# Embedded System Lab
# Final Project Report

## Let There Be Light

(Bus Stop Brightness Sensor)


By

| | |
|---|---|
| Chotpisit Adunsehawat | 6531313221 |
| Warissara Booranamaitree | 6532152721 |
| Winithra Manolertthewan | 6530375021 |
| Poopha Suwananek | 6532141821 |



This report is for the course 2110366 (2022/2) - Embedded System Laboratory
Computer Engineering - Chulalongkorn University

# Project

**Project name:** Let There Be Light

**Description:**

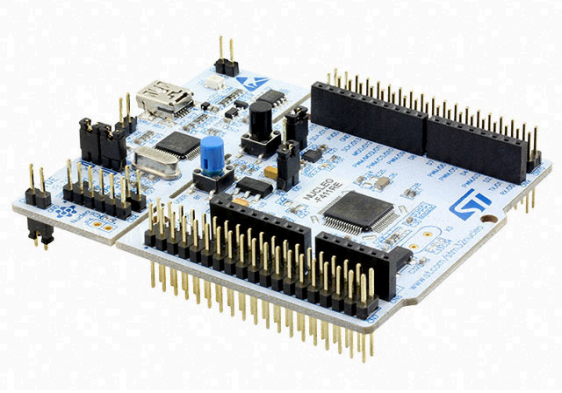
This project is based on Chadchart's Policy, "Safe, well-lit bus stops with information on bus routes", https://www.chadchart.com/policy/6214b2f9204d4c4f8ab8c820/

A device for measuring the luminosity of bus stop's lights and signs using a distance sensor and brightness sensor.

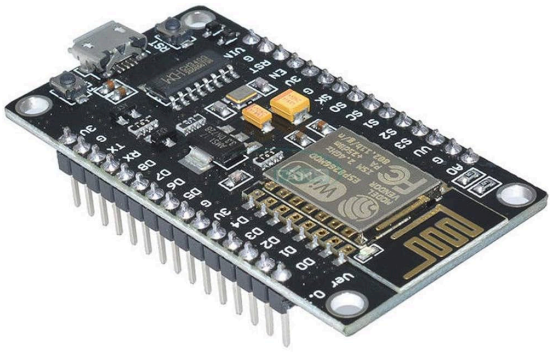
**GitHub Link:** https://github.com/nacnano/embbed-lab-final-project

**Equipment:**

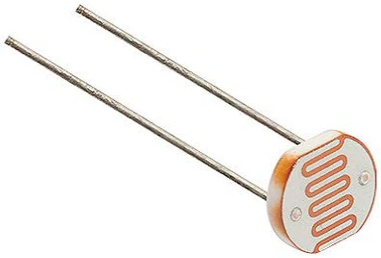
1. STM32 Board
2. ESP8266 NodeMCU
3. Light Dependent Resistor (LDR)
4. Ultrasonic Sensor
5. Breadboard
6. Cable



