KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG - UIT

BÁO CÁO THỰC HÀNH

Lab 2

CÔNG NGHỆ INTERNET OF THINGS HIỆN ĐẠI

Arduino và một số loại cảm biến

GVHD: **Phan Trung Phát**

Lớp: **NT532.O21.MMCL.2**

Họ và tên	MSSV
Tống Võ Anh Thuận	21522652
Trịnh Vinh Đại	21521915
Lê Huỳnh Quanh Vũ	21522797

ĐÁNH GIÁ KHÁC (*):

Nội dung	Kết quả
Tổng thời gian thực hiện bài thực hành trung bình (1)	8 days
Link Video thực hiện (2) *(nếu có)*	Lab02_Code_and_Video_Demo
Ý kiến (3) *(nếu có)* + Khó khăn + Đề xuất ...	
Điểm tự đánh giá (4)	10/10

(*): phần (1) và (4) bắt buộc thực hiện.

Phần bên dưới là báo cáo chi tiết của nhóm/cá nhân thực hiện.

WARNING

LƯU HÀNH NỘI BỘ

Câu hỏi 1. Xây dựng ứng dụng lùi xe, kịch bản gồm có 1 đèn LED và 1 cảm biến khoảng cách siêu âm. Cảm biến được gắn cố định và khi xe lùi về càng gần cảm biến thì đèn càng chớt tắt nhanh hơn để báo hiệu. Chia mức độ cảnh báo thành 5 mức từ chậm đến nhanh.

1. Setup:

Components used:

- 1 HC-SR04 sensor.
- 1 Arduino Uno board.
- 1 LED.
- 4 Jumper wires (Male-to-Female).
- 5 Jumper wires (Male-to-Male).

Wiring connection:

- The HC-SR501 sensor has four pins: VCC, GND Trigger and Echo.
  - VCC: Power supply to the sensor, typically +5V.
  - GND: This pin is grounded.
  - Trigger: Initiates the measurement by sending ultrasonic waves.
  - Echo: Receive ultrasonic waves reflected to the sensor.
- The sensor's Trigger pin is attached to Digital port 13, the sensor's Echo pin is attached to Digital 12 port, and the LED anode is attached to Digital 9 port.

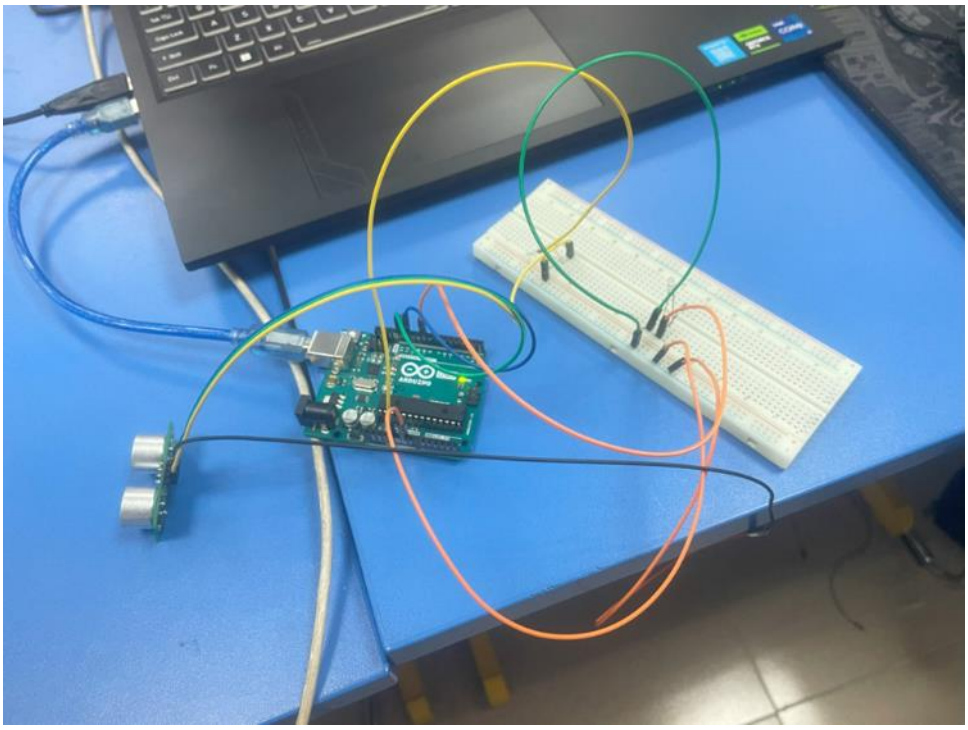

Figure 1. Settup for Question 1

2. Code explanation:

```
12  void loop() {
13    digitalWrite(triggerPin, LOW);
14    delayMicroseconds(2);
15    digitalWrite(triggerPin, HIGH);
16    delayMicroseconds(10);
17    digitalWrite(triggerPin, LOW);
18
19    duration = pulseIn(echoPin, HIGH);
20
21    // Tính toán khoảng cách dựa trên thời gian
22    distance = duration * 0.034 / 2;
23
24    // Hiển thị khoảng cách qua Serial Monitor
25    Serial.print("Distance: ");
26    Serial.print(distance);
27    Serial.println(" cm");
```

