# ESP32-S3-SIM7670G-4G

From Waveshare Wiki
Jump to: navigation, search

## Overview

**ESP32-S3-SIM7670G-4G**

(https://www.waveshare.com/esp32-s3-sim7670g-4g.htm)

ESP32-S3R2
Type C UART GPIO

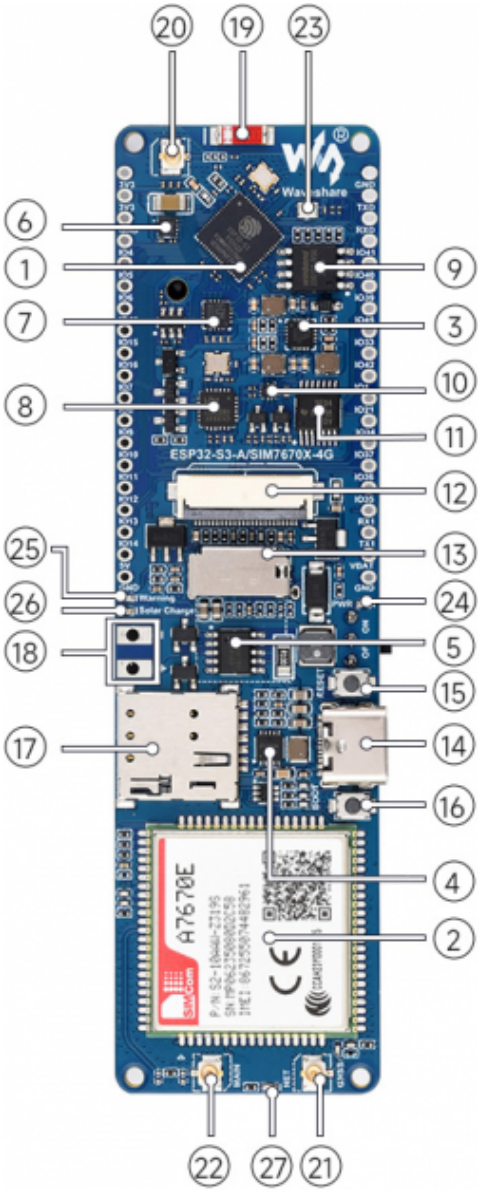
The ESP32-S3-SIM7670G-4G (hereinafter referred to as the development board) is a multifunctional, high-performance microcontroller development board designed by Waveshare. It integrates an SIM7670G 4G communication module, a universal OV camera interface, a TF card slot, RGB colorful LED, an 18650 battery holder (18650 battery is NOT included), a battery voltage measurement IC, a solar panel charging interface, and other peripherals. It employs the ESP32-S3R2, a System On Chip (SoC) that integrates low-power Wi-Fi and BLE5.0. Additionally, it comes with an external 16MB Flash and 2MB PSRAM. The SoC incorporates a hardware encryption accelerator, random number generator (RNG), HMAC, and digital signature modules, meeting the security requirements of the Internet of Things (IoT).
The A7670E 4G communication module provides mobile network capabilities, enabling functionalities like portable Wi-Fi and IoT data transmission when combined with the ESP32-S3R2. Its various low-power operating modes cater to power consumption demands in IoT, mobile devices, outdoor monitoring, smart home applications, and other scenarios.

### Features

- Comes with a high-performance Xtensa® 32-bit LX7 dual-core processor with up to 240MHz.
- Support 2.4GHz WiFi (802.11 b/g/n) and Bluetooth® 5 (LE) with onboard antenna.
- Built-in 512KB of SRAM and 384KB ROM, with onboard 2MB PSRAM and an external 16MB Flash memory.

### Hardware Description

- Onboard patch antenna, you can use it or short it to connect the external antenna as shown in ⑲ and ⑳.



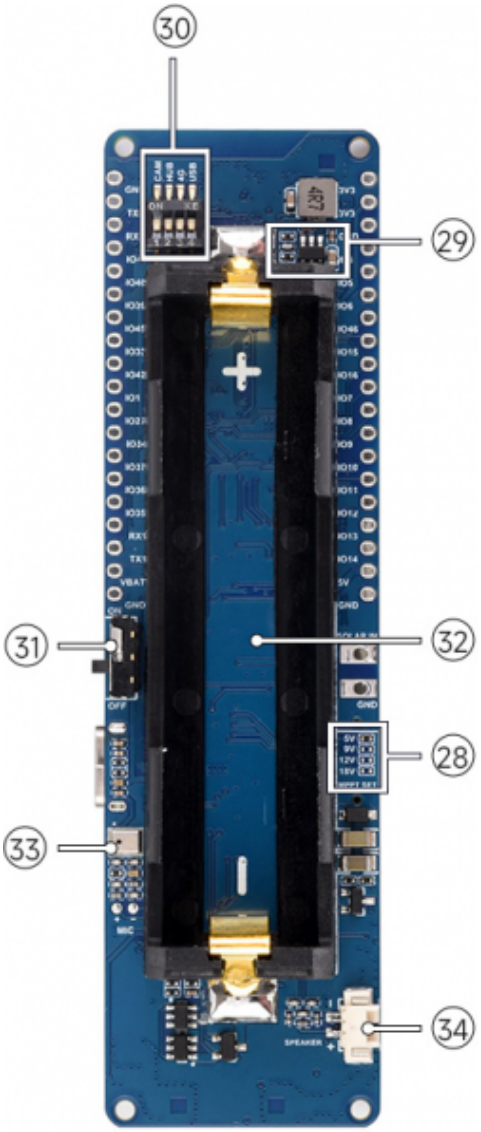
(/wiki/File:ESP32-S3-A7670E-4G-details-intro2.png)

- Onboard RGB colorful beads, WS2812B driver as shown in ㉓.
- Onboard Camera interface: using 24pin camera interface as shown in ⑫.

The supported camera list is shown below:

| model | max resolution | color type | Len Size |
|---|---|---|---|
| OV2640 | 1600 x 1200 | color | 1/4" |
| OV3660 | 2048 x 1536 | color | 1/5" |
| OV5640 | 2592 x 1944 | color | 1/4" |
| OV7670 | 640 x 480 | color | 1/6" |
| OV7725 | 640 x 480 | color | 1/4" |
| NT99141 | 1280 x 720 | color | 1/4" |
| GC032A | 640 x 480 | color | 1/10" |
| GC0308 | 640 x 480 | color | 1/6.5" |
| GC2145 | 1600 x 1200 | color | 1/5" |
| BF3005 | 640 x 480 | color | 1/4" |
| BF20A6 | 640 x 480 | color | 1/10" |
| SC101IOT | 1280 x 720 | color | 1/4.2" |
| SC030IOT | 640 x 480 | color | 1/6.5" |
| SC031GS | 640 x 480 | color | 1/6" |

- Onboard TF-Card slot, support storing files and pictures as shown in ⑬.
- Onboard solar panel charging interface as shown in ⑱.
    - You can select different resistor values on the back to switch the solar input voltage.
    - When the solar panel is charging, the onboard green LED will light up, as shown in figures ㉖ and ㉘.
- The onboard circuit switch allows you to control the circuit on/off when using the 18650 power supply, as shown in figure ㉛.
- Onboard USB to UART chip and automatic download circuit, after connecting the Type-C cable to program the demo and firmware, as shown in ⑦.
- Using ESP32-S3 USB to connect to the A/SIM7670X USB connector, and using TinyUSB protocol to realize ppp dial-up network access as the portable WIFI.
- The onboard 18650 battery interface is designed for a single 3.7V 18650 lithium battery. Pay attention to the polarity markings on the lithium battery interface.
    - When the battery is reversed, the onboard yellow LED will light up as a warning, as shown in ㉕.
- The development board reserves GPIO pins for external device connections, which can be flexibly configured as I2C, SPI, and other peripheral functions. For detailed functions, please refer to the GPIO allocation description.

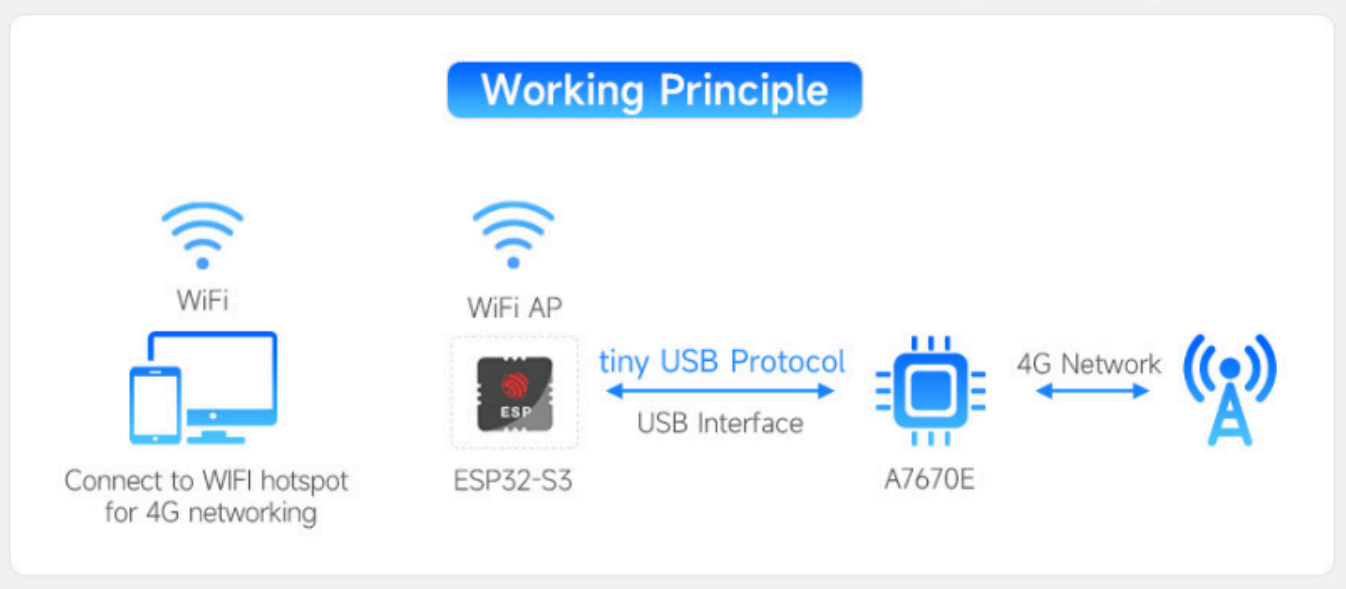


(/wiki/File:ESP32-S3-A7670E-4G-details-intro1.png)

- Onboard a GNSS IPEX1 interface. After powering on, you can enable the GNSS positioning function using the relevant commands, as shown in ㉑.
- Furthermore, there is a DIP switch onboard for convenient control of the Camera, USB HUB circuit, and 4G module power supply. It can also control the USB circuit switching of the 4G module, as depicted in figure ㉚.
- Onboard LED Description:
    - Onboard battery anti-reverse warning LED, the yellow LED is on when the batteries are reversely connected as shown in ㉕.
    - Onboard green solar panel charging LED, the LED is on when the solar panel input voltage is active, as shown in ㉖.
    - Onboard blue power indicator, lights on when the power is connected to start up, as shown in ㉔.
    - The onboard module network indicator will turn on in red once the module is powered on. After successfully registering to the network, it will flash at a frequency of 200ms, as shown in figure ㉗.

## Hardware Connection

This development board's ESP32-S3 UART to USB module and 4G module USB use the same Type-C interface, and you can choose the connection between the USB interface of the 4G module and the ESP32-S3 USB connector or the Type-C interface through the USB channel of the dip switch on the back of the development board. This function is mainly used when the ESP32-S3 uses Tiny USB 4G module communication as a portable WiFi, wireless hotspot.




(/wiki/File:ESP32-S3-A7670E-4G-details-11.png)

(/wiki/File:Esp32-s3-a-sim7670x_240124_05.png)

(/wiki/File:Esp32-s3-a-sim7670x_240124_04.png)

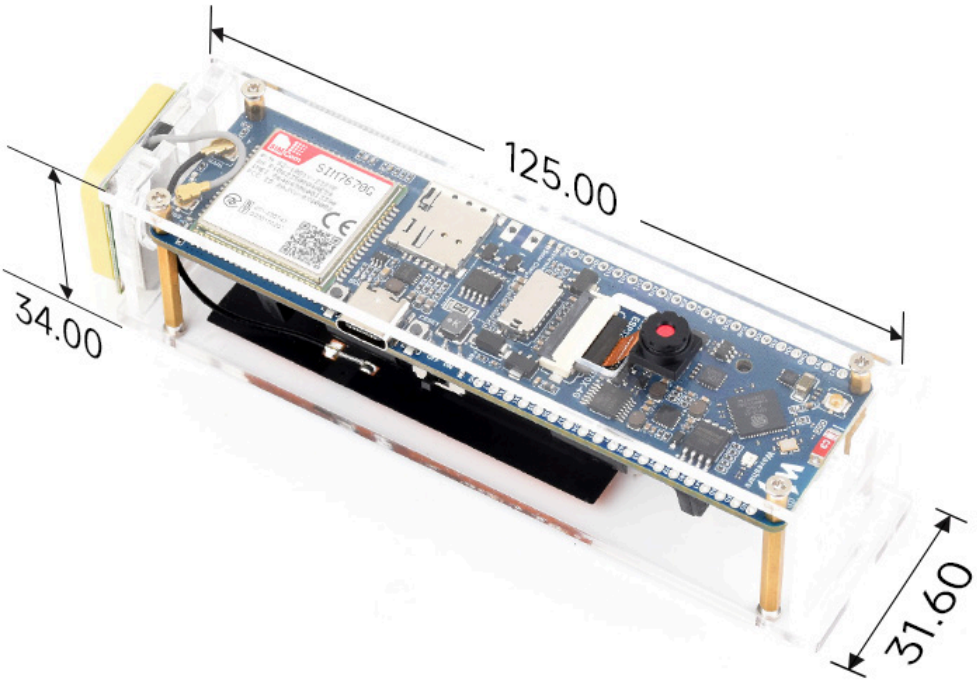
## Solar Panel Charging

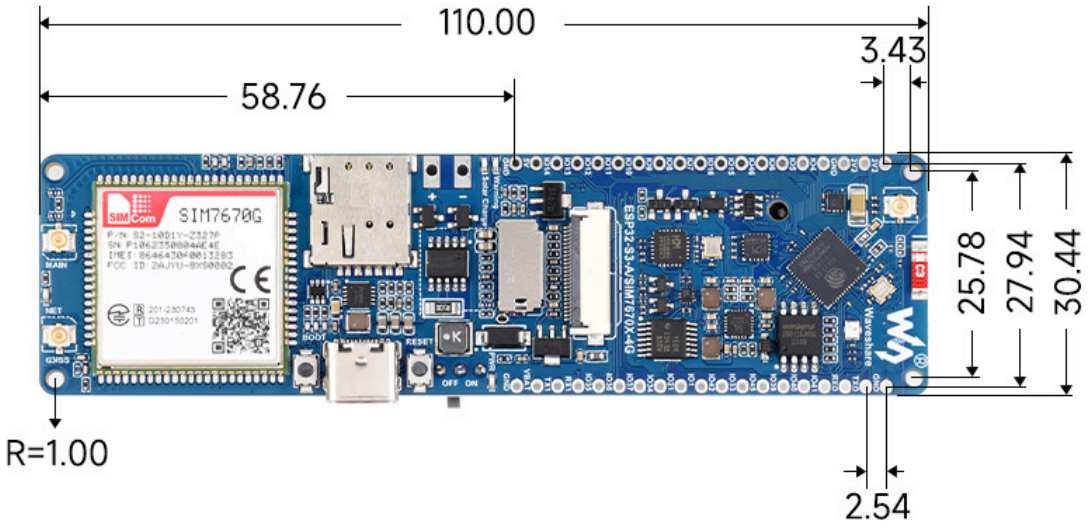
The solar input selection resistor on the back of the development board can switch the maximum voltage of solar input. By default, the 0-ohm resistor is used to connect to the 5V marking position, supporting solar panels with 5~6V voltage input. When using solar panels

with higher voltage input, the corresponding voltage solder joints should be shorted.

(/wiki/File:ESP32-S3-A7670E-4G-Solar.jpg)

## Dimensions




Unit:mm

(/wiki/File:ESP32-S3-SIM7670G-4G-Dim.jpg)

# Development Environment Setup

- **The following development system is Windows by default.**

## ESP-IDF

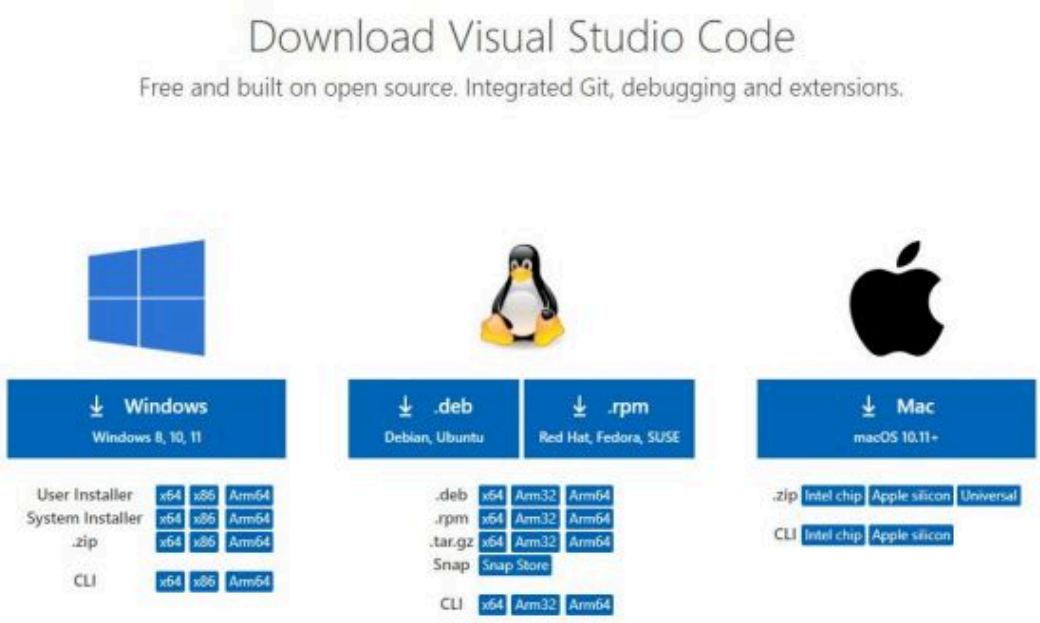
[Collapse]

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

### Develop with VSCode Plug-in

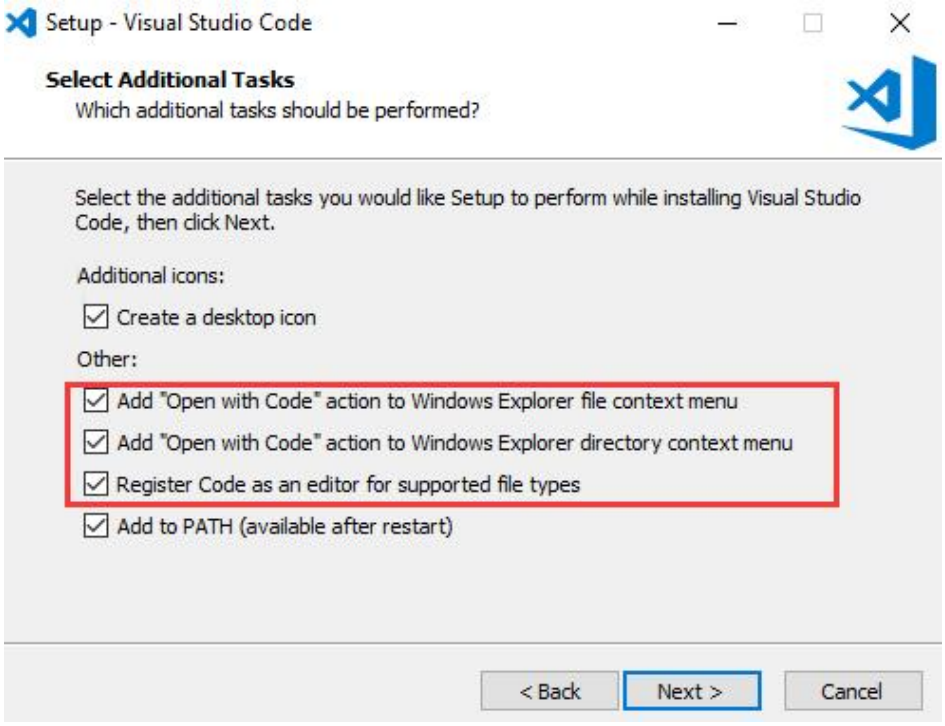
#### Install VSCode

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.



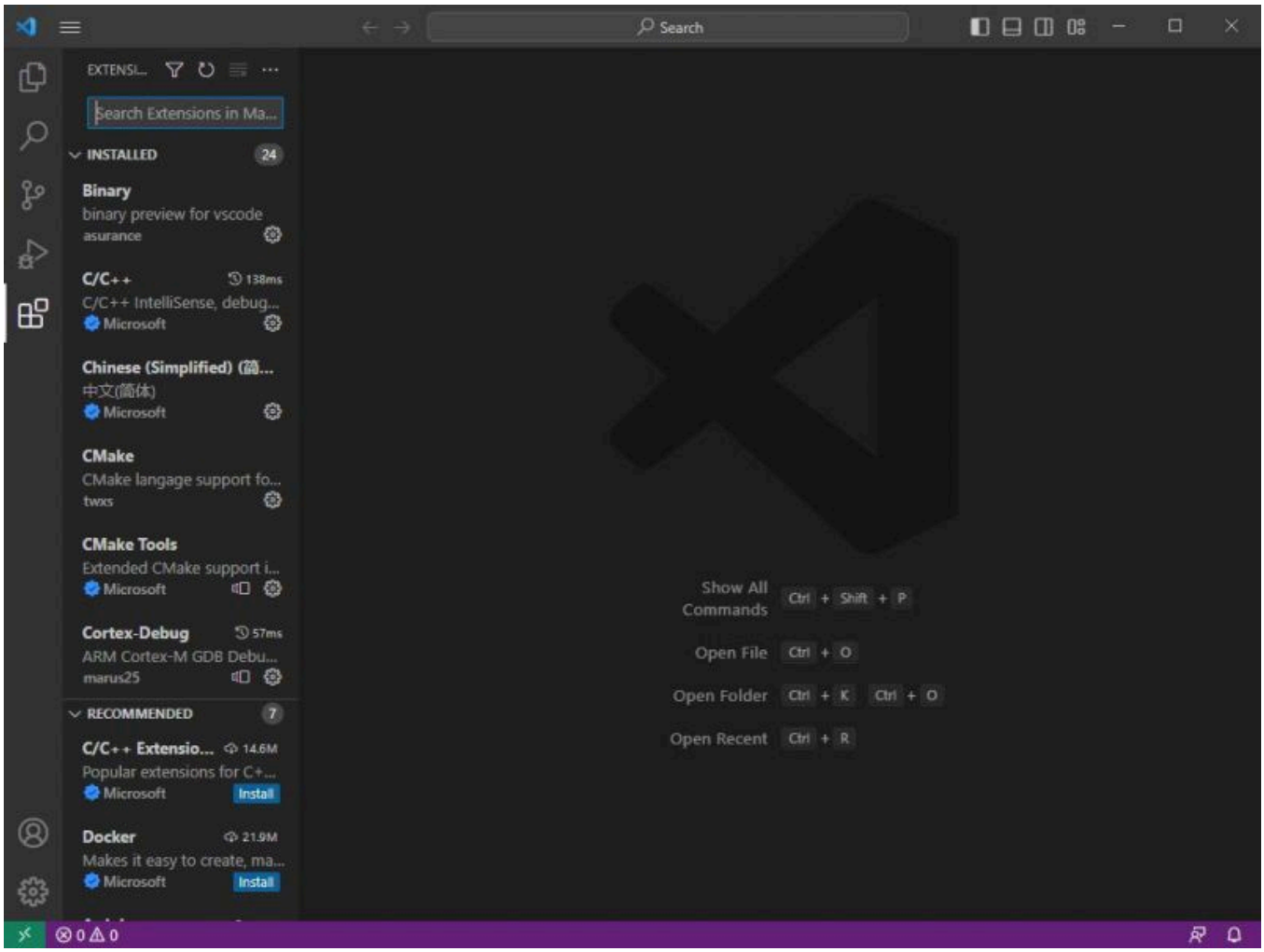
(/wiki/File:ESP32-S3-Pico_06.jpg)

- - 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

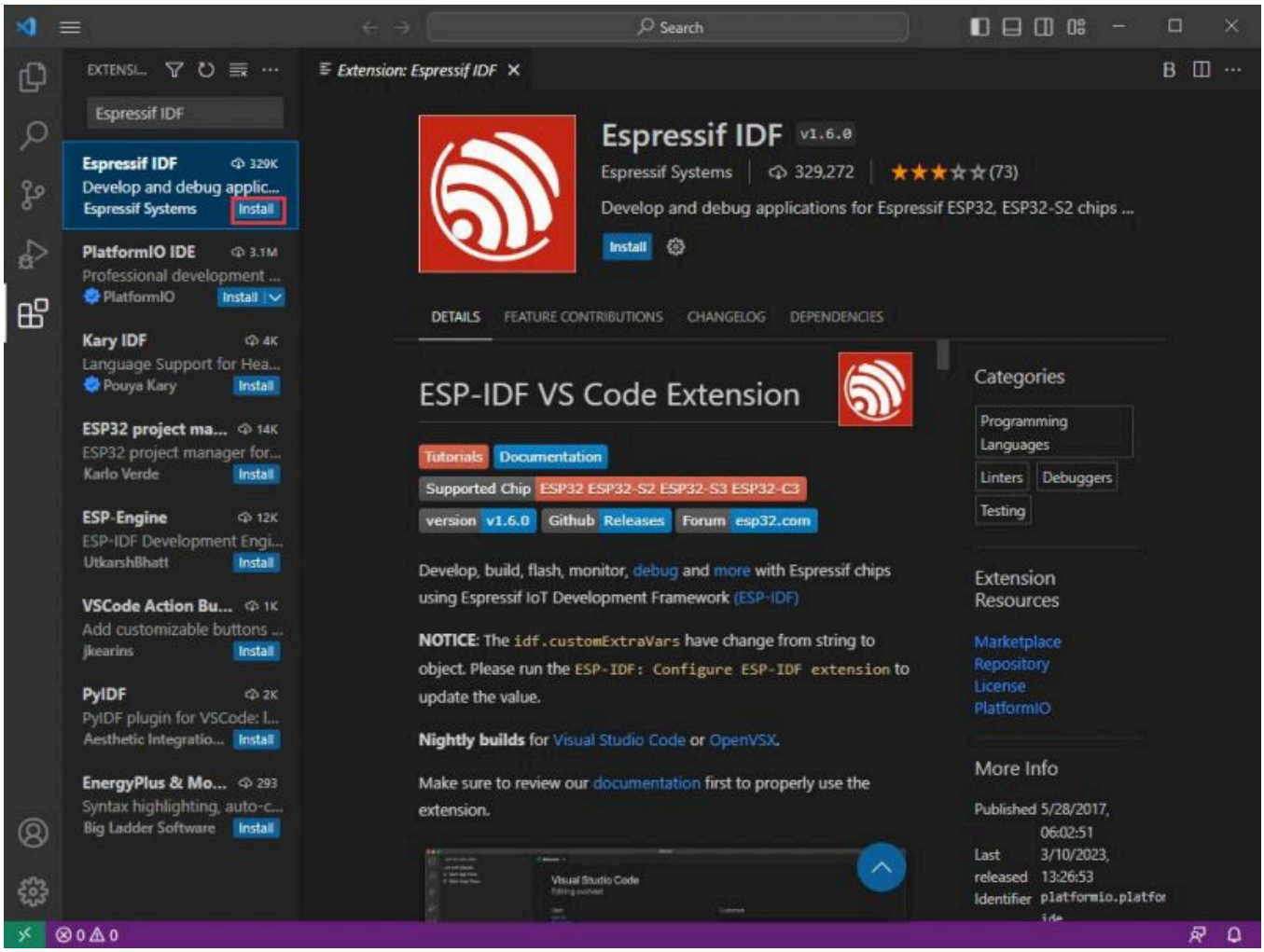
- Note: The latest version of the current plug-in is V1.6.0, for a consistent experience, users can choose the same version as us.

