Winter 2018

Soil Moisture Sensor

Jaclyn Miller
jmm329@zips.uakron.edu

Derek Bitecofer

Seung Jun Lee

Soil Moisture Sensor

Final Design Report

Design Team Number: 02

Derek Bitecofer

Seung Jun Lee

Jaclyn Miller

Faculty Advisor: Dr. Kye-Shin Lee

Senior Design Coordinator: Gregory A. Lewis

Date:

Additional Contributors: Mohammed Albusaleh

Table of Contents

Table of Figures

Table of Tables

Abstract

Because water is an important resource and not all communities around the world can afford to be liberal with their water needs; it has become important to use available water as efficiently as possible, especially in agriculture. For the purpose of reducing the overwatering of crops, an unattended ground moisture sensor can be implemented to measure the current moisture level in the soil surrounding the plants. This will allow a farmer to know when to water/stop watering his crop. For convenience, the moisture data information should be transmitted wirelessly to the user. The design of an unattended ground moisture sensor and wireless communication/user interface system is discussed. The sensor design consists of a Wheatstone bridge for determining the resistance of the soil, followed by a differential amplifier for converting the measured resistance into a voltage. This is done because there exists a correlation between moisture and resistance. This voltage is interpreted by a microcontroller as moisture data and sent wirelessly to a Lora communication receiving node by a Lora communication transmitting node. The receiving node then relays that information to a PC host for access by the user. The system has a power circuit that consists of a battery and a linear regulator.

1.0 Problem Statement

1.1 Need

With an increase in globalization and urbanization, a need for more productive and efficient methods for growing crops for an increasing population and a decreasing amount of farmland is very prominent. There has also been an increase in pollution in the global hemisphere that has and will continue affect the production of crops. Thus, a need arises to create a soil moisture system with statistical analysis of the data to aid in the monitoring of crop growth and production.

Furthermore, pollution will be decreased with the production of a self-sufficient device which does not require external electrical power

[MA, JM]

1.2 Objective

The objective is to create moisture sensor that is convenient for farmers. The system will measure the moisture of the soil periodically throughout the user-selected time period. It will report the measured data to the user through a statistical interface which will allow for data analysis.

[MA, JM]

1.3 Research Survey

1.3.1 Patent Search

Three patents regarding moisture sensors were found. The first one is US 5424649 A, published on June 13, 1995. The patent is for a soil moisture sensor using sensor electrodes to sense capacitance which is proportional to the moisture of the soil [1]. It is similar to the proposed design because both the patented project and the idea regards sensing the moisture of soil. However, there are many differences between the project and the patented one. Primarily, the device will be self-sufficient, in that it will not require being plugged into a wall. This device will also record the data measured and send it to the user for statistical analysis.

The second patent is US 20080199359 A1, published on August 21, 2008. The patent is for a soil moisture sensor which measures the moisture of the soil and stores the values in its memory [2]. It is similar to our proposed design because, like the previous patent, both the patented project and our idea regards sensing the moisture of soil. Both projects also store the sensor readings in the memory. However, this project is different because it will provide the stored data to the user for statistical analysis.

The third patent is US 5621669 A, published on April 15, 1997. It is a probe used to measure moisture sensor and a controller for external actuators [3]. It is similar to our proposed design because, again, both the patented project and our idea regards sensing the moisture of soil. However, it is different because this device is simply a probe, not a fully-functioning measurement system. It does not store the measured values, nor does it provide the user with the data. Finally, it can be used as a controller, for which this project cannot be used.

[MA]

1.3.2 Energy Harvesting Component

Due to the fact that this moisture sensor system needs to be as automated and as self-sufficient as possible, an energy harvesting component needs to be added to the system. This can be done numerous ways, however, since this system is designed to be a low-cost system that can be used in various areas, a solar panel is the best option for this system. The solar panel will power all of the electronics in the soil moisture sensor. However, solar panels experience many issues in supplying a constant power, especially on cloudy days. Thus, a backup battery will be needed for the solar panel. On sunny days, the solar panel will power the whole system and will charge the backup battery supply. The battery will power the electronics when the solar panel cannot. For safe measure, a sensor should be attached to the solar panel and the DC battery supply to measure the power output.

Due to weather conditions, there is a chance that the power output could be zero because no energy has been harvested over a long period of time. If this were to happen, a notification would be sent to the user informing him/her to switch the depleted battery with a fully charged battery. In the article, “Automated Irrigation System Using Solar Power,” a similar system is used to power their irrigation system. Differences between their proposed model and this system's proposed model is that this system has an extra power check to make sure there is always power to the system [4].

[JM]

1.3.3 Soil Moisture Sensor System

Soil moisture sensor systems can be implemented in a variety of ways. A simple resistive soil moisture sensor can be used to measure the resistance of the soil, which is proportional to the amount of water content in the soil. However, the resistance of soil is also dependent on temperature, thus an accompanying temperature sensor would increase the accuracy of the sensor. Another strategy would be to use a dielectric moisture sensor. These sensors are essentially capacitors, where the soil becomes the dielectric part of a capacitor and since the dielectric constant (ratio of permittivity of a material to the permittivity of air) changes as moisture builds (as the permittivity changes), so will the capacitance of the sensor [5]-[6]. [DB]

1.3.4 Soil Moisture Sensor Technology Comparison

The research survey of current soil moisture sensor technology is needed to select the most efficient and feasible sensor for this project. The research details a case study of four different types of moisture sensor (Capacitive dielectric sensor, two different types of conductivity sensors, and a resistance block sensor) to compare their performance based on the amount of water (in inches) applied with ET (EvapoTranspiration) information. The ET information simply tells how much water should be used and by subtracting the rainfall from ET values will give the required amount of water should be used. The case study was conducted both in clay soil and sand soil from May to September. Each plot is roughly 9' x 9' in size and each sensor was programmed to water daily with four start times of five minutes each, with at least one hour between cycles [7]. The results show that the best water saving sensor in clay soil is Capacitive dielectric sensor (Total of 11.30 inches while water requirement is 15.57 inches), and the best water saving sensor in sand soil is the resistance block sensor (Total of 22.88 inches while water requirement is 15.57 inches). These results implies that the environmental factors impact the moisture sensor performance and accuracy in multiple different soils. Other than environmental factors, there are many tradeoffs between cost and accuracy, and also power consumption. The overview of the sensor technology comparison is shown below Table 1.

Sensor	Method	Advantage	
Fiber Optics	-Measures the change of the intensity of the light travel exiting the probe to soil and entering back to the probe from soil.	-Takes into consideration weather predictions	-Expensive -Very new technology
Tensiometers	-The built-in vacuum gauge inside the plastic tube measures the pressure -The pressure changes by the water pulled out of the soil -Measured in centimeter: Higher reading means less moisture and a lower reading means high moisture	-Easy to read the data -Easy to operate -Fast measurement -Low power consumption	-$45-80 (vary by the length, 6-48" of the probe) -Needs routine maintenance -Cannot use during winter
Electrical Resistance Blocks: Granular Matrix (Watermark)	-Similar method to Gypsum Blocks	-Better Version of Gypsum Blocks -High accuracy -Easy to operate -Low cost ($25-$50) -Wide range of soil moisture -Long duration (5-7 years) -Low power consumption	-Lower accuracy compared to dielectric sensor -Slow
Electrical Resistance Blocks: Gypsum Blocks	-Two electrodes inside the porous material, such as gypsum, measures the resistance itself -The water from the soil moves into the gypsum decreases the resistance and water pulled from the gypsum increases the resistance -Low resistance means higher moisture level and vice versa	-Low cost -Accurate in clay soil -Easy to operate -Low power consumption	-Duration is around 1-2 years -Low accuracy in sandy soils -Less repeatability
Electrical Conductivity Probes	-Measure the current of electricity between two probes (direct contact with soil) -More moisture have better the conductivity and vice versa	-High accuracy in clay soil -Low cost	-Very sensitive to the spacing of the probes and soil type -Less repeatability
Heat Dissipation	-The ceramic medium of sensor measure heat dissipated by the soil -Higher dissipation has higher moisture level and vice versa	-Independent of soil type or salinity influences	-High power consumption
Dielectric: Capacitance	-Two electrodes of dielectric have direct contact with soil, and high oscillating frequency is applied to the electrodes and measures resonant frequency -The resonant frequency vary by moisture level of soil -Large change in frequency have higher moisture level and vice versa	-High accuracy -Good for research use -Read soil volumetric water content directly -Low maintenance	-Expensive -Not practical for controlling irrigation system
Dielectric: TDR (Time Domain-Reflectometry)	-Measurement of time travel along the length of probe rod by electromagnetic pulse -More travel time means higher moisture level and vice versa	-High accuracy -Good for research use -Read soil volumetric water content directly -Low maintenance	-Expensive -Very complex -Not practical for controlling irrigation system

Table 1 - Soil Moisture Sensor Technology Comparison

[SJL]

1.3.5 User Interface

The user-interface should allow the user to record the useful data (moisture level) and remotely control the sensor device. There also needs to be a notification system, either by e-mail or text, that notifies users if there are any changes in the soil. The user-interfaces can be achieved by module interface via the IoT (The Internet of Things) system between the sensor, microcontroller, wireless communication, the database, and the users' devices (either webpage or mobile application). The components of the IoT system can include the sensor, the microcontroller, the wireless communication, the database and the user application. During data transmission from the moisture sensor to the user's device, the data is sent from the microcontroller through the wireless module. The data will be stored in the main server and the user can access and modify the data through the application. The application will be available in a web based application and a mobile application which will be supported on both ios and android. The mobile application engine can be designed by the PhoneGap application which provides an integrated program environment for both web-based and mobile-based application. The article, "Smart Discrete Water Quality Sensor" shows the user-interface between user and water sensor so that the user can retrieve the data in specific time of graph using PIC microcontroller, wi-fi module and mobile application.

The microcontroller has to be operated all the time. The clock of the microcontroller should always keep track of real time data even when the microcontroller is in sleep mode. This can be done by a RTCC module. Since the power supply of the system is a small battery, the system has to be operated efficiently. The article entitled “Design and Implementation of a Soil Moisture Wireless Sensor Network” details the efficient wireless network architecture to improve the lifetime of the sensor. A possible solution can be a scheduling of collecting data (sleep and awake periods). This technology is useful when a small number of sensors are used to cover a broad area. On the user’s side, the data should be available in a graph so the data can be analyzed by within a given period of time. There can be several features to analyze the data.

[SJL]

1.5 Objective Tree

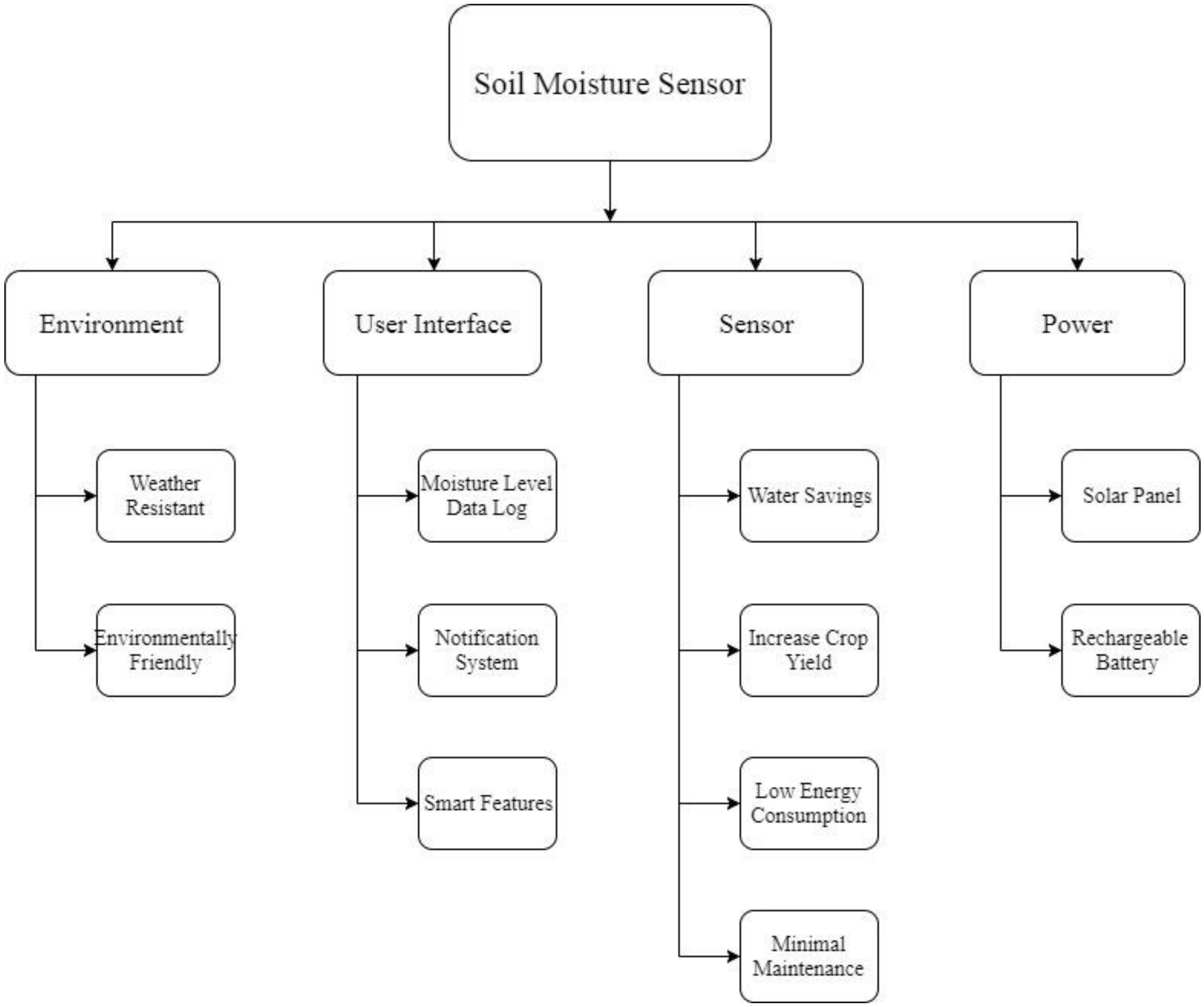


Figure 1 - Objective Tree

[JM, SJL]

2.0 Design Requirement Specification

The following section contains the set of marketing requirements to be satisfied by the proposed design. Definitions of the terminology used in the marketing requirements are as follows:

The term ‘system’ refers to the completed project.

The system consists of a battery, a sensor designed by the team, a wireless communication (the Lore End-device with the built-in microcontroller and the Lora Gateway as a base station), a means to transport data and a user interface in the form of a phone app and/or desktop app.

2.1 Engineering and Marketing Requirements

Marketing Requirements	Engineering Requirements	Justification
1	The system must measure the soil moisture level 24 times a day.	The system should gather enough data for the user to analyze.
2, 7	The system must be able to operate within 0 to 50 degrees Celsius.	The power unit will not operate outside of this temperature region.
3, 6	The system will use less than 6W when measuring moisture.	The power unit does not supply more than 6W.
4	The system should relay the moisture level to the user within 5 minutes of the latest measurement taken.	The user may want to immediately know the moisture level of the crops.
1, 4	The system will notify the user of continuous non-optimum levels for 48 hours.	The user will want to know if the soil is not at the moisture level that the crops need.Therefore, the user can take corrective action to mitigate the problem.
4	The system will complete its measurement process within 5 minutes.	The measurements need to be taken in a small time period to ensure that the numerical values are not thrown off by outliers caused by environmental factors.
4	The system should notify the user if the moisture level reads a number 3 or more standard deviations away from the last number taken.	This will ensure that if there is a dramatic change such as flooding or the removal of the device, the user will be notified.
3, 6	The sensor will draw 0A when not measuring the moisture level.	The system should use as little energy as possible when not in use. Therefore, the battery can power the system during long periods of insufficient light.
Marketing Requirements 1. The system should sense and store the soil moisture level. 2. The system should be weather resistant. 3. The system's power should be supplied by the power unit. 4. The system will have a user interface that allows the user to control the system with smart features that may include, but not limited to averages, charts, graphs and notifications. 5. The system will be environmentally friendly. 6. The system will be energy efficient as possible. 7. The system will require minimal maintenance and be durable.		

Table 2 - Engineering and Marketing Requirements

[JM]

3.0 Accepted Technical Design

Any design idea written within this section of "Accepted Technical Design" may or may not be improved upon or changed throughout the implementation phase of this project. See section 4.0 "Accepted Implemented Design" for further details.

3.1 Hardware Design

3.1.1 Block Diagrams and System Overview

The hardware of the moisture sensor will consist of three main parts: power, moisture sensor, and communications. The power section contains a battery. The solar panel will take energy from the sun and use it to power the system or charge the battery. The battery will be large enough to last the duration of the planting and harvesting season. Thus, the system will be made as energy efficiently as possible. The sensor will simply be a moisture sensor used to sense the amount of moisture in the soil. The communications contain a microprocessor and wireless communication modules. The sensor will transmit data to the microprocessor and the microprocessor will transmit data to the user on a computer or phone.

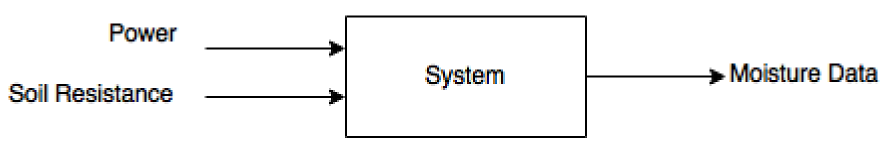


Figure 2 - Hardware Block Diagram Level Zero

Module	Unattended Ground Moisture Sensor
Designers	Mohammed Albusaleh & Derek Bitecofer
Inputs	Power from the Battery Soil Resistance (to determine moisture level)
Output	Moisture Data
Description	The system will be powered by the battery and will acquire soil moisture data.

Table 3 - Hardware Functional Requirement Level Zero

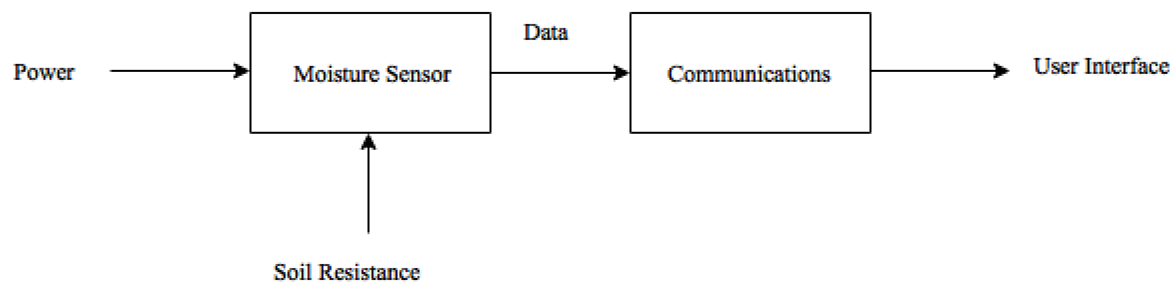


Figure 3 - Hardware Block Diagram Level One

Module	Moisture Sensor
Designers	Mohammed Albusaleh & Derek Bitecofer
Inputs	Battery Power Soil Resistance
Output	Data
Functionality	The moisture sensor will calculate the moisture level of the soil based on the soil resistance and provide the data to the microcontroller.

Table 4 - Hardware Functional Requirement Level One (Sensor)

Module	Communications
Designers	Mohammed Albusaleh & Derek Bitecofer
Input	Sensor Moisture Level Data
Output	Moisture Level Data for User Interface
Functionality	The moisture sensor system will provide the user with data about the moisture level of the soil.

Table 5 - Hardware Functional Requirement Level One (Communications)

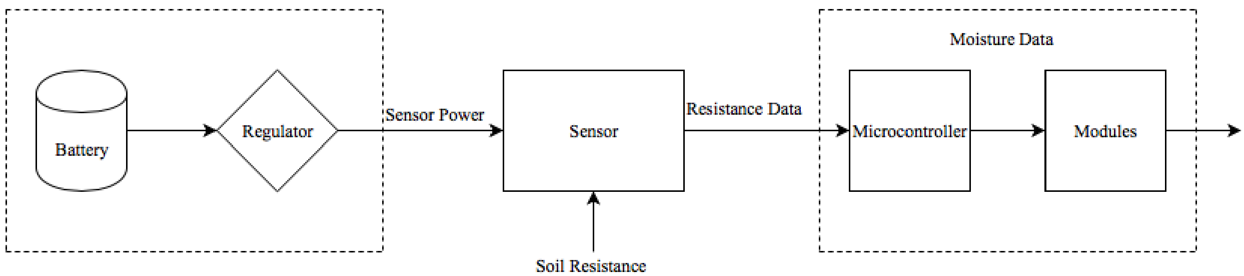


Figure 4 - Hardware Block Diagram Level Two

[DB]

Module	Communication & Microcontroller
Designers	Mohammed Albusaleh & Derek Bitecofer
Input	Data
Output	User Interface
Functionality	The microcontroller's onboard ADC will convert the analog data collected from the sensor into digital data. The data will be sent to the user using In Module / GateWay

Table 6 - Hardware Functional Requirement Level Two (Communication & Microcontroller)

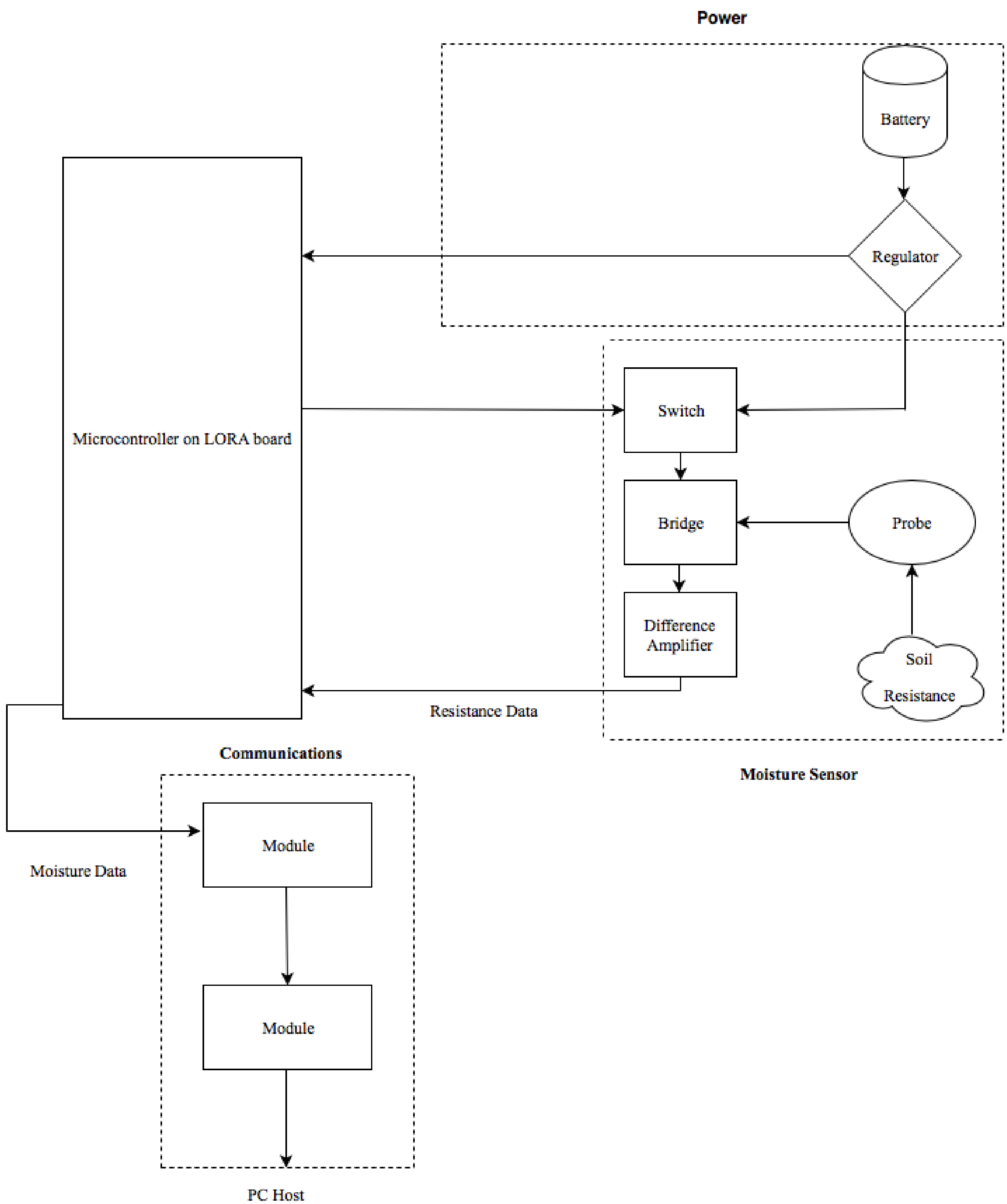


Figure 5 - Hardware Block Diagram Level Three (System Overview)

[DB]

Module	Sensor
Designers	Mohammed Albusaleh & Derek Bitecofer
Inputs	Power Resistance of Soil Signal out for Switch
Outputs	Moisture Sensor Data for Communications
Functionality	The sensor will have two probes that with the soil, form a resistor for a circuit. As the resistance of the soil changes, so will the output of the circuitry. The correlation between moisture, resistance and the output signal will be utilized in determining the proper moisture of the soil. To make the sensor as energy as efficient as possible, the circuit should not be powered continuously. Therefore, a signal from the microcontroller will be used to control a switch that will determine if the sensor is being powered.

