Smart Microwave Model

ICS 2202: Operating System

151168: Isaboke, Fidel Agade

School of Computing and Engineering Sciences

Strathmore University

7 November 2023

Smart Microwave Model

Part 1: Introduction

A. Background Information

The microwave oven is one of the most commonly used home appliances globally. It can get food warmed or cooked in no time, by using microwaves, which are transformed into heat energy. This useful technology is time-saving and presents advancements in cooking over the years.

B. Problem Statement

One of the main setbacks facing microwave ovens is the integration of wireless communication with microwave ovens. Since Bluetooth, Wi-Fi, and microwave use a similar frequency, there tends to be electromagnetic interference, which distorts the signals, making it difficult to communicate.

This project aims to model a simple microwave model, integrated with a Bluetooth module, using the Tinkercad simulation software.

Given the limitations of the Tinkercad simulation tool, a very simplified design of a microwave oven will be made to demonstrate its working.

C. Objectives

Main Objective

The main objective of this project is to demonstrate the simple operations of a microwave oven and control it remotely via Bluetooth

Specific Objectives

1. Design a circuit that includes a button, buzzer, DC motor, and LCD for the microwave oven model.

2. Implement a ‘power on’ and ‘power off’ operation via button switch presses.
3. Display the various options the user can perform on the LCD.
4. Develop an operation that runs the microwave by sending a command from serial input to the serial monitor. This should also activate the DC motor that simulates the cooking of food.
5. Display the timer on the LCD.
6. Alert the user about completion of the timer via tones from the buzzer and a message, displayed on the LCD.
7. Alert the user of an invalid input or option selected via buzzer tones and an error message on the LCD.

Part 2: Literature Review

A myriad of projects and research related to the system being developed have been undertaken.

a. Microwave Oven (by Beng Chet)

This project was developed in Penang SEA Makerthon 2016. This is a more complex implementation that involves buttons for navigation, a fan, and a heater.

Source Code link: Microwave Oven by bengchet

b. Bluetooth-enabled Microwave Ovens for EMI Compatibility

This is a study conducted by Neelakanta et al [1], focusing on the impact of microwave devices such as microwave ovens on wireless communication via Bluetooth. EMI is a phenomenon that occurs when an external electromagnetic device produces waves that may distort the original signal of transmission [2].

The high-powered microwave devices share the same band as the Bluetooth signal, hence interference occurs, effectively reducing the efficiency of communication, since the carrier signal

malfunctions upon encountering noise [1]. This poses a big challenge in developing a microwave oven that can be operated wirelessly.

c. Mitigating Microwave Oven EMI over Bluetooth Communication

This article, written by Neelakanta and Jesada Sivaraks [1] describes the techniques that can be used to reduce or prevent EMI. One of the techniques involves the microwave device alerting the Bluetooth devices to re-route the carrier signal to prevent noise or interference.

Apparatus

Various apparatus required to model the microwave oven include:

1. An HC05 Bluetooth module - For Bluetooth Communication
2. Arduino Uno R3 microcontroller
3. Connecting Wires
4. Piezo Buzzer
5. DC Motor
6. LCD (16 × 2)
7. Button switch
8. A USB-compatible computer with Arduino IDE installed
9. Tinkercad software - for simulation
10. 2 Breadboards
11. 2 Resistors (one rated 220 Ω)
12. Bluetooth-enabled smartphone

Part 3: Methodology

A. System Design

The use case diagram (Figure 1 on the next page) points out the various events and use cases of the system. It gives a simple picture of the various actions that can be carried out by the user of the system. From this diagram, the circuit is designed.

