An-Najah National University

Faculty of Engineering & Information Technology

Presented in partial fulfillment of the requirements for

Bachelor degree in Computer Engineering

Graduation Project 2

Ras Al-Abed

Students:

Tala Yaseen

Samaa Yasin

Supervisor:

Dr. Mahmoud Assad Dwikat

m_assad@najah.edu

February 10, 2025

Disclaimer

This report was written by students Tala Yaseen and Samaa Yasin at the Computer Engineering Department, Faculty of Engineering, An-Najah National University. It has not been altered or corrected, other than editorial corrections, as a result of assessment and it may contain language as well as content errors. The views expressed in it together with any outcomes and recommendations are solely those of the students. An-Najah National University accepts no responsibility or liability for the consequences of this report being used for a purpose other than the purpose for which it was commissioned.

Acknowledgment

No success is achieved without the support of those around us. This project became possible thanks to Allah and the incredible people who supported us along the way. We are deeply grateful to our project supervisor, Dr. Mahmoud Asaad, for his guidance, support, and expertise throughout the project. His dedication kept us motivated and focused.

We would also like to express our thanks to our friends, classmates, and families for their constant support, valuable feedback, and helpful suggestions. Their unwavering patience, encouragement, and understanding were essential in refining and improving the quality of our work. Without their support, we would not have been able to fully dedicate ourselves to the project and bring it to completion.

We sincerely acknowledge the effort, time, and support of everyone who contributed to making this project a success. We are genuinely grateful to all those who stood by us throughout this journey.

Abstract

Raas Alabed is one of the most popular treats among children and adults, particularly in winter when the industry thrives. Many local factories in Palestine, such as the Al-Arz factory, produce this marshmallow-based, chocolate-coated candy. Its foam-like structure and rich chocolate coating make it a favorite childhood sweet, associated with energy and warmth during the colder months.

This project aims to fabricate a production line for Raas Alabed while preserving its traditional taste and cultural significance, particularly in the Levantine region. The goal is to enhance local factories' capabilities by integrating modern computer technologies into traditional production methods. The project focuses on automating the production process through embedded systems. Key objectives include automating production using sensors and microcontrollers. The methodology involves designing mechanical components, integrating them with electronics, and programming software for automation. While similar systems exist for other chocolate-coated marshmallows, this project is unique in its emphasis on blending Levantine traditional methods with modern technology.

The production line consists of four stages. In the first stage, the cream is prepared—whipped, and mixed based on user input (standard or cocoa-flavored). The second stage involves placing the biscuits accurately, followed by dispensing the cream onto them. In the third stage, the melted chocolate is poured to cover the cream completely.

The main features of this production line are:

• User-controlled customization: The system allows operators to input preferences for cream type (standard or cocoa).

• Automated precision: The line is equipped with sensors that ensure accurate biscuit placement and precise dispensing of both cream and chocolate, minimizing human error and increasing efficiency.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General background

Traditional methods of making sweets, especially sweets such as Ras Al-Abed, suffer from many problems that affect the quality and efficiency of production because manual production is inaccurate and varies in size and the amount of cream and chocolate added, which leads to consumer dissatisfaction in addition to high costs and the inability to compete in the market because the production quantity is limited if labor is relied upon.

There is also a problem related to cleanliness because the product is handled manually, so there is a great risk of contamination, which affects safety. Let's not forget that manual production is slow, which limits high productivity, and consequently, large losses due to the inability to compete in the market.

All these reasons led to resorting to the idea of automation, as the process is more accurate, faster, and safer, thus increasing consumer satisfaction and success in the labor market.

Figure 1.1: Traditional Ras Al-Abed Production Process

1.2 Objectives

The main objective of this project is to automate the production process of Ras Al-Abed to ensure efficiency and high quality and reduce material losses, thus obtaining a product that competes with companies and satisfies the user. It also ensures a safe work environment free from pollution, thus a healthy product.

1.3 Significance of the Work

The automation of Ras Al-Abed production is of great significance due to the increasing demand for high-quality, consistent, and hygienic food products in the market. Market studies show that automation in food manufacturing leads to increased productivity, reduced waste, and enhanced profitability. The global confectionery market is experiencing steady growth, and companies that adopt automated production methods can achieve a competitive edge by reducing costs and improving product quality.

Moreover, food safety is a crucial concern for consumers and regulatory bodies. Automated production minimizes human contact, reducing the risk of contamination and ensuring compliance with health and safety standards. By implementing automation, manufacturers can scale production efficiently, meet market demands, and enhance brand reputation through consistent quality assurance.

1.4 Organization of the Report

This report presents an in-depth examination of the challenges associated with traditional Ras Al-Abed production and the necessity for automation to improve efficiency, consistency, and hygiene standards. The introduction outlines the limitations of manual production and the motivation behind implementing an automated system.

Following the introduction, a comprehensive literature review is presented, discussing previous research and techniques related to food production automation. The methodology section details the approach taken in designing and implementing the automated system, including the components and mechanisms employed.

