الجامعة الإسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يونيبرسيتي إسلام انتارابغسا مليسيا

(Company No. 101067-P)

Garden of Knowledge and Virtue

MCTE 3203

MECHATRONICS SYSTEM INTEGRATION

SEM 2 23/24

SECTION 1

Mini Project Report :

Washing Machine

Group C:

NO.	STUDENT'S NAME	MATRIC NUMBER
1.	Abdallah Mahamat Abakar Maeyd	2117777
2.	KHAN TOQI TAHMID	2025209
3.	Almasri Suhail Jihad	2128771
4.	Ibrahim Bin Nasrum	2116467

TABLE OF CONTENTS

1.0 Introduction

Embedded systems are a crucial part of modern technology, seamlessly blending into our everyday lives and serving important roles in various devices. These systems are designed to perform specific tasks within larger systems, often improving efficiency and functionality for particular purposes.

From simple household appliances like microwaves and washing machines to advanced industrial controls and smart IoT devices, embedded systems are everywhere.

At their core, embedded systems consist of a microcontroller or microprocessor, memory, and various peripherals working together to complete specific tasks. The performance and reliability of an embedded system depend largely on how well its components interact and communicate. This careful coordination is what enables the system to function effectively within its intended environment.

2.0 objectives

- Interfacing: Aims to enable communication and interaction between components within a system.
- System Integration: Aims to create a seamless and functional whole by coordinating interactions between different subsystems.
- Embedded System: Aims to perform specific functions within a larger system or device.

3.0 Design Problem

This particular project involves designing and creating a basic automatic washing machine using a microcontroller called Arduino. The machine should include the washing cycle the filling process, the wash process, the rinse process, the spinning and the drainage process. A few of the components that will be used as subsystems include sensors, actuators and a user interface that will help mimic the functioning of a washing machine.

4.0 Equipment / Component

Component	Its function	Picture
Arduino Mega	Serves as the central control unit, managing inputs from sensors and executing the washing machine's program logic.	
LCD Display (I2C, 20x4)	Provides a user interface to display operational status and settings of the washing machine.	
HW-834 8 Pin 16 Push Button Module	Offers a keypad for user input, allowing selection of wash settings and control commands.	
Stepper Motor (US-17HS4401)	Drives the rotation of the washer drum, allowing precise control over its speed and position for different wash cycles.	

L298N Motor Driver Module	Powers and controls the stepper motor, enabling bidirectional rotation and speed regulation.	
2 Relays Module	Acts as a switch to control high-power components, like water pumps, using low-power signals from the Arduino.	
LEDs (Red, Yellow, Green)	Indicate different statuses of the washing machine, such as power on, washing in progress, and completed cycles.	
Water Pumps (Inlet and Outlet)	Manage water flow into and out of the washing machine, controlling the filling and draining of the washer drum.	

<table>
<tr>
<td>RC522 and valid, invalid card (rfid reader and writer at 13.56mhz)</td>
<td>RC522 RFID Reader/Writer is used to detect and authenticate RFID cards, allowing access to the washing machine's control system when a valid card is scanned. Valid cards will permit the machine to operate, while invalid cards will deny access, ensuring secure and authorized use.</td>
<td>

</td>
</tr>
<tr>
<td>150 ohm resistors</td>
<td>Used to limit the current flowing through the LEDs and other components, protecting them from damage due to excessive current.</td>
<td>
</td>
</tr>
</table>

5.0 Experiment setup process

5.1 Circuit Assembly:

- Connect the LCD display to the Arduino using the I2C interface.
- Wire the push button module to the Arduino, connecting the columns (colPins) and rows (rowPins) as defined in the code.
- Connect the LEDs to the designated pins (redLedPin, yellowLedPin, greenLedPin) on the Arduino.
- Wire the stepper motor to the L298N motor driver module and connect the driver module to the Arduino (IN1Pin, IN2Pin, IN3Pin, IN4Pin).
- Connect the relay module to the Arduino and wire the relays to the water pumps (relayPin1 for the inlet pump, relayPin2 for the outlet pump)

5.2 Component Initialization:

- Initialize the LCD display, push-button module, LEDs, stepper motor, and relay module in the Arduino setup() function.
- Ensure that all output pins are set to the correct initial states (e.g., relays are off, LEDs are off).

5.3 User Interface and Control:

- Display a welcome message on the LCD and prompt the user to press the start button to begin the washing cycle.
- Implement button press detection logic to monitor user inputs and transition between different states of the washing machine.

