A Low-Cost Device for Flood Disaster Preparation Using an Arduino Board

Bryce Rainey
Earlham College Department of Computer Science
801 National Rd W
Richmond, Indiana
barainey12@earlham.edu

ABSTRACT

This paper lists and explains the steps taken to create a low-cost and energy-efficient device that detects rising water and alerts the proper personnel via email of a flood situation. It describes how to assemble the Arduino, YL-69 soil humidity sensor, which is used as a water level sensor, and YL-38 module, as well as how to program it in C using the software provided by Arduino. The particular Arduino model used is the Yun because it contains built-in WiFi.

This paper also provides reasons and motivations for creating such a device, including the statistics, safety benefits, and practicality of the application. Most details of the technology incorporated in the project will be thoroughly explained for readers that are not familiar with the area of computer science.

1. INTRODUCTION

Advancements in technology are made day by day, and as new gadgets are being released, new ideas are generated in the minds of humans. Many people assume that these ideas are aimed at bettering the lives of people in technologically advanced areas such as the United States and parts of Europe. While these gadgets and ideas may be helpful and entertaining, most do not benefit, nor protect, the people living in parts of the world that are not as technologically advanced.

Instead of focusing on bettering the already-convenient lives and safety of people with new, complex contraptions, the main focus needs to shift to implementing simple technologies into areas that are prone to natural disasters, specifically flooding. In 2013, accounting for 44 percent of the total deaths from natural disasters, flooding was ranked number one in the world[1]. This statistic alone demonstrates the need for technology to be implemented in places that are highly susceptible to flooding.

Although the devastation from natural disasters seems difficult to prevent, the technology used is surprisingly modest. When I speak with non-computer science students about various topics in the field, they often become confused and tell me what I do is too complicated for them to understand. Although they are able to operate their electronic devices at a more basic level, they seem to be entirely unaware of what the hardware and software based levels of their devices consist of. While there are many complicated things in computer science that require complex resources and knowledge one cannot quickly attain, there are multiple things that can be done with just a few basic components. At Earlham College, we have access to basic components, like Arduino boards and sensors, which can assist in the preparation for flooding. The device I set out to create falls under the category of technologies that can be designed by combining those components with a computer programming language called C.

This is why I set out to produce a low-cost and energy-efficient flood preparation/prevention device that could be deployed in various areas in the world. This device has the potential to save lives, and to give people a chance to save important documents and irreplaceable items.

2. BACKGROUND

There are devices like this that are currently available and being used. The Iowa Flood Center (IFC) teamed up with the U.S. Geological Survey (USGS) to design a device that detects a rise in water level and sends a text message to whomever needs to be notified. Those devices are expensive, especially when one plans to mass-distribute them. The USGS operates these types of devices and they normally cost between $15,000 to $20,000 each. The IFC wanted a cheaper alternative, so they designed a device, shown in Figure 1, that contains a sonar sensor for detecting the distance between the water and itself, a built-in cell phone modem for communication, and a power source charged by solar energy. The USGS agreed to operate them for a much cheaper price of $3,500 due to their simplicity[2].

It is logical that one's initial thought would be that these devices would be implemented in, or near, bodies of water, but they could actually be placed anywhere. Just because a particular area does not have a body of water near it does not mean that a flood could not happen. For example, it is common to hear of basements flooding from substantial rainfall. The reason this happens is because those houses were built in low spots of the terrain. When it rains, the water runs

Figure 1: Water level sensor used by the IFC, provided by Stephen Mally/The Gazette, 2014.

down from the high spots into the low spots where these houses are located, and the water travels through openings and cracks that lead down to the basement. Not only does this cause property damage, it has the potential to ruin irreplaceable items. These damages could be prevented by the implementation of the simple device by providing homeowners with a warning so they can take appropriate action.

The development of a low-cost device offers numerous improvements to all areas in the world. Its simplicity and small size gives people the ability to place this device in almost any area with a WiFi connection.

