# 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.

- 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

# Contents

# Preface

ESP8266 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 Micyoco(SCM) chip, or connected to the Internet and used as an Internet of Things device.

ESP8266 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 ESP8266 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, ESP8266 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.

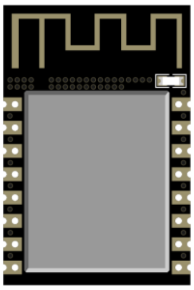
We divide each project into four parts, namely Component List, Component Knowledge, Circuit and Code. Component List helps you to prepare material for the experiment more quickly. Component Knowledge allows you to quickly understand new electronic modules or components, while Circuit helps you understand the operating principle of the circuit. And Code allows you to easily master the use of ESP8266 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**

# ESP8266

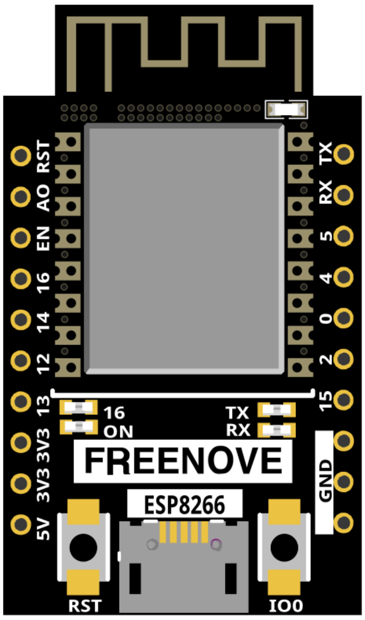
ESP8266 has PCB on-board antenna. The PCB on-board antenna is an integrated antenna in the chip module itself, so it is convenient to carry and design.

PCB on-board antenna

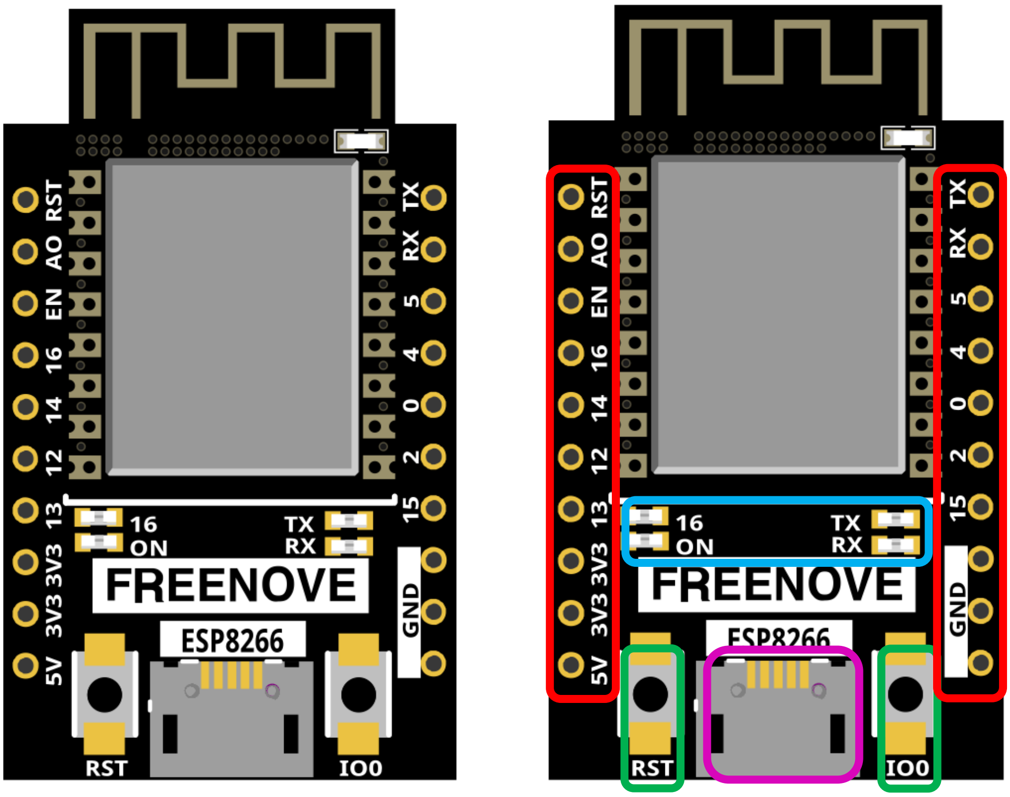


In this tutorial, the ESP8266 development board is designed based on the PCB on-board antenna-packaged ESP8266 module. The following tutorials will be based on the ESP8266 development board.

ESP8266 development board



The hardware interfaces of ESP8266 are distributed as follows:



Compare the left and right images. We've boxed off the resources on the ESP8266 in different colors to facilitate your understanding of the ESP8266 development board.

| Box color | Corresponding resources introduction |
|---|---|
| 🟥 | **GPIO pin** |
| 🟦 | **LED indicator** |
| 🟩 | **Reset button, Boot mode selection button** |
| 🟪 | **USB port** |

| NO. | Pin Name | Functional Description |
|---|---|---|
| 1 | RST | Reset Pin, Active Low |
| 2 | ADC | AD conversion, Input voltage range 0~1V, the value range is 0~1024. |
| 3 | EN | Chip Enabled Pin, Active High |
| 4 | IO16 | Connect with RST pin to wake up Deep Slee |
| 5 | IO14 | GPIO14; HSPI_CLK |
| 6 | IO12 | GPIO12; HSPI_MISO |
| 7 | IO13 | GPIO13; HSPI_MOSI; UART0_CTS |
| 8 | VCC | Module power supply pin, Voltage 3.0V ~ 3.6V |
| 9 | GND | GND |
| 10 | IO15 | GPIO15; MTDO; HSPICS; UART0 |
| 11 | IO2 | GPIO2; UART1_TXD |
| 12 | IO0 | GPIO2; UART1_TXD |
| 13 | IO4 | GPIO4 |
| 14 | IO5 | GPIO5;IR_R |
| 15 | RXD | UART0_RXD; GPIO3 |
| 16 | TXD | UART0_TXD; GPIO1 |

Description of the ESP8266 series module boot mode:

| Mode | CH_PD(EN) | RST | GPIO15 | GPIO0 | GPIO2 | TXD0 |
|---|---|---|---|---|---|---|
| Download mode | high | high | low | low | high | high |
| Running mode | high | high | low | high | high | high |

Notes: Some of the pins inside the module have been pulled or pulled down.

For more information, please visit: https://docs.ai-thinker.com/_media/esp8266/docs/esp-12s_product_specification_en.pdf

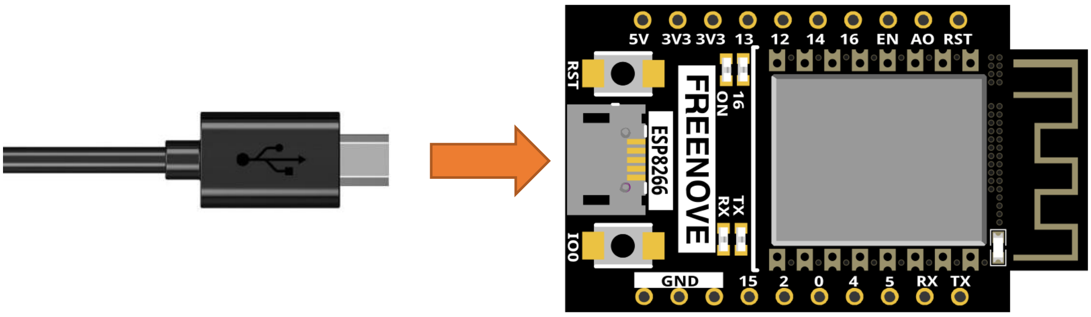
# CH340

ESP8266 uses CH340 to download codes. So before using it, we need to install CH340 driver in our computers.

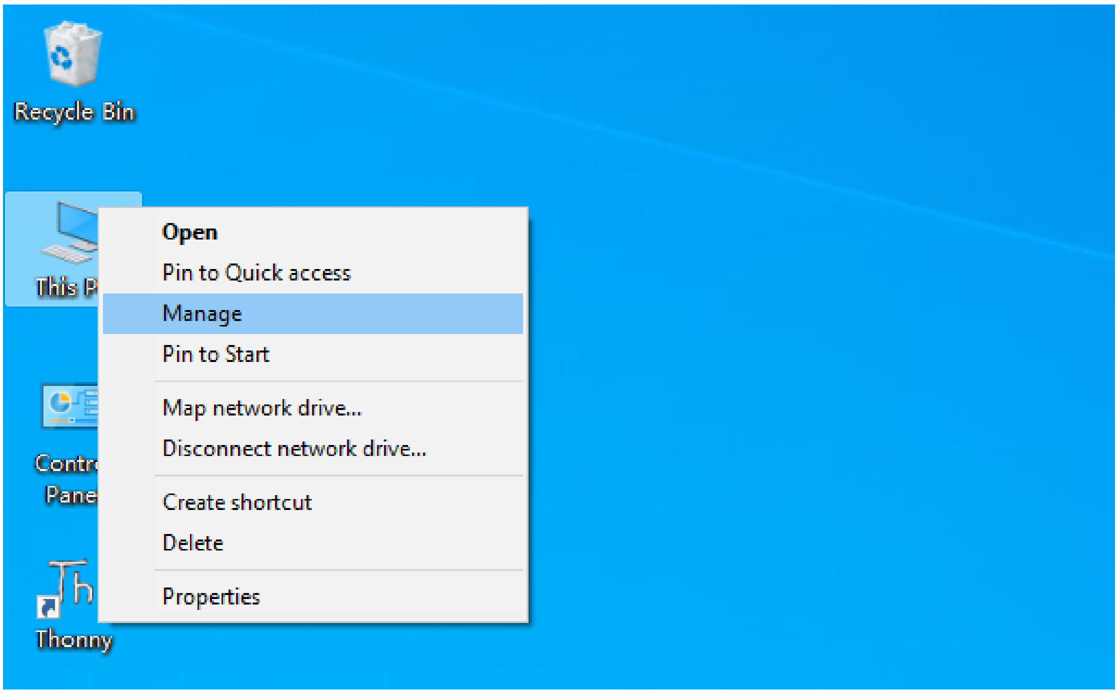
## Windows

### Check whether CH340 has been installed

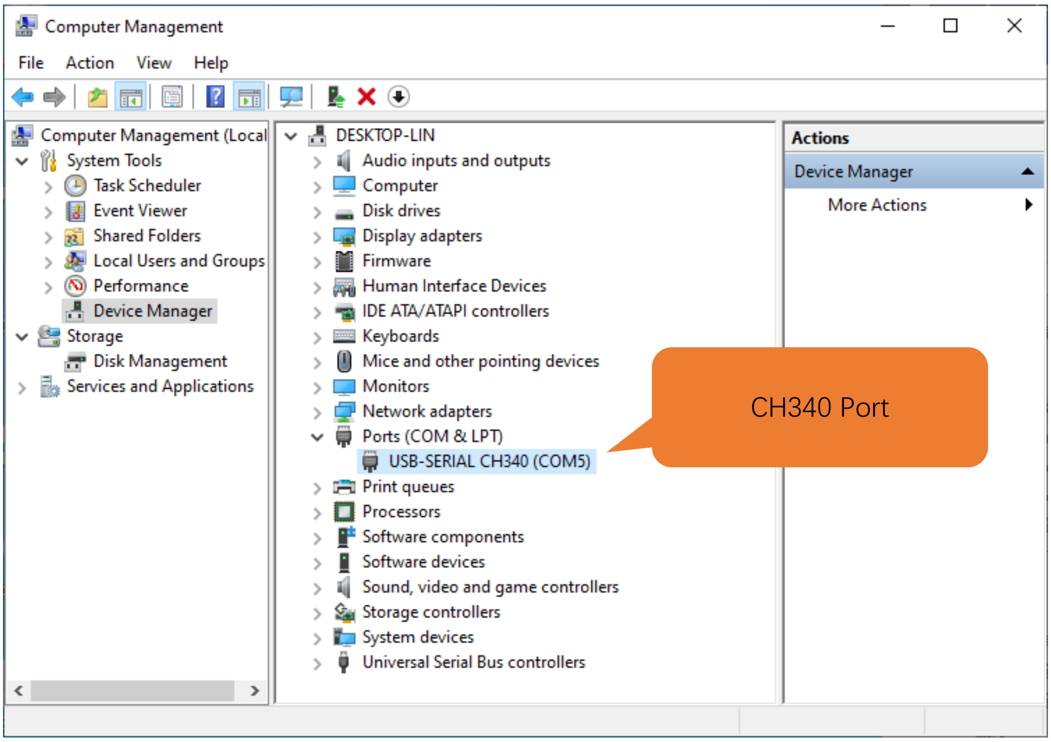
1. Connect your computer and ESP8266 with a USB cable.



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



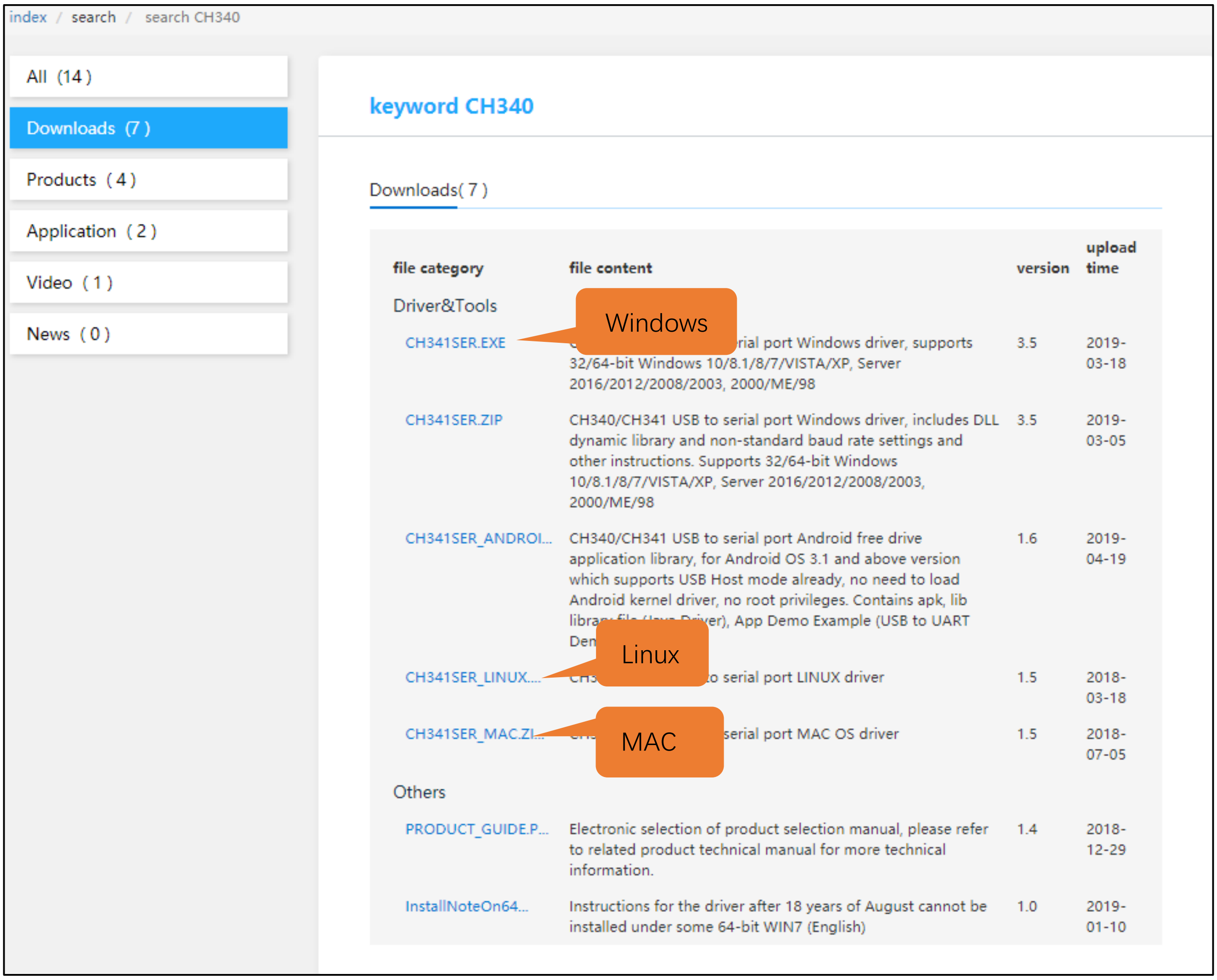
3. Click “Device Manager”. If your computer has installed CH340, you can see“USB-SERIAL CH340 (COMx)”. And you can click here to move to the next step.



CH340 Port

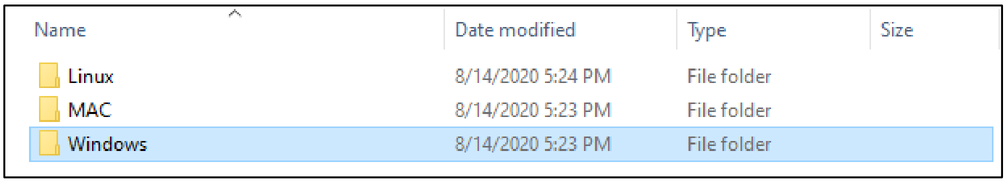
## Installing CH340

1. 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.



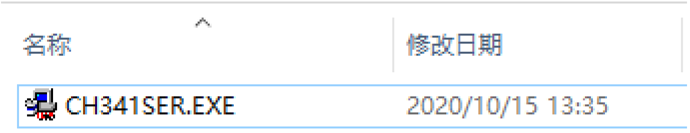
| file category | file content | version | upload time |
|---|---|---|---|
| Driver&Tools | | | |
| CH341SER.EXE | C rial port Windows driver, supports 32/64-bit Windows 10/8.1/8/7/VISTA/XP, Server 2016/2012/2008/2003, 2000/ME/98 | 3.5 | 2019-03-18 |
| CH341SER.ZIP | CH340/CH341 USB to serial port Windows 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_ANDROI... | 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 (USB to UART Dem | 1.6 | 2019-04-19 |
| CH341SER_LINUX.... | CH3 to serial port LINUX driver | 1.5 | 2018-03-18 |
| CH341SER_MAC.ZI... | CH 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.



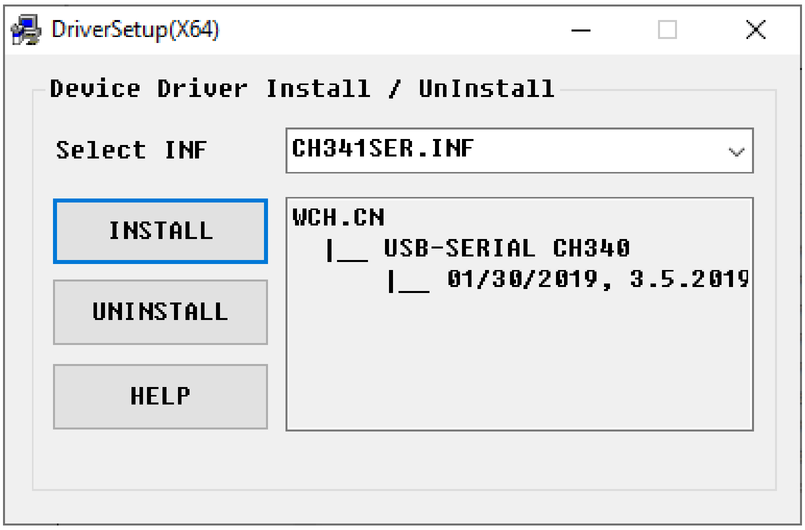
| Name | Date modified | Type | Size |
|---|---|---|---|
| Linux | 8/14/2020 5:24 PM | File folder | |
| MAC | 8/14/2020 5:23 PM | File folder | |
| Windows | 8/14/2020 5:23 PM | File folder | |

2. Open the folder “**Freenove_LCD_Module/CH340/Windows/**”



CH341SER.EXE 2020/10/15 13:35

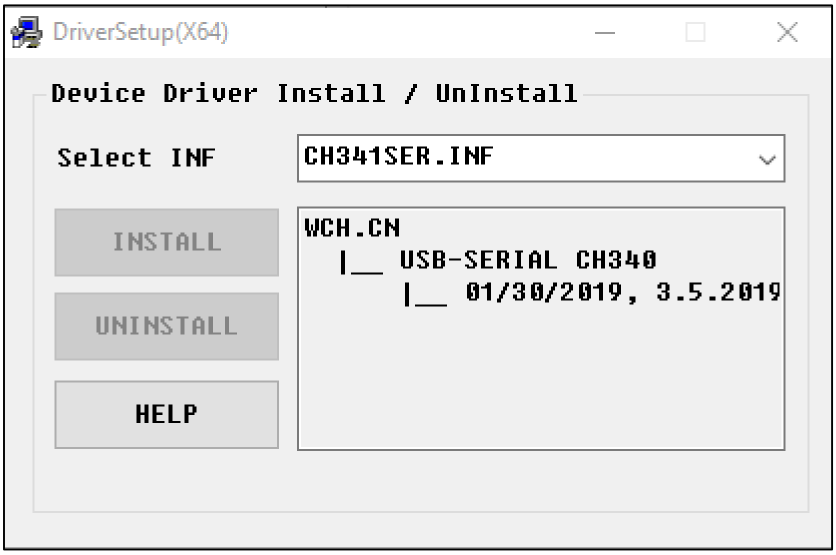
3. Double click “**CH341SER.EXE**”.



Select INF CH341SER.INF

```
WCH.CN
   |__ USB-SERIAL CH340
        |__ 01/30/2019, 3.5.2019
```

4. Click “INSTALL” and wait for the installation to complete.



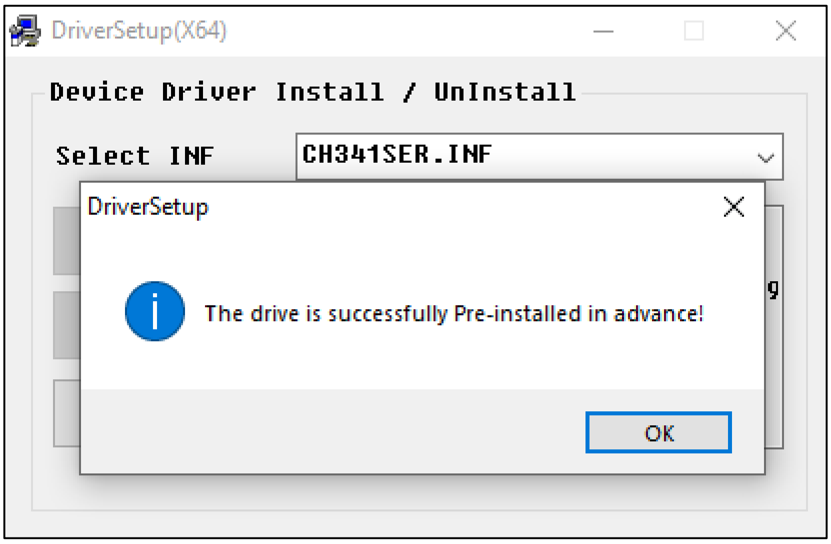
DriverSetup(X64)

Device Driver Install / UnInstall

Select INF: CH341SER.INF

```
WCH.CN
  |__ USB-SERIAL CH340
      |__ 01/30/2019, 3.5.2019
```

5. Install successfully. Close all interfaces.



DriverSetup(X64)

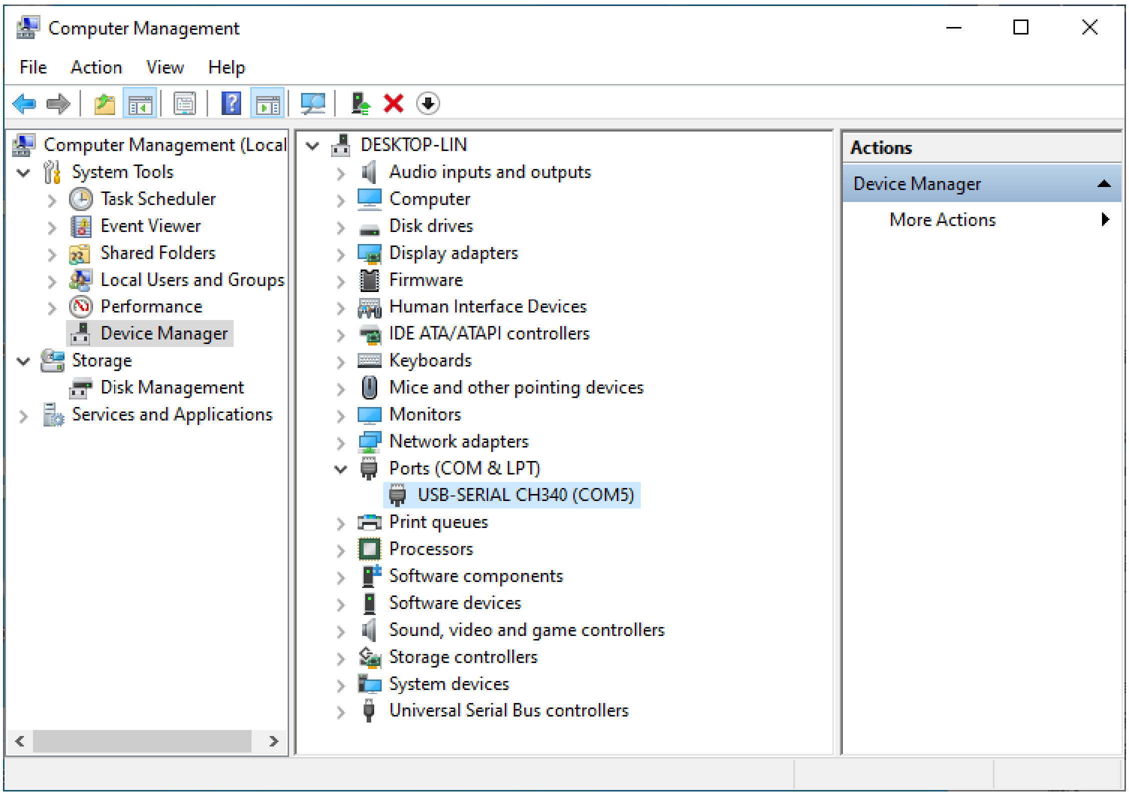
Device Driver Install / UnInstall

Select INF: CH341SER.INF

DriverSetup

The drive is successfully Pre-installed in advance!

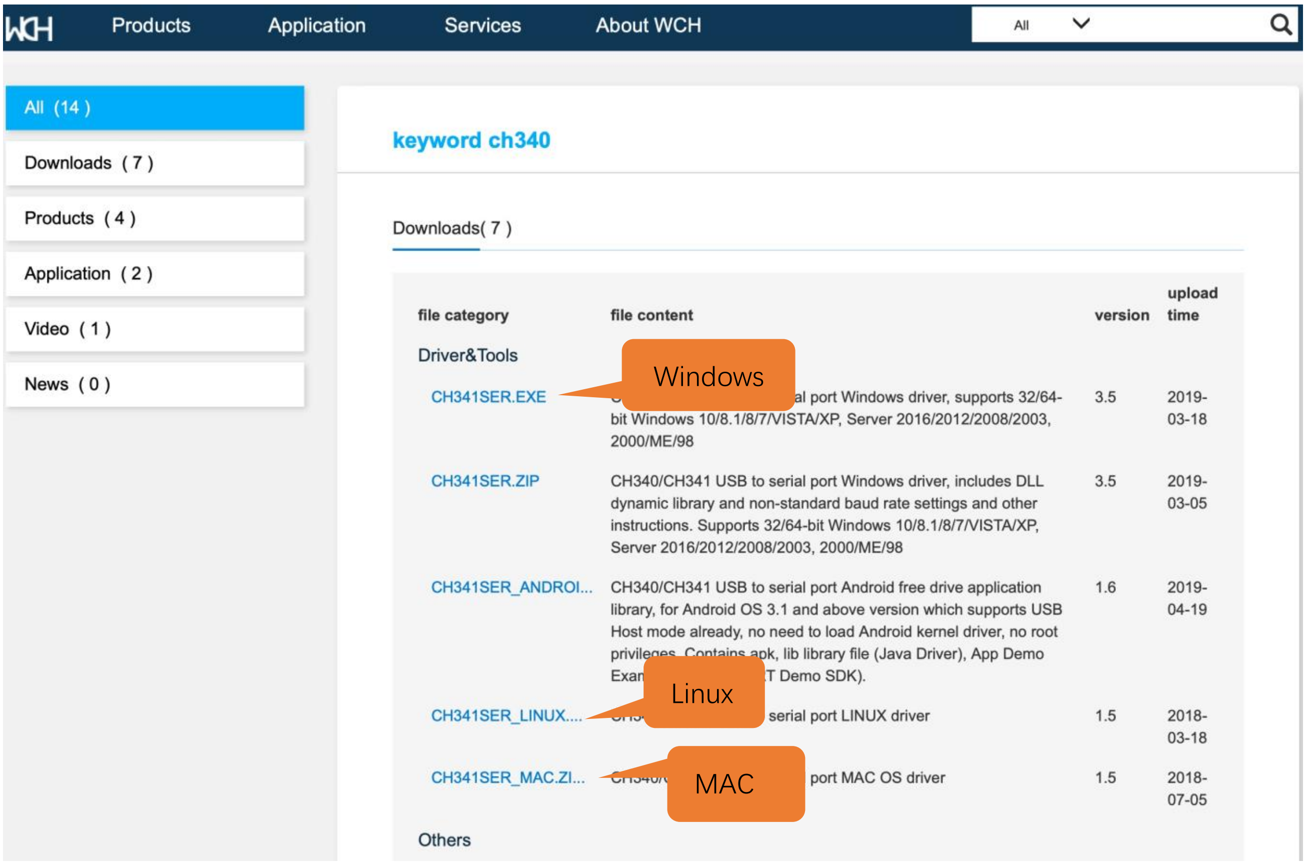
6. When ESP8266 is connected to computer, select “This PC”, right-click to select “Manage” and click “Device Manager” in the newly pop-up dialog box, and you can see the following interface.



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.



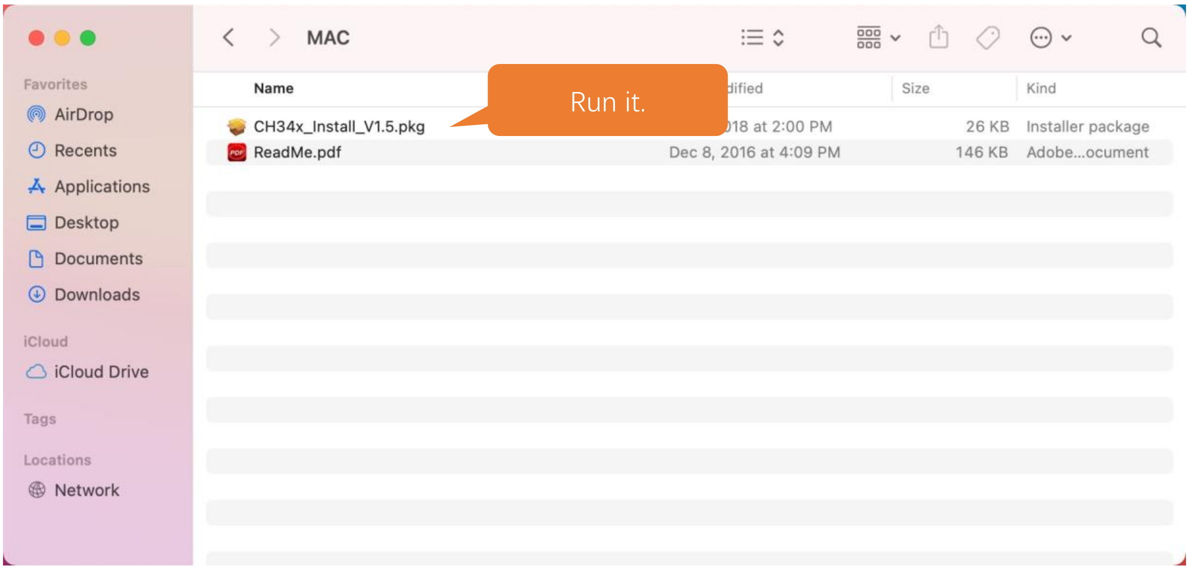
keyword ch340

Downloads( 7 )

| file category | file content | version | upload time |
|---|---|---|---|
| Driver&Tools | | | |
| CH341SER.EXE | C al 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_ANDROI... | 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 Exan T Demo SDK). | 1.6 | 2019-04-19 |
| CH341SER_LINUX.... | CH3 serial port LINUX driver | 1.5 | 2018-03-18 |
| CH341SER_MAC.ZI... | CH340/ port MAC OS driver | 1.5 | 2018-07-05 |
| Others | | | |

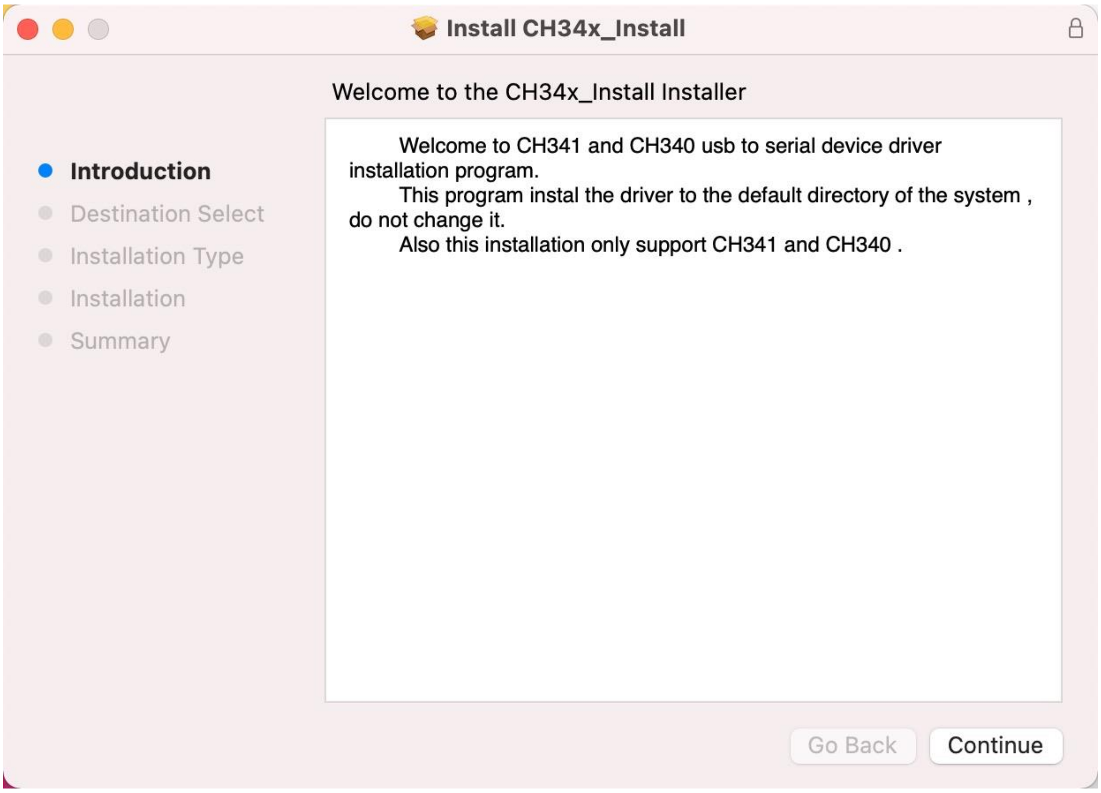
If you would not like to download the installation package, you can open **"Freenove_LCD_Module/CH340"**, we have prepared the installation package.

Second, open the folder "**Freenove_LCD_Module/CH340/MAC/**"



| Name | dified | Size | Kind |
|---|---|---|---|
| CH34x_Install_V1.5.pkg | 018 at 2:00 PM | 26 KB | Installer package |
| ReadMe.pdf | Dec 8, 2016 at 4:09 PM | 146 KB | Adobe...ocument |

Third, click Continue.



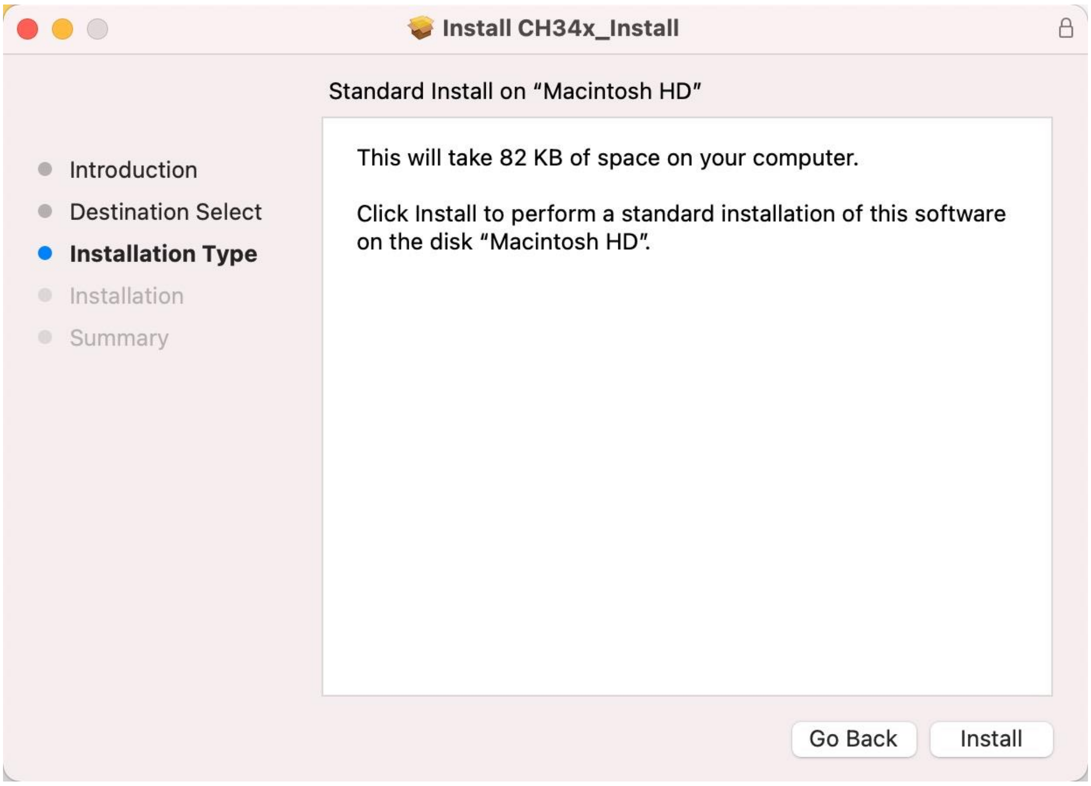
Welcome to the CH34x_Install Installer

Welcome to CH341 and CH340 usb to serial device driver installation program.

This program instal the driver to the default directory of the system , do not change it.

Also this installation only support CH341 and CH340 .

Fourth, click Install.

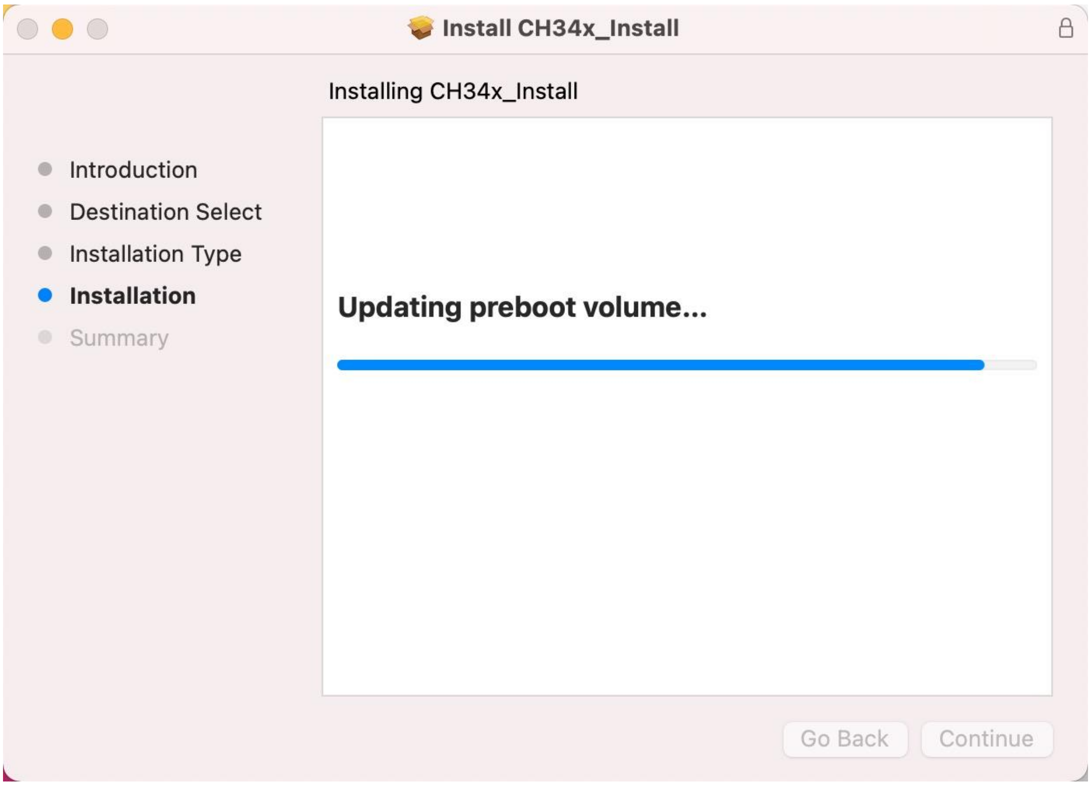


Standard Install on “Macintosh HD”

This will take 82 KB of space on your computer.

Click Install to perform a standard installation of this software on the disk “Macintosh HD”.

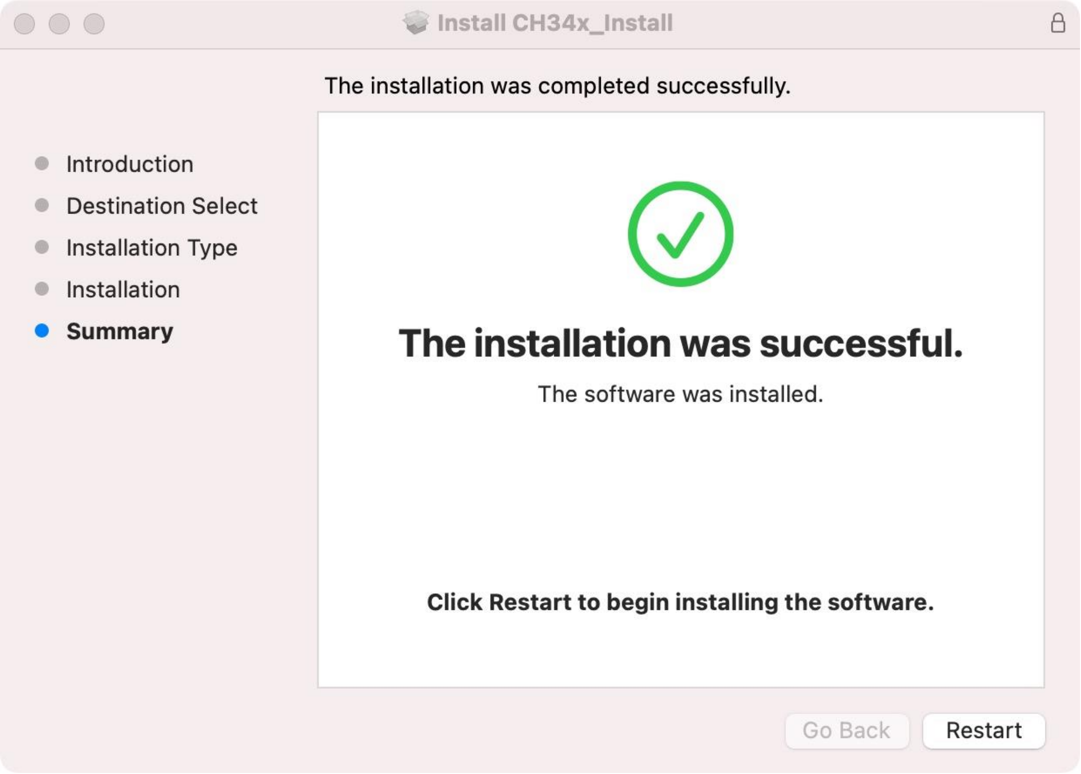
Then, waiting Finsh.



Installing CH34x_Install

Updating preboot volume...

Finally, restart your PC.



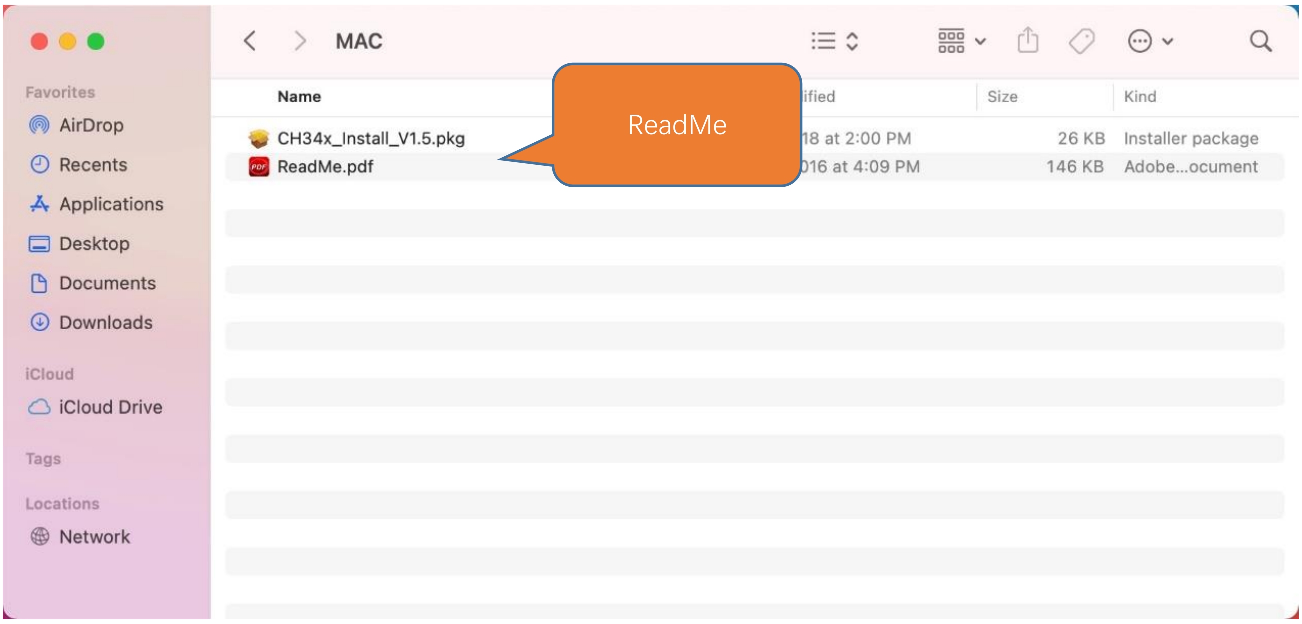
The installation was completed successfully.

The installation was successful.

The software was installed.

Click Restart to begin installing the software.

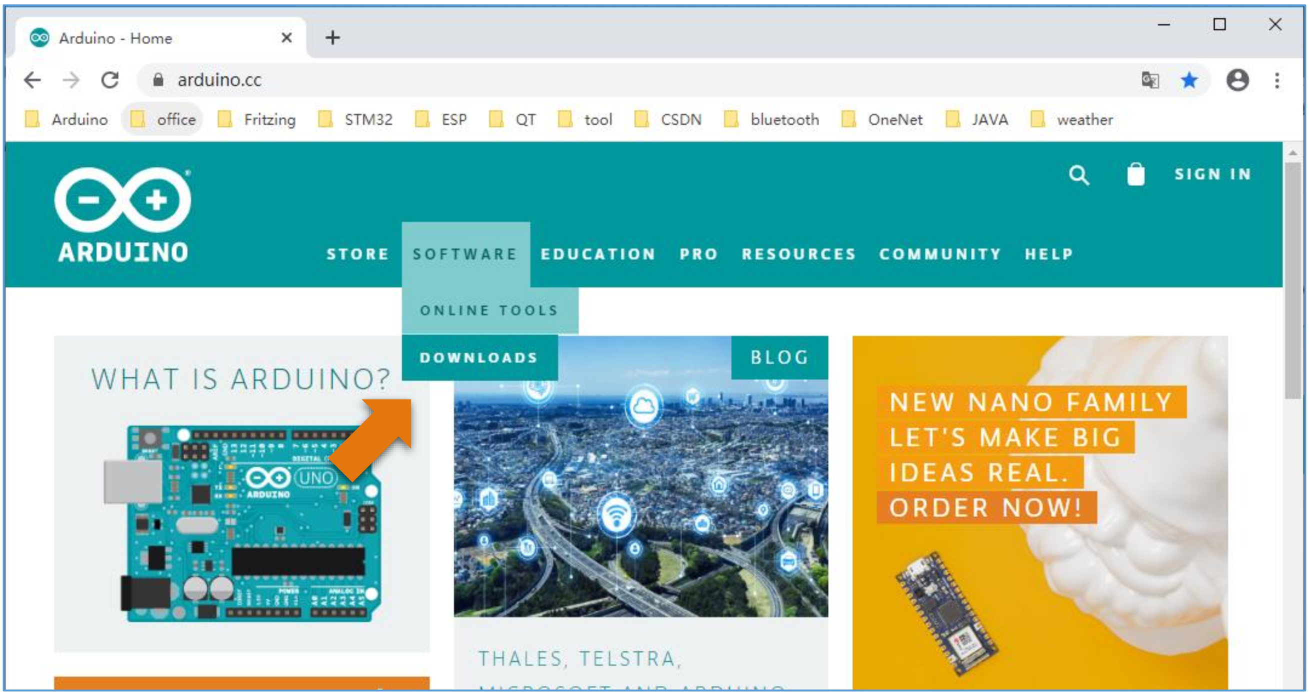
If you still haven't installed the CH340 by following the steps above, you can view readme.pdf to install it.



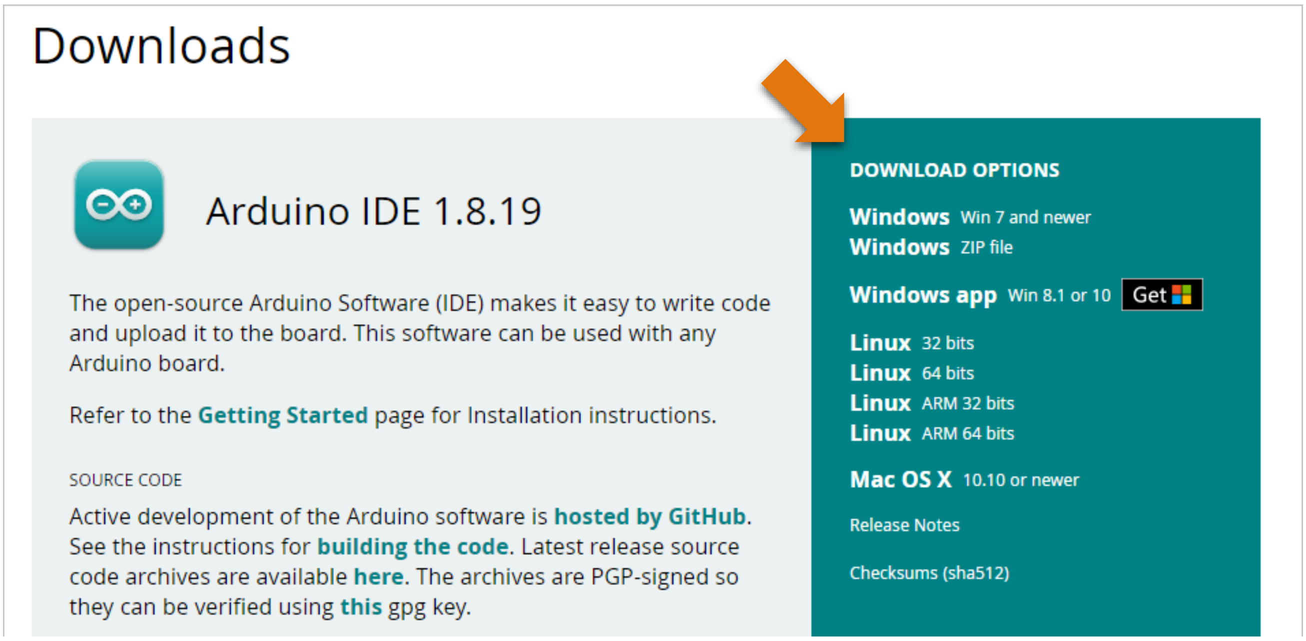
| Name | …ified | Size | Kind |
|---|---|---|---|
| CH34x_Install_V1.5.pkg | …18 at 2:00 PM | 26 KB | Installer package |
| ReadMe.pdf | …016 at 4:09 PM | 146 KB | Adobe...ocument |

## 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.



# Downloads

## Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the **Getting Started** page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is **hosted by GitHub**. See the instructions for **building the code**. Latest release source code archives are available **here**. The archives are PGP-signed so they can be verified using **this** gpg key.

DOWNLOAD OPTIONS

**Windows** Win 7 and newer
**Windows** ZIP file

**Windows app** Win 8.1 or 10 Get

**Linux** 32 bits
**Linux** 64 bits
**Linux** ARM 32 bits
**Linux** ARM 64 bits

**Mac OS X** 10.10 or newer

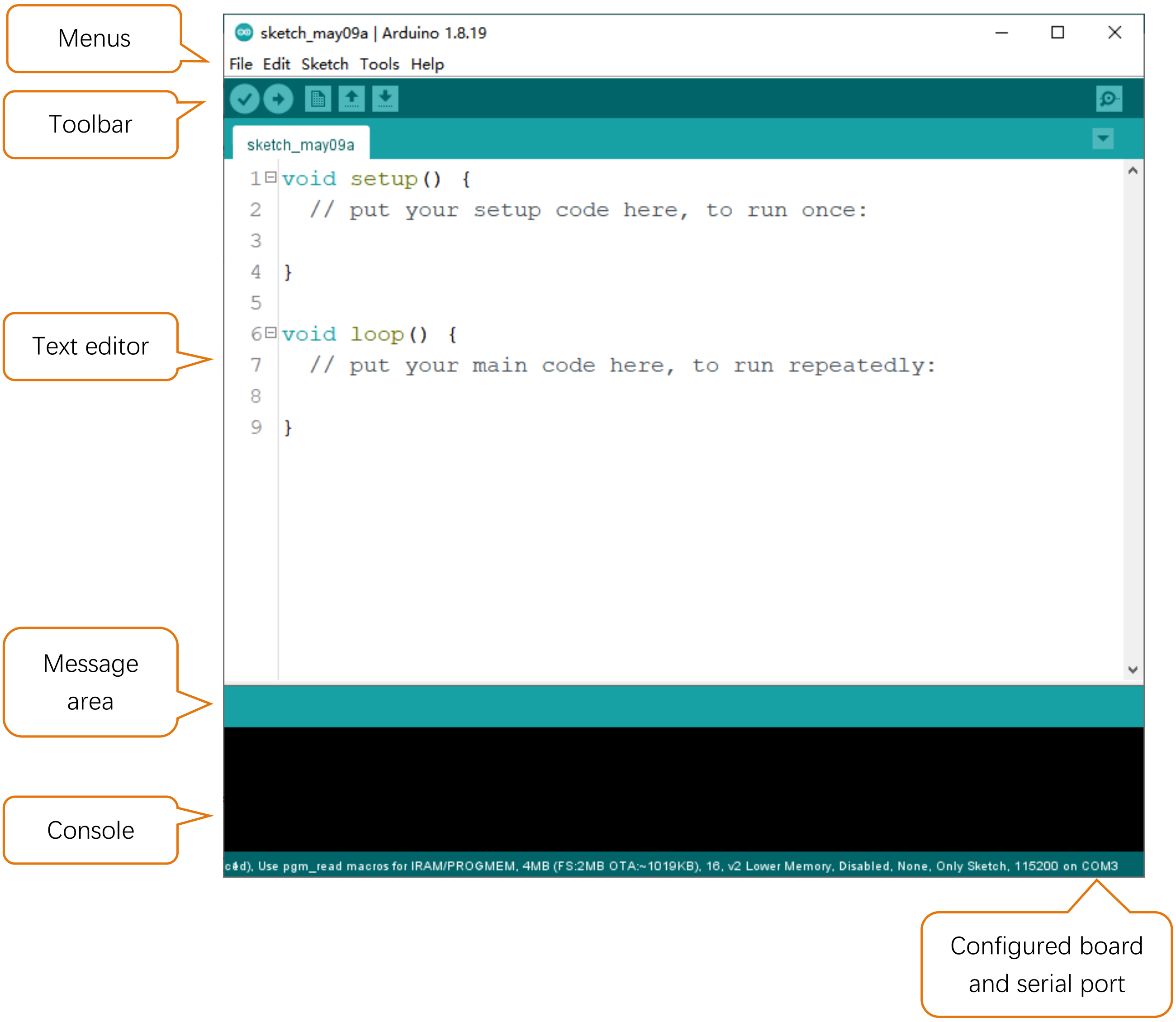
Release Notes

Checksums (sha512)

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.

Arduino

The interface of Arduino Software is as follows:


Menus
Toolbar
Text editor
Message area
Console
Configured board and serial port
sketch_may09a | Arduino 1.8.19
File Edit Sketch Tools Help
sketch_may09a

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

c#d), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA:~1019KB), 16, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM3

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.

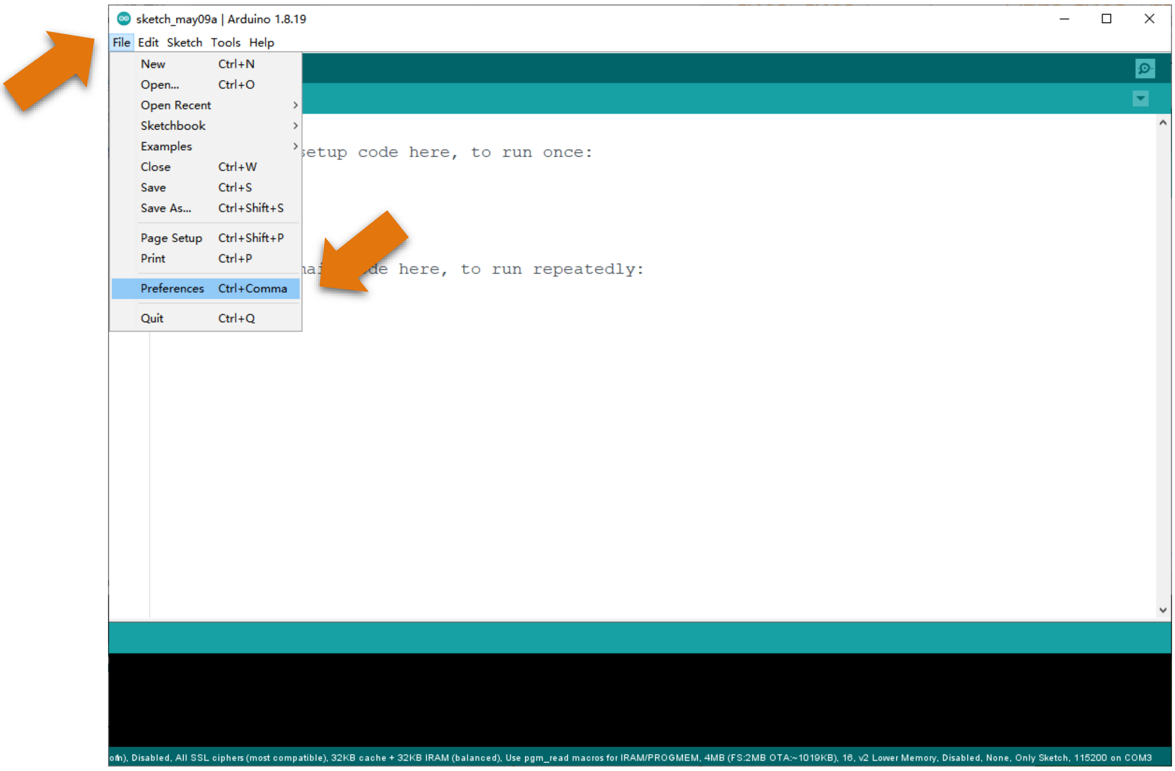
⬇ Save
Save your sketch.

🔍 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

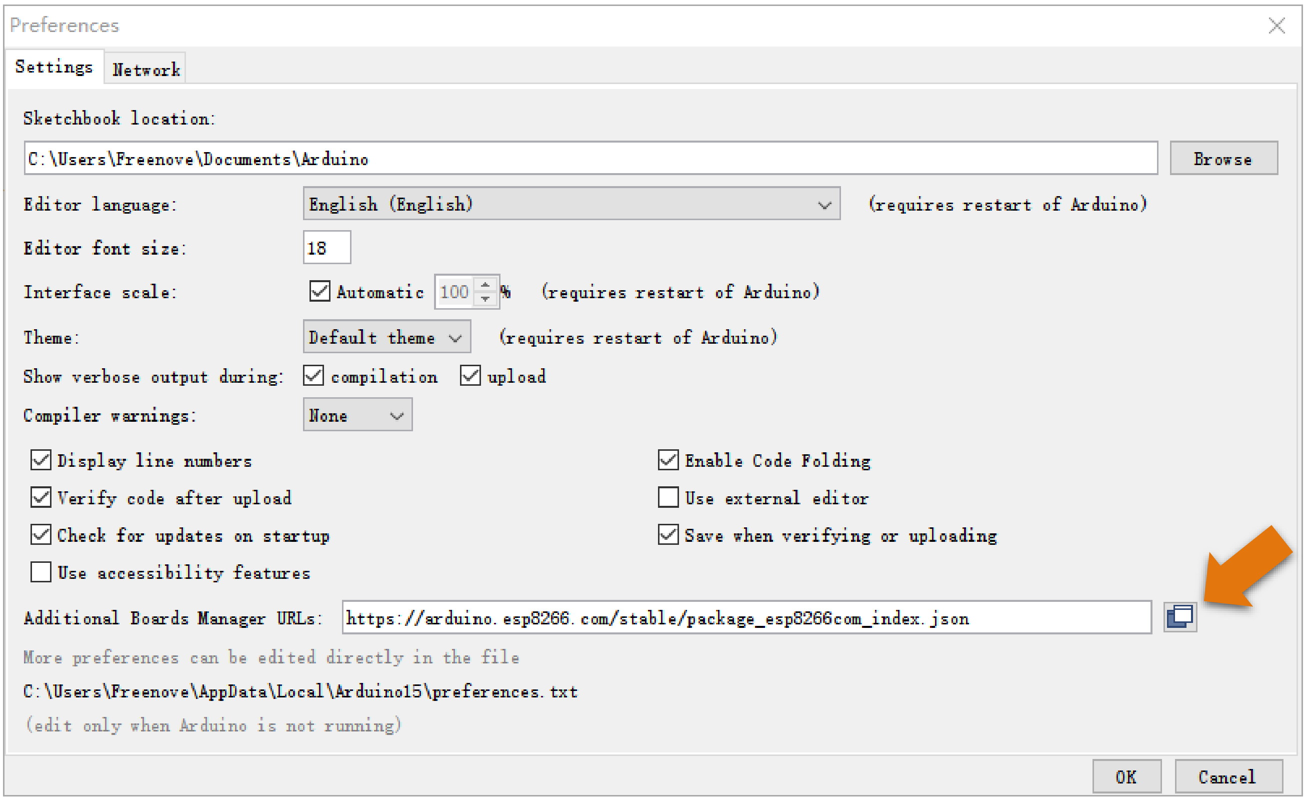
First, open the software platform arduino, and then click File in Menus and select Preferences.



```
etup code here, to run once:

ai   de here, to run repeatedly:
```

orh), Disabled, All SSL ciphers (most compatible), 32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA:~1019KB), 16, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM3

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



Sketchbook location:

C:\Users\Freenove\Documents\Arduino

Editor language: English (English) (requires restart of Arduino)

Editor font size: 18

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☑ compilation ☑ upload

Compiler warnings: None

☑ Display line numbers ☑ Enable Code Folding

☑ Verify code after upload □ Use external editor

☑ Check for updates on startup ☑ Save when verifying or uploading

□ Use accessibility features

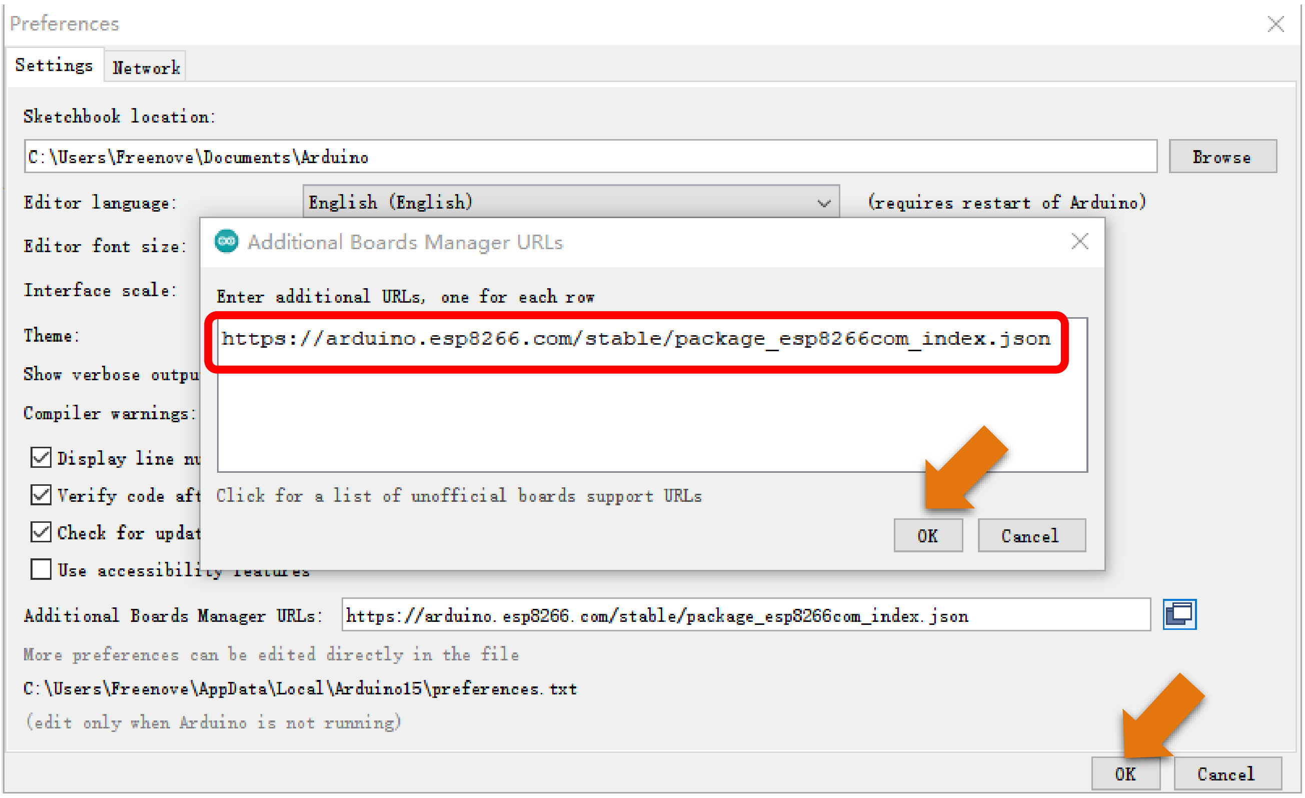
Additional Boards Manager URLs: https://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file

C:\Users\Freenove\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

Third, fill in https://arduino.esp8266.com/stable/package_esp8266com_index.json in the new window, click OK, and click OK on the Preferences window again.



Preferences

Settings | Network

Sketchbook location:

C:\Users\Freenove\Documents\Arduino

Editor language: English (English) (requires restart of Arduino)

Editor font size:

Interface scale:

Theme:

Show verbose outpu

Compiler warnings:

☑ Display line nu

☑ Verify code aft

☑ Check for updat

☐ Use accessibility features

Additional Boards Manager URLs: https://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file

C:\Users\Freenove\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

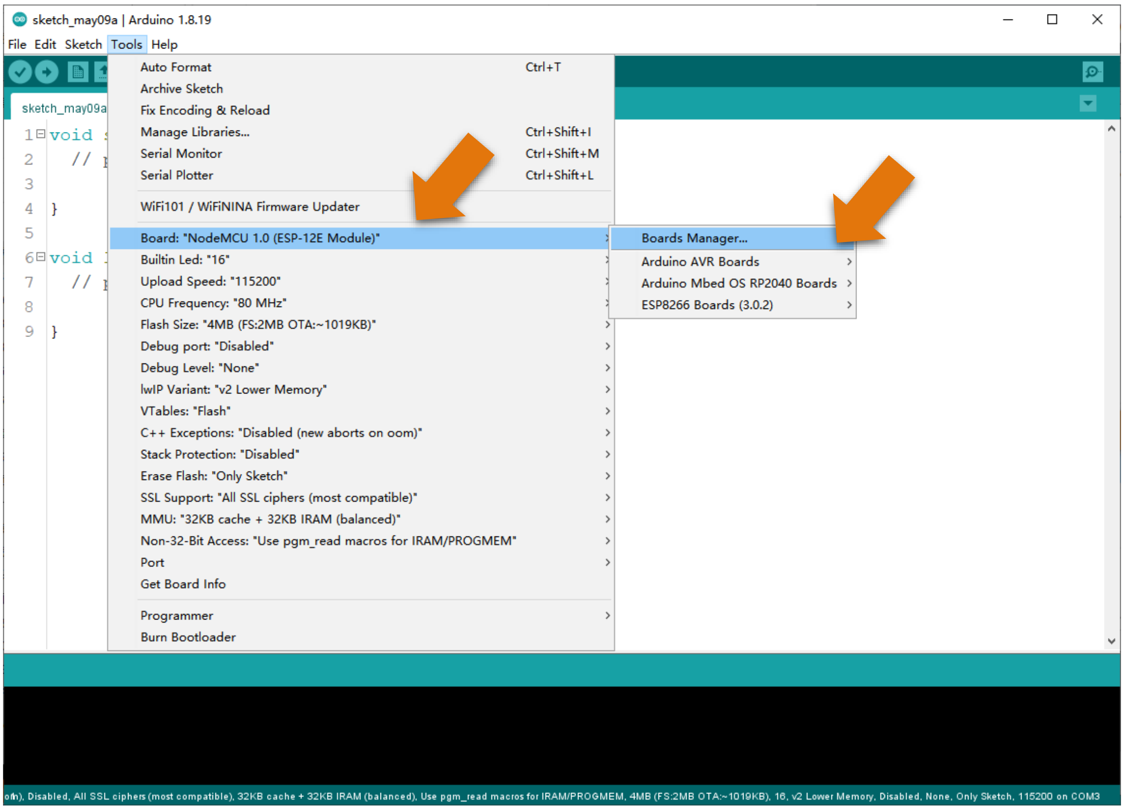
Additional Boards Manager URLs

Enter additional URLs, one for each row

```
https://arduino.esp8266.com/stable/package_esp8266com_index.json
```

Click for a list of unofficial boards support URLs

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



sketch_may09a | Arduino 1.8.19

File Edit Sketch Tools Help

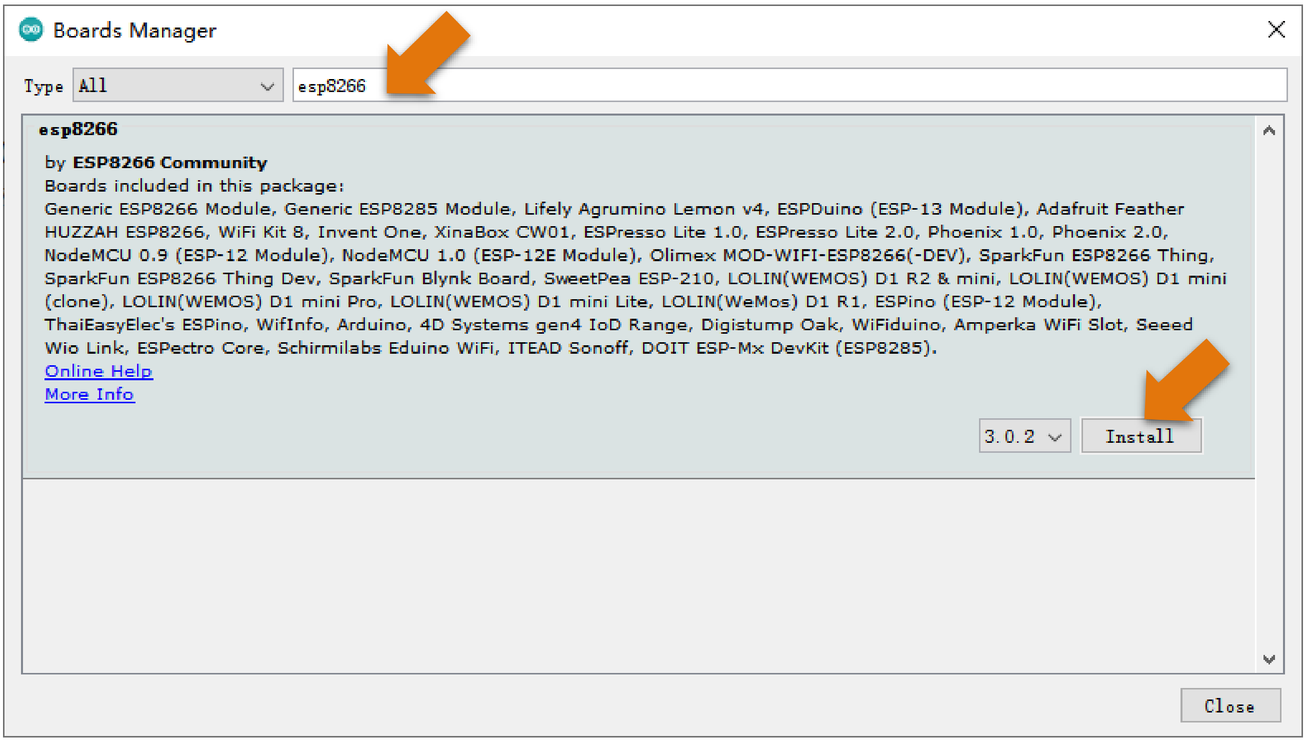
```
void setup() {
  // p
}

void loop() {
  // p
}
```

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: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "16"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port
Get Board Info
Programmer
Burn Bootloader

Boards Manager...
Arduino AVR Boards
Arduino Mbed OS RP2040 Boards
ESP8266 Boards (3.0.2)

om), Disabled, All SSL ciphers (most compatible), 32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA:~1019KB), 16, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM3

Fifth, input "esp8266" in the window below, and press Enter. click "Install" to install.



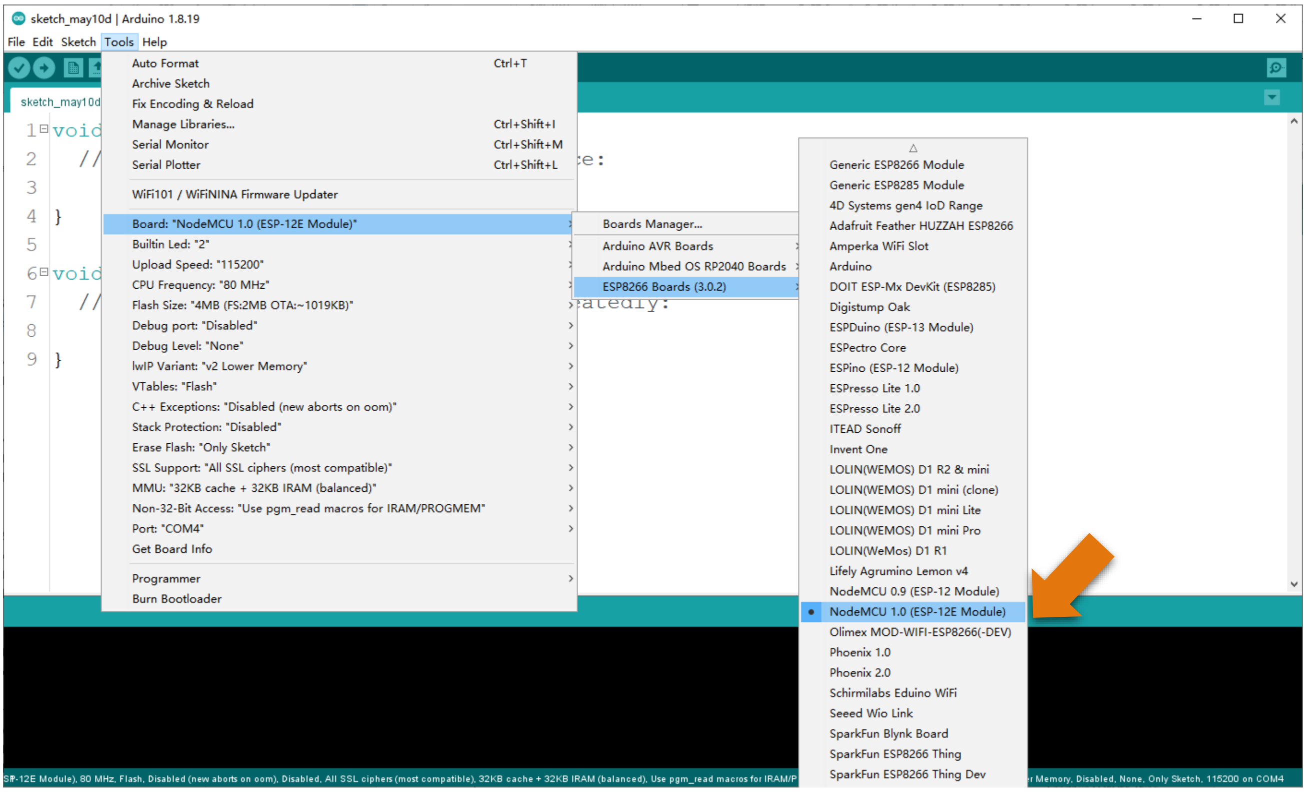
**esp8266**

by **ESP8266 Community**

Boards included in this package:

Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, WiFi Kit 8, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini (clone), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).

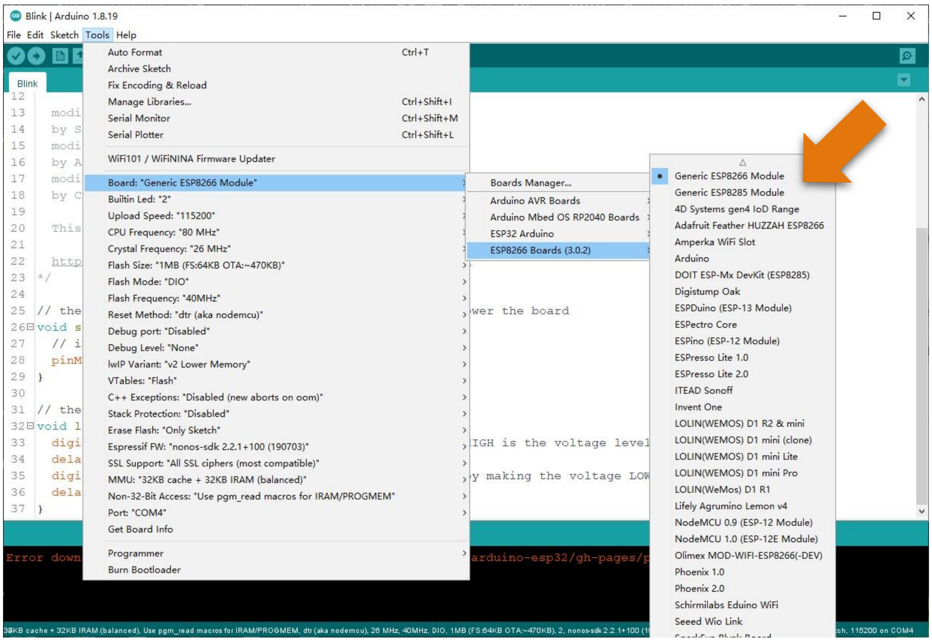
When finishing installation, click Tools in the Menus again and select Board: "NodeMCU 1.0(ESP-12E Module)", and then you can see information of ESP8266 click "NodeMCU 1.0(ESP-12E Module)" so that the ESP8266 programming development environment is configured.



Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

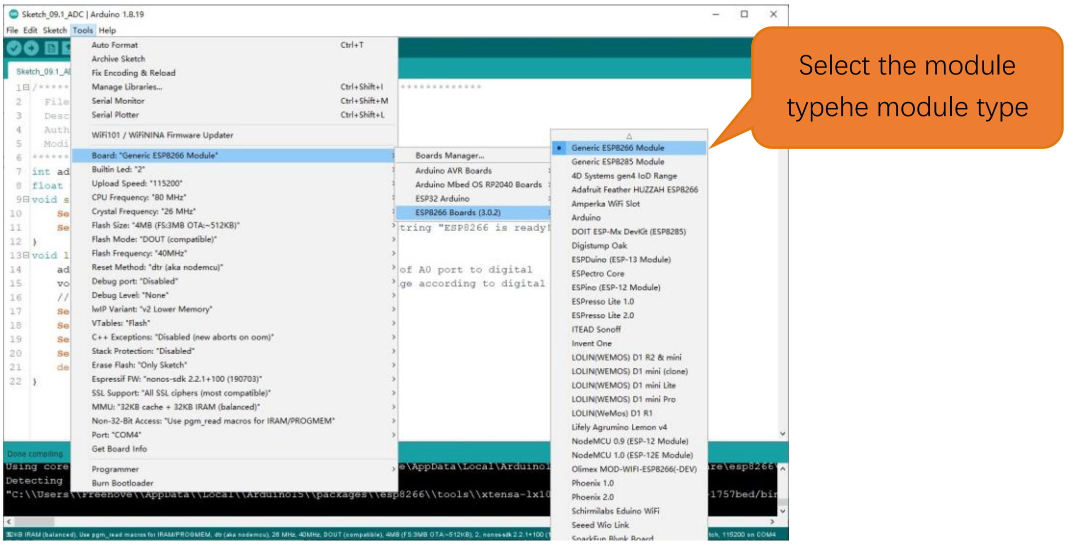
In our tutorial, we chose "NodeMCU 1.0(ESP-12E Module)" as the development board Module.This choice will facilitate learning and understanding of ESP8266.Of course, you can choose "Generic ESP8266 Module". Select "Generic ESP8266 Module" to apply to all Generic ESP8266 modules.Of course, this setup will have some more common configuration.

When you select "Generic ESP8266 Module", the interface is as follows:



Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "1MB (FS:64KB OTA:~470KB)"
Flash Mode: "DIO"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

Select the module typehe module.Here you can choose the appropriate module based on your requirements.

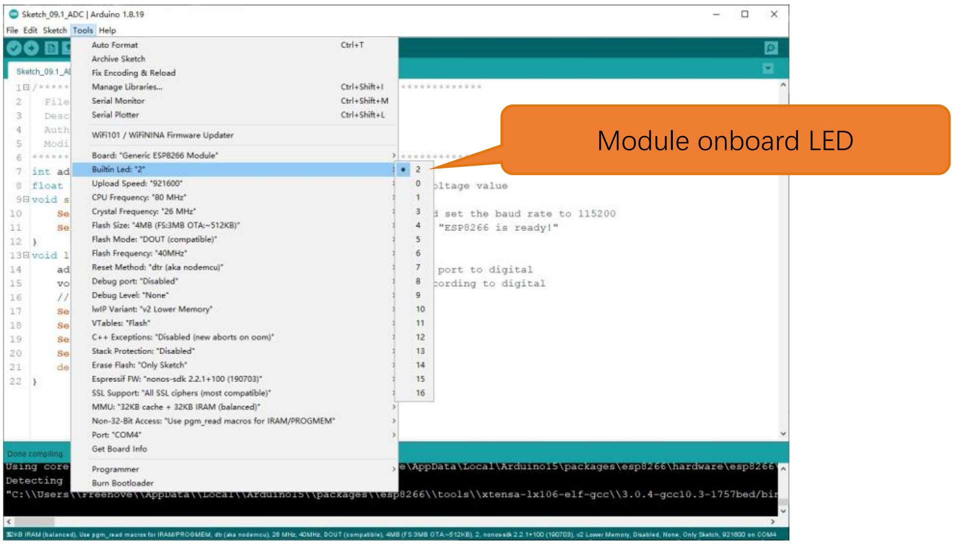


Select the module typehe module type

Sketch_09.1_ADC | Arduino 1.8.19

Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "4MB (FS:3MB OTA:~512KB)"
Flash Mode: "DOUT (compatible)"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

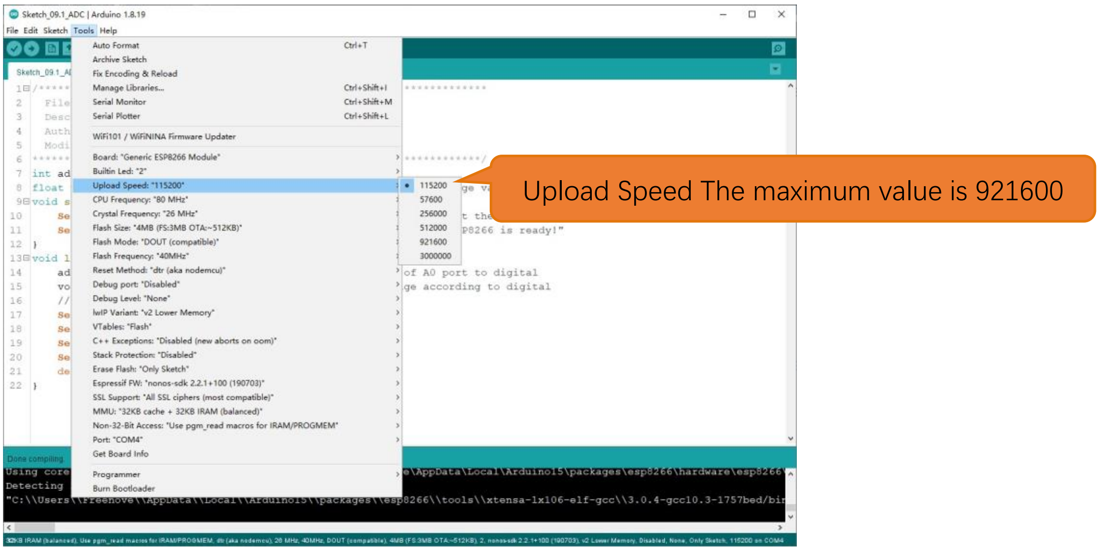
Module Onboard LED, in our ESP8266 development board, has an onboard LED of 2.



Module onboard LED

Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "921600"

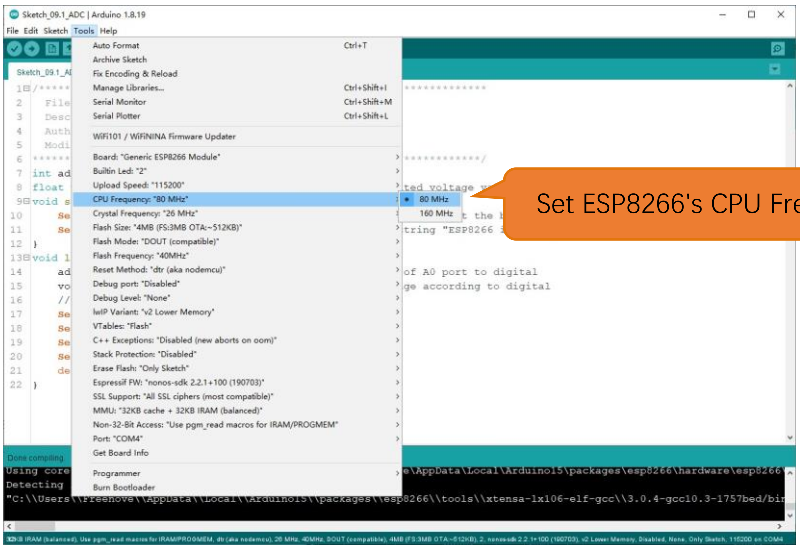
Upload Speed The maximum value is 921600.By default, Upload Speed is 115200.You can choose according to your needs.



Upload Speed The maximum value is 921600

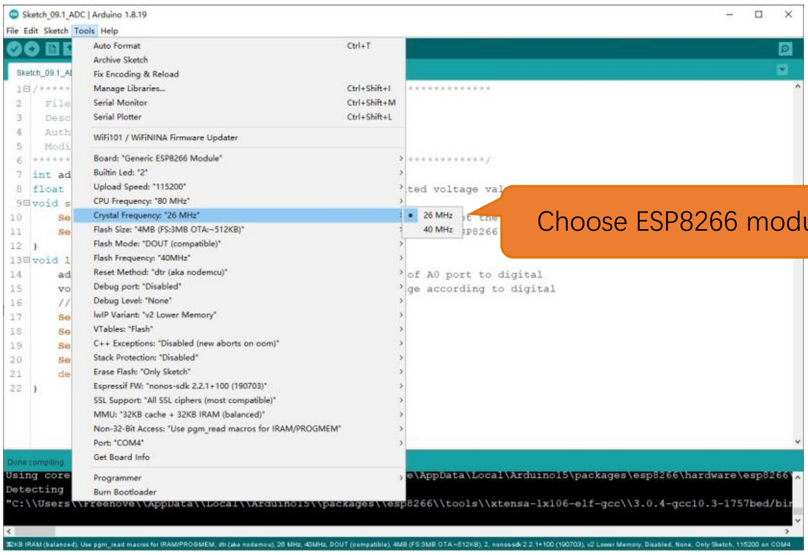
Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"

ESP8266's CPU frequency standard is 80MHz, which can be changed to 160MHz.


Sketch_09.1_ADC | Arduino 1.8.19
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: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "4MB (FS:3MB OTA:~512KB)"
Flash Mode: "DOUT (compatible)"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"
Get Board Info
Programmer
Burn Bootloader
● 80 MHz
160 MHz

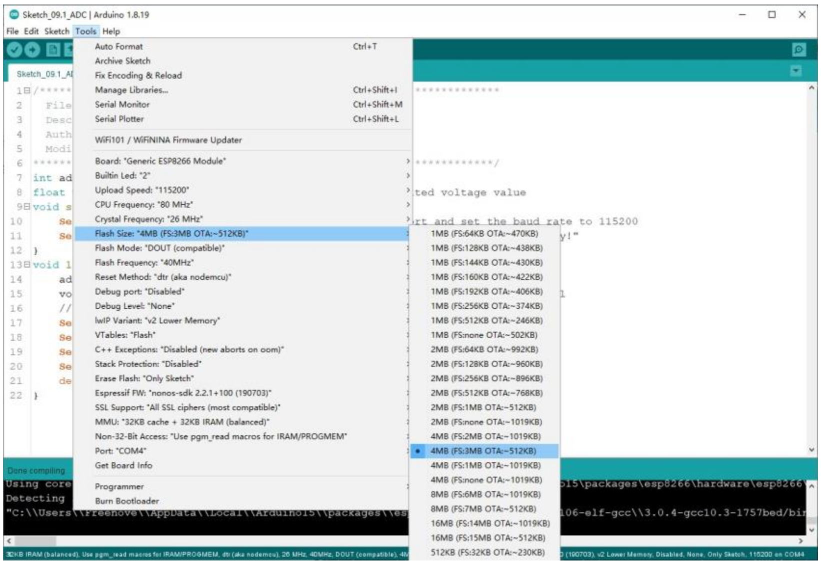
Set ESP8266's CPU Frequency to 80MHZ

Most ESP8266 modules use 26 MHz crystals, but some have other values.


Sketch_09.1_ADC | Arduino 1.8.19
File Edit Sketch Tools Help
Crystal Frequency: "26 MHz"
● 26 MHz
40 MHz

Choose ESP8266 modules using 26MHZ

Choose the appropriate Flash size based on your ESP8266 module type. In our ESP8266 development board, we chose 4MB (FS: 3MB OTA: ~512KB).


Sketch_09.1_ADC | Arduino 1.8.19
File Edit Sketch Tools Help
Flash Size: "4MB (FS:3MB OTA:~512KB)"
1MB (FS:64KB OTA:~470KB)
1MB (FS:128KB OTA:~438KB)
1MB (FS:144KB OTA:~430KB)
1MB (FS:160KB OTA:~422KB)
1MB (FS:192KB OTA:~406KB)
1MB (FS:256KB OTA:~374KB)
1MB (FS:512KB OTA:~246KB)
1MB (FS:none OTA:~502KB)
2MB (FS:64KB OTA:~992KB)
2MB (FS:128KB OTA:~960KB)
2MB (FS:256KB OTA:~896KB)
2MB (FS:512KB OTA:~768KB)
2MB (FS:1MB OTA:~512KB)
2MB (FS:none OTA:~1019KB)
4MB (FS:2MB OTA:~1019KB)
● 4MB (FS:3MB OTA:~512KB)
4MB (FS:1MB OTA:~1019KB)
4MB (FS:none OTA:~1019KB)
8MB (FS:6MB OTA:~1019KB)
8MB (FS:7MB OTA:~512KB)
16MB (FS:14MB OTA:~1019KB)
16MB (FS:15MB OTA:~512KB)
512KB (FS:32KB OTA:~230KB)

Here we need to select Flash mode.On our ESP8266 development board, choose "DIO" mode or "DOUT" mode for better compatibility.If the ESP8266 module is abnormal, check whether the ESP8266 module works in the two modes.

Flash works in DOUT, DIO, QOUT, and QIO modes.

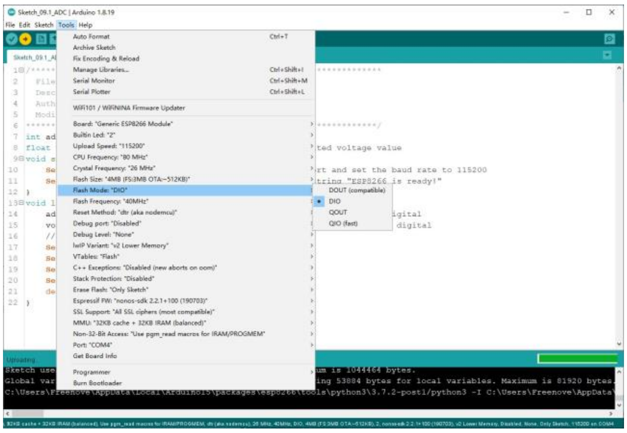
1.DOUT: Address is input in 1-line mode and data is output in 2-line mode.

2.DIO: Address is input in 2-line mode and data is output in 2-line mode.

3.QOUT: Address is input in 1-line mode and data is output in 4-line mode.

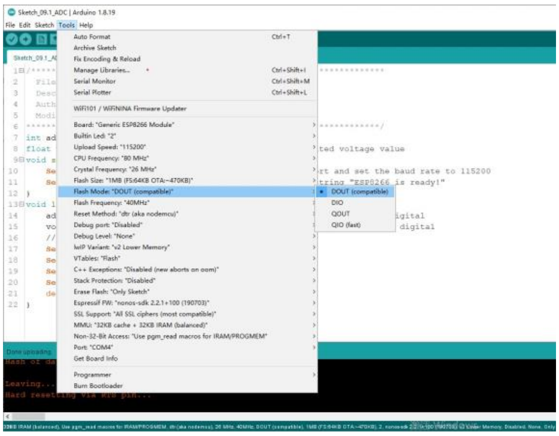
4.QIO: Address is input in 4-line mode and data is output in 4-line mode.

If you need to use the QIO mode, ensure that the Flash supports the QIO mode.


Sketch_09.1_ADC | Arduino 1.8.19
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
Get Board Info
Programmer
Burn Bootloader
DOUT (compatible)
● DIO
QOUT
QIO (fast)
Uploading...

Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "4MB (FS:3MB OTA:~512KB)"
Flash Mode: "DIO"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

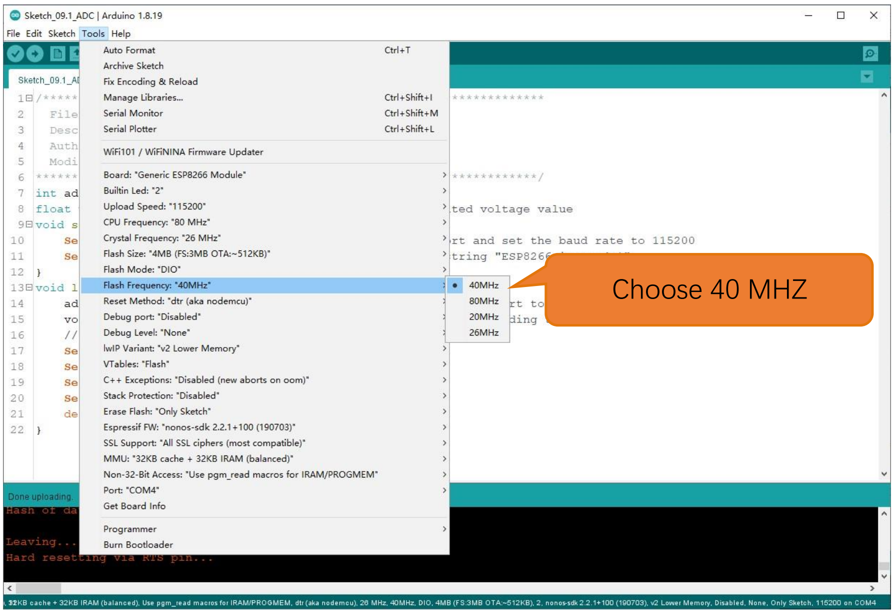
```
Sketch use                                um is 1044464 bytes.
Global var                                ing 53884 bytes for local variables. Maximum is 81920 bytes.
C:\Users\Freenove\AppData\Local\Arduino15\packages\esp8266\tools\python3\3.7.2-post1/python3 -I C:\Users\Freenove\AppData\
```


Sketch_09.1_ADC | Arduino 1.8.19
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
Get Board Info
Programmer
Burn Bootloader
● DOUT (compatible)
DIO
QOUT
QIO (fast)
Done uploading.

Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "1MB (FS:64KB OTA:~470KB)"
Flash Mode: "DOUT (compatible)"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

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

The flash chip connected to most chips operates at 40MHz clock speed, but you can try a lower value if the device fails to boot.The highest flash clock speed of 80MHz will provide the best performance, but can cause crashes if the flash or board design cannot achieve this speed.


Sketch_09.1_ADC | Arduino 1.8.19
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
Get Board Info
Programmer
Burn Bootloader
● 40MHz
80MHz
20MHz
26MHz
Done uploading.

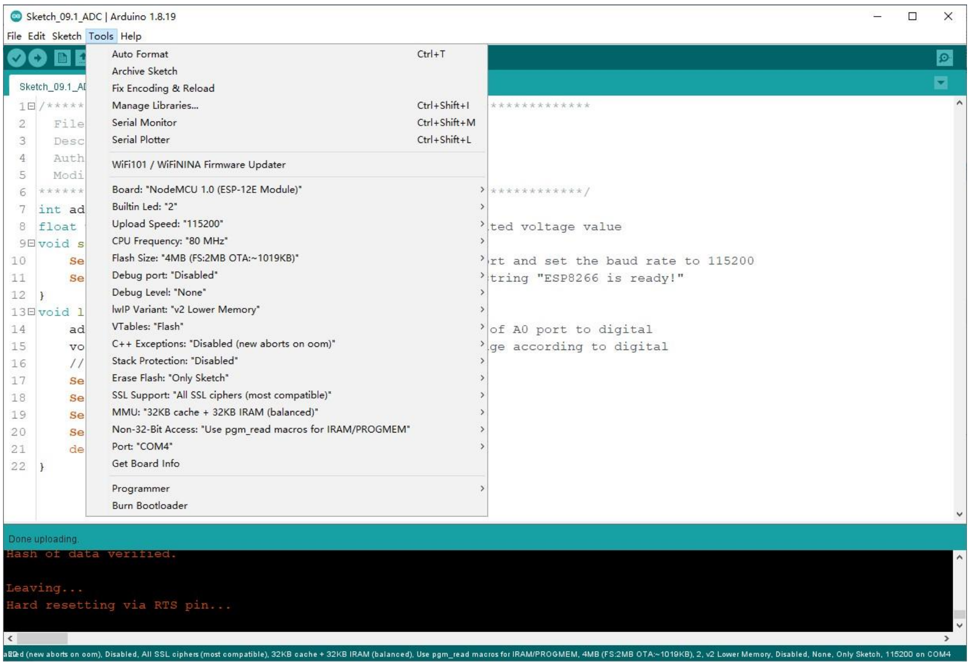
Board: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "4MB (FS:3MB OTA:~512KB)"
Flash Mode: "DIO"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

```
Hash of da
Leaving...
Hard resetting via RTS pin...
```

32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRAM/PROGMEM, dtr (aka nodemcu), 26 MHz, 40MHz, DIO, 4MB (FS:3MB OTA:~512KB), 2, nonos-sdk 2.2.1+100 (190703), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM4

Choose 40 MHZ

If you select NodeMCU 1.0(ESP-12E Module), the following interface is displayed:


Sketch_09.1_ADC | Arduino 1.8.19
File Edit Sketch Tools Help
Sketch_09.1_ADC
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
Get Board Info
Programmer
Burn Bootloader

Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM4"

Done uploading.

```
Hash of data verified.

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

abled (new aborts on oom), Disabled, All SSL ciphers (most compatible), 32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM4

Here, you can see that this is similar to "Generic ESP8266 Module" in that the omitted parts are configured with default values.If you have problems working through this tutorial, try using the "Generic ESP8266 Module" configuration.

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

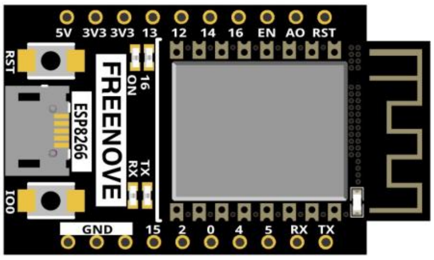
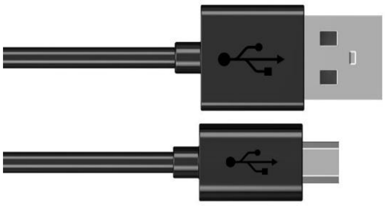
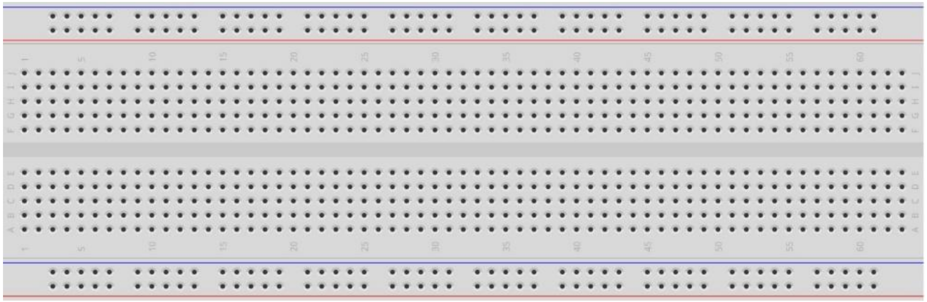
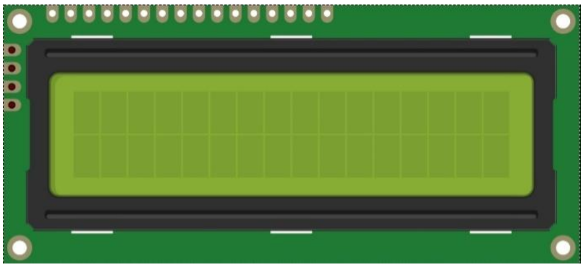
# Chapter 1 LCD1602

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

| | |
|---|---|
| ESP8266 x1<br><br> 5V 3V3 3V3 13 12 14 16 EN AO RST<br>RST<br>IO0<br>FREENOVE<br>ESP8266<br>16<br>ON<br>TX<br>RX<br>GND 15 2 0 4 5 RX TX | USB cable<br> |
| Breadboard x1<br> | |
| LCD1602 Module x1<br> | Jumper wire F/M x6<br> |

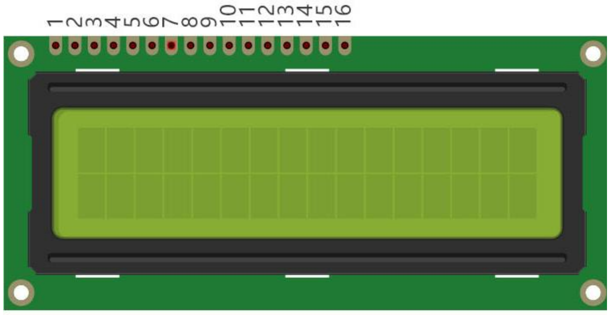
## Component knowledge

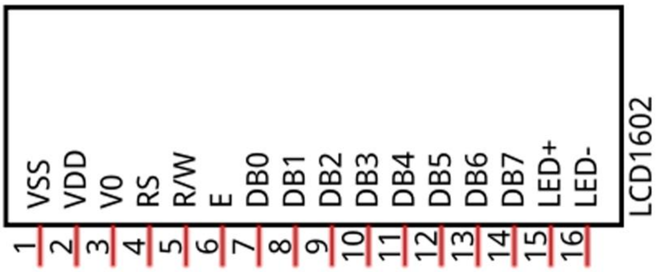
### I2C communication

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.

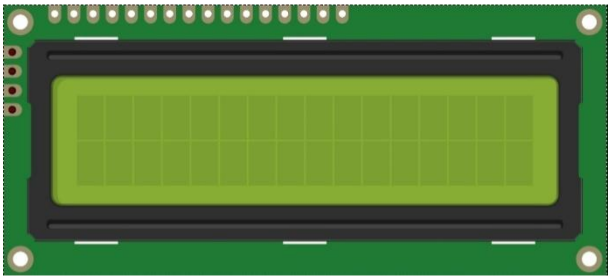
### LCD1602 communication

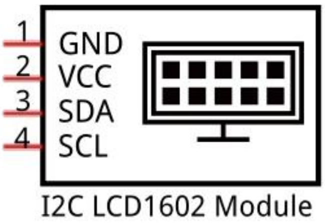
The LCD1602 display screen can display 2 lines of characters in 16 columns. It is capable of displaying numbers, letters, symbols, ASCII code and so on. As shown below is a monochrome LCD1602 display screen along with its circuit pin diagram.





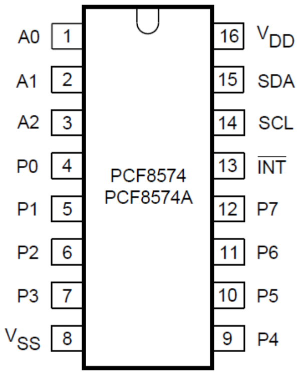
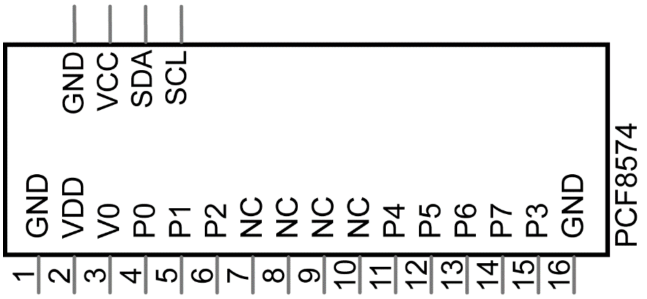
I2C LCD1602 display screen integrates a I2C interface, which connects the serial-input & parallel-output module to the LCD1602 display screen. This allows us to only use 4 lines to the operate the LCD1602.



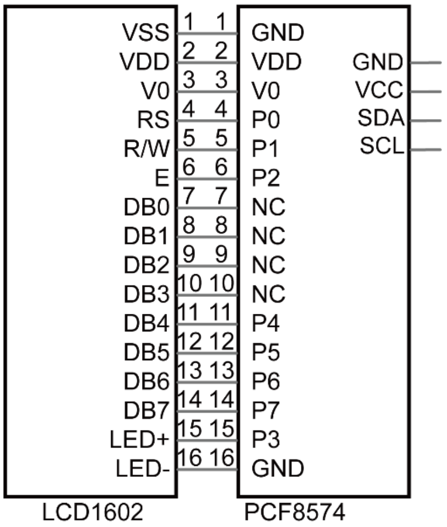


The serial-to-parallel IC chip used in this module is PCF8574T (PCF8574AT), and its default I2C address is 0x27(0x3F).

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

| PCF8574 chip pin diagram | PCF8574 module pin diagram |
|---|---|
|  |  |

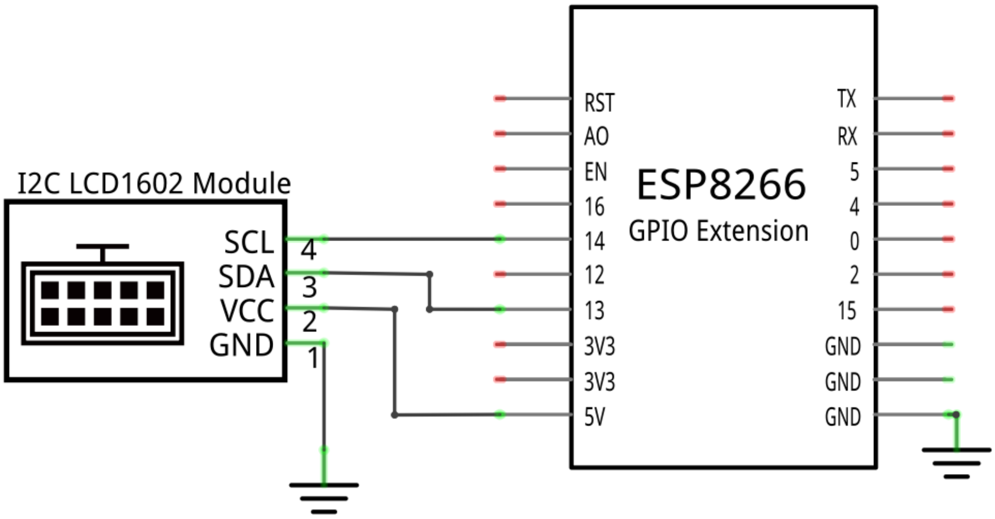
PCF8574 module pin and LCD1602 pin are corresponding to each other and connected with each other:



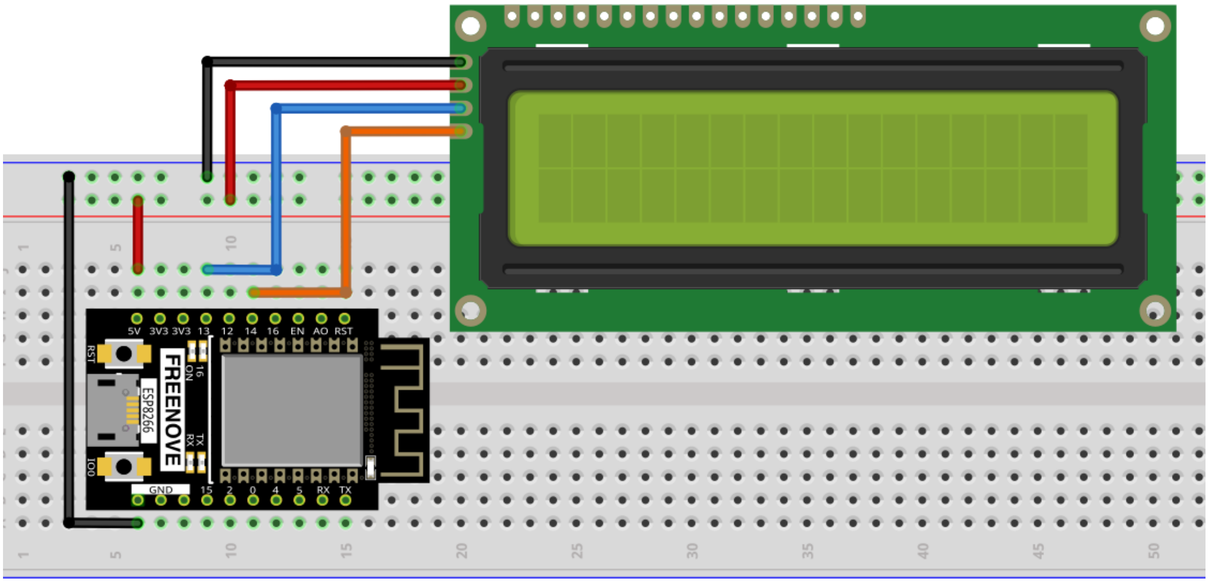
So we only need 4 pins to control the 16 pins of the LCD1602 display screen through the I2C interface. In this project, we will use the I2C LCD1602 to display some static characters and dynamic variables.

## Circuit

Schematic diagram



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

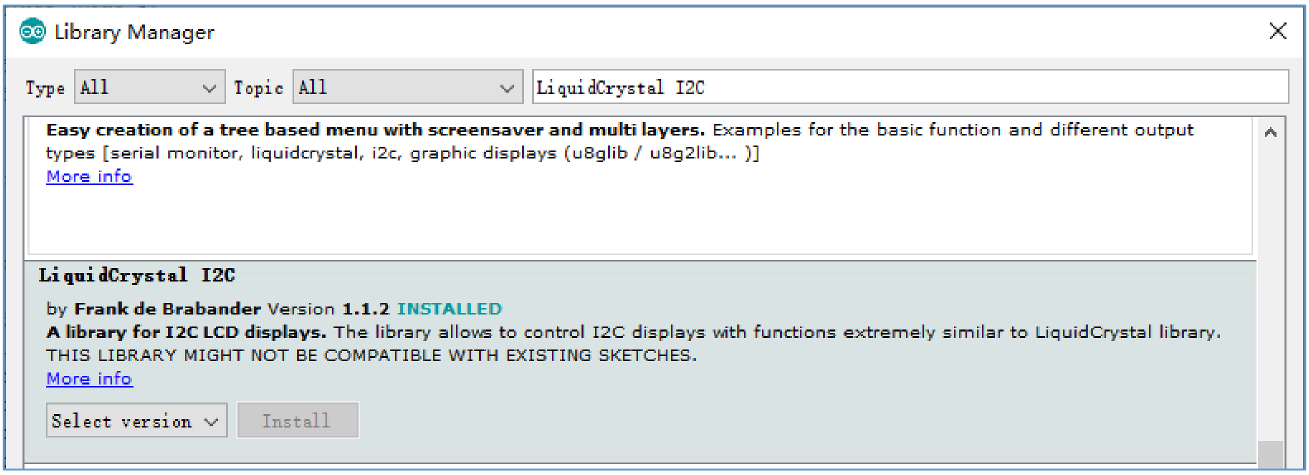


## Sketch

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.



**Easy creation of a tree based menu with screensaver and multi layers.** Examples for the basic function and different output types [serial monitor, liquidcrystal, i2c, graphic displays (u8glib / u8g2lib... )]
More info

**LiquidCrystal I2C**

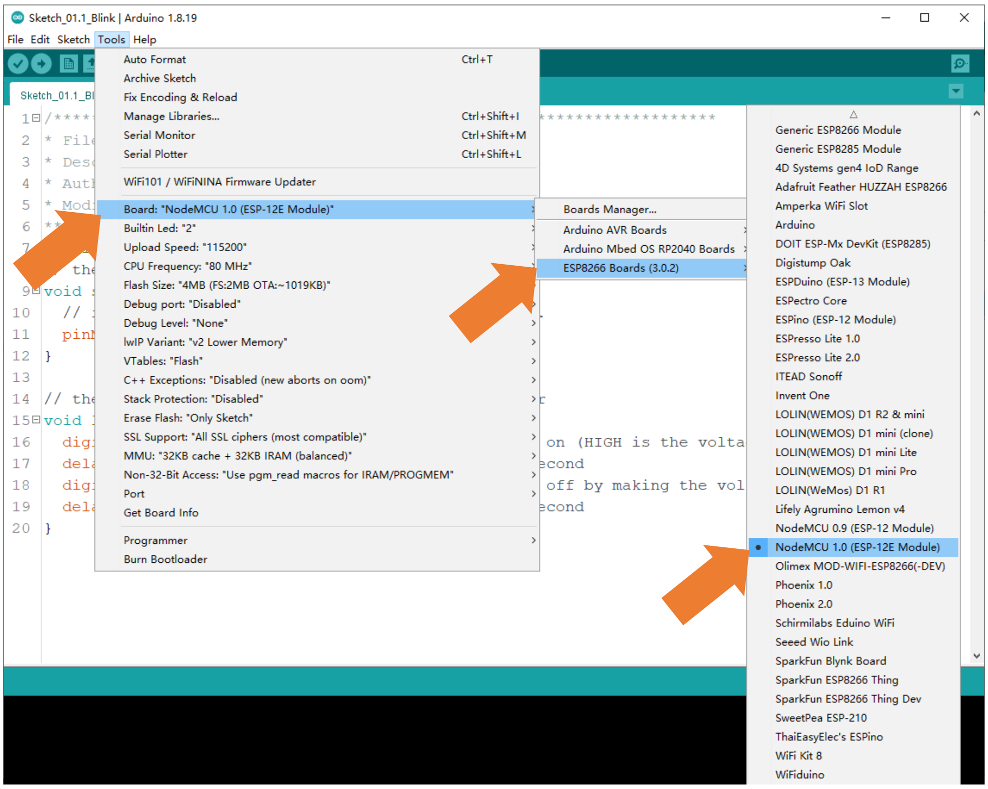
by **Frank de Brabander** Version **1.1.2** **INSTALLED**
**A library for I2C LCD displays.** The library allows to control I2C displays with functions extremely similar to LiquidCrystal library. THIS LIBRARY MIGHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.
More info

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 form the ZIP file we provided). This library can facilitate our operation of I2C LCD1602.

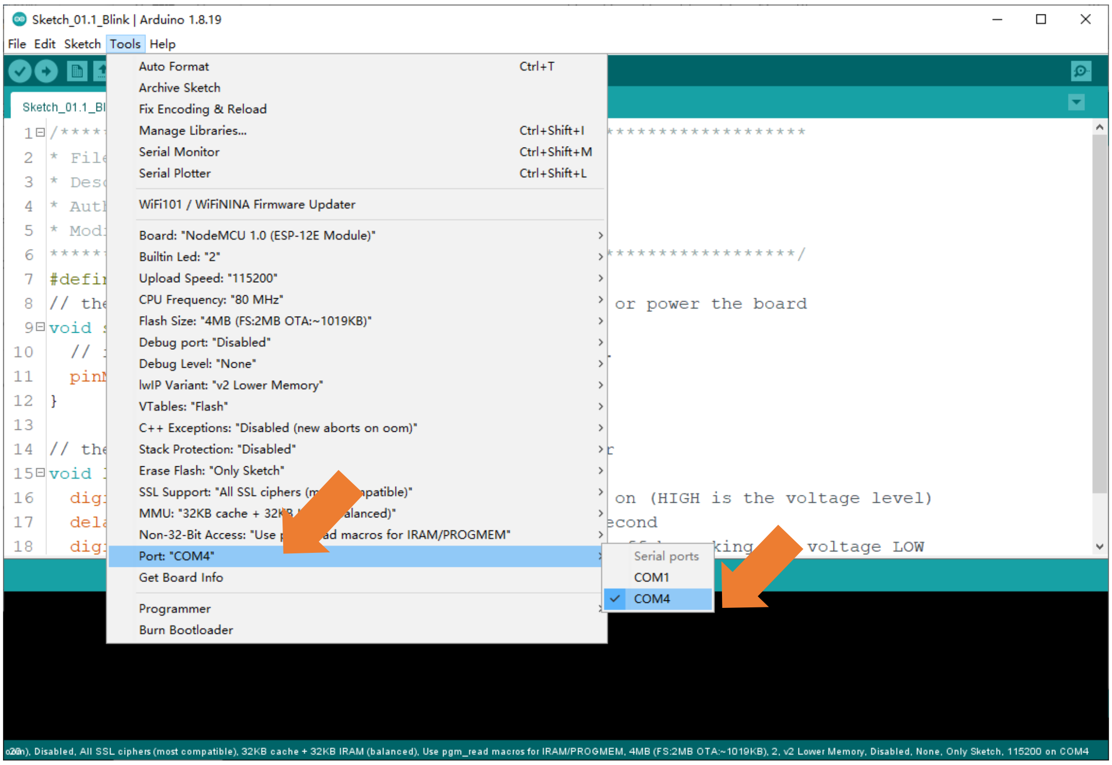
Use I2C LCD 1602 to display characters and variables.

Before uploading the code, click "**Tools**", "**Board**" and select "**NodeMCU 1.0（ESP-12E Module）**".



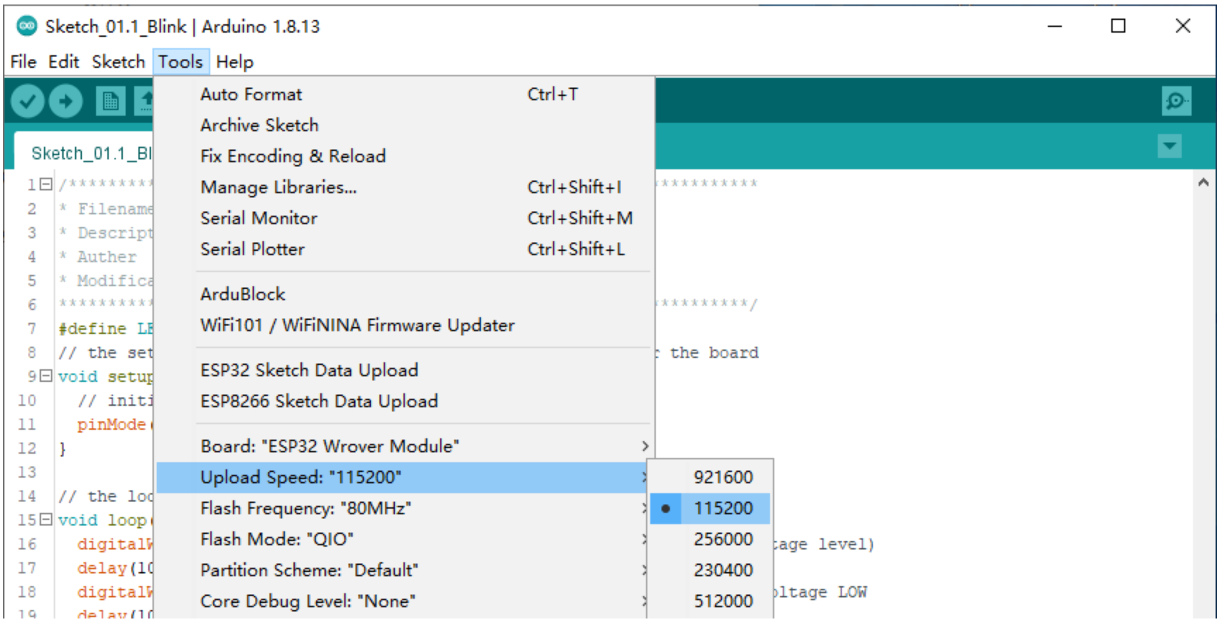
Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"

Select the serial port.

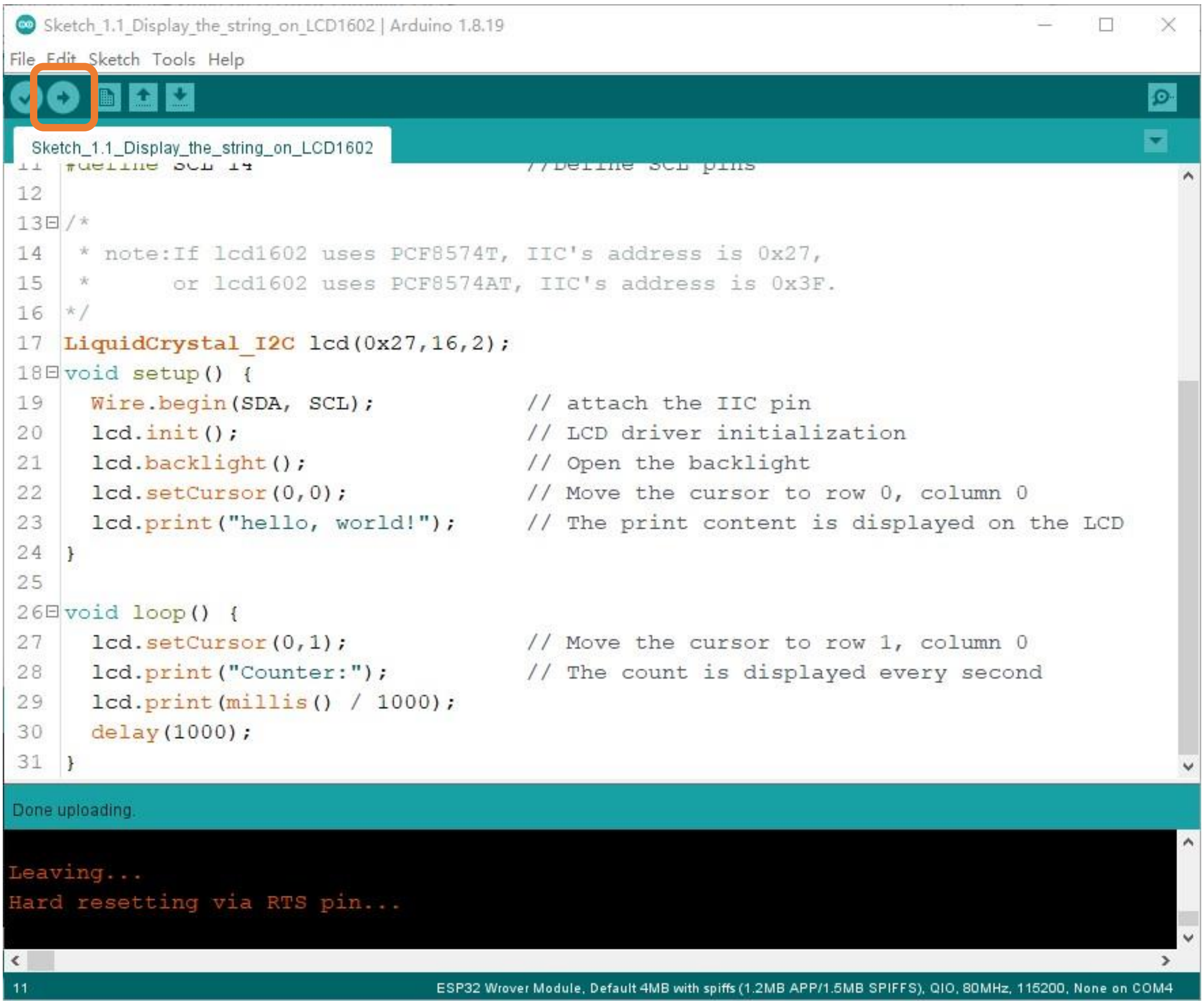

Sketch_01.1_Blink | Arduino 1.8.19
File Edit Sketch Tools Help
Sketch_01.1_Bl
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (m patible)"
MMU: "32KB cache + 32K alanced)"
Non-32-Bit Access: "Use p ad macros for IRAM/PROGMEM"
Port: "COM4"
Get Board Info
Programmer
Burn Bootloader
Serial ports
COM1
✓ COM4
or power the board
on (HIGH is the voltage level)
econd
king voltage LOW

oom), Disabled, All SSL ciphers (most compatible), 32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM4

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


Sketch_01.1_Blink | Arduino 1.8.13
File Edit Sketch Tools Help
Sketch_01.1_Bl
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
ArduBlock
WiFi101 / WiFiNINA Firmware Updater
ESP32 Sketch Data Upload
ESP8266 Sketch Data Upload
Board: "ESP32 Wrover Module"
Upload Speed: "115200"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Partition Scheme: "Default"
Core Debug Level: "None"
921600
● 115200
256000
230400
512000
the board
age level)
ltage LOW

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

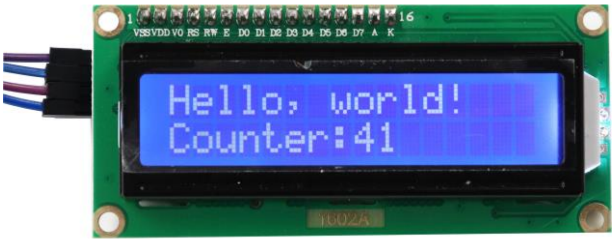
```
#define SCL 14                      //Define SCL pins

/*
 * note:If lcd1602 uses PCF8574T, IIC's address is 0x27,
 *      or lcd1602 uses PCF8574AT, IIC's address is 0x3F.
*/
LiquidCrystal_I2C lcd(0x27,16,2);
void setup() {
  Wire.begin(SDA, SCL);           // attach the IIC pin
  lcd.init();                     // LCD driver initialization
  lcd.backlight();                // Open the backlight
  lcd.setCursor(0,0);             // Move the cursor to row 0, column 0
  lcd.print("hello, world!");     // The print content is displayed on the LCD
}

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

Done uploading.

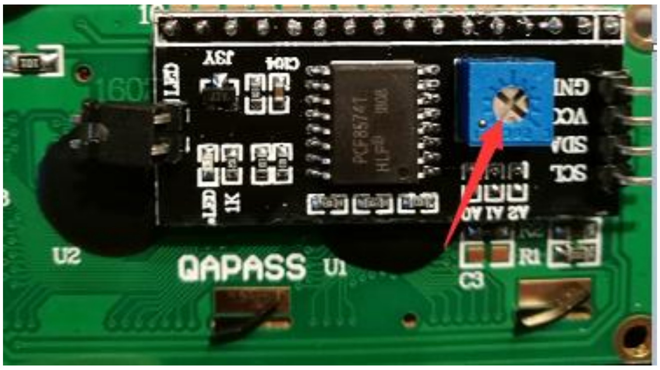
```
Leaving...
Hard resetting via RTS pin...
```

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 ESP8266 and the LCD1602 displays characters.



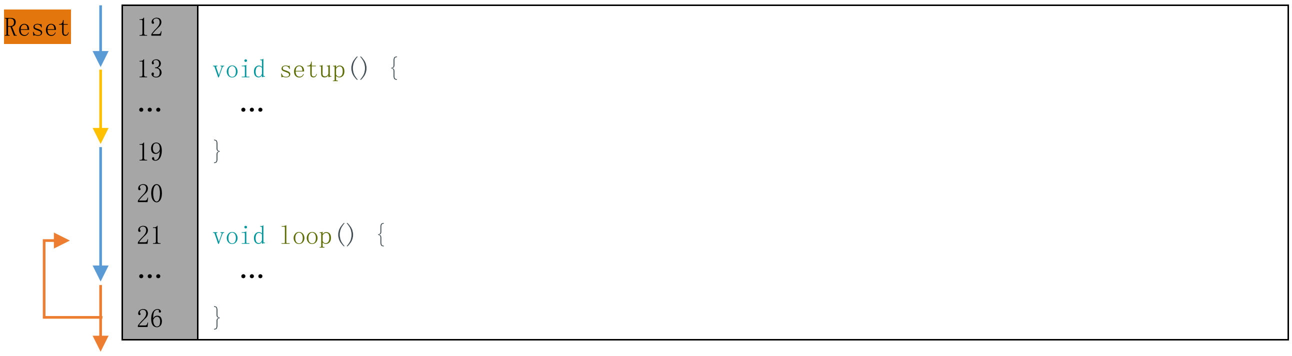
If you cannot see anything on the display or the display is not clear, try rotating the white knob on back of LCD1602 slowly, which adjusts the contrast, until the screen can display clearly.



The following is the program code:

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>

#define SDA 13                    //Define SDA pins
#define SCL 14                    //Define SCL pins

/*
 * note: If lcd1602 uses PCF8574T, IIC's address is 0x27,
 *      or lcd1602 uses PCF8574AT, IIC's address is 0x3F.
*/
LiquidCrystal_I2C lcd(0x27,16,2);

void setup() {
  Wire.begin(SDA, SCL);           // attach the IIC pin
  lcd.init();                     // LCD driver initialization
  lcd.backlight();                // Turn on the backlight
  lcd.setCursor(0,0);             // Move the cursor to row 0, column 0
  lcd.print("hello, world! ");     // The print content is displayed on the LCD
}

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

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.



```

void setup() {
  …
}

void loop() {
  …
}
```

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

```
#include <LiquidCrystal_I2C.h>
#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.

```
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.

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

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

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

Print the number on the second line of LCD1602.

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

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

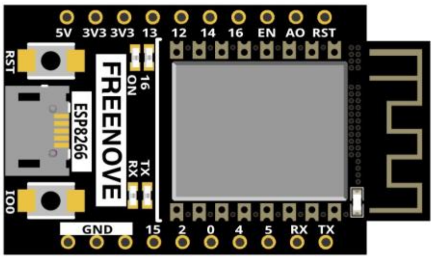
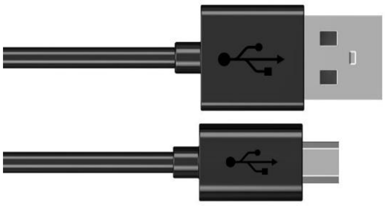
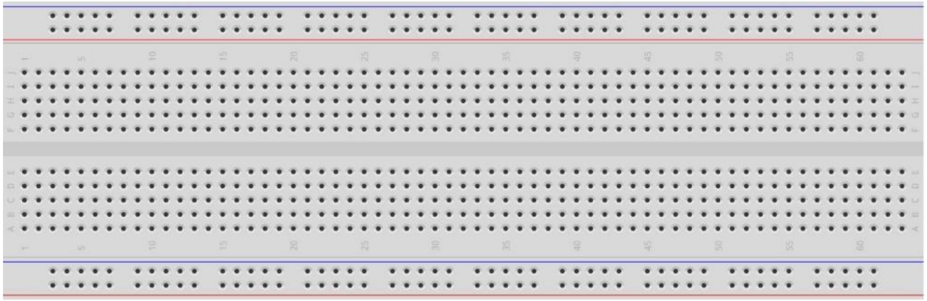
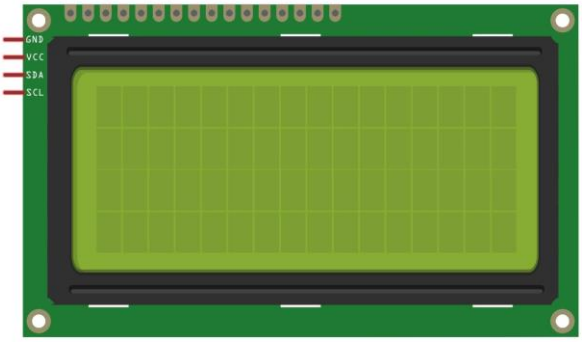
# Chapter 2 LCD2004

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

| | |
|---|---|
| ESP8266 x1<br><br> 5V 3V3 3V3 13 12 14 16 EN AO RST<br>RST<br>FREENOVE<br>ESP8266<br>16<br>ON<br>TX<br>RX<br>IO0<br>GND 15 2 0 4 5 RX TX | USB cable<br> |
| Breadboard x1<br> | |
| LCD2004 Module x1<br><br> | Jumper F/M x5<br> |

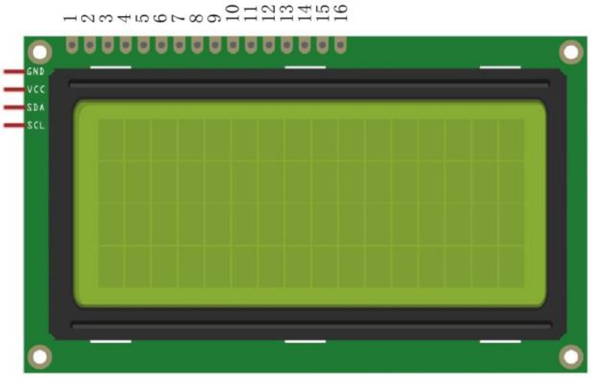
## Component knowledge

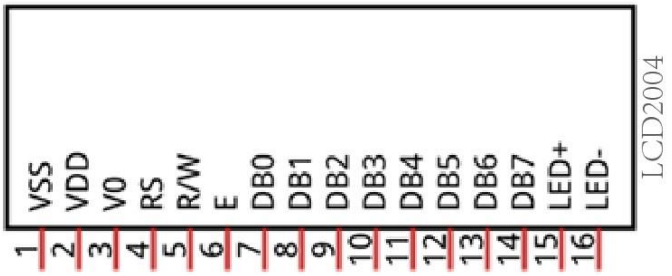
### I2C communication

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.

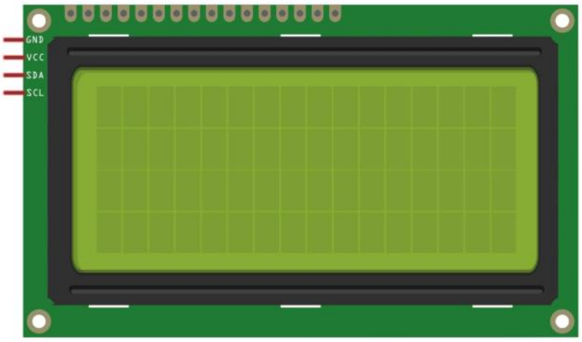
### LCD2004 communication

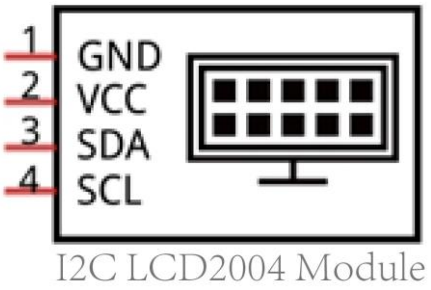
The LCD2004 display screen can display 4 lines of characters in 20 columns. It is capable of displaying numbers, letters, symbols, ASCII code and so on. As shown below is a monochrome LCD2004 display screen along with its circuit pin diagram.





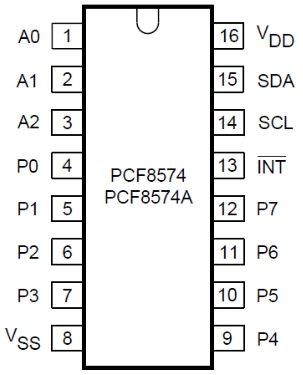
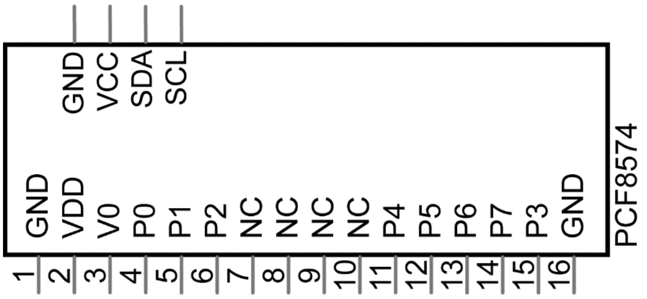
I2C LCD2004 display screen integrates a I2C interface, which connects the serial-input & parallel-output module to the LCD2004 display screen. This allows us to only use 4 lines to the operate the LCD2004.



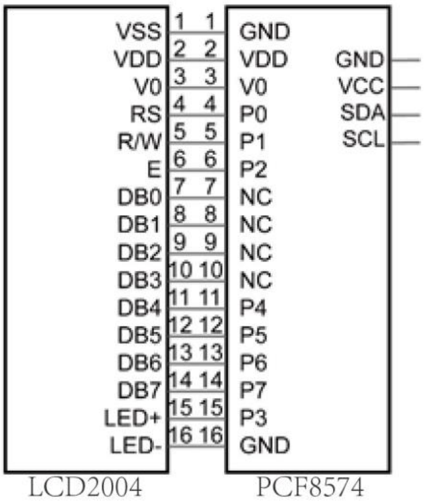


The serial-to-parallel IC chip used in this module is PCF8574T (PCF8574AT), and its default I2C address is 0x27(0x3F).

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

| PCF8574 chip pin diagram | PCF8574 module pin diagram |
|---|---|
|  |  |

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

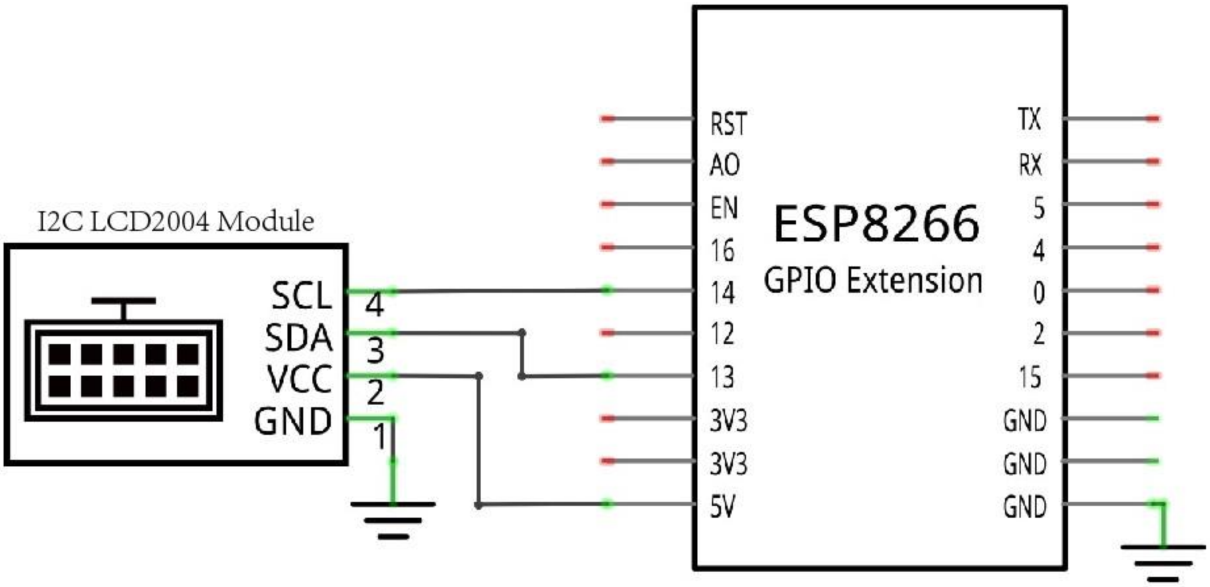


So we only need 4 pins to control the 16 pins of the LCD2004 display screen through the I2C interface.

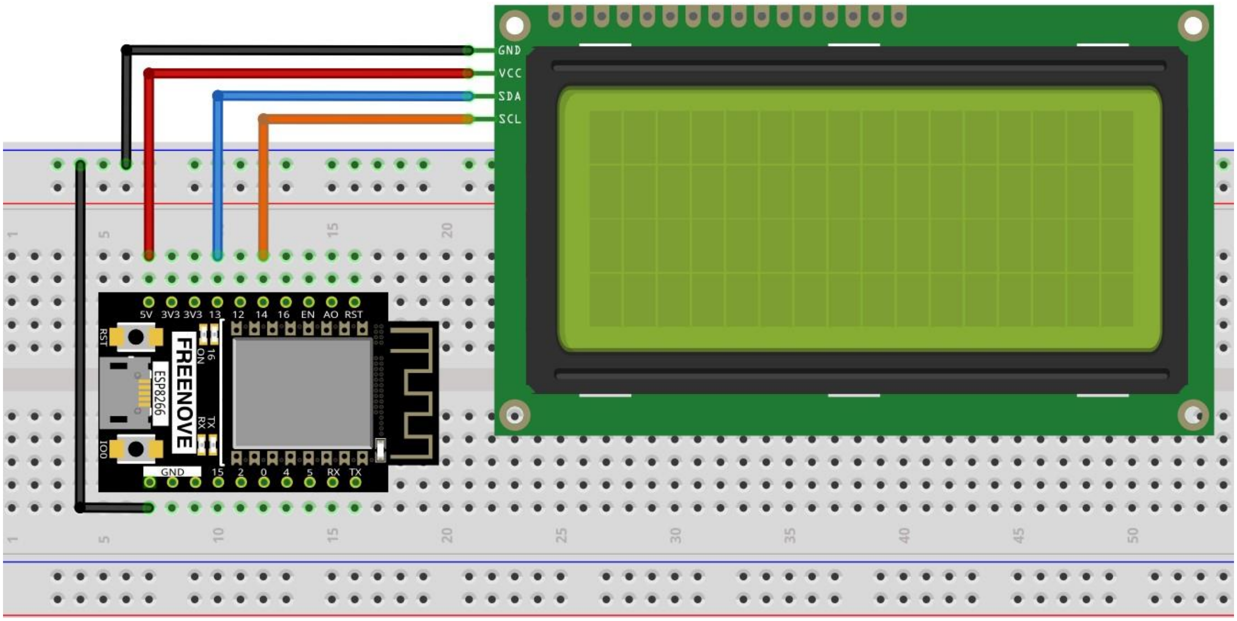
In this project, we will use the I2C LCD2004 to display some static characters and dynamic variables.

## Circuit

Schematic diagram



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

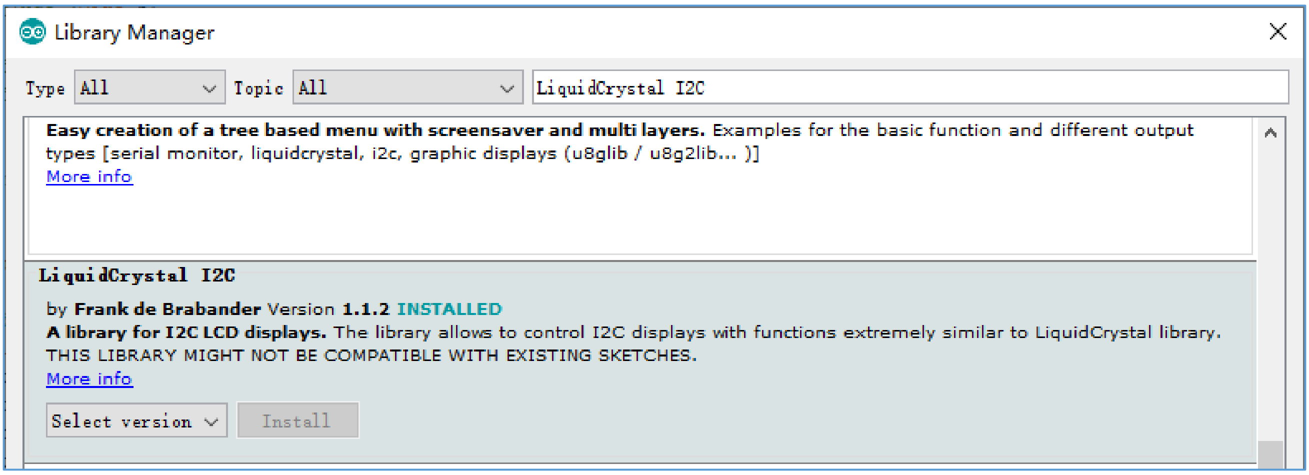


## Sketch

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.



**Easy creation of a tree based menu with screensaver and multi layers.** Examples for the basic function and different output types [serial monitor, liquidcrystal, i2c, graphic displays (u8glib / u8g2lib... )]
More info

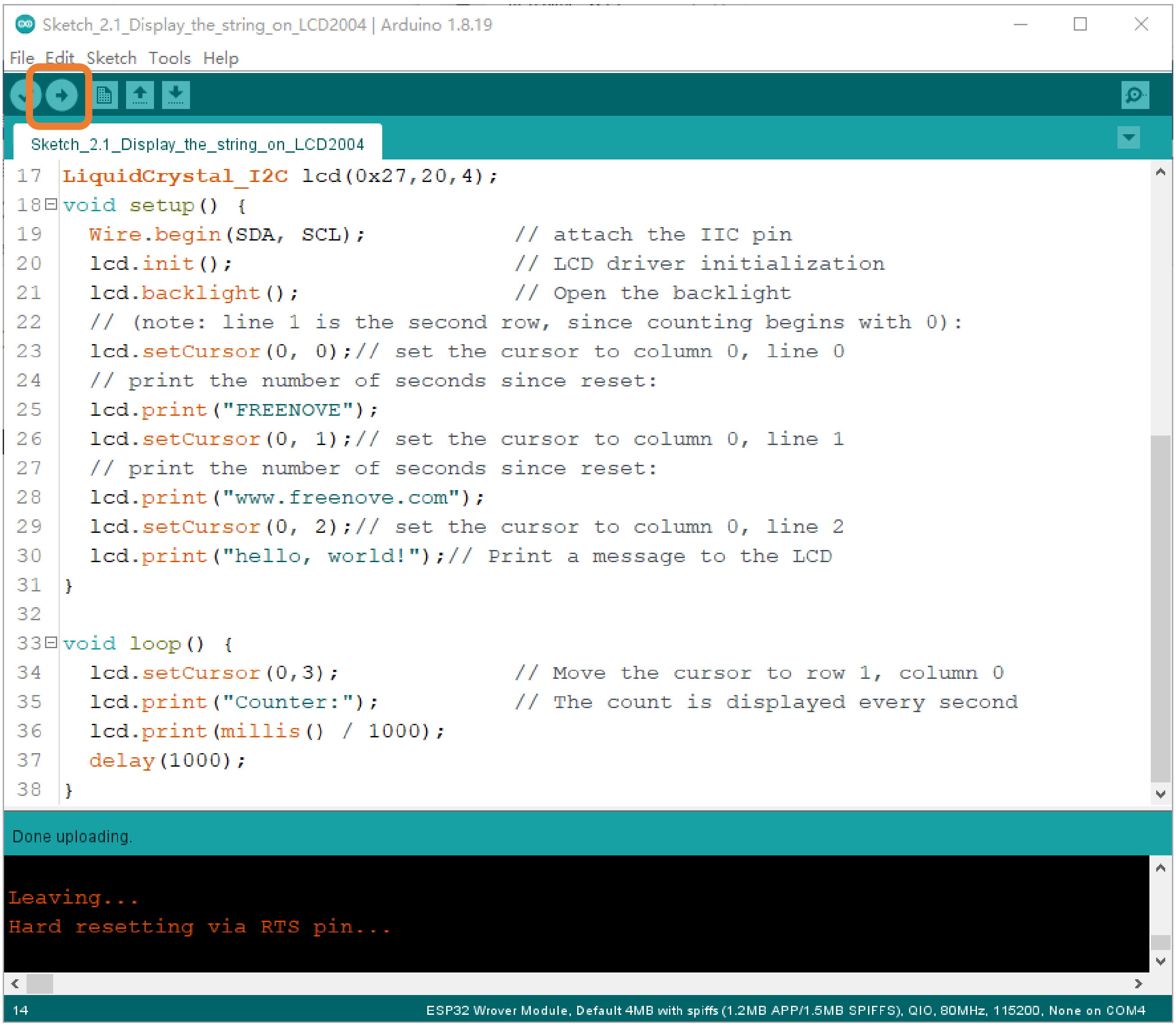
**LiquidCrystal I2C**

by **Frank de Brabander** Version **1.1.2** **INSTALLED**
**A library for I2C LCD displays.** The library allows to control I2C displays with functions extremely similar to LiquidCrystal library. THIS LIBRARY MIGHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.
More info

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 form 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

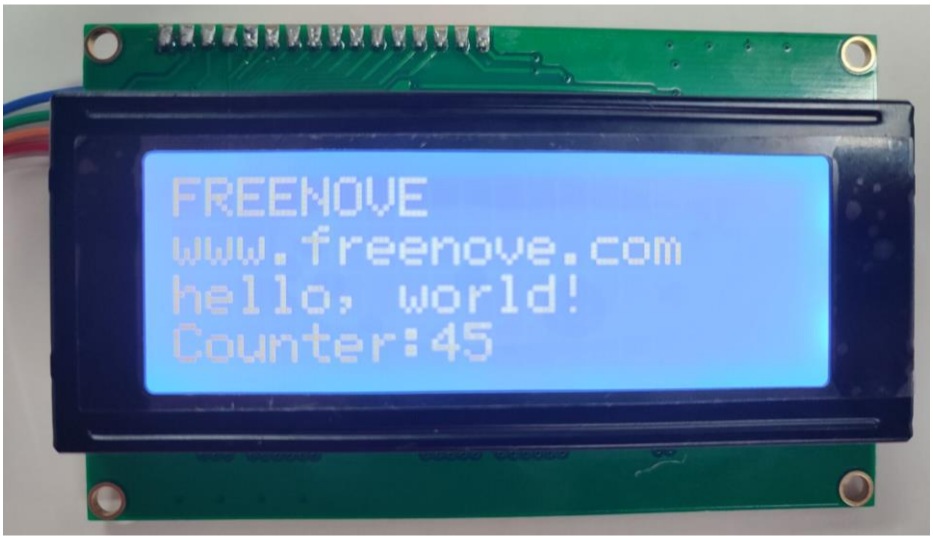
```
LiquidCrystal_I2C lcd(0x27,20,4);
void setup() {
  Wire.begin(SDA, SCL);           // attach the IIC pin
  lcd.init();                     // LCD driver initialization
  lcd.backlight();                // Open the backlight
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 0);// set the cursor to column 0, line 0
  // print the number of seconds since reset:
  lcd.print("FREENOVE");
  lcd.setCursor(0, 1);// set the cursor to column 0, line 1
  // print the number of seconds since reset:
  lcd.print("www.freenove.com");
  lcd.setCursor(0, 2);// set the cursor to column 0, line 2
  lcd.print("hello, world!");// Print a message to the LCD
}

void loop() {
  lcd.setCursor(0,3);             // Move the cursor to row 1, column 0
  lcd.print("Counter:");          // The count is displayed every second
  lcd.print(millis() / 1000);
  delay(1000);
}
```

Done uploading.

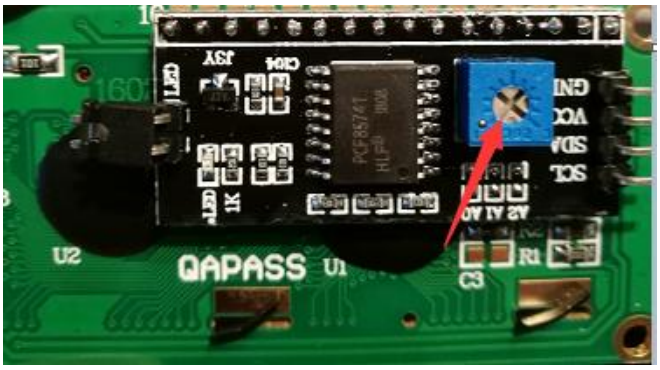
```
Leaving...
Hard resetting via RTS pin...
```

14
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 ESP8266 and the LCD2004 displays characters.



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:

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>

#define SDA 13                    //Define SDA pins
#define SCL 14                    //Define SCL pins

/*
 * note: If lcd2004 uses PCF8574T, IIC’s address is 0x27,
 *      or lcd2004 uses PCF8574AT, IIC’s address is 0x3F.
*/
LiquidCrystal_I2C lcd(0x27,20,4);

void setup() {
  Wire.begin(SDA, SCL);           // attach the IIC pin
  lcd.init();                     // LCD driver initialization
  lcd.backlight();                // Turn on the backlight
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 0);// set the cursor to column 0, line 0
  // print the number of seconds since reset:
  lcd.print("FREENOVE");
  lcd.setCursor(0, 1);// set the cursor to column 0, line 1
  // print the number of seconds since reset:
  lcd.print("www.freenove.com");
  lcd.setCursor(0, 2);// set the cursor to column 0, line 2
  lcd.print("hello, world!");// Print a message to the LCD

}

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

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

```
#include <LiquidCrystal_I2C.h>
#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.

```
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.

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

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

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

Print the number on the fourth line of LCD2004.

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

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.