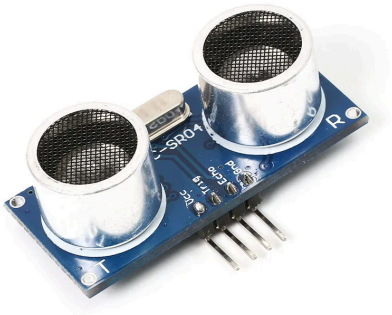
STM32 board



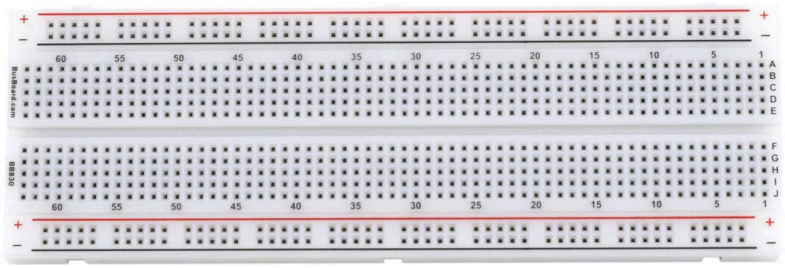
nodeMCU



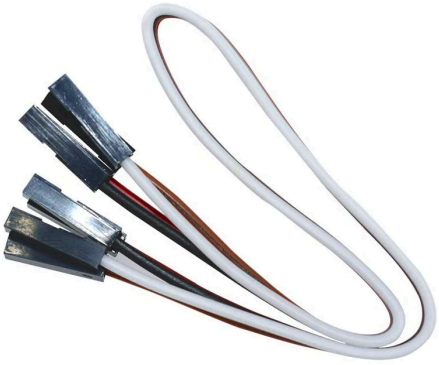
LDR



Ultrasonic Sensor



Breadboard



Cable

# Responsibilities

## Chotpisit Adunsehawat

Role: Web Design and Development, Database Engineering

Web App
- Design UX/UI of the Web App
- Create the Web App using NextJS
- Fetch data from the database
- Responsive design for any display size

Database
- Init and set firebase config
- Setup to store data from sensors

NodeMCU
- Debug minor database connection issues

STM32
- Research ways for Ultrasonic Sensor implementation

## Warissara Booranamaitree

Role: Embedded System Development

NodeMCU
- Connect NodeMCU to Wi-Fi
- Receive data from STM32 through UART
- Connect NodeMCU to the Firebase database
- Send data to the Firebase database

STM32
- Debug minor UART connection issues

Supplier
- Preparing equipment

## Winithra Manolertthewan

Role: Embedded System Development

STM32

- Connect circuitry between STM32 and NodeMCU
- Programmed STM32 to transmit data to NodeMCU through UART
- Debug major UART connection issues