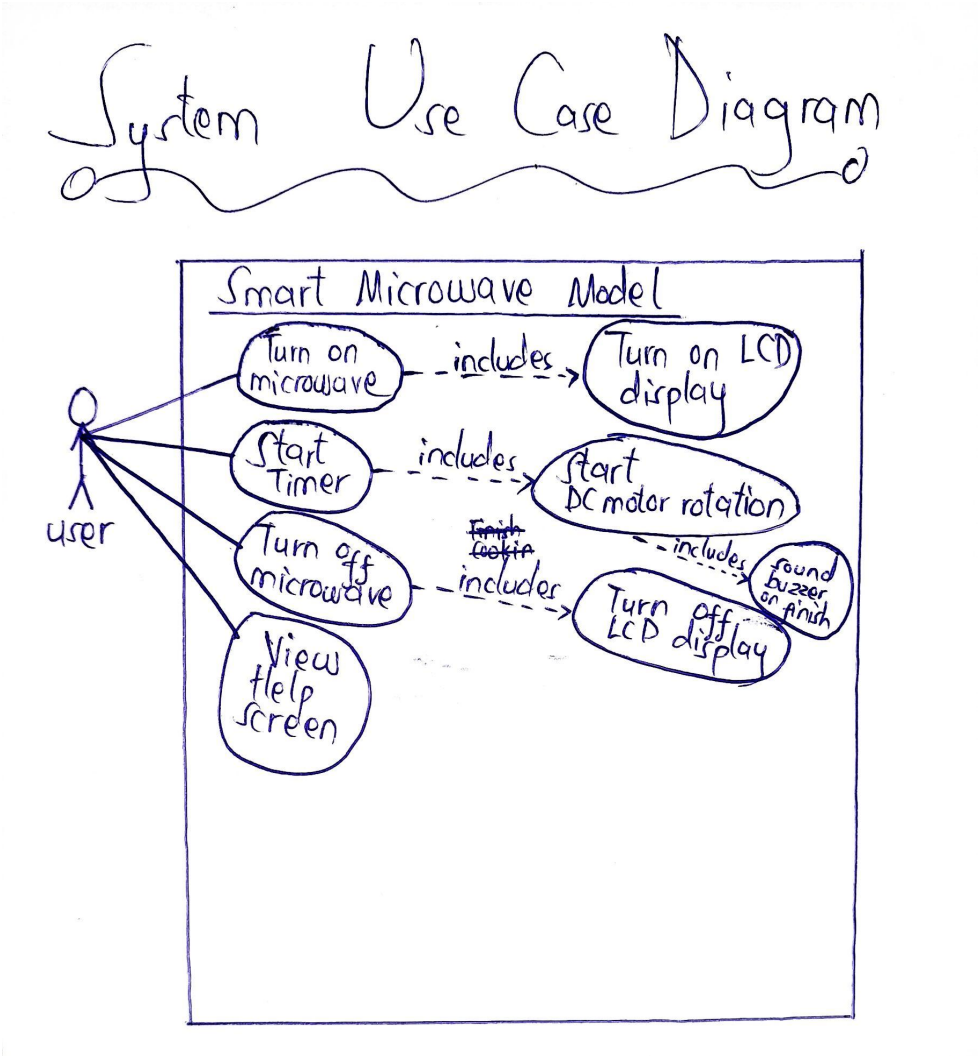

Figure 1: Use case diagram

B. Circuit Setup

Using the apparatus listed on page 4, follow the steps below to get the project running:

1. Connect the GND pin on the Arduino R3 to the negative line of both breadboards using a wire, to provide grounding.
2. Connect the 5V pin on the Arduino R3 to the positive line of both breadboards, to provide a 5V voltage supply.
3. Next, fix the LCD on one breadboard as shown in Figure 2. From left to right, make the following connections using wires to the Arduino R3:

a. GND pin goes to the ground.
b. VCC to 5V power supply
c. VO (Contrast) to ground
d. RS (Register Select) to digital pin 6
e. RW (Read/Write) to ground
l. LED Cathode to ground via the 220 Ω resistor
f. E (Enable) to digital pin 7
g. DB4 to digital pin 8
h. DB5 to digital pin 9
i. DB6 to digital pin 10
j. DB7 to digital pin 11
k. LED Anode to 5V

4. Next, fix the Piezo buzzer onto the breadboard with the LCD. Connect the positive terminal of the buzzer to pin 4, and the negative terminal to the ground.
5. Fix the DC Motor onto the breadboard with the LCD. Connect the left terminal to pin 2, and the right one to pin 3.
6. Fix the button onto the breadboard with the LCD. Connect the upper-left terminal to pin 5 of the Arduino R3. Connect the lower left terminal to GND using the other resistor, and the lower right terminal to the 5V power supply.
7. Finally, fix the Bluetooth module (represented by wires) on the other breadboard as shown. Connect VCC to the 5V power supply, GND to the ground, the TX pin of the module to the RX pin of the Arduino R3, and the RX pin of the module to the TX pin of the Arduino R3.
8. Verify that the connections have been made correctly (See Figure 2).
9. Open Arduino IDE on your computer, and copy the script shown on page 7.
10. Verify the script, connect the Arduino R3 to your computer, and upload the script shown on page 8.

