IT Technology
Project Part 2 Report
Group 2 Solar Panel

Authors

Henrik Holm Hansen – hhha28796@edu.ucl.dk
Thobias Jon Selmann – tjse28878@edu.ucl.dk
Dainty Lyka Olsen – dlbo28887@edu.ucl.dk
Aubrey Sebastian Jones – asjo28903@edu.ucl.dk
Aleksandra Voronina – alvo28899@edu.ucl.dk
Gladys Waithera – gwma28853@edu.ucl.dk

December 10th, 2021

Characters: 60.667

Table of Contents

1 Introduction

This is the project report for the second portion of the Solar Panel Project by Group 2. This report covers what the members of this group have learned and achieved during the second portion of this project.

The scope of the second part of the project is to create a Solar Tracking system. What that means is that the working goal for this part of the project is to create an automatic response from the solar panel and automatic movement so that the panel will face wherever the sunlight is strongest and work in the most efficient manner possible so that the solar panel can gather as much energy as possible.

One of the major improvements that had to be approached was that the Graphical User Interface. While the GUI from the first part of the project was functional, it was only just so. It was not user friendly and the display from the Proof-of-Concept phase left out a lot of information that the end user might find useful so there was some focus on improving that for the Minimum Viable Product phase of this project. (The user interface had to be improved)

Another idea that had been brought up in the group's brainstorm was the idea that a PCB could be designed so that the project could move away from working with a messy and impractical breadboard and instead the solar panel could be controlled via a PCB of our own design. The motivation behind this was both to make the final product look more professional and additionally to test the skills that have been acquired in another course that the group is taking at the same time, which is the PCB Design Course. (PCBs had to be made.)

Be aware: this is the second portion of the project report. This report only covers the topics which have not yet been covered in the previous report. If you would like to learn more about the previous stages of this project you can follow along with the journey by checking out the project at our Gitlab Repository

2 Background Description

After the first portion of the projects implementation the system consisted of a Solar panel, a BNO055 sensor, and an Arduino Nano. The Solar panel does not go out of its boundaries and a simple graphical user interface has been put in place.

Therefore, the scope of the second part of the project was to make a functional solar tracking system so that the solar panel can follow the position of the sun without any input from a human.

3 Problem Statement

A described in the Background Description, team had a task for the 2nd part of the project to continue improving the Solar Panel. This has been divided into these three sub-issues:

- How do we make a solar tracking system using the minimum amount of power while still keeping the power generation efficient?
- How can it be combined with the system from the first portion of the project which stops when it has reached its limitations?
- What improvements can be made to the user interface? Is there anything that can be done to improve the end user experience?

4 Definition of Purpose

The purpose of the project remains the same. The project's purpose is to help UCL make a better and more functional system for the existing solar panel which is unfortunately not smart enough yet to be able to track the sun or to stay within its mechanical boundaries.

After setting up limitations so that the solar panel does not break itself if it attempts to move beyond the mechanical thresholds, the next step was to create a method in which the solar panel could autonomously track the location of the sun while additionally maximizing energy generation while minimizing energy expenditure.

5 Project Management

As two separate teams were merged after the end of the first portion of the project, the team had to implement some new Project Management strategies to utilize the bigger team size. The weekly team meetings have become mandatory for all of the team members whether they are at school though it is possible to attend from home if they cannot be present that day. The most convenient and comfortable time for everyone has been decided upon with considerations given to the jobs and other arrangements people that our group members have.

The team used the SCRUM and Sprint meeting techniques. Meetings have started by going around the group and having each team member talking about what they are currently working on as well as any help that they need to continue working. This was not a completely new practice as the team has used SCRUM and Sprint to coordinate teamwork for work on the previous semester's project.

A shared Gitlab page has been created wherein team members can easily see the progress that has been made and a list of the tasks that still need to be worked upon.

5.1 Brainstorming

The Brainstorming for the 2nd portion of the project had been completed separately by two parts of the team that still had not been merged in at that point. The first step after the merger was to make a new Brainstorm after combining the two separate teams into one. This was done to utilize the groups larger size to generate more ideas and test more ideas quickly in a shorter time frame.

The technique that the group had chosen to work with was the "Mind Mapping" technique. This method consists of team members writing ideas down in a public space and mapping the ideas onto a whiteboard so that there will be a visual demonstration for everybody regarding what the group has decided to make. Everybody had a chance to write questions and add and contribute to the already existing ideas.

The brainstorm was originally made using the whiteboard in one of the classrooms, and then the images were remade into the more readable digital versions that you can see here:

Part 1:

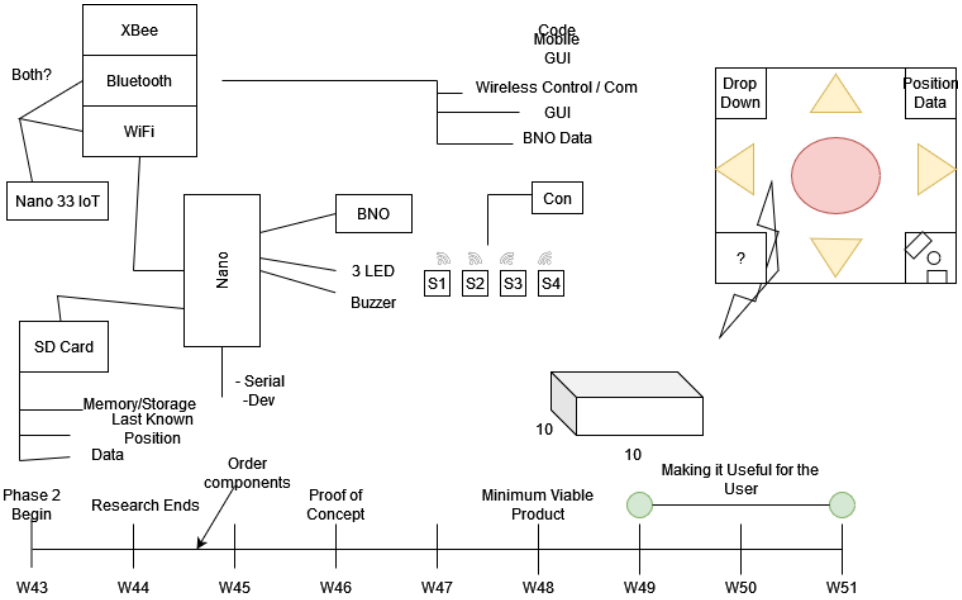


Figure 5.1 1Brainstorm Part 1

Part 2:

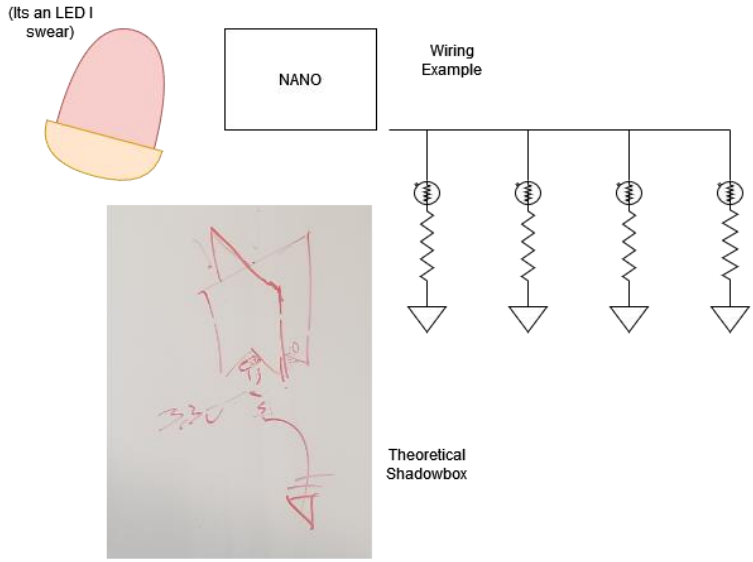


Figure 5.2 Brainstorm Part 2

5.2 Team Structure

The team had to formulate an entirely new team structure for the second portion of the project because of the team merger that occurred after the first portion of the project had successfully concluded. It was challenging to get things started at first because the two separate teams had already been formed and had their own roles and work structures implemented.

The combination of the two teams was done as efficiently as possible, however, to make sure that everybody had a good idea of what work needed to be done so everyone could focus on the parts of the project that they were most interested in.

6 Unified Process

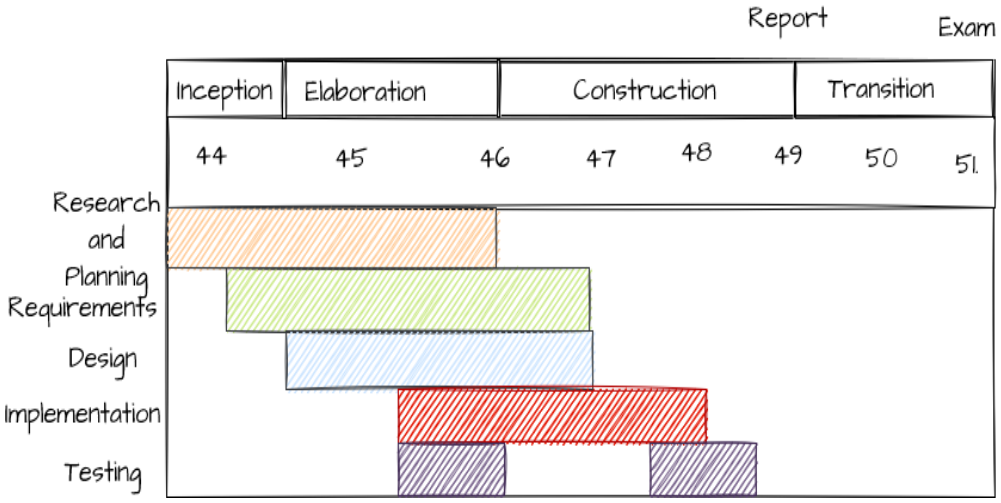


Figure 6 Timetable

During this period of the project, the class was introduced to another software development process which is called the Unified Process framework. It is an iterative and incremental process during which the phases are divided into iterations which results in an increment that contains improved functionality in addition to the previous stage of the project.

Inception Phase: As shown on the graph, inception phase is the shortest phase among all phases. This is where a lot of focus is being put into planning the project. The group started brainstorming for ideas relevant to the system that is going to be built. This was also the phase where the group started to analyze the feasibility of the project, in terms of the group's capabilities to achieve the goals within the scope of the project considering the financial and time constraints involved.
At the end of this phase, the group has a basic idea of the preliminary cost and schedule of moving forward with the project.

