Auto-Adjustable Micro-Terrarium

By

Colin Lu (colinlu2)
Joyce Hu (xhu43)
Robert Vitek (rvitek2)

Final Report for ECE 445, Senior Design, [Fall 2020]
TA: Shaoyu Meng

09 December 2020
Project No. 05

Abstract

The microterrarium's objective is to create an optimal growth environment for any plant. It comprises four submodules which interface to control the internal environment: the power unit, control unit, environment regulator unit and physical display unit. The environment regulator adjusts the environment to an ideal range specified by the microcontroller, which in turn is fed data from our sensor components. The resulting data is output to the LCD display embedded in the display case which composes the display module. The objective of this report is to explain the motivations and process in which our microterrarium is designed and implemented.

Contents

1. Introduction

The COVID epidemic has inspired millions of people to try their hand at gardening while following shelter-in-place regulations; some statistics even show that over half of Americans have or are dabbling in some sort of gardening or lawn care[1]. However, many new gardeners may experience difficulties growing the plants they want due to inexperience or lack of proper equipment and home setups, which may severely limit the variety of plants they are able to grow in their homes. The main motivation for our group this semester was to address these issues by designing and building a self-regulating microterrarium that will auto-adjust the environment based on user input parameters. This would eliminate the need for gardeners to micromanage their plants by managing the environment of the plants for them instead. These environmental factors include soil moisture, lighting, soil pH, and temperature, which are the most important factors in determining the growth rate of a plant.

The design processing introduced at the upcoming chapter explains the design of each individual system and how they work together.

2 Outline of Subject Matter

2.1 Design Block Diagram

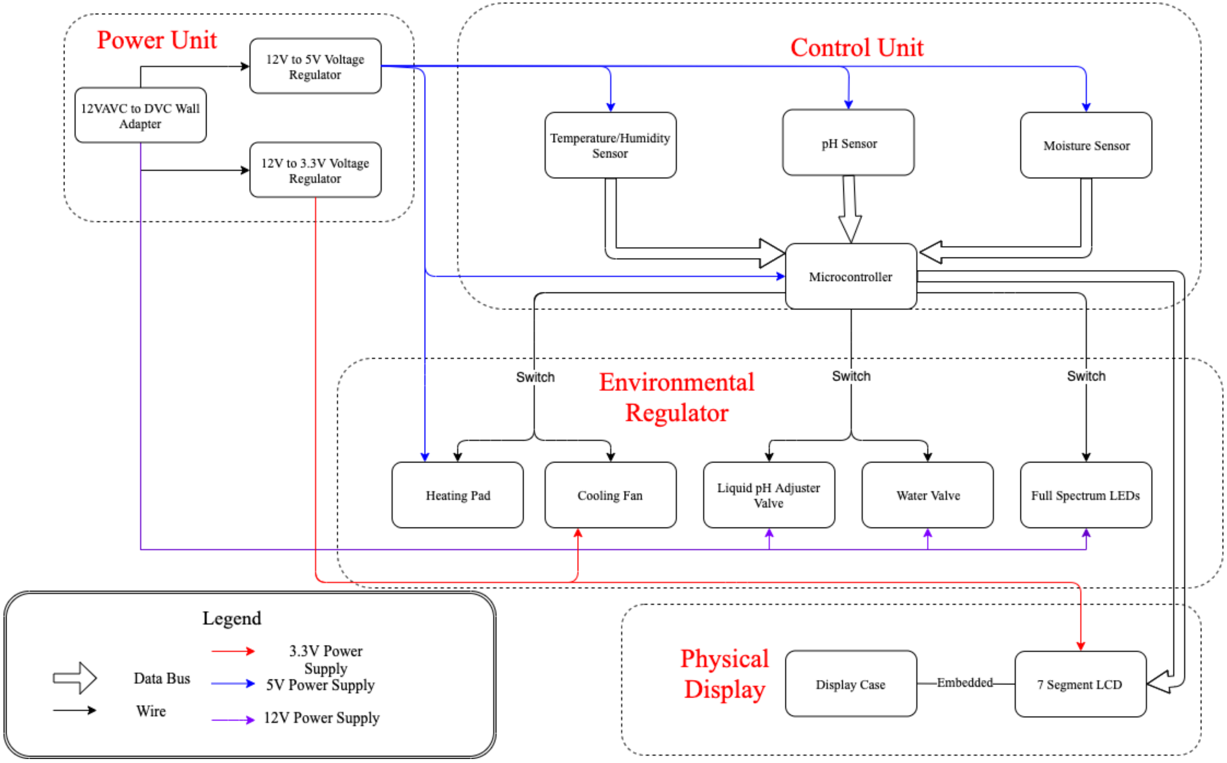

Figure 1 Design Block Diagram

Figure 1 above shows the block diagram of our completed project. The logic behind the finished microterarrium is comprised of four subsystems that interface with each other to accomplish our design goals. The power circuit takes the wall outlet voltage and steps it down to 5V and 3.3V which is necessary to supply the rest of the components, namely the control unit block, the environmental regulation block, and the display/user interface block. The control unit block comprises of our three sensors -- pH, moisture, and temperature/humidity -- which will then feed the data into the microcontroller, which will output signals to control the switches to enable the corresponding environmental regulation unit. Finally, the internal state of the microterrarium is displayed on our dot-matrix LCD display in our display block. The LCD as well as all our hardware components are then embedded into our physical container.

Our microterrarium design went through several modifications throughout the duration of our project. Firstly, we decided to eliminate the light sensor due to the fact that most plants prefer 16 hours of sunlight, something that can be directly controlled by using the delay capabilities of our microcontroller to time the “on” duration of our full spectrum LEDs. This reduced some unnecessary complexity in our project. We also were unable to implement a touchscreen LCD due to handshake issues between the microcontroller and touchscreen. Due to time constraints we replaced the component with a dot-matrix display instead, which also reduced the complexity of our project, but unfortunately removed any user-input functionality our microcontroller had.