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



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

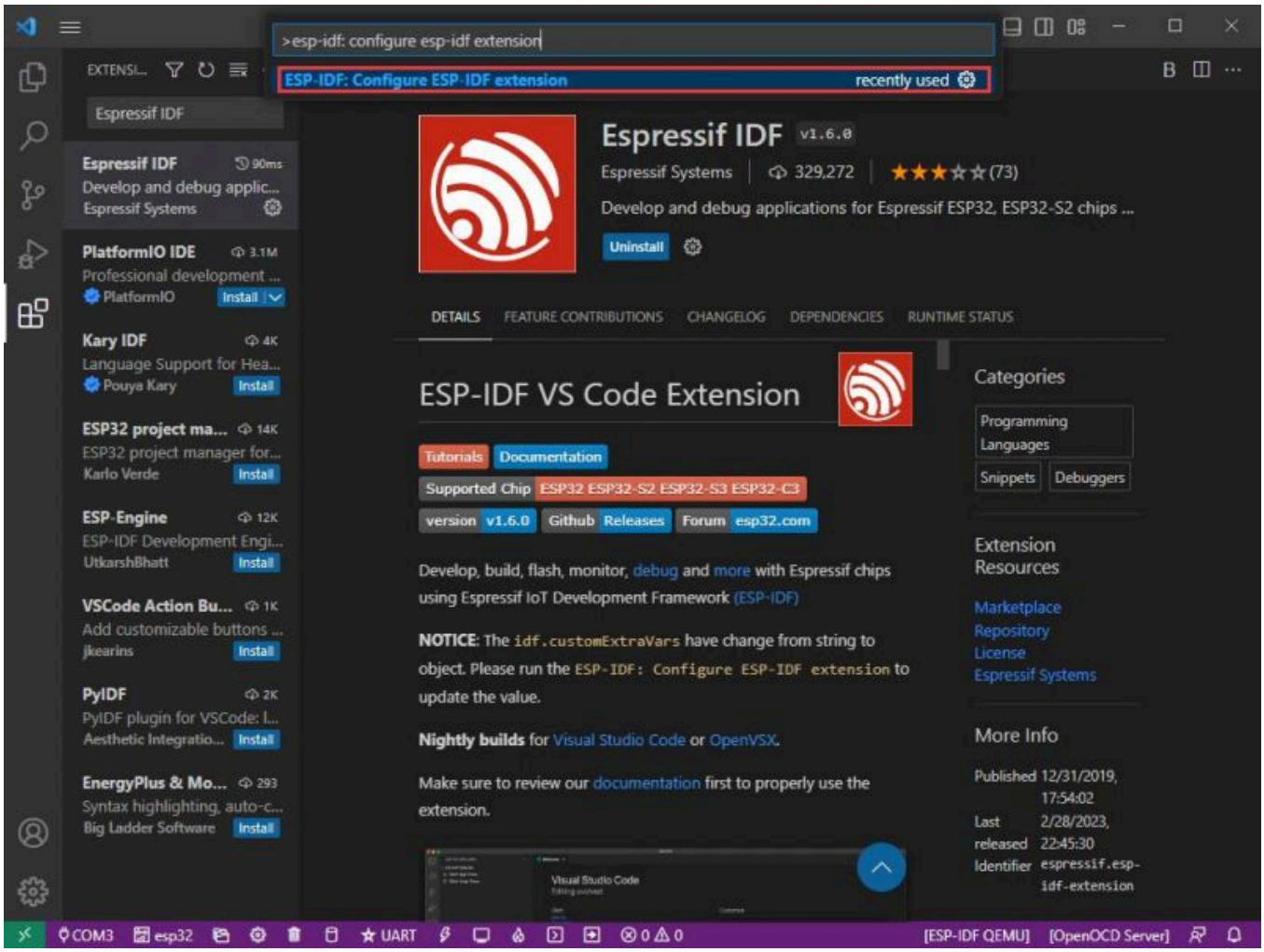
2. In the search bar, type Espressif IDF, select the corresponding plug-in, and click install.



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

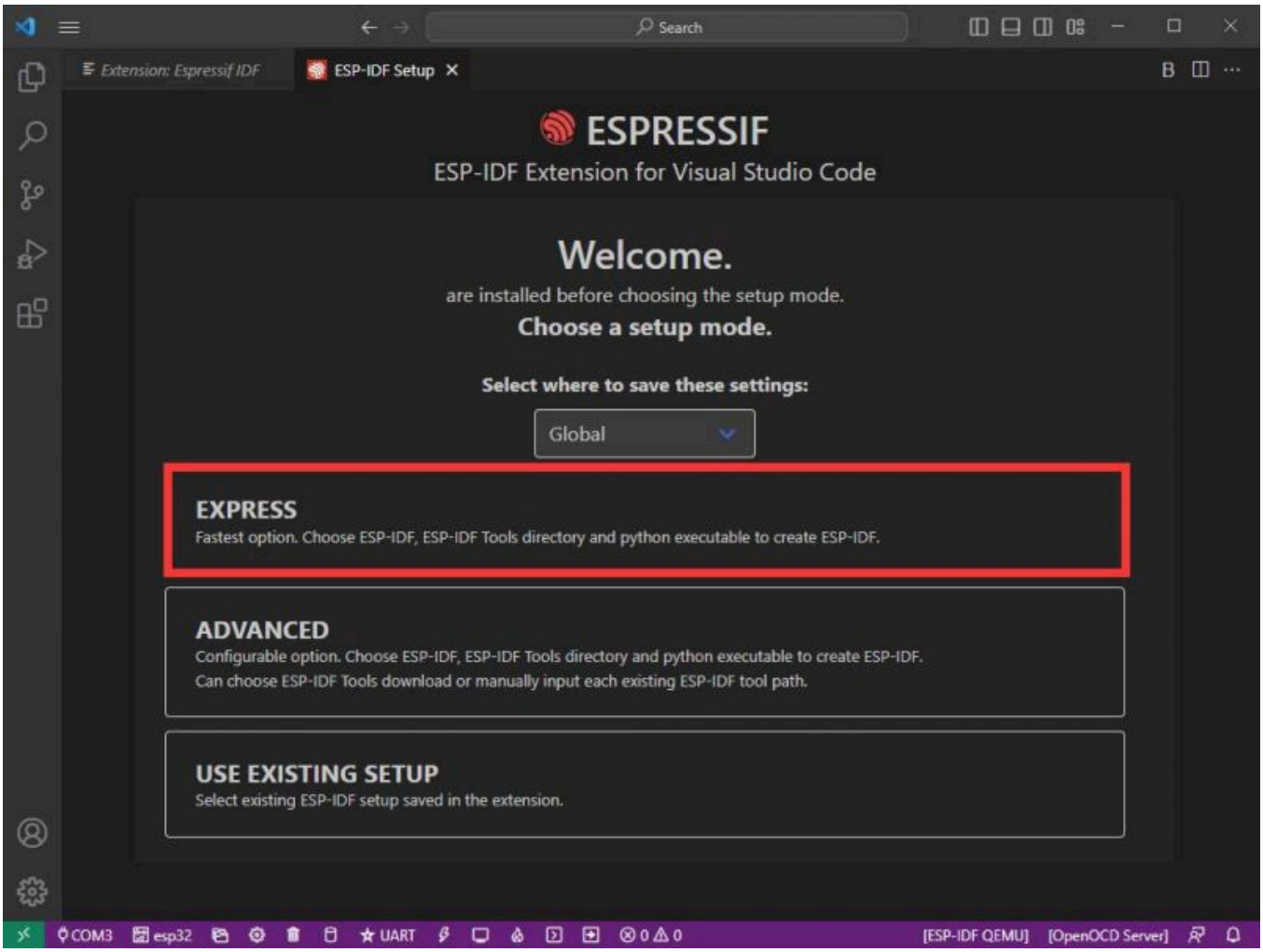
3. Press F1 to enter:

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



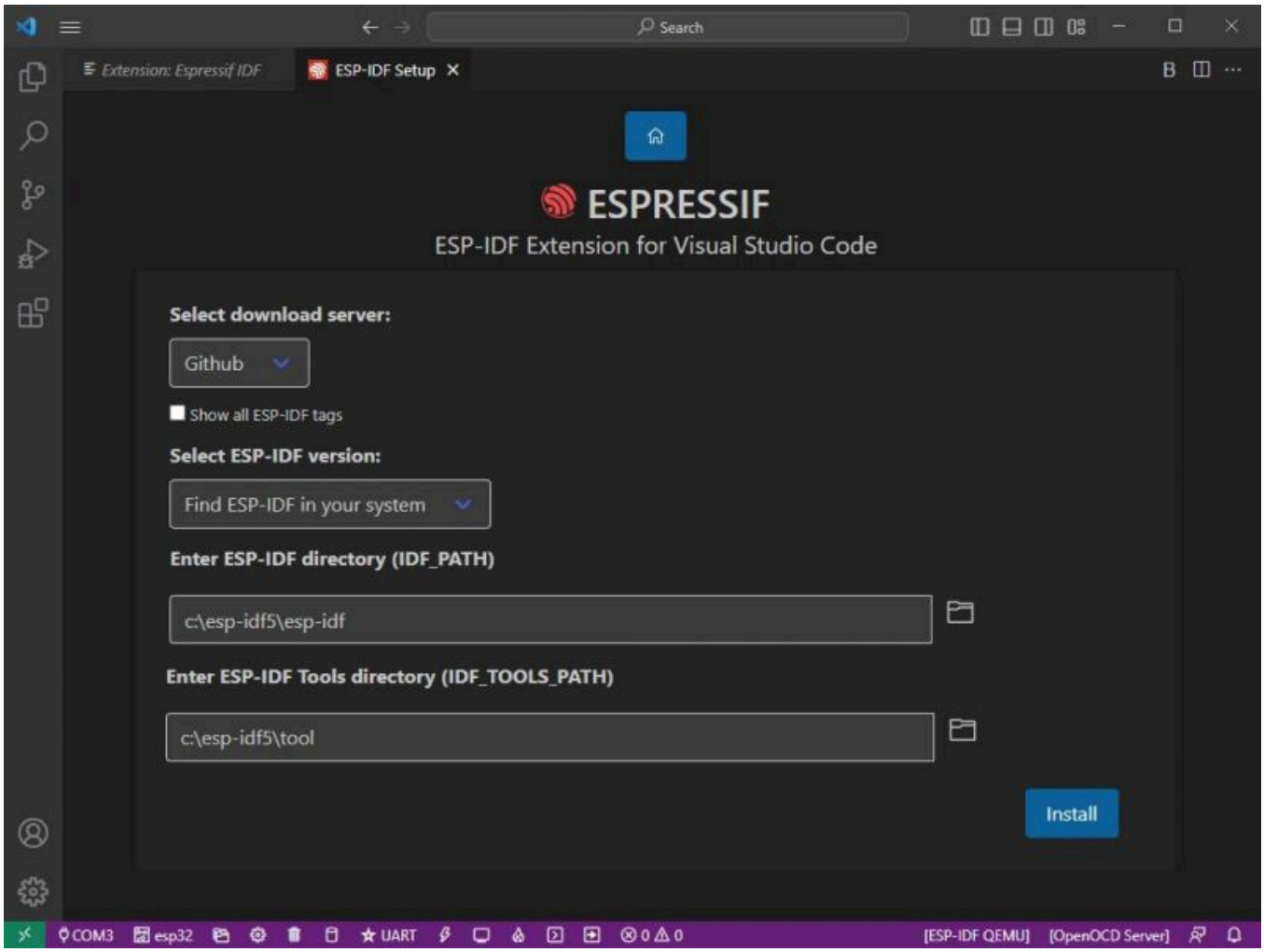
(/wiki/File:ESP32-S3-Pico_09.jpg)

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



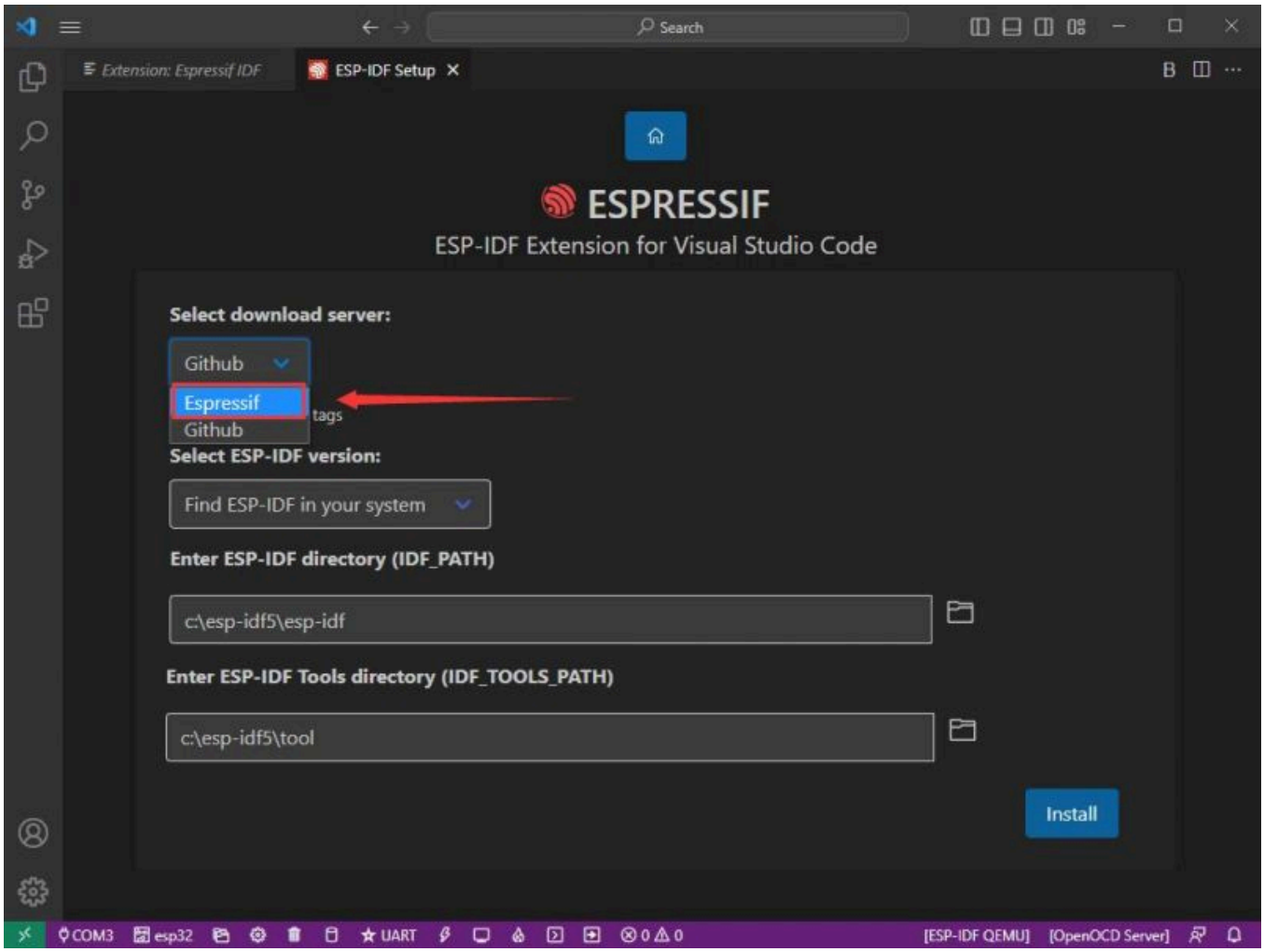
(/wiki/File:ESP32-S3-Pico_10.jpg)

5. Open and display this screen.



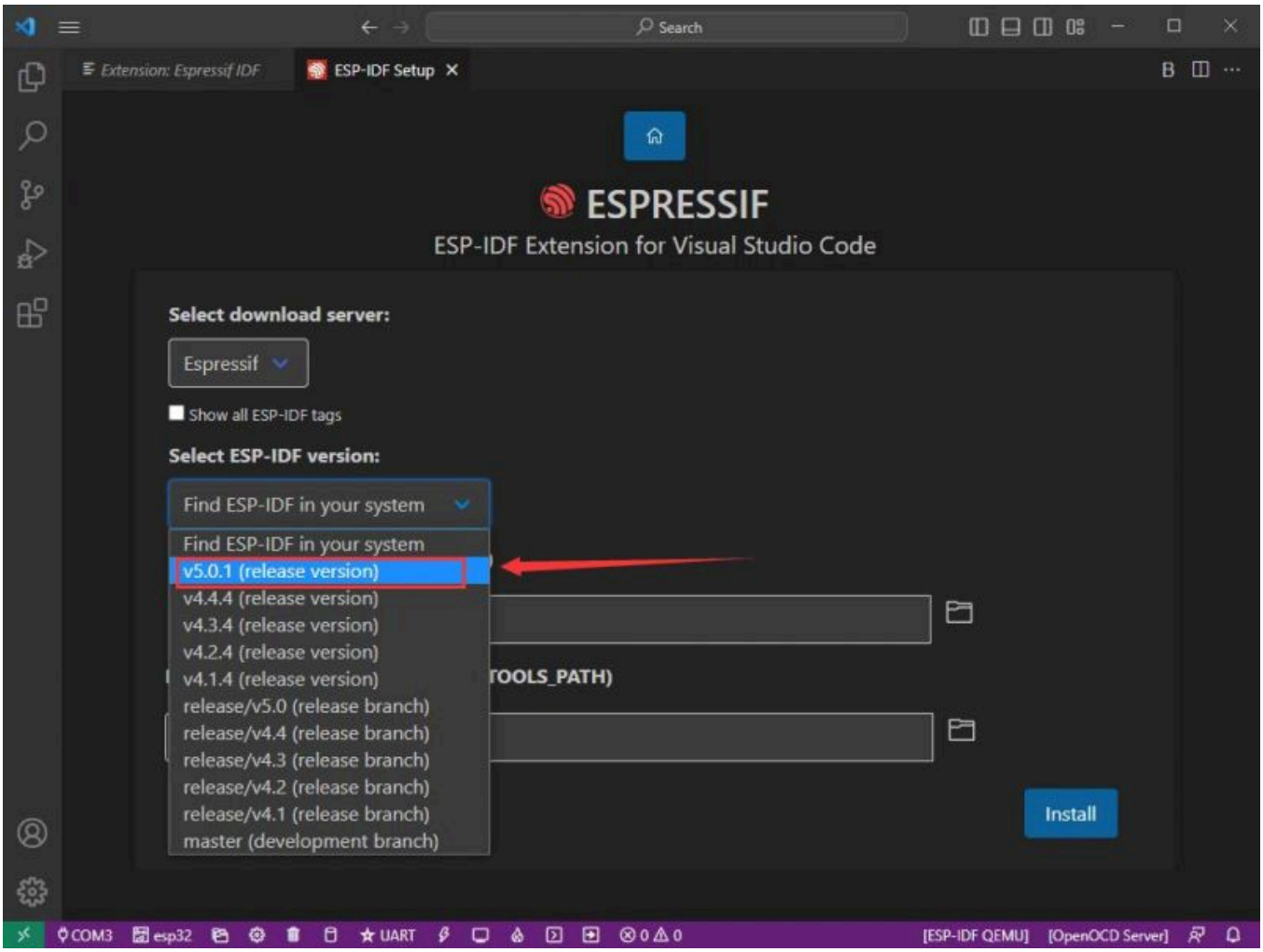
(/wiki/File:ESP32-S3-Pico_11.jpg)

6. Choose a server to download.



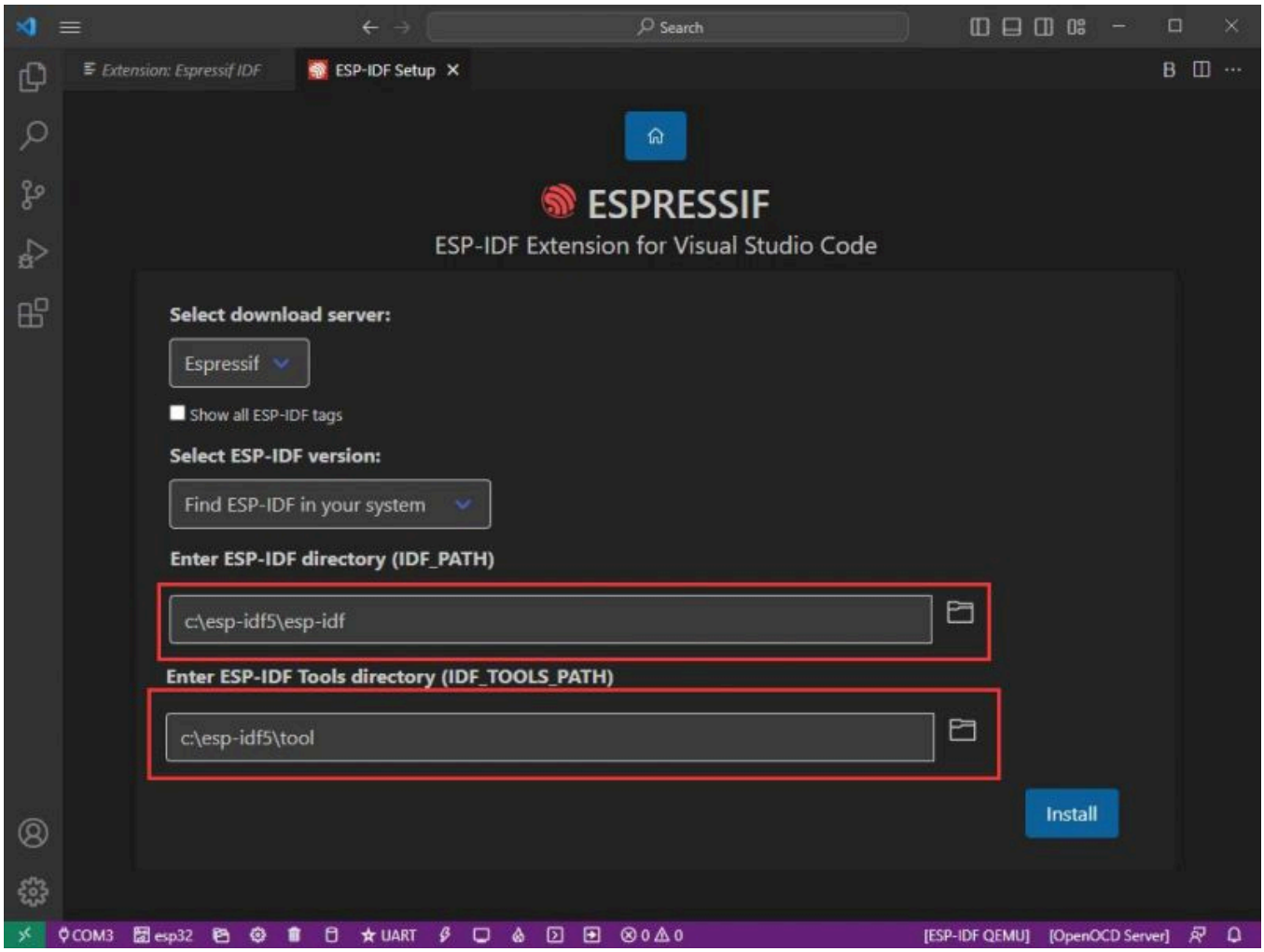

(/wiki/File:ESP32-S3-Pico_12.jpg)

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)

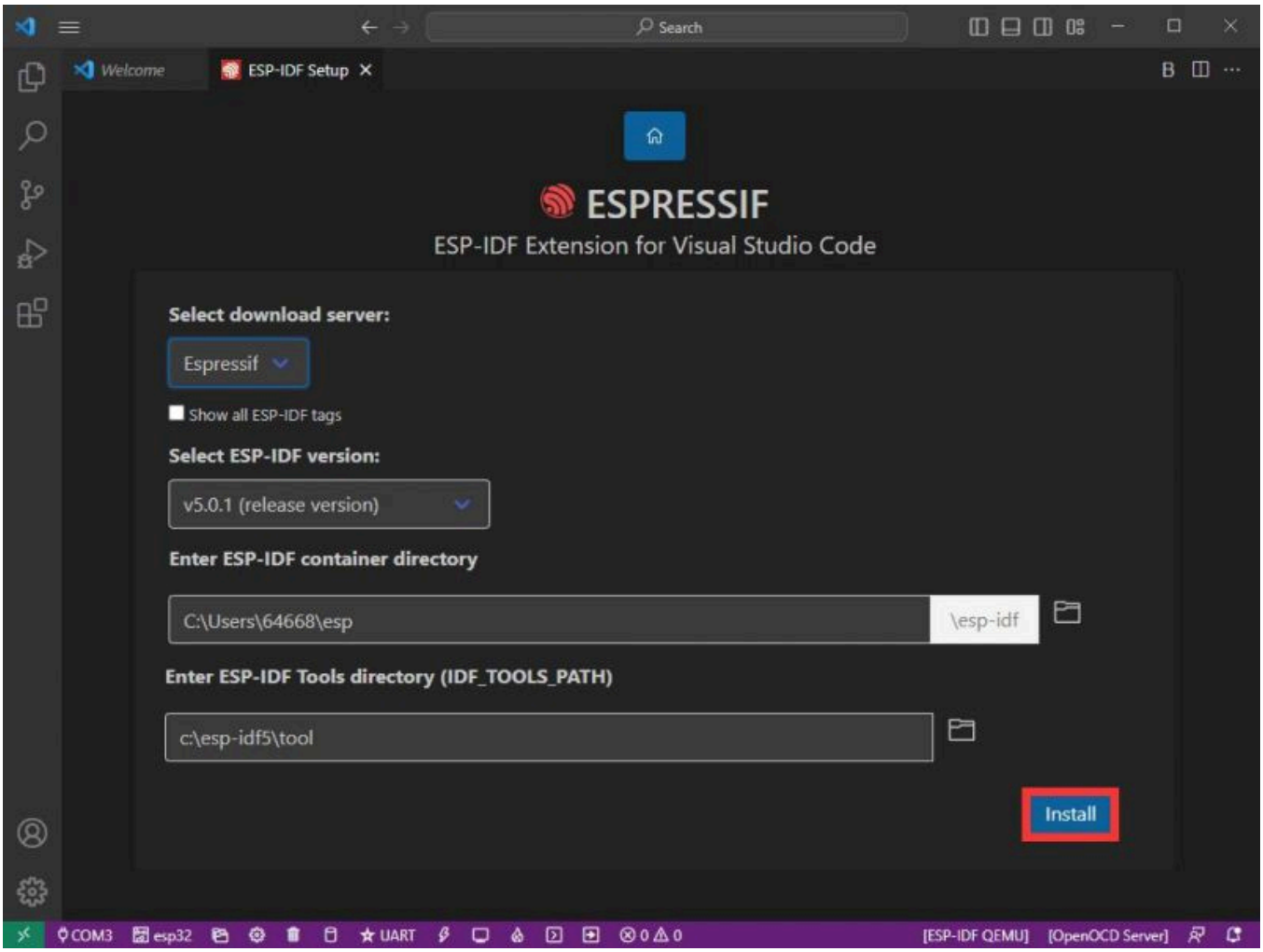
8. The following two are the ESP-IDF directory installation address and the ESP-IDF required tools installation address respectively.



