Cyber Physical Systems

Basics of Arduino

Sambit Sahoo

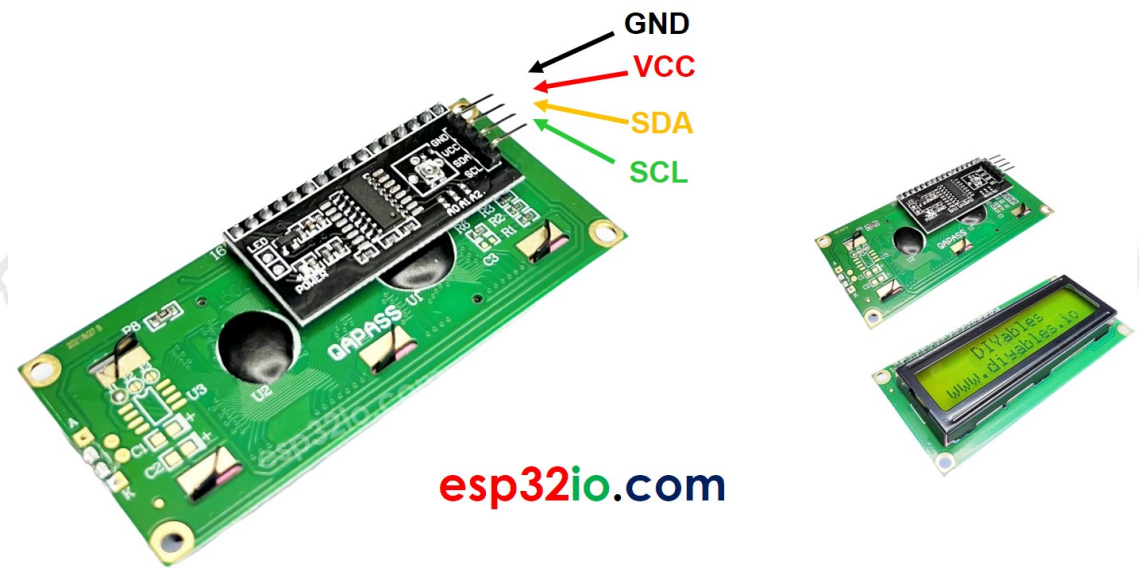
sambitsahoo.k@gmail.com
GitHub Link

May 30, 2025

Table of Contents

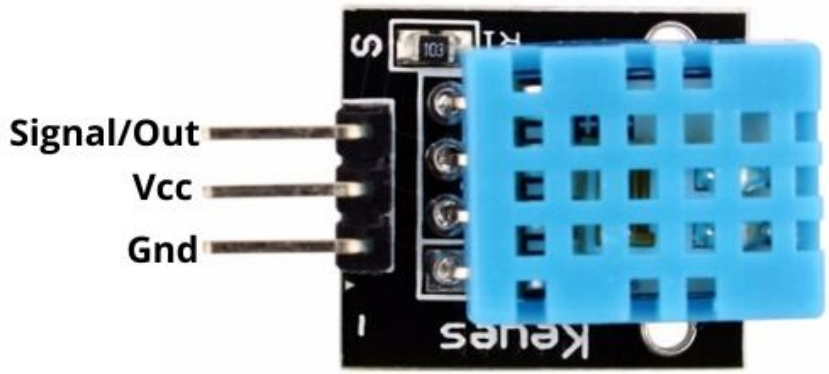
1. DHT11 Sensor Values in LCD

- LCD (Liquid Crystal Display) is a display module commonly used to show alphanumeric characters, symbols, and even custom patterns in embedded projects. The most popular LCD used with Arduino is the 16x2 LCD, which can display 2 lines with 16 characters each. For the experiment, we would be using the LCD with I2C interface.

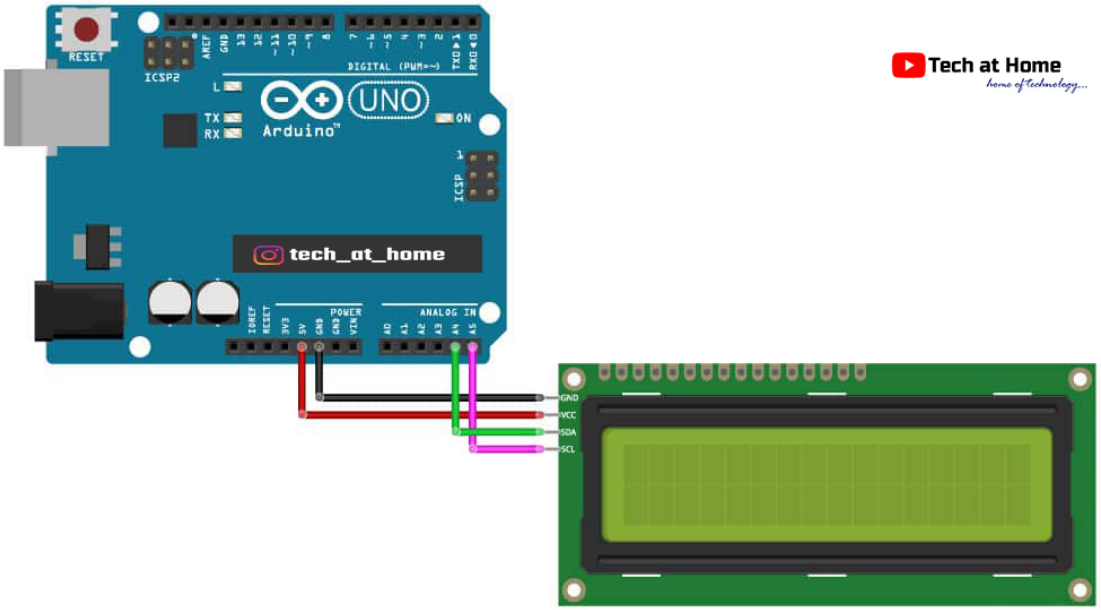


1. DHT11 Sensor Values in LCD

- DHT11 is a basic, low-cost digital sensor used to measure temperature and humidity. It contains a capacitive humidity sensor and a thermistor to measure the surrounding air and provides a digital signal on the data pin.

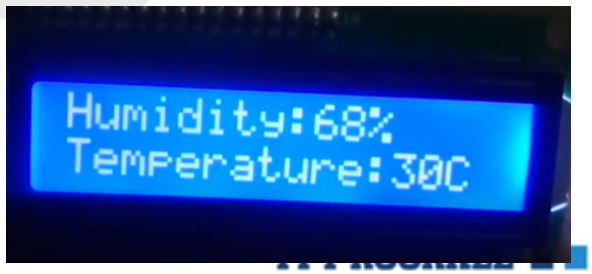


1. DHT11 Sensor Values in LCD




```
DHT dht(DHTPIN, DHTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
  lcd.begin();
  lcd.backlight();
  lcd.setBacklight(HIGH);
}

void loop() {
  int h = dht.readHumidity();
  int t = dht.readTemperature();
  int f = dht.readTemperature(true);
  //check if any reads fail and exit early (to try again)
  if (isnan(h)||isnan(t)||isnan(f)){
    Serial.println("Failed to read");
    return;
  }
  lcd.setCursor(0,0);
  lcd.print("Humidity:");
  lcd.print(h);
  lcd.print("%");
  delay(50);S
  //---------similarly temp
}
```

- For using LCD module we include libraries, "`LiquidCrystal_I2C`" and "`Wire`". The output looks like:



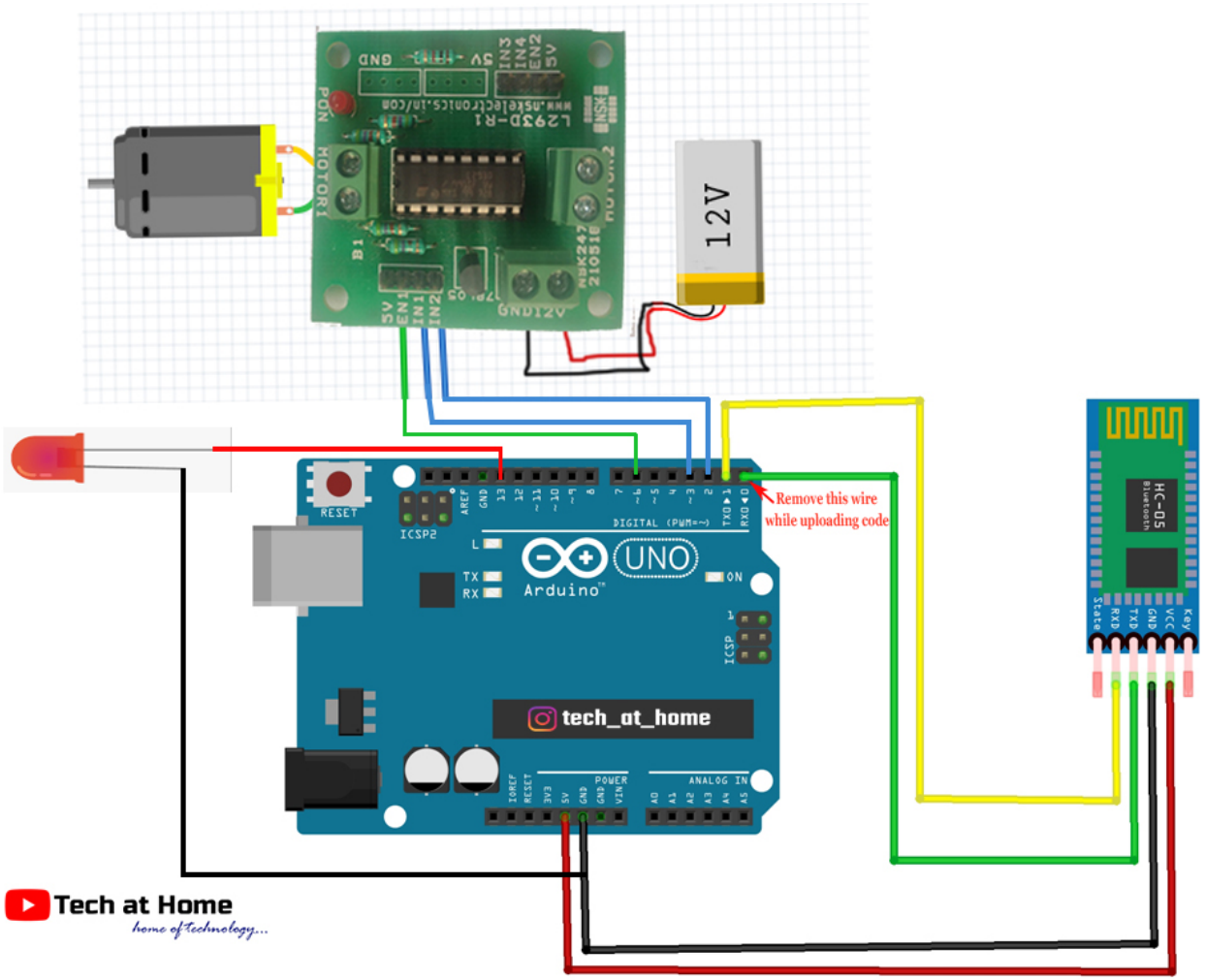
2. Home Automation Project (Fans, Light)

btmodule2_fanregulator_lightswitch.ino

```
1   int in1 = 2;
2   int in2 = 3;
3   int en1 = 6;
4   int led = 13;
5
6   void setup() {
7     Serial.begin(9600);
8     pinMode(led,OUTPUT);
9     pinMode(in1,OUTPUT);
10    pinMode(in2,OUTPUT);
11    pinMode(en1,OUTPUT);
12  }
13
14  void loop() {
15    while (serial.available() > 0){
16      data = serial.read();
17      Serial.println(data);
18    }
19
```

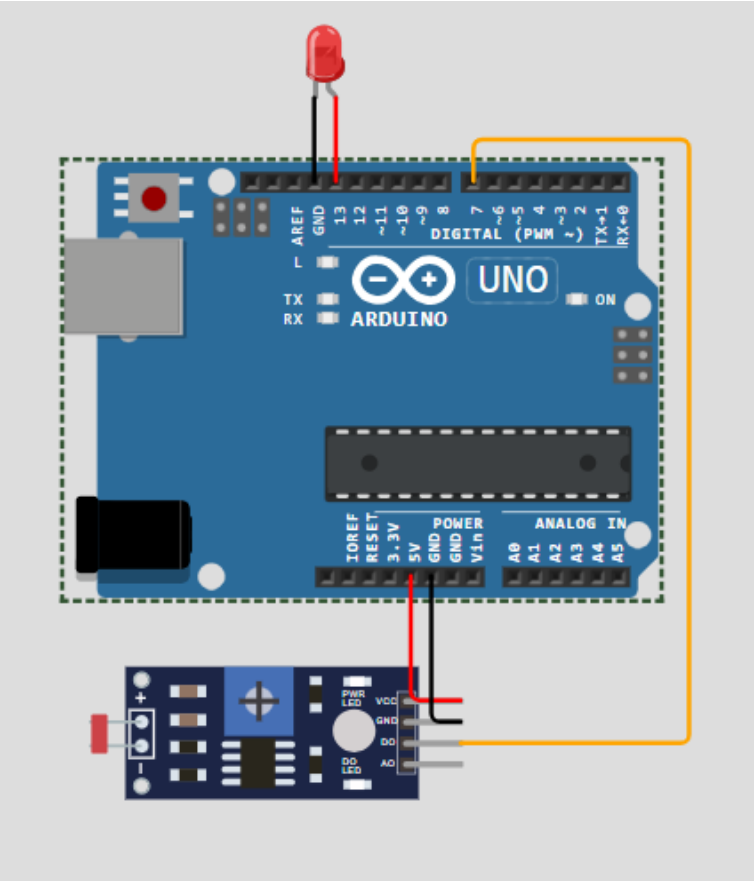
```
20    if (data == "A"){
21      digitalWrite(13,HIGH);
22    }
23
24    if (data == "B"){
25      digitalWrite(13,LOW);
26    }
27
28    if (data == "C"){
29      digitalWrite(in1, HIGH);
30      digitalWrite(in2, LOW);
31      analogWrite(en1, 0);
32    }
33
34    if (data == "D"){
35      digitalWrite(in1, HIGH);
36      digitalWrite(in2, LOW);
37      analogWrite(en1, 100);
38    }
39
40    if (data == "E"){
41      digitalWrite(in1, HIGH);
42      digitalWrite(in2, LOW);
43      analogWrite(en1, 255);
44    }
45  }
```

2. Home Automation Project (Fans, Light)



3. Smart Street Lamps

In this project, we built smart street lamps that during the day (when detecting sunlight) are switched off and switched on during the night.




```
1   //automatic street lamp control using LDR sensors
2   int ldr = 7;
3   int led = 13;
4   int x;
5   void setup() {
6     Serial.begin(9600);
7     pinMode(ldr, INPUT);
8     pinMode(led, OUTPUT);
9   }
10
11  void loop() {
12    x = digitalRead(ldr);
13    Serial.println(x);
14
15    delay(200);
16
17    if (x == 1){
18      digitalWrite(led, LOW);
19    }
20    else{
21      digitalWrite(led, HIGH);
22    }
23  }
24
```

