ESP32-S3-Touch-LCD-1.69

From Waveshare Wiki
Jump to: navigation, search

Overview

Introduction

ESP32-S3-Touch-LCD-1.69 is a cost-effective, high-performance microcontroller development board designed by Waveshare. On the small plates, onboard a 1.69inch capacitive touch LCD screen, Lithium battery charging chip, 6-axis sensor (3-axis accelerator and 3-axis gyroscope), and peripheral interface such as RTC, easy to develop and embed into products.

ESP32-S3-Touch-LCD-1.69

(https://www.waveshare.com/esp32-s3-touch-lcd-1.69.htm)

1.69inch, 240 × 280
USB, UART, I2C

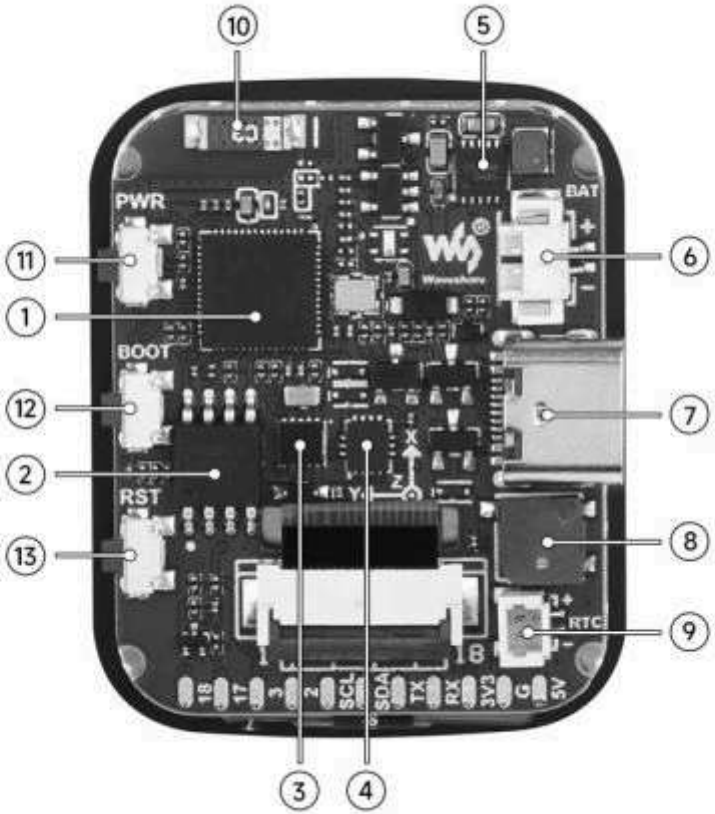
Features

- Equipped with high-performance Xtensa®32-bit LX7 dual-core processor, main frequency running up to 240MHz.
- Support 2.4 GHz Wi-Fi (802.11 b/g/n) and Bluetooth® 5 (LE), and onboard antenna.
- Built-in 512KB SRAM and 384KB ROM, 8MB PSRAM, and external 16MB Flash.

- Onboard 1.69inch capacitive touch LCD screen, 240×280 resolution, 262K colors, can clearly display color pictures.

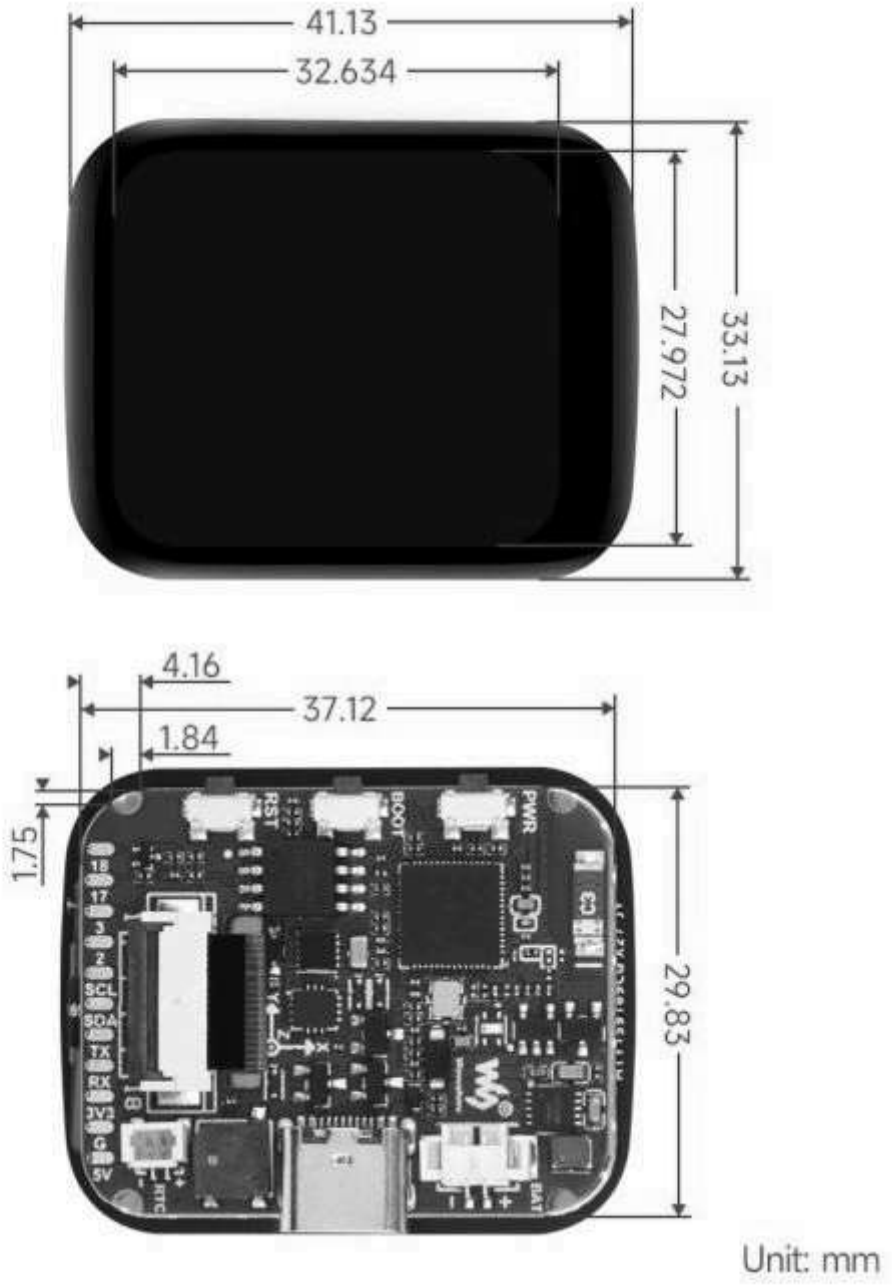
Hardware Description

- Onboard patch antenna, uses 0 ohm shorted optional external antennas, as shown in ⑩.
- Onboard PCF85063 RTC chip with reserved SH1.0 RTC battery header (support charging), supports timing function, as shown in ③ and ⑨.
- Onboard QMI8658C 6-axis Inertial Measurement Unit (IMU), including a 3-axis gyroscope and a 3-axis accelerator, as shown in ④.
- Onboard ETA6089 high-performance Lithium battery charging chip, M1.25 Lithium battery interface, easy to install lithium batteries charge and discharge for long-term usage, as shown in ⑤ and ⑥.
- Onboard buzzer can be utilized as an audio peripheral, as shown in ⑧.
- Onboard Type-C interface, connect to ESP32-S3 USB for demo programming and log printing, as shown in ⑦.
- Onboard BOOT and RST function buttons, easy to reset and enter the download mode, as shown in ⑫ and ⑬.
- Onboard function circuit button, can be customized as power-on button, and can identify single pressing, double pressing, and long pressing, as shown in ⑪.


10
5
PWR
BAT
11
6
1
BOOT
12
7
2
RST
13
8
RTC
9
3
4
18
17
3
2
SCL
SDA
TX
RX
3V3
G
5V

(/wiki/File:ESP32-S3-Touch-LCD-1.69-Hard.jpg)

Dimensions


41.13
32.634
27.972
33.13
4.16
37.12
1.84
1.75
29.83
RST
BOOT
PWR
18
17
SCL
SDA
TX
RX
3V3
G
5V
BAT
Unit: mm

(/wiki/File:ESP32-S3-Touch-LCD-1.69_Dim.jpg)

Development Environment Setup

- **The following development system is Windows by default, it is recommended to use VSCode plug-in for development.**

ESP-IDF

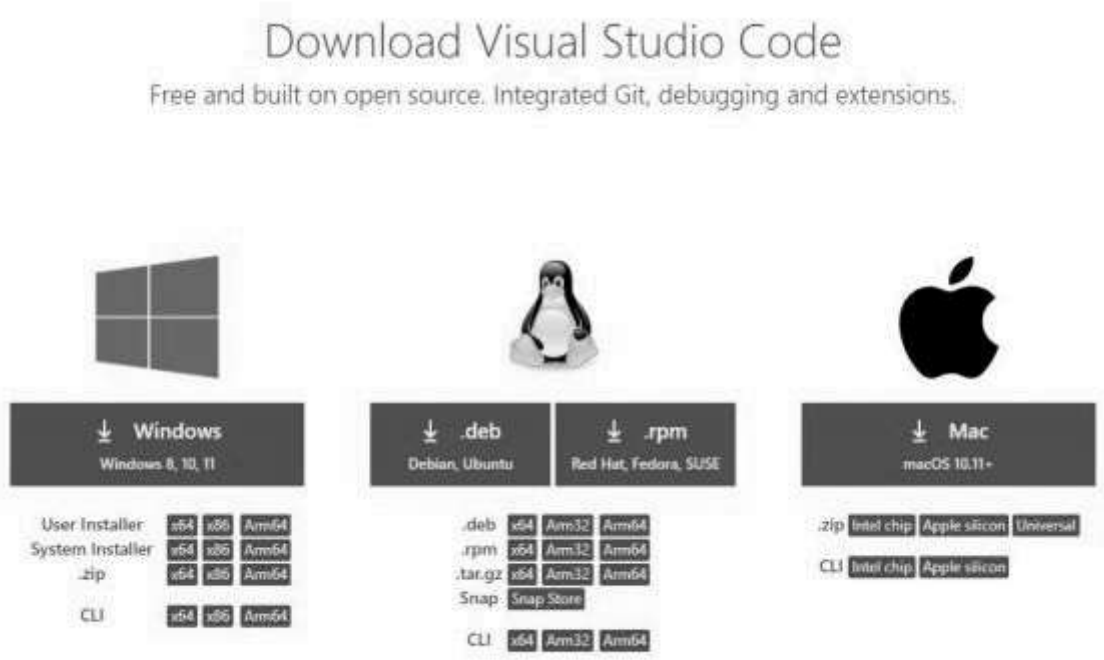
[Collapse]

- It is recommended to develop with the VSC plug-in.

Develop with VSCode Plug-in

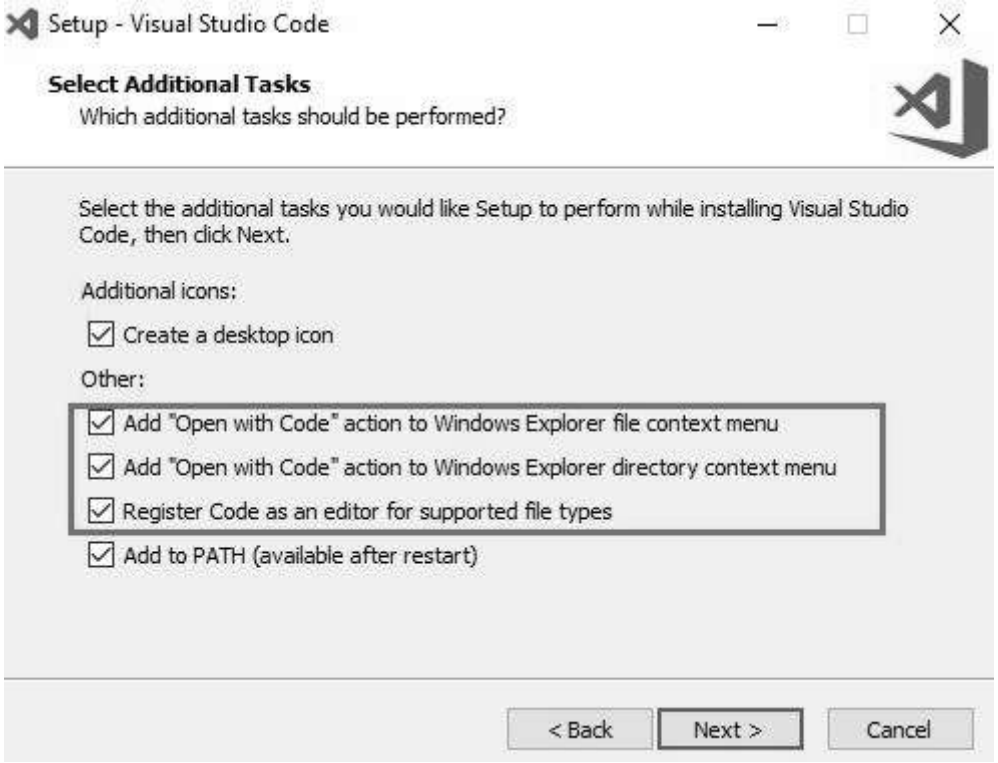
Install VSCode

1. Open the download page (https://code.visualstudio.com/download) of the official VSCode website, and select the corresponding system and system bit to download.



(/wiki/File:ESP32-S3-Pico_05.jpg)

2. After running the installation package, the rest can be installed by default, but here for the subsequent experience, it is recommended to check boxes 1, 2, and 3.



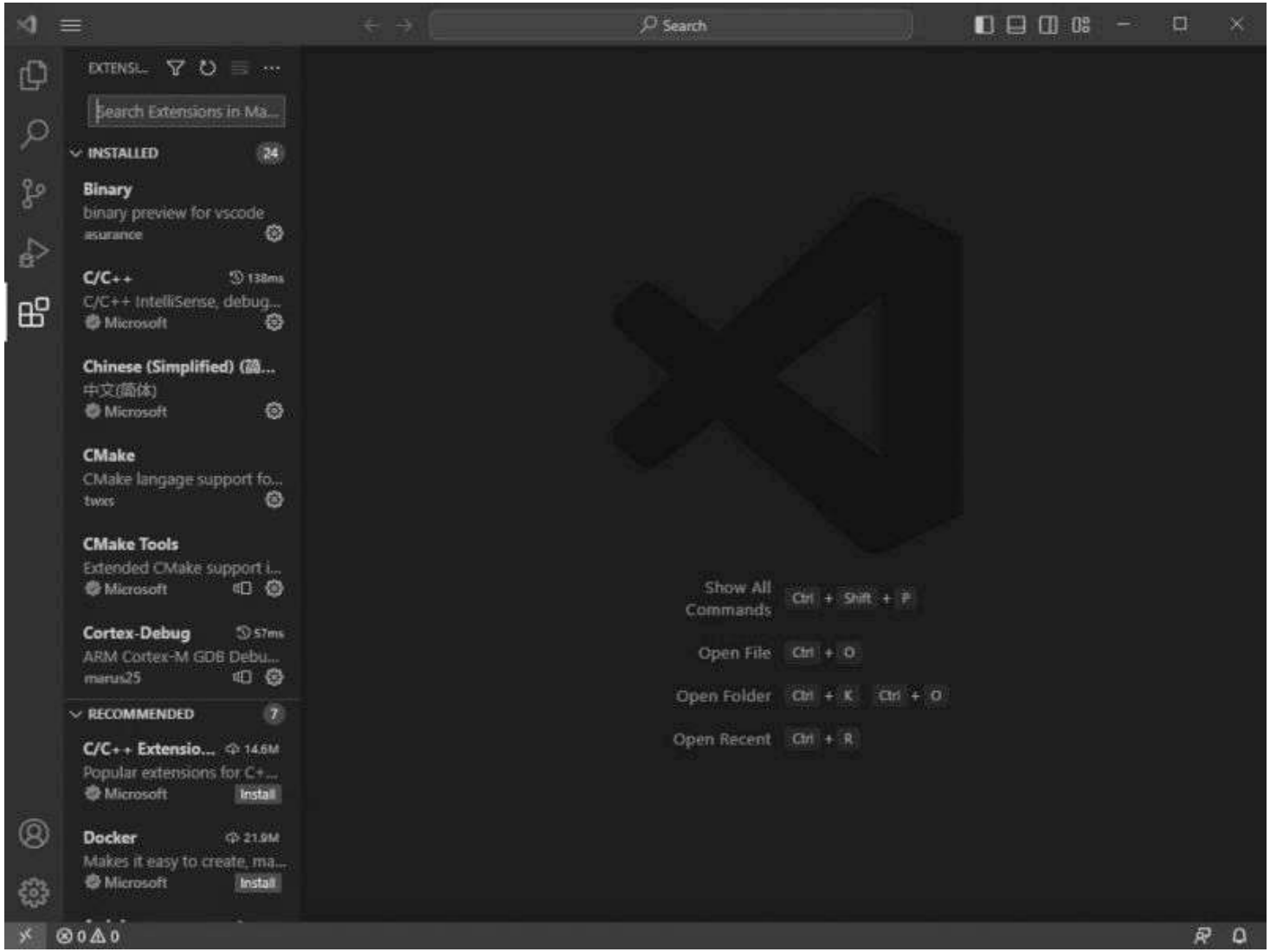
(/wiki/File:ESP32-S3-Pico_06.jpg)

- - After the first and second items are enabled, you can open VSCode directly by right-clicking files or directories, which can improve the subsequent user experience.
  - After the third item is enabled, you can select VSCode directly when you choose how to open it.

Install Espressif IDF Plug-in

- Note: The latest version of the current plug-in is V1.6.0, for a consistent experience, users can choose the same version as us.

1. Open VSCode and use the shortcut key Shift + Ctrl + X to enter the plugin manager.



(/wiki/File:ESP32-S3-Pico_07.jpg)

2. In the search bar, type Espressif IDF, select the corresponding plug-in, and click install.

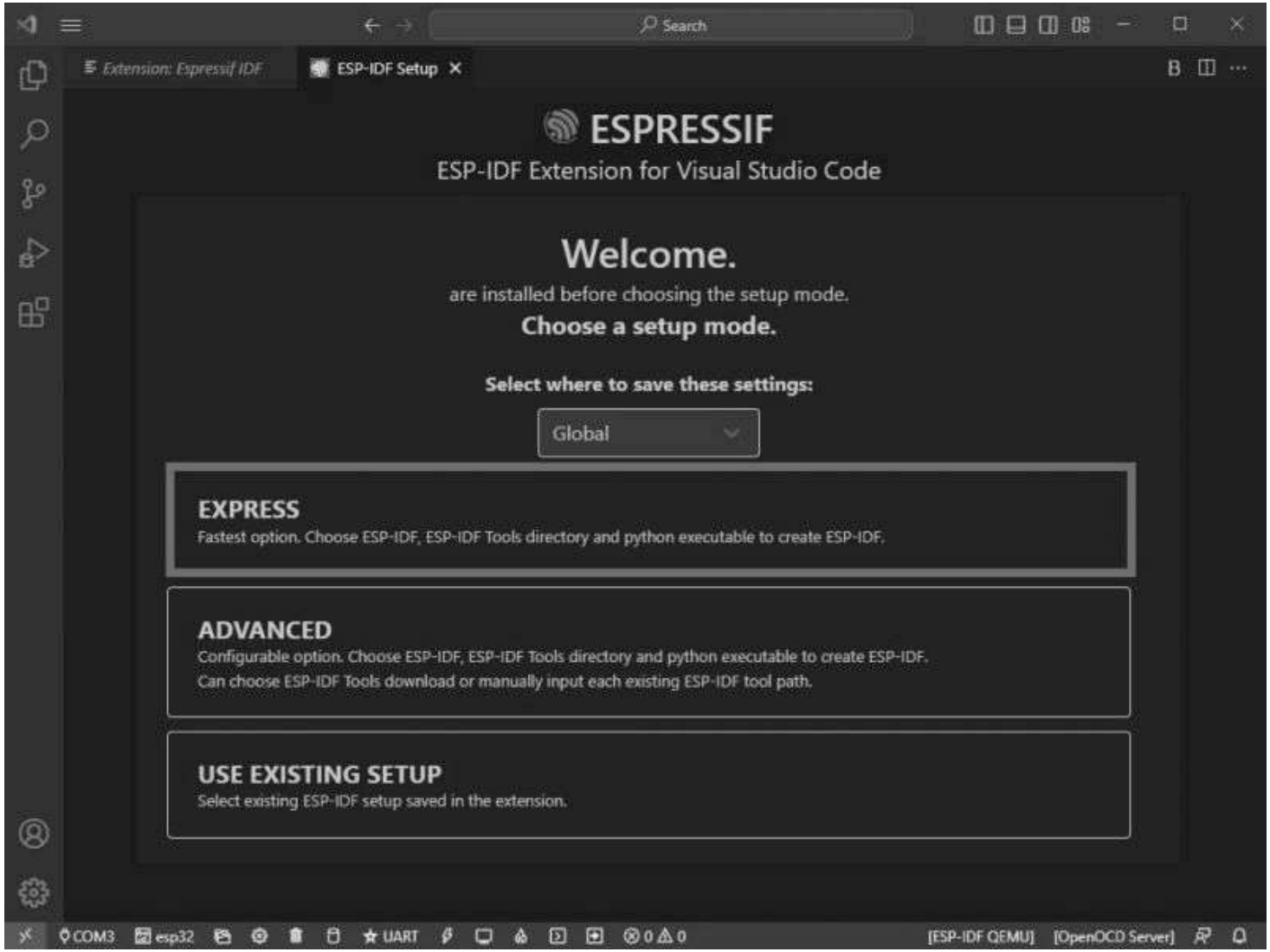
(/wiki/File:ESP32-S3-Pico_08.jpg)

3. Press F1 to enter:

```
esp-idf: configure esp-idf extension
```

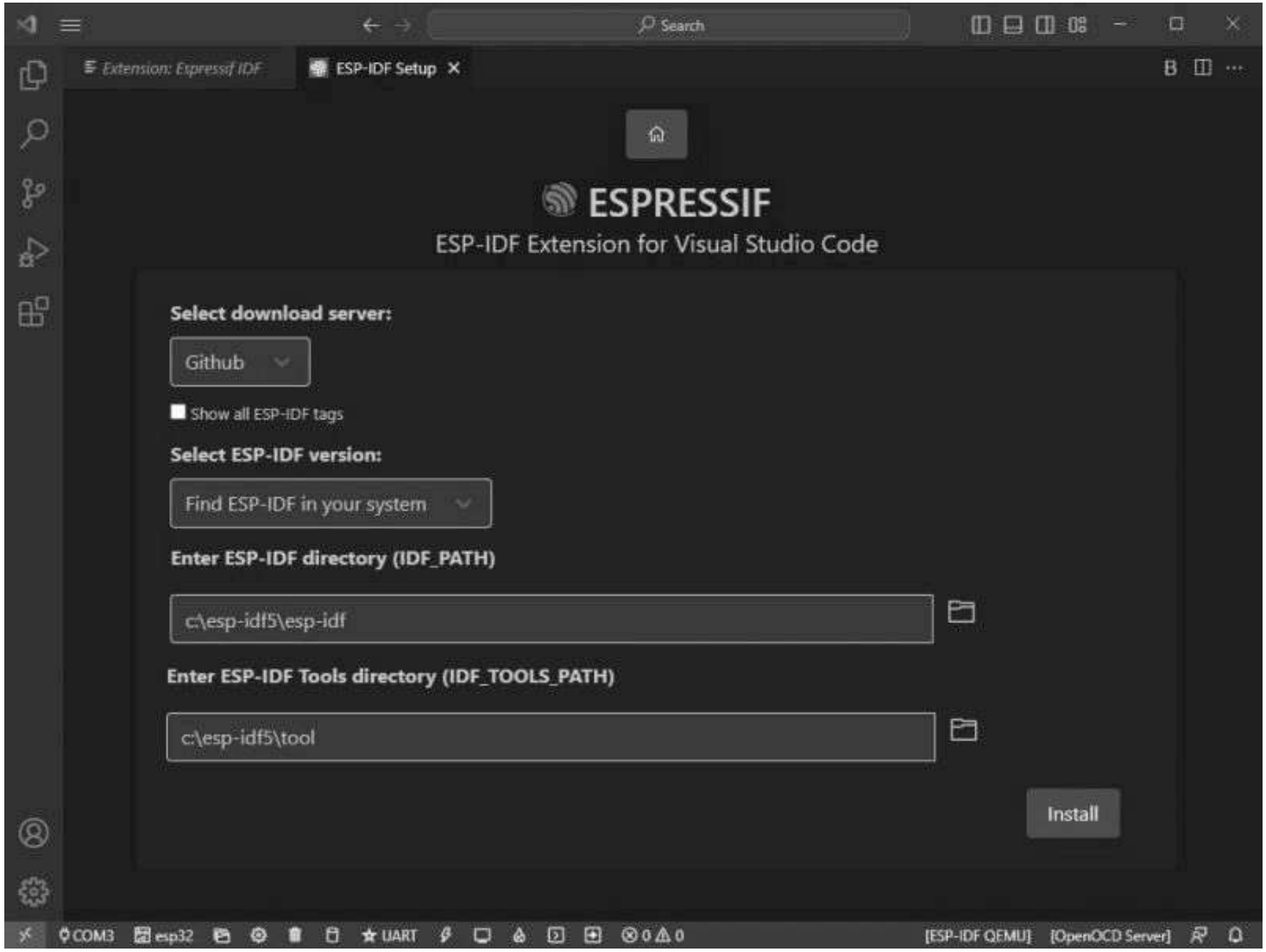
(/wiki/File:ESP32-S3-Pico_09.jpg)

