Automated Greenhouse Watering and Heating System for the Schenectady ARC

Yusen Meng
*Union College - Schenectady, NY*

Aikaterini Petridou
*Union College - Schenectady, NY*

Follow this and additional works at: [https://digitalworks.union.edu/theses](https://digitalworks.union.edu/theses)

Part of the Controls and Control Theory Commons, Digital Communications and Networking Commons, and the Electrical and Electronics Commons

**Recommended Citation**
[https://digitalworks.union.edu/theses/2452](https://digitalworks.union.edu/theses/2452)

This Open Access is brought to you for free and open access by the Student Work at Union | Digital Works. It has been accepted for inclusion in Honors Theses by an authorized administrator of Union | Digital Works. For more information, please contact digitalworks@union.edu.
Automated Greenhouse Watering and Heating System for the
Schenectady ARC

Submitted in partial fulfillment of the requirements for Honors in the Department of Electrical, Computer and Biomedical Engineering

Authors: Aikaterini Petridou and Yusen Meng
Thesis Advisor: James Hedrick
Department of Electrical, Computer and Biomedical Engineering

UNION COLLEGE
June 2021
Abstract

Everyone wants to feel useful and to be able to contribute to their community. The Schenectady ARC aims to provide people with developmental disabilities the resources, services, and support that enable them to advocate and participate within their communities. [1] The program seeks to encourage these people to develop skills and hobbies that give them independence and purpose. One way is operating the ARC’s greenhouse. Individuals at the Maple Ridge Center are responsible for operating the water system to irrigate the plot daily and ensuring that the proper amount of water is distributed to the plants. This is a rewarding activity and offers them the opportunity to develop useful skills in greenhouse management and maintenance. However, despite offering manual and automatic options, the current equipment of the greenhouse is not user-friendly and vulnerable if water leaks. The objective of our project is to communicate with the Schenectady ARC to develop an automatic controller for heating and water delivery that will be easy to use, safe, robust, affordable and easy to maintain. This project was started by Guo Qianyue CPE ‘16, kept on by Stengel Kyle CPE ‘18, continued by Lisa Gu CPE ‘19, and followed by Larissa Umulinga CPE ‘20. [2] [3] [4] [5] The system we created is a wireless sensor system that reads and transmits the moisture and temperature data wirelessly to control the watering and heating system at the ARC greenhouse. This paper describes the problem, goals, design specifications, testing plan, standards and ethics for this project.
Table of Contents

Abstract 2

Table of Contents 3

Problem Definition 4
  Topic Overview 4
  Problems and Project Goals 5
  Literature Review 6

Design Specifications 7
  Interface and accessibility 7
  Automatic and Manual Control 18
  Sensing Moisture 18
  Water Delivery 18
  Sensing Temperature 19
  Performance 20
  Safety 20
  Economic 21
  Data Collection 21

Alternative Design 22
  Wired Data Transmission 22
  Moisture Sensing 23
  Temperature Sensing 24
  User Interface 24

Testing 25

Main Controller 26

Ethical Considerations 27

Future Work 27

Standards 27

Cost Analysis 28

Acknowledgments 29

References 30
Problem Definition

I. Topic Overview

Located in Albany, New York, the Schenectady ARC is a non-profit organization that is dedicated to support individuals with developmental disabilities. To promote individualism amongst the people at the center, the ARC allows them to participate in activities such as the maintenance of their greenhouse. Currently, the installed water delivery system requires manual work and knowledge of electronics to troubleshoot and maintain. Therefore, the purpose of the project is to implement a water delivery and heating system tailored to the ARC’s greenhouse in terms of use, cost, and intuitive operation for people with disabilities. For our capstone project, we chose to continue Larissa Umulinga’s project on the Greenhouse Watering System for the Schenectady ARC. Larissa, a Union College Class of 2020 CPE student, improved the system by automating the moisture level using an Arduino MEGA 2560. The following figure shows the top level design of the overall system.

![Figure 1. Top Level Design of System](image-url)
The project consists of an electric socket, a 120V AC power supply, one solid state relay which acts like a switch for each heating pad and another one that controls each solenoid for the water delivery system, a wireless sensor module which provides moisture and temperature readings, a main controller which controls the touch screen to display the data using a Raspberry Pi. The system we focus on is the wireless sensor module and the main controller.