Elaboration Phase: This is the phase where the group has spent a lot of time and quite a bit of effort on defining the requirements. A lot of activities happened around capturing the requirements in terms of use case. This is also the phase where emphasis on what could go wrong, how the team mitigated and resolved known risks that could arise while creating the project. System design has also been defined as creating a baseline for building the core components of the system. Making sure that the product will work and the group at the concept level can validate functionality to establish an estimate for the next phase of constructing the system.

Construction Phase: This is the largest phase of this project. So here the group started building the hardware and software parts of the system by listing all the functions and figuring out the connections between them. In the construction step, a lot of effort has been made into the implementation of the software process and testing. This is also divided into multiple iterations each week, where the team was constantly checking by enhancing or replacing the current version and learning if they were going in the right direction.

Transition Phase: The last phase is called the Transition phase, where the final step of testing is done and finally the deployment of the final product. This phase has been focused on a lot of small improvements to code efficiency as well as finalizing the appearance of the product and making strides towards making the product appear and function as well as the team had imagined at the beginning of the second portion of the Project.

7 Requirement Draft

Functional Requirement:

- As the end user I want to make sure that the solar panel does not go out of its mechanical boundaries, so it will not run into any objects in the vicinity around it so that it does not break itself or anyone in the immediate surrounding area.

- As the end user I want to be able to have manual control over the solar panel so that it is possible to manually stop the system just in case damage needs to be repaired or the panel needs to be manually moved into a certain position.

- As the end user I want to have a Guided User Interface available so that I can control the solar panels linear actuators.

- As the end user I want the solar panel to have a tracking system, so that the solar panel can follow the sun's direction without any meaningful input from me.

Non-Functional Requirement:

- The system must have data measurement capabilities for the power output created on the interface

- The system must have a usability test that can be run by the end user.

8 Block Diagram

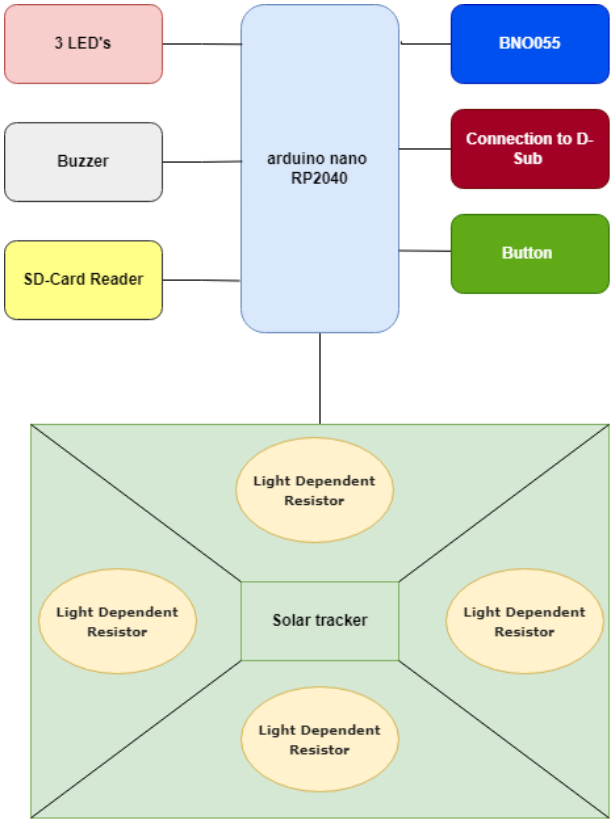


Figure 8 Block Diagram

- The Arduino Nano RP2040 is the centerpiece of the design. It is the brain that controls and directs and obtains all the other signals from all the other modules and redirects them towards where they need to be for the entire unit to function.

- The D-Sub connection is actually a thermal block which connects with the solar panel's input signals to control its linear actuators.

- The push button is connected for the purpose of testing LEDs and Buzzer to make sure that they are functioning correctly. A capacitor is added to the voltage input pin to make sure that it gets a steady flow of current.

- The BNO055 is the safety mechanism that makes sure that the solar panel doesn't go out of boundaries. For that reason, it is essential that this module also gets a steady voltage and current flow and for that reason the capacitor has been added to the voltage input as well.

- The LED's and the buzzer are visual and vocal acknowledgments for users that are using the product. They inform the user about the current working status of the product.

- The SD card reader is mainly for storing calibration data from the BNO055 but can have other benefits for the project in the long run.

- The solar tracker is an external module that connects to the Arduino Nano RP2040. It is built from 4 light-dependent resistors installed onto a 3D Printed shadowbox.

9 Use Case

9.1 Actor context

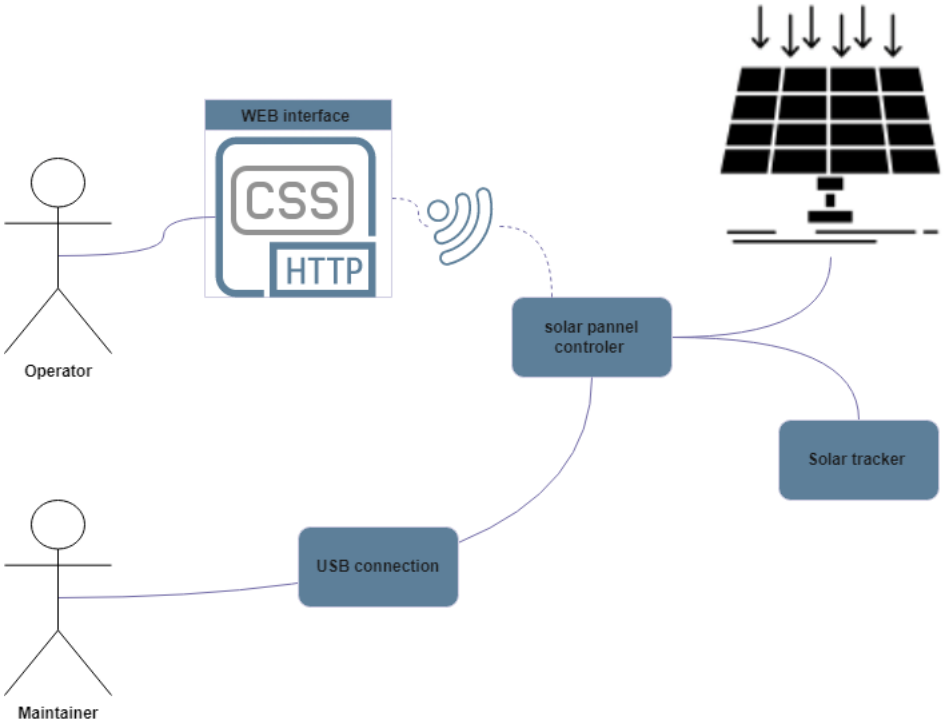


Figure 9.1 Use Case

The Operator can do the following by using the Web Interface:

- They can access the Web Interface to manually control the Solar Panel's linear actuators.
- From the Web Interface the operator can insert values of directional limitations to the solar panel.

The Maintainer can do the following by using the Web Interface:

- They can access the Web Interface to manually control the Solar Panel's linear actuators.
- From the Web Interface the operator can insert values of directional limitations to the solar panel.

The Maintainer can do the following by connecting to the Solar Panel with a USB Micro-B Cable:

- They can access the source code and manually edit it.
- They can access the source code and manually override it.
- They can access the terminal output to see debugging information regarding the solar panel's operation.

10 Overview of Hardware Components

10.1 Components.

10.1.1 Nano RP2040 Connect

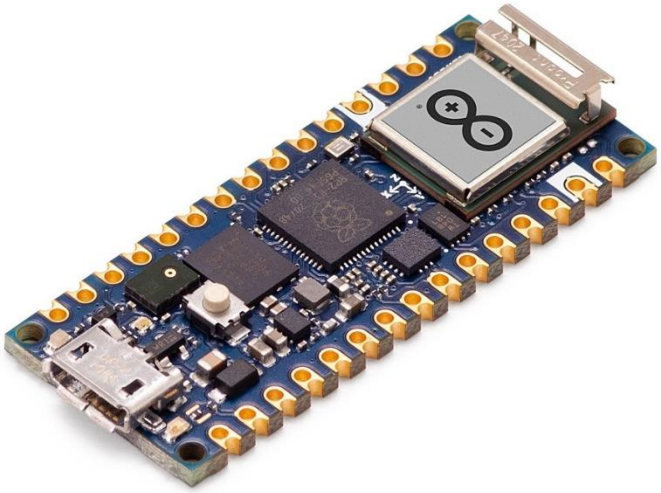


Figure 10.1.1 Nano RP2040 Connect

Due to the inability of the Arduino Nano Every to connect wirelessly to the project from the first portion of the third semester, the group has chosen to use the new Nano RP2040 Connect board which has a built-in Wi-Fi module. This has opened up a variety of possible paths forward for the project. Since the new Nano RP2040 Connect has some great advantages precisely in this direction this project

- Python Support
  - Python is a useful and flexible programming language and has easy syntax, and it is one that the team was considering shifting to because the team has more experience coding in Python than they do with C++.

- U-blox® Nina W102
  - Nina is a useful feature because it allows the module to connect wirelessly and communicate with other modules.

- ST LSM6DSOX 6-axis IMU
  - The team considered utilizing the on board IMU to control the linear actuators which manage the movement limitations of the solar panel instead of the BNO055 due to the problems the team experienced regarding the BNO's issues with calibration.

- Dual Core 32-bit Arm® Cortex
  - The team would have liked to use 1 of the 2 cores available exclusively for the transfer and utilization of BNO055 data. This is because it is important that the data that the BNO55 transfers is not interrupted by other processes in the pipeline. The safety of the system relies on the data that is sent by the BNO055.

- Step down Converter
  - The Step Down Converter is a convenient way to use outcoming voltage outputs to power the system.

