ECE 2242 Final Project

The Breakfast Club

B.E.Sc Electrical & Computer Engineering

Logan Derry (251152160)

Matthew Dauphinais (251163218)

Lyudmil Marinov

April 1, 2022

We, Matthew Dauphinais and Logan Derry hereby certify that the proceeding project and report are our own work, except for where specifically referenced.

Matthew Dauphinais

Logan Derry

Abstract

The purpose of the project is to familiarize oneself with the Arduino software and hardware to design a device that solves a known problem. Western students are not eating breakfast frequently enough and require a way to prepare a healthy meal to start their day. The purpose of this project is to design a device that makes eating this meal more accessible and less time intensive. Through empathizing, it was found that the users have a specific set of needs that must be targeted in development. The main constraints associated with the project are safety and speed. An idea was found that fit the constraints effectively: the stovetop control. Following this decision, the Arduino library of actuators and sensors was explored. A total of 4 actuators and 4 sensors were chosen for the final circuit. This list includes: the LCD screen, DC relay, servo motor, IR sensor, Bluetooth module, real-time clock, ultrasonic proximity sensor, and thermistor. These components were individually tested and researched before implementing into a single circuit. All required libraries were downloaded and pasted into the Arduino program folder. Using EAGLE, a circuit schematic was created using pin headers to connect the required components. The circuit was tested and debugged until it functions as intended with minimal bugs. Initially, the circuit was made using a breadboard to ensure functionality. After successful testing and soldering, the components were transferred to the printed circuit board. All debugging procedures and solutions are stated in their associated sections of the report. The main functionality of the circuit is to turn on a stovetop until the food reaches the desired temperature, as set by the user via a Bluetooth connection. Depending on the state of the value returned from the IR sensor, a clock module is used as a timer to cook the food at a certain temperature. During the final demonstration, the circuit was able to run as intended. In conclusion, the design process was effectively followed, leading to a device that meets the constraints set by the user analysis.

Acknowledgments:

We would like to thank the professors of ECE 2242, and the electronic shop staff, for playing instrumental roles in the progress and completion of this project. ECE professors provided assistance and guidance through the progression of the report and course, while electronic shop staff provided the proper equipment to execute the final design.

Contents

Glossary

PCB – Printed Circuit Board

IR Sensor – Infrared Sensor. Detects infrared waves emitted by its corresponding remote. Decodes these signals into usable codes

RTC – Real time clock. Used in the final circuit as a timer to determine when the food is done

LCD – Liquid Crystal Display. Used to display information to the user.

Introduction

Western students are not eating breakfast frequently enough and require a way to prepare a healthy meal to start their day.

According to a study conducted by Gregory W. Philips looking at the correlation between eating breakfast and the success on a biology exam (Figure 1 – Students grades with and without breakfast), it was found that eating breakfast generally coincides with a greater grade earned [1]. This helps to prove that a healthy breakfast is a necessity for a student's success. In this study, 65.5% of students' self-report eating a breakfast, which is concerning, considering almost half of students are not reaping the benefits of a hearty breakfast. The leading cause of this missed meal, as explained by a study from The University of Bangladesh, is time management [2]. It was discovered that young adults and university students miss breakfast due to poor time management or lack of free time. This study found that 48% of breakfast-skipping students at Nigeria University, were doing so due to a lack of time.

Having a meal every morning has significant benefits on a student's wellbeing, including but not limited to: better weight management, higher energy, better concentration, and a more effective memory [3]. These are all very beneficial traits to a developing student, so it is essential that they eat something to start the day right.

The problem stems from many different factors, but mostly from a lack of time to cook [4]. Having to run to an 8:30 morning class does not leave a whole lot of time to eat for most students who

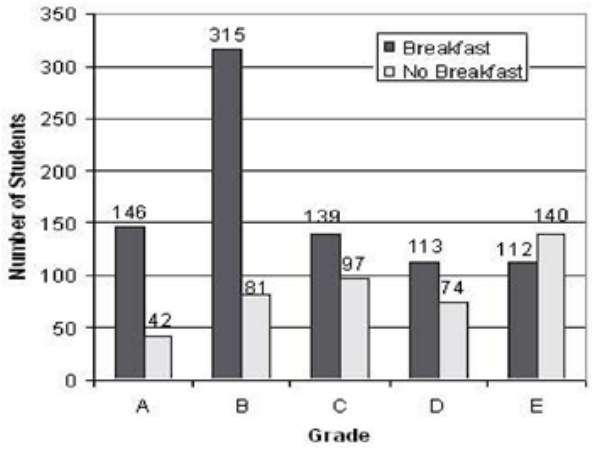
rush out of the house to catch the bus. A quick meal such as yogurt or milk could be substituted for a home-cooked meal of eggs and/or toast, but in most cases, fasting is the chosen option by students sprinting out the door. In research conducted by EatForLonger.Com, it was found that the average American spends between 10-30minutes preparing and eating breakfast [5]. This becomes 10-30 minutes many students cannot spare throughout their mornings. Moreover, current breakfast preparation times are too long, causing students to forego the meal altogether.

Something must be done to make it easier and quicker for students to prepare and eat breakfast considering their already busy schedule to give them the best chance to succeed in their academics.

The prime focus of this report is to achieve just this; to make a device that can ease the morning routine of Western students to allow the population to eat breakfast more often. A deeper dive into customer needs and wants will be conducted, and eventually an idea will be chosen that best fits the requirements of the project's success. The chosen project will also be developed and created over the course of the semester. A look into the individual actuators and sensors will eventually turn into the full integration of components into a circuit, complete with a Printed Circuit Board (PCB).

From this project, a potential business opportunity could arise. Students need an effective way to make food quickly and easily in the mornings, potentially opening a new market that our design can effectively target.

Figure 1 – Students grades with and without breakfast



Constraints

Primary:

To consider our project successful:

- a student must be able to successfully cook and eat a breakfast without consuming a great amount of time to do so. To put it in numbers, our device **must not take away more than 5 minutes** of a student's time in total to prepare their meal. This is considering their already busy schedule, and the fact that it is hard to justify allocating any more time to preparing a meal, since if they had time, they would cook something themselves. This is a reasonable benchmark, as this halves the current average breakfast prep-time, while being within reason to produce an effective meal. Overall, this holds the most weight of all constraints.

- The device must be **versatile** enough to be able to prepare at least one food and one drink item, but more is preferred to prevent repetition fatigue [6].This constraint holds lesser weight, because safety and speed are more important to the student in the morning.

- The device must have **safety procedures** in place to prevent unwanted injury from the operator. An example of this would be to cover any heating element to prevent burns and steam vents to prevent injury from hot/boiling condensation (beverages and such. This constraint holds great weight because any device needs to be safe for the consumer.

- The device must be **easy to use and straightforward** for even the least technologically savvy user. This is important because less injury is possible if the device is easy to use.

- Our device must give a **clear indication that the food is prepared and ready to eat.** This can be done audibly or visually. Rarely more important than safety and speed, this is still a necessity for the overall device.

- Our device must reliably **cook any food item to the same quality**, (No undercooking/overcooking). This is a must, considering most food is required to be cooked to be considered safe for ingestion.

- Our device must produce food within a **reasonable temperature range** (Not too hot, not too cold), as it is assumed the user will be eating produced food directly after cooking. This is the least important, considering students can have a single meal everyday.

In summary, our device must: take less than 5 minutes to use, be versatile, be safe to use, be easy to use, give a clear indication of being done, cook with consistent quality, be able to cook at a wide variety of temperatures

Below is a table weighing the constraints based on the importance to the customer. The column is the one that is used to compare to the options in the rows (ice, when looking at 'safety', it is compared to quick, then versatile, then safety, etc.). The determined number is the importance relative to the option being compared. Ex: 'Quick' is 1.25x more important than 'versatile'.

Table 1: Constraint Weighing

	Quick	Versatile	Safety	Easy to use	Clear indication of done	Consistency	Wide temp range	**Total**	**weighting**
Quick	1	1.25	0.33	2	1.333333333	2	2	**9.913333**	**0.150635**
Versatile	0.8	1	0.2	0.5	0.9	1.2	2	**6.6**	**0.100288**
Safety	3	5	1	1.2	3	4	4	**21.2**	**0.322138**
Easy to use	0.5	2	0.833333	1	1.333333333	3	4	**12.66667**	**0.192472**
Clear indication of done	0.75	1.111111	0.333333	0.75	1	1	1.3	**6.244444**	**0.094885**
Consistency	0.5	0.833333	0.25	0.33333333	1	1	1	**4.916667**	**0.07471**
Wide temp range	0.5	0.5	0.25	0.25	0.769230769	1	1	**4.269231**	**0.064872**

Secondary:

Although not a necessity, our device should:

- Have a way to **heat up** the meal to offer a warm breakfast to our user. This can be done for either coffee or classic breakfast items like eggs or bacon.

- **Be easy to clean** to minimize time consumption, post-usage. This is not important since the dirty dishes could be left in the sink until the student has more time to clean them.

- Be **appealing to look at** with nice to look at colors and shapes.

- Be **compatible with various types of products** such as milk alternatives (different boiling points) and various cereals.

- Be **easy to set up** to be ready to work the next morning
- **Operate on a timer,** to ease the breakfast process

In summary, it would be beneficial for our device to: be able to heat up the food, be easy to clean, be aesthetically appealing, be compatible with lots of food, be easy to set up, be operated on a timer.

Along with the above customer constraints, there is also a very strict time constraint. Whatever is pursued must be able to be fully completed in a two-month period. This includes testing of components, shipping of materials, and assembly of final product.

The cost constraint of the design must also be followed. A tight budget of less than $150 will be followed including the $60 Arduino first purchased for use in the design course.

Literature Review

Considering the theme of food for this report, a thorough look into the health and safety standards was conducted. As per the USDA, there is a list of internal cooking temperatures that must be achieved to consider a food safe to eat. This list is in Figure 2 :

Figure 2 [7] – USDA internal temp table

Product	Minimum Internal Temperature & Rest Time
Beef, Pork, Veal & Lamb Steaks, chops, roasts	145 °F (62.8 °C) and allow to rest for at least 3 minutes
Ground Meats	160 °F (71.1 °C)
Ground Poultry	165 °F
Ham, fresh or smoked (uncooked)	145 °F (62.8 °C) and allow to rest for at least 3 minutes
Fully Cooked Ham (to reheat)	Reheat cooked hams packaged in USDA-inspected plants to 140 °F (60 °C) and all others to 165 °F (73.9 °C).
All Poultry (breasts, whole bird, legs, thighs, wings, ground poultry, giblets, and stuffing)	165 °F (73.9 °C)
Eggs	160 °F (71.1 °C)
Fish & Shellfish	145 °F (62.8 °C)
Leftovers	165 °F (73.9 °C)
Casseroles	165 °F (73.9 °C)

In terms of the project, these are the temperatures the food must be cooked to if a heated option is chosen. There must be a way to effectively measure the temperature of the food to consider it safe for consumption. For this purpose, a thermistor is introduced into the project. The main purpose of the thermistor is to have its resistance dependent on the temperature it is subjected to [8].

