ECE 2799 Final Report

THE SAFE SHOWER

Ali Akhtar: Box 6 , Patrick Bodreau : Box 38 , Nathan Wells : Box 296

Team 8

Table of Contents

List of Figures

List of Tables

1 Introduction

This document details the design for The Safe Shower, a shower based water temperature sensor. The Safe Shower contains a digital temperature sensor, a 7-segment display, and an alarm mechanism, which are all powered by a 9V power supply consisting of 6 AA batteries. Our product would be marketed towards parents of small children, but we feel it may have a broader appeal.

We made significant improvements over other products on the market, such as adding a moveable display, Celsius and Fahrenheit modes, and an alarm. Based on these improvements, we feel that our product would be very competitive. Our materials for each product will cost a little under $15, allowing us to make a significant product while remaining at a price close to our competitors.

1.1 Problem Statement:

The accepted safe temperature for bathing is considered to be about 100°F or 37°C. Unfortunately, in residential housing, showers have to be set to have a max temperature of about 112°F in order to pass inspection. This might not seem too much hotter than 100°F, but at 120°F it only takes about 5 minutes of constant contact with the water to develop a third degree burn. Maybe the homeowner bumps up the temperature of their water heater a few degrees because they feel they run out of hot water too quick. This could cause the temperature of the shower water to be way too hot for safe use.

The Safe Shower is a product that will help to alert the user if the shower is too hot for safe use. It will monitor the temperature of the water, and once the temperature reaches a certain value, the display will turn on to show what that temperature actually is. If the temperature were to reach 120°F then a couple LEDs and a buzzer will turn on to alert the user that the water is much too hot for them to safely bathe for an extended period of time. Because young children are especially susceptible to injury, the focus for this product will be parents with young children.

1.2 Market Research:

To conduct market research we first decided on the ways we should do our market research and came up with the following few methods:

- Research on the world wide web using different search engines
- Research on households in the U.S with infants (Children under the age of 5)
- Interview people in the specific industry
- Conduct an online survey from a group of people

The research conducted on the Internet gave us some valuable information regarding the types of products out in the market which resemble our product. We also got some valuable customer review information for these products from different websites like Amazon and eBay. The last part of the online research was to look for the patents, which have been submitted/approved on the U.S government patent site relating to our product.

Research regarding the average household in the US revealed that the average family size is 2.6 people. There are 313,914,040 people in the US as of 2012, and according to the census that was done the same year, 6.4% of those people are children under the age of 5. That corresponds to about 20,090,498 people under 5 years old, which is a pretty good-sized market for this product.

For interviews, we interviewed James Boudreau, a plumber who has been in the business for quite a while. We asked him some questions relating to the product we are aiming to design for this class and asked for his suggestions:

- Is there anything in the market that is similar to the product I described to you?
    - There are some showerheads with similar technology, and there are also valves that have incorporated a temperature sensor/display.
- What companies should we look into for similar products?
    - Honeywell and Johnson Controls are both bigger companies that create controls for different systems.
- Any insight as to how it could be implemented into the shower?

    - Attach the sensor somewhere between the shower head and shower arm.
- What do you know about scalding/what temperature is that?
    - Third degree burns can begin to develop around 120°F, which is why the max temperature of showers are set to about 112°F.

The last method of conducting research was creating a survey and sending it out to people to answer. Unfortunately we sent it to college students as opposed to people that have, or have had children. Initially, our intent was to market our product to college students, and people of a similar age group due to some extra features we thought we were going to be adding to the final product. We still got a good response from college students regarding whether or not they liked the idea of being able to monitor the temperature of the water and be alerted if it got too hot for safe use, so we assumed parents with children would also like the idea.

2 Product Requirements

- Readable display
- Celsius and Fahrenheit display modes
- Water resistant
- Power efficient
- Affordable
- Easy to install
- Easy to use

The final product must have a readable display and be able to switch between Celsius and Fahrenheit. These are a couple requirements that were determined through the market research that was done. Having a readable display is important, because if the user can't see what temperature the water is, defeats the entire purpose of the product. Being able to switch the display between Celsius and Fahrenheit was a common complaint seen of other similar products. This is important because some people prefer to measure temperature in Celsius while others do so in Fahrenheit.

Water resistance is a key requirement due to environment in which The Safe Shower will be used. If it is not water resistant, then the components inside the device could, and probably would get wet causing a system failure and damage to the product. It will also have to be power efficient, because it is something that cannot be plugged into an outlet for safety reasons. Therefore, the product will have to run on batteries, or something of the sort. Another requirement is that it is affordable. This is an implicit requirement, because nobody wants to overpay for anything. They want a good product for as cheap as possible.

The last two product requirements are that it is easy to install, and easy to use. The plan for The Safe Shower is to have it installed between the showerhead and the shower arm, so the customer would have to take apart the shower head to install it. We need to make sure that this is the only thing the customer needs to do in order to use the product. We also want it to be easy to use.

2.1 Customer Requirements:

- Good quality i.e. long lasting – worth the money
- Inexpensive
- Good temperature display
- Temperature display in both Celsius and Fahrenheit
- Easy to install
- Waterproof
- Safe

Good quality and inexpensive kind of go hand in hand in their influence upon the product requirements. The better quality the components are, the more expensive the final product will be, and the lesser the quality, the cheaper it will be. There is a compromise that will have to be made to insure that it is a good product, while still maintaining at a reasonable price.

The customers required that the display be of a good quality. This is to insure that they are able to easily read the temperature of the water without struggling. They also required that the display is able to switch between Celsius and Fahrenheit. This just means that there will have to be some extra line of code in the programming to switch the display between the temperature values.

Other requirements that the customer had were that the product is waterproof, easy to install, and safe. The waterproofing, or being water resistant, is for the same reasons stated earlier. The components need to be protected from the water. Ease of installation is important, because they want to set it up in no time and be able to use it right away. Finally, it has to be safe. They don't want to get electrocuted when they use our product.

2.2 Competitive Value Analysis:

Customer Criteria

In order to make the best product possible, the customer requirements need to be taken into account. If all requirements are met, then it is very likely that the product will be purchased, thus confirming success of the product. One of the most important criterion provided in the market research was that the product cost somewhere between ten and forty dollars. Another feature most customers thought was important was that the temperature be displayed in Fahrenheit, as well as Celsius. Being able to set a temperature at which an alarm goes off to alert the user that the water is too hot for safe usage is another important feature. Being able to turn the device on and off is something that is implied when designing something that uses some sort of power. The display has to be easy to read, and the overall weight of the product shouldn't be too heavy. Another criterion that was expressed as being important was that the display can be moved to a convenient location. In order for it to be an effective means of measuring temperature, it needs to be sensitive, accurate, and have a wide enough range to capture any temperature water the shower might output.

