WELCOME

Thank you very much for choosing our products.
The data package we provide is a zip compression package. Please unzip the data package before learning. Start from this PDF course.

Technical support

If you have any questions about the product, don't worry. You can view the "Common problem.PDF" file or contact us.
Our email is straysnail-wiki@outlook.com
We will reply you within one working day and give you a reasonable solution.

Safety matters

- This product is recommended for children over 6 years old.
- This product needs battery power supply. It is not waterproof. Don't play in water.
- Turn off the power switch when not in use.

About Stray Snail

- Stray Snail is the brand trademark of Shenzhen Snail Man Intelligent Technology Co., Ltd.
- Mainly for STEAM education products, self developed, including hardware and software design.
- The main products are smart cars, 3D printers and DIY writing plotters, mechanical arms, and some e-learning kits.
- The main control board includes Arduino, raspberry pi, Micro:bit and ESP series.
- Provide customization services: hardware customization, such as the design of sensor modules or the entire product.
- Software customization: Bluetooth and WiFi control APP of Android and iOS, and secondary development customization of scratch3.0 graphical programming.

If you want to know more about Stray Snail, you can visit our official website:

https://www.straysnail.com

Copyright

Stray Snail

Ⅰ. Introduction of Intelligent Windmill

The Netherlands is known as “the country of windmills”. When it comes to Dutch windmill, one can imagine that a huge windmill stands on an endless grassland with a river beside it, and the picture is as beautiful as a fairy tale. The beautiful things are pleasing to the eyes and easy to inspire creative inspiration, so we designed this intelligent windmill for STEAM education.

Since it is a Dutch windmill, there is a big windmill that can rotate. The windmill rotates slowly, seemingly driven by the gentle breeze of the grassland. On one side of the roof, a mechanical butterfly fluttered its wings vividly. On the other side of the roof is an infinite mirror tunnel lamp. The neat light tunnel changes wonderful colors, increasing the illusion.

With the popularization of smart home, we have also added some automatic devices

close to life. For example, human body sensing lamp, automatic window closing device, password door, temperature and humidity detection, music box, laser projector and mobile phone APP control. These functions can be applied to real life.

The material of the intelligent windmill is wood, which is very easy to color. Pick up your color pens and draw your own artwork.

目录

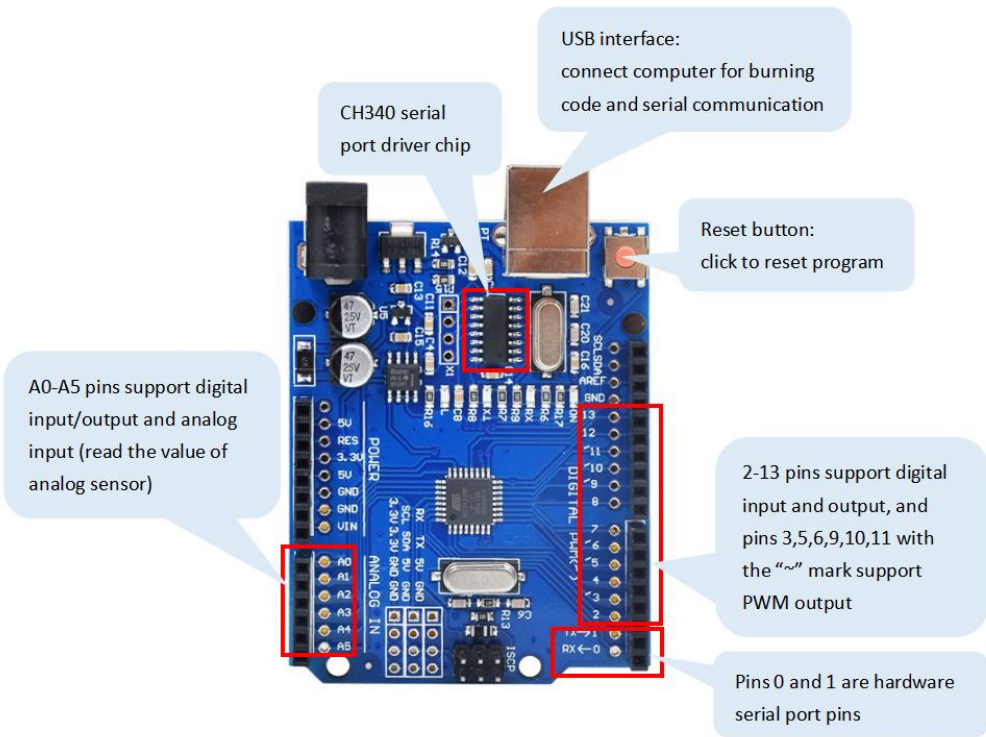
Ⅱ. Product features

- The structure and appearance are beautiful, and it is also a good artistic decoration on the table.
- It is equipped with color pens to create your own works of art.
- There are many kinds of electronic modules, such as digital sensor, analog sensor, driver sensor, etc. Sound, communication and screen, after learning these, you can develop your own products.
- The large windmill can rotate.

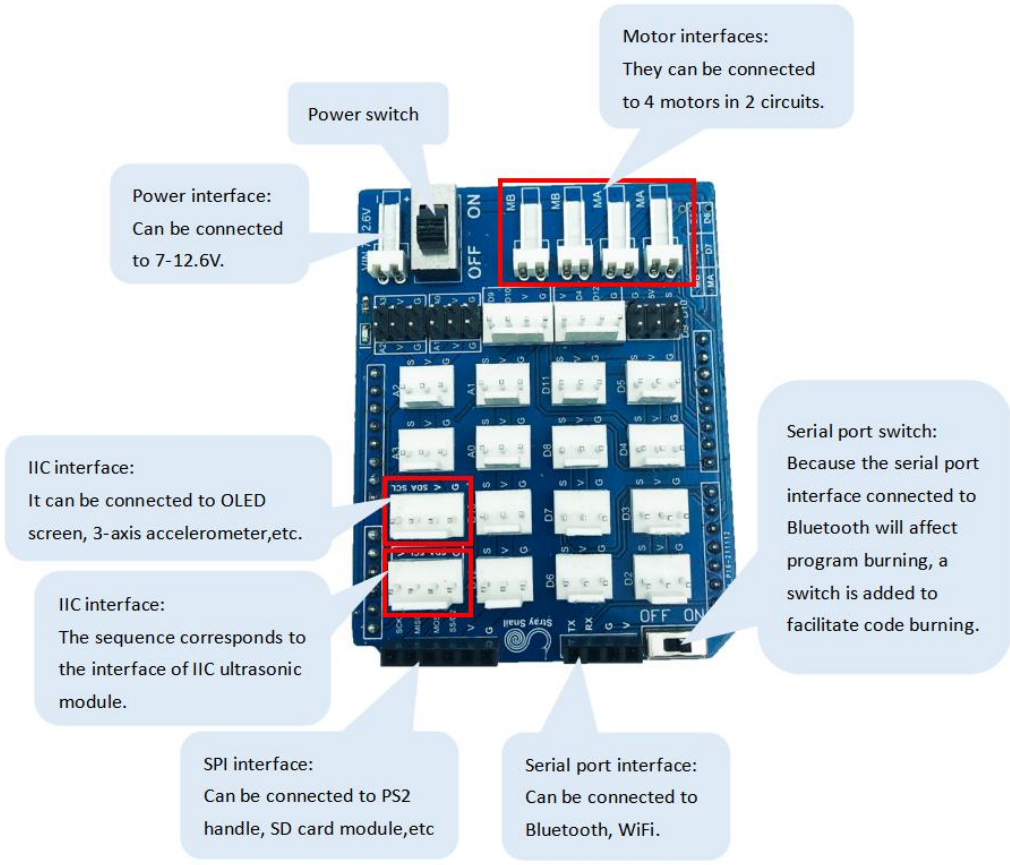
- The mechanical butterfly can flutter its wings.
- Simple laser projector.
- There is a wonderful special effect lamp--the infinite mirror tunnel lamp, which is really cool when it comes on.
- The screen displays multi-function menu, button selection and switching functions, which you can learn to write screen interaction code.
- Morse code gate.
- Temperature and humidity meter.
- The window can be closed automatically when it rains.
- Music box.
- Automatic switch lamp.
- IOS and Android APP control.
- Support C language and Scratch 3.0 compatible graphical programming.

Ⅲ. Products parameters

Input voltage of USB interface of Arduino UNO main board : 5V
Input voltage of external power interface : 7~12.6V
Working voltage of all sensor modules : 5V
The illustration of Arduino UNO main board is as follows :



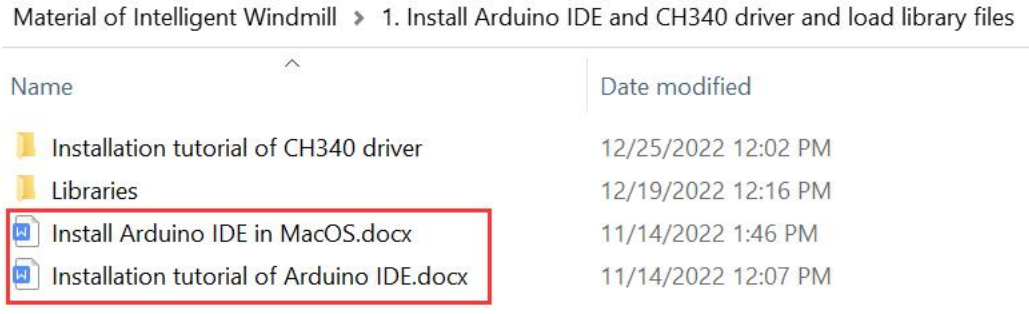
The illustration of expansion board is as follows:



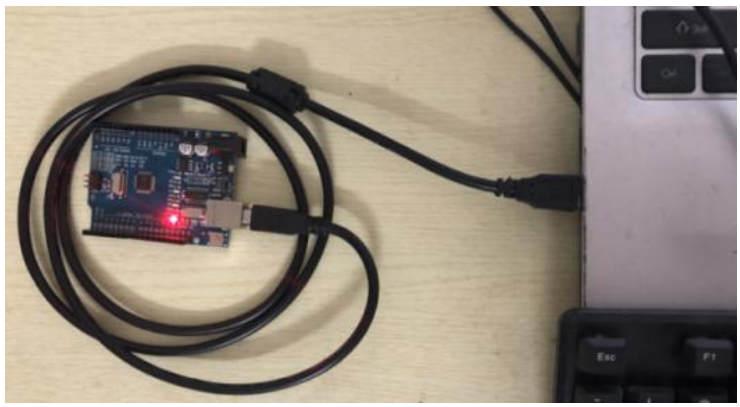
Ⅳ. Install software and its environment configuration

1. Install Arduino IDE and serial port driver

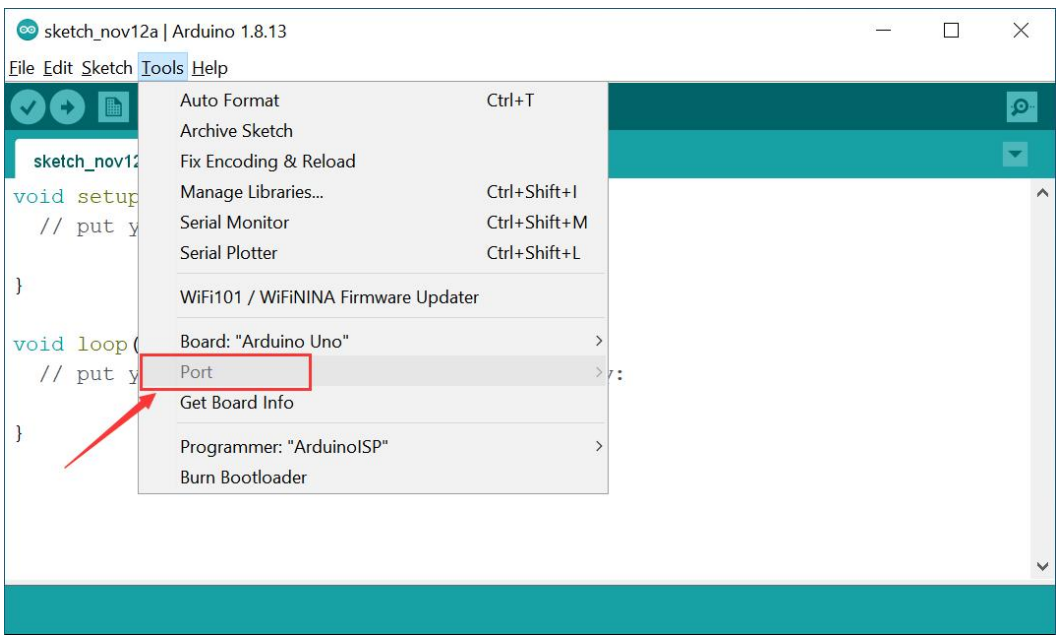
(1) If you are still a novice, please click this link install the Arduino IDE and follow the installation of the Arduino IDE software.



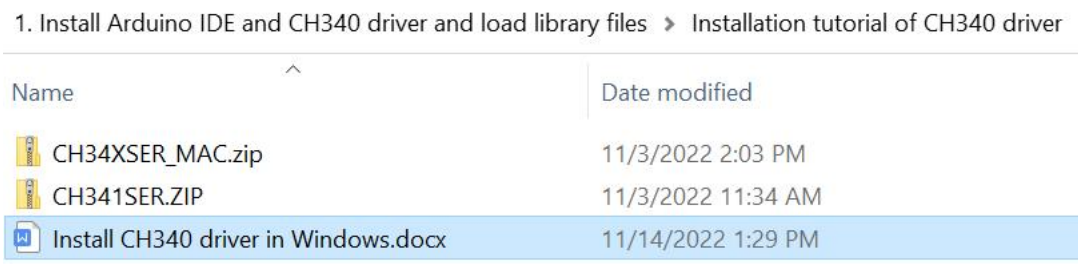
(2) Connect the USB data cable between the Arduino UNO main board and the USB of the computer.



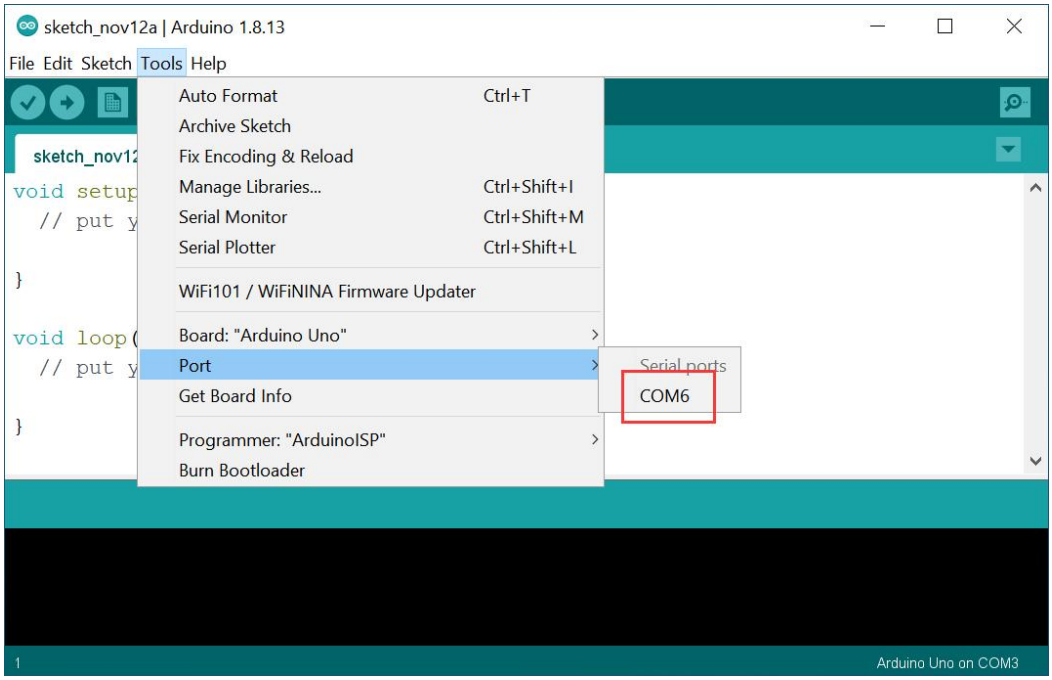
Open the Arduino IDE and click “Tools”. If the CH340 driver cannot be automatically identified, as shown in the following figure, you need to manually install the CH340 driver.



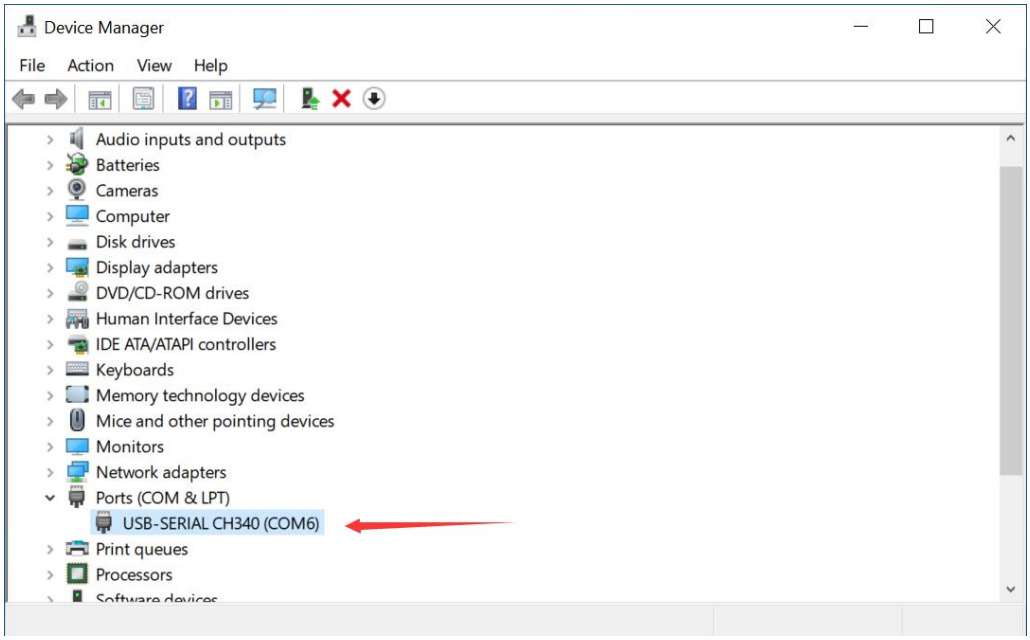
In the material, we provide a CH340 driver installation tutorial. Open the document and follow the steps to install the driver.



If it can be identified, as shown in the figure below.



Open the device manager of the computer to see the CH340 driver, as shown in the following figure.

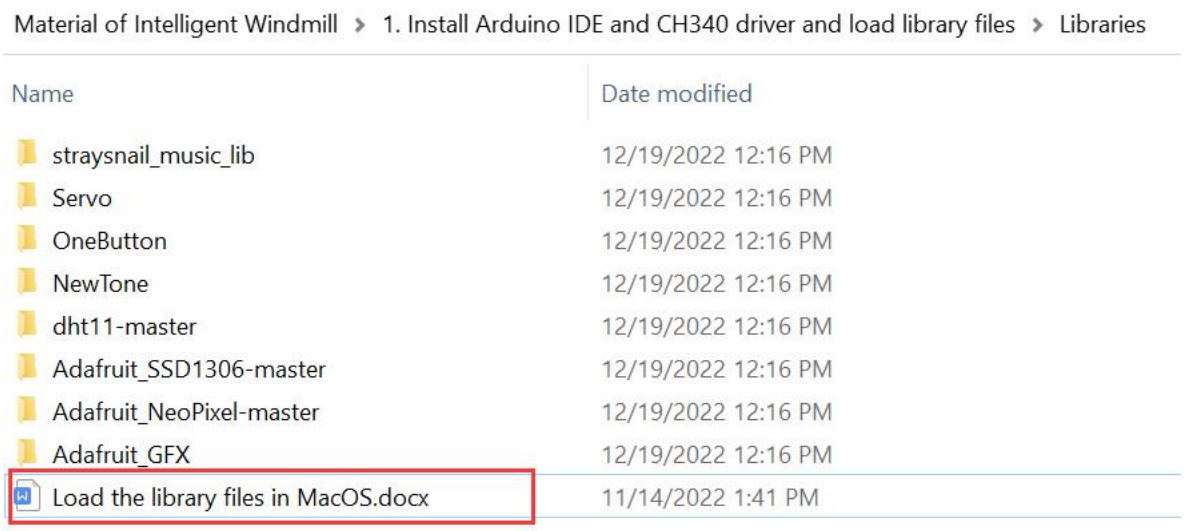


2. Load library file

After installing the software, load the library files provided by us, which will be used in later code learning.

