LROBRUYA

Basic Starter Kit for Arduino Uno (CH340)

Content

Preface

Company Profile

Founded in 2014, Shenzhen Lonten Technology Co., Ltd. focuses on the design, research production of Electronics Module for robotics related products. Consisting of professional researchers and skilled engineers, our R&D team constantly strives for creative function and excellent user experience. The company's R&D investments on arduino kits raspberry pi kits, as well as 3D printer and robots that back up STEAM education.

Customer Service

Our self-owned factory is certificated with BSCI and SO, covering an area of 5,000 square meters, and achieving an annual production capacity of over 10,000 units. Our products are all certified to CE, FCC, and ROHS standards, have exported to more than 100 countries including, but not limited to France, the United States of America, Australia, Russia, the United Kingdom, Germany, Singapore, Egypt, and India, bringing technological innovation to all walks of life.

By the way, We also look forward to hearing from you and any of your critical comment or suggestions. Pls email us by lonten3@qq.com or info@lontentech.com，if you have any questions or suggestions.

As a continuous and fast growing company. We keep striving our best to offer you excellent products and quality service.

Our Store

store: https://www.lontentech.com/

Brand: LONTEN

Product Catalog

https://www.lontentech.com/collections/sensor-module

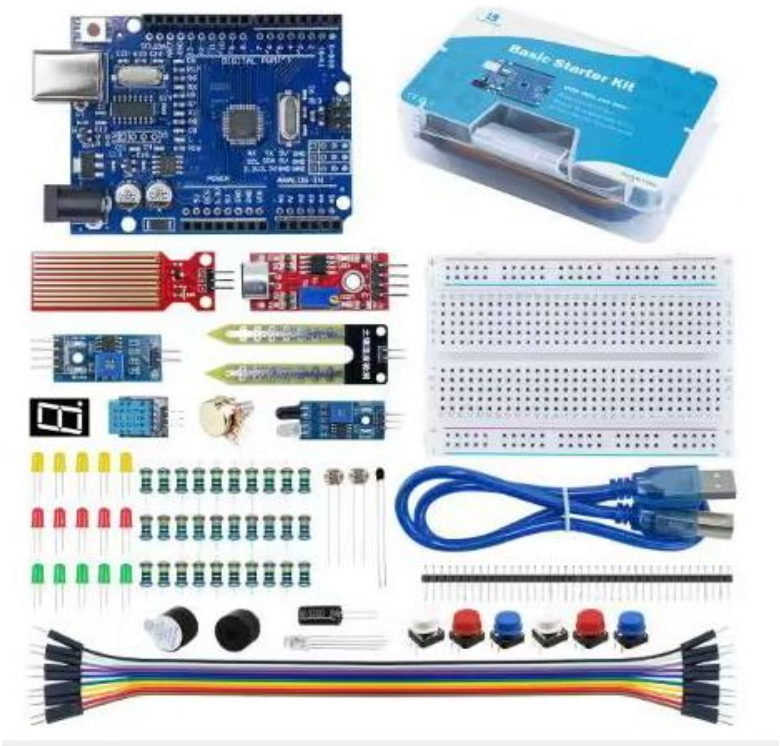
Tutorial

This tutorial include codes, libraries and detailed user documentation. It is designed for beginners. You will learn all the basic knowledge about how to use Arduino controller board, sensors and components.

LROBRUYA

Kit Introduction

This is a basic start learning kit for Arduino, rolled out by LONTEN.



This kit walks you through the basics of using the Arduino in a hands-on way. You'll learn through building several creative projects. Starting the basics of electronics, to more complex projects, the Kit will help you control the physical world with sensor and actuators.

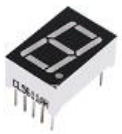
Now, let's start from the basic lessons.

LONTEN

Kit List

Basic Starter Kit

https://www.lontentech.com/pages/contact-us

 R3 CH340 1PCS	 Breadboard 1PCS	 USB Cable 1PCS	 Obstacle Avoidance Module 1PCS
 Soil Humidity Sensor 1PCS	 Sound Sensor Module 1PCS	 Water Lever Detection Sensor Module 1PCS	 1 Dight 7-segment Display 1PCS
 DHT11 1PCS	 Passive Buzzer 1PCS	 Active Buzzer 1PCS	 Potentiometer(10K) 1PCS
 Button Switch 6PCS	 3 Kinds of Resistor 30pcs	 M-M Dupont Wire 10PCS	 Photoresistor 2PCS
 Tilt Switch 1PCS	 Thermistor 1PCS	 RGB LED 1PCS	 Yellow/Red/Green Led each one 5PCS

How to Install Arduino IDE

Introduction

The Arduino Integrated Development Environment (IDE) is the software side of the Arduino platform.

In this Project, you will learn how to setup your computer to use Arduino and how to set about the Projects that follow.

The Arduino software that you will use to program your Arduino is available for Windows, Mac and Linux. The installation process is different for all three platforms and unfortunately there is a certain amount of manual work to install the software.

STEP 1: Go to https://www.arduino.cc/en/software.

The version available at this website is usually the latest version, and the actual version may be newer than the version in the picture.

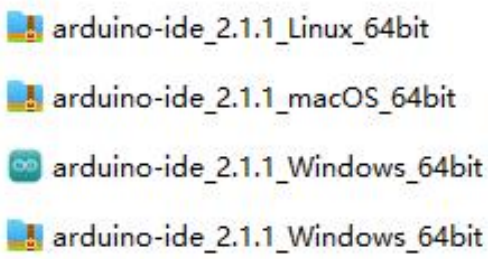
STEP2：Download the development software that is compatible with the operating.

system of your computer. Take Windows as an example here.

Click Windows Win 10 and newer,64 bits.

Click JUST DOWNLOAD.

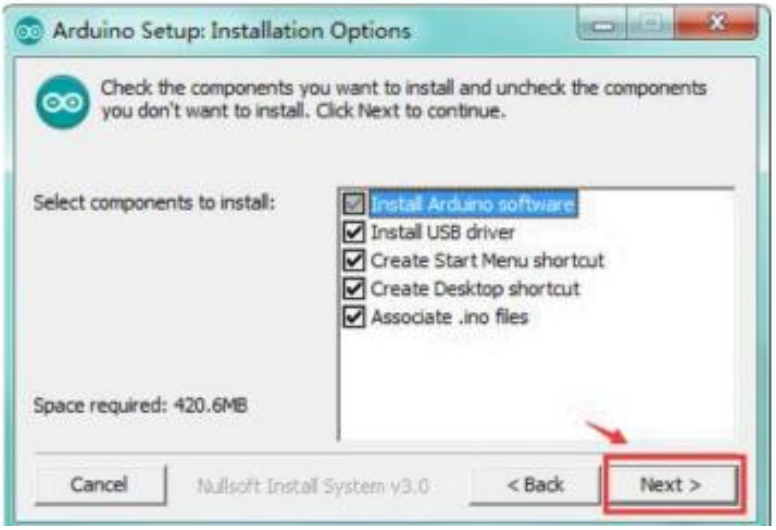
Also version 2.1.1 is available in the material we provided, and the versions of our materials are the latest versions when this course was made.



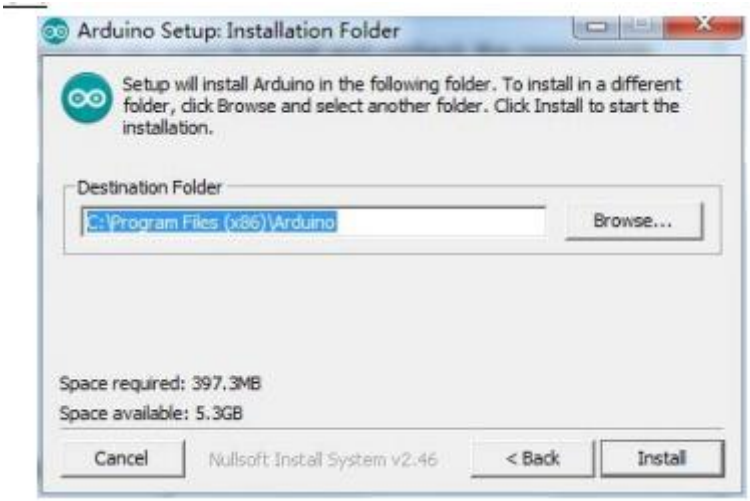
Installing Arduino (Windows)

Install Arduino with the exe. Installation package.

Click I Agree to see the following interface.

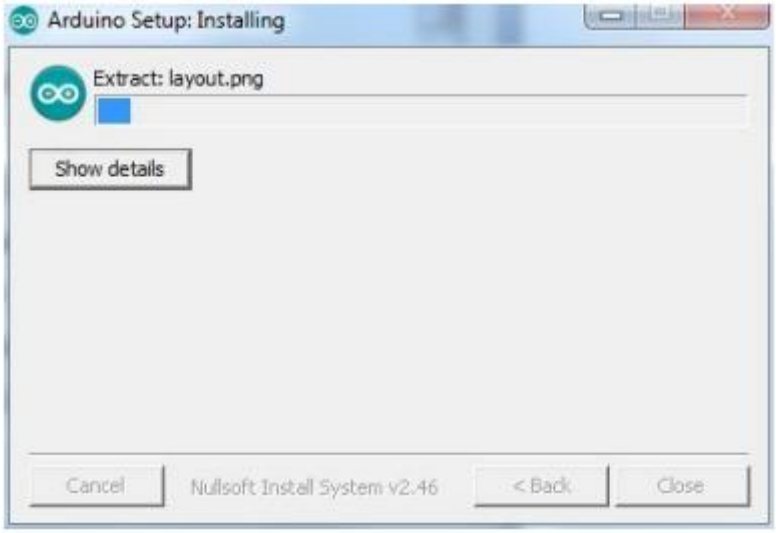


Click Next

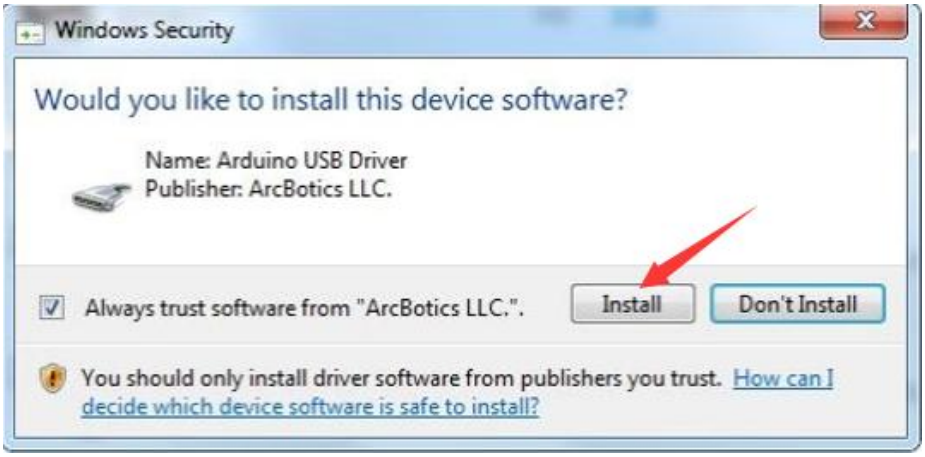


You can press Browse… to choose an installation path or directly type in the directory you want.

Click Install to initiate installation

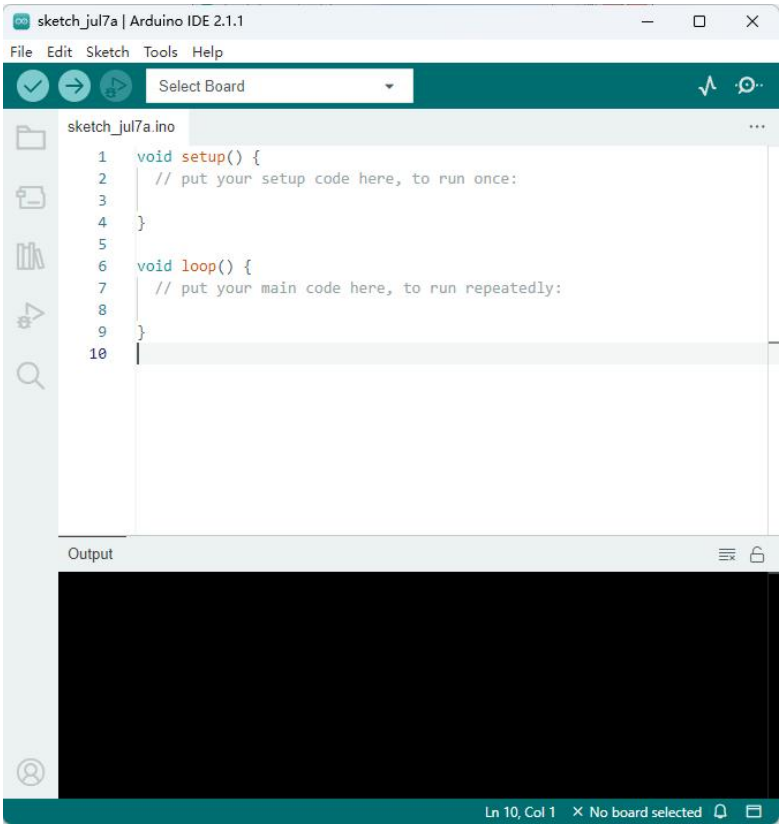


Finally, the following interface appears, click Install to finish the installation.



Next, the following icon appears on the desktop

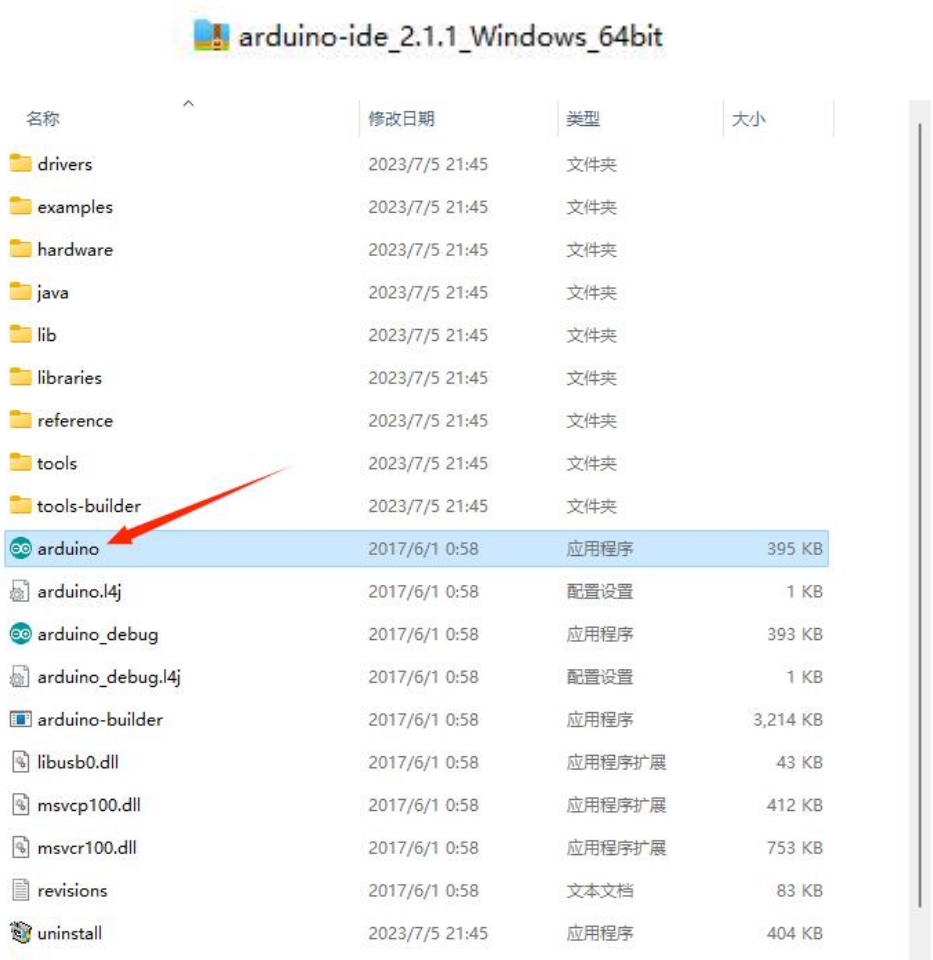
Double-click to enter the desired development environment

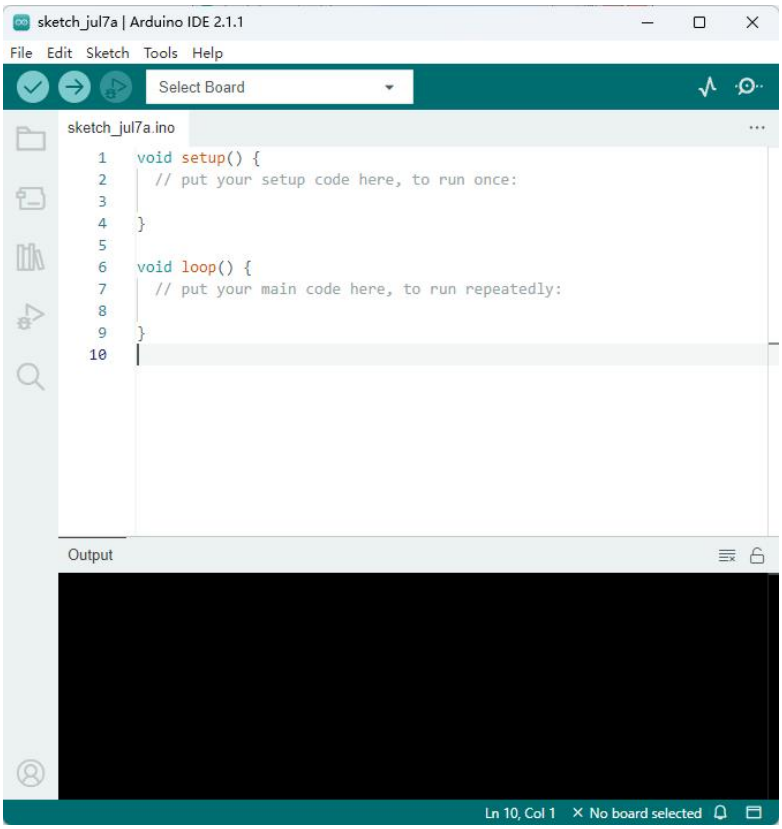


You may directly choose the installation package for installation and

skip the contents below and jump to the next section. But if you want to learn some methods other than the installation package, please continue to read the section.

Unzip the zip file downloaded, Double-click to open the program and enter the desired development environment.





Installing Arduino (Mac OS X)

Download and Unzip the zip file, double click the Arduino.app to enter Arduino IDE; the system will ask you to install Java runtime library if you don't have it in your computer. Once the installation is complete you can run the Arduino IDE.

Installing Arduino (Linux)

You will have to use the make install command. If you are using the Ubuntu system, it is recommended to install Arduino IDE from the software center of Ubuntu.

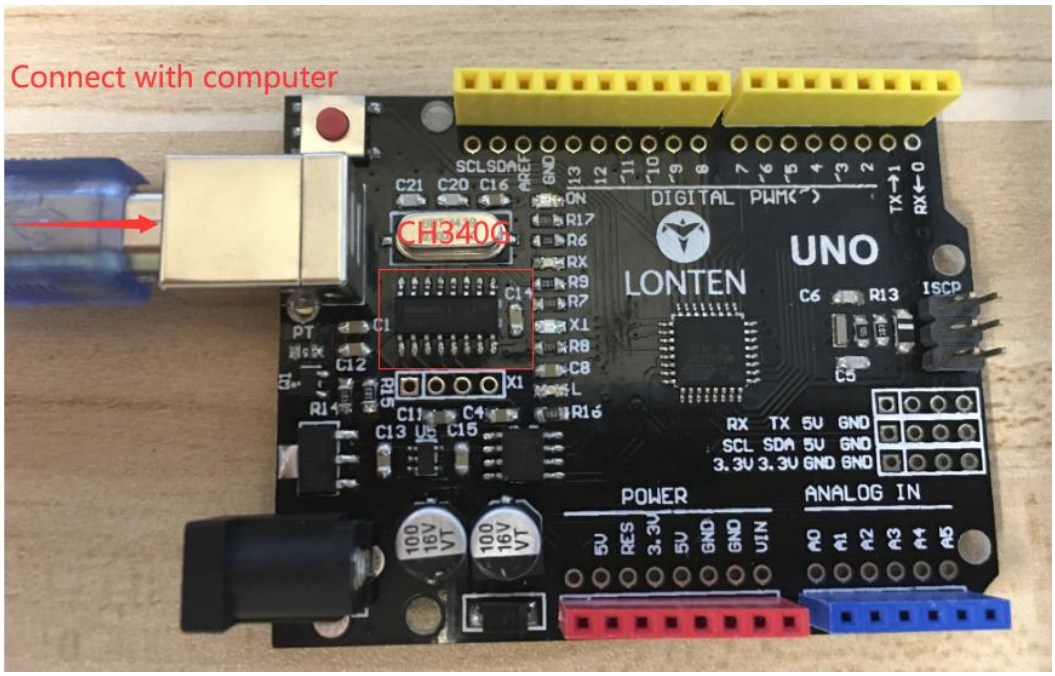
LROBRUYA

arduino-ide_2.1.1_Linux_64bit

How to Install Arduino Driver

For Windows

Arduino UNO(serial conversion chip is CH340G)

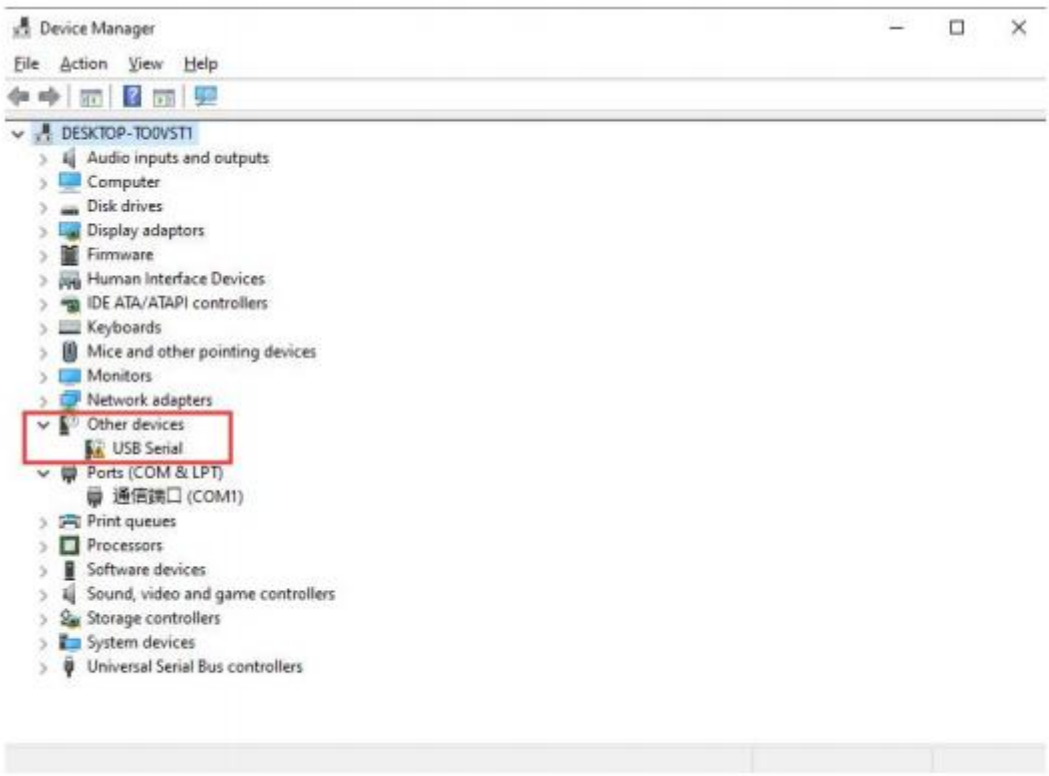


The USB to serial port chip of this control board is CH340G. So you need to install the driver for the chip. You can click the driver file here.