4. Choose express (This tutorial is for first-time users, so only the first general installation tutorial is covered.)



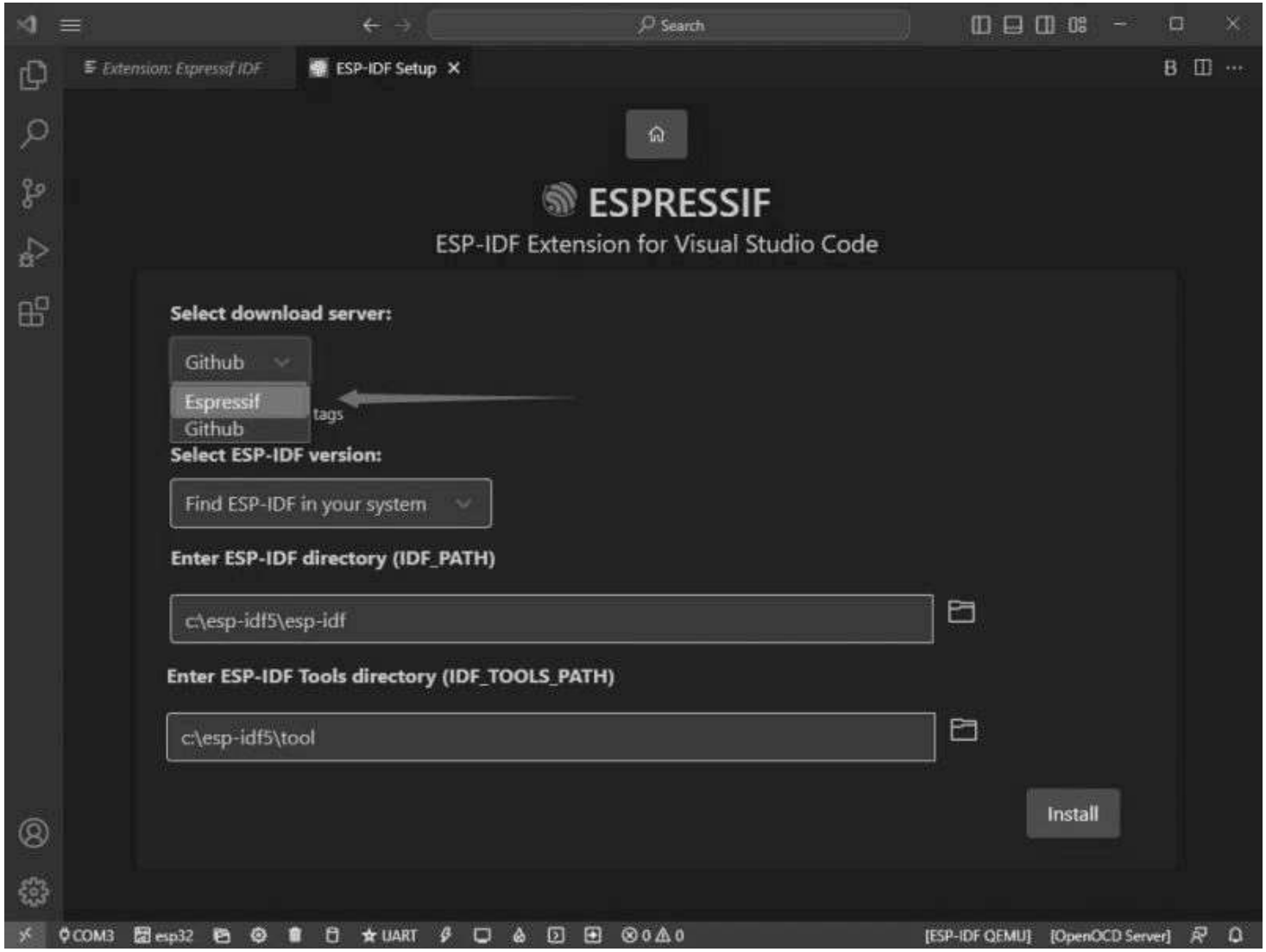
(/wiki/File:ESP32-S3-Pico_10.jpg)

5. Open and display this screen.



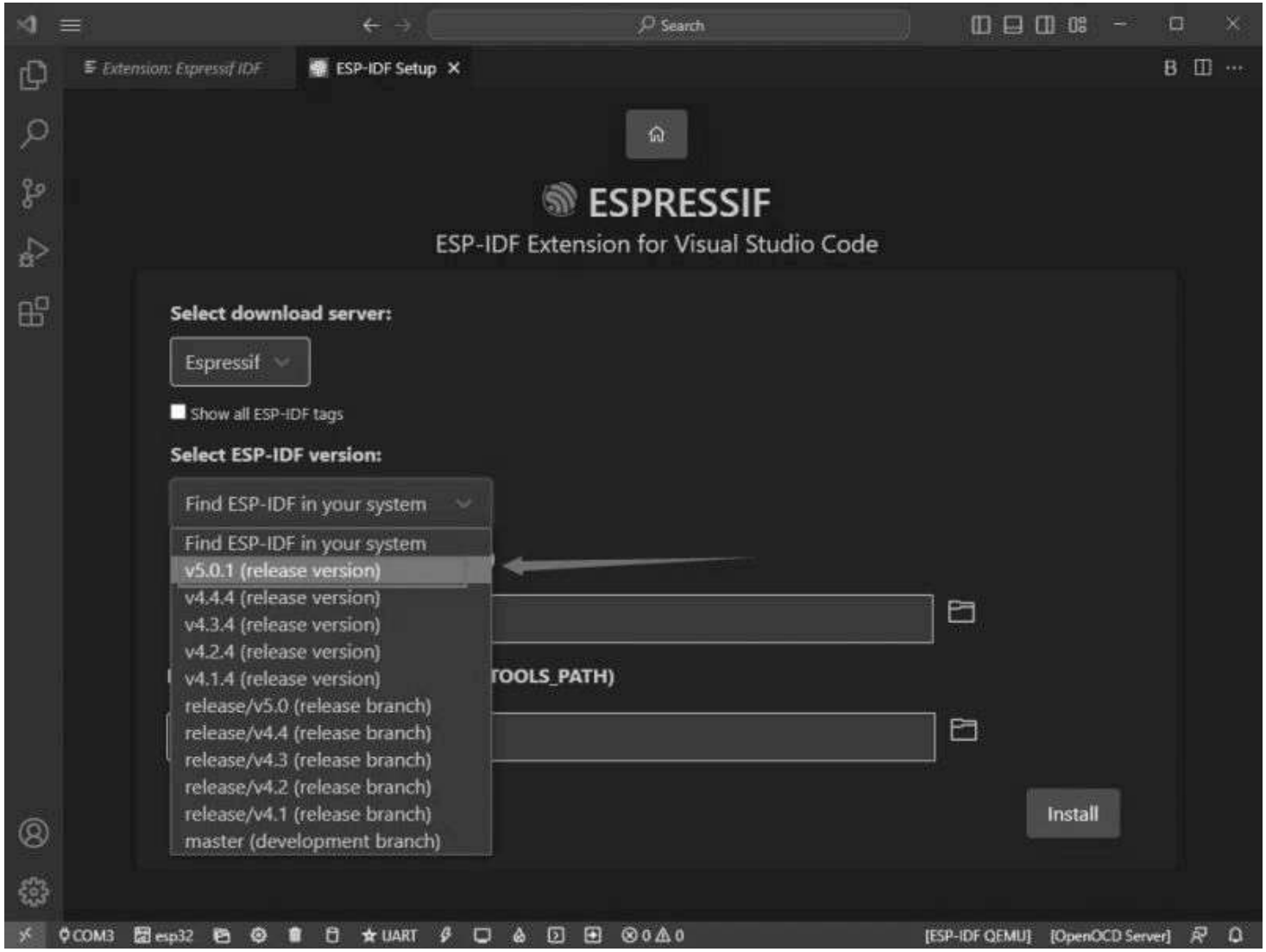
(/wiki/File:ESP32-S3-Pico_11.jpg)

6. Choose a server to download.



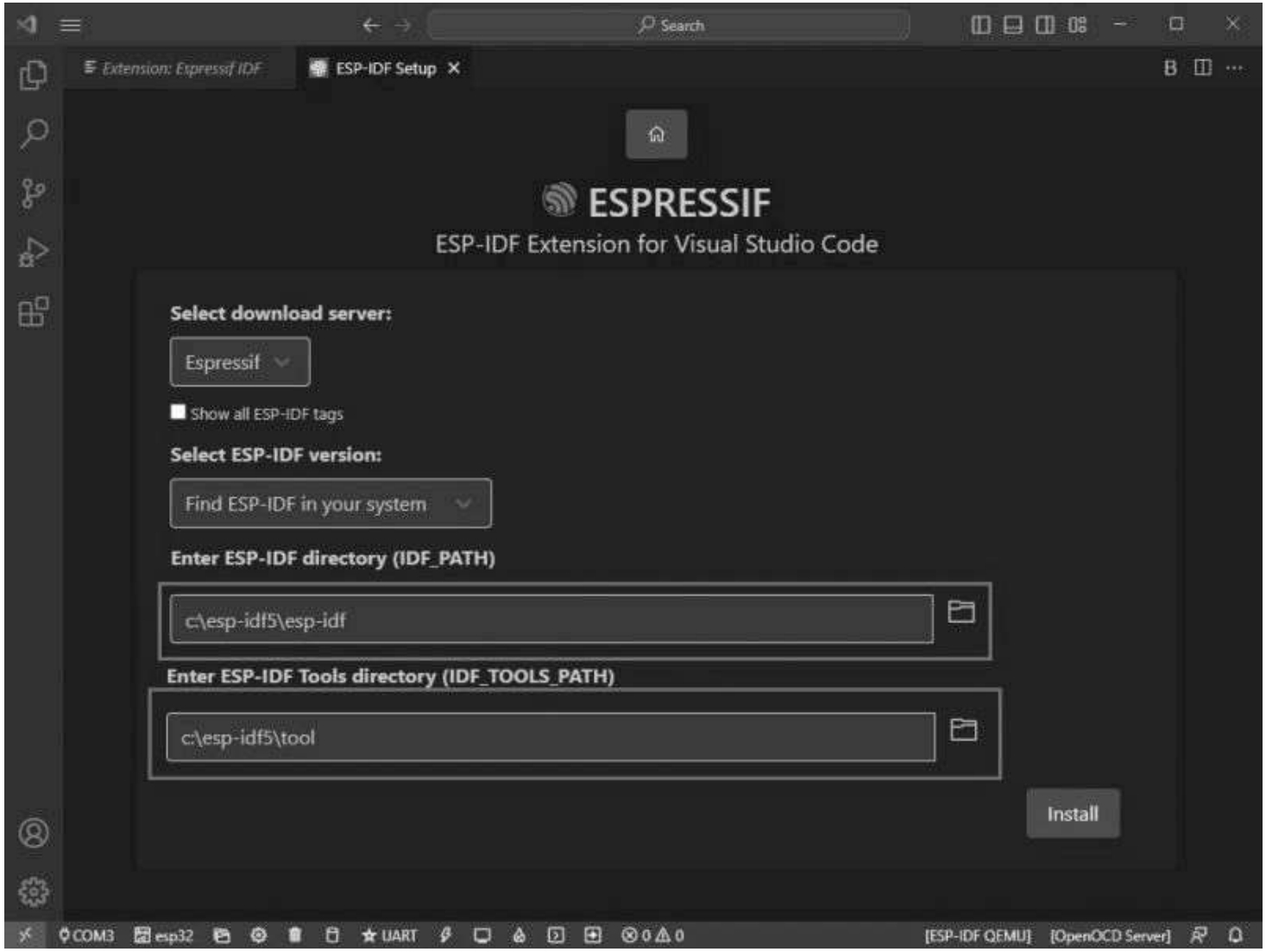
(/wiki/File:ESP32-S3-Pico_12.jpg)

7. Select the ESP-IDF version you want now, we choose the latest V5.0.1 (note that ESP-IDF started to support ESP32-S3 only after V4.4).



(/wiki/File:ESP32-S3-Pico_13.jpg)

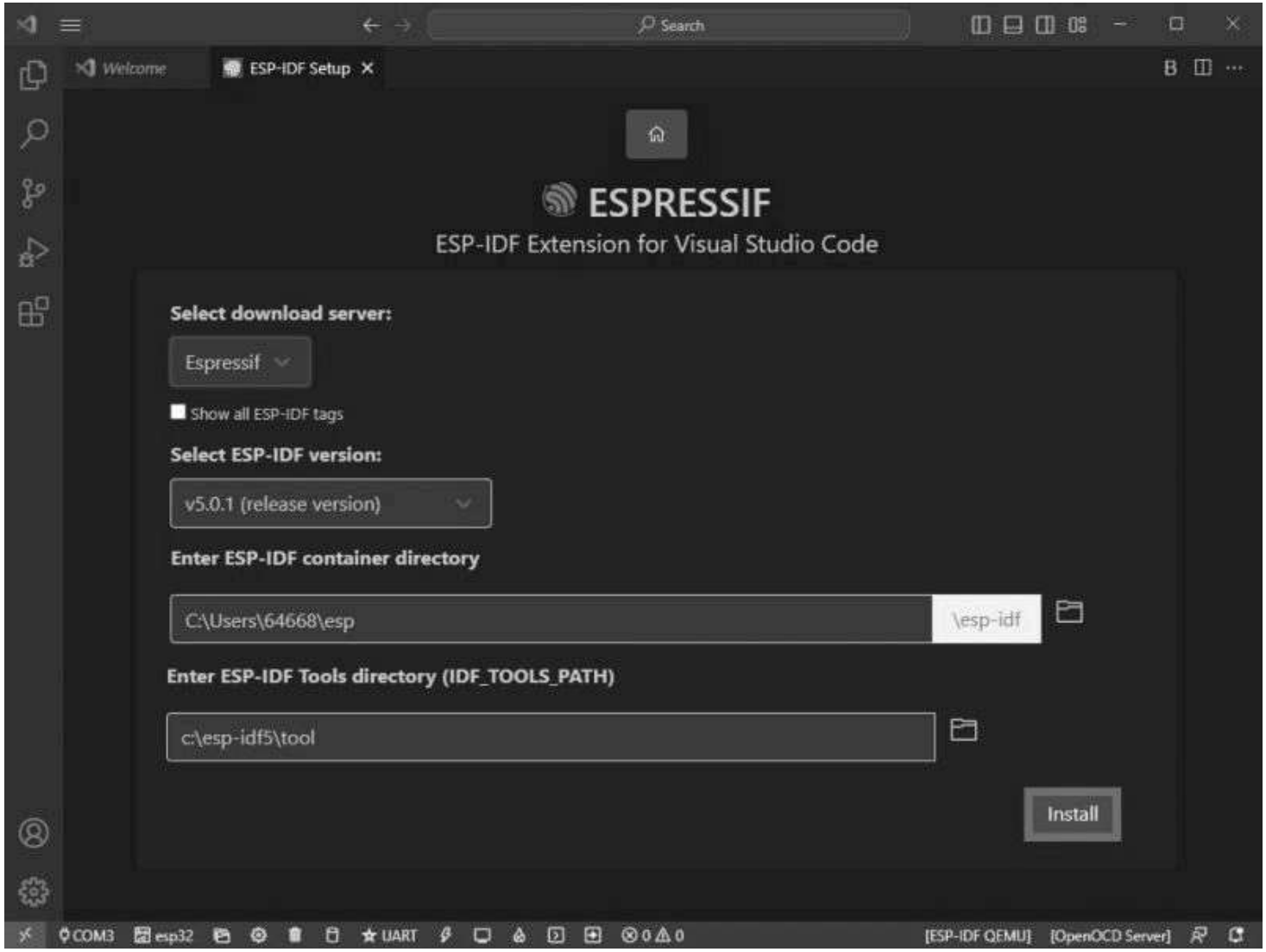
8. The following two are the ESP-IDF directory installation address and the ESP-IDF required tools installation address respectively.



(/wiki/File:ESP32-S3-Pico_14.jpg)

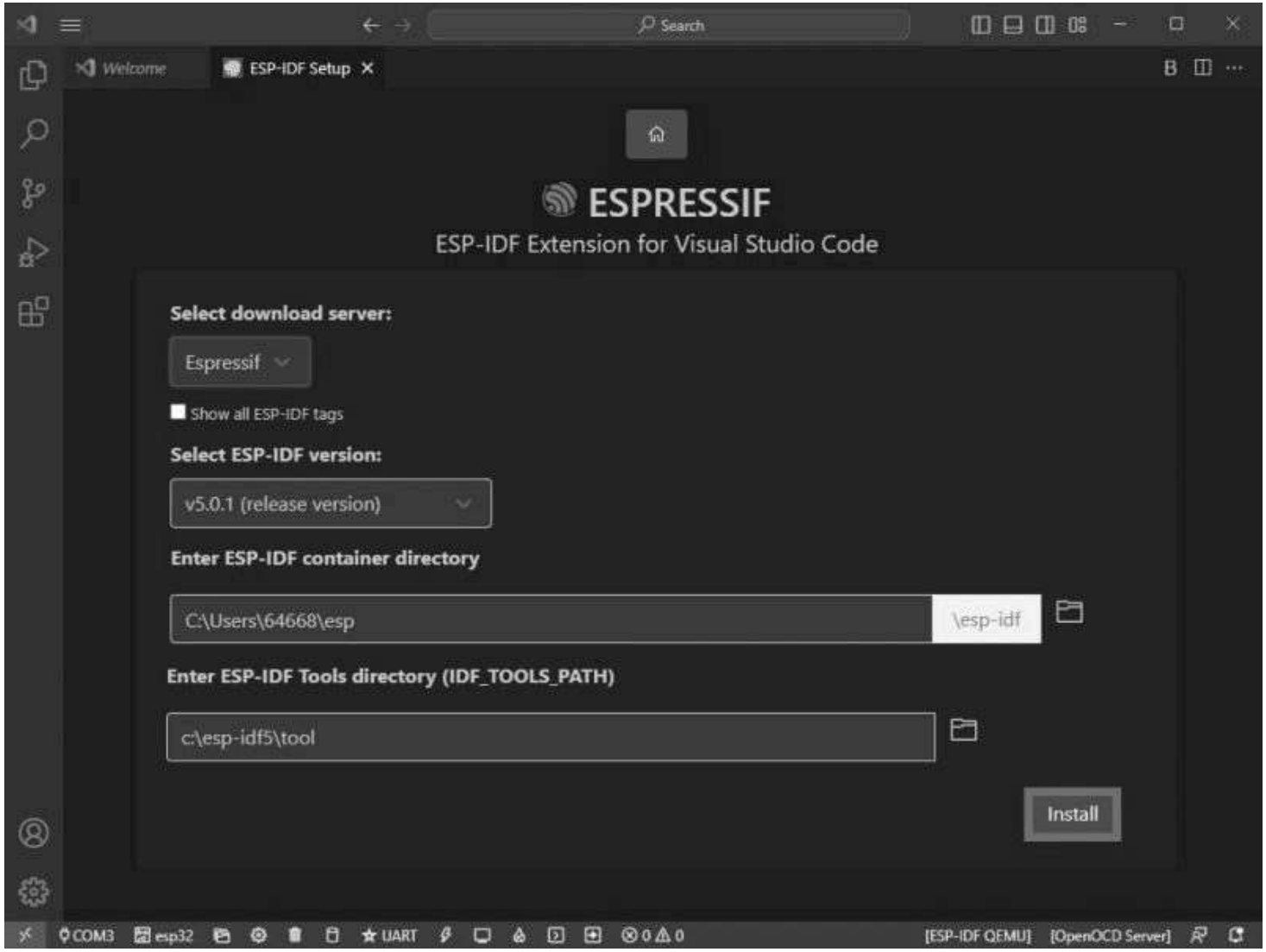
- **Note: If you have installed ESP-IDF before, or if it has failed, please make sure to delete the file completely or create a new path.**

9. Once the configuration is finished, click "install" to download.



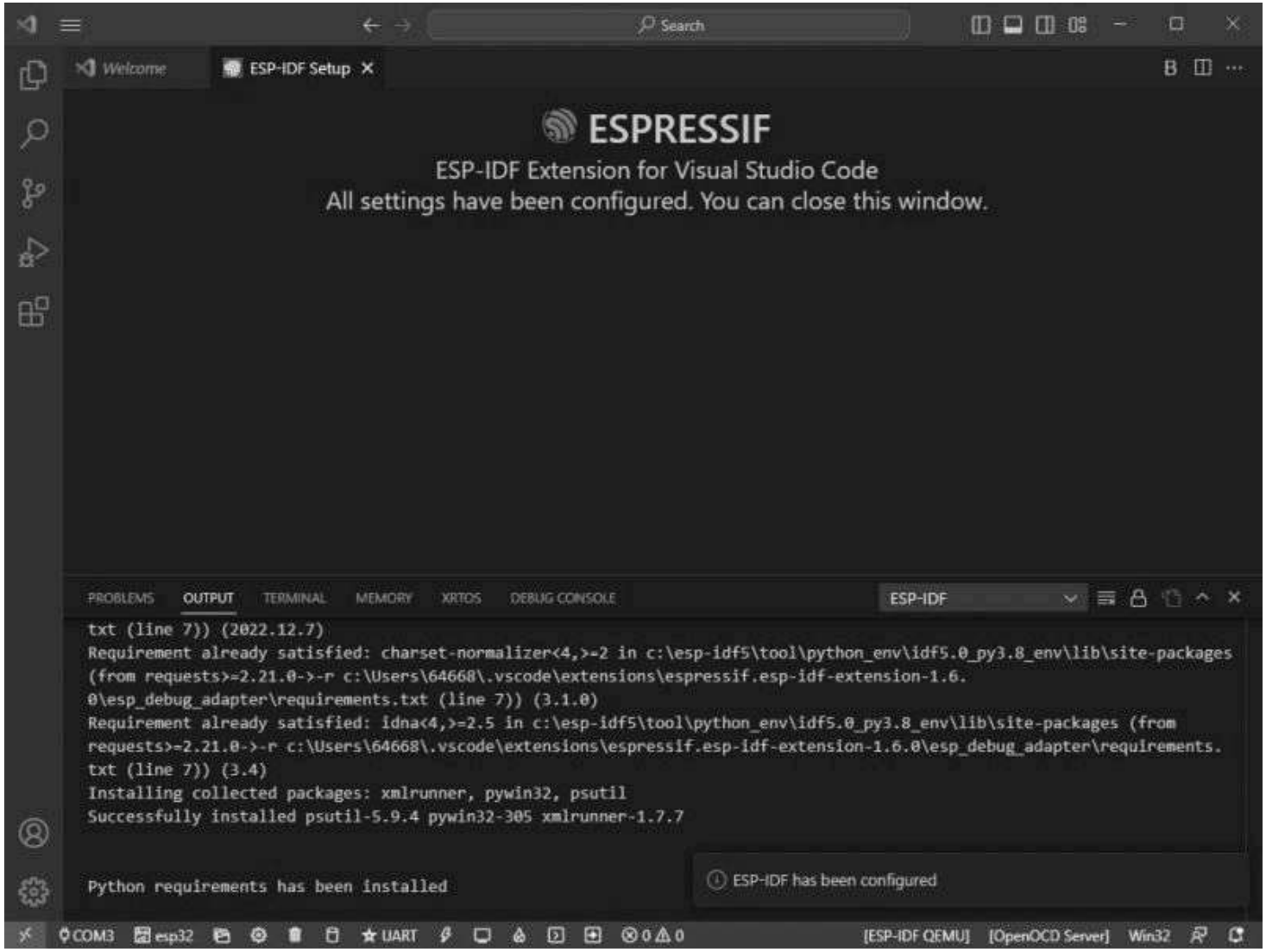
(/wiki/File:ESP32-S3-Pico_15.jpg)

10. Enter the download page, and it will automatically install the corresponding tools and environment, just wait a moment.