MacOS system loading library:

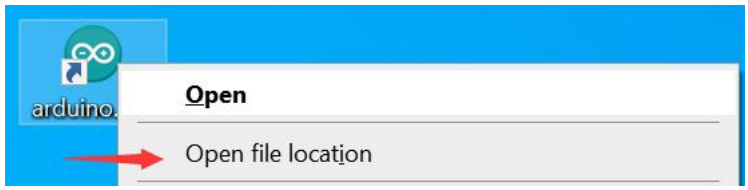
Please open “Load the library files for MacOS”, as shown below.



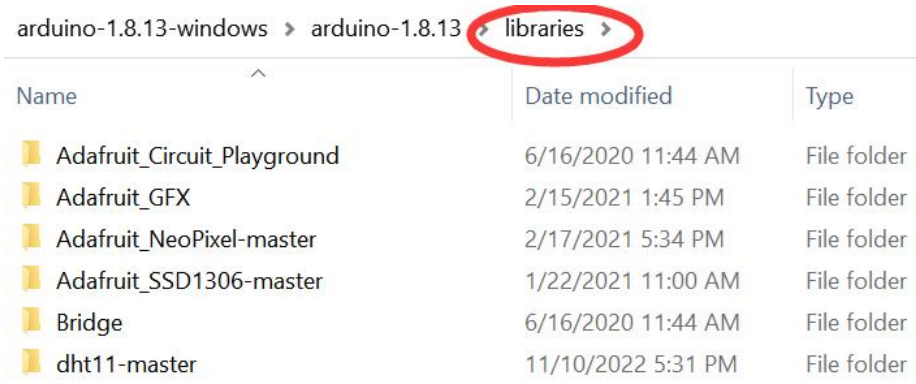
Load the library files for Windows system:

Open the “Intelligent windmill libraries” folder provided by us, copy all library files, and paste them into “libraries” in the Arduino IDE, as shown in the following figure.

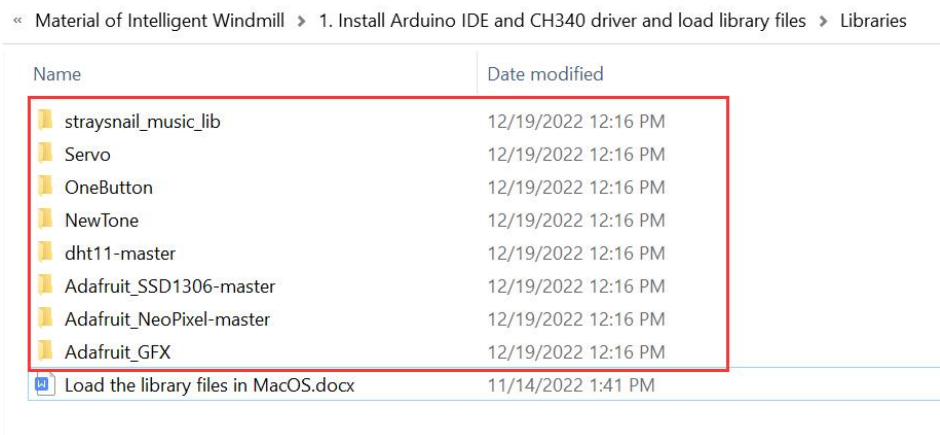
(1) Right click the icon of the Arduino IDE on the desktop and select “Open file location”.



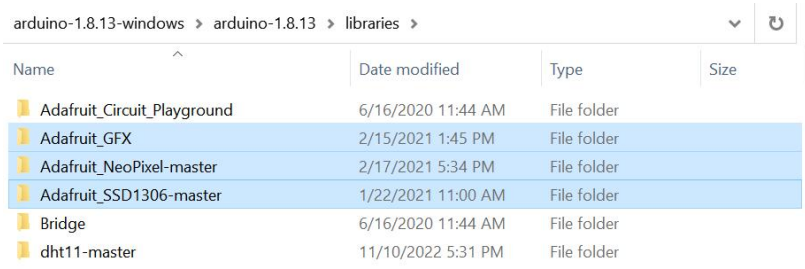
You will enter the installation location of the Arduino IDE and open the “Libraries” folder.



(2) Open the library folder provided by us and copy all the library files.



(3) Paste them into the “libraries” folder of the Arduino IDE you just opened. Done.



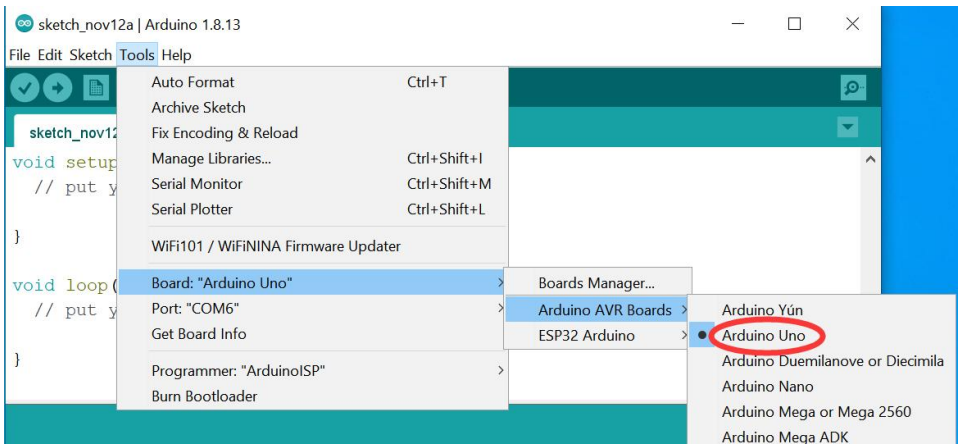
3. Turn on the on-board LED of Arduino IDE

Arduino IDE comes with a lot of example codes. Now we learn to open the code that makes the LED flash in the example and burn it to the Arduino UNO main board.

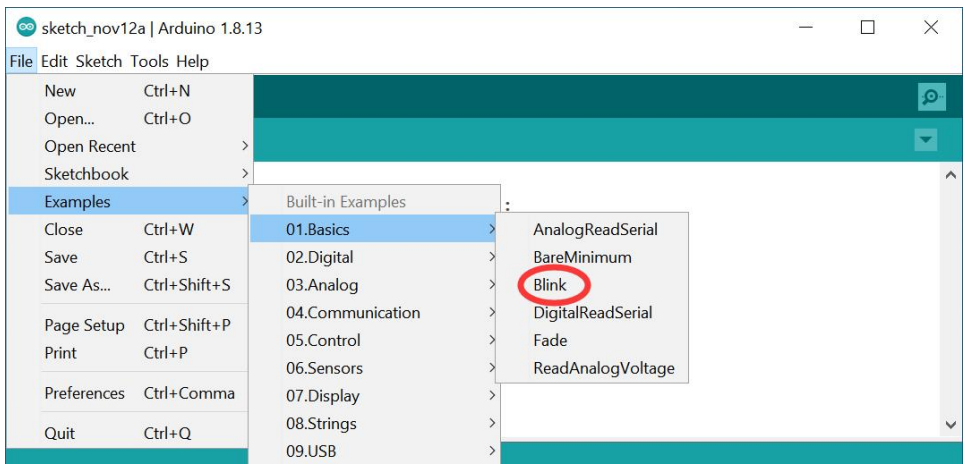
(1) Take out the data cable provided by us, and connect the computer USB and Arduino UNO USB interface.

(2) Open the Arduino IDE and select the main board and COM port in “Tool”

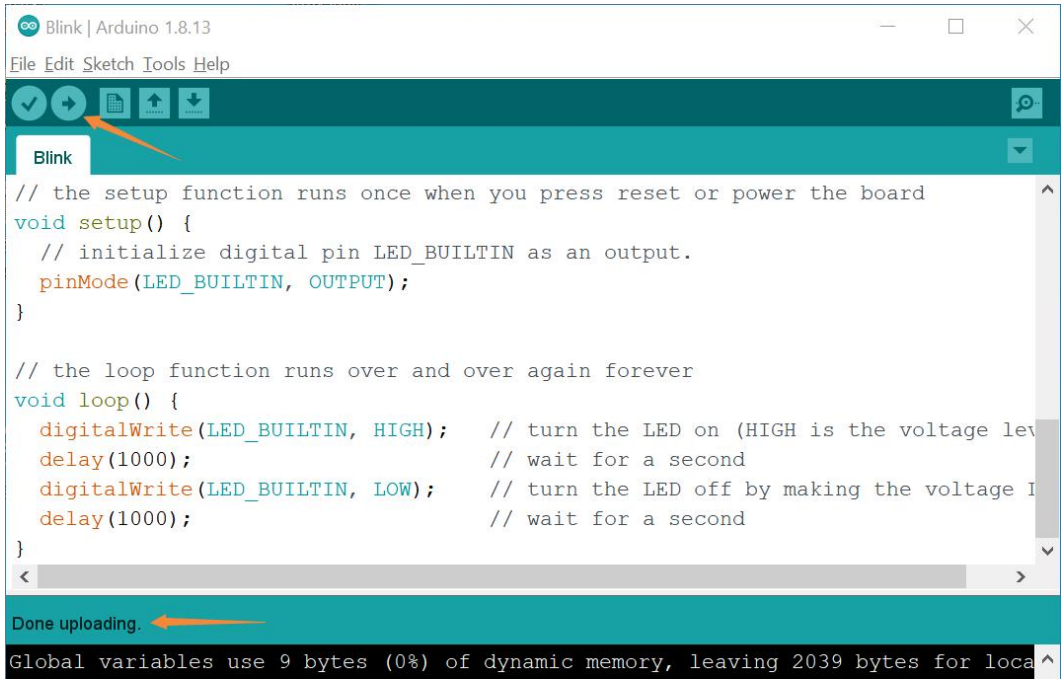
Select “Arduino UNO” as the main board. Pay attention to the COM port. Select the one recognized by your computer, which my computer recognizes is COM4.



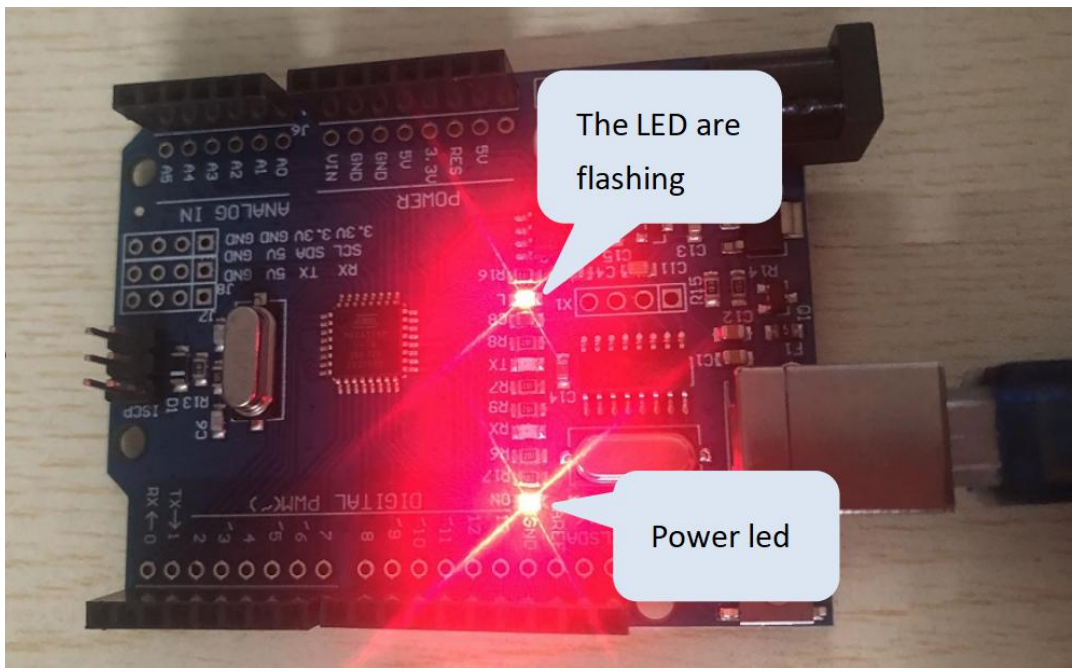
(3) Open the example code coming with the Arduino IDE for turning on the flashing LED.



(4) Click the upload button directly to upload the code to the Arduino UNO main board. If it is successful, you will see that “The upload is successful”.



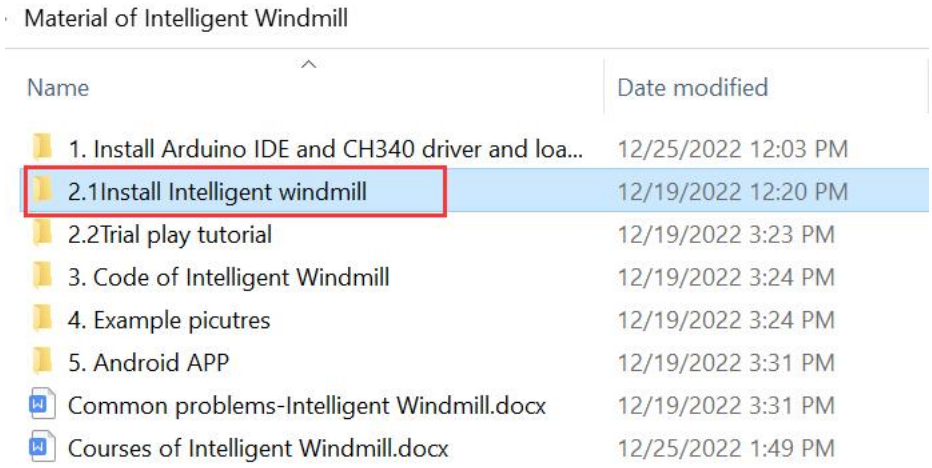
(5) After successful burning, normally you can see the on-board LED of Arduino UNO flashing.



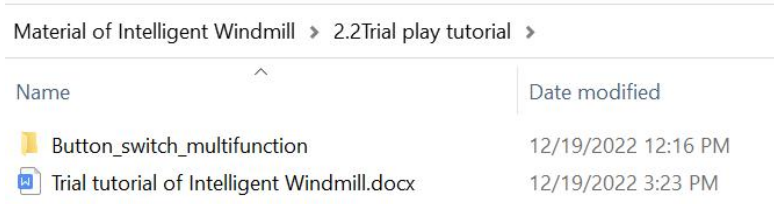
By now, I believe you have a basic understanding of the operation of the Arduino IDE.

Ⅴ. Assemble the Dutch windmill

Please open the tutorial provided by us and follow the steps to install the windmill.



After the installation is finished, you may want to play various functions of the windmill now. Open the trial tutorial document provided by us.

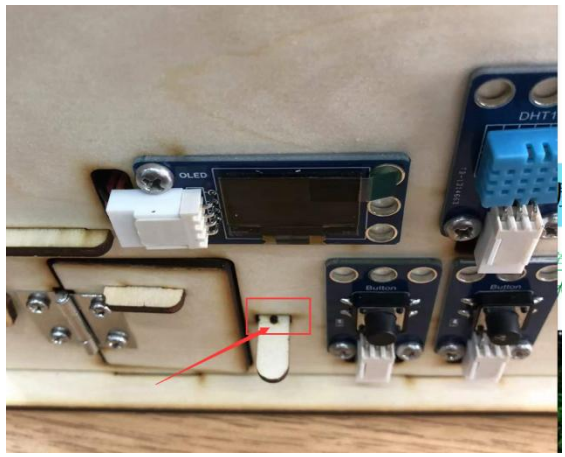


Of course, if you want to learn it step by step and write the code to control the windmill, you can continue to learn code programming.

Ⅵ. Learn to control the windmill

The power switch is turned on by dragging it outward, as shown in the figure below.

Note: The code can be uploaded successfully only when the Bluetooth switch is turned off. The status of Bluetooth off and on is shown in the following figure.



Official Arduino C language learning website:

https://www.arduino.cc/reference/en/

1. Control the LED

In the first lesson, we will learn to control the LED lamp, which is the simplest output module. We can control the LED light through digital output and analog output, and the digital output is “on” and “off”, the analog output can control the brightness of LED light.

Principle of LED lamp: It is composed of light-emitting diode and protective shell. LED can convert electric energy into light energy.

The IO port of main board or expansion	A3

board to connect	

(IO is the input and output port.)

1.1 LED blinking

Use the digital output function to control the LED to flash like a firefly.

(1) Example code

```
/*
create by straysnail
2022/11/11
*/
void setup() {  //Initial function, the program will only run once
  pinMode(3,OUTPUT); //Set pin 3 to output state
}

void loop() {  //Cyclic function, the program will run circularly all the time
  digitalWrite(3,HIGH); //Digital output function, controlling pin 3 output high level
  delay(300); //Delay 300ms
  digitalWrite(3,LOW);  //controlling pin 3 output low level
  delay(300);
}
```

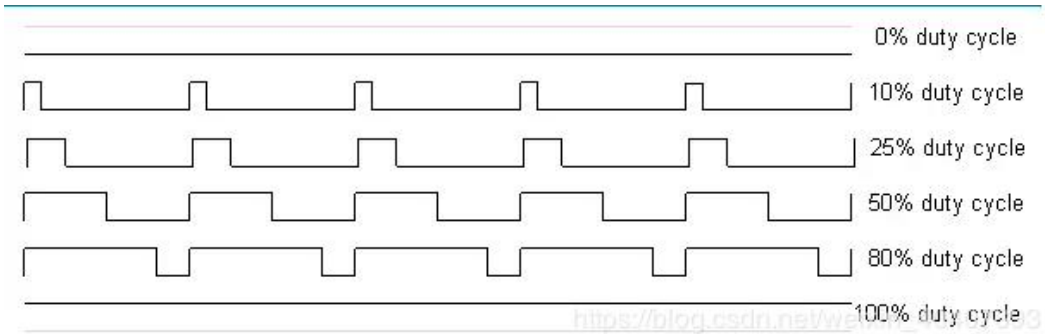
(2) Experimental phenomenon

The LED light in the windmill house flashes at an interval of 300ms.

Tip: *It is recommended to modify the value of the delay function, and then observe whether the flashing frequency of the LED light has changed. Remember to reburn the code after modification.*

1.2 Control the brightness of LED

Use analog output function to control LED to emit light of different brightness.
The principle of analog output controlling LED brightness is the proportion of high level in a certain period of time. The higher the proportion, the greater the brightness, as shown in the following figure. The maximum output voltage of the I/O port of our control board is 5V, so the voltage corresponding to the high level proportion is 0%-100% = 0-5V.



(1) Example code

```
/*
create by straysnail
2022/11/11
*/
void setup() {
  pinMode(3,OUTPUT); //Set pin 3 to output state
}

void loop() {
  analogWrite(3,50); //Analog output function with output value of 50
  delay(300); //Delay 300ms
  analogWrite(3,120); //Analog output function with output value of 120
  delay(300);
  analogWrite(3,200); //Analog output function with output value of 200
  delay(300);
  analogWrite(3,255); //Analog output function with output value of 255
  delay(300);
}
```

(2) Code knowledge

Analog output ports	The analog output ports of Arduino UNO main board are 3、5、6、9、10、11
analogWrite(3,255);	Analog output function, ranging from 0-255

(3) Experimental phenomenon

The LED emits light of 4 kinds of different brightness.

1.3 LED breathing lamp

Control the LED to lighten and darken gradually.

(1) Example code

```
/*
create by straysnail
2022/11/11
*/
void setup() {
  pinMode(3,OUTPUT); //Set pin 3 to output state
}

void loop() {
  //Control LED to light up gradually
  for(int i=0; i<255; i++) {
    analogWrite(3,i); //Analog output function
    delay(10);
  }
  //Control LED to darken gradually
  for(int i=255; i>0; i--) {
    analogWrite(3,i); //Analog output function
    delay(10);
  }
}
```

(2) Code knowledge

for(int i=0; i<255; i++) {
analogWrite(3,i); //Analog output function
delay(10);
	int i=0 is a local variable and the initial value is 0.
	for loop statement:
	for(expression 1; expression 2; expression 3)

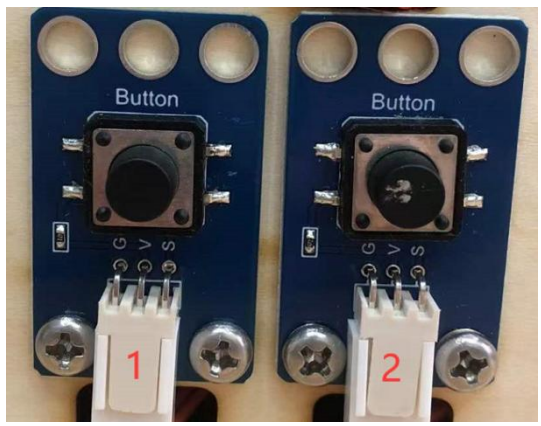
}	{
	Loop body;
	}
	Expression 1 execute first, only once.
	Expression 2 judges. If it is true, the loop body will be executed. If it is false, the loop body will not be executed and the for loop statement will jump out and run down.
	Expression 3 execute after execute the loop body. The expression 3 here is i++, that is , i=i+1.
	Then execute expression 2 to judge whether it is true or false, and continue to loop until it is judged to be false.