The resistance equation of a thermistor is as follows:

Equation 1 [8] – Thermistor equation

$$R = R_0 e^{B \cdot (1/T - 1/T_0)}$$

With R representing the current resistance, R0 representing the resistance of a known temperature (usually at 25 degrees Celsius), T representing current temperature and T0 representing the known temperature (25 degrees Celsius). B is a constant for the material of the thermistor. The value for B is typically approximately 4000.

Design

Initial Ideas and Designs

From the problem statement, a list of possible ideas was generated:

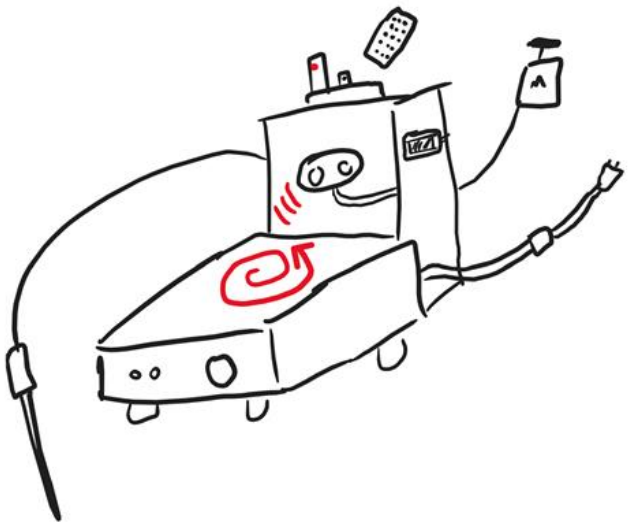
Heating element	Like a kitchen stove, but can be automated so it so it comes on at a specific time that best suits the user's schedule (can make eggs on command). Could also put a heat limit on it to prevent overcooking and/or burning down of the users' house. Easy to implement and can allow for a lot of different sensors, but safety is a big factor. An additional internal stirring component to stir eggs/flip bacon to ensure no burn
Coffee Maker	Automate a coffee maker to have it heat up water at a specific time or when the user turns on the device. Could also pour the liquid into a cup and/or add milk and sugar. Could have a relay to an actual coffee machine or can use nichrome wire to heat up the liquid. This comes with a safety risk, however [9]. Can also use to make tea.
Cereal Pourer	Have a cavity to hold cereal and when the user chooses, it can open a trap door to pour the food into a bowl. Can also have a milk container to pour milk into the cereal. Easiest to implement and could allow for different snacks to be held. Is also very safe.
Toaster	Can control a toaster to cook some toast. Could possibly be on a relay, so the toaster still gets the power from the wall socket. Very versatile, since one can put more than just bread in the toaster, but the use of high voltage (through relay) comes with a safety issue. Is also fast: the user can turn it on and walk away until it's ready. But a regular toaster does the same thing.
Heating element in the coffee cup:	Prevents user injury since the coil is less accessible contained in a cup than out in the open. Also effectively heats up the coffee.

Turning on an actual oven:	Instead of having a small version of the heating coil, why not just connect a relay to a household oven and hope the house doesn't burn down in the process. Not safe at all, takes more than 5 minutes to cook stuff, but it is versatile and heats up food.
Heat Bulb:	Inspired by my pet lizard, we could turn on a heat bulb to warm the food instead of a heating coil [10]. The only issue with this, is that they usually consume a lot of power and require a great voltage to function [11]. It also would take longer to cook food, but the user could walk away and come back.
Call Uber eats:	Have an order placed in the morning so that the user's favorite food is on the doorstep by the time they leave for school. Relies too heavily on the uber driver's schedule.
Drink dispenser:	Acts like the coffee dispenser, but for other kinds of drinks. The only issue comes with having to refrigerate the drinks in between uses. The user can pour them in before every use, but they could just pour it straight into a glass instead.

From this list, three of the best contenders were chosen and analyzed more in depth:

Heating Element

Figure 3 – Stovetop



First up is the automated hotplate. This idea uses a nichrome wire coil to heat up the food. Assuming a 10-ohm resistance per foot [12], and a 9v power supply, with 2 feet of coil, a current of 0.45A is expected [13]. This will heat up the coil to approximately 300 degrees Fahrenheit (Figure 5 – Nichrome wire Current-Temp datasheet), which is more than enough to cook an egg or heat a liquid. The safety of this device is most concerning due to the heat being produced by the coil. A procedure must be implemented to prevent users burning themselves. This can be done by having a cover over the element or trusting them to not burn themselves like an actual stove. If the 32-gauge nichrome wire doesn't heat up the contents fast enough, a pre-built burner can be used that runs through the relay and into the wall for 120V [14]. There is more confidence in this option since it uses more power and is more likely to heat up the food. There can also be a DC motor used to stir the contents to ensure even cooking of a liquid. A mixer can be glued onto motor to allow the mixing of the liquid and have a potentiometer control the speed of the mixer. A thermistor can be used to sense the temperature of the food and close the relay to turn off the burner when it reaches the safe internal temperature (Figure 2 – USDA internal temp table). To turn on/off the device, a remote/infrared sensor combo can be used. This acts as a kill switch, meaning if the food starts burning and the user wants to turn it off, the user can just hit the remote instead of running over to turn it off.

The following block diagram corresponds with the hotplate:

Figure 4 – Stovetop block diagram

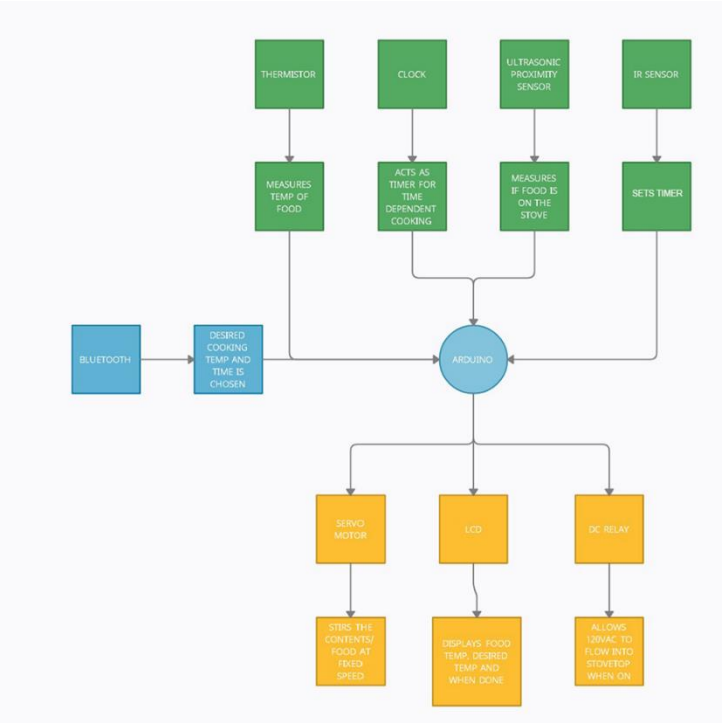


Figure 5 – Nichrome wire Current-Temp datasheet

Approximate current (A) to heat a straight oxidized wire to a given temperature[3]

AWG	DIA-IN	400°F	1000°F	2000°F
8	.128	22.4	52	128
10	.102	16.2	37.5	92
12	.081	11.6	26.5	65
22	.0253	2.9	5.6	12.5
32	.0080	0.68	1.36	2.76
40	.0031	0.24	0.43	0.79

The following table shows the expected costs of the stovetop:

Table 2 – Stovetop costs

Stovetop Costs	
Stovetop	*$30 (Amazon)*
Arduino and Sensors	*$60 (Electronic Shop)*
BBQ probe	*$20 (Traeger Grills)*
Grapple Bar	*$10 (Home Hardware)*
Total	*$120*

Coffee Maker

Figure 6 – Coffee machine

Can use the same mechanism as the heating element to heat up water for use in a coffee to make the device compact and versatile. This idea probably will not be chosen, considering there is such a thing as automated coffee makers. They can be turned on and come back to later when the drinks are done. If this option is pursued, there could be the coil contained in the cup to prevent user injury and have a few sensors that sense the liquid temperature and set the desired temperature. A dc motor can also be used to stir the liquid to ensure even heating. As for controlling it, a potentiometer can be used to set the turn-off temperature that the thermistor will seek.

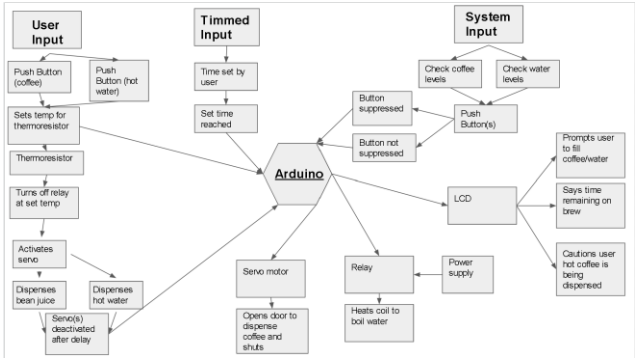
The following table displays the costs of the coffee machine:

Table 3 – Coffee machine costs

Coffee Machine Costs	
Coffee Machine	$50 (Amazon)
Arduino and Sensors	$60 (Electronic Shop)
Total	$110

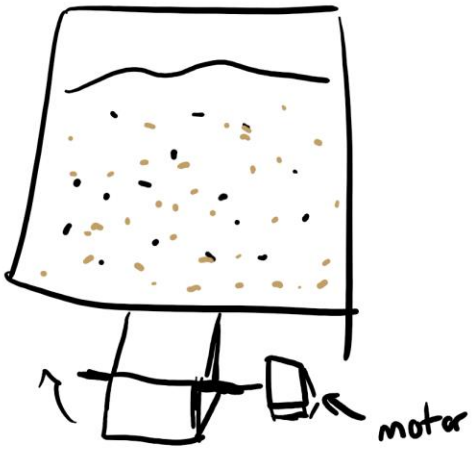
The following figure is the proposed block diagram for the coffee maker:

Figure 7 – Coffee machine block diagram



Cereal Dispenser

Figure 8 – Cereal dispenser



Lastly is the proposed idea for a cereal dispenser. In this design, the Arduino controls a trapdoor that dispenses a type of snack. This can be cereal, or oatmeal, or even M&M's. For the actual container, a cereal dispenser can be used and have the Arduino turn the dial that opens the trapdoor. One can be found on amazon for $25 [15]. This option does not allow much room for actuators and sensors, although it is much safer than using a 120V kitchen top burner. A servo motor can be used to open and close the trapdoor. Other than that, there is not much to be with this option. A potentiometer can be used to control cereal drop rate and volume, and a button can be used to turn the machine on. A light can also come on whenever the cereal is ready, but it's redundant since the user can obviously see the cereal is poured.

