

ESP32-S3-Pico

来自Waveshare Wiki

跳转至: [导航](#)、[搜索](#)

说明

产品简介

ESP32-S3-Pico是一款低成本、高性能的微控制器开发板,体积小,外设接口丰富。

主芯片采用ESP32-S3R2,其是一款集成 2.4 GHz Wi-Fi 和 Bluetooth 5 (LE) 的 MCU 芯片

接口芯片采用CH343与CH334,使用一个USB-C口就可以体验USB和UART开发的乐趣,不再为切换接口而烦恼

DC-DC芯片采用MP28164,高效率降压-升压芯片,其采用PWM 定频电流控制模式,可使电路稳定性和响应速度达到最优

可选择 ESP-IDF、Arduino、Micropython等开发环境来进行开发,从而可以轻松快速地入门,并将其应用于产品中

产品特性

1. 主芯片采用乐鑫ESP32-S3R2
2. 搭载Xtensa® 32位LX7双核处理器,主频高达240 MHz
3. 内置512 KB SRAM、384KB ROM、2MB的片上PSRAM、板载16MB Flash 存储器
4. 采用USB Type-C 接口,无需纠结正反插,和老旧接口说拜拜
5. 板载沁恒的CH343与CH334,使用一个USB-C口就可以体验USB和UART开发的乐趣
6. 板载 DC-DC 芯片 MP28164,为高效率 DC-DC 降压-升压芯片,最大负载电流可达 2A
7. 支持多种低功耗工作状态,可调节通信距离、数据率和功耗之间的最佳平衡,满足各种应用场景的功耗需求
8. 多达 27 个多功能的 GPIO 引脚引出
9. 邮票孔设计,可直接焊接集成到用户自主设计的底板上



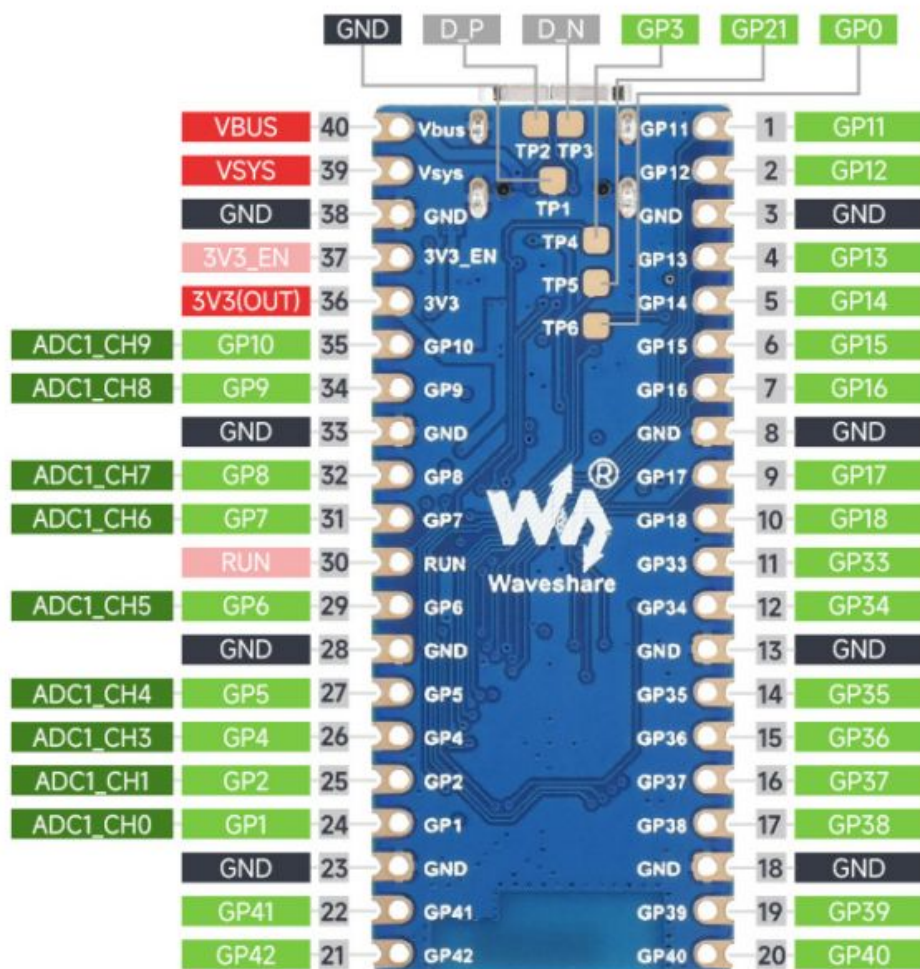
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-01.jpg)

功能简介

| | |
|----|---|
| 主控 | ESP32-S3 |
| 接口 | USB Type-C (/w/index.php?title=%E5%88%86%E7%B1%BB:USB_Type-C%E6%8E%A5%E5%8F%A3&action=edit&redlink=1) |

10. 拥有丰富的外设接口,包括全速 USB OTG、SPI、I2C、UART、ADC、PWM、DVP (8位~16位摄像头接口)、LCD接口 (8位~16位并行RGB、I8080、MOTO6800) 等,让您可以更加灵活地实现各种功能。

引脚分布

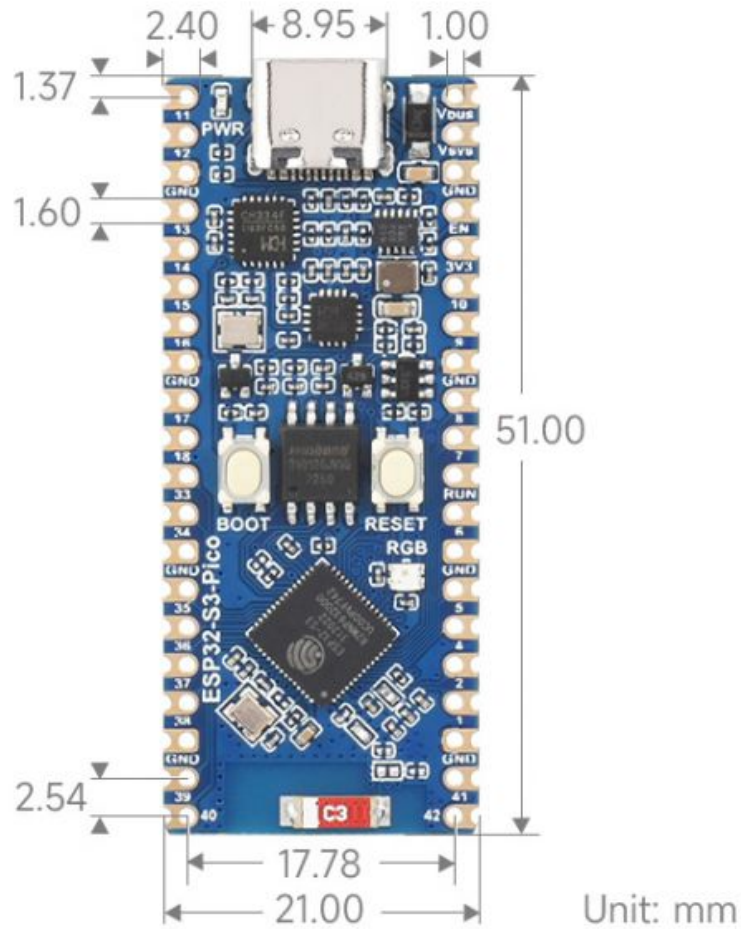


■ Power
 ■ Ground
 ■ GPIO
 ■ ADC
 ■ USB
 ■ System Control

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-details-inter-1.jpg)

- PS、SPI、I2C、UART等数字接口可以通过 GPIO 交换矩阵和 IO MUX 映射到绝大数 GPIO 上,详情见数据手册

产品尺寸



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-details-size.jpg)

开发环境配置

- 以下开发系统默认为Windows

ESP-IDF

- 推荐使用VSC插件进行开发

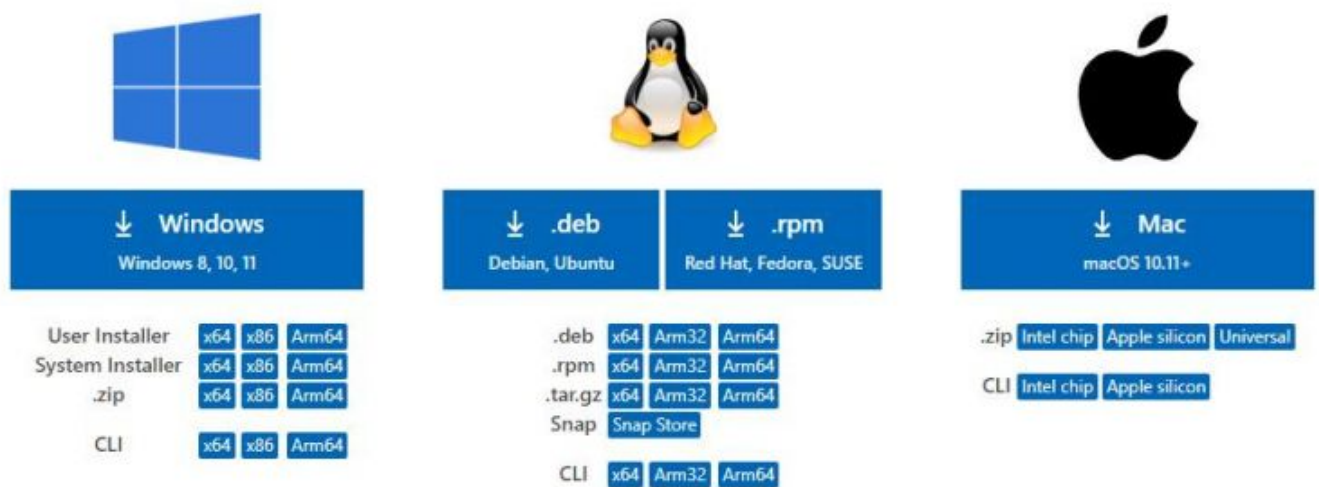
使用VSCode插件开发

安装VSCode

1. 打开VSCode官网的下载页面 (<https://code.visualstudio.com/download>), 选择对应系统和系统位数进行下载

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

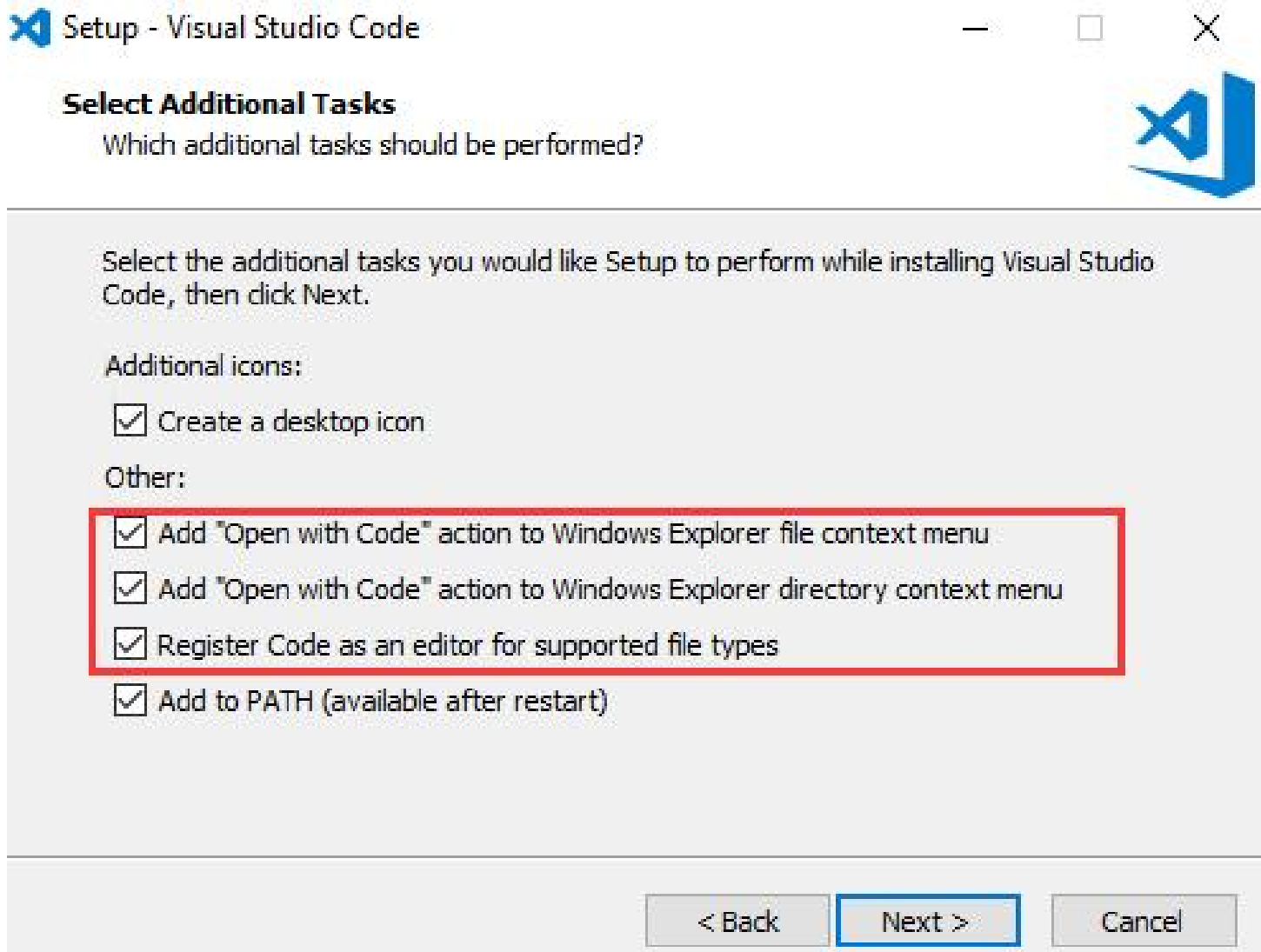


The image shows the download options for Visual Studio Code. It is divided into three main sections: Windows, Linux, and Mac. Each section lists available installers and their supported architectures.

| Platform | Installer Type | Architecture |
|----------|------------------|---------------------------|
| Windows | User Installer | x64, x86, Arm64 |
| | System Installer | x64, x86, Arm64 |
| | .zip | x64, x86, Arm64 |
| | CLI | x64, x86, Arm64 |
| | None | None |
| Linux | .deb | x64, Arm32, Arm64 |
| | .rpm | x64, Arm32, Arm64 |
| | .tar.gz | x64, Arm32, Arm64 |
| | Snap | Snap Store |
| | CLI | x64, Arm32, Arm64 |
| | None | None |
| | Mac | .zip |
| CLI | | Intel chip, Apple silicon |
| None | | None |

(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscod-01.jpg)

2. 运行安装包后，其余均可以默认安装，但这里为了后续的体验建议，建议在此处勾选框中的1、2、3项



The image shows the 'Setup - Visual Studio Code' window. The title bar includes the Visual Studio Code logo and window controls. The main heading is 'Select Additional Tasks' with the question 'Which additional tasks should be performed?'. Below this, there is a section for 'Additional icons' with a checked box for 'Create a desktop icon'. Under 'Other:', there are four checkboxes, the first three of which are highlighted with a red box: 'Add "Open with Code" action to Windows Explorer file context menu', 'Add "Open with Code" action to Windows Explorer directory context menu', and 'Register Code as an editor for supported file types'. The fourth checkbox is 'Add to PATH (available after restart)'. At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

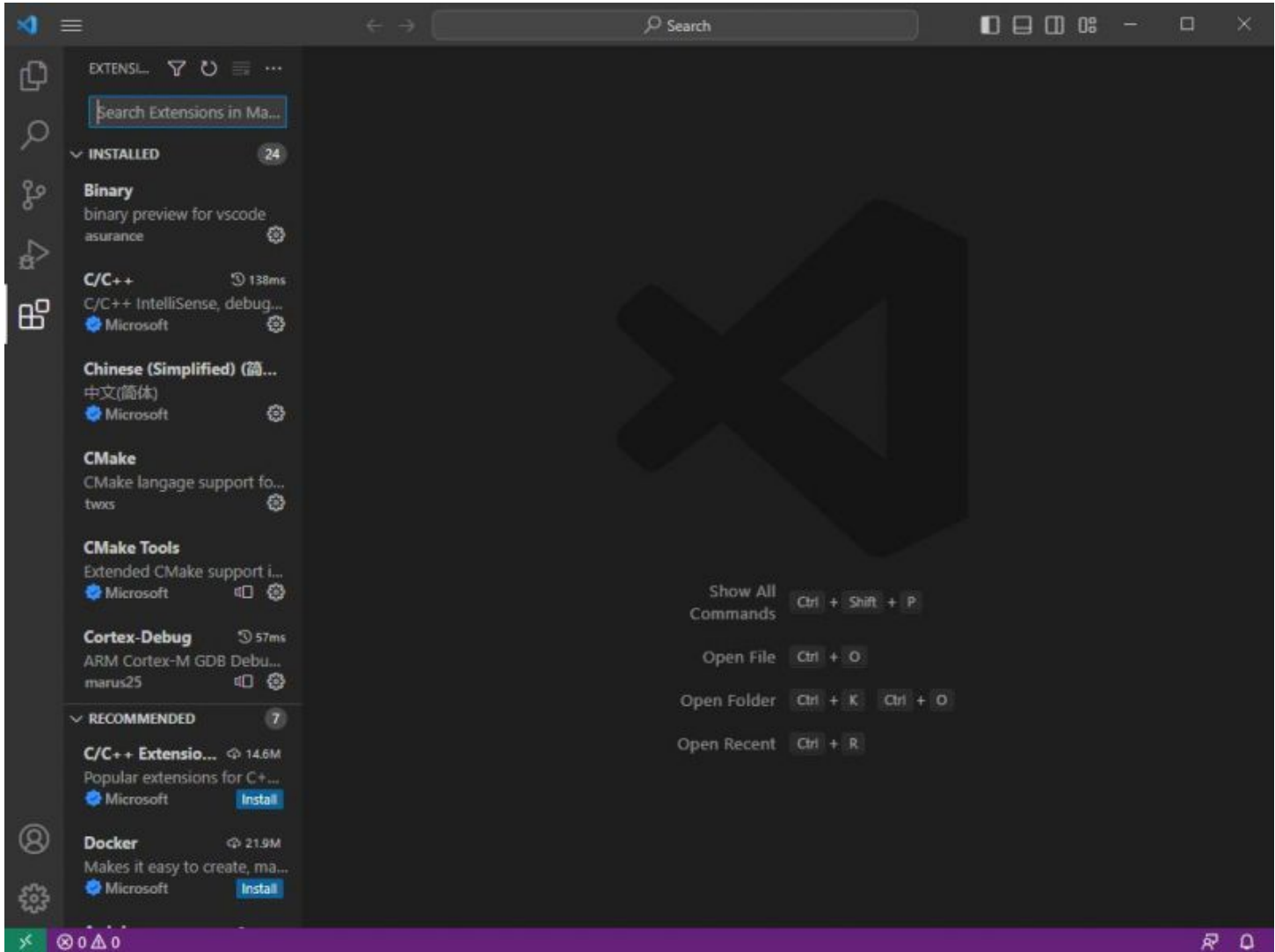
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscod-02.jpg)

