SOME ASSEMBLY REQUIRED: AUTONOMOUS PLANT NURSERY

A Research Paper submitted to the Department of Electrical and Computer Engineering
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Electrical Engineering

By

Ryland Buck

March 30, 2023

Technical Team Members:

Joshua Garrison

Boluwatife Raufu

Joseph Sam

Mckayla Thomas

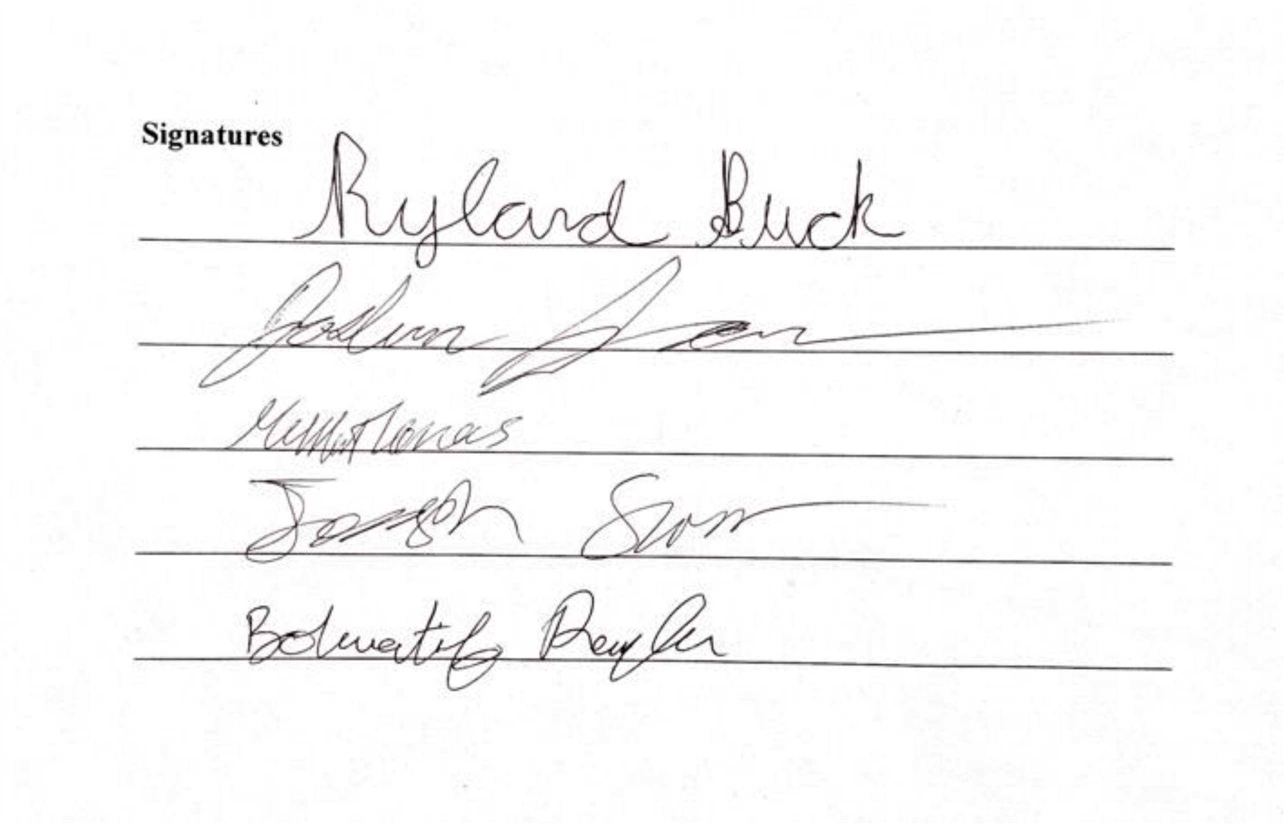
ADVISOR
Harry C. Powell Jr., Department of Electrical and Computer Engineering

Some Assembly Required

Joseph Sam, Ryland Buck, Joshua Garrison, Mckayla Thomas, Boluwatife Raufu

December 13, 2022

Capstone Design ECE 4440 / ECE4991



Signatures

Ryland Buck

[illegible]

[illegible]

[illegible]

[illegible]

Statement of Work:

Ryland Buck

Ryland's work on the Autonomous Plant Nursery pertained mainly to working with Joseph on the development and testing of the project printed circuit board (PCB) and the designing of the chassis. He helped design and implement all subsystems on the board, and helped test these systems once the board was completed and printed. For the power management subsystem used to ensure power delivery to all subsections, he researched the 3.3V, 5V, and 12V voltage regulators needed to power the subsystems of the board by examining datasheets for output voltage and current applicable to our design, and helped Joseph test the performance of these parts by connecting a digital multimeter across the regulator using jumpers soldered to the board. For the water pump interface that delivered water to the plots of the device, he spoke with Joshua about the motor driver chip and used the datasheet in order to quantify the resistors and capacitors needed to get the chip to run, and during testing he and Joseph used test pins to test the input and output voltages to the motor driver chips. For the light interface to turn on lights for the plant plots, he assisted Joseph in researching for the relays used to turn the lights on and off according to the 3.3V logic signal sent from the microcontroller, and during the testing of the board he used the pins allocated for voltage testing to ensure the voltage was applied to the correct light. For the moisture sensor interface to read moisture values from the plant plots, he assisted Joseph in finding the relay and integrated circuit chip that could read one plot at a time, and process the values read by the soil sensors, respectively. by referencing datasheets for components with the desired functionality and operating parameters, and during testing he ensured the subsystem was properly set up before any data was applied through the system. For the client interface for the user to select and edit plant growth parameters, he integrated the screen and pushbuttons into the design, and helped Joseph with designing the pushbutton subcircuits and routing each of the applicable pins of the LCD screen to the correct pin on the microcontroller.

Ryland's work on the Autonomous Plant Nursery also involved the chassis design for the device. He designed a proof-of-concept of the chassis for the project that provided a baseline of how to design the chassis, he explored several iterations of the chassis design with Joshua, and helped coordinate the logistics and delivery of the final design of the chassis with Joshua as well.

Joshua Garrison

Josh worked on a large amount of the code that was contributed. He developed the Finite State Machines that were used to deliver the water and nutrients as well as the one that was used to properly navigate the LCD options. He also developed the logic to use the LCD, and wrote the interrupts that control the timers that operate the lights and the intervals that operate the pumps. His final contribution to the code was setting up all of the general-purpose input and output pins that allowed the microcontroller to communicate with the rest of the system. He also designed test plans to test the GPIO using an adapter board to make sure that a signal was being received when the system was high, and that there was no signal when the system was low. When he had set up the clocks and timer interrupts, he designed the test plan to select an option to turn the light on for one minute, five minutes, and twenty minutes and timed the light to make sure that the clocks and timer interrupts were working as expected. He also designed a way to test the push button interrupts and LCD screen before the board arrived so that all of the code except the inter-integrated circuit I2C could be verified to work as expected early on in the project construction.

Josh also built the schematic for the pumps and designed this section of the circuit including connection to input 5V output to motor, and all necessary resistor and capacitor values that accompanied this section. He also found the motor controllers and the water pumps that were used in the design. The motor controllers did not get used in the final design due to issues setting we found with the motor drivers in our setup during testing.

Josh also helped design the original design of the chassis. When the final design was decided and agreed on by the entire group Josh actually constructed the final chassis that was used in the final demonstration.

Mckayla Thomas

Mckayla mainly contributed to the I2C software implementation. This task consisted of various tasks including extensive research about generic I2C processes and interpretation of the AD5933 data sheet. Mckayla began her task by developing a Start function that was responsible for sending the start sequence that would allow the SDA (data line) to change when the SCK (clock line) is high. Then Mckayla began developing the Stop sequence which was responsible for setting both clock and SDA low at the end of the transaction with the slave device. Following this she worked on the SendByte function which sends sending information bit by bit and waiting process for the acknowledge bit delivered by the slave device. These functions were tested using a network analyzer on the virtual bench, she was able to see the correct bits were being sent.

Following the implementation of the functions mentioned above, she began developing the initialization function for the impedance chip. The purpose of this function was to send the start frequency, frequency increment, number of increments, number of time cycles, and control to the correct registers and values. This function was also tested using a network analyzer which displayed the correct register and values were being sent. Following this Mckayla began creating the function that would handle the complex number delivered by the impedance chip.

The chip delivered impedance using a real and imaginary register specified by the datasheet. Each real and imaginary data was sent in sections (upper and lower) which correspond to 4 separate registers. She designed a function that would take the complete real and imaginary numbers and compute them into one impedance number. Next, she worked on the read functionality of the registers.

Developing the read function was difficult because she could not develop with the chip because the chip was too small to test on a breadboard so she had to wait for the completion of the PCB. But she still developed the function by using existing read solutions. The function would send the start sequence and then send the slave address with a read bit then SDA was set to read. Then the function would read the byte response bit by then combine then into a singular number then send a no to acknowledge bit and stop the sequence.

Unfortunately, once the PCB was complete, she realized the impedance chip was not sending the response bits. Throughout the process, she consulted professors who were familiar with I2C and was under the impression that generic I2C would be sufficient for the chip but realized generic I2C was not fit to handle the communication process of the chip. She also assisted in the other aspects of the development process including the development of the water pump FSM, chassis design, and power supply testing.

Joseph Sam

Joseph Sam contributed to the design of the Printed Circuit Board, the subsystem testing procedure, and the I2C implementation.

For his contributions to the Printed Circuit Board, Joseph designed the subsystems of Power Management, Light Interface, Moisture Interface, and Client Interface. The task of implementing the power management subsystem involved calculating the total power draw of the entire system to ensure the AC/DC transformer was appropriately sized, selecting regulators capable of producing voltage supplies at the correct levels of 3.3, 5, and 12V. The task of implementing the Light Interface involved calculating and designing a relay switch circuit that can take inputs from the MSP430 to drive a relay that is capable of connecting the lights to the 24V power supply line to turn them on. The design of the moisture interface involved the usage of an AD5933 impedance converter. The design of the circuitry around this device incorporated designing a calibration circuit and selecting a feedback resistor for the internal transimpedance amplifier

In assisting with the I2C implementation, Joseph helped Mckayla understand the information presented within the datasheet of the AD5933 for figuring out how to perform the various I2C transactions that would be needed. Joseph assisted in setting up the testing apparatus to monitor the data and clock lines and interpret the results.

Lastly Joseph created the testing plan for all of the hardware components, detailing exactly how to systematically go through each subsystem safely and ensure that they are performing as expected.

Boluwatife Raufu

Bolu contributed to the research of materials and the coding standard revision. He had modeled the overall design for the chassis of the project, making it easier to visualize the placements of the hardware and components. As progress was made on the project the design did receive revisions, due to realizing the scale and difficulty that came with the idea, or out of efficiency. The revisions were made in conjunction with Ryland and Joshua, assisting them in the construction of the physical chassis, this includes the light and water pump installation. Research on the lights was done with the assistance of Ryland. Wire stripping was done in order to make the wire fit into the PCB. Bolu also assisted with the multiple testing that took place, this included the I2C and water pump connection to the PCB.

Contents

Table of Figures

Abstract

The autonomous plant growing environment is a system to automate the process of growing juvenile plants. The project supplied the plant with required water, nutrients, and light. Once the plant is placed into the device, the user is allowed to set the conditions needed for the plant. If the plant needs less water and longer periods of light then the user can adjust accordingly. This project is useful for people who travel or just struggle with caring for their plants. The control unit of the design is the MSPEXP430FR2355 microcontroller [1]. The MSP is configured for specific input/out requirements for the functionality that will track the Ultra Violet (UV) light and moisture levels inside the soil. The moisture sensors are used to monitor the content of the soil which will cause the MSP to determine the distribution of appropriate amounts of water, nutrients, and light. The outputs of MSP connected to the water pumps. For each water pump, there is one tube that goes to the plant and another goes to the one of the reservoirs. The Serial Peripheral Interface (SPI) Liquid Crystal Display (LCD) is also connected to the MSP to allow the user to interface with the software. The LCDs displays current readings from the MSP, options for water, and lighting options. Throughout the construction of the project, the team learned a variety of knowledge. The project met some of the expectations and partially performed. [1]

Background

The idea for our project came from a friend of Josh. He was growing separate plants in different plots that all had different lighting and watering requirements for optimized growth. When the group was discussing possible projects, the group decided that an autonomous plant growing environment is something many people could benefit from. In addition, Mckayla has a history of growing many types of plants so this assisted in the design acknowledging certain specifications that could be essential for overall plant health. The idea for the project is that the device would allow all users to grow different kinds of plants in separate plots at the same time. Since different plants need different amounts of water and sunlight, it was important that we give the user some way to manage how much water and light each plot received. In addition, we wanted to provide the user with a device that would continuously monitor the moisture levels and automatically supply the needed water. Our model was on a smaller scale so it was built for only two plots, but the project could be expanded to be used on a larger scale in the agricultural industry.