This code is simple. The first sends an ultrasonic pulse to activate the sensor. Then read the time from the sensor, calculate distance based on time. Print the results to Serial Monitor.

```
28
29    // Điều khiển đèn LED dựa trên khoảng cách
30    if (distance >= 0 && distance < 20) {
31
32      digitalWrite(ledPin, HIGH);
33      delay(100);
34      digitalWrite(ledPin, LOW);
35      delay(100);
36    } else if (distance >= 20 && distance < 40) {
37
38      digitalWrite(ledPin, HIGH);
39      delay(300);
40      digitalWrite(ledPin, LOW);
41      delay(300);
42    } else if (distance >= 40 && distance < 60) {
43
44      digitalWrite(ledPin, HIGH);
45      delay(600);
46      digitalWrite(ledPin, LOW);
47      delay(600);
48    } else if (distance >= 60 && distance < 80) {
49
50      digitalWrite(ledPin, HIGH);
51      delay(900);
52      digitalWrite(ledPin, LOW);
53      delay(900);
54    } else if (distance >= 80) {
55
56      digitalWrite(ledPin, HIGH);
57      delay(1200);
58      digitalWrite(ledPin, LOW);
59      delay(1200);
60    }
61  }
```

If the distance is from 0-20 cm, the light will blink continuously with a delay of 100ms.

If the distance is 20-40 cm, the light will blink continuously with a delay of 300ms.

If the distance is 40-60 cm, the light will blink continuously with a delay of 600ms.

If the distance is 60-80 cm, the light will blink continuously with a delay of 900ms.

If the distance is more than 80cm, the light will blink continuously with a delay of 1200ms.

Câu hỏi 2. Xây dựng mô hình gồm có 1 LED 7 đoạn và 1 cảm biến khoảng cách siêu âm (Ultrasonic Distance Sensor). Dựa vào khoảng cách tiếp cận gần hay xa của vật thể thì LED 7 đoạn sẽ thể hiện bằng các con số. Nếu khoảng cách nằm ngoài vùng phủ của cảm biến hoặc bị sai thì LED 7 đoạn hiển thị số 0.

1. Setup:

Components used:

- 1 HC-SR04 sensor.
- 1 Arduino Uno board.
- 1 7-segment LED.
- 4 Jumper wires (Male-to-Female).
- 12 Jumper wires (Male-to-Male).

Wiring connection

- The HC-SR501 sensor has four pins: VCC, GND Trigger and Echo.
    - VCC: Power supply to the sensor, typically +5V.
    - GND: This pin is grounded.
    - Trigger: Initiates the measurement by sending ultrasonic waves.
    - Echo: Receive ultrasonic waves reflected to the sensor.
- The 7-segment LED has 10 pins, of which 8 pins are connected to the LED (A, B, C, D, E, F, G, and DP). In this article, the DP pin will not be used. The remaining 2 pins are VCC and GND pins.

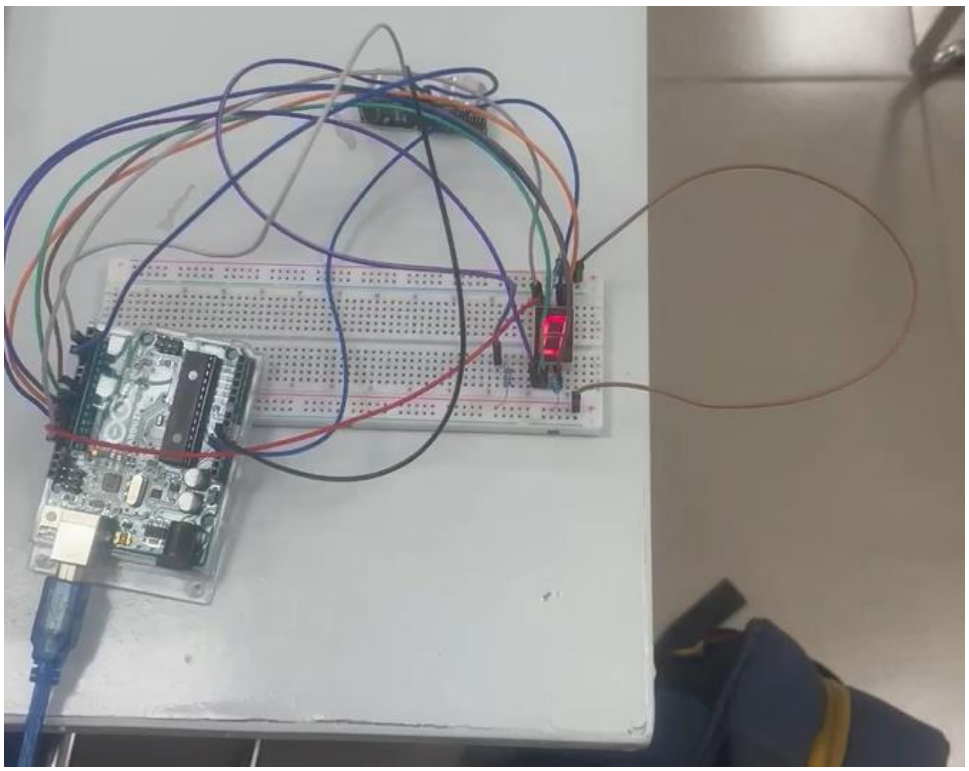

Figure 2. Settup for Question 2.

- The pin of segment A plugs into Digital port 11. The pin of segment B plugs into Digital port 10. The pin of segment C plugs into Digital port 7. The pin of segment D plugs into Digital port 8. The pin of segment E plugs into Digital port 9. The pin of segment F is plugged into Digital port 12. The pin of segment G is plugged into Digital port 12. The Trigger and Echo pins of the sensor are plugged into Digital ports 2 and 3, respectively.