![Solid State Relay](image)

**Figure 2.** Solid State Relay

### II. Problems and Project Goals

Larissa’s prototype, with the new board for the main controller, was lost during the COVID-19 pandemic, and that is a contretemps because we were planning to test it with the greenhouse watering system for compatibility and measurement purposes. The previous subsystems included the keypad, the slide switch, and the LCD which we replaced with a touchscreen. In addition to that, it needs a wireless sensor system because at the moment, the system is quite vulnerable with the wires and connection being exposed near the plants and water. Furthermore, there is a need for more prototypes to offer diverse functionality in different greenhouses. The ultimate goal for the greenhouse is to be easily operated by people with disabilities without supervision.
III. Literature Review

The Schenectady ARC, as a non-profit organization, provides resources to help individuals with disabilities [1]. Care and support is provided to over 1000 people with a wide range of ages and developmental disabilities. The center is equipped with a greenhouse which serves as a platform for the individuals at the center to acquire greenhouse maintenance skills, but is also used as a means to collect funds by selling plants. The watering system of the greenhouse is provided by the Maple Ridge Center to provide enough water to sustain the moisture of the soil for the plants [6]. In 2015, Qianyue Guo built a prototype of an automated watering system which consisted of a solenoid connected to a drip tube and moisture sensors [2]. Reading levels and the calibrations were the constraints to the prototype. Similarly, Kyle Stengel designed a Linux-run controller with a touchscreen GUI in relation to the heating of the seedlings [3]. Lisa Gu combined both works into a single, simple, and user-friendly manual controller for the systems that were triggered respectively to the user input with a control of the solenoids and manual switches [4]. Then, Larissa Umulinga implemented the system so that it offers both manual and automated control [5].

Figure 3. The Schenectady ARC Watering System Setup
Design Specifications

From the already installed watering system, the new implemented watering system for the greenhouse should satisfy design requirements given by the Director of Schenectady ARC Donna Vincent.

I. Interface and accessibility

The watering system is implemented to be used by people with intellectual and developmental disabilities; therefore, it needs to be easily operated and quick to learn. The system does not require any special knowledge to be operated or maintained. Thus, no user manual will be necessary. Simple operations such as touching a screen or pushing a button will allow the user to control the system. In addition, the user does not need to understand the inner workings, such as the circuitry of the system, to run it. The previous system did not contain a touchscreen and we replaced it with a Raspberry Pi so that we integrate a touchscreen to create a more user-friendly interface. The following block diagrams will demonstrate how each component is connected to the system and its purpose.

Figure 4. Wireless Sensor Block-diagram
There are two remote units; the wireless sensor module on the left and the main controller on the right. The wireless sensor module will be placed at the seedling beds and it consists of a Raspberry Pi 3 B+, an XBee, a moisture and a temperature sensor. The main controller includes an Xbee radio module, a Raspberry Pi 3B+ and a touch screen. The wireless sensor module will read data from the sensors and transmit it wirelessly to the main controller. Then it computes the readings from the two sensors and displays the temperature and moisture accordingly.

Figure 5. Remote Sensor Unit

The Remote Sensor Unit consists of a Raspberry Pi, an XBee module, a moisture sensor and a temperature sensor. Each seedling bed will get one module for collecting temperature and moisture data. The XBee will read the data from the temperature sensor and moisture sensor through two separate ADC ports. The Raspberry Pi will process the data and the XBee will broadcast the data every hour. After further testing with the moisture sensor, we got consistent results. Thus, we decided to use one moisture sensor to reduce the overall cost.
The XBee network consists of one coordinator and multiple end devices. We established the network by using the XBee python library. All the XBees are on the same channel and every end device has two-way communication enabled with the coordinator. Since we are connecting the Xbees to a Raspberry Pi, which is a computer running on a linux system, we used python instead of micro-python. We chose the XBee Python Library because it can support multiple XBee devices and protocols and for compatibility reasons. To install the library, first ensure you have python 3 installed and then use the following line in the terminal: pip3 install digi-xbee.
The XBee will receive the data from the remote sensor unit, and the Raspberry Pi will recognize the data type (temperature or moisture) and calculate the moisture based on the reference voltage that is 2.5V. Finally, the Raspberry Pi will display the data on the touch screen.

Figure 8. Wireless Sensor Module Settings
XCTU is the software we used to configure the XBees. To set up the parameters for the End Device:

1. Configure channel for all XBees to ‘C’ and PAN ID to 2244.
2. Set the device role to ‘End Device’.
3. Set up source address and destination address for two-way communication.
4. Set up reference voltage for reading sensors.
5. Configure the XBee to API Mode Without Escapes.
6. Set Baud Rate to 9600, Disable Parity check, One UART stop bit.
7. Configure DIO1 and DIO 3 to ADC pins.

The reference voltage can be set to 1.25V, 2.5V, or external. The output voltage of both sensors range from 0 to 2V, so the best fit is if we use a 2.5V reference voltage to cover the entire range of the sensor reading. We divide 2.5V with max reading of a 10 bit ADC, which is 1024. This gives us an accuracy of 0.00244V.
Figure 9. Coordinator Settings
For the coordinator you need to:

1. Configure channel for all XBees to ‘C’ and PAN ID to 2244.
2. Set the device role to Coordinator.
3. Set up source address and destination address for two-way communication.
4. Configure the XBee to API Mode Without Escapes.
5. Set the Baud Rate to 9600, Disable Parity check, One UART stop bit.

The end device and the coordinator are configured to have a two-way communication. First, we set the channel, and PAN ID of both XBees so that they can be on the same network. We configure the destination address of the end device to be the same as the 16-bit source address of the coordinator; this will allow the end device to send data to the coordinator. We also configured the coordinator’s destination address to be the same as the 16-bit source address of the end device. This will allow the coordinator to send data to the end device, thus establishing a two-way communication.
Figure 10. System Pin Diagram

On Figure 10 a complete pin diagram of the wireless sensor system is shown. The Raspberry Pi on top is connected to the coordinator XBee Radio module through a USB connection and a touch screen that is shown below in the picture. The other Raspberry Pi is connected to the End Device XBee Radio Module through a USB connection. This Xbee is connected to the capacitive moisture sensor via the DIO1 and to the temperature sensor via DIO3 pin. The moisture and temperature sensors are connected to ground and VCC respectively.
The temperature sensor’s data pin is the yellow wire, which is connected to the DIO3 pin on the XBee.

Similarly, a capacitive moisture sensor’s data pin is the green wire, which is connected to the DIO1 pin on the XBee. Both DIO1 and DIO3 are configured as ADC pins.
The XBee is connected to the Raspberry Pi through a USB port. The Raspberry Pi also has a touchscreen connected to it for displaying the data.

### Code for XBee communication

1. **Code for the sender Raspberry Pi:**

```python
#!/usr/bin/env python                      # import all libraries
from digi.xbee.devices import XBeeDevice
from digi.xbee.io import * #IOSample, IOLine
import time

device = XBeeDevice("/dev/ttyUSB0",9600)   # Instantiate an XBee
device.open()                                # device object
```

**Figure 12. Main Controller**
io_sample = device.read_io_sample()
print("IO sample was read successfully")
while 1:
    # data from moisture
    io_line = IOLine.DIO1_AD1
    moistureValue = device.get_adc_value(io_line)
    print("The moisture is: ", moistureValue)
    device.send_data_broadcast(str(moistureValue))
    # data from temperature
    io_line = IOLine.DIO3_AD3
    temperatureValue = device.get_adc_value(io_line)
    print("The temperature is: ", temperatureValue)
    device.send_data_broadcast(str(temperatureValue))
    time.sleep(0.5)

2. Code for receiver Raspberry Pi:

    from digi.xbee.devices import XBeeDevice
    import time
    device = XBeeDevice('/dev/ttyUSB0', 9600)      # Instantiate an XBee
    device.open()                                  # device object
    while 1:
        xbee_message = device.read_data()
        if xbee_message != None:
            message = xbee_message.data.decode()        # decode the message sent
            value = int(message)
            if value < 500:
                voltage = value*0.00244141
                temperature = (voltage-0.6)/0.008          # convert the value to °C
                print("The temperature is: ", temperature, "°C")
            else:
                print("The moisture is: ", value)
II. Automatic and Manual Control

The system we created is a wireless sensor system that can be easily implemented with the infrastructure existing at the greenhouse. This system allows automatic and manual control for the temperature and the humidity. Previous students worked on the water delivery and heating systems at the greenhouse.

III.

IV. Sensing Moisture

Water is regulated according to the measured moisture of the seeding beds. Air conditions and soil moisture should be measured every hour (as it is the average time the moisture level changes), and the collected data should be sent to the system to control water in real-time. Given that the moisture sensors are inserted in the soil, they should not harm the growth of the seedlings'. Seedlings are affected by both air and moisture conditions at early stages of growth; hence the systems would control the water accordingly. The moisture sensor is connected to the end device through a data pin. Then the XBee will broadcast the data wirelessly to the preset channel. The communication is completely wireless, allowing us to build a waterproof system.

V. Water Delivery

From previous research done by the past students, the optimal way of water delivery is by utilizing overhead misters because the water distributes evenly throughout the seedling beds. This method is cost effective, precise and efficient in the conservation of water.

The following block diagram shows the implementation of the water delivery system. The Raspberry Pi will send a digital signal to the solid state relay and this will trigger the solenoid to turn the water misters on or off accordingly. This depends on the threshold value we set for the plants. If the output from the Figure 13. Water Mister
moisture sensors are higher than the desired moisture level it will turn them off, otherwise it will turn them on.

**Figure 14. Water Delivery Block Diagram**

VI. Sensing Temperature

Temperature will be delivered through heating pads that are placed under the seedling bed. Moisture and temperature data will be measured every hour. The temperature sensors will send the data to the system wirelessly, and control the temperature in real-time. An upper limit for temperature pads will also be set in case any malfunction occurs from the temperature sensor.

VII. Heat Control

The waterproof heating pads are placed directly under the seedling beds. It was decided to purchase heating pads whose length was the same or similar in size to the width of the table. Their purpose is to warm the rooting area to improve germination and rooting. The heating pads are controlled by the solid state relay, which is directly controlled by the Raspberry Pi. The heating pads will turn on based on the reading of the temperature sensor.

**Figure 15. Heating Pad**
VIII. Performance

Moisture levels will be measured every hour to maintain the moisture levels by spreading water onto 8’ by 11’ seedling beds. Given how large the area is, it was divided into four quadrants according to Donna Vincent’s request, with no quadrant acquiring more water than another. Each quadrant’s moisture readings indicate the level of moisture on the entire quadrant. The difference between the readings of the moisture should be insignificant. The moisture sensor is designed to operate in the range from -10 °C to 80 °C.

The heat will be delivered through a 120V AC heating pad. Each heating pad will have a maximum of one seedling bed on it. Each seedling bed will get one module for collecting temperature and moisture data.

IX. Safety

Safety is our top priority when designing this project. Since we are designing a very practical system, we need to make sure that we cover all the safety concerns and follow the IEEE safety standards. The main controller that assembles all the components involves a lot of wiring. Both the remote sensor unit and the main controller have wireless communication to avoid any of the wires conducting high voltages that could possibly endanger any person in its surroundings. From the Schenectady ARC, the hardware design should fit in its respective place without causing any harm to other activities carried out in the greenhouse. In addition, a waterproof case is made to avoid water from leaking to electronics accidentally. We used armored tubes for the wiring inside the greenhouse, so we can ensure the wires are waterproof.
and impact resistant. An upper limit temperature for the heating pads will also be set in case of any malfunctions.

![Image of Waterproof Case](image)

Figure 17. Waterproof Case

X. **Economic**

The Schenectady ARC is a non-profit organization, implying that funding for the project is not as highly available. In this case, all the necessary purchases will have to be done through the provided funds from New York State Industries for the Disabled (NYSID) through the CREATE project. CREATE provides assistance to engineers to allow them to put their skills into use and solve a real-life issue through their Capstone projects.

XI. **Data Collection**

It is not yet required from Ms. Vincent, but after talking to Professor Hedrick, we concluded it would be a good idea to keep track of the moisture and temperature sensor’s data and keep history files. In addition, we want them to be uploaded to the cloud. Given that we will use a Raspberry Pi, we can use an ethernet cable, send the data wirelessly and perhaps analyze them in the future.
Alternative Design

I. Wired Data Transmission

We established wireless communication between the remote sensor unit and the main controller. Although wired connections were also considered, we proceeded with wireless communication, which is clearly superior. If we had decided to apply a wired connection solution, all the wires would run through an armored tube and waterproof case. Any exposure of the system could potentially short the circuitry and damage the system. This is the problem that this project tackled, hence making wireless communication much more potent.

We decided to use a Xbee 3 wireless module because it has a wireless communication range of 60 meters indoors. The Xbee 3 is a RF 250Kbps device, we will be using transmitting data over the 2.4Ghz frequency band. The Xbee 3 only requires 2.1 V to 3.6 V for power supply, thus it is power efficient. The Xbee 3 also comes with multiple A/D converters, allowing us to connect multiple sensors on it. The Xbee 3 costs around $20, which will reduce the expenses for the Schenectady ARC.

Figure 18. Xbee - wireless communication
II. Moisture Sensing

We also investigated alternatives for the moisture sensor. We delved into capacitive and resistive moisture sensors. We mainly considered capacitive moisture sensors due to its high accuracy in measurement and robustness. We picked two potential replacements for the moisture sensor; these are the Dfrobot SEN0193 (Capacitive moisture sensor) and the Resistive Soil Moisture Sensor from IS (Resistive moisture sensor). The base values to indicate the desired moisture levels had been provided by Donna Vincent from testing with a tensiometer. The size, the cost, and the compatibility to multiple control systems from both sensors were given priority over the accuracy of the readings of moisture levels of the soil. The sensors were small to fit into the seedling beds, and the cost was within an affordable price for the Schenectady ARC. Due to the exposure of wires to outside conditions, the moisture data are sent wirelessly to the coordinator to promote independence of subunits in terms of troubleshooting and maintaining the system. In Figure 19 below, the capacitive moisture sensor is shown.

![Capacitive Moisture Sensor](image)

**Figure 19.** Analog Capacitive Soil Moisture Sensor
III. Temperature Sensing

We had a few alternatives for temperature sensors. The temperature sensor we chose for the project is TMP36, shown in the figure below. We used a 0.1μF bypass capacitor between VCC and ground to decouple the AC and DC signals. The alternative sensors included: Vegetronix Soil Temperature Sensor Probe and LM35 temperature sensor. Due to limited time, we were unable to put a new order and replace the current temperature sensor with a waterproof temperature sensor. However, the code is relatively the same for the existing conversion code from an analog value to degrees Celsius.

Figure 20. Temperature Sensor

IV. User Interface

To improve on the previous projects, we decided to update the user interface with a touch screen. We are aware Larissa tested the functionality of the slide switch to choose between automatic and manual control that is displayed on the LCD screen. The next step for us was to rebuild the system and use a touch screen which is thoroughly tested. The implementation of the touch screen is essential not only for displaying purposes but also for testing. The Schenectady ARC greenhouse is meant to be utilized by people with developmental disabilities. Since it is more intuitive to use a touchscreen than a 4 x 4 keypad and a switch with an 16 x 2 LCD screen, we concluded using a touch screen we can create a substantially more
user-friendly interface. The touch screen we use is a Raspberry Pi LCD, it is 7 inches, 800 x 480 touch screen. It comes with a DSI connection, which works with the Raspberry Pi B+ 3 controller. It supports 10-finger touch and a virtual ‘on-screen’ keyboard with the latest Raspbian OS (Raspberry Pi controller OS). Hence, the user can use the touch screen to access the system without connecting a keyboard or a mouse.

![Touch screen compatible with Raspberry Pi](image1.png)

**Figure 21.** Touch screen compatible with Raspberry Pi

**Testing**

Initially, we tested the wireless connection between two XBees. We used the software XCTU, in which apart from configuring the settings for the XBees, there is an option for communication between the two on a monitor inside XCTU. We achieved both one-way and two-way communication. Even if the final system would not run on the XCTU, it was necessary for testing if the source and destination addresses were correct for one-way or two-way communication that would later be used. After doing this we found another alternative using micropython to establish a connection but soon we discovered a python 3 library that could be used. After carefully reading the documentation, we realized this python library is ideal for our application, especially because we use Raspberry Pi which includes a complete python 3 development system. Another difficulty we encountered was which python library should we use; there were an
abundance of options online and before we started running scripts on Raspberry Pi, not all of the libraries were able to execute a script correctly on school computers using Windows PowerShell. Thankfully, once we tested these libraries on the Raspberry Pi, they were executed with no errors and with some debugging, we accomplished communication between the two XBees. This process of testing and debugging took several weeks but knowing what works and what doesn’t is helpful for further development.

Then we proceeded by testing each component/ sensor individually and then combining them together. For example, we tested the moisture and temperature sensors alone, the Xbees alone, and then all together to check if the signal is transmitted accurately and efficiently. After weeks of testing, we successfully established a connection between XBees using Raspberry Pi and read and transmitted the data from moisture and temperature sensors, in addition to decoding them to their corresponding units.

