Tests with different sensors by using ESP32

Assembling of basic sensors to help students to develop the Arduino Project

26th August 2022

Contents

1

Introduction

Today's laboratory session has the objective of assembling different types of sensors, putting into practice the preliminary concepts developed in the documentation in the *Instruction manual ESP32*.

The main goal of the session is to learn how to use these sensors with our ESP32 board, their functions, the different codes and print the results on our computer screen. Furthermore, there will be some instructions to help the student.

The sensors we will learn about are the following:

- Infrared motion sensor
- Temperature sensor
- Ambient light sensor
- Sound sensor

And another modules that aren't sensors but will learn how to use them:

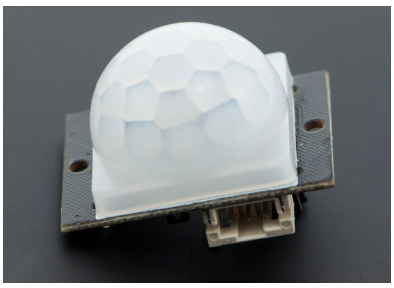
- Push button
- Voltage divider
- Arduino LCD display

2

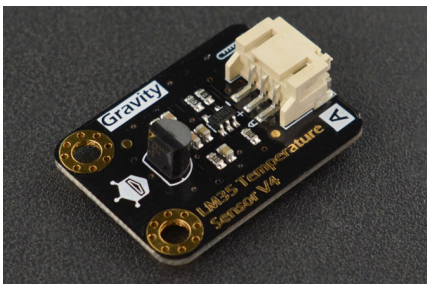
Theory Resume

Every module has it's own function.

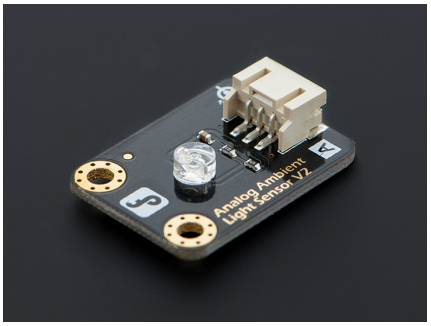
The infrared motion sensor is a simple to use motion sensor. Power it up and wait 1-2 seconds for the sensor to get a snapshot of the still room. If anything moves after that period, the 'alarm' pin will go low, and you can put a LED to make it turn on, to act as alarm.



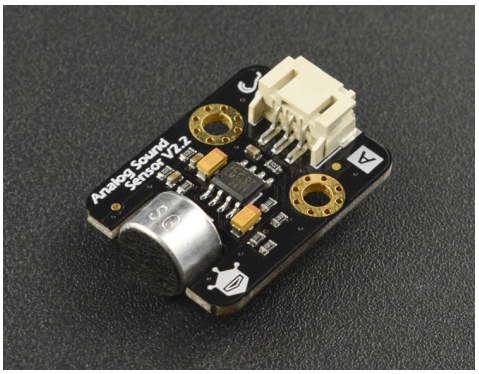
The temperature sensor can be used to detect ambient temperature and offers a functional range between 0 to 150°C. To see the temperature you can put *Serial.println(")* and open the Serial Monitor.



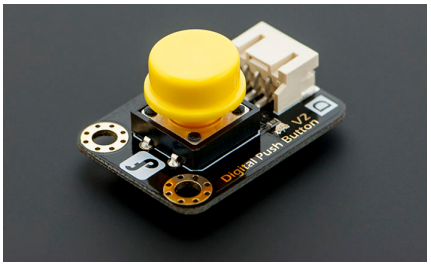
The ambient light sensor can help you to detect the light density of your enviroment and reflect this information back via an analog voltage signal to your Arduino controller. You can set the threshold of voltage level to trigger another aspect of your project. This sensor can be applied in any project where light density is involved. For example, you can use this sensor to automatically adjust the lights depending on the suns intensity. As in the temperature sensor you can see the luminosity in the Serial Monitor putting *Serial.println(")*



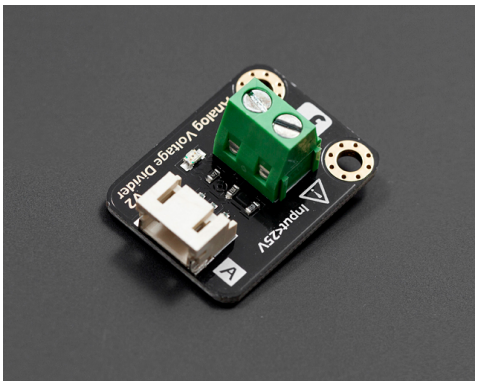
Analog Sound Sensors are typically used in detecting the ambient loudness in your environment. The Arduino can collect its output signal by imitating the input interface. You may use it to make some fun interactive projects such as a voice operated switch.



The push button gives your Arduino project the first touch of the physical world. Simply plug into the IO expansion board for quick and easy plug and play. Use the button to trigger another sensor, build a gamepad, etc.



