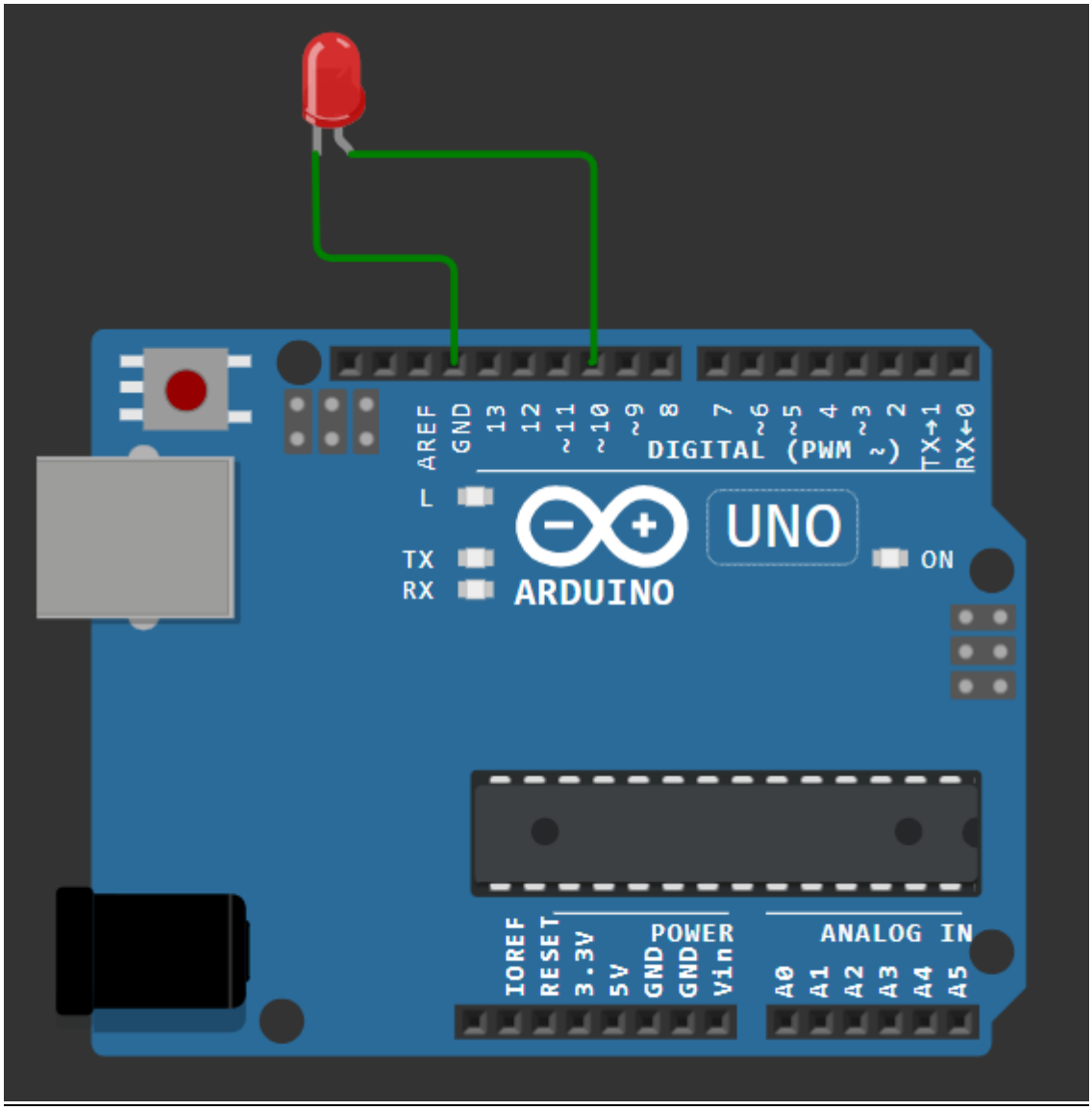
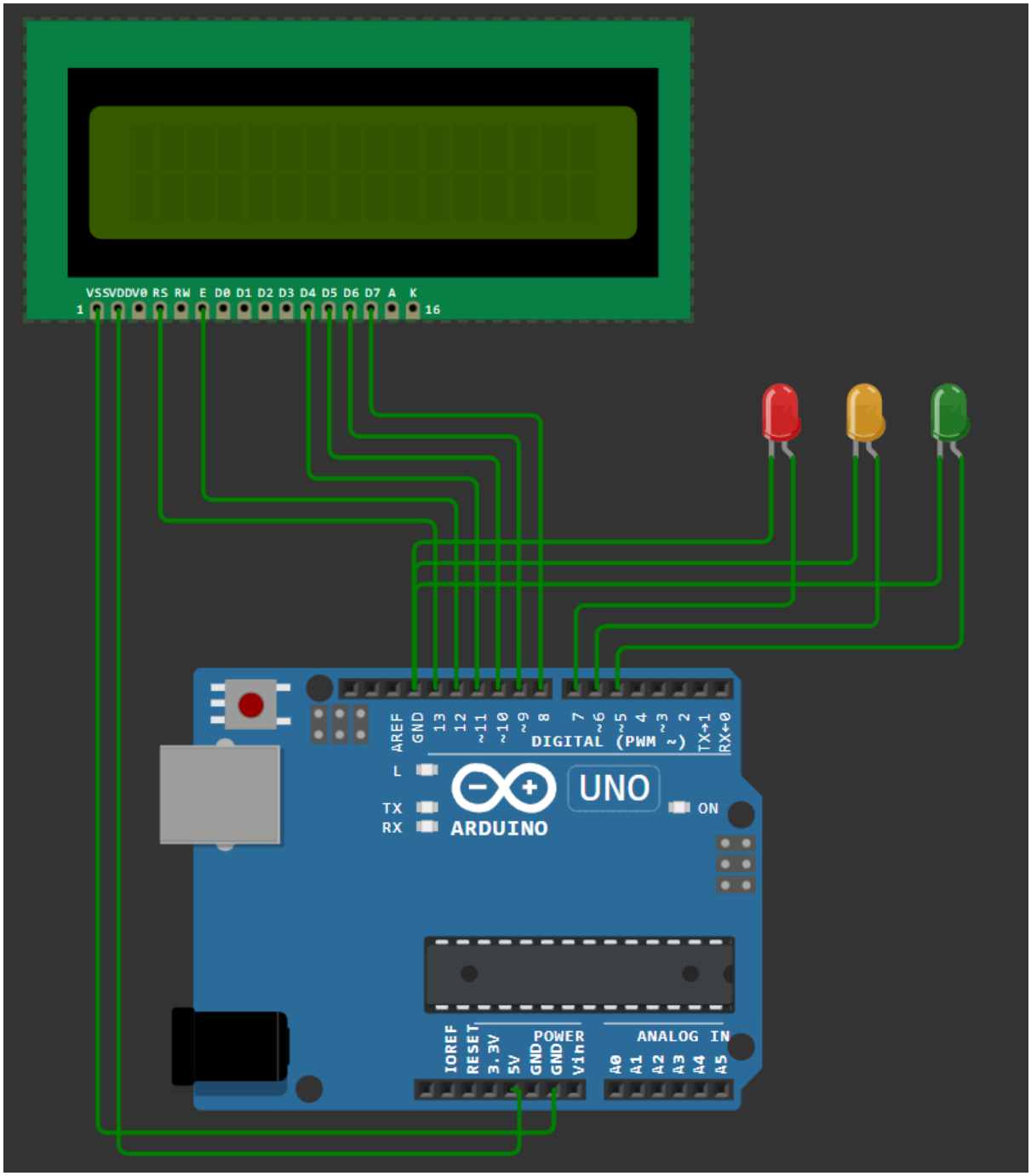
(1) Blinking LED:




```
const int ledPin=10;
void setup() {
  // put your setup code here, to run once:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

(2.1) Traffic Light (LCD Normal):

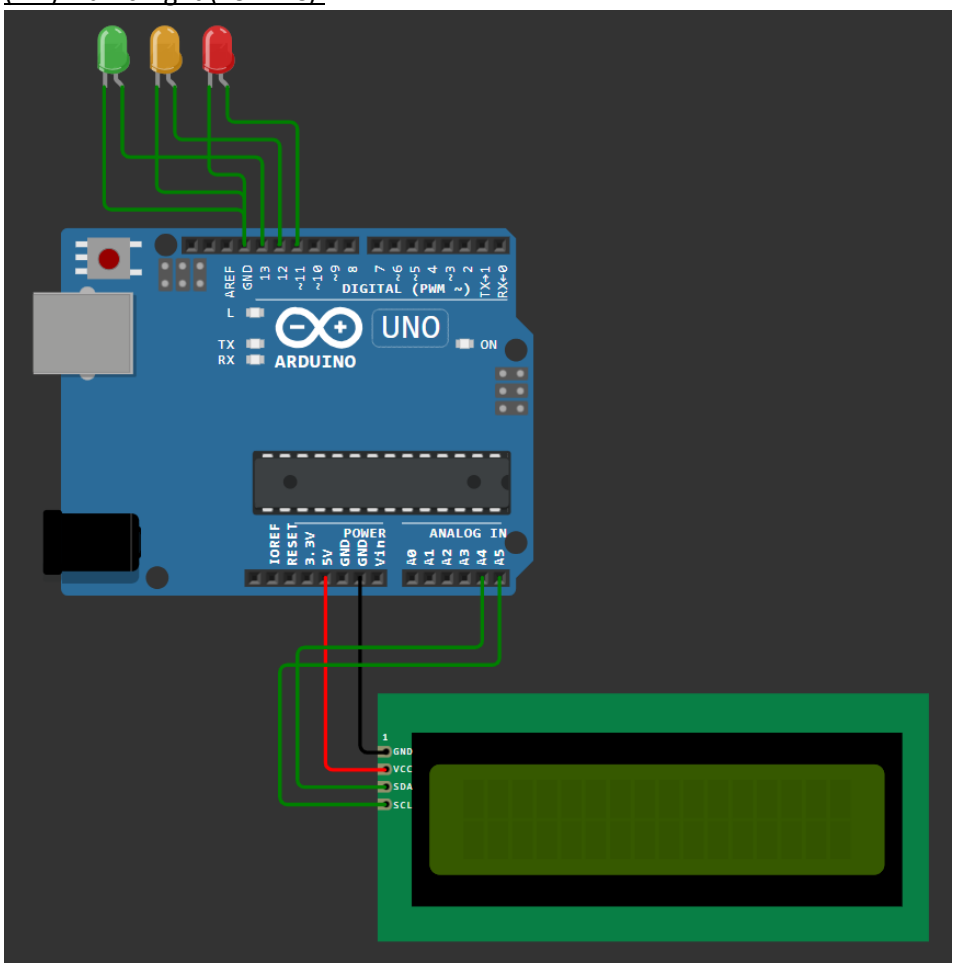