10.1.2 Adafruit MicroSD card breakout board+

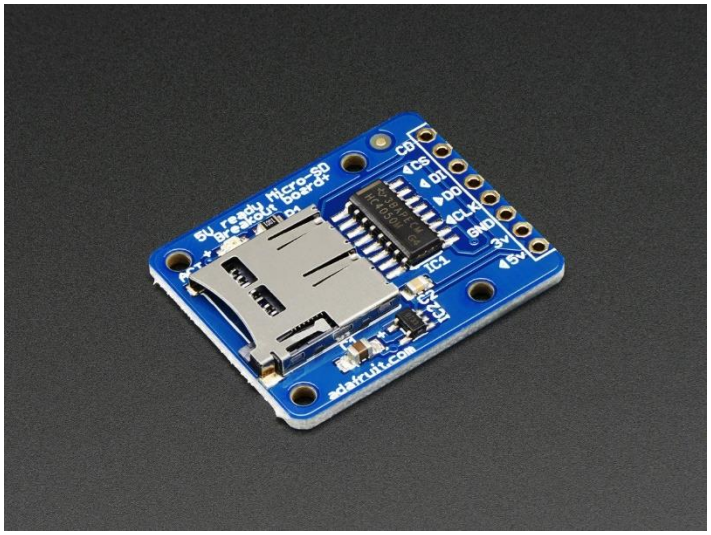


Figure 10.1.2 Adafruit MicroSD card breakout board +

This module is not just an ordinary SD card reader because it is specifically designed to work with the Arduino. This makes it extremely easy to connect to the Arduino because with just 4 wires it is ready to connect. From the Arduino to the SD Card Reader, you only need to make a few connections to make the piece work correctly.

You need:

- Connect GND to ground
- 5V to 5V
- CLK to pin 13
- DO to pin 12
- DI to pin 11
- CS to pin 10

Once everything has been connected properly Arduino IDE's SD library can support FAT and FAT32 SD card connections. After a bit of research, the group has concluded that this is the best SD Card Reader solution for this project because the Nano RP2040 connecting pinouts are running low in our PCB module. This piece takes up relatively few connecting pinouts, so it matches the needs of our project very well.

11 Web Based Graphical User Interface (GUI)

This is the web based Graphical User Interface. It can be used to manually control the directional movement of the solar panel. It can make the solar panel move Up, Down, Left, or Right. When the panel is in movement you manually must stop it with the big red stop button currently in the middle of the GUI. Options have also been given to insert the limits of the solar panel and you can activate them by pressing the green buttons on the GUI that correspond to the direction that the operator wishes to limit. This is a fast way to calibrate the Solar Panel's directional limitations in 3-Dimensional Space.

Figure 11 Graphical User Interface

12 Software Design

12.1 GUI Code

```
<style>
    .container {
        margin: 0;
        padding-top:0;
        border: none;
        top: 0em;
        left: 0em;
        text-align: center;
        position:fixed;
        width: 100%;
    }
    .header {
        height: 4.5em;
        width: auto;
```

```
        background-color: teal;
        text-align: left;

    }
    h1 {
        position: absolute;
        color: white;
        border: none;
        font-size: 2.9em;
        margin: 0;
        padding: 0;
        top: 0.1em;
        left: 2em;
    }

    button {
        color: white;
        width: 100px;
        height: 100px;
        border-radius: 33%;
        margin: 20px;
        border: none;
        font-size: 20px;
        outline: none;
        transition: all 0.2s;
    }
    .red{background-color: rgb(196, 39, 39);}
    .green{background-color: rgb(39, 121, 39);}
    .blue {background-color: rgb(5, 87, 180);}
    button:hover{cursor: pointer; opacity: 0.7;}

    .toggle {
        position: absolute;
        top:0em;
        left: 0em;
    }


</style>

<div class='container'>
    <div class="header">
        <h1>Solar Panel Control</h1>
    </div>
    <div class="">
        <button class='green' type='submit'
onmousedown='location.href="/UPLIMIT"'>UP limit</button>
        <button class='blue' type='submit'
onmousedown='location.href="/UPHIGH"'>UP</button>
```

```
        <button class='green' type='submit'
onmousedown='location.href="/RIGHTLIMIT"'>RIGHT Limit</button>
        <br>
        <button class='blue' type='submit'
onmousedown='location.href="/LEFTHIGH"'>LEFT</button>
        <button class='red' type='submit'
onmousedown='location.href="/ALLSTOP"'>STOP</button>
        <button class='blue' type='submit'
onmousedown='location.href="/RIGHTHIGH"'>RIGHT</button>
        <br>
        <button class='green' type='submit'
onmousedown='location.href="/DOWNLIMIT"'>DOWN Limit </button>
        <button class='blue' type='submit'
onmousedown='location.href="/DOWNHIGH"'>DOWN</button>
        <button class='green' type='submit'
onmousedown='location.href="/LEFTLIMIT"'>LEFT Limit</button>
    </div>
</div>
```

This is how the GUI was designed in HTML before it was transformed into code code that was printable for the Arduino Nano code.

13 Project Testing

The requirement of testing during the project life cycle is important because it could help us:

- *To identify defects*
- *To reduce flaws in the component or system*
- *Increase the overall quality of the system*

The group was able to do some primary testing on a breadboard (as shown below). This helped us get an overview of the solar tracker and the chance to redo parts of the project which the group thought needed to get done. Testing the product also helped the group in detecting that the buzzer was not working properly due to the voltage. The group was able to solve the problem which would have been more difficult to fix after all the assembling.

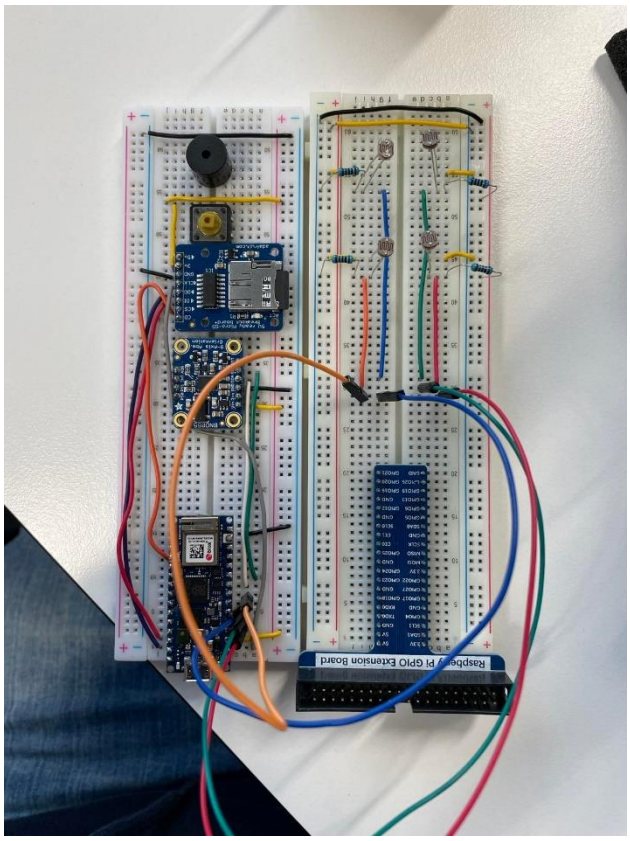


Figure 13 Breadboard with the components connected to it

14 System Recreation

14.1 Software

- Arduino IDE
  - Libraries needed
    - Adafruit Unified Sensor by Adafruit
    - Adafruit BNO005 by Adafruit
    - Adafruit BusIO by Adafruit
    - WiFiNINA by Arduino
  - Boards Needed
    - Arduino Mbed OS RP2040 Boards
- Any kind of Browser

To add the libraries, go in your Arduino IDE click on Tools > Manage Libraries
In the library manager search for the libraries mentioned above and install them.
For GUI install processing and extract it to its own folder.

14.2 Hardware

14.2.1 Inventory

1 Arduino Nano RP2040 Connect
1 BNO055
4 Photo Sensors
1 Buzzer
1 SD Card Reader
1 SD Card
1 Button
1 USB-C Cable
3 LED’s of various color
1 Container full of your favorite brand of wire

14.2.2 Visual wire connection guide

Here is a visual guide to reconstructing the project on a breadboard. To do this, you will need a variety of parts which are listed above.

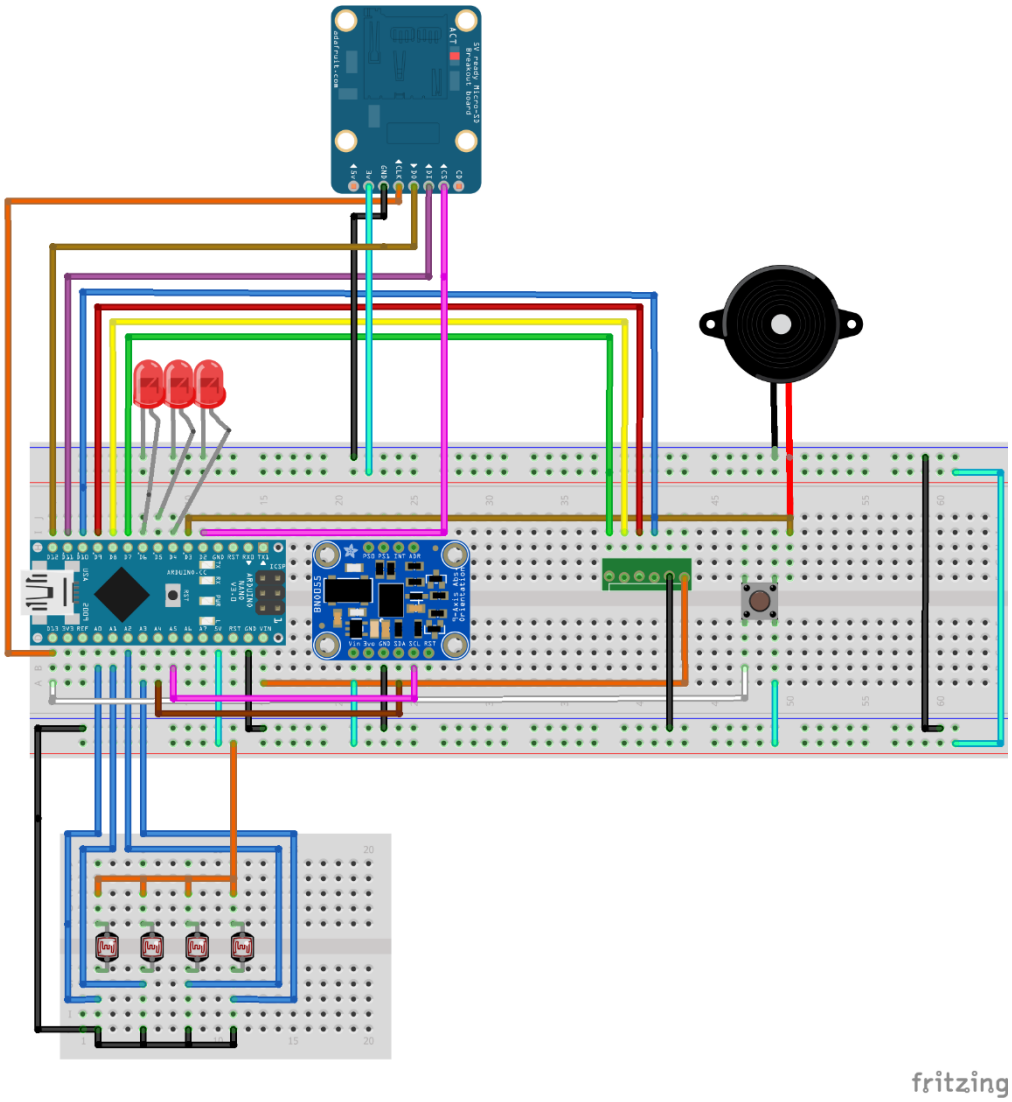


Figure 14.2 Schematic of the hardware connection

If the above diagram is hard to follow for replication purposes, do not worry! The team has created a secondary table to help with reconstruction of the project. Simply follow the table below, connecting a wire from the entry position on the left to the entry position on the right.

