****

**Design & Innovation Project (DIP)**

**Project Final Report**

**Prototyping of a solar-powered charging kiosk for portable power banks**

**Project Group: E003**

|  |  |
| --- | --- |
| **Na Shi Chen** | **U1722168J** |
| **Lai Wen Pang** | **U1720911H** |
| **Foo Ce Jun Joel** | **U1721298H** |
| **Ernest Low Qi En** | **U1721084H** |
| **Chong Mei Ling, Cindy** | **U1721973C** |
| **Lim Kim Yew** | **U1622681E** |
| **Wang Zi** | **U1622649L** |
| **Lim Li Ting, Rachel** | **U1620655D** |
| **Stacy Ng Yan Rong** | **U1622185E** |
| **Hernandez Nevin Miguel Livelo** | **U1620770F** |

**School of Electrical and Electronic Engineering**

**Academic Year 18/19**

**Semester 1**

Table of Contents

[List of Tables 3](#_Toc530122622)

[Abstracts 4](#_Toc530122623)

[Acknowledgments 4](#_Toc530122624)

[Introduction 4](#_Toc530122625)

[Project Description 4](#_Toc530122626)

[Objective 5](#_Toc530122627)

[Milestone 5](#_Toc530122628)

[Schedule 6](#_Toc530122629)

[Product Design 7](#_Toc530122630)

[Overview 7](#_Toc530122631)

[Challenges 7](#_Toc530122632)

[Outcome 11](#_Toc530122633)

[Software Design and Development 12](#_Toc530122634)

[How does our component work? 13](#_Toc530122635)

[Challenges 16](#_Toc530122636)

[Electrical Design 17](#_Toc530122637)

[Analysis and Finding 20](#_Toc530122638)

[Challenges 20](#_Toc530122639)

[Cost Analysis 25](#_Toc530122640)

[Lesson Learned 26](#_Toc530122641)

[Future Work 27](#_Toc530122642)

[References 28](#_Toc530122643)

[Appendix A 29](#_Toc530122644)

[Appendix B 29](#_Toc530122645)

[Appendix C 30](#_Toc530122646)

[Appendix D 32](#_Toc530122647)

[Appendix E 35](#_Toc530122648)

[Appendix F 41](#_Toc530122649)

[Appendix G 51](#_Toc530122650)

[Appendix H 52](#_Toc530122651)

[Appendix I 53](#_Toc530122652)

[Appendix J 54](#_Toc530122653)

[Appendix K 56](#_Toc530122654)

[Appendix L 59](#_Toc530122655)

[Appendix M 60](#_Toc530122656)

[Appendix N 61](#_Toc530122657)

List of Figures

[Figure 1. Gantt Chart for E003 Project term 5](#_Toc530002941)

[Figure 2. Prototype Sketch Version 1 7](#_Toc530002942)

[Figure 3. Prototype design development phase 8](#_Toc530002943)

[Figure 4. Prestech PTP 3030-8 9](#_Toc530002944)

[Figure 5. Pillar support for electronic component mounting 10](#_Toc530002945)

[Figure 6. Finalized Prototype 11](#_Toc530002946)

[Figure 7. Flow Chart for Solar Kiosk 12](#_Toc530002947)

[Figure 8. System Overview 13](#_Toc530002948)

[Figure 9. Touch Screen User Interface 13](#_Toc530002949)

[Figure 10. Circuit Diagram 18](#_Toc530002950)

[Figure 11. Impedance across 2 Arduino Mega2560 20](#_Toc530002951)

[Figure 12. Arduino Mega2560 Schematic Diagram 21](#_Toc530002952)

[Figure 13. Circuit Diagram for Lead Acid Direct Power supply 21](#_Toc530002953)

[Figure 14. Lead Acid Battery Supply only 22](#_Toc530002954)

[Figure 15. Normal Operation Measured Current 22](#_Toc530002955)

[Figure 16. Lead Acid Charging Method Figure 17. 12V 48Ah Lead Acid Battery Voltage 23](#_Toc530002956)

[Figure 18. 4 Solar Panel to system circuit diagram 24](#_Toc530002957)

# List of Tables

[Table 1. Arduino Mega2560 Front End 14](#_Toc530002958)

[Table 2. Arduino Mega2560 Back End 15](#_Toc530002959)

[Table 3. Arduino Serial Communication Protocol 17](#_Toc530002960)

[Table 4. System Operating Power Consumption 19](#_Toc530002961)

[Table 5. Cost Analysis Breakdown 25](#_Toc530002962)

# Abstracts

Solar energy is all around us and we can harness energy out of light. By converting light wavelength into electrical energy through the help of a semiconductor. The semiconductor is able to excite through light or thermal excitation that would create electron-hole pairs and as a result, would allow charge to flow in a direction that current is able to be produced.

Solar power is the conversion of sunlight into electrical power, and it is currently the most available source of energy. Conversion of energy can be easily done using Photovoltaics’. Also referred to as solar electric, Photovoltaics’ offers the ability to generate electricity in a clean, quiet and renewable way. It makes use of the abundant energy from the sun, to generate electricity without the production of harmful carbon dioxide (CO2) emissions. Solar energy is used to charge the battery for various applications, for example, mobile phones, electric vehicles etc. However, the output voltage of the solar panel is unstable. To increase the reliability, we integrate the whole system with a battery and used a solar charge controller to ensure stability and reliability of the whole system.

# Acknowledgments

I would like to express my gratitude to Dr. Foo Yi Shyh Eddy, Mdm Chia-Nge Tak Heng ,and Ms Lin Zhiren for their continuous assistance and support towards the completion of the project.

# Introduction

## Project Description

The increasing demands from the public have changed the rental industry entirely. As a result, technopreneur, entrepreneur and engineers have to come up with innovative and creative ideas to provide rental services to our consumer. In this modern world, smartphones, iPad and tablets are a norm in our daily lives and due to it being light the weight of the battery reduces proportionally with it, as a result, the electrical device’s battery will deplete exponential depending on the usage. Therefore, with the setup of the power bank rental station, we can provide an immediate solution to our consumer and we offer to bring convenience and satisfactory result to them.

Our power bank rental station offers our consumer of charging on the go capability and mount function that would make you feel more comfortable when holding onto your phone which would reduce the hassle of power bank obstructing your movement. We have a reward system where our consumer can earn points through our rental services and exchange this point in food and beverage outlet (F&B) for discount upon purchase.

As part of the green initiative, solar Photovoltaics (PVs) have been commonly used in many ways. One of the examples is the innovative bike where it was used to power up the mechanism to lock or unlock the bicycle. However, most solar systems had to compliment with a battery due to inconsistent weather condition and constraint hours with sunlight. With proper modelling of the system, we could achieve a high reliability of the system. Thus, our kiosk station is unique from current existing power bank rental kiosk station as we consume our power through solar energy and as a result it can be retrofitted into any environment such as school, parks and so on. Another advantage of the power bank is it can be shared amongst the people where we promote a sharing economy.

Over the years, Solar panels have matured significantly such that it has been proven to be cost effective in powering up houses with a very ideal payback period. It is also commonly used to cover up the gaps for places that do not have accessibility to the main power grid.

Upon recognizing the potential of the solar panels and sharing economy we are able to work towards the sustainability and eco-friendly concept. As part of this project, my team and I are expected to build a prototype of the power bank rental kiosk (chagrining station) with solar panel integration and optimize its capability in term of charging capabilities. We will work in the Clean Energy Research Laboratory (CERL) located at S2-b7c-05 during this project.

## Objective

In Singapore context, we are working towards a sustainable and smart nation which harnessing energy from the environment is part of the plan. Thus, by leveraging the Sun during the day to harvest the solar energy then convert to electrical energy to store it in a battery to increase the system reliability and likewise to do parallel consumption to the load we can work towards the sustainability. Through our effort, we aimed to help school, office place and household to reduce their electrical bills significantly. Our advantage is in the current market there is not many solar kiosk station that dispense power bank located around the world and since Singapore is a populated and developed country it serve as a good location to have this type of machine being set up around the city.

## Milestone

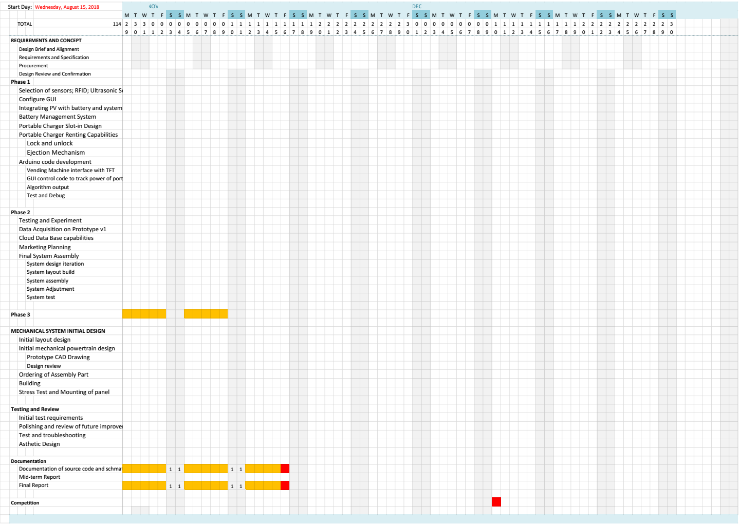
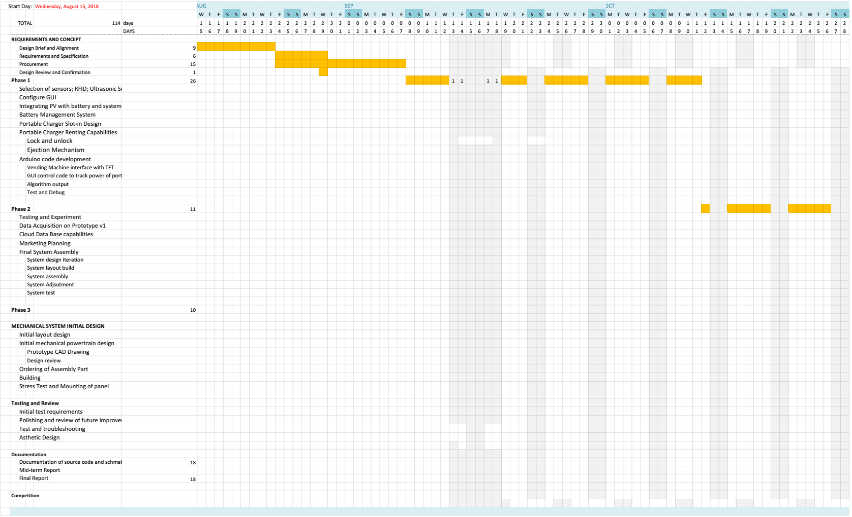


Figure 1. Gantt Chart for E003 Project term

To work towards the end goal of our project, firstly I split our team into different teams which are Software Development Team, Mechanical Team and Electrical Team (Appendix A) then I divided the project into 3 different phases and within this 3 phases there are several sub-phases that we would need to accomplished before we can move on to the next main phase.

During phase 1, we aligned our main objective of the project by identifying our version 1 of the prototype key features.

1. 1 x 45-Watt panel to power the whole system
2. 1 x 12Ah 12V Lead Acid Battery
3. 1 x 4200mAh Power Bank
4. Authentication Function
5. Graphical User Interface for rent and return process

For phase 2, we planned to build and develop a data base to store user data through this data we can then improve our system functionality and improve our application that user can rent and return power bank with convenience.

For phase 3, we planned to create a network system where we set up rental station over the country they are able to identify at which location the user is able to return their power bank to with convenience as well as giving us the live update of how many power bank is left within the station so we are able to deploy manpower to do the necessary top up to the station. Aside from this, we will be able to interlink all the machine together and create a network map to trace our power bank after it is installed with tracking module.

## Schedule

|  |  |  |
| --- | --- | --- |
| **PHASE** | **Planned Milestone Date (2018)** | **Actual Milestone Date (2018)** |
| **Initiating Phase** | From 13 Aug – 24 Aug (Week 1 -2) | 23 Aug 2018 (week 1) |
| **Planning Phase** | From 27 Aug - 7 Sep (Week 2 – 4)  Expected to finalize all planning at 7 Sep | 14 Sep 2018 (Week 2- 4) |
| **Execution Phase** | From 7 Sep – 12 Oct (Week 4 -8) | 16 Oct (week 4 – 9) |
| **Closing Phase** | From 12 Oct – 30 Oct (Week 8 – 11) | 7 Nov (Week 12) |
| **Project End Date** | From 31 Oct -16 Nov (Week 11 – 13) | 16 Nov (Week 13) |

During the initiating phase, we were familiarizing with the rest of the project members and introducing all the individual task and roles to achieve project success.

During the Planning Phase, after all the roles and responsibilities have been sorted out during the first 2 weeks. We began to draft our work plan for the project schedule with reference to the project specifications and the given deadline. Since we are given with a tight schedule we tried to keep our product as simple as possible as we intend to build a working physical prototype as our end goal. During the planning phase, originally, we planned to complete at 7 September but due to several changes made to the design draft, we had to postpone the assembly process. However, to reduce the amount of workload concurrently we went to procure all the confirmed and necessary materials.

During the Execution Phase, we took reference from the blueprint which we finalized during the planning phase and work towards the objective. Our project team was expected to finish our phase 1 by the end of week 8 of the academic schedule but due to numerous technical problem arises we had to delay progress. In the end, we only managed to complete phase 1 from the original plan to get the full working prototype due to time constraint.

During the Closing Phase, we had to troubleshoot, and test runs the prototype before we close the whole project. During the progress, we met with a few accidents such as accidentally burned electronic components, insufficient power to the system and unable to regulate the voltage and current. Which result in the need to use the budget to get a replacement for the electronic components.

# Product Design

## Overview

As shown in Figure 2, the prototype will be designed in such a way that it is divided into 2 sections with the solar panel implemented onto the top of the prototype. For the top section, the front acrylic will consist of 1 power bank slot, Radio Frequency Identification (RFID), the ultrasonic sensor, light intensity sensor, a Touchscreen, and 2 Arduino Mega2560. The power bank slot is a platform that allow the power bank to sit on it until a rental procedure is initiated. The user will have to follow the instructions on the touchscreen and tap the card onto the RFID for the charged power bank to be ejected out of the slot. For the bottom section of the prototype, it will contain battery and electrical system (converter, voltage regulator and solar charge controller). The solar panel will be implemented onto the top of the prototype with an angle of 8 degrees. The prototype will also consist of 4 caster wheels, 2 with brake and 2 without, for convenient manoeuvring of the prototype. At the back of the prototype, both the top and bottom sections will each have a door in order for the electrical and software team to make changes easily. The acrylic sheets are white in colour as white reflects heat. Thus, preventing the prototype from overheating.

## Challenges

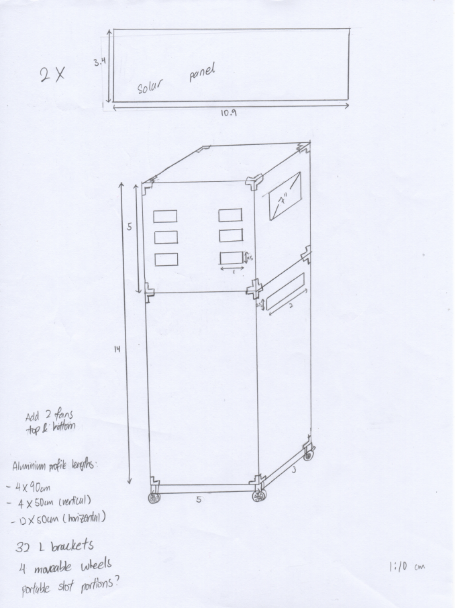


Figure 2. Prototype Sketch Version 1

We sketch out the first version of our prototype with the reference from our project specification. While designing our prototype there were a few key considerations that we constantly asked ourselves:

1. The height and width of the prototype?
2. Where should the placement of the components be located at?
3. How should the solar panel be mounted onto our system for convenience purpose?
4. Can the whole structure sustain the weight of the panel and the system inside?
5. What materials should we use for our design?

To address the issue above we did some online research on the existing power bank rental station and identify our market potential in terms of where we should place it at and how should we design to attract more attention by making it look more aesthetic.

Thus, we decided to take reference from an existing drink vending machine height and we came up with a dimension of 1400mm X 650mm X 500mm (H x L x W) and selected acrylic and aluminium profile as our base material for creating our structure.

We then further identify how we can design the product to our convenience for troubleshooting and assembly work. We adopted the SCAMPER technique while brainstorming for the idea as follows:

* Substitute
* Combine
* Adapt
* Modify
* Put to another use
* Eliminate
* Reverse

In the end, we decided to create 2 boxes and merge them together to become a pillar. The upper box is used to store components and our main components will be situated in Box A as seen in Figure 6. Whereas all the electrical components such as battery and the electrical circuit will be stored in Box B as seen in Figure 6.

During the progress, we made several alterations from version 1 and finally arriving in version 4 which was most suitable in terms of outlook and purpose as shown in figure 3 below. With the slim structure that made it look fashionable and the simple outlook that attracts passer-by regardless of age to take a sneak peek of the machine. The orientation of the compartment is taken into consideration to make it simple to understand and user-friendly too.