3. HARDWARE

The hardware used in this project consists of a microcontroller, a soil humidity sensor and module, and a portable source of power, all contained in a waterproof container. Cost and functionality were the main factors in determining which pieces of hardware to use.

3.1 Microcontroller

The main component of this project is the microcontroller. This piece of hardware is programmed to control all other components connected to it. The particular microcontroller I chose to use is the Arduino Yun, shown in Figure 2, because the integrated development environment (IDE) to program it is free to download, and it contains built-in WiFi. The WiFi is essential to the project because it provides a simple method of communication, which is email, between the microcontroller and the receiver of the email. In order for the Arduino Yun to determine whether an email needs to be sent or not, it must compare the readings it gets from the sensor.

The Arduino receives the readings in the form of voltage through an analog input pin. After receiving the voltage, the Arduino converts the analog value into a digital value between 0 and 1023 (0 corresponds to 0V and 1023 corresponds to 5V)[3]. Those readings represent a numerical value that can be compared to one another. These two types of input pins and their values will be further explained in the next section.

3.2 Soil Humidity Sensor & Module

There are two common ways to detect and measure water level. One way is to use a float-arm and a potentiometer. It is comprised of a floating object that is attached to the end of an arm. The other end of the arm is connected to the potentiometer. As the water rises, the floating object also rises. The potentiometer measures the angle of the arm to determine the water level. This method is often used for measuring fuel in the gas tanks in vehicles because of its durability, reliability, and low cost[4]. Since my device is designed to be applicable outdoors, a float-arm has the potential to be damaged by its surroundings. The other way is to use a sensor that detects moisture. The YL-69 sensor, shown in Figure 2, is what is used in this project. It is originally intended to be used as a soil humidity sensor, but it functions the same as a water level sensor by detecting moisture on its prongs.

Along with the YL-69, a module by the name of YL-38, shown in Figure 2, is used to connect the sensor and the Arduino together. The sensor communicates the same as most sensors do by sending measurements in voltage. Voltage, or sometimes referred to as current, is the most common method of communication between sensors the microcontrollers they are synced with. The way the sensors are able to communicate with the microcontrollers are through two kinds of input pins. Sensors connected to a microcontroller through an analog input would read out a smooth and continuous voltage output that is constantly being relayed to its microcontroller for analysis. Contrary to analog inputs, devices that are connected to a microcontroller through a digital input would read out a definitive voltage that remains constant with no continuous fluctuation[3].

Analog inputs are in the form of waves. As time passes by and the connected device is relaying data to the microcontroller, the data can be logged and transformed into a visual representation to simplify the way it is understood. Figure 3 shows that over an unspecified period of time, there is a smooth and continuous wave of voltage being read. An example that utilizes this fluctuating looking signal would be a microphone. Within a microphone, you have a sound sensor that detects when sound is being directed into it. Upon detecting the sound, an analog signal is then sent to the microcontroller to let it know that it is sensing something and action needs to be taken. The microcontroller analyzes the voltage it is receiving from the sensor. It then begins communicating with its other devices connected to it that control the amplification and output of the audio signal through whatever method of amplification being used[5]. This may seem like a complex and time-consuming process, but it is in fact quicker than we could possibly imagine. A more feasible way of thinking about an analog input is to think of it as a roller coaster. Although the roller coaster has a maximum and minimum height (voltage) it can achieve, it can read out any height between that maximum and minimum.

Not only can analog inputs be used for sound, but digital inputs can be too. While their forms of communication using voltage are similar, the actual signal from digital inputs are different than analog inputs. The digital readings that the microcontroller receives look like boxes when represented visually in a graph, shown in Figure 3.