```
#include <LiquidCrystal.h>
LiquidCrystal lcd(13,12,11,10,9,8);
const int green = 5;
const int orange = 6;
const int red = 7;
void setup() {
  // put your setup code here, to run once:
  lcd.begin(16, 2);
  pinMode(green, OUTPUT);
  pinMode(orange, OUTPUT);
  pinMode(red, OUTPUT);
```

```
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(green,HIGH);
  lcd.setCursor(7,0);
  lcd.print("GO");
  delay(3000);
  digitalWrite(green, LOW);
  lcd.clear();

  digitalWrite(orange, HIGH);
  lcd.setCursor(4,0);
  lcd.print("READY TO");
  lcd.setCursor(4,1);
  lcd.print("STOP");
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(300);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(300);
  digitalWrite(orange, LOW);
  lcd.clear();

  digitalWrite(red, HIGH);
  lcd.setCursor(4,0);
  lcd.print("STOP");
  delay(7000);
  digitalWrite(red, LOW);
  lcd.clear();

  digitalWrite(orange, HIGH);
  lcd.setCursor(4,0);
  lcd.print("READY TO");
  lcd.setCursor(4,1);
```

```
  lcd.print("START");
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(300);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(300);
  digitalWrite(orange, LOW);
  lcd.clear();
}
```

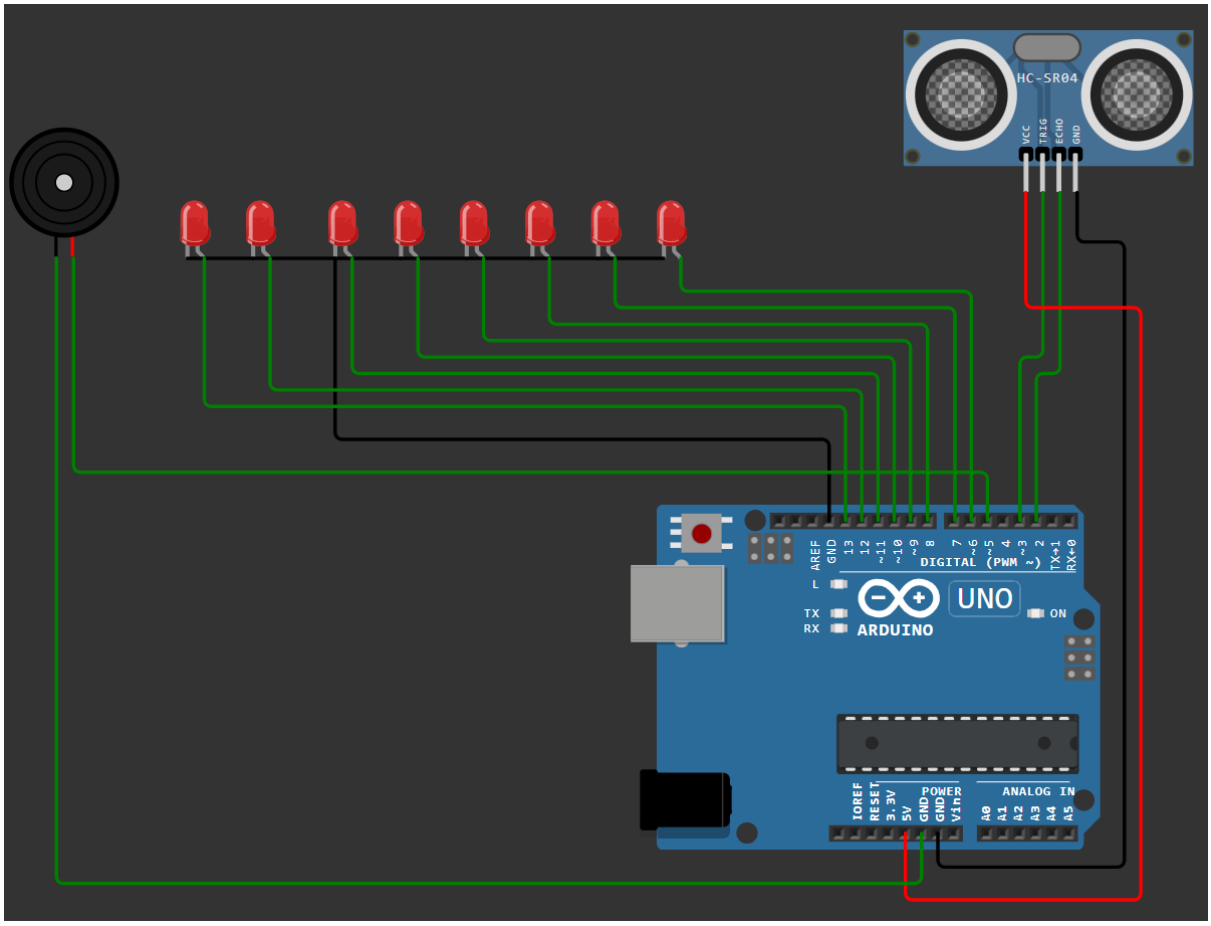
(2.2) Traffic Light (LCD I2C):



```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
const int green = 13;
const int orange = 12;
const int red = 11;
void setup() {
  // put your setup code here, to run once:
  lcd.init();
  lcd.backlight();
  pinMode(green, OUTPUT);
  pinMode(orange, OUTPUT);
  pinMode(red, OUTPUT);

}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(green,HIGH);
  lcd.setCursor(7,0);
  lcd.print("GO");
  delay(3000);
  digitalWrite(green, LOW);
  lcd.clear();

  digitalWrite(orange, HIGH);
  lcd.setCursor(4,0);
  lcd.print("READY TO");
  lcd.setCursor(4,1);
  lcd.print("STOP");
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(100);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(300);
  digitalWrite(orange, LOW);
  delay(100);
  digitalWrite(orange, HIGH);
  delay(300);
  digitalWrite(orange, LOW);
  lcd.clear();
```

```
 digitalWrite(red, HIGH);
 lcd.setCursor(4,0);
 lcd.print("STOP");
 delay(7000);
 digitalWrite(red, LOW);
 lcd.clear();

