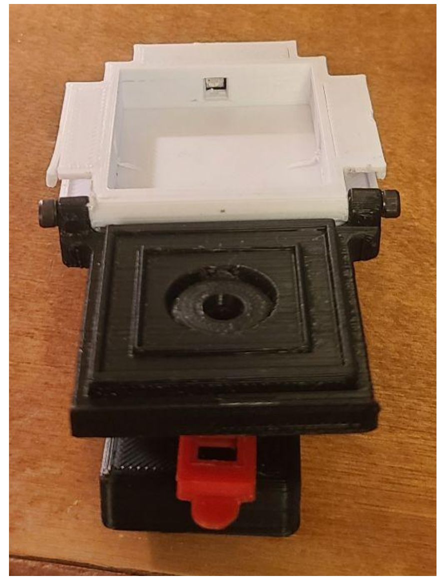


Figure 9: The new base with the LED board's LED visible through a viewing window

LED Circuit Validation

LED components were selected before extensive design was completed to ensure improved system accuracy; This led to the semi-arbitrary constraint that each LED's output spectrum has a width of less than 50nm Full-Width Half Maximum (FWHM). After research, the following Cree LED components were selected: XTEARY-00-0000-000000K01, XBDGRN-00-0000-000000B01 and XBDAMB-00-0000-000000701. A modular board was designed to allow for any LED from the XTE or XBD lines to be replaced with requiring a redesign for the alternate footprints or current re-calculation due to differing diode forward voltages. Figures 5 and 6 were used to estimate CV specifications at various bias points. A 5.1 Ω resistor was selected to allow acceptable current with either LED. Cree LEDs are notably hotter than conventional LEDs, so extra care was taken to allow for thermal dissipation through copper when designing the PCBs. Finalized PCB renders are pictured in Figure 7.

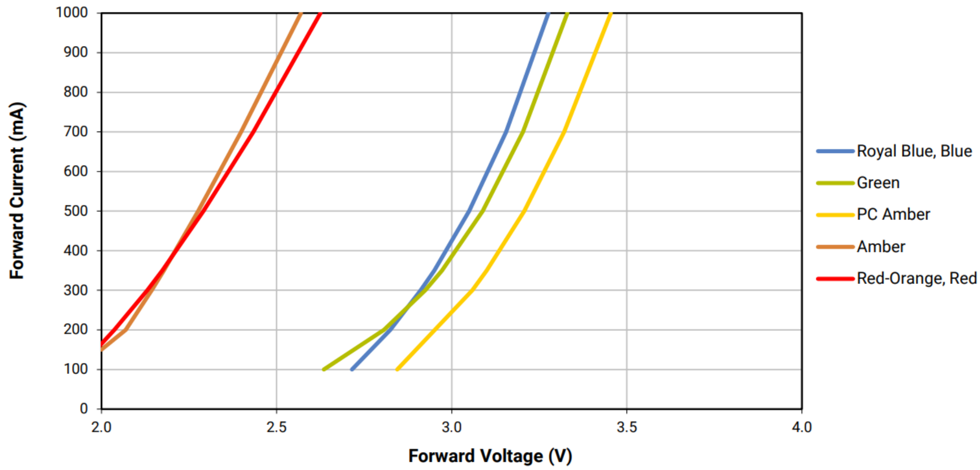


Figure 10: Cree XBD series CV curves

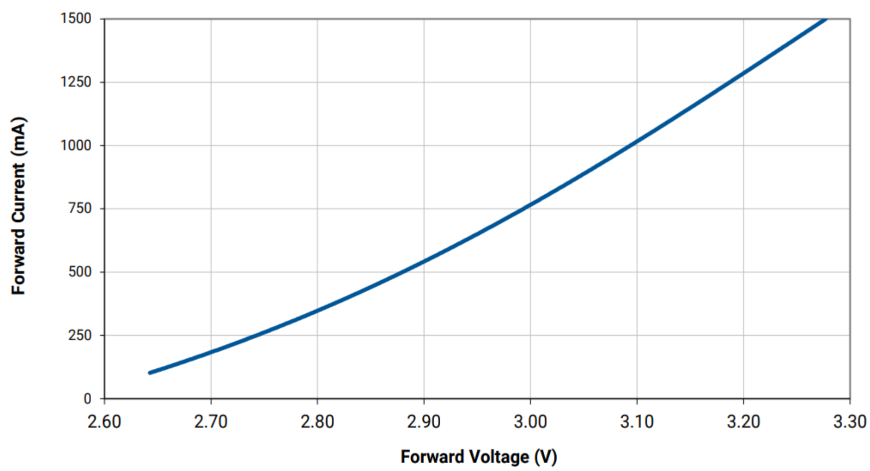


Figure 11: Cree XTE series CV curve for the "Royal Blue" LED

Interface Validation

The following interfaces are given with their validation methods below.

Interface: prtblnclsr_flrscnc_dtctr_mech		
Property	**Reason**	**Validation**
Fasteners: Screw spacing: corners of a 1.25"x1.25" square (on center) ± 0.1"	This offers a stable base for the fluorescence detector and will ensure that both mechanical designs mate together	• Typical 3d printed designs have nominal tolerances of ±0.5mm, corresponding to 0.019." This design takes this into account, see mechanical

		base dimensions in Subsection 6.2.
Fasteners: Slide access port minimum height: 1.2"	This is a large estimate of user finger width chosen for inclusivity and ease of use	• Typical 3d printed designs have nominal tolerances of ±0.5mm, corresponding to 0.019." This design takes this into account, see mechanical base dimensions in Subsection 6.2.
Fasteners: #6-32 screws mounted to enclosure, extending less than 0.25" into the fluorescence detector	This provides easily accessible hardware supply	• Typical 3d printed designs have nominal tolerances of ±0.5mm, corresponding to 0.019." This design takes this into account, see mechanical base dimensions in Subsection 6.2.
Other: Slide access: Slide access port minimum width: 2.5"	This is a large estimate of user finger height chosen for inclusivity and ease of use	• Typical 3d printed designs have nominal tolerances of ±0.5mm, corresponding to 0.019." This design takes this into account, see mechanical base dimensions in Subsection 6.2.

Interface: rspht_flrscnc_dtctr_dcpwr		
Property	**Reason**	**Validation**
Inominal: 450 mA	This provides a reasonable compromise between the forward voltage and luminous intensity of all three different LED models	• The three LEDs have an average forward voltages of 2.7 V at 450 mA (See Figures 5 and 6 from Subsections 6.3-6.4). With the 5.1 Ω resistor, expected current is 450 mA
Ipeak: 500 mA	This provides a reasonable compromise between the forward voltage and luminous intensity of all three different LED models	• All three LEDs have a minimum forward voltages of 2.5 V at 500 mA (See Figures 5 and 6 from Subsections 6.3-6.4). With the 5.1 Ω resistor, and at 5.0 V, peak current is < 500 mA
Vmin: 0 V	LEDs are not subject to reverse bias	• At V=0 V, this circuit provides 0 A of current
Vmax: 5.0V	LEDs are not subject to more than 500 mA of current	• All three LEDs have a minimum forward voltages of 2.5 V at 500 mA (See Figures 5 and 6 from Subsections 6.3-6.4). With the 5.1 Ω resistor, peak current is < 500 mA
Other: Logic-Level: Active	To ensure communication with "Raspi Embedded" and "Raspi	• Connecting the positive terminal to positive voltage ensures this case

high // send signal to turn LEDs on	HAT” blocks	

Verification Process

This block will be tested by the three methods detailed below.

Voltage Supply Methods

A variable voltage power supply/ ammeter will test the rspht_flrscnc_dtctr_dcpwr interface by:

1. Connecting positive 5.0v to the LED positive terminal
2. Connecting power supply ground to the LED negative terminal
3. Ensuring that the LEDs turn on
4. Ensuring that the circuit draw less than 500mA
5. Ensuring that the LEDs turn off

Mechanical Inspection Methods

1. The mechanical enclosure will be physically inspected using mechanical calipers and a square.
2. Bolt hole spacing must measure corners of a 1.25"x1.25" square (on center) ± 0.1”
3. Portable enclosure slide access port must have a minimum width of 2.5"
4. Portable enclosure slide access port must have a minimum height of 1.2"

References and File Links

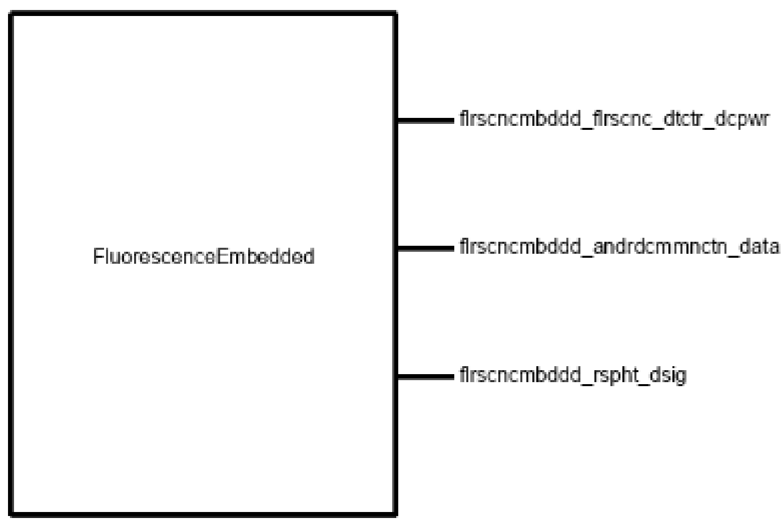
- Mechanical Footprint Drawing: https://drive.google.com/file/d/11T9UgWb4UfL-_TggU3O_KP6RQSkz7KC5/view?usp=sharing
- Mechanical Base Drawing: https://drive.google.com/file/d/1wnCOP8V7AcI4q0z5oenMjjhSQ6vdi4m3/view?usp=sharing
- XBDGRN-00-0000-000000B01 and XBDAMB-00-0000-000000701 (shared) Datasheet: https://cree-led.com/media/documents/ds-XBD.pdf
- XTEARY-00-0000-000000K01 Datasheet: https://cree-led.com/media/documents/XLampXTE.pdf

4.2. Fluorescence Embedded

Description

This block encompasses the functional code for the Raspberry Pi, not including the communications code which will handle the Bluetooth communication between the Pi and the mobile application. This code will be written in Python and run under Python 2.7. Its function in the system will be to control the Raspberry Pi hardware in order to power and control the excitation LEDs, set the debug LEDs, take pictures, analyze RGB spectrums, and send data off to the wireless communications block to be transmitted to the mobile application.

Design



The above block diagram illustrates that the Fluorescence Embedded Code will interface bidirectionally with the Android Communication Block, and will output information through two other interfaces. Those interfaces are with the Fluorescence Detector and the Raspberry Pi HAT. Some C style pseudo code for the logical flow of the embedded code can be seen below.

```
int main()
{
      while(1)
      {
            Broadcast(bluetooth_address); //
      enable bluetooth
                                                  // and
                                              wait for
                                                  //
reception
            Interpret received message
            if(settings command)
                  Update settings
            else if(take picture command)
                  Take picture
                  SendImg(Imgdata);
            else if(rgb command)
                  Take picture
                  Analyze RGB
                  Plot spectrum
                  Send images and spectrum to
phone
            else if(close connection)
                  DisableBluetooth();
                  return 0;
      }
}
```

The exact commands and settings list is too exhaustive to be listed here in this document, but includes picture settings like brightness and exposure time along with commands like quit and rgb. The full list of commands is linked below.

Comprehensive Commands List

General Validation

This block fits the design needs of the system. It will not incur material costs or require parts aside from a raspberry pi because it is exclusively code. It should not require extensive engineering time as there are team members very familiar with Raspberry Pi programming and the code structure is very simple. Because it is coded in Python, its technical performance will lag behind a compiled program written in a language like C++, but since we do not have latency constraints on this project, Python will be more than fast enough to meet our needs. It will interact with other blocks purely through software. To be perfectly accurate, code running in modern operating systems does not directly control hardware, but rather controls it indirectly through system calls to the OS. That is a level of abstraction too low for the scope of this project, so for simplicity we will say that the code controls the hardware. This block interaction will be through software methods like function calls and operator overloads. The same will be true of the interaction between the embedded code and the communications block.

Interface Validation

flrscncmbddd_flrscnc_dtctr_dsig: Output

GPIO Pin Control: `output(pinNum,pinVal)`	This method is the most used method of GPIO pin control of the Raspberry Pi, and sets the pin at pinNum to HIGH or LOW depending on what is passed into pinVal.	Python is fully capable of controlling GPIO pins on a Raspberry Pi using a single function call. The program will be running with full permissions so privilege should not be an issue. This function is part of the RPi.GPIO library and is the standard library for controlling GPIO pins, so it is highly reliable.
Pin Array: 6, 13, 19, and 26 according to BCM naming conventions	The interface will utilize these four listed pins. These pins were selected because they are GPIO pins making them optimal for this simple IO use case.	GPIO pins were selected because of their ease of use in enabling and disabling. These pins will be fully capable of driving the LEDs they will be hooked up to. This is well documented in the RPi data sheets.
Pin Assignments: UV, brightField, blue, and green	These GPIO pins will be responsible for UV, brightField, blue, and green excitation LEDs respectively.	These are chosen because each pin is equally viable for each output.

Vmax: 3.4 Volts	3.4 Volts was selected because the pins will be driven by a 3.3 volt supply from the RPi.	The RPi will be powering these GPIO pins using a 3.3 volt supply, so the pins should be incapable of exceeding this voltage without special hardware.
Vmin: 0 Volts	0 Volts was selected because the pins should never drop to negative voltages given they will be grounded via the RPi as well.	Because the RPi ground will be used, it should be impossible to incur a negative voltage on the GPIO pins.