Subsequently, the implementation and results section provides an analysis of the system's performance, evaluating its effectiveness in comparison to traditional methods. Finally, the report concludes with a discussion of the findings, emphasizing the significance of automation in the confectionery industry and potential future improvements for further optimization.

Chapter 2

Theoretical Background and Previous Work

2.1 Theoretical Background

2.1.1 Overview of Chocolate-Coated Marshmallow Treats

Chocolate-coated marshmallow treats, commonly known as Ras Al-Abed, consist of a marshmallow-like filling placed on a biscuit base and coated with chocolate. These treats have been popular for decades, especially in the Levant region, where they are widely consumed during colder months.

The traditional production of Ras Al-Abed involves multiple steps, including whipping the marshmallow, preparing the biscuit and coating with chocolate. While these steps have been historically performed manually, technological advancements have introduced automation into confectionery production, improving efficiency, precision, and hygiene.

2.1.2 Automation in Food Manufacturing

Automation in food manufacturing involves integrating mechanical, electronic, and software systems to streamline production, minimize human intervention, and enhance quality control. Several key technologies are widely used in food production automation:

- **Conveyor Systems:** Automate the transportation of raw materials and finished prod-

ucts between different stages of production.

- **Automated Mixing and Depositing Systems:** Ensure uniformity in ingredient proportions and precise placement of marshmallow on biscuits.
- **Chocolate Enrobing Machines:** Enable consistent coating of chocolate on the marshmallow treat.
- **Quality Control Sensors:** Detect product defects, ensuring consistent size, weight, and coating thickness.

These automated solutions help reduce labor costs, increase production output, and improve the overall safety and consistency of confectionery products.

2.1.3 Food Safety and Hygiene Considerations

A critical challenge in traditional confectionery production is ensuring food safety. Manual handling of food products increases the risk of contamination, inconsistent quality, and hygiene concerns. Automation plays a key role in mitigating these risks by:

- Reducing direct human contact with food products.
- Maintaining precise temperature control during chocolate melting and cooling.
- Implementing sanitary equipment designs to comply with food safety standards.

Automated food production lines are designed to meet regulatory requirements, ensuring that products are safe for consumption while maintaining traditional taste and texture.

2.1.4 Previous Work

Traditional vs. Automated Chocolate-Coated Marshmallow Production

Several studies and industrial implementations highlight the benefits of automation in confectionery production:

Traditional Production Challenges:

- Inconsistent marshmallow and chocolate proportions.
- High labor costs and slow production rates.
- Increased risk of contamination due to manual handling.

Automated Production Advantages:

- **Precision and Consistency:** Automation ensures uniformity in shape, weight, and

coating.

- **Increased Productivity:** High-speed systems allow for mass production.
- **Hygiene and Safety Compliance:** Reduces microbial contamination risks.

Case Studies on Automated Confectionery Production

- **Nestlé and Cadbury Production Lines:** These companies utilize high-speed chocolate enrobing and cooling tunnels, improving efficiency and maintaining product quality.
- **Implementation of AI and IoT in Food Processing:** Advanced AI-based vision systems are used in quality control to detect inconsistencies and defects in confectionery production.
- **Automated Depositors in Danish Flødeboller Production:** Studies show that automated depositing systems improve marshmallow texture and adherence to biscuit bases, ensuring consistent product quality.

Chapter 3

Constraints, Standards/ Codes and Earlier course work

3.1 Constraints

1. Mechanical Design Complexity: One of the biggest problems we faced was the difficulty of designing the project structure from scratch because this is a mechanical engineering specialty and we are a computer science major with little experience in designing the structure. Our greatest experience is in programming.

2. Inaccuracy of Components: Inaccuracy in some parts such as sensors and limit switches, which leads to system errors and illogical behavior.

3. High Component Costs: The high prices of electronic parts compared to their efficiency .

4. Challenges in Handling Cream Consistency: The difficulty of dealing with the consistency of the cream, as we could not move it after whipping to another place to press it, because the pumps available to us could not pump it, so we thought of an alternative to mix the cream and press it in the same place without the need for transportation.

5. Slow movement of components: The movement of some things, such as moving from the stage of mixing the cream to the stage of lowering it, is a slow movement and raising and lowering the mixers this was mainly due to the limitations of the motors used, which were not

highly efficient, affecting the overall speed and efficiency of the production process.

6. Lack of experience in some matters led to redoing things more than once to correct the mistake.

3.2 Standards/ Codes

The following list contains the hardware components that have been used in this project. We have a 1*4 keypad. The user chooses whether to add cocoa to the cream or not, then the system starts working.

Figure 3.1: Keypad

3.2.1 Mixing Part

Two Mixers: Used to blend the cream efficiently.

Figure 3.2: Mixers

DC Wiper Motor: Used to raise and lower mixers to allow the cream pot to move after mixing to the pouring stage.

Figure 3.3: DC Wiper Motor

Relay for DC Motor: 2-channel relay module was used to to control the direction of rotation of the DC motor to allow it to mix when it is down and raise it to move the pot to the next stage, which is pouring.