Even though voltage readings from a digital input are constant with no continuous fluctuation, a graph of digital read-

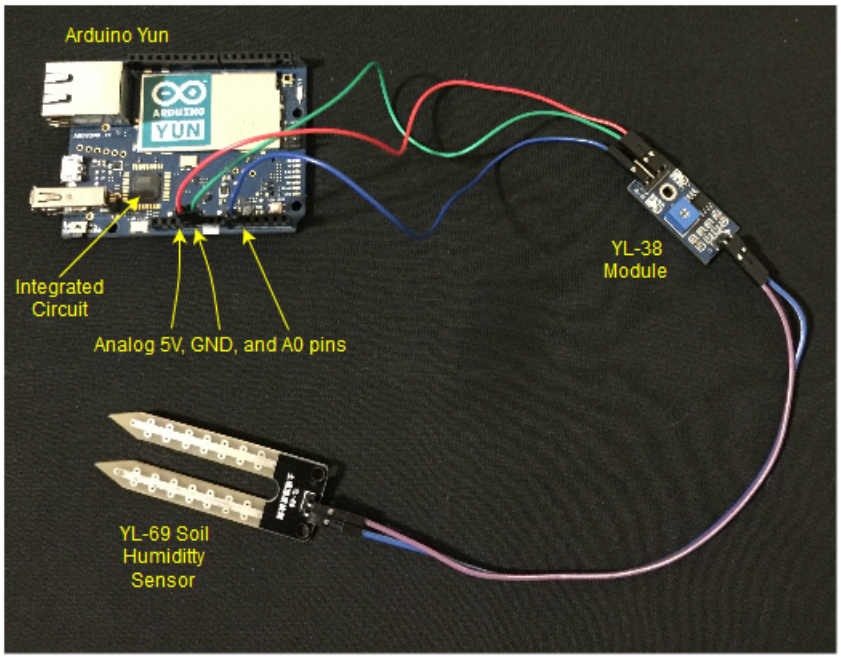

Figure 2: Image showing the hardware and how it is all connected.

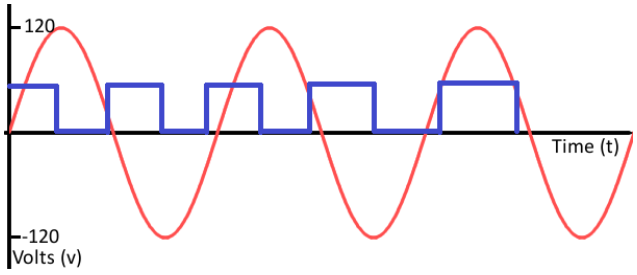

Figure 3: Graph of voltage through an analog input (red) and a digital input (blue), over time, provided by SparkFun Electronics.

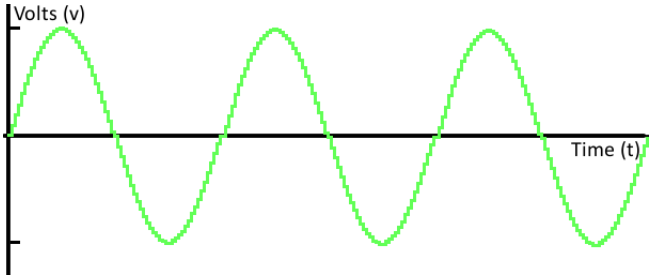

Figure 4: Zoomed out graph of voltage through a digital input over time, provided by SparkFun Electronics.

ings can look like a graph of analog readings. By zooming very far out on a graph of digital readings, as shown in Figure 4, one can see that the graph looks very similar to the red waves in Figure 3. Taking a closer look at Figure 4 shows that the graph still maintains the box-shaped voltage measurements even though it appears to be a continuous wave of voltage.