- 第一二项开启后，可以直接通过鼠标右键文件或者目录打开VSCode，可以提高后续的使用体验。
- 第三项开启后，选择打开方式时，可以直接选择VSCode

安装Espressif IDF插件

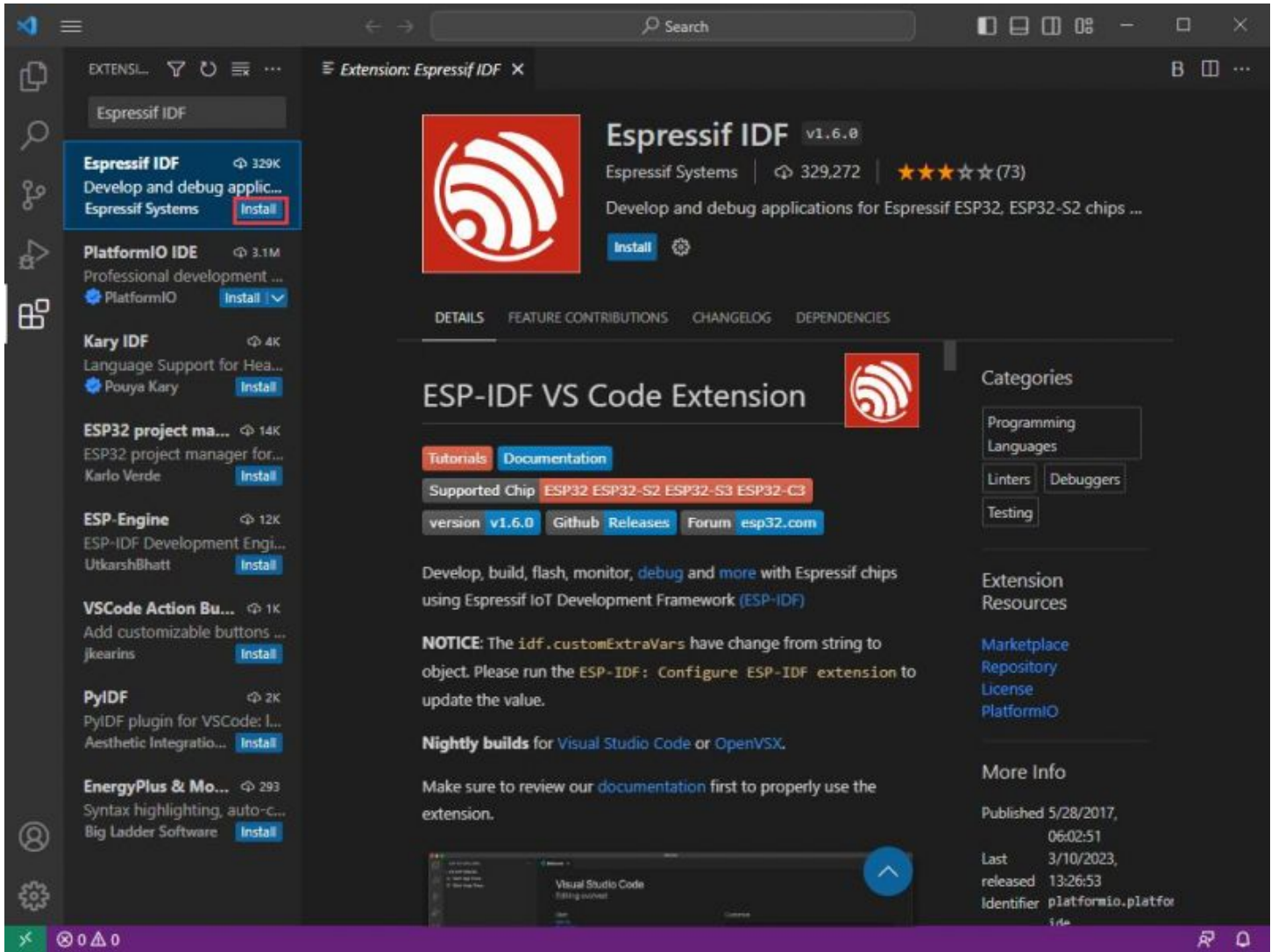
- 注：当前插件最新版本为V1.6.0，为体验一致，用户可以选择与我们一样的版本

1. 打开VSCode,使用快捷键Shift+Ctrl+X, 进入插件管理器



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-03.jpg)

2. 在搜索栏中，输入Espressif IDF，选择对应的插件点击 install即可



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-04.jpg)

3. 使用快捷键F1，输入

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

>esp-idf: configure esp-idf extension

ESP-IDF: Configure ESP-IDF extension recently used

Espressif IDF v1.6.0

Espressif Systems | 329,272 | 4.5 (73)

Develop and debug applications for Espressif ESP32, ESP32-S2 chips ...

Uninstall

DETAILS FEATURE CONTRIBUTIONS CHANGELOG DEPENDENCIES RUNTIME STATUS

ESP-IDF VS Code Extension

Tutorials Documentation

Supported Chip ESP32 ESP32-S2 ESP32-S3 ESP32-C3

version v1.6.0 Github Releases Forum esp32.com

Develop, build, flash, monitor, debug and more with Espressif chips using Espressif IoT Development Framework (ESP-IDF)

NOTICE: The `idf.customExtraVars` have change from string to object. Please run the ESP-IDF: Configure ESP-IDF extension to update the value.

Nightly builds for Visual Studio Code or OpenVSX.

Make sure to review our [documentation](#) first to properly use the extension.

Categories

- Programming Languages
- Snippets
- Debuggers

Extension Resources

- Marketplace
- Repository
- License
- Espressif Systems

More Info

Published 12/31/2019, 17:54:02

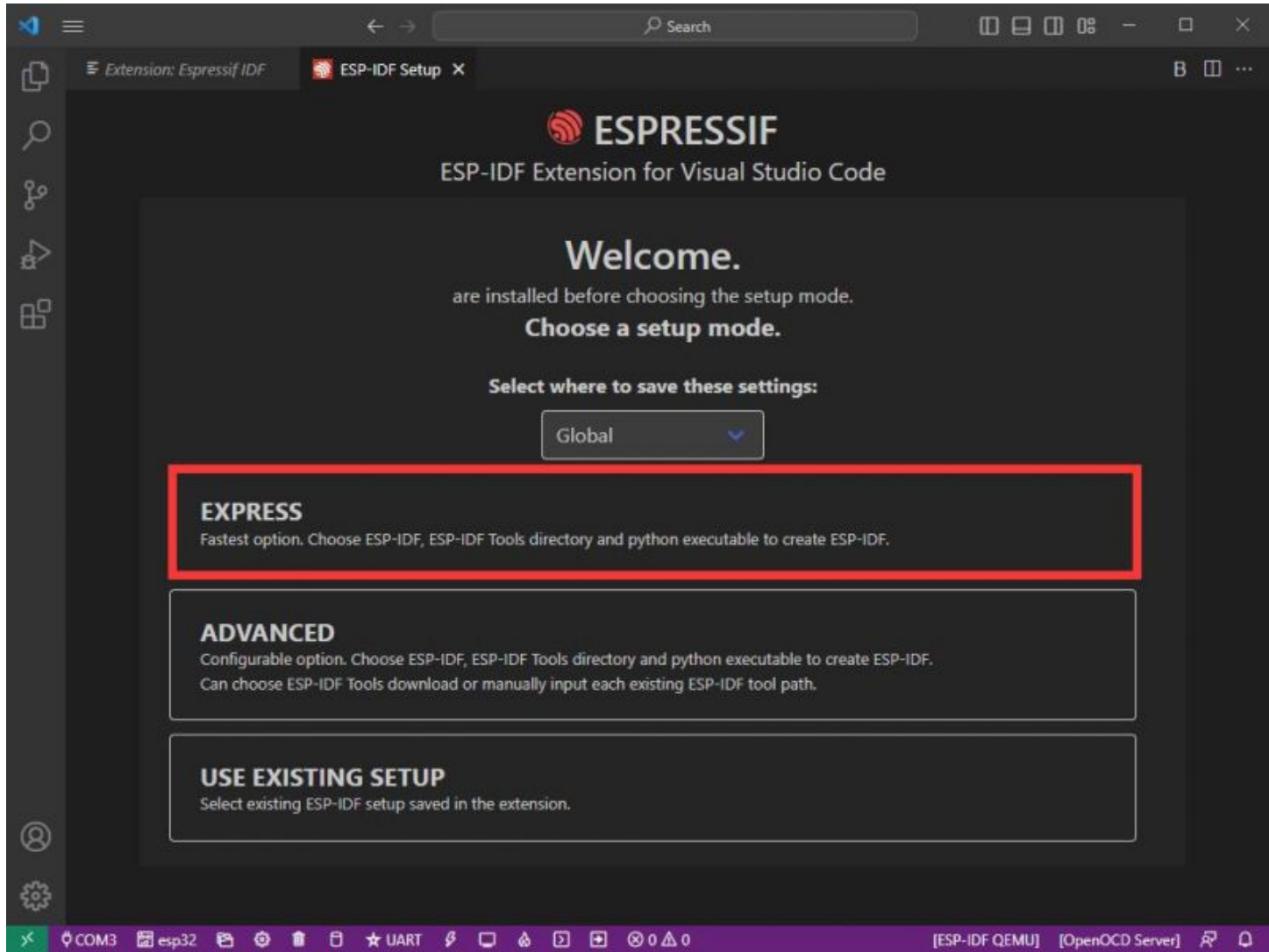
Last released 2/28/2023, 22:45:30

Identifier espressif.esp-idf-extension

[ESP-IDF QEMU] [OpenOCD Server]

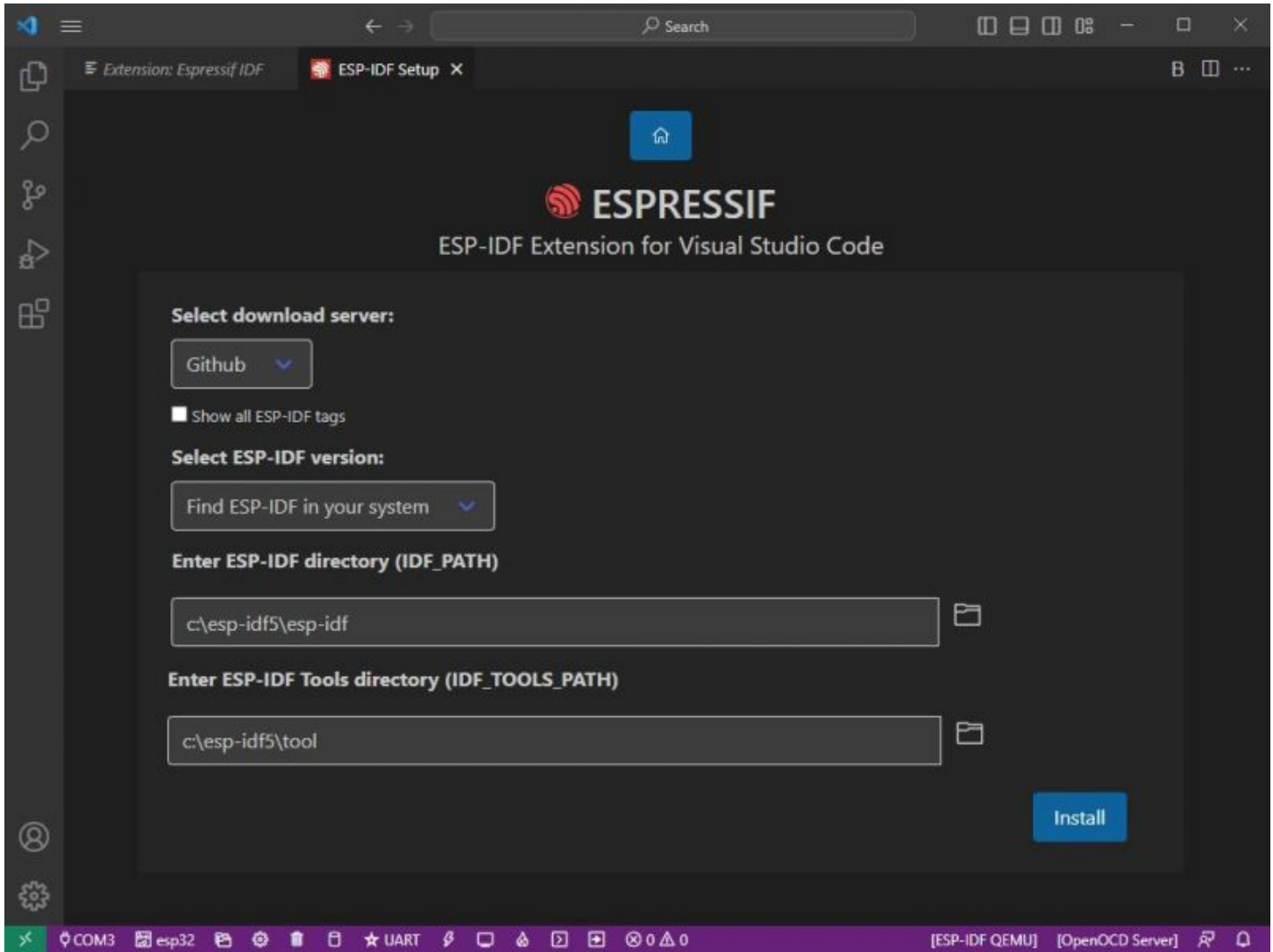
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-05.jpg)

4. 选择express (此教程针对第一次安装的用户, 故只讲述初次的通用安装教程)



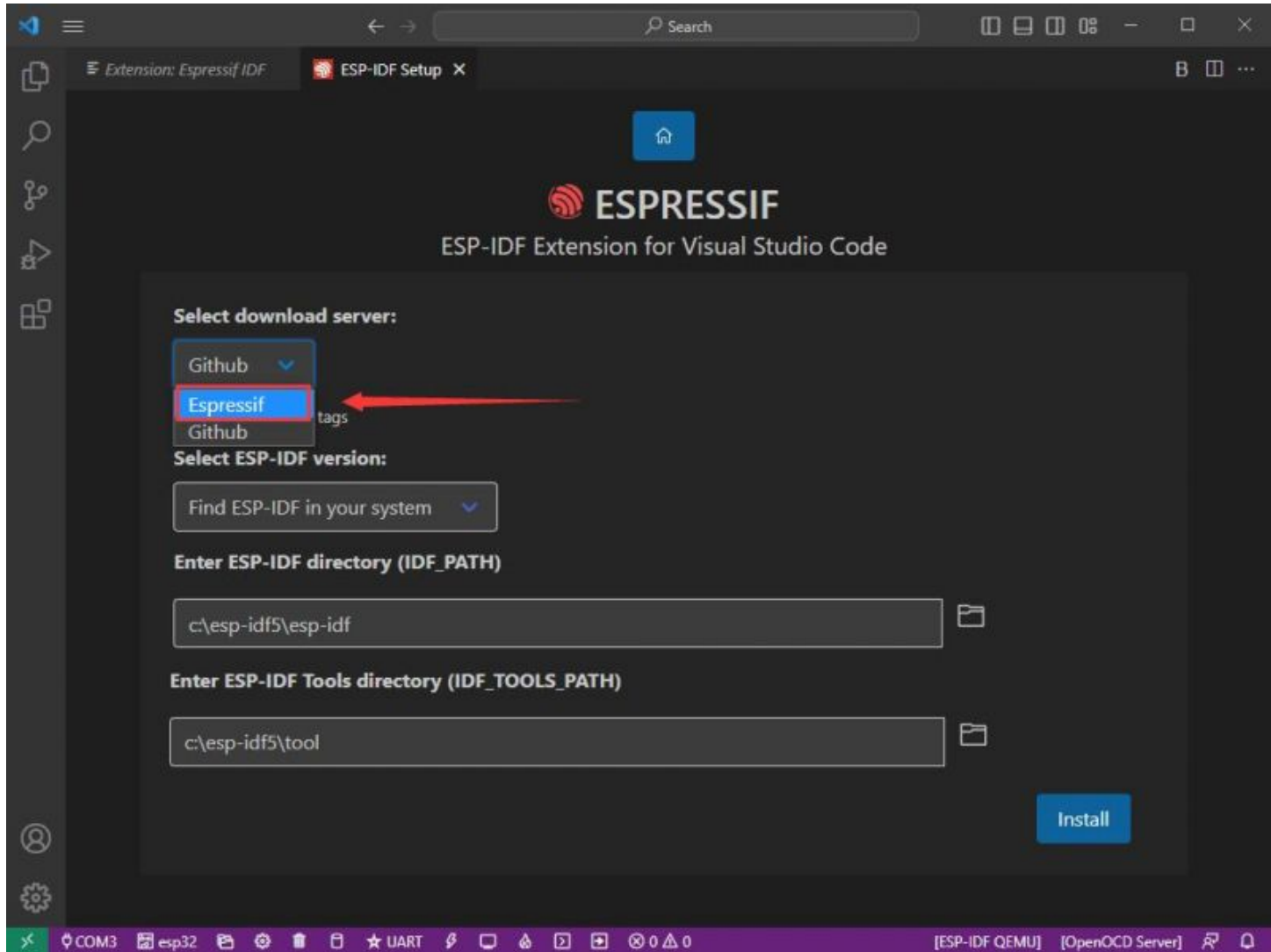
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-06.jpg)

