Final Undergraduate Research Opportunity Program Report
to the Louisiana Sea Grant College Program

January 31, 2022

Custom Design, Fabrication, and Testing of Environmental Data Systems for Use in Oyster Aquaculture

Undergraduate Researchers

Mary H. Conde, LSU Department of Biological Engineering
Caleb L. Gryder, LSU Department of Electrical and Computer Engineering

Project Advisors

Sarah Bodenstein, doctoral student, LSU School of Renewable Natural Resources
Dr. Jack Koch, postdoctoral researcher, Aquatic Germplasm and Genetic Resources Center
Dr. Terrence Tiersch, professor and director, Aquatic Germplasm and Genetic Resources Center

Abstract

Oyster aquaculture has been significant to the Louisiana seafood industry for more than a century. Emerging challenges are driving the need to better understand environmental factors involved in farming oysters. This project aimed to design an open-hardware logger system that would allow farmers and researchers to measure some of these factors. The Oyster Logger, a data logging device for use in field applications, was designed and tested at the Aquatic Germplasm and Genetic Resources Center (AGGRC), with an emphasis placed on casing testing and measurement accuracy. The goal was to fabricate an open-hardware device with advanced data collection capabilities that could help to improve the productivity of oyster aquaculture. The objectives of this project were to: 1) design prototype devices; 2) fabricate prototype devices which collect, store, and transfer temperature and movement data, and 3) test prototype devices for proper operation and to improve the design. While not completely waterproof itself, the casing kept the electrical components of the Oyster Logger in a compact, stable position for data collection purposes. From an engineering perspective, faults in the real time clock (RTC) module and other components caused minor measurement errors, but these inaccuracies were deemed to be insignificant in applied settings. These results show potential for the project to grow and act as a foundation for other logger systems that can be customized to the needs of oyster farmers and researchers.

1. Introduction

Aquaculture is the cultivation of aquatic organisms including freshwater and saltwater populations under controlled or semi-natural conditions (Fisheries, 2011). Oyster aquaculture includes the practice of spawning oysters in hatcheries and raising them in cages or bags in natural waters for the market (Fisheries, 2021). According to the Louisiana Sea Grant strategic plan, Louisiana is the leading state in oyster aquaculture, producing 59% of oyster landings in the Gulf of Mexico (*"Strategic plan 2018-23"*). Sustainable aquaculture farms continue to increase to meet the demands of the growing population and provide year-round jobs which provide sustainable economic growth. Oyster aquaculture in Louisiana helps maintain healthy estuaries and sustain the local fishing economy and food culture (Leonhardt et al., 2017). Since the world's oyster reefs have become depleted due to overharvesting, Louisiana has been one of the last bastions of wild oysters (Safina et al., 2011).

In recent years, Louisiana's wild oyster populations have declined due to overfishing, hurricane damage, and a drop in water quality (Fisheries, n.d.). Some oyster farming, such as off-bottom farming, allows farmers to raise and lower their oyster lines from the water. Off-bottom farming allows for some control over conditions in which oysters are raised, providing several advantages for raising oysters. One of the more important factors in off-bottom farming is the effect of wave action due to oysters being suspended in water (Banks et al., 2016). Wave action influences the feeding, growth, and health of oysters, and water quality, including salinity, temperature, dissolved oxygen, and pH, also affects their health and growth (Banks et al., 2016). The water quality data can also be used to scout an area's potential to become a farm site. For this reason, the team aimed to make a device that would collect and store temperature and movement data over time. This way oyster farmers could examine this data and determine how oyster growth was being affected by these environmental factors and adjust accordingly.

Current devices used to collect environmental data are expensive and do not have software a user or programmer can change. A device that could be made with consumer components, assembled by a novice, and be customized would allow any individual to build and use the device as they need while still being economical. This would also be more beneficial for farmers in Louisiana who wish to use more than one device for their different sites. In this project, the team proposed to develop a technology-based solution to standardize measurements of environmental factors to help understand their effects on cultured oysters grown off-bottom in the field. The goal of the project was to fabricate an open-source device, the Oyster Logger, with advanced data collection capabilities that could help to improve the productivity of oyster aquaculture. The objectives of this project were to: 1) design prototype devices; 2) fabricate prototype devices which collect, store, and transfer temperature and movement data, and 3) test prototype devices for proper operation and to improve the design.

2. Methods

2.1. Design

The two major parts of this project were the housing and the electrical components that went into the housing. The electrical components were designed to collect, store, and transfer temperature and movement data. The housing was designed to keep the electrical components compact and provide a barrier from water.

2.1.1. Housing

The housing for the Oyster Logger was designed in two separate components, a 3-D printed base and a 3-D printed lid. All versions of the base and lid were rendered in Fusion 360 (2020, Autodesk , San Rafael, California) by using the line, three-point circle, extrusion, fillet, and shell tools. Several changes were made in a second version of the lid: Texture was added to the top of the lid using the extrusion command in Fusion 360; larger corners were added with the fillet command to make the appearance more organic; and the lid wall height was increased in to have the lid fit firmly on the base.