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

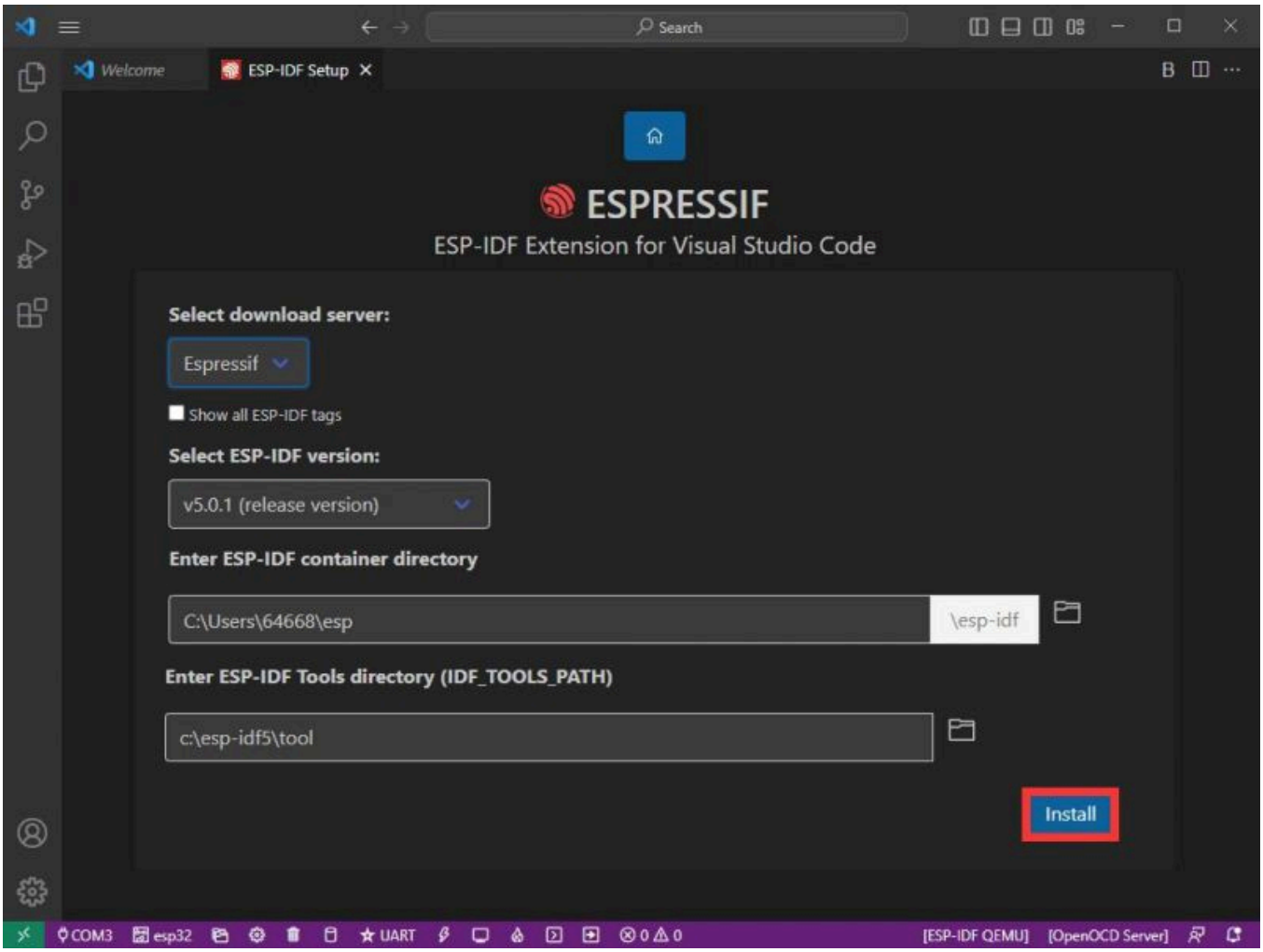
- - **Note: If you have installed ESP-IDF before, or if it has failed, please make sure to delete the file completely or create a new path.**

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



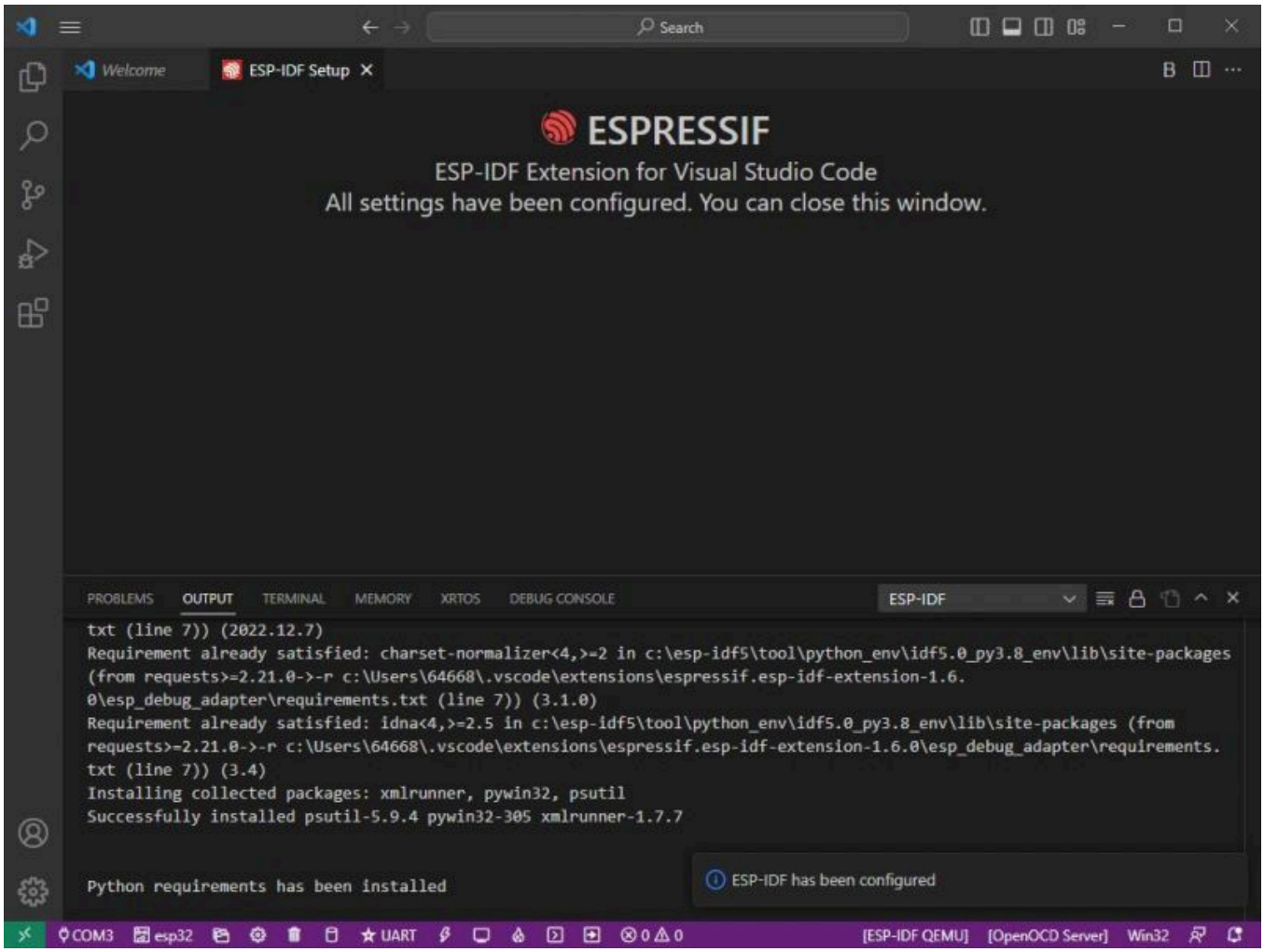

(/wiki/File:ESP32-S3-Pico_15.jpg)

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)

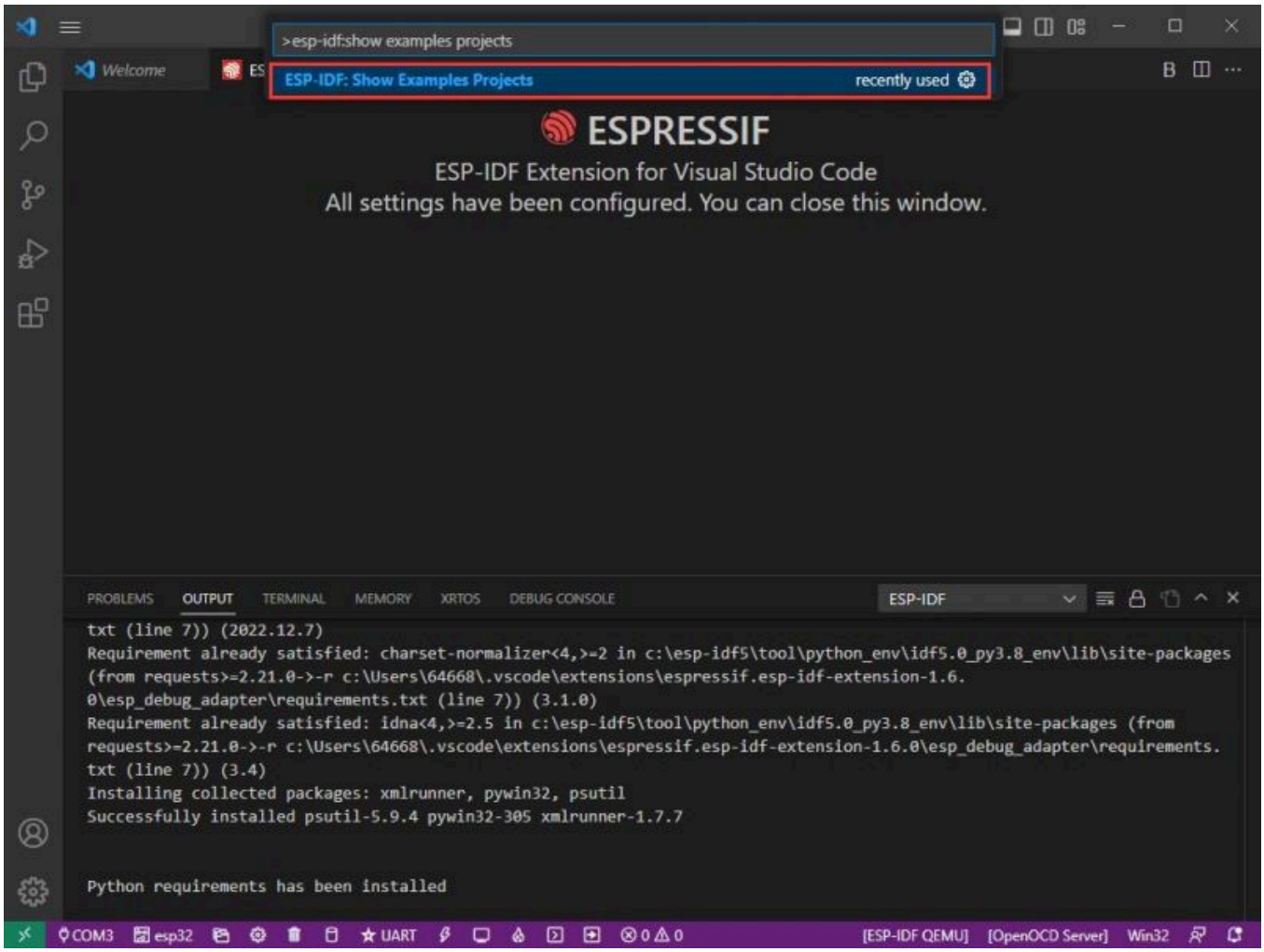
### Official Demo Usage

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

## Creating a Demo

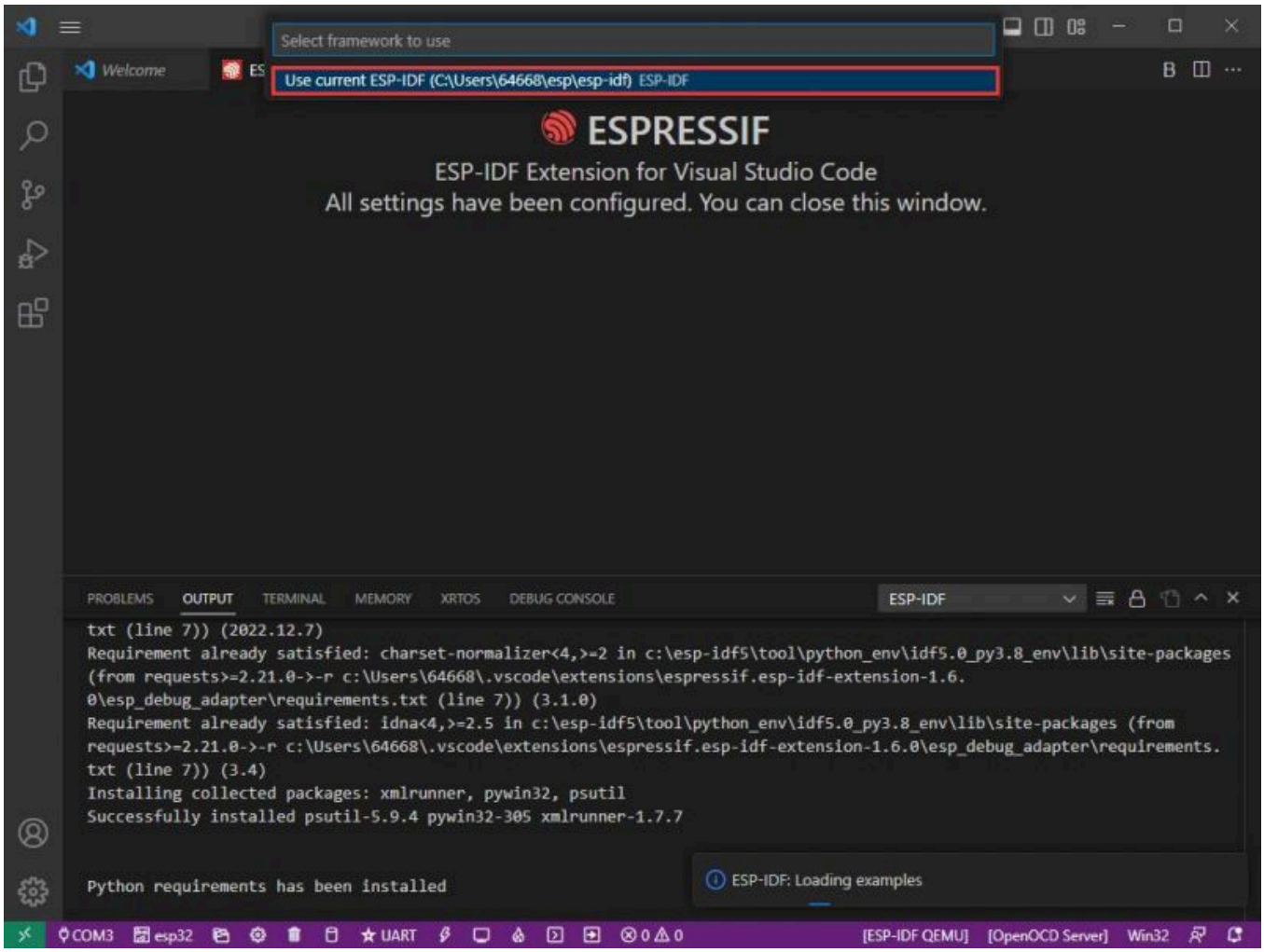
1. Using the shortcut F1, type:

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



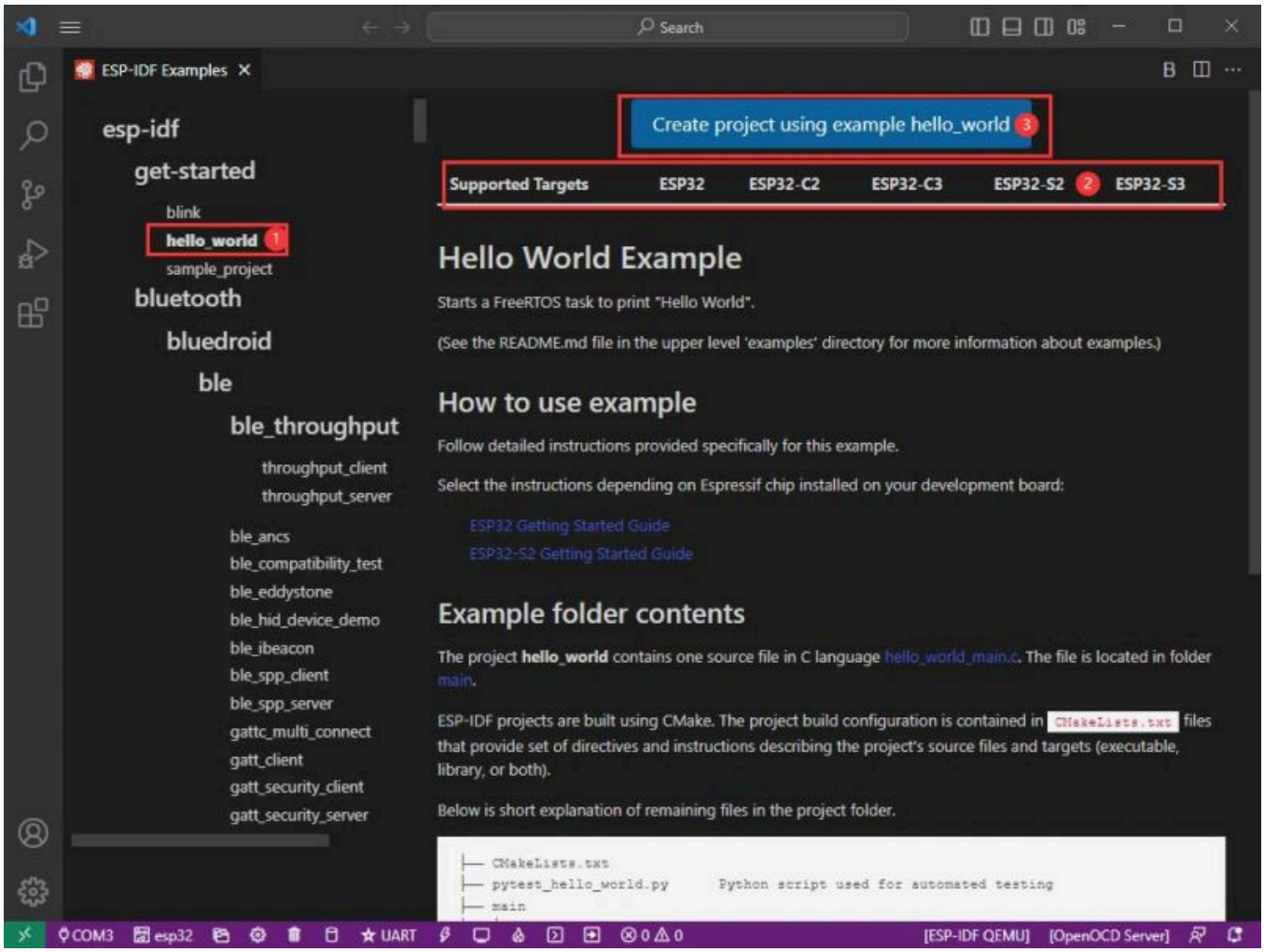
(/wiki/File:ESP32-S3-Pico_18.jpg)

2. Choose your current IDF version:



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

3. Take "Hello World" as an example:



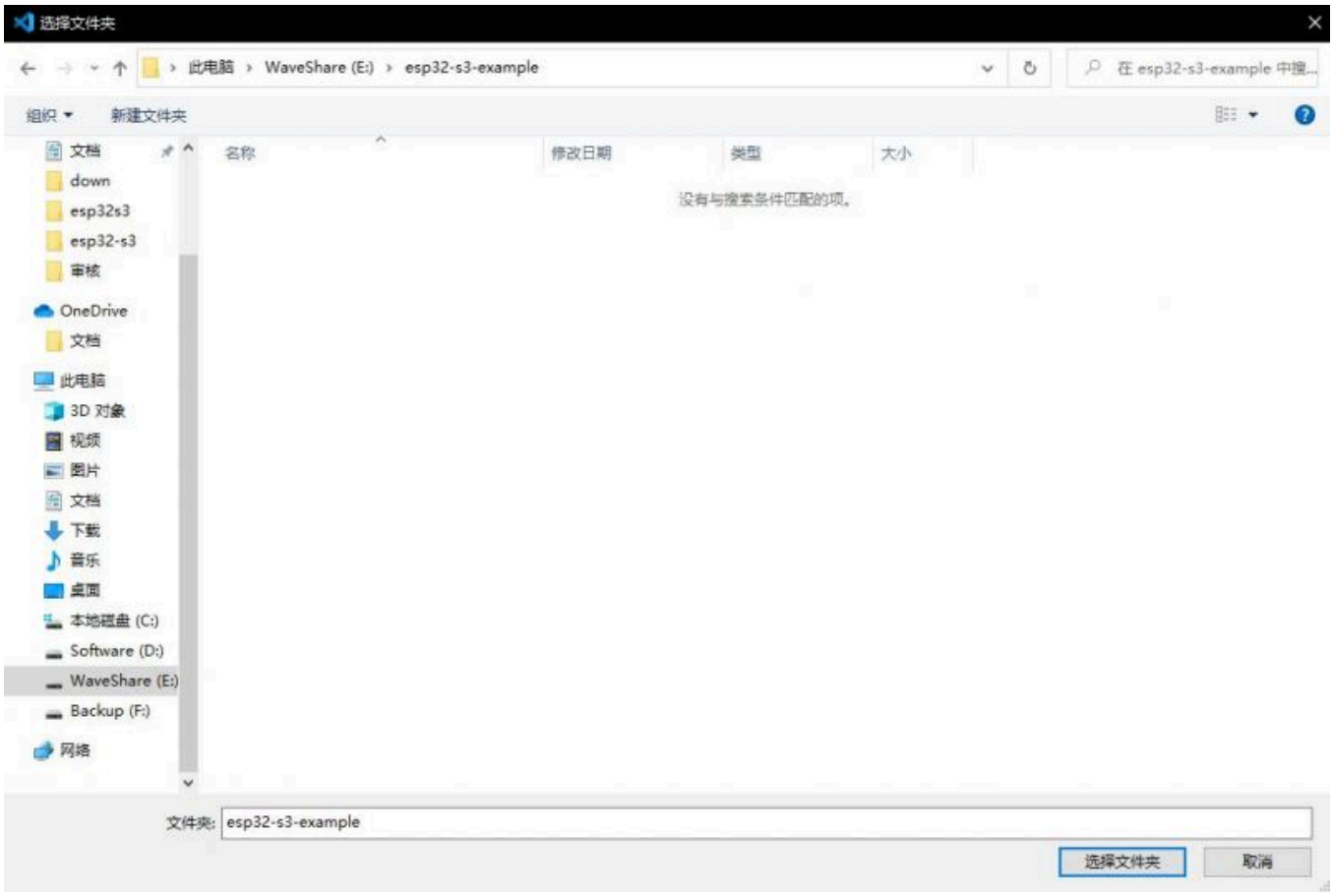
(/wiki/File:ESP32-S3-Pico_20.jpg)

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.

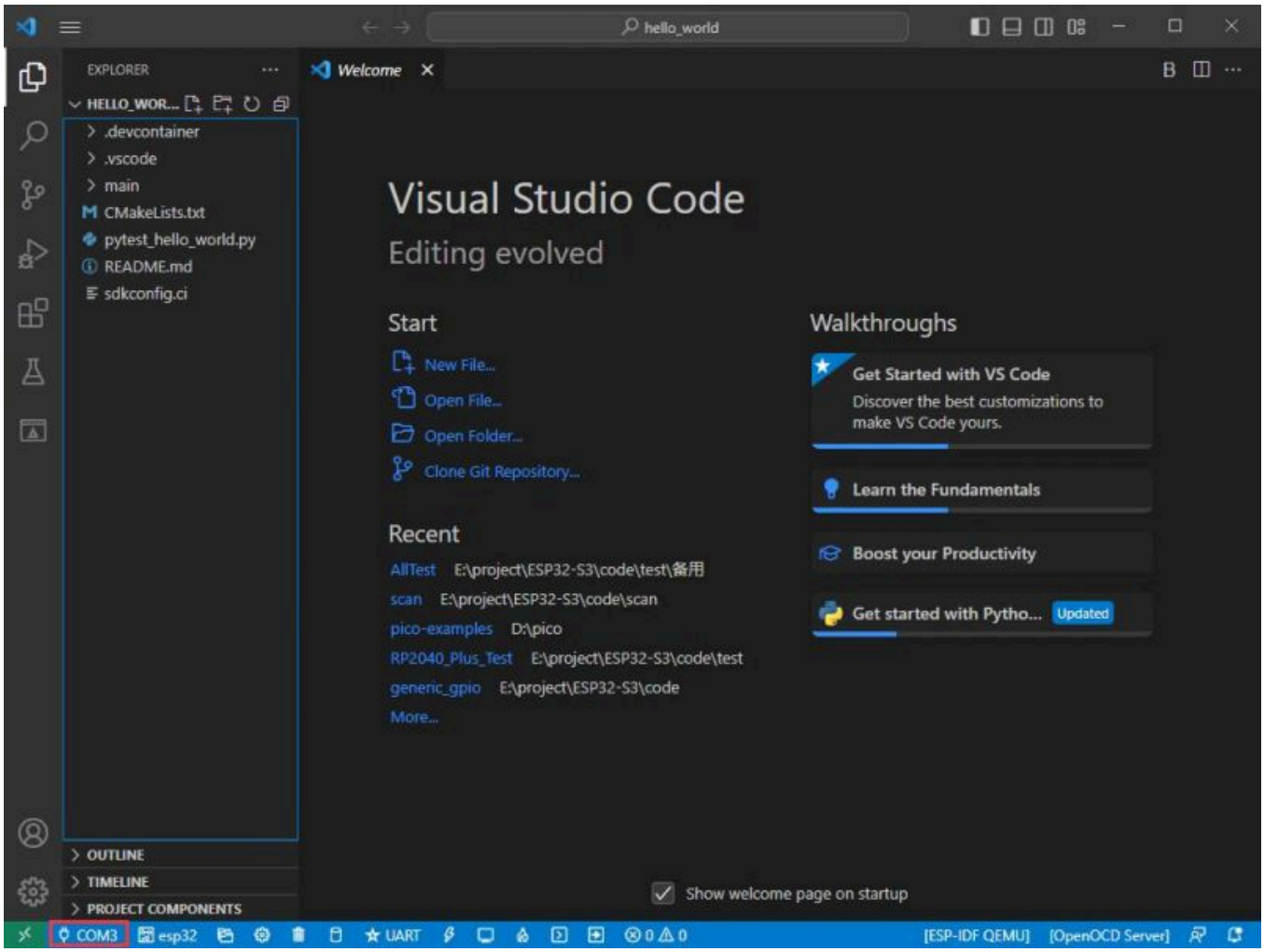
7. Choose the path to place the demo and ensure that there is no folder with the same name as the demo.




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

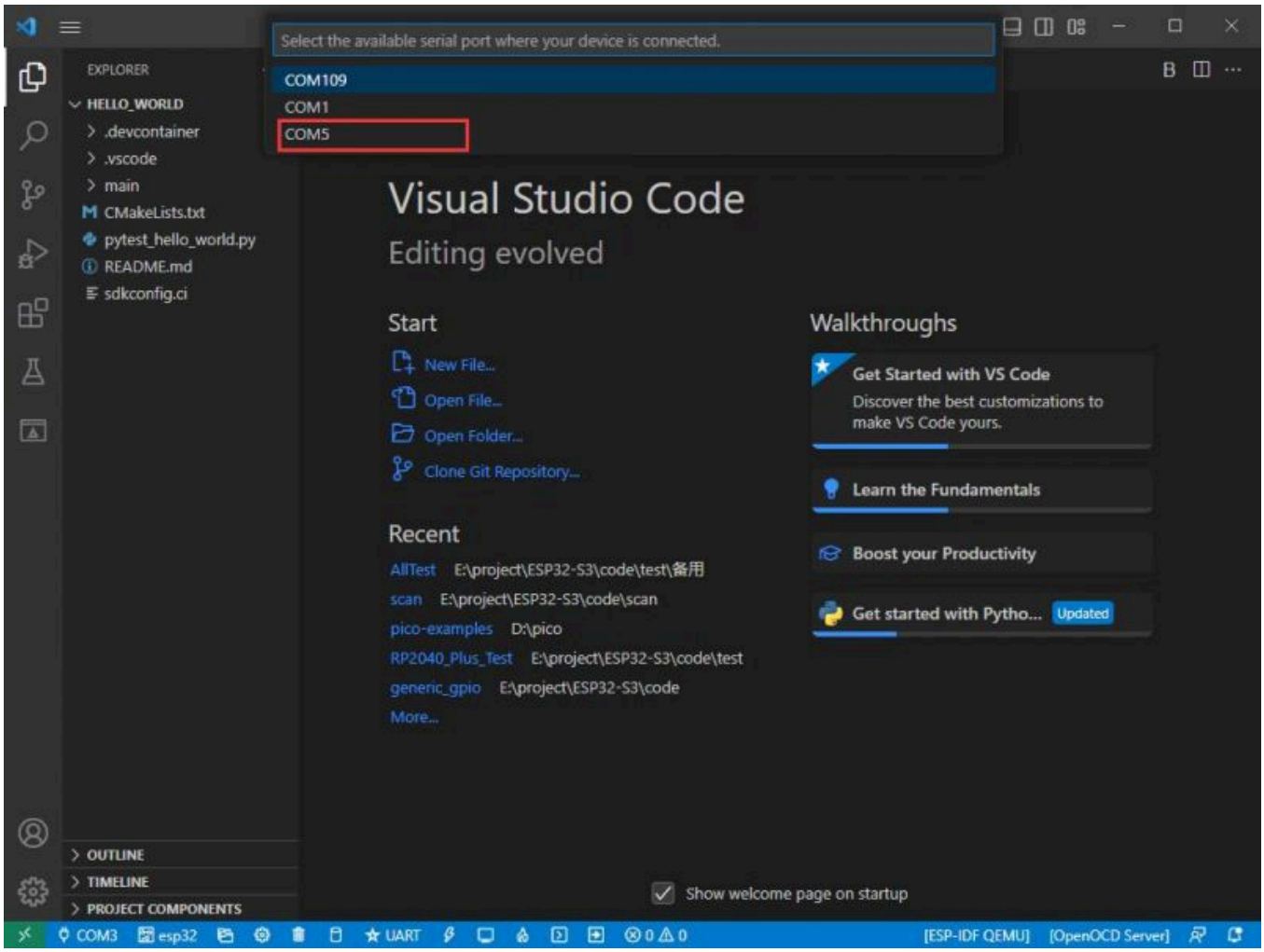
## Modify COM Port

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



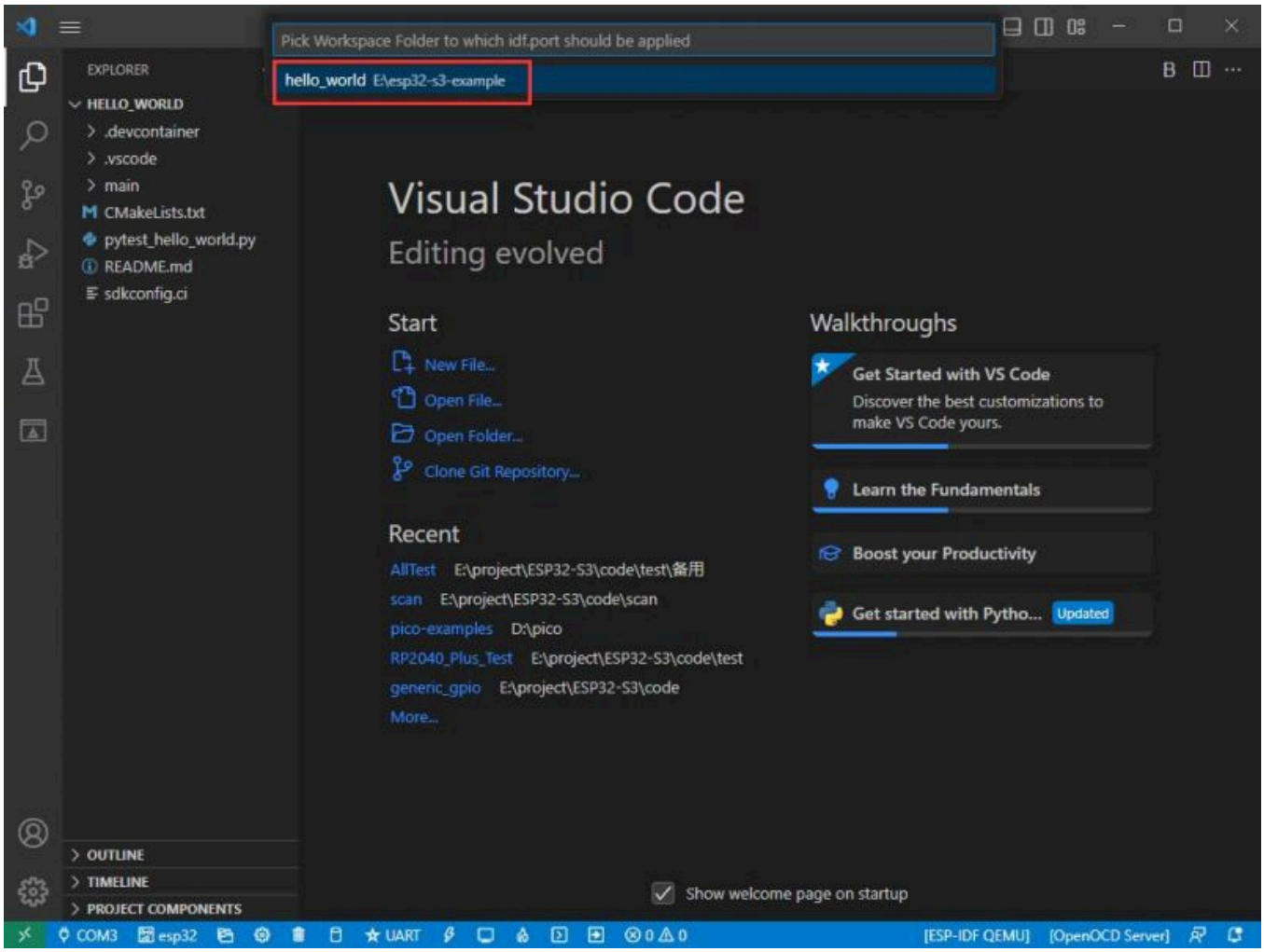
(/wiki/File:ESP32-S3-Pico_22.jpg)

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.