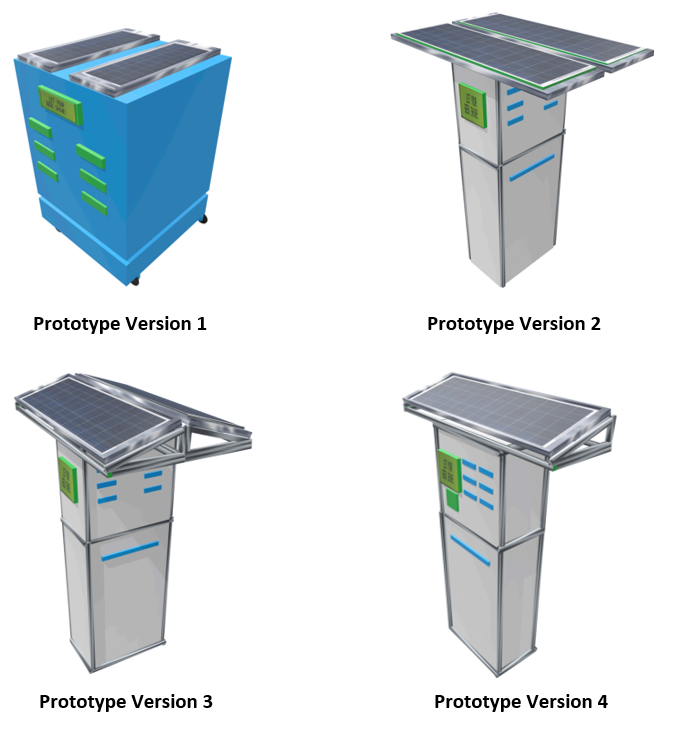


Figure 3. Prototype design development phase

In Figure 3 version 1, we implanted all the necessary parts into our prototype and we started off with a fat box. After the first design is out, we further improved it to make it look sleek and fashionable but shrinking the size of it to a dimension of 1400mm X 650mm X 500mm (H x L x W) as our prototype version 2. Then we realized that having our solar panel flat is not very efficient, which upon experimenting, we find the ideal angle to be 8 degrees. Hence, we decided to create an elevation of 8 degrees in angle in accordance to the optimal energy capture outside of Nanyang Technological University Clean Energy Research Laboratory that led us to prototype version 3. In prototype version 3, we identify some flaws such as the collection/return point of our power bank was not very feasible as the solar panel is protruding out of the prototype which causes hazard while collecting it. So, we reoriented our design and compress and compact all the external function into one side which increases the aesthetic view of it and make it look simple and clean from the front view which makes prototype design version 4 as our finalized blueprint to proceed to the assembly phase of it as seen in Figure 4.

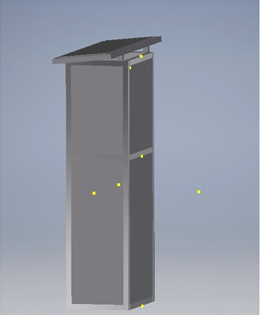
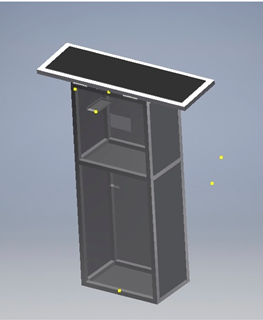
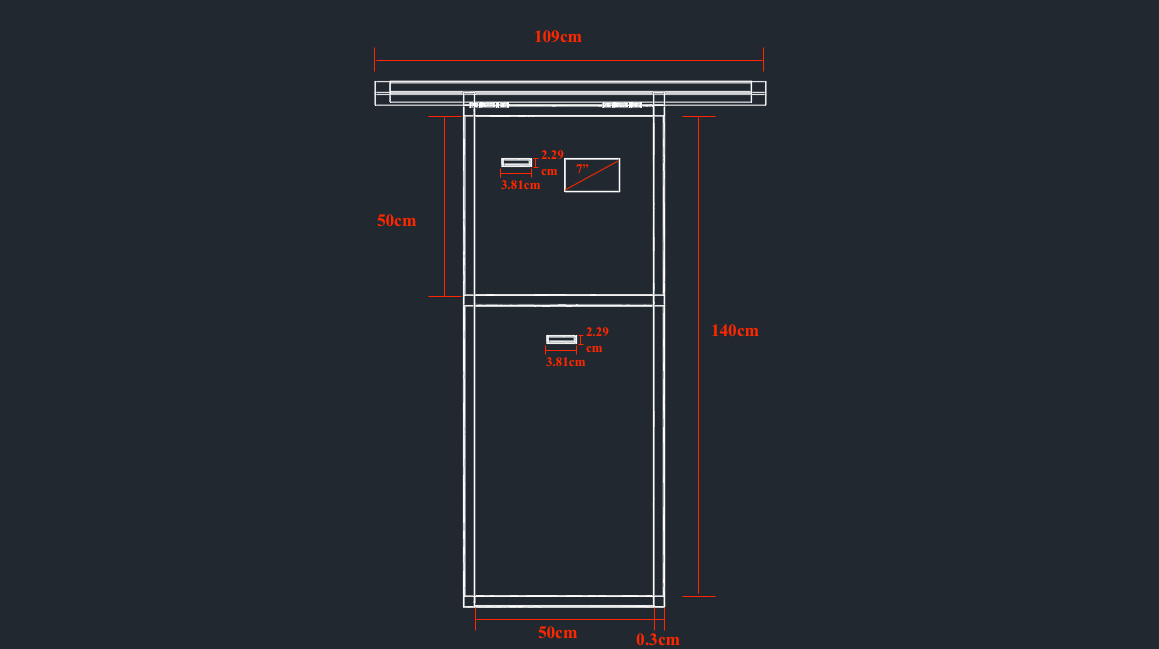


Figure 4. Prototype Blueprint

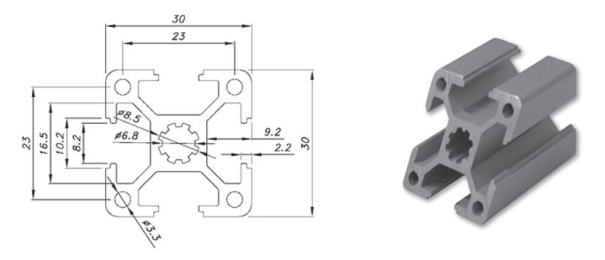


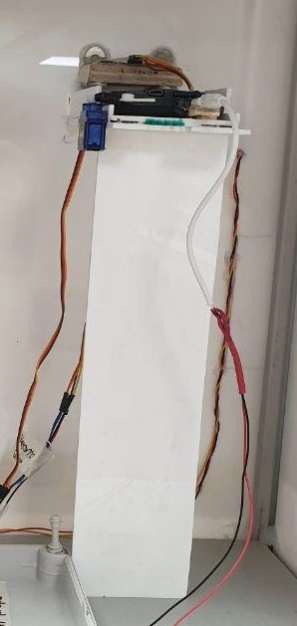
Figure 5. Prestech PTP 3030-8

During the process, we only had difficulty in the selection of the material as we need to consider how thick we should need. To resolve this, we went and consulted professional in the industry and there they proposed we used PTP 30mm x 30 mm aluminium profile (shown in figure 5) as it provides more option to modify for future expansion.

The next difficulty we faced was to decide the thickness of the acrylic sheets. As there is a need for us to cut the acrylic sheets to fit our designed prototype and the limited equipment made available for us to use, there is a dilemma in choosing the thickness of the acrylic sheets. With the equipment that is being provided, it is almost impossible to cut thick acrylic, so we choose a slightly thinner acrylic (3 mm thickness) from Dama Trading Pte Ltd. Due to the dimension of the profiles (30mm x 30mm) bought, the acrylic sheets were too thin, as a result it created excess space in between the profiles and acrylic which will lead to our prototype being very flimsy. To resolve this issue, we filled the excess spacing with silicone and small acrylics to increase its stability. In addition, we originally intended to use the acrylic to be the base of the prototype but due to the thickness it is not ideal to support the weight of the components that we will be placing on it thus we swap the material to thick plywood which can sustain the weight and for future development if there is a need to place more stuff in it.

Originally, we wanted to use the excess aluminium profile to construct an adjustable solar panel platform such that we can change the angle of the solar panel by adjusting the height of one side of the panel. By doing so, it ensures that the maximum amount of solar energy is absorbed at different time and location. Due to the technical difficulty, we decided to switch an alternative option of using wood to create a fixed joint instead.

Upon completion of our assembly of the prototype, we realized that we did not take the mounting of interior electronic mounting such as servo motor and the ultrasonic sensor into consideration. The purpose of the servo motor is used to eject the power bank out of the slot while ultrasonic sensor is used to check the availability of the slot. Which impose a huge problem for our next stage of development, to resolve this we immediately created a pillar for the support of the servo motor to be mounted on and a wooden holder that is crafted and trimmed to the size of the ultrasonic sensor. After machining the required parts, we used heavy duty glue to secure it in place as seen in Figure 6 below.



Supporting Pillar

Power Bank

Ultrasonic Sensor

Servo Motor

Figure 6. Pillar support for electronic component mounting

## Outcome

We aimed to make our prototype as user-friendly as possible and bring convenience to the user while at the same time achieving the green intiative.

In week 7, we had finished the assembly work and created a structure as shown in Figure 7 below.

****

Power Bank Adapter

DC/DC Converter (12v output)

12V 48Ah Lead Acid Battery

Solar Charge Controller

12V 7Ah Lead Acid Battery

Junction Box

(Software Compartment)

Radio Frequency Identification

Light Intensity Sensor

Power Bank Collection Point

RFID Scanner

Light Intensity Indicator

Solar Panel

3.2” TFT Touch Screen

Power Bank dispenser system

Figure 7. Finalized Prototype

To illustrate how our project intercomponent are interlinked with each other, I’ve colour coded the parts indicated as follows:

* Green 🡪 Structure Design
* Orange 🡪 Operation System Design
* Yellow 🡪 Electrical Circuit Design

# Software Design and Development

With our specification of the rental station system being defined, we can narrow down what we are supposed to use for our prototype design. Before we initiate our tasking, we brainstorm and came out with a flowchart as shown in Figure 8 to determine how our system should flow. Then from the flowchart we can filter out what are the necessary components that we need to achieve the system diagram. After reviewing the flowchart, we decided to use Arduino as our main control system as it is user-friendly and a useful tool for rapid prototyping purposes.

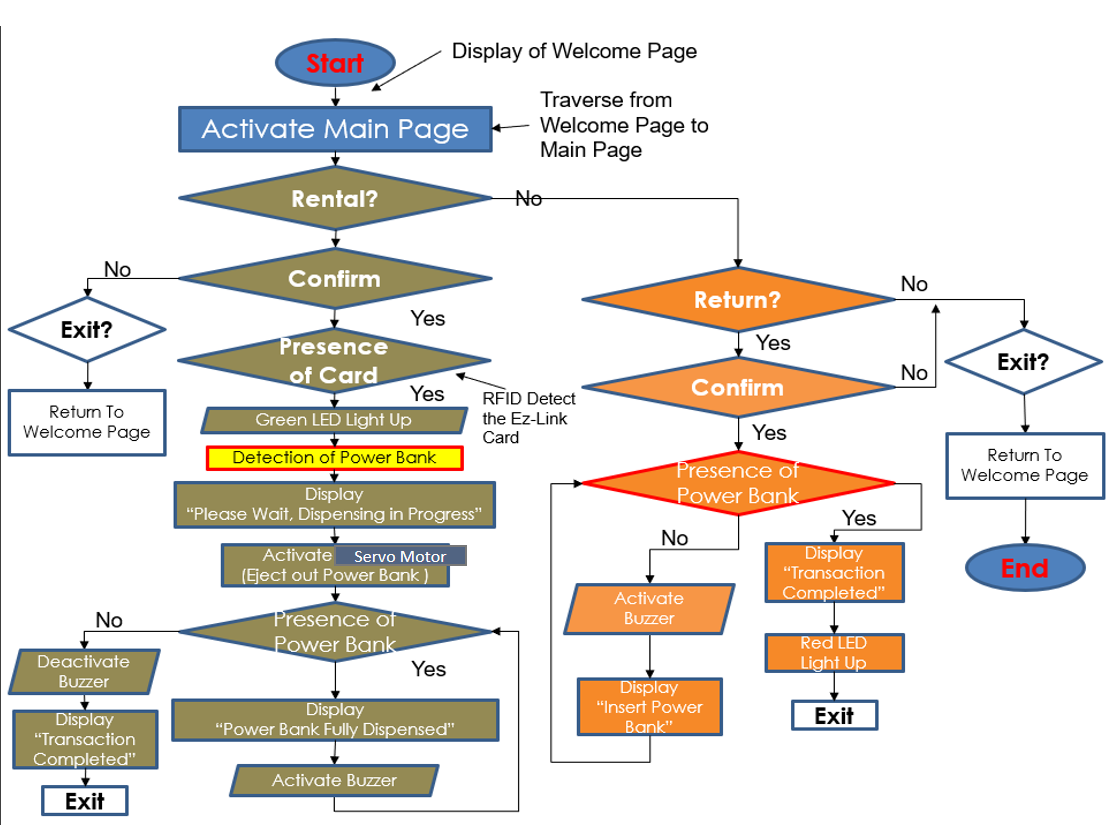


Figure 8. Flow Chart for Solar Kiosk

The components we define are as follow:

1. Ultrasonic Sensors
2. Radio Frequency Identification (RFID)
3. 3.2” Touch Screen TFT Display
4. Light Emitting Diode Strip (LED)
5. Servo Motors
6. Buzzer
7. Light Dependent Resistor (LDR)

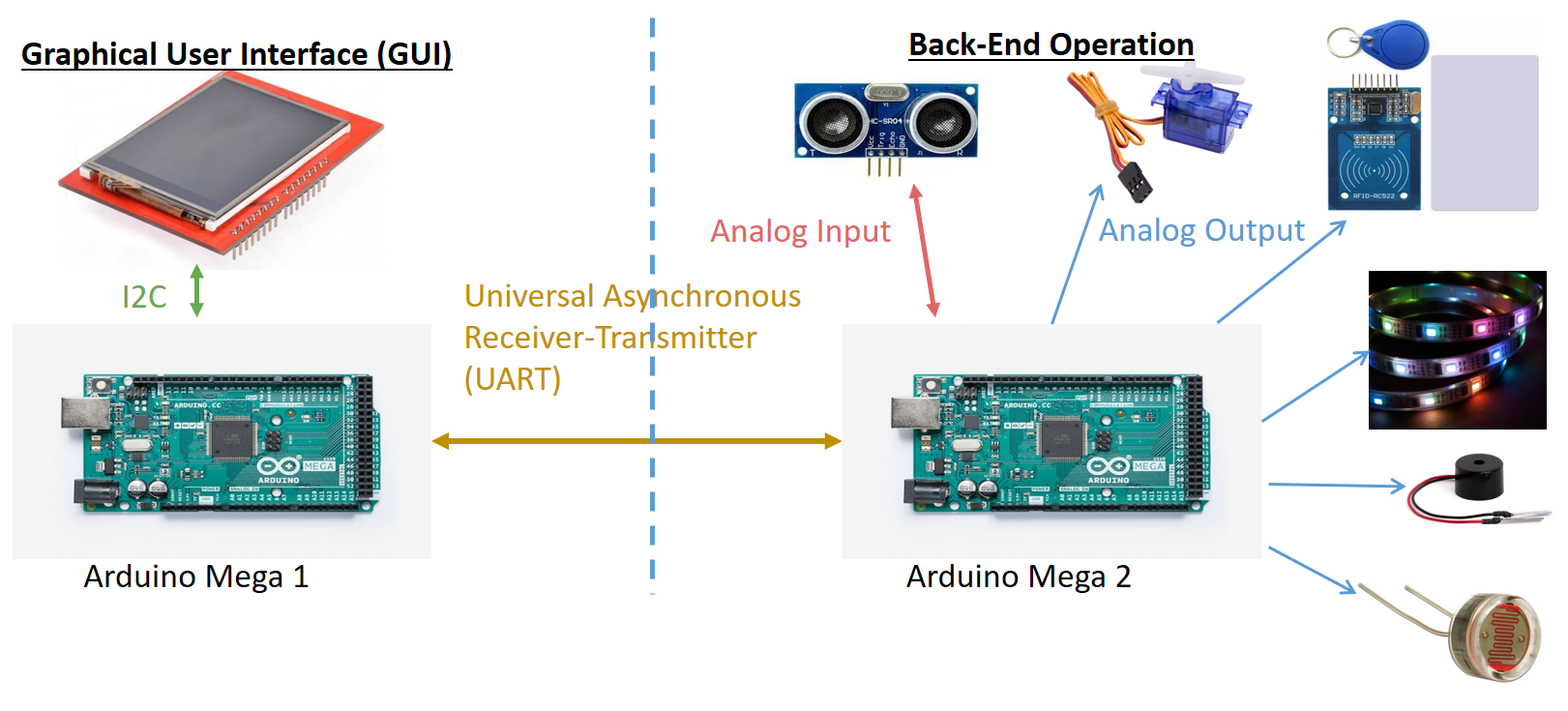


Figure 9. System Overview

In Figure 9, is the system overview of our whole prototype and a summary system diagram can be seen in Appendix B. After defining the components we need, we realized that the thin film transistor touch screen (TFT Touch Screen) occupied most of the pins that is allocated on the Arduino Mega2560 and it poses a huge problem to us as we still had other electronic components to integrate to the system thus, I broke schematic diagram into 2 categories that one of it solely do the Graphical User Interface while the other will operate the system function in this case we will require 2 Arduino Mega2560 to execute this. For the Arduino Mega2560 to execute these capabilities we decided to use Universal Asynchronous Receiver-Transmitter as our communication protocol instead of SPI or I2C. The reason why we use UART was that our system doesn’t require fast transmission speed either complex encrypting function. This narrow down to a term call **Machine to Machine system (M2M)** which we have two different machine that can communicate with each other without the presence of the human controlling it. This is achievable after we define our communication protocol specifically for our system referring to Appendix C.