(3) Experimental phenomenon

The LED lighten and darken gradually. It circulates all the time, as if the light is breathing.

2. Button beside the door

Principle of button: The reset button is used. Press the button, and after releasing the hand, the button cap will spring up and restore to its original position. It is connected to the circuit as a digital input, and can read two states. When pressed or not, the corresponding values are 0 and 1.



The button	The IO port of main board or expansion board to connect
Left button 1	D2
Right button 2	D13

2.1 Read the state values of the buttons

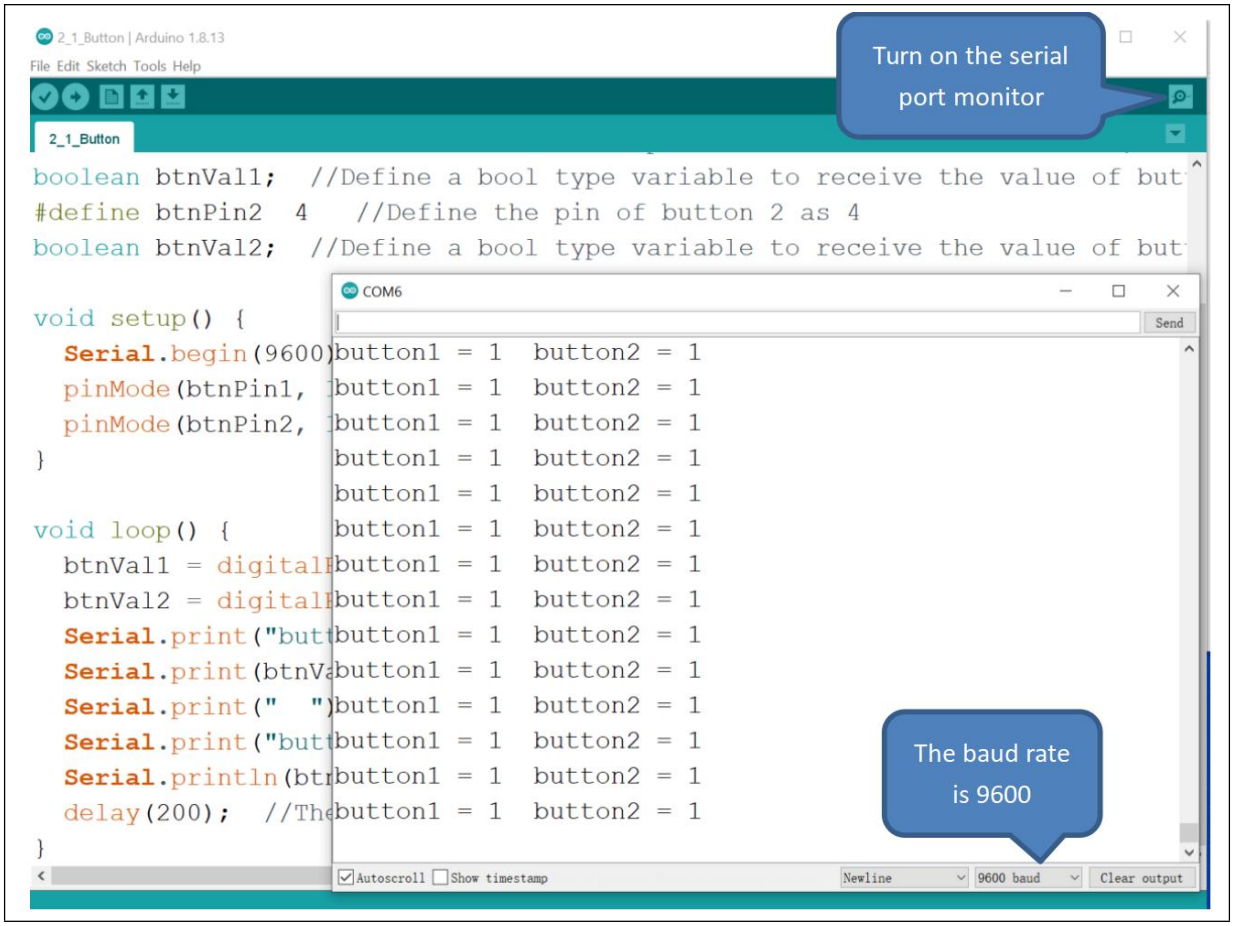
Read the state values when two buttons are pressed and released.

(1) Example code

```
/*
create by straysnail
2022/11/11
*/
#define btnPin1  2   //Macro define the pin name of button 1 as btnPin1, and the
corresponding IO port is 2
boolean btnVal1;  //Define a bool type variable to receive the value of button 1
#define btnPin2  4   //Define the pin of button 2 as 4
boolean btnVal2;  //Define a bool type variable to receive the value of button 2

void setup() {
  Serial.begin(9600);  //Set the baud rate of serial communication to 9600
  pinMode(btnPin1, INPUT); //Set button pin as input type
  pinMode(btnPin2, INPUT);
}

void loop() {
  btnVal1 = digitalRead(btnPin1); //Read the value of the button and assign it to btnVal1
  btnVal2 = digitalRead(btnPin2); //Read the value of the button and assign it to btnVal2
  Serial.print("button1 = "); //Serial.print(); function is serial port printing
  Serial.print(btnVal1); //Print the value of button 1
  Serial.print("  ");
  Serial.print("button2 = ");
  Serial.println(btnVal2);  //Serial.println(); function is serial port newline
printing
  delay(200);  //The delay is used to adjust the printing speed of the serial port
}
```

(2) Experimental phenomenon

Open the serial port monitor of the Arduino IDE and **set the baud rate value to 9600** according to the operation shown in the following figure, and then you can see the values printed in the code. Press the button to observe whether the printed value become 0.



Tip: *Modify the string printed in "Serial. print()" function and observe whether the string you modified will be printed. The serial port monitor is a very useful tool, which you can directly see the printed values, check where the code stops, etc. You should be good at using the serial port monitor.*

2.2 Buttons control the LED lamp

Two buttons, one to control the LED light on, the other to control the LED light off.

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/11
 */
#define ledPin 3  //Define the pin of LED lamp
#define btnPin1 2  //Define the button pin
#define btnPin2 4  //Define the button pin
//The variable is used to receive the value of the button
boolean btnVal1;
boolean  btnVal2;
```

```
void setup() {
  Serial.begin(9600);
  pinMode(btnPin1,INPUT);  //Set the pin to input state
  pinMode(btnPin2,INPUT);  //Set the pin to input state
  pinMode(ledPin, OUTPUT);
}

void loop() {
  btnVal1 = digitalRead(btnPin1);  //Read the value of the button and assign it to btnVal1
  btnVal2 = digitalRead(btnPin2);  //Read the value of the button and assign it to btnVal2
  Serial.println(btnVal2);
  if(btnVal1 == 0)  //If function, and it means that if button 1 is pressed.
  {
    digitalWrite(ledPin, HIGH);  //LED is on
  }
  if(btnVal2 == 0)  //If button 2 is pressed
  {
    digitalWrite(ledPin, LOW);  //LED is off
  }
}
```

(2) Experimental phenomenon

Press the button on the left, the LED lights up, and then press the button on the right, the LED lights out.

Tip: *Try to modify the code by yourself. Click button 2 to turn on the LED, and the click button 1 to turn off the LED.*

2.3 Small table lamp

There are many simple table lamps, when press the button, the table lamp will light up, press again, the table lamp will be turned off.

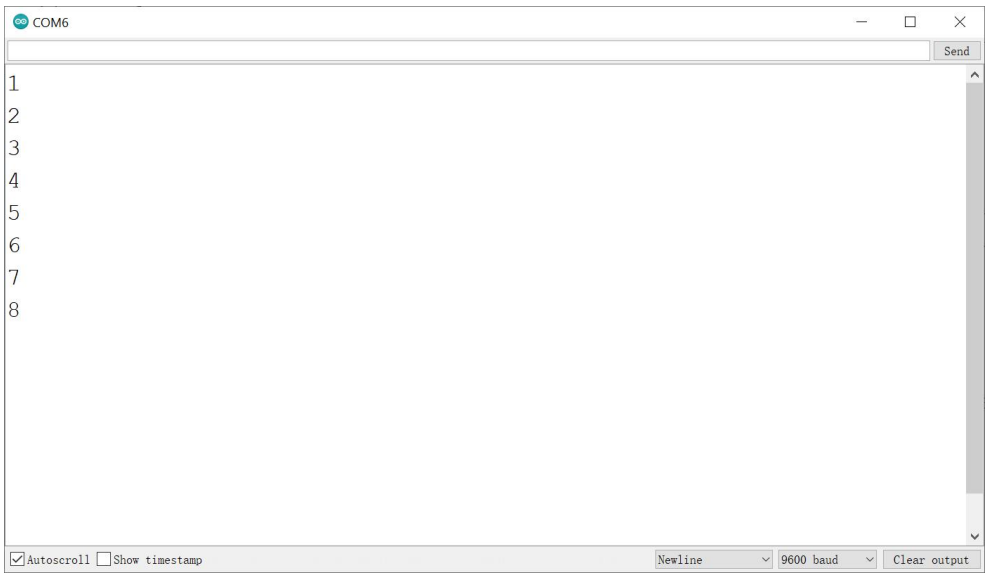
(1) Example code

```
/*
 * create by straysnail
 * 2022/11/26
 */
#define ledPin 3 //Define the pin of LED as 3
#define btnPin 2  //Define the button pin as 2
boolean btnVal;   //Variable, used to receive the value detected by the button
int count = 0; //Variable, used to record the number of times the button is clicked
boolean flag = 0; //Used to switch the state of button press and release
int data;

void setup() {
  Serial.begin(9600);
  pinMode(btnPin,INPUT);  //Set the button pin to input state
}

void loop() {
  btnVal = digitalRead(btnPin);  //Read the value of the button and assign it to btnVal
  if(btnVal == 0)  //If function, and it means that if the button is pressed
  {
    delay(20); //Delay to eliminate the shake of the button
    btnVal = digitalRead(btnPin);
    if(btnVal == 0)
    {
      flag = 1;
      while(flag == 1) //While cyclic function. The cycle will not exit until the button is
released and flag==0
      {
        btnVal = digitalRead(btnPin); //Detect state of the button again
        if(btnVal == 1)  //Judge. If the button is released
        {
          count = count + 1; //Record the number of times the button is clicked
          Serial.println(count);  //Print the number of times the button is clicked
          flag = 0; //Exit the state of button pressing
        }
      }

      data = count % 2;  //Calculate the remainder of the number of clicks on the button to
2. If it is singular, data is equal to 1. If it is even, data is equal to 0
      if(data == 1) //If it is singular
```

```
        {
          digitalWrite(ledPin, HIGH);  //LED is on
        }
        else
        {
          digitalWrite(ledPin, LOW);  //LED is off
        }
      }

    }
}
```

(2) Experimental phenomenon

Press the button on the left once, and the LED lights up. Press the button again, and the LED lights off. Open the serial port monitor of the Arduino IDE, click the button, and you can see that the serial port is printing the number of times the button was clicked.



Tip: *In the example code, you can calculate the number of times a button was pressed, and then calculate the remainder of 2 to get 0 or 1. This method is very common. Try writing it yourself several times to get familiar with this usage.*

3. Use of buzzer

Principle of passive buzzer: The passive buzzer uses electromagnetic induction to drive the diaphragm to make sound when the electromagnet forms after the voice coil is connected to the alternating current to attract or repel the permanent magnet. It can only make sound when it is connected or disconnected. That is to say, you need to control the frequency of the high and low levels of output to produce sound, and control the frequency to produce

different tones.

The IO port of main board or expansion board to connect	D11

3.1 Buzzer sounds

We control the output frequency by controlling the high and low levels and the delay function to make the buzzer sound.

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/11
 */

#define buzPin  11  //Define the pin of the buzzer

void setup() {
  // put your setup code here, to run once:
  pinMode(buzPin, OUTPUT); //Set to output state
}

void loop() {
  //Set the frequency of the buzzer by delay
  digitalWrite(buzPin, HIGH);
  delayMicroseconds(500);  //Microsecond delay. Within a certain range, adjust the delay, and the buzzer will make different sounds
  digitalWrite(buzPin, LOW);
  delayMicroseconds(500);
}
```

(2) Experimental phenomenon

The buzzer sends out a piercing “Di……”sound.

Tip: *Modify the value of the subtle delay function "delayMicroseconds(500)", and listen to the tones generated by the frequencies of different delay values.*

3.2 Use Tone function to control buzzer

It is too troublesome to use high and low level and delay functions to control the tone, so Arduino officially wrote the Tone function using timers, which is convenient for us to write some songs and make music boxes.

(1) Example code

```
#define buzPin 11  //Set the pin of the buzzer
void setup(){
  pinMode(buzPin, OUTPUT); //Set to output state
}

void loop(){
  tone(buzPin,262);  //do
  delay(500);
  tone(buzPin,294);  //re
  delay(500);
  tone(buzPin,330);  //mi
  delay(500);
  tone(buzPin,349);  //fa
  delay(500);
  tone(buzPin,392);  //so
  delay(500);
  tone(buzPin,440);  //la
  delay(500);
  tone(buzPin,494);  //si
  delay(500);
  tone(buzPin,532);  //do
  delay(500);
  noTone(buzPin);     //Stop making sound
  delay(1000);
```

```
}
```

(2) Experimental phenomenon

The buzzer emits the sound of “do re mi fa so la si do”.

3.3 A song

Because the Arduino official Tone function uses timers, while our electronic modules are quite numerous, which will affect the centralized use of multiple functions, we use the “NewTone” library file, which is the same as Tone, but doesn’t need timers. Let’s use the “NewTone” function to write a piece of music.

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/11
 */

#include <NewTone.h> //Import a library file of sound frequencies
#define buzzerPin 11   //buzzer PIN
#define btnPin 2  //button PIN

void setup() {
  Serial.begin(9600);
  pinMode(buzzerPin, OUTPUT);
  pinMode(btnPin, INPUT);
}

void loop() {
  boolean val = digitalRead(btnPin);
  if(val == 0)
  {
    play_music(buzzerPin);  //Execute the subfunction and play the tune
  }
  else
  {
    NewNoTone(buzzerPin);  //stop
  }
}

void play_music(int buzzer_pin)
{
```

```
  //NewTone Three parameters: PIN , frequency, duration
  NewTone(buzzer_pin,294,250);
  NewTone(buzzer_pin,440,250);
  NewTone(buzzer_pin,392,250);
  NewTone(buzzer_pin,532,250);
  NewTone(buzzer_pin,494,500);
  NewTone(buzzer_pin,392,250);
  NewTone(buzzer_pin,440,250);
  NewTone(buzzer_pin,392,250);
  NewTone(buzzer_pin,587,250);
  NewTone(buzzer_pin,532,500);
  NewTone(buzzer_pin,392,250);
  NewTone(buzzer_pin,784,250);
  NewTone(buzzer_pin,659,250);
  NewTone(buzzer_pin,532,250);
  NewTone(buzzer_pin,494,250);
  NewTone(buzzer_pin,440,250);
  NewTone(buzzer_pin,698,375);
  NewTone(buzzer_pin,659,250);
  NewTone(buzzer_pin,532,250);
  NewTone(buzzer_pin,587,250);
  NewTone(buzzer_pin,532,500);
  NewNoTone(buzzer_pin);
}
```

(2) Experimental phenomenon

Click the button on the left, and the buzzer will play a song of Happy Birthday. When it finishes, you can click to play it again.

Tip: *If you want to write your favorite music, you need to know some theoretical knowledge of music, Understanding the frequency of each tone.*

4. Light-operated lamp

In some residential areas, the lights will be turned on automatically at night. Many of them are realized by using photosensitive sensors. As long as the measured light is less than a certain value, the lights will be turned on automatically. We use the photosensitive sensor and the LED lamp of the windmill to achieve this function.

Photosensitive sensor: The photosensitive sensor installed on the windmill is an analog sensor, which are mainly made according to the characteristics of the photosensitive resistor. The resistance of the photosensitive resistor will change with the change of the light intensity, that is, within a certain range, the stronger the light, the greater the measured value.

The IO port of the left photosensitive sensor	A1

4.1 Read the values of the photosensitive sensor

Print the value measured by the photosensitive sensor in the serial port monitor.

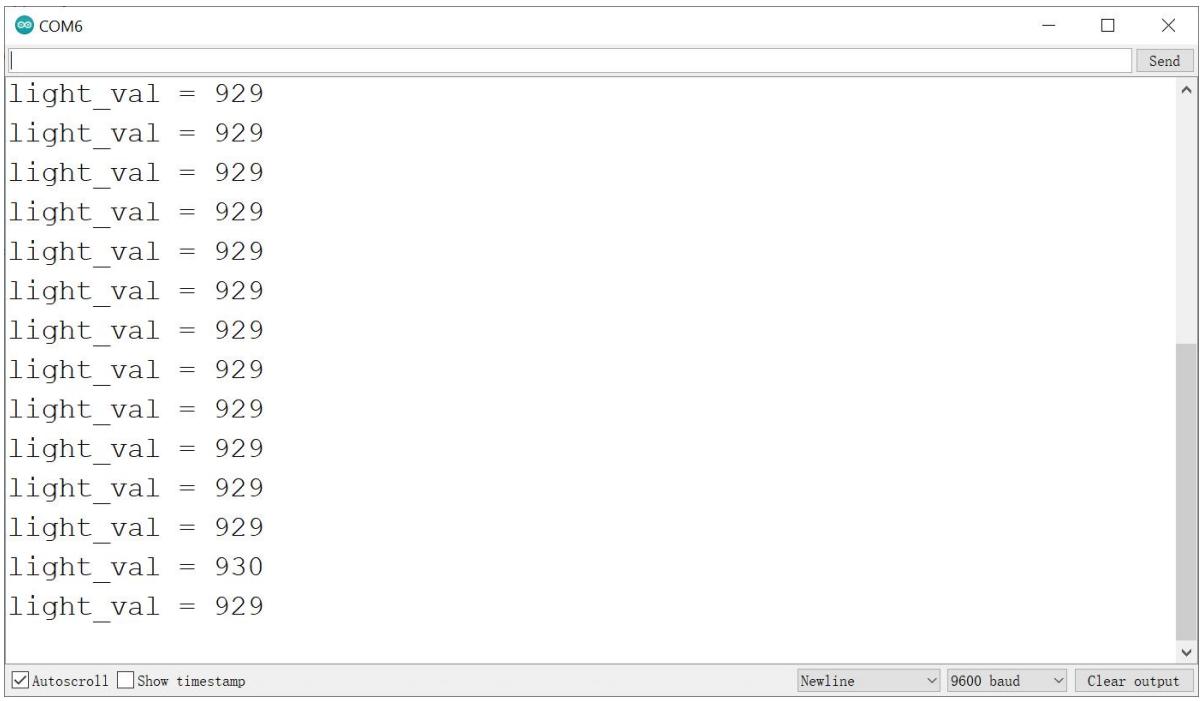
(1) Example code

```
/*
 * create by straysnail
 * 2022/11/14
 */
#define lightPin A1  //Define the pin of photosensitive sensor
int light_val;

void setup() {
  Serial.begin(9600);
  pinMode(lightPin, INPUT);  //Set to input state
```

```
}

void loop() {
  light_val = analogRead(lightPin);  //Read the analog value
  Serial.print("light_val = ");
  Serial.println(light_val);
  delay(100);
}
```

(2) Experimental phenomenon

Open the serial port monitor and check the value detected by the photosensitive sensor.



4.2 Light-operated lamp

When the photosensitive sensor detects that the surrounding environment is dark, the LED will turn on automatically.

(1) Example code