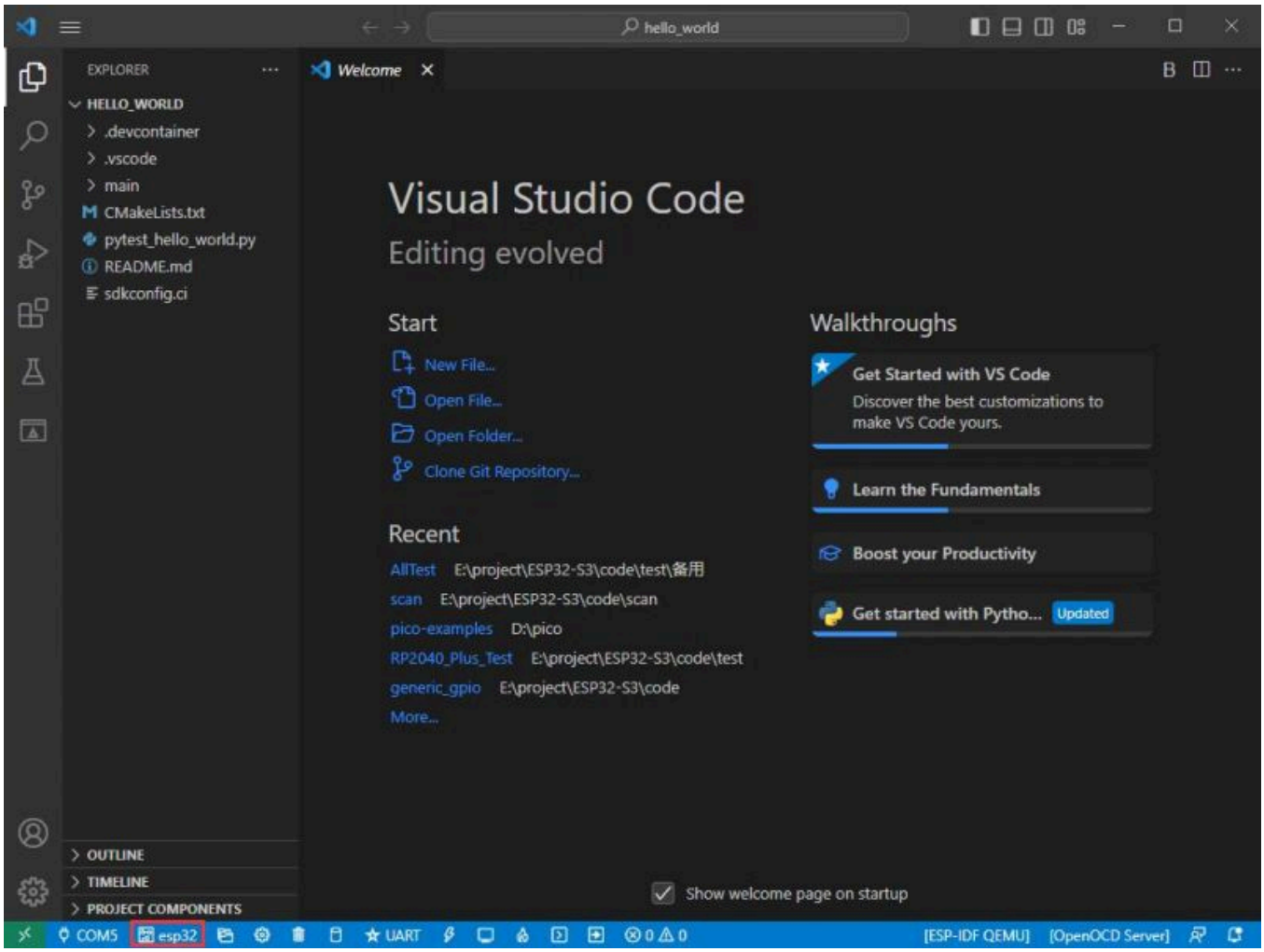


(/wiki/File:ESP32-S3-Pico_24.jpg)

4. Then the COM port is modified.

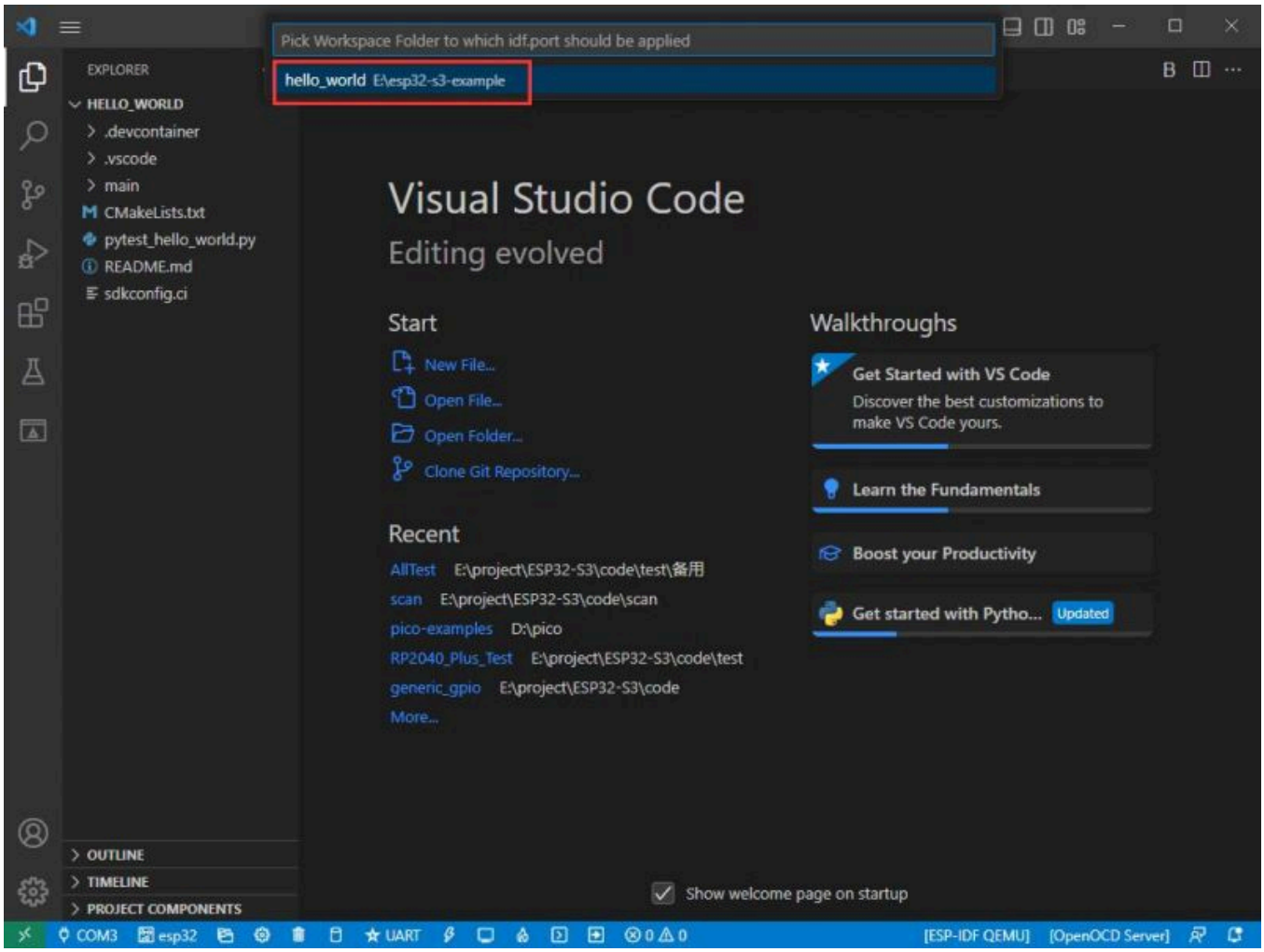
## Modify the Driver

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



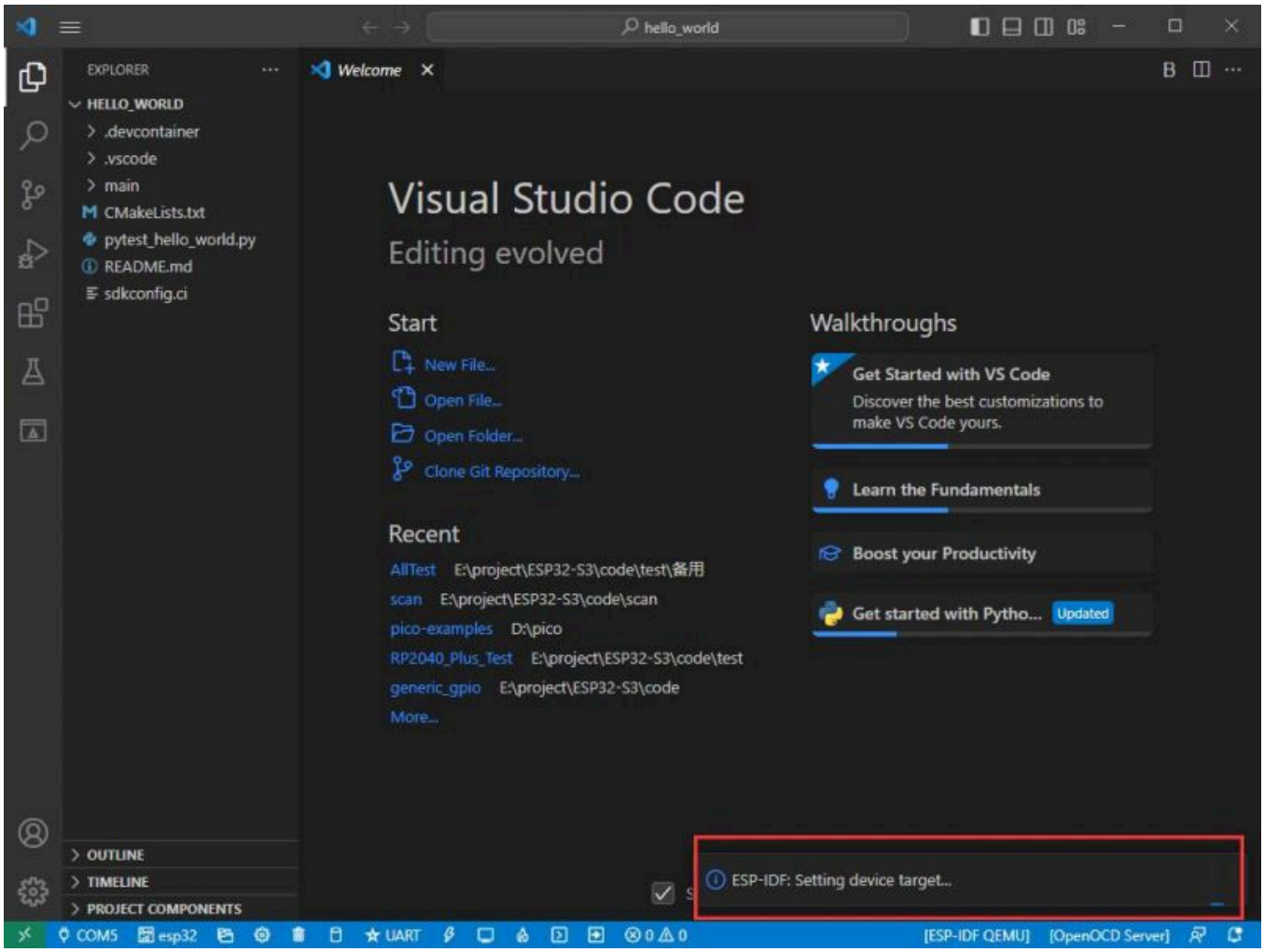
(/wiki/File:ESP32-S3-Pico_25.jpg)

2. Choose the project or demo:



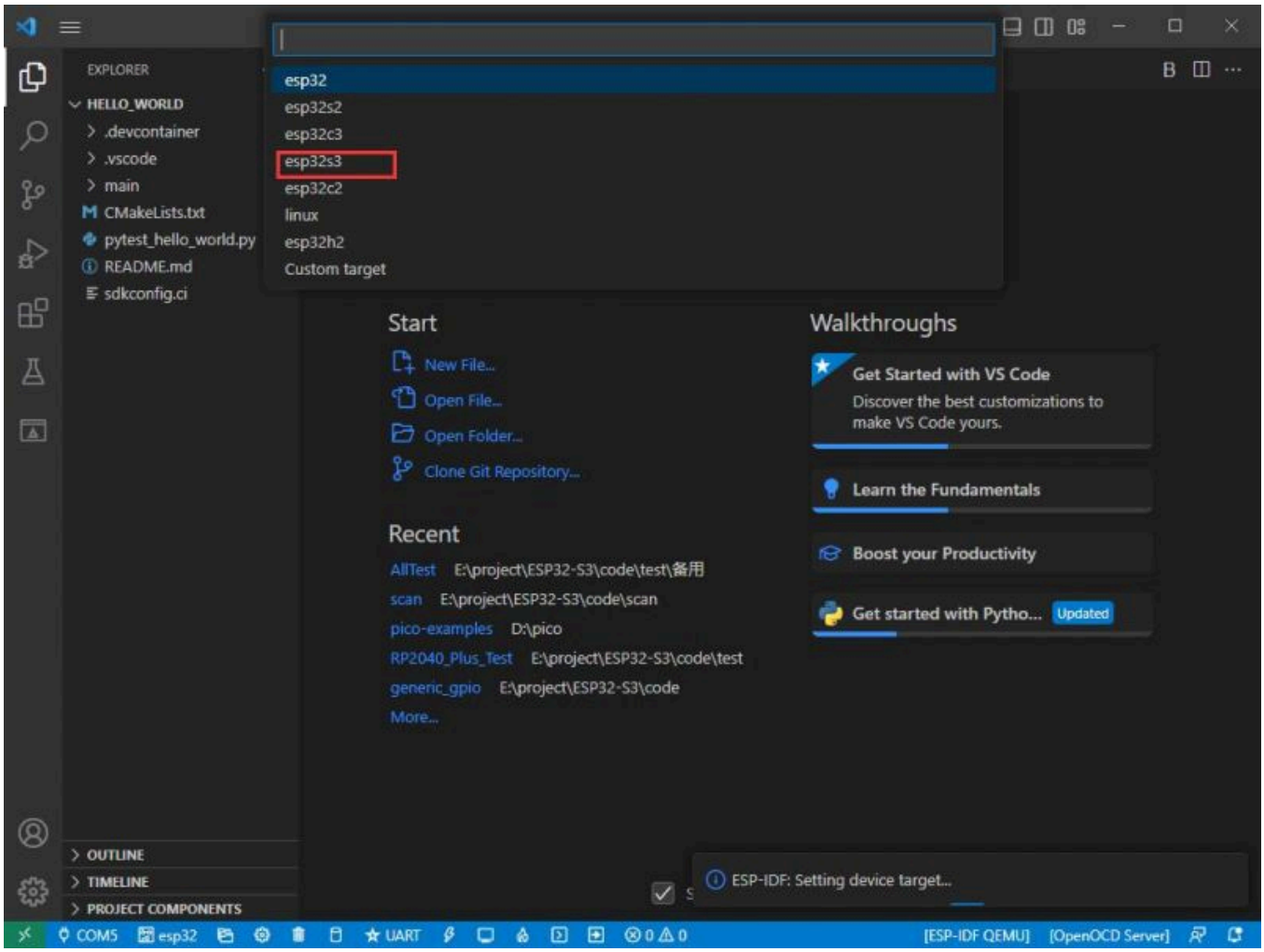
(/wiki/File:ESP32-S3-Pico_24.jpg)

3. Wait for a few seconds after clicking.



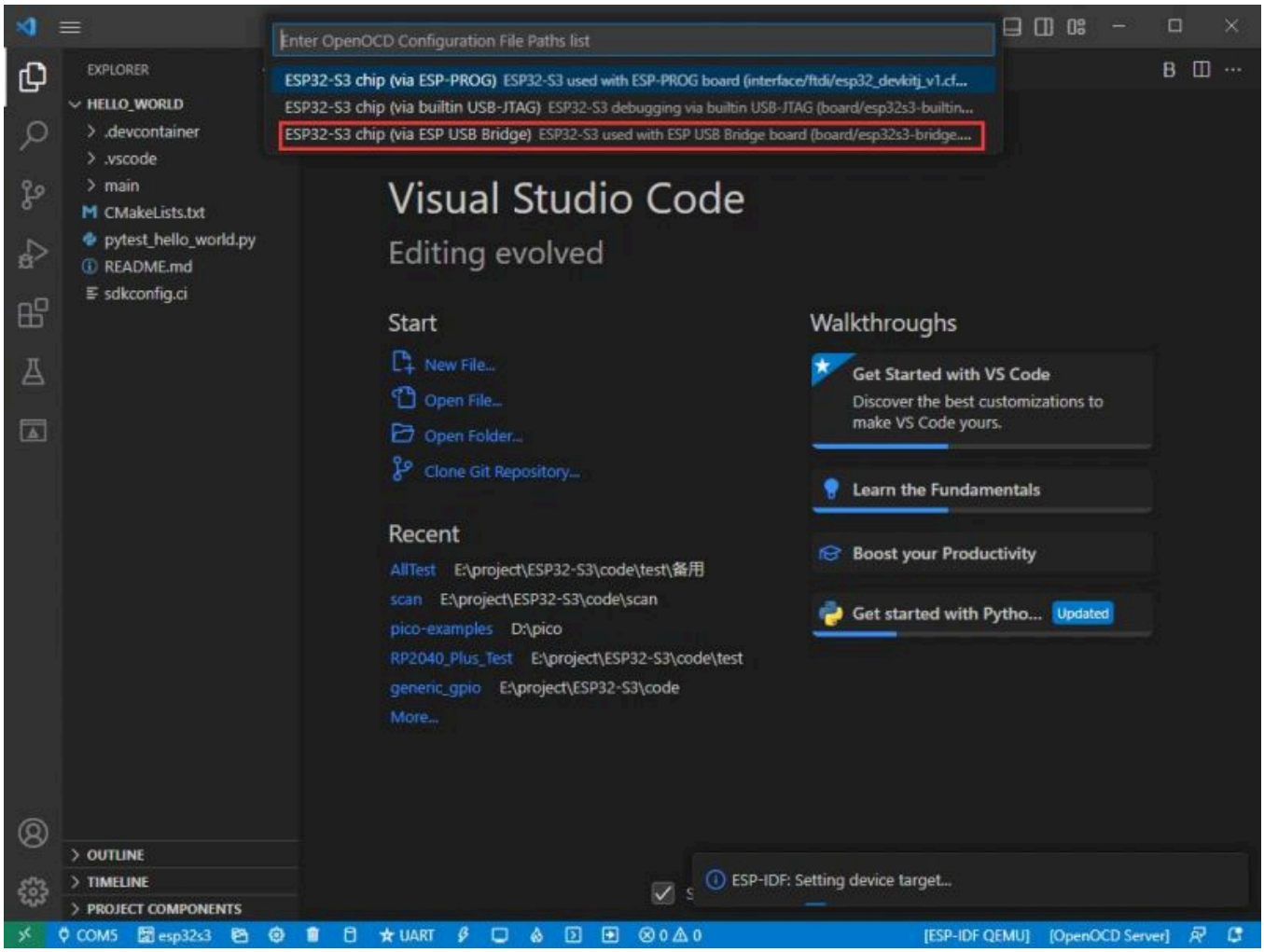
(/wiki/File:ESP32-S3-Pico_27.jpg)

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



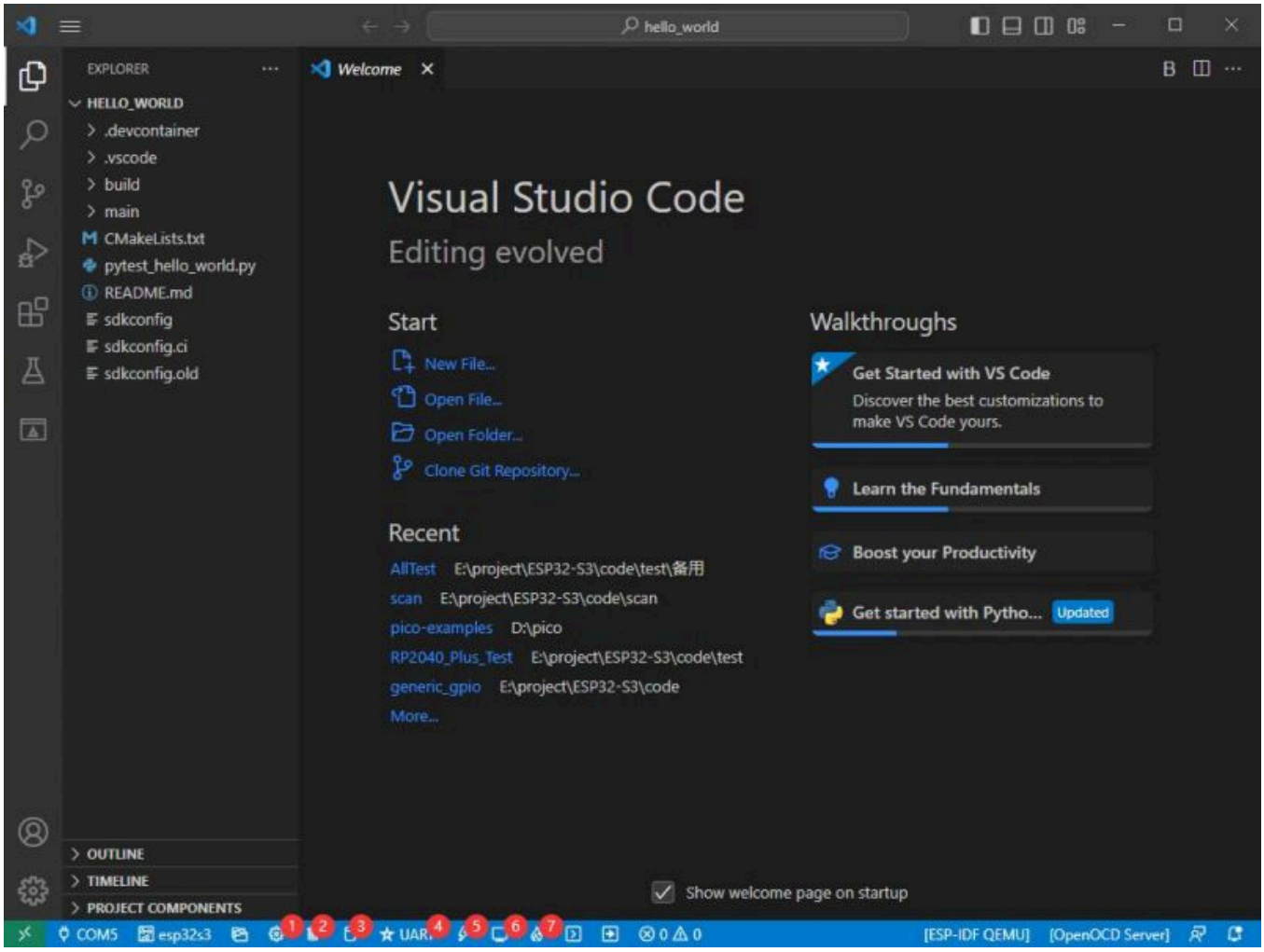
(/wiki/File:ESP32-S3-Pico_28.jpg)

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

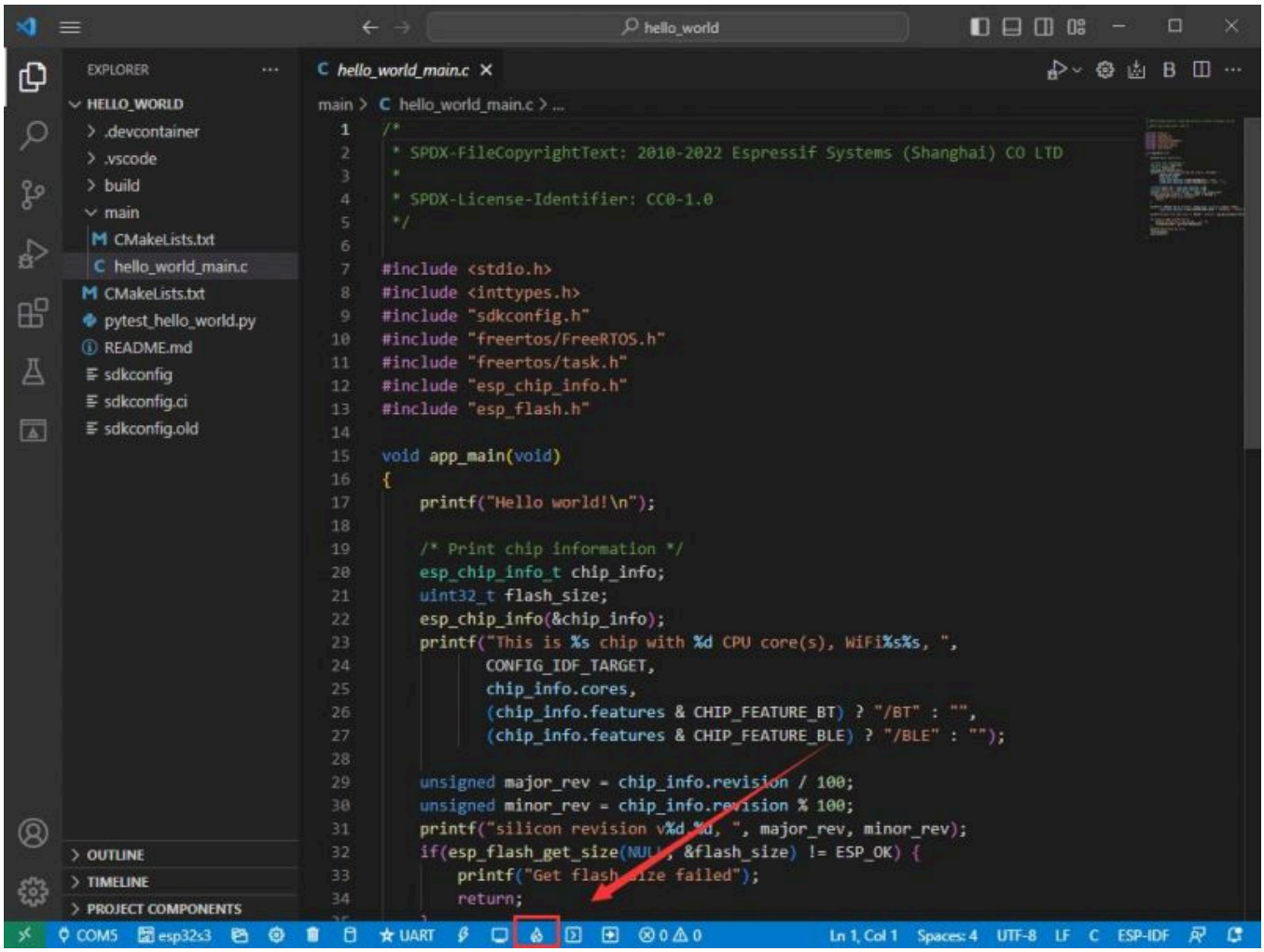



(/wiki/File:ESP32-S3-Pico_30.jpg)

- 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).

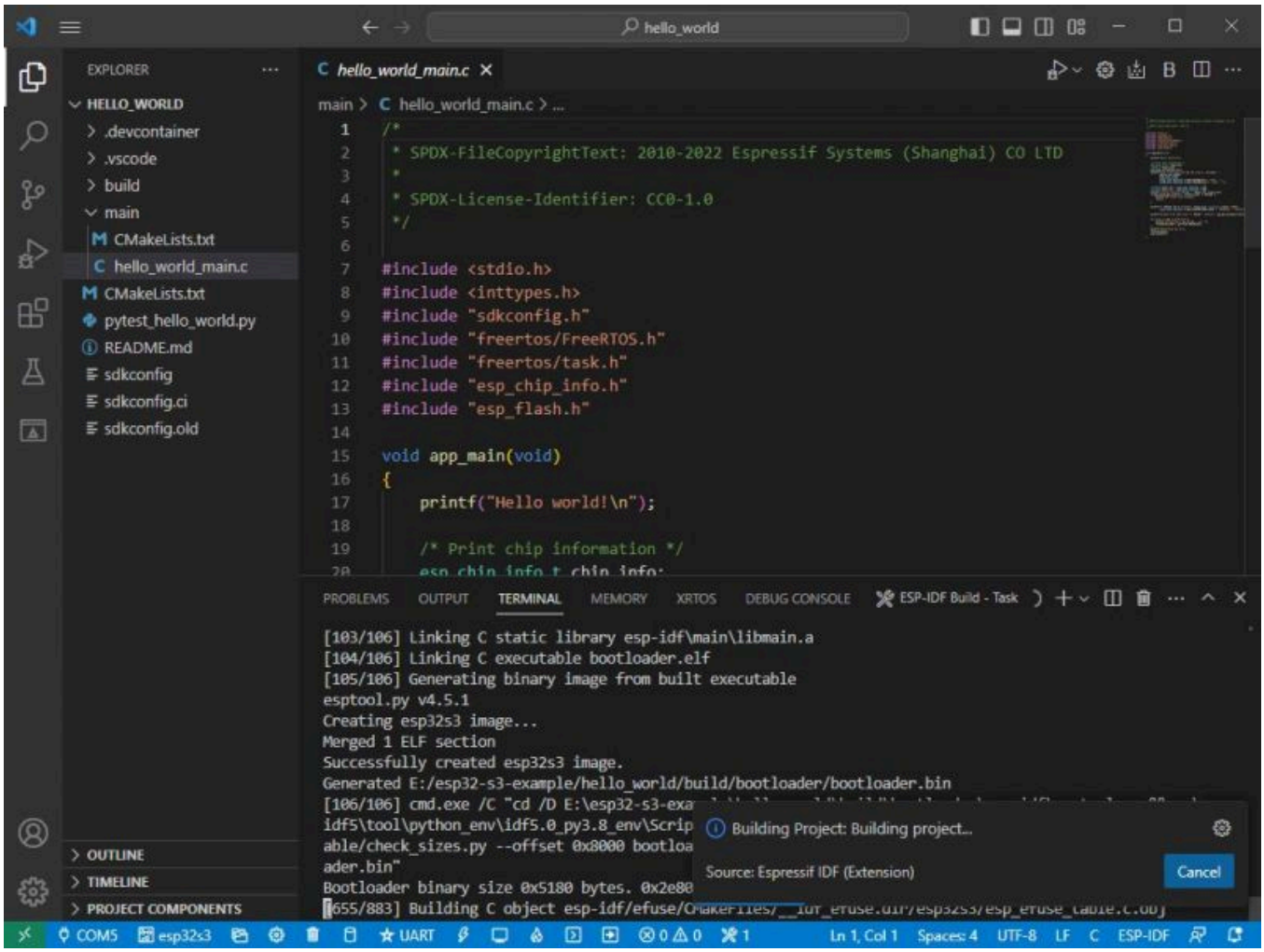
## Compile, Program, and Serial Port Monitoring

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



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

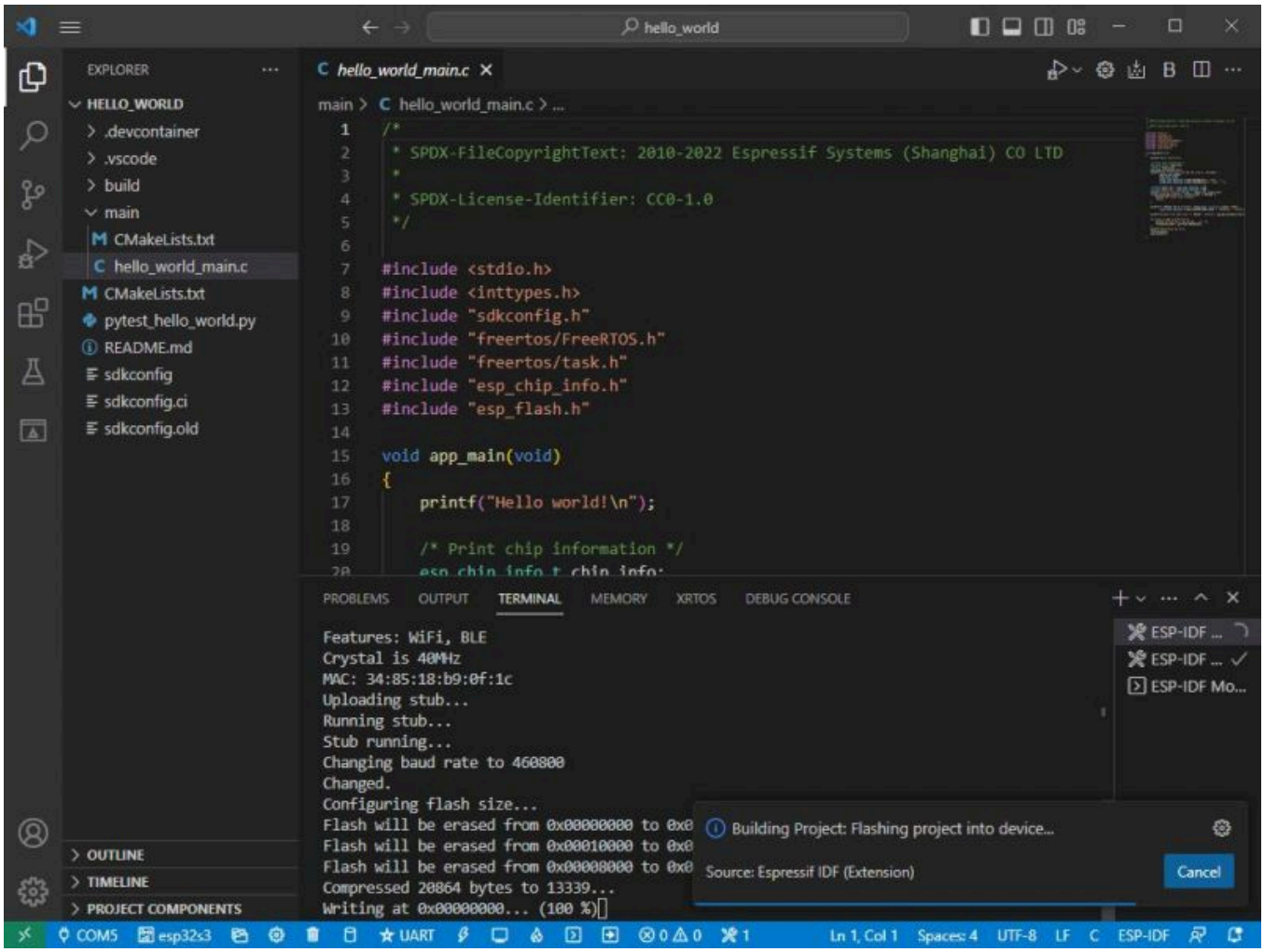
2. It may take a long time to compile, especially for the first time.



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

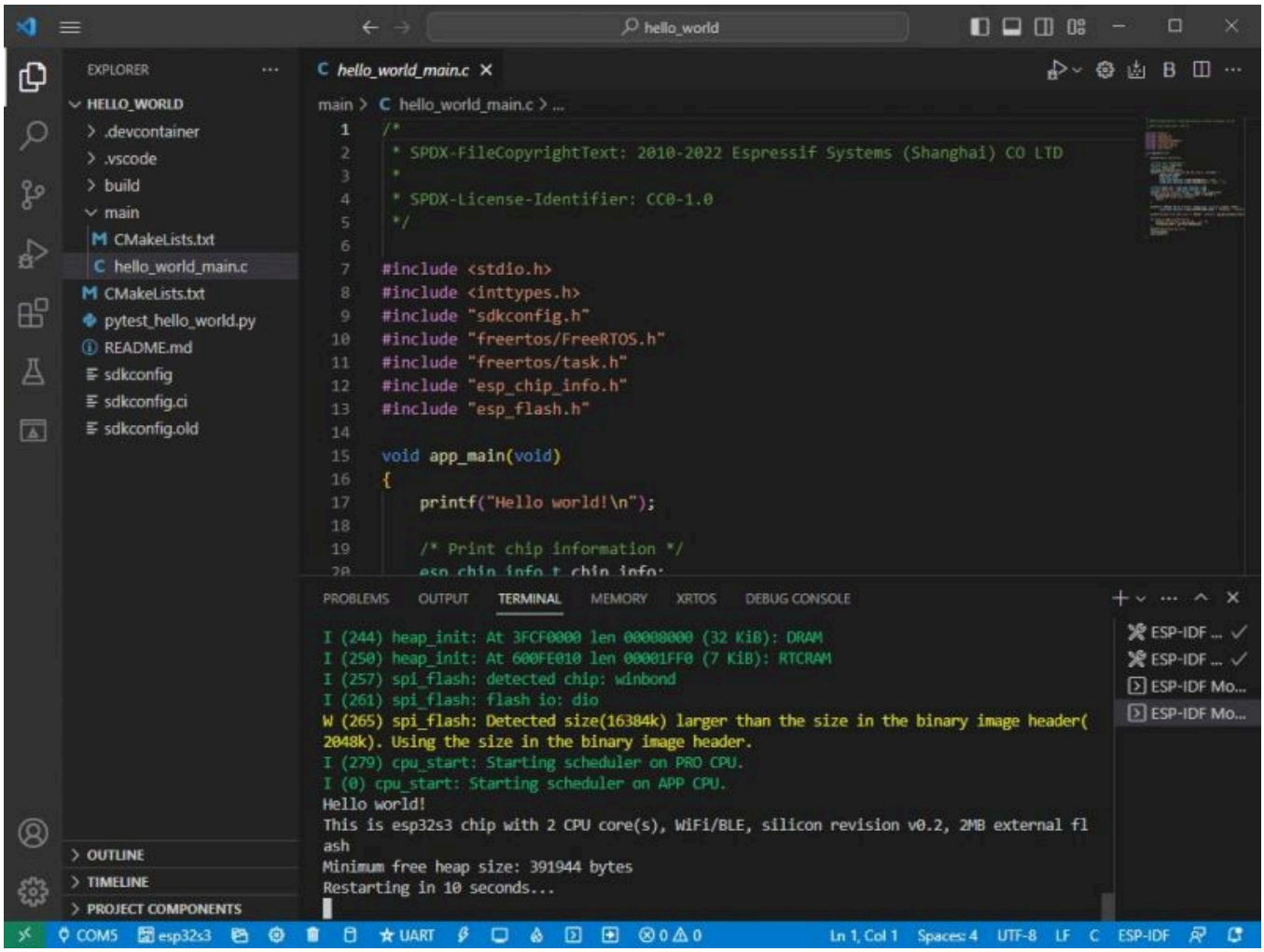
- During this process, ESP-IDF may take up a lot of CPU resources and therefore may cause system lag.

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)

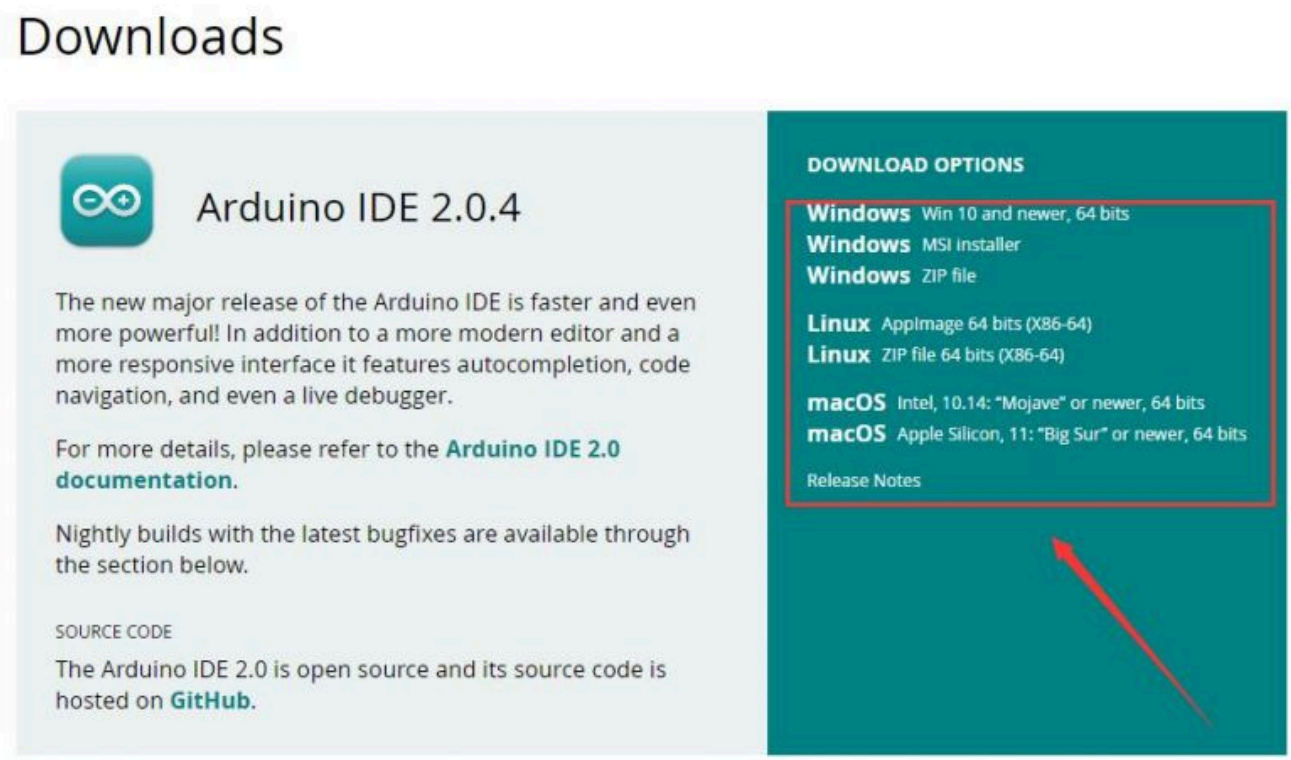
## Arduino

[Collapse]

- **If you do not use arduino-esp32 before, you can refer to this link (https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html).**

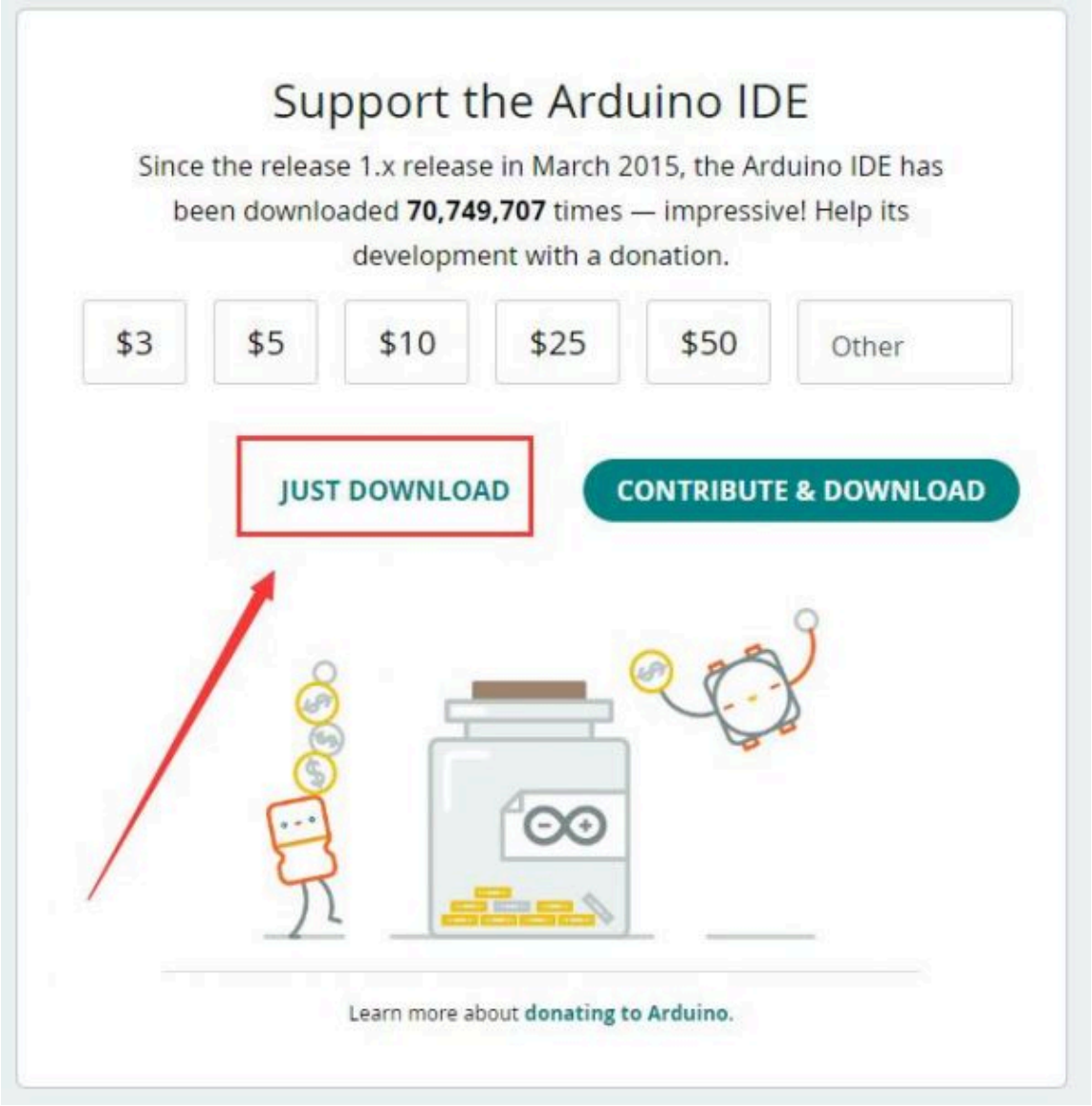
### Install Arduino IDE

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".

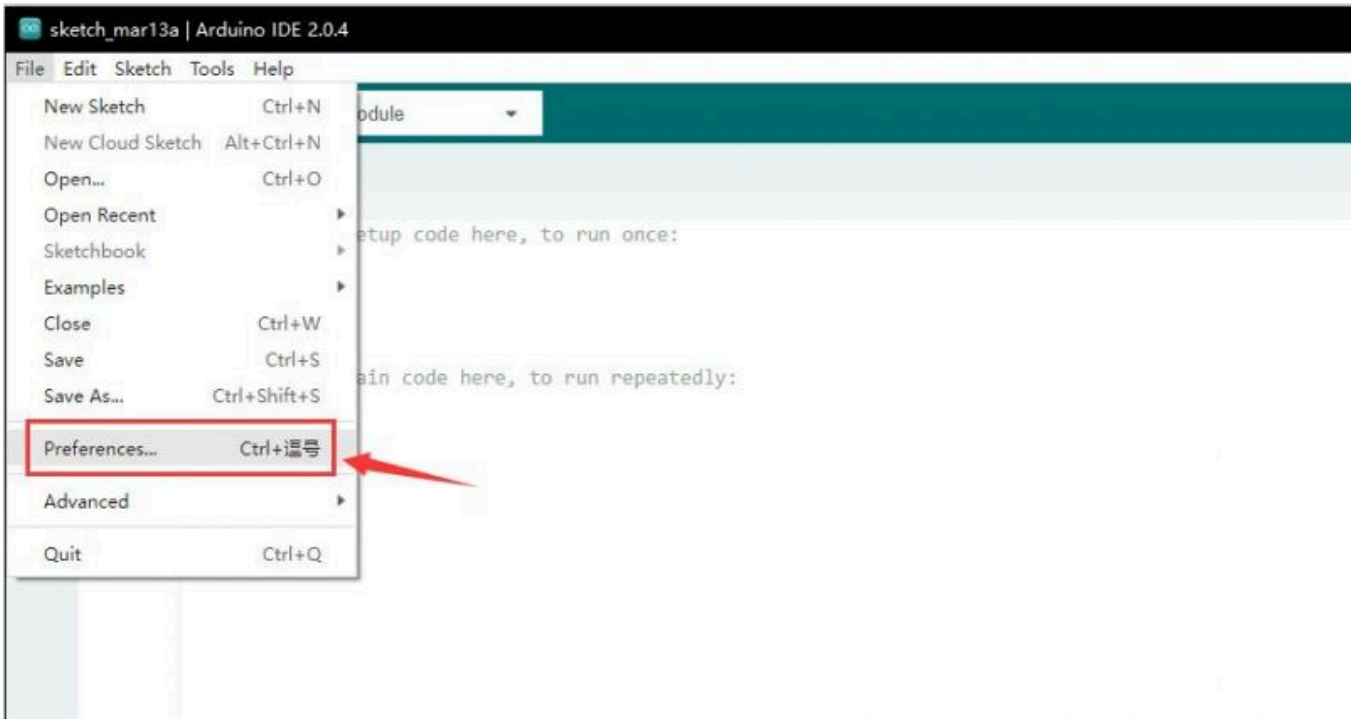



(/wiki/File:ESP32-S3-Pico_36.jpg)

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

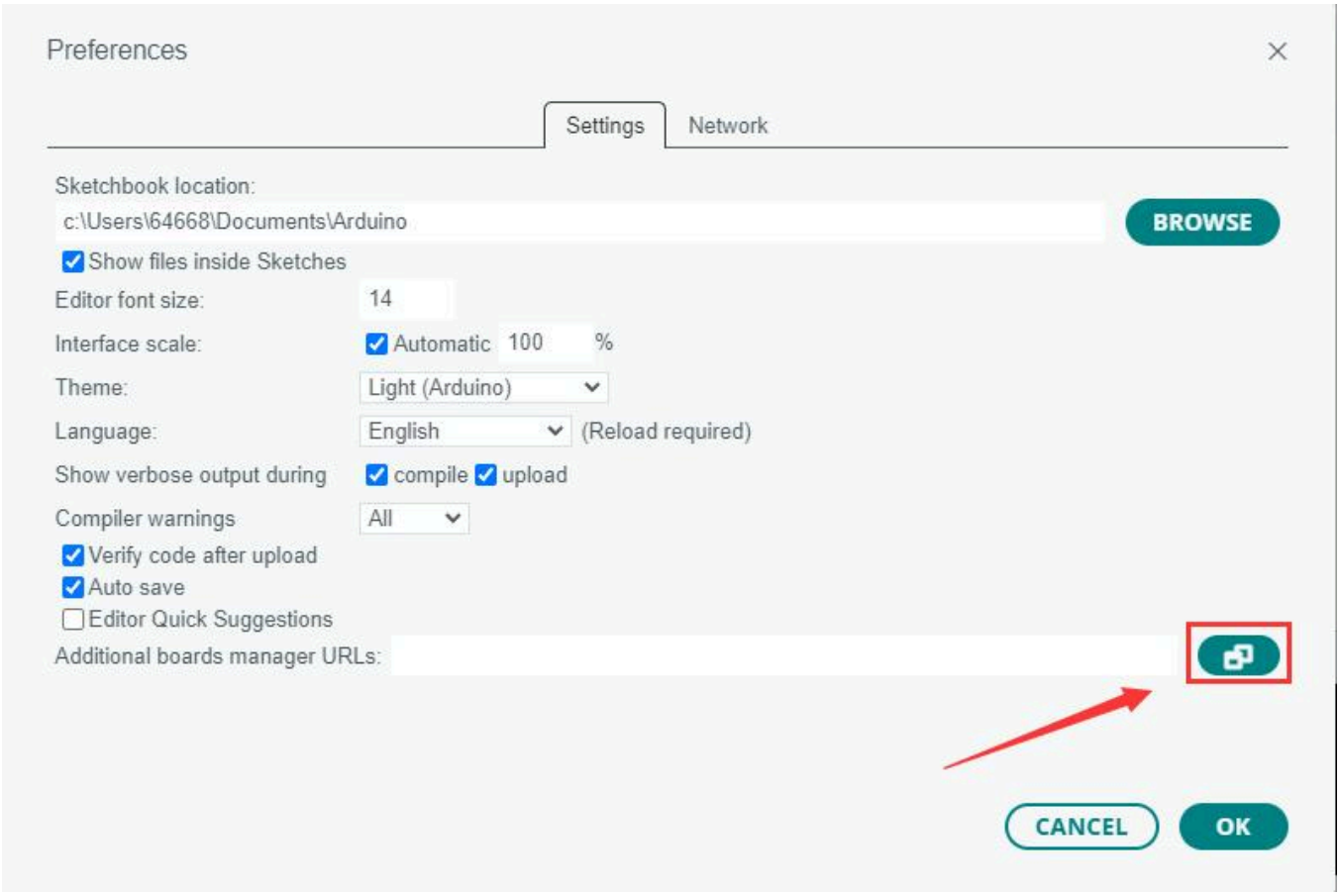
## Install arduino-esp32 Online

1. Open Preferences.




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

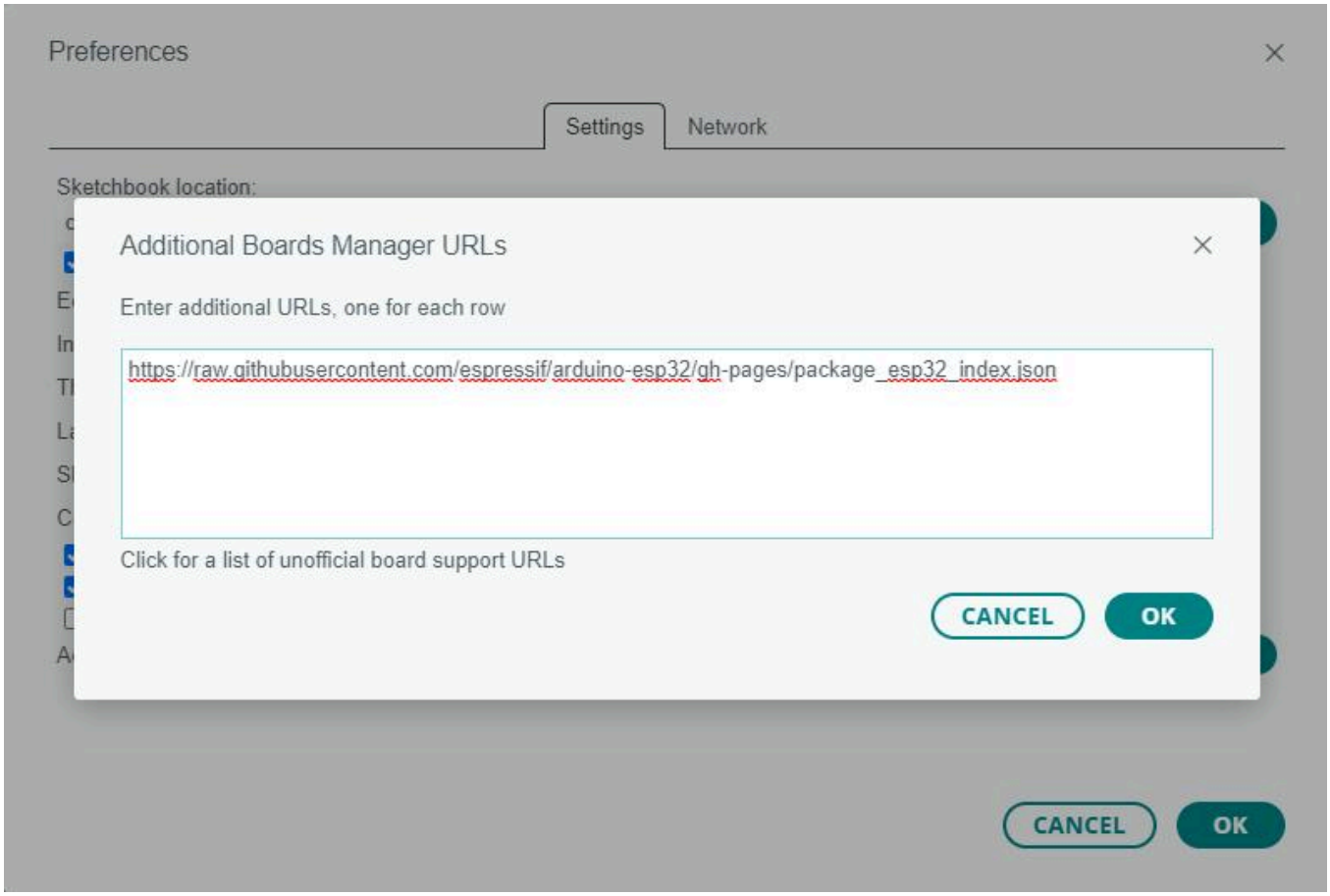
2. Add the corresponding board manager URLs and click the button.