Relays for Mixers: A 2-channel relay module was used to control the on and off of the mixers.

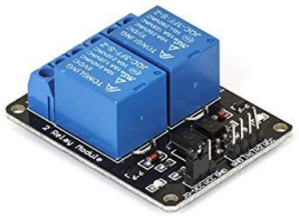

Figure 3.4: Relay 2ch

Two Limit Switches Upper Limit Switch: Stops the motor when the Mixers reach the highest position, preventing over-travel. Lower Limit Switch: When pressed, the Mixers will reach the mixing point.

Figure 3.5: Limit Switch

3.2.2 Cream pouring part

NEMA 23 Stepper Motor: It was used to control the pouring of the cream press tool to be deposited on the biscuit.

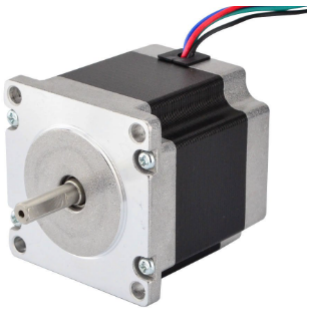

Figure 3.6: Stepper Motor NEMA 23

Microstep Driver: used to control the NEMA 23 stepper motors, providing smoother movement and better precision.

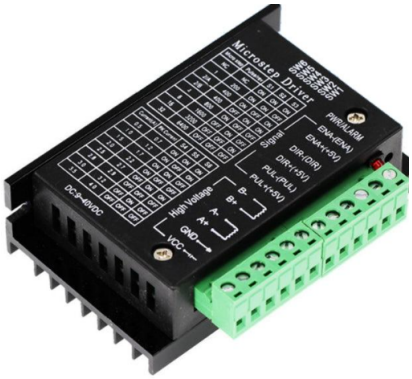

Figure 3.7: MicroStep Driver

Two Limit Switches Upper Limit Switch: Stops the motor when the Cream press tool reach the highest position, preventing over-travel. Lower Limit Switch: When pressed, the Cream press tool will reach the bottom of the pot (pot become empty).

Figure 3.8: Limit Switch

DC Motor with Gearbox: Used to open and close the pipe that dispenses the cream to ensure controlled flow and preventing leakage.

Figure 3.9: DC Motor with Gearbox

2-Channel Relay: Used to control the DC motor opening and closing it operate in both directions for the dispensing pipe as needed.

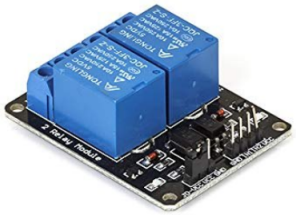

Figure 3.10: 2-Channel Relay

3.2.3 Biscuit part

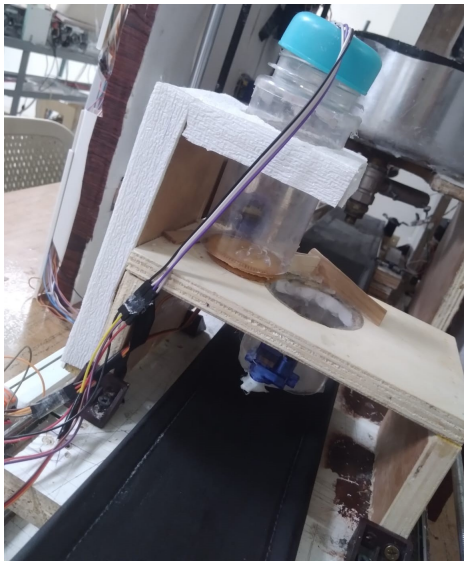

Figure 3.11: Biscuit

Two Servo Motors: Used to control the placement of the biscuits onto the production line to ensure accuracy and consistency.

Figure 3.12: Servo Motor

Ultrasonic Sensor:This sensor is used to know whether the biscuit box is empty or full.

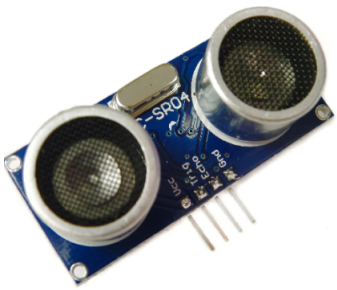

Figure 3.13: Ultrasonic Sensor

3.2.4 Chocolate part

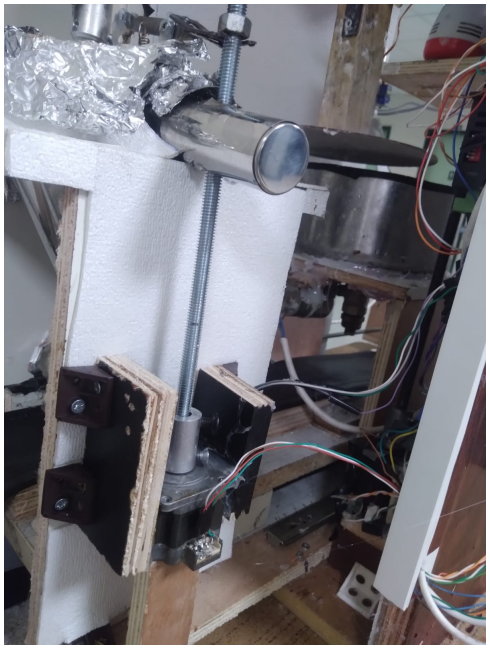

Figure 3.14: Chocolate part

Heater: An electric heater was used to heat the water and the steam melted the chocolate.

