

ESP32 Arduino IDE Development environment

Contents

1. Arduino IDE Software installation package download.....	3
2. Arduino IDE Software installation.....	5
3. Arduino IDE Software introduction.....	10
3.1. Menu bar.....	10
3.1.1. File.....	10
3.1.2. Edit.....	13
3.1.3. Sketch.....	13
3.1.4. Tool.....	14
3.1.5. Help.....	19
3.2. Tool.....	20
4. Install the Arduino-ESP32 core software library.....	21
4.1. Arduino IDE board Manager Installation.....	21
5. Compile, download, and run the ESP3-S3 example program.....	24
5.1. Configure the development board.....	24
5.2. Compile, download and run the program.....	28

1. Arduino IDE Software installation package download

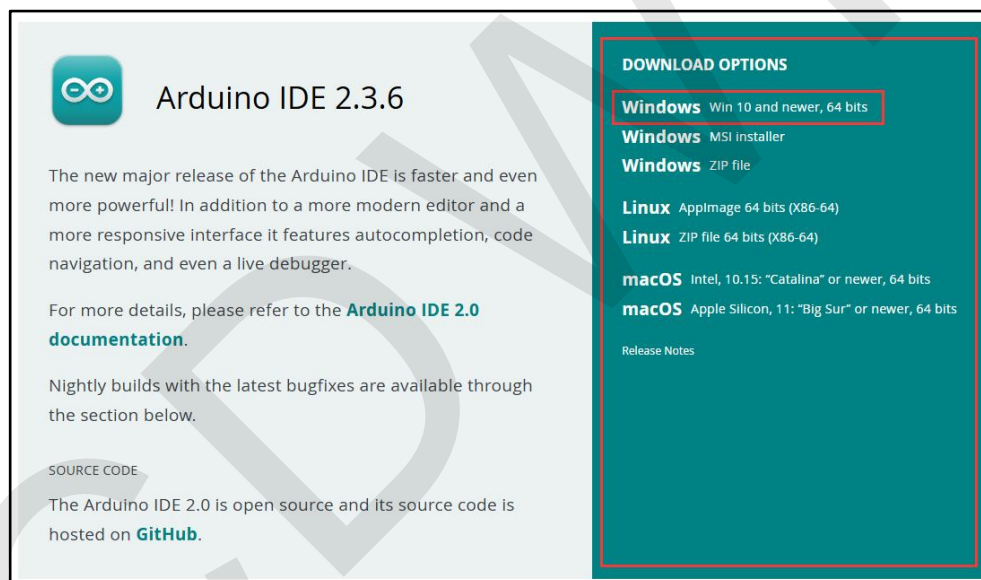
There are two Windows versions of the Arduino IDE: Arduino IDE1.x and Arduino IDE2.x. The Arduino IDE1.x version only supports Win and below, and it is very old and not maintained by the official. The Arduino IDE2.x version supports

Win10 and above, and it is the latest version and official main product. Here we only introduce the Arduino IDE2.x version.

The Arduino IDE2.x software installation package can be downloaded directly from the official website, and address is:

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

After entering the official website software download page, find the Arduino IDE software installation package download column, as shown in the following figure:



Picture1.1 Arduino IDE software installation package download interface 1

From the download options, select the appropriate version for your computer system. Here, Windows system is used, so simply click on **"Windows Win 10 and, 64 bits"** to download. You can also download the ZIP compressed package file and MSI download.

After clicking the download option, a pop-up interface will appear asking if you want to provide team funding support, as shown in the following figure:



Download Arduino IDE & support its progress

Since the 1.x release in March 2015, the Arduino IDE has been downloaded **84,926,804** times — impressive! Help its development with a donation.

\$3 \$5 \$10 \$25 \$50 Other

CONTRIBUTE AND DOWNLOAD

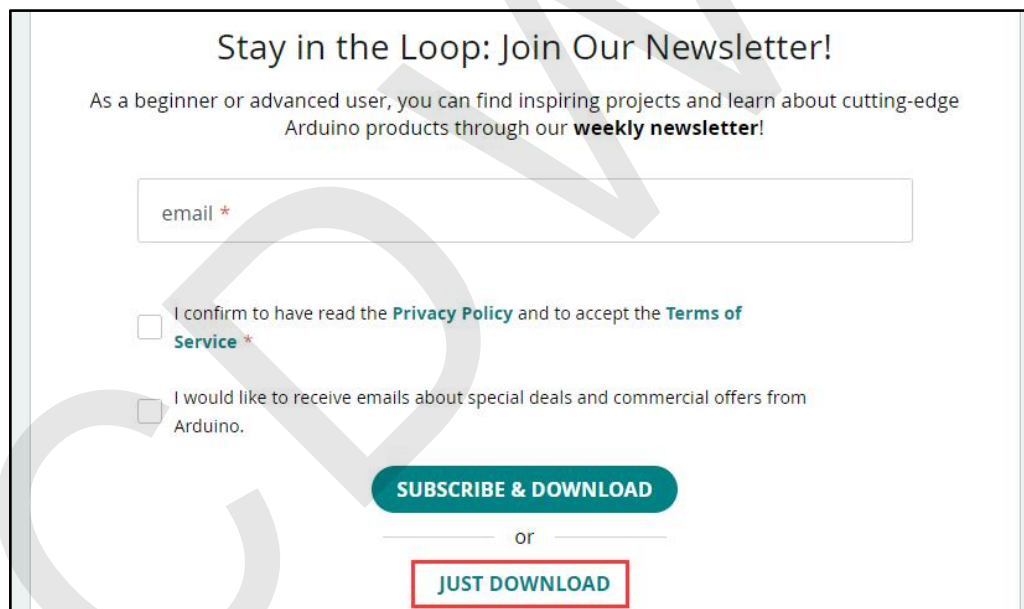
or

JUST DOWNLOAD

Picture1.2 Arduino IDE software installation package download interface 2

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

After clicking the button, a pop-up interface will appear asking if you want to enter your email to receive Arduino information. You can ignore it and directly click the "JUST DOWNLOAD" button, as shown in the following figure:



Stay in the Loop: Join Our Newsletter!

As a beginner or advanced user, you can find inspiring projects and learn about cutting-edge Arduino products through our **weekly newsletter!**

email *

☐ I confirm to have read the [Privacy Policy](#) and to accept the [Terms of Service](#) *

☐ I would like to receive emails about special deals and commercial offers from Arduino.

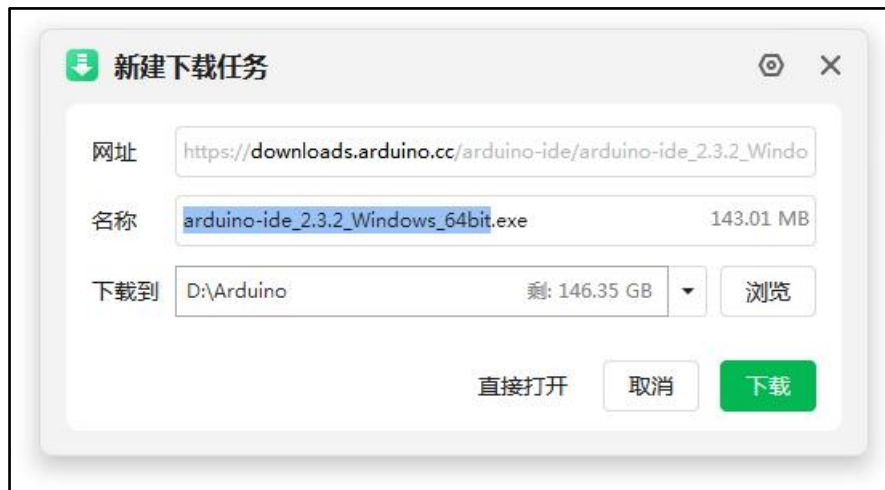
SUBSCRIBE & DOWNLOAD

or

JUST DOWNLOAD

Picture1.3 Arduino IDE software installation package download interface 3

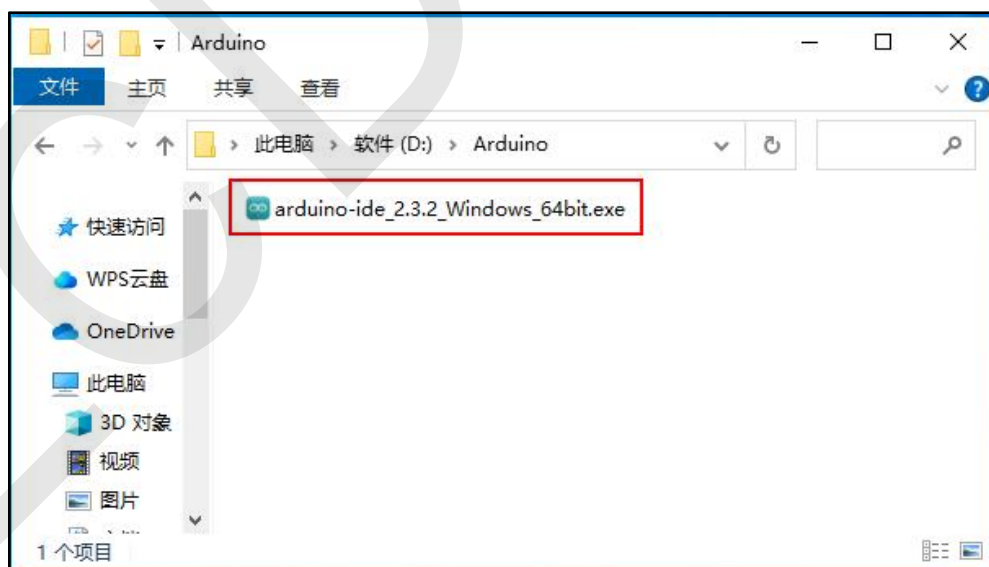
After clicking the button, a "New Download Task" window will pop up. Click the "Browse" button to select the save path for the software installation package, and then click the "Download" button to start downloading, as shown in the following figure:



