# Sun Tracking Solar Panel using an Arduino

**Components**

* 1 x Arduino Uno
* 1 x Servo motor
* 1 x Solar panel
* 2 x LDR
* 2 x 10k Resistor
* Jumper wires
* 1 x MDF board

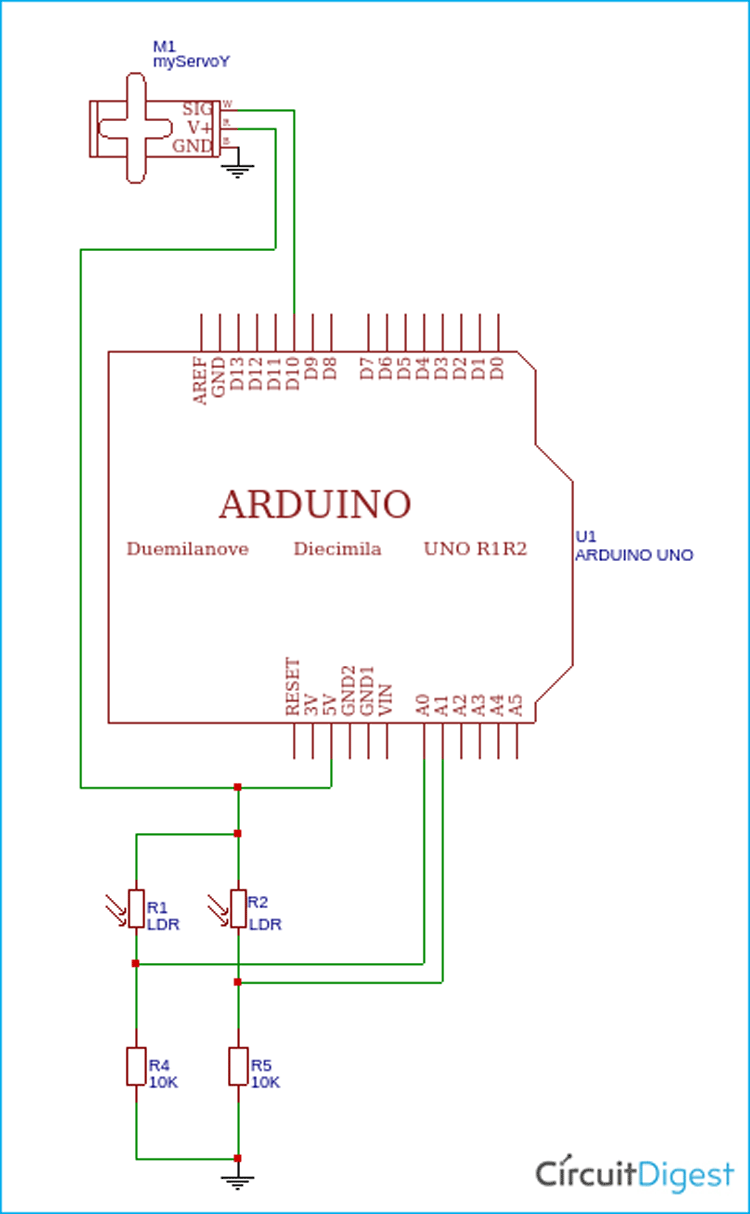
**Overview**

The project involves designing and constructing a sun-tracking solar panel system utilizing an Arduino microcontroller. The primary goal is to enhance the efficiency of solar energy collection by automating the adjustment of the solar panel's angle to continuously face the sun. This tracking mechanism ensures that the panel is positioned optimally to capture maximum sunlight throughout the day.

**Application**

a sun-tracking solar panel system has diverse applications ranging from enhancing residential and commercial solar power systems to supporting educational and research initiatives. By improving the efficiency of solar panels, it plays a crucial role in advancing the adoption and effectiveness of solar energy technologies.

**Circuit Diagram**



**Code**

#include <Servo.h>

Servo myservo; // Create Servo object

#define LDR\_1 A0 // Pin for the first LDR

#define LDR\_2 A1 // Pin for the second LDR

int pos = 90; // Starting position of the servo (centered)

int Resistance = 20; // Threshold for LDR difference to detect light imbalance

void setup() {

myservo.attach(10); // Attach the servo to pin 4

pinMode(LDR\_1, INPUT); // Set LDR1 pin as input

pinMode(LDR\_2, INPUT); // Set LDR2 pin as input

myservo.write(pos); // Initialize servo position to the middle (90 degrees)

Serial.begin(9600); // Initialize serial communication at 9600 baud

delay(1000); // Wait for a moment for initialization

}

void loop() {

int value\_1 = analogRead(LDR\_1); // Read the value from LDR1

int value\_2 = analogRead(LDR\_2); // Read the value from LDR2

// Print LDR values for debugging

Serial.print("Value1: ");