4. Motor Control with Auto and Manual Mode

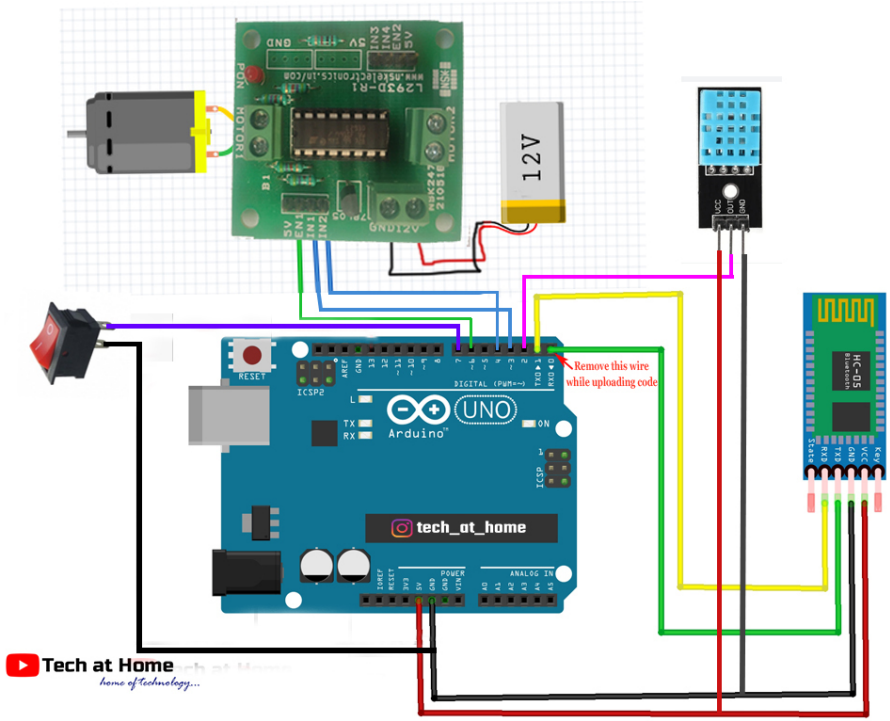
```
//using dht, lcd, btmodule, toggle
#include "DHT.h"
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht (DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd (0x27,16,2);

int in1 = 3;
int in2 = 4;
int en = 6;
int MODE_SW = 7;
int data;
int Mode;

void setup() {
  Serial.begin(9600);
  dht.begin();
  lcd.begin(16,2);
  lcd.backlight();
  lcd.setBacklight(HIGH);

  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(en, OUTPUT);
  pinMode(MODE_SW, INPUT);
}
```

```
void loop() {
  Mode = digitalRead(MODE_SW);

  if (Mode == HIGH){ //Manual mode
    lcd.setCursor(0,0);
    lcd.print("Mode:Manual");
    while (Serial.available()>0){
      data = Serial.read();
      if (data = "C"){
        lcd.setCursor(0,1);
        lcd.print("Fan:OFF");
        delay(10);
        digitalWrite(in1, HIGH);
        digitalWrite(in2, LOW);
        analogWrite(en, 0);
      }
      if (data = "D"){
        lcd.setCursor(0,1);
        lcd.print("Fan:LOW");
        delay(10);
        digitalWrite(in1, HIGH);
        digitalWrite(in2, LOW);
        analogWrite(en, 100);
      }
      if (data = "E"){
        lcd.setCursor(0,1);
        lcd.print("Fan:HIGH");
        delay(10);
        digitalWrite(in1, HIGH);
        digitalWrite(in2, LOW);
        analogWrite(en, 255);
      }
    }
```

```
  if (Mode == LOW){ //Automatic Mode
    lcd.setCursor(0,0);
    lcd.print("Mode:Automatic");
    delay(700);

    float t = dht.readTemperature();
    lcd.setCursor(0,1);
    lcd.print("Temperature:");
    lcd.print(t);
    lcd.print("C");
    delay(20);

    if (t<28){
      digitalWrite(in1, HIGH);
      digitalWrite(in2, LOW);
      analogWrite(en, 0);
    }
    else if(t>=28 && t<32){
      digitalWrite(in1, HIGH);
      digitalWrite(in2, LOW);
      analogWrite(en, 100);
    }
    else if(t>=32){
      digitalWrite(in1, HIGH);
      digitalWrite(in2, LOW);
      analogWrite(en, 255);
    }
  }
}
```

- 2nd image shows manual mode using "BT module" and 3rd image shows automatic mode using "DHT11 sensor".

4. Motor Control with Auto and Manual Mode

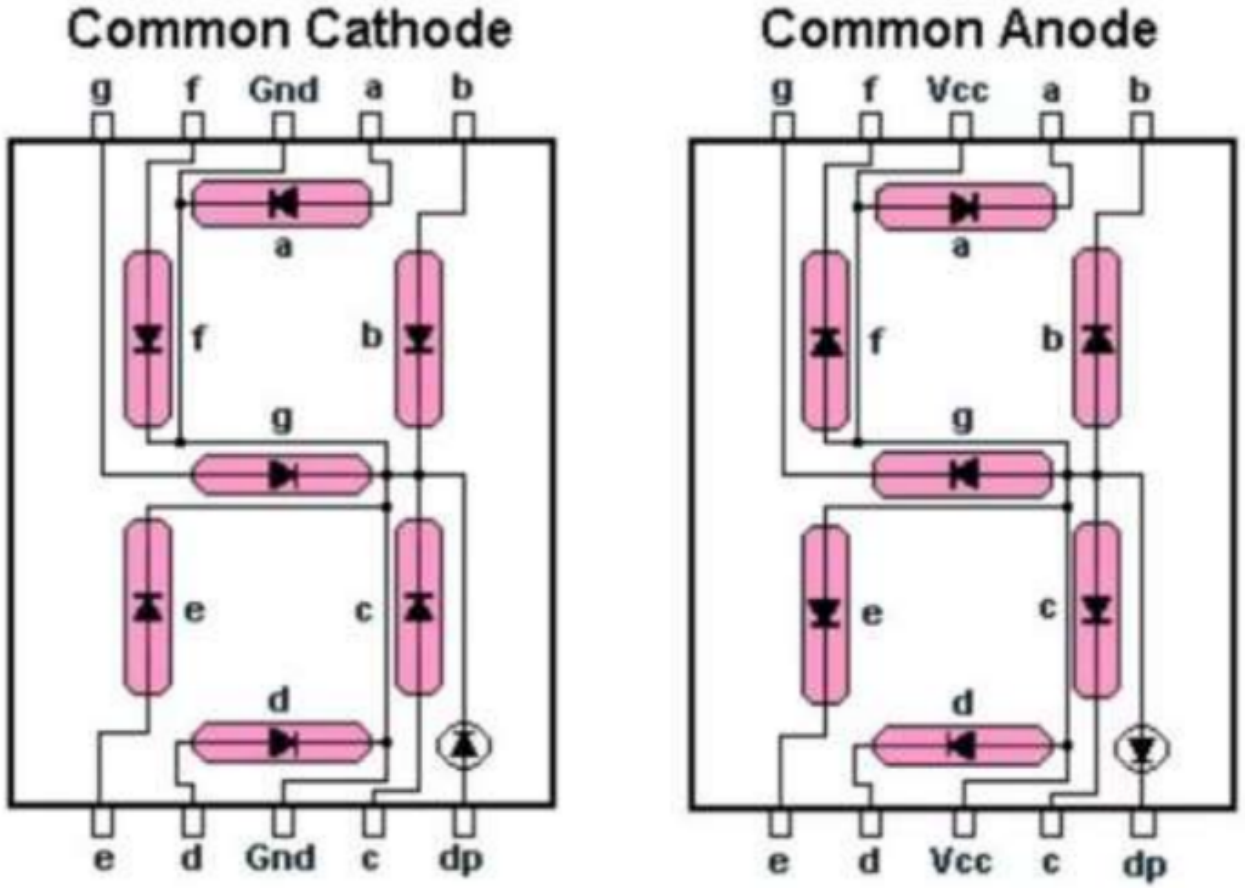


- Toggle switch is used to switch between auto and manual mode.
- The results get displayed using the LCD monitor.
- We can also modify the auto mode to make use of the map function to control the speed of fan according to each temp value by `DHT11`.