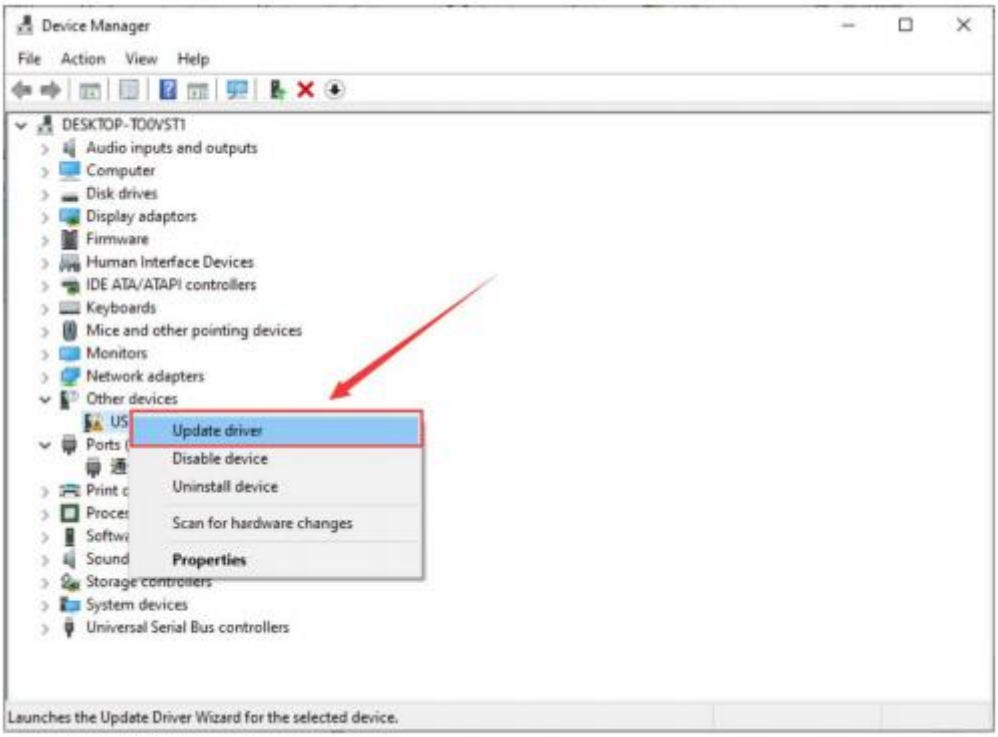
In different systems, the driver installation is similar. Here we start to install the driver on the Win10 system. You can find the "USB_Drive_CH341_3_1" folder in the information we provide, this is the driver file we want to install.

Plug one end of your USB cable into the Arduino UNO CH340 Board and the other into a USB socket on your computer.

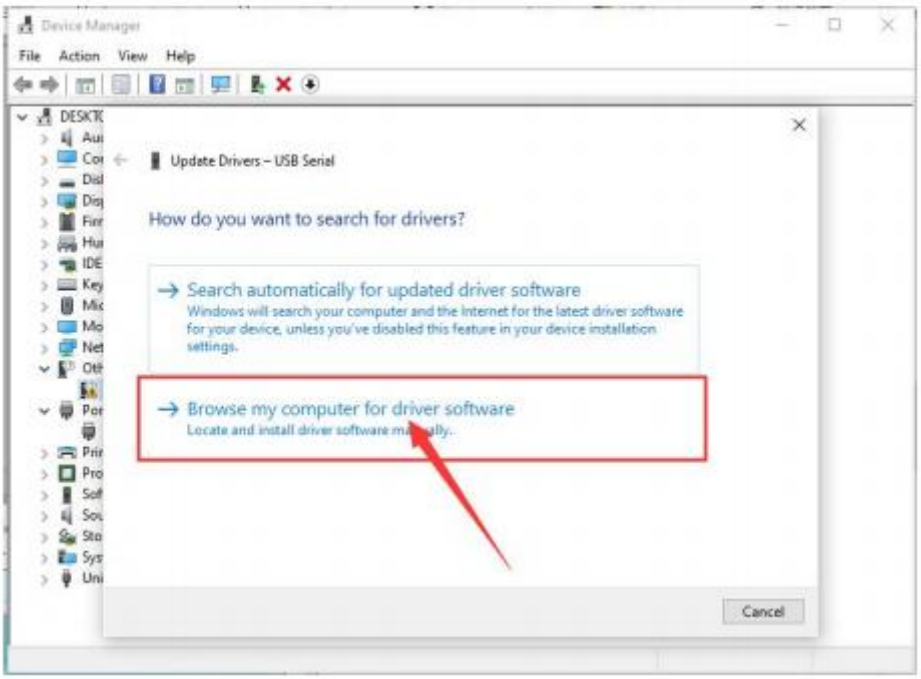
When you connect the Arduino UNO CH340 Board to your computer at the first time, right click your “My Computer”—>for “Properties”—>click the “Device manager”, under Other devices, you should see the “USB-Serial” or “Unknown device ”.Or you can search for "devi" in your computer, or you can open the device manager of your computer.



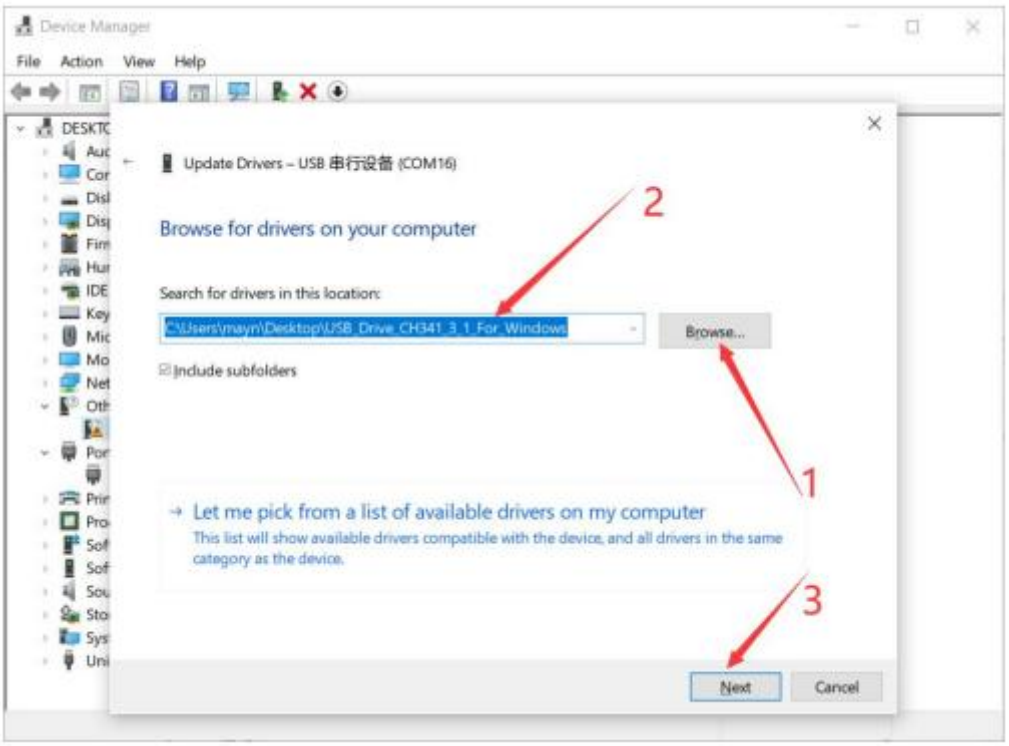
Then right-click on the device and select the top menu option (Update Driver Software...) shown as the figure below.



Then it will be prompted to either “Search Automatically for updated driver software” or “Browse my computer for driver software”. Shown as below. In this page, select “Browse my computer for driver software”.

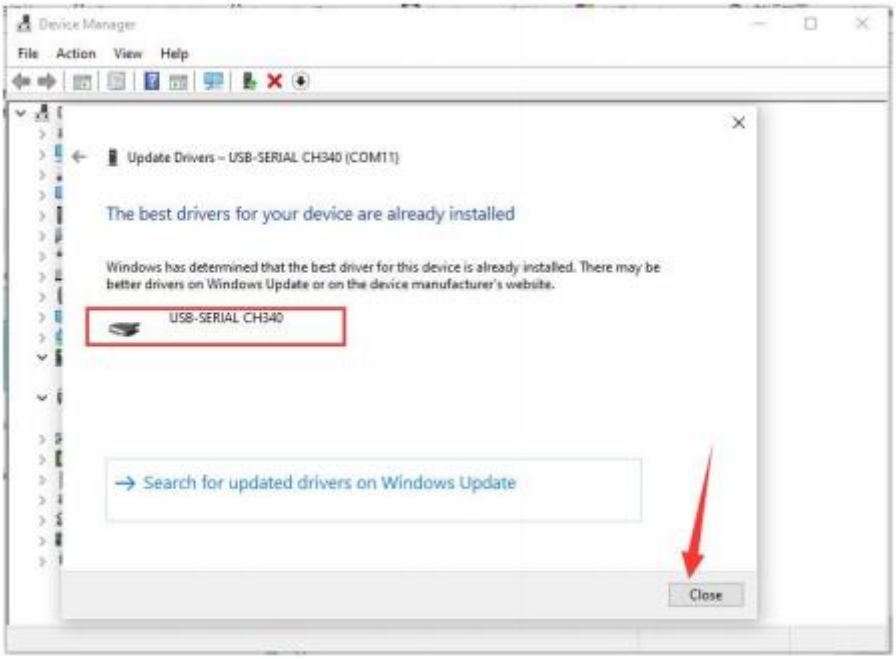


After that, select the browse option and navigate to the drive folder "USB_Drive_CH341_3_1", which can be found in the information we provide.(Note that the file path selects the location of the .For example, I store this driver file on the computer desktop, so the file path I choose is C:\Users\mayn\Desktop\USB_Drive_CH341_3_1_For_Windows)

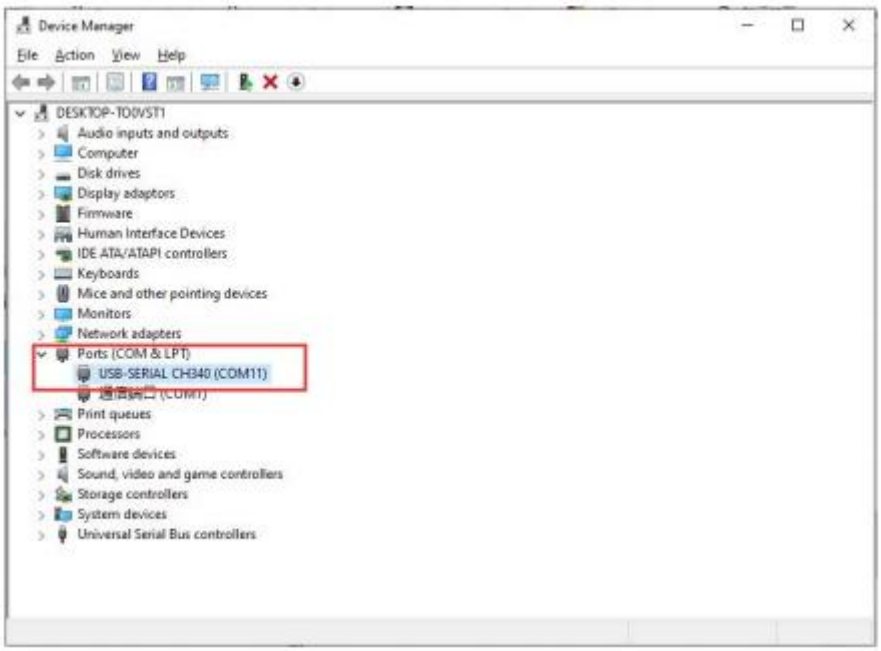


Once the software has been installed, you will get a confirmation message.

Installation completed, click “Close”.



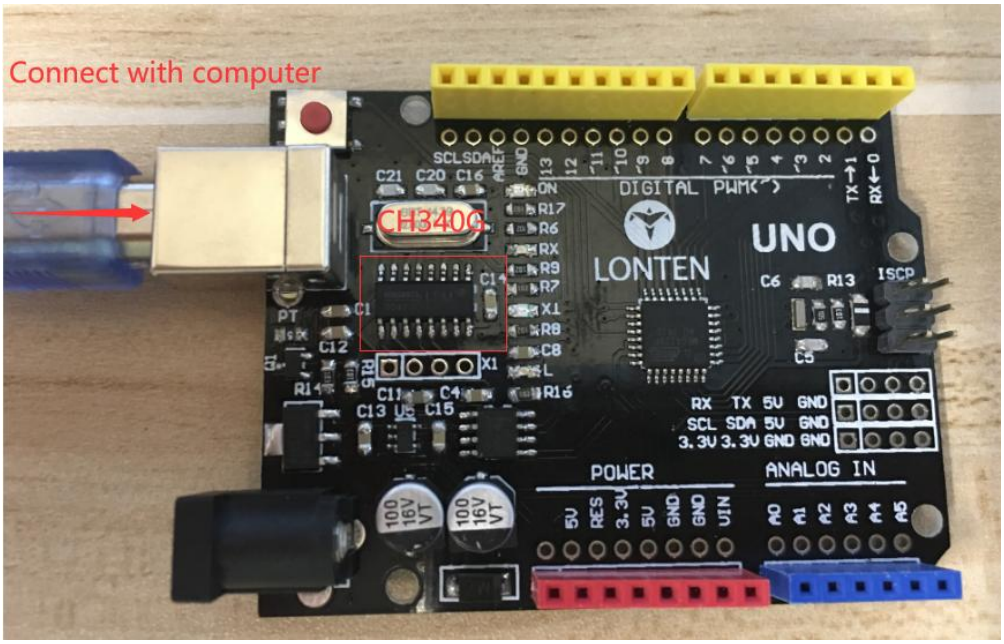
Up to now, the driver is installed well. Then you can right click “My Computer”—>for “Properties”—>click the “Device manager”, you should see the device as the figure shown below. Or you can search for "devi" in your computer, or you can open the device manager of your computer.



For MAC System

Arduino UNO(serial conversion chip is CH340G)

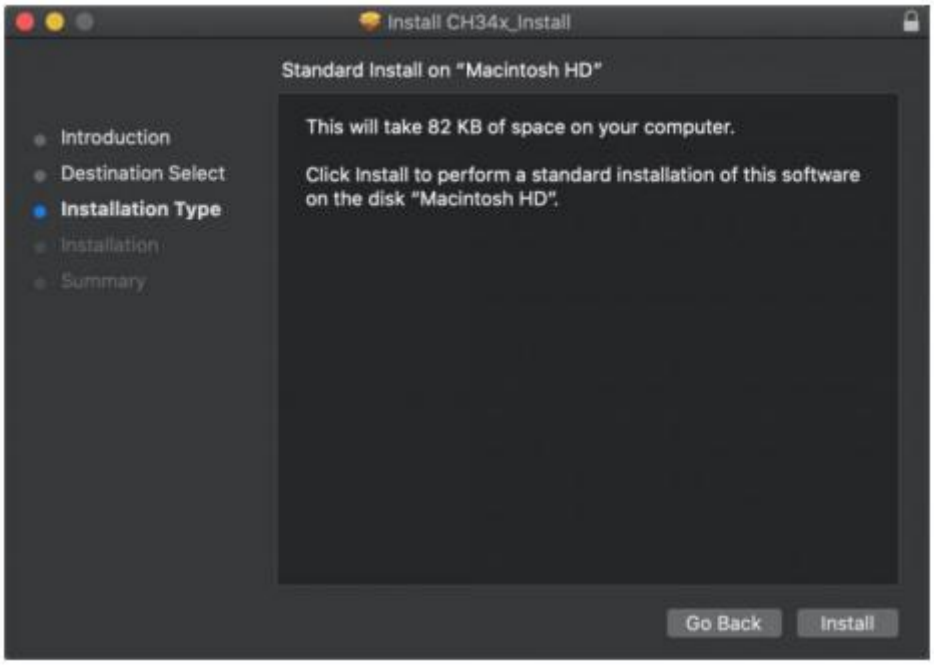
Plug one end of your USB cable into the Arduino UNO CH340 Board and the other into a USB socket on your computer.



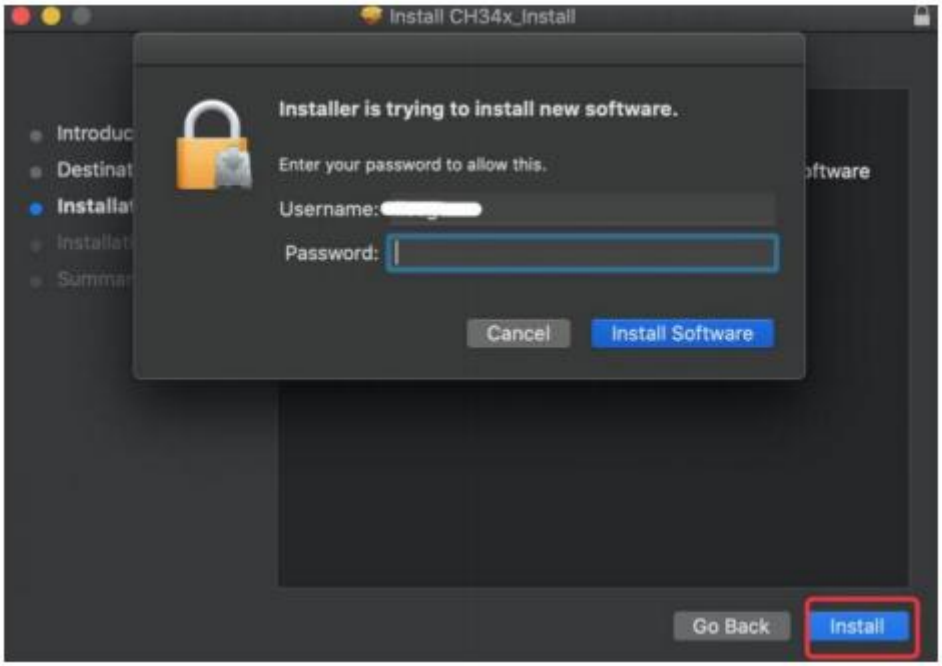
The driver file of the CH340G of the MAC system is provided in the tutorial data package.

Double-click installation package and tap Continue

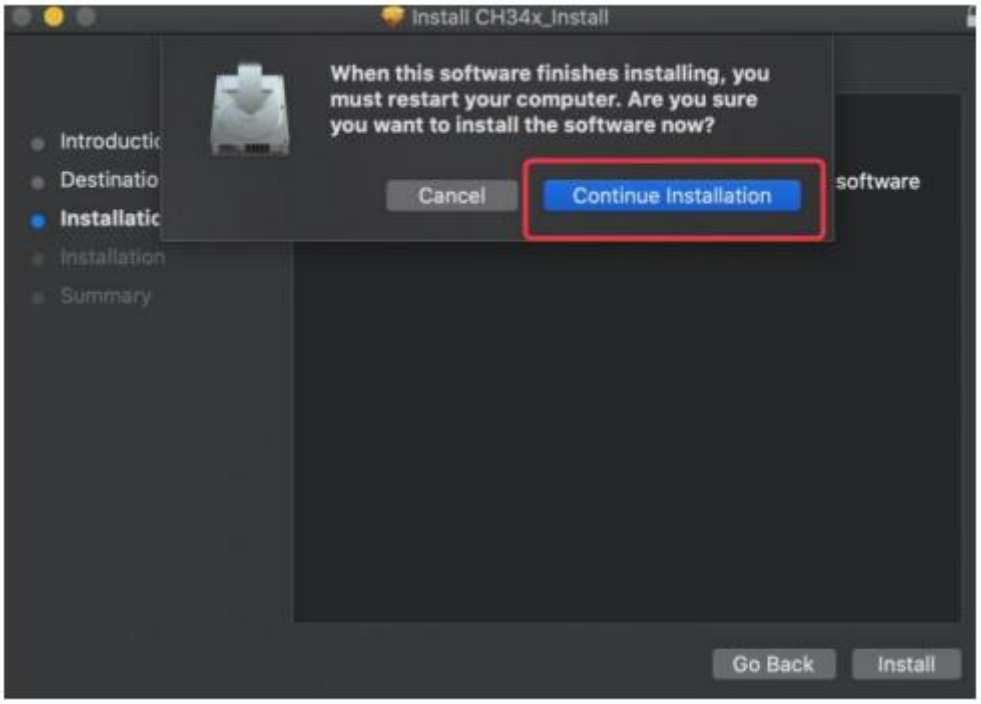
Click Install



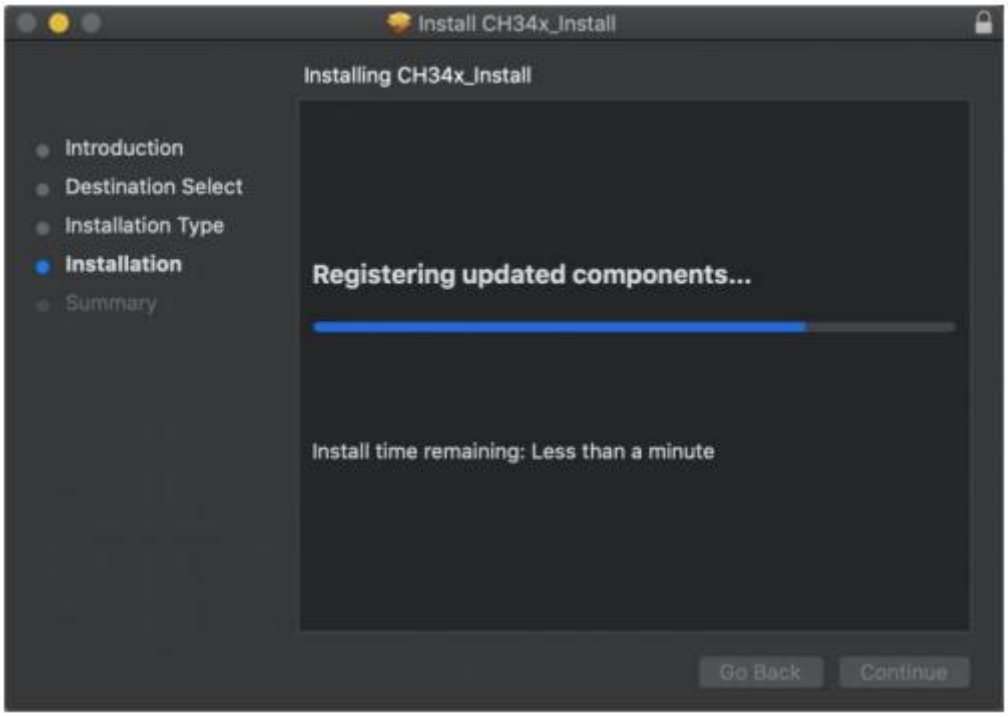
Input your user password and click Install Software



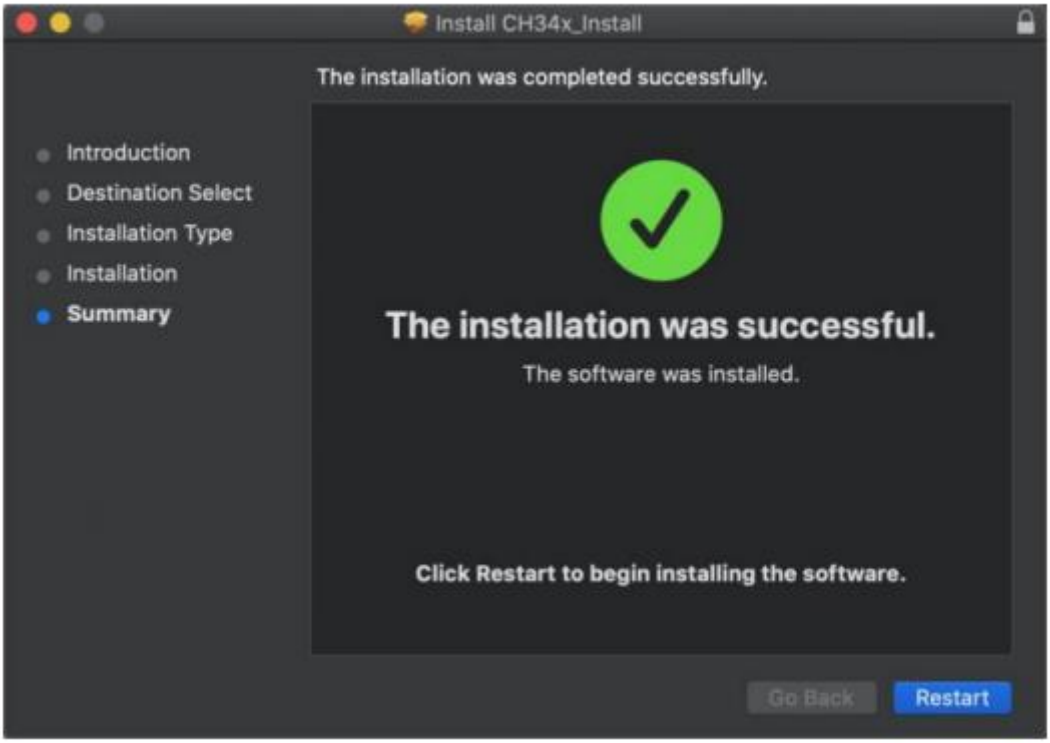
Tap Continue Installation



Wait to install



Click Restart after the installation is finished



How to Add Arduino Libraries

Installing Additional Arduino Libraries

Once you are comfortable with the Arduino software and using the built-in functions, you may want to extend the ability of your Arduino with additional libraries.