2. Code explanation:

```
1   void loop() {
2     digitalWrite(triggerPin, LOW);
3     delayMicroseconds(2);
4     digitalWrite(triggerPin, HIGH);
5     delayMicroseconds(10);
6     digitalWrite(triggerPin, LOW);
7     duration = pulseIn(echoPin, HIGH);
8     distance = duration * 0.034 / 2;
9     delay(200);
10    Serial.println(distance);
```

This code is simple. The first sends an ultrasonic pulse to activate the sensor. Then read the time from the sensor, calculate distance based on time. Print the results to 7-segment LED.

```
11  if (distance > 180 ) {
12    zero();
13  }
14  else if (distance <= 20) {
15    nine();
16  }
17  else if (distance <= 40) {
18    eight();
19  }
20  else if (distance <= 60) {
21    seven();
22  }
23  else if (distance <= 80) {
24    six();
25  }
26  else if (distance <= 100) {
27    five();
28  }
29  else if (distance <= 120) {
30    four();
31  }
32  else if (distance <= 140) {
33    three();
34  }
35  else if (distance <= 160) {
36    two();
37  }
38  else if (distance <= 180) {
39    one();
40  }
41 }
```

If the distance is from 0-20 cm, the value on the 7-segment LED is 9.

If the distance is from 20-40 cm, the value on the 7-segment LED is 8.

If the distance is from 40-60 cm, the value on LED 7 segment is 7.

If the distance is 60-80 cm, the value on the 7-segment LED is 6.

If the distance is 80-100 cm, the value on the 7-segment LED is 5.

If the distance is 100-120 cm, the value on the 7-segment LED is 4.

If the distance is from 120-140 cm, the value on the 7-segment LED is 3.

If the distance is from 140-160 cm, the value on the 7-segment LED is 2.

If the distance is from 160- 180 cm, the value on the 7-segment LED is 1.

If the distance is farther than 180cm, the value on the 7-segment LED is 0.

Câu hỏi 3. Xây dựng kịch bản cảnh cáo chất lượng không khí, kịch bản gồm có 1 đèn LED và 1 cảm biến MQ-135. Tìm hiểu về thông số PPM để thiết đặt thông số chất lượng không khí bất thường và cảnh báo thông qua đèn. Tùy vào mật độ chất lượng không khi thì đèn sẽ chớp tắt theo 3 mức cảnh báo là nhanh - trung bình - chậm.

1. Setup:

Components used:

- 1 Arduino Uno board.
- 1 MQ135 sensor.
- 6 Jumper wires (male-to-male).
- 1 yellow LED.
- 2 resistors.

Wiring connection:

- The MQ135 sensor has four pins: VCC, GND, D0 and A0.
    - VCC: Connect this pin to the Arduino's 5V pin thought 1 resistor. This provides power to the sensor.
    - GND: Connect this pin to any GND pin on the Arduino. This grounds the sensor.
    - A0: Connect this pin to an anolog pin on the Arduino. We used pin A0 in this exercise.
    - D0: Not in use.
- 1 yellow LED:
    - Cathode: Connect to any GND pin on the Arduino. This grounds the led.
    - Anode: We use digital pin 13 on the Arduino. This pin controls LED output.

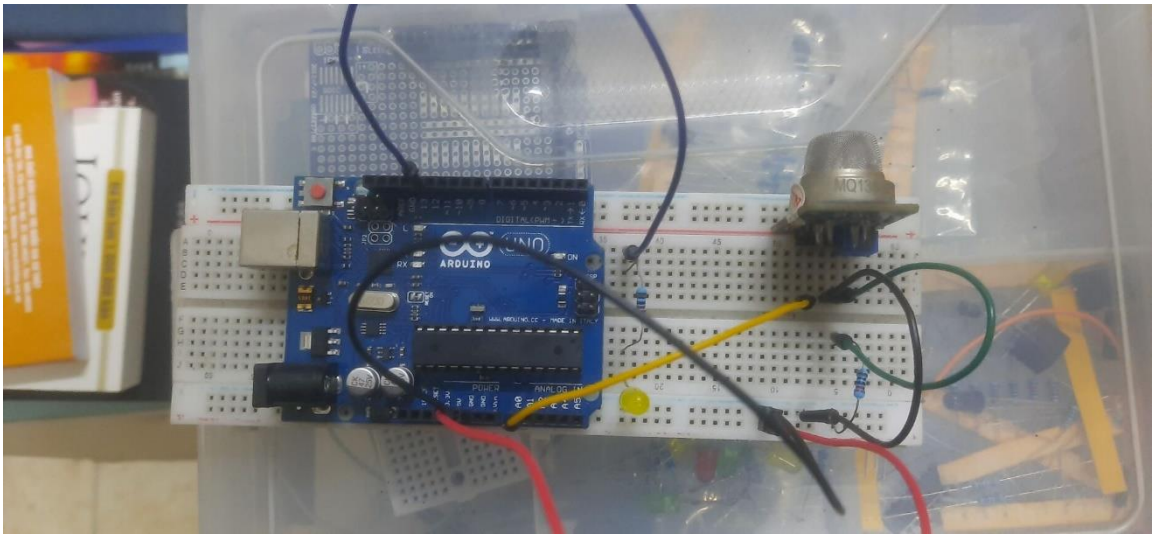

Figure 3. Settup for Question 3

PPM value(in air quality usecases):

- **Definition**: PPM represents the number of parts of a substance per one million parts of a solution or mixture.
- **Calculation**: To calculate ppm, divide the mass or volume of the substance by the total mass or volume of the solution and then multiply the result by one million.
- **Application**: Experts use ppm to measure pollutants in the air. It indicates how many milligrams of pollutants there are per square meter of air .