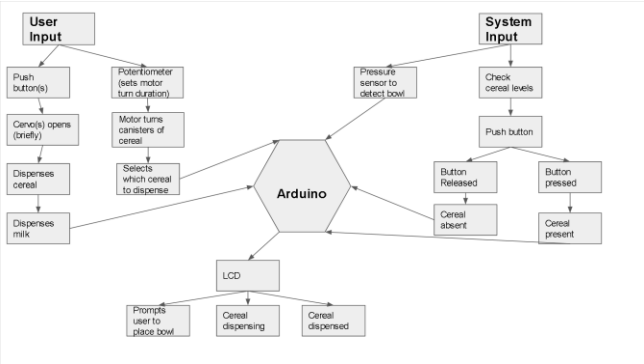
The following table displays expected costs associated with the cereal dispenser:

Table 4 – Cereal dispenser costs

Cereal Dispenser Costs	
Cereal Containers	$40 (Amazon)
Arduino and Sensors	$60 (Electronic Shop)
Total	$100

The following figure represents the block diagram for the proposed cereal dispenser:

Figure 9 Cereal dispenser block diagram



From the three proposed ideas, a fit can be done with the constraints in the introduction:

Figure 10 Constraint Fitting

Primary Criteria

	5 mins	Safe	Versatile	Straightforward	Indicator	Consistent	Immediately Eat
Stovetop	4	2	5	3	5	5	5
Coffee	5	4	1	5	5	5	2
Cereal Dis.	5*	5	3	5	5	5	5

* Requires longer for eating since in a bowl

Stovetop - 29
Coffee - 27
Cereal Dispenser - 33** (**Doesn't have enough potential for sensors)

Although the cereal dispenser was the winner in terms of safety and ease, the stovetop control design was ultimately chosen solely based on the potential for sensors and actuators, since our design needs to meet a strict sensor and actuator limit constraint.

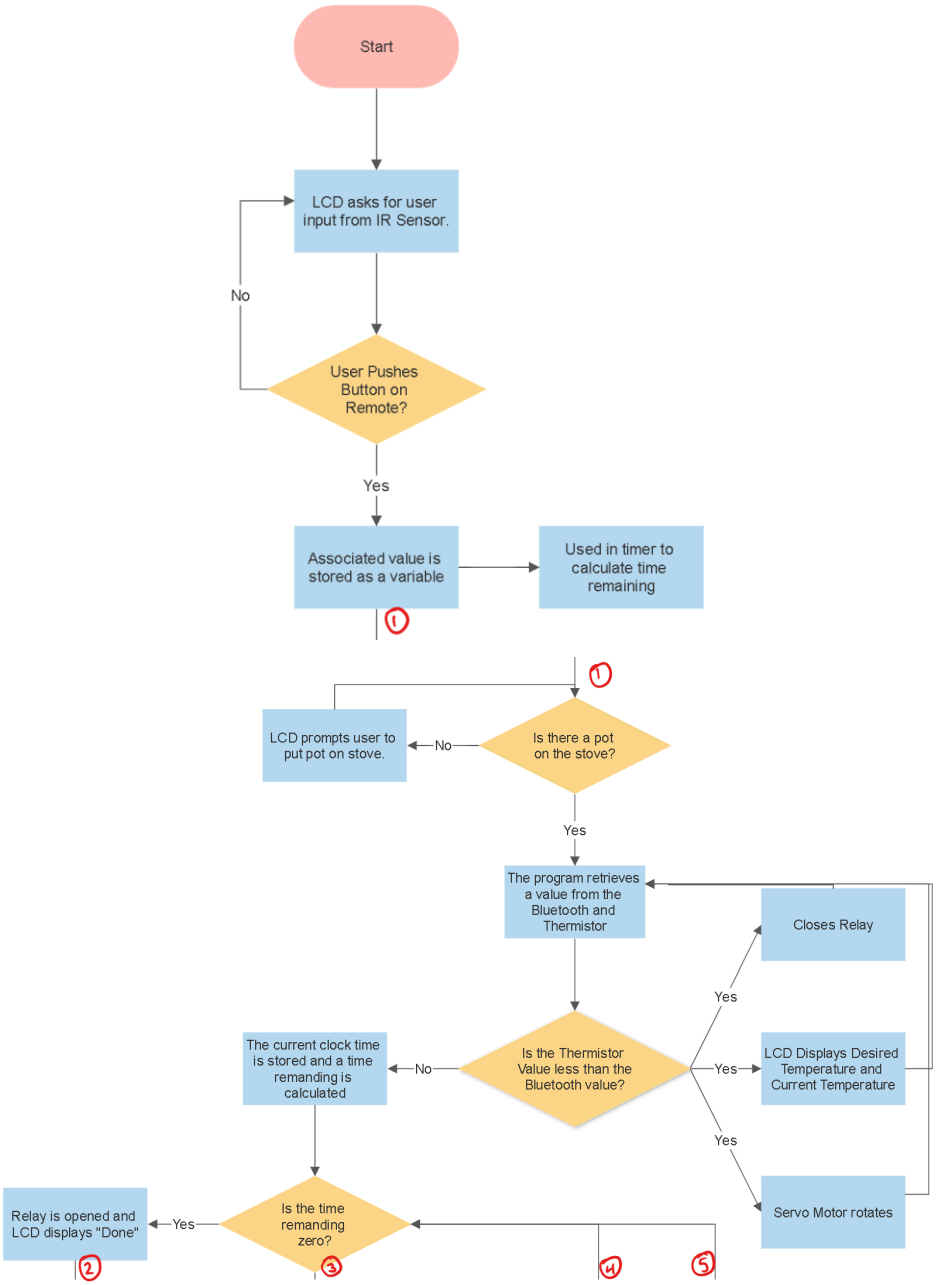
The Automated Stovetop

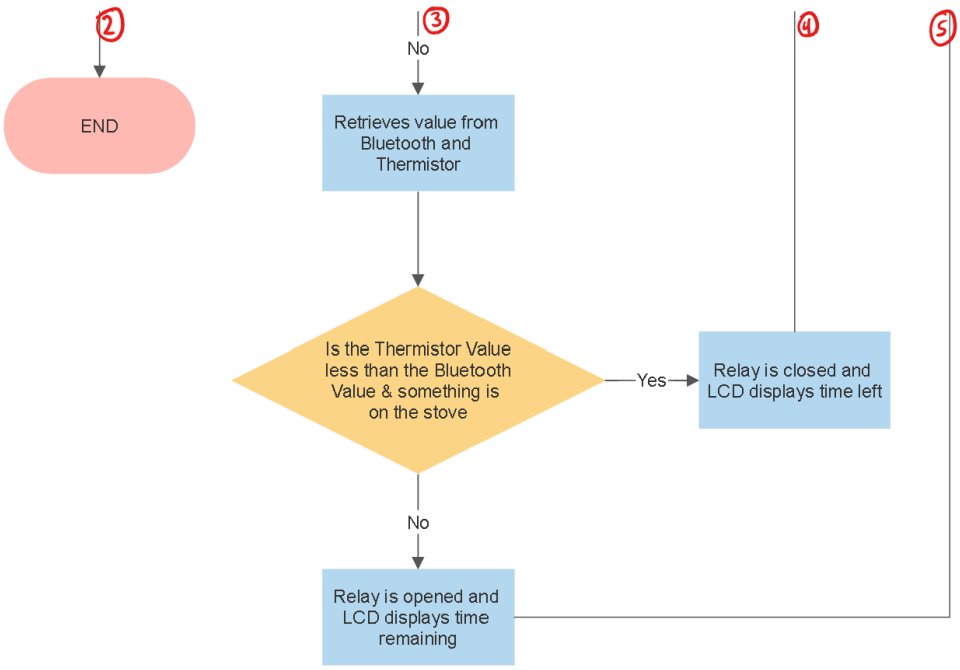
Now that a design is chosen, a more thorough design is conducted.

Algorithm

The following algorithm is created for the stovetop control:

Figure 11 – Algorithm



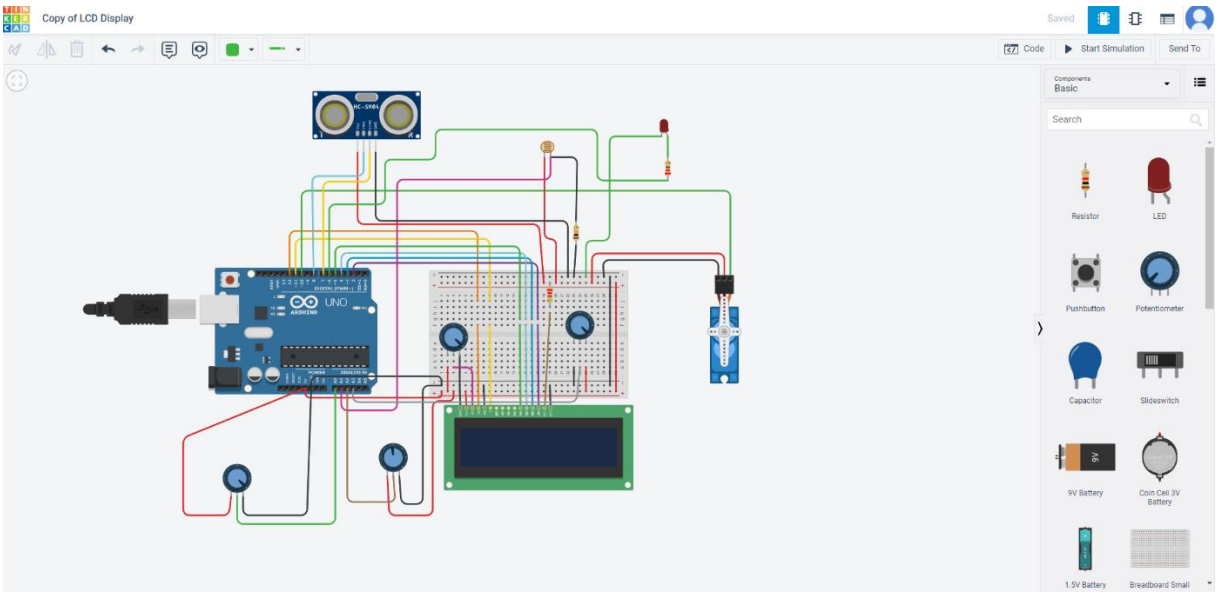


Tinkercad

Using this algorithm, a Tinkercad circuit is constructed:

https://www.tinkercad.com/things/76GQFS7oXG5-lderry-mdauphinais-project-tinkercad/editel?sharecode=XALHYcvF_EiGUVBgtBjbRG6hdSfwsuYtlIRcBsxsu8s

Figure 12 - Tinkercad



This circuit is broken up into a series of six subsystems. These systems control the Servo motor mixing arm, the proximity sensor, the desired temperature and thermistor, the LCD, the IR sensor, and timing circuit.

Firstly, the Servo motor is controlled directly from the Arduino. This output is to be controlled by the Bluetooth module (Substituted in simulation for potentiometer due to limited components in Tinkercad) also connected to the Arduino. A user will be able to adjust the speed of the mixing arm from the associated app on their phone, through the Bluetooth module.