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

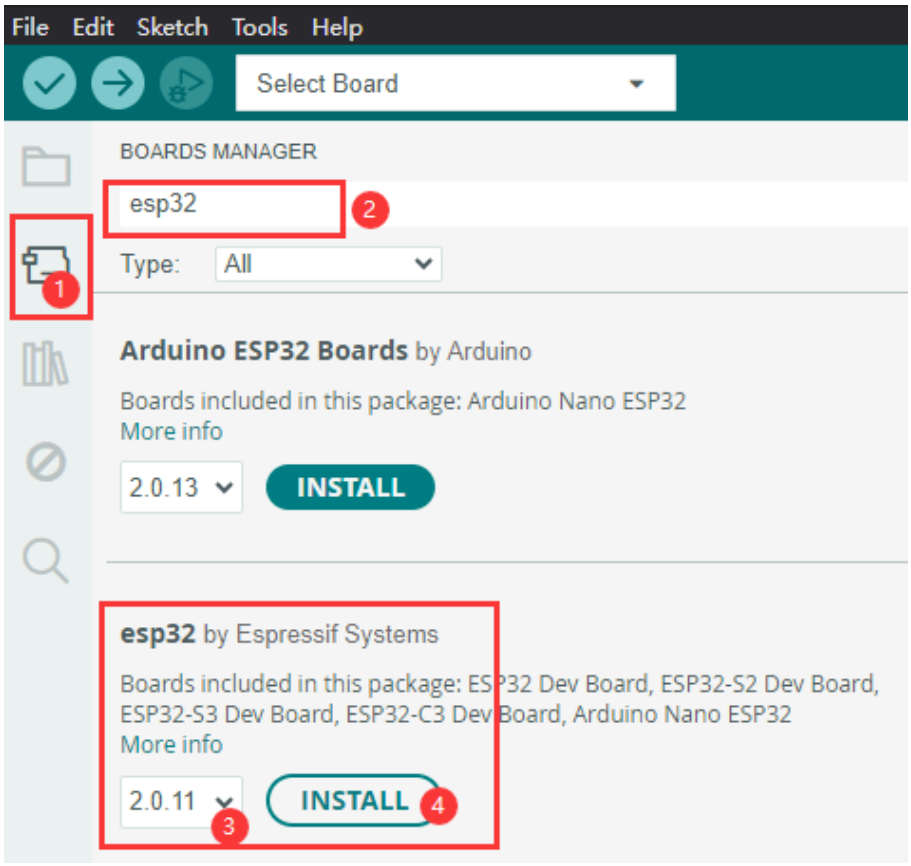
3. Add the following content in the first blank.

```
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.json)
```



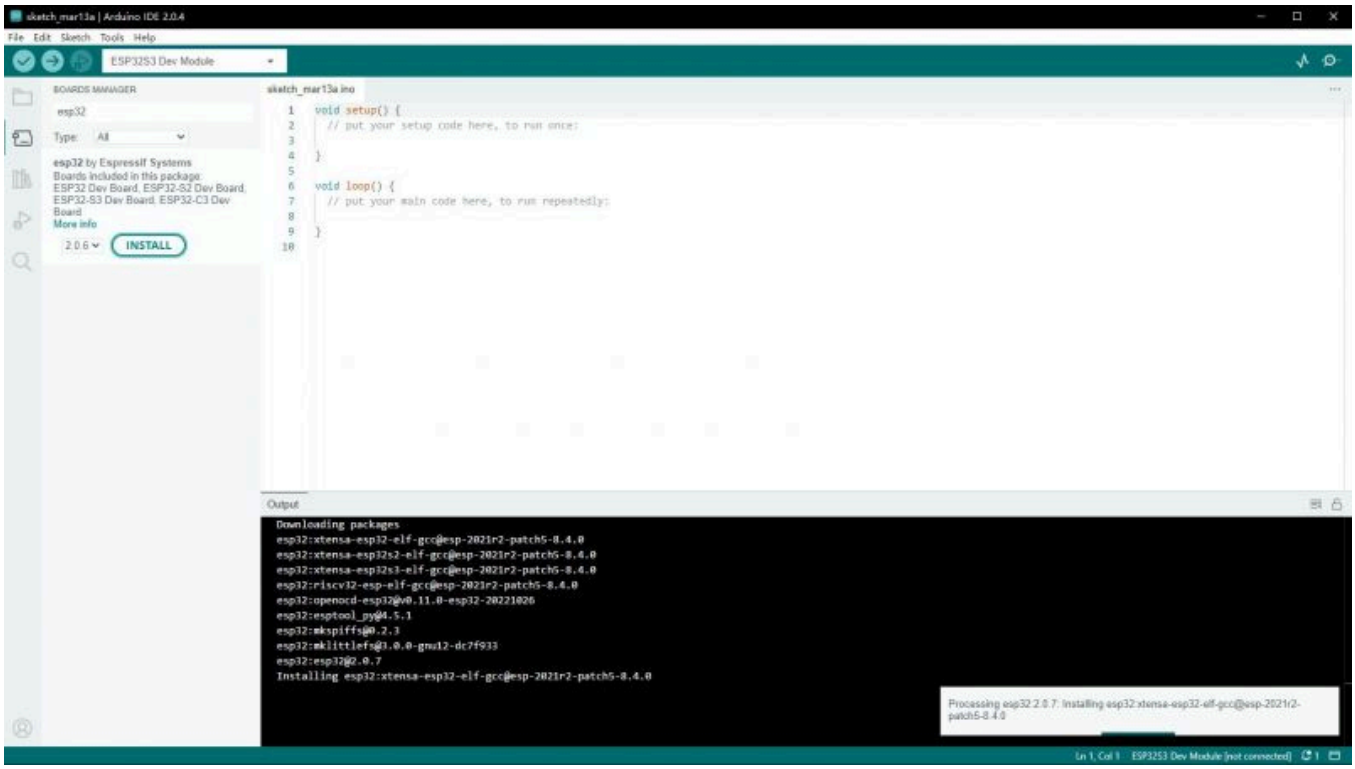

(/wiki/File:ESP32-S3-Pico_39.jpg)

4. Save the setting.
5. Open the board manager, enter ESP32 in the search bar, and select version **2.0.11**.



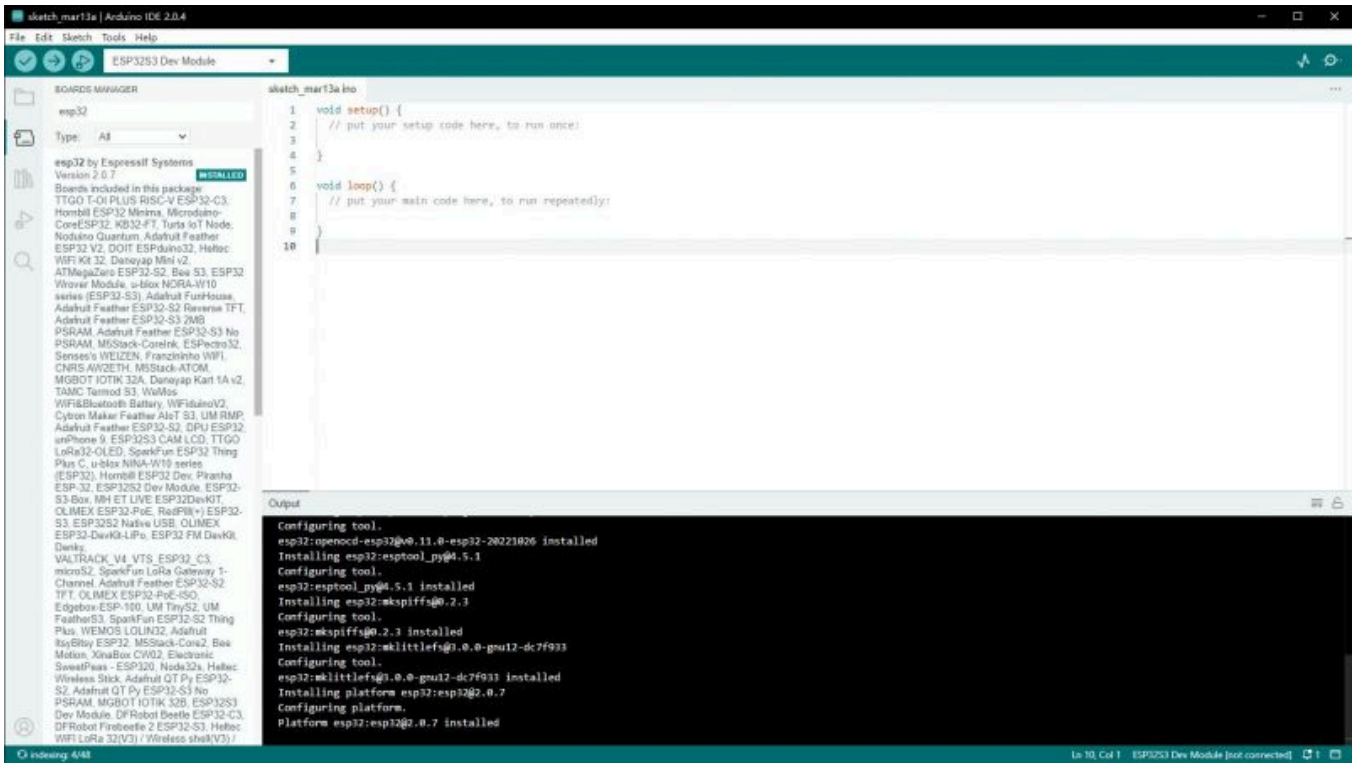
(/wiki/File:ESP32-S3-Pico-Ar-06-240708.png)

6. Wait for downloading.



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

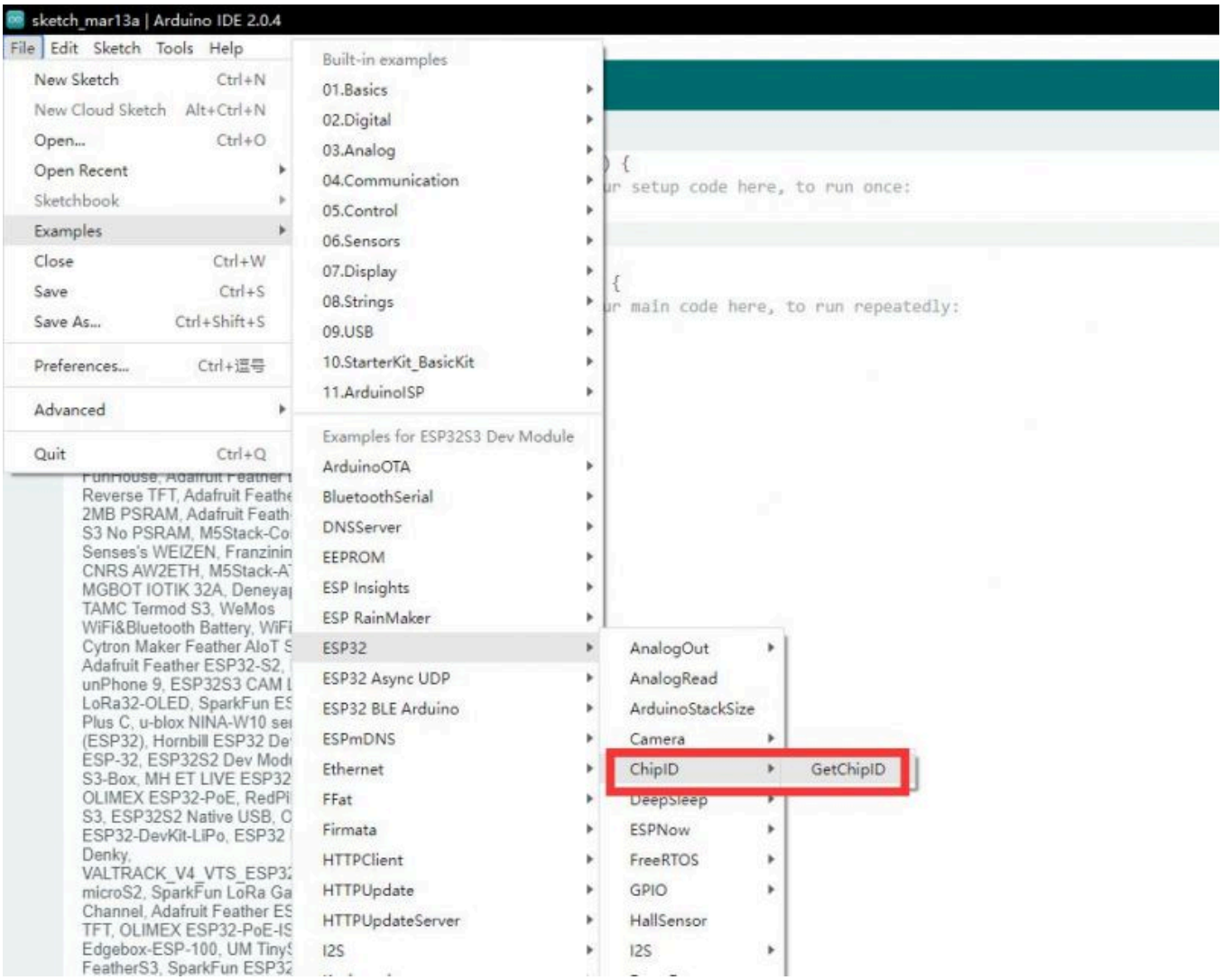
7. arduino-esp32 is downloaded.



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

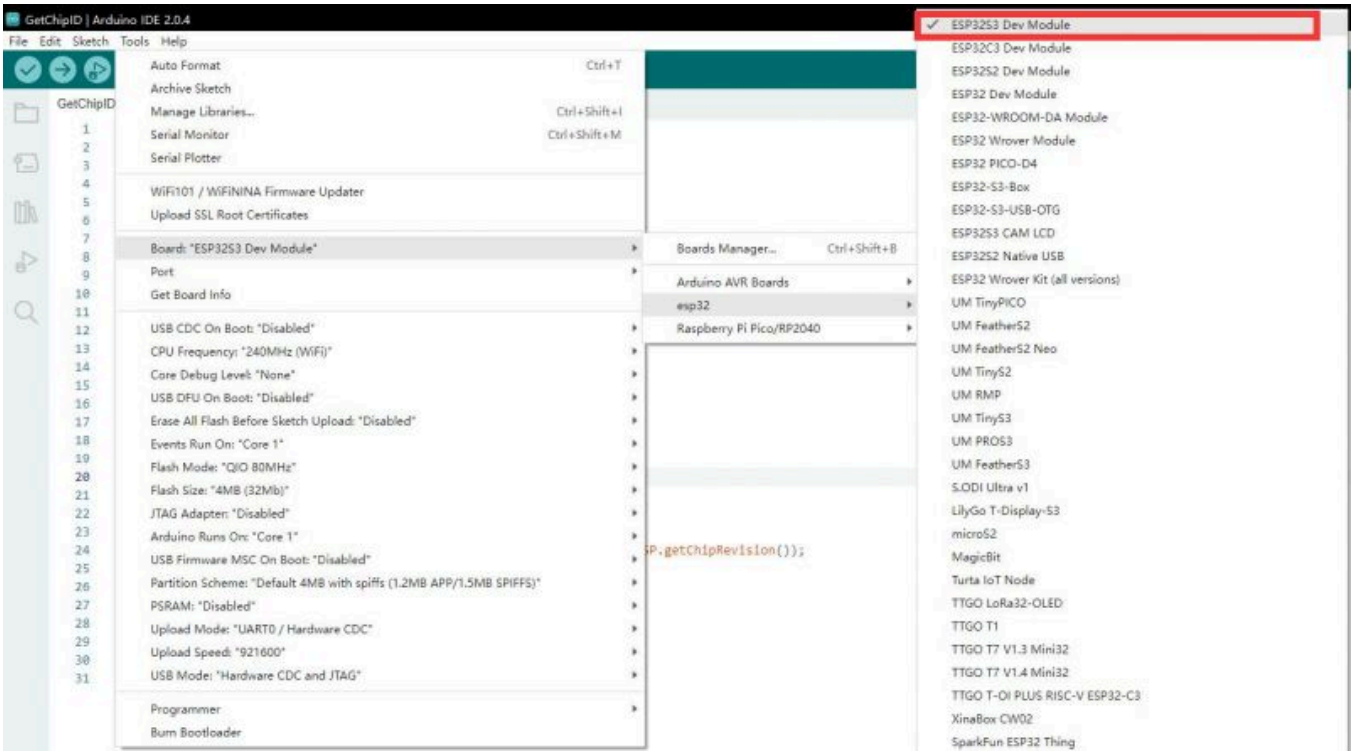
## Use Arduino Demo

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



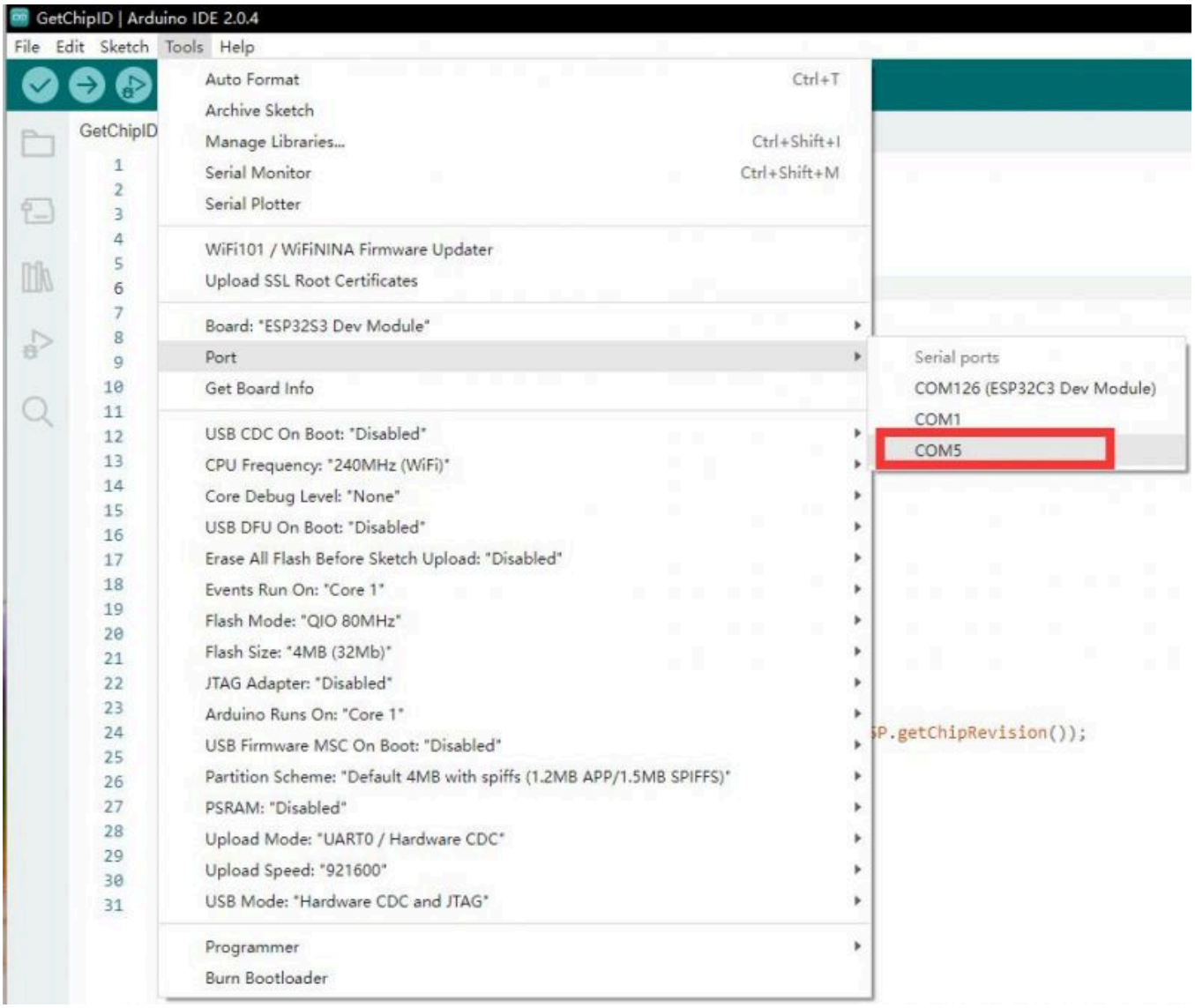

(/wiki/File:ESP32-S3-Pico_43.jpg)

2. Select the board as ESP32S3 Dev Module.



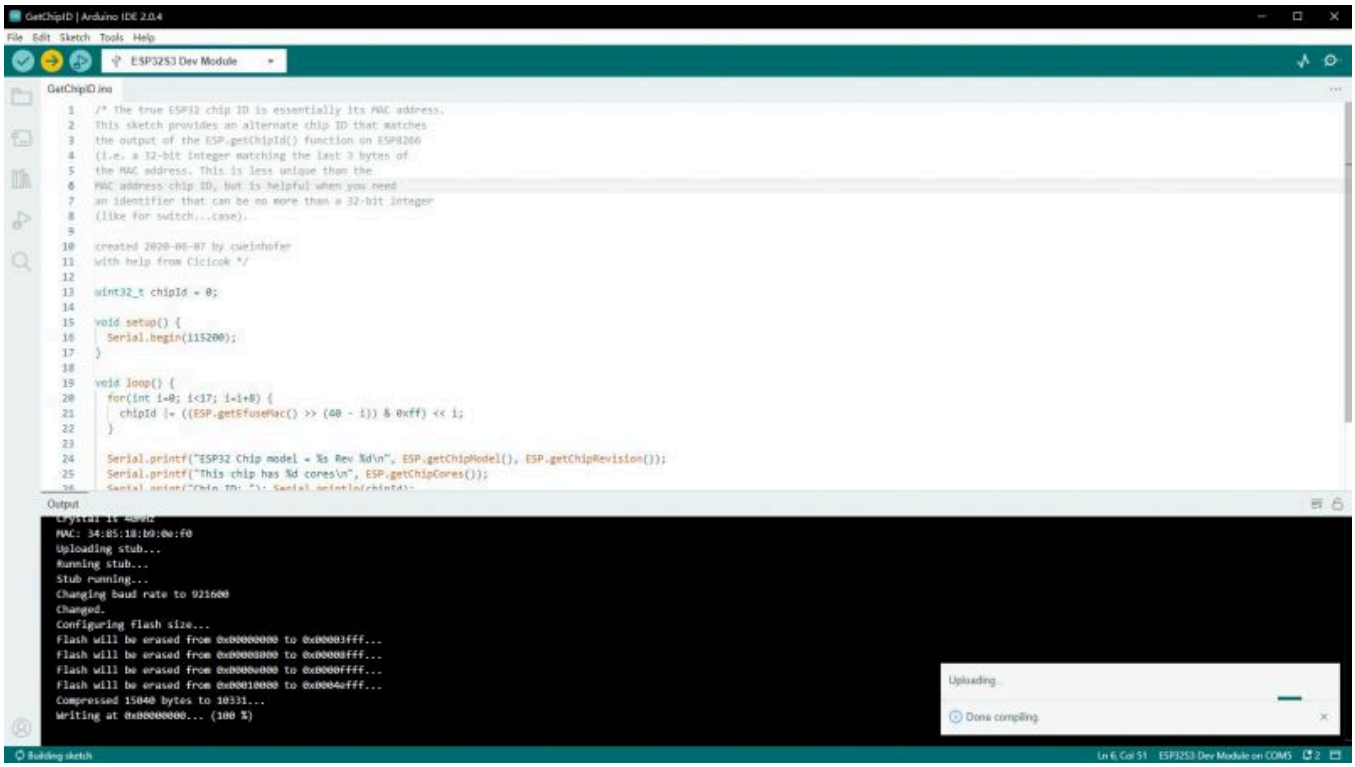

(/wiki/File:ESP32-S3-Pico_44.jpg)

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



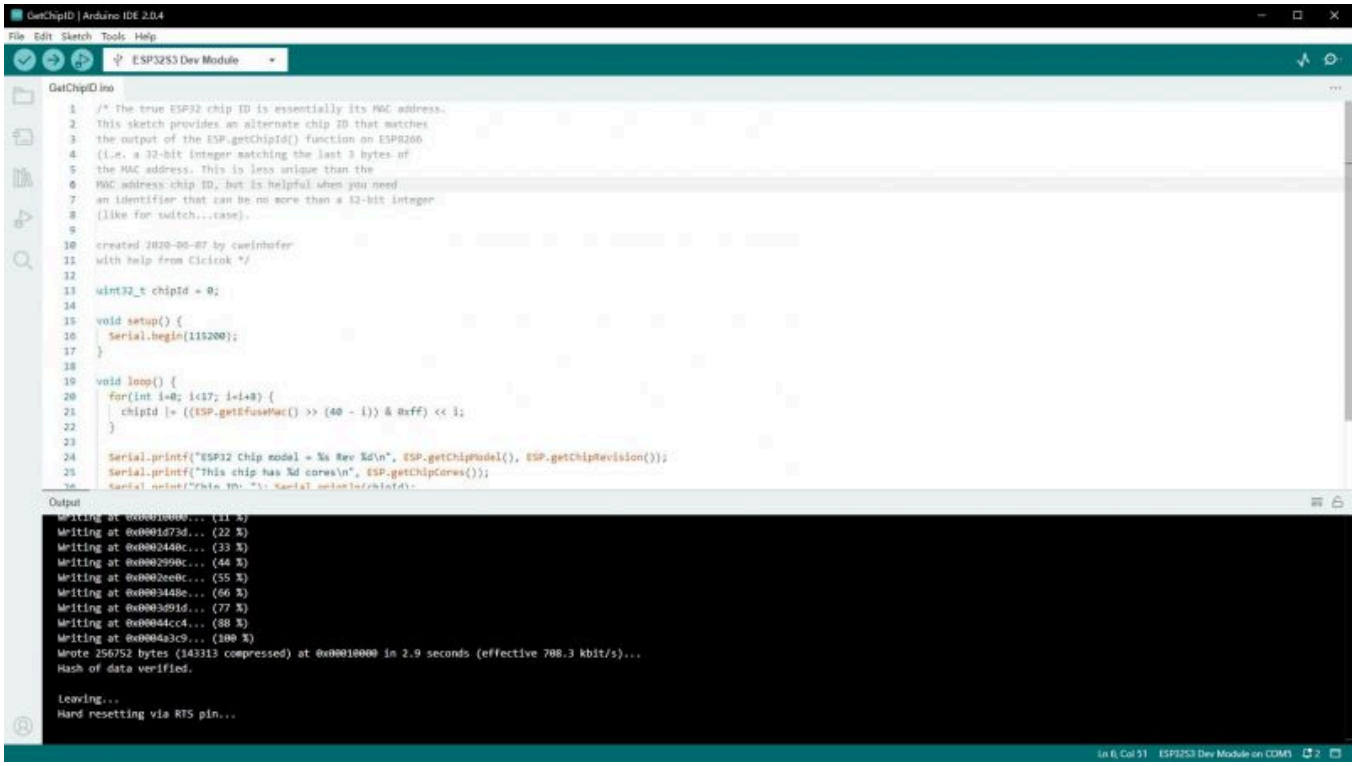

(/wiki/File:ESP32-S3-Pico_45.jpg)

