Automated Hydroponic Greenhouse

Michael D'Anna, EE, Samantha de Groot, EE, Maxwell Joyce, CSE, and Shaun Palmer, EE

***Abstract*— The Gro-Pro completely automates the process of growing plants in a system designed to fit within one's home. It uses a Raspberry Pi to analyze readings from four different sensors and uses that data to optimize growing conditions for the desired plants being grown. Useful feedback is given to the user about the condition of the plants via an Android application, which also helps the user learn about gardening. It utilizes hydroponics, a more efficient method for growing plants than traditional potted soil. Everything from lighting and watering cycles to nutrient and pH regulation is automated, as to make it simple for anybody to grow plants. All you have to do is plant your seeds and press go. This paper discusses the operation and design of the project including the following subsystems: sensors, android application and control unit, lighting and voltage regulation, and hydroponics.**

I. INTRODUCTION

IT is estimated that the average American meal travels about 1500 miles to get from farm to plate [1]. This not only adds to the price of food, but has environmental consequences as well, especially when shipped via plane and when temperature control is required. The carbon footprint of shipping and distributing food plays a role in worsening the global warming crisis [12]. This is an issue around the entire world, as many places do not have climates conducive to growing the produce that the population wishes to consume throughout the year. As an alternative to obtaining produce from afar, people can buy locally or garden on their own. However, local produce is more expensive and availability is restricted by season and climate [13]. Gardening is time consuming, requires both space and knowledge and is again imposed by the seasons. The viability of these two options is further reduced in urban environments and food deserts, where there is a lack of yard space and farms are far away.

Our design aims to make it easy to grow produce locally within the bounds of one's own home. By utilizing an automated hydroponic design with the adaptability to cater to a wide variety of different plants, the user is able to grow produce worry free. Detailed in table 1 below are the specifications of the project. It is important that the greenhouse is automated and uses sensors to determine how to water, give light to and fertilize the plants through hydroponics, while requiring no action from the user. The android application makes it easy to monitor the status of the plants and sensor readings inside the greenhouse. Being a closed loop system, we are able to reuse the hydroponic solution over and over again as it runs through the plants, until it is depleted. Lastly, we aim to seek a balance between size and yield, being small enough to fit comfortably within ones home while also yielding 5-8 fruiting plants.

TABLE 1

Specifications
Automated: lighting, hydroponics, nutrient dispersal, pH regulation
Must fit inside studio apartment ~(2'x4')
Yield 5-8 fruiting plants
Closed loop system to recycle water
Reusable- able to grow new plants over and over again
Android app that has a simple UI and is easy for people to use

M. D'Anna from Westford, MA (e-mail: mjdanna@umass.edu).
S. de Groot from Marion, MA (e-mail: sdegroot@umass.edu).
M. Joyce from Hopkinton, MA (e-mail: mljoyce@umass.edu).
S. Palmer from Hopkinton, MA (e-mail: shaunpalmer@umass.edu).

II. DESIGN

A. Overview

Our design provides the user with a fully automated way to grow plants within their home. By using sensor readings from hygrometers, a temperature & humidity sensor, a pH electrode and float switches, along with information from a programmed plant database, our design can control all aspects of growing plants. This includes light cycles, watering cycles, giving the plants nutrients and controlling pH, all of which play an important role in a plants ability to grow effectively. All of the sensor data is relayed to a Raspberry Pi, which acts as the brain and controls the operation of the greenhouse. A Raspberry Pi is a small bare bones computer containing a processor, peripherals such as USB, SD and HDMI, and plenty of I/O pins that can be used to control the different parts of the greenhouse. By implementing a state machine in code, cycles are adjusted over time, as to cater to a plants specific needs as it grows. By using a drip hydroponic system, we can control watering precisely, operate at maximum efficiency and conserve water because it is a closed loop system. There are many other types of hydroponic and aeroponic options, including ebb and flow, deep water culture and wick hydroponics, but this was the most suitable one to automate and control. As far as lights go, our design uses LED fixtures rather than high pressure sodium lights. Despite the higher cost, LEDs are better suited because their distance from the plants does not need to be adjusted as the plants grow and heat from the fixtures is not a concern. They also grow plants more optimally due to the wavelengths of light they emit, which is discussed in the *Lighting and Voltage Regulation Section*. Our design is broken up into four main parts, which can be seen in the block diagram below (Fig. 1).

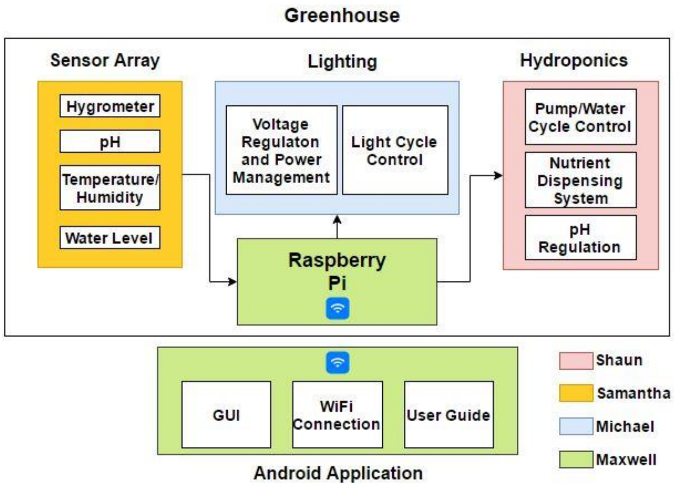


Fig. 1 System block diagram

The first block is the sensors, which Samantha is responsible for. It consists of interfacing the hygrometers, pH electrode, temperature/humidity sensor and float sensors with the Raspberry Pi and using them to control various other pieces. The hygrometer ensures that the plants are getting sufficient amounts of water and will increase watering cycles if the growing medium is dry. The pH sensor ensures that the pH of the nutrient-rich solution stays within a specified range and will control the release of an acid and bases to adjust pH. The temperature/humidity sensor will tell the user the conditions of the greenhouse. The water level sensors notify the user when basins need to be refilled and control the amount that pumps will fill given reservoirs. The sensors control and adjust what is delivered to the plants based on the plants needs and conditions.

