DIGIDOUGH

By

Abhitya Krishnaraj

Alec Thompson

Jake Hayes

Final Report for ECE 445, Senior Design, Spring 2024

TA: Tianxiang Zheng

1 May 2024

Project No. 3

Abstract

The primary purpose of this project is to create a device to assist a user when making bread dough or sourdough starter.

DigiDough is a device that monitors the temperature and height of rising dough to help the user make sure the dough is in a good growing environment and decide when the dough has risen enough. Knowing how much dough has risen is an essential part of making dough that leads to airy and tasty bread as spending either too little or too much time rising can lead to unsatisfactory bread.

The device uses sensors to measure height and temperature and displays the start, current, and peak height of the dough along with the current temperature in the container on an E-Ink display. The user simply attaches the device to the topside and underside of a container lid of their choice, sets the proper heights for the container and the dough, and watches the display until the desired height is reached.

Contents

1. Introduction

We will be discussing the process of creating DigiDough, a device that allows a user to understand what is happening to the dough inside of a container through displaying sensor information on the display. This is useful as the dough proofing process is one of the most important parts of making bread from scratch and it is difficult to get right on the first try when you do not know what is occurring inside a container.

1.1 Problem

Making bread at home, especially sourdough, has become very popular because it is an affordable way to get fresh-baked bread that's free of preservatives and other ingredients that many people are not comfortable with. Sourdough also has other health benefits such as a lower glycemic index and greater bioavailability of nutrients.

However, the bulk fermentation process (letting the dough rise) can be tricky and requires a lot of attention, which leads to many people giving up on making sourdough. Ideally, the dough should be kept at around 80 degrees F, which is warmer than most people keep in their homes, so many people try to find a warm place in their home such as in an oven with a light on, but it's hard to know if the dough is kept at a good temperature. Other steps need to be taken when the dough has risen enough, but rise time varies greatly, so you can't just set a timer, and if you wait too long the dough can start to shrink again. In the case of activating dehydrated sourdough starter, this rise, and fall is normal and must happen several times; and its peak volume is what tells you when it's ready to use.

1.2 Solution

Our solution is to design a device with a distance sensor and temperature sensor that can be attached to the top and underside of most types of lids, using magnets. The sensors will be controlled with a microcontroller; and an e-ink display will show the minimum, current, and maximum heights of the dough along with the temperature. This way the user can see immediately how much the dough has risen, whether it has already peaked and started to shrink, and whether the temperature is acceptable or not. There is no need to remove it from its warm place and uncover it, introducing cold air; and there is no need to puncture it to measure its height or use some other awkward method.

A typical use case will proceed as follows:

1. The device is attached to a container's lid with the sensor enclosure on the underside of the lid and with the display and buttons on the topside, and the lid is placed on the container.
2. The button to set container height is pressed. The device measures the distance to the bottom of the container and temporarily displays the result so the user can confirm.
3. A ball of dough that needs to rise is placed in the container, and the button to set dough height is pressed. The device records and displays the current height as both starting height and maximum height.
4. The container is placed in a warm location so the dough can rise. The device periodically updates the temperature, current height, and peak height on the display. The starting height continues to be displayed unchanged.
5. When the dough reaches the desired height, or when the user sees that the current height is lower than the peak height, then the dough is removed.

A similar process can be followed for sourdough starter, except that the starter will peak and fall after every feeding, and the user can optionally press the set dough height button each time. The user will decide when the starter is ready based on the peak height rather than the current height. The device will not attempt to tell the user when the dough/starter is ready because that is a judgment call that depends on the type of bread, the recipe, and the baker's personal preference. Instead, the device makes it easy for the user to see the information they need to make their decision.

1.3 High-Level Requirements

- Supply the necessary power to each subsystem using a rechargeable battery that lasts for at least 10 hours between charges.
- Accurately read and store distance values (with an accuracy of ±1 cm below 10 cm and ±10% between 10-20 cm) and temperature values (within 3°C).
- Display the minimum height, maximum height, current height, and current temperature values on a display, updated at least once every five minutes.

