CHAPTER ONE
AUTOMATIC LIGHTING SYSTEM

Learning Outcomes

When you have completed this module, you will:

- Understand how to integrate an LDR with Arduino.
- build an automatic lighting system.
- identify the usefulness of an automatic lighting system.

Introduction

Electricity, being one of the most vital resources for humans, ought to be utilized appropriately. Imagine putting on the street lights in your neighbourhood by yourself or having to turn on and off the security lights in your house everyday. That could be really stressful, right? And at times, one might just forget to switch on the security lights in your house, which is bad for security reasons, or simply forgetting to switch off this light which results in energy wastage. Now, let us imagine a better way to take care of this problem and make life easier for ourselves. Imagine a light system that switches on and off by itself at the appropriate time.

Do not think too far, the solution to this problem is right here at our fingertips. It is called an *automatic lighting system.* The next question is how does this work? Very simple, the lamp automatically lights up when it becomes dark to provide light for security and/or for passers-by.

Components

To build this project, we will require some electronic components which are;

- Arduino UNO
- LDR (Light Dependent Resistor)

- Resistor (one 100kΩ and 330Ω)
- LED - 1
- Connecting wires
- Breadboard
- And lastly, a PC (personal computer) with the Arduino IDE installed to write the Arduino code

Schematics

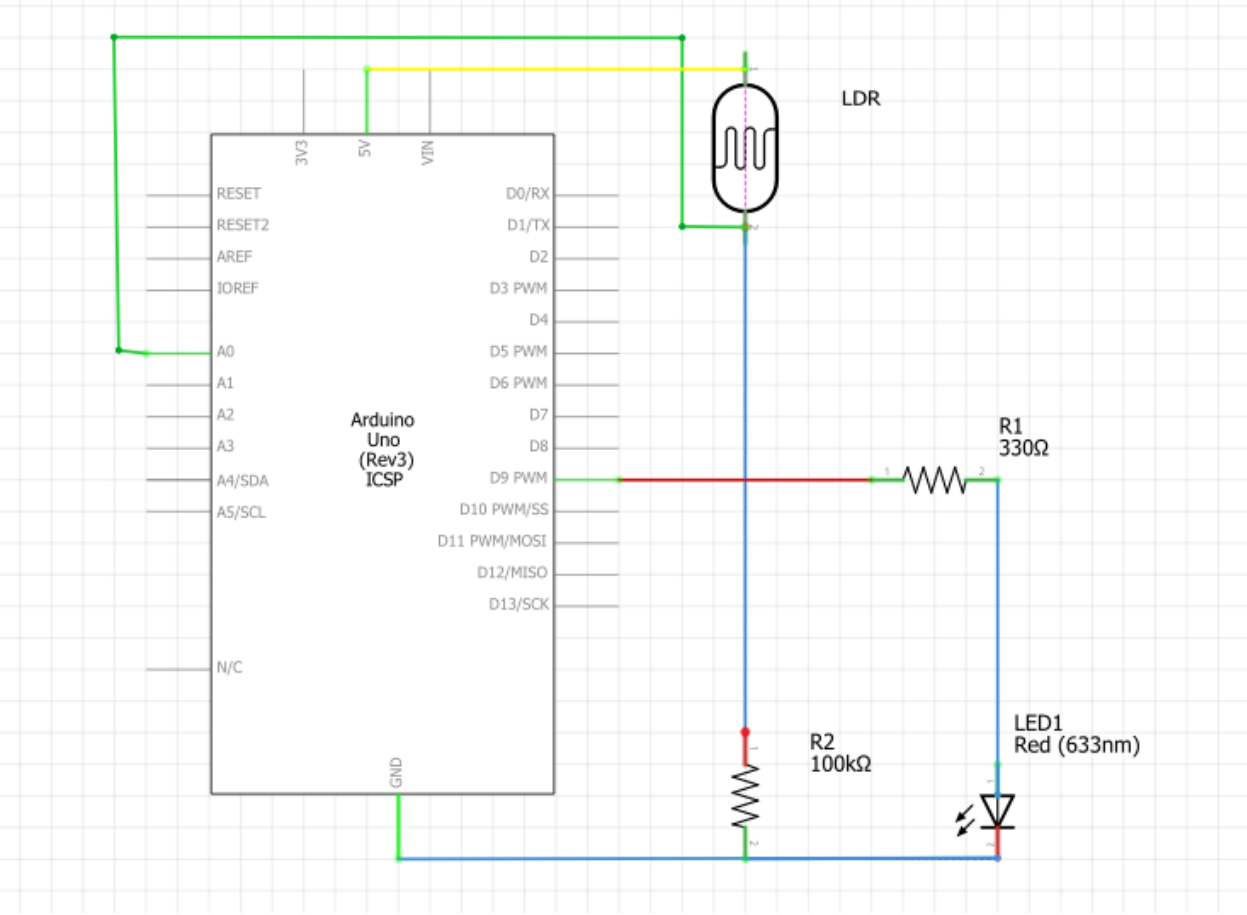


Figure 1.0

The image above shows the schematic of the automatic lighting system circuit

Connections

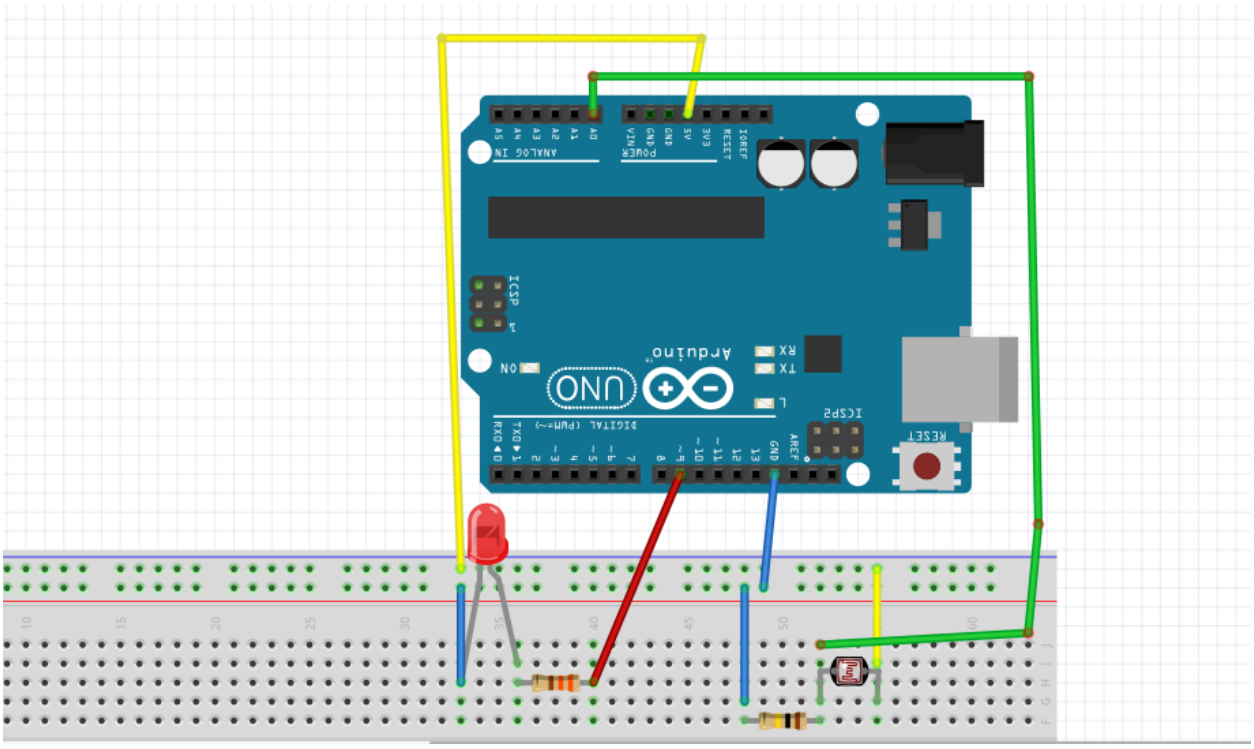


Figure 1.1

Connection procedure

1. Connect the shorter leg of your LED to the common ground of the Arduino as shown in figure 1.2.

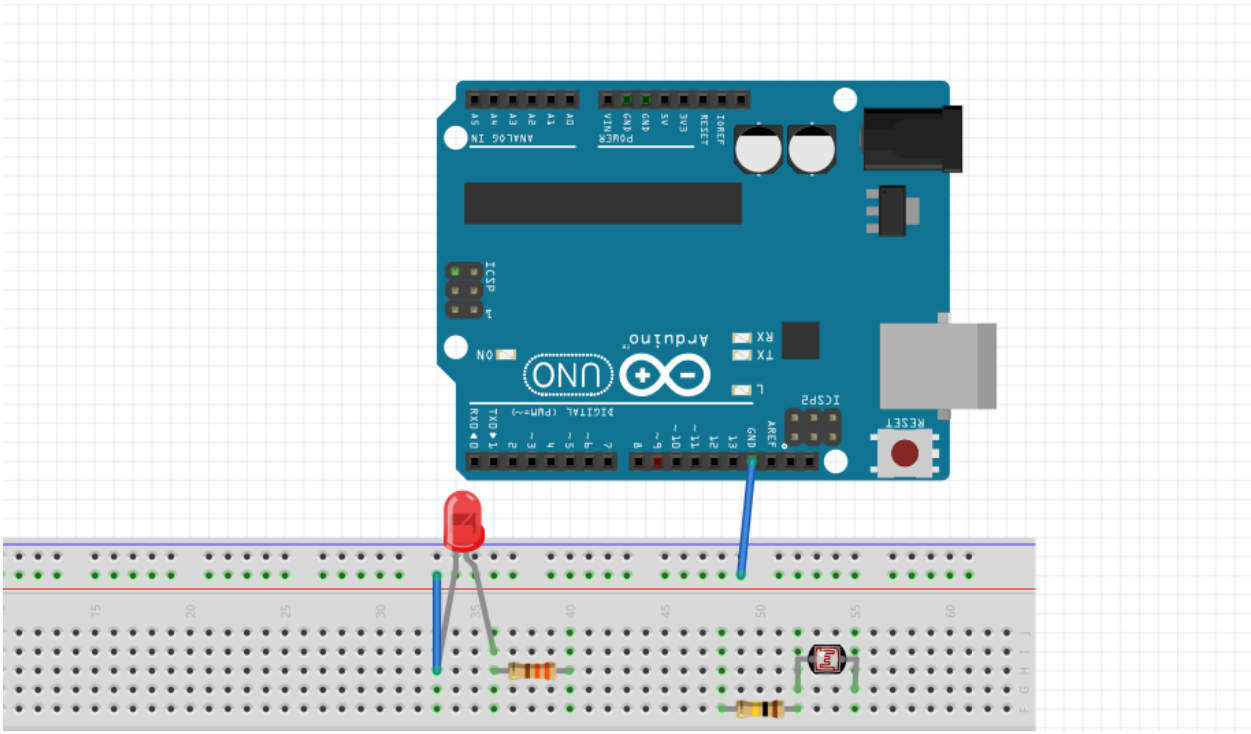


Figure 1.2

2. Now connect the longer leg of your LED to one of the legs of the 330Ω and the other end of the resistor to pin 9 on your Arduino as shown in figure 1.3.