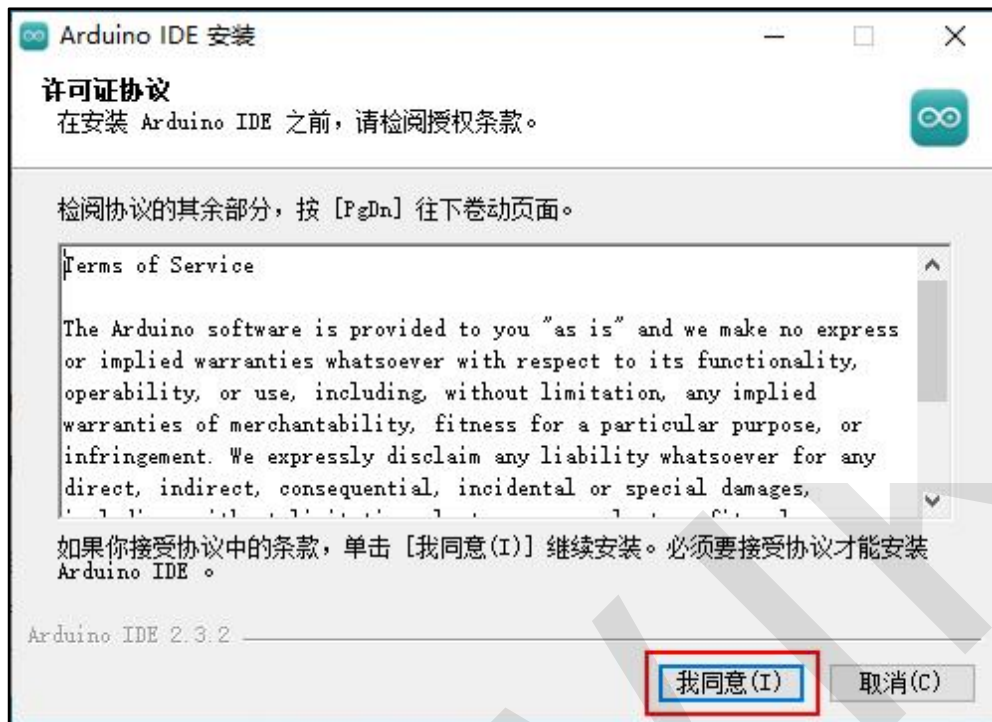
Picture1.4 Arduino IDE software installation package download task

2. Arduino IDE Software installation

Find the installation package save path of Arduino IDE software, then double-click the exe file, enter the program installation, as shown in the figure below:



Picture2.1 Arduino IDE installation package exe file



Picture2.2 Arduino IDE software installation license agreement

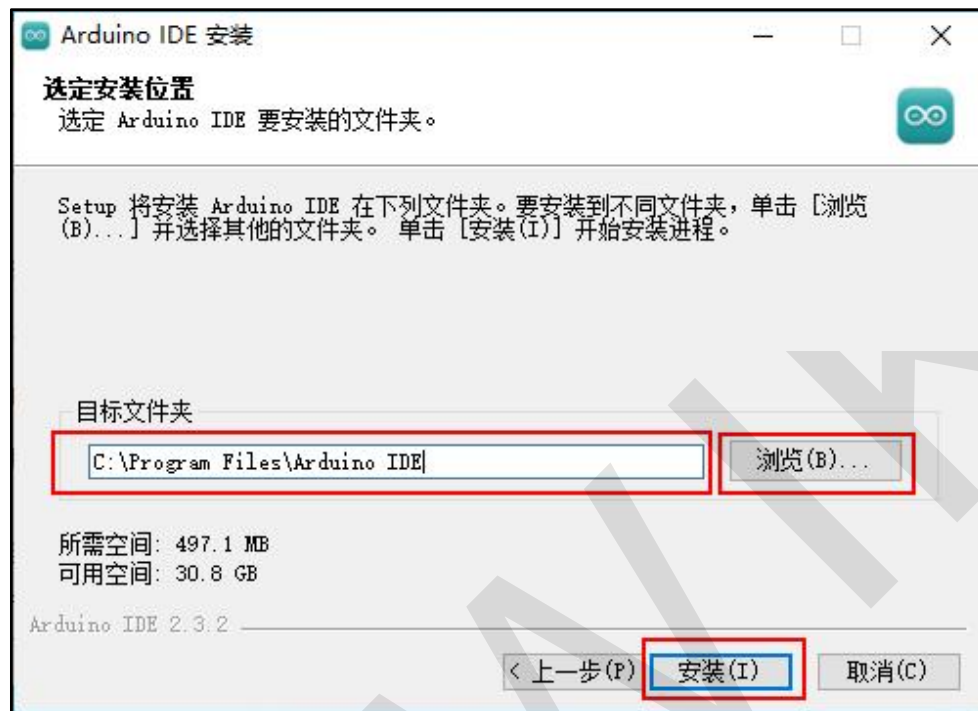
Click the "I Agree" button to enter the installation user selection interface, select whether to install for the current user or all users, generally select "Install anyone who uses this computer (all users)" as shown in the following figure:



Picture2.3 Arduino IDE Software installation user selection

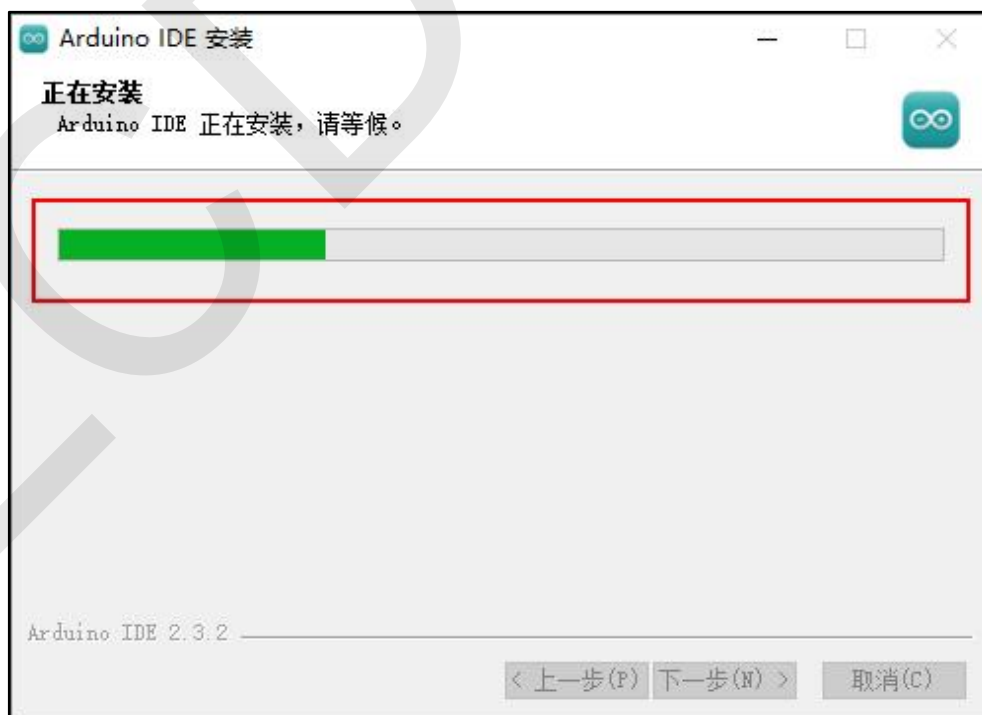
Next, click the "Next" button to enter the installation directory setup

interface, click the "Browse (B)..." button to select the installation or enter the installation directory directly, as shown in the following figure:



Picture2.4 Arduino IDE Software installation directory selection

Next, click the "Install" button to start the installation, you can see the progress bar changing as shown in the following figure:



Picture2.5 Arduino IDE Software installation process

After the progress bar has finished scrolling, the installation completed interface will pop up. If "Run Arduino IDE(R)" is checked, then after clicking the "Finish (F)" button to close the interface, the Arduino IDE software will automatically open. If it is not checked, it will not. As shown in the following figure



Picture2.6 Arduino IDE Software installation completed

When you open the Arduino IDE software for the first time, it will prompt you to install some software libraries and drivers, such as Adafruit Industries LLC (COM and LPT), Arduino srl Arduino USB Driver, Arduino SA Arduino USB Driver, Genuino USB Driver, as shown below:


```
Output
已安装 Arduino_BuiltIn@1.0.0
正在下载 Firmata@2.5.9
Firmata@2.5.9
正在安装 Firmata@2.5.9
已安装 Firmata@2.5.9
正在下载 Servo@1.2.1
Servo@1.2.1
正在安装 Servo@1.2.1
已安装 Servo@1.2.1
正在下载 LiquidCrystal@1.0.7
LiquidCrystal@1.0.7
正在安装 LiquidCrystal@1.0.7
已安装 LiquidCrystal@1.0.7
正在下载 Mouse@1.0.1
Mouse@1.0.1
正在安装 Mouse@1.0.1
已安装 Mouse@1.0.1
正在下载 SD@1.2.4
SD@1.2.4
正在安装 SD@1.2.4
已安装 SD@1.2.4
正在下载 Keyboard@1.0.5
Keyboard@1.0.5
正在安装 Keyboard@1.0.5
已安装 Keyboard@1.0.5
正在下载 Stepper@1.1.3
Stepper@1.1.3
正在安装 Stepper@1.1.3
已安装 Stepper@1.1.3
正在下载 Ethernet@2.0.2
Ethernet@2.0.2
正在安装 Ethernet@2.0.2
已安装 Ethernet@2.0.2
```

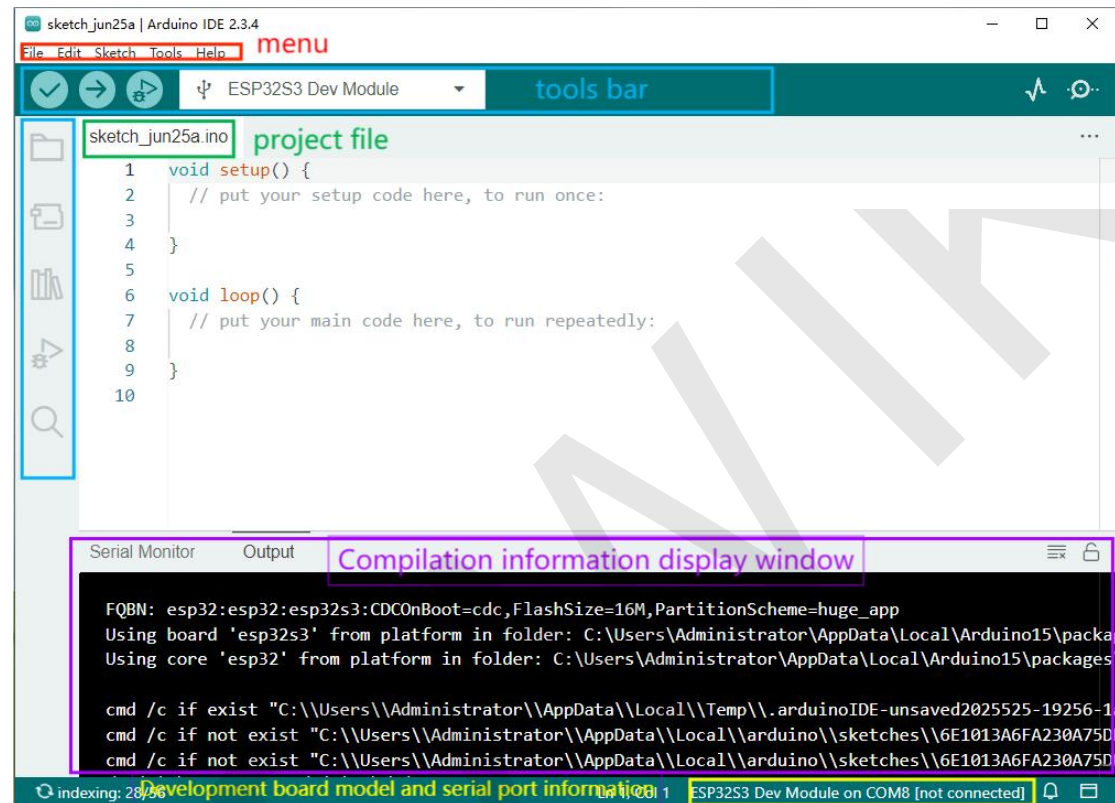
Picture2.7 Arduino IDE Software library installation



