Rafael Mejia
rmeji3

Analog Input: Photoresistor (LDR – Light Dependent Resistor)

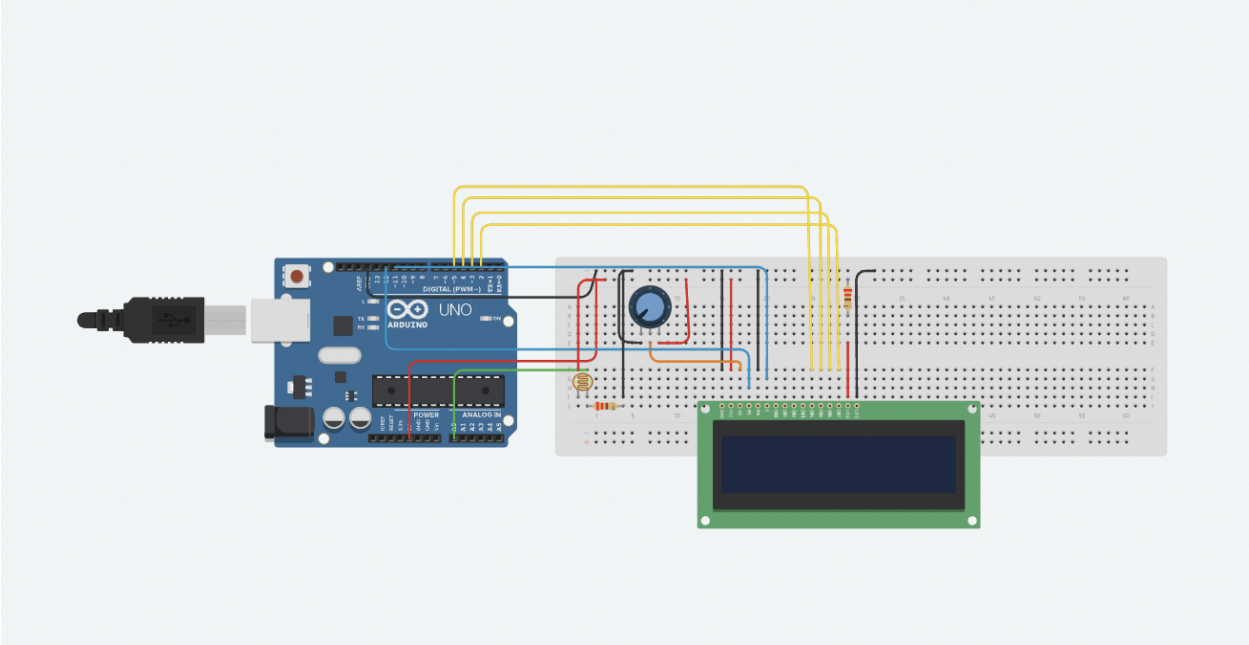
Summary

In this lab we created a circuit and program that will use a photoresistor and the 16x2 display. The display states the amount of light in the room as one of 5 predefined text values based on the value read from the photoresistor. The 5 predefined text values are dark, partially dark, medium, fully lit, brightly lit. These predefined values are displayed on the top line of the LCD display. The bottom line of the LCD displays the time in milliseconds since the arduino has reset and these values are continuously updating.

Required Hardware

- 20 wires
- 2 220 ohm resistors
- 1 photoresistor
- 1 potentiometer
- 1 16x2 LCD
- 1 breadboard
- 1 arduino uno

Design



Code

```
/*
// Rafael Mejia, 679124181 rmeji3
// Lab 4 - Analog Input: Photoresistor (LDR - Light Dependent Resistor)
// Description - Create a circuit and program that will use a photoresistor and the
16x2 display.
The display should state the relative amount of light in the room as one of 5
predefined text values based
on the value read from the photocell. The 5 predefined text values are:
• dark
• partially dark
• medium
• fully lit
• brightly lit
These values are to be displayed on the top line of the 16x2 LCD display.
On the bottom line of the LCD display, you are to display the number of milliseconds
since that reset of
the Arduino. This value must be continuously updating.
// References -
https://www.circuitbasics.com/how-to-use-photoresistors-to-detect-light-on-an-arduino/
https://docs.arduino.cc/learn/electronics/lcd-displays/
*/
#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int photoPin = A0;
int dark = 20, partiallyDark = 35 , medium = 60, fullyLit = 85;
unsigned long previousMillisSensor = 0; // stores last time sensor was read
const long sensorInterval = 1000;  // interval for reading the photoresistor (1
second)

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

void loop() {
 unsigned long currentMillis = millis();
 if (currentMillis - previousMillisSensor >= sensorInterval) {
```

```
  previousMillisSensor = currentMillis;
  int light = analogRead(photoPin);
  lcd.setCursor(0, 0);
  lcd.print("                "); // Clear the line
  lcd.setCursor(0, 0);
  if(light <= dark){
    lcd.print("dark");
  } else if(light <= partiallyDark){
    lcd.print("partially dark");
  } else if(light <= medium){
    lcd.print("medium");
  } else if(light <= fullyLit){
    lcd.print("fully lit");
  } else {
    lcd.print("brightly lit");
  }
  // lcd.print(analogRead(photoPin));
 }
 lcd.setCursor(0, 1);
 lcd.setCursor(0, 1);
 lcd.print(currentMillis);      // Print the updated millis value
}
```

References

- LCD setup
- Photoresistor setup

Photoresistor Experiments

The first thing I did to test the photoresistor was make it print values. I wanted to see the values under different conditions. Obviously the easiest to test were brightly lit and dark.

- For the brightly lit I just used my iPhone flashlight and saw values around 90, so I also went up a bit for margin.
- For fully lit i just let the photoresistor use the light from the room
- For medium I just put some slight shade over the photoresistor
- For partially dark i put my finger slightly above the photoresistor.
- For dark I just completely covered the photoresistor and saw the values which were around 18 so i went up a little bit for margin.

Conclusion

In conclusion this experiment was a good way to test hardware with code. For example, I used my code to print out the values of the photoresistor to test what values I should set my program to. I think it was also really nice to learn about how the photoresistor just resists the electricity.