Figure 3.15: Heater

1-Ch Relay : Used to control the heater's operation and shutdown if it reaches a certain temperature.

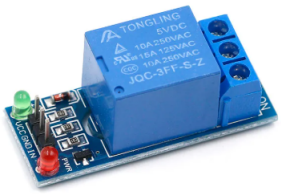

Figure 3.16: 1-Ch Relay

Sensor max6675: To track the temperature of the water vapor to control the heater shutdown.

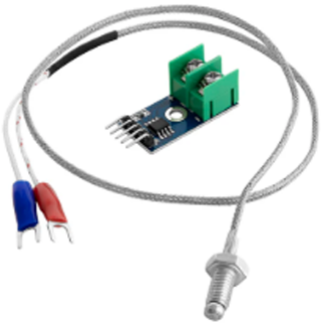

Figure 3.17: Temperature sensor

NEMA 23 Stepper Motor: It was used to control the opening and closing of the chocolate container.

H-Bridge: Used to control NEMA 23 stepper motor to allow bidirectional movement to open and close the chocolate container as needed.

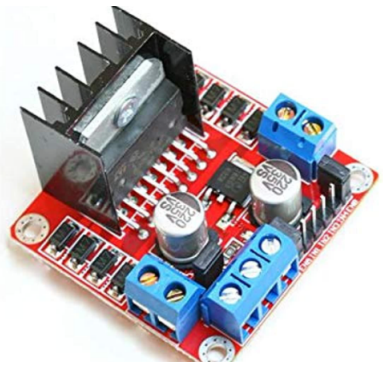

Figure 3.18: H bridge

3.2.5 Container Movement Mechanism

DC Wiper Motor: Powers the movement of the track allowing the container to transition between different stages of production. 2-Channel Relay : To Control the DC motor enable forward and reverse movement as needed. Limit Switches: Prevent over-travel and allow accurate positioning at each stage of the process.

Figure 3.19: Limit Switches

3.2.6 Production line Movement

DC Wiper Motor:Used to drive the conveyor belt to ensure continuous movement of products along the production line. 1-Ch Relay:To control the line on and off at specific locations.

Power supply was used in the Ras El Abed production line to ensure stable and reliable power for the automated machines it provided the necessary voltage and current to operate the mixing, dispensing, and chocolate coating systems efficiently, ensuring the production process runs smoothly.

Figure 3.20: Power Supply

3.2.7 Controller Part

Arduino:Arduino Mega 2560 is used as the main microcontroller for this project due to its advanced features and functionalities. It plays a vital role in controlling various motors, sensors, and relays.

Figure 3.21: Arduino Mega

ESP32: is used in this project to achieve wireless communication and remote control of the system through a mobile application. It plays a key role in monitoring and controlling various stages of the production process.

Figure 3.22: Esp-32

Application interfaces:

Figure 3.23: App

Figure 3.24: App

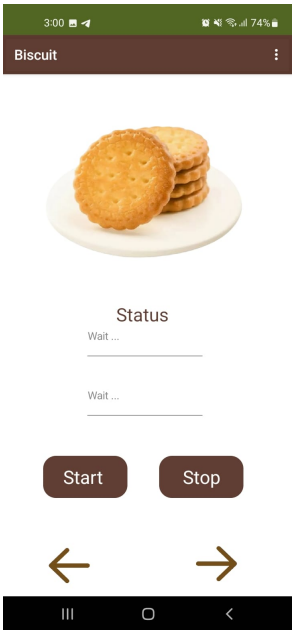

Figure 3.25: App

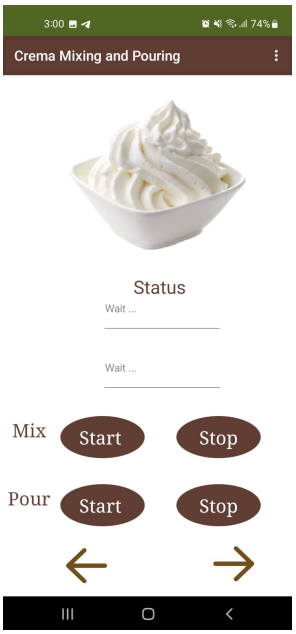

Figure 3.26: App

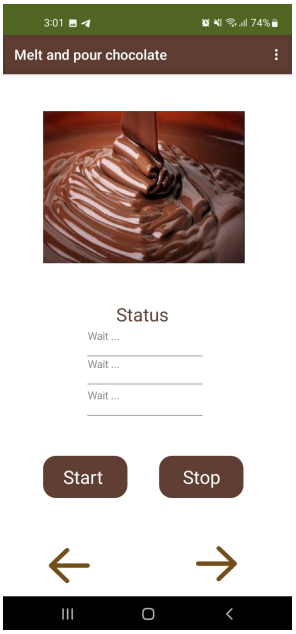

Figure 3.27: App

Figure 3.28: App

3.3 Earlier course work

• Microcontroller: The microcontroller provides basic information to understand the PIC microcontroller and program the hardware components. Also, the labs of this course provide instructions on how to download the PIC microcontroller device code and understand all the pins and functions in it. Therefore, it is one of the most important materials that helps us