1.4 Block Diagram

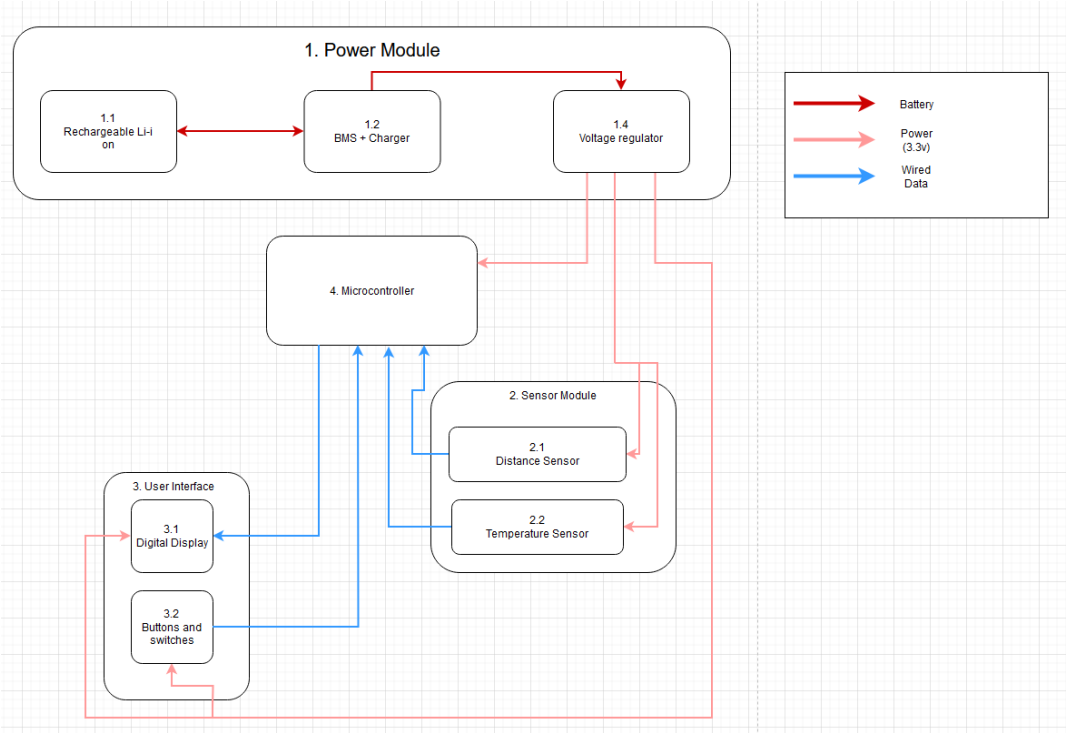

Figure 1: Block Diagram of Subsystems and Modules within Device

1.5 Final Design

Our final design consists of the main PCB covered by a 3-d printed enclosure which also holds up the display, the sensor board is enclosed by a fitted piece holding the board in place while the sensors are facing downward, and magnets on either end of the 3-d printed pieces to connect the enclosures through a container lid. The main enclosure is a two piece 3-d printed box that contains the main PCB, battery, recharge port, and has openings for the wired connections to the sensor board and display as seen in Figures 3 and 5. The display is vertically mounted on top of the main enclosure and is surrounded by the 4 buttons for setting measurements and the power switch for the device as seen in Figure 2. The sensor piece has exposed areas for the sensors to record measurements in the containers and for the wires to escape and connect to the main board as seen in Figure 6. Figure 4 demonstrates the measurements and design utilized to 3-d print the enclosures for the main and sensor PCBs.

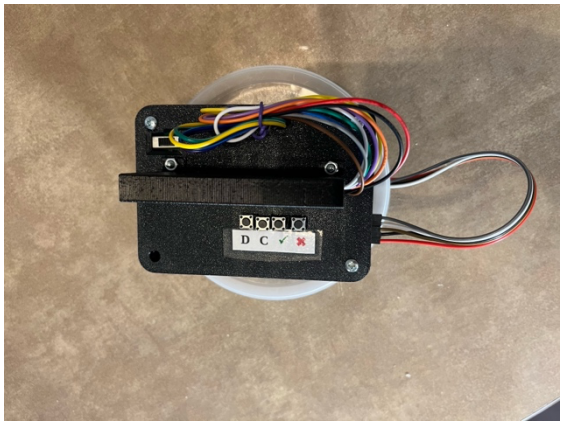

Figure 2: Top View of Device

Figure 3: Front View of Device

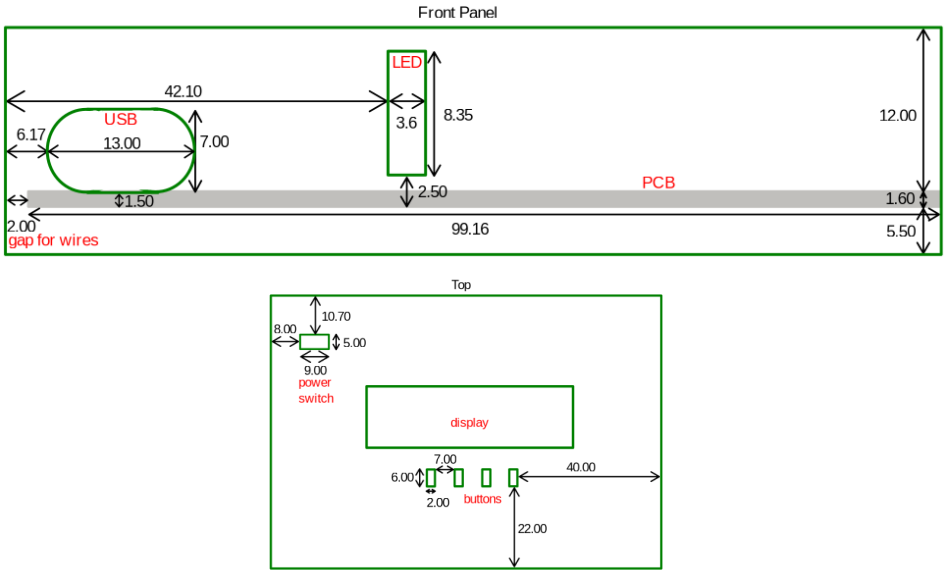

Figure 4: Enclosure Design Diagrams

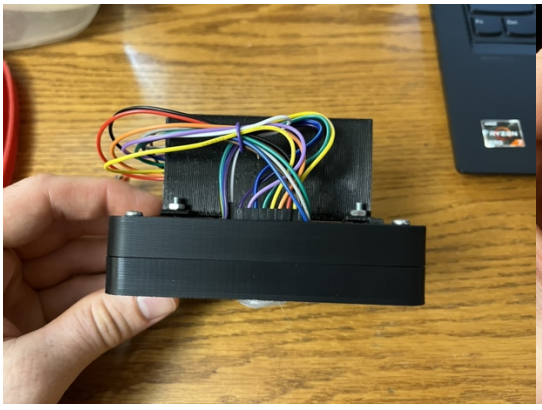

Figure 5: Back View of Device

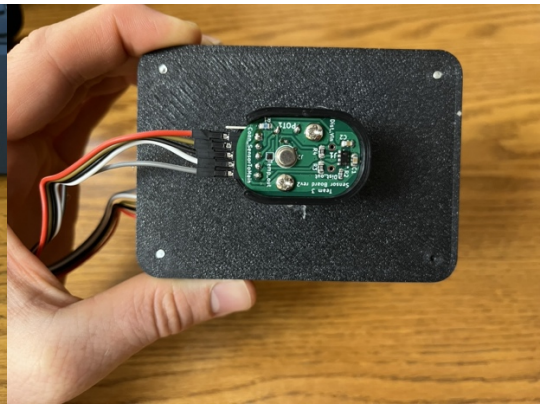

Figure 6: Bottom View of Device

2 Design Procedure and Details

Throughout our project our general design outlook remained the same, and the initial idea we proposed was maintained with the same subsystems. The main changes were in the selection of individual components. The final temperature and distance sensors were both different from the initial proposal. In most cases, the changes were made due to availability or price differences. In a few cases, modifications were made as we conducted more research and found components more suitable for this project. On the software side, we anticipated using STM32CubeIDE for everything before finding out our display requires Arduino IDE libraries and running into some serial output and IDE issues (caused by incorrect pin mappings). We bounced around between STM32CubeIDE, Arduino IDE, and PlatformIO before settling on PlatformIO. Many of the challenges we had were a result of trying to work with a variety of complex libraries that were necessary for the project with non-standard pin mappings. However, we worked through these challenges, and the project worked well once we were able to program our device correctly.