The second block is the android application, which Max is responsible for. It is made up of: development of an android application, establishing communication between the Raspberry Pi and a phone via Wi-Fi, and compiling a plant database that will adjust lighting and watering cycles based on plant type. The application must be intuitive and easy for anyone to use. This block also consists of the control unit, which is the Raspberry Pi itself, and involves all team members. This is where the state machine that controls all aspects of the greenhouse will be implemented.

The third block is lighting and voltage regulation, which Mike is responsible for. It consists of light cycle control along with design of a custom PCB to be used for voltage regulation. The lights must be able to be turned on and off using the Raspberry Pi. The voltage regulator must supply different voltages across the greenhouse in order to power various sensors and solenoid valves.

The final block is hydroponics, which Shaun is responsible for. It is made up of pump control for watering plants, the nutrient dispersal system and pH regulation system. The pumps must be able to be controlled by the Raspberry Pi to turn on and off. One pump must move two gallons from a large reservoir to a smaller reservoir, while the other must be able to pump water through tubing to water the plants. The nutrient dispersal system must be able to reliably dispense 25 ml of nutrients into the small reservoir. The pH regulation system must reliably dispense 10 ml of acid or base into the small reservoir in order to adjust pH.

B. Sensors

The greenhouse makes use of four different types of sensors to help regulate plant cycles and give the user useful information to further aid the growing process. The four sensors are hygrometers, float sensors, a temperature & humidity sensor, and a pH sensor meter. Each sensor has either a digital or analog output. Once the sensor output data is successfully read, the data will be thoughtfully presented in graphs and tables to the user.

A hygrometer is a sensor which measures the moisture content within a growing medium. This sensor is useful in determining watering cycles of plants and ensuring each plant is receiving an equal amount of liquid. The model of hygrometer chosen for implementation is a resistive type. A resistive hygrometer is made up of two metal stakes which are inserted into the growing medium. The electrical resistance between the two stakes is measured and used to determine the moisture content of the medium. Due to the type of metal the hygrometer is made of, the stakes tend to oxidize if constantly powered and left in a saturated medium. For this reason, power to the hygrometers will be software-controlled and limited to turning on once an hour for only enough time to read an output. The hygrometer has an analog output which reads a number between 0 and 1024, with 0 being very saturated and 1024 being completely dry [2]. When read in, the number is stored as a percentage (moisture content) and is graphically compared with previous hygrometer readings. The hygrometers and code were tested by reading outputs of the sensor directly before and after a water cycle to determine if the readings were accurate. Based on the readings the hygrometers correctly measured the moisture content of the growing medium.

The greenhouse employs 4 float sensors, two in the small nutrient-rich solution basin, one in the large water basin, and one in the nutrient container. The float sensors use a magnet within the float part of the sensor to trigger the closing of a switch sending digital high/low output (see Appendix 1). The two switches within the nutrient-rich solution basin are used to control the pumping of water into the nutrient-rich solution basin by controlling the power to the water pump within the large water basin. One switch is secured at the bottom of the basin to turn on the water pump to begin filling the basin, and the other sensor is secured to the 2gal level in the basin to trigger the stop of the pump. The float switch within the large water basin is used to send a notification the user when the basin needs to be refilled. This sensor is put at a level within the basin which will allow the user up to 24 hours to refill the basin before the system runs out of water. The sensor within the nutrient container is used in the same way as the one in the large water basin. However, due to the small size of the nutrient container, the notification may give the user less than 24 hours to refill the container. The float sensor has been tested by connecting a 5V input to one end of the sensor and then, using a multimeter, checking the voltage at the other end of the sensor and triggering the switch.

There is one temperature & humidity sensor placed on the side wall of the greenhouse. This sensor waits for a start signal to be sent by the user and after ~2 seconds, sends out a serial digital output of 40 bits. The first 16 bits of the output are converted in code to a decimal number and then divided by 10 to represent the humidity percentage. The second 16 bits are interpreted similarly, however the decimal output represents the temperature in degrees Celsius. The last 8 bits of the output are the check-sum bits, which are used to ensure accuracy of the received bits. Part of the datasheet of this sensor module can be found in Appendix 2. The data from this sensor is not used to control any parts of the state machine, however it will be useful data for the user to have. This information is represented visually on the user's phone application. This sensor has been tested in a room of known temperature and humidity to determine the accuracy of the sensor.

Finally, the glass electrode pH sensor is mounted within the nutrient-rich solution mixture to ensure the pH levels are in the correct range (5.5-6.5) to stimulate plant growth. This sensor is needed due to the fact that the runoff water from the plants goes directly back into the solution tank. When the pH sensor measures a pH level below or above the desired range, a measured amount of an acid or base will be added into the basin and the pH sensor will check the levels again. The sensor will continue checking and triggering the addition of an acid or base until the pH level is back within the desired range. The sensor itself is made up of a glass electrode and a reference electrode, as shown in the figure below (Fig. 2).

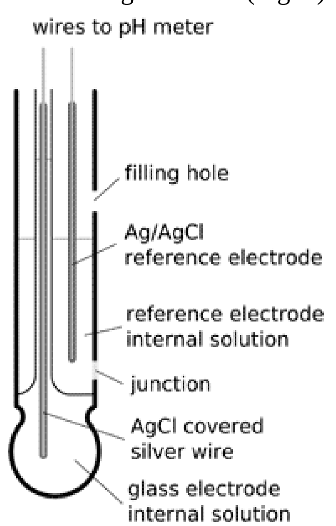


Fig. 2 Diagram of a pH electrode sensor

The pH of the solution the sensor is placed in is determined by comparing the voltage difference between the glass and reference electrodes [3]. When the glass electrode is inserted into the solution the hydrogen ion-sensitive solution within the glass electrode effects the voltage on the AgCl covered silver wire within the glass electrode based on the pH of the solution. The electrode is driven with 12V and outputs an analog signal. Since the Raspberry Pi being used can only read in a digital signal, an A/D converter is used to convert the output. The data from the pH meter is used to both control the input of acids and bases into the solution, but also the data is displayed to the user.

C. Hydroponics

The hydroponic system consists of everything needed to feed and water the plants. The subsystems included are the pump control, nutrient dispersal system and pH regulation. Our design makes use of a drip hydroponics system (shown in Figure 3 below).