(/wiki/File:ESP32-S3-Pico_15.jpg)

11. After the installation is completed, it will enter the following screen, indicating that the installation is finished.



(/wiki/File:ESP32-S3-Pico_17.jpg)

Official Demo Usage

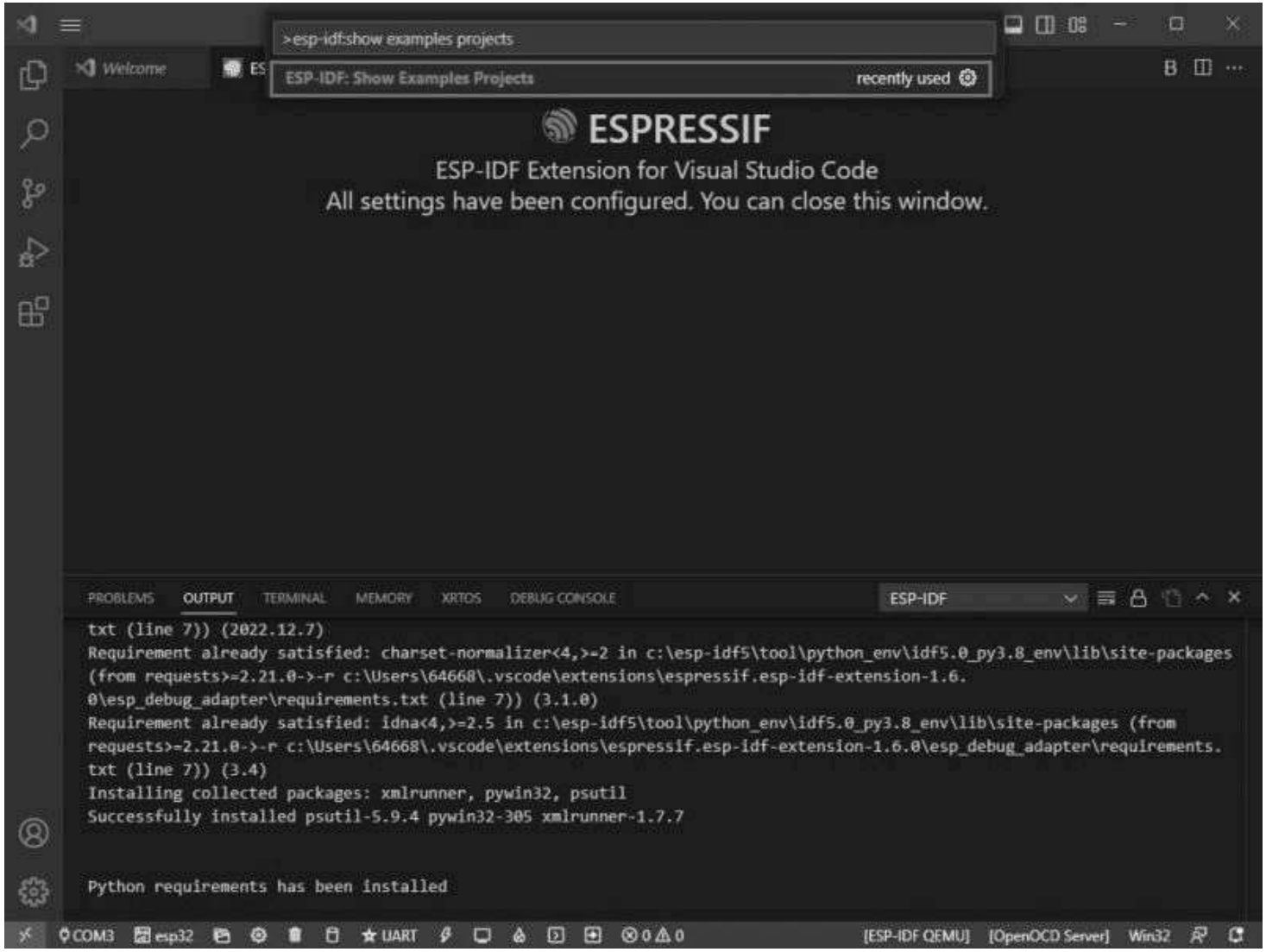
- Click here (https://github.com/espressif/esp-idf/tree/master/examples) to view more details

provided by the official ESP.

Creating a Demo

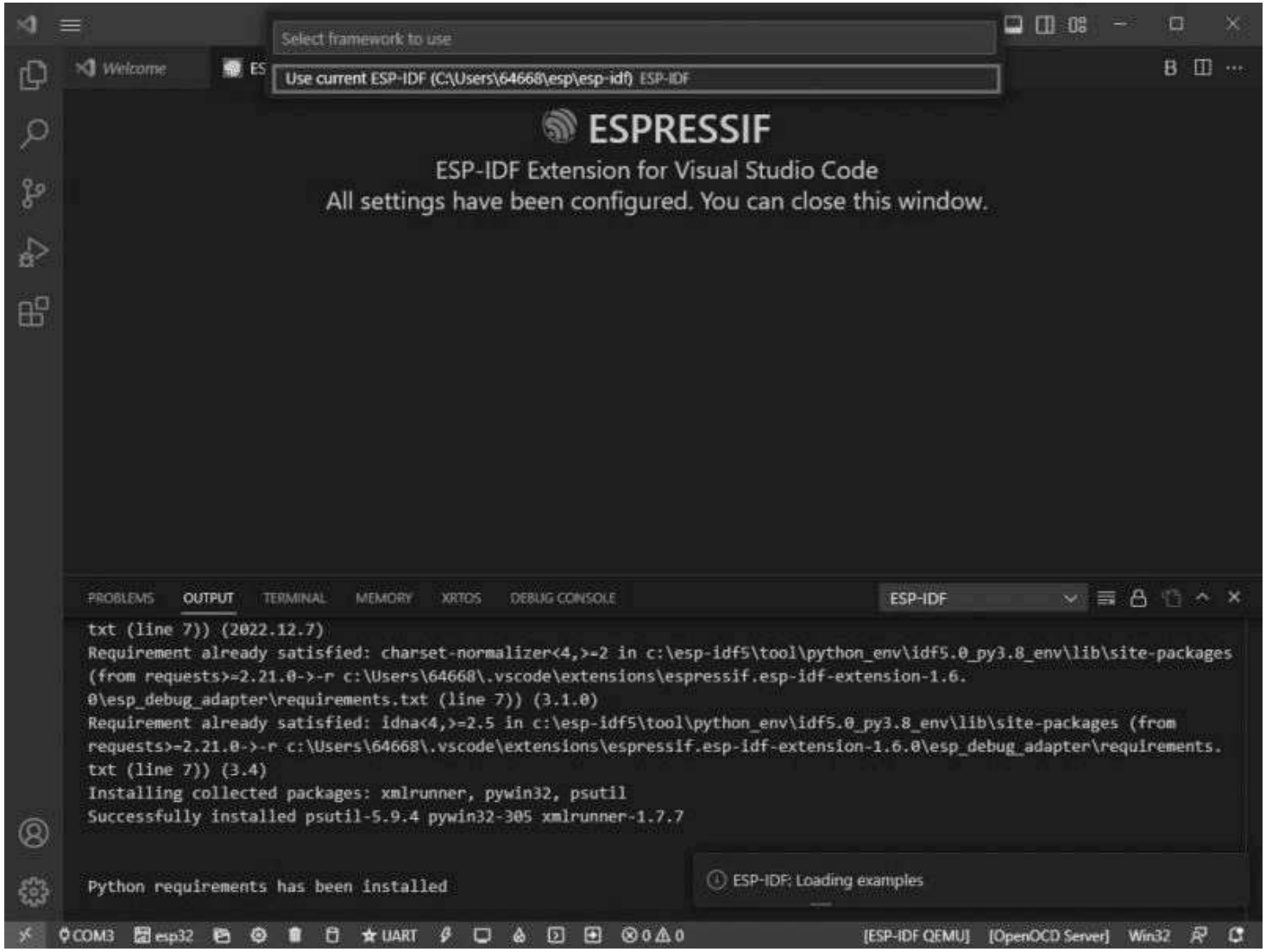
1. Using the shortcut F1, type:

```
esp-idf:show examples projects
```



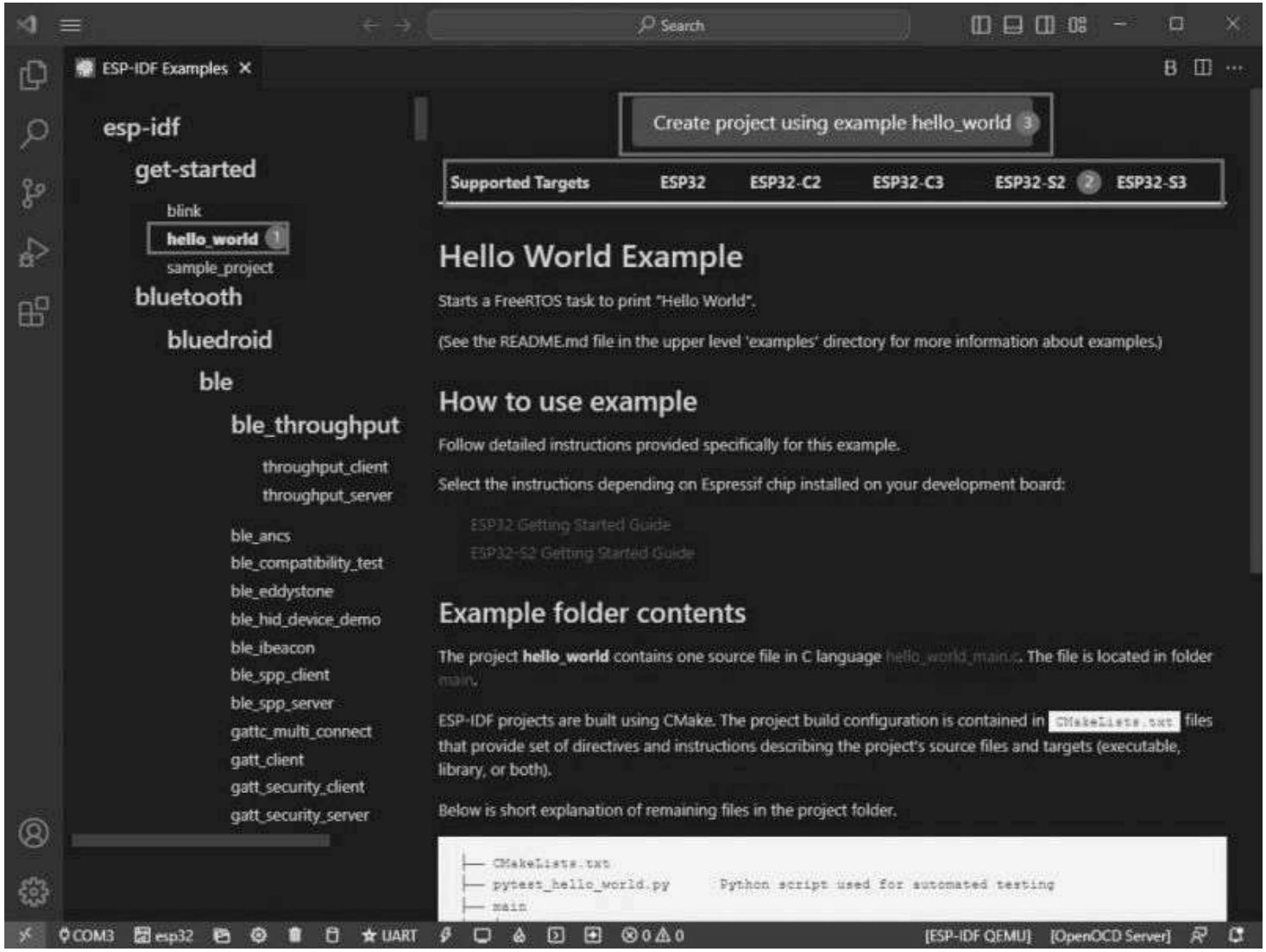
(/wiki/File:ESP32-S3-Pico_18.jpg)

2. Choose your current IDF version:



(/wiki/File:ESP32-S3-Pico_19.jpg)

3. Take "Hello World" as an example:



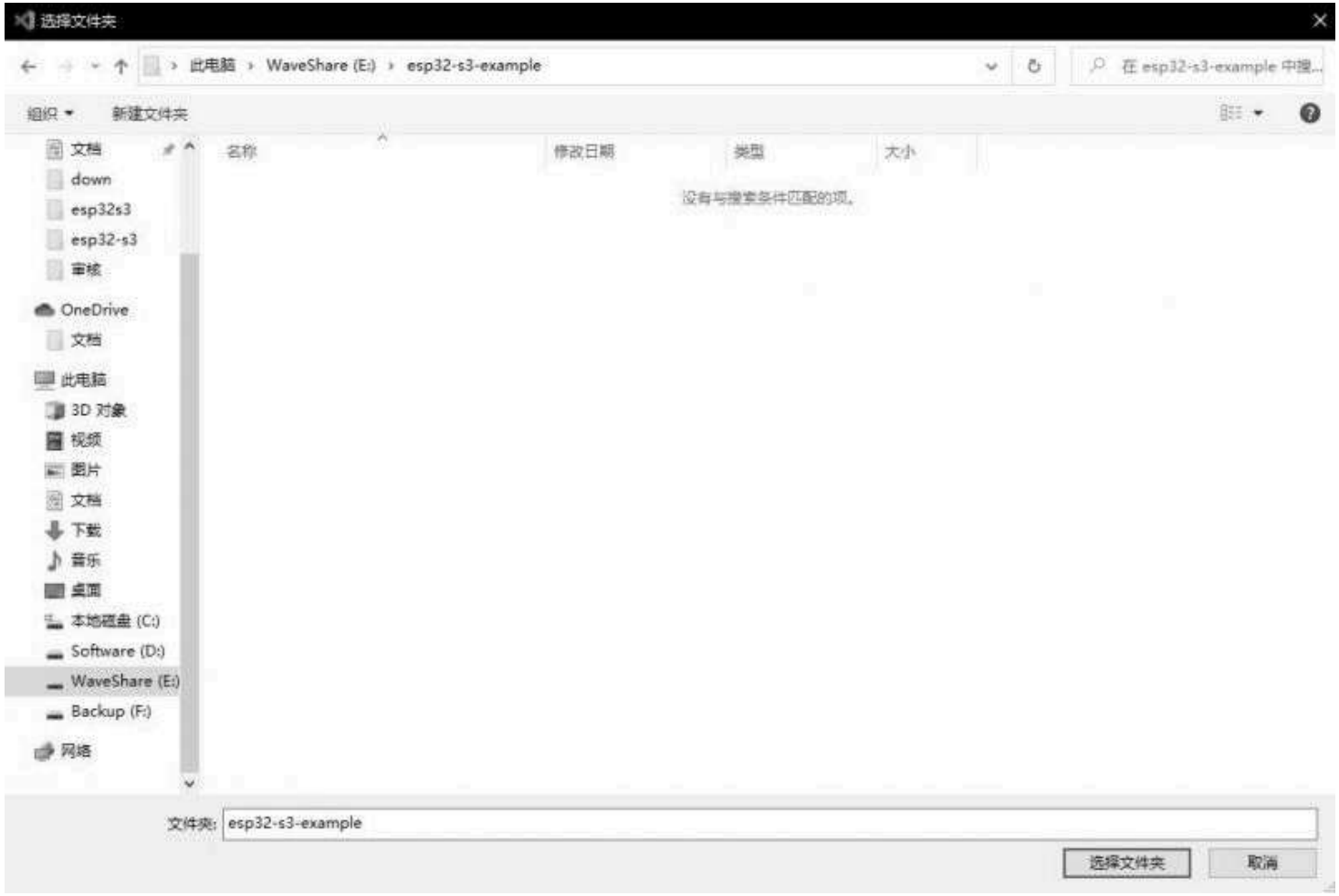
(/wiki/File:ESP32-S3-Pico_20.jpg)

4. Choose the corresponding demo.

5. The readme file will explain which chip the demo is suitable for (the following section will introduce how to use the demo and its file structure, which is omitted here).

6. Click to create the demo.

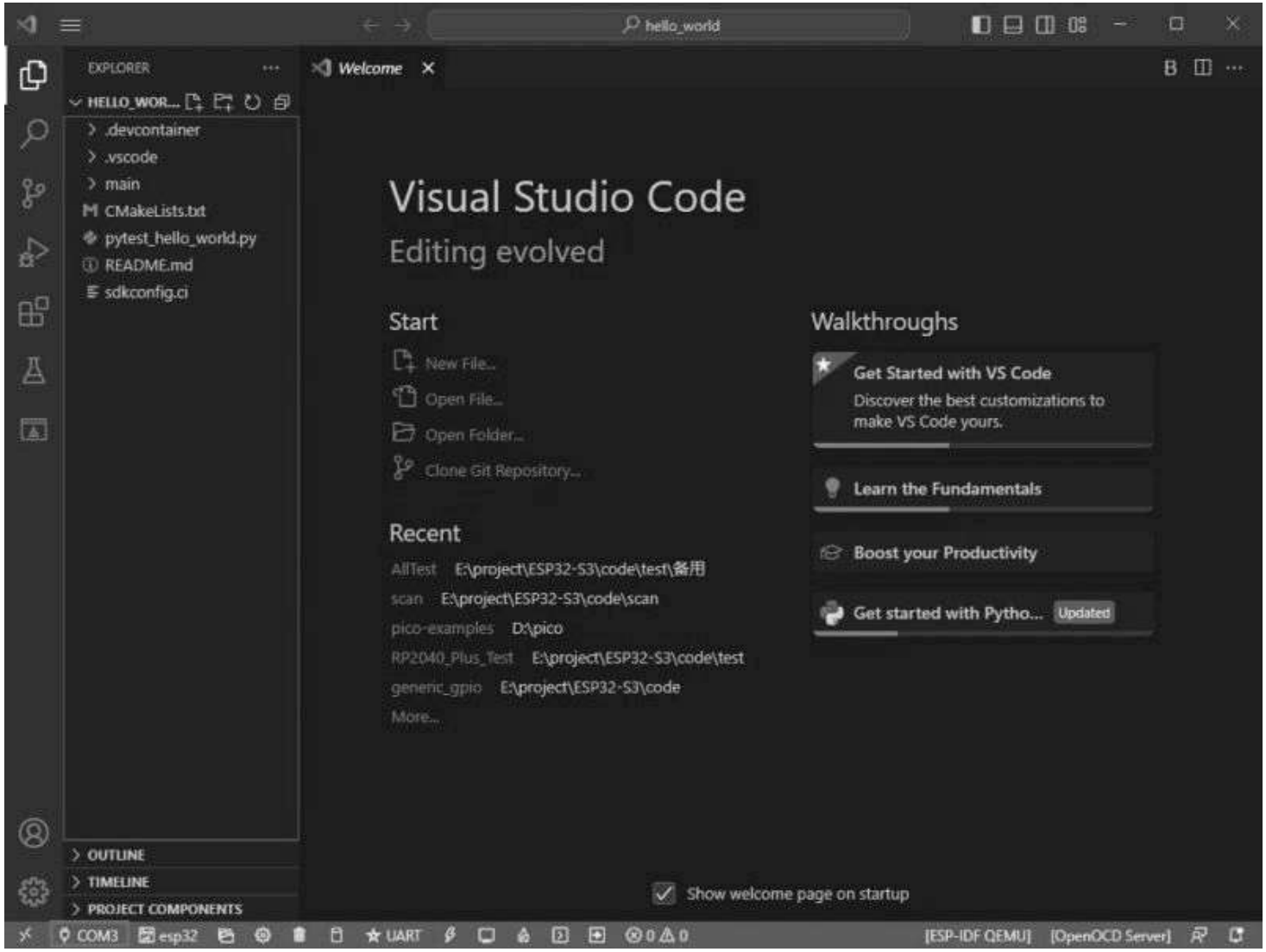
7. Choose the path to place the demo and ensure that there is no folder with the same name as the demo.



(/wiki/File:ESP32-S3-Pico_21.jpg)