There have been projects that have similar functionality to our project. These devices contained a lighting system, irrigation system, and different sensors that obtained information regarding the plant and relayed the information to the user. Their design also contained control that would automatically adjust the lights and watering systems to optimize the growth of the plant [2]. Our project did contain slight differences because our project is designed to allow people to grow different types of plants with varying amounts of the necessary resources at the same time by having separate plots which the user can control and monitor. Our device focuses on the growth process for in house use. So our delivery system was implemented differently with an emphasis on delivering a set amount of light and the light exposure that was adjusted according to time.

Another system similar to our Autonomous plant nursery is the Autonomously controlled greenhouse cultivation system. This system has an autonomously controlled greenhouse which reads the environment condition via sensors. These conditions include illumination, temperature, humidity, carbon dioxide, and water supply [3]. The main difference between our project and this project is that our project allows for outside elements to have an effect and focuses on wanting to control the environment around the system while their project is made for greenhouses which completely isolates the plants. This means the environment is completely controlled.

Our project required multiple working components and subsystems that we have used or utilized in previous courses. In order to allow the user to choose how each plot will be managed we had to set up a microcontroller that communicates with multiple sensors that will be used which connected to the Printed Circuit Boards (PCB) that was responsible for powering the lights and water pumps as well the motors that will choose which plot the water will be delivered to.

Due to previous exposure to microcontrollers and how to communicate with additional boards with serial peripheral interface (SPI) and inter-integrated circuit(I2C). In the Embedded Computer Systems (ECE 3430), we got experience with rotary encoders and LCD screens that allow the user to have a visual option to communicate with the microcontroller. This class used Code Composer Studio (CCS) as an IDE (Integrated Development Environment) which we used for all of the programming that we did on the microcontroller. [1]

In addition to Embedded Computer Systems (ECE 3430), everyone in the group took the electric fundamentals courses, ECE Fundamentals 1 ECE 2630, ECE Fundamentals 2 ECE 2660, and ECE Fundamentals 3 ECE 3750. These courses prepared us by teaching us how to interpret datasheets and practice with designing and building PCBs for complex projects. This knowledge was vital to designing the power supply to the water pumps as well as for the UV lights for the plants.

Ryland and Josh were exposed to the utilization of temperature sensors from the Embedded Computing Robotics ECE 3501 course. In the course they used sensors to control a robot. This particular set of knowledge was useful for properly installing and coding the moisture sensing implementation.

In many of our courses we used Finite State Machines (FSMs) but mainly we got a deep understanding from the Digital Logic Design CS 2330 course. We used Finite State Machines when developing the control of the water pumps and setting thresholds for moisture levels. For instance, we wanted the user to set a desired moisture. So, the system would deliver water until the soil moisture was at that specified level then stopped. The program was designed to be in a water supply state until the level is reached then goes into a hold state. FSM are ideal for applications like this.

Physical Constraints

Design Constraints

There was an assortment of design constraints that were recognized for this project. In terms of the physical constraints, the Autonomous Plant Nursery had to be large enough to accommodate a variety of plants comfortably but small enough to be transportable by a maximum of two people, and quickly adapt to changing parameters, such as moisture level. and easily manipulable by the user. The plant growing parameters, water/nutrient delivery, and lighting had to work in sync with the program running on the MSP so as not to have delivery times and quantities uncharacteristic of the values specified by the user.[1] Additional constraints pertaining to the design of the project included the LCD screen, which possessed a limit of displaying information on a total of four lines with a limit of twenty characters per line. Since spaces counted as characters for this LCD screen, we were constrained into developing a User Interface for the LCD screen that could display all of the necessary information for the user without running out of characters on any particular line. In addition, the LCD screen did not provide seamless feedback with the microcontroller we used; therefore, we were constrained to having the user select parameters from a select number of options as opposed to allowing custom definitions for these parameters to be created.

Additional logistical constraints included the ordering of parts and PCB verification and production. Some of the parts used in the final PCB design, such as the APAN3103 relays, were highly specialized parts that required more time for researching for parameters due to limited quantities of availability, verification of intended operation, and ordering from vendors such as Digikey. In addition, the time needed to have the fully-designed PCB verified and made by 3W Electronics further hindered progress on the project since every subsystem utilized the same single PCB board.

Cost Constraints

Most of the components used for the Autonomous Plant Nursery were not readily available, and some components were costly due to their scarcity. Some of the materials used for the Autonomous Plant Nursery, such as two of the four water pumps, the LCD screen, and the microcontroller were already available to use when the design was being designed. There were additional materials that were required to be purchased in order to fully implement the Autonomous Plant Nursery, such as plywood for the chassis, two additional water pumps, 24V light bulbs with easily connectable terminals, low-gauge wire, and electric tape. The total cost of all components purchased for the construction of the Autonomous Plant Nursery was around four-hundred dollars, below the allocated budget of five-hundred dollars, therefore the cost of producing the device was not an immediate concern.

Tools Employed

Many tools were employed to design, construct, and test our project. For the hardware side of the project, tools from KiCad Services were utilized, specifically the PCB and schematic editors. These editors were used to plan out the placement of the components, first schematically with the schematic editor and then electrically using the PCB editor. In addition, the PCB editor was used to perform an electrical rules check (ERC) to ensure all connections designed in the schematic editor were made and would not cause failures in the PCB implementation. In addition, FreeDFM from Advanced Circuits was used to scan the completed PCB design for any potential errors before it was to be printed. Once the PCB was manufactured and shipped, 3W Electronics soldered the provided electrical components onto the PCB. For the software side of the project, Texas Instruments MSP430 was used as the microcontroller that operated the project. The code that ran the system entirely was written in the C language with Texas Instruments' own integrated development environment, Code Composer Studio (CCS).[1] In conjunction with the LCD, the code was tested to make sure the right output was being displayed. Autodesk Fusion 360 was used to design the overall shape of the chassis framework. The National Instruments VirtualBench was used to test the hardware and made sure it met the necessary benchmarks. For the documentation of the code, we employed Doxygen

Societal Impact Constraints

Environmental Impact

The project implements an LCD screen and a DC motor, these two components have big environmental impacts due to the methods of obtaining materials, their energy consumption, and their process of disposal. The LCD screen, also known as the Liquid Crystal Display, is made with glass, and indium, while the DC motor is made of copper and graphite. These materials require mining and thorough refinement, these processes are associated with negative impacts on the environment, which can lead to deforestation, contamination of local water sources, and increases in dust and noise pollution. Increases in habitat fragmentation have also been attributed to mining. As the system is in continuous use, there will be times when these parts of the system may need to be replaced, which can cause waste pollution. These components cannot be thrown into the trash alongside other regular waste as they may contaminate the environment with lead and other toxic chemicals. Their damage to the environment has made these parts to be classified as hazardous waste.

Sustainability

The system provides a sustainable design since it implements a power supply that connects to the wall. The lack of batteries prevents the need for constant replacement and the potential risk of decay. The design of the project also had a way to limit current, this was used to prevent too much power from being drawn.

The largest sustainability concern that comes from this device is the need to refill both water and nutrients. Although there is some small concern with the ability for our product to overwater if there is a malfunction or some user error causes the device to provide too much. There is the matter of the plastic containers that the nutrients and water are stored in, which is a much larger sustainability concern. In the initial prototype of our design that we created, we used changeable water and nutrient systems. Obviously, the water system could just be refilled with water which could be done without using plastic, but replacing the apple cider vinegar that our system uses to provide nutrients to the plant plots would require either exchanging the container or refilling the nutrients with more, which would likely require the purchase of the plastic containers that it usually is contained in.

Besides the plastic and water that our device will use, there is the power consumption that our device is going to use as well that must be considered. Our device uses two 12V, 500 mA grow lights that will be consuming power whenever they are on as well as four 12V, 300 mA peristaltic water pumps that will be used to deliver the water and nutrients to the device. The system also includes the powering of the MSP430FR2355 and the LCD screen. Although this system will likely never use all of the power that the system provides at one time, this power consumption is certainly a sustainability component that must be considered. [1]

Health and Safety

The system does deliver two different liquids which means that there is a potential danger for this liquid to interact with the electric system and cause a short that could be dangerous. Because of this, we designed an enclosed box to keep these systems separate to prevent objects or water from shorting the power.

There is also the risk of the electrical wiring to the light delivering system and the wires that are present at the front of our systems PC. To make sure that it is safe for customers to use, we covered the exposed areas of those wires and made sure as little as possible did not fit into the design of the device. There is also some user interaction that occurs while the user is interacting with the LCD screen so we made sure to make sure that these wires are nowhere near where these wires are being used. Because of this possibility, the device is completely constructed before purchase so that no mistakes will be made during construction and the safety of the user can be confirmed.

Ethical, Social, and Economic Concerns

Although this system is only useful during the early stages of plant growth, the ability of the plant nursery to monitor the plots and provide the appropriate amount of light, water, and nutrients do set a precedent. If this system was expanded it could be used to grow plants all the way through their entire life cycle. If this was expanded upon it could even be used in the agricultural industry to grow food for consumption. This could replace the need for some work in the agricultural department and take occupations away from those in the workforce. Of course, it could also result instead of reducing jobs to shifting them, because these machines would have to be checked on and monitored. Either way, these consequences need to be weighed.

External Considerations

External Standards

1. IPC PCB Design Standards - Standards maintained by the Institute for Printed Circuits (IPC) define general requirements for printed circuit board design. The standard known

as IPC-2221A provides standardization for spacing of parts and traces [4]. This was considered when determining how much space was needed for components and ultimately how large the PCB needed to be to fit all components. The standard known as IPC-A600J provides acceptance criteria standardization for printed circuit board holes, materials, plating, and other characteristics [5]. This was considered when determining which traces needed to cross copper layers and what material was needed for the board. The standard known as IPC-4101C provides information about materials used for multilayer board bases [6]. This was considered when the PCB of our project required a Copper Top and Copper Bottom layer in order for all traces to be completed on the board.

2. SMD Component Packages - Surface Mount Technology (SMT) standards provide information pertaining to Surface Mount Device (SMD) components. For the standardization of the size of SMD components, the trade organization JEDEC provides the most widely accepted specifications [7]. These standards pertaining to SMD were considered when designing our PCB board as most of the components soldered onto the board were SMD components.

3. Embedded C Coding Standard - The Embedded C Coding Standard Michael Barr is the author of The Embedded C Coding Standard, which is widely used to circumvent unnecessary errors and expedite the process of software development [8]. The standards authored by Barr, in particular rules pertaining to comments and white space, were relevant and followed throughout the coding of the project code.

4. I2C Serial Communication Protocol - The I2C protocol is a serial protocol for two-wire interface for communicating with peripheral devices in embedded systems, and is based on the master-slave model [9]. The protocol was used to communicate with the AD5933 chip to retrieve water and nutrient content from the plots.

5. Compliance with RoHS (Restriction of Hazardous Substances) Standards – RoHS is a set of standards maintained by the European Union that restricts the use of materials that are hazardous in nature found in electrical and electronic products [10]. In our project, most of the components used were compliant with the RoHS standards, which minimizes the risk of harmful environmental impact of the device and endangerment of the user's safety.

Intellectual Property Issues

The hydroponic smart gardening device described in [11] consists of three sections of interest: a "means for delivering electricity to said smart garden device", "at least one timer", and a "means for determining, receiving, sending, or processing data regarding the status of said component or characteristic of said gardening or hydroponics device". The patent outlines the construction and operation of an autonomous plant-growing device that utilizes hydroponics, a method of growing plants without the use of soil, and monitors the water level, nutrient level, and lighting times of plants grown with the device. The monitoring functionality is "comprises a preprogrammed or programmable storage device, [either] a circuit board [or] a computer chip" capable of “determining, receiving, sending, or processing data regarding the status of two or more components or characteristics…and a means for displaying the status of two or components or characteristics,” [11]. The lamp used for lighting of plants, the LCD used to display critical growing information, and the liquid level run on a “two-week cycle” timing system in which that status of these conditions are updated at the end of the cycle [11]. This system of monitoring the plants is similar to the methods used for the Autonomous Plant Nursery in that the same characteristics are monitored over a certain period of time. An intellectual property conflict is not likely to result, as the Autonomous Plant Nursery does not make use of hydroponic technology, its timing system is based on one day and not two weeks, and the parameters for growing can be changed by the user. The Autonomous Plant Nursery allows the user to change growth parameters without needing direct intervention, instead of the device simply choosing and using predefined growth parameters.