5. Traffic Light Control and 7 Segment Display

Seven Segment Display

- Two types: Common Cathode (CC) and Common Anode (CA).
- The difference is the polarity of the LEDs and common terminal.

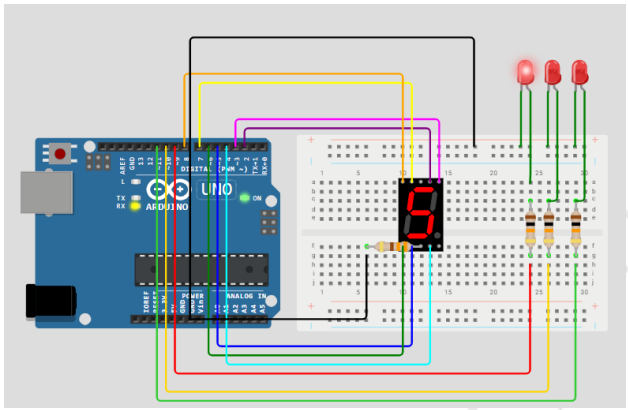


5. Traffic Light Control and 7 Segment Display

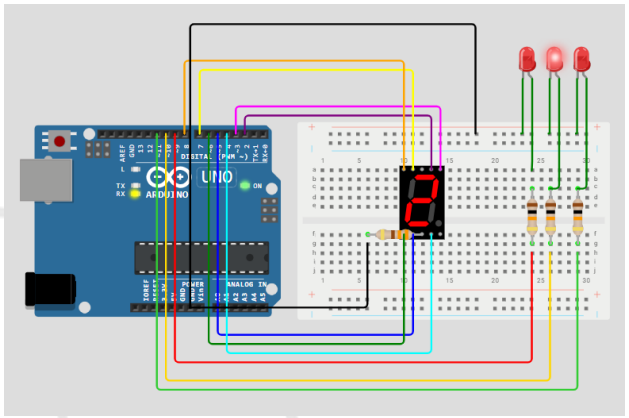
```
#include "SevSeg.h"
SevSeg S;
byte CommonPins[] = {};
byte SegPins[] = {2,3,4,5,6,7,8};
int red =9;
int yellow = 10;
int green = 11;

void setup() {
  S.begin(COMMON_CATHODE, 1, CommonPins, SegPins, 1);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);

  digitalWrite(red, HIGH); //it begins with red on
}

void loop() {
  for(int  i = 9; i>=0; i--){
    display(i); //displays number
    delay(2000);
    if(i <= 3){
      digitalWrite(yellow, HIGH);
      digitalWrite(red, LOW);
    }
  }

  digitalWrite(green , HIGH);
  digitalWrite(yellow, LOW);

  for(int  i = 9; i>=0; i--){
    display(i); //displays number
    delay(2000);
    if(i <= 3){
      digitalWrite(yellow, HIGH);
      digitalWrite(green, LOW);
    }
  }

  digitalWrite(red, HIGH);
  digitalWrite(yellow, LOW);
  digitalWrite(green, LOW);
}

void display(int num) {
  S.setNumber(num);
  S.refreshDisplay();
  delay(20);
}
```

- Here I used the "SevSeg" library to display numbers on 7 segment displays by handling segment control automatically.

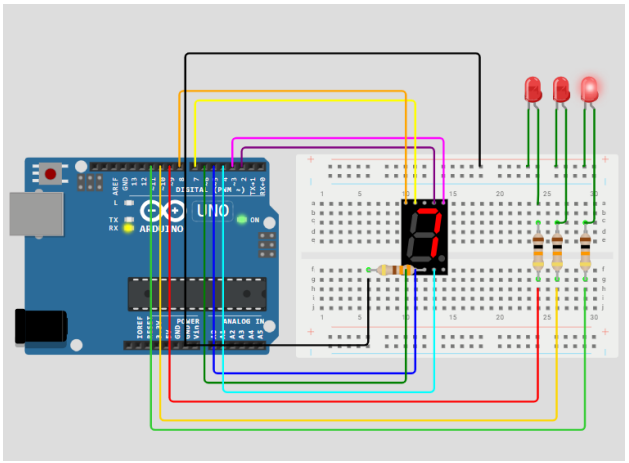
5. Traffic Light Control and 7 Segment Display



Red Light Phase



Yellow Light Phase



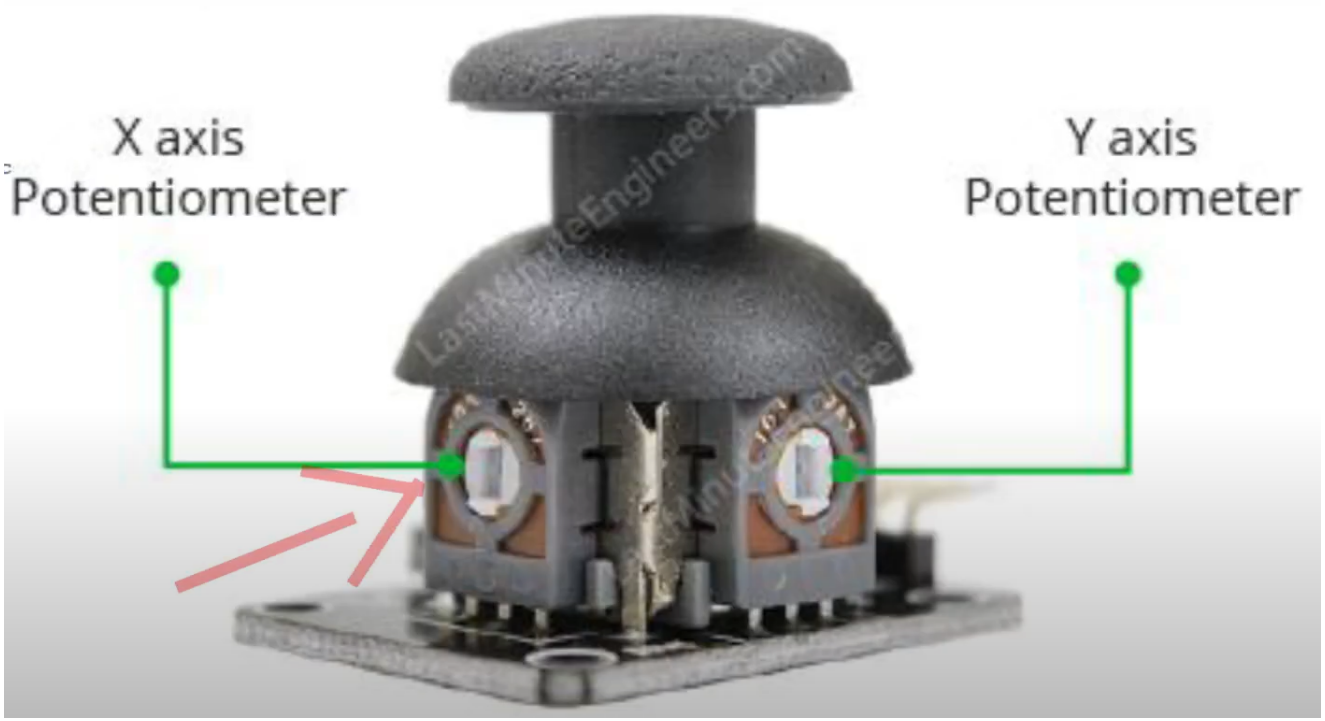
Green Light Phase

- Click here to view the simulation video

6. Controlling lights with Joystick Module

Joystick Module

- Important component in game controllers
- Has two axes, one potentiometer mounted separately for each axes.