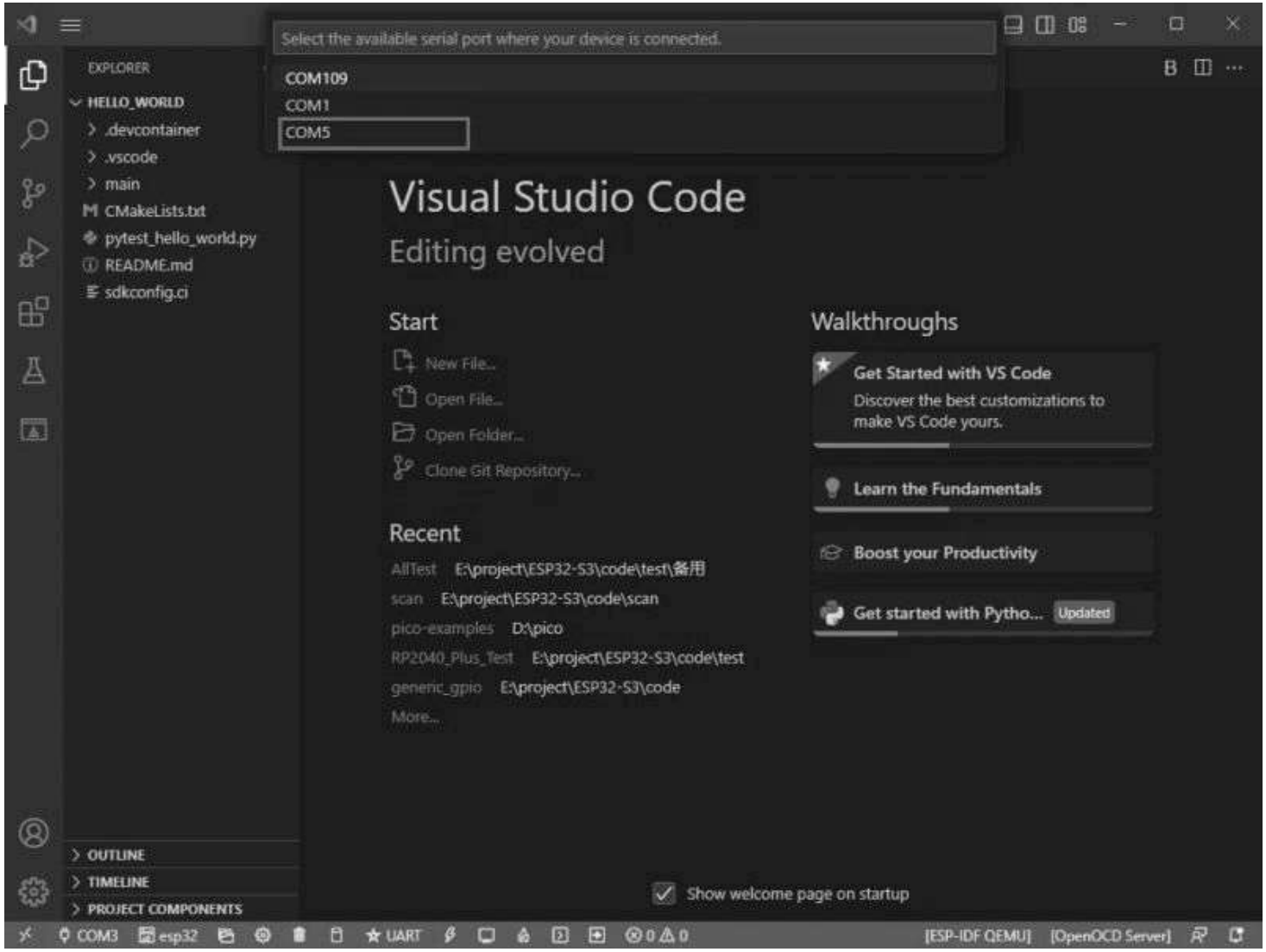
Modify COM Port

1. The corresponding COM port is displayed here, click on it to modify.



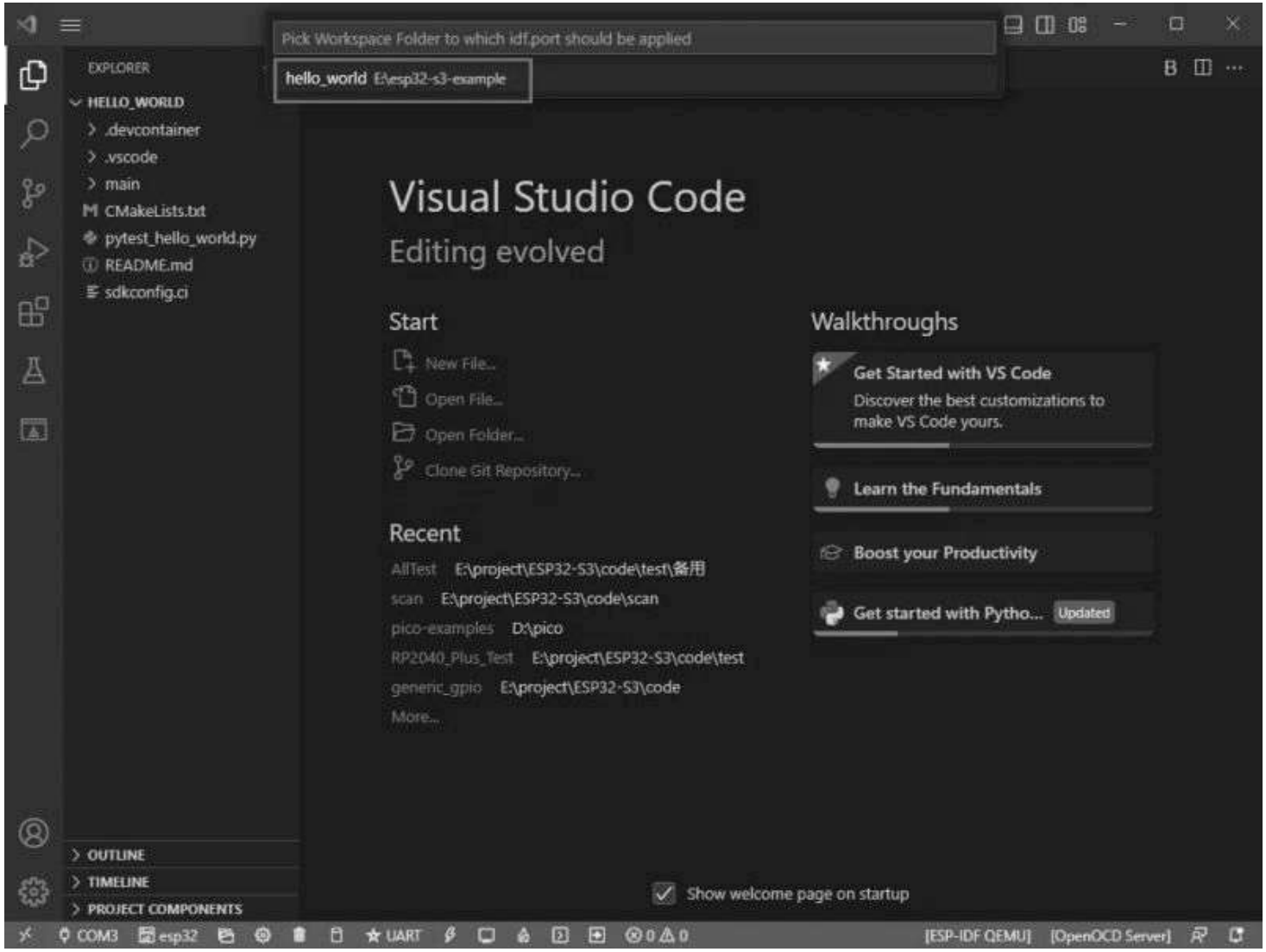
(/wiki/File:ESP32-S3-Pico_22.jpg)

2. We check the device manager COM port, and select COM5, please select your corresponding COM port:



(/wiki/File:ESP32-S3-Pico_23.jpg)

3. Choose the project and demo.

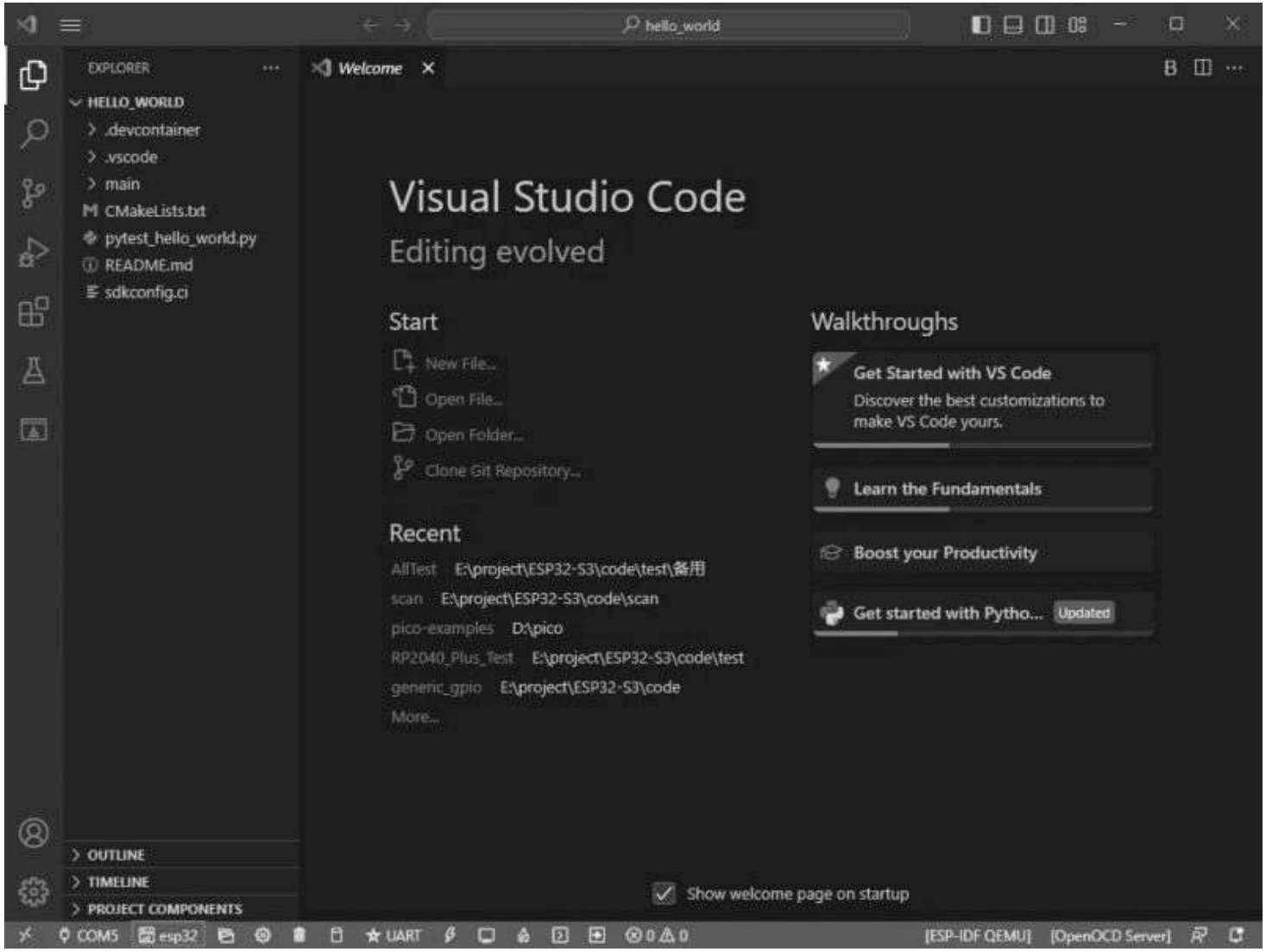


(/wiki/File:ESP32-S3-Pico_24.jpg)

4. Then the COM port is modified.

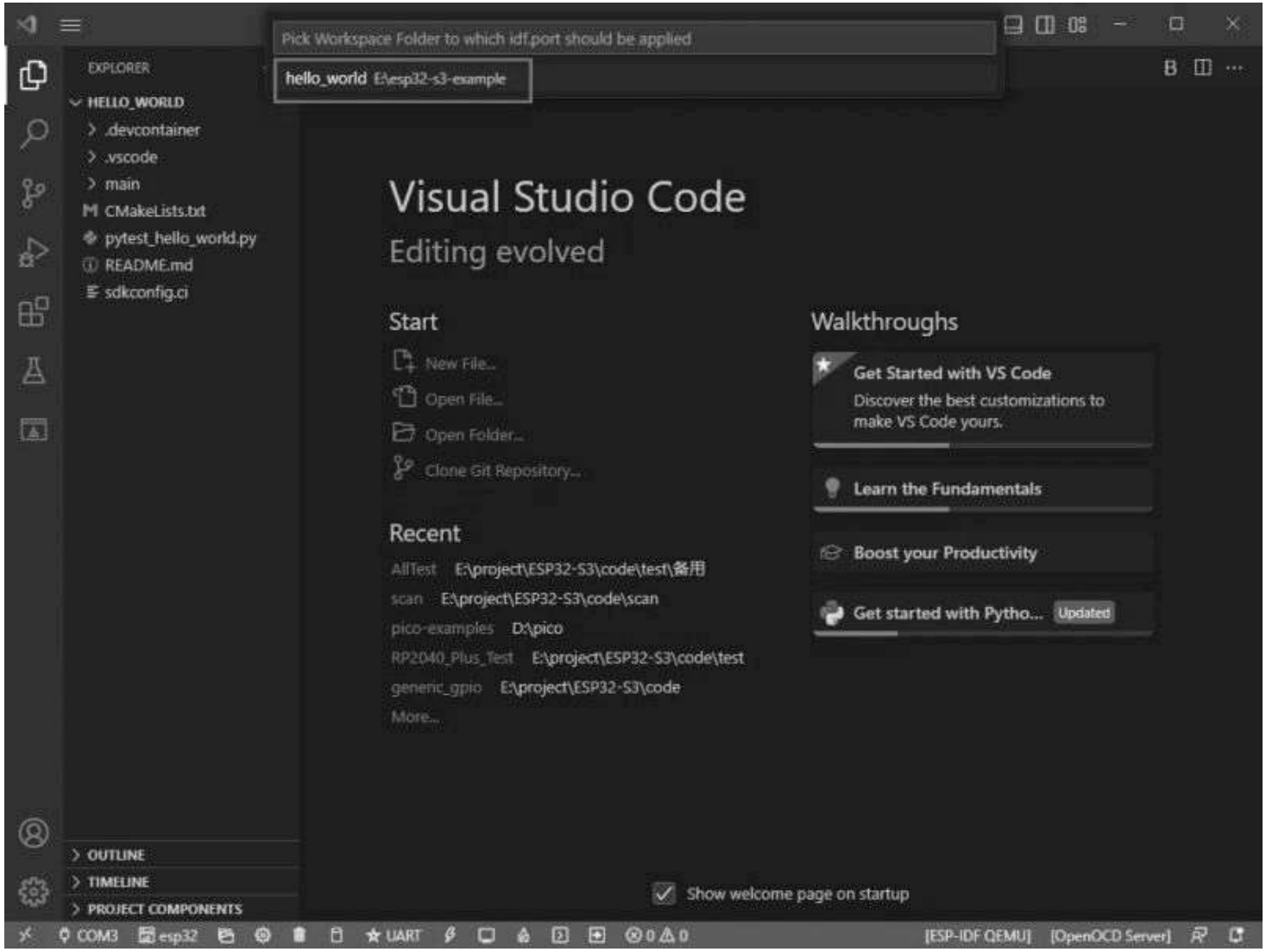
Modify the Driver

1. Here shows the driver used, click here to modify the corresponding driver:



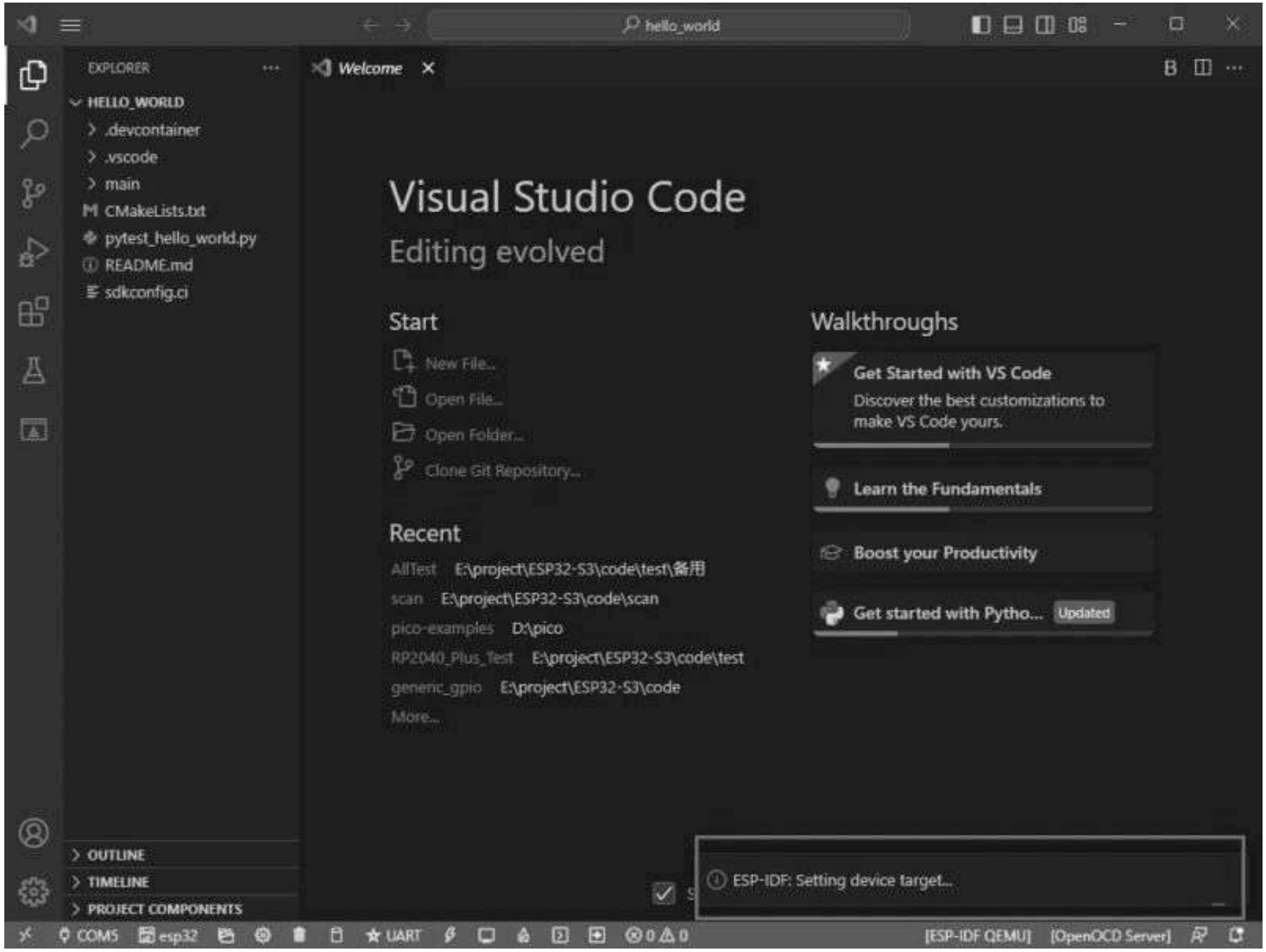
(/wiki/File:ESP32-S3-Pico_25.jpg)

2. Choose the project or demo:



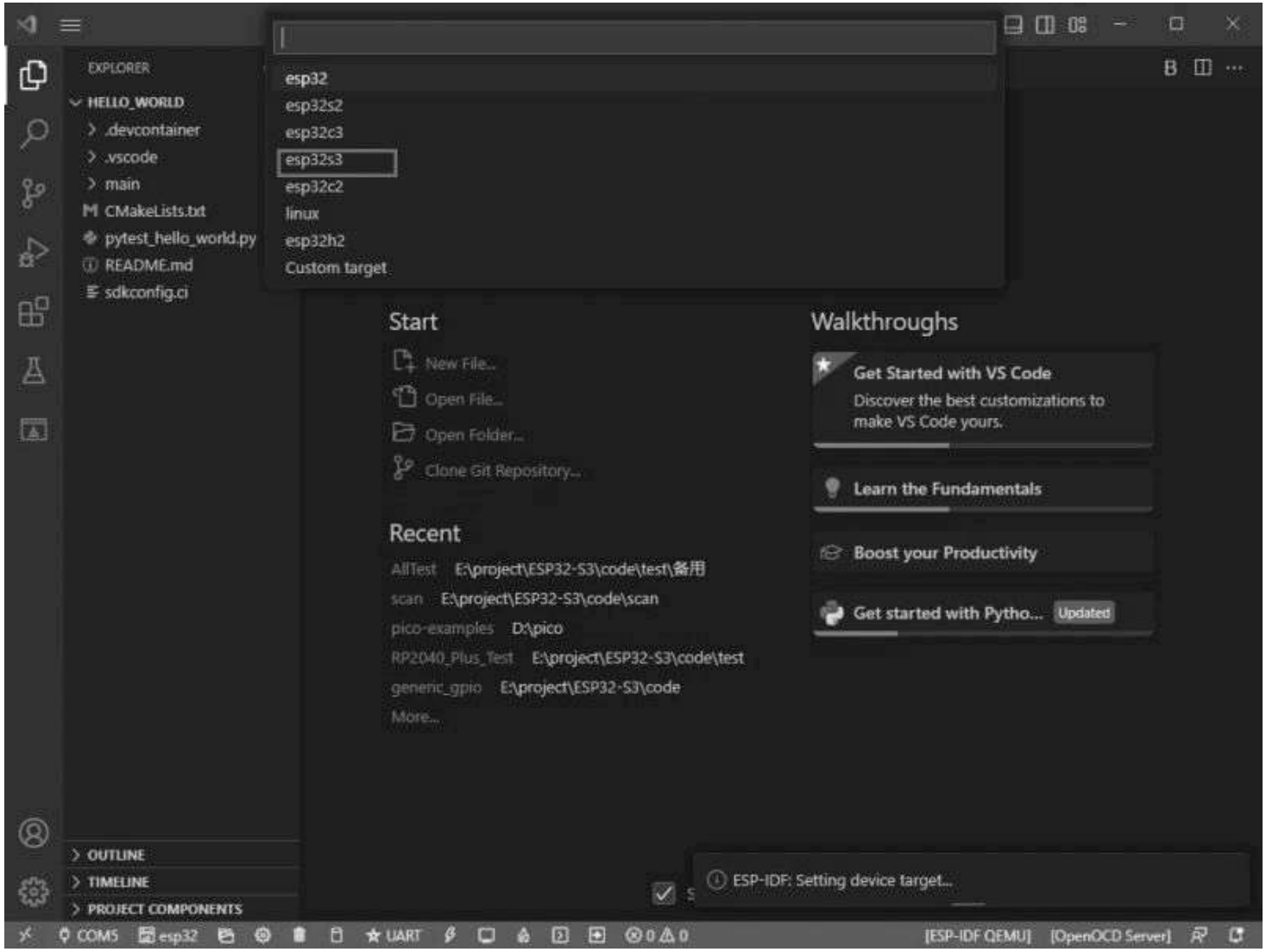
(/wiki/File:ESP32-S3-Pico_24.jpg)

3. Wait for a few seconds after clicking.



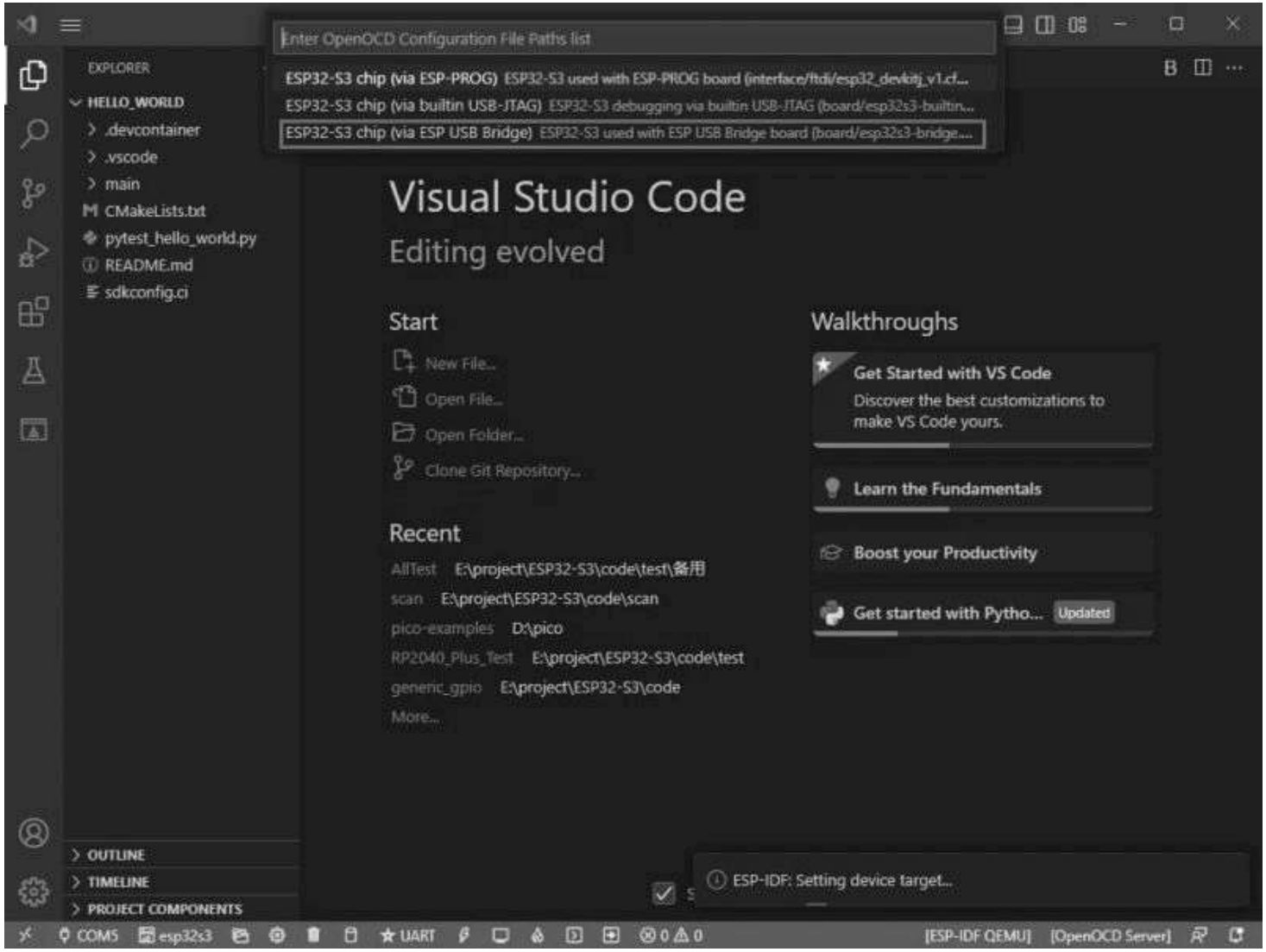
(/wiki/File:ESP32-S3-Pico_27.jpg)