An example that uses digital input is an integrated circuit. An integrated circuit is the little black chip on the microcontroller. Figure 8 has an arrow pointing to the integrated circuit on the Arduino Yun. This little black chip contains many pathways filled with transistors, resistors, and capacitors that regulate voltage signals traveling around in the microcontroller. Programming the integrated circuit allows for personal customization of the microcontroller. A simple way to think about this is to picture a switch that turns a light on and off. When you turn the switch on, voltage is allowed to travel through an integrated circuit and provide power to the light bulb. By then turning the switch off, the integrated circuit terminates the voltage from the switch and the light bulb. These regulators open and close pathways for voltage instantly, which explains the vertical jumps in voltage from 0V to 5V, as well as 5V to 0V, in Figure 3.

3.3 Power Source and Waterproof Container

In order for the Arduino Yun to be able to operate, it needs a USB power source. Depending on where this device is implemented, a portable source of power may need to be provided. I decided to use a rectangular, rechargeable battery stick, shown in Figure 5, that provides 5V to the device it is connected to via USB and a micro-B port on the other. It fits perfectly in the container with the Arduino Yun and other pieces.

3.4 Total Cost

The main focus of this project is disaster preparation, but the main goal is to develop a device that costs much less to produce and operate while maintaining the efficiency and

Figure 5: Portable and rechargeable 5V battery.

reliability existing devices offer. The costs of the hardware components used in this project are as follows:

- Arduino Yun - $69.95
- YL-69 Sensor & YL-38 Module - $2.99
- Portable Battery - $4.49
- Waterproof Container - $19.95
- Total - $97.38

Compared to the IFC's $3,500 device, this device costs significantly less. To be exact, the total cost for this device costs 97.2% less than the IFC's device.

4. SOFTWARE

The software portion of the project requires an integrated development environment (IDE) that enables the Arduino Yun to be programmed in C. As the Yun contains built-in WiFi, I have chosen to use a platform called Temboo that utilizes email as a method of communication.

4.1 Arduino Software

One of the best characteristics of the Arduino IDE[1] required to program the Arduino Yun is that it is free. This is a crucial part to keeping the cost of this device as low as possible. Some may think that since the software is free to download, there must be a catch or that the software does not come with everything necessary to program Arduino boards. However, this is not the case as the software comes with templates, examples, and a library of functions.

Within the Arduino IDE, there are a couple features that prove to be greatly beneficial to first-time users. After opening the software, the user is presented with a template that briefly explains how the general Arduino programs are structured and operate. From there they have the option to create their own program from that template, or they have the opportunity to start from an example.

The software allows the user to choose from many different example sketches. The reason this is so beneficial is that it provides a way for novices to get comfortable with the programming as well as the interface of the software. Not only do the sketches give users examples of programs, but they provide brief explanations of how to connect the necessary hardware to the Arduino board for testing them. Once the user is comfortable with the software, they can begin creating their own programs. Another useful tactic users can adopt is finding an example program within the Arduino sketches and use it as a building block for creating their own program.

These example sketches would not be able to run properly without the proper libraries that define the functions used to communicate with the Arduino microcontroller. In order for the software to successfully communicate with the Arduino, functions like pinMode(), delay(), and analogRead() need to be defined. Function definitions are provided in Table 1. Arduino has conveniently provided those libraries as a built-in feature in the download of the IDE.

pinMode()	Declares the type of pin, analog or digital, and whether it is outputting or inputting data.
delay()	Declares how long of a pause the program takes before continuing.
analogRead()	tells the Arduino board to retrieve data from a specified pin.

Table 1: List and explanation of the functions mentioned.

4.2 Languages and Platforms

There are multiple programming languages used in the world including C, C++, Python, and Java, just to name a few. The programming language the Arduino software utilizes is C. While it is not the easiest language to learn to code in, C is currently the most-used programming language. The functions mentioned in section 4.1 are examples of the C code used in the program for this device.

The majority of the necessary functions are provided in the Arduino library, but this device required a way of being able to communicate via email so further libraries were needed to enable the use of Temboo. Temboo is a platform that runs and manages pieces of code called Choreos[7]. Choreos have many uses, but this device uses the Choreos to send emails. Temboo is then linked to a Gmail account, which will be explained in the next section of the paper.