Picture2.8 Arduino IDE Software-driven installation

3. Arduino IDE Software introduction

Arduino IDE It has the functions of project creation, program code editing, debugging, compilation, uploading, software library management, and development board management, and the interface is shown in following figure:

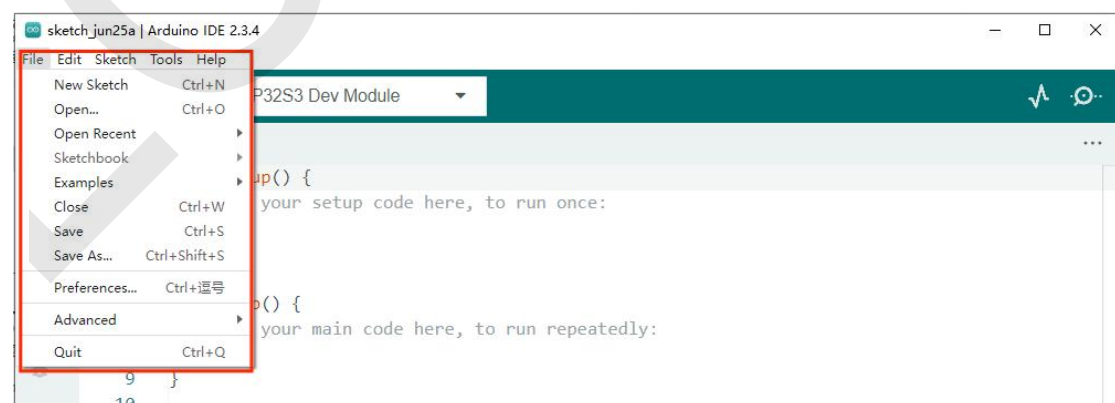


Picture3.1 Arduino IDE Interface

3.1. Menu bar

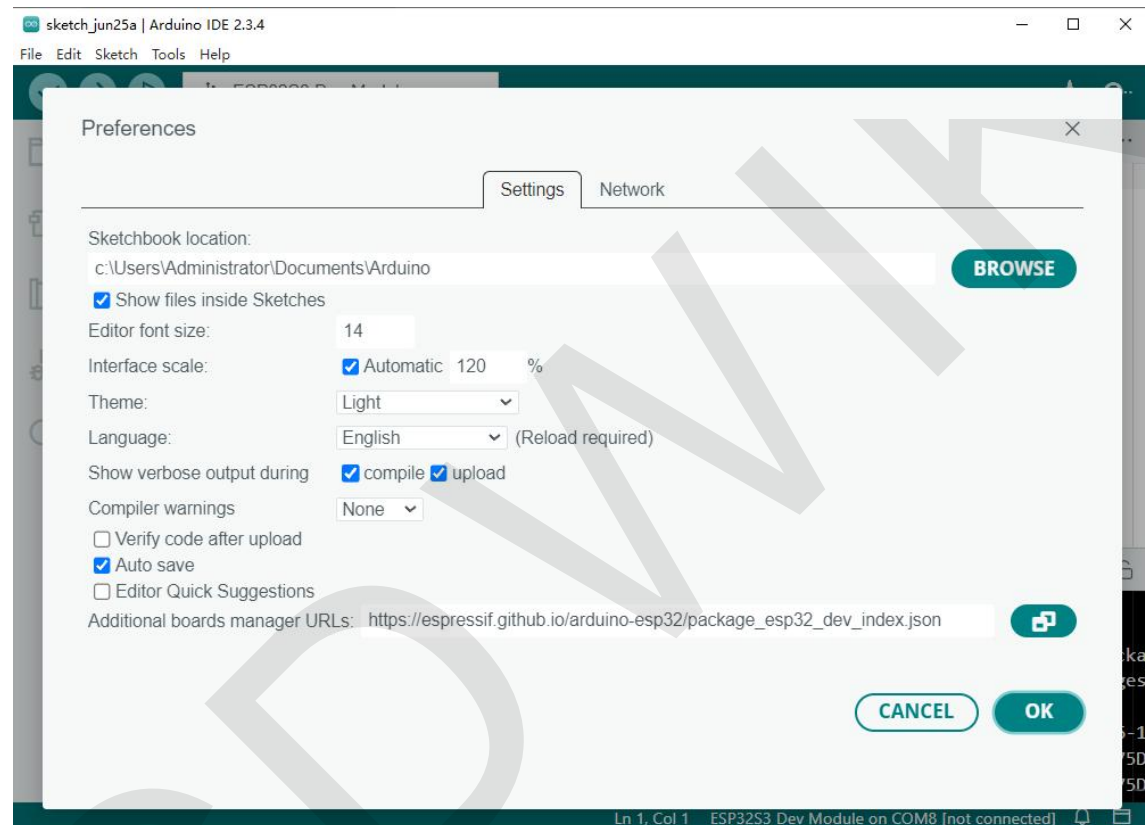
3.1.1. File

The content of the file menu in the menu bar is shown in the following figure:



Picture3.2 Arduino IDE File bar

Inside are basically operations such as creating, opening, saving, and closing projects. The **"Examples"** option can select to open the example programs that come with third-party software library and the core library of the development board. The **"Advanced Settings"** option can set the software shortcuts. Here we focus on the Preferences menu, click the **"Preferences"** option, as shown in the following figure:



Picture3.3 Arduino IDE Preferences menu

In the Preferences menu, you can make settings in the following aspects:

A. Project Folder Location: This is the default location where the software saves the project when creating a new one. You modify this location as needed. The libraries directory under this location is specifically used to store third-party software libraries. If you select the "Show Folders in Project" option you can display the project files in the project folder using the first button on the left toolbar of IDE.

B. Editor Font Size and Interface Ratio: You can the font size for the content

displayed in the editor and the size of the interface.

C. Color Theme: You can set the color display style of the editor, with options: Bright, Dark, Bright Contrast, and Dark Contrast.

D. Editor Language: You can set various languages. After setting, the software will automatically restart

E. Show Detailed Output: You can select the "Compile" and "Upload" options. If you select "Compile," the compilation information will be displayed in information display window during the compilation process; if you select "Upload," the upload information will be displayed in the information display window when uploading binary files to the MCU. To detailed information, it is recommended to select both.

F. Editor Warnings: You can choose "None," "Default," "More," "All, etc. Selecting "None" will not display any information, selecting "More" or "All" will display more comprehensive compilation information, but it will slow down the compilation speed

G. Verify Code After Upload: This is to verify whether the code is correct after the binary file is uploaded. You can select it.

H. Auto: It is recommended to select this option to prevent code loss in case of computer crash.

I. Editor Quick Suggestions: It is recommended to select this option. After, when editing code and calling a function, it will prompt the function's parameters, allowing you to write code quickly and correctly.

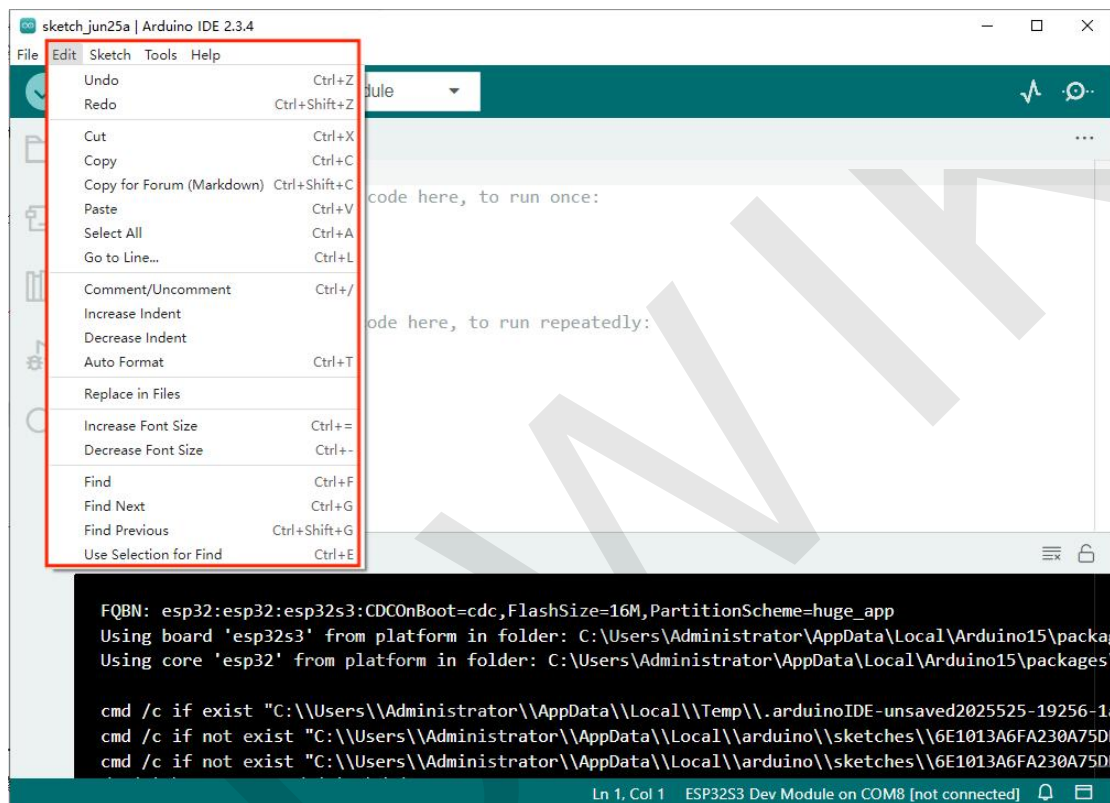
J. Other Board Manager URLs: you cannot find the development board you want to add in the development board manager of the IDE (non-official Arduino development board),

you need to add the board's address.