5.4 State Machine Logic:

- Define the various states of the washing machine: IDLE, FILLING, WASHING, RINSING, SPINNING, DRAINING, and FINISHED.
- Use a switch-case structure in the loop() function to manage the washing machine's operations based on the current state.
- Implement the specific actions for each state, such as activating relays for water pumps, controlling the stepper motor, and updating the LCD display.
-

6.0 Methodology

6.1 Power Up and Initialization:

- Power up the Arduino and initialize all connected components.
- Display the initial message on the LCD.

6.2 User Interaction:

- Wait for the user to press the start button to begin the washing cycle.

- Continuously monitor the push button module to detect button presses.

6.3 Cycle Execution:

- Transition through each stage of the washing machine based on the control logic:
    - IDLE: Wait for the start button press.
    - FILLING: Activate the inlet water pump relay for a specified duration.
    - WASHING: Control the stepper motor to simulate washing action (forward and backward rotations).
    - RINSING: Perform rinse action using the stepper motor.
    - DRAINING: Activate the outlet water pump relay to drain water.
    - SPINNING: Spin the drum using the stepper motor.
    - FINISHED: Indicate the completion of the washing cycle and return to the IDLE state.

6.4 Observation and Debugging:

- Use the serial monitor for debugging and observing the washing machine's state transitions and actions.
- Adjust delay times and other parameters based on observations to optimize the washing cycle.

7.0 Code

7.1 main code

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>


// Define LCD with I2C address 0x27 and size 20x4

LiquidCrystal_I2C lcd(0x27, 20, 4);


// Define push button module pin numbers

const int colPins[] = {30, 31, 32, 33}; // Columns (output)

const int rowPins[] = {34, 35, 36, 37}; // Rows (input)

int buttonStates[4][4]; // To store the state of each button


// Define LED pin numbers

const int redLedPin = 38;

const int yellowLedPin = 39;

const int greenLedPin = 40;
```

```
// Define L298N motor driver pins for stepper motor (US-17HS4401)
const int IN1Pin = 9;
const int IN2Pin = 10;
const int IN3Pin = 11;
const int IN4Pin = 12;

// Define relay pins for water pumps
const int relayPin1 = 5; // Water pump 1 (inlet)
const int relayPin2 = 6; // Water pump 2 (outlet)

// Washing machine states
enum WashingState {
 IDLE,
 FILLING,
 WASHING,
 RINSING,
 SPINNING,
 DRAINING,
 FINISHED
};

WashingState currentState = IDLE; // Initial state

void setup() {
 // Initialize LCD
 lcd.init();
 lcd.backlight();

 // Initialize push button module columns as outputs
 for (int i = 0; i < 4; i++) {
  pinMode(colPins[i], OUTPUT);
  digitalWrite(colPins[i], HIGH); // Set to HIGH initially
 }
```

```
 // Initialize push button module rows as inputs with pull-up resistors
 for (int i = 0; i < 4; i++) {
   pinMode(rowPins[i], INPUT_PULLUP);
 }

 // Initialize LED pins as outputs
 pinMode(redLedPin, OUTPUT);
 pinMode(yellowLedPin, OUTPUT);
 pinMode(greenLedPin, OUTPUT);

 // Initialize stepper motor driver pins as outputs
 pinMode(IN1Pin, OUTPUT);
 pinMode(IN2Pin, OUTPUT);
 pinMode(IN3Pin, OUTPUT);
 pinMode(IN4Pin, OUTPUT);

 // Initialize relay pins for water pumps as outputs
 pinMode(relayPin1, OUTPUT);
 digitalWrite(relayPin1, HIGH); // Ensure relay1 is initially off
 pinMode(relayPin2, OUTPUT);
 digitalWrite(relayPin2, HIGH); // Ensure relay2 is initially off

 // Start serial communication for debugging
 Serial.begin(9600);

 // Initial LCD display message
 lcd.setCursor(0, 0);
 lcd.print("Welcome!");
 lcd.setCursor(0, 1);
 lcd.print("Press start");
}

void loop() {
```

```
// Example washing machine operation loop
lcd.setCursor(0, 1); // Move to second line for status updates

// Simulate button press detection
detectButtonPress();