2.2 Design

2.2.1 Power Unit Design Procedure

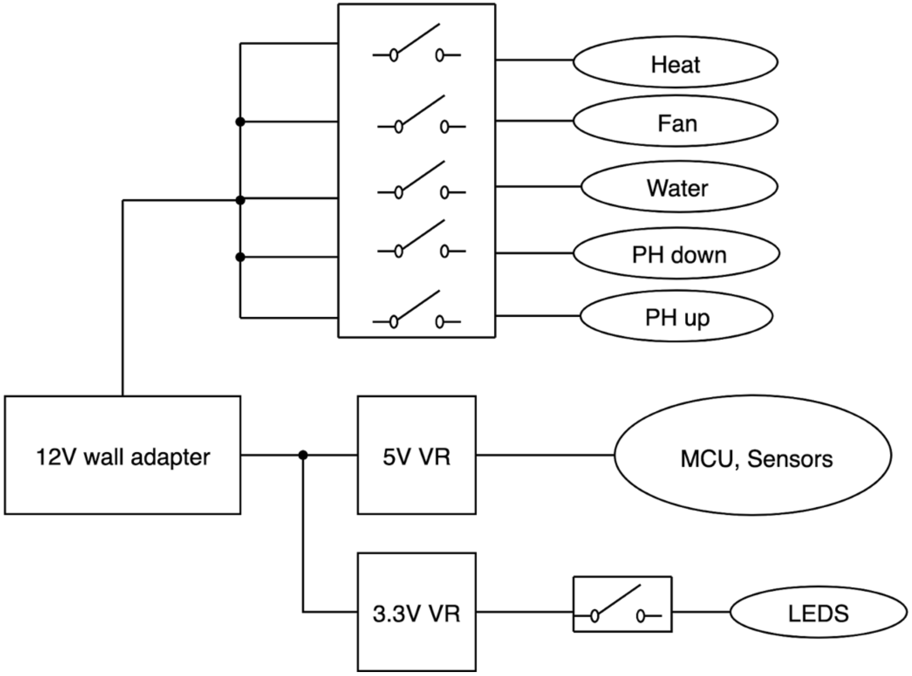

Figure 2 Power unit block diagram

The power unit needs to provide energy to make sure that the sensors, microcontrollers, and environment regulators can continuously work. We need three different power sources to supply the whole system: 12V, 5V and 3.3V. We can choose any of the three as the input voltage then boost up or step down to the desired level. Since the operation voltage for most parts is 12V and the convenience of the user to use at home, we decided to use a 12V wall adaptor as the input source. We chose the LM317 voltage regulator to step down to 5V and 3.3V because LM317 has a large range of input voltage from 1.5V to 37V. The load and line regulation of LM317 is very small to secure the stability of the output voltage and LM317 is very easy to implement.

$$\mathrm{Vout} = \mathrm{Vadj} * \left(1 + \frac{\mathrm{R2}}{\mathrm{R1}}\right) + \mathrm{Iadj} * \mathrm{R2} \qquad \text{Equation 1}$$

where Iadj is around 50uA which is small enough to negligible and We can simply equation 1 to equation 2[2].

$$\mathrm{Vout} = \mathrm{Vadj} * \left(1 + \frac{\mathrm{R2}}{\mathrm{R1}}\right) \qquad \text{Equation 2}$$

Where Vadj is equal to 1.25V. The Vout is easy to build up since it only depends on two resistors.

2.2.2 Power Unit Design Detail

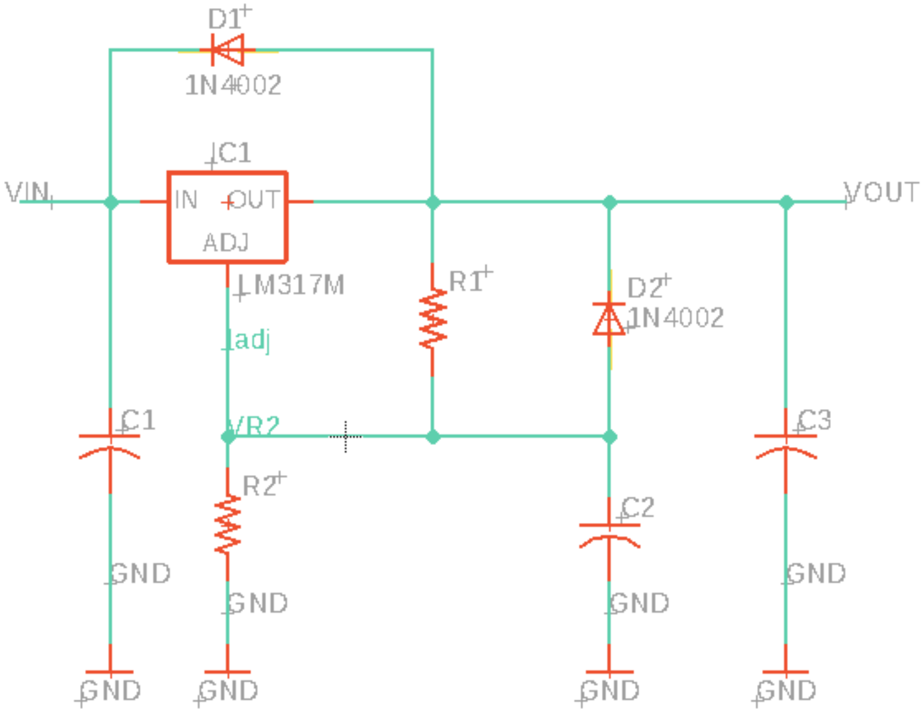

Figure 3 Eagle schematic of voltage regulator

Capacitor C1 provides sufficient bypass. C2 helps to stabilize the voltage at the adjustment pin, which helps reject noise. Diode D1 exists to discharge C3 in case the output is shorted to ground. C3 improves transient response. Diode D2 provides a low-impedance discharge path to prevent the capacitor C2 from discharging into the output of the regulator [2].

