# Heliostat Control Tracking

Samuel Dixon
Jordan George
Erica Quist

## Concept of Operations

REVISION – Draft D
16 November 2022

# Concept of Operations
for
# Heliostat Control Tracking

Team <66>

Approved by:

| | |
|---|---|
| Jordan George | 10/13/2022 |
| Project Leader | Date |
| Dr. Kalafatis | 10/13/2022 |
| Prof. Kalafatis | Date |
| Dalton Cyr | 10/13/2022 |
| T/A | Date |

## Change Record

| Rev. | Date | Originator | Approvals | Description |
|---|---|---|---|---|
| **A** | 9/9/2022 | Erica Quist, Samuel Dixon, Jordan George | | Draft Release |
| **B** | 9/30/2022 | Erica Quist, Samuel Dixion, Jordan George | | Updated System Description to accurately display new subsystem division. Updated block diagram to represent new systems. Updated Executive Statement. |
| **C** | 10/13 | Samuel Dixon | | Removed highlighting, updated subsystems, and fixed grammar issues based on feedback. |
| **D** | 11/16 | Samuel Dixon | | Updated project changes, repartitioning of subsystems |

# Table of Contents

# List of Tables

No table of figures entries found.

# List of Figures

# 1. Executive Summary

The goal of this research project is to help develop a heliostat for a small-scale solar thermal system consisting of a heliostat to Fresnel lens to target optical path. Currently, heliostat manufacturers design their products for very large CSP systems (concentrating solar power, i.e. power towers) that are not suitable for this R&D project as it would be too cumbersome, costly, possible proprietary issues, and the timeline would be unpredictable.

To demonstrate basic system performance in a timely manner, we will assemble a heliostat and demonstrate solar tracking, through a lens, to a focused target spot with motor control for two axis positioning of the heliostat into a fixed lens/target fixture and incorporate position feedback for closed loop operation. Having a stand-alone heliostat unit that can be easily setup, measured under sun, and returned to the lab for storage is desired. A small battery and possibly a PV panel and controller should be integrated onto the heliostat unit. Wireless control, sensor feedback, possibly a camera on the target area, data acquisition and data logging are desirable using a laptop (MS Windows) and Matlab. The solar input should be measured and an estimate of the solar energy delivered to the target area made (more on this later, we have to measure the direct normal irradiance - ie. DNI - separate from the global horizontal irradiance using a homemade rotating shadowband on an Apogee pyranometer).

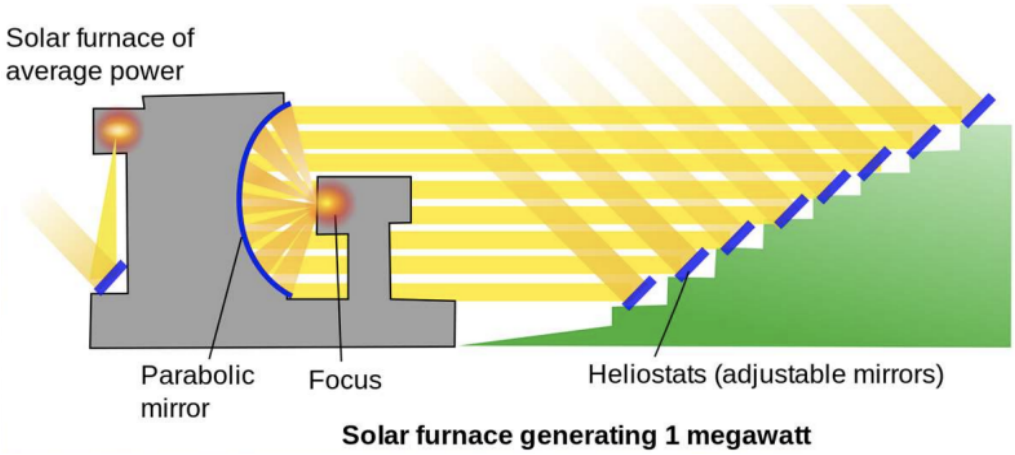


*Figure 1: General concept of a solar furnace*

# 2. Introduction

This paper provides an introduction to a Solar Tracking Mount for a solar furnace. This tracking mount will rotate and tilt the planar mirrors based on where the sun is located. The purpose of this would be to allow for the mirrors to concentrate more of the sun's energy onto the focal point of the solar furnace. A more efficient solar furnace would allow for the varying equipment that are powered by these devices to have a stronger source of power.

## *2.1. Background*

The project will improve energy infrastructure and generation systems around the world by focusing light onto a target from the sun. Making use of the sun's available energy limits the dependence on non-renewable energies, which are currently being depleted at an unsustainable rate. Additionally, this sustains less environmental impact. This project seeks to improve the feedback control aspect for tracking the sun's position. This will be done by providing the control based on permutations of a multitude of parameters. Additionally, the method of connecting the mirrors and the motors to a provided frame for optimal optical alignment will be improved. This will be done through the use of hardware fasteners, actuators, and motors. This new configuration will provide optimal support for collecting the maximum amount of thermal energy on the target, while using minimal amounts of input power.

## *2.2. Overview*

The Solar Tracking Mount will be started off by assembling the array of planar mirrors onto a single frame. The frame will have the ability to rotate and tilt the mirrors. Each mirror will have a motor to allow each mirror to have its own tilt. Also, there will be actuators that will be assisting the motors in the mission to tilt the mirrors in the optimal position. There will be sensors that will detect the optical output and input for each mirror. That data will be used to adjust the mirror until the maximum optical output is obtained. A sun tracking algorithm will be coded into the microprocessor to keep the mirrors in the most optimal position to output the most of the sun's energy. There will also be data visualization that the user of this tracking mount will be able to view. This will provide insight to the user of what exactly is happening with the mirrors and their optics information.

## *2.3. Referenced Documents and Standards*

- https://www-sciencedirect-com.srv-proxy1.library.tamu.edu/science/article/pii/S0960148116311624
- http://www.powerfromthesun.net/Book/chapter08/chapter08.html
- https://www-sciencedirect-com.srv-proxy1.library.tamu.edu/science/article/pii/S0038092X00001560
- https://www.sunearthtools.com/dp/tools/pos_sun.php?lang=en#annual

# 3. Operating Concept

## *3.1. Scope*

The Solar Tracking Mount will accelerate the transition to renewable energy by adaptively focusing the sun's light onto a target through the use of a rotating mirror array, commonly referred to as a heliostat. The target will be equipped with a sensor used to provide feedback into the processor of the motor drivers, actuating the adjustment of mirrors to an optimal point for maximized thermal output. The system will provide a data visualization device to communicate system-level information, such as sensors, current mode of operation, and log criteria observed for decision making.

## *3.2. Operational Description and Constraints*

This project will be used in thermal energy conversion applications, particularly involving a thermal plate or furnace, in which heat is converted into another form of energy. Particularly, this system will serve as an input to a larger system, where electrical energy can be provided to consumers through renewable energy.

## *3.3. System Description*

The Solar Mount Tracking will consist of controlling the motors that can rotate and tilt the mirrors, measuring the optical output and input of the mirrors, and an algorithm to allow the array of mirrors to track the sun and rotate accordingly. There are 6 subsystems for this project:

- PCB design
    - The PCB board will interface the tracking controls processor with the power supply, motors, data visualization tool, and sensors. The board is a place where the electrical components are organized and the signals are routed to their respective positions.The board also handles the power distribution and delivery to all the various components.
- Data visualization
    - This tool will provide valuable insight to verify different aspects of the system and communicate different metrics across the system to the user.
- Tracking Controls
    - The tracking controls are the processor and brain of the system. Through feedback received from the sensors, time of day, and other sources of valuable information, will decide how to change the positioning of the mirrors in the system.
- Optical Mirror Setup
    - The physical setup is how the motors, mirrors, actuators, and frame will be configured. By creating a controllable array of mirrors on a frame, the controller's job will be easier to effectively channel the light onto the target. Although

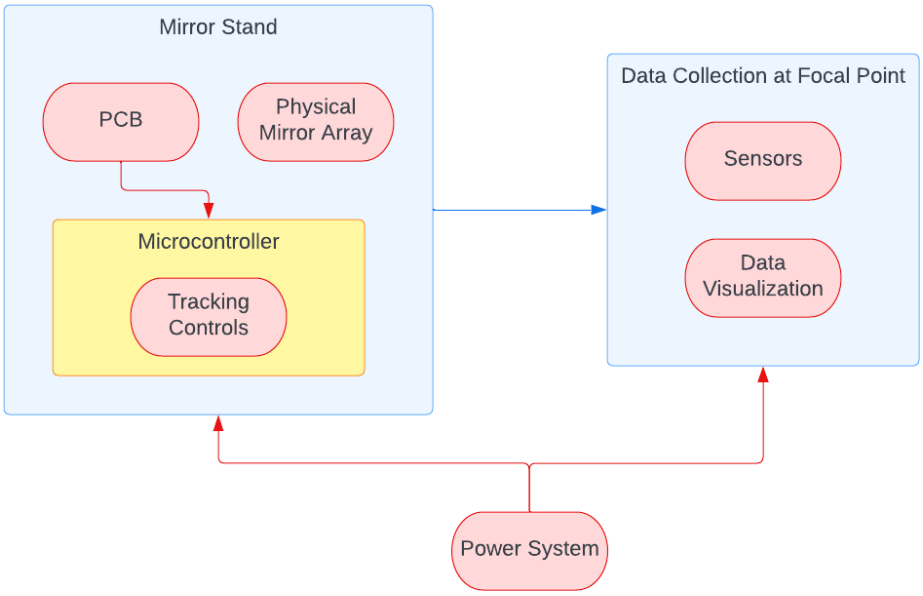


*Figure 2: Subsystem block diagram*

## 3.4. Modes of Operations

The system will have different modes of operations depending on the time, weather, and season. Specifically, the system will have four modes: a sleep mode (for times when it is impractical to have the system, such as night), full range operating (for times when the sun is fully available), partial range operating (for seasons, such as winter, or weather, such as cloudy for when the optimum amount of sunlight is not available), and lastly, the system will have a debug mode (where the system is expecting to be tested, changed, and manipulated).

## 3.5. Users

Researchers, utility providers, and customers are all potential users of this system. This system intends to be a basis for providing sustainable energy for use. By providing robust tracking of the movements in the sun, change in weather, and different seasons, the system gives researchers a head start in thermal energy conversion systems, where heat can be converted into a more useful form, such as electricity. This would provide utility providers a method of incorporating a renewable energy source to distribute power to different businesses.

## *3.6. Support*

Support will be provided through a user manual. The document will describe the scope of the system, the purpose, and how to change operating modes and parameters in the program. Additionally, the manual will provide information on how to “power on” the system and use the Android application for data interpretation.

# 4. Scenario(s)

## *4.1. Campsite bathroom lighting*

When camping there is commonly a public bathroom available for people to use. Sometimes these public bathrooms are transportable and have no electrical power, but other times they are established bathrooms with lighting. A potential use case of this application would be to replace the energy source of the bathrooms with lighting. Fortunately, when camping there is typical natural light during the day. Using this application to collect and store energy during the day, when natural light is available in the bathroom would let non-renewable energy be replaced in the evening, when lighting is needed.