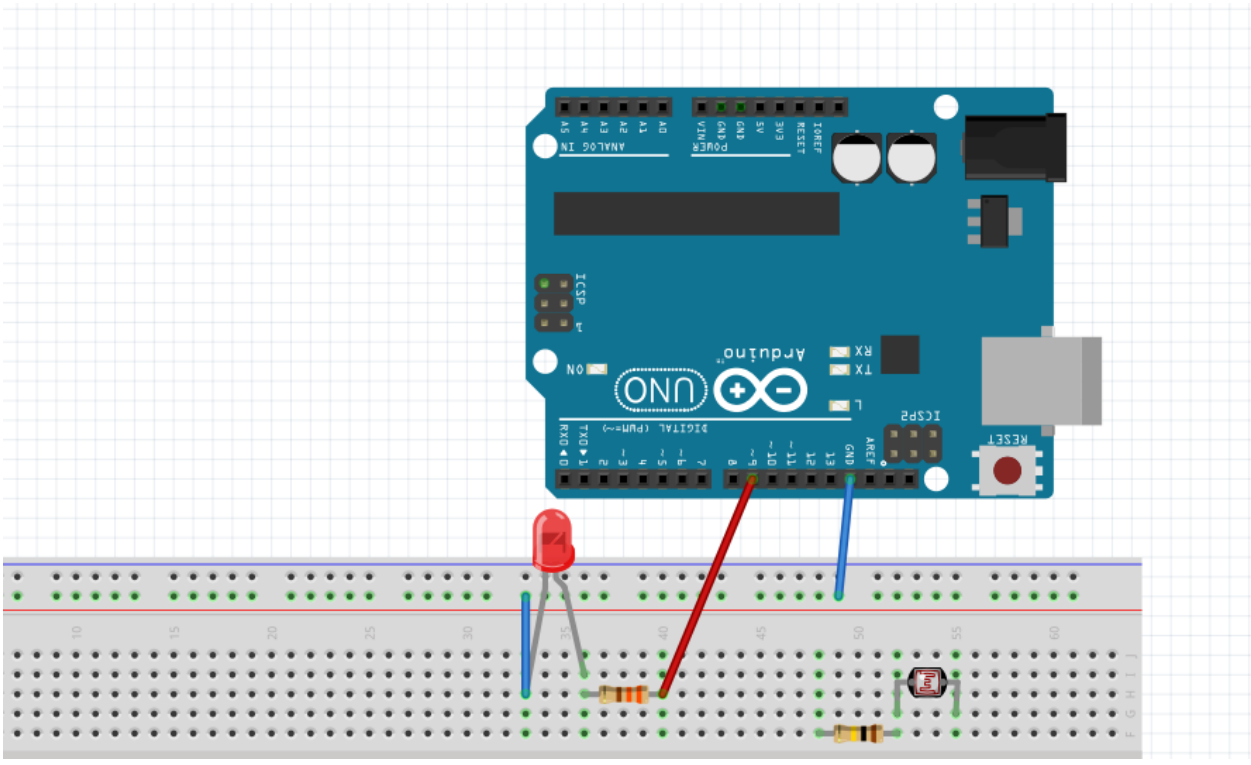


Figure 1.3

3. Next, we will connect one leg of our 100kΩ to our LDR and ground the other as shown in figure 1.4.

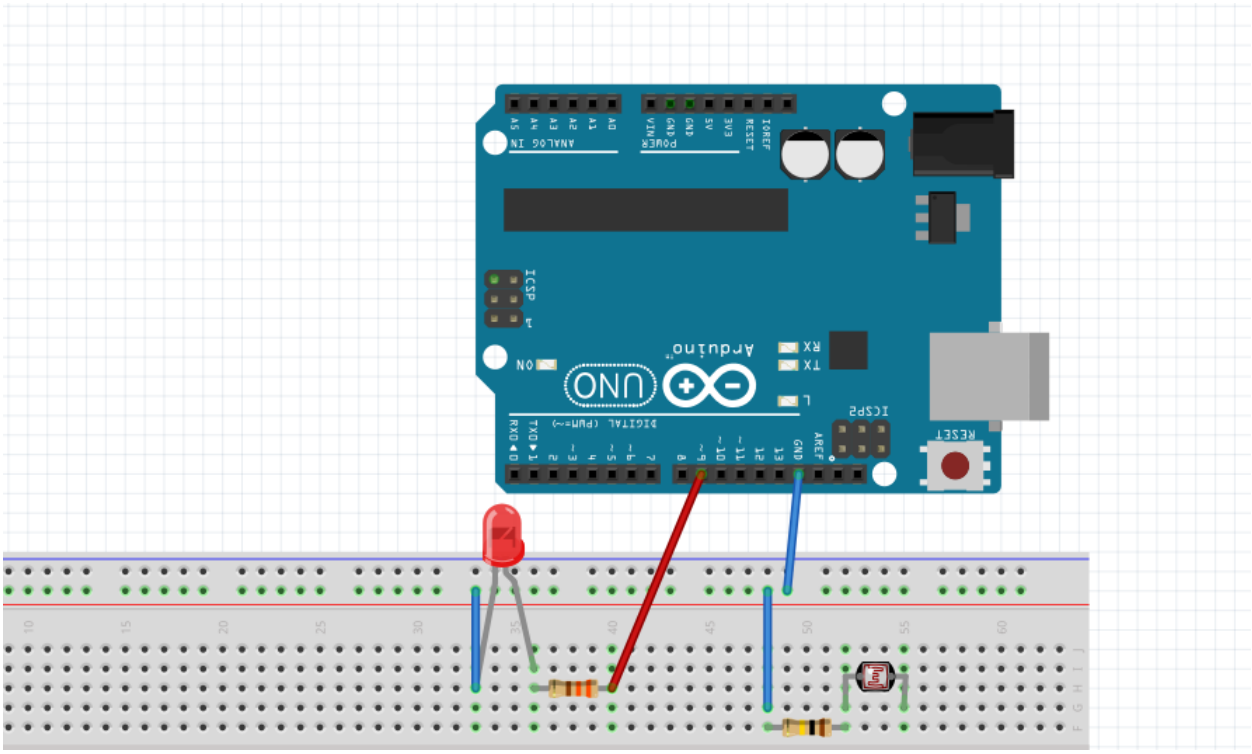


Figure 1.4

4. Next step is connecting the second leg of our LDR to the 5V pin on our Arduino as shown in figure 1.5.

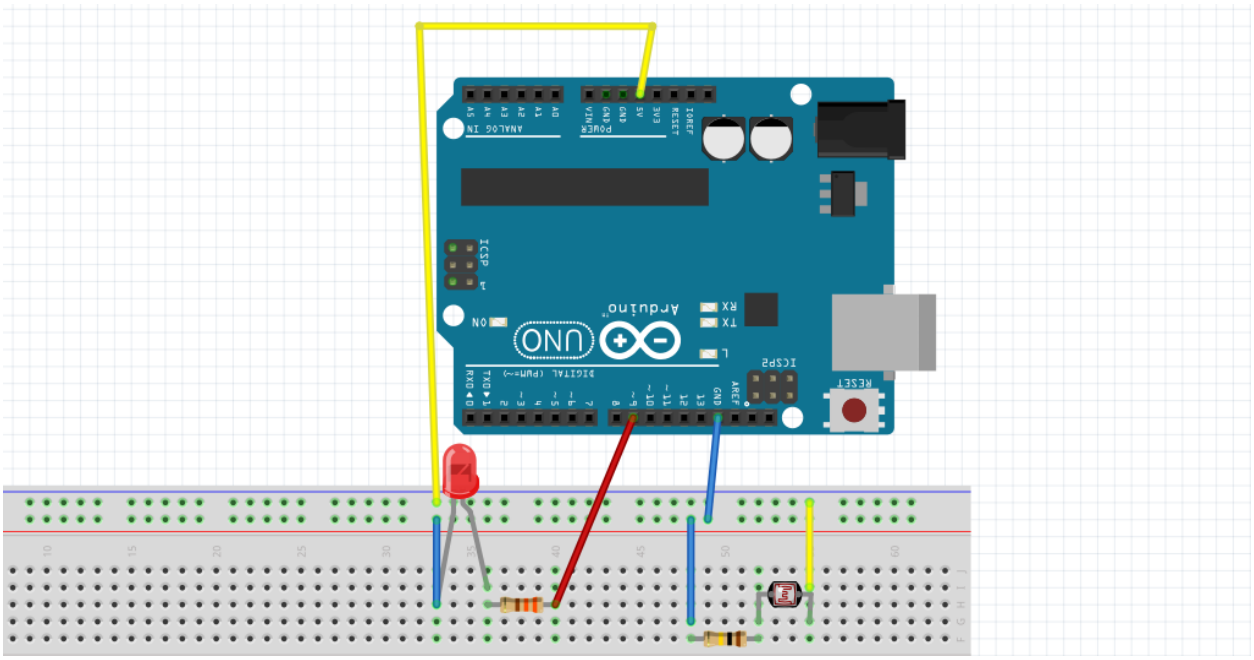


Figure 1.5

5. Next step is connecting the second leg of the LDR to pin A0 as shown in figure 1.6.

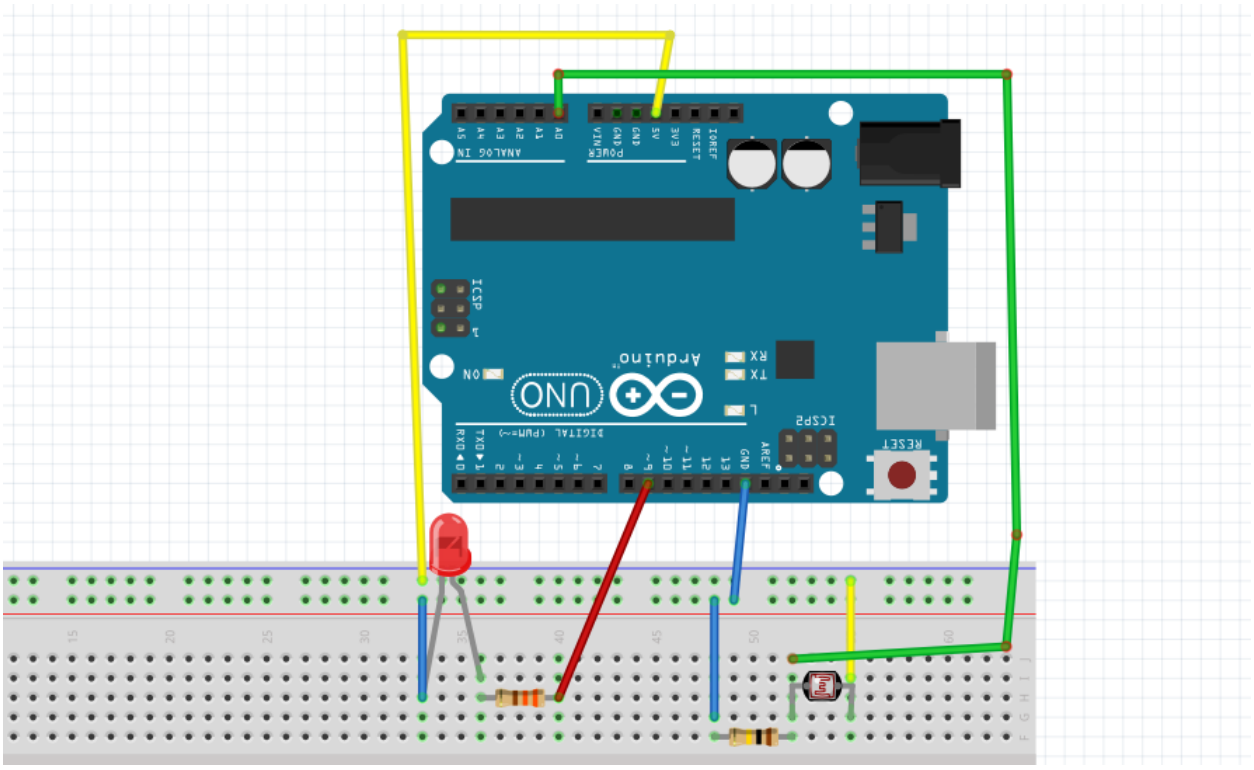


Figure 1.6

Code

After properly connecting the circuit, the next step is to type and upload the code to the Arduino board to enable the circuit work. Each line of code shall be explained during the process of writing. But first, we shall define the pins for LDR and LED.

```
// Project - Automatic lightning system

int LED = 9;

int LDR = A0;
```

Setting up the LED as an Output pin, and LDR as an input pin.

```
pinMode(LED, OUTPUT);

pinMode(LDR, INPUT);
```

Reading the voltage analog value through the A0 pin of the Arduino: This analog Voltage will be increased or decreased according to the resistance of LDR.

```
int LDRValue = analogRead(LDR);
```

Giving the condition for darkness and brightness: If the value is less than 700, then, it is dark and the LED or Light turns **ON** but if the value is greater than 700 then it is bright and the LED or light turns **OFF**.

```
if (LDRValue <=700)

{

digitalWrite(LED, HIGH);

digitalWrite(relay, HIGH);
```

```
Serial.println("It's Dark Outside; Lights status: ON");

}

else

{

digitalWrite(LED, LOW);

digitalWrite(relay, LOW);

Serial.println("It's Bright Outside; Lights status: OFF");

}
```

After typing this code, the next thing to do is upload this code into the arduino and power it up.