Cost:

Inexpensive	3
Moderate	2
Expensive	1

Fahrenheit and Celsius display:

Both	3
One or the other	2
No specification	1

Alarm:

Programmable	3
Preset Value	2
None	1

Clear Display:

Easy to read	2
Difficult to read	1

Weight:

Light	3
Moderate	2
Heavy	1

Moveable Display:

Free range	4
Long wire	3
Rotate screen	2
Unable to move	1

Sensitivity:

Very sensitive	3
Moderate	2
Poor	1

Accuracy:

Very Accurate	3
Moderate	2
Poor	1

Range:

Wide	3
Moderate	2
Small	1

Customer weights

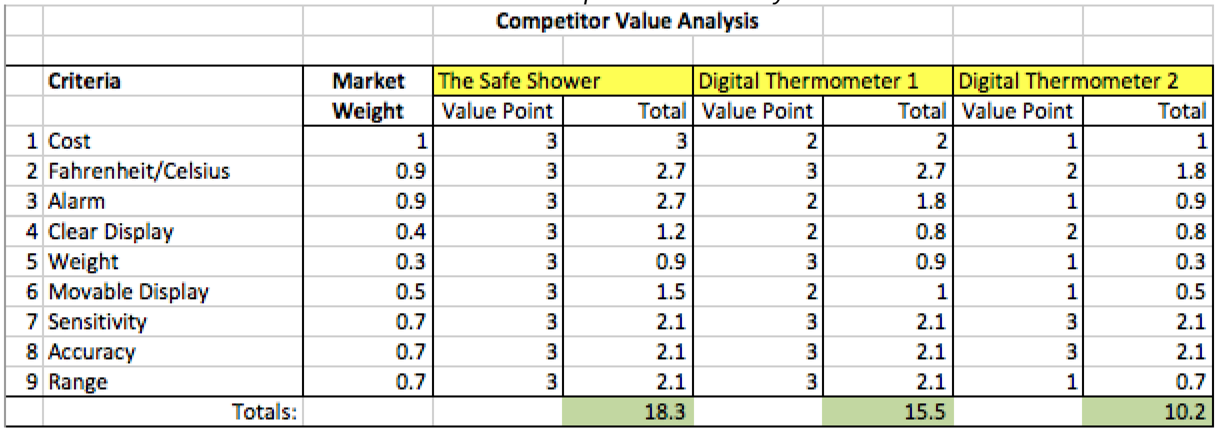
The weights assigned to the various criteria are dependent on how the customers value them. Cost is the most important factor for most people when deciding between similar products. Extra features, such as the alarm, and the ability to switch between Fahrenheit and Celsius are also very important in the customer's eyes. The sensitivity, accuracy, and range of the temperature sensor is more of an implicit feature that the customers value highly. Being able to move the display isn't the most important aspect, but it is still relevant. Most displays that are available in the market are pretty easy to read, so it isn't too important, as is the weight, especially since it will be mounted to the shower once installed.

Weights:

- Cost (1)
- Alarm (0.9)
- Fahrenheit/Celsius (0.9)
- On/Off (0.8)
- Sensitivity/Accuracy/Range (0.7)
- Movable Display (0.5)
- Clarity of Display (0.4)
- Weight (0.3)

The following table compares our product to two other products in the market. This is in order to determine whether our product will be successful compared to the other products.

Table 1: Competitor Value Analysis



Competitor Value Analysis

	Criteria	**Market**	The Safe Shower		Digital Thermometer 1		Digital Thermometer 2	
		Weight	Value Point	Total	Value Point	Total	Value Point	Total
1	Cost	1	3	3	2	2	1	1
2	Fahrenheit/Celsius	0.9	3	2.7	3	2.7	2	1.8
3	Alarm	0.9	3	2.7	2	1.8	1	0.9
4	Clear Display	0.4	3	1.2	2	0.8	2	0.8
5	Weight	0.3	3	0.9	3	0.9	1	0.3
6	Movable Display	0.5	3	1.5	2	1	1	0.5
7	Sensitivity	0.7	3	2.1	3	2.1	3	2.1
8	Accuracy	0.7	3	2.1	3	2.1	3	2.1
9	Range	0.7	3	2.1	3	2.1	1	0.7
	Totals:			18.3		15.5		10.2

As you can see, The Safe Shower finishes better than the other two products. The biggest factors in determining which product was best were the cost, the ability to switch the display between Fahrenheit and Celsius, and the alarm functionality. Overall, our product will be superior to the other products already available.

3 Design Approach

In order to obtain a competitive product that meets the customer requirements while still adhering to the requirements, there will be a tradeoff between cost and quality. The entire product will be broken down into a few different blocks in order to distribute the workload among everyone, and it will allow the person responsible to become the group “expert” on that specific module.

3.1 System Architecture:

The purpose of the Safe Shower is to provide the user with quick and accurate information about the temperature of the water in the shower. With the market intended for parents with young children, it is important that the Safe Shower be accurate, and provides a good alarm for when the temperature reaches a certain point. Figure 1 is the block diagram that was created to represent the different modules of the Safe Shower.

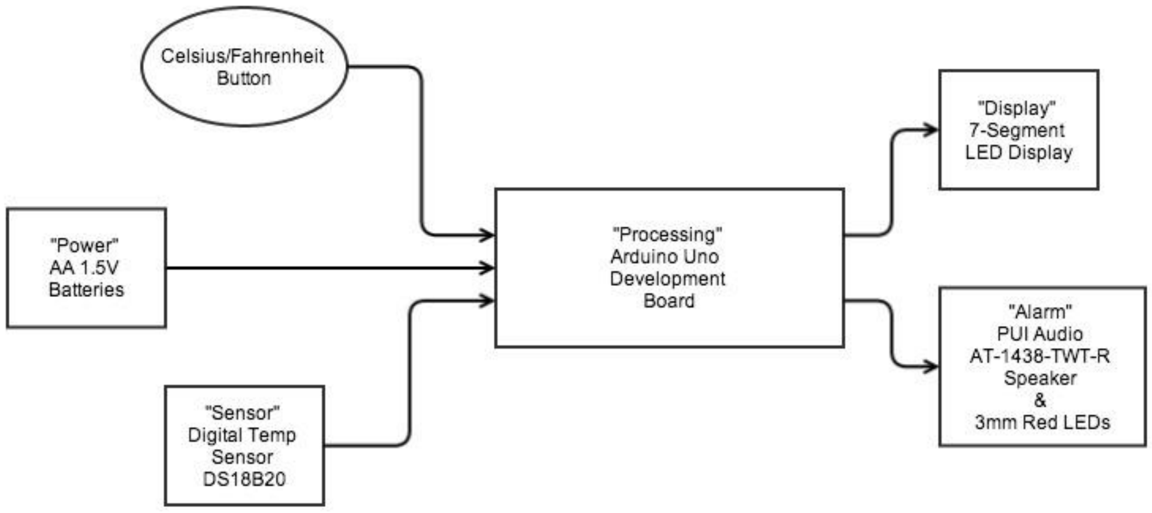

Celsius/Fahrenheit Button
"Power" AA 1.5V Batteries
"Sensor" Digital Temp Sensor DS18B20
"Processing" Arduino Uno Development Board
"Display" 7-Segment LED Display
"Alarm" PUI Audio AT-1438-TWT-R Speaker & 3mm Red LEDs

