MCTA 3203 SYSTEM INTEGRATION LAB

Final Project: Mini Washer Machine

SECTION : 1
GROUP NUMBER : 7 (G)

LECTURER'S NAME : DR ZULKIFLI BIN ZAINAL ABIDIN

GROUP MEMBERS :

NO :		COURSE	MATRIC
1.	MUHAMMAD NAQIB AIMAN BIN YUSNI	BMCT	2117765
2.	MUHAMAD AZRI BIN ABDUL AZIZ	BMCT	2113537
3.	MUHAMMAD HAFIDZUDDIN HANIF DANIAL BIN NORIZAL	BMCT	2123651
4.	SHAREEN ARAWIE BIN HISHAM	BMCT	2116943

TABLE OF CONTENTS

Abstract

This project aims to create a prototype of a mini washing machine that can mimic and simulate the basic function of a washer machine. We have made a sketch to list down all the hardware that we need to fulfill all the characteristics of an ideal washer machine. The washer machine is built by using a pail as its main body. For the rotating part, we have used a plastic container. For the washer machine system to function as intended, we use microcontroller Arduino and its programming software Arduino IDE to complete the coding part of the project. The mini washing machine is built by using a plastic water pump, tube, DC motor for rotating the washer machine, servo motor for the automated cover that used ultrasonic motion sensor ,button for user input and LCD to display the timer count.

Introduction

In this course MCTA 3203, we were asked to design a mini washing machine but with a twist with our own creativity. The washer machine that we designed has the characteristics that are very user friendly, it can be used by a wide range of people regardless of their age. Furthermore, we try to replicate a real washer machine as accurately as possible by the mode it provides. We have washing mode and even drying mode. In addition we have made sure to minimize the noise level of the washer machine while operating, by this it will avoid disturbing the peace in the house. Additionally, by using a pail we have plenty of space to maximize the laundry capacity, that will be reliable in bigger families.

Our final goal is to successfully integrate all the components that we use to complete the mini washer machine using Arduino microcontroller as our main component. Throughout the designing and building process, we have added several new components like the LCD and ultrasonic sensor to improve the quality and usability of the washer machine. Our final design is very similar to the end result, just with some minor added features.

We began the project by dividing our task to make the process more efficient and smooth. The task is divided by parts like pump, DC motor, servo motor with ultrasonic motion sensor and LCD with the timer. We have coded the system by using one arduino board each to do our part.