## *4.2. Airport Charging Applications*

When traveling there is a varying demand of people needing to charge appliances in airports. This variation is due to the nature of supply and demand. Increased travel typically occurs when people have an abundance of time, cultural holidays, and the price of tickets. In airports, there is a large amount of variation in power required to charge electrical appliances, such as phones or laptops. In a smaller airport near the tropical, sunny climate, these spikes of variation could be accounted for by this problem.

## *4.3. Thermal Energy System Research*

There is substantial research being done in thermal energy conversion systems. This system would provide a valuable input into those systems, as the optimal supply would be provided to enable the largest throughput possible for usable energy at the output of their energy conversion system.

# 5. Analysis

## *5.1. Summary of Proposed Improvements*

This system will provide improvements in tracking, by basing the feedback on a diverse suite of parameters. Additionally, the proposed system will provide data driven-insights based on the light collected on the target and the positioning of the motors. Lastly, the system will offer in-program functionality adjustments and calibration to accommodate different geographic locations and conditions.

## *5.2. Disadvantages and Limitations*

The proposed system will have some apparent limitations. Quite notably, is the fluctuation in the ability to focus light during inclement weather, different seasons, and the time of day. Additionally, there may be substantial tradeoffs in the power needed to power the system and the amount of light focussed on the target.

## *5.3. Alternatives*

An alternative solution may be to use solar panels rather than mirrors when tracking the light, although this alternative could potentially be more costly to implement for collecting the sun's energy. Additionally, maintenance for mirror damage is likely less expensive than the cost of damage incurred on a solar panel. Another alternative could be to control the thermal plate to collect the energy, rather than focus the light through mirrors. This method would reduce the complexity of having both a target and a focussing system, although without focussing the light, a reduction in luminous intensity would be observed. The tradeoff here is observed between complexity and energy collected. Additionally, there would be less spatial coverage, as heat dissipates with a larger surface area.

## *5.4. Impact*

The project impacts the way society uses energy. Currently, the majority of the world's energy consumption is derived from fossil fuels. Recent trends and investments show that the world is on track to change the majority of reliance to more sustainable alternatives. Through applications like this the world can accelerate the push to sustain the shift to a different form of energy. The implications of this shift would be an increase in the availability of fossil fuel supply, an alleviation of environmental stress and pollution ,a transformation of job markets in certain industries, and a shift in the marketplace value of certain commodities and businesses.

# Heliostat Control Tracking

Samuel Dixon
Jordan George
Erica Quist

# INTERFACE CONTROL DOCUMENT

REVISION – Draft B
13 October 2022

# Interface Control Document
for
# Heliostat Control Tracking

Prepared by: Erica, Jordan, Samuel

| | |
|---|---|
| Erica, Jordan, Samuel | 11/16/2022 |
| Author | Date |

Approved by:

| | |
|---|---|
| Jordan George | 11/16/2022 |
| Project Leader | Date |
| Dr. Kalafatis | 11/16/2022 |
| Dr. Kalafatis | Date |
| Dalton Cyr | 10/13/2022 |
| T/A | Date |

## Change Record

| Rev. | Date | Originator | Approvals | Description |
|---|---|---|---|---|
| **A** | 9/30/2022 | Erica Quist, Samuel Dixon, Jordan George | | Draft Release |
| **B** | 10/13/2022 | Samuel Dixon | | Removed highlighting and made revisions based on draft feedback |

# Table of Contents

# List of Tables

# List of Figures

No table of figures entries found.

# 1. Overview

The Interface and Control Document (ICD) will explore in-depth detail regarding the subsystem information described in the Concept of Operations (ConOps) and Functional System Requirements (FSR) respective documents. This will involve detailing the specific descriptions of the hardware and software being used in this project and how they interface with each other. Lastly, the ICD will give information regarding the characteristics of protocols being used for design.

# 2. References and Definitions

## *2.1. References*

**IEEE Standard 802.11**
**Standard for WIFI Networks**

**Android Studio API Online Reference**

The other Reference Documents are located in Section 2.2 of the Functional System Requirements document.

## *2.2. Definitions*

| | |
|---|---|
| API | Application Programmable Interface |
| DNI | Direct Normal Irradiance |
| g | Grams |
| GUI | Graphical User Interface |
| in | Inches |
| lbs | Pounds |
| mA | Milliamp |
| mW | Milliwatt |
| V | Voltage |
| Baud Rate | |

# 3. Physical Interface

## *3.1. Weight*

### 3.1.1. Weight of PCB Shield and ESP-32 System

| Component | Weight |
|---|---|
| ESP-32 PCB System | 13 g |
| L298n Motor Driver | 26 g |

*Table 1: Weight of PCB Shield and ESP-32*

### 3.1.2. Weight of Battery

| Component | Weight |
|---|---|
| Battery | 4.6 lbs |

*Table 2: Weight of the battery*

## *3.2. Dimensions*

### 3.2.1. Dimension of PCB Shield and ESP-32 System

| Component | Dimensions |
|---|---|
| ESP-32 System | 4.00 in x 2.10 in |
| L298n Motor Driver | 1.69 in x 1.69 in x 1.02 in |

*Table 3: Dimensions of PCB Shield and ESP-32*

### 3.2.2. Dimension of Frame

| Component | Length | Width | Height |
|---|---|---|---|
| Frame | 48 in | 48 in | 60 in |

*Table 4: Dimensions of the frame*

#### 3.2.3. Dimension of Mirror

| Component | Length | Width | Height |
|---|---|---|---|
| Planar Mirror | 12 in | 12 in | 0.16 in |

*Table 5: Dimensions of the planar mirrors*

#### 3.2.4. Dimension of Battery

| Component | Length | Width | Height |
|---|---|---|---|
| Battery | 5.94 in | 2.56 in | 3.70 in |

*Table 6: Dimensions of the battery*

## *3.3. Mounting Locations*

#### 3.3.1. Mounting of Sensors

There will be two magnetometers mounted on the system frame. Also, a photodiode will be mounted to a target that will be used to measure how much of the sun's energy the Heliostat Control Tracking system outputs.

#### 3.3.2. Mounting of Mirrors

The planar mirrors will be mounted onto the frame in a 3x3 array since there will be nine mirrors in this system. Support rails, epoxy, and fasteners will be used to complete this mounting task. The frame will have rotational and tilt capabilities to allow the mirror array to be angled for optimal output of solar energy.

#### 3.3.3. Mounting of ESP-32 Control Unit

The ESP-32 Control Unit will be within a metal box that is mounted on one of the sides of the frame. Most of the circuitry, circuit boards, and wires will be located in this box.

# 4. Electrical Interface

## 4.1. Primary Input Power

| Component | Max Voltage | Max Current |
|---|---|---|
| Battery | 12 V | 7 A |

*Table 7: Max Voltage and Current Values for the battery*

## 4.2. Max Voltage and Current Values

| Component | Max Voltage | Max Current |
|---|---|---|
| ESP-32 Wroom | 3.6 V | 260 mA |
| Magnetometer Sensor | 12 V | 40 mA |
| L298n Motor Driver | 35 V | 2 A |

*Table 8: Max Voltage and Current Values for PCB Shield and ESP-32*

## 4.3. Signal Interfaces

### 4.3.1. Rotation & Tilt Sensors

A triple axis magnetometer will be used to get the rotation and tilt of the mirror array. This magnetometer will be connected to the ESP-32 mega which will communicate with the Android app to display this data.

### 4.3.2. Luminosity Sensor

To measure the output of the system, a photodiode will be attached to a target. This target will be set up to receive the outputted solar energy from the planar mirrors. The ESP-32 also will be responsible for relaying the data from this sensor to the Android app to visualize the data. This data will be useful to find the most optimal angle of the mirror array for the highest output.

## 4.4. User Control Interface

The user control interface is an android application that will provide the user with data visualization and system analytics through bluetooth communication of the ESP-32 and wireless connection to the internet. The bluetooth communication will service instantaneous data received from the ESP-32 and the wireless communication will service historical data accumulated over time.

# 5. Communications / Device Interface Protocols

## *5.1. Wireless Communications (WiFi)*

In order for the android device to store the sensor data received through bluetooth communication from the microcontroller, it must connect to the internet and upload the data to the particular web server. The device will adhere to the Wireless Communication Protocol as specified in the IEEE Standard 802.11.

## *5.2. Device Peripheral Interface*

The respective TX and RX pins of the microcontroller will be sending and receiving serial data to and from the smartphone though the ESP-32 rover microcontroller. This data received on the pins will be adhering to serial communication protocol, specifically UART. The data being shared between the two devices is asynchronous, meaning the devices do not have their own clocks defining when information is sent or received. Therefore, it is important that these devices adhere to the protocol specifications and agree on a common data sharing rate, which is commonly known as the baud rate.

# Heliostat Control Tracking

Samuel Dixon
Jordan George
Erica Quist

# FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – Draft B
10/13/2022

# Functional System Requirements
FOR
# Heliostat Control Tracking

Prepared by: Erica, Jordan, Samuel

| | |
|---|---|
| Erica, Jordan, Samuel | 11/16/2022 |
| Author | Date |

Approved by:

| | |
|---|---|
| Jordan George | 11/16/2022 |
| Project Leader | Date |
| Dr. Kalafatis | 11/16/2022 |
| Dr. Kalafatis. | Date |
| Dalton Cyr | 10/13/2022 |
| T/A | Date |

## Change Record

| Rev. | Date | Originator | Approvals | Description |
|---|---|---|---|---|
| **A** | 9/30/2022 | Erica Quist, Samuel Dixon, Jordan George | | Draft Release |
| **B** | 10/13/2022 | Samuel Dixon | | Removed highlighting and made revisions based on draft feedback. |
| **C** | 11/16/2022 | Samuel Dixon | | Updated system requirements after changes done to subsystem organization from midterm presentation feedback. |

# Table of Contents

# List of Tables

# List of Figures

# 1. Introduction

## *1.1. Purpose and Scope*

Most heliostat products are made for large solar power systems. For this R&D project, a much smaller-scale heliostat system is desired. This is because larger scale heliostats are costly, burdensome, and have timelines that are unpredictable. The purpose of this research project is to create a stand-alone heliostat unit that can be set up easily, output data, and can be stored in the lab whenever desired. This solar tracking mount will consist of a mirror array that will be tilted and rotated based on where the sun is. A sun tracking algorithm will be coded onto an Arduino board that will move the mirror array accordingly with the help of motors. There will be an Android app that will visualize the data such as the input and output optical rays as well as the rotation and tilt angle of the array.