Table 1 Design values for voltage regulator

VIN(V)	C1(F)	C2(F)	C3(F)	R1(Ω)	R2(Ω)	VOUT(V)
12	0.1u	10u	1u	240	720	5
12	0.1u	10u	1u	240	394	3.3

2.2.3 Control Module Design Procedure

The control module comprises the microcontroller and sensor interface which is responsible for detecting the microterrarium environment and outputting the appropriate signals to its environmental regulator switches. The most important functionality of this block is that the sensors constantly detect and send data to the microcontroller for processing, with a delay of no more than 1 second. This is due to the fact that the microterrarium needs to be kept in a near constant state and can fall no more than ± 5% of the user's input range.

For our sensors, we had several options available but opted to prioritize cost-effectiveness first and ease of implementation second. Our first temperature/humidity sensor we considered was the HDC1010 all-in-one which had an easy to implement I2C peripheral; however, it was much more expensive than the sensor we finally decided to go with, the DHT11[3]. The tradeoff was that the latter was more difficult to implement due to a more complex data transmission protocol. In addition, the soil moisture sensor SEN-13322[4] delivered an analog signal and was well documented. Thus, these sensors were easy to implement following guidelines given in documentation.

For our pH sensor we did not have many options to choose from, as pH sensors are usually very expensive, and the majority were outside the scope of our budget. This made sense since the vast majority of commercial terrariums we researched online had no pH adjusting capabilities, and it was infeasible to balance cost-effectiveness. Fortunately we were able to find a relatively cheap pH meter online and reverse engineer its detection capabilities[5].

The microcontroller we ultimately decided to go with was the ATMega32L, as it could accommodate for all the functionality we needed to implement for our project, namely analog-to-digital signal conversion, digital signal processing, and the ability to control the voltage output on output pins in order to enable and disable the switches without needing another step-down voltage converter. Furthermore the price was extremely reasonable, and there is a lot of documentation online for ease of implementation and programming.

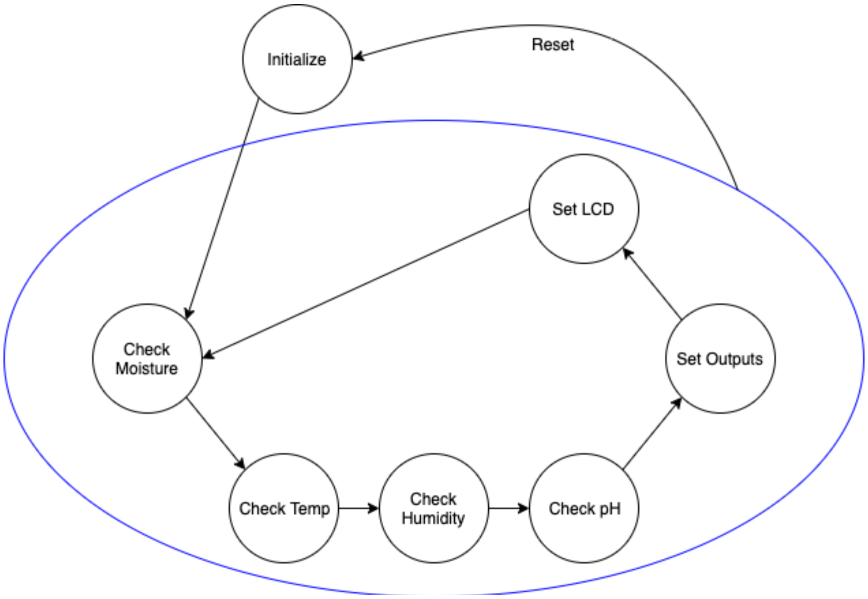

Figure 4 Microcontroller State Machine

In regard to the microcontroller implementation, we realized we needed to design the logic behind the code as a state machine. The microcontroller needed to either have discrete states which will sequentially detect sensors and update or have one state which idled and waited for changes in sensor readings to manipulate the regulator switches. The latter implementation, while less complex codewise, would be much more problematic and may run into issues with race conditions during sensor detection which may negatively impact our circuit and cause power issues. Thus we opted for the first implementation, and our final state machine sequentially checks readings and updates after all sensors are read from. A reset interrupt is enabled which will reinitialize the microcontroller at any point.

2.2.4 Control Module Design Details

In order to implement our control module, we first individually unit tested all our sensors after supplying them with a 5V power source. In order to reverse engineer the pH sensor, we manipulated three samples of soil to have pH values of 6.5, 7.0, and 8.0. Next, we attached a multimeter to the metal probes of the pH meter, and we measured the voltage across the probes for each sample of soil. The average measured value of voltage can be found in Table 2.

Table 2 Measured voltage across probes and corresponding pH of soil sample

pH of Soil	Average Measured Voltage across Probes
7.0	-0.0021 V
8.0	0.6035 V
6.5	-0.2998 V

Plotting this data, as can be seen in Figure 4, we arrived at an equation that could convert measured voltage across the probes to a pH value:

$$\mathrm{pH} = 0.602 * \mathrm{V(probes)} - 4.22 \qquad \text{Equation 3}$$

This process allowed us to incorporate pH sensing technology into the microterrarium, a feature unique to our design that separates our product from competitors.