Flrscncmbddd_andrdcmmnctn_data: Bidirectional

Enable Bluetooth: `Broadcast(btAddr)`	This method will enable bluetooth discoverability and wait for a connection from the mobile application. This method was chosen so that it can be called during the beginning of the program and will wait to execute the rest of the program until it has received the connection. The parameter btAddr was decided on to provide an explicit address to connect to.	This is a simple function call which Python is fully capable of executing. There is no return type so it need not be handled as seen in the pseudo code from Section 2. It need only be called in order to transfer execution control to the Android Communication block.
Image Transmission: `SendImg(ImgData)`	This method was selected because the only transmission to the mobile application will need to be an image. Thus only a method for sending an image is needed. This method will have a void return type	This is a simple function call which Python is fully capable of executing. There is no return type so it need not be handled as seen in the pseudo code from Section 2. It is called only after an image has been taken, so no

	and the ImgData parameter will be built using Pillow, a fork of the Python Imaging Library. This was selected to make handling image files easier.	faulty data will ever be sent. The Pillow library is more than capable of meeting our image storing needs as it is the standard for image handling in Python. It is adopted into large Linux distributions like Debian and Ubuntu so it is highly reliable.
Disable Bluetooth: `DisableBluetooth()`	This method was selected because at the end of the program, we want to close out any open bluetooth connection or discoverability. This helps restore the system to its default state when our execution is finished.	This is a simple function call which Python is fully capable of executing. There is no return type so it need not be handled as seen in the pseudo code from Section 2. It is called only after a 'close connection' command has been received, so it will never close the bluetooth connection unless it is intended.
Data Variable: `Global variable "data"`	This solution was chosen for its ease of use and universal access. This simplifies dealing with variables and given the size of the project, is not a design problem.	The size of this code is fairly small and contained, so the use of global variables will be acceptable. Access to this variable will be available to both this block as well as Android Communication making it a property of their interfacing.

flrscncmbddd_rspht_dsig: Output

GPIO Pin Control: `output(pinNum,pinVal)`	This method is the most used method of GPIO pin control of the Raspberry Pi, and sets the pin at pinNum to HIGH or LOW depending on what is passed into pinVal.	Python is fully capable of controlling GPIO pins on a Raspberry Pi using a single function call. The program will be running with full permissions so

		privilege should not be an issue. This function is part of the RPi.GPIO library and is the standard library for controlling GPIO pins, so it is highly reliable.
Pin Array: 5, 12, 16, 20, 21, 23-25 according to BCM naming conventions	The interface will utilize these eight listed pins. These pins were selected because they are GPIO pins making them optimal for this simple IO use case.	GPIO pins were selected because of their ease of use in enabling and disabling. These pins will be fully capable of driving the LEDs they will be hooked up to. This is well documented in the RPi data sheets.
Pin Assignment: General Use Debug LEDs	These LEDs will function as general use debug LEDs.	These pins were chosen to function as LEDs because each is fully capable of toggling a digital signal to output to an LED.
Vmax: 3.4 Volts	3.4 Volts was selected because the pins will be driven by a 3.3 volt supply from the RPi.	The RPi will be powering these GPIO pins using a 3.3 volt supply, so the pins should be incapable of exceeding this voltage without special hardware.
Vmin: 0 Volts	0 Volts was selected because the pins should never drop to negative voltages given they will be grounded via the RPi as well.	Because the RPi ground will be used, it should be impossible to incur a negative voltage on the GPIO pins.

Verification Process

The process for testing this block will consist of two parts. A test for the rsphrdwr_flrscncmbddd_data interface and a test for the flrscncmbddd_andrdcmmnctn_data interface. Each will consist of a testing function which will run each method to test it. The steps are as follows:

Running the rsphrdwr_flrscncmbddd_data test function.

1. (On hardware) Attach a voltmeter from Pin 6 to GROUND.
2. (In test function) Call GPIO.output(6, GPIO.HIGH)
3. (On hardware) Observe that voltmeter reads at least 2.3 volts.
4. (In test function) Call GPIO.output(6, GPIO.LOW)
5. (On hardware) Observe that the voltmeter reads at most 1 volt.
6. (In test function) Call GPIO.output(6, GPIO,HIGH)
7. (On hardware) Observe that the voltmeter reads at least 2.3 volts.
8. (In test function) Call GPIO.cleanup()
9. (On hardware) Observe that the voltmeter reads at most 1 volt.
10. (In test function) Call camera.capture('/home/pi/Desktop/testImage.png')
11. (In Raspbian) Observe that the picture is present.

Running the flrscncmbddd_andrdcmmnctn_data test function.

1. (In code) Call Broadcast(btAddr) method.
2. (In test function) Execute mimic method.
3. (In console) Observe that Broadcast function has been passed.
4. (In test function) Change the brightness of the camera settings.
5. (In test function) Execute a take picture command.
6. (In directory) Observe that a picture has been taken and that its brightness has changed.
7. (In code) Call SendImg(ImgData) method.
8. (In test function) Execute mimic method.
9. (In console) Observe that execution returns to self.
10. (In code) Call DisableBluetooth() method.
11. (In test function) Execute mimic method.
12. (In console) Observe that execution has completed and has not returned to self.

References and File Links

[1] *Picamera*. (v 1.13), Dave Jones. Accessed: Jan. 15. 2022. [Python Library]. Available: https://picamera.readthedocs.io/en/release-1.13/

[2] *RPi.GPIO.* (v 0.7.0), Ben Croston. Accessed: Jan. 15. 2022. [Python Library].Available: https://pypi.org/project/RPi.GPIO/

[3] *Pillow.* (v 9.0.0), Alex Clark et. al. Accessed: Jan. 15. 2022. [Python Library]. Available: https://pillow.readthedocs.io/en/stable/

4.3. Power Supply

Description

This design block consists of a power supply which will provide battery charging, battery discharging, and voltage regulation to the portable subsystem of the Portable Microscopic Fluorescence Detection Platform (PMFDP). This design block supports the "portable" requirement of our portable subsystem by allowing the system to operate without the constraint of a wall power supply.

Considered as a black box, this design block takes a regulated five volt DC power supply as input via a USB micro connector. It then both: 1) charges a lithium polymer battery with this power and 2) converts power to an acceptable range of output power (see interface definitions below). When outside power is not available, the power supply continues to power the system through energy stored in the battery. This design block will be mechanically secured to the portable subsystem's mechanical enclosure with two screws and requires that the enclosure leave access to the USB micro port for power input.

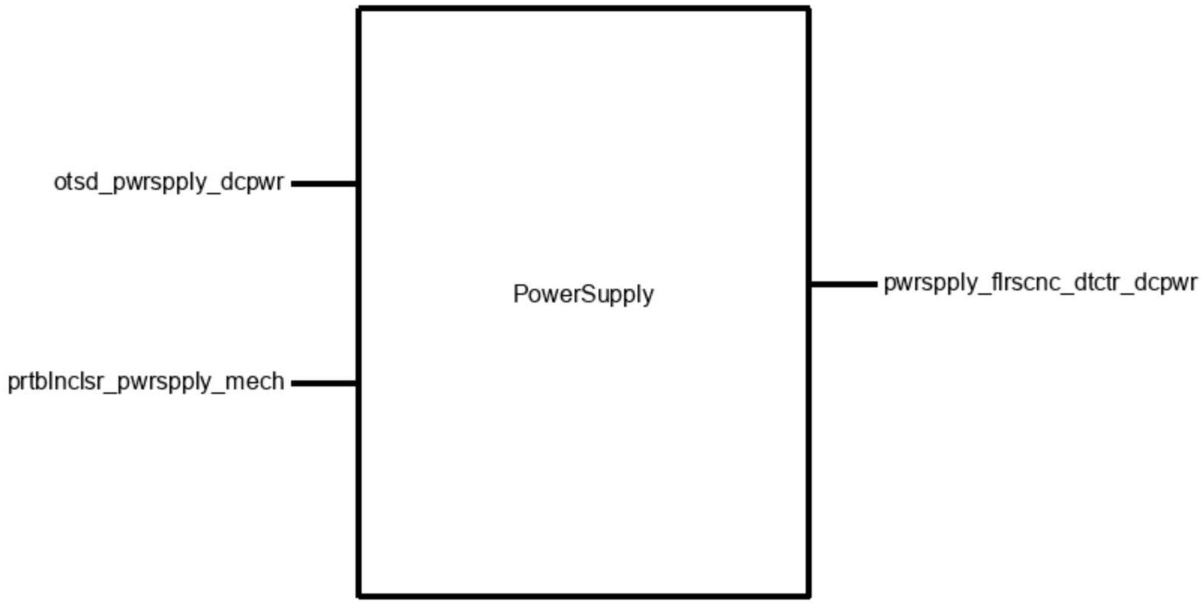


Figure 12: Power Supply Overview Block Diagram

Design

This power supply can be further broken down into the following four functional blocks:

1. Lithium Polymer Battery Charging subcircuit
2. Boost Conversion Subcircuit
3. Lithium Polymer Battery Selection
4. PCB Design

These internal blocks interact according to Figure 2. This configuration allows all devices powered by this block to operate either from a standard USB micro five volt power supply or from power stored in the battery.

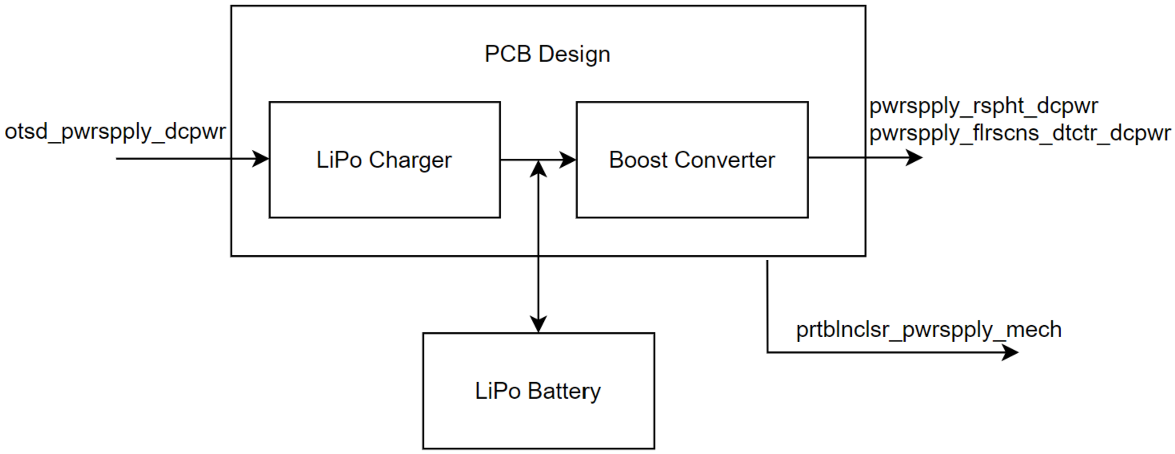


Figure 13: Power supply internal design blocks

General Validation

This block validation requires that the circuit provide a voltage of 5.0 (+0.3, -0.7) Volts when on, fully charging the LiPo battery, and that the circuit be capable of providing at least 2.4 Amps of current. These constraints were achieved by paying special attention to the design recommendations in the two datasheets linked in Section 6.

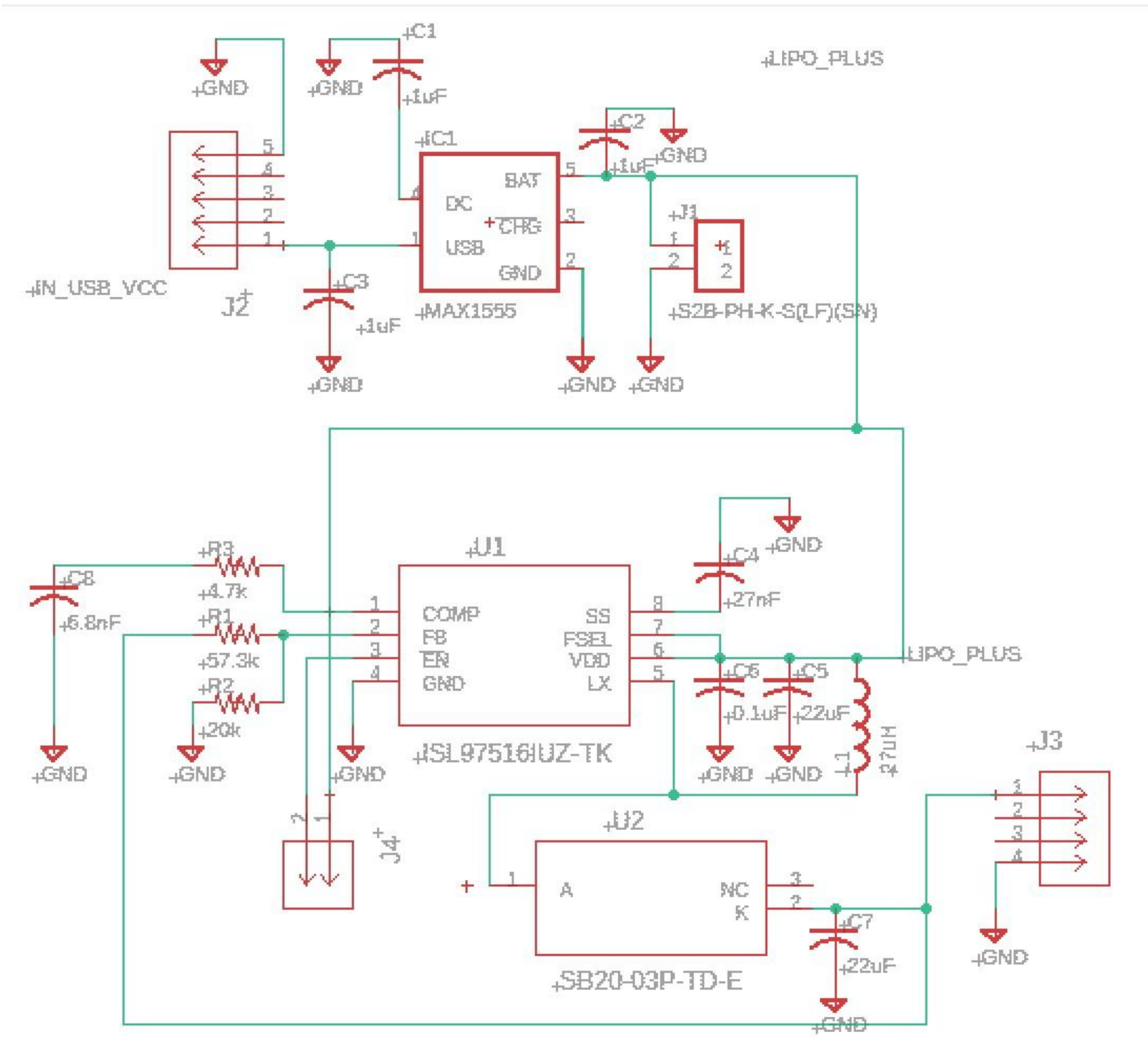


Figure 14: Electrical Schematic of the entire circuit

Charging Sub-Circuit Validation

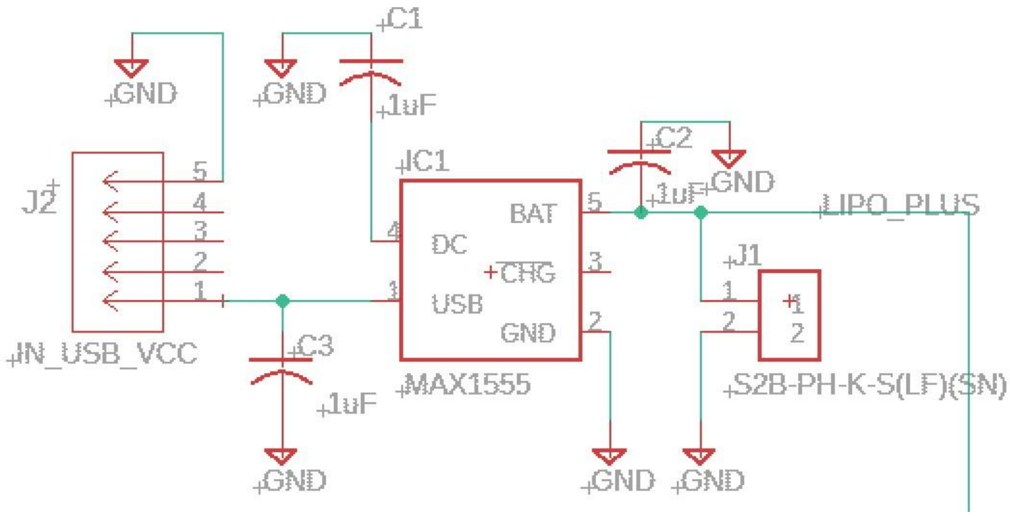


Figure 15: The Charging Sub-Circuit

The charging sub-circuit was designed around the MCP73811 integrated circuit (IC) represented by the MAX1555 schematic element in Figures 4 and 5. This IC Accepts an input voltage of 3.75-6.0 Vdc. The MCP73811 was also one of the only LiPo charging ICs capable of charging the three-celled battery chosen for the power supply.