5. 打开后显示该界面



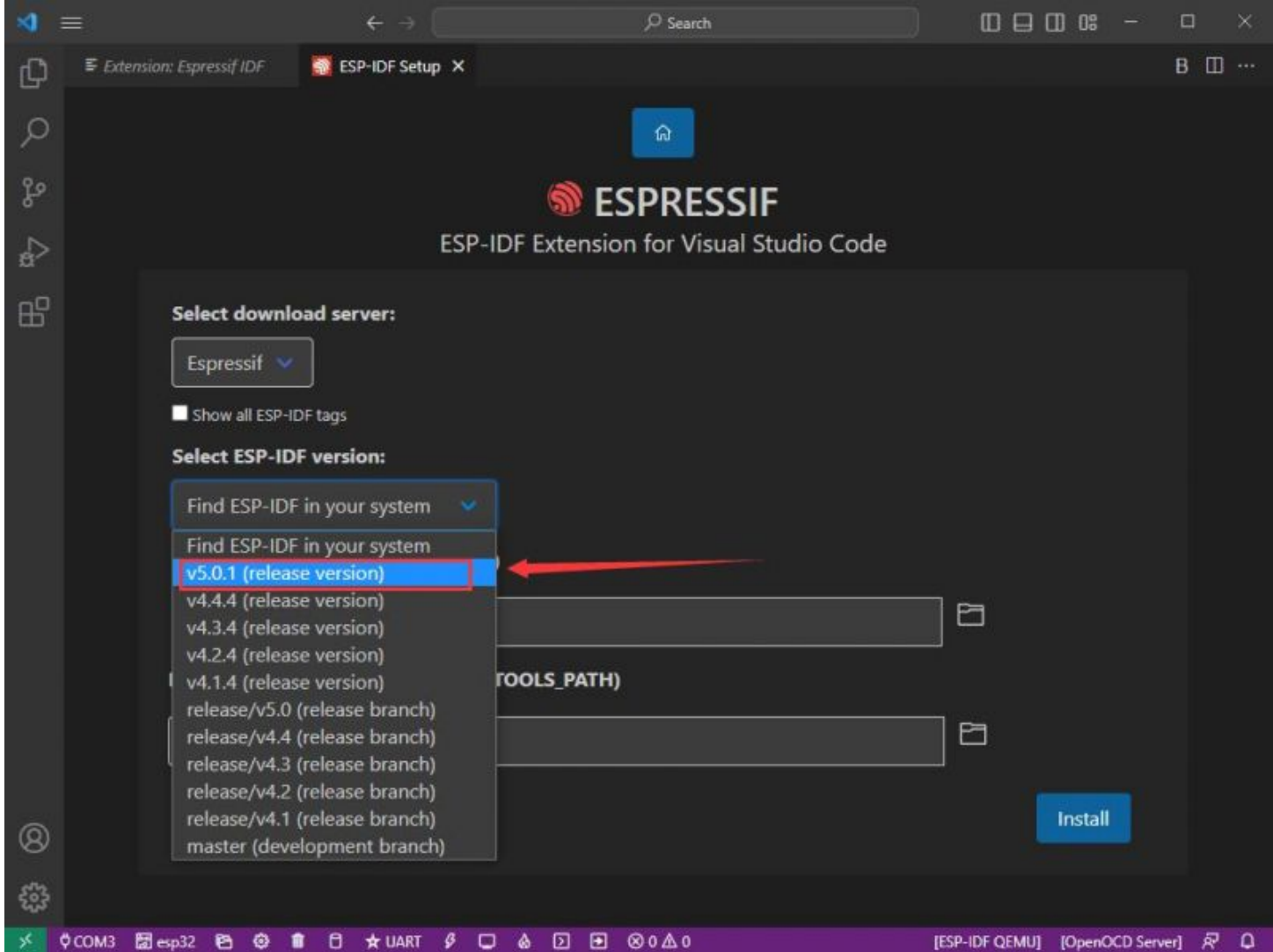
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-07.jpg)

6. 选择下载服务器，我们推荐国内用户使用Espressif作为你的下载服务器



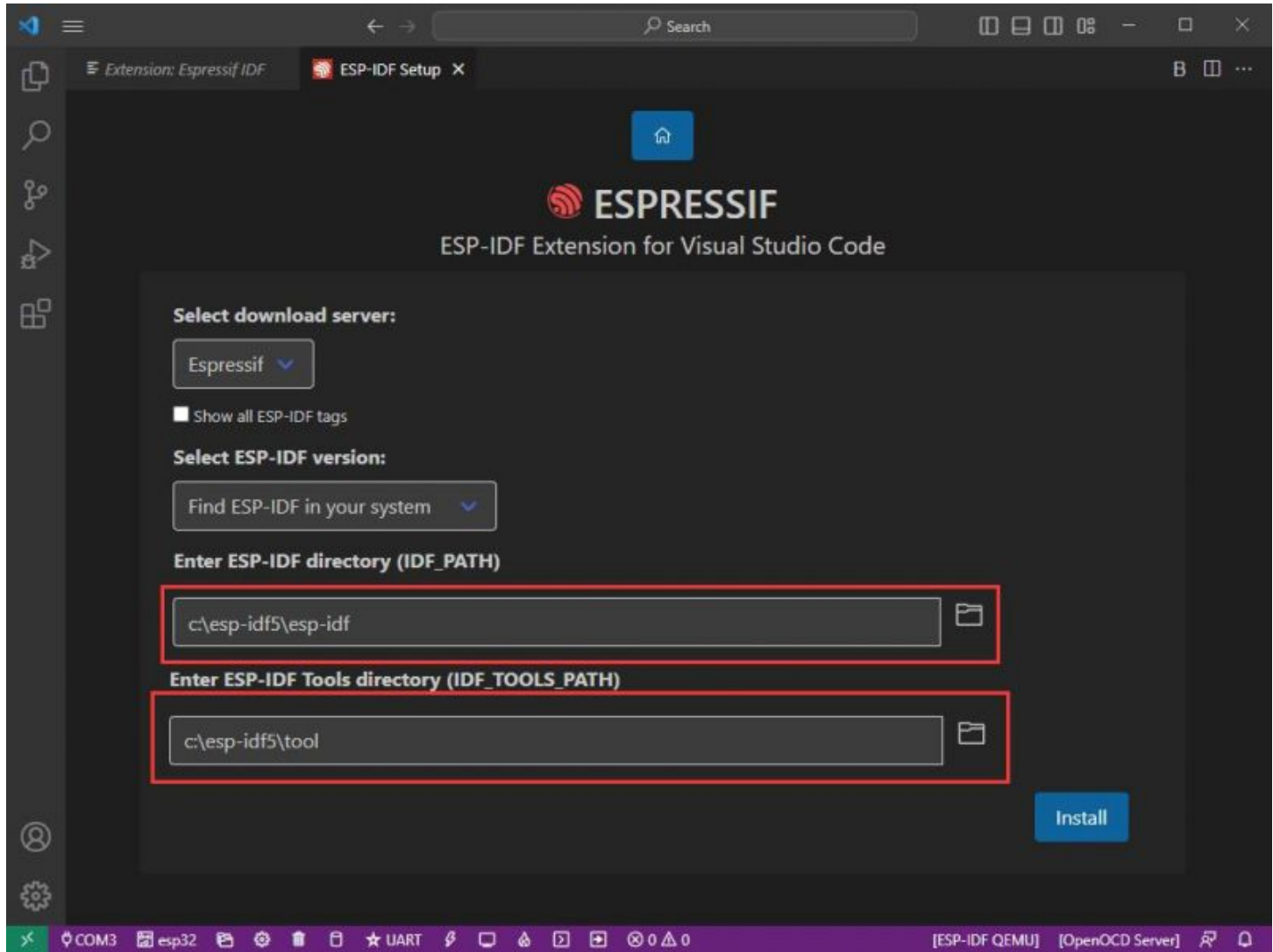
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-08.jpg)

7. 选择想要现在的ESP-IDF版本，我们选择最新的V5.0.1(注意ESP-IDF从V4.4版本后才开始支持ESP32-S3)



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscode-09.jpg)

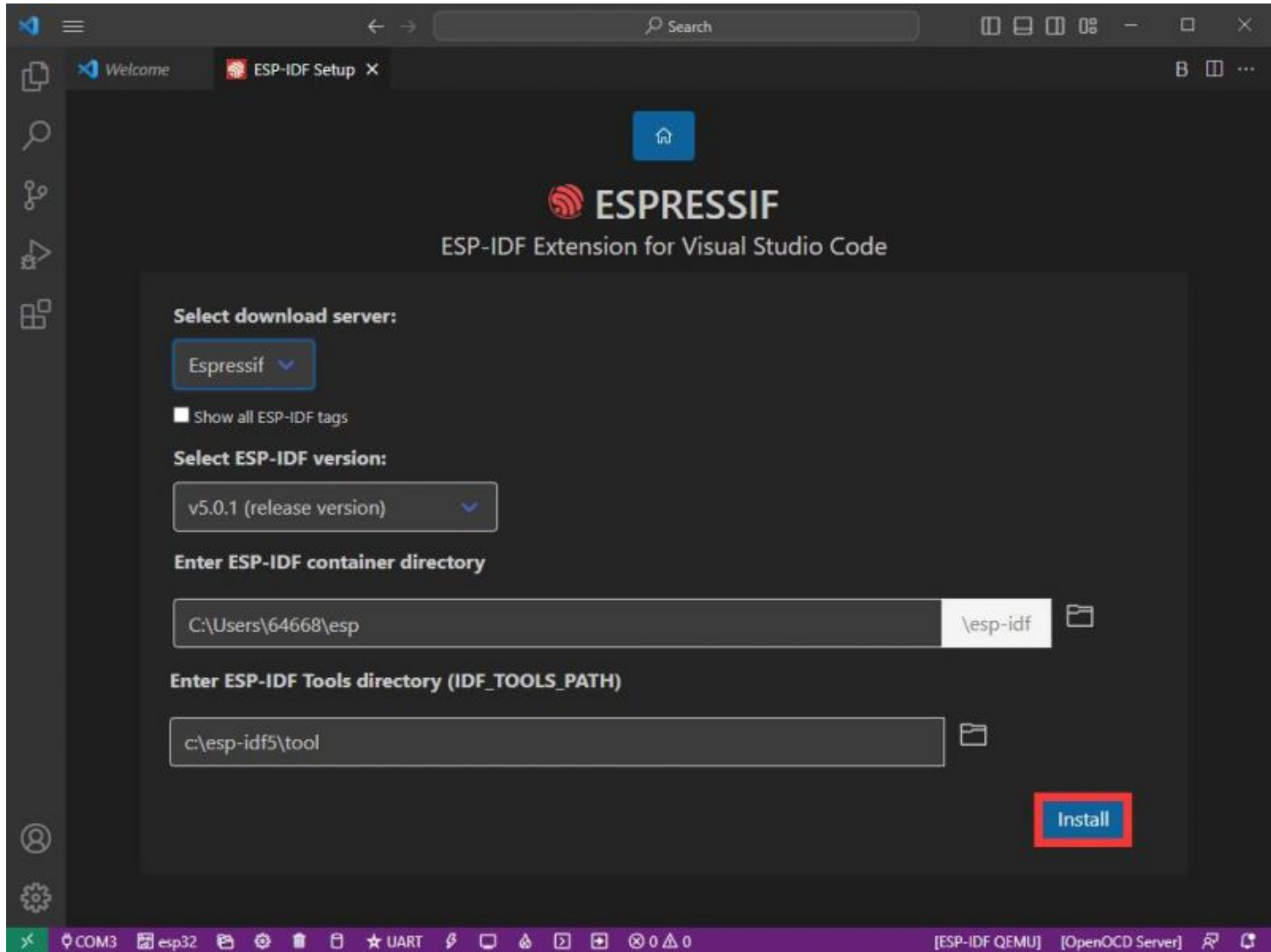
8. 下面两个分别为ESP-IDF容器安装地址和ESP-IDF所需的工具安装地址，



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-10.jpg)

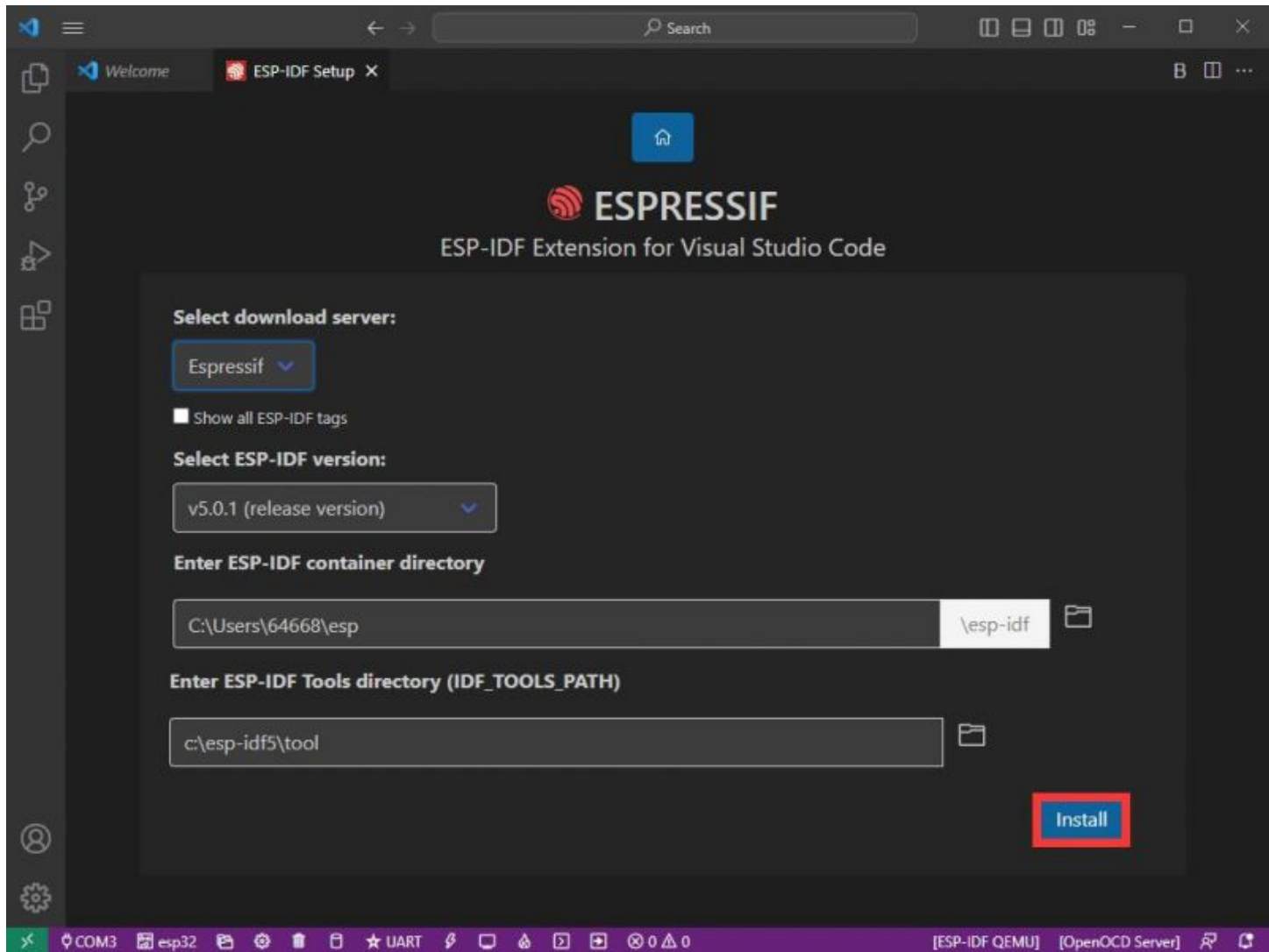
- **注意：如果之前有安装过ESP-IDF，或者失败过的，请务必彻底删除文件或者创建全新的无中文路径**

9. 配置完成后, 点击 install 进行下载



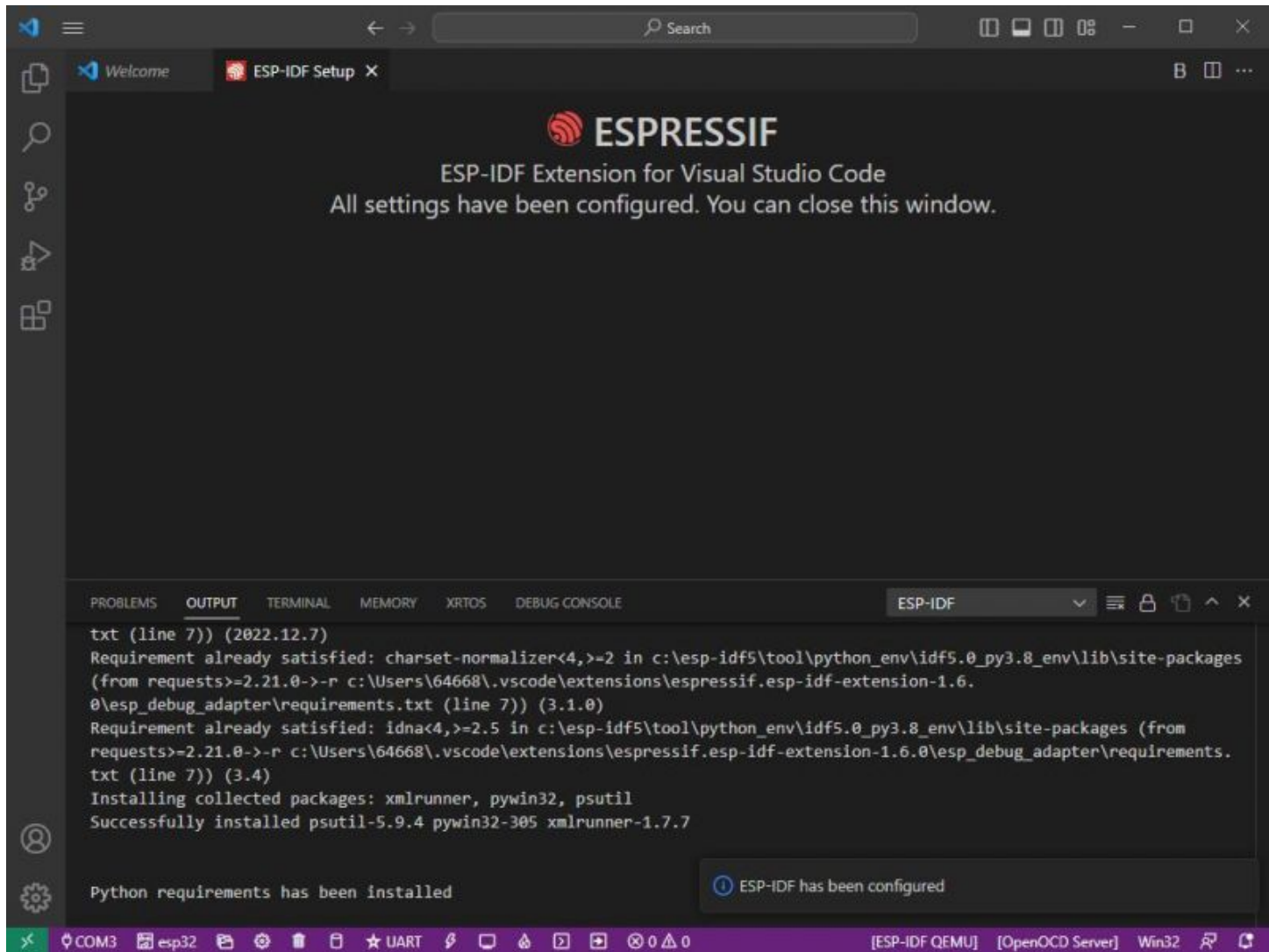
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-11.jpg)