2. Code explanation:

```
float resistance = mq135_sensor.getResistance();
float ppm = mq135_sensor.getPPM();

Serial.print("Resistance: ");
Serial.print(resistance);
Serial.print("\t PPM: ");
Serial.print(ppm);
Serial.println(" ppm");

if (ppm > 50 && ppm ⩽ 150) {
    Serial.println("Moderate air quality!");
    blynk(500);
} else if (ppm > 150) {
    Serial.println("Unhealthy air quality!");
    blynk(100);
} else {
    Serial.println("Good air quality!");
    blynk(1000);
}
```

This code is simple. MQ135 read PPM and Resistance value. Resistance is the sampling range that sensor uses to take sample. If ppm in range 50 to 150, we defined that air quality in moderated level, led blynk 3 times with 500ms delay. If ppm is higher than 150, we defined it as unhealthy level, led blynk 3 times with 100ms delay. In normal, or good level, led blynk 3 times with 1s delay.

Câu hỏi 4. Xây dựng ứng dụng tảng nhiệt cho DataCenter, kịch bản gồm 1 cảm biến nhiệt độ, độ ẩm và 3 đèn LED. Nếu nhiệt độ quá cao thì điều khiển bật quạt (tượng trưng bằng các đèn LED) để tảng nhiệt cho các Server. Tương ứng với 3 mức của quạt được mô phỏng bằng 3 đèn LED, tùy theo nhiệt độ trong DataCenter. Bên cạnh đó, nếu độ ẩm trong không

khí của DataCenter quá cao khi quạt được bật, đồng thời nhiệt độ cũng trả trở lại mức bình thường thì điều khiển để quạt tắt đi (3 đèn tắt).

1. Setup:

Components used:

- 1 Arduino Uno board.
- 1 DHT11 sensor.
- 7 Jumper wires (male-to-male).
- 3 yellow LEDs.
- 3 resistors.

Wiring connection:

- The DHT11 sensor has three pins: VCC, GND, and DATA.
    - VCC: Connect this pin to the Arduino's 5V pin thought 1 resistor. This provides power to the sensor.
    - GND: Connect this pin to any GND pin on the Arduino. This grounds the sensor.
    - DATA: Connect this pin to a digital input pin on the Arduino. We used pin number 2 in this exercise.
- 3 yellow LEDs:
    - Cathode: Connect to any GND pin on the Arduino. This grounds the led.
    - Anode: We use digital pin from 5 to 7 on the Arduino. Those pins control LEDs output.

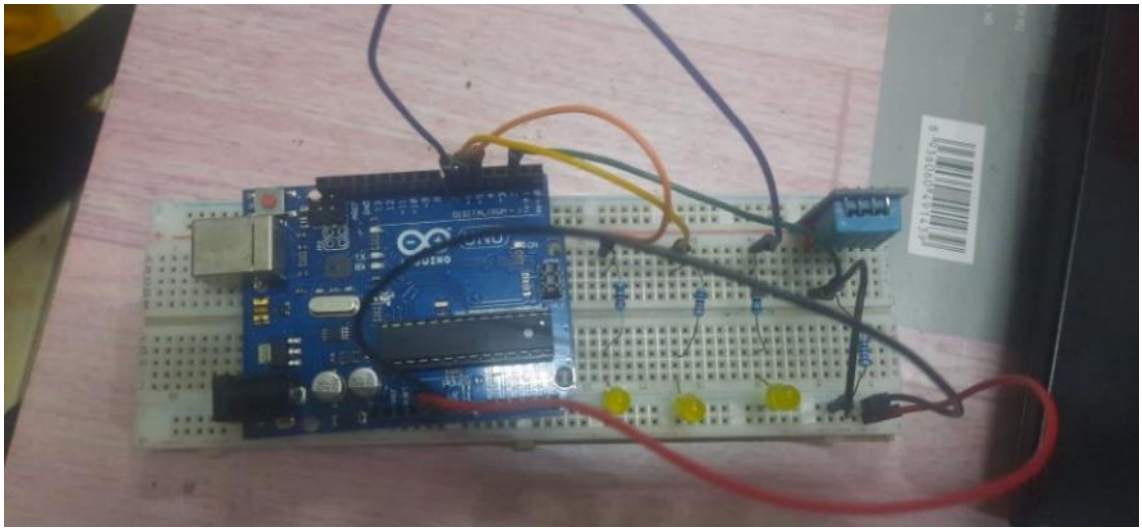

Figure 4. Settup for Question 4

2. Code explanation:

```
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();

    if (isnan(temperature) || isnan(humidity)) {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }

    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.print(" °C\tHumidity: ");
    Serial.print(humidity);
    Serial.println(" %");

    if (temperature > TEMP_THRESHOLD_NORMAL && temperature ≤ TEMP_THRESHOLD_H1) {
        Serial.println("Temperature is little high");
        digitalWrite(FAN_PIN_1, HIGH);
        digitalWrite(FAN_PIN_2, LOW);
        digitalWrite(FAN_PIN_3, LOW);
    } else if (temperature > TEMP_THRESHOLD_H1 && temperature ≤ TEMP_THRESHOLD_H2) {
        Serial.println("Temperature is high");
        digitalWrite(FAN_PIN_1, HIGH);
        digitalWrite(FAN_PIN_2, HIGH);
        digitalWrite(FAN_PIN_3, LOW);
    } else if (temperature > TEMP_THRESHOLD_H2 && temperature ≤ TEMP_THRESHOLD_H3) {
        Serial.println("Temperature is too high");
        digitalWrite(FAN_PIN_1, HIGH);
        digitalWrite(FAN_PIN_2, HIGH);
        digitalWrite(FAN_PIN_3, HIGH);
    } else {
        Serial.println("Temperature is normal");
        digitalWrite(FAN_PIN_1, LOW);
        digitalWrite(FAN_PIN_2, LOW);
        digitalWrite(FAN_PIN_3, LOW);
    }

    if ((FAN_PIN_1 == HIGH || FAN_PIN_2 == HIGH || FAN_PIN_3 == HIGH) && humidity > HUMIDITY_THRESHOLD_HIGH) {
        Serial.println("Server cooled down, turn off the fan");
        digitalWrite(FAN_PIN_1, LOW);
        digitalWrite(FAN_PIN_2, LOW);
        digitalWrite(FAN_PIN_3, LOW);
    }

    Serial.println("====================================");

delay(2000);  // Chờ 2 giây trước khi đọc dữ liệu từ cảm biến lại
```