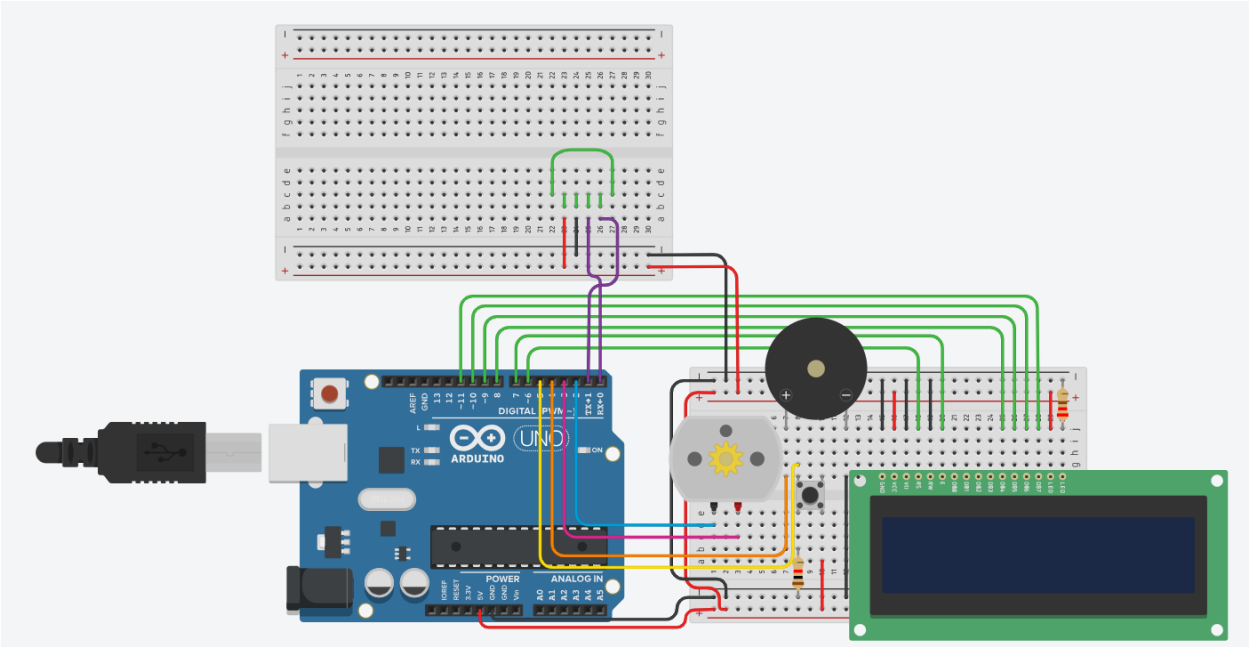

Figure 2: Tinkercad Circuit

Script

Copy the code below on the code editor of the IDE:

```
/*
Microwave Modelling - by Fidel Isaboke
Simulating a microwave using Arduino
Circuit built on Tinkercad simulator
*/

#include <LiquidCrystal.h>
LiquidCrystal lcd(6, 7, 8, 9, 10, 11);

// What can the microwave do?
// Check at the end of 'void loop()' for full definitions
void checkPower();
void togglePower(bool microwave_state);
void powerOn();
void powerOff();
void displayMenu();
void processOption(int option_input);
```

```
void setTimer(); // Unused - requires further implementation
void displayHelp();
void run(unsigned int seconds);
void activateMotor(unsigned int seconds);
void cancel(); // Undefined - No implementation
bool checkOption(int option_input);
void displayOptionError();
void beep(int no_of_beeps, int tone_delay); // For the buzzer

// Setting pin constants
const int motor_left = 2;
const int motor_right = 3;
const int buzzer = 4;
const int power_button = 5;

/* Important variables (definitions) */
int option;
int previous_btn_state;
int current_btn_state;
bool microwave_on = false;

// Array of accepted options:
int options[2] = {1, 2};


/* The microwave functionality */
// Check the power button
void checkPower(){
  previous_btn_state = current_btn_state;
  current_btn_state = digitalRead(power_button);

  if(previous_btn_state == LOW && current_btn_state == HIGH){
    microwave_on = !microwave_on;
    delay(20);
    togglePower(microwave_on);
  }
}

// Toggle the power on or off
void togglePower(bool microwave_state){
```

```
  if(microwave_state == true){
    powerOn();
  }
  else{
    powerOff();
  }
}

// Power on the microwave
void powerOn(){
  // Startup display
  lcd.setCursor(0, 0);
  lcd.print("Microwave Model");
  lcd.setCursor(0, 1);
  lcd.print("By Fidel Isaboke");
  delay(1500);

  lcd.clear();
  delay(500);

  displayMenu();
}

// The microwave menu
void displayMenu(){
  if(Serial.available()>0){
    lcd.clear();
    option = Serial.parseInt();
    Serial.println(option);
    bool is_an_option = checkOption(option);
    if(is_an_option){
      // Proceed to process option
      processOption(option);
    }else{
      // Display an error
      displayOptionError();
    }
  }else{
    lcd.clear();
    lcd.setCursor(0, 0);
```

```
    lcd.print("1 - Run");
    lcd.setCursor(0, 1);
    lcd.print("2 - Help");
  }
}

// Check if an option is valid
bool checkOption(int option_input){
  bool valid_option = false;
  for(int i = 0; i < sizeof(options); i++){
    if(option_input == options[i]){
      valid_option = true;
      break;
    }
  }
  return valid_option;
}

// Process the option and redirect to submenu
void processOption(int option){
  if(option == 1){
    run(30);
  }else if(option == 2){
    displayHelp();
  }else{
    displayOptionError();
  }
}

// Set microwave timer
// Unused, may implement in future
void setTimer(){
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Timer (sec)");
  lcd.setCursor(0, 1);
  lcd.print("0 - 3600 sec");
}

void displayHelp(){
```

```