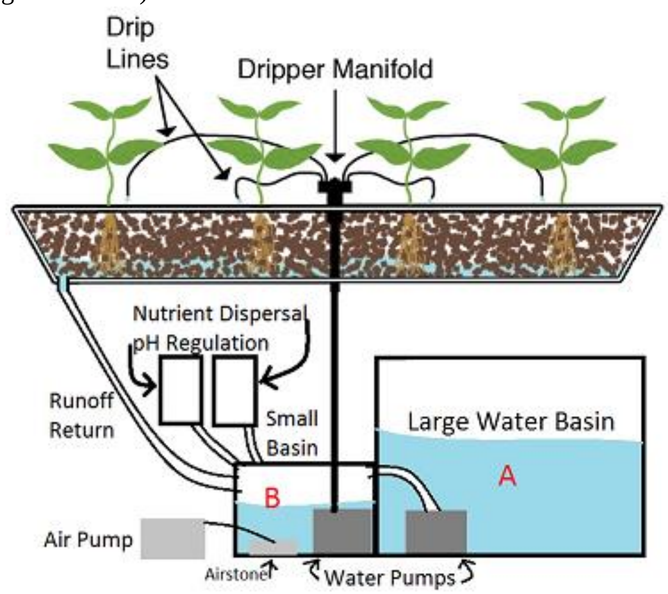


Fig. 3 Diagram of hydroponic setup

Two gallons of water from the large basin (A) is pumped into the small basin (B), where it is mixed with 25ml of liquid nutrients that is delivered from the nutrient dispersal system. The function of the airstone is to mix the nutrients and water, as well as provide dissolved oxygen in the mixture, which promotes healthy plant growth. This nutrient-rich solution is then pumped up to the plants periodically using another pump, through the dripper manifold, to the bases of each plant through the drip lines. The dripper manifold was fabricated in such a way that it forms a closed loop around all of the plants, as to ensure equal water pressure is being delivered to each of them. Lastly, the excess water that is pumped through the plants returns to basin B through the runoff return tubing. The plants are housed in an angled grow tray to allow excess nutrient solution to collect at one end and flow back into basin B. Each plant is potted in a fabric pot to allow the solution to drain through. The medium that the plants will be growing in consists of a 50/50 mix of coco coir and perlite. The advantages of this mix are as follow. It is completely reusable, so after the plants are done growing they can be taken out and new plants can immediately be planted. It is lightly packed, which gives roots access to more oxygen. It also retains water well, and makes use of a coconut husk, a repurposed byproduct of the coconut industry [8].

The system uses two pumps that are controlled by the Raspberry Pi. Since they both require AC power and are plugged into standard wall outlets, they can easily be turned on and off using the same relay circuit used to control the lights, as described in the *Lighting and Voltage Regulation* section.

The first pump is used to pump water from basin A to basin B. When it is time for basin B to be refilled, the pump in basin A is turned on until the float sensor triggers it to stop at the 2 gallon mark. The second pump is used to pump nutrient-rich solution from basin B up to the plants. This is done intermittently for short bursts of time; the watering cycles vary based on plant size, type, and stage in the growth cycle.

The two types of liquid nutrients that will be used in growing our plants are Dyna-Gro Foliage Pro (N-P-K 9-3-6) and Dyna-Gro Bloom (N-P-K 3-12-6). The N-P-K is the ratio of nitrogen, phosphorus and potassium, which specifies the amount of each of these key nutrients in the mixes [9]. The foliage pro will be used in the beginning while the bloom will be used towards the end to promote the most efficient plant growth. Based on the instructions, it is recommended that 25ml of nutrients (since it is so concentrated) is mixed with 2 gallons of water, so the nutrient dispersal system was designed to dispense that volume of liquid to be mixed with the 2 gallons of water in basin B. The design for the system is shown in figure 4 below.

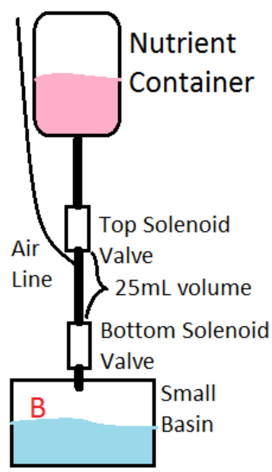


Fig. 4 Diagram of nutrient dispersal design

The nutrient container is affixed to the back of the greenhouse, well above basin B. Both solenoid valves are normally closed. When it is time for nutrients to be added to basin B, the top valve opens, allowing nutrients to fill the 25ml volume of tubing between the two valves. Ample time is given for this happen because as the nutrient container becomes emptier, the rate of flow is reduced. After the 25ml volume of tubing between the two valves is filled, the top valve closes. A small amount of time is waited for the top valve to close, upon the bottom valve opening and dispensing the 25ml nutrients into basin B. Once all the nutrients have dripped into the basin, the bottom valve closes and the system is ready to go for next time. The length of tubing between the valves was chosen such that the volume inside was exactly 25ml. Since $1ml=1cm^3$ and the inner diameter of the tubing was 6mm:

$$V = \pi r^2 h \qquad h = 25/(\pi * 0.6^2) = 22.1cm$$

So the tubing in between the two valves was cut to a length of 22.1cm. A graduated cylinder was used to measure the volume that the nutrient dispersal system dispenses. Upon testing, the measured volume was exactly 25ml. Another thing to note is the presence of the air line; if it were not there, a vacuum would be created between the two valves and nothing would come out when the bottom valve is opened. By inserting an airline, it eliminates the vacuum and allows for the nutrients to dispense when the bottom valve is opened. The valves can be opened and closed using the I/O on the raspberry pi and a simple circuit. Since the I/O pins only output 5V at 3.186mA, they are insufficient to drive the valves. A circuit using a TIP120 Darlington Power Transistor was used to attain the necessary current gain to drive the valves [10]. The current and voltage achieved across the valve during PSpice simulation was 294.7mA and 11.52V. In practice, the values measured slightly higher and were sufficient to open and close the valve. The circuit is shown in Appendix 3.