Testing

Once the circuit is powered on;

1. Hold out a flashlight pointing on the LDR, notice that the LED does not light up.
2. Now switch off the flashlight and subject the circuit to darkness, you will observe that the LED lights up.
3. Now if your circuit does not do this, check the connections and the codes and then, try again.

Summary Of Module One

CHAPTER TWO
DIGITAL TEMPERATURE SENSOR

Learning Outcomes

When you have completed this module, you will:

- Understand how to integrate a thermistor with Arduino.
- Know how to build a digital temperature sensor system.
- Know the usefulness of a digital temperature sensor system.

Introduction

The digital temperature sensor system we will build is a simple circuit whose basic function is to tell you the exact temperature at a particular time. The digital temperature sensor system on an industrial scale could be further designed to automatically adjust temperature within a certain temperature range such that when the temperature rises above a preset upper range, it stops heating and when the temperature drops below the preset lower range, it starts heating up again, therefore, maintaining that particular range of temperature. A common application of this design is in your ovens at home.

Components

For this project, you will require this list of components;

- Arduino UNO
- Thermistor (100kΩ)
- Resistor (one 220kΩ)
- Connecting wires
- Breadboard

- And lastly, a PC (personal computer) with the Arduino IDE installed to write the Arduino code

Schematics

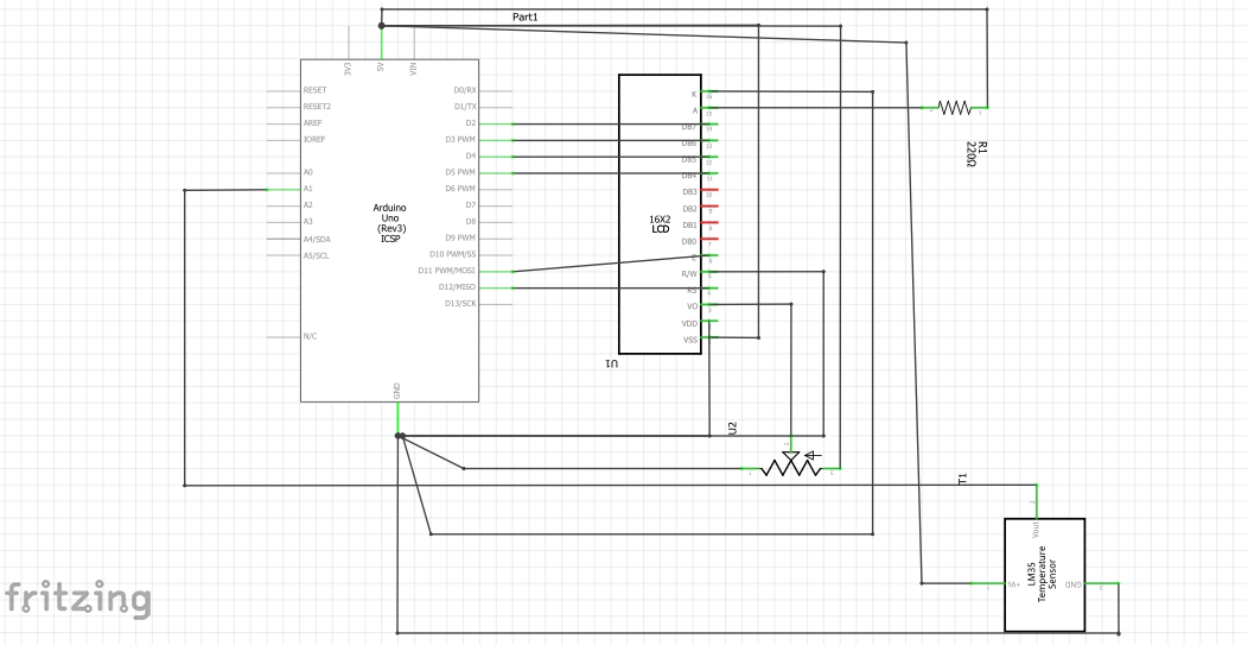


Figure 2.0

The image above shows the schematic of the digital temperature sensor system circuit