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Use 1 or 2");
  lcd.setCursor(0, 1);
  lcd.print("to navigate");
}

// Display an error message and sound the buzzer
void displayOptionError(){
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Invalid option");
  lcd.setCursor(0, 1);
  lcd.print("Options: ");
  lcd.setCursor(10, 1);
  lcd.print("1 or 2");
  beep(3, 200);

  // Redirect to menu
  displayMenu();
}

// Run the microwave
void run(unsigned int seconds = 20){
  activateMotor(seconds);
  for(int i = 0; i < 3; i++){
    beep(3, 500);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Done cooking!");
    delay(500);
  }
  displayMenu();
}

// Power off the microwave
void powerOff(){
  lcd.clear();
}
```

```
// Sound the buzzer - notification
void beep(int no_of_beeps, int tone_delay = 200){
  for(int i = 0; i < no_of_beeps; i++){
    tone(buzzer, 500);
    delay(tone_delay);
    noTone(buzzer);
    delay(tone_delay);
  }
}

void activateMotor(unsigned int seconds){
  analogWrite(motor_left, 0);
  analogWrite(motor_right, 128);

  // Display time left
  for(int i = seconds; i > 0; i--){
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Time (sec):");
    lcd.setCursor(0, 1);
    lcd.print(i);
    delay(1000);
  }
}


/* Setup */
void setup(){
  /* pinMode(motor_left, OUTPUT);
  pinMode(motor_right, OUTPUT); */
  pinMode(buzzer, OUTPUT);
  pinMode(power_button, INPUT);
  lcd.begin(16, 2);
  Serial.begin(9600);

  // Initial button state - LOW
  current_btn_state = LOW;
}
```

```
/* Loop - runs the microwave */
void loop(){
  checkPower();
}
```

Explanation of the script

The script implements the specific objectives to develop the operations of the entire system. It uses several functions that operate together to make the system work:

1. **checkPower() -** This is the first function to be called by the script. It basically acts as the entry point for the system. It checks if the microwave oven is on, based on the previous and current button states. It waits for a button press, then toggles the power accordingly via the boolean variable 'microwave_on', which is passed as an argument to the 'togglePower()' function.
2. **togglePower(bool microwave_state) -** This function is called after the 'checkPower()' function. It takes in an argument that checks whether the microwave is on or not. If and only if the former is true, the system is activated via the 'powerOn()' function. Otherwise, the 'powerOff()' function is called.
3. **powerOn() -** This 'activates' the system, in the sense that the LCD displays a welcome message on what the system is. After 2 seconds, 'displayMenu()' is called to show the menu of the microwave oven.
4. **displayMenu() -** This function displays the various options available to the user and receives serial input of the option selected by the user. If the user enters an option, a check is performed on it to determine whether it's valid by passing it into a function called 'checkOption()' that returns a boolean value. The option is then processed by the

'processOption()' function if it is valid, otherwise, an error is displayed by the 'displayOptionError()' function.