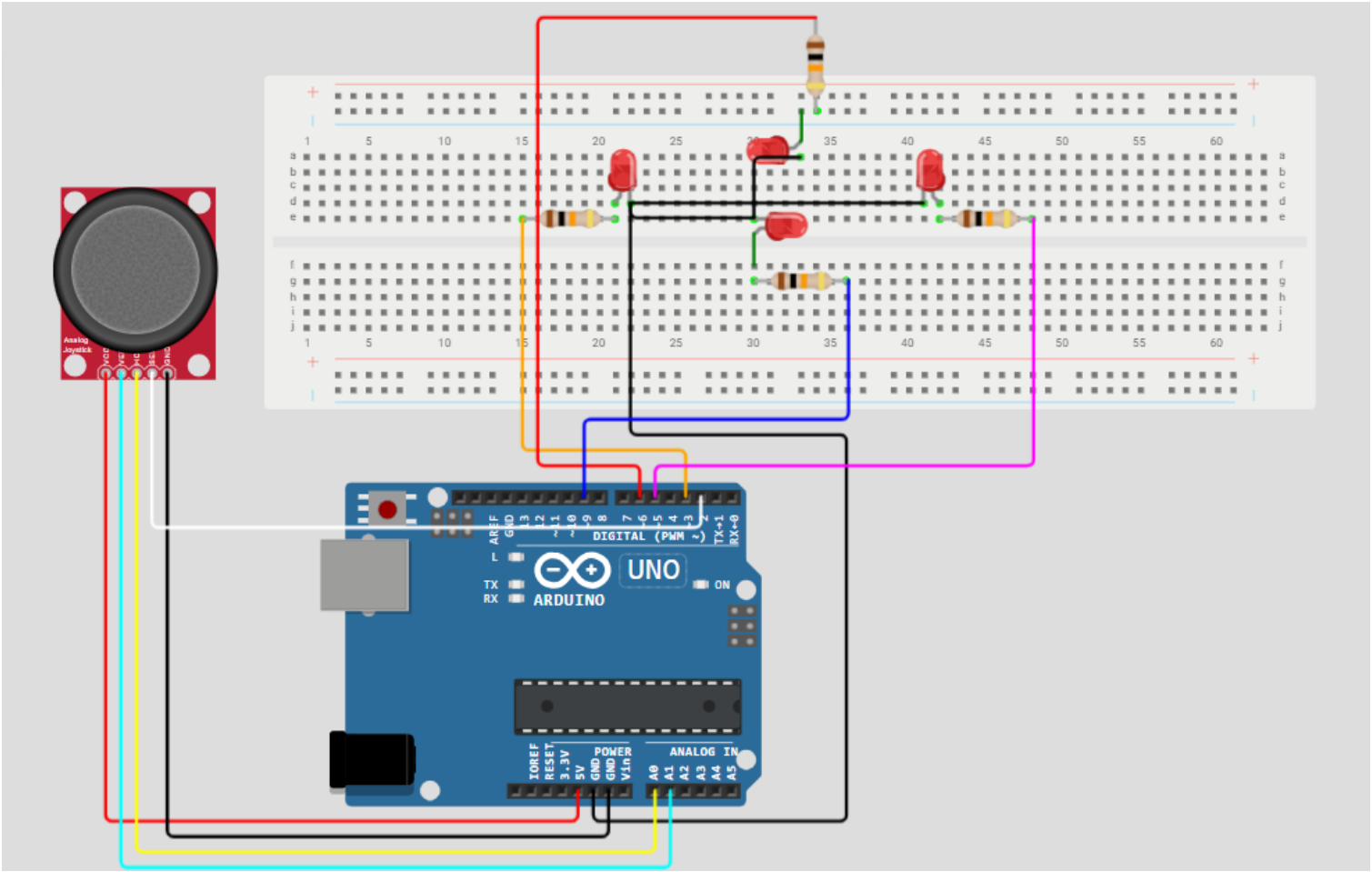
6. Controlling lights with Joystick Module

Joystick_LED.ino

```
int x_pin = A0; //Joystick X axis
int y_pin = A1; //Joystick y axis
int sw_pin = 2;
int led_F = 3;
int led_B = 5;
int led_R = 6;
int led_L = 9;

void setup(){
  Serial.begin(9600);
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(2, INPUT);
  digitalWrite(2, HIGH);
  pinMode(3,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(9,OUTPUT);
}
void loop(){
  int x_data = analogRead(A0);
  int y_data = analogRead(A1);
  int sw_data = digitalRead(2);
  Serial.print("x_data:");
  Serial.print(x_data);
  Serial.print("\t");
  Serial.print("y_data:");
  Serial.print(y_data);
  Serial.print("\t");
  Serial.print("sw_data:");
```

```
if(y_data <= 500){  //Forward LED
  int brightness_F = map(y_data, 500, 0, 0, 255);
  analogWrite(led_F, brightness_F);
}

else if(y_data >= 550){ //Backward LED
  int brightness_B = map(y_data, 550, 1022, 0, 255);
  analogWrite(led_B, brightness_B);
}

else if(x_data >= 550){  //Right LED
  int brightness_R = map(x_data, 550, 1022, 0, 255);
  analogWrite(led_R, brightness_R);
}

else if(x_data <= 500){  //Left LED
  int brightness_L = map(x_data, 500, 0, 0, 255);
  analogWrite(led_L, brightness_L);
}

else{
  digitalWrite(led_F, LOW);
  digitalWrite(led_B, LOW);
  digitalWrite(led_R, LOW);
  digitalWrite(led_L, LOW);
}
```

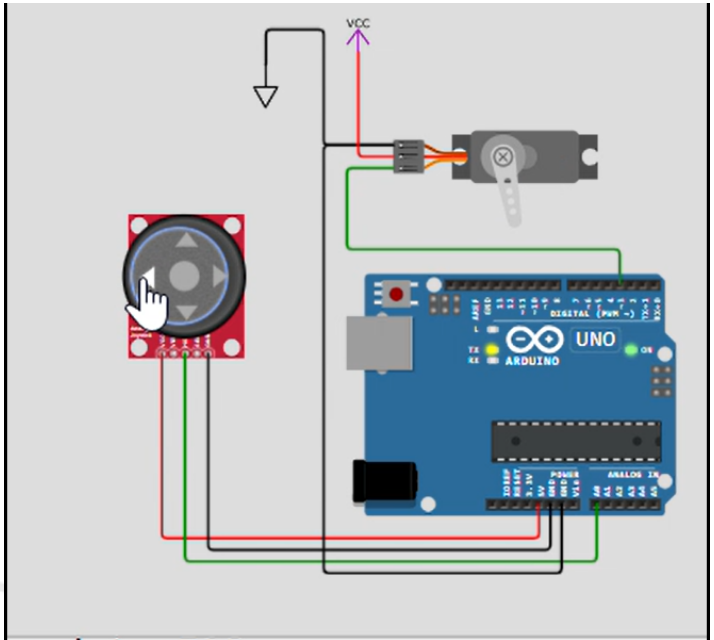
6. Controlling lights with Joystick Module