From RP2040	SD-card Reader
D12 CIPO	DO
D11 COPI	DI
D2	CS
3.3 V	3 V
GND	GND
From RP2040	D-SUB
D10	PIN 1
D9	PIN 2
D8	PIN 3
D7	PIN 4
VIN	PIN 6
GND	PIN 8
From RP2040	To LEDs and buzzer
D6	Green led
D5	yellow led
D4	Red led
D3	buzzer
GND	ALL MODULS (Cathode)
From RP2040	SOLAR TRACEKER
A0/D14	LDR 'UP' (Cathode)
A1/D15	LDR 'DOWN (Cathode)
A2/D16	LDR 'RIGTH' (Cathode)
A3/D17	LDR 'LEFT' (Cathode)
3.3 V	ALL LDRS (Anode)
GND	ALL LDRs (Cathode)
From RP2040	BNO055
3.3 V	VIN
GND	GND
D18	SDA
D19	SCL

14.3 Code

In the sections below there will be found an explanation of the code that was created for the second portion of the project work, as well as some clippings of the code itself.

14.3.1 Solar Tracking code

14.3.1.1 solar_tracking

```
void solar_tracking(){
  Serial.println("Im in the solar tracking function wee"); // Only foe debugging
purpose
// Reading the pins
  int topLeft = analogRead(Solar_1);
  int topRight = analogRead(Solar_2);
  int bottonLeft = analogRead(Solar_3);
  int bottomRight = analogRead(Solar_4);
// Making average
  int avgtop = ((topLeft+topRight)/2);
  int avgbot = ((bottomRight+bottonLeft)/2);
  int avgleft = ((topLeft+bottonLeft)/2);
  int avgright = ((topRight+bottomRight)/2);
// Solar Tracking Values
  Serial.println(avgtop);
  Serial.println(avgbot);
  Serial.println(avgleft);
  Serial.println(avgright);
// Solar tracking function is being called 4 times one for each direction
  solar_tracking_output(avgtop,avgbot,DSUB_UP,"Up","Top");
  solar_tracking_output(avgbot,avgtop,DSUB_DOWN,"Down","Bottom");
  solar_tracking_output(avgright,avgleft,DSUB_RIGHT,"Right","Left");
  solar_tracking_output(avgleft,avgright,DSUB_LEFT,"Left","Right");
}
```

The Solar tracking function is the main function for controlling the automated solar tracking software. Four values are being read from the light dependent resistors in the shadow box, they are processed into the average values and forwarded to the solar tracking output function.

14.3.1.2 solar_tracking_output

```
void solar_tracking_output(int value1, int value2, int value3,String value4,String
value5){
// This function compaires two values to determin what direction to turn the solar
panel
  if (value1 - value2 > Acceptable){
    digitalWrite (value3, HIGH);
    Serial.println("im going "+value4);
    delay(100);
  }
  else if (value1 - value2 < Acceptable){
    digitalWrite (value3,LOW);
    Serial.println(value5+" hit the mark");
    delay(100);
  }
}
```

The solar tracking output function takes the values and parameters forwarded by the main function and weigh then against a value of acceptance that determines if and what direction the solar tracker is moving.

14.3.2 Web-Interface code

14.3.2.1 wifi_module_setup

```
void wifi_module_setup(){
  // check for the WiFi module:
  if (WiFi.status() == WL_NO_MODULE) {
    Serial.println("Communication with WiFi module failed!");
    // don't continue
    while (true);
  }

  String fv = WiFi.firmwareVersion();
  if (fv < WIFI_FIRMWARE_LATEST_VERSION) {
    Serial.println("Please upgrade the firmware");
  }

  // by default the local IP address will be 192.168.4.1
  // you can override it with the following:
  // WiFi.config(IPAddress(10, 0, 0, 1));

  // print the network name (SSID);
  Serial.print("Creating access point named: ");
  Serial.println(ssid);

  // Create open network. Change this line if you want to create an WEP network:
  status = WiFi.beginAP(ssid, pass);
  if (status != WL_AP_LISTENING) {
    Serial.println("Creating access point failed");
    // don't continue
    while (true);
  }

  // wait 10 seconds for connection:
  delay(10000);

  // start the web server on port 80
  server.begin();

  // you're connected now, so print out the status
  printWiFiStatus();
}
```

The Wi-Fi module setup function sets up the Wi-Fi and starts the webserver on system start up.

14.3.2.2 wifi_module_status

```
void wifi_module_status(){
  // compare the previous status to the current status
  if (status != WiFi.status()) {
    // it has changed update the variable
    status = WiFi.status();

    if (status == WL_AP_CONNECTED) {
      // a device has connected to the AP
      Serial.println("Device connected to AP");
      Solar_Auto_OFF = true;
      digitalWrite(DSUB_UP, LOW);
      digitalWrite(DSUB_DOWN, LOW);
      digitalWrite(DSUB_RIGHT, LOW);
      digitalWrite(DSUB_LEFT, LOW);
    } else {
      // a device has disconnected from the AP, and we are back in listening mode
      Serial.println("Device disconnected from AP");
      Solar_Auto_OFF = false;
    }
  }
}
```

The Wi-Fi module status function is the main function for controlling the switch between manual and auto control, it looks for whether a device has been connect to its access point and switches to manual mode and set all the solar D-Sub pins to low to make sure there is no residual signaling from the solar tracker, and upon disconnection from an access point turns on auto mode.

14.3.2.3 wifi_module_server

```
void wifi_module_server(){

WiFiClient client = server.available();   // listen for incoming clients

   if (client) {                             // if you get a client,
    Serial.println("new client");           // print a message out the serial port
    String currentLine = "";                // make a String to hold incoming data
from the client
    while (client.connected()) {            // loop while the client's connected
      if (client.available()) {             // if there's bytes to read from the
client,
        char c = client.read();             // read a byte, then
        Serial.write(c);                    // print it out the serial monitor
        if (c == '\n') {                    // if the byte is a newline character

          // if the current line is blank, you got two newline characters in a row.
          // that's the end of the client HTTP request, so send a response:
          if (currentLine.length() == 0) {
            // HTTP headers always start with a response code (e.g. HTTP/1.1 200
OK)
```

```
            // and a content-type so the client knows what's coming, then a blank
line:
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println();
            client.print("<style>");
            client.print(".container {margin: 0; padding-top:0; border: none;");
            client.print("top: 0em; left: 0em; text-align: center;
position:fixed;width: 100%;}");
            client.print(".header {height: 4.5em; width: auto; background-color:
teal; text-align: left;}");
            client.print("h1 {position: absolute;color: white;");
            client.print("border: none;font-size:2.9em;margin: 0;padding: 0;top:
0.1em;left: 2em;}");
            client.print("button {color: white; width: 100px;height: 100px; border-
radius: 33%;margin: 20px; border: none;");
            client.print("font-size: 20px; outline: none; transition: all 0.2s;}");
            client.print(".red{background-color: rgb(196, 39, 39);}");
            client.print(".green{background-color: rgb(39, 121, 39);}");
            client.print(".blue {background-color: rgb(5, 87, 180);}");
            client.print("button:hover{cursor: pointer; opacity: 0.7;}");
            client.print(".toggle {position: absolute; top:0em;left: 0em;");
            client.print("</style>");
            client.print("<div class='container'>");
            client.print("<div class='header'> <h1>Solar Panel Control</h1>
</div>");
            client.print("<div class=''>");
            client.print("<button class='green' type='submit'
onmousedown='location.href=\"/UPLIMIT\"'>UP limit</button>");
            client.print("<button class='blue' type='submit'
onmousedown='location.href=\"/UPHIGH\"'>UP</button>");
            client.print("<button class='green' type='submit'
onmousedown='location.href=\"/RIGHTLIMIT\"'>RIGHT Limit</button>");
            client.print("<br>");
            client.print("<button class='blue' type='submit'
onmousedown='location.href=\"/LEFTHIGH\"'>LEFT</button>");
            client.print("<button class='red' type='submit'
onmousedown='location.href=\"/ALLSTOP\"'>STOP</button>");
            client.print("<button class='blue' type='submit'
onmousedown='location.href=\"/RIGHTHIGH\"'>RIGHT</button>");
            client.print("<br>");
            client.print("<button class='green' type='submit'
onmousedown='location.href=\"/DOWNLIMIT\"'>DOWN Limit </button>");
            client.print("<button class='blue' type='submit'
onmousedown='location.href=\"/DOWNHIGH\"'>DOWN</button>");
            client.print("<button class='green' type='submit'
onmousedown='location.href=\"/LEFTLIMIT\"'>LEFT Limit</button>");
            client.print("</div>");
            client.print("</div>");
```

```
            // The HTTP response ends with another blank line:
            client.println();
            // break out of the while loop:
            break;
          } else {    // if you got a newline, then clear currentLine:
            currentLine = "";
          }
        } else if (c != '\r') {  // if you got anything else but a carriage return
character,
          currentLine += c;      // add it to the end of the currentLine
        }

        // Ui directional control buttons
        if (currentLine.endsWith("GET /UPHIGH")){
          digitalWrite(DSUB_UP, HIGH); // Up Signal
        } else if (currentLine.endsWith("GET /DOWNHIGH")){
          digitalWrite(DSUB_DOWN, HIGH); // Down Signal
        } else if (currentLine.endsWith("GET /RIGHTHIGH")){
          digitalWrite(DSUB_RIGHT, HIGH); // Left Signal
        } else if (currentLine.endsWith("GET /LEFTHIGH")){
          digitalWrite(DSUB_LEFT, HIGH); // Right Signal
        }

        // Stop Signal Sets all pinout to LOW
        if (currentLine.endsWith("GET /ALLSTOP")) {
          digitalWrite(DSUB_UP, LOW);
          digitalWrite(DSUB_DOWN, LOW);
          digitalWrite(DSUB_RIGHT, LOW);
          digitalWrite(DSUB_LEFT, LOW);
        }

        // Ui set limits Buttons
        if (currentLine.endsWith("GET /UPLIMIT")){
          ta_UPPERBOUND = theta; // Up Limit
        } else if (currentLine.endsWith("GET /DOWNLIMIT")){
          ta_LOWERBOUND = theta; // Down Limit
        } else if (currentLine.endsWith("GET /RIGHTLIMIT")){
          phi_LOWERBOUND = phi; // Right Limit
        } else if (currentLine.endsWith("GET /LEFTLIMIT")){
          phi_UPPERBOUND = phi; // Up Signal
        }

      }
    }
    Serial.print("Up limit is now: ");
    Serial.println(ta_UPPERBOUND);
    Serial.print("Down limit is now: ");
    Serial.println(ta_LOWERBOUND);
    Serial.print("Right limit is now: ");
    Serial.println(phi_LOWERBOUND);
```

```
    Serial.print("Left limit is now: ");
    Serial.println(phi_UPPERBOUND);
    // close the connection:
    client.stop();
    Serial.println("client disconnected");
  }
}
```

The Wi-Fi module server function is the main code for the web interface and running the website. This code is responsible for posting and showing the Web-GUI and taking in inputs from the user to manually control the solar tracking system and setting up the limits for the safety system.