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

3.1.2 .Edit

The menu bar Edit menu interface is shown in the following figure:

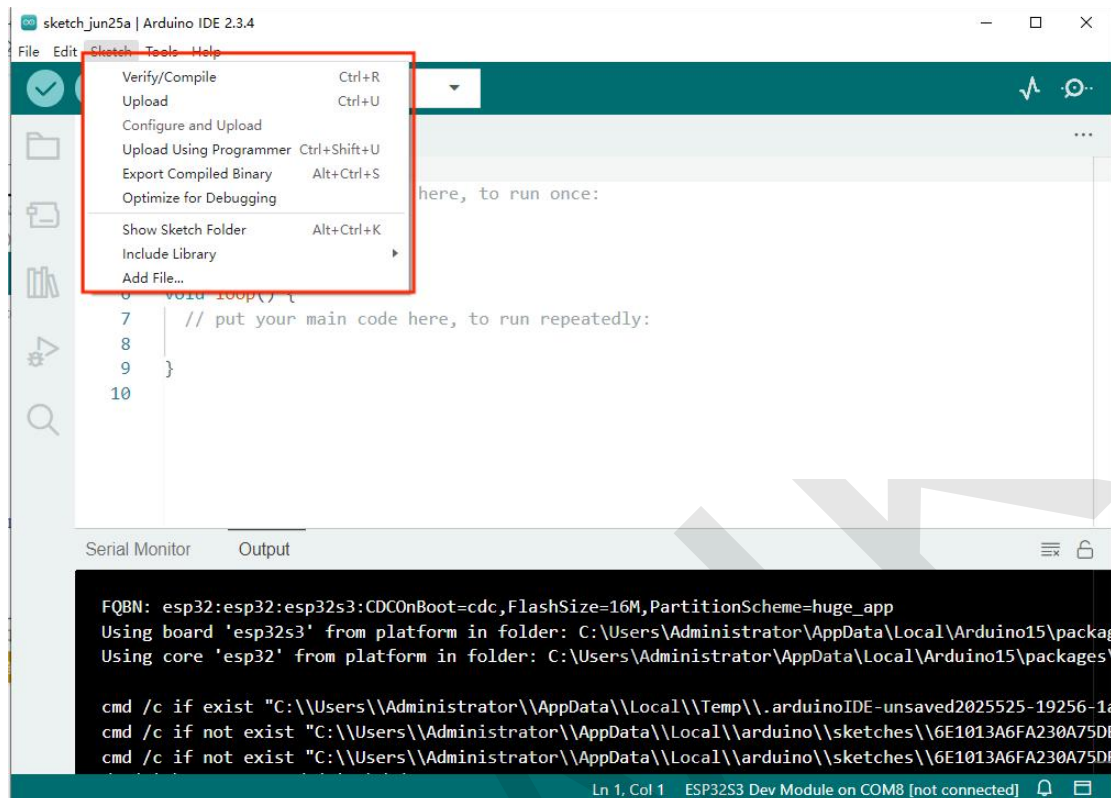


Picture3.4 Arduino IDE Edit bar

The edit menu mainly includes copy, cut, undo, paste, find, modify font size and other editing operations for the content of the project file

3.1.3. Sketch

The menu bar item menu interface is shown in the following figure:

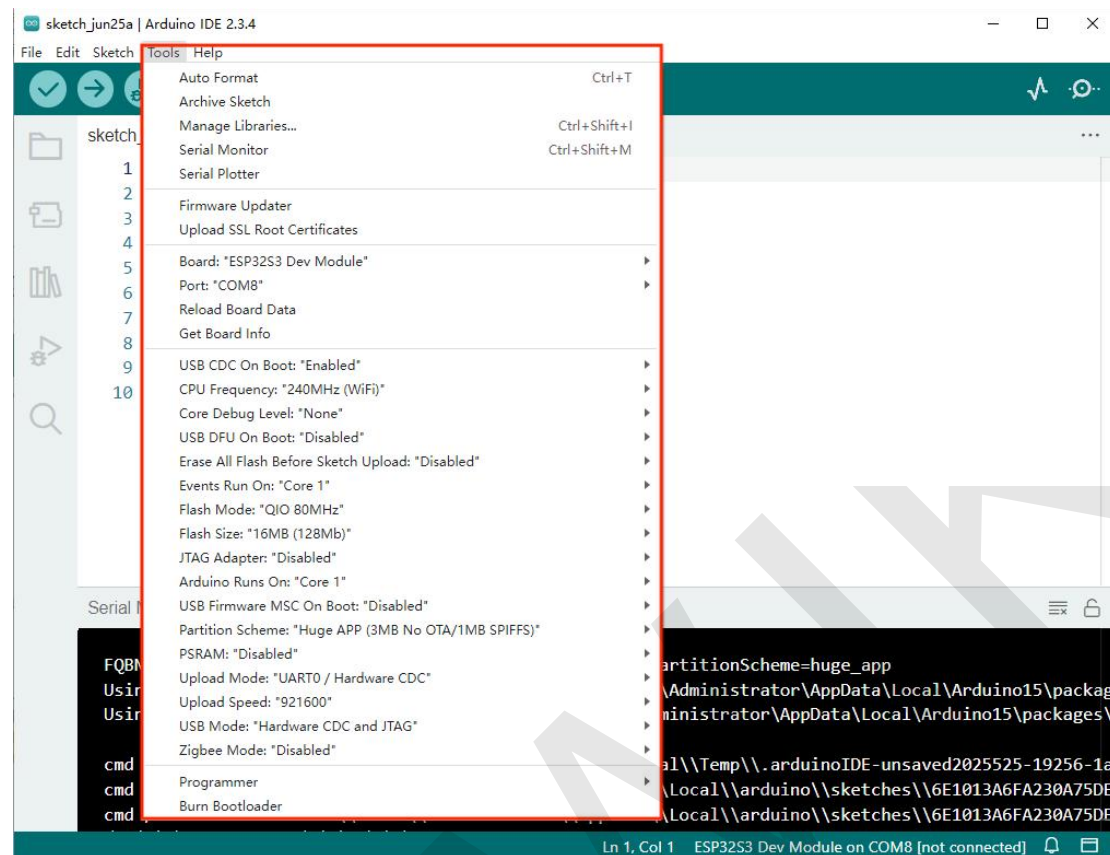


Picture3.5 Arduino IDE Sketch bar

Sketch menu bar is mainly for compiling, debugging, uploading, exporting, loading library files, etc..

3.1.4. tool

tool menu interface is shown in the following figure:



Picture3.6 Arduino IDE tool bar

In the tool menu, you can set the following:

A. Auto Format can automatically format the project code format, such as alignment, etc.

B. Project Archive can ZIP the entire project and save it.

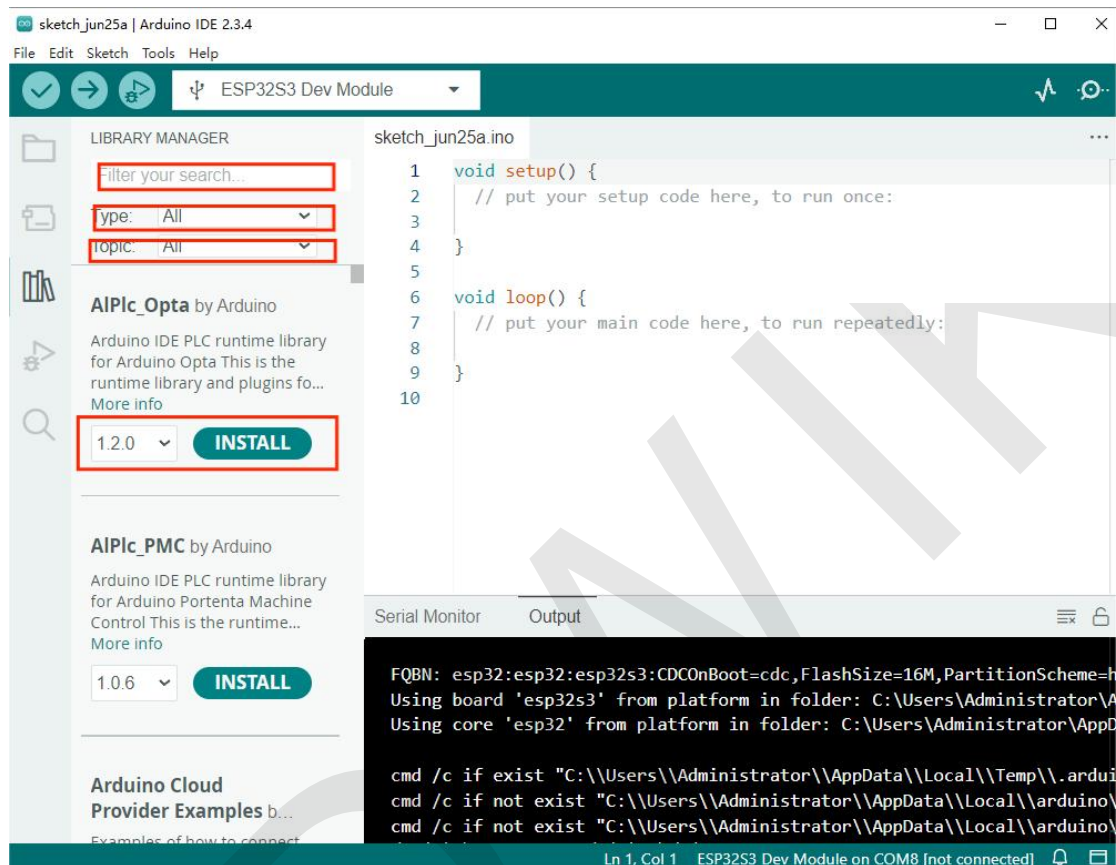
C. The Library Manager can search for, download, and install third-party software libraries. Click to enter, as shown in the following figure.

the library management interface, you can filter the library according to the type of software library and topic, or directly output the library name to search for the library. When the search is, select the library version and click the "Install" button to install it. The final software library is installed in the

"C:\Users\Administrator\Documents\Ar\libraries" directory (this is the default directory, of course, you can also modify it in the

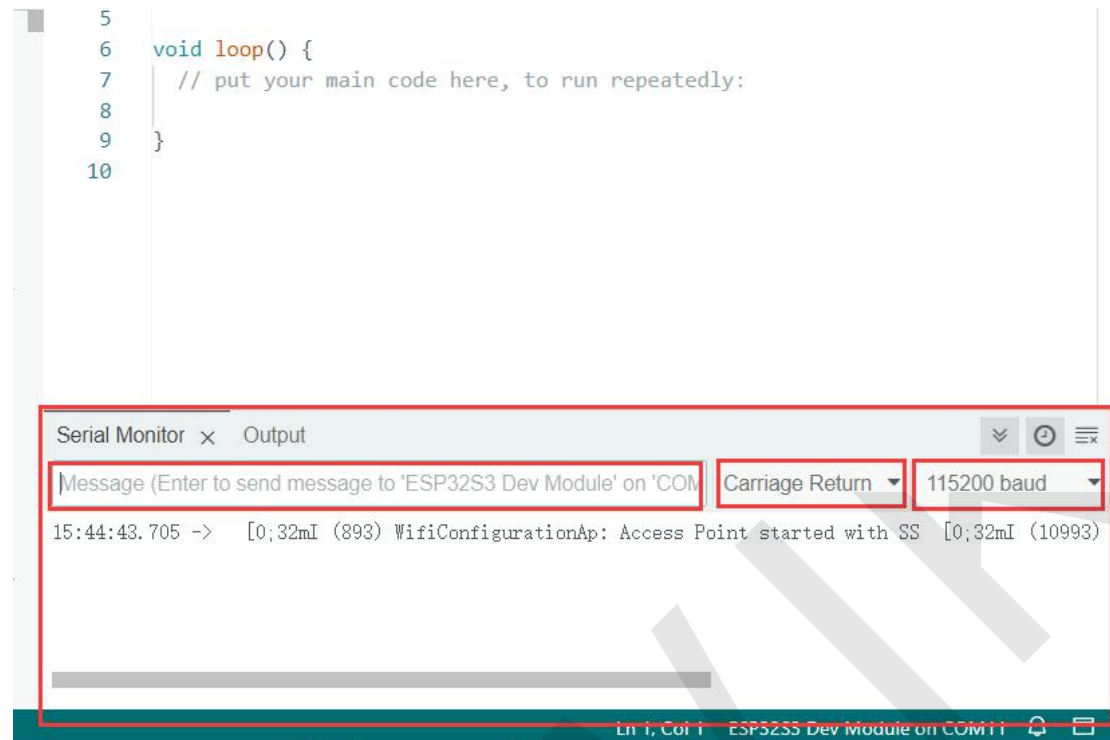
File->Preferences interface, the red font is the actual user name of computer). Of course, you can also install the software library without

going through the library manager. You can manually download the software library (which needs to be unzipped) and then copy it to the "C:\Users\Administrator\Documents\Arduino\libraries" directory.



