

ECE 445: Senior Design Laboratory

Fall 2021

# Advanced Interface Box for Solar Panels

Design Documentation

---

## **Team #10**

Nikhil Mathew Sebastian (nikhils4)

Sydney Li (sydneyl3)

Maram Safi (msafi2)

# Contents

<b>1 Introduction</b>	<b>1</b>
1.1 Problem Statement	1
1.2 Solution Overview	1
1.3 Visual Aid	2
1.4 High-Level Requirements	2
<b>2 Design</b>	<b>4</b>
2.1 Block Diagram	4
2.2 Subsystem Descriptions	5
2.2.1 Power Subsystem	5
2.2.2 Monitoring Subsystem	6
2.2.3 Microcontroller Subsystem	9
2.2.4 External Access Portal: Research Hub	11
2.2.5 Peripheral Parts	15
2.2.5.1 Solar Panels	15
2.2.5.2 12-Volt Power Supply	15
2.2.5.3 Thermocouples	16
2.2.5.4 Passive External Load	16
2.3 Supporting Material	17
2.3.1 Mechanical Design and Mounting Diagrams	17
2.3.2 Mechanical Design and Mounting Explanation	18
2.4 Tolerance Analysis	18
2.5 Protection Method and Analysis	19
<b>3 Cost and Schedule</b>	<b>21</b>
3.1 Cost Analysis	21
3.2 Future Schedule	24
<b>4 Ethics and Safety</b>	<b>26</b>
4.1 Ethical Issues	26
4.2 Safety Concerns	26
4.3 Safety and Regulatory Standards	26
<b>5 Conclusion</b>	<b>28</b>
5.1 Success Criteria	28
<b>References</b>	<b>29</b>

# 1 Introduction

## 1.1 Problem Statement

There are 60 solar panels on top of the ECEB building, currently used for research, which are not producing any power as of now and can potentially be integrated into the power grid. Additionally, they are not adequately monitored at the moment and this poses a large hazard, especially considering there are no protection interfaces between the panels and their connections to the power inverter.

In the Fall 2019 semester, a team of students was able to create an interface which was able to display a single panel's voltage and current, but the solution could not be scaled up to interface with multiple panels as is required [1]. But, each solar panel was directly connected to an ethernet cable that would only allow for the display of that particular solar panel onto a remote display. Not only was this a physical impracticality due to the ethernet cables, there was also no way to collectively display the data of multiple solar panel's parameters in an organized manner onto a singular remote display. Following this earlier project, the solar panels have designated box mounts (Attabox) which also gives us a constrained size 8 x 6 x 4 inches in which we must utilize in our solution to send and receive data between multiple solar panels, our interface box and external access points.

## 1.2 Solution Overview

Our solution is to design a smart interface box for these panels to allow for large-scale system behavior and output monitoring, as well as to support panel up-keep, to prevent any potential disasters like fires while also opening the possibility of future integration of these solar panels into other avenues. The goal of our project is to monitor and maintain the research solar panels while being able to interface with multiple solar panels to produce a single wireless gateway of collective panel information that feeds into a visually attractive remote access portal for observation and access to research panel data. In addition, it will also provide a means to control the solar panel's configuration.

The system will have a 12-volt isolated power supply in order to provide power to the microcontroller and other respective components. The power generated by each monitored solar panel will run through our smart interface box, giving us the ability to detect overvoltage and overcurrent conditions and disconnect the panels if necessary to prevent hazardous situations. Other features of the box will include reconfigurable tapping to allow users to determine which solar cells themselves are being observed. The various configurations are determined through a relay subsystem where we are able to output a set number of cells (32-cell output, 64-cell output, and 128-cell output) for researchers conducting experiments on solar panels. For example, choosing a 128-cell output will produce more voltage than the 32-cell output configuration and this is due to the fact that the surface area of the solar panels used to capture the sunlight is larger and therefore, producing more power. This configuration of solar panels is controlled through a wireless interface, allowing users to configure and monitor the solar panel remotely through an external server/portal.

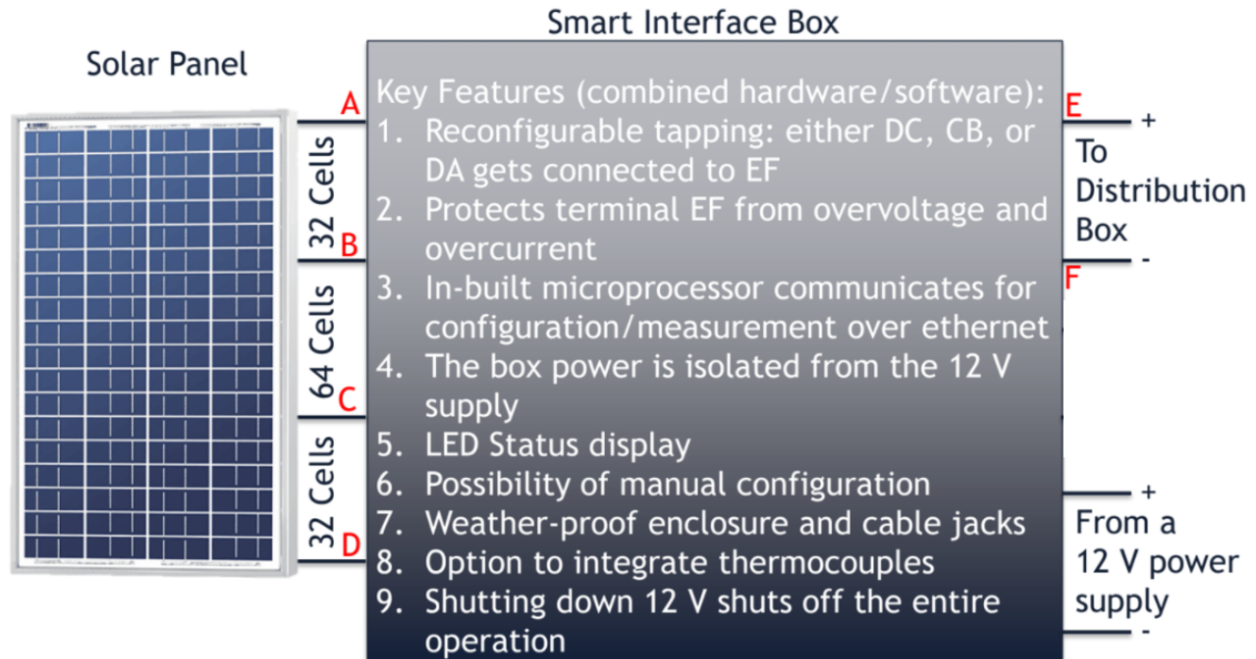


Figure 1: External Connections of the Smart Interface Box, the Solar Panels, the Load, and the External 12-Volt Power Supply

### 1.3 Visual Aid

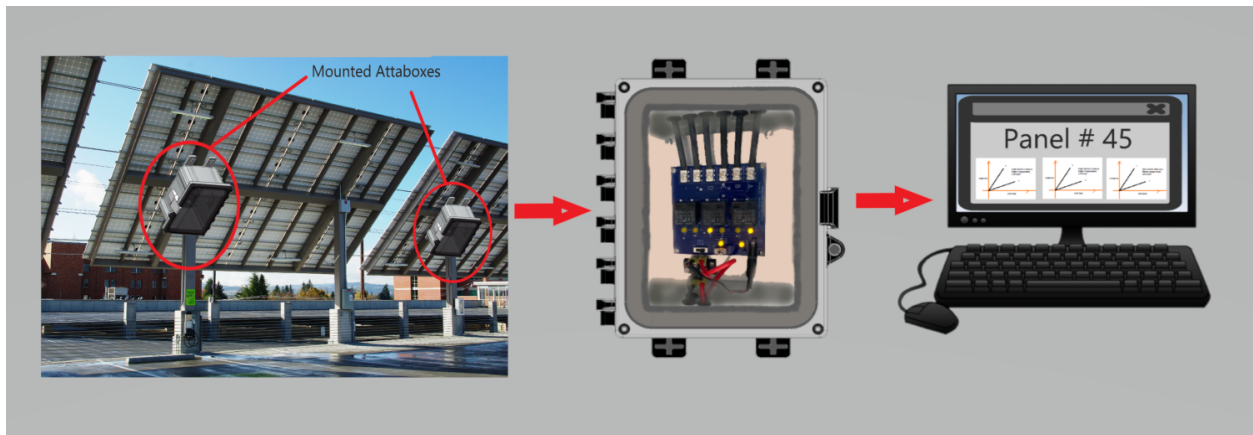


Figure 2: Diagram of the Connection Between Different Components of the Interface

### 1.4 High-Level Requirements

1. Record 3 key solar panel parameters at the interface box level: (Variations in reading these values is due to the nonlinear nature of the microcontroller's ADC pins)
  - a. Voltage - Accurately monitor the voltage of the relay output within an error range of  $\pm 300\text{mV}$

- b. Current - Accurately monitor the current of the relay output within an error range of 1.5%
  - c. Temperature - Between -10°C to +85°C, expect  $\pm 0.5\%$  accuracy
- 2. Wireless communication capability with interface box through a remote external access portal that shows researchers intuitive visualizations of obtained solar panel data for observation as well as configuring which section of the solar panel is being monitored.
- 3. Scalability - capability of interface box retrieving data and communicating with more than one solar panel. Starting with 1 solar panel and interface box, we want to be able to show scalability to a set of 2 solar panels and their corresponding interface box.