 digitalWrite(orange, HIGH);
 lcd.setCursor(4,0);
 lcd.print("READY TO");
 lcd.setCursor(4,1);
 lcd.print("START");
 delay(100);
 digitalWrite(orange, LOW);
 delay(100);
 digitalWrite(orange, HIGH);
 delay(100);
 digitalWrite(orange, LOW);
 delay(100);
 digitalWrite(orange, HIGH);
 delay(100);
 digitalWrite(orange, LOW);
 delay(100);
 digitalWrite(orange, HIGH);
 delay(300);
 digitalWrite(orange, LOW);
 delay(100);
 digitalWrite(orange, HIGH);
 delay(300);
 digitalWrite(orange, LOW);
 lcd.clear();
}
```

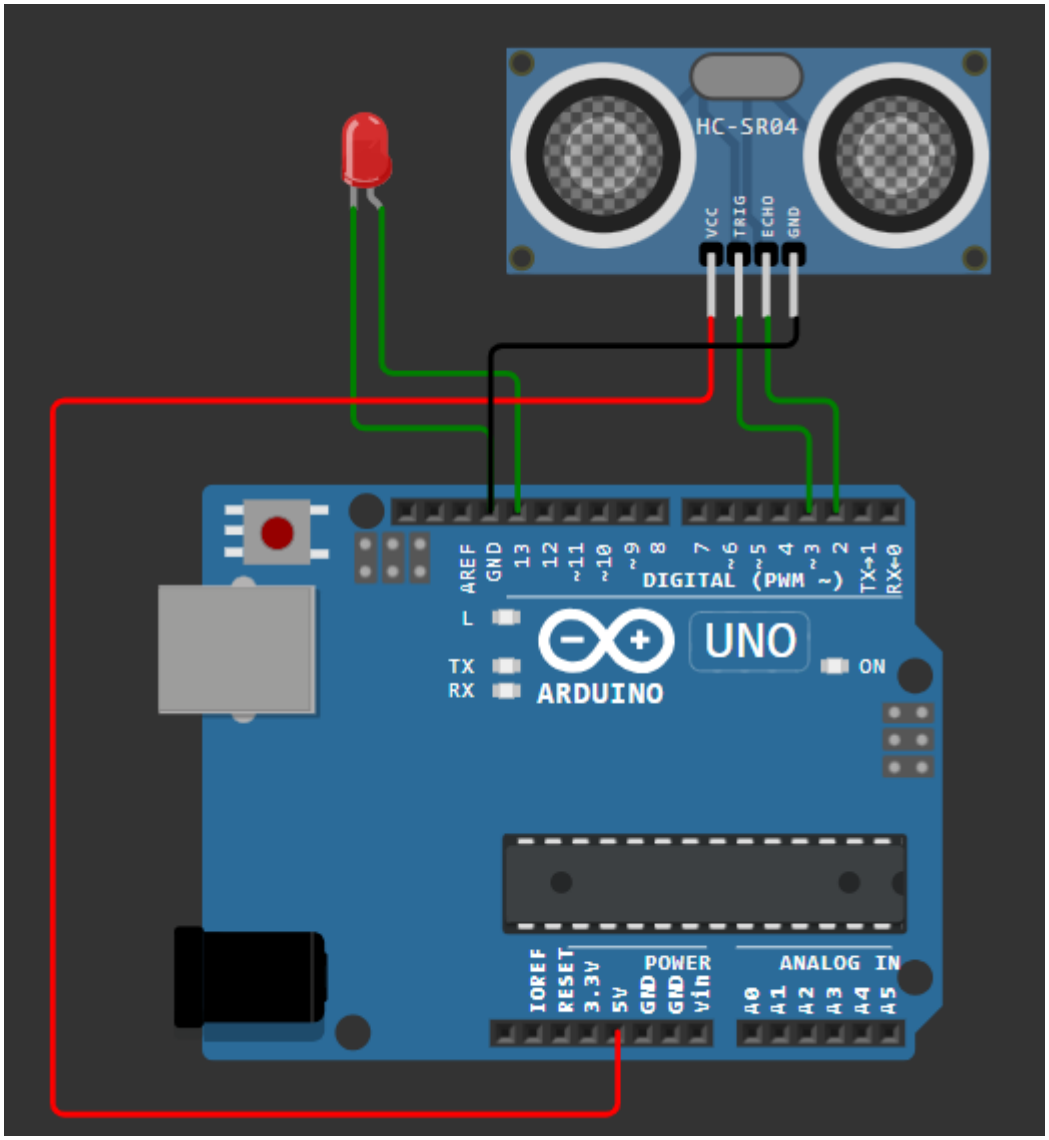
(3.1) Ultrasonic Sensor (8 LED's):




```
#define ECHO_PIN 2
#define TRIG_PIN 3
int LED_PIN1= 6;
int LED_PIN2= 7;
int LED_PIN3= 8;
int LED_PIN4= 9;
int LED_PIN5= 10;
int LED_PIN6= 11;
int LED_PIN7= 12;
int LED_PIN8= 13;
float buzzer= 5;
void setup() {
  Serial.begin(115200);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(LED_PIN1, OUTPUT);
  pinMode(LED_PIN2, OUTPUT);
  pinMode(LED_PIN3, OUTPUT);
  pinMode(LED_PIN4, OUTPUT);
  pinMode(LED_PIN5, OUTPUT);
  pinMode(LED_PIN6, OUTPUT);
```

```
  pinMode(LED_PIN7, OUTPUT);
  pinMode(LED_PIN8, OUTPUT);
  pinMode(buzzer, OUTPUT);
}

int readDistanceCM() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  int duration = pulseIn(ECHO_PIN, HIGH);
  return round(duration * 0.0342 / 2);
}

void loop() {
  digitalWrite(buzzer, LOW);
  int distance = readDistanceCM();
  Serial.print("Measured distance: ");

  boolean d1= distance>0;
  boolean d2= distance>50;
  boolean d3= distance>100;
  boolean d4= distance>150;
  boolean d5= distance>200;
  boolean d6= distance>250;
  boolean d7=distance>300;
  boolean d8=distance>350 && distance<=400;
  boolean buzz =  distance<50;
  Serial.println(distance);
  digitalWrite(buzzer, buzz);
  digitalWrite(LED_PIN1, d1);
  digitalWrite(LED_PIN2, d2);
  digitalWrite(LED_PIN3, d3);
  digitalWrite(LED_PIN4, d4);
  digitalWrite(LED_PIN5, d5);
  digitalWrite(LED_PIN6, d6);
  digitalWrite(LED_PIN7,d7);
  digitalWrite(LED_PIN8,d8);
  delay(100);
}
```

(3.2) Ultrasonic Sensor:

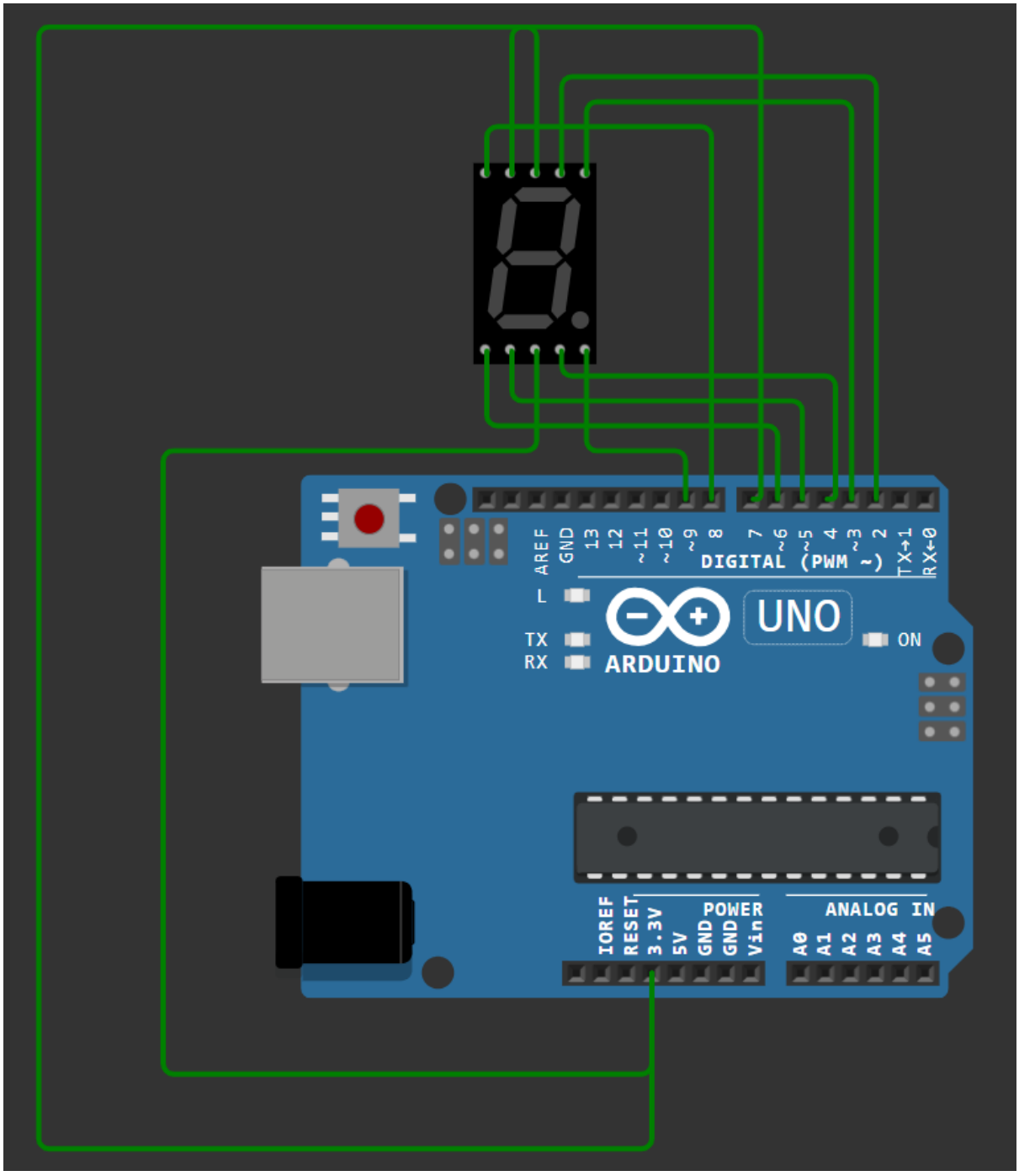