5. **checkOption(int option_input) -** An option is validated to confirm whether it is in the domain of accepted options (1, 2). The function returns 'true' upon success, otherwise a 'false' is returned.
6. **processOption(int option_input) -** Another check is performed, to determine the next action to perform. A '1' entered calls the 'run()' function which basically runs the microwave oven timer, and a '2' displays the help menu by calling 'displayHelp()'.
7. **displayOptionError() -** This function displays an error message, alerting the user that they have selected an invalid option. Simultaneously, the piezo buzzer beeps 3 times, courtesy of the 'beep()' function
8. **displayHelp() -** This just displays 'Use 1 or 2 to navigate', informing the user to enter either '1' or '2' on the microwave menu to operate it.
9. **run(unsigned int seconds) -** This function runs the timer and runs the motor, specified by the number of seconds set. The 'unsigned' notation is used here to show that only positive integers are accepted.
10. **activateMotor(unsigned int seconds) -** This function is executed within the 'run()' function to rotate the motor in a clockwise direction.
11. **beep(int no_of_beeps, int tone_delay) -** This function simply sounds the buzzer. It is used in alerts (errors, and completion of cooking).
12. **void setup() -** This is the generic Arduino function that simply defines the pin modes of the connected components. Additionally, Serial communication is established with a baud

rate of 9600, and the LCD is activated. A variable, ‘current_btn_state’ is initialized to ‘LOW’.

13. **void loop()** - This is typically where all Arduino programs are run, hence the calling of checkPower() is done here. It is called continuously to monitor the state of the system (whether it is on or off).

In the script above, the ‘setTimer()’ and ‘cancel()’ functions are unused at the time of development, hence have no impact on the functioning of the system.

Part 4: Expected Results

The images on the next pages demonstrate the various messages displayed by the LCD depending on the various events that occur while running the system on the Tinkercad simulation software.