10. 进入下载页面，其会自动安装对应工具与环境，稍等片刻即可



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-11.jpg)

11. 安装完成后, 会进入以下界面, 说明安装完成



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-13.jpg)

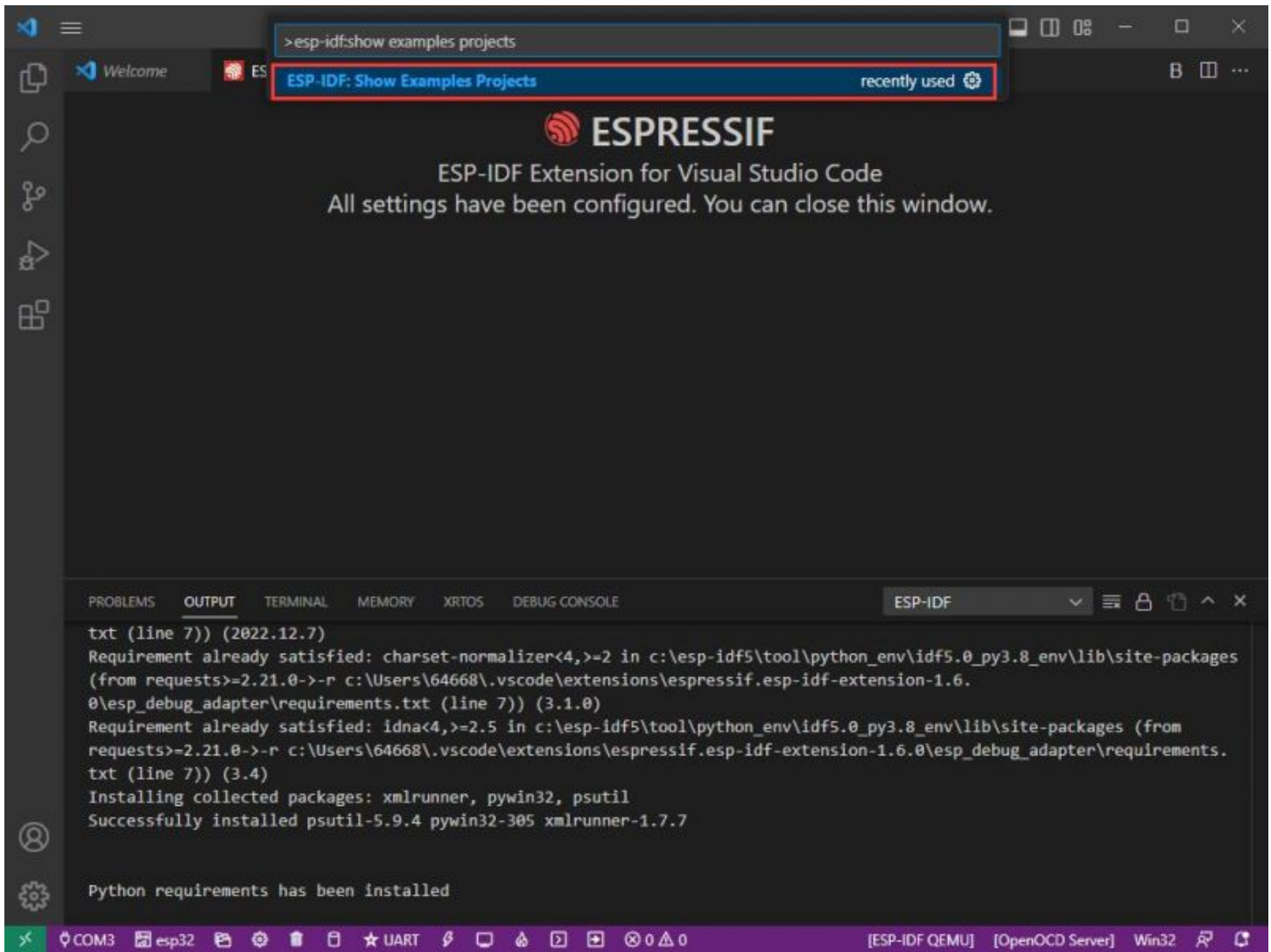
使用官方例程

- ESP官方为我们提供大量的例程,并且写详细的使用方法与效果, 点击此处查看 (<https://github.com/espressif/esp-idf/tree/master/examples>)

创建例程

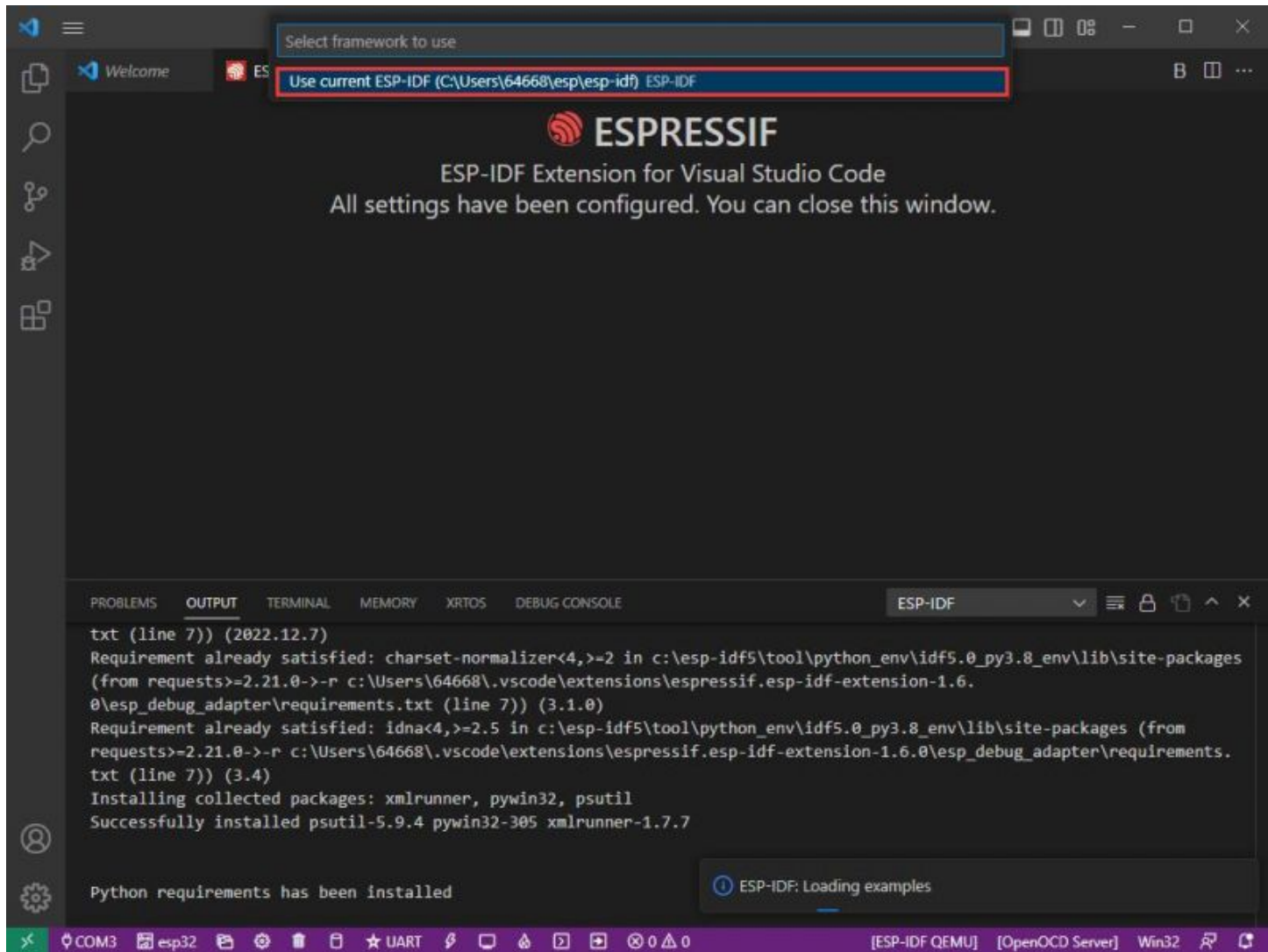
1. 使用快捷键F1, 输入

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



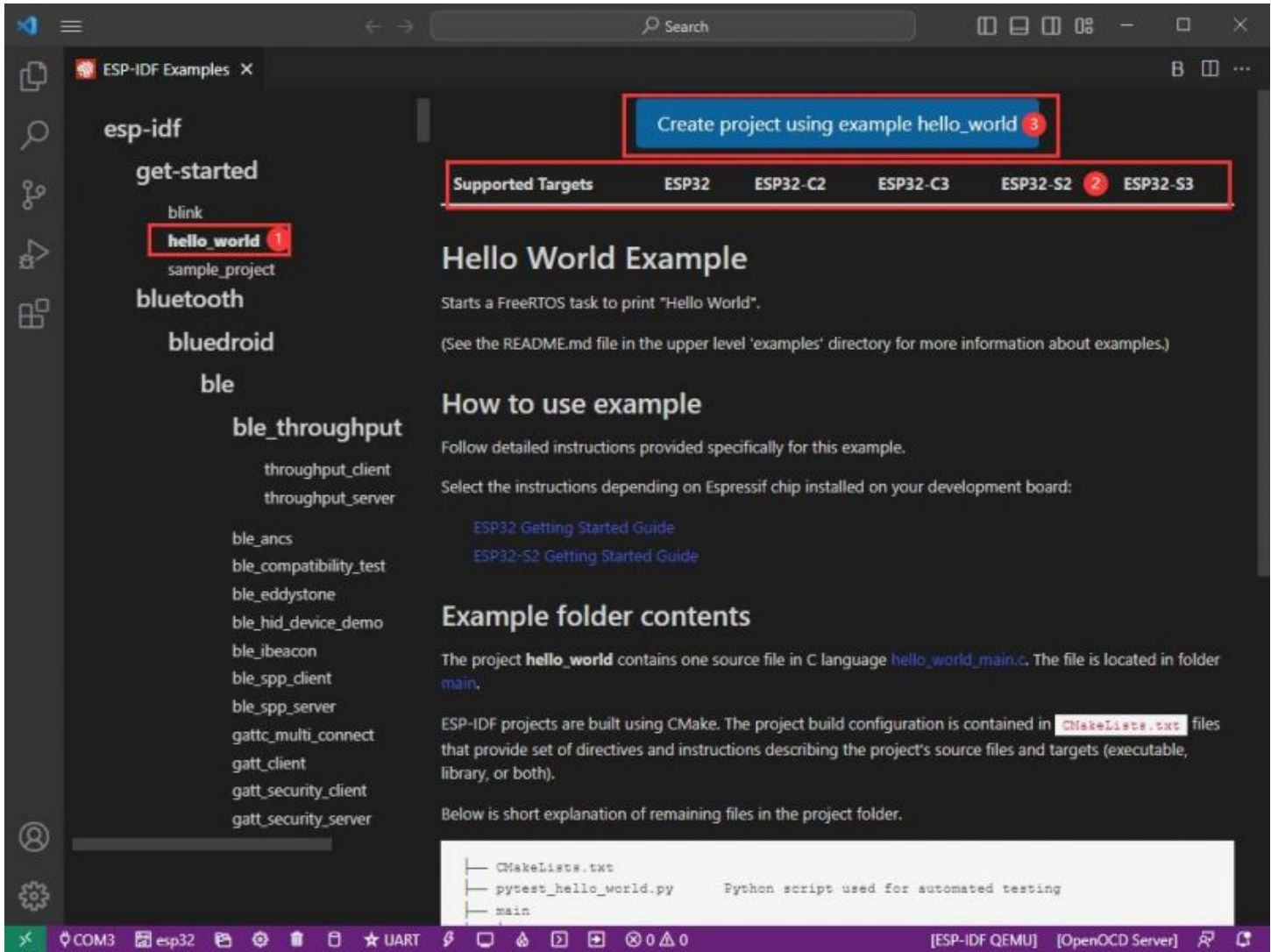
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsco-14.jpg)

2. 选择你当前的IDF版本



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-15.jpg)

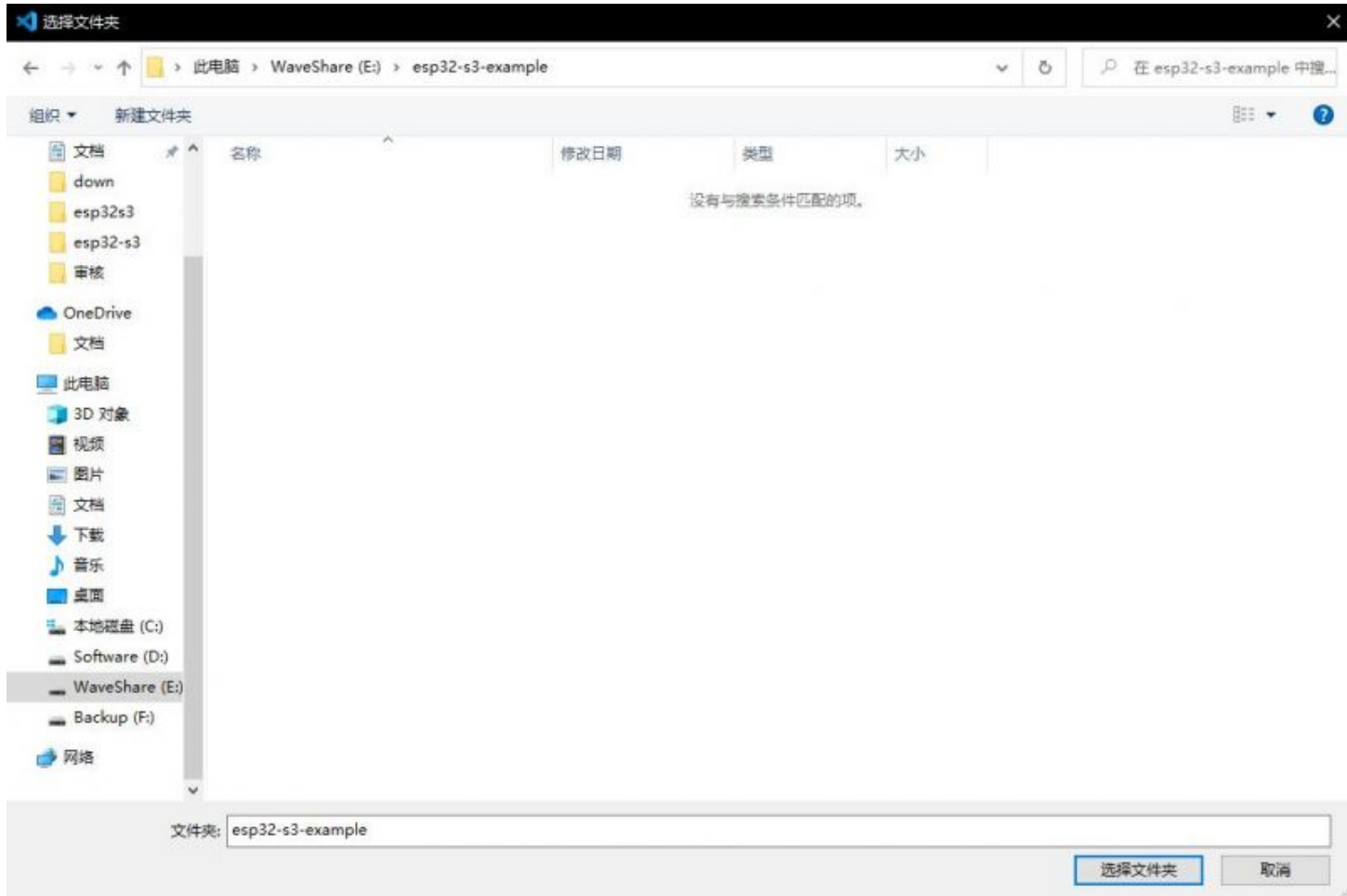
3. 以Hello world例程为例



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscode-16.jpg)