5. CONFIGURATION OF THE PROJECT

In the process of designing this device, there were connections that needed to be set up in order for the Arduino Yun to be programmed. After the setup and programming is completed, the device is ready to be tested on a small scale.

[1]Arduino has IDE versions available for Windows, OS X, and Linux[6].

5.1 Setup

The first step was to connect the Arduino Yun to WiFi. This is the most crucial step of the process because without WiFi, there would be no method of communication to alert someone of the rising water level. When powering on the Arduino, it creates a WiFi signal. In order to connect the Arduino to the WiFi, I had to connect my laptop to the signal of the Arduino and type in the IP address of the Arduino in a web browser. From there, I was able to log into the board and choose the WiFi signal I wanted it to connect to. The Arduino then performs a reboot. Once finished, I opened up a program called PuTTY, entered in the IP address of the Arduino associated with Earlham College's WiFi, and created a Secure Shell (SSH)[2], completing the connection to the internet.

Now that there is a connection, making emails possible requires the setup of accounts that send and receive the emails. My initial plan was to write a program to create an email client that uses the SMTP protocol to send the email. Due to the security constraints on Earlham College's WiFi once again, the simplest way to do this was to create a Temboo account and two Gmail accounts. The reason for creating two Gmail accounts is one of them will be receiving the email alerts and the other will be the sender. The Gmail account sending emails will be linked with the Temboo account. Temboo acts as the middle-man in this process. In order to link the Gmail account and the Temboo account, a series of privacy forms that allow access between the two accounts had to be completed. After all permissions were granted, it was time for the last two pieces of hardware to be set up.

The YL-69 soil humidity sensor and the YL-38 module are connected to one another, shown in Figure2, in order to create a connection to the Arduino Yun. The YL-69 has two prongs coming off of it that allow wires to be connected to it. Those wires connect to one end of the module. On the other end of the module, there are four prongs labeled A0 (analog), D0 (digital), GND (ground), and VCC (voltage). This device requires the wires to be connected to the pins A0, GND, and VCC. The opposite ends of those wires are what connect to the Arduino Yun. The A0 wire connects to the A0 pin, the GND wire connects to the GND pin, and the VCC wire connects to the 5V pin. All of these pins are connected to the analog side of the Arduino, not the digital side. The reason the digital pins are not used is because digital readings only read out 0 and 5V, while analog readings read everything in between. Recall that this concept was explained in subsection 3.2. Everything is now set up and ready to be programmed.

5.2 Programs

The first phase was getting an email sent successfully. The example sketches that are included in the Arduino software contained a sketch regarding sending an email through Temboo. This is where the first step was taken because like I said earlier, people often use these sketches as building blocks to their own creations. The code gives the programmer the option to change the target email address to their liking. There is also an additional program called a header file, which contains the personal information required for the use of the Temboo account. The second phase was acquiring readings from the sensor. This process requires just a few lines of code in which an input pin is assigned and two integer variables are created and assigned the data readings received from the sensor. The creation of these programs is shown in Figure 6.

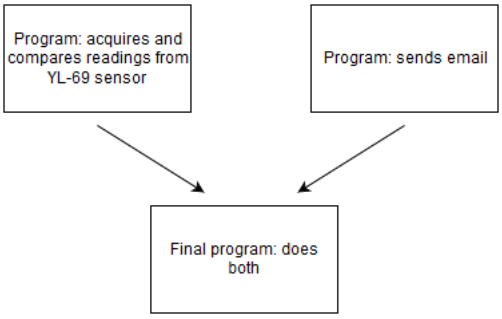

Figure 6: Diagram showing the combined programs making the final program.