## How does our component work?

To navigate you better let me start with Arduino Mega 1 where we have the user interface system in it.

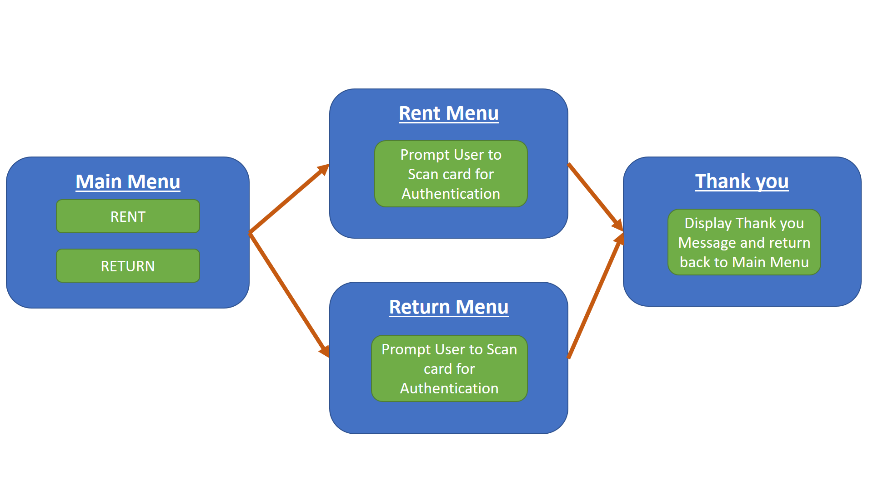


Figure 10. Touch Screen User Interface

|  |  |  |
| --- | --- | --- |
| Arduino Mega 1 Function | | |
| Image | Component Name | Function/Purpose |
|  | Arduino Mega2560 | **Main processor** to transmit and receive feedback of the touchscreen that output graphical user interface (GUI).    On top of that, serial communication is established between Arduino Mega 2 |
|  | TFT Touchscreen 3.2” | Serves as Graphical User Interface (GUI) with ‘**Rent**’ and ‘**Return**’ buttons  Subsequently, display message according to the user command.  For instance, when either button is activated commands will be sent by the Arduino Mega 1 to Arduino Mega 2 for execution of task then it will feedback to Arduino Mega 1. |
| Image result for touch screen mega shield | TFT Touchscreen Shield | The shield is to allow us to **retrofit** the existing touchscreen easily to the Arduino Mega2560 without pulling multiple wires out. |

Table 1. Arduino Mega2560 Front End

From the User interface, we designed our Main Menu, Rent Menu, Return Menu, and End Menu. Each menu will be switch accordingly upon receiving a unique code from Arduino Mega 2 where Arduino Mega 2 will only generate this unique code upon completing the different task. Table 1 shows you the key components that we used to generate our menu.

Moving on to Arduino Mega 2, we had a few components, and each serves different purposes. I’ve created a Table 2 below for reference of each individual component function.

|  |  |  |
| --- | --- | --- |
| Arduino Mega 2 Function | | |
| Image | Component Name | Function/ Purpose |
|  | Arduino Mega2560 | **Main processor** for the entire system to execute task through digital communication by sending signal to respective components.    Transmits and receives digital and analog signals from incorporated devices.  On top of that, serial communication is established to Arduino Mega 1 |
|  | Radio Frequency Identification Module | For authentication purpose to simulate the transaction process when renting or returning the power bank.   * **Detection of Serial Card** that would be scanned by the user * Transmit storage of rental information back to Arduino Mega2560 |
|  | Ultrasonic Sensor | To **detect the presence of the power bank** by measuring the distance. We have a solid platform as our reference point so every instance if there is a significant change in the proximity value it would mean there is some object being inserted into it. |
|  | Servo Motor | It is used as a mechanism to **eject** the power bank out of the platform. |
|  | LED Strip | It is used to **display the light intensity** measured in the environment to show how much power our solar panel is generating. We had 7 Led on the strip that goes from the range of 1 lighted diode being the weakest to 7 lighted diodes the strongest solar irradiance. |
| Image result for light detecting sensor | Light Detecting Resistor | It is used to **measure the light intensity** in the environment and feedback to the system. With the sensor value we created a simple equation to calculate the solar irradiance and with some calibration, we translate this value to the LED Strip. |
|  | Buzzer | To **alert** the user to collect the power bank. Variation in frequency is designed after each tasked. For completion of collecting power bank will emit long buzz follow by short buzz. For error in the process will emit long buzz. |

Table 2. Arduino Mega2560 Back End

The system proceeds in this manner for the renting option. Firstly, the user will select the rent function on the touch screen and Arduino Mega 1 will transmit a signal over to Arduino Mega 2. Arduino Mega 2 will then proceed to use the ultrasonic sensor to detect the presence of the power bank upon verifying the presence of the power bank it will then feedback to Arduino Mega 1 with a signal and proceed to the Rent Menu for authentication purpose then it will send back a signal to Arduino Mega 2. Arduino Mega 2 buzzer will then sound to notify the user to scan their card at the RFID to authenticate the procedure after it had verified that the card id is correct it will activate the servo to eject the power bank out of the slot to allow the user to collect from the slot. After the user has collected the power bank, it will send a signal back to Arduino Mega 1 to display thank you message and return to the main menu and this complete the cycle for our rent option.

For the return option, firstly, the user will select the return function on the touch screen and Arduino Mega 1 will transmit a signal over to Arduino Mega 2. Upon receiving the signal, Arduino Mega 2 will then proceed to use the ultrasonic sensor to detect if the slot is empty for the user to return else user can slot in the storage box as an alternative to return the power bank. If the slot is empty Arduino Mega 2 will then send a signal back to Arduino Mega 1 to proceed to the Return Menu for authentication purpose, then it will send a signal to Arduino Mega 2. Arduino Mega 2 will then sound the buzzer to notify the user that it is ready to receive the loan out power bank from the user by inserting back to the original slot. After the user has successfully slot back to the original slot Arduino Mega 2 will send a signal back to Arduino Mega 1 to display a thank you message and return to the main menu and this complete the cycle for the return option.

Our pin connection for Arduino Mega 1 and 2 can be found in Appendix D.

Arduino Mega 1 and Arduino Mega 2 program code can be found in Appendix E & F respectively.

## Challenges

The main challenge our group (Ernest, Cindy, and Joel) faced was the lack of experience in software development. In which we spent most of the time learning it while building our actual system concurrently to make our learning experience more productive. As we become more familiarized and understood the basics of coding, the process became much easier to handle compared to before.

During our development, we realized there is a need to use 2 separate Arduino Mega2560 as TFT LCD Touchscreen occupied of the Digital Pins on the Arduino Mega2560 board which left with limited Input/Output (I/O) Pin for us to used and taking into consideration that we still have other electronic components. Therefore, we purchased an additional Arduino and simplified our design into 2 systems. The first Arduino Mega2560 will solely control the touch screen and manage the user interface while the second Arduino Mega2560 will control the system functionality such as activation of the servo, Ultrasonic sensor, RFID, LED strip and Buzzer. Ultimately, the two Arduino Mega2560 will communicate with each other through a communication protocol which is Universal Asynchronous Receiver and Transmitter. UART is a complex communication protocol as we are required to identify our packet data in the form of byte (8bit). Thus, to make it simple to understand and program we identify a few key parameters as our command as shown in the table 3 below.

|  |  |  |
| --- | --- | --- |
| Communication Protocol | | |
| Command | Byte | Purpose |
| ‘a’ | 0110 0001 | Arduino Mega 1 🡪 Arduino Mega 2  Send out the command to execute the **renting function** |
| ‘n’ | 0110 1110 | Arduino Mega 1 🡨 Arduino Mega 2  Receive command to display on the screen that all power bank has been **loaned out** |
| ‘c’ | 0110 0011 | Arduino Mega 1 🡨 Arduino Mega 2  Receive command to display for the **collection of power bank** from the slot |
| ‘g’ | 0110 0111 | Arduino Mega 1 🡪 Arduino Mega 2  Send command to execute **collection function** in Arduino Mega 2 |
| ‘e’ | 0110 0101 | Arduino Mega 1 🡨 Arduino Mega 2  Receive command to **display ready to receive power bank** from the user on the screen |
| ‘f’ | 0110 0110 | Arduino Mega 1 🡨 Arduino Mega 2  Receive command after detection of power bank return into the system to **display thank you message** on the screen |

Table 3. Arduino Serial Communication Protocol

In addition, we also minimize the communication error by refreshing the packet data through the means of clearing it after every succession. This will avoid unnecessary overlapping of the data and in turn increase the reliability of the system. There are many ways to approach this problem, one of the it is to do data comparison where we have a set of pre-assigned binary in the main system as data get transmitted from elsewhere the microprocessor will convert the received data into readable values. This value will then be used to compare with the pre-defined binary to execute the appropriate task else it will be remove and wait for the new set of packet data to be send. However, after several considerations and troubleshoots we selected the **‘Refresh and Replace’** method as our key algorithm in the communication process.

# Electrical Design

Our system requires a constant 12V power supply to run smoothly and to achieve that we used a 45 Watts Solar Panel, 12V Charge Controller LCC02/06, Diode 1N5400, 12V 12 Ah Lead Acid Battery, 12V Direct Current to Direct Current (DC/DC) Converter, Power adapter (Ansmann USB charger) and 12V 48 Ah Lead Acid Battery.

Solar Charge Controller LCC02/06 is a voltage/current regulator to keep batteries from overcharging. It regulates the voltage and current coming from the solar panels into the battery. It also serves to prevent overcharging and may protect against overvoltage, which may reduce battery performance or lifespan, and pose a safety risk. Another function is that it can prevent completely draining a battery also known as deep discharge state with its low voltage disconnection ability. The low voltage disconnection function always keeps track of the voltage level of the lead acid battery if it drops below 11.5V it will cut off all supply to the system and divert all power coming from the solar panel into the battery. (Refer to Appendix G) This is extremely important if we are building our system towards sustainability concept as we would like to prolong the life our system as long as possible.

DC/DC converter is an electrical circuit that helps us to step down the voltage and ensure that the system does not exceed the maximum rating of the current of 1.25A as stated in the specification sheet. (Refer to Appendix H) Another function of the DC/DC converter is that it serves as a primary circuit protection so in an event of an overload or over surge of current in the system occurs it will cut off all supply immediately by exploding its internal fuse. To select our DC/DC converter firstly we need to identify how much voltage is needed for our system in our case we have Arduino have the highest input voltage thus we required at least a constant 12V power supply and next we must identify what is the maximum current your system will draw from your power supply to measure this you can either use a programmable power supply to test or identifying the impedance across your system to determine the maximum current your resistance should be the minimum value to achieve the highest possible current. After you managed to determine all this parameter you will be able to select the most suitable converter to use.

The diode is installed before the system prevents reverse current flow and it is a must to have it in the circuit design whenever we have an electrical component that does not tolerate reverse current flow. It helps to protect and maintain our solar panel to ensure that it will not be damaged by the reverse current flow. To select your diode, you must check the forward biased current that it can work until in our case I recycled a diode used previously by another project group it requires a forward bias voltage of 1V and can have a forward bias current up to 3A. (refer to Appendix K)

Ansmann USB charger was used to step down the voltage to the power bank as without it the power bank can absorb up to 1A for fast charging capabilities but that will defeat the purpose of sustaining our entire system, so we step down the voltage to 5V constant output for charging state.

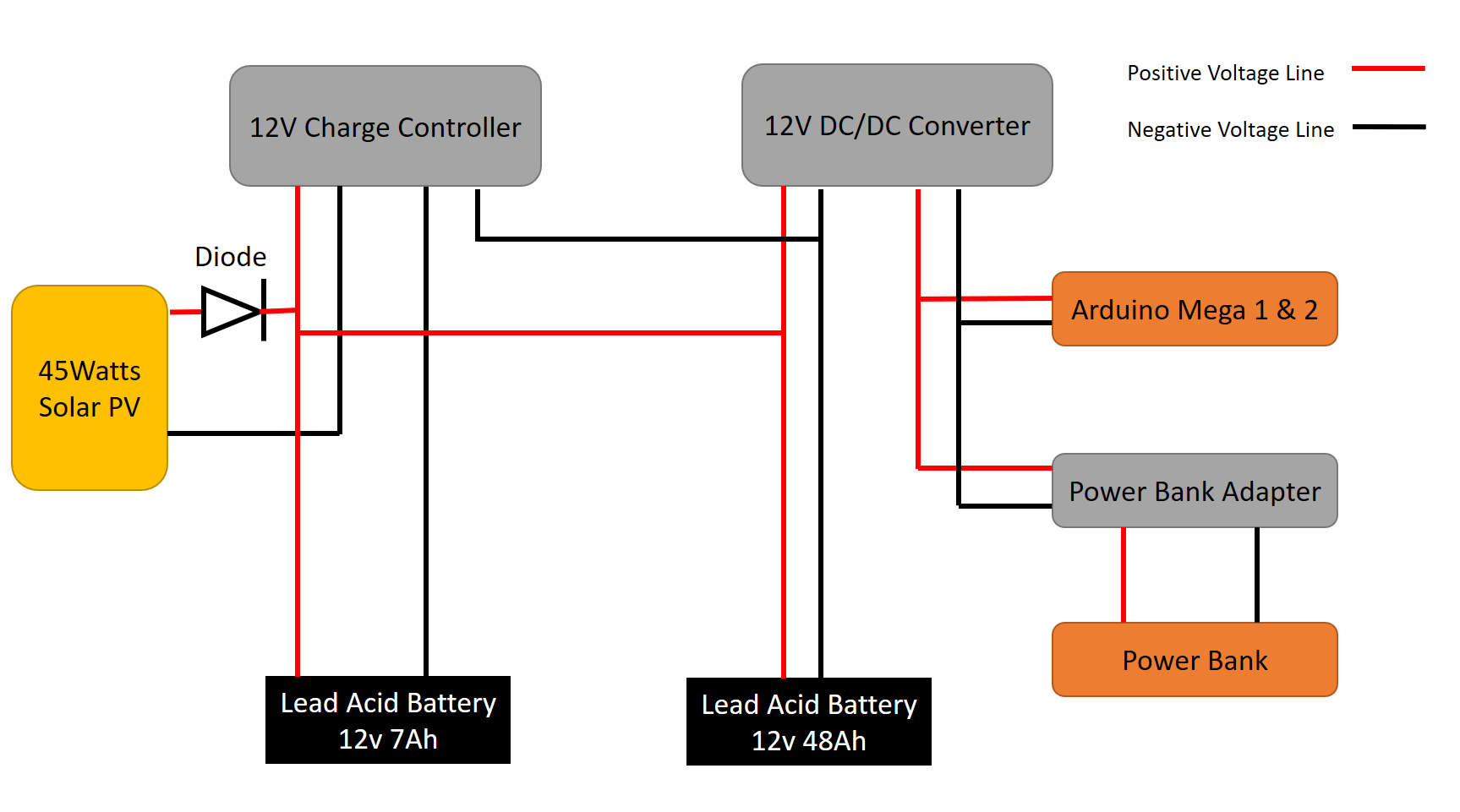


Figure 11. Circuit Diagram

Figure 11 shows our circuit connection for our system. Our circuit is designed with a backup power taken into consideration. The whole system is able to function with just one lead acid battery connecting to the charge controller. The secondary lead-acid battery comes into picture when the first-lead acid battery is undergoing its charging state. Practically, we wanted to create a programmable relay circuit that is able to charge both lead-acid battery systematically when either one of it is in idle mode but due to time constraint, we decided to do active discharging of the secondary lead-acid battery. The motivation behind this is to ensure that our system is independent, reliable and sustainable by its own for a long period of time. The table below shows the amount of current, voltage and power consumption by the system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| System Operation Power Consumption | | | | |
| Mode of operation | Voltage | Current | Power | Life Cycle (W/55 Ah Battery) |
| Switch On | 12 Volts | 400 mA | 4.8W | - |
| Normal Operation (W/active charging power bank) | 12 Volts | 0.517A | 6.2W | 106.38 hours |
| Rent Function Operation | 12 Volts | 0.96A | 11.52W | 57.29 hours |
| Return Function Operation | 12 Volts | 0.54A | 6.48W | 101.85hours |

Table 4. System Operating Power Consumption

We can observe from the Table 4 under the power consumption column under Normal Operation the whole system consume up to 6.2 Watts of power, so in theoretically speaking we should be able to power the whole system independently with just one solar panel since our panel is able to produce 45 Watts at ideal condition (Solar Irradiance : 1000 W/m^2 and at temperature of 25ºC). However, practically it is not possible as there was numerous factor such as inconsistent weather condition (Sunny, partially cloudy, cloudy, rainy), wear and tear (Overall efficiency of the Solar Panel will drop over time), Day and Night-time (No power can be generated by the panel during night) and maintenance of the panel to ensure it operates at optimal efficiency. Thus, there is a need to integrate the system with a suitable size of battery for the system to run independently by itself taking in account of the factors that will result in system failure.

The Life Cycle seen in Table 4 means the amount of time the system can last with 48Ah and 7 Ah supplying power to the system. To calculate that value simply using . However, in practise, we will not let the whole system fully deplete the lead acid battery as it will result in deep discharge characteristic (50 – 80% of its rated capacity) for it to maintain at its optimal condition.