4. ①选择对应例程
5. ②其readme会说明该例程适用于什么芯片（下文有介绍例程怎么使用与文件结构，这里略）
6. ③点击创建例程

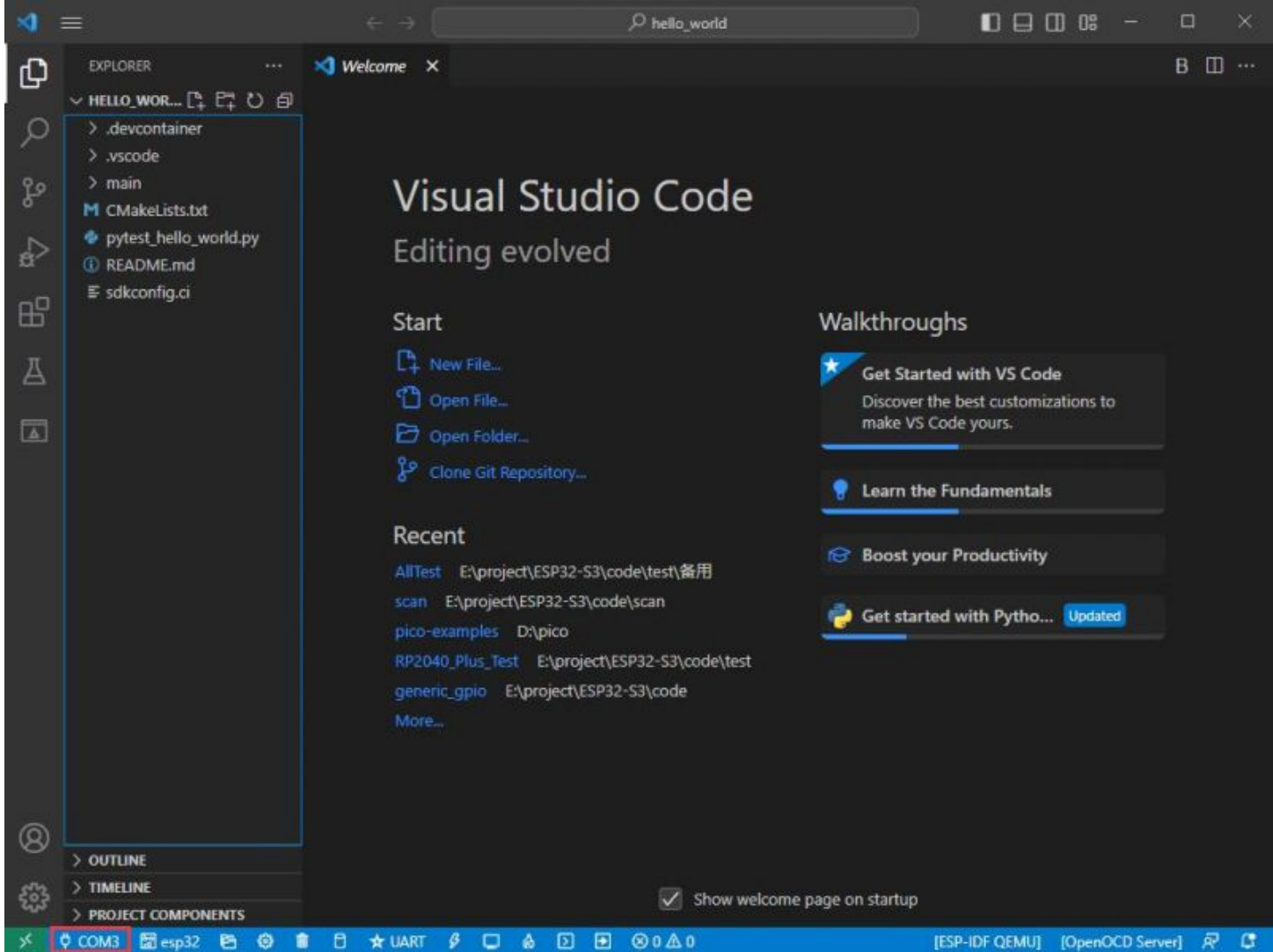
7. 选择放置例程的路径, 要求无例程同名文件夹



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-17.jpg)

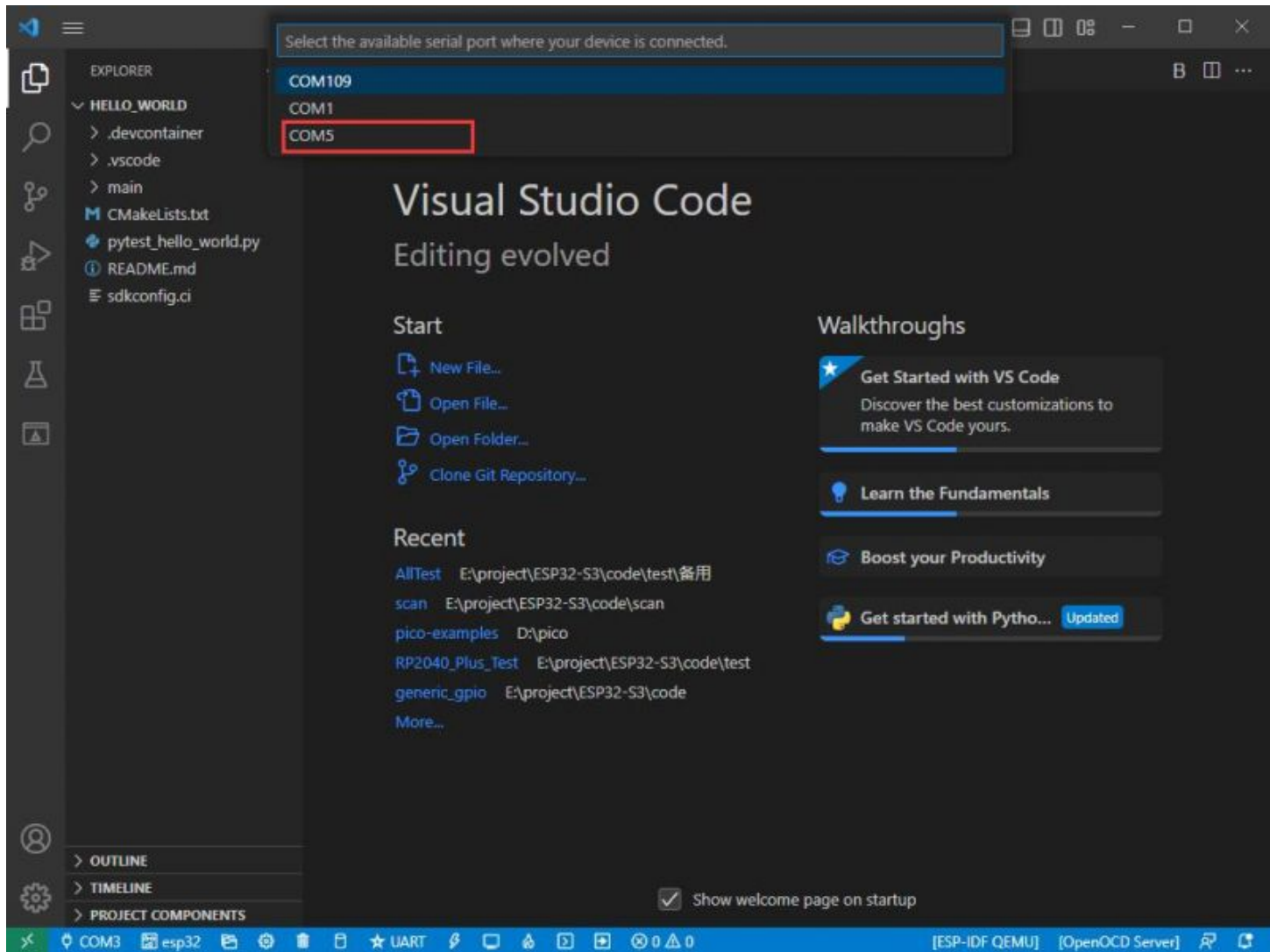
修改COM口

1. 此处显示使用对应的COM口, 点击可以修改对应COM口



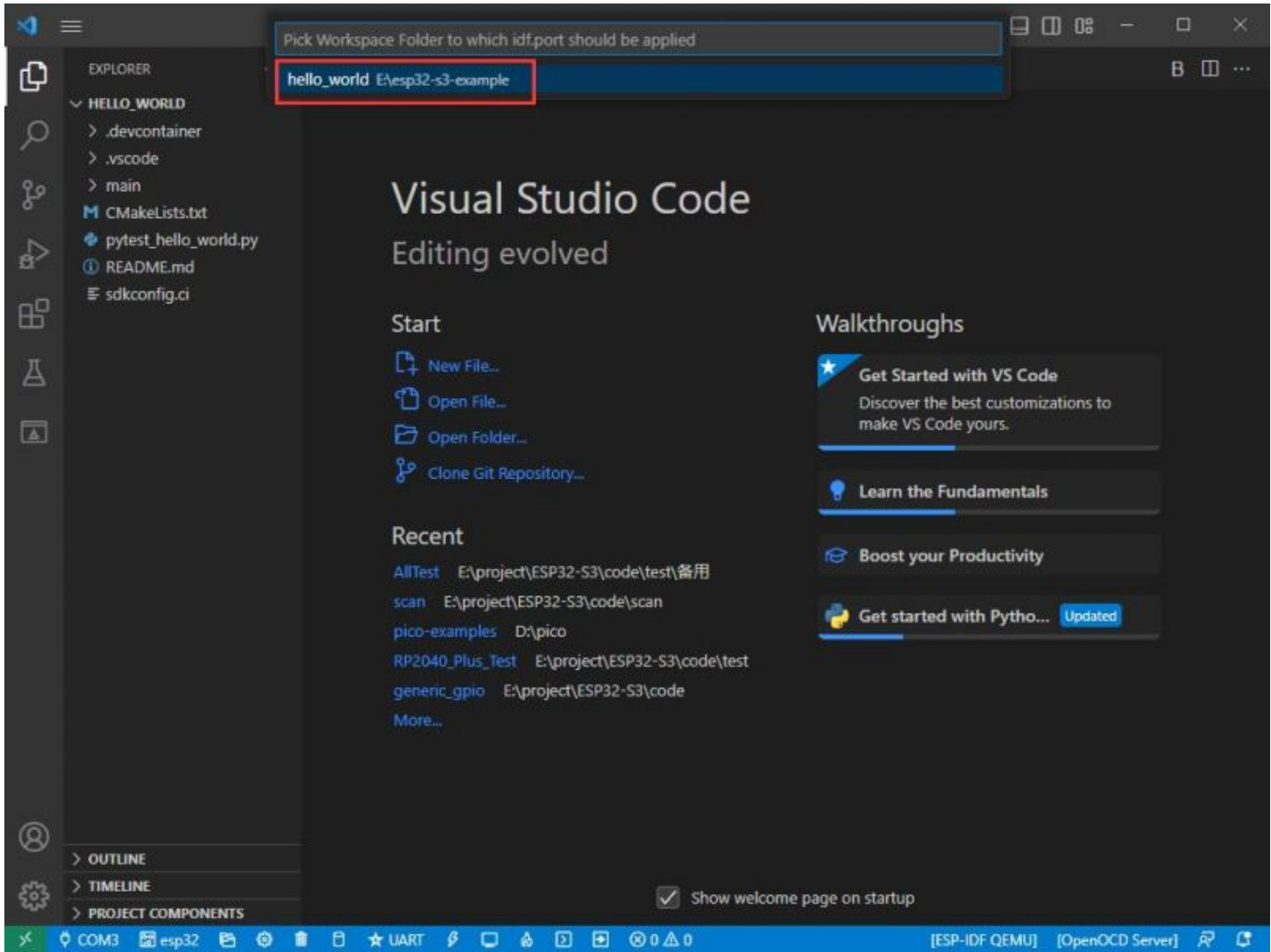
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscode-18.jpg)

2. 我们的CH343的COM为COM5，所以我们选择COM5，请根据自己CH343对应COM口进行选择



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-19.jpg)

3. 选择使用的工程或者例程

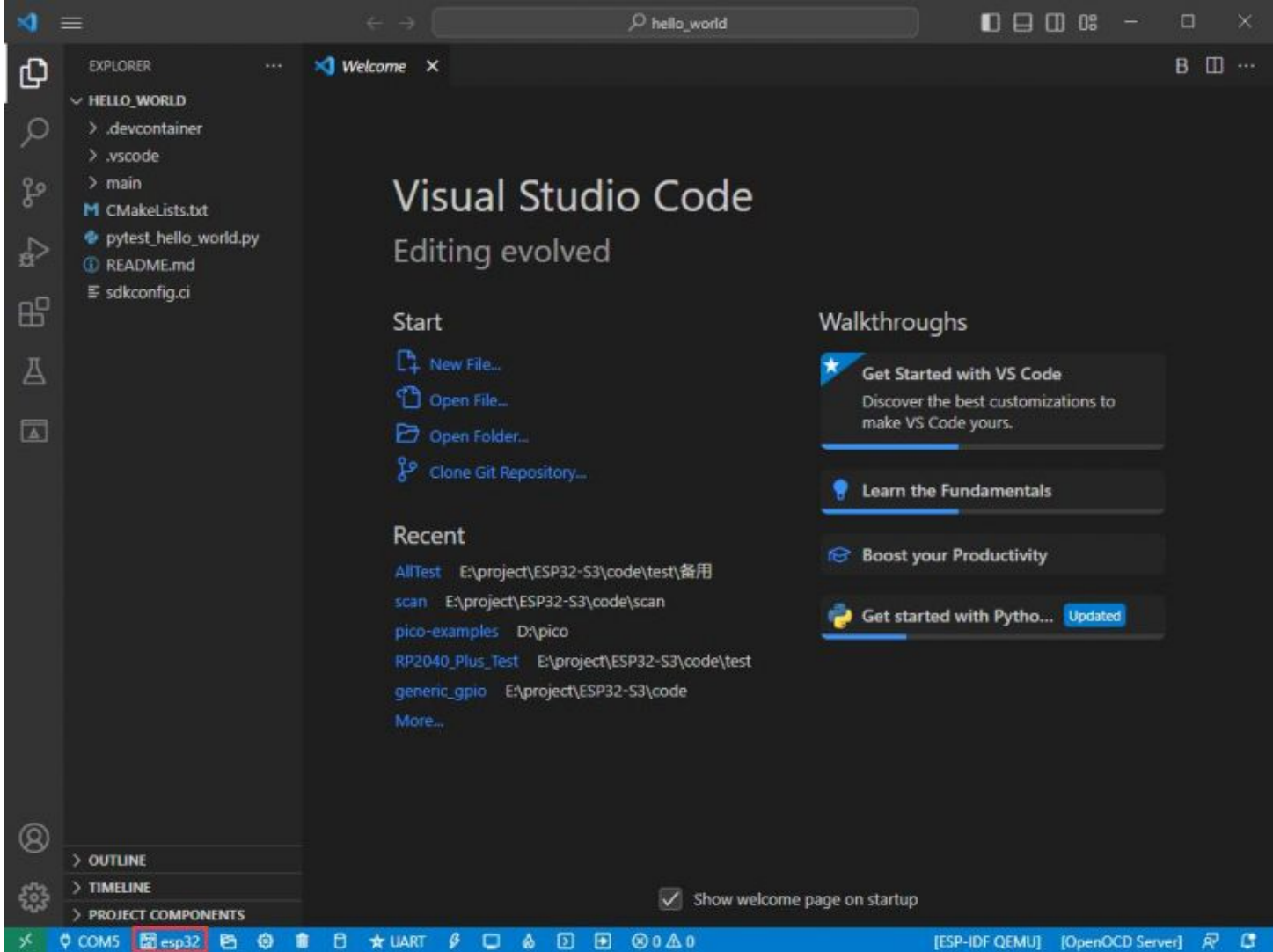


(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-20.jpg)

4. 然后我们的COM口就修改好了

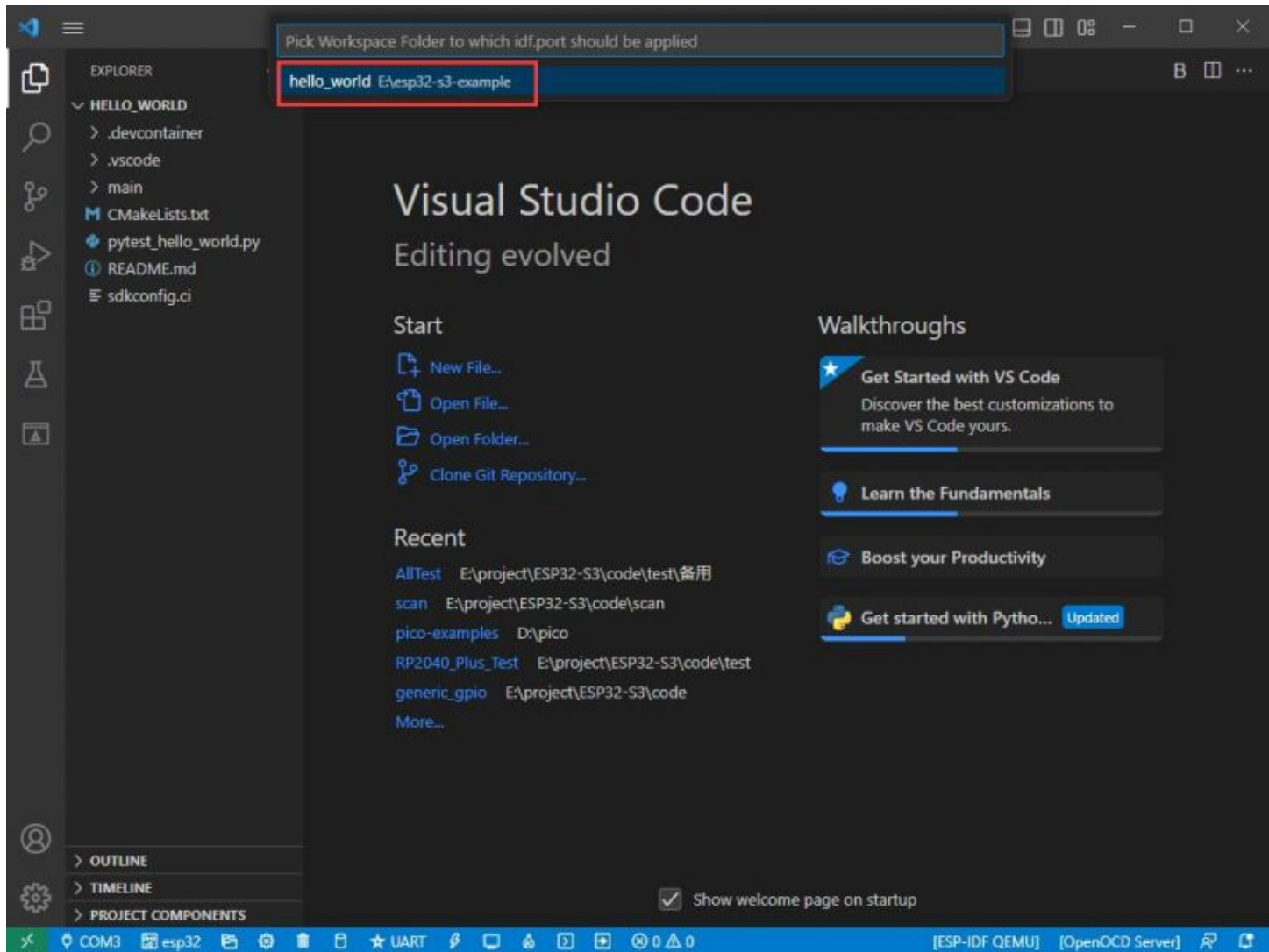
修改驱动对象

1. 此处显示的是使用的驱动对象，点击可以修改对应驱动对象



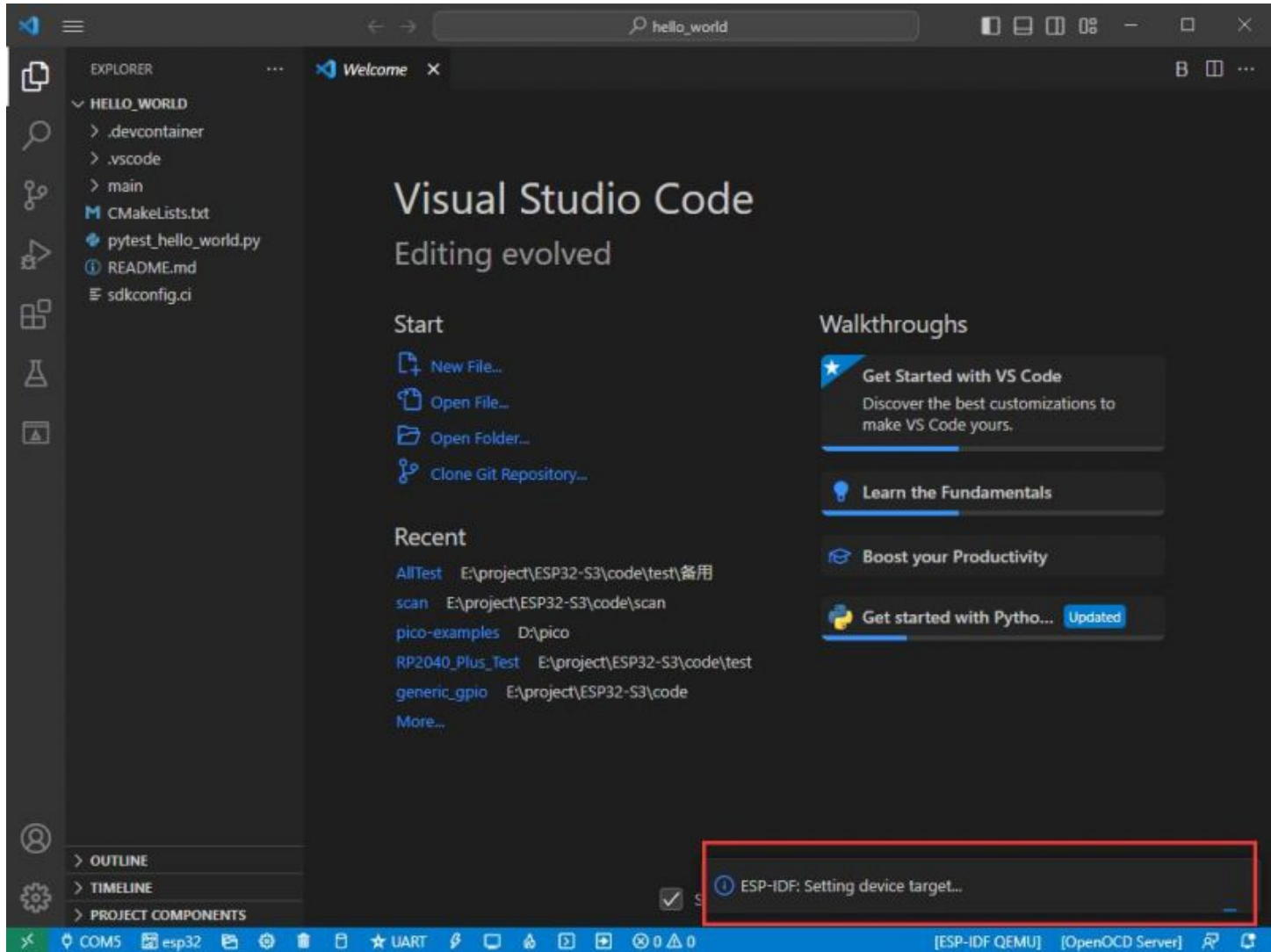
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-21.jpg)

2. 选择使用的工程或者例程



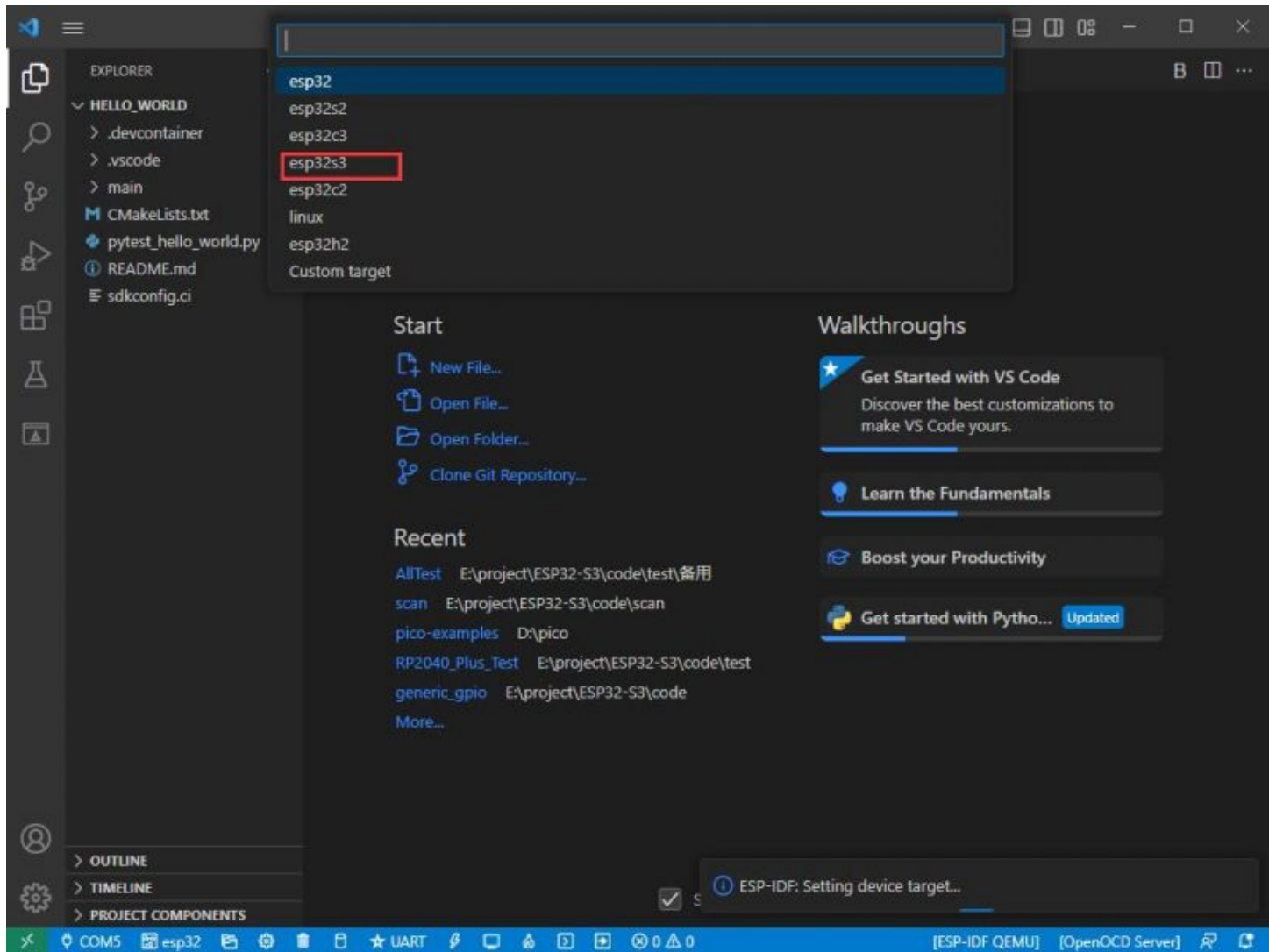
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-20.jpg)

3. 点击后需要稍等片刻



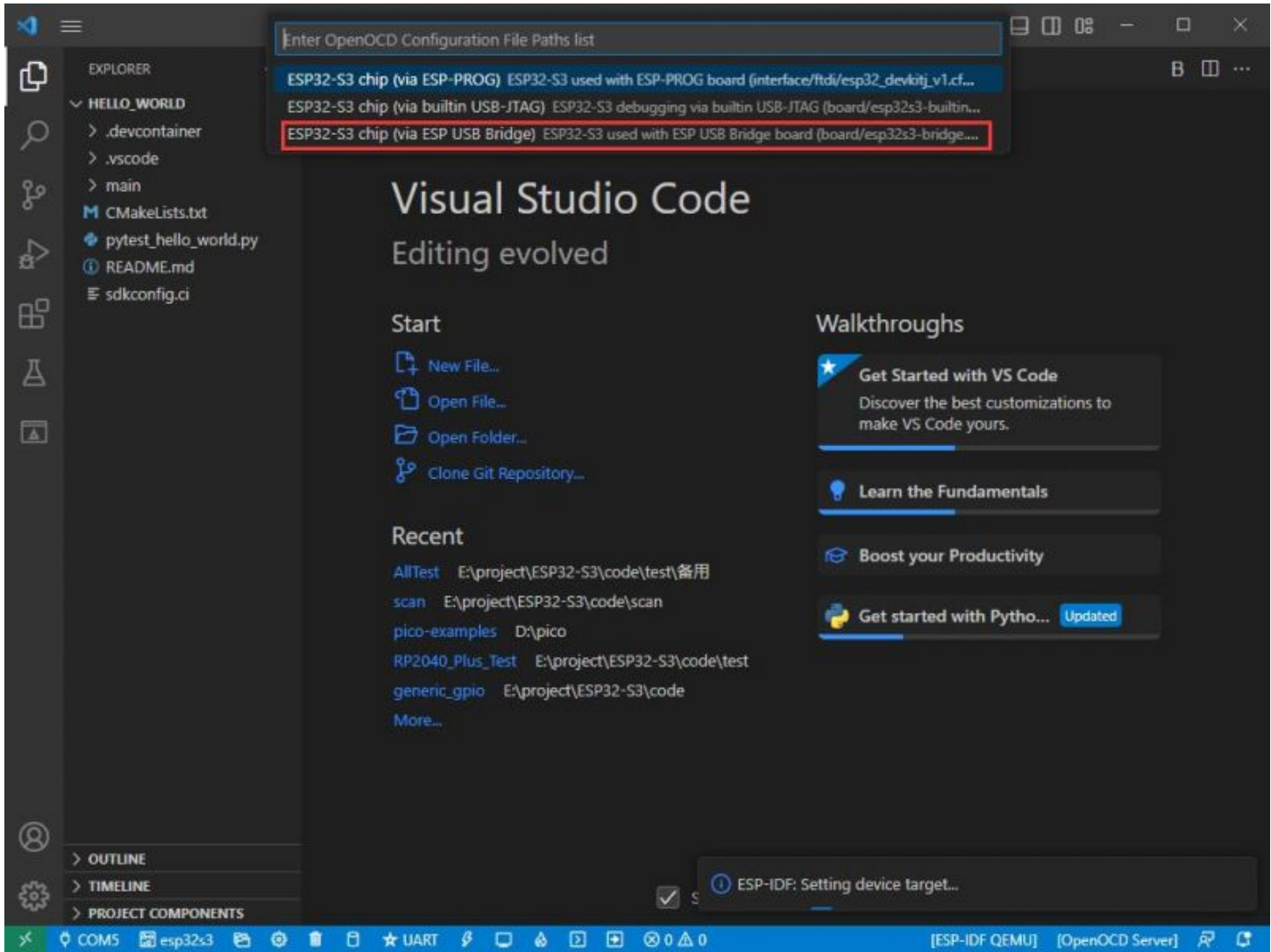
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscod-22.jpg)

4. 选择我们需要驱动的对象，也就是我们的主芯片为ESP32S3



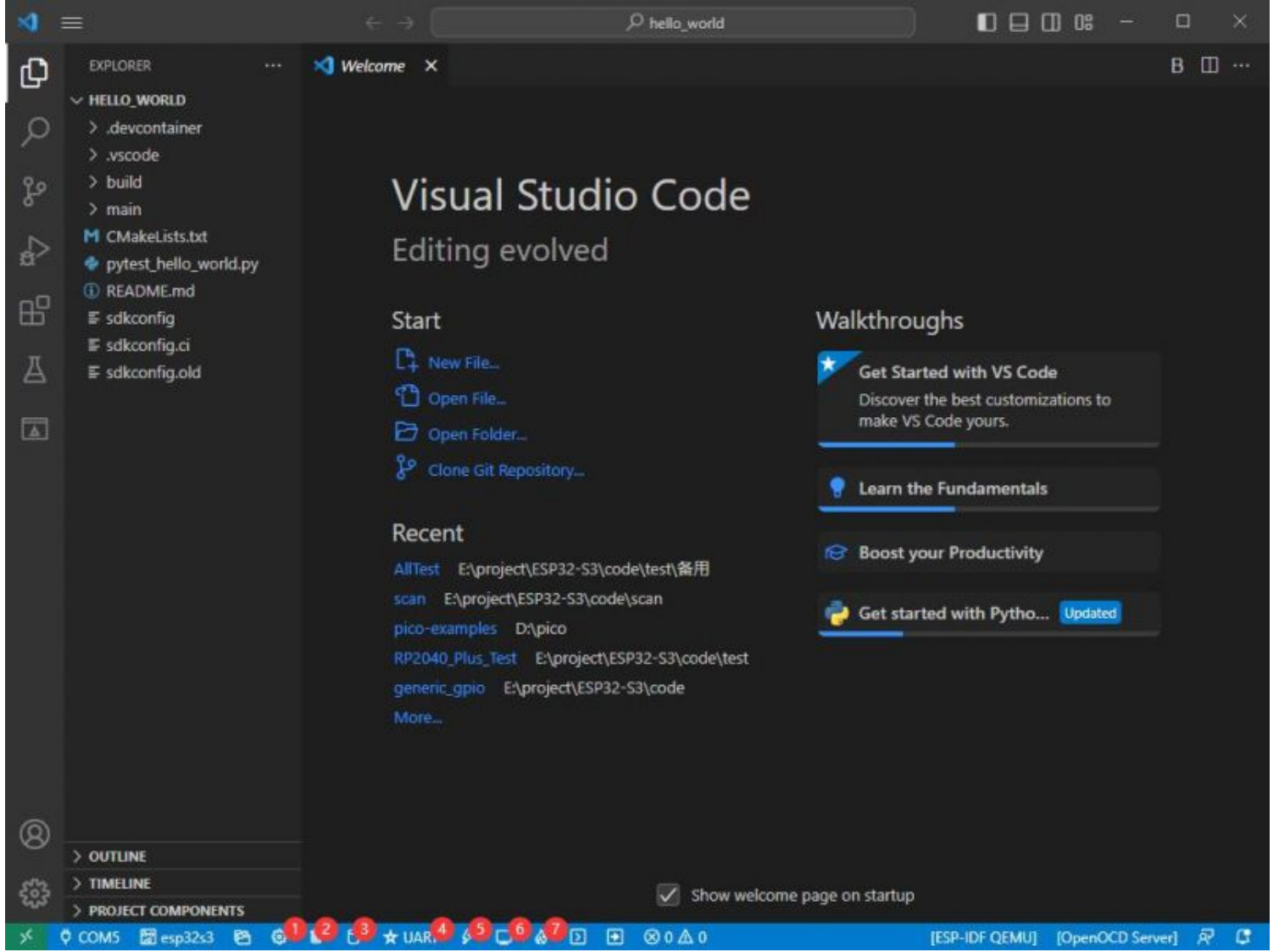
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-23.jpg)

5. 选择openocd的路径，这里对我们没有影响，所以我们随便选择一个即可



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-24.jpg)