```
#define lightPin A1
#define LEDPin 3

int light_val;

void setup() {
```

```
  Serial.begin(9600);
  pinMode(lightPin, INPUT);
  pinMode(LEDPin, OUTPUT);
}

void loop() {
  light_val = analogRead(lightPin);
  Serial.print("light_val = ");
  Serial.println(light_val);
  delay(100);
  if(light_val < 500)  //When it is lower than the set value, the LED will light up.
  {
    digitalWrite(LEDPin, HIGH);
  }
  else
  {
    digitalWrite(LEDPin, LOW);
  }
}
```

(2) Experimental phenomenon

When the photosensitive sensor is covered by hand or in a dark place, the LED will light up.

Little task: *The brightness of LED changes with the size of the value detected by the photosensitive sensor. The larger the value, the lower the brightness.*

5. Intelligent stair lamp

The intelligent stair lamp will light up only when the environment is dark and someone passes by, which can not only illuminate, but also save energy and protect the environment. The stair lights in many residential areas have the function of automatically turning on the light.

Now we use the modules in the windmill to realize the intelligent stair lamp.
The modules used are: Human body infrared pyroelectric sensor, photosensitive sensor, LED lamp.
Human infrared pyroelectric sensor: It is a digital sensor, which has only two status values, that is, only detected person and not detected person.
The central wavelength of the infrared ray radiated by human body is 9-10 um, while the wavelength sensitivity of the detecting element is almost stable in the range of 0.2-20 um. A window with a filter lens is set at the top of the sensor. The wavelength range of the light that can pass through the filter is 7-10 um, which is just suitable for the detection of human infrared radiation, while the infrared of other wavelengths is absorbed by the filter, thus forming an infrared sensor specially used to detect human radiation.
Note: It can only be detected when someone moves in front of the sensor.

The IO port of the left photosensitive sensor	A3

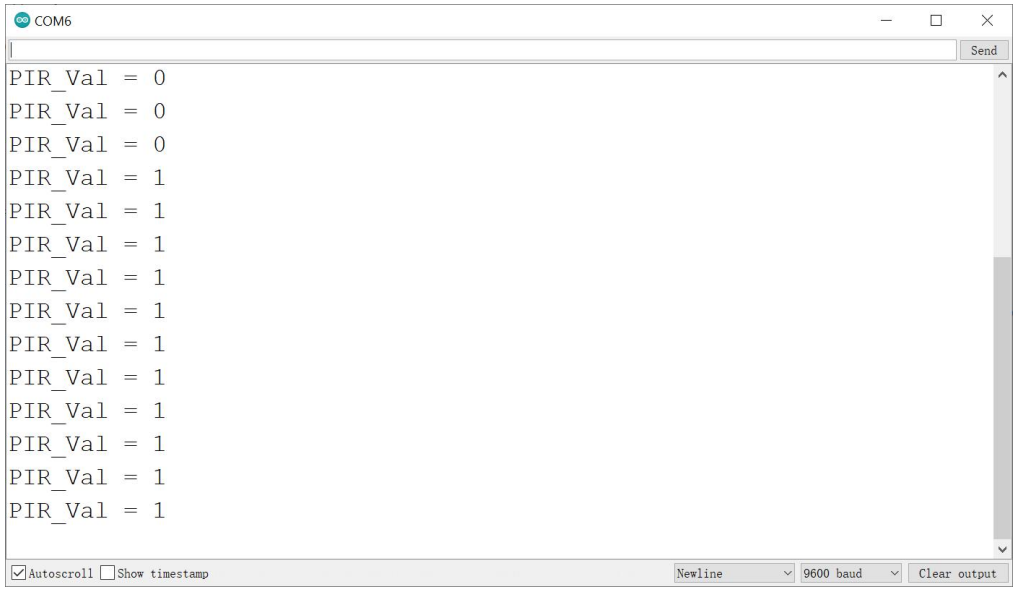
5.1 Read the value of human infrared pyroelectric sensor

(1) Example code

```
/*
create by straysnail
2022/11/14
*/
#define PIR_pin  A3   //Define the pin of pyroelectric sensor
boolean PIR_Val;  //Define a variable of bool type

void setup() {
  Serial.begin(9600);  //Set the baud rate of serial port communication as 9600
  pinMode(PIR_pin, INPUT); //Set the pin of the button to input
}

void loop() {
  PIR_Val = digitalRead(PIR_pin); //Read the button value and assign to PIR_Val
  Serial.print("PIR_Val = ");
  Serial.println(PIR_Val);
  delay(200);  //The delay is used to adjust the printing speed of the serial port
}
```

(2) Experimental phenomenon

Turn on the serial port monitor and move your hand in front of the sensor. It can be seen that the value is 1 when the human infrared pyroelectric sensor detects someone is moving, and it is 0 when no one is moving.



5.2 Simulate the intelligent stair lamp

When photosensitive sensor detects it is dark, the LED lights up when the human body infrared pyroelectric sensor detects a person.

(1) Example code

```
/*
create by straysnail
2022/11/14
*/
#define lightPin A1
#define LEDPin 3
#define PIR_pin  A3   //Define the pin of pyroelectric sensor
boolean PIR_Val;  //Define a variable of bool type
int light_val;

void setup() {
  Serial.begin(9600);  //Set the baud rate of serial port communication as 9600
  pinMode(PIR_pin, INPUT); //Set the pin of the button to input
  pinMode(lightPin, INPUT);
  pinMode(LEDPin, OUTPUT);
}
```

```
void loop() {
  light_val = analogRead(lightPin);
//  Serial.print("light_val = ");
//  Serial.println(light_val);
  if(light_val < 500) {   //When it is dark
    PIR_Val = digitalRead(PIR_pin); //Read the button value and assign to PIR_Val
    Serial.println("evening");
    if(PIR_Val == 1) {  //Person detected
      digitalWrite(LEDPin, HIGH);  //The LED lights up
      Serial.println("someone");
    } else {   //No person detected
      digitalWrite(LEDPin, LOW);  //The LED turns off
      Serial.println("nobody");
    }
  }
  else {  //When it is daytime
    digitalWrite(LEDPin, LOW);  //The LED turns off
    Serial.println("morning");
  }
}
```

(2) Experimental phenomenon

First cover the photosensitive sensor with one hand, which means that it is dark. Then move the other hand in front of the human body infrared pyroelectric sensor , and you can see that the LED is on. If no one is detected or the hand covering the photosensitive sensor is removed, the LED will turn off.

6. Laser and DOE

Laser： Laser, whose full name is “Light Amplification by Stimulated Emission of Radiation”, was discovered by Einstein in 1916. Its principle is that when electrons in atoms absorb energy and then transition from low energy level to high energy level, and then fall back to low energy level, the released energy is in the form of photons, and the characteristics of emitted photons are highly consistent. Therefore, laser has 4 characteristics: high brightness, high directivity, high monochromaticity and high coherence.

According to the type of gain medium, lasers can be divided into liquid lasers, gas lasers, semiconductor lasers and solid lasers. The red laser used in this intelligent windmill is a semiconductor laser, which has the advantages of small size, good reliability, long life and low power consumption. It is most widely used laser type.

Laser is widely used, including laser projector, laser cutting, optical fiber communication, laser radar, laser beauty, laser scanning, etc.

Laser cutting machine

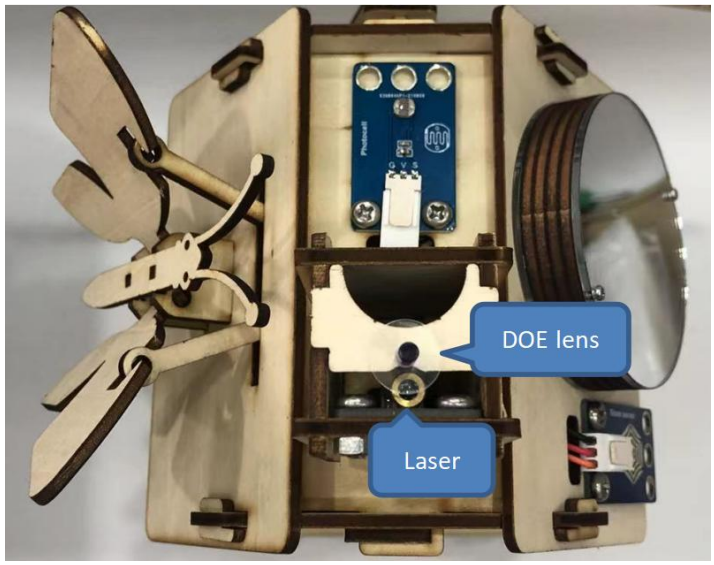
Laser projector

DOE: The full name of DOE is Diffractive Optical Elements. Diffraction units of micron or even nanometer level are formed on the substrate material by etching or laser direct writing process. Each diffraction unit can have a specific shape, refractive index, etc, to fine control the wavefront phase distribution of the laser. The laser diffracts after passing through each diffraction unit, and interferes after a certain distance to form a specific light intensity distribution. It is mainly used for laser beam shaping, such as homogenization, collimation, focusing, forming specific patterns, etc. It plays an important role in laser processing, medical treatment, imaging, remote sensing, pattern shaping and other fields.

Most DOEs used today are digital optical elements. They were first developed by Lincoln Laboratory of MIT in the 1970s, and then commercialized by Holo/Or Company in Israel in the 1990s.

Its advantages are small size, high damage threshold, simple use, etc.

The DOE used in the intelligent windmill belongs to the pattern shaping class. After passing through a specially designed diffraction unit, the laser beam is adjusted into Santa Claus, candy cane and other patterns.



6.1 Control the laser and DOE

Turn on the laser to irradiate the DOE and project the patterns.

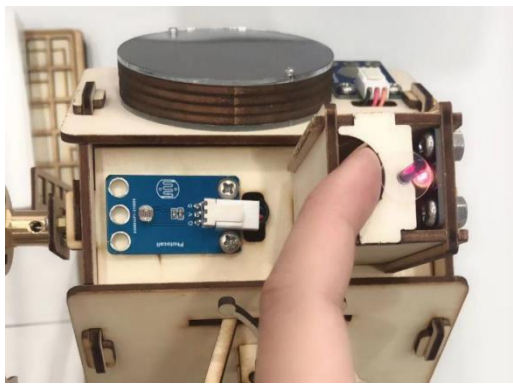
(1) Example code

```
/*
 * create by straysil
 * 2022/11/25
 */
#define laserPin 13 //Define the pin of the laser
#define btnPin 2  //Define the pin fo the button
boolean btnVal;   //The variable is used to receive the value detected by the button
int count = 0; //The variable is used to record the number of times the butotn is clicked
boolean flag = 0; //The variable is used to switch the state of button press and release
int data;

void setup() {
  Serial.begin(9600);
  pinMode(btnPin,INPUT);  //Set the pin of the button to input
}

void loop() {
  btnVal = digitalRead(btnPin);  //Read the button value and assign it to btnVal
  if(btnVal == 1)  //Judge if the button is pressed
  {
    delay(10); //Delay to eliminate the shake of the button
    if(btnVal == 1)  //Judge if the button is pressed
    {
      flag = 1;
      while(flag == 1) //While cyclic function. The cycle will not exit until the button is
released and flag==0
      {
        btnVal = digitalRead(btnPin); //Detect state of the button again
        if(btnVal == 0)  //Judge If the button is released
        {
          count = count + 1; //Record the number of times the button is clicked
          Serial.println(count);  //Print the number of times the button is clicked
          flag = 0; //Exit the state of button pressing
        }
      }

      data = count % 2;  //Calculate the remainder of the number of clicks on the button to
```

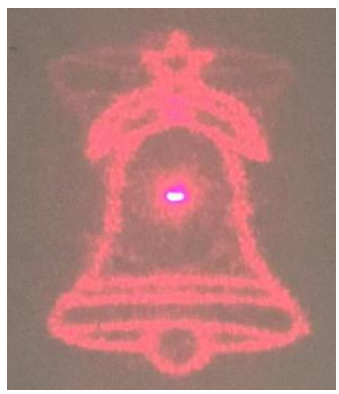
```
2. If it is singular, data is equal to 1. If it is even, data is equal to 0
      if(data == 1) //If it is singular
      {
        digitalWrite(laserPin, HIGH);  //The laser turns on
      }
      else
      {
        digitalWrite(laserPin, LOW);  //The laser turns off
      }
    }
  }
}
```

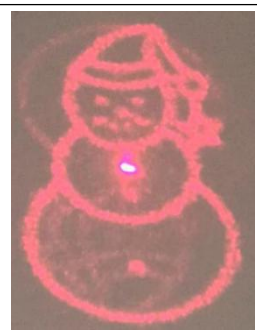
(2) Experimental phenomenon

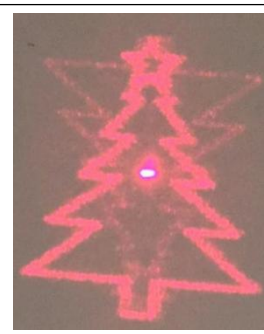
Click button 1 to turn on the laser, and then you can see the projected pattern on the near plane. Rotate the DOE to switch patterns. There are 6 patterns in total.



Rotate the DOE



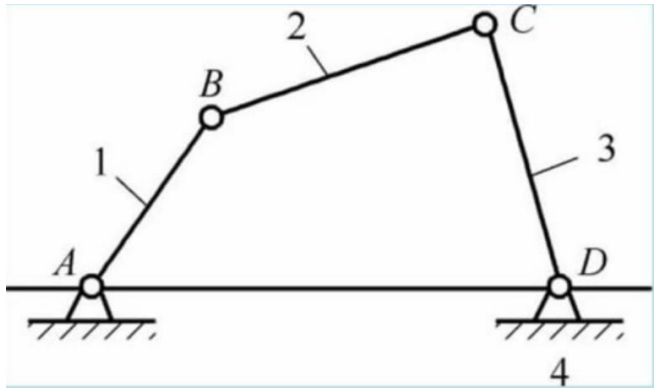




7. Automatic door and window

The power sources of the automatic door and window are the steering gears, which can rotate nearly 180 degrees. The transmission structure principle of the door and window is

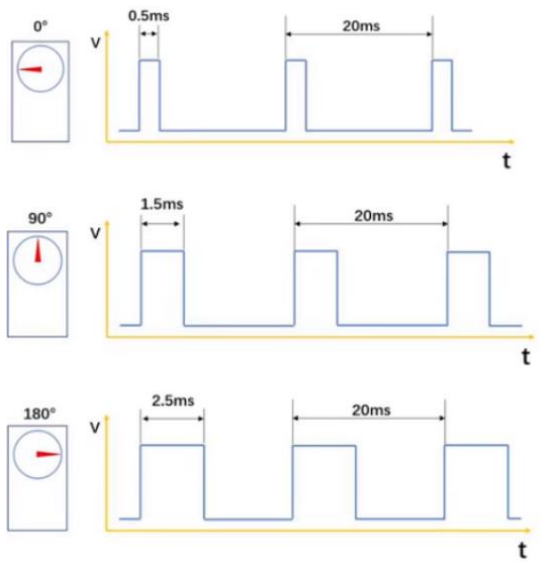
hinge four-bar mechanism.
The basic form of hinged 4 bar mechanism:



There are three basic forms of hinge four-bar mechanism:

- Crank rocker mechanism
- Double crank mechanism
- Double rocker mechanism

The transmission form of our door and window is double rocker mechanism.
Steering gear: Steering gear is a kind of servo driver, which can control the rotation angle accurately. It is widely used and commonly used in the joint movement of robots.
Control principle of steering gear: The angle of the steering gear is determined by the duration of the control signal pulse, which is called pulse code modulation(PCM). The control of the steering gear generally requires a time base pulse of about 20ms. The high level time in the time base pulse is within the range of 0.5ms~2.5ms, and the corresponding angle for controlling the steering gear is 0~180 degrees, as shown in the figure below.



Steering gear	The IO port of main board or expansion board to connect
Steering gear for the door	D9
Steering gear for the window	D10

7.1 Steering gear rotation

Note: As the steering gears of the door and window are limited by the rotation angle, we require the steering gear to be turned to 90 degrees before installation. If you do not control the steering gears to 90 degrees and then install them, please reinstall, otherwise it is easy to damage the steering gears and door and window.

Here we write code to control the steering gear according to the steering gear control principle mentioned above, so as to open and close the door.

Steering angle for opening and closing the door:

Angle for opening the door: 130	Angle for closing the door: 50

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/24
 */
#define door_servoPin 9  //Set the pin of the steering gear
int pulsewidth = 0; //Variable: used to calculate the pulse value of the steering gear

void setup() {
  pinMode(door_servoPin,OUTPUT); //Set the pin to output state
}

void loop() {
  procedure(door_servoPin, 130);  //Turn the steering gear to 130 degrees and open the door
  delay(500);       //Time for the steering gear to turn
  procedure(door_servoPin, 50);  //Turn the steering gear to 50 degrees and close the door
  delay(500);
}

//Function to control the angle of steering gear according to the principle of steering fear
void procedure(int serPin, int myangle)   //There are 2 parameters of the function, serPin is the pin of the steering gear, and myangle is the angle of the steering gear
{
```

```
  for(int i=0; i<10; i++)
  {
    //Calculate the pulse value, that is, the high level time. The range of myangle is 0-180, and the corresponding pulse width range is 500-2480
    pulsewidth = myangle * 11 + 500;
    digitalWrite(serPin,HIGH);
    delayMicroseconds(pulsewidth);  //Delay calculated high level time
    digitalWrite(serPin,LOW); //Set to low level
    delay((20 - pulsewidth / 1000));  //Delay the remaining low level time
  }

}
```

(2) Experimental phenomenon

The door keeps opening and closing.

Note: Due to the deviation of steering gear and installation, the closing angle of your machine may not be 50 degrees. Adjust the closing angle yourself.

Tip: *"void procedure(int serPin, int myangle)" is a sub function in the code. Sub functions are often reused in code. Just write a sub function and call it again, which will make the code look concise. Therefore, be familiar with the use of sub functions.*

7.2 Use of Servo steering gear library files

Using servo steering gear library files will make the code more concise and simple, and the internal timer is used to keep the steering gear at the specified angle.

In this lesson, we learn to control the opening and closing of the window.

Steering angle for opening and closing the window:

Angle for opening the window: 50	Angle for closing the window:： 130

(1) Example code

```
#include <Servo.h>  //Import the library files of steering gear
Servo windowServo;  //Create a steering gear instance to control a steering gear

void setup() {
  windowServo.attach(10,500,2500);  //Set the pin of the steering gear as 10, and the pulse range is 500-2500ms
}

void loop() {
  windowServo.write(50);   //Turn the steering gear to 50 degrees and open the window
```

```
  delay(500);   //Delay for the steering gear to turn
  windowServo.write(130);  //Turn the steering gear to 130 degrees and close the window
  delay(700);
}
```

(2) Experimental phenomenon

The window keeps opening and closing.

Note: Due to the deviation of steering gear and installation, the closing angle of your machine may not be 50 degrees. Adjust the closing angle yourself.

Thinking: *How to control the speed of opening and closing the door and window?*

7.3 Control the door and window to open and close

In the last two lessons, we realized the control of the door and window opening and closing, but there was no interaction. In this lesson, simple interactive actions are added. The door is controlled by button 1, and the window is controlled bu button 2.

(1) Example code

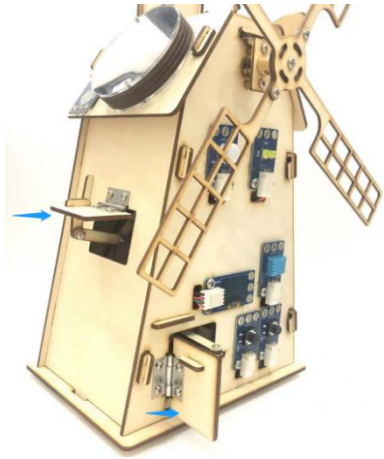
```
/*
 * create by straysnail
 * 2022/2/26
 */
#include <Servo.h>  //Import the library files of steering gear
Servo door_servo;  //Create a steering gear instance to control the door steering gear
Servo window_servo;  //Create a steering gear instance to control the window steering gear
#define btnPin1 2 //IO port of the button
#define btnPin2 4
boolean btnVal1 = 0; //Used to receive the button value
boolean btnVal2 = 0;
int count1 = 0;  //Calculate the click times of the button
int count2 = 0;
boolean door_state = 0; //Control the door state
boolean window_state = 0; //Control the window state

void setup() {
  Serial.begin(9600);
  pinMode(btnPin1,INPUT);
  pinMode(btnPin2,INPUT);
  door_servo.attach(9,500,2500);  //Set the pin of the steering gear as 9, and the pulse range is 500-2500ms
  window_servo.attach(10,500,2500);  //Set the pin of the steering gear as 10, and the pulse range is 500-2500ms
```

```
  door_servo.write(50); //Close the door
  delay(300);
  door_servo.write(130);   //Open the door
  delay(500);   //Delay for the steering gear to turn
  window_servo.write(130); //Close the window
  delay(300);
  window_servo.write(50);  //Open the window
  delay(500);
  door_servo.write(50); //Close the door
  delay(300);
  window_servo.write(130); //Close the window
  delay(300);
}

void loop() {
  btnVal1 = digitalRead(btnPin1);  //Read the value of the button
  btnVal2 = digitalRead(btnPin2);
  if(btnVal1 == 0)  //If button 1 is pressed
  {
    delay(10); //Eliminate wrong judgement caused by the shake of the button
    if(btnVal1 == 0)
    {
      boolean i1 = 1;
      while(i1 == 1)
      {
        btnVal1 = digitalRead(btnPin1);
        if(btnVal1 == 1)  //Button 1 is released
        {
          count1 = count1 + 1; //Record the click times
          Serial.print("button1 = ");
          Serial.println(count1);
          door_state = count1 % 2; //Calculate the remainder. If it is singular, the value
is equal to 1, and if it is even, the value is equal to 0
          if(door_state == 1)
          {
            door_servo.write(130); //Open the door
            delay(300);
          }
          else
          {
            door_servo.write(50); //Close the door
            delay(300);
```

```
          }
          i1 = 0; //Exit the cycle with i = 0
        }
      }
    }
  }

  if(btnVal2 == 0)  //If button 2 is pressed
  {
    delay(10);
    if(btnVal2 == 0)
    {
      boolean i2 = 1;
      while(i2 == 1)
      {
        btnVal2 = digitalRead(btnPin2);
        if(btnVal2 == 1)  //Button 2 is released
        {
          count2 = count2 + 1; //Record the click times
          Serial.print("button2 = ");
          Serial.println(count2);
          window_state = count2 % 2; //Calculate the remainder. If it is singular, the value
is equal to 1, and if it is even, the value is equal to 0
          if(window_state == 1)
          {
            window_servo.write(50); //Open the window
            delay(300);
          }
          else
          {
            window_servo.write(130); //Close the window
            delay(300);
          }
          i2 = 0; //Exit the cycle with i = 0
        }
      }
    }
  }
}
```

(2) Experimental phenomenon

If you have initialized the steering gears when install the door and window, the phenomenon is to open the door, open the window, close the door and close the window. Then click button 1 once to open the door, and click again to close the door. Click button 2 once to open the window, and click again to close the window.