What are Libraries?

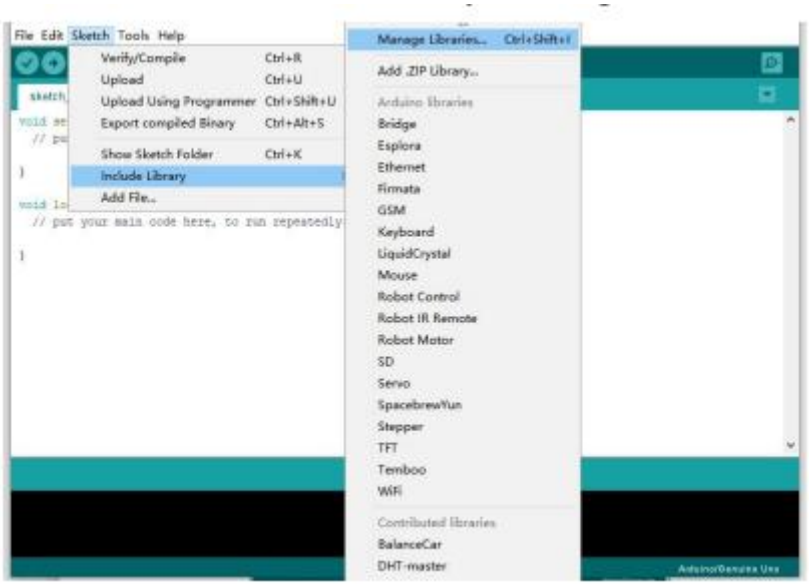
Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc. For example, the built-in Liquid Crystal library makes it easy to talk to character LCD displays. There are

hundreds of additional libraries available on the Internet for download. The built-in libraries and some of these additional libraries are listed in the reference. To use the additional libraries, you will need to install them.

How to Install a Library

Using the Library Manager

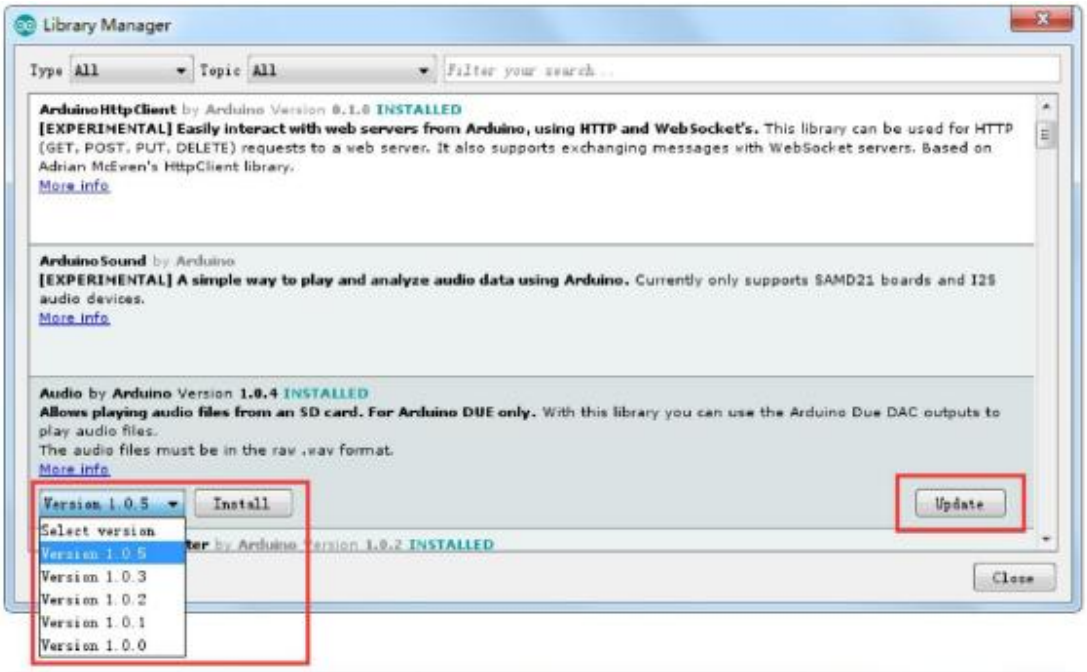
To install a new library into your Arduino IDE you can use the Library Manager (available from IDE version 1.8.0). Open the IDE and click to the "Sketch" menu and then Include Library > Manage Libraries.

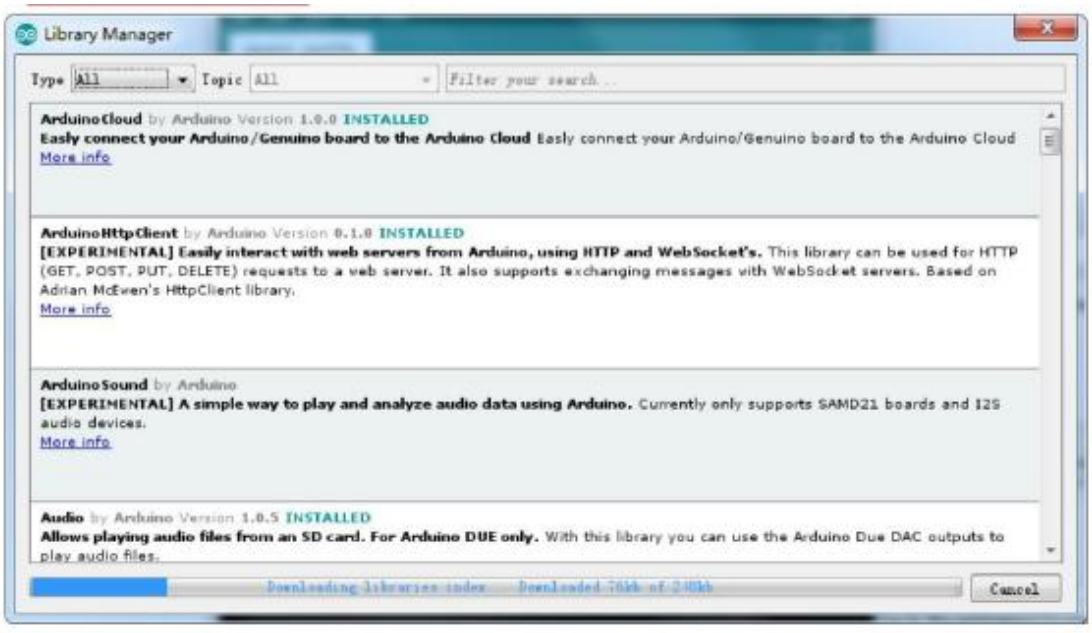


Then the library manager will open and you will find a list of libraries that are already installed or ready for installation. In this example we will install the Bridge library. Scroll the list to find it, then select the version of the library you want to install. Sometimes only one version of the

library is available. If the version selection menu does not appear, don't worry: it is normal.

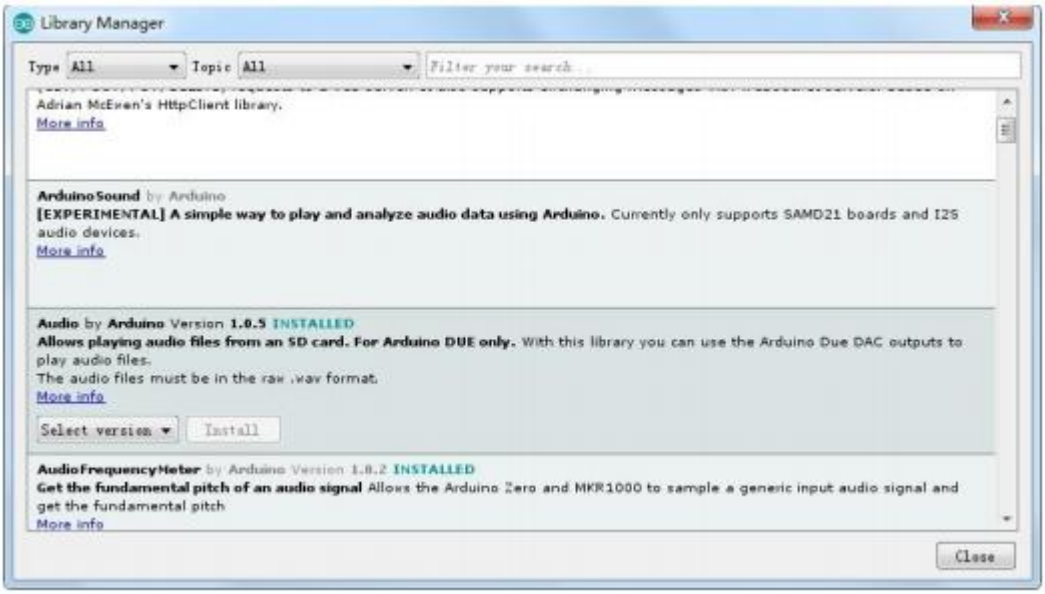
There are times you have to be patient with it, just as shown in the figure. Please refresh it and wait.





Finally click on install and wait for the IDE to install the new library.

Downloading may take time depending on your connection speed. Once it has finished, an Installed tag should appear next to the Bridge library. You can close the library manager.



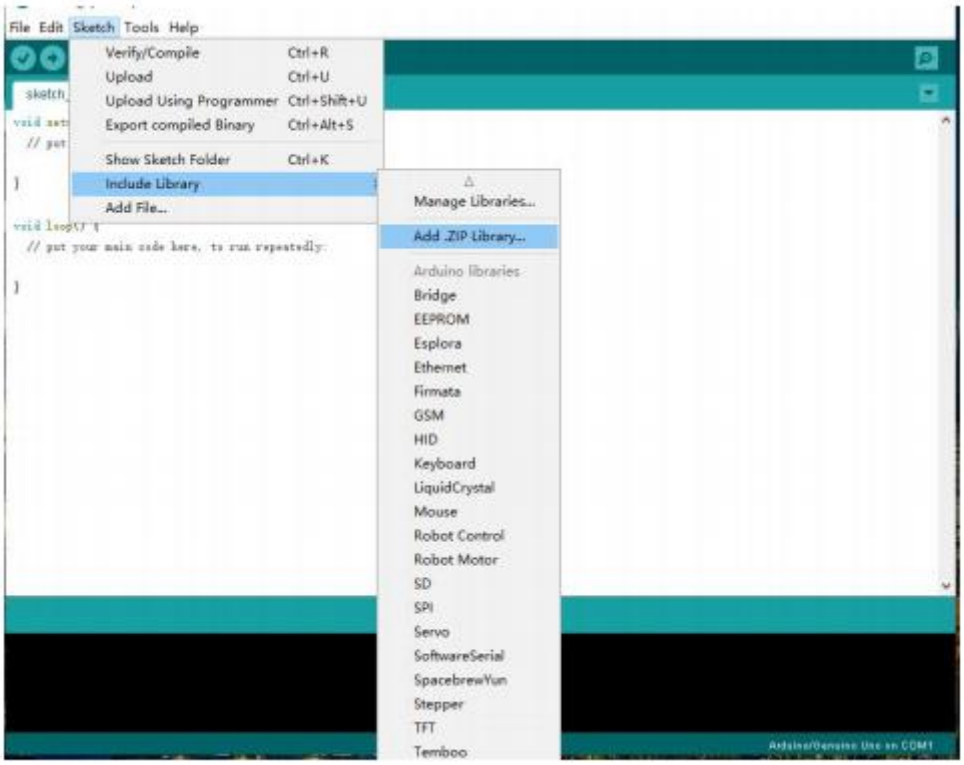
You can now find the new library available in the Include Library menu. If you want to add your own library open a new issue on Github.

Importing a .zip Library

Libraries are often distributed as a ZIP file or folder. The name of the folder is the name of the library. Inside the folder will be a .cpp file, a .h file and often a keywords.txt file, examples folder, and other files required by the library. Starting with version 1.0.5, you can install 3rd party libraries in the IDE. Do not unzip the downloaded library, leave it as is.

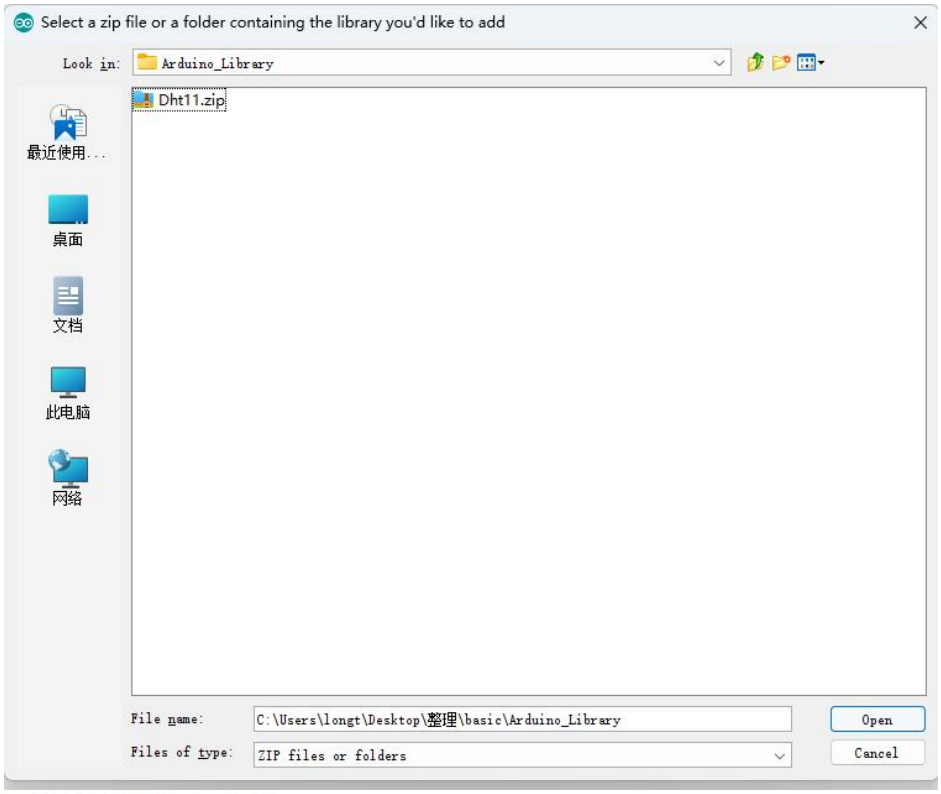
In the Arduino IDE, navigate to Sketch > Include Library. At the top of

the drop down list, select the option to "Add .ZIP Library".



You will be prompted to select the library you would like to add.

Navigate to the .zip file's location and open it.


Select a zip file or a folder containing the library you'd like to add
Look in:
Arduino_Library
Dht11.zip
最近使用...
桌面
文档
此电脑
网络
File name:
C:\Users\longt\Desktop\整理\basic\Arduino_Library
Open
Files of type:
ZIP files or folders
Cancel


File Edit Sketch Tools Help
sketch_nov16b
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}
Library added to your libraries. Check "include library" menu
Arduino/Genuino Uno

Return to the Sketch > Import Library menu. You should now see the library at the bottom of the drop-down menu. It is ready to be used in your sketch. The zip file will have been expanded in the libraries folder in your Arduino sketches directory. NB: the Library will be available to use in sketches, but examples for the library will not be exposed in the File > Examples until after the IDE has restarted.

Blink Test

Overview

In this Project, you will learn how to program your UNO R3 controller board to blink the Arduino's built-in LED, and how to download programs by basic steps.

Component Required:

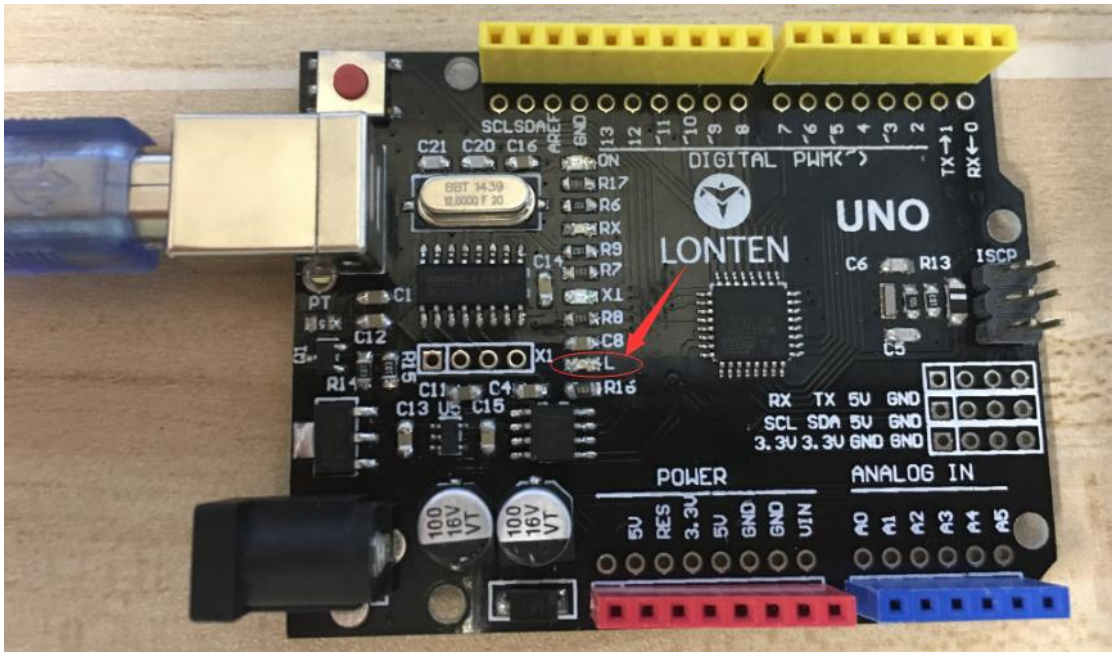
LONTEN Uno R3 Board* 1

Principle

The UNO R3 board has rows of connectors along both sides that are used to connect to several electronic devices and plug-in 'shields' that extends its capability.

It also has a single LED that you can control from your sketches. This LED is built onto the UNO R3 board and is often referred to as the 'L'

LED as this is how it is labeled on the board.

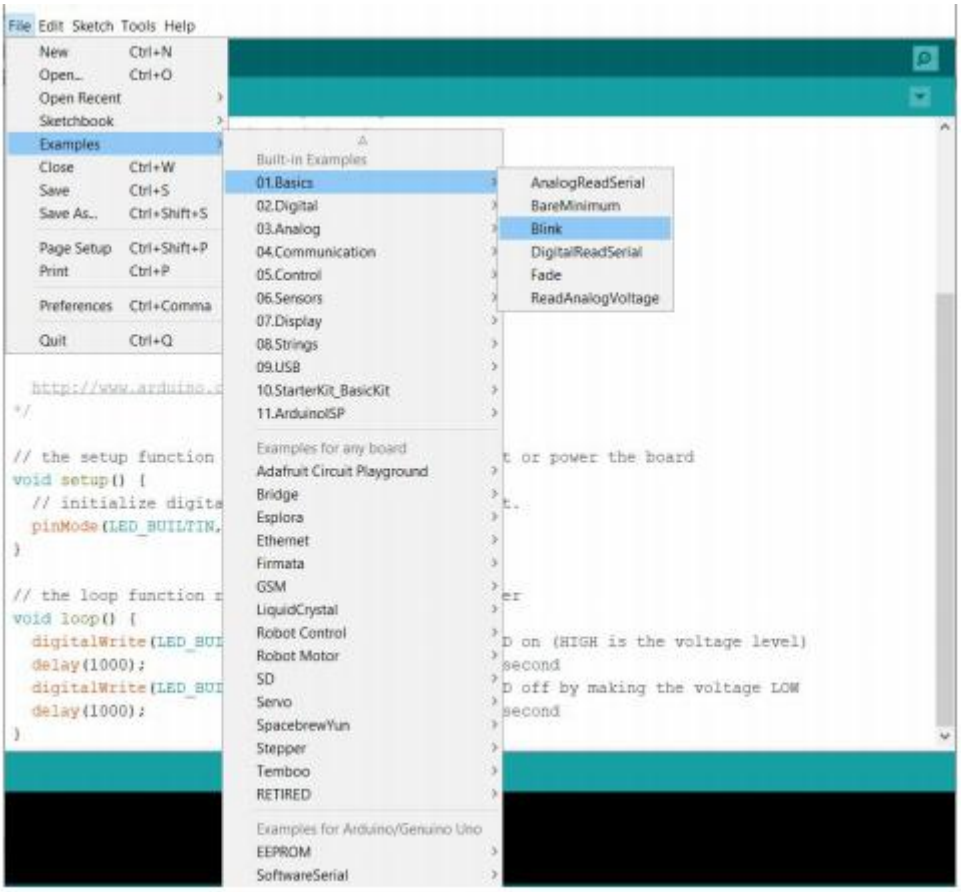


In this Project, we will reprogram the UNO board with our own Blink sketch and then change the rate at which it blinks.

In the previous chapter-How to install Arduino IDE, you set up your Arduino IDE and made sure that you could find the right serial port for it to connect to your UNO board. The time has now come to put that connection to the test and program your UNO board.