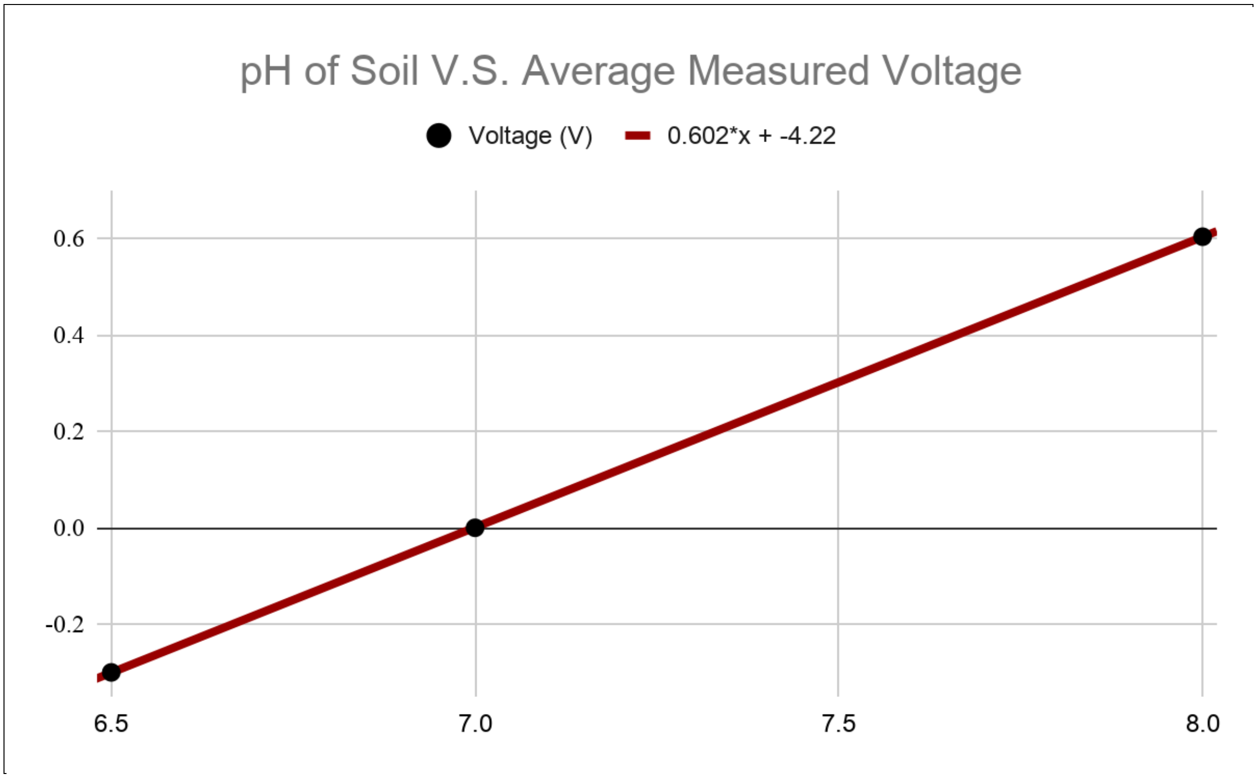

Figure 5 pH of soil sample versus measured voltage across probes

After confirming the sensors were working as intended, we needed to make sure they could interface with the microcontroller which would then process the data and output information to the LCD display and manipulate the regulator switch controls.

There are three main tasks the microcontroller needed to accomplish, namely analog-to-digital conversion, digital signal packet parsing, and LCD communication. ADC must be implemented so that the pH and moisture sensors will be able to communicate with the microcontroller, as they output analog signals. This can be accomplished by initializing the ADMUX and ADCSRA registers on the microcontroller, setting the ADC enable pin to high, and polling the data bus until the pin drops to low again[7].

```
    ADMUX=0b11000000; //Initializes the input pin to read from
    ADCSRA=0b10000111;

    ADCSRA|=(1<<6); //Set bit 6 n ADCSRA to start conversion
    while(get_bit(ADCSRA,6)==1); // poll ADCSRA till it is back to zero again
    ADCOut=ADCL|(ADCH<<8); // Save the ADC reading into an integer variable
ADCOut.
```

Figure 6 ADC Code Snippet

As shown in Figure 6 the two registers are first initialized, and then the data on the pin associated with ADMUX is polled and the result is saved. ADMUX will change initialization values depending on which sensor on portA of the microcontroller needs to be polled.

To parse digital data for our temperature/humidity sensor, we first needed the microcontroller to signal to the sensor that it was requesting data by toggling the corresponding pin. The data packets are then parsed into five fields, the temperature integer value, the temperature decimal value, the humidity integer value, the humidity decimal value, and a checksum which compares the total of the first four packets to itself to make sure no data is lost during transmission. Our implementation of this functionality is shown below in Figure 7.

```
Request();    /* send start pulse */
Response();   /* receive response */
I_RH=Receive_data();  /* store first eight bit in I_RH */
D_RH=Receive_data();  /* store next eight bit in D_RH */
I_Temp=Receive_data();  /* store next eight bit in I_Temp */
D_Temp=Receive_data();  /* store next eight bit in D_Temp */
CheckSum=Receive_data();/* store next eight bit in CheckSum */
```

Figure 7 DSP Code Snippet

After we made sure our sensor data handling works as intended, a bias voltage of 3.3V is then fed to the output switches for our environmental regulator and data is passed to the LCD to parse.

2.2.5 Environmental Regulator Module Design Procedure

The environment regulator module adjusts the environment based on the output of the MCU. Water and PH solution tanks are controlled by solenoids to manage the moisture and pH level of the soil. The temperature of the environment increases by turning on the heating pad and decreases by turning on the cooling fan. The operation of each regulator is controlled by a switch. We chose TPS22810 as the switch module because it handles up to 2A of continuous current and has a larger input voltage range: 2.7V to 18V. The operation of the switch simply depends on the enable pin. When EN is high, VOUT is equal to VIN, otherwise VOUT is equal to 0.

