PREDICTIVE PLANT CARE

By

Charlotte Fondren

Thomas Wolf

Tom Danielson

Final Report for ECE 445, Senior Design, Spring 2023

TA: Selva Subramaniam

2 May 2023

Project No. 58

Abstract

This project aims to automatically care for a plant's fundamental needs such as light, water, and soil nutrients. This project was successful and produced a device that calculated and adjusted the amount of light administered to the plant through a grow light bulb, and measured the amount of moisture in the soil and administered water accordingly. While the device was unable to automatically administer soil nutrients, the device measures the pH in the soil and alerts the user when it is needed.

Contents

1. Introduction

1.1 Problem

Owning a plant can be a great way to liven up the room. However, many plants require consistent, oftentimes everyday care, and forgetting to take care of a plant can be detrimental to the plant's health. Leaving a plant unattended for a long time, such as the plant parent going on a vacation, can also result in the plant dying out unless it's a low-maintenance plant. Figure 1 shows that In 2021, 73.2% of recreational gardeners killed one or more plants in a year, with almost ⅙ of those killing 6 or more plants. Many people struggle to take care of their houseplants, especially those who are inexperienced or even recreational gardeners.

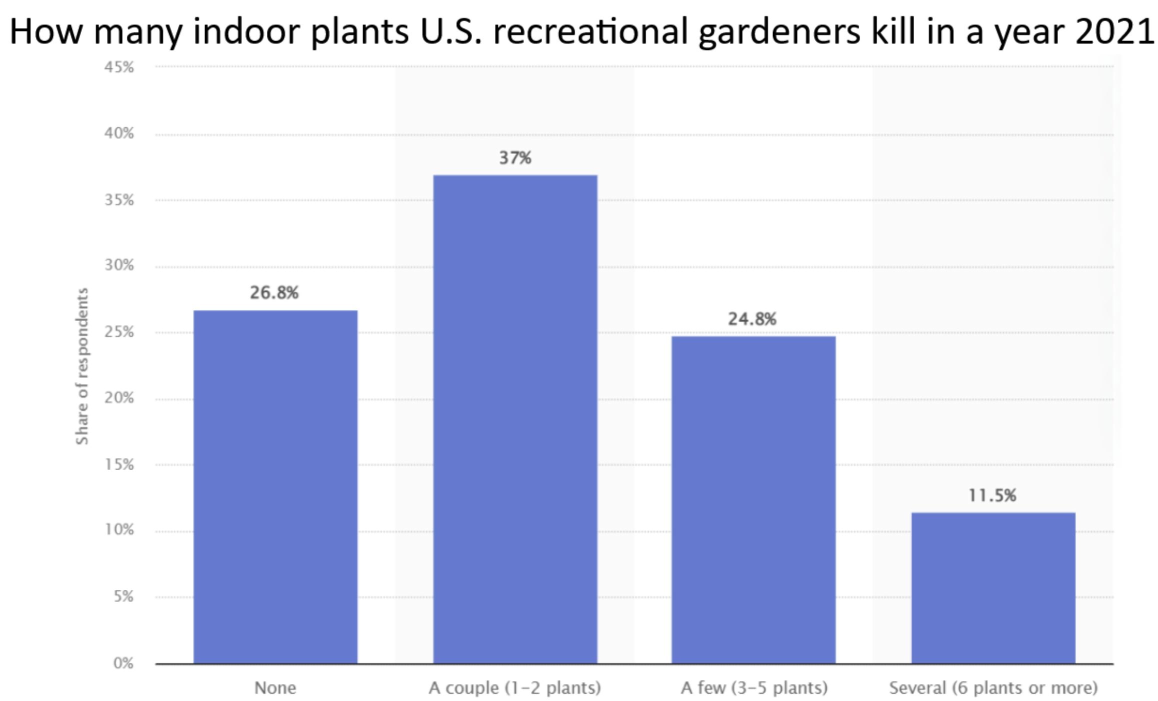

Figure 1 How Many Indoor Plants U.S. Recreational Gardeners Kill in a Year, 2021

1.2 Solution

In order to eliminate the dependence on human interaction, a plant care system was built that will take care of the plant's various needs such as water, sunlight, and soil pH and nutrients. Each of the respective resources will have a sensor that monitors how much of that resource the plant has, such as the moisture level of the soil, how much light the plant is receiving, and soil pH. The plant's needs will be taken care of automatically based on either time or sensor outputs. The overall goal of the device is to keep a plant alive for a long period of time. Specifically, the device will take care of a sunflower and allow it to grow and thrive throughout the project. A sunflower requires a lot of light, which allows the device to show its full capabilities in terms of light care.

While there are some automated solutions available, these devices only automate the water portion of a plant's needs and tend to be on the more expensive side - typically $100+ for a high-quality automated watering system alone. This device aims to minimize user input, including completely automating both water and light care, and making soil care simple and easy for users.

1.3 High Level Requirements

The plant (sunflower) is able to grow at least 6 inches.

Light levels are maintained between 12 and 20 mol·m-2·d-1 of Daily Light Integral (DLI) for the sunflower. DLI is the amount of photo-synthesizable light that the plant receives within a day.

Moisture levels of the soil are maintained between 12.1%-13.1% water weight per soil weight during flowering stages and 10.5%-11.5% water weight per soil weight during other times for the sunflower.

2 Design

The overall design and major subsystems are shown in Figure 2. There are five primary categories: light subsystem, soil subsystem, moisture subsystem, control subsystem, and power subsystem. The light, soil, and moisture subsystems are a part of the resource management system, the part of the device that directly takes care of the plants needs. The control and power subsystems are a part of the power system. The control subsystem is responsible for taking in the outputs from and sending out signals to the resource management system. The power subsystem is responsible for powering the rest of the subsystems and ensuring each component receives the voltage and current it requires. The light subsystem is responsible for ensuring the plant receives enough, but not too much, light. The soil subsystem is responsible for managing pH and alerting the user when to add fertilizer. Finally, the moisture subsystem is responsible for watering the plant when moisture levels in the soil are low.

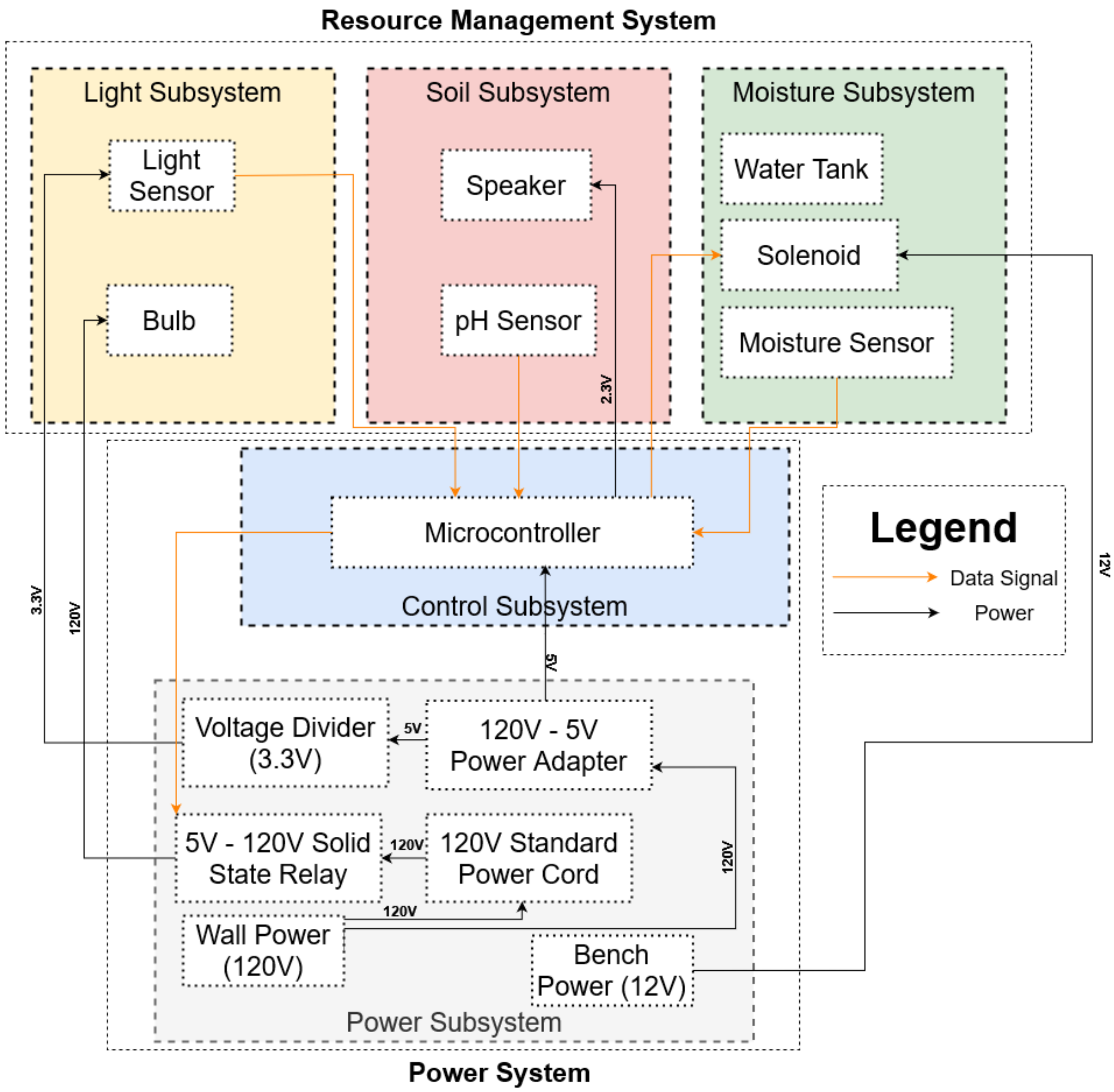

Figure 2 Block Diagram

2.1 Moisture Subsystem

The goal of the moisture subsystem is to continuously measure the amount of moisture in the soil and provide water to the plant when the soil is dry. This subsystem contains the water tank, water valve, and moisture sensor. It is one of the most important subsystems of the device. The moisture sensor must be able to monitor the moisture content of the soil every 5 seconds, and water must be dispensed with a timed release when the moisture content is low.

The soil moisture sensor sends a signal to the microcontroller, providing updates on the level of moisture in the soil. When the moisture level gets too low, the microcontroller sends out a timed signal to the solenoid, releasing water to the sunflower for 6.5 seconds, waiting 15 seconds, then re-evaluating the moisture levels.

2.1.1 Moisture Level Required

In order to accurately create a model for the moisture level of the plant, the acceptable soil moisture range had to be calculated. The moisture levels cannot be too high or too low, so the system needs to find a good balance.

The optimal moisture content of the soil depends on which soil is used. The sunflower used in this project was planted in sandy loam soil. The optimal moisture of the soil can be calculated based on the soil's available water capacity and the sunflower's optimal water depletion. Available water capacity is the field capacity subtracted by the wilting point. The field capacity is the maximum water held in the soil without draining due to gravity and the wilting point is the minimum amount of water in the soil in which a plant could still extract water. Sandy loam soil has a field capacity of 0.23 inches of water per inch of soil depth and a wilting point of 0.10 inches of water per inch of soil depth, which means that sandy loam's available water capacity is 0.13 inches of water per inch of soil depth.

Sunflowers grow best in moisture levels of 80% water weight per soil weight of field capacity at flowering stages and 70% water weight per soil weight of field capacity at all other times. For a moisture level of 80% of field capacity, the optimal water content within sandy loam soil is 0.184 inches of water per inch of soil depth, which would be 5.2 cubic inches of water per inch of soil depth. Assuming a 6" diameter pot with a 5" depth, this would amount to 25.99 cubic inches of water total in the pot, which is 425.9761 grams of water. The density of sandy loam soil is 1460 kg per cubic meter, which means the optimal moisture level for a flowering sunflower would be 12.6% of water weight per soil weight.

For a moisture level of 70% water weight per soil weight of field capacity, this means the optimal water content within sandy loam soil would be 0.161 inches of water per inch of soil depth, which would be 4.55 cubic inches of water per inch of soil depth. Assuming the same pot size as before, this would amount to 22.75 cubic inches of water total in the pot, which is 372.8725 grams of water. Assuming the same density of sandy loam soil as previously, the optimal moisture level for a growing non-flowering sunflower would be 11% of water weight per soil weight.

2.1.2 Water Distribution System

While a sunflower prefers drier soil, it does need water. As such, it was necessary to set up a method of watering the soil when the soil was below the optimal moisture level. To do so, this device used part of a previous project's water distribution system. It contained a water tank, tubes to and from a solenoid, a weight sensor, and wooden planks connecting everything. The weight sensor was removed. The solenoid used was a 2W025-08 12V solenoid from Tailonz and was used because it was already connected to the rest of the water distribution system. Initially, the SSM3K36MFV,L3F MOSFET was chosen to allow current to flow through the solenoid when a signal was applied to the gate from the microcontroller. However, this surface-mount part was not practical for the breadboard, nor would it have been able to handle the 1A from the solenoid. Two IRF510 MOSFETs were settled on as they could handle 1A when in parallel with a gate voltage of 5V.

The initial design was to make the power subsystem use a 100-240V AC to 12V DC power adapter to accommodate for the solenoid. A linear voltage regulator was initially planned to step this 12V down to 5V. However, this would have resulted in a massive power loss with the 7V drop and current being drawn for the rest of the subsystems, likely causing the regulator to heat up past its limit. As such, a separate power supply was used.

2.1.3. Soil Moisture Sensor

A sunflower requires more dry soil than many plants, so it is important to monitor the soil moisture levels. There are a lot of products that perform this same task, but this was an important resource for our project to monitor. The soil moisture sensor is intended to measure the moisture levels of the soil, and send that information to the microcontroller.

This device used the FC-28 soil moisture sensor which has a supply voltage between 3.3V to 5V and a current of 15mA. This sensor works by measuring the resistance between its two probes. The higher the resistance, the lower the water content of the soil. The output value ranges between 0V-5V and is interpreted by the microcontroller as values ranging from 0-1023, 1023 being dry soil or 0% water content, and 0 being very wet soil or almost 100% water content in theory. This was not the case in practice, however, as soil that was at field capacity would only reach a value of around 200.

2.2 Light Subsystem

The goal of the light subsystem is to measure the amount of light present and provide light to the plant. This subsystem contains the light sensor, a solid state relay, and the lightbulb. It is also one of the most important subsystems of the device, especially considering a sunflower requires a significant amount of light to grow and thrive. The light sensor must be able to measure the lux for the system. This subsystem must be able to monitor and control the light duration between 12 and 20 mol·m-2·d-1 of Daily Light Integral (DLI). The DLI is the amount of Photosynthetic Active Radiation (PAR) per 24 hours. PAR is the type of light required for photosynthesis to occur.

The light sensor sends a signal to the microcontroller, providing updates on the lux present. The integral controller then calculates the DLI continuously. An integral controller is a type of controller where the output is proportional to the integral of an error signal. The error signal is the difference between the input value and the target value. The input value for this device's integral controller is the readings from the light sensor, and the target value is the amount of DLI required. Once the DLI reaches target levels, the integral controller sends out a signal to the solid state relay, which turns off the lightbulb. The integral controller resets every 24 hours to simulate "days" for the sunflower.

The initial design was to use a Proportional-Integral-Derivative (PID) controller to control the duration of time that the light is on. A PID controller is a control loop mechanism that has a target, and uses the difference between the target and the input as error. The goal of a PID controller is to minimize the error, which it does through a series of calculations (proportions, integrations, and derivatives). However, PID controllers are generally used to control the luminosity of a lightbulb instead of the duration that the lightbulb is on. The lightbulb chosen for the design (see section 2.2.3) was not dimmable, making a PID controller more complicated than needed for this design. The final design instead used a more simple integral controller, described previously.

2.2.1 Light Level Required

While the sunflower is still growing (pre-flowering), it is recommended to have a Daily Light Integral (DLI) of 12-16 mol·m-2·d-1. Once the sunflower bud emerges, it is recommended to have a Daily Light Integral of up to 20 mol·m-2·d-1. To encourage flowering, it is recommended to have a photoperiod of more than 12 hours but less than 20 hours a day. The photoperiod is the amount of time a plant receives light within 24 hours. The PAR38 GE Grow Light for Plants (see section 2.2.3) supplies 50 micromoles per second of photosynthetic photon flux (PPF). This means that to supply a minimum of 12 mol·m-2·d-1 of DLI, excluding natural light, this light would need to be on for a minimum of 14.5 hours a day. To supply a maximum of 20 mol·m-2·d-1 of DLI, excluding natural light, this light would need to be on for a maximum of 24 hours a day.

2.2.2 Light Sensor

Most cost-effective light sensors simply measure whether light is present, and don't measure the amount of light present. Low quality light sensors that do measure the amount of light are $80+, which are simply not affordable given budget constraints. The light sensor is intended to be used to measure the amount of light present, and send that information to the microcontroller.

The initial design was to incorporate the AS7341 light sensor with a PCB. This light sensor measured the amount of photosynthetic flux density (PPFD) present. PPFD measures the amount of PAR present. The control system would then calculate the Daily Light Integral (DLI). However, there were difficulties working with the PCB and the device ended up breadboarded instead. With the breadboarded device, the AS7341 was no longer usable, so the modified design used the BH1750 light sensor. The BH1750 light sensor measured the amount of lux present. Lux is the unit of illuminance, or the luminous flux per unit area, in the International System of Units. The integral controller then calculates the Daily Light Integral from lux.

The BH1750 sensor requires 3.3 volts and uses an I2C serial interface to communicate with the microcontroller. The voltage requirement was satisfied using a voltage divider circuit to step down the voltage from 5 volts to 3.3 volts.

2.2.3 Supplemental Light System

As previously mentioned, a sunflower requires a significant amount of light. This device is meant to be placed inside, where most plant parents would have most need for an automated plant care system. This means that the plant would require supplemental light, as the amount of light that can be provided on a windowsill is not enough to grow a sunflower.

The device used the PAR38 GE Grow Light for Plants. A sunflower requires a lot of light in order to grow, much more than a window could provide. While it is the most expensive component of the device, this light produces 50 micromoles/second of PPF, which will allow the plant to get sufficient photosynthetic light.

The initial design was to incorporate the light bulb into the PCB, which would allow the microcontroller to directly turn the lightbulb on and off. However, this was a very unsafe design, as the lightbulb requires 120 volts. The SSR-40DA solid state relay was used instead to safely control the lightbulb. This relay has an input voltage of 3-32 volts DC and an output voltage of 24-380 volts AC. The microcontroller would output 5 volts to the relay when the DLI was less than the target, effectively turning on the lightbulb. It would also output 0 volts when the DLI reached the target, effectively turning off the lightbulb.

2.3 Soil Subsystem

The goal of the soil system is to determine the state of the soil over time through monitoring the pH and alert the user when to resupply nutrients, such as fertilizer (see section 2.3.3). The target range for the pH of the soil is between 6.5-7.5. If the pH is too low or too high, the plant's caretaker will be notified by a visual on the system that will let them know to add certain materials to bring the soil level back within range. The soil subsystem must be able to monitor the pH of the soil and notify the user when fertilizer is to be added once pH levels are outside the specified range.

The initial design was to keep the pH meter in the soil and constantly monitor the pH. However, the pH meter (see section 2.3.1) is not recommended to be kept in the soil for more than 15-20 minutes at a time. To remedy this, another concept included mechanically removing and putting the pH sensor in the soil when needed. However, this required mechanical design, which would've taken away from our electrical and computer design. Instead, the device will allow the user to place the pH meter into the soil, and the system will then alert them if the pH is not in the target range. Because soil pH changes over the course of weeks unlike with light or moisture, it is not required that the user frequently remember about measuring the soil pH.

Other initial design concepts included adding the fertilizer automatically once the soil pH was not in the target range. This would require either having a mixing mechanism with the soil or adding the fertilizer directly to the water tank. However, an automatic mixing mechanism could be dangerous to the plant, as it could destroy roots and disrupt the plant. Adding the fertilizer directly to the tank could work, but it would contaminate the entire water supply, potentially adding too much fertilizer to the plant. Instead, the user will have to manually add a small amount of fertilizer if the pH is not within the target range.

2.3.1 Soil pH Meter

A soil's pH is an indication of the amount of nutrients available to plants in the soil. The most common way of testing pH that most people think about is using Litmus paper. Litmus paper works well with liquid solutions; however, this would be difficult to test with soil. For soil, using a two-pronged sensor will work much better. The voltage measured in soil and soil pH are proportional. Therefore, measuring the voltage between the two prongs of the sensor will allow an easier time measuring the pH of the soil. The pH meter is intended to be used to measure the pH of the system, and send that information to the microcontroller.

The Sonkir soil pH meter was used not only for its ease of use, as it only relies on being placed in the soil to give a pH reading, but also for its low cost and accuracy, as many other soil pH meters were either a little far out of our budget or noted as not accurate or good to use by online reviewers. However, there were some complications with using this specific meter that were discovered too late into the project to change. For one, the meter is recommended not to stay in soil for more than 15-20 minutes, as it is typically only used for infrequent readings of soil, and if left in soil for too long, the pins could rust and the meter could be unusable. This also meant the system would have a tough time being fully automatic, since there would have to be a system to automatically add or remove the meter in the soil, and wipe the pins off. This just did not seem practical which is when the system requires some manual input instead (see section 2.3.2). In addition, this meter itself was not specifically designed as an electrical part used for other purposes, so it was difficult to find out where to actually hook up connections to get an accurate reading of the pH onto the breadboard for use interacting with the ATMega chip. If this project were to be redone or progressed in the future, a different pH meter would certainly be used instead.

2.3.2 User Alert System

If the soil's pH is measured to be outside of the targeted range, the device needs to be able to alert the user to add more nutrients in the form of fertilizer (see section 2.3.3). To accomplish this, the visual display on the Sonkir pH meter was used, along with a speaker that would emit a small noise whenever the pH was outside of the target range. This would notify the user, who would then manually adjust the pH values by adding fertilizer to the plant.

2.3.3 Nutrient Additive

When the pH of the soil is out of the target range, this is an indication that nutrients need to be added in order to bring the pH back into the target range.

The initial design was to ideally add specific nutrients such as Nitrogen, Phosphorus, and Potassium to the soil separately. However, this ended up being too impractical and expensive to buy and measure these quantities for the user to add to the soil. Instead, a fertilizer, the Advanced Nutrients 6360-12 fertilizer, rich in these nutrients was used.

2.4 Power Subsystem

The goal of the power subsystem is to provide power to the rest of the subsystems. The power subsystem must be able to provide the requisite amount of voltage to each subsystem that requires it and must be able to provide power to multiple devices without overloading.

The initial design was to allow the Grow Light from the light subsystem to receive power from the wall. The design was to use a 120-12 volt power adapter for the solenoid, a voltage regulator to step down the voltage from 12 volts to 5 volts to power the microcontroller, and a second voltage regulator to step down the voltage from 5 volts to 1.8 volts to power the light sensor. However, the first regulator would not have worked. This is because there is a 7V drop and the current drawn for all of the components would have caused it to overheat. In the final design, a separate power supply was used for the solenoid instead to avoid this problem. In addition, a voltage divider was implemented for 3.3V as a new light sensor was being used which required a different voltage. Very little current was drawn, so a voltage divider was acceptable.

2.5 Control Subsystem

The goal of the control system is to connect the sensing system and power through to the microcontroller to allow every component to work together. It will be directly connected to the sensing systems so that if any of these systems have an overabundance or lack of resources, the microcontroller can send commands to give less or more resources respectively. The control subsystem must be able to utilize the capabilities of the microcontroller to send out and read in signals. This will affect the automation of the entire plant care system. The microcontroller must be able to send voltages out to multiple different devices it is connected to.

2.5.1 Microcontroller

The microcontroller takes in the outputs from the other subsystems, performs computations, and sends out signals accordingly (see Appendix D for microcontroller code).

Due to the team's familiarity, the ATMEGA328P was selected. Due to a large amount of measuring signals and sending out signals in response to those, the design required a chip that was capable of having multiple input and output signals. The ATMega328 has a large number of GPIO pins which would suit the needs of the project quite well. The soil moisture sensor and module was built to interact with Arduino, sending a signal ranging from 0 to 5 Volts. Using a chip that is on an Arduino (the Uno) will allow for easier use. Initially, the design used an on-board ATMEGA328P to incorporate into the PCB design. However, since a breadboarded design was used instead, a pin ATMEGA328P was used instead.

2.5.2 Light Subsystem

The BH1750 Lux sensor sends an I2C signal to the microcontroller, indicating light levels. The integral controller then computes the DLI using the lux readings as the input value. When the device is first powered on, the lightbulb will also turn on. Once the DLI reaches required levels, the integral controller will send out a signal to the solid state relay attached to the lightbulb to turn off the lightbulb.

2.5.3 Soil Subsystem

The pH sensor sends a signal ranging from 0-2 volts to the ATMega chip, indicating the pH of the soil. The microcontroller takes the average from 200 readings taken every 0.1 seconds from the pH sensor and calculates the voltage between the two prongs on the pH sensor. The pH value is then calculated from its corresponding voltage value between the two prongs of the sensor. If the pH is outside of the target range, the microcontroller sends an AC current to the speaker, alerting the user.

2.5.4 Moisture Subsystem

The FC-28 sensor sends a voltage signal ranging from 0-5 volts to the microcontroller, indicating soil moisture. If the value read from the moisture sensor is too high, indicating dry soil, the microcontroller sends a 6.5 second signal to the MOSFET, allowing the solenoid to receive 12 volts, causing water to flow. The microcontroller then waits 15 seconds to allow the soil to absorb the water before re-evaluating the soil moisture.

3. Design Verification

In this section, qualitative and quantitative metrics have been included to evaluate the functionality of the project. See Appendix A for a comprehensive list of the requirements and verifications.

3.1 Moisture Subsystem

The first requirement listed in the Moisture Subsystem Requirements and Verification (R&V) table in Table 4 of Appendix A states that the moisture content of the soil must be monitored every 5 seconds. This was verified by filling two separate cups with soil. One cup was left to dry out and the other cup was watered to field capacity. The moisture sensor was placed in the dry soil cup and then placed into the wet soil cup. By checking the serial monitor on Arduino IDE, the sensor outputs a moisture value every 5 seconds, which satisfies this requirement, as seen in Figure 3. The difference between the two values also indicates that the sensor works properly.

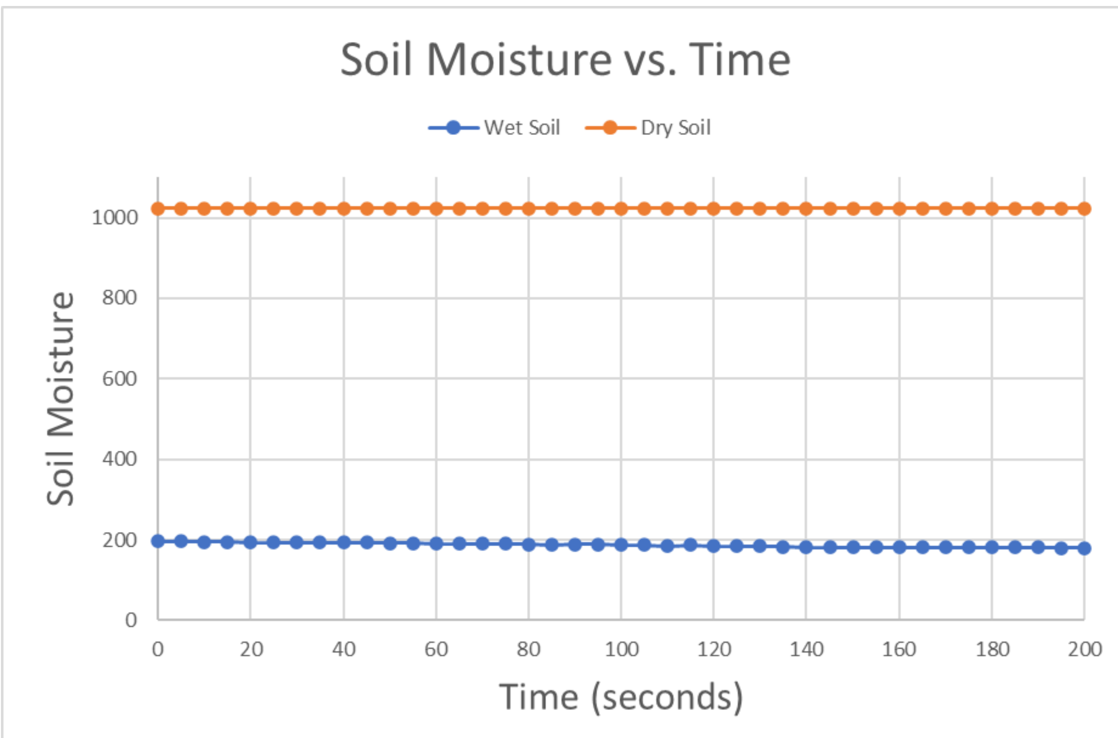

Figure 3 Soil Moisture Sensor Output Values vs. Time

The second requirement from Table 4 states that water must be dispensed with a 6.5 second timed release when the moisture content is low. This was verified by using the same setup as the first requirement, and also connecting the solenoid and filling the water tank. The moisture sensor was placed in the dry cup of soil. The solenoid then opened for 6.5 seconds, allowing water to flow. The moisture sensor was then placed into a wet cup of soil. The solenoid stayed closed, allowing no water to be dispensed. This satisfies the second requirement of the moisture subsystem.

3.2 Light Subsystem

The first requirement listed in the Light Subsystem Requirements and Verification (R&V) table in Table 5 of Appendix A states that the light sensor must be able to accurately measure lux for the system within 1 lumen per square meter. This was verified by placing the light sensor and a lux-measuring app next to each other under the lamp, and comparing the values outputted by the light sensor on the serial monitor in Arduino IDE and the lux app. Initially, the lux values measured by the sensor did not match the lux app within 1 lumen per square meter. This was resolved by adding a constant calibration value to the integral controller of 0.00825. After adding the calibration value, the lux values measured by the sensor and the lux-measuring app were within 1 lumen per square meter.

The second requirement listed from Table 5 state that the integral controller must be able to monitor and control the light duration between 12 and 20 mol·m-2·d-1 of DLI. This was verified by calculating the amount of DLI that would reach the light sensor in 60 seconds, using the specifications from the lightbulb (supplies 50 micromoles per second of photosynthetic photon flux). This was calculated to be 0.0138 mol·m-2·d-1 of DLI. As seen in Figure 4, the DLI calculated by the integral controller reached approximately 0.0138 mol·m-2·d-1 of DLI in 60 seconds.

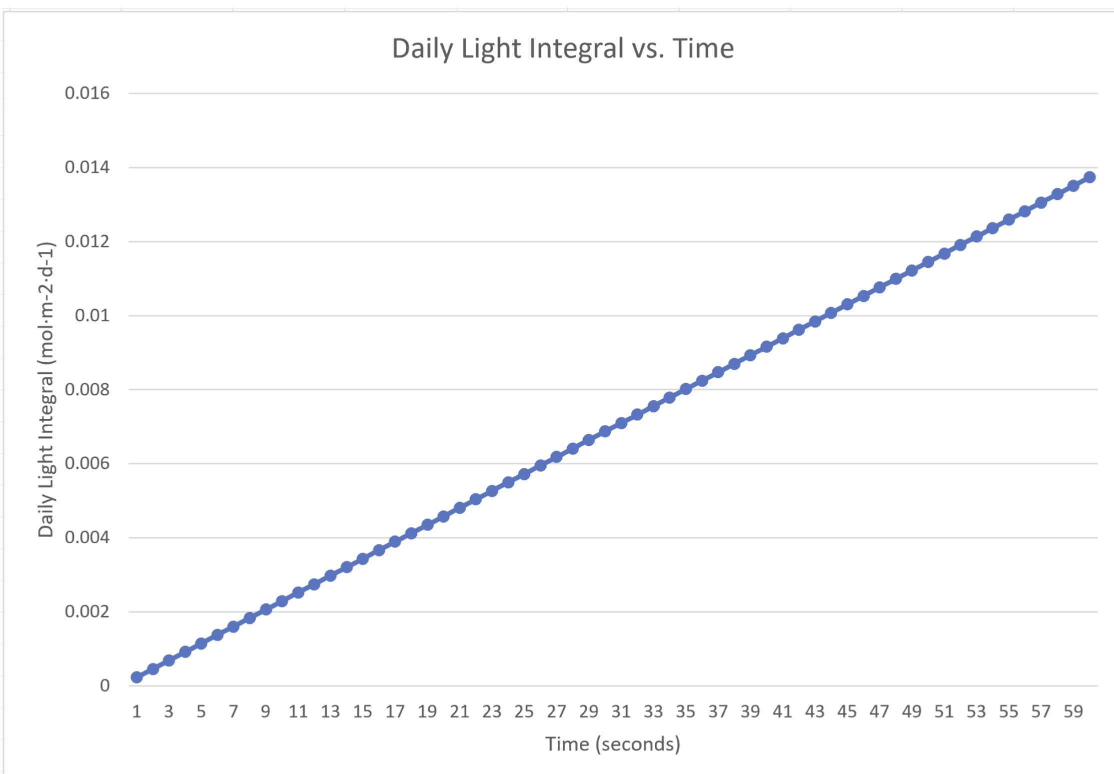

Figure 4 Daily Light Integral vs. Time Calculated by the Integral Controller

3.3 Soil Subsystem

The first requirement listed in the Soil Subsystem Requirements and Verifications (R&V) table in Table 6 of Appendix A states that the pH of the soil must be monitored every minute. The first was to make sure the pH system calibrated quickly once the sensor was placed in the soil. This would ideally be constant, but a goal was set to have this done within one minute. This was verified by placing the pH sensor into a cup of soil and monitoring the output on the serial monitor on Arduino IDE. The pH values become close to constant within 60 seconds, which mostly satisfies this requirement. This requirement wasn't able to be fully fulfilled due to the pH sensor not being able to be in the soil for more than 15-20 minutes at a time. However, once the pH sensor is placed into the soil, the system is able to consistently monitor the pH.

The second requirement from Table 6 states that the user must be notified to add fertilizer once pH levels fall out of the target range of 6.5-7.5. This not only required accurately measuring the pH but also having a signal to output when the pH was not correct. To verify this, tests were done with the sunflower's soil along with a cup of soil filled with lemon juice, which is notably more acidic. The pH would be read on the breadboard and output a signal to emit a sound from a speaker if the pH is out of range, in addition to the visual indicator on the sensor. Figure 5 shows the test results from 5 trials with both sets of soil. For the soil with the lemon juice, the speaker output sound once the pH sensor was placed into the soil. This satisfies the second requirement.

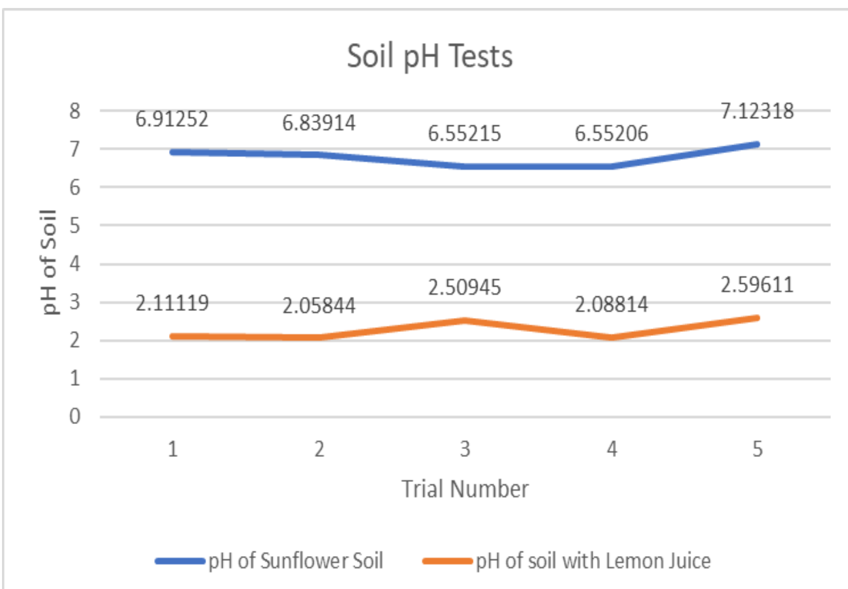

Figure 5 Soil pH tests

There were some slight calibration errors with this subsystem, as noted by Figure 5 since soil pH does not change very quickly over time. While this could cause some error in the results of this system, the output was mostly correct and would be able to help calculate the pH of the soil mostly accurately.

3.4 Control Subsystem

The first requirement listed in the Control Subsystem Requirements and Verification (R&V) table in Table 7 of Appendix A states that the control subsystem must be able to utilize the microcontroller to send out and receive signals from and to different components. This was verified by seeing that the other subsystems work. Since the ATMega chip provided all of the signals for necessary functionality, the control subsystem worked by seeing that the other subsystems worked, which satisfies this requirement.

3.5 Power Subsystem

The first requirement listed in the Power Subsystem Requirements and Verification (R&V) table in Table 8 of Appendix A states that the power subsystem must be able to provide the requirement amount of voltage and current to each subsystem.This requirement was verified by the entire integrating system working, generating the right amount of power to run each part while also not overloading the circuit. The power subsystem used 120V power from a wall plug for the light, ran it through a relay to 5V for many smaller parts, and used a power supply to generate 12V for the solenoid on the water tank, which satisfies this requirement.

4. Cost and Schedule

4.1 Parts

Table 1 Parts Costs

Part	Manufacturer	Design QTY	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
Big Smile Sunflower Seeds	Johnny's Selected Seeds	1	$5.20	N/A	$5.20
GE Grow Light for Plants PAR38 + lamp	General Electric	1	$43.97	N/A	$43.97
AS7341 11-Channel Spectral Color Sensor	AMS	1	$0	N/A	$0
8 inch Plastic Plant Saucer	Vigoro	1	$0.98	N/A	$0.98
Trade 1 Ga. Black Thermoformed Nursery Pot	Trade	1	$1.48	N/A	$1.48
Soil Moisture Sensor and Module	ARCELI	1	$5.99	N/A	$5.99
Microcontroller	Microchip Technology	1	$2.86	N/A	$2.86
1 uF Capacitor	Samsung Electro-Mechanics	1	$0.10	N/A	$0.10
10 kΩ Resistor	YAGEO	1	$0.31	N/A	$0.29
120V AC - 12V DC Converter	Facmogu	1	$10.79	N/A	$10.79
16 MHz clock	Raltron Electronics	1	$0.18	N/A	$0.17
Low Power MAX485 Transceiver Module	Maxim Integrated	5	$0.31	$8.19	$8.19

Table 1 Parts Costs (Continued)

Part	Manufacturer	Design QTY	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
pH Sensor	Sonkir	1	$9.99	N/A	$9.99
Display Module	Solomon Systech	1	$2.24	N/A	$7.59
Small Speaker	Soberton Inc.	1	$2.23	N/A	$2.02
Liquid Fertilizer	Sensi Cal Mag Xtra	1	$11.35	N/A	$11.35
Barrel Power Jack	Wurth Elektronik	1	$1.22	N/A	$1.10
IRF510 MOSFET	Vishay	2	$1.41	$2.82	$1.41
BH1750 Lux Sensor	HiLetgo	5	$1.49	$7.49	$7.49
SSR-40DA	Fotek	1	$12.99	N/A	$12.99
TOTAL COST			$115.09	N/A	$133.96

4.2 Labor

Table 2 Labor Costs

Labor Hours	Average Starting Salary for major (according to Payscale)	Hourly rate based on yearly rate	Estimated total number of hours on project	Total Labor Cost
Charlotte Fondren	$75,097	$36.10	150	$5,415
Thomas Wolf	$80,765	$38.83	150	$5,825
Tom Danielson	$80,765	$38.83	150	$5,825
TOTAL LABOR COST				$17,065

4.3 Schedule

Table 3 Schedule

Week	All	Charlotte Fondren	Tom Danielson	Thomas Wolf
2/27	Design review, Order parts, Safety Training	Design light subsystem	Design PCB	Design PCB
3/6	First Round PCB orders, Teamwork Evaluation 1, Order Parts	Design PCB	Design base	Make modifications to moisture subsystem
3/20	Plant sunflower	Write PID controller	Assemble soil subsystem	Assemble power subsystem
3/27	Individual Progress Reports	Modifications to light subsystem design	Soil sensor testing	Assemble moisture subsystem
4/3	Assembly and Unit Testing	Assemble light subsystem, write integral controller	Soil subsystem testing	Assemble control subsystem, breadboarding microcontroller
4/10	Team Contract Fulfillment	Light subsystem testing	Revisions to PCB design	Moisture subsystem testing
4/17	Mock demos, Conduct full system testing			
4/24	Final demos			
5/1	Final Presentation, Final Papers, Lab Notebook			

5. Conclusion

5.1 Accomplishments

Overall, this project was successful. The moisture and light subsystems not only fulfilled their requirements, but were also able to function simultaneously, indicating the proper functionality of both the power and light subsystems. The soil subsystem was also able to work alongside this, and was able to fit its new requirements, however it did not fit the old requirements that were set, as will be discussed in uncertainties. The indication that each of the resource subsystems worked and were able to function simultaneously means that the power and control subsystems worked as well.

5.2 Uncertainties

The soil subsystem, while operational, did not turn out quite as great as hoped for near the beginning. While the calibration did fit the subsystem goal of being within 1 minute, it was not constant as initially planned, and typically took 20-30 seconds to recalibrate when the soil sensor was in the soil. In addition, its inability to be fully automatic hurts the overall design, as this was one of the initial requirements set for it, and although it doesn't need to be cared for quite as often as the other resource subsystems, there will come a time where its results will be needed and manual input will be required for a system that is otherwise autonomous.

5.3 Ethical considerations

One of the single biggest safety concerns in the use of this project is the lightbulb breaking and cutting the user. Also, the use of a 120V plug introduces the risk of a severe electric shock. Since both of these risks are present in the project, they pose a safety concern to any user and are of concern in regard to IEEE Code of Ethics Section I.1[1]. To minimize the risk of shock, a warning for a shock risk can be placed on the wall plug and the lightbulb can be packed separately with a warning of fragility and the risk of sharp glass.

Also, this product can be misused to grow illegal plants with this device. This would include mostly any plant that could be produced into illegal drugs such as opioid plants or marijuana, depending on the location. While there is not much that can be done to prevent this, the misuse of this product in this manner can be heavily discouraged. Since high voltages are used for the lightbulb, this group took additional safety training as a requirement to safely implement the light bulb. In addition, the utmost caution was taken when integrating the individual subsystems as water was being used

5.4 Future work

There are a few areas in which future work can be done. While this system makes sense for engineers to interact with, it does not really consider the ease of use of the general audience. A way to implement this would be to have an interface in which people could select the desired plant which would select the proper values used in the control code. In addition, the water level of the tank could be monitored and then indicate to the user when it is low. The soil subsystem could also have improved functionality, particularly in its alert system. The sound does not indicate whether the pH is too high or low which could be indicated either by the pitch of the sound or through differently colored LEDs.

References

[1] IEEE. “IEEE Code of Ethics.” *Ieee.org*, June 2020, www.ieee.org/about/corporate/governance/p7-8.html.

[2] A. “How to Test Soil with Arduino and an FC-28 Moisture Sensor: Arduino.” *Maker Pro*, Maker Pro, 21 Feb. 2023, https://maker.pro/arduino/projects/arduino-soil-moisture-sensor.

[3] Admin. “DIY Soil Ph Meter Using Soil Ph Sensor and Arduino.” *How To Electronics*, 21 Aug. 2022, https://how2electronics.com/diy-soil-ph-meter-using-soil-ph-sensor-arduino/#Bill_of_Materials.

[4] *Estimating Soil Moisture - USDA*. The United States Department of Agriculture, https://www.wcc.nrcs.usda.gov/ftpref/wntsc/waterMgt/irrigation/EstimatingSoilMoisture.pdf.

[5] Hasan, Saad. “How to Use FC-28 Soil Moisture Sensor Module with Arduino Uno: DIY.” *Electronics 360*, 12 Feb. 2022, https://www.electronics360.org/how-to-use-fc-28-soil-moisture-sensor-module-with-arduino-uno-diy/#What_is_FC-28_Soil_Moisture_Sensor.

[6] *Moisture Sensor (SKU:sen0114) - Datasheet Hub*. DFRobot, https://datasheethub.com/wp-content/uploads/2022/08/SEN0114_Web.pdf.

[7] Pioneer. “Soil Water Management.” *Soil Water Management | Pioneer® Seeds*, Pioneer, 29 Apr. 2019, https://www.pioneer.com/us/agronomy/soil_water_mgmt.html#AvailableSoilWater_5.

[8] “Section C - UNL Water.” *USDA-NRCS National Engineering Handbook*, The Board of Regents of the University of Nebraska, 1999, https://water.unl.edu/documents/Section%20C.pdf.

[9] Shahid, Shaukat Ali, et al. “Improvement in the Water Retention Characteristics of Sandy Loam Soil Using a Newly Synthesized Poly(Acrylamide-Co-Acrylic Acid)/alznfe2o4 Superabsorbent Hydrogel Nanocomposite Material.” *Molecules (Basel, Switzerland)*, U.S. National Library of Medicine, 3 Aug. 2012, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6268066/.

[10] “Soil Water Storage Properties.” *Soil Water Storage Properties - Minnesota Stormwater Manual*, Minnesota Pollution Control Agency, 15 Dec. 2022, https://stormwater.pca.state.mn.us/index.php/Soil_water_storage_properties.

[11] Kreindler, R Jordan, and Instructables. “Controlling High Power with an Arduino.” *Instructables*, Instructables, 29 Apr. 2021, https://www.instructables.com/Arduino-Tutorial-Handling-High-Power-Devices/.

[12] “PJ-102AH: Digi-Key Electronics.” *Digi*, https://www.digikey.com/en/products/detail/cui-devices/PJ-102AH/408448?s=N4IgTCBcDaIA4CsC0BGADGAhgCxAXQF8g.

[13] “Max485 Module RS-485 TTL Turn to RS485 max485csa Converter Module for Arduino Microcontroller MCU Development Accessories - Integrated Circuits - Aliexpress.” *Aliexpress.com*, https://www.aliexpress.us/item/3256804071797770.html?pdp_npi=2%40dis%21USD%21US+%240.38%21%240.31%21%21%21%21%21%402101c84b16772132418284391ed62c%2112000028548112118%21btf&_t=pvid%3A9e22fc69-39ca-49ce-8ac8-e80727eb2216&afTraceInfo=1005004258112522__pc__pcBridgePPC__xxxxxx__1677213242&spm=a2g0o.ppclist.product.mainProduct&gatewayAdapt=glo2usa&_randl_shipto=US.

[14] “4Pin White and Blue Color SSD1315 0.96 Inch 128x64 Yellow Blue OLED Display Sceen Monitor.” *Temu*, https://www.temu.com/subject/n9/googleshopping-landingpage-a-psurl.html?goods_id=601099513246319&_bg_fs=1&_p_rfs=1&_x_ads_channel=google&_x_ads_sub_channel=shopping&_x_login_type=Google&_x_vst_scene=adg&sku_id=17592195271227&_x_ns_sku_id=17592195271227&_x_gmc_account=647900107&_x_ads_account=5532219654&_x_ads_set=19707106532&_x_ads_id=144854333446&_x_ads_creative_id=648566662998&_x_ns_source=g&_x_ns_gclid=Cj0KCQiAutyfBhCMARIsAMgcRJQ_NRGeOcAZLCknsdFM1xPZS1SzqdYgBxs1EhS1ytQ9eKl-okp7sdUaAhmOEALw_wcB&_x_ns_placement=&_x_ns_match_type=&_x_ns_ad_position=&_x_ns_product_id=17592195271227&_x_ns_wbraid=CjgKCAiAutyfBhBkEigAtM2iJXlbvumhct-tC6ltRhV33EucS-nk2x8YmlnMv24KJ_1__22NGgJhKg&_x_ns_gbraid=0AAAAAo4mICFEE2gPDB85AkZKctUJS6LhN&_x_ns_targetid=pla-1942474457269&gclid=Cj0KCQiAutyfBhCMARIsAMgcRJQ_NRGeOcAZLCknsdFM1xPZS1SzqdYgBxs1EhS1ytQ9eKl-okp7sdUaAhmOEALw_wcB.

[15] “SP-2306-1: Digi-Key Electronics.” *Digi*, https://www.digikey.com/en/products/detail/soberton-inc./SP-2306-1/9924408?utm_adgroup=Speakers&utm_source=google&utm_medium=cpc&utm_campaign=Shopping_Product_Audio+Products_NEW&utm_term=&utm_content=Speakers&gclid=Cj0KCQiAutyfBhCMARIsAMgcRJTMWkF1enGtQT7yDRvVhwBOfUUKUbqnnkgeAKqvayuiwX-m9ggkCWwaAgl9EALw_wcB.

[16] “Sonkir Soil Ph Meter, MS02 3-in-1 Soil Moisture/Light/Ph …”, *Amazon.co,* https://www.amazon.com/Sonkir-Soil-Moisture-Tester-Gardening/dp/B07BR52P26.

[17] “Expert Gardener Flower Plant Food, 10-10-10 Fertilizer, 4 Lb.” *Walmart.com*, https://www.walmart.com/ip/Expert-Gardener-Flower-Plant-Food-10-10-10-Fertilizer-4-lb/1192421211?wmlspartner=wlpa&selectedSellerId=0&wl13=1548&adid=22222222227000000000&wl0=&wl1=g&wl2=c&wl3=42423897272&wl4=pla-51320962143&wl5=9022196&wl6=&wl7=&wl8=&wl9=pla&wl10=8175035&wl11=local&wl12=1192421211&veh=sem&gclid=Cj0KCQiAutyfBhCMARIsAMgcRJQpPpSQ1P_iee7iCdE5sSJr4Kq_BCAhU_95HHMzjmnz4kPSmN2rIG0aAlBVEALw_wcB.

Appendix A Requirement and Verification Table

Table 4 Moisture Subsystem Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. The moisture content of the soil must be monitored every 5 seconds.	1. Connect the moisture sensor and the MOSFET to an Arduino Uno (used for testing purposes). Connect the solenoid to the MOSFET. Provide 12 volts to the MOSFET. Load the Arduino with the moisture code. Verify on the serial monitor that the sensor outputs a moisture value every 5 seconds.	Yes
2. Water must be dispensed with a 6.5 second timed release when the moisture content is low.	2. Connect the moisture sensor and the MOSFET to an Arduino Uno. Connect the solenoid to the MOSFET. Provide 12 volts to the MOSFET. Load the Arduino with the moisture code. Place the sensor in a cup of dry soil. Verify that the solenoid opens and releases water for 6.5 seconds.	Yes

Table 5 Light Subsystem Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
3. The light sensor must be able to accurately measure lux for the system within 1 lumen per square meter.	3. Connect the light sensor to the Arduino Uno using I2C. Place a phone front-facing camera-up next to the light sensor with the lux-measuring app loaded. Load the Arduino with the light sensor code. Verify that the lux values output to the serial monitor are within 1 lumen per square meter as the values given on the lux-measuring app.	Yes
4. The integral controller must be able to monitor and control the light duration between 12 and 20 mol·m-2·d-1 of DLI. (Use 0.016 mol·m-2·d-1 of DLI for testing, equivalent to 60 seconds under the lamp).	4. Connect the light sensor to the Arduino using I2C. Connect the solid state relay to the Arduino. Load the Arduino with the integral controller code. Start a timer. Verify that the lightbulb turns off after 60 seconds.	Yes

Table 6 Soil Subsystem Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
5. The pH of the soil must be monitored every minute.	5. Connect the pH sensor to the Arduino Uno. Load the Arduino with the pH code. Place the pH sensor into a cup of soil. Verify that pH values output to the serial monitor and become constant within 60 seconds.	No (pH sensor must be taken out of the soil after 15 minutes)
6. Users must be notified to add fertilizer once pH levels are outside the target range of 6.5-7.5.	6. Connect the pH sensor and the speaker to the Arduino Uno. Load the Arduino with the pH code. Place the pH sensor into a cup of soil with lemon juice (very acidic mixture). Verify that the speaker makes a sound.	Yes

Table 7 Control Subsystem Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
7. The control system must be able to utilize the microcontroller to send out and receive signals from and to different components.	7. Connect the light, moisture, soil, and power subsystems to the microcontroller. Load the microcontroller with the integrated system code. a. Place the moisture sensor into a dry cup of soil. Verify that the solenoid opens and releases water. b. Place the light sensor under the lamp. Verify that the lightbulb turns off after 60 seconds. c. Place the pH sensor in a cup of lemon juice soil (very acidic mixture). Verify that the speaker makes a sound.	Yes

Table 8 Power Subsystem Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
8. The power subsystem must be able to provide the required amount of voltage and current to each subsystem.	8. Connect the light, moisture, soil, and power subsystems to the microcontroller. Load the microcontroller with the integrated system code. a. Place the moisture sensor into a dry cup of soil. Verify with a multimeter that 12 volts are being supplied to the solenoid. b. Place the light sensor under the lamp. Verify with a multimeter that 5 volts are being supplied to the input side of the solid state relay and 3.3 volts are being supplied to the light sensor. c. Place the pH sensor in a cup of lemon juice soil (very acidic mixture). Verify with a multimeter that an AC current is being supplied to the speaker. d. Verify with a multimeter that 5 volts are being supplied to the microcontroller.	Yes

Appendix B Initial PCB Design

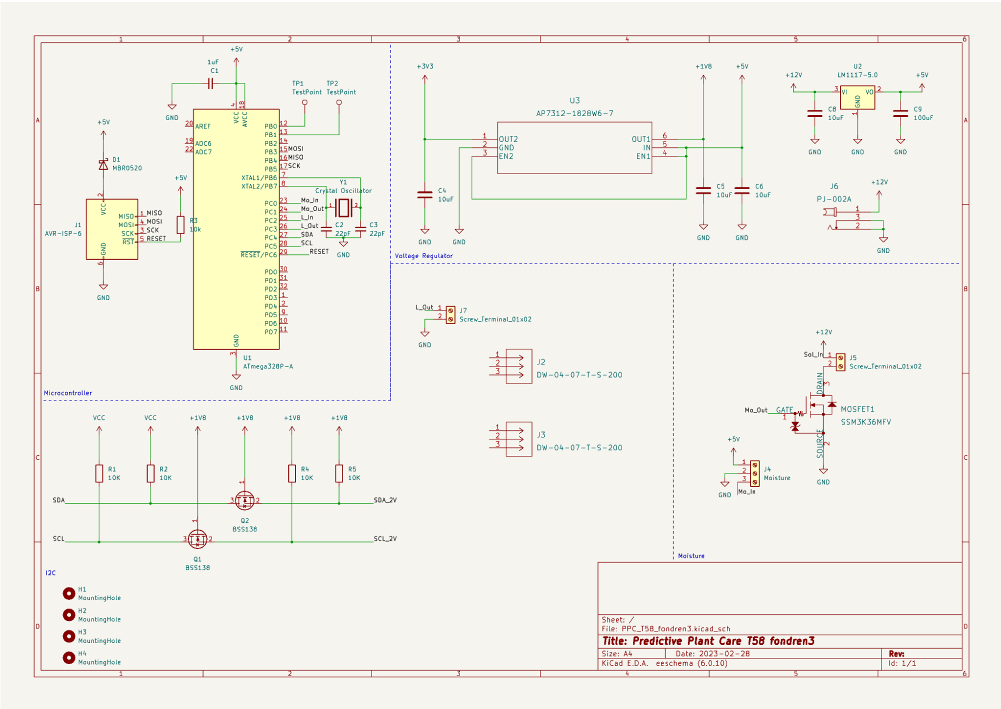

Figure 6 Initial main circuit design

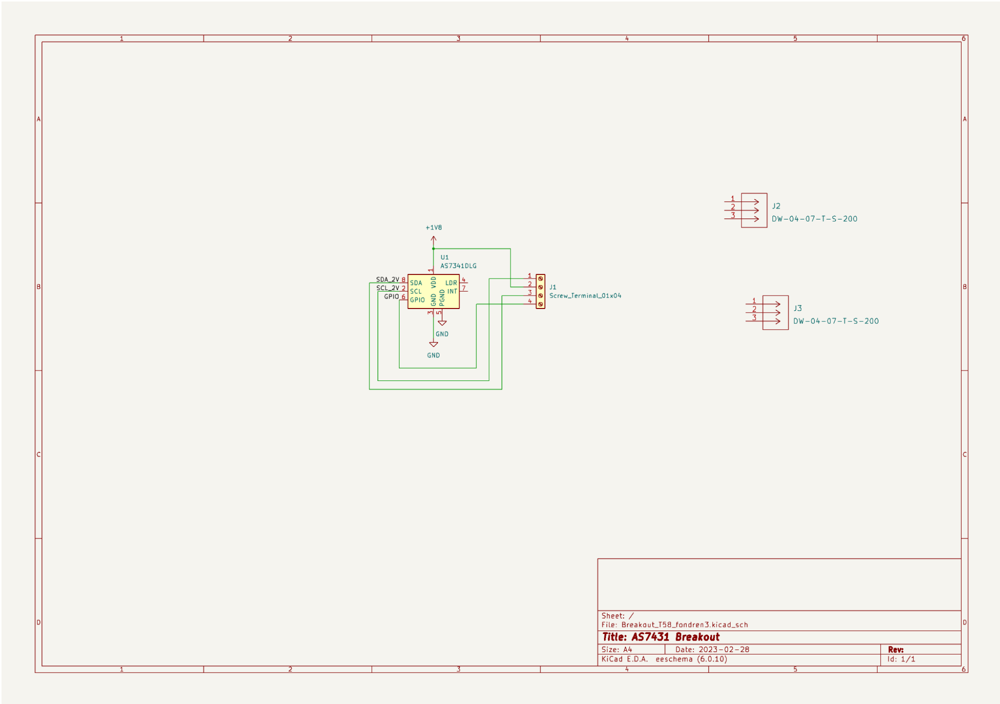

Figure 7 Initial breakout circuit design

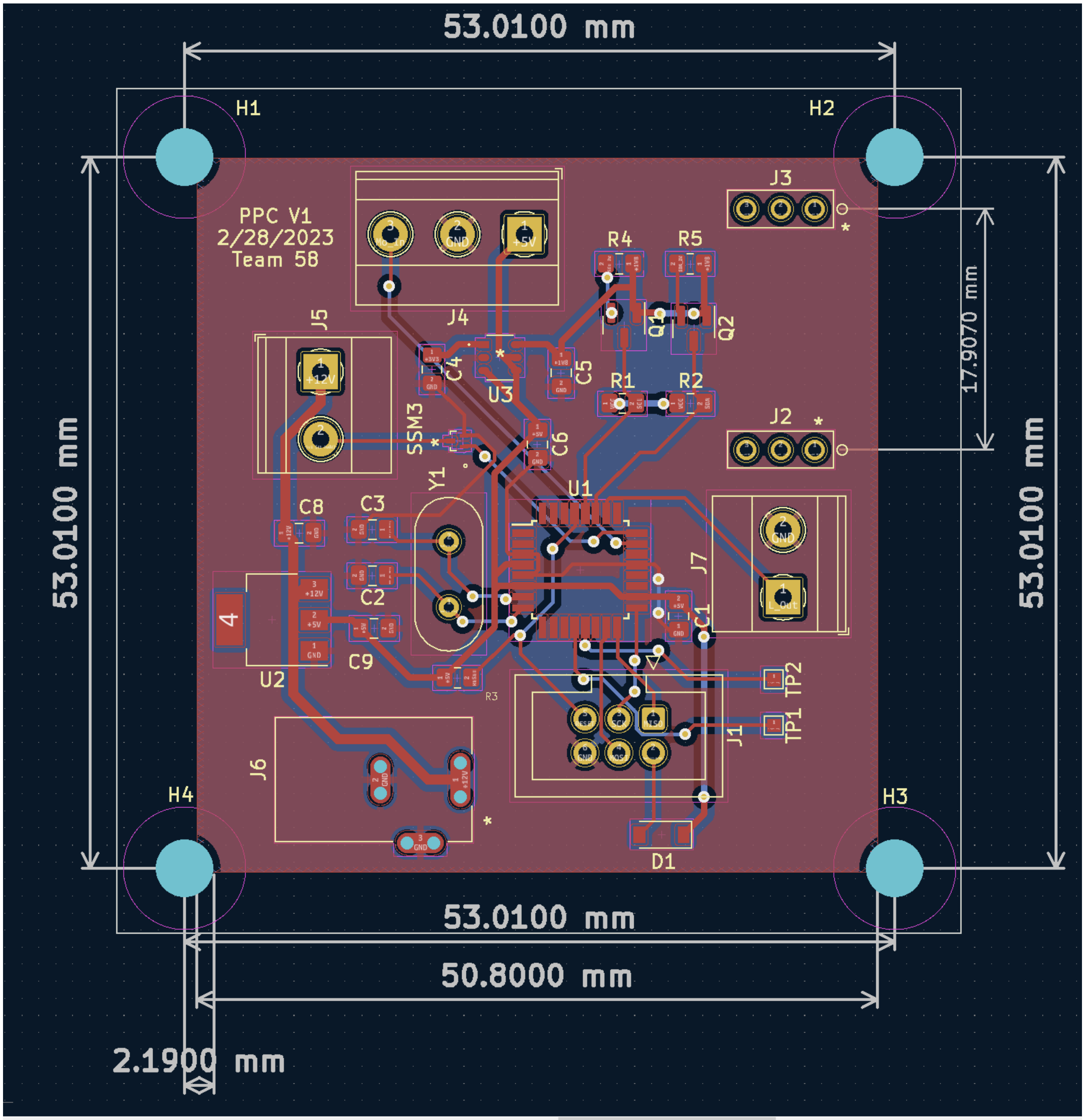

Figure 8 Initial main PCB design

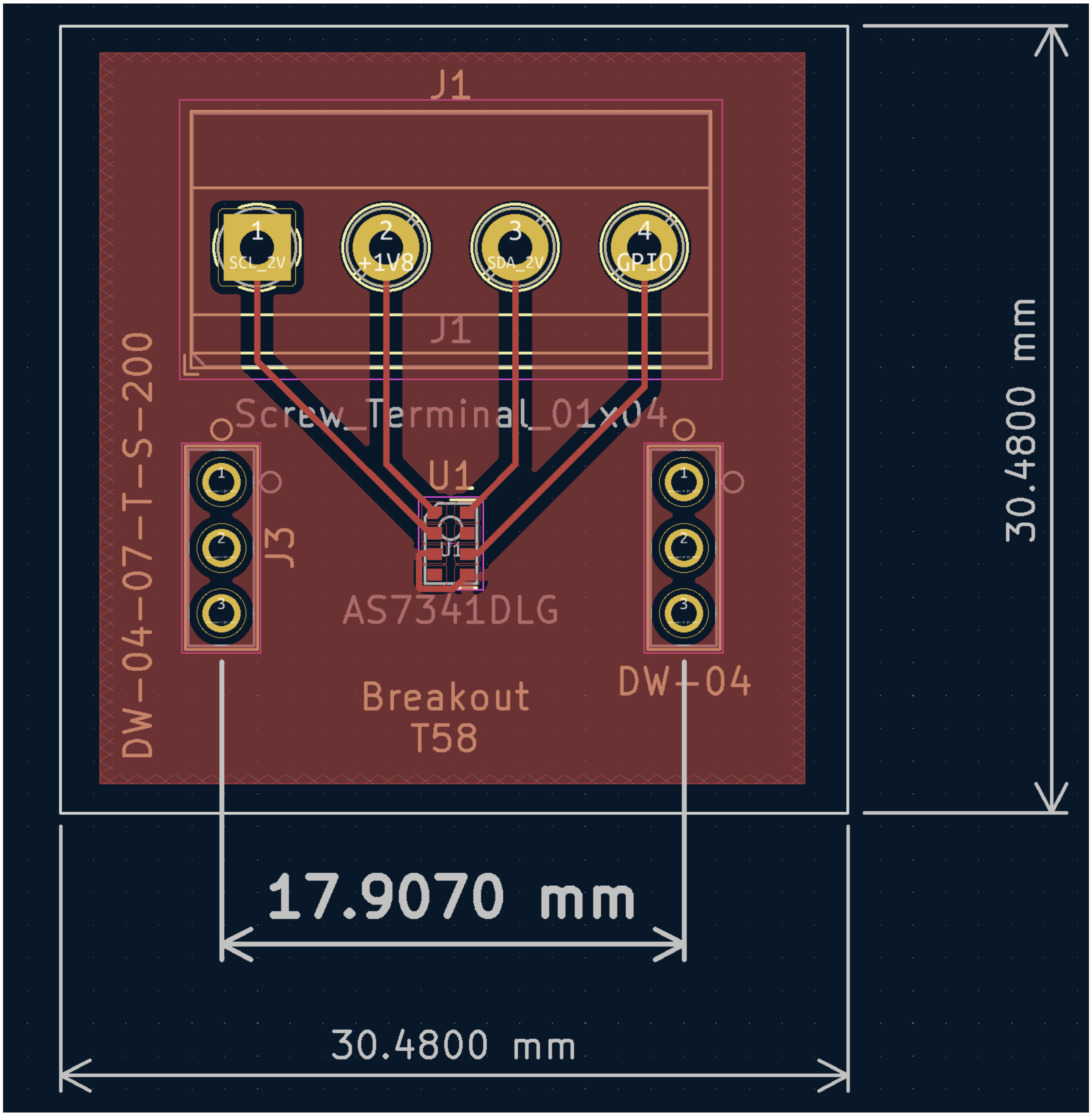

Figure 9 Initial breakout PCB design

Appendix C Revised PCB Design

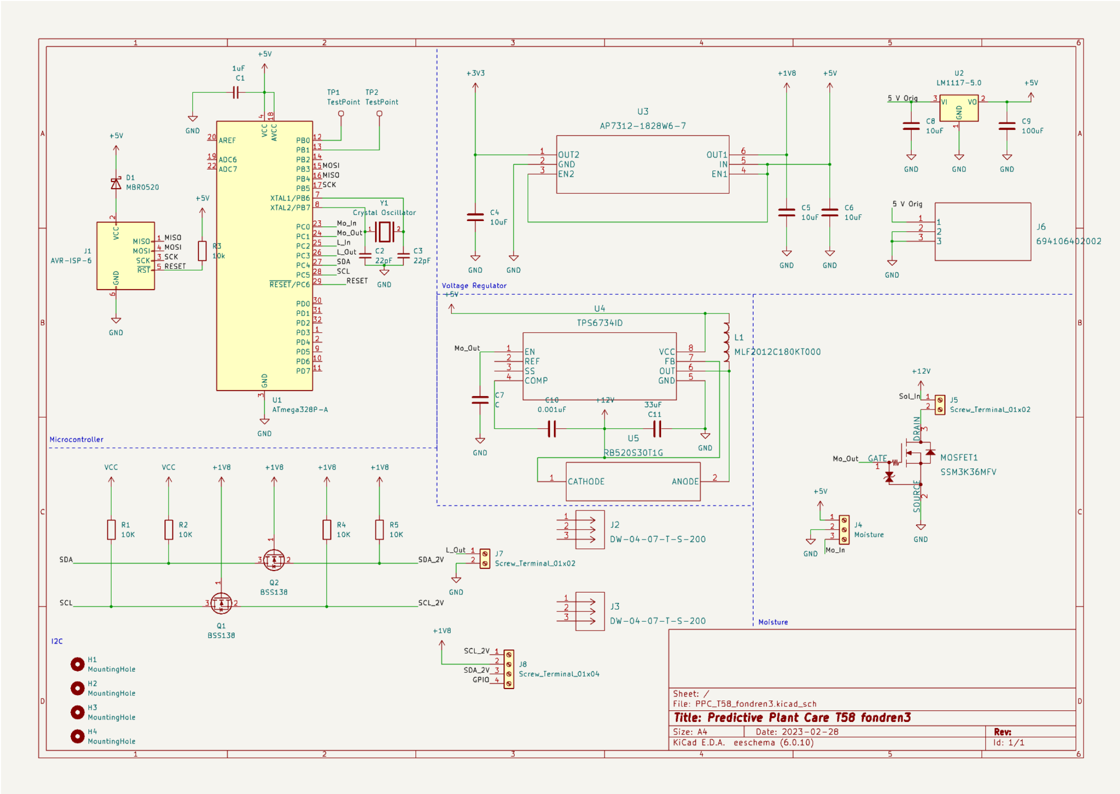

Figure 10 Final main circuit design

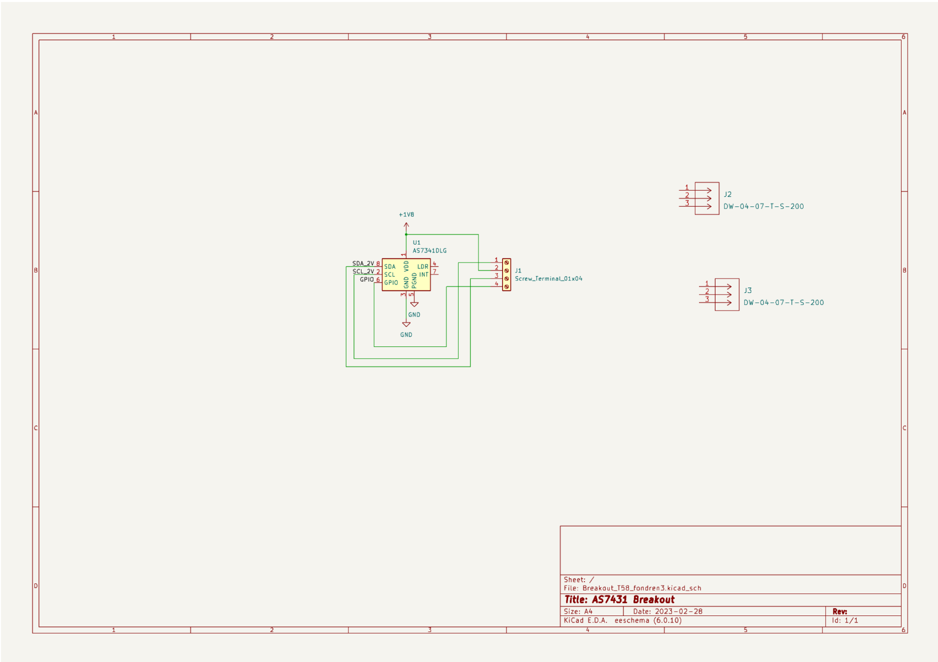

Figure 11 Final breakout circuit design

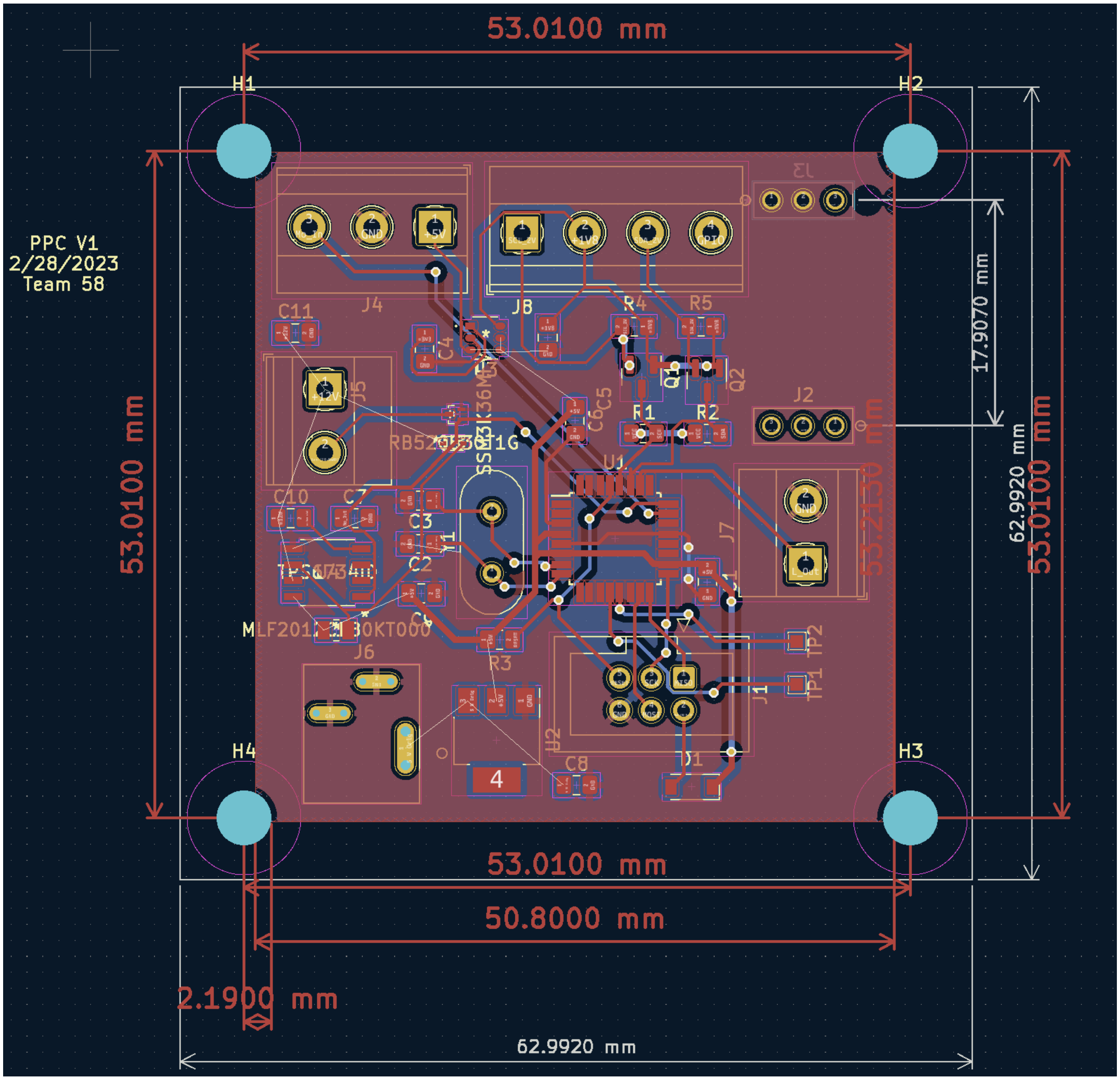

Figure 12 Final main PCB design

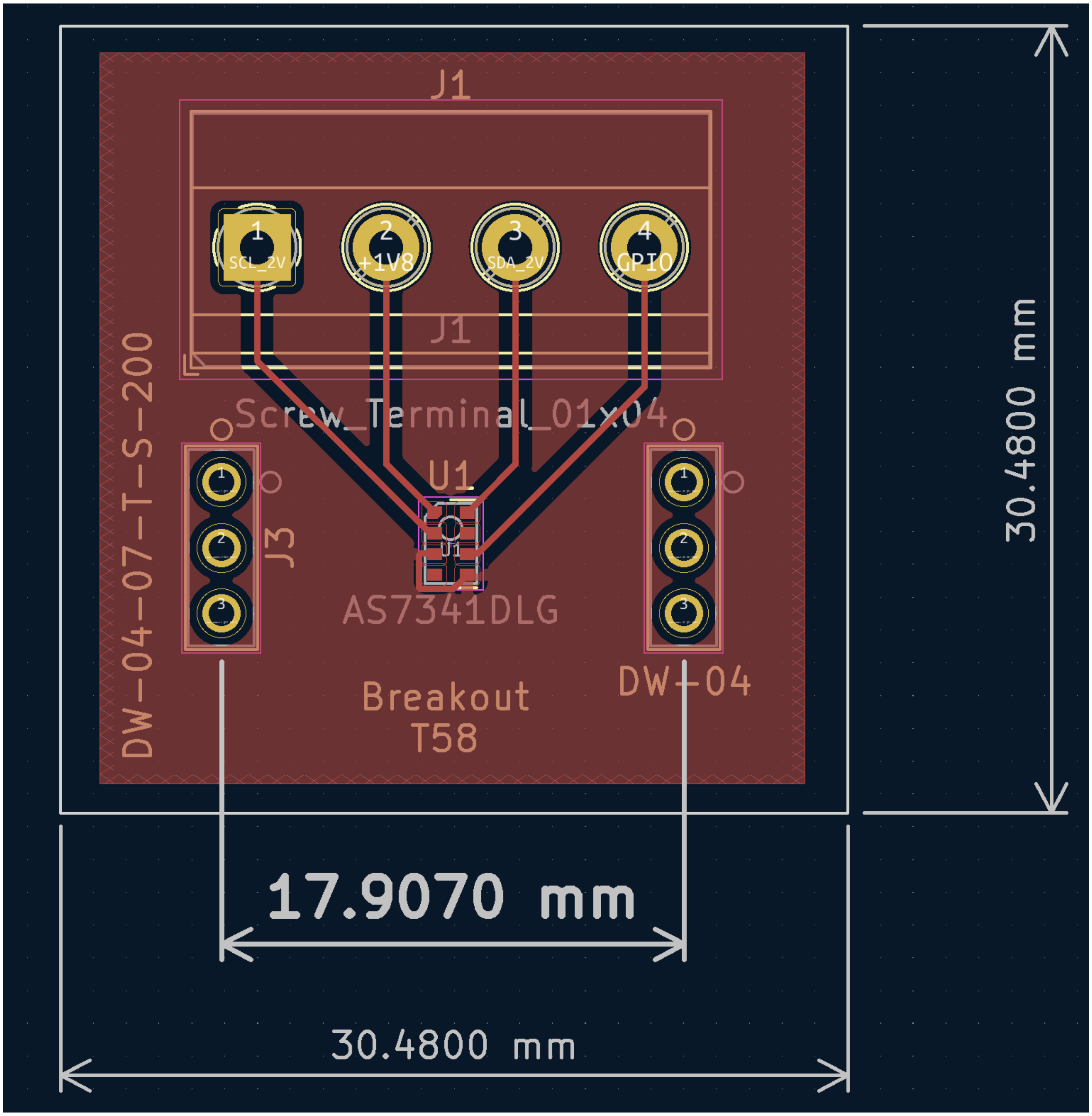

Figure 13 Final breakout PCB design

Appendix D Microcontroller Code

```
#include <Wire.h>
#include <BH1750.h>
#include <math.h>

BH1750 GY302;      // initialize BH1750 object
int PowerPin = 10;  // pin to go to relay
int on = 1;
float lighthrs;
float seconds;
float dli;
float time;

int msensor = A0;
int mout = PD7;
int msvalue = 0;
int cooldown = 0;

// number of analog samples to take per reading
#define NUM_SAMPLES 200

int sum = 0;                    // sum of samples taken
unsigned char sample_count = 0; // current sample number
float voltage = 0.0;            // calculated voltagef
float ph;
// int soundPin = 9;
int acPin = PD6; // digital pin 7 - now 6
int virgrd = PB0; // digital pin 8
double val = 0 ;
int freq = 50;
double t = 0;
const double pi = 3.1415;
const double fs=1000;
int sound;

void setup()
{
  Serial.begin(9600);
  Wire.begin();
  GY302.begin();  // initialize GY-302 module
```

```
  pinMode(PowerPin, OUTPUT);
  pinMode(msensor, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(mout, OUTPUT);
  pinMode(acPin, OUTPUT);

  seconds = 0;
  on = 1;
  lighthrs = 0;
  dli = 0;

}

void loop()
{
  // get reading from module
  // integral controller
  float lux = GY302.readLightLevel();
  float error signal = lux * 0.0165 / 2;        // multiply lux by the conversion factor to get error signal,
calibrated
  time = time / 3600;
  dli += error signal * lighthrs * 0.0036;  // dli calculation
  time = 0;
  if (dli >= 16) {
    on = 0;  // turns the light off when dli gets high enough
  }
  if (on == 1) {
    digitalWrite(PowerPin, HIGH); // if on = 1, turn on relay
  }
  else {
    digitalWrite(PowerPin, LOW); // if on = 0, turn off relay
  }

  while (sample_count < NUM_SAMPLES) {
       sum += analogRead(A2); // analog pin 2
       sample_count++;
       delay(10);
    }
  time = time + 2;
  seconds = seconds + 2;

  voltage = ((float)sum / (float)NUM_SAMPLES * 5.015) / 1024.0;
```

```
// send voltage for display on Serial Monitor
// voltage multiplied by 11 when using voltage divider that
// divides by 11. 11.132 is the calibrated voltage divide
// value
voltage = 250 - (voltage * 11.132 * 10000);
ph = 7 - voltage/ 57.14;
Serial.print("ph: ");
Serial.println(ph);
if (ph <= 3) { // if ph is too low
 // digitalWrite(soundPin, HIGH); // alert user..
 float starttime = millis();
 // float endtime = starttime;
 while ((millis() - starttime) <=5000) {
  t = millis();
  val = 127*sin(2*pi*(freq/fs)*t)+127;
  analogWrite(acPin,val);
  analogWrite(virgrd,127);
  sum = analogRead(A2);
 }
 time = time + 5;
 seconds = seconds + 5;
}
else if (ph >= 9) { // if ph is too high
 // digitalWrite(soundPin, HIGH); // alert user
 float starttime = millis();
 // float endtime = starttime;
 while ((millis() - starttime) <=5000) {
  t = millis();
  val = 127*sin(2*pi*(freq/fs)*t)+127;
  analogWrite(acPin,val);
  analogWrite(virgrd,127);
  sum = analogRead(A2);
 }
 time = time + 5;
 seconds = seconds + 5;
}
else {
 analogWrite(acPin,LOW);
 analogWrite(virgrd,LOW);
}
```

```
  sample_count = 0;
  sum = 0;

  msvalue = analogRead(msensor); // read value on moisture sensor

  if ( ( msvalue >= 459 ) && ( cooldown == 0 ) ) {// 10.5% lower bound
    digitalWrite(LED_BUILTIN, HIGH); // open solenoid
    digitalWrite(mout, HIGH); // open solenoid
    delay(6500); // open solenoid for 6.5 seconds, should get it from 10.5% to 11.5%
    seconds = seconds + 6.5;
    time = time + 6.5;
    digitalWrite(LED_BUILTIN, LOW); // once at optimal point, close solenoid
    digitalWrite(mout, LOW);
    msvalue = analogRead(msensor); // reread value on moisture sensor
    cooldown = 3;
  }
  else { // wait for soil to dry out
    digitalWrite(LED_BUILTIN, LOW); // solenoid closed
    digitalWrite(mout, LOW); // solenoid closed
    delay(5000);
    seconds = seconds + 5;
    time = time + 5;
    if ( cooldown > 0 ) {
    cooldown--;
    }
  }

  lighthrs = seconds / 3600;
  if (lighthrs >= 24) {  // reset every day
    lighthrs = 0;
    seconds = 0;
  }
}
```