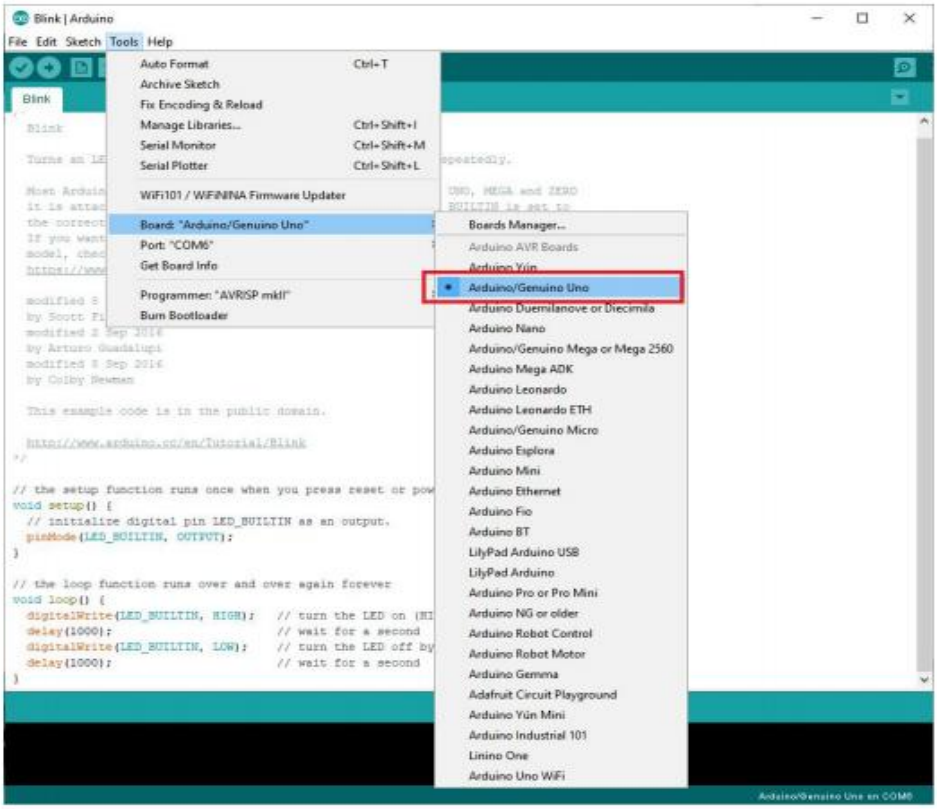
The Arduino IDE includes a large collection of example sketches that you can load up and use. This includes an example sketch for making the 'L' LED blink.

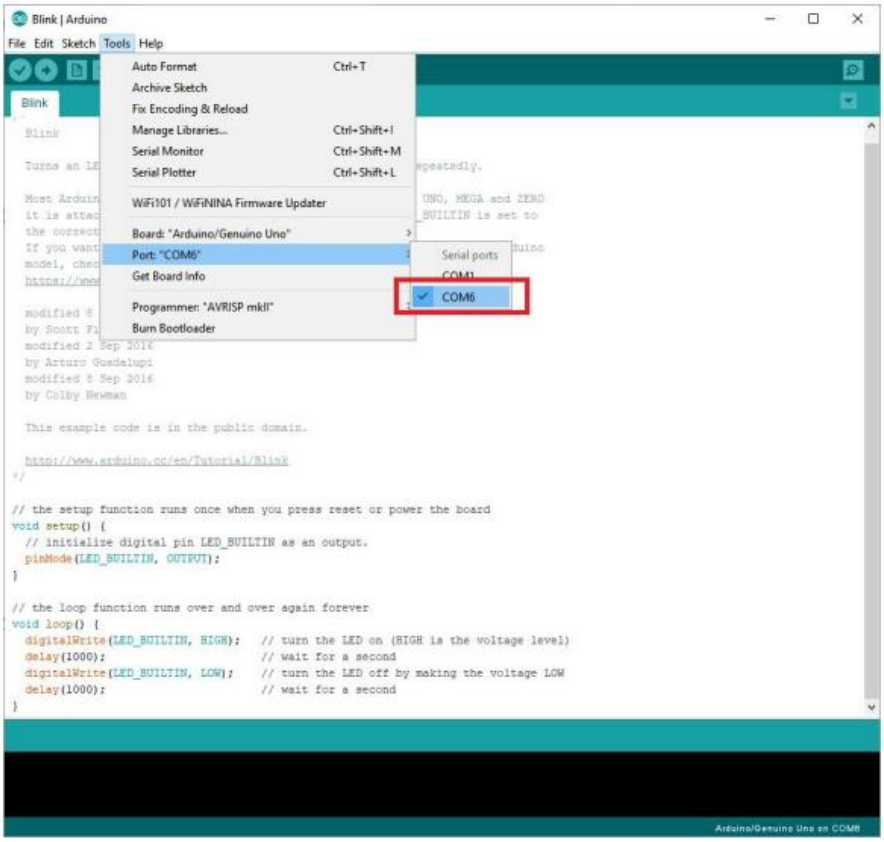
Load the 'Blink' sketch that you will find in the IDE's menu system under File > Examples > 01.Basics>Blink

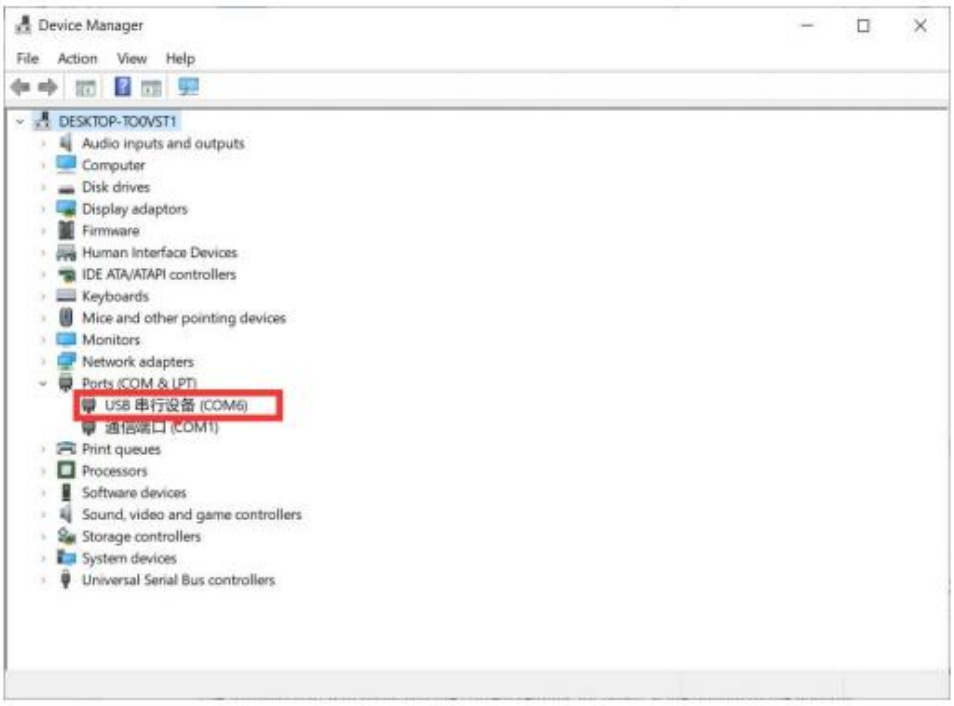


When the sketch window opens, enlarge it so that you can see the entire sketch in the window.

Attach your Arduino board to your computer with the USB cable and check that the 'Board Type' and 'Serial Port' are set correctly.


Blink | Arduino
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM6"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
Arduino/Genuino Mega or Mega 2560
Arduino Mega ADK
Arduino Leonardo
Arduino Leonardo ETH
Arduino/Genuino Micro
Arduino Esplora
Arduino Mini
Arduino Ethernet
Arduino Fio
Arduino BT
LilyPad Arduino USB
LilyPad Arduino
Arduino Pro or Pro Mini
Arduino NG or older
Arduino Robot Control
Arduino Robot Motor
Arduino Gemma
Adafruit Circuit Playground
Arduino Yún Mini
Arduino Industrial 101
Linino One
Arduino Uno WiFi

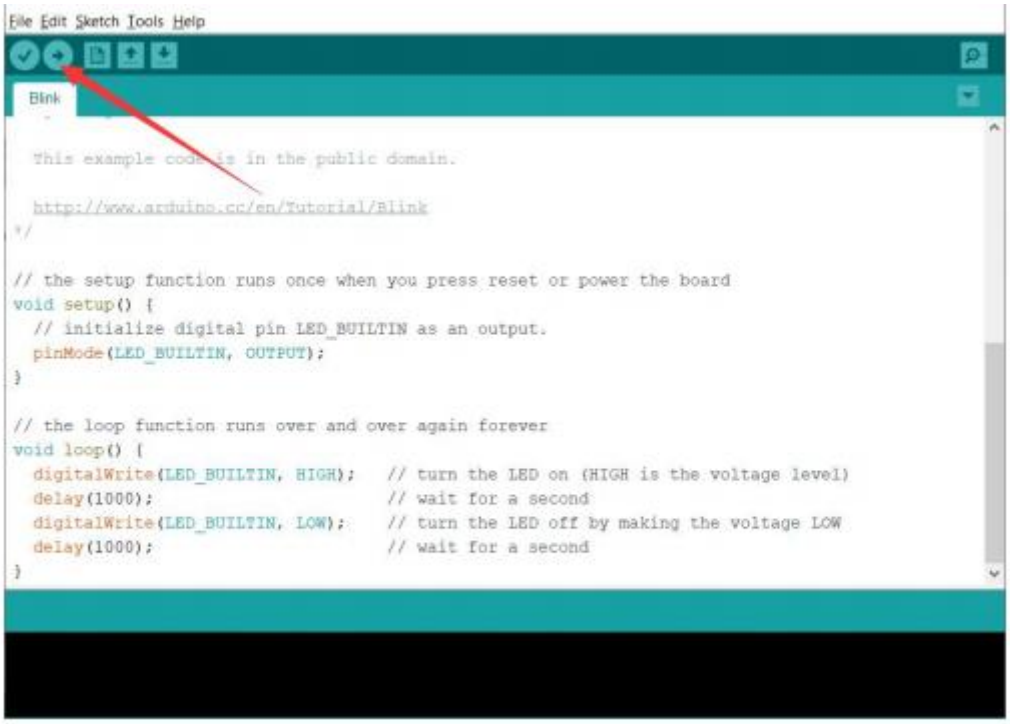

Blink | Arduino
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM6"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM1
COM6
// the setup function runs once when you press reset or power the board
void setup() {
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}



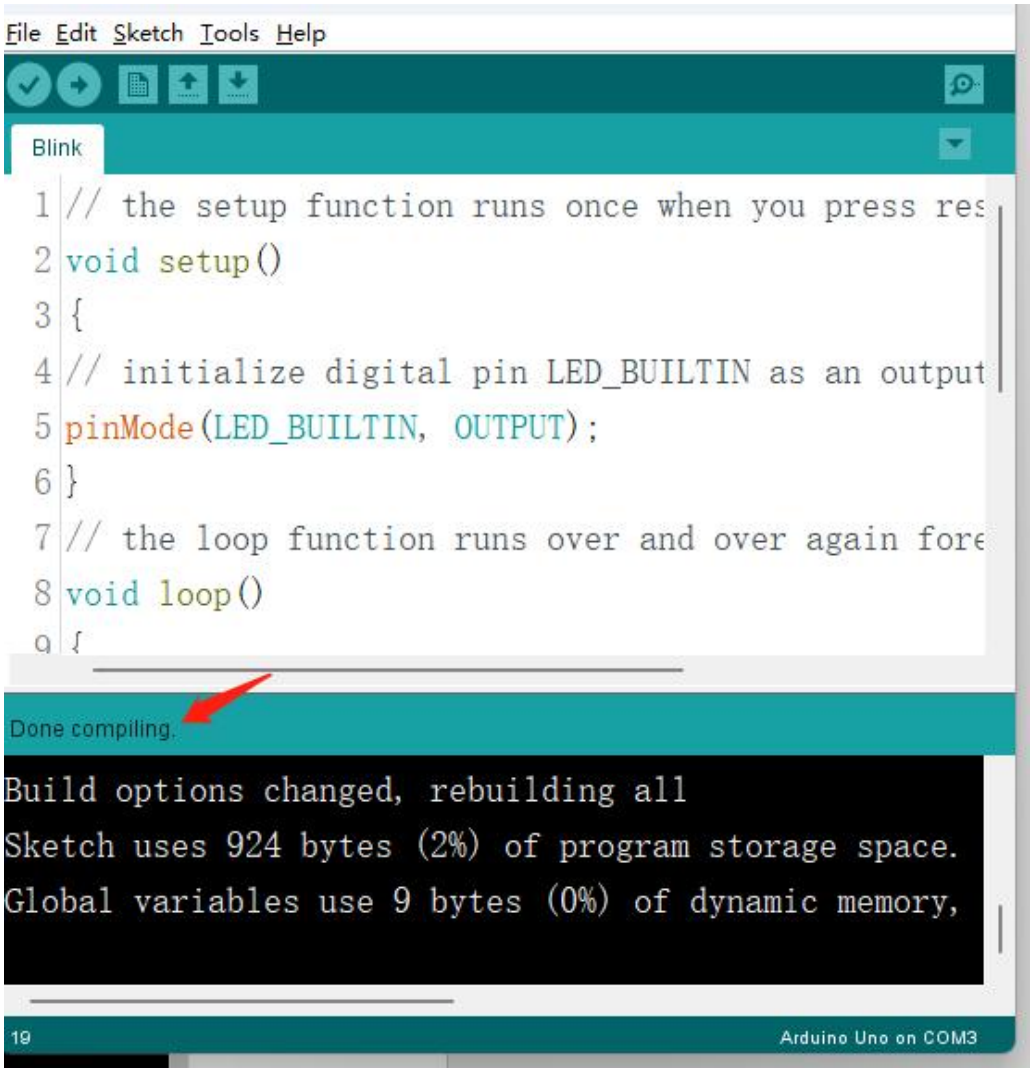
Note: The Board Type and Serial Port here are not necessarily the same as shown in picture. If you are using UNO, then you will have to choose Arduino UNO as the Board Type, other choices can be made in the same manner. And the Serial Port displayed for everyone is different, despite COM 6 chosen here, it could be COM3 or COM4 on your computer. A right COM port is supposed to be COMX (arduino XXX), which is by the certification criteria.

The Arduino IDE will show you the current settings for board at the bottom of the window.

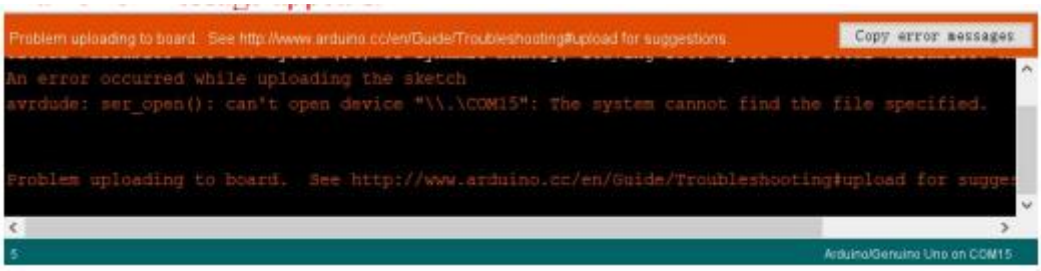
Click on the 'Upload' button. The second button from the left on the toolbar.



When the status bar prompts "Done uploading", it means the code upload is successful



If an error message appears.

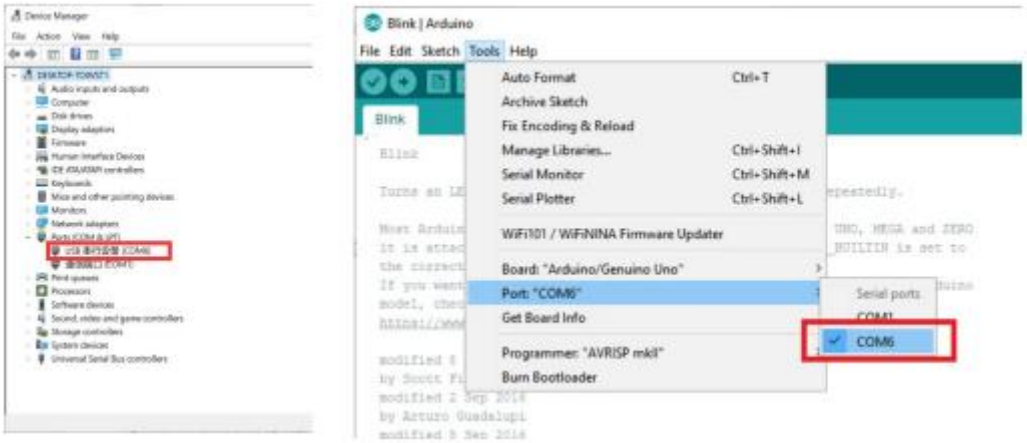


There can be several reasons:

1. The arduino uno driver software is not installed successfully, please

refer to the course for the installation steps: How to Install Arduino Driver.

2. The communication serial port selection of arduino uno is wrong; you can check the communication port COMx of your arduino uno in the computer in the device manager.

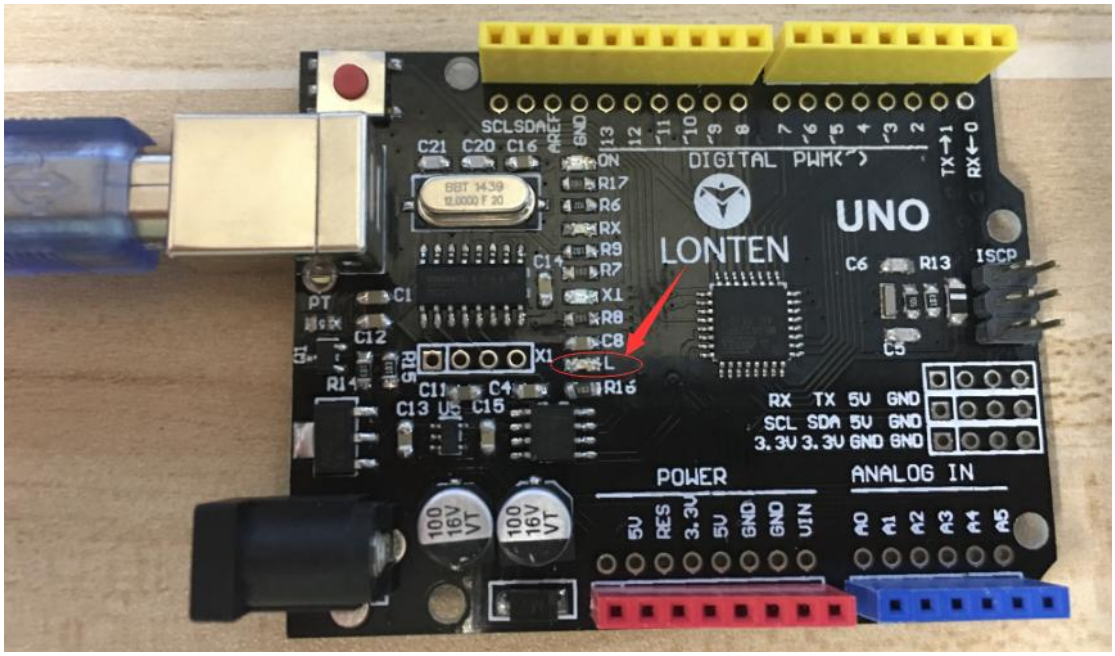


3. If your Arduino uno is connected to a Bluetooth module, it will occupy the communication serial port. You need to remove the Bluetooth module connection before uploading the code.

4. The USB data cable is not firmly connected. Check if there are any of the above problems. After correcting, follow the previous steps to re-operate.

Sample Program

```
// the setup function runs once when you press reset or power the board
void setup()
{
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop()
{
digitalWrite(LED_BUILTIN, HIGH);
// turn the LED on (HIGH is the voltage level)
delay(1000);
// wait for a second
digitalWrite(LED_BUILTIN, LOW);
// turn the LED off by making the voltage LOW
delay(1000);
// wait for a second
}
```



After the code is successfully uploaded, the "L" character LED will flash once per second. So far, you have completed the testing process of your first program.

Project 1 Hello World

Overview

As for starters, we will begin with something simple. In this project, you only need an Arduino and a USB Cable to start the "Hello World!" experiment. It is not only a communication test of your Arduino and PC, but also a primer project for you to have your first try in the Arduino world!

Component Required:

1.LONTEN Uno R3 Board *1

2.USB Cable *1

Arduino Serial Monitor (Windows, Mac, Linux)

The Arduino Integrated Development Environment (IDE) is the software side of the Arduino platform. And, because using a terminal is such a big part of working with Arduino and other microcontrollers, they decided to include a serial terminal with the software. Within the Arduino environment, this is called the Serial Monitor.

Sample Program

Open the program “Project_1_Hello_World.ino” in the folder Arduino_Code—>Project_1_Hello_World.

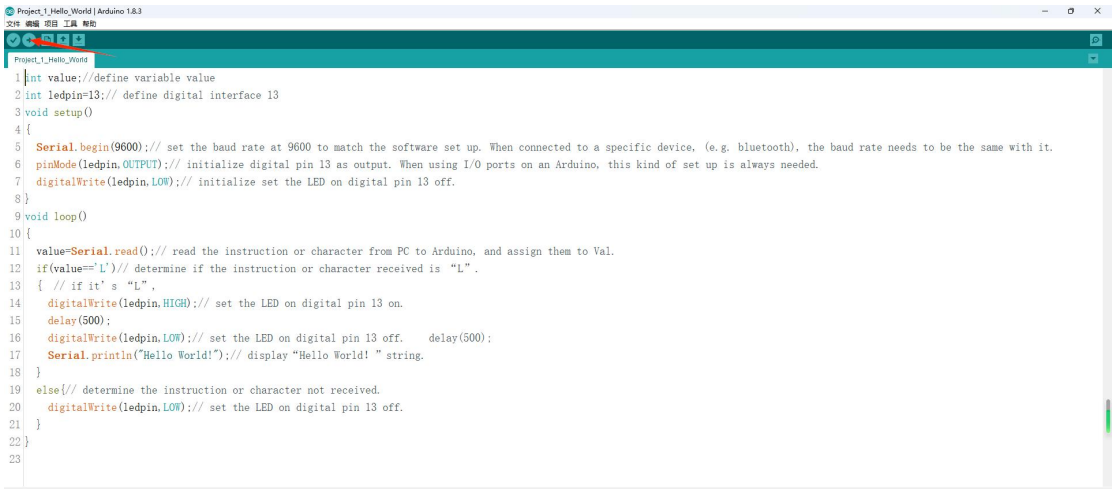
```
//////////////////////////////////////////////////////////
int value;//define variable value
int ledpin=13;// define digital interface 13
void setup()
{
   Serial.begin(9600);// set the baud rate at 9600 to match the software set up. When connected to a specific device, (e.g. bluetooth), the baud rate needs to be the same with it.
```

```
    pinMode(ledpin,OUTPUT);// initialize digital pin 13 as output. When using I/O ports on an Arduino, this kind of set up is always needed.
    digitalWrite(ledpin,LOW);// initialize set the LED on digital pin 13 off.
}
void loop()
{
    value=Serial.read();// read the instruction or character from PC to Arduino, and assign them to Val.
    if(value=='L')// determine if the instruction or character received is “L”.
    {    // if it’s “L”,
        digitalWrite(ledpin,HIGH);// set the LED on digital pin 13 on.
        delay(500);
        digitalWrite(ledpin,LOW);// set the LED on digital pin 13 off.
delay(500);
        Serial.println("Hello World!");// display“Hello World！”string.
    }
    else{// determine the instruction or character not received.
        digitalWrite(ledpin,LOW);// set the LED on digital pin 13 off.
    }
```

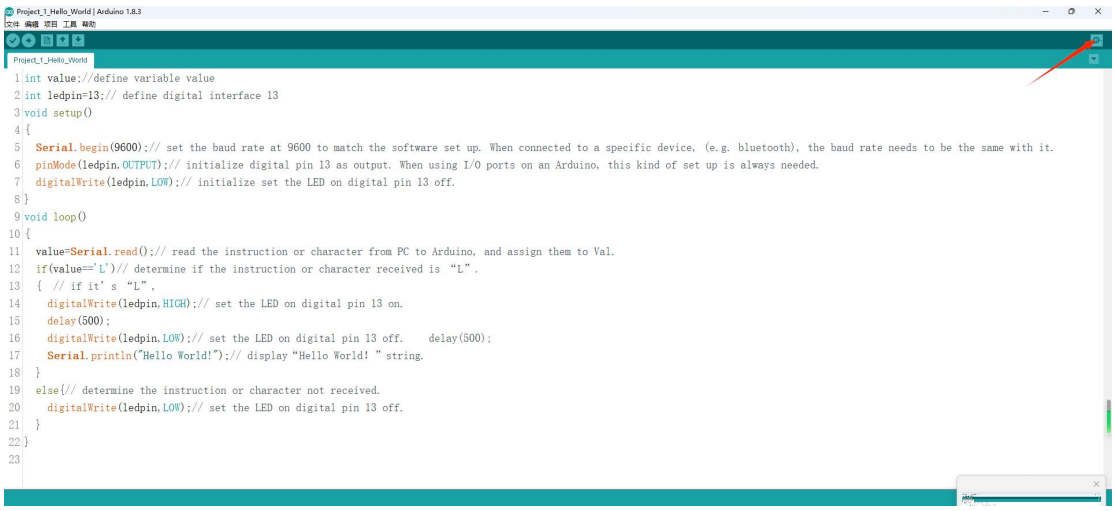
}//

Code Test

Select the correct COM communication serial port, click the button to upload the code. if you encounter an error during the code upload process, you can refer to the previous course “Blink Test”.

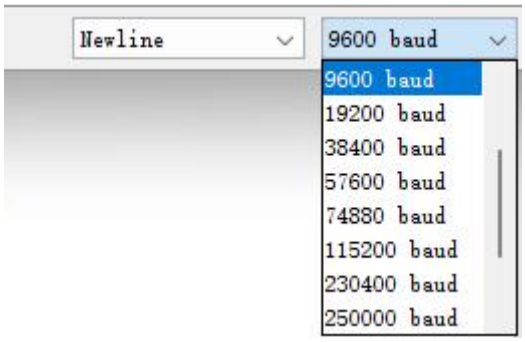


After completing the code upload, click the Serial Monitor icon to open it.

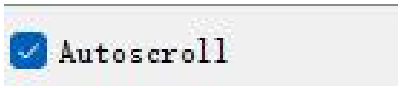


Click on the baud rate drop-down menu to select the correct baud rate.

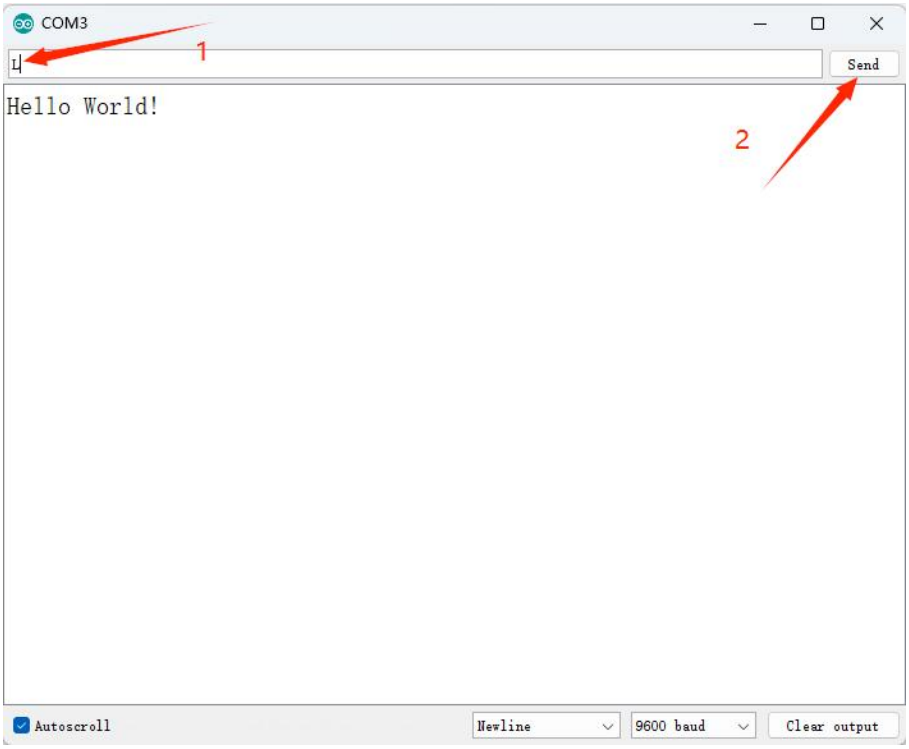
(9600 baud)



Last, you can set the terminal to Autoscroll or not by checking the box in the bottom left corner.



Click to open serial port monitor, input an “ L”, LED 13 will blink once, PC will receive the information from Arduino: Hello World!



Project 2 LED Blinking

Introduction

Blinking LED experiment is quite simple. In the “Hello World!” program, we have come across LED. This time, we are going to connect an LED to one of the digital pins rather than using LED13, which is soldered to the board. Except an Arduino and an USB cable, we will need extra parts as below.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3. Red M5 LED*1

4. 220Ω Resistor*1

5. Breadboard*1

6. Breadboard Jumper Wires* Several

LED:

LEDs make great indicator lights. They use very little electricity and they pretty much last forever.

In this Project, you will use perhaps the most common of all LEDs: a 5mm red LED. 5mm refers to the diameter of the LED. Other common sizes are 3mm and 10mm. You cannot directly connect an LED to a

battery or voltage source because 1) the LED has a positive and a negative lead and will not light if placed the wrong way and 2) an LED must be used with a resistor to limit or ‘choke’ the amount of current flowing through it; otherwise, it will burn out!

If you do not use a resistor with an LED, then it may well be destroyed. almost immediately, as too much current will flow through, heating it and destroying the ‘junction’ where the light is produced.

There are two ways to tell which is the positive lead of the LED and which the negative.

Firstly, the positive lead is longer.

Secondly, where the negative lead enters the body of the LED, there is a flat edge to the case of the LED.

If you happen to have an LED that has a flat side next to the longer lead, you should assume that the longer lead is positive.

RESISTORS:

As the name suggests, resistors resist the flow of electricity. The higher the value of the resistor, the more it resists and the less electrical current.

will flow through it. We are going to use this to control how much electricity flows through the LED and therefore, how brightly it shines.

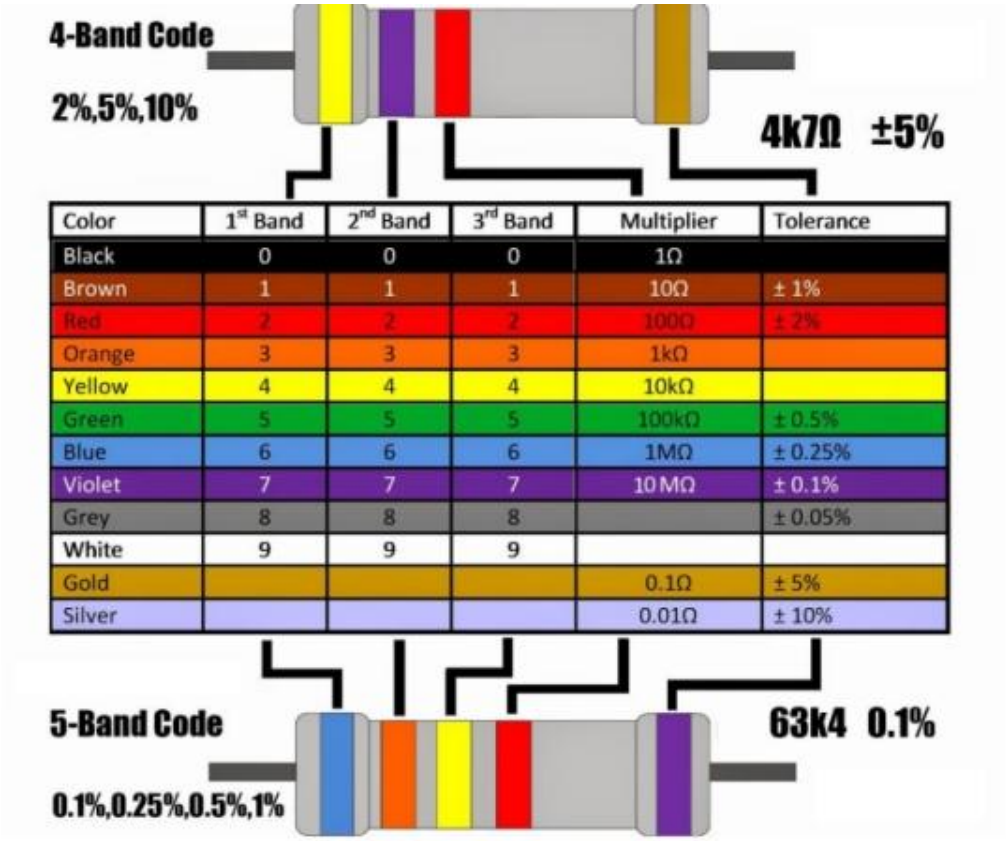
But first, more about resistors…

The unit of resistance is called the Ohm, which is usually shortened to Ω the Greek letter Omega. Because an Ohm is a low value of resistance (it doesn't resist much at all), we also denote the values of resistors in Kω (1,000 Ω) and MΩ (1,000,000 Ω).

These are called kilo-ohms and mega-ohms.

In this learning kit, we are going to use three different values of resistor: 220Ω, 1Kω and 10Kω. These resistors all look the same, except that they have different colored stripes on them. These stripes tell you the value of the resistor.

The resistor color code has three colored stripes and then a gold stripe at one end.



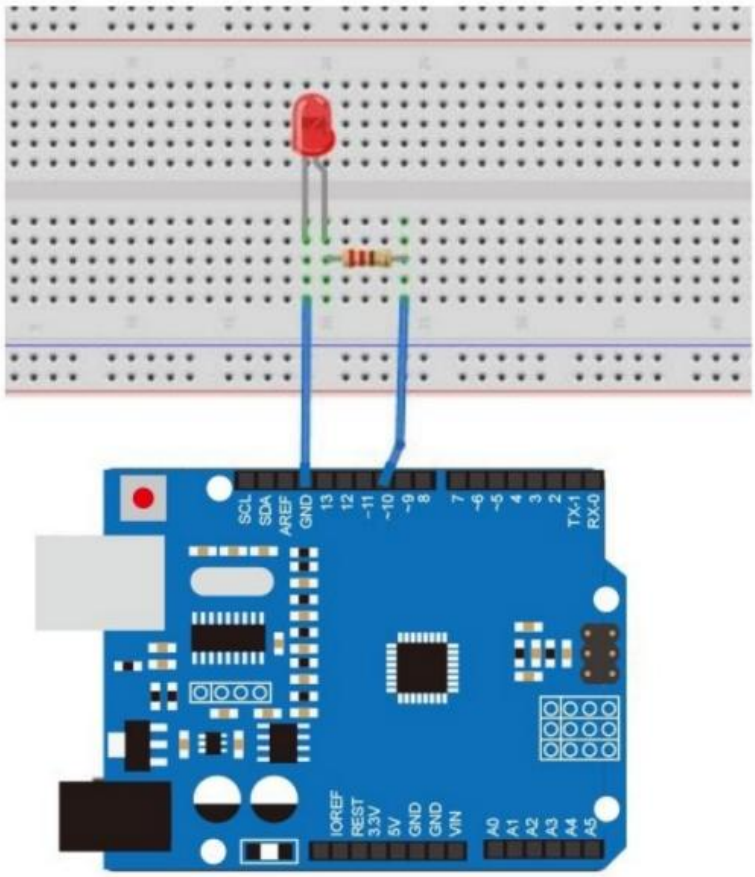
Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1%
Red	2	2	2	100Ω	± 2%
Orange	3	3	3	1kΩ	
Yellow	4	4	4	10kΩ	
Green	5	5	5	100kΩ	± 0.5%
Blue	6	6	6	1MΩ	± 0.25%
Violet	7	7	7	10 MΩ	± 0.1%
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1Ω	± 5%
Silver				0.01Ω	± 10%

Unlike LEDs, resistors do not have a positive and negative lead. They can be connected either way around.

If you find this approach method too complicated, you can read the color ring flag on our resistors directly to determine its resistance value. Or you may use a digital multimeter instead.

We follow below diagram from the experimental schematic link. Here we use digital pin 10. We connect LED to a 220-ohm resistor to avoid high current damaging the LED.

Circuit Connection



Sample Program

According to the above circuit, you can start compiling the program, turning the LED 1 second on and 1 second off. This program is simple and like one that comes with Arduino except it's connected to digital pin 10.

```
//////////////////////////////////////////////////////

int ledPin = 10; // define digital pin 10.

void setup()
```

```
{
pinMode(ledPin, OUTPUT);// define pin with LED connected as output.
}
void loop()
{
digitalWrite(ledPin, HIGH); // set the LED on.
Delay(1000); // wait for a second.
DigitalWrite(ledPin, LOW); // set the LED off.
Delay(1000); // wait for a second
}
//////////////////////////////////////////////////////
```

Result

After downloading this program, in the experiment, you will see the LED connected to pin 10 turning on and off, with an interval approximately one second.

Project 3 Traffic Light

Introduction

In the previous program, we have done the LED blinking experiment with one LED. Now, it's time to up the stakes and do a bit more complicated experiment-traffic light. These two experiments are similar.

While in this traffic lights experiment, we use 3 LEDs with different color other than 1 LED.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3.Red M5 LED*1

4.Yellow M5 LED*1

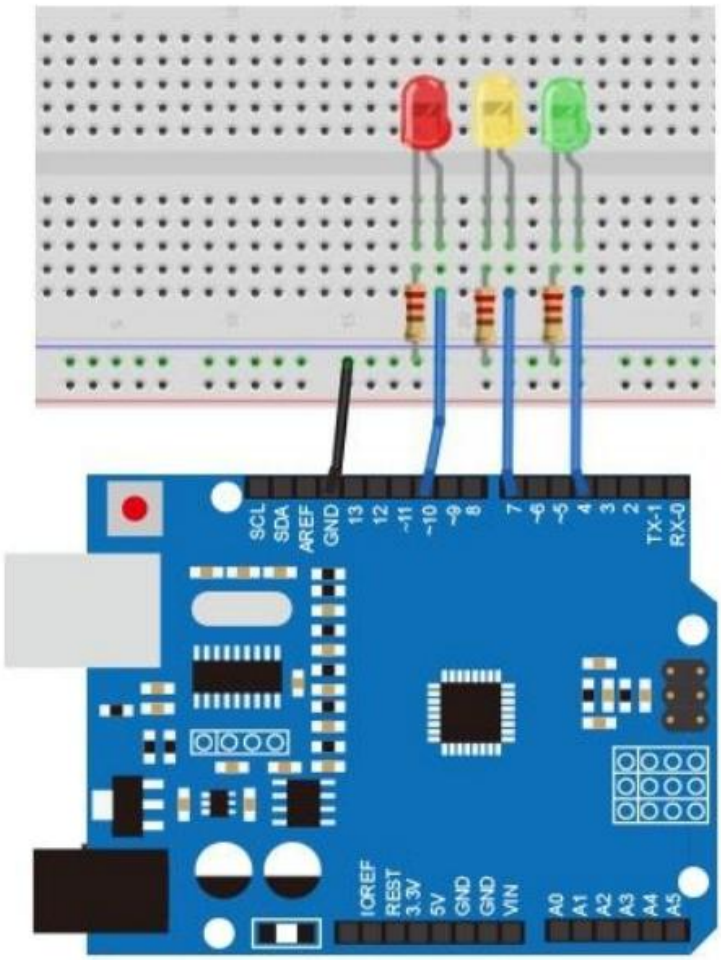
5.Green M5 LED*1

6.220Ω Resistor *3

7.Breadboard*1

8.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

Since it is a simulation of traffic lights, the blinking time of each LED should be the same with those in traffic lights system. In this program, we use Arduino delay () function to control delay time, which is much simpler than C language.

```
//////////////////////////////////////////////////////
int redled =10; // initialize digital pin 10.
int yellowled =7; // initialize digital pin 7.
int greenled =4; // initialize digital pin 4.
void setup()
{
   pinMode(redled, OUTPUT);// set the pin with red LED as “output”
   pinMode(yellowled, OUTPUT); // set the pin with yellow LED as
“output”
   pinMode(greenled, OUTPUT); // set the pin with blue LED as “output”
}
void loop()
{
   digitalWrite(greenled, HIGH);//// turn on green LED
   delay(2000);// wait 2 seconds
   digitalWrite(greenled, LOW); // turn off green LED
   for(int i=0;i<3;i++)// blinks for 3 times
   {
      delay(500);// wait 0.5 second
      digitalWrite(greenled, HIGH);// turn on green LED
```

```
    delay(500);// wait 0.5 second
    digitalWrite(greenled, LOW);// turn off green LED
  }
  digitalWrite(yellowled, HIGH);//// turn on yellow LED
  delay(3000);// wait 3 seconds
  digitalWrite(yellowled, LOW); // turn off yellow LED
  digitalWrite(redled, HIGH);// turn on red LED
  delay(3000);// wait 3 second
  digitalWrite(redled, LOW);// turn off red LED
}**********************
```

Result

When the uploading process is completed, we can see traffic lights of our own design.

Describe: The green light will be on for 2 seconds, and then off, followed by the green light blinking for 3 times, and then the yellow light on for 3 seconds, and then the red light on for 3 seconds, forming a cycle. Cycle then repeats.

LROBRUYA

Project 4 Digital Inputs

Introduction

I/O port means interface for INPUT and OUTPUT. Up until now, we have only used its OUTPUT function. In this experiment, we will try to use the input function, which is to read the output value of device connecting to it. We use 1 button and 1 LED using both input and output to give you a better understanding of the I/O function. Push Buttons, familiar to most of us, are a switch value (digital value) component. When it's pressed, the circuit is in closed (conducting) state.

Hardware Required

1. LONTEN Uno R3 Board *1

2.USB Cable *1

3.Push Button*1

4.Red M5 LED*1

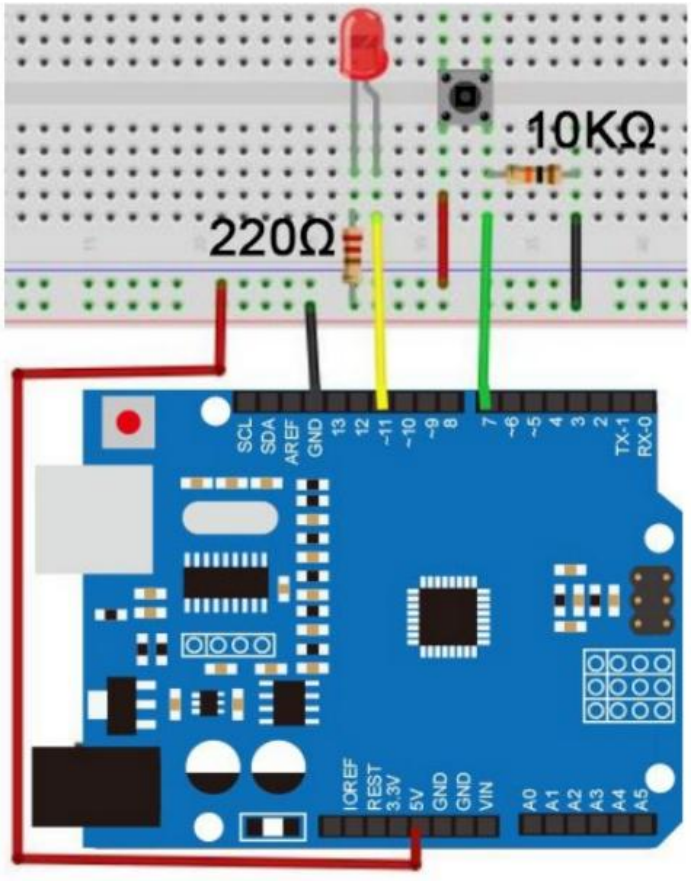
5.220Ω Resistor*1

6.10KΩ Resistor*1

7.Breadboard*1

8.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

Now, let's begin the compiling. When the button is pressed, the LED will be on. After the previous study, the coding should be easy for you. In this

program, we add a statement of judgment. Here, we use an if () statement. Arduino IDE is based on C language, so statements of C language such as while, switch etc. Can certainly be used for Arduino program. When we press the button, pin 7 will output high level. We can program pin 11 to output high level and turn on the LED. When pin 7 outputs low level, pin 11 also outputs low level and the LED remains off.

```
//////////////////////////////////////////////////////
int Redled=11;// initialize Redled 11
int inpin=7;// initialize pin 7
int val;// define val
void setup()
{
    pinMode(Redled,OUTPUT);// set LED pin as “output”
    pinMode(inpin,INPUT);// set button pin as “input”
}
void loop()
{
    val=digitalRead(inpin);// read the level value of pin 7 and assign if
to val
```

```
    if(val==LOW)// check if the button is pressed, if yes, turn on the
LED
    {
      digitalWrite(Redled,HIGH);
    }
    else
    {
      digitalWrite(Redled,LOW);
    }
}//////////////////////////////////////////////////////
```

Result

When the button is pressed, LED is on, otherwise, LED remains off.

After the above process, the button controlled LED experiment is completed. The simple principle of this experiment is widely used in a variety of circuit and electric appliances. You can easily come across it in your every day life. One typical example is when you press a certain key of your phone, the backlight will be on.

LROBRUYA

Project 5 Active Buzzer

Introduction

Active buzzer is widely used on computer, printer, alarm, electronic toy, telephone, timer etc.. It has an inner vibration source. Simply connect it with 5V power supply, it can buzz continuously.

Hardware Required

1. LONTEN Uno R3 Board *1

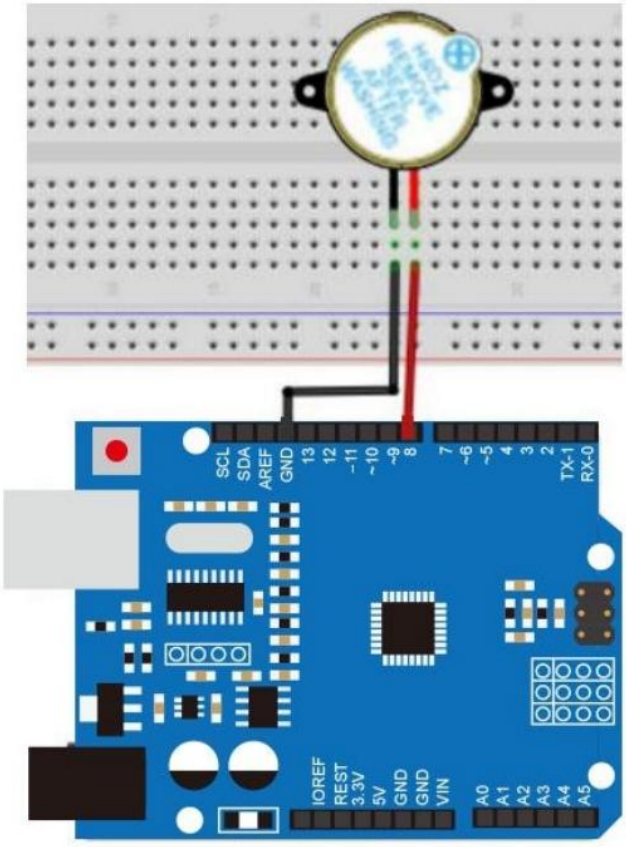
2.USB Cable *1

3.Active Buzzer*1

4.Breadboard*1

5.Breadboard Jumper Wires* Several

Circuit Connection



When connecting the circuit, pay attention to the positive & the negative poles of the buzzer. In the photo, you can see there are red and black lines.

When the circuit is finished, you can begin programming.

Sample Program

Program is simple. You control the buzzer by outputting high/low level.

```
//////////////////////////////////////////////////////

int buzzer=8;// initialize digital IO pin that controls the buzzer
```

```
void setup()
{
pinMode(buzzer,OUTPUT);// set pin mode as “output”
}
void loop()
{
digitalWrite(buzzer, HIGH); // produce sound
delay(1000);
digitalWrite(buzzer, LOW);
delay(1000);
}//////////////////////////////////////////////////////
```

Result

After downloading the program, the buzzer experiment is completed. You can see the buzzer is ringing.

Project 6 Passive Buzzer

Introduction

We can use Arduino to make many interactive works of which the most commonly used is acoustic-optic display. All the previous experiment has something to do with LED. However, the circuit in this experiment can produce sound. Normally, the experiment is done with a buzzer or a speaker while buzzer is simpler and easier to use. The buzzer we introduced here is a passive buzzer. It cannot be actuated by itself, but by external pulse frequencies. Different frequencies produce different sounds. We can use Arduino to code the melody of a song, which is actually quite fun and simple.

Hardware Required

1. LONTEN Uno R3 Board *1

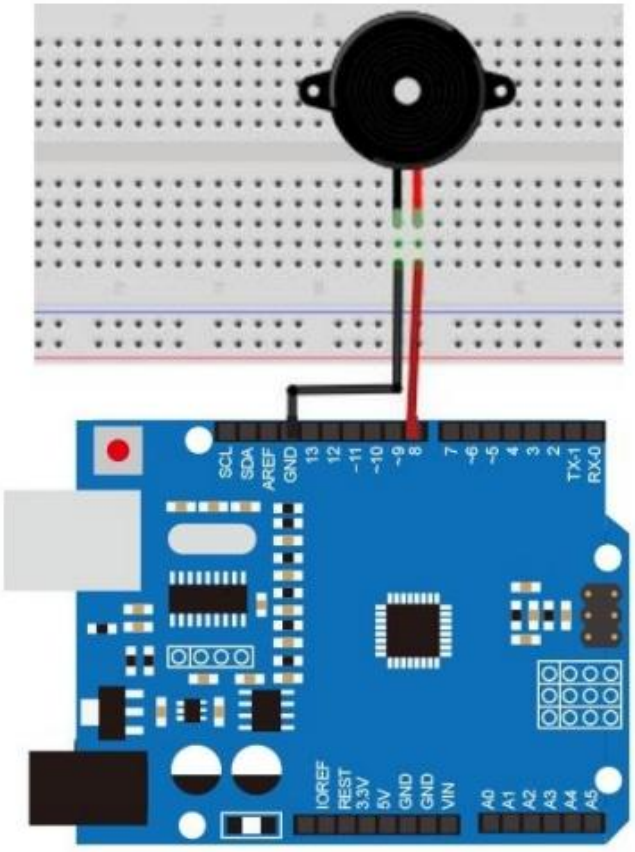
2.USB Cable *1

3.Passive Buzzer*1

4.Breadboard*1

5.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

```
//////////////////////////////////////////////////////

int buzzer=8;// select digital IO pin for the buzzer

void setup() {

pinMode(buzzer,OUTPUT);// set digital IO pin pattern, OUTPUT to be
```

```
output }
void loop() {
unsigned char i,j;//define variable
while(1) {
for(i=0;i<80;i++)// output a frequency sound
{ digitalWrite(buzzer,HIGH);// sound
delay(1);//delay 1ms
digitalWrite(buzzer,LOW);//not sound
delay(1);//delay 1ms }
for(i=0;i<100;i++)// output a frequency sound
{ digitalWrite(buzzer,HIGH);// sound
delay(1);//delay 1ms
digitalWrite(buzzer,LOW);//not sound
delay(2);//delay 2ms }
}
}
//////////////////////////////////////////////////////
```

Result

After downloading the program, The buzzer will emit a rapid alarm sound.

Project 7 RGB LED

Introduction

Tricolor principle to display various colors PWM controlling ports to display full color Can be driven directly by Arduino PWM interfaces.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3.RGB LED *1

4.220Ω Resistor*3

5.Breadboard*1

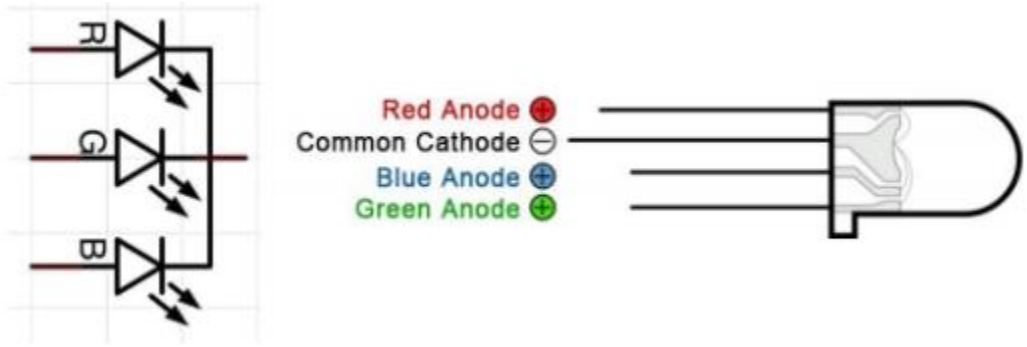
6.Breadboard Jumper Wires* Several

RGB LED:

At first glance, RGB (Red, Green and Blue) LEDs look just like regular LEDs. However, inside the usual LED package, there are actually three LEDs, one red, one green and one blue. By controlling the brightness of

each of the individual LEDs you can mix pretty much any color you want. We mix colors the same way you would mix paint on a palette - by adjusting the brightness of each of the three LEDs. Fortunately for us, UNO R3 board has an analogWrite function that you can use with pins marked with a ~ to output a variable amount of power to the appropriate LEDs.

The RGB LED has four leads. There is one lead going to the positive connection of each of the single LEDs within the package and a single lead that is connected to all three negative sides of the LEDs.



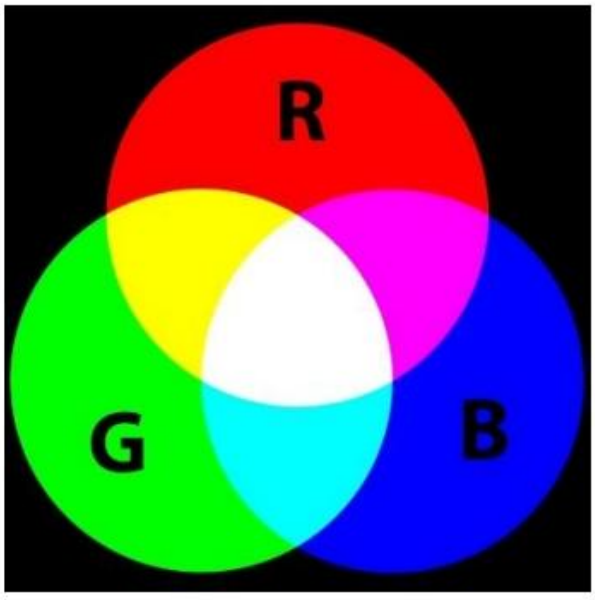
Here on the photographs you can see 4 electrode LED. Every separate pin for Green or Blue or Red color is called Anode. You will always connect "+" to it. Cathode goes to "-"(ground). If you connect it other way round the LED will not light.

The common negative connection of the LED package is the second pin from the flat side. It is also the longest of the four leads and will be

connected to the ground. Each LED inside the package requires its own 220Ω resistor to prevent too much current flowing through it. The three positive leads of the LEDs (one red, one green and one blue) are connected to UNO output pins using these resistors.

Three Primary Colors:

The reason that you can mix any color you like by varying the quantities of red, green and blue light is that your eye has three types of light receptor in it (red, green and blue). Your eye and brain process the amounts of red, green and blue and convert it into a color of the spectrum. In a way, by using the three LEDs, we are playing a trick on the eye. This same idea is used in TVs, where the LCD has red, green and blue color dots next to each other making up each pixel.



If we set the brightness of all three LEDs to be the same, then the overall color of the light will be white. If we turn off the blue LED, so that just the red and green LEDs are the same brightness, then the light will appear yellow.

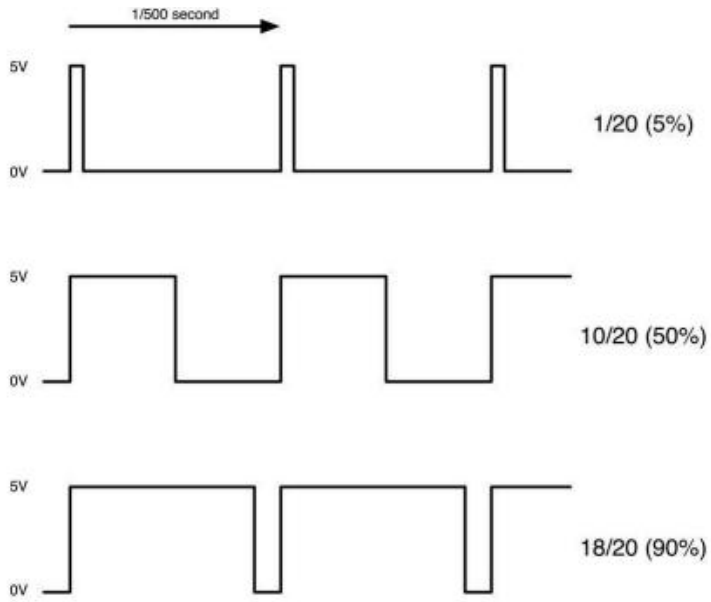
We can control the brightness of each of the red, green and blue parts of the LED separately, making it possible to mix any color we like.

Black is not so much a color as an absence of light. Therefore, the closest we can come to black with our LED is to turn off all three colors.

Theory (PWM)

Pulse Width Modulation (PWM) is a technique for controlling power. We also use it here to control the brightness of each of the LEDs.

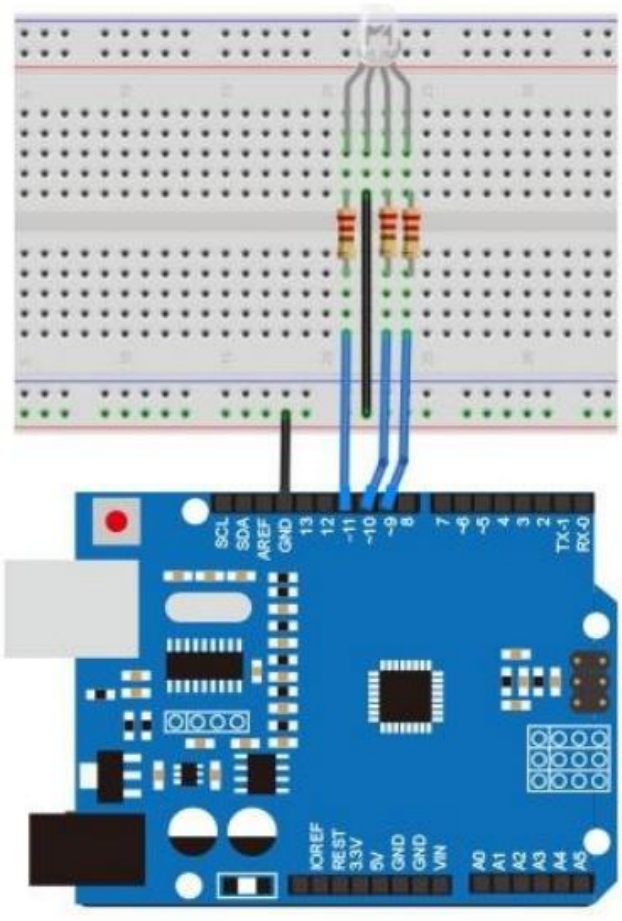
The diagram below shows the signal from one of the PWM pins on the UNO.



Roughly every 1/500 of a second, the PWM output will produce a pulse. The length of this pulse is controlled by the “analogWrite” function. So “analogWrite(0)” will not produce any pulse at all and “analogWrite(255)”will produce a pulse that lasts all the way until the next pulse is due, so that the output is actually on all the time.

If we specify a value in the “analogWrite” that is somewhere in between 0 and 255, then we will produce a pulse. If the output pulse is only high for 5% of the time, then whatever we are driving will only receive 5% of full power. If, however, the output is at 5V for 90% of the time, then the load will get 90% of the power delivered to it. We cannot see the LEDs turning on and off at that speed, so to us, it just looks like the brightness is changing.

Circuit Connection



Sample Program

```
//////////////////////////////////////////////////////////

int redpin = 11; //select the pin for the red LED

int bluepin =10; // select the pin for the blue LED

int greenpin =9;// select the pin for the green LED

int val;

void setup() {

pinMode(redpin, OUTPUT);
```

```
pinMode(bluepin, OUTPUT);
pinMode(greenpin, OUTPUT);
Serial.begin(9600);
}
void loop()
{
for(val=255; val>0; val--)
{
analogWrite(11, val);
analogWrite(10, 255-val);
analogWrite(9, 128-val);
delay(1);
}
for(val=0; val<255; val++)
{
analogWrite(11, val);
analogWrite(10, 255-val);
analogWrite(9, 128-val);
delay(1);
}
```

```
Serial.println(val, DEC);
}
//////////////////////////////////////////////////////////
```

Result

Directly copy the above code into Arduino IDE, and click upload , wait a few seconds, you can see a full-color LED. The three primary colors of different brightness can be mixed to create new colors.

Project 8 Analog Value Reading

Introduction

In this experiment, we will begin the learning of analog I/O interfaces. On an Arduino, there are 6 analog interfaces numbered from A0 to A5. These 6 interfaces can also be used as digital ones numbered as D14-D19. After a brief introduction, let's begin our project. Potentiometer used here is a typical output component of analog value that is familiar to us.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3.Potentiometer *1

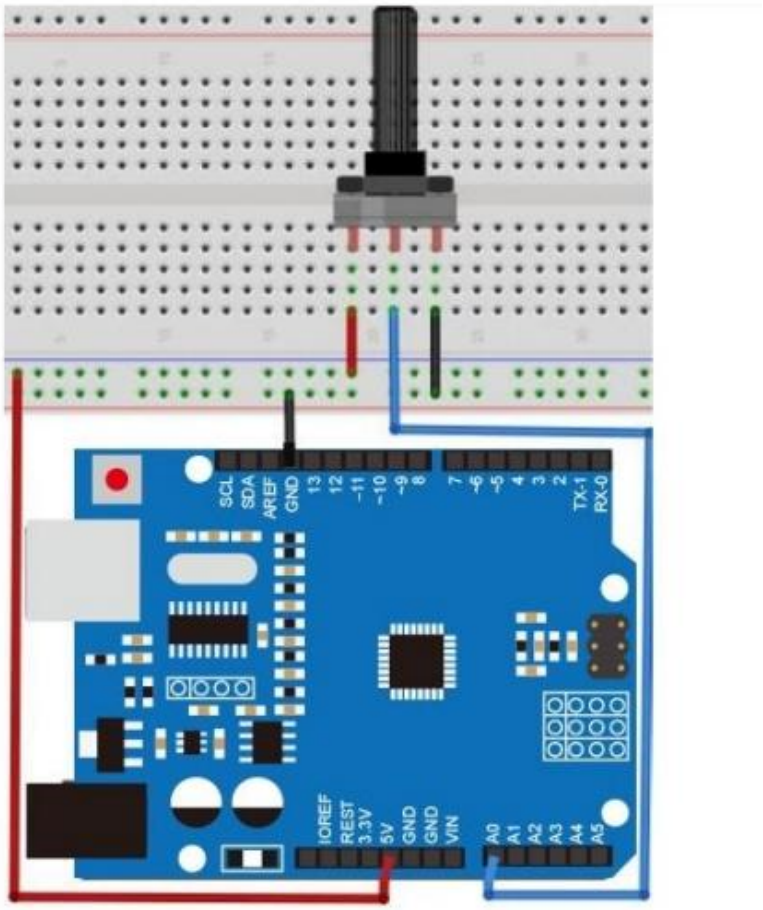
4.Breadboard*1

5.Breadboard Jumper Wires* Several

Circuit Connection

In this experiment, we will convert the resistance value of the potentiometer to analog ones and display it on the screen. This is an application we need to master well for our future experiments.

Connection circuit as below:



The analog interface we use here is interface A0.

Sample Program

The program compiling is simple. An analogRead () Statement can read the value of the interface. The A/D acquisition of Arduino 328 is in 10 bits, so the value it reads is among 0 to 1023. One difficulty in this project is to display the value on the screen, which is actually easy to learn. First, we need to set the baud rate in void setup (). Displaying the

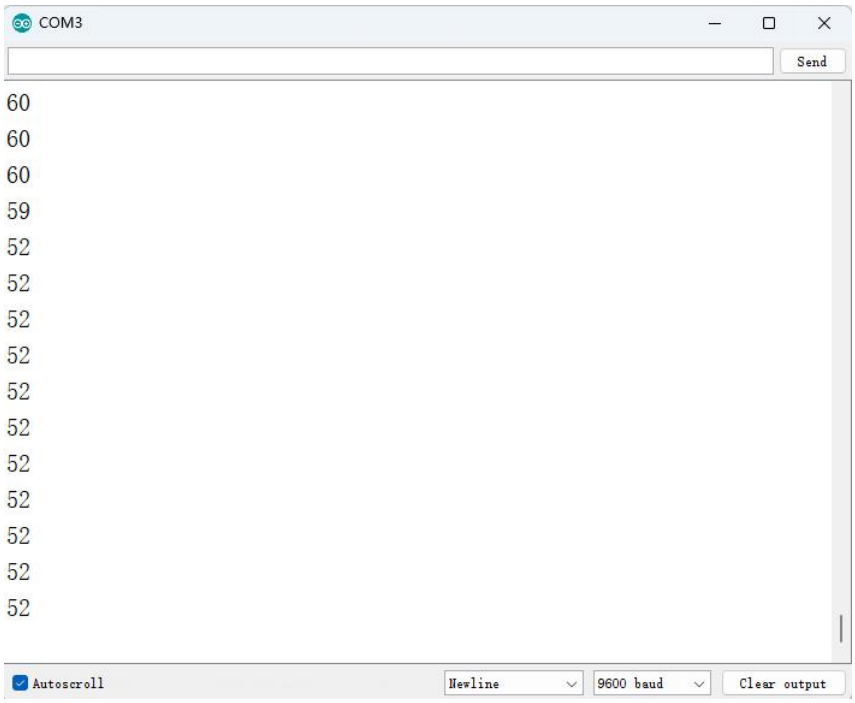
value is a communication between Arduino and PC, so the baud rate of the Arduino should match the one in the PC's software setup. Otherwise, the display will be messy codes or no display at all. In the lower right corner of the Arduino software monitor window, there is a button for baud rate set up. The set up here needs to match the one in the program. The statement in the program is Serial.begin(); enclosed is the baud rate value, followed by statement for displaying. You can either use Serial.print() or Serial.println() statement.

```
//////////////////////////////////////////////////////
int potpin=0;// initialize analog pin A0
int ledpin=13;// initialize digital pin 13
int val=0;// define val, assign initial value 0
void setup()
{
pinMode(ledpin,OUTPUT);// set digital pin as “output”
Serial.begin(9600);// set baud rate at 9600
}
void loop()
{
digitalWrite(ledpin,HIGH);// turn on the LED on pin 13
```

```
delay(50);// wait for 0.05 second
digitalWrite(ledpin,LOW);// turn off the LED on pin 13
delay(50);// wait for 0.05 second
val=analogRead(potpin);// read the analog value of analog pin 0, and
assign it to val
Serial.println(val);// display val's value
}
////////////////////////////////////////////////////////
```

Result

The Sample Program uses the built-in LED connected to pin 13. Each time the device reads a value, the LED blinks. Below is the analog value it reads.



When you rotate the potentiometer knob, you can see the displayed value changes. The reading of analog value is a very common function since most sensors output analog value. After calculation, we can have the corresponding value we need.

Project 9 Photo Resistor

Introduction

After completing all the previous experiments, we acquired some basic understanding and knowledge about Arduino application. We have learned digital input and output, analog input and PWM. Now, we can begin the learning of sensors applications. Photo resistor (Photovaristor) is a resistor whose resistance varies according to different incident light strength. It's made based on the photoelectric effect of semiconductor. If the incident light is intense, its resistance reduces; if the incident light is weak, the resistance increases.

Photovaristor is commonly applied in the measurement of light, light control and photovoltaic conversion (convert the change of light into the change of electricity). Photo resistor is also being widely applied to various light control circuit, such as light control and adjustment, optical switches etc. We will start with a relatively simple experiment regarding photovaristor application. Photovaristor is an element that changes its resistance as light strenth changes. So we will need to read the analog values. We can refer to the PWM experiment, replacing the potentiometer with photovaristor. When there is change in light strength, there will be corresponding change on the LED.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3.Photo Resistor*1

4.Red M5 LED*1

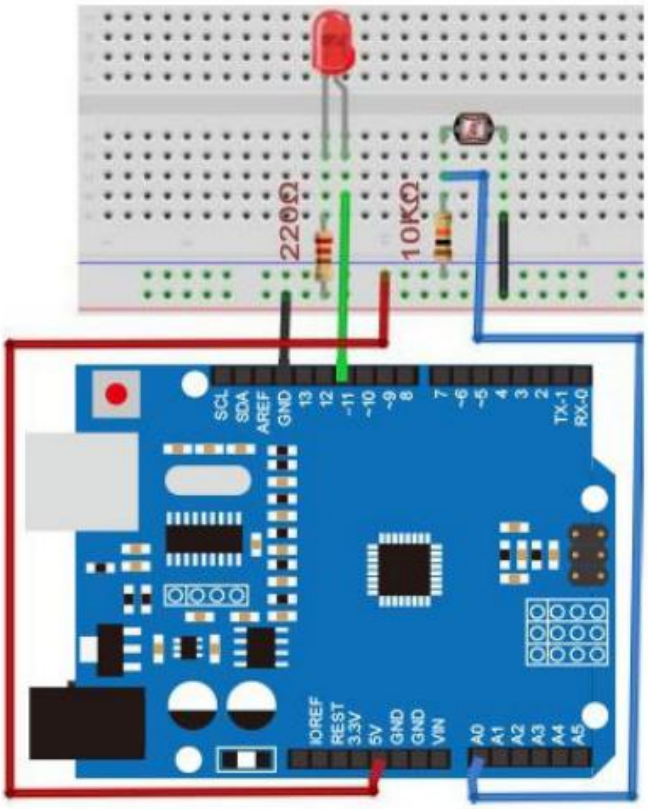
5.10KΩ Resistor*1

6.220Ω Resistor*1

7.Breadboard*1

8.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

After the connection, let's begin the program compiling. The program is similar to the one of PWM. For change detail, please refer to the Sample Program below.

```
//////////////////////////////////////////////////////////

int potpin=0;// initialize analog pin A0, connected with photovaristor

int ledpin=11;// initialize digital pin 11, output regulating the brightness of LED

int val=0;// initialize variable val
```

```
void setup()
{
pinMode(ledpin,OUTPUT);// set digital pin 11 as “output”
Serial.begin(9600);// set baud rate at “9600”
}
void loop()
{
val=map ( analogRead(potpin) , 0 , 1023 , 0 , 255);
// read the analog value of the sensor and assign it to val
Serial.println(val);// display the value of val
analogWrite(ledpin,val);
// turn on the LED and set up brightness（maximum output value 255）
delay(10);// wait for 0.01
}
//////////////////////////////////////////////////////////
```

Result

After downloading the program, you can change the light strength around the photovaristor and see corresponding brightness change of the LED. Photovaristors has various applications in our everyday life. You can make other interesting interactive projects base on this one.