Tip: *The code also uses the method of calculating the number of key clicks and then finding the remainder of 2 to get 0 and 1, which should be consolidated.*

8. Raindrop sensor

Principle of rain drop sensor: It is an analog input sensor. The more water drops cover in the detection area, the greater the measured value. It can be applied to smart farms, automatic clothes collection and window closing devices in rain.

The IO port of main board or expansion board to connect	A2

8.1 Read the value of the raindrop sensor

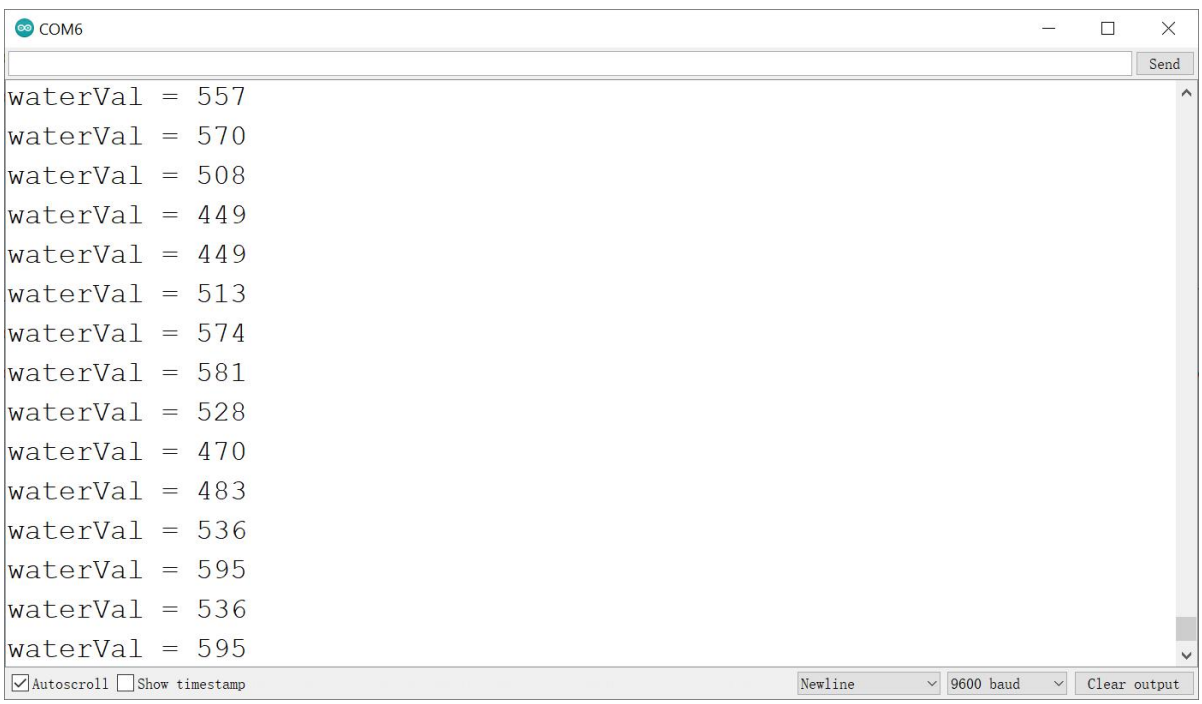
The serial port prints the value detected by the raindrop sensor.

(1) Example code

```
/*
 * create by straysnail
 * 2022/2/26
 */
#define waterPin A2  //Define the IO port of the raindrop sensor as A2
int waterVal = 0;

void setup() {
  Serial.begin(9600);  //Set the baud rate of serial communication to 9600
  pinMode(waterPin,INPUT);  //Set the pin to input
}

void loop() {
  waterVal = analogRead(waterPin);  //Read the analog value detected by the raindrop sensor
  Serial.print("waterVal = ");
  Serial.println(waterVal);  //Print the value detected by the raindrop sensor with new line
  //delay(200); //Delay to adjust the printing speed of serial port
}
```

(2) Experimental phenomenon

Open the serial port monitor of Arduino IDE, and you can see that the value detected by the raindrop sensor is printed out. Touch the detection area with your hand, and it is found that the value will become larger because there is water on your hand skin.



8.2 Experiment of window closing when it rains

When I'm outside, I find it is raining. Then I remember that the clothes are hanged outside and the window is not closed. This is a troublesome thing. I think it may be better if I can automatically get the laundry and close the window. Now we do the experiment of window closing automatically when it rains. The principle is very simple. When the rain sensor detects raindrops, the window will be closed automatically.

(1) Example code

```
#include <Servo.h>  //Import the library files of steering gear
Servo window_servo;  //Create a steering gear instance to control the window steering gear
#define waterPin A2  //Define the IO port of the raindrop sensor as A2
int waterVal = 0;

void setup() {
  Serial.begin(9600);  //Set the baud rate of serial communication to 9600
  pinMode(10, OUTPUT);
  pinMode(waterPin,INPUT);  //Set the pin to input
  window_servo.attach(10,500,2500);  //Set the pin of the steering gear as 10, and the pulse range is 500-2500ms

  window_servo.write(130);
  delay(300);
}

void loop() {
  waterVal = analogRead(waterPin);  //Read the analog value detected by the raindrop sensor
  Serial.print("waterVal = ");
  Serial.println(waterVal);  //Print the value detected by the raindrop sensor with new line
  //delay(200); //Delay to adjust the printing speed of serial port
```

```
  if(waterVal > 100)  //Greater than the set value, that is, when it rains
  {
    window_servo.write(130); //Close the window
    delay(300);
  }
  else
  {
    window_servo.write(50); //Open the window
    delay(300);
  }
}
```

(2) Experimental phenomenon

At the beginning, the window is open. Then touch the detection area of the rain sensor with your hand, and the window will be automatically closed.

Tip: *If you touch the rain sensor detection area and the window is not closed, it may be that your hands are too dry. Try again when touching the water, or debug the value set in the code yourself.*

9. Infinite mirror tunnel lamp

As soon as the infinite mirror tunnel lamp lights up, it looks very scientific and technological, and can be used as a characteristic atmosphere lamp.

Principle of infinite mirror tunnel lamp: When installing this, you may know something about it. It is very simple. There are two parallel reflectors, and the light from the middle strip light is constantly mirrored between them, which produces an infinite number of luminous lamps. The upper mirror is semi reflective and semi transparent, that is, half of the light can be transmitted and half can be reflected back, so you can see the internal light mirror tunnel.

The IO port of main board or expansion	A0

board to connect	
LED quantity	11

9.1 Colors of tunnel lamp

There are 256 * 256 * 256=16777216 colors of the WS2812RGB lamp. Let's control the display of several common colors first.

The link of RGB color comparison table:

https://tool.oschina.net/commons?type=3

(1) Example code

```
#include <Adafruit_NeoPixel.h>
#define Neo_PIN  A0  //Define the pin
#define NUMPIXELS  11 //Define the LED number
//Declare our NeoPixel strip object
Adafruit_NeoPixel pixels(NUMPIXELS, Neo_PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  pinMode(A0, OUTPUT);
  pixels.begin(); // Initialize NeoPixel
}

void loop() {
  //Use the for statement to make all LED on
  for(int i=0; i<NUMPIXELS; i++) {
    pixels.setPixelColor(i, pixels.Color(0, 150, 0));//Set the pin of the LED and RGB color
  }
  pixels.show();//Set the pin of the LED and RGB color
```

```
  delay(500);
  pixels.clear(); //Clean up
  for(int i=0; i<NUMPIXELS; i++) {
    pixels.setPixelColor(i, pixels.Color(150, 0, 0));
  }
  pixels.show();
  delay(500);
  pixels.clear();
  for(int i=0; i<NUMPIXELS; i++) {
    pixels.setPixelColor(i, pixels.Color(0, 0, 150));
  }
  pixels.show();
  delay(500);
  pixels.clear();
  for(int i=0; i<NUMPIXELS; i++) {
    pixels.setPixelColor(i, pixels.Color(150, 150, 0));
  }
  pixels.show();
  delay(500);
  pixels.clear();
  for(int i=0; i<NUMPIXELS; i++) {
    pixels.setPixelColor(i, pixels.Color(0, 150, 150));
  }
  pixels.show();
  delay(500);
  pixels.clear();
  for(int i=0; i<NUMPIXELS; i++) {
    pixels.setPixelColor(i, pixels.Color(150, 150, 150));
  }
  pixels.show();
  delay(500);
  pixels.clear();
}
```

(2) Experimental phenomenon

The different colors of the infinite mirror tunnel lamp are switched in cycles.

Tip: *The parameter "i" in the code "Pixels.setPixelColor(i, pixels.Color(0, 150, 0));" specifies the lamp number, and the last three parameters are the RGB values of the lamp, whose range is 0-255. Modify these parameters and then observe the lamp. Remember to add the display statement "pixels.show();" To display. You can choose the RGB values according to the RGB color comparison table:*

https://tool.oschina.net/commons?type=3

9.2 Mirror tunnel breathing lamp

The light is getting brighter and darker. It seems to be breathing.

(1) Example code

```
#include <Adafruit_NeoPixel.h>
#define Neo_PIN  A0  //Define the pin
#define NUMPIXELS  11 //Define the LED number
//Declare our NeoPixel strip object
Adafruit_NeoPixel pixels(NUMPIXELS, Neo_PIN, NEO_GRB + NEO_KHZ800);
int brightness = 0; //The variable is used to set the lamp brightness, and the maximum brightness value is 255

void setup() {
  pixels.begin(); //Initialize NeoPixel
}

void loop() {
  //Realize the purple breathing lamp
  for(int bright_val=0;bright_val<256;bright_val++)
  {
    displayPiexls(NUMPIXELS, bright_val, 0, bright_val);
    pixels.show();//Display
    delay(10);
  }
```

```
  for(int bright_val=255;bright_val>0;bright_val--)
  {
    displayPiexls(NUMPIXELS, bright_val, 0, bright_val);
    pixels.show();//Display
    delay(10);
  }
}

//Set a function to control all the LED and their brightness
int displayPiexls(int num, int redVal, int greenVal, int blueVal)
{
  //Use for loop statement to make all LED on
  for(int i=0; i<num; i++) {
    pixels.setPixelColor(i, pixels.Color(redVal, greenVal, blueVal));//Set the pin of the LED
and RGB color

  }
}
```

(2) Experimental phenomenon

The purple tunnel light, gradually light and gradually dark.

Tip: *Modify the code to make the breath lamp emit the color you like.*

9.3 Switch lamp colors with a button

(1) Example code

```
/*
 * create by straysnail
 * 2022/2/26
 */
#include <Adafruit_NeoPixel.h>
#define Neo_PIN  A0  //Define the pin
#define NUMPIXELS  11 //Define the LED number
//Declare our NeoPixel strip object
Adafruit_NeoPixel pixels(NUMPIXELS, Neo_PIN, NEO_GRB + NEO_KHZ800);

#define btnPin 2  //Define the button pin
#define btnPin2 4
boolean btnVal1;   //The variable is used to receive the value detected by the button
```

```
boolean btnVal2;
int count = 0; //The variable is used to record the click times of the button
boolean flag1 = 0; //Used to switch the state of button press and release
boolean flag2 = 0;

void setup() {
  Serial.begin(9600);
  pinMode(btnPin,INPUT);  //Set the pin to input
  pixels.begin(); // Initialize NeoPixel
  pixels.clear(); //Clean up
  pixels.show();//Display
}

void loop() {
  btnVal1 = digitalRead(btnPin);  //Read the button value and assign it to btnVal1
  while(btnVal1 == 0)  //Judge if the button is pressed
  {
    delay(10);
    flag1 = 1;
    while(flag1 == 1) //When the button is pressed
    {
      btnVal1 = digitalRead(btnPin); //Detect the button state again
      if(btnVal1 == 1)  //Judge if the button is released
      {
        count = count + 1; //Record the click times of the button
        Serial.println(count);  //Print the click times of the button
        flag1 = 0; //Exit the pressed state
        if(count >= 6)
        {
          count = 6;
        }
        switch(count)
        {
          case 0: displayPiexls(NUMPIXELS, 200, 0, 0); pixels.show(); break;
          case 1: displayPiexls(NUMPIXELS, 0, 200, 0); pixels.show(); break;
          case 2: displayPiexls(NUMPIXELS, 0, 0, 200); pixels.show(); break;
          case 3: displayPiexls(NUMPIXELS, 200, 200, 0); pixels.show(); break;
          case 4: displayPiexls(NUMPIXELS, 0, 200, 200); pixels.show(); break;
          case 5: displayPiexls(NUMPIXELS, 200, 200, 200); pixels.show(); break;
          case 6: displayPiexls(NUMPIXELS, 0, 0, 0); pixels.show(); break;
        }
      }
```

```
    }
  }
  btnVal2 = digitalRead(btnPin2);  //Read the button value and assign it to btnVal1
  while(btnVal2 == 0)  //Judge if the button is pressed
  {
    delay(10);
    flag2 = 1;
    while(flag2 == 1) //When the button is pressed
    {
      btnVal2 = digitalRead(btnPin2); //Detect the button state again
      if(btnVal2 == 1)  //Judge if the button is released
      {
        count = count - 1; //Record the click times of the button
        Serial.println(count);  //Print the click times of the button
        flag2 = 0; //Exit the pressed state
        if(count <= 0)
        {
          count = 0;
        }
        switch(count)
        {
          case 0: displayPiexls(NUMPIXELS, 200, 0, 0); pixels.show(); break;
          case 1: displayPiexls(NUMPIXELS, 0, 200, 0); pixels.show(); break;
          case 2: displayPiexls(NUMPIXELS, 0, 0, 200); pixels.show(); break;
          case 3: displayPiexls(NUMPIXELS, 200, 200, 0); pixels.show(); break;
          case 4: displayPiexls(NUMPIXELS, 0, 200, 200); pixels.show(); break;
          case 5: displayPiexls(NUMPIXELS, 200, 200, 200); pixels.show(); break;
          case 6: displayPiexls(NUMPIXELS, 0, 0, 0); pixels.show(); break;
        }
      }
    }
  }
}

//Define a function to control all the LED and their brightness
int displayPiexls(int num, int redVal, int greenVal, int blueVal)
{
  //Use for loop statement to make all LED on
  for(int i=0; i<num; i++) {
    pixels.setPixelColor(i, pixels.Color(redVal, greenVal, blueVal));//Set the pin of the LED
and RGB color
  }
```

```
}
```

(2) Experimental phenomenon

Click two buttons to switch the color of tunnel light back and forth.

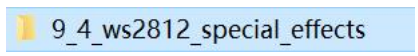
Tip: *The code uses the button counting function, as well as "switch(count){case 1: break;}" conditional statements. Compared to "if", when there are many conditions to judge, use the switch statement to make the code look more concise.*

9.4 Special lighting effect

Control the tunnel light to display various special light effects.

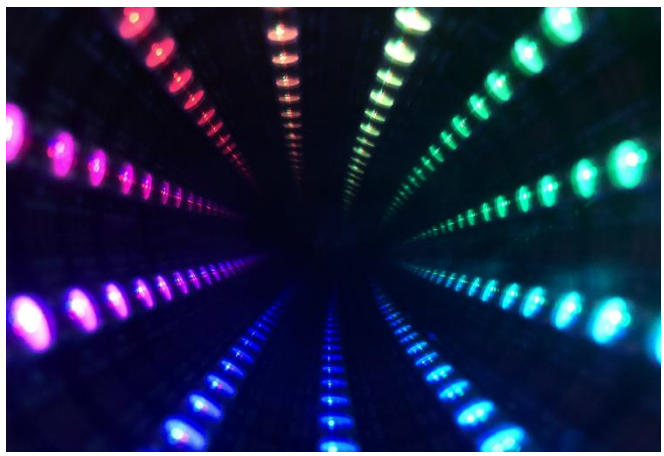
(1) Example code

Because the code is so long, please open the example code to view.



(2) Experimental phenomenon

The tunnel light shows special light effects.



Tip: *The code of special light effects is difficult to write. If you are interested in making some atmosphere lights, you should spend some time to understand the code of special light effects.*

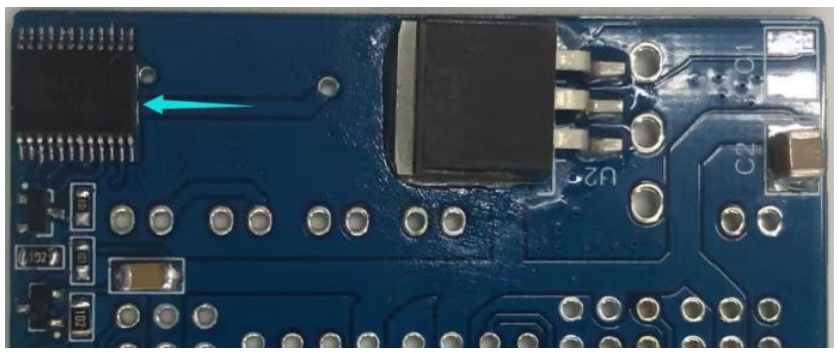
10. Great windmill

The great windmill turns slowly and looks beautiful.

TB6612fng motor driver chip and N20 deceleration DC motor are used here.

TB6612 driver chip: TB6612 chip is integrated into the expansion board, and we have made some adjustments to its peripheral circuit, so that it can control two circuits of motors from 7 IO ports to 4 IO ports, reducing the occupation of IO ports. When the work is relatively large, the IO ports of Arduino UNO are very valuable.

The TB 6616fng driver chip is on the back of the expansion board, as shown in the following figure.



Function	The IO port of main board or expansion board to connect
Controlling direction	D8
Controlling speed	D5

10.1 Rotating windmill

The IO port of controlling direction outputs high level HIGH or low level LOW, and the IO port of controlling speed outputs analog value as 0-255, so that the forward and backward speed of the motor can be controlled. When the speed value is 0, the windmill stops rotating.

IO port for direction: D8	IO port for speed: D5
HIGH（rotate anticlockwise）	PWM value: 10-255
LOW（rotate clockwise）	PWM value: 10-255

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/24
 */
#define INB 8  //Define pin 8 to control the direction
#define ENB 5  //Define pin 5 to control the speed

void setup() {
  pinMode(INB, OUTPUT);//Set the pin of controlling the motor to output
  pinMode(ENB, OUTPUT);
}

void loop() {
  anticlockwise();  //Rotate anticlockwise
  delay(2000);
  Stop();           //Stop
  delay(200);
  clockwise();      //Rotate clockwise
  delay(2000);
  Stop();           //Stop
  delay(200);
}

//Rotate anticlockwise
void anticlockwise()
{
  digitalWrite(INB, HIGH);  //Control the direction
  analogWrite(ENB, 50);    //Control the speed which range is 10-255
}
//Rotate clockwise
void clockwise()
{
  digitalWrite(INB, LOW);
  analogWrite(ENB, 100);
}
//Stop
void Stop()
{
  digitalWrite(INB, HIGH);
  analogWrite(ENB, 0);
```

```
}
```

(2) Experimental phenomenon

The windmill first rotates anticlockwise for 2 seconds, then clockwise for 2 seconds, and keep cycling.

Tip: *Modify the analog output value "analogWrite(ENA, 100);, which range is 0-255, changing the speed of the windmill. If the PWM value is too small, the power will be insufficient. It is recommended to write more than 10.*

10.2 Button to control the windmill

Use button 1 to be the switch of controlling the windmill.

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/24
 */
#define INB 8  //Define pin 8 to control the direction
#define ENB 5  //Define pin 5 to control the speed
#define btnPin 2  //Define the button pin
boolean btnVal;   //The variable is used to receive the values detected by the button
int count = 0; //The variable is used to record the click times of the button
boolean flag = 0; //Used to switch the state of button press and release
int data;

void setup() {
  Serial.begin(9600);
  pinMode(INB, OUTPUT);//Set the pin of controlling the motor to output
  pinMode(ENB, OUTPUT);
  pinMode(btnPin,INPUT);  //Set the pin of the button to input
}

void loop() {
  btnVal = digitalRead(btnPin);  //Read the button value and assign it to btnVal
  if(btnVal == 1)  //Judge if the button is pressed
  {
    delay(10); //Delay to eliminate the shake of the button
    flag = 1;
```

```
    while(flag == 1) //While loop function, it will not exit the loop until the button is released and flag == 0
    {
      btnVal = digitalRead(btnPin); //Detect the button state again
      if(btnVal == 0)  //Judge if the button is released
      {
        count = count + 1; //Record the click times of the button
        Serial.println(count);  //Print the click times of the button
        flag = 0; //Exit the state of button press
      }
    }

    data = count % 2;  //Calculate the remainder. If it is singular, the value is equal to 1, and if it is even, the value is equal to 0

    if(data == 1) //If it is singular
    {
      anticlockwise();  //Rotate anticlockwise
    }
    else
    {
      Stop(); //Stop
    }
  }
}

//Rotate anticlockwise
void anticlockwise()
{
  digitalWrite(INB, HIGH);  //Control the direction
  analogWrite(ENB, 50);    //Control the speed whose range is 10-255
}
//Stop
void Stop()
{
  digitalWrite(INB, HIGH);
  analogWrite(ENB, 0);
}
```

(2) Experimental phenomenon

Click button 1 once, the windmill rotates. Click again, it stops.