7. Controlling Servo Motor using Joystick

Controlling_Servomotor_with_Joystick.ino | diagram.json

```
#include<Servo.h>
int x_pin = A0;
Servo s1;
int pos = 90;
void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);
  s1.attach(3);
  s1.write(pos);//start with 90 deg
}

void loop() {
  int x_data = analogRead(A0);
  Serial.print("x_data:");
  Serial.println(x_data);
  if (x_data >= 550){
    if(pos <= 180){
      pos = pos + 10;
      s1.write(pos);
    }
  }
  else if (x_data <= 500){
    if(pos >= 0){
      pos = pos - 10;
      s1.write(pos);
    }
  }
}
```




```
x_data:512
x_data:512
x_data:512
x_data:512
x_data:512
x_data:1023
x_data:1023
```

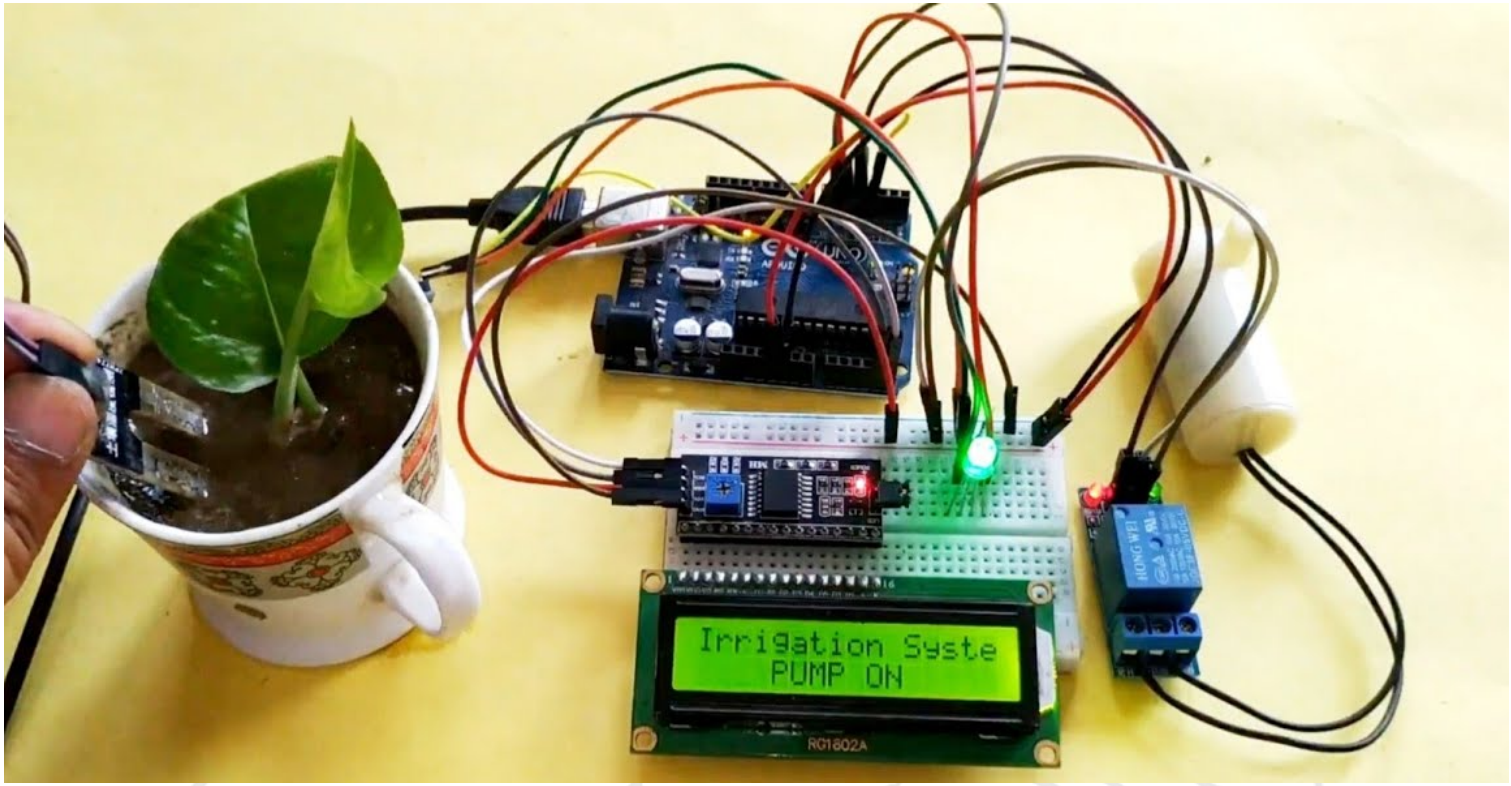
- Simulation Link

8. Smart Irrigation System

Smart_Irrigation_System.ino

```
#include<LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);

int sensor_pin = A0;
int relay_pin = 7;

void setup(){
  Serial.begin(9600);
  lcd.begin(16,2);
  lcd.backlight();
  lcd.setBacklight(HIGH);
  pinMode(sensor_pin, INPUT);
  pinMode(relay_pin, OUTPUT);
}

void loop(){
  int sensor_data = analogRead(sensor_pin);
  Serial.print("Sensor_data:");
  Serial.print(sensor_data);
  Serial.print("\t | ");

  if(sensor_data > 950){
    Serial.println("No moisture, Soil is dry");
    digitalWrite(relay_pin, HIGH);
    lcd.setCursor(0,0);
    lcd.print("Soil Dry");
    lcd.setCursor(0,1);
    lcd.print("Motor ON");
  }
  else if(sensor_data >= 400 && sensor_data <= 950){
    Serial.println("There is some moisture, Soil is medium");
    digitalWrite(relay_pin, LOW);
    lcd.setCursor(0,0);
    lcd.print("Soil Medium");
    lcd.setCursor(0,1);
    lcd.print("Motor OFF");
  }
  else if(sensor_data < 400){
    Serial.println("Soil is wet");
    digitalWrite(relay_pin, LOW);
    lcd.setCursor(0,0);
    lcd.print("Soil Wet");
    lcd.setCursor(0,1);
    lcd.print("Motor OFF");
  }
  delay(100);
}
```

8. Smart Irrigation System



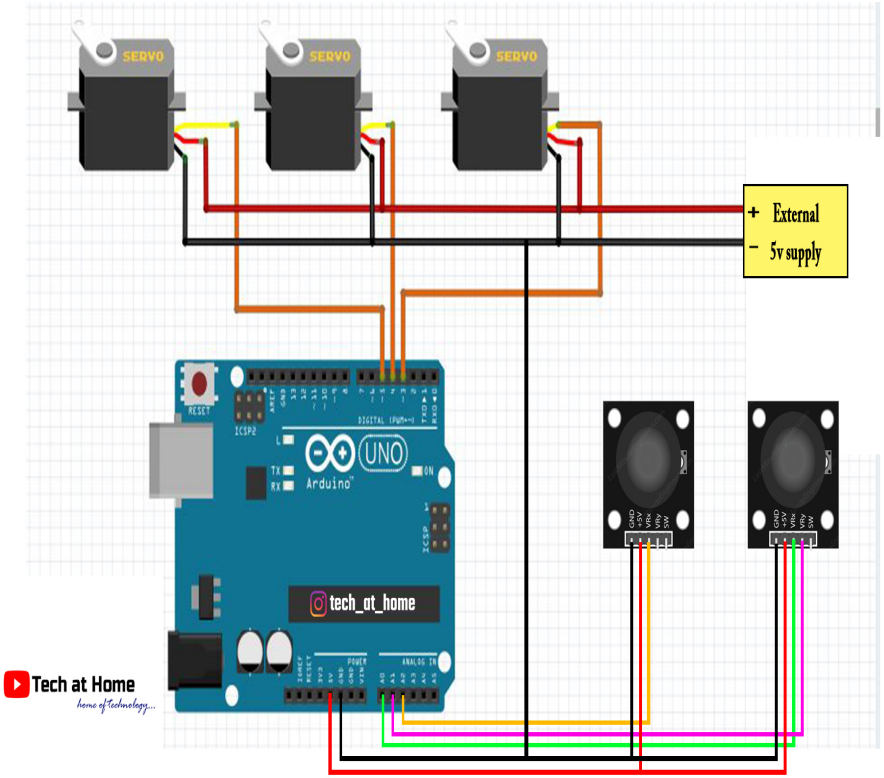
- Soil moisture sensor measures the moisture and waters the crops accordingly which are being shown in lcd.

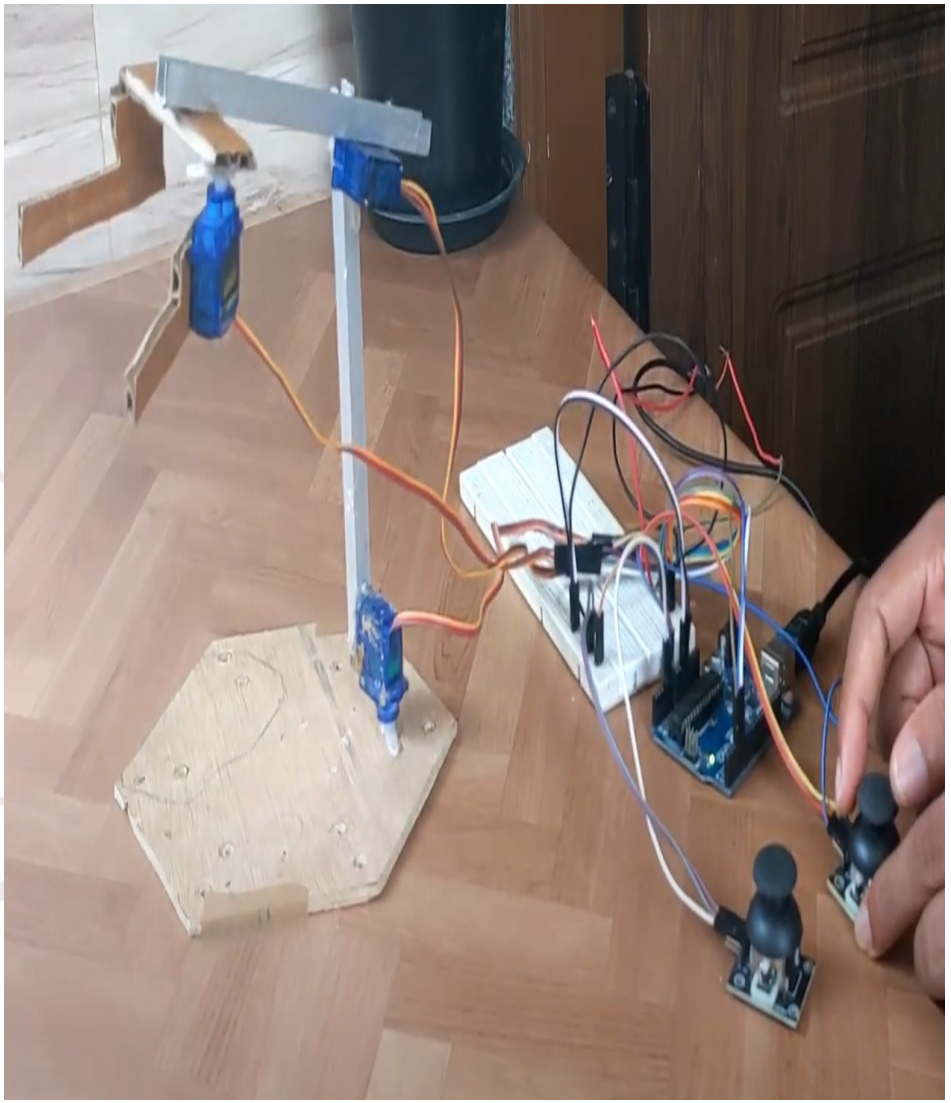
9. Joystick Controlled Robotic Arm

ARM.ino

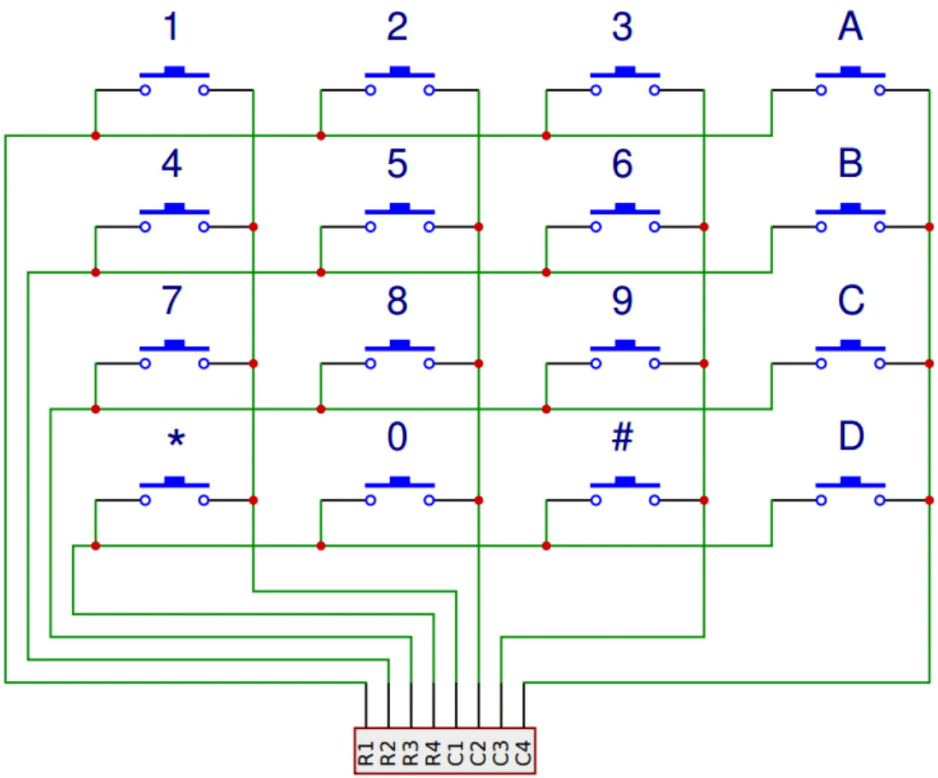
```
#include<Servo.h>
int x_pin1 = A0;  //Joystick-1(X and Y axis)
int y_pin1 = A1;  //Joystick-2(X axis)
int x_pin2 = A2;
int pos1, pos2, pos3 = 0;//Initialize with 0 deg
Servo s1;
Servo s2;
Servo s3;

void setup(){
  Serial.begin(9600);
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  s1.attach(3);    //Servo motor attached to pin 3
  s1.write(pos1);
  s2.attach(4);    //Servo motor attached to pin 4
  s2.write(pos2);
  s3.attach(5);    //Servo motor attached to pin 5
  s3.write(pos3);
}

void loop(){
 //Reading X and Y axis of joystick-1
 int x_data1 = analogRead(A0);
 int y_data1 = analogRead(A1);
 //Reading X axis of joystick-2
 Serial.print("x_data1:");
 Serial.print(x_data1);
 Serial.print("\t");
 Serial.print("y_data1:");
 Serial.print(y_data1);
 Serial.print("\t");
 Serial.print("x_data2:");
 Serial.print(x_data2);
 delay(20);
//Servo 1 control with joystick-1
 if(x_data1 >= 550){
   if(pos1 <= 180){
   pos1 = pos1 + 3;
   s1.write(pos1);
   }
 }
 else if(x_data1 <= 450){
   if(pos1 >= 0){
   pos1 = pos1 - 3;
   s1.write(pos1);
   }
 }
 //Servo 2 control with joystick-1
 if(y_data1 >= 550){
   if(pos2 <= 180){
   pos2 = pos2 + 2;
   s2.write(pos2);
   }
 }
 else if(y_data1 <= 450){
   if(pos2 >= 0){
   pos2 = pos2 - 2;
   s2.write(pos2);
   }
 }


 //Servo 3 control with joystick-2
 if(x_data2 >= 550){
   if(pos3 <= 180){
   pos3 = pos3 + 3;
   s3.write(pos3);
   }
 }
 else if(x_data2 <= 450){
   if(pos3 >= 0){
   pos3 = pos3 - 3;
   s3.write(pos3);
   }
 }
}
```