Project 10 Water Level Detection Sensor Module

Introduction

In this lesson, you will learn how to use a water level detection sensor module. This module can perceive the depth of water and the core component is an amplifying circuit which is made up of a transistor and several pectinate PCB routings. When put into the water, these routings will present a resistor that can change along with the change of the water's depth. Then, the signal of water's depth is converted into the electrical signal, and we can know the change of water's depth through the ADC function of UNO R3.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3.Water Level Detection Sensor Module*1

4.Breadboard*1

5.Breadboard Jumper Wires* Several

Water sensor:

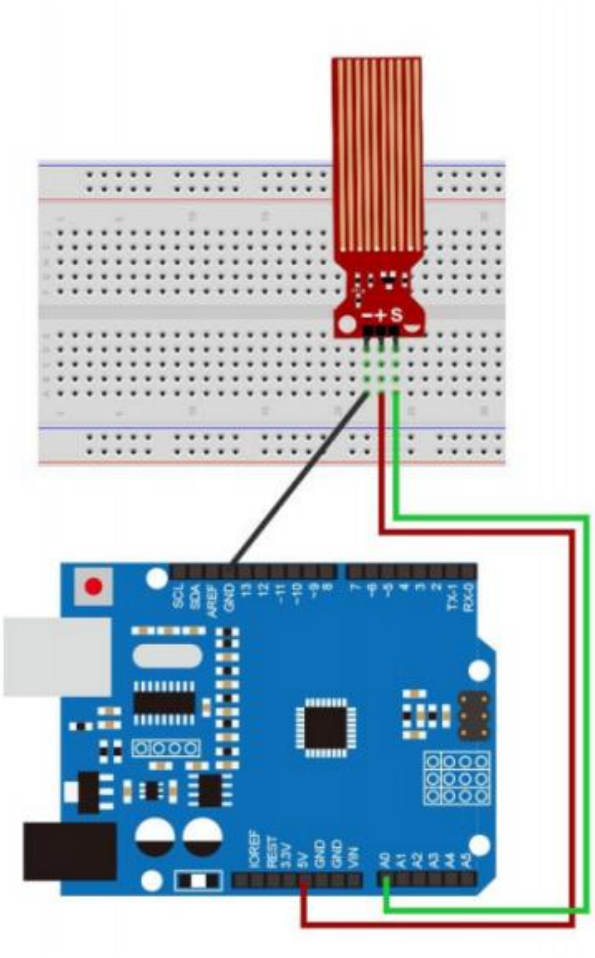
A water sensor brick is designed for water detection, which can be widely used in sensing the rainfall, water level, even the liquate leakage. The brick is mainly composed of three parts: an electronic brick connector, a

1 MΩ resistor, and several lines of bare conducting wires. You can use it with the analog pins to detect the amount of water induced contact between the grounded and sensor traces. This item can judge the water level through with a series of exposed parallel wires stitch to measure the water droplet/water size. It can easily change the water size to analog signal, and output analog value can directly be used in the program function, then to achieve the function of water level alarm. It has low power consumption, and high sensitivity.

Features:

1. Working voltage: 5V
2. Working Current: <20ma
3. Interface: Analog
4. Width of detection: 40mm×16mm
5. Working Temperature: 10℃~30℃
6. Output voltage signal: 0~4.2V

Circuit Connection



Wiring tips: Power supply (+) is connected to 5V of UNO R3 board, ground electrode (-) is connected to GND. Signal output (S) is connected to the ports (A0-A5) which have function of inputting analog signal in UNO R3 board, random one is OK, but it should define the same demo code as the routine.

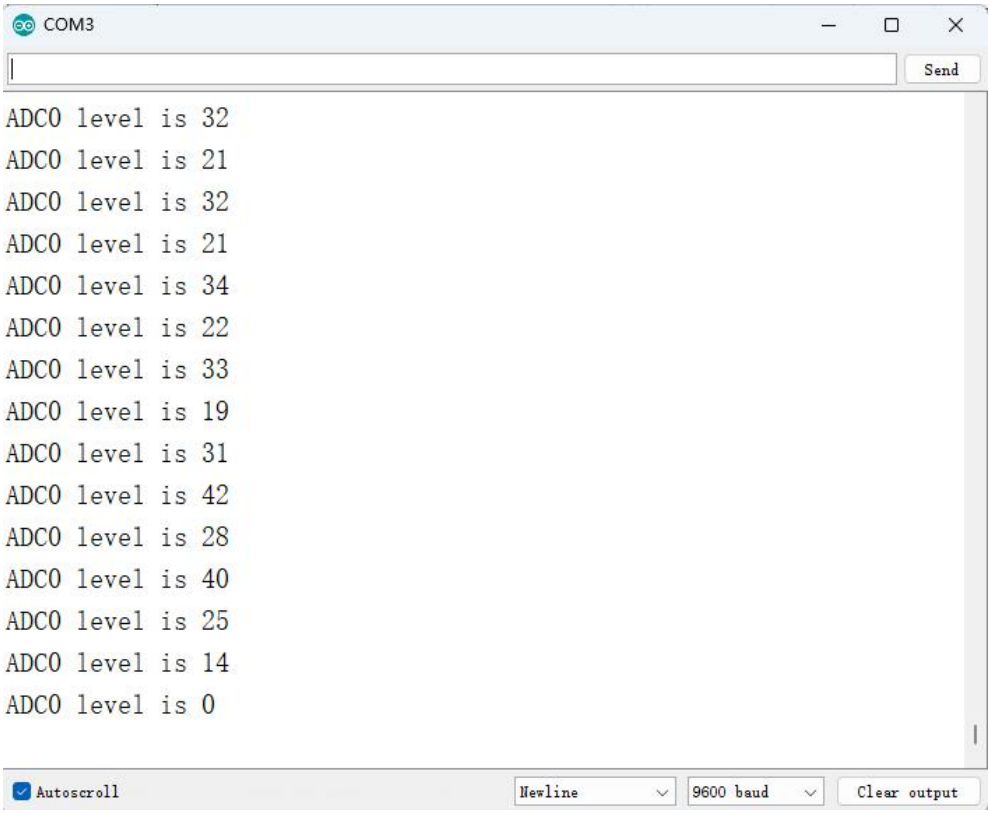
Sample Program

The voltage value received by the analog pin represents the water level value. Each time the change is greater than 10, the value can be updated in the serial port.

```
//////////////////////
int adc_id = 0;
int HistoryValue = 0;
char printBuffer[128];
void setup() {
Serial.begin(9600);
}
void loop() {
int value = analogRead(adc_id); // get adc value
if(((HistoryValue>=value) && ((HistoryValue - value) > 10)) ||
((HistoryValue<value) && ((value - HistoryValue) > 10))) {
sprintf(printBuffer,"ADC%d level is %d\n",adc_id, value);
Serial.print(printBuffer);
HistoryValue = value;
} }
//////////////////////
```

Result

After the program is uploaded. Open the monitor then slowly insert the water level sensor into the water. you can see the data as below:



Project 11 Sound Sensor Module

Introduction

In this lesson, you will learn how to use a sound sensor module. This module has two outputs:

AO: analog output, real-time output voltage signal of microphone

DO: when the intensity of the sound reaches a certain threshold, the output is a high or low level signal. The threshold sensitivity can be achieved by adjusting the potentiometer.

To make sure the microphone can detect your voice normally, please try to change its sensitivity by turning the blue precise potentiometer on the module.

Hardware Required

1.LONTEN Uno R3 Board *1

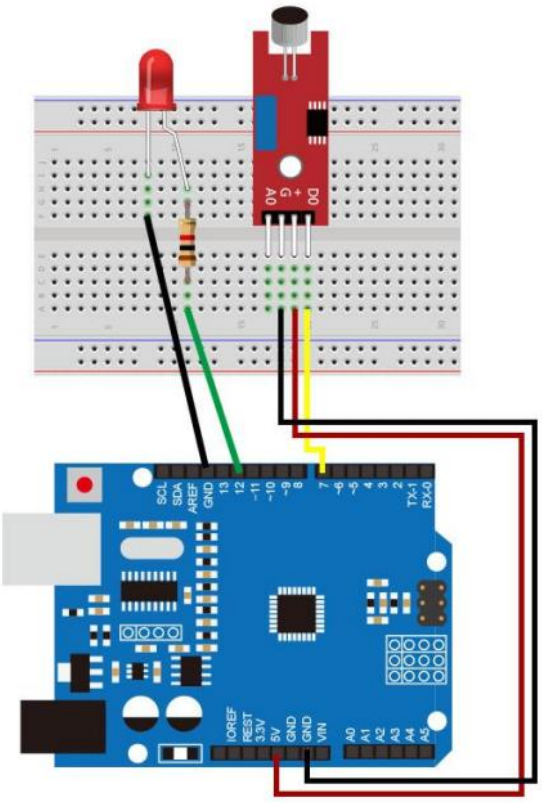
2.USB Cable *1

3.Sound Sensor Module*1

4.Breadboard*1

5.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

When the sound is greater than the sound threshold set by the blue potentiometer, the D0 interface will output a high level "1", and the LED will light up for 3 seconds and then turn off. This is the sound-controlled light in the corridor in our lives.

```
//////////////////////////////

int sensorPin = 7;

int ledPin = 12;

// select the pin for the LED
```

```
void setup()
{
pinMode(ledPin,OUTPUT);
pinMode(sensorPin,INPUT);
}
void loop(){
if(1==digitalRead(sensorPin))
{digitalWrite(ledPin,HIGH);
delay(3000);
digitalWrite(ledPin,LOW);
}else
{digitalWrite(ledPin,LOW);
}
}
//////////////////////////////
```

Result

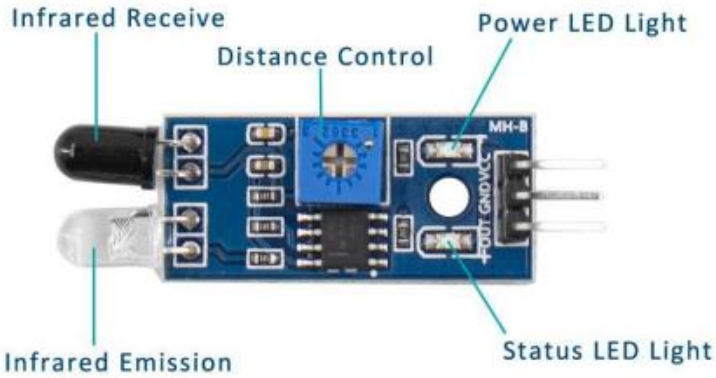
You can blow a breath at the microphone of the sound module or clap your hands with both hands next to the microphone, the LED light will light up and go out after 3 seconds.

LROBRUYA

Project 12 Infrared Obstacle Avoidance Module

Introduction

The infrared obstacle detector sensor has a pair of infrared transmitting and receiving tubes. The transmitter emits an infrared rays of a certain frequency. When the detection direction encounters an obstacle(reflecting surface), the infrared rays are reflected back, and receiving tube will receive it. At this time, the indicator (green LED) lights up. After processed by the circuit, the signal output terminal will output Digital signal. You can rotate the potentiometer on the shield to adjust the detection distance. It is better to adjust the potentiometer to make the green LED in a state between on and off. The detection distance is the best, almost 10cm.



Read the signal level of obstacle detector sensor to judge whether detect obstacles or not.

When detects an obstacle, sensor's signal pin outputs LOW (display 0); otherwise, output HIGH (display 1).

Show the result on the serial monitor, and control the external LED module turn ON/OFF.

Hardware Required

1. LONTEN Uno R3 Board *1

2.USB Cable *1

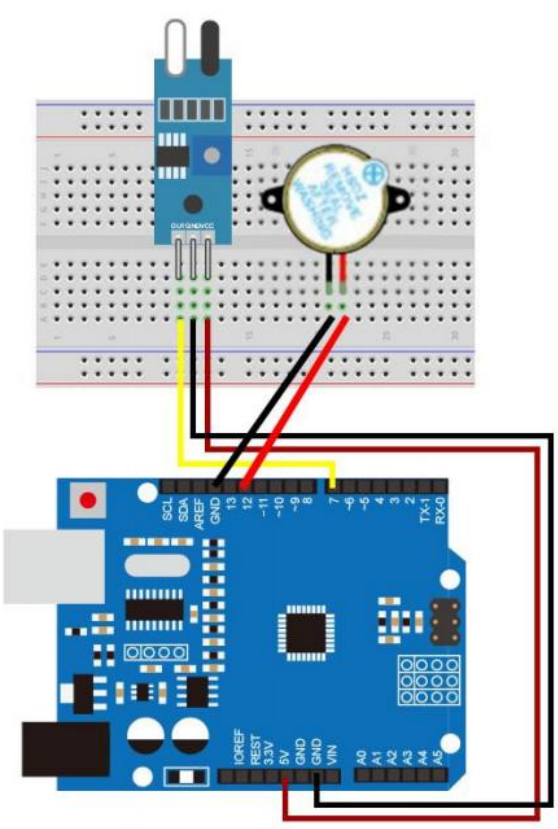
3.Infrared Obstacle Avoidance Module*1

4.Active Buzzer*1

5.Breadboard*1

6. Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

When an obstacle is detected, the OUT pin of the module outputs a low level "0", and the buzzer will sound.

```
//////////////////////////////

int   sensorPin   =   7;

int   buzzPin   =   12;

void setup(){

   pinMode(buzzPin,OUTPUT);

   pinMode(sensorPin,INPUT);
```

```
}
void loop(){
    if(0==digitalRead(sensorPin)){
        digitalWrite(buzzPin,HIGH);
    }
    else{
    digitalWrite(buzzPin,LOW);
  }
}//////////////////////////
```

Result

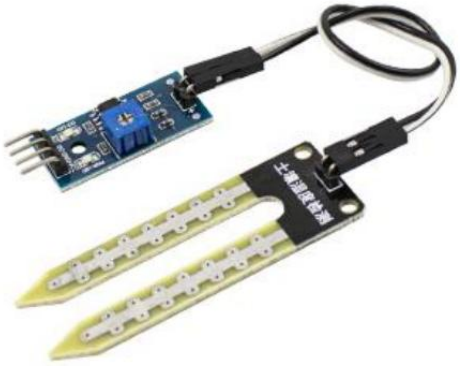
You can put your hand directly in front of the infrared pair tube of the sensor. The buzzer will sound. After removing the handle, the buzzer stops beeping.

Project 13 Soil Moisture Module

Introduction

Used to detect the degree of soil drought. The AO interface of the module outputs the analog voltage value of soil moisture (0-1023). 1023 means the driest. 0 means the highest moisture content.

When the soil moisture of the module exceeds the set threshold, the DO port will output a low level "0". When the soil moisture and humidity are lower than the set threshold, the module D0 will output a high level "1";



Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

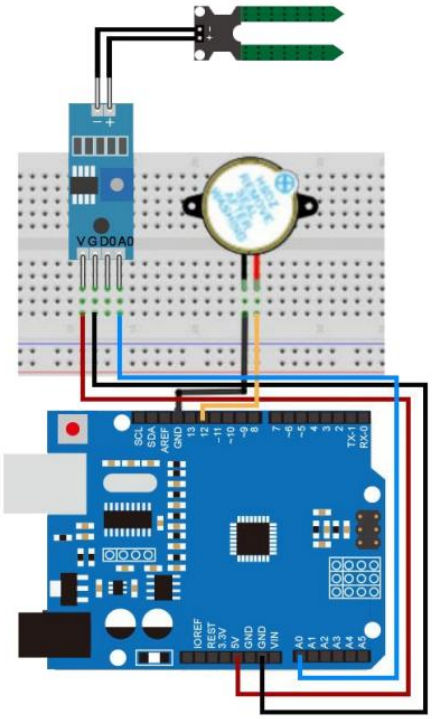
3.Soil Moisture Module*1

4.Active Buzzer*1

5.Breadboard*1

6. Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

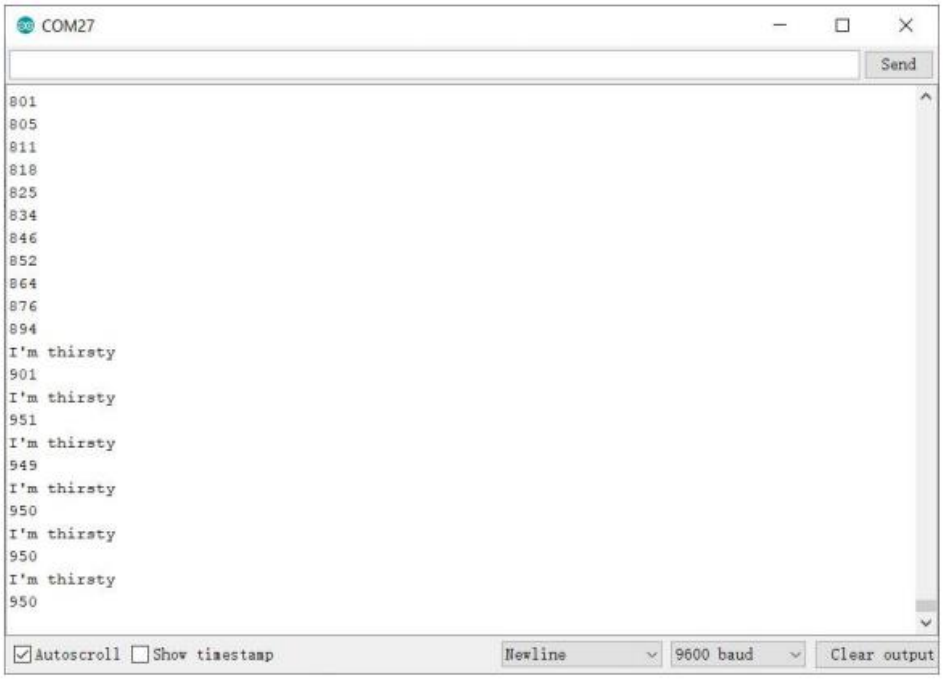
The AO pin receives the analog voltage value (0-1023) of the soil sensor, 0 represents the highest moisture content of the soil, and 1023 represents the lowest moisture content. The analog voltage value soil_value >900 detected by the soil sensor is set in the program, and the buzzer will sound an alarm.

```
//////////////////////

int adc_id = 0;

int buzzer_pin=12;

int soil_value=0;
```

```
void setup()
{Serial.begin(9600);
pinMode(buzzer_pin,OUTPUT);
}
void loop()
{
soil_value = analogRead(adc_id); // get adc value
if(soil_value>900)
{digitalWrite(buzzer_pin,HIGH);
delay(500);
digitalWrite(buzzer_pin,LOW);
delay(500);
Serial.println("I'm thirsty");
}
else{digitalWrite(buzzer_pin,LOW);}
Serial.println(soil_value);
}
//////////////////
```

Result

After the program is uploaded. Insert the soil moisture sensor into the soil, open the serial monitor, and observe the data sent back from the soil sensor to the serial monitor. When the soil changes from moist to drought, and soil_value > 900, the buzzer will alarm.



Project 14 DHT11 Temperature and Humidity Sensor

Introduction

This DHT11 Temperature and Humidity Sensor features calibrated digital signal output with the temperature and humidity sensor complex. Its technology ensures high reliability and excellent long-term stability. Ahigh-performance 8-bit microcontroller is connected. This sensor

includes a resistive element and a sense of wet NTC temperature measuring devices. It has excellent quality, fast response, anti-interference ability and high cost performance advantages. Each DHT11 sensor features extremely accurate calibration data of humidity calibration chamber. The calibration coefficients stored in the OTP program memory, internal sensors detect signals in the process, and we should call these calibration coefficients. The single-wire serial interface system is integrated to make it quick and easy. Qualities of small size, low power, and 20-meter signal transmission distance make it a wide applied application and even the most demanding one. Convenient connection, special packages can be provided according to users need.

Hardware Required

1.Uno R3 Board *1

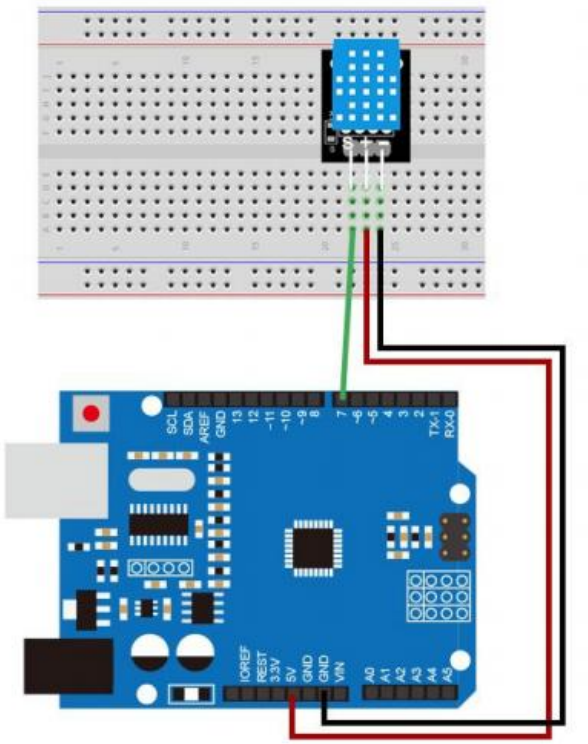
2.USB Cable *1

3.Temperature and Humidity Unit *1

4.Breadboard*1

5.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

You need to add the library file <DHT11> before compiling and downloading the program, otherwise an error will occur in the compiled file. For details on adding library files, refer to the course How to Add Arduino Libraries.