Furthermore, the proximity sensor encompasses the safety objective of the project by detecting the presence of cookware on the heating element. This system only passes current through a DC relay (represented by an LED) to the heating element. This means the heating element will never simultaneously be open and powered, being a potential fire hazard. This component is connected directly to the Arduino which then controls the output signal of the sensor and activates/deactivates the DC relay.

Moreover, the thermistor (represented by the photoresistor) is connected to the analog input of the Arduino. This sensor is representative of a temperature probe to be used in the final design. This probe monitors the temperature of the food on the cooking surface. This signal is then processed by the Arduino to the Servo motor, a timing circuit (clock) and DC relay. The user sets the desired temperature from the corresponding Bluetooth app on their phone, to which the thermistor then checks the temperature of the food. Once the current temperature matches the desired temperature, either the clock circuit is activated (cooking the food for a certain duration) or the DC relay and LCD are tripped, turning off the element and indicating that the food is ready.

The IR sensor is not used in the Tinkercad simulation due to the limitations of the program. There is little reason to have the sensor without the clock module due to them depending on each other.

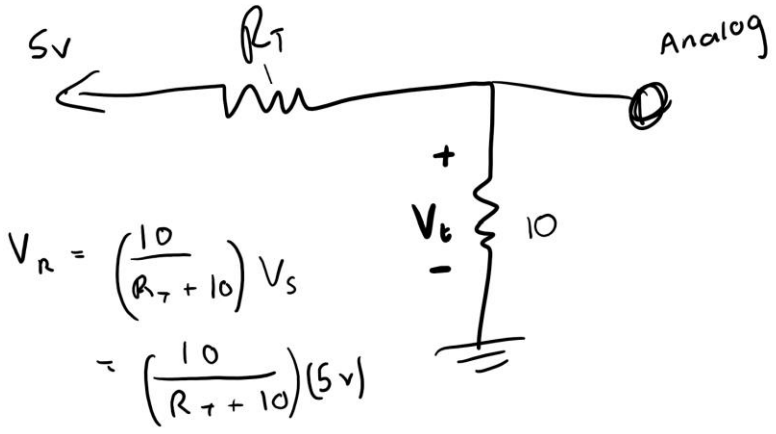
Sensors and Actuators

There are many different possibilities for how this device can be approached. The main components are the relay to turn on the stove, thermistor to measure the real-time temperature, some sort of sensor to set the desired cooking temperature, a clock to offer a timer, a display to communicate to the user and a safety feature that only turns on the stove when there is no chance for injury. Below are the choices made in terms of components and a brief explanation of what they do in the circuit, as well as how they are implemented physically.

Thermistor

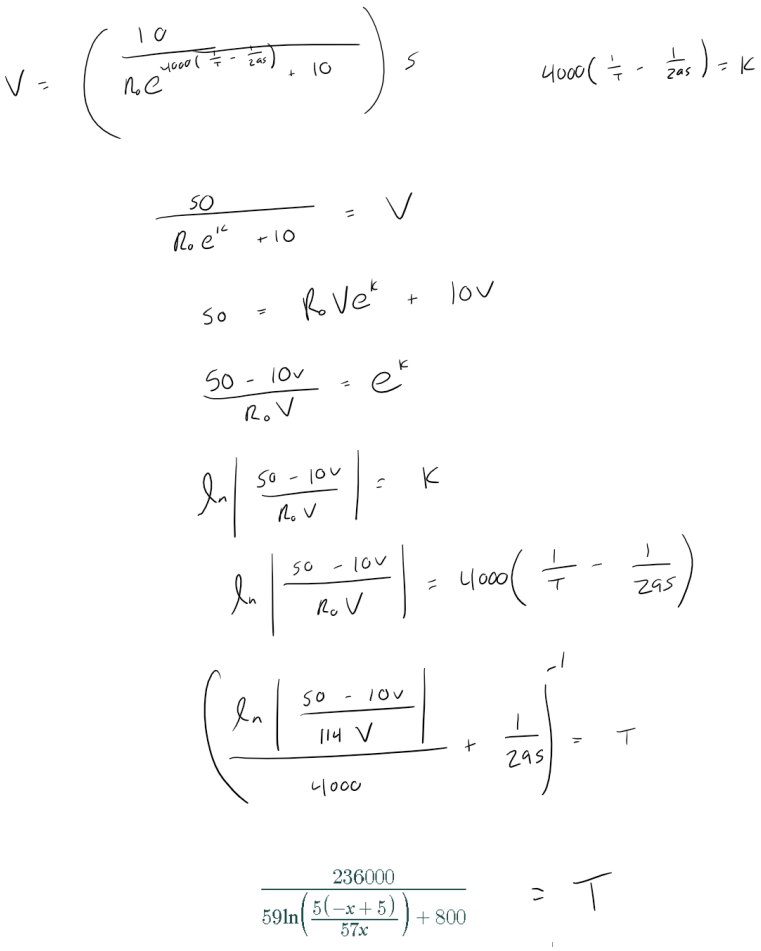
The main purpose of the thermistor is to measure the temperature of the food being cooked and compare this value to the desired temperature returned from the Bluetooth module. To properly measure the temperature, the temperature must be converted to a signal the Arduino can effectively use. Using the resistance-temperature characteristics described in the literature review section, a voltage divider circuit can be created where the voltage being read by the Arduino is the voltage across a 10kOhm resistor:

Figure 13 – Voltage Divider



The thermistor equation (equation 1) can be subbed into the voltage divider equation as Rt:

Figure 14 Temp Probe Equation



This equation is used in code to convert the measured voltage to a temperature in Kelvin. This temperature is then converted to Fahrenheit for display on the LCD screen. This portion of the code looks like:

Figure 15 – Temp Probe Code

```
  double tempProbeValue = analogRead(A1);
  double tempProbeVoltage = tempProbeValue * (5.0/1023.0);
double tempProbe = 236000/(59*log((5*(5-tempProbeVoltage))/(57*tempProbeVoltage))+800);
tempProbe = tempProbe*1.8 - 459.67;
```

Use in code: The value tempProbe is always being compared to the desired temperature value; the stove is turned on when tempProbe<desiredTemp.

Bluetooth Module

Figure 16 [[16] – Bluetooth Module

The main purpose of the Bluetooth module is to collect user input to set a desired cooking temp for the food. The chip used is the HM10. This choice is made due to Apple products being compatible with it. To simplify the process, an app called Dabble was used [17]. An important thing to note is that to use specific tools, a specific library is required. In this case, the Dabble.h library is required to use the Bluetooth module (Figure 17 – Dabble library). The below code is used to include the required library:

Figure 17 – Dabble library

```
#include <Dabble.h>
```

To use the dabble app, the following code must be used:

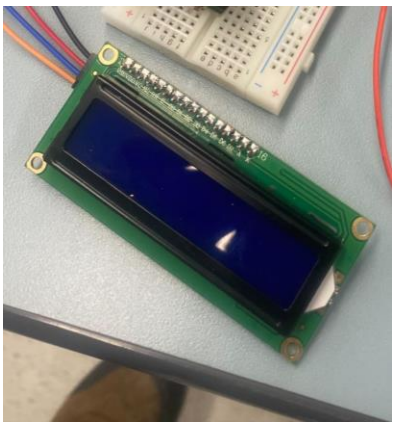
Figure 18 – Bluetooth code

```
Dabble.processInput();  // checks bluetooth input to see temp
value = (int)LedControl.readBrightness();
temp = value*4;



while(ultrasonicValue<30 && tempProbe<temp){
 digitalWrite(6, HIGH);
```

The processInput() function reads the input from the app and translates it to a language the rest of the code can use. This input is then casted to int and used to compare to the probe temperature.

LCD Screen

Figure 19 – LCD Screen



The LCD screen is the main communication actuator. It displays the real-time temperature as well as the temperature set by the user through the Bluetooth module. To use the LCD, the proper library is required:

Figure 20 – LCD code

```
#include <LiquidCrystal.h>

#include <LiquidCrystal_I2C.h>
```

These libraries offer streamlined functions that make using the LCD easy.

Through testing, it was found that the LCD occupied the 0x27 location. To initialize the LCD, the following code must be used:

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

As well as the following in the setup() function:

```
lcd.begin();
lcd.backlight();
```

Once the LCD is turned on and initialized it is relatively straightforward. To display a value on the screen, one must call the lcd.print() function. For the purposes of this device, the current probe temperature as well as the desired temperature will be displayed for the user initially, with the countdown being displayed once the food reaches temperature.

Servo

The servo is used to stir the contents of the bowl or pot if needed. It is on as long as the device is on, even if the stove itself is turned off. To properly use the component, one must initialize the servo using the following code before the setup:

```
Servo myservo;

double angle;
int wait;
int pos = 0;
```

With the following code in the setup function to start the component:

```
myservo.attach(10);
```

To cause the servo motor to move, one must call the function:

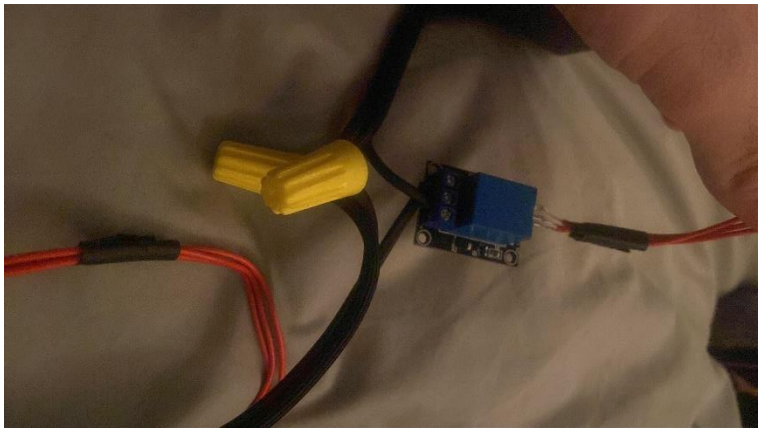
```
myservo.write(0);
```

with the parameter being the angle at which the servo will turn (in degrees). The servo has a range of 0 to 359, but for the device's purposes, 0 to 180 degrees range is used. It is also important to include a delay to allow the servo to fully rotate, otherwise, it will begin twitching.

The servo motor will have an arm that is placed in the water or other liquid and will rotate to stir it. This can be used in stews or pasta to evenly cook the ingredients.

Relay

Figure 21 - Relay



The relay is the most vital part of the device. It is responsible for suppling power to the stovetop. The relay itself acts like an LED in the sense that all is required for operation is the code:

```
digitalWrite(6, HIGH);
```

(or digitalWrite(6, LOW); if turned off)

with an initialization code:

```
pinMode(6, OUTPUT);
```

In the final code, the relay turns on when a set of conditions are met (the probe temp is less than the desired temp and the ultrasonic sensor senses an object on the stovetop). Otherwise, the relay is opened to ensure safety is maintained to go with our constraints.