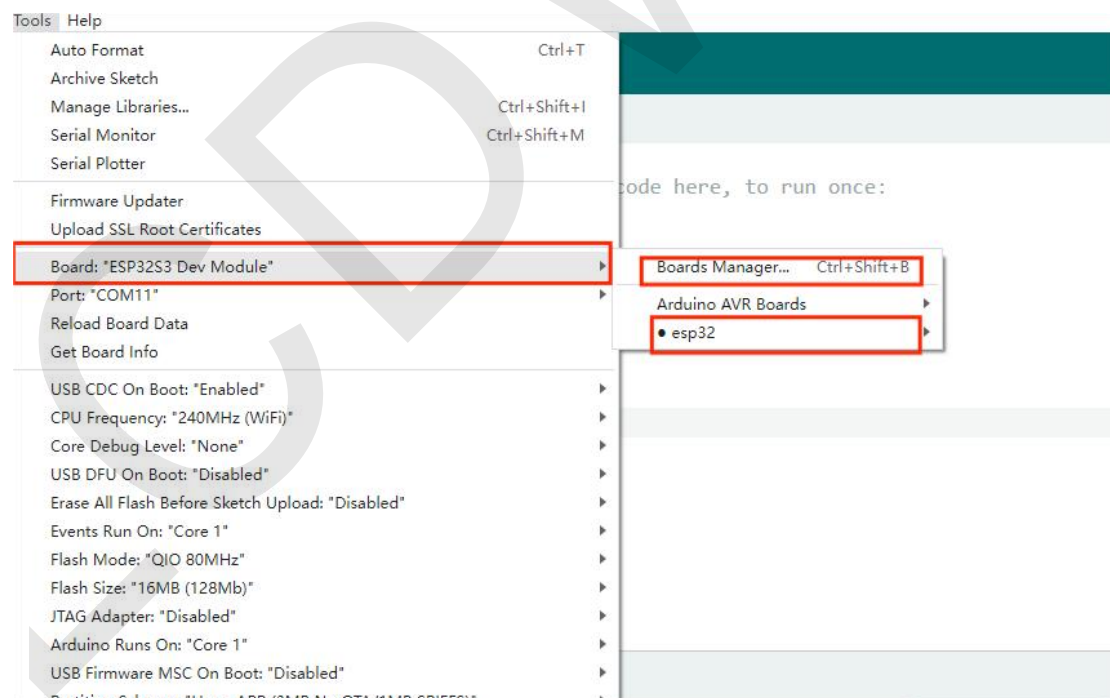
Picture3.7 Arduino IDE LIBRARY MANAGER

D. Serial Port Monitor and Serial Port Plotter are both open serial port interfaces, set the serial port baud rate, display serial port output information, and the serial port to send messages. (Note that you must connect the development board and correctly identify the serial port before you can use the serial port). As shown in the following:



Picture3.8 Arduino IDE Serial Port Output Window

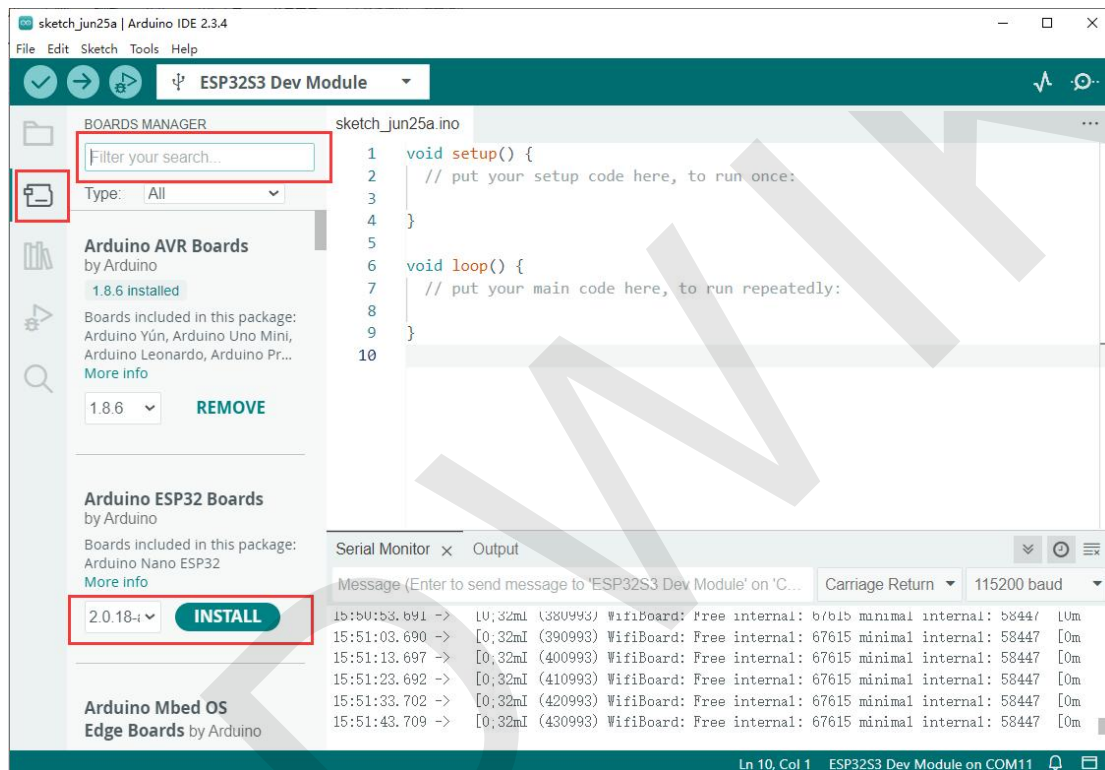
E. The development board contains two parts: the development manager and the development board selector, as shown in the following figure:



Picture3.9 Arduino IDE Development board menu

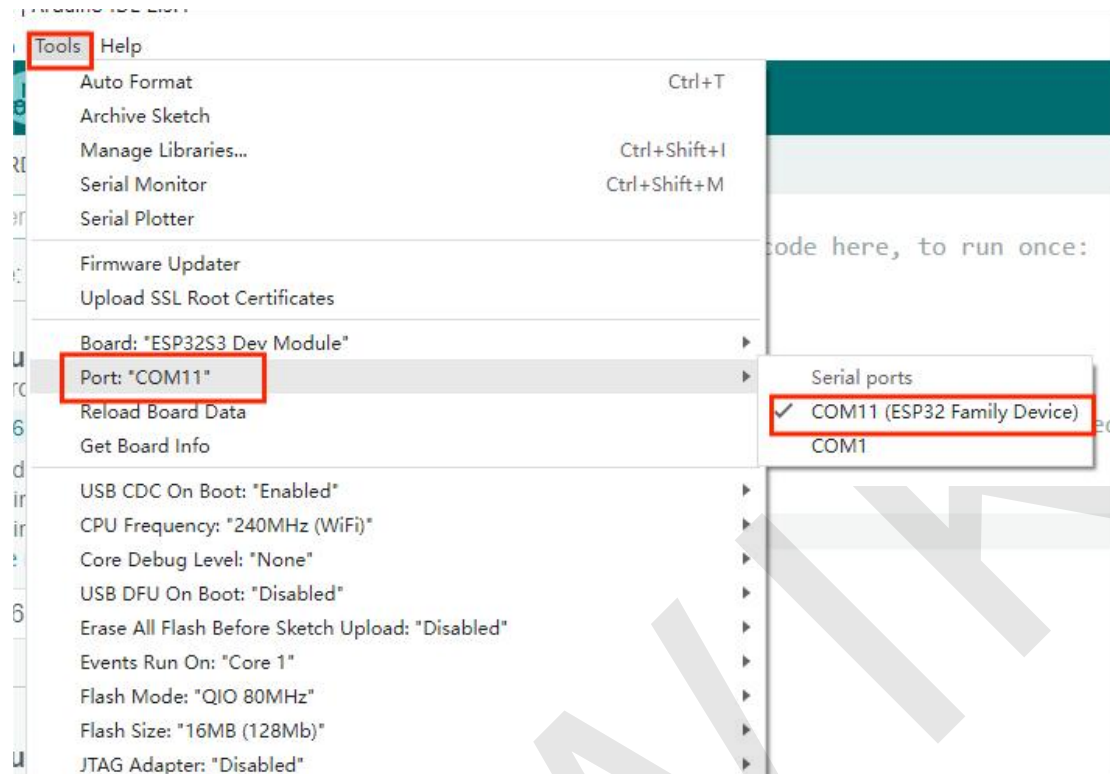
The development board option function is to select the current development board in use, and once the selection is successful, it will be displayed behind

the option. If development board being used does not exist, the core software library of the development board needs to be installed. In this case, the development board manager is required. The interface of the board manager is shown in the figure below. You can search for the development board and then select the version, and click the **"Install"** button to install the core software library of development board.



Picture3.10 Arduino IDE BOARDS MANAGER

F、 **Port** is the serial port connected to the development board, as shown in the following figure, the serial port will be displayed only after it is connected to the board.



Picture3.11 Arduino IDE Port Choose

Other options are basically not used, keep the default settings.