Table 7 - Hardware Functional Requirement Level Three (Sensor)

Module	Microchip
Designers	Mohammed Albusaleh & Derek Bitecofer
Inputs	Power from Battery ADC Inputs from Voltage Monitors
Outputs	PWM for Switches Moisture Sensor Data for Communications
Functionality	The microchip is the brain of the system. It gives PWM outputs for the switches and the moisture data to the communications for the user interface.

Table 8 - Hardware Functional Requirement Level Three (Microchip)

Module	Communications
Designers	Mohammed Albusaleh & Derek Bitecofer
Inputs	Power from Battery PWM Input from PIC
Output	Moisture Sensor Data for User Interface
Functionality	The communications module transfers the moisture data from the sensor to the user interface.

Table 9 - Hardware Functional Requirement Level Three (Communications)

3.1.2 Design Theory

The hardware of this project boils down to three main sections: sensor, communications, and microprocessor. Each section is discussed in detail below.

3.1.2.1 Power Design Theory

Please see the appendix *10.1 Initial Power Design* for the extensive design that was originally made. Due to the loss of a team member, the power design needed to be reduced in complexity. There will no longer be the complicated switching conditions or the low battery conditions. Instead, there is the following:

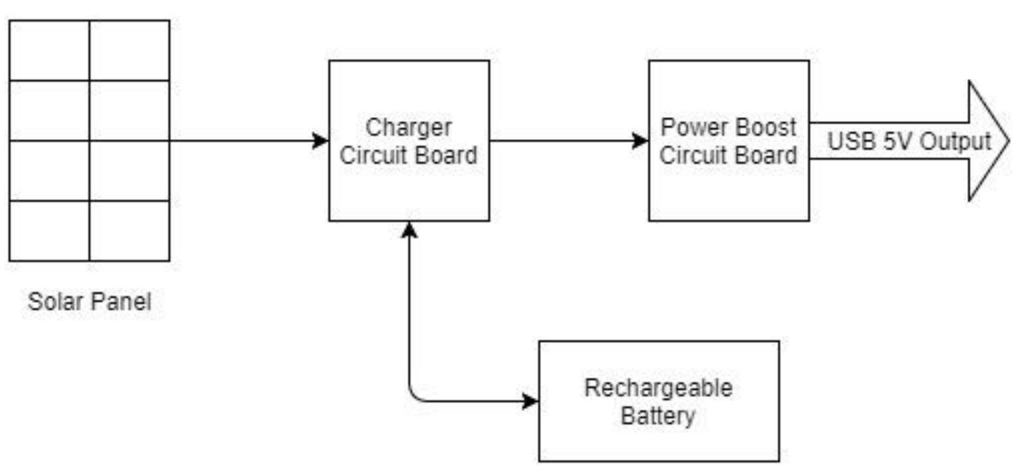


Figure 6 - Power Design

The voltage coming from the solar panel is quite uneven, thus the charger circuit board is needed to smooth out the voltage. This board will then be able to recharge the battery. The charger board also has another output to the power boost circuit board. This board will boost a voltage to approximately 3V to 5V. There are multiple different power boost circuit boards that can be used that have varying output current. During nighttime or during a very cloudy time, the rechargeable battery will supply power to the charger circuit board which can then power the boost circuit board. The 5V output will be used to supply the rest of the components in the hardware or software design. The 5V output was used because the majority of microcontrollers as well as many other components require a 5V input.

[JM]

3.1.2.2 Moisture Sensor

The moisture sensor will measure the resistance fluctuations in soil and send that information to a microcontroller in the form of a voltage. The voltage information will then be converted into moisture data using equations and a model. Firstly, the resistance measured will be related to the output signal by an equation. This resistance will then be related to moisture content by a measured model. The details of which are discussed below.

[DB]

3.1.2.2.1 Wheatstone Bridge

A wheatstone bridge circuit is capable of determining an unknown resistance when compared with three known resistances. The circuit acts like a “scale” where the difference in the “weight” is a voltage. The voltage in question is measured at the junction of two of the known resistances; and at the junction with the other known resistance, along with the unknown resistance. See

Figure 7 - Wheatstone Bridge Example. An intriguing fact about the nature of the wheatstone bridge that renders the circuit useful for this type of measurement is that the Current-Resistance plot is linear, which means the error will always be the same no matter what resistance you are measuring.

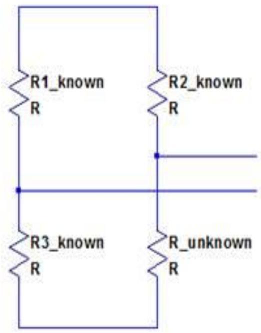


Figure 7 - Wheatstone Bridge Example

The equation that relates the resistances together can be seen below.

$$R_{unknown} = \frac{\frac{R_3}{R_{1+R_3}} + \frac{V_m}{V_s}}{1 - (\frac{R_3}{R_{1+R_3}} + \frac{V_m}{V_s})} R_2$$

Equation 1

In the formula, Vm is the voltage being measured and Vs is the supplied voltage. To measure the voltage difference, a difference amplifier can be used. The formula was determined by analyzing the current in each branch, for the purpose of finding the voltage in between both R1, R3 and in between R2, R_unknown. Taking the difference of the voltage in the right branch and in the left branch relates the output voltage to the input voltage and all 4 of the resistances. After some simplification, the above formula is discovered. [17]

[DB]

3.1.2.2.2 Difference Amplifier

A difference amplifier circuit takes the voltage potential at two points, subtracts them, applies a gain and then outputs that voltage. See Figure 8 - Difference Amplifier. This circuit essentially acts as a voltmeter.

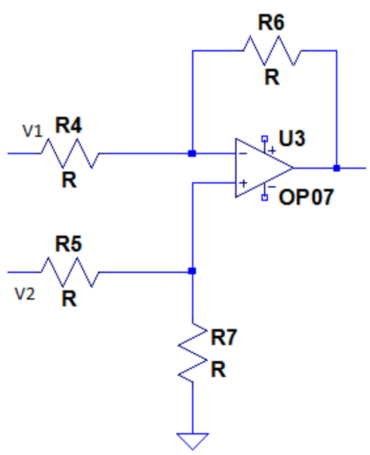


Figure 8 - Difference Amplifier

The output of the circuit is described by the equation

$$V_{OUT} = \frac{R_6}{R_4}(V_2 - V_1) \tag{22}$$

The circuit was analyzed using superposition; first, the currents through resistors R4, R5, and R6 were found using Ohm's law and nodal analysis. Then V2 was set to 0V and the output voltage was found. Then V1 was set to 0V and an expression for the output voltage was found. By the superposition method, both output voltages were added together to form the final formula. After some simplification, the final expression (Wquation 22) is derived [18].

[DB]

3.1.2.2.3 Loading of the Wheatstone Bridge

For a wheatstone bridge to function properly, there must be no loading where the voltage difference is being measured. This means the input impedance of the difference amplifier must be much higher than the output impedance of the wheatstone bridge. To ensure no loading occurs, two buffer circuits can be placed at the inputs of the difference amplifier as shown in Figure 9 - Op-Amp Voltage Follower (Buffer).

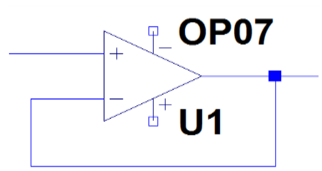


Figure 9 - Op-Amp Voltage Follower (Buffer)

[DB]

3.1.2.2.4 Final Potential Circuit

Connecting the bridge to the difference amplifier through two buffer circuits yields a potential soil resistance measuring circuit, see Figure 10 - Wheatstone Bridge Circuit Followed by a Different Amplifier (Sensor). The input voltage and values of resistors have not yet been determined, but so long as R1=R2=R3 and the gain R6/R4 provided by the amplifier is taken into account, the circuits operation should be satisfactory. The final potential circuit design is shown below, where Rx is the resistance of the soil.

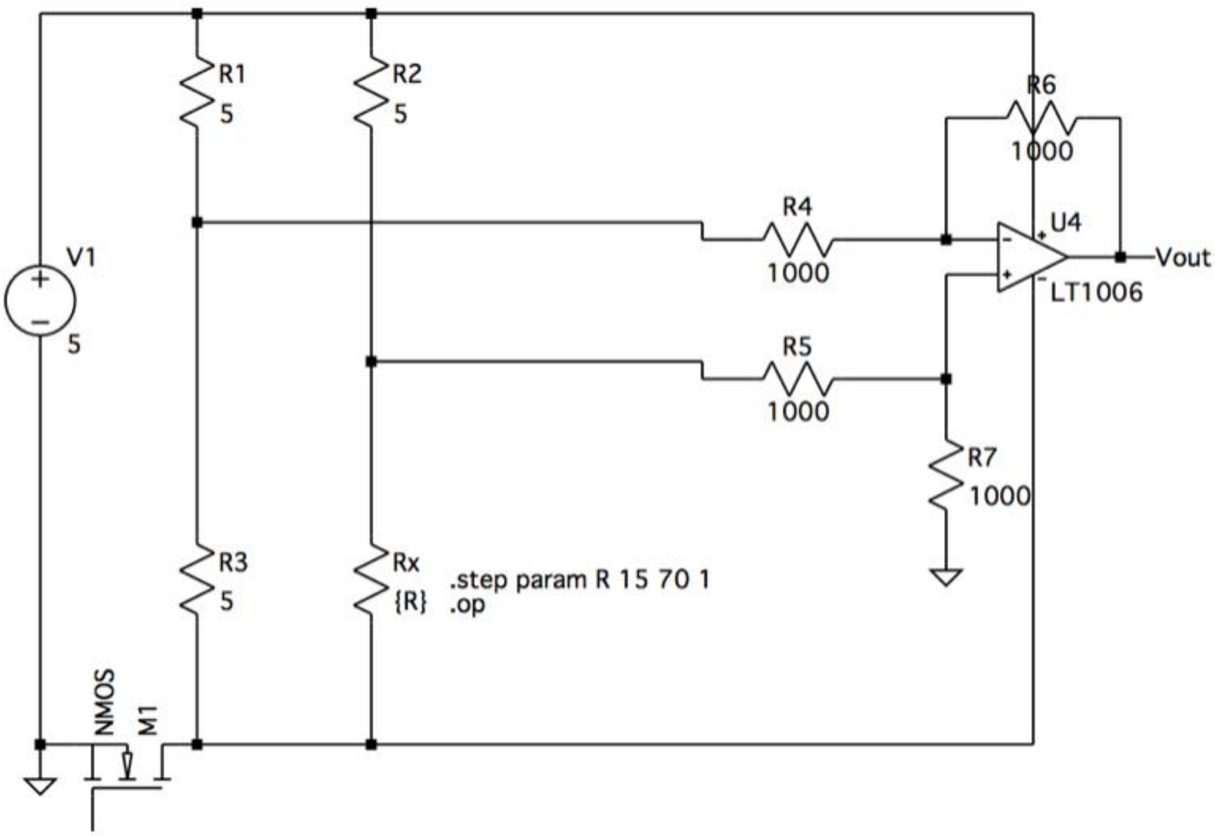


Figure 10 - Wheatstone Bridge Circuit Followed by a Different Amplifier (Sensor)

The equation that relates the resistance at Rx to the output voltage was found by substituting the difference amplifier output voltage into the wheatstone bridge formula. The final formula that dictates what the measured resistance is, is shown in Equation 2.

$$R_X = \frac{\frac{R_3}{R_{1+R_3}} + \frac{\frac{R_6}{R_4}(V_{right} - V_{left})}{V_{source}}}{1 - (\frac{R_3}{R_{1+R_3}}) + \frac{\frac{R_6}{R_4}(V_{right} - V_{left})}{V_{source}}} R_2$$

Equation 2

However, the above equation does not take into account the resistance of each probe. The equation can be modified such that $Rx=R_{probe1}+R_{probe2}+R_{soil}$.

[DB]

3.1.2.2.5 Soil Resistance Calculations

It will be necessary to measure the resistance of the soil that the sensor will be expected to monitor beforehand. This is because the resistivity of soil depends on the soil's chemistry, which differs depending on the region. A range of soil resistances can be determined by gathering a sample of soil, drying it out and simply measuring the resistance with an ohmmeter. A controlled amount of water should then be added and the resistance measured; up to the point where the soil is completely saturated. This data will allow for the creation of a moisture-to-resistance model and will greatly aid in the design and operation of the sensor. The model should be able to refer a volumetric water percentage given a measured resistance. To have a baseline, measurements taken by Md. Abdus SALAM, Quazi Mehbubar RAHMAN, Swee Peng ANG, and Fushuan WEN using the Wenner four pole method at a probe distance of 30cm yielded a soil resistance range of ~15-70 ohms. Where 15Ω represents wet soil and 70Ω is dry soil. It was not specified how much water was added to make the soil "wet" or how little for "dry" soil. This is another reason to do measurement experiments. [16]

[DB]

3.1.2.2.6 Simulation

A simulation of the sensor circuitry yielded the following graph. By measuring the output while varying the resistor Rx in a range consistent with researched soil resistance, an expected resistance to output voltage graph was formed.

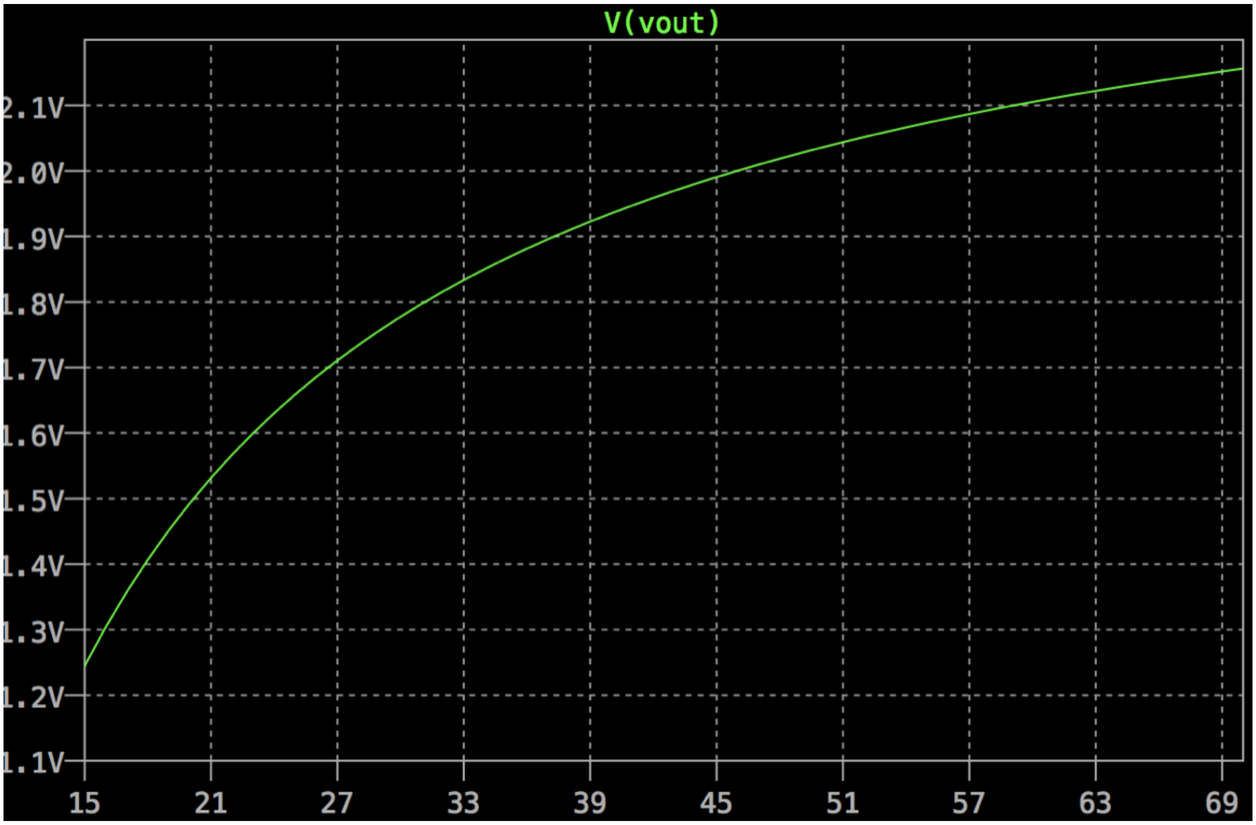


Figure 11 - Simulation of Sensor

3.1.2.3 Structural Design

A simple structural design is shown below:

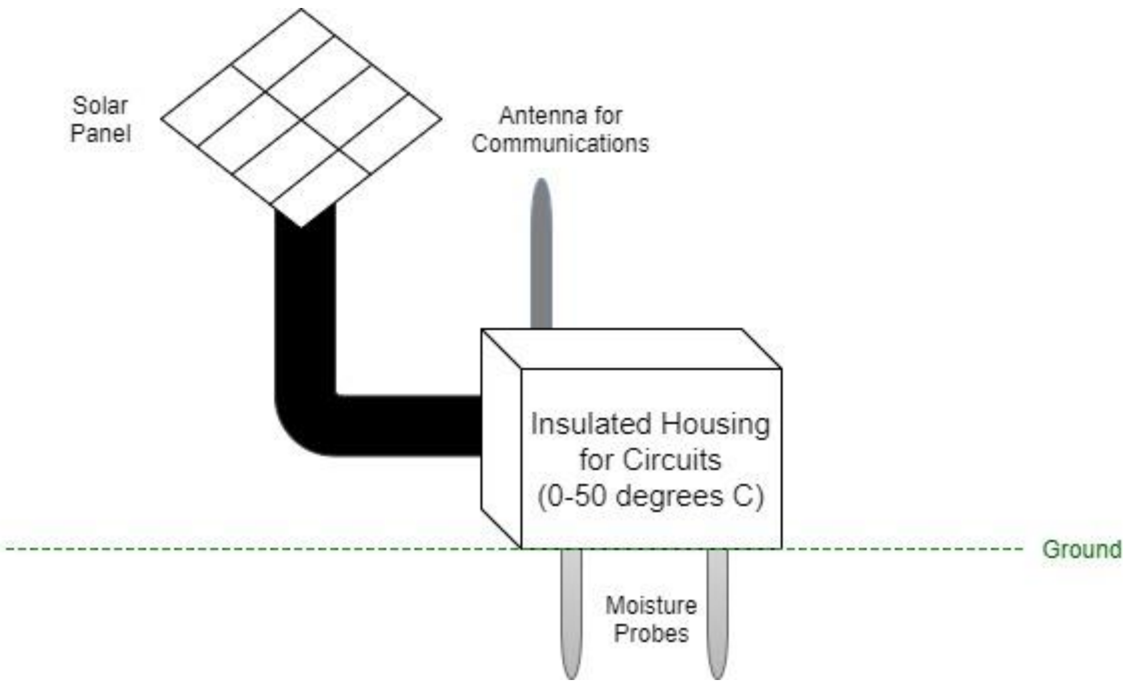


Figure 12 - Structural Design

The solar panel will be elevated above the ground. It will have an adjustable angle for the user to optimize. The housing will be insulated so that the circuitry will not go below 0 degrees Celsius or above 50 degrees Celsius. The components, especially the Lithium Ion Rechargeable Battery, will overheat and/or not work outside of this temperature range. In addition, the housing will be waterproofed so that the circuitry will not be damaged. Keeping that in mind, the opening for the solar panel connection to the housing should be on the side of the box to try to diminish any amount of water leakage. Lastly, there will be an antenna protruding from the top of the insulated housing for communication purposes.

[JM]

3.2 Software Design

3.2.1 Communications

A possible solution would be to use an End-Node module which will send the moisture data from the sensor to another End-Node module. The moisture data would then be accessible by the user through a user interface on a mobile device computer. The LoraWan technology was considered because it has a wide communication range (up to 10 km) and low power-consumption (Rx current:13.5mA and Sleep Mode: 22uA). The Lora device will be in sleep mode automatically if there is no input for more than 30 seconds. However, there are some limitations for using those in this project due to the high price. There are some alternative options for using Lora technology, but with the limited LPWAN (Low Power Wide Area Networks) technology as a trade-off. Another new option is Xbee. The full application of LoraWan needs three devices; Class A end-node, Class B gateway and LoraWan Server. Class B gateway has more receiving slots to listen to multiple nodes and IP forwarding protocol to the server. The communication between the node use Lora Radio Transceiver with 915 MHz band and FSK (Frequency-Shift Keying) modulation. Instead of using multiple listener of Gateway, the same end-node can also be communicated with each other. However, the limitation is that it only communicates with one node and it cannot connect to the LPWAN directly since it has no decrypt mode and IP forwarding protocol. But it can still connect to the LPWAN through the PC and act as Gateway.