// State machine logic for automatic washing machine
switch (currentState) {
 case IDLE:
   digitalWrite(redLedPin, LOW);
   digitalWrite(yellowLedPin, LOW);
   digitalWrite(greenLedPin, HIGH); // Green LED on
   lcd.print("Press start      ");
   // Check for start button press to begin washing cycle
   if (buttonStates[0][0] == LOW) {
     currentState = FILLING;
     delay(500); // Debounce delay
   }
   break;

 case FILLING:
   digitalWrite(greenLedPin, LOW);
   digitalWrite(yellowLedPin, HIGH); // Yellow LED on
   lcd.print("Filling...      ");
   digitalWrite(relayPin1, LOW); // Activate relay1 to start water pump (inlet)
   delay(5000); // Simulated fill cycle duration (5 seconds)
   digitalWrite(relayPin1, HIGH); // Deactivate relay1 to stop water pump (inlet)
   currentState = WASHING;
   delay(3000); // Transition delay
   break;

 case WASHING:
   digitalWrite(greenLedPin, LOW);
   digitalWrite(yellowLedPin, HIGH); // Yellow LED on
```

```
  lcd.print("Washing...      ");
  for (int i = 0; i < 2; i++) {
    startStepperMotor(); // Start stepper motor for wash cycle (forward)
    delay(3000); // Simulated wash cycle duration (3 seconds)
    stopStepperMotor();
    delay(1000); // Short delay before reversing

    reverseStepperMotor(); // Start stepper motor for wash cycle (backward)
    delay(3000); // Simulated wash cycle duration (3 seconds)
    stopStepperMotor();
    delay(1000); // Short delay before next cycle
  }
  currentState = RINSING;
  delay(5000); // Transition delay
  break;

case RINSING:
  lcd.print("Rinsing...      ");
  startStepperMotor(); // Start stepper motor for rinse cycle
  delay(7000); // Simulated rinse cycle duration (7 seconds)
  stopStepperMotor();
  currentState = SPINNING;
  delay(2000); // Transition delay
  break;

case SPINNING:
  lcd.print("Spin cycle      ");
  spinStepperMotor(); // Start stepper motor for spin cycle
  delay(5000); // Simulated spin cycle duration (5 seconds)
  stopStepperMotor();
  currentState = DRAINING;
  break;

case DRAINING:
```

```
    lcd.print("Draining...     ");
    digitalWrite(relayPin2, LOW); // Activate relay2 to start water pump (drain)
    delay(5000); // Simulated drain cycle duration (5 seconds)
    digitalWrite(relayPin2, HIGH); // Deactivate relay2 to stop water pump (drain)
    currentState = FINISHED;
    break;

   case FINISHED:
    digitalWrite(yellowLedPin, LOW);
    digitalWrite(redLedPin, HIGH); // Red LED on
    lcd.print("Finished       ");
    break;
 }
 delay(200); // Delay for stability and to reduce processing load
}
// Function to detect button presses
void detectButtonPress() {
 // Scan the buttons
 for (int col = 0; col < 4; col++) {
  digitalWrite(colPins[col], LOW); // Set the current column to LOW
  for (int row = 0; row < 4; row++) {
    buttonStates[col][row] = digitalRead(rowPins[row]); // Read the state of the current button
  }
  digitalWrite(colPins[col], HIGH); // Set the current column back to HIGH
 }
}
// Function to start the stepper motor for washing cycle
void startStepperMotor() {
 for (int i = 0; i < 1000; i++) { // Number of steps
  stepMotor(1); // Clockwise
  delay(1); // Adjust delay for desired speed
 }
}
// Function to reverse the stepper motor for washing cycle
```

```
void reverseStepperMotor() {
  for (int i = 0; i < 1000; i++) { // Number of steps
    stepMotor(0); // Counterclockwise
    delay(1); // Adjust delay for desired speed
  }
}
// Function to start the stepper motor for spin cycle
void spinStepperMotor() {
  for (int i = 0; i < 2000; i++) { // Number of steps
    stepMotor(0); // Counterclockwise
    delay(1); // Adjust delay for desired speed
  }
}

// Function to step the motor
void stepMotor(int direction) {
  if (direction == 1) { // Clockwise
    digitalWrite(IN1Pin, HIGH);
    digitalWrite(IN2Pin, LOW);
    digitalWrite(IN3Pin, HIGH);
    digitalWrite(IN4Pin, LOW);
    delayMicroseconds(1000); // Adjust for speed
    digitalWrite(IN1Pin, LOW);
    digitalWrite(IN2Pin, HIGH);
    digitalWrite(IN3Pin, HIGH);
    digitalWrite(IN4Pin, LOW);
    delayMicroseconds(1000); // Adjust for speed
    digitalWrite(IN1Pin, LOW);
    digitalWrite(IN2Pin, HIGH);
    digitalWrite(IN3Pin, LOW);
    digitalWrite(IN4Pin, HIGH);
    delayMicroseconds(1000); // Adjust for speed
    digitalWrite(IN1Pin, HIGH);
    digitalWrite(IN2Pin, LOW);
```