3.1.5. Help

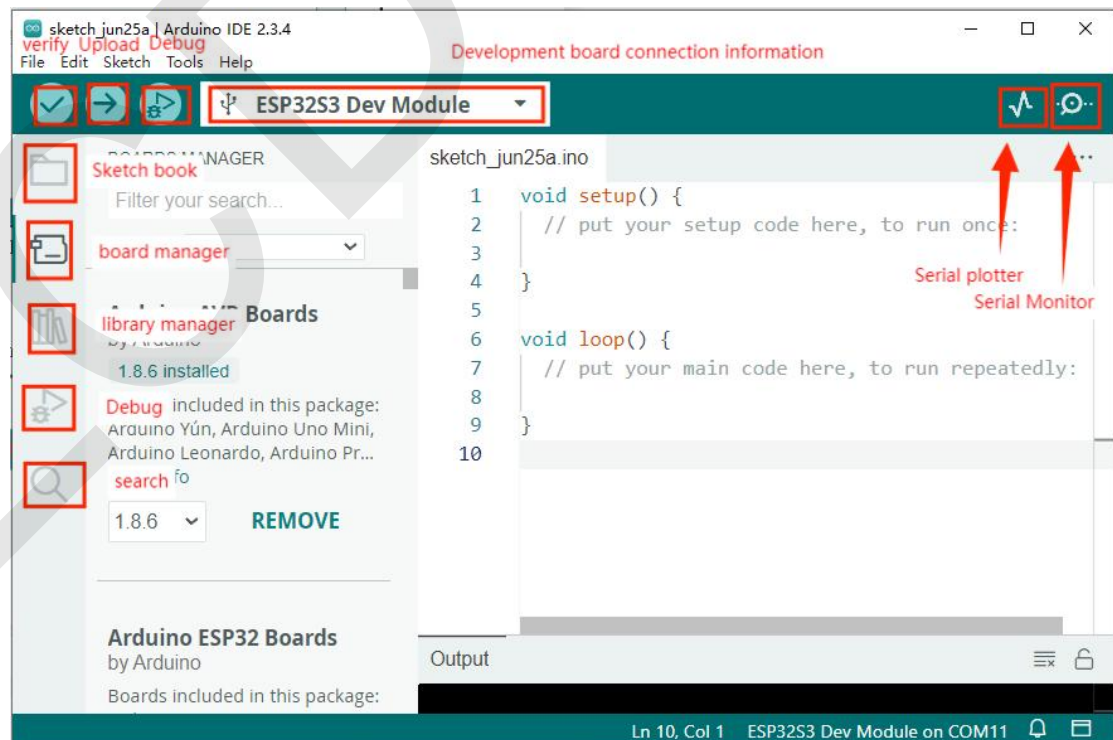
The **Help** menu interface is shown in the following figure, mainly linked to the official website's software usage instructions documentation.



Picture3.12 Arduino IDE Help menu

3.2. Tool

The interface of the toolbar is shown in the following figure:



Picture3.13 Arduino IDE tool bar

- A、**Verification:** Compile and check if the program is correct, if correct, the compilation is successful, and the binary file is generated
- B、**Upload:** Compile the program to generate a binary file and upload it to the MCU of the development board.
- C、**Debugging:** Debug the code, can see the Debug process. The Arduino IDE does not support ESP32 debugging, so this function is not used.
- D、**Development Board Connection Information:** Pull down to which development board is connected to which serial port.
- E、**Serial Port Plotter:** You can see the data and convenience received by the development board more intuitively.
- F、**Serial Monitor:** Output the code running situation in text information, and can also input text information to the development board.
- G、**Project Folder:** You can display the file in the project folder (the project folder address is set in the Settings in the Preferences).
- H、**Development Board Manager:** Select the development board that has been installed and development board that needs to be installed, and the function is the same as the development board option in the tool menu bar.
- I、**Library Management:** Download and install third- software libraries, and the function is the same as the manage library option in the tool menu bar.
- J、**Debugging:** The same as the debugging function in item C above
- K、**Search:** Used to search for functions and other information.

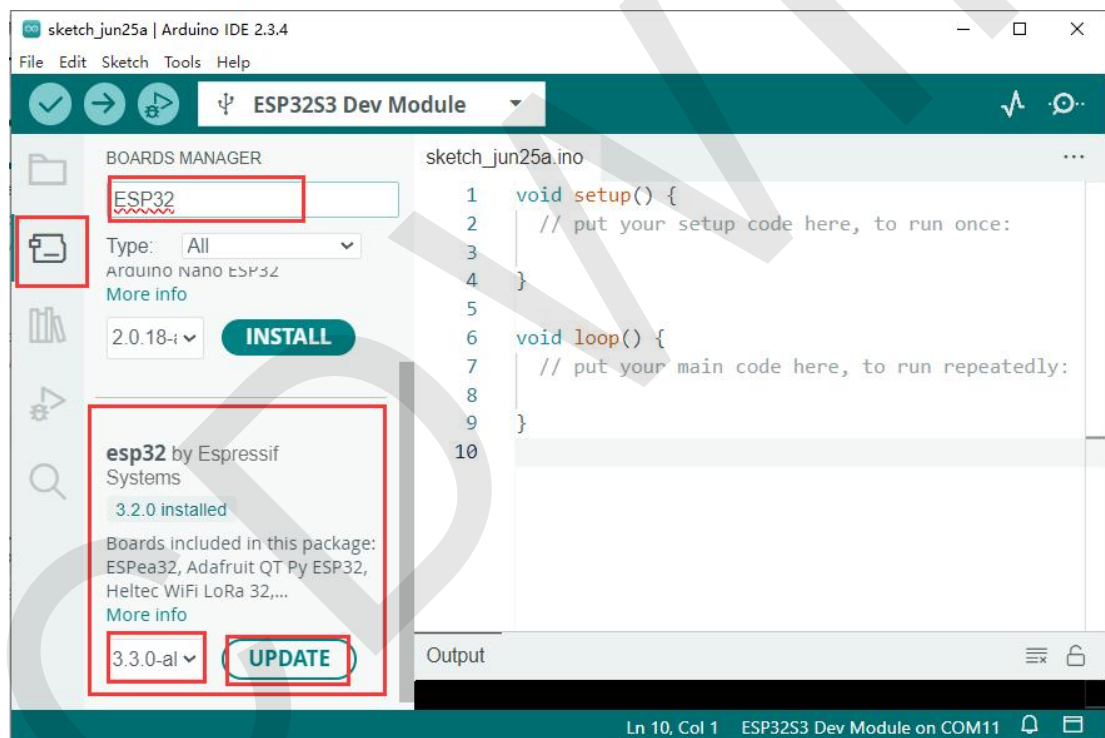
4. Install the Arduino-ESP32 core software library

The Arduino-ESP32 Core software library is a plug-in for the Arduino platform that provides low-level support for software development on the ESP32 chip of the Arduino platform. Because the Arduino IDE does not support ESP32 by default, it is necessary to install the Arduino-ESP32 Core software library.

4.1. Arduino IDE board Manager Installation

A. Open the Arduino IDE software, click **Tools** → **Board** → **Board Manager**, or directly click the Board Manager icon in the left toolbar. In the bar of the Board Manager interface, enter ESP32, and the ESP32 search results will appear, as shown in the following figure.

Note: If you cannot find ESP32 core software library, you need to click **File** → **Preferences**, enter `"https://espressif.github.io/arduino-esp32/packageesp32_index.json"` in the Other Board Manager URLs in the Preferences interface, and then search again according to the above steps.



Picture4.1 Arduino-ESP32 Core software library search

B. Select **"esp32 by Espressif Systems"**, then select the version number, and finally click on the Install button, as in the following figure.

Note: Version 3.0 is developed based on ESP32 idf 5.1, and version 2.0 is developed based on32 idf 4.4. There are differences in the APIs of Bluetooth, Timer, I2S driver, LEDC driver, and Timer in the software the two versions, so if the examples in version 2.0 involve the above APIs, and you use version 3.0 to compile, errors will be reported. Please attention to the version selection.

The installation time is relatively long, and there may be a case where the download fails during the installation process, so you need to try the installation times.

The downloaded installation file compressed packages are all saved in the

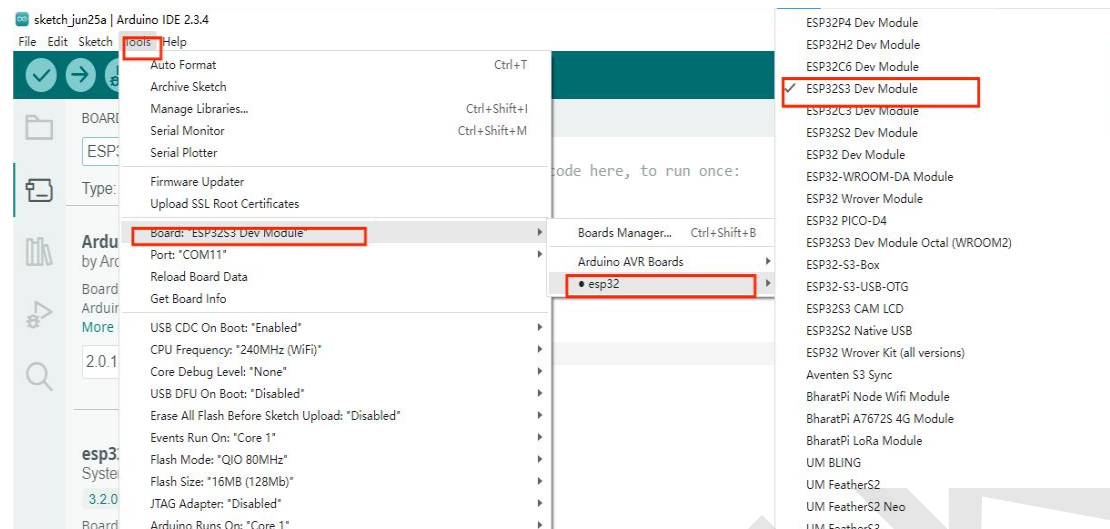
"C:\Users\Administrator\AppData\Local\Arduino15\staging\" directory

(The red part is the actual user name of the computer, and the AppData directory is a hidden directory. You need to click on the **Tools->Folder OptionsView->Select Show hidden files**, folders, and drives in the folder menu bar, and then click OK to save)



Picture4.2 Arduino-ESP32 Core software library search

C、After installation is completed, close the board manager, click **Tools->Board**, you can see the "esp32" option, click this option, you see a lot of ESP32 development boards, as shown in the figure below:



Picture4.3 ESP32 board choose

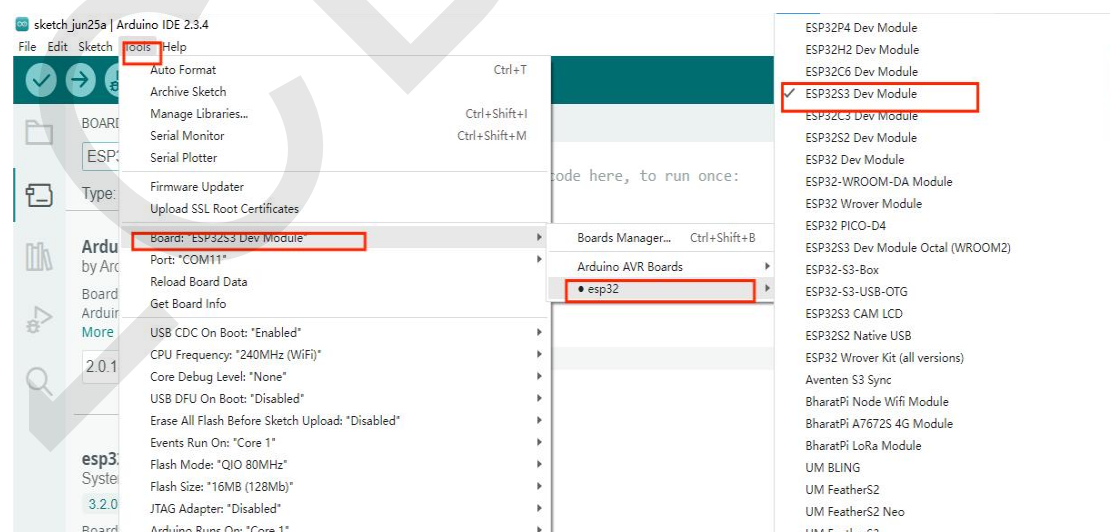
5. Compile, download, and run the ESP3-S3 example program