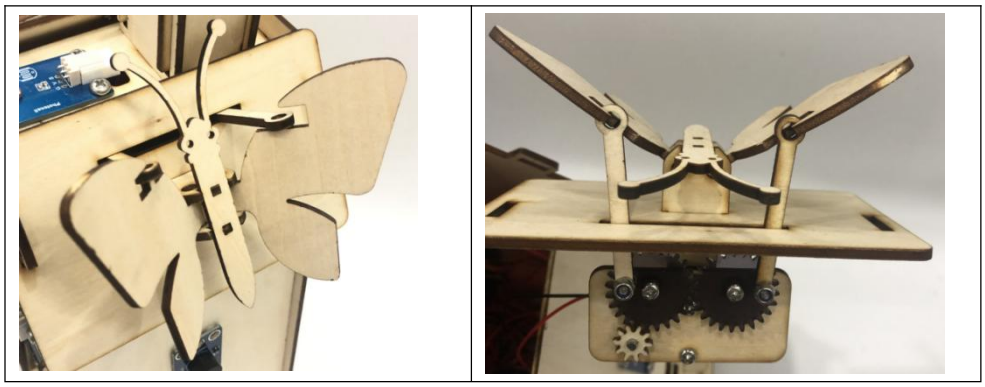
Exercise after class: *Click button 1 to control the rotation of the windmill, and click button 2 to stop the windmill?*

11. Mechanical butterfly

Considering that the environment around the Dutch windmill is grassland, there will naturally be dancing butterflies, so there should also be a mechanical butterfly on the windmill house.

The power of the mechanical butterfly is also that TB6612fng driver chip drives the N20 deceleration DC motor to rotate.

We can take down the butterfly to see its structure:



It can be seen that the small gear on the N20 motor drives the two synchronous gears to rotate at the same time. The synchronous gears drive the connecting rod to push up and down, and the wings of the butterfly are stirred up.

The ports of the butterfly motor:

IO port for direction: D7	IO port for speed: D6
HIGH（rotate anticlockwise）	PWM value: 10-255
LOW（rotate clockwise）	PWM value: 10-255

11.1 Control the butterfly to flutter its wings

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/24
 */
#define INA 7  //Define pin 7 to control the direction
```

```
#define ENA 6  //Define pin 6 to control the speed


void setup() {
  pinMode(INA, OUTPUT);//Set the pin of controlling the motor to output
  pinMode(ENA, OUTPUT);
}

void loop() {
  butterfly();
}

//Rotate anticlockwise
void butterfly()
{
  digitalWrite(INA, HIGH);  //Control the direction
  analogWrite(ENA, 100);    //Control the speed whose range is 50-255
}
```

(2) Experimental phenomenon

The wings of the butterfly flutter slowly.

11.2 The frightened butterfly

In the nature, butterflies are feed on pollen. When someone approaches, it is easy to frighten the butterfly, and it immediately flutters its wings and flies away.

In this lesson, we use human infrared pyroelectric sensor and mechanical butterfly to simulate this process.

(1) Example code

```
/*
 * create by straysnail
 * 2022/11/24
 */
#define INA 7  //Define pin 7 to control the direction
#define ENA 6  //Define pin 6 to control the speed
#define PIR_pin  A3   //Define the pin of pyroelectric sensor
boolean PIR_Val;  //Define a variable of bool type

void setup() {
  pinMode(INA, OUTPUT);//Set the pin of controlling the motor to output
  pinMode(ENA, OUTPUT);
  pinMode(PIR_pin, INPUT); //Set the pin to input
}

void loop() {
  PIR_Val = digitalRead(PIR_pin); //Read the value detected by the human body infrared
pyroelectric sensor and assign it to PIR _Val
  if(PIR_Val == 1) {  //Person detected
    butterfly();  //The butterfly flutters
    delay(2000);  //Flutters for 2s
  } else {   //No person detected
    Stop();  //Stop
  }
}

//Rotate anticlockwise
void butterfly()
{
  digitalWrite(INA, HIGH);  //Control the direction
  analogWrite(ENA, 200);    //Control the speed whose range is 50-255
}
void Stop()
{
  digitalWrite(INA, LOW);
  analogWrite(ENA, 0);
}
```

(2) Experimental phenomenon

Somebody dangled in front of the human infrared pyroelectric sensor, and the mechanical butterfly flutters its wings. When nobody is around, it does not move.

12. OLED screen

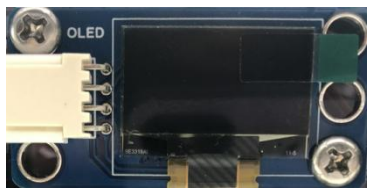
Since it is natural for an intelligent windmill to have a screen, a 0.96-inch blue OLED screen is installed.

OLED screen: Display screen is the most commonly used module for the frequently-used devices for modern people, such as mobile phones, computers, televisions, etc. An electronic device will look backward without a display screen. The display screen really makes the interaction between electronic devices and people intuitive, simple and practical.

The IO port for the OLED display SDA pin	A4
The IO port for the OLED display SCL pin	A5

12.1 OLED display screen

Although the OLED screen is small, there are many functions that can be realized. Now let's learn the basic functions of controlling the OLED screen.



OLED display characters.

(1) **Example code**

```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // The display is 128 pixels long
#define SCREEN_HEIGHT 64 // The screen is 64 pixels wide
#define OLED_RESET     4 // Reset pin # (or -1 if sharing Arduino reset pin)
```

```
#define SCREEN_ADDRESS 0x3C //OLED   IIC address of the screen
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
#define NUMFLAKES     10

void setup() {
  Serial.begin(9600);
  //Initialize display
  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
   // Clear screen
  display.clearDisplay();
  display.display();
}

void loop() {
  display.setTextSize(2);             // Set font size
  display.setTextColor(SSD1306_WHITE);        // Font color, there's only one color here
  // The starting position coordinates of the display, 0,0 is the top left corner
  display.setCursor(0,0);
  //display.print(F("Windmill"));           //Display character
  display.println(F("Windmill"));           //Line feed display character
  display.println(F("Snail man"));
  display.display(); //display
  display.clearDisplay();    //clear screen
  delay(200);
}
```

(2) Experimental phenomenon

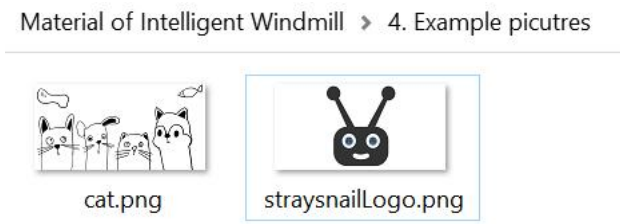
The OLED display displays the string, as shown below

12.2 OLED display pictures

As a screen, it should be able to display pictures. Now let's learn how to display pictures with OLED.

(1) Get your pictures ready

The pixels of our OLED are 128 * 64P, so we need to prepare 128 * 64P pictures. We have provided sample pictures in the “4. Example picture” folder.

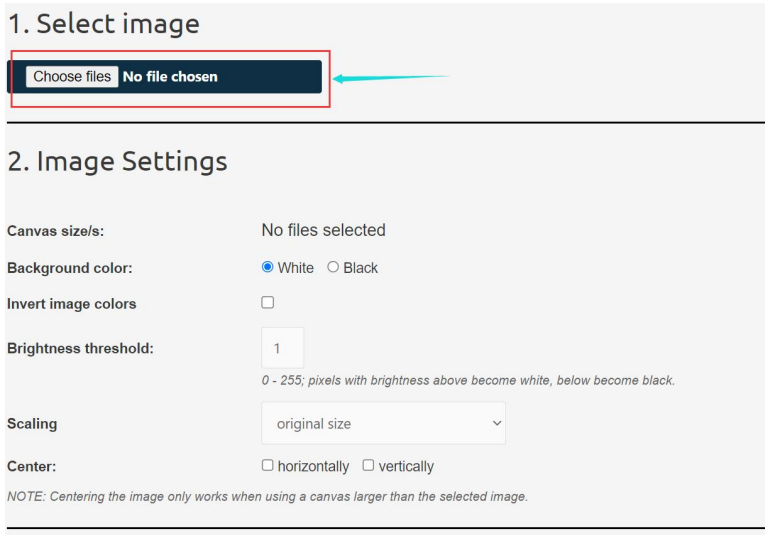


(2) Convert the picture to hexadecimal

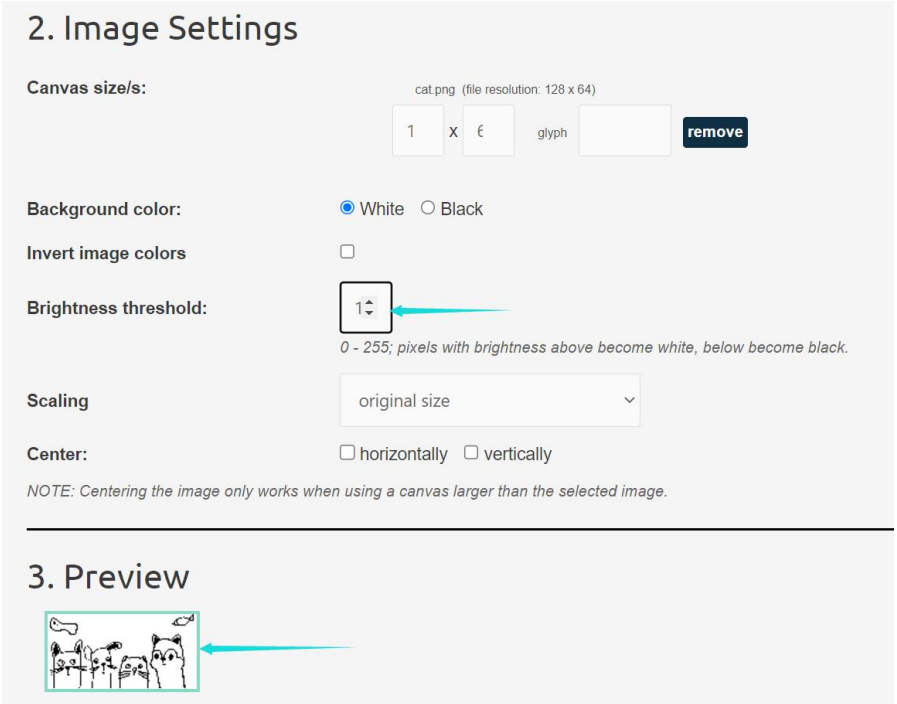
Link for picture to hexadecimal:

https://diyusthad.com/image2cpp

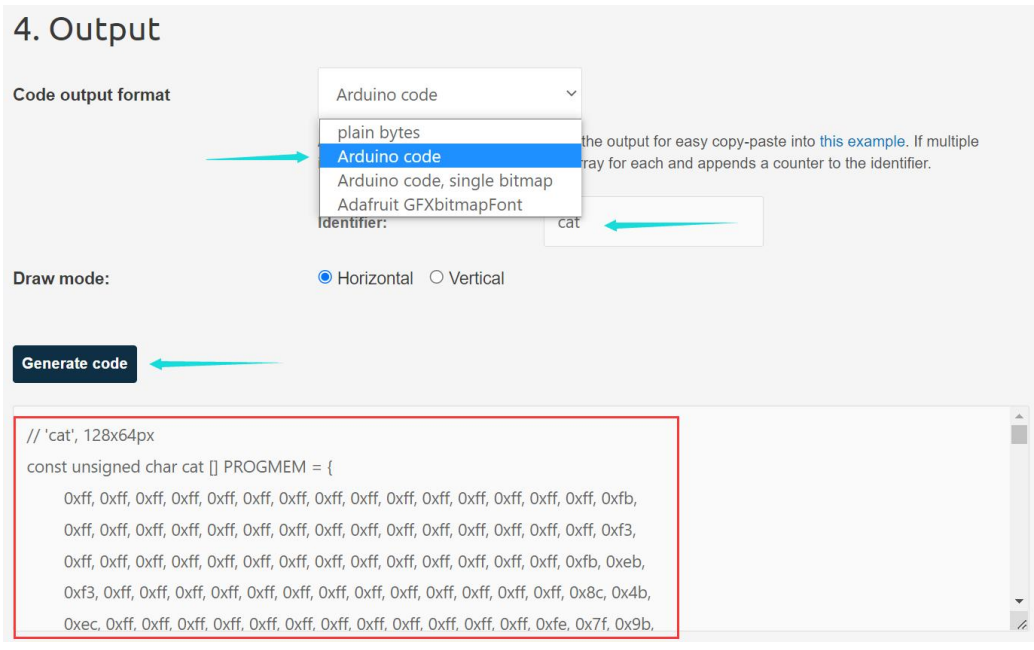
Open the link, and drag the “cat.png” sample image to the “Select file”.



When you just bring in the picture, you can see that the effect picture in “3. Preview” is not very clear. You can click the up arrow of Brightness threshold to adjust it until the picture becomes clear, as shown below.



Select “Arduino code”, rename it “cat”, and then click “Generate code” to generate the hexadecimal code for Arduino C. Copy and paste the hexadecimal code into the example code provided by us.



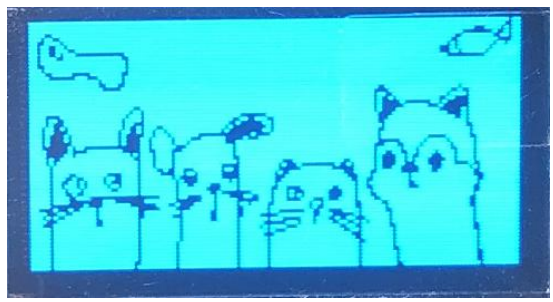
(3) Example code

Because the code is so long, please open the example code to view.

If you convert other pictures, remember to modify the name of the corresponding hexadecimal code in the code, such as “cat display. drawBitmap(0, 0, cat, 128, 64, 1);”.

(4) Experimental phenomenon

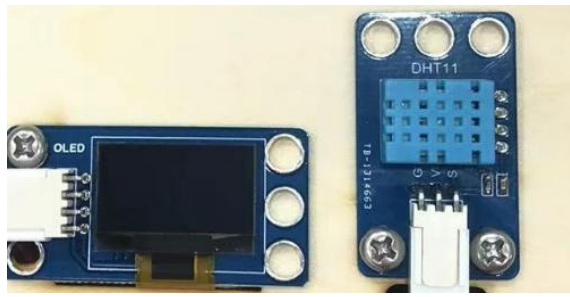
The OLED screen displays the picture successfully.



13. Temperature and humidity meter

Modern people pay more and more attention to the quality of living environment. They often use a meter to detect the temperature and humidity of the indoor environment. If the air is too dry, they can spray some water vapor to make the environment more comfortable.

The model of the temperature and humidity sensor of the intelligent windmill is DHT11, which can accurately detect the temperature and humidity values of the surrounding environment. Then use the OLED display screen to display the measured values.



13.1 DHT11

Display the values measured by the temperature and humidity DHT11 in the serial port monitor.

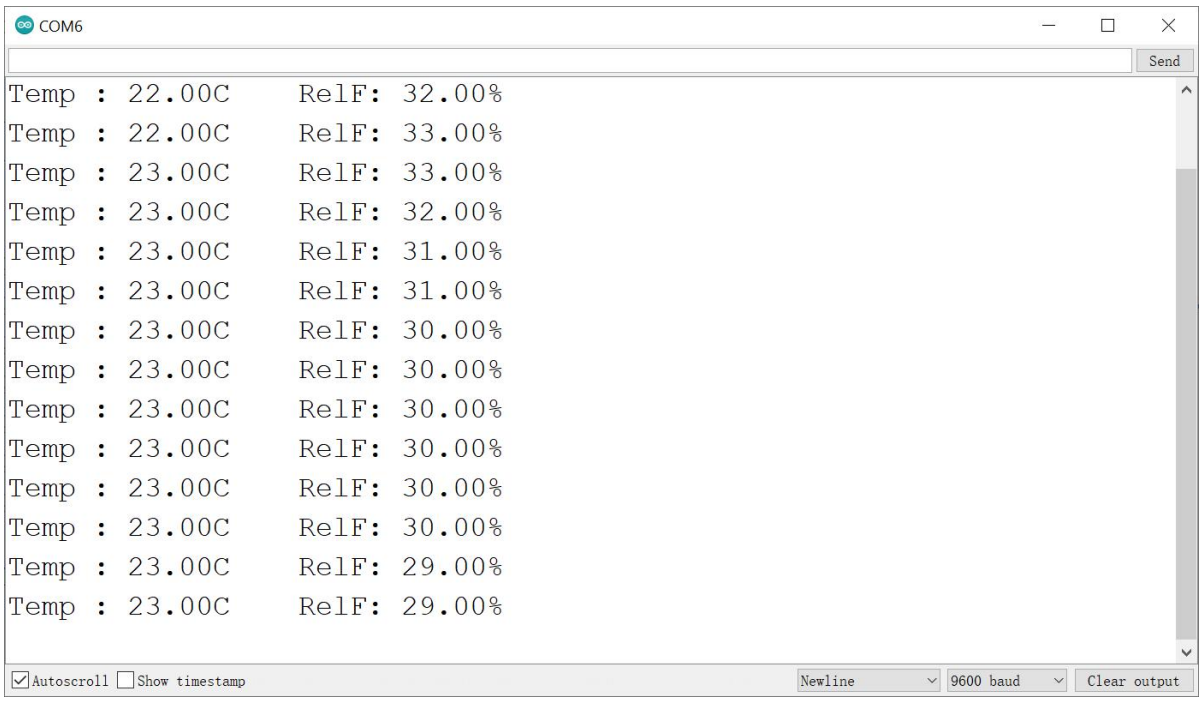
(1) Example code

```
#include <dht11.h>
dht11 DHT11;
#define DHT11PIN 12

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  int chk = DHT11.read(DHT11PIN);  //Read the value of the DHT11
  Serial.print("Temp : ");
  Serial.print((float)DHT11.temperature);  //Read the temperature value and float is a
floating point value.
  Serial.print("C");
  Serial.print("    ");
  Serial.print("RelF: ");
  Serial.print((float)DHT11.humidity);   //Read the humidity value
  Serial.println("%");
```

```
  delay(2000);
}
```

(2) Experimental phenomenon

The serial port monitor prints the values of temperature and humidity.



13.2 OLED screen displays the values of temperature and humidity

Temperature and humidity meter should have a screen for display. Here, OLED screen is used to display the values.

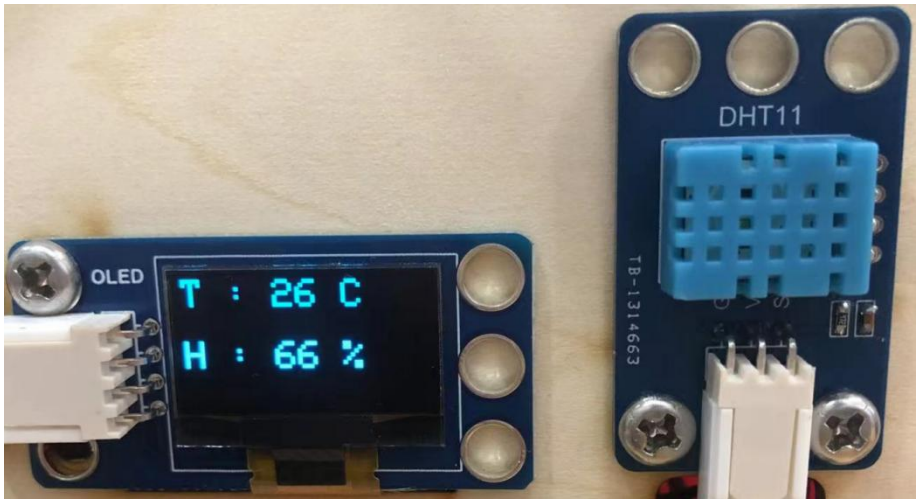
(1) Example code

```
#include <dht11.h>
dht11 DHT11;
#define DHT11PIN 12

#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET     4 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32
```

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
#define NUMFLAKES     10 // Number of snowflakes in the animation example

void setup() {
  Serial.begin(9600);
  // SSD1306_SWITCHCAPVCC = generate display voltage from 3.3V internally
  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }

   // Clear the buffer
  display.clearDisplay();
  display.display();
}

void loop() {
  int chk = DHT11.read(DHT11PIN); //Read the values detected by the DHT11
  display.setTextSize(2);             // Normal 1:1 pixel scale
  display.setTextColor(SSD1306_WHITE);        // Draw white text
  display.setCursor(0,0);             // Start at top-left corner
  display.print(F("T : "));
  display.print(DHT11.temperature);
  display.println(F(" C"));
  display.println(F(""));
  display.print(F("H : "));
  display.print(DHT11.humidity);
  display.println(F(" %"));
  display.display();
  display.clearDisplay();
  delay(2000);
}
```

(2) Experimental phenomenon

OLED screen displays the values of temperature and humidity.