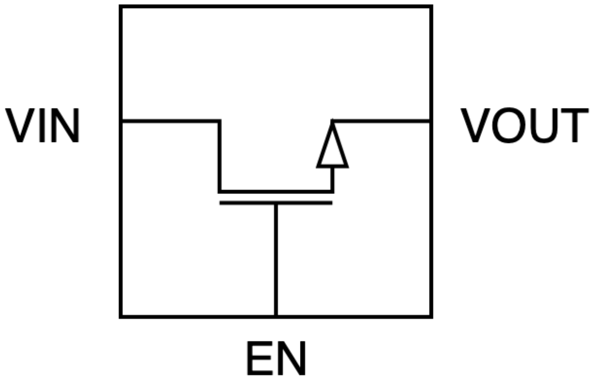

Figure 8 Simple Switch Schematic

2.2.6 Environmental Regulator Module Design Details

The solenoid is normally closed to block the liquid flow and it turns on when a 12V source is applied. Since it takes time for the liquid to immerse the sensor, we let the valve only open for 5 secs when the sensor reads high. Then it waits for 5 mins for the next open base on the signal of the sensor. The optimal light intensity is 200 to 600 $umolm^{-2}s^{-1}$for the best growth of the plant and each of the LED's intensity is 100 $umolm^{-2}s^{-1}$. Therefore 3 LEDs are sufficient. The heating pad and cooling fan are off until the sensor reads a desired valve.

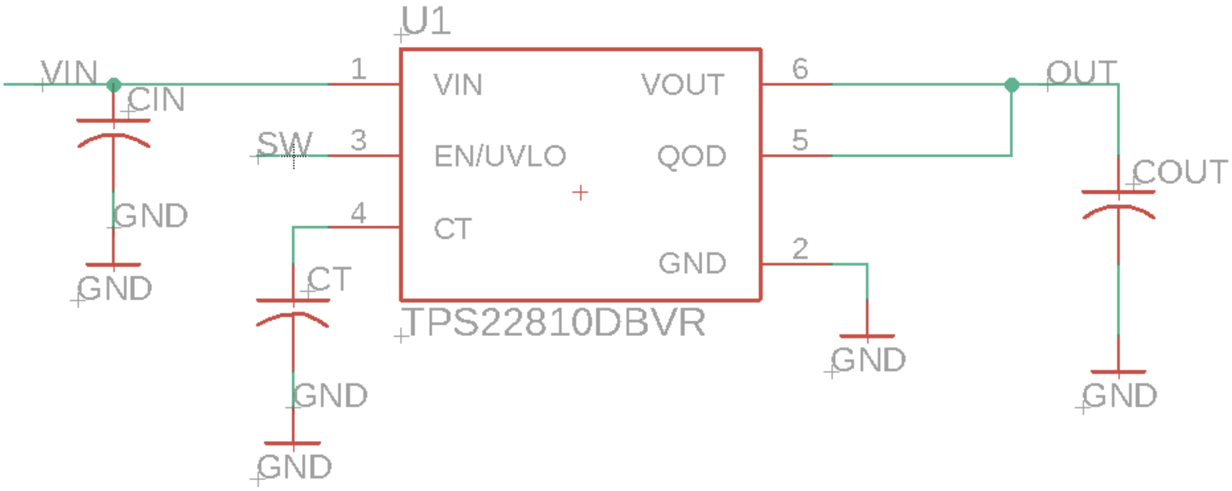

Figure 9 TPS22810 Eagle schematic

Capacitor CT sets the slew rate of the switch, since the environment of the terrarium is normally constant, and the sensor continuously reads the signal. Thus the system does not require the switch to have an extremely quick response.

$$\mathrm{SR} = \frac{46.62}{CT} \qquad \text{Equation 4}$$

SR is the slew rate (in V/µs) and CT is the capacitance value on the CT pin (in pF). Pin QOD is shorted with VOUT. This allows the discharge rate after the switch becomes disabled to be controlled by the value of the internal resistance RPD. CIN limits the voltage drop on the input supply due to the inrush current. A COUT: CIN with ratio 1:10 is recommended to minimize VIN dip caused by inrush current[6].

Table 3 Design value for switch module

CIN(F)	COUT(F)	CT(F)
1u	0.1u	2.2n

2.2.7 Display Module Design Procedure

Our display module consists of the physical display case and the dot-matrix LCD display embedded into the case. The final design of the microterrarium case came after a few revisions we made to improve the functionality of our design. Firstly, the material we used for the walls of the microterrarium was extremely important, as it was central to making sure heat and moisture did not escape the interior of the case too quickly. We considered four potential materials -- Polycarbonate, Polyethylene, Glass, and Plexiglass -- and compared their price per cubic inch to their thermal conductivity. Table 4 summarizes our findings.

Table 4 Summary of Material Cost and Conductivity

Material	Cost $/in3	Thermal Conductivity W/(m*K)	Density kg/m^3	Heat Capacity J/(kg*K)
Polycarbonate	0.429	0.20	1.21 x 10^3	1.25 x 10^3
Polyethylene	0.271	0.44	0.94 x 10^3	1.90 x 10^3
Glass	0.147	0.80	2.50 x 10^3	0.84 x 10^3
Plexiglass	0.336	0.18	1.18 x 10^3	1.45 x 10^3

As shown in Table 4 above, Plexiglass is the most insulating material while having a reasonable price tradeoff. Thus we ultimately decided to use it for our display panels.

Another aspect of our microterrarium is the physical appearance. Because terrariums are usually designed to be aesthetically pleasing, we wanted to make sure the circuitry did not get in the way. Therefore we mounted our PCB underneath the lid and hid the circuitry and tubing behind the case, and covered the back with a solid panel. Originally we wanted to place the tubing and PCB within a base underneath the microterrarium or directly beneath the lid, but due to awkward spacing of the solenoids as well as physical design constraints we were unable to.

User interactivity was another feature we wanted to implement for our terrarium. Originally, we had wanted to use a touch screen display in order to incorporate user input into our project while maintaining an aesthetically pleasing, slick design. Thus, we had purchased the NHD-1.8-128160EF-CTXI#-FT, a resistive touch, 1.8” TFT display. However, upon working with the display, we found little documentation and experienced severe struggles implementing the communication protocol. Consequently, we made a design decision to step away from our original plans and instead chose to implement a well-documented, common 16x2 1602A LCD display.