The garden monitoring and management system described in [12] consists of a “sensor placed in soil in a garden [to measure] a plurality of measurement of at least one soil parameter during a first predetermined time period”, a “[method of] operating at one treatment system to control the growth of a plant in the garden”, a “[method of] storing a history of the operational parameters of the at least one operational system”, and a“[method of] operating the at least one treatment system…to the stored history of the operational parameters during a second predetermined time period”. The overall claim made by this system is that “the plurality of measurements of the at least one soil parameter…with at least one sensor…[including] moisture content level, …potential hydrogen (pH)…, temperature level…, and an intensity level of light,” [12]. These measurements are used, in two other claims, to create “a method of identification of plants for planting in a garden,” and a management system that consists of “a treatment system for the garden…to treat a plant,” and “a server to receive…[and] store a history of operational parameters…during the first predetermined time period in a memory,” [12]. The system described in this patent is similar in nature to the Autonomous Plant Nursery in that sensors are placed into the soil to record various measurements of the plants, and a management system is used to reference the recorded values with preset values to make corrective plant management actions. The two devices differ in that the Autonomous Plant Nursery measures water and nutrient content using an impedance converter placed into the soil of the plant plots, rather than using dedicated sensors for each parameter like the device in the patent. In addition, the Autonomous Plant Nursery gives a user the option to set the growing parameters of the plots (water/nutrients and lighting times) by themselves or use preselected settings and display selected settings on a LCD screen, whereas the device in the patent uses a large catalog to identify and care for plants and sends the data over a server.

The intelligent gardening system and methodology described in [13] consists of a “moisture level sensor with a display into a pot”, as well multiple abilities, such as “launching a gardening application on a mobile device”, “acquiring video data from the display of the moisture level sensor”, “storing the video data into a cache memory”, “narrowing the video data to the pot ID tag and…current moisture level data", "deriving the current moisture level data", "storing the current moisture level", "analyzing to current moisture level…[to generate] gardening recommendations", and "rendering…gardening recommendations on…[a] mobile device". An independent claim the patent makes is that the intelligent gardening system implemented by the described device consists of a mobile device "configured to acquire video data from the moisture level sensor display…to derive a current moisture level data" and "render gardening recommendations to a user," [13]. The overall methodology between the patent and the Autonomous Plant Nursery are the same, in that the moisture content of the soil is recorded and used to change the system. In addition, both devices actively convey measured parameters to the user in order for the user to make desired changes to such parameters. The Autonomous Plant Nursery uses the moisture level data in order to determine if water is needed in the plot, and provide it if necessary, whereas the device in the patent uses the moisture data in order to recommend the user to add water manually. Furthermore, the Autonomous Plant Nursery uses the numerical reading on the data sensor to make a moisture threshold determination, whereas the patent uses a visual reading to make a moisture threshold determination

Based on the survey conducted on these three patents, the Autonomous Plant Nursery would not receive resistance towards patentability as a novel product.

Detailed Technical Description of Project

Hardware:

This section of the Detailed technical report is dedicated to the hardware section of the design process. A high-level diagram is shown in Figure 1, and from it, the subsections of Power Management, Water Interface, Lighting Interface, Moisture Interface, and Client Interface can be seen. At this level, it can be seen that power will enter the board through a barrel jack connector that will receive and deliver 24V 5A, 120W of power from an AC wall outlet then through an AC/DC transformer. These exact values were selected due to the total current draw of the entire board sitting right around 4.2A, and the highest voltage device requiring 24V exactly. A few of the subsystems also output to combicon connectors, as that is how we will be connecting the positive and negative leads of our external devices to the board. From this top level, it can also be seen that there are 2 20 pin connectors located on the bottom of the board, which correspond with the 2 2x10 header pins of the MSP430FR2355 which is the candidate microprocessor for our design [1]. The MSP was chosen because of our familiarity with the MSP430 design and usage, and the FR2355 specification was selected due to the availability of pins to control our various subsystems. The left most 20 pin connector represents the leftmost 2x10 connector of the MSP430FR2355 [1]. Pins 1-10 represent the rightmost 10 pins starting at the top of either connector, and pins 11-20 represent the leftmost 10 pins of a given connector.

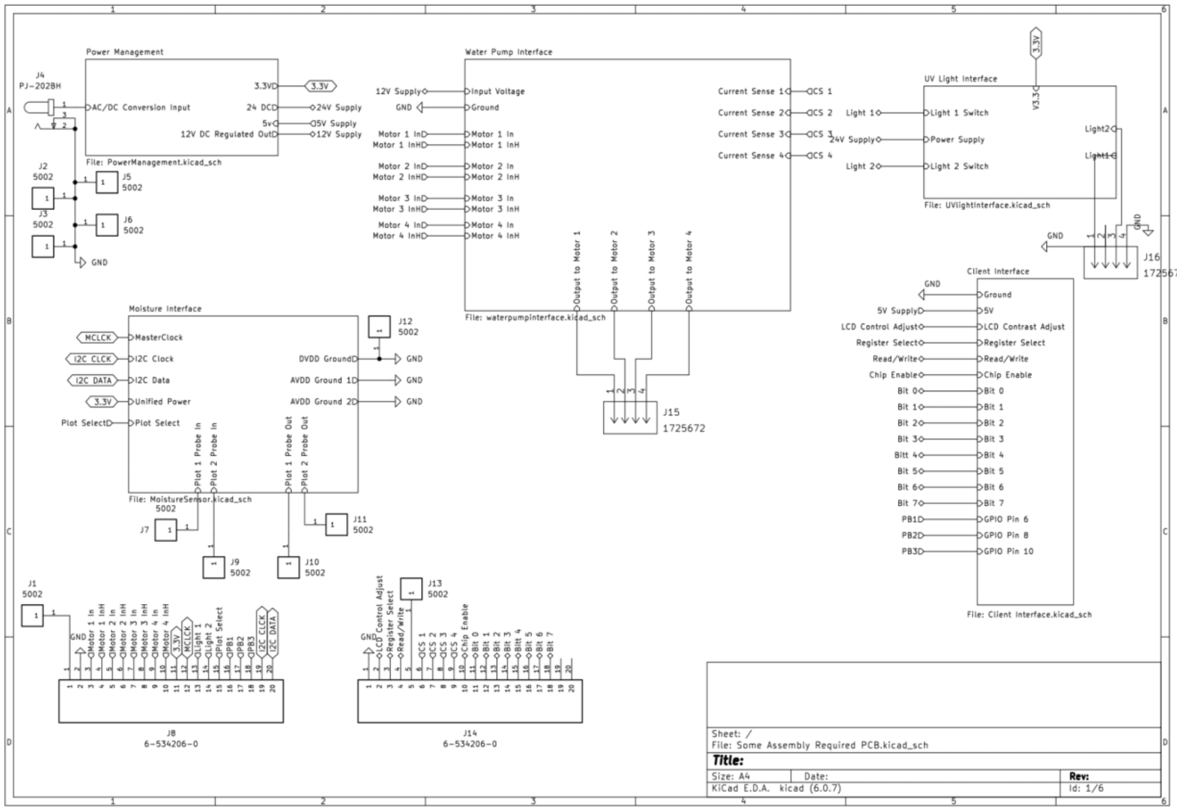

Power Management
Water Pump Interface
UV Light Interface
Moisture Interface
Client Interface
File: PowerManagement.kicad_sch
File: waterpumpinterface.kicad_sch
File: UVlightinterface.kicad_sch
File: MoistureSensor.kicad_sch
File: Client Interface.kicad_sch
Sheet: /
File: Some Assembly Required PCB.kicad_sch
Title:
Size: A4
KiCad E.D.A. kicad (6.0.7)
Id: 1/6

Figure 1: Kicad Schematic of Autonomous Plant Nursery Subsystems

Navigating through the subsystems left to right, top-down, the first subsystem is going to be the power management system. As shown in Figure 2, a line of 24V 5A will enter the circuit and connect to Jumper J36. This will allow us to disconnect the 24V powerline in the event of failure and use an alternative source to power our system. Once shunted, this jumper will allow the flow of power to 3 separate voltage regulators, the topmost outputting 3.3V 1.5A which will be used to power all ICs, the center regulator outputting 12V 1.2A which will be used to operate the 12V water pumps, and the final one outputting 5V to provide a 5V source for our LCD backlighting. The 24V line also connects to our lighting fixture to provide the necessary voltage for the lighting array. Each regulator is paired with bypass capacitors sized according to the specification outlined in the datasheet of each respective regulator.

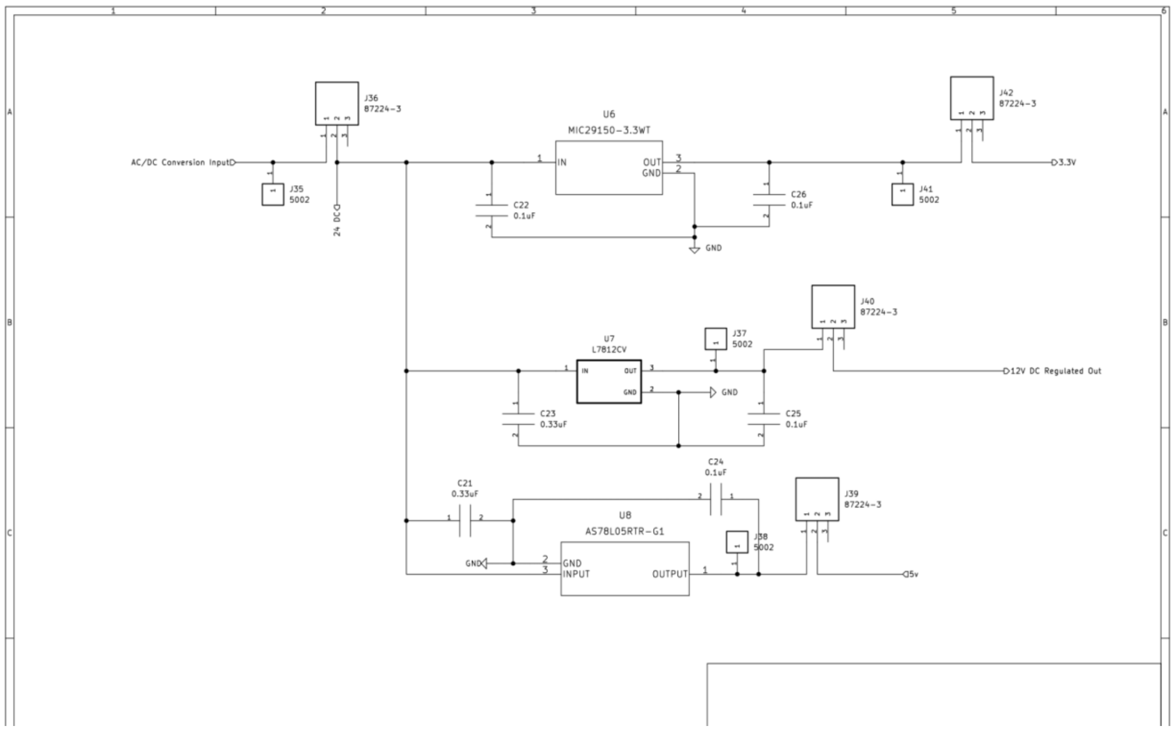


Figure 2: Kicad Schematic of Autonomous Plant Nursery Power Management System

The next subsystem of our design is the Waterpump Interface showcased in Figure 3. The water and nutrients are designed to be delivered with 12V DC Peristaltic pumps [14]. The motors are controlled with an integrated high current half bridge motor driver (BTN8962TA) [15]. These motor controllers are designed to be able to handle voltages up to 40 Volts and 30 Amp currents so this chip is beyond a conservative estimate for our purposes. The chip is powered by the 12V 1.5A sub system. This produces an output of 12V and 1.5A to the motors when the chip's IN pin is set to high. The INH pin is set to receive a 3.3V input from the microcontroller at all times, and the IN pin is set to only receive a 3.3V input when the system is ready to turn on the motors. The SR pin sends data back to the MSP430 which will be used to shut down the entire circuit if an over current issue appears [1]. The output pin is only activated when IN and INH pins are both high and the voltage received from the input goes across the output of the motor which delivers the water and nutrients to the plant.