Connection

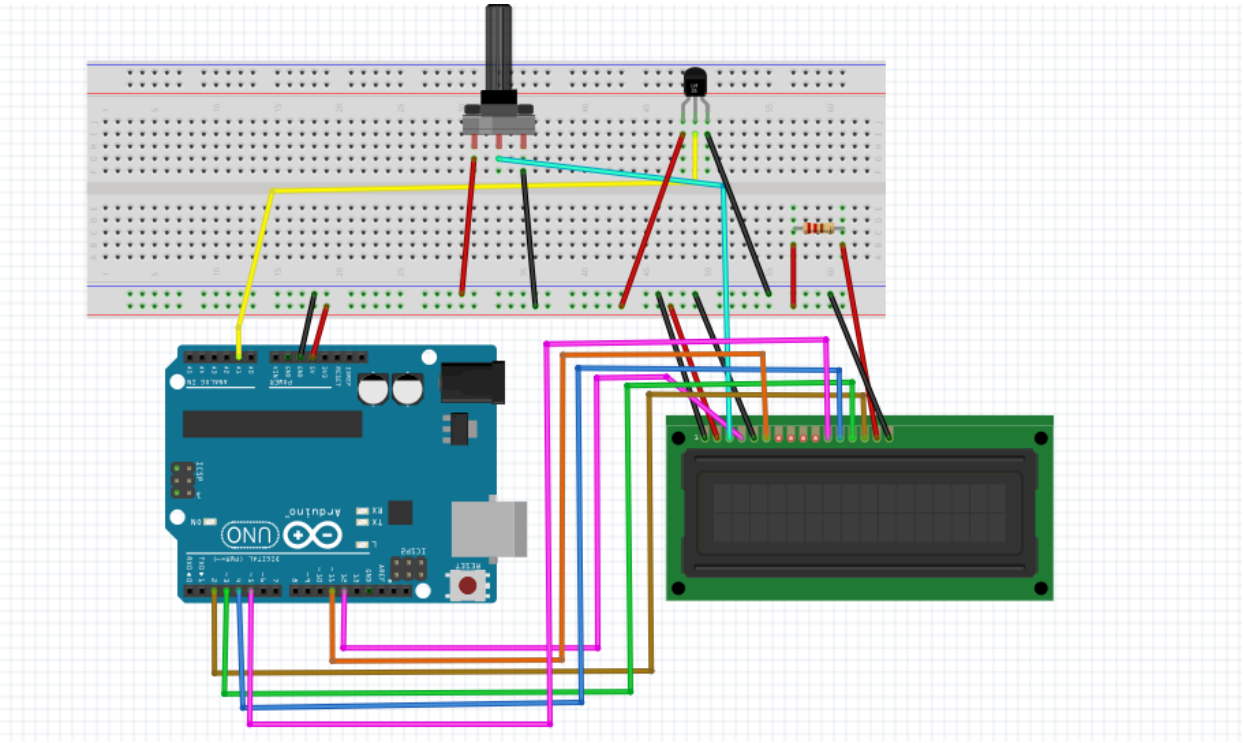


Figure 2.1

To achieve the connection of this circuit, the following steps have to be followed.

Steps

- The first step is to connect the 5V and ground terminal from the Arduino to the breadboard as shown below:

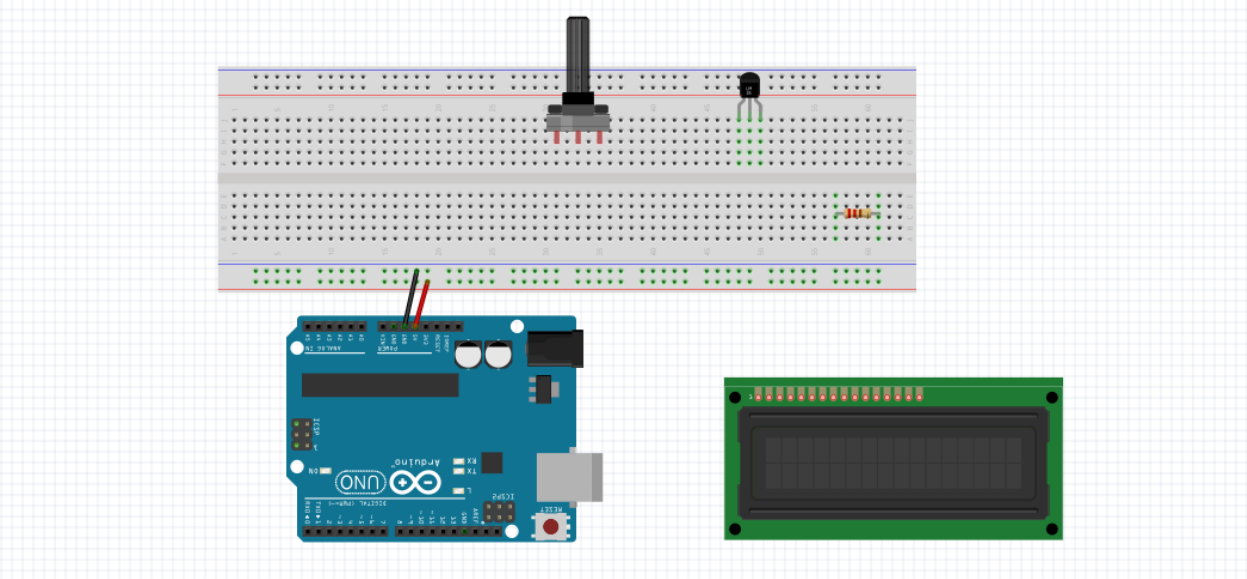


Figure 2.2

- Now we will connect our thermistor and potentiometer to both ground and the supply voltage on the breadboard as shown in figure 2.3 below.

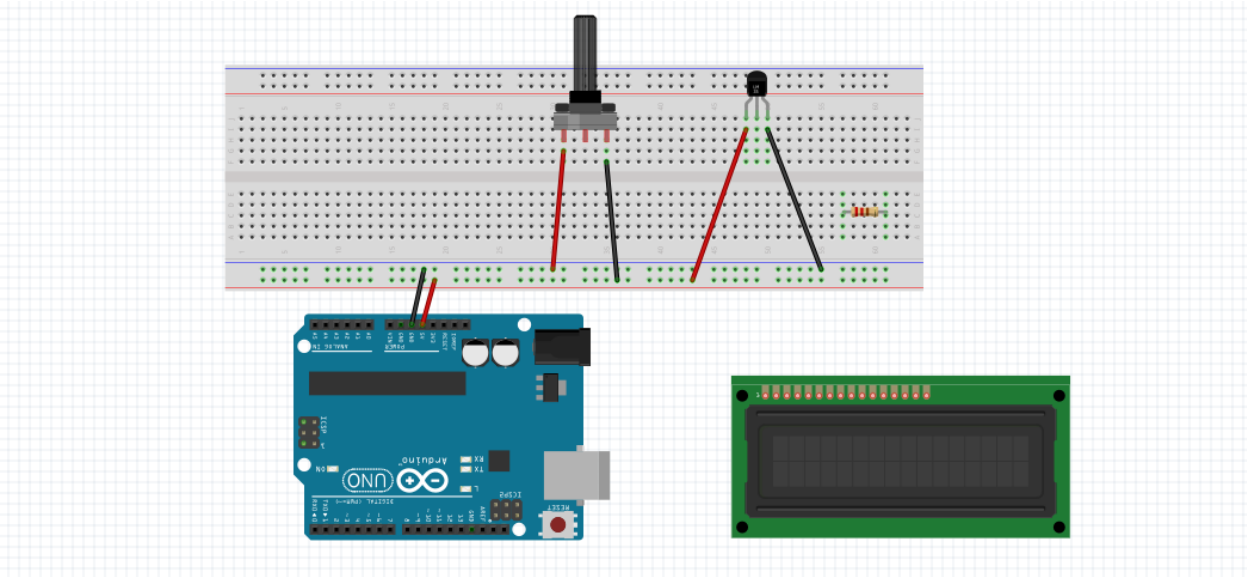


Figure 2.3

- Let us connect our LCD, shall we?