4. Choose the driver we need, that is, the main chip ESP32S3.



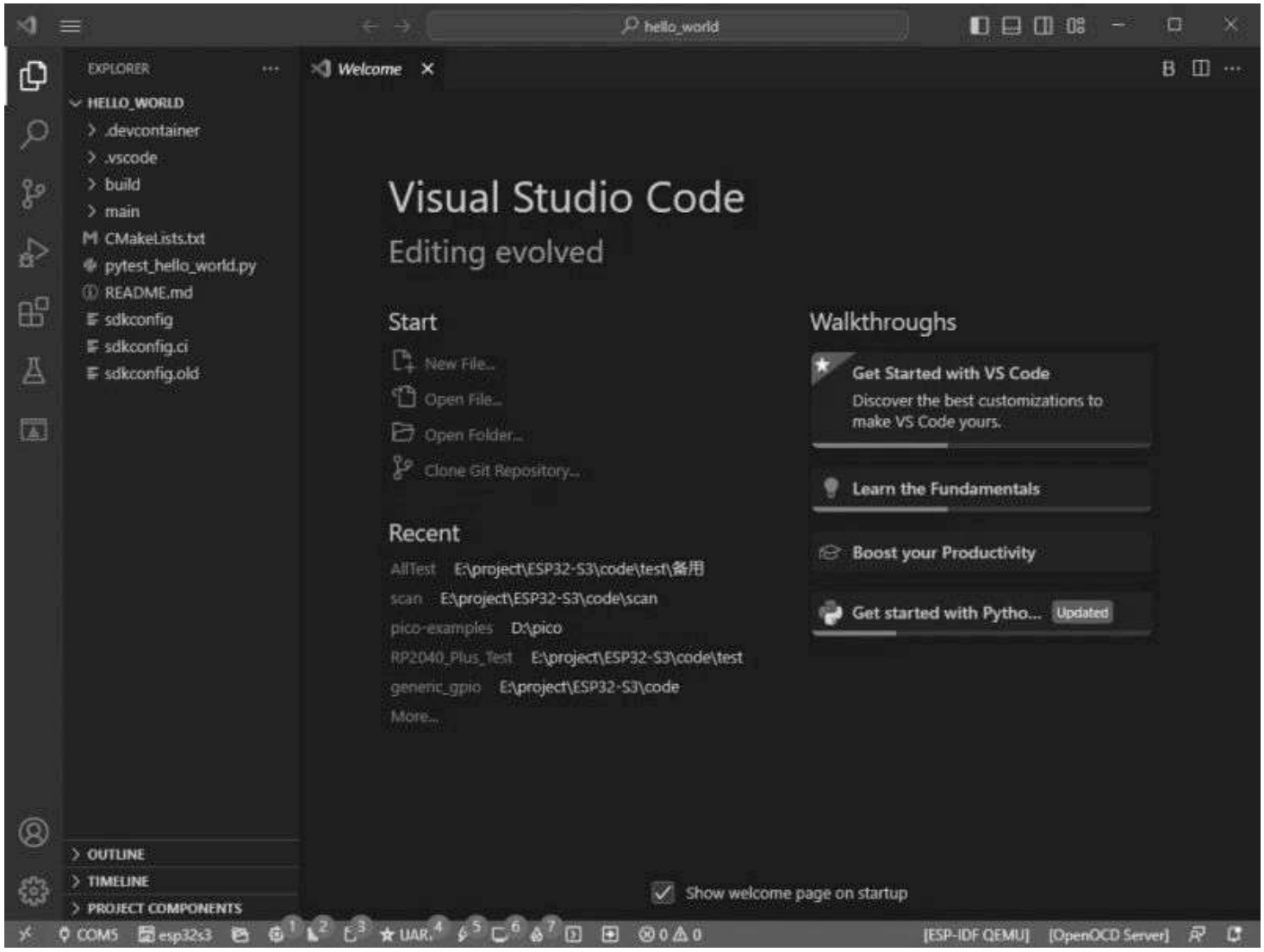
(/wiki/File:ESP32-S3-Pico_28.jpg)

5. Choose the openocd path, we can just choose one at random as it doesn't matter.


Enter OpenOCD Configuration File Paths list
ESP32-S3 chip (via ESP-PROG) ESP32-S3 used with ESP-PROG board (interface/ftdi/esp32_devkitj_v1.cf...
ESP32-S3 chip (via builtin USB-JTAG) ESP32-S3 debugging via builtin USB-JTAG (board/esp32s3-builtin...
ESP32-S3 chip (via ESP USB Bridge) ESP32-S3 used with ESP USB Bridge board (board/esp32s3-bridge....
EXPLORER
HELLO_WORLD
.devcontainer
.vscode
main
CMakeLists.txt
pytest_hello_world.py
README.md
sdkconfig.ci
Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Recent
AllTest E:\project\ESP32-S3\code\test\备用
scan E:\project\ESP32-S3\code\scan
pico-examples D:\pico
RP2040_Plus_Test E:\project\ESP32-S3\code\test
generic_gpio E:\project\ESP32-S3\code
More...
Walkthroughs
Get Started with VS Code
Discover the best customizations to make VS Code yours.
Learn the Fundamentals
Boost your Productivity
Get started with Pytho... Updated
OUTLINE
TIMELINE
PROJECT COMPONENTS
ESP-IDF: Setting device target...
COM5 esp32s3 UART [ESP-IDF QEMU] [OpenOCD Server]

(/wiki/File:ESP32-S3-Pico_29.jpg)

The Rest of the Status Bar Introduction



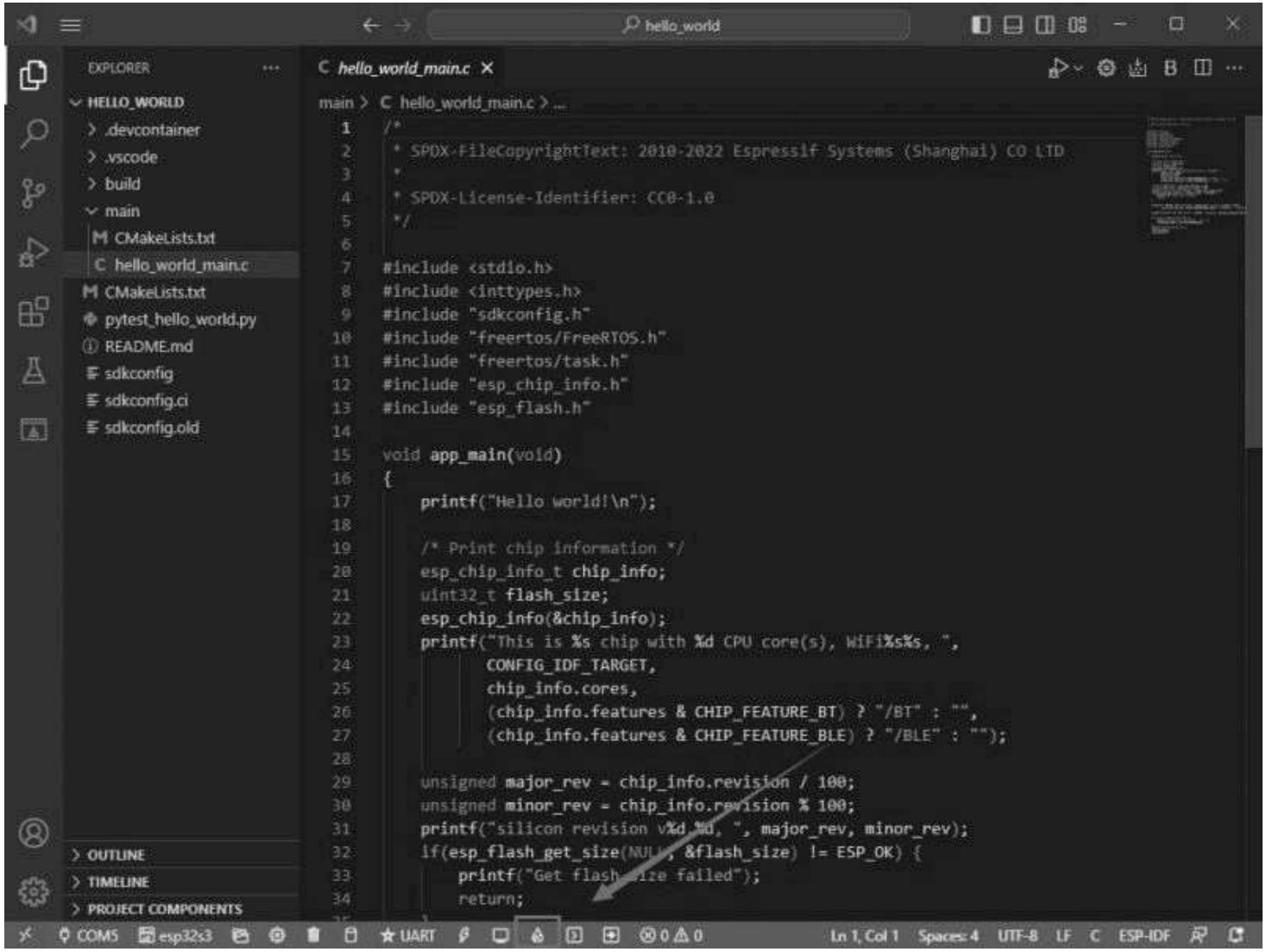
(/wiki/File:ESP32-S3-Pico_30.jpg)

- ① SDK configuration editor: many functions and configurations of ESP-IDF can be modified within it.

- ② Clean up everything and delete all compiled files.
- ③ Compile.
- ④ Current download method, default is UART.
- ⑤ Program the current firmware, please do it after compiling.
- ⑥ Open the serial monitor to view serial information.
- ⑦ Combined button for compiling, programming, and opening the serial monitor (most commonly used during debugging).

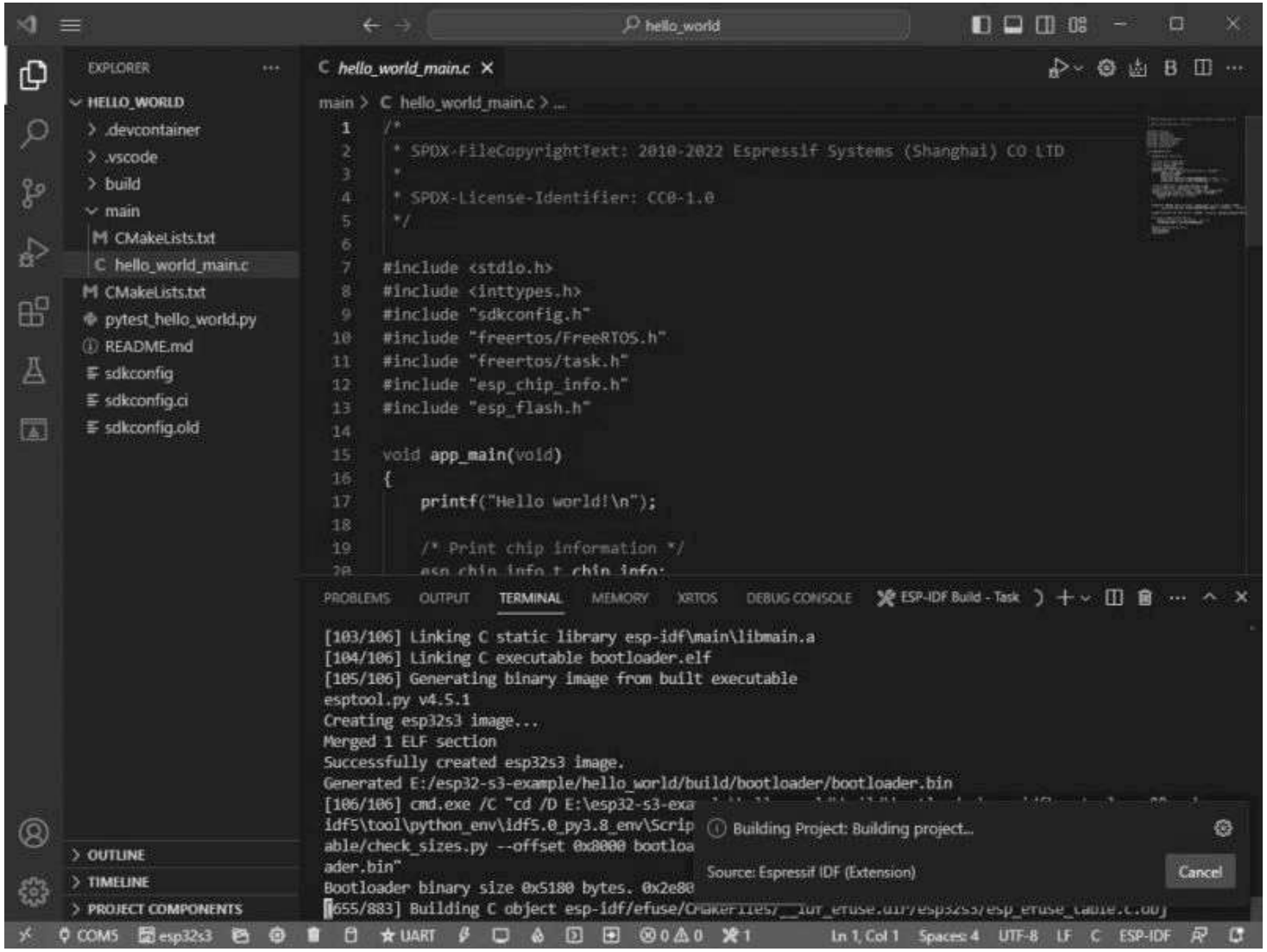
Compile, Program, and Serial Port Monitoring

1. Click on the Compile, Program, and Open Serial Monitor buttons we described earlier.



(/wiki/File:ESP32-S3-Pico_31.jpg)

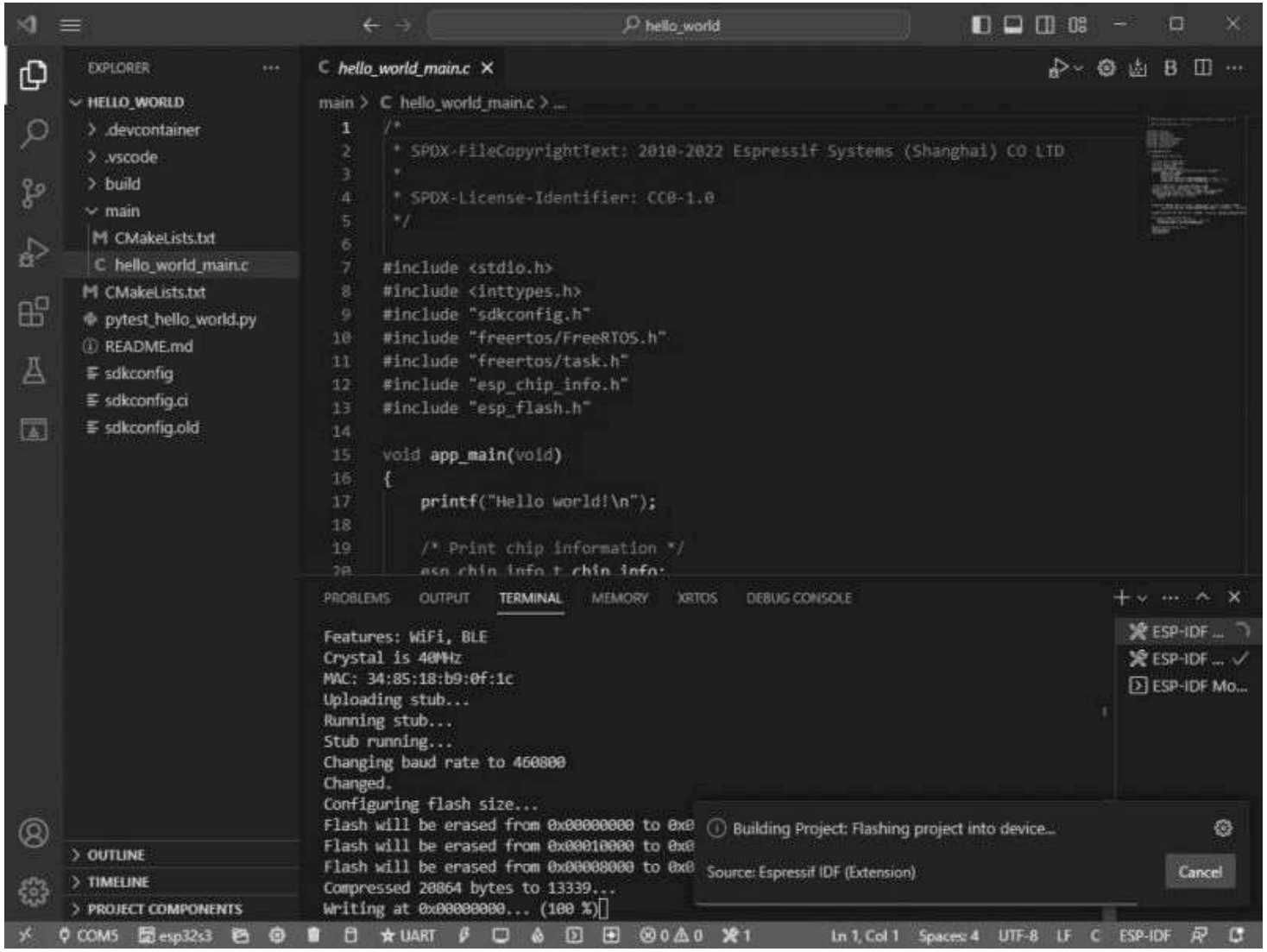
2. It may take a long time to compile, especially for the first time.



(/wiki/File:ESP32-S3-Pico_32.jpg)

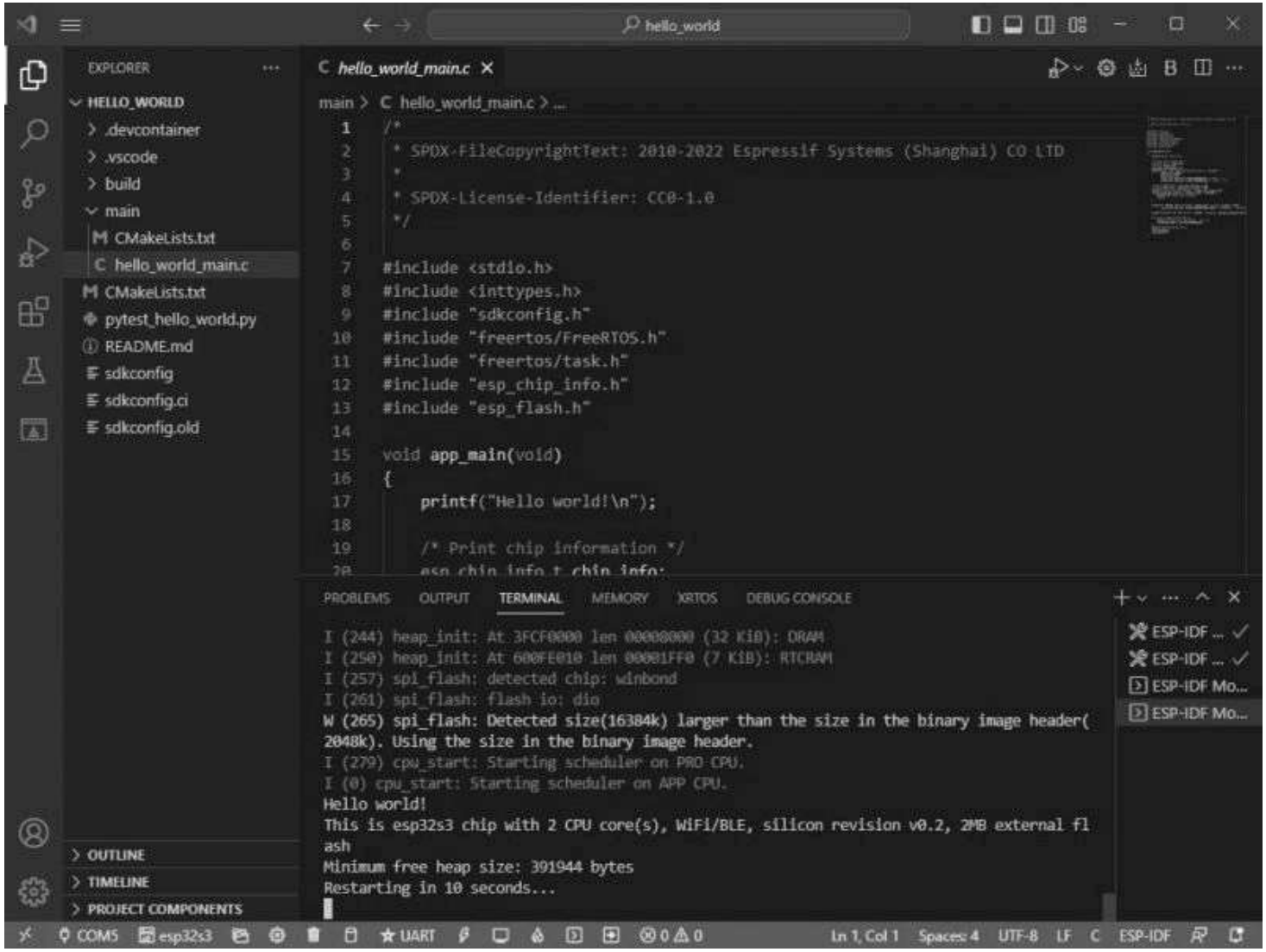
- During this process, ESP-IDF may take up a lot of CPU resources and therefore may cause system lag.

3. Because we use CH343 as a USB to serial port chip, and the on-board automatic download circuit, it can be downloaded automatically without manual operation.



(/wiki/File:ESP32-S3-Pico_33.jpg)

4. After successful download, it will automatically enter the serial monitor, and you can see the corresponding information output from the chip and prompt to reboot after 10s.



(/wiki/File:ESP32-S3-Pico_34.jpg)

Arduino

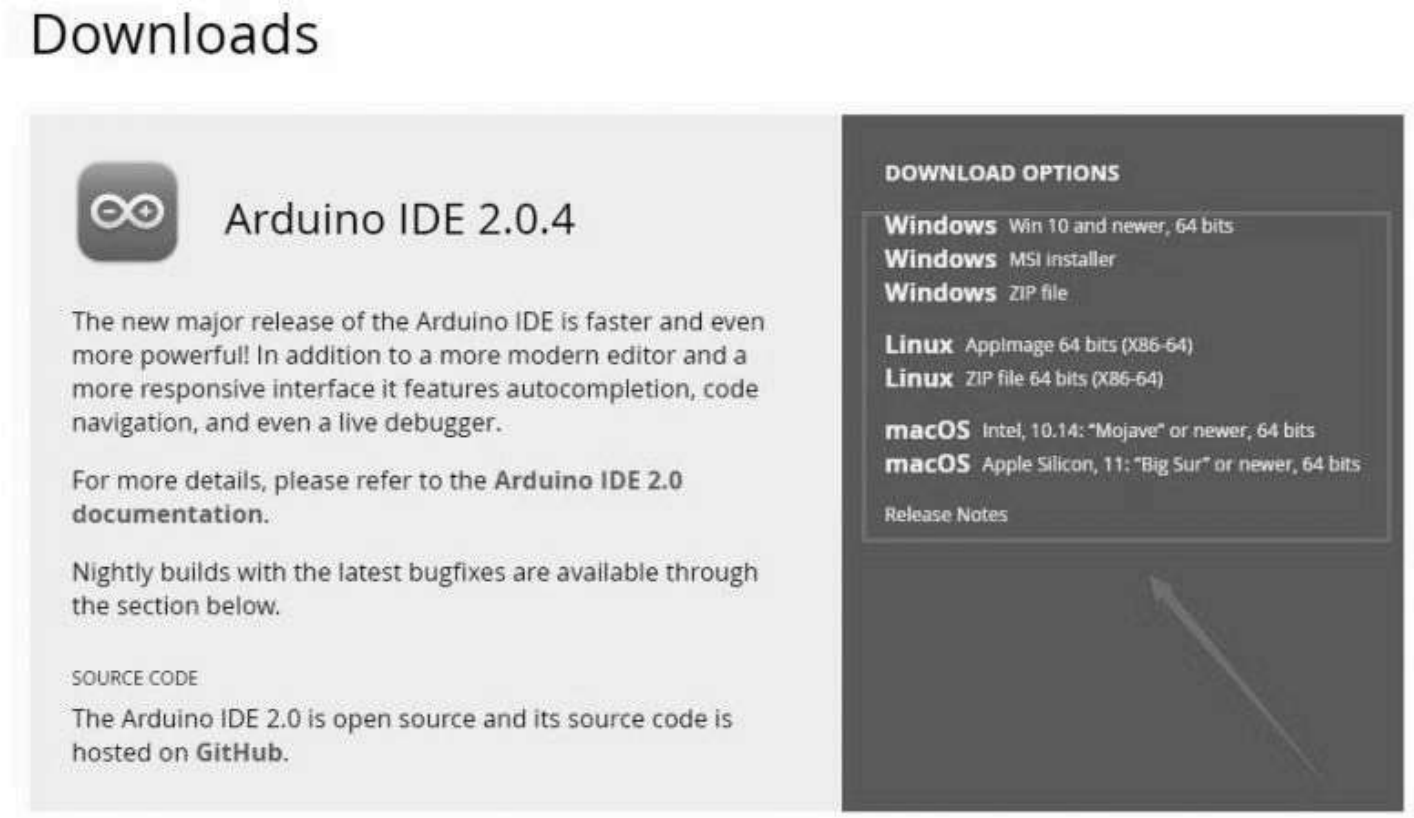
[Collapse]

- **If you do not use arduino-esp32 before, you can refer to this link (https://docs.espressi**

f.com/projects/arduino-esp32/en/latest/index.html).