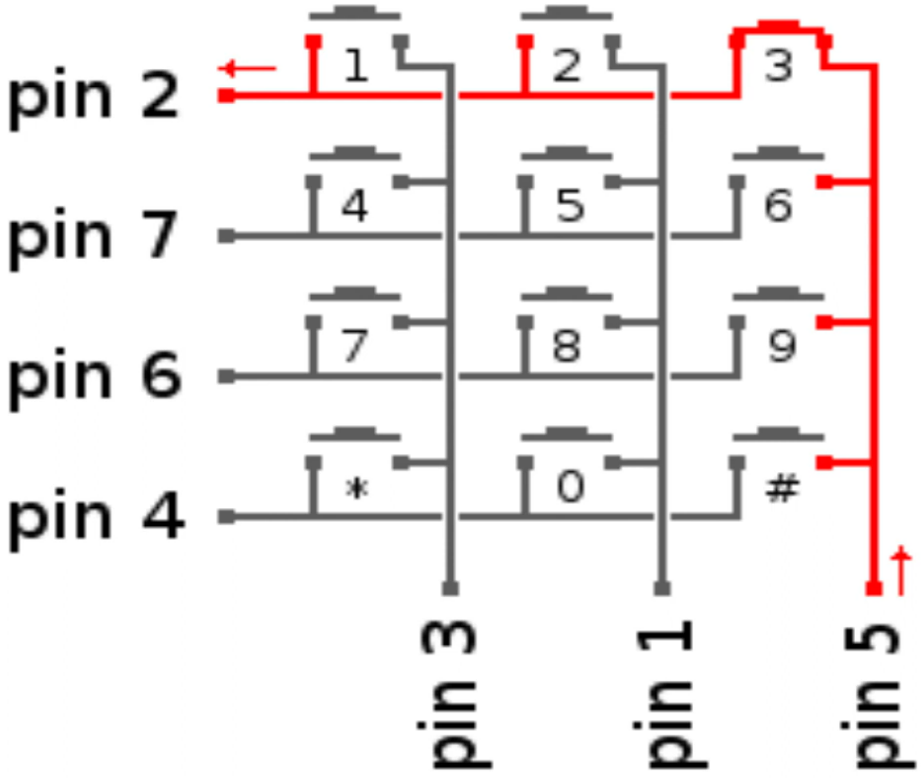
9. Joystick Controlled Robotic Arm





10. 4x4 Keypad Module with LCD



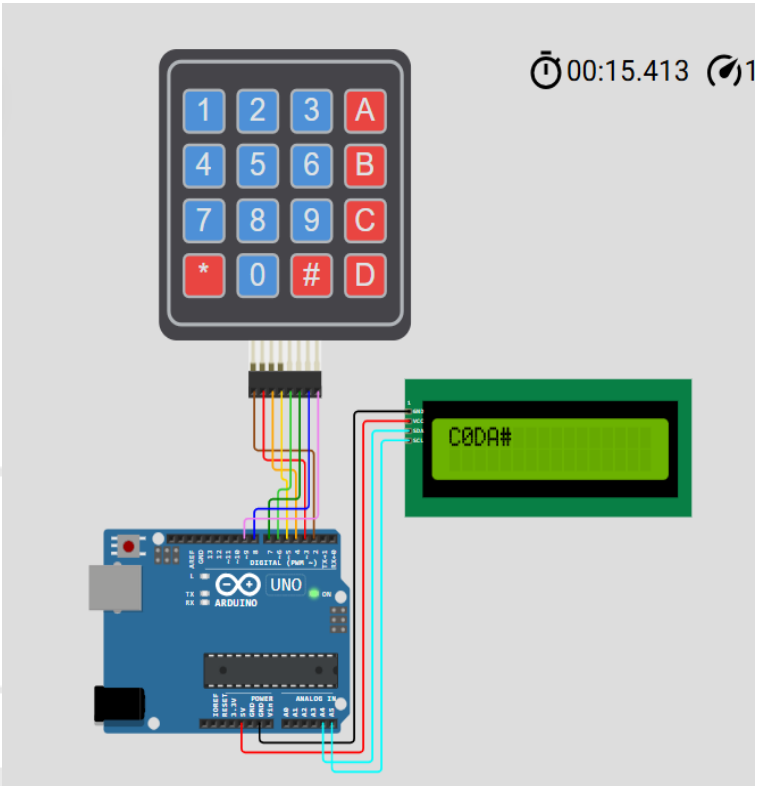


- Internal structure of a keyboard module contains multiple push buttons are connected in a matrix format where pressing one button connects one row and a column.

10. 4x4 Keypad Module with LCD

Arduino_keypad_with_LCD_module.ino | diagram.json

```ino
#include <Keypad.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
LiquidCrystal_I2C lcd(0x27,16,2);
int i;
const char number_of_rows = 4;
const char number_of_columns = 4;

char row_pins[number_of_rows] = {2,3,4,5};
char column_pins[number_of_columns] = {6,7,8,9};
char key_array[number_of_rows][number_of_columns]= {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

Keypad k = Keypad(makeKeymap(key_array),row_pins,column_pins,number_of_rows,number_of_columns);

void setup() {
  Serial.begin(9600);
  lcd.begin(16,2);
  lcd.backlight();
  lcd.setBacklight(HIGH);
}

void loop() {
  char key_pressed = k.getKey();
  if (key_pressed){
    Serial.println(key_pressed);
    lcd.setCursor(i,0);
    lcd.print(key_pressed);
    i = i+1;
  }
}
```



References

- **YouTube:** Arduino Projects
- **Book:** Simon Monk, "30 Arduino Projects for the Evil Genius"
- **Simulations:** Wokwi,Tinkercad

Thanks.