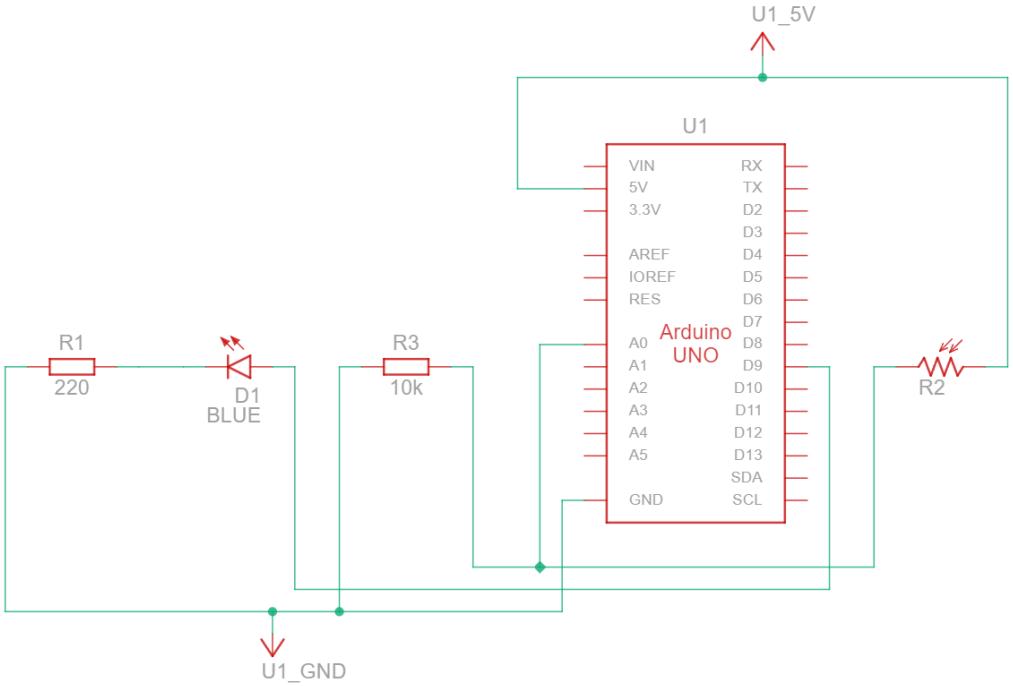
Tutorial 4: Analog Input Digital Output

In this tutorial, we will learn how to use a photoresistor (also known as an LDR - Light Dependent Resistor) to control an LED using an Arduino microcontroller. The LED will turn on when the ambient light level is below a certain threshold detected by the photoresistor. This tutorial is designed for high school students who are new to Arduino and want to explore using a photoresistor as an analog input to control output devices like an LED.

Hardware used:

1. Arduino UNO board
2. LED
3. Resistor
4. Breadboard
5. Photoresistor module

Step 1: Set up the Circuit



1. Connect the 5V pin Arduino to the +ve rail and GND pin to -ve rail on the breadboard.
2. Connect the VCC pin of the photoresistor module to the +ve rail on the breadboard and GND pin to the -ve rail on the breadboard.
3. Connect the A0 pin of the photoresistor module to any analog pin on the Arduino. For example, we use A0.
4. Connect one leg resistor to the -ve leg(cathode) of the LED and another leg of resistor connect to the -ve GND rail.
5. Connect the +ve leg(anode) of LED to any digital pin on Arduino pin.
6. Repeat steps 4 and 5, for the second LED.

Step 2: Write the Arduino code

1. We start by defining constants PHOTOSENSOR_PIN for the analog input pin connected to the photoresistor and LED_PIN for the digital output pin

```
const int photoresistorPin = A0; // Analog input pin for the photoresistor
const int whiteled = 2;           // Digital output pin for the LED
const int redled = 3;            // Digital output pin for the LED
```

2. In the setup () function, we set the whiteled and redled pin as an output using pinMode(). Next, we initialize the serial communication with a baud rate of 9600 using Serial.begin(). This allows us to view the sensor readings and LED state in the Serial Monitor.

```
void setup() {
  pinMode(whiteled, OUTPUT);      // Set the LED pin as OUTPUT
  pinMode(redled, OUTPUT);       // Set the LED pin as OUTPUT
  Serial.begin(9600);            // Initialize Serial Monitor at 9600 baud rate
}
```

3. Inside the loop() function, we read the analog value from the photoresistor using analogRead() and store it in the sensorValue variable.
4. Using an if statement, we check if the sensorValue is below a threshold value (500 in this example) to determine whether the ambient light is below the threshold.
5. If the light is below the threshold, we turn on the whiteled by setting the LED pin to HIGH using digitalWrite(). Otherwise, we turn off the LED by setting the LED pin to LOW.
6. We use Serial.println() to print the sensor value to the Serial Monitor.
7. Finally, we introduce a delay of 500 milliseconds to ensure stability and avoid rapid fluctuations in readings.

```
void loop() {
  int sensorValue = analogRead(photoresistorPin); // Read the analog value from the photoresistor

  // Print the raw analog value to the Serial Monitor (optional)
  Serial.print("Analog Value: ");
  Serial.println(sensorValue);

//Compare the reading if it is above 500
  if (sensorValue > 500) {
    // If the light intensity is above the threshold, turn on the LED
    digitalWrite(whiteled, HIGH);
  } else {
    // Otherwise, turn off the LED
    digitalWrite(whiteled, LOW);
  }


}
```

Step 3: Upload the Code to the Arduino

1. Connect the Arduino board to your computer using a USB cable.
2. Click on the "Upload" button in the Arduino IDE to compile and upload the code to the Arduino board.
3. Once the upload is complete, open the Serial Monitor by clicking on the magnifying glass icon in the Arduino IDE's toolbar or by going to "Tools" -> "Serial Monitor".
4. Set the baud rate in the Serial Monitor to 9600.
5. Observe the sensor values and the LED turning on or off based on the ambient light conditions.

Exercise

Turn on the red LED when the photoresistor detects that it is dark (low reading) and turn on white LED when the photoresistor detects (high reading).