Figure 1: Block Diagram for the Safe Shower

The power block is essential for providing power to all the other modules. This isn’t necessarily a customer requirement, but more of an essential requirement. This will most likely be implemented with batteries, due to where the device will be used, and having it connected to a wall outlet would be very impractical, and potentially very dangerous.

The processing block is another essential part of the product. The information from the sensor will be coming into the microcontroller in the form of a 16-bit signal. The microcontroller will have to process it and display that temperature on the display screen. Besides that, the microcontroller needs to determine if the temperature exceeds the upper limit set by the user. If that happens, the microprocessor needs to output a logic high to the digital I/O ports that are connected to the buzzer and the LEDs so they can go off in the form of an alarm.

Without the temperature sensor, this product would not be able to function. It is essential for reading the temperature of the water and sending that information to the microcontroller, so the temperature can be displayed. Based on the customer requirements, the sensor needs to be accurate and fast. It also needs to be easy to work with and waterproof, due to the intended environment.

A display screen is required in order to display the temperature of the water flowing out of the shower. The processing unit of the product will provide the information that it will display. There will also be a button that allows the user to change the display between degrees Celsius and degrees Fahrenheit.

Finally, the purpose of the alarm is to alert the user when the temperature of the water in the shower is too hot for safe use. When the survey was conducted, the participants were asked if they would be interested in a product that, along with displaying the temperature of the water, would alert them if the water exceeded a certain value. This module's purpose is to fulfill that requirement. The preferred implementation of this module will be hardware, such as a visual and audible alert, that will go off when the set temperature is reached and/or exceeded.

3.2 Module Definition:

3.2.1 Development Board & Microcontroller

We are using the Arduino Uno R3 ATmega328P for our development board, which can be seen in Figure 2 below. The Arduino Uno is an essential component of the design, because it will act as the brains of the product.

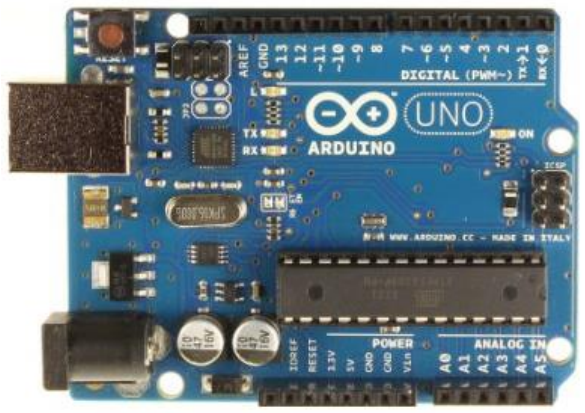


Figure 2: Arduino UNO R3 ATmega328P

Power will be supplied to the Arduino Uno from the 9V battery source. Once the power enters the development board, the voltage regulator will regulate the voltage from 9V to 5V. The microcontroller is directly connected to the DS28B20 temperature sensor probe, the piezoelectric buzzer, 4 digit 7 segment display and the LEDs. Since the Arduino Uno is connected to four other components, it is necessary to make sure it is supplying enough power to all those sources. Other sections will talk in detail about how those components are being powered.

The microcontroller has 14 Digital I/O pins. Each of those pins has an internal pull up resistor, which can be turned on and/or off. It also has 6 Analog input pins, which can also be configured as digital I/O. It also has an on board voltage supply of 5V and 3.3V along with a couple ground pins. The ATmega328P microcontroller has a maximum power rating of 280 mW and a flash memory of 32KB, which should be enough for our project. Figure 3 below shows a schematic taken from Arduino’s website of how the Arduino board is laid out.

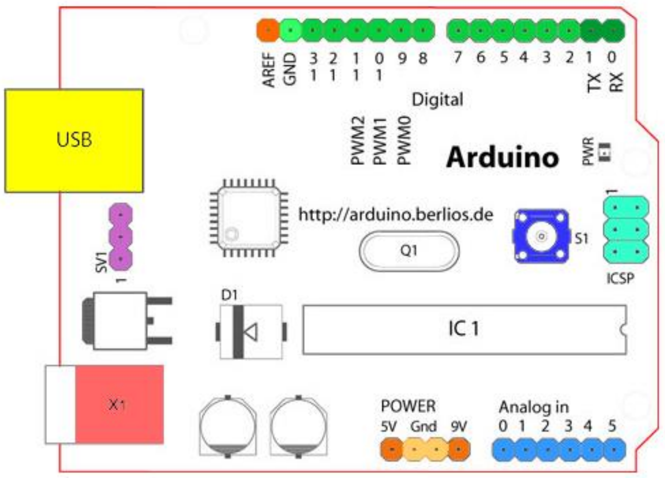


Figure 3: Arduino UNO R3 Schematic starting clockwise from Top- left

This is a good visual aid, because it clearly labels what everything is on the board. The Arduino Board and USB cable will cost $22.25, which will account for the largest portion of our $50 budget.

3.2.2 Temperature Sensor

The sensor will be taking temperature readings from the shower and sending them to the microcontroller. The sensor that will be used is the DS18B20 Digital Temperature Sensor, and it can be seen in Figure 4 below.

Figure 4: DS18B20 Digital Temperature Sensor

The sensor will be powered by the 5V DC output of the development board and can communicate over one of the board's digital I/O pins. The sensor's temperature reading can be configured to be anywhere from 9-12 bits. When the sensor receives the Convert T command, this data is stored in the scratchpad in a 16 bit, sign-extended two's complement format. This can then be read using the Read Scratchpad command. This sends the temperature information through the 1-Wire interface, least significant bit first.

The sensor was found to have an operating current of 1.5mA, so the power that this component will consume is 7.5mW. Once the sensor is in hand, testing can be done using hot coffee (or water) to verify the sensor is operating properly. Though example code for this sensor is available online, the code will need to be tested on its own before adding more components. Once it is verified that everything is working properly, then more aspects of the Safe Shower can be added.