5.1. Configure the development board

After creating a new one or opening an existing example program in the Arduino IDE, the first thing to do is to configure the development board. The steps are as:

A. Power on the development board by connecting it to the computer's USB port, then select the target development board model, here ESP32S3 click the "Tools" button, and select Board->esp32->ESP32S3 Dev Module,

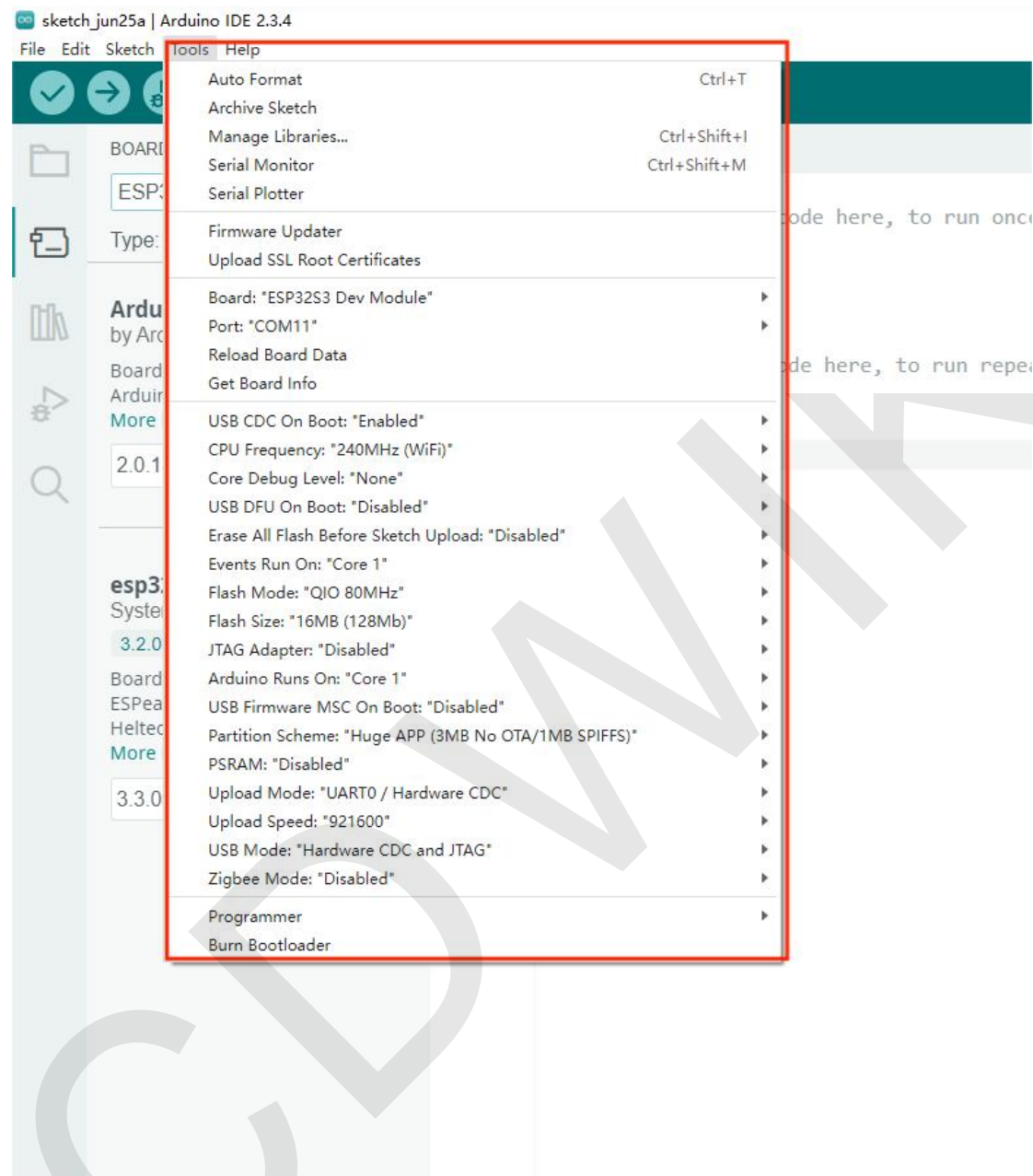
as shown in the following figure:



Picture5.1 ESP32 board choose

B. Click the "Tools" button, you can see the default configuration of the

ESP32 development board, as shown in the following figure:



Picture5.2 ESP32 development board configuration

Here is an introduction to the configuration parameters:

Port: Select the serial port number connected to the ESP3 development board, which will generally be automatically recognized.

USB CDC On Boot: USB virtual serial port (USB-Serial-JTAG mode) output enable switch, select "Enabled" to enable the serial port function, select "Disabled" to close the serial port function. If the development board does not have a USB to serial port function, it is recommended enable this function.

Note: Only Windows 10 and above support USB virtual serial ports

CPU Frequency: CPU clock frequency, the optional parameters are: 24MHz (WiFi), 160MHz (WiFi), 80MHz (WiFi), 40MHz, 20MHz, 1MHz. Generally speaking, the higher the frequency, the greater the power consumption, and you can choose according to your needs. Here we do not consider power consumption, and select the maximum frequency of 240MHz to give full play to the best performance. It should be noted that the 240MHz, 160MHz, 80MHz these three frequencies can guarantee the normal operation of WiFi and BT, other frequencies can not guarantee the normal operation of WiFi and BT, but can only guarantee the operation of the CPU.

Core Debug Level: Arduino core debug log level, output via serial port, optional parameters are: None, Error, Warn, Info, Debug, verbose. Among them,

None: Do not output any debug logs;

Error: Only output error-level debug logs;

Warn: Only output warning and above-level debug logs;

Info: Only output information and above-level debug logs;

Debug: Only output debug and above-level debug logs;

Verbose: Output levels of debug logs in the kernel debugging; In general, there is no need to pay attention to the kernel debugging logs, unless you are developing some functions related to the kernel. So here you can choose none.

USB DFU On Boot: Configure whether to upgrade the device firmware via the USB interface during the device startup phase. Select "Enabled" to enable (need to be set to "USB-OTG" mode), select "Disabled" to close.

Erase ALL Flash Before Sketch Upload: Configure to completely erase the entire Flash when uploading code, the optional parameters are: Disabled, Enabled. Select "Disabled" if you do not need to completely erase, select "Enabled" if you need to completely erase. If you choose to completely erase, the upload code will be slower, and in addition, frequent complete erasure of Flash will affect the life of Flash, so here you can choose "Disabled".

Event Runs On: Configure the ESP32 core on which the Arduino interrupt event runs, the parameters are: Core0, Core1. You can choose according to the situation, here

the default is Core1. The cores configured here can be the same as those in Arduino Runs On, or they can be different. When configured to be the same, it can reduce the power consumption of ESP32; when configured to be different, can improve the efficiency of program operation.

Flash Mode: Flash communication mode and frequency mounted on ESP32, the selectable **parameters are:** QIO 80, QIO 120MHz, DIO 80MHz, OPI 80MHz. Among them, QIO is used with 4 SPI data for Flash write and read. DIO is used with 2 SPI data lines for Flash write and read. OPI is used with 8 SPI data lines for Flash write read. Select according to the actual connection method of Flash. Here, 4 SPI data lines are used for Flash write and read, so QIO is selected.

Flash: The capacity of Flash mounted on ESP32, the selectable parameters are: 4MB (32Mb), 8MB (64Mb) 16MB (128Mb), 32MB (256Mb). Select according to the actual capacity of Flash. Here, the used is 16MB, so select 16MB (128Mb).

JTAG Adapter: Configure the JTAG Adapter, the select parameters are: Disabled, integrated USB JTAG, FTDI Adapter, ESP USB Bridge. It is more convenient to use JTAG to debug the code. Select according to demand. The default is "Disabled".

Arduino Runs On: Configure the ESP32 core on which the Arduino Core task code runs, the selectable are: Core0, Core1. ESP32 has two cores, Core0 and Core1, respectively. Each core can run different code tasks. You can choose freely according the situation. Here, Core1 is selected by default.

USB Firmware MSC On Boot: Configure another method to upgrade the firmware via USB. When starting this option a removable storage disk will be created on the computer, and the firmware can be directly dragged into this disk for upgrade. Select "Enabled" to enable (need to be set "toUSB-OTG" mode), select "Disabled" to close.

Partition Scheme: The partition method of the Flash space mounted on ESP32. In order make more reasonable use of the Flash space, the Arduino IDE has designed more than a dozen partition methods, which will not be introduced one by one here. If you are interested, can learn by yourself. Here, the Flash used is 16MB, generally select "16MB flash (2MB APP/12.5MB FATFS)", if the project file is large and the compiled binary file is large, you can select "16MB flash (3MB APP/9.9MB FATFS)"

PSRAM: Configure ESP32's external PSRAM, with optional parameters: Disabled, QSPI PSRAM, 0 PSRAM. Some ESP32s have built-in SRAM and also hang an external PSRAM for memory expansion. In this case, you need to choose PSRAM. Some ESP32s only have internal SRAM, in which case you need to choose "Disabled". The ESP32 used here is mounted with an 8- PSRAM, so choose "OPI PSRAM".

UploadMode: Configure the device code upload interface. Optional parameters: UART0/Hardware CDC, USBOTG CDC (TinyUSB). UART0/Hardware CDC indicates that the code is uploaded using the onboard USB to serial or USB virtual serial port. USB-G CDC (TinyUSB) indicates that a software-emulated USB hardware interface is used to upload the code. Here, the USB virtual serial port is used to upload the, so choose "UART0/Hardware CDC".

Upload Speed: The rate at which the code is uploaded, with optional parameters: 51200 230400, 256000, 115200, 921600. Select according to the rate supported by the USB to serial port on the development board. For example, the maximum rate supported by the CH340C used here is 2Mbps, so maximum value of 921600 is selected.

USB Mode: Configure the USB interface mode. Optional parameters: Hardware CDC and JTAG, USB-OT (TinyUSB). Hardware CDC and JTAG indicates the use of USB virtual serial port and JTAG debugging. USB-OTG (TinyUSB) indicates the use a software-emulated USB hardware interface. When using USB OTG mode, this interface needs to be configured.

Note: If you use USB Mode, you must connect the's built-in USB interface, not the onboard USB to serial port.