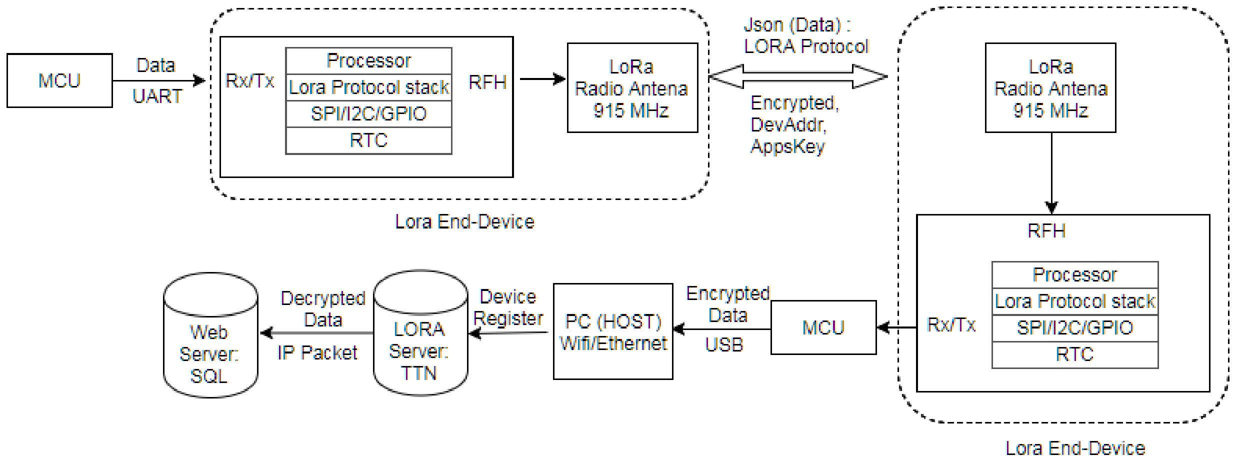


Figure 13 - Node-to-Node (Class A) Communication without Gateway

[DB, SJL]

3.2.2 General Design

The software of the moisture sensor will consist of four main parts: microcontroller, wireless module, server and user-interface. The microcontroller has ADC (convert analog to digital data), wireless module (send data to server using GSM module), server side (data stored in MySql database) and User-Interface (data log, retrieving data).

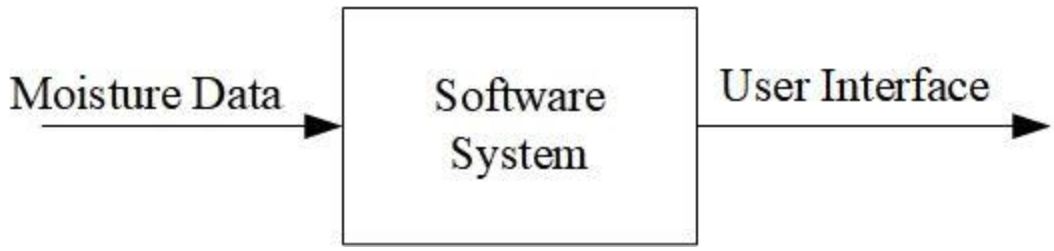


Figure 14 - Software Block Diagram Level Zero

Module	Software System
Designer	Jaclyn Miller and Seung Jun Lee
Inputs	Moisture Data
Outputs	User Interface
Description	Take the input, the moisture data, from the moisture sensor, process it, and display it to the user.

Table 10 - Software Functional Requirement Level Zero - Software System

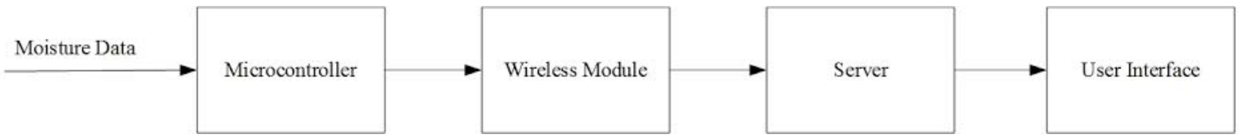


Figure 15 - Software Block Diagram Level One

Module	Microcontroller
Designer	Jaclyn Miller and Seung Jun Lee
Inputs	Moisture Data
Outputs	Wireless Module
Functionality	Process data. This will be done by keeping track of the time such that every 12 or 24 hours the microcontroller will activate the sensor to take a moisture level reading. It will then process this information and transfer it to the Wireless Module.

Table 11 - Functional Requirement Level One - Microcontroller

Module	Wireless Module
Designer	Seung Jun Lee
Inputs	Microcontroller Moisture Level Data
Outputs	Server
Functionality	The wireless module will transmit the information or the moisture level from the sensor that has been processed by the microcontroller to the server. This may be done by wi-fi, Zibee, or any other similar technology.

Table 12 - Functional Requirement Level One - Wireless Module

Module	Server
Designer	Seung Jun Lee
Inputs	Wireless
Outputs	User Interface
Functionality	The server will allow the transfer of data from the wireless module to the user interface.

Table 13 - Functional Requirement Level One - Server

Module	User Interface
Designer	Seung Jun Lee
Inputs	Analog Data Input
Outputs	Graph in Web/Mobile Application
Functionality	The user interface will allow the user to visualize the data and analyze it.

Table 14 - Functional Requirement Level One - User Interface

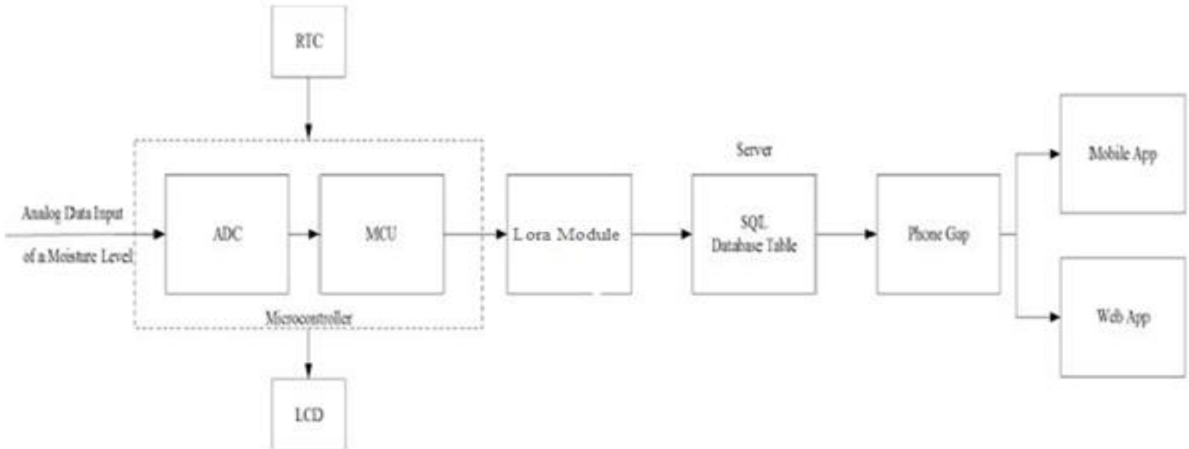


Figure 16 - Software Block Diagram Level Two

Module	Microcontroller
Designer	Seung Jun Lee
Inputs	ADC Analog Input
Outputs	Digital Data
Functionality	The digital data input as Microcontroller and transmit to SQL Data server by GSM Module. However, we decided not to use GSM Module due to their power consumption and price. We will update the block diagram once final design is accepted.

Table 15 - Functional Requirement Level Two - Microcontroller

Module	Lora Module
Designer	Seung Jun Lee
Inputs	Digital Data
Outputs	Data Packet with Notification System
Functionality	The digital data input as Microcontroller and transmit to SQL Data server by Lora Module.

Table 16 - Functional Requirement Level Two - GSM Module

Module	Server Side
Designer	Seung Jun Lee
Inputs	Data will be stored in the SQL Database
Outputs	Display data in Mobile phone
Functionality	The web and mobile application is developed by the PhoneGap platform, which will create mobile app with web source code. The data in the database is retrieved by the sql query and display on the mobile app and web application.

Table 17 - Functional Requirement Level Two - Microcontroller

3.2.3 Microcontroller Requirements

As defined by the hardware team, the following pins are needed in the Microcontroller:

1 PWM output for the moisture level sensor

1 ADC inputs for the moisture level sensor

1 possible clock reference for the power optimization

In addition to this, other basic pins are needed for powering the microcontroller or for the software. These pins include the following:

Vcc, this is normally 5v or 3.3v

GND

An Internal Timer, this is needed for the software and will be described later

Lastly, the microcontroller must use as little power as possible. To do this, the microcontroller will have three modes of operation:

Idle Mode

Turn On/Off Mode

On Mode

[JM]

3.2.3.1 PWM Outputs

PWM or Pulse Width Modulation is a term that explains a digital signal or square wave. PWM can be used to control a voltage output to to dictate a component's on and off time. In this system's case, PWM will be used to control five switches, four of which will be in the power optimization section and the last will be used in the moisture sensor. The PWM is defined by the duty ratio, D. The duty ratio is defined by the following:

$$D = \frac{T_{ON}}{T}$$

Equation 3

Where T_{ON} is the time in which the signal is on and where T is the period. Thus, D will range from 0 to 1. The figure below, Figure 17 - PWM Graph, from Professor Forrest Sheng Bao's Embedded Systems Interfacing lecture notes, helps explain the PWM signal. The frequency at which the signal repeats will also be an important factor in the switching process. For example, the switch going to the sensor will have a very low duty ratio and frequency. The frequency will be very low because the sensor does not need to be on all day long. The duty ratio will also be very small, closer to 0 than 1, because the sensor only needs to be on for a short amount of time each time it takes a measurement. The PWM for the switches in the power optimization area will most likely be at a higher duty ratio and frequency. They may also be dynamic or dependent on other factors such as the modes of operation of the power supply.

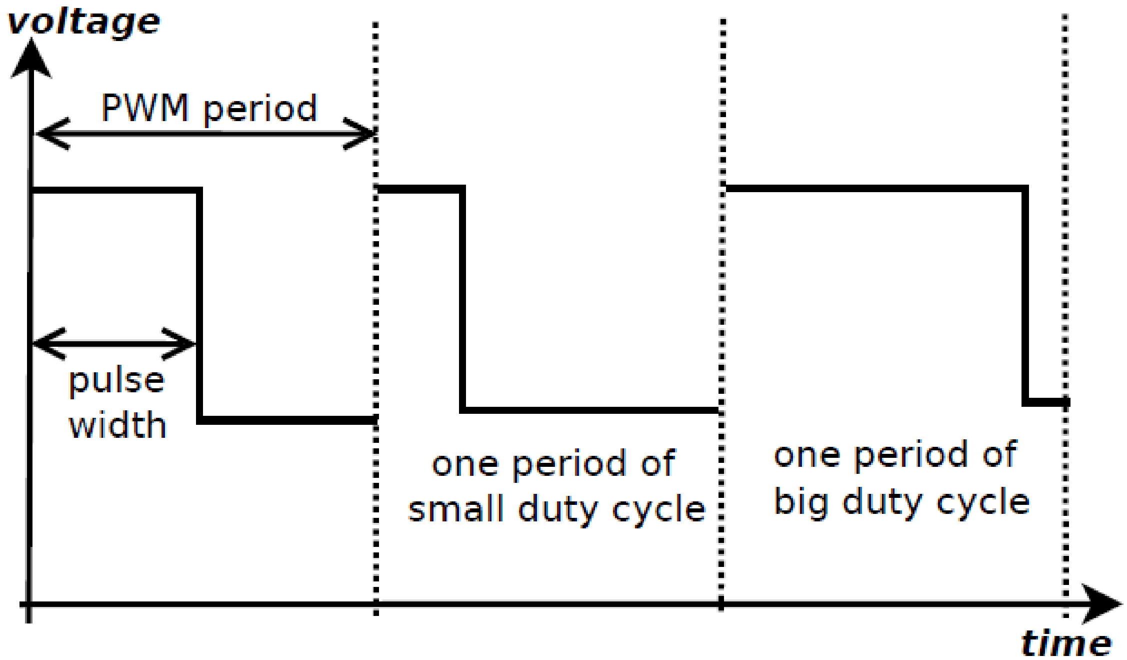


Figure 17 - PWM Graph

[JM]

3.2.3.2 ADC Inputs

ADC, or Analog to Digital Converter, is used to convert an analog signal to a digital signal so that the microcontroller can process the signal's information. Converting the signal is completed by periodically sampling the signal voltages. For each of the voltages a finite amount of binary bits are used to represent the voltage value. After this process has occurred the signal has been digitalized.

Two of the ADC inputs will be used in the power optimization part of the system and the other will be used in the sensor. The two in two power optimization will be used for the voltage monitors. It will let the microcontroller know the status of the solar panel and the battery. The other ADC input will be used in the sensor. In the sensor, the moisture of the soil will be converted into a voltage level and that voltage level will be ADC input. This input will then be processed by the microcontroller and be sent to the user.

[JM]

3.2.3.3 Clock Reference

The clock reference may be used for the power optimization. The clock reference will be used to control the voltage regulators. At this point in the design it is not known if this is needed. This may not be decided until the implementation process and will be discussed further at that time.

[JM]

3.2.3.4 Internal Timer

An internal timer is needed to control the switches and the active and sleep periods of the microcontroller. This timer will be running in the background non-stop even in the sleep periods. This will tell the microcontroller when to wake up. When awake, it will regulate the switches and will be able to tell the sensor to turn on. The internal timer may also be used for other uses that are not yet determined.

[JM]

3.2.3.5 Modes of Operation

As mentioned above, there will be three modes of operation:

Idle Mode

Turn On/Off Mode

On Mode

During the idle mode very few things will be running on the microprocessor such as the timer. This mode will consume the least amount of power. The idle mode will be the mode the microcontroller is in the majority of the time. The turning on/off mode will consume more energy, however, this mode will not happen for a long period of time. The energy needed to turn on or off the microcontroller is different for each microcontroller. The on mode will use the most amount of energy and should not be on for a significant period of time. The duration of this mode will depend on the time needed to process the information from the sensor. The total energy needed will be determined by the summation of the energy needed in each mode. To help visualize this concept, see Figure 18 - Required Energy for the Microcontroller.

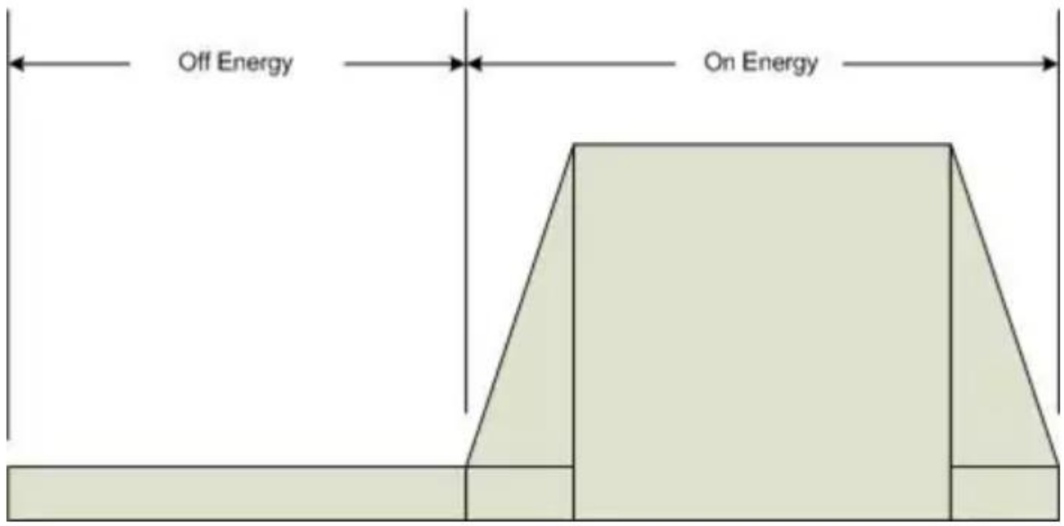


Figure 18 - Required Energy for the Microcontroller

[JM]

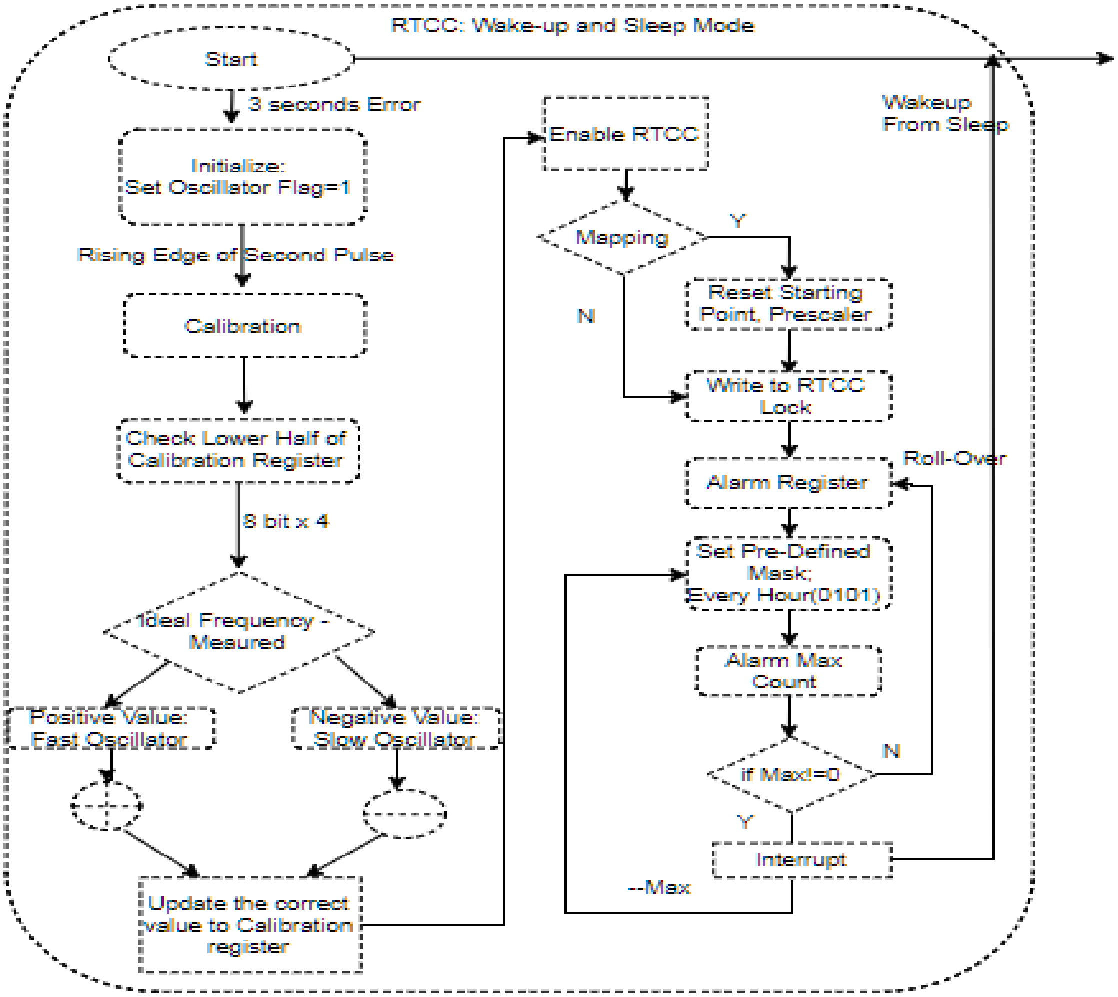
3.2.4 Microcontroller Data Flow

The input data from the moisture sensor is a very small data bit in terms of the data transfer rate on communication part (Range of 40Mb/s), which the input data is around 2 or 3 bit (of range of 10 integer). So there is not much logic to design in the microcontroller software development. However, utilizing the Interrupt service for sleep mode and UART communication will result in a better output in terms of resource management in the microcontroller. The PIC24 family microcontroller has been chosen for standards for this process.

[SJL]

3.2.4.1 RTCC Module

When the microcontroller first powers up, it initializes the ISR priority (set as 1) for RTCC alarm module. Since this implementation is only collecting and sending data during certain times of day, the whole CPU in the microcontroller and the Wireless Module (Lora) should be in sleep mode. The power consumption of the microcontroller and Lora device is very low while in sleep mode. This is within the range of 20mA*Fixed voltage for microcontroller and 30mA for Lora device. There also is a deep-sleep mode which has a range of nano ampere. The overall steps of RTCC module can be shown below. First, check the calibration for minor errors (PIC24 microcontroller has an average of 3 seconds delay every start). Then, it will mapping the pre-defined prescaler and set the wake up time based on the mask (4 bit) which is configured. It will call the interrupt service and continue executing from the last instruction of the register. It needs an external crystal oscillator (32.768kHz for PIC24 family).



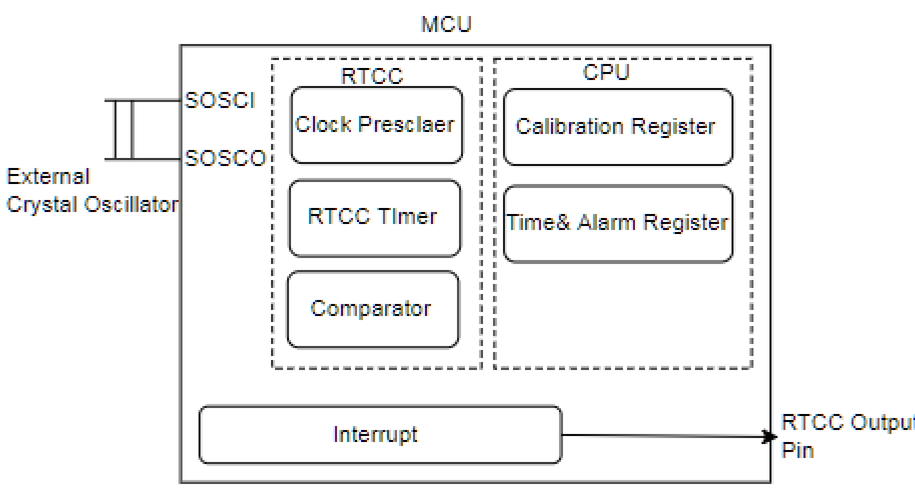


Figure 19 - Software Block Diagram Level Three: RTCC Module (Sleep Mode)

Module	RTCC-Calibration
Designer	Seung Jun Lee
Inputs	Measure current frequency
Outputs	Update correct frequency into calibration register
Functionality	This calibration runs every minute and check the 8-bit register in lower half of the calibration register. The equation for measuring the offset of current frequency is subtract from the ideal frequency, which is 32768 in this case. If the subtraction result negative, which means the current frequency is larger (faster) and vice versa. The correct value of frequency when it is faster is to subtract the value and add the value when slow. This will update the correct value of RTCC frequency.

Table 18 - Functional Requirement Level Three- RTCC-Calibration

Module	RTCC-ISR Alarm Register
Designer	Seung Jun Lee
Inputs	Pre-defined Mask : 0101 (Every Hour)
Outputs	Alarm for Interrupt service
Functionality	The mask 0101 (every hour in our project) will call the interrupt, which I set this ISR priority as 1 when initialize the microcontroller, and wake up the system from the sleep mode.

Table 19 - Functional Requirement Level Three- RTCC-ISR Alarm Register

[SJL]

3.2.5 ADC/LCD Module

The LCD module had read and write bits, enable bits and an 8 bit register for data transmission. This is configured by the MCU as shown below. The ADC module have two channels (channel 0 for an 10 bit and other channel for an 8 bit). Since there is not a large amount of data, channel 0 can be selected, which has output range of [0-256]. With the given sampling time, in this case 3* T_AD, the digital will save in the ADC buffer register and later will be sent to the UART register.