To set up the hardware, the power cord of the stove was stripped. One of the active leads was connected in series with the relay inputs. The other two leads were mended back together using Marrettes. There are also four-foot leads going from the Arduino to the relay to keep the board as far from the heat as possible.

Ultrasonic Sensor

The main purpose of the ultrasonic sensor is to sense if there is a pot on the stove. The value returned from the ultrasonic function is used in a conditional statement that turns on the stove. To code the ultrasonic sensor, a pre-written function was used to read the sensor and return the distance in centimeters [18].

Figure 22 – Ultrasonic Code

```
int distanceThreshold = 0;
int cm = 0;
int inches = 0;
```

```
long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave travel time in microseconds
  return pulseIn(echoPin, HIGH);
}
```

To call the function in the main code, the following code is required:

```
long ultrasonicValue = 0.01723 * readUltrasonicDistance(8, 7);
```

The factor 0.01723 converts the function from seconds to centimeters, since the ultrasonic sensor returns the time it took between sending the pulse and retrieving it. The returned value is compared to a value of 30. If the object is less than 30cm (1 foot) from the sensor, then the stove will turn on.

IR Sensor

To start the device, an initial input is required. It was decided that the IR sensor will receive a code from the remote to begin the program. The received value is the timer value (the one that will get added to the current time to determine the remaining time). Using the Serial monitor, the following codes were determined for each remote button:

Table 5 – IR Testing Values

Number Pushed on remote	**Corresponding Double Value**
0	16738455
1	16724175
2	16718055
3	16743045
4	16716015
5	16726215
6	16734885
7	16728765
8	16730805
9	16732845

The codes are then converted to minute values. Push 0 and it returns 0 minutes, 1 for 1 minute, etc. These values are summed with the start time in the countdown phase to get a end time. The current time is compared to the end time to return the time remaining, which is displayed on the LCD.

Figure 23 – Setup Code

```
      case 16738455:
      clockValue = 1;
      break;

      case 16724175: //when you press the 1 button
        clockValue = 60;
        break;
      case 16718055: //when you press the 4 button
       clockValue = 120;
        break;
       case 16743045: //when you press the 2 button
        clockValue = 180;
        break;
       case 16716015: //when you press the 5 button
        clockValue = 240;
        break;
       case 16726215: //when you press the 3 button
        clockValue = 300;
        break;
       case 16734885: //when you press the 6 button
        clockValue = 360;
        break;
        case 16728765:
        clockValue = 420;
        break;
        case 16730805:
        clockValue = 480;
        break;
        case 16732845:
        clockValue = 540;
        break;
    }
    //Serial.println(clockValue);

  }
  delay(10);
```

Clock

The clock is a vital component in the backend of the program timeline. It is what runs the timer and eventually turns off the stove when the timer reaches 0. The RtcDS1302.h library is required to run the clock.

The following code is used to create and use the timer. The IR sends a clock value that represents how many minutes the timer will run for. When the temperature of the probe reaches the desired temperature, the clock returns a start time. The desired time will be start time summed with the clock value. The time left is equal to the desired time minus the current time. When the time left reaches 0, the food is done, and the stove turns off.

```
#include <RtcDS1302.h>
#define countof(a) (sizeof(a) / sizeof(a[0]))
ThreeWire myWire(12,11,13); // IO, SCLK, CE
RtcDS1302<ThreeWire> Rtc(myWire);
#include <IRremote.h>
```

```
IRrecv irrecv(IR_Recv);
decode_results results;
int clockValue = -1;
```

```
void setup() {
```

```
  lcd.begin();          // starts LCD
  lcd.backlight();
  myservo.attach(10);   // starts servo
  Serial.begin(9600);
  pinMode(6, OUTPUT);   // assigns relay as output
  pinMode(5, OUTPUT);   // assigns LED as output
  Dabble.begin(9600);   // starts bluetooth module
  Rtc.Begin();
  irrecv.enableIRIn();
  // Starts the receiver

   lcd.setCursor(0,0);
  lcd.print("Ready for Input");
  delay(10);
  Serial.println("Ready for Input");
while (clockValue == -1){
    if (irrecv.decode(&results)){
    long int decCode = results.value;
    Serial.println(results.value);
```

```
    //switch case to use the selected remote control button
    switch (results.value){
```

```
  ultrasonicValue = 0.01723 * readUltrasonicDistance(8, 7);
  int startTime = (int)Rtc.GetDateTime();
  int desiredTime = (int)startTime+clockValue;
  int now = (int)Rtc.GetDateTime();
  double timeLeft = desiredTime-now;
  tempProbeValue = analogRead(A1);
  tempProbeVoltage = tempProbeValue * (5.0/1023.0);
  tempProbe = 236000/(59*log((5*(5-tempProbeVoltage))/(57*tempProbeVoltage))+800);
  Dabble.processInput();  // checks bluetooth input to see temp
  value = (int)LedControl.readBrightness();
  temp = value*4;
```

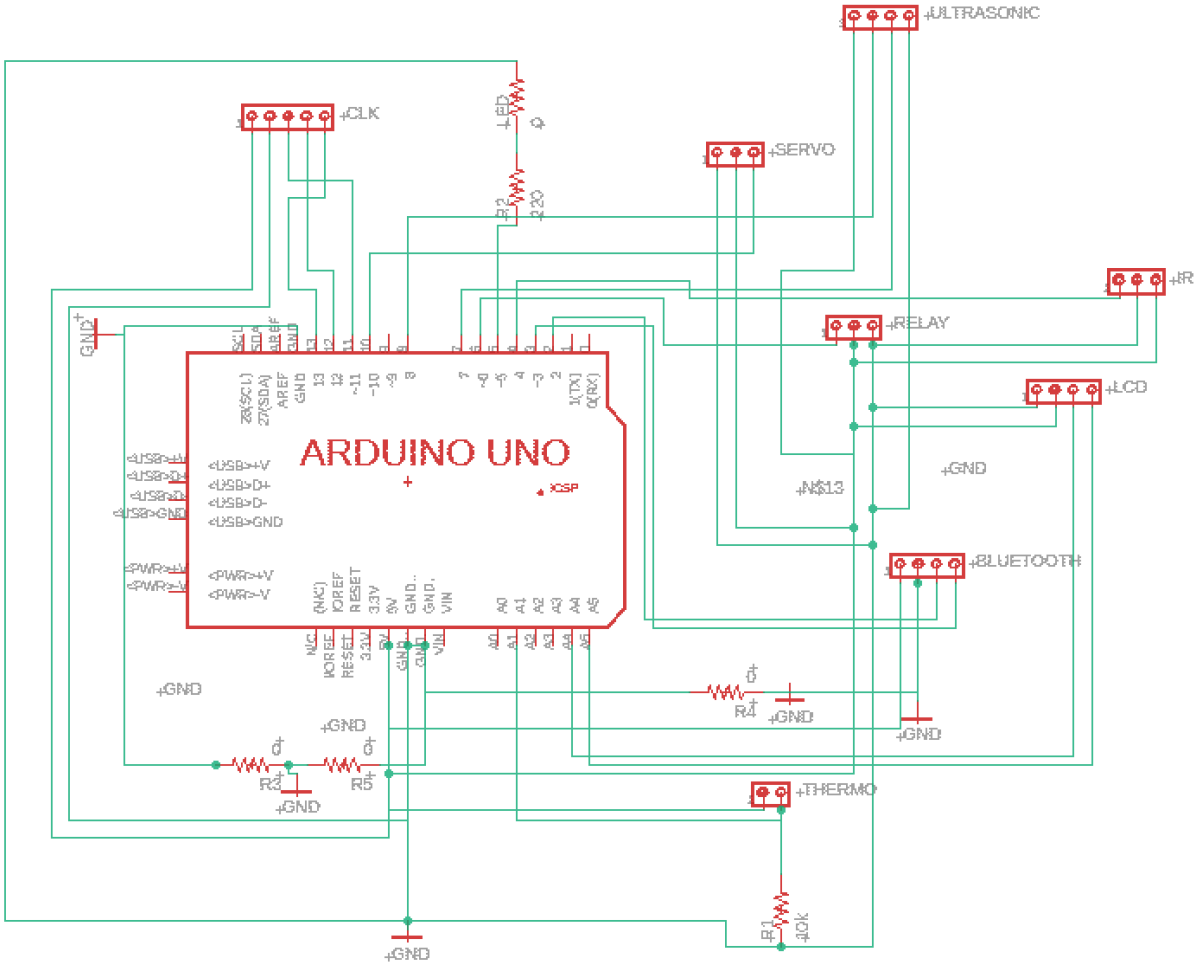
```
while(now<=desiredTime){
  int now = (int)Rtc.GetDateTime();
  Serial.println(now);
        double timeLeft = (double)desiredTime-now;
        Dabble.processInput();  // checks bluetooth input to see temp
        value = (int)LedControl.readBrightness();
        temp = value*4;

        myservo.write(0);
        delay(10);
        tempProbeValue = analogRead(A1);
        tempProbeVoltage = tempProbeValue * (5.0/1023.0);
        tempProbe = 236000/(59*log((5*(5-tempProbeVoltage))/(57*tempProbeVoltage))+800);
        tempProbe = tempProbe*1.8 - 459.67;
        ultrasonicValue = 0.01723 * readUltrasonicDistance(8, 7);
        Serial.println(timeLeft);
```

```
    case 16738455:
    clockValue = 1;
    break;

    case 16724175: //when you press the 1 button
      clockValue = 60;
      break;
    case 16718055: //when you press the 4 button
     clockValue = 120;
      break;
     case 16743045: //when you press the 2 button
      clockValue = 180;
      break;
     case 16716015: //when you press the 5 button
      clockValue = 240;
      break;
     case 16726215: //when you press the 3 button
      clockValue = 300;
      break;
     case 16734885: //when you press the 6 button
      clockValue = 360;
      break;
      case 16728765:
      clockValue = 420;
      break;
      case 16730805:
      clockValue = 480;
      break;
      case 16732845:
      clockValue = 540;
      break;
  }
  //Serial.println(clockValue);

}
delay(10);
```

EAGLE PCB

Below is the created schematic on EAGLE:

Figure 24 - Schematic



IMPORTANT THINGS TO NOTE ABOUT THE SCHEMATIC:

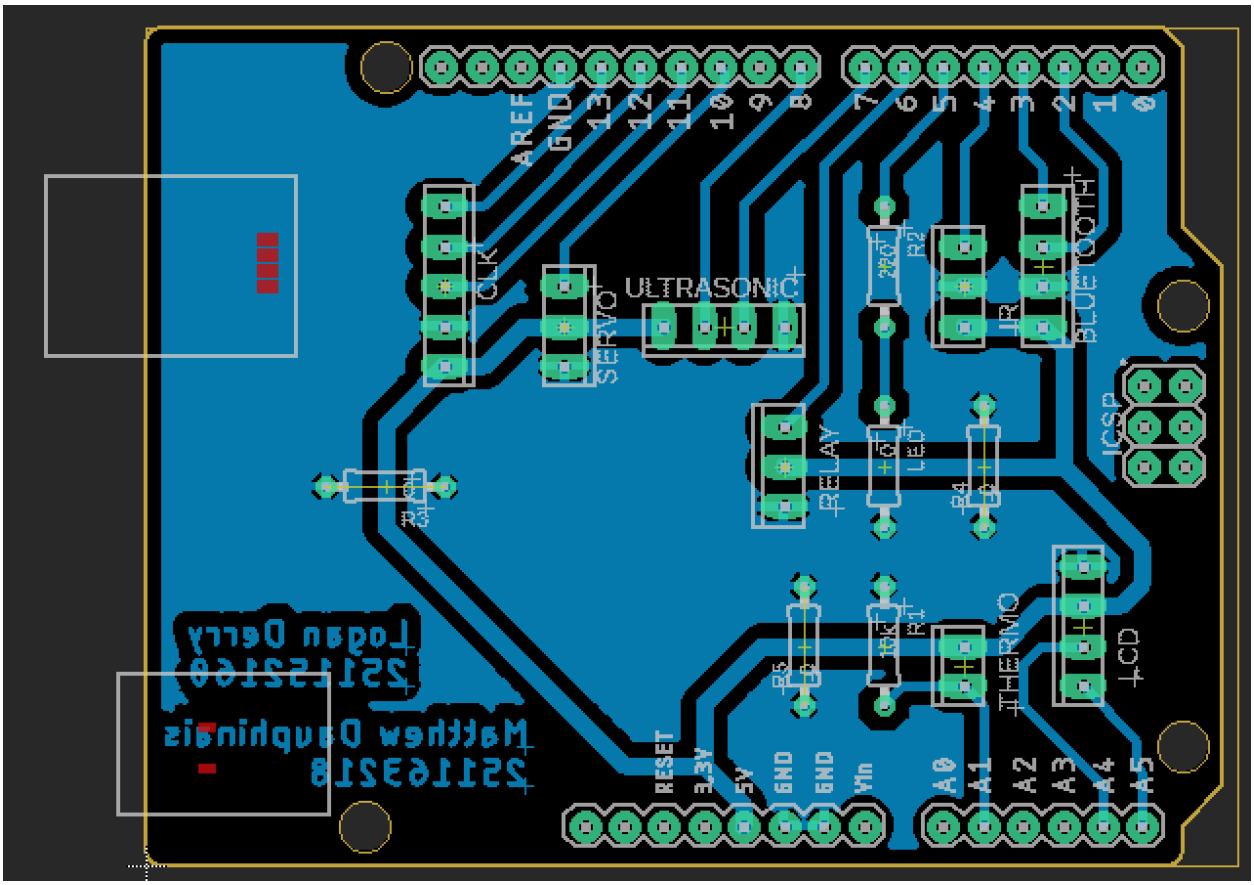
The MTA pin headers to attach the wires of the different components this was done to allow quick removal of components since they are not directly soldered to the board.

All the Arduino GND pins were grounded to reduce the impedance of having one ground.

The Arduino shield library was downloaded to get the Arduino component.

- Made the final board from this schematic:

Figure 25 – Board Diagram



Things to note about board

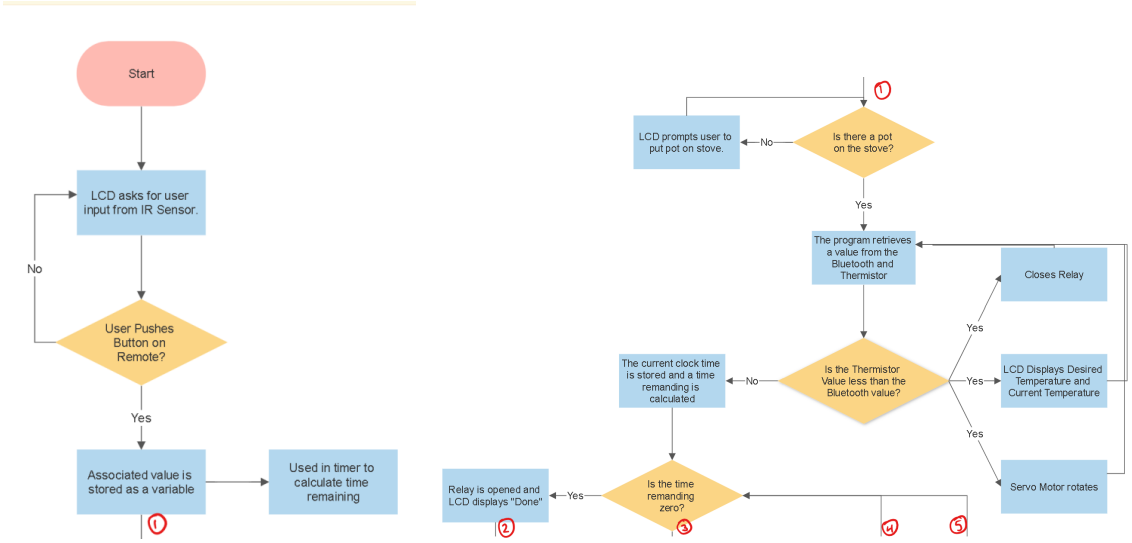
To reach all the ground pins, dummy resistors as placeholders were required for jumper wires to connect the different ground planes. Initially, the header pins crossed to make the board “pretty” but had to make sure the pins were the same order as on the different components.

Made the 5V power line 0.04” instead to allow for a greater current to flow. There was also an attempt to orient all the pin headers to have GND facing the centre of the middle plate. Could not do for all components, so had to install those jumper wires to connect the multiple ground plates.

Implementation

The final design of this project uses each of the integrated sensors and actuators to create a breakfast meal that can be started remotely and be prepared in under five minutes. This system initiates with the IR sensor, connected to the powered Arduino, looking for a start-up signal from the coupled IR remote (Table 5 – IR Testing Values). Once this signal is detected, a clock value for use in the timer is determined. The microcontroller then receives a value from the ultrasonic sensor. If there is a pot on the stove the code looks to the temperature sensor and Bluetooth for a conditional statement that controls the relay.

Figure 26 – Algorithm



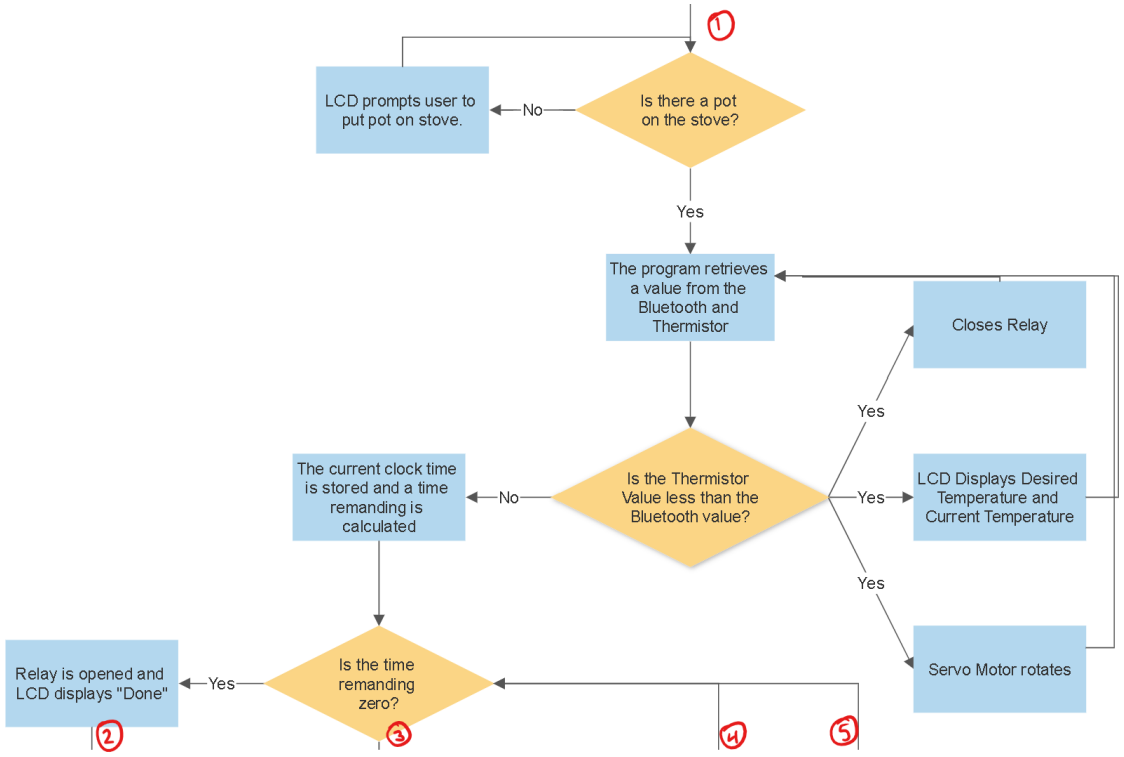
If any of the conditions ate not met, the Arduino continues to keep the device powered and will continue looking for the correct user inputs. The conditional statement is as follows:

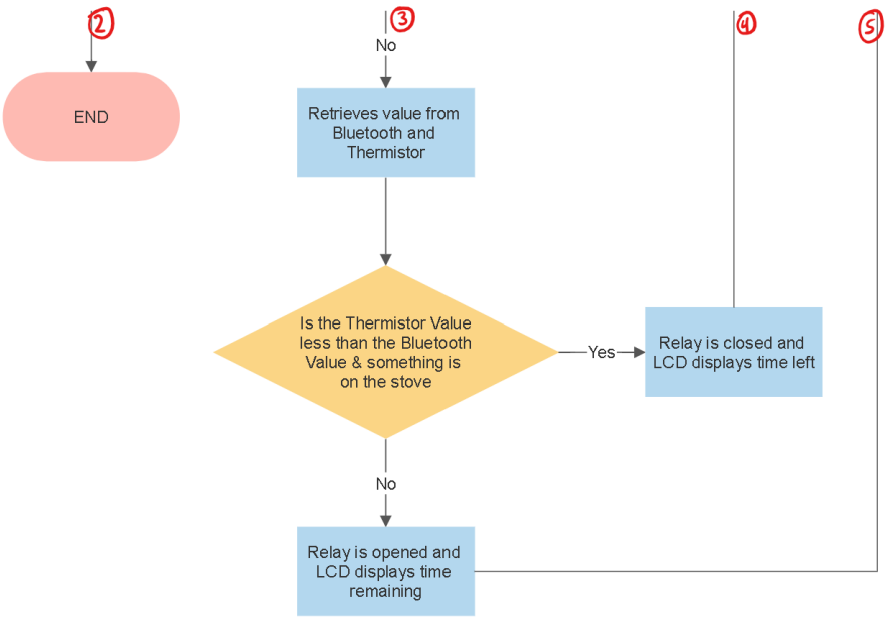
```
while(ultrasonicValue<30 && tempProbe<temp){
 digitalWrite(6, HIGH);
 digitalWrite(5, HIGH);
 Dabble.processInput();  // checks bluetooth input to see temp
 value = (int)LedControl.readBrightness();
 temp = value*4;
```

This is done as a safety feature to ensure that cooking element is only powered while covered and cannot be turned on another way. This specific proximity sensor was provided as per guidance of both the ECE2242 Instructors and Electronics Shop staff.