## 2 Design

### 2.1 Block Diagram

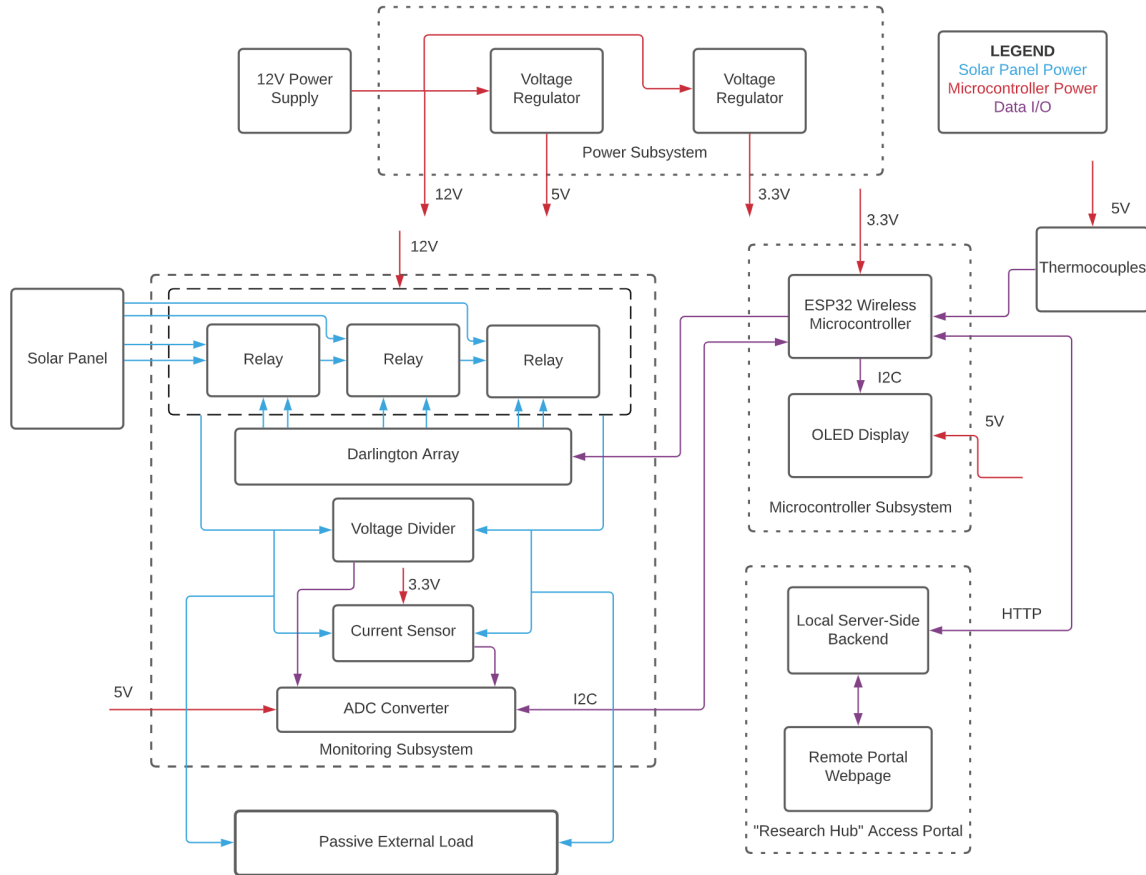


Figure 3: Labeled Block Diagram for an Advanced Interface Box

Our block diagram provides a high-level overview of how we expect to meet our high-level requirements. The power subsystem's primary function is to convert a 12-volt external supply and provide 3.3-volt and 5-volt supply lines to power up components like our ESP32 microcontroller, current sensor, ADC converter, etc. Our monitoring subsystem is responsible for relaying information about the voltage and current measurements to the ESP32 microcontroller through the ADC converter utilizing I2C. Within the monitoring subsystem, the ESP32 also dictates which of the relays are closed to determine which configuration of solar cells that we want to measure and output to the load. The voltage of the solar panel will also be stepped-down to a 0-3.3V range to be read by the microcontroller. The current sensor will also measure the current and an ADC converter is also utilized to provide galvanic isolation between the solar panel and the microcontroller so that possible voltage surges will not affect the operation of the equipment. The voltage, current, and temperature readings of the solar panels will be wirelessly communicated to our external research hub access portal so that the data of the solar panels can be tracked and displayed on a website for user configuration. These same readings will also be shown on the OLED display mounted inside the Attabox for convenience.

## 2.2 Subsystem Descriptions

### 2.2.1 Power Subsystem

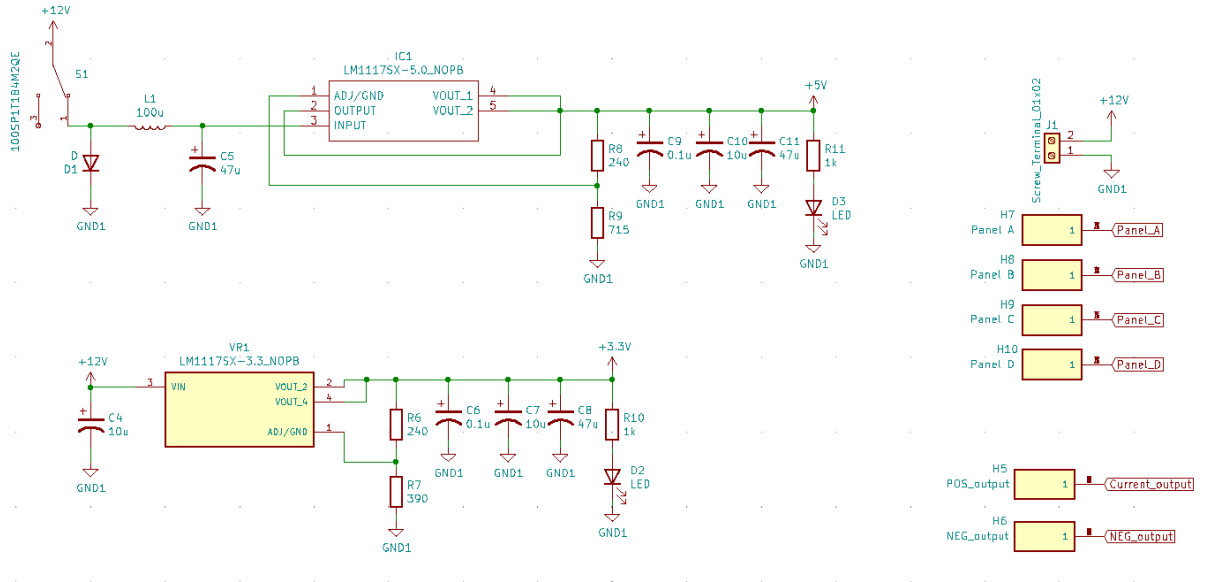


Figure 4: Linear Regulator Utilized to Step-Down the 12-Volt to Separate 3.3-Volt and 5-Volt Lines to Provide Power To Different Components

As an external 12-volt supply is provided, it allows for the step-down to lower voltages to power up our individual components and chips that we may need to create our smart interface box. A 12-volt supply is useful because it is simpler to step-down a DC 12-volt voltage in comparison to a constantly changing voltage output supply provided by the solar panel itself. Energy storage with a battery is also considered but for simplicity, we are utilizing a 12-volt supply. This 12-volt supply voltage will be stepped down through two linear regulators, to provide available 3.3V and 5V supplies. Thermocouples and a current sensor require a 3.3V supply whereas the OLED, ESP32 microcontroller, ADC converter, and our decoder are powered with a 5V supply. In addition, LEDs are also available to allow the user to see whether a 3.3V and 5V supply is available.

Table 1: R&V Table for Power Subsystem

Requirements	Verifications
The 12-volt power rail provides a 12-volt DC input within the range of $\pm 10\%$ [2]	1. Utilize a DC-power supply and input 12-volts into our screw terminal via jumper wires
Converts a 12-volt input into a 3.3-volt DC output within the range of $3.235V \leq V_{out} \leq 3.365V$ [3]	1. Probe the output of the linear regulator LM1117-3.3 with a multimeter and see if a stable 3.3-volt output is available
Converts a 12-volt input into a 5-volt DC	1. Probe the output of the linear regulator

output within the range of  $4.90V \leq V_{out} \leq 5.10V$  [3]