```
    digitalWrite(IN3Pin, LOW);
    digitalWrite(IN4Pin, HIGH);
    delayMicroseconds(1000); // Adjust for speed
  } else { // Counterclockwise
    digitalWrite(IN1Pin, HIGH);
    digitalWrite(IN2Pin, LOW);
    digitalWrite(IN3Pin, LOW);
    digitalWrite(IN4Pin, HIGH);
    delayMicroseconds(1000); // Adjust for speed
    digitalWrite(IN1Pin, LOW);
    digitalWrite(IN2Pin, HIGH);
    digitalWrite(IN3Pin, LOW);
    digitalWrite(IN4Pin, HIGH);
    delayMicroseconds(1000); // Adjust for speed
    digitalWrite(IN1Pin, LOW);
    digitalWrite(IN2Pin, HIGH);
    digitalWrite(IN3Pin, HIGH);
    digitalWrite(IN4Pin, LOW);
    delayMicroseconds(1000); // Adjust for speed
    digitalWrite(IN1Pin, HIGH);
    digitalWrite(IN2Pin, LOW);
    digitalWrite(IN3Pin, HIGH);
    digitalWrite(IN4Pin, LOW);
    delayMicroseconds(1000); // Adjust for speed
  }
}
// Function to stop the stepper motor
void stopStepperMotor() {
  digitalWrite(IN1Pin, LOW);
  digitalWrite(IN2Pin, LOW);
  digitalWrite(IN3Pin, LOW);
  digitalWrite(IN4Pin, LOW);
}
```

7.2 WASHER SAFE SWITCHING SYSTEM

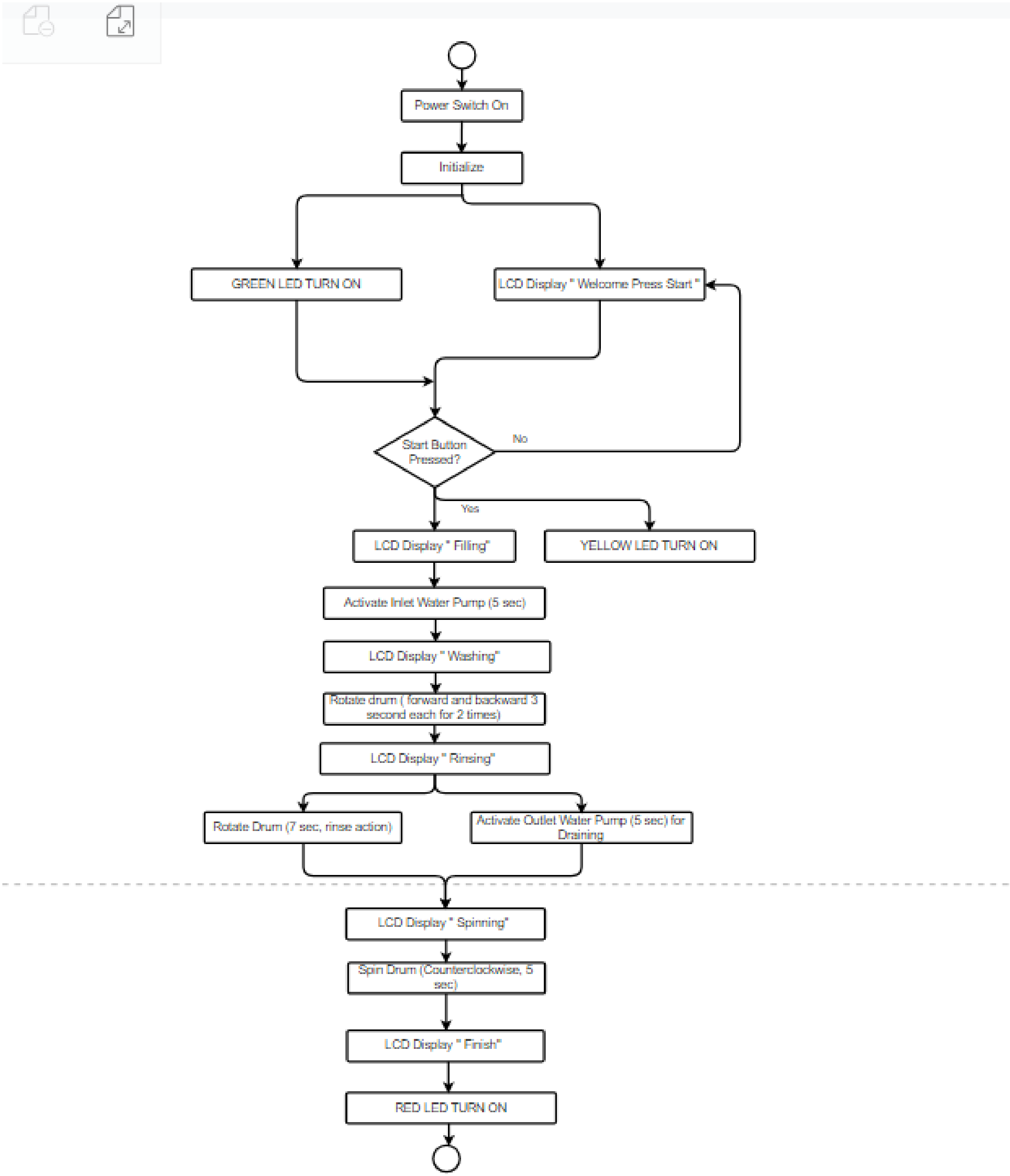
CODE :