understand how to handle the Arduino parts in the project through the knowledge of how to handle the microcontroller, as the labs of this material mainly help us get started into the project easier.

• Electronic circuits: This course has primarily helped us work with electrical circuits and their connections. Consequently, this course offers essential knowledge on managing various circuits and how to connect them properly.

• Embedded systems course

Chapter 4

Methodology

In this project, we relied on many principles of programming and design to develop an integrated production line for producing Ras Al-Abd sweets.

1. Design and Planning: We began by understanding the constraints and standards that governed the project. This included mechanical design complexities due to our team's limited experience in mechanical engineering, and challenges such as inaccuracies in some components, high component costs, and handling issues like cream consistency. Additionally, we referred to earlier course work related to microcontrollers, electronic circuits, and embedded systems, which provided us with a solid foundation for managing hardware components and implementing control systems.

2. Component Selection and Integration:In our project we used many electronic parts Microcontroller and Control System: The Arduino Mega 2560 was chosen as the main microcontroller due to its advanced capabilities for controlling motors, sensors, and relays. Additionally, the ESP32 was integrated to enable wireless communication and remote control through a mobile application. there is a combination of DC motors, stepper motors, servo motors, and relays selected to control the movement of various parts in the production line, including mixing, cream dispensing, and biscuit placement also Ultrasonic sensors and limit switches used for detecting component positions, ensuring accuracy .

4. Hardware Implementation: The production line was designed to move through distinct stages: Biscuit Placement: Servo motors controlled the placement of biscuits onto the conveyor. Cream Mixing: Two mixers were used for whipping the cream, with the DC wiper motor controlling the lifting and lowering of the mixing device. Cream Dispensing: The NEMA 23 stepper motor controlled the movement of the cream press tool for dispensing the cream onto

the biscuits. A DC motor with a gearbox regulated the flow of cream. Chocolate Coating: An electric heater was used to melt the chocolate, with the NEMA 23 stepper motor used to control the opening and closing of the chocolate container.

5. Testing and Evaluation: The system was tested to ensure that each part of the production line functioned as expected. The mixing, cream dispensing, biscuit placement, and chocolate coating processes were tested individually and in combination to ensure accuracy and smooth operation. We iterated on the design, making adjustments as needed, based on the outcomes of our testing.

Chapter 5

Literature Review

This research talks about the importance of automating the food manufacturing process to ensure high efficiency and quality and talks about the reason for avoiding automation, which is the complexity and cost of the process [1].

This research talks about chocolate and its manufacture and discusses the way chocolate is dealt with in terms of covering things [2].

This research talks about dealing with sensors, microcontrollers and ESP [3].

This research includes information about operating DC Motors and Stepper Motors [4].

Chapter 6

Results and Discussion

The results confirmed a significant improvement in the automation of the production of the slave head, as the accuracy became better and the production speed increased.

Despite these improvements, we faced many problems due to the inefficiency of motors and sensors, so there was a percentage of error.

When compared to traditional manufacturing methods, automated production is better in terms of accuracy, speed and cleanliness, but due to a number of challenges, there is a percentage of error that can be improved in the future by improving the parts and algorithms used.

Chapter 7

Conclusion and Future Work

Automation has reduced the change in the shape of the product and the amount of materials covering it. It has also greatly increased the speed of production, reduced human intervention, and reduced the waste of materials through precise control of the components. Despite the presence of many restrictions that affected production, the improvement was noticeable. Among the suggestions for further improvement and overcoming the restrictions:

Designing an integrated system connected to the Internet and using better sensors and motors to increase accuracy, thus ensuring expansion and competition in the market and achieving high success

Bibliographic

[1] Darwin G Caldwell et al. "Automation in food processing". In: *Springer handbook of automation* (2009), pp. 1041–1059.

[2] M Aebi. "Chocolate panning". In: *Industrial Chocolate Manufacture and Use* (2009), pp. 367–385.

[3] Yusuf Kurnia and Jeksen Li Sie. "Prototype of Warehouse Automation System Using Arduino Mega 2560 Microcontroller Based on Internet of Things". In: *bit-Tech* 1.3 (2019), pp. 122–128.

[4] Jayantha Katupitiya and Kim Bentley. "Driving Motors-DC & Stepper". In: *Interfacing with C++ Programming Real-World Applications* (2006), pp. 197–271.