1. **On starting the simulation**

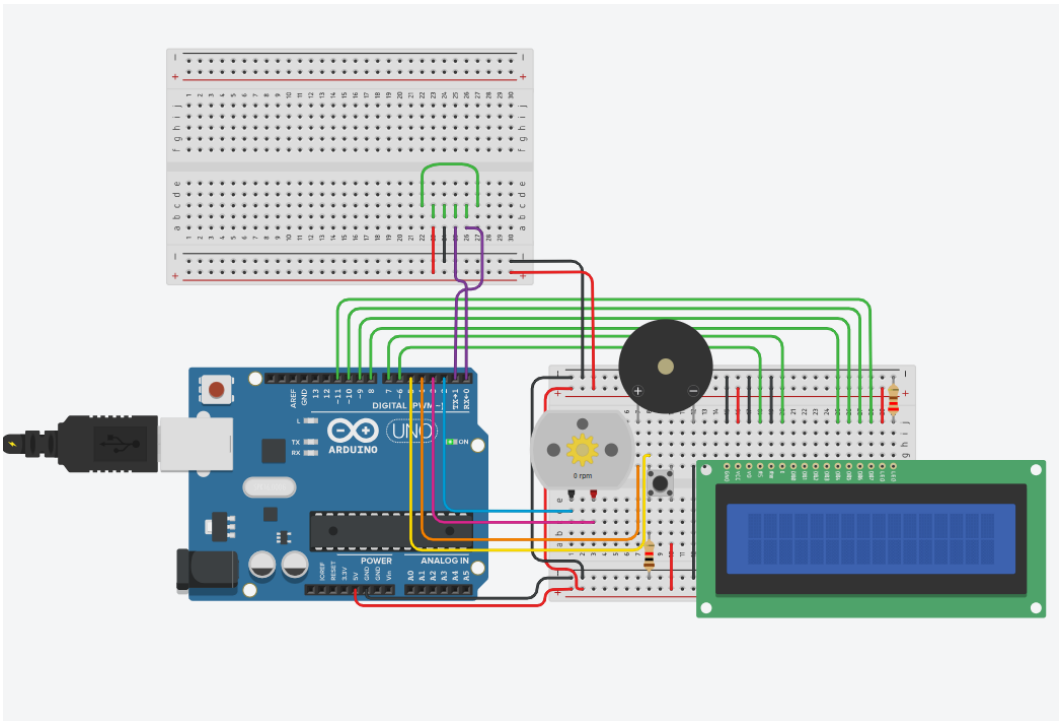

Figure 2: Starting simulation

2. **Pushing the button to power on the system**

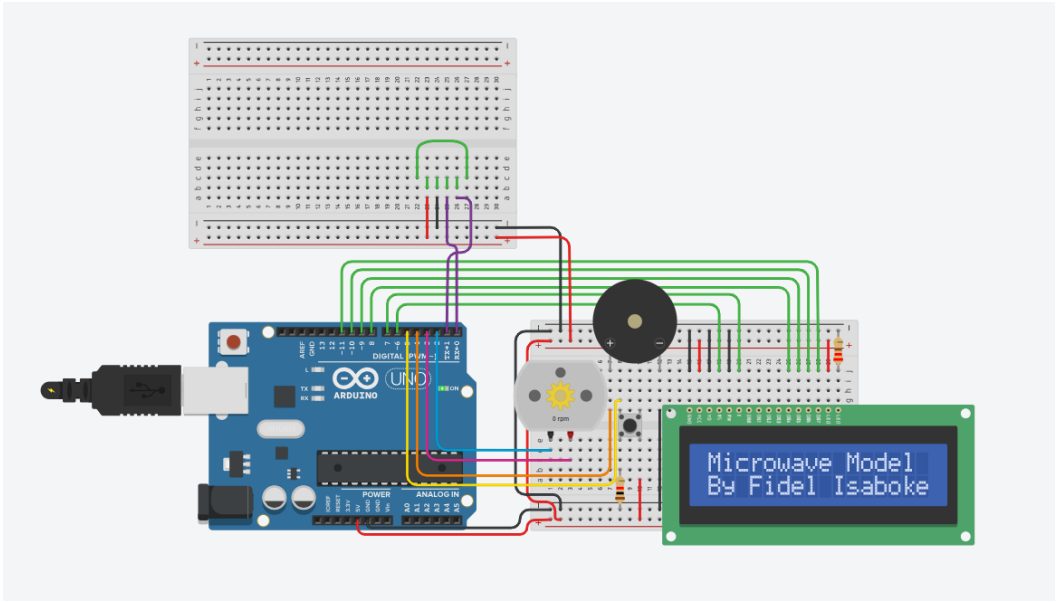

Figure 3: On startup

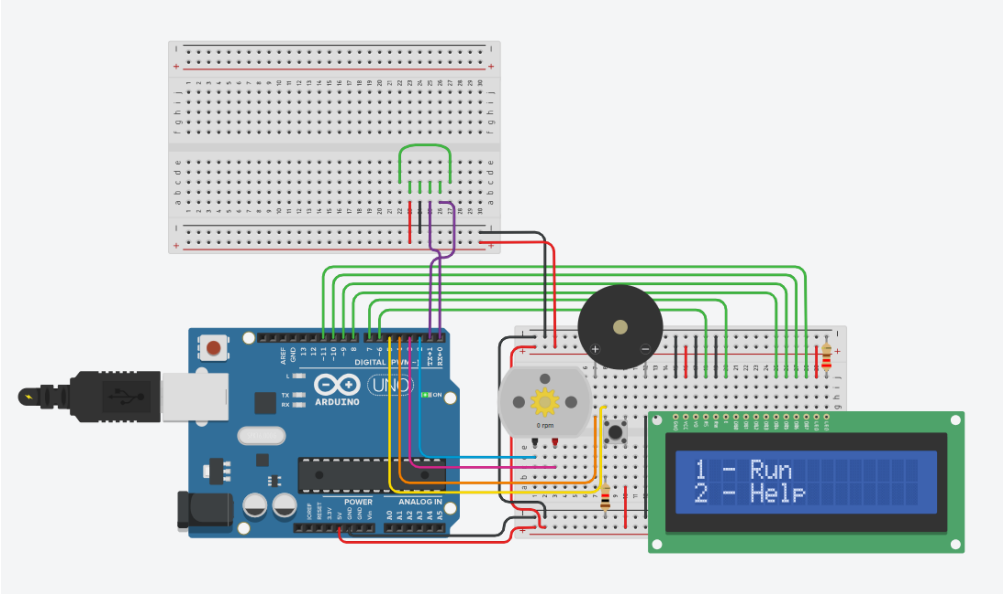

Figure 4: After the welcome message (2 seconds later)