Finally, the third phase was blending the two together to create the final program, along with additional code to compare values. Combining the two programs together is simple, but the code for comparing readings with one another is a different story. In order for the readings to be compared, they need to be assigned variables. Using the two integer variables I created, the first two readings are acquired and compared to determine the difference between them. Overwriting each variable by getting new readings is not a valid way of doing this because it would create incorrect comparison results when checking for a rise in water. To solve this problem, an integer variable is created as a counter and set to 0, which is followed by an if statement. This if statement checks the value of the counter, and if it is 0, then a reading is taken from the sensor and the counter is incremented by 1. From here, a second variable acquires a reading from the sensor, and the two readings are compared. Since the if statement only executes one time due to the increased value of the counter, the first variable no longer gets any readings from the sensor. This is where the last line of code comes in. The value of the second variable reading is assigned as the first variable reading. This eliminates the problem of incorrect comparisons by essentially storing the previous reading in a variable that is used for comparison. The difference that is calculated from comparing the readings. The user can change how much the difference has to be. Once the difference between the readings is great enough, an email is sent to notify an individual. Otherwise no action is taken and another reading is acquired. Figure 7 illustrates the process of the data within the device.

5.3 Testing

The Arduino Yun, YL-38 module, and portable battery are all fitted into the waterproof container, shown in Figure 8, while the YL-69 sensor is located outside to obtain water readings. Using the Arduino software, the code is compiled and uploaded to the board. The sensor is then dipped into a bowl of water by hand, and the depth at which the sensor is positioned is varied to simulate rising and falling water

[2]Earlham College's security constraints on their WiFi require the use of PuTTY and using a SSH to be granted internet access.

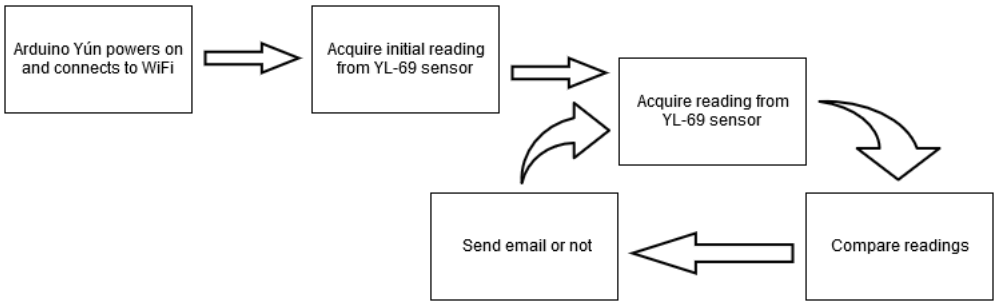

Figure 7: Data flow diagram showing how the device operates.

levels. For testing purposes, a serial monitor is pulled up on the screen to see the values taken from the sensor. Based on the values on the monitor and the actions taken by the Temboo account, one can clearly see how successfully the device is or not. Through just a few test runs and some troubleshooting of the code, the device worked perfectly.

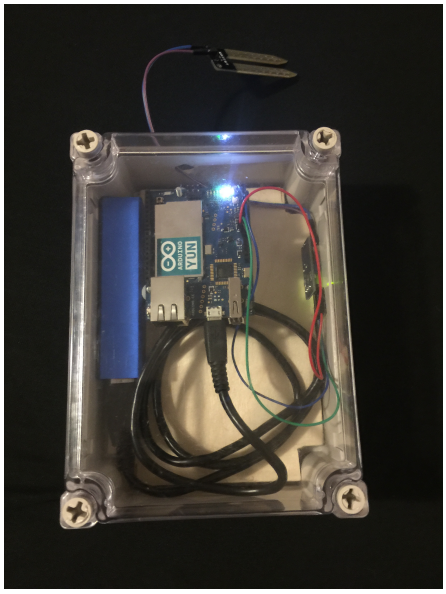

Figure 8: Waterproof container that protects the hardware.