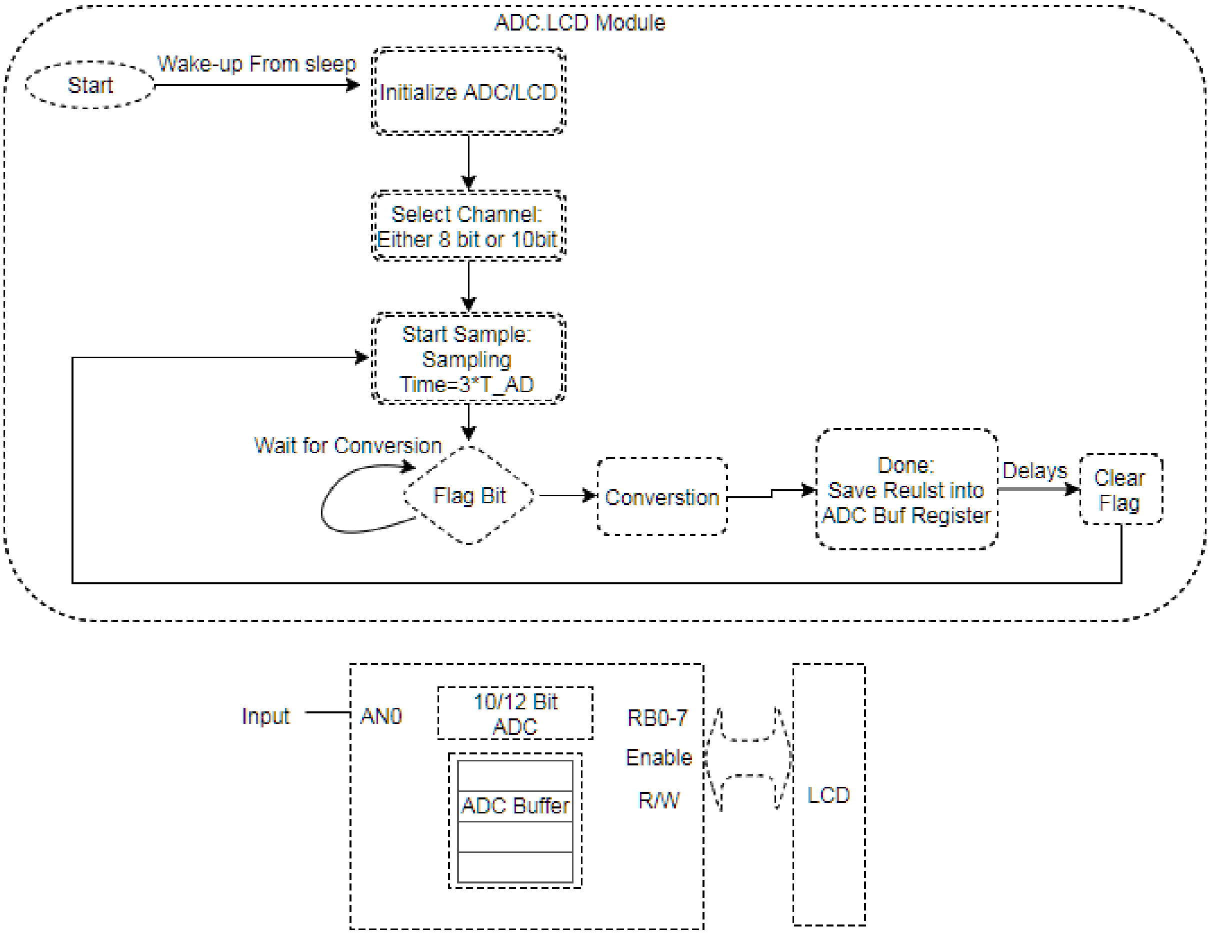


Figure 20 - Software Block Diagram Level Three: ADC/LCD Module

Module	ADC/LCD
Designer	Seung Jun Lee
Inputs	Analog Input
Outputs	Sequence of 8 bit Digital
Functionality	After the channel number and sampling time is selected, the analog input wait for flag bit for the conversion. The converted digital data will stored in ADC buffer register and shift it to the UART register.

Table 20 - Functional Requirement Level Three-ADC/LCD Module

[SJL]

3.2.5.1 ISR UART Module

This UART module is interrupt service driven. The data from the ADC buffer will first shift to the UART FIFO buffer. The interrupt will be called every time the tail of the buffer is not the same as the head (which indicates the buffer is not full). The tail will increment when the interrupt is called and it will keep track of the buffer space. The tail and head is a pointer and the data will be sent to the Tx of UART if they are same (full). This will make the chain process of data register between FIFO buffer and UART register, without waiting for the flag to be set free like the ADC part.

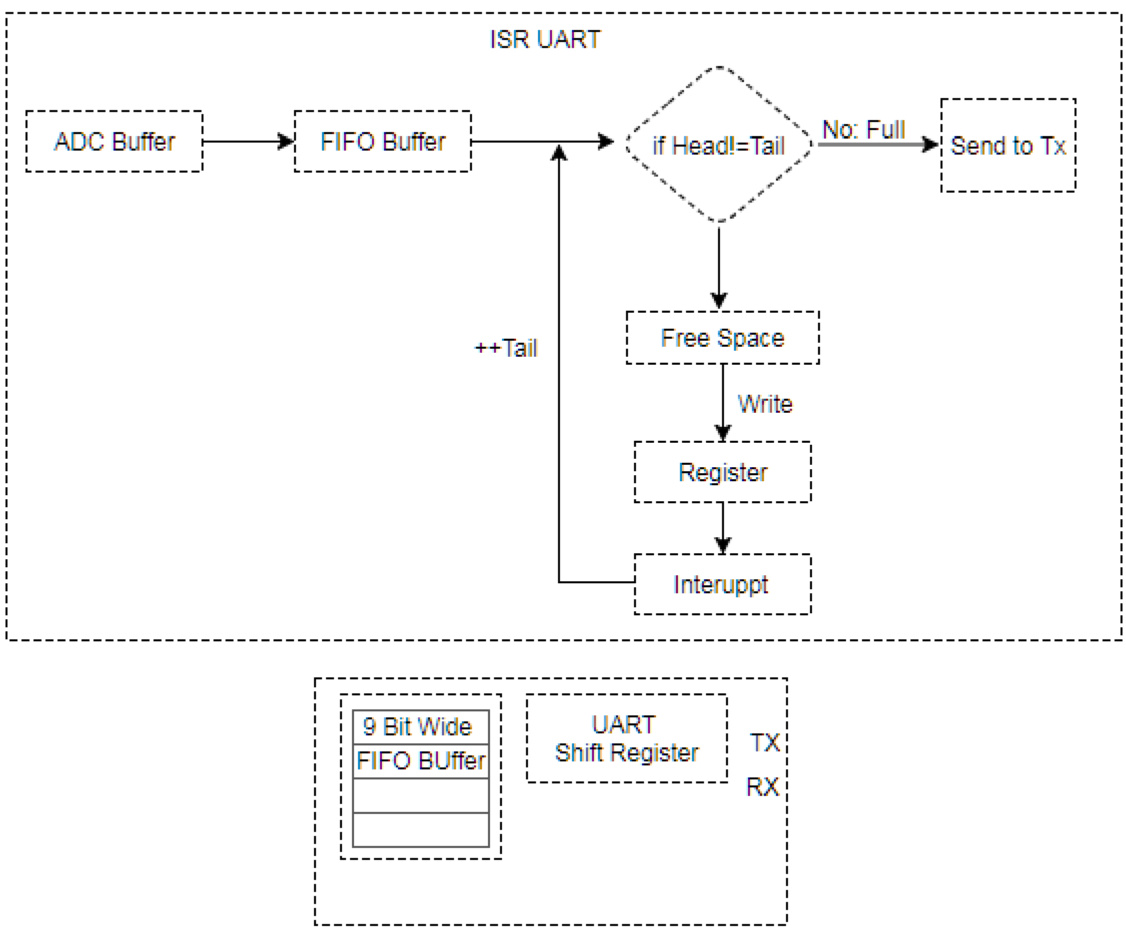


Figure 21 - Software Block Diagram Level Three: ISR UART Module

Module	ISR UART Module
Designer	Seung Jun Lee
Inputs	Digital Data from the ADC buffer
Outputs	Digital Data send to Tx
Functionality	The FIFO buffer is 9 bit wide and it holds the data before move to the shift register. Without ISR service, the data should be hold until all the data in shift register is sent to the Tx. With ISR, as soon as the one bit of data is send to the Tx, the next bit will be sent by the FIFO buffer.

Table 21 - Functional Requirement Level Three-ISR UART Module

[SJL]

3.2.6 Building Wireless Network (Lora)

Possible Option 1: Node-to-Node (Class A) Communication without Gateway

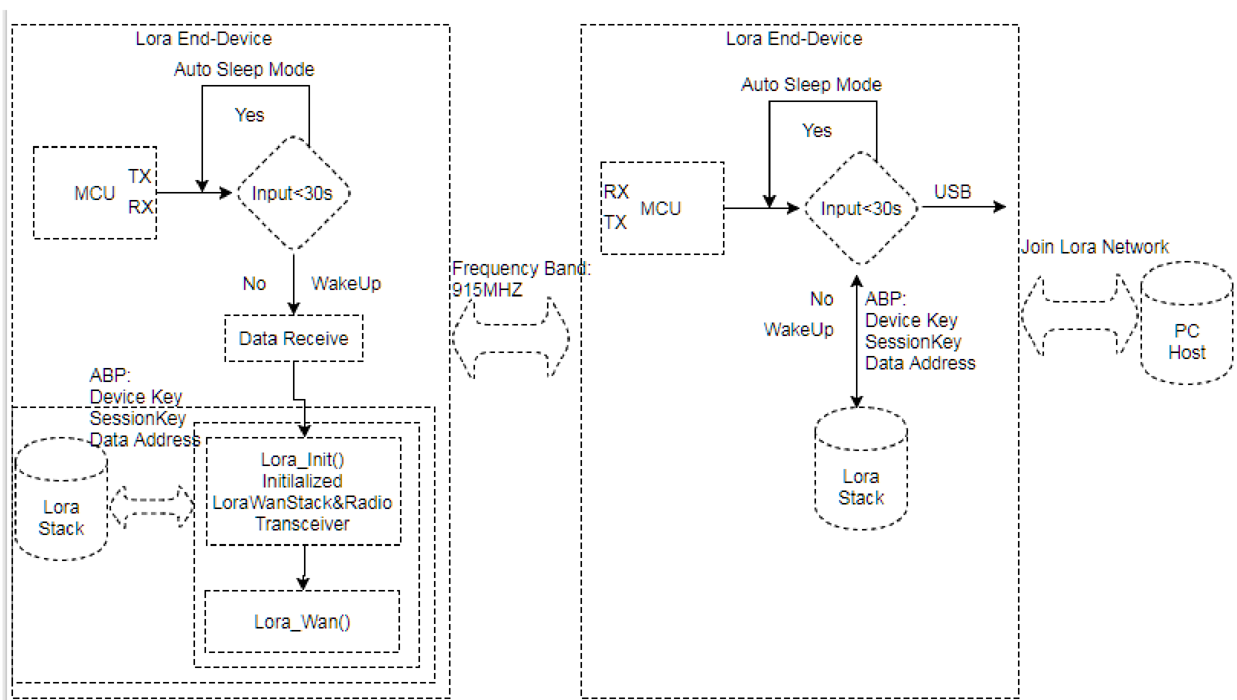


Figure 22 - Software Block Diagram Level Three: Lora Network

Module	Lora End-Device
Designer	Seung Jun Lee
Inputs	Digital Data UART (MCU)
Outputs	Digital Data send to another Lora End-Device using Transceiver
Functionality	The device will wakeup if there is any external input or data coming from the MCU. Then the end-device will call the Lora_init() and Lora_wan() function to make the data protocol with data received from the UART, its device key, data address and session key. This will protocol will be encrypted and transfer to another end-device connected to the host PC.

Table 22 - Functional Requirement Level Three-Lora End-Device

Module	Lora End-Device (Host Side)
Designer	Seung Jun Lee
Inputs	Encrypted Data Protocol from Lora End-Device
Outputs	Data Protocol send to PC via USB
Functionality	Since this Host-side is not a gateway, it has no decryption and IP protocol forwarding, so it needs to be connected to PC to make connection to the Lora Network to see the data. After the data protocol is received, the host-side end devices will register its device id to the network via USB to host computer and send the data packet. The data will be availalbe in the any Lora hosting web site.

Table 23 - Functional Requirement Level Three-Lora End-Device (Host Side)

[SJL]

3.2.7 User Interface (Web/Mobile Application)

The front end tool, which is a client-side interface, will be developed in HTML and java-scripts. The back end tool, which is a server-side interface with a database, will be developed in PHP and MYSQL. The front-end page scripts are “login.htm”, “register.htm”, “notify_set.htm” and “data.htm.” The back end server scripts are “register.php” and “member_req.php”. Between the client-side and the server-side, “action.php” and “dbcon.php” classify the request from the users and direct to the right function of the server-side scripts. The “dbcon.php” script sends MYSQL query to the server to make a connection, and input, retrieve and update the data. The system design is categorized into three modules; a user management, a notification management and a real-time graph. The user management enables new users to have a registration into the database with an ID, encrypted password and contact information (Email or Phone). These are saved into the database by calling the server side script, “register.php.” The existing users are distinguished by their own keyword in the database. The existing users can login through “login.htm‟ and register each water sensor on “notify_add_mod.htm.” This also enables users to setup the notification by inputting the maximum or minimum values desired of the collected moisture level. Users can add, delete or update the notification setup for each sensor on “notification.htm.‟ These requests of user management and sensor management are sent to “mem_req.php‟ on the server side through “action.php.” The main feature of the system is monitoring the real-time data by graph. The graph on “data.htm” enables users to input a preference of dates (“today”, “yesterday”, “last seven days”, “this month” and “last month”) to see a real-time data read by the sensor. The user id and device id sensor and the range of selected date from “data.html” are used to retrieve corresponding data from the database via “member_req.php.” For the daily-based graph, the data is retrieved and displayed in an hour scale by calculating an average value of data in each hour. For the monthly-based graph, the data is retrieved and displayed in a day scale by sorting the maximum and the minimum value in each day. These processes can be achieved by the server side of “member_req.php‟ with sending SQL query to the database.

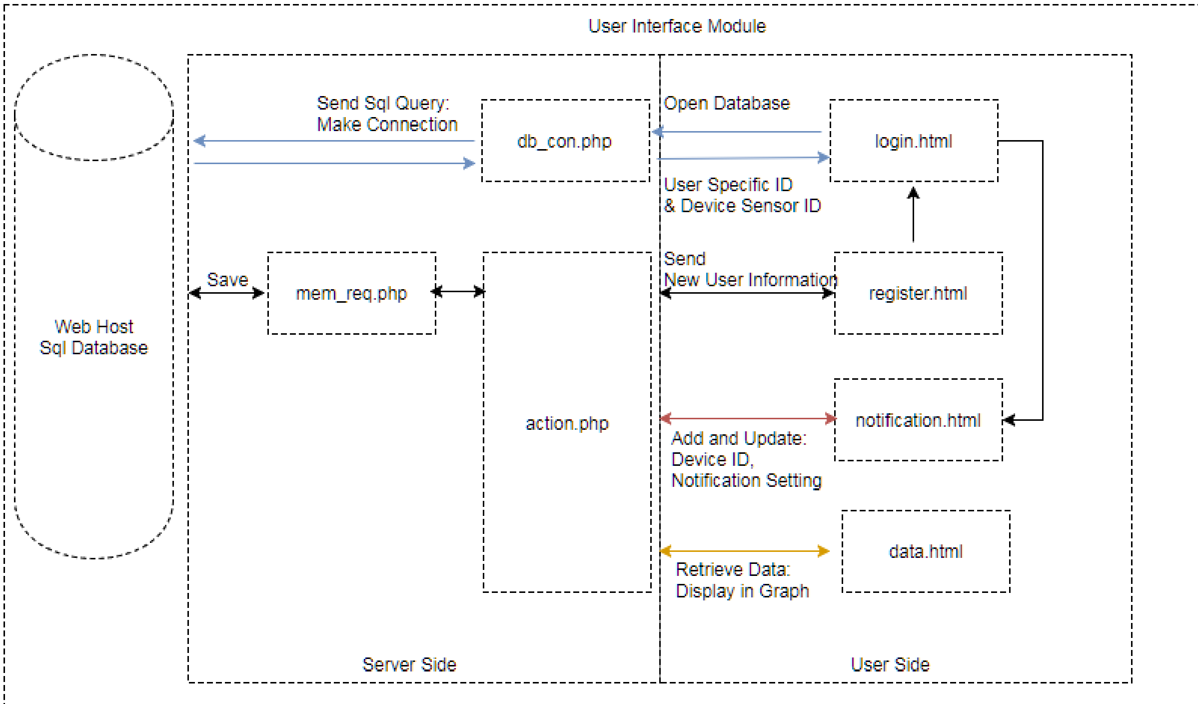


Figure 23 - Software Block Diagram Level Three: User-Interface (Web/Mobile Application)

[SJL]

4.0 Accepted Implemented Design

4.1 Power Design Implementation

The implemented power design changed multiple times within the semester due to various restrictions. At the beginning of the semester it was determined that there would be a buck-boost and a buck circuit in the design. The buck-boost converter would take a voltage of 9V to 4V and either buck or boost it to the desired 5V, shown in Figure 24 and Figure 25. Similarly, the buck converter would take the 9V to 4V input and always buck it to 3.3V, shown in Figure 26.

For the buck-boost circuit the following equations and table of values were calculated.

$$V_{IN} = -V_o\left(\frac{1-D}{D}\right), when\ V_{IN} = 4V\ D = 0.556\ and\ when\ V_{IN} = 9V\ D = 0.357$$

For an inductor ripple current of 10%:

$$\frac{\Delta I_L}{I_L} = 0.1$$

$$I_L = \frac{V_{IN} D}{R(1-D)^2}$$

$$L_{CLOSED} = \frac{V_{IN} DT}{\Delta I_L}$$

$$C = \frac{D}{Rf \frac{\Delta V_o}{V_o}}$$

Assume f=100kHz

Parameters	**I_L**	**Δ I_L**	**L_{CLOSED}**	**C**
V_{IN}=4V, R_L=15Ω	0.752A	0.0752	296mH	3.7µF
V_{IN}=4V, R_L=70Ω	0.161A	0.0161	1.38mH	794nF
V_{IN}=9V, R_L=15Ω	0.001397A	0.0001397	230mH	2.38µF
V_{IN}=9V, R_L=70Ω	0.111	0.0111	2.89mH	510nF

Table 24 - Calculated Values for Buck-Boost Converter

The chosen inductance value must be large enough to store the required amount of energy to ensure continue current. The circuit must stay in continuous current mode. Thus, the scenario 3 must be used for the inductance because it has the smallest current through the inductor and the largest value of the inductor. L_c was therefore calculated to be 230mH. For the capacitor, scenario 1 must be used because it has the largest capacitance. C= was therefore calculated to be 3.7μF.

For the buck circuit the following equations and table of values were calculated.

$$D = \frac{V_o}{V_{IN}}, when\ V_{IN} = 4V\ D = 0.825\ and\ when\ V_{IN} = 9V\ D = 0.367$$

$$L_{MIN} = \frac{(1-D)R}{2f}$$

$$C = \frac{1-D}{8L(\Delta V_o/V_o)f^2}$$

Parameters	L_{MIN}	C
V_{IN}=4V, R_L=15Ω	13mH	98.5pF
V_{IN}=4V, R_L=70Ω	61mH	98.5pF
V_{IN}=9V, R_L=15Ω	47mH	356pF
V_{IN}=9V, R_L=70Ω	222mH	356pF

Table 25 - Calculated Values for Buck Converter

Similarly, to the buck-boost converter, the largest inductor and capacitor values are chosen to keep the converter in continuous current mode, thus, L=222mH and C=356pF.

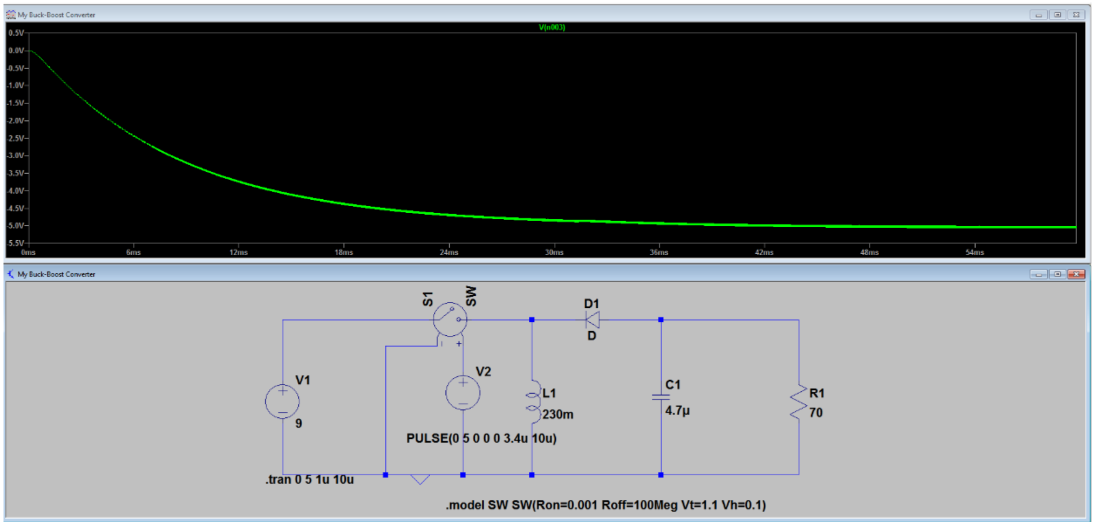


Figure 24 - Buck-Boost Converter Buck Mode Circuit and Simulation

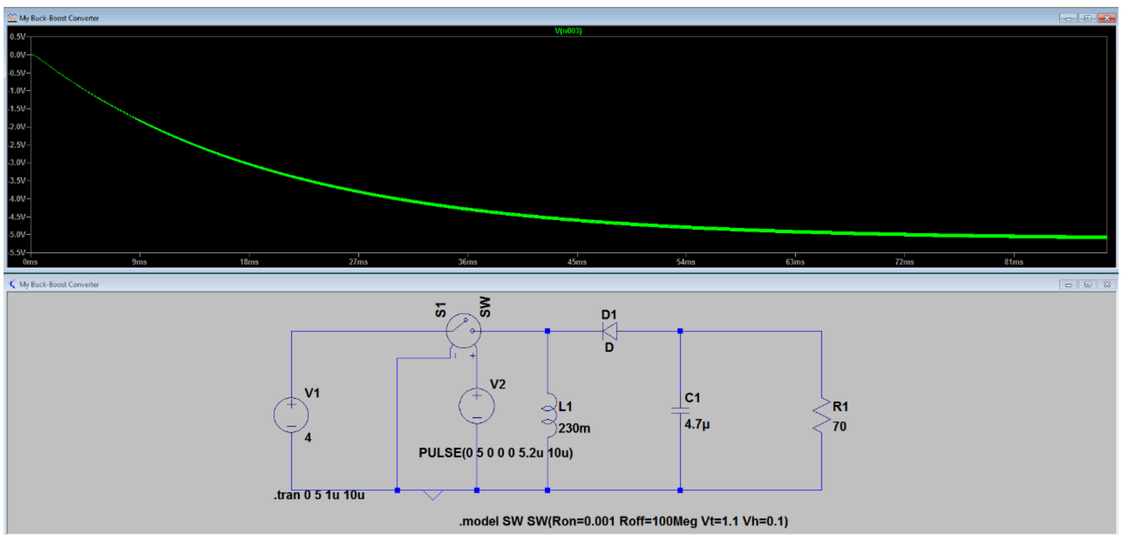


Figure 25 - Buck-Boost Converter Boost Mode Circuit and Simulation

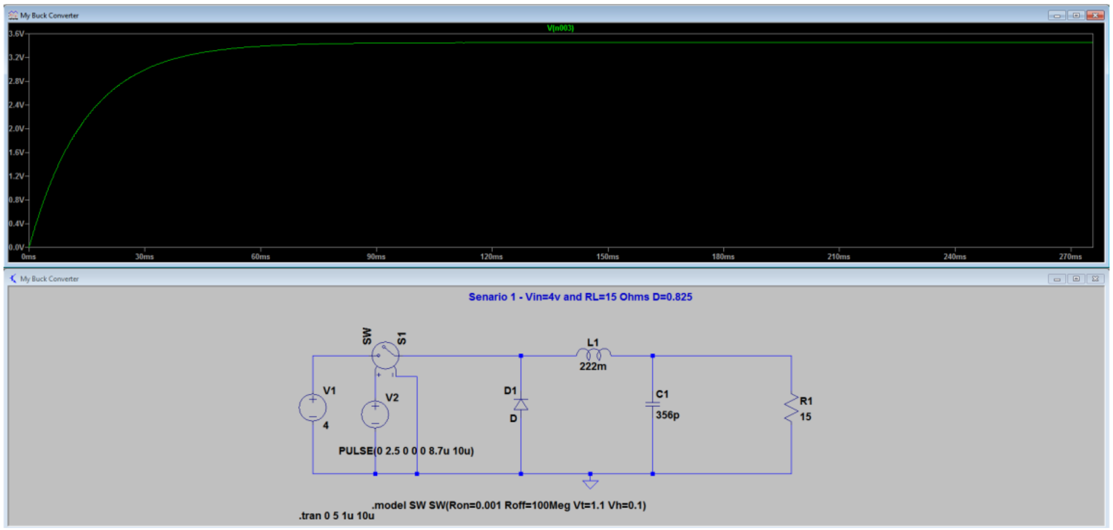


Figure 26 - Buck Converter Circuit and Simulation

This would directly follow the solar panel. The solar panel could vary anywhere from 9V-4V at approximately 6W and would still be able to supply enough energy for the whole circuit. Upon further calculations, the power that the system needed was dramatically less than initially thought. There were also other restrictions from the solar panel such as the size of the solar panel impeding with crops. Another reason it was left out was the maintenance to the unit would be greatly higher. As stated in the marketing requirements, any maintenance should be minimal. Thus, the solar panel was removed from the design. Since the solar panel was removed, a 9V battery pack was used instead. The benefit of still using the converter design was that the battery could deplete all the way to 4V and still be able to properly power the system. The 5V output of the converter would then power the rest of the circuit. A PIC16F690 was used to create a PWM as the switching component. The following equations show some of the values that were used in setting up the PWM in the PIC.

$$T = (PR2 + 1)4T_{OSC} (1)$$

If T_{OSC}=8M and T=us, then PR2=19 decimal.

$$D = \frac{x}{4(PR2+1)}, thus\ x = 44.48\ decimal = 00101100\ hex$$

The PIC was able to create a PWM that would theoretically drive the Mosfet. However, the design overlooked the simple fact that the drain of the Mosfet was at a higher voltage than the gate. Because of this, the switch would never turn on. Thus, the design of both of the converters are very flawed. The switching component of the converter should have been more fully thought through, however this was a simple mistake that was overlooked. Due to the sudden halt in the converter design, the power design was once again changed.