2.2.8 Display Module Design Details

The NHD-128x160-EF communicates with the microcontroller similarly to the sensors, through the idea that a handshake needs to be established before any data can be transmitted[8]. However, the protocol is more complex because the handshake involves three signals: read/write, enable, and command/data.

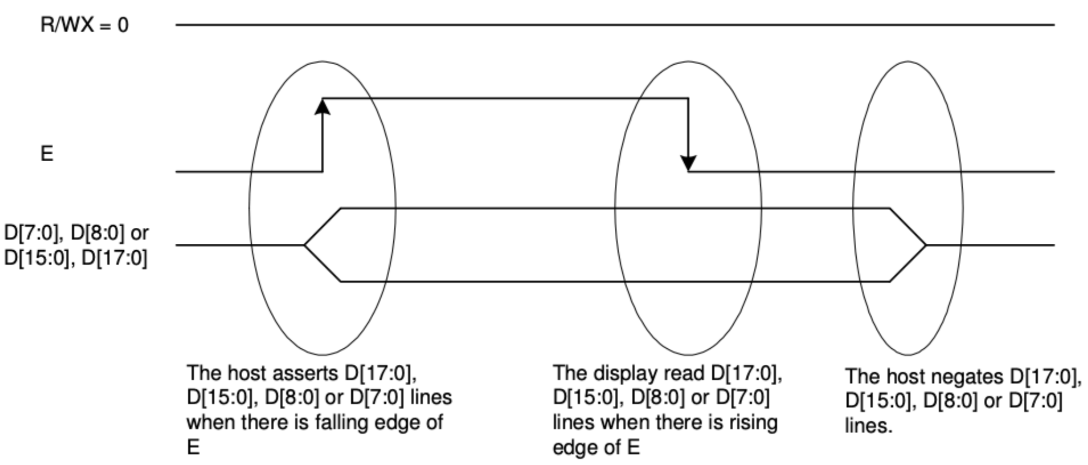
R/WX = 0
E
D[7:0], D[8:0] or D[15:0], D[17:0]
The host asserts D[17:0], D[15:0], D[8:0] or D[7:0] lines when there is falling edge of E
The display read D[17:0], D[15:0], D[8:0] or D[7:0] lines when there is rising edge of E
The host negates D[17:0], D[15:0], D[8:0] or D[7:0] lines.

Figure 10 Write Sequence Example

Figure 10 shows an example of the handshake protocol the NHD-EF uses, in this case a write to the touchscreen LCD [9]. We set the R/WX pin to low and toggle the enable pin which would allow data to be read from the bus. The command/data signal tells the microcontroller whether a command or data is being transmitted, with commands usually being defined by an opcode, and any necessary subsequent data defined by the opcode’s functionality. Unfortunately, due to time constraints we were unable to debug the handshake and decided to scrap the user interactivity aspect.

The display is well documented and easily wired. Therefore, we implemented the user interface using the guidelines documented by the manufacturer. One drawback of choosing this display is that it eliminated the user interaction facet of our product. Had we made the switch to the 1602A earlier in the project’s progression, we would have been able to add a button and keypad interface to the design of the PCB and product. However, with no time to order new PCBs or other parts, we forced ourselves to scrap the user input feature of the product and load the optimal conditions of the plant onto the microcontroller in order to demonstrate its capabilities.

2.3 Verification

2.3.1 Power Unit Verification

The result of the power unit soldered onto the actual PCB met our expectations. We were not able to find a resistor with an exact value of 720Ω and 394Ω, but we found 740Ω and 390Ω which is very close.

Table 5 Test values in PCB for voltage regulator

VIN(V)	C1(F)	C2(F)	C3(F)	R1(Ω)	R2(Ω)	VOUT(V)
12.08	0.1u	10u	1u	240	740	5.147
12.08	0.1u	10u	1u	240	390	3.281

The error percentage is calculated by

$$\text{error\%} = \frac{\boldsymbol{VOUT{,}test{-}VOUT{,}ideal}}{\boldsymbol{VOUT{,}ideal}} \qquad \text{Equation 5}$$

Table 6 Error value of the VR on PCB

VOUT,ideal(V)	VOUT,test(V)	Error(%)
5	5.147	2.94
3.3	3.281	0.5

2.3.2 Control / Display Unit Verification

In order to verify that the control unit worked as expected, we tested whether the MCU could correctly receive data from the sensors and display it on the 1602A. Figure 11 portrays one example of this verification; here, we test the soil moisture sensor, ADC converter, and MCU-display interface by reading soil moisture data and displaying it accurately on the 1602A.