  First, connect each pin of the LCD to 5V and ground terminal as shown below;

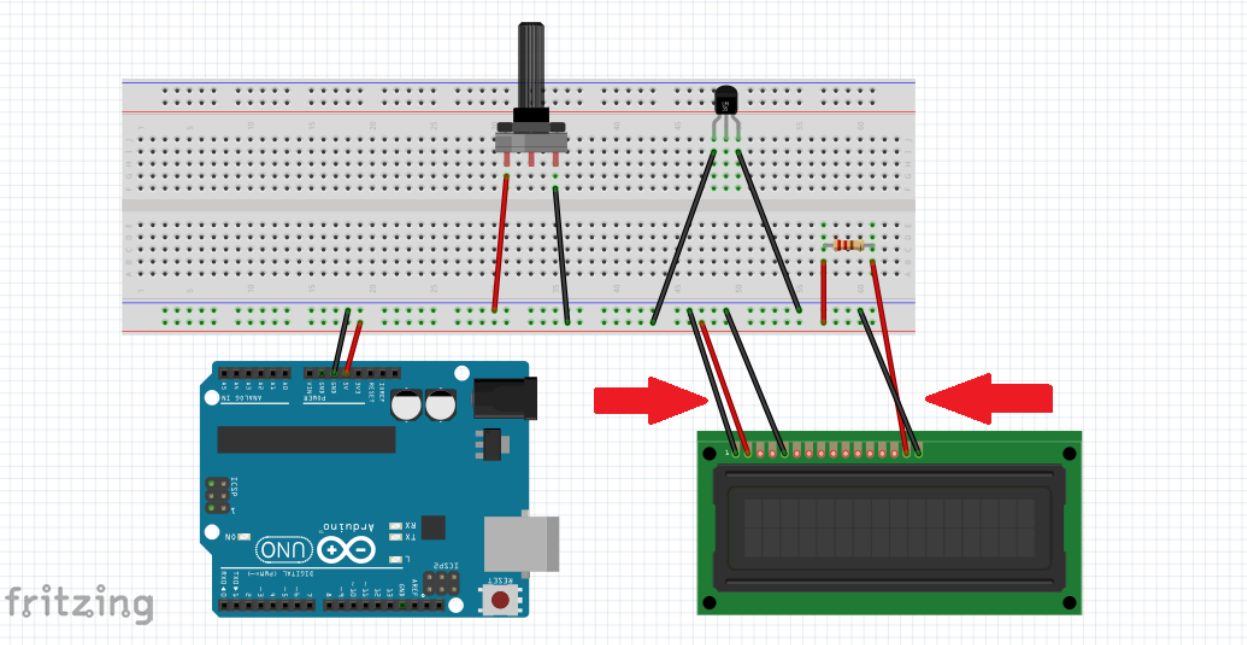


Figure 2.4

LCD connection Reminder:

1. The first two pins from the left labelled GND and VCC are the supply pins ground and 5V respectively.
2. The last two labelled A (anode) and K (cathode), these are used to power up the backlighting.

- Next, we shall connect the signal pins of our thermistor and potentiometer to pin A1 on the Arduino and Vo pin on our LCD respectively as shown in figure 2.5 below.

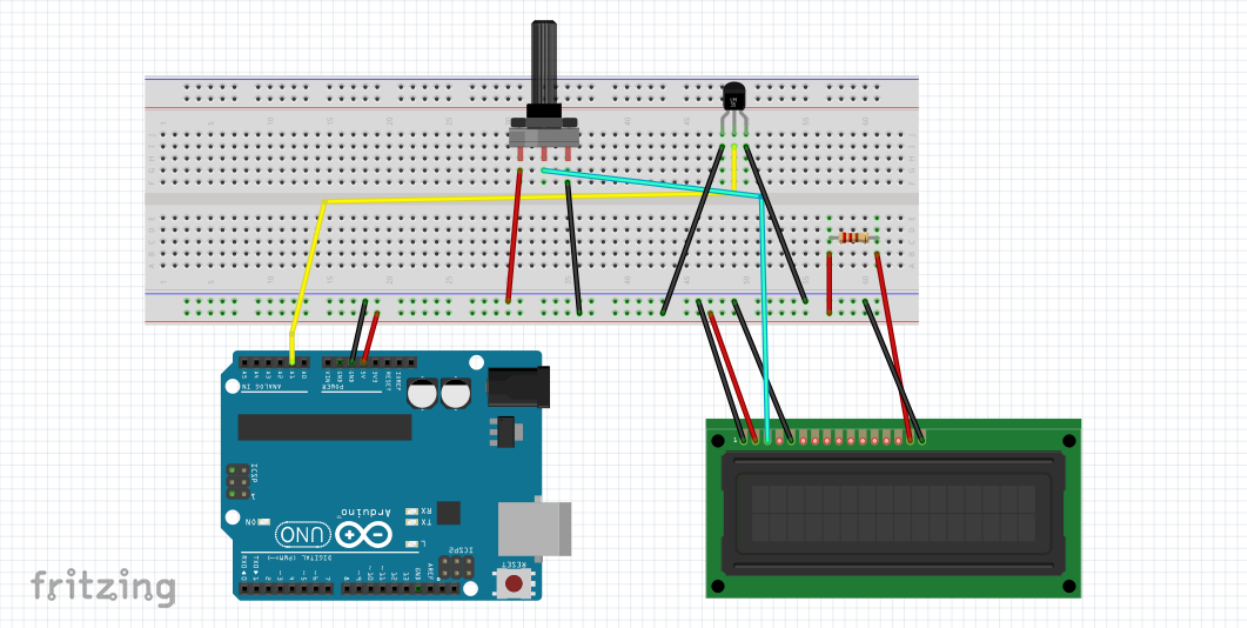


Figure 2.5

- The register ***select pin*** and ***enable pin*** is then connected to pins 12 and 11 on the Arduino respectively as shown below.