14. Morse code gate

Many people should have seen some stories in the movie that use Morse code to communicate, such as sending Morse code to each other by tapping, using a flashlight, etc.

Little knowledge: Morse code is a kind of signal code that goes on and off, expressing different English letters, numbers and punctuation marks through different permutations. Morse code consists of two basic signals: short dot signal " •", read "Di", and long signal "—" for a certain time, read "Da". Interval time: Di=1t, Da=3t, Di Da interval=1t, character interval=3t, word interval=7t.

A •—	M ——	Y —•——	6 —••••
B —•••	N —•	Z ——••	7 ——•••
C —•—•	O ———	Ä •—•—	8 ———••
D —••	P •——•	Ö ———•	9 ————•
E •	Q ——•—	Ü ••——	. •—•—•—
F ••—•	R •—•	Ch ————	, ——••——
G ——•	S •••	0 —————	? ••——••
H ••••	T —	1 •————	! ••——•
I ••	U ••—	2 ••———	: ———•••
J •———	V •••—	3 •••——	" •—••—•
K —•—	W •——	4 ••••—	' •————•
L •—••	X —••—	5 •••••	= —•••—

14.1 OneButton

The "OneButton" library file allows you to use various ways of buttons, including double click, long press and other functions. See the link for details:

https://github.com/mathertel/OneButton

(1) Example code

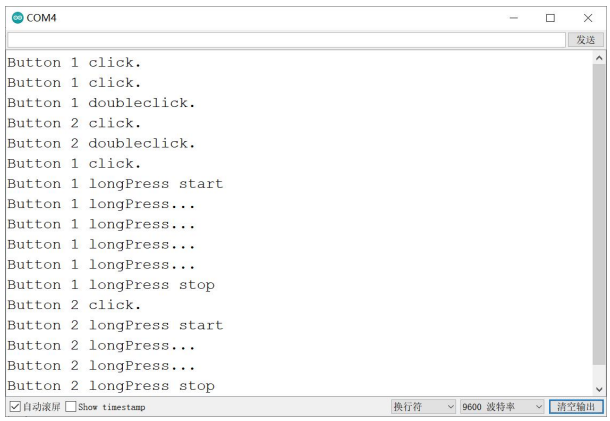
```
#include "OneButton.h"

// Setup a new OneButton on pin 2.
OneButton button1(2, true);
// Setup a new OneButton on pin 4.
OneButton button2(4, true);

void setup() {
  Serial.begin(9600);
  // link the button 1 functions.
  button1.attachClick(click1);
  button1.attachDoubleClick(doubleclick1);
  button1.attachLongPressStart(longPressStart1);
  button1.attachLongPressStop(longPressStop1);
  button1.attachDuringLongPress(longPress1);

  // link the button 2 functions.
  button2.attachClick(click2);
  button2.attachDoubleClick(doubleclick2);
  button2.attachLongPressStart(longPressStart2);
  button2.attachLongPressStop(longPressStop2);
  button2.attachDuringLongPress(longPress2);
}

void loop() {
  // keep watching the push buttons:
  button1.tick();
  button2.tick();
}

// ----- button 1 callback functions

// This function will be called when the button1 was pressed 1 time (and no 2. button press
followed).
void click1() {
  Serial.println("Button 1 click.");
} // click1


// This function will be called when the button1 was pressed 2 times in a short timeframe.
```

```
void doubleclick1() {
  Serial.println("Button 1 doubleclick.");
} // doubleclick1


// This function will be called once, when the button1 is pressed for a long time.
void longPressStart1() {
  Serial.println("Button 1 longPress start");
} // longPressStart1


// This function will be called often, while the button1 is pressed for a long time.
void longPress1() {
  Serial.println("Button 1 longPress...");
} // longPress1


// This function will be called once, when the button1 is released after beeing pressed for
a long time.
void longPressStop1() {
  Serial.println("Button 1 longPress stop");
} // longPressStop1


// ... and the same for button 2:

void click2() {
  Serial.println("Button 2 click.");
} // click2


void doubleclick2() {
  Serial.println("Button 2 doubleclick.");
} // doubleclick2


void longPressStart2() {
  Serial.println("Button 2 longPress start");
} // longPressStart2


void longPress2() {
```

```
  Serial.println("Button 2 longPress...");
} // longPress2

void longPressStop2() {
  Serial.println("Button 2 longPress stop");
} // longPressStop2
```

(2) Experimental phenomenon

Open the serial port monitor, click the button, double click the button, and long press the button to see the corresponding string printed.

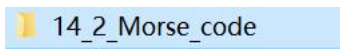


14.2 Morse code gate

Open the door by entering the Morse code. Morse code “F” is used as the password in the code.

(1) Example code

Because the code is so long, please open the example code to view.



(2) Experimental phenomenon

Enter the Morse password by short press and long press of button 1. The OLED screen displays the entered password. The Morse code for opening the door is "F", that is, " • •— •". If the input is correct, the OLED displays “success” and the door will open; If it is an error input, the OLED displays “error”. Wait for 1 second, and then enter the password again when the OLED displays “again”.

15. Music box

The movement of the music box is composed of sound tube, sound board, gear, spring (or other power source), damping and other components, with a very delicate structural design. The voice is clear and moving.

Although the buzzer we use is not as beautiful as the sound of the music box, it can also produce many tones. In this lesson, we will use the buzzer to make a music box.

15.1 music box

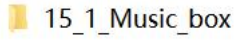
Program design idea:

- Click button 2 to select the song you want to play
- Long press button 2 to play the selected song
- Click button 1 to trigger the interruption and stop playing the song

Click button 1 to trigger the interrupt. Here, the external interrupt function is used. The external interrupt pins of UNO are 2 and 3, and our button 1 is connected to pin 2.

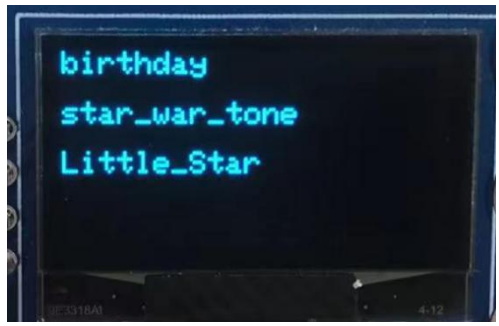
(1) Example code

Because the code is so long, please open the example code to view.



(2) Experimental phenomenon

OLED displays the names of songs that can be played.



Click button 2 to select a song, and long press button 2 to play the selected song.
In the process of playing a song, click button 1 to trigger an external interrupt to stop playing.

(3) Code knowledge

attachInterrupt (Parameter 1, Parameter 2, Parameter 3); attachInterrupt is the name of the external trigger interrupt function. The external interrupt pins of UNO are 2 and 3. Parameter 1, the pin of triggering interrupt; Parameter 2, the function to be executed after triggering the interrupt; Parameter 3, condition for triggering external interrupt function. LOW (low level trigger), CHANCE (triggered on change), RISING (trigger from low level to high level), FALLING (trigger from high level to low level)	
void(* resetFunc) (void) = 0;	Reset function of the software

16. Press button to switch multiple functions

In this lesson, we write an integrated multi-functional code. The function menu is displayed on the OLED screen, and then two buttons are used to switch functions. The code in this lesson is also the code in our demo tutorial.

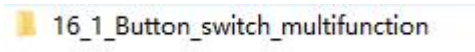
16.1 Button and screen interactive selection function

Code design idea:

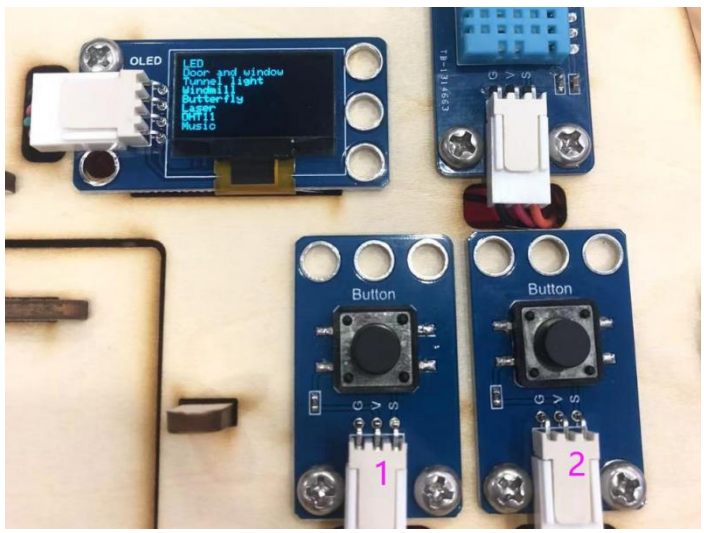
- OLED screen displays the function menu
- Click button 1 once to select function
- Double click button1 to enter the selected function
- Press and hold button 1, then press button 2 to exit the function

(1) Example code

The code is long. Open the example code “16_ 1_ Button_ switch_ multifunction” provided by us by using the Arduino IDE.



(2) Experimental phenomenon and operation steps



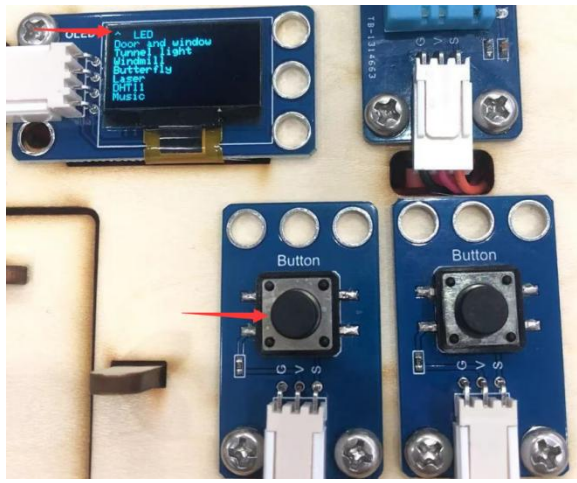
1) Easy trial steps

- The OLED screen displays the names of 8 functions.
- Click button 1 slowly to select the function.
- Double click button 1 quickly to enter the function, and the screen will display the operation method of the corresponding function.
- Press and hold button 1 and then press button 2 to exit the current function.

2) Detailed trial operation steps

① **LED**

Click button 1 slowly to choose the LED function. The “^”in front of “LED”means that it is selected, as shown below.



Double click button 1 quickly to enter the function of controlling the LED. Click button 2 once, and LED comes on. Click button 1 once, and LED is off.

Press and hold button 1, and click button 2 immediately, the current function will be exited.

② **Door and window**

Click button 1 slowly to choose Door and window function.

Double click button 1 quickly to enter the function of controlling door and window. Click button 2 once, and the door and window open. Click button 2 again, they close.

Press and hold button 1, and click button 2 immediately, the current function will be exited.

③ **Tunnel light**

Click button 1 slowly to choose Tunnel light function.

Double click button 1 quickly to enter the function of controlling tunnel light. Then click button 2 to change the color of it, as shown below.

Press and hold button 1, and click button 2 immediately, the current function will be exited.

④ **Windmill**

Click button 1 slowly to choose windmill function.

Double click button 1 quickly to enter windmill function. Then click button 2 to turn the

windmill, and click button 1 to stop the windmill.

Press and hold button 1, and click button 2 immediately, the current function will be exited.

⑤ **Butterfly**

Click button 2 slowly to select butterfly function,as shown below.

Double click button 1 to enter the butterfly function, then click button 2, the butterfly will flutter its wings. Click button 1 to stop the butterfly.

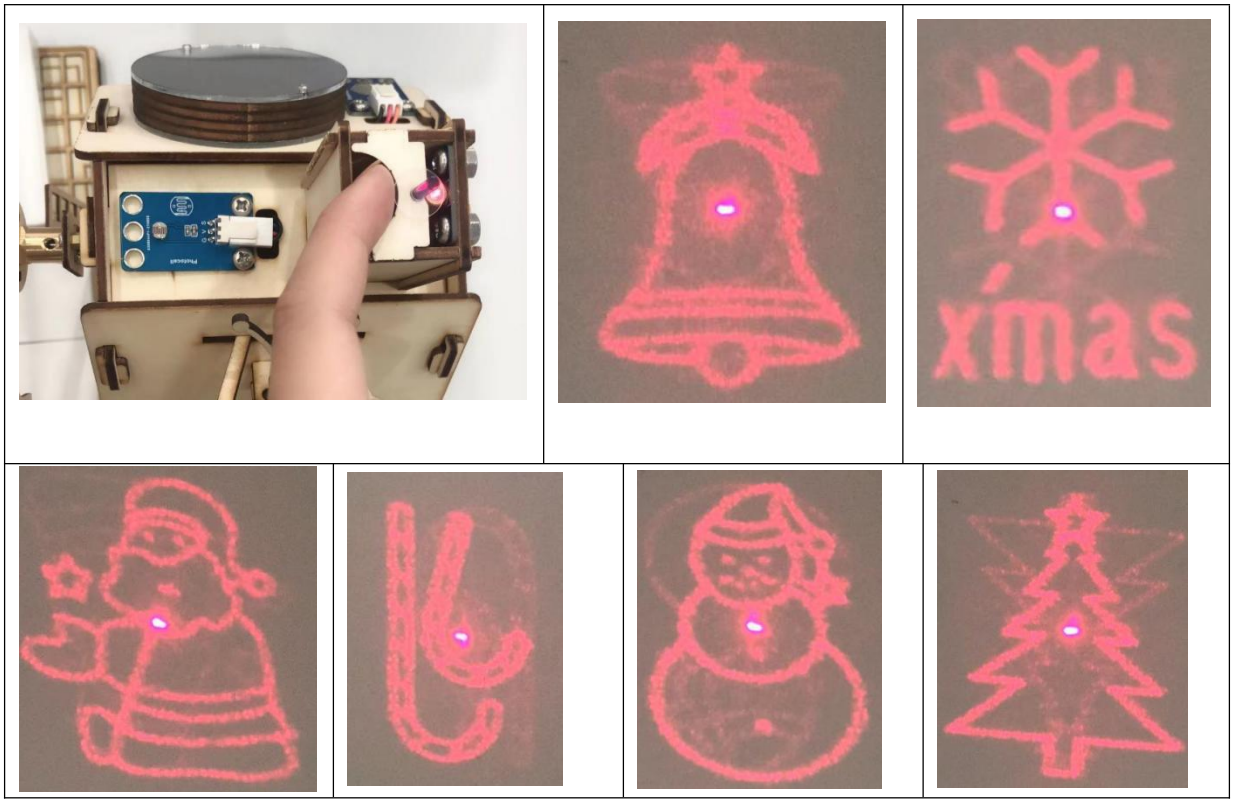
Press and hold button 1, and click button 2 immediately, the current function will be exited.

⑥ **Laser**

Laser projection function

Click button 1 slowly to select laser function.

Double click button 1 quickly to enter laser function. Click button 2 to turn on the laser,and then turn the DOE to switch the projected pattern.



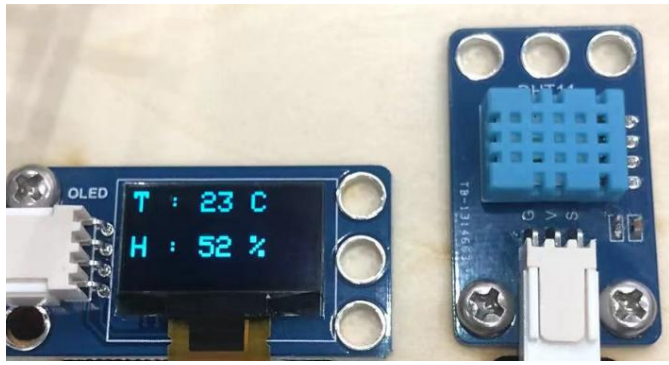
Press and hold button 1, and click button 2 immediately, the current function will be exited.

⑦ **DHT11**

DHT11 temperature and humidity meter

Click button 1 slowly to select DHT11.

Double click button 1 to enter the DHT11 function, and then click button 2 to display the temperature and humidity values on the OLED screen.



Press and hold button 1, and click button 2 immediately, the current function will be exited.

⑧ **Music**

The buzzer plays Happy Birthday.

Click button 1 slowly to select Music function.

Then double click button 1 to enter the Music function, and click button 2 once to play Happy Birthday.

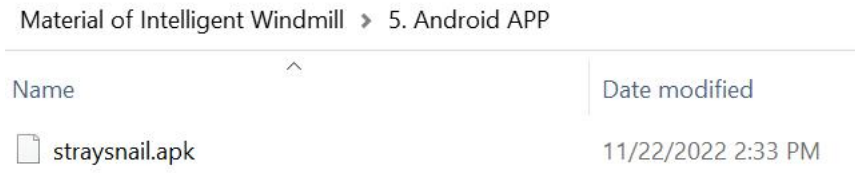
Press and hold button 1, and click button 2 immediately, the current function will be exited.

17. APP controls the windmill

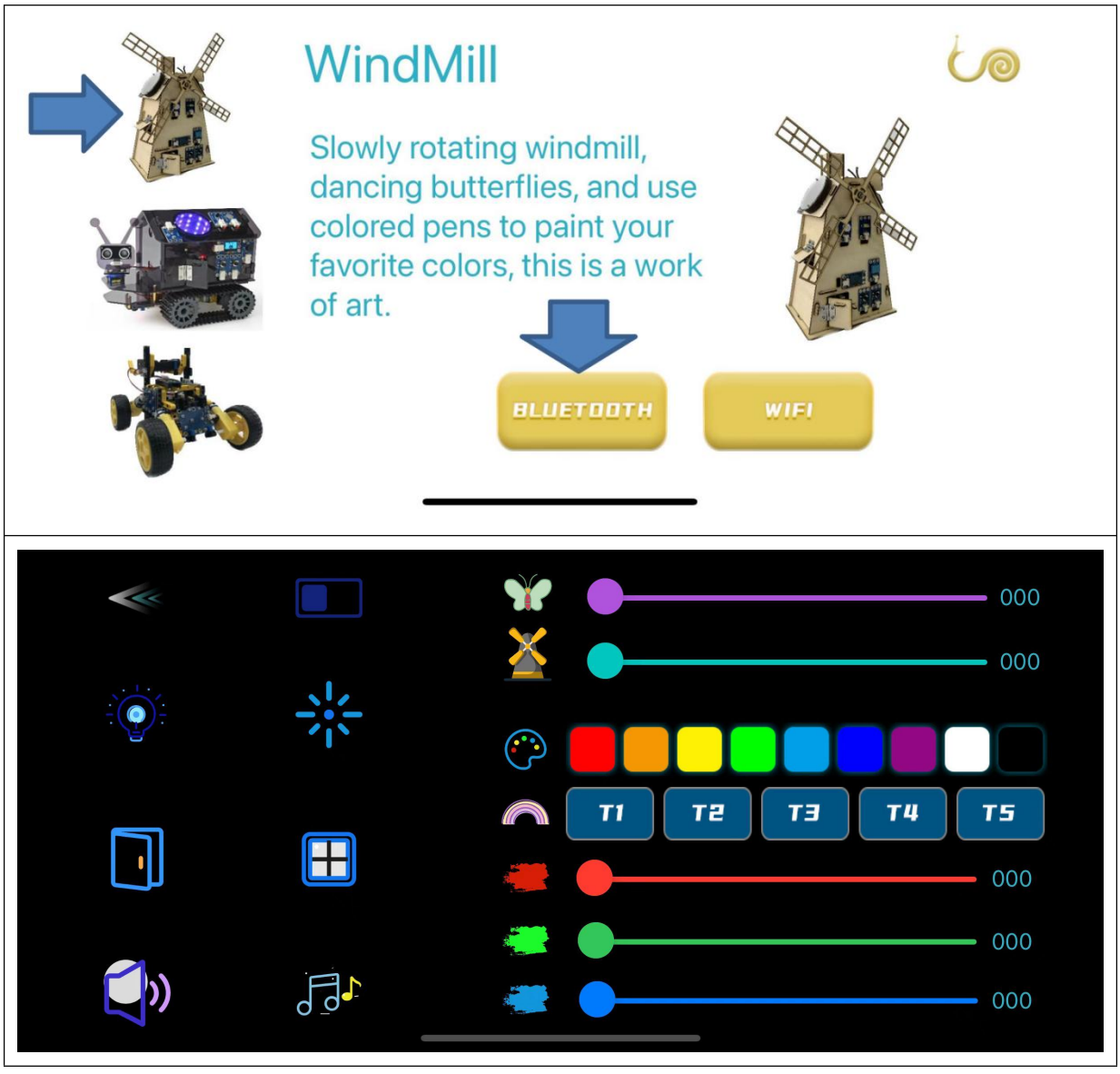
Mobile phones are essential for modern young people. Bluetooth is an essential function of mobile phones, so we have designed and developed a Bluetooth APP to specifically control various functions of the windmill.