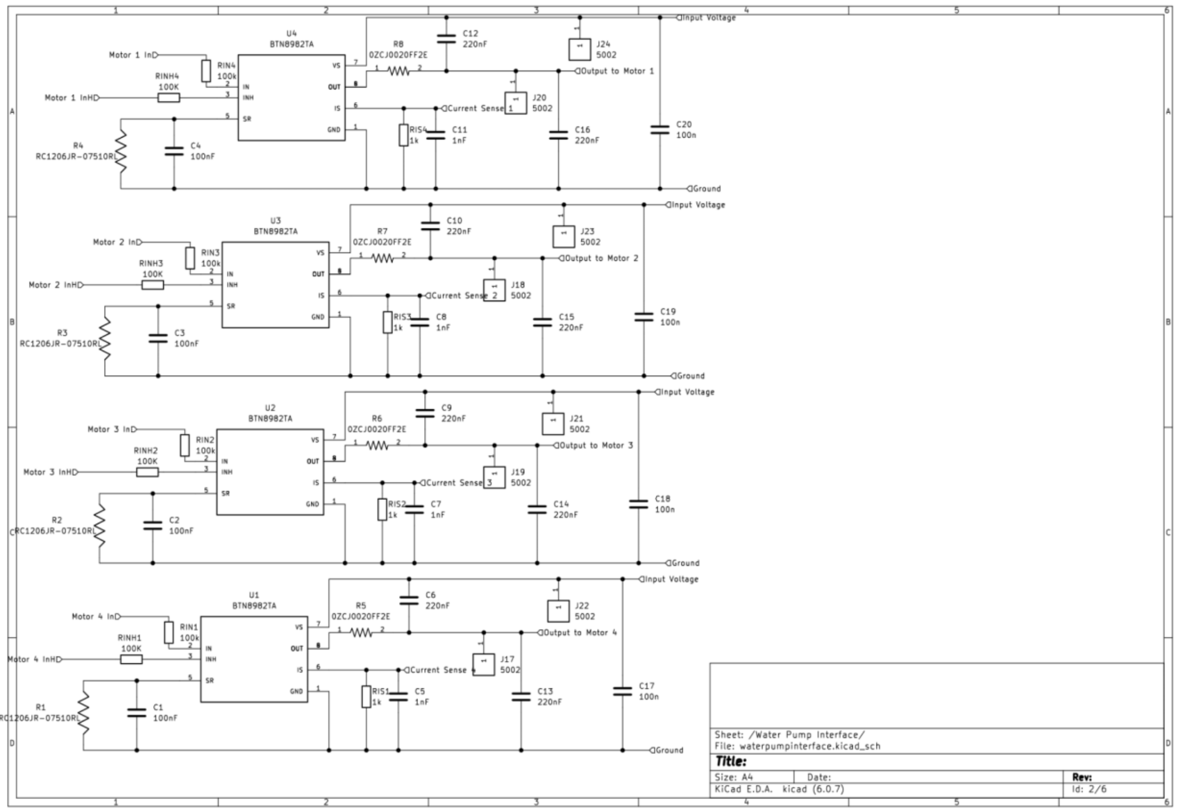


Figure 3: Kicad Schematic of Autonomous Plant Nursery Waterpump Interface

The next subsystem of our design is the Lighting Interface showcased in Figure 4. The provision of light is accomplished by using a relay rated for a switching voltage of 24V and a switching current of 500mA for each bulb. The current needed to power operate the relay was 33 mA, which meant that the MSP430 alone was incapable of sending a logic high capable of driving the relay coil that would close the circuit and deliver the full 24V hanging on the Normal Open terminal of the relay [1]. So additional circuitry was created featuring an NPN transistor in between the logic line of the MSP and the input to the relay coil. The NPN transistor chosen was a 2n3904, and this would be used to act as a switch for the MSP430, so when a logic high was detected, and the 2mA of current that the MSP430 could output was sensed at the base of the transistor, it would create a channel from the 3.3V power supply line to ground, and allow the coil to draw current from the power line so long as the MSP was delivering a logic high. When low, the connection across the transistor was closed, and the coil would be seen as a dangling load, therefore no current can flow through. Between the MSP430 and the 2n3904 is also a 1300 ohm resistor. When the logic signal is high from the MSP, this resistor will draw 2.5mA of current which will be applied to the base of the NPN transistor to fully saturate it. 2 more components were added to the design, the first being a 1n4001 diode which acts as a flyback diode. Because the relay operates off of a coil that generates a magnetic field to facilitate the action of opening and closing the relay, once the NPN transistor is no longer being saturated, the coil will still have an inductive current that must flow somewhere. If this current were to flow into the transistor without it in the saturation region, it would likely permanently damage the transistor. So the flyback diode is in place to allow the excess current to continuously flow through the diode and ultimately be converted into heat energy. The second added component was a 600mA polyfuse. This was added as a protection measure in case the light bulbs would attempt to draw too much current, as they would be expected to be left on for extended periods of time, if they were to build up heat, and draw more current as a result, the fuse would blow, forcing them to turn off, and naturally reset itself.

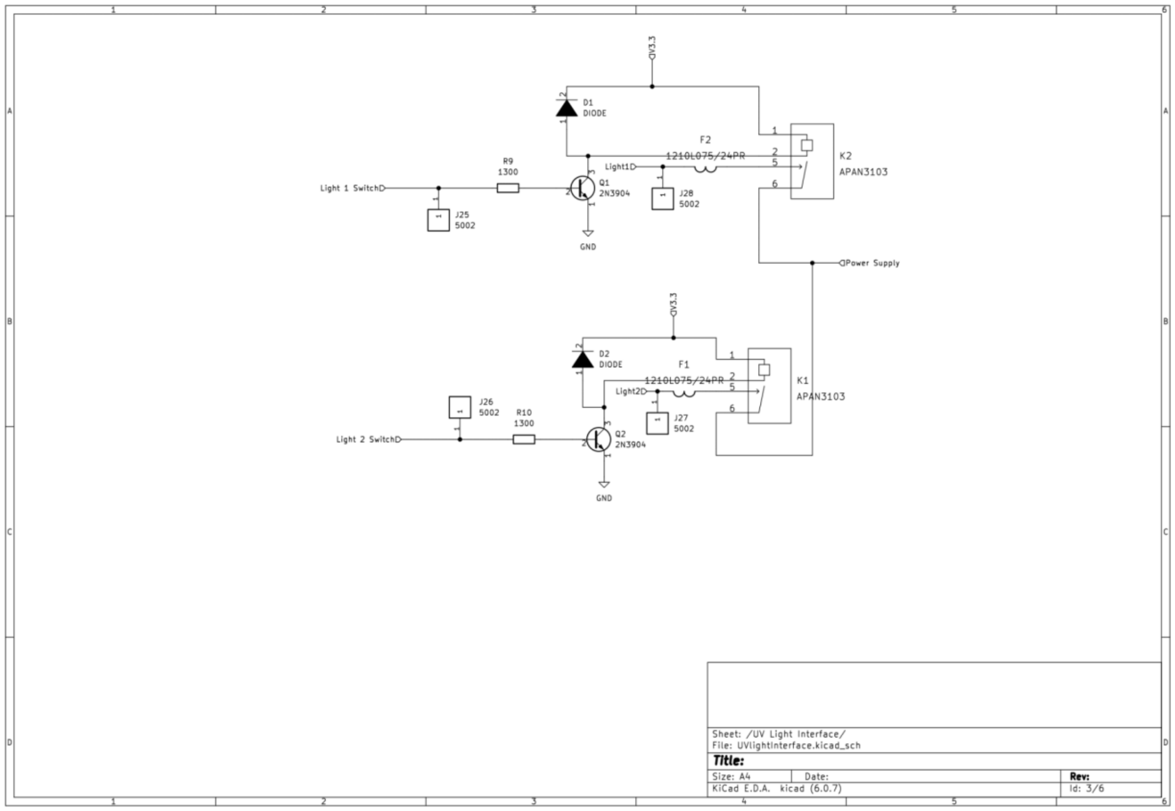


Figure 4: Kicad Schematic of Autonomous Plant Nursery Lighting Interface

The next subsystem is the moisture and nutrient sensor shown in Figure 5. This subsystem, similar to the last, features a relay that instead of toggling the distribution of power on and off, allows the selection of an input between 2 sets of probes that would be used to monitor the impedance of the soil within the 2 plants the device can support. The monitoring of the soil impedance is facilitated via an IC made by Analog Digital called the AD5933. It is an impedance converter and network analyzer and functions by injecting a test signal from the Vout Pin and measuring its strength at the Vin pin. This pin feeds into an internal transimpedance converter, and requires that the user specifies the feedback resistor to ensure that the current conversion to a voltage signal maps appropriately within the variable range of the VDD supply.

Given that we are using a 2V p-p signal from Vout, and supplying a VDD of 3.3 volts. According to the datasheet, the maximum current output by the Vout pin is 5.8 mA, so in calculating RFB, the following equation can be used.

RFB = (Voutmax - Voutmin) / Imax = 4V / 5.8mA = 689, where Voutmax is positive 2V, Voutmin = -2V, and the maximum output current is 5.8mA. However, the closest value to 689 we were able to order was 660, so our circuit uses a 660-ohm resistor as the feedback resistor for the transimpedance converter. The final piece of circuitry involves a 200k Ohm resistor which is selected manually by connecting jumpers J33 and J34 to read from the calibration resistance as opposed to the unknown impedance tested via the probes. This was chosen due to the requirement of the AD5933, as part of the conversion of impedance, a calibration component was necessary.

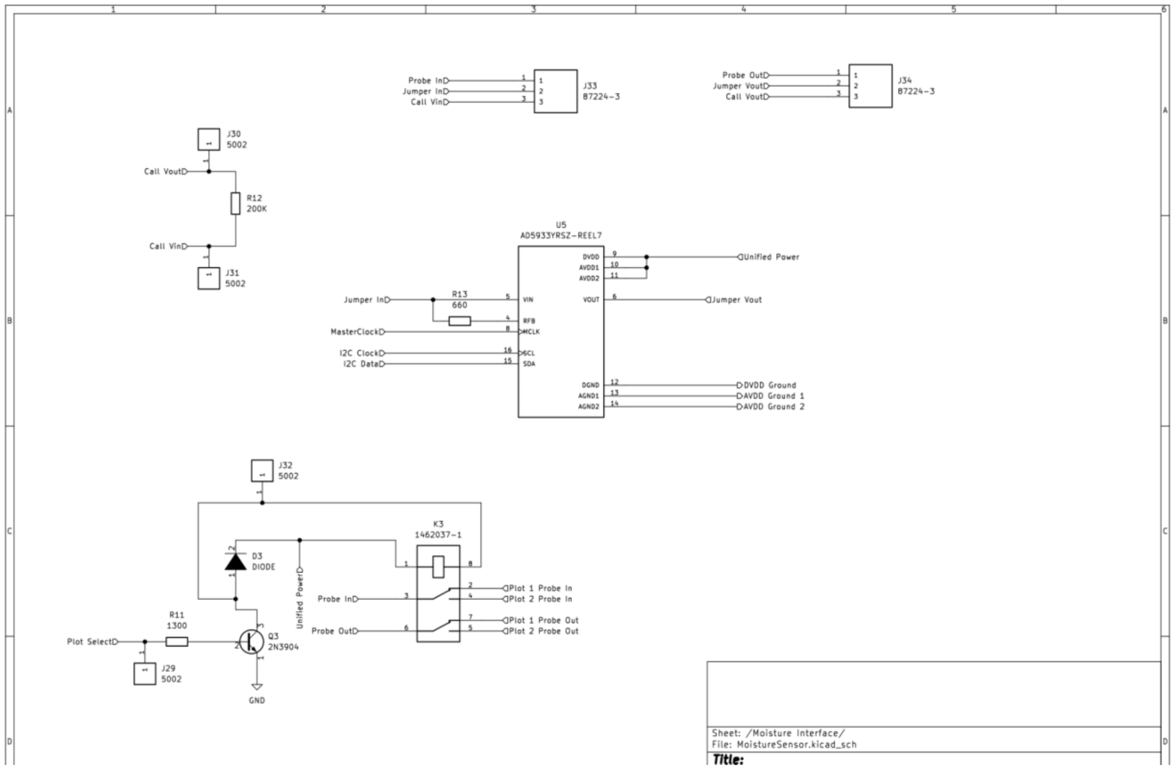


Figure 5: Kicad Schematic of Autonomous Plant Nursery Water/Nutrient Interface

The final subsystem of the device is the client interface subsystem showcased in Figure 6. This subsystem features 3 pushbuttons configured with a pull-up resistor of 3.3k Ohms attached to the 3.3 supply line, allowing for 1mA of current to sink into the MSP430 while the pushbutton is open, which is within the safety specification of sink current for any given pin. When closed, the 1mA of current is sufficient to keep the contacts of the pushbutton free of corrosion, and ensure the lifespan of presses is capable of being satisfied. The next resistor works with the 3.3k Ohm resistor to create a voltage divider, when current is able to flow through the closed connection of the pushbutton, it will pull the voltage line down 0.065V, significantly low enough to be detected as a logic low. The value of this resistor is 66 ohms. There is also a TVS diode which will prevent any electromagnetic signals from being picked up and sent as unintentional pushbutton inputs or preventing intentional pushbutton inputs. The final component of this interface is a 2x8 connector for a ribbon cable that will connect our MSP pins directly to the pins of the LCD.