It is crucial to control the pH of the nutrient-rich solution as to maximize the plants' ability to absorb nutrients. The purpose of the pH regulation system is to keep the pH of the solution between 5.5 and 6.5 so the plants are able to absorb nutrients with optimum efficiency (to play it safe, we extended these bounds to 5 and 7 because over adjusting pH is worse than under adjusting it). If the pH falls too far outside of the range the plants can die [11]. The pH regulation system makes use of the same concept and as the nutrient dispersal system, using solenoid valves, except dispenses a product called "pH up" (a base used to raise pH) and "pH down" (an acid used to lower pH) instead of nutrients. There is one container fastened onto the back of the greenhouse holding "pH up" and one container holding "pH down". If the pH ever falls outside of that range, the system will dispense "pH up" or "pH down" in small quantities to adjust it. The volume between the two solenoids is $10ml = 10cm^3$ so the following calculation was used:

$$V = \pi r^2 h \qquad h = 10/(\pi * 0.6^2) = 8.84 \text{ cm}$$

And the tubing was cut to 8.84 cm.

Our team has measured water consumption using a log file that gets written to every time the 2 gallon basin gets refilled from the large basin. Over the growth cycle of our plants (61 days at this point in time), the 2 gallon bucket was refilled 16 times, which equates to 16 gallons per month. Since our large basin is capable of holding 20 gallons of water, so it only needs to be refilled roughly every 38 days. This is the only action the user will ever have to perform during the life cycle of the plants.

D. Android Application and Control Unit

The smart greenhouse is connected to an android application through a Raspberry Pi hosting a web server. Currently a Raspberry Pi b+ is being used, with Apache hosting the web server. The app has a few purposes. The first one is to provide live and past sensor information to the user so that the plants can be monitored from anywhere. This is done through a button showing the live readings of the plants and through graphs plotting the values for the past week or so. This means that a user will be able to see if something has gone wrong with the system. The final two purposes of the app are to start new plants, and also to provide useful information about handling and harvesting the plants. The informational part of the application is important because the seeds must be put in the machine correctly or it is less likely that they will grow to healthy fruiting plants. Also, the application must notify the user when the large basin is out of water (about every 3-4 weeks), and when the nutrients need to be switched to the bloom cycle (once per plant cycle).

Currently the android application is an HTML webpage, that if connected to the proper internet connection could be available anywhere. Some PHP scripts are used to retrieve sensor data from text files for the live readings, and the graphs are drawn with the D3.js Javascript library. When the user clicks live readings, the android application requests the latest sensor data from the Raspberry Pi. Also on the sensor readings page are buttons to graphs of past readings for the week for temperature, humidity, and hygrometer readings. The android application also features a plant page with a list of plants and a button to start the cycle for a new set of plants. Upon clicking a new plant the app warns the user that this will erase previous plant data, and upon confirmation starts a new cycle for that plant. The app also contains a main welcome page with a title and a drop down menu to navigate to all of the other pages. Screen shots from the App are shown below as Figure 5.

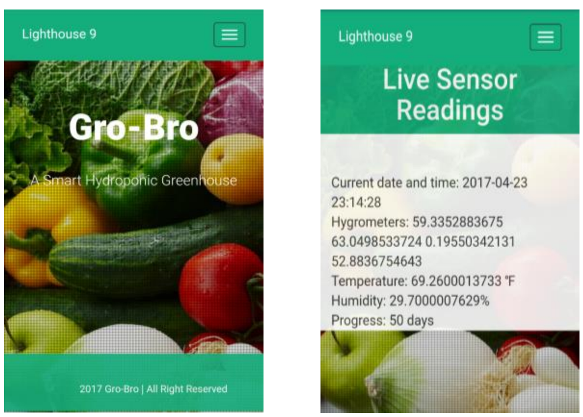


Fig. 5 App Screenshots

The Raspberry Pi is the central control unit for the entire system. Each group of similar type plants has a python script specifically designed for that plant that times the lighting and nutrition cycles to match the optimal cycle for that plant. When the user clicks the button to start a new plant that program is run on the Raspberry Pi. The structure of the program is based on our finite state machine and uses many small scripts to execute tasks at the correct times. The program will use system time to tell when to turn on the lights and pump the plants with water. Also the control unit will monitor the pH of the nutrient rich solution and the levels of the small water basin to make sure it is filled when it is empty, and in an acceptable pH range. Additionally the program must be able to stably run for a long period of time. This means that it must be able to reboot and regain its state in the case of a power outage. To solve this we save the exact state in the plant cycle often, so that if the control unit is rebooted, it will check to see where it left off and continue from there. For our own debugging purposes we added a log file that recorded all the actions of the machine so that if anything did go wrong we could find the exact time and place and correct it.

E. Lighting and Voltage Regulation

Using the proper lights for growing plants was critical to the success of the project. The main metric for choosing lights is the amount of usable energy that the lights produce. Most standard light bulbs, especially incandescent bulbs, waste most of their energy as heat. This is a problem because they can burn the plants as they grow closer to the lights. For this reason, the lighting options were narrowed down to LED or fluorescent lights as they produce the least amount of heat. Next, the efficiency of light absorption during photosynthesis was explored. The absorption of light by the chloroplast in plants depends on the absorption of light by various pigments in the leaves of the plants. The most important being chlorophyll-a and chlorophyll-b [4]. As shown below in Figure 6, these pigments are substantially better at absorbing light energy at red and blue wavelengths. LED grow lights can be purchased as arrays of alternating red and blue LEDs. This maximizes the amount of useable energy produced by the lights which lowers the overall required energy output of the lights. LED lights also have a much longer lifespan than fluorescent lights making them the obvious choice. Two 132W LED light arrays were purchased and installed on the top of the greenhouse structure. Each of these lights has the equivalent usable power output of a 300W high pressure sodium bulb, the most common light bulb for greenhouses, and runs 70% cooler [5]. The combined energy output for these lights allowed for the growing of full size plants, rather than dwarf plants that are common in household greenhouses. Lastly, a UV blocking film was added to the plexiglass to both protect from potential UV light produced by the blue LEDs and reduce the intensity of the ambient light leaving the greenhouse, giving the user a much more pleasant experience.