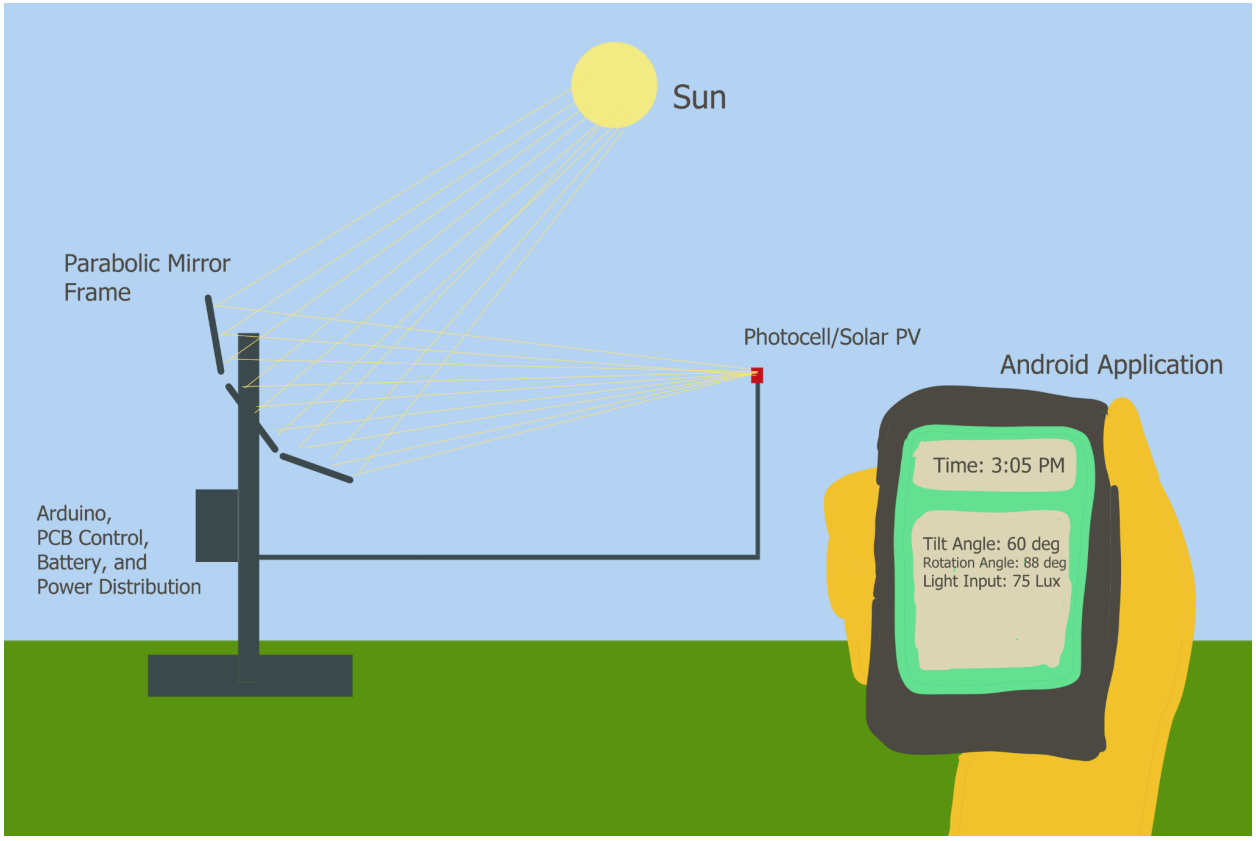


**Figure 1. Heliostat Concept Image**

## *1.2. Responsibility and Change Authority*

| Subsystem | Responsibility |
|---|---|
| PCB design & Power Delivery | Jordan George |
| Data Visualization | Samuel Dixon |
| Tracking Algorithm | Erica Quist |
| Physical Mirror Array | Jordan George, Samuel Dixon, Erica Quist |

*Table 1: Subsystem partitioning*

# 2. Applicable and Reference Documents

## *2.1. Applicable Documents*

The following documents, of the exact issue and revision shown, form a part of this specification to the extent specified herein:

| Document Number | Revision/Release Date | Document Title |
|---|---|---|
| 978-1-491-92176-0 | First Edition - 5/5/2016 | Arduino A Technical Reference |
| IEEE 802.11-2016 | 12/14/2016 | Standard for WIFI Networks |
| IEEE 802.15.1-2015 | 06/14/2005 | Standard for Bluetooth Communication |

*Table 2: Applicable Documents*

## *2.2. Reference Documents*

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

| Document Number | Revision/Release Date | Document Title |
|---|---|---|
| 1 | 01/03/2017 | Run-time detection and correction of heliostat tracking errors |
| 2 | 10/2003 | Concentrator Optics |
| 3 | 9/15/2001 | Computing the solar vector |
| 4 | 09/13/2022 | Android Studio API Online Reference |

*Table 3: Reference Documents*

## *2.3. Order of Precedence*

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.

All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

# 3. Requirements

## *3.1. System Definition*

This project entails refining the tracking controls, structural arrangement, hardware organization, and data visualization of a small-scale heliostat control system. These refinements include tracing the trajectory of the sun at a given longitude and latitude, while receiving feedback from sensors, incorporating planar mirrors with increased stability, routing signals on a PCB shield, and utilizing internet and bluetooth protocols to visualize system metrics in real-time and on a historical basis.

This proposed solution contains the following subsytems: PCB Design and Power Delivery, Data Visualization, Tracking Algorithm, and Physical Mirror Array. PCB Design and Power involves the physical hardware and the routing of electrical signals, while interfacing the capabilities of the Arduino. Data Visualization involves receiving bluetooth data from sensors and displaying the data in real time, while also storing the historical data into a remote server for other data driven insights. The Tracking Algorithm is the C programming to orchestrate the signals to actuate the motor control and adjustment of the mirrors as a function of the sun's position.

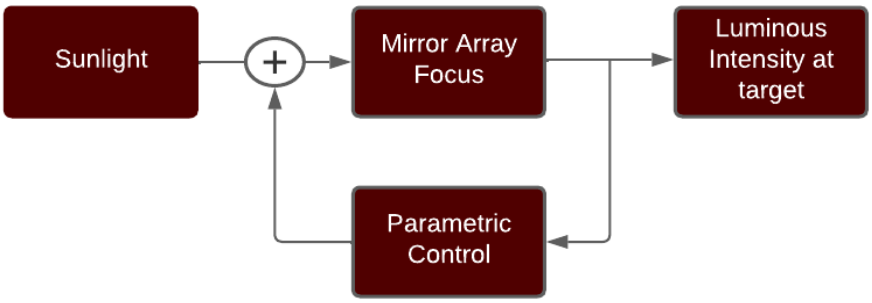


**Figure 2. Block Diagram of System**

The block diagram in Figure 2 visualizes the control aspects of this project. The input to the system is the position of the sun at a given time. In the forward path, the mirror array is focused based on the time and longitude data. The system then adjusts the position of the mirror array though other aspects of parametric control, such as data received from feedback in the program. This reduces the error signal between the optimal maximum luminous intensity and the observed maximum intensity at a given time.

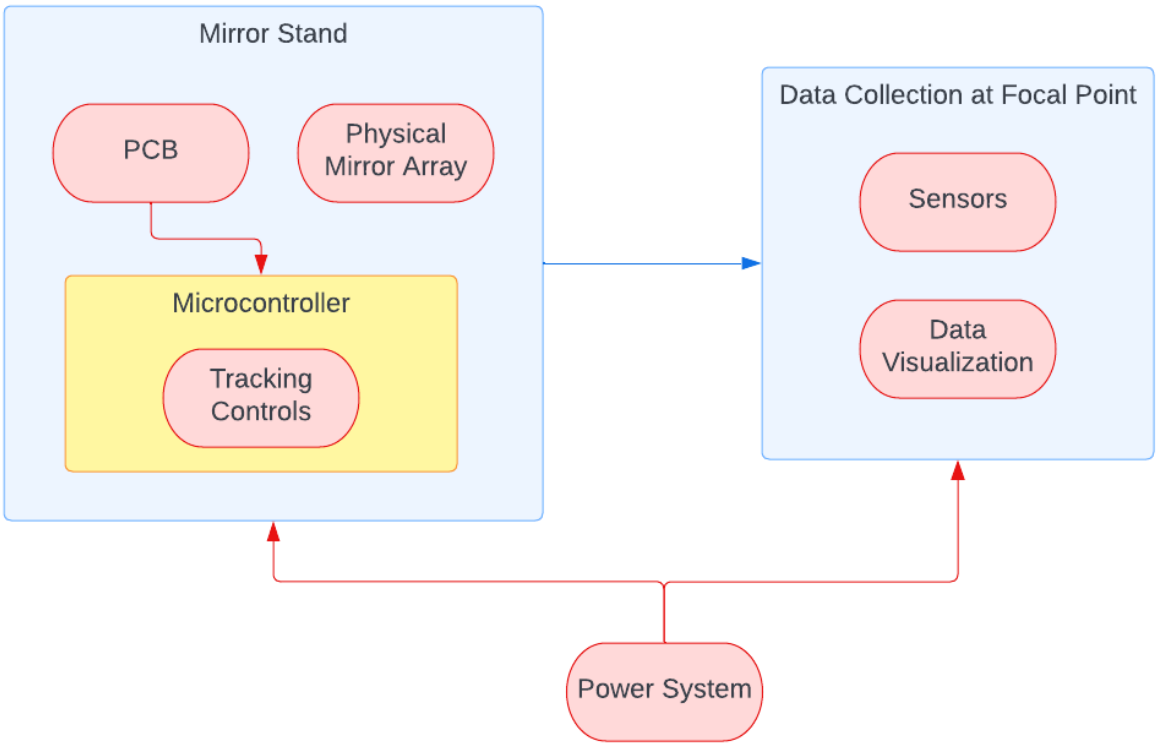


**Figure 3. System Relation Diagram**

The relational diagram in Figure 3 provides the relationship between the subsystems of the project. Items contained in the respective blue boxes are groups that correspond to each other symbolically and physically. On the left hand side, the PCB shield interfaces the capabilities of the Arduino microcontroller, which serves as the brain for the tracking control of the mirrors. The mirror array and hardware are physically connected together routing light to the right hand of the diagram, the target. At the target, there will be a photodiode with an optic filter for providing feedback to the microcontroller. The target also provides the source of the data for the visualization portion of the code.

## *3.2. Characteristics*

### 3.2.1. Functional / Performance Requirements

#### 3.2.1.1. Tracking the Sun

The Heliostat tracking system will adjust motors every ten minutes and record ten data points from the photodiode for an average measurement.

*Rationale: This is the core system performance requirement. Operating conditions occur in Texas A&M Research Park (30.6280, -96.3344).*

#### 3.2.1.2. Data Collection and Processing

The Heliostat tracking system will process the averaged sampled data points from the photodiode. The photodiodes voltage will be communicated serially with the Arduino. Upon receiving the measurements and doing the averaging, the Arduino will use the HC-05 bluetooth module package to send information in packets through bluetooth (if master-slave pair available) to an android phone application. From the phone application, the most recent reading will be displayed in an immediate view. Additionally, the reading will be uploaded through the internet to a web server, utilizing the firebase framework. When applicable, this should be done every 10 minutes when a new reading and average is taken in the program.

### 3.2.2. Physical Characteristics

#### 3.2.2.1. Volume Envelope

The volume envelope of the Heliostat Tracking Control System shall be less than or equal to 60 inches in height, 48 inches in width, and 48 inches in length.

