

Welcome

Thank you for choosing Freenove products!

How to Start

When reading this, you should have downloaded the ZIP file for this product.

Unzip it and you will get a folder containing tutorials and related files. Please start with this PDF tutorial.

! Unzip the ZIP file instead of opening the file in the ZIP file directly.

! Do not move, delete or rename files in the folder just unzipped.

Get Support and Offer Input

Freenove provides free and responsive product and technical support, including but not limited to:

- Product quality issues
- Product use and build issues
- Questions regarding the technology employed in our products for learning and education
- Your input and opinions are always welcome
- We also encourage your ideas and suggestions for new products and product improvements

For any of the above, you may send us an email to:

support@freenove.com

Safety and Precautions

Please follow the following safety precautions when using or storing this product:

- Keep this product out of the reach of children under 6 years old.
- This product should be **used only when there is adult supervision present** as young children lack necessary judgment regarding safety and the consequences of product misuse.
- This product contains small parts and parts, which are sharp. This product contains electrically conductive parts. **Use caution with electrically conductive parts near or around power supplies, batteries and powered (live) circuits.**
- When the product is turned ON, activated or tested, some parts will move or rotate. **To avoid injuries to hands and fingers keep them away from any moving parts!**
- It is possible that an improperly connected or shorted circuit may cause overheating. **Should this happen, immediately disconnect the power supply or remove the batteries and do not touch anything until it cools down!** When everything is safe and cool, review the product tutorial to identify the cause.
- Only operate the product in accordance with the instructions and guidelines of this tutorial, otherwise parts may be damaged or you could be injured.
- Store the product in a cool dry place and avoid exposing the product to direct sunlight.

Any concerns?  support@freenove.com

- After use, always turn the power OFF and remove or unplug the batteries before storing.

About Freenove

Freenove provides open source electronic products and services worldwide.

Freenove is committed to assist customers in their education of robotics, programming and electronic circuits so that they may transform their creative ideas into prototypes and new and innovative products. To this end, our services include but are not limited to:

- Educational and Entertaining Project Kits for Robots, Smart Cars and Drones
- Educational Kits to Learn Robotic Software Systems for Arduino, Raspberry Pi and micro: bit
- Electronic Component Assortments, Electronic Modules and Specialized Tools
- **Product Development and Customization Services**

You can find more about Freenove and get our latest news and updates through our website:

<http://www.freenove.com>

Copyright

All the files, materials and instructional guides provided are released under [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](https://creativecommons.org/licenses/by-nc-sa/3.0/). A copy of this license can be found in the folder containing the Tutorial and software files associated with this product.



This means you can use these resource in your own derived works, in part or completely but **NOT for the intent or purpose of commercial use.**

Freenove brand and logo are copyright of Freenove Creative Technology Co., Ltd. and cannot be used without written permission.



Any concerns? [✉ support@freenove.com](mailto:support@freenove.com)

Contents

Welcome.....	1
Contents.....	1
Preface.....	2
ESP32-WROVER	3
Extension board of the ESP32-WROVER	6
CH340	7
Programming Software	17
Environment Configuration	20
Notes for GPIO.....	24
Chapter 1 LCD1602	26
Project 1.1 LCD1602.....	26
Chapter 2 LCD2004	36
Project 2.1 LCD2004.....	36
What's next?	44
What's next?(others)	44
End of the Tutorial.....	44

Preface

ESP32 is a micro control unit with integrated Wi-Fi launched by Espressif, which features strong properties and integrates rich peripherals. It can be designed and studied as an ordinary Single Chip Microcontroller (SCM) chip, or connected to the Internet and used as an Internet of Things device.

ESP32 can be developed using the Arduino platform, which will definitely make it easier for people who have learned Arduino to master. Moreover, the code of ESP32 is completely open-source, so beginners can quickly learn how to develop and design IOT smart household products including smart curtains, fans, lamps and clocks.

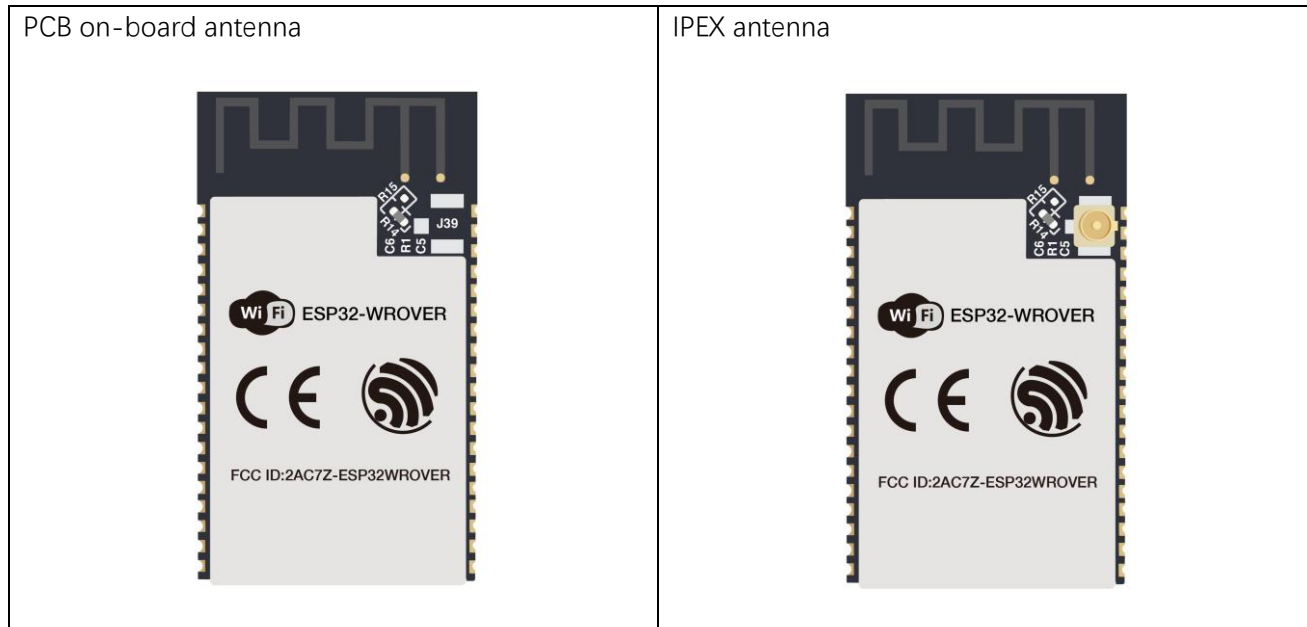
Generally, ESP32 projects consist of code and circuits. Don't worry even if you've never learned code and circuits, because we will gradually introduce the basic knowledge of C programming language and electronic circuits, from easy to difficult. Our products contain all the electronic components and modules needed to complete these projects. It's especially suitable for beginners.

We divide each project into four parts, namely Component List, Component Knowledge, Circuit and Code. Component List helps you to prepare material for the experiment more quickly. Component Knowledge allows you to quickly understand new electronic modules or components, while Circuit helps you understand the operating principle of the circuit. And Code allows you to easily master the use of ESP32 and accessory kit. After finishing all the projects in this tutorial, you can also use these components and modules to make products such as smart household, smart cars and robots to transform your creative ideas into prototypes and new and innovative products.

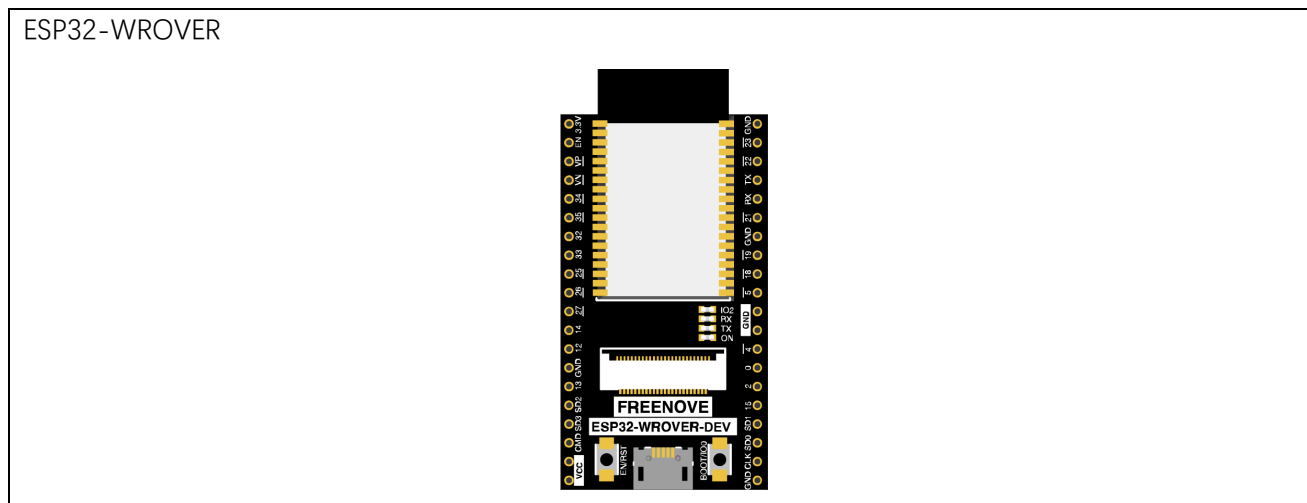
In addition, if you have any difficulties or questions with this tutorial or toolkit, feel free to ask for our quick and free technical support through support@freenove.com

ESP32-WROVER

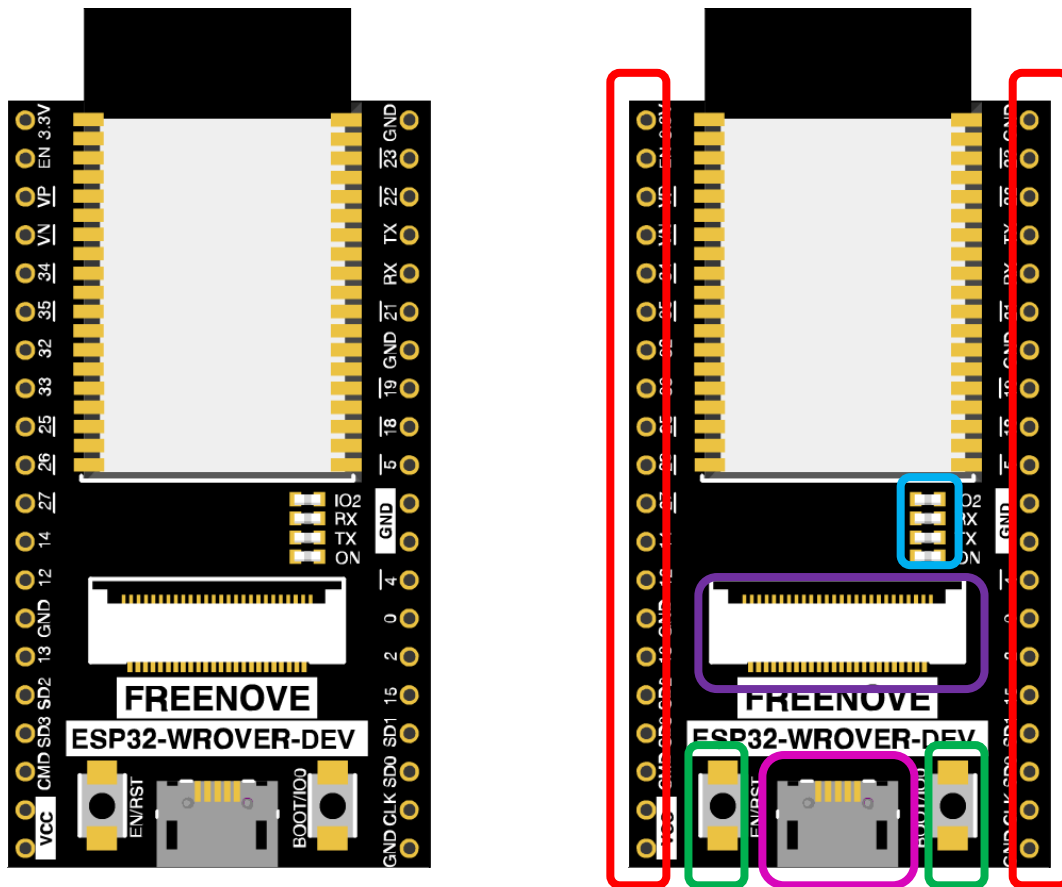
ESP32-WROVER has launched a total of two antenna packages, PCB on-board antenna and IPEX antenna respectively. The PCB on-board antenna is an integrated antenna in the chip module itself, so it is convenient to carry and design. The IPEX antenna is a metal antenna derived from the integrated antenna of the chip module itself, which is used to enhance the signal of the module.








In this tutorial, the ESP32-WROVER is designed based on the PCB on-board antenna-packaged ESP32-WROVER module.



The hardware interfaces of ESP32-WROVER are distributed as follows:



Compare the left and right images. We've boxed off the resources on the ESP32-WROVER in different colors to facilitate your understanding of the ESP32-WROVER.

Box color	Corresponding resources introduction
	GPIO pin
	LED indicator
	Camera interface
	Reset button, Boot mode selection button
	USB port

Name	No.	Type	Function
GND	1	P	Ground
3V3	2	P	Power supply
EN	3	I	Module-enable signal. Active high.
SENSOR_VP	4	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_VN	5	I	GPIO39, ADC1_CH3, RTC_GPIO3
IO34	6	I	GPIO34, ADC1_CH6, RTC_GPIO4
IO35	7	I	GPIO35, ADC1_CH7, RTC_GPIO5
IO32	8	I/O	GPIO32, XTAL_32K_P (32.768 kHz crystal oscillator input), ADC1_CH4, TOUCH9, RTC_GPIO9
IO33	9	I/O	GPIO33, XTAL_32K_N (32.768 kHz crystal oscillator output), ADC1_CH5, TOUCH8, RTC_GPIO8
IO25	10	I/O	GPIO25, DAC_1, ADC2_CH8, RTC_GPIO6, EMAC_RXD0
IO26	11	I/O	GPIO26, DAC_2, ADC2_CH9, RTC_GPIO7, EMAC_RXD1
IO27	12	I/O	GPIO27, ADC2_CH7, TOUCH7, RTC_GPIO17, EMAC_RX_DV
IO14	13	I/O	GPIO14, ADC2_CH6, TOUCH6, RTC_GPIO16, MTMS, HSPICLK, HS2_CLK, SD_CLK, EMAC_TXD2
IO12 ¹	14	I/O	GPIO12, ADC2_CH5, TOUCH5, RTC_GPIO15, MTDI, HSPIQ, HS2_DATA2, SD_DATA2, EMAC_TXD3
GND	15	P	Ground
IO13	16	I/O	GPIO13, ADC2_CH4, TOUCH4, RTC_GPIO14, MTCK, HSPID, HS2_DATA3, SD_DATA3, EMAC_RX_ER
SHD/SD2 ²	17	I/O	GPIO9, SD_DATA2, SPIHD, HS1_DATA2, U1RXD
SWP/SD3 ²	18	I/O	GPIO10, SD_DATA3, SPIWP, HS1_DATA3, U1TXD
SCS/CMD ²	19	I/O	GPIO11, SD_CMD, SPICS0, HS1_CMD, U1RTS
SCK/CLK ²	20	I/O	GPIO6, SD_CLK, SPICLK, HS1_CLK, U1CTS
SDO/SD0 ²	21	I/O	GPIO7, SD_DATA0, SPIQ, HS1_DATA0, U2RTS
SDI/SD1 ²	22	I/O	GPIO8, SD_DATA1, SPID, HS1_DATA1, U2CTS
IO15	23	I/O	GPIO15, ADC2_CH3, TOUCH3, MTDO, HSPICS0, RTC_GPIO13, HS2_CMD, SD_CMD, EMAC_RXD3
IO2	24	I/O	GPIO2, ADC2_CH2, TOUCH2, RTC_GPIO12, HSPWP, HS2_DATA0, SD_DATA0
IO0	25	I/O	GPIO0, ADC2_CH1, TOUCH1, RTC_GPIO11, CLK_OUT1, EMAC_TX_CLK
IO4	26	I/O	GPIO4, ADC2_CH0, TOUCH0, RTC_GPIO10, HSPHD, HS2_DATA1, SD_DATA1, EMAC_TX_ER
NC1	27	-	-
NC2	28	-	-
IO5	29	I/O	GPIO5, VSPICS0, HS1_DATA6, EMAC_RX_CLK
IO18	30	I/O	GPIO18, VSPICLK, HS1_DATA7
IO19	31	I/O	GPIO19, VSPIQ, U0CTS, EMAC_TXD0
NC	32	-	-
IO21	33	I/O	GPIO21, VSPHD, EMAC_TX_EN
RXD0	34	I/O	GPIO3, U0RXD, CLK_OUT2
TXD0	35	I/O	GPIO1, U0TXD, CLK_OUT3, EMAC_RXD2
IO22	36	I/O	GPIO22, VSPWP, U0RTS, EMAC_TXD1
IO23	37	I/O	GPIO23, VSPID, HS1_STROBE
GND	38	P	Ground

Notice:

- GPIO12 is internally pulled high in the module and is not recommended for use as a touch pin.
- Pins SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3 and SCS/CMD, namely, GPIO6 to GPIO11 are connected to the SPI flash integrated on the module and are not recommended for other uses.