Some defined variables:

- TEMP_THRESHOLD_NOMAL(35 cencius degree),
- TEMP_THRESHOLD_H1(38 cencius degree),
- TEMP_THRESHOLD_H2(40 cencius degree),
- TEMP_THRESHOLD_H3(42 cencius degree).
- HUMIDITY_THRESHOLD_HIGH(80%).
- FAN_PIN_1(5).
- FAN_PIN_2(6).
- FAN_PIN_3(7).

Read tempurature and humidity from DHT11. If the temperature is in range 35 to 38, turn on 1 fan. If the temperature is in the range of 38 to 40, turn on 2 fans. If the temperature is in the range of 40 to 42, turn on 3 fans. If the

temperature is below 35, turn off all fans. While fans are turning on, if humid is over 80%, turn off all fans.

Câu hỏi 5. Sử dụng cảm biến chuyển động hồng ngoại PIR (HC-SR501), tiến hành đọc giá trị từ cảm biến và in ra serial. Tìm hiểu cách thức vận hành, sơ đồ mạch điện và cách thức sử dụng của loại cảm biến này.

1. Setup:

Components used:

- 1 Arduino Uno board.
- 1 HC-SR501 PIR sensor.
- 3 Jumper wires (male-to-female).

Wiring connection - The HC-SR501 sensor has three pins: VCC, GND, and OUT.

- VCC: Connect this pin to the Arduino's 5V pin. This provides power to the sensor.
- GND: Connect this pin to any GND pin on the Arduino. This grounds the sensor.
- OUT: Connect this pin to a digital input pin on the Arduino. We used pin number 7 in this exercise.

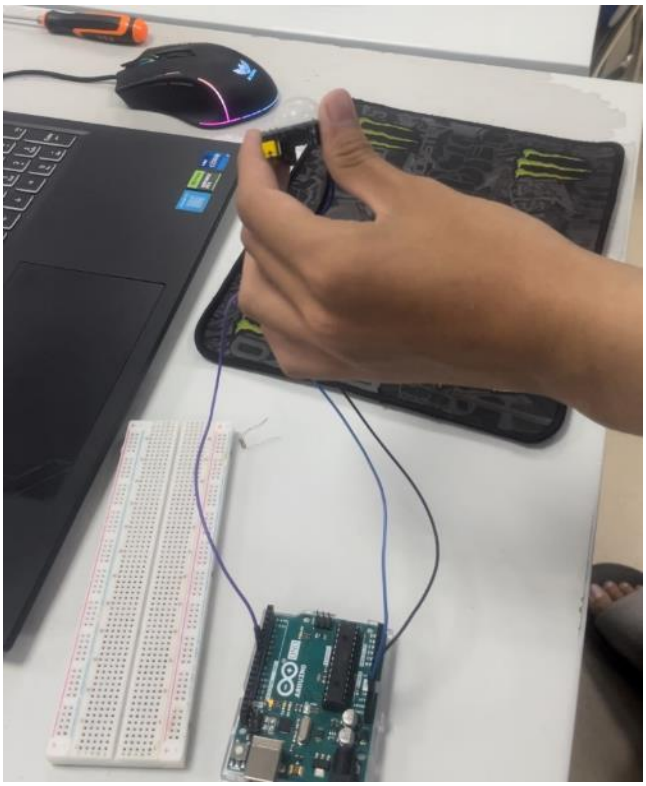

Figure 5. Arduino connected to the PIR (HC-SR501) sensor.

How the HC-SR501 sensor works:

- All objects, including the human body, at temperatures above absolute zero emit heat energy in the form of infrared radiation. The hotter an object is, the more radiation it emits. This radiation is not visible to the human eye because it is emitted at infrared wavelengths. The PIR sensor is specifically designed to detect such levels of infrared radiation.

- A PIR sensor consists of two main parts:
  - A **pyroelectric sensor**, which can be seen in the image below as a round metal with a rectangular crystal in the center. The pyroelectric sensor consists of a window with two rectangular slots and is made of a material (typically coated silicon) that allows infrared radiation to pass through. Behind the window, there are two separate infrared sensor electrodes, one responsible for producing the positive output and the other for producing the negative output. The two electrodes are wired such that they cancel each other out. *When there is no movement around the sensor*, both slots detect the same amount of infrared radiation, resulting in a zero output signal. *But when a warm body like a human passes by*, it first intercepts half of the sensor. This causes a positive differential change between the two halves. When the warm body intercepts the other half of the sensor (leaves the sensing region), the opposite happens, and the sensor produces a negative differential change. By reading this change in voltage, motion is detected.