#### 3.2.2.2. Mounting

There will be nine 12 inch by 12 inch planar mirrors that will be mounted to a frame. This frame rotation and tilt capabilities. They will be mounted with epoxy and 3-d printed fasteners.

### 3.2.3. Electrical Characteristics

#### 3.2.3.1. Inputs

The inputs of the Heliostat Tracking system include the light given from the sun as well as data from a csv file that includes the elevation and azimuth of the sun throughout the day depending on the latitude of College Station and the date of the year. This data will determine the tilt and rotation of the mirror array.

*Rationale: This system is meant to output the optimal amount of solar waves so it is required that the frame is angled to best serve this purpose.*

#### 3.2.3.1.1 Power Consumption

The maximum peak power of the system shall not exceed 84 watts.

*Rationale: This is a requirement specified by our customer due to the value of the pre-existing battery (12 V, 6A rated) within the system.*

#### 3.2.3.1.2 Input Voltage Level

The main battery of the Heliostat Tracking System can supply up to 12 Volts.

### 3.2.3.2. Outputs

The output of the Heliostat Tracking System is the positioning of the mirrors, which is realized through motor control and voltage inputs. Also, the direct normal irradiance will be measured with a rotating shadowband on a pyranometer.

#### 3.2.3.2.1 Data Output

The Heliostat Tracking Control System will contain an android application that will have a view where the customer can obtain information about the readings of the sensors. This will be crucial in ensuring the system is operating as intended and to validate that the product offered is within the operating guidelines and functional requirements specified above.

#### 3.2.3.2.2 Diagnostic Output

The Heliostat Tracking Control System will contain an android application that will have a GUI view where the customer can externally debug and test certain functionalities, rather than adjustments being made by the program. This external interface will allow the customer to rotate the two motors to move the mirror array translationally and rotationally. Additionally, the application will provide verification of connectivity and option to halt system execution.

*Rationale: Provides the ability to debug the system and manually control the mirrors for rotating the mirror around two different axis points.*

## 3.2.4. Environmental Requirements

The Heliostat Tracking System shall be designed to withstand and operate in the environments and laboratory tests except for days with heavy rain, days that are very cloudy, and throughout the night.

*Rationale: This is because the system is meant to track the sun so when the sun is not in the sky or is blocked heavily, then the heliostat would not be optimal in these conditions.*

### 3.2.5. Failure Propagation

The Heliostat Tracking System will not allow propagation of error to occur in the system beyond the visual interface. For software, when the sensor consistently is off with respect to the ideal value ( determined by experimental testing ), the system interface will alert that there are either non-ideal environmental conditions occurring or a physical bug in the system. In hardware, there will be an overcurrent detection device to ensure that the pins of the esp-32 are not being strained past their operating conditions. Additionally, relative location of the esp-32 will be discussed to prevent strain on the mechanical system. Additionally, a safety switch will provide extra security for stopping the process. This is critical for devices involving mechanical motion, where objects can become twisted and broken.

# 4. Support Requirements

Provided support for this project include an activity page within the android application, involving purpose statement, navigation, and change date. Additionally, there will be a feedback form within the application, where users can query for support to the service team. The service team will be able to consult subsystem owners for expertise regarding specific questions and provide valuable feedback in a timely manner to the customers as needed. The requirement to access this technical support would be a device that utilizes google's android operating system, can download applications from the play store, and has the capability of connecting to the internet.

# Appendix A: Acronyms and Abbreviations

| | |
|---|---|
| BIT | Built-In Test |
| CCA | Circuit Card Assembly |
| DNI | Direct Normal Irradiance |
| EMC | Electromagnetic Compatibility |
| EMI | Electromagnetic Interference |
| EO/IR | Electro-optical Infrared |
| FOR | Field of Regard |
| FOV | Field of View |
| GPS | Global Positioning System |
| GUI | Graphical User Interface |
| Hz | Hertz |
| ICD | Interface Control Document |
| kHz | Kilohertz (1,000 Hz) |
| LCD | Liquid Crystal Display |
| LED | Light-emitting Diode |
| mA | Milliamp |
| MHz | Megahertz (1,000,000 Hz) |
| MTBF | Mean Time Between Failure |
| MTTR | Mean Time To Repair |
| mW | Milliwatt |
| PCB | Printed Circuit Board |
| RMS | Root Mean Square |
| TBD | To Be Determined |
| USB | Universal Serial Bus |

# Appendix B: Definition of Terms

| | |
|---|---|
| HC-05 | Bluetooth Module |

# Appendix C: Interface Control Documents

The Interface Control Document is attached below

# Heliostat Control Tracking

Samuel Dixon
Jordan George
Erica Quist

## SCHEDULE

| | Oct. 5th | Oct. 19th | Nov. 2nd | Nov. 16th |
|---|---|---|---|---|
| **PCB Design** | Get Altium Schematic Drawn Up | Get Altium layout completed and pass all DRC checks. Order PCB and components | Assemble the voltage regulator part of the PCB and test varying input voltages. | Final soldering, testing, and debugging of the PCB. |
| **Data Visualization** | Investigate and begin development android packages for visuals | Save and store data in arduino and map out front end app | Program manual control functionality with firebase database | Final testing of application and additional tweaks |
| **Tracking Control** | Understand ESP32 configuration, research sun tracking equations | Align motors according to given input, locate the sun in the sky, model the appropriate angle the mirror should point | Move the motors in accordance with the sun while taking in sensor data | Setup a “daily” script |
| **Mirror Array** | Order mechanical parts | Put together parabolic array | Attach to overall frame | Test for vibrations |

# Heliostat Control Tracking

Samuel Dixon
Jordan George
Erica Quist

# VALIDATION PLAN

| Status Indicators | |
|---|---|
| Completed | |
| On Schedule/ In Progress | |
| Behind Schedule | |

| Task | Deadline | Current Status | |
|---|---|---|---|
| PCB Subsystem | | Completed | |
| Get an understanding of the desired functionality of the PCB | Sept. 28 | Completed | |
| Draw up schematic in Altium | Oct. 5 | Completed | |
| Order components | Oct. 5 | Completed | |
| Design PCB layout in Altium and pass all design rule check tests | Oct. 19 | Completed | |
| Create gerber files and NC drill files to order PCB | Oct. 19 | Completed | |
| Solder on voltage regulator part of the circuit for test for a constant 3.3V output based on a range of input voltages | Nov. 2 | Completed | |
| Assemble the rest of components on the board for final testing and debugging | Nov. 16 | Completed | |
| User Interface / Data Visualization Subsystem | | Completed | |
| Download Android Studio and investigate examples for programming in java | Oct. 5 | Completed | |
| Front End of Android Application 2/4 front end activity pages complete | Oct. 12 | Completed | |
| Front End of Android Application all front end activity pages complete | Oct. 19 | Completed | |

| | | | |
|---|---|---|---|
| Prototype Wifi communication on Application for data reading and Investigate Web Server for storage, and research Optometrika | Oct. 27 | Completed | |
| Integrate firebase database through wifi on device | Nov. 3 | Completed | |
| Debug application, test on phone, and programmatically check for wifi connection | Nov. 11 | Completed | |
| Integrate firebase database through wifi and api key on device | Nov. 18 | Completed | |
| Test reading, writing, and simple motor control utilizing database flag section as ISR routine | Nov. 24 | Complete | |
| Tracking Controls Software and Sensor Interface | (Project Member Discontinuation) | (Project Member Discontinuation | |

**Performance on Execution Plan:**

The performance on the execution plan is as follows. The PCB system design was drawn up, soldered, and tested. The Data Visualization system design was completed, constructed, and tested. The Tracking Controls Software and Sensor Interface was discontinued, as the last teammate was unable to continue with this project. The other two subsystems were followed through execution according to the schedule given in the format above. However, for the data visualization subsystem some advice was given to revise the current organization to an alternative database.

**Performance on Validation Plan**

The performance on the validation plan is indicated by the status markers in the table above. The tasks involving the two subsystems, PCB and data visualization, were able to be validated according to the plan above. After demoing, a few more tests needed to be run for the PCB subsystem. These tests included testing the noise levels for the voltages as well as testing the voltage with a varying load. Also, the data visualization sub system needs some modification in terms of the database which will be investigated during the break. Additionally, the last subsystem, tracking controls and sensor interface,  was not able to be validated by the end of the system. Therefore, there will be more work to be done before integration in the next portion of the project scope.

# Heliostat Control Tracking

Samuel Dixon
Jordan George
Erica Quist

## Subsystem Report

REVISION – Draft A
December 2, 2022

# Subsystem Report
FOR
# Heliostat Control Tracking

Team <66>

Approved by:

| | |
|---|---|
| Jordan George | 12/2/2022 |
| Project Leader | Date |
| Dr. Kalafatis | 12/2/2022 |
| Prof. Kalafatis | Date |
| Dalton Cyr | 12/2/2022 |
| T/A | Date |

## Change Record

| Rev. | Date | Originator | Approvals | Description |
|---|---|---|---|---|
| **A** | 12/2/2022 | Samuel Dixon, Jordan George | | Draft Release |

# Table of Contents

# Table of Contents

# List of Tables

# List of Figures

# 1. Introduction

This project intends to develop the control methodology and design behind an energy application device known as a Heliostat. The device seeks to track the rotating motion of the sun throughout the day and rotate some mirrors to focus light onto a thermal plate for further energy conversion. The subsystems in this project entailed design and fabrication of a PCB, development of a user interface and data visualization, and tracking controls software for the microcontroller.

# 2. PCB Subsystem Report

## 2.1. Subsystem Introduction

The PCB subsystem intends to service the power to the microcontroller as well as provide a common interface for uploading code to the ESP32 Wroom through a micro USB connector as well as being able to access the GPIOs of the ESP32 Wroom for the various sensors of this project. The PCB will be powered by a 12V / 7A battery. There will be a voltage regulator part of this circuit to step down the 12V to 3.3V as the microcontroller and USB-UART bridge require around 3.3V to operate.