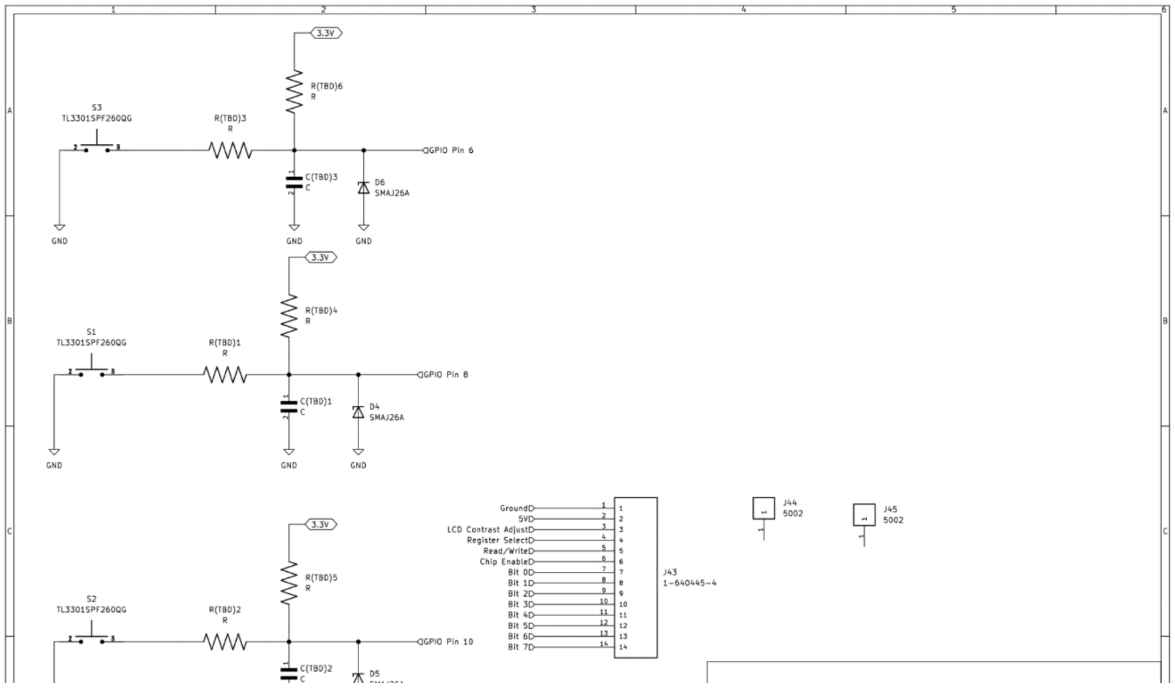


Figure 6: Kicad Schematic of Autonomous Plant Nursery Client Interface

The software technical description is split into both the Initial Software Process that was originally implemented and the Final Software Process that the project used because of certain hardware or software issues that appeared. These two processes behaved very similarly with some small exceptions which will be discussed in that section.

Software:

Initial Software Process

The initial software process involved general purpose input and out (GPIO) processing to control the motors, lights, and push-buttons that controlled the Liquid Crystal Display (LCD) display [16] and allowed the user to set up the system. This system calls for multiple Finite State

Machines that correctly set up the motor output components depending on what input comes in, and correctly communicate this information with the user through the LCD screen. The system was also designed with several interrupts that would operate the lights based off of a time that the user set up and also registered when the push buttons were being used. The final software structure of the original design used an Inter-Integrated Circuit (I2C) system to allow communication between the MSP430FR2355 (MSP430) [1] and the high precision impedance converter (AD5933) [17] to measure impedance in the soil that would control the rest of the system.

The Push Button and LCD Network

The pushbutton network is set up using different port interrupts that were built into the MSP430[1] in combination with a complicated FSM network that adjusted the LCD screen and would set up the desired values for the light, water, and nutrients. The system first works by throwing a flag on the port of whichever push-button was pressed which would tell the system to halt. Depending on which button was pressed the LCDs FSM will change its current state to either move the cursors up, down, or go to the next stage of the screen. Once that has been done the interrupt flag is cleared and the system continues to update and run the motor finite state machines until another flag is hit.

The LCD is set up with two ports that are used to control the LCD, then is used to turn on the LCD screen, clear the screen, set the cursor to home, and turn off the cursor. The LCD FSM is then used to display messages to the user, and its output will be used to control the rest of the system.

The initial powering of the LCD requires setting up four control pins that correlate to the Register select pin, read or write pin, enable pin, and LCD contrast pin. The register select pin allows the system to switch between sending system controls such as turning the display on and off, or setting the cursor position when set to low and sending the read and write controls to the board when set high. The read/write pins allow the user to either read or write the data. The enable bin is used to enable the registers and let the LCD know that there is about to be a data or command transfer, and the LCD contrast pin sets the light level of the system. The system then sets up 8 pins used to send characters of data to the LCD that correlate to ASCI variables that will be written on the screen. The FSM state will decide what is written on the screen using the screen write command and then sending the character bits from the MSP430[1] to the LCD screen. This was tested on a breadboard before being attached to the screen as shown in Figure 7.

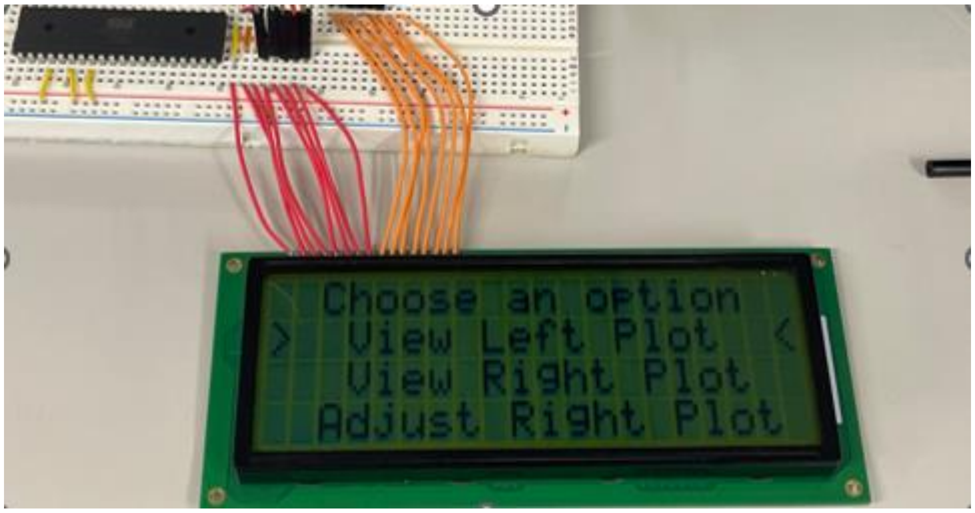


Figure 7: Initial options of the Liquid Crystal Display Screen

A quick walkthrough for the function and states of the LCD FSM makes clear how this system operates before going into detail about how it was controlled with the pushbuttons. The LCD FSM is built with only three possible inputs that control which state the machine goes into but has over 50 actual possible states at any time. The inputs that control this FSM are the three pushbuttons and correspond to ENTER, UP, or DOWN. Where up and down are designed to just scroll the screen and enter allows the user to actually select an option such as viewing a plot's moisture level. Because we used a finite state machine for this, every single up and down is considered a new state with the cursor moved up or down by one depending on the button pressed. The system is also designed in such a way that if the user is on the final option and presses the down button the cursor will be set to the top option of that screen. The output is only the changes of the screen state unless an amount of water or light are given. There is a short example FSM of the initial idea shown in Figure 8.

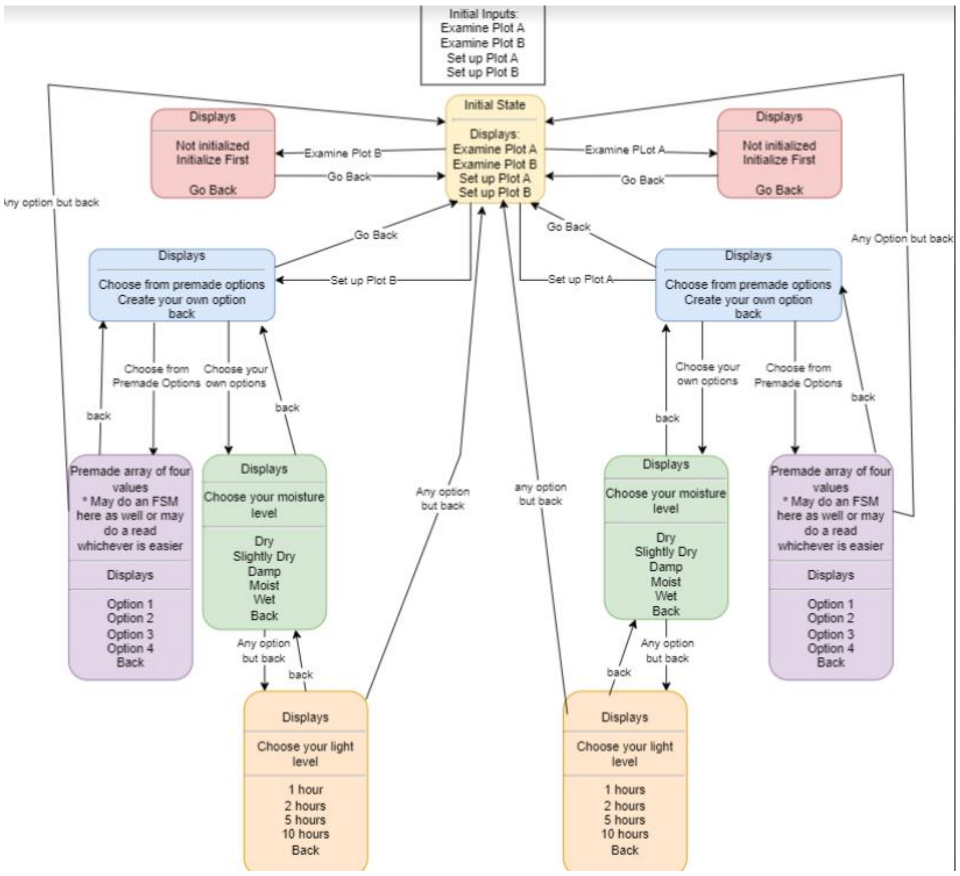

Initial Inputs: Examine Plot A Examine Plot B Set up Plot A Set up Plot B
Initial State
Displays: Examine Plot A Examine Plot B Set up Plot A Set up Plot B
Displays
Not initialized Initialize First
Go Back
Examine Plot B
Examine PLot A
Any option but back
Any Option but back
Displays
Choose from premade options Create your own option back
Set up Plot B
Set up Plot A
Choose from Premade Options
Choose your own options
back
Premade array of four values * May do an FSM here as well or may do a read whichever is easier
Option 1 Option 2 Option 3 Option 4 Back
Choose your moisture level
Dry Slightly Dry Damp Moist Wet Back
Any option but back
Choose your light level
1 hour 2 hours 5 hours 10 hours Back

Figure 8: Original FSM for the LCD Controls, States, and Outputs

When the user chooses to set up a plot, they must first decide which plot that they want to choose. Because there are two plots that each correlate to their own specific water pumps and light bulb, the FSM is designed with states for the first plot selected and the second plot selected effectively doubling the number of states that are needed. Once the user decides on a plot to adjust the FSM directs them to an option on the screen that first allows them to select the light level for their plant. Because of the constraint of reading data from the LCD the board is designed in a way to give the user several light options that they can choose instead of letting them choose a custom option. The user is able to set the system either one hour, three hours, six hours, twelve hours, a full day, or the test time of one minute for demonstration purposes. The next state function then sets the light threshold as an integer in the form of whatever time that the user selected, for example: one minute would be 60. After an amount of light has been chosen, the FSM immediately changes the state so that the moisture options are displayed for the user to choose. Once the user chooses a moisture level of the soil the FSM will assign the specific moisture level chosen to correlate to specific upper and lower thresholds for the plot that the user initially chose that the moisture level will be contained to, as well as upper and lower nutrient levels. The system then updates the global variable that indicates if a plot is set or not to let the device know to start monitoring this system and give light and water as needed. After this is finished the FSM transitions to its original state where the user can adjust the plot they just created, adjust a different plot, or view an already working plot.