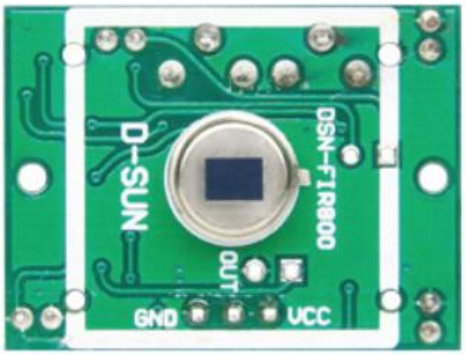

Figure 6. The pyroelectric sensor (the black rectangle in the middle).

  - A special lens called a **fresnel lens** which focuses the infrared signals on the pyroelectric sensor. This is what increases the range and field of view of the PIR sensor. A Fresnel lens consists of a series of concentric grooves carved into plastic. These contours act as individual refracting surfaces, gathering parallel light rays at a focal point. As a result, a Fresnel lens, although smaller in size, is able to focus light similarly to a conventional optical lens.

Figure 7. The fresnel lens.

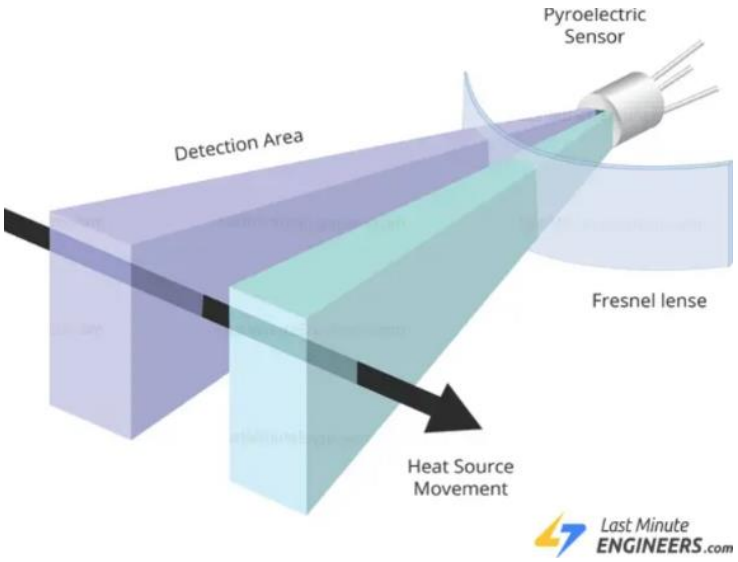

Figure 8. Illustration of the sensor and the lens.

HC-SR501 sensor configuration:

- Jumper Settings: This HC-SR501 module has a small jumper (a tiny movable piece of plastic) on the back. This jumper setting determines the trigger mode. There are two modes as follows:
    - **H (High):** This enables repeat triggering. The sensor will output a HIGH signal for a set time (determined by the delay time potentiometer) whenever motion is detected within its range. The output will then go LOW again until the next motion is detected.
    - **L (Low):** This disables repeat triggering. The sensor will only output a short HIGH pulse (also determined by the delay time potentiometer) when motion is first detected. Further detection is blocked until the output returns to LOW at the end of the time delay.
- Potentiometer Adjustments: The HC-SR501 has two small potentiometers (adjustable dials) on the sensor body:
    - **Sensitivity:** This adjusts the range of motion the sensor will detect. Turning the dial clockwise increases the sensitivity, detecting motion from a larger distance. Conversely, turning it counter-clockwise reduces the sensitivity.
    - **Delay Time:** This sets the amount of time the sensor output stays HIGH after motion is detected. Turning the dial clockwise increases the delay time, while turning it counter-clockwise reduces it.

Reference: https://lastminuteengineers.com/pir-sensor-arduino-tutorial/

With the above findings as well as experiments of our own, the final configuration that worked for us in this exercise is as follows:

- Trigger mode is set to Low since we did't want it to continuously detect motions leading to overloaded output in a short period of time.
- The sensitivity potentiometer was left in its default position as the range was enough for us to test.

- The delay time potentiometer was turned counter-clockwise to the maximum. Since the movement was rather quick, we didn't want to have too much delay after the motion had been detected.

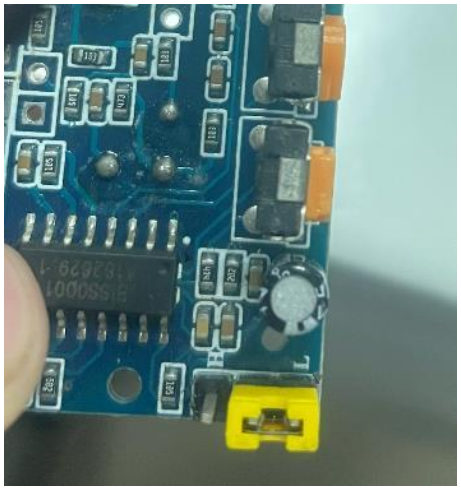

Figure 9. Trigger mode set to L (by yellow block).

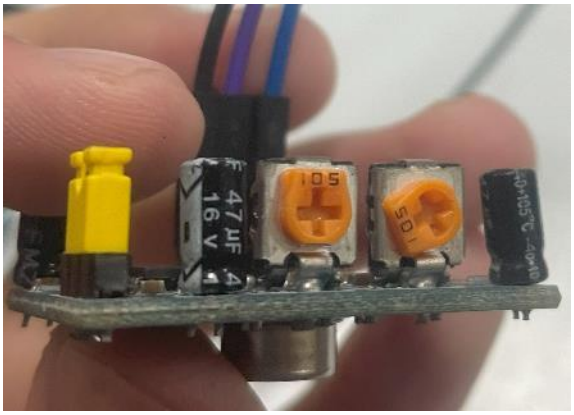

Figure 10. Sensitivity (left) and delay time (right) potentiometers.