NodeMCU

- Debug minor Wi-Fi connection issues

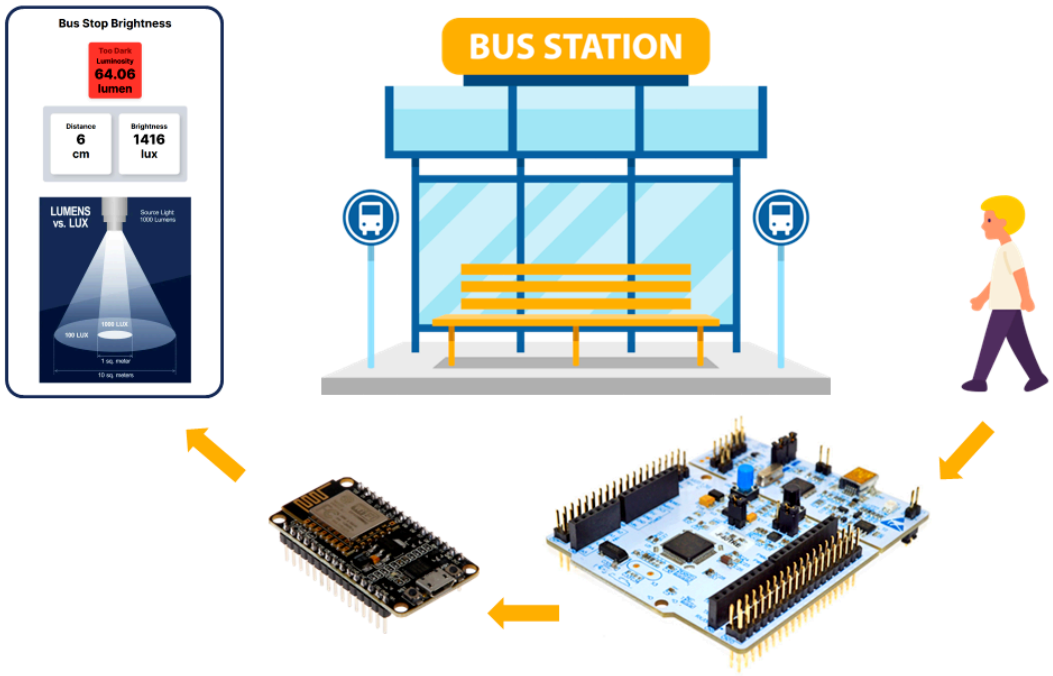
## Poopha Suwananek

Role: Embedded sensors and circuit design

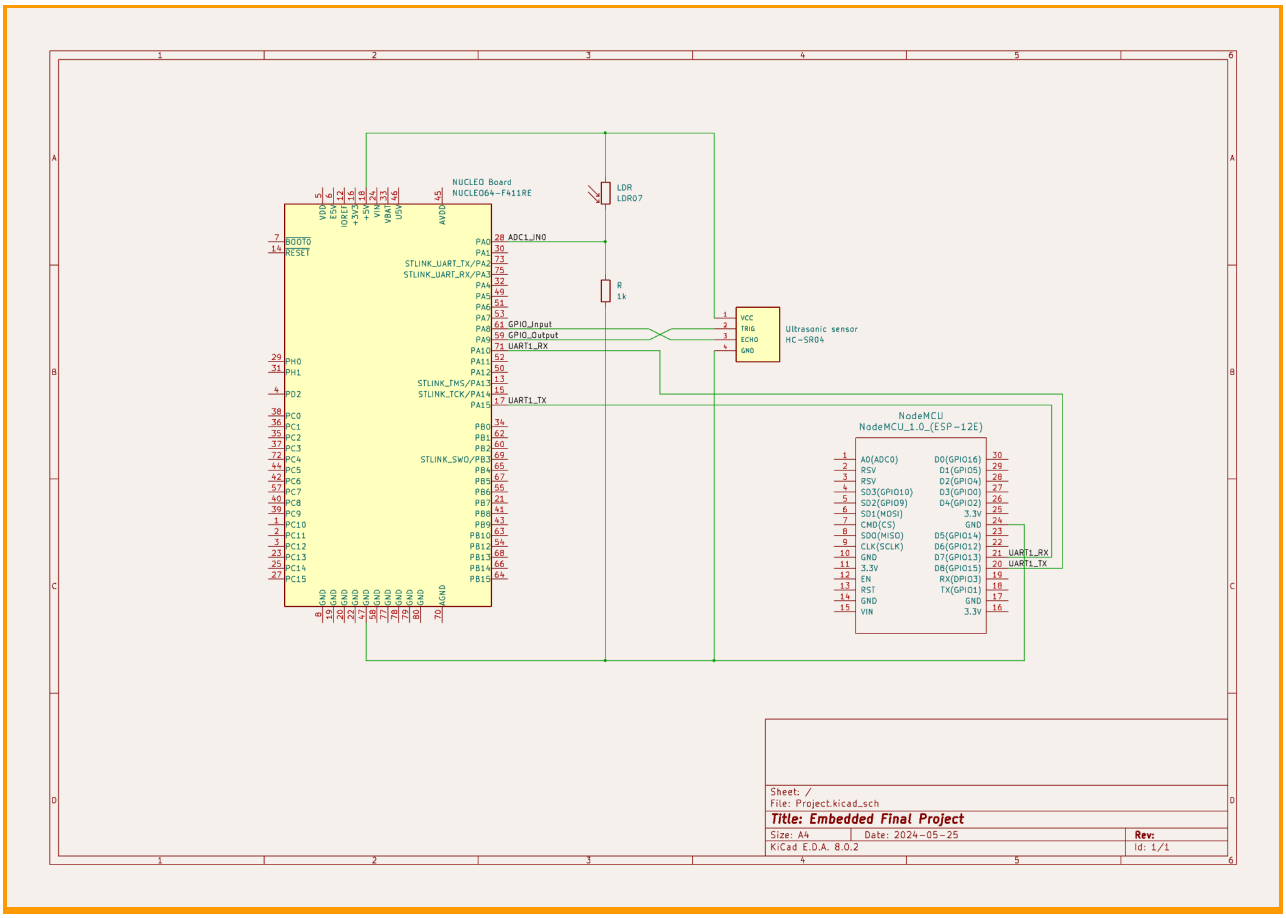
STM32

- Created the circuit for LDR for measuring light intensity via changes in the circuit resistance and voltage
- Connects hc-sr04 ultrasonic sensor to the STM32 board
- Gather the data on the correlation between lux and LDR resistance
- Calculate LDR resistance to lux conversion equation via logarithmic transformation
- Implemented a method for STM32 to convert the time difference data from the hc-sr04 ultrasonic sensor into the distance in centimeters
- Implements a method for STM32 to convert ADC value from LDR to voltage and LDR resistance concerning the circuit designed