LM1117-5.0 with a multimeter and see if a stable 5.0-volt output is available

## 2.2.2 Monitoring Subsystem

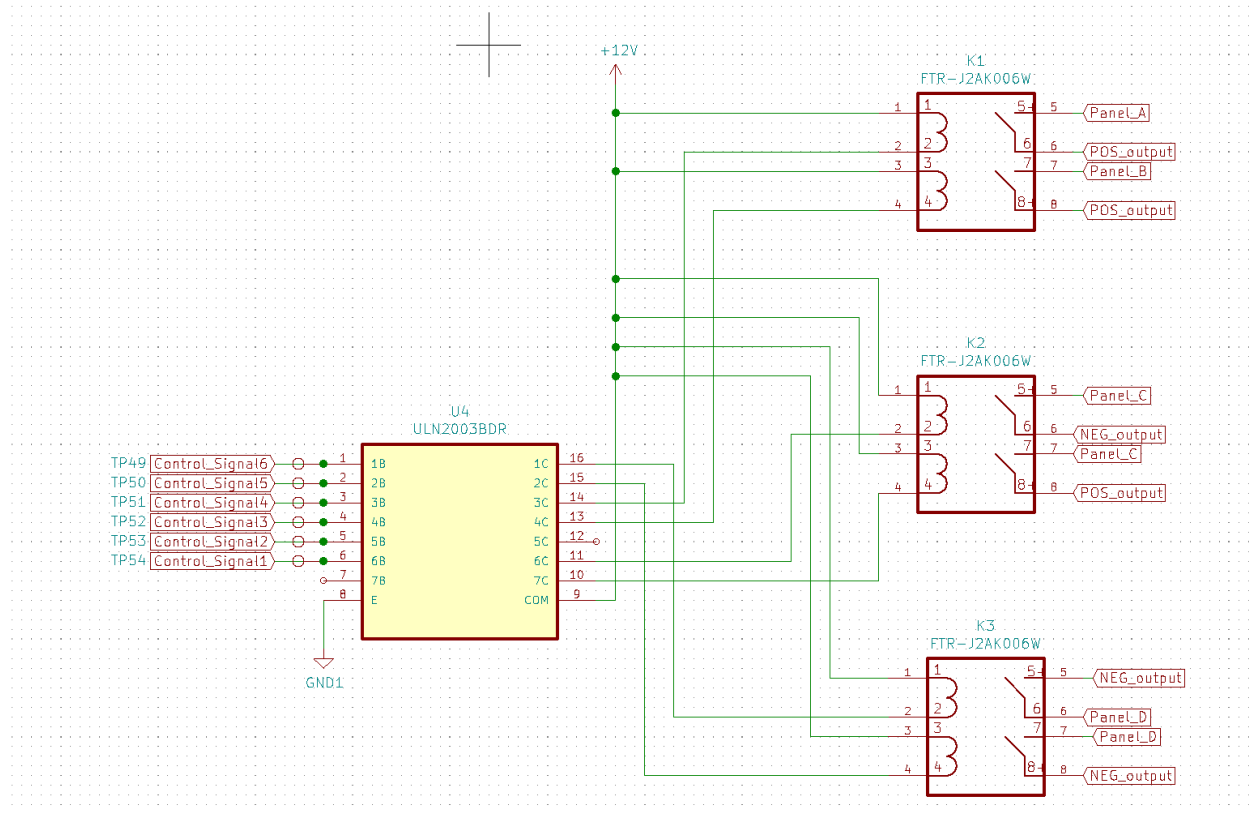
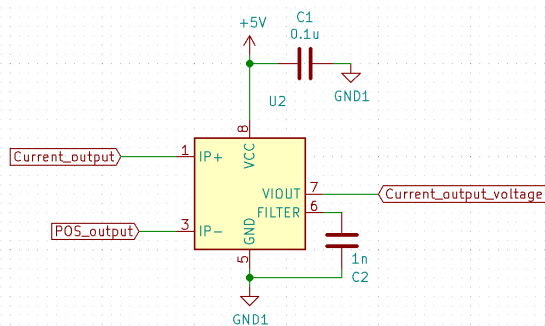


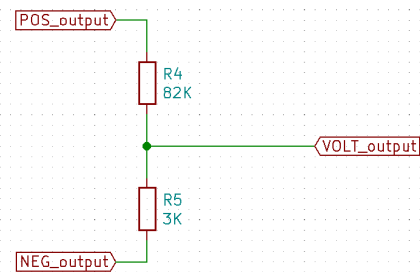
Figure 5: Relay System to Configure Which Solar Cells Are Input

Relays will be utilized to provide the ability to reconfigure which sections of the solar panel are being displayed to the wireless interface. The different configurations include 0-cells, 32-cells, 64-cells, and 128-cells acting as output. In order to be able to configure our switches properly we are utilizing three ultra-small high-voltage DC FTR-J2 Series Relays. These control signals for the relays will be connected to I/O pins on our ESP32 microcontroller. Utilizing a darlington array as the driver between the microcontroller and the relays will determine the choice of the 4 possible configurations as the solar cell output. These relays were chosen and given to our group as a design constraint due to the voltage and current ratings consistent with those of the solar panels. Since these relays require around a 12-volt signal in order for the relay switch to close, we don't connect the output of the microcontroller directly to the relays which is why we need a driver, to amplify the voltage output enough to toggle the switch closed. The output of the Darlington Array will drive the relays and thereby allow the user to attain the various configurations of solar cells that they desire by signalling the microcontroller to modify array configuration.





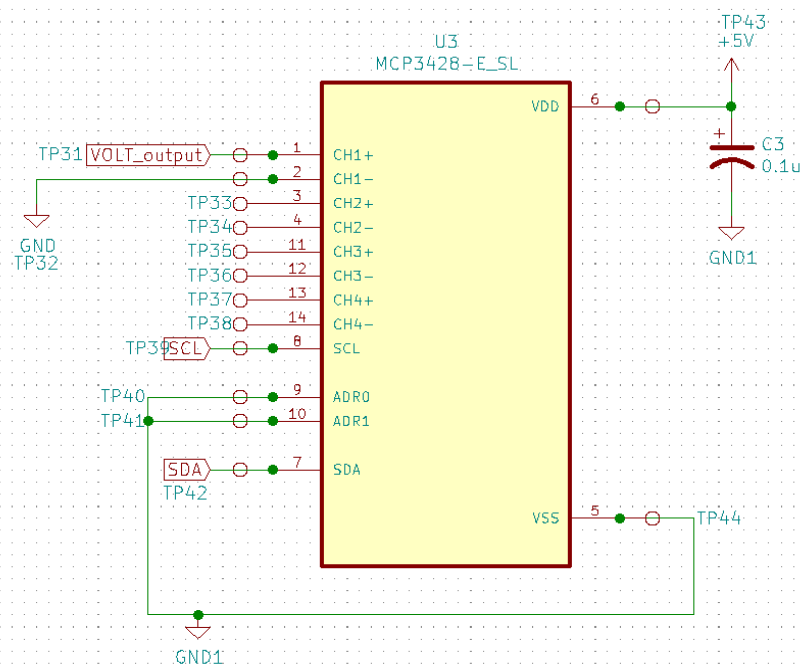
## Current Monitoring



## Voltage Monitoring

Figure 6: Voltage Divider Stepping-Down the Voltage for Voltage Monitoring and an ACS714 Current Sensor for Current Monitoring

Once the desired configuration is determined by the user, the solar output feeds directly into our voltage divider to step-down the voltage to be able to communicate its voltage in a range acceptable to the ADC pins of the microcontroller specified by the datasheet. This range of voltage levels varies between 0V and 3.3V. The current measurements will also be recorded with an ACS723 current sensor which detects the amount of current given the voltage that is input into the IC.



## ADC Converter

*Figure 7: An ADC Converter Utilized to Provide Galvanic Isolation Between the Solar Panels and the Microcontroller*

The outputs of both the voltage divider and the current sensor will be fed into an MCP3428 16-bit ADC in order to provide galvanic isolation between the voltage of the microcontroller and the voltage output of the solar panels. This way, we can separate the input and output supplies to enable that power transfer between these two circuits are not connected. This is a safety measure implemented so the user is not exposed to high voltage in the event that they come in contact with the microcontroller.

*Table 2: R&V Table for Monitoring Subsystem*

Requirements	Verifications
<p>The relay subsystem must be able to configure and choose between the different solar cell configurations: 32-cells (CD), 64-cells (BC), and 128-cells (AD). The relay output must be the same as the solar panel output minus the contact voltage and current drop [4]</p> <ul style="list-style-type: none"> <li>- Initial Voltage Contact Drop: <math>\pm 0.1V</math></li> <li>- Initial Current Contact Drop: <math>\pm 100mA</math></li> </ul>	<ol style="list-style-type: none"> <li>1. Connect a 15-volt DC power supply between CD screw terminals</li> <li>2. Between the Current_Output and NEG_Output screw terminals, measure the voltage with a multimeter and see whether we observe the 15-volt input when configured to CD</li> <li>3. Repeat A1 and A2 with screw terminals BC and AD</li> </ol>
<p>The voltage divider must be able to step down input voltages within the ranges of 0V - 85.6V and corresponding output voltages within the range of 0V - 3.3V [6]</p>	<ol style="list-style-type: none"> <li>1. Apply a DC power supply to the Current_Output and NEG_Output screw terminal*</li> <li>2. Utilize a multimeter to probe VOLT_Output to observe whether there is a voltage within a 0V - 3.3V range [7]</li> <li>3. Vary the range of the DC power supply to ensure it works for the 0V - 85.6V range</li> </ol>
<p>The current sensor must accurately read the current of the relay output within an error range of 1.5% [8]</p>	<ol style="list-style-type: none"> <li>1. Connect a DC power supply between CD screw terminals and make the input current 3A</li> <li>2. Measure the current between Current_Output and NEG_Output</li> <li>3. Measure the output of the ACS714 current sensor and observe whether the output is within 1.5% range of the current measured in D2</li> </ol>
<p>The ADC Converter should output a 16 bit digital code to our ESP32 that corresponds to the analog input into the IC once decoded.</p>	<ol style="list-style-type: none"> <li>1. Connect to a DC voltage supply of 5V to an input pin and ground its corresponding pin.</li> </ol>

	<ol style="list-style-type: none"> <li>2. Capture 16 bit digital code sent by ADC to ESP32 using serial monitor.</li> <li>3. Decode what was written to the ESP32 by multiplying the output code with the LSB and dividing by the PGA setting.</li> <li>4. Verify that it is the respective voltage that was inputted into the ADC.</li> </ol>
--	--

### 2.2.3 Microcontroller Subsystem

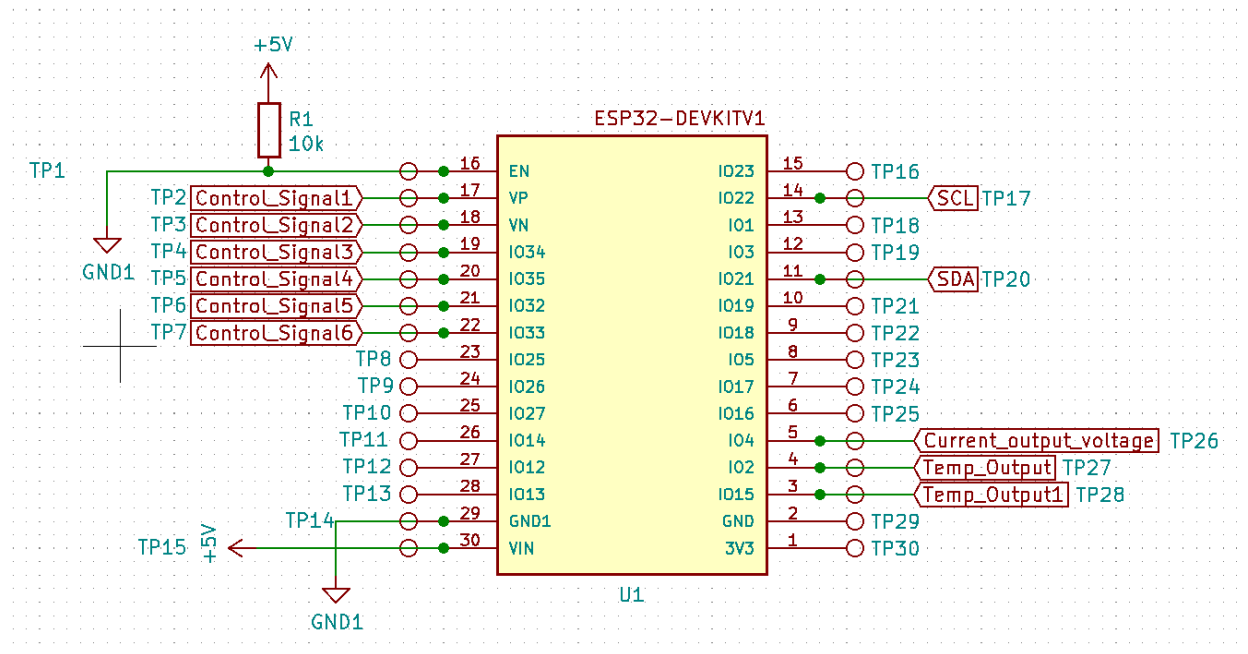


Figure 8: ESP32 Microcontroller Communicates User Input and Dictates Solar Panel Operation

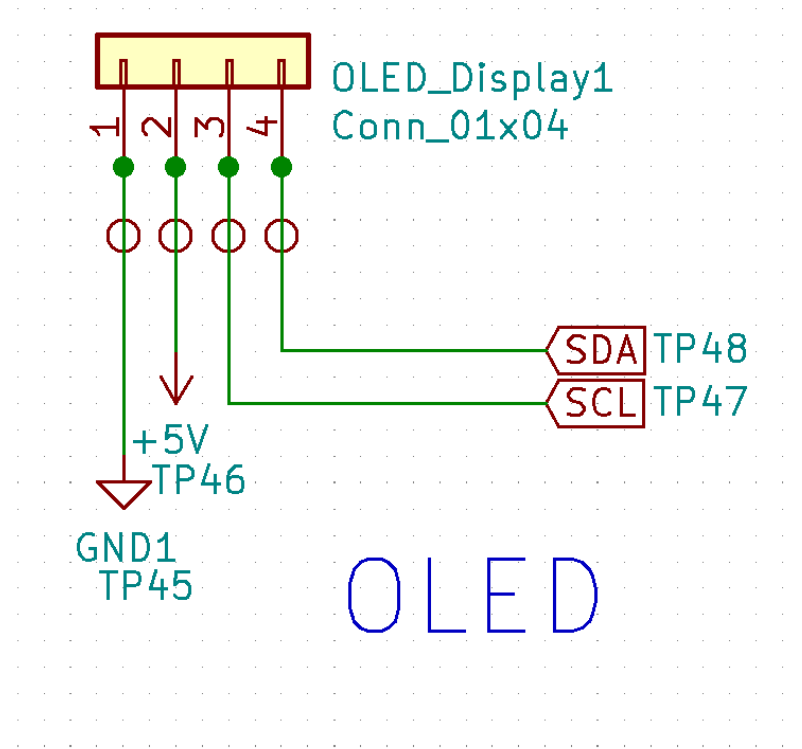


Figure 9: OLED Display to see voltage, current, and temperature readings of the solar panels

The ESP32 is the microcontroller responsible for much of the communication and instructions that we are trying to enforce in order to gather the solar parameters for the smart interface box. The ESP32 will take in the output from the ADC through I2C and communicate our voltage and current readings of the solar panel. We are also using two thermocouples directly connected to different areas of the solar cell, one at the center of the panel and another at the edge to get different temperature readings. If any temperature, voltage, or current reading exceeds what we set as the threshold parameters, then the configuration between the positive and negative output terminals of the relays will open and no solar cell will be connected to our microcontroller or to the passive external load.

Our OLED is what displays the current status of information gathered from the panels such as input voltage, current, and solar panel temperature. It communicates with the ESP32 microcontroller with I2C to display the real-time solar panel parameters that the user is interested in seeing. The SCL pin is the clock line and is used to synchronize our data as it is transferring over the bus. On the same bus is the SDA data line, which is used to transfer the data. The decision to implement an OLED visual display is solely for convenience to observe the solar parameters on site at each individual smart interface box.

Table 3: R&V Table for Microcontroller Subsystem

Requirements	Verifications
Depending on which configuration we are operating the solar panels on, the ESP32 microcontroller's corresponding IO pins	1. Choose CD configuration through the user interface

<p>must be able to provide a DC output within the range of <math>2.7V \leq V_{out} \leq 3.3V</math> to each Darlington pair.</p> <p>The following IO pins correspond to different configurations:</p>	<ol style="list-style-type: none"> <li>2. Use a multimeter to measure the output of the ADC2 pins corresponding to PANEL_C with Current_Output and PANEL_D with NEG_Output and observe whether it is 3.3V. The rest of the ADC2 pins should be 0V</li> <li>3. Change the configurations to BC and AD and repeat A1 and A2 with the according changes (i.e.: BC configuration measures output pins corresponding to PANEL_B with Current_Output and PANEL_C with NEG_Output)</li> </ol>
<p>The ESP32 can communicate the current solar panel parameters to the OLED display</p>	<ol style="list-style-type: none"> <li>1. Connect a Yokogawa power meter across the Current_Output and the NEG_Output probes and measure their DC voltage</li> <li>2. Connect the Yokogawa power meter probes across POS_Output and Current_Output to measure the current</li> <li>3. Use a temperature gun to measure the solar panel temperature at the locations where the thermocouples are attached (if digital temperature can be retrieved and displayed from the web interface, then the option to compare the OLED temperature readings and the temperature displayed on the web interface is viable)</li> <li>4. Observe the OLED display and see whether the voltage is displayed <math>\pm 300mV</math> [9], current within a 1.5% error range inline with the readings we retrieve from the power meter.</li> <li>5. Observe the OLED display and see whether the temperature is within the range of <math>\pm 0.5\%</math> accuracy we retrieve from the thermal imaging camera [10]</li> </ol> <p>* The Yokogawa power meter has the option of displaying the DC voltage and DC current</p>

\* Note: Keithley DC Power Supplies provided in the lab have a max voltage range of 30V. For safety, the voltage divider is simple enough to verify the output voltage range in simulation but for safety, we will not input a max 85.6-volt supply for safety reasons and because there is no voltage supply available for 85.6-volts.

## 2.2.4 External Access Portal: Research Hub

An external web-based server system - Research Hub - will be set up with two-way communication with the box's wireless network capabilities to allow for easy managing of the solar panels. Only authorized ECEB personnel will have access to this portal and it is intended for internal use, so the interface will be secured as such. The focus will be on being able to receive and monitor key solar panel data points, as well as porting the in-box button and switch capabilities to a remote setting for usage.

We will be using the Django framework to help manage the front-end webpage itself and the back-end database system for managing data from the interface box. The front-end will be scripted in HTML for the structure itself, while using Python for managing Django applications for different website functionalities. It is through this Django project that the security/user authentication and routing to the webpage will also be handled. The data management itself will be a combination of managing data signals at the microcontroller level and using a Python-based framework (primarily REST API concepts) to handle the back-and-forth data communication stream with the Django framework. This communication itself will be done through HTTP requests.

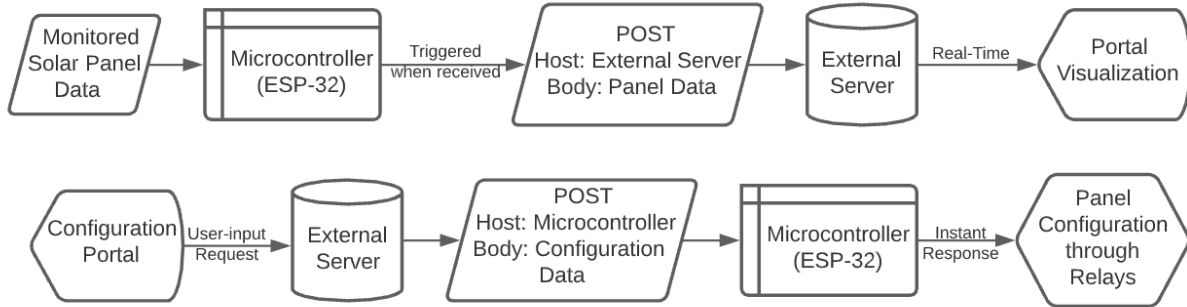


Figure 10: Flowchart Visualization of Remote Portal's Parallel Functionalities

The primary goal of the external portal is **Data Observation**. This is to ensure that the data collected and reported for monitoring purposes will be collected and then presented in an aesthetic manner on the access portal itself. A web-based public access of the data could also then be adapted and provided for public display as well. Aesthetic data visualization itself can be achieved with a combination of Python-based data science tools along with HTML to build it up on a webpage. This will be built on top of the Django webpage project, with a dedicated model for data visualization. This data will be updated in real time as and when the microcontroller receives new data, allowing for a constant stream of observable data. Dynamic updates will be achieved through Django using websockets and Django channels, with support from AJAX scripting. The website itself will be built with 4 key functionalities in mind (implemented through the Django project/applications) - managing multiple solar panels while also providing each panel dedicated pages, handling user authentication with respect to stored user credentials, providing team-specific information for continuation/improvement of the project, and of course plotting obtained data itself for observation and research.

The second goal of this external portal is **Wireless Panel Configuration**. The objective is to port the physical switch capability and the microcontroller-based panel section adjustments for monitoring solar panels, so as to allow for remote access. Since these are controlled through the switching subsystem and relays, we will process user-specified configurations on the front-end to be sent through the microcontroller to the relay system. This will use the same structure as mentioned above, but will utilize the Django REST framework on the back-end for communication to the microcontroller through HTTP requests.

Table 4: R&V Table for Research Hub Subsystem

Requirements	Verifications
--------------	---------------

Front-end website should successfully route between pages and display solar panel data based on security and observation use cases	<ol style="list-style-type: none"> <li>1. Landing page on accessing website with base URL (<a href="http://localhost:8000/">http://localhost:8000/</a>) should be the login page</li> <li>2. Successful login should be the only way to access panel data - direct URL modifications should reroute to login page</li> <li>3. Solar panel data is accessed through an index (should be easy to add additional panels for scalability)</li> <li>4. Website routing should go from login page to authentication to solar panel index which also has options for logout and team/project info</li> </ol>
<p>Working user authentication on the remote portal that restricts Research Hub access to required ECEB personnel only</p> <p><i>(Note: new users will only be added through administrative editing to allow for further security)</i></p>	<ol style="list-style-type: none"> <li>1. Add a user other than the Django superuser to the “users” Django application (through Django administration)</li> <li>2. Access the website and try to login through the welcome/initial landing page when live</li> <li>3. Website should welcome the user explicitly by username and route them to solar panel data if successful</li> </ol>
The back-end of the Django framework should have HTTP communication ability (2-way POST requests)	<ol style="list-style-type: none"> <li>1. Base test: <ol style="list-style-type: none"> <li>a. Send a POST request to the Postman API (hardcoded) from the Django back-end and it should be received and readable on the API</li> <li>b. Send a POST request from the Postman API to the Django back-end IP and it should be received and readable on the web server</li> </ol> </li> <li>2. Integration test: <ol style="list-style-type: none"> <li>a. Send a POST request to the ESP-32 server IP (hardcoded) from the Django back-end and it should be received and readable at the board level</li> <li>b. Send a POST request from the ESP-32 server (hardcoded) to the Django back-end IP and it should be received and readable on the web server</li> </ol> </li> </ol>
	<ol style="list-style-type: none"> <li>1. Base test: <ol style="list-style-type: none"> <li>a. Send a POST request to the Postman API (hardcoded) from the ESP-32 server and it should be received and readable</li> <li>b. Send a POST request from the Postman API to the ESP-32 server IP</li> </ol> </li> </ol>

<p>The ESP-32 Microcontroller should have HTTP communication ability (2-way POST requests)</p>	<p>and it should be received and readable at the board level</p> <ol style="list-style-type: none"> <li>2. Integration test: <ol style="list-style-type: none"> <li>a. Send a POST request from the ESP-32 server (hardcoded) to the Django back-end IP and it should be received and readable on the web server</li> <li>b. Send a POST request to the ESP-32 server IP (hardcoded) from the Django back-end and it should be received and readable at the board level</li> </ol> </li> </ol>
<p>Real-time observed data that is received should be processed correctly to allow for plot/visualization updates for the specific panel</p>	<ol style="list-style-type: none"> <li>1. Upload stream of arbitrary data points to the Django back-end for a specific panel by its model instance (can be randomly generated)</li> <li>2. Dedicated panel webpage should show update in data visualization/plotted charts in a time series manner</li> <li>3. Integration: follow same approach but by triggering the data update through a Postman API POST request to the Django framework with information</li> </ol>
<p>User-input configuration settings should be panel-specific and submission should trigger back-end communication</p>	<ol style="list-style-type: none"> <li>1. User-input through GUI on panel-specific webpage for configuration settings should update Django back-end database for the specific panel <ol style="list-style-type: none"> <li>a. It should also update visual cues on the webpage for that specific panel</li> </ol> </li> <li>2. Any back-end database change for a panel should trigger an outgoing POST request to the corresponding microcontroller</li> </ol>



## 2.2.5 External Components

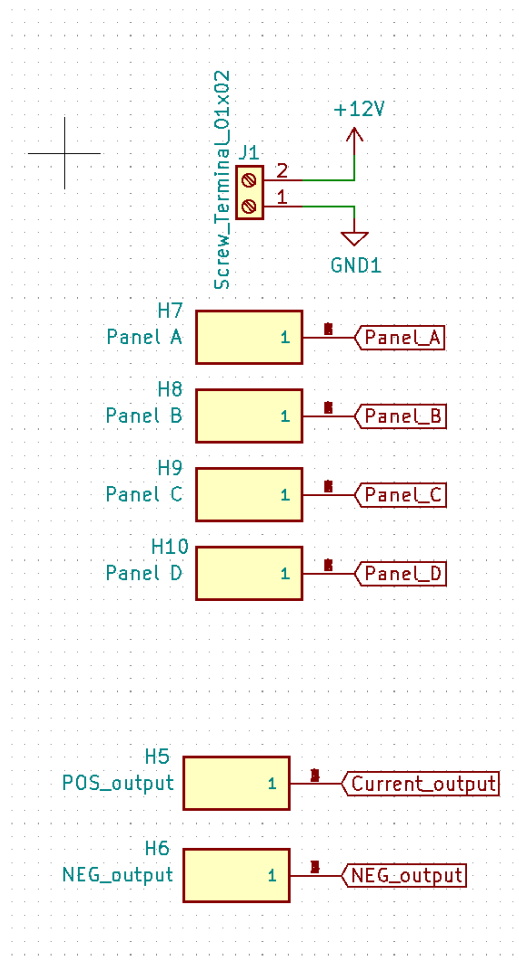


Figure 11: 12-Volt Supply, External Connections to Solar Panels, Output of Our PCB

### 2.2.5.1 Solar Panel

The main processor for our interface box which has functionality such as the ability to communicate the data received from the solar panels, shut down its 12V operations or limit any protection from overvoltage/current, and to determine areas in which certain panels may be overheating. Brings about wifi

functionality to push data at high speeds from the interface box to the research hub. In order to scale up the project to have our microprocessor communicate with multiple solar panels, there needs to be a wireless node network. We cannot strictly rely on wiring which communicates one panel's information, instead having a range of sensor nodes spatially dispersed to monitor and record the conditions of each individual solar panel which help bring in collective data to our display.

### 2.2.5.2 12-Volt Power Supply

A 12-volt supply line runs through all the solar panels as a means to provide a DC voltage to power up the components of the smart interface box. There is also a line of ground as well and this has already been pre-implemented prior to our solar panel design for the general purpose of powering up different interface boxes in a simple and easy manner.

### 2.2.5.3 Thermocouples

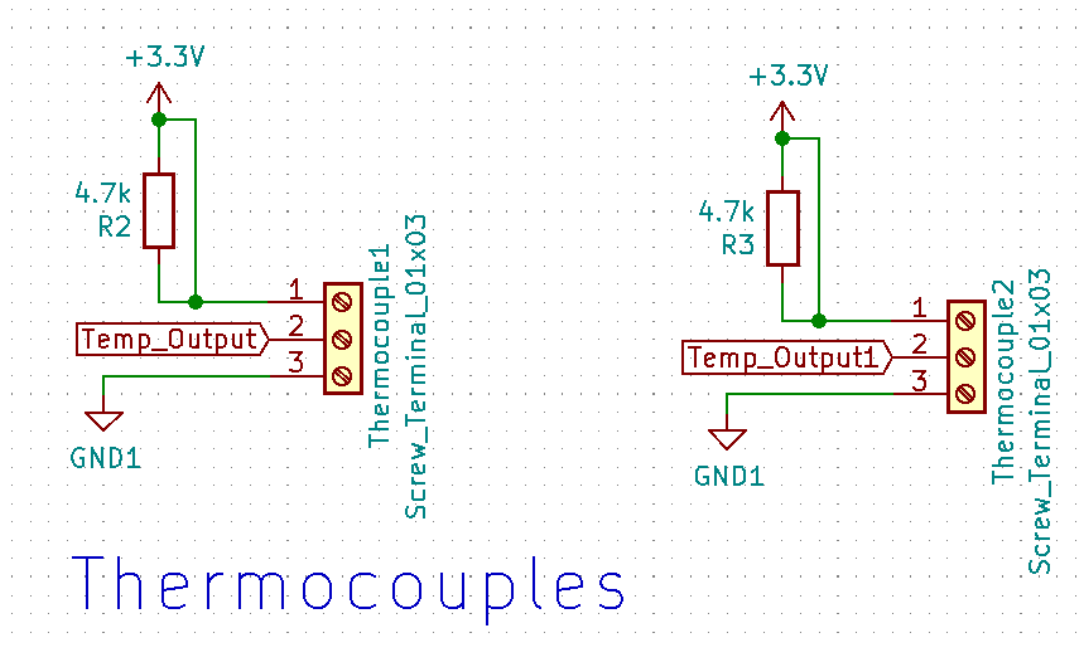


Figure 12: Thermocouples Utilized to Sense Solar Panel Temperature

A pair of thermocouples will be attached to each of the solar panels. The chosen thermocouple utilized is the waterproof DS18B20 digital temperature sensor and this specific part is chosen. The DS18B20 has a precise 1-wire digital temperature sensor that connects to the digital pins of our ESP32 microcontroller and gives up to 12-bits of precision with an onboard digital-to-analog converter. It is also powered with a 3.3-volt power supply from the output of our power subsystem and each thermocouple has a unique 64-bit ID on the chip allowing us to distinguish between different temperature readings.

### 2.2.5.4 Passive External Load

The output load of our solar panel is taken from the output of our small relay subsystem in which given the configuration command by the user through the ESP32 wireless microcontroller, we can decide which cell configuration will feed into our external load. We will not be utilizing an inverter to convert our DC

power to AC power as it would then be expected that we will be feeding power into the grid. The sole purpose of the smart interface box is to ensure that we can provide protective measures assisted with the capability of reading the voltage, current, and temperature parameters of the solar panels. A wireless research hub to display our data in a user-friendly and an aesthetically pleasing manner is also an addition from the previous design that had been implemented by a group of students in the Fall 2019 semester. Our load will consist of entirely passive components and mostly resistive just to dissipate the power. Once the design can prove that it works as expected, then additional features such as energy storage, satellite imaging, and the possible integration of power into the grid is a definite option.

## 2.3 Supporting Material

### 2.3.1 Mechanical Design and Mounting Diagrams

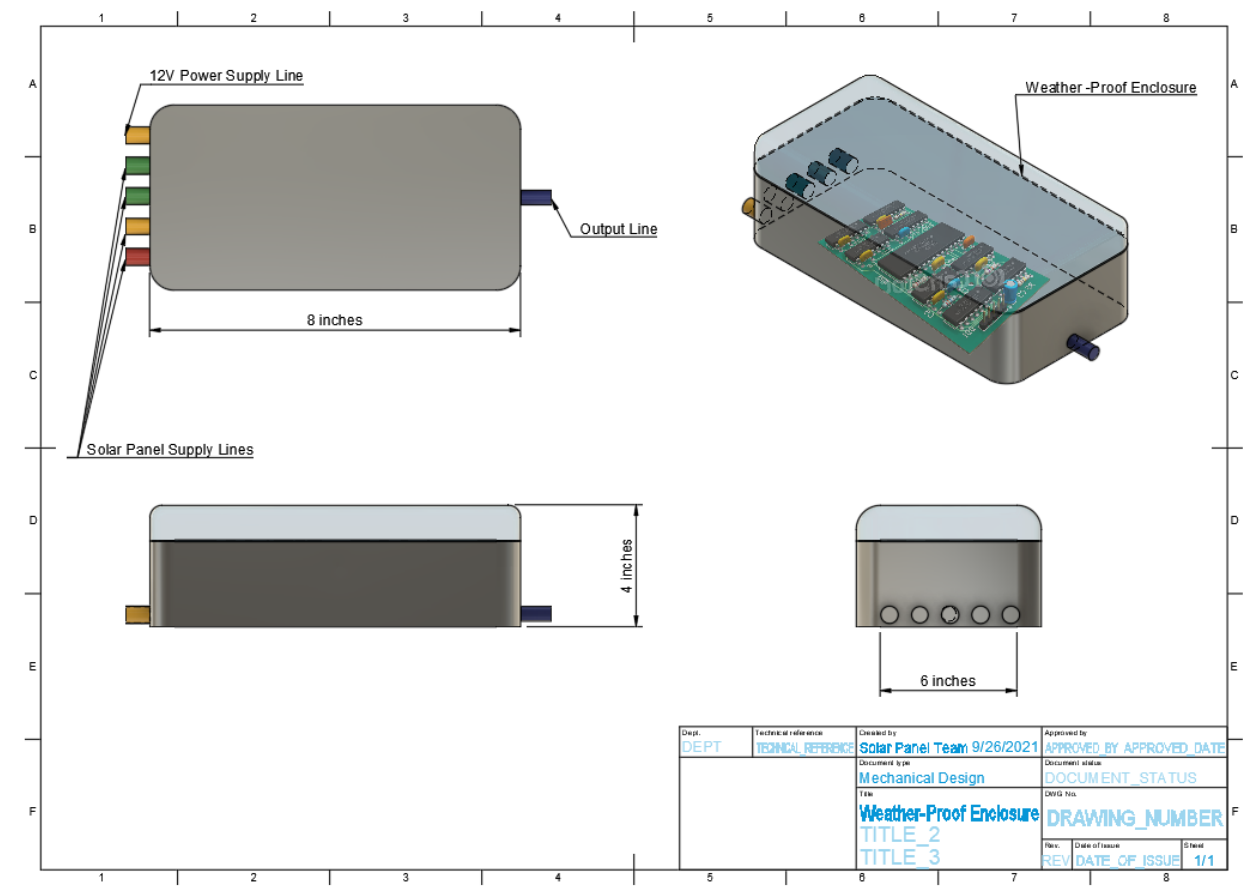


Figure 13: 3-D Visual Aid for the Box Enclosure



*Figure 14: Mounted Attaboxes on the Solar Panels (Roof of ECEB)*

### 2.3.2 Mechanical Design and Mounting Explanation

The mechanical design of our project is only our weatherproof enclosure. This was one of our design restraints when taking this project due to the weather proof enclosures already being purchased for the 60 solar panels. Our enclosure constraints are 8 x 6 x 4 inches as referenced from our enclosures part number [11]. This box is to be mounted on the solar panels on the roof of ECEB as shown in Figure 9. We know this is feasible as this design and mounting process have been implemented on eight of the 60 solar panels that are currently being housed on the roof.

## 2.4 Tolerance Analysis

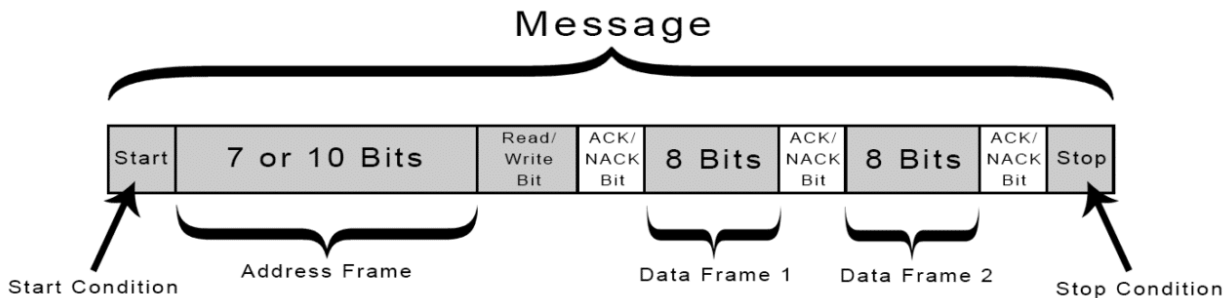
Our biggest risk while implementing this project is our ability to have a functioning wireless microcontroller that is able to communicate with our research hub and solar panels without the use of ethernet cables. The following is a mathematical calculation on the amount of data we would be able to store in our ESP32's flash memory in case the WIFI on the roof of the Electrical and Computer Engineering building drops out or is not able to support wireless communication at any point. We will be taking a snapshot of the data going into the ESP32 every minute. The ESP32 flash memory is able to hold up to 4MBs.

*ESP32 Data Transmit Speed = 802.11n @ 2.4 Ghz – 150 Mbit/s*

ESP32 Flash Memory Capacity = 4MBs

Snapshot of data occurrence = 1 min

Size of data snapshot: Max 30 bits [12]



*Figure 15: Visualization of bits that an I2C wire holds*

$$4 \text{ MBs} / (30 \text{ bits}) = 1066666.66 \text{ times we can snapshot before memory is filled}$$

$$1066666.66 (1/\text{min}) * (1/525600) = 2.029426674277 \text{ years}$$

The formula above states that in order to fill up our ESP32's 4MBs of flash memory, while we are pulling 30 bits of data every minute, we would need a little more than 2 calendar years, with no WIFI. A typical WIFI signal is regained within a day or a week in extreme matters, therefore our 2 years max gives room to store our data so that it is not lost, if our wifi capabilities somehow fail.

## 2.5 Protection Methods and Analysis

While implementing our senior design project, safety and safe protection of the solar panels are high priorities. However, there are already some safety mechanisms in place in order to protect the ECE building and its users from the solar panels such as:

- Failsafe button that is able to shut off power being drawn from all solar panels
- An electrically approved and powered panel that draws the solar panel outputs, each wire protected by a circuit breaker

We recognize that there are still hazards which require solar panels isolation. We would also like our solution to be implemented and therefore we brought forth two major safety measures for our design:

The first consists of manual switches that are able to shut down our design and stop pulling all Measurements from the solar panel. While we will be attempting to implement a switch within our online program, we recognize the dangers of unstable WiFi, therefore we have manual switches that are able to stop our analysis on the solar panels instantaneously in case of an emergency.

In case of unstable online data, we also plan to connect physical LEDs on our solar panels. These LEDs will be able to visually communicate with viewers if our board is communicating correctly and is

functioning. It will also be able to change colors to indicate if there is a problem with our design's functionality or if our design is not functioning at all.

### 3 Cost and Schedule

#### 3.1 Cost Analysis

Table 5: Parts to be bought for use (0 build hours)

Part Name	Vendor	Vendor Part #	Manufacturer Part #	Quantity	Cost x # Units = Total Cost
EVAL BOARD FOR ESP-WROOM-32	Amazon	1965-1000-ND	ESP32-DEVKITC-32D	2	$\$7.30 \times 2 = \$14.60$
SENSOR CURRENT HALL 30A AC/DC	Newark	SEN-13679 ROHS	ACS714ELCTR-30A-T	2	$\$7.08 \times 2 = \$14.16$
Dorhea 6PCS 0.96" OLED Display Module 12864 128x64 Pixel	Amazon	15758-1	7.01715E+11	2	$\$3.33 \times 2 = \$6.66$
WATERPROOF DS18B20 DIGITAL TEMPE	DigiKey	1738-1311-ND	DFR0198	4	$\$6.90 \times 4 = \$27.60$
DIODE SCHOTTKY 40V 3A SMA	DigiKey	B340AE-13DITR-ND	B340AE-13	6	$\$0.41 \times 6 = \$2.46$
DIODE RECT 100V 20A DPAK	DigiKey	497-16946-2-ND	FERD20H100SB-TR	15	$\$0.97 \times 15 = \$14.55$
IC INVERTER 6CH 6-INP 14TSSOP	DigiKey	296-26491-5-ND	SN74HCT04PW	3	$\$0.49 \times 3 = \$1.47$
IC REG LIN 3.3V 800MA DPAK	DigiKey	LM1117SX-3.3/NOPBTR-ND	LM1117SX-3.3/NOPB	4	$\$1.93 \times 4 = \$7.72$
IC REG LIN 5V 800MA DPAK/TO263	DigiKey	LM1117SX-5.0/NOPBTR-ND	LM1117SX-5.0	4	$\$1.41 \times 4 = \$5.64$
CAP ALUM 47UF 20% 25V SMD	DigiKey	493-3935-2-ND	UCL1E470MCL1GS	6	$\$0.57 \times 6 = \$3.42$
CAP CER 10UF 25V X5R 0805	DigiKey		C2012X5R1E106M125AB	6	$\$0.35 \times 6 = \$2.10$

		445-5985-2-ND			
CAP CER 0.1UF 50V X7R 0805	DigiKey	399-17644-2-ND	C0805J104K5 RACAUTO	5+5	\$0.53 x 10 = \$5.30
CAP CER 1000PF 25V X7R 0805	DigiKey	399-15433-2-ND	C0805C102K3 RAC7800	2	\$0.19 x 2 = \$0.38
RES SMD 240 OHM 1% 1/10W 0603	DigiKey	YAG3582CT-ND	AC0603FR-07 240RL	4	\$0.10 x 4 = \$0.40
RES SMD 1K OHM 1% 1/10W 0603	DigiKey	13-AC0603FR-131KLTR-ND	AC0603FR-13 1KL	6+4+4	\$0.10 x 14 = \$1.40
RES SMD 390 OHM 1% 1/10W 0603	DigiKey	311-390HRCT-ND	RC0603FR-07 390RL	2	\$0.10 x 2 = \$0.20
RES SMD 4.7K OHM 1% 1/10W 0603	DigiKey	YAG3613CT-ND	AC0603FR-07 4K7L	2	\$0.10 x 2 = \$0.20
RES SMD 560 OHM 1% 1/10W 0603	DigiKey	13-AF0603FR-07560RLTR-ND	AF0603FR-07 560RL	6	\$0.10 x 6 = \$0.60
RES SMD 10K OHM 1% 1/10W 0603	DigiKey	13-RT0603FR-D0710KLTR-ND	RT0603FRD07 10KL	6+6+8	\$0.15 x 20 = \$3.00
RES 3K OHM 1% 1/10W 0602	DigiKey	RMCF0603FT 3K00TR-ND	RMCF0603FT 3K00	2	\$0.10 x 2 = \$0.20
RES 240 OHM 1% 1/10W 0603	DigiKey	311-240HRTR-ND	RC0603FR-07 240RL	2	\$0.10 x 2 = \$0.20
IC PWR RELAY 7NPN 1:1 16SOIC	DigiKey	296-41065-2-ND	ULN2003BDR	2	\$0.71 x 2 = \$1.42
RES 715 OHM 1% 1/10W 0603	DigiKey	311-715HRTR-ND	RC0603FR-07 715RL	2	\$0.10 x 2 = \$0.20
RES 82K OHM 1% 1/10W 0603	DigiKey	RMCF0603FT 82K0TR-ND	RMCF0603FT 82K0	2	\$0.10 x 2 = \$0.20



SWITCH TOGGLE SPDT 5A 120V	DigiKey	EG2355-ND	100SP1T1B4 M2QE	2	$\$2.59 \times 2 = \$5.18$
LED GREEN CLEAR 0603 SMD	DigiKey	732-4980-1-N D	150060VS750 00	8	$\$0.15 \times 8 = \$1.20$
TERM SCREW 6-32 4 PIN PCB	DigiKey	36-8191-ND	8191	12	$\$0.47 \times 12 = \$3.76$
IC GATE DRVR LOW-SIDE 8SOIC	DigiKey	MCP14A0452- E/SN-ND	MCP14A0452- E/SN	12	$\$1.29 \times 12 = \$15.48$
IC DECODER/DEMUX 1X2:4 16SOIC	DigiKey	296-8229-5-N D	SN74HC139D	2	$\$0.47 \times 2 = \$0.96$
DEPEPE 30 Pcs 40 Pin 2.54mm Male and Female Pin Headers	Amazon	DE37566	7.1028E+11	1	\$5.39
Primary PCB	PCBWay	-	-	2	$\$4.90 \times 2 = \$9.80$
Weather Proof Enclosure	Amazon	-	-	2	$\$30.99 \times 2 = \$61.98$
Cable Glands	Amazon	-	-	16	$\$0.40 \times 16 = \$6.40$
MC4 Connectors	Amazon	-	-	12	$\$9.99/6 \times 2 =$ $\$19.98$
FTR-J2 Series Relay	Provided	N.A.	FTR-J2AK012 W	4	\$0
<b>Total</b>	-	-	-	-	$= \$219.93$

As seen in Table 2, we will need a total of \$219.93 budgeted to obtain the parts that we need for our project, i.e. to obtain the parts needed to construct two separate solar panel interface boxes.

Furthermore, this is a four credit hour class, which implies a minimum of 8 hours of work a week toward our project and its goals. In alignment with our future schedule planned for the remainder of the semester, we have 11 weeks or 88 hours of work ahead individually. Now, as per industry standard, UIUC Electrical Engineering undergraduates on average are paid \$80,000 per annum, while UIUC Computer Engineering undergraduates are paid \$100,000 per annum [13]. Allowing for some variance, we budget that an employee team member would be paid about \$40 per hour as employees on this project.

So, as per the assigned class budgeting formula, our project cost would be:  $(\$219.93) + [(\$40/\text{hour}) \times 2.5 \times 88 \text{ hours}] = \$9019.93 \approx \mathbf{\$9100}$

## 3.2 Future Schedule

Table 6: Planned Schedule of Work (per teammate)

Week	Sydney	Maram	Nikhil
09/27/21	Have a clear layout of the assembly of the design and assist with the first draft PCB schematic & board layout	Create a concrete first draft for PCB schematic design and send it to TA	Complete research on communication paths for remote configuration (panel->relay->microcontroller->server)
10/04/21	Completed cost analysis with approval and ordered parts for the assembly of the project	Have an approved/ nearly approved PCB design, finalized schematic, finalized relay configuration	Finalize wireless communication framework between external server and microcontroller
10/11/21	Gathering additional information on possible storage capability to document	Ordered all the parts for the PCB, awaiting for arrival	Finish coding out preliminary server-side framework; Do local testing within ECEB
10/18/21	Soldering top-layer PCB and observing any points of improvement; Determine measurement accuracy	Build PCB design and test for completion; Analyze board design to determine if version 2 of PCB is needed	Configure and code client-side framework on microcontroller; Integration testing with external server
10/25/21	Interface testing with the research panels per approval to determine how if/well our first-level design works; Individual readings of waveforms to determine possible rating spikes	Completed testing and finalized PCB design, if needed PCB v2 has passed audit and has been ordered, if not begin soldering components in preparation for interface testing	Complete wireless HTTP path between microcontroller and server; Begin working on configuration settings between relays and microcontroller
11/01/21	Isolating zones of our design to determine any alarming/unexpected characteristics on the PCB	Begin connecting interface functionalities to the PCB; Test wireless connection between panels and output results	Complete on-board setup for panel configuration; Begin connecting configuration capability with wireless network
11/08/21	Utilize lab equipment to determine any faults between the microcontroller or wireless communication	Finalize interface troubleshooting; Mount PCB board inside of Attabox; Begin configuring OLED and troubleshooting manual switches	Finalize wireless integration of server-board as well as board-panel communication; Begin working on wireless transfer of monitored panel data (4 key metrics)
11/15/21	May require a third PCB order for possible safety measures or points of improvement	Finalize OLED and manual switch analysis; Ensure functionality with the wireless network	Complete wireless network functionality including monitoring critical data; Set up web portal/GUI for project front-end
	Testing to determine whether	Mount attabox on the roof of	Finish setting up external

11/22/21	there is wireless communication capability between input readings of two or multiple solar panels	ECEB to be permanently connected to solar panels; Test all functionalities to ensure completion.	web-based portal; Finalize data visualization on portal; Complete verification of data reading across all points of communication
11/29/21*	Complete modular and integration testing; Demonstrate project working; Prepare project presentation	Complete modular and integration testing; Demonstrate project working; Prepare project presentation	Complete modular and integration testing; Demonstrate project working; Prepare project presentation
12/06/21*	Finalize presentation and present; Prepare and submit final paper	Finalize presentation and present; Prepare and submit final paper	Finalize presentation and present; Prepare and submit final paper

*Note: \* toward the completion of our project, work will become more combined with respect to reports and demonstrations. Hence, the tasks of the last 2 weeks have been entered as the same.*

## 4 Ethics and Safety

### 4.1 Ethical Issues

- Project Issues:
  - Solar energy should not contribute to net greenhouse gas emissions and exacerbate global climate change, and our peripheral device additions should not affect this either
    - Fix: We are exploring potentially recharging the power supply to our interface box by diverting some of the cleanly made solar energy
- General Solar Panel Issues:
  - Costs and benefits of solar energy should be distributed in an equitable way by regulatory agencies
  - Solar Energy should be environmentally sustainable - the recycling of solar panels at the moment is a huge impediment to their overall sustainability [14]
  - The manufacture of solar panels has also seen negative outcry due to problems like forced labor camps for polysilicon [15]

### 4.2 Safety Concerns

- Arc Flashes - Flash Burn and Blast Hazards
- Electric Shock, with particular scrutiny on faulty wiring as well
- Falls - due to the location itself of the solar panels for this project (ECEB roof)
- Thermal Burn Hazards
- Over-current/voltage - ideally will be directly countered by our interface box

### 4.3 Safety and Regulatory Standards

Given below are standards and procedures we intend to implement when designing and implementing our project. We plan to adhere to these guidelines carefully in order to keep ourselves and others safe, while maintaining the integrity of our work:

1. IEEE Ethics Code #1: Hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment' [16]
2. IEEE Ethics Code #7: To seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others' [16]

3. IEEE 1547-2018: IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces [17]
4. NREL/TP-550-38603 October 2005: Procedure for Measuring and Reporting the Performance of Photovoltaic Systems in Buildings [18]

## 5 Conclusion

Our final goal is to have an easy-to-use interface supported by our smart box that allows for accurate and convenient monitoring and up-keep of multiple ECEB research solar panels. We aim to have a prototype that can easily be scaled to meet the requirements whilst not exceeding the combined ratings of available solar panels, and in the end be successfully deployed in the ECEB!

### 5.1 Success Criteria

- With a focus on scaling up to meet the requirements of the solar panels available, we need to be able to interface with at least 10 research solar panels to be successful.
- Interfacing with a solar panel successfully encompasses accurately monitoring its voltage, current, power output, and temperature while simultaneously reporting this data to an external server.
- Remote Wireless Access toward the panels for the authorized ECEB personnel will be successful when it allows for these personnel to configure the solar panels from an external system.
- A successful interface will also provide aesthetic visualizations of panel data for observation (and possibly for general viewing).
- A successful prototype will also maintain the ability to manually control the interface box with in-box buttons and switches as a failsafe

# References

- [1] D. Vadgama, D. J. Lee, and S. Reddy, “Smart Interface Box for Solar Panels Final Report,” *ECE 445*, 12-Dec-2019. [Online]. Available: <https://courses.engr.illinois.edu/ece445/getfile.asp?id=16307>.
- [2] “60W Single Output Switching Power Supply.” [Online]. Available: <https://www.circuitspecialists.com/content/58308/PS1-60W-SL.pdf>. [Accessed Sept 25. 2021]
- [3] “LM1117 800-mA, Low-Dropout Linear Regulator,” *TI*. [Online]. Available: <https://www.ti.com/lit/ds/symlink/lm1117.pdf?HQS=dis-dk-null-digikeymode-dsf-pf-null-ww&ts=1634572215589>. [Accessed Sept 25. 2021]
- [4] “FTR-J2AK012W Fujitsu: Mouser,” *Mouser Electronics*. [Online]. Available: <https://www.mouser.com/ProductDetail/Fujitsu/FTR-J2AK012W?qs=EvTR%252B%2FkpER9cjG7hKs6Mbg%3D%3D>. [Accessed Sept 25. 2021]
- [5] “MCP1406/07 – 6A High-Speed Power MOSFET Drivers.” [Online]. Available: <https://ww1.microchip.com/downloads/en/DeviceDoc/20002019C.pdf>. [Accessed Sept 21. 2021]
- [6] “Analog Read Accuracy - ESP32 Forum,” *Espressif ESP32*. [Online]. Available: <https://esp32.com/viewtopic.php?t=13089>. [Accessed Sept 25. 2021]
- [7] James, R. Jones, Alex, Blackneron, S. Santos, Amjad, Mahdar, M. Fajari, Norbert, David, Farra, Mike, Phil, H. Allah, Rk, T. Baltazar, O. Lopez, M. Kelly, Zygryd, Alam, and Futterama, “ESP32 analog input with Arduino IDE,” *Random Nerd Tutorials*, 11-Nov-2020. [Online]. Available: <https://randomnerdtutorials.com/esp32-adc-analog-read-arduino-ide/>. [Accessed Sept 25. 2021]
- [8] “ACS714 Hall Effect Current Sensor,” *DigiKey*. [Online]. Available: <https://www.digikey.com/en/product-highlight/a/allegro-microsystems/acs714-automotive-grade-hall-effect-current-sensor>. [Accessed Sept 26. 2021]
- [9] Espressif, “ADC - Issue #1804 - Espressif Arduino ESP32,” *GitHub*. [Online]. Available: <https://github.com/espressif/arduino-esp32/issues/1804>. [Accessed Sept 21. 2021]
- [10] “DFR0198 datasheet by DFRobot,” *Digi*. [Online]. Available: [https://www.digikey.com/htmldatasheets/production/2423516/0/0/1/dfr0198.html?utm\\_adgroup=xGeneral&utm\\_source=google&utm\\_medium=cpc&utm\\_campaign=Dynamic+Search\\_EN\\_Product&utm\\_term=&utm\\_content=xGeneral&gclid=CjwKCAjwz5iMBhAEEiwAMEAwGBQII0BBa8TLN6clv0pc3m26AJrk8NiFfkHFsCgklkJPg2OzAhDIURoCcZ8QAvD\\_BwE](https://www.digikey.com/htmldatasheets/production/2423516/0/0/1/dfr0198.html?utm_adgroup=xGeneral&utm_source=google&utm_medium=cpc&utm_campaign=Dynamic+Search_EN_Product&utm_term=&utm_content=xGeneral&gclid=CjwKCAjwz5iMBhAEEiwAMEAwGBQII0BBa8TLN6clv0pc3m26AJrk8NiFfkHFsCgklkJPg2OzAhDIURoCcZ8QAvD_BwE). [Accessed Sept 21. 2021]
- [11] “Heartland 8 x 6 x 4 inches AH864C enclosure - Attabox: Non-metallic ENCLOSURES - polycarbonate and fiberglass,” *Attabox*, 15-Jun-2021. [Online]. Available: <https://attabox.com/product/heartland-8-6-4-ah864c-enclosure/>. [Accessed: 27-Sep-2021].

- [12] “Basics of the I2C communication protocol,” *Circuit Basics*, 11-Apr-2017. [Online]. Available: <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>. [Accessed: 12-Nov-2021].
- [13] Grainger Engineering Office of Marketing and Communications, “Salary averages,” *Electrical & Computer Engineering | UIUC*. [Online]. Available: <https://ece.illinois.edu/admissions/why-ece/salary-averages>. [Accessed: 25-Sep-2021].
- [14] V. Taylor, “The ethics of solar power are more complicated than you think,” *Mic*, 07-May-2021. [Online]. Available: <https://www.mic.com/p/the-ethics-of-solar-power-are-more-complicated-than-you-think-75484220>. [Accessed: 15-Sep-2021].
- [15] J. Gudmundsen, 2020 Jason Hayes | March 25, and 2020 Jack McHugh | December 18, “There are ethical concerns surrounding solar energy,” *Mackinac Center for Public Policy*, 12-Jul-2021. [Online]. Available: <https://www.mackinac.org/the-ethical-concerns-surrounding-solar-energy>. [Accessed: 15-Sep-2021].
- [16] “IEEE Code of Ethics.” Institute of Electrical and Electronics Engineers. 2016. <https://www.ieee.org/about/corporate/governance/p7-8.html> [Accessed Sept 19. 2021]
- [17] “IEEE 1547-2018 - IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power SYSTEMS INTERFACES.” *IEEE SA - The IEEE Standards Association - Home*, [standards.ieee.org/standard/1547-2018.html](https://standards.ieee.org/standard/1547-2018.html).
- [18] Pless, S., et al. “Procedure for Measuring and Reporting the Performance of Photovoltaic Systems in Buildings.” *UNT Digital Library*, National Renewable Energy Laboratory (U.S.), 3 Dec. 2015, [digital.library.unt.edu/ark:/67531/metadc785614/](https://digital.library.unt.edu/ark:/67531/metadc785614/).