2.1 Power Subsystem

The power subsystem is responsible for providing power to all other subsystems, and its main components are shown in Figure 7. The primary power source is a 3.7 V rechargeable lithium-ion polymer battery that can be charged via a charging module with a USB-C charging port. The voltage was converted to a stable 3.3 V by a linear regulator. A power-select header allowed for an external power source to be used and for the regulator to be optionally bypassed during development by jumping the appropriate pins.

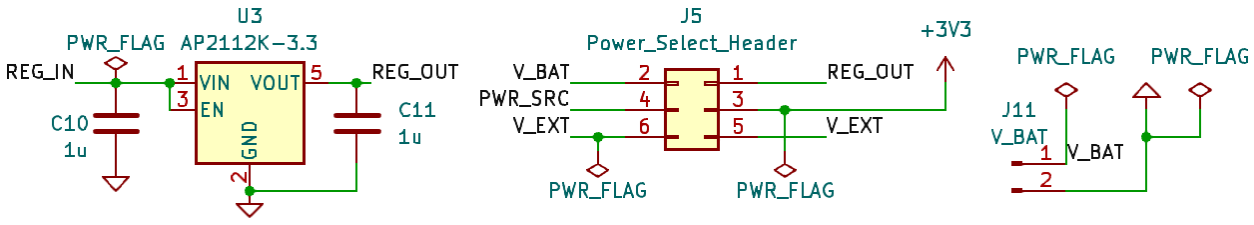

Figure 7: Schematic for Power System

2.1.1 Battery and Charger

We chose a small 420 mAh rechargeable Lithium-Ion battery (about the size of a quarter) since our design does not have high power consumption and we wanted the device to be relatively

small. The battery charging module includes a USB-C charging port, a red LED to indicate charging, and a green LED to indicate full charge.

2.1.2 Linear Regulator

All components of the device were chosen to be able to operate at 3.3 V, so one 3.3 V linear regulator was sufficient to control the voltage for the entire device. For this regulator, we chose the Diodes Incorporated AP2112K-3.3TRG1 due to its low cost, low power consumption, and low dropout voltage (5-125 mV when delivering between 10-300 mA). According to datasheet recommendations [9], 1 µF decoupling capacitors were placed in parallel with the regulator at both its input and output to filter high-frequency noise and provide a more stable voltage.

Before parts were ordered, a tolerance analysis was performed to ensure that the power consumption and temperature would be within a safe operating range for the regulator. The current draw for each component when operating at full capacity was taken from their datasheets and seem om Table 1. The total current draw was calculated to be 207.94 mA, safely below the 600 mA maximum for the AP2112K-3.3TRG1.

Table 1: Maximum Current Draws for Components at 3.3V

Part	**Max Current Draw**	**Comments**
STM32F303K8	160 mA [8]	Average case is much lower than the worst case.
VCNL36826S	20 mA [4]	VCSEL Sensor
LM335AZ/NOPB	5 mA [5]	Temperature Sensor
E2266PS0C2	2.94 mA [6]	E-Ink TFT Display
CTL0603FGR1T	20 mA [7]	LED indicator
Total	207.94 mA	

Table 2 shows the relevant values from the voltage regulator datasheet which were used with the equation $T_j = i_{out}(v_{in} - v_{out})(\theta_{jc} + \theta_{ca}) + T_a$ to estimate the junction temperature of the

voltage regulator. With these values, the junction temperature T_j = 122.17 °C, which is less than the maximum of 150 °C, meaning temperature and power risks were not high for our circuit.

Table 2: Variables and Values for Tolerance Equation

Variable	Value	Comments
max (T_j)	150 °C	Max operating temperature of AP2112K-3.3TRG1
i_{out}	207.94 mA	Max current draw @ 3.3V
v_{in}	5.5 V	Max input voltage from USB charging port
v_{out}	3.3 V	Operating Voltage
θ_{jc}	95 °C/W	Junction-to-case thermal resistance of AP2112K-3.3TRG1
θ_{ca}	89 °C/W	Max Junction-to-ambient was 184 °C/W for AP2112K-3.3TRG1
T_a	38 °C	Assume around 100 °F if a warm dough environment

2.2 Sensor Subsystem

The sensor subsystem consists of a temperature sensor and a distance sensor. These sensors were soldered to a separate PCB and housed in an enclosure that attaches to the underside of the lid on which the main board is placed. This is required as both sensors need to be in contact with the environment in which the dough is rising and maintain a static position throughout the process. The temperature sensor measures the ambient temperature near the dough to ensure the dough is kept at an acceptable temperature. A proximity sensor is used to measure the height of the dough at different points in time, based on how far the sensor is from its initial reading of the bottom of the container to the sensor's proximity to the dough at the time of measurement. The schematic design shown in Figure 8 follows the recommendations in the datasheets of each sensor.

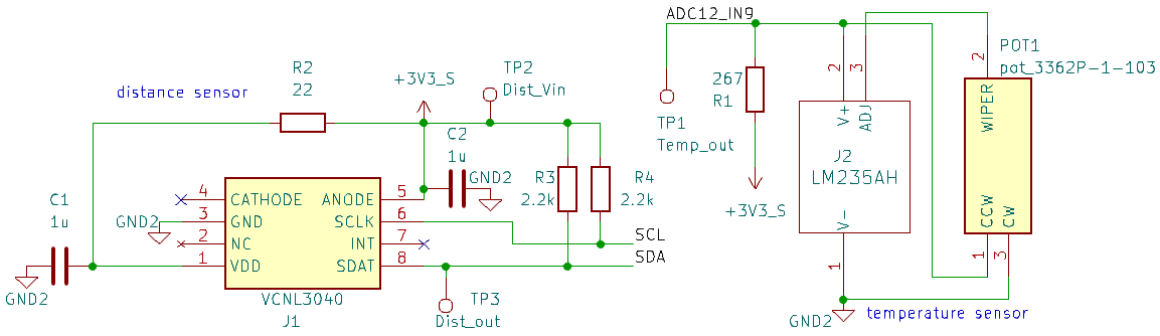

Figure 8: Schematic for Sensor System

2.2.1 Temperature Sensor

Our temperature sensor needed to be something that was cheap and easy to work with, but still had a decent range of record able temperatures and good levels of accuracy. Through our research, we discovered that the IC sensor fit all these criteria. In our initial proposal we chose the LM335AZ from Texas Instruments as it was a low-cost analog sensor that fit all our requirements for accuracy and speed. However, as we continued working on the project, we discovered that the machine shop had the LM235AH temperature sensor in stock which was the next generation of the same temperature sensor and decided to use that instead for ease of access.

This is a low-cost analog temperature sensor with measurement tolerances that work for this project since extremely accurate temperature readings are not necessary for the user to determine if they are close to their required temperature. Since our accuracy does not need to be extremely high, and operating temperatures will typically be at or above room temperature, the IC sensor is very attractive with its low cost, power consumption, and complexity. The MCU receives measurements of temperature as the voltage across the temperature sensor and based on the calibration of the sensor perform a conversion into °C.

The temperature sensor was extremely easy to work with. It sent data through an analog connection to the MCU and even in basic testing was able to output temperature measurements that just needed to be converted from voltage into Celsius to be readable by the user. The final conversion rate we used 10mV/ºK and this was done after the final calibration was completed between the temperature sensor and the sensor board.

The major challenge with the temperature sensor dealt with its calibration. The first iteration of sensor board did not include a potentiometer, which we discovered was needed to calibrate our readings.

One idea we came up with to ensure that the sensor reads the temperature outside the enclosure rather than inside was putting a pre-1981 penny in an opening in the enclosure and in contact with the temperature sensor so that the copper could conduct heat from outside to the sensor. A silicon washer was also placed around the sensor to help isolate it from the air inside the enclosure.

2.2.2 Distance Sensor

After some research we decided to use the VCNL3040 from Vishay Semiconductors, a proximity sensor (PS) that also utilizes an infrared emitting diode (IRED) to determine its values. The proximity functions are sent to the MCU through I^2C interface protocols. This sensor has the best of both worlds, a low power requirement, and a narrow beam that allows for more precise measurements directly in front of the sensor.

This distance was quite finicky to work with, mainly because the values from the sensor did not correspond to distances of a known unit, instead being based on the number of readings that increased logarithmically. Additionally, each of the 7 modes that the sensor had for the current flowing through the IRED produced different outputs and completely different mapping patterns from the PS to centimeters. Eventually we worked through some of the pin mapping and IDE issues we were having with the sensor and kept all the default configurations for the sensor, except for increasing the output resolution of the data we were reading. This led us to manually map the sensor output to centimeters as seen in Figure 9.

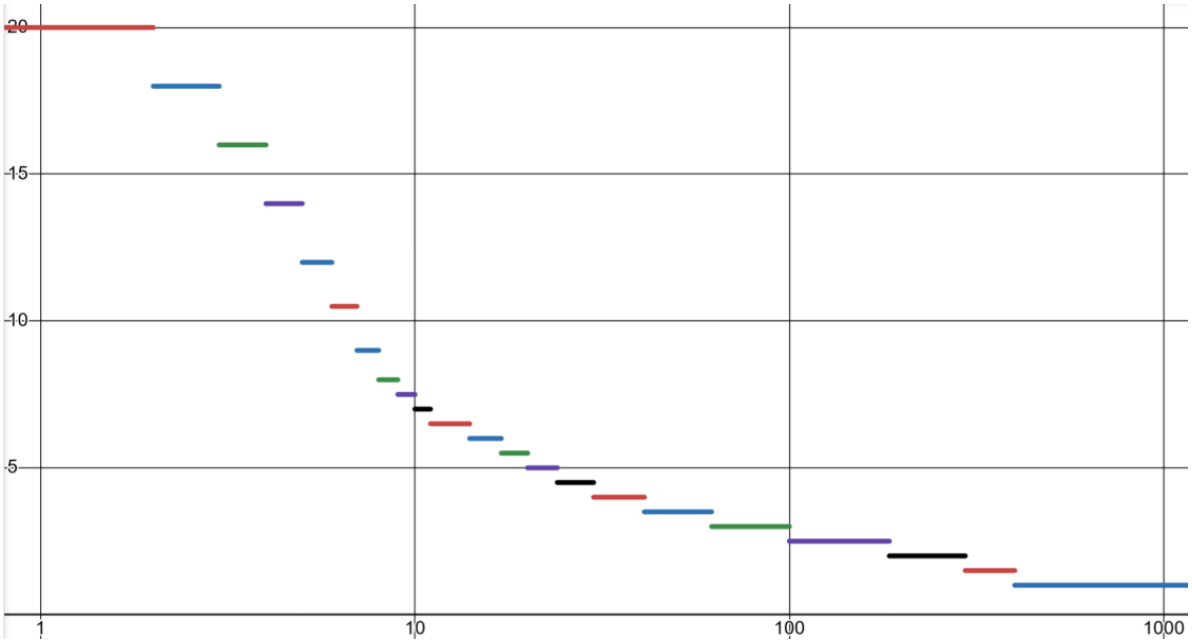

Figure 9: Mapping Output, cm vs. PS Output Data

2.3 User Interface Subsystem

The User Interface Subsystem consists of an E-Ink display for the user to see the recorded measurements, a power switch for the device, and 4 buttons to record the initial measurements that the device requires. The buttons and switch are the only things the user must interact with when using the device and both directly correlate to changes on the display. As shown in Figure 10, the power switch is connected to a resistor and capacitor for debouncing, the buttons are wired to be active-low, and the LED has a 60.4 Ω current-limiting resistor to achieve a current near the recommended 20 mA. This was calculated as $I = \frac{3.3 - 2.15V}{60.4\Omega} = 19.0mA$.

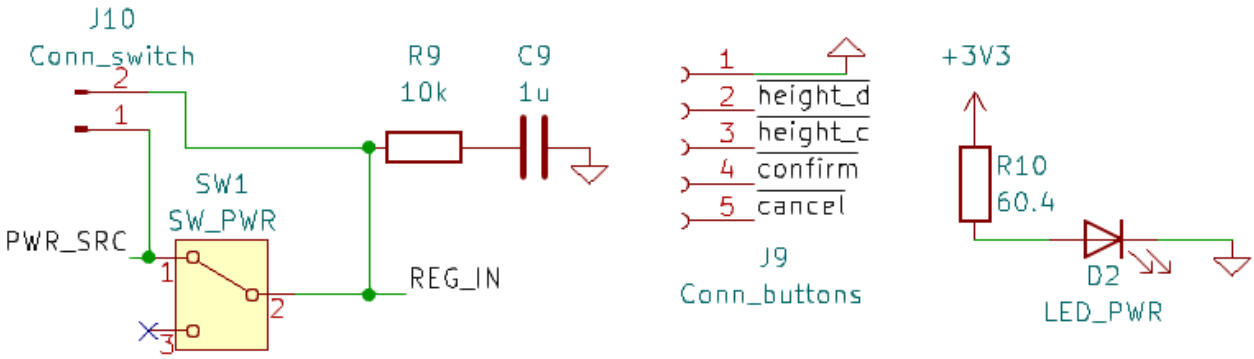

Figure 10: Schematic for User Interface System

2.3.1 E-Ink Display

We decided on an E-Ink display for several reasons. First, we thought it would be an interesting challenge and might give us more flexibility than a character display. Second, the image an e-ink display shows is crisp, even in ambient light, so no integrated light source is needed. Finally, an e-ink display does not need power to keep displaying an image. Since dough doesn't rise very quickly, this makes an e-ink display ideal for us from a power standpoint. Our initial choice was a 2.66" Pervasive Displays Monochrome display since it provided us with enough screen space to be flexible in what we output. However, since that screen was out of stock when we went to order it, we switched to the Spectra 4 (four color e-ink display) instead.

The flex cable from the display connects to an EXT3-1 development kit that connects to our main board via SPI and other GPIOs for a total of ten pins. We utilized the Arduino libraries from the manufacturer for our display to initialize it and help us display text in a nice format. The display shows the start height, current height, max height, and current temperature in its base state. We also utilize it for our startup sequence to walk the user through the initial container and dough measurements and similarly whenever they want to reset the container or dough height.

2.3.2 User Experience

The device does not have power when the switch is in the "OFF" position. At this time if the user wishes to charge the device and plugs in a live USB-C cable the "Charging" LED will turn on. When the device is unplugged and still in the "OFF" position, the user will be able to power on the device by moving the switch to the "ON" position, which also lights up the "ON" LED. Once powered on, the device enters a startup state that is waiting on user input for the container height and the starting dough height. After placing the device around a lid and placing it on a container, the user will be prompted to set the container height, once the user presses two buttons, "Set Container Height" and then "Confirm", the container height is successfully recorded, and the device moves onto the dough height setting. At this screen the user should press "Set Dough Height" and then "Confirm" to record that measurement. Once these steps have been taken the device enters its standard state and displays all the recorded information. During the height setting process, if the user accidentally presses a set button when not meaning to, they can press "Cancel" to go back to the prior state and restart the height setting process.

2.4 Microcontroller Subsystem

The microcontroller (MCU) subsystem interacts with the other subsystems at least once every five minutes. Both sensors are connected to the MCU's PCB by wires, and the MCU performs the computations for both, by reading and converting the voltage at the output of the temperature sensor and reading and mapping the PS data sent by the distance sensor. The MCU take 10 readings within 10 seconds for both sensors, calculates the average, stores the data, and sends it to the display using SPI protocol with a wired connection and a development kit for the E-ink display. The MCU also takes input from the buttons and switches in the user interface subsystem, which is essential on the startup of the device to get the correct initial readings and make sure the output on the display is accurate.

The state machine in Figure 11 shows how our buttons interfacing with the MCU works. The modification connections show that the stored values are changed at those points and that user input has no impact at those states. The base state of the MCU occurs after all the initial heights are set and consists of the display showing the: peak height, start height, max height, and current temperature.

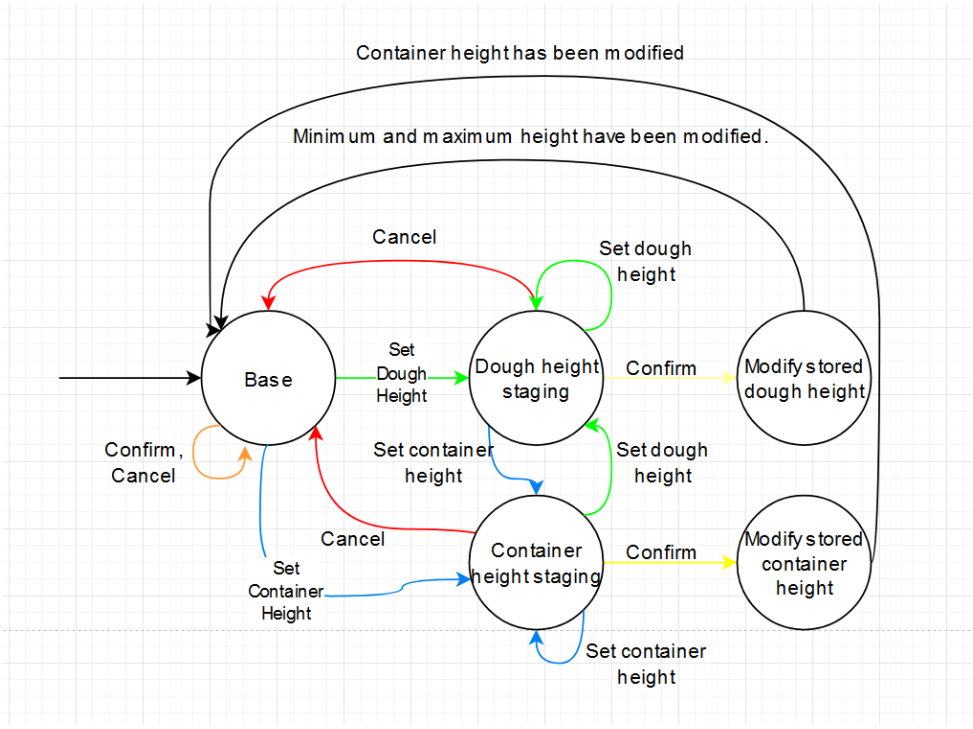

Figure 11: State Diagram of MCU

3. Design Verification

The design verification section details how each subsystem was individually tested and the success or failure of certain requirements that were established. The full requirements and verification table can be found in Appendix A.

Apart from the individual subsystem testing, we also performed tests to ensure that the whole device worked as intended and that each subsystem was able to work with each other without any errors occurring. This complete test was run many times in different conditions, including type of lid, container, and temperature to ensure that the device would be in complete working condition and any situation that the user might want to use it in.

3.1 Power Subsystem

The two major testing requirements for the power subsystem were testing the lifespan of the lithium-ion battery and ensuring the other subsystems received the correct voltage or current after passing through the voltage regulator.

3.1.1 Battery Life and Charging

Our first high-level requirement included a battery life of at least 10 hours. To test this, we left the device running under normal operating conditions and then checked it after 12 hours. It was still on, and when we pressed the buttons to measure a container and measure dough, it correctly took the measurements and updated the display.

3.1.2 Power Supply to Subsystems

Perhaps the most important part of our first high-level requirement was to provide the necessary power to each subsystem. The acceptable operating ranges for voltage or current were found in the datasheets of each component. With the device powered on, the voltage was measured at the output of the battery, the input to the MCU, the input to the display, and the input to the distance sensor. All were within the acceptable ranges. For the temperature sensor and LED, providing the right amount of current was more important. The current through the temperature sensor was tested by measuring the voltage across a resistor in series with it and calculating the current as I = V/R. The current through the LED was measured before soldering it to the PCB by placing it in a breadboard and connecting it to the PCB in series with an ammeter. Both currents were very

close to the ideal ones specified in the datasheets. Table 4 shows the datasheet values compared to the tested values.

Table 3: Voltage Readings at Different Components

	Datasheet Range	Tested Value
Battery	3.0 – 4.2 V	4.13 V
MCU	2.4 – 3.6 V	3.30 V
Display	2.4 – 3.6 V	3.30 V
Distance Sensor	2.5 – 3.6 V	3.30 V
Temp. Sensor	0.4 – 5.0 mA	1.4 mA
LED	20 mA ideal	18.9 mA

3.2 Sensor Subsystem

The next high-level requirement revolved around the sensor subsystem. The distance and temperature sensors both needed to have accuracies within given ranges for their measurements and have consistent outputs while the environment was static.

3.2.1 Temperature Sensor

The temperature sensor was quite easy to test, we connected the device to a laptop and placed the sensor board in an area where the temperature was known. In our requirements we gave ourselves a range of ±3 °C (~±5 °F using 9/5 conversion). As seen in Figure 12 the temperature in the testing area was 79 °F and in Figure 13 we can see that the temperature's terminal output readings are all well within the allowed range of 84 °F and 74 °F, verifying the accuracy of the temperature sensor.

Figure 12: Thermometer Temperature Recording

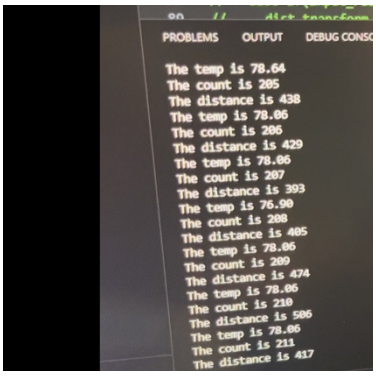

Figure 13: Terminal Temperature Output from Sensor

3.2.2 Distance Sensor

The measurements from the distance sensor also needed to be within a certain range of the actual distance, at below 10 cm, ±1 cm and from 10-20 cm ±10%. This was conducted by placing the distance sensor in a set location and removing objects with a known width from in front of the sensor, this allowed us to test exact distances from 1.5 cm to 19.5 cm and measure the output of the distance sensor vs the measured distance as seen in Figure 14.

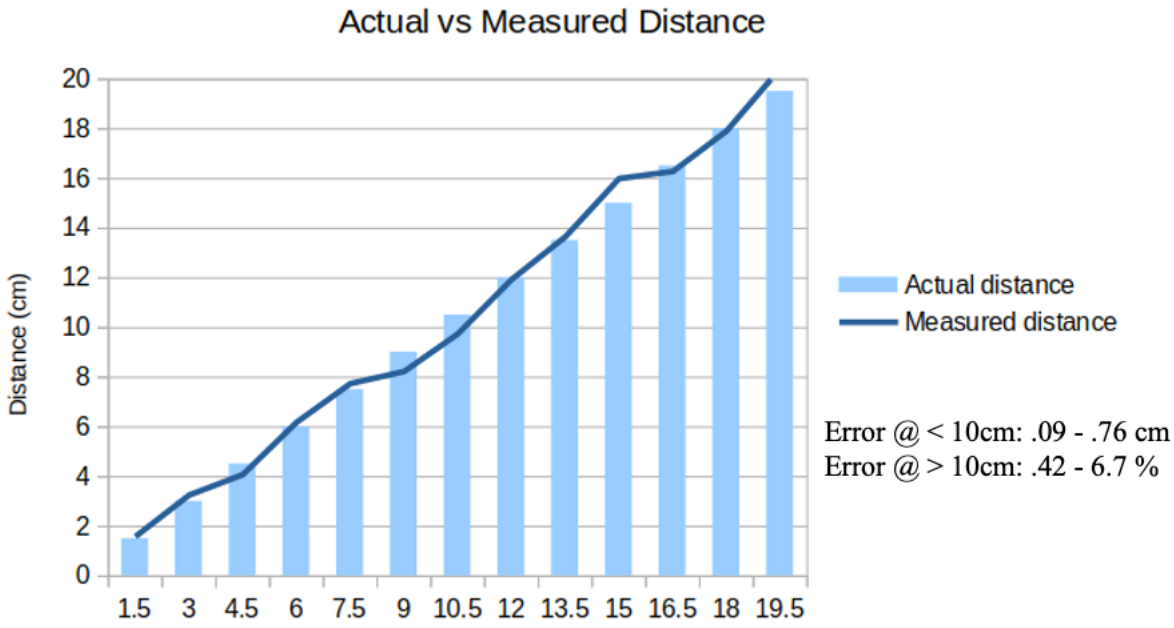

Figure 14: Actual Distance vs Distance Sensor Measurement

3.3 User Interface Subsystem

The primary testing for the UI system took place on a Nucleo board initially to verify that our code worked as expected and the user experience was acceptable. This made working through IDE issues later since we had verified the code, and the buttons were responding the way we expected. Because many of our verifications for the UI subsystem were simply about it functioning the way we had it at the demo and the way a user would expect it to, we did not need much quantitative data for these verifications.

3.3.1 E-Ink Display

Our main display verification was to display the start height, current height, max height, and current temperature. Our final demo did this, and we passed these verifications.

3.3.2 User Experience

The UX verifications were more focused around code structure and the device behaving the way the user would expect. For example, each of our buttons does what it was designed to do, and pressing cancel before pressing confirm does not take us into a weird state in our state machine. Since our buttons and device worked as expected (along with printing out states to the serial console to verify our state machine worked), these verifications were met.

3.4 Microcontroller Subsystem

Testing for the microcontroller system involves ensuring that the computations that the MCU does for the various dough heights are correct and the values are properly stored and maintained by outputting the values and current state to the serial console. The MCU can also be tested along with the display by ensuring that the display is updated at least every 5 minutes.

4. Costs

Our calculated total costs were $101,304.15 with $54.15 coming from the cost of parts (Appendix B, Table 6) required for the device and $101,250 from our theoretical labor cost.

The average starting salary of ECE grads can be found on the ECE Illinois website [3] showing that the average starting salary of a member of our group is about $102k (two Computer Engineers and one electrical engineer). Assuming a regular work week and two weeks of paid vacation per year, that total comes out to about $50 an hour. We had around 30 hours per team member per week over 9 weeks of work with the 2.5 overhead cost multiplier, bringing our cost per team member to $\$50/_{hr} * 30\ hours * 9\ weeks * 2.5$ = $33,750 for each team member, and $101,250 in total labor costs.

5. Conclusion

5.1 Accomplishments

This project was overall a huge success as the dough monitor works as intended in all the tested environments, every major and minor requirement we established was successfully verified, and every problem that arose throughout the project was swiftly dealt with and overcome.

5.2 Ethical considerations

The major safety concern with this device is food contamination as we have created a device that works with bread dough and needs to be sanitary and uncontaminated throughout the whole process [1]. The 3-d printed materials we utilized throughout the exterior of the product do not have any toxic materials and have no possibility of contamination. Unfortunately, due to the design of the sensor board and its enclosure it is impossible for the product to withstand a general cleaning and sanitation process and may come into contact with the dough. The sensor board is the only possible part that can come in contact with the dough, so this product likely can't be used completely risk free. There are also some electrical safety concerns regarding the wiring between the top and bottom halves of the solution which will need to withstand continual movement and pressures and could also meet other contaminants. The battery and live electrical components used within the device are not exposed to them, and the device is safe for long-term use and operation, as components will not pose a risk to the user.

Unfortunately, the product doesn't meet all the major standards required for electronic products to utilize with food but is generally safe otherwise.

5.3 Future work

The major next step for this project is developing it on a board with Wi-Fi/Bluetooth capabilities and integrating the device with a phone app. This could be achieved with an ESP32 or a Raspberry Pi Pico. Other improvements could also include using a different distance sensor and display as they were the two most troublesome parts to work with due to software restrictions for the display and general reading and testing difficulties with the distance sensor.

References

[1] B. Kelechava, "NSF/ANSI 2-2022: Food Handling Equipment - ANSI Blog," *The ANSI Blog*, Aug. 16, 2023. https://blog.ansi.org/nsf-ansi-2-2022-food-equipment-standard/ (accessed Feb. 22, 2024).

[2] IEEE, "IEEE Code of Ethics," *ieee.org*, Jun. 2020. https://www.ieee.org/about/corporate/governance/p7-8.html (accessed Feb. 22, 2024).

[3] "Salary Averages," *ece.illinois.edu*. https://ece.illinois.edu/admissions/why-ece/salary-averages (accessed Feb. 22, 2024).

[4] "Proximity Sensor With Interrupt, VCSEL, and I 2 C Interface," Vishay Semiconductors, Feb. 2022. Accessed: Feb. 22, 2024. [Online]. Available: https://www.vishay.com/docs/84964/vcnl36826s.pdf

[5] "LM135, LM135A, LM235, LM235A, LM335, LM335A LMx35, LMx35A Precision Temperature Sensors 1 Features 3 Description Basic Temperature Sensor Simplified Schematic Calibrated Sensor," Texas Instruments, Feb. 2015. Accessed: Feb. 22, 2024. [Online]. Available: https://www.ti.com/lit/ds/symlink/lm335.pdf?ts=1708655741439&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM335%252Fpart-details%252FLM335AZ%252FNOPB

[6] "E2266PS0C1 E2266PS0C2 Product Specifications," Pervasive Displays, Aug. 2022. Accessed: Feb. 22, 2024. [Online]. Available: https://www.mouser.com/datasheet/2/1089/Pervasive_displays_8_24_2022_1P332_00_01_E2266PS0C-3009734.pdf

[7] "CTL0603FGR1T DATASHEET," Venkel LTD. Accessed: Feb. 22, 2024. [Online]. Available: https://page.venkel.com/hubfs/Resources/Datasheets/LEDs/CTL0603FGR1T.pdf

[8] "STM32F303x6/x8," STMicroelectronics, Jul. 2018. Accessed: Feb. 22, 2024. [Online]. Available: STM32F303x6/x8

[9] "600mA CMOS LDO Regulator with Enable," DIODES Incorporated, Jun. 2017. Accessed: Feb. 22, 2024. [Online]. Available: https://www.diodes.com/assets/Datasheets/AP2112.pdf

Appendix A: Requirement and Verification Table

Table 4: System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
The battery must last for at least 10 hours.	Leave the device operating as in normal usage (taking measurements and refreshing the display every 5 minutes). Measure the output of the voltage regulator once per hour for 10 hours. Verify that the voltage stays at or above 3.0 V, even during measurements and display refresh.	Y
Recharge the battery through a 5V USB-C port that is exposed to the user in less than six hours.	• The USB-C port must be exposed enough and, in a location, where one can plug a USB-C cable into the port. • The CHG LED on the BMS + Charging module needs to light up to indicate the battery is being charged when connected to power (if it is not fully charged). • Take a discharged battery (3.1V±.1V) and connect it to the charging module. Power the USB-C charging module. If the CHG light goes off and the DONE light comes on within six hours, check the battery voltage with a multimeter and verify it is 4.1V ± .1V. If the CHG LED stays on for the full six hours, check the battery voltage at six hours and verify it is 4.1V ± .1V.	Y
Measure the ambient temperature inside a container with an accuracy of ±3°C	• Place our device in a container along with a known accurate thermometer. The temperature from the board's microcontroller via serial debugging must be ±3°C of the temperature from the thermometer.	Y

and transmit the data to the MCU at a rate of at least 1 reading/second.	• Output the time between readings after each reading to the serial debugging terminal and confirm that the MCU is getting a temperature value at least once a second	
Measure the distance from the sensor to the bottom of an empty container and the distance to an object inside the container with an accuracy of ±1 cm at distances below 10 cm and ±10% at distances from 10-20 cm and transmit the data to the MCU at a rate of at least 1 reading/second.	• Measure the height of a container with a ruler. Attach the sensor to the underside of a lid placed on the container and read the distance from the board's microcontroller via serial debugging. Repeat the above with an object inside the container, after measuring the object's height with a ruler. The distance read from the sensor must be ±1 cm of the ground-truthed value at distances below 10 cm and ±10% at distances from 10-20 cm • Output the time between readings after each reading to the serial debugging terminal and confirm that the MCU is getting a distance value at least once a second	Y
Taking the lid off the container and putting it back on does not significantly change the distance value if it should be the same (nothing added to the inside of the container).	• The readings from the serial debugging terminal after replacing the lid must be ±.5 cm of the initial value if the initial value was under 10 cm total or ±5% if the initial value was between 10-20 cm.	Y
Turn the device on and off with a switch on top of the device.	• Use a voltmeter to measure a 3.3V ± .3V. output from the voltage regulator whenever the switch is moved from the off to the on position	Y

	• Verify that the display has blank outputs for the readings or prompts the user to set the height values when the switch is moved from off to on • When the switch is moved from on to off, use a voltmeter to measure 0V ± .1V from the voltage regulator	
Measure or reset the starting height of the dough with a button.	• When the device is on, pressing the "Set dough height" button and then confirming with "Confirm" should reset the minimum and maximum dough heights shown on the display • Verify the minimum height does not change when the maximum height is updated via the sensor data unless it is reset or the device is power cycled	Y
Measure or reset the container height with a button.	• When the device is on, pressing the "Set container height" button and then confirming with "Confirm" should reset the minimum and maximum dough heights output to the serial console • Verify this value displayed is not modified unless it is reset or the device is power cycled	Y
Temporarily display the container height when measured, just for confirmation.	• When the "Set container height" button is pressed, display current height measured and the current stored container height, even before the "Confirm" is pressed • Pressing confirm or cancel needs to go back to the normal screen displays all of the dough statistics	Y
Display the minimum height, maximum height, current height, and current temperature	• Use a serial console to verify the time between updates sent to the display is at least once every five minutes • The maximum height should be taken as the maximum reading from the distance sensor since	Y

values on a display, updated at least once every five minutes.	the last reset or power cycle. Verify there is always a value shown on the display when the display is showing dough statistics • The minimum height will need to be set if it hasn't been since the last power cycle. If it has been set, verify there is always a value shown on the display when the display is showing dough statistics. If it hasn't been set yet, the display should indicate that to the user in its value field • Verify there is always a current height value shown on the display when the display is showing dough statistics. • Verify there is always a temperature value shown on the display when the display is showing dough statistics.	
Pressing the "Cancel" button should not update, even with a subsequent push of the "Confirm" button	• Ensure the current dough height is at least a 1 cm difference from the minimum height. Press the "Set dough height" button. Press the "Cancel" button. Press the "Confirm" button. Verify the value on the display for minimum dough height has not changed.	Y
Store the recorded values from the sensor subsystem for the current "rise", specifically distances (container, dough starting, dough peak, and dough current) and the temperature.	• Print out ten values every ten seconds for each sensor to the serial console during testing	Y

Compute the average of several consecutive readings to reduce the effect of noise.	• Print ten values taken in a ten-second span for each sensor. Take the average of the ten values for both sensors. Print out the stored temperature value that is used to update the display. Verify the average of these ten values is the value used to update the display for each sensor (within ±0.01 for rounding differences)	Y
Use the recorded distance values to calculate dough height (container floor distance minus dough distance).	• Ensure values exist already for dough height and container height by printing those values to the serial console. Verify the {container distance} - {dough distance} = {dough height stored for display}	Y
Update the values on the display at least once every five minutes or within 15 seconds of the user pressing a measure/reset button	• Ensure the current distance is substantially different than the minimum (at least 1cm difference). Press the "Set dough height" button and then the "Confirm" button. The time it takes for the value on the display to update needs to be less than 15 seconds. • When no user interaction has occurred with the buttons or the switches, time the interval between the display updates and print this value to the serial console once the display is updated, ensuring it is less than 300 seconds (5 minutes).	Y

Appendix B: Cost of Parts

Table 5: Cost of Parts

Part	Manufacturer	Quantity	Actual Cost ($)
STM32 MCU	STMicroelectronics	1	$6.29
Analog Temperature Sensor	Texas Instruments	1	$1.40

I2C IR Distance Sensor	Vishay Semiconductors	1	$2.07
On-off switch	E-Switch	1	$0.67
LED indicator	Venkel	2	$0.04
Tactile Switch Buttons	CUI Devices	4	$0.40
Linear Voltage Regulator	Diodes Incorporated	1	$0.35
420 mAh Li-Ion Battery	Adafruit	1	$6.95
Battery charger	Adafruit	1	$5.95
External power connector	TE Connectivity AMP Connectors	1	$0.14
Power multiplexer	GLF Integrated Power	1	$0.48
Schottky diode	Toshiba	2	$0.64
TVS diode	Diodes Incorporated	1	$0.36
2.66” EPD (display)	Pervasive Displays	1	$7.18
EPD dev kit	Pervasive Displays	1	$16.00
1x10 EPD header	Adam Tech	1	$0.17
1x6 UART header	Sullins Connector Solutions	1	$0.52
2x5 debug header	On Share Technology	1	$0.33
1x3 header	Adam Tech	4	$0.40
1x2 jumper	Sullins Connector Solutions	4	$0.40
Neodymium magnets	MAGXCENE	2	$2.00
Capacitor 10n	Samsung Electro-Mechanics	1	$0.10
Capacitor 100n	Samsung Electro-Mechanics	6	$0.50
Capacitor 1µ	Samsung Electro-Mechanics	6	$0.20
Capacitor 4.7µ	Samsung Electro-Mechanics	1	$0.10
Resistor 22	Cal-Chip Electronics	1	$0.002
Resistor 60.4	Cal-Chip Electronics	2	$0.006
Resistor 270	Cal-Chip Electronics	1	$0.002
Resistor 2.2k	Cal-Chip Electronics	2	$0.004
Resistor 10k	Cal-Chip Electronics	5	$0.008
Resistor 100k	Cal-Chip Electronics	3	$0.006
Total			**$54.15**