```
#include <SPI.h>
#include <MFRC522.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define SS_PIN 10
#define RST_PIN 9
#define GREEN_LED_PIN 7
#define RED_LED_PIN 8
#define SIGNAL_PIN 6 // Define the pin for the 5V output signal

MFRC522 mfrc522(SS_PIN, RST_PIN);
LiquidCrystal_I2C lcd(0x27, 20, 4);  // Set the LCD address to 0x27 for a 20 chars and 4 line display

const String VALID_RFID = "12307989";

void setup() {
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();
  pinMode(GREEN_LED_PIN, OUTPUT);
  pinMode(RED_LED_PIN, OUTPUT);
  pinMode(SIGNAL_PIN, OUTPUT);
  digitalWrite(SIGNAL_PIN, LOW); // Ensure the signal pin is initially LOW

  lcd.init();
  lcd.backlight();

  printInitialMessages();
}
```

```
void loop() {
  if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
    String rfid = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      rfid += String(mfrc522.uid.uidByte[i], HEX);
    }
    rfid.toUpperCase();
    if (rfid == VALID_RFID) {
      toggleGREENLED();
    } else {
      indicateInvalidRFID();
    }
    delay(1000); // 1 second delay before reading another RFID
    printInitialMessages();
  }
}
void printInitialMessages() {
  Serial.println("GOOD DAY FOR A WASH");
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("GOOD DAY FOR A WASH");
  if (digitalRead(GREEN_LED_PIN) == HIGH) {
    Serial.println("PLACE THE SETUP RFID TO TURN OFF THE WASHER");
    lcd.setCursor(0, 1);
    lcd.print("PLACE RFID TO STOP  ");
  } else {
    Serial.println("PLACE THE SETUP RFID TO TURN ON THE WASHER");
    lcd.setCursor(0, 1);
    lcd.print("PLACE RFID TO START ");
  }
}
```

```
void toggleGREENLED() {
  if (digitalRead(GREEN_LED_PIN) == HIGH) {
    digitalWrite(GREEN_LED_PIN, LOW);
    digitalWrite(SIGNAL_PIN, LOW); // Turn off the 5V output signal
    Serial.println("WASHER -> TURNED OFF");
    Serial.println(" ");
    lcd.setCursor(0, 2);
    lcd.print("WASHER -> TURNED OFF");
  } else {
    digitalWrite(GREEN_LED_PIN, HIGH);
    digitalWrite(SIGNAL_PIN, HIGH); // Turn on the 5V output signal
    Serial.println("WASHER -> TURNED ON");
    Serial.println(" ");
    lcd.setCursor(0, 2);
    lcd.print("WASHER -> TURNED ON");
  }
}