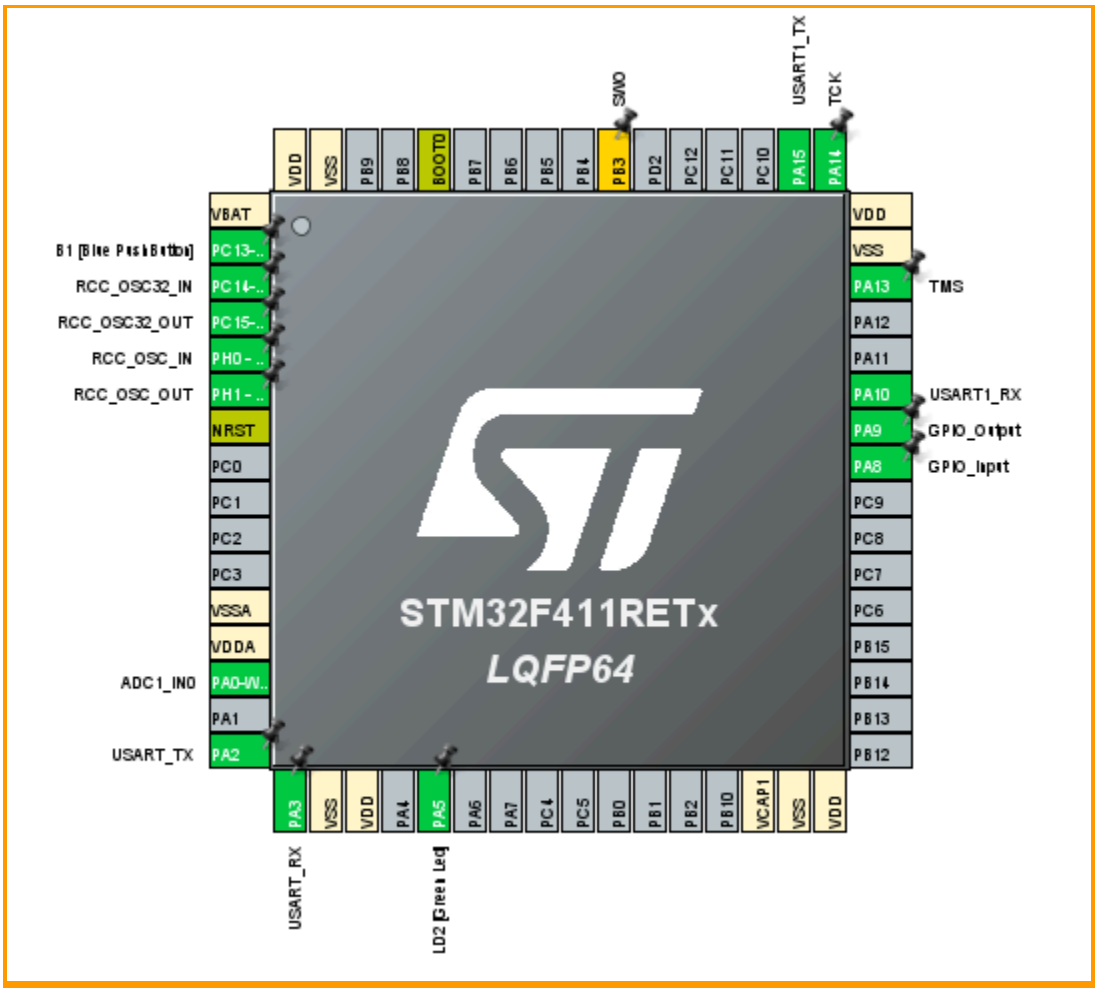
# Overview


Bus Stop Brightness
Too Dark
Luminosity
64.06
lumen
Distance
6
cm
Brightness
1416
lux
LUMENS
vs. LUX
Source Light
1000 Lumens
100 LUX
1000 LUX
1 sq. meter
10 sq. meters
BUS STATION

## Architecture



IOC


STM32F411RETx
LQFP64
VDD
VSS
PB9
PB8
BOOT0
PB7
PB6
PB5
PB4
PB3
PD2
PC12
PC11
PC10
PA15
PA14
SWO
USART1_TX
TCK
VBAT
PC13-..
PC14-..
PC15-..
PH0-..
PH1-..
NRST
PC0
PC1
PC2
PC3
VSSA
VDDA
PA0-W..
PA1
PA2
B1 [Blue Push Button]
RCC_OSC32_IN
RCC_OSC32_OUT
RCC_OSC_IN
RCC_OSC_OUT
ADC1_IN0
USART_TX
VDD
VSS
PA13
PA12
PA11
PA10
PA9
PA8
PC9
PC8
PC7
PC6
PB15
PB14
PB13
PB12
TMS
USART1_RX
GPIO_Output
GPIO_Input
PA3
VSS
VDD
PA4
PA5
PA6
PA7
PC4
PC5
PB0
PB1
PB2
PB10
VCAP1
VSS
VDD
USART_RX
LD2 [Green Led]