The design was changed to use a linear regulator. Figure 27 shows the linear regulator circuit that was used.

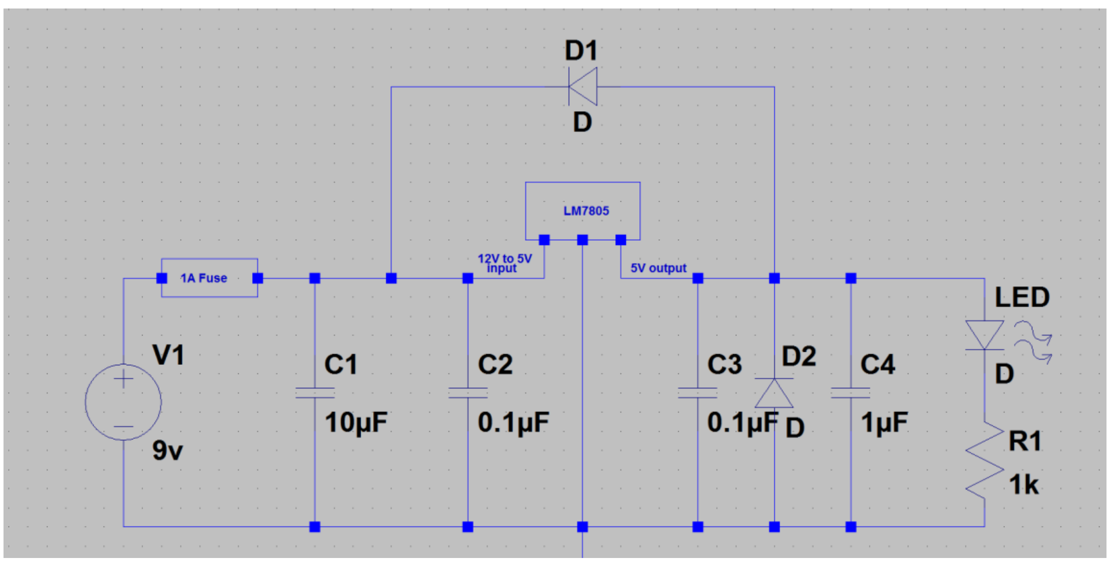


Figure 27 - Implemented Linear Regulator Design

A fuse and diode were used to help mitigate any issues with the system. The output of the circuit is 5, which then goes to the moisture sensor. The 9V or the 5V may be used to connect the LORA devise to power. Once this new design was implemented there was no more issues with the power design. Below is a figure showing the linear regulator circuit implemented on a perf board.

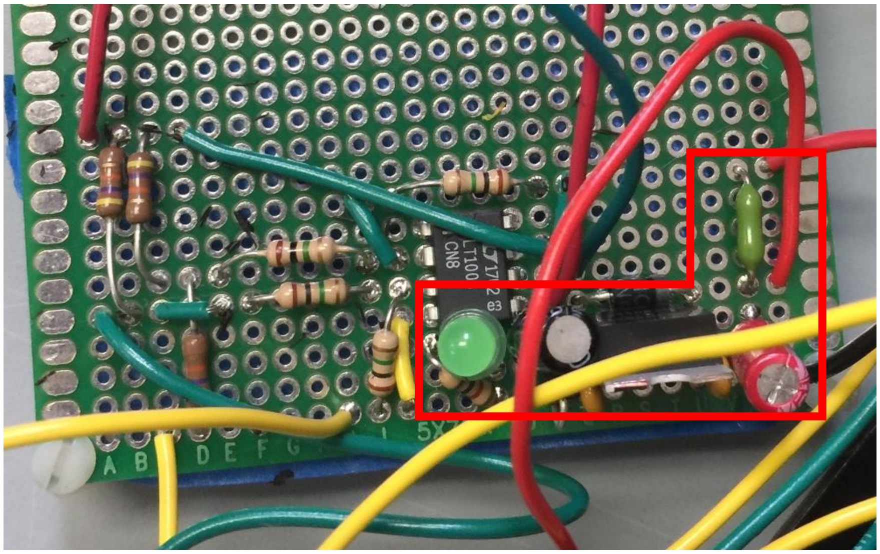


Figure 28 - Final Implemented Linear Regulator Perf Board

[JM]

4.3 Power Calculation Update

Through observation, the system takes about 30 seconds to connect to the network and send the data measured by the sensor to the gateway. The time needed for this process depends greatly on the environmental surroundings of the Lora node. Therefore, 30 seconds is only an approximation. For the power calculation, 30 seconds was assumed to be the time needed. A 9 volt energizer battery has 5.49 watt hours which is equal to 19764 watt seconds. Our system measures moisture 24 times a day which requires that the system be on for 30 seconds each measurement, which translates to 720 seconds a day. Since the system uses 122mW while on, 2mW from the sensor and 120mW from the Lora board, the system will use 87.84 watt seconds each day. 19764/87.84 equals 225 days of measuring. If a user needs the system to last longer, the number of measurements taken each day can be reduced to greatly improve battery life. For example, should the user be content with only measuring 6 times a day, the system should last for approximately 900 days on a single 9 volt battery. Thus, the system can sufficiently last an entire planting season without the aid of an external solar panel charging system.

[DB]

4.4 Sensor Design Implementation Update

Measurements of damp potting soil were taken and found to be ~50kOhms. The measurements were taken using a Fluke digital multimeter. Because of these measurements, the sensor circuitry had to be changed. That is to say, the previous design did not produce the results needed for this application. The balanced state of the Wheatstone bridge was chosen to be 47 kOhms, with the intention of producing a circuit that has a voltage difference of zero volts between the two branches. Since the moist soil has a resistance of ~50 kOhms the voltage difference will be close to zero volts. Therefore, a higher voltage level translates to a greater resistance between the two probes; which means less moisture is prevalent in the soil. The difference amplifier circuit maintained a gain of 1 and the resistors were chosen to be 1 Mega Ohm. The resistors needed to be sufficiently high enough to prevent current from leaving the Wheatstone bridge circuit. Because the resistors were greatly increased, the buffer circuits were removed, as they were no longer needed. The sensor probe design consists of two stainless steel rods cut to 4 inches in length for the purpose of reaching the optimal depth for most plants. Diameter of 1/8 of an inch with a separation distance of 2 inches. Stainless steel was chosen due to its corrosion resistance properties. Figure 29 shows the final sensor circuit that was implemented. Figure 30 shows the power that is absorbed by the sensor.

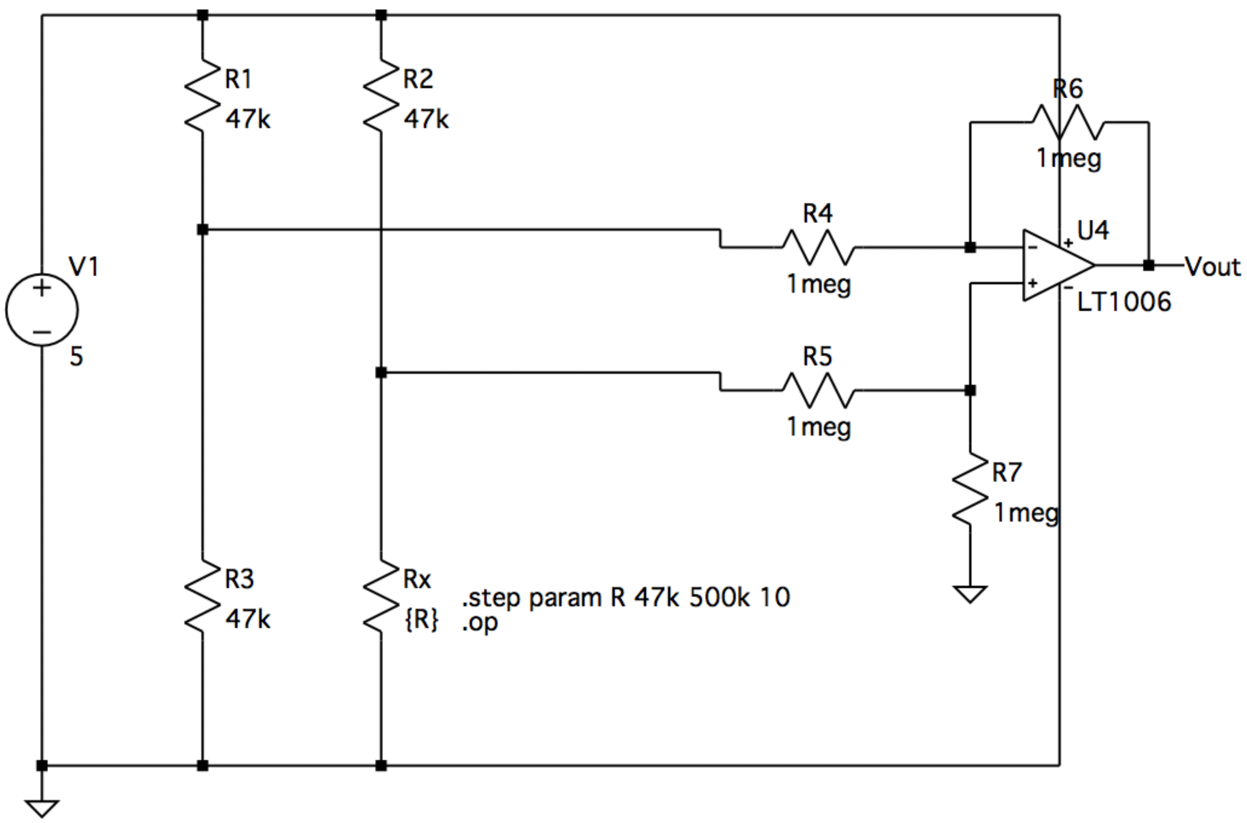


Figure 29 - Implemented Sensor Circuit

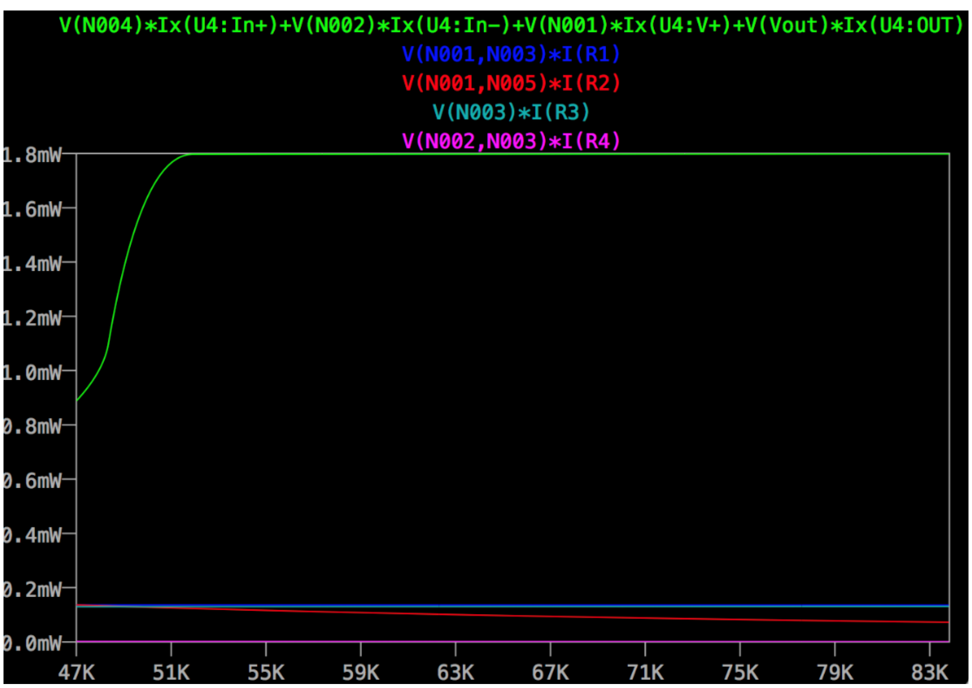


Figure 30 - Power Absorbed by the Sensor

[DB]

4.5 Structural Design Implementation Update

The only thing in the structural design that changed was the removal of the solar panel. The following figures show the updated structural design.

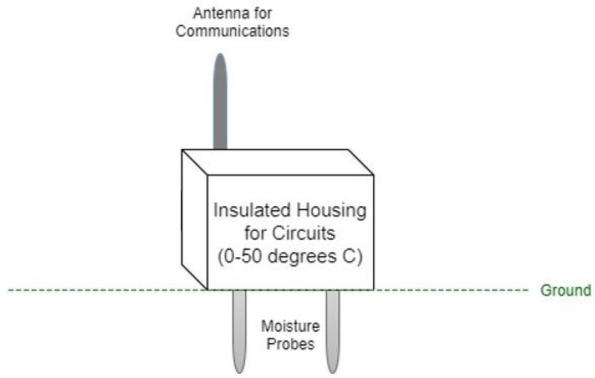


Figure 31 - Updated Structural Design

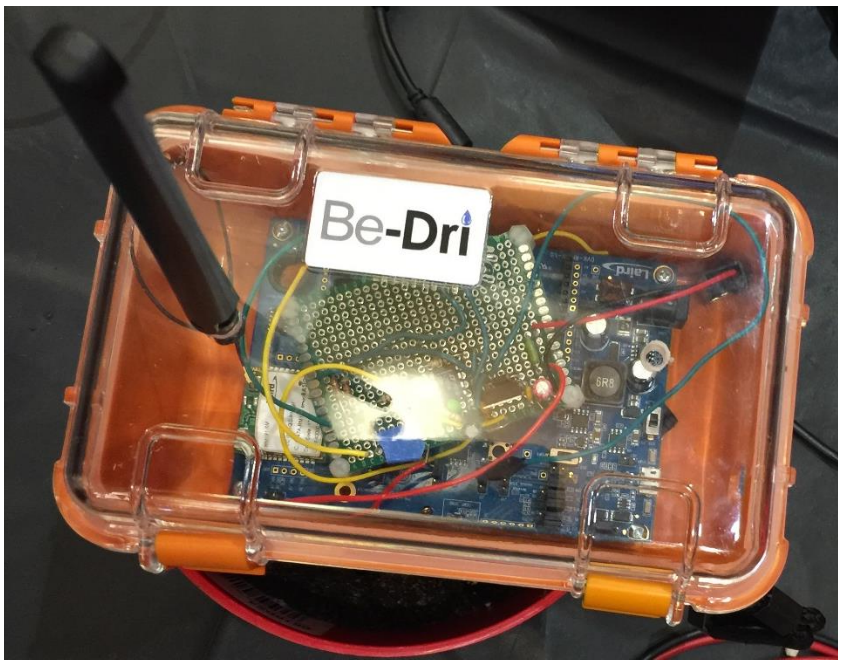


Figure 32 - Implemented Structural Design

[JM]

4.6 Software Design Implementation Update

4.6.1 The Theory of Operation

The software will consist of three main components: microcontroller (Laird RM191), wireless communication (Lora end-device and gateway) and the user-interface (web/mobile application). At first, the microcontroller component provides three operations: sleep mode, ADC and UART interface. The sleep mode closes the UART interface and Lora network joining and wake up the system by the event handler, such as timer (set by user) or GPIO input (reset button on the device) mode provides the low power mode (draws range of 20mA current). The ADC module enables sampling time (3xA/D Conversion Clock) and perform conversion of sequence of 8/10 bit of analog data into digital value and store into ADC buffer register. The UART interface is ISR driven and enables FIFO buffer (9 bit wide) to hold the data and send to Tx (uplink) of interface or receive in Rx (downlink). ISR performs keep track of buffer size and chain process of data movement between buffer and UART register. At second, the wireless component provides Lora Technology: enables 915MHz radio antenna, Frequency-Shift Keying (FSK) and Lora protocol (encrypted JSON format). The JSON data packet contains the value from UART register, Device Address, Session Key and Data Address. This component enables wireless communication between end-device and gateway. The registered Lora gateway joins the Lora Wan network to decrypt the data and forward Lora data packet to standard IP protocol (Web Server). The data format is sent to SQL server by HTTP integration. At last, the user-interface component provides two module: server side and user side. The server side performs sql query to retrieve, store and modify data on database. The user side provides GUI interaction with users; the real-time graph, remote power control and email notification system.

[SJL]

4.6.2 Wireless Communication

The LoRaWan technology is considered because it has a wide communication range (up to 10 km) and low power-consumption (Rx current:13.5mA and Sleep Mode: 22uA). The Lora device will be in sleep mode operation if there is no input more than given period decided by the users. The moisture data would then be accessible by the user through a user interface on a mobile device. The communication use Lora Radio Transceiver with 915 MHz band and FSK (Frequency-Shift Keying) modulation. Multiple node can be registered on LoraWan network and listen to the Gateway, the data will be sent in decrypt mode and using IP forwarding protocol. The communication is bi-directional (Uplink/Downlink) between the Lora end-device and web-application. The data will be stored in the SQL server by HTTP Integration. The HTTP integration sends and receives the data in JSON format (convert hexadecimal from Lora Network by decoder/encoder) to web/mobile application.

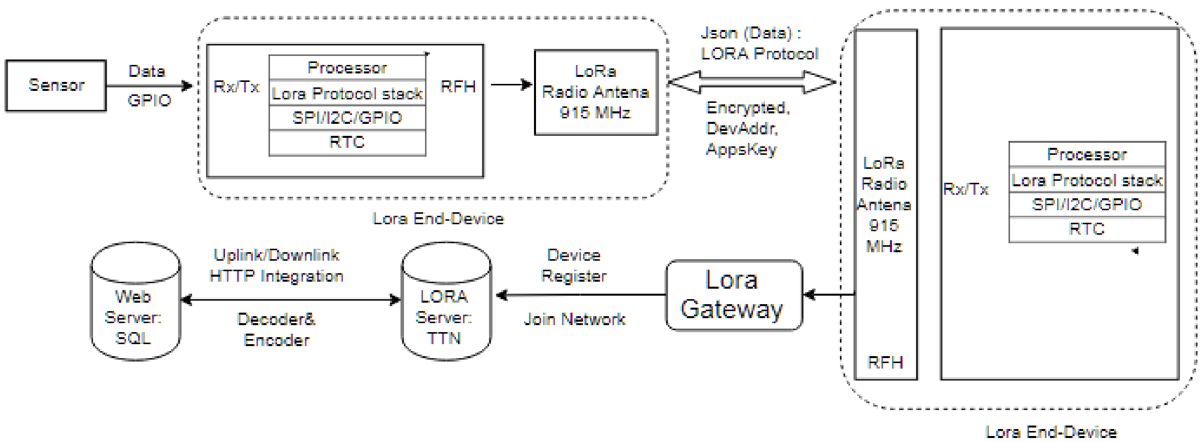


Figure 33 - Wireless Communication with Lora Network

[SJL]

4.6.3 Software Functional Table and Pseudocode

The software system will consist of three main modules: MCU (Laird RM191), wireless communication (LoraWan), and user-interface (Mobile/Web Application). The MCU provides

ADC, UART interface and sleep mode. The wireless communication (uplink/downlink) provides data transfer between MCU and mobile devices. User-Interface will provides real-time graph, remote power control, retrieve data and email notification system.

Module	Microcontroller
Inputs	Analog Moisture Level, Timer, GPIO Input
Outputs	Digital Moisture Level, Sleep Mode
Functionality	It will perform 10-bit ADC to read moisture level using GPIO and send to wireless module in hexadecimal format using UART. It will convert analog input to digital data using GPIO pins in Laird RM191board. The sleep mode operation is entered if there is downlink data from Rx handler. The system is wake up by the timer event or GPIO input.

Table 26 - Functional Table - MCU

Module	Wireless Module
Inputs	Microcontroller Moisture Level in hexadecimal
Outputs	JSON data
Functionality	The encrypted data forwarded to the Lora Gateway using radio frequency. Once gateway is joined to the Lora Network, the data is decrypted and convert into integer value as payload and send to SQL server as JSON format using HTTP integration. The downlink from the mobile devices needs encoder to convert JSON object to integer value.

Table 27 - Functional Table- Wireless Module

Module	User Interface
Inputs	Moisture Level Data
Outputs	Real-time Graph, Email Notification, Remote Power Control
Functionality	With the data sent from the sensor, the user can display the data in graph in real-time, and set the threshold to data receive alerts by email. It also sends command to the device to enter sleep mode or deep-sleep mode.

Table 28 - Functional Table- User Interface

4.6.4 Sleep Mode

The sleep mode is initiated when the Rx event is triggered by the “HandlerRxData()”. When there is any downlink data (sleep mode period) from the user devices, it will start the recurring timer, “HandlerTimer0()”. Inside the timer function, it will disable the Lora Network and UART communication during the given period of sleep mode, which is multiple of 10 second by the user input of any integer from the mobile phone. After the given period of sleeping mode, “reset()” function will wake up the system again. If the downlink data is 1 given by the user, then it will trigger the one-shot timer, “HandlerTimer1()”. This will turn-off the device by passing 0 integer value to “SystemStateSet()”. The device can be turned on by pressing GPIO button on the device.

Module	Sleep Mode
Designer	Seung Jun Lee
Inputs	Rx(Downlink) from Mobile Phone
Outputs	Sleep Mode
Functionality	The sleep mode operation is driven by the internal timer and reset (GPIO) button. The sleep mode operation period is decided by the user via downlink from mobile devices.