If the user decides to view a plot the FSM acts in very much the same way as before where it shifts to different states updating the output that is displayed on the screen as its going. If the chosen plot to view has been set the FSM will display four pieces of information to the user: the desired moisture level, the actual moisture level that correlates to the level received from the AD5933[17], the amount of time in minutes that the light has been on that day, and the amount of time that the light has left to be on before it turns off.

Motor Control and Finite State Machine

The motors are controlled by the MSP430[1] to either turn off or on depending on the information stored in the Pump control FSM. Each of the four motors has three pins that are connected to the MSP430[1]. The first pin controls whether or not the motor will be on or off. These pins are immediately turned on when the system starts, so that each motor is ready to go as soon as the system is operating. The second pin on each motor connected to the MSP430 is put in for the system's protection. This pin detects over current that would shut down the system if there happens to be a current problem that could damage the system. The final pin is set to high or low depending on the information stored on the FSM.

The motor FSM is constructed to initialize the GPIO pins used to turn the motors on and off and then set them to off until updated to do otherwise. The design of the FSM for a single plot which delivers water and nutrients can be seen in Figure 9. These FSMs are set up with the current state of the motor either being on or off, the high threshold that is set with the LCDFSM as seen in the Push Button and LCD Network section, the low threshold that is set with the LCDFSM, and the correlating pump number. The system operates in a simple way where if the moisture level received from the AD5933[17] falls below the desired level it will turn the motors on that will deliver moisture and nutrients until the upper threshold is reached and then those motors will turn off. There is a timer set up that will turn the system off after 15 minutes if it has been on the entire time, and the system is set up in a way that it only checks the moisture and nutrient level once a day so that over watering will not occur.

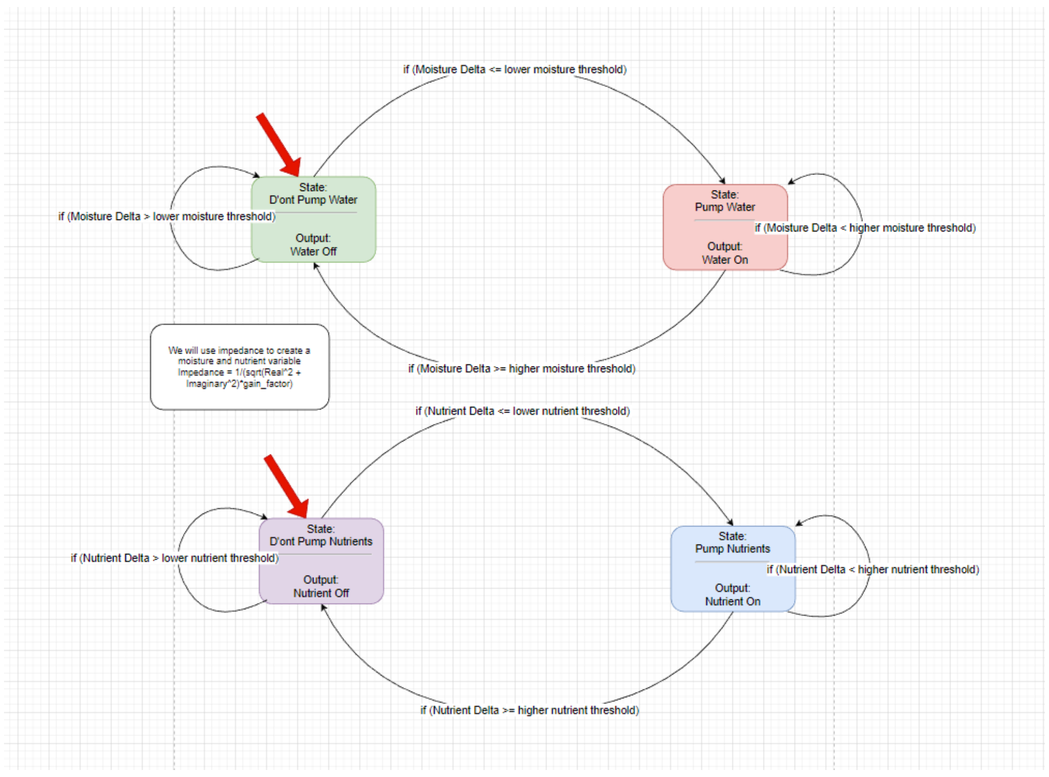

if (Moisture Delta <= lower moisture threshold)
State: D'ont Pump Water
Output: Water Off
if (Moisture Delta > lower moisture threshold)
State: Pump Water
Output: Water On
if (Moisture Delta < higher moisture threshold)
if (Moisture Delta >= higher moisture threshold)
We will use impedance to create a moisture and nutrient variable Impedance = 1/(sqrt(Real^2 + Imaginary^2)*gain_factor)
if (Nutrient Delta <= lower nutrient threshold)
State: D'ont Pump Nutrients
Output: Nutrient Off
if (Nutrient Delta > lower nutrient threshold)
State: Pump Nutrients
Output: Nutrient On
if (Nutrient Delta < higher nutrient threshold)
if (Nutrient Delta >= higher nutrient threshold)

Figure 9: Finite State Machine Showing How Water and Nutrients are Delivered

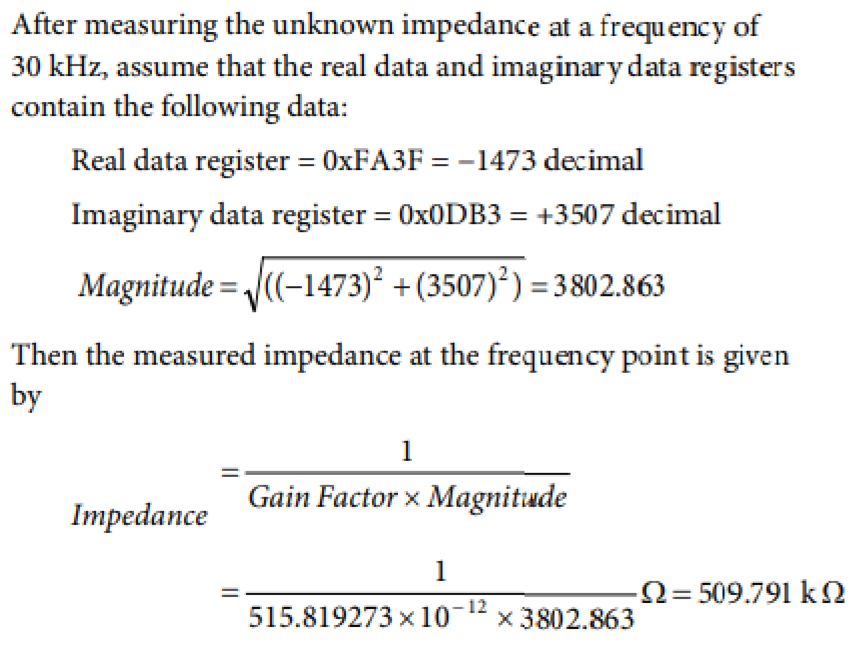
Light System

The light system works with GPIO pins that are initialized exactly like the motor pins. This system is then set up with a global timer that runs for an entire day and then resets. When the amount of time the light is needed is set, the device turns the light on, sets that timer, and using a clock interrupt set at one second continues to decrement that amount until the light time equals zero. Once the lights timer equals zero the MSP430[1] sends a logical low to the system turning off the light. The light will remain off until the global daily timer is reset which will reset the lights value to its value that is chosen by the user, turning the light back on and starting the whole process again.

Impedance Measuring System

The entire pump system is set to operate off of the moisture and nutrient values that the plant currently has and produce an output depending on the readings of that system. This works by switching back and forth between plots and reading the impedance of the soil with the AD5933[17]. An I2C implementation was constructed to communicate between the MSP430 and the AD5933[17]. Then the values that AD5933[17] gave to the system had to be adjusted since they were given in complex and real components.

In order to get useful components from the AD5933[17] we had to turn the values that it gave us into impedance of the soil. This was done by taking the values that the AD5933[17] gave us and splitting them into two numbers one for real and one for imaginary. We would then find the Magnitude of these numbers by taking the sum of each number squared. Then we would take that number and multiply it by the gain factor that is decided given the frequency of the sweep which we decided to do at 30kHz. An example of this equation can be seen in Figure 10.


After measuring the unknown impedance at a frequency of 30 kHz, assume that the real data and imaginary data registers contain the following data:

Real data register = 0xFA3F = −1473 decimal

Imaginary data register = 0x0DB3 = +3507 decimal

$$Magnitude = \sqrt{((-1473)^2 + (3507)^2)} = 3802.863$$

Then the measured impedance at the frequency point is given by

$$Impedance = \frac{1}{Gain\ Factor \times Magnitude}$$

$$= \frac{1}{515.819273 \times 10^{-12} \times 3802.863}\Omega = 509.791\ \mathrm{k}\Omega$$

Figure 10: Example Calculation of Impedance from AD5933

This system uses an I2C configuration to communicate between the AD5933 and the MSP430. [1] This is done by following a strict I2C setting where the MSP430 sets up a clock for the I2C network and then sends a start bit across that clock so that the AD5933 knows that the MSP430 wants us to send data to the AD5933. Once the start bit has been received the MSP430 sends the register value of the AD5933 that contains the information of either the real or imaginary component and then sends a high bit informing the AD5933[17] that the MSP430 wants the information in this register to be sent back. The AD5933[17] then sends an acknowledge bit to let the MSP430 know that it is sending the data and then sends the data one bit at a time over the constructed clock cycle back to the MSP430. Once that is done a non-acknowledge bit is sent and then a stop bit is sent informing the AD5933 that the MSP430[1] is done requesting information. The MSP430 does this for each register and then after using the equation in Figure 11 on the values stores the impedance and then the main method runs the motor FSMs and compares the FSM thresholds with those numbers to see if the system needs to deliver more nutrients or moisture.

Final Software Process

The final construction of the design was not able to use the AD5933[17] impedance chip because of either an issue with the I2C of the chip not being generic or an issue with the chip itself not working correctly. To compensate for these problems, we adjusted the moisture and nutrients to be delivered using clock interrupts much like the water, where the system would deliver 30 seconds of water and nutrients periodically throughout the day. Depending on the moisture level chosen the period between delivery times is either shorter or higher with heavy moisture requiring plants to receive water and nutrients every hour whereas plants that require very little moisture only receive water and nutrients once or twice a day.

The final construction also had to lead away from the motor drivers that the system was initially designed to use and went with a switch and diode setup that only needed one input or output from the MSP430 setting the motor to either high or low [1]. This was an easy change that was achieved by deactivating the pins for current reading and setting the motors to always be on and then just using the motor GPIO pins that were explained in the Motor Control and Finite State Machine subsection to either turn the motors off and on.

Project Timeline

The proposed timeline below shows the timeline the group proposed during the start of the project. The Gantt chart lacked specific dates, and holidays and did not allow extra time for unforeseen changes. Our chart has three main sections: PCB design, Embedded programming, and chassis design. These sections consisted of many individual tasks that were meant to be done in parallel. The parallel tasks were PCB design, Chassis design, and writing embedded programs for water pumps and LCD screens. These tasks did not depend on each other completely. But there were tasks that depended on the completion of other tasks. These were done in series. For instance, the AD5933 chip required board completion before I2C code could be tested due to the chip's inability to be tested on a breadboard.