```
#define ECHO_PIN 2
#define TRIG_PIN 3

void setup() {
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
}

float readDistanceCM() {
```

```
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  int duration = pulseIn(ECHO_PIN, HIGH);  // Fixed typo: pulseIn to pulseIn
  return duration * 0.034 / 2;
}

void loop() {
  float distance = readDistanceCM();
  bool isNearby = distance < 100;
  digitalWrite(LED_BUILTIN, isNearby);
  Serial.print("Measured distance: ");
  Serial.println(distance);  // Changed to print 'distance' variable
  delay(100);
}
```

(4.1) Continuous Increment:

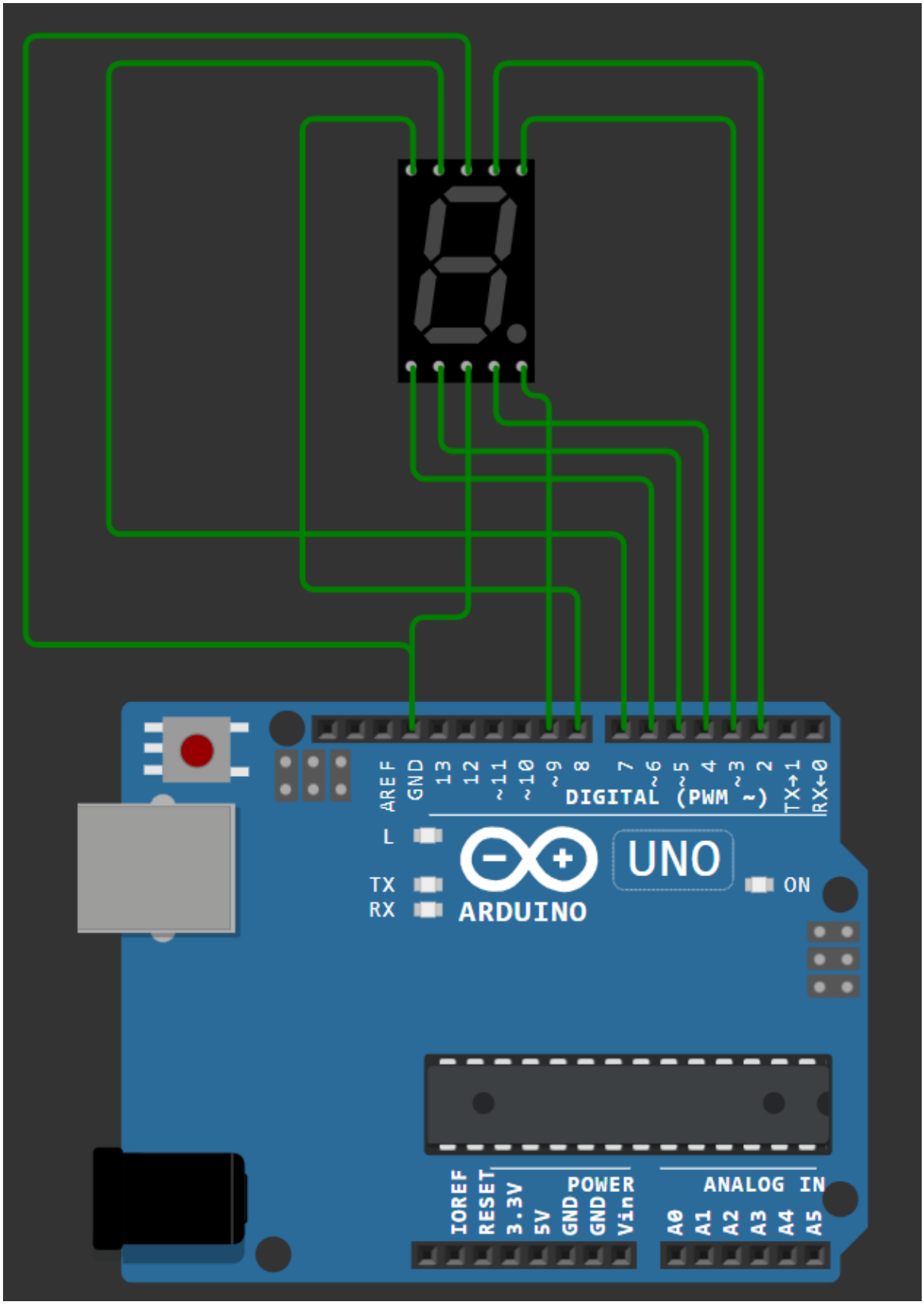
(i) Anode




```
int a = 2;
int b = 3;
int c = 4;
int d = 5;
int e = 6;
int f = 7;
int g = 8;
int point = 9;
void setup()
{
```

```
  pinMode(a, OUTPUT); //a
  pinMode(b, OUTPUT); //b
  pinMode(c, OUTPUT); //c
  pinMode(d, OUTPUT); //d
  pinMode(e, OUTPUT); //e
  pinMode(f, OUTPUT); //f
  pinMode(g, OUTPUT); //g
  pinMode(point, OUTPUT); //point
}
void showNumber(int number)
{
  if(number != 1 && number != 4)
    digitalWrite(a,LOW);
  if(number != 5 && number != 6)
    digitalWrite(b,LOW);
  if(number != 2)
    digitalWrite(c,LOW);
  if(number != 1 && number != 4 && number !=7)
    digitalWrite(d,LOW);
  if(number == 2 || number == 6 || number == 8 || number == 0)
    digitalWrite(e,LOW);
  if(number != 1 && number != 2 && number != 3 && number != 7)
    digitalWrite(f,LOW);
  if (number != 0 && number != 1 && number != 7)
    digitalWrite(g,LOW);
}
void turnOff()
{
  digitalWrite(a,HIGH);
  digitalWrite(b,HIGH);
  digitalWrite(c,HIGH);
  digitalWrite(d,HIGH);
  digitalWrite(e,HIGH);
  digitalWrite(f,HIGH);
  digitalWrite(g,HIGH);
  digitalWrite(point,HIGH);
}
void loop()
{
  for(int i=0;i<10;i++)
  {
    showNumber(i);
    delay(1000);
    turnOff();
  }
}
```

(ii) Cathode

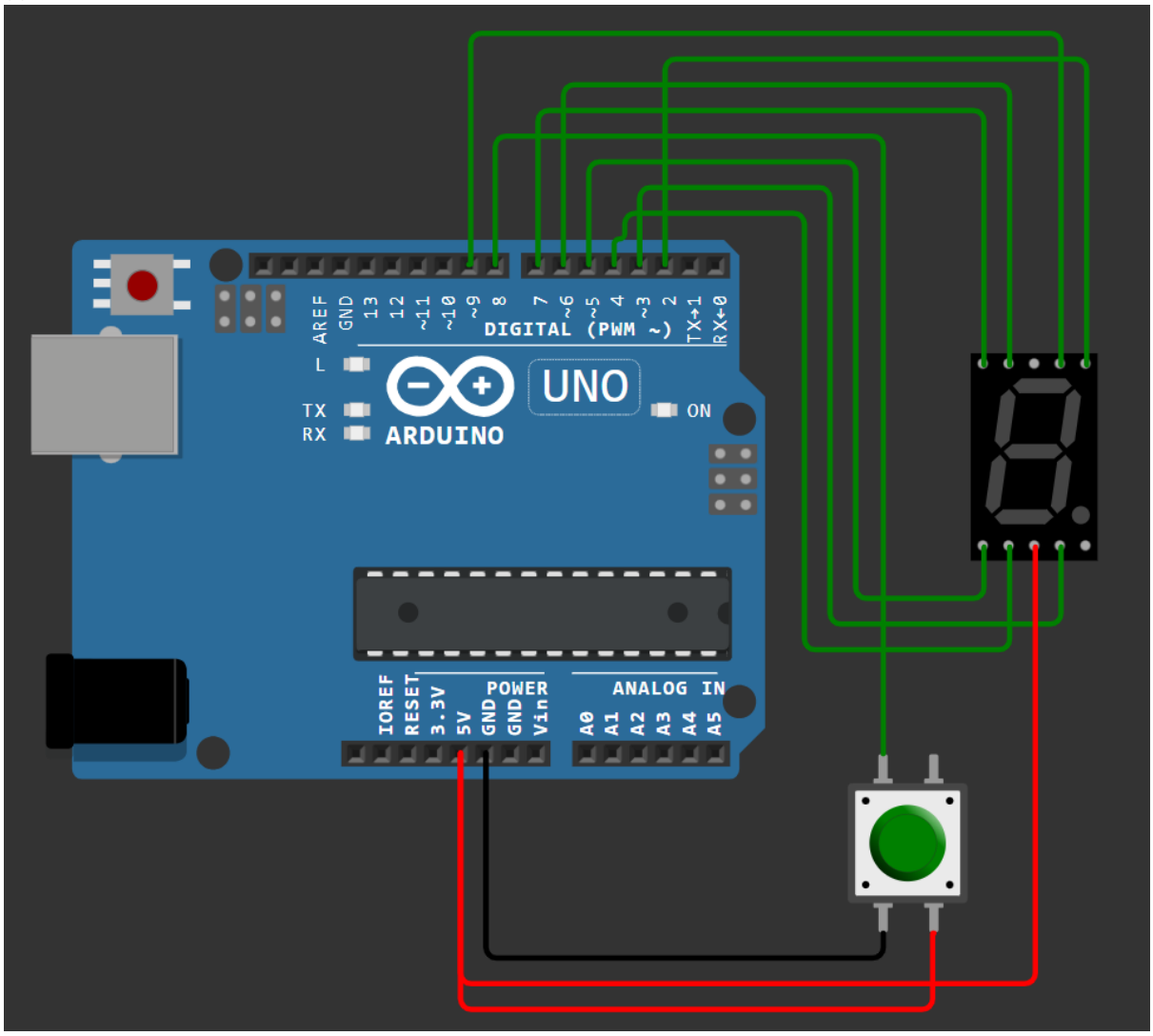

AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM ~)
L
TX
RX
ARDUINO
UNO
ON
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