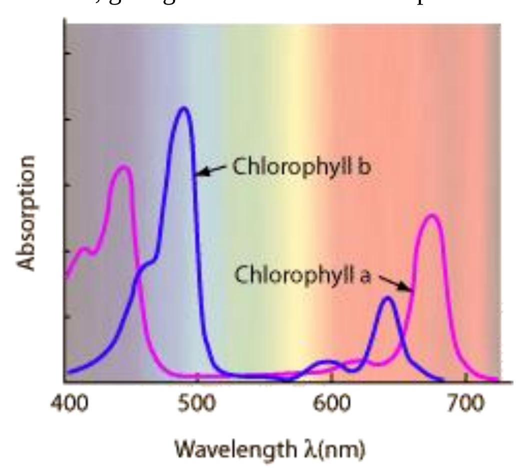


Fig. 6 Light spectrum of chlorophyll a and b [6]

The lights and hydroponic pumps need to be turned on and off in specific cycles in order to mimic the sunlight and water cycles of plants grown outdoors. This was accomplished by writing python code to trigger I/O pins on the Raspberry Pi. These I/O pins are connected to the bases of a MOSFETs which drive electromechanical relays. These relays act as high voltage switches allowing for toggling of power to a standard 120V 60Hz AC outlet. As shown in below in Figure 7, electromechanical relays work by running DC current through an inductor wrapped around a piece of metal. This current creates a magnetic field which pulls the armature toward the coil which causes the opposite end of the armature to move a mechanical switch allowing 120V AC to flow. The design for the greenhouse consists of an array of 8 electromechanical relays connected to 8 individual standard U.S. outlets. The plugs for the lights and hydroponic pumps are plugged into the outlets allowing for them to be turned on and off using python commands on the Raspberry Pi. The cycles for the lights and pumps are controlled using timing threads which are adjusted during the lifespan of the plants. The timing is also specific to the needs of each type of plant and are set according to the types of plants chosen in the app.

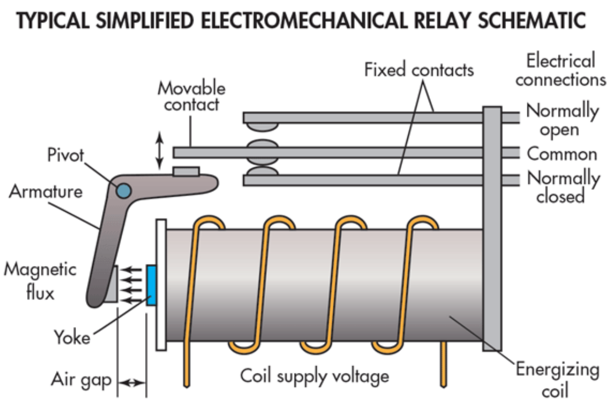


Fig. 7 Diagram of electromechanical relay [7]

Voltage regulation is required for this project since the solenoid valves and PH sensor require different voltage levels than the standard 5V and 3.3V DC. A voltage regulator circuit was designed to convert 5V DC to 12V DC using an LT1935 IC. The final circuit diagram is shown below as Figure 8. The custom PCB for the voltage regulator was designed using EAGLE by Autodesk and printed by OSH Park.

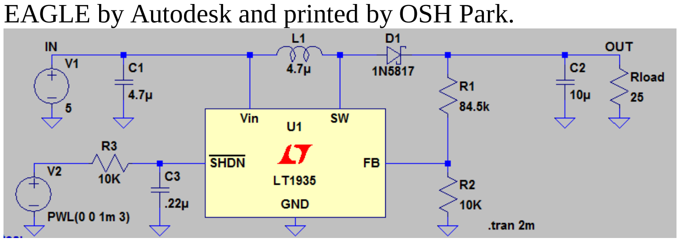


Fig. 8 Voltage Regulator Schematic

III. Project Management

Table 2
Final Deliverables

Deliverable	Status
Assembled hydroponics	Completed
Functional nutrient dispersal system	Completed
All sensors interfaced with pi	Completed
Lighting and pumps interfaced with pi	Completed
Communication with pi over internet	Completed
Android app with menus	Completed
Assembled greenhouse structure	Completed
UV film and Plexiglas applied to structure	Completed
Successfully Grew 6 Plants	Completed

The table above details our group's Final deliverables. Following the recommendation of our faculty evaluators after MDR, we made a more aggressive list and completed more than the deliverables we initially planned to. We developed and code a state machine and scaled up the cycle to a full plant growth from seed to harvest. Samantha interfaced the last of her sensors, determined how sensor data will be used in the FSM designed how it will be passed off to Max who represented it graphically in the android application. Max added notifications, and completed the android application, including polishing up the GUI. Mike and Shaun worked together to design and order a custom PCB used for voltage regulation. Shaun fabricated the pH regulation system, which is essentially a duplicate of the nutrient dispersal system. Lastly, the aesthetics of the greenhouse were addressed.

Our team worked exceptionally well together. The way our project was broken up ensured that everyone had something to do and by following our Gantt chart (see Appendix 4) closely, we have been able to meet or exceed all of our deadlines. We also met once per week with our advisor, Professor Jackson, and once per week as a team. This helped us all stay on the same page and ensured that we completed some amount of measurable progress each week, which Professor Jackson strongly encouraged. Due to the fact that we are good friends and that we were all thoroughly invested in our project made it a fun time to work on and created the desire to get things done and create a high quality product.

Each member of the group brought something different to the table. Sam, the group leader, has exceptional organizational skills and handled all interaction with professors and scheduling. Her skillset also includes hardware and circuit design. Mike is an expert in python and took the lead on the control unit code. Shaun has experience in embedded systems and PCB design and worked closely with Mike on the control unit and PCB. In addition, Shaun has skills in woodworking. Max is the software expert, specializing in android application development.

IV. Conclusion

The project has successfully been completed and the plants that we grew all thrived. Our team successfully grew radishes, spinach, basil and garden beans, all of which we harvested and ate. At the 54 day mark, we are still waiting for our tomatoes and hot peppers, which both take longer to grow than the other vegetables. We have several green tomatoes at around 2" diameter, which will be ready to harvest in about a week. All of our plants grew much faster than the traditional method of growing them in potted soil, giving them access to sunlight and watering them by hand; this is a testament to hydroponics, which grow plants more efficiently. This can be seen below in Figures 9 and 10. We performed two tests to compare our hydroponic system to a soil-planted control plant which was left in the sun and watered by Samantha. The first was a comparison of radish width and the second was a comparison of tomato plant height. As you can see, our system outperformed traditional gardening.

