



Arduino IDE1 development environment construction for ESP32

CONTENTS

1. Download Arduino IDE1 software installation package	3
2. Arduino IDE1 software installation	5
3. Arduino IDE1 software introduction	8
3.1. Menu bar	9
3.1.1. File menu	9
3.1.2. Edit menu	10
3.1.3. Sketch menu	11
3.1.4. Tools Menu	12
3.1.5. Help Menu	15
3.2. Tool bar	16
4. Install the Arduino-ESP32 core software library	16
4.1. Arduino IDE development board Manager is installed online	16
4.2. Manual Offline installation	18
5. Compile, download and run the ESP32 sample program	20
5.1. Install the USB-to-serial IC driver	20
5.2. Configure the development board	21
5.3. Compile, download, and run the program	25

1. Download the Arduino IDE1 software installation package

There are two versions of the Arduino IDE: Arduino IDE1 and Arduino IDE2. Arduino IDE1 is an old version that only supports Win7 and below systems, while Arduino IDE2 is a new version that supports Win10 and above systems. I'm only going to introduce the Arduino IDE1 here.

The Arduino IDE1 software installation package can be downloaded directly from the official website at:

<https://www.arduino.cc/en/software>

After entering the software download page of the official website, scroll down to find the Arduino IDE1 software installation package download column, as shown in the following picture:

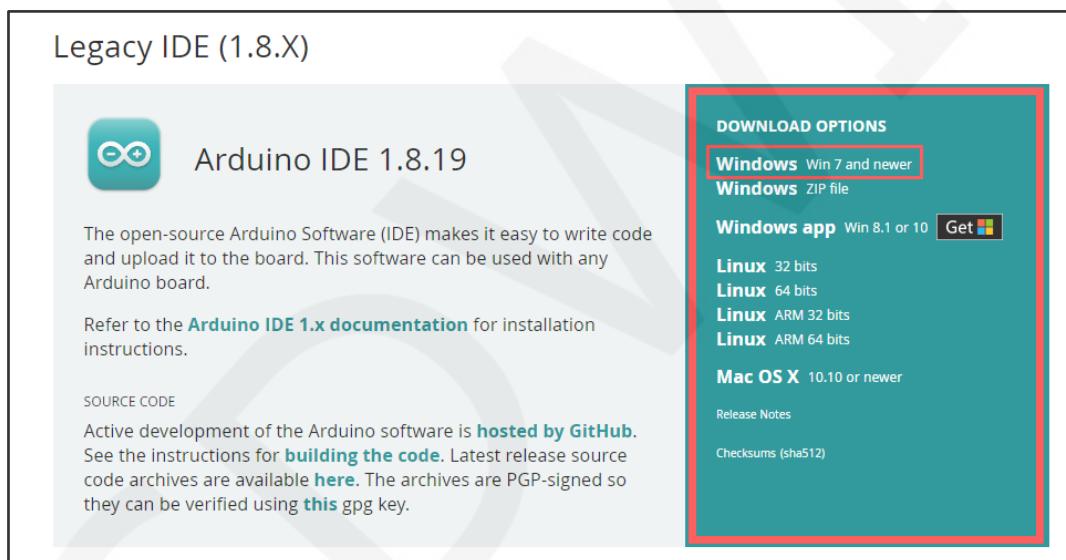


Figure 1.1 Arduino IDE1 software installation package download interface 1

Select the appropriate version for your computer system from the download options. For example, if you use Windows, click "**Windows Win 7 and newer**" to download. You can also download ZIP files.

After clicking on the download option, a screen will pop up asking whether to provide team financial support, as shown below:



Figure 1.2 Arduino IDE1 software installation package download interface 2

You can ignore this option and simply click the "**JUST DOWNLOAD**" button.

After clicking the button, an interface will pop up asking whether to enter an email to receive Arduino information. You can ignore it and directly click the "**JUST DOWNLOAD**" button, as shown in the picture below:

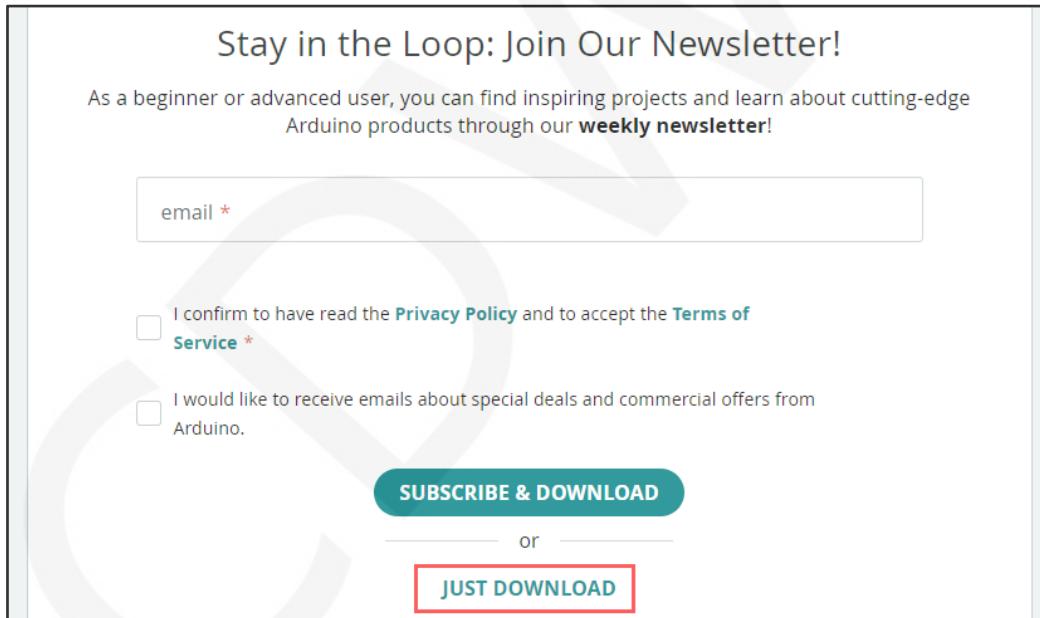


Figure 1.3 Arduino IDE1 software installation package download interface 3

After clicking the button, a "**New Download Task**" window will pop up. Click the "**Browse**" button to select the saving path of the software installation package, and then click the "**Download**" button to start downloading, as shown in the following picture:

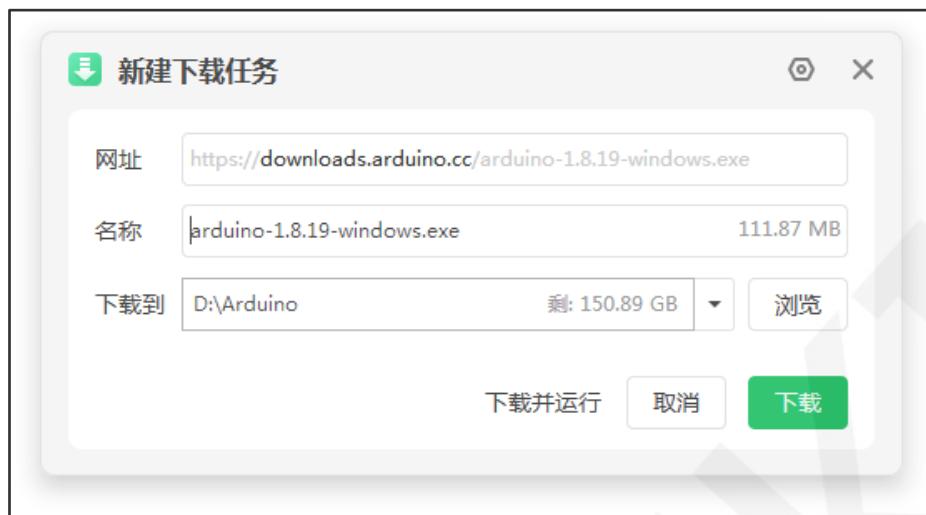


Figure 1.4 Arduino IDE1 software installation package download task

2. Arduino IDE1 software installation

Find the path to save the Arduino IDE1 software installation package, and then double-click the exe file to enter the program installation (if the window asking whether to run the file pops up, directly click the "Run" button), as shown in the following picture:

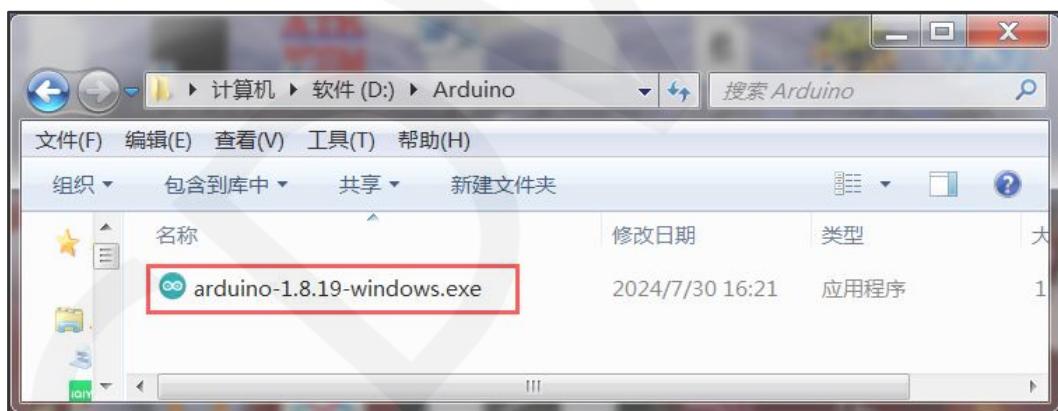


Figure 2.1 Arduino IDE1 software installation package exe file



Figure 2.2 Arduino IDE1 software installation license Agreement

Click the "I Agree" button to enter the installation content selection interface, select some basic software, drivers, and shortcuts to be installed. If you do not want to install a piece of software, remove the corresponding check mark. Generally, no operation is required, and you can keep the default, as shown in the following figure:

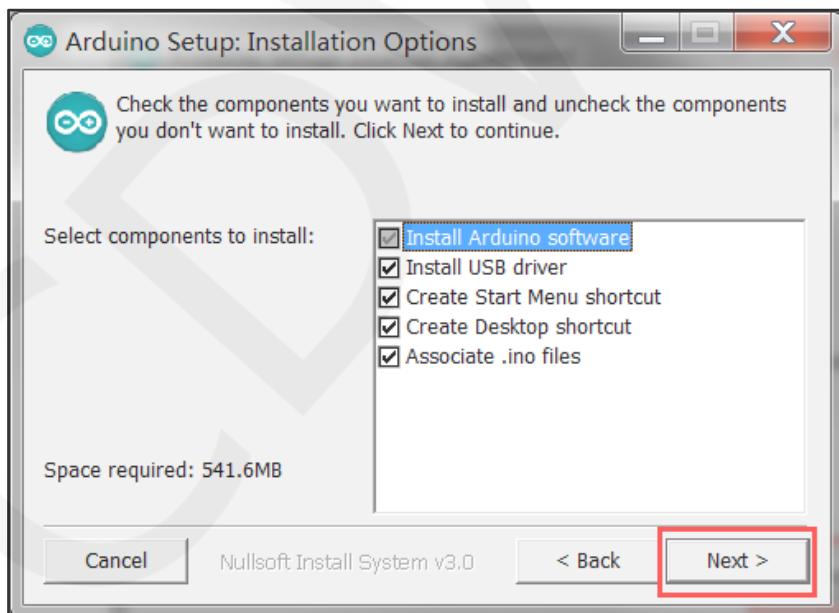


Figure 2.3 Selection of Arduino IDE1 software installation content

Then click "Next" button to enter the installation directory setting interface, click "Browse" button to select the installation directory or directly enter the installation directory, as shown in the following picture:

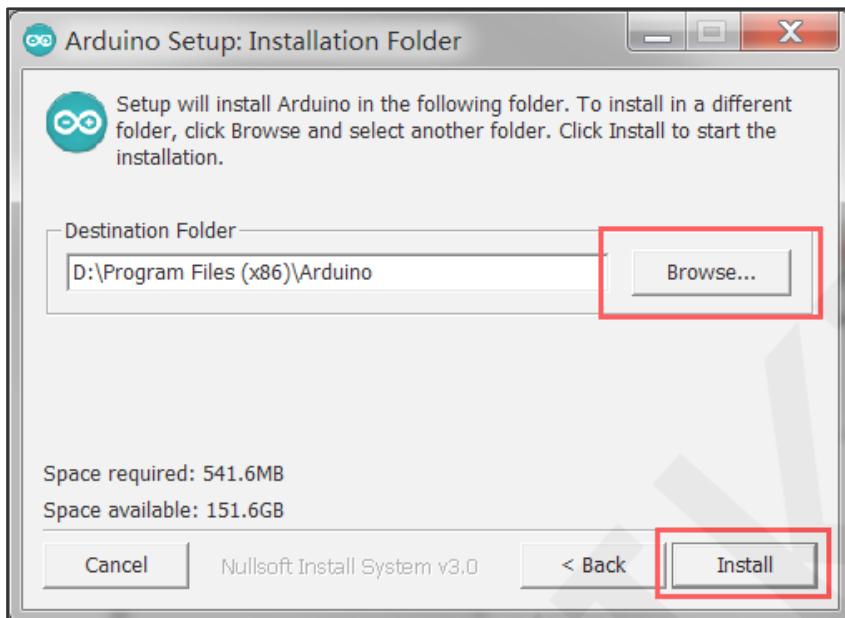


Figure 2.4 Selection of Arduino IDE1 software installation directory

Then click the "**Install**" button to start the installation, and you can see the change of the installation progress bar (if the window that needs to install some device software pops up during the installation, click the "**Install**" button directly), as shown in the following picture:

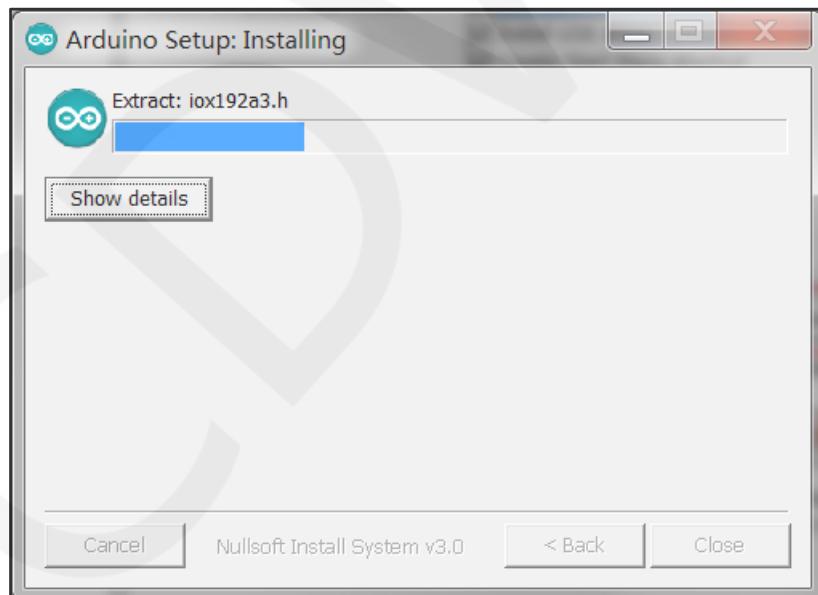


Figure 2.5 Arduino IDE1 software installation process

When the progress bar is Completed, the information "**Completed**" will be displayed, indicating that the software installation is complete. Click "**Close**" to finish the installation process, as shown in the following picture:

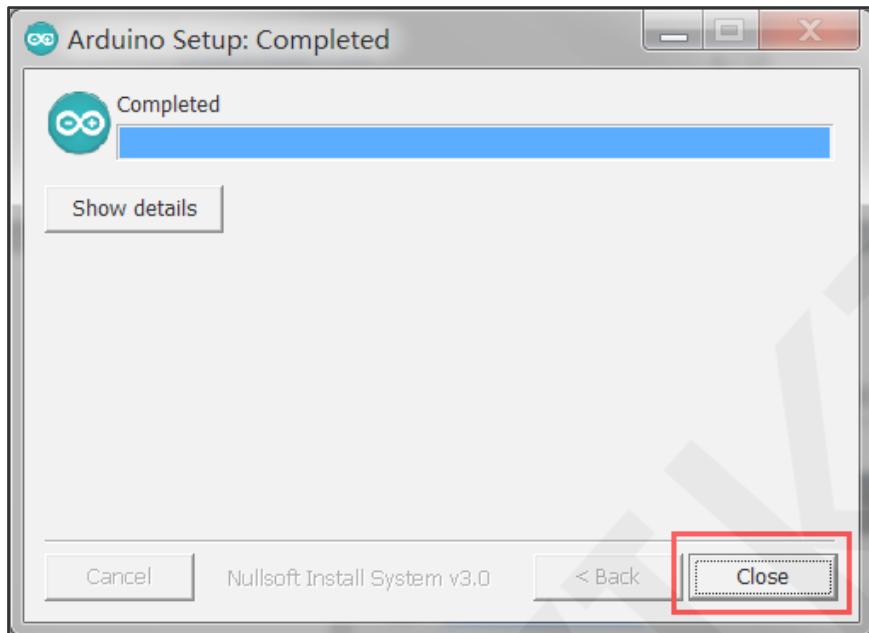


Figure 2.6 Arduino IDE1 software installation complete

3. Arduino IDE1 software introduction

Arduino IDE1 has the functions of project creation, program code editing, debugging, compilation, upload, software library management, development board management, etc.

The interface is shown as follows:

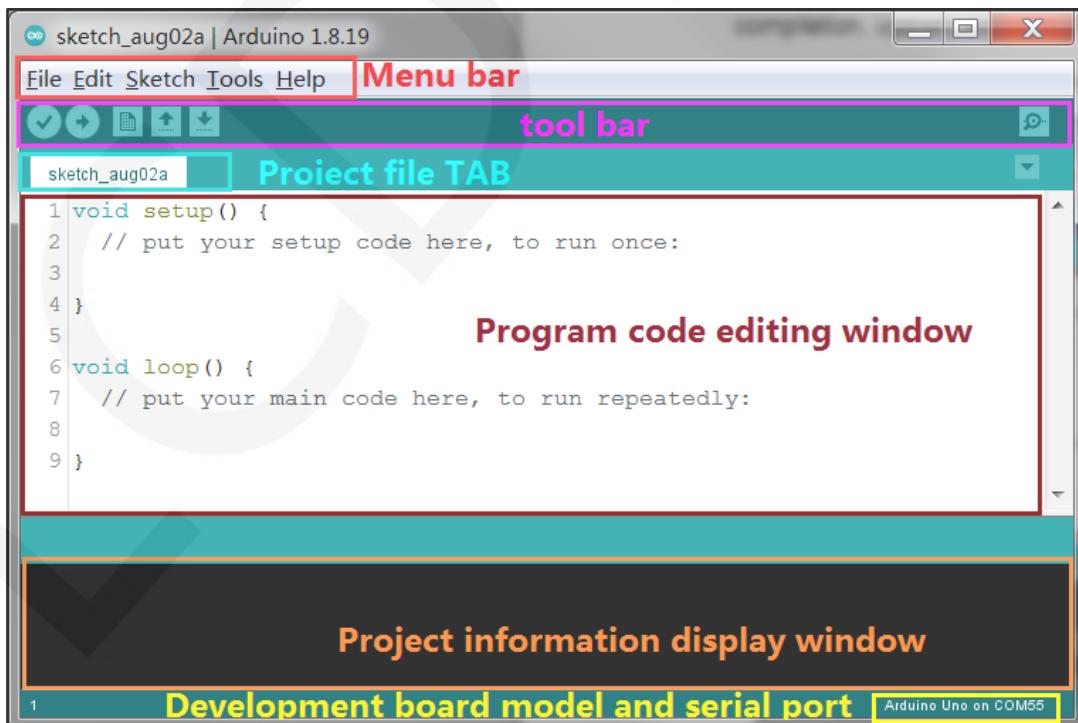


Figure 3.1 Arduino IDE1 interface

3.1. Menu bar

3.1.1. File menu

The menu contents of the menu bar file are shown as follows:

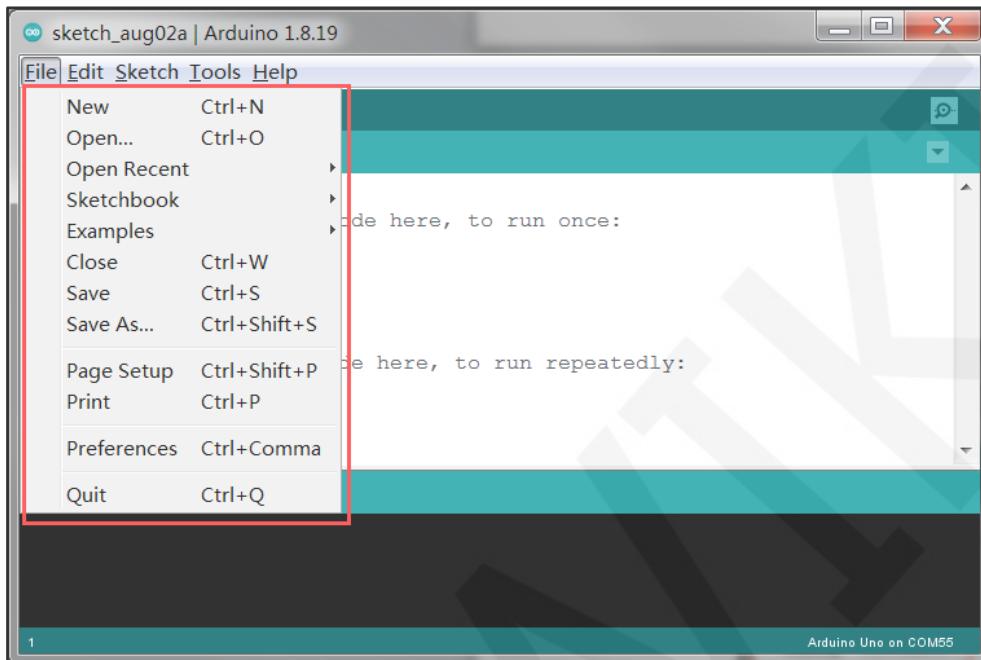


Figure 3.2 Arduino IDE1 file menu bar

Basically, it is to create, open and save the project. You can select the **Examples** option to open the third-party software library and the sample program of the development board core library. Here the preferences menu is highlighted, click on the "**Preferences**" option, as shown below:

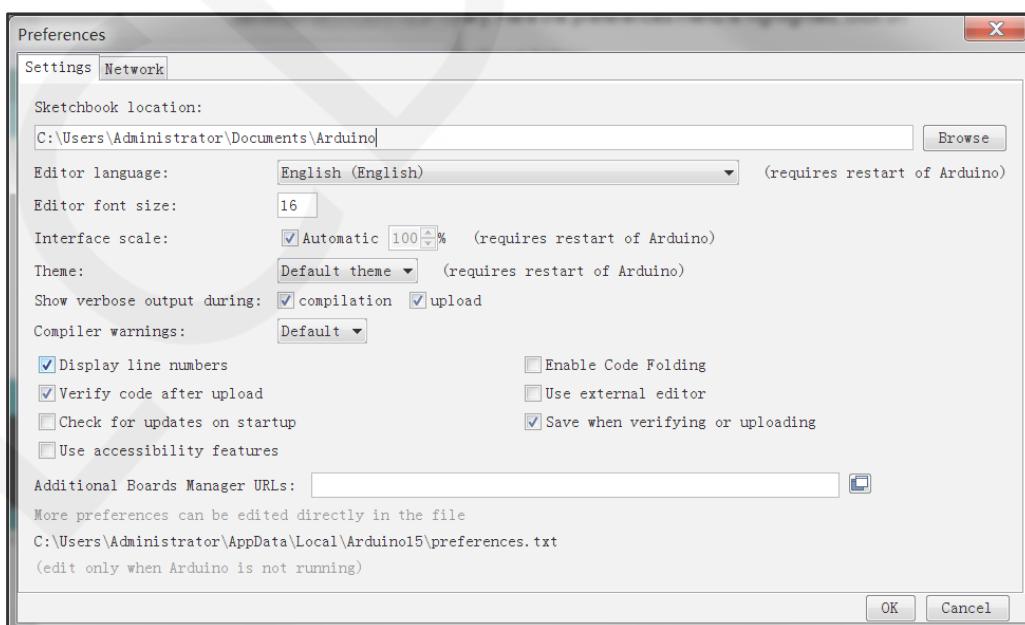


Figure 3.3 Arduino IDE1 Preferences menu

In the Preferences menu, you can make the following Settings:

- A. Sketchbook location, is the new project, the software default setting of the project save location, you can modify the location. The libraries directory in this location is used to store third-party software libraries.
- B. Editor language, you can set multiple languages, after the setting needs to restart the Arduino IDE software to take effect.
- C. Editor Interface Settings, keep the theme, code editing window and so on Settings.
- D. Compiler warnings, you can select "none", "default", "more", "all", etc., select "none", will not display any information, select "more" or "all", will display more comprehensive compilation information, but will slow down the compilation speed.
- E. Additional Boards Manager URLs, when you want to add the development board in the IDE's development board manager can not be found (non-Arduino official development board), you need to add the address of the development board here.
- F. more detailed setting options, you can open the "**preferences.txt**" file to view (directory see the preferences interface)

After setting, click the "**OK**" button to save.

3.1.2. Edit menu

Menu bar editing menu interface is shown as follows:

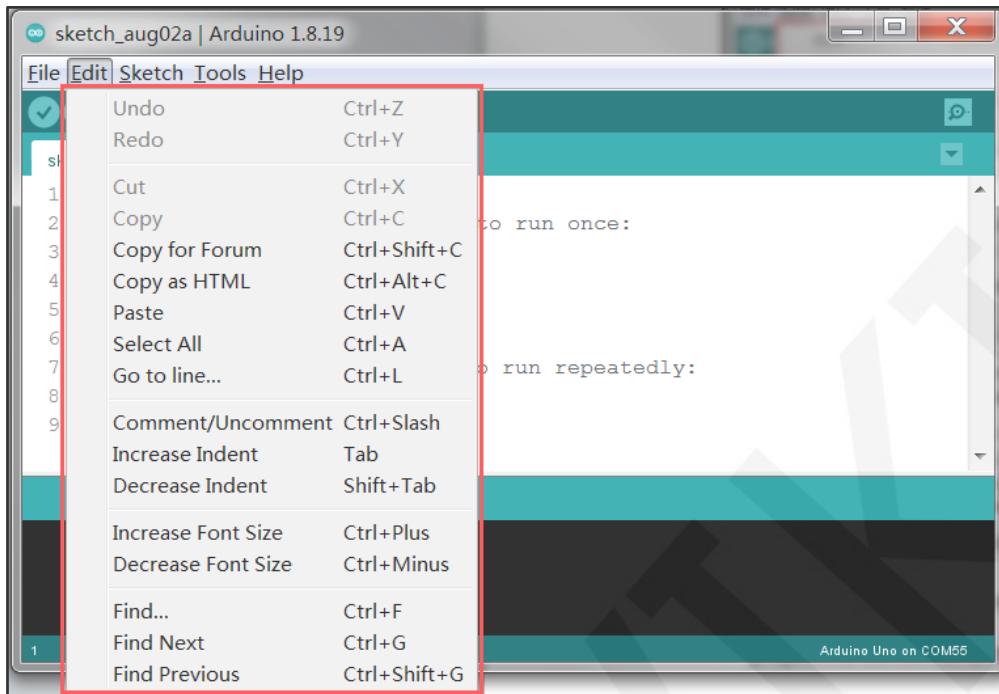


Figure 3.4 Arduino IDE1 Edit menu bar

The editing menu is mainly to copy, cut, undo, paste, find, modify the size of the project file content and other editing operations.

3.1.3. Sketch menu

Menu bar Sketch menu interface is shown as follows:

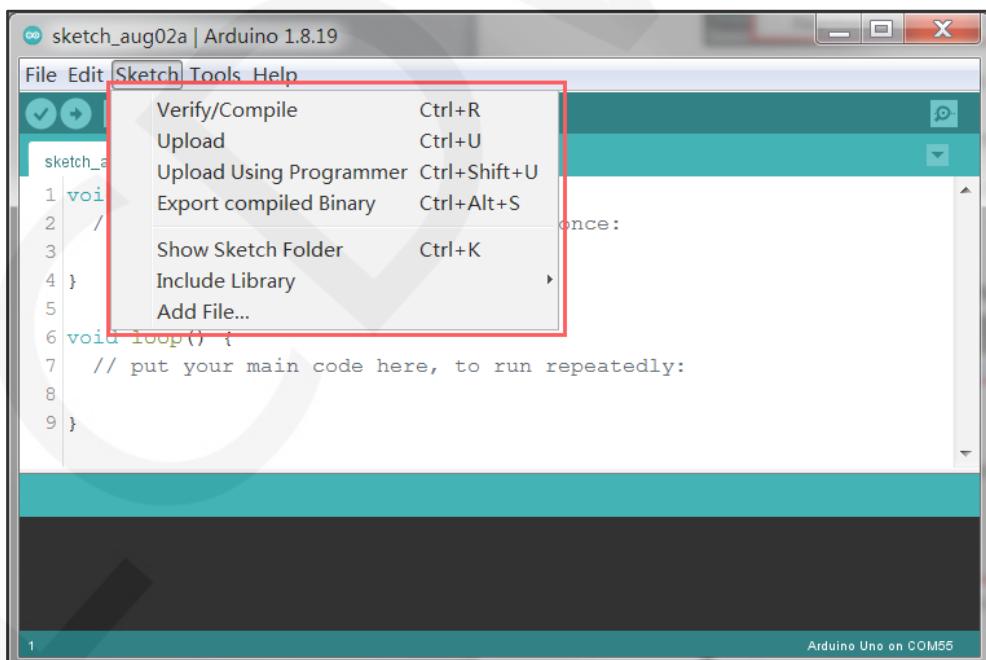


Figure 3.5 Arduino IDE1 project menu bar

The Sketch menu bar is mainly used to compile, upload, export and load library files for the project.

3.1.4. Tools Menu

The menu bar tools menu interface is shown as follows:

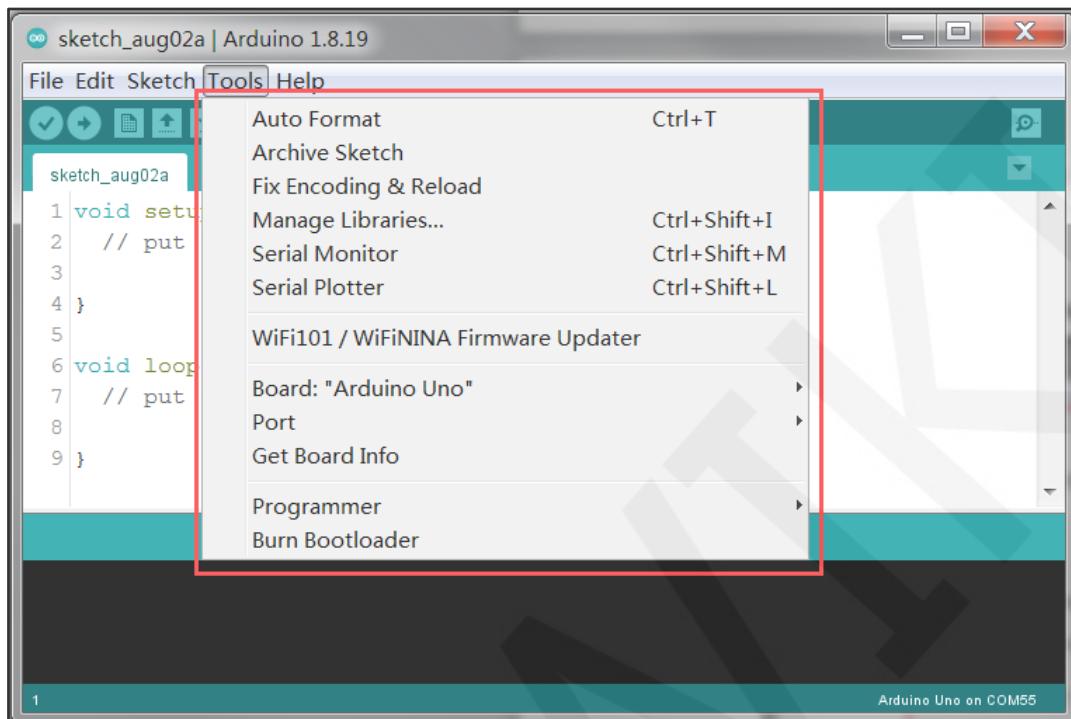


Figure 3.6 Arduino IDE1 tool menu bar

The following Settings can be made in the tools menu:

- Auto Format can automatically typeset the project code format, such as alignment and so on.
- Archive Sketch can ZIP package and save the entire project file.
- Fix Encoding & Reload You can restore the program to its previous state and reload it.
- Manager Libraries can search, download and install the third-party software library, and click to enter, as shown in the following figure:



Figure 3.7 Arduino IDE1 library Manager

In the library manager interface, you can filter the library according to the type and Topic of the software library, or you can directly output the name of the library to search the library. After the search is complete, select the library version and click the "Install" button to install. Finally, the library is installed in the "**C:\Users\Administrator\Documents\Arduino\libraries**" directory (this is the default directory, of course, you can change it in the **File -> Preferences** interface, the actual user name of the computer is in red font). Of course, you can also install the software library without using the library manager. You can download the library manually (you need to extract it) and copy it to the **C:\Users\Administrator\Documents\Arduino\libraries** directory.

E. Serial Monitor and Serial Plotter open the serial port interface, set the serial port baud rate, display the serial port output information, and send messages through the serial port. (Note that the development board should be connected, and the serial port can be used only after the serial port is correctly identified).

F. Board consists of two parts: Boards manager and Boards selection, as shown in the following figure:

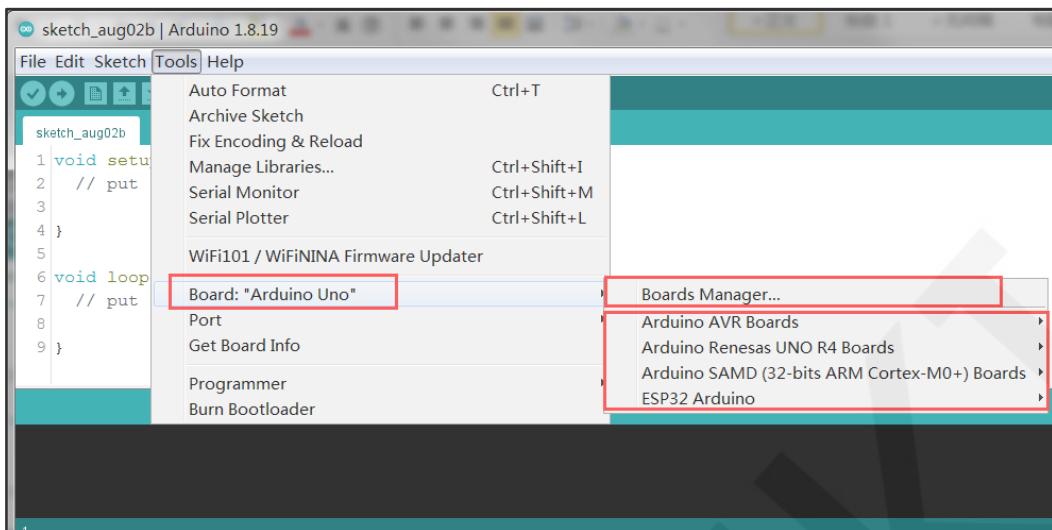


Figure 3.8 Arduino IDE1 development board menu

The Board option function is to select the currently used development board.

Once the selection is successful, it will be displayed after the option. If the development version used does not exist, the core software library of the development board needs to be installed, in which case the development board manager is used.

The Boards manager is to install the core software library of the development board. The interface is as follows: search the development board, select the version, and click the "Install" button to install the core software library of the development board.

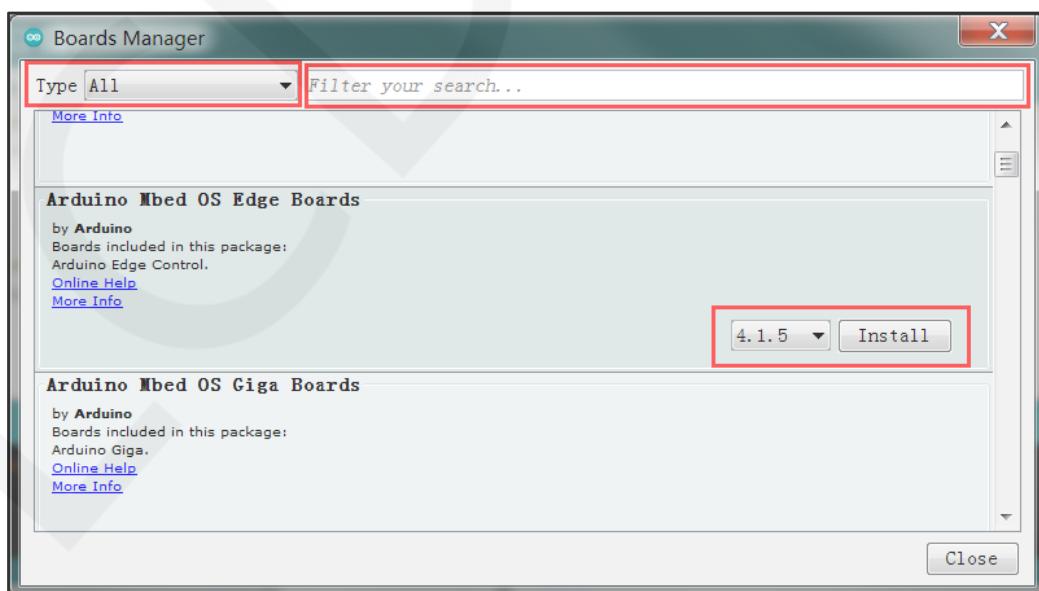


Figure 3.9 Arduino IDE1 development board Manager

G. Port is to select the serial port connected to the development board, as shown in the following figure, the serial port needs to be connected to the development board will be displayed:

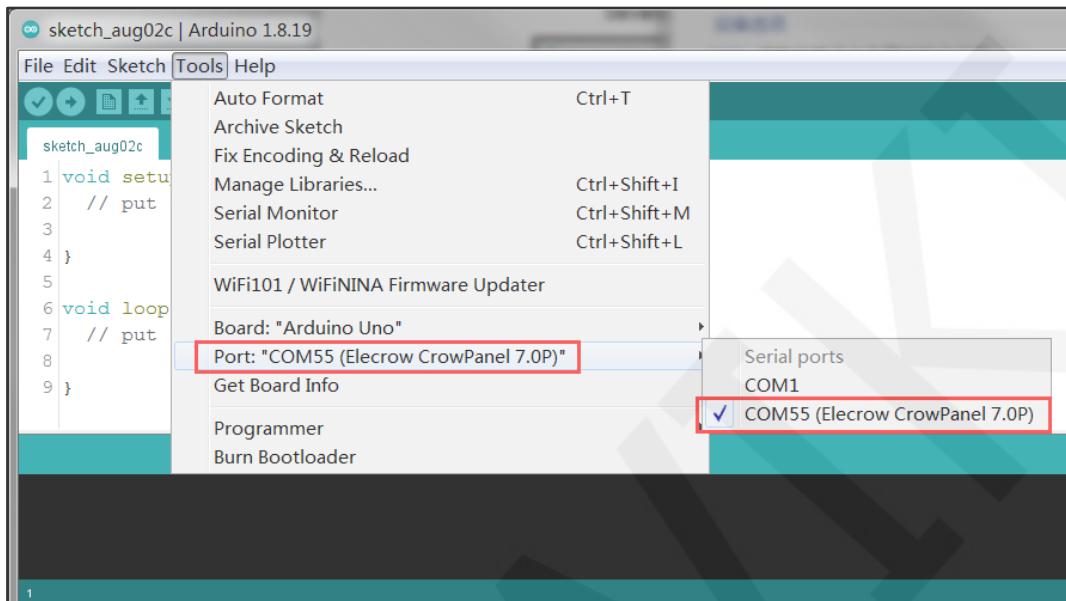


Figure 3.10 Arduino IDE1 port

Some of the other options are not used, so keep the default Settings.

3.1.5. Help Menu

The help menu interface is shown in the following figure, which mainly links to the software usage instructions on the official website.

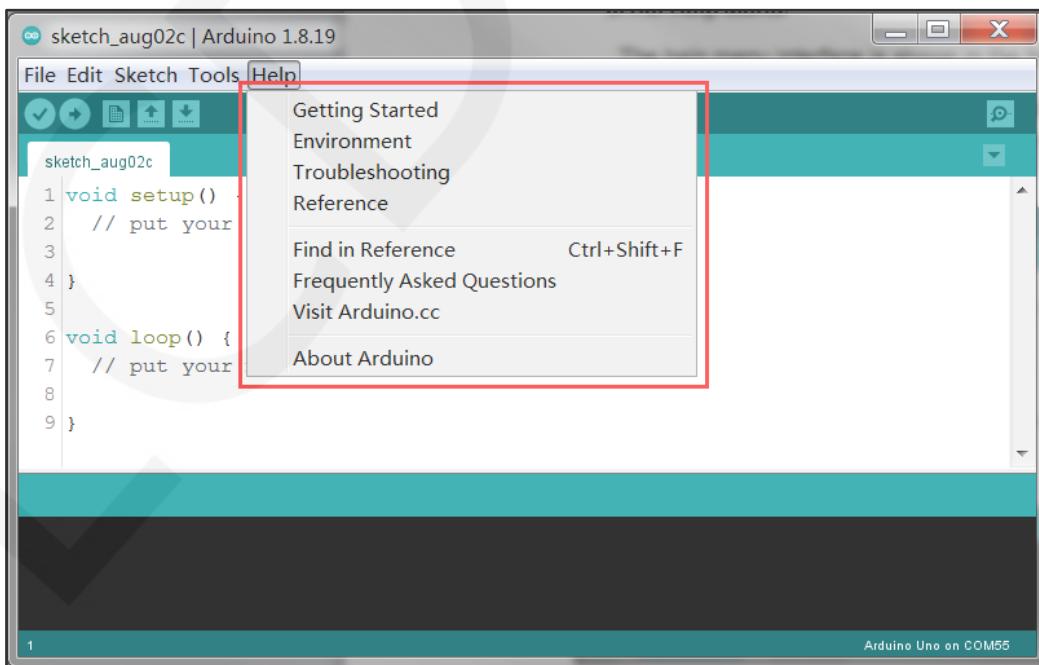


Figure 3.11 Arduino IDE1 Help menu

3.2. Tool bar

The tool bar interface is shown as follows:

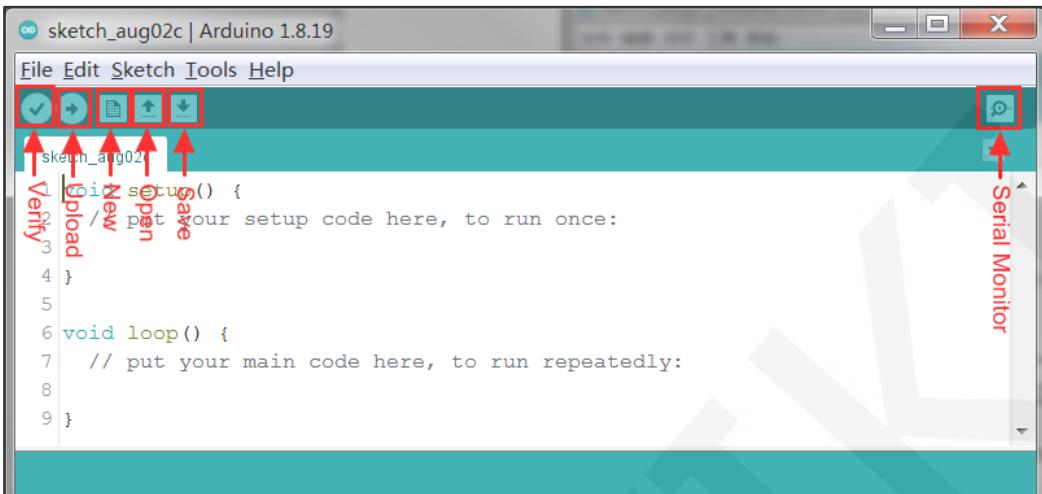


Figure 3.12 Arduino IDE1 toolbar

- A. **Verify:** compile and check whether the program is correct, if correct, compile through, generate binary files.
- B. **Upload:** the compiler generates binary files and uploads them to the MCU of the development board.
- C. **New, open, save:** create, open or save a project file.
- D. **Serial Monitor:** The serial port window is displayed.

4. Install the Arduino-ESP32 core software library

Arduino-esp32 Core software library is a plug-in of Arduino platform, which is the Arduino platform ESP32 chip. The software development provides low-level support. Because the Arduino IDE does not support ESP32 by default, you must install the Arduino-ESP32 core software library.

There are two ways to install the Arduino-ESP32 core software library: online installation of the Arduino IDE development board manager and manual offline installation.

4.1. Arduino IDE development board Manager is installed online

- A. Open the Arduino IDE software, click **Tools** -> **Board** -> **Boards Manager**, and input ESP32 in the search bar when the information of the development board is

loaded, the ESP32 search results will appear, as shown in the figure below.

Note: If the search is less than ESP32 core library, you may need to click **file -> preferences**, the attachment in the Additional Boards Manager URLs "https://espressif.github.io/arduino-esp32/package_esp32_index.json" in the input, Then follow the above steps to search.

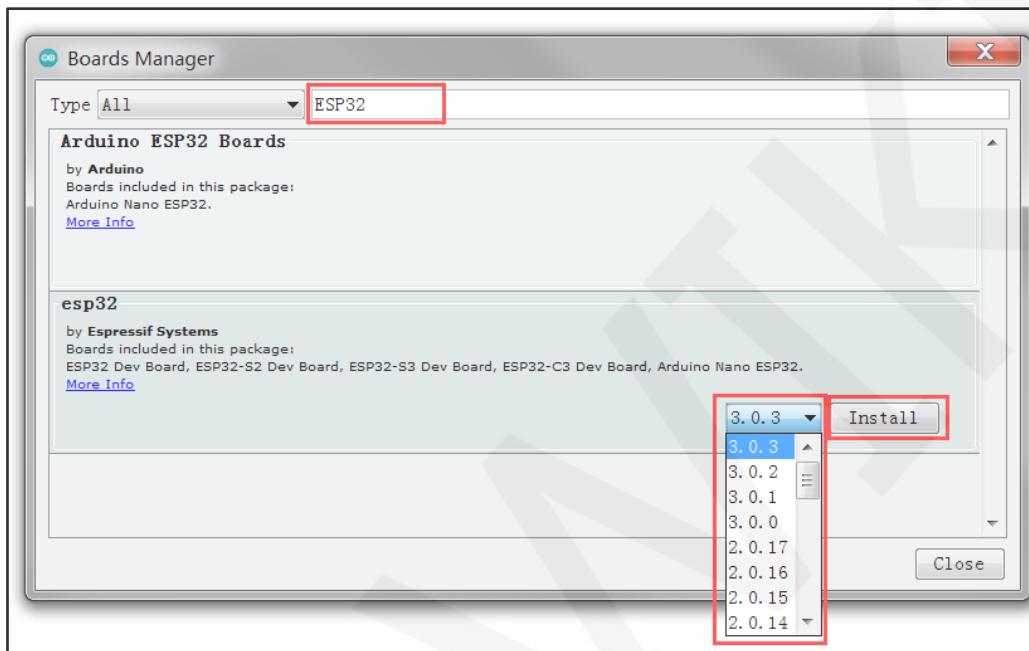


Figure 4.1 Arduino-ESP32 core software library search

B. select "esp32 by Espressif Systems", then select the version number, and finally click **Install**, as shown below.

Note: Version 3.0 was developed based on ESP32 idf 5.1 and version 2.0 was developed based on ESP32 idf 4.4. The two versions have different apis for Bluetooth, timer, I2S driver, LEDC driver, timer, and other software. Therefore, the example of version 2.0 involves the above apis. If the 3.0 version is used, an error will be reported. Pay attention to the version selection.

The installation takes a long time, and the download may fail during the installation. Therefore, you need to try the installation several times.

The downloaded installation files are compressed and saved in "**C:\Users\Administrator\AppData\Local\Arduino15\staging\packages**" directory (The red part is the actual user name of the computer, **AppData** directory is a hidden directory, you need to click the folder menu bar **tools ->**

Folder options -> View -> select Show hidden files, folders and drives, and then click OK to save)

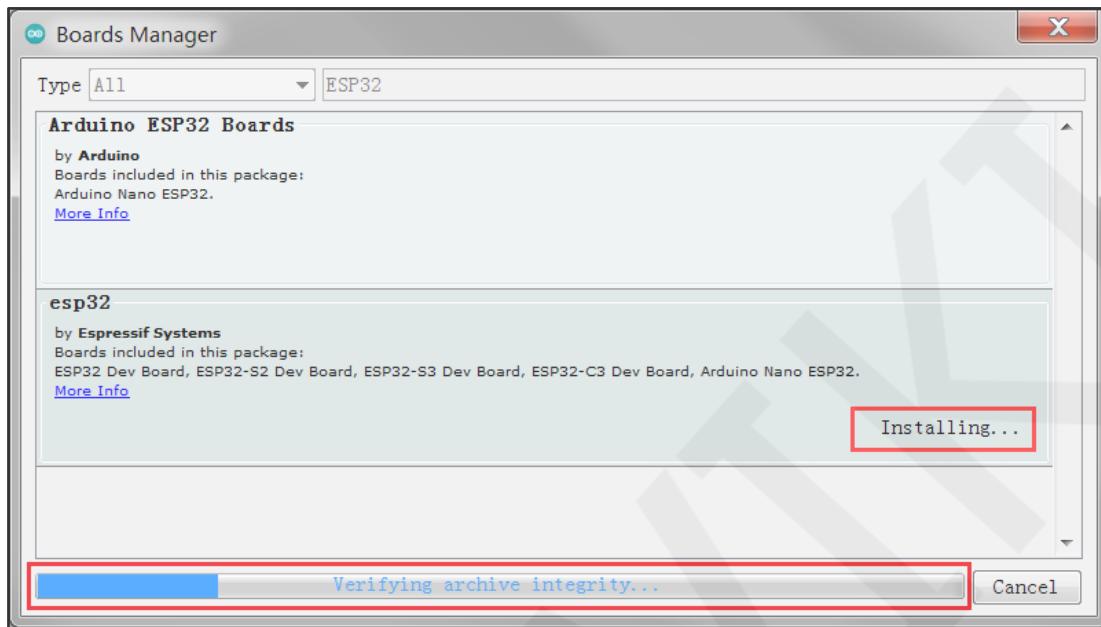


Figure 4.2 Installation of Arduino-ESP32 core software library

C. After installation, close the development board manager, click **Tools -> Board**, you can see the "**ESP32 Arduino**" option, click this option, you can see many ESP32 development boards, as shown in the following picture:

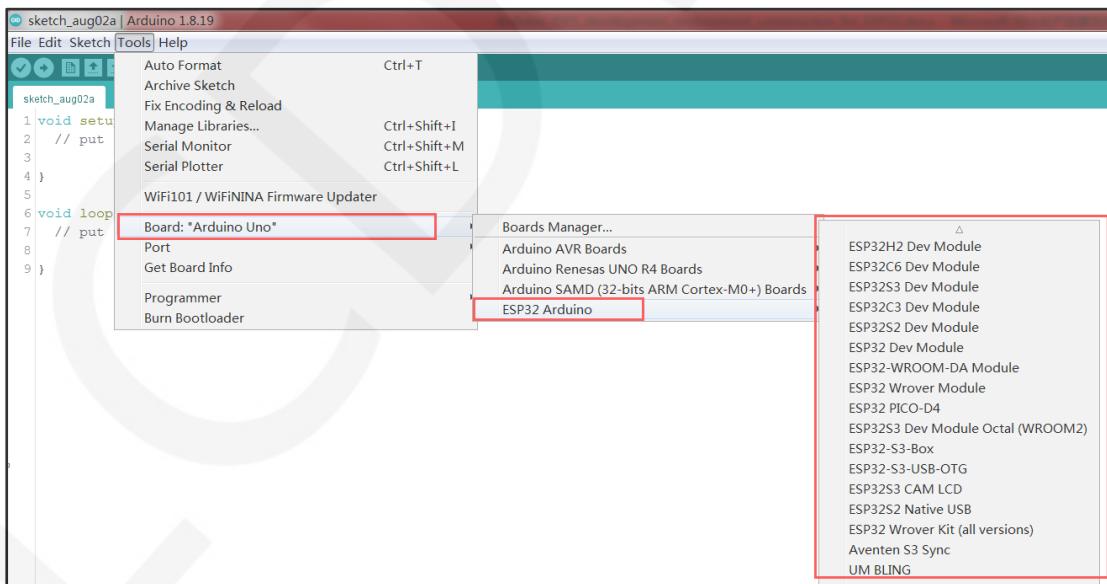


Figure 4.3 Selection of ESP32 development board

4.2. Manual Offline installation

The Developer Board Manager online installation is to download the installation

file zip from github, and then unzip the installation. In a poor network environment, access to github may fail, resulting in an online installation failure. A manual offline installation method is provided as follows:

Note: First make sure you have 7-Zip installed on your computer.

A. Download the installation file package from the following URL:

https://pan.baidu.com/s/1F2tc8uJY_llqK-KFrWT2MQ?pwd=3lqe

Among them, 3.0.3 is ESP32 core software library 3.0 version, 2.0.17 is ESP32 core software library 2.0 version, choose to download according to demand.

B. After the installation file is downloaded, double-click the file to pop up the decompression window, and enter

"C:\Users\Administrator\AppData\Local\Arduino15\packages" in the **Extract to** text box (the red part is the actual user name of the computer). Then click the "Extract" button to extract and install the file, as shown below:

Note: If the **esp32** folder already exists in the destination folder, you need to delete the **esp32** folder before decompressing the file.

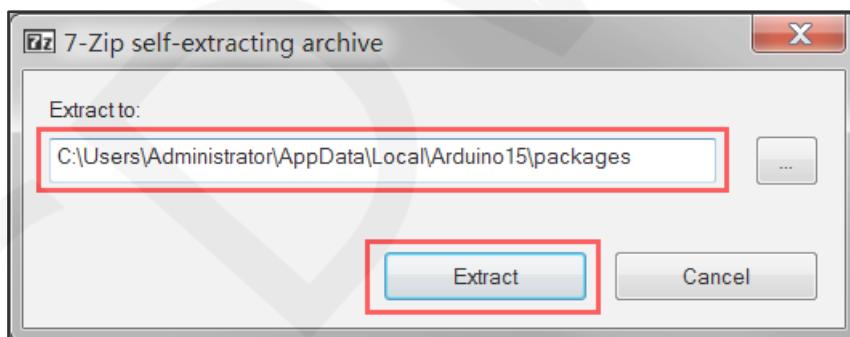


Figure 4.4 Offline installation of Arduino-ESP32 core software library

C. After the file is installed, open the Arduino IED software again, click **Tools -> Development Board**, you can see the "**ESP32 Arduino**" option, click this option, you can see many ESP32 development boards, which is consistent with the online installation step C.

5. Compile, download and run the ESP32 sample program

5.1. Install the USB-to-serial IC driver

When using the serial port to download the program, you need to install the USB to serial port IC driver on the computer, if no, the computer cannot identify the serial port. Different development boards use different USB-to-serial IC, and corresponding drivers need to be installed. The CH340C USB-to-serial driver IC is used here, so to install the CH340 driver, the steps are as follows (if already installed, you can skip the following steps) :

- A. Locate the **USB-SERIAL_CH340.zip** package in the "7-工具软件_Tool_software" folder and decompress it.
- B. go to the folder after decompression, double-click "**CH341SER.EXE**" executable program, pop up the installation window, and then click "Install" button to continue the installation, as shown in the following picture:



Figure 5.1 Installing the CH340C driver

- C. After the installation is successful, click the window OK button to exit. Connect the computer USB to the development board and power on, and then enter the computer device manager, you can see that the CH340 port is identified under the port, as shown in the following picture:



Figure 5.2 Identifying CH340 ports

5.2. Configure the development board

After you create or open an existing sample program in the Arduino IDE, you first need to configure the development board. The steps are as follows:

- Connect the development board to the USB port of the computer and power on, then select the model of the target development board, here select ESP32, click the "Tools" button, select **Board ->ESP32 Arduino->ESP32 Dev Module**, As shown in the picture below:

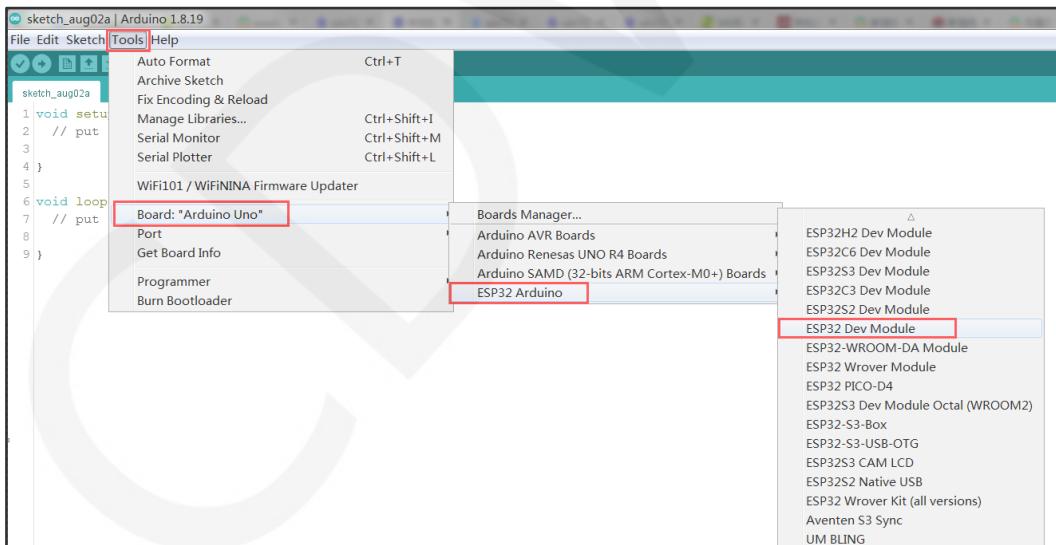


Figure 5.3 Select ESP32 development board model

- Click the "Tools" button, you can see the default configuration of ESP32 development board, as shown in the following picture:

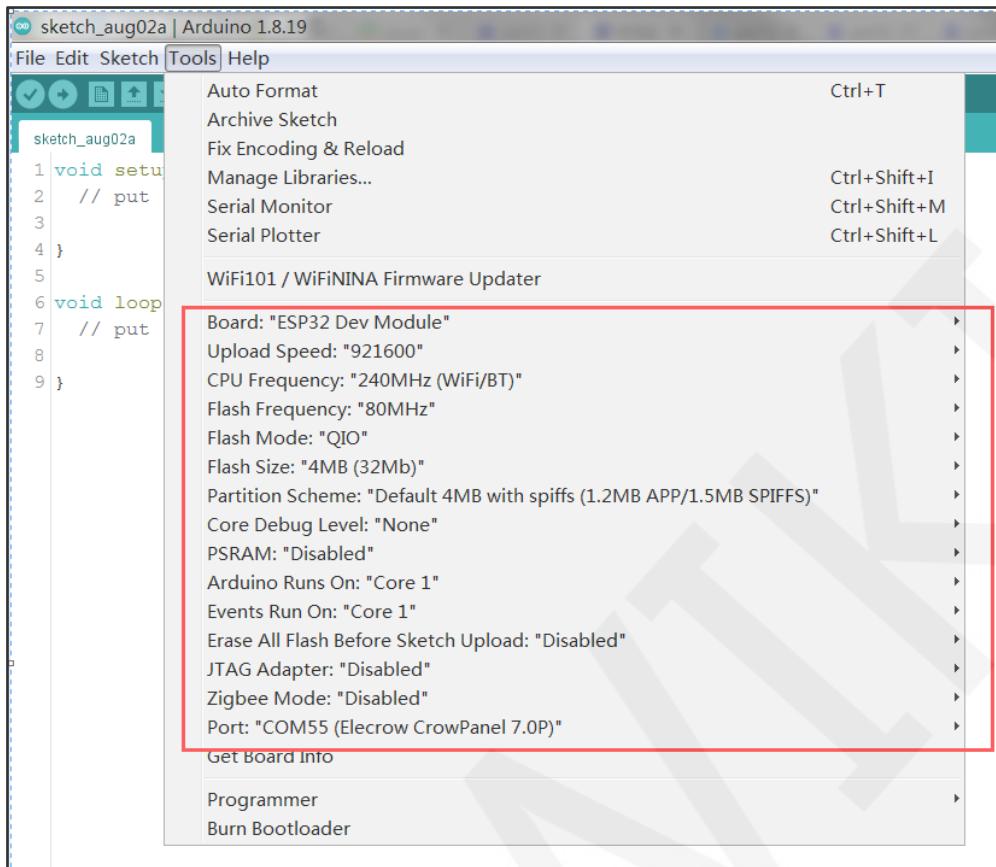


Figure 5.4 ESP32 development board configuration

Here are the configuration parameters:

Upload Speed: Code upload rate. Available parameters are 51200, 230400, 256000, 115200, and 921600. Select according to the maximum rate supported by the USB-to-serial port on the development board, such as the one used here. The CH340C supports a maximum rate of 2Mbps. Therefore, the maximum rate is 921600.

CPU Frequency: CPU clock frequency, optional parameters: 240MHz(WiFi/BT), 160MHz(WiFi/BT), 80MHz(WiFi/BT), 40MHz, 26MHz, 20MHz, 13MHz, 10MHz. Generally speaking, the higher the frequency, the higher the power consumption, which can be selected according to the demand. Power consumption is not considered here, direct selection. Select a maximum frequency of 240MHz for optimal performance. Points to note: 240MHz, 160MHz, These three frequencies of 80MHz can ensure the normal operation of WiFi and BT, and other frequencies cannot be guaranteed. Verify

the normal operation of WiFi and BT, and only ensure the basic functions of the CPU.

Flash Frequency: Clock frequency of the Flash SPI bus mounted on the ESP32.

Optional parameters are: 80MHz,40MHz. In order to improve the Flash read and write speed, the high frequency of 80MHz is generally selected.

Flash Mode: Flash communication mode mounted on the ESP32. Available parameters are QIO and DIO. QIO uses four SPI data lines for Flash writing and reading. DIO uses two SPI data lines for Flash writing and reading. Select the connection mode based on the actual Flash connection mode. Here we use 4 SPI data lines for Flash write and read, so QIO is selected.

Flash Size: Flash capacity mounted on ESP32. The value can be 4MB(32Mb), 8MB(64Mb), or 4MB(32 MB),2MB(16Mb), 16MB(128Mb). The value is selected based on the actual capacity of the Flash.The Flash is 4MB, so choose 4MB(32Mb).

Partition Scheme: Method of mounting Flash space partition on ESP32. To make better use of Flash,The Arduino IDE has designed more than a dozen partitioning methods, which are not introduced here. If you are interested, you can learn by yourself. The Flash used here is 4MB. Generally, "Default 4MB with spiffs(1.2MB APP/1.5MB SPIFFS)" is selected. If there are many project files, the compiled binary file is relatively large. You can select "Huge APP(3MB No OTA/1MB SPIFFS)".

Core Debug Level: Arduino kernel debug log level, output through the serial port.

The options are as follows: None,Errors, Warn, Info, Debug, and Verbose. Among them:

None: No debug log is displayed;

Error: Only debug logs with error levels are output;

Warn: Only debug logs of warning or higher levels are

generated;

Info: Only debugging logs of the information and higher levels are displayed;

Debug: Displays only debug logs of a higher level;

Verbose: Displays debugging logs of all levels in kernel debugging;

In general, you do not need to pay attention to kernel debug logs unless you are developing some kernel-related functionality. So the choice here is nothing.

PSRAM: The options are Disabled and Enabled. Some ESP32 divide

In addition to the built-in SRAM, external PSRAM is attached for memory expansion. In this case, select Enabled. Some ESP32s only have internal SRAM. In this case, select Disabled. The ESP32 used here does not have PSRAM mounted, so select Disabled.

Arduino Runs On: Configure the ESP32 kernel that runs the Arduino Core task code. The optional parameters are Core0 and Core1. The ESP32 has two cores, Core0 and Core1, each of which can run different code tasks. You can choose according to the situation, the default is Core1.

Event Runs On: Configure the ESP32 kernel that the Arduino interrupt event runs on. The optional parameters are Core0 and Core1. You can choose according to the situation, the default is Core1. The kernel configured here can be the same as that configured in Arduino Runs On, or it can be different. When configured in the same way, the ESP32 power consumption can be reduced. When configured differently, the efficiency of the program can be improved.

Erase ALL Flash Before Sketch Upload: Specifies whether to erase the entire Flash when uploading code. The options are Disabled and Enabled. If Disabled is selected, full erasure is not required, and

if Enabled is selected, full erasure is required. If full erase is selected, the rate of code upload slows down. In addition, frequent full erase affects the service life of the Flash. Therefore, Disabled is selected.

JTAG Adapter: The options are Disabled, FTDI Adapter, and ESP USB Bridge.

Debugging code with JTAG is easier, but the Arduino IDE does not support ESP32 debugging. So select "Disabled".

PORT: Select the serial port number connected to the ESP32 development board.

Generally, it will be automatically identified.

5.3. Compile, download, and run the program

The example that comes with the Arduino-ESP32 core software library is used as a demonstration. You can also create a new project to compile, download and run, and you can open an already completed project to operate.

A. Open Arduino IDE, click the "File" button, and select

Examples->ESP32->ChipID->GetChipID, as shown in the following picture:

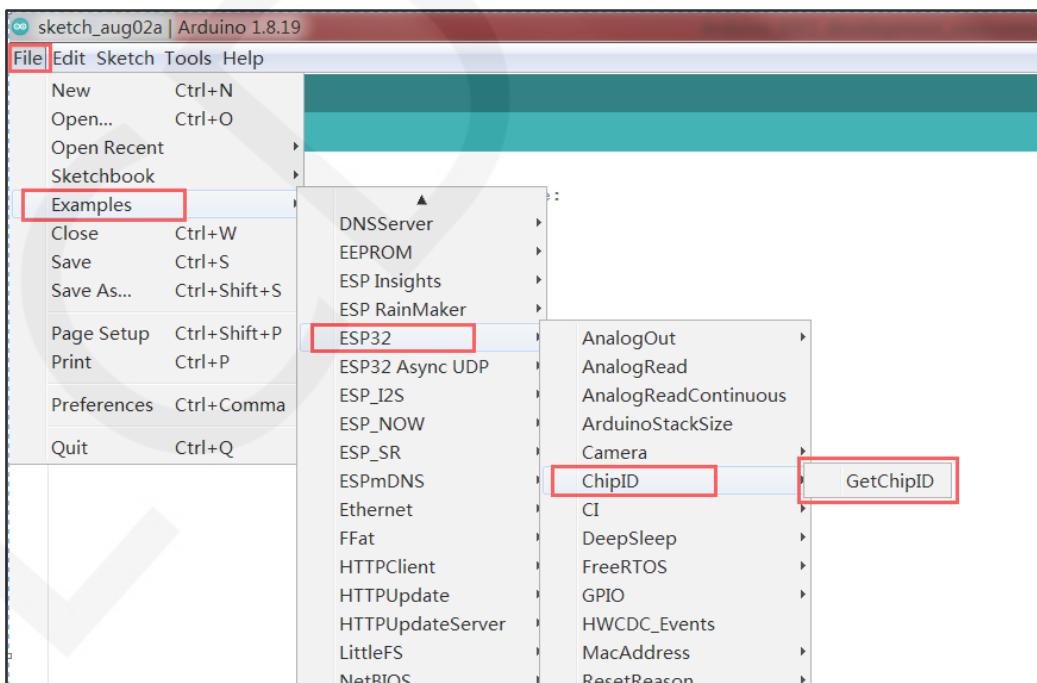


Figure 5.5 ESP32 sample program

B. click the "Upload" button, you can see "**compiling Sketch...**" Prompt, as shown in the picture below:

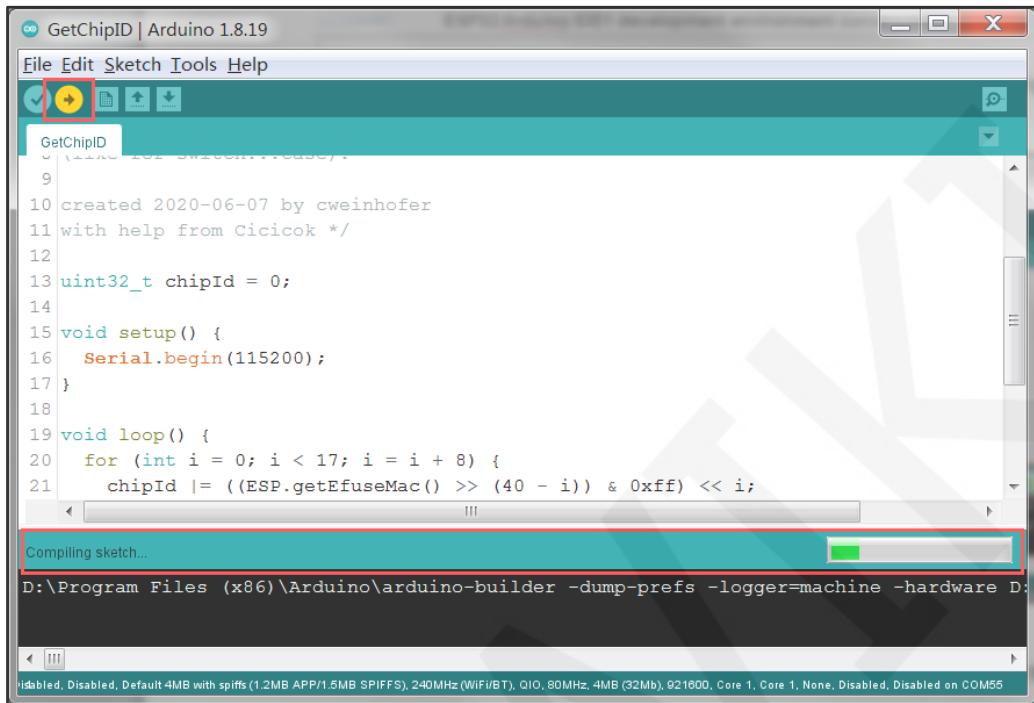


Figure 5.6 ESP32 project compilation

C. After the compilation is successful, "**Uploading...**" will be displayed. At the same time, the information output window will output a compilation success message, as shown in the following figure:

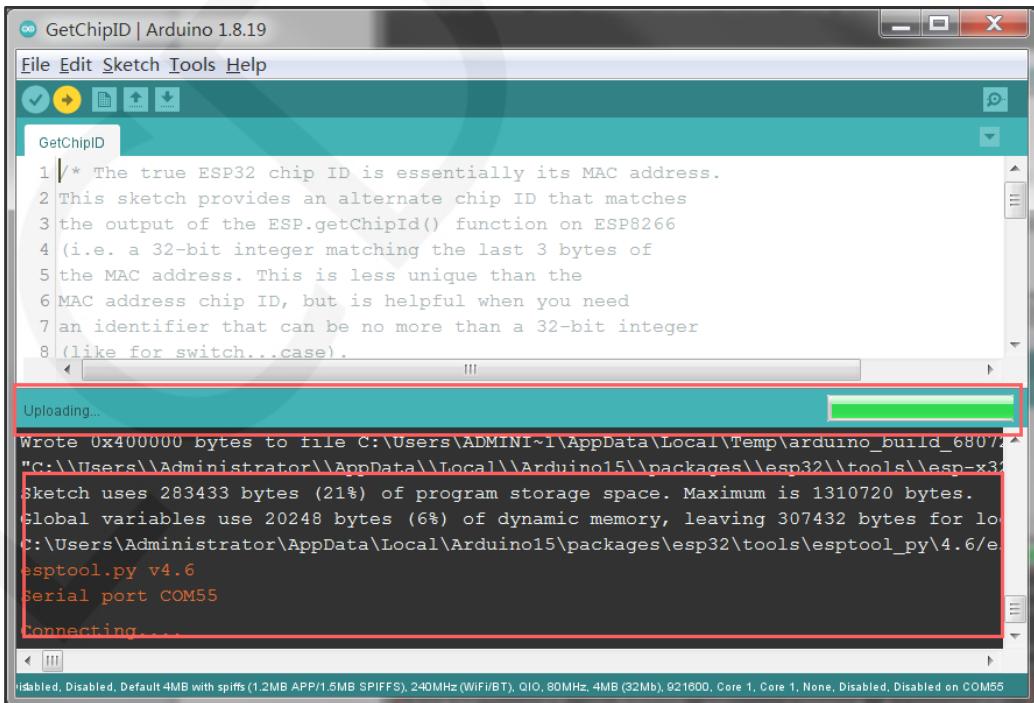


Figure 5.7 ESP32 project compiled successfully

D. After the upload is successful, the message "**Upload Success**" will appear, and the information output window will output the uploaded information and the program operation prompt, as shown in the following figure:

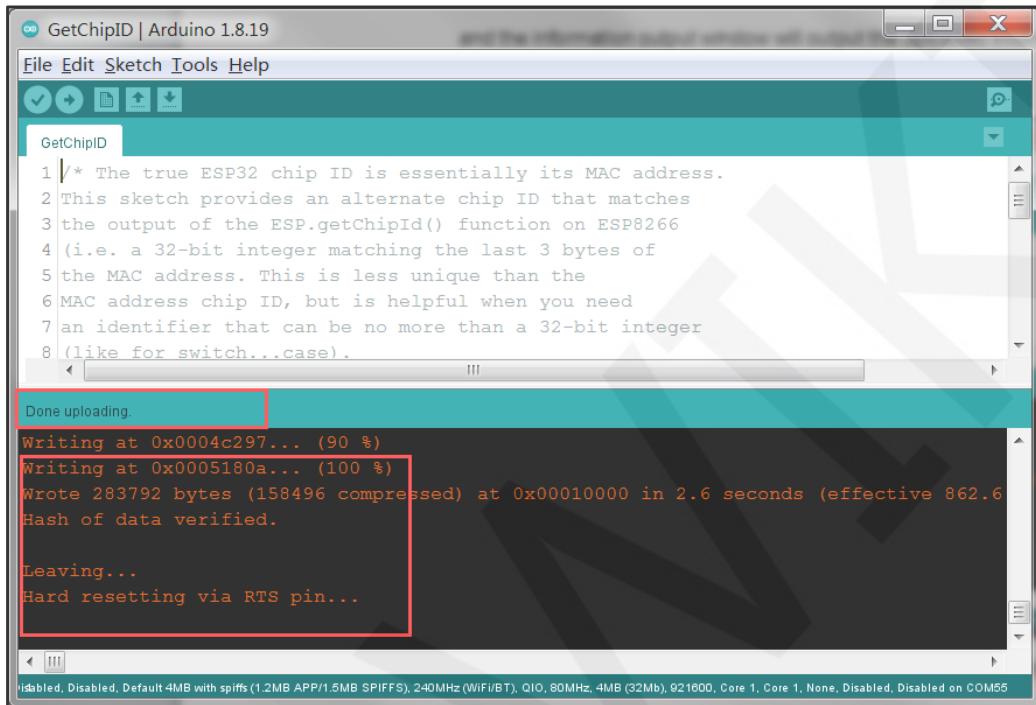


Figure 5.8 ESP32 project uploaded and running successfully

E. click the menu bar **tool** -> **Serial port monitor**, the serial port interface pops up, set the baud rate to 115200, you can see the serial terminal information output, then the program runs successfully, as shown in the following picture:

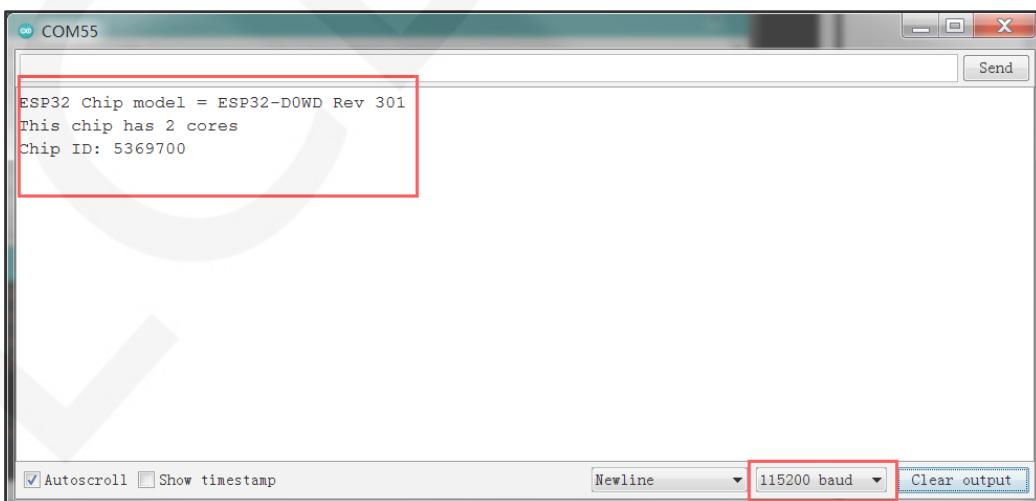


Figure 5.9 Serial port output of ESP32 program