```
int a = 2;
int b = 3;
int c = 4;
int d = 5;
int e = 6;
int f = 7;
int g = 8;
int point = 9;

void setup()
{
  pinMode(a, OUTPUT); //a
  pinMode(b, OUTPUT); //b
  pinMode(c, OUTPUT); //c
  pinMode(d, OUTPUT); //d
  pinMode(e, OUTPUT); //e
  pinMode(f, OUTPUT); //f
  pinMode(g, OUTPUT); //g
  pinMode(point, OUTPUT); //point
}

void showNumber(int number)
{
  if(number != 1 && number != 4)
    digitalWrite(a,HIGH);
  if(number != 5 && number != 6)
    digitalWrite(b,HIGH);
  if(number != 2)
    digitalWrite(c,HIGH);
  if(number != 1 && number != 4 && number != 7)
    digitalWrite(d,HIGH);
  if(number == 2 || number == 6 || number == 8 || number == 0)
    digitalWrite(e,HIGH);
  if(number != 1 && number != 2 && number != 3 && number != 7)
    digitalWrite(f,HIGH);
  if (number != 0 && number != 1 && number != 7)
    digitalWrite(g,HIGH);
}

void turnOff()
{
  digitalWrite(a,LOW);
  digitalWrite(b,LOW);
  digitalWrite(c,LOW);
  digitalWrite(d,LOW);
  digitalWrite(e,LOW);
  digitalWrite(f,LOW);
  digitalWrite(g,LOW);
```

```
  digitalWrite(point,LOW);
}

void loop()
{
  for(int i=0;i<10;i++)
  {
    showNumber(i);
    delay(1000);
    turnOff();
  }
}
```

(4.2) Increment with Push button:

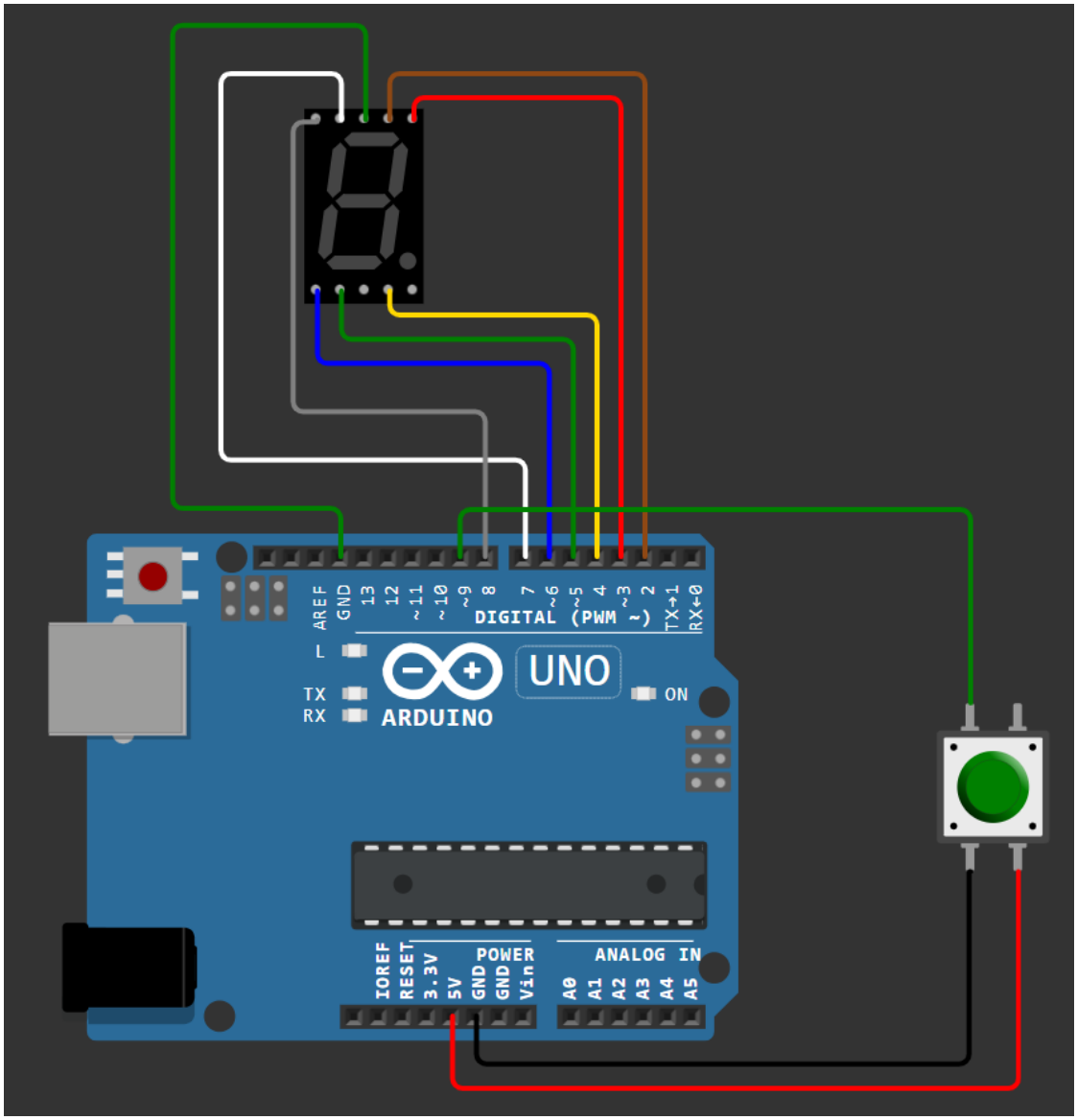
(i) Anode



```
int A = 9;
int B = 2;
int C = 3;
int D = 4;
int E = 5;
int F = 6;
int G = 7;
int switchUpPin = 8;
int counter = 0;
int buttonUpState = 0;
int lastButtonUpState = 0;

void setup() {
  Serial.begin(9600);
  pinMode(A, OUTPUT);
  pinMode(B, OUTPUT);
  pinMode(C, OUTPUT);
  pinMode(D, OUTPUT);
  pinMode(E, OUTPUT);
  pinMode(F, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(switchUpPin, INPUT);
}

void loop() {
  buttonUpState = digitalRead(switchUpPin);

  if (buttonUpState != lastButtonUpState) {
    if (buttonUpState == HIGH) {
      counter = (counter + 1) % 10; // Increment and wrap to 0 after 9
      showNumber(counter);
      delay(300); // Debounce delay
    }
    lastButtonUpState = buttonUpState;
  }
  delay(50); // Small delay to prevent bouncing
}

void showNumber(int number) {
  // Turn off all segments before setting the new number (HIGH for anode)
  digitalWrite(A, HIGH);
  digitalWrite(B, HIGH);
  digitalWrite(C, HIGH);
  digitalWrite(D, HIGH);
  digitalWrite(E, HIGH);
  digitalWrite(F, HIGH);
  digitalWrite(G, HIGH);
```