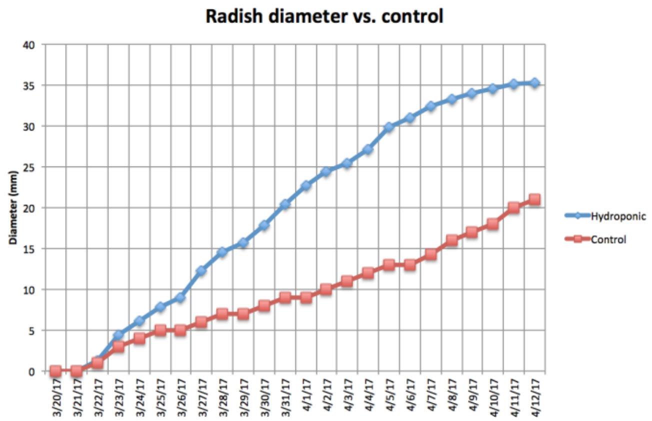


Fig. 9 Comparison of radish width from our hydroponic system and a traditionally potted plant

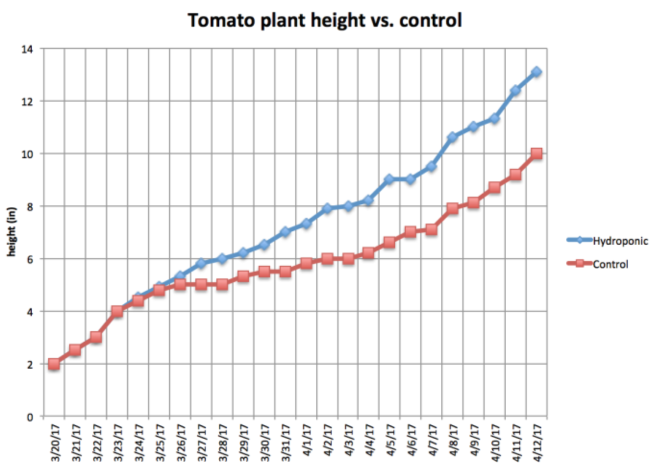


Fig. 10 Comparison of tomato height from our hydroponic system and a traditionally potted plant

In order to quantize the cost of buying our system and growing plants over and over again compared to buying produce from a store over and over again, we performed a cost analysis. The upfront cost to make the greenhouse was about \$925. We measured the energy consumption to be 4.2 kWh per day using a wall meter. At a cost of \$0.12/kWh of electricity and an estimated 60 days to grow tomatoes from seed to harvest, it would cost \$30.24 to grow 6 tomato plants, which each yield ~12-15lb of tomatoes: ~80lb total. Nutrients cost about \$15 per 60 days, so the total cost is about \$45 for 80lb of tomatoes, which comes out to \$0.56/lb. Store bought tomatoes average about \$2.00/lb. The total cost vs. pounds of tomatoes harvested/purchased is plotted in Figure 11.

We can see from the graph that the user will break even after harvesting 642lb of tomatoes, which equates to 8 (60 day) seed to harvest growth cycles. Assuming the user spaces cycles 1 month apart and is constantly growing, it will take 2 years to break even.

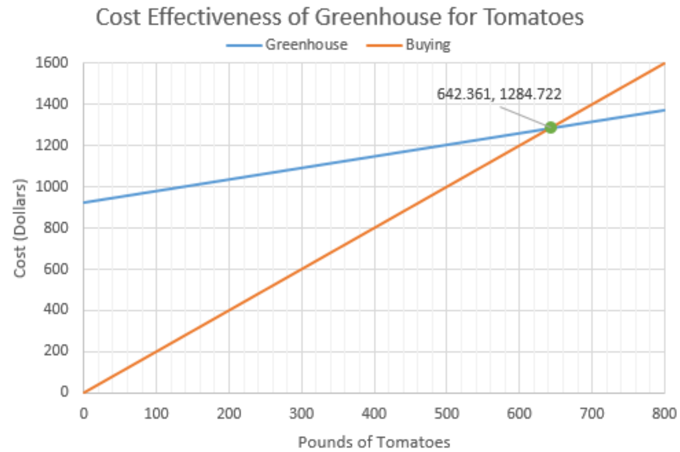
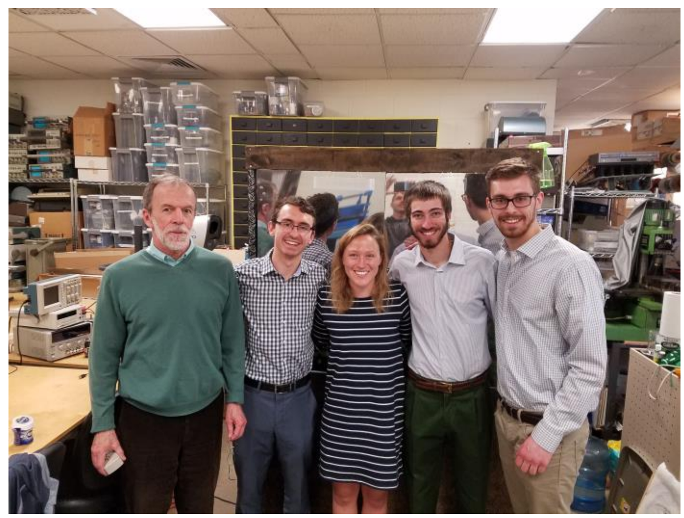


Fig. 11 Cost analysis of growing vs. buying tomatoes

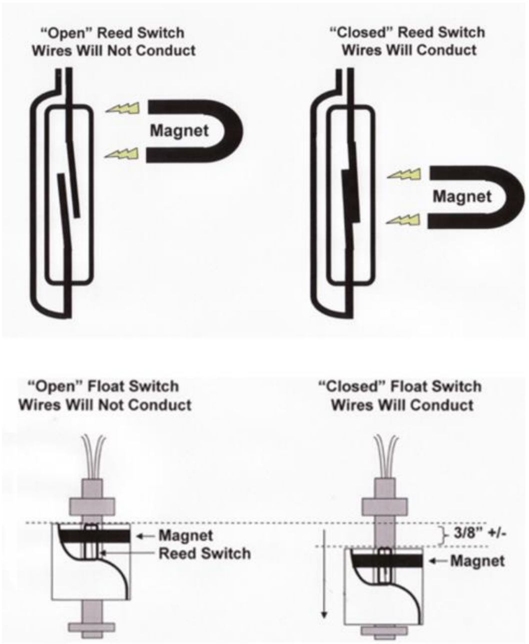
Overall, this project has been extremely successful, rewarding, and fun. We even received a compliment from our faculty evaluators stating: “the students spent a lot of time perfecting this project”.