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



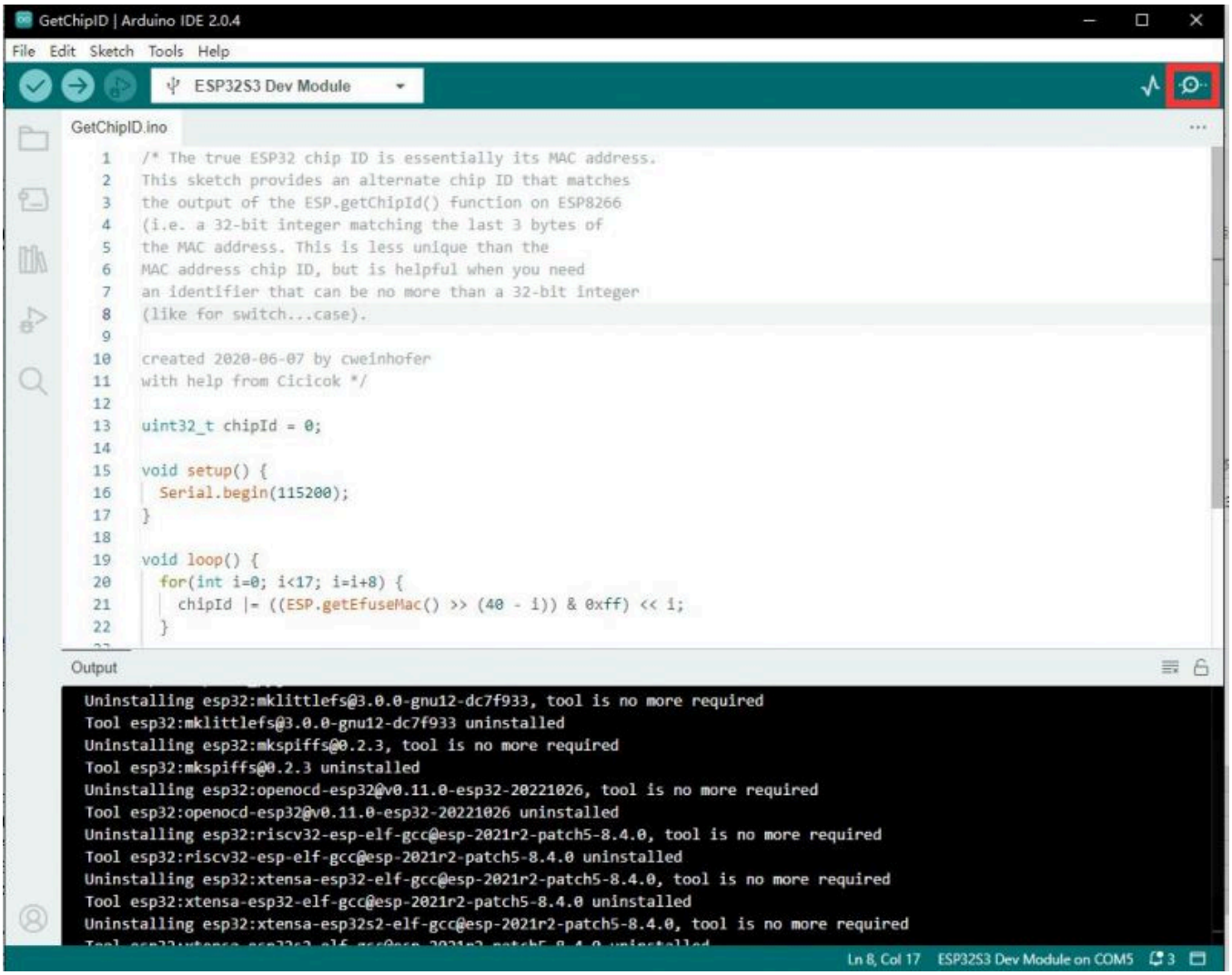

(/wiki/File:ESP32-S3-Pico_47.jpg)

5. Finish.



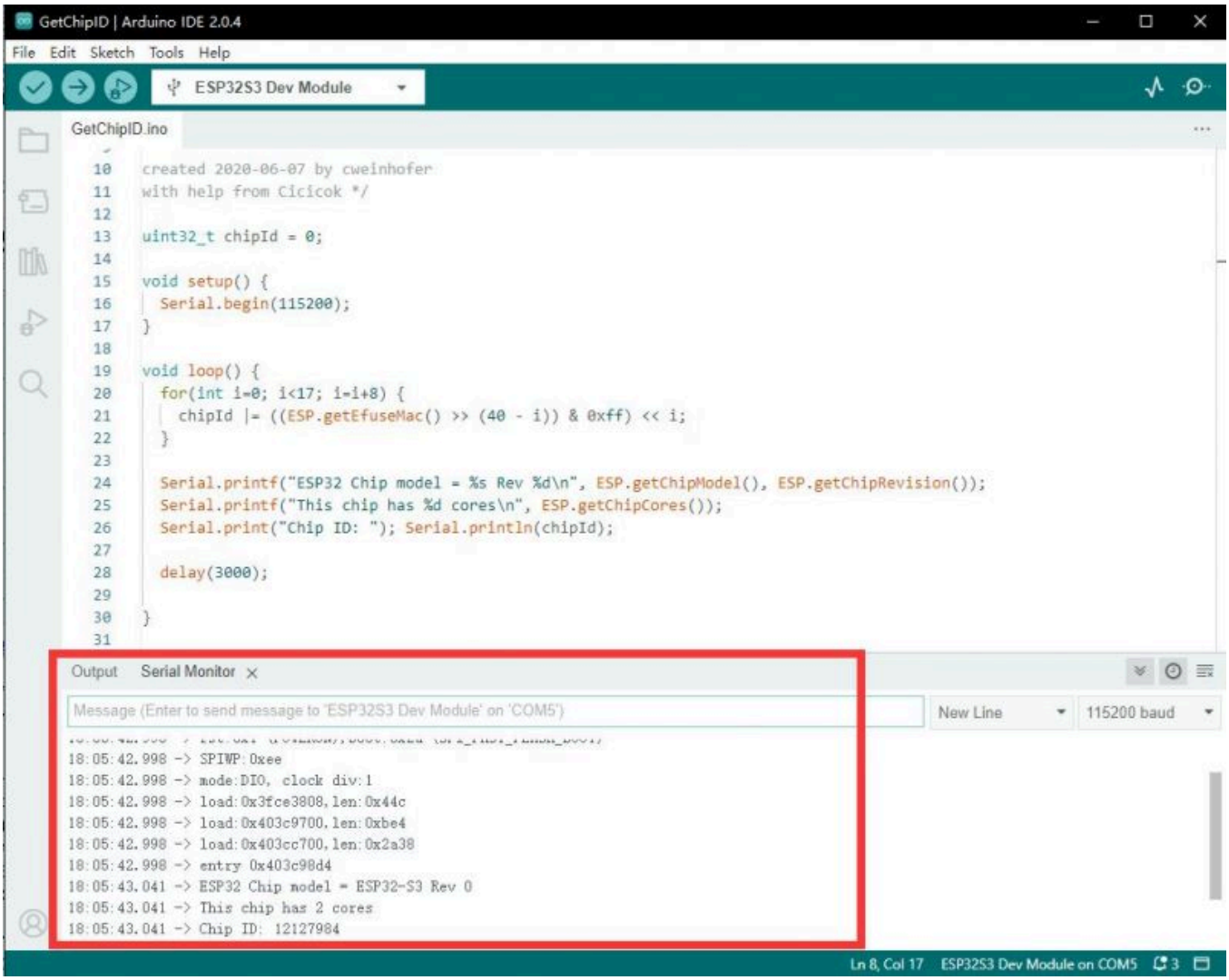

(/wiki/File:ESP32-S3-Pico_48.jpg)

6. Open the Serial Port Monitor.




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

7. See the chip ID of the loop output.



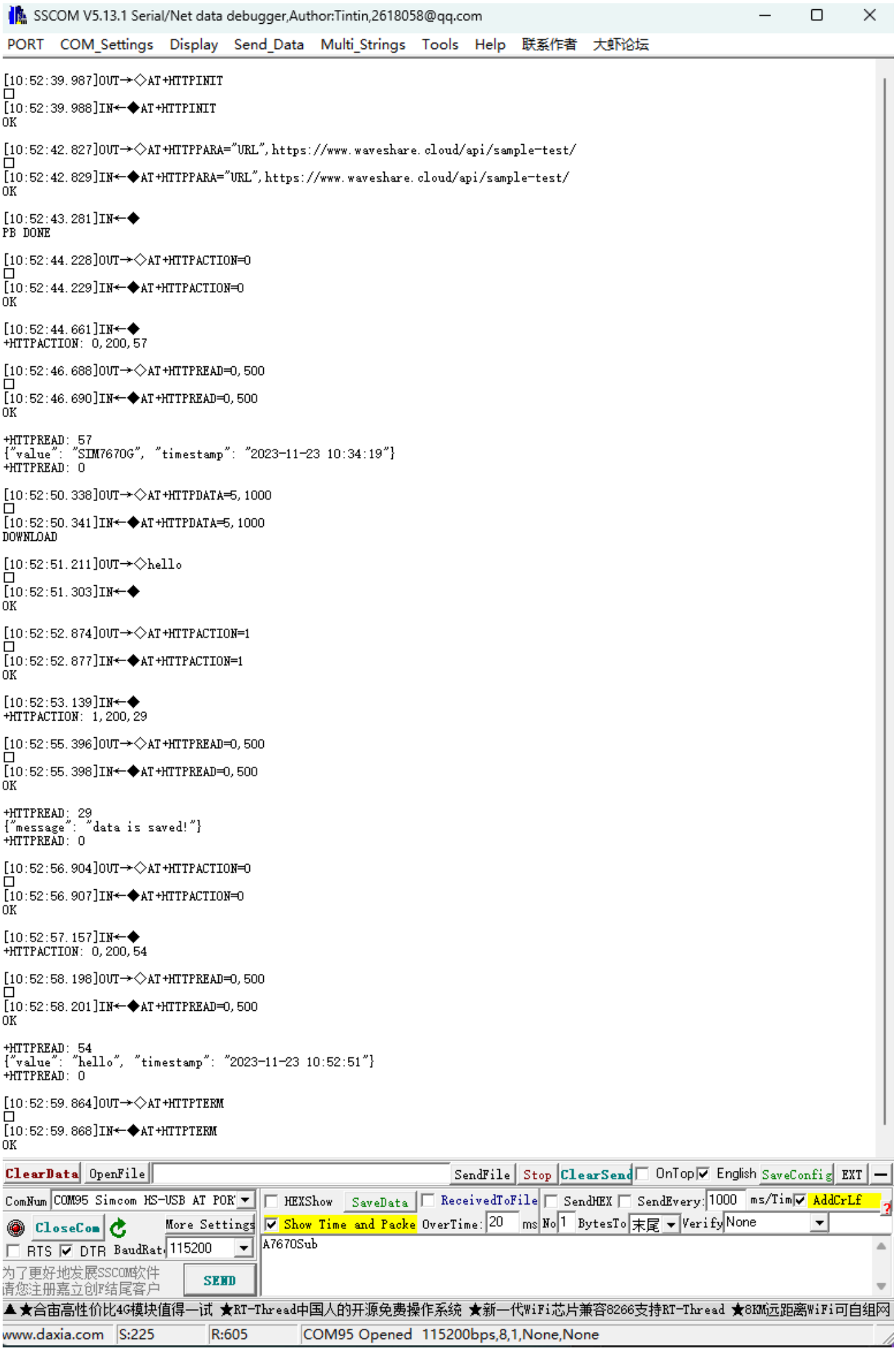

(/wiki/File:ESP32-S3-Pico_50.jpg)

# Cat-1 Module Command Set

## HTTP

| AT Command | Command Description | Return |
|---|---|---|
| AT+HTTPINIT | Open HTTP service | OK |
| AT+HTTPPARA="URL",https://www.waveshare.cloud/api/sample-test/ (https://www.waveshare.cloud/api/sample-test/) | Connect to the remote server | OK |
| AT+HTTPDATA=5,1000 | Input the data | DOWNLOAD <Enter hello OK |

| | | |
|---|---|---|
| AT+HTTPACTION=0 | Open HTTP request, 0:GET; 1:POST; 2:HEAD; 3:DELETE; 4:PUT | OK<br>+HTTPACTION: 0,200,54 |
| AT+HTTPTERM | Close HTTP service | OK |
| AT+HTTPPARA | Set HTTP parameters | OK |
| AT+HTTPHEAD | Read the HTTP response header message | OK |
| AT+HTTPREAD | Read the HTTP response message | OK |


SSCOM V5.13.1 Serial/Net data debugger,Author:Tintin,2618058@qq.com
PORT COM_Settings Display Send_Data Multi_Strings Tools Help 联系作者 大虾论坛
[10:52:39.987]OUT→◇AT+HTTPINIT
□
[10:52:39.988]IN←◆AT+HTTPINIT
OK
[10:52:42.827]OUT→◇AT+HTTPPARA="URL",https://www.waveshare.cloud/api/sample-test/
□
[10:52:42.829]IN←◆AT+HTTPPARA="URL",https://www.waveshare.cloud/api/sample-test/
OK
[10:52:43.281]IN←◆
PB DONE
[10:52:44.228]OUT→◇AT+HTTPACTION=0
□
[10:52:44.229]IN←◆AT+HTTPACTION=0
OK
[10:52:44.661]IN←◆
+HTTPACTION: 0,200,57
[10:52:46.688]OUT→◇AT+HTTPREAD=0,500
□
[10:52:46.690]IN←◆AT+HTTPREAD=0,500
OK
+HTTPREAD: 57
{"value": "SIM7670G", "timestamp": "2023-11-23 10:34:19"}
+HTTPREAD: 0
[10:52:50.338]OUT→◇AT+HTTPDATA=5,1000
□
[10:52:50.341]IN←◆AT+HTTPDATA=5,1000
DOWNLOAD
[10:52:51.211]OUT→◇hello
□
[10:52:51.303]IN←◆
OK
[10:52:52.874]OUT→◇AT+HTTPACTION=1
□
[10:52:52.877]IN←◆AT+HTTPACTION=1
OK
[10:52:53.139]IN←◆
+HTTPACTION: 1,200,29
[10:52:55.396]OUT→◇AT+HTTPREAD=0,500
□
[10:52:55.398]IN←◆AT+HTTPREAD=0,500
OK
+HTTPREAD: 29
{"message": "data is saved!"}
+HTTPREAD: 0
[10:52:56.904]OUT→◇AT+HTTPACTION=0
□
[10:52:56.907]IN←◆AT+HTTPACTION=0
OK
[10:52:57.157]IN←◆
+HTTPACTION: 0,200,54
[10:52:58.198]OUT→◇AT+HTTPREAD=0,500
□
[10:52:58.201]IN←◆AT+HTTPREAD=0,500
OK
+HTTPREAD: 54
{"value": "hello", "timestamp": "2023-11-23 10:52:51"}
+HTTPREAD: 0
[10:52:59.864]OUT→◇AT+HTTPTERM
□
[10:52:59.868]IN←◆AT+HTTPTERM
OK
ClearData OpenFile SendFile Stop ClearSend OnTop English SaveConfig EXT
ComNum COM95 Simcom HS-USB AT POR HEXShow SaveData ReceivedToFile SendHEX SendEvery: 1000 ms/Tim AddCrLf
CloseCom More Settings Show Time and Packe OverTime: 20 ms No 1 BytesTo 末尾 Verify None
RTS DTR BaudRat 115200
A7670Sub
为了更好地发展SSCOM软件
请您注册嘉立创P结尾客户
SEND
▲★合宙高性价比4G模块值得一试 ★RT-Thread中国人的开源免费操作系统 ★新一代WiFi芯片兼容8266支持RT-Thread ★8KM远距离WiFi可自组网
www.daxia.com S:225 R:605 COM95 Opened 115200bps,8,1,None,None

(/wiki/File:ESP32-S3-A-SIM7670X-4G_HAT-HTTP.png)

## MQTT

| AT Command | Command Description | Return |
|---|---|---|
| AT+CMQTTSTART | Open MQTT service | OK |
| AT+CMQTTACCQ=0,"Waveshare-7670X",0 | Apply for MQTT client | OK |
| AT+CMQTTCONNECT=0,"tcp://mqtt.easyiothings.com",20,1 | Send MQTT request, connect to the private MQTT server (MQTTS) | OK |
| AT+CMQTTTOPIC=0,8 | Input the message to publish the topic | >A7670Pub OK |
| AT+CMQTTPAYLOAD=0,9 | Input the message to be published | OK >waveshare |
| AT+CMQTTPUB=0,0,60 | Publish the message | OK +CMQTTPUB: 0,0 |
| AT+CMQTTSUB=0,8,1 | Subscribe to message topic | >A7670Sub OK<br>+CMQTTSUBTOPIC: 0,0<br>[10:03:39.665]Receive←◆<br>+CMQTTRXSTART: 0,8,15<br>+CMQTTRXTOPIC: 0,8<br>A7670Sub<br>+CMQTTRXPAYLOAD: 0,15<br>{"data":"test"}<br>+CMQTTRXEND: 0 |

| | | |
|---|---|---|
| AT+CMQTTSTOP | Stop MQTT service | OK |
| AT+CMQTTREL | Release the client | OK |
| AT+CMQTTUNSUBTOPIC | Unsubscribe the topic | OK |
| AT+CMQTTUNSUB | Release subscription | OK |

SSCOM V5.13.1 Serial/Net data debugger,Author:Tintin,2618058@qq.com

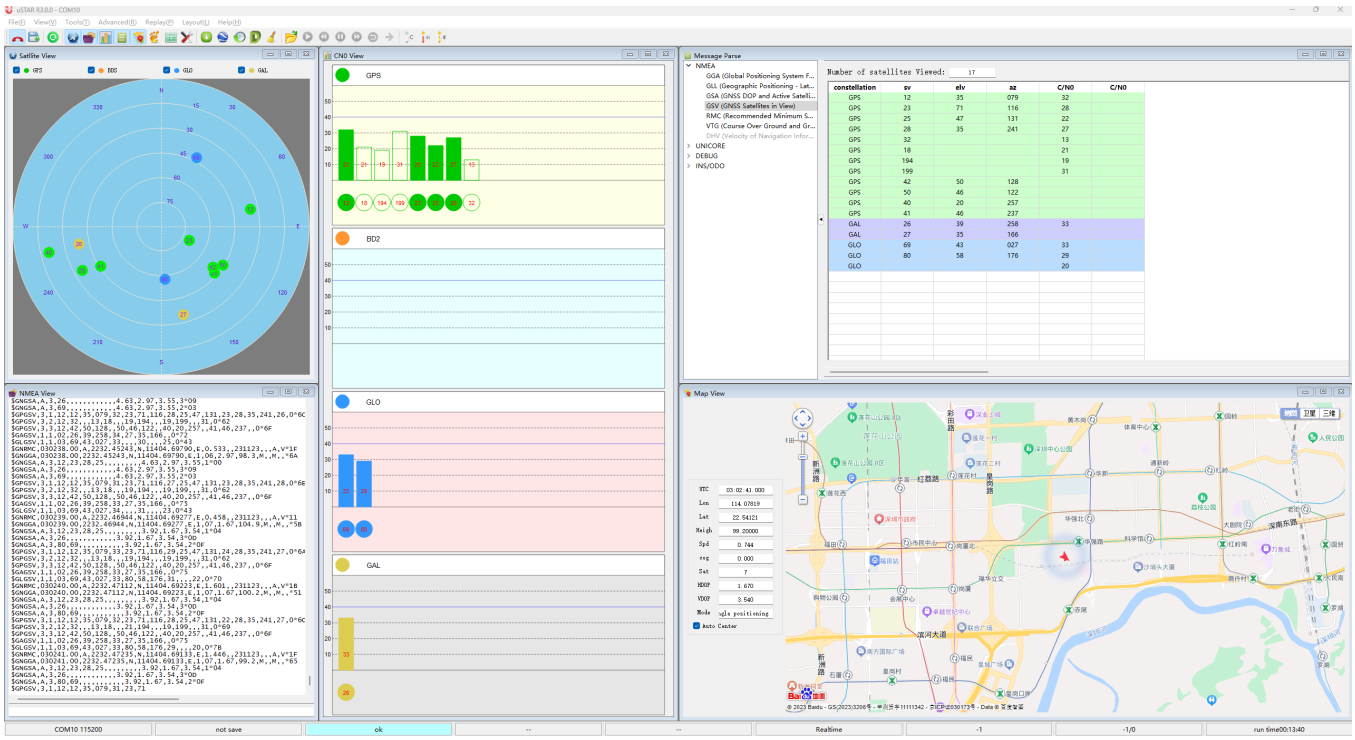
PORT COM_Settings Display Send_Data Multi_Strings Tools Help 联系作者 大虾论坛

```
[10:54:45.978]IN←◆AT+CMQTTACCQ=0,"Waveshare-7670X",0
OK

[10:54:47.051]OUT→◇AT+CMQTTCONNECT=0,"tcp://mqtt.easyiothings.com",20,1
>
[10:54:47.053]IN←◆AT+CMQTTCONNECT=0,"tcp://mqtt.easyiothings.com",20,1
[10:54:47.249]IN←◆
OK

+CMQTTCONNECT: 0,0

[10:54:48.036]OUT→◇AT+CMQTTTOPIC=0,8
>
[10:54:48.039]IN←◆AT+CMQTTTOPIC=0,8
>
[10:54:49.119]OUT→◇A7670Pub
>
[10:54:49.122]IN←◆
OK

[10:54:49.993]OUT→◇AT+CMQTTPAYLOAD=0,9
>
[10:54:49.996]IN←◆AT+CMQTTPAYLOAD=0,9
>
[10:54:50.533]OUT→◇waveshare
>
[10:54:50.536]IN←◆
OK

[10:54:51.334]OUT→◇AT+CMQTTPUB=0,0,60
>
[10:54:51.337]IN←◆AT+CMQTTPUB=0,0,60
OK

+CMQTTPUB: 0,0

[10:54:53.884]OUT→◇AT+CMQTTSUB=0,8,1
>
[10:54:53.887]IN←◆AT+CMQTTSUB=0,8,1
>
[10:54:54.474]OUT→◇A7670Sub
>
[10:54:54.478]IN←◆
OK

[10:54:54.539]IN←◆
+CMQTTSUB: 0,0

+CMQTTRXSTART: 0,8,15
+CMQTTRXTOPIC: 0,8
\7670Sub
+CMQTTRXPAYLOAD: 0,15
{"data":"test"}
+CMQTTRXEND: 0

[10:54:56.450]IN←◆
+CMQTTRXSTART: 0,8,15
+CMQTTRXTOPIC: 0,8
\7670Sub
+CMQTTRXPAYLOAD: 0,15
{"data":"test"}
+CMQTTRXEND: 0

[10:54:59.385]OUT→◇AT+CMQTTUNSUBTOPIC=0,8
>
[10:54:59.389]IN←◆AT+CMQTTUNSUBTOPIC=0,8
>
[10:55:00.135]OUT→◇A7670Sub
>
[10:55:00.139]IN←◆
OK

[10:55:01.102]OUT→◇AT+CMQTTUNSUB=0,1
>
[10:55:01.105]IN←◆AT+CMQTTUNSUB=0,1
OK

[10:55:01.220]IN←◆
+CMQTTUNSUB: 0,0

[10:55:01.855]OUT→◇AT+CMQTTDISC=0,120
>
[10:55:01.858]IN←◆AT+CMQTTDISC=0,120
[10:55:01.939]IN←◆
OK

+CMQTTDISC: 0,0

[10:55:02.428]OUT→◇AT+CMQTTREL=0
>
[10:55:02.437]IN←◆AT+CMQTTREL=0
OK

[10:55:02.994]OUT→◇AT+CMQTTSTOP
>
[10:55:03.008]IN←◆AT+CMQTTSTOP
```

(/wiki/File:ESP32-S3-A-SIM7670X-4G_HAT-MQTT.png)

## GNSS

| | | |
|---|---|---|
| AT+CGNSSPWR=1 | Open GNSS | +CGNSSPWR: READY! |
| AT+CGNSSTST=1 | Open GNSS data output | OK |


(/wiki/File:ESP32-S3-A-SIM7670X-4G_HAT-GNSS.png)

# Demo Explaination

## ESP32-S3 Application

### Camera

This demo is based on the CameraWebServer demo of the ESP32.

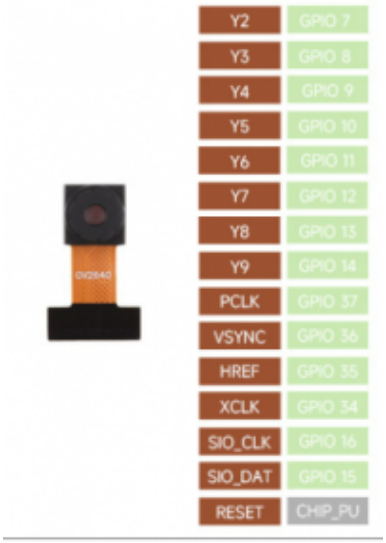
- Firstly, you need to set the WiFi name and password and switch the hardware to ESP32S3 by default.

- Please turn on the CAM of the DIP switch on the back of the development board, and connect to the supported cameras.


(/wiki/File:Esp32-s3-a-sim7670x_240124_01.png)

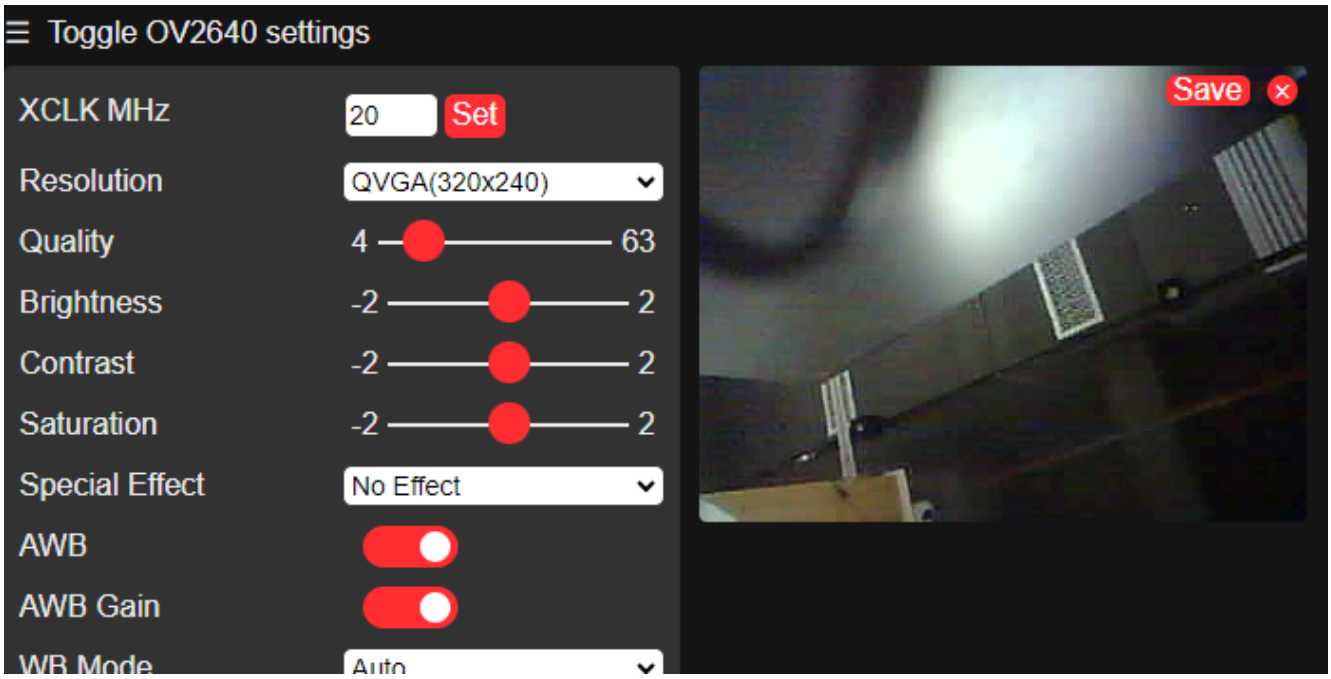
- Please check CameraPins.

```
#define PWDN_GP
IO_NUM      -1
#define RESET_G
PIO_NUM     -1
#define XCLK_GP
IO_NUM      34
#define SIOD_GP
IO_NUM      15
#define SIOC_GP
IO_NUM      16
#define Y9_GPIO
_NUM        14
#define Y8_GPIO
_NUM        13
#define Y7_GPIO
_NUM        12
#define Y6_GPIO
_NUM        11
#define Y5_GPIO
_NUM        10
#define Y4_GPIO
_NUM        9
#define Y3_GPIO
_NUM        8
#define Y2_GPIO
_NUM        7
#define VSYNC_G
PIO_NUM     36
#define HREF_GP
IO_NUM      35
#define PCLK_GP
IO_NUM      37
```




(/wiki/File:Esp32-s3-a-sim7670x_240124_02.png)

- Program the demo Burn the code and open the terminal to access the prompted IP.