In Summary, our system draws a total current of 0.517Ampere that comes from 3 key components (0.327 Ampere from 2 and 0.19 Ampere from power bank during its charging state). Thus, the overall power consumption of our system in normal operation state is 6.2 Watts and it is capable of self-sustaining up to 4 days in the ideal weather and environment condition\*.

\*Ideal condition – Solar irradiance of 1000W/m­­2, temperature of 25oC and Air Mass(AM) Coefficient of 1.5

## Analysis and Finding

We wanted to study how the solar panel operates with the change in solar irradiance as we were curious that why under room lighting it is able to generate an Open Circuit Voltage (Voc) of 12V so we took the data analysis at the open field outside of Nanyang Technological University of Singapore Clean Energy Research Laboratory on 24 October 2018. Through this result, we were able to use the Solar Panel Specification Sheet and determine our supposed output power by using the I-V characteristic curve of the solar panel as seen in Appendix N.

## Challenges

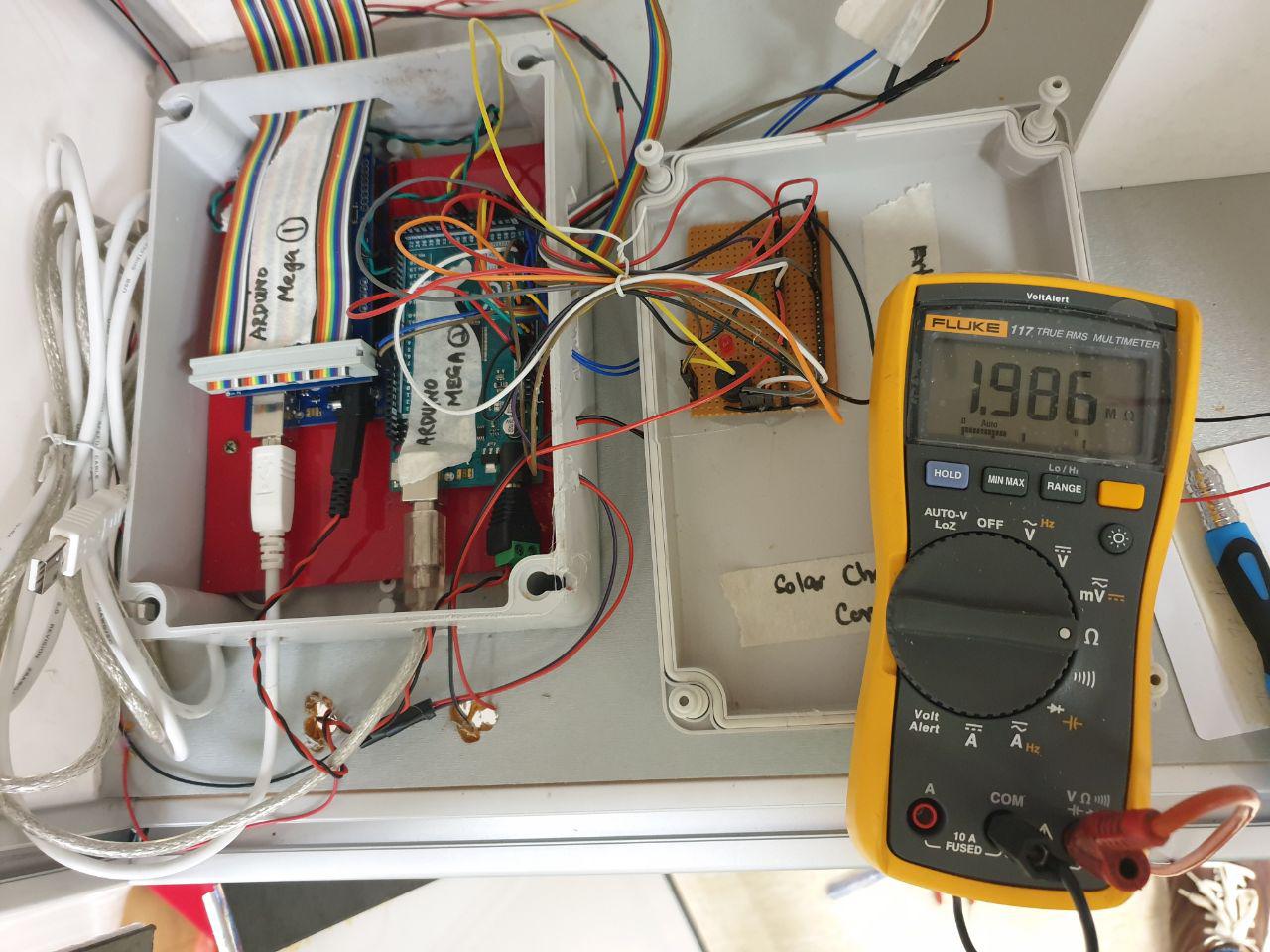
The main challenge we faced is the electrical system design by taking into the consideration of all the electrical components how much current is being transferred to the whole system. To achieve a rough estimation after connecting all the components, we measured the impedance across the 2 Arduino Mega2560 in parallel as that is our intended connection to the power supply and we measured a value of 1.985MΩ as shown in figure 12. However, we do need to take note that the impedance that is measured across is the Thevenin resistance in the whole circuit. This impedance value that we obtained was extremely high and it doesn’t seem right as if we supply 12V into the system theoretically speaking if we use ohms law we will get which was extremely low for any of the electrical component to operate.

Figure 12. Impedance across 2 Arduino Mega2560

We then did some research by studying the Arduino Mega2560 circuit diagram. We had difficulty to trace the circuit from the power supply as the schematic diagram does not reveal much of the internal circuit connection as seen in the Appendix I. However, we can rule out 2 possibilities that result in the high impedance as seen in Figure 13, firstly we suspect that in the ATMEGA8U2-MU internally have multiple switch circuit which result in open circuit resistance that gave us a very high impedance value, but it would never be infinity. Secondly, we suspect that the Operational Amplifier (LM358D) as circled in Figure 13 that contribute to the high impedance value as from what we studied in EE2002 Analog Electronics we understood that for ideal Operational Amplifier it has a very large input impedance which will result in us obtaining the high resistance value. [2]

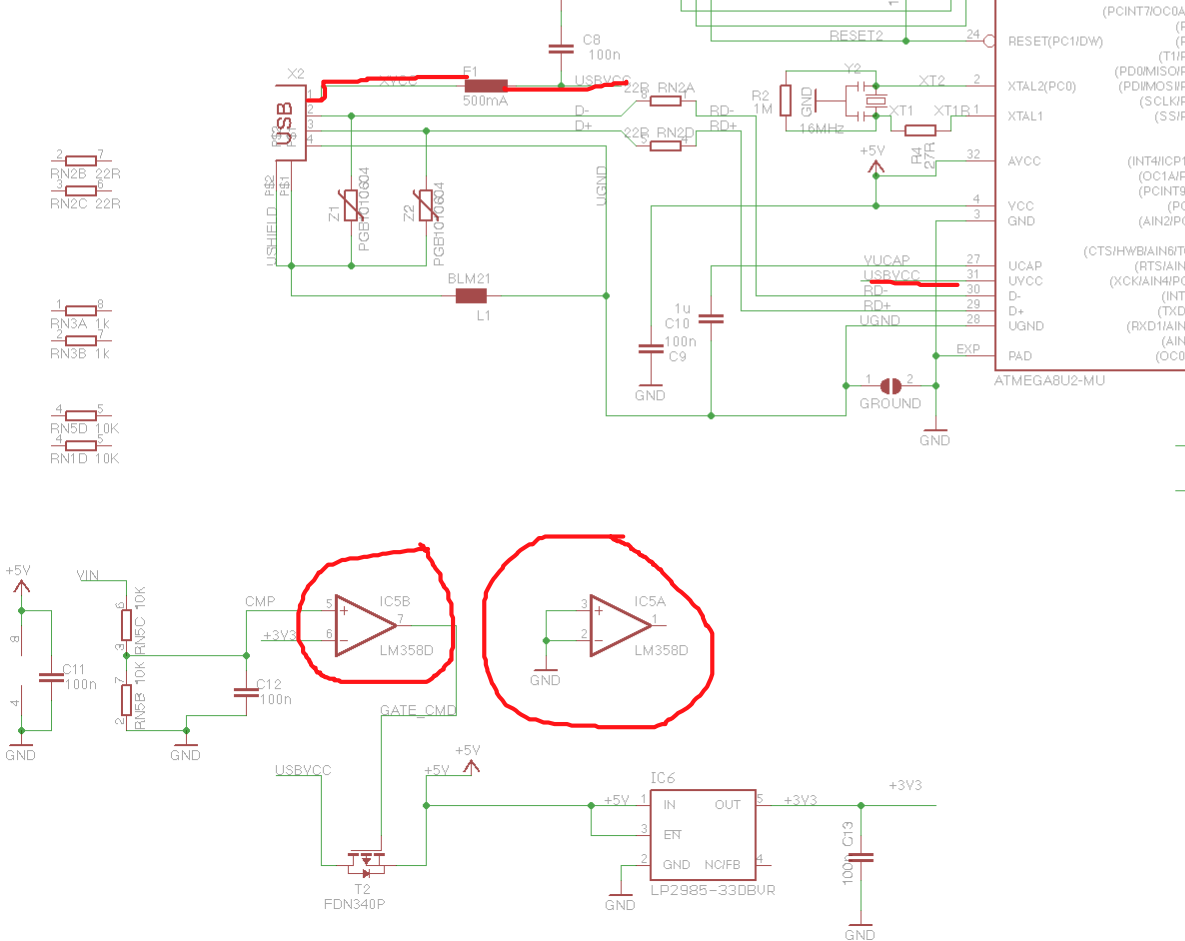


Figure 13. Arduino Mega2560 Schematic Diagram

Then, we measured the current by connecting the whole system set up directly to the lead-acid battery as shown in Figure 14 below. This way we can calculate our impedance across the device by using ohms law. However, we then realized this approach was very dangerous as there was no regulator in between the source and the load. In an event if the load impedance is extremely low it will draw very high current which will result in electronic failures.

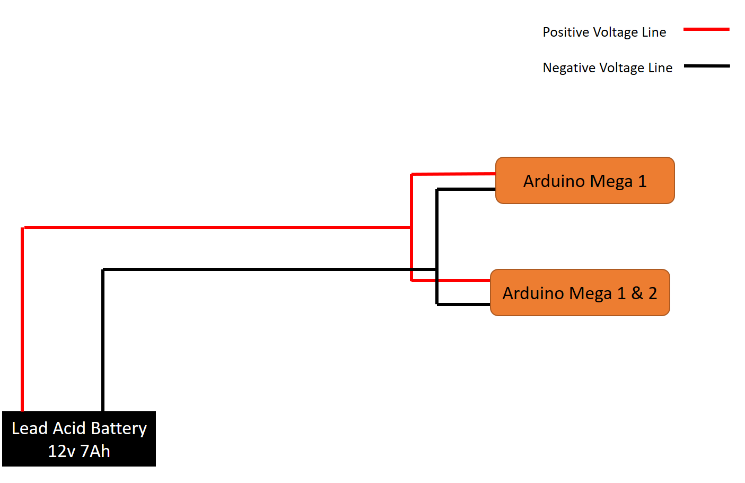


Figure 14. Circuit Diagram for Lead Acid Direct Power supply

We achieved a value of 0.577Amps and a drop-in voltage of 9.77Volts as shown in Figure 15. This was not a very good indication as more current is drawn from the lead-acid battery even though the system is in its idle mode. This is a critical mistake that we made which burned the whole Arduino Mega2560 board even though internally it has a current limiter and not after we observed that the current increases in a step size of 0.2Amp after every 15 seconds. To resolve this issue, we made some improvement by connecting it to a DC/DC converter to output constant voltage and ensure the current does not exceed the maximum limit as seen in Figure 11.

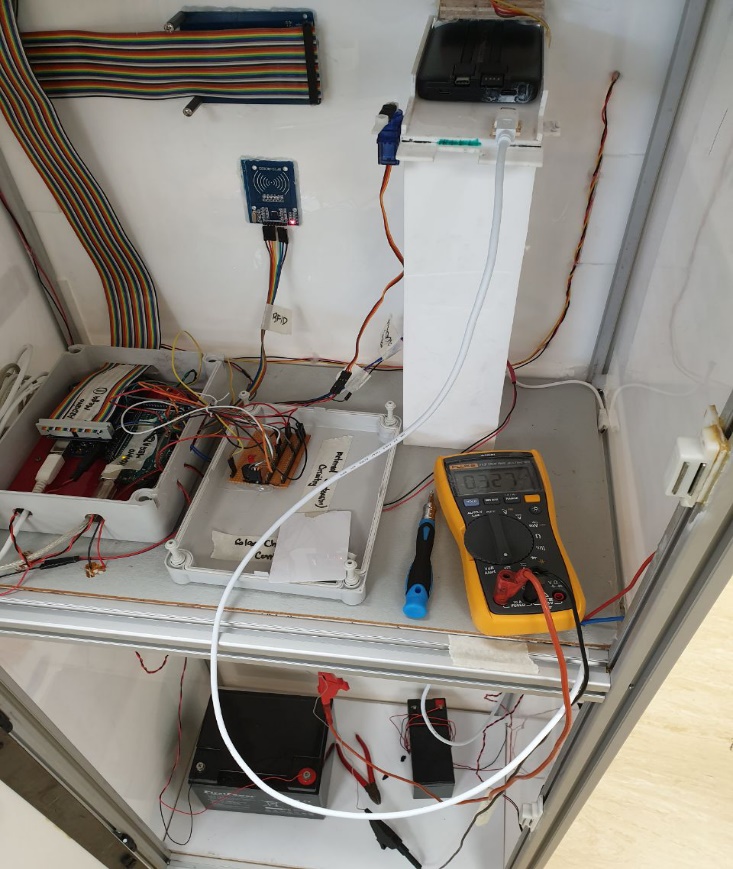


Figure 15. Lead Acid Battery Supply only

Learning from our mistake previously made, we redesigned the whole circuit by incorporating with a 12V constant power supply from the DC/DC converter which also limit the current to 1.25Amps as shown in Figure 11 and Appendix H. We measured the current supplied into the system is found to be 0.327A without supplying power to power bank as shown in Figure 15 which mean that our connection resistance across is . Take note that this current value is the normal operating condition for our whole system under its operation mode where we activate the ‘Rent’ function it will switch on the servo and it will draw a peak current of 0.96A for a short period of time before it reverts to the normal operating current. Our overall system total resistance taking account of the power bank is found to be .

Figure 16. Normal Operation Measured Current without power bank connection

We then tried to identify what the issue in the rise in current through connection we then realized the main fault was coming from the servo motor. The power supply of our servo motor was tapped directly from I/O pins of the Arduino Mega2560 and in the specification sheet for Arduino Mega2560 it clearly states that the maximum rated current output from the pin is 40mA. As we activate the servo motor it draws very high current and as a result it damages the voltage regulator in the Arduino Mega2560.

Learned from our mistake previously, we then integrate the system with a DC/DC converter to maintain the current output. When we select our DC/DC converter we also take in consideration of maximum operating current. We realized that the peak current the whole system can operate is 0.96A so we picked a current rating of the DC/DC converter higher than its peak current. Thus we selected, SD-15A-12 as our medium for all the power supply since it can supply a constant of 12V and a current rating of 1.25A also it can overload up to 160% of the rated power which mean for our case is 24W (. Which is sufficient if our system fails and result in over surge of current the DC/DC converter will come in and break the whole system for safety purpose.

Subsequently, the lead-acid battery handled to us was from the previous final year project inventory and as a result, it was not well maintained after we took over it which impose a huge problem. Why was this an issue? Since all batteries, will undergo self-discharge and the rate of self-discharge is depend on the storage or operating temperature. After we took over the inventory we were not aware of how long the lead-acid battery was being stored as a result it will have severe sulfation, which causes loss of capacity and loss in its efficiency. Theoretically speaking, if the lead-acid battery is left not maintained for a long time we can still charge it till its optimal voltage value but the discharge rate will be extremely high so there is a possibility that the battery can only last for a short while due it high discharge rate if being used and strictly speaking it is not very safe to use it.

There was a severe mistake we did while we were handling with lead acid battery which is overcharging it and it is a common mistake made in general. Initially we were quite concerned on how we should charge our lead-acid battery as we thought there is a need for an adapter to charge our lead-acid battery but after we asked our seniors and experience lab technician at Nanyang Technological University School of Electrical and Electronic Engineering Robotics Design Lab they clarified with us that as long we monitor the lead-acid battery voltage level we can charge it directly with the power supply by adhering to the datasheet specified charging voltage in our case is between 13.5-13.8V charging voltage as seen in Figure 18.