Install Arduino IDE

1. Open the official software download webpage (https://www.arduino.cc/en/software), and choose the corresponding system and system bits to download.



(/wiki/File:ESP32-S3-Pico_35.jpg)

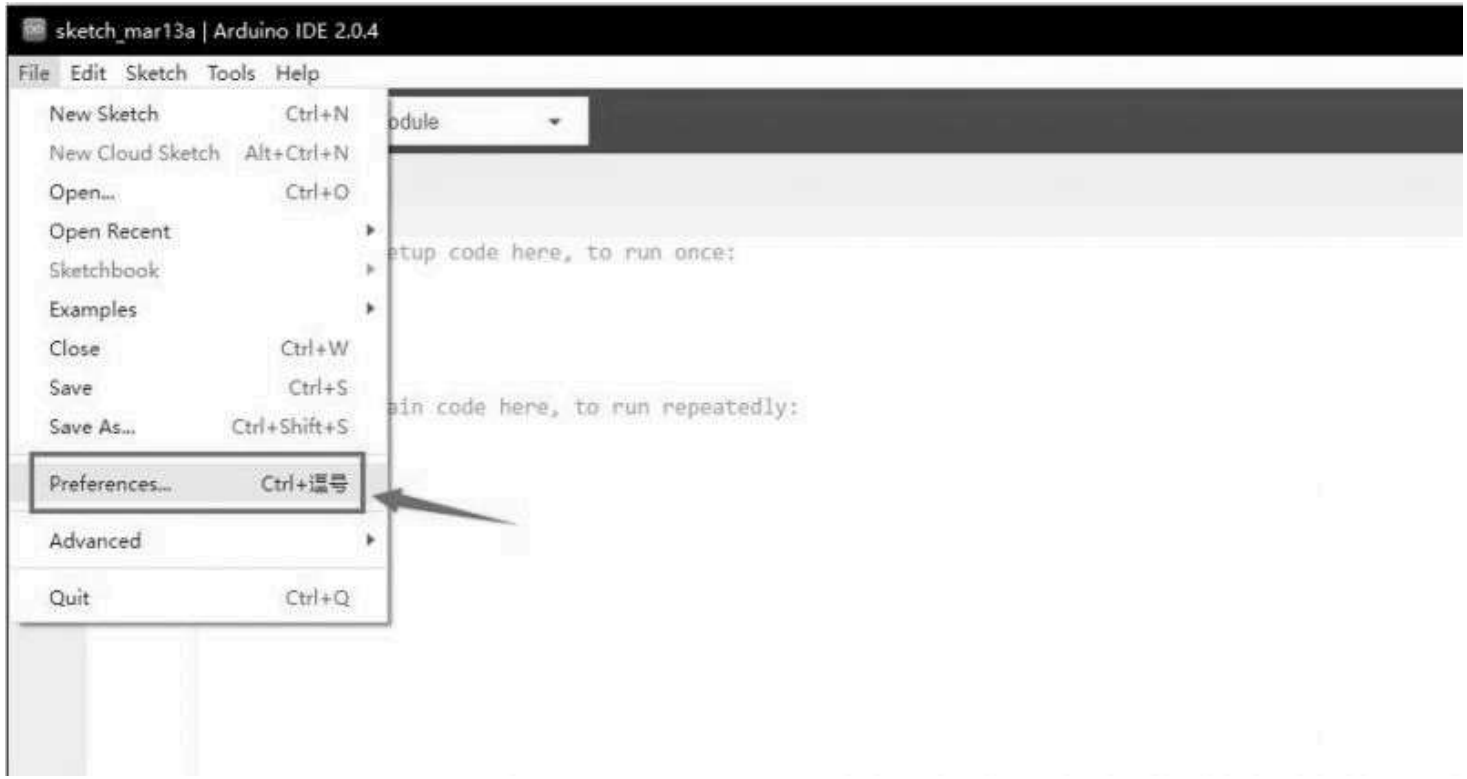
2. You can choose "Just Download", or "Contribute & Download".

(/wiki/File:ESP32-S3-Pico_36.jpg)

3. Run to install the program and install it all by default.

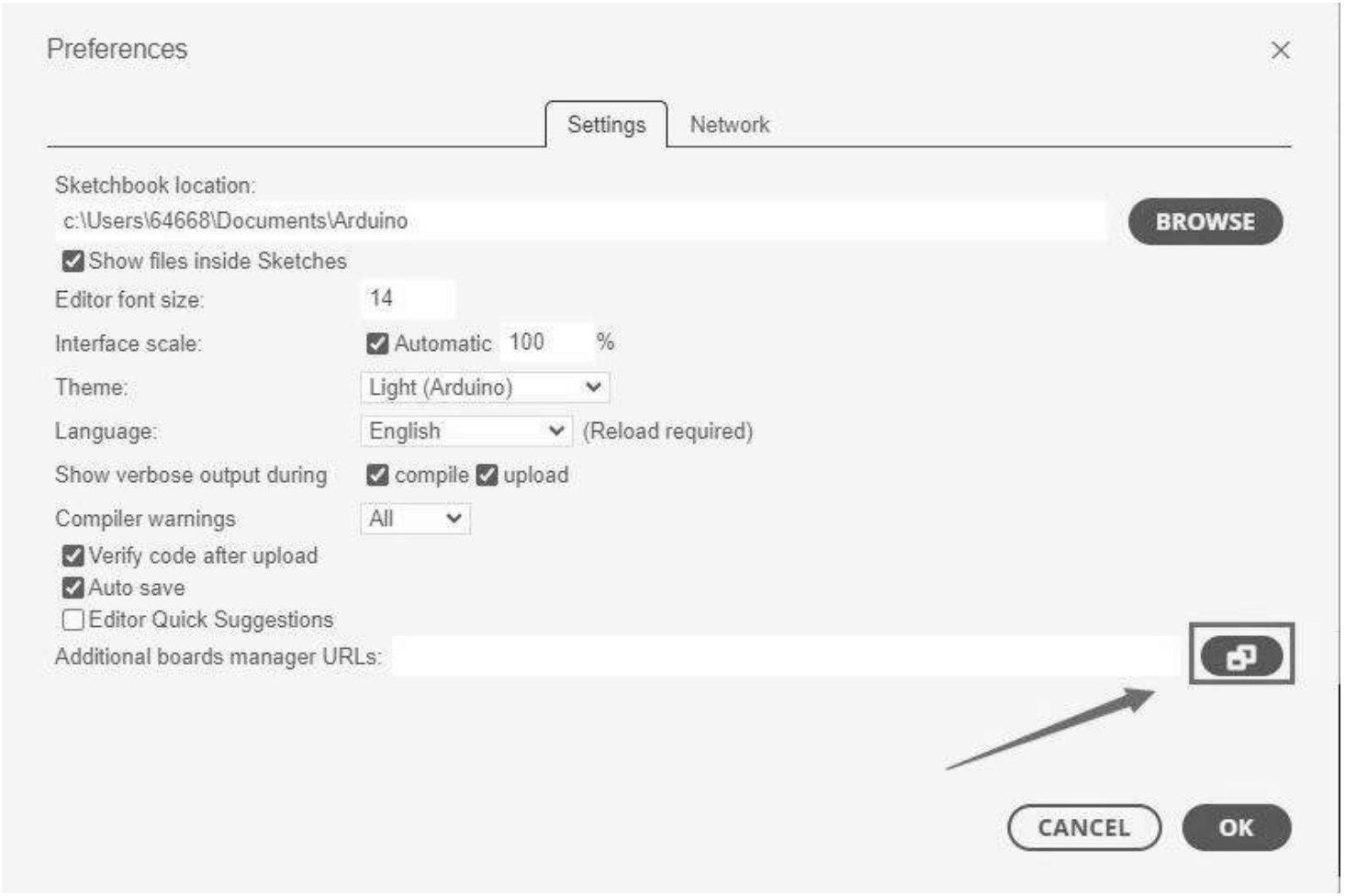
Install arduino-esp32 Online

1. Open Preferences.



(/wiki/File:ESP32-S3-Pico_37.jpg)

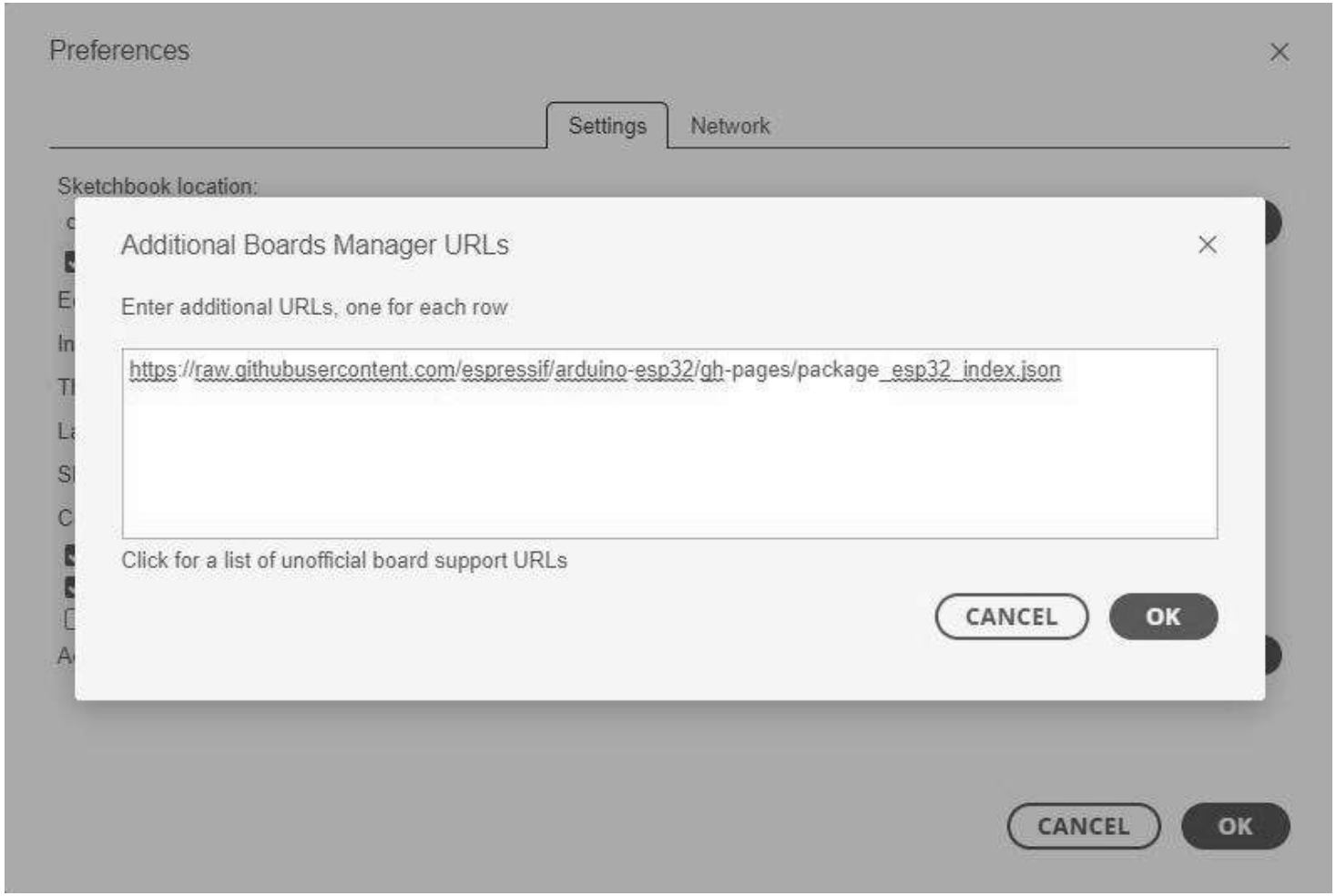
2. Add the corresponding board manager URLs and click the button.



(/wiki/File:ESP32-S3-Pico_38.jpg)

3. Add the following content in the first blank.

```
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
(https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.jso
n)
```

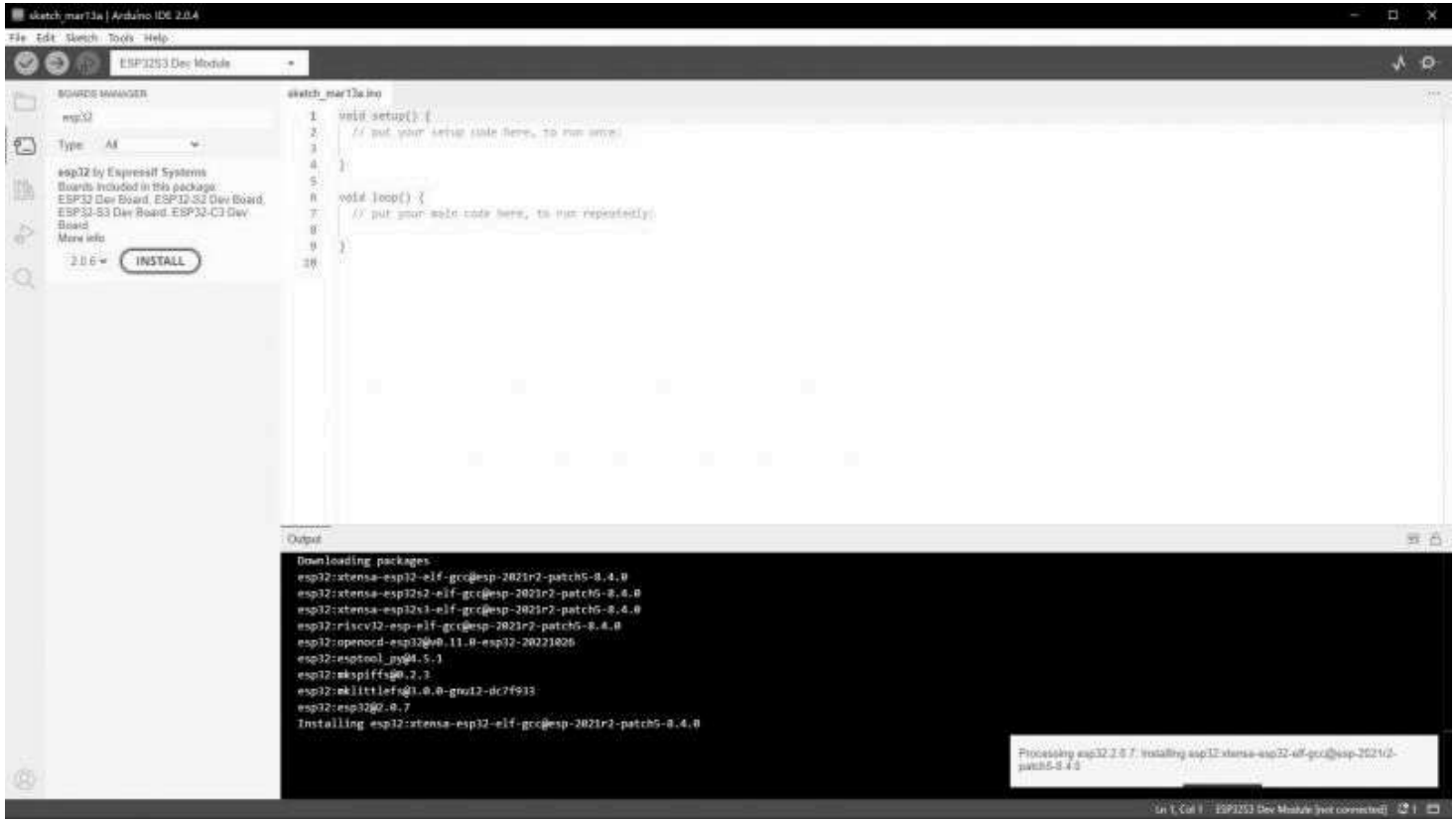


(/wiki/File:ESP32-S3-Pico_39.jpg)

4. Save the setting.
5. Open the board manager and enter ESP32 in the search bar.

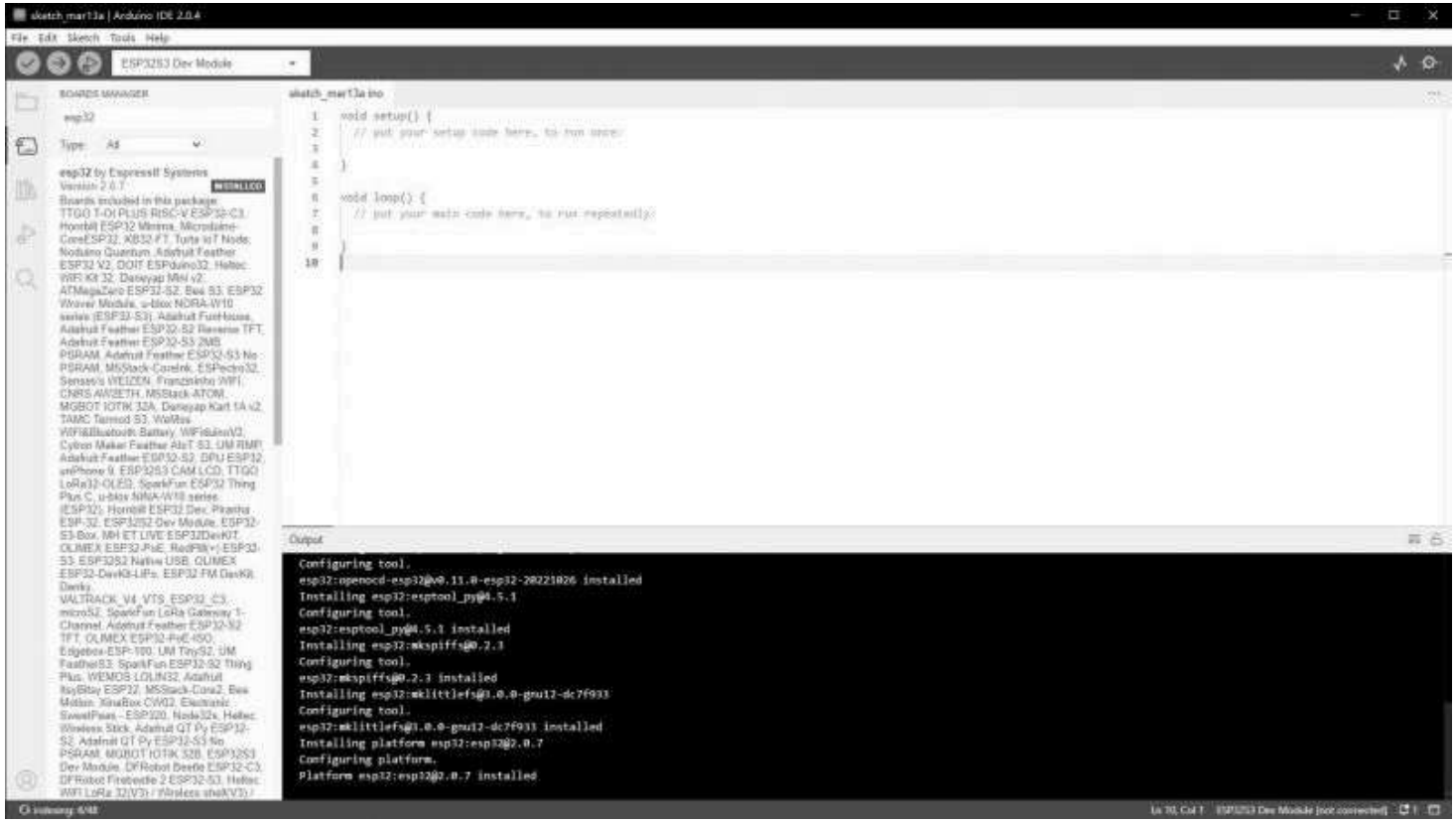
(/wiki/File:ESP32-S3-Pico_40.jpg)

6. Wait for downloading.



(/wiki/File:ESP32-S3-Pico_41.jpg)

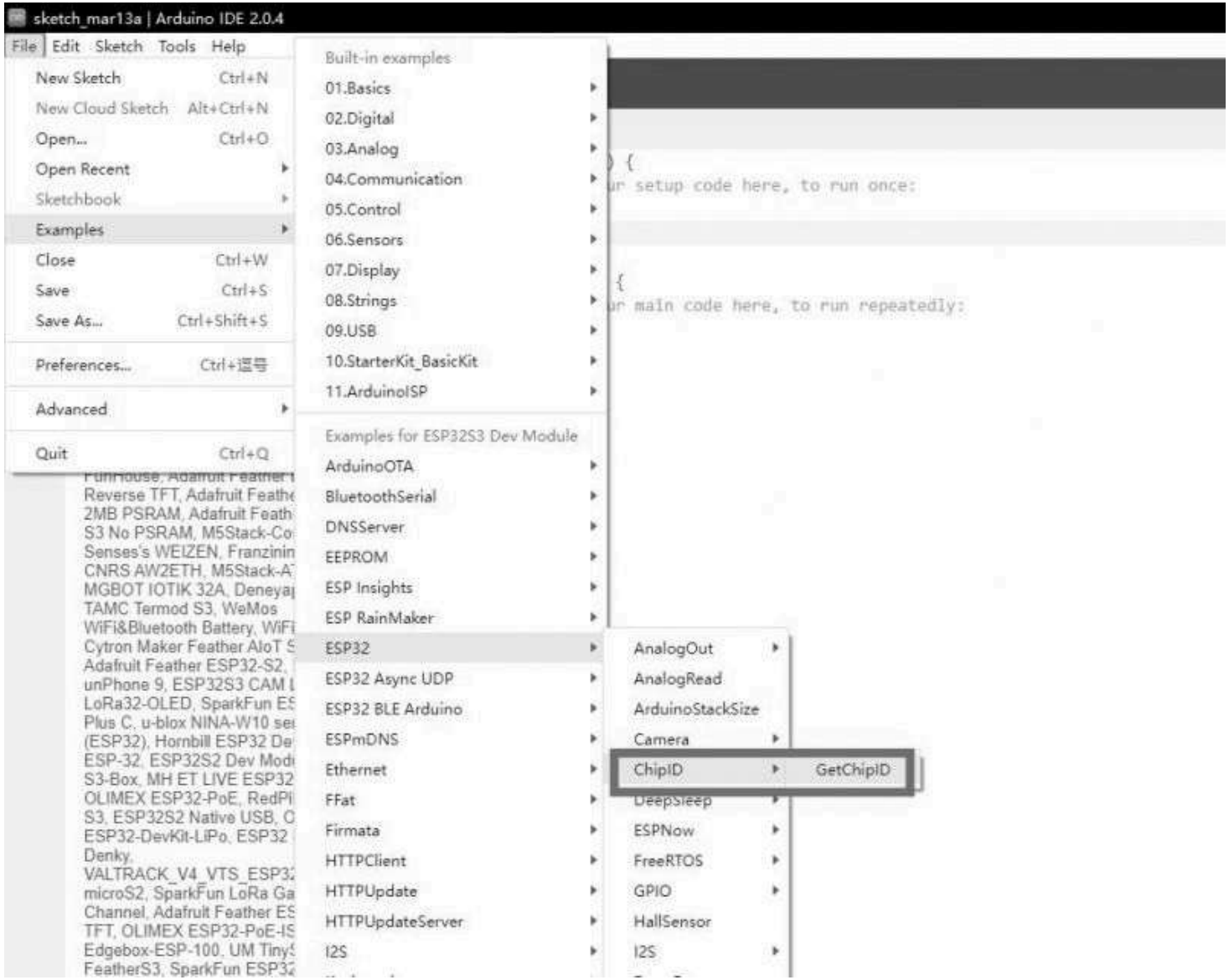
7. arduino-esp32 is downloaded.