void indicateInvalidRFID() {
  digitalWrite(RED_LED_PIN, HIGH);
  Serial.println("INVALID RFID FOUND");
  Serial.println(" ");
  lcd.setCursor(0, 3);
  lcd.print("INVALID RFID FOUND");
  delay(1000); // 1 second delay
  digitalWrite(RED_LED_PIN, LOW);
  lcd.setCursor(0, 3);
  lcd.print("                    "); // Clear the specific line
}
```

8.0 Flow chart of washing machine



9.0 Discussion

When the code is uploaded and the system is powered on, the following sequence of events occurs: the LCD refreshes with a ‘welcome to ‘ message while the system initializes and waits for the user to press the start button. Thus the process of the system in this state is invite and only when the ‘start’ button is pressed does the system enter into the active working stage. It in the filling state, the inlet water pump switches on to admit water into the washing chamber for a particular time interval. Eventually, during the washing state, the stepper motor in the machine rotates in the forwards and backward direction to mimic the washing action. The rinsing state is the next step where the stepper motor rotates the drum to rinse the clothes, after which the draining state occurs where the outlet water pump expels water from the chamber. Finally, in the spinning state, the motor causes the drum to spin and remove excess water from the clothes. The system then enters the finished state, indicating the end of the washing cycle.

Employing the Advanced Outdoor Washer Safe Switching System as a feature to our washing machine prototype offers improved safety for personal property as well as restricted operation to only those who own functional RFID cards that they biologically acquire through subscription. Using the RFID card, the user establishes identification upon approaching the machine reader for starting or stopping processes thereby providing safe switching avoiding operator interference. This system not only confuses the identity of the user but also ensures that when the RFID cards are damaged or lost, the replacement process is so fast, making access convenient with a lot of security in the outside environment.

9.1 Reflection and Challenges Encountered

9.1.1 Challenges Encountered

- Component Integration: Ensuring all components worked seamlessly together required careful wiring and pin configuration.
- Debouncing Issues: Reliable detection of button presses was initially problematic, necessitating the implementation of debouncing logic.
- Timing Adjustments: Fine-tuning delay times for each state transition was critical to simulate realistic washing durations without overloading the system.
- Power Management: Ensuring stable power supply and voltage regulation was essential to prevent component damage.

9.1.2 Reflections

- The project provided a comprehensive understanding of system integration, interfacing, and embedded systems. Each component's role within the larger system highlighted the importance of coordination and control.