2.1.2. Electrical Components

While the housing was being developed, the electrical components were designed and tested with the appropriate code and combined into a functional design on a breadboard, which is a common tool used to test circuits by providing a common base for electrical components to be connected and replaced. These components included the Rocket Scream Mini Ultra (Mini Ultra 8), MCP9808 Temperature Sensor Breakout Board (MCP9808 High Accuracy I2C Temperature Sensor), ADXL343 Accelerometer (ADXL343 - Triple-Axis Accelerometer), DS3231 Real Time Clock (RTC) (Adafruit DS3231 Precision RTC), and MicroSD card Breakout Board (MicroSD card breakout board+). The code for each component was adopted from Arduino libraries, for which download instructions are found in the modules' associated web pages, along with example codes provided for each component.

The accelerometer, temperature sensor, and real time clock (RTC) module all operated using an Inter-Integrated circuit (I^2C) communication protocol, a set of formatting rules and regulations for digital communication (Communication Protocols, 2020). This made combining the three onto a breadboard simple because they shared the same two pins of the Arduino for communication (Figure 1). The SD card module used a Serial Peripheral Interface (SPI) communication protocol that required four separate communication wires. Because it was the only component that required SPI, the additional wires did not take up much more space than one of the I^2C-based components. Each I^2C-based component had a different communication address, preventing any potential communication interference. Uniting the individual codes was made simple as each component acted separately and could be initialized and utilized as such.

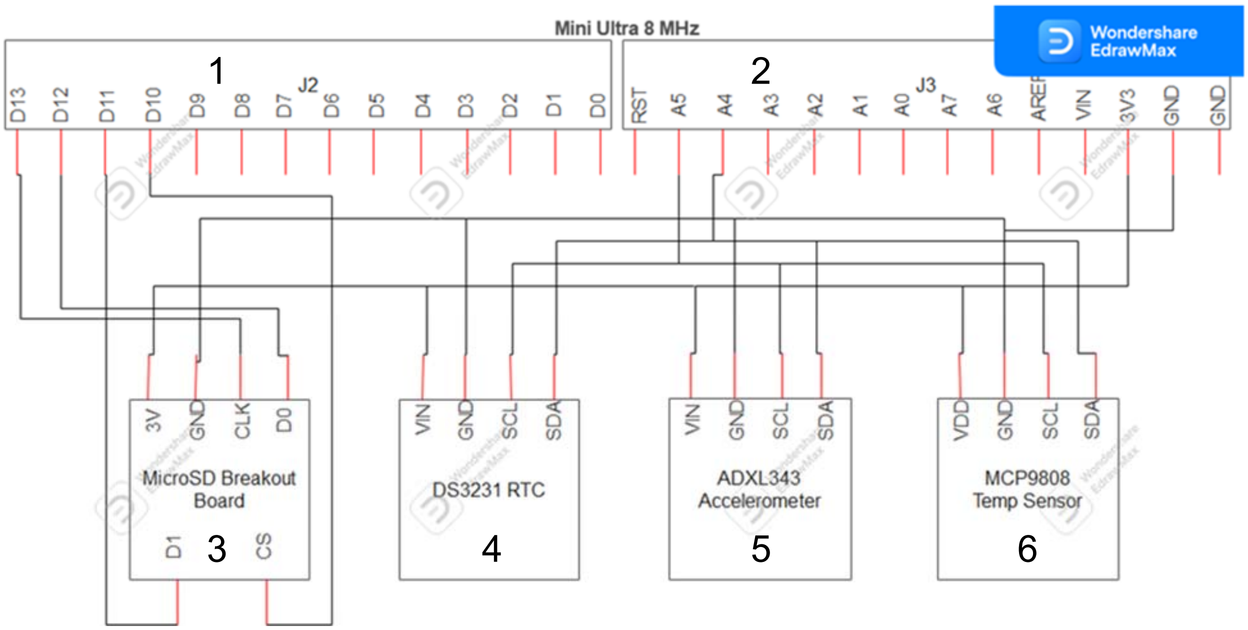

Figure 1: The wiring diagram for the functional logger prototype. Blocks 1 and 2 represent the separate rows of pins on each side of the Mini Ultra (Arduino). Blocks 3, 4, 5, and 6 represent the microSD breakout board, the RTC module, the accelerometer, and the temperature sensor, respectively. Abbreviations: GND- Ground; 3V/VIN- 3 Volt Input; 3V3- 3 Volt Output; CLK- Clock; CS- Chip Select; SCL- Serial Clock; SDA- Serial Data

Each component, when able, was tested individually from the other components to assure that the code and wiring were functional. The temperature sensor and accelerometers were first tested independently (described below), and then the microSD card was tested by storing data from the operational temperature sensor onto it as a text file. Initially, the plan was to store the temperature and accelerometry data in two separate files on the SD card. However, attempting to do so caused unexpected issues, so, instead, both sets of data were saved as one line for each measurement increment. The data was then separated using the data partitioning options on Excel. Lastly, the RTC module was tested individually (described below) and then combined with the other components onto a breadboard (Figure 2). Code is provided in Appendix 2.

2.2. Fabrication

2.2.1 Housing

The Oyster Logger housing base and lid were 3-D printed on an Original Prusa i3 Mk3 (using settings described in Appendix 1) using polylactic acid (PLA) filament (ZYLtech Engineering, LLC,

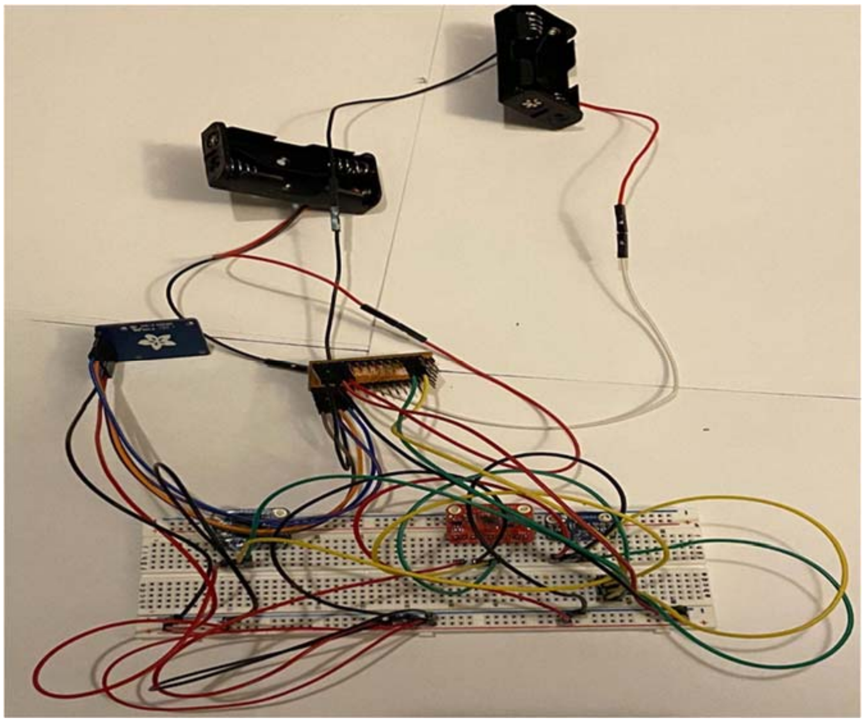

Figure 2: The fully constructed breadboard-based prototype.

Houston, Texas). The second version of the lid for the oyster logger was printed with modified settings (Appendix 1).

2.2.2 Electrical Components
To move from a testing system to an official prototype, the breadboard-based design was transferred onto a more compact base that could fit within a smaller case. Initially, the components were going to be connected to the Arduino by soldering the wires to the pins because multiple components had to use the same pins of the Arduino. However, this method did not work as planned as it did not provide reliable output. While this initial method was being analyzed, a new protoboard-based design was tested. This was smaller than a breadboard but allowed for a similar level of organization. Because attaching and detaching the components, which required soldering, would be more difficult than doing so on a breadboard, special care was taken to plan how the components would be connected. After a sketch was made to guide the process, the I^2C components were attached to the protoboard while the Arduino and microSD card module were kept separate to create the first Oyster Logger prototype.

2.3. Prototype Testing

2.3.1. Housing
To test if the housing for the Oyster Logger was waterproof, the housing was placed under water with pH paper and tissue inside which were used as water leak indicators. Prior to the placing the housing in the water, the weight of the tissue was measured and recorded, and a picture was taken of the pH paper (Figure 3) to determine if the color changed. If no water leaked through the housing, the pH paper would remain orange, but if exposed to water the paper would change from orange to green. If the tissue had the same weight as measured initially, that would indicate no exposure to water as well. To prevent the housing from floating

due to the buoyancy of the materials, it was secured in a net and at a depth of 3 ft After 24 hr under water, the housing was examined for water, the pH paper was examined for color change, and the tissue was weighed.

Figure 3: Housing base and lid version one before testing.

In the second test of water resistance, the second version of the lid was used for testing. The same method as described above was used with the following modifications. A silicone sealant, a liquid form of adhesive, was applied to the inside of the lid. The base and lid of the housing were pressed together to make the seal and left to cure for 24 hr before being placed in water at a depth of 3 ft After 48 hr the housing was taken out of water to observe the pH paper and tissue.

2.3.2. Electrical Components

Before constructing the system, each electronic module was tested individually. These modules included the temperature sensor, accelerometer, RTC module, and microSD card module. The temperature sensor and accelerometer were individually connected to the Arduino and the appropriate pins. Code from coding libraries taken from the modules' affiliated websites and used to command the modules to perform the tasks of reading information from the sensors and writing that information onto the Arduino compiler's serial monitor. These libraries also provided a method for checking that the Arduino could detect each module by giving a pre-written output if a signal was not received from the module. The microSD card module was tested by taking example code from the provided libraries and combining it with that of the temperature sensor. This code created a new file, named "datalog.txt", on the microSD card and wrote the output from the temperature sensor into the new file as well as onto the Arduino serial monitor. If the new file was created and written onto with an output that matched the pre-written output placed in the code and the results on the serial monitor, the microSD card module was working properly. Due to time constraints, the RTC module was tested after all the other electrical components were connected and tested as a whole.

To test the system when connected on a breadboard, it was placed in the AGGRC temperature-controlled room, which maintained a known series of temperature changes. If the temperatures logged from the breadboard system showed a similar series of fluctuations, the temperature sensor-to-microSD card part of the logger was functional. During this experiment, notes were taken on when the logger was placed in the cold. If the times that the temperature began to decrease along with when the accelerometer detected movement coincided with those of the notes taken, the RTC and accelerometry modules were deemed functional. Similar tests were done inside the AGGRC roll-up door room, which had a temperature profile similar to ambient conditions outside of the AGGRC.

To gain a more precise assessment of how accurate the temperature and accelerometry sensors were, the breadboard system was tested against a commercial temperature sensor or through analysis of when it began and ended. For the temperature measurements, the breadboard system was left inside the roll-up door room along with a HOBO temperature logger from Onset Computer Corporation (Bourne, MA), which contained three functional temperature probes that each record their own measurements. To test the logger's accelerometry, a separate test was performed. The logger was set to record and was shaken for certain intervals of time. These intervals were recorded and compared to the accelerometry results. Once the breadboard-based system was tested against the HOBO logger, trials to imitate real-world conditions that the Oyster Logger would experience began with the new protoboard-based design. To do so, the electronics were vacuum sealed in a plastic bag, using a vacuum sealer, and placed inside the finished casing. The prototype, the finished electronics system and the case, were submerged inside a tank of water, approximately 3 ft deep, along with one of the HOBO temperature probes. Both loggers were submerged for 6 d before being taken out of the water, and their results were compared.

The four data sets from the temperature tests with the HOBO logger in the AGGRC roll-up door room, one from the breadboard system, and three from the probes, were placed on a graph and compared. The average differences between the breadboard temperature measurements and each of the probe measurements were calculated. To simplify the graph, the data points from each of the three probe data sets were averaged, and the average was graphed alongside the data from the prototype.

3. Results

3.1. Design

3.1.1 Housing

The housing dimensions of the sketch rendered in Fusion 360 for the base and lid were 110 mm by 110 mm with a height of 45 mm.

3.1.2 Electrical Components

To indicate if the wiring connections were properly made, the accelerometer and temperature sensor tests would yield one of two results on the Integrated Development Environment's (IDE's) serial monitor, either an error message or readings from the sensors. One new file was created, on which both sets of data, accelerometry and temperature readings, were recorded due to issues in recording on two separate files. The RTC module, initially, recorded a constant default time instead of the time during which it operated. Once changes to the code were made, it began providing a time that was accurate to when it began recording, incremented at intervals that matched the delay written into the program.

3.2. Fabrication

3.2.1 Housing
The results from the print of the first version of the housing can be seen in Figure 4. The printed result of the second version of the lid (Figure 5) was fabricated to increase the surface area touching the base of the housing and smooth out the corners.

Figure 4: Housing base and lid version one.

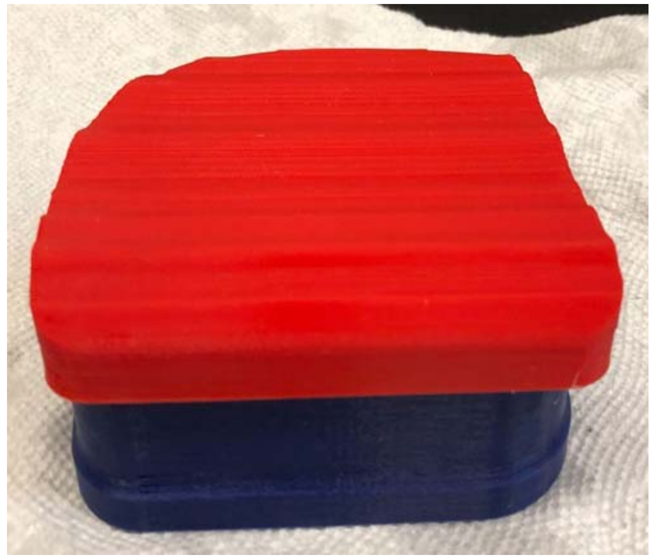

Figure 5: Housing base and lid version two.

3.2.2 Electrical Components
While the breadboard design worked, it was too large to fit within a the case along with the batteries needed to power it. The next design, which used direct wire connections between each of the components, would not provide any sensor output. The protoboard-based design, however, fit inside the casing and provided testable output.

3.3. Prototype Testing

3.3.1 Housing
After the housing was recovered from its underwater testing, water was observed in the base. The pH paper had turned from orange to a dark green, meaning that it had been exposed to water. In addition, the tissue weighed 2.4 g more after the housing was taken out of the water. After the second test, water was observed spilling out of the housing. The pH paper color changed from orange to white. The tissue gained 2.9 g which also indicated exposure to water.

3.3.2. Electrical Components
Testing in the temperature-controlled room resulted in a temperature range of 15-17°C, which matched the 15-17°C range observed within the room. The test done alongside the HOBO

temperature logger resulted in four plots, three from the HOBO and one from the breadboard system. The three HOBO probe plots remained consistent with each other while the graph from the breadboard system matched the fluctuations and variations of the averaged data set (Figure 6) within less than a degree (there was a 0.2°C difference). The time recorded by the breadboard system was also approximately 3 min later than the actual time that it began. The acceleration data graph (Figure 7) showed three separate instances of movement, which matched the number of times that the prototype was shaken. For the test done in water, a hole in the vacuum-sealed bag allowed water to enter, damaging the electronics and preventing data collection.

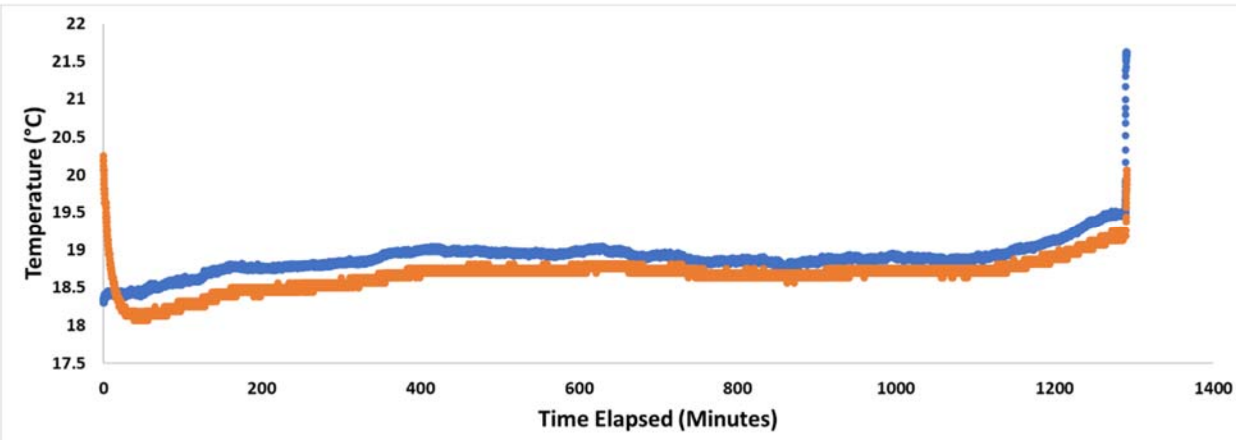

Figure 6: The temperature recordings from the logger prototype (blue) and the average of the three HOBO logger temperature probe measurements (orange) over 21 hrs.

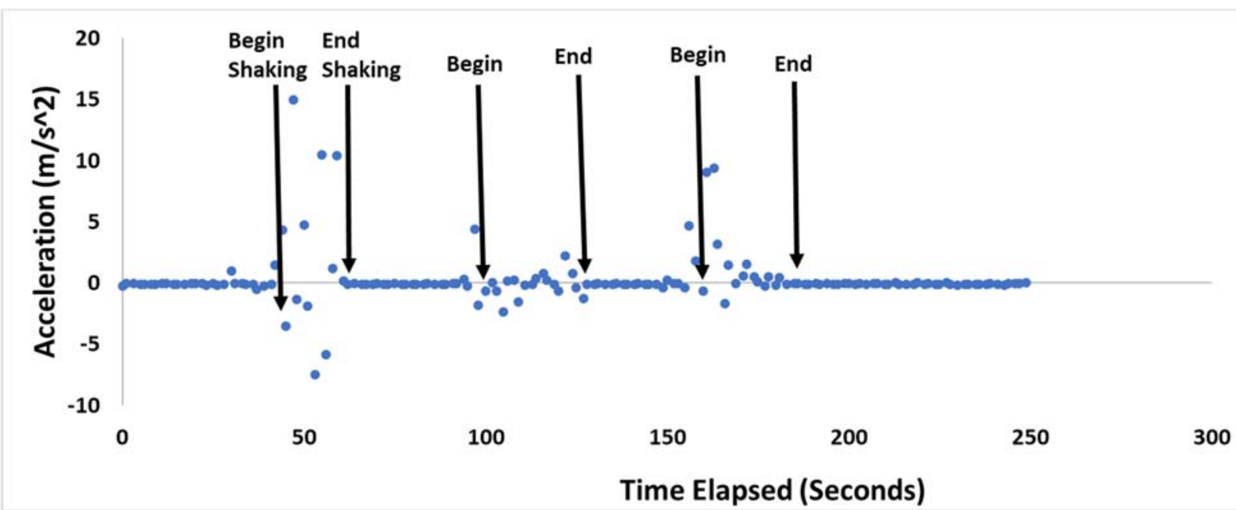

Figure 7: The acceleration recording of the logger prototype and instances during which it was shaken across a 4-min period.

4. Discussion

4.1. Design

4.1.1. Housing

The housing for this device was designed to be placed in hanging baskets used in longline off bottom systems of oyster aquaculture, where baskets are suspended at different heights for growth and cleaning. Therefore, this casing may not be suitable for all types of oyster bags used in oyster aquaculture.

4.1.2. Electrical Components

Each of the modules chosen had to be capable of operating separately to support the logger's proposed open-source functionality. This meant that coding each component individually made it simple to adjust parts of the code. This would allow users to target components that had to be changed to fit their specific needs. The program checking each module individually before performing its main functions made troubleshooting simple for users as well.

4.2. Fabrication

4.2.1 Housing

When printing the housing the casing was made to have a higher infill density to try and prevent water leakage. The base was printed upside down to create a smooth surface where an adhesive could be applied. Since 3-D prints deposit one layer at a time, tiny gaps can form between layers and allow leakage. To prevent this, the infill density was increased from 15% to 45%.

4.2.2 Electrical Components

When constructing the prototype, size and flexibility were major factors to take into consideration. The initial breadboard-based design was useful for testing the system's functionality, but too large and unstable to last as the final prototype. Connecting the components on the protoboard allowed flexibility with some of the components while allowing those that shared the same communication protocol to stay connected in an organized fashion. This design proved to be more promising than the previous ones in that it was more compact and stable due to the soldered connections, and adjustable based on the users' needs.

4.3. Prototype Testing

4.3.1. Housing

The housing for the oyster logger was not completely waterproof, but it was used to keep the electronic components compact and in a standard position. This way the measurements recorded for acceleration would not be affected by shifting or sliding of the electronic components. Using filaments such as polypropylene (PP) or polyethylene terephthalate glycol-modified (PETG) could make the housing waterproof in the future, because these filaments have interlayer adhesion and better waterproofing characteristics. Some possible techniques to waterproof the housing would be to print thicker layers, to over-extrude the filament when printing, and adding more shells. The use of an alternative additive manusufacturing technology, 3-D resin printing with photo-curable resin, might also be a good idea for future versions because of its precise layer heights of 0.01 mm.

4.3.2. Electrical Components

The oyster logger project provided a system that can assist with the study and commercial farming of oysters. Its two main functions, temperature sensing and accelerometry, appeared to be accurate when tested. Based on other research, the temperature difference between the prototype and the HOBO was insignificant (0.2˚C) for recording the oyster environment. While the accelerometry was not quantitatively tested against a commercial accelerometer, the qualitative testing demonstrated that the recorded acceleration matched with when the logger was moved. For future study, the temperature and accelerometry tests could be expanded by adapting the housing to be fully waterproof and placing the entire Oyster Logger inside an oyster cage along with a commercial accelerometer and temperature logger. Such test conditions would evaluate how accurately the accelerometer would portray the movements of the Oyster Logger within the field along with how much time it takes for the temperature sensor to detect changes in the water temperature.

5. Conclusions

This project provides a solid foundation to help future users. For example, applications such as testing salinity, pH levels, and water presence can expand the capabilities of this device. As previously stated, the accelerometer could be tested against commercial models to gain a better idea of the accuracy. One aspect that could be improved beyond the precision of the sensors is the transfer of data from the logger to the user. Currently, the logger stores data onto a microSD card that is removed from the device when the card is full, or the device battery dies. This process requires the user to retrieve the device. Using Bluetooth to deliver the data would reduce the number of times that the user would have to retrieve the logger. It would also be possible to extend the battery life to further reduce the number of times that the device would need to be removed from the water. This may be done by coding sleep states for the logger so that it consumes less power between each logging interval. It also may be effective to use smaller, coin-cell batteries. This style of open-hardware device provides availability for change so that users can adjust features to fit their own needs. While the logger provides a stable basis in accelerometry and temperature sensing along with a protective casing, this flexibility would allow researchers and farmers to custom design the measurements they deem necessary to better understand environmental factors in the field.

References

Adafruit DS3231 Precision RTC Breakout. *Adafruit*, Retrieved January 29, 2022, from https://www.adafruit.com/product/3013?gclid=Cj0KCQjw7MGJBhD-ARIsAMZ0eetiAt_y1b5PIqdbFvHEZtpSA7An-QDQXb8SdWl9Ybt8bOSwqo2MnDYaAk7FEALw_wcB

ADXL343 - Triple-Axis Accelerometer (+-2g/4g/8g/16g) w/ I2C/SPI. *Adafruit*, Retrieved January 29, 2022, from https://www.adafruit.com/product/4097#description

Banks, P., Beck, S., Chapiesky, K., & Isaacs, J. (2016, November 23). Louisiana Oyster. Retrieved January 29, 2022, from https://www.wlf.louisiana.gov/assets/Resources/Publications/Marine_Fishery_Management_Plans/2016_Oyster_Fishery_Management_Plan.pdf

Beddows, P. A., & Mallon, E. K. (2018). Cave pearl data logger: A flexible arduino-based logging platform for long-term monitoring in harsh environments. *Sensors*, *18*(2), 530. https://doi.org/10.3390/s18020530

Casas, S. M., Lavaud, R., La Peyre, M. K., Comeau, L. A., Filgueira, R., & La Peyre, J. F. (2018). Quantifying salinity and season effects on eastern oyster clearance and oxygen consumption rates. *Marine Biology*, *165*(5). https://doi.org/10.1007/s00227-018-3351-x

Communication Protocol (2020, September 30). *Techopedia*, https://www.techopedia.com/definition/25705/communication-protocol

Fisheries, N. O. A. A. (2021, November 4). A look at Oyster Aquaculture. NOAA. Retrieved January 29, 2022, from https://www.fisheries.noaa.gov/feature-story/look-oyster-aquaculture

Fisheries, N. O. A. A. (n.d.). *Eastern oyster*. NOAA. Retrieved January 31, 2022, from https://www.fisheries.noaa.gov/species/eastern-oyster