(/wiki/File:ESP32-S3-Pico_42.jpg)

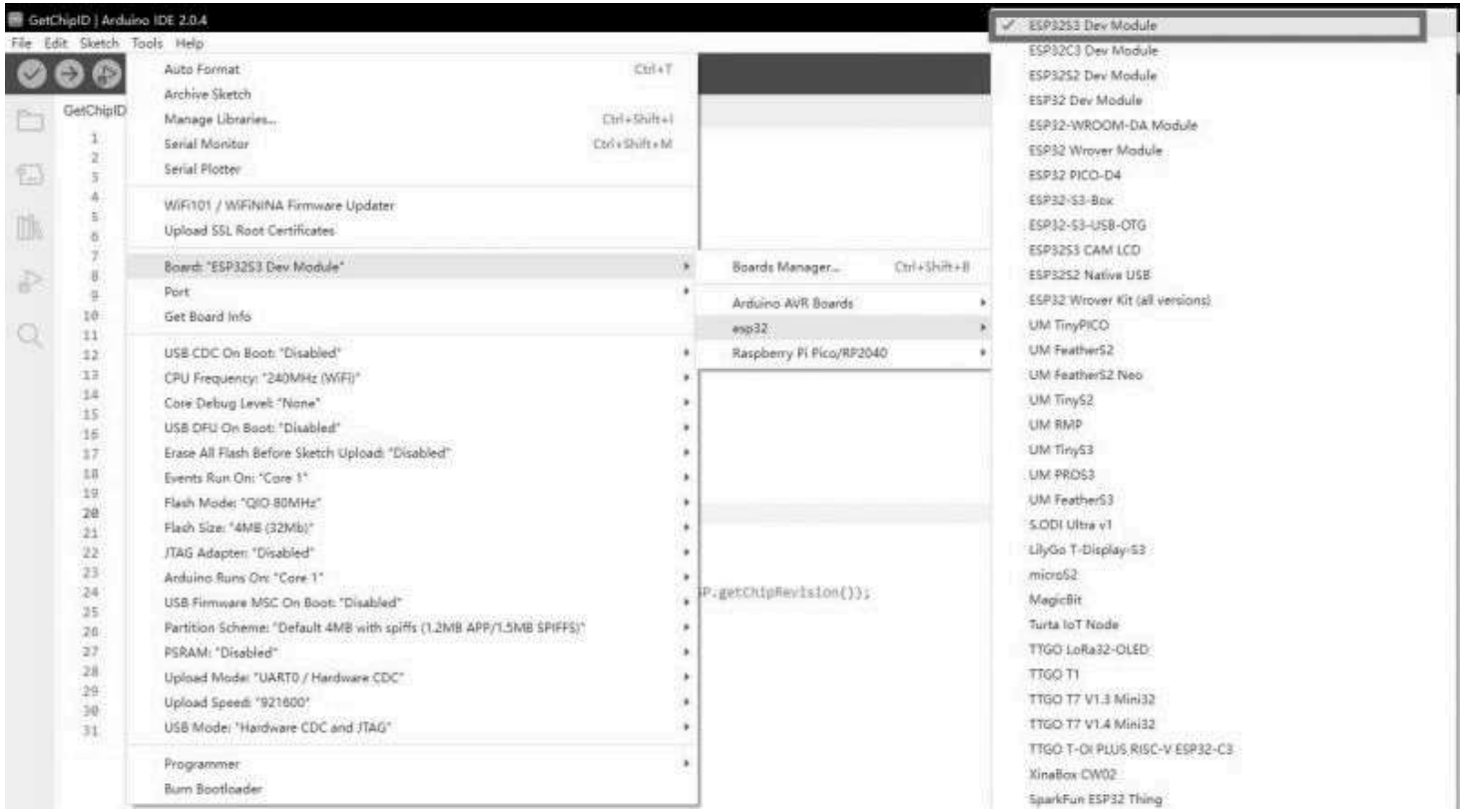
Use Arduino Demo

1. Select the demo, here we choose the demo to get the chip ID.



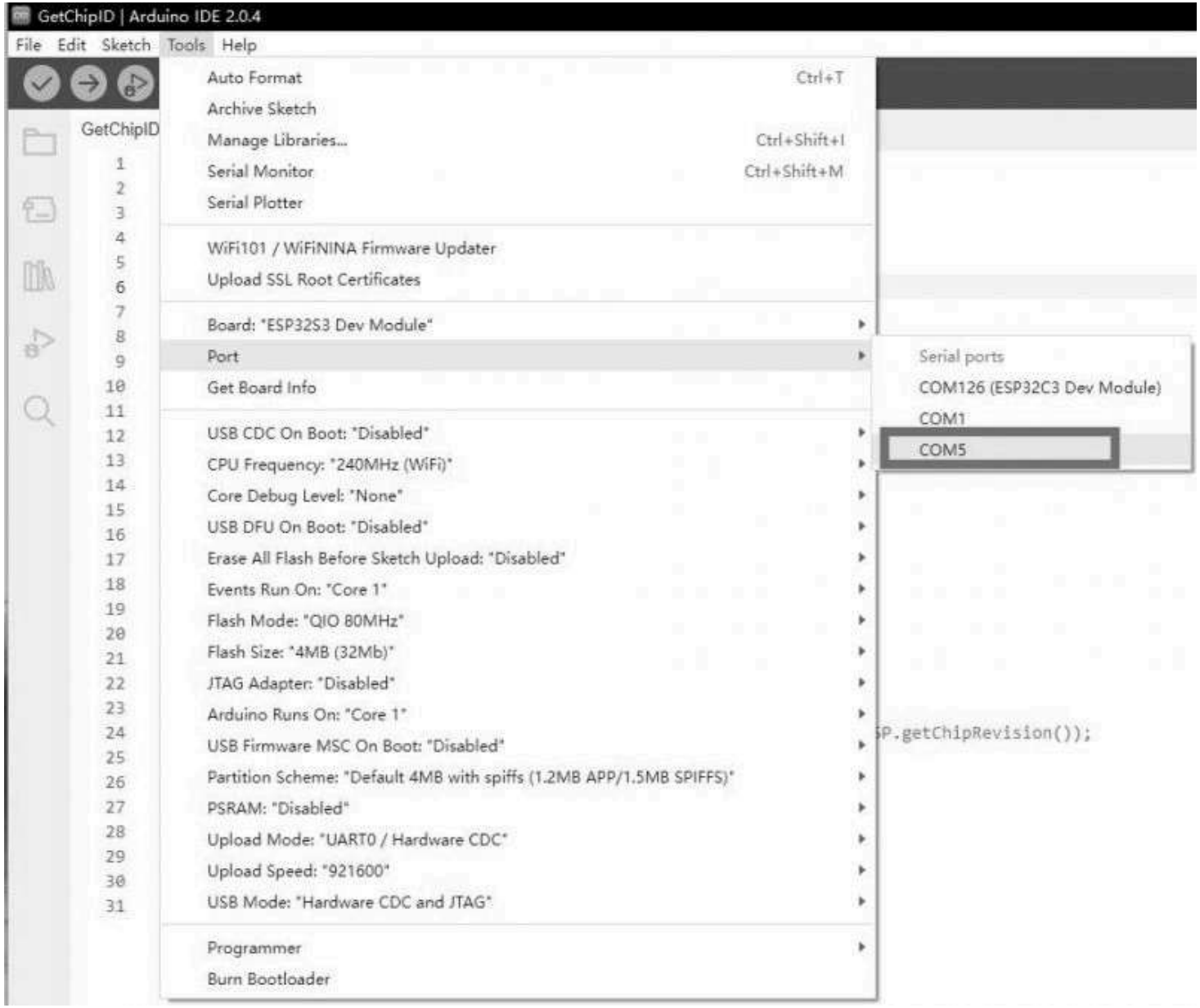
(/wiki/File:ESP32-S3-Pico_43.jpg)

2. Select the board as ESP32S3 Dev Module.



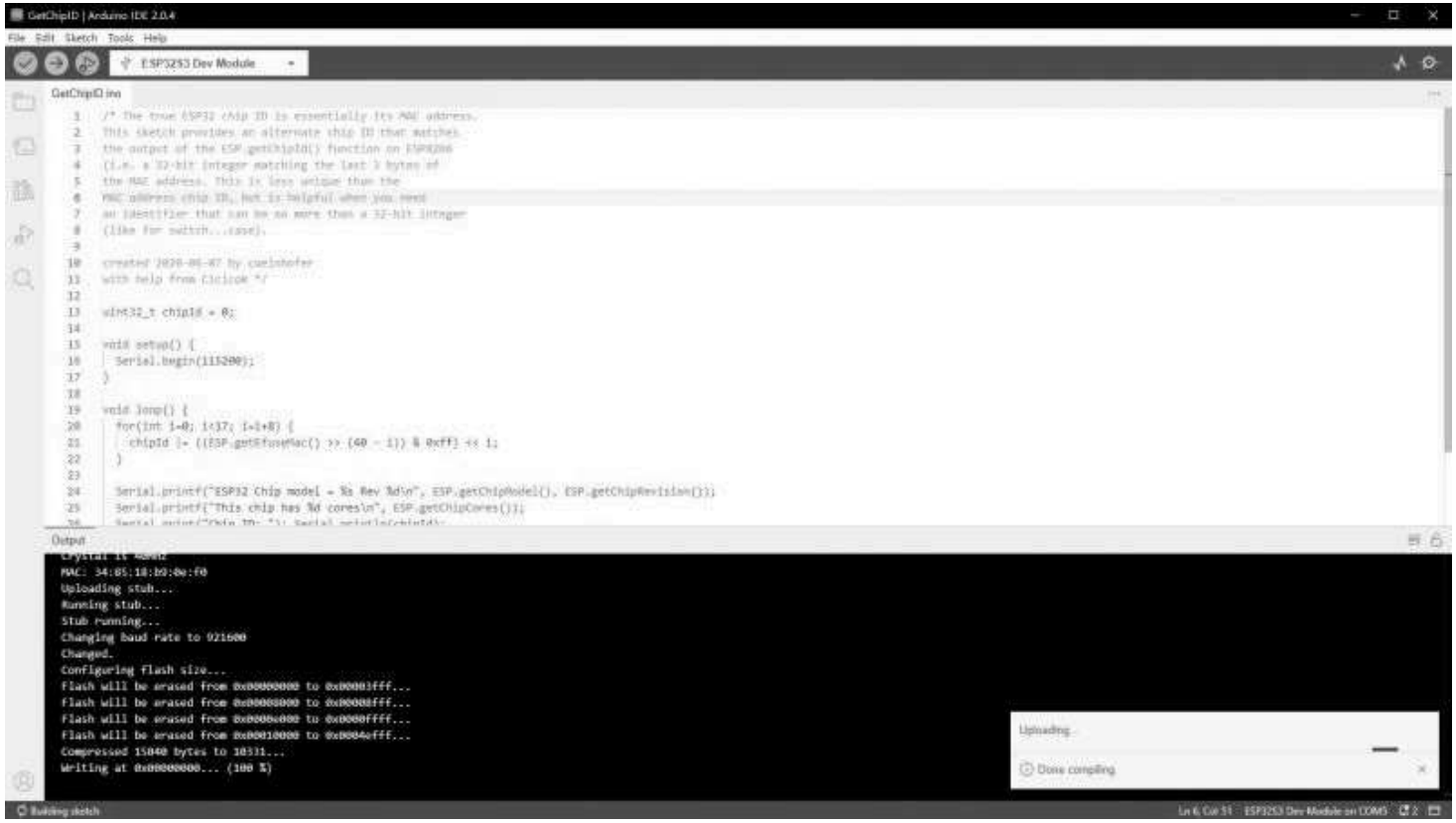
(/wiki/File:ESP32-S3-Pico_44.jpg)

3. Choose the COM5 port of ESP32-S3 USB.



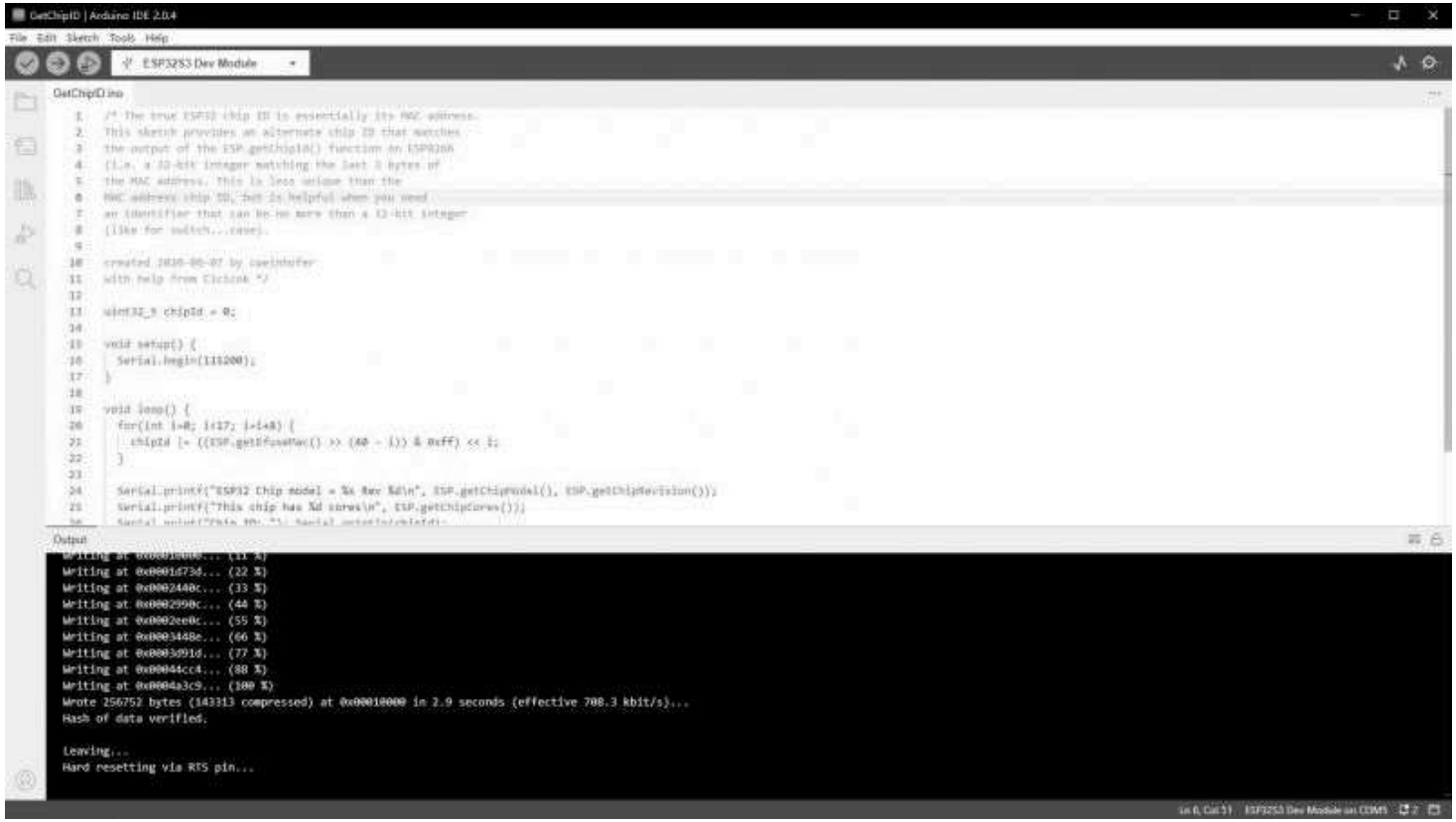
(/wiki/File:ESP32-S3-Pico_45.jpg)

4. Click the download button, then it compiles and downloads automatically.



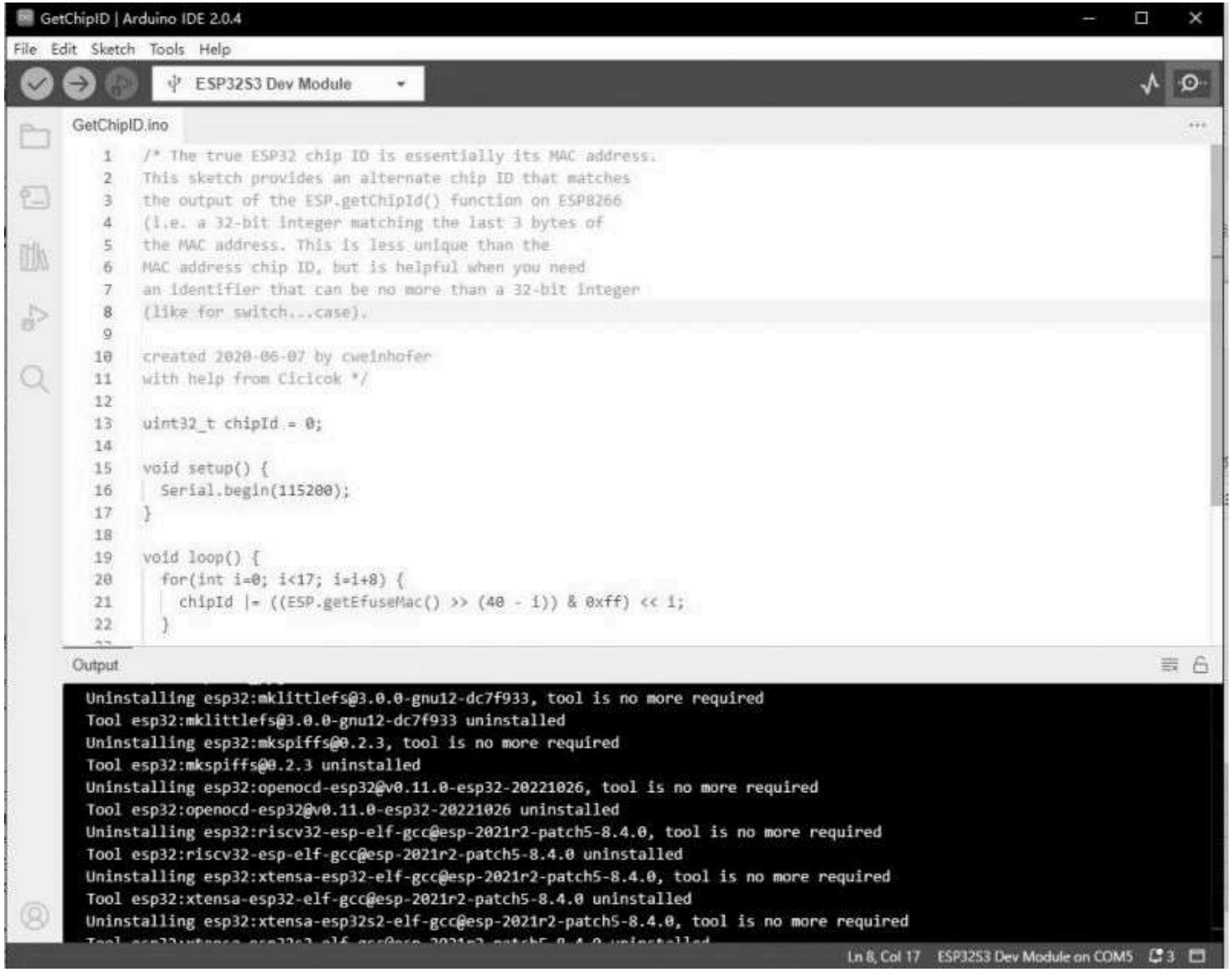
(/wiki/File:ESP32-S3-Pico_47.jpg)

5. Finish.



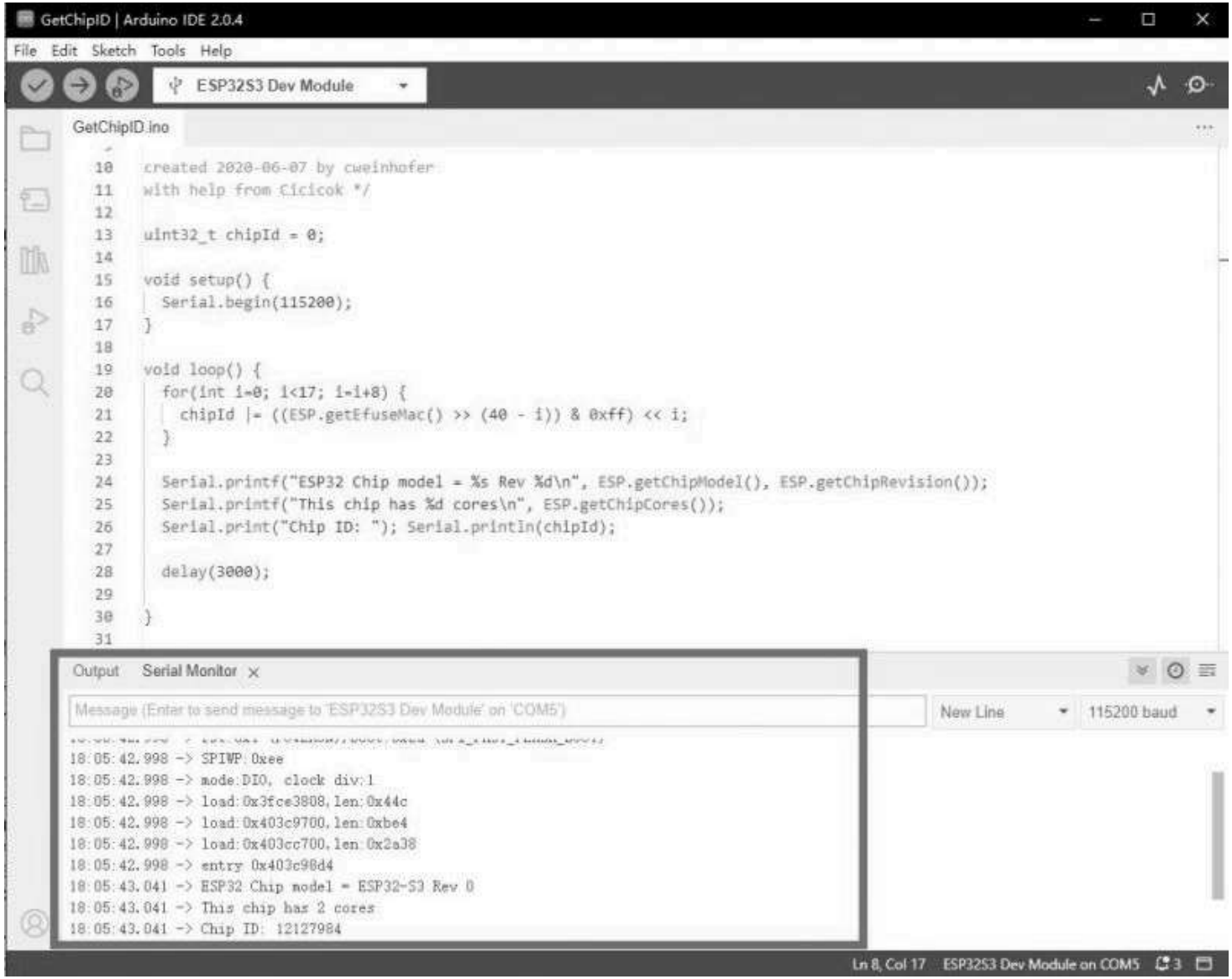
(/wiki/File:ESP32-S3-Pico_48.jpg)

6. Open the Serial Port Monitor.



(/wiki/File:ESP32-S3-Pico_49.jpg)

7. See the chip ID of the loop output.



(/wiki/File:ESP32-S3-Pico_50.jpg)

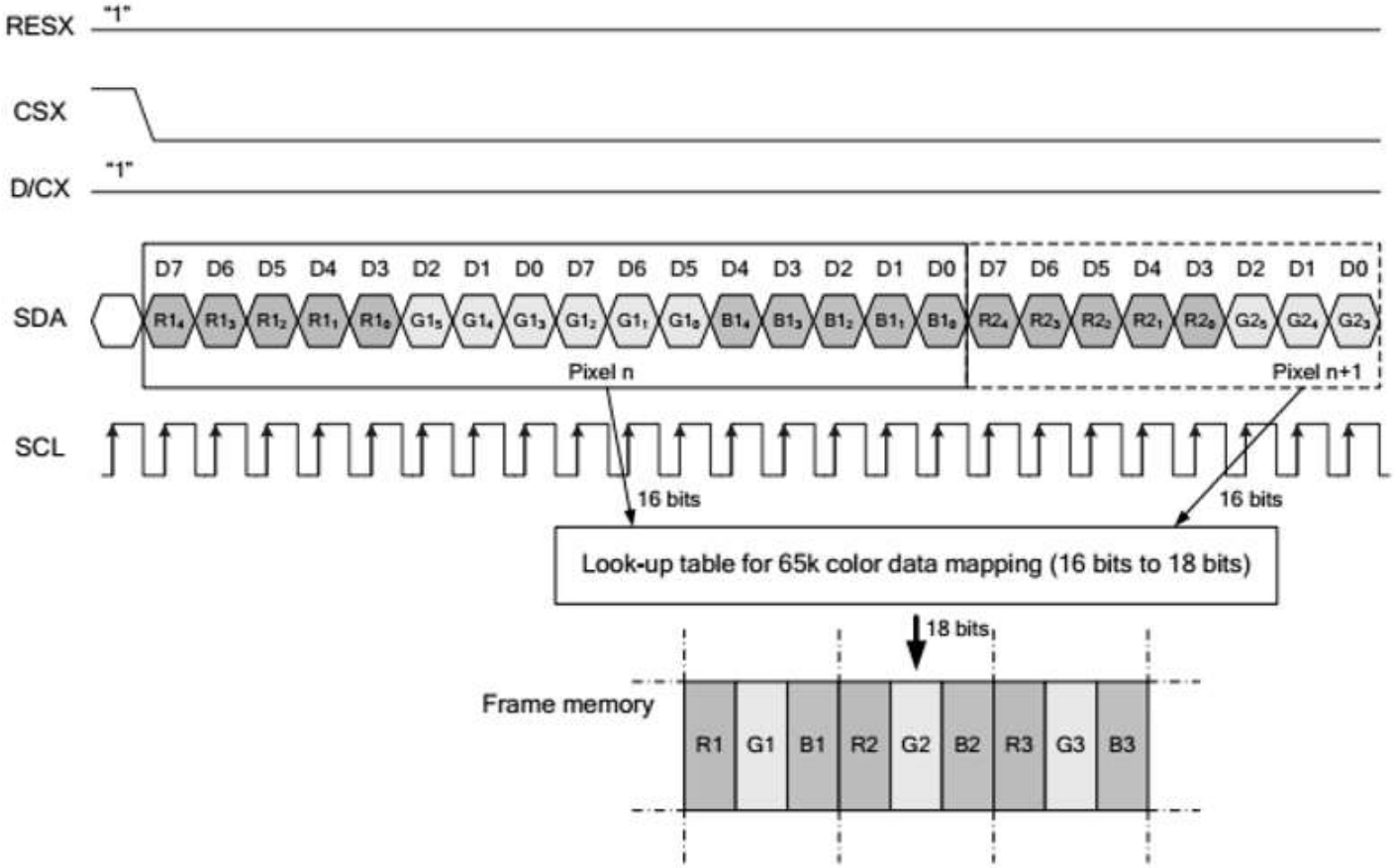
LCD Description

LCD and its Controller

- The built-in controller used in this LCD is ST7789V2, which is an LCD controller with 240 × RGB × 320 pixels, while the pixels of this LCD are 240(H) × RGB × 280(V). Hence, the LCD's internal RAM is not fully used.
- This LCD supports 12-bit, 16-bit, and 18-bit per pixel input color formats, namely RGB444, RGB565, and RGB666 color formats, this demo uses RGB565 color format, which is also commonly used RGB Format.
- This LCD uses a four-wire SPI communication interface, which can greatly save the GPIO port, and the communication speed will be faster.