Acknowledgment

Our group would like to thank our advisor, Professor Jackson, for giving us such great insight and advice throughout our project. We would also like to thank Fran Caron, our faculty evaluators, Professor Tessier and Professor Polizzi, and Linear Technologies for providing us with a free voltage regulator test board and sample parts.

APPENDIX



Appendix 1: Float switch diagram [2]

DATA=16 bits RH data+16 bits Temperature data+8 bits check-sum

Example: MCU has received 40 bits data from AM2302 as

0000 0010 1000 1100 0000 0001 0101 1111 1110 1110

16 bits RH data 16 bits T data check sum

Here we convert 16 bits RH data from binary system to decimal system,

0000 0010 1000 1100 → 652

Binary system Decimal system

RH=652/10=65.2%RH

Here we convert 16 bits T data from binary system to decimal system,

0000 0001 0101 1111 → 351

Binary system Decimal system

T=351/10=35.1℃

When highest bit of temperature is 1, it means the temperature is below 0 degree Celsius.

Example: 1000 0000 0110 0101, T= minus 10.1℃

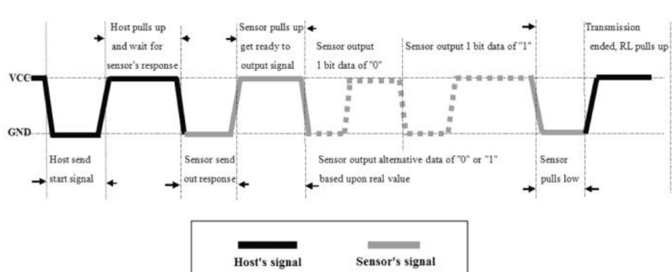
16 bits T data

Sum=0000 0010+1000 1100+0000 0001+0101 1111=1110 1110

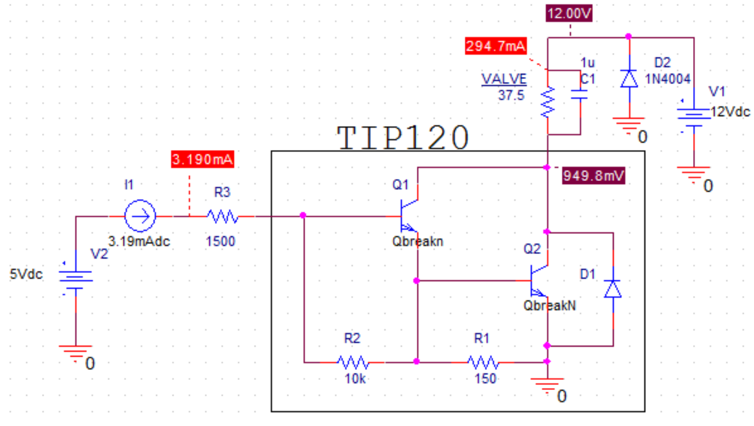
Check-sum=the last 8 bits of Sum=1110 1110

When MCU send start signal, AM2302 change from standby-status to running-status. When MCU finishs sending the start signal, AM2302 will send response signal of 40-bit data that reflect the relative humidity and temperature to MCU. Without start signal from MCU, AM2302 will not give response signal to MCU. One start signal for one response data from AM2302 that reflect the relative humidity and temperature. AM2302 will change to standby status when data collecting finished if it don't receive start signal from MCU again.

See below figure for overall communication process, **the interval of whole process must beyond 2 seconds.**



Appendix 2: AM2302 Data Sheet



Appendix 3: PSpice schematic of circuit used to drive solenoid valves

Appendix 4: Team Gantt Chart (see final page)

REFERENCES

[1] "How Far Does Your Food Travel to Get to Your Plate?" *CUESA*. Center for Urban Education about Sustainable Agriculture, n.d. Web. 22 Dec. 2016.

[2] "Liquid level sensors from chicago sensor, Inc," in *Chicago Sensor*.

[3] C. W. 2009, "How do pH meters work? | measuring acidity," in *EXPLAINTHATSTUFF!*, Explain that Stuff, 2016. [Online].

[4] Moore, Randy, and W. Dennis Clark. *Botany*. Dubuque: Wm. C. Brown, 1995. Print.

[5] "SYGAVLED 300W LED Grow Light." *Amazon*. SYGAVLED, n.d. Web. 22 Dec. 2016.

[6] Nave, R. *Light Absorption for Photosynthesis*. Digital image. HyperPhysics Biology. Georgia State University, n.d. Web. 22 Dec. 2016.

[7] Teschler, Leland. *Typical Simplified Electromechanical Relay Schematic*. Digital image. *Machine Design*. N.p., 31 July 2014. Web. 22 Dec. 2016.