3.2.3 Four Digit 7 Segment Display

We will be using a 4 digit 7-segment display to display the temperature, and it can be seen in Figure 5 below. The reason a 7-segment display was chosen over an LCD screen is because they were cheaper than LCD screens, and are much easier to read. Unfortunately this comes at the cost of significantly higher power consumption.

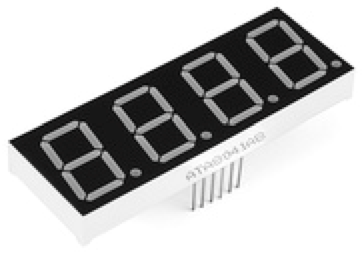


Figure 5: 4 Digit 7 Segment Display

This specific 4-digit 7-segment display will cost only $1.50. The maximum power rating for this 7-segment display is 38mW per segment. Continuous forward current needed by the LEDs in this 4-digit segment display is 20 mA, which can easily be provided by one of the digital I/O pins of our Arduino UNO R3 board. The 7-segment display will not always be turned on while a consumer is taking a shower. It will turn on and off at a set temperature in order to limit the power consumption. When it does turn on, it will continuously display the temperature of water that was detected by the sensor and processed by the microcontroller.

3.2.4 Alarm

For this module, some sort of visual and audible alert for the user was necessary. The target market for the Safe Shower is parents with young children, so if the alarm goes off, the child taking a shower can see/hear the alarm go off, and the parent might be able to hear the alarm go off as well. The implementation of the alarm will be a combination of two red 5mm LEDs and a single piezo transducer, a type of speaker/buzzer. The piezo transducer in Figure 6 below is the AT-1438-TWT-R transducer, manufactured by PUI Audio.

Figure 6: AT-1438-TWT-R Transducer

This is a relatively small device; 1.3cm tall and 1.38cm across, therefore, it should not take up much room at all. The reasoning behind this choice was that this particular device has a small power consumption compared to other options. The power consumption (P) = VI = (5V)(1mA) = 5mW, which is very minimal compared to the microcontroller/development board. It is also able to operate between -30 ~ +80 °C, which is optimal for use in a shower, because a shower will most likely be between 25 and 50 °C. This particular transducer produces an 80dB sound at 10cm, so depending on where the device is located in the shower, the alarm will be somewhere between 60 and 70 dB, which won’t be too loud for the user, but probably not loud enough for a parent to hear it from another room. The 5mm red LED in Figure 7, is the T-1 3/4 (5mm) SOLID STATE LAMP, manufactured by Kingbright.

Figure 7: LED

The LED will be responsible for the visual aspect of the alarm, as the transducer was responsible for the audible aspect. The reasoning behind this choice was that this particular device has smaller power consumption than some comparable options. The power consumption (P) = VI = (2V)(20mA) = 40mW, which is also much smaller than the microcontroller/development board, but it is still a significant amount. After digging through the data sheet a little bit more, it was found that the LED is able to operate between -40 ~ +85 °C, which is optimal for use in a shower, because a shower will most likely be between 25 and 50 °C. This particular LED has a wavelength of 625 nm. It has a brightness of about 45 mcd, and a viewing angle of 30°, so if the device is located at or about head-level, the LEDs should be visible when the alarm goes off.

3.2.5 Power

The power for the Safe Shower will be in the form of six AA 1.5V Alkaline batteries, seen in Figure 8 below. They will be responsible for providing the system with approximately 9V DC.

Figure 8: AA 1.5V Alkaline Battery

These batteries were chosen due to their good nominal voltage, their very good milliamp-hours capacity, and their operating temperature. There is a significant amount of power being used by the components in the Safe Shower, so it was necessary to find an adequate power supply to meet the demand. The Watt-hours that can be produced by these batteries is 6 batteries * 1.5V * 2Ah (with a constant current draw of 500mA) = 18Wh. After all the other device's powers were determined, they came up to be 746.5mW.

The usage for this device in the shower was determined to be about 15min per user, with 2 uses per day (twice the number of average children in a household). Because the market is intended for parents with young children, so they could make sure their child was safe while taking a shower, it was assumed that the Safe Shower would be used by the children, not the parents. To account for unexpected spikes in the power consumption, a buffer was considered in the usage time, so instead of 15min/day, 30min/day was used in the Watt-hours calculation. The calculations get a little complicated at this point to account for the average time each aspect of the system would be on. In the end, the alarm consumed 0.01417Wh/mth, the Arduino development board consumed 4.41Wh/mth, the sensor accounted for 0.05625Wh/mth, and the display was 1.08Wh/mth. This came out to a total of 5.56Wh/mth. In order to calculate how long the batteries would last, the Watt-hours of the batteries was divided by the power consumption of the components. At these rates, the batteries would last for 3.24 months.

4 Product Results

4.1 Product Functionality:

While we were generally happy with the functionality of our product, there are a few areas it could be improved. Our alarm buzzer was quieter than anticipated, so it was decided that a louder one would be implemented in any further redesign of our product. The Celsius/Fahrenheit button had some issues with receiving phantom button presses, which we discovered were due to it not fitting properly into our breadboard. Finally, the 7 segment display still flashes due to the time it takes to read from the sensor.

4.2 System Testing & Results:

When testing our product, we determined that all of the major systems were functioning properly. When testing the temperature sensor, we immersed it in hot coffee, and we were able to get an accurate reading. Once the temperature was high enough, the alarm mechanism worked as planned.

5 Cost Analysis

We went through rigorous process to determine the production and selling cost for our product. We opted for a fairly simple route for a business plan. To determine our cost analysis, we first determined how we were going to produce and market our product. Following is the list of four steps we came up with:

- Produce a 1000 units
- Rent out a garage
- Hire four workers
- Sell products online

5.1 Initial Investment:

We want to produce only a 1000 units for our product and then wait until we sell those 1000 units. The main reason for doing this is because we don't want to invest way too much in our product right now. We will see how the results come out for the first batch of 1000 units produced. If it seems like people are more interested in purchasing our products, we will go ahead and turn this startup venture into a business. The following table represents the bill of materials for producing 1000 units:

Table 2: BOM for 1000 Units

Item	Quantity	Cost per Unit (US Dollars)	Total Cost (US Dollars)
ATMega328P	1000	1.675	1,675.0
Temperature Sensor	1000	5.000	5,000.0
4 Digit 7 Segment Display	1000	1.200	1,200.0
Jumper Wires	2 Boxes	200	400.0
LEDs	1000	0.050	50.0
Speaker	1000	0.742	742.0
Case & T shape	1000	5.000	5,000.0
PCB Board	1000	0.500	50.0
Total	**1000**	**14.317**	**14,317.0**

The sum of the BOM for a 1000 products came out to $14,317. That means cost per unit was $14.317. After figuring out the cost for our BOM, the next step was to calculate our fixed costs. Following items were included in our fixed costs:

- Facility
- Utilities
- Website Maintenance
- Salaries

We will be required to rent out a facility to produce our products and store them. We figured, a garage would be a good place to start, which would cost around $300 per month. The utilities for the garage would cost around $150 per month. We also decided to sell our products online through a website compared to selling them through retailers and distributors. The final price of a product is almost three to four times higher than the production cost if you go through retailers and distributors to sell your product. This would raise the cost for our product to approximately $100 and we figured no one would be willing to pay that much for our product. This was the reason why we decided to go with a website to sell our products. Selling products through website would cost an extra $50 per month in maintenance. Finally we also included $8,800 as salaries of the four workers who will be working on producing the product. Following table represents the details of our fixed cost analysis.