其余状态栏简介



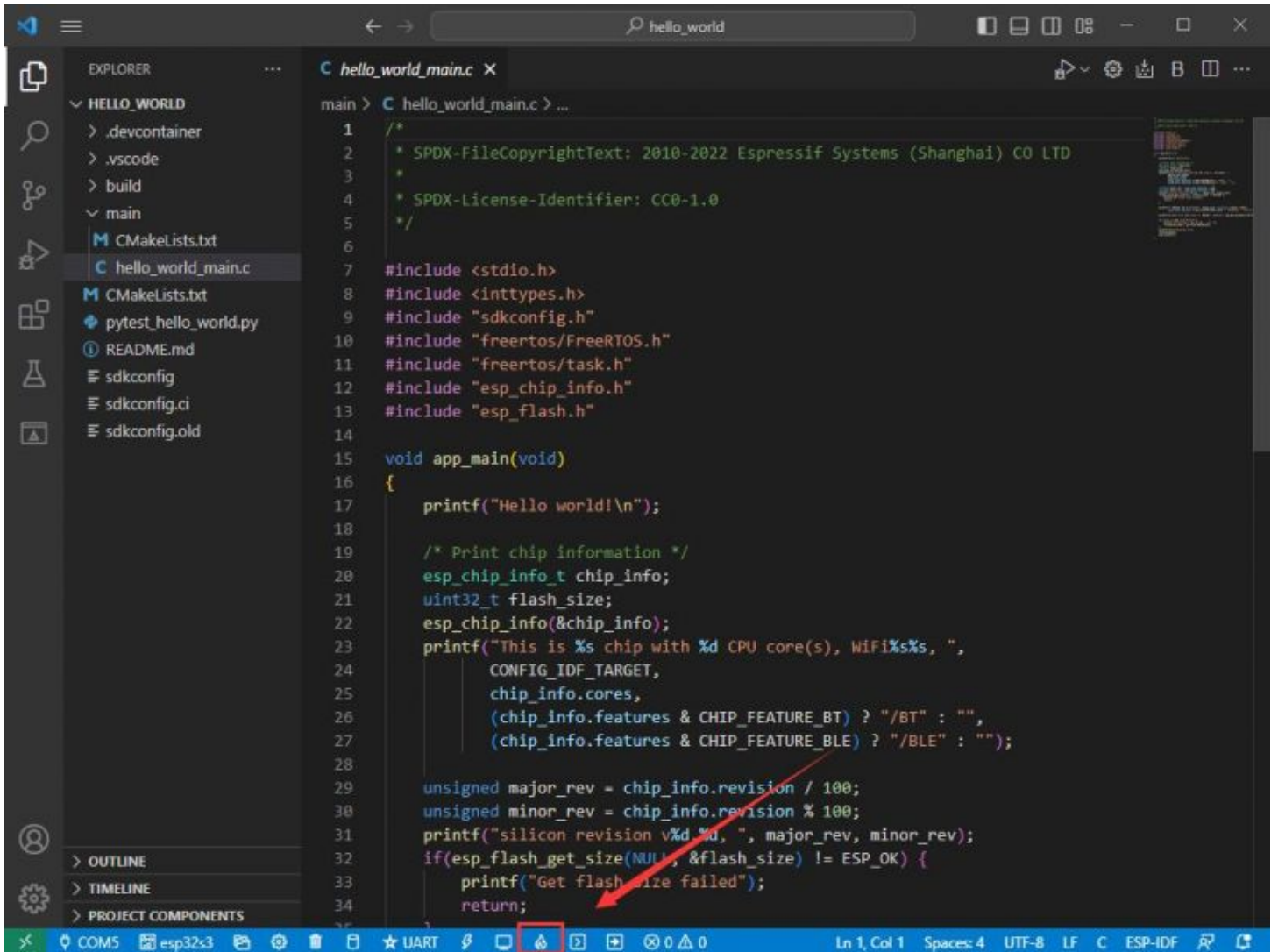
(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-25.jpg)

- ①SDK 配置编辑器，ESP-IDF很多功能与配置可以在其内修改
- ②全部清理，清空所有编译文件，
- ③编译
- ④当前下载方式，默认为UART
- ⑤烧录当前固件，请在编译后进行
- ⑥打开串口监视器，用于查看串口信息

- ⑦编译，烧录，打开串口监视器 一体按键（调试时最常用）

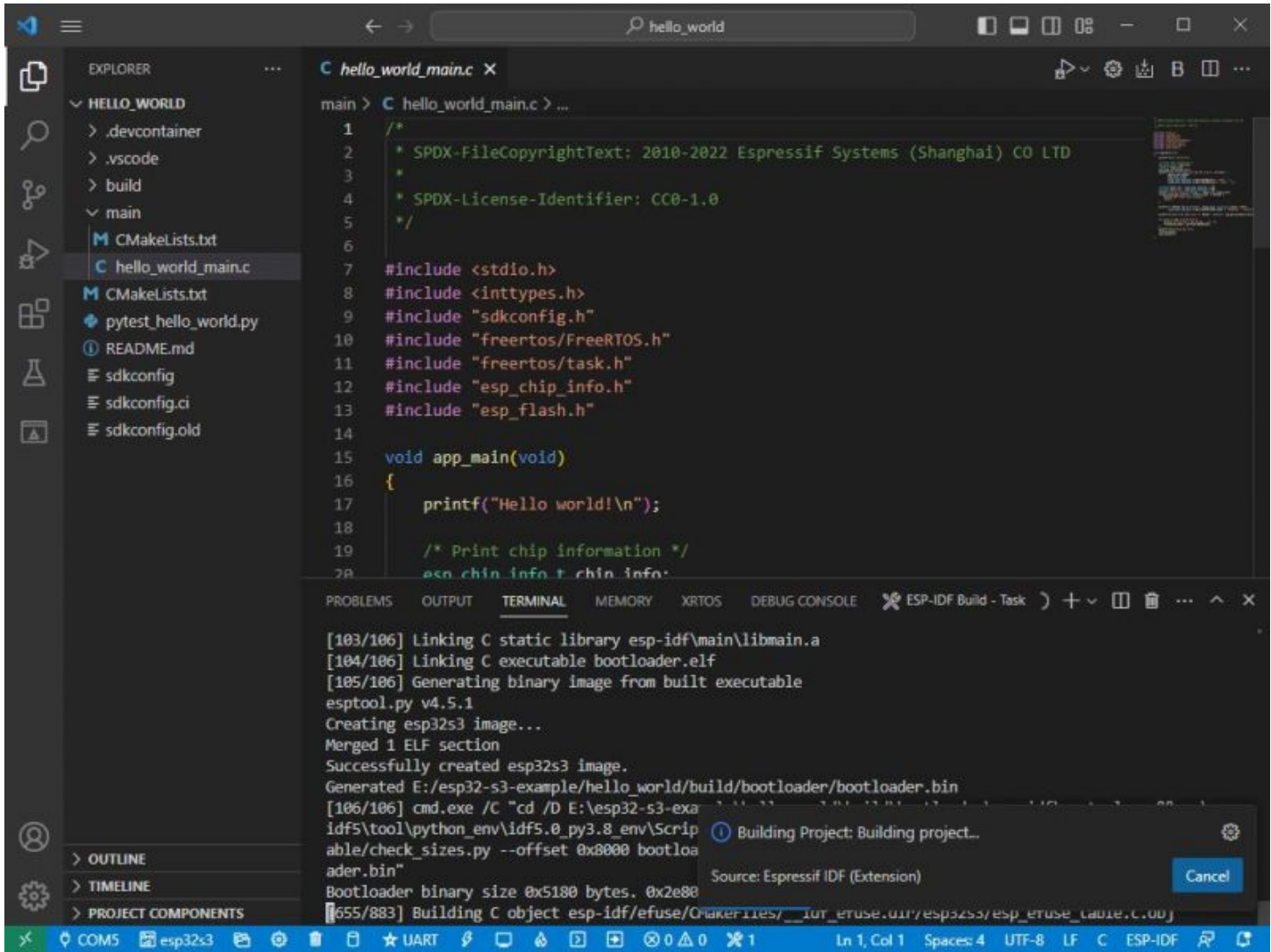
编译、烧录、串口监视

1. 点击我们之前介绍的 编译，烧录，打开串口监视器按键



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vscode-29.jpg)

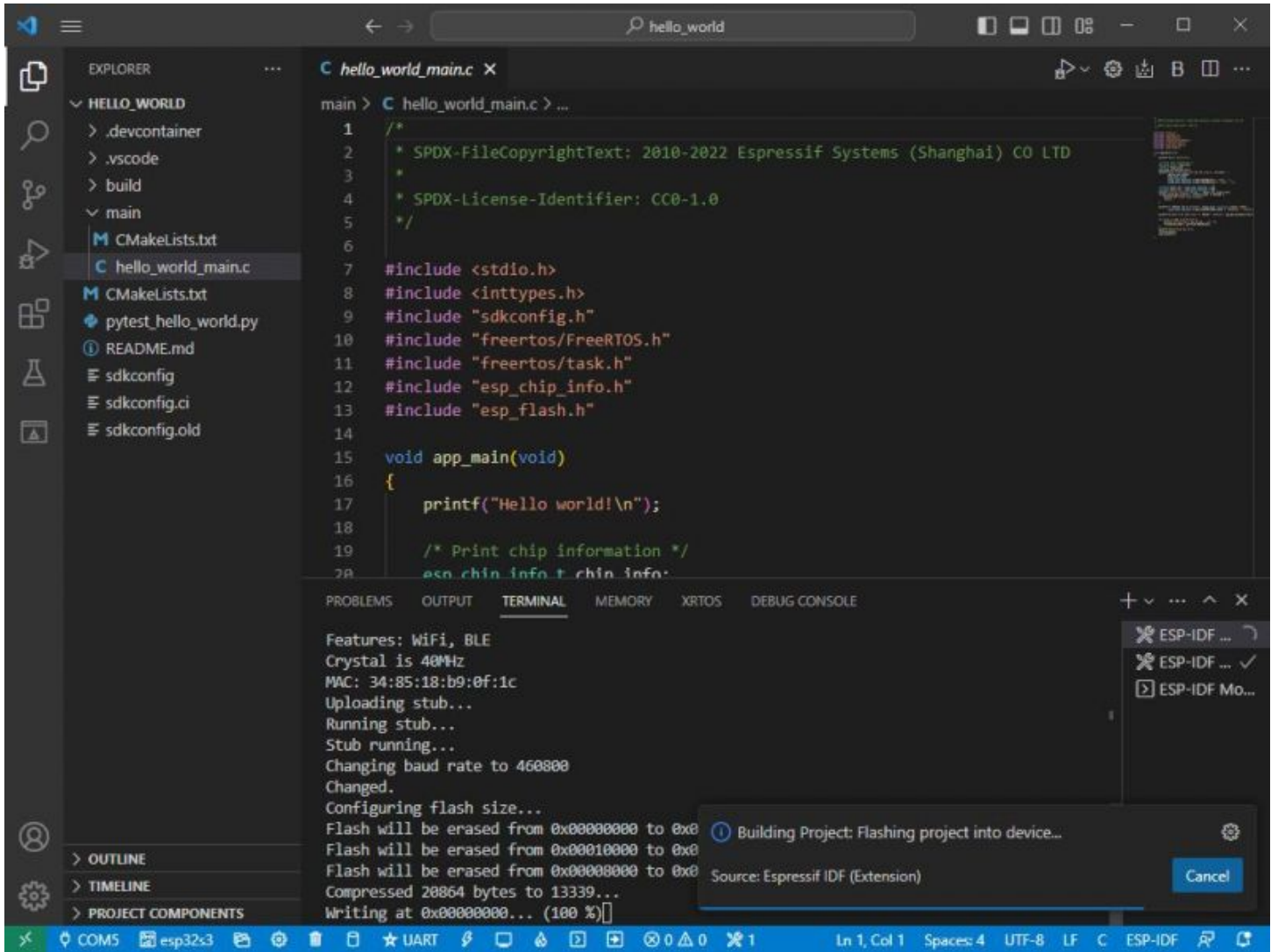
2. 编译可能需要较长时间才能完成，尤其是在第一次编译时。



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-26.jpg)

- 在此过程中，ESP-IDF可能会占用大量CPU资源，因此可能会导致系统卡顿。

3. 因为我们使用的是CH343为USB转串口芯片，并且板载自动下载电路，无需手动操作即可自动下载



(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-27.jpg)

4. 下载成功后，自动进入串口监视器，可以看到芯片输出对应的信息并提示10S后重启

```
main > C hello_world_main.c > ...
1  /*
2  * SPDX-FileCopyrightText: 2010-2022 Espressif Systems (Shanghai) CO LTD
3  *
4  * SPDX-License-Identifier: CC0-1.0
5  */
6
7  #include <stdio.h>
8  #include <inttypes.h>
9  #include "sdkconfig.h"
10 #include "freertos/FreeRTOS.h"
11 #include "freertos/task.h"
12 #include "esp_chip_info.h"
13 #include "esp_flash.h"
14
15 void app_main(void)
16 {
17     printf("Hello world!\n");
18
19     /* Print chip information */
20     esp_chip_info_t chip_info;
```

```
I (244) heap_init: At 3FCF0000 len 00008000 (32 KiB): DRAM
I (250) heap_init: At 600FE010 len 00001FF0 (7 KiB): RTCRAM
I (257) spi_flash: detected chip: winbond
I (261) spi_flash: flash io: dio
W (265) spi_flash: Detected size(16384k) larger than the size in the binary image header(
2048k). Using the size in the binary image header.
I (279) cpu_start: Starting scheduler on PRO CPU.
I (0) cpu_start: Starting scheduler on APP CPU.
Hello world!
This is esp32s3 chip with 2 CPU core(s), WiFi/BLE, silicon revision v0.2, 2MB external fl
ash
Minimum free heap size: 391944 bytes
Restarting in 10 seconds...
```

(/wiki/%E6%96%87%E4%BB%B6:Esp32-vsod-28.jpg)

Arduino

- 如果没用使用过arduino-esp32的基础建议仔细阅读官方文档，点击此处查看 (<https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html>)

安装Arduino IDE

1. 打开官网软件下载页面 (<https://www.arduino.cc/en/software>),选择对应的系统和系统位数下载

Downloads



 **Arduino IDE 2.0.4**

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through the section below.

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

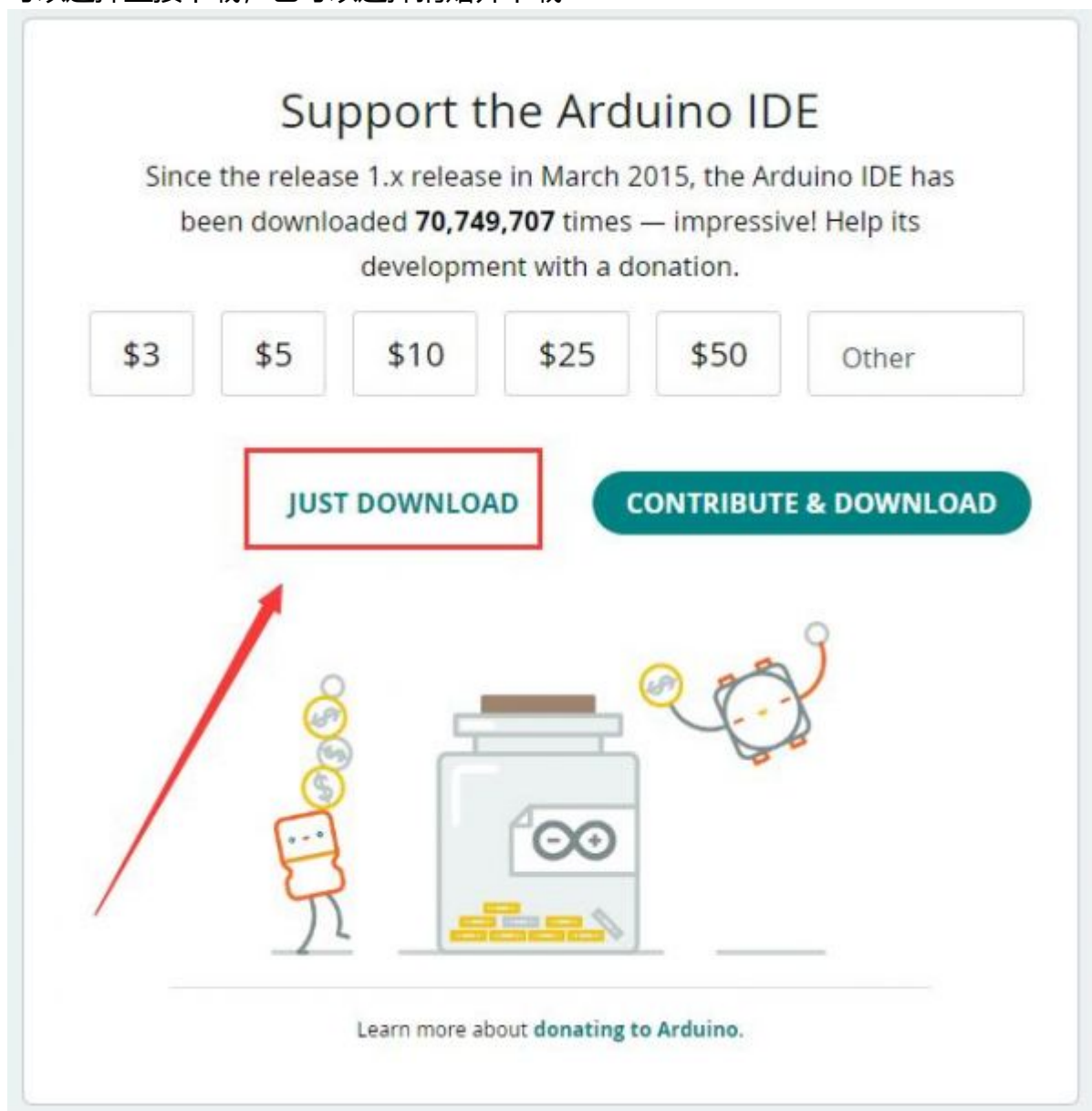
DOWNLOAD OPTIONS

- Windows** Win 10 and newer, 64 bits
- Windows** MSI Installer
- Windows** ZIP file
- Linux** Applmage 64 bits (X86-64)
- Linux** ZIP file 64 bits (X86-64)
- macOS** Intel, 10.14: "Mojave" or newer, 64 bits
- macOS** Apple Silicon, 11: "Big Sur" or newer, 64 bits

[Release Notes](#)

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-01.jpg)

2. 可以选择直接下载，也可以选择捐赠并下载



Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **70,749,707** times — impressive! Help its development with a donation.