Table 29 - Functional Table – Sleep Mode

```
Function HandlerTimer0()
    print "\n"
    print clock
    print "sec in sleep\n";

    if clock/wake_period_sec*10 == 1 then
        PRINT "System Wakeup from Low Power Mode\n"
        TimerStart(1,5000,0)
        clock=0
        reset(0)
    endif
    clock = clock + 10
ENDFUNC shot //recurring: remain in waitevent

Function HandlerTimer1()
    uartclose()//Closes the UART
    rc = LoraMacSleepMode()
    //Put module into deep sleep
    rc = SystemStateSet(0)
    ENDFUNC 0 //one shot

    // Sleep for ms milliseconds
    sub Sleep(ms)
        uartclose()//Closes the UART
        rc = LoraMacSleepMode()
    // Start Timer 0 as a recurring timer
        TimerStart(0,ms,1)
endsub
```

```
//-------------------------------------------------------------------------
// RxData event - data has been received from the gateway
//-------------------------------------------------------------------------
FUNCTION HandlerRxData() As Integer
    dim datastr$ as string
    dim rssi$ as integer
    dim port$ as integer
    dim snr$ as integer
    dim framepending$ as integer
    dim packettype$ as integer
    rc = LORAMACRxData(datastr$,rssi$,port$,snr$,framepending$,packettype$)

    wake_period_sec=StrGetChr(datastr$,0)
    print "\ndata value: "
    print wake_period_sec
    print " received from the gateway\n"

    UartFlush(01)//clear Rx buffer
    if wake_period_sec==1 then
    //deep sleep mode
    print "\n-----------------------------------------------------------------------
    print "Deep Sleep Mode"
    TimerStart(1,5000,0) //5 sec later call deep sleep

    elseif wake_period_sec>1 then
    print "\n-----------------------------------------------------------------------
    Print "Low Power Mode\n"
    Sleep(10000) //10 second
    endif

endfunc 1
```

Pseudo Code 1 – Sleep Mode

[SJL]

4.6.5 Downlink Encoder

The encoder is implemented in LoraWan network (The Things Network). The "Encoder()" function has an object parameter passed from HTTP Post method via port 0. This function will return the first element of object (time period) as bytes.

Module	Encoder
Inputs	Timer Period in JSON format
Outputs	Integer Value
Functionality	Encoder converts sleep period of JSON format into integer value. The converted integer value is used for timer period.

Table 30 - Functional Table -Encoder

```
function Encoder(object, port) {
  // Encode downlink messages sent as
  // object to an array or buffer of bytes.
  var bytes = [];

  if (port === 0)
  bytes[0] = object.wake_period_sec;

  return bytes;
}
```

Pseudo Code 2 – Encoder

[SJL]

4.6.6 ADC

The GPIO pin in the RM191 board can be configured to analog input by “GpioSetFunc()”. It will return the analog input to the hexadecimal and send to the Lora Wan server as string.

Module	ADC
Designer	Seung Jun Lee
Inputs	Analog Input
Outputs	Sequence of 10 bit Digital
Functionality	“GpioSetFunc()” function read the analog value and convert into hexadecimal value. The data send to Lora Wan network as string format.

Table 31 - Functional Table -ADC

```
function HandlrNextTx()
    print "\n-----------------------------------------------------------------
    //sprint #str$,"\nAdr received (Type: ";PacketType;") - "
    rc = LORAMACGetOption(LORAMAC_OPT_TX_POWER, pow$)
    rc = LORAMACGetOption(LORAMAC_OPT_TEMPERATURE, temp$)

    mc=GpioSetFunc(MOIST_PIN,3,0x13)
    moist_level = GpioRead(MOIST_PIN)
    SPRINT #moist_string$,integer.h' moist_level
    data$ = moist_string$

    print "mositlevel=0x"+moist_string$+"\n"

    sprint #str$,"The Output Power: ";pow$;"dB\n";
    Print str$
    PRINT "Supply voltage: "; ReadPwrSupplyMv();"mV\n"
    PRINT "The Temperature: "
    sprint #str4$,temp$;"Celcius Degree";
    Print str4$
    print "\n-----------------------------------------------------------------

    rc = LORAMACTxData(2,data$, 1)

endfunc 1
```

Pseudo Code 3 – ADC

[SJL]

4.6.7 HTTP Integration

The downlink and uplink from Lora Wan network is sent as JSON object to the mobile device's using HTTP Post method. Using Post method, the JSON object is decoded and encoded in PHP code.

```php
<?

//JSON DATA FORMAT:
{"app_id":"moisturesensor","dev_id":"0000000000000001","hardware_serial":"0000000000000001","port":2,"counter":25,"co
nfirmed":true,"payload_raw":"TW9pc3RMZXZlbDogMDAwMDAwMDM=","metadata":{"time":"2018-03-04T21:59:19.946441817Z","frequ
ency":904.5,"modulation":"LORA","data_rate":"SF7BW125","coding_rate":"4/5","gateways":[{"gtw_id":"rg1xx2938ff","gtw_t
rusted":true,"timestamp":4041358643,"time":"","channel":3,"rssi":-16,"snr":9.75,"rf_chain":0,"latitude":41.074627,"lo
ngitude":-81.51283}]},"downlink_url":"https://integrations.thethingsnetwork.org/ttn-us-west/api/v2/down/moisturesenso
r/node1?key=ttn-account-v2.hnHzA4-leW-UKLvNB2jHmdNn9l1eqOvcl8CiA6LuaFs"}
include "reum_db.php";
   session_start();
   $_SESSION["wakeup_period"] = $_REQUEST["wakeup_period"];
   $wakeup_period = $_SESSION["wakeup_period"];
   date_default_timezone_set('America/New_York');
   $db = new Reum_DB();

   //UPLINK: Receive JSON data.
   $request_body = file_get_contents('php://input');
   $dev = json_decode($request_body, true);
   $dat["app_id"] = $dev["app_id"];
   $dat["dev_id"] = $dev["dev_id"];
   $dat["data"] = hexdec(base64_decode($dev["payload_raw"]))*3.6/1024;
   $dat["raw"] = $request_body;
   $dat["timestamp"] = date("Y-m-d H:i:s");
   $db->insData("app_id", $dat);
   $db->insData("dev_id", $dat);
   $db->insData("moist_sensor1", $dat);
   $rtn = array();
   $rtn["err"] = 0;
   echo json_encode($rtn);

    //DOWNLINK: Send the JSON data to the End-Node
    $data = '{"dev_id": "0000000000000001","port": 0,"confirmed": false,"payload_fields": {"wake_period_sec": '.
$wakeup_period.'}}';
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL,
"https://integrations.thethingsnetwork.org/ttn-us-west/api/v2/down/moisturesensor/node1[redacted]");
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
    curl_setopt($ch, CURLOPT_POST, 1);
    curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Type: application/json; charset=utf-8"));
    curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
    $data = curl_exec($ch);
    curl_close($ch);

    //Send the response back to the Javascript code
    echo $data;
?>
```

Pseudo Code 4 – HTTP Integration (Lora Device APP Key cannot be shared)

[SJL]

4.6.8 Real-Time Graph

The real-time graph engine is designed using jQuery Mobile. It will retrieve the most recent data saved in SQL database and display in the graph.

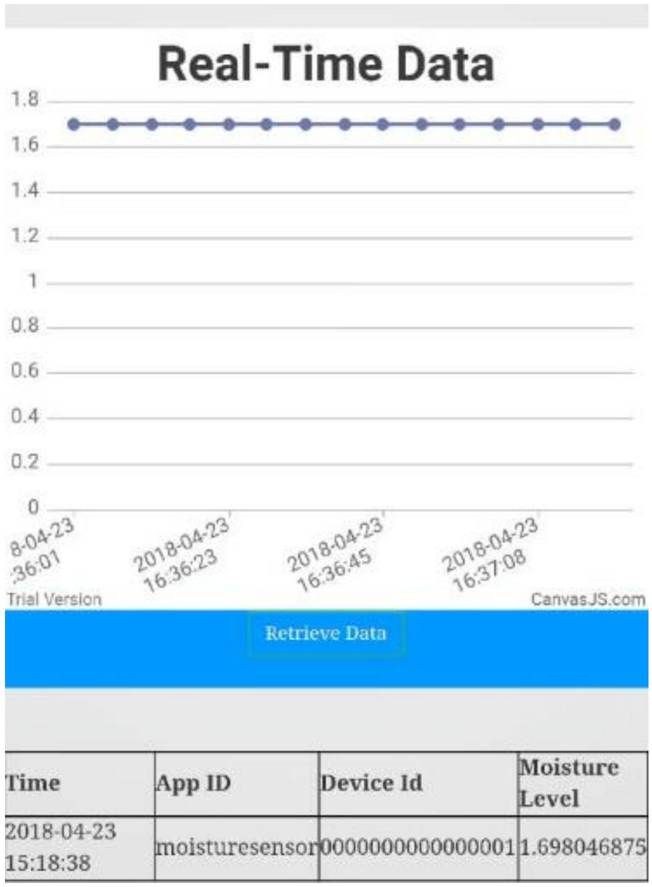
```
<script>
var xValue = 0;
var yValue = 10;
var newDataCount = 20;
var dev_id = "0000000000000001";
var dataPoints = [];
var chart = null;

function retrieve_data(){
        $.getJSON("http://reumapp.cafe24.com/action.php?tp=retrieve", function(data){
                //alert("First Success");
                var time = data['timestamp'];
                var app_id = data['app_id'];
                var dev_id = data['dev_id'];
                var data = data['data'];
                    document.getElementById("time").innerHTML = time;
                    document.getElementById("app_id").innerHTML = app_id;
                    document.getElementById("dev_id").innerHTML = dev_id;
                    document.getElementById("data").innerHTML = data;
        })
}

function updateData() {

    $.getJSON("action.php?tp=graph&dev_id="+dev_id+"", function(data) {
        dataPoints.push({label: data.check.timestamp, y: parseFloat(data.check.data)});
        if(dataPoints.length>newDataCount) {
            dataPoints = dataPoints.slice(1);
            //dataPoints.shift();
        }
        chart.render();
        setTimeout("updateData()", 5000);
    });
}
window.onload = function() {
    console.info("B");
    setTimeout("updateData()", 1000);
}
</script>
```

Pseudo Code 5 – Real Time Graph



Time	App ID	Device Id	Moisture Level
2018-04-23 15:18:38	moisturesensor	0000000000000001	1.698046875

Table 32 - Real Time Graph UI

[SJL]

4.6.9 Email Notification System

The email notification system is triggered when the data in real-time graph is larger than the threshold value set by the user. It will input the user's email and send the warning messages.

```
case "graph":
    $dev_id = $_REQUEST["dev_id"];
    $db = new Reum_DB();
    $rtn = $db->queryOne("SELECT id, timestamp, app_id, dev_id, data FROM moist_sensor1 order by id desc
limit 1");
    $rtn["timestamp"] = date("Y-m-d H:i:s");
    if($rtn["data"]>$_SESSION["threshold"]){
    $msg = "Moisture Level is Too Dry₩n Please Check";

    // use wordwrap() if lines are longer than 70 characters
    $msg = wordwrap($msg,70);

    // send email
    mail($_SESSION["email"],"Moisture Notification",$msg);
    }

    $send['check'] = $rtn;
break;
```

Pseudo Code 6 – Email Notification System

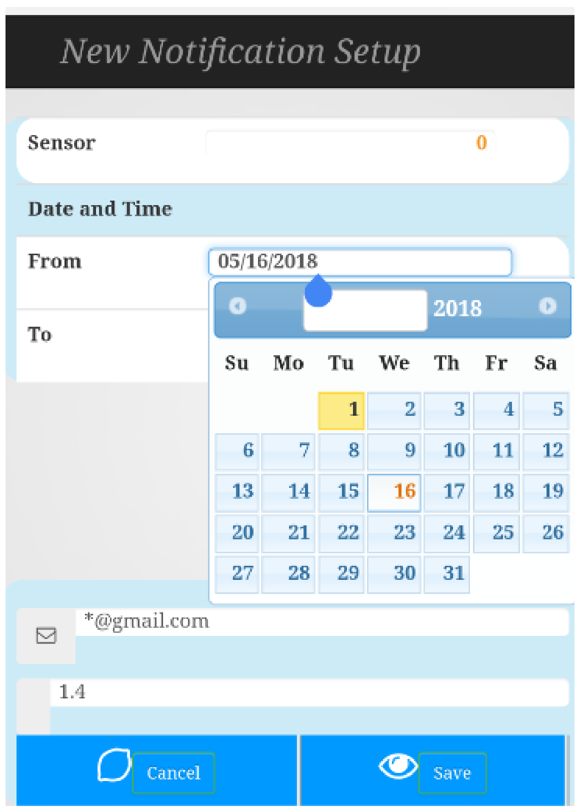


Table 33 - Notification Setup UI

Table 34 - Email Notification

[SJL]

4.6.10 Remote Power Control

The remote power control can choose the wake-up period when user clicks the desired button. When the button is clicked, it will use the HTTP Post method in "Pseudo Code 4" to downlink the wake-up period value into the device.

```
<script>
function wakeup0( ){
    alert("Real Time");
    var wakeup_period = document.getElementById("period0").value;
        $.getJSON("http://reumapp.cafe24.com/set.php?wakeup_period="+wakeup_period+"", function(data){
                //alert("First Success");
        })
}
function wakeup1( ){
    alert("Sleep-Mode Run: 30 Second");
    var wakeup_period = document.getElementById("period1").value;
        $.getJSON("http://reumapp.cafe24.com/set.php?wakeup_period="+wakeup_period+"", function(data){
                //alert("First Success");
        })
}
function wakeup2( ){
    alert("Sleep-Mode Run: 1 minute");
    var wakeup_period = document.getElementById("period2").value;
        $.getJSON("http://reumapp.cafe24.com/set.php?wakeup_period="+wakeup_period+"", function(data){
                //alert("First Success");
        })
}
function wakeup3( ){
    alert("Deep-Sleep Mode: Power Off");
    var wakeup_period = document.getElementById("period3").value;
        $.getJSON("http://reumapp.cafe24.com/set.php?wakeup_period="+wakeup_period+"", function(data){
                //alert("First Success");
        })
}
</script>
```

Pseudo Code 7 – Remote Power Control

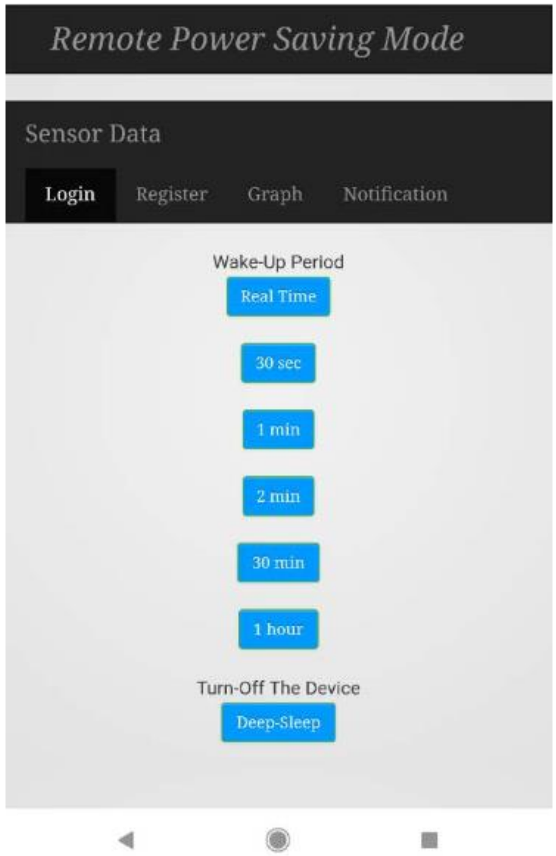


Table 35 - Remote Power Module UI

[SJL]

5.0 Operation, Maintenance, and Repair Instructions

5.1 Calibration and Interpretation

Calibration is needed for different soil types, as well as different soil densities; this would need to be done in software. By adjusting the threshold voltage levels we can adjust for different soil conditions. The system created for demonstration was calibrated for potting soil that was loosely poured into a pot. Should a user wish to utilize this system for different conditions, a few steps would should be taken. Firstly, a resistance measurement should be taken for soil the user determines to be the "target level" of moisture. Then in software a different voltage threshold can be set based on the users preference. The default settings of the sensor yield a voltage output of

15 mV for desired soil moisture. This should be a suitable setting for most users. Over 2 volts suggests the soil is quite dry and is in need of watering.

[DB]

5.1 Application Operation

The operation of the application which can be used on a web based platform or a mobile application is fairly user friendly. The application was made in a way that any user who is familiar with web or mobile applications can easily use the application. See section 4.6 Software Design Implementation Update for further explanation.

6.0 Parts List

Qty.	Refdes	Part Num.	Description
1			Medium 6V 2W Solar Panel
1			USB/DC/Solar Lithium Ino/Polymer Charger-v2
1			Male DC Power adapter - 2.1mm plug to screw terminal block
1			3.5/1.3mm or 3.8/1.1mm to 5.5/2.1mm DC Jack Adapter Cable
1			Lithium Ion Polymer Battery-3.7V 1200mAh
1			PowerBoost 500 Basic-5V USB Boost @ 500mA from 1.8V+
1			PowerBoost 1000 Basic-5V USB Boost @ 1000mA from 1.8V+
1			Lithium Ion Polymer Battery-3.7V 2000mAh
1			LT1006
4			1kΩ Resistor
3			5Ω Resistor
1			ALD212900
2			LoRa Mote

Table 36 - Parts List

6.1 Budget

Qty.	Part Num.	Description	Unit Cost	Total Cost
1		Medium 6V 2W Solar Panel	$29.00	$29.00
1		USB/DC/Solar Lithium Ino/Polymer Charger-v2	17.50	17.50
1		Male DC Power adapter - 2.1mm plug to screw terminal block	2.00	2.00
1		3.5/1.3mm or 3.8/1.1mm to 5.5/2.1mm DC Jack Adapter Cable	0.95	0.95
1		Lithium Ion Polymer Battery-3.7V 1200mAh	9.95	9.95
1		PowerBoost 500 Basic-5V USB Boost @ 500mA from 1.8V+	9.95	9.95
1		PowerBoost 1000 Basic-5V USB Boost @ 1000mA from 1.8V+	14.95	14.95
1		Lithium Ion Polymer Battery-3.7V 2000mAh	12.50	12.50
1		LT1006	4.49	4.49
4		1kΩ Resistor		
3		5Ω Resistor		
1		ALD212900	3.11	3.11
2		Lora Mote	73.49	146.98
			Total	$251.38

Table 37 - Material Costs

A special thanks goes to Laird for supplying us with a Lora Module.

7.0 Design Team Information

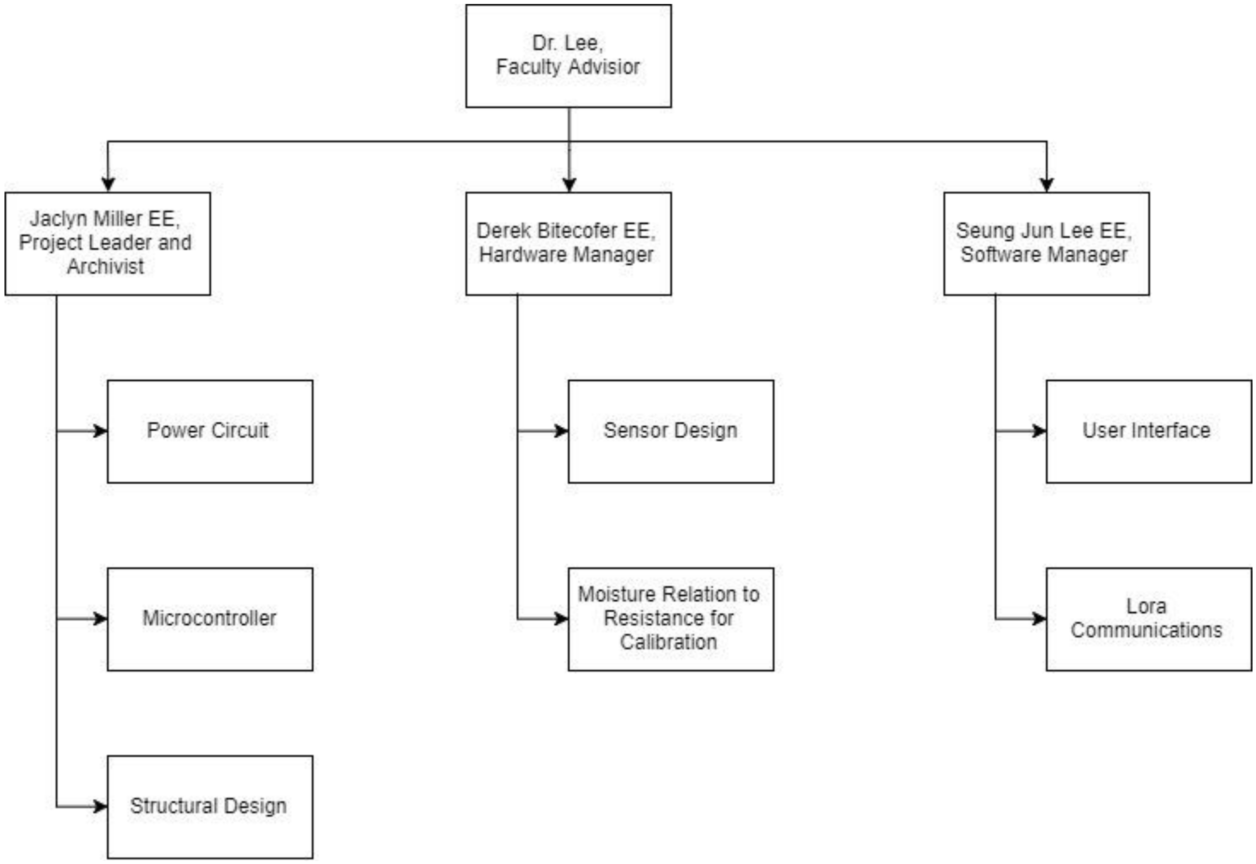


Figure 34 - Design Team Information

8.0 Conclusions and Recommendations

8.1 Hardware Conclusions and Recommendations

The design of the system's hardware should allow for the construction of an unattended ground moisture sensor. The sensor will be able to distinguish between wet and dry soil and relay that information to a microcontroller for wireless transmission. The structural design is intended to protect the circuitry and keep the components at a sufficient operating temperature.

The sensor relies on measuring the resistance of the soil. This is the simplest approach to the problem. The tradeoff is that the accuracy of the sensor will suffer compared to other methods. The accuracy could be improved by taking into account the temperature of the soil, as temperature affects resistance. Another approach altogether would be to measure capacitance and let the capacitance of the soil affect the oscillations of an oscillatory circuit. By counting the zero crossings of the oscillations you could determine the frequency of the oscillations. Thus, the frequency of the output would be related to the moisture of the soil.

[DB]

8.2 Software Conclusions and Recommendations

Our software design will allow the user to remotely monitor the soil moisture data on the mobile and web application. It also allows the user more visibility of collected data through the graphs and allow more interactive monitoring with the sensor through the notification feature. Allowing the user remote access to data requires wireless communication. The Lora device provides long range and low power operation. However, the constraint budget of this project limits the number of Lora devices that can connect to the server. The MCU sleep mode provides more sustainable power operation along with power supply from solar panel. If there is more budget and time, the capacity of wireless communication node can be enhanced.

[SJL]

9.0 References

[1] Israel Gluck, Anatoly Friedman, Naftali Feniger. US: 5424649 A. 13 6 1995.