- Iterative testing and debugging were crucial in refining the washing machine's functionality, demonstrating the importance of a systematic approach in engineering projects.

9.2 Advantages and Disadvantage

9.2.1 Advantages

- Automated Control: The washing machine is a machine for washing clothes; all the activities required to wash the clothes are done by this machine and no intervention is required.
- User-Friendly Interface: The LCD display and the push button make up a user-friendly control panel where users can interact with the device.
- Energy Efficiency: Stepper motor and relays in controlling and switching on/off are managed intelligently and energy optimally.
- Customization: The parameters of the washing cycle are also flexible, and can be changed easily if some other modifications are carried out in the code, with the intent of excluding some definite types of laundry.

9.2.2 Disadvantages

While the washing machine prototype offers several advantages, there are also some limitations and potential disadvantages to consider:

- Durability: The prototype may not be as robust as store-bought machines due to its use of locally available parts and simpler mechanisms.
- Lack of Features: It lacks advanced functions like automatic detergent dispensers, additional washing modes, and a built-in drying system.
- Manual Adjustments: Changes to washing parameters require direct modifications to the code, which may not be user-friendly for non-programmers.
- Mechanical Reliability: Components such as the stepper motor and relays may require frequent replacement compared to commercial-grade parts.
- Water Management: Without advanced sensors, it may experience occasional inefficiencies or water wastage.
- Safety: It lacks comprehensive safety certifications and may pose risks if not properly maintained or used.
- Scalability: Scaling up for larger loads or integrating more features would require significant redesign and additional resources.

9.3 Future Improvements

- Enhanced User Interface: It may also be possible to get greater interactivity if more buttons are added or, instead, if a touchscreen is used to take inputs.
- Sensors Integration: Mentioned parameters, including the water level and temperature, could be used to enhance the amount of washing efficiency.
- Wireless Control: Wireless communication could be another feature to be applied to the washing machine, which would allow its remote control and monitoring.

10.0 Safety

Some Important Safety Factors in the Designing of the Washing Machine Prototype. Design for safety should be acknowledged in the washing machine prototype to avoid potential risks, safeguard the parts and components, and perform effectively. The following measures have been implemented:

10.1 Electrical Safety

- Insulation and Shielding: The various wires that make the electrical connections are all enclosed and covered to avoid short circuits or accidental contact. Thus, sensitive components of the hardware are protected from the effects of electromagnetic interference.
- Overcurrent Protection: Relay or fuse or circuit breakers are incorporated to avoid damages due to over current or short circuit.