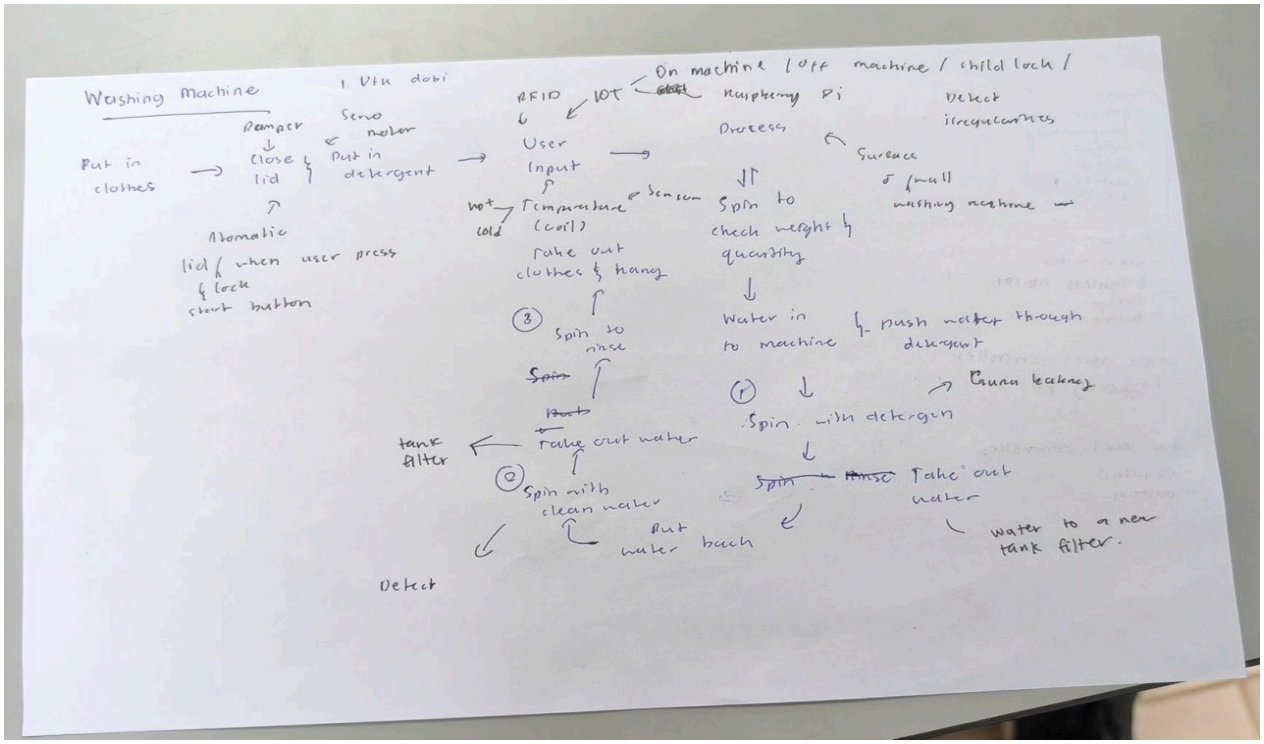


Fig 1: Initial sketch of washing machine design

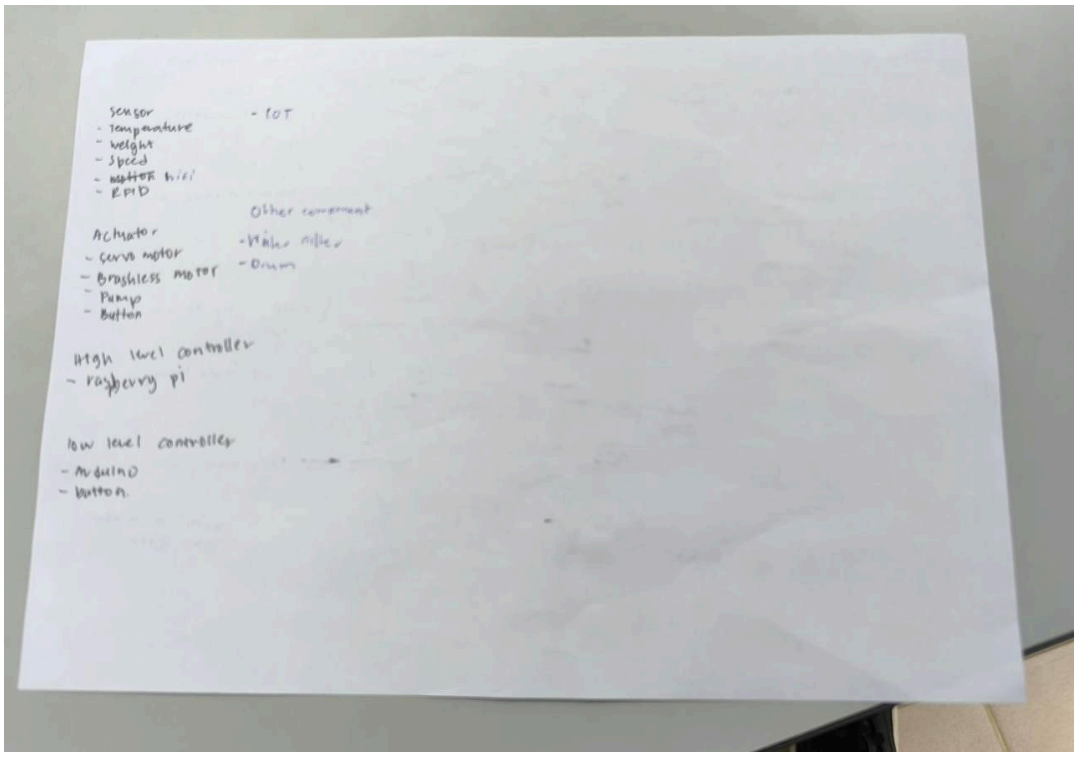


Fig 2: Initial hardware/component list

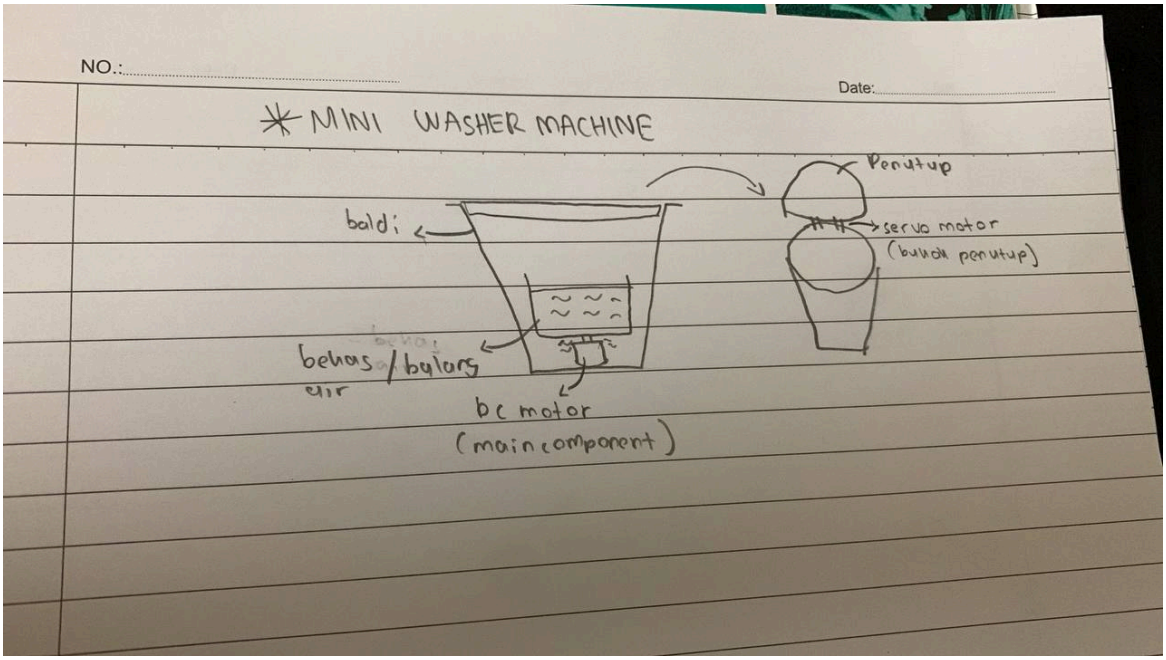


Fig 3: Initial project shape sketch

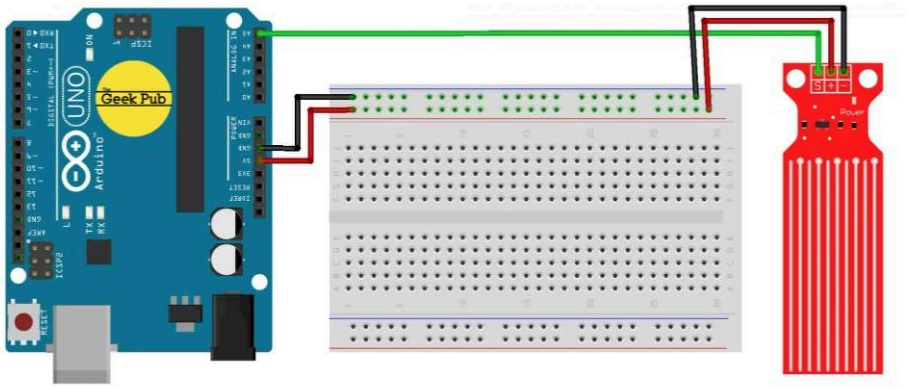
Material And Equipment

- Arduino MEGA
- Arduino UNO
- Jumper Wire
- Push Button
- Breadboard
- Servo motor
- DC Motor
- Water pump
- Pipe
- 12C LCD
- Ultrasonic Motion Sensor
- Water level sensor
- Pail
- Plastic Container
- 9V battery
- LED Lights
- Buzzer

Circuit And Experimental Setup

1. Hardware Setup:

Experimental Setup for water level sensor:



1) Connect the Soil Moisture Sensor to the Breadboard

- Place the soil moisture sensor module on the breadboard.

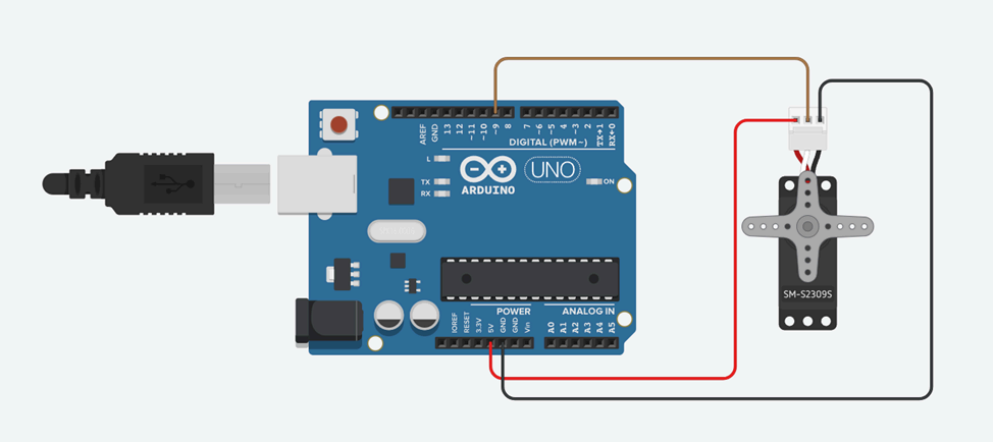
2) Connect the Soil Moisture Sensor to the Arduino

- VCC on the sensor to 5V on the Arduino.
- GND on the sensor to GND on the Arduino.
- A0 (Analog output) on the sensor to A0 (Analog pin 0) on the Arduino.

3) Connect the Breadboard Power Rails

- Connect the positive rail of the breadboard to 5V on the Arduino using a jumper wire.
- Connect the negative rail of the breadboard to GND on the Arduino using a jumper wire.