Code for sensors and transferring data to NodeMCU

```
/* USER CODE BEGIN PV */
// For HC-SR04 sensor
#define TRIG_PIN GPIO_PIN_9 //D8
#define TRIG_PORT GPIOA
#define ECHO_PIN GPIO_PIN_8 //D7
#define ECHO_PORT GPIOA
// Use Timer1 as counter with HCLK of 72 and prescaler of 72-1.
// For LDR sensor
// Use adc1 with the resolution of 12 bits at port A0.
// Both uses 5V, Resistor used 1,000 ohms

uint32_t pMillis;
uint32_t Value1 = 0;
uint32_t Value2 = 0;
uint16_t Distance  = 0;  // cm
uint16_t ldr = 0;
double voltage_used = 5.0;
double max_adc_decimal = 4095.0;
double resistor_used = 1000.0;
char buf[256];
char confirmation[1];
/* USER CODE END PV */
```

```
/* USER CODE BEGIN 0 */
uint32_t read_ADC_value(void) {
    HAL_ADC_Start(&hadc1); // Start the ADC conversion
    HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY); // Wait for the conversion to complete
    uint32_t adc_value = HAL_ADC_GetValue(&hadc1); // Get the ADC value
    HAL_ADC_Stop(&hadc1); // Stop the ADC conversion
    return adc_value;
}
/* USER CODE END 0 */
```

```c
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_USART2_UART_Init();
  MX_TIM1_Init();
  MX_ADC1_Init();
  MX_USART1_UART_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start(&htim1);
  HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);  // pull the TRIG pin low
  /* USER CODE END 2 */
```

```c
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    // Debug light
    HAL_Delay(500);
    if (HAL_UART_Receive(&huart1, confirmation, 1, 10) != HAL_OK) {
      HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
      continue;
    }
  // Distance sensor
    // Trigger 10 micro second pulse
    HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_SET);  // pull the TRIG pin HIGH
    __HAL_TIM_SET_COUNTER(&htim1, 0);
    while (__HAL_TIM_GET_COUNTER (&htim1) < 10);  // wait for 10 us
    HAL_GPIO_WritePin(TRIG_PORT, TRIG_PIN, GPIO_PIN_RESET);  // pull the TRIG pin low

    // Detect High Output
    pMillis = HAL_GetTick(); // used this to avoid infinite while loop  (for timeout)
    // wait for the echo pin to go high
    while (!(HAL_GPIO_ReadPin (ECHO_PORT, ECHO_PIN)) && pMillis + 10 >  HAL_GetTick());
    Value1 = __HAL_TIM_GET_COUNTER (&htim1);

    // Detect Low Output
    pMillis = HAL_GetTick(); // used this to avoid infinite while loop (for timeout)
    // wait for the echo pin to go low
    while ((HAL_GPIO_ReadPin (ECHO_PORT, ECHO_PIN)) && pMillis + 50 > HAL_GetTick());
    Value2 = __HAL_TIM_GET_COUNTER (&htim1);

    // Distance limit = [2, 200] cm
    Distance = (Value2-Value1)* 0.034/2;
//    UART2 Debug
//    sprintf(buf, "dis = %d\r\n", Distance);
//    HAL_UART_Transmit(&huart2, buf, strlen(buf), 1000);
```

```c
//    Light sensor
    int adc_value = read_ADC_value(); // Read the ADC value
    double Rvoltage = ((double)adc_value / max_adc_decimal) * voltage_used; // Convert ADC value to Resistor voltage
    double LDRvoltage = voltage_used - Rvoltage;
    double LDRresistance = LDRvoltage / Rvoltage * resistor_used;
//    sprintf(buf, "adc = %d   RV = %d   LDRV = %d   RLDR = %d\r\n",adc_value, (int)Rvoltage, (int)LDRvoltage, (int)LDRresistance);
//    HAL_UART_Transmit(&huart2, buf, strlen(buf), 10);
//    sprintf(buf, "%dx", Distance);
//    HAL_UART_Transmit(&huart1, buf, strlen(buf), 10);
//    sprintf(buf, "%dx", adc_value);
//    HAL_UART_Transmit(&huart1, buf, strlen(buf), 10);
    // Transimit to node
    sprintf(buf, "%d,%dx", Distance, (int)LDRresistance);
    HAL_UART_Transmit(&huart1, buf, strlen(buf), 10);
//    UART2 Debug
//    sprintf(buf, "adc = %d\r\n", adc_value);
//    HAL_UART_Transmit(&huart2, buf, strlen(buf), 1000);

    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
    HAL_Delay(500);
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

Code for getting sensor data from STM32 and saving them into the database

```
void loop() {      warboo, last week • Connect to Firebase
  String data;

  if (comm.available()) {

    Serial.println("Connecting Comm ...");

    data = comm.readStringUntil('x');
    Serial.println(data);
    comm.write('1');

    yield();

    if (Firebase.ready()) {
      Serial.println("In if");
      Serial.printf("Set json... %s\n", Firebase.RTDB.setString(&fbdo, "/sensor", data) ? "ok" : fbdo.errorReason().c_str());
    }
    else {
      Serial.println("In else");
    }
  }
  else {
    Serial.println("Not Com available");
  }

  delay(1000);
}
```

Code for fetching database data from Firebase to the web

```
const firebaseConfig = {
  apiKey: process.env.NEXT_PUBLIC_FIREBASE_API_KEY,
  authDomain: process.env.NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN,
  databaseURL: process.env.NEXT_PUBLIC_FIREBASE_DATABASE_URL,
  projectId: process.env.NEXT_PUBLIC_FIREBASE_PROJECT_ID,
  storageBucket: process.env.NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET,
  messagingSenderId: process.env.NEXT_PUBLIC_FIREBASE_MESSAGING_SENDER_ID,
  appId: process.env.NEXT_PUBLIC_FIREBASE_APP_ID,
};

