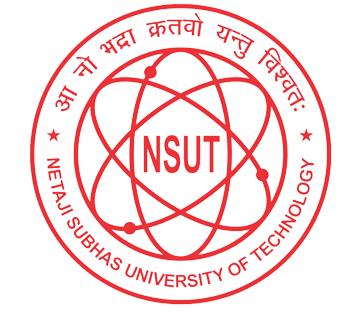
## 

IOT Assignment-3



Name – Akhil Dubey

Roll – 2020UCO1673

BRANCH -COE-3

# Light Sensor (Photoresistor) With Arduino in Tinkercad

### COMPONENTS REQUIRED:

Breadboard, Arduino, LED, 2 Resistors, Photoresistor,

## **Step 1: Build the Circuit**

Identify the photoresistor, LED, resistors, and wires connected to the Arduino in the Tinkercad Circuits workplane.

Drag an Arduino Uno and breadboard from the components panel to the workplane, next to the existing circuit.

Connect breadboard power (+) and ground (-) rails to Arduino 5V and ground (GND), respectively, by clicking to create wires.

Extend power and ground rails to their respective buses on the opposite edge of the breadboard (optional for this circuit but good common practice).

Plug the LED into two different breadboard rows so that the cathode (negative, shorter leg) connects to one leg of a resistor (anywhere from 100-1K ohms is fine). The resistor can go in either orientation because resistors aren't polarized, unlike LEDs, which must be connected in a certain way to function.

Connect other resistor leg to ground.

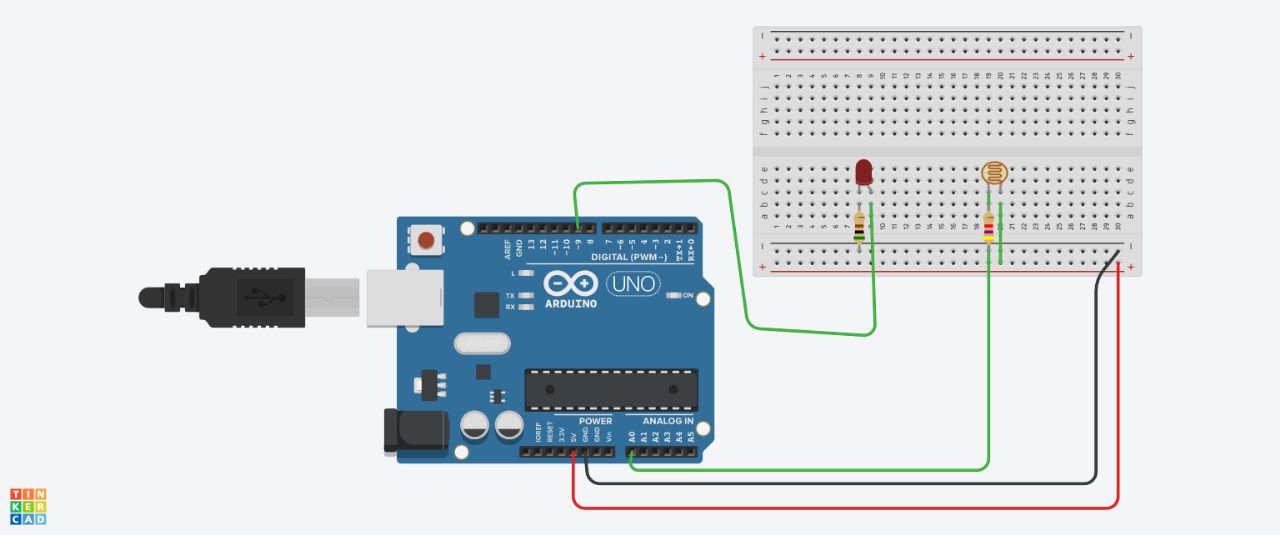
Wire up the LED anode (positive, longer leg) to Arduino pin 9.