Table 3: Fixed Costs

Item	**Cost (US Dollar)**
Facility (Rent a garage)	$300/month
Utilities	$150/month
Cost for Website Maintenance	$50/month
Salaries	$8,300/month
Total Fixed Costs	**$8,800**

As discussed above, we will require hiring four workers to manufacture our product. Three out of those four workers would be us ourselves. Therefore we will only need to hire one worker from outside. That worker would be given the responsibility to assemble the product. Whereas, the rest of us will be responsible for wiring the products. We want to wire the products ourselves to make sure they are of good quality. The following table represents the details for the salaries we would be paying to the workers

Table 4: Salaries of Workers

Worker	**Salary**
1 worker from outside	$10/hour
3 workers (ourselves)	$15/hour

5.2 Cost of Production:

After calculating all of our variable and fixed costs, the next phase in the process was to determine the cost of production for our product. The table below represents in detail the cost of production per unit. As mentioned in the previous section, the cost per unit of our product was $14.317 for the bill of materials. Manufacturing cost for our product was divided into two sub portions. The first segment is 'Assembly.' We estimated the time of 20 minutes to assemble one product. Since the outside worker would be assembling products at the salary of $10/hour that meant the cost of assembling one unit was $3.33. We also estimated the same time period to wire one product. Wiring one product at $15/hour meant it took $5.00 to wire one product. We also added $0.50 to account for the rent of the garage and website maintenance. At the end, our product came out to around $23.15.

Table 5: Detailed Cost Breakdown of Each Unit

Item	**Cost (US Dollar)**
BOM	14.317
Manufacturing	8.333
Assembly (20 minutes at $10/hour)	*3.333*
Wiring (20 minutes at $15/hour)	*5.000*
Warehouse & Website	0.500
Total Unit Cost	**23.15**
100% Margin	**46.30**
Final Cost Price	**44.99**

A 100% markup on our product would raise its price to $46.30. We decided to decrease that price a bit to make it look nice. Finally we came to an agreement to sell our product at $44.99. This is a 94.3% mark up on our cost and will give us a good return on investment as well. We will talk about the return on investment in more detail in the next section.

5.3 Return on Investment:

To determine our Return on Investment we used the help of different plots, which are shown below. Theoretically, our return on investment was calculated around 94.3%.

ROI = (Gain – Cost of Production)/Cost of Production

= (44990-23150)/23150

= 94.3%

The plot below represents our Revenue vs. Unit Sales. The dark blue line on the top in the plot represents the revenue generated by selling our 1000 units. The yellow line represents the total cost of production for each of those units, which is obtained by the sum of the remaining two lines represented in the plot. The grey line represents the fixed costs associated with producing our units whereas the orange line at the bottom represents our variable costs. These variable costs are the costs associated with the bill of materials and how much it costs to produce each unit. The point of intersection between the dark blue (revenue) and yellow (fixed costs) represents our breakeven point which is 287 units.

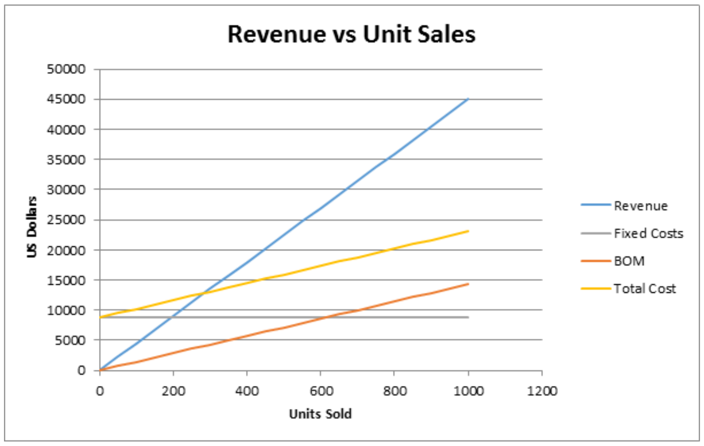


Figure 9: Plot for Revenue vs Unit Sales

The second plot below is a plot of Profit vs. Sales of our product. This plot verifies our breakeven point of 287 units. The slope of the line represents the cost per unit, which is $44.99. After selling 287 units we will be independent and any money made after that would count towards our profit.

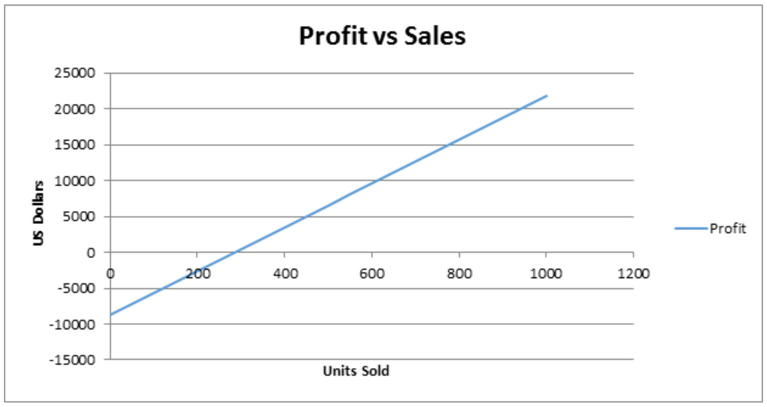


Figure 10: Plot for Profit vs Units Sold

The third plot below represents Units Sold vs. Time. The blue line represents the line of the breakeven point. This line has a small positive slope because the break even number will increase every month. This is because we will need to pay rent for the warehouse and website maintenance fees. After doing some calculations we figured the breakeven point would increase by 16 units every month. If we sell 50 units/month, we can easily break even within 7 month. Whereas if we sell 40 units/month we will break within one year and by selling 30 units/month we will break even in almost two years.