2. Code explanation:

```
const int pirPin = 7;

void setup() {
  Serial.begin(9600);
  pinMode(pirPin, INPUT);
}

void loop() {
  int sensorValue = digitalRead(pirPin);

  if (sensorValue == HIGH) {
    Serial.println("Motion detected!");
  } else {
    Serial.println("No motion detected.");
  }

  delay(500);  // Short delay to avoid overwhelming serial output
}

```

Figure 11. Code for Question 5.

The above code is rather straightforward. We define the pin number for the input from the sensor and then read the value from that. If the sensor's value is HIGH, we know that there is movement and thus, print "Motion detected!". Otherwise, print "No motion detected" when the value is LOW. Finally, a little delay is added so the serial output doesn't get overwhelmed.

Câu hỏi 6. Mô phỏng máy giám sát vật thể, xây dựng kịch bản gồm có 1 cảm biến chuyển động, 2 đèn LED. Khi đối tượng cần quan sát chuyển động thì 2 đèn LED sẽ chớp tắt liên tục để báo động. Khi không có chuyển động thì trở lại trạng thái bình thường.

1. Setup:

Components used:

- 1 Arduino Uno board.
- 1 breadboard.
- 1 HC-SR501 PIR sensor.
- 2 LEDs.
- 2 resistors.
- 3 Jumper wires (male-to-female).
- 3 Jumper wires (male-to-male).

Wiring connection:

- The wiring connection between the sensor and the Arduino is almost identical to Question 5 (pin number is still 7), only needs more wires since there are breadboard and LEDs in-between.
- LEDs and resistors wiring connection method are also similar to previous questions (pin number 8 and 9).

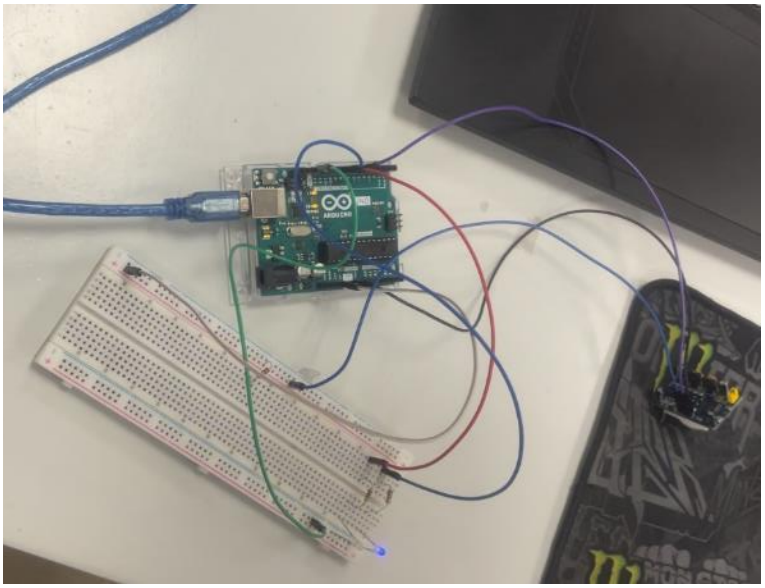

Figure 12. Setup for Question 6.

2. Code explanation:

```
void loop() {
  int sensorValue = digitalRead(pirPin);

  if (sensorValue == HIGH) {
    Serial.println("Motion detected!");
    blinkLEDs();
  } else {
    Serial.println("No motion detected.");
    digitalWrite(ledPin1, LOW);
    digitalWrite(ledPin2, LOW);
  }

  delay(500);  // Short delay to avoid overwhelming serial output
}

void blinkLEDs() {
  // Blink LEDs continuously until no motion is detected
  while (digitalRead(pirPin) == HIGH) {
    digitalWrite(ledPin1, HIGH);
    digitalWrite(ledPin2, LOW);
    delay(500);
    digitalWrite(ledPin1, LOW);
    digitalWrite(ledPin2, HIGH);
    delay(500);
  }
}
```

Figure 13. Code for Question 6.

The logic for detecting motion is likewise to Question 5. The only addition is that it will also call the blinkLEDs function when there are movements from the target (the hand movement in the video demo). The blinkLEDs function is to have the 2 LEDs switching continuously between LOW and HIGH, thus, creating the blinking effect.

Câu hỏi 7. Mô phỏng phòng đọc sách thông minh, xây dựng kịch bản gồm có 1 cảm biến ánh sáng, 10 đèn LED, 1 nút bấm. Tuỳ theo cường độ ánh sáng đo được, nếu ánh sáng quá kém thì tự động bật tối đa số đèn, tương tự như thế điều chỉnh số lượng đèn sáng dựa vào cường độ ánh sáng để người đọc sách có đủ ánh sáng. Nút bấm dùng để thay đổi chế độ đèn để tiết kiệm điện: chế độ 1 là sử dụng 5 đèn khi bấm 1 lần, chế độ 2 là sử dụng tối đa 10 đèn nếu bấm liên tiếp 2 lần.

1. Setup:

Components used:

- 1 Arduino Uno board.
- 1 breadboard.
- 1 BH1750 light sensor.
- 10 LEDs.
- 10 resistors.
- 4 Jumper wires (male-to-female).
- 14 Jumper wires (male-to-male).

Wiring connection:

- The BH1750 sensor typically has four pins: VCC, GND, SCL, and SDA.
    - Connect the VCC pin of the BH1750 sensor to a row of positive power rails on the breadboard.
    - Connect the GND pin of the BH1750 sensor to a row of negative ground rails on the breadboard.
    - Locate the SCL and SDA pins of the BH1750 sensor. Connect them to the SCL (usually pin A5) and SDA (usually pin A4) pins of the Arduino Uno, respectively.
- LEDs and resistors wiring connection method are also similar to previous questions (pin number from 4 to 13).