Serial.print(value\_1);

Serial.print("\tValue\_2: ");

Serial.println(value\_2);

// Compare LDR values and move the servo based on the difference

if(abs(value\_1 - value\_2) > Resistance) {

if(value\_1 > value\_2) {

pos++; // Move servo right (increase position)

} else {

pos--; // Move servo left (decrease position)

}

}

// Ensure the servo position stays within the valid range (0 to 180 degrees)

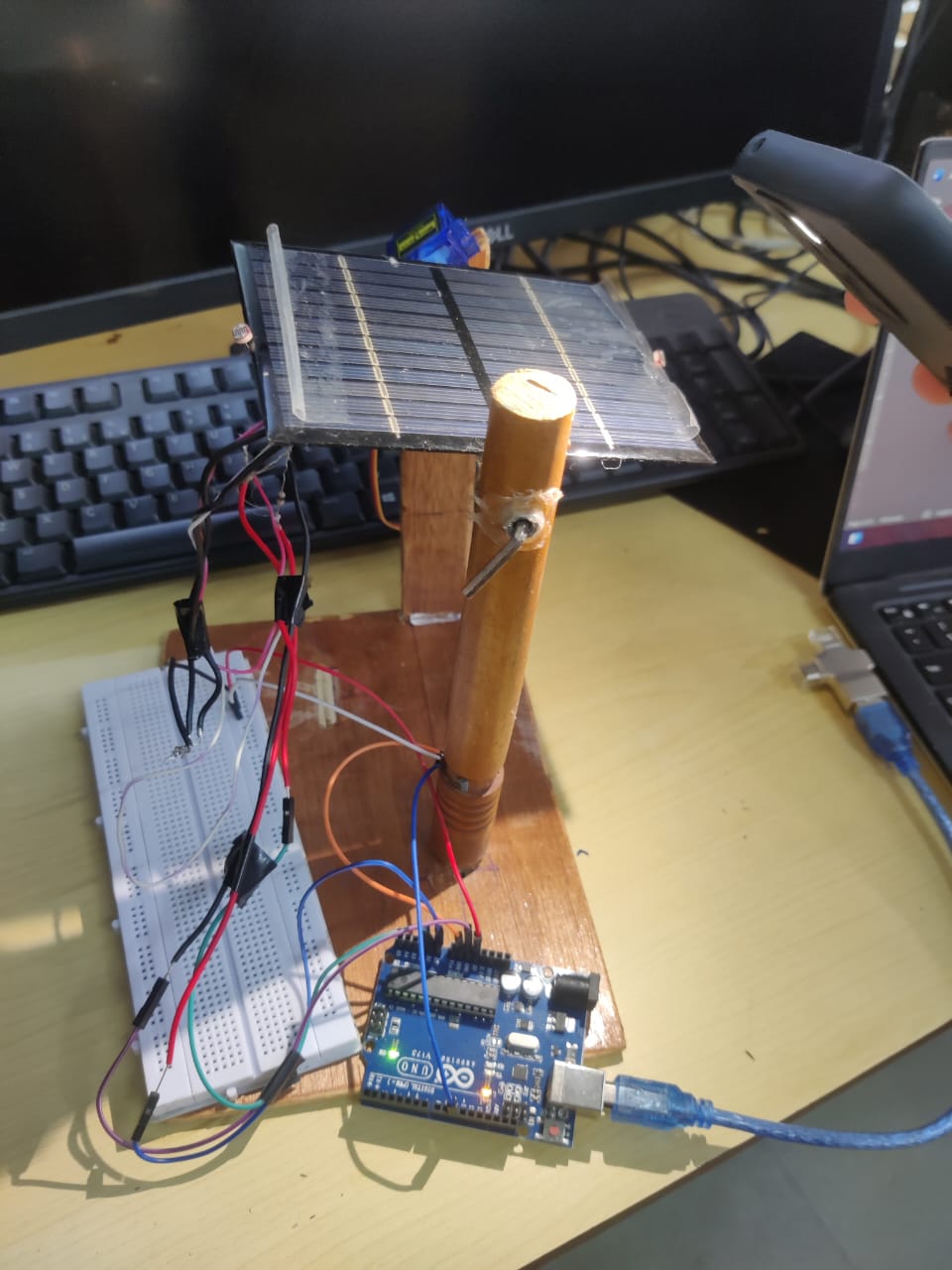
pos = constrain(pos, 0, 180);

myservo.write(pos); // Move the servo to the new position

delay(50); // Small delay for smooth operation

}

**Result**



The task was successfully tested and performed as intended.