6. FUTURE WORK

The project may be complete but improvements in the sensor and microcontroller could be made. The device could also benefit from additional add-ons such as a new method of communication, a way of saving battery power, and another sensor. Adding these features to the device would greatly increase its effectiveness as well as increase the market in which it could be used for.

6.1 Improvements

This device could be improved by implementing it on a bigger scale with a bigger sensor. The current sensor's prongs are only an inch or two long. So for the device to be used for detecting rising water levels of large bodies of water, the sensor would need longer prongs to compensate for the larger fluctuations of water levels. When dealing with large areas of water, it would make sense to have multiple sensors monitoring the water levels. In the event of a flood, it would be expected that all of the sensors would be alerting the person of authority. If one of the sensors is not detecting flooding, then it is clear that sensor is not functioning properly and needs to be fixed. When the opposite of that event happens, problems arise. If there are multiple sensors in an area and only one of them is reporting flooding, then it could be possible the device is malfunctioning or there is actually flooding occurring. Having an architecture that recognizes which sensors are placed where and which sensors would report flooding first prevents the occurrence of false alarms when a device malfunctions.

When collecting water level readings, the code currently only compares two readings. If a fish were to splash water on the sensor, this would cause the device to receive a reading that indicates a flood. To keep this from happening, an array could be created that collects ten readings in ten seconds, and then the average of those readings is calculated. Ten minutes later, another array of the same size could collect ten new readings and calculate the average of them. The two averages could then be compared, determining whether an email needs to be sent. By creating even more arrays, it makes it possible to store larger amounts of readings. These averages of the readings could be used to create a simple graph showing the rise and/or fall of the water level. This graph could be included in the email alert to provide the receiver with a simple way of interpreting the change in water level over x amount of time.

A question that I have not explored myself is, are there other programmable microcontrollers for a cheaper cost that maintain the efficiency and simplicity of the Arduino? There is an abundance of microcontrollers available, therefore it makes sense to believe there are cheaper options. On the other hand, if there are other cheaper options, then why have companies like the IFC not taken advantage of them? One possible reason is that the cheaper microcontrollers may not offer the same reliability as their current devices.

6.2 New Add-ons

In the beginning of the paper it was said that the IFC's device uses cellular communication instead of WiFi. The good thing about a cellular connection is that there is less of a chance of running into the issues of a power outage since cellular towers have generators. With WiFi, the device relies on power to be supplied to the router it is connected

to. Not all homes have generators so without that power, the device becomes useless. Although when the router does have power, the signal strength is always going to provide a stable connection. Since cellular towers are not all built close together, the only con of a SIM card is that the signal strength varies in different areas. Although both forms of communication have their pros and cons, cellular communication is the direction I plan to take in the future.

To use a Subscriber Identity Module (SIM) card with the Arduino board, an additional piece of hardware called a GPRS 2.0 shield, shown in Figure 9, that acts as a cell phone is required. On the front, the black rectangle connected to the wire is the antenna for the device, and on the back of the shield there is a slot for a SIM card to be inserted. The only con of this method of communication is that it raises the total cost of the device since a SIM card would require money to be loaded onto it to pay for the text messages sent.

Figure 9: GPRS 2.0 shield that utilizes a SIM card for text messaging.

If the device is placed in an area where no power source is available, then this is where the portable battery comes into play. The only problem with the battery is that it will eventually have to be recharged. There are a couple solutions to this problem. Solution 1 would be to use a WDT (Watchdog Timer). A WDT is a timer that can be implemented in code to control when the system resets, uses power save mode, or powers down completely[8]. Using the power save mode would be beneficial to this device because it would allow it to acquire a reading from the sensor, send an email or not, enter power save mode for ten minutes, and repeat the process. Although the battery would still have to be recharged, it would not happen near as often. Solution 2 involves the method of self-charging. Again, the IFC uses this method of recharging the device's battery through a solar panel. Solar energy would charge the battery during the day, while the device runs on the battery power during the night. The only problem a solar panel could have is if clouds were in between the panel and the sun. Cloudy days could prevent the solar panel from receiving enough light to charge the battery. However, combining solutions 1 and 2 by using a WDT in conjunction with the solar panel solves all of the potential issues of using a portable power source. The battery would without a doubt have enough power to supply the microcontroller with on those cloudy days.