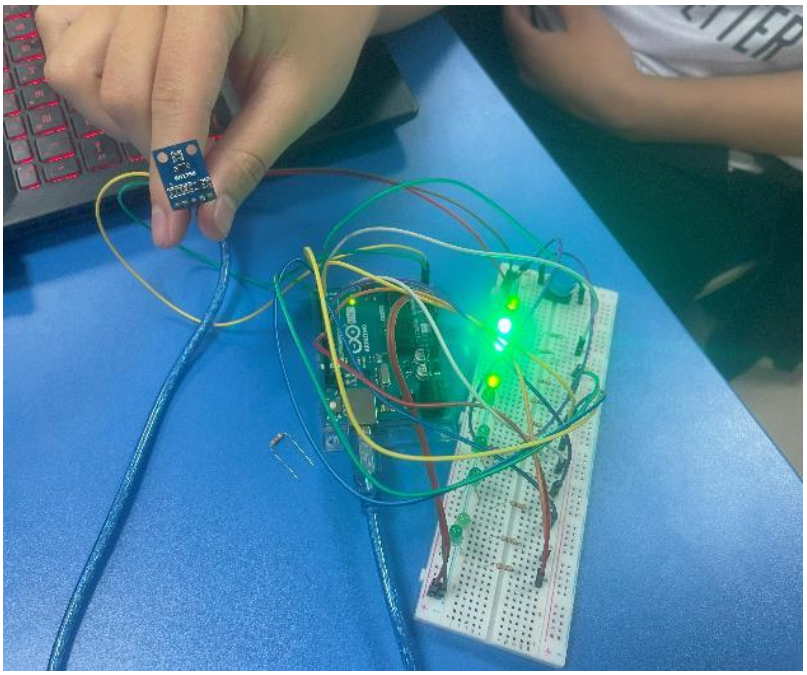

Figure 14. Setup for Question 7.

2. Code explanation:

Figure 11 shows the loop function which contains the primary flow for Question 7. A summary of the code is as follows:

1) Read Light Sensor: The code gets a reading from the light sensor.
2) Check Button and Mode Change:
    a. It waits for a button press. If pressed and a specific flag allows it (first flag), the mode changes:
    b. First press: Switches to 5-LED mode (sets currentMode to 0).
    c. Second press within a short time window (debounce): Switches to 10-LED mode (sets currentMode to 1).
    d. The flag (first) ensures only one mode change happens per button press and the time window prevents accidental double-taps.
3) Calculate LED Count based on Light: Use map function to convert the light sensor reading into a value representing the maximum number of LEDs that can be lit for good visibility (more light = fewer LEDs).
4) Adjust LED Count: Based on the current mode (5 or 10 LEDs), limit the calculated maximum if it exceeds the mode's limit.

5) Control LEDs: Iterate through all 10 LEDs, turning them on if their index is less than the number of LEDs to light (based on mode and light sensor). Otherwise, they are turned off.

```
int lightValue = lightSensor.readLightLevel();

  // Xác định chế độ đèn
  delay(500);
  if (digitalRead(buttonPin) == LOW && first == false) {
    currentMode = 0;
    Serial.println("1st pressed");
    first = true;
    delay(500);
    if (digitalRead(buttonPin) == LOW && first == true) {
      currentMode = 1;
      Serial.println("2nd pressed");
      first = false;
    }
  }

  Serial.println(currentMode);
  first = false;
  maxLedsOn = map(lightValue, 0, 1000, 0, 10);

  // Điều chỉnh số lượng đèn LED
  switch (currentMode) {
    case 0: // Chế độ 5 đèn
        if(maxLedsOn > 5){
          maxLedsOn = 5;
        }
        numLedsOn = 5 - maxLedsOn;
      break;
    case 1: // Chế độ 10 đèn
        if(maxLedsOn > 10){
            maxLedsOn = 10;
        }
        numLedsOn = 10 - maxLedsOn;
      break;
  }

  Serial.println(numLedsOn);
  Serial.println(maxLedsOn);

  // Bật/tắt đèn LED
  for (int i = 0; i < 10; i++) {
    if (i < numLedsOn) {
      digitalWrite(ledPins[i], HIGH);
    } else {
      digitalWrite(ledPins[i], LOW);
    }
  }
```

Figure 15. The loop function of Question 7.

YÊU CẦU CHUNG

1) Đánh giá

- Chuẩn bị tốt các yêu cầu đặt ra trong bài thực hành.
- Sinh viên hiểu và tự thực hiện được bài thực hành, trả lời đầy đủ các yêu cầu đặt ra.
- Nộp báo cáo kết quả chi tiết những đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả *(nếu có)*; giải thích cho quan sát *(nếu có)*.
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

2) Báo cáo

- File .PDF hoặc .docx. Tập trung vào nội dung, giải thích.
- Nội dung trình bày bằng Font chữ Cambria hoặc Times New Roman (*tuy nhiên, phải chuyển đổi hết báo cáo này sang 1 font chữ thống nhất*) – cỡ chữ 13. Canh đều (Justify) cho văn bản. Canh giữa (Center) cho ảnh chụp.
- Đặt tên theo định dạng: Mã lớp-LabX_MSSV1_MSSV2. (trong đó X là Thứ tự buổi Thực hành).

  Ví dụ: NT532.O21.1-Lab01_25520001_25520002
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Không đặt tên đúng định dạng – yêu cầu, sẽ KHÔNG chấm điểm bài thực hành.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.

HẾT