Appendix

A Full Code in C

```
#include <Arduino.h>
#include <Keypad.h>
#include <Servo.h>
#include <Stepper.h>
#include "max6675.h"

#define pushlimitSwitchUp A1
#define pushlimitSwitchDown A2
#define valveOpenPin 40
#define valveClosePin 41
#define directionPin 3
#define stepPin 25

#define stepsPerRevolution 6400

#define railwaylimitSwitchForward 15
#define railwaylimitSwitchReverse 14
#define railwayForward 4
#define railwayReverse 5

#define mixerlimitSwitchUp A0
#define mixerlimitSwitchDown 18
#define mixerOne 26
#define mixerTwo 27
#define mixerUp 7
```

```
#define mixerDown 6



//    IR
#define IR_SENSOR_1_PIN 16 //  IR
#define IR_SENSOR_2_PIN 8  //  IR
#define RELAY_PIN 22       //


const byte ROWS = 1; //
const byte COLS = 2; //   ( )


//     MAX6675
int ktcSO = 12;
int ktcCS = 11;
int ktcCLK = 13;

const int trigPin = 39; //   TRIG
const int echoPin = 38; //   ECHO


long duration;
int distance;
const int THRESHOLD = 6; //
bool biscuitDropped = false; //


//
const int heaterPin = 23; //

//pins nema17
//     H-Bridge
const int IN1 = 50;  //
const int IN2 = 51;  //
const int IN3 = 48; //
```

```
const int IN4 = 49; //


char keys[ROWS][COLS] = {
  {'1', '2'} //
};

byte pin_rows[ROWS] = {32};       //
byte pin_column[COLS] = {31,30};  //
Keypad keypad = Keypad(makeKeymap(keys), pin_rows, pin_column, ROWS, COLS);


//
Servo myServo;
//
Servo myServoUP;    //
Servo myServoDown;  //
Servo myServo2;

volatile bool systemStart = true;
volatile bool stopForward = false;
volatile bool stopReverse = false;

volatile bool stopMixerUP = false;
volatile bool stopMixerDown = false;
volatile bool isMixing = false;


volatile bool isCocoaDone = false;


//
bool creamFlag = false;       //
bool chocolateFlag = false;   //
//bool biscuitDropped = false; //
bool creamIR=false;
bool stepperC=false;
```

```
bool up=false;
#define mixerStartTime 1200000  // 6 seconds in milliseconds (change to 1200000 for 20 minutes)

unsigned long mixerStartMillis = 0;
bool isMixerRunning = false;



#define STEPS_PER_REV 200

//   MAX6675
MAX6675 ktc(ktcCLK, ktcCS, ktcSO);

//nema23
//   Stepper   H-Bridge
Stepper myStepper(STEPS_PER_REV, 46, 47, 44, 45);
//
//nema17
Stepper stepper(STEPS_PER_REV, IN1, IN2, IN3, IN4);
  bool mixersMoveDownFlag=false;
   bool mixersMoveUpFlag=false;
    bool makeMixersMoveUpFlag=false;
  bool mixersOn=false;
  bool makeMixersOn=false;
   bool IRBiscuate=false;
  bool makeIRBiscuate=false;
  bool pushFlag=false;
  bool makePushFlag=false;


void forwardFunction() {
  digitalWrite(railwayForward , HIGH);
  digitalWrite(railwayReverse, LOW);
  Serial.println("Moving forward...");
}

void reverseFunction() {
  digitalWrite(railwayForward , LOW);
```

```
  digitalWrite(railwayReverse, HIGH);
  Serial.println("Moving in reverse...");
}

void stopRailwayMovement() {
  digitalWrite(railwayForward, HIGH);
  digitalWrite(railwayReverse, HIGH);
  Serial.println("Movement stopped.");
}

void mixerMoveUp() {
  Serial.println("Mixer Motor moving up.");
  digitalWrite(mixerUp , LOW);
  digitalWrite(mixerDown, HIGH);
}

void mixerMoveDown() {
  Serial.println("Mixer Motor moving down.");
  digitalWrite(mixerDown , LOW);
  digitalWrite(mixerUp, HIGH);
}

void stopMixersMotor() {
  Serial.println("Mixers Motor stopped.");
  digitalWrite(mixerDown , HIGH);
  digitalWrite(mixerUp, HIGH);
}

void startMixers() {
  Serial.println("Mixers started.");
  digitalWrite(mixerOne , LOW);
  digitalWrite(mixerTwo, LOW);
  mixerStartMillis = millis();  // Store the time when mixers start
  isMixerRunning = true;
}

void stopMixers() {
```

```
  Serial.println("Mixers stopped.");
  digitalWrite(mixerOne , HIGH);
  digitalWrite(mixerTwo, HIGH);
}

bool valveOpen() {
  digitalWrite(valveOpenPin,LOW);
  digitalWrite(valveClosePin, HIGH);
  Serial.println("Valve opening.");

}

bool valveClose() {
  digitalWrite(valveClosePin,LOW);
  digitalWrite(valveOpenPin, HIGH);
  Serial.println("Valve closing");

}

void stopValveMovement() {
  digitalWrite(valveClosePin,HIGH);
  digitalWrite(valveOpenPin, HIGH);
  Serial.println("Valve Stoped");
}