14.3.3 SD-Card Code

14.3.3.1 sdCard_Setup

```
void sdCard_Setup(){

  if (SD.begin(chipSelect)) {
      sdavailable = true;
      Serial.println("card initialized.");
      if (SD.exists("data.dat")){
              Serial.println("Calibration File found");

              dataFile = SD.open("data.dat", FILE_READ);

              adafruit_bno055_offsets_t calibrationData;
              sensor_t sensor;

              myIMU.getSensor(&sensor);
              dataFile.read((uint8_t *)&calibrationData, sizeof(calibrationData));

              dataFile.close();
              displaySensorOffsets(calibrationData);



              Serial.println("\n\nRestoring Calibration data to the BNO055...");
              myIMU.setSensorOffsets(calibrationData);
              Serial.println("\n\nCalibration data loaded into BNO055");
              foundCalib = true;

              /*---------Check Calibration-------------*/

              Serial.println("Checking Offset____________________");
                  adafruit_bno055_offsets_t checkOffset;
                      myIMU.getSensorOffsets(checkOffset);


              displaySensorOffsets(checkOffset);
```

```
            Serial.println("Checking Offset end_____________");
            /*-------------------------------------*/
      }
      else {
          Serial.println("No Calibration File found");
        }}
  else {
        Serial.println("Card failed, or not present");
        }
  delay(1000);
}
```

The sdCard_Setup function look to see if an SD-card is present and then it looks for a data.dat file that holds the calibration data for the BNO, if such a file is found it loads the data in to the BNO.

14.3.3.2 sdCard_newCali

```
void sdCard_newCali(){

  sensors_event_t event;
  myIMU.getEvent(&event);
  if (foundCalib){
      Serial.println("Move sensor slightly to calibrate magnetometers");
      while (!myIMU.isFullyCalibrated())
      {
          myIMU.getEvent(&event);
          displayCalStatus();
          Serial.println(" ");
          delay(BNO055_SAMPLERATE_DELAY_MS);

      }
  }
  else
  {
      Serial.println("Please Calibrate Sensor: ");
      while (!myIMU.isFullyCalibrated())
      {
          myIMU.getEvent(&event);

          Serial.print("X: ");
          Serial.print(event.orientation.x, 4);
          Serial.print("\tY: ");
          Serial.print(event.orientation.y, 4);
          Serial.print("\tZ: ");
          Serial.print(event.orientation.z, 4);

          /* Optional: Display calibration status */
          displayCalStatus();

          /* New line for the next sample */
          Serial.println("");
```

```
        /* Wait the specified delay before requesting new data */
        delay(BNO055_SAMPLERATE_DELAY_MS);
    }
  }

  Serial.println("\nFully calibrated!");
  Serial.println("--------------------------------");
  Serial.println("Calibration Results: ");
  adafruit_bno055_offsets_t newCalib;
  myIMU.getSensorOffsets(newCalib);
  displaySensorOffsets(newCalib);

  Serial.println("\n\nStoring calibration data to SDCARD...");

    if (!sdavailable) {
      Serial.println("Card failed, or not present");
      return;
  }

  Serial.println("card initialized.");
  dataFile = SD.open("data.dat", FILE_WRITE);

  dataFile.write((const uint8_t *)&newCalib, sizeof(newCalib));
  dataFile.close();

  Serial.println("Data stored to SD-Card.");

  Serial.println("\n--------------------------------\n");
  delay(500);

}
```

The sdCard_newCali function is responsible for controlling the calibration of the BNO, and calibrate it if necessary, it then takes the calibration data and stores it on SD-card for future calibration.

14.4 PCB Production

For the purposes of this project a PCB was created. The PCB has all of the same functionality that the breadboard would have but it is packed in a tighter, more compact, and resilient package.

14.4.1 Creating the PCB.

In the following segment you can read about the process of creating the PCB that the group went through for this project. A PCB was decided upon because it was a more resilient and professional looking package for the final product than another breadboard would have been.

14.4.1.1 Creating the board parameters

To start, the board outline was created to be 10 centimeters by 10 centimeters, to make certain that every component could fit on the PCB. It was important to make sure that there was enough size on the board for every component that the team needed for the project to function.

14.4.1.2 Create the stack up

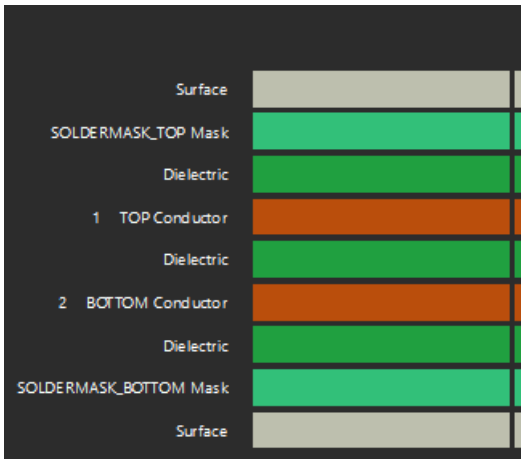


Figure 14.4.1.2. Stack up of the PCB

After the outline was created, some modifications to the stack up, also known as the cross section, had to be created. In the start of the project the board was created in four layers, but it was quickly realized that a two-layer board would be sufficient. Solder mask, top and bottom were also created in the stack up and dielectric layers was placed in between the different layer. This can be seen in figure 14.4.1.2 above.

14.4.1.3 Placement of the components

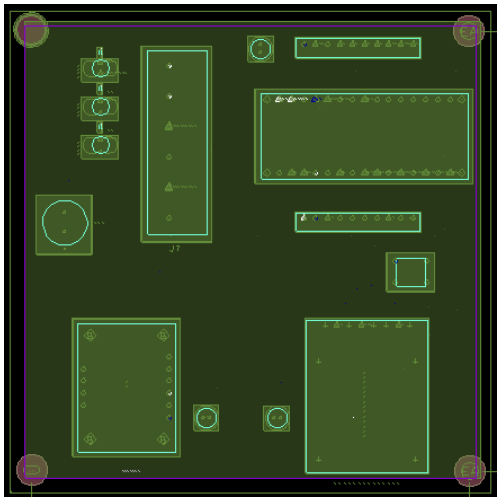


Figure 14.4.1.3. placement of the components on the PCB

Placement of components had to be thought through due to the fact that the Arduino RP2040 and the SD card reader had to be placed in such a way that it would be easy to insert the external devices there has to go into them. The rest of the components were placed so they had a fair amount of distance between each other so there would not be a problem with overheating, and to make soldering the components to the board easier. Resistors were placed near the modules that they are attached to. Placement of the components can be seen in figure 14.4.1.3 above.

14.4.1.4 Changing the physical constraints

Objects			Referenced Physical CSet	Line Width		Neck
Type	S	Name		Min	Max	Min Width
				mm	mm	mm
*	*	*	*	*	*	*
Dsn		▼ pcb-project	DEFAULT	0.500	0.000	0.100
PCS		▶ DEFAULT		0.500	0.000	0.100
PCS		▶ 01MM		0.100	0.000	0.100
PCS		▶ 05MM		0.500	0.000	0.300

Figure 14.4.1.4. cutout from the constraint manager

In the physical constraints a default setting for a line width was set to be 0.5 mm to make sure that no trace would be unable to supply the signal going through it.

14.4.1.5 Connecting the tracks

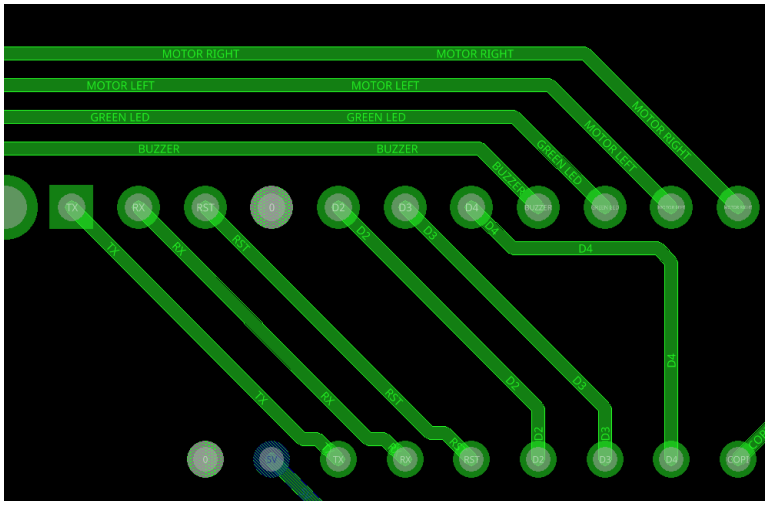


Figure 14.4.1.5. trace routes for a piece of the PCB

When connecting all the traces the group made sure of that none of the wires had any 90 degree turns because this could lead to manufacturing mistakes, so all the corners were rounded off to be 45 degrees minimum. An attempt was made to find the shortest and most direct route to implement. An example can be seen in figure 14.4.1.5 above.