The voltage divider can detect the supply voltage up to 25V, is based on resistor divider principle. The voltage detection module allows the input voltage to reduce 5 times. As the Arduino analog input voltage is up to 5V, so voltage detection module's input voltage can not be greater than the 5Vx5 = 25V.



Arduino LCD with RGB Backlight Display module comes with RGB full color backlight. Usually, Arduino LCD display projects will run out of pin resources easily, especially with Arduino Uno. This only needs 4 pins for the LCD display: VCC, GND, SDA, SCL. It will saves at least 4 digital / analog pins on Arduino.

3

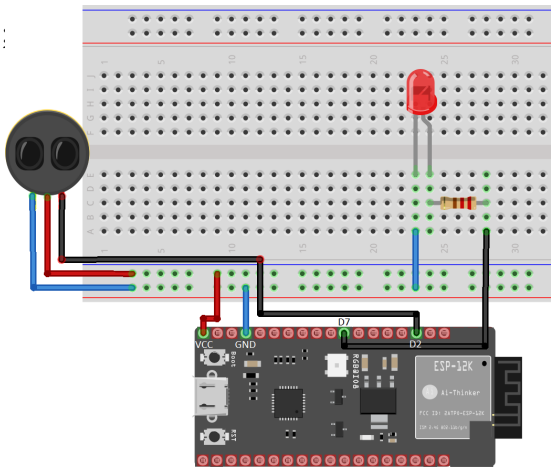
Example Codes

To give you an idea of how each sensor works, here are some basic codes for understanding how they work. Depending it what sensor you use, you will have to install the corresponding libraries, they are specified in the code as *# include name_of_the_library.h*

3.0.1 Infrared motion sensor

```
const int buttonPin = D2;
const int ledPin =  D7; //Connecting a led at the digital
 ↪  pin 6 to detect when something moves.
void setup(){
  pinMode(ledPin, OUTPUT); //declare LED as output
  pinMode(buttonPin, INPUT); //declare sensor as input
}
void loop(){
  if (digitalRead(buttonPin) == HIGH)
  {
    digitalWrite(ledPin, HIGH);
  }
  else {
```

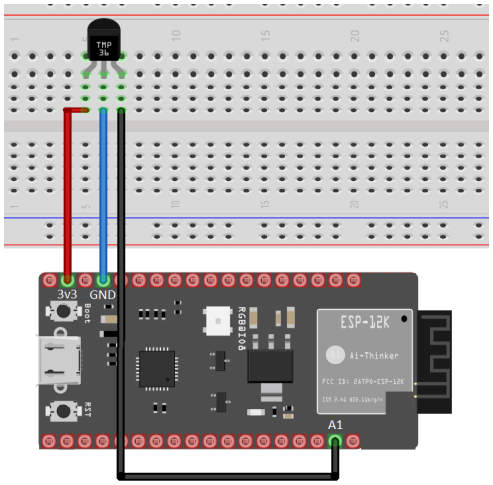
```
    digitalWrite(ledPin, LOW);
  }
}
```



3.0.2 Temperature sensor

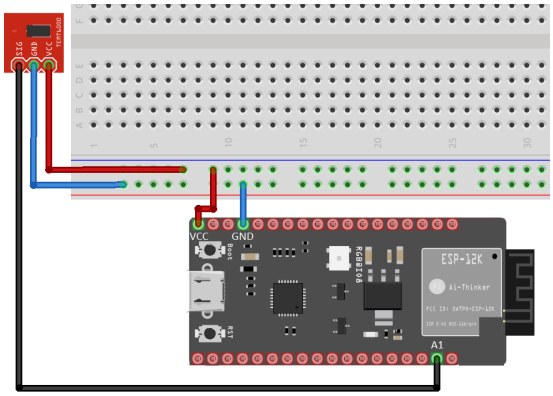
```
const int tempPin = 2; //analog input pin constant
int tempVal; //temperature sensor raw readings
float volts; //variable for storing voltage
float temp; //actual temperature variable

void setup(){
    Serial.begin(9600); //Set Baud Rate to 9600 bps
}

void loop(){
  //read the temp sensor and store it in tempVal
  tempVal = analogRead(tempPin);

  volts = tempVal/1023.0; //normalize by the maximum
   ↪ temperature raw reading range

  temp = (volts - 0.5) * 100 ; //calculate temperature from
   ↪ voltage as per the equation found on the sensor spec
   ↪ sheet.

  Serial.print(" Temperature is:   ");
  Serial.print(272.2-temp);

  Serial.println (" degrees C");
```

```
22    delay(1000);
23  }
```