void coacoAdding(char key){

  if (key) {
    Serial.print("Key pressed: ");
    Serial.println(key);

    if (key == '1') {            //      1
      myServo.write(0);         //    0
      delay(2000);              //
      myServo.write(90);        //      (90 )
    } else if (key == '2') {     //      2
      myServo.write(180);       //    180
```

```
      delay(2000);              //
      myServo.write(90);        //      (90 )
    }
  }

}

void moveStepper(int direction, int speed, int steps) {
  digitalWrite(directionPin, direction); // Set the direction of the stepper

  int delayTime = speed; // Convert speed (steps per second) to delay in microseconds

  for (int i = 0; i < steps; i++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(delayTime / 2); // Half the delay for HIGH state
    digitalWrite(stepPin, LOW);
    delayMicroseconds(delayTime / 2); // Half the delay for LOW state
  }
}
int measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  int calculatedDistance = duration * 0.034 / 2;
  return calculatedDistance;
}

void dropBiscuit() {
  distance = measureDistance();
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
```

```
  if (distance > THRESHOLD)  //
    Serial.println("  !   ...");
    if (!biscuitDropped) {
      myServoUP.write(0);
      delay(170);
      myServoUP.write(72);
      delay(2000);

      myServoDown.write(0);
      delay(500);
      myServoDown.write(100);
      delay(800);
      biscuitDropped = true;


  }
}
void handleChocolate() {

  double tempC = ktc.readCelsius();
  Serial.print("Chocolate Temperature (C): ");
  Serial.println(tempC);

  Serial.println("Chocolate heated. Opening nozzle...");
  for (int i = 0; i < 6; i++) {
    myStepper.step(STEPS_PER_REV); //
    // stepper.step(STEPS_PER_REV);
  }
       openServo();
      digitalWrite(RELAY_PIN, HIGH);
       delay(100);
        digitalWrite(RELAY_PIN, LOW);
       delay(100);
        digitalWrite(RELAY_PIN, HIGH);
       delay(100);
          digitalWrite(RELAY_PIN, LOW);
       delay(100);
```

```
            digitalWrite(RELAY_PIN, HIGH);
        delay(100);
        digitalWrite(RELAY_PIN, LOW);

  delay(15000);
  for (int i = 0; i < 6; i++) {
    myStepper.step(-STEPS_PER_REV); //
    // stepper.step(STEPS_PER_REV);
  }
  closeServo();

  Serial.println("Nozzle closed.");
}



void openServo() {
  myServo2.write(120);
    delay(700);
    myServo2.write(90);
}

void closeServo() {
   myServo2.write(60);
    delay(430);
    myServo2.detach();
}



void setup() {
  Serial.begin(9600);

  pinMode(railwaylimitSwitchForward, INPUT_PULLUP);
  pinMode(railwaylimitSwitchReverse, INPUT_PULLUP);
  pinMode(railwayForward, OUTPUT);
  pinMode(railwayReverse, OUTPUT);

  pinMode(mixerlimitSwitchUp, INPUT_PULLUP);
```

```
 pinMode(mixerlimitSwitchDown, INPUT_PULLUP);
 pinMode(mixerOne, OUTPUT);
 pinMode(mixerTwo, OUTPUT);
 pinMode(mixerUp, OUTPUT);
 pinMode(mixerDown, OUTPUT);


pinMode(pushlimitSwitchUp, INPUT_PULLUP);
   pinMode(pushlimitSwitchDown, INPUT_PULLUP);

   // Initialize output pins
   pinMode(valveOpenPin, OUTPUT);
   pinMode(valveClosePin, OUTPUT);
   pinMode(directionPin, OUTPUT);
   pinMode(stepPin, OUTPUT);

 digitalWrite(mixerOne, HIGH);
 digitalWrite(mixerTwo, HIGH);

digitalWrite(valveClosePin,HIGH);
 digitalWrite(valveOpenPin, HIGH);
  myServo.attach(33);             //     D9
 myServo.write(90);



  myServoUP.attach(37);
 myServoDown.attach(36);
 myServoUP.write(72);
 myServoDown.write(100);
 pinMode(IR_SENSOR_1_PIN, INPUT);
 pinMode(IR_SENSOR_2_PIN, INPUT);
 pinMode(RELAY_PIN, OUTPUT);
digitalWrite(RELAY_PIN,HIGH);
 pinMode(heaterPin, OUTPUT);
 digitalWrite(heaterPin, HIGH);
 myStepper.setSpeed(30);
```

```
//nema17
   stepper.setSpeed(30);
  //nema17
  //
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);

    pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
   myServo2.attach(A3);    // Attach second servo to pin 10

  myServo2.write(90);
}

static bool startupComplete = false;
char key='T';
void loop() {
    while(key!='1'&&key!='2'){
     key=keypad.getKey();
    // mixerMoveUp();