Experimental Setup for servo motor:



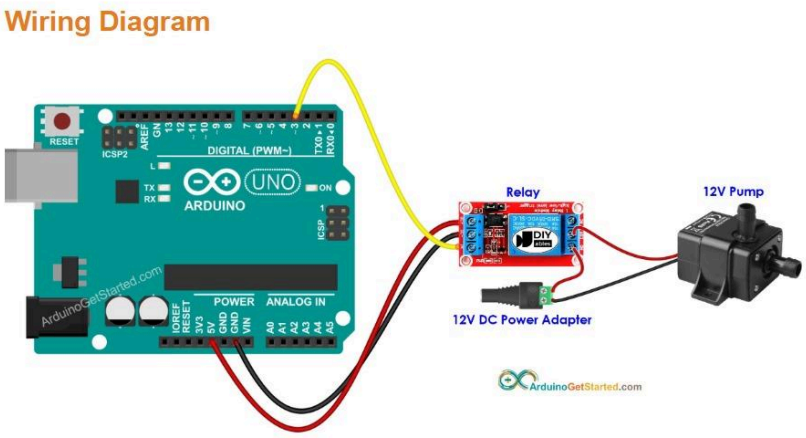
1. Keep the setup from part 1 of the experiment.

2. Connect the Servo's Signal Wire to a digital pin (e.g., D9).

3. Connect the servo's power wire (usually red) to the 5V output on the Arduino board.

4. Connect the servo's ground wire (usually brown) to one of the ground (GND) pins on the Arduino.

Experimental Setup for 5V pump water:



1) Setting Up the Voltage Regulator (LM7805):

- Input (Vin): Connect to the positive terminal of the 12V DC power adapter.
- Ground (GND): Connect to the negative terminal of the 12V DC power adapter.
- Output (Vout): Provides 5V output to power the Arduino and relay.

2) Powering the Arduino:

- Arduino Vin: Connect to the output (Vout) of the LM7805.
- Arduino GND: Connect to the ground (GND) of the LM7805.

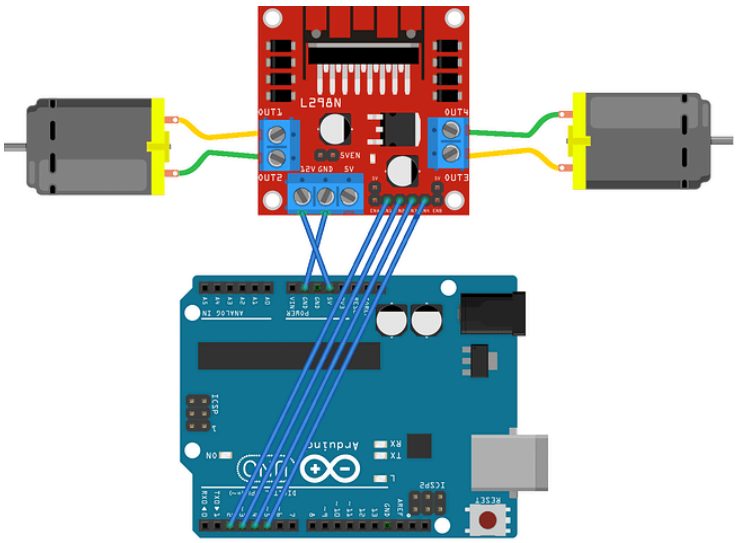
3) Wiring the Relay Module:

- Relay VCC: Connect to the output (Vout) of the LM7805.
- Relay GND: Connect to the ground (GND) of the LM7805.
- Relay IN: Connect to a digital pin on the Arduino (e.g., pin 8).

4) Connecting the Pump to the Relay:

- Relay COM (Common): Connect to one terminal of the pump.
- Relay NO (Normally Open): Connect to the positive terminal of the 12V power supply (directly from the adapter).
- Pump GND: Connect to the GND terminal of the power supply (directly from the adapter).

Experimental Setup for motor:



1) Connect the Motor Driver to the DC Motors

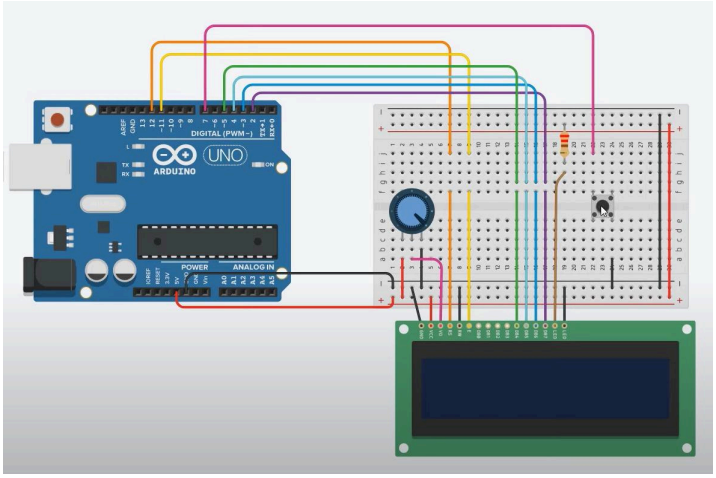
- Motor 1:
    - Connect the yellow wire of Motor 1 to OUT1 on the Motor Driver.
    - Connect the green wire of Motor 1 to OUT2 on theMotor Driver.
- Motor 2:
    - Connect the yellow wire of Motor 2 to OUT3 on the Motor Driver.
    - Connect the green wire of Motor 2 to OUT4 on the Motor Driver.