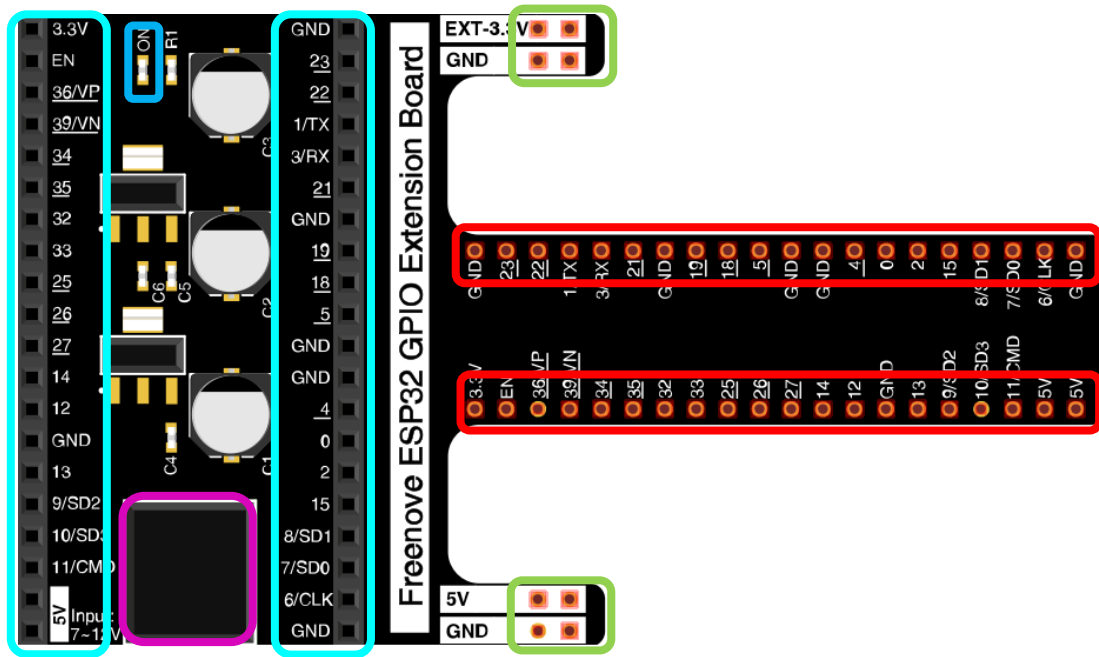
For more information, please visit: https://www.espressif.com/sites/default/files/documentation/esp32-wrover_datasheet_en.pdf

Any concerns? ✉ support@freenove.com






Extension board of the ESP32-WROVER

And we also design an extension board, so that you can use the ESP32 more easily in accordance with the circuit diagram provided. The followings are their photos.

The hardware interfaces of ESP32-WROVER are distributed as follows:



We've boxed off the resources on the ESP32-WROVER in different colors to facilitate your understanding of the ESP32-WROVER.

Box color	Corresponding resources introduction
	GPIO pin
	LED indicator
	GPIO interface of development board
	power supplied by the extension board
	External power supply

In ESP32, GPIO is an interface to control peripheral circuit. For beginners, it is necessary to learn the functions of each GPIO. The following is an introduction to the GPIO resources of the ESP32-WROVER development board.

In the following projects, we only use USB cable to power ESP32-WROVER by default.

In the whole tutorial, we don't use T extension to power ESP32-WROVER. So 5V and 3.3V (including EXT 3.3V) on the extension board are provided by ESP32-WROVER.

We can also use DC jack of extension board to power ESP32-WROVER. In this way, 5v and EXT 3.3v on extension board are provided by external power resource.

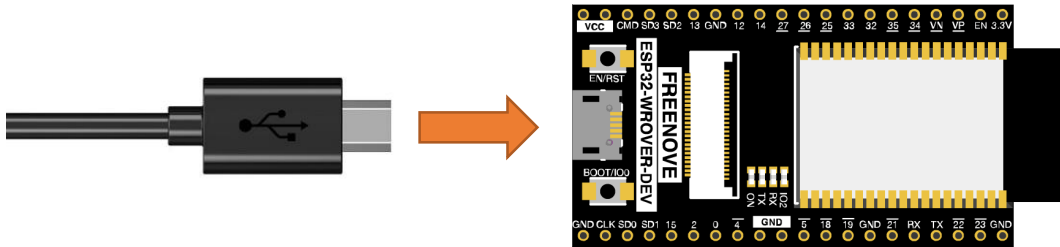
CH340

ESP32 uses CH340 to download codes. So before using it, we need to install CH340 driver in our computers.

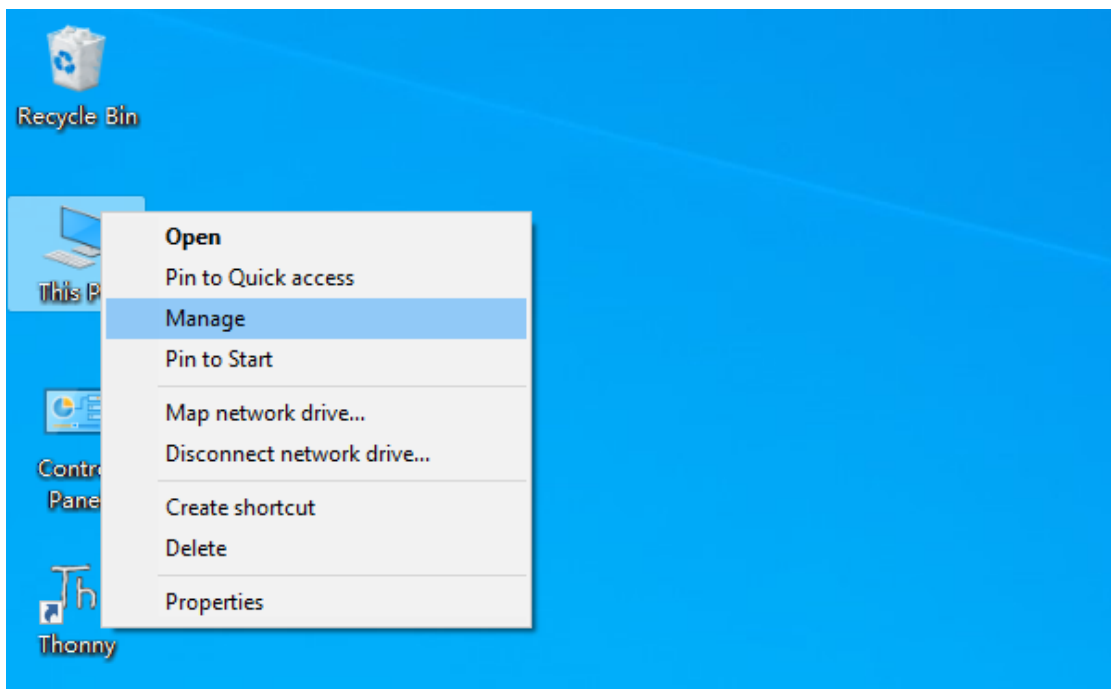
Windows

Check whether CH340 has been installed

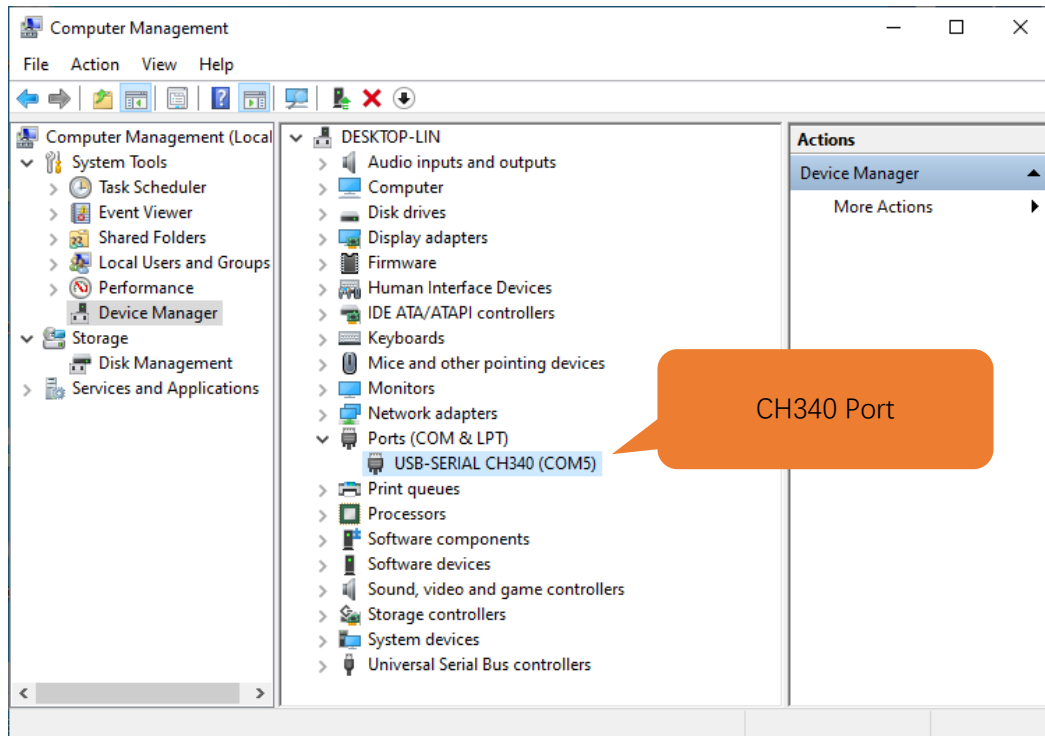
1. Connect your computer and ESP32 with a USB cable.



2. Turn to the main interface of your computer, select "This PC" and right-click to select "Manage".



- Click “Device Manager”. If your computer has installed CH340, you can see “USB-SERIAL CH340 (COMx)”. And you can click [here](#) to move to the next step.



Installing CH340

- First, download CH340 driver, click <http://www.wch-ic.com/search?q=CH340&t=downloads> to download the appropriate one based on your operating system.

index / search / search CH340

All (14)

Downloads (7)

Products (4)

Application (2)

Video (1)

News (0)

keyword CH340

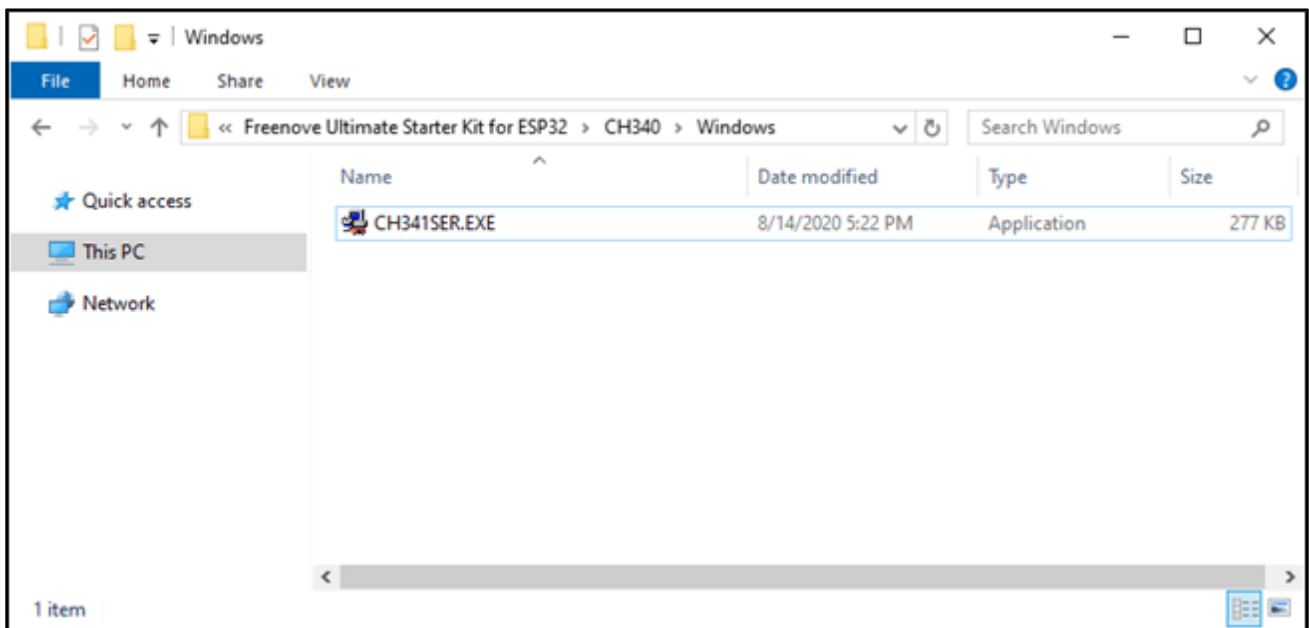
Downloads(7)

file category	file content	version	upload time
Driver&Tools			
CH341SER.EXE	CH340/CH341 USB to serial port Windows driver, supports 32/64-bit Windows 10/8.1/8/7/VISTA/XP, Server 2016/2012/2008/2003, 2000/ME/98	3.5	2019-03-18
CH341SER.ZIP	CH340/CH341 USB to serial port Windows driver, includes DLL dynamic library and non-standard baud rate settings and other instructions. Supports 32/64-bit Windows 10/8.1/8/7/VISTA/XP, Server 2016/2012/2008/2003, 2000/ME/98	3.5	2019-03-05
CH341SER_ANDROID...	CH340/CH341 USB to serial port Android free drive application library, for Android OS 3.1 and above version which supports USB Host mode already, no need to load Android kernel driver, no root privileges. Contains apk, lib library, file (Host Driver), App Demo Example (USB to UART Demo)	1.6	2019-04-19
CH341SER_LINUX...	CH340/CH341 USB to serial port LINUX driver	1.5	2018-03-18
CH341SER_MAC.ZIP	CH340/CH341 USB to serial port MAC OS driver	1.5	2018-07-05
Others			
PRODUCT_GUIDE.P...	Electronic selection of product selection manual, please refer to related product technical manual for more technical information.	1.4	2018-12-29
InstallNoteOn64...	Instructions for the driver after 18 years of August cannot be installed under some 64-bit WIN7 (English)	1.0	2019-01-10

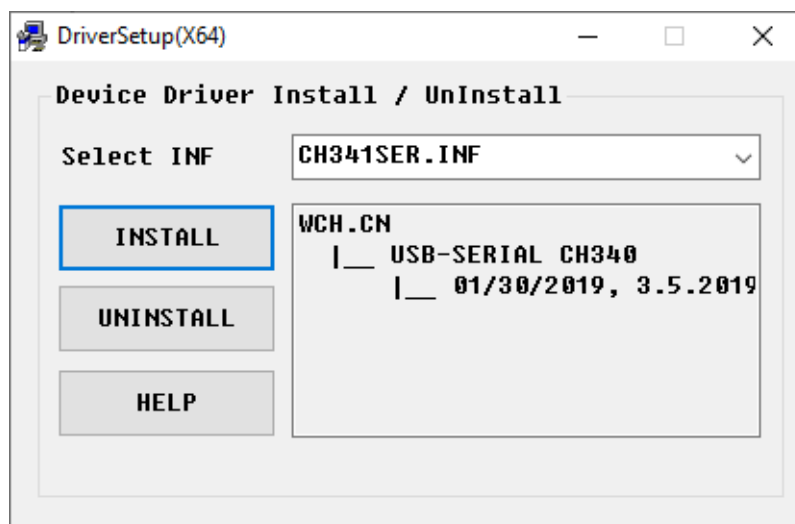
If you would not like to download the installation package, you can open “Freenove_LCD_Module/CH340”, we have prepared the installation package.

Name	Date modified	Type	Size
Linux	8/14/2020 5:24 PM	File folder	
MAC	8/14/2020 5:23 PM	File folder	
Windows	8/14/2020 5:23 PM	File folder	

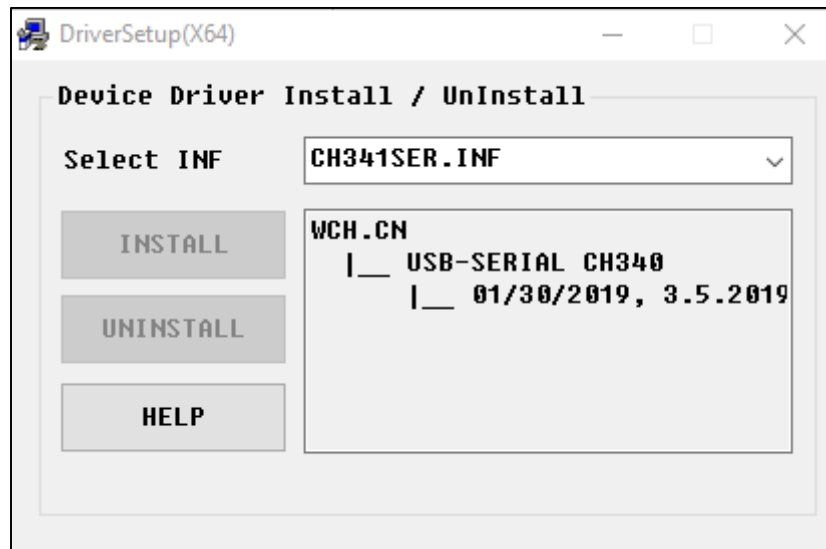
2. Open the folder "Freenove_LCD_Module/CH340/Windows/ch341ser"



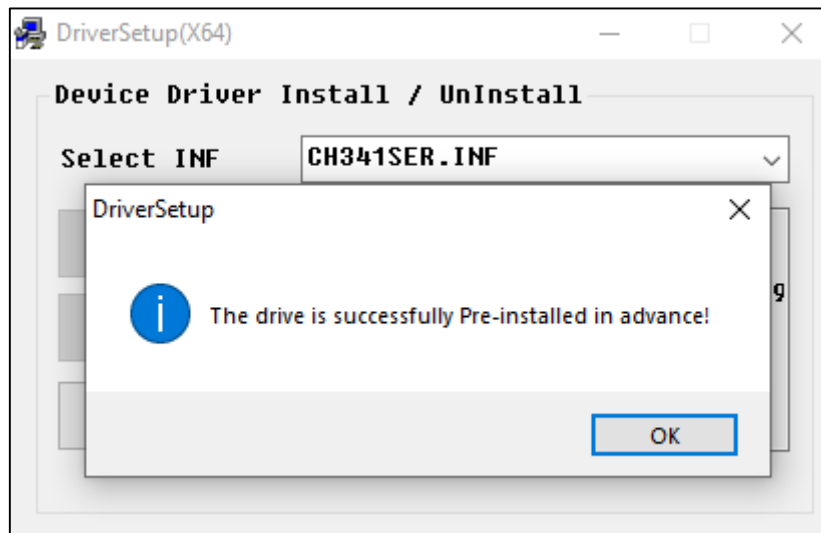
3. Double click "CH341SER.EXE".