3. **Selecting option '1'**

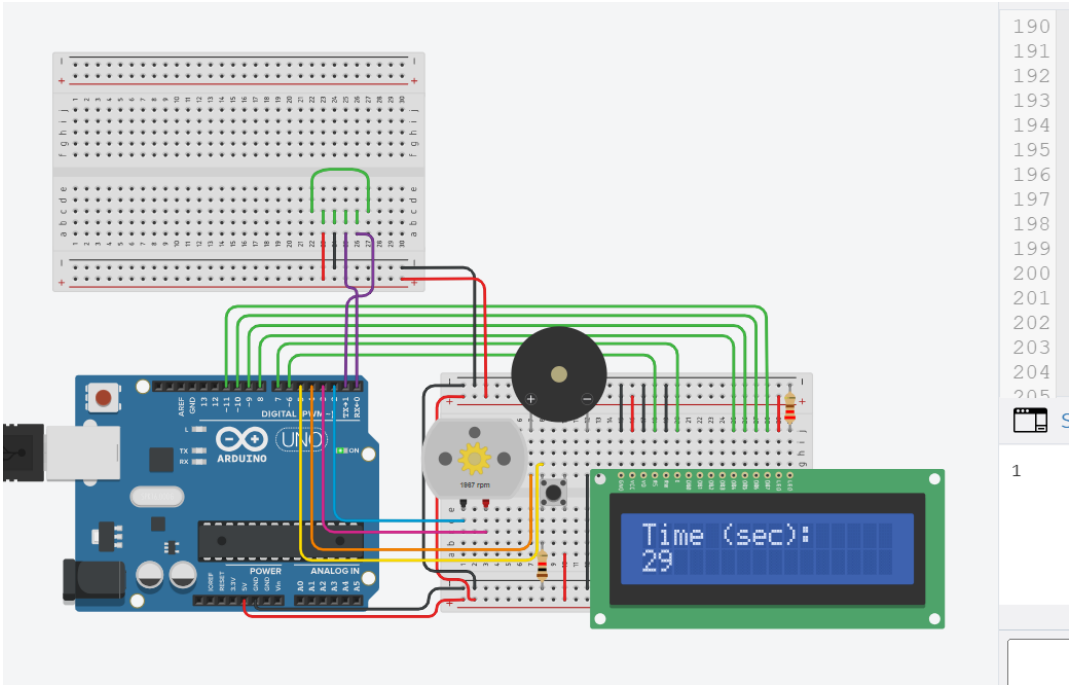

Figure 5: Option 1 - Run

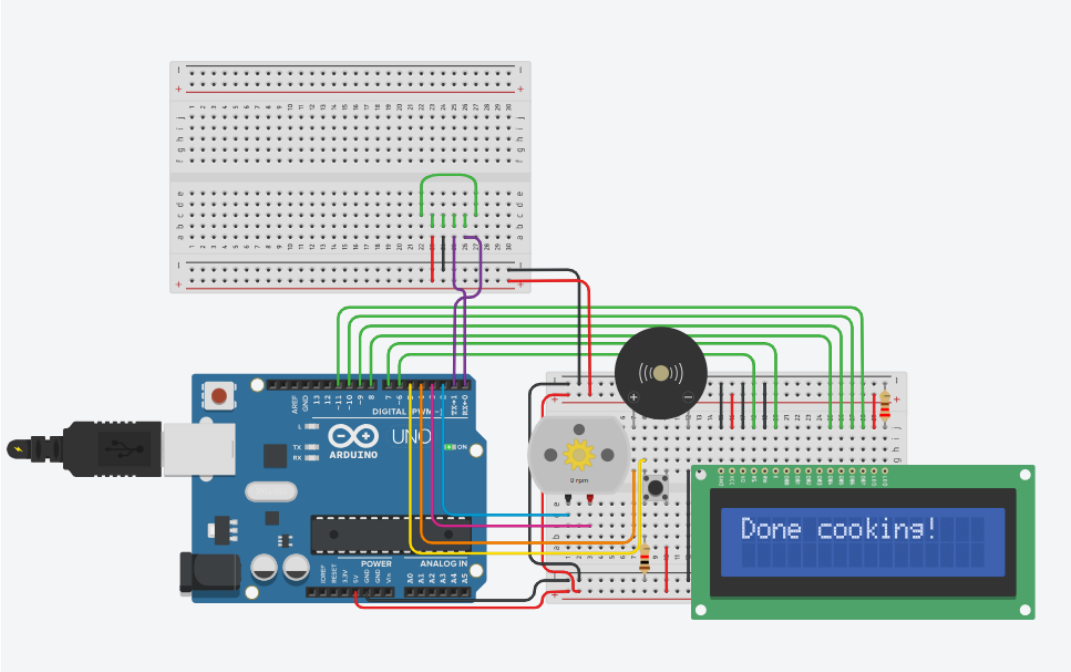

Figure 6: Upon completion of the timer

4. **Selecting option '2'**

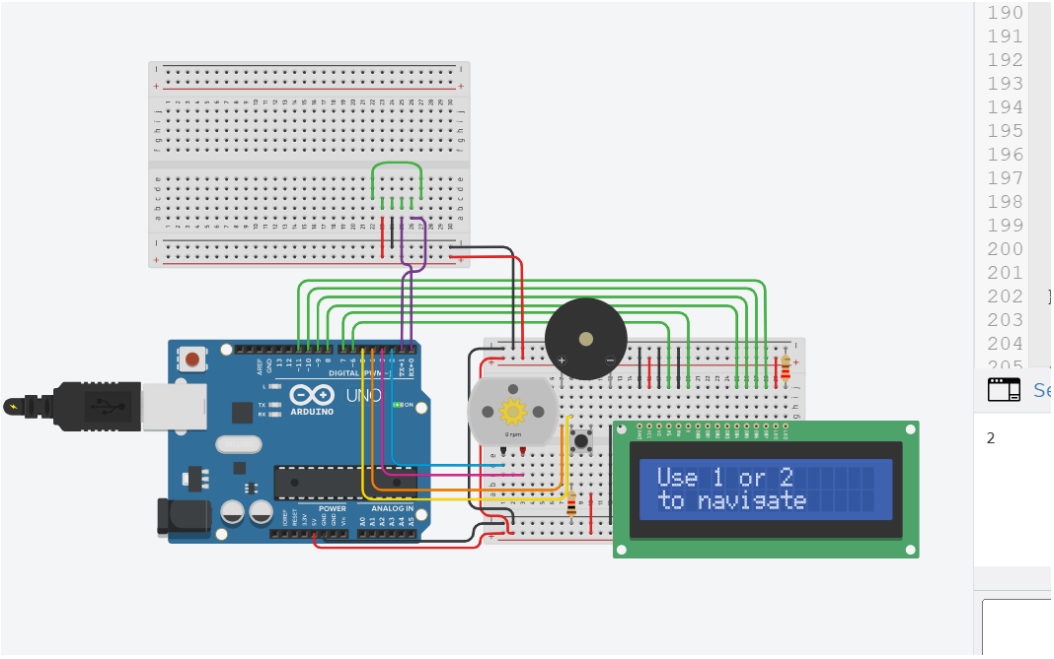

Figure 7: Option 2 - Help

Part 5: Conclusion and Recommendations

To sum it up, this project looks at a way of modelling a microwave oven in a simplified manner, making use of the Arduino R3 microcontroller. Moreover, it also demonstrates remote control of the system designed, via Bluetooth and serial communication

Therefore, it is possible to model a simple microwave oven using the Arduino R3 and various electrical components that work with the controller, bearing in mind the existing limitations. Bluetooth communication is possible, thanks to the HC05 Bluetooth module, which enables a smartphone with the However, one recurring problem with this setup is the need to repeatedly turn off and on the system via the button switch. This might be due to the serial input being delayed by the seemingly infinite execution of the ‘displayMenu()’ function. Otherwise, it operates as implemented.

Further research and enhancement can be made to this setup to include microwave functionality to actually cook foodstuffs, keeping in mind the existing Bluetooth module. A way of reducing potential interference can be looked at.

References

[1] J. S. P.S. Neelakanta, “Bluetooth-enabled microwave ovens for EMI compatibility,” Microwave Journal, https://www.microwavejournal.com/articles/3118-bluetooth-enabled-microwave-ovens-for-emi-compatibility (accessed Nov. 7, 2023).

[2] G. Wright, “electromagnetic interference (EMI),” *Mobile Computing*, Mar. 08, 2022. https://www.techtarget.com/searchmobilecomputing/definition/electromagnetic-interference#:~:text=What%20is%20an%20electromagnetic%20interference,malfunction%20or%20stop%20working%20completely (accessed Nov. 7, 2023).

[3] Bengchet, “Arduino-Projects/microwave-oven/microwave-oven.ino at master · bengchet/Arduino-Projects,” *GitHub*. https://github.com/bengchet/Arduino-Projects/blob/master/microwave-oven/microwave-oven.ino (accessed Nov. 7, 2023).