   //forwardFunction();
    }

 if (!startupComplete) {
    startupComplete = true;
     reverseFunction();
  }
 /*if(Serial.available()>0){

    String input = Serial.readStringUntil('\n');
    input.trim();
    if(input.equals("b")){
      dropBiscuit();
    }
```

```
  else if(input.equals("R")){
    String result="";
        if(biscuitDropped){
          result = "Dropped";
        }else {
              result = "Un Dropped";
        }
        result = result +","+ String(5);
        Serial.print(result);
  }
  else if(input.equals("S")){

  }else if(input.equals("start_choc")){
          handleChocolate();

  }else if(input.equals("ref_choc")){
     String resultC="";
        if(Heater){
          resultC = "Heater ON : "+ String(tempC);
        }else {
              resultC = "Heater Off";
        }
        if(isCocoaDone){
          resultC = resultC +","+ "Openned" ;

        }
        else{
            resultC = resultC + "," +"Closed" ;
        }
        Serial.print(resultC);


  }



}*/
```

```
if (digitalRead(railwaylimitSwitchReverse) == LOW) {  // Reverse switch pressed
 if(!mixersMoveDownFlag) {
  mixersMoveDownFlag=true;
  forwardFunction();
  delay(1500);
  stopRailwayMovement();
  coacoAdding(key);
  mixerMoveDown();  // Start moving the mixer down when reverse button is pressed
  makeMixersOn=true;
 }
}

if (digitalRead(mixerlimitSwitchDown) == LOW) {  // Mixer Down switch pressed
if(!mixersOn && makeMixersOn){
   makeMixersOn=false;
  mixersOn=true;
  mixerMoveUp();
  delay(2000);
  stopMixersMotor();
  if (!isMixerRunning) {
    startMixers();  // Start mixers when mixerDown switch is pressed
  }
}
}



// If mixers are running and 20 minutes have passed (change to 1200000 ms for 20 minutes)
if (isMixerRunning && (millis() - mixerStartMillis >= mixerStartTime)) {
  stopMixers();  // Stop the mixers after 20 minutes
  mixerMoveUp();  // Move the mixer up after 20 minutes
  delay(5000);
  isMixerRunning = false;
  makeMixersMoveUpFlag=true;
}

if (digitalRead(mixerlimitSwitchUp) == LOW) {  // Mixer Up switch pressed
```

```
  if(!mixersMoveUpFlag&&makeMixersMoveUpFlag)
{
    makeIRBiscuate=true;
    mixersMoveUpFlag=true;
     makeMixersMoveUpFlag=false;
    mixerMoveDown();
    delay(1500);
    stopMixersMotor();
   forwardFunction();
   dropBiscuit();

  }
  }


    if (digitalRead(IR_SENSOR_1_PIN) == LOW) {

      if(!IRBiscuate&&makeIRBiscuate){
         digitalWrite(RELAY_PIN, LOW);


    if (creamFlag) {
      creamFlag = false;
      delay(500);
      digitalWrite(RELAY_PIN, HIGH);
      delay(3000);
      creamIR=true;
      chocolateFlag=true;
        IRBiscuate=true;
         makeIRBiscuate=false;
    }

        makePushFlag=true;
      }
  }
```

```
if (digitalRead(railwaylimitSwitchForward) == LOW) {
 if(!pushFlag&&makePushFlag){

   reverseFunction();
   delay(1500);
   stopRailwayMovement();
   valveOpen();
   delay(5000);
   stopValveMovement();
   for(int i=0;i<50;i++)
   moveStepper(LOW, 70, stepsPerRevolution);
   for(int i=0;i<60;i++){
      moveStepper(LOW, 150, stepsPerRevolution);
      delay(100);///test
      if(digitalRead(pushlimitSwitchDown)==LOW)
         break;


   }

 pushFlag=true;
 makePushFlag=false;

   creamFlag=true;
 }
 }
```

```
if (digitalRead(pushlimitSwitchDown) == LOW) {
 if(creamIR==true&&isCocoaDone==true){
   creamIR=false;
   isCocoaDone=false;
```

```
  while (digitalRead(pushlimitSwitchUp) == HIGH) {
     moveStepper(HIGH, 150, stepsPerRevolution);
   }
    reverseFunction();
  delay(5000);
      valveClose();
           delay(5000);
           stopValveMovement();

 }
}



 if (digitalRead(IR_SENSOR_2_PIN) == LOW) {
   digitalWrite(RELAY_PIN, LOW);

   if (chocolateFlag) {
     handleChocolate();
    chocolateFlag = false;
     isCocoaDone =true;
     delay(500);
     //digitalWrite(RELAY_PIN, HIGH);
     delay(3000);


       mixersMoveDownFlag=false;
        mixersMoveUpFlag=false;
  makeMixersMoveUpFlag=false;
 mixersOn=false;
 makeMixersOn=false;
  IRBiscuate=false;
 makeIRBiscuate=false;
 pushFlag=false;
 makePushFlag=false;
```

```
    }
  }
}
```