## 2.2. Subsystem Details

The PCB subsystem was created with the help of the PCB design software called Altium. Below is the schematic drawing of the PCB:

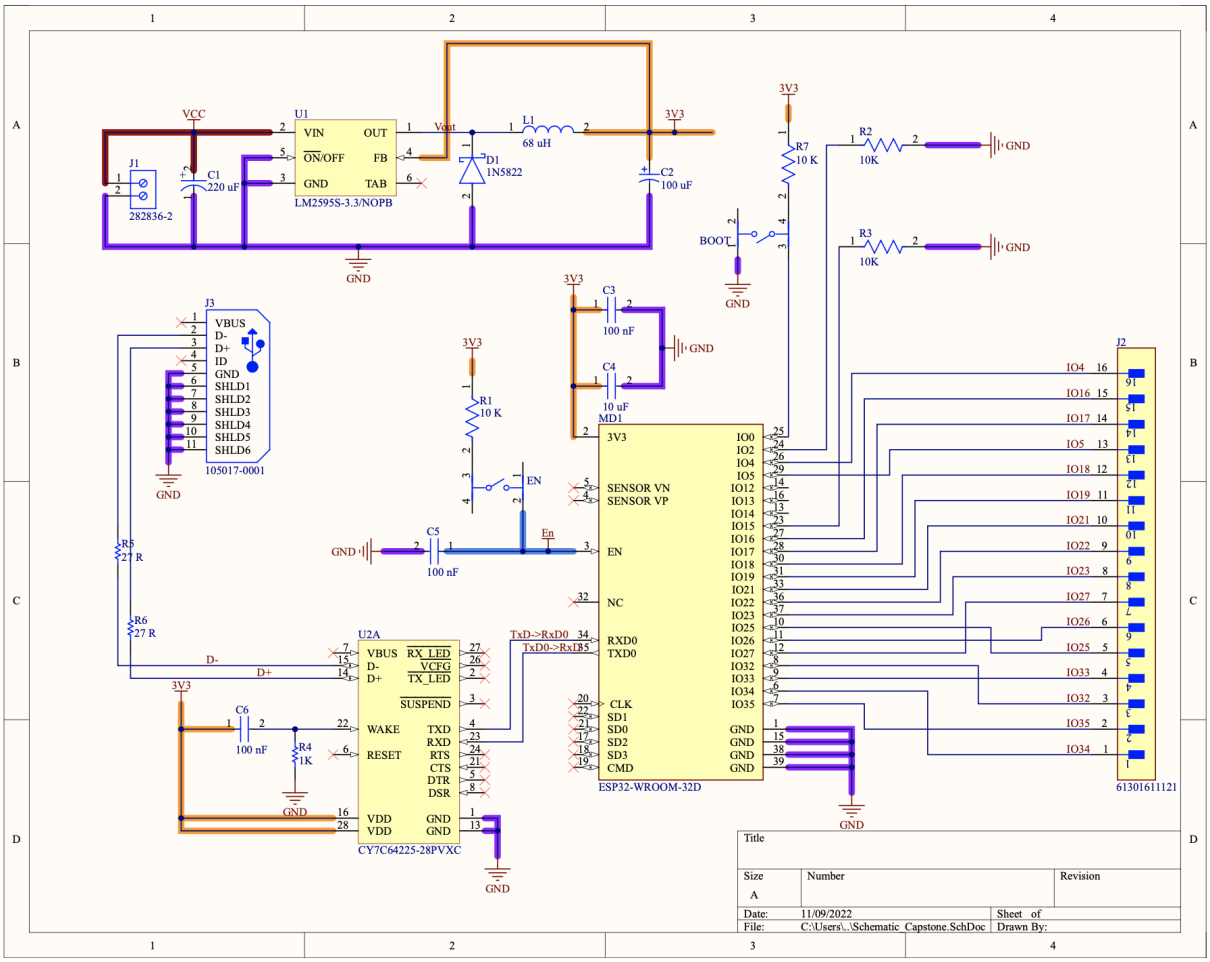


*Figure 1: PCB Schematic*

In the top left corner of the schematic is the voltage regulator that will step down the input voltage of 12V to 3.3V for the rest of this schematic. The input voltage of 12V is color coded by the dark red lines. The orange colored lines represent the 3.3V node while the purple colored lines represent the common ground node. The datasheet of the ESP32, voltage regulator, and the USB-UART bridge were read through to help best set up this schematic.

On the right side of the schematic are the 16 header pins that will be used for the multiple GPIOs of the ESP32. These will be hooked up to the different sensors in this project. In order to turn the ESP32 into bootloader mode, the BOOT button must be held down and shortly after, the EN button will be pressed. After, both buttons may be released. The next part of design process was the PCB layout pictured below:

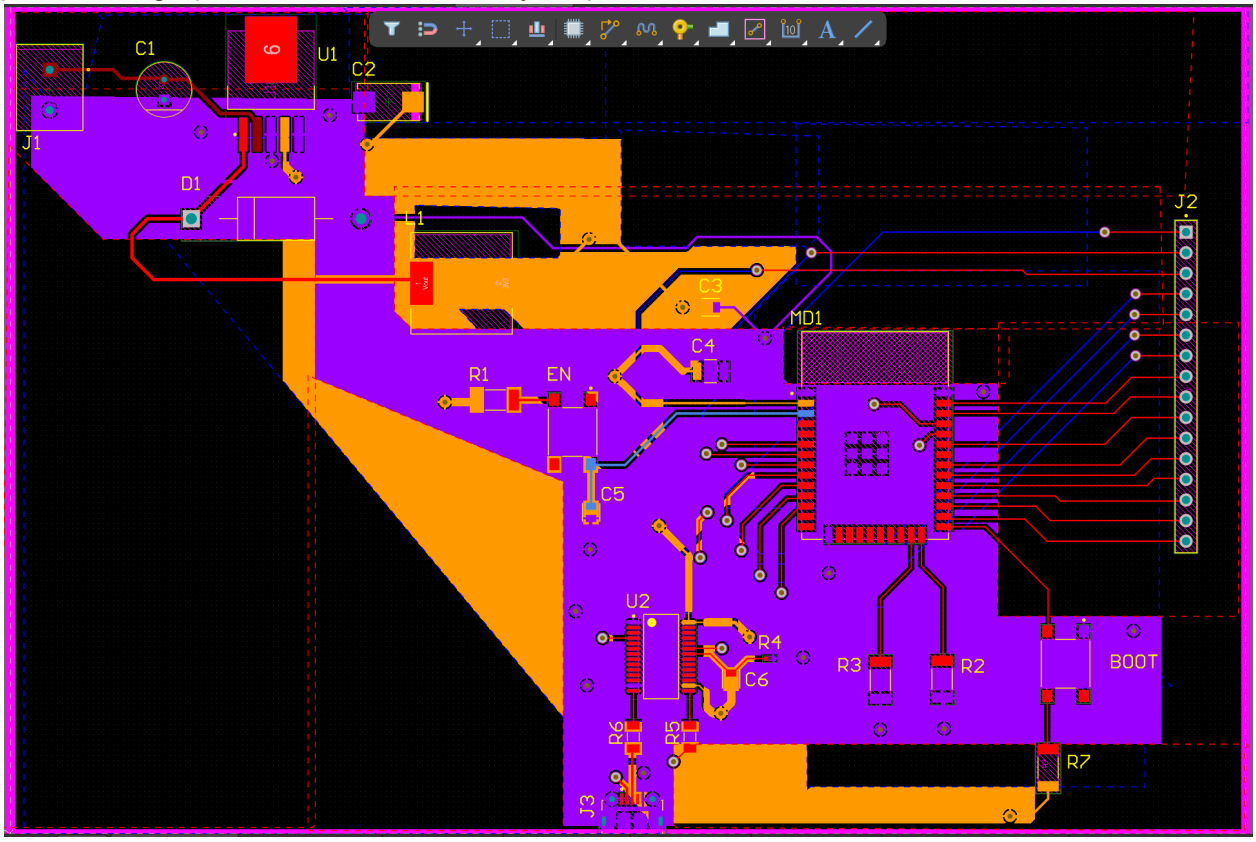


*Figure 2: PCB layout*

This layout starts with the screw terminals named J1 in the top left of this picture. This is where the input power will be plugged into. The board was able to be designed as a 2-layer board. There are two polygon pours to create a common node for needed voltages. The orange polygon pour is on the bottom layer and is the 3.3V net, while the purple polygon pour is on the top layer and is the common ground net. This Altium design was sent to JLCPCB for manufacturing. Once the board got here, all the components were soldered where they were designed to be, and it resulted into the following.

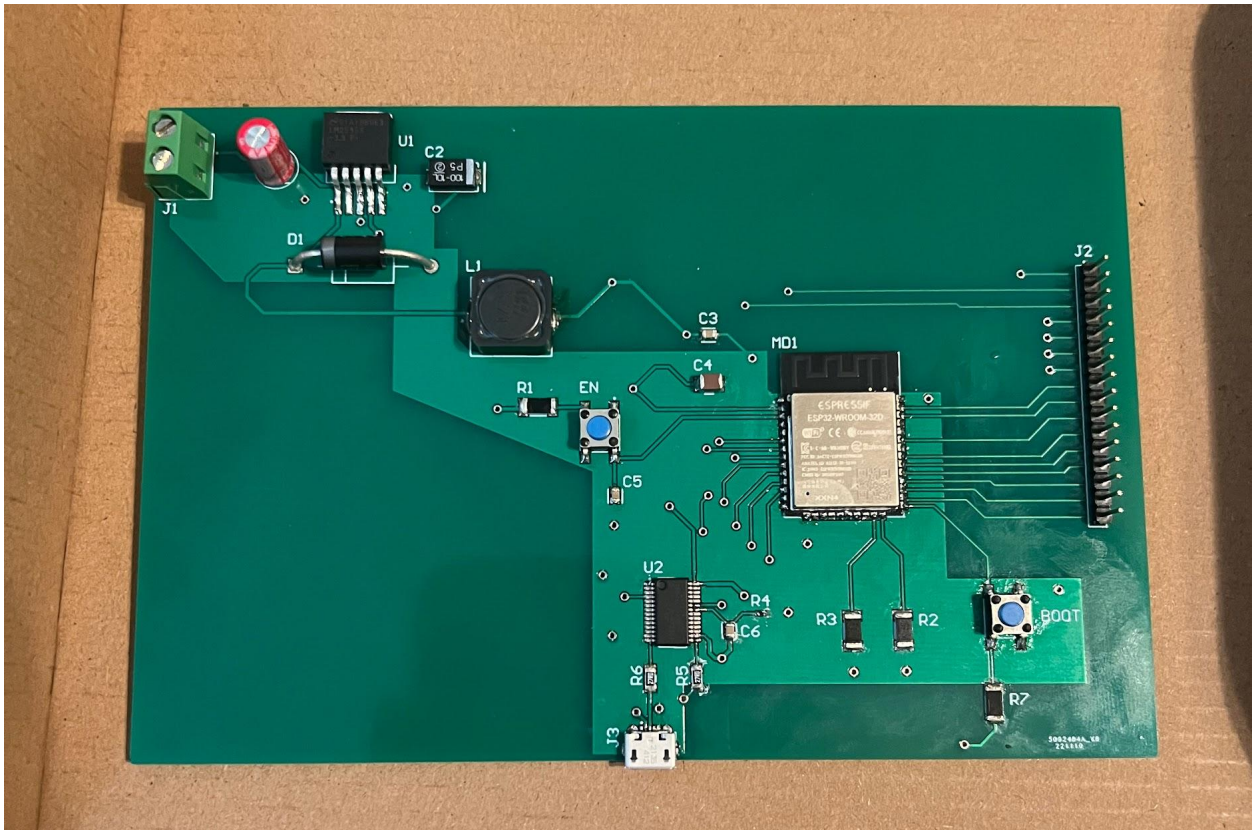


*Figure 3: PCB with components soldered on*

## 2.3. Subsystem Validation

The voltage regulator chip went under a lot of testing to see if it could hold a constant output of 3.3V. The first thing that was tested was varying the value of the input voltage.