14.4.1.6 STEP mapping

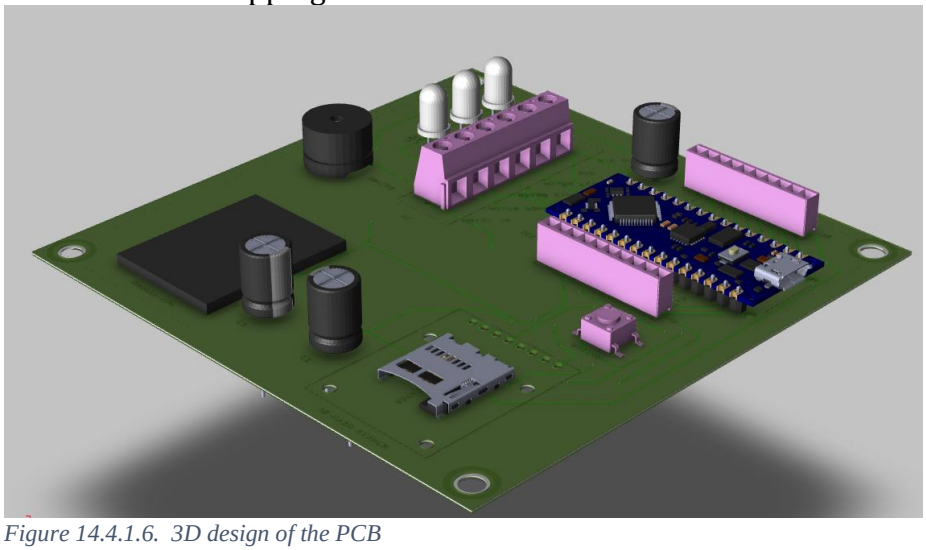


Figure 14.4.1.6. 3D design of the PCB

To make sure that none of the components that had been chosen overlapped with each other the process of step mapping began. Similar components for the ones used were found online and inserted onto the PCB to make a reality check that there was actually space for every component that had been placed onto the PCB.

14.4.1.7 Creating the silkscreen top

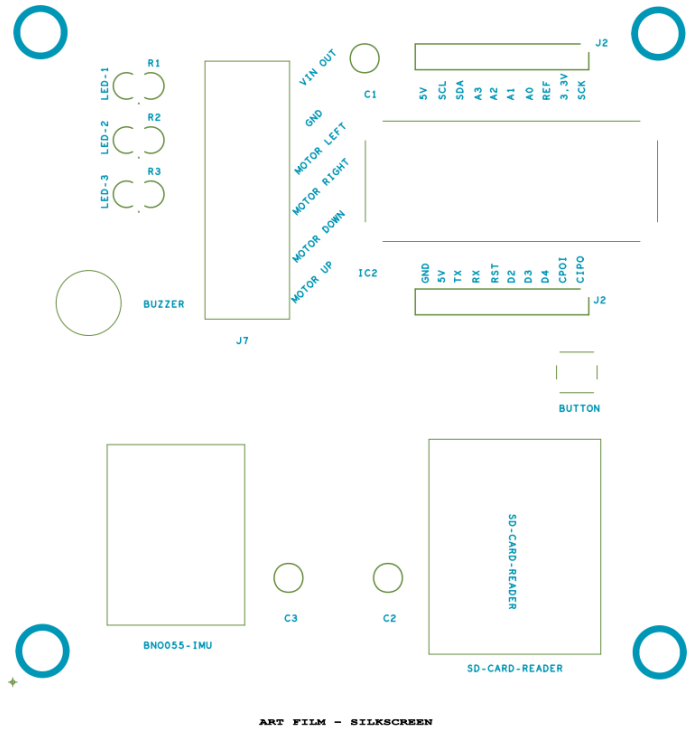


Figure 14.4.1.7 The PCBs silkscreen top layer.

To make it easier and faster to assemble the PCB when it arrived, a choice of making a very detailed silkscreen top was decided. Insertion of the the text on the different pinouts has been added, so users of the product will have an easy and understandable way to connect new devices to the developer board.

14.4.1.8 Dimension environment

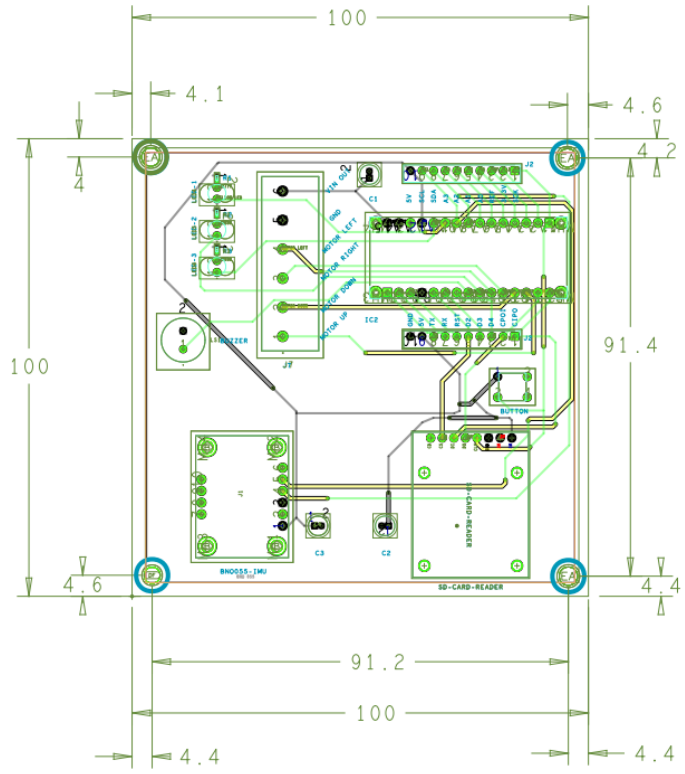


Figure 14.4.1.8. A picture of the PCB with dimensions

Due to the fact that the PCB must be used in this project, the dimension environment tools was utilized to make some measurements of where the holes in the PCB are, so a mobile containment unit could be created for it.

14.4.2 Ordering the PCB.

The following section will detail how to order the PCB if there is a desire to replicate the system without reconstructing the breadboard. If you would like to order the PCB that was designed for this project, you can find the gerber files (the .art and the .drl files) already pre-zipped and ready to deliver to your PCB Creation Company of choice. (We recommend JLCPCB because they made the ones that were used for this project, and they worked quite well for the purposes of this project.)

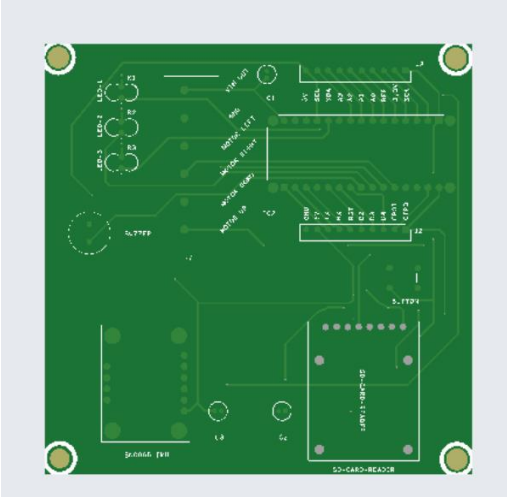
14.4.2.1 Generating Drill and gerber Files

BOTTOM.art	ART File
nc_param.txt	Text Document
nc_tools_auto.txt	Text Document
outline.art	ART File
pcb-project-1-2.drl	DRL File
pcb-project-1-2-np.drl	DRL File
PLANE_2.art	ART File
SILKSCREEN.art	ART File
SODERMASKBOTTOM.art	ART File
SODERMASKTOP.art	ART File
TOP.art	ART File

Figur 14.4.2.1. Drill and gerber artwork

When ordering a PCB it is very important to generate the correct drilling and Gerber files. If you do not then you will end up with a PCB that does not contain, for example, a solder mask or silkscreen top. In the picture above it can be seen what artworks that that have been chosen to include in the Gerber files.

14.4.2.2 Ordering the PCB from JLCPCB.



Figur 14.4.2.2. the PCB from JLCPCB Gerber viewer

When ordering a PCB from JLCPCB the Gerber and drill files must be zipped, Hereafter A possibility of viewing the Gerber and drill files in their online Gerber viewer a given, and when the PCB design and the Gerber viewer looks similar then it is time to order the PCB.

14.4.2.3 Correspond with JLCPCB

If there for some odd reason, should be a problem with the PCB, then JLCPCP has a client service that makes sure that all PCB are looked over by a competent worker, and they are informing about any inconvenience that might be with the PCB design that they're looking at. in case of this project the silkscreen top had been forgotten in the first Gerber and information about this was relayed and fixed.

14.5 3D Printed Enclosure and Shadowbox

The Minimum Viable Product works but during unit testing the group realized that the device would be more secure if there were a protective covering for it. Additionally, there was needed some way to secure a shadowbox to the solar panel for the purpose of correctly tracking the location of the sun. To this end the group has designed a container for the printed circuit board and a shadowbox module to attach to the solar panel.

14.5.1 First Draft Containment Unit

The first thing that the group would need was a containment unit for the PCB. The original model was found on Thing verse, edited in Free CAD, and printed using UCL's FabLab. The first print worked well to keep dust and water off the sensitive electronics and securely hold the PCB, but the first draft had a critical flaw in its design which sent the team back to the drawing board.

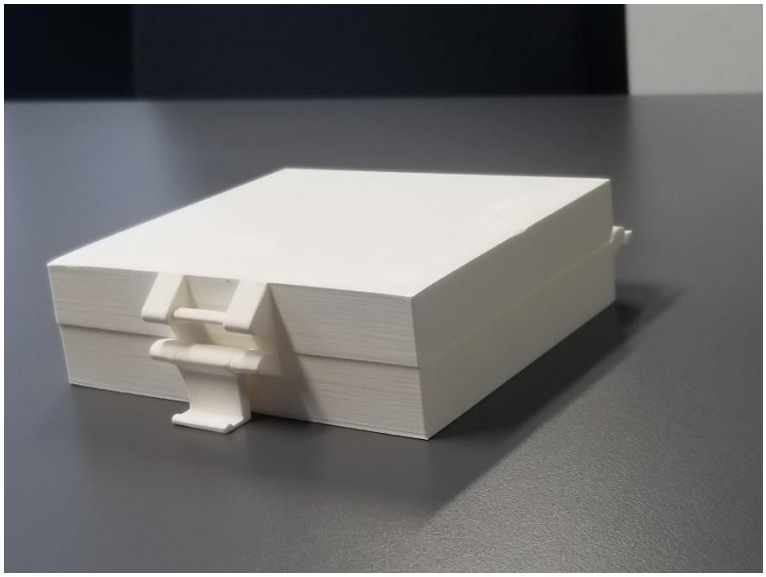


Figure 14.5.1.1 First Draft Containment Unit

The critical flaw was in the design of the hinges. As shown in the following diagram neither set of hinges can move, which means that the box cannot be closed or locked. In fact, to even get the containment unit in a presentable state it had to be broken. With these failures in mind and with a few new design necessities in mind the group went back to work redesigning the containment unit.