Boost Converter Sub-Circuit Validation

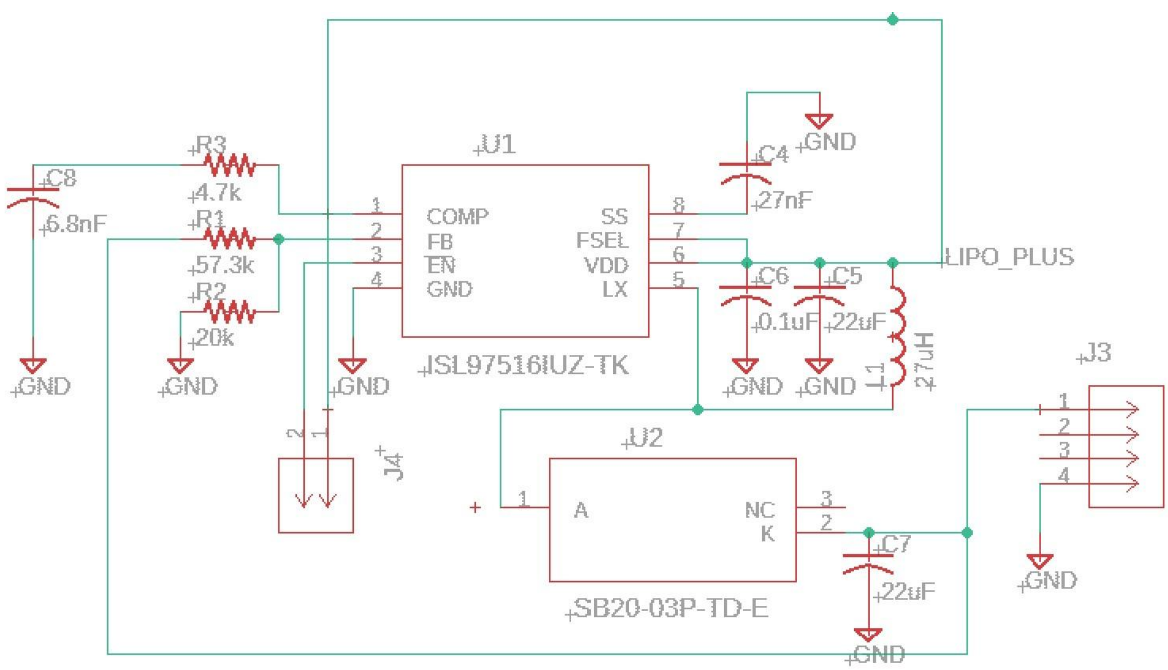


Figure 16: The Boost Converter Sub-Circuit

The boost converter sub-circuit steps up the battery to a final output voltage of 5.0 Volts. This circuit uses the equation given in Figure 6 to determine output voltage. In this application, $R_{top} = R_1 = 57.3\ k\,\Omega$ and $R_{bot} = R_2 = 20.0\ k\Omega$, yielding an output voltage of 5.001 Volts (See p.17 of the MCP1642 datasheet).

5.2 Adjustable Output Voltage Calculations

To calculate the resistor divider values for the MCP1642B/D, the following equation can be used. Where R_{TOP} is connected to V_{OUT}, R_{BOT} is connected to GND and both are connected to the V_{FB} input pin:

EQUATION 5-1:

$$R_{TOP} = R_{BOT} \times \left(\frac{V_{OUT}}{V_{FB}} - 1 \right)$$

Figure 17: Feedback resistance equations

Lithium Polymer Battery Functional Block Validation

The battery selected was chosen to provide adequate battery life for thirty images to be taken with as controlled by our central microcontroller- a Raspberry Pi Zero. These devices are known to be fairly power hungry with 0.4 Watts given as typical power consumption. An ICR18650 battery pack was chosen to provide a nominal 6600 mAh.

Mechanical Functional Block Validation

The mechanical functional block encloses the battery and physical circuit from outside damage and allows attachment points to secure this functional block to the

Interface Validation

The following interfaces are given with their validation methods below.

Interface: otsd_pwrspply_dcpwr

Property	Reason	Validation
Inominal: 1.0 A	This nominal current was chosen to slightly overshoot the needs of all systems in the portable enclosure subsystem.	• The MCP73811 draws a peak current of 450 mA, which bottlenecks all components in the system and will require a redesign if the system is to run while plugged in • The LiPo battery has a maximum discharge current of 3300 mA, equating to 2660 mA of effective output power (p.2)
Vmax: 6.0 V	This is the maximum input power that the MCP73811 is rated for with a margin of 1.0 Volts.	• The MCP73811 has an absolute maximum rating of 7.0 Volts, and characteristic maximum of 6.0 V in standard temperatures (p.3)
Vmin: 3.75 V	The MCP73811 lists this as a characteristic minimum voltage.	• The MCP73811 lists this as a characteristic minimum voltage (p.3)
Vnominal: 5.0 V	This is the nominal voltage for all USB micro chargers.	• See definition for "vSafe5V" on p. 283 of the USB Power Delivery Spec (Download here)

Interface: pwrspply_flrscns_dtctr_dcpwr		
Property	**Reason**	**Validation**
Ipeak: 400mA	This is the maximum current draw estimated to turn on our excitation LEDs.	• The MCP1642 lists a max current output of 1.8 A (p.4)
Vmax: 5.3 V	This is the same specification as given in the USB standard and is reasonable for limiting current to excitation LEDs.	• See the "Boost Converter Sub-Circuit Validation" Section above for a derivation of the 5.001 expected output voltage
Vmin: 4.3 V	This allows for some power drop in the system.	• See the "Boost Converter Sub-Circuit Validation" Section above for a derivation of the 5.001 expected output voltage

Interface: pwrspply_rspht_dcpwr		
Property	**Reason**	**Validation**

Inominal: 700 mA	This is the maximum power required by the Raspberry Pi Zero W with a margin for indicator lights (Imax=230mA according to this article).	• The MCP1642 lists a max current output of 1.8 A (p.4)
Vmax: 5.3 V	This is the same specification as given in the USB standard and is reasonable for limiting current to excitation LEDs.	• See the "Boost Converter Sub-Circuit Validation" Section above for a derivation of the 5.001 expected output voltage
Vmin: 4.3 V	This allows for some power drop in the system.	• See the "Boost Converter Sub-Circuit Validation" Section above for a derivation of the 5.001 expected output voltage

Interface: prtblnclsr_pwrspply_mech		
Property	**Reason**	**Validation**
Fasteners: 2mm thread-forming screws	These are standard screw sizes for small PCBs.	• Board manufacturer JLCPCB lists a hole size tolerance of +0.13/-0.08mm (See this link)
Screw hole spacing (on center): Top left and bottom right corners of a 21.59mm x 34.29 rectangle ±0.2mm	The PCB dictated the screw placing for this specification to fit with the rest of the board layout.	• Board manufacturer JLCPCB lists a CNC routing dimension tolerances of ±0.2 mm and hole size tolerance of +0.13/-0.08mm (See this link)
USB micro input cutout- See: document Drawing and Mechanical Interface Properties Reference 6.4, partially pictured in Figure 8	The USB micro connector chosen for the power supply sits flush with the edge of the board and requires a mechanical cutout to offer USB access.	• Board manufacturer JLCPCB lists a CNC routing dimension tolerances of ±0.2 mm and hole size tolerance of +0.13/-0.08mm (See this link) • The measurements given in figure 8 were generated from the largest USB micro that the developer had access to, leaving extra margin for error

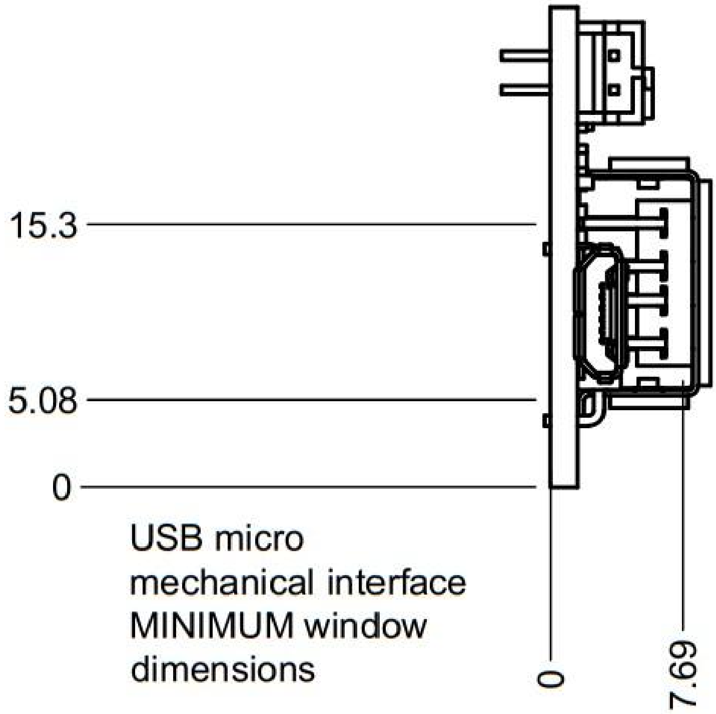


Figure 18: USB micro window spacing

Verification Process

This block will be tested by the three methods detailed below.

Voltage Meter Methods

A voltage meter will test the output voltage of the block by:

1. Making an electrical contact with nodes J3_1 and J3_4 (See Figure 4)
2. The voltage meter will be set to measure voltage
3. Output voltage will be considered sufficient if it does not leave the specified range of 4.3-5.3 Vdc.

Virtual Load Methods

1. A virtual load will be connected to nodes J3_1 and J3_4 to measure output power.
2. The load will be set to provide a resistance corresponding to a constant current of 1.1 Amps. Average voltage will be measured.
3. Output voltage will be considered sufficient if it does not leave the specified range of 4.3-5.3 Vdc. This will also verify output current.

Mechanical Inspection Methods

1. The mechanical enclosure will be physically inspected using mechanical calipers and a square.
2. Bolt hole spacing must represent the top left and bottom right corners of a 21.59mm x 34.29 rectangle ±0.2mm.
3. The PCB design will be considered sufficient if hole spacing is within the specifications given in (2)

References and File Links

- MCP1642B (Boost Converter) datasheet: https://ww1.microchip.com/downloads/en/DeviceDoc/20005253A.pdf
- MCP73811 (LiPo Charger) datasheet: https://ww1.microchip.com/downloads/en/DeviceDoc/22036b.pdf
- ICR18650 (battery) datasheet: https://cdn-shop.adafruit.com/product-files/353/C450_-_ICR18650_6600mAh_3.7V_20140729.pdf
- Drawing and mechanical interface properties: https://drive.google.com/file/d/1ZeoxY8CpSKaK5-gObWN2B-NZmWjVKZbt/view?usp=sharing

4.4. Portable Enclosure

Description

The portable enclosure is designed to store all electronic components and provide a convenient way to transport and store portable micro fluorescence detector.