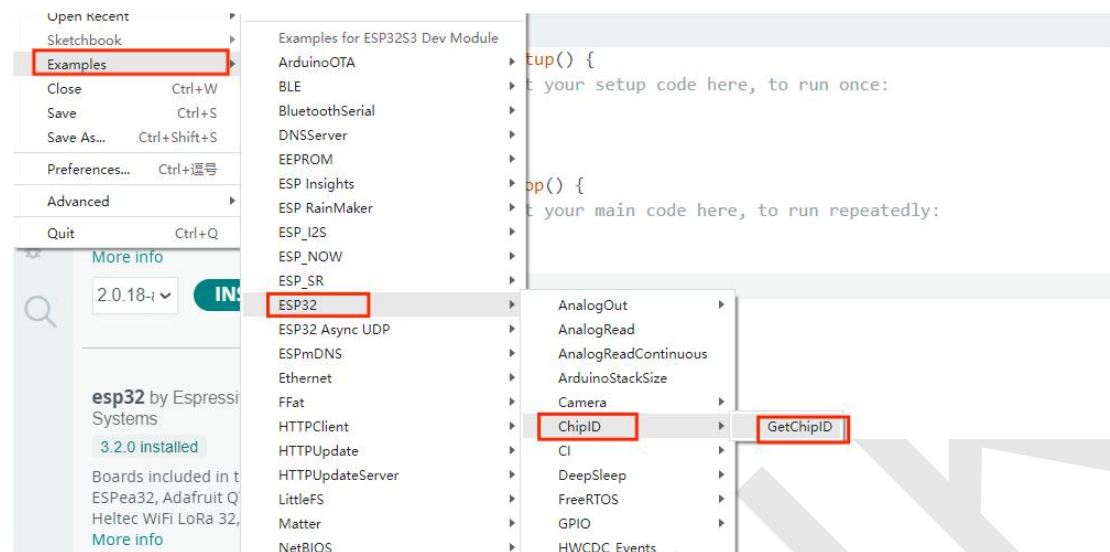
Zigbee Mode: Configure the Zigbee operating mode. Optional parameters: Disabled Zigbee ZCZR. Disabled indicates that the Zigbee function is turned off. Zigbee ZCZR indicates that the Zigbee function is on. Since Zigbee is not used here, choose to turn it off.

5.2. Compile, download and run the program

The examples used here are the examples that come with the Arduino-ESP32 core software library. You can also create a new project to compile, download, run, and you can open the completed project to operate.

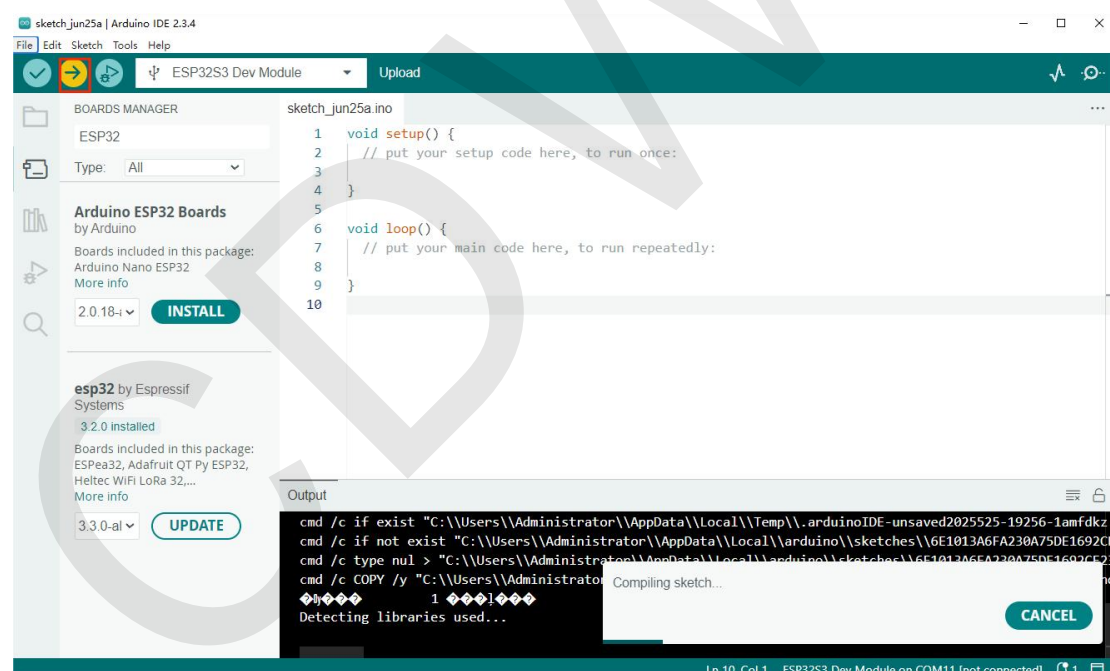
A、Open the Arduino IDE, click on the "File" button, select **Examples ->**

ESP32 -> ChipID -> GetChipID, as shown in following figure:



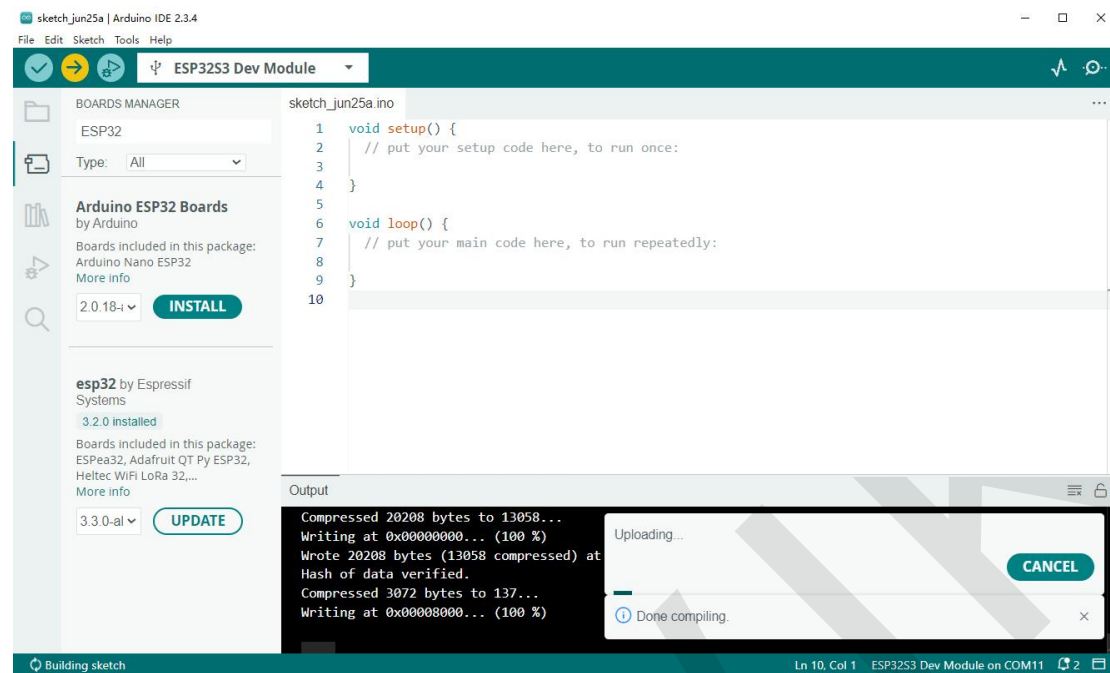
Picture5.3 ESP32 Demo

B、Click the "Upload" button, you can see the "Compiling sketch..." prompt, as shown in the following figure:



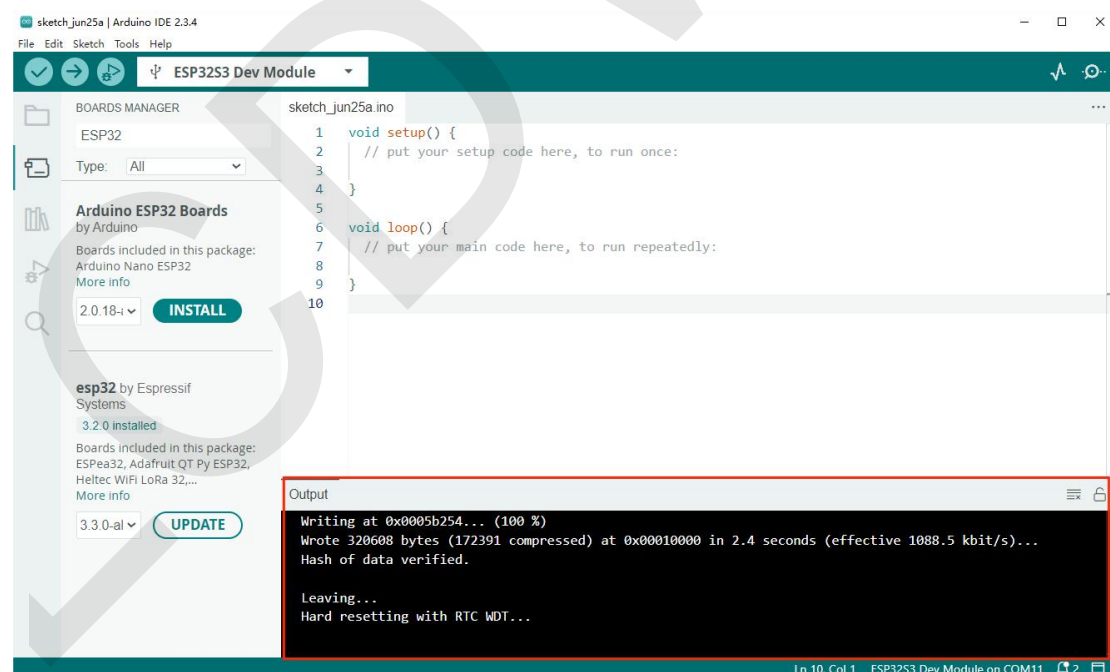
Picture5.4 ESP32 项目编译

C、After successful compilation, the message "Uploading..." will be displayed, and the message output window will output the compilation success information, as shown the following figure:



Picture5.5 ESP32 sketch compiled successfully

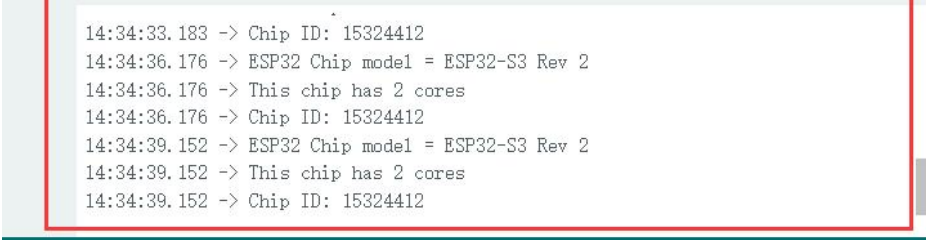
D、After the upload is successful, the information output window will output the upload information and a prompt for program operation, as shown in the following figure:



Picture5.6 ESP32 sketch upload successful and running

F、Click the menu bar Tools->Serial Monitor, pop up the serial port interface, set the baud rate to 115200, you see the information output in the serial port

terminal, and the program runs successfully, as shown in the following figure:

A screenshot of a serial terminal window with a light gray background and a red border. The window displays the following text:

```
14:34:33.183 -> Chip ID: 15324412
14:34:36.176 -> ESP32 Chip model = ESP32-S3 Rev 2
14:34:36.176 -> This chip has 2 cores
14:34:36.176 -> Chip ID: 15324412
14:34:39.152 -> ESP32 Chip model = ESP32-S3 Rev 2
14:34:39.152 -> This chip has 2 cores
14:34:39.152 -> Chip ID: 15324412
```

Picture5.7 ESP32 Program serial port output