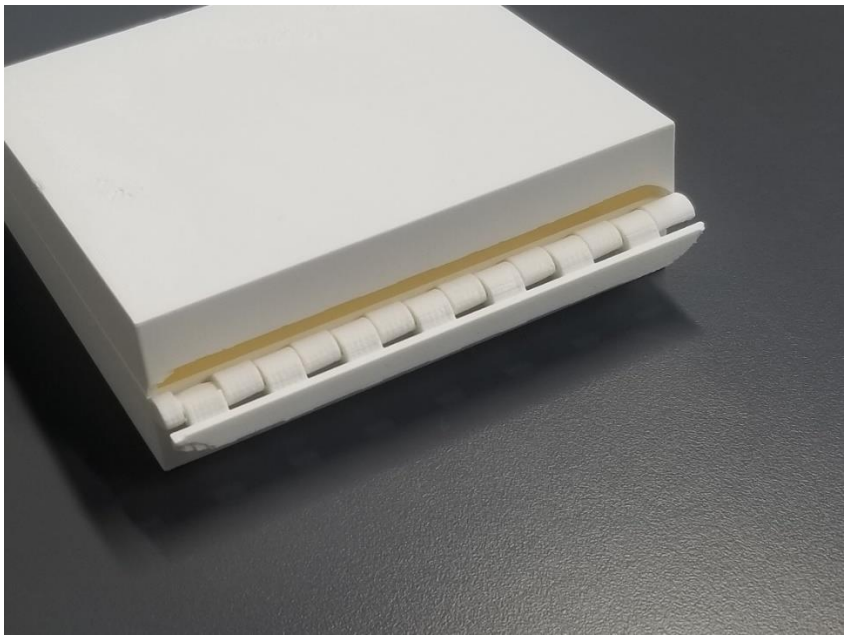


Figure 14.5.1.2 First Draft Containment Unit

14.5.2 First Draft Shadowbox

The second piece of 3D printed hardware that was developed was the Shadowbox. From early on in development it was known that the desired solution to the design problem of how to track the sun was a shadowbox, but the specifics of the module were unknown. Once the module was designed and printed, however, a slight problem presented itself. The positioning that the team desired for the shadowbox did not mesh with the wiring that was required for the device to connect so the group designed a mount for it. The mount features a hole in the base which allows the wiring to run from the Shadowbox mounted on the solar panel to the RP2040. The Shadowbox in the first draft is connected to the mount via some glue.

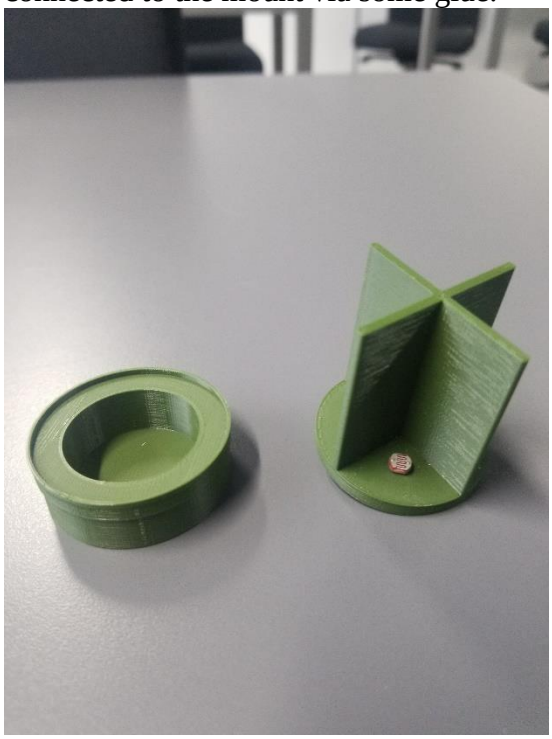


Figure 14.5.2.1 First Draft Shadow Box

The combined shadowbox and mount modules have a couple of issues that came up after initial printing that need to be addressed in a second draft. The position where the shadowbox connects to the mount module needs to be higher by a couple of millimeters to make running wires through easier. The wall of the mount needs to be thinner by a few millimeters as well. Finally, the connection between the shadowbox needs to be a tighter fit which means the second draft needs to be a tighter fit. This means tightening the circumference of the circular containment wall on the mount by a few millimeters.

Figure 14.5.2.2 First Draft Shadow Box

14.5.3 Final Draft Containment Unit

The second and final version of the Containment Unit for the PCB turned out nicely. Where the first draft had a critical failure upon closing and an inability to latch shut the Final draft can open, close and latch. Additionally, the stands to hold the PCB up are lower to account for the thermal block on the PCB. The containment unit is not perfect, and if a third draft were to be designed more attention would be paid to the size of the parts that were 3D Printed. The two halves of the containment unit are not perfectly symmetrical and the rods that make up the hinges are too big. This of course does not detract from its ability to serve as a Minimum Viable Product. The containment unit does work for its intended purpose of holding and securing the box against atmospheric hazards.

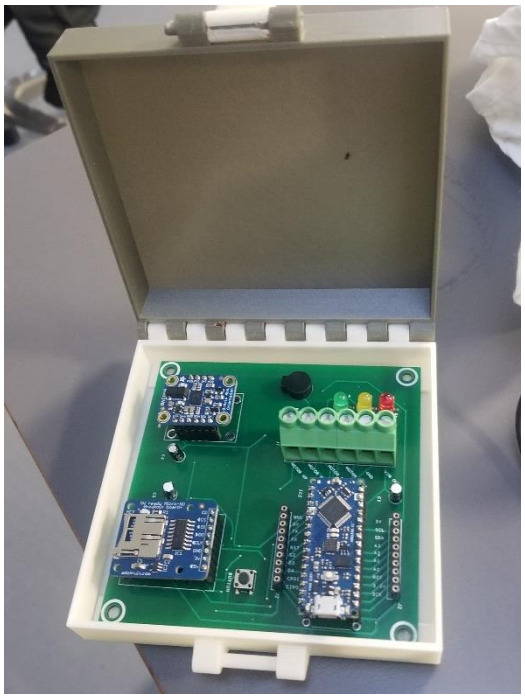


Figure 14.5.3 Final Draft Containment Unit

14.5.4 Final Draft Shadow Box

The second shadow box turned out very nicely. The first draft was already quite good, so the second printing only had a few minor issues to improve upon. The major difference between the two drafts is the Cup Module. The second draft is taller with thinner walls which gives enough room for all the required wires to run through the Shadowbox Cup and towards the PCB.

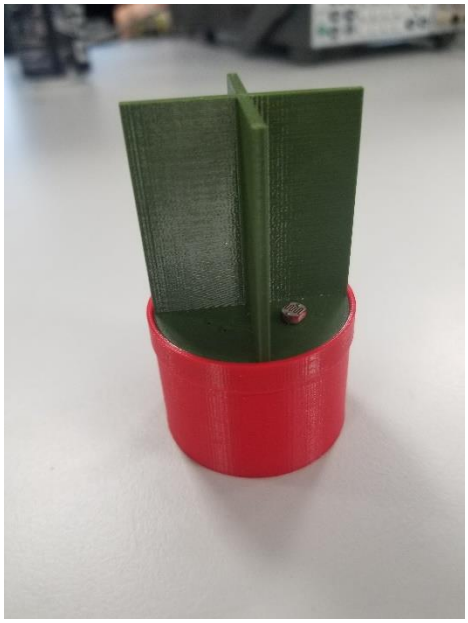


Figure 14.5.4 Final Draft Shadow Box

15 Risk Analysis

Risks	Likelihood Scale: 1-10 10= high risk	Severity Scale: 1-10 10= high risk	Product of Likelihood and Severity (1-100)	Risk Mitigation Preventive and Responsive actions
The system is not detecting or integrating properly.	5	10	50	Rechecking the connections and testing the code.
Unit calibrated in a wrong way.	4	8	32	Making sure to get the measurement correctly.
Inability to get necessary materials for crafting	6	8	48	Plan to use more common components
Another lockdown.	4	6	24	WFM, be available always.
Failure of the team merger	4	8	32	Work harder to hold each other accountable for work

Here is the more detailed overview of risk management:

15.1 The system is not detecting or integrating properly

There is always the chance that the developers might miss a small but important detail which could slow down or even stop forward momentum for the entire project if not detected early enough in the project. For example, the team is using a PCB, which must be designed and ordered a week or more in advance. The entire process of ordering and resubmitting PCBs can take weeks if a mistake is made. To help prevent any mistakes from occurring two team members made separate PCBs for the project. They compared their results before ordering them therefore the two separate design philosophies helped the team to mitigate any mistakes before the physical PCBs were ordered.

15.2 Unit calibrated in a wrong way.

The entire project would fall apart if the unit were calibrated incorrectly. Incorrect Unit Calibrations has been a problem that the group had run into at one point regarding the BNO055 regarding positional drift, but a solution was found. The team has been vigilantly looking for errors stemming from incorrect unit calibration specifically because it was something that had come up during the brainstorm as a possibility.

15.3 Inability to get necessary materials for crafting

To prevent this issue from happening the team has first done a lot of research into the available components online to be sure they are not out of stock. Additionally, the team members have mostly chosen to work with the materials that they already had access to from other class projects or earlier classes, replacing them with more advanced and relevant pieces as they were acquired. This meant that the minimum viable product could potentially be made without receiving any new equipment.

15.4 Another lockdown.

With the rising of COVID 19 cases in Denmark there is a greater risk of the future lockdowns, which could become a big issue considering that the team did not have all the components for the hardware each team member might need to construct for the purposes of working on the project. Luckily, the Lockdown did not happen, but the team still made sure to prepare for it.
The online communication channels suitable for everybody have been established. The tasks were divided in a way in which some team members could bring components home to work on the project while others could continue contributing to the project's progress without having them.

15.5 Failure of the team merger

The two parts of the team were working on two distinct phases of the project before the merger occurred, so it was important to make sure that everybody was on the same page and up to date with everything related to the primary team's work and research. The team all had to agree on future steps and improvements for the project's future.