- Voltage Regulation: Voltage stabilization is performed through voltage regulators and protects equipment and electronic circuits from overvoltage destruction.

10.2 Mechanical Safety

- Component Mounting: Every element is fixed with extreme measures in order to avoid translation, oscillation or even detachment when the machine is running. Further, the stepper motor used in the design will be correctly positioned to help reduce mechanical stress.
- Motor and Relay Safety: It is ensured that the motor and relay loads are rated adequately with no overloading that leads to heat up or eventual failure. insufficient flow of fresh and clean air, while adequate cooling requirements are met.

10.3 User Safety

- Safe Operation Messages: Its control is very simple and gives good feedback through a LCD display that informs users on the status of the machine.
- Emergency Stop: A specific button to stop the entire process bring the machine to a stop and neutral set up at once.
- Physical Safety: Contact sensitive components or parts are fully enclosed or protected and the geographic orientation does not contain sharp edges that could result in customers' injury.

10.4 Water Safety

- Leakage Prevention: All water connections are well closed in a way that no contamination by water can reach the circuit and cause electricity leakage or destruction.
- Water Pump Control: Watering regimes are managed through relays controlling the water pumps thus ensuring that water is used in the correct measure to avoid water wastage.
- These measures help to ensure that the washing machine prototype is safe to use by the public; the users, the components used in the construction of the washing machine, and that it performs its function efficiently.

11.0 Conclusion and Further Works

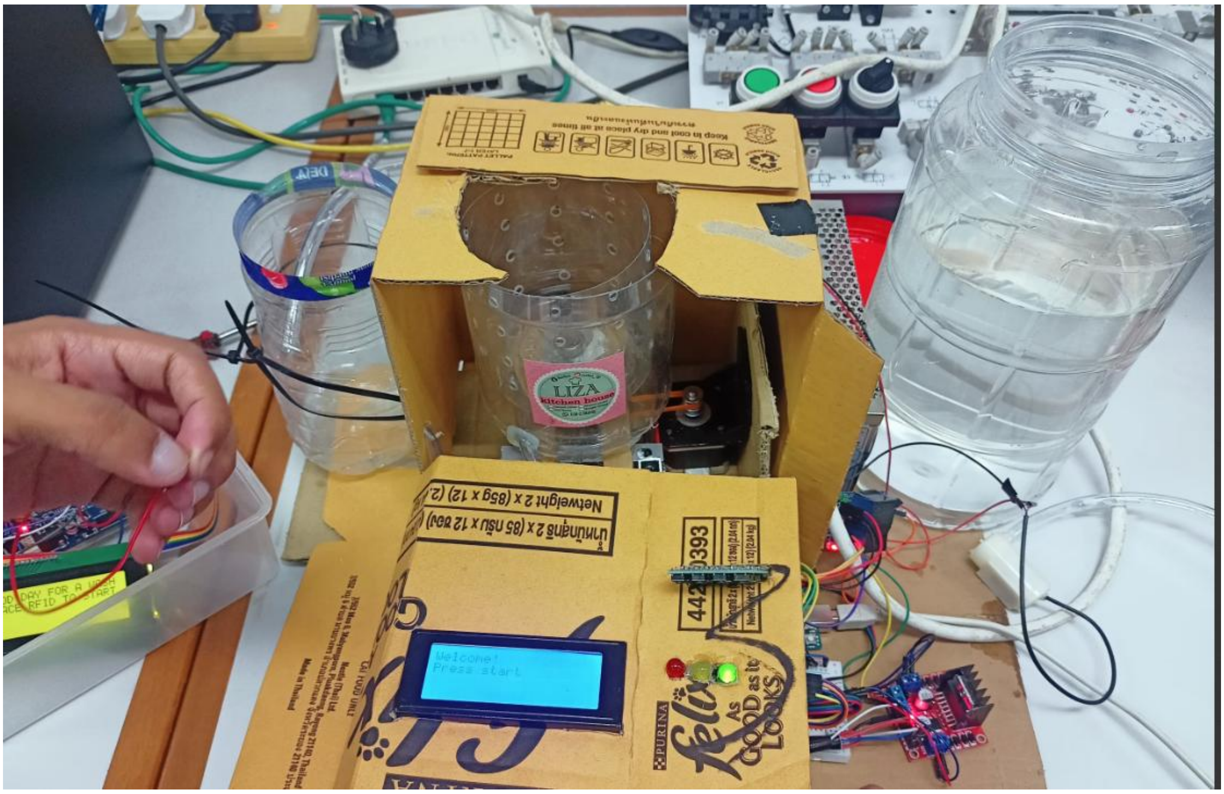
By the end of this project, we will have gained a thorough understanding of how embedded systems are designed, developed, and interfaced. This hands-on experience will illustrate the critical role that effective interfacing and integration play in the functionality and reliability of these systems. We will explore how to connect and manage various components such as motors, sensors, and displays within an embedded system, emphasizing their interplay and synchronization.

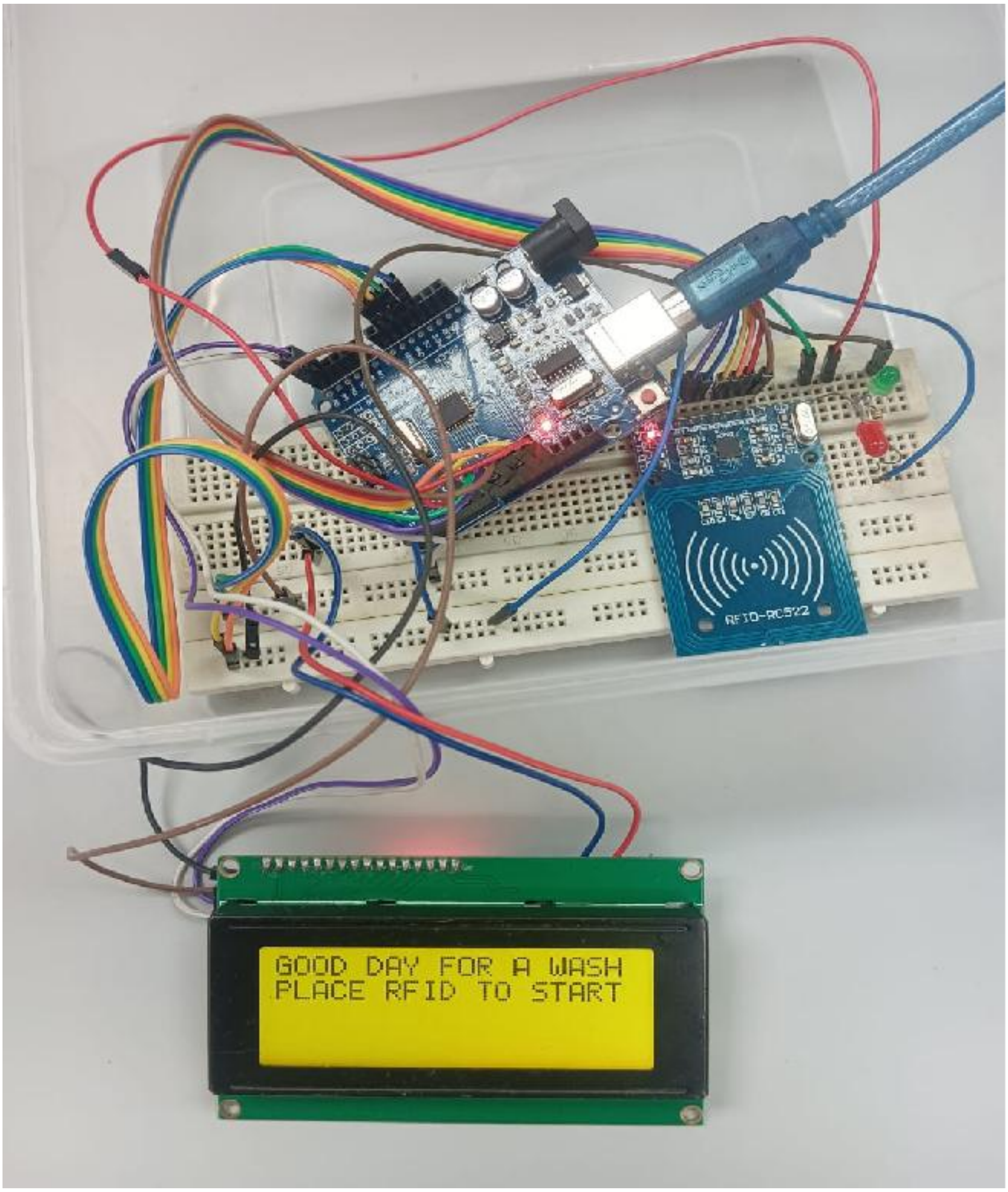
Additionally, this project will lay a strong foundation for future exploration into more advanced topics in embedded and IoT technologies. We will be better equipped to understand how these technologies can be scaled and adapted for more complex applications. As we look ahead, this knowledge will be invaluable for delving into sophisticated IoT systems that integrate numerous devices and sensors, providing intelligent and connected solutions across various domains.

12.0 Appendix


LIZA
kitchen house


PURINA
felix


LIZA
kitchen house
Welcome!
Press start
PURINA
felix
As GOOD as it LOOKS


GOOD DAY FOR A WASH
PLACE RFID TO START
RFID-RC522