(/wiki/File:ESP32-S3-A-SIM7670X-4G02.png)

## TF-Card

- Insert the TF-Card into the TF card slot.
- Pinout definition:

```
const int SDMMC
_CLK = 5;
const int SDMMC
_CMD = 4;
const int SDMMC
_DATA = 6;
const int SD_CD
_PIN = 46;
```


(/wiki/File:Esp32-s3-a-sim7670x_240124_03.jpg)

- Program the demo, and open the terminal to display the file content:

```
11:39:04.960 -> SD_MMC Card Type: SDHC
11:39:04.960 -> SD_MMC Card Size: 59640MB
11:39:04.960 -> Listing directory: /
11:39:04.960 ->   DIR : System Volume Information
11:39:04.960 ->   DIR : bat
11:39:04.960 ->   DIR : CameraWebServer
11:39:04.960 ->   DIR : RGB
11:39:04.960 ->   DIR : SD
11:39:04.960 ->   FILE: ut.h  SIZE: 3953
11:39:04.960 ->   DIR : AP
```

(/wiki/File:ESP32-S3-A-SIM7670X-4G03.png)

## RGB

Onboard a WS2812b RGB LED, and the signal pin is 38.

After programming the sample demo, the LED light is expected to display a gradient color.

(/wiki/File:ESP32-S3-A-SIM7670X-4G-5.gif)

## BAT

This development board utilizes the MAX17048 as the battery charge measurement IC.

- First, confirm the I2C pin.

```
#include <Wire.h>

#define MAX17048_I2C_ADDRESS 0x36

void setup() {
  Wire.begin(3, 2);
  Serial.begin(9600);
}
```

(/wiki/File:ESP32-S3-A-SIM7670X-4G-08.png)

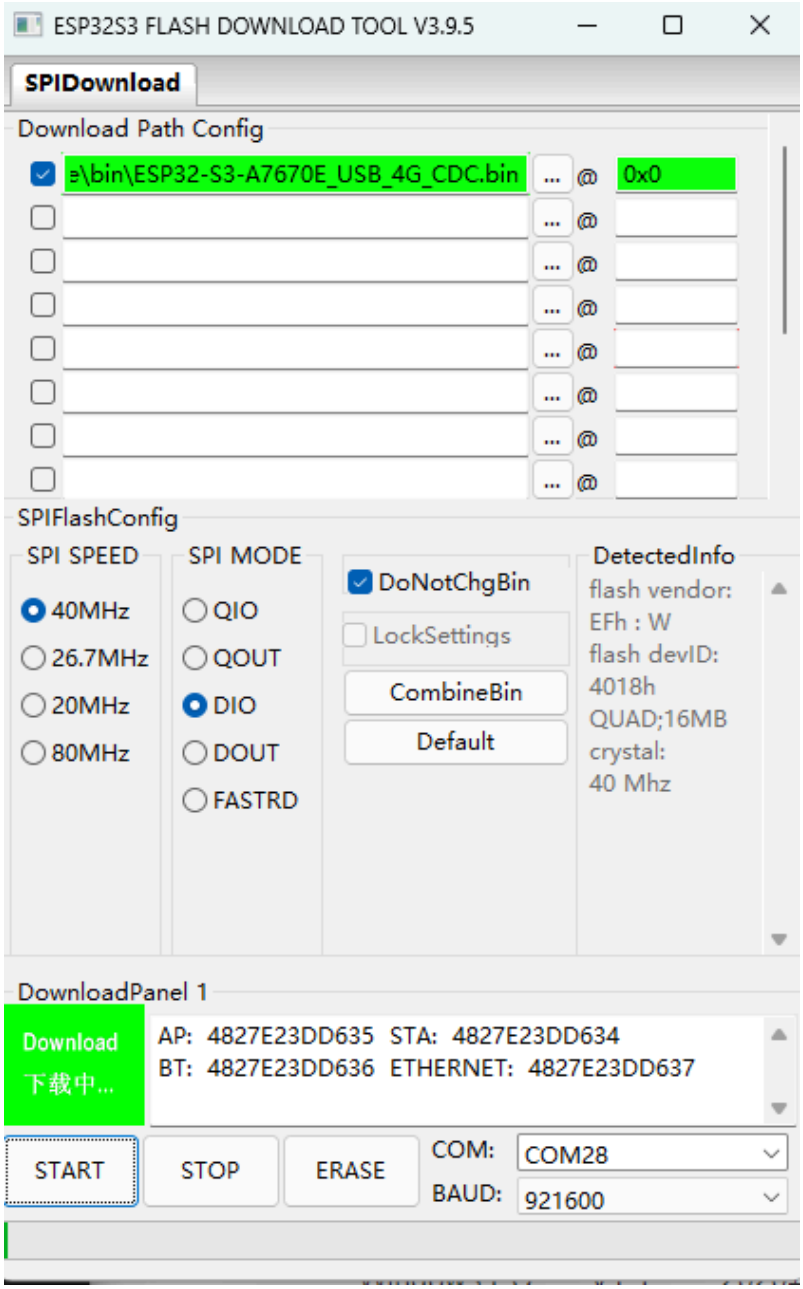
- Program the code and change the threshold.

```
11:55:10.316 -> Battery Level: 83.06%
11:55:11.282 -> Battery Level: 83.06%
11:55:12.322 -> Battery Level: 83.11%
11:55:13.303 -> Battery Level: 83.11%
11:55:14.324 -> Battery Level: 83.11%
11:55:15.289 -> Battery Level: 83.11%
11:55:16.318 -> Battery Level: 83.11%
11:55:17.332 -> Battery Level: 83.11%
11:55:18.327 -> Battery Level: 83.11%
11:55:19.325 -> Battery Level: 83.11%
```

(/wiki/File:ESP32-S3-A-SIM7670X-4G-09.png)

## Portable WIFI Demo

- This demo uses the TinyUSB protocol stack to communicate with a 4G Cat-1 module, using ppp dial-up to provide the network to the ESP32-S3.
- This demo uses the compiled firmware, please download Flash Tools (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/flash_download_tool_3.9.5_0.zip) first.
- Download the firmware:
  - ESP32-S3-A7670E_USB_4G_CDC.bin (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/ESP32-S3-A7670E_USB_4G_CDC.zip)
  - ESP32-S3-SIM7670G_USB_4G_CDC.bin (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/ESP32-S3-SIM7670G_USB_4G_CDC.zip)
- Open the **Flash Tools**, select the development mode, choose the firmware, with the address set to "0x0", as shown in the diagram, and insert the SIM card to download and start the program.

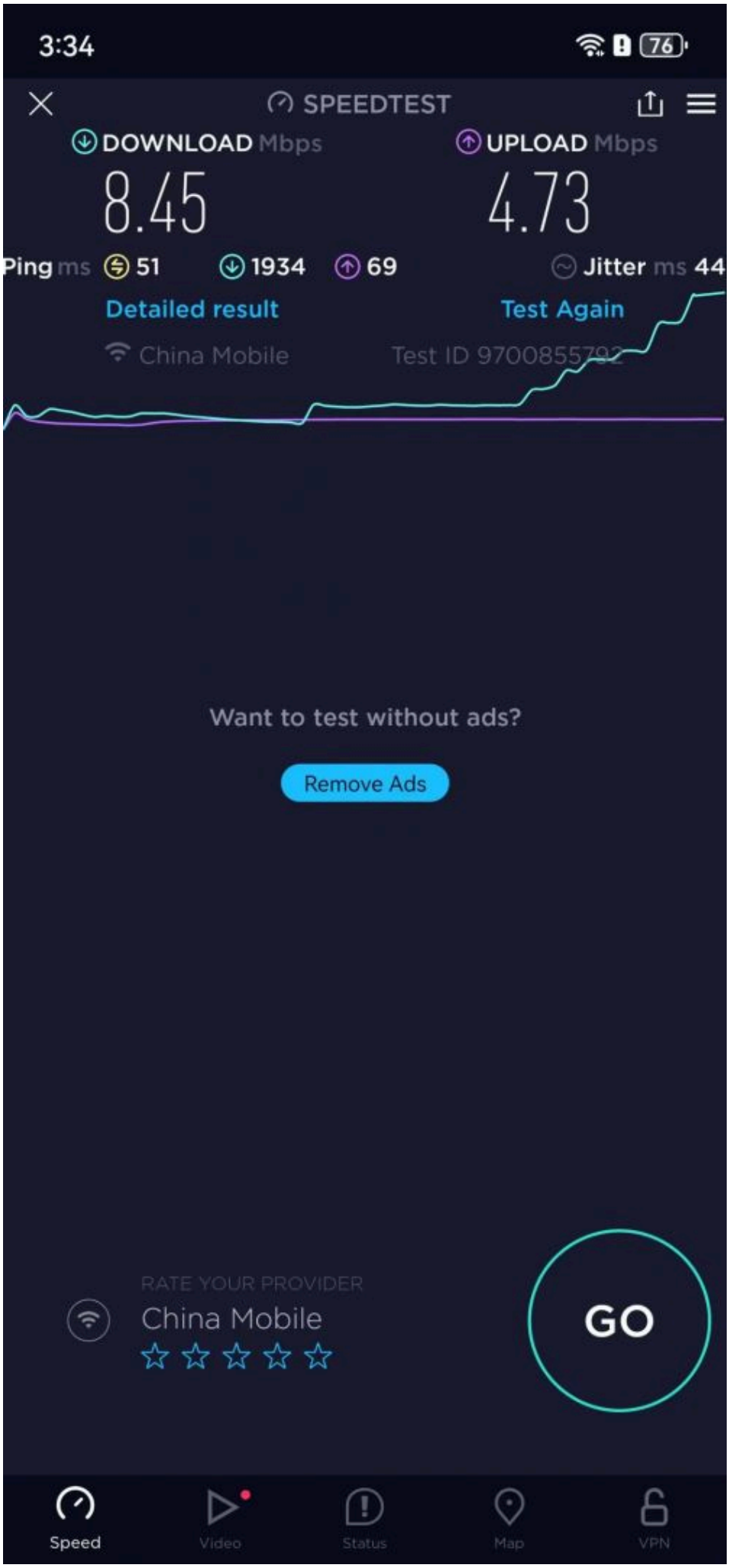


(/wiki/File:ESP32-S3-A-SIM7670X-4G_HAT_wIfI.png)

- The back of the development board toggle switch 4G on, USB off, re-power on the development board, wait for the LED display red, open the phone to connect to WIFI: ESP32-S3-A-SIM7670X-4G-HAT, password: 12345678 to access the Internet.


(/wiki/File:ESP32-S3-A-

SIM7670X-4G_HAT_wIfI-3.jpg)




(/wiki/File:ESP32-S3-A-

SIM7670X-4G_HAT_wIfI04.jpg)

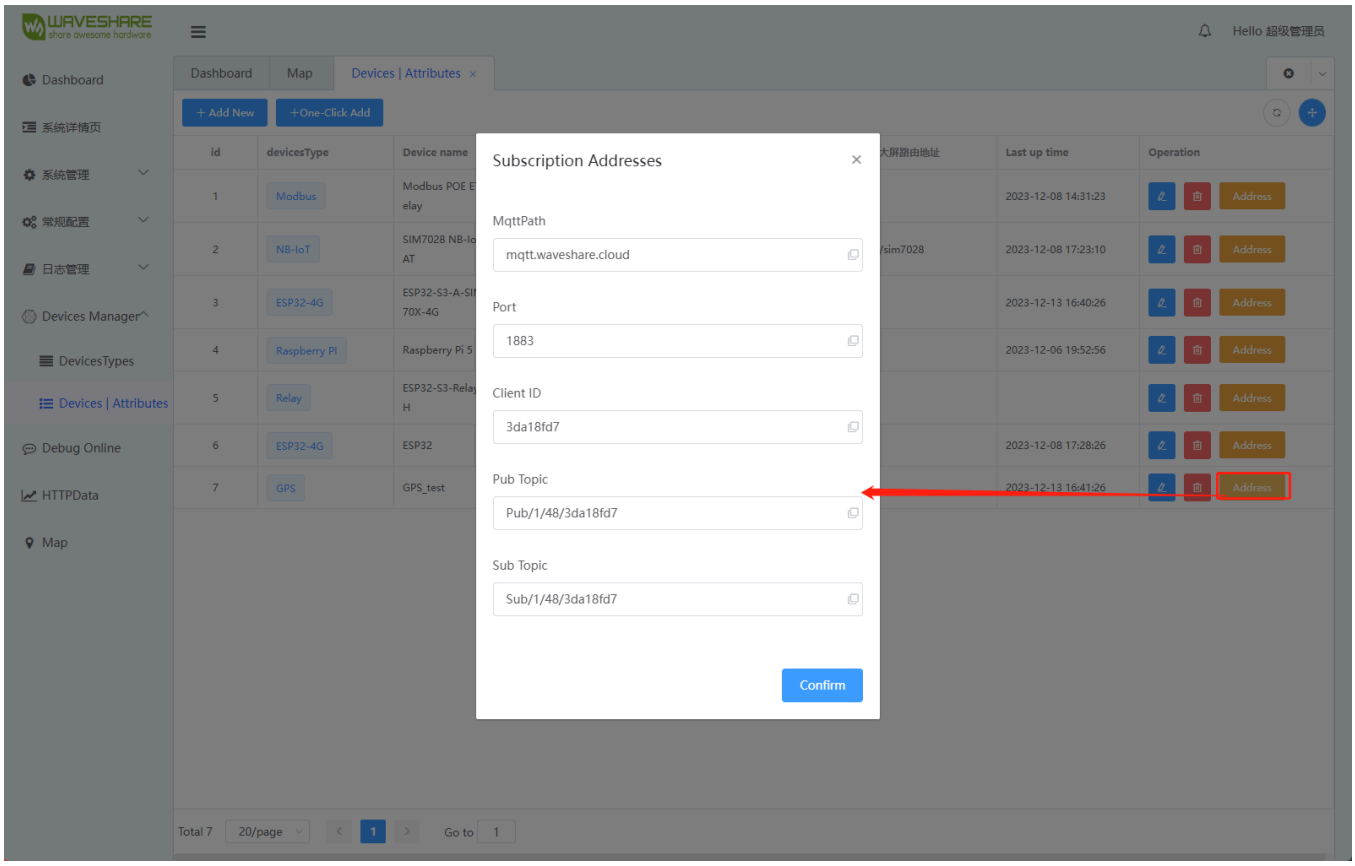
(/wiki/File:Esp32-s3-a7670e-wifi-comaned.png)

- For more details, you can refer to sample demo (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/ESP32-S3-A-SIM7670X_4G.zip).

## Waveshare Cloud Application

Please download the demo (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/ESP32-S3-A-SIM7670X_4G.zip), and open the **GNSS-With-WaveshareCloud** sample demo.
In this application, communication between ESP32-S3 and A7670E-FASE is established using ESP32-S3's software serial port. By sending AT commands, the GNSS (Global Navigation Satellite System) is activated, and NMEA GNSS data is parsed and uploaded to the Waveshare Cloud. The specific location of the development board is then displayed on a web view map page.

Here, we take the map service provided by Waveshare Cloud (https://waveshare.cloud/#/yourlocation) as an example:
1. Through **Device|Attribute** to create any devices, and obtain the MQTT connection data.



(/wiki/File:A7670E_Cat-1-GNSS-HAT_faq.png)

2. Enter the parameters in the **GNSS-With-WaveshareCloud** demo.

```
#define MSG_BUFFER_SIZE (50)
#define STASSID "yourWifi ssid"
#define STAPSK "yourwifi passwd"

#define MAX17048_I2C_ADDRESS 0x36
const char *clientID = ""; // Client ID
char sub[] = ""; // Sub Topic
char pub[] = ""; // Pub Topic
const char *mqtt_server = "mqtt.waveshare.cloud";
```

(/wiki/File:GNSS_with_waveshareclоud.png)

(/wiki/File:Gnss_esp32-s3-a7670E-waveshareCLoud.jpg)

# Resource

## Document

- Schematic (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/ESP32-S3-A-SIM7670X-4G-Sch.pdf)
- MicroPython Development Document (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)

- 3D drawing (https://files.waveshare.com/wiki/ESP32-S3-A-SIM7670X-4G-HAT/3D/esp32-s3-a-sim7670x-4g-pcba.zip)

## Demo

- Demo (https://files.waveshare.com/wiki/ESP32-S3-A-SIM7670X-4G-HAT/Demo/ESP32-S3-A-SIM7670X_4G.zip)

## Tools & Driver

- SSCOM tool (https://files.waveshare.com/wiki/ESP32-S3-A7670E-4G/Sscom_7670X.zip)
- Thonny Python IDE (https://thonny.org/)
- Arduino IDE (https://www.arduino.cc/en/software/)
- GPS debug tool (https://files.waveshare.com/wiki/A7670E-Cat-1-GNSS-HAT/uSTAR3.zip)
- A7600X Windows Driver (https://files.waveshare.com/upload/5/59/A7600X-Windows-Driver.7z)

## Application Note

- ESP32-S3 Datasheet (https://www.espressif.com/en/support/documents/technical-documents?keys=&field_type_tid%5B%5D=842)
- SIM7672X SIM7652X Series FTP(S) Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_FTP(S)_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series HTTP(S) Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_HTTP(S)_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series MQTT(S) Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_MQTT(S)_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series NETWORK Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_NETWORK_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series SSL Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_SSL_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series Sleep Mode Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_Sleep_Mode_Application_Note_V1.00.pdf)

- SIM7672X SIM7652X Series TCPIP Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_TCPIP_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series UART Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_UART_Application_Note_V1.00.pdf)
- SIM7672X SIM7652X Series UIM HOT SWAP Application Note V1.00 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_SIM7652X_Series_UIM_HOT_SWAP_Application_Note_V1.00.pdf)
- SIM7672X Series Hardware Design V1.01 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672X_Series_Hardware_Design_V1.01.pdf)
- SIM7672 Series SPEC Preliminary 230518 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672_Series_SPEC_Preliminary_230518.pdf)
- SIM7672 Series Spec 231205 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM7672_Series_Spec_231205.pdf)
- SIM767XX Series AT Command Manual V1.01 (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM767XX_Series_AT_Command_Manual_V1.01.pdf)
- SIM767XX V1.01 KDL(230309) (https://files.waveshare.com/wiki/ESP32-S3-SIM7670G-4G/SIM767XX_V1.01_KDL(230309).pdf)

# FAQ

### Question: How does ESP32-S3 dial-up internet using the 4G module's USB in the example?

**Answer:**

ESP32-S3 can perform PPP dial-up internet using both serial port and USB. In this example, TinyUSB protocol is utilized, and USB enumeration addresses are used for dial-up internet.

### Question: How can the ESP32-S3 dial-up internet code be modified?

**Answer:**

The code in this example is compiled and flashed using esp-idf. If using Arduino IDE, porting of TinyUSB and handling of PPP packets will be necessary.

### Question:How to modify the ESP32-S3 dial-up code?

**Answer:**

The code in this example is compiled and flashed by esp-idf. To modify ESP32-S3 dial-up code, Arudino IDE needs to be ported tinyUSB and ppp packet processing, etc.

### Question:Can you assist with code modification?

**Answer:**

We do not assist in modifying the code, please do it yourself.

### Question: Can the module itself connect to WaveshareCloud and report latitude and longitude data?

**Answer:**

Currently, the development board connects to the A/SIM7670X 4G module's serial port using software serial port. After enabling GNSS functionality with AT commands, the module outputs satellite signal data through its serial port. AT command can also be sent to the serial port at the moment. Execute the Publish command to publish the required data and filtering of NMEA signal data is required for listening to the returned values from the platform.

### Question:Does the camera included in the package support auto-focus?

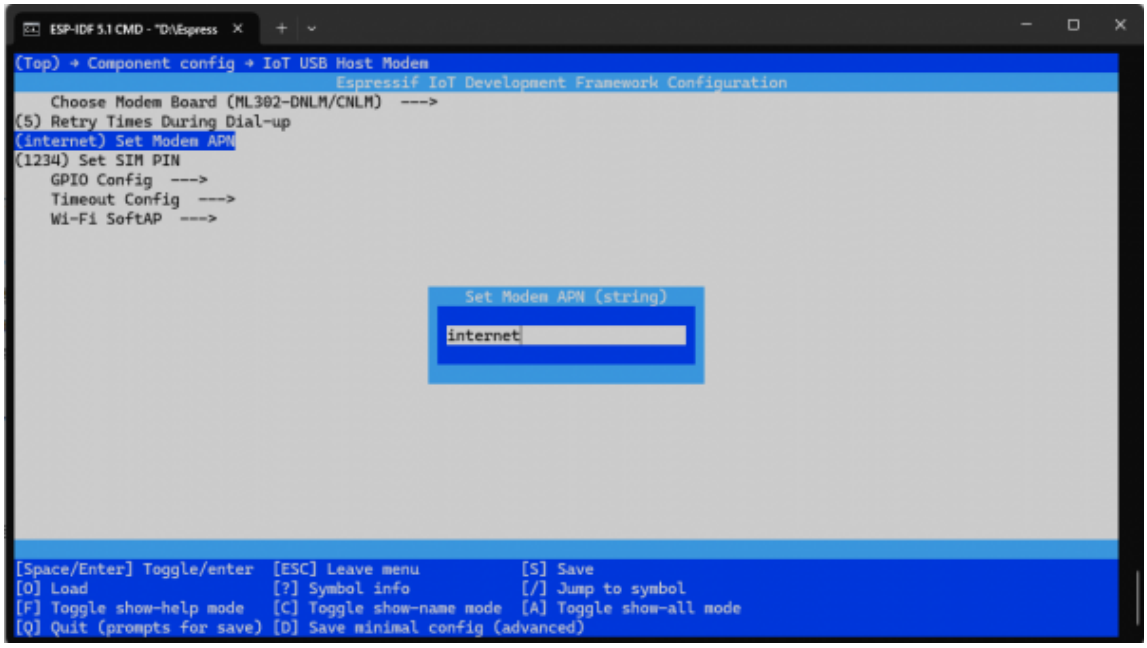
**Answer:**

No.

### Question:For the portable WiFi demo, how to set module APN manually by selecting different operators?

**Answer:**

The default setting is null for the portable WiFi demo, if the module can't recognize the SIM card APN operator automatically, you need to modify the provided source code, refer to README.md in the directory of ESP32-S3-A-SIM7670X-4G-example.
The specific steps are as follows:
1. Refer to the ESP-IDF chapter of the development environment configuration, install the ESP-IDF development environment and vscode programming tools.
2. Use vscode to open the example program and enter menuconfig to set APN manually.



(/wiki/File:ESP32-S3-A7670E.png)
3. Upload the demo to the development board, power off, and reboot the development board.

### Question:Why is it common to encounter situations where header files are missing when using the provided dmeos? Can library files be provided?

**Answer:**

All the example demos provided use libraries downloaded and installed from the Arduino IDE library. These library files are constantly being updated and iterated upon, so often a library might be missing and can be directly downloaded and installed in the Arduino IDE.

### Question:What if I can't receive the GPS signal and don't get the location information?

**Answer:**

Plug in the GPS antenna to the GNSS antenna socket and place the receiver label face down in an open outdoor area (note that it can't be tested in cloudy and rainy weather), and you need to wait for about 1 minute to receive the positioning signal when powering on;

> As GPS indoor search is not stable, please put the module or antenna next to the balcony or window, or directly test it under the outdoor visible sky.


(/wiki/File:Visible-sky.png)

## Question:Does this development board have any demos to remotely transmit the video from the camera over 4g?

**Answer:**

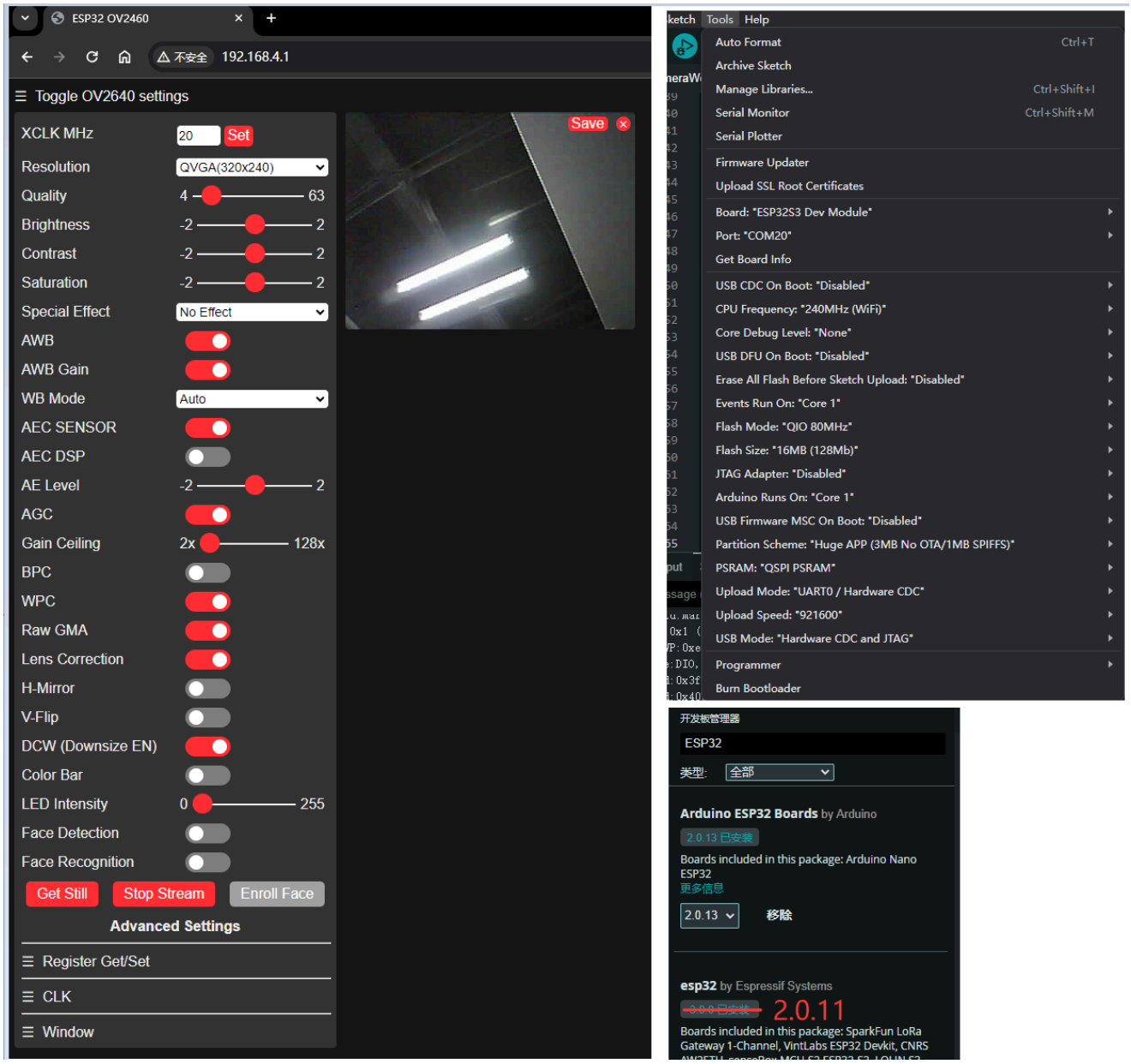
- This is not readily available sample demo, please program yourself and achieve it by secondary development.
- This Module only provides data connection, ESP32 capture camera side is a video stream, this video stream is temporarily displayed in html, that is, directly displayed on the web page!

- If the video stream data through the module can be mapped out to achieve the function of map transmission; the theory is no problem; need to build a public IP server and so on.

## Question:Why is the camera screen black?

**Answer:**

Please configure as shown below:




(/wiki/File:7670-CAM.png)

## Question:Why does the corresponding hotspot not appear after burning the AP sample program?

**Answer:**

To realize the hotspot function:

```
0) X7670X to register to the network and successfully dial up to the Internet
1) Turn on the 4G dip switch on the back of the board, turn off the USB, and re-power o
n the board.
2) Correctly download the corresponding firmware, don't confuse the A7670 and SIM7670 f
irmware, remember to check the box.
```

**Question:Can the ESP32-S3-SIM7670G-4G be turned on with the battery when it is not connected to the power supply?**

**Answer:**

(1) the first time on the battery (that is, after the installation of the battery) need to be connected to the power supply to activate the protection mechanism (this mechanism is to prevent the reverse connection), the battery is fully charged, you can not need to connect the power supply!
(2) In addition, it can also be discharged to activate, typec interface in addition to charging, but also for external equipment power supply, so that the module to external equipment power supply to achieve the purpose of discharging, but also activated!

# 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/)