Drag a photoresistor from the components panel to your breadboard, so its legs plug into two different rows.

Click to create a wire connecting one photoresistor leg to power.

Connect the the other leg to Arduino analog pin A0.

Drag a resistor from the components panel to connect the photoresistor leg connected to A0 with ground, and adjust its value to 4.7k ohms.



## **Step 2: Code**

## 

int sensorValue = 0;

void setup()

{

pinMode(A0, INPUT);

pinMode(9, OUTPUT);

Serial.begin(9600);

}

void loop()

{

// read the value from the sensor

sensorValue = analogRead(A0);

// print the sensor reading so you know its range

Serial.println(sensorValue);

// map the sensor reading to a range for the LED

analogWrite(9, map(sensorValue, 0, 1023, 0, 255));

delay(100); // Wait for 100 millisecond(s)

}

## INTENSITY VALUES ON SLIDING LEFT TO RIGHT:

6

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

26

389

389

389

652

700

787

797

814

861

866

898

903

919

920

923

923

923

923

923

923

923

923

923

923

923

# PIR Motion Sensor With Arduino in Tinkercad

### COMPONENTS REQUIRED:

Breadboard, Arduino, PIR motion sensor, LED, resistor

## **Step 1: Build the Circuit**

Identify the PIR motion sensor, LED, resistor, and wires connected to the Arduino.

Drag an Arduino Uno and breadboard from the components panel to the workplane.

Connect breadboard power (+) and ground (-) rails to Arduino 5V and ground (GND), respectively, by clicking to create wires.

Extend power and ground rails to their respective buses on the opposite edge of the breadboard by creating a red wire between both power buses and a black wire between both ground buses.

Plug the LED into two different breadboard rows so that the cathode (negative, shorter leg) connects to one leg of a resistor (anywhere from 100-1K ohms is fine). The resistor can go in either orientation because resistors aren't polarized, unlike LEDs, which must be connected in a certain way to function.

Connect other resistor leg to ground.

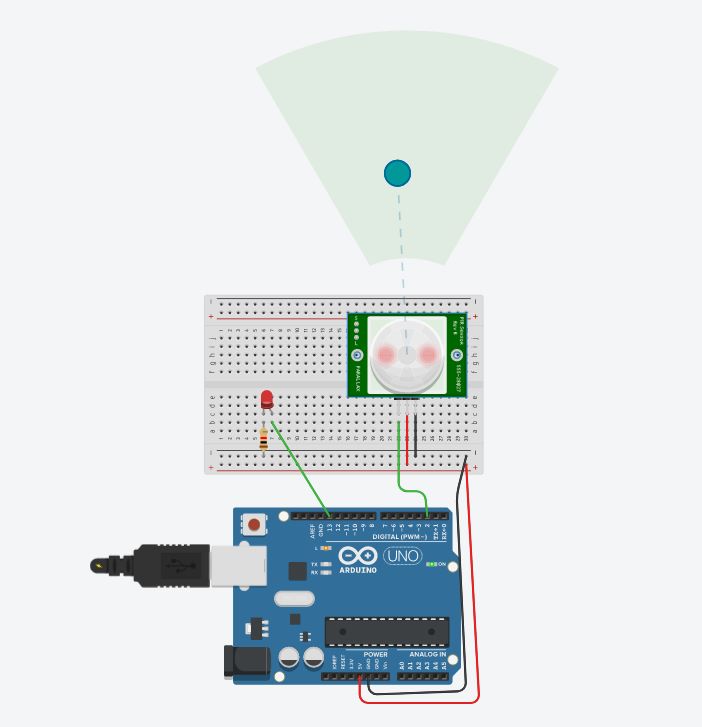
Wire up the LED anode (positive, longer leg) to Arduino pin 13.

Drag a PIR motion sensor from the components panel to your breadboard, so its legs plug into three different rows.

Click to create a wire connecting the rightmost leg to power.

Connect the center leg to ground.

Create a wire connecting the leftmost leg to Arduino analog pin A0.



## **Step 2: Code**

int sensorState = 0;

void setup()

{

pinMode(2, INPUT);

pinMode(13, OUTPUT);

Serial.begin(9600);

}

void loop()

{

// read the state of the sensor/digital input

sensorState = digitalRead(2);

// check if sensor pin is HIGH. if it is, set the

// LED on.

if (sensorState == HIGH) {

digitalWrite(13, HIGH);

Serial.println("Sensor activated!");

} else {

digitalWrite(13, LOW);

}

delay(10); // Delay a little bit to improve simulation performance

}

During the experiment of interfacing an LDR (Light Dependent Resistor) sensor with an Arduino microcontroller, you would have gained the following concepts, experimental, and programming skills:

Concepts:

Understanding of LDR sensor and how it works as a resistive sensor. Understanding of Analog to Digital Conversion (ADC) and how it works. Understanding of how to interface sensors with the microcontroller. Understanding of electrical circuit concepts such as voltage, current, and resistance.

Experimental skills:

Ability to connect and wire the LDR sensor to the Arduino board.

Ability to measure and read the analog voltage values from the LDR sensor. Ability to convert the analog voltage values to digital values using the ADC of the microcontroller.

Programming skills:

Ability to write code in C/C++ to control the Arduino board.

Ability to read analog values from the LDR sensor using the analogRead() function.

Ability to process the sensor data and perform operations such as conditionals, loops, etc.

Ability to communicate with the sensor and interact with the real-world environment using the microcontroller.

Reasons for errors in the experiment could be:

1. Sensor sensitivity: The LDR sensor's sensitivity to light could vary depending on the type and manufacturer of the sensor, which could affect the accuracy of the results.
2. Electrical noise: Interference from other electrical signals or devices in the environment could cause fluctuations in the analog readings from the sensor, leading to errors in the results.
3. ADC resolution: The ADC resolution of the microcontroller determines the number of bits of accuracy in the digital values obtained from the analog readings. A lower ADC resolution could result in lower accuracy of the results.
4. Calibration: The LDR sensor may need to be calibrated to obtain accurate results, especially if it is being used in a specific light environment.
5. Software errors: Errors in the code or logic used to process the sensor data could also impact the accuracy of the results. For example, using incorrect formulas, scaling factors, or conversions could cause errors in the output. It's important to consider these sources of errors and minimize them as much as possible to obtain accurate results from the experiment.

## Maximum value of LDR sensor= 923

## Minimum value of LDR sensor= 26

5 real time applications of:

# LDR Sensor

1. Street Lighting: LDR sensors can be used in street lighting systems to turn on/off the lights based on the ambient light levels.
2. Automated Room Lighting: LDR sensors can be used in smart home systems to adjust the brightness of room lights according to the natural light levels.
3. Camera Exposure Control: LDR sensors are often used in cameras to measure the ambient light levels and adjust the camera's exposure accordingly.
4. Security Systems: LDR sensors can be used in security systems to detect movements in low-light conditions and trigger an alarm.
5. Agricultural Irrigation: LDR sensors can be used in agriculture to measure the light intensity and control the amount of water supplied to crops, reducing water waste.

# PIR Motion Sensor

1. Security Systems: PIR motion sensors can be used in security systems to detect movement within a specified area and trigger an alarm.
2. Automated Lighting: PIR motion sensors can be used in smart homes to turn lights on and off as people move in and out of rooms.
3. Motion-Activated Displays: PIR motion sensors can be used to activate displays in retail stores and museums, providing an interactive experience for customers.
4. Energy Efficiency: PIR motion sensors can be used to control the lighting and heating in a room based on occupancy, reducing energy waste.
5. Automated Door Openers: PIR motion sensors can be used to open and close doors automatically, improving accessibility and convenience in various settings such as hospitals, offices, and public buildings.