Moreover, if an IR signal is detected and the proximity sensor determines that the cooktop is covered, the Arduino will seek input from the Bluetooth module for the desired cooking temperature. Once this value is set, a signal is passed from the Arduino to the LCD. This actuator then displays the food's current temperature, and the desired temperature set by the user. This is done to allow users to see the current temperature of their food. This lets users approximate the progress of the cook but while also allowing the user to adjust the temperature of the element accurately. Furthermore, when the relay is closed (lets current flow), the servo motor rotates to stir the contents. A Servo motor was selected for this project as opposed to a DC motor to manage the stir speed of the mixing arm. The Servo motor contains a gearbox that allows for precise movement and operation at lower speeds. The DC motor used in the project's preliminary phases cannot operate reliably at lower speeds and raised concerns about splashing the cooktop's contents at higher speeds. This resulted the change from DC motor to the selected Servo motor.

Figure 27: Algorithm end





If a cook time has been set using the IR sensor, once the thermistor reaches the desired temperature as selected by the user, signal is passed from the Arduino to a clock actuator which begins to count. As soon as the probe temperature reaches the desired temperature, the real time clock returns a value. The desired time (when the food is done) is calculated by summing the current time with the IR sensor clock value. When the counted value of the clock (in seconds) matches the desired cook time, signal is sent to the Arduino which then turns off the Servo responsible for the stirring arm, as well as the relay controlling the stove. The LCD is updated to read that the food/drink is ready to eat. It was decided to have the clock actuator count after the food has reached temperature as opposed to counting starting as the element is powered. This was done to ensure consistency between cooks, as certain temperatures will take longer to reach than others, which would cause confusion in selecting cook times. Foods that require no additional cook time once reaching temperature are dealt with the zero button on the IR Sensor when initializing (Table 5 – IR Testing Values).

As the temperature of the food increases this is detected by the thermistor. As the temperature increases or decreases, the input signal from the thermistor is sent to the Arduino and processed to update the LCD with the current temperature as seen by the sensor. Once the desired temperature matches the temperature being measured by the thermistor sensor, this signal is read by the Arduino and the DC relay is shut off (if no cook time has been set). This shut off function will make the device safer, contributing to the corresponding design constraint.

All the shut off functions have been built into the systems code in high priority, such that if at any time an off signal is detected from either the proximity sensor or Bluetooth sensor, the entire system will shut off instantly. This is done as a safety feature, as shutoffs are required to make this device safe to operate remotely.

```
while(ultrasonicValue<30 && tempProbe<temp){
 digitalWrite(6, HIGH);
```

During the implementation of this design, there was the issue that the selected Bluetooth module (HC06) was not compatible with the developer's model of phone. To address this issue, a new model of Bluetooth sensor was selected that was compatible with a larger number of phone models (HM10). Furthermore, the design team also encountered the issue that the DC motor included in the design during its preliminary phases, would rotate too quickly for the cooktop, causing potentially hazardous spillage during operation of the mixing arm. This problem was addressed by replacing the DC motor in favour of the Servo motor. The Servo motor contains a gearbox allowing for precision operation under low speed, creating a solution to the issues caused by the DC motor.

It was also found that deriving the equation for the thermistor proved difficult and required three tries to eventually get right. To debug the thermistor, the use of a multimeter was used to measure the voltage going into the analog pin. The measured voltage was not consistent with the equation. Eventually, the correct equation was found. The result is an accurate temperature sensor that effectively returns the real time temperature value of the food being prepared.

Results and Evaluation

To test this design, it first began by testing each individual subsystem.

Servo Motor

Firstly, the Servo motor was tested for its ability to rotate on full range at variable speeds. This subsystem was tested first in Tinker CAD, in which the Servo was able to rotate fully at variable speed when attached to a potentiometer (no Bluetooth module in Tinkercad). This system was tested again with physical components, in which the Servo behaved the same as was simulated.

LCD & Thermistor

The LCD actuator was then tested alongside the temperature probe (thermistor) as these systems work in tandem with each other. The input of the thermistor determines the display message of the

LCD. This was tested in the circuit in (Figure 12 - Tinkercad). The main test for this pair of systems is to test that the thermistor will properly read the temperature, the LCD will display the thermistor and desired temperatures and will display a final message once the temperatures match. These subsystems operated perfectly in simulation and operated as well in practical testing.

Ultrasonic Proximity Sensor & DC Relay & IR Sensor

The proximity sensor was tested next along with the DC relay, also since these systems work in tandem. These were tested first in simulation for their accuracy in detecting an object on the stove element of the design and shutting the relay when the object is removed. This test was also passed with practical components, as the relay would audibly trigger as an object was waved in front of the proximity sensor. Furthermore, the IR sensor was tested in simulation for its ability to produce the same codes consistently with each button press. This test was passed separately in simulation and in physical testing as the remote and sensor pair produced different codes in simulation than in physical components. However, the system still succeeds as the different codes were produced consistently, only requiring minor changes to final code.

These tests are catered towards the physical operation of the systems and their direct results. This type of testing is focused specifically on the outputs of the systems, this leaves room for error in the systems coding or overall wiring. This can lead to unforeseen issues in the project if left as the only system test. However, this is a good test to run before assembling the final circuit to ensure that each subsystem is working as intended before being added to the final circuit.

This design does comply with the design constraints and requirements. The selected final design can prepare a meal in under five minutes, is designed with safety features, involves as few steps for operation as possible, contains an indicator for the user and allows them to eat immediately. The stovetop can be programmed ahead of time by the user, allowing the user to determine their meal preparation time with their schedule (presumably less than five min). Moreover, this also allows the user to eat immediately should they so choose to do so. This device also contains an ultrasonic sensor to ensure the element is never powered and uncovered as a safety feature for its users.

Technical Testing and Debugging

Issue 1: Thermistor gives wrong values

Suspected Problem 1: Equation was translated wrong into code

What was done: Checked over the brackets to see if it matches the equation

Outcome: The syntax was all correct. The equation was correctly translated. Still gives the wrong values.

Suspected Problem 2: The voltage being read from the Analog Pin is across the Thermistor, not the 10kOhm resistor.

What was done: Used the multimeter to measure the voltage being read by the Analog Pin

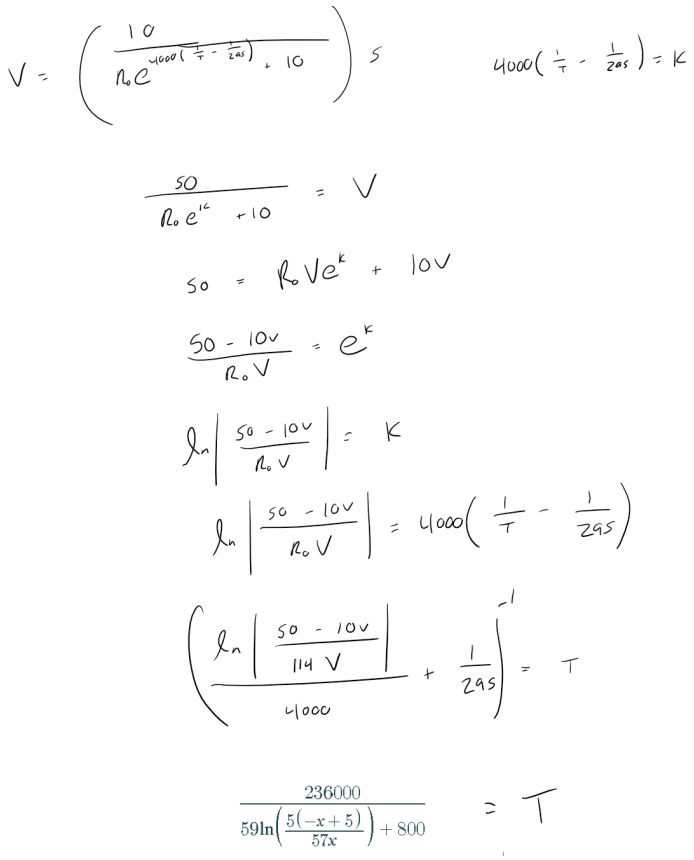
Outcome: The voltage was too low to be across the thermistor, which has a resistance of 100kOhms at room temperature. Should be greater than 4V if across thermistor, was reading 0.5V.

Suspected Problem 3: The equation itself was incorrect.

What was done: A review of the algebra was done to make sure the equation was correct.

Outcome: The equation was, indeed, incorrect. The new equation is as follows:

Figure 28 - Thermistor equation



After being implemented into the code. The LCD was displaying a far more reasonable number of 70 degrees Fahrenheit. After boiling water, it read 212 when the water boiled, which coincides with the boiling temperature of water.

Issue 2: During countdown, the timer was not decrementing

Suspected Problem 1: The function is returning an unusable data type

Actions: Casted the function to be int, so it can be used in the LCD.print function.

Outcome: The counter was still constant, it was not decrementing

Suspected Problem 2: The clock function was getting skipped in the code, so the new clock value is never retrieved and the timeLeft is never calculated.

Actions: Recheck the code and use the Serial monitor to see what value is getting sent to the LCD

Outcome: The clock value was not changing, the code was the exact same as in testing, so it should work effectively. Must be something else.

Suspected Problem 3: The clock component is wired incorrectly

Actions: Check the wires and ensure they are plugged in to the correct pin. Compare to the eagle board diagram.

Outcome: The GND and CLK pins were swapped. Re-connected them in the right configuration. The timeLeft counter worked correctly again.

Issue 3: Bluetooth Not working during final PCB test

Suspected Problem 1: Code was not saved right and had something missing.

Action: Rechecked the code to ensure nothing was wrong,

Outcome: Nothing appeared to be missing.

Suspected Problem 2: Wires are either crossed or unplugged.

Action: Check the wires to see their state.

Outcome: The high-power wire was unplugged. Plugged it back in and it works again.

Issue 4: IR Scanner doesn't return a proper timer value

Suspected Problem 1: The switch is not actually running, instead it skips it.

Action: Add a serial print debug statement

Outcome: The serial prints out the statement, the switch is not skipped

Suspected Problem 2: The remote sends a default value

Action: add a serial print statement to see what is being sent to the device.

Outcome: The serial print reads nothing. Nothing is sent to the device.

Suspected Problem 3: The switch goes through once and starts the program with a clock value of 0.