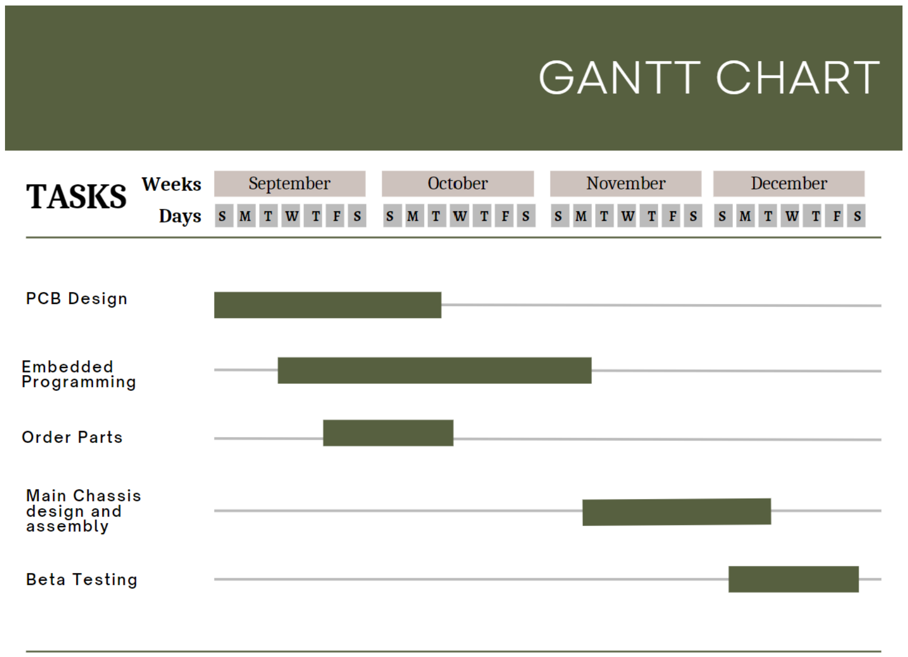


Figure 11: Original Gantt Chart Timeline of Autonomous Plant Nursery

Ryland's primary tasks were PCB design with extra focus on individual components and chassis design. His secondary responsibility was assisting with debugging the embedded code and testing the logic following the connection to another subsystem.

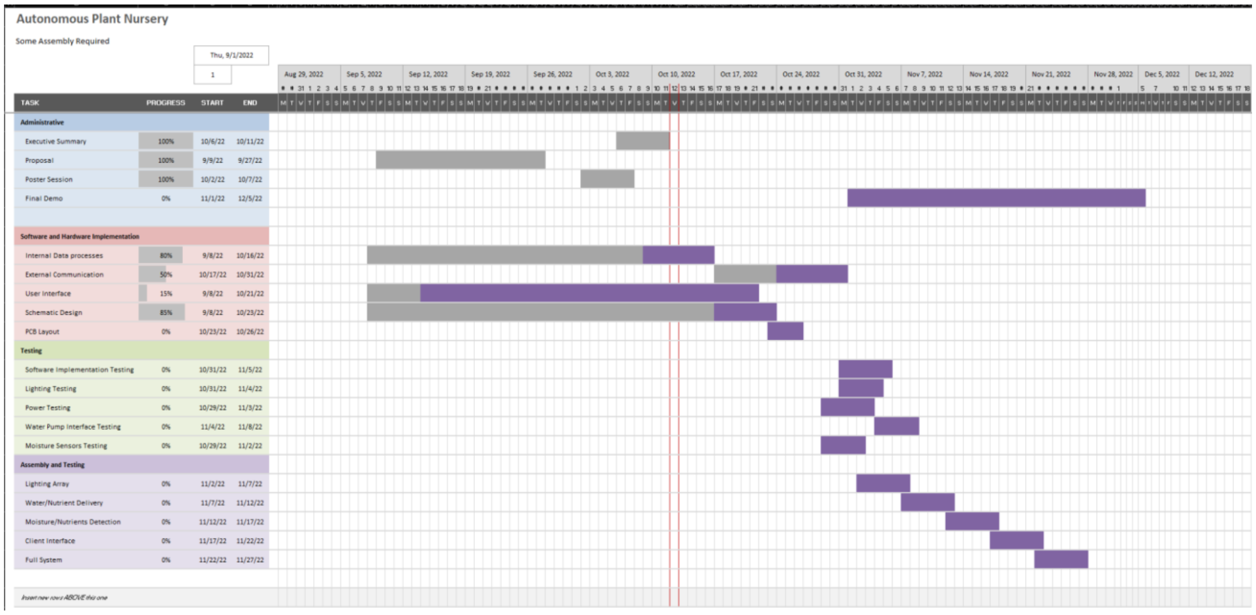
Joseph's primary responsibility was PCB design with an additional focus on the design of the power supply and input protection. His Secondary responsibilities were assisting with the UV lighting logic and sensor functionality.

Joshua's primary responsibility was embedded design with an additional focus on the FSM's that controlled the watering and lighting systems. His secondary responsibilities were the LCD functionality and button control.

Mckayla's primary responsibilities were embedded design with an emphasis on I2C communication with the AD5933 chip and embedded code for the LCD screen functionality. Her secondary responsibilities were assisting the design of the Chassis design.

Boluwatife's primary responsibilities were wiring of the lights and water pumps, Electrical Standard Compliance, and Sensor functionality. His secondary responsibilities were LCD functionality.

Figure 12 showcased the final timeline which details the schedule towards the end of the project. This Gantt Chart included a hard deadline for more specific tasks. There were many issues that took more time than we anticipated. There were difficulties with the I2C communication and PCB supplying expected voltages once a load was added. These delays cost us time that was designated for testing.


Autonomous Plant Nursery

Some Assembly Required

Thu, 9/1/2022

1

TASK	PROGRESS	START	END
Administrative			
Executive Summary	100%	10/6/22	10/11/22
Proposal	100%	9/9/22	9/27/22
Poster Session	100%	10/2/22	10/7/22
Final Demo	0%	11/1/22	12/5/22
Software and Hardware Implementation			
Internal Data processes	80%	9/8/22	10/16/22
External Communication	50%	10/17/22	10/31/22
User Interface	15%	9/8/22	10/21/22
Schematic Design	85%	9/8/22	10/23/22
PCB Layout	0%	10/23/22	10/26/22
Testing			
Software Implementation Testing	0%	10/31/22	11/5/22
Lighting Testing	0%	10/31/22	11/4/22
Power Testing	0%	10/29/22	11/3/22
Water Pump Interface Testing	0%	11/4/22	11/8/22
Moisture Sensors Testing	0%	10/29/22	11/2/22
Assembly and Testing			
Lighting Array	0%	11/2/22	11/7/22
Water/Nutrient Delivery	0%	11/7/22	11/12/22
Moisture/Nutrients Detection	0%	11/12/22	11/17/22
Client Interface	0%	11/17/22	11/22/22
Full System	0%	11/22/22	11/27/22

Insert new rows ABOVE this one

Figure 12: Updated Gantt Chart Timeline of Autonomous Plant Nursery

Test Plan

Hardware Test Plan

The first section of the test plan involved testing the 24V, 5A input supply inputted into the PCB board via the barrel connector J4. The barrel jack connector was connected to the 24V 5A AC/DC transformer, and the jumper J35 was used to determine if the voltage and current parameters of the input supply were met. In addition to this supply voltage, J41 was used and tested to determine if the 3.3V supply voltage was created by the 3.3V voltage regulator, J37 was used and tested to determine if the 12V supply voltage was created by the 12V voltage regulator, and J38 was used and tested to determine if the 5V supply voltage was created by the 5V voltage regulator. After the verification of the power supplies, the jumpers of J2, J3, J5, J6, and J12 were tested to determine if these pins were ground (GND) pins.

After the first stage of tests were completed, the second stage of steps were completed utilizing a Virtual Bench set up to output a 3.3V DC input signal and a jumper attached on pins 1 and 2 on the jumper J42. After these setup steps were performed, the barrel jack connector was attached to the board. The first test performed during this section of testing was reading the voltage present at jumper J28 and ensuring the reading was 0V. Once this was verified, a load resistor with a resistance of 48 ohms was attached to the jumper J28 using clips, with one terminal of the resistor connected to J28, and the other terminal of the resistor was connected to ground. The 3.3V DC input supply provided by the Virtual Bench was connected to the jumper

J25 and turned on. The voltage and current present on the resistor were measured, and the voltage present on the resistor was verified to be 24V and the current flowing through the resistor was verified to not exceed 600 mA.

The next stage of our test plan was to ensure that the pushbuttons were producing a constant high when not in use, and a logic low when pressed down. The 24V DC supply was connected to the barrel jack connector, a multimeter set to watch the voltage levels that would be passed into the pin socket 5.3 which would act as a GPIO input to our MSP. The voltage at the pin should measure to be 3.3V while the pushbutton is not activated and should drop down to 0 upon activation. The current measured at the pin should never exceed 1mA.

The next stage of our test plan was to ensure the write functionality of our MSP to the LCD. To perform this test the 24V barrel jack should be connected to the AC/DC wall transformer to supply the 3.3V line for the MSP and LCD. Next, using the GPIO pins designated on the MSP, begin to send the sequence to turn on the screen with 4 rows enabled, and visually confirm that the screen turns on and has all rows enabled.

The next stage of our test plan was to attempt to communicate with the AD5933 by performing an I2C read on its control register, where the SDA line should return the byte 0xA0. To perform this test, connect the 24V AC/DC transformer to the PCB via the barrel jack connector, and using the MSP430, perform the I2C sequence of master read from slave device on the designated register address.

Software Test Plan

For testing the GPIO pins we used the Analog Discovery 2 (AD2) to view the outputs of the port pins that we wanted to use and then set them as high in the CCS library and made sure to view a signal output of 3.3 volts. We tested each pin this way to make sure that they delivered the desired result. We then tested the FSM that went along with the pin by making sure that after initializing them as off the system would update the threshold pins that the pins operated off of to decrement over time so that the signal would have to eventually be set to low. This was also a way for us to check that the clock was operating at the frequency that we expected and allowed us to test the clock interrupts since that is what decremented the value seen by the FSM each iteration. A simple screenshot of the scope channel validation can be seen in Figure 13.

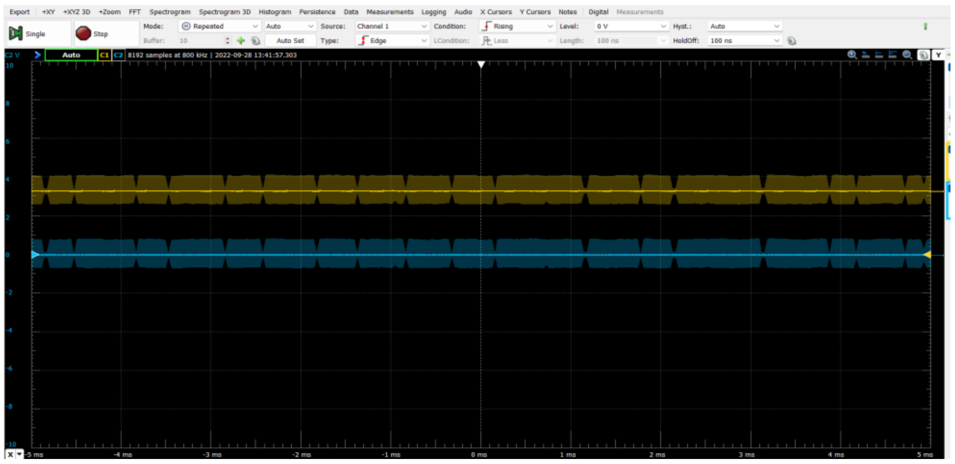

Mode: Repeated Auto
Buffer: 10
Source: Channel 1
Type: Edge
Condition: Rising
LCondition: Less
Level: 0 V
Length: 100 ns
Hyst.: Auto
HoldOff: 100 ns
8192 samples at 800 kHz | 2022-09-28 13:41:57.303

Figure 13: Screenshot of scopes from AD2 validating MSP430 GPIO pins

To test that the LCD code delivered what was expected we wired the LCD across a breadboard and connected it to the MSP430 and made sure that the input that was being created on the MSP430 is what actually displayed on the LCD screen. This test can be seen in the software subsection of the technical description section.

Final Results

The final result of the project fell short of our expectations. The main problems with the project were I2C functionality and distribution of voltage with a load attached. The I2C functionality with the AD5933 chip did not work properly due the lack of ability to test since the chip could only be completely tested with the PCB completed. This means we were unable to measure the moisture and nutrients of the soil. So we had to alter the code to function on fixed values. Due to this, we are unable to display the current levels of moisture and nutrients on the LCD. In addition to this issue, PCB was unable to supply voltage to all of the pumps so we had to create a voltage system which only allowed power to one of the pumps. But the system did meet some of the requirements in the proposal. The system was able to receive power from a wall socket and distribute nutrients and water based on the information received from the system. Overall, the system did work as expected for some of the core functionality but did not perform as expected for other aspects.