[2] Stephen Charles Davis, Peter Johnson, John Huberts. US: 20080199359 A1. 21 8 2008.

[3] Eyjolf S. Bjornsson. US: 5621669 A. 15 4 1997.

[4] J. Uddin, S. M. T. Reza, Q. Newaz, J. Uddin, T. Islam and J. M. Kim, "Automated irrigation system using solar power," *2012 7th International Conference on Electrical and Computer Engineering*, Dhaka, 2012, pp. 228- 231.

[5] Ida, N. (2015). *Engineering Electromagnetics.* Springer.

[6] Studio, S. (2017). *A cheap soil moisture sensor - GardenBot*. *Gardenbot.org*. Retrieved 1 March 2017

[7] Brent Q. Mecham "*A Practical Guide to Using Soil Moisture Sensors to Control Landscape Irrigation*." Northern Colorado Water Conservancy District

[8] CHU, RUI HONG. *Solar Power and I-V Characteristics*. School of Electrical and Information Engineering, University of Sydney, www.cse.iitb.ac.in/synerg/lib/exe/fetch.php?media=public:students:sujesha:solar-panel.pdf.

[9] “Spectral Irradiance.” *Spectral Irradiance | PVEducation*, www.pveducation.org/pvcdrom/properties-of-sunlight/spectral-irradiance.

[10] Zientara, Ben. “How Much Electricity Does a Solar Panel Produce?” *Solar Power Rocks*, 9 Mar. 2017, solarpowerrocks.com/solar-basics/how-much-electricity-does-a-solar-panel-produce/.

[11] Barry Brents | Feb 28, 2014. “Understanding Lithium Batteries In Portable Electronics.” *Electronic Design*, 13 Mar. 2014, www.electronicdesign.com/power/understanding-lithium-batteries-portable-electronics.

[12] Albright, Greg, et al. *A Comparison of Lead Acid to Lithium-Ion in Stationary Storage Applications.* AllCell Technologies LLC, Mar. 2012, http://www.batterypoweronline.com/wp-content/uploads/2012/07/Lead-acid-white-paper.pdf.

[13] “How to Calculate Battery Run-Time.” *PowerStream Technology*, Lund Instrument Engineering, Inc., 29 Jan. 2017, https://www.powerstream.com/battery-capacity-calculations.htm.

[14] Greg, Bhaskar. *Introduction to Flow Batteries: Theory and Applications*. Bhaskar Garg, 22 Mar. 2012, http://large.stanford.edu/courses/2011/ph240/garg1/.

[15] *Boost/Inverting DC/DC Converter with 2A Switch, Soft-Start, and Synchronization*. Linear Technology Corporation, cds.linear.com/docs/en/datasheet/3580fg.pdf.

[16] SALAM, M.A., RAHMAN, Q.M., ANG, S.P. et al. J. Mod. Power Syst. Clean Energy (2017) 5: 290. https://doi.org/10.1007/s40565-015-0153-8

[17] ES.333, (May 7, 2014). *Circuits in Practice: The Wheatstone Bridge, What It Does, and Why It Matters* [Video File]. Retrieved from https://www.youtube.com/watch?v=-G-dySnSSG4

[18] Electronics Tutorials. “*The Differential Amplifier*., www.electronics-tutorials.ws/opamp/opamp_5.html

10.0 Appendices

10.1 Initial Power Design

Two important concepts are discussed in the power section of this report: the power sources, and power optimization. Both are important aspects, because the power sources chosen require optimization in order to function. Furthermore, optimization is heavily based on the type of power sources used. They are dependent on each other. Each topic is discussed separately below.

[MA]

10.1.1 Sources

Two power sources are used in this system: a solar panel and a battery. Each is discussed below in more detail.

[MA]

11.1.1.1. Solar Panel

Photovoltaic conversion can be used to convert sunlight into electricity. This is accomplished by using a solar panel, which consists of many cells made of large-area semiconductor diodes. Silicon absorbs sunlight and generates extra charges. Some of the cells are connected in series. Those series chains are then connected in parallel to increase the ability of the solar panel to generate power. The generated power is then used to power electronics and circuits. In this system, a solar panel will be used to provide power to the system and recharge the battery, thus allowing the system to be self-sufficient. Figure 35 - I-V Characteristics of Solar Panel [8] shows the I-V characteristics of a typical solar panel.

[MA]

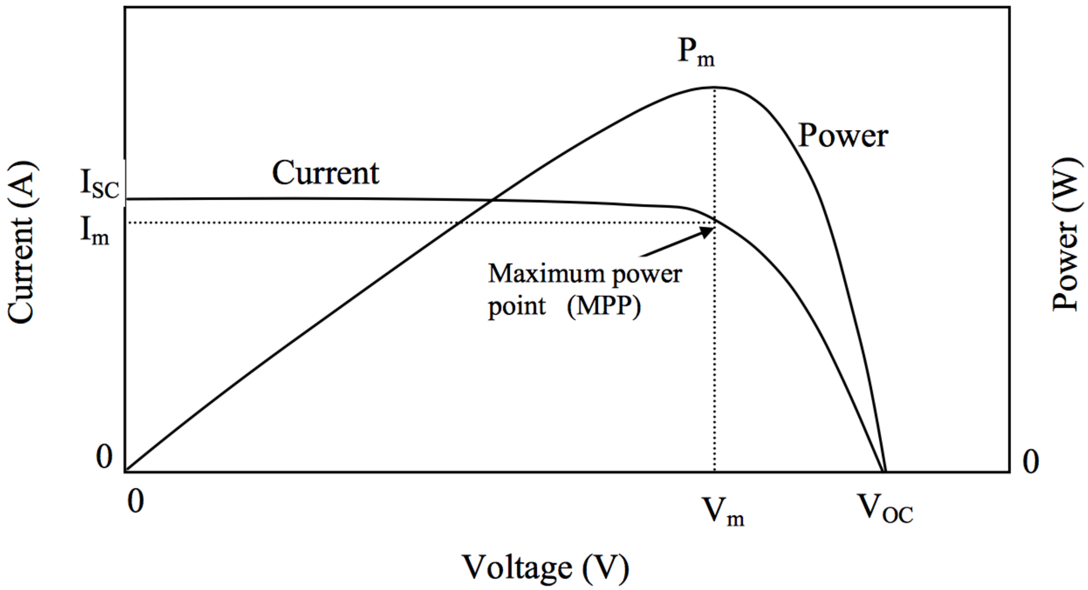


Figure 35 - I-V Characteristics of Solar Panel [8]

The power is proportional to the area of the solar panel. The ratio of power per square meter is called spectral irradiance. Spectral irradiance is based on photon flux, which is the number of photons per second per unit area. When photon flux is given, spectral irradiance is calculated using Equation 4, where:

- $F(\lambda)$ = spectral irradiance ($Wm^{-2}um^{-1}$)
- Ø = photon flux (#photons*$m^{-2}sec^{-1}$)
- E = energy of the photon (J)
- λ = wavelength of the photon (m) [9]

Note that irradiance varies with efficiency. Modern-day efficiencies for solar panels are around 20%. SOURCE. Equation 5 shows the approximate irradiance for solar panels with an efficiency of about 20%, after calculations. [10]

$$F(\lambda) = \frac{\phi E}{\Delta\lambda}$$

Equation 4

$$Irradiance = 0.108\, {}^{W}/_{in^2} = 0.01667\, {}^{W}/_{cm^2}$$

Equation 5

[MA]

11.1.1.2 Battery

One Lithium Ion (Li-Ion) battery will be used in this device to store power and supply the device when the solar panel does not have enough power to supply the device itself. Li-Ion will be used because it allows for the smallest size with the greatest energy capacity. [11] Furthermore, Li-Ion has good performance during temperature change. Figure 36 - Capacity vs. Temperature Graph of Various Battery Types [12]shows the capacity of various batteries, including Li-Ion, over the temperature range of -20 to 40C. It is clear that Li-Ion has the best performance, barely varying capacity over the entire temperature.

[MA]

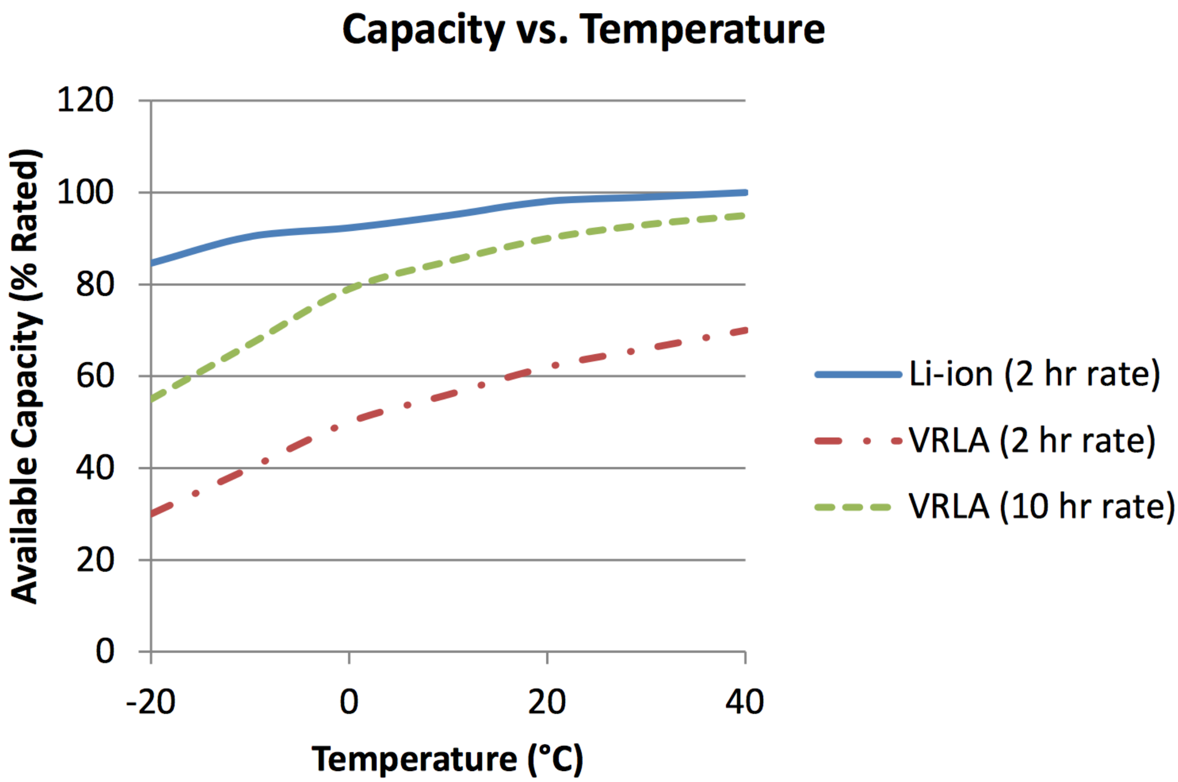


Figure 36 - Capacity vs. Temperature Graph of Various Battery Types [12]

There are many important characteristics of a battery. One is capacity, represented in amp-hours, which is how batteries are rated. This rating can be used to determine how many amps the battery will provide the system based on the hours it will be used, or how many hours the battery will run based on the amount of amps will be drawn from it. This will play a very important role in power optimization, to be discussed later. The equation for capacity is shown in Equation 6, or the current being drawn from the battery times the time. Another important characteristic is the energy stored in the battery. The energy changes with discharge, so the equation used to calculate that is shown in Equation 7, or the capacity times the nominal voltage.

$$C = It$$

Equation 6

$$E = CV_{nom}$$

Equation 7

It is important to be able to calculate and model the charging and discharging of the battery being used in this system. Equation 6 above can be solved for t to determine the time the battery takes to charge, in hours. In order to calculate the discharge time, in hours, Equation 8 is used, or the battery capacity times the nominal voltage of the battery over the load power. [13]

$$t = \frac{CV_{nom}}{P_{load}}$$

Equation 8

Finally, there are many efficiencies that are important for batteries: voltage efficiency, charge efficiency, power efficiency, and energy efficiency. The equations for all four efficiencies are shown in Equation 9,Equation 10, Equation 11, and Equation 12, respectively, where:

- $V_{discharge}$ = voltage of the battery when discharging
- V_{charge} = voltage being provided to the battery when charging it
- $Q_{discharge}$ = charge of the battery when discharging

- Q_{charge} = charge being provided to the battery when charging it
- $E_{discharge}$ = energy of the battery when discharging
- E_{charge} = energy being provided to the battery when charging it [14]

$$\eta_v = \frac{V_{discharge}}{V_{charge}}$$

Equation 9

$$\eta_c = \frac{Q_{discharge}}{Q_{charge}}$$

Equation 10

$$\eta_p = \frac{IV_{discharge}}{IV_{charge}}$$

Equation 11

$$\eta_E = \frac{E_{discharge}}{E_{charge}}$$

Equation 12

[MA]

11.1.2 Optimization Algorithm

For optimization, there are three cases for powering the system using a solar panel and a battery:

I. Solar panel powers the system and charges the battery

II. Solar panel powers the system alone

III. Battery powers the system alone

The system will know when to switch based on whether or not the solar panel can support the power requirements of the system and charge the battery simultaneously, since the power of the solar panel depends on how much sun it is exposed to. There will be a voltage sensor to sense the amount of voltage that the solar panel has. That data will go to the microcontroller. The microcontroller will determine if it has enough power to power the system and charge the battery. If it doesn't, it will then determine if it has enough power to at least power the system. If it is unable to do that as well, the battery will be used to turn on the system. The microcontroller will be able to do this by changing the PWM outputs of the microcontroller. The system will switch between cases using MOSFETs as switches and Power MOSFET drivers, controlled by a PWM output from the microchip, so by allowing the microcontroller to change its PWM outputs, the switches will turn on and off accordingly. The switching logic is as follows for the three cases:

I. Switches on: A, B, C; switches off: D

II. Switches on: A; switches off: B, C, D

III. Switches on: D; switches off: A, B, C

See Figure 37 - Block Diagram of Switch Operation for Power Cases I, II, and III below for the block diagram, as well as the switch locations. Note that there must be a delay in turning various switches on and off to ensure that there is never a moment that all power is disconnected from the system. Also, it is important to decide if the signal from the microchip needs to be high (1) or low (0) to turn specific switches on and off. This is important because if a switch needs to be on for an extended amount of time, a 0 should be used to turn it to save on power. If a 1 is used instead, the microchip will have to output that voltage level for the same amount of time which will consume power.

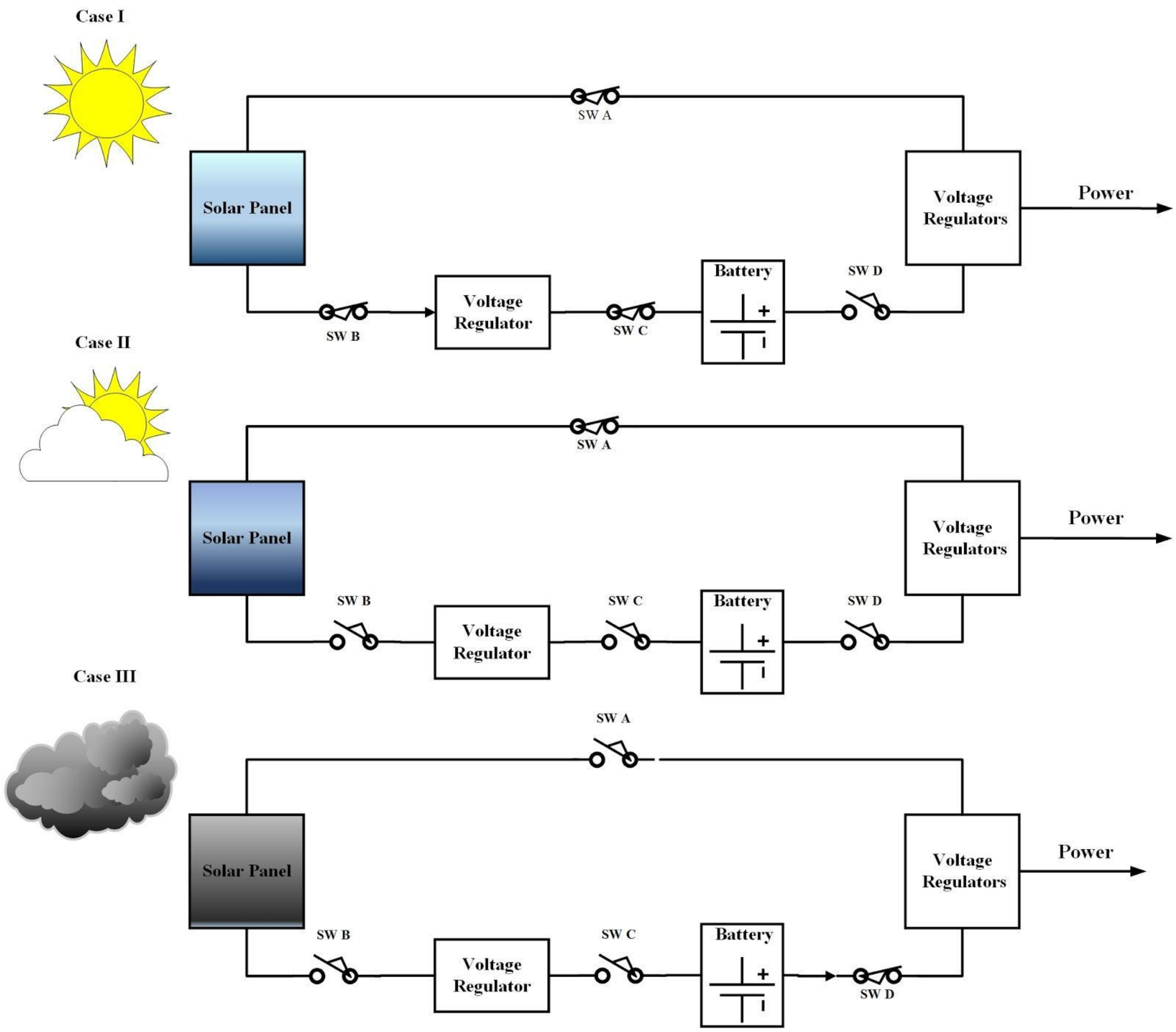


Figure 37 - Block Diagram of Switch Operation for Power Cases I, II, and III

There must also be a low-power mode in case there is no sun for a significant amount of time, to ensure that the battery is able to supply the system. This will be accomplished using a battery state-of-charge sensor. This will only be active when the system is in case 3. If the battery goes below a specific state-of-charge, the system will take soil moisture measurements less frequently in order to save power.

[MA]

11.1.3 Voltage Regulators

Various voltage regulators will be implemented in the circuit as well. There will be one voltage regulator between the solar panel and the battery, since the solar panel output will most likely need to be increased in order to charge the battery. There will also be multiple voltage regulators at the input to the system, based on the different voltages required. They will accept a large range of input voltages, since the voltage provided by the solar panel will likely fluctuate, but still provide a specific output voltage, although the output current will vary. This topology is called SEPIC. A typical circuit configuration for a SEPIC voltage regulator is shown in Figure 38 - Voltage Regulator Circuitry (SPICE Topology).

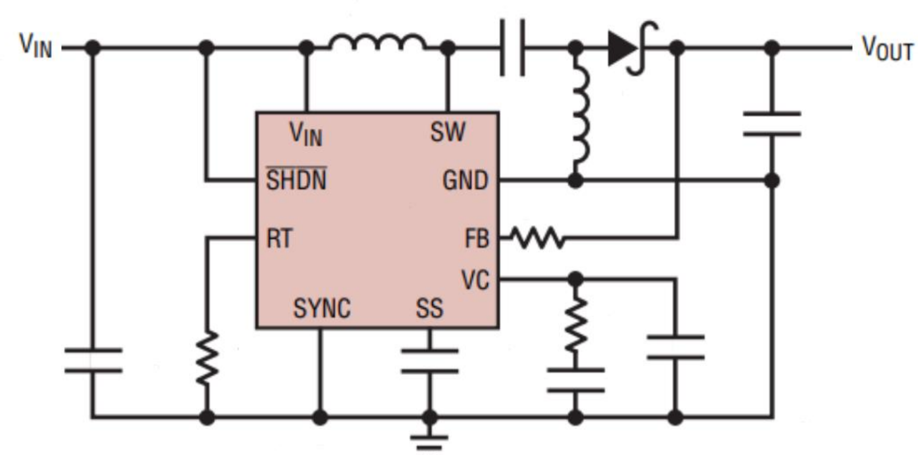


Figure 38 - Voltage Regulator Circuitry (SPICE Topology)

There are many equations involved in the design of a voltage regulator. The equations that will follow are specific for a SEPIC converter. Equation 13 shows the equation to calculate the resistance needed between the FB pin and ground to get a specific output voltage. The equation for the duty cycle is shown in Equation 14. In order to minimize loss in the inductors, high frequency core materials should be used like ferrite, and large volumes should be used. DCR should also be low. Equation 15 shows the calculation for the minimum inductance, and Equation 16 shows the calculation for the maximum inductance value. Finally, in order to prevent saturation and efficiency loss, inductors with ratings greater than their peak current must be chosen. The peak current is calculated using Equation 17. At the output, capacitors are needed to minimize the output ripple. Low ESR capacitors should be used with an X5R or X7R dielectric. A ceramic capacitor can be used to decouple the input. If too much ripple remains, an LC pi filter can be used at the output to filter out ripple even more. Schottky diodes should be used since they have low forward voltage drops and fast switching speeds. Finally, the resistor which sets the switching frequency based on the oscillating frequency is calculated shown in Equation 18. Also, the higher the switching frequency, the more efficiency loss, but the less board space used.

$$R_{FB} = \frac{V_{out - 1.215}}{83.3x10^{-6}}$$

Equation 13

$$DC \approx \frac{|V_{out}| + V_D}{V_{IN} + |V_{out}| + V_D - V_{CESAT}}$$

Equation 14

$$L > \frac{DC * V_{IN}}{2f(I_{LIM} - \frac{|V_{OUT}|I_{OUT}}{\eta V_{IN}} - I_{OUT})}$$

Equation 15

$$L_{MAX} = \frac{V_{IN} - V_{CESAT}}{I_{MIN} - RIPPLE}\frac{DC}{f}$$

Equation 16

$$I_{L-PEAK} = I_{OUT} + \frac{|V_{OUT}|(1 - D)}{2Lf}$$

Equation 17

$$R_T = \frac{91.9}{f_{OSC}} - 1$$

Equation 18

Multiple forms of power losses exist for voltage regulators: switch I^2R loss, NPN base drive (AC), NPN base drive (DC), and additional input current. The equations related to power loss are in Equation 19, Equation 20, Equation 21, Equation 22, and Equation 23, where R_{sw} is the switch resistance, DC is the duty cycle (from equation 11 above), and η is efficiency.

$$I_{IN} = \frac{V_{OUT} I_{OUT}}{\eta V_{IN}}$$

Equation 19

$$P_{SW} = DC * I_{IN}^2 * R_{SW}$$

Equation 20

$$P_{BAC} = 13\eta I_{IN} V_{OUT} f$$

Equation 21

$$P_{BDC} = \frac{DC * V_{IN} I_{IN}}{50}$$

Equation 22

$$P_{INP} = 7mA * V_{IN}$$

Equation 23

For heat regulation and thermal consideration, multiple vias should be used in the layout to guide heat to the copper layer in the PCB. In order to reduce noise, thick, short traces should be used in the layout of the PCB for the components corresponding to the voltage regulators. However, the VC and FB components should be placed far away from the switching node, and have a separate ground from the switch node as well. Suggested component placement for a SEPIC topology voltage regulator is shown in Figure 39 - Suggested Layout of Voltage Regulator Components (SPICE Topology). [15]