Action: Add a conditional statement that it repeats the switch until the clock value is not 0

Outcome: The if statement works, the code doesn't start without an initial input. The only issue now is we want to use the value of 0 for food that doesn't need a timer. We will assign clockValue as -1 and have the conditional statement as follows:

Figure 29 - IR sensor code

```
while (clockValue == -1){
    if (irrecv.decode(&results)){
    long int decCode = results.value;
    Serial.println(results.value);


    //switch case to use the selected remote control button
    switch (results.value){



      case 16738455:
      clockValue = 1;
      break;

      case 16724175: //when you press the 1 button
        clockValue = 60;
        break;
      case 16718055: //when you press the 4 button
       clockValue = 120;
        break;
       case 16743045: //when you press the 2 button
        clockValue = 180;
        break;
       case 16716015: //when you press the 5 button
        clockValue = 240;
        break;
       case 16726215: //when you press the 3 button
        clockValue = 300;
        break;
       case 16734885: //when you press the 6 button
        clockValue = 360;
        break;
```

Issue 5: Servo motor would tick with clock

Suspected Problem 1: The servo motor is connected in the Arduino code to the clock, causing it to tick as the clock pulses.

Action: Added a delay between servo position settings to let it complete cycle before returning to initial position.

Outcome: The clock began to malfunction, due to the unwanted delay.

Suspected Problem 2: The servo code itself is getting skipped, meaning it goes from one position to the same position

Action: add a serial print statement to see what is being sent to the device.

Outcome: The serial prints reads nothing. Nothing is sent to the device.

Action: Removed Servo code during countdown phase since both negatively affect the other. Both are time dependent and any delay from the servo will impact the time display on the LCD

Outcome: Code works as expected. Time left is displayed on LCD in real time. Unfortunate that the stirring halts in countdown phase, but this change was necessary.

Issue 6: Dabble sends a zero or placeholder character

Suspected Problem: The function returns an unusable variable type

Action: Casted the function to return an int

Outcome: The Serial monitor returns a value from 0-100 depending on the location in the slider. It works as intended

Future Work

Going forward, research should be done to improve upon the safety of the device, as this was the lowest scoring constraint for this idea. This is also a difficult issue to tackle, as safety is challenging to implement to wireless cooking. This issue was addressed in this final design by implementing device characteristics that would shut the device off under certain conditions. Although this was sufficient for this project, parts of this device still present a burn or splash/splatter hazard to its users. Moving forward, a system to cover the contents of the cooktop or hot parts of the device would work to greatly improve the safety of the device.

This device may also be improved upon by expanding upon the cooking techniques the device is capable of. This version of the device is only capable of stirring its contents, whereas a new version should be capable of flipping or mixing its contents. This would diversify the foods that can be prepared in the device and increase the devices' overall usefulness.

Furthermore, to reduce the device's meal preparation time, food storage systems should be implemented. This would eliminate preparation time as the device would be able to pull ingredients instantly rather than requiring its user to load it with fresh ingredients.

Conclusion

Throughout this project, a deeper understanding of Arduino, its components, as well as the design process was achieved. The design of the device evolved to its final state through empathizing the user's needs. A series of constraints was made and weighed to better choose a solution. From this, one idea was chosen: the automated stovetop.

Developing the circuit soon followed, starting with perfecting the individual components and eventually a fully integrated circuit. The breadboard was replaced with a custom designed PCB, which connects right onto the Arduino and provides user-friendly connecting of components with the Arduino wires.

Bibliography

[1] G. Philips, "Does Eating Breakfast Affect the Performance of College Students on Biology Exams?," n.d.. [Online]. Available: https://files.eric.ed.gov/fulltext/EJ876514.pdf .

[2] A. Mohiuddin, "Skipping Breakfast Everyday Keeps Wellbeing Away," 2018. [Online]. Available: https://www.fortunejournals.com/articles/skipping-breakfast-everyday-keeps-wellbeing-away.html.

[3] Marketing, "The important benefits of breakfast for kids and students," 2018. [Online]. Available: https://www.linkhc.org.au/the-benefits-of-breakfast/.

[4] L. Wright, "Many college-aged students skip breakfast, study shows," 2017. [Online]. Available: https://www.dailyhelmsman.com/news/many-college-aged-students-skip-breakfast-study-shows/article_93b31df2-a3df-11e7-.

[5] C. Watson, "How long does it take to eat breakfast [average times]," n.d.. [Online]. Available: https://eatforlonger.com/how-long-does-it-take-to-eat-breakfast/ .

[6] Merriam-Webster, "Fatigue definition & meaning," n.d.. [Online]. Available: https://www.merriam-webster.com/dictionary/fatigue.

[7] USDA, "Safe Minimum Internal Temperature Chart," 11 05 2020. [Online]. Available: https://www.fsis.usda.gov/food-safety/safe-food-handling-and-preparation/food-safety-basics/safe-temperature-chart. [Accessed 20 02 2022].

[8] J. Rubin, "Thermistors," 2013. [Online]. Available: https://www.juliantrubin.com/encyclopedia/electronics/thermistor.html.

[9] Toppr, "What is Nichrome Wire used for?," n.d.. [Online]. Available: https://www.toppr.com/guides/physics/electronics/what-is-nichrome-wire-used-for/.

[10] Bulbs.com, "Heat: Light bulb types," 01 03 2022. [Online]. Available: https://www.bulbs.com/learning/heat.aspx#:~:text=Heat%20lamps%20are%20incandescent%20lamps%20used%20for%20the,incandescent%20lamps%2C%20but%20produce%20much%20more%20infra.

[11] Home Depot, "Philips 175W PAR38 Heat Lamp Red Hard Glass | The Home Depot Canada," 01 03 2022. [Online]. Available: https://www.homedepot.ca/product/philips-175w-par38-heat-lamp-red-hard-glass/1000417838.

[12] Amazon, "Nichrome 80 - 250' - 32 Gauge Resistance Wire," 01 03 2022. [Online]. Available: https://www.amazon.ca/gp/product/B07CJ63YYT/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1.

[13] StackExchange, "Voltage - How do I find Nichrome Temperature," 2014. [Online]. Available: https://electronics.stackexchange.com/questions/84516/how-do-i-find-nichrome-temperature.

[14] Amazon, "Salton Single Coil Portable Electric Cooktop with Large Burners, Variable Temperature Control and Dual Indicator Lights, Perfect for Camping, 1000 Watts, Black (HP1940)," 02 03 2022. [Online]. Available: https://www.amazon.ca/Salton-Portable-Tempertature-Indicator-HP1940/dp/B088KV3X5Z/ref=pd_sbs_5/146-8681096-8358929?pd_rd_w=NV7E0&pf_rd_p=5e3de09f-c23c-4218-88f9-a2511d573f0c&pf_rd_r=J6764CA6YVMDJ3K338MA&pd_rd_r=a6a7c455-9523-4026-b84a-6a7ef0063020&pd_rd_w.

[15] Amazon, "Zevro KCH-06129 Compact Dry Food Dispenser, Single Control, White/Chrome," 01 03 2022. [Online]. Available: https://www.amazon.ca/KCH-06129-Compact-Dispenser-Single-Control/dp/B0016LT6SY/ref=sr_1_5?adgrpid=1353499042539413&hvadid=84593845331018&hvbmt=be&hvdev=c&hvlocphy=124479&hvnetw=o&hvqmt=e&hvtargid=kwd-84593859863118%3Aloc-32&hydadcr=20575_10230624&keywords.

[16] electronicsComp.com, "HM-10 Bluetooth Module," n.d.. [Online]. Available: https://www.electronicscomp.com/hm-10-bluetooth-module. [Accessed 30 03 2022].

[17] M. Eaton, "Getting started with the HM-10: Easy Arduino Bluetooth integration for iOS and Android!," AnyoneCanBuildRobots.com, 31 03 2020. [Online]. Available: https://www.anyonecanbuildrobots.com/post/getting-started-with-the-hm-10-easy-arduino-bluetooth-integration-for-ios-and-android. [Accessed 14 03 2022].

[18] InstructablesCircuits, "Ultrasonic Distance Sensor in Arduino With Tinkercad," n.d.. [Online]. Available: https://www.instructables.com/Ultrasonic-Distance-Sensor-Arduino-Tinkercad/. [Accessed 09 03 2022].

[19] S. Saini, "Project Hub," 24 10 2021. [Online]. Available: https://create.arduino.cc/projecthub/sainisagar7294/arduino-based-ir-remote-decoder-ac67c4. [Accessed 20 03 2022].

[20] Instructables Circuits, "Real Time Clock with Arduino," n.d.. [Online]. Available: https://www.instructables.com/Real-Time-Clock-Module-With-Arduino/. [Accessed 21 03 2022].

Appendices

User Manual

- Thank you for purchasing your new breakfast club. Here is how to use it effectively and safely.
  - *****KEEP YOUR BODY AWAY FROM THE HOT HEATING ELEMENT – IT WILL HURT YOU*****
- Firstly, plug the stovetop into a 120VAC outlet and plug the Arduino board into either a 9V battery or into your computer via the printer cable.

MAKE SURE TO NOT HAVE ANYTHING COMBUSTABLE NEAR THE STOVETOP

- Next, place your cookware onto the stove and fill it with the food you would like to cook.
- Secure the BBQ probe into the food either through puncturing the meat or placing into he liquid.
- To connect your phone to the Bluetooth module:
  - Open the dabble application
  - Click on the outlet icon in the top right corner (beside settings)
  - Scroll down until you see "DSD Tech"
  - Click on "DSD Tech"
  - Congratulations, it is now connected
    - If cannot see "DSD Tech", Unplug and re-plug in the Arduino.
    - Repeat until successful
  - Now that you're connected. Go back to the main menu and click on the "LED Brightness Control" icon.
  - Press the "On" button in the middle of the screen
  - Pick up the remote control and point it at the Arduino
  - Press a button from 0-9 (this is how long you want to cook the food in minutes).
  - Look back at your phone screen and adjust the slider until you reach your desired cooking temperature (read on the top line of the LCD screen) (see internal cooking temperatures) – Boiling Water is 212
  - Wait until the timer reaches zero and the LCD displays "DONE"
  - ENJOY YOUR WARM MEAL

Product	Minimum Internal Temperature & Rest Time
Beef, Pork, Veal & Lamb Steaks, chops, roasts	145 °F (62.8 °C) and allow to rest for at least 3 minutes
Ground Meats	160 °F (71.1 °C)
Ground Poultry	165 °F
Ham, fresh or smoked (uncooked)	145 °F (62.8 °C) and allow to rest for at least 3 minutes
Fully Cooked Ham (to reheat)	Reheat cooked hams packaged in USDA-inspected plants to 140 °F (60 °C) and all others to 165 °F (73.9 °C).
All Poultry (breasts, whole bird, legs, thighs, wings, ground poultry, giblets, and stuffing)	165 °F (73.9 °C)
Eggs	160 °F (71.1 °C)
Fish & Shellfish	145 °F (62.8 °C)
Leftovers	165 °F (73.9 °C)
Casseroles	165 °F (73.9 °C)

*****KEEP YOUR BODY AWAY FROM THE HOT HEATING ELEMENT – IT WILL HURT YOU*****