2) Connect the Motor Driver to the Arduino

3) Connect the Power Supply

- +12V on Motor Driver to the positive terminal of your power supply.
- GND on the Motor Driver to the negative terminal of your power supply.
- 5V on Motor Driver to the 5V pin on the Arduino
- GND on Motor Driver to the GND pin on the Arduino.

Experimental Setup for16x2 LCD Display:



1) Connect the LCD to the Breadboard

- Place the LCD on the breadboard.
- Ensure all the pins of the LCD are accessible for connections.

2) Connect the Potentiometer

- Place the potentiometer on the breadboard.
- Connect one terminal of the potentiometer to 5V on the breadboard.
- Connect the other terminal of the potentiometer to GND on the breadboard.
- Connect the wiper (middle terminal) of the potentiometer to the V0 pin of the LCD.

3) Connect the LCD to the Arduino

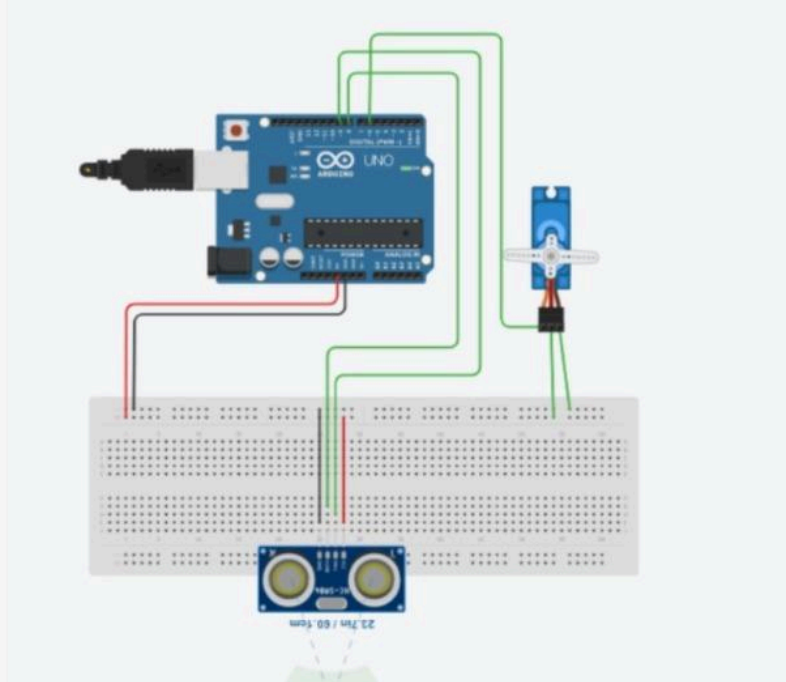
- LCD Pin 1 (VSS) to GND on the Arduino.
- LCD Pin 2 (VDD) to 5V on the Arduino.
- LCD Pin 3 (V0) to the wiper of the potentiometer.
- LCD Pin 4 (RS) to Digital Pin 12 on the Arduino.
- LCD Pin 5 (RW) to GND on the Arduino.
- LCD Pin 6 (E) to Digital Pin 11 on the Arduino.
- LCD Pin 11 (D4) to Digital Pin 5 on the Arduino.
- LCD Pin 12 (D5) to Digital Pin 4 on the Arduino.
- LCD Pin 13 (D6) to Digital Pin 3 on the Arduino.
- LCD Pin 14 (D7) to Digital Pin 2 on the Arduino.
- LCD Pin 15 (A / LED+) to 5V .
- LCD Pin 16 (K / LED-) to GND.

4) Connect the Push Button

- Place the push button on the breadboard.
- Connect one leg of the push button to GND on the breadboard.
- Connect the other leg of the push button to a digital pin (e.g., Digital Pin 7) on the Arduino.

- Connect a 10kΩ resistor from the digital pin (e.g., Digital Pin 7) to 5V to pull the pin high when the button is not pressed.

Experimental Setup for Ultrasonic sensor:



1) Connect the Ultrasonic Sensor:

- VCC: Connect to the 5V pin on the Arduino.
- GND: Connect to the GND pin on the Arduino.
- Trig: Connect to digital pin 9 on the Arduino.
- Echo: Connect to digital pin 10 on the Arduino.

2) Connect the Servo Motor:

- VCC: Connect to the 5V pin on the Arduino.
- GND: Connect to the GND pin on the Arduino.
- Signal: Connect to digital pin 3 on the Arduino.