```
  // Set segments according to the number (LOW to turn on in anode configuration)
  if (number != 1 && number != 4)
    digitalWrite(A, LOW);
  if (number != 5 && number != 6)
    digitalWrite(B, LOW);
  if (number != 2)
    digitalWrite(C, LOW);
  if (number != 1 && number != 4 && number != 7)
    digitalWrite(D, LOW);
  if (number == 2 || number == 6 || number == 8 || number == 0)
    digitalWrite(E, LOW);
  if (number != 1 && number != 2 && number != 3 && number != 7)
    digitalWrite(F, LOW);
  if (number != 0 && number != 1 && number != 7)
    digitalWrite(G, LOW);
}
```

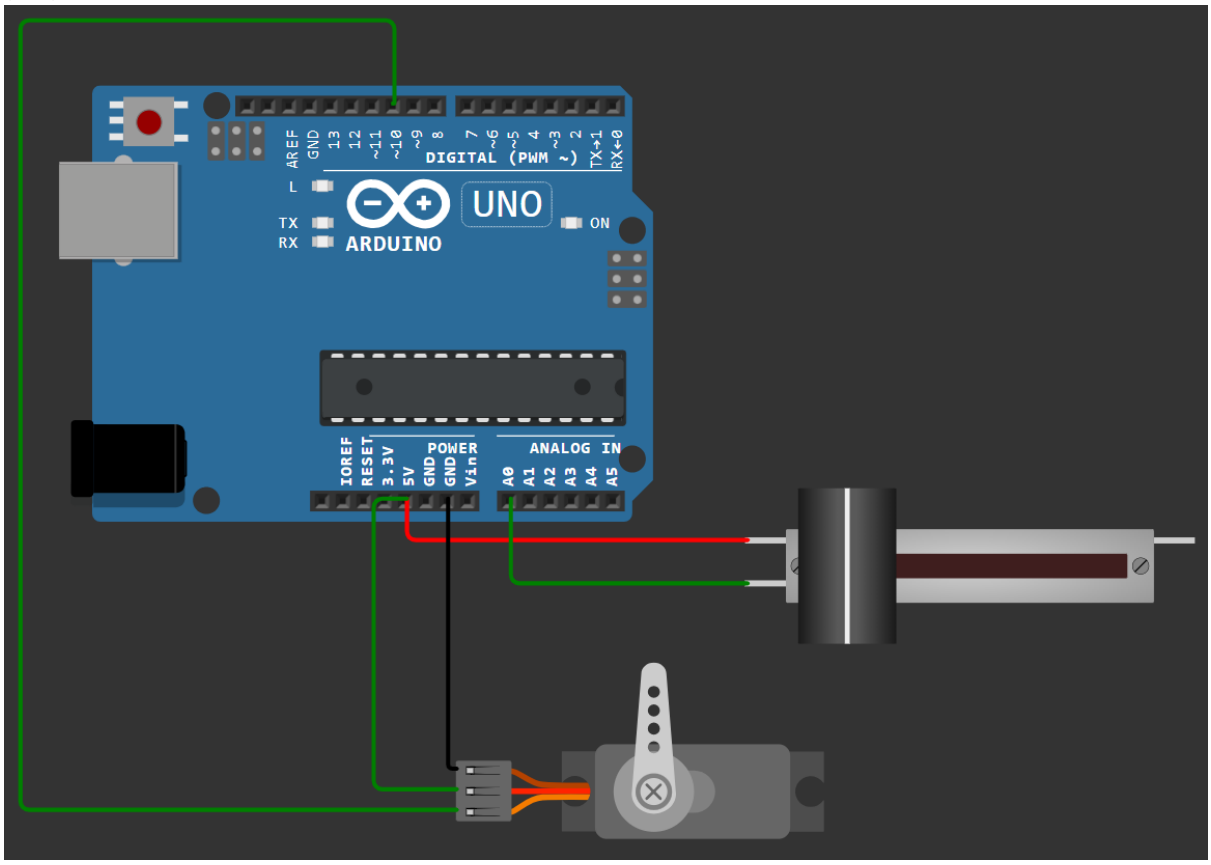
(ii) Cathode



```
int A = 2;
int B = 3;
int C = 4;
int D = 5;
int E = 6;
int F = 7;
int G = 8;
int switchUpPin = 9;
int counter = 0;
int buttonUpState = 0;
int lastButtonUpState = 0;

void setup() {
  Serial.begin(9600);
  pinMode(A, OUTPUT);
  pinMode(B, OUTPUT);
  pinMode(C, OUTPUT);
  pinMode(D, OUTPUT);
  pinMode(E, OUTPUT);
  pinMode(F, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(switchUpPin, INPUT);
}

void loop() {
  buttonUpState = digitalRead(switchUpPin);

  if (buttonUpState != lastButtonUpState) {
    if (buttonUpState == HIGH) {
      counter = (counter + 1) % 10; // Increment and wrap to 0 after 9
      showNumber(counter);
      delay(300); // Debounce delay
    }
    lastButtonUpState = buttonUpState;
  }
  delay(50); // Small delay to prevent bouncing
}

void showNumber(int number) {
  // Turn off all segments before setting the new number
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, LOW);
```

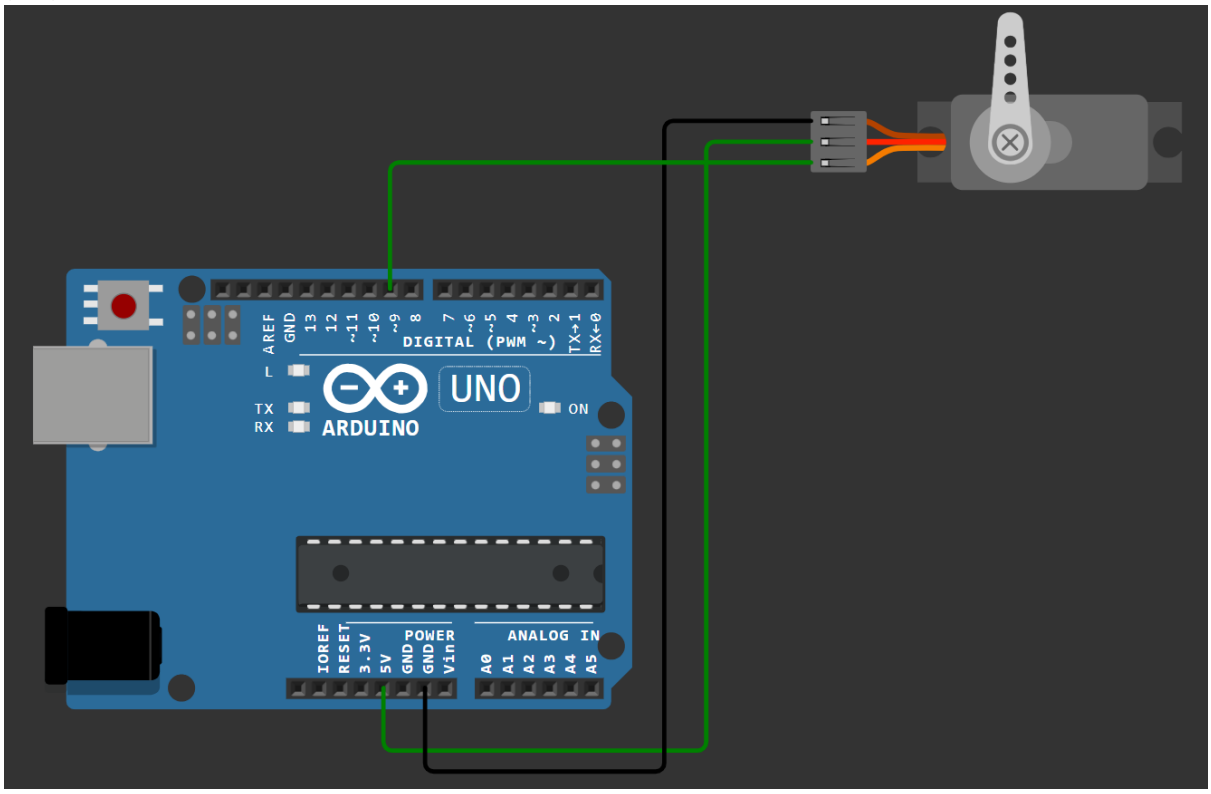
```
  // Set segments according to the number
  if (number != 1 && number != 4)
    digitalWrite(A, HIGH);
  if (number != 5 && number != 6)
    digitalWrite(B, HIGH);
  if (number != 2)
    digitalWrite(C, HIGH);
  if (number != 1 && number != 4 && number != 7)
    digitalWrite(D, HIGH);
  if (number == 2 || number == 6 || number == 8 || number == 0)
    digitalWrite(E, HIGH);
  if (number != 1 && number != 2 && number != 3 && number != 7)
    digitalWrite(F, HIGH);
  if (number != 0 && number != 1 && number != 7)
    digitalWrite(G, HIGH);
}
```

(5.1) Servo with Potentiometer:



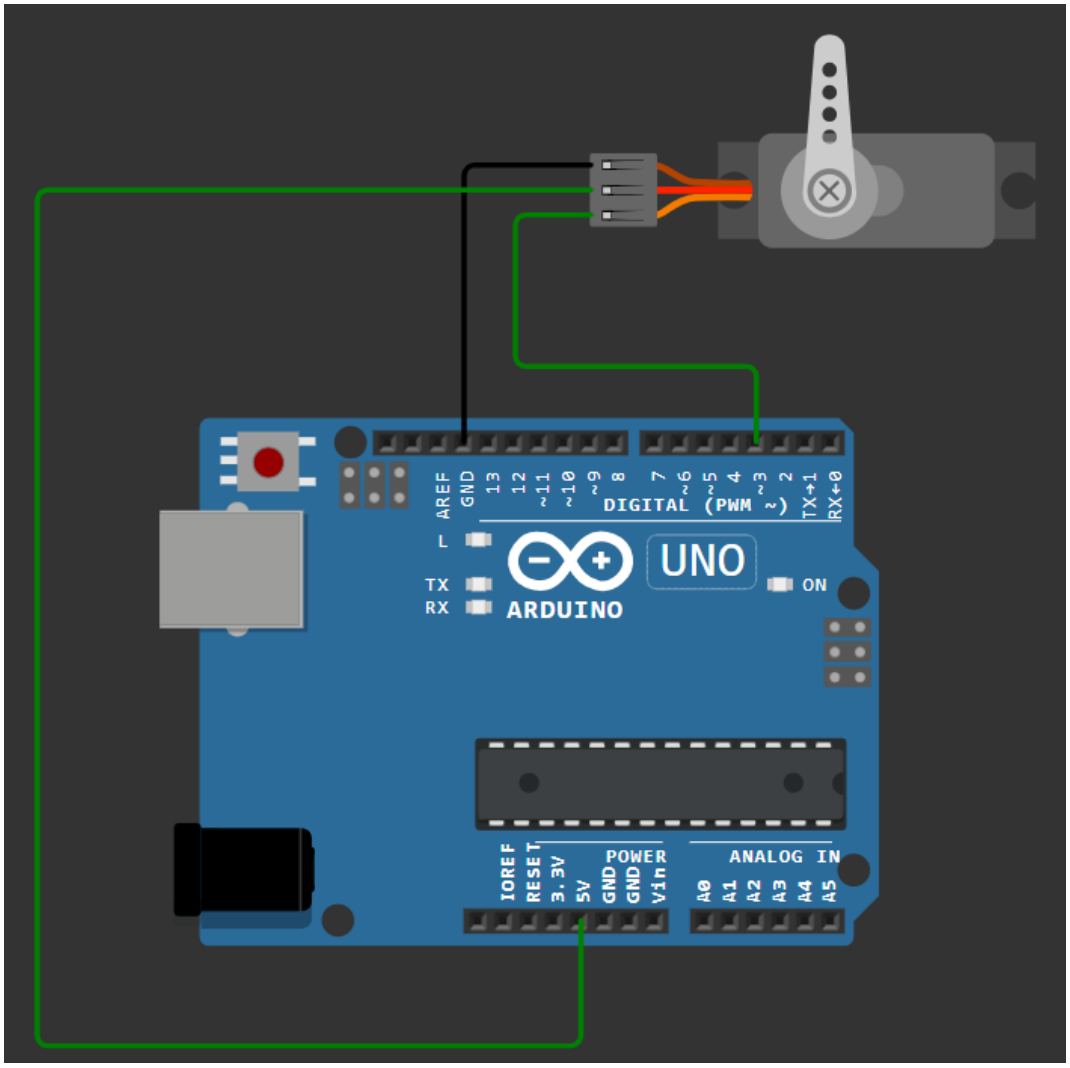
```
#include <Servo.h>

Servo myservo;
const int servo=10,
potentiometer=A0;
int pos=0;
void setup() {
  myservo.attach(servo);
  myservo.write(pos);
}
void loop() {
  int value=analogRead(potentiometer);
  pos=map(value,0,1023,0,180);
  myservo.write(pos);
  delay(20);
}
```

(5.2) Servo To and Fro:



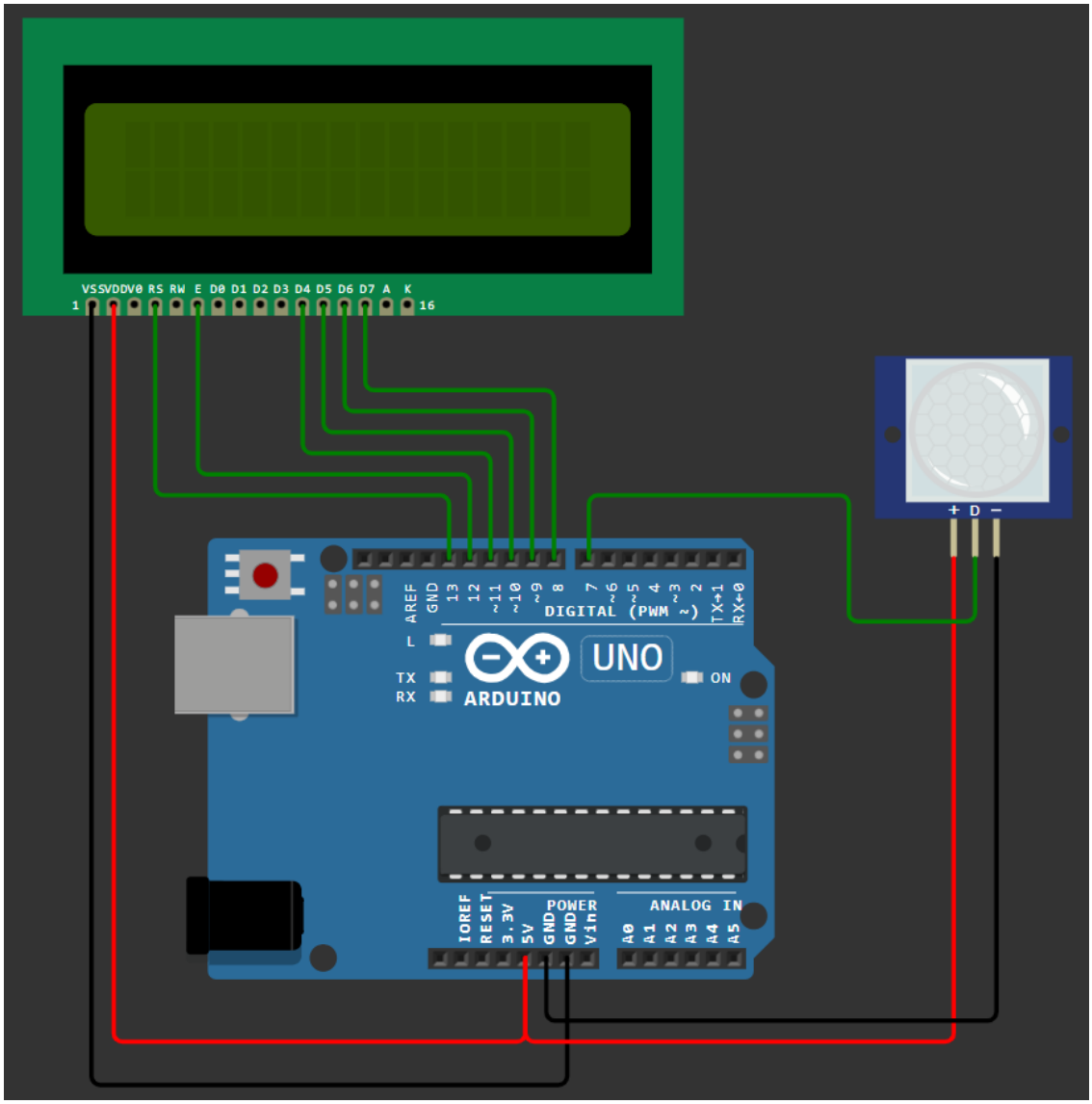
```
#include<Servo.h>

Servo myservo;
void setup() {
  myservo.attach(9);
  myservo.write(0);
}
void loop() {
  for(int i=0; i <= 180; i++){
    myservo.write(i);
    delay(15);
  }
  for(int j=180;j>=0;j -- ){
    myservo.write(j);
    delay(15);
  }
}
```

(5.3) Servo Full Rotation:



```
#include <Servo.h>
Servo myservo;