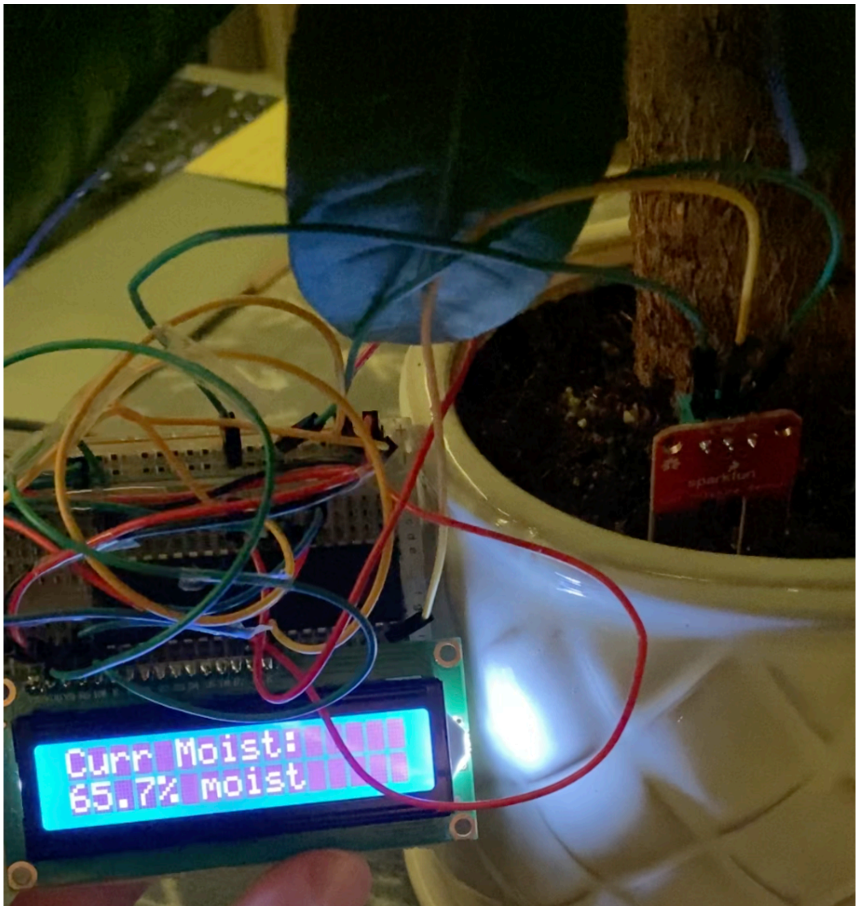

Figure 11 Reading and displaying soil moisture data

After verifying that the MCU could accurately read data from each sensor, we needed to verify that the MCU could step through the various states of the state machine both autonomously and in the correct order. To accomplish this, we decided it would be best to assign one output port of the microcontroller to each state of the state machine, and then connect LEDs to each output port. Then we programmed the MCU to step through the state machine, turning on each state's respective LED when it was performing the functions within that state. At all points of the process only one LED was on at a time, and the LEDs turned on in the correct order, correctly verifying the state machine.

2.3.3 Environment Regulator Unit Verification

We tested the solenoid by applying a 12V voltage supply and a water flow rate of 200mL in 11.25s. Within 5s of the solenoid opening, about 90mL liquid flows to the soil which prevents the soil from oversaturating. The flux intensity of 3 LEDs is about 294 FL and the LEDs operate 16 hours/day to provide enough light. Our 5V supply heating pads take a long time to increase the temperature of the terrarium environment, about 20 mins to increase the temperature of the terrarium by 1℃ with two heating pads. The operation current for a singular heating pad is 750mA, but the rate current for 5V VR is only 1.5A. Thus, to increase our number of heating pads, we need to implement more VR circuits within our circuit design. However, this factor may be mitigated with higher feed voltage into the heating pads; we tested the heating pad with 12V supply, and the temperature increased by 2.5℃ in 20mins, a much faster rate. In contrast, our cooling fans use much less current, about 30mA each. With them, we were able to decrease temperature by 2.5℃ within 15 minutes. We tested the switch by applying high and low signals to the enable pin and observed the output valve with a 12V input.

Table 7 Test values in PCB for Switch

SW(V)	VIN(V)	VOUT(V)
5	12.08	12.07
0	12.08	50.12m

The test result verified the switch circuit works well; the error of the output is less than 1%.

3 Cost

3.1 Cost analysis

As young engineers graduating from ECE at Illinois, we anticipate an hourly wage of around $40 during the research and development of our product. In addition, we plan to have the final prototype completed by the end of 12 weeks of work, with each team member contributing an average of 8 hours of work every week. Thus, the total labor cost of the development of the product, including overhead, is:

3 employees x $40/hour x 8 hours/week x 12 weeks x 2.5 (hypothetical overhead) = $28,800

The total costs of parts for both a single prototype as well as products manufactured in bulk are measured shown in Table 12. The total cost of development for the prototype, incorporating parts, labor, and overhead, is $28,923.96.

Table 8 Parts Costs

Part (Distributor)	Cost (Prototype)	Cost (Bulk)
Acrylic Display Case (Amazon)	$26.99	$26.99
5V DC Heating Pad (Digikey / COM11288) x 2	$7.90	$7.90
12V DC Cooling Fan (Digikey / HA40101V4-1000U-A99) x 2	$6.28	$3.91
LCD TFT Touch Screen Display (Digikey / NHD-1.8-128160EF-CTXI#-FT)	$15.04	$13.01
12V One-Way Solenoid Valve (Digikey / ROB-10456) x 3	$6.9	$6.9
Chanzon 1W LED Grow-Light (Amazon) x 10	$6.50	$6.50
Humidity / Temperature Sensor (Adafruit /DHT11)	$5	$4
ATMEGA32 8-bit MCU (Digikey)	$5.46	$4.53
Soil Moisture Sensor (Amazon)	$5.99	$5.99
Buck converter x 2 (TI/LM317)	$0.80	$0.80
Assorted wires, resistors, transistors, etc (Digikey)	$5.00	$0.50
pH Meter (Amazon)	$8.99	$8.99
Jack connector	$0.82	$0.40

(digikey/PJ-009AH)		
12V wall adapter (digikey/SW15-12-N-P5)	$6.95	$4.83
LCD 1602 Module(Amazon)	$6.49	$6.49
Atmega32 socket(Amazon)	$1.75	$1.75
Total	$116.86	$103.69

4. Conclusion

4.1 Accomplishments

The power unit supplies the desired voltage to all parts of the system. The control unit is able to communicate with all of the sensors and give correct order to the environment regulator. The majority of the functionality works well on the breadboard except the touch screen display. We were not able to figure out the touch screen LCD so we used a dot-matrix LCD instead which displays the correct sensor output. However, the dot-matrix LCD can only display the value of the sensor but not communicate with the user.

4.2 Failure

The PCB is not able to operate with the MCU we programmed, but the MCU works properly on the breadboard and the PCB works fine by manually applying voltage to simulate the MCU output.

4.3 Ethical considerations

One of our biggest goals for this project is to bring the joy and excitement of gardening to prospective gardeners by demonstrating how new technologies can aid them through the difficulty of maintaining plants. Thus, following the IEEE Code of Ethics #2, we intend to "improve the understanding by individuals and society of the capabilities...of conventional and emerging technologies" [10].