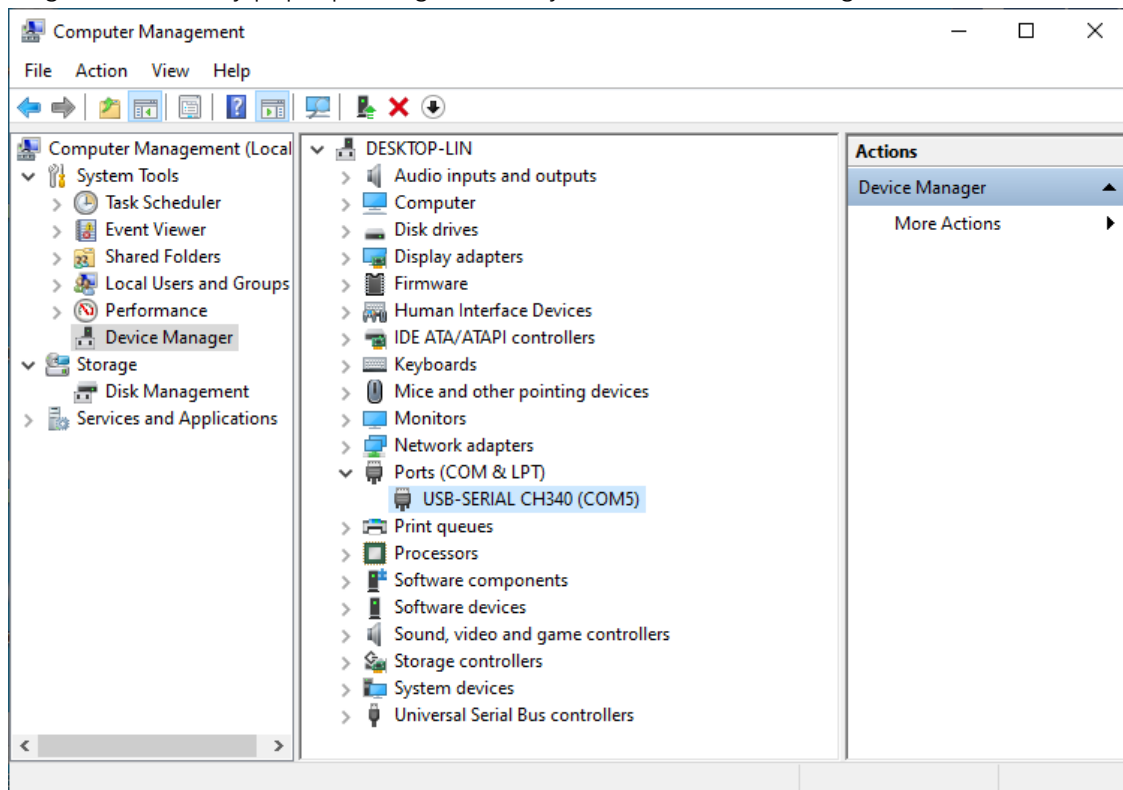
4. Click "INSTALL" and wait for the installation to complete.



5. Install successfully. Close all interfaces.



6. When ESP32 is connected to computer, select “This PC”, right-click to select “Manage” and click “Device Manager” in the newly pop-up dialog box, and you can see the following interface.



7. So far, CH340 has been installed successfully. Close all dialog boxes.

MAC

First, download CH340 driver, click <http://www.wch-ic.com/search?q=CH340&t=downloads> to download the appropriate one based on your operating system.

keyword ch340

Downloads (7)

file category	file content	version	upload time
Driver&Tools			
CH341SER.EXE	CH340/CH341 USB to serial port Windows driver, supports 32/64-bit Windows 10/8.1/8/7/VISTA/XP, Server 2016/2012/2008/2003, 2000/ME/98	3.5	2019-03-18
CH341SER.ZIP	CH340/CH341 USB to serial port Windows driver, includes DLL dynamic library and non-standard baud rate settings and other instructions. Supports 32/64-bit Windows 10/8.1/8/7/VISTA/XP, Server 2016/2012/2008/2003, 2000/ME/98	3.5	2019-03-05
CH341SER_ANDROID...	CH340/CH341 USB to serial port Android free drive application library, for Android OS 3.1 and above version which supports USB Host mode already, no need to load Android kernel driver, no root privileges. Contains apk, lib library file (Java Driver), App Demo Example, T Demo SDK).	1.6	2019-04-19
CH341SER_LINUX...	CH340/CH341 USB to serial port LINUX driver	1.5	2018-03-18
CH341SER_MAC.ZI...	CH340/CH341 USB to serial port MAC OS driver	1.5	2018-07-05

Others

If you would not like to download the installation package, you can open “Freenove_LCD_Module/CH340”, we have prepared the installation package.

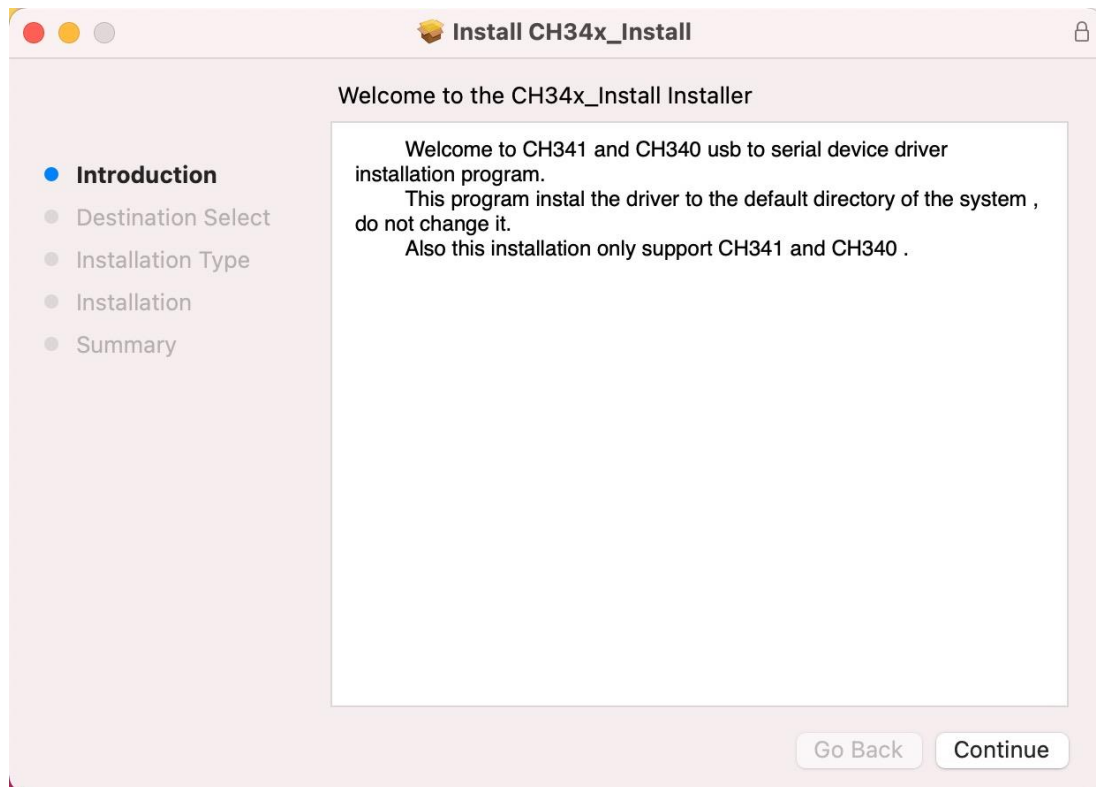
Second, open the folder “Freenove_LCD_Module/CH340/MAC/”

MAC

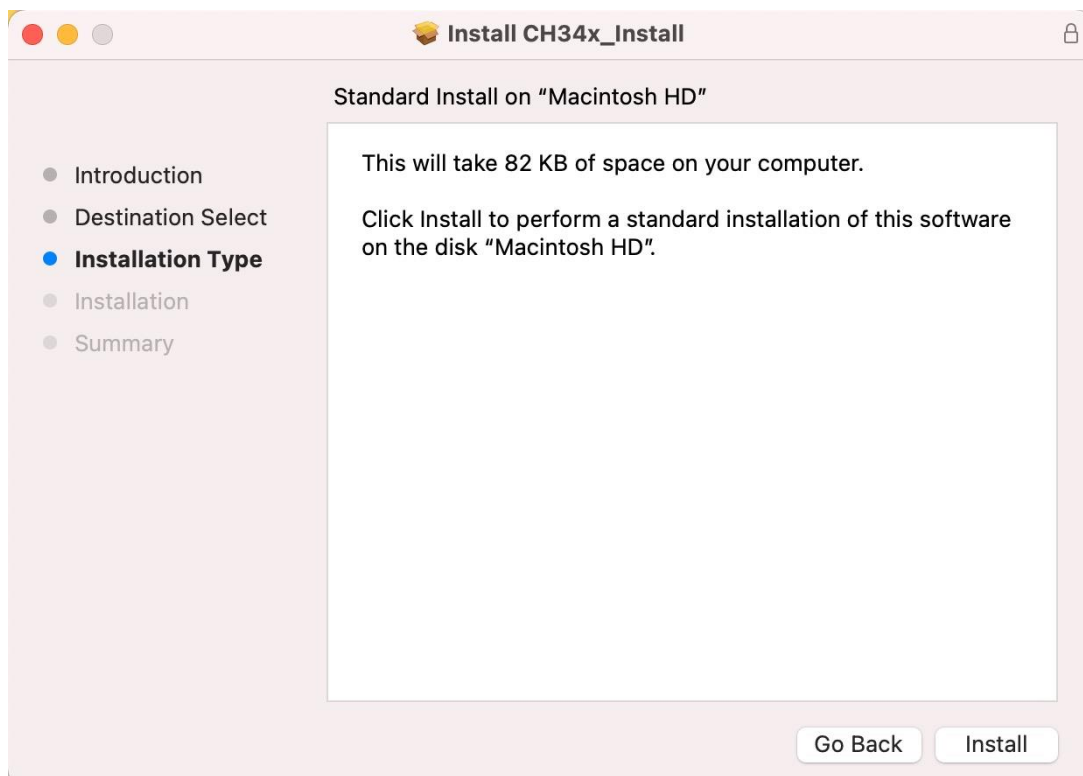
Name	Modified	Size	Kind
CH34x_Install_V1.5.pkg	Dec 8, 2016 at 2:00 PM	26 KB	Installer package
ReadMe.pdf	Dec 8, 2016 at 4:09 PM	146 KB	Adobe...ocument

Any concerns? [✉ support@freenove.com](mailto:support@freenove.com)

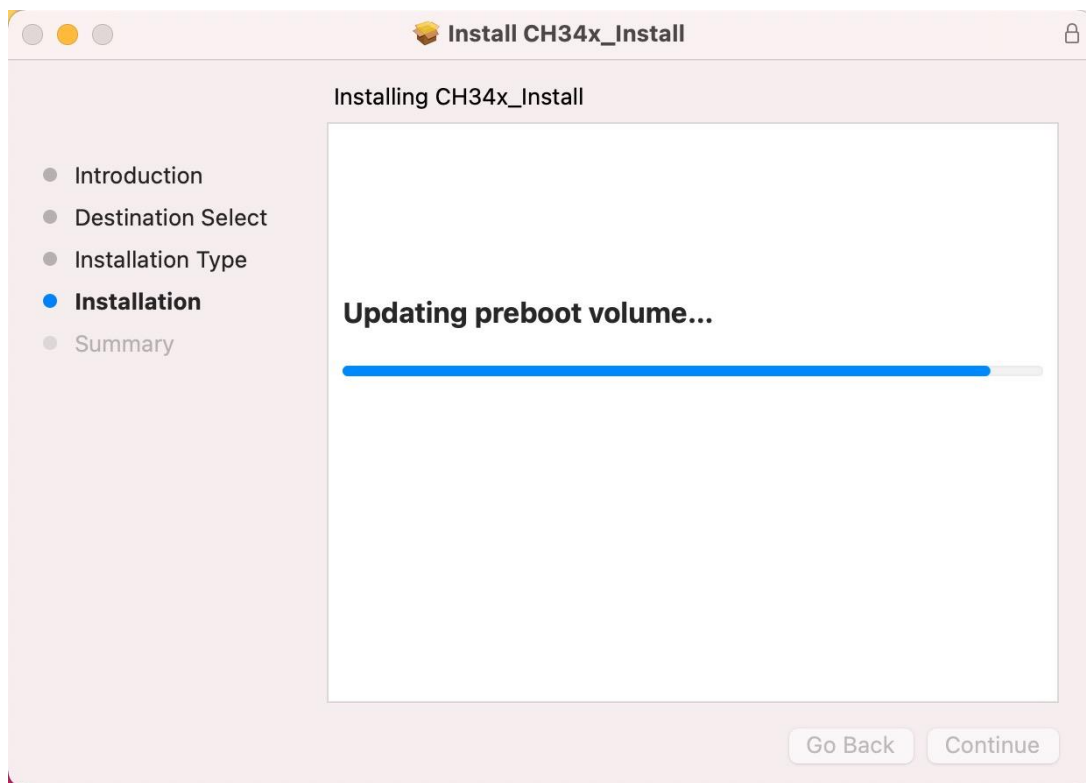
Third, click Continue.



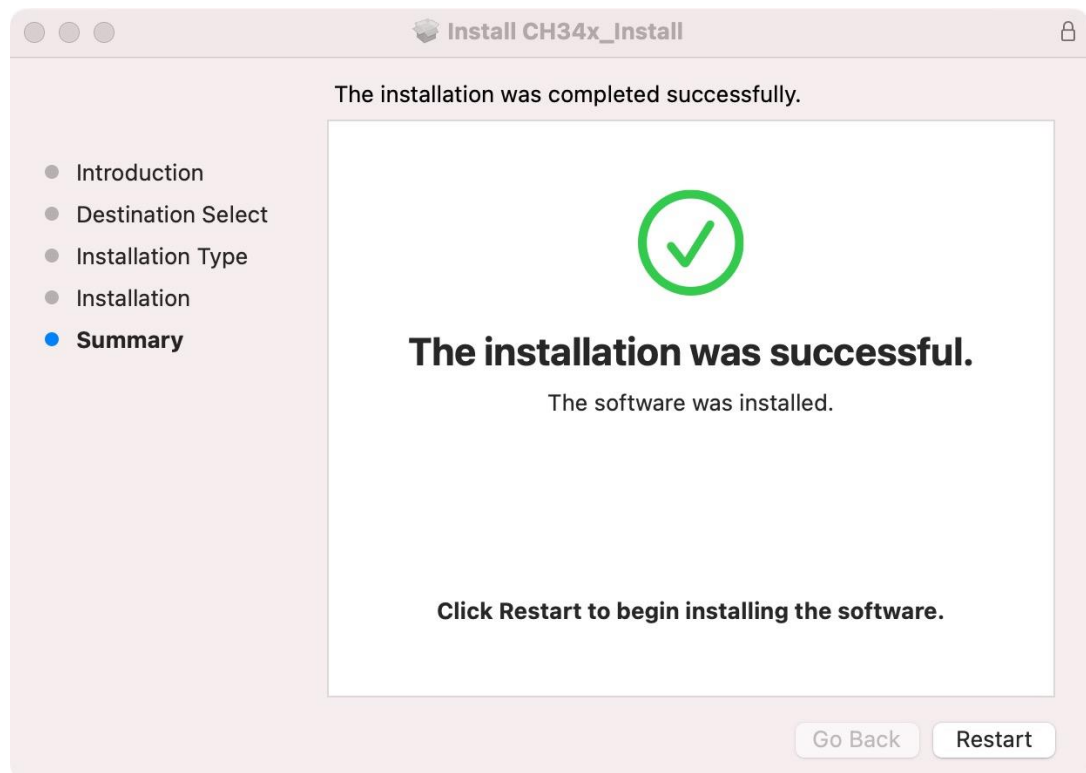
Fourth, click Install.