void setup() {
  myservo.attach(3);  // Attach the servo on pin 3
}

void loop() {
  myservo.write(0);
  delay(2000);
  myservo.write(30);
  delay(2000);
  myservo.write(60);
  delay(2000);
  myservo.write(90);
  delay(2000);
  myservo.write(120);
  delay(2000);
  myservo.write(150);
  delay(2000);
  myservo.write(180);
  delay(2000);
  myservo.write(150);
  delay(2000);
  myservo.write(120);
  delay(2000);
  myservo.write(90);
  delay(2000);
  myservo.write(60);
  delay(2000);
  myservo.write(30);
  delay(2000);
  myservo.write(0);
  delay(2000);
}
```

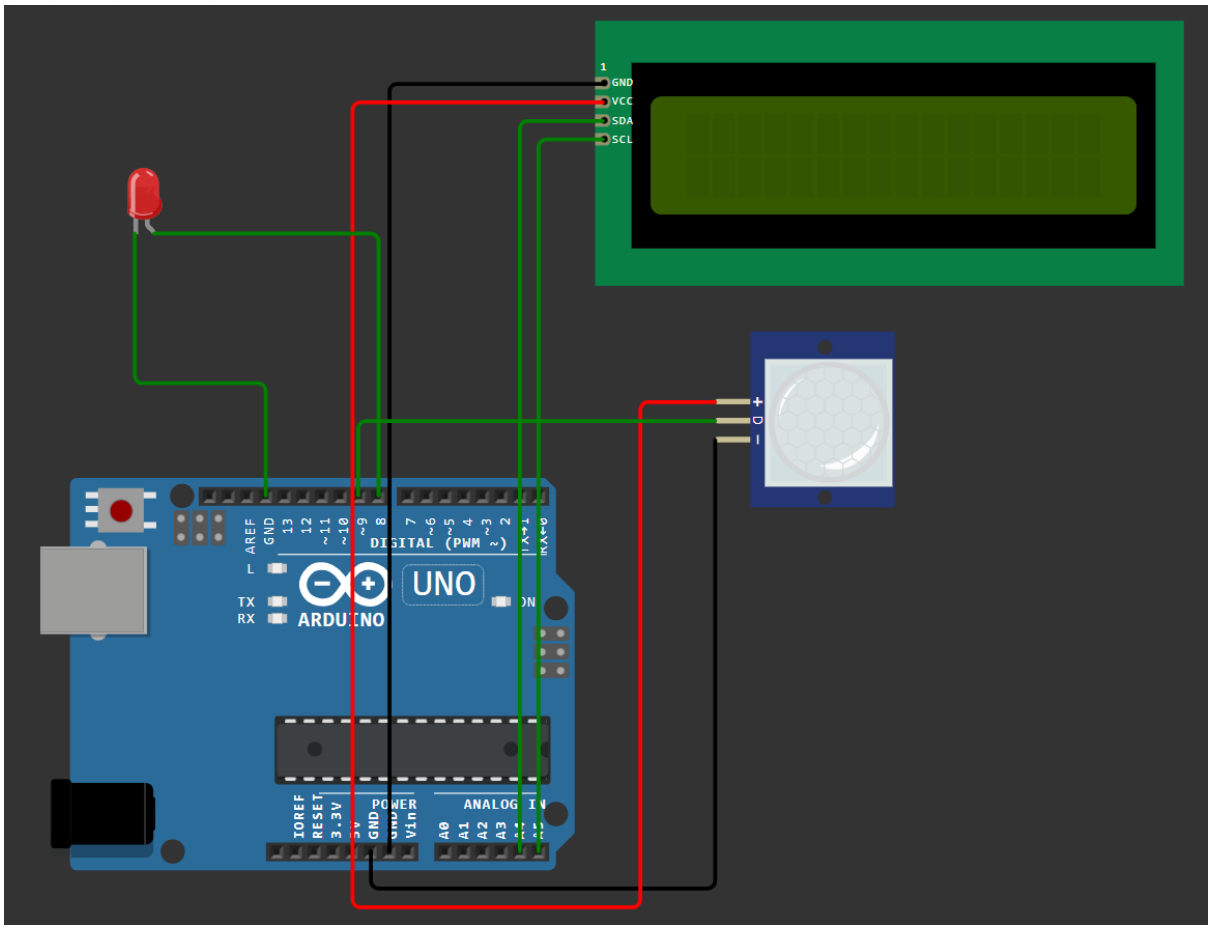
(6.1) PIR (LCD Normal):




```
#include <LiquidCrystal.h>
LiquidCrystal lcd(13,12,11,10,9,8);
int pir = 7;
int state = LOW;
int val = 0;
void setup() {
  pinMode(pir, INPUT);
  Serial.begin(9600);
  lcd.begin(16, 2);
}
void loop(){
  val = digitalRead(pir);
  if (val == HIGH) {
    delay(100);
    if (state == LOW) {
      lcd.clear();
      lcd.setCursor(0,0);
```

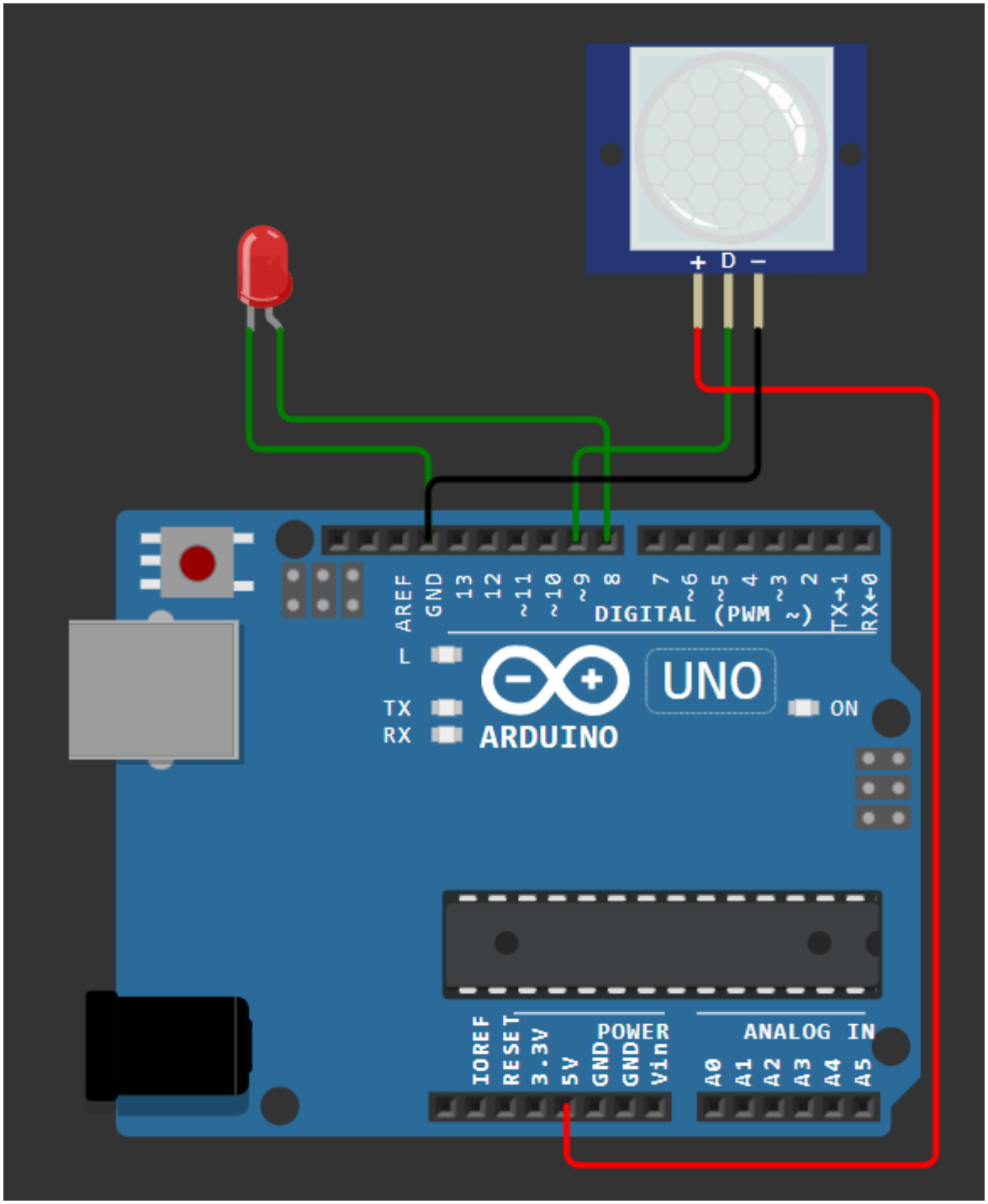
```
    lcd.print("Movement");
    lcd.setCursor(0,1);
    lcd.print("Detected!");
    state = HIGH;
   }
  }
  else{
   lcd.setCursor(0,0);
   lcd.print("No");
   lcd.setCursor(0,1);
   lcd.print("Movement");
   delay(200);
   if (state==HIGH){
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("No");
    lcd.setCursor(0,1);
    lcd.print("Movement");
    state=LOW;
   }
  }
}
```

(6.2) PIR (LCD I2C):



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

int led = 8;              // the pin that the LED is atteched to
int sensor = 9;           // the pin that the sensor is atteched to
int state = LOW;          // by default, no motion detected
int val = 0;              // variable to store the sensor status (value)

void setup() {
  lcd.init(); // Turn on the LCD backlight
  lcd.backlight();
 pinMode(led, OUTPUT);      // initalize LED as an output
 pinMode(sensor, INPUT);    // initialize sensor as an input
 Serial.begin(9600);       // initialize serial
}

void loop(){
 val = digitalRead(sensor);   // read sensor value
 if (val == HIGH) {           // check if the sensor is HIGH
   digitalWrite(led, HIGH);   // turn LED ON
   delay(100);                // delay 100 milliseconds

   if (state == LOW) {
     lcd.clear();
     Serial.println("Motion detected!");
     lcd.print("Motion detected");
     state = HIGH;            // update variable state to HIGH
   }
 }
 else {
     digitalWrite(led, LOW); // turn LED OFF
     delay(200);             // delay 200 milliseconds

     if (state == HIGH){
       lcd.clear();
       Serial.println("Motion stopped!");
       lcd.print("Motion stopped");
       state = LOW;       // update variable state to LOW

   }
 }
}
```

(6.3) PIR without LCD:




```
int led = 8;
int pir = 9;
int state = LOW;
int val = 0;

void setup() {
  pinMode(led, OUTPUT);
  pinMode(pir, INPUT);
  Serial.begin(9600);
}
```

```
void loop() {
  val = digitalRead(pir);

  if (val == HIGH) {
    digitalWrite(led, HIGH);
    delay(100);

    if (state == LOW) {
      Serial.println("Movement detected!");
      state = HIGH;
    }
  } else {
    digitalWrite(led, LOW);
    delay(200);

    if (state == HIGH) {
      Serial.println("No movement!");
      state = LOW;
    }
  }
}
```