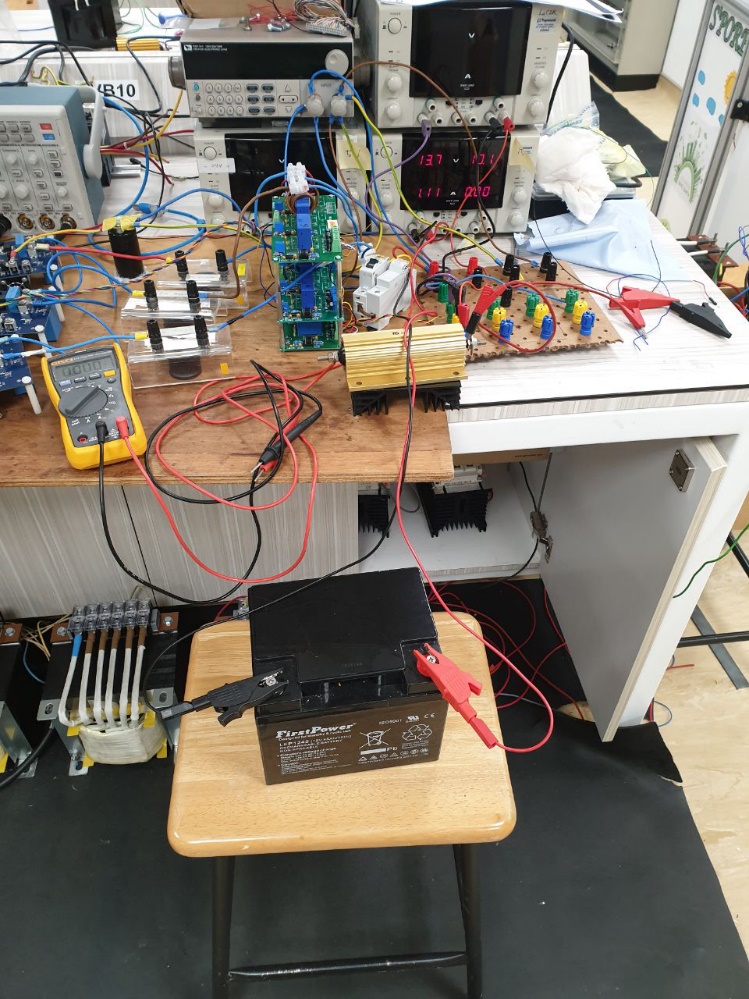


Figure 17. Lead Acid Charging Method Figure 18. 12V 48Ah Lead Acid Battery Voltage

However, we have to take note that by using direct charging method we have to closely monitor the voltage level as lead-acid battery is able to consistently take in power so there would not be an event that we can charge to a point where the power supply will know that the battery is at 100% capacity and supply no current out and lead to overcharging. Typically, a 12V lead-acid battery should output at least 12V from its terminal if we measure with a digital multimeter as seen in Figure 17. We can see an output of 13.04V this is due to overcharging of the lead acid battery however with that said it will not cause immediate failure of the battery but over the long run it will wear down the battery quickly. Of course, there is a peak voltage the lead-acid battery can reached even though it is stated as 12V on its plating, but it can reach up to 13.8Vmax, however it is not recommended to do as a good way to maintain the battery is between 50-80% of its capacity cycle. Continuous charging will cause accelerated corrosion of the positive plates, excessive water consumption and damage the battery. It is recommended that the Lead acid batteries should be charged after each discharge of more than 50% of its rated capacity and during or after prolonged storage of 30 days or more. Generally, this was not much of a concerned for us as we connected our lead-acid battery to the solar charge controller that helps us to manage the power going into the battery bank from the solar array also to prevent overcharging of our solar panel.

Lastly, we did a test run to run the whole system with just solar panel only to see if it is capable to execute the system function with just solar panels in parallel since we have 4 solar panels of 45Watts in our inventory. Theoretically, our system consumes up to 11.52Watt peak under operation from table 4 so if we have one single panel that is operating at peak condition output of 45Watts we should be able to draw enough power to supply to the system. However, due to the inconsistent weather condition, we were only able to draw up to 0.36 ampere which is just sufficient enough for the whole system to stay turn on. To resolve this issue, we integrate it with a battery to allow it to operate as a stand-alone system. Our circuit connection for the stand alone system without the battery can be seen in Figure 19.

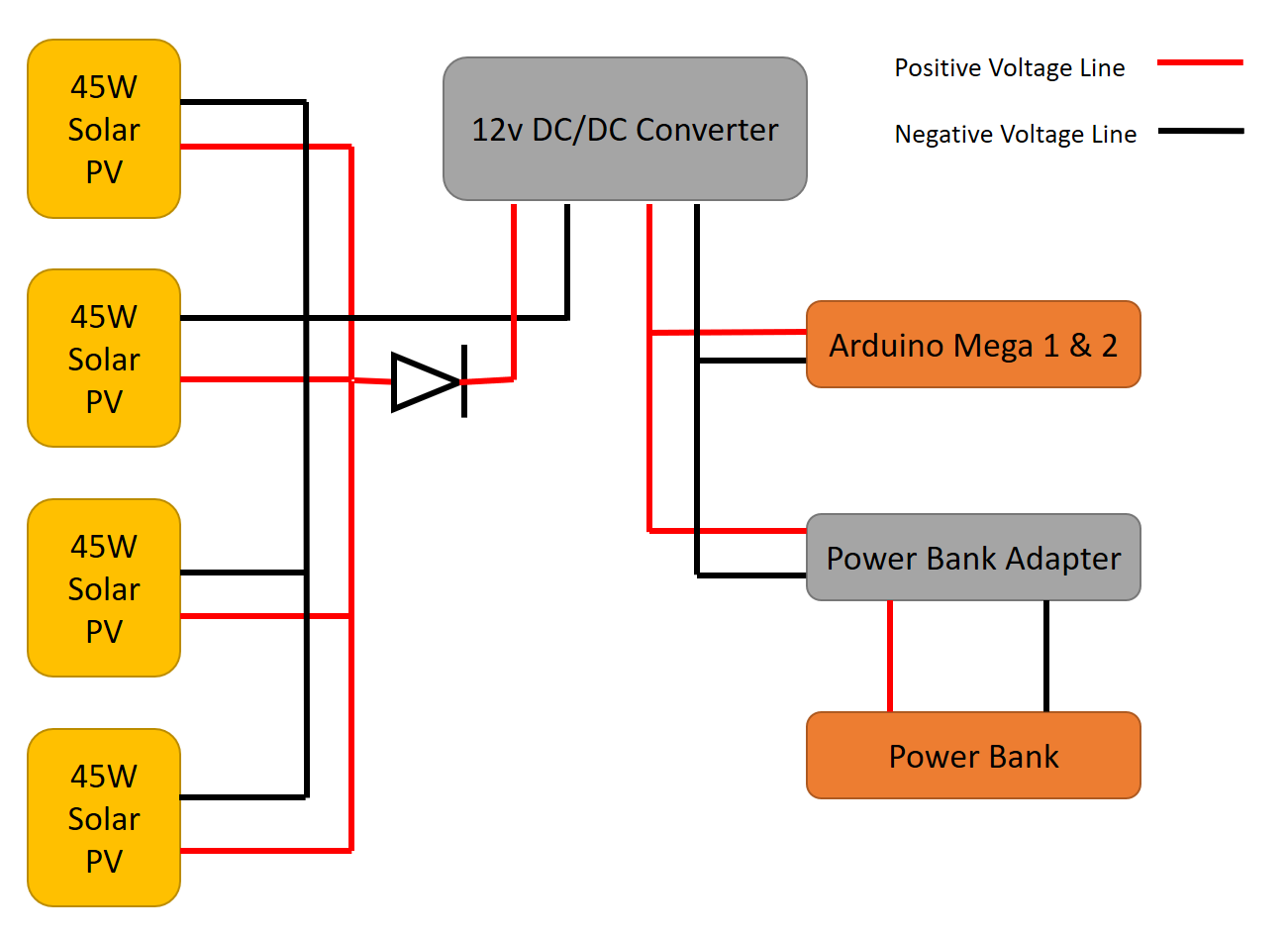


Figure 19. 4 Solar Panel to system circuit diagram

# Cost Analysis

|  |  |  |
| --- | --- | --- |
| **PHASE** | **Planned Costs (S$)** | **Actual Costs (S$)** |
| **Initiating Phase** | **-** | **-** |
| **Planning Phase** | **600.00** | **529.00** |
| **Execution Phase** | **400.00** | **135.24** |
| **Closing Phase** | **200.00** | **171.51** |
| **Project Total Costs** | **1,200.00** | **835.75** |

Table 5. Cost Analysis Breakdown

With the project budget fund of S$1600, we allocated our budget accordingly as seen in Table 5. The planned cost is what we allocated for our budget initially after sourcing and purchasing all our necessary materials and equipment’s we arrived at our actual cost. The actual cost can sometimes be lower than the planned cost. However, for our project, we planned our financial appropriately to avoid unnecessary expenditure.

During the Execution Phase, we undergo several changes from the original blueprint of the prototype and as a result, we required to buy additional materials to complete our system. One of the examples was mentioned under the Product Design section where we did not take in account of the prototype door frame and the solar panel mounting bracket and as a result, we brought a long metal piece and cut out the required part to reinforce our prototype.

Near the closing phase, we accidentally burned an Arduino Mega2560 board because we did not regulate the voltage into the Arduino as a result, there is an over surge in current which in turn burn the board. Thus, we had to immediately get a replacement for it and the amount of money that we spent is reflected in Table 5.

# Lesson Learned

Product Design Team

1. Actual vs. Theoretical dimensions:  
   Theoretically, we planned to have a top compartment sized 500x500mm, however, due to the tapping (Creating of threading path) of M8 to the aluminium profiles, the actual measurements were slightly smaller. Hence there is a need to trim the acrylic of (500mm x 500mm) to fit nicely into the aluminium profile slot. Although there is a deviation in the actual prototype size it does not affect the entire prototype functionality and the outcome of the prototype was scale down slightly to 1400x560x360mm.  
   We have learned there will always be deviation as we progress towards the end product however as long as it did not affect the original purpose of the system functionality we can exercise our judgement to make necessary alteration.
2. Software limitations:  
   The 3D builder software by Microsoft that was used is unable to create an internal layout thus there is a need to switch to Autodesk Inventor which suit our purpose better. In Autodesk inventor provide a function to design the exterior as well as interior and it is more suited for designing prototype in parts then assemble them together and display in a blueprint.  
   An online platform such as Youtube offered an effective tutorial in aiding the design and drawing of the prototype.
3. Cutting of acrylics:  
   Machining of parts often involves estimation and adjustment as our prototype is done by human instead of the machine so there will bound to be some error. Thus, we always ensure that whenever we do cutting we offset about 1-2mm from the actual dimension then trim it accordingly.
4. Solar panel bracket mounting:  
   We encounter the problem where we did not have enough materials to create the solar panel mounting bracket so under such circumstances we had to improvise to create the necessary joints to fix the panel on top of our prototype. During project development, there were bound to have times where we encounter lack of resource and time constraint, so we had to always keep our mind open and think of a creative way to replace it as an alternative option without affecting the functionality of the product.

# Future Work

We planned to improve the overall system firstly by starting to build a battery management system and expand the number of power bank it can store within one machine. The intention was to have at least a minimum of 20 power bank within one system so that it can be commercialized in the market.

Secondly, to improve the mechanism inside our system currently the issue we have right now is the servo we have is not very suitable for the current system setup due to the amount of current it draws when in operation although it is for a short period of time there might be a risk of servo failure and result in excessive current drawn by it and even breakdown in the process. Thus, what I recommend is to change the charging mechanism of the power bank to wireless charging mode and create an elevation dispensing mechanism that whenever there is a need to dispense the power bank with a slight angle of elevation and with the help of smooth surface it can slide out to the collection point.

In addition, another benefit is wireless charging is extremely popular in the market. Ideally, the next expansion of our power bank should have wireless charging setup within it and it must be capable of toggling between taking in power or supplying power to complement our system and the end user as well.

Thirdly, to build and develop a network system for each of the machine to communicate with each other. In other words, each machine is like a hub that is able to send information back to our main database, so we are able to keep track of how many power banks is left in that machine and also, we can monitor the system efficiency and profitability by monitoring the transaction as well as the health of the system.

Fourthly, since from the previous point we have set up each machine to be a hub station we should integrate all the power bank with a tracking device. To do this the most cost-effective way is to make use of each machine location and from a certain distance apart from the power bank built in Bluetooth module can link up with the hub whenever it is near it then it will transmit the location to a database that has been set up. It does not necessarily have to be real-time as usually the transaction of the power bank should last within a 24 hours’ time span. The motivation behind this is to keep track of all our power bank as it is extremely costly to get a replacement for it furthermore I will recommend to come up with a business model such that after loaning the power bank they will have a loan period of 24 hours after that the user will have to pay the full price of the power bank and they will get to have 100% ownership of the power bank.

Lastly, to enhance our user experience, we also plan our financial model with the reward system in it by collaborating with shopping outlet, food, and beverage outlet and other shops. With the points earned through the usage of our power bank, the customer can exchange this point for vouchers or discounted meal price. Our profits come from advertisement and survey platform from other company through monthly subscription options. Our motivation behind this is a company such as Grab, Blue Sg and many more have attracted a lot of customers not only because of the convenience of their product but also the rewards they will reap out of it.

Ultimately, the system should be able to effectively execute task smoothly and be able to retrofit existing cash transaction system and even wireless network for authentication purpose. The machine should be as cost-effective as possible to be able to set up many stations around the region. Ideally, we planned to set up around the campus/schools and public transport areas such as Mass Rapid Transit Station, Bus interchange and even shopping malls as it brings convenience and also work towards the green initiative in the world.

# References

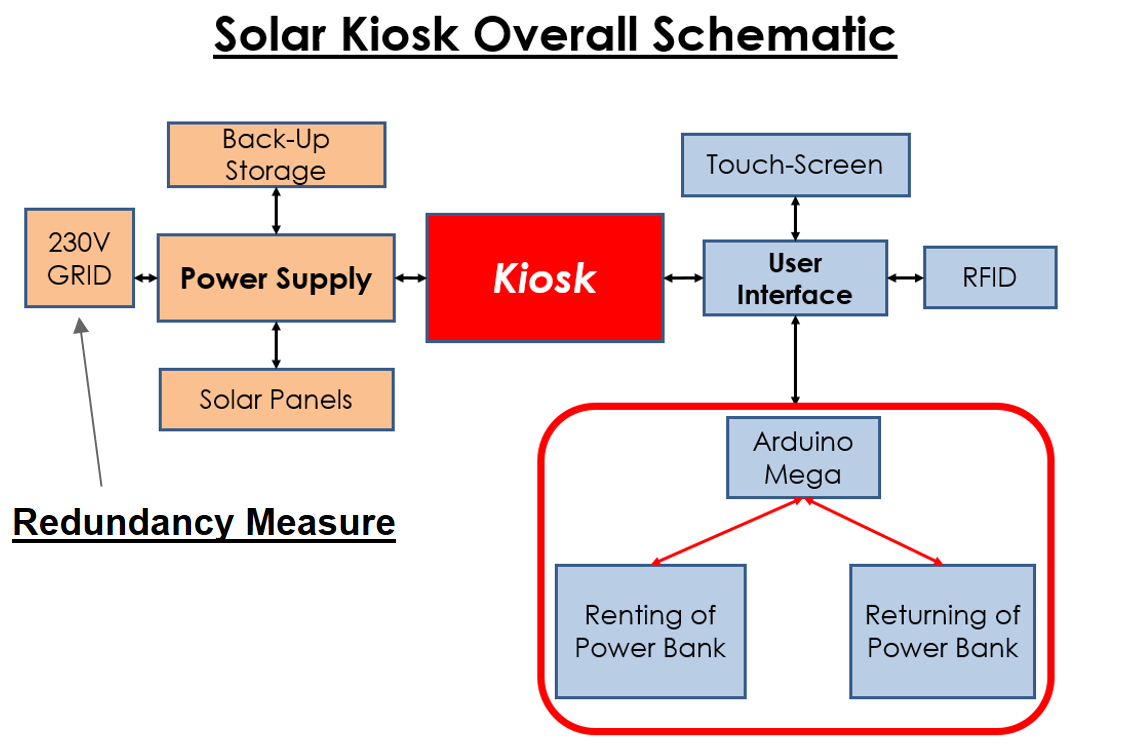
[1] Z.W. Lim, “Prototyping of a solar-powered charging station for energy storage applications,” Nanyang Technological University, 2018

[2] ARTICLES, T., PRODUCTS, N., ELECTRONICS, G., PROJECTS, C., MICRO, E., Lectures, V., Webinars, I., Training, I., Search, P., DB, T., Tool, B. and Designs, R. (2018). *The “Operational” Amplifier | Operational Amplifiers | Electronics Textbook*. [online] Allaboutcircuits.com. Available at: https://www.allaboutcircuits.com/textbook/semiconductors/chpt-8/the-operational-amplifier/ [Accessed 13 Nov. 2018].