The thing that has been done for the project has been the Gitlab issue board for system recreation. Each of the new team members had to recreate the first portion of the project using the recreation guide. When the new team members had all reconstructed the Proof of Concept the Big Brainstorm occurred. This is where the team decided on the future workload and improvements for the team's work. The team also decided upon the project management methods that would be used.

Overall, the team seemed to agree that the team merger went well. The risk of leaving somebody behind has been successfully managed and the workload seems to have been evenly distributed to ease the workload for the entire team.

16 Milestone

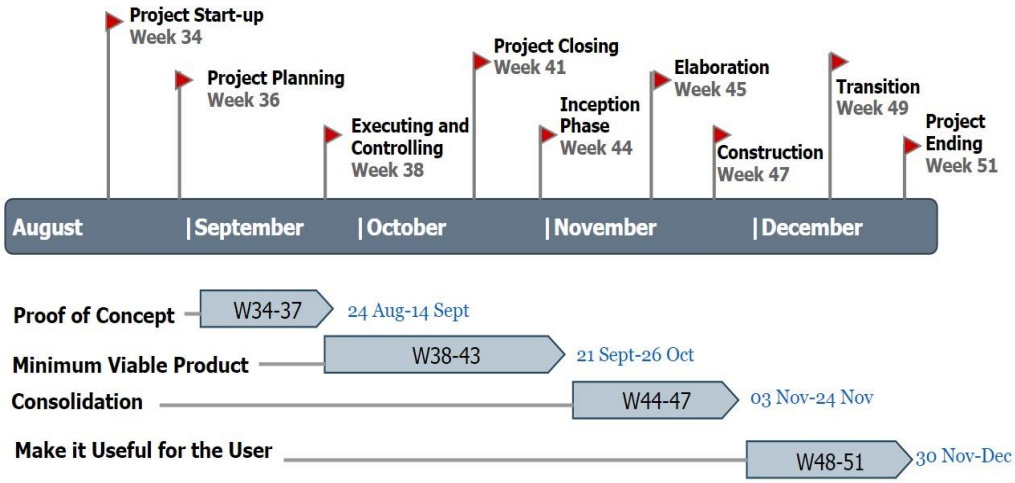


Figure 16 Milestones

The figure above shows the complete overview of the development of the project over the course of this semester. Week 44 to Week 51 covers the second portion of the project which is explained in detail in the Unified Process chapter.

In reference to project part 1, where the team's goal was to find solution for the limitation on solar panel movement. The goal of the second part is to make further improvements by making a functional solar tracking system.
For the second stage of the project, the group achieved their primary deliverable by following a strict timeline to ensure that they stay on track to reach each milestone's deadline.
They created a minimum viable product, where they have expanded the solutions for completing the entire project.
Finally, the group successfully consolidated each part and made them into a single coherent whole.
Therefore, as a result the group managed to reach the goal of making the project useful for the users and completed the learning goals for the project by the end of this semester.

17 Challenges

There have been many challenges that the team could not have planned for. In the following section a few of the largest challenges that have come up will be explained in more detail.

17.1 The change of the MCU

During the process of making the project the team produced additional ideas and improvements that they would like to implement. This would not have been possible with the MCU from the first portion of the project and so the team discussed getting a second option for a microcontroller.
That meant the team had to align the PCBs with the new unit, and needed to design a second PCB that used the second microcontroller just in case.

17.2 New components

The group sat down and brainstormed about what improvements they could make to the project for the second phase, especially since the first phase had gone so well for them. One of the things most agreed upon was that there were components that were better suited for what the group was attempting to do than the old components that they had worked with all semesters. While this idea has borne fruit and the new components have improved the project quite a bit, there was a learning curve regarding the new components, both in figuring out how to best integrate them into the system as a whole and learning how to use and program with them since a lot of them were not fully compatible with old code. For a board like the Arduino RP2040, which had some features that could be useful in the project, we would have to start from nothing through all its features so that we could get to know how it all works.

17.3 The 3D printing

3D Printing is a troublesome process to get right, and the team went around in circles trying to get a proper 3D print for the various pieces of hardware that the team had. The Containment Unit for the solar panel controller took two tries to get even close to functional and correct, and there are still multiple upgrades and improvements that could be made to make the container work better for the purposes of the project. The Shadowbox came out much better but even that piece has improvements that could be made if there were a further phase of work being done in regard to this project.

17.4 The BNO calibration

The problem that the group experienced with the BNO was a lack of documentation. The module needs a full system calibration on startup, which is an annoying problem but completely manageable on its own.

The problem that our group ran into was that the magnometer was unable to access and use the stored inputs from previously stored calibrations. This then triggers ANOTHER unnecessary whole device calibration which wastes device uptime as well as device memory. There was not a good workaround for this because the two sensors that were necessary for the project would not function without calibrations from the magnometer.

The group’s long-term solution is to replace the BNO055 because this is a problem with the microchip, not a problem that can be programmed out of. Before that juncture, however, is to tell potential users that they will need to calibrate on startup for the device to function correctly.

17.5 The RP2040 Nano Connect calibration

Calibration has been one of the group’s most consistent issue. One of the reasons as to why the group decided to use the rp2040 nano connect was because of its inbuilt IMU which meant we would eliminate the BNO and work on a more compact module. This was not the case due to the calibration issue which meant the group would need to calibrate just as they did with the BNO module. This led the group back to the original idea (the BNO).

There was also not a lot of documentation on calibration or even the module itself online which made it a bit difficult to work with the IMU part of the Nano Connect. It would have been nice to figure it out because it would leave the group with a lot of possibilities in case the group decided to take the project further for example, its step converter, the microphone and using it for Machine

Learning opportunities with our Solar Tracker system.

18 Future Improvements

Although this is the end of the Project the team still has some improvement ideas for the future both regarding the actual product they have built and the overall teamwork and structure in general.

So here is the list of the improvements that can be made:

18.1 Mobile Application

The idea behind the Mobile Application was the convenience of being able to use it to access and work with the solar panel from the comfort of home, but that requires an internet connection which is not secure to use without any additional security measures. As it would require a lot of additional time and a lot of additional work the team decided to switch to instead using the local WIFI without internet connection. Using the local network means that you still must still be within the range of the Wi-Fi signal, which means that it loses the original purpose of making it.

Of course, the concept would still be beneficial to use, especially if for example we have multiple solar panels which might be able to possibly be controlled at the same time. The idea is not closed off forever, so it is one of the "nice to have" future improvements.

18.2 Better code scripts

At the current juncture in the project's code a lot of things are hard coded to work. An improvement that it would be nice to have would be dividing the programming into classes and redesigning the code with the idea of Object-Oriented Programming. This would make the code cleaner and easier to read as well as simpler to improve upon in the future.

18.3 Teamwork

Since the last report was written a lot of things have been improved regarding the group's teamwork and project management, but there are still some improvements to be made. For example, time management. The team finished the project on time, but the time spent could possibly be managed in a more efficient way. Although the Scrums and sprint meetings were in place the team could use the Gitlab page more efficiently, so even if people are not present due to sickness or other circumstances, they still could have a better overview of the things they missed and have a better possibility to work from home.

18.4 Replacing the BNO with the newer version

As mentioned earlier the current version of the BNO cannot store magnometer data or function without it which wastes valuable operating time and the time of the end user. One solution that was brainstormed was to upgrade from the BNO055 to the BNO080, a newer, more powerful version of the chip. The hope would be that an updated version of the chip would have fixed that problem. Overall, our project does not require a magnometer at all, but the group was forced to use one due to the BNO's calibration issues.

19 Gitlab Project Page

The following link leads directly to the GitLab Repository for the Solar Panel project. There the schematics, code and other relevant information regarding this project can be found.
https://gitlab.com/21a-itt3-project-student-group/itt3_solar_panel_group2

20 Conclusion

At the second part of the project the team has successfully managed to complete the goals they have placed for themselves.

This report presents the design and realization of a Solar Tracking System which is in place, meaning the solar panel can track the sun and supply power in the most efficient way. The tracking controller is implemented by means of a RP2040 controller and a BNO055.

The Graphical User Interface has been improved from the first portion of the project. It is more streamlined and there is now a way to hold positional data.

The PCBs were successfully produced. This means not only is it easier for the group to work on the hardware of the project, but also the recreation guide for potential users is simplified because there is no reconstruction of messy breadboards or difficult wiring required. This means that the user can order a PCB instead of doing the troublesome work of recreation themselves which means the probability for an error has been decreased.

Although there were challenges along the way the team has successfully managed to work through them all. Additionally, all the planned parts of the project have been completed.

The Future Improvements are not essential part of the project and mostly nice to have, because there is always something more to add to the things you work with. Replacing some of the components with the newer versions is one of the things that would also be useful to do. So, although there are more improvements the team wishes to make, but the team has completed the original purpose of the project.

21 Literature

Webpages:

https://store.arduino.cc/products/arduino-nano-every
https://www.tandyonline.com/adafruit-9-dof-absolute-orientation-imu-fusion-breakout-bno055.html
https://www.youtube.com/watch?v=tv00NfjB46c
https://www.mindtools.com/pages/article/newTMC_07.htm
https://gitlab.com/21a-itt3-project-student-group/itt3_solar_panel_group2

Additional links used for resources:

https://www.mindtools.com/pages/article/newTMC_07.htm
https://toptechboy.com
https://www.hackster.io/hardikrathod/control-arduino-using-gui-arduino-processing-2c9c6c
DIY Solar Tracker || How much solar energy can it save?
https://youtu.be/_6QIutZfsFs/
https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview
https://github.com/adafruit/Adafruit_BNO055
https://processing.org/tutorials
https://docs.arduino.cc/hardware/nano-every
https://gitlab.com/21a-itt3-project-student-group/itt3_solar_panel_group2/-/blob/main/Documentation/solar_panel_joystick_controller_Cicuite.pdf
https://www.arduino.cc/en/software
https://processing.org/download
https://www.thingiverse.com/thing:4491069 - First Draft Containment Unit
https://www.instructables.com/DIY-Miniature-Solar-Tracker/ - Shadow Box Design

Documents:

https://cdn-learn.adafruit.com/assets/assets/000/036/832/original/BST_BNO055_DS000_14.pdf
https://cdn-learn.adafruit.com/downloads/pdf/adafruit-bno055-absolute-orientation-sensor.pdf
http://wiki.amperka.ru/_media/products:arduino-nano-every:atmega4809-datasheet.pdf