Design

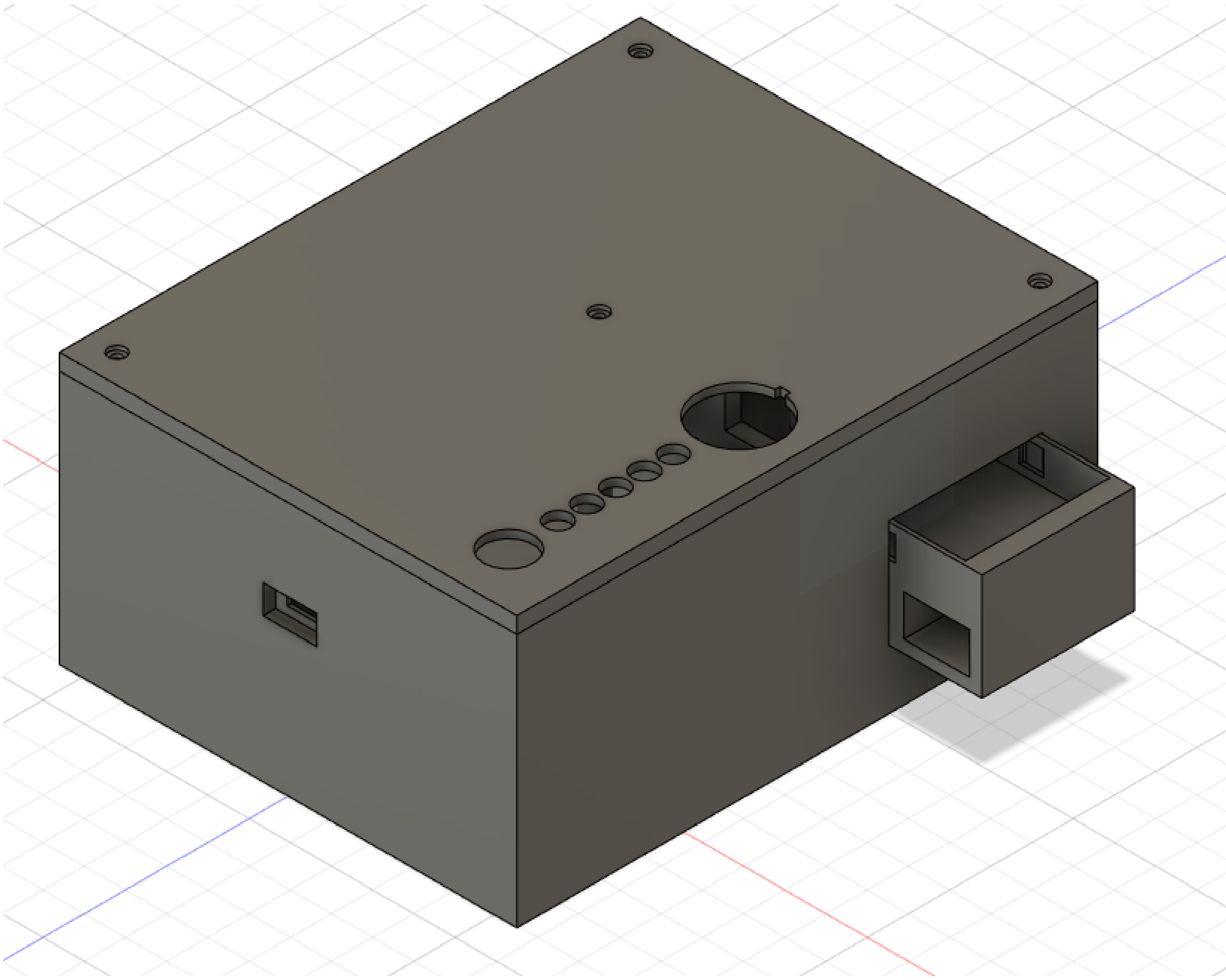


Figure 19: Assembled Portable Enclosure

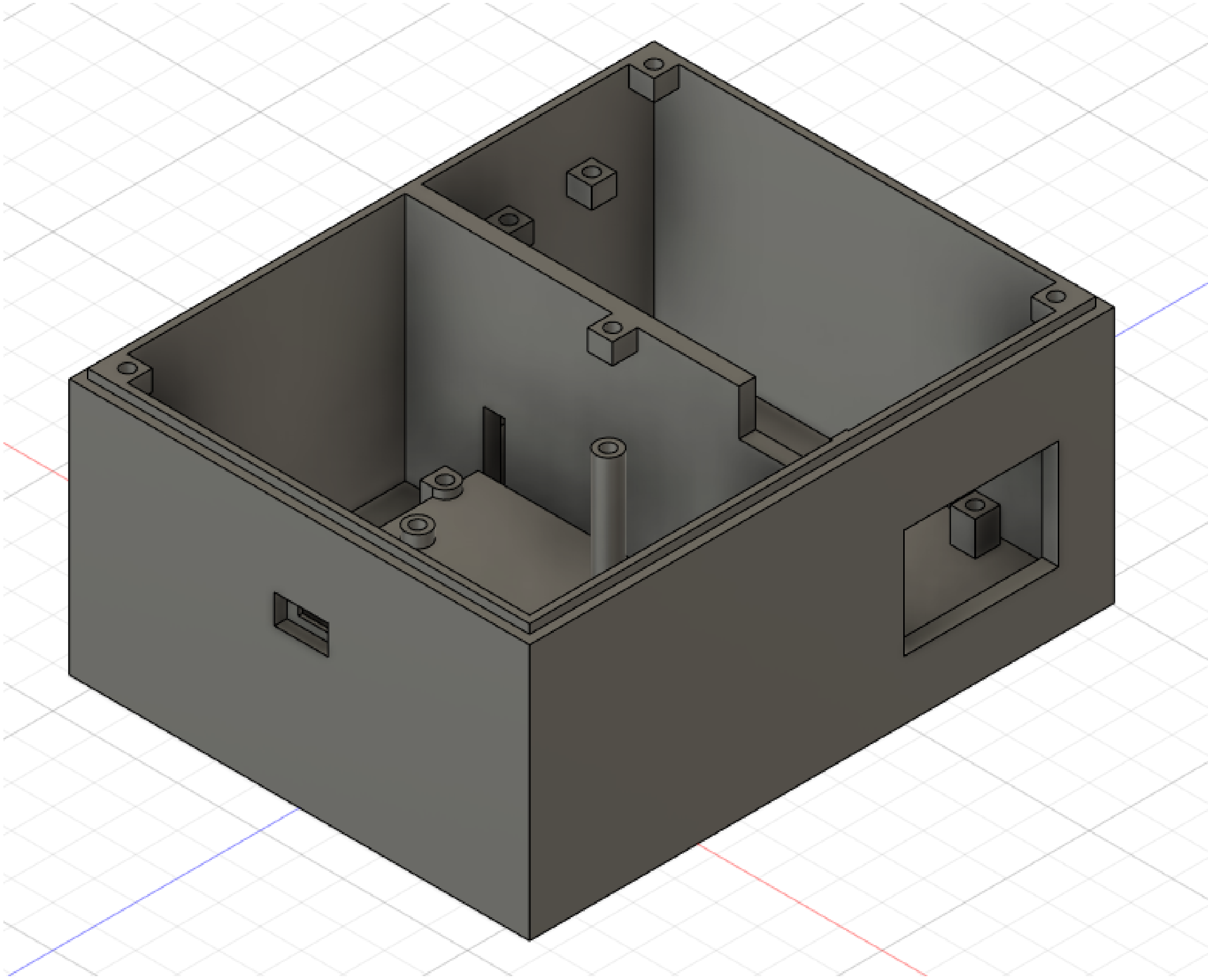


Figure 20: Enclosure

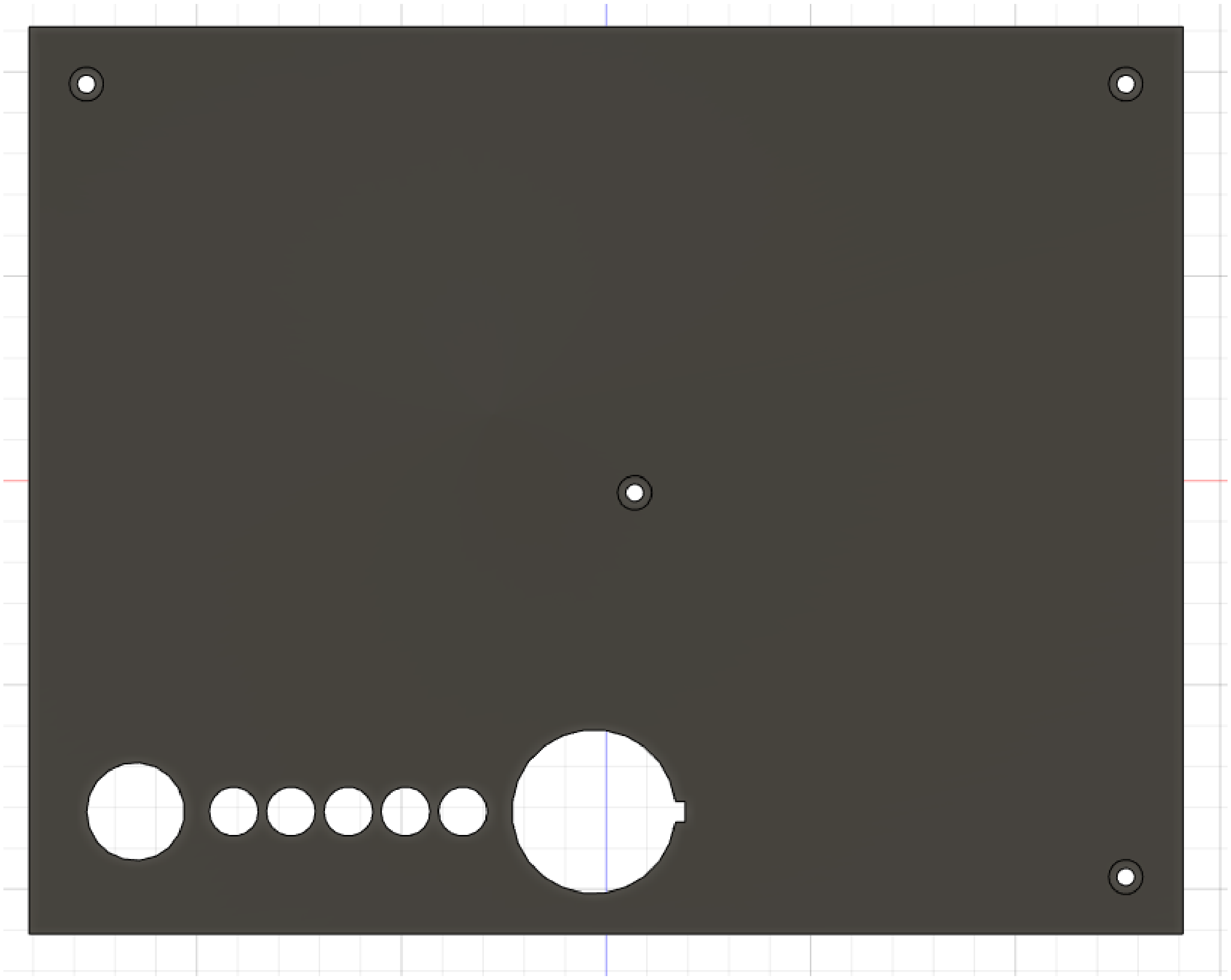


Figure 21: Cover

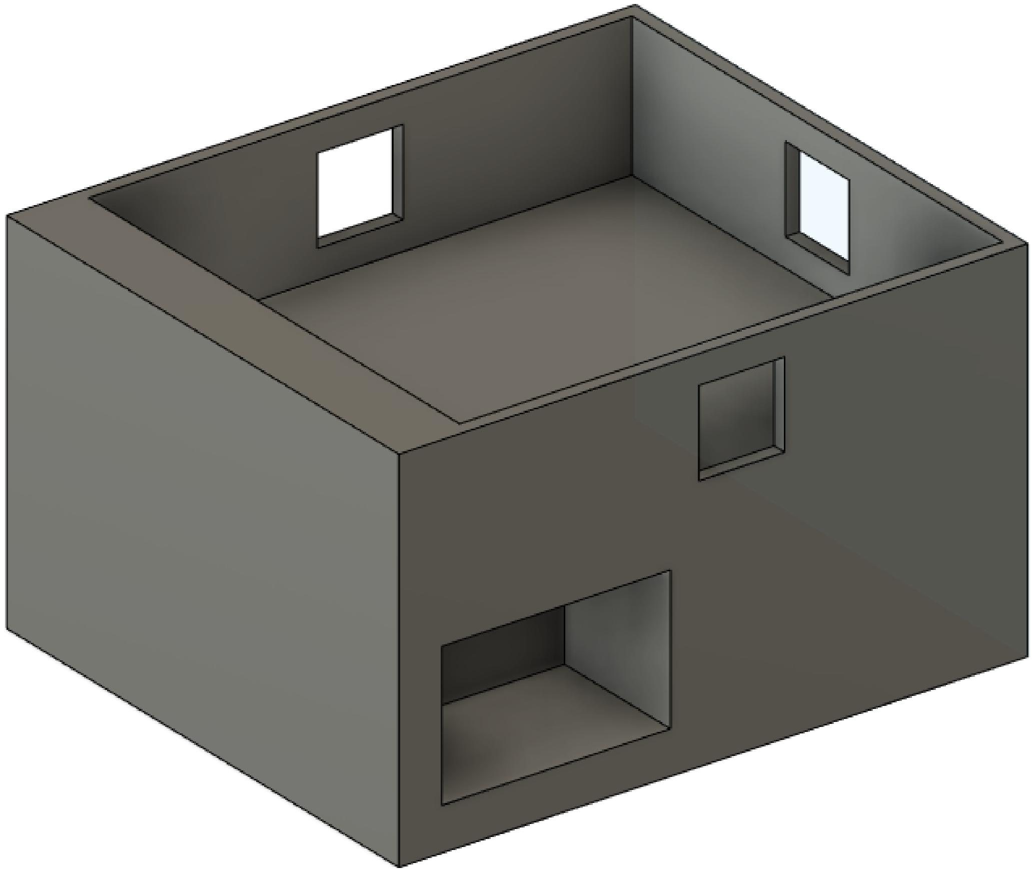


Figure 22: Sample holder

General Validation

Portable enclosure was designed to house electronic components and fluorescence detector. Sample holder was designed to include camera and excitation LEDs to analyze the sample pleased in the sample holder. The cover was designed for users to have easy access to debugging LEDs, and hardware on/off switch. The cover is detachable, so a user can have access to the electronic components stored in the portable enclosure. The battery is stored in a separate section of the enclosure and has a detachable cover as well, so a user can change the battery in case it is needed.

Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
INTERFACE: prtblnclsr_pwrspply_mech : Output		
Fasteners: 2mm thread-forming screws	Power supply screw hole dimensions	Measurement are correct since power supply PCB was successfully attached to the enclosure
INTERFACE: prtblnclsr_flrscnc_dtctr_mech : Output		
Fasteners: Screw spacing: diagonals of a 1.25"x1.25" square (on center)	Detector dimensions	The detector was successfully attached to the enclosure meaning the design is correct

Verification Process

Step 1: Visually inspect the enclosure for loose/unsecured components.
Step 2: Note the positions of specified enclosure components (see list below).
Step 3: Rotate the enclosure 180˚ to be upside down, then another 180˚ to return to its original position
Step 4: Check all sub-components for movement. This verification will fail if any sub-components move more than 2mm.

List:

- Power supply PCB
- Debugging LEDs PCB
- Raspberry Pi Zero
- Camera
- Excitation LEDs
- Battery

References and File Links

Portable enclosure design files:
https://drive.google.com/drive/folders/1j5efrWwN0sy_g5Wa8R-DBMXaBwrgaquo?usp=sharing

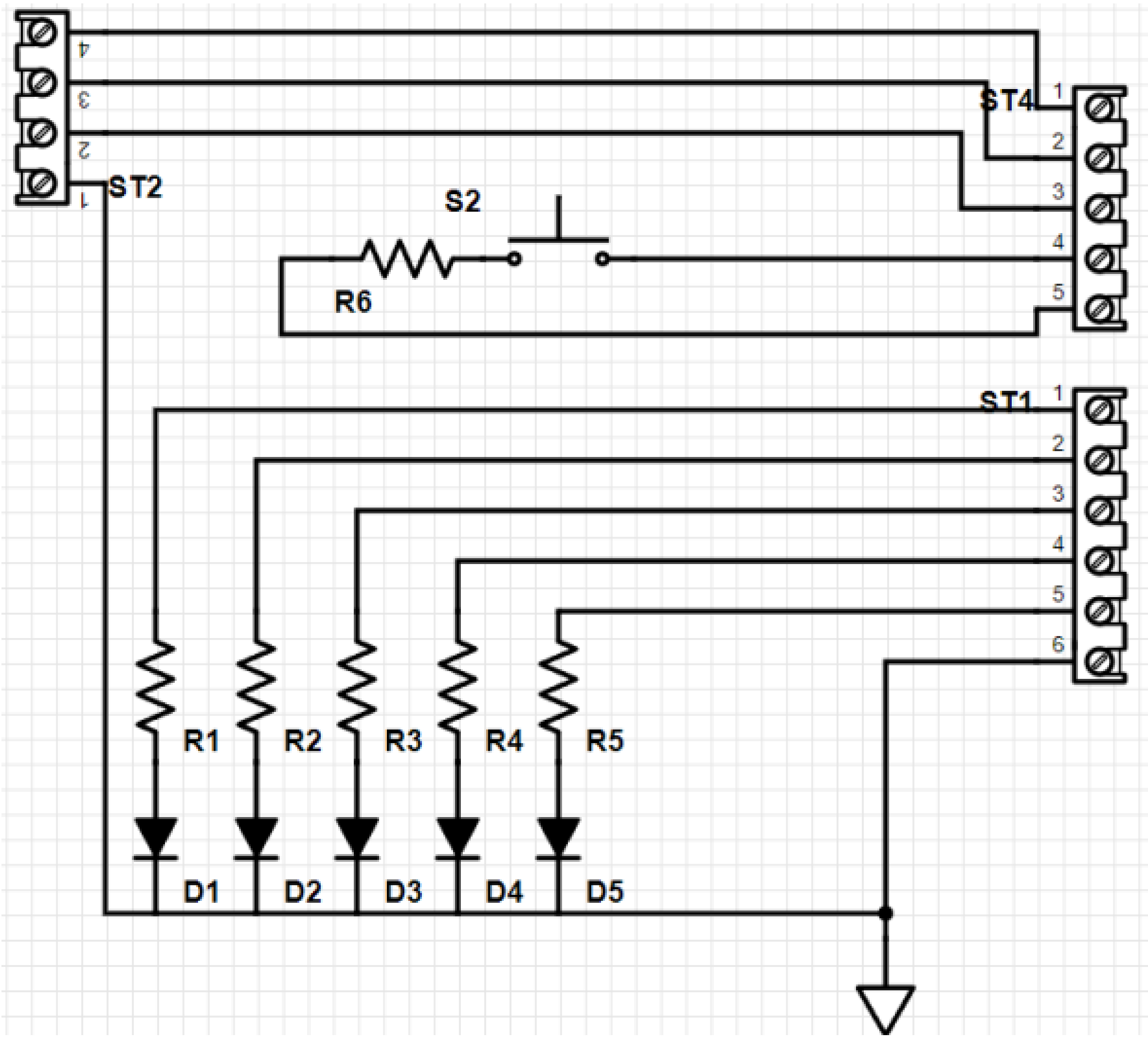
4.5. RasPi Hat

Description

This block controls all sensing and communication inputs and outputs of the system as well as provides connection status of modules through debugging LEDs. For power efficiency the debugging LEDs output is controlled through the user pushbutton input.

Design

Debugging LEDs (D1-D5): debugging LEDs show the connection state of different modules like camera, Bluetooth connection, and Raspberry Pi initialization status. Each LED receives digital signal from andrdcmmnctn_rspht_dsig from corresponding Raspberry Pi GPIO to turn on LED and show modules' states described above representing rspht_otsd_usrout. Green LEDs will be used as a more intuitive color for an on state of a module. Raspberry Pi GPIO provide 3.3V output with 16mA current as described here. Greed LED has 2V forward voltage, therefore the value of R1-R5 resistors is 81 Ohm, however for simplicity 100 Ohm resistors will be used. Switch (S2): Switch is a pushbutton and it is a user input otsd_rspht_usrin. When user wants to know the state of modules, they push the button. The switch completes the circuit allowing current to pass through and rspht_andrdcmmnctn_dsig receives signal through Raspberry Pi GPIO to turn on debugging LEDs. For safety reasons 1K Ohm current limiting resistor R6 will be used. Pins ST2: Pins receive digital signal from Android communication block to turn on LEDs in rspht_flrscnc_dtctr_dsig pwrspply_rspht_dcpwr interface is not shown on schematics because power supply will be directly connected to Raspberry Pi, thus any intermediate connections are not necessary.



General Validation

Debugging LEDs is an intuitive way to represent information like a state of a particular module and will be controlled by Android communication block. For example, an LED will show the state of Bluetooth connection. When Bluetooth connection is established with smartphone and pushbutton is pressed the LED will turn on. If pushbutton is pressed and Bluetooth connection is not established the LED will remain turned off. This type of interaction will help to debug the system and more importantly specify what the problem is. LEDs are cheap (~$0.7 per LED), widely available, and easy to replace. Pushbutton is an intuitive input. When the button is pressed, debugging LEDs will turn on. Pushbutton is used for power efficiency. It is better to turn on LEDs when needed rather than keeping them turned on all time. This is especially important since the fluorescence detector is portable and operates on rechargeable battery. Pushbutton is cheap (~$0.25 per switch), widely available, and easy to replace. Pins ST2 are connected to fluorescence detector LEDs. While they can be directly connected to Raspberry Pi, it is more convenient to have all-in all-out interface communication. For example, if a fluorescence detector needs to be replaces, it is easier to disconnect it from dedicated pins on PCB rather than looking for them on Raspberry Pi, which has other connections.

Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
INTERFACE: otsd_rspht_usrin : Input		
Type: Push button Turn on	Type: Push button Turn on LEDs when needed rather than keeping them constantly on	Block is designed to display state of modules through debugging LEDs
INTERFACE: pwrspply_rspht_dcpwr : Input		
Inominal: 2.0 A	Recommended supply current of Raspberry Pi	Documentation suggest to use this value
Vmax: 5.3 V	Max supply voltage of Raspberry Pi	Documentation suggest to use this value
Vmin: 4.3 V	Minimum supply voltage of Raspberry Pi	Documentation suggest to use this value
INTERFACE: andrdcmmnctn_rspht_dsig : Input		
Vmin: 0V	If we need to keep LEDs off the output voltage is 0	Documentation suggest to use this value
Vmax: 3.3V	Raspberry Pi GPIO use 3.3V logic	Documentation suggest to use this value
Logic-Level: Active	Based on the properties of	Raspberry Pi GPIO can be

high //send signal to turn LEDs on	Raspberry Pi GPIO	configured to use high/low logic
INTERFACE: rspht_otsd_usrout : Output		
Other: Debugging LEDs display connection status	Debugging LEDs are designed to display the state of modules	Block is designed to display state of modules through debugging LEDs
Type: LEDs	LED is a convenient way to represent an on/off state of module	Block is designed to display state of modules through debugging LEDs
INTERFACE: rspht_flrscnc_dtctr_dsig : Output		
Logic-Level: Active high // send signal to turn LEDs on	Based on the properties of Raspberry Pi GPIO	Raspberry Pi GPIO can be configured to use high/low logic
Vmax: 3.3V	Raspberry Pi GPIO use 3.3V logic	Documentation suggest to use this value
Vmin: 0V	If we need to keep LEDs off the output voltage is 0	Documentation suggest to use this value

Verification Process

Pushbutton test

1. Attach voltmeter across pushbutton resistor R6.
2. Press pushbutton and read voltage value to be 3.3V
3. Observe Raspberry Pi reads HIGH input on connected GPIO

Debugging LEDs test

1. Connect LEDs to Raspberry PI GPIO
2. Attach voltmeter across LED resistor

3. Send HIGH signal from Raspberry Pi (Android communication) from corresponding GPIO
4. Read voltage across each resistor to be 3.3V
5. Observe LEDs turn on

References and File Links

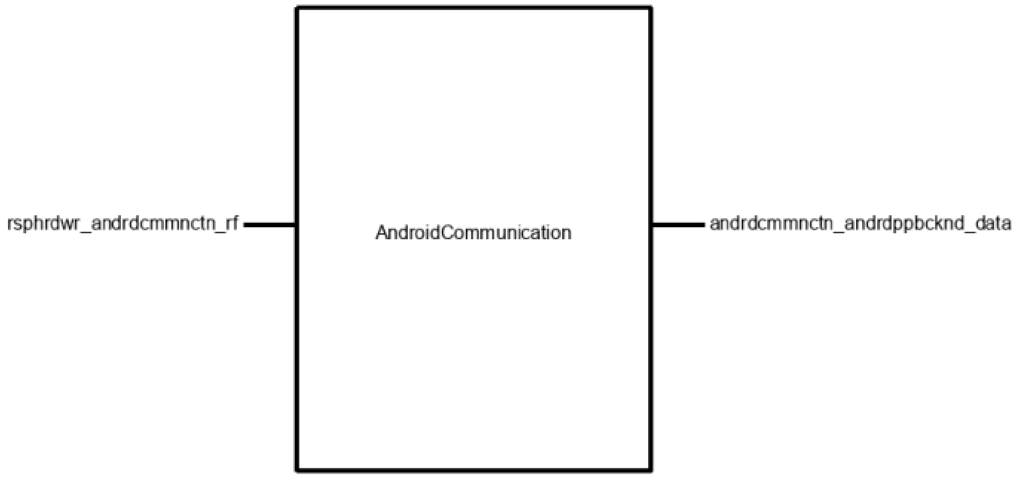
"Raspberry Pi Documentation", raspberrypi.com. [Online]. Available: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html [Accessed May. 6, 2022]

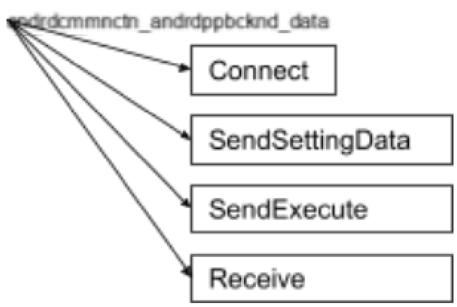
4.6. Android Communication

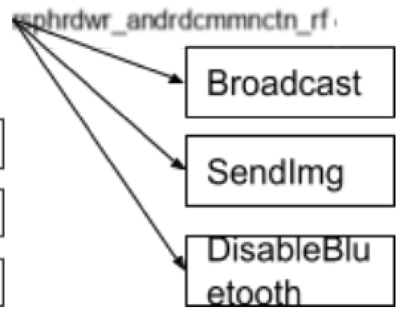
Description

The block, Android Communication, provides bluetooth connectivity between the raspberry pi, and the mobile device. It achieves this by making the raspberry pi create a bluetooth signal that the mobile device can connect to. It then facilitates the communication of various signals needed for the system to function, such as image data, setting data, as well as execution commands to raspberry pi to execute. This is a software data carrying and communication block, rather than any kind of control block.

Design







General Validation

This design fits the needs of the system purely as the communication block between the android and the raspberry pi. All commands, and data would be self-contained, with the block purely caring about communication of the data, rather than what the data contains, or what the data would do later in the system.

Because this block is planning on using built-in bluetooth, and is all software, it will not be anymore costly to the project. One advantage of being a software based block, is that it can change, and if needed, can be changed in accordance with other hardware based blocks that couldn't be changed as easily. The main concern is the communication accuracy, bandwidth, error detection, transmission range, as well as the time it takes to develop the software, as it often takes longer than predicted. The project partner requires a certain range, and if the built-in bluetooth devices don't work properly, then external bluetooth chips might need to be added.

That being said, this block facilitates communication between a mobile device and a raspberry pi. We need to communicatie various settings from the mobile device to the raspberry pi, and various images from the raspberry pi to the mobile device. Using python allows the software developer to use prebuilt libraries to use bluetooth, and bluetooth being often used allows easier debugging, and overall faster/more efficient development

Interface Validation

Interface Property	Why is this interface this value?	Why do you know that your design details for this block above meet or exceed each property?
INTERFACE: rsphrdwr_andrdcmmnctn_rf (Raspberry Side)		
Broadcast(bluetooth _address)	Why is this interface this value?	Due to us needing to use bluetooth communication, we need to create a signal on the raspberry pi for the mobile device to connect to
SendImg(ImgData)	We need send an image to the phone	Due to us needing to be able to send images to the mobile device, we need a function that will be able to communicate a given image to the mobile device
DisableBluetooth()	We need to be able to stop communications, if the system no longer needs it	Due to us eventually needing to shut the device down, it will be better to shut the device down cleanly, rather in a way that could cause memory problems
INTERFACE: andrdcmmnctn_andrdppbcknd_data (mobile device side)		
Connect(bluetooth_a ddress)	We need to be able to connects to bluetooth address signal given off by the raspberry pi	Due to us needing to use bluetooth communication, we need to be able to connect the mobile device to the raspberry pi
SendSettingData(Set tingDataStruct data)	We need to be able to connects to bluetooth address signal given off by the raspberry pi	Due to us needing to be able to send various settings to the raspberry pi, we need a function that will be

		able to communicate a given image to the mobile device
SendExecute()	We need to tell the raspberry block to do something (mainly take a picture)	Due to us needing to take a picture of a slide according to various settings, we need a function to tell another block to do that
ImgData Receive()	We need to receive an image back from the raspberry pi block	Due to us needing to display the image, and see what the slide looked like after an image, we need a function to grab that data

Verification Process

For this, each function can be tested by either an external device, or by first testing one block, then using that as a testing interface.

1. Broadcast(bluetooth_address)
    a. What this needs to do is create a bluetooth signal that is viewable by a mobile
    Device
    b. To test this, simply use a phone to look at available bluetooth devices, and
    determine the address
        i. If there is a device that matches both the address, and the device, then
        this function succeeds passing the test
2. SendImg(ImgData)
    a. This function needs to send image over to the phone, via bluetooth
    b. To test this, you can simply send a very small image over, and analyze the traffic with wireshark to see if the data matches
        i. If the data matches, then you can send multiple tests until satisfied image
        is being sent correctly
3. DisableBluetooth()
    a. This function just needs to stop bluetooth broadcasting/receiving

b. To test this, you can simply use a phone to look at available bluetooth devices,
and then determine if the network is no longer there

4. Connect(bluetooth_address)
    a. This function just needs to connect to the raspberry pi device
    b. To test this, you can either use wireshark to analyze traffic, or write a test program (on the raspberry pi side) that outputs everything it gets as a string, and if anything gets sent, you know a connection has been made
5. SendExecute()
    a. This function just needs to send something other than a connection signal to the raspberry pi device
    b. To test this, you can either use wireshark to analyze traffic, or write a test program (on the raspberry pi side) that outputs everything it gets as a string, and
    if anything gets sent, you know a connection has been made
6. ImgData Receive()
    a. This function just needs to receive something other than a connection signal or to the raspberry pi device
    b. To test this, you can simply send a very small image over, and analyze the traffic
    with wireshark to see if the data matches
        i. If the data matches, then you can send multiple tests until satisfied image is being sent correctly
    c. To test that you are indeed receiving it properly, you can just try to display the image, as it won't be the image you took, it doesn't work, and if it is the one you took, then it does work.

References and File Links

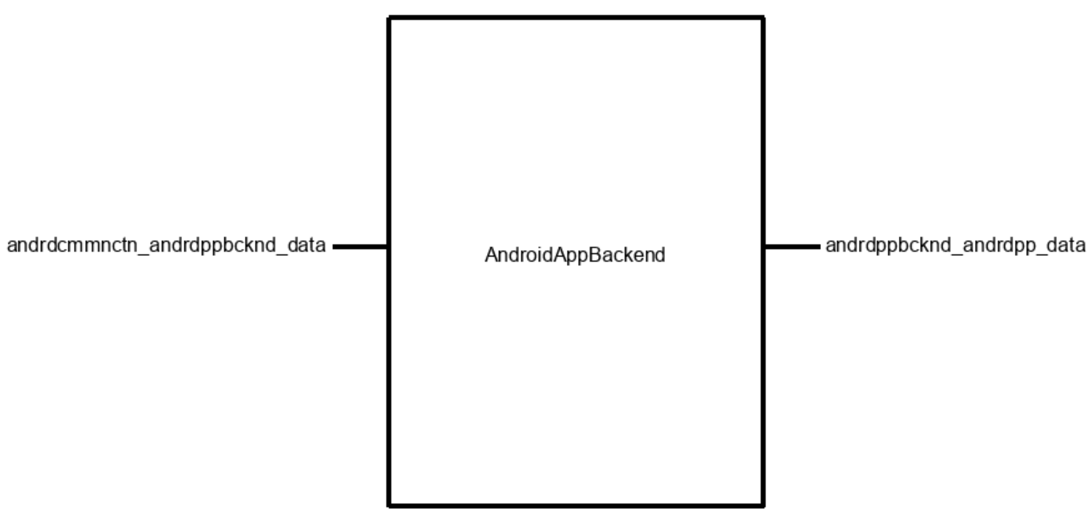
https://pyserial.readthedocs.io/en/latest/pyserial_api.html
https://towardsdatascience.com/sending-data-from-a-raspberry-pi-sensor-unit-over-serial-bluetooth-f9063f3447af?gi=f97fb285af34
https://wiki.wireshark.org/Bluetooth

4.7. Android App Backend

Description

This block fits into a larger project which will analyze a biological sample. It will do so on a mobile platform that can be transported easily. Due to its design nature, the most intuitive method of controlling the platform is with a mobile phone application. Mobile phone applications are generally built with two distinct parts. There is the style code which controls setting up the GUI with elements like buttons and images usually written in HTML, and then there is the functional code which handles the programmatic logic during the applications runtime. For example the HTML code might display a button, but without the event handler (functional code) the button won't actually do anything. This block encompasses the functional code for the Android Application; this code will be written largely in Kotlin, as it will be for Android. It's function in the system will be to handle the actions the user wishes to execute, which will be conveyed between the user and this block via the Android Application UI block. This functional code block will be responsible mostly as a control for communication to the embedded program, but will only interface with the physical Pi and detection blocks indirectly through the Android Communication block.

Design



The above block diagram illustrates that the Android Application Backend Code will interface with the Android Application UI block through elements of the UI linked with their event handlers in this block. It will also interface with the Android Communication Block through function calls to send commands and receive data to and from the embedded device. Pseudocode for the event handlers can be seen below.

Pseudocode:

Setting sliders, switches, and dropdowns are straightforward and just adjust a global variable based on the event.

```
main()
{
      Create handles to each element of the GUI;
      Create default settings struct;
      Set event handlers to each interactable element
of the GUI;
      Connect(bluetooth_address);
}

uploadSettingsEvent()
{
      pull settings struct;
      Verify settings struct for invalid values;
      SendSettingsData(struct settingsData);
}

takePictureEvent()
{
      uploadSettingsEvent();
      SendExecute("take pic");
      Receive();
}

rgbAnalysisEvent()
{
      uploadSettingsEvent();
      SendExecute("rgb");
      Receive();
}
```

The exact commands and settings list is too exhaustive to be listed here in this document, but includes picture settings like brightness and exposure time along with commands like quit and rgb. The full list of commands is linked below.

[Comprehensive Commands List]

General Validation

This block fits the design needs of the system. It will not incur material costs or require parts aside from an Android mobile phone for use testing as it is composed entirely of code. It will require an average amount of engineering effort because, while I am familiar with Kotlin and Android programming, I am not an expert and due to the language, it will not be as quick to develop as a language like Python. This will be a compiled program, so its execution speed should be more than adequate. This code will be runnable on any modern Android phone, and will be compatible with most older Android phones within reason. This block's interaction will be done through software methods including global variables, function calls, and return values.

Interface Validation

andrdcmmnctn_andrdppbcknd_data: Output

Connect to Pi: `Connect(bluetooth_adres);`	This method of a function call was chosen because it only needs to be called once, at the beginning of the program when it initially connects to the Raspberry Pi. A function call is the easiest way to do so.	This will work because Kotlin is fully capable of executing function calls as needed. It supports simple variable passing, so that will not be an issue either. The return type will be a boolean to represent if an issue connecting was encountered or if the connection was accepted, which is the most intuitive method of handling success/failure.
Grab Image Data: `Receive();`	This method of a function call was chosen because it will need to be called at multiple points when grabbing received images from the queue. No	This will work because Kotlin is fully capable of executing function calls as needed. The return type will be the image(s) received to be

	parameters will be necessary as the function itself will simply dump the bluetooth queue. It's return value will be the images received.	saved as needed by this block to the phone's permanent storage.
Send a Command: `SendExecute(cmd);`	This method of a function call was chosen because it will need to be called at multiple points when attempting to communicate with the Pi. A parameter will be used, the command to be sent, in order to make the function general use. This allows for only one function instead of one for each command that can possibly be sent.	This will work because Kotlin is fully capable of executing function calls as needed. The return type will be a boolean representing if the transmission was successful. This will not be an ACK confirmation, simply that the command was transmitted. The boolean return value will be optimal as it is the best way to error handle success/failure.
Update Settings: `SendSettingData(settingsData);`	This method of a function call was chosen to handle updating all Pi settings upon user submission. A parameter will be used in order to make the function general use, so that a separate function is not required to update each and every individual setting. The return type will be a boolean.	This will work because Kotlin is fully capable of executing function calls as needed. The return type will be a boolean representing if the transmission was successful. This will not be an ACK confirmation, simply that the command was transmitted. The boolean return value will be optimal as it is the best way to error handle success/failure.

andrdppbcknd_andrdpp_data: Bidirectional

Take Picture Button:	This method of a	This will work because

`takePictureEvent();`	function call was chosen because it will need to be called every time the UI communicates that the user wishes to take a picture of the sample. No parameter is necessary. A boolean type return variable will indicate if the command was sent.	Kotlin is fully capable of executing function calls as needed. The return type of boolean will be acceptable because it is the best way to error handle success/failure. This will not be an ACK confirmation, just a transmission confirmation.
Settings Sliders: `Event linked variables`	An event linked variable was chosen because no functionality will be needed until the user decides to submit the changed settings to the Pi, done via the Update Settings interface property. These variables will be all integers because they will represent numerical values.	This will be adequate because the granularity of a float will be unnecessary and no functional code needs be executed each time a slider is adjusted. Simply a variable will need to be updated.
Settings Dropdowns: `Event linked variables`	An event linked variable was chosen because no functionality will be needed until the user decides to submit the changed settings to the Pi, done via the Update Settings interface property. These variables will be of an enumeration because they will be selected via a dropdown menu.	This will be optimal as for these settings, only certain selections are valid, which will be listed on the dropdown. There will be no need for granular control like with an integer variable. No functional code need be executed upon interaction via this property, only a value updated.
LED Switches: `Event linked toggle functions`	Event linked toggle functions were chosen as a method because upon switching an LED on or off, the command corresponding to it will need to be called in the	This will be optimal as this allows immediate changing of the LEDs' status without the need for a call to the Update Settings interface property. Kotlin is fully

	event linked toggle function to update the LED status immediately.	capable of handling these event linked function calls.
Analyze RGB Spectrum: `rgbAnalysisEvent();`	This method of a function call was chosen because it will need to be called every time the UI communicates that the user wishes to execute an RGB analysis on the sample. No parameter is necessary. A boolean type return variable will indicate if the command was sent.	This will work because Kotlin is fully capable of executing function calls as needed. The return type of boolean will be acceptable because it is the best way to error handle success/failure. This will not be an ACK confirmation, just a transmission confirmation.

Verification Process

The process for testing this block will consist of two parts. A test for the andrdcmmnctn_andrdppbcknd_data interface and a test for the andrdppbcknd_andrdpp_data interface.

Testing andrdppbcknd_andrdpp_data:

1. (On UI) Click button element.
2. (In Code) Ensure that it enters the correct event handler using debug output.
3. (Iteration) Repeat for each button element.
4. (On UI) Change slider variable.
5. (In Code) Ensure that the slider variable has been updated accordingly using debug output.
6. (Iteration) Repeat for each slider element.
7. (On UI) Select from a dropdown menu element.
8. (In Code) Ensure that the enumeration variable has been updated accordingly using debug output.
9. (Iteration) Repeat for each dropdown element.
10. (On UI) Flip a switch element.
11. (In Code) Ensure that it enters the correct event handler using debug output.
12. (Iteration) Repeat for each switch element.

Testing andrdcmmnctn_andrdppbcknd_data:

1.	(In code)	Call Connect(bluetooth_address) method.
2.	(In test function)	Execute mimic method.
3.	(In console)	Observe that the Connect function has been passed and program execution continues.
4.	(In test function)	Call Receive() method.
5.	(In Code)	Execute mimic method.
6.	(In Console)	Observe that the image returned by mimic method is saved.
7.	(In Code)	Call SendCmd() method.
8.	(In Code)	Execute mimic method.
9.	(In Console)	Observe that the function has been passed and program execution continues.
10.	(In Code)	Call updateSettings() method.
11.	(In Code)	Execute mimic method.
12.	(In Console)	Observe that the function has been passed and program execution continues.

References and File Links

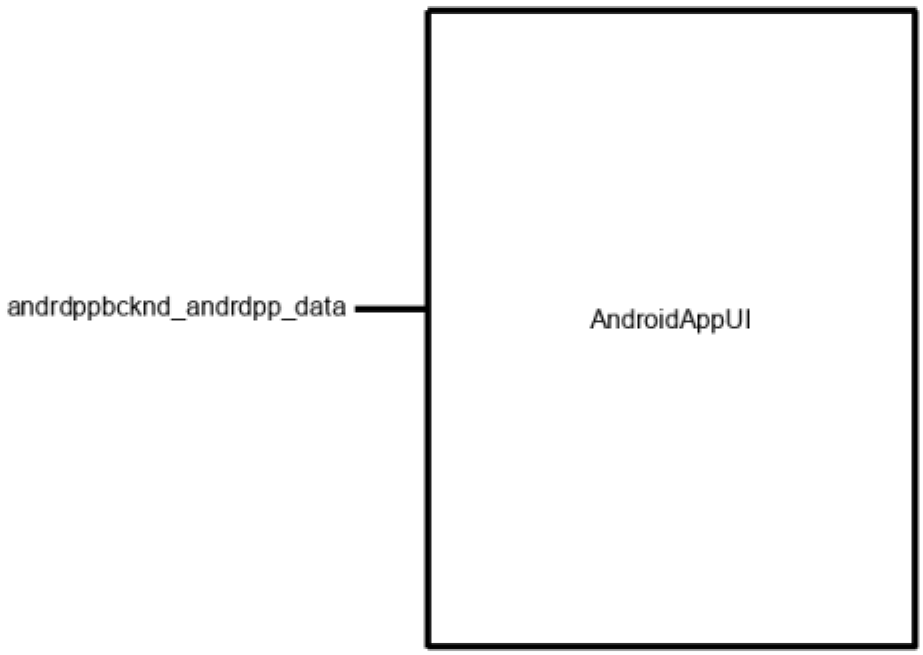
N/A

4.8. Android App UI

Description

The block, Android App UI, provides the user (on the phone) to interact and give commands to
the PI. It achieves this by interacting with the Android App Backend block, changing values, and
ultimately facilitating the user calling commands in a layout that makes sense (to people). This
is a control block, letting the user directly manage what the system overall is doing, as well as
how to do it.

Design



andrdppbcknd_andrdpp_data
- takePicture();
- slider variables;
- dropdown variables;
- switch variables;
- rgbAnalysis();

General Validation

This design fits the needs of the system purely as the main control block between the android and the raspberry pi. All commands, and data would be self-contained, with this block purely providing the user options in giving commands to the Backend Android block. Because this block is planned on Kotlin, and is all software, it will not be any more costly to the project. One advantage of being a software based block, is that it can change, and if needed, can be changed in accordance with other hardware based blocks that couldn't be changed as easily. The main concern is the time it takes to develop the software, as it often takes longer than predicted. The project partner requires certain time restraints in retrieving stored data.

That being said, this block facilitates communication between the user and most of the system. We need to communicatie various settings from the mobile device to the raspberry pi, and various images from the raspberry pi to the mobile device. Using Kotlin allows the software developer to use prebuilt libraries to call backend code, making a clean transition between the blocks, and faster development time.

Interface Validation

andrdppbcknd_andrdpp_data : Input

Other: rgbAnalysis();	We need to have a function that analyzes the image we receive, and therefore need the user to be able to CALL that function	My design details exceed the property, because technically, all this block needs to do is provide a button to call rgbAnalysis on the backend
Other: dropdown variables;	We need to be able to have various options for image taking, as well as analysis	My design details exceed the image taking, due to the amount of options these dropdown variables provide
Other: takePicture();	We need to have a function that tells the PI to take a picture, and therefore need the user to be able to CALL that function	My design details exceed the property, because technically, all this block needs to do is provide a button to call takePicture on the backend
Other: switch variables;	We need to be able to have various options for image taking, as well as analysis	My design details exceed the image taking, due to the amount of options these switch variables provide
Other: slider variables;	We need to be able to have various options for image taking, as well as analysis	My design details exceed the image taking, due to the amount of options these slider variables provide

Verification Process

For this, each function can be tested by either an external device, or by first testing one block,then using that as a testing interface

1. rgbAnalysis();
    a. All this needs to do is provide the user the ability to call the backend “rgbAnalysis”

b. To test this, simply use a phone to click a button, and then have some kind of visual confirmation that in the backend code, the function was called

i. If the function is called, it will change the value of some text on the screen, renaming it to “rgbAnalysis”

2. takePicture();

a. All this needs to do is provide the user the ability to call the backend “takePicture”

b. To test this, simply use a phone to click a button, and then have some kind of visual confirmation that in the backend code, the function was called

i. If the function is called, it will change the value of some text on the screen, renaming it to “takePicture”.

dropdown variables;

a. This function simply needs to provide dropdown boxes for the user to change the values of dropdown variables

b. To test this, you can simply use the phone to interact with these variables, and change the value of some text on the screen, renaming it to the value chosen

4. switch variables;

a. This function simply needs to provide dropdown boxes for the user to change the values of switch variables

b. To test this, you can simply use the phone to interact with these variables, and change the value of some text on the screen, renaming it to the value chosen

5. slider variables;

a. This function simply needs to provide dropdown boxes for the user to change the values of slider variables

b. To test this, you can simply use the phone to interact with these variables, and change the value of some text on the screen, renaming it to the value chosen

References and File Links

https://tutorial.eyehunts.com/android/android-spinner-with-example-in-kotlin/

https://developer.android.com/reference

4.9. Revision Table

3/3/2021	Ben: Populated the section outline to allow copy-pasting and consistent formatting.
5/6/2022	Jacob: Populated the section with all necessary subsections and ported in my two blocks' information. Michael: Populate RasPiHat and Portable enclosure sections Ben: Populated Power supply and Fluorescence detector blocks Graham: populated my two blocks' information

5. System Verification Evidence

5.1. Universal Constraints

The following system constraints were selected by the ECE 44x instruction team in coordination with our project partners at the Cheng Research Group. Validations for each requirement are given below.