# Appendix A

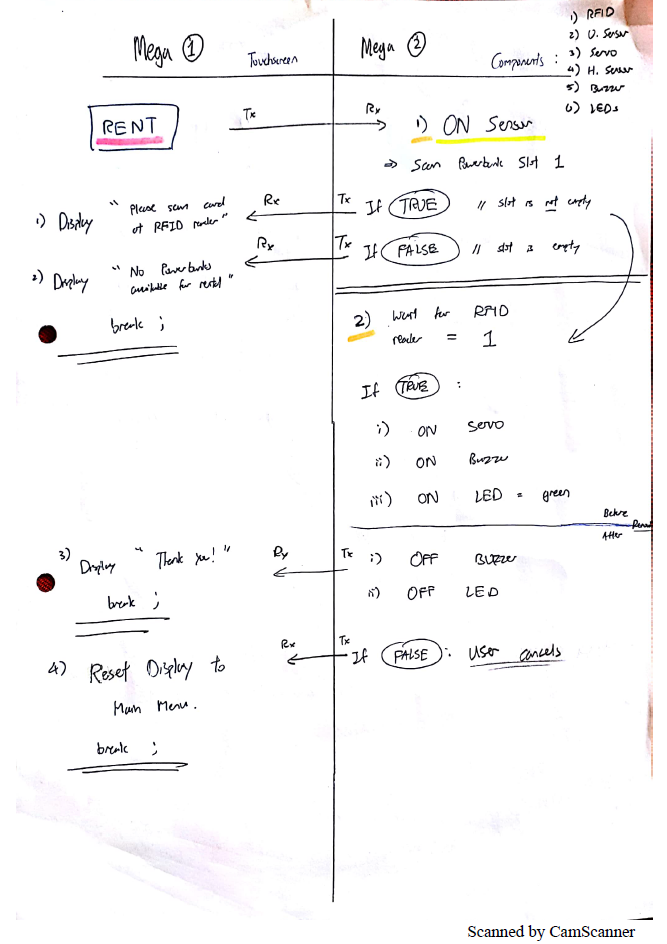
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name** | **Project contributions** | **Report Contribution** |
| 1 | NA SHI CHEN | Team IC, Final Project Integration (Electrical/Software) | 1-6,11,19-61 |
| 2 | LAI WEN PANG | Team 2IC, Mechanical | 7-10 |
| 3 | FOO CEJUN JOEL | Treasurer, Software Developer | 12-17, 29-50 |
| 4 | ERNEST LOW QI EN | Software Developer | 12-17, 29-50 |
| 5 | CHONG MEI LING, CINDY | Software Developer | 12-17, 29-50 |
| 6 | LIM KIM YEW | Mechanical | 7-10 |
| 7 | WANG ZI | Mechanical | 7-10 |
| 8 | LIM LI TING RACHEL | Product Design | 7-10, 24 |
| 9 | STACY NG YAN RONG | Product Design | 7-10 |
| 10 | HERNANDEZ NEVIN MIGUEL LIVELO | Electrical | 17-19 |

# Appendix B

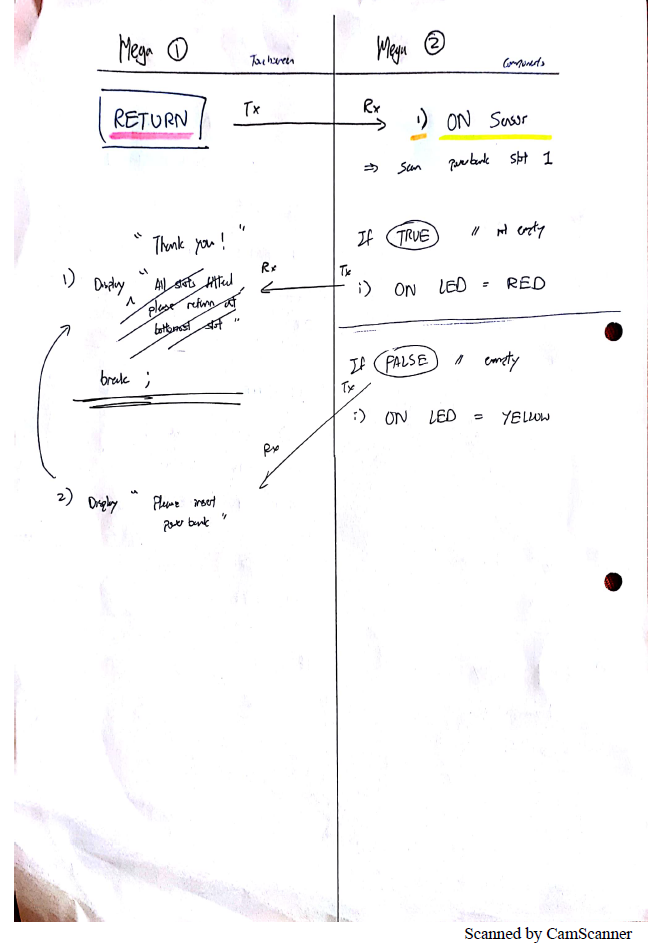


# Appendix C

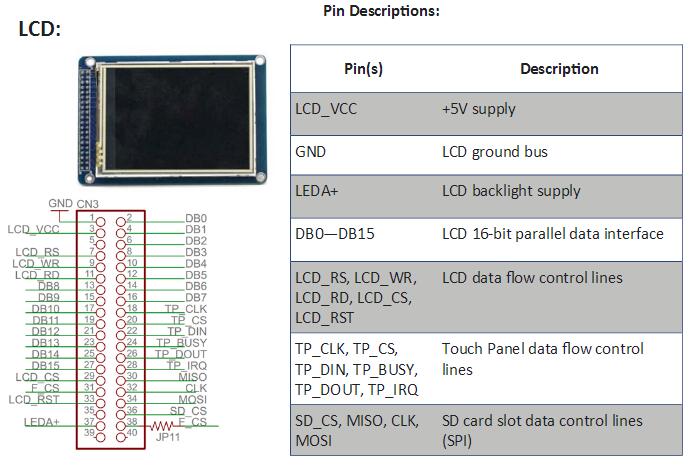
Rent Function Guide Chart

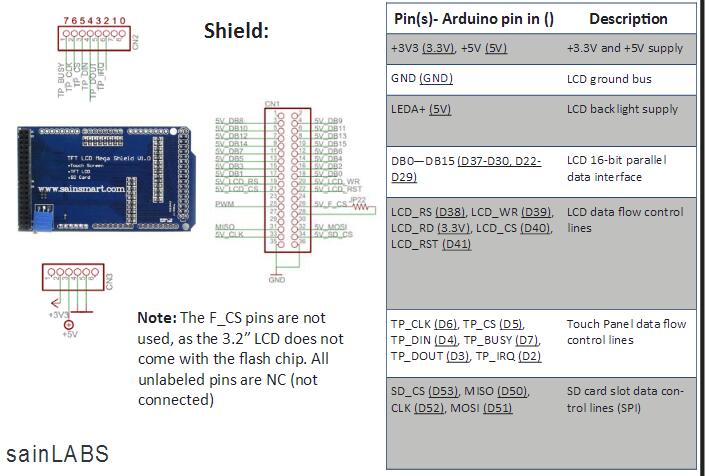


Return Function Guide Chart



# Appendix D





|  |  |
| --- | --- |
| **LCD TFT Touchscreen** | |
| **Name** | **Descriptions** |
| LCD\_VCC | +5V supply |
| GND | LCD ground bus |
| LEDA+ | LCD backlight supply |
| DB0-DB15 | LCD 16-bit parallel data interface |
| LCD\_RS | LCD data flow control lines |
| LCD\_WR |
| LCD\_RD |
| LCD\_CS |
| LCD\_RST |
| TP\_CLK | Touch Panel data flow control |
| TP\_CS |
| TP\_DIN |
| TP\_BUSY |
| TP\_DOUT |
| TP\_IRQ |
| SD\_CS | SD card slot data control lines (SPI) |
| MISO |
| CLK |
| MOSI |

|  |  |  |
| --- | --- | --- |
| **Shield** | | |
| **Name** | **Arduino pin** | **Description** |
| 3.3V | 3.3V | Power supply |
| 5V | 5V |
| GND | GND | LCD ground bus |
| LEDA+ | 5V | LCD back light supply |
| DB0-DB15 | D37-D30, D22-D29 | LCD 16-bit parallel data interface |
| LCD\_RS | D38 | LCD data flow control lines |
| LCD\_WR | D39 |
| LCD\_RD | 3.3V |
| LCD\_CS | D40 |
| LCD\_RST | D41 |
| TP\_CLK | D6 | Touch Panel data flow control lines |
| TP\_CS | D5 |
| TP\_DIN | D4 |
| TP\_BUSY | D7 |
| TP\_DOUT | D3 |
| TP\_IRQ | D2 |
| SD\_CS | D53 | SD card slot data control lines (SPI) |
| MISO | D50 |
| CLK | D52 |
| MOSI | D51 |

|  |  |  |
| --- | --- | --- |
| **Arduino Mega 2** | | |
| **Name** | **Arduino pin** | **Description** |
| LED strip | A0 | To display the light intensity value at the current time |
| Light Detecting Resistor | A1 | To measure the light intensity and pass the value into the Arduino and convert into readable data to display on the LED strip |
| RST | D8 |  |
| SDA | D9 |  |
| HC-SR04 Echo | D13 |  |
| HC-SR04 Trig | D14 |  |
| Buzzer | D15 |  |
| Transmission | D18 | Serial Communication (Baud Rate 9600) |
| Receiver | D19 | Serial Communication (Baud Rate 9600) |
| Servo | D41 |  |
| Master in Slave Out | D50 |  |
| Master Out Slave In | D51 |  |
| SCK | D52 |  |

# Appendix E

**Arduino Mega 1 code for User Interface Touch Screen**

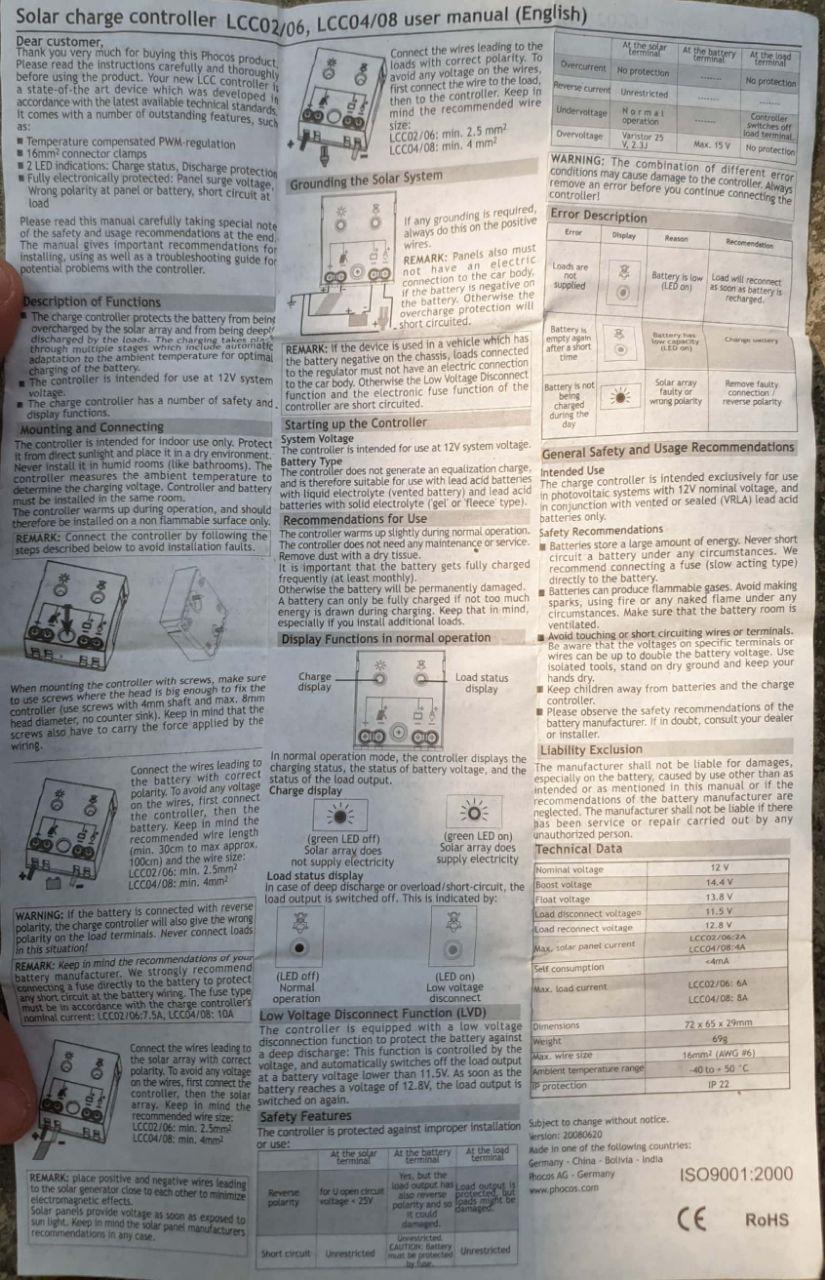
|  |
| --- |
| // This is UPDATED code for touchscreen interface done by SC  #include <UTFT.h>  #include <URTouch.h>  //==== Creating Objects =====  UTFT myGLCD(ILI9341\_16, 38, 39, 40, 41); //Parameters should be adjusted to your Display/Schield model  URTouch myTouch( 6, 5, 4, 3, 2);  //==== Defining Variables ======  extern uint8\_t SmallFont[];  extern uint8\_t BigFont[];  extern uint8\_t SevenSegNumFont[];  int x, y;  //Serial Communicatio Data Holder  char str[4];  char currentPage, selectedUnit;  //==== Ultrasonic Sensor ======  const int VCC = 13;  const int trigPin = 11;  const int echoPin = 12;  long duration;  int distanceInch, distanceCm;  //==============================================================================  void setup() {  //Serial Communication Establish  Serial.begin(9600);  Serial1.begin(9600); //use Serial port 1 for arduino to arduino communication  // Initial setup  myGLCD.InitLCD();  myGLCD.clrScr();  myTouch.InitTouch();  myTouch.setPrecision(PREC\_MEDIUM);  drawHomeScreen(); // Draws the Home Screen  currentPage = '0'; // Indicates that we are at Home Screen  selectedUnit = '0'; // Indicates the selected unit for the first example, cm or inches  }  //================================================================================  void loop() {  int i = 0;  //check for serial data  if (Serial.available()) {  int inByte = Serial.read();  Serial1.write(inByte);  }  if (Serial1.available()) {  delay(100); //allows all serial sent to be received together  while (Serial1.available() && i < 4) {  str[i++] = Serial1.read();  }  str[i++] = '\0';  }  if (i > 0) {  Serial.write(str, 4);  }  //Home page;  if (currentPage == '0') {  if (myTouch.dataAvailable()) {  myTouch.read();  x = myTouch.getX(); // X coordinate where the screen has been pressed  y = myTouch.getY(); // Y coordinates where the screen has been pressed  // If we press the Rent Button  if ((x >= 35) && (x <= 285) && (y >= 90) && (y <= 130)) {  drawFrame(35, 90, 285, 130); // Custom Function -Highlighs the buttons when it's pressed  currentPage = '1'; // Indicates that we are the first example  myGLCD.clrScr(); // Clears the screen  Serial1.write('o');// a - Rent Sequence; send 'o' to mega2  }  // If we press the Return Button  if ((x >= 35) && (x <= 285) && (y >= 140) && (y <= 180)) {  drawFrame(35, 140, 285, 180);  currentPage = '2';  myGLCD.clrScr();  Serial1.write('r');  }  }  }  if (currentPage == '1') { //at rental page; after clicking rent button  if (myTouch.dataAvailable()) {  myTouch.read();  x = myTouch.getX();  y = myTouch.getY();  }  // If we press the Back Button  if ((x >= 10) && (x <= 60) && (y >= 10) && (y <= 36)) {  drawFrame(10, 10, 60, 36);  currentPage = '0'; // Indicates we are at home screen  Serial1.write('b');  myGLCD.clrScr();  drawHomeScreen(); // Draws the home screen  }  }  if (currentPage == '2') { //at return page; after clicking return button  if (myTouch.dataAvailable()) {  myTouch.read();  x = myTouch.getX();  y = myTouch.getY();  //Back button  if ((x >= 10) && (x <= 60) && (y >= 10) && (y <= 36)) {  drawFrame(10, 10, 60, 36);  currentPage = '0';  Serial1.write('b');  myGLCD.clrScr();  drawHomeScreen();  }  }  }  //====================================================================  if (str[0] == 'a') //rent portion; there is stock, proceed to scan card  {  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Please Scan Card", CENTER, 60); // Prints the string on the screen  myGLCD.print("|||",CENTER, 90);  myGLCD.print("|||",CENTER, 110);  myGLCD.print("|||",CENTER, 140);  myGLCD.print("-------", CENTER, 170);  myGLCD.print("-----", CENTER, 180);  myGLCD.print("---", CENTER, 190);  myGLCD.print("-", CENTER, 200);  //Back button  myGLCD.setColor(100, 155, 203);  myGLCD.fillRoundRect (10, 10, 60, 36);  myGLCD.setColor(255, 255, 255);  myGLCD.drawRoundRect (10, 10, 60, 36);  myGLCD.setFont(BigFont);  myGLCD.setBackColor(100, 155, 203);  myGLCD.print("<-", 18, 15);  myGLCD.setBackColor(0, 0, 0);  myGLCD.setFont(SmallFont);  myGLCD.print("Back to Main Menu", 70, 18);  }  if (str[0] == 'n') //rent portion; no stock  {  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Out of Stock", CENTER, 30); // Prints the string on the screen  delay(2000); //flash meassage for 2 secs  myGLCD.clrScr();  currentPage = '0';  drawHomeScreen();  }  if (str[0] == 'c') //rent portion; collection  {  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Please Collect", CENTER, 30);  }  if (str[0] == 't') // return portion; no slot for return; t stands for taken  {  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("All Slots", CENTER, 30); // Prints the string on the screen  myGLCD.print("Are Used", CENTER, 60); // Prints the string on the screen  delay(2000); //flash meassage for 2 secs  myGLCD.clrScr();  currentPage = '0';  drawHomeScreen();  }  if (str[0] == 'e') // return portion; able to return; e stands for empty slot  {  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Please Scan Card", CENTER, 60); // Prints the string on the screen  myGLCD.print("|||",CENTER, 90);  myGLCD.print("|||",CENTER, 10);  myGLCD.print("|||",CENTER, 140);  myGLCD.print("-------", CENTER, 170);  myGLCD.print("-----", CENTER, 180);  myGLCD.print("---", CENTER, 190);  myGLCD.print("-", CENTER, 200);    //back button  myGLCD.setColor(100, 155, 203);  myGLCD.fillRoundRect (10, 10, 60, 36);  myGLCD.setColor(255, 255, 255);  myGLCD.drawRoundRect (10, 10, 60, 36);  myGLCD.setFont(BigFont);  myGLCD.setBackColor(100, 155, 203);  myGLCD.print("<-", 18, 15);  myGLCD.setBackColor(0, 0, 0);  myGLCD.setFont(SmallFont);  myGLCD.print("Back to Main Menu", 70, 18);  }  if (str[0] == 'g') // return portion; return powerbank after used; g stands for give back  {  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Please Insert", CENTER, 30); // Prints the string on the screen  myGLCD.print("Power Bank", CENTER, 60); // Prints the string on the screen  }  //display Thank you message; for both rent and return sides  if (str[0] == 'f')  { // when press return button at touchscreen and off the LED at mega2; touchscreen received 'f' after LED on mega2 is off and display thank you message on touchscreen  myGLCD.clrScr();  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Thank you!", CENTER, 30); // Prints the string on the screen  delay(2000); //flash meassage for 2 secs  myGLCD.clrScr();  currentPage = '0';  drawHomeScreen();  for ( int j = 0; j < sizeof(str); ++j ) {  str[j] = (char)0;  }  }  for ( int j = 0; j < sizeof(str); ++j ) {  str[j] = (char)0;  }  }  //================================================================================  // ====== Custom Funtions ======  // drawHomeScreen - Custom Function  void drawHomeScreen() {  // Title  myGLCD.setBackColor(0, 0, 0); // Sets the background color of the area where the text will be printed to black  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(BigFont); // Sets font to big  myGLCD.print("Powerbank Rental", CENTER, 10); // Prints the string on the screen  myGLCD.setColor(255, 0, 0); // Sets color to red  myGLCD.drawLine(0, 32, 319, 32); // Draws the red line  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.setFont(SmallFont); // Sets the font to small  myGLCD.print("by NTU Students", CENTER, 41); // Prints the string  myGLCD.setFont(BigFont);  myGLCD.print("Please Select", CENTER, 64);  // Button - Rent  myGLCD.setColor(16, 167, 103); // Sets green color  myGLCD.fillRoundRect (35, 90, 285, 130); // Draws filled rounded rectangle  myGLCD.setColor(255, 255, 255); // Sets color to white  myGLCD.drawRoundRect (35, 90, 285, 130); // Draws rounded rectangle without a fill, so the overall appearance of the button looks like it has a frame  myGLCD.setFont(BigFont); // Sets the font to big  myGLCD.setBackColor(16, 167, 103); // Sets the background color of the area where the text will be printed to green, same as the button  myGLCD.print("RENT", CENTER, 102); // Prints the string  // Button - Return  myGLCD.setColor(16, 167, 103);  myGLCD.fillRoundRect (35, 140, 285, 180);  myGLCD.setColor(255, 255, 255);  myGLCD.drawRoundRect (35, 140, 285, 180);  myGLCD.setFont(BigFont);  myGLCD.setBackColor(16, 167, 103);  myGLCD.print("RETURN", CENTER, 152);  }  // Highlights the button when pressed  void drawFrame(int x1, int y1, int x2, int y2) {  myGLCD.setColor(255, 0, 0);  myGLCD.drawRoundRect (x1, y1, x2, y2);  while (myTouch.dataAvailable())  myTouch.read();  myGLCD.setColor(255, 255, 255);  myGLCD.drawRoundRect (x1, y1, x2, y2);  } |