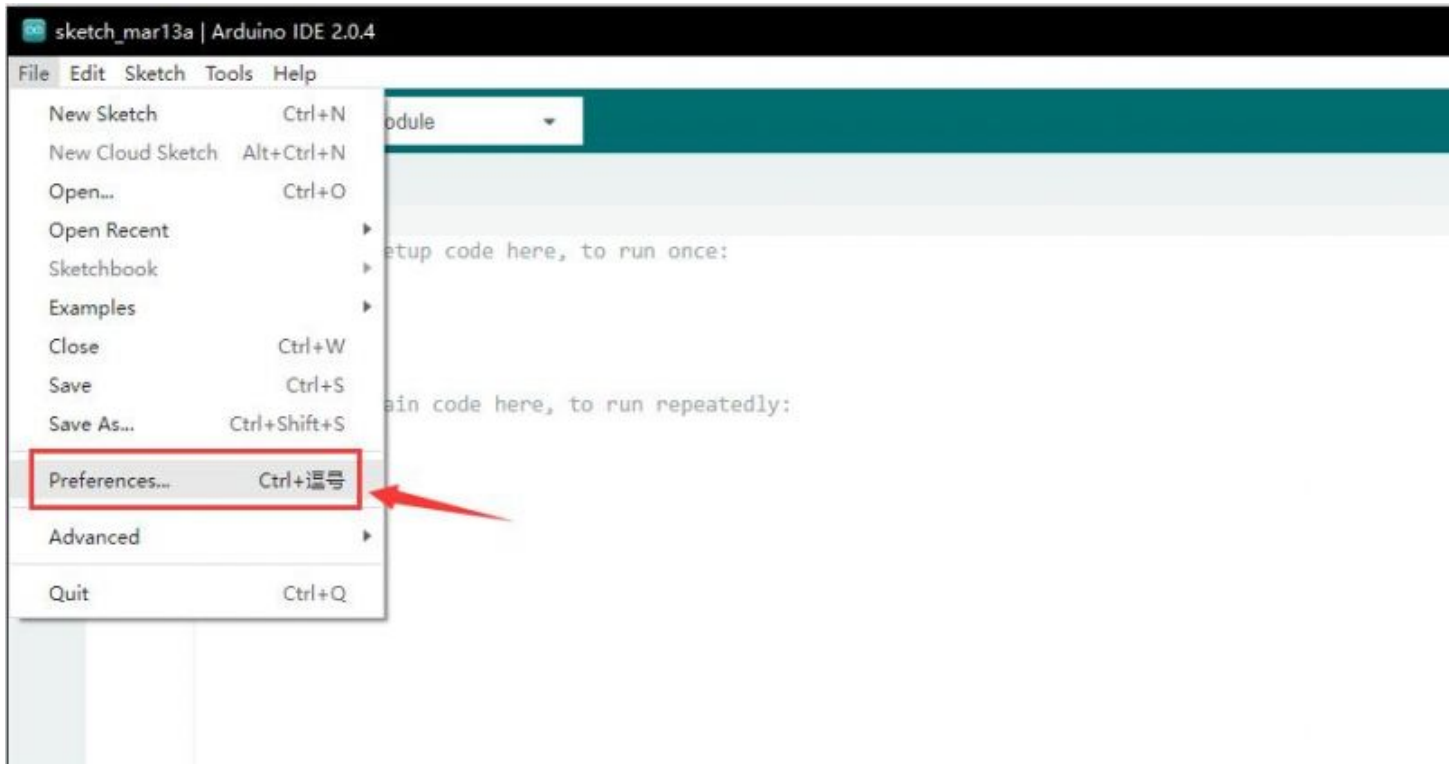
Learn more about [donating to Arduino](#).

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-02.jpg)

3. 运行安装程序，全部默认安装即可

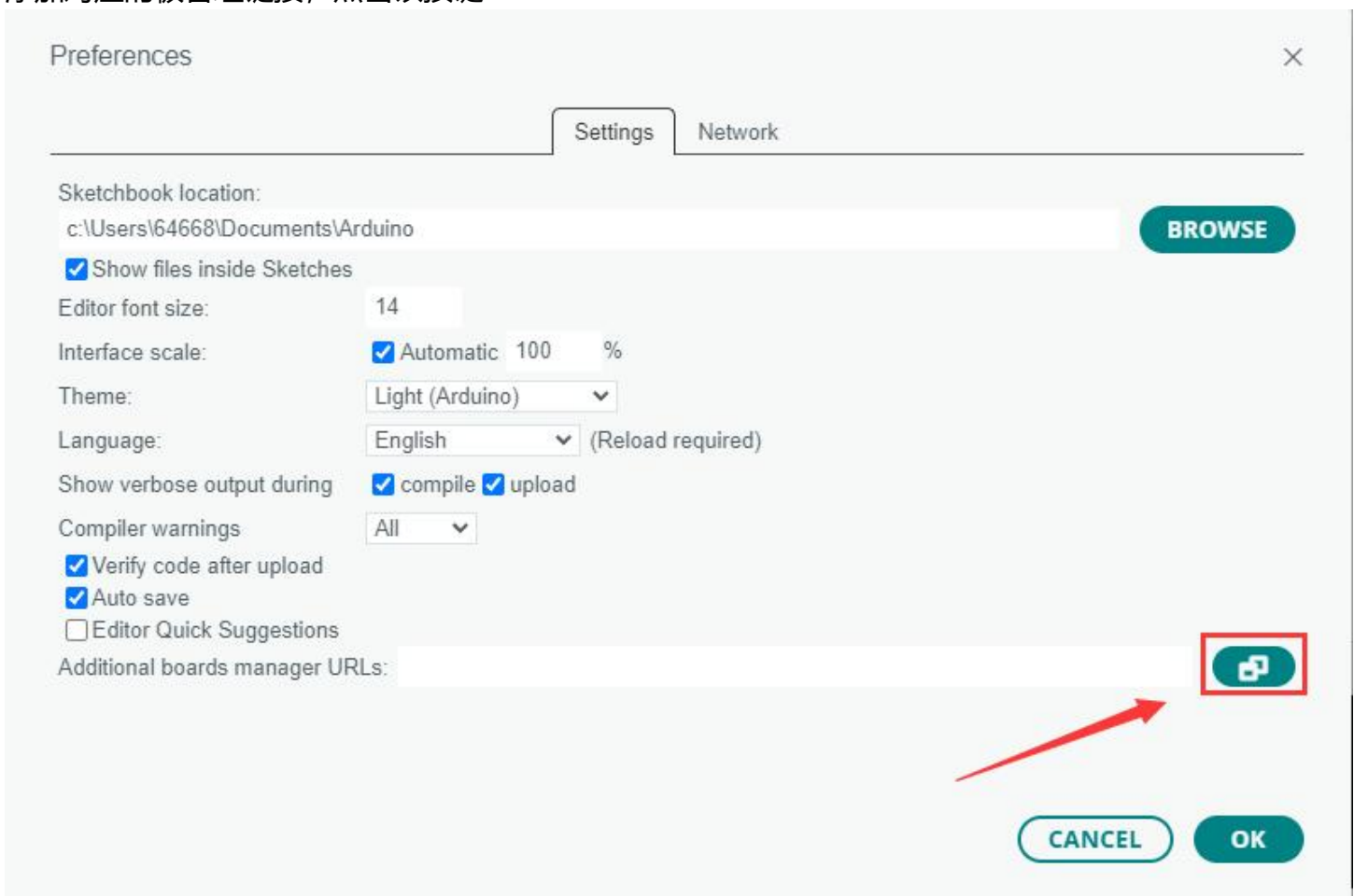
在线安装arduino-esp32

1. 打开首选项



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-03.jpg)

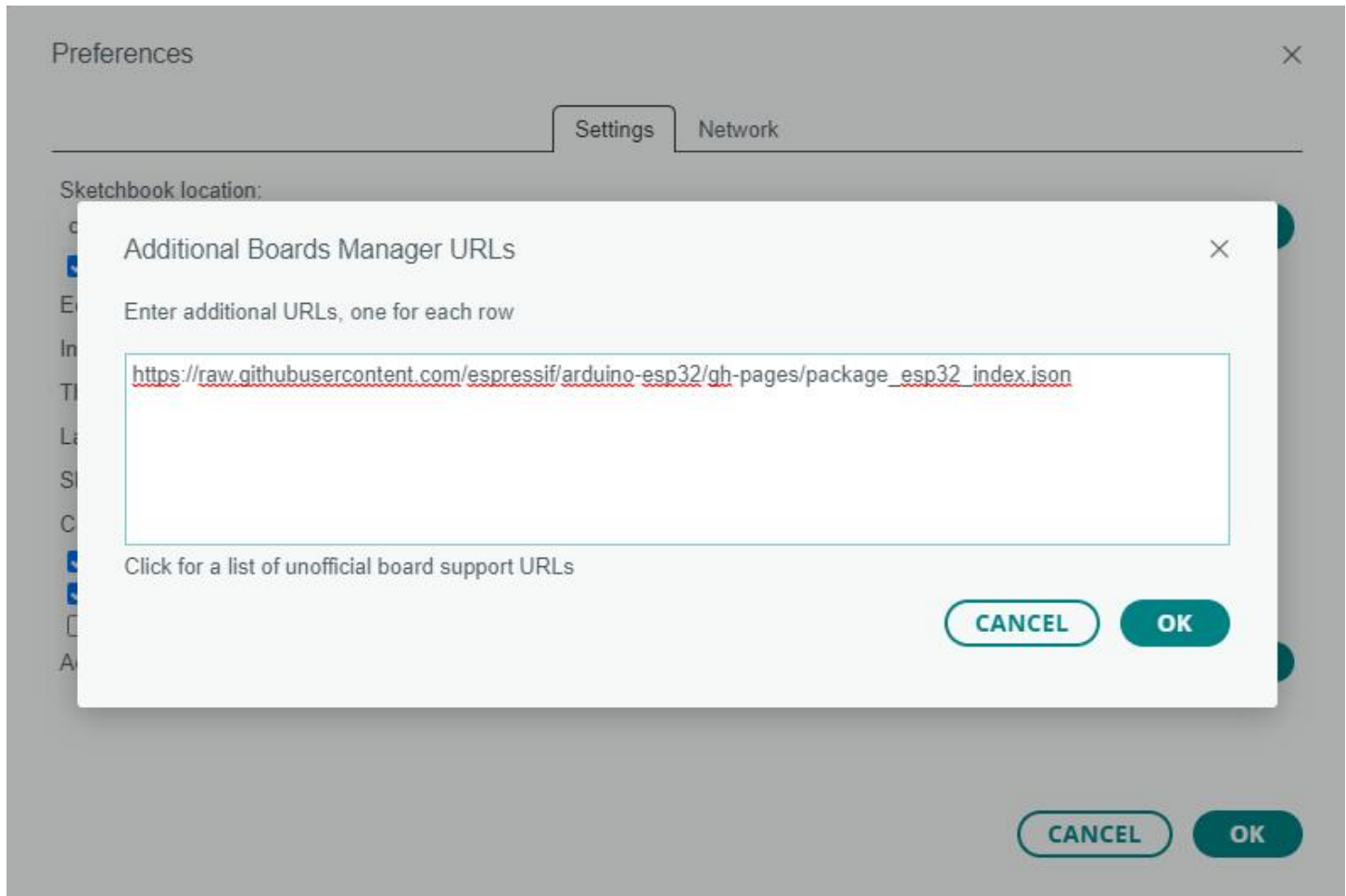
2. 添加对应的板管理链接，点击该按钮



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-04.jpg)

3. 在第一个空白处，添加下文

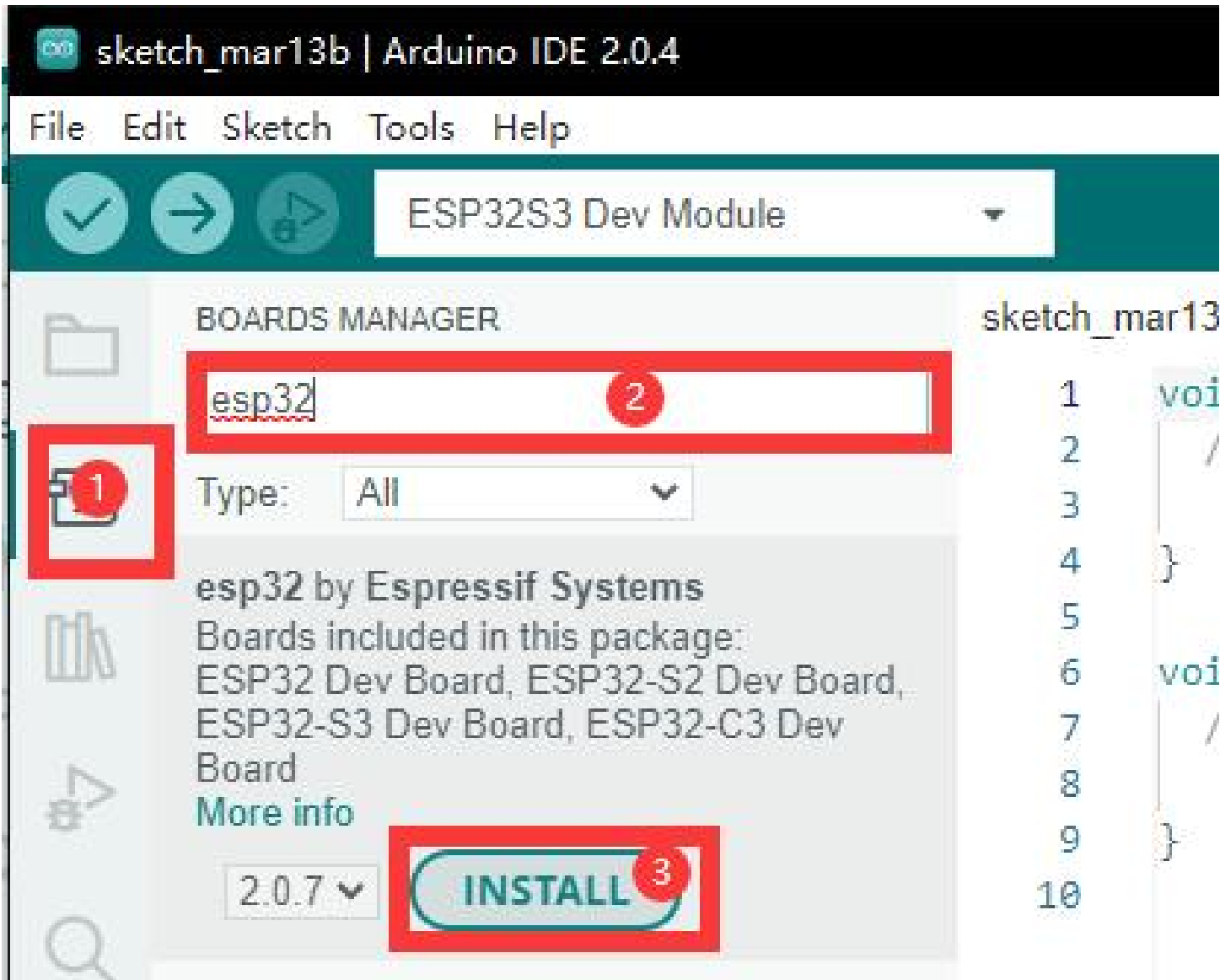
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-05.jpg)

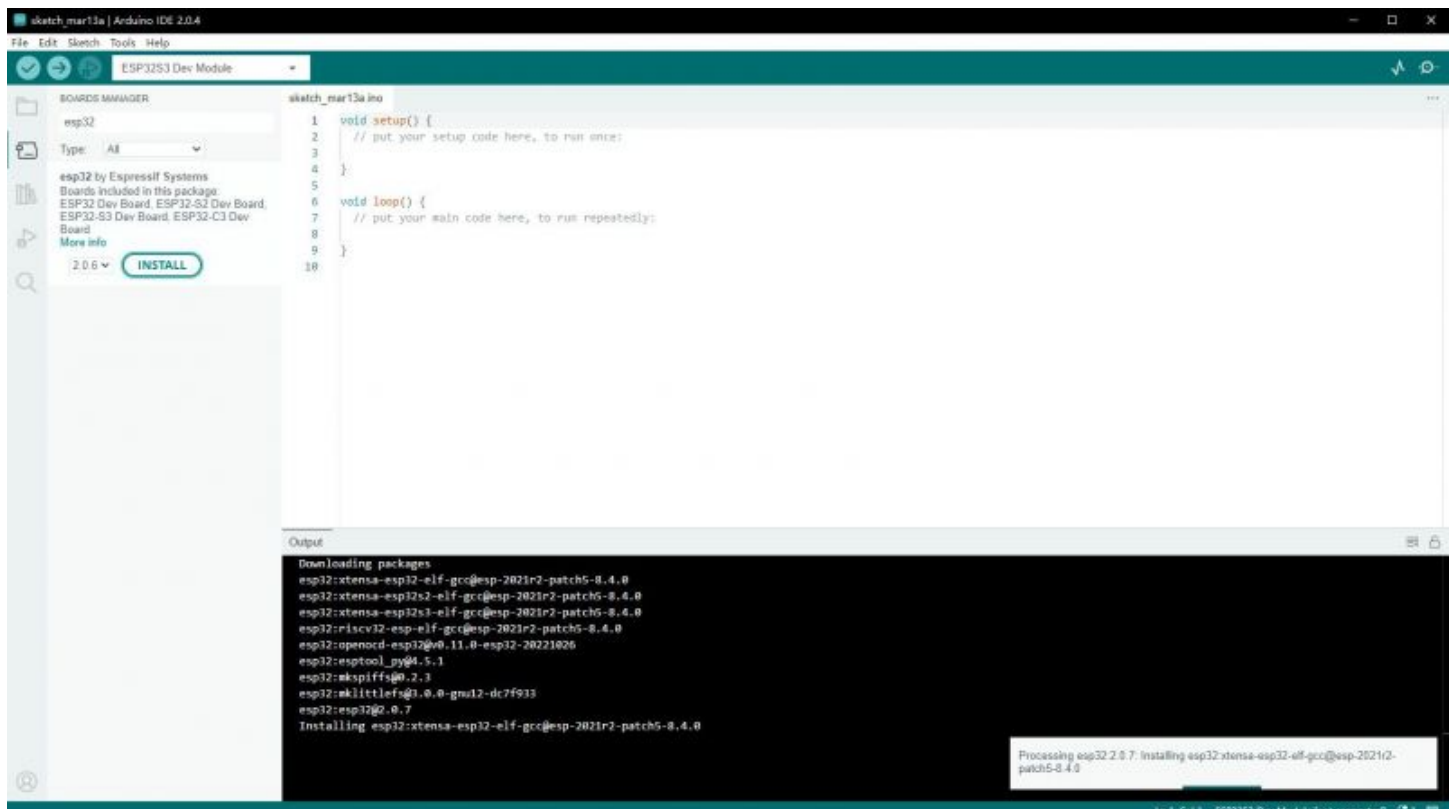
4. 保存设置

5. 打开板管理器并搜索输入ESP32