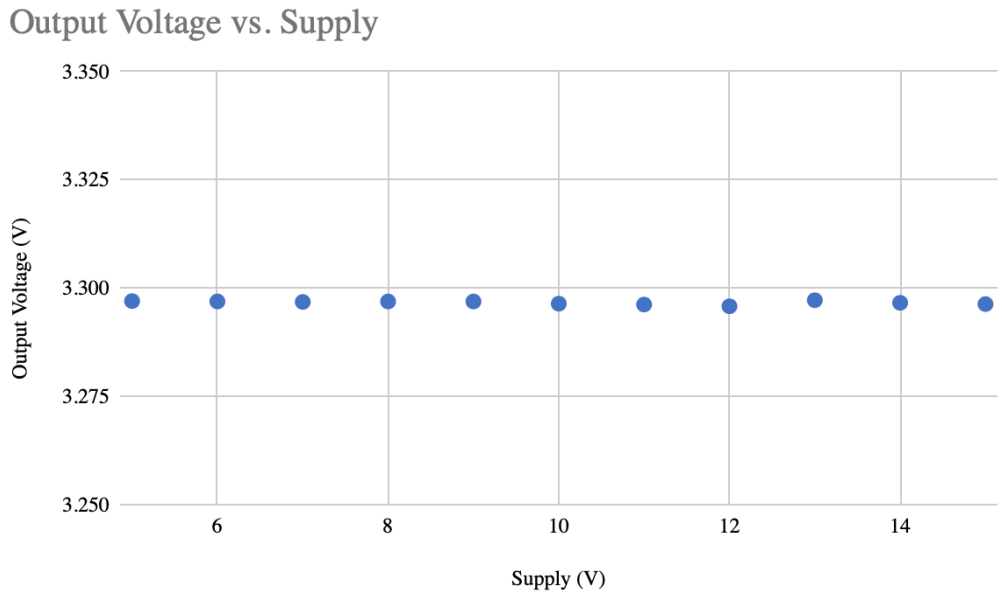


*Figure 4: Output Voltage vs Supply of Voltage Regulator*

As shown above, the output constantly holds slightly under 3.3. The nominal voltage of this experiment is 12V so this is working as desired, since the ESP32 and the UART bridge both operate at 3.3V. The next thing that was tested was by varying the load of the voltage regulator to see if the output still holds at 3.3V. The input voltage for these tests was held at a constant 12V.

| Varying Load Tests | | | |
|---|---|---|---|
| **Supply (V)** | **Regulator Output (V)** | **Load being tested (Ohms)** | **Current At the Load (A)** |
| 12 | 3.2958 | Open circuit | 0.006 |
| 12 | 3.263 | 15.83980583 | 0.206 |
| 12 | 3.227 | 7.665083135 | 0.421 |
| 12 | 3.193 | 5.295190713 | 0.603 |
| 12 | 3.155 | 3.933915212 | 0.802 |
| 12 | 3.122 | 3.144008056 | 0.993 |

*Table 1: Voltage Regulator with varying loads*

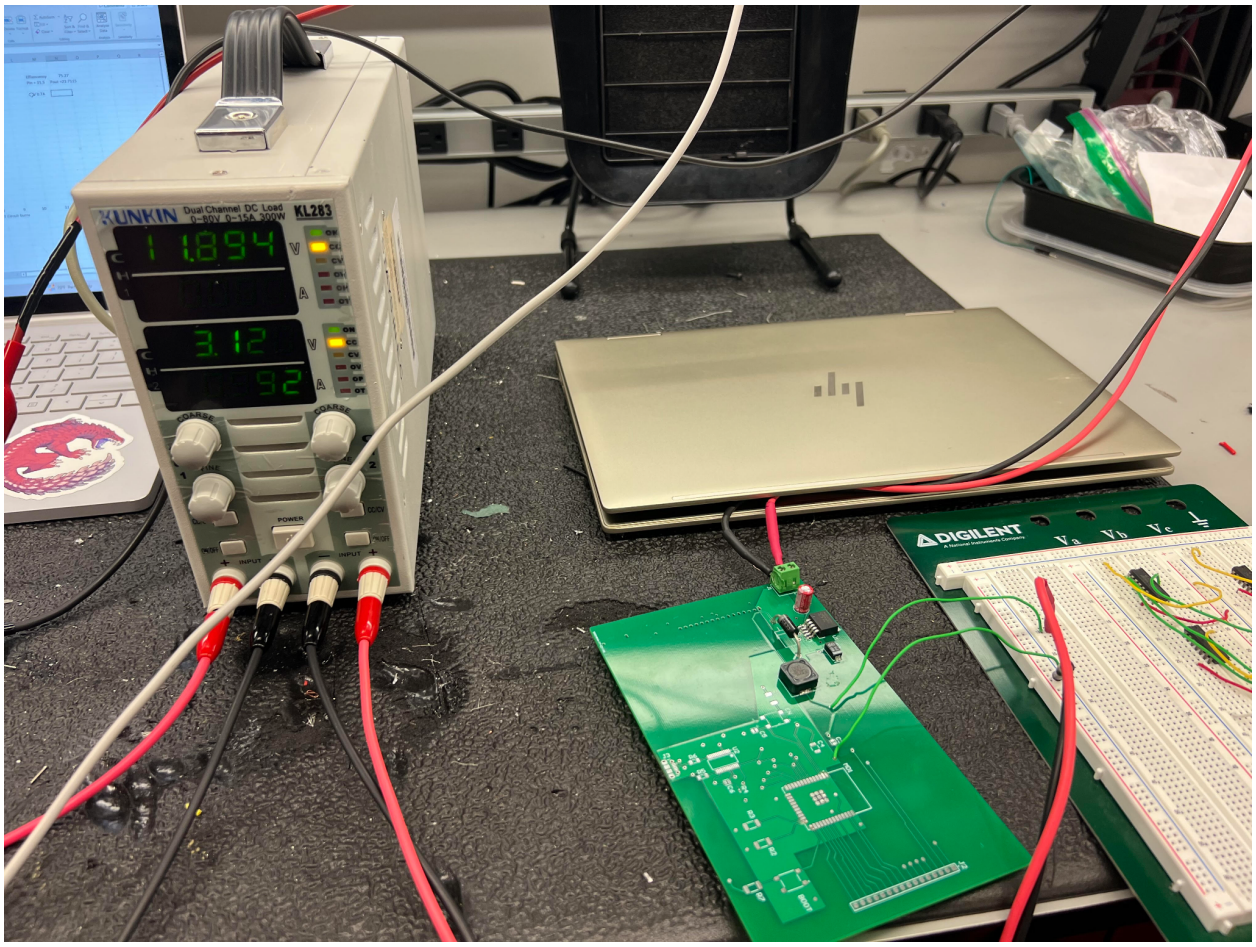


*Figure 5: Voltage Regulator with varying loads*

The max output current of the voltage regulator is 1 Amp so various data points were taken as the current drawn approached 1A. From Table 1, it can be seen that the voltage does

start to decrease slightly as the load decreases. The next set of data that was recorded was the efficiency of the voltage regulator. This was done by using one of the data points from Table 1, specifically the third row of the table. The input current was recorded by seeing how much current was being drawn from the power supply which was about 0.152 Amps.

| Power In (W) | Power Out (W) | Efficiency |
|---|---|---|
| 1.824 | 1.358567 | 0.7448283991 |

*Table 2: Efficiency of the Voltage Regulator*

The next set of testing was done on the circuit board that had all the components soldered on (Figure 3). First, it was tested to see if the input voltage would be stepped down to provide 3.3V to the ESP32. When the ESP32 was not in bootloader mode, almost no current was being drawn from the power supply. So the data in the following table was recorded while the ESP32 was on.

| ESP32 when turned on | | |
|---|---|---|
| **Supply (V)** | **ESP32 Voltage (V)** | **Current Drawn From Supply (A)** |
| 5 | 3.2889 | 0.103 |
| 6 | 3.2893 | 0.086 |
| 7 | 3.29 | 0.08 |
| 8 | 3.3018 | 0.066 |
| 9 | 3.303 | 0.059 |
| 10 | 3.3022 | 0.057 |
| 11 | 3.2977 | 0.049 |
| 12 | 3.2978 | 0.045 |
| 13 | 3.2975 | 0.042 |
| 14 | 3.2968 | 0.04 |
| 15 | 3.2967 | 0.038 |

*Table 3: Voltage at ESP32 with a varying input*

This table shows that the circuit was working as expected. The output voltage would hold at a constant 3.3V which means the orange polygon pour from Figure 2 was at this voltage. This showed that the ESP32 and the USB-UART bridge were both being powered at 3.3V which is within their operating range. The last set of data recorded was taken by using the oscilloscope to measure the input and output voltage to check for any noise. Below are the pictures of what the oscilloscope showed for the input voltage of 12V and the output voltage of 3.3V.