Constraint 1 – Breadboards

The system may not include a breadboard.

This system design utilizes PCBs, and various connectors, which is a system without a breadboard. The system therefore doesn't include a breadboard.

Constraint 2 – Mobile Application

The final system must contain both: a student designed PCB and a custom Android/PC/Cloud application.

This system design utilizes a student designed raspberry pi HAT (PCB) to act as a place for connections to be made between various parts of the system, as well as a power supply PCB to provide power to the system. Kotlin was used to create an android app, allowing the overall system to be controlled via an app.

Constraint 3 - Enclosure

The contents of the enclosure must be ruggedly enclosed/mounted as evaluated by the course instructor.

Enclosure and enclosure components are custom designed and 3D-printed using PLA plastic. Enclosure is designed to use threaded heat set inserts to mount enclosure components, PCBs and electronic modules. Each component is mounted to the enclosure using 2mm screws.

Constraint 4 - External Electrical Interface

All wire connections to PCBs and going through an enclosure (entering or leaving) must use connectors.

This system design utilizes the following connections between PCBs and through the enclosure. For each connection, a spec is given for the connector.

Connection Name	Connector Type	Manufacturer	Part Number
External Power Input	Micro USB Type B	Amphenol	10103594-0001LF
Raspi Hat to Fluorescence Detector LEDs (3)	2-Pin Ribbon Cable	Molex	530480210
Raspi to Fluorescence Detector Camera	22 pin ribbon cable	Molex	54548-2271

Table 8: External Connection Table

Constraint 5 - Energy Efficiency

All power supplies in the system must be at least 65% efficient.

The power supply circuit is shown below in Figure 4. Section 4.3 gives a detailed description of the design block and its functional components. The two limiting factors for an energy efficient design are the individual efficiencies of the MAX1555 battery charger and the ISL97516 boost regulator; Datasheets for these two components are given below in sections 5.10.1 and 5.10.2, respectively.

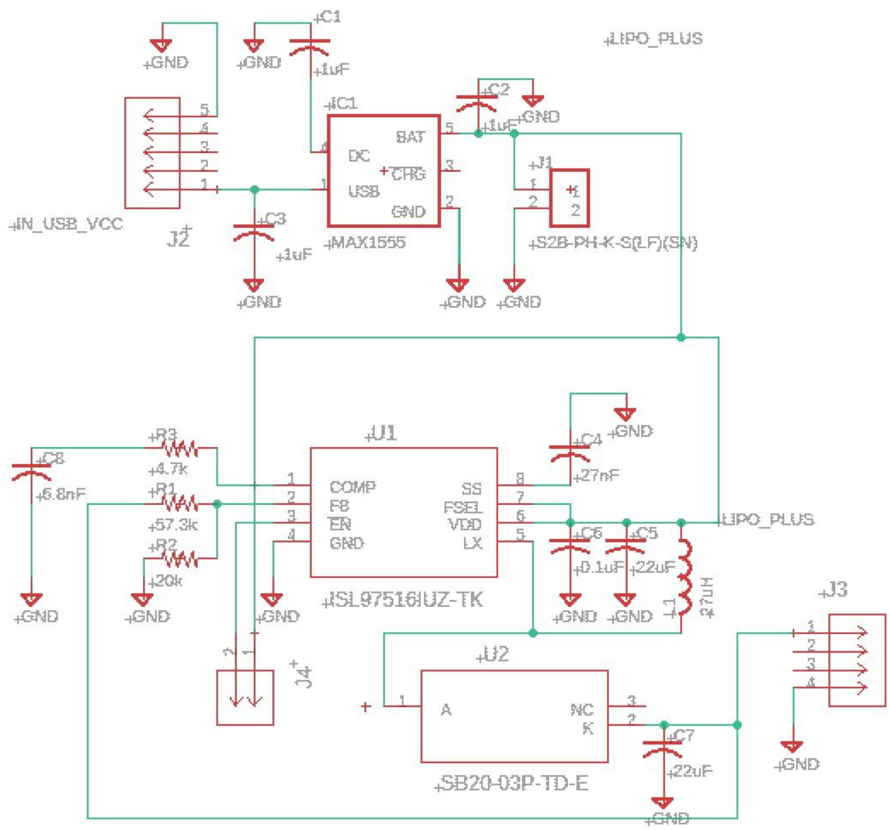


Figure 4: Electrical Schematic of the entire circuit

The Max1555 datasheet does not give an efficiency spec, but one can be calculated from other parameters. A maximum DC supply current is given as $3mA$, and "DC to Bat on-resistance" is $2\ \Omega$. These parameters give a maximum power loss of $P_{loss} = I^2R = 18\mu W$. Using the portable subsystem's maximum input power, $P_{in,max} = 1.5A * 5V = 7.5\ W$ yields:

$$\eta_{1555,min} = \frac{P_{out}}{P_{in}} = (P_{in})\frac{1-P_{loss}}{P_{in}} = 99.98\%$$

The ISL97516 has a much more helpful datasheet, and minimum efficiency is given as $\eta_{ISL,min}$ =90%. The entire system efficiency is then derived to be:

$$\eta_{min} = \eta_{1555,min} \times \eta_{ISL,min} = 89.98\%$$

Constraint 6 – Purchased Modules

The system may be no more than 50% built from purchased modules.

With regards to the mobile application subsystem, there are no purchased modules used at all. This subsystem was built within the free to use Android Studio IDE. It uses a number of open source libraries including OpenCV and Android Material Design. The embedded code from the portable device subsystem also uses no purchased modules. All imported libraries are open source and all code was either legacy code from previous iterations of the same project provided by our project partner, or our own code.

The enclosure for the portable device subsystem was also designed and printed by us, making it void of any purchased modules. The same is true for the PCB which we designed and printed for the power supply. The power supply itself, as seen in Figure 4 above, is made of a combination of purchased modules as well as raw electronic materials like resistors and capacitors. However, a cursory visual inspection of the schematic shows that these purchased modules do not dominate the schematic, and certainly are not great enough to comprise 50% of the entire project either. It is clear that this project satisfies this constraint, as purchased modules are only present in the power supply block, and do not even dominate that block to begin with.

5.2. Requirement 1 – Portable

Requirement

The portable device sub-system needs a secure enclosure, with an opening to house a sample slide. Internal components (list is provided below) shall be secured to the enclosure and shall comply with the following verification method.

Testing Processes

Step 1: Visually inspect the enclosure for loose/unsecured components.

Step 2: Note the positions of specified enclosure components (see list below).

Step 3: Rotate the enclosure 180° to be upside down, then another 180° to return to its original position

Step 4: Check all sub-components for movement. This verification will fail if any sub-components move more than 2mm.

List:

1. Power supply PCB
2. Debugging LEDs PCB
3. Raspberry Pi Zero
4. Camera
5. Excitation LEDs
6. Battery

Testing Evidence

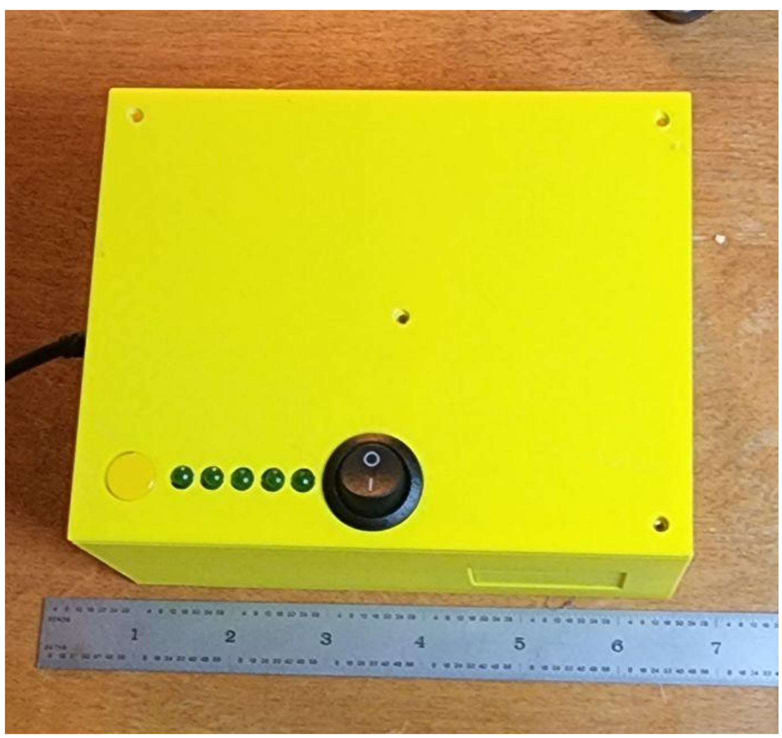


Figure 5: a Portable subsystem

5.3. Requirement 2 – Excitation Bandwidth

Requirement

Engineering Requirement 2: The system shall use at least three LED excitation sources with a light spectrum of less than 50 nm as measured at full width half maximum (FWHM). Note: this is a project constraint.

Testing Processes

Step 1: An excitation LED will be powered on by the portable subsystem
Step 2: The subsystem will be placed into the spectrometer in the Cheng Research Lab located in Room 302 of Oregon State University's Dearborn Hall.
Step 3: The spectrometer will measure the emission spectrum of the LED
Step 4: The data gathered by the spectrofluorometer shall be analyzed to determine the LED's FWHM in nanometers.
Step 5: Steps 1-4 will be repeated for each LED excitation source.

Testing Evidence

Data collected from the Cheng Lab spectrometer is included in the file linked in section 5.10.3. This data was plotted in MATLAB yielding Figure 6, below. The FWHM was computed using the two closest data points to half of the peak intensity (note this is why the FWHM are not horizontal on the graph). The MATLAB script used to generate this Figure is included in section 5.10.4.

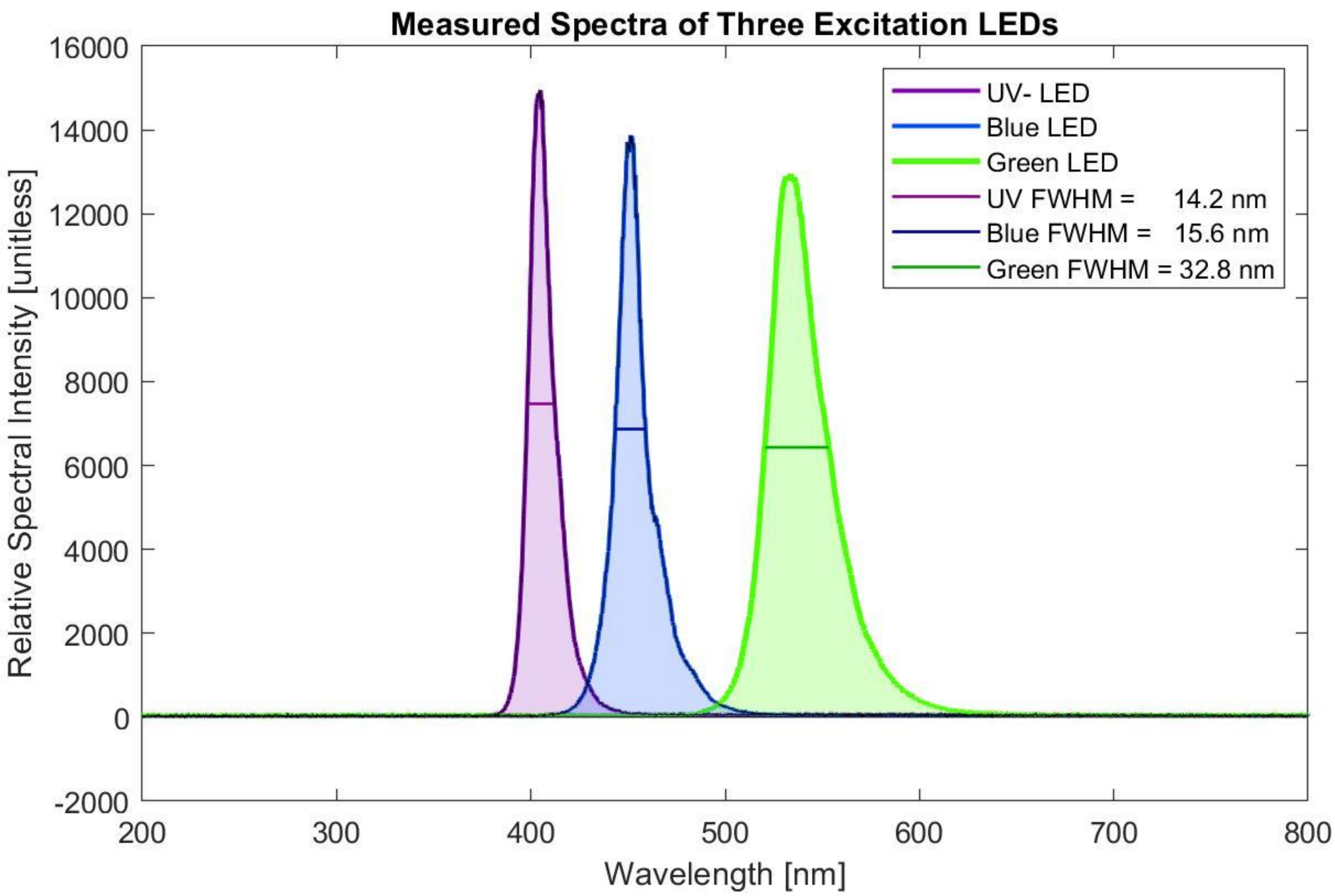


Figure 6: LED spectral outputs and their measured FWHM values

5.4. Requirement 3 – Excitation and Camera Control

Requirement

The device will change camera settings and excitation LEDs from an Android device via a bluetooth app interface.

Testing Processes

Step 1: The Android based mobile application subsystem will be demonstrated to wirelessly connect with the portable subsystem.
Step 2: A specific excitation parameter will be selected
Step 3: A user will demonstrate initiating the image capture by only using the mobile application subsystem.
Step 4: Steps 2-3 will be repeated using different excitation parameters.

Step 5: Image data displayed on the mobile application subsystem will be demonstrated to be different for each excitation parameter captured.

Testing Evidence

Testing is completed in this video link.

5.5. Requirement 4 - Wireless Communication Resilience

Requirement

The portable device subsystem will wirelessly send images to the mobile platform from at least twelve feet away.