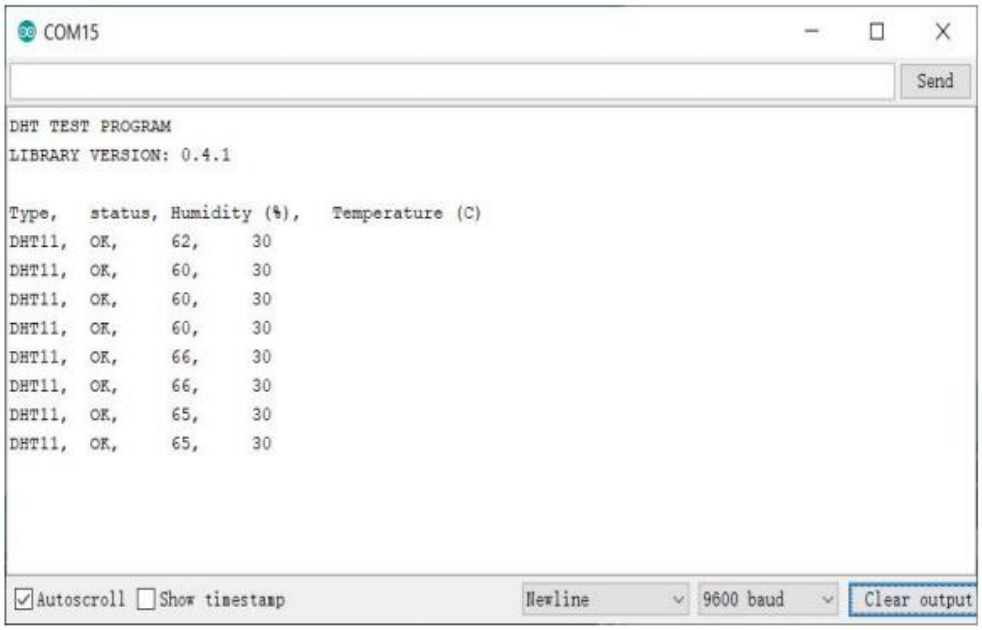
```
//////////////////////

#include <dht11.h>

dht11 DHT;

#define DHT11_PIN 7

void setup()
```

```
{
Serial.begin(9600);
Serial.println("DHT TEST PROGRAM ");
Serial.print("LIBRARY VERSION: ");
Serial.println(DHT11LIB_VERSION);
Serial.println();
Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");
}
void loop()
{
int chk;
Serial.print("DHT11, \t");
chk = DHT.read(DHT11_PIN);
// READ DATA
switch (chk){
case DHTLIB_OK:
Serial.print("OK,\t");
break;
case DHTLIB_ERROR_CHECKSUM:
Serial.print("Checksum error,\t");
```

```
break;
case DHTLIB_ERROR_TIMEOUT:
Serial.print("Time out error,\t");
break;
default:
Serial.print("Unknown error,\t");
break;
}
// DISPLAT DATA
Serial.print(DHT.humidity,1);
Serial.print(",\t");
Serial.println(DHT.temperature,1);
delay(1000);
}
//////////////////
```

Result

After uploading the code, the serial monitor will print the temperature and humidity values of the current environment.



Project 15 Tilt Switch

Introduction

Tilt switch controls the ON and OFF of an LED. Tilt sensors (tilt ball switch) allow you to detect orientation or inclination. They are small, inexpensive, low-power and easy-to-use. If used properly, they will not wear out. Their simplicity makes them popular for toys, gadgets and appliances.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

3.Red M5 LED*1

4.Ball Switch*1

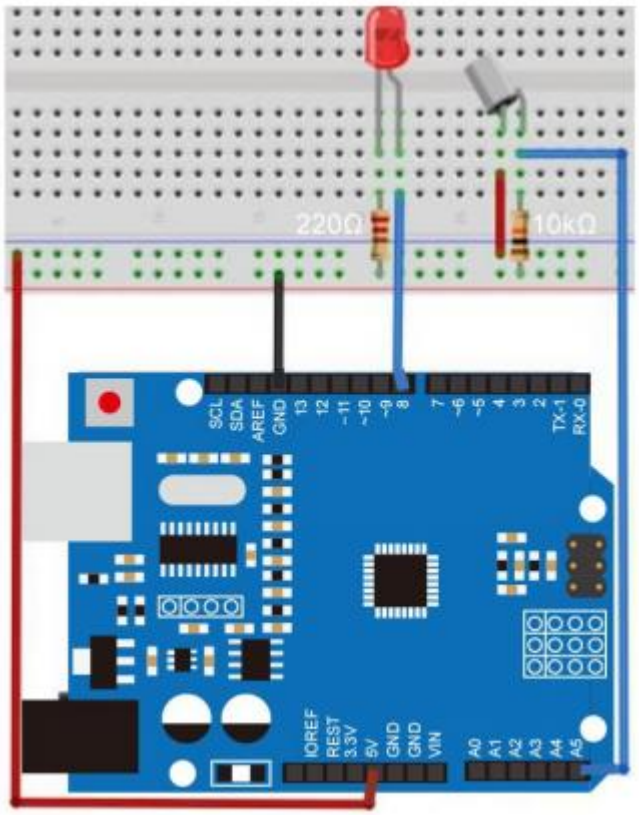
5.220Ω Resistor *1

6.10kΩ Resistor*1

7.Breadboard*1

8.Breadboard Jumper Wires* Several

Circuit Connection



Connect the controller board, shield, breadboard and USB cable according to Arduino tutorial.

Connect the LED to digital pin 8, ball switch to analog pin 5.

Experiment Principle

When one end of the switch is below horizontal position, the switch is on. The voltage of the analog port is about 5V (1023 in binary). The LED will be on. When the other end of the switch is below horizontal position, the switch is off. The voltage of the analog port is about 0V (0 in binary). The LED will be off. In the program, we determine whether the switch is on or off according to the voltage value of the analog port, whether it's above 2.5V (512 in binary) or not.

Sample Program

```
//////////////////////////////////////////////////////////
#define led 8
#define vaule 5
void setup()
{
   pinMode(led,OUTPUT);// set digital pin 8 as “output”
}
void loop()
```

```
{
    int i;// define variable i
    i=analogRead(vaule);// read the voltage value of analog pin 5
    if(i<512)// if larger that 512（2.5V）
    {
      digitalWrite(8,HIGH);// turn on LED
    }
    else
    {
      digitalWrite(8,LOW);// turn off LED
    }
}
//////////////////////////////////////////////////////////
```

Result

Hold the breadboard with your hand. Tilt it to a certain extent, the LED will be on. If there is no tilt, the LED will be off.

The principle of this experiment can be applied to relay control.

LROBRUYA

Project 16 Thermometer

Introduction

A thermistor is a thermal resistor - a resistor that changes its resistance with temperature. Technically, all resistors are thermistors - their resistance changes slightly with temperature - but the change is usually very small and difficult to measure. Thermistors are made so that the resistance changes drastically with temperature so that it can be 100 ohms or more of change per degree!

There are two kinds of thermistors, NTC (negative temperature coefficient) and PTC (positive temperature coefficient). In general, you will see NTC sensors used for temperature measurement. PTC's are often used as resettable fuses - an increase in temperature increases the resistance which means that as more current passes thru them, they heat up and 'choke back' the current, quite handy for protecting circuits!

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

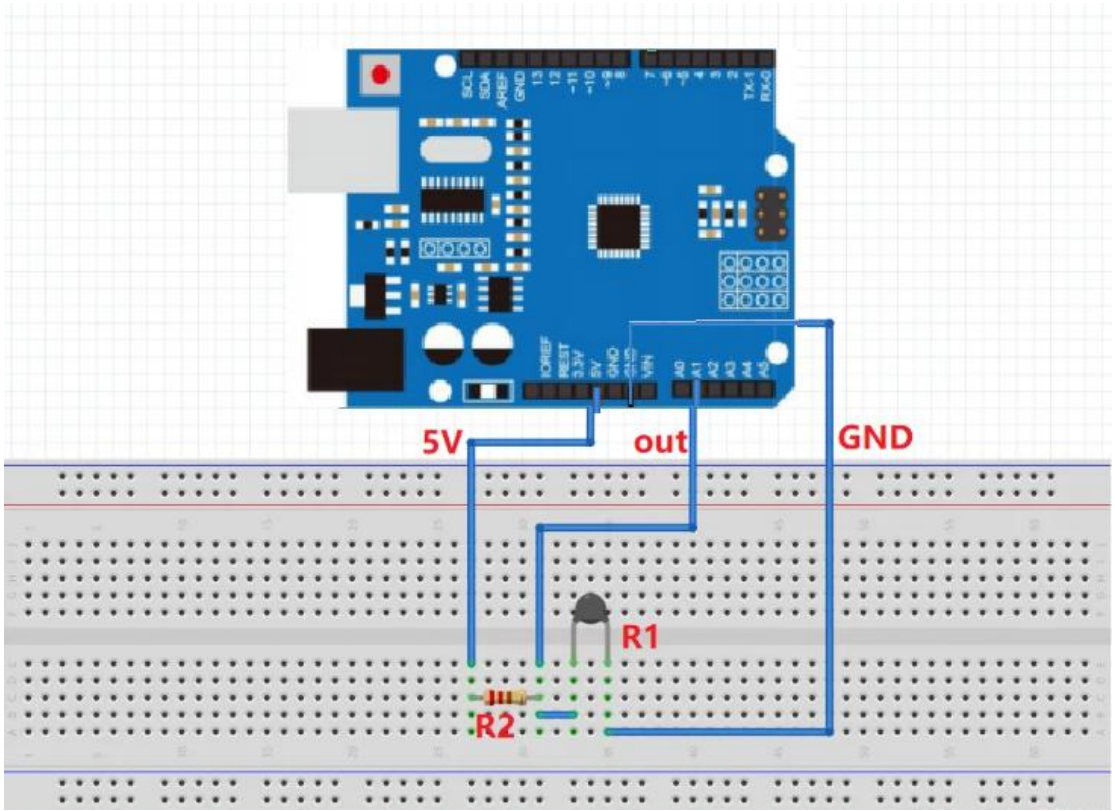
3.Thermistor *1

4.10k ohm resistor *1

5.Breadboard*1

6.Breadboard Jumper Wires* Several

Circuit Connection



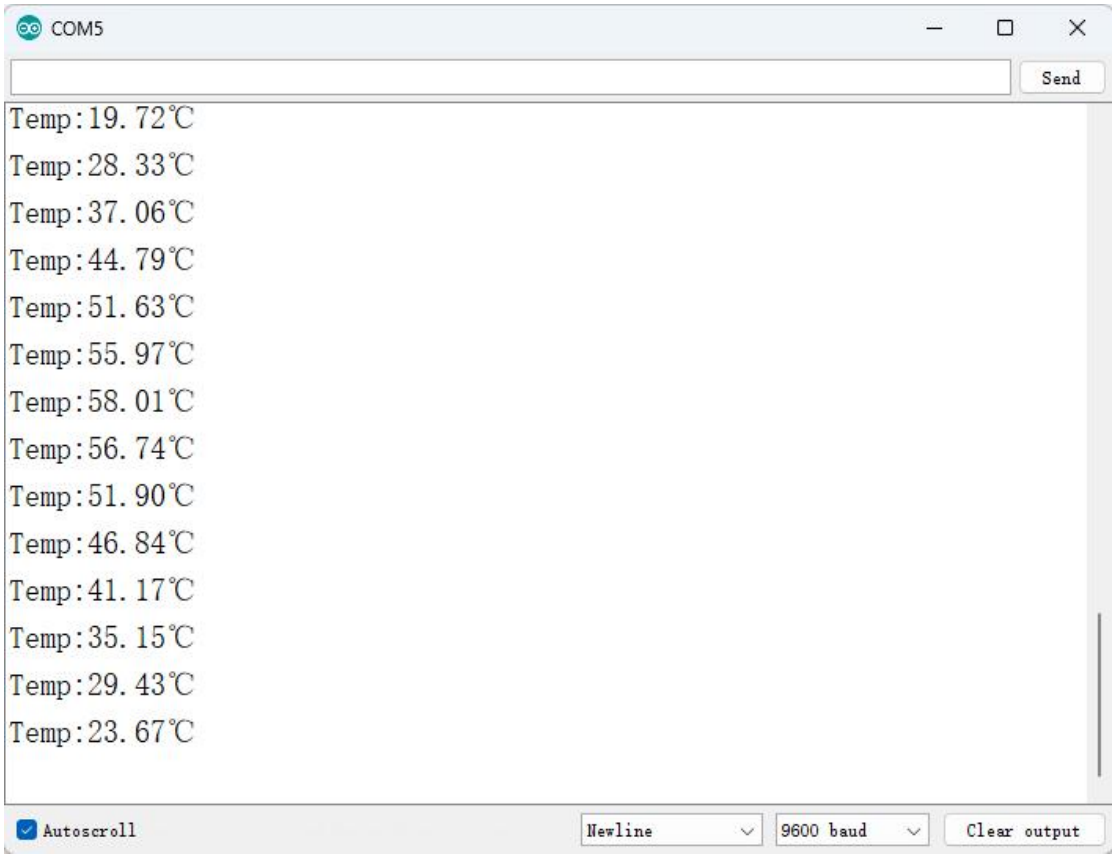
Sample Program

```
//////////////////////////////////////////////////////////

int tempPin = A1;

void setup()

{

   Serial.begin(9600);

}

void loop()

{

   int tempReading = analogRead(tempPin);

   // This is OK

   double tempK = log(10000.0 * ((1024.0 / tempReading - 1)));

   tempK = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 *

tempK * tempK )) * tempK );           //    Temp Kelvin

   float tempC = tempK - 273.15;      // Convert Kelvin to Celcius

   float tempF = (tempC * 9.0)/ 5.0 + 32.0; // Convert Celcius to

Fahrenheit

   Serial.print("Temp:");

   Serial.print(tempC);

   Serial.println("℃");
```

```
    delay(500);
}
//////////////////////////////////////////////////////
```

Result

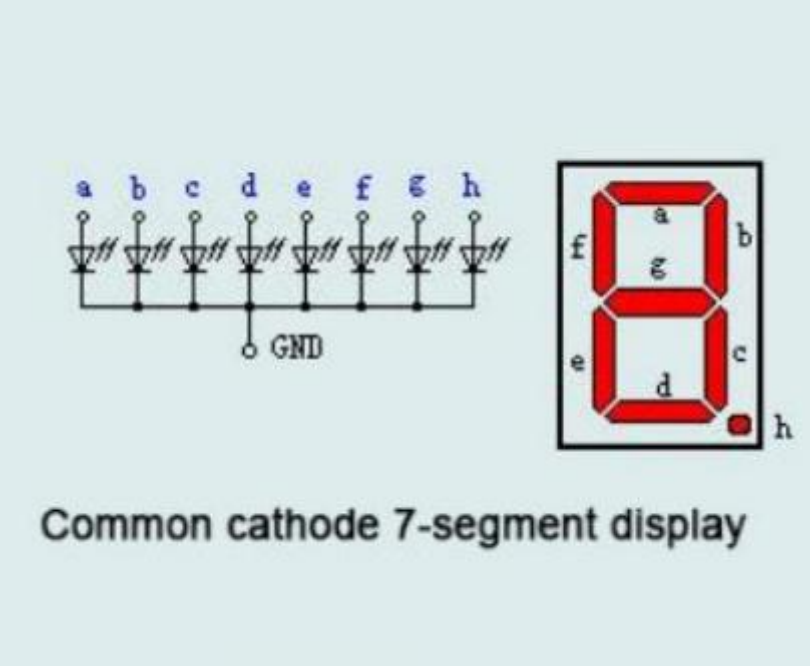
Upload the program, open the monitor then you can see the data as below:



Project 17 Digit LED Segment Display

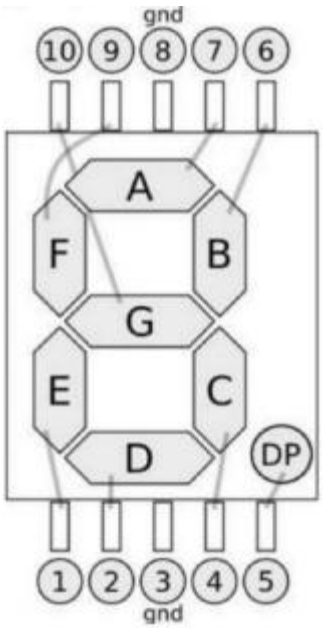
Introduction

LED segment displays are common for displaying numerical information. It's widely applied on displays of electromagnetic oven, full automatic washing machine, water temperature display, electronic clock etc. It is necessary that we learn how it works. LED segment display is a semiconductor light-emitting device. Its basic unit is a light-emitting diode (LED). LED segment display can be divided into 7-segment display and 8-segment display according to the number of segments. 8-segment display has one more LED unit (for decimal point display) than 7-segment one. In this experiment, we use a 8-segment display.



Common cathode 7-segment display

According to the wiring method of LED units, LED segment displays can be divided into display with common anode and display with common cathode. Common anode display refers to the one that combine all the anodes of LED units into one common anode (COM).For the common anode display, connect the common anode (COM) to +5V. When the cathode level of a certain segment is low, the segment is on; when the cathode level of a certain segment is high, the segment is off. For the common cathode display, connect the common cathode (COM) to GND. When the anode level of a certain segment is high, the segment is on; when the anode level of a certain segment is low, the segment is off.



Each segment of the display consists of an LED. So when you use it, you also need use a current-limiting resistor. Otherwise, LED will be burnt

out. In this experiment, we use a common cathode display. As we mentioned above, for common cathode display, connect the common cathode (COM) to GND. When the anode level of a certain segment is high, the segment is on; when the anode level of a certain segment is low, the segment is off.

Hardware Required

1.LONTEN Uno R3 Board *1

2.USB Cable *1

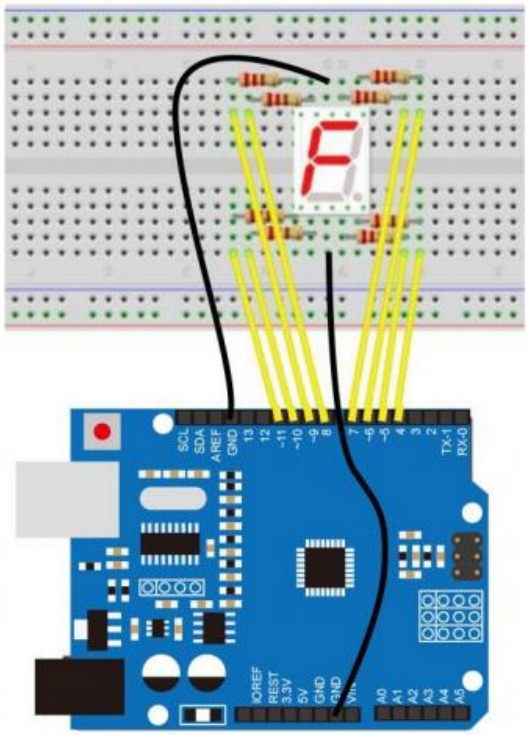
3.1-digit Segment Display*1

4.220Ω Resistor *8

5.Breadboard*1

6.Breadboard Jumper Wires* Several

Circuit Connection



Sample Program

There are seven segments for numerical display, one for decimal point display. Corresponding segments will be turned on when displaying certain numbers. For example, when displaying number 1, b and c segments will be turned on. We compile a subprogram for each number, and compile the main program to display one number every 2 seconds, cycling display number 0 ~ 9.The displaying time for each number is subject to the delay time, the longer the delay time, the longer the displaying time.

```
//////////////////////////////////////////////////////
// set the IO pin for each segment
int a=7;// set digital pin 7 for segment a
int b=6;// set digital pin 6 for segment b
int c=5;// set digital pin 5 for segment c
int d=10;// set digital pin 10 for segment d
int e=11;// set digital pin 11 for segment e
int f=8;// set digital pin 8 for segment f
int g=9;// set digital pin 9 for segment g
int dp=4;// set digital pin 4 for segment dp
void digital_0(void) // display number 0
{
   unsigned char j;
   digitalWrite(a,HIGH);
   digitalWrite(b,HIGH);
   digitalWrite(c,HIGH);
   digitalWrite(d,HIGH);
   digitalWrite(e,HIGH);
   digitalWrite(f,HIGH);
   digitalWrite(g,LOW);
```

```
    digitalWrite(dp,LOW);
}
void digital_1(void) // display number 1
{
    unsigned char j;
    digitalWrite(b,HIGH);// turn on segment b
    digitalWrite(c,HIGH);// turn on segment c

    for(j=7;j<=11;j++)// turn off other segments
    digitalWrite(j,LOW);
    digitalWrite(dp,LOW);// turn off segment dp
}
void digital_2(void) // display number 2
{
    unsigned char j;
    digitalWrite(b,HIGH);
    digitalWrite(a,HIGH);
    for(j=9;j<=11;j++)
    digitalWrite(j,HIGH);
    digitalWrite(dp,LOW);
```

```
    digitalWrite(c,LOW);
    digitalWrite(f,LOW);
}
void digital_3(void) // display number 3
{
    digitalWrite(g,HIGH);
    digitalWrite(a,HIGH);
    digitalWrite(b,HIGH);
    digitalWrite(c,HIGH);
    digitalWrite(d,HIGH);
    digitalWrite(dp,LOW);
    digitalWrite(f,LOW);
    digitalWrite(e,LOW);
}
void digital_4(void) // display number 4
{
    digitalWrite(c,HIGH);
    digitalWrite(b,HIGH);
    digitalWrite(f,HIGH);
    digitalWrite(g,HIGH);
```

```
    digitalWrite(dp,LOW);
    digitalWrite(a,LOW);
    digitalWrite(e,LOW);
    digitalWrite(d,LOW);
}
void digital_5(void) // display number 5
{
    digitalWrite(a,HIGH);
    digitalWrite(b, LOW);
    digitalWrite(c,HIGH);
    digitalWrite(d,HIGH);
    digitalWrite(e, LOW);
    digitalWrite(f,HIGH);
    digitalWrite(g,HIGH);
    digitalWrite(dp,LOW);
}
void digital_6(void) // display number 6
{
    unsigned char j;
    for(j=7;j<=11;j++)
```

```
    digitalWrite(j,HIGH);
    digitalWrite(c,HIGH);
    digitalWrite(dp,LOW);
    digitalWrite(b,LOW);
}
void digital_7(void) // display number 7
{
    unsigned char j;
    for(j=5;j<=7;j++)
    digitalWrite(j,HIGH);
    digitalWrite(dp,LOW);
    for(j=8;j<=11;j++)
    digitalWrite(j,LOW);
}
void digital_8(void) // display number 8
{
    unsigned char j;
    for(j=5;j<=11;j++)
    digitalWrite(j,HIGH);
    digitalWrite(dp,LOW);
```

```
}
void digital_9(void) // display number 9
{
    digitalWrite(a,HIGH);
    digitalWrite(b,HIGH);
    digitalWrite(c,HIGH);
    digitalWrite(d,HIGH);
    digitalWrite(e, LOW);
    digitalWrite(f,HIGH);
    digitalWrite(g,HIGH);
    digitalWrite(dp,LOW);
}
void setup()
{
    int i;// set variable
    for(i=4;i<=11;i++)
    pinMode(i,OUTPUT);// set pin 4-11as “output”
}
void loop()
{
```

```
digital_0();// display number 0
delay(2000);// wait for 2s
digital_1();// display number 1
delay(2000);// wait for 2s
digital_2();// display number 2
delay(2000); // wait for 2s
digital_3();// display number 3
delay(2000); // wait for 2s
digital_4();// display number 4
delay(2000); // wait for 2s
digital_5();// display number 5
delay(2000); // wait for 2s
digital_6();// display number 6
delay(2000); // wait for 2s
digital_7();// display number 7
delay(2000); // wait for 2s
digital_8();// display number 8
delay(2000); // wait for 2s
digital_9();// display number 9
delay(2000); // wait for 2s
```

```
}

//////////////////////////////////////////////////////
```

Result

LED segment display repeat displays number 0 to 9.