**Main Controller**

We use a Raspberry Pi and a touchscreen to implement the previous lost system and make it more user friendly to the user. Larissa replaced the Arduino with the Arduino Mega 2560. We replaced Arduino Mega 2560 with Raspberry Pi 3 B+.

![Figure 22. Replace Arduino with Raspberry Pi 3 B+](image)
Ethical Considerations

For the transmission of the moisture levels, via Xbee, we will be using radiofrequency waves and it is essential to follow the Federal Communications Commission (FCC) regulations for all the equipment that is installed so that it does not cause interference with other devices. The solenoids that open to provide water to the plants do not operate with 120V, but 24V AC so that it is lower voltage. The heating pads operate with 120V AC directly and thus the outlets that we will install have to be waterproof to avoid the chance of someone getting electrocuted. Lastly, the design of the equipment has to be made to be user-friendly and not able to hurt the user. To avoid that, we offer two modes: user mode and supervisor mode and if there are voltages that could be exposed they would have to be locked and only available to the supervisor and would require supervisor access. These are some examples of how we tackle issues that could be caused unintentionally.

Future Work

The system uses a Raspberry Pi with XBee for the wireless sensor module; however, the XBee can be configured with the XCTU software to send data to a specific network address periodically using sampling rate. This would reduce the power consumption of the remote sensor unit. This is a wireless module and having a long battery life is essential since not only it reduces the cost and effort to maintain the system, but it also increases the ease of use significantly. A graphical user interface can also be implemented to improve the user experience.

Standards

Since we use the Xbee module for transmitting and receiving the signal for getting moisture level information, we need to ensure it follows the below IEEE standard.
I. IEEE C63.26-2015 - IEEE/ANSI Standard for Compliance Testing of Transmitters Used In Licensed Radio Services. This standard gives different procedures one must follow when using technology that includes transmitters.

Also, Xbees work wirelessly and need to follow the below standard.

II. IEEE C63.27-2017, American National Standard for Evaluation of Wireless Coexistence. ANSI C63. The standard provides a protocol based process and test methods to validate the ability of wireless devices to coexist with other wireless services that operate in the RF bands of a given wireless device.

The Greenhouse project offers both the manual and automatic control options. That is in case the automatic stops working someone can operate it manually. There is an IEEE standard it has to follow for the above.

III. This standard applies to, and provides the basis for the definition, specification, performance analysis, and application of systems used for supervisory control, data acquisition or automatic control, or both, in attended or unattended electric substations, including those associated with generating stations, and power utilization and conversion facilities.

Lastly, to ensure safety measures are taken for the heating pads that will be operating at 120 Volts, we will use Underwriters Laboratories (UL) equipment. UL is a global independent safety science company with more than a century of expertise innovating safety solutions.

Cost Analysis

The total cost from all the components used to implement the automated water delivery system as it is shown in the table below. The upcoming implementation involves the listed parts below with a total of $442.96 but note that we included ten moisture sensors instead of two that we used in the demo. Upon
completion of the project, the items will be given to the school for further implementation of prototype testing and at the ARC Greenhouse. Overall we achieved a low cost system well below the maximum budget.

<table>
<thead>
<tr>
<th>Parts</th>
<th>Links</th>
<th>Quantity</th>
<th>Unit Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi B+ 3</td>
<td>digiKey</td>
<td>2</td>
<td>35</td>
</tr>
<tr>
<td>Touchscreen</td>
<td>digiKey</td>
<td>1</td>
<td>79</td>
</tr>
<tr>
<td>Temperature Sensor</td>
<td>digiKey</td>
<td>1</td>
<td>1.62</td>
</tr>
<tr>
<td>Capacitor Moisture Sensor</td>
<td>amazon</td>
<td>10</td>
<td>8.31</td>
</tr>
<tr>
<td>XBee 3</td>
<td>digiKey</td>
<td>4</td>
<td>32.56</td>
</tr>
</tbody>
</table>

| Total: 363.96                |        |          |                |

Table 1. Cost Analysis

Acknowledgments

We want to thank Professor James Hedrick for the constant support and guidance he provided us throughout this project. We really appreciate his dedication and time he spent with us debugging the system. We also want to thank the Schenectady ARC Greenhouse to provide us this opportunity to work on this meaningful project. Last but not least, we want to thank the ECBE department for the funds and support for this project.
References