const Home = () => {
  const [brightness, setBrightness] = useState<number>();
  const [distance, setDistance] = useState<number>();

  const app = initializeApp(firebaseConfig);
  const database = getDatabase(app);

  useEffect(() => {
    const Ref = ref(database, "sensor");

    const unsub1 = onValue(Ref, (snapshot) => {
      const data = snapshot.val() as String;
      console.log(data);

      const [distanceData, resistanceData] = data
        ? data.split(",")
        : ["-1", "-1"];

      const brightnessData = calculateLux(parseInt(resistanceData));

      setDistance(parseInt(distanceData));
      setBrightness(brightnessData);
    });
    return () => {
      unsub1();
    };
  }, []);
```

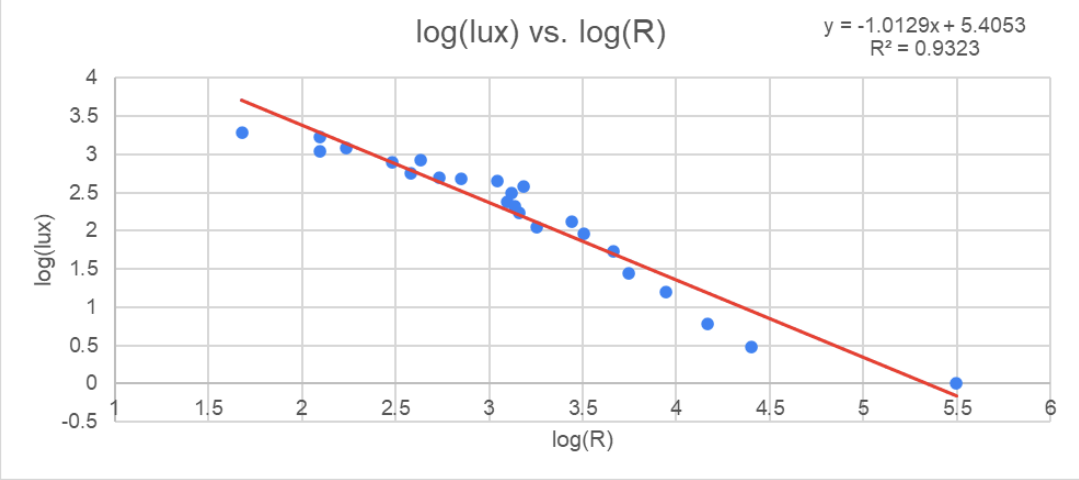
# Illuminace(Lux) & LDR Resistance correlation

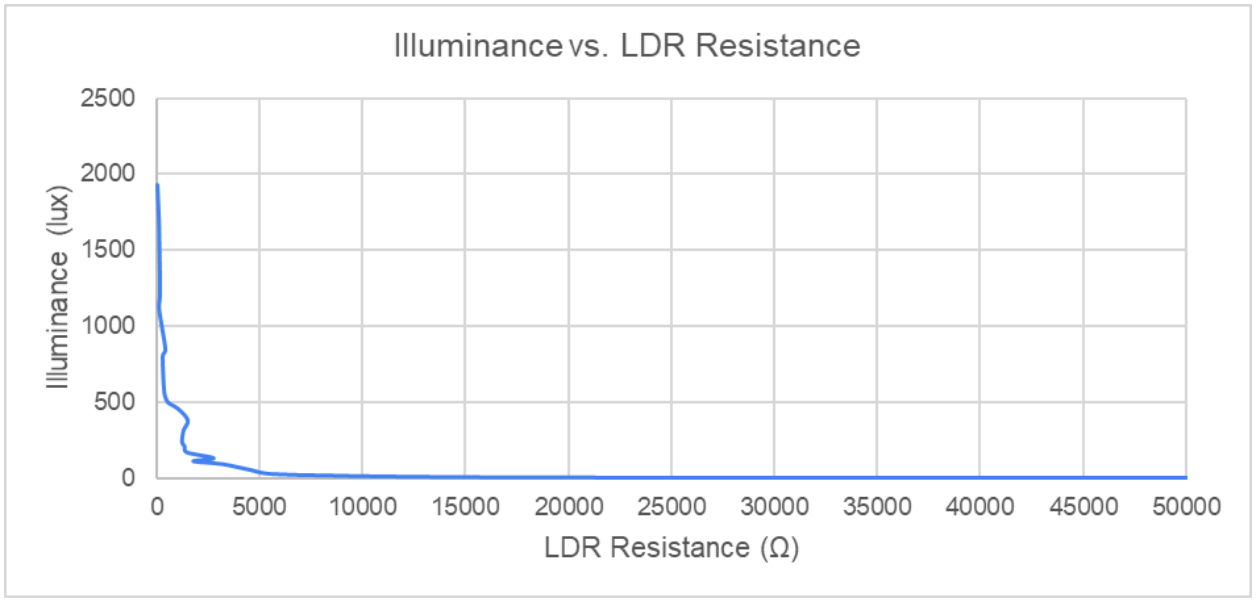
Sensors measure the brightness and distance of the target. Those data can be calculated to luminosity using the formula

$$lux = 254267.4432 \times resistance^{-1.0128565}$$

This formula is calculated by reading LDR Resistance along with lux from the phone's lux meter application in different environments. Then plot a line graph to see its relation.

| LDR Resistance (Ω) | Illuminance (lux) | log(R) | log(lux) |
|---|---|---|---|
| 314000 | 1 | 5.496929648 | 0 |
| 25082.80255 | 3 | 4.399376059 | 0.477121255 |
| 14689.65517 | 6 | 4.167011601 | 0.77815125 |
| 8750 | 16 | 3.942008053 | 1.204119983 |