Instead of just this water level sensor, an additional sensors could be paired with the Arduino to give it multiple uses in areas. A sensor that would be simple to work with would be one that detects harmful gases. For example, the device could be placed in the basement of a house that is built in a low spot to monitor flooding while it simultaneously monitors the contents of the air for formaldehyde. Formaldehyde is a gas that can cause irritation of the skin as well as trigger asthma attacks. The harmful gas is also believed to be a possible carcinogen[9]. Enabling a single device to have multiple purposes can greatly increase the safety of individuals.

7. CONCLUSIONS

This section is titled "Conclusions" instead of "Conclusion" because throughout the entirety of this project, there was not one time where I was not learning something new. Before I started this project I was not too familiar with the lower level aspect of the combination of hardware and software. I have a much greater understanding of how hardware components work, and that my knowledge in software is going to allow me to continue my work with this device.

In the beginning of the paper, I mentioned how non-computer science students think my field is too complicated for them to understand. It is in my hopes that the people who read this paper are able to grasp a better understanding of how hardware and software work together. Although there are parts of technology that are quite complex, not everything with a computer is as confusing as one might think, especially this device.

Not only does the device I set out to create work, but I accomplished the main goal of keeping the total cost at a more than reasonable amount of less than $100. Although this device has its pros and cons, I strongly believe that it can be improved to be just as effective as the IFC's device at a fraction of the cost. A low-cost device that can monitor multiple potentially harmful things in all areas of the world would prove to be extremely beneficial to the health of the human race. After all, it is the well-being and safety of us that should ultimately matter most.

8. ACKNOWLEDGMENTS

I would like to thank Garrett York for providing me with the idea of this device and for sharing his knowledge of Arduino boards with me. I would like to thank George Crowson as well for his assistance with the WiFi configuration of the Arduino Yun. Finally, a special thank you to Charlie Peck for answering my questions throughout this project.

9. REFERENCES

[1] D. Guha-Sapir, R. Below, and P. Hoyois. (1994) Em-dat: The cred/ofda international disaster database. [Online]. Available: http://emdat.be/disaster_list/index.html

[2] O. Love and T. Gazette. (2014) Sensor to help better gauge flood threats. [Online]. Available: http://www.thegazette.com/subject/news/sensor-to-help-better-gauge-flood-threats-20141009

[3] V. Boonsawat, J. Ekchamanonta, K. Bumrungkhet, and S. Kittipiyakul, "Xbee wireless sensor networks for temperature monitoring," in *the second conference on application research and development (ECTI-CARD 2010), Chon Buri, Thailand*, 2010.

[4] W. J. Fleming, "Overview of automotive sensors," *Sensors Journal, IEEE*, vol. 1, no. 4, pp. 296–308, 2001.

[5] Jimbo. (2013) Analog vs. digital. [Online]. Available: https://learn.sparkfun.com/tutorials/analog-vs-digital

[6] Massimo, David, Tom, and David. (2008) Arduino. [Online]. Available: https://www.arduino.cc/

[7] T. Temboo. (2015) Arduino yÃžn: Temboo comes preloaded on each arduino yÃžn. [Online]. Available: https://temboo.com/arduino/yun/yun-and-temboo

[8] M. Barr. (2001) Introduction to watchdog timers. [Online]. Available: http://www.embedded.com/electronics-blogs/beginner-s-corner/4023849/Introduction-to-Watchdog-Timers

[9] D. J. Walke. (2005) Nrdc: Dangerous chemicals in the home. [Online]. Available: http://www.nrdc.org/health/home/fchems.asp