# Appendix F

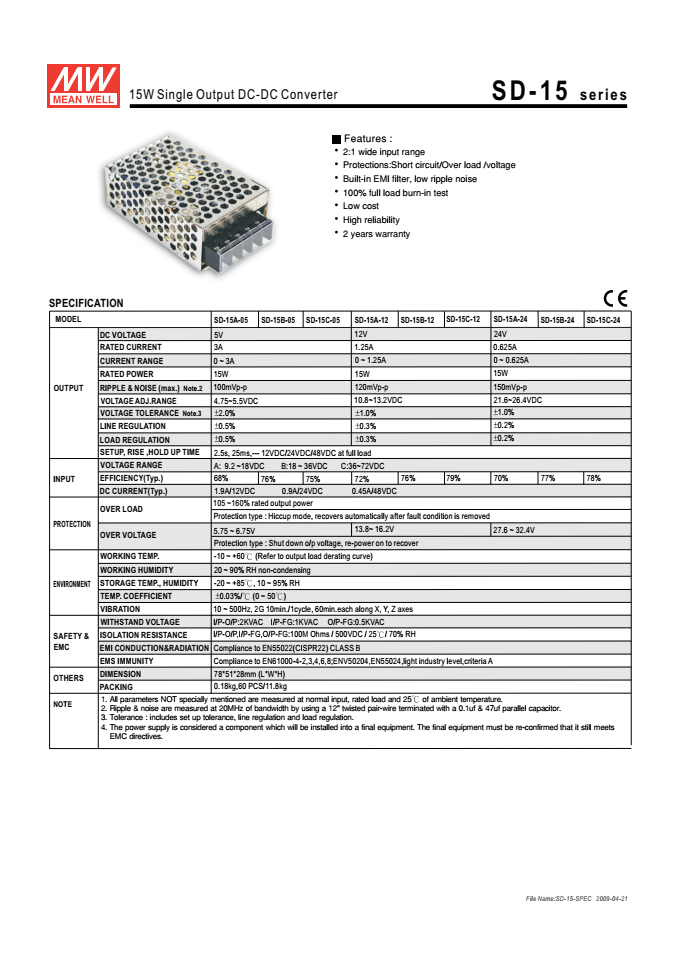
**Arduino Mega 2 code for Back End operation for Ultrasonic sensor, RFID, Servo, LED strip, Light intensity sensor**

|  |
| --- |
| #include <SPI.h>  #include <MFRC522.h>  #include <Servo.h>  /\*For Neopixel ledstrip troubleshooting\*/  #include <Adafruit\_NeoPixel.h>  #ifdef \_\_AVR\_\_  #include <avr/power.h>  #endif  #define RGBstrip A0  #define NUMPIXELS 7  #define sensorPin A1  Adafruit\_NeoPixel pixels = Adafruit\_NeoPixel(NUMPIXELS, RGBstrip, NEO\_GRB + NEO\_KHZ800);  uint32\_t yellow = pixels.Color(255, 100, 0);  uint32\_t green = pixels.Color(0, 150, 0);  /\*  PINOUT:  RC522 MODULE Uno/Nano MEGA  SDA D10 D9  SCK D13 D52  MOSI D11 D51  MISO D12 D50  IRQ N/A N/A  GND GND GND  RST D9 D8  3.3V 3.3V 3.3V  \*/  #define SDA\_PIN 9  #define RST\_PIN 8  //MFRC522 mfrc522(SDA\_PIN, RST\_PIN);  MFRC522 mfrc522(9, 8);  bool state;  // Set constant variables for pins used by the sensor, buzzer, servo and LEDs:  const int trigPin = A14; // Note: set these numbers to what ever pin you attached them to  const int echoPin = A13;  const int ledPin = A9;  const int actLedPin = A8;  const int buzzPin = A15;  //boolean alarmActive = false;  const String targetUID = "70 EC 7A A6";  //String targetUID[1]={"70 EC 7A A6", "70 17 7C A6"}  int servoPin1 = 40;  int servoPin2 = 41;  Servo servo; // create servo object to control a servo  int pos = 90; // default servo position in degrees  //=======================================  /\*For light detection purpose\*/  unsigned int intensity = 0;  int sensorValue = 0;  //we need to declare another ultrasonic sensor pin and names  //Variable for Ultrasonic Sensor  float flightTime\_microsec, flightTime\_sec;  float dist\_dist\_cm;  float soundspeed\_dist\_cmps;  float soundspeed\_mps = 343.5;  float dist\_cm;  char str[4];  const int LED = 13;  int inByte;  //========================================================================  void setup() {  Serial.begin(9600); //use for computer  Serial1.begin(9600); // use for boards  // Initialize RFID Reader:  SPI.begin();  mfrc522.PCD\_Init();  // Initialize LEDs:  //pinMode(ledPin, OUTPUT);  //pinMode(actLedPin, OUTPUT);  pinMode(buzzPin, OUTPUT);  pinMode(RGBstrip, OUTPUT);  // Initialize ultrasonic-sensor:  pinMode(trigPin, OUTPUT);  pinMode(echoPin, INPUT);  // Initialize servo:  servo.attach(servoPin1);  servo.attach(servoPin2);  //servo.write(); // Turn SG90 servo to X degrees  // Ensure alarm is set to off to prevent premature activation:  //alarmActive = false;  digitalWrite(actLedPin, LOW);  //Reset Servo Position  for (pos = 180; pos--;) //move the servo from 0 degrees to 180 degrees  {  servo.write(pos); // tell servo to go to position in variable 'pos'  delay(5); // increase 1 deg per 10millisecond  }  servo.detach();  // This is for Trinket 5V 16MHz, you can remove these three lines if you are not using a Trinket  #if defined (\_\_AVR\_ATtiny85\_\_)  if (F\_CPU == 16000000) clock\_prescale\_set(clock\_div\_1);  #endif  // End of trinket special code  pixels.begin(); // This initializes the NeoPixel library.  pixels.show();  }  //========================================================================  void loop() {  int i = 0;  if (Serial.available()) { //Writing TX after it send it flush by itself  int inByte = Serial.read();  Serial1.write(inByte);  }  if (Serial1.available()) { //Receiving RX  delay(100); //allows all serial sent to be received together  while (Serial1.available() && i < 4) {  str[i++] = Serial1.read();  }  str[i++] = '\0'; //delimiter  }  if (i > 0) {  Serial.write(str, 4);  }  if (str[0] == 'o') // if mega2(this board) receive 'o' after pressing the 'RENT' button on touchscreen(which send 'o' to mega2)  {  runSensor(); //on the ultrasonic Sensor to check stock for renting  }  if (str[0] == 'r') // return; if mega2 received 'r' after pressing the 'RETURN' button on touchscreen(which send 'f' to mega2)  {  mDist();  runReturn(); //on the ultrasonic Sensor to check availability for return  }  else if (str[0] == 'b') // back button is pressed  {  state = false;  }  for (int j = 0; j < sizeof(str); ++j)  {  str[j] = (char)0;  }  pixels.clear();  lightDetection();  for (int i = 0; i < intensity; i++) {  pixels.setPixelColor(i, yellow);  delay(1);  }  pixels.show();  }  //=====================================================================================  /\*Measure my distance interval between ultrasonic sensor to the ground\*/  float mDist()  {  digitalWrite(trigPin, HIGH);  delayMicroseconds(10);  digitalWrite(trigPin, LOW);  flightTime\_microsec = pulseIn(echoPin, HIGH);  flightTime\_sec = flightTime\_microsec \* pow(10, -6);  soundspeed\_dist\_cmps = soundspeed\_mps \* 100;  dist\_dist\_cm = soundspeed\_dist\_cmps \* (flightTime\_sec / 2);  Serial.println("My Current Distance:");  Serial.print(dist\_dist\_cm);  Serial.println();  dist\_cm = dist\_dist\_cm;  }  //===========================================================================  void runSensor() //under void loop  {  // Reset alarm at initialization to avoid premature activation:  noTone(buzzPin);  digitalWrite(ledPin, LOW);  //Measure the distance with Ultrasonic Sensor and print the distance  mDist();  // Check if there is any stock(powerbank):  checkStock(dist\_cm);  // Check if alarm has been triggered:  // checkAlarm(dist\_cm);  Serial.println("first run complete");  // Delay for 50 milliseconds before looping over again  // This ensures that the sensor is pulsing fast enough to detect fast-moving objects  // but still slow enough for the RFID Reader to check if a tag/card has been presented:  delay(100);  }  //======================================================================================  void checkStock(long dist\_cm) //subpart of runSensor(); rent portion  {  if (dist\_dist\_cm < 8)  //if (dist\_dist\_cm < 3) //less than 3dist\_cm; TRUE  {  Serial.println("Available"); //there is powerbank to rent  //digitalWrite(ledPin, HIGH); //LED light up to show there is powerbank  Serial1.write('a'); // send 'a' to touchscreen to display "Scan card"; a stands for available  state = true;  }  else //more than dist\_cm; FALSE  {  Serial.println("No Stock");  //digitalWrite(ledPin, LOW);  Serial1.write('n'); // send 'n' to touchscreen to display "Out of Stock"; n stands for no stock  delay(500);  state = false;  }  while (state)  {  readForCard();  }  pixels.clear();  }  //======================================================================  void readForCard() //under void loop  {  Serial.println("I'm reading your RFID");  // Look for new cards:  while ( ! mfrc522.PICC\_IsNewCardPresent())  {  return;  }  // Select one of the cards:  while ( ! mfrc522.PICC\_ReadCardSerial())  {  return;  }  // Get UID from card/tag:  String content = "";  byte letter;  for (byte i = 0; i < mfrc522.uid.size; i++)  {  content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));  content.concat(String(mfrc522.uid.uidByte[i], HEX));  }  // Check UID for match and set alarm:  content.toUpperCase();  // Change the UID below to the tag/card that you want to use for access  // Use DumpInfo (File > Examples > MFRC522 > DumpInfo) to get this infomation  if (content.substring(1) == targetUID)  {  servo.attach(servoPin1);  servo.attach(servoPin2);  //digitalWrite(actLedPin, HIGH); //on led  // Set off a short tone and an LED to signal a sucessful activation/deactivation:  tone(buzzPin, 500); //on  delay(50);  noTone(buzzPin); //off  delay(50);  tone(buzzPin, 500); //on  delay(50);  noTone(buzzPin); //off  Serial1.write('c'); // send 'c' to touchscreen to display message  analogWrite(RGBstrip, 255);  //on servo to push powerbank out of the slot  for (pos = 0; pos <= 180; pos++) //move the servo from 0 degrees to 180 degrees  {  servo.write(pos); // tell servo to go to position in variable 'pos'  delay(5); // increase 1 deg per 10millisecond  }  for (pos = 180; pos >= 0; pos--) //move the servo from 180 degrees to 0 degrees  {  servo.write(pos); // tell servo to go to position in variable 'pos'  delay(5); // increase in 1 deg per 10millisecond  }  servo.detach();  sensorbuzz();  }  else // Make a short tone to indicate that the alarm has failed to activate:  {  Serial.println("ACCESS DENIED - ERROR");  tone(buzzPin, 800);  delay(200);  }  if (str[0]== 'b'){  state = false;  }  state = false;  }  //==========================================================================  void sensorbuzz() //under void loop  {  // Reset alarm at initialization to avoid premature activation:  //Ultrasonic Sensor read and measure the distance then return back (1 time)  pixels.clear();  for (int i = 0; i < NUMPIXELS; i++) {  pixels.setPixelColor(i, green); // Moderately bright green color.  pixels.show(); // This sends the updated pixel color to the hardware.  }  // Check if alarm has been triggered:  //checkAlarm(dist\_cm);  do {  mDist();  tone(buzzPin, 2000, 500);  } //while (dist\_cm < 3); // still detecting powerbank  while (dist\_cm > 3);  //Power Bank Taken and send Thank you message  //digitalWrite(ledPin, LOW);  Serial1.write('f');  state = false;  // Delay for 50 milliseconds before looping over again  // This ensures that the sensor is pulsing fast enough to detect fast-moving objects  // but still slow enough for the RFID Reader to check if a tag/card has been presented:  delay(50);  }  //===================================================================  void runReturn()  {  // Reset alarm at initialization to avoid premature activation:  noTone(buzzPin);  //digitalWrite(ledPin, LOW);  mDist();  // Check if any space is occupied:  checkbank(dist\_cm);  Serial.println("Return Process Complete");  // Delay for 50 milliseconds before looping over again  // This ensures that the sensor is pulsing fast enough to detect fast-moving objects  // but still slow enough for the RFID Reader to check if a tag/card has been presented:  delay(100);  }  //=================================================================================  void checkbank(long cm) //subpart of runReturn(); for return portion; taking  {  if (dist\_cm > 4)  //if (dist\_cm < 3) //powerbank space is occupied  {  Serial.println("Occupied");  //digitalWrite(ledPin, LOW); //LED off when there is no available space  Serial1.write('t'); // send 't' to touchscreen; t stands for taken  //RGB led strip show RED for no available space  pixels.setPixelColor(7, pixels.Color(255, 0, 0));  pixels.show();  delay(1000);  }  else //there is an empty space, more than or equal to 3cm  {  Serial.println("Empty");  //digitalWrite(ledPin, HIGH); //LED light up to show there is slot to return  Serial1.write('e'); // send 'e' to touchscreen; e stands for empty  state = true;  //RGB led strip show GREEN for available space  pixels.setPixelColor(7, pixels.Color(0, 255, 0));  pixels.show();  delay(1000);  }  while (state)  {  CardforReturn();  }  }  //==========================================================================================  void CardforReturn() //under void loop  {  Serial.println("I'm reading your card for returning");  // Look for new cards:  if ( ! mfrc522.PICC\_IsNewCardPresent())  {  return;  }  // Select one of the cards:  if ( ! mfrc522.PICC\_ReadCardSerial())  {  return;  }  // Get UID from card/tag:  String content = "";  byte letter;  for (byte i = 0; i < mfrc522.uid.size; i++)  {  content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));  content.concat(String(mfrc522.uid.uidByte[i], HEX));  }  // Check UID for match and set alarm:  content.toUpperCase();  // Change the UID below to the tag/card that you want to use for access  // Use DumpInfo (File > Examples > MFRC522 > DumpInfo) to get this infomation  if (content.substring(1) == targetUID)  {  digitalWrite(actLedPin, HIGH); //on led  // Set off a short tone and an LED to signal a sucessful activation/deactivation:  tone(buzzPin, 500); //on  delay(50);  noTone(buzzPin); //off  delay(50);  tone(buzzPin, 500); //on  delay(50);  noTone(buzzPin); //off  Serial1.write('g'); // send 'g' to touchscreen to display insert powerbank message  SenReturn();  }  else // Make a short tone to indicate that the alarm has failed to activate:  {  Serial.println("ACCESS DENIED - ERROR");  tone(buzzPin, 800);  delay(200);  }  state = false;  }  //==========================================================================  void SenReturn()  {  // Reset alarm at initialization to avoid premature activation:  noTone(buzzPin);  //digitalWrite(ledPin, LOW);  //Check if powerbank is returned:  /\*do {  mDist();  } while (dist\_cm >= 3); //checking when no powerbank is detected  laststep(dist\_cm);\*/  do {  mDist();  } while (dist\_cm < 4.5);  if (dist\_cm < 8)  {  mDist();  delay(100);  }  else {  SenReturn();  }  laststep(dist\_cm);  // Delay for 50 milliseconds before looping over again  // This ensures that the sensor is pulsing fast enough to detect fast-moving objects  // but still slow enough for the RFID Reader to check if a tag/card has been presented:  delay(100);  }  //=================================================================================  // Checks if the sensor has detected an object and triggers the alarm  void laststep(long cm) //subpart of SenReturn()  {  if (cm < 10)  //if (cm < 3)  {  Serial.println("Received");  Serial1.write('f'); //send 'f' to touchscreen to display Thank you message  //digitalWrite(ledPin, HIGH);  pixels.setPixelColor(7, pixels.Color(0, 255, 0)); //show green after success transaction  pixels.show();  delay(1000);  tone(buzzPin, 500); //RFID and alarm are activated, Buzzer on when still detecting powerbank in the range  delay(200);  tone(buzzPin, 300);  delay(200);  tone(buzzPin, 100);  delay(500);  noTone(buzzPin);  }  /\*else if (alarmActive == false) //alarm is not activated but powerbank in range  {  Serial.println("Waiting for card reader");  digitalWrite(ledPin, LOW);  noTone(buzzPin);  //delay(150);  }\*/  else // more than 3cm  {  Serial.println("Not in range"); //powerbank is taken  //digitalWrite(ledPin, LOW);  noTone(buzzPin);  }  }  //=======================================================================================  void lightDetection()  {  sensorValue = analogRead(sensorPin);  intensity = sensorValue / 50;  /\*Set an indication of maximum intensity at 7 led strip  low light intensity at 1  so it is from the range of 1-7 brightness indication  \*/  } |