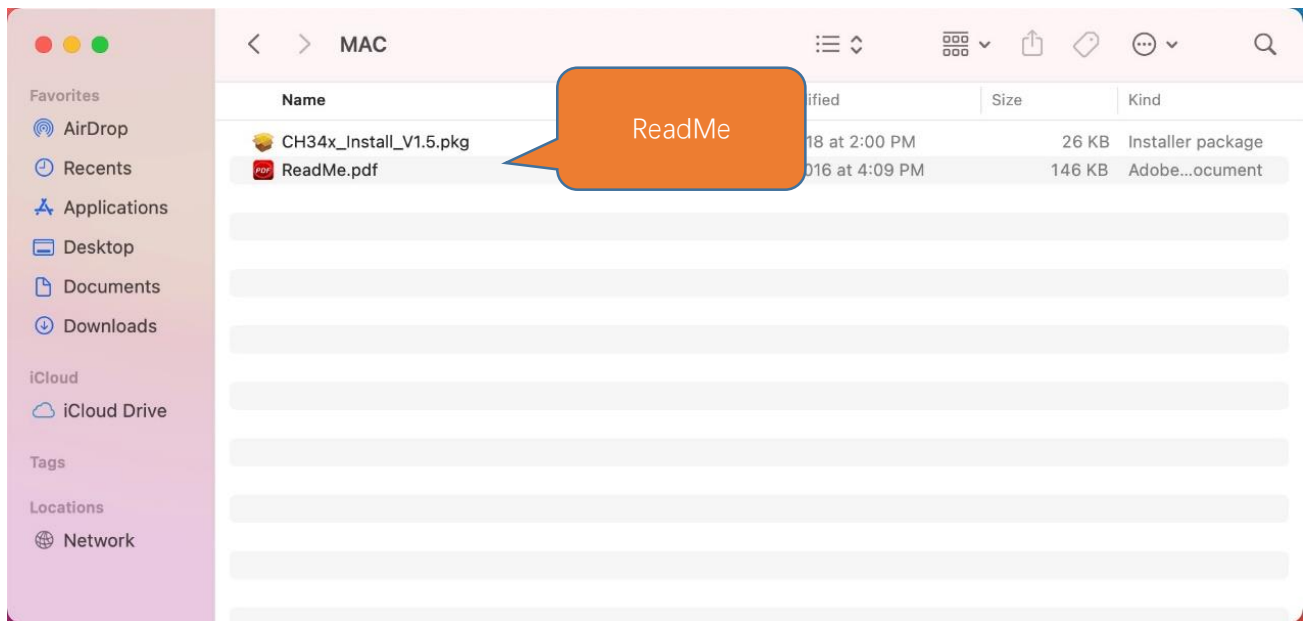
Then, waiting Finish.



Finally, restart your PC.



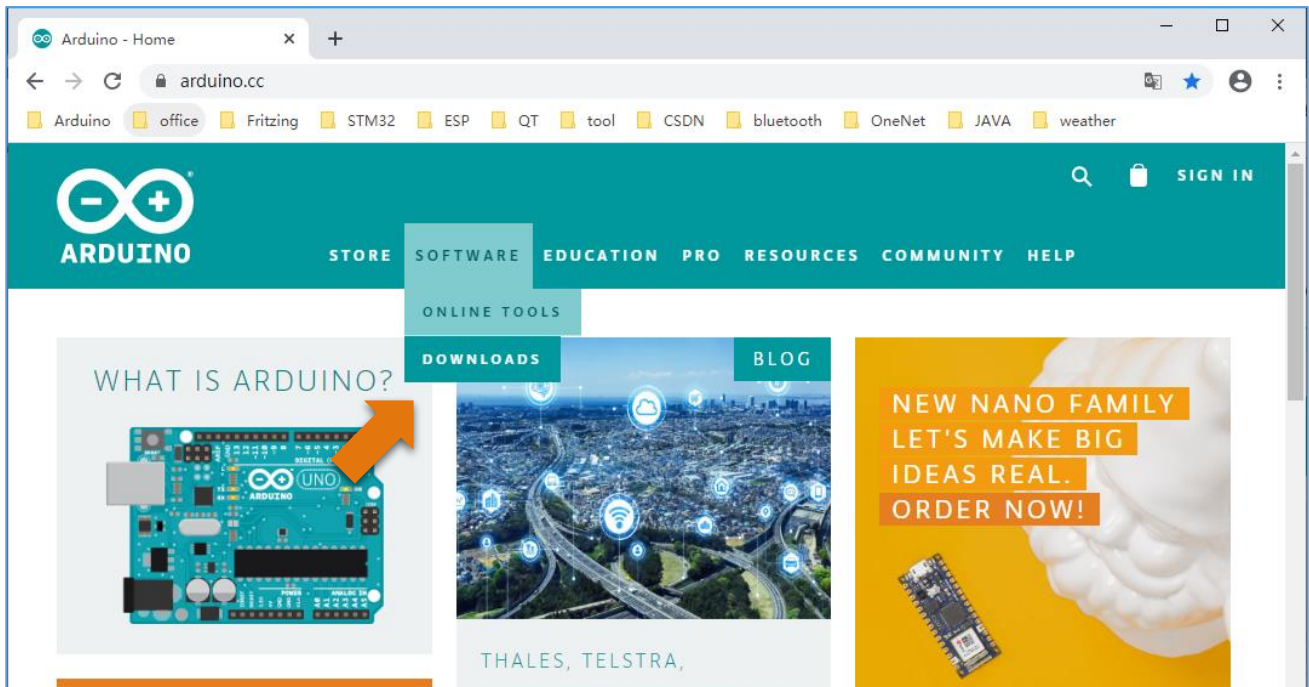
If you still haven't installed the CH340 by following the steps above, you can view readme.pdf to install it.



Programming Software

Arduino Software (IDE) is used to write and upload the code for Arduino Board.

First, install Arduino Software (IDE): visit <https://www.arduino.cc>, click "Download" to enter the download page.



Select and download corresponding installer according to your operating system. If you are a windows user, please select the "Windows Installer" to download to install the driver correctly.



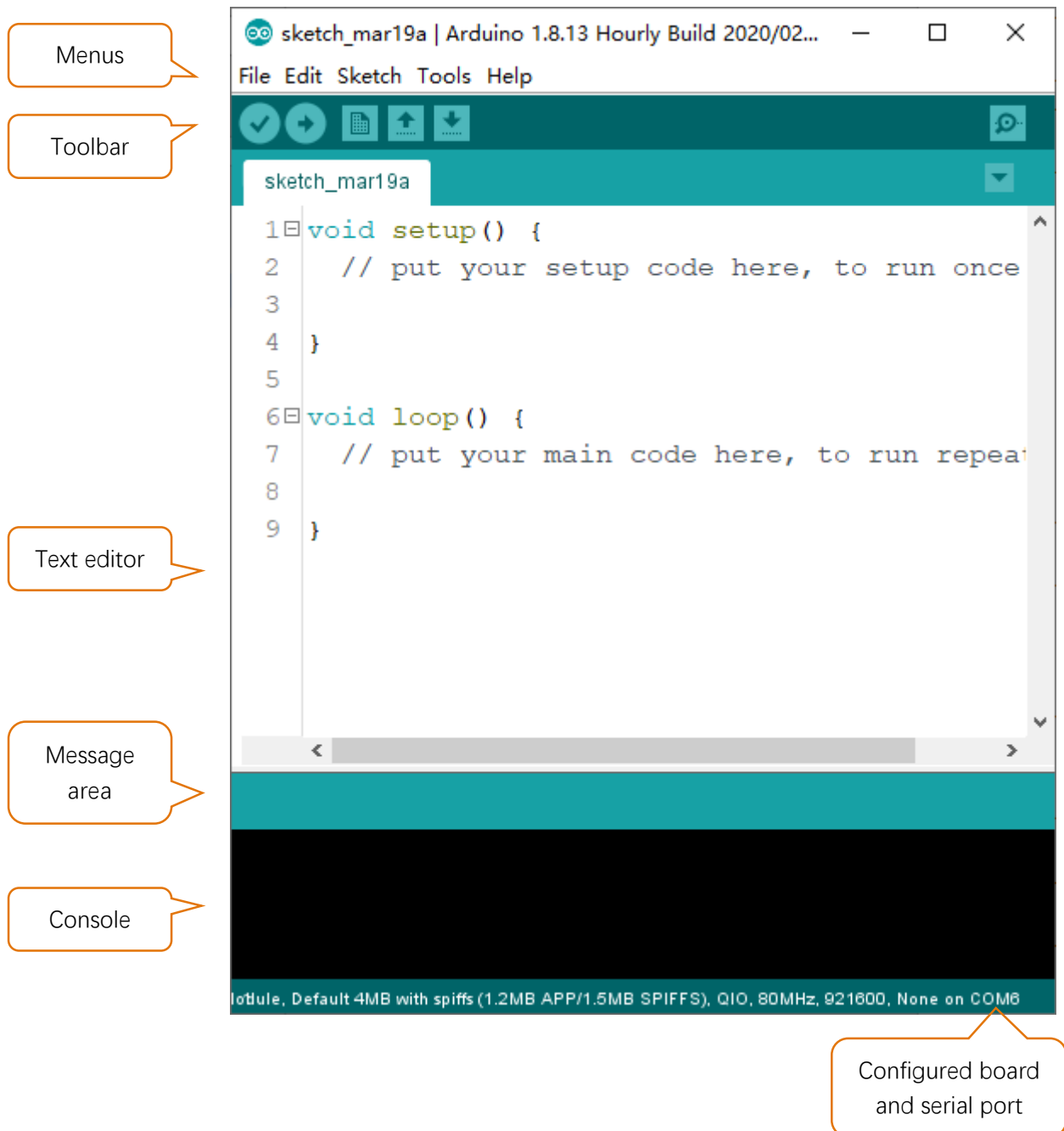
After the download completes, run the installer. For Windows users, there may pop up an installation dialog box of driver during the installation process. When it popes up, please allow the installation.

After installation is complete, an Arduino Software shortcut will be generated in the desktop. Run the Arduino Software.

Any concerns? ✉ support@freenove.com



The interface of Arduino Software is as follows:



Programs written with Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and saved with the file extension **.ino**. The editor has features for cutting/pasting and searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.



Verify

Check your code for compile errors .



Upload

Compile your code and upload them to the configured board.



New

Create a new sketch.



Open

Present a menu of all the sketches in your sketchbook. Clicking one will open it within the current window and overwrite its content.



Save

Save your sketch.



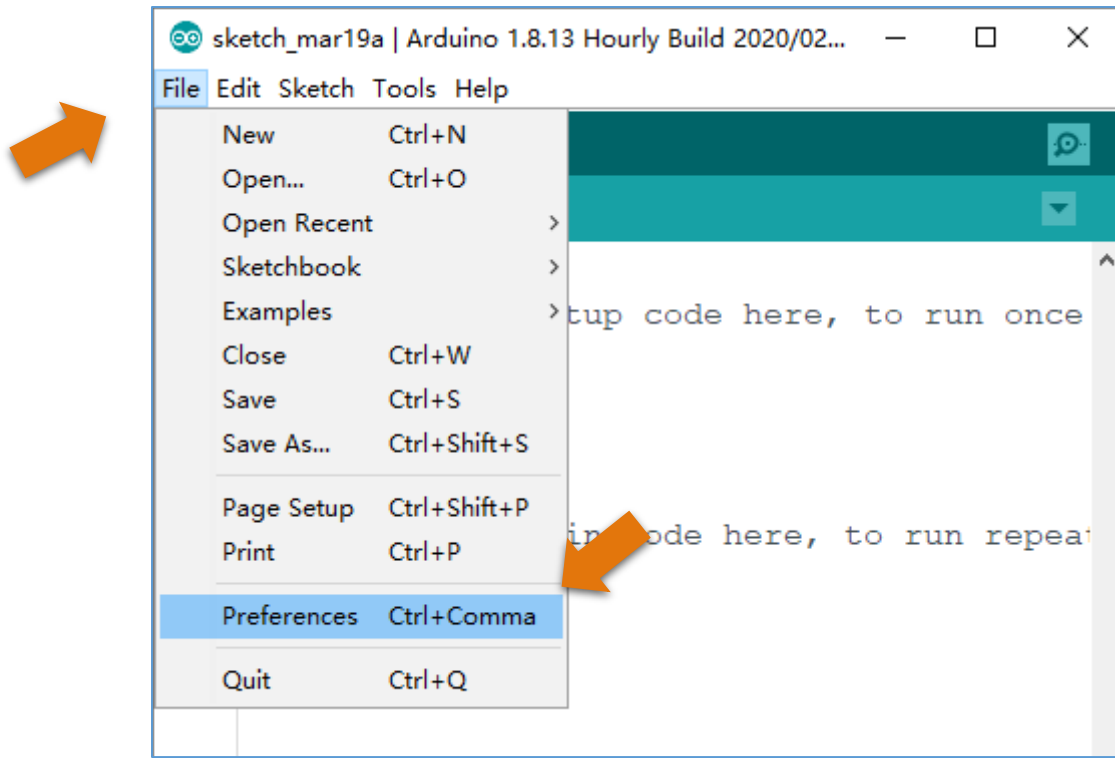
Serial Monitor

Open the serial monitor.

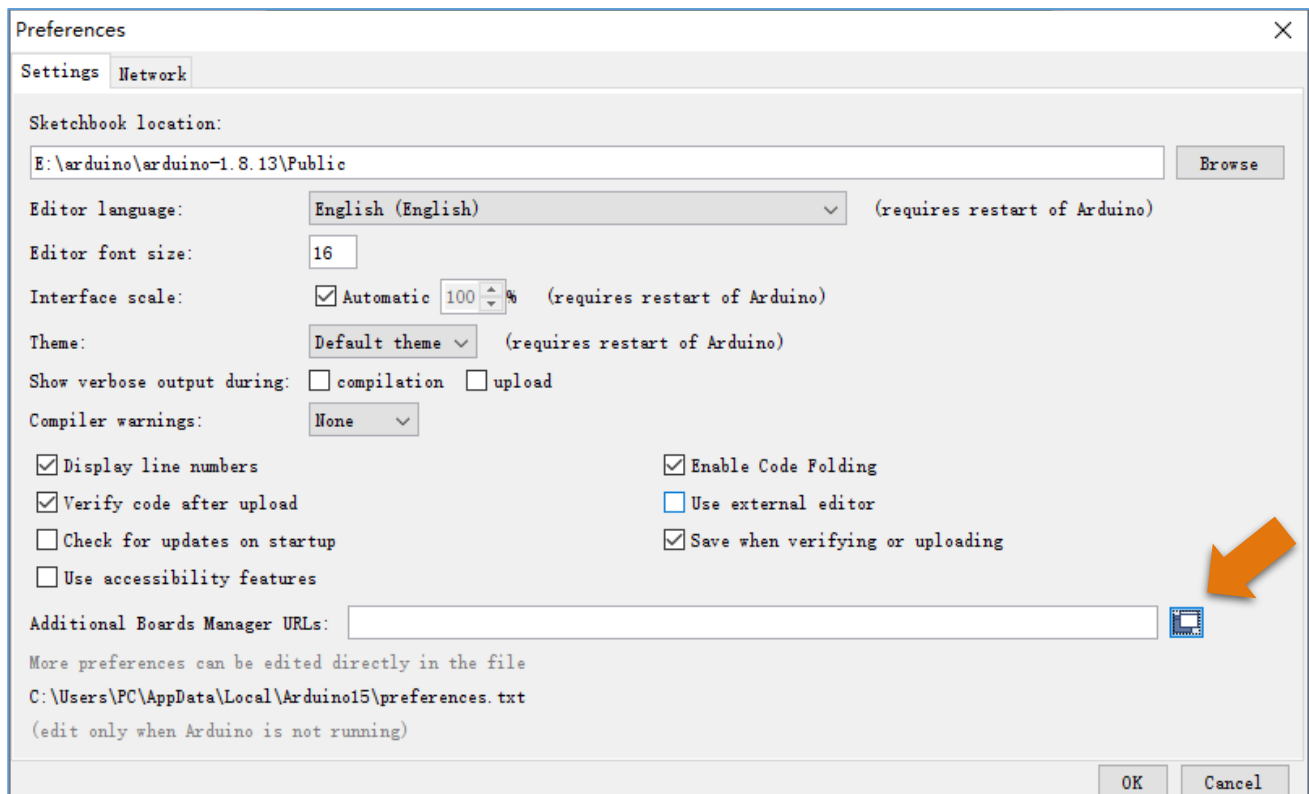
Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

Environment Configuration

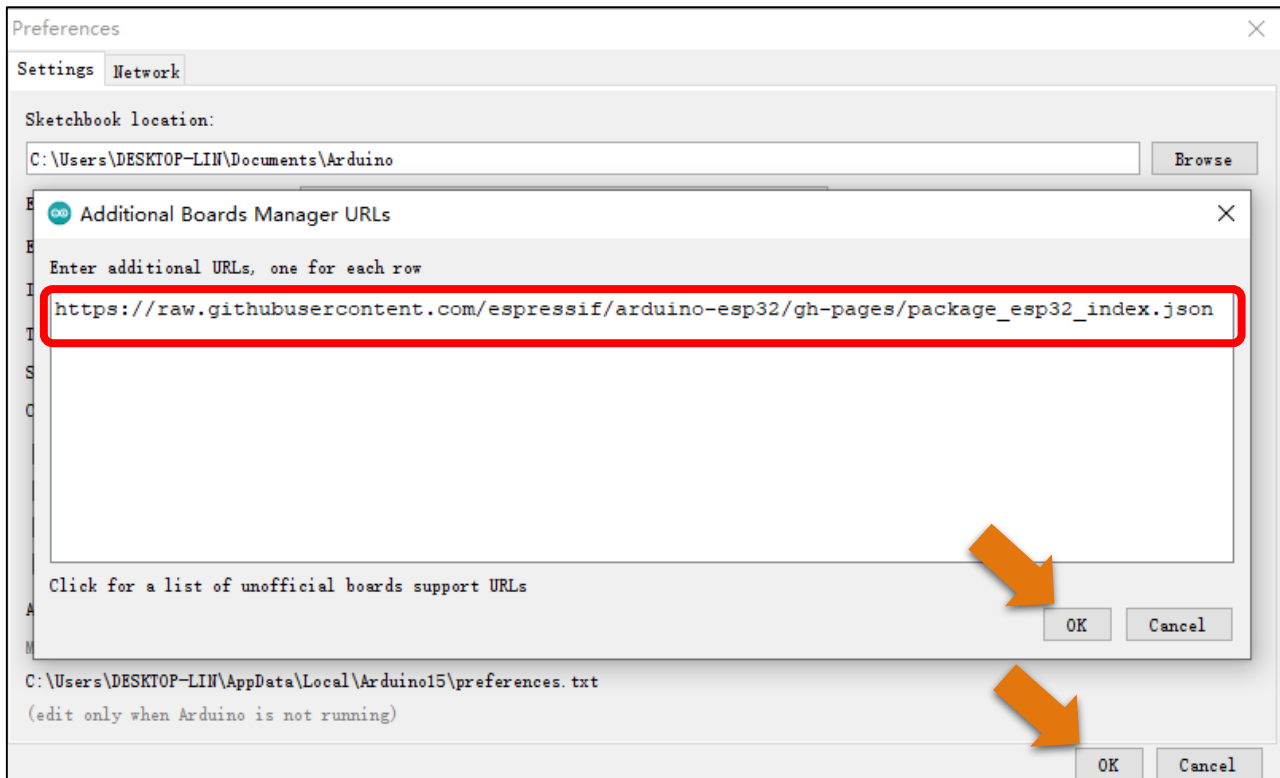
First, open the software platform arduino, and then click File in Menus and select Preferences.



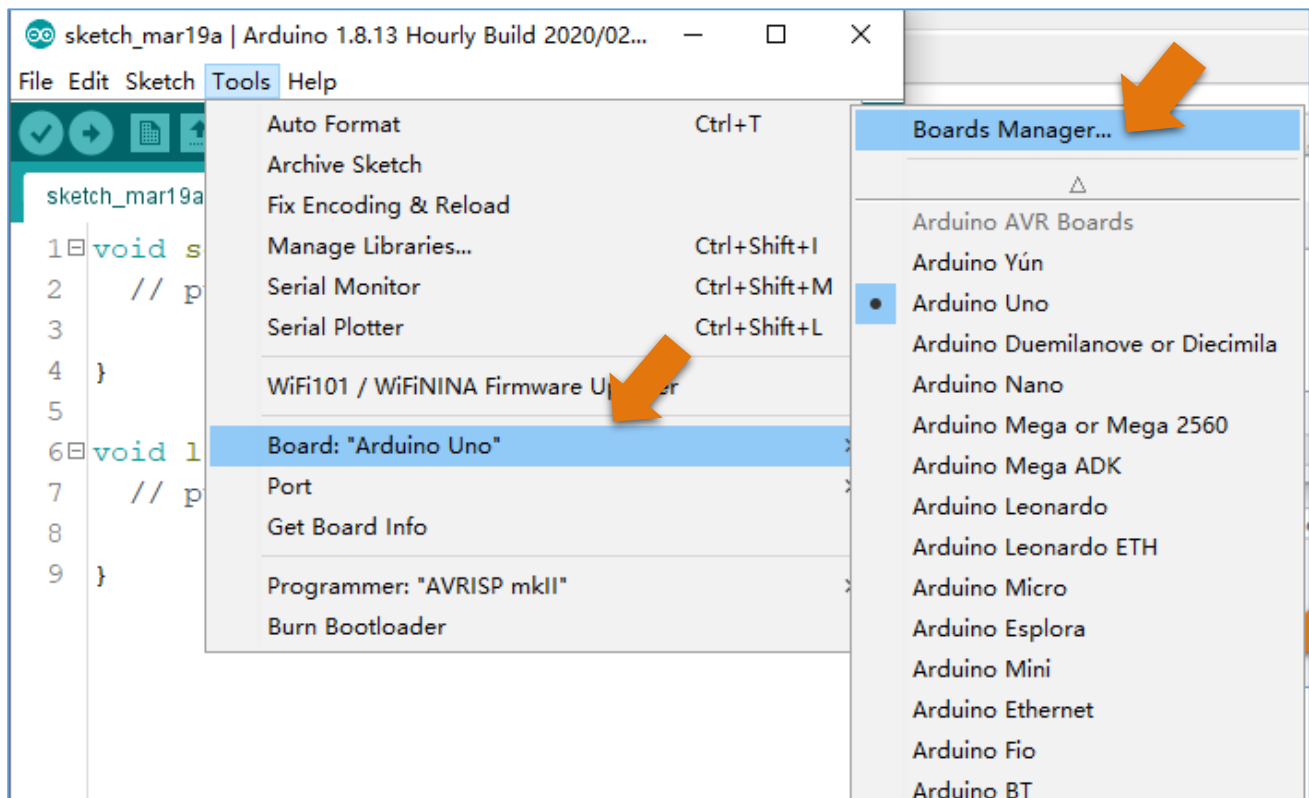
Second, click on the symbol behind "Additional Boards Manager URLs"



Third, fill in https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json in the new window, click OK, and click OK on the Preferences window again.



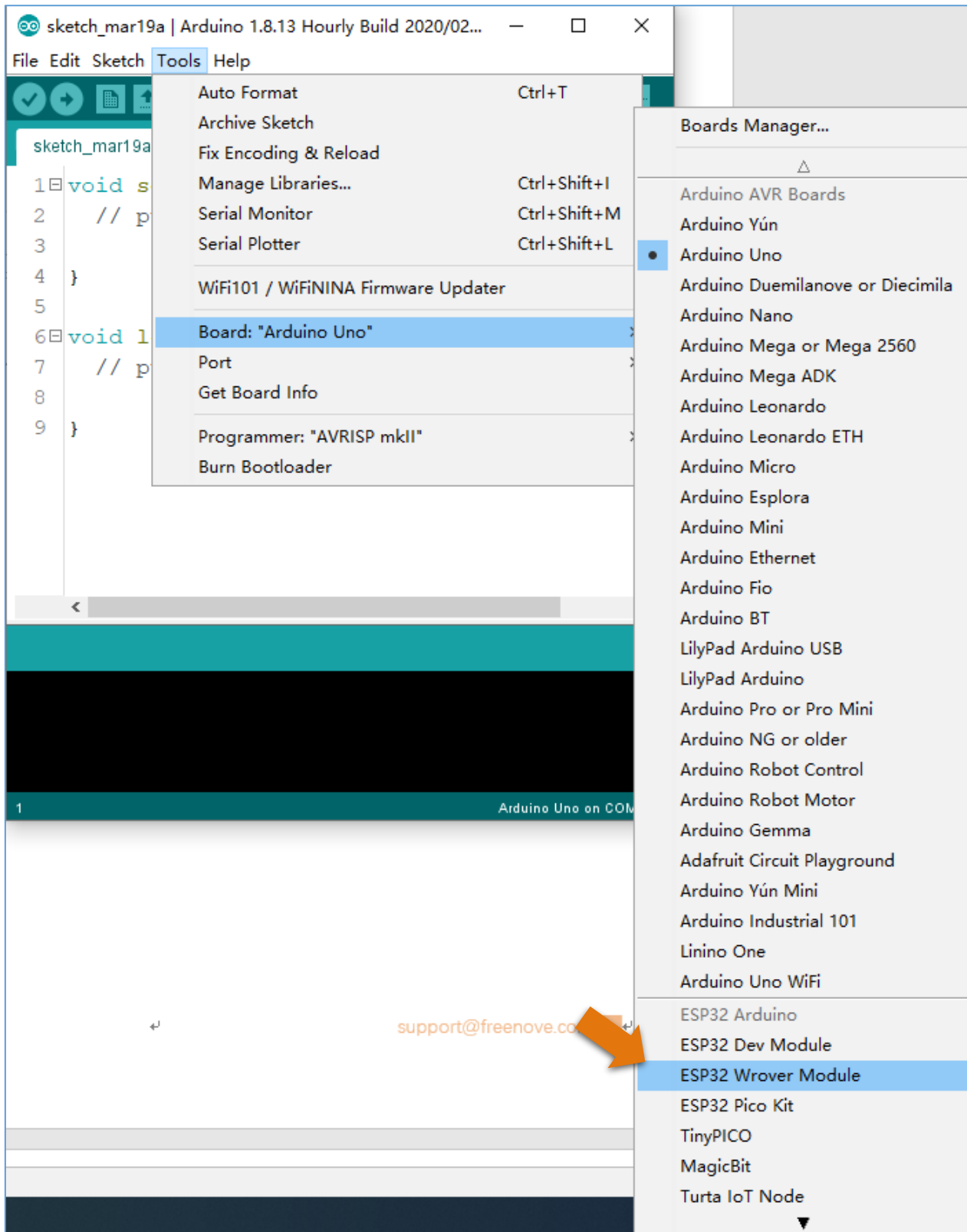
Fourth, click Tools in Menu, select Board:"ArduinoUno", and then select "Boards Manager".



Fifth, input "esp32" in the window below, and press Enter. click "Install" to install.



When finishing installation, click Tools in the Menus again and select Board: "Arduino Uno", and then you can see information of ESP32-WROVER. click "ESP32-WROVER" so that the ESP32 programming development environment is configured.



Notes for GPIO

Strapping Pin

There are five Strapping pins for ESP32: MTDI、GPIO0、GPIO2、MTDO、GPIO5。

With the release of the chip's system reset (power-on reset, RTC watchdog reset, undervoltage reset), the strapping pins sample the level and store it in the latch as "0" or "1" , and keep it until the chip is powered off or turned off.

Each Strapping pin is connecting to internal pull-up/pull-down. Connecting to high-impedance external circuit or without an external connection, a strapping pin's default value of input level will be determined by internal weak pull-up/pull-down. To change the value of the Strapping, users can apply an external pull-down/pull-up resistor, or use the GPIO of the host MCU to control the level of the strapping pin when the ESP32's power on reset is released.

When releasing the reset, the strapping pin has the same function as a normal pin.

The followings are default configurations of these five strapping pins at power-on and their functions under the corresponding configuration.

Voltage of Internal LDO (VDD_SDIO)					
Pin	Default	3.3 V		1.8 V	
MTDI	Pull-down	0		1	
Bootling Mode					
Pin	Default	SPI Boot		Download Boot	
GPIO0	Pull-up	1		0	
GPIO2	Pull-down	Don't-care		0	
Enabling/Disabling Debugging Log Print over U0TXD During Bootling					
Pin	Default	U0TXD Active		U0TXD Silent	
MTDO	Pull-up	1		0	
Timing of SDIO Slave					
Pin	Default	Falling-edge Sampling Falling-edge Output	Falling-edge Sampling Rising-edge Output	Rising-edge Sampling Falling-edge Output	Rising-edge Sampling Rising-edge Output
MTDO	Pull-up	0	0	1	1
GPIO5	Pull-up	0	1	0	1

Note:

- Firmware can configure register bits to change the settings of "Voltage of Internal LDO (VDD_SDIO)" and "Timing of SDIO Slave" after bootling.
- The MTDI is internally pulled high in the module, as the flash and SRAM in ESP32-WROVER only support a power voltage of 1.8 V (output by VDD_SDIO).

If you have any questions about the information of GPIO, you can click [here](#) to go back to ESP32-WROVER to view specific information about GPIO.

If you have any difficulties or questions with this tutorial or toolkit, feel free to ask for our quick and free technical support through support@freenove.com at any time.

or check: https://www.espressif.com/sites/default/files/documentation/esp32-wrover_datasheet_en.pdf

Any concerns? ✉ support@freenove.com

Flash Pin

GPIO6-11 has been used to connect the integrated SPI flash on the module, and is used when GPIO 0 is power on and at high level. Flash is related to the operation of the whole chip, so the external pin GPIO6-11 cannot be used as an experimental pin for external circuits, otherwise it may cause errors in the operation of the program.

GPIO16-17 has been used to connect the integrated PSRAM on the module.

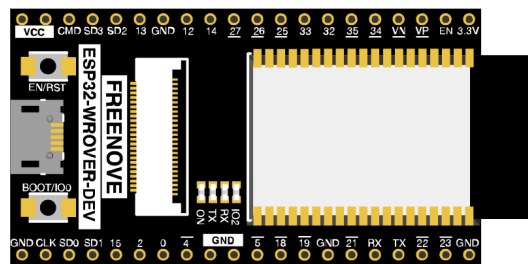
Because of external pull-up, MTDI pin is not suggested to be used as a touch sensor. For details, please refer to Peripheral Interface and Sensor chapter in "ESP32 Data_Sheet".

For more relevant information, please check:

https://www.espressif.com/sites/default/files/documentation/esp32-wrover_datasheet_en.pdf

Cam Pin

When using the camera of our ESP32-WROVER, please check the pins of it. Pins with underlined numbers are used by the camera function, if you want to use other functions besides it, please avoid using them.



CAM_Pin	GPIO_pin
I2C_SDA	GPIO26
I2C_SCL	GPIO27
CSI_VYSNC	GPIO25
CSI_HREF	GPIO23
CSI_Y9	GPIO35
XCLK	GPIO21
CSI_Y8	GPIO34
CSI_Y7	GPIO39
CSI_PCLK	GPIO22
CSI_Y6	GPIO36
CSI_Y2	GPIO4
CSI_Y5	GPIO19
CSI_Y3	GPIO5
CSI_Y4	GPIO18

If you have any questions about the information of GPIO, you can click [here](#) to go back to ESP32-WROVER to view specific information about GPIO.

or check: https://www.espressif.com/sites/default/files/documentation/esp32-wrover_datasheet_en.pdf

Chapter 1 LCD1602

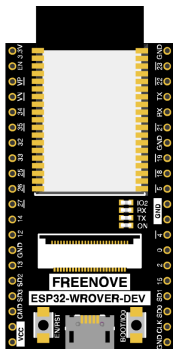
In this chapter, we will learn about the LCD1602 Display Screen

Project 1.1 LCD1602

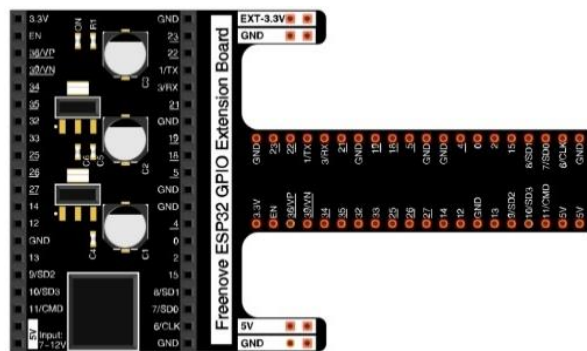
In this section we learn how to use LCD1602 to display something.

Component List

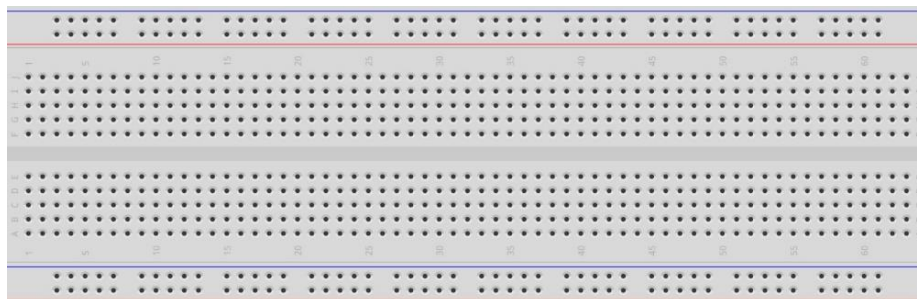
ESP32-WROVER x1



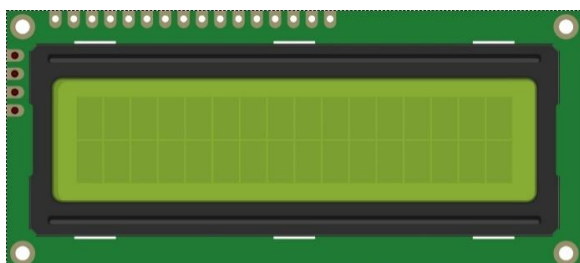
GPIO Extension Board x1



Breadboard x1



LCD1602 Module x1



Jumper F/M x4



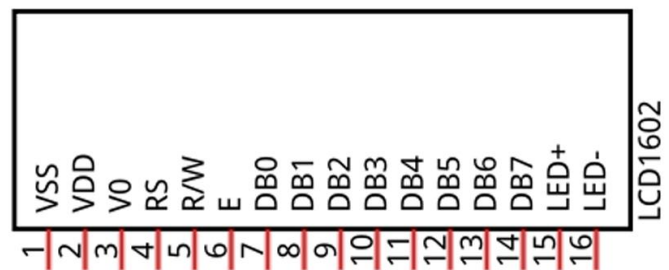
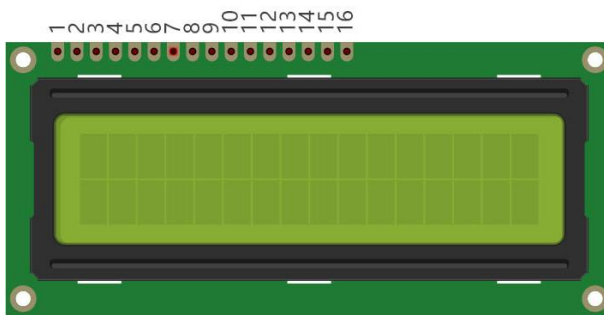
Component knowledge

I2C communication

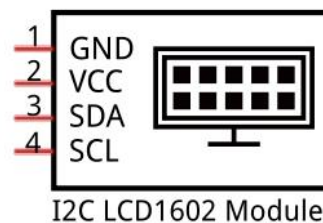
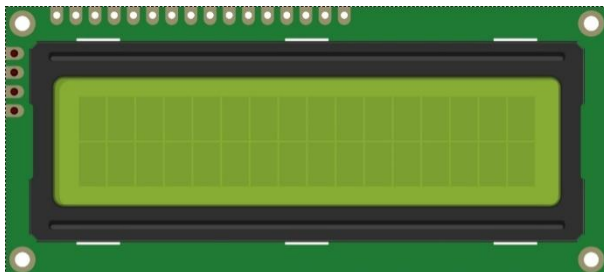
I2C (Inter-Integrated Circuit) is a two-wire serial communication mode, which can be used for the connection of micro controllers and their peripheral equipment. Devices using I2C communication must be connected to the serial data (SDA) line, and serial clock (SCL) line (called I2C bus). Each device has a unique address and can be used as a transmitter or receiver to communicate with devices connected to the bus.

LCD1602 communication

The LCD1602 display screen can display 2 lines of characters in 16 columns. It is capable of displaying numbers, letters, symbols, ASCII code and so on. As shown below is a monochrome LCD1602 display screen along with its circuit pin diagram

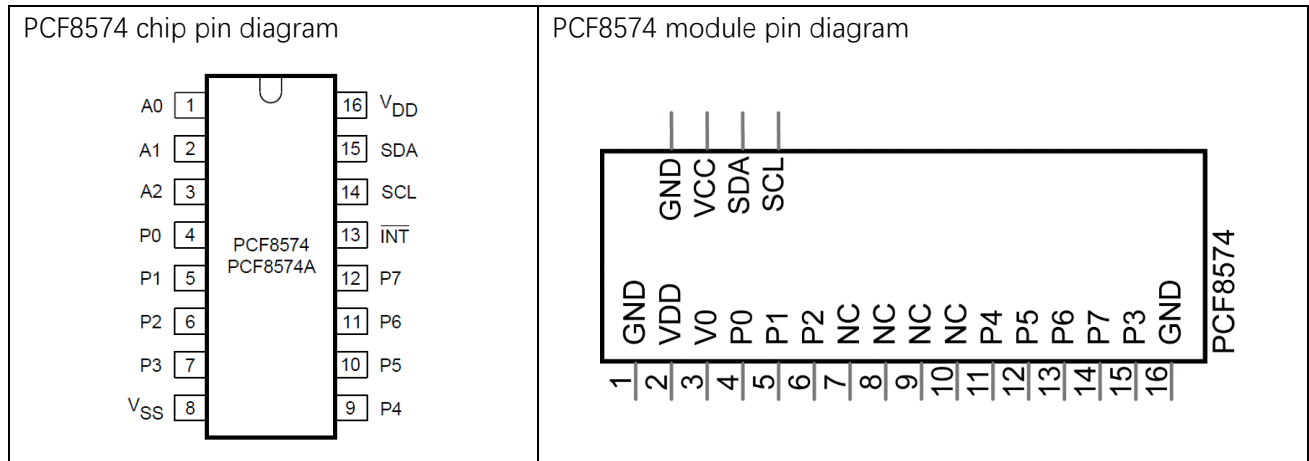


I2C LCD1602 display screen integrates a I2C interface, which connects the serial-input & parallel-output module to the LCD1602 display screen. This allows us to only use 4 lines to operate the LCD1602.

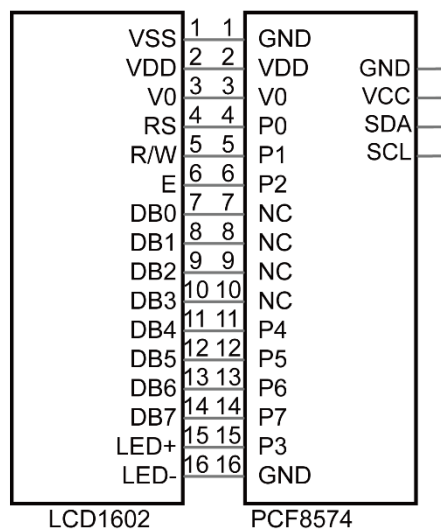


The serial-to-parallel IC chip used in this module is PCF8574T (PCF8574AT), and its default I2C address is 0x27(0x3F).

Below is the PCF8574 pin schematic diagram and the block pin diagram:



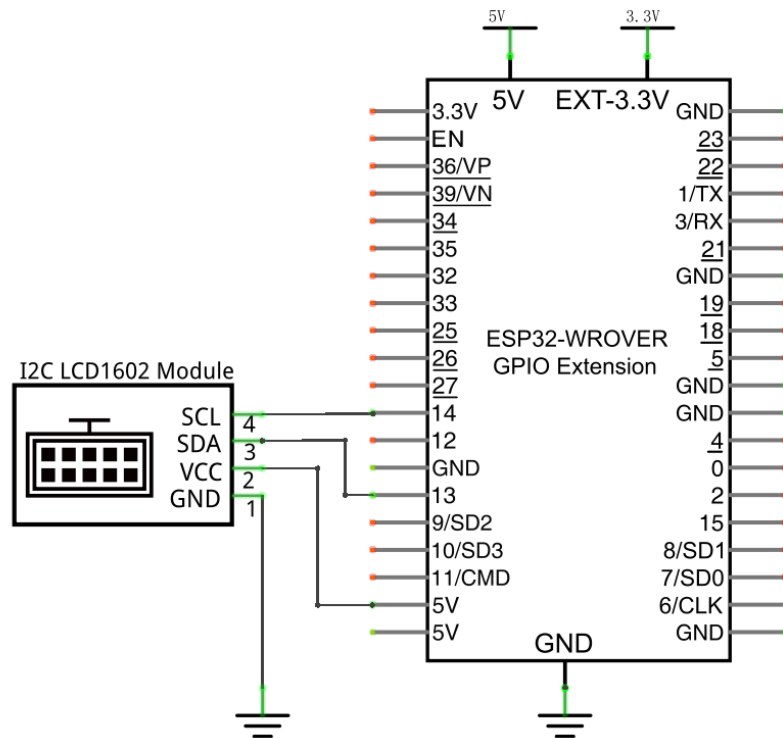
PCF8574 module pin and LCD1602 pin are corresponding to each other and connected with each other:



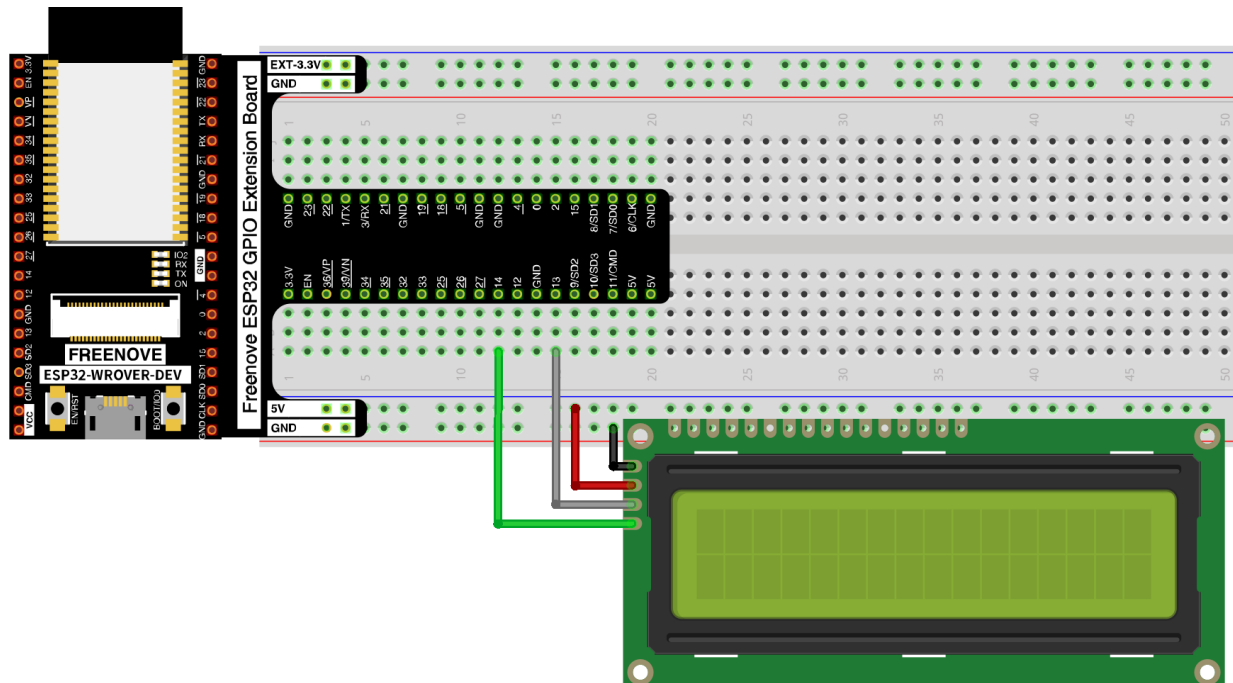
So we only need 4 pins to control the 16 pins of the LCD1602 display screen through the I2C interface. In this project, we will use the I2C LCD1602 to display some static characters and dynamic variables.

Circuit

Schematic diagram



Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com

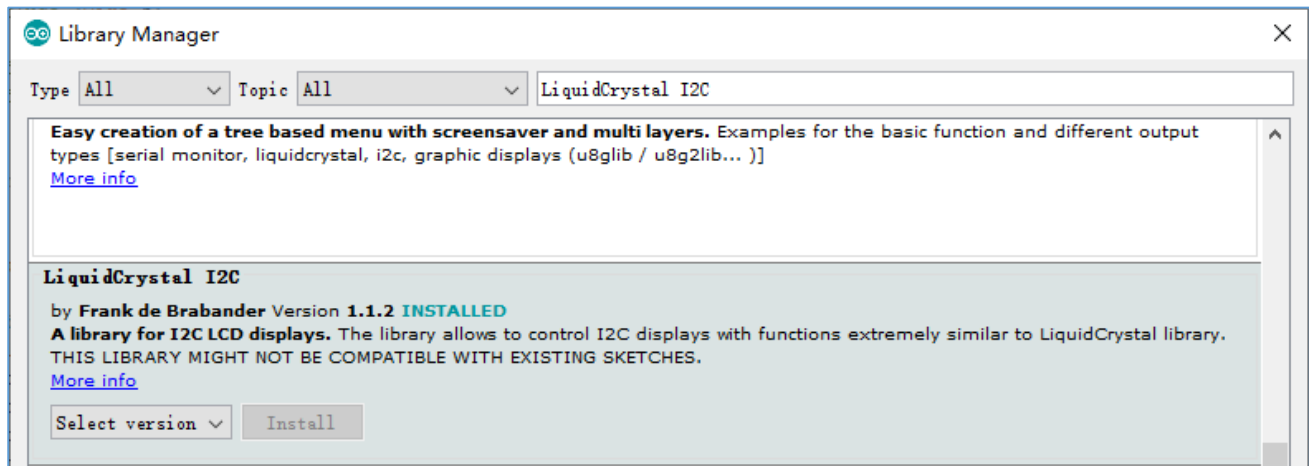


Sketch

Before writing code, we need to import the library needed.

How to install the library

We use the third party library LiquidCrystal I2C. If you haven't installed it yet, please do so before learning. The steps to add third-party Libraries are as follows: open arduino->Sketch->Include library-> Manage libraries. Enter " LiquidCrystal I2C" in the search bar and select " LiquidCrystal I2C " for installation.

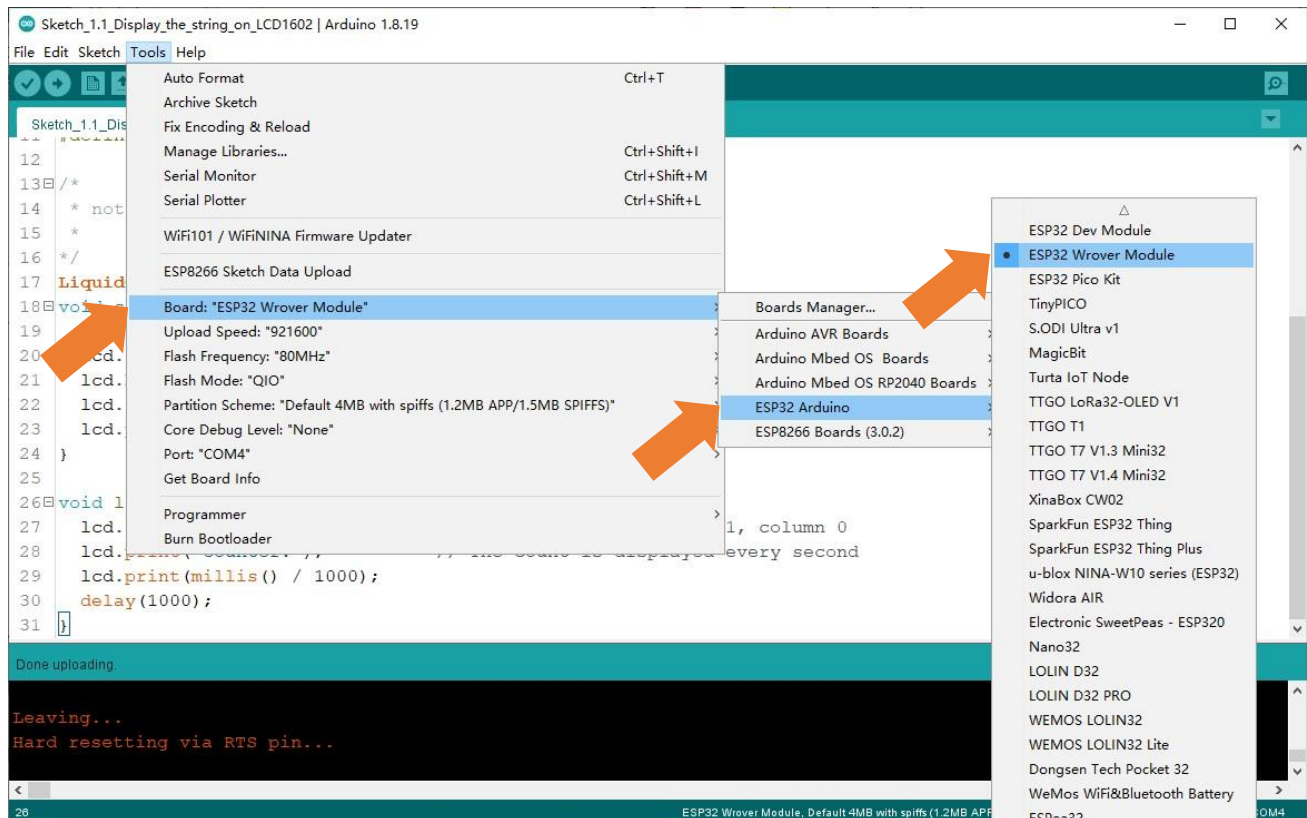


There is another way you can install libraries.

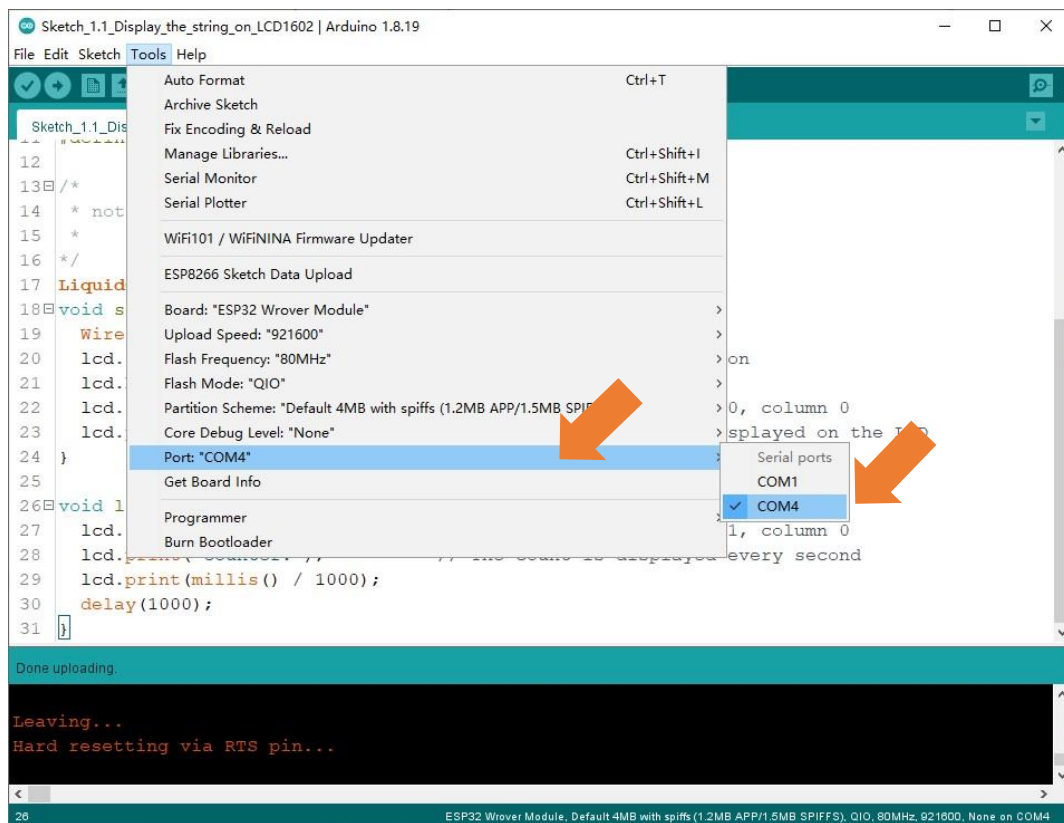
Click "Add .ZIP Library..." and then find **LiquidCrystal_I2C.zip** in libraries folder (this folder is in the folder unzipped from the ZIP file we provided). This library can facilitate our operation of I2C LCD1602.

Use I2C LCD 1602 to display characters and variables.

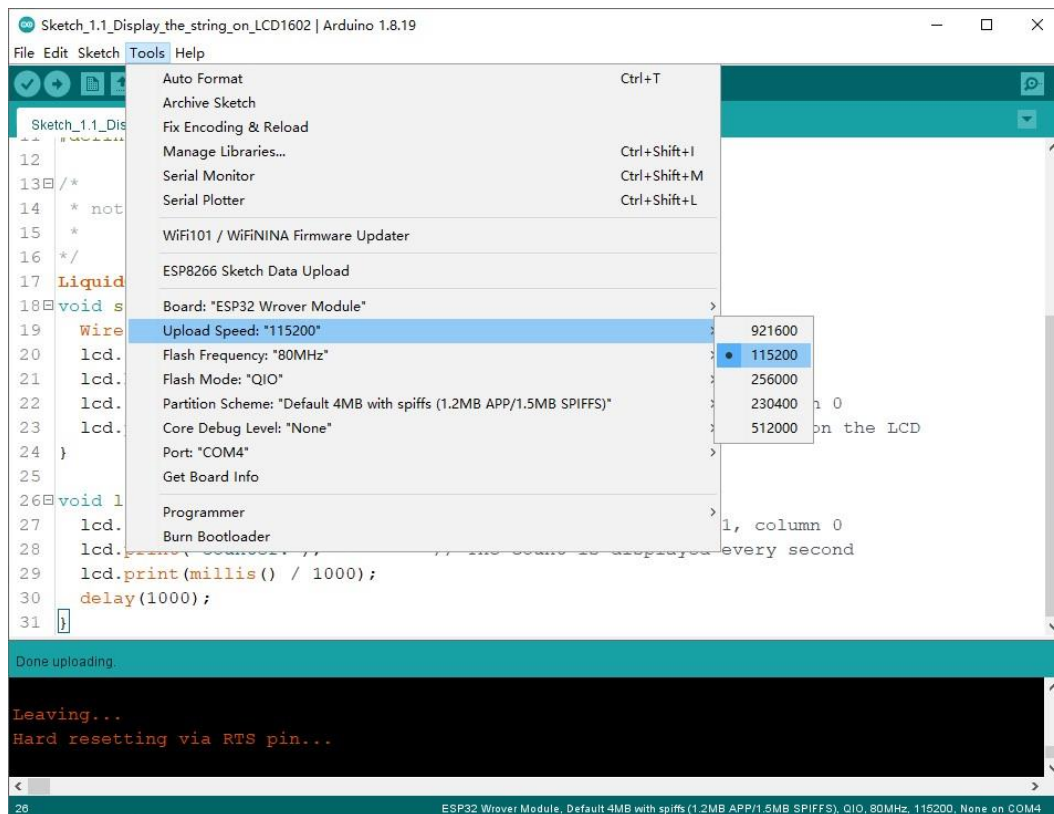
Before uploading the code, click "Tools", "Board" and select "ESP32 Wrover Module".



Select the serial port.

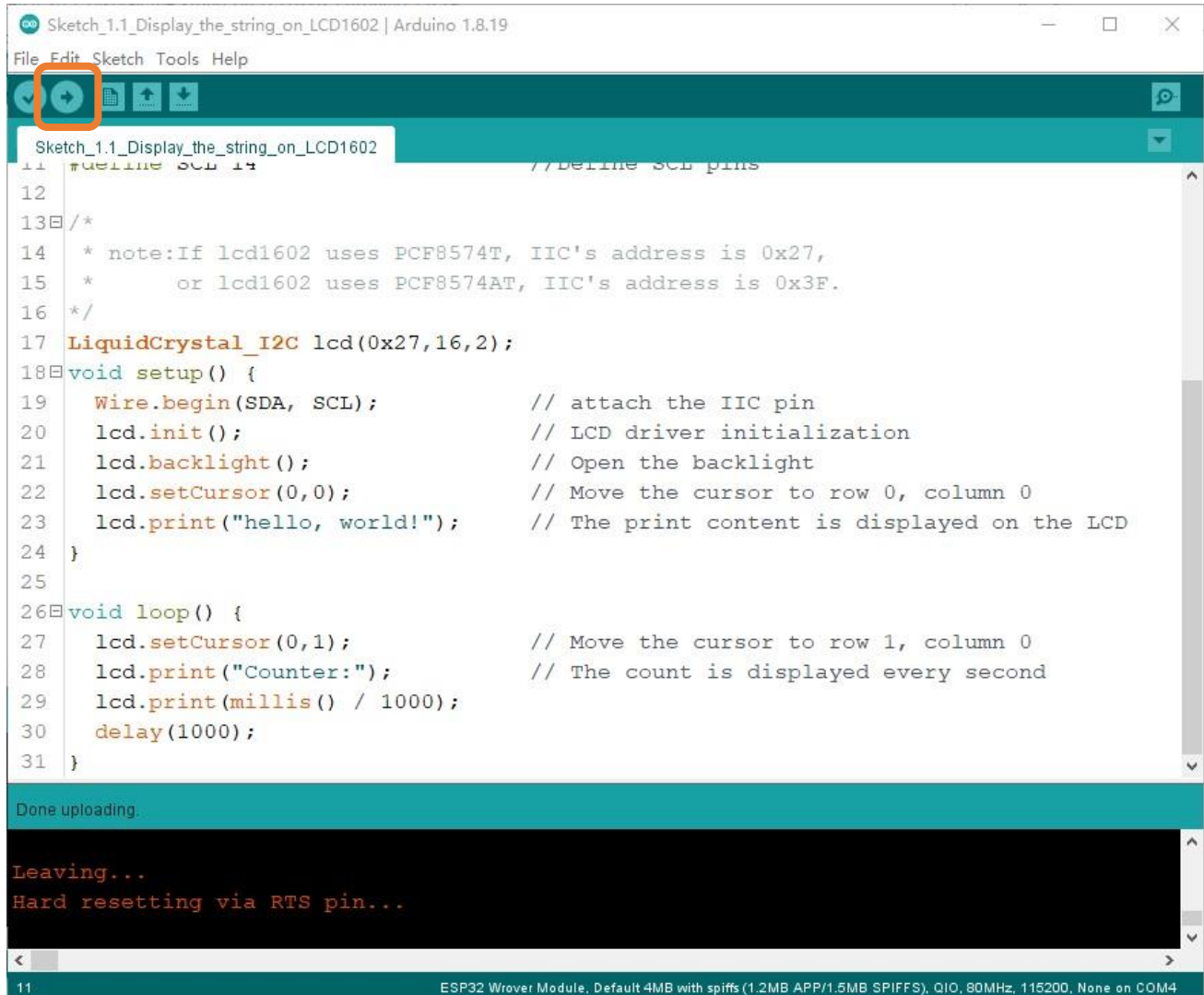


Note: For macOS users, if the uploading fails, please set the baud rate to 115200 before clicking "Upload Using Programmer".



Any concerns? [✉ support@freenove.com](mailto:support@freenove.com)

Sketch_1.1_Display_the_string_on_LCD1602



```

Sketch_1.1_Display_the_string_on_LCD1602 | Arduino 1.8.19
File Edit Sketch Tools Help
Sketch_1.1_Display_the_string_on_LCD1602
11 #define SCL 14 //Define SCL pins
12
13 /*
14  * note:If lcd1602 uses PCF8574T, IIC's address is 0x27,
15  *       or lcd1602 uses PCF8574AT, IIC's address is 0x3F.
16  */
17 LiquidCrystal_I2C lcd(0x27,16,2);
18 void setup() {
19   Wire.begin(SDA, SCL);           // attach the IIC pin
20   lcd.init();                     // LCD driver initialization
21   lcd.backlight();               // Open the backlight
22   lcd.setCursor(0,0);            // Move the cursor to row 0, column 0
23   lcd.print("hello, world!");    // The print content is displayed on the LCD
24 }
25
26 void loop() {
27   lcd.setCursor(0,1);            // Move the cursor to row 1, column 0
28   lcd.print("Counter:");         // The count is displayed every second
29   lcd.print(millis() / 1000);
30   delay(1000);
31 }

```

Done uploading.

Leaving...
Hard resetting via RTS pin...

11 ESP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 115200, None on COM4

Compile and upload the code to ESP32-WROVER and the LCD1602 displays characters.



If you cannot see anything on the display or the display is not clear, try rotating the white knob on back of LCD1602 slowly, which adjusts the contrast, until the screen can display clearly.



The following is the program code:

```

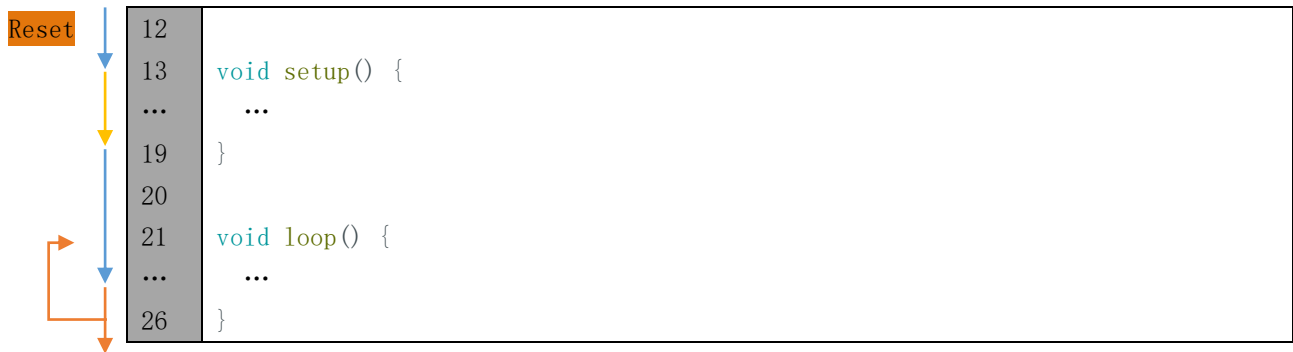
1  #include <LiquidCrystal_I2C.h>
2  #include <Wire.h>
3
4  #define SDA 13           //Define SDA pins
5  #define SCL 14          //Define SCL pins
6
7  /*
8   * note: If lcd1602 uses PCF8574T, IIC's address is 0x27,
9   *       or lcd1602 uses PCF8574AT, IIC's address is 0x3F.
10  */
11  LiquidCrystal_I2C lcd(0x27, 16, 2);
12
13  void setup() {
14      Wire.begin(SDA, SCL);           // attach the IIC pin
15      lcd.init();                     // LCD driver initialization
16      lcd.backlight();                // Turn on the backlight
17      lcd.setCursor(0, 0);            // Move the cursor to row 0, column 0
18      lcd.print("hello, world! ");    // The print content is displayed on the LCD
19  }
20
21  void loop() {
22      lcd.setCursor(0, 1);            // Move the cursor to row 1, column 0
23      lcd.print("Counter:");          // The count is displayed every second
24      lcd.print(millis() / 1000);
25      delay(1000);
26  }

```

The Arduino IDE code usually contains two basic functions: void setup() and void loop().

After the board is reset, the setup() function will be executed firstly, and then the loop() function.

setup() function is generally used to write code to initialize the hardware. And loop() function is used to write code to achieve certain functions. loop() function is executed repeatedly. When the execution reaches the end of loop(), it will jump to the beginning of loop() to run again.



Reset

Reset operation will lead the code to be executed from the beginning. Switching on the power, finishing uploading the code and pressing the reset button will trigger reset operation.

Include header file of Liquid Crystal Display (LCD)1602 and I2C.

```

1 #include <LiquidCrystal_I2C.h>
2 #include <Wire.h>

```

Instantiate the I2C LCD1602 screen. It should be noted here that if your LCD driver chip uses PCF8574T, set the I2C address to 0x27, and if uses PCF8574AT, set the I2C address to 0x3F.

```

11 LiquidCrystal_I2C lcd(0x27, 16, 2);

```

Initialize I2C and set its pins as 13,14. And then initialize LCD1602 and turn on the backlight of LCD.

```

14 Wire.begin(SDA, SCL);           // attach the IIC pin
15 lcd.init();                     // LCD driver initialization
16 lcd.backlight();                // Turn on the backlight

```

Move the cursor to the first row, first column, and then display the character.

```

17 lcd.setCursor(0, 0);            // Move the cursor to row 0, column 0
18 lcd.print("hello, world! ");    // The print content is displayed on the LCD

```

Print the number on the second line of LCD1602.

```

21 void loop() {
22   lcd.setCursor(0, 1);           // Move the cursor to row 1, column 0
23   lcd.print("Counter:");         // The count is displayed every second
24   lcd.print(millis() / 1000);
25   delay(1000);
26 }

```

Reference

class LiquidCrystal

The LiquidCrystal class can manipulate common LCD screens. The first step is defining an object of LiquidCrystal, for example:

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

Instantiate the Lcd1602 and set the I2C address to 0x27, with 16 columns per row and 2 rows per column.

init();

Initializes the Lcd1602's device

backlight();

Turn on Lcd1602's backlight.

setCursor(column, row);

Sets the screen's column and row.

column: The range is 0 to 15.

row: The range is 0 to 1.

print(String);

Print the character string on Lcd1602

Chapter 2 LCD2004

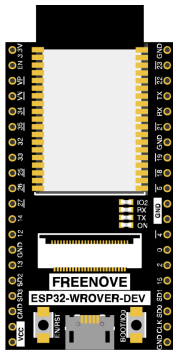
In the previous chapter, we studied the LCD1602 display. In order to display more content, In this chapter, we will learn about the LCD2004 Display Screen.

Project 2.1 LCD2004

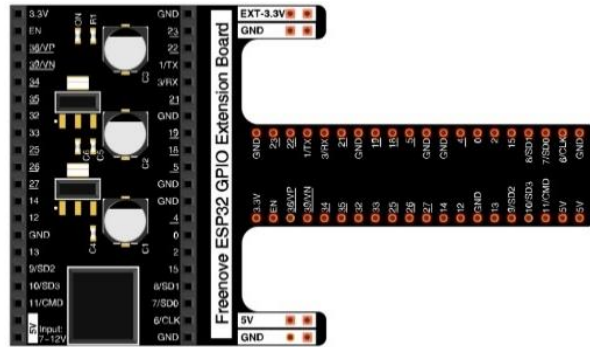
In this section we learn how to use LCD2004 to display something.

Component List

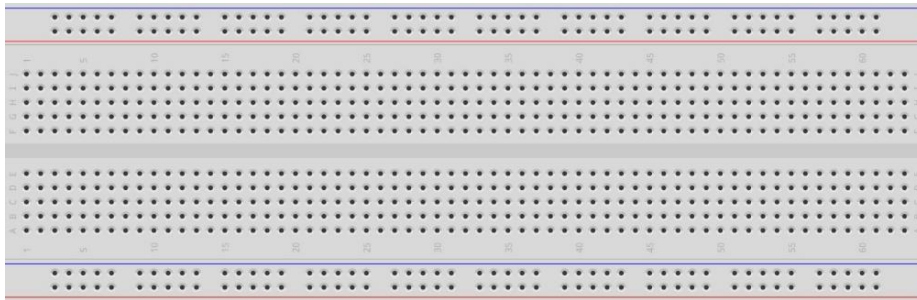
ESP32-WROVER x1



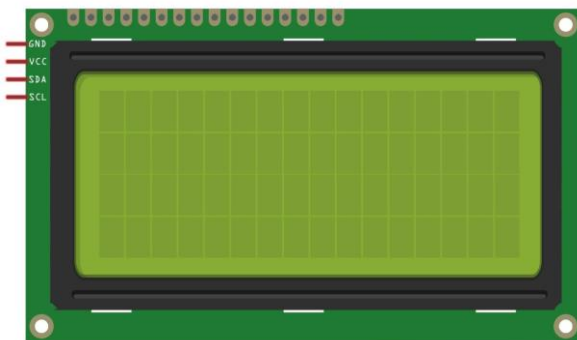
GPIO Extension Board x1



Breadboard x1



LCD2004 Module x1



Jumper F/M x4



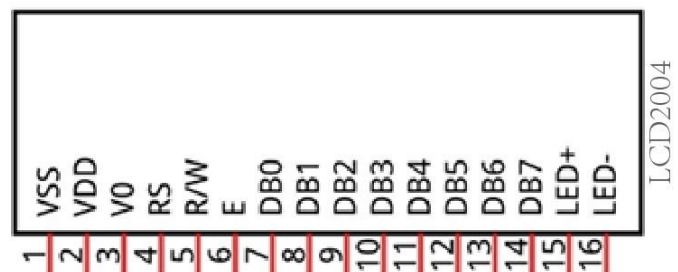
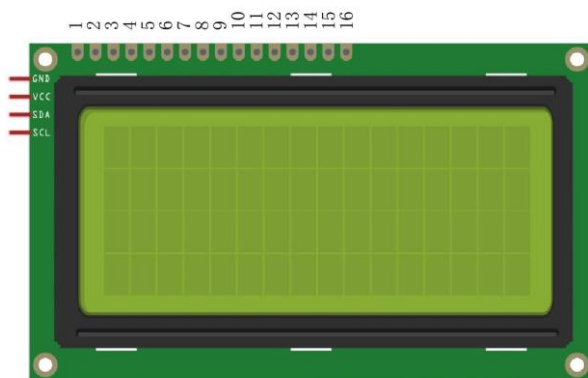
Component knowledge

I2C communication

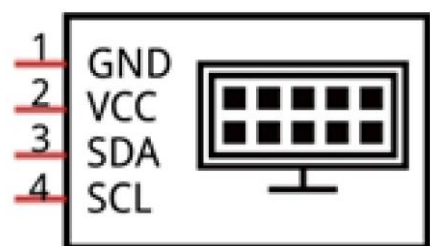
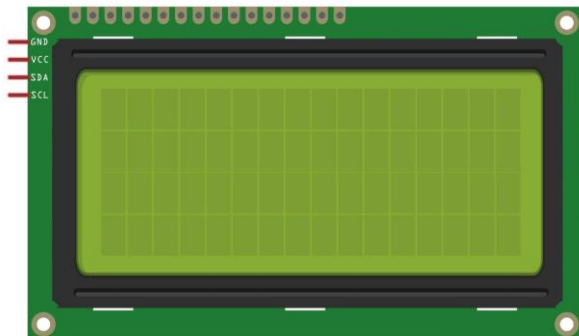
I2C (Inter-Integrated Circuit) is a two-wire serial communication mode, which can be used for the connection of micro controllers and their peripheral equipment. Devices using I2C communication must be connected to the serial data (SDA) line, and serial clock (SCL) line (called I2C bus). Each device has a unique address and can be used as a transmitter or receiver to communicate with devices connected to the bus.

LCD2004 communication

The LCD2004 display screen can display 4 lines of characters in 20 columns. It is capable of displaying numbers, letters, symbols, ASCII code and so on. As shown below is a monochrome LCD2004 display screen along with its circuit pin diagram.



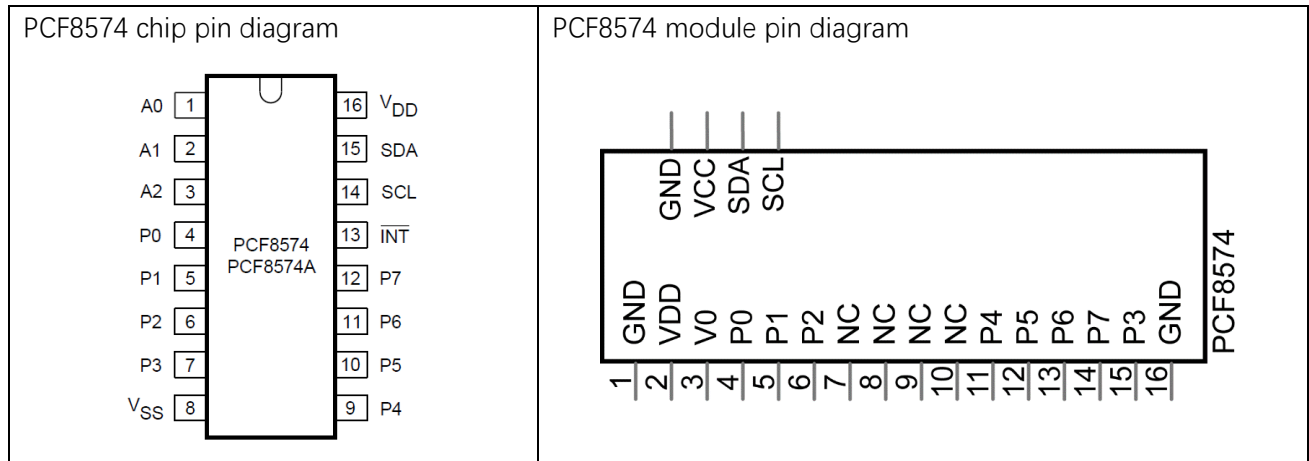
I2C LCD2004 display screen integrates a I2C interface, which connects the serial-input & parallel-output module to the LCD2004 display screen. This allows us to only use 4 lines to the operate the LCD2004.



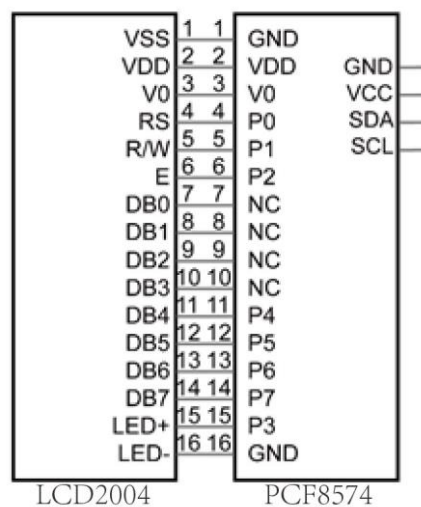
I2C LCD2004 Module

The serial-to-parallel IC chip used in this module is PCF8574T (PCF8574AT), and its default I2C address is 0x27(0x3F).

Below is the PCF8574 pin schematic diagram and the block pin diagram:



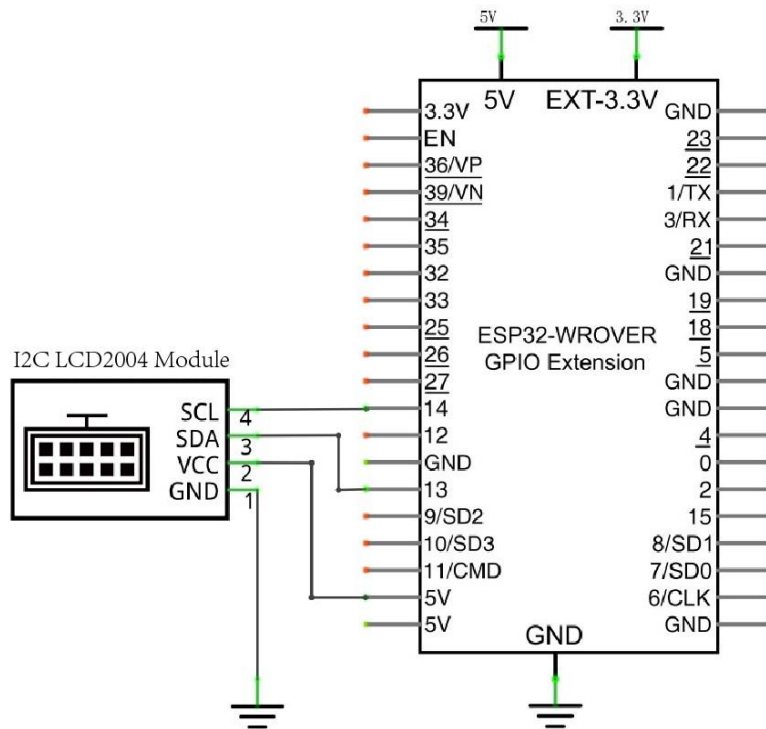
PCF8574 module pin and LCD2004 pin are corresponding to each other and connected with each other:



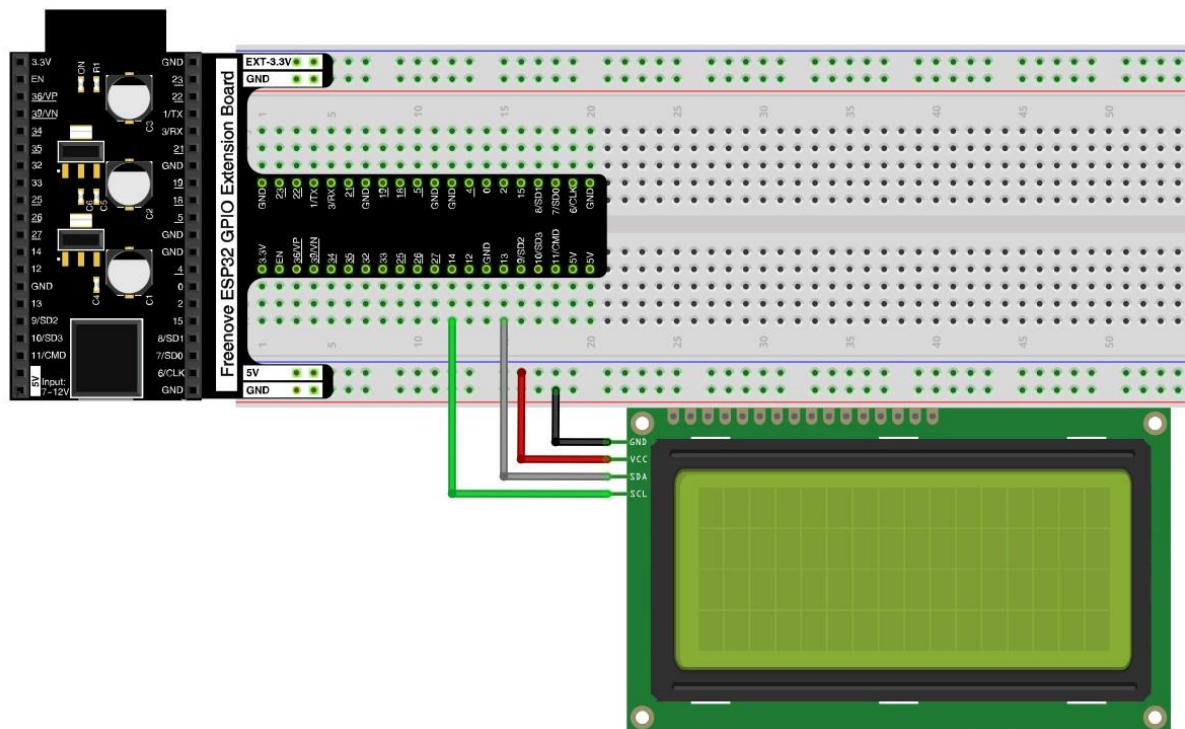
So we only need 4 pins to control the 16 pins of the LCD2004 display screen through the I2C interface. In this project, we will use the I2C LCD2004 to display some static characters and dynamic variables.

Circuit

Schematic diagram



Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com

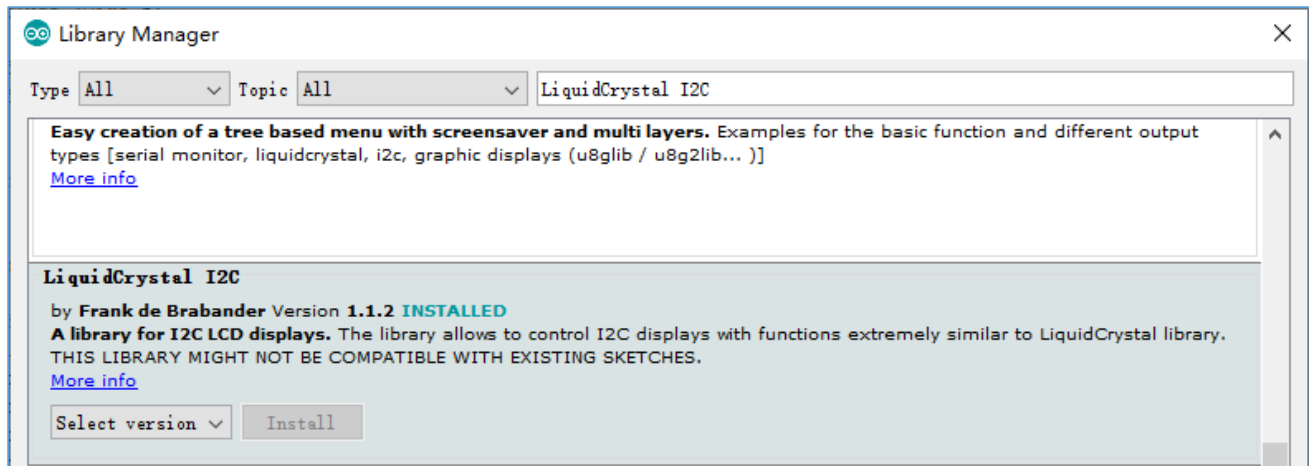


Sketch

Before writing code, we need to import the library needed. Skip this section if you have already installed it, or proceed if you haven't.

How to install the library

We use the third party library LiquidCrystal I2C. If you haven't installed it yet, please do so before learning. The steps to add third-party Libraries are as follows: open arduino->Sketch->Include library-> Manage libraries. Enter " LiquidCrystal I2C" in the search bar and select " LiquidCrystal I2C " for installation.



There is another way you can install libraries.

Click "Add .ZIP Library..." and then find **LiquidCrystal_I2C.zip** in libraries folder (this folder is in the folder unzipped from the ZIP file we provided). This library can facilitate our operation of I2C LCD2004.

Use I2C LCD 2004 to display characters and variables.

Sketch_2.1_Display_the_string_on_LCD2004

```
Sketch_2.1_Display_the_string_on_LCD2004 | Arduino 1.8.19
File Edit Sketch Tools Help

Sketch_2.1_Display_the_string_on_LCD2004
17 LiquidCrystal_I2C lcd(0x27,20,4);
18 void setup() {
19   Wire.begin(SDA, SCL);           // attach the IIC pin
20   lcd.init();                     // LCD driver initialization
21   lcd.backlight();                // Open the backlight
22   // (note: line 1 is the second row, since counting begins with 0):
23   lcd.setCursor(0, 0); // set the cursor to column 0, line 0
24   // print the number of seconds since reset:
25   lcd.print("FREENOVE");
26   lcd.setCursor(0, 1); // set the cursor to column 0, line 1
27   // print the number of seconds since reset:
28   lcd.print("www.freenove.com");
29   lcd.setCursor(0, 2); // set the cursor to column 0, line 2
30   lcd.print("hello, world!"); // Print a message to the LCD
31 }
32
33 void loop() {
34   lcd.setCursor(0,3);             // Move the cursor to row 1, column 0
35   lcd.print("Counter:");          // The count is displayed every second
36   lcd.print(millis() / 1000);
37   delay(1000);
38 }
```

Done uploading.

Leaving...
Hard resetting via RTS pin...

14 ESP32 Wrover Module, Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 115200, None on COM4

Compile and upload the code to ESP32-WROVER and the LCD2004 displays characters.



Any concerns? ✉ support@freenove.com

Note: If you cannot see anything on the display or the display is not clear, try rotating the white knob on back of LCD2004 slowly, which adjusts the contrast, until the screen can display clearly.



The following is the program code:

```

1  #include <LiquidCrystal_I2C.h>
2  #include <Wire.h>
3
4  #define SDA 13                //Define SDA pins
5  #define SCL 14               //Define SCL pins
6
7  /*
8   * note: If lcd2004 uses PCF8574T, IIC's address is 0x27,
9   *       or lcd2004 uses PCF8574AT, IIC's address is 0x3F.
10  */
11  LiquidCrystal_I2C lcd(0x27, 20, 4);
12
13  void setup() {
14      Wire.begin(SDA, SCL);      // attach the IIC pin
15      lcd.init();               // LCD driver initialization
16      lcd.backlight();          // Turn on the backlight
17      // (note: line 1 is the second row, since counting begins with 0):
18      lcd.setCursor(0, 0); // set the cursor to column 0, line 0
19      // print the number of seconds since reset:
20      lcd.print("FREENOVE");
21      lcd.setCursor(0, 1); // set the cursor to column 0, line 1
22      // print the number of seconds since reset:
23      lcd.print("www.freenove.com");
24      lcd.setCursor(0, 2); // set the cursor to column 0, line 2
25      lcd.print("hello, world!"); // Print a message to the LCD
26
27  }
28
29  void loop() {
30      lcd.setCursor(0, 3);      // Move the cursor to column 0, row 3
31      lcd.print("Counter:");    // The count is displayed every second
32      lcd.print(millis() / 1000);

```

```

33     delay(1000);
34 }

```

Include header file of Liquid Crystal Display (LCD)2004 and I2C.

```

1  #include <LiquidCrystal_I2C.h>
2  #include <Wire.h>

```

Instantiate the I2C LCD2004 screen. It should be noted here that if your LCD driver chip uses PCF8574T, set the I2C address to 0x27, and if uses PCF8574AT, set the I2C address to 0x3F.

```

11 LiquidCrystal_I2C lcd(0x27, 20, 4);

```

Initialize I2C and set its pins as 13,14. And then initialize LCD2004 and turn on the backlight of LCD.

```

14 Wire.begin(SDA, SCL);           // attach the IIC pin
15 lcd.init();                     // LCD driver initialization
16 lcd.backlight();                // Turn on the backlight

```

Move the cursor to the third row, first column, and then display the character.

```

24 lcd.setCursor(0, 2);            // Move the cursor to row 2, column 0
25 lcd.print("hello, world! ");    // The print content is displayed on the LCD

```

Print the number on the fourth line of LCD2004.

```

29 void loop() {
30     lcd.setCursor(0, 3);         // Move the cursor to column 0, row 3
31     lcd.print("Counter:");       // The count is displayed every second
32     lcd.print(millis() / 1000);
33     delay(1000);
34 }

```

Reference

class LiquidCrystal

The LiquidCrystal class can manipulate common LCD screens. The first step is defining an object of LiquidCrystal, for example:

```
LiquidCrystal_I2C lcd(0x27, 20, 4);
```

Instantiate the Lcd2004 and set the I2C address to 0x27, with 20 columns per row and 4 rows per column.

```
init();
```

Initializes the Lcd2004's device

```
backlight();
```

Turn on Lcd2004's backlight.

```
setCursor(column, row);
```

Sets the screen's column and row.

column: The range is 0 to 19.

row: The range is 0 to 3.

```
print(String);
```

Print the character string on Lcd2004.

What's next?

Thanks for your reading. This tutorial is all over here. If you find any mistakes, omissions or you have other ideas and questions about contents of this tutorial or the kit and etc., please feel free to contact us:

support@freenove.com

We will check and correct it as soon as possible.

If you want to learn more about Arduino, Raspberry Pi, smart cars, robots and other interesting products in science and technology, please continue to focus on our website. We will continue to launch cost-effective, innovative and exciting products.

<http://www.freenove.com/>

What's next?(others)

Thanks for your reading. This tutorial is all over here. If you find any mistakes, omissions or you have other ideas and questions about contents of this tutorial or the kit and etc., please feel free to contact us:

support@freenove.com

We will check and correct it as soon as possible.

If you want to learn more about Arduino, Raspberry Pi, smart cars, robots and other interesting products in science and technology, please continue to focus on our website. We will continue to launch cost-effective, innovative and exciting products.

<http://www.freenove.com/>

End of the Tutorial

Thank you again for choosing Freenove products.