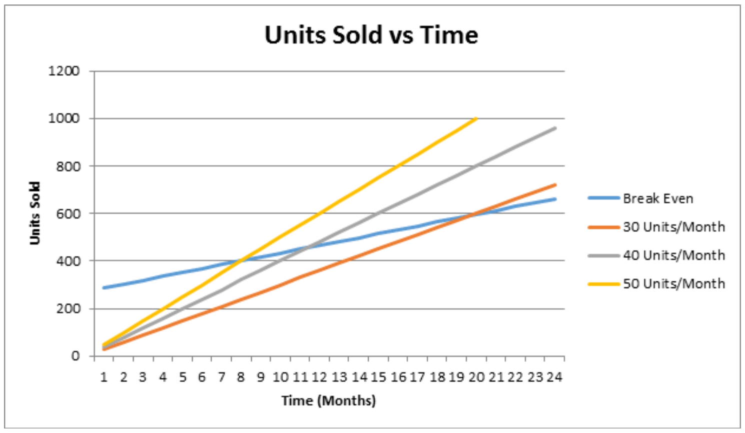


Figure 11: Plot for Units Sold vs Time

6 Failure & Hazard Analysis

The display on our prototype right now turns on only when the temperature of water exceeds 25 degrees Celsius. We have this feature incorporated in the Safe Shower to save the battery life. For the prototype, the blinking light of the Arduino can represent the device to be working. However, in reality we won't be using Arduino boards, therefore we are trying to include a green LED in our product which would keep going off all the time even when the display screen is not displaying any temperature. This would represent the device to be functional.

Our current prototype has a buzzer, which is not very loud. So that can be a reason for failure as well. We will try to replace our buzzer with a much louder one so the parents of the children can actually listen to the alarm once it goes off. If there is some fault in our circuit, one of the LEDs can burn and that can be a hazard. To prevent that hazard we will make sure the LEDs go inside a plastic casing but are still visible to the user. Therefore, even if the LEDs blow off, they won't cause damage to the person taking the shower.

There could be another hazard associated with our prototype. Since our display is connected to the sensor through a wire, there might be a chance that the wire is a source of annoyance to the person taking the shower. In case of panic, maybe the person can get his arm stuck in the wire as well, provided the wire is loose and is not attached properly to the shower. Besides these hazards, we don't think our device has any other hazards associated with it. There is no hazard relating to the screen or the alarm burning or breaking. This is because the components we chose could operate very high temperatures (Appendix C).

7 Recommendations and Conclusion

If future work is to be done on The Safe Shower, the biggest thing that will have to be done is reduce its size. Right now it is too big to fit nicely into a waterproof box. One way this can be done is to have the components soldered onto a PCB. Another thing that will need to be done is to replace the development board with just a microprocessor. This will also help to reduce the size of the final product. The development board also consumed quite a bit of power, so the elimination of the board and just having the processor will also help to reduce power consumption. Another big power consumer was the 7-segment display that was used. An LCD screen could be used instead which would greatly reduce the power consumption, but at the disadvantage of not being as clear of a display as the 7-segment. Finally, the buzzer that was purchased for this project turned out to be too quiet for our intended application. We would need to look into getting a louder buzzer in order for the customer to be able to hear it while bathing.

8 Appendices

Appendix A: Schematic & Parts List

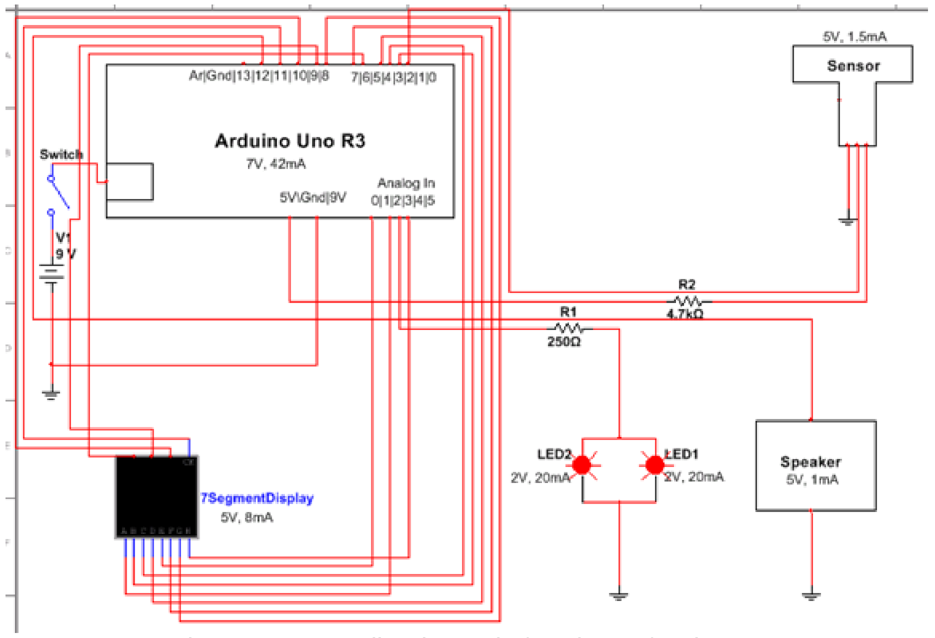


Figure 12: Overall Schematic for The Safe Shower

Table 6: Parts List for the Safe Shower

Component	Price	Source
Arduino UNO ATMega328P	$22.25	https://www.amazon.com
Digital Temperature Sensor	$11.00	https://www.sparkfun.com
4 Digit 7 Segment Display	$01.50	https://www.sparkfun.com
Piezoelectric Speaker	$02.03	http://www.mouser.com
LEDs	$00.50	http://www.mouser.com

*Note: This part list represents the parts we bought to make our prototype for this project. The parts required to make the actual product are discussed in section 5.1 of this report under BOM

Appendix B: Computer Code

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged into port 2 on the Arduino
#define ONE_WIRE_BUS 12