Testing Processes

Step 1: The portable device subsystem will be placed at least twelve feet from the mobile device running the mobile application subsystem as measured by a tape.
Step 2: A distinct image will be initiated by the mobile device.
Step 3: The system will pass the test when the image is transmitted successfully from the portable enclosure to the mobile device

Testing Evidence

A video of completed testing is available at this link.

5.6. Requirement 5 - Fluorescence Spectrum Analysis

Requirement

The application will generate a fluorescence emission spectrum graph.

Testing Processes

Step 1: The mobile application subsystem will be demonstrated with a user selecting arbitrary image capture parameters while a given sample is loaded into the portable device subsystem.
Step 2: The button to take an RGB analysis of the sample will be clicked.
Step 3: The resulting RGB spectrum will be observed when it is saved to the mobile device.

Testing Evidence

There is currently an unresolved bug in the OpenCV code that computes the spectrum graph. The RGB analysis button correctly issues a take image command to the Pi wirelessly, receives the picture over bluetooth, saves it, and attempts to calculate the spectrum but the bug prevents it from completing the calculation at this time.

5.7. Requirement 6 – System Effective Power

Requirement

The portable device shall capture at least 30 images with one battery life.

Testing Processes

Step 1: Ensure that the device is not plugged into mains (120 VAC) power.
Step 2: The portable device subsystem will be demonstrated connecting to the mobile phone application subsystem
Step 3: The mobile phone application will initiate a picture from the portable subsystem
Step 4: Step three will be repeated until thirty image captures are initiated. This test is meant to measure the portable battery life, therefore resultant image quality will not be a factor.

Testing Evidence

The proof video was too long to upload as a single file. It has been split into three parts at the following links: Part 1, Part 2, Part 3.

5.8. Requirement 7 – Battery Life

Requirement

The portable device will have a battery life of over 24 hours of continuous "on" operation without data transmission.

Testing Processes

Step 1: Ensure that the device is *not* plugged into mains (120 VAC) power.
Step 2: The portable device will be demonstrated to be "on" by flipping a switch on the side to the "on" position and noting the presence of an indicator LED.
Step 3: Immediately after Step 2 a timer will be started.
Step 4: The portable device will be checked at two specified intervals during a 24 hour period measured by the timer in step 3. At each check time, one time stamped photo will be taken such that an indicator LED is clearly visible. These intervals are listed below.
Step 5: When at least 24 hours has elapsed since starting the timer in step 3, One final check will be performed. One final time stamped photo will be taken such that an indicator LED is clearly visible.

Step Number	**Interval Timing (Hours: Minutes)**
4.A.	06:00- 10:00
4.B.	14:00- 18:00
5.	>24:00

Testing Evidence

System integration between hardware and software was not completed until 14 hours before System Verification. Instead of rewriting this requirement, we decided to prioritize proving the most requirements. This should be easily provable with time.

5.9. Requirement 8 - Status Detection

Requirement

The mobile application subsystem will have at least 5 status indicators.

Testing Processes

Step 1: Request an image to be taken with the BLUE excitation LED on, and when the camera goes to take the picture look at the status LEDs, and make sure that both the BLUE excitation LED is on, as well as the corresponding status LED (far left).

Step 2: Request an image to be taken with the GREEN excitation LED on, and when the camera goes to take the picture look at the status LEDs, and make sure that both the GREEN excitation LED is on, as well as the corresponding status LED (second from far left).

Step 3: Request an image to be taken with the AMBER excitation LED on, and when the camera goes to take the picture look at the status LEDs, and make sure that both the AMBER excitation LED is on, as well as the corresponding status LED (third from far left).

Step 4: Request to run the error code example from the testbench script (execute cmd), and make sure the 1st error code LED turns on and off (third from far right).

Step 5: Request to run the error code example from the testbench script (execute cmd), and make sure the 2nd error code LED turns on and off (second from far right).

Step 6: Start with the system off, and make sure that the power status LED is off (far right). Then turn it on with the power button, and make sure that the power status LED is on.

Testing Evidence

The current code is correctly calling the GPIO output function that changes the voltage on the pin to high or low. However, when measured with a DMM on the pin with respect to ground, it remains reading 0 volts. We are almost certain that this is not a software issue. Troubleshooting this seems to point to a hardware issue: the Raspberry Pi Zero may not be regulating down to 3.3v DC for GPIO pins- instead regulating down to 0V.

5.10. References and File Links

MAX1555 Battery Charger Datasheet (pdf)
ISL97516 Boost Regulator Datasheet (pdf)
LED Spectrum Data
MATLAB Script to calculate measured FWHM values

5.11. Revision Table

3/3/2021	Ben: Outline placeholder population for formatting consistency.
3/6/2021	Team: Populated sections 5.1-5.11 with system requirements from revised sections 2.1-2.8. Added testing evidence where applicable.
4/29/2022	Team: Revised requirements after review from project partners and instructor team
5/2/2022	Team: Added video links for verifications.

6. Project Closing

6.1. Future Recommendations

Technical Recommendations

Recommendation 1: Spectrometer Accuracy

One of the biggest limiting factors for widespread use and adoption of the current platform is the spectral response of the detection camera. Our system records individual Red, Green, and Blue (RGB) values using the Raspberry Pi V2 camera as the main sensor. Recorded RGB values are measured and each pixel's RGB triplet is matched to a corresponding peak wavelength. This method gives an effective average wavelength for each pixel instead of a more granular spectral response- e.g. a sample with two peak wavelengths at 600 nm 650 nm would be measured by the system as having a peak wavelength of 625 nm.

Furthermore, the spectral responses of the R,G, B bands inside the sensor are fairly wide and not easily modeled. A detailed series of measurements of the Raspberry Pi V2 camera was completed by Pagnutti et al. 2017 (See Reference 6.4.1), yielding the spectral response graph in Figure 7, below.

These spectral response bands do not have clean empirical mappings to allow system processing to determine specific peak wavelengths.

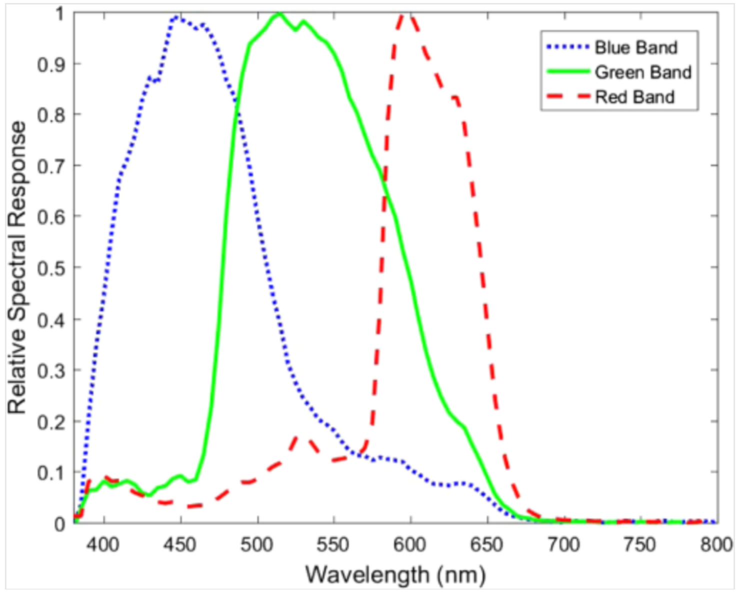


Figure 7: RasPi Spectral Response, published by Pagnutti et al. (See Reference in section 6.4.)

An overview of spectroscopy instrumentation design techniques is beyond the scope of this section, but a number of relatively inexpensive and easy to integrate spectral sensors are available. One example sensor would substitute the three bands above with 18 evenly spaced bands with clean gaussian spectral responses. Integration of this sensor would greatly increase the system's spectral accuracy.

Recommendation 2: Optical Considerations

A number of further optical design techniques could be used to increase the accuracy limited by the optical design of the system.

The current system uses a 405-nm high-pass filter that doubles as a microscope lens, despite exposing the sample to peak emission wavelengths as large as 550 nm (See Fig. 6 above). An ideal system would use a high-pass spectral filter with a cut off wavelength above peak excitation wavelengths- here around 600 nm. Pre-made optical filter components were outside of the budget for our system, and time constraints prevented the design and testing of homemade filters.

The current fluorescence detection design block does not use an aperture to collimate the light emitted by the excitation LEDs. Solutions to this problem could be as simple as adding a 3D printed aperture to each LED's optical path to collimate the light.

The current system does not use optical best practices to isolate transmitted light. Ideal fluorescent spectrometers use at least one dichroic mirror, and many use at least two. The use of dichroic mirrors in the design would require precise alignment and mechanisms for realignment- this in turn would limit the portability and usability of the system. A good review of microscopic fluorescence optical design was published in Sanderson et al. 2014, (see Reference In Section 6.4.).

Recommendation 3: Automated System Procedures

One issue that hindered the testing process was the lack of automated system testing procedures. This functionality was planned to be implemented, but not executed because of lack of time before deadlines. Many of the system requirements detailed in Section 5 could have been integrated into the system with redesigns amounting to about two weeks worth of work.

Examples of automated system procedures include:

- Developing a hidden menu in the mobile application which would automate the batch collection of data for Requirement 6 (Section 5.7)
- Redesigning the battery/power system to be directly wired to built-in analog to digital converters on the Raspberry Pi to monitor battery power. This would require a rewrite of Requirement 7 (Section 5.8), but would result in a more portable and easily repeatable system test.

Recommendation 4: Improving Bluetooth Communication

Another possible future iteration on the current state of the project might be a revamp of the bluetooth application. In its current state, it is fully functional, but revisiting it with ease of use in mind would be beneficial. For example, currently the images transmitted from the Pi are saved onto the phone in its internal storage, but in order to access them, the user must leave the application and enter your file management system. Inclusion of a built in photo reel for all images received that is fully integrated into the application would do wonders for ease of use. Along with that, a specific menu for settings would reduce the need to scroll up and down on the application, and may be a considerable improvement for usability.

Global Impact Recommendations

Recommendation 1: Reducing Lithium Dependance

The most prudent recommendation for reducing the negative impact globally of this project would certainly be the elimination of the Lithium battery (LiB) from the system. As soon as cost effective alternatives to current batteries are available, integrating them into this project in place of the LiB would be an excellent way to reduce both the environmental and human impacts that this project contributes to by its reliance on LiBs. See document Section 2.2 for more detail on these impacts. Strategies, motivations, and methods for reducing lithium and cobalt dependencies for battery energy storage are also discussed further in the Reference 6.4.

Recommendation 2: Maintaining Cost Efficiency

The second recommendation that we can give for future engineers iterating on our work is to maintain the low cost per unit. One of the main constraints of this project was to ensure that price creep does not occur. It is in the best interest of the end user that this product remains low cost, as it is intended to provide this technology to people who could otherwise not afford it, along with for use cases where main power is not provided. These cases often coincide. Thus it would be fitting with the design direction of the team to improve the project while not unnecessarily increasing the cost per unit, as doing so would price out the intended user. If this or a derivative system does end up contributing to major advances in healthcare or biological sciences, it is ethically imperative that the system cost does not add to already inflating prices. A thorough account of healthcare inflation is given in reference 6.4.

Teamwork Recommendation

Recommendation 1: Collaboration Standards

Our team was harmed by the lack of early and persistent communication of standards for expected contributions from each team member. Although we did discuss expectations for grades and deadlines at an early team meeting, when these standards were communicated to the member not present at the meeting, they were told to him rather than built with him. This may have contributed to some perceived antagonism that was allowed to grow as deadline frequency increased.

Recommendation 2: Deadline Navigation

Our team's collective navigation of deadlines was not ideal. This especially led to issues with system integration where block dependencies tended to cascade and delay the entire project. This could have been improved by valuing co-working time during team meetings, thus ensuring that when deadlines would not be met, all stakeholders would have a thorough understanding of complicating issues and a chance to weigh in on troubleshooting these issues. This could also be navigated by better communication standards and practices- for example texting when other messaging platforms did not elicit a timely response.

Recommendation 3: Contingency Plans

One aspect of the collaboration process that was lacking in our teamwork was the development of contingency plans- especially in relation to teamwork risks. This could have helped other team members shoulder the burden of any team members who missed deadlines, and would have motivated all team members to better contribute to the collective effort. If a deadline was missed, contributing team members would have a back-up plan in place, and would be able to move forward without having to worry about what would happen if the absent team member decided to contribute at the last minute. More details and recommendations on contingency planning are given in Section 6.4.

Recommendation 4: Contribution Equity

Finally, the contribution equity between group members was not balanced. This issue is not as simple as a single member choosing not to contribute. To look at it through that lens is unproductive. Instead, we recommend that in the future, teams should lay out divisions of work early and often, and work to hold each other accountable for missed deadlines both soft and hard. Meeting often as a team to discuss upcoming tasks and divide labour early would help to prevent this from becoming an issue. Along with that, regular check-ins with group members on their progress in their section of the task would help keep group members accountable for their contributions and ensure that they make those in a timely manner.

6.2. Project Artifact Summary

Sample Holder Tray Redesign Drawing
Sample Holder Tray Redesign STL File
LiPo Charger/Power Distribution Board Schematic
LiPo Charger/Power Distribution Board PCB Design
LED Board Schematic
LED Board PCB Design
LED Switch Board Schematic
LED Switch Board PCB Design
LED Switch Board Mounting Bracket STL

6.3. Presentation Materials

Project Poster

This poster was designed to conform to the standards of Oregon State University's 2022 Engineering Expo.

Project Poster Revision Table

The revision table for this poster is included at this link.

6.4. References and File Links

Pagnutti et al. "Laying the foundation to use Raspberry Pi 3 V2 camera module imagery for scientific and engineering purposes," in Journal of Electronic Imaging, Volume 26, 2017, doi: 10.1117/1.JEI.26.1.013014. Link

Sanderson, Michael J et al. "Fluorescence microscopy." Cold Spring Harbor Protocols vol. 2014,10, 1 Oct. 2014, doi:10.1101/pdb.top071795. Link

Office of Energy Efficiency and Renewable Energy. "Reducing Reliance on Cobalt for Lithium-ion Batteries". Energy.gov. Published April 6th, 2021. https://www.energy.gov/eere/vehicles/articles/reducing-reliance-cobalt-lithium-ion-batteries (Accessed Dec. 2nd, 2021)

PGP Foundation. April 20, 2020. “Why Are Americans Paying More for Healthcare”. Peter G. Peterson Foundation. https://www.pgpf.org/blog/2020/04/why-are-americans-paying-more-for-healthcare (Accessed Oct. 26th, 2021)
Heimann, J. F. (2000). Contingency planning as a necessity. Paper presented at Project Management Institute Annual Seminars & Symposium, Houston, TX. Newtown Square, PA: Project Management Institute. Link

6.5. Revision Table

5/6/2022	Ben: Outline & placeholder population for formatting consistency. Tech Recs 1-3, Team Recs 1-3, 9 artifacts, poster and revision table
5/6/2022	Jacob: Completed remaining sections and general formatting.