- The resolution of this module is 240(H) × RGB × 280(V), but sometimes it may not fully display your images as it is round in corners.

SPI Communication Protocol



(/wiki/File:0.96inch_lcd_module_spi.png)

Note: The difference from the traditional SPI protocol is that the data line sent from the slave to the host is hidden because it only needs to be displayed. Please refer to Datasheet Page 66 for the table.

RESX is reset, it is pulled low when the module is powered on, usually set to 1;

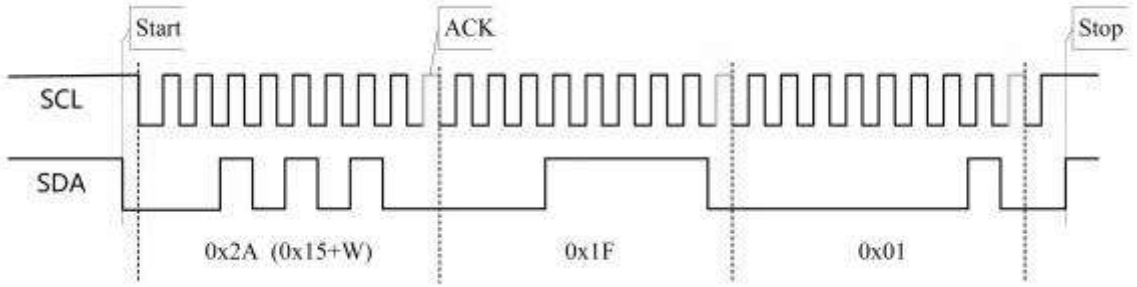
CSX is the slave chip selection, and the chip will be enabled only when CS is low.
D/CX is the data/command control pin of the chip, when DC = 0, write command, when DC = 1, write data.
SDA is the transmitted data, that is, RGB data;
SCL is the SPI communication clock.
For SPI communication, data is transmitted with timing, that is, the combination of clock phase (CPHA) and clock polarity (CPOL):
The level of CPHA determines whether the data is collected on the first clock transition edge or the second clock transition edge of the serial synchronization clock. When CPHA = 0, data acquisition is performed on the first transition edge;
The level of CPOL determines the idle state level of the serial synchronous clock. CPOL = 0, which is a low level.
As can be seen from the figure, at the first falling edge of SCLK it starts to transmit data, 8-bit data is transmitted in one clock cycle, using SPI0, bit-by-bit transmission, high bit first, and low bit last.

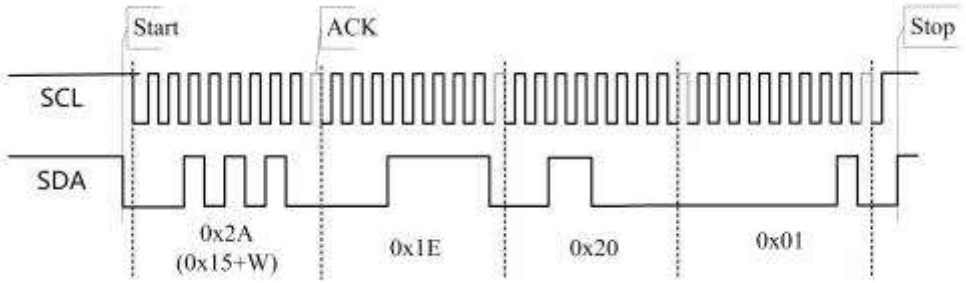
Touch and its Controller

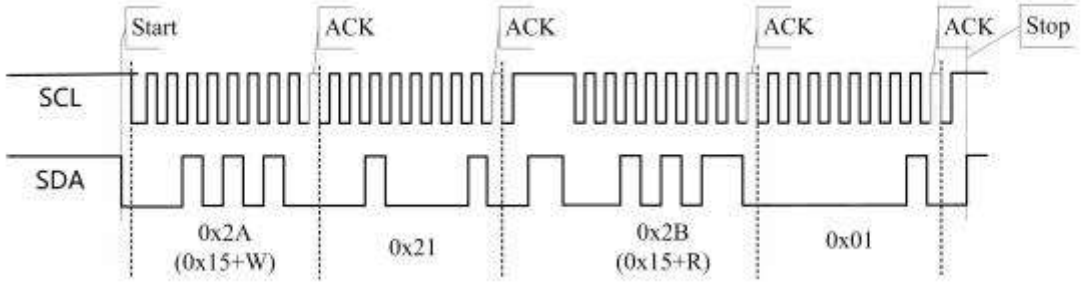
- This LCD is composed of surface toughened glass + thin film FILM material, which has high strength, strong hardness, and good light transmittance. The matching driver chip is CST816D self-capacitance touch chip, which supports the standard I2C communication protocol standard, which can realize a 10Khz~400Khz configurable communication rate.

I2C Communication Protocol

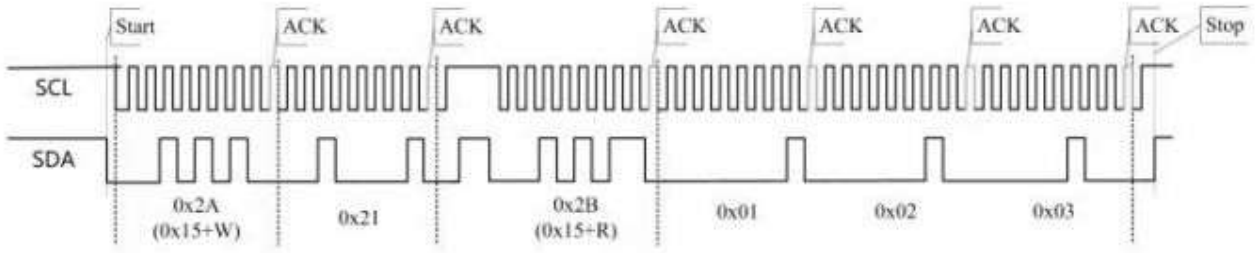
- The 7-bit device address of the chip is 0x15, that is, the device writing address is 0x2A, and reading address is 0x2B.

- Waveform introduction:
  - Write a single byte: (write 0x01 to 0x1F register)
  - 

    (/wiki/File:1.28inch_Touch_LCD02.jpg)
  - Write multiple bytes consecutively (write 0x20, 0x01 to 0x1E and 0x1F respectively)
  - 

    (/wiki/File:1.28inch_Touch_LCD03.jpg)
  - Read a single byte (read a single byte from 0x21)
  - 

    (/wiki/File:1.28inch_Touch_LCD04.jpg)

- Read multiple bytes consecutively (read 3 bytes from 0x21, 0x22, 0x23)
- 

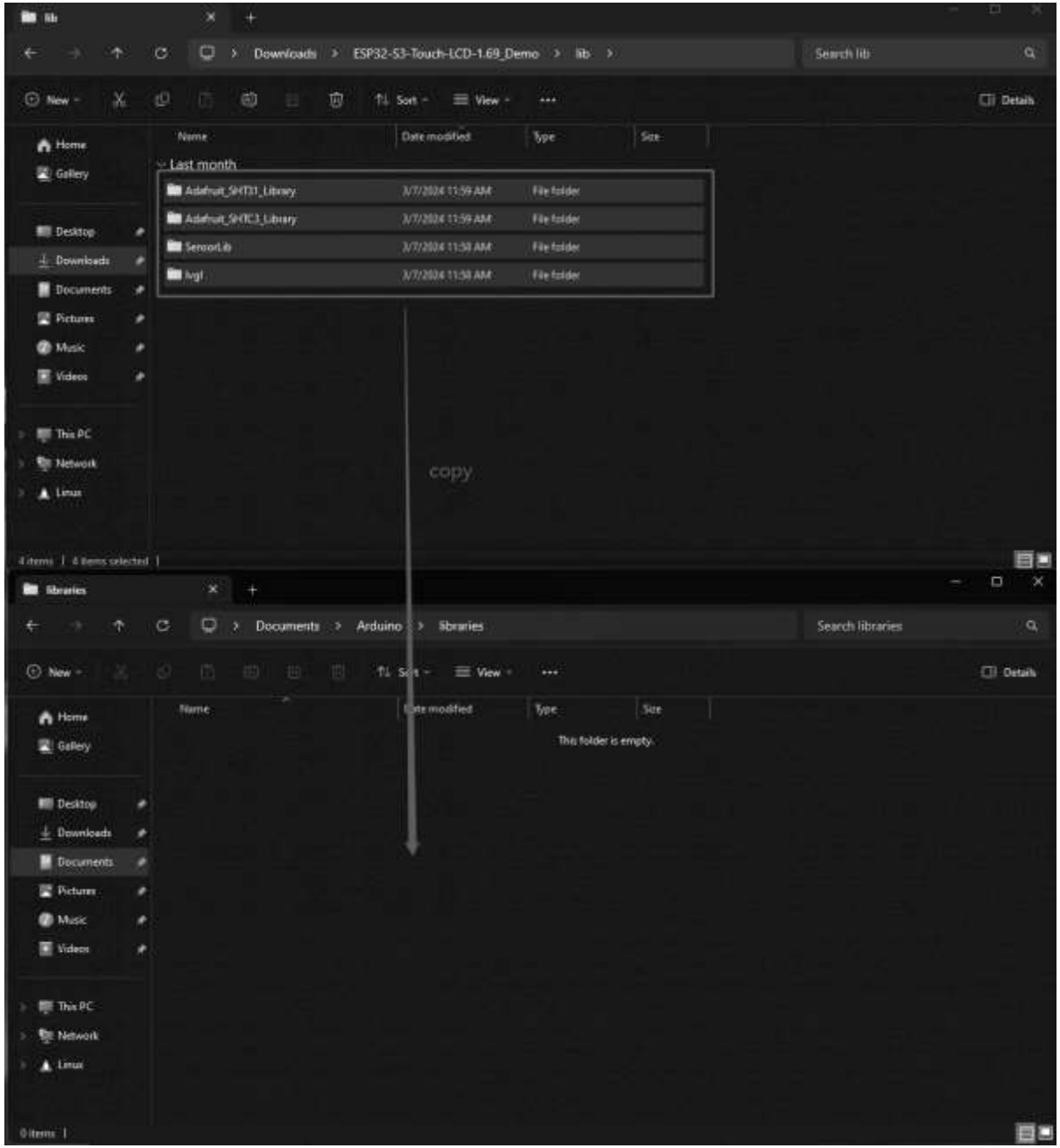
(/wiki/File:1.28inch_Touch_LCD05.jpg)

Sample Demo

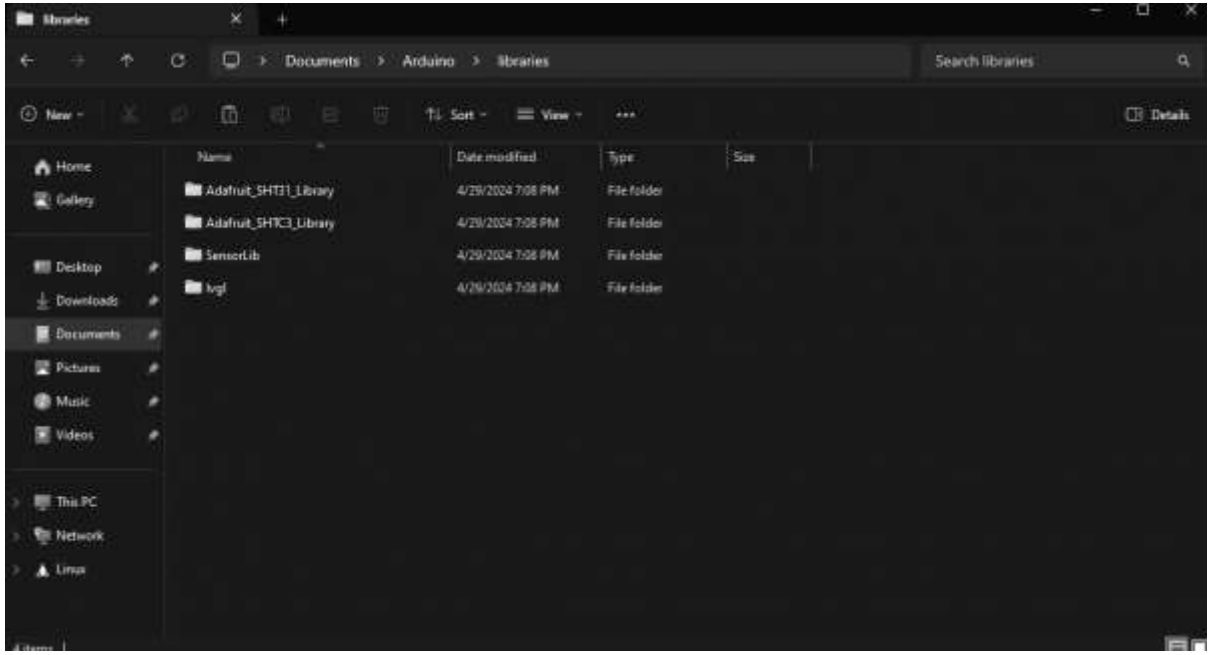
Library Installation

It is recommended to directly use the "lib" library in the sample demo when using the LVGL library and modifying the corresponding screen configuration.

Copy the library file folder to "C:\Users\xxxx\Documents\Arduino\libraries ", "xxxx" is your PC username.



(/wiki/File:ESP32-S3-Touch-LCD-1.69_240429_01.png)



(/wiki/File:ESP32-S3-Touch-LCD-1.69_240429_02.png)

Arduino_LVGL

LVGL (Light and Versatile Graphics Library) is a free and open-source graphic library for embedded systems and is very suitable for devices with limited resources such as ESP32-S3. In the sample demo, you can use "LVGL_Arduino" to check the screen effect.

(/wiki/File:ESP32-S3-Touch-LCD-1.69-240423_03.png)

User-defined PWR_Key

This key is designed to address the limited functionality of external buttons. The working principle is as follows:

Pressing PWR can power the battery, and then the system boots. The system should define GPIO35 to continuously output a high level to maintain power-on effect. At this point, releasing PWR will not power off. The function of PWR at this time is to pull down GPIO36. The system detects pressing, double pressing, and long pressing of GPIO36, and can customize shutdown control operations. For example, in long press mode, the system sets GPIO35 to a low level to disconnect the battery power, thereby enabling the multi-function button to be used.


Keys_Bee | Arduino IDE 2.3.2
File Edit Sketch Tools Help
ESP32S3 Dev Module
Keys_Bee.ino
62 }
63 }
64 }
65
66 // Check if the button is in long press state
67 if (buttonState == LOW && (millis() - lastDebounceTime >= longPressDuration)) {
68 // Long press button
69 printf("Long Press\n");
70 tone(beePin, 4000);
71 longPressDetected = true; // Set long press detection flag
72 clickDetected = false; // Reset single click detection flag
73 doubleClickDetected = false; // Reset double click detection flag
74 }
75 lastButtonState = reading;
76 }
77
Output Serial Monitor
Message (Enter to send message to 'ESP32S3 Dev Module' on 'COM26')
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press
Long Press Released
Single Click
Double Click
Double Click
Double Click
Double Click
Double Click
Double Click
Double Click

(/wiki/File:ESP32-S3-Touch-LCD-1.69-240423_04.png)

PCF85063 RTC Clock Chip

The PCF85063 RTC Clock facilitates the use of scheduled tasks, provides accurate time tracking, and low-power wake-up functions.
Through example demos, the RTC function can be easily utilized.

```
37
38    rtc.setDateTime(year, month, day, hour, minute, second);
39  }
40
41
42  void loop() {
43    if (millis() - lastMillis > 1000) {
44      lastMillis = millis();
45      RTC_DateTime datetime = rtc.getDateTime();
46      printf(" Year : %d", datetime.year);  // %d for integers
47      printf(" Month: %d", datetime.month);
48      printf(" Day  : %d", datetime.day);
49      printf(" Hour : %d", datetime.hour);
50      printf(" Minute: %d", datetime.minute);
51      printf(" Sec  : %d", datetime.second);
52      printf("\n");  // Prints a new line
53    }
54  }
```

Output Serial Monitor ×

Message (Enter to send message to 'ESP32S3 Dev Module' on 'COM40')

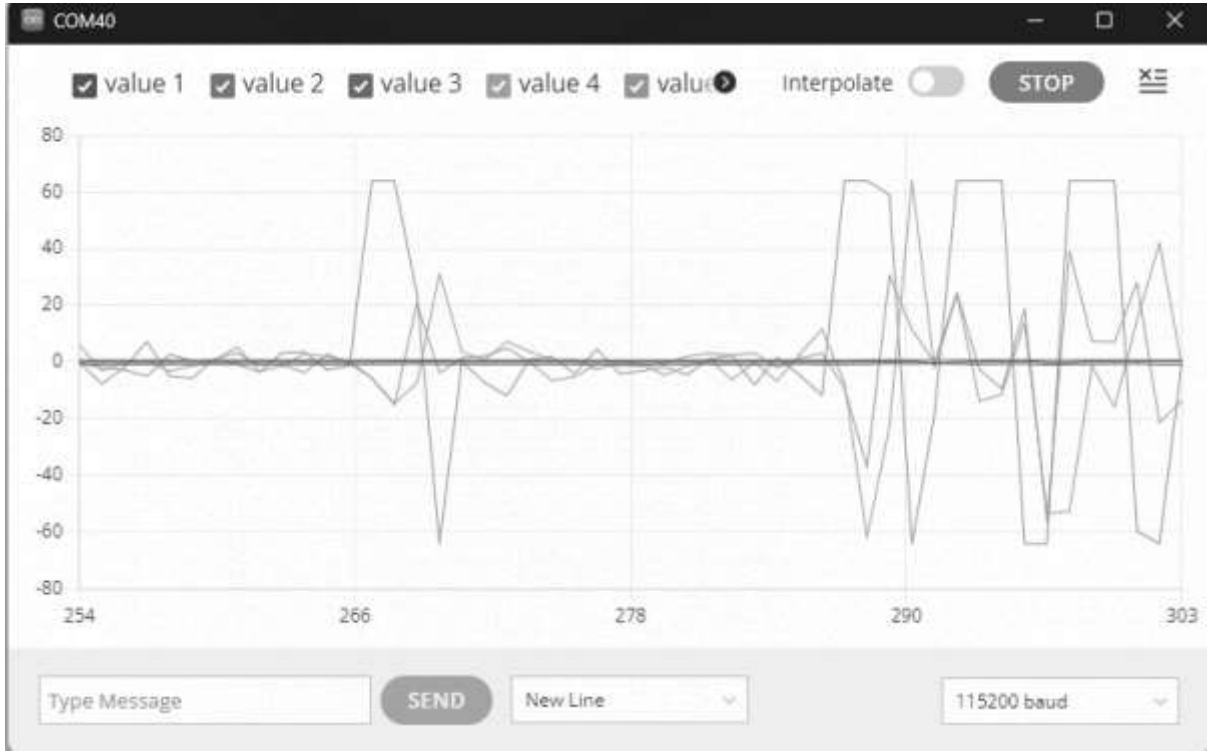
```
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 52
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 53
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 54
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 55
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 56
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 57
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 58
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 9 Sec  : 59
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 0
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 1
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 2
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 3
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 4
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 5
```

```
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 6
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 7
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 8
Year : 2024 Month: 4 Day  : 23 Hour : 11 Minute: 10 Sec  : 9
```

(/wiki/File:ESP32-S3-Touch-LCD-1.69-240423_02.png)

QMI8658 6-axis IMU

The QMI8658 can be used for applications such as posture detection, gait analysis, and fall detection. When embedded in this development board, it can be applied to smart wearables. Example demos can be used to verify changes in multi-axis acceleration, and the data can be observed directly through a visualization tool.



(/wiki/File:ESP32-S3-Touch-LCD-1.69-240423_01.png)

Buzzer

By combining RTC, it's possible to implement timed tasks, scheduled alarms, and other applications.

Resource

Document

- Schematic (https://files.waveshare.com/wiki/ESP32-S3-Touch-LCD-1.69/ESP32-S3-Touch-LCD-1.69-Sch.pdf)

- MicroPython Development Doc (https://docs.micropython.org/en/latest/)
- ESP32 Arduino Core's documentation (https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html)
- arduino-esp32 (https://github.com/espressif/arduino-esp32)
- ESP-IDF (https://github.com/espressif/esp-idf)

Datasheet

- ST7789V2 datasheet (https://files.waveshare.com/upload/c/c9/ST7789V2.pdf)
- CST816S datasheet (https://www.waveshare.com/w/upload/5/51/CST816S_Datasheet_EN.pdf)
- CST816S register description (https://www.waveshare.com/w/upload/c/c2/CST816S_register_declaration.pdf)

Demo

- Demo (https://files.waveshare.com/wiki/ESP32-S3-Touch-LCD-1.69/ESP32-S3-Touch-LCD-1.69_Demo.zip)

Software

- Zimo221 (https://www.waveshare.com/w/upload/c/c6/Zimo221.7z)
- Image2Lcd2.9 (https://www.waveshare.com/w/upload/b/bd/Image2Lcd2.9.zip)
- Image Extraction (https://www.waveshare.com/wiki/Image_Extraction)

- Ink Screen Font Library Tutorial (https://www.waveshare.com/wiki/Ink_Screen_Font_Library_Tutorial)

FAQ

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8
(Monday to Friday)

Submit Now (https://service.waveshare.com/)