Costs

As can be seen in Appendix 1, the total cost of producing the prototype for this machine was around 306. The cost of components capable of being ordered from distributors such as Digikey came out to be 206 dollars, and an additional 100 dollars were spent at local hardware stores to create the chassis, mount the lightbulbs, and acquire the materials for general electrical work.

In the event that we were going to attempt to manufacture 10000 of these devices, many parts are already available at such a cheap price, costing pennies per item, that assessing the bulk savings in ordering them is hardly meaningful. Meanwhile for other components such as the MSP430, buying in bulk nearly halves the price per item, which would significantly reduce the overall cost as we attempt to assemble 10000 units. However, this does bring other issues and considerations. The ability to program a microcontroller without an associated launchpad is likely very different. There is no USB interface to just load or inject code onto, and from an assembly standpoint, having to program them 1 by 1 using a USB if that was an option would be unbelievably slow. The cost of assembly is unknown given our limited understanding of the assembly process, but it is something that we should take into consideration

Future Work

One of the biggest improvements that could be made to the project is using a different type of system to take in and display user input. The LCD and push-buttons that the project used did allow users to navigate through options and select certain water and light levels but never allowed them to choose anything except the premade options. We would suggest using a system that could communicate better with the microcontroller that the new team decides to use that would allow the user to give an actual figure that they wanted to use and then read that information and use that as the threshold for light, water, and nutrient levels.

When we constructed our board, we decided to use push buttons to allow the user to select plots, light, and water levels. Using push buttons to make these selections was a fine choice but we designed everything on one board which forced us to have all of our circuitry exposed since the buttons needed to be displayed for the user to actually use. We would suggest looking at what parts of the board need to be exposed to the user and which parts do not and building multiple boards to better separate the two.

A very reasonable way to improve upon our project would be attaching a wireless API that contained multiple plants and their nutrient, water, and light levels that the user could select from which would select much less arbitrary numbers than the ones we selected and allow users to look up certain plants that they wanted to grow without having to guess. This system could also be set up so the user could use their phone or computer or some other Bluetooth device to select plots instead of the LCD and push-button method that we went which would allow them to skip over some of the biggest concerns that were already listed.

Many of the unforeseen issues that we encountered were due to the chip we used to measure soil impedance (AD5933). This chip had difficulty using the I2C libraries and also returned real and complex data that then had to be converted to actual data which took time and resources to process. The real issue with this chip though was that it was impossible to test before being placed on the board and it was quite expensive. We would suggest staying away from the AD5933 and trying to find a new option. If this is the best option, we recommend finding a way to test it early such as building the board for this component early on and then attaching it to the system later.

References

[1] “MSP430G2553 | MSP430G2x/i2x | MSP430 ultra-low-power MCUs | Description & parametrics.” [Online]. Available: http://www.ti.com/product/MSP430G2553. [Accessed: 05-Sep-2022]

[2] “WO2017103922A1 - autonomous plant growing system,” *Google Patents*. [Online]. Available: https://patents.google.com/patent/WO2017103922A1/en. [Accessed: 12-Sep-2022].

[3] “US20150007495A1 - autonomously controlled Greenhouse Cultivation System,” *Google Patents*. [Online]. Available: https://patents.google.com/patent/US20150007495A1/en. [Accessed: 12-Sep-2022].

[4] IPC-2221A: Generic standard on printed board design, IPC: Association Connecting Electronics Industries, May 2003 [Online]. Available: http://www.ipc.org/TOC/IPC2221A.pdf [Accessed: 08-Sep-2022]

[5] IPC-A-600F: Acceptability of printed boards, IPC: Association Connecting Electronics Industries, Nov. 1999 [Online]. Available: http://www.ipc.org/TOC/IPC-A-600F.pdf

[6] IPC-4101C: Specification for base materials for rigid and multilayer printed boards, IPC: Association Connecting Electronics Industries, Aug. 2009 [Online]. Available: http://www.ipc.org/toc/ipc-4101c.pdf [Accessed: 08-Sep-2022]

[7] SMD Component Packages - Surface Mount Device (SMD) components conform to industry standards outlined by Surface Mount Technology (SMT) packages.

[8] Embedded C Coding Standard - The Embedded C Coding Standard authored by Michael Barr was used to accelerate the software development process and avoid potential bugs. Some standards set by Barr include comment rules, white space rules, and statement rules.

[9] “UM10204 I2C-bus specification and user manual,” vol. 2014, p. 64, 2014.

[10] RoHS Guide. “RoHS compliance FAQ.” rohsguide.com. https://www.rohsguide.com /rohs-faq.htm [Accessed 16-Nov-2022].

[11] SMART GARDEN DEVICES AND METHODS FOR GROWING PLANTS, W. Michael Bissonnette, B, (2006, Dec 7)US 2006/0272210 A1[Online] Available: https://patents.google.com/

[12] SYSTEM AND METHOD FOR GARDEN MONITORING AND MANAGEMENT, MONITORING AND MANAGEMENT, (2015, Jun, 18) US 2015/0164009 A1[Online] Available: https://patents.google.com/

[13] INTELLIGENT GARDENING SYSTEM AND METHOD, (2016, Mar, 1)US 9,271.454 B1[Online] Available: https://patents.google.com/

[14] “Peristaltic pump w/ silicon tube,” Adafruit Industries LLC. https://www.digikey.com/en/products/detail/adafruit-industries-llc/1150/5638299 [Accessed 05-Nov-2022]

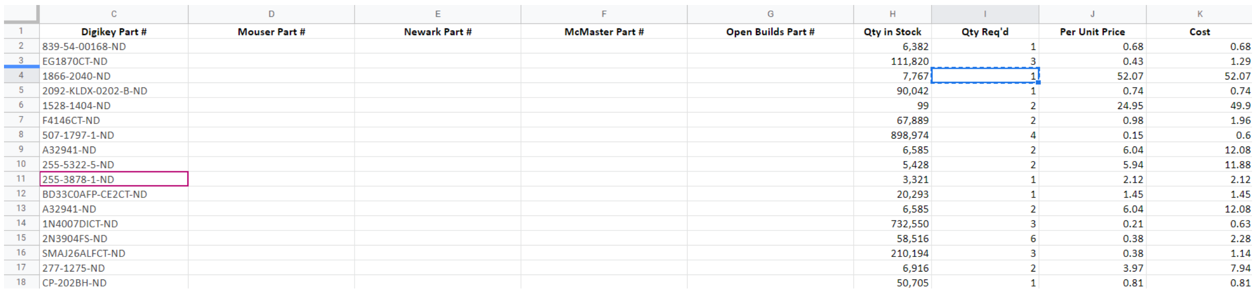
[15]“BTN8962TAAUMA.” Infineon Technologies. https://www.digikey.com/en/products/detail/infineon-technologies/BTN8962TAAUMA1/4772018?s=N4IgTCBcDaIE YBcB2AOAnANggXQL5A [Accessed 05-Nov- 2022].

[16] *“ACM 2004 F Series.”* A-Z Displays Inc. https://www.azdisplays.com/PDF/acm2004f.pdf [Accessed: 10-Oct-2022]

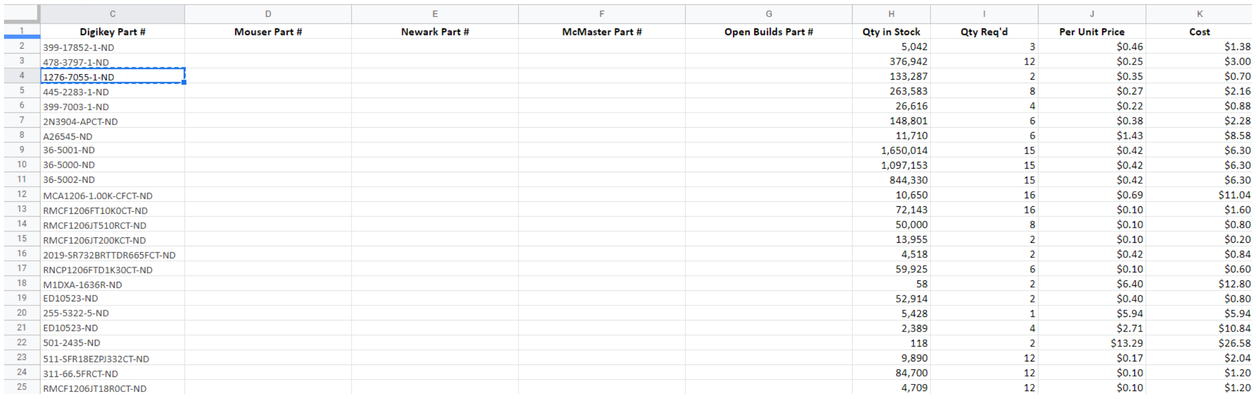
[17] “AD5933YRSZ-REEL7.” Analog Devices Inc. https://www.analog.com/media/en/technical-documentation/data-sheets/AD5933.pdf [Accessed: 21-Oct-2022]

Appendix

Appendix 1:



	C	D	E	F	G	H	I	J	K
1	Digikey Part #	Mouser Part #	Newark Part #	McMaster Part #	Open Builds Part #	Qty in Stock	Qty Req'd	Per Unit Price	Cost
2	839-54-00168-ND					6,382	1	0.68	0.68
3	EG1870CT-ND					111,820	3	0.43	1.29
4	1866-2040-ND					7,767	1	52.07	52.07
5	2092-KLDX-0202-B-ND					90,042	1	0.74	0.74
6	1528-1404-ND					99	2	24.95	49.9
7	F4146CT-ND					67,889	2	0.98	1.96
8	507-1797-1-ND					898,974	4	0.15	0.6
9	A32941-ND					6,585	2	6.04	12.08
10	255-5322-5-ND					5,428	2	5.94	11.88
11	255-3878-1-ND					3,321	1	2.12	2.12
12	BD33C0AFP-CE2CT-ND					20,293	1	1.45	1.45
13	A32941-ND					6,585	2	6.04	12.08
14	1N4007DICT-ND					732,550	3	0.21	0.63
15	2N3904FS-ND					58,516	6	0.38	2.28
16	SMAJ26ALFCT-ND					210,194	3	0.38	1.14
17	277-1275-ND					6,916	2	3.97	7.94
18	CP-202BH-ND					50,705	1	0.81	0.81



	C	D	E	F	G	H	I	J	K
1	Digikey Part #	Mouser Part #	Newark Part #	McMaster Part #	Open Builds Part #	Qty in Stock	Qty Req'd	Per Unit Price	Cost
2	399-17852-1-ND					5,042	3	$0.46	$1.38
3	478-3797-1-ND					376,942	12	$0.25	$3.00
4	1276-7055-1-ND					133,287	2	$0.35	$0.70
5	445-2283-1-ND					263,583	8	$0.27	$2.16
6	399-7003-1-ND					26,616	4	$0.22	$0.88
7	2N3904-APCT-ND					148,801	6	$0.38	$2.28
8	A26545-ND					11,710	6	$1.43	$8.58
9	36-5001-ND					1,650,014	15	$0.42	$6.30
10	36-5000-ND					1,097,153	15	$0.42	$6.30
11	36-5002-ND					844,330	15	$0.42	$6.30
12	MCA1206-1.00K-CFCT-ND					10,650	16	$0.69	$11.04
13	RMCF1206FT10K0CT-ND					72,143	16	$0.10	$1.60
14	RMCF1206JT510RCT-ND					50,000	8	$0.10	$0.80
15	RMCF1206JT200KCT-ND					13,955	2	$0.10	$0.20
16	2019-SR732BRTTDR665FCT-ND					4,518	2	$0.42	$0.84
17	RNCP1206FTD1K30CT-ND					59,925	6	$0.10	$0.60
18	M1DXA-1636R-ND					58	2	$6.40	$12.80
19	ED10523-ND					52,914	2	$0.40	$0.80
20	255-5322-5-ND					5,428	1	$5.94	$5.94
21	ED10523-ND					2,389	4	$2.71	$10.84
22	501-2435-ND					118	2	$13.29	$26.58
23	511-SFR18EZPJ332CT-ND					9,890	12	$0.17	$2.04
24	311-66.5FRCT-ND					84,700	12	$0.10	$1.20
25	RMCF1206JT18R0CT-ND					4,709	12	$0.10	$1.20