Coding

Servo motor and Ultrasonic sensor

```
#include <Servo.h>

#define trigPin 9

#define echoPin 8

Servo servo;

void setup() {

  Serial.begin(9600);

  pinMode(trigPin, OUTPUT);

  pinMode(echoPin, INPUT);

  servo.attach(3);

}



void loop() {

  long duration, distance;

  // Send a short pulse to trigger the sensor

  digitalWrite(trigPin, LOW);
```

```
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Read the echo pulse
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance in cm
  distance = (duration / 2) / 29.1;
  // Print the distance to the Serial Monitor
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  // Actuate the servo based on the distance
  if (distance < 10) {
    servo.write(140);
  } else if (distance > 10) {
    servo.write(230);
  } else {
    Serial.println("The distance is more than 180 cm");
  }
```

```
  // Wait for 500 ms before the next loop
  delay(500);
}
```

LCD,LED And Push Button

```
#include <Wire.h>
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2; // input LCD
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
const int speedPlus = 7; // to add speeds
const int Start = 6;     // to execute wash cycle
const int Pause = 9;     // to pause/resume the wash cycle
const int tempPin = 8;   // pin for temperature setting
int speeds = 0;          // wash speed and time per minute
int secs = 59;           // seconds counter
int temp = 1;            // temperature setting
const int mins = 5;      // Fixed wash time
bool LastBtnState = HIGH; // to know the last condition
bool pauseState = HIGH;   // to know the pause condition
void setup() {
```

```
  pinMode(speedPlus, INPUT_PULLUP); // speed add button
  pinMode(Start, INPUT_PULLUP);     // execute button
  pinMode(Pause, INPUT_PULLUP);     // pause/resume button
  pinMode(tempPin, INPUT_PULLUP);   // detect temperature input
  Serial1.begin(9600);
  lcd.begin(16, 2);                 // initialize LCD with 16 columns and 2 rows
  lcd.setCursor(0, 0);
  lcd.print("WASHING MACHINE");
  lcd.setCursor(5, 1);
  lcd.print("TIMER");
  delay(1000);
  lcd.clear();
  lcd.setCursor(5, 0);
  lcd.print("GROUP H");
  lcd.setCursor(0, 1);
  lcd.print("ASSALAMUALAIKUM");
  delay(1000);
  lcd.clear();
}
void loop() {
```

```
// ---------- LCD for wash speed and temperature----------
lcd.setCursor(0, 0);
lcd.print("SPEED ");
lcd.print(speeds);
lcd.setCursor(0, 1);
lcd.print("TEMP: ");
printTemp(temp);
setspeeds();
setTemp();


//-------------- Starting or resetting wash cycle --------------------
bool NewBtnState = digitalRead(Start);
if (NewBtnState == LOW && LastBtnState == HIGH) {
  Serial1.write('1');
  delay(50);
  unsigned long pressTime = millis();
  while (digitalRead(Start) == LOW) {
    if (millis() - pressTime > 2000) { // long press detected
      setup(); // reset the system
      return;
```

```
  }
}
// short press detected
LastBtnState = LOW;
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("REM TIME ");
lcd.print(mins - 1);
delay(1000);
speeds = mins - 1;
while (speeds > 0 && LastBtnState == LOW) {
  bool NewBtnState1 = digitalRead(Start);
  bool NewPauseState = digitalRead(Pause);
  delay(100);
  // Pause/Resume logic
  if (NewPauseState == LOW) {
    pauseState = !pauseState;
    delay(200); // debounce delay
  }
  if (NewBtnState1 == LOW) {
```

```
      LastBtnState = HIGH;
      setup();
    }
    if (pauseState == HIGH) {
      delay(1000);
      secs--;
      if (secs < 0) {
        secs = 59;
        speeds--;
      }
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("REM TIME ");
      lcd.print(speeds);
      lcd.print(":");
      if (secs < 10) lcd.print("0");
      lcd.print(secs);
    }
  }
  LastBtnState = HIGH; // Reset button state
```

```
  }
}


//----------------------------- speed setting ----------------------------
void setspeeds() {
  if (digitalRead(speedPlus) == LOW) {
    delay(200); // debounce delay
    speeds++;
    if (speeds > 3) {
      speeds = 1;
    }
  }
}


//----------------------------- temperature setting ----------------------------
void setTemp() {
  if (digitalRead(tempPin) == LOW) {
    delay(500); // debounce delay
    temp++;
    if (temp > 3) {
```

```
      temp = 1;
    }
  }
}


//----------------------------- print temperature ----------------------------
void printTemp(int temp) {
  switch (temp) {
    case 1:
      lcd.print("Low    ");
      break;
    case 2:
      lcd.print("Medium ");
      break;
    case 3:
      lcd.print("High   ");
      break;
  }
}
```