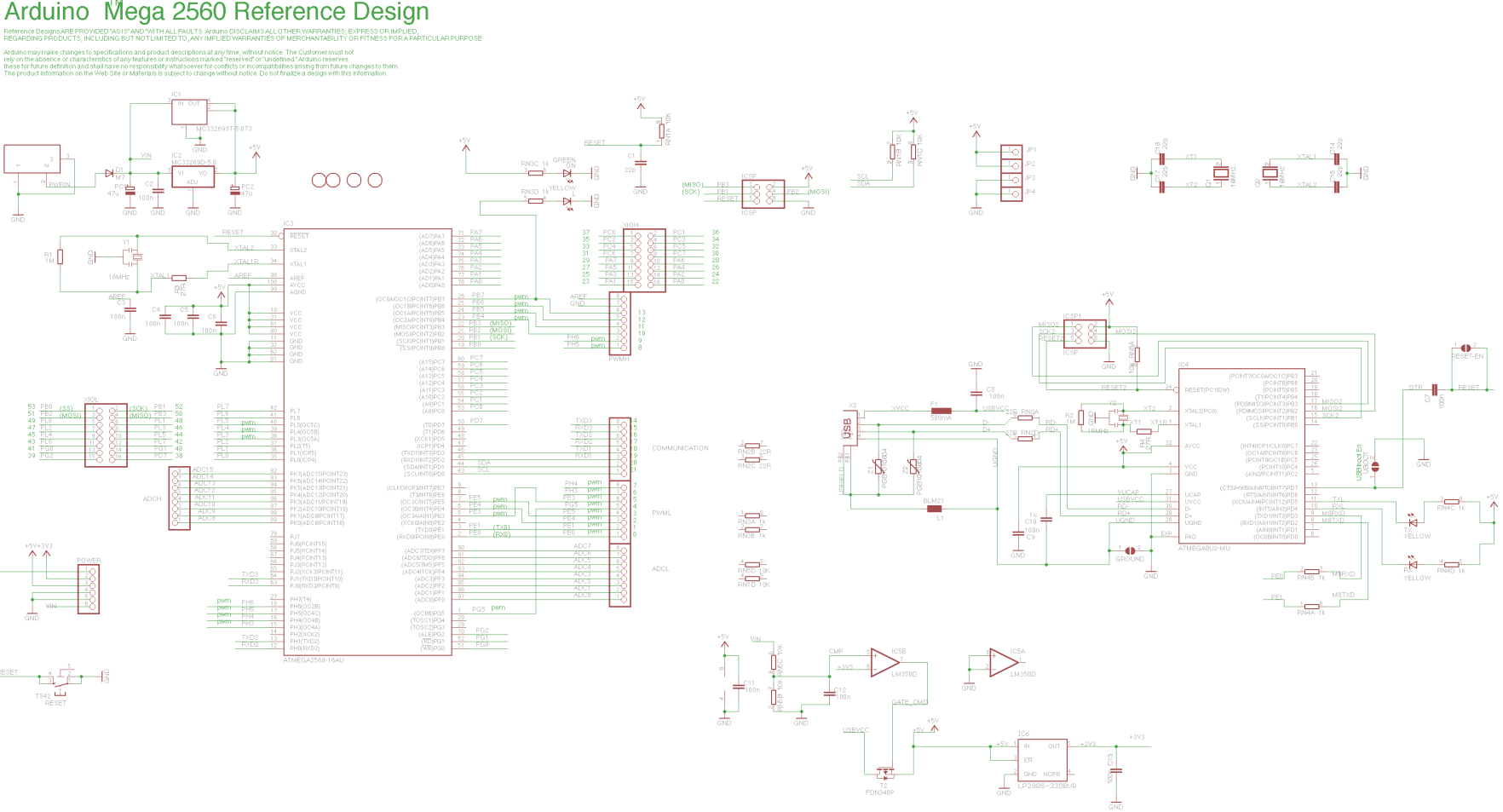
# Appendix G



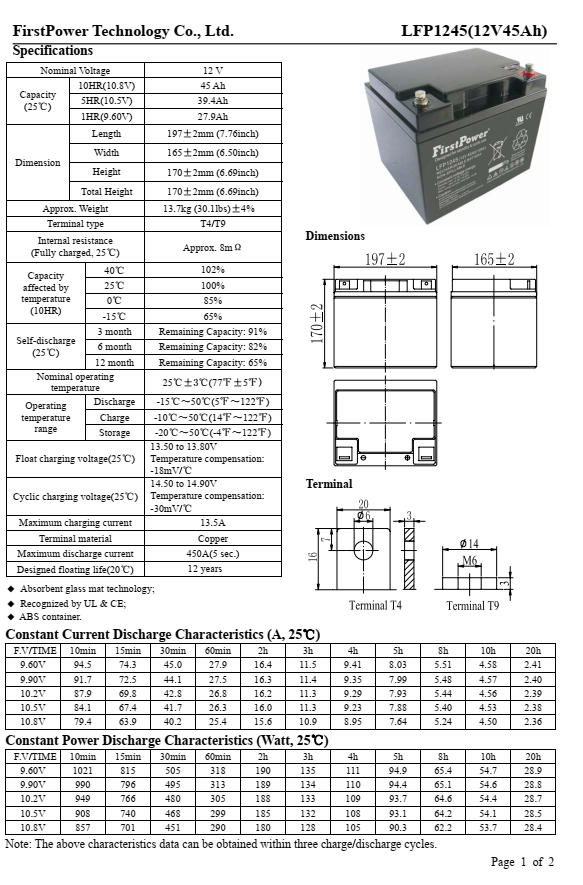
# Appendix H

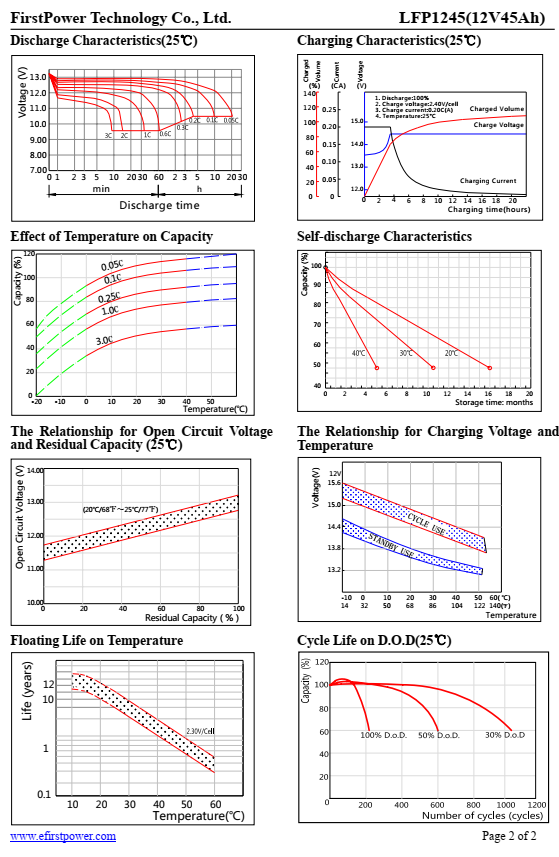


# Appendix I

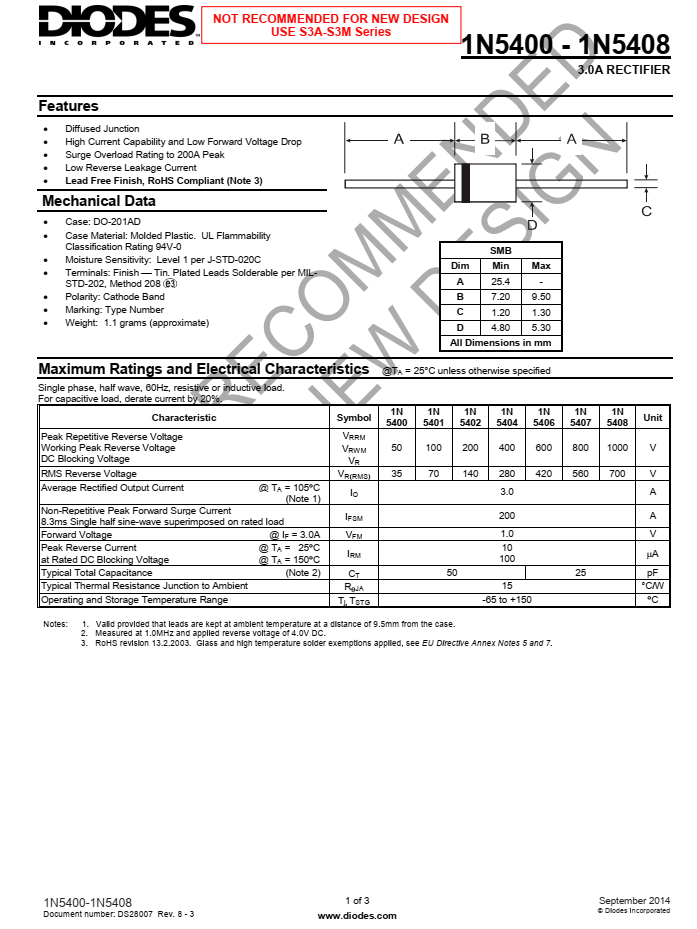


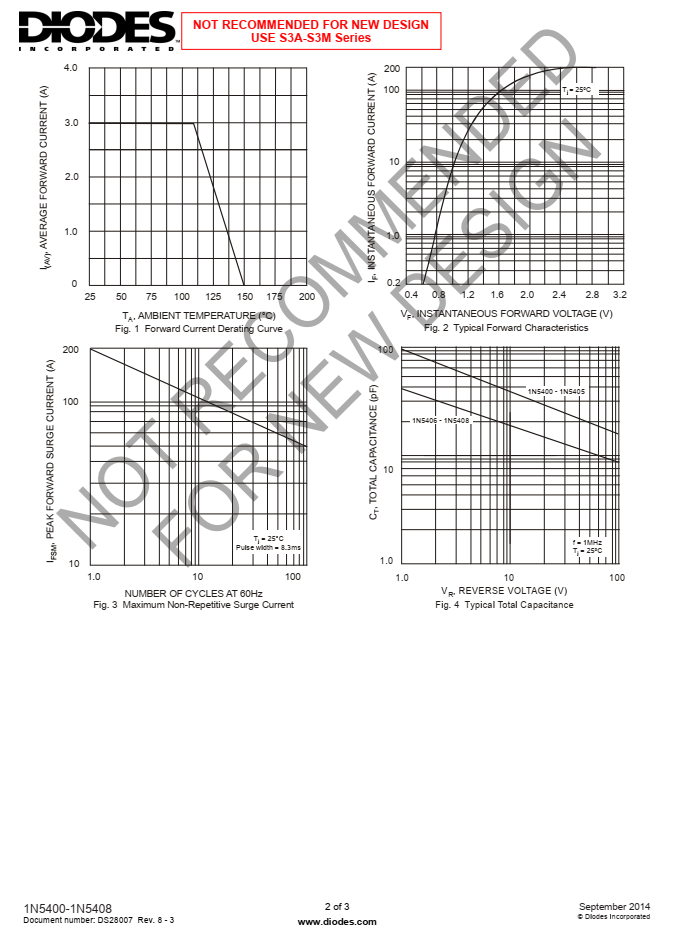
# Appendix J

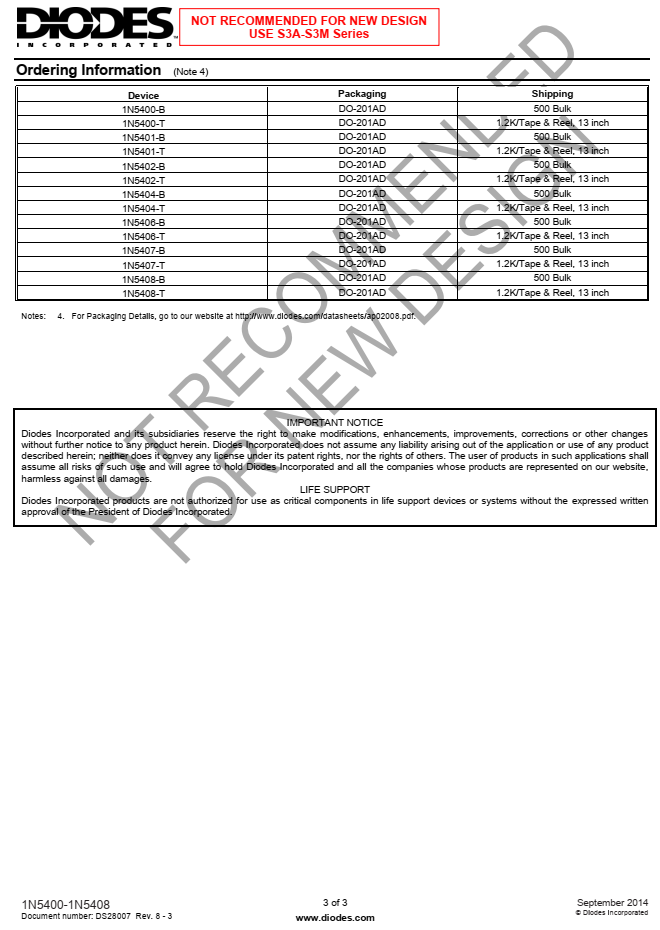




# Appendix K

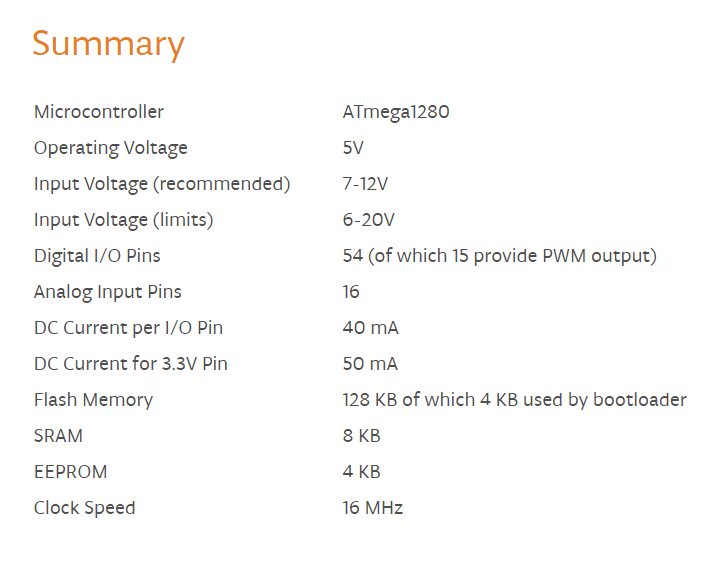






# Appendix L

# Appendix M



# Appendix N