3.0.3 Ambient light sensor

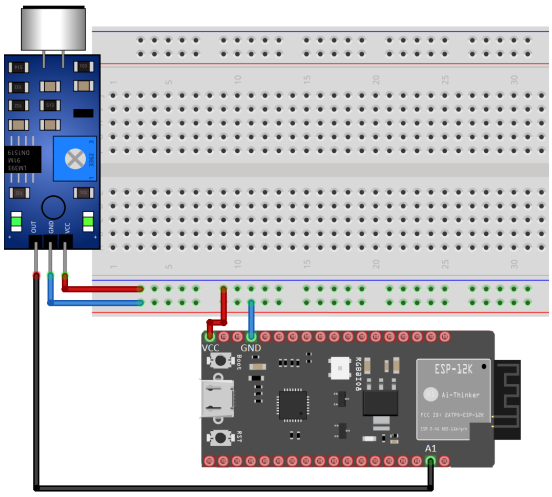
```
1  void setup(){
2    Serial.begin(9600);
3  }
4
5  void loop(){
6    int val;
7    val=analogRead(A1);    //connect sensor to analog 1
8    Serial.print("Lux: "); //Read Lux and print
9    Serial.print(val);
10   Serial.println(" lx");
11   delay(500);
12 }
```



3.0.4 Sound sensor

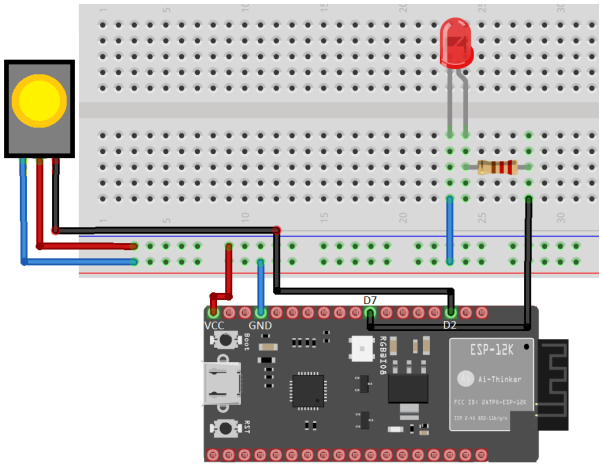
```
1  cont int pin = A1;
2  int val;
3
```

```
void setup(){
  Serial.begin(9600);
}
void loop(){
    val=analogRead(pin);
    Serial.println(val,DEC);
    delay(100);
}
```



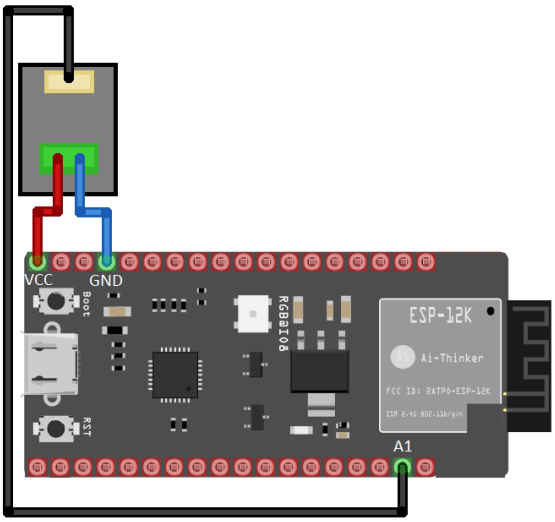
3.0.5 Push button

```
//When you push the digital button, the Led 7 on the board
  ↪  will turn off. Otherwise, the led turns on.
int ledPin = D7; //choose the pin for the LED
int inputPin = D2; //Connect sensor to input pin 3

void setup(){
  pinMode(ledPin, OUTPUT); //declare LED as output
  pinMode(inputPin, INPUT); //declare push button as input
}

void loop(){
  int val = digitalRead(inputPin); //read input value
  if (val == HIGH) { //check if the button is pressed
    digitalWrite(ledPin, LOW);
  } else {
    digitalWrite(ledPin, HIGH);
  }
}
```



3.0.6 Voltage divider

```
void setup(){
  Serial.begin(9600);
}
void loop(){
  int val;
  float temp;
  val=analogRead(A2); //This divider module will divide the measured voltage by 5, the maximum voltage it can measure is 25V.
  temp=val/40.92;
  val=(int)temp;
  Serial.println(val);
  delay(100);
}
```



3.0.7 Arduino LCD display

```
#include "DFRobot_RGBLCD1602.h"

DFRobot_RGBLCD1602 lcd(/*lcdCols*/16,/*lcdRows*/2);  //16 characters and 2 lines of show

void breath(unsigned char color){
    for(int i=0; i<255; i++){
        /**
          *  brief set backlight PWM output
          *  param color  backlight color PreferencesREG_RED\REG_GREEN\REG_BLUE
          *  param pwm  color intensity   range(0-255)
```

```
            */
            lcd.setPWM(color, i);
            delay(5);
        }

        delay(500);
        for(int i=254; i>=0; i--){
            lcd.setPWM(color, i);
            delay(5);
        }

        delay(500);
}

void setup() {
        lcd.init();
        // Print a message to the LCD.
        lcd.print("hello, world!");

}

void loop() {
        breath(REG_RED);
        breath(REG_GREEN);
        breath(REG_BLUE);
}
```