// Setup a oneWire instance to communicate with any OneWire devices
(not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

int aPin = 10;  //           A
int bPin = 6;  //        ________
int cPin = A3;  //      |        |
int dPin = 4;  //   F   |        |  B
int ePin = 5;  //       |    G   |
int fPin = 9;  //       |________|
int gPin = A4;  //      |        |
int GND1 = 11;  //      |        |
int GND2 = 8; //    E   |        |   C
int GND3 = 7; //        |________|
int GND4 = A5; //           D
int decPt = A2;
int unitButton = 2;
float num;
int dig1 = 0;
int dig2 = 0;
int dig3 = 0;
int dig4 = 0;
int DTime = 4;
int led = 13;
int buzzer = 3;
const unsigned int debounce_time = 500;
static unsigned long last_interrupt_time = 0;
int C = 0;
int F = 1;
int unit = 0;

void setup()
{
  pinMode(aPin, OUTPUT);
  pinMode(bPin, OUTPUT);
  pinMode(cPin, OUTPUT);
  pinMode(dPin, OUTPUT);
  pinMode(ePin, OUTPUT);
  pinMode(fPin, OUTPUT);
  pinMode(gPin, OUTPUT);
  pinMode(GND1, OUTPUT);
  pinMode(GND2, OUTPUT);
  pinMode(GND3, OUTPUT);
  pinMode(GND4, OUTPUT);
  pinMode(decPt, OUTPUT);
  pinMode(led, OUTPUT);
  pinMode(buzzer, OUTPUT);
  attachInterrupt(0, tempISR, RISING);
  Serial.begin(9600);
  sensors.begin();
}
void loop()
{
  int a = 1;
  int b = 1;
  int decimal;
  int displaynum;
  digitalWrite( GND1, HIGH);
  digitalWrite( GND2, HIGH);
  digitalWrite( GND3, HIGH);
  digitalWrite( GND4, HIGH);

  if (a > 0)
  {
    Serial.print("Requesting temperature...");
    sensors.requestTemperatures(); // Send the command to get
temperature
    Serial.println("DONE");
    num = sensors.getTempCByIndex(0);
    displaynum = num;
    if (num > 50)
    {
      digitalWrite(led, HIGH);
      delay(100);
      digitalWrite(led, LOW);
      delay(100);
      beep(200);
      digitalWrite(led, HIGH);
      delay(100);
      digitalWrite(led, LOW);
      delay(100);
      beep(200);
      digitalWrite(led, HIGH);
      delay(100);
      digitalWrite(led, LOW);
      delay(100);
      beep(200);
      digitalWrite(led, HIGH);
      delay(100);
      digitalWrite(led, LOW);
      delay(100);
      beep(200);
      digitalWrite(led, HIGH);
      delay(100);
      digitalWrite(led, LOW);
      delay(100);
      beep(200);
    }
    Serial.print("Temperature is: ");
    if (unit == F)
    {
      num = num*1.8 +32;
    }
    Serial.println(num);
    if (num >= 100)
    {
      num *= 10;
      dig1 = num / 1000;
      num = num - (dig1 * 1000);
      dig2 = num / 100;
      num = num - (dig2 * 100);
      dig3 = num / 10;
      dig4 = num - (dig3 *10);
      decimal = 3;
    }
    else
    {
      num *= 100;
      dig1 = num / 1000;
      num = num - (dig1 * 1000);
      dig2 = num / 100;
      num = num - (dig2 * 100);
      dig3 = num / 10;
      dig4 = num - (dig3 *10);
      decimal = 2;
    }
  }
  if (displaynum >= 30)
  {
  for(b=0; b< 100; b++)
  {
  digitalWrite( GND4, LOW);    //digit 4
  pickNumber(dig4);
  delay(DTime);
  digitalWrite( GND4, HIGH);

  digitalWrite( GND3, LOW);    //digit 3
  pickNumber(dig3);
  if (decimal == 3)
  {
    digitalWrite( decPt, HIGH);
  }
```

```
  delay(DTime);
  if (decimal == 3)
  {
    digitalWrite( decPt, LOW);
  }
  digitalWrite( GND3, HIGH);

  digitalWrite( GND2, LOW);   //digit 2
  pickNumber(dig2);
  if (decimal == 2)
  {
    digitalWrite( decPt, HIGH);
  }
  delay(DTime);
  if (decimal == 2)
  {
    digitalWrite( decPt, LOW);
  }
  digitalWrite( GND2, HIGH);

  digitalWrite( GND1, LOW);   //digit 1
  pickNumber(dig1);
  delay(DTime);
  digitalWrite( GND1, HIGH);
  }
  }
}

void tempISR()
{
  unsigned long interrupt_time = millis();
  if (interrupt_time - last_interrupt_time > debounce_time)
  {
    //if(m==1){
    if (unit == C)
    {
      unit = F;
      //m=0;
    }
    else
    {
      unit = C;
      //m=0;
    }
   // }
    Serial.println("press");
    Serial.println(unit);
  }
  last_interrupt_time = interrupt_time;
}
```

```
void beep(unsigned char delayms)
{
  analogWrite(3, 20);      // Almost any value can be used except 0
and 255
                            // experiment to get the best tone
  delay(delayms);          // wait for a delayms ms
  analogWrite(3, 0);       // 0 turns it off
  delay(delayms);          // wait for a delayms ms
}


void pickNumber(int x){
   switch(x){
     case 1: one(); break;
     case 2: two(); break;
     case 3: three(); break;
     case 4: four(); break;
     case 5: five(); break;
     case 6: six(); break;
     case 7: seven(); break;
     case 8: eight(); break;
     case 9: nine(); break;
     default: zero(); break;
   }
}

void clearLEDs()
{
  digitalWrite(  2, LOW); // A
  digitalWrite(  3, LOW); // B
  digitalWrite(  4, LOW); // C
  digitalWrite(  5, LOW); // D
  digitalWrite(  6, LOW); // E
  digitalWrite(  7, LOW); // F
  digitalWrite(  8, LOW); // G
}

void one()
{
  digitalWrite( aPin, LOW);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, LOW);
  digitalWrite( ePin, LOW);
  digitalWrite( fPin, LOW);
  digitalWrite( gPin, LOW);
}

void two()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, LOW);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, HIGH);
  digitalWrite( fPin, LOW);
  digitalWrite( gPin, HIGH);
}

void three()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, LOW);
  digitalWrite( fPin, LOW);
  digitalWrite( gPin, HIGH);
}

void four()
{
  digitalWrite( aPin, LOW);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, LOW);
  digitalWrite( ePin, LOW);
  digitalWrite( fPin, HIGH);
  digitalWrite( gPin, HIGH);
}

void five()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, LOW);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, LOW);
  digitalWrite( fPin, HIGH);
  digitalWrite( gPin, HIGH);
}

void six()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, LOW);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, HIGH);
  digitalWrite( fPin, HIGH);
  digitalWrite( gPin, HIGH);
```

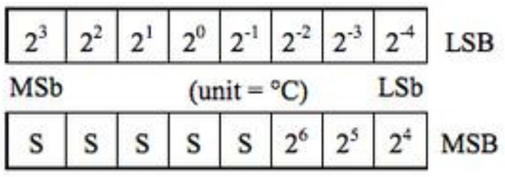
```
}

void seven()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, LOW);
  digitalWrite( ePin, LOW);
  digitalWrite( fPin, LOW);
  digitalWrite( gPin, LOW);
}

void eight()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, HIGH);
  digitalWrite( fPin, HIGH);
  digitalWrite( gPin, HIGH);
}

void nine()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, LOW);
  digitalWrite( fPin, HIGH);
  digitalWrite( gPin, HIGH);
}