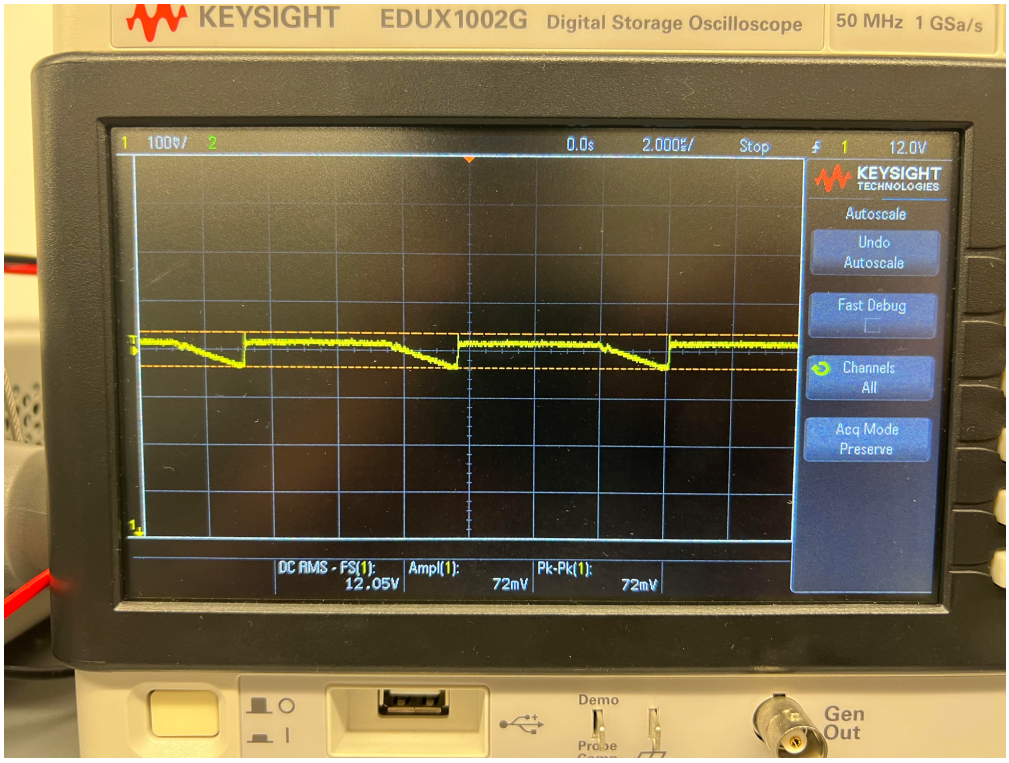


*Figure 6: Oscilloscope Image of the Input Voltage (12V)*

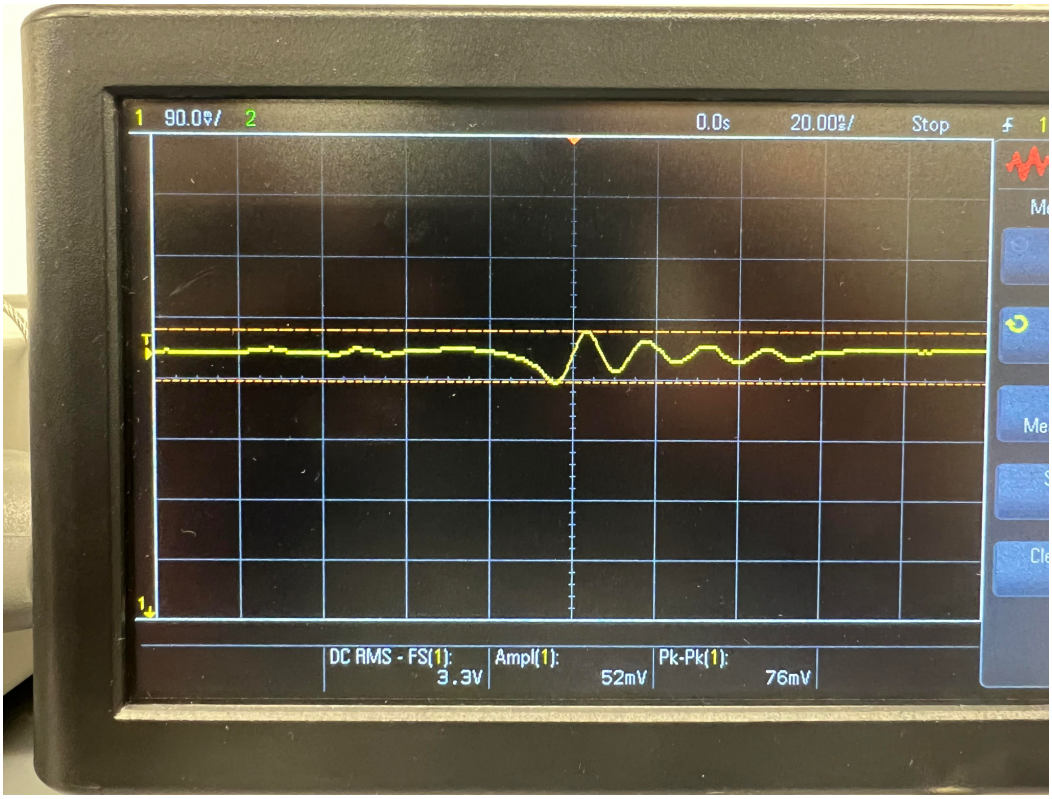


*Figure 7: Oscilloscope Image of the Output Voltage (3.3V)*

### *2.4. Subsystem Conclusion*

From the validation tests shown in the section above, the PCB subsystem works as planned. This shows that the 12V battery that is being used for the whole project can be stepped down to power the ESP32 and USB-UART bridge. With the ESP32 and USB-UART bridge being operational, code can be written to these devices that will be able to control the motors of the project.

# 3. Data Visualization and User Interface

## *3.1. Introduction*

The data visualization and user interface seeks to provide human machine interaction to the heliostat energy system. This interaction is done through use of a database, a web application, and a cellphone application. The database provides flags, which enable the user to interact with the motors in the heliostat on the cellphone application via wifi communication. The web application displays various sensor and timestamp readings for specific sensors from the database.

## *3.2. Subsystem details*

This subsystem involved creating a cell phone and web application to provide a front-end visualization for the sensor readings stored in the online database of the system. This was done through using the google software suite. This was a convenient choice as it integrates their NoSQL non-relational, real-time database with android applications. Additionally, this software provides support for hosting websites on the cloud.

**Heliostat Controls Web Data**

| | Sensor | Reading | Date |
|---|---|---|---|
| 1 | Photodiode1 | 25 | Tuesday, November 29 2022 08:48:33 |
| 2 | Magnetometer2 | 83 | Tuesday, November 29 2022 08:48:33 |
| 3 | Magnetometer1 | 15 | Tuesday, November 29 2022 08:48:33 |