Due to the nature of our project, we are aware that users may use the terrarium to grow dangerous plants. This will contradict #4 and #9 of the IEEE Code of Ethics, which state respectively "to avoid unlawful conduct in professional activities" and "to avoid injuring others, their property, reputation, or employment by false or malicious actions.... or physical abuses"[10]. Unfortunately, because there will be no way to control what plants the user decides to grow in the terrarium, we believe that the benefit of the terrarium will outweigh the potential downsides of having a small malicious user base. We believe by disclosing these caveats that we comply with the IEEE Code of Ethics #6, "to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations" [10].

4.4 Future work

The main issue we were struggling with is we did not have the programmer set up on our PCB which made it extremely hard to debug the MCU. Furthermore the socket holder we had for the MCU chip does not function well; the MCU chip functioned properly when directly soldered on PCB but has not worked when the holder is soldered on PCB. Therefore, we plan to have a more robust PCB redesign in order to debug and troubleshoot our microcontroller more easily. The other important component we were unable to implement was the touchscreen LCD to communicate with the user and the MCU. We plan to also implement a more well documented touch screen LCD and potentially even introduce a bluetooth module. The bluetooth module would be able to directly communicate from a user's phone to the touchscreen LCD.

Reference

[1] Walljasper, Christopher, and Tom Polansek. “Home Gardening Blooms around the World during Coronavirus Lockdowns.” *Reuters*, Thomson Reuters, 20 Apr. 2020, www.reuters.com/article/us-health-coronavirus-gardens/home-gardening-blooms-around-the-world-during-coronavirus-lockdowns-idUSKBN2220D3?feedType=RSS.

[2] *LM317M 3-Terminal Adjustable Regulator*, datasheet, Texas Instrument, Inc., 2020. Available at: https://www.ti.com/lit/ds/symlink/lm317m.pdf?ts=1607543004374&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM317M%253Futm_source%253Dgoogle%2526utm_medium%253Dcpc%2526utm_campaign%253Dapp-null-null-GPN_EN-cpc-pf-google-wwe%2526utm_content%253DLM317M%2526ds_k%253DLM317M%2526DCM%253Dyes%2526gclid%253DCjwKCAiAiMLBRAAEiwAuWVggtkOUDFZg6glyVaTUujuuDyRBNDs95gUL6do06ykZuZFR7AtSbLP7xoClHgQAvD_BwE%2526gclsrc%253Daw.ds

[3] DHT22. (n.d.). Retrieved September 19, 2020, from https://cdn-shop.adafruit.com/datasheets/Digital+humidity+and+temperature+sensor+AM2302.pdf

[4] Soil Moisture Sensor: Sensors & Modules. (n.d.). Retrieved September 18, 2020, from https://www.electronicwings.com/sensors-modules/soil-moisture-sensor

[5] Amazon. (n.d.). PH Soil Meter. Retrieved September 17, 2020, from https://www.amazon.com/iPower-Moisture-Garden-Gardening-Outdoor/dp/B075LRY5M7/

[6] *TPS22810, 2.7-18-V, 79-mΩ On-Resistance Load Switch With Thermal Protection*,datasheet,Texas Instrument, Inc., 2020. Available at: https://www.ti.com/lit/ds/symlink/tps22810.pdf?ts=1607527362476&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTPS22810

[7] “ATmega32.” *ATmega32 - 8-Bit AVR Microcontrollers*, www.microchip.com/wwwproducts/en/ATmega32

[8] “ILI9163 Datasheet PDF.” *ILI9163 Datasheet | ILITEK - Datasheetspdf.com*, datasheetspdf.com/datasheet/ILI9163.html

[9] NHD-1.8-128160EF-CTXI#-FT. (n.d.). Retrieved September 18, 2020, from https://www.digikey.com/en/products/detail/newhaven-display-intl/NHD-1-8-128160EF-CTXI-FT/4429439

[10] IEEE Code of Ethics. (n.d.). Retrieved September 18, 2020, from https://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A Requirement and Verification Table

Table 9 Requirement and Verification

Requirement	Verification	Verification status (Y or N)
1. Voltage regulator circuits step 12V down to 5V and 3.3V ± 5%.	1. Use the oscilloscope to measure the output voltage of the converter to check whether the output voltage is within 5% of 5V and 3.3V for each respective circuit.	**Y**
2. Temperature sensor correctly reads temperatures within range 55-85°F	2. a. Verify reading is correct by looking at LCD display and comparing with thermometer within ± 5% b. Use multimeter to confirm packets are being sent to the microcontroller	**Y**
3. MCU correctly takes the pH meter's voltage as input and reads pH values from 3.5 to 8.	**3.** **a.** Check pH of soil using litmus paper and pH meter. **b.** Confirm that pH derived in MCU matches the measured value in part A within ± 5%.	**Y**
4. The soil sensor needs to output values within the range 0-5V to accurately detect soil moisture.	**4.** Test sensor range using a dry paper towel and gradually wetting it, the sensor should start by outputting 0V but gradually increase to 5V.	**Y**
5. Microcontroller must modify appropriate control modules each clock cycle.	**5.**Confirm that correct pins output a 5V± 5% when the corresponding subsystem of the environmental regulator must be turned on.	**Y**
6. a.. LEDs must stay on for 16 hours straight, then stay off for 8 hours (or whatever	**6.** a. Check that the LEDs automatically turn on/off after a given amount of time.	**Y**

lengths of time are needed, as commanded by the MCU). b.LED’s intensity must be 200 to 600 umolm-2s-1	b. Use the light intensity meter to check whether the light intensity is in the range of 200 to 60umolm-2s-1with ± 5% variation	
7. a. Display must correctly portray all sensor measurements of the environment. b. Interface must allow the user to input optimal ranges for plant growth and send to MCU.	**7.** a. Check what sensor values the MCU is reading and confirm that they match those displayed. b. Use the touch screen to input optimal temperature that is much higher than current range. Confirm MCU received input by checking if heating pads turn on.	**N**