Motor,Pump And Water Sensor

```
//change it when finalizing

int motor1 = 2;

int motor1rev = 3;

//water sensor

int resval = 0; // holds the value

int respin = A5; // sensor pin used

int speed1 = 200; // Speed for motor 1

int time = 0;

char *inByte = '0';

const int pumpin = 11;

const int pumpout = 12;

void setup() {

  pinMode(motor1, OUTPUT);

  pinMode(pumpin, OUTPUT);

  pinMode(pumpout, OUTPUT);

  pinMode(8, OUTPUT);

  pinMode(9, OUTPUT);

  pinMode(10, OUTPUT);

  Serial1.begin(9600);

  Serial.begin(19200);
```

```
}
void loop() {
  MotorOFF();
  PumpOff();
 if(Serial1.available())
  {
    inByte = Serial1.read();
    if(inByte=='1'){
        digitalWrite(10, HIGH);
      Serial.println("JadiOi");
      delay(1000);
      PumpIn();
      MotorSpin();
      PumpOut();
      PumpIn();
      MotorSpin();
      PumpOut();
      MotorRinse();
      inByte = '0';
    }
  }
```

```
}
void PumpOut(){
  digitalWrite(pumpout, HIGH);
  delay(10000);// add water sensor function
  digitalWrite(pumpout, LOW);
}
void PumpIn(){
  digitalWrite(pumpin, HIGH); // turn on pump 5 seconds
  if (resval > 330) {
  Serial.println("Water Level: High");
 delay(3000);
  digitalWrite(pumpin, LOW);
  }
}
void PumpOff(){
  digitalWrite(pumpin, LOW); // turn on pump 5 seconds
  digitalWrite(pumpout, LOW);
}
void MotorRinse(){
  Motor1CCW();
  delay(100000);
```

```
  if(Serial.available()){
  inByte = Serial.read();
  }
}
void MotorSpin(){
  for(int i=0; i<=60 && inByte == '1'; i++){
    if(Serial.available())
    inByte = Serial.read();
    Motor1CW();
    delay(1000);
  }
}
//motor movement
void Motor1CW(){
//Progression: Masuk air, Spin w detergent, Keluar air, Masuk air, Spin kosong, keluar air, spin/ rinse, siap
  digitalWrite(motor1, HIGH);
  digitalWrite(motor1rev, LOW);
  analogWrite(9, 100);
}
void Motor1CCW(){
  digitalWrite(motor1, LOW);
```

```
  digitalWrite(motor1rev, HIGH);
  analogWrite(9, 100);
}
void MotorOFF(){
  digitalWrite(motor1, LOW);
  digitalWrite(motor1rev, LOW);
}
```

Operation

The washing machine operation begins with the setting up of various components, including the LCD display, water pump, water level sensor, ultrasonic sensor, temperature sensor, motors, and the relay module. The LCD monitor and buttons allow users to choose the motor speed and temperature that they desire for the washing cycle. For further application, these options will be saved as variables.

After the settings have been set up, the water pump turns on via the relay module to start filling the washing machine. The water level sensor continuously monitors the water level, and the pump is turned off once the level that is wanted is reached. Once the water has been added, the washing cycle starts. To ensure that the clothes are properly washed, the motors attached to the shaft and drum revolve in opposite directions. The speed of the motor is controlled based on the user's selection.

After the wash cycle, the machine moves into the rinse cycle. An additional pump works to remove the water inside the machine by sucking out the used water. After draining, the machine is filled again with new water, and the rinse process happens similarly to the wash cycle. When the rinsing is complete, the spinning cycle starts. To quickly spin the drum and remove extra water from the clothes, the motor speed is increased. More efficient drying of the clothes is possible through this spinning.

Additionally, the washing machine has an automated door system. The washing machine door's ultrasonic sensor collects up on any things or objects in front of it. For the purpose to ensure user safety and practicality, a servo motor is set up to automatically open or close the washing machine door based on this detection.

The first Arduino serves as the user interface and manages water level detection. It initializes and controls the 16x2 LCD display, allowing users to interact via push buttons to select motor speed and temperature settings for the washing machine cycle. The Arduino continuously monitors the The first Arduino controls the water level monitoring system and functions as the user interface. It sets up and controls the 16x2 LCD screen, allowing customers to select the washing machine cycle's motor speed and temperatures using push buttons. To make sure the washing machine fills to the suitable level, the Arduino keeps monitoring the water level using a water level sensor. The first Arduino instructs the second Arduino to use a relay module to operate the water pump and turn it off to prevent overflow when the water reaches the level that has been set.

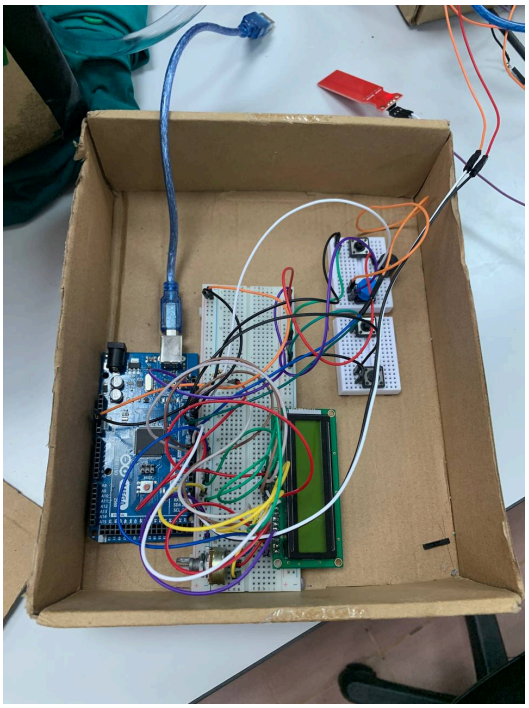
The second Arduino is assigned to the washing machine's operational functions. It uses motor driver modules for controlling the motors that rotate the shaft and drum. During the washing and spinning cycles, the clothes are stirred with the drum motor, and the washing process increases from the shaft motor rotating in the opposite direction. Based on signals regarding the status of the water level received from the first Arduino, this Arduino additionally controls the operation of the water pump.