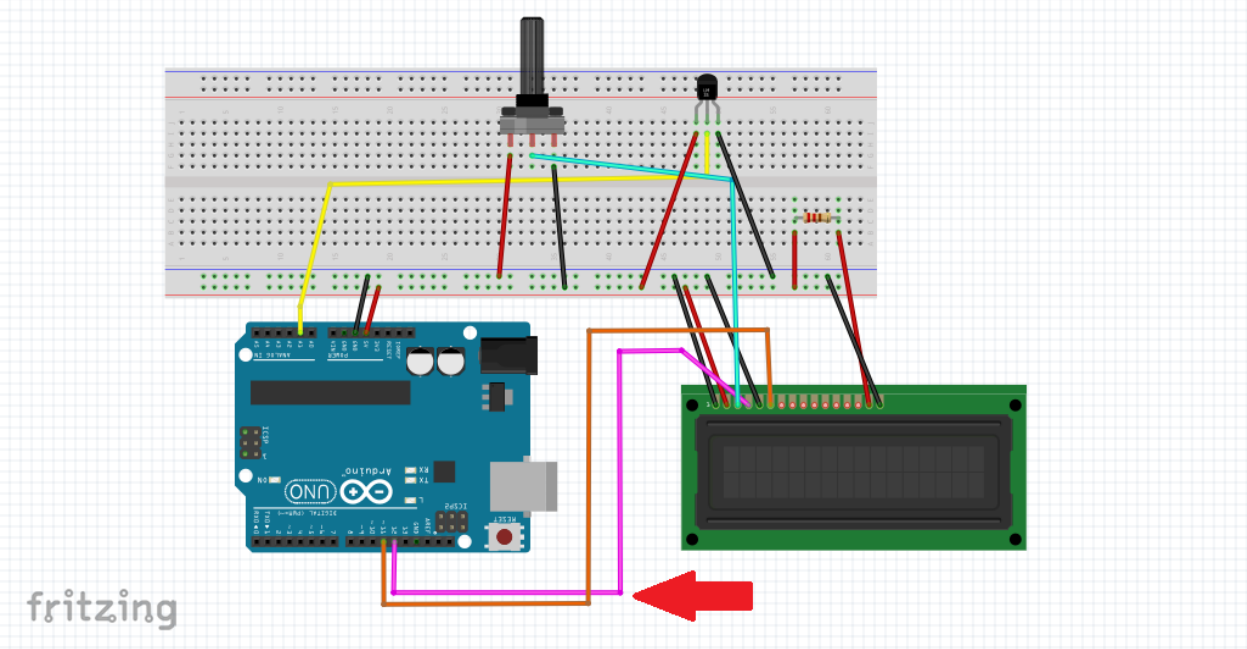
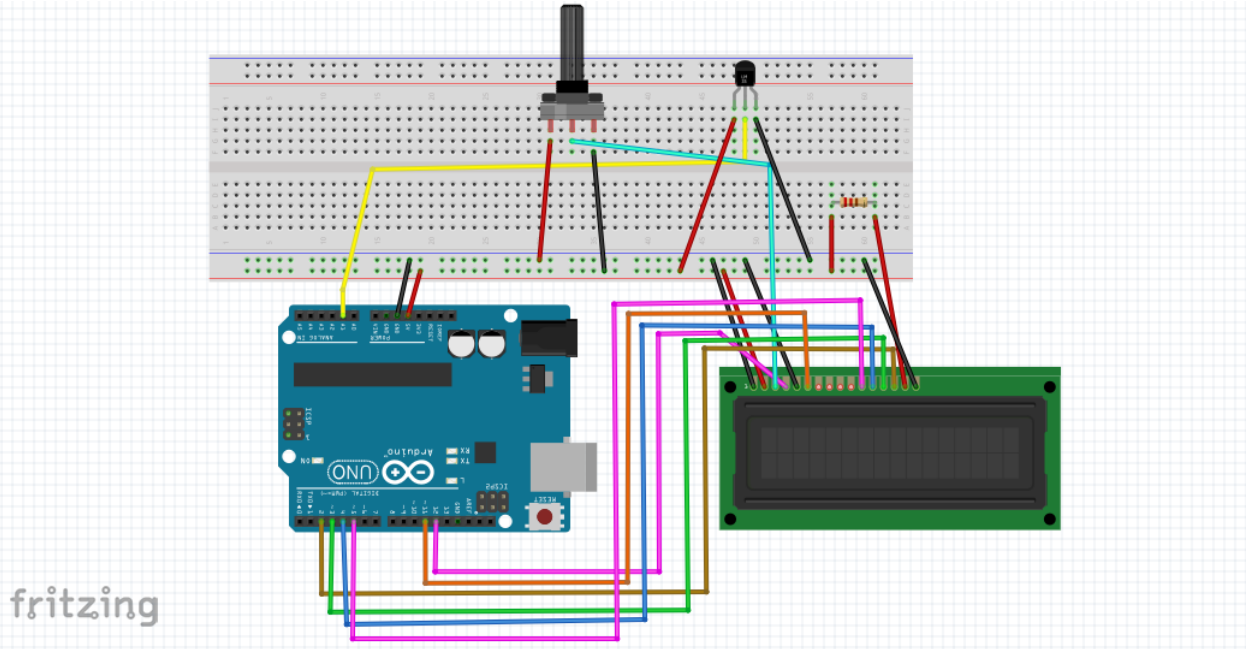


Figure 2.6

- Finally, we will connect data pins 4-7 on your LCD to pins 2-5 on your Arduino respectively as shown below.



Code

Type the code on your Arduino IDE, compile and export to your board.

```
#include <LiquidCrystal.h> // this adds the LCD library to enable your LCD work

LiquidCrystal lcd (12, 11, 5, 4, 3, 2); // this line of code is declaring the arduino pins connected to the lcd

int val; // this line of code initializes a variable

int tempPin = 1; // this line of code sets the initialized variable to pin 1

void setup() {

  lcd.begin(16,2);

}


void loop() {

 val = analogRead(tempPin); // this line code tells the arduino to read from the analog pin 1

 float mv = (val/1024.0)*50000; // this line of code is a pure mathematical code that computes the value read from the analog pin

 float cel = mv/10; // this line of code convert the computed value to celcius

 float farh = (cel*9)/5+32; // this converts celsius to fahrenheit

 lcd.print("Temperature=");

 lcd.setCursor(0,1);

 lcd.print(cel);

 lcd.setCursor(5,1);

 lcd.print("*c");

 delay(1000);
```

```
lcd.clear();
```

Summary Of Module Two