*Figure 8 Website Table*

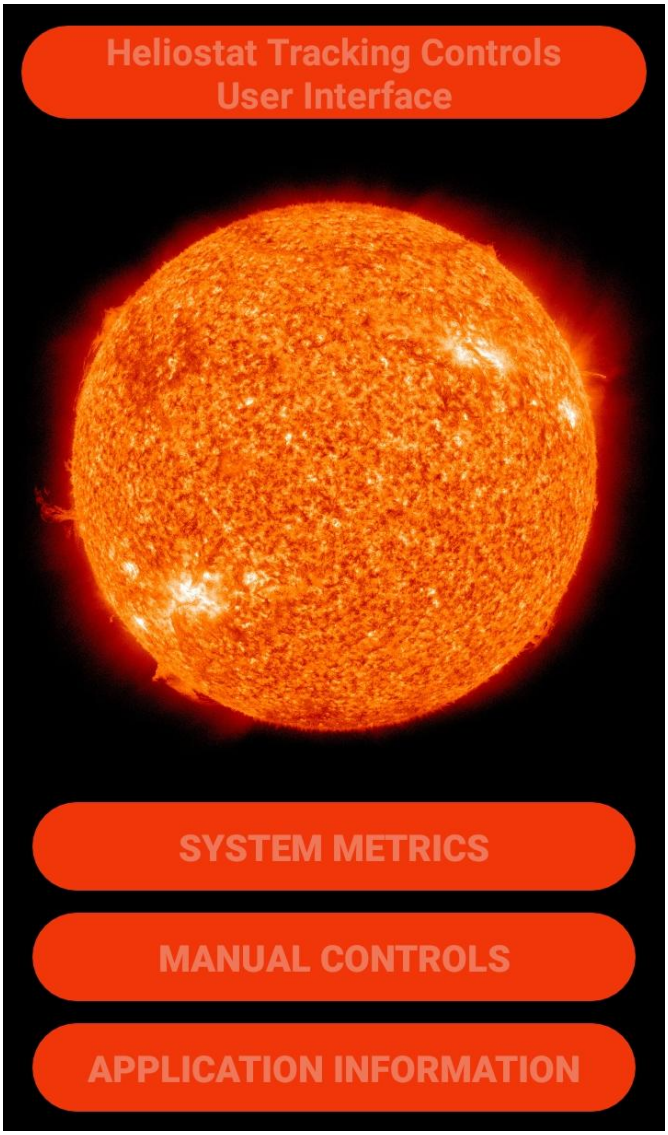


*Figure 9: Main Activity Page*

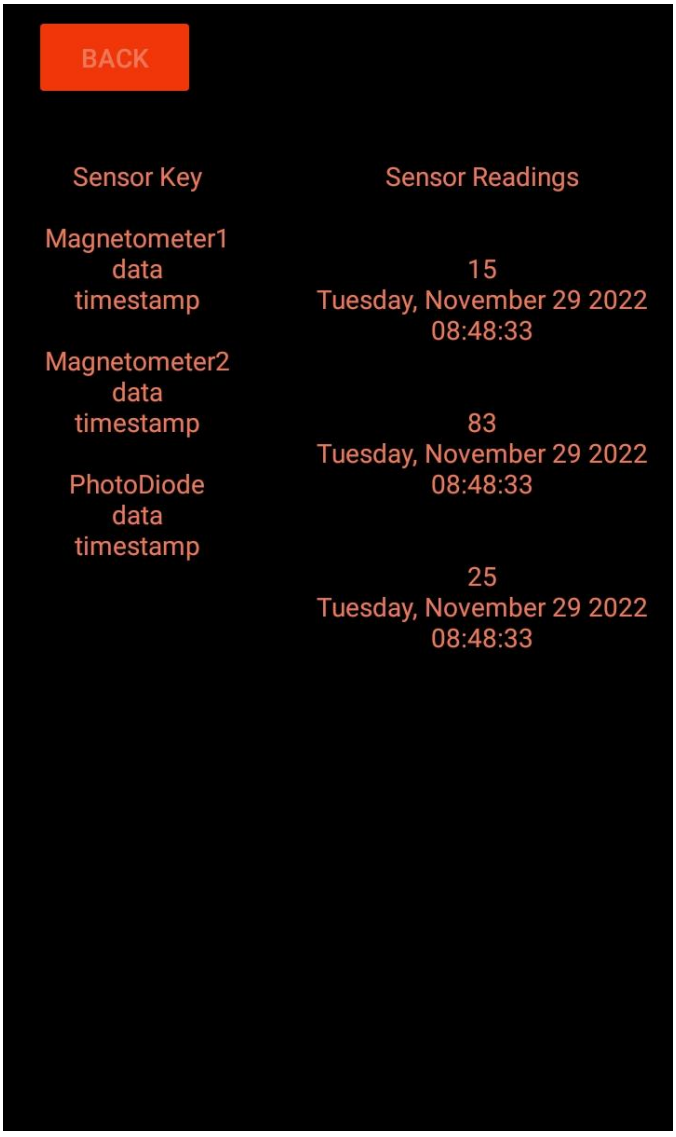


*Figure 10: System Metrics*

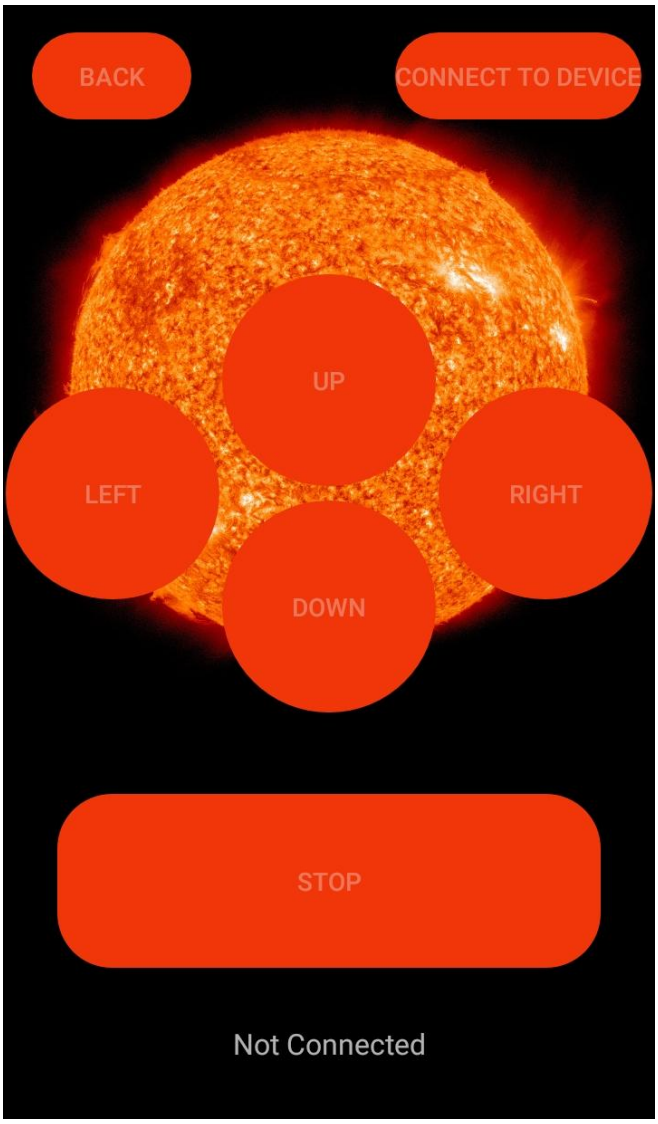


*Figure 11 Manual Controls*

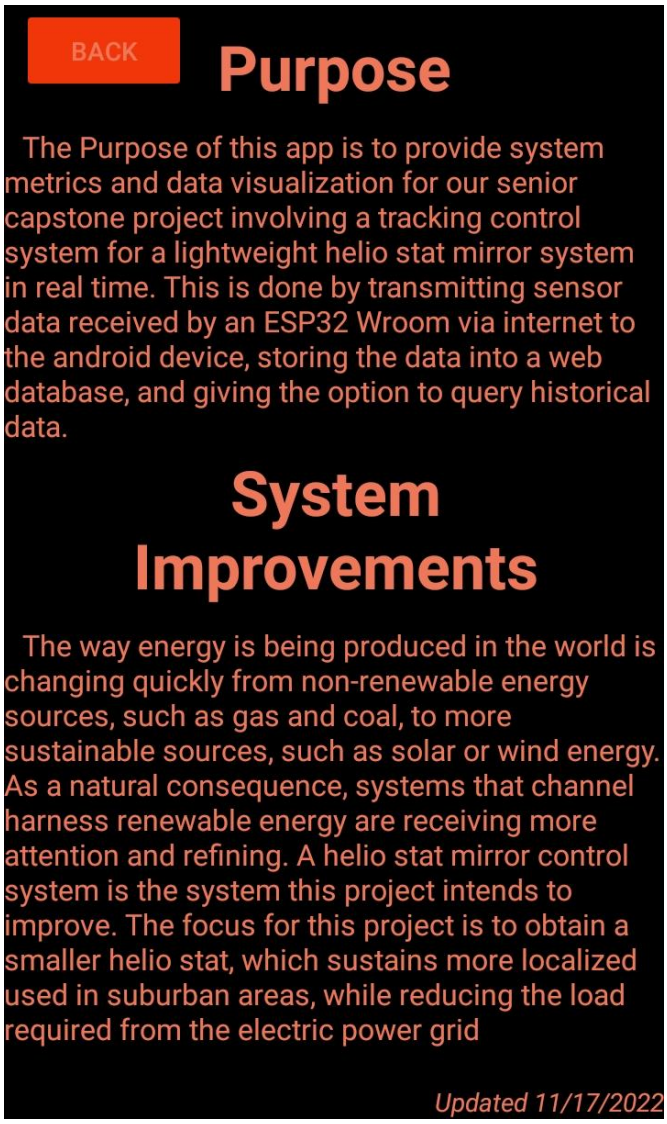


*Figure 12: Application Information*

### *3.3. Database*

The firebase real-time database was used to store sensor data and service communication between the microcontroller and cellphone application. This database is a NoSQL non-relational database, which assumes a tree-like structure. Under the main parent node, there are two child nodes, Flags and Sensors. Flags represent the boolean flags, which can serve as service routine requests between the phone application and the esp32 microcontroller. Sensors contain all of the stored data sent from the esp32, which enables visualizing data analytics in both the web and phone application.

### 3.4. Phone Application

The phone application will provide user interface and data visualization for the system. The user interface is within the manual controls activity and provides users the opportunity to manual control the motors. There are two motors that have two directions of motion, so there are consequently four buttons to control the motors. Additionally, there is a connect to device button that acts as a handshake between the esp32 microcontroller and the phone application. These buttons all act through flags within the NoSQL database, and the esp32 is continuously querying these flags to change state. The data visualization is within the system metrics activity and provides users the opportunity to see the sensor readings in a text view display. This data comes from the Sensors node of the database as the most recent readings. Further room could be to have historical graphs within the application, rather than instantaneous readings.

### 3.5. Web Application

The Web Application is a simple website to also display the instantaneous data readings for the sensors of the system. This website is integrated with the database and the phone application within the firebase google software suite and is deployed on google's cloud hosted servers. The website involves html, javascript, and css, which use google's data visualization API to store the sensor data, which is a simple table seen in the figure below. This table also allows for column alphanumeric sorting, seen through the tabs at the top with the arrows.

### 3.6. Validation

Validation for these applications was performed by adding additional code to ensure the system functioned as intended. For example, in the phone application, in the system metrics and manual controls page, there must be an established wifi connection to interact with the database. This is seen in *Figure 13* below.

```
if (!connect) { // if no wifi, create alert dialog to inform user this function is necessary for application functionality
    builder = new AlertDialog.Builder( context: this);
    builder.setMessage("Connect to wifi or quit")
            .setCancelable(false)
            .setPositiveButton( text: "Connect to WIFI", (dialog, id) -> startActivity(new Intent(Settings.ACTION_WIFI_SETTINGS)))
            .setNegativeButton( text: "Quit", (dialog, id) -> openMain());
    AlertDialog alert = builder.create(); // creating alert builder
    alert.show(); // showing the content of the alert builder
}
```

```
private boolean haveNetworkConnection() {
    // initialize booleans assuming there is no wifi connection
    boolean haveConnectedWifi = false;
    boolean haveConnectedMobile = false;
    // Open connectivity manager and get the context for the system's wifi service
    ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
    // Create an array of network info to store the data from the connectivity manager getAllNetworkInfo method
    NetworkInfo[] netInfo = cm.getAllNetworkInfo();
    for (NetworkInfo ni : netInfo) {
        if (ni.getTypeName().equalsIgnoreCase( anotherString: "WIFI")) // Ignoring case (non case-sensitive) check for wifi
            if (ni.isConnected())
                haveConnectedWifi = true; // retract assumption, as network method informs otherwise
        if (ni.getTypeName().equalsIgnoreCase( anotherString: "MOBILE")) // Ignoring case (non case-sensitive) check for Mobile
            if (ni.isConnected())
                haveConnectedMobile = true; // retract assumption, as network method informs otherwise
    }

    if (!(haveConnectedWifi || haveConnectedMobile)) {
        Toast.makeText( context: ManualControls.this,  text: "Must connect to wifi to use this activity page", Toast.LENGTH_SHORT).show();
    }

    return haveConnectedWifi || haveConnectedMobile; // returning or for the two boolean states, as either implies a network connection
}
```

*Figure 13: Programmatically checking for internet connection*

Additionally, the full program flow and transition between activities and loading in the data from the database was confirmed through running the emulator in the android studio software suite and loading the application onto an android device. For the flag routine, this was done by first checking if the node existed in the database, and then ensuring that the node was of an integer type. This ensured that the correct data type was being written into the database and no type incompatibility problems ensued. In *Figure 14* this is seen below. The flag is first validated to be the correct type and then queried to get the integer at the database reference location. In *Figure 15*, a method was implemented to callback the table population function and create a new table that will allow for the updated sensor readings to be observed.

```
if (Firebase.RTDB.getInt(&fbdo, "Flags/Connect")) { // make sure that the connection flag has been set
  if (fbdo.intData() == 1) {  // if the reference is 1, then we can do movement
    if (Firebase.RTDB.getInt(&fbdo, "Flags/Right")) { // now accessing the firebase database object and setting the right flag
      if (fbdo.intData() == 1) { // if the value of the flg is 1

        Serial.println("right == 1 , clockwise"); // print serial data and note direction of motor

        Firebase.RTDB.setInt(&fbdo, "Flags/Right", 0); // reset the right flag to be 0

        myStepper.step(stepsPerRevolution); // use stepper module to actuate the movement of the motor

        Firebase.RTDB.setFloat(&fbdo, "Sensors/Magnetometer1/reading/data", random(1,100)); // testing a random datapoint to be set to the database
        update_time("Magnetometer1"); // call subroutine to update the time of the magnetometer1 sensor

        Firebase.RTDB.setFloat(&fbdo, "Sensors/Magnetometer2/reading/data", random(1,100)); // testing a random datapoint to be set to the database
        update_time("Magnetometer2"); // call subroutine to update the time of the magnetometer2 sensor

        Firebase.RTDB.setFloat(&fbdo, "Sensors/PhotoDiode/reading/data", random(1,100)); // testing a random datapoint to be set to the database
        update_time("PhotoDiode"); // call subroutine to update the time of the photodiode sensor

        Firebase.RTDB.setInt(&fbdo, "Flags/Right", 0);  // otherwise print 0 to indicate no movement

      } else {
        Serial.println("right == 0, no movement"); // otherwise we do not move and serial print to the screen
        delay(500); // delay 500 ms so that we can see what is happening logically
      }
```

*Figure 14: Testing to write to firebase reference by first checking if getting an integer from flag returns true*

```
onValue(sensors, (snapsot) => { // creating onValue to  reference the parent nodes of the sensors

    const sensor1 = ref(db, "/Sensors/Magnetometer1/reading/"); // getting reference to the reading of sensor1
    var d1, d2, d3, x, y, z; // instantiating variables to hold the values for the sensor data

    onValue(sensor1, (snapshot) => { // when any child of sensor1 changes, get new data
        x = snapshot.val()["timestamp"]; // child node key-value pair get time stamp
        d1 = snapshot.val()["data"]; // child node key-value pair get the data
    });


    const sensor2 = ref(db, "/Sensors/Magnetometer2/reading/"); // getting reference reading of sensor2

    onValue(sensor2, (snapshot) => { // when any child of sensor2 changes, get new data
        y = snapshot.val()["timestamp"]; // child node key-value pair get time stamp
        d2 = snapshot.val()["data"]; // child node key-value pair get data
    });

    const sensor3 = ref(db, "/Sensors/PhotoDiode/reading/"); // getting reference to the reading of sensor3

    onValue(sensor3, (snapshot) => { // when any child of sensor2 changes, get new data
        z = snapshot.val()["timestamp"]; // child node key-value pair get time stamp
        d3 = snapshot.val()["data"]; // child node key-value pair get data
    });

    google.charts.load('current', { 'packages': ['table'] }); // load the google chart api table
```

*Figure 15: Implementing callback method to update the chart every time the sensor reading was changed*

## *3.7. Diagnostic and Mitigation*

For the sake of this project, the historical capture was determined not to be sufficient. There is no way currently to see a visualization of the previous sensor data. This will be implemented by adding a graph view page into the phone application. Additionally, the database organization has been determined to be insufficient for the sensor readings of the project. Two weeks will be taken to see if a revision of the current database will be sufficient for the purposes of this project, otherwise a new approach will need to be taken through utilizing a SQL database scheme.

## *3.8. Subsystem Conclusion*

Through the validation and diagnostic and mitigation section of the report above, the subsystem has some functionality that is currently sufficient for the purposes of this project. However, there are still places where the subsystem needs to be improved for integration. This includes database organization and historical capture.