Result

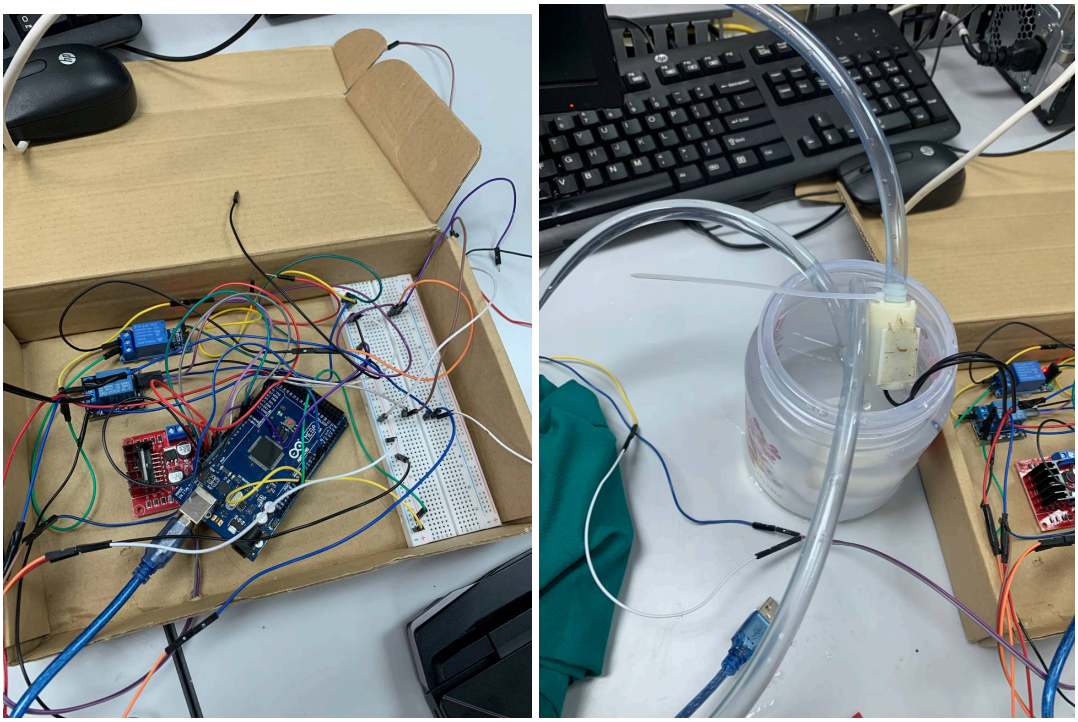
- Our project was successful in achieving the goal and design plans.
- All of the features and system are functioning as intended.
- We successfully presented our Washer Machine system in the lab session.
- Overall we achieved the goal of this mini project by designing and integrating the system to make a mini washing machine.

Picture of the combined Project

a) The Washer Machine main body and wiring for LCD and Button for user input



b)Wiring for Water Pump, Water Sensor and DC Motor



c) Wiring for Servo Motor and Ultrasonic motion sensor

Discussion

The mini washing machine project demonstrated the feasibility of integrating multiple components to create a functional prototype. The use of Arduino microcontrollers facilitated the control and coordination of various parts such as the DC motor, servo motor, water pump, and sensors.

During the testing phase, the washer machine successfully performed the basic washing and drying functions. The use of a pail for the main body for the washing machine, and the plastic container effectively handled the rotation. The integration of an ultrasonic motion sensor to automate the cover and the inclusion of an LCD for displaying the timer count enhanced the user experience, making the device more intuitive and user-friendly.

However, several challenges were encountered. The primary issue was the synchronization between different components, especially in maintaining a consistent operation cycle. There were also occasional glitches in the Arduino code that caused the system to malfunction. Other than that the DC motor and water pump had a hard time in providing the power supply but we encountered it by giving more power by adding additional power supply.

Future improvements could include IOT in the washing machine such as voice recognition, face detection and also connect it with smartphones. Overall, the project provided valuable hands-on experience in system integration and reinforced the importance of thorough testing and iterative development.

Conclusion

In conclusion, the mini washing machine project successfully achieved its objectives by creating a functional, user-friendly prototype using Arduino microcontrollers and various sensors and actuators. The project replicated the core functionalities of a standard washing machine and incorporated additional features like an ultrasonic motion sensor and LCD display, enhancing usability. Despite facing challenges in component synchronization and occasional code glitches, the team's collaborative effort and iterative problem-solving led to a final design that closely matched the initial concept. This project not only provided valuable hands-on experience in system integration but also highlighted areas for future improvement and innovation in small-scale automated washing systems.