void zero()
{
  digitalWrite( aPin, HIGH);
  digitalWrite( bPin, HIGH);
  digitalWrite( cPin, HIGH);
  digitalWrite( dPin, HIGH);
  digitalWrite( ePin, HIGH);
  digitalWrite( fPin, HIGH);
  digitalWrite( gPin, LOW);
}
```

Appendix C: Component Specifications

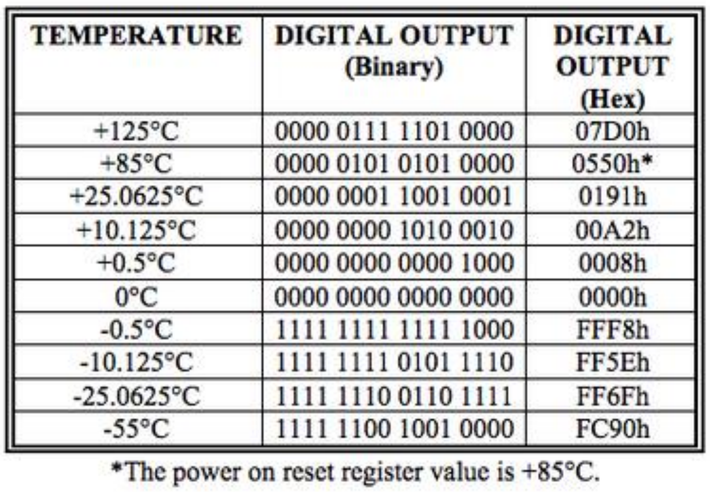
Table 7 – Properties of AT328P Micro-controller

ATmega328P	**Values**
Digital I/O Pins	14 of which 6 provide PWM output
Analog Input Pins	6
DC Current per I/O pin	40mA
Flash Memory	32KB
Maximum Voltage Rating	7 – 12V
Maximum Power Rating	280 mW

Table 8: Temperature and Corresponding Digital Output from Temperature Sensor



2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	LSB
MSb			(unit = °C)				LSb	
S	S	S	S	S	2^6	2^5	2^4	MSB



TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+125°C	0000 0111 1101 0000	07D0h
+85°C	0000 0101 0101 0000	0550h*
+25.0625°C	0000 0001 1001 0001	0191h
+10.125°C	0000 0000 1010 0010	00A2h
+0.5°C	0000 0000 0000 1000	0008h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1111 1000	FFF8h
-10.125°C	1111 1111 0101 1110	FF5Eh
-25.0625°C	1111 1110 0110 1111	FF6Fh
-55°C	1111 1100 1001 0000	FC90h

*The power on reset register value is +85°C.

Table 9: Specifications for the 4 digit 7 segment display

4 Digit 7 Segment Display	Maximum Rating
Power Dissipation	38mW
Peak Forward Current	60mA
Continuous Forward Current	20mA per segment
Forward Voltage	1.9V
Operating Temperature Range	-25 - +65 °C
Storage Temperature Range	-20 - + 70 °C
Static Voltage	1000V

Table 10: Specifications for the AT-1438-TWT-R Transducer

AT-1438-TWT-R Transducer	Values
Rated Voltage	5V
Operating Voltage Range	1 - 20V
Rated Current (Max)	1mA
Capacitance	13,000 +/- 30% pF
Minimum SPL @ 10cm	80 dBA
Resonant Frequency	3,800 +/- 500 Hz
Operating Temperature	-30 - +80°C
Weight	1g

Table 11: Specifications for the red 5mm LED

Red 5mm LED	**Values**
Wavelength	625nm
Forward Voltage	2V
Forward Current	20mA
Reverse Voltage	5V
Reverse Current	10□A
Capacitance	15pF

Table 12: Specifications for AA 1.5V Alkaline Batteries

Alkaline AA Batteries	**Values**
Nominal Voltage	1.5V
Operating Temperature	-20°C - 54°C (-4°F - 130°F)
Max Discharge (single battery)	1A cont.
Typical Weight	24g
Milliamp-Hours Capacity	2000mAh (500mA constant discharge)

9. References:

PUI Audio, AT-1438-TWT-R Transducer
http://www.puiaudio.com/pdf/AT-1438-TWT-R.pdf

Duracell, LF1500 AA Lithium Iron DiSulfide Batteries
http://professional.duracell.com/downloads/datasheets/product/Ultra%20Lithium/Ultra-Lithium_AA.pdf

Kingbright, T-1 3/4 (5mm) SOLID STATE LAMP, LED
http://www.mouser.com/ds/2/216/WP7113ID-48891.pdf

Mouser Electronics, Kingbright, T-1 3/4 (5mm) SOLID STATE LAMP, LED
http://www.mouser.com/ProductDetail/Kingbright/WP7113ID/?qs=sGAEpiMZZMtmwHDZQCdlqXYfWhYL4D7CladL6GeZsOU%3d

Mouser Electronics, PUI Audio, AT-1438-TWT-R Transducer
http://www.mouser.com/ProductDetail/PUI-Audio/AT-1438-TWT-R/?qs=%2fha2pyFaduiCd9mCPpRmHCZrxFVWsmTcBzjekGkYqm6taPPGwyJjqA%3d%3d

PUI Audio, Speaker Power and Distance
http://www.digikey.com/Web%20Export/Supplier%20Content/PUI_668/PDF/PUI_speaker_power_distance.pdf?redirected=1

Sparkfun, Temperature Sensor - Waterproof (DS18B20)
https://www.sparkfun.com/products/11050

Dallas Semiconductor, DS18B20 Programmable Resolution 1-Wire® Digital Thermometer
http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Temp/DS18B20.pdf

Arduino, Arduino UNO R3 – AT328P
https://www.arduino.cc/en/Reference/Board

Sparkfun, 4 Dgit 7 Segment Display
https://www.sparkfun.com/products/11045

10 Authorship Page

1 Introduction	Nate
1.1 Problem Statement	Pat
1.2 Market Research	Pat
2 Product Requirements	Pat
2.1 Customer Requirements	Pat
2.2 Competitive Value Analysis	Pat
3 Design Approach	Pat
3.1 System Architecture	Pat
3.2 Module Definitions	Pat
4.1 Product Functionality	Nate
4.2 System Testing & Results	Nate
5 Cost Analysis	Ali
5.1 Initial Investment	Ali
5.2 Cost of Production	Ali
5.3 Return on Investment	Ali
6 Failure & Hazard Analysis	Ali
7 Recommendations and Conclusion	Pat
8 Appendices	
Appendix A	Ali
Appendix B	Nate
Appendix C	Pat
9 References	Ali
10 Authorship Page	Pat
Formatting	Pat & Ali