| 5531.100478 | 28 | 3.742811548 | 1.447158031 |
| 4579.019074 | 53 | 3.660772453 | 1.72427587 |
| 3200 | 92 | 3.505149978 | 1.963787827 |
| 1789.509537 | 111 | 3.252734017 | 2.045322979 |
| 2781.163435 | 131 | 3.444226511 | 2.117271296 |
| 1455.035971 | 170 | 3.16287373 | 2.230448921 |
| 1362.954414 | 210 | 3.134481331 | 2.322219295 |
| 1243.835616 | 238 | 3.094762988 | 2.376576957 |
| 1310.948081 | 313 | 3.117585492 | 2.495544338 |
| 1523.10536 | 380 | 3.182729947 | 2.579783597 |
| 1102.156057 | 450 | 3.042243092 | 2.653212514 |
| 706.25 | 485 | 2.848958461 | 2.685741739 |
| 542.3728814 | 502 | 2.734297967 | 2.700703717 |
| 380.182002 | 560 | 2.579991553 | 2.748188027 |
| 301.6528926 | 802 | 2.479507494 | 2.904174368 |
| 430.3178484 | 850 | 2.63378936 | 2.929418926 |
| 124.0735657 | 1117 | 2.093679264 | 3.048053173 |
| 173.0163277 | 1216 | 2.23808709 | 3.084933575 |
| 124.691019 | 1665 | 2.095835174 | 3.221414238 |
| 47.85056295 | 1932 | 1.679887051 | 3.286007122 |




Illuminance vs. LDR Resistance
Illuminance (lux)
2500
2000
1500
1000
500
0
0
5000
10000
15000
20000
25000
30000
35000
40000
45000
50000
LDR Resistance (Ω)

# Website


**Website link:** [Bus Stop Brightness (embbed-lab-final-project.vercel.app)](embbed-lab-final-project.vercel.app)

**Display:**

- Distance in cm
- Brightness in Lux
- Luminosity in Lumen
- Brightness Level: Too dark, too bright, and OK using this criteria.

```
export const threshold = {
  dark: 1000,
  bright: 2000,
};
```

# Pictures