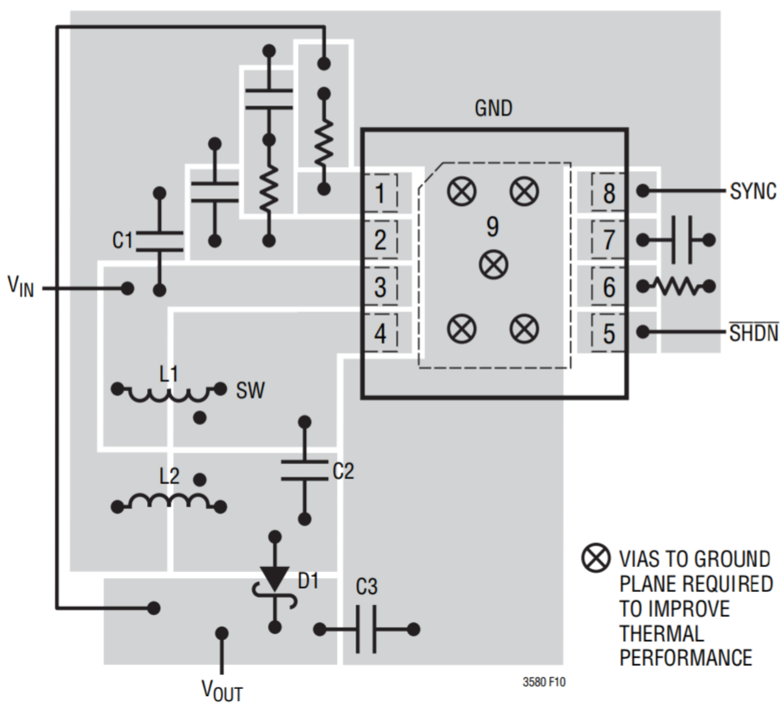


Figure 39 - Suggested Layout of Voltage Regulator Components (SPICE Topology)

[MA]

11.2 Pseudo Code

11.2.1 RTCC Pseudo Code

```
Start
sleep_for(seconds(3))
Initialize
     Set oscillator flag to 1
Calibrate
Check calibration
if (MeasuredFrequency==IdealFrequency)
{
Enable RTCC
Goto Mapping
}
else if (MeasuredFrequency>IdealFrequency || MeasuredFrequency<IdealFrequency)
{
//First Scenario: Positive Value, Fast Oscillator
//Second Senario: Negative Value, Slow Oscillator
Update correct value to calibration register
}
Mapping:
if(Mapping==True)
{
Reset Starting Point, Prescaler
}
else if
{
Write to RTCC
}
Alarm Register:
Set Mask = 0101 //Every Hour
Alarm Max Count
if (Max!=0)
{
Interrupt;
Max--;
Wake Up from SLeep Mode
}
else if
{
goto AlarmRegister
}
```

11.2.2 ADC/LCD Pseudo Code

```
Start
Initialize ADC/LCD
Select Channel //10bit
for (i=0; i>100; i++) //i>100 may need to be changed
{
time=3*T_AD
flag_bit=1;
sleep_for(nanoseconds(WaitTime)); //Need to figure out the WaitTime
Convert
ADC_Buf_Register=Conversion;
sleep_for(nanoseconds(AnotherWaitTime)); //Also figure out this wait time
flag_bit=0;
}
```

11.2.3 ISR UART Pseudo Code

```
FIFO Buf=ADC_Buf_Resigter;
if (Head!=Tail)
{
Free_space
Write to Register
Interrupt
Tail++
}
else if
{
Send to Tx
}
```

11.2.4 Communication (Using Lora RN2903 (PIC18) Library for MPLAB)

```
A. Initialization (Lora Stack and Radio Transceiver)
    1. Save Data to be sent: 8 bit
          Data = RxData(int* Data, int length, OpStatus_t Status);

2.      Activation Indication: 0 for connected and 1 for failed
           Status = RxJoinResponse();

3.      Initilization
           LORAWAN_Init(Data, status);

B. Activation (End Device)
    1. LORAWAN_SetNetworkSessionKey(nwkSKey); // nwkSKey = Array of 16 bytes
    2. LORAWAN_SetApplicationSessionKey(appSKey); // appSKey = Array of 16 bytes
    3. LORAWAN_SetDeviceAddress(devAddr);  // devAddr = 32 bits

C. Communication (Once the device is connected and Activated)
    1. LORAWAN_Send (PortNumber, DataPointer, DataSize);
```

11.2.5 Communication (Using Lora RN2903 (Connected to PC) Command Line)

```
A. Initialization & Activation: Using mac commands
< mac set devaddr “deviceAddress”
< mac set nwkskey “networkKey”
< mac set appskey “ApplicationKey”
< mac join abp // join the configured network
B. Communication: Using radio command
< radio cw on // set the Lora radio module into continuous wave transmission
< radio get mod // configure the default radio setting
< radio rx // receive data
```

11.2.6 User login.html

```
A. Login Form: Event Driven
<form method="post">
<input id="email" placeholder="email"></input>
<input type="password" id="password" placeholder="PASSWORD"></input>
<button type="button" name="login" onClick="login();"> </button>
</form>

B. Check if ID Exists in the Database by jQuery AJAX
function login(){
		var email = document.getElementById("email").value;
		var password= document.getElementById("password").value;
		$.getJSON("URL/action.php?tp=login&email="+email+"&password="+password+"",
function(data){
		if(data[email]==null) {
			alert("No Such User");

		}else {
			sessionStorage.setItem("email", data[email]);
		}
});
```

11.2.7 User register.html

```
A. Input the User Information
<form method="post">
<input id="email" placeholder="email"></input>
<input type="password" id="password" placeholder="PASSWORD"></input>
<input placeholder="dev_id"></input>
<button type="button" name="Register" onClick="regi();"> </button>
</form>

B. Check if ID exists. If not, pass user information to Server Side Script
function regi(){
		var email = document.getElementById("email").value;
		var password= document.getElementById("password").value;
		var dev_id= document.getElementById("dev_id").value;



	$.getJSON("URL/action.php?tp=login&email="+email+"&password="+password+"&dev_id="+dev_id+"",
function(data){
		if(data[email]!=null) {
			alert("Email Exist");

		}else {
			sessionStorage.setItem("email", data[email]);
		}
});
```

11.2.8 User notification.html

```
<form method="post">
<input id="minimum" placeholder="minimum"></input>
<input id="maximum" placeholder="maximum"></input>

$(function() {
   $( "#from" ).datepicker({
     defaultDate: "+1w",
     changeMonth: true,
     numberOfMonths: 1,
     onClose: function( selectedDate ) {
       $( "#from" ).datepicker( "option", "minDate", selectedDate );
               start_date = selectedDate;
     }
   });

   $( "#to" ).datepicker({
     defaultDate: "+1w",
     changeMonth: true,
     numberOfMonths: 1,
     onClose: function( selectedDate ) {
       $( "#to" ).datepicker( "option", "maxDate", selectedDate );
               end_date = selectedDate;
     }
   });
<button type="button" name="Register" onClick="notify();"> </button>
</form>

B. Pass User Notification Information to the Server
$.getJSON("URL/action.php?tp=notification&email="+email+"&start_date="+start_date+
"&end_date="+end_date+"&minimum="+minimum+&maximum="+maximum+"", function(data){});
```

11.2.9 Server action.php

```
A. Classify the User Request
<?
include "dbcon.php";
include "member_req.php";
	$tp = $_REQUEST["tp"];

	switch($tp) {
		case "login":
			$email = $_REQUEST["email"];
			$password = $_REQUEST["password"];
			$mem = new MemReq();
			$rtn = $mem->Login($email, $password);
			$send = $rtn;
			break;

		case "register":
			$email = $_REQUEST["email"];
			$upw = $_REQUEST["password"];
			$dev_id = $_REQUEST["dev_id"];
			$mem = new MemReq();
			$rtn =$mem->Member($email, $upw, $dev_id);
			$send = $rtn;
			break;

		case "alert_set":
			$email = $_REQUEST["email"];
			$dev_id = $_REQUEST["dev_id"];
			$start_date = $_REQUEST["start_date"];
			$end_date = $_REQUEST["end_date"];
			$minimum = $_REQUEST["minimum"];
			$maximum = $_REQUEST["maximum"];
			$add_alert = new Member_req();
			$add_alert->add_alert_info($email, $dev_id, $start_date, $end_date,
$minimum, $maximum);
			break;
	}
	echo $_GET['callback'];
	echo "(".json_encode($send).");";
?>
```

11.2.9 Server mem_req.php

```
A. Login Function
B. Register Function
C. Check Function
D. Retrieve Data Function
E. Notification Function
<?
include "db_con.php"; // for Database connection
class MemReq{

	function Login($email, $password){
		$db = new DBHelper(); // from db_con.php
		$db->Connect();
		$db->SetTBnm('user_info');
		$check=db->Query("select * from user_info where email = '".$email."'");
		//return
	       if($check){
		  $rtn[email] = null;
	       }
	}
	//register user
	function add_mem($email, $password, $num_sensor){
		$db = new DBHelper();
		$db->Connect();
		$db->SetTBnm("user_info"); //select table

		$db->Query("select * from user_info where email = '".$email."'");
		if($check){
			$db->InsDat("email", $email,1); //insert data
			$db->InsDat("password", $password,1);// 1 for string and 0 for number
			$db->InsDat("num_sensor", $num_sensor,1);
			$db->InsDat("date", "now()",0);
			$db->InsRun(); //
		}
	}

    function add_alert_info($email, $dev_id, $start_date, $end_date, $minimum, $maximum){
		$db = new DBHelper();
	       $db->Connect();
		$db->SetTBnm("alert_info"); //select table

		$check=$db->Query("select * from alert_info where email = '".$email."'");
		if($check){
			$db->InsDat("dev_id", $dev_id,1); //insert data
			$db->InsDat("start_date", $start_date,0);
			$db->InsDat("end_date", $end_date,0);
		       $db->InsDat("minimum", $minimum,0);
		       $db->InsDat("maximum", $maximum,0);
			$db->InsDat("date", "now()",0);
			$db->InsRun(); //
		}

}

return $rtn;
?>
```

11.2.9 Server db_con.php

```
A. Open Database Connection
B. Connect to MySql
C. Select Database
D. Make into Query
E. Insert into Database
F. Check SQL Query
class DB
{
	public $conn; public $result; public $sql; public $strTBName; public $strField;
	public $strValues; public $strUpSet;
	A.Open Database Connection
	function Connect()
	{
		$db_host="Host URL";
		$db_user="DB ID";
		$db_pass="DB Password";
		$db_name="DB Name";

	B. Connect to MySQL
		$this->conn = mysqli_connect($db_host, $db_user, $db_pass,$db_name) or
Error("DB ERROR");

	    C. Select the database
		mysqli_select_db($this->conn, $db_name) or Error("DB Select ERROR");
		return;
	}
	D.Make into Query
	function InsDat($dname, $dvalue, $type)
	{
		if ($dvalue != NULL)
		{
			if ($this->strValues != NULL) {
				$this->strField=$this->strField.",";
				$this->strValues=$this->strValues.",";
			}
			//if ($typedvalue == 1)
{$dvalue="'".str_replace("'","''",$dvalue)."'";}
			if ($type == 1) {$dvalue="'".$dvalue."'";}
			if ($type == 0) {$dvalue=$dvalue;}
			$this->strField=$this->strField.$dname;
			$this->strValues=$this->strValues.$dvalue;
		}
	}
	E.Insert into Database
	function InsRun()
	{
		$sql="INSERT INTO ".$this->strTBName." (".$this->strField.") Values (".$this-
>strValues.")";
		$result = mysqli_query($this->conn,$sql);
		if($result==0){echo "Error : INSERT INTO ".$this->strTBName." (".$this-
>strField.") Values (".$this->strValues.")";}
		$this->strDBName = "";
		$this->strField = "";
		$this->strValues = "";
	}
      F. Check SQL Query
      function check($sql)
```

```
        {
                $result = $this->conn->query($sql);
                return $result;
        }
?>
```

11.2.9 Notification Google API (GCM Server and curl function)

(a) Obtain the api-key,device ID and registration ID from Google API registration for each user's device
(b) Send Request to GCM Server

```
function notification($apikey, $registarationID, $message)
        {
                //Code from Google API Document
                //$header has $apikey
                //$data has $registration ID and $message
                 $ch = curl_init();
                curl_setopt( $ch, CURLOPT_HTTPHEADER, $headers );
                curl_setopt( $ch, CURLOPT_URL, "https://android.googleapis.com/gcm/send" );
                curl_setopt( $ch, CURLOPT_SSL_VERIFYHOST, 0 )
                curl_setopt( $ch, CURLOPT_SSL_VERIFYPEER, 0 );
                curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
                curl_setopt( $ch, CURLOPT_POSTFIELDS, json_encode($data) );

                $response = curl_exec($ch);
                curl_close($ch);

                return $response;
        }

    // If Data is smaller or larger than threshold call notification function
    $result=$db->Query("select * from alert_info where email = '".$email."'");
    if($data < $result['minimum'] || $data > $result['maximum'])
       notification($apikey, $registarationID, $message);
```

12.0 Project Schedule

12.1 Preliminary Fall 2017 Gantt Chart

	Task Name	Planned Start Date	Planned Finish Date	Assigned
1	Senior Design Soil Moisture Sensor	8/29/2017	12/5/2017	
2	Project Design	8/29/2017	12/5/2017	
3	Preliminary Design Report	8/29/2017	9/18/2017	
4	Problem Statement	8/29/2017	9/11/2017	
5	Need	8/29/2017	9/11/2017	Jaclyn, Mohammed
6	Objective	8/29/2017	9/11/2017	Jaclyn, Mohammed
7	Background	8/29/2017	9/11/2017	Derek, Jaclyn, ...
8	Marketing Requirements	8/29/2017	9/11/2017	Seung Jun, Jaclyn
9	Objective Tree	8/29/2017	9/11/2017	Jaclyn, Seung Jun
10	Preliminary Design Gantt Chart	9/5/2017	9/18/2017	
11	Block Diagrams Level 0, 1, ... w/ FR tables	9/5/2017	9/18/2017	
12	Hardware Module 0	9/5/2017	9/5/2017	Derek, Mohammed
13	Hardware Module 1	9/5/2017	9/18/2017	Derek, Mohammed
14	Software Module 0	9/5/2017	9/5/2017	Jaclyn, Seung Jun
15	Software Module 1	9/5/2017	9/5/2017	Jaclyn, Seung Jun
16	Preliminary Design Presentation 9:55am	9/19/2017	9/19/2017	Derek, Jaclyn, ...
17	Midterm Report	9/19/2017	10/16/2017	
18	Design Requirements Specification	9/19/2017	9/25/2017	Jaclyn
19	Midterm Design Gantt Chart	9/19/2017	10/16/2017	Jaclyn
20	Design Calculations	9/19/2017	10/16/2017	
21	Electrical Calculations	9/19/2017	10/16/2017	
22	Power Optimization	9/19/2017	10/16/2017	Mohammed
23	Sensor Design	9/19/2017	10/16/2017	Derek
24	Microcontroller Requirements	9/19/2017	10/16/2017	Jaclyn
25	Communications	9/19/2017	10/16/2017	Seung Jun
26	Block Diagrams Level 2 w/ FR tables & ToO	9/19/2017	9/25/2017	
27	Hardware Module 2	9/19/2017	9/25/2017	Derek, Mohammed
28	Software Module 2	9/19/2017	9/25/2017	Jaclyn, Seung Jun
29	Block Diagrams Level 3 w/ FR tables & ToO	9/26/2017	10/2/2017	
30	Hardware Module 3	9/26/2017	10/2/2017	Derek, Mohammed
31	Midterm Design Presentations 9:55-11:35am Part 1	10/17/2017	10/17/2017	Derek, Jaclyn, ...
32	Project Poster	10/31/2017	11/13/2017	Jaclyn
33	Final Design Report	10/17/2017	11/27/2017	
34	Abstract	10/17/2017	11/27/2017	Seung Jun
35	Software Design	10/17/2017	11/27/2017	
36	Modules 1...n	10/17/2017	11/27/2017	
37	User Interface Psuedo Code	10/17/2017	10/17/2017	Seung Jun
38	Microcontroller Psuedo Code	10/17/2017	11/27/2017	Jaclyn
39	Hardware Design	10/17/2017	11/27/2017	
40	Modules 1...n	10/17/2017	11/27/2017	
41	Power Optimization Simulations	10/17/2017	11/27/2017	Mohammed
42	Sensor Schematics	10/17/2017	11/27/2017	Derek
43	Parts Request Form	10/17/2017	11/27/2017	Derek
44	Budget (Estimated)	10/17/2017	11/27/2017	Jaclyn
45	Implementation Gantt Chart	10/17/2017	11/27/2017	Jaclyn
46	Conclusions and Recommendations	10/17/2017	11/27/2017	Derek, Jaclyn, ...
47	Final Design Presentations 9:55-11:35am Part 1	11/28/2017	11/28/2017	Derek, Jaclyn, ...
48	Final Design Presentations 9:55-11:35am Part 2	12/5/2017	12/5/2017	

Figure 40 - Preliminary Fall 2017 Gantt Chart

12.2 Final Fall 2017 Gantt Chart

ID	Task Name	Duration	Start	Finish	Resource Names
1	**SDP1 fall2017**				
2	**Project Design**				
3	**Preliminary Design Report**				
4	**Problem Statement**	21 days	Tue 8/29/17	Mon 9/18/17	
5	Need	14 days	Tue 8/29/17	Mon 9/11/17	Jaclyn,Mohammed
6	Objective	14 days	Tue 8/29/17	Mon 9/11/17	Jaclyn,Mohammed
7	**Background**	**14 days**	**Tue 8/29/17**	**Mon 9/11/17**	
8	Patent Search	14 days	**Tue 8/29/17**	**Mon 9/11/17**	Mohammed
9	Energy Harvsting Component	14 days	**Tue 8/29/17**	**Mon 9/11/17**	Jaclyn
10	Soil Moisture Sensor and Technology	14 days	**Tue 8/29/17**	**Mon 9/11/17**	Derek,Seung Jun
11	User Interface	14 days	**Tue 8/29/17**	**Mon 9/11/17**	Seung Jun
12	Marketing Requirements	14 days	Tue 8/29/17	Mon 9/11/17	Jaclyn,Seung Jun
13	Objective Tree	14 days	Tue 8/29/17	Mon 9/11/17	Jaclyn,Seung Jun
14	Preliminary Design Gantt Chart	14 days	Tue 9/5/17	Mon 9/18/17	Jaclyn
15	**Block Diagrams Level 0, 1, ... w/ FR tables**	**14 days**	Tue 9/5/17	Mon 9/18/17	
16	Hardware Module 0	14 days	Tue 9/5/17	Mon 9/18/17	Derek,Mohammed
17	Hardware Module 1	14 days	Tue 9/5/17	Mon 9/18/17	Derek,Mohammed
18	Software Module 0	14 days	Tue 9/5/17	Mon 9/18/17	Jaclyn,Seung Jun
19	Software Module 1	14 days	Tue 9/5/17	Mon 9/18/17	Jaclyn,Seung Jun
20	Preliminary Design Presentation 9:55am	1 day	Tue 9/19/17	Tue 9/19/17	Derek,Jaclyn,Mohammed,Seung Jun
21	**Midterm Report**	**28 days**	**Tue 9/19/17**	**Mon 10/16/17**	
22	Design Requirements Specification	7 days	Tue 9/19/17	Mon 9/25/17	Jaclyn
23	Midterm Design Gantt Chart	28 days	Tue 9/19/17	Mon 10/16/17	Jaclyn
24	**Design Calculations**	**28 days**	**Tue 9/19/17**	**Mon 10/16/17**	
25	**Electrical Calculations**	**28 days**	**Tue 9/19/17**	**Mon 10/16/17**	
26	Power Optimization	28 days	Tue 9/19/17	Mon 10/16/17	Mohammed
27	Sensor Design	28 days	Tue 9/19/17	Mon 10/16/17	Derek
28	Microcontroller Requirements	28 days	Tue 9/19/17	Mon 10/16/17	Jaclyn
29	Communications	28 days	Tue 9/19/17	Mon 10/16/17	Seung Jun
30	**Mechanical Calculations**	**28 days**	**Tue 9/19/17**	**Mon 10/16/17**	
31	Structural Design	28 days	Tue 9/19/17	Mon 10/16/17	
32	**Block Diagrams Level 2 w/ FR tables & ToO**	**7 days**	**Tue 9/19/17**	**Mon 9/25/17**	
33	Hardware Module 2	7 days	Tue 9/19/17	Mon 9/25/17	Derek,Mohammed
34	Software Module 2	7 days	Tue 9/19/17	Mon 9/25/17	Jaclyn,Seung Jun
35	**Block Diagrams Level 3 w/ FR tables & ToO**	**7 days**	**Tue 9/19/17**	**Mon 9/25/17**	
36	Hardware Module 3	7 days	Tue 9/26/17	Mon 10/2/17	Derek
37	Software Module 3	7 days	Tue 9/26/17	Mon 10/2/17	Seung Jun
38	Midterm Design Presentations 9:55-11:35am Part 1	1 day	Tue 10/17/1	Tue 10/17/17	Derek,Jaclyn,Mohammed,Seung Jun
39	**Final Design Report**	**42 days**	**Tue 10/17/1**	**Mon 11/27/17**	
40	Abstract	42 days	Tue 10/17/1	Mon 11/27/17	Derek
41	**Software Design**	**42 days**	**Tue 10/17/1**	**Mon 11/27/17**	
42	**Pseudo Code**	**42 days**	**Tue 10/17/1**	**Mon 11/27/17**	
43	RTCC	42 days	Tue 10/17/1	Mon 11/27/17	Jaclyn
44	ADC/LCD	42 days	Tue 10/17/1	Mon 11/27/17	Jaclyn
45	ISR UART	42 days	Tue 10/17/1	Mon 11/27/17	Jaclyn
46	Communications	42 days	Tue 10/17/1	Mon 11/27/17	Seung Jun
47	User Interface	42 days	Tue 10/17/1	Mon 11/27/17	Seung Jun
48	Server	42 days	Tue 10/17/1	Mon 11/27/17	Seung Jun
49	Notification	42 days	Tue 10/17/1	Mon 11/27/17	Seung Jun
50	**Hardware Design**	**42 days**	**Tue 10/17/1**	**Mon 11/27/17**	
51	**Modules 1**	**42 days**	**Tue 10/17/1**	**Mon 11/27/17**	
52	Sensor Simulation	42 days	Tue 10/17/1	Mon 11/27/17	Derek
53	Sensor Schematic	42 days	Tue 10/17/1	Mon 11/27/17	Derek
54	Redesign of Power Design	7 days	Tue 11/21/1	Mon 11/27/17	Jaclyn
55	Implementation Gantt Chart	42 days	Tue 10/17/1	Mon 11/27/17	Jaclyn
56	Conclusions and Recommendations	42 days	Tue 10/17/1	Mon 11/27/17	Derek,Seung Jun
57	Parts Request Form	4 days	Tue 11/28/1	Fri 12/1/17	Jaclyn
58	Budget (Estimated)	4 days	Tue 11/28/1	Fri 12/1/17	Jaclyn
59	Final Design Presentations 9:55-11:35am Part 1	1 day	Tue 11/28/1	Tue 11/28/17	Derek,Jaclyn,Seung Jun
60	Project Poster	7 days	Tue 12/5/17	Mon 12/11/17	Jaclyn,Derek,Seung Jun

Figure 41 - Final Fall 2017 Gantt Chart