Leonhardt, J. M., Casas, S., Supan, J. E., & La Peyre, J. F. (2017). Stock assessment for Eastern oyster seed production and field grow-out in Louisiana. *Aquaculture*, *466*, 9–19. https://doi.org/10.1016/j.aquaculture.2016.09.034

MCP9808 High Accuracy I2C Temperature Sensor Breakout Board, Adafruit, Retrieved January 29, 2022, from https://www.adafruit.com/product/1782?gclid=CjwKCAjw0qOIBhBhEiwAyvVcf8HiRC16pOG6gQew28NBtIPVbXWUheLBMDawvGwLGAFk5LMzuQmFLRoCGiIQAvD_BwE

MicroSD card breakout board+, *Adafruit*, Retrieved January 29, 2022, from https://www.adafruit.com/product/254

Mini Ultra 8 MHz (Arduino Compatible), *Adafruit*, Retrieved January 29, 2022, from https://www.rocketscream.com/blog/product/mini-ultra-8-mhz-arduino-compatible/

Rybovich, M., Peyre, M. K., Hall, S. G., & Peyre, J. F. (2016). Increased temperatures combined with lowered salinities differentially impact oyster size class growth and mortality. *Journal of Shellfish Research*, *35*(1), 101–113. https://doi.org/10.2983/035.035.0112

Safina, C., Safina, C., Rubin, S., Rubin, S., George, Z. S., & George, Z. S. (2011, May 29). *Oyster aquaculture offers hope for Louisiana Fishery and the Gulf*. Yale E360. Retrieved January 30, 2022, from https://e360.yale.edu/features/oyster_aquaculture_offers_hope_for_louisiana_fishery_and_the_gulf

Strategic plan 2018-23. (n.d.). Retrieved January 29, 2022, from http://strategicplanning.baruch.cuny.edu/wp-content/uploads/sites/12/2020/04/Baruch-College-Strategic-Plan-2018-2023.pdf

What is aquaculture? What is aquaculture? | National Oceanic and Atmospheric Administration. (2011, June). Retrieved January 29, 2022, from https://www.noaa.gov/stories/what-is-aquaculture

Appendices

Appendix 1: Printer Settings

Appendix 1.1: 3-D printing settings* for housing base.

Quality	Layer Height	0.2 mm
	Initial Layer Height	0.2 mm
	Line Width	0.4 mm
Walls	Wall Line Count	2
	Horizontal Expansion	0
	Z Seam Alignment	Sharpest Corner
Top/Bottom	Top Layers	3
	Bottom Layers	3
Infill	Infill Density	45%
	Infill Pattern	Lines
Material	Printing Temperture	215°C
	Printing Temperture Initial Layer	215°C
	Build Plate Temperture	60°C
	Flow	100%
Speed	Print Speed	60 mm/s
	Infill Speed	60 mm/s
	Wall Speed	30 mm/s
	Travel Speed	120 mm/s
	Initial Layer Speed	30 mm/s
Travel	Enable Retraction	Yes
	Retraction Distance	0.8 mm
	Retraction Speed	35 mm/s
	Z Hop When Retraction	No
Cooling	Enable Print Cooling	Yes
	Fan Speed	100%
Support	Generate Supprt	Yes
	Support Placement	Everywhere
	Support Overhang Angle	50 °
	Support Pattern	Zig Zag
	Support Density	15%
Build Plate Adhesion	Build Playe Adhesion Type	Skirt
	Skirt Line Count	1
	Skirt Distance	3mm
Cura Settings	Size	90%
	X	104.47 mm
	Y	104.47 mm
	Z	36 mm
	Anticipated Weight	87 g

*The Original Prusa i3 mk3s+ (Prusa Research, Partyzánská 188/7A Praha 7 - Holešovice, 170 00 Czech)

Appendix 1.2: 3-D printing settings* for housing lid version one.

Quality	Layer Height	0.2 mm
	Initial Layer Height	0.2 mm
	Line Width	0.4 mm
Walls	Wall Line Count	2
	Horizontal Expansion	0
	Z Seam Alignment	Sharpest Corner
Top/Bottom	Top Layers	3
	Bottom Layers	3
Infill	Infill Density	25%
	Infill Pattern	Lines
Material	Printing Temperture	215°C
	Printing Temperture Initial Layer	215°C
	Build Plate Temperture	60°C
	Flow	100%
Speed	Print Speed	60 mm/s
	Infill Speed	60 mm/s
	Wall Speed	30 mm/s
	Travel Speed	120 mm/s
	Initial Layer Speed	30 mm/s
Travel	Enable Retraction	Yes
	Retraction Distance	0.8 mm
	Retraction Speed	35 mm/s
	Z Hop When Retraction	No
Cooling	Enable Print Cooling	Yes
	Fan Speed	100%
Support	Generate Supprt	Yes
	Support Placement	Everywhere
	Support Overhang Angle	50 °
	Support Pattern	Zig Zag
	Support Density	10%
Build Plate Adhesion	Build Playe Adhesion Type	Skirt
	Skirt Line Count	1
	Skirt Distance	3mm
Cura Settings	Size	90.5%
	X	134.5 mm
	Y	124.39 mm
	Z	25.52 mm
	Anticipated Weight	69 g

*The Original Prusa i3 mk3s+ (Prusa Research, Partyzánská 188/7A Praha 7 - Holešovice, 170 00 Czech)

Appendix 3.3: 3-D printing settings* for housing lid version two.

Quality	Layer Height	0.2 mm
	Initial Layer Height	0.2 mm
	Line Width	0.4 mm
Walls	Wall Line Count	2
	Horizontal Expansion	0
	Z Seam Alignment	Sharpest Corner
Top/Bottom	Top Layers	3
	Bottom Layers	3
Infill	Infill Density	45%
	Infill Pattern	Lines
Material	Printing Temperture	215°C
	Printing Temperture Initial Layer	215°C
	Build Plate Temperture	60°C
	Flow	100%
Speed	Print Speed	60 mm/s
	Infill Speed	60 mm/s
	Wall Speed	30 mm/s
	Travel Speed	120 mm/s
	Initial Layer Speed	30 mm/s
Travel	Enable Retraction	Yes
	Retraction Distance	0.8 mm
	Retraction Speed	35 mm/s
	Z Hop When Retraction	No
Cooling	Enable Print Cooling	Yes
	Fan Speed	100%
Support	Generate Supprt	Yes
	Support Placement	Touching Building Plate
	Support Overhang Angle	50 °
	Support Pattern	Zig Zag
	Support Density	15%
Build Plate Adhesion	Build Playe Adhesion Type	Skirt
	Skirt Line Count	1
	Skirt Distance	3mm
Cura Settings	Size	91%
	X	105.05 mm
	Y	105.05 mm
	Z	16.42 mm
	Anticipated Weight	58 g

*The Original Prusa i3 mk3s+ (Prusa Research, Partyzánská 188/7A Praha 7 - Holešovice, 170 00 Czech)

Appendix 2: Operation Code

This code was used for the prototype to record sensor readings and store them onto a microSD card. The first section before "void setup" initialized the temperature sensor, accelerometer, and RTC. The "void setup" section tested each of the modules to see if they are functional. Once these steps were completed, the "void loop" section took data from the sensors and RTC, opened a file on the microSD card, and stored a string containing the sensor data onto the microSD card.

```
#include <SPI.h>
#include <SD.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL343.h>
#include "Adafruit_MCP9808.h"
#include "RTClib.h"

RTC_DS3231 rtc;
// Create the MCP9808 temperature sensor object
Adafruit_MCP9808 tempsensor = Adafruit_MCP9808();
// Create ADXL343 sensor
Adafruit_ADXL343 accel = Adafruit_ADXL343(12345);

char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday"};

const int chipSelect = 10;

void setup() {

 //test for temp sensor
 if (!tempsensor.begin(0x18)) {
  while (1);
 }
 //test for accelerometer
 if(!accel.begin())
 {
  //there may be a problem with the accelerometer
  while(1);
 }
 //adjust the accelerometer range
 accel.setRange(ADXL343_RANGE_2_G);

 //test for RTC
 if (! rtc.begin()) {
  abort();
 }
```

```
 if (rtc.lostPower()) {
  // When time needs to be set on a new device, or after a power loss, the
  // following line sets the RTC to the date & time this sketch was compiled
  rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
 }


 //Setup SD card module
 // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect)) {
   // don't do anything more:
   while (1);
  }
}
void loop() {

  DateTime now = rtc.now();

  //start up the temperature sensor
  tempsensor.wake();
  // Read out the temperature
  float c = tempsensor.readTempC();
  float f = tempsensor.readTempF();

  //get new accelerometer senser event
  sensors_event_t event;
  accel.getEvent(&event);

  //write string with temperature data
  String dataStringTemp = "Temp:" + String(c, 4) + " *C";
  //write string with acceleration data
  String dataStringAccel = "Acceleration: X: " + String(event.acceleration.x) + " Y: " +
String(event.acceleration.y) + " Z: " + String(event.acceleration.z) + " m/s^2";

  //store temperature data string onto file on SD card
  File dataFile = SD.open("datalog.txt", FILE_WRITE);

  // if the file is available, write to it:
  if (dataFile) {
   dataFile.print(now.year(), DEC);
   dataFile.print('/');
   dataFile.print(now.month(), DEC);
   dataFile.print('/');
   dataFile.print(now.day(), DEC);
   dataFile.print(" (");
   dataFile.print(daysOfTheWeek[now.dayOfTheWeek()]);
   dataFile.print(") ");
```

```
   dataFile.print(now.hour(), DEC);
   dataFile.print(':');
   dataFile.print(now.minute(), DEC);
   dataFile.print(':');
   dataFile.print(now.second(), DEC);
   dataFile.print(",  ");
   dataFile.print(dataStringTemp);
   dataFile.print(",  ");
   dataFile.println(dataStringAccel);
   dataFile.close();
  }
  // if the file isn't open, pop up an error:
  else {
   dataFile.print("datalog.txt not open");
 }
  delay(450);
  tempsensor.shutdown_wake(1); // shutdown MSP9808 - power consumption ~0.1 mikro Ampere,
stops temperature sampling
  delay(50);
}
```