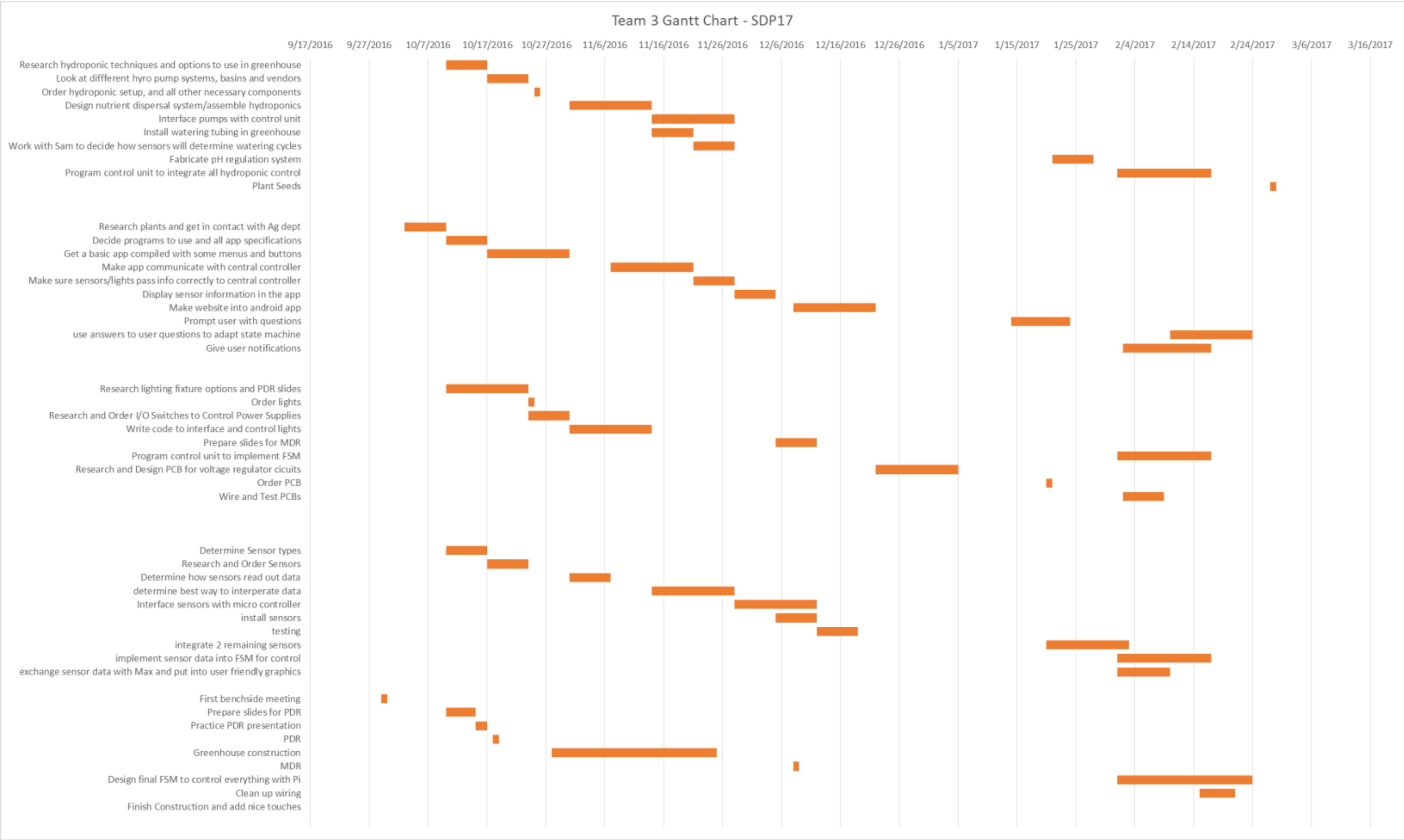
[8] "Hydroponic Growing Media." *Epic Gardening*. N.p., n.d. Web. 22 Dec. 2016. <http://www.epicgardening.com/hydroponic-growing-media/>.

[9] "A Crash Course in Fertilizers." *Sunset*. Time Inc. Lifestyle Group, n.d. Web. 22 Dec. 2016. <http://www.sunset.com/garden/garden-basics/crash-course-fertilizers>.

[10] Techbitar. "Use Arduino with TIP120 Transistor to Control Motors and High Power Devices." *Instructables.com*. N.p., 25 Oct. 2011. Web. 22 Dec. 2016. <http://www.instructables.com/id/Use-Arduino-with-TIP120-transistor-to-control-moto/>.

[11] "PH." *Simply Hydroponic*. N.p., n.d. Web. 22 Dec. 2016. <http://www.simplyhydro.com/ph.htm>.

[12] Boye, Joyce I., and Yves Arcand. *Green Technologies in Food Production and Processing*. New York: Springer, 2012. *Cleanmetrics.com*. Web.
[13] "Local Harvest Calendar - FarmFresh.org -." *Local Harvest Calendar*. Farm Fresh Rhode Island, n.d. Web. 06 Feb. 2017.


Team 3 Gantt Chart - SDP17
9/17/2016
9/27/2016
10/7/2016
10/17/2016
10/27/2016
11/6/2016
11/16/2016
11/26/2016
12/6/2016
12/16/2016
12/26/2016
1/5/2017
1/15/2017
1/25/2017
2/4/2017
2/14/2017
2/24/2017
3/6/2017
3/16/2017
Research hydroponic techniques and options to use in greenhouse
Look at diffferent hyro pump systems, basins and vendors
Order hydroponic setup, and all other necessary components
Design nutrient dispersal system/assemble hydroponics
Interface pumps with control unit
Install watering tubing in greenhouse
Work with Sam to decide how sensors will determine watering cycles
Fabricate pH regulation system
Program control unit to integrate all hydroponic control
Plant Seeds
Research plants and get in contact with Ag dept
Decide programs to use and all app specifications
Get a basic app compiled with some menus and buttons
Make app communicate with central controller
Make sure sensors/lights pass info correctly to central controller
Display sensor information in the app
Make website into android app
Prompt user with questions
use answers to user questions to adapt state machine
Give user notifications
Research lighting fixture options and PDR slides
Order lights
Research and Order I/O Switches to Control Power Supplies
Write code to interface and control lights
Prepare slides for MDR
Program control unit to implement FSM
Research and Design PCB for voltage regulator cicuits
Order PCB
Wire and Test PCBs
Determine Sensor types
Research and Order Sensors
Determine how sensors read out data
determine best way to interperate data
Interface sensors with micro controller
install sensors
testing
integrate 2 remaining sensors
implement sensor data into FSM for control
exchange sensor data with Max and put into user friendly graphics
First benchside meeting
Prepare slides for PDR
Practice PDR presentation
PDR
Greenhouse construction
MDR
Design final FSM to control everything with Pi
Clean up wiring
Finish Construction and add nice touches