Download Android APP: Search “straysnail” in Google play

Or use the installation package provided by us:



Download Apple APP: Search “straysnail” in APP Store
APP interface introduction:
On the left of the APP interface, you can select a product. Select the windmill shown below, and then click “Bluetooth”.

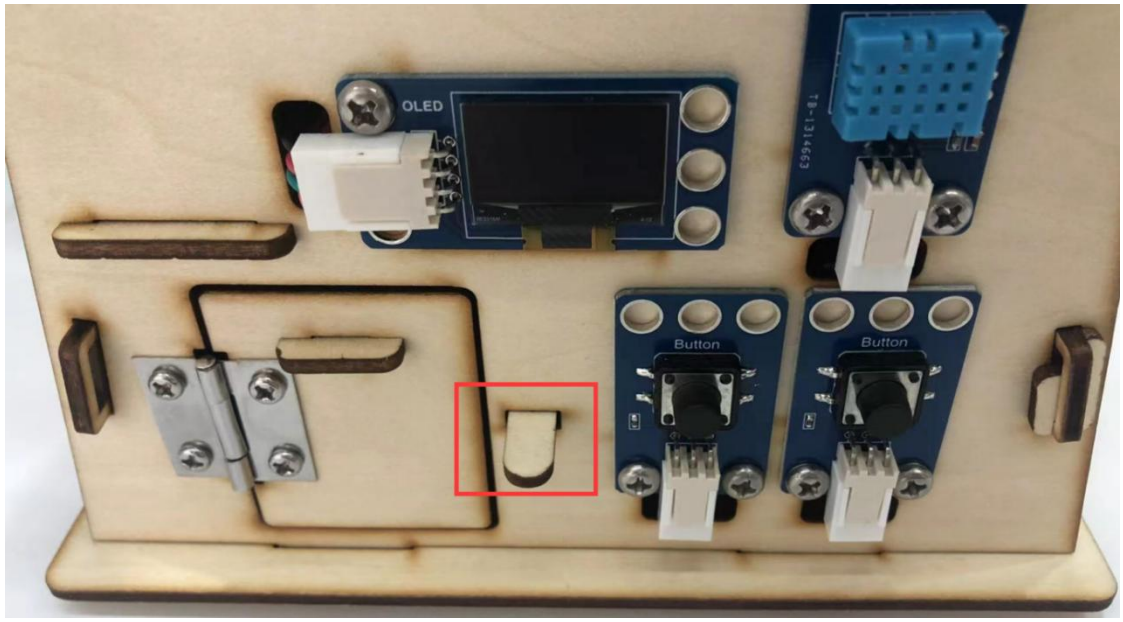


17.1 Read the values of Bluetooth APP

The Bluetooth APP controls the windmill by sending characters or digital commands to the Bluetooth module of it. What characters are the buttons of our app? In order to intuitively see what characters are received, we print the received characters on the serial port monitor.

Note: Before uploading the code, turn off the Bluetooth switch, because both Bluetooth

and USB use the hard serial port 0,1 of Arduino UNO, which will cause code upload failure. Push the push rod beside the door in and close the Bluetooth switch as shown in the figure below.



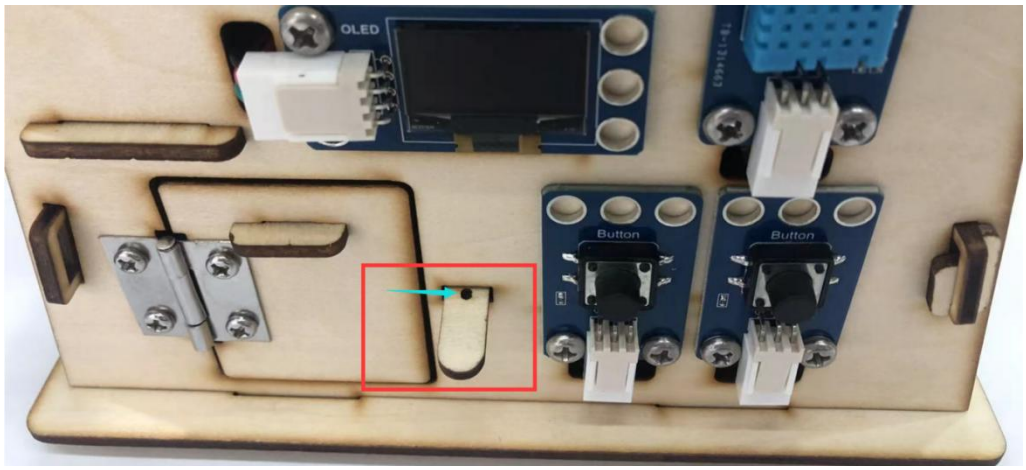
(1) Example code

```
char bleStr = "";

void setup() {
  Serial.begin(9600);

}

void loop() {
  while(Serial.available() > 0)  //Judge whether the serial port area receives value
  {
    bleStr = Serial.read();  //Read the value of serial port area
    Serial.println(bleStr);
    delay(10);
  }
}
```

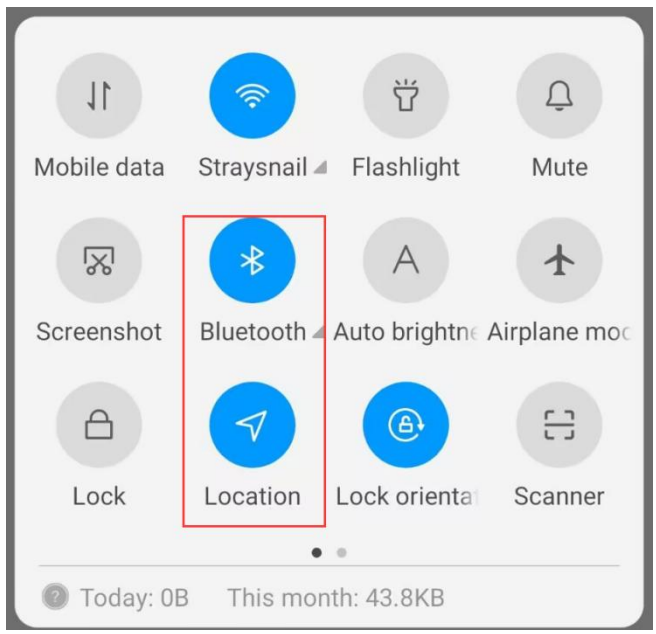
(2) Experimental phenomenon

Turn on the serial port monitor, turn on the Bluetooth switch of the windmill (that is, pull the switch on the right side of the door and the complete hole on the switch can be seen), turn on the mobile phone APP, connect the Bluetooth, click the button on the interface, and you can see that the serial port monitor prints the received characters.

1) Turn on the Bluetooth switch of the windmill after burning the code (be careful not to turn it on first and then burn the code, which will result in code burning failure). Pull the rod beside the door outward to open the Bluetooth switch, as shown in the figure below.



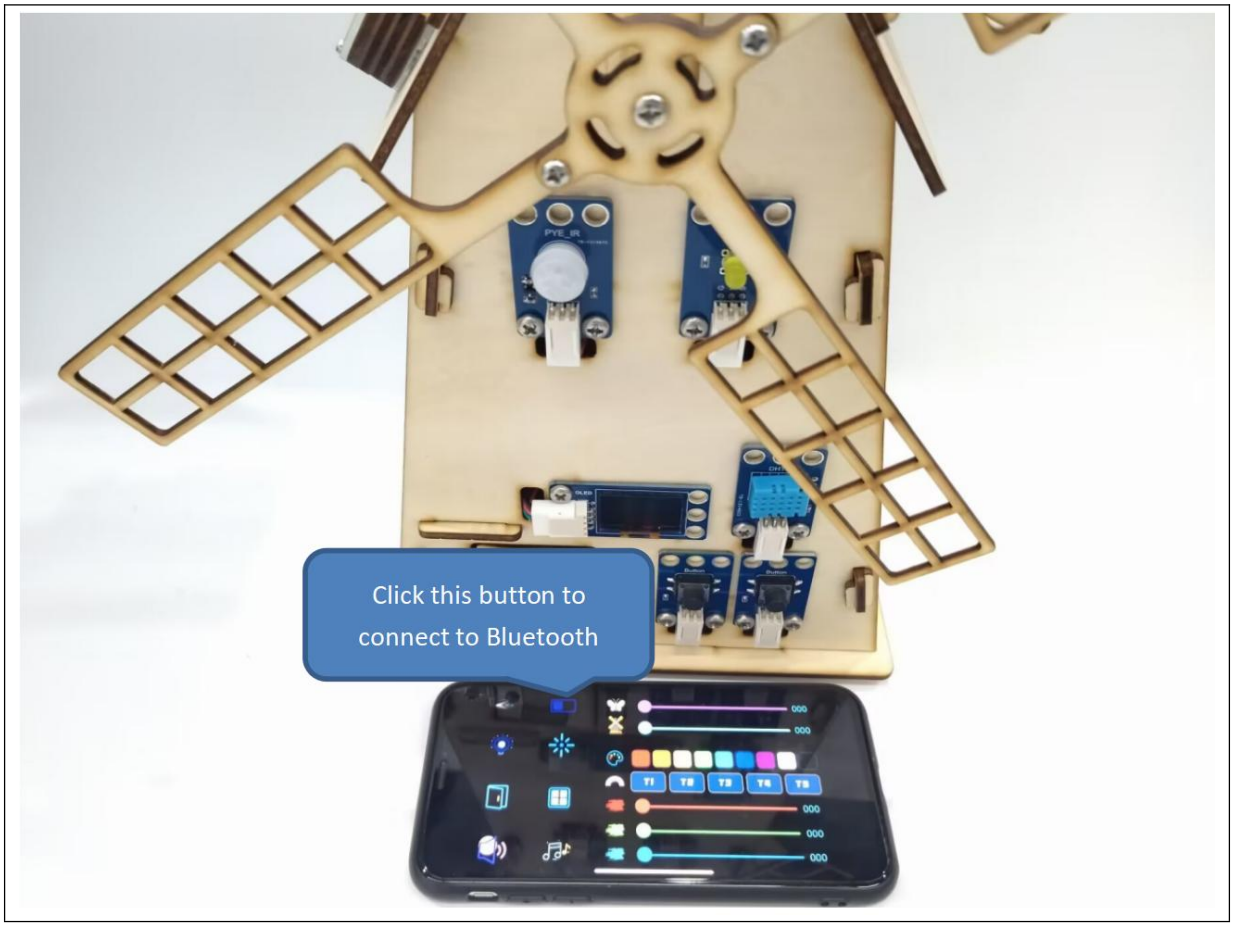
2) Open the APP for Bluetooth search and connection. Android phone needs to open its Bluetooth and location information, as shown below. iPhone just needs to turn on the Bluetooth.



There are two ways to connect the Bluetooth:

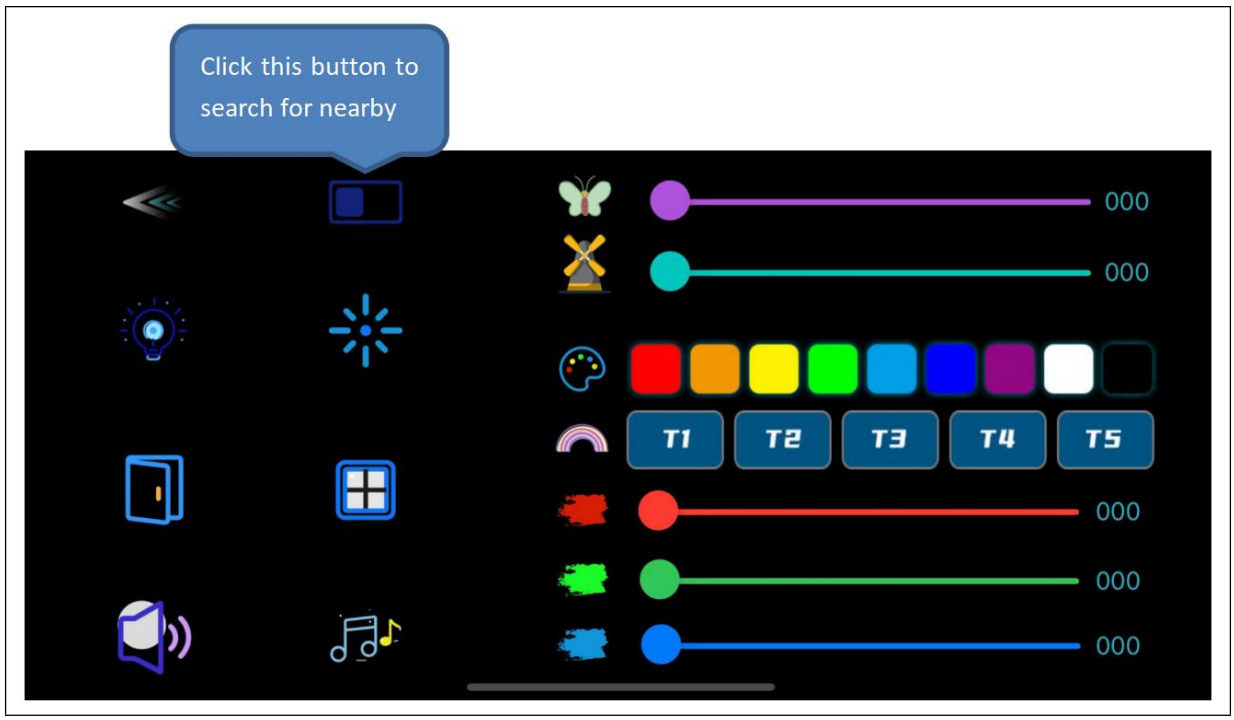
① **Close to connect**

Turn on the power of the windmill, and turn on the APP on your mobile phone. Get close to the windmill, and click the Bluetooth button on the interface. Bluetooth can automatically connect.

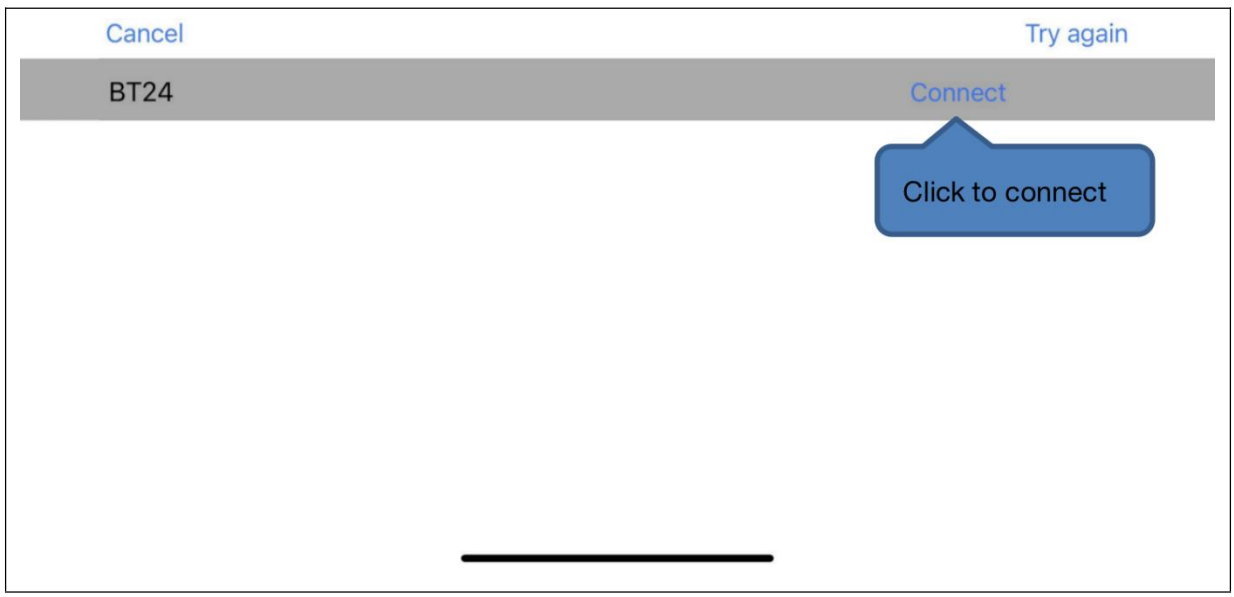


② **Connect manually**

Open the APP interface and click the Bluetooth button, as shown below.

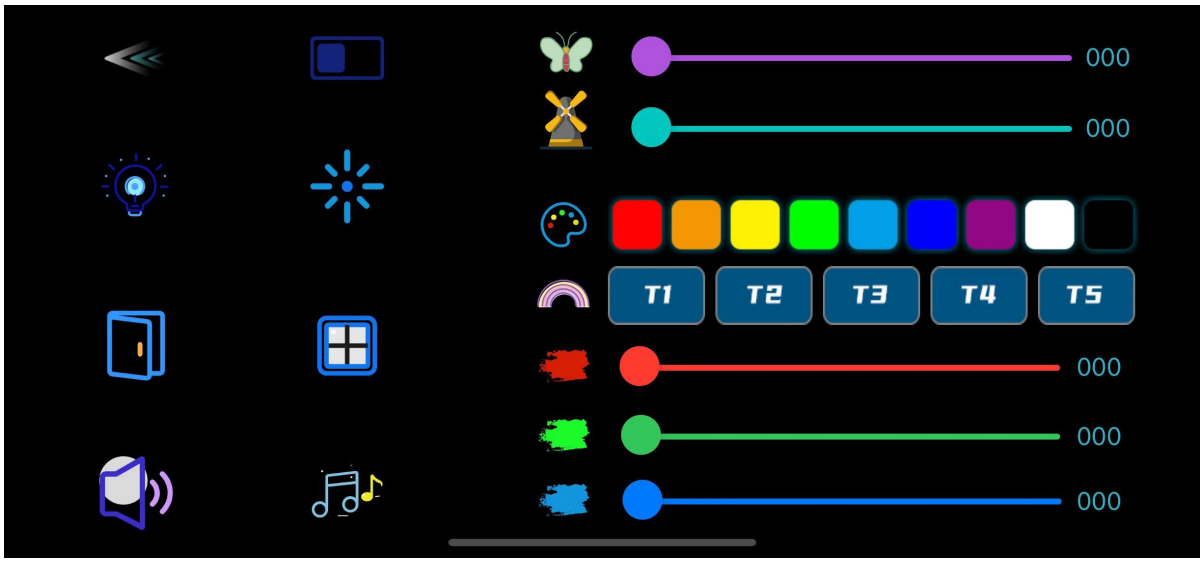


Bluetooth name is BT24 or JDY-23.



③ **Send an order**

After the connection is successful, you can click the button on the interface to print the received values in the Arduino IDE serial port monitor.



Open the serial port monitor of Arduino IDE to see the received Bluetooth button values.

17.2 Mobile phone APP controls LED

In the last lesson, we know the button value of the mobile phone APP, so we can use the light button on the APP interface to control the LED of the intelligent windmill.

(1) Example code

```
char bleVal = "";
#define ledPin 3
```

```
void setup() {
  Serial.begin(9600);
  pinMode(3, OUTPUT);
}

void loop() {
  while(Serial.available() > 0)  //Judge whether the serial port area receives value
  {
    String bleVal1 = Serial.readStringUntil('*'); //Read the string received by the Bluetooth until "*" is received
    if(bleVal1[0] == '$')  //If the first character of the string is "$"
    {
      bleVal = bleVal1[1]; //The second character of the string is the command we need and assign it to bleVal
      Serial.println(bleVal);
      if(bleVal == 'a')  //If the command received is "a"
      {
        digitalWrite(ledPin, HIGH); //Turn on the LED
      }
      if(bleVal == 'b')  //If the command received is "b"
      {
        digitalWrite(ledPin, LOW); //Turn off the LED
      }
    }
  }
}
```

(2) Experimental phenomenon

Turn on the Bluetooth of the windmill. After the mobile phone APP is successfully connected to Bluetooth, click the LED button on the APP interface once, and the LED of the windmill will light up. Click the button again, and the LED will turn off.

Small thought: *Try to write the code to control the windmill rotation in the mobile phone by yourself.*

17.3 APP controls multiple functions

(1) Example code

The code is relatively long. Open the sample code (17_3_BLE_multi_function) provided by us using Arduino IDE.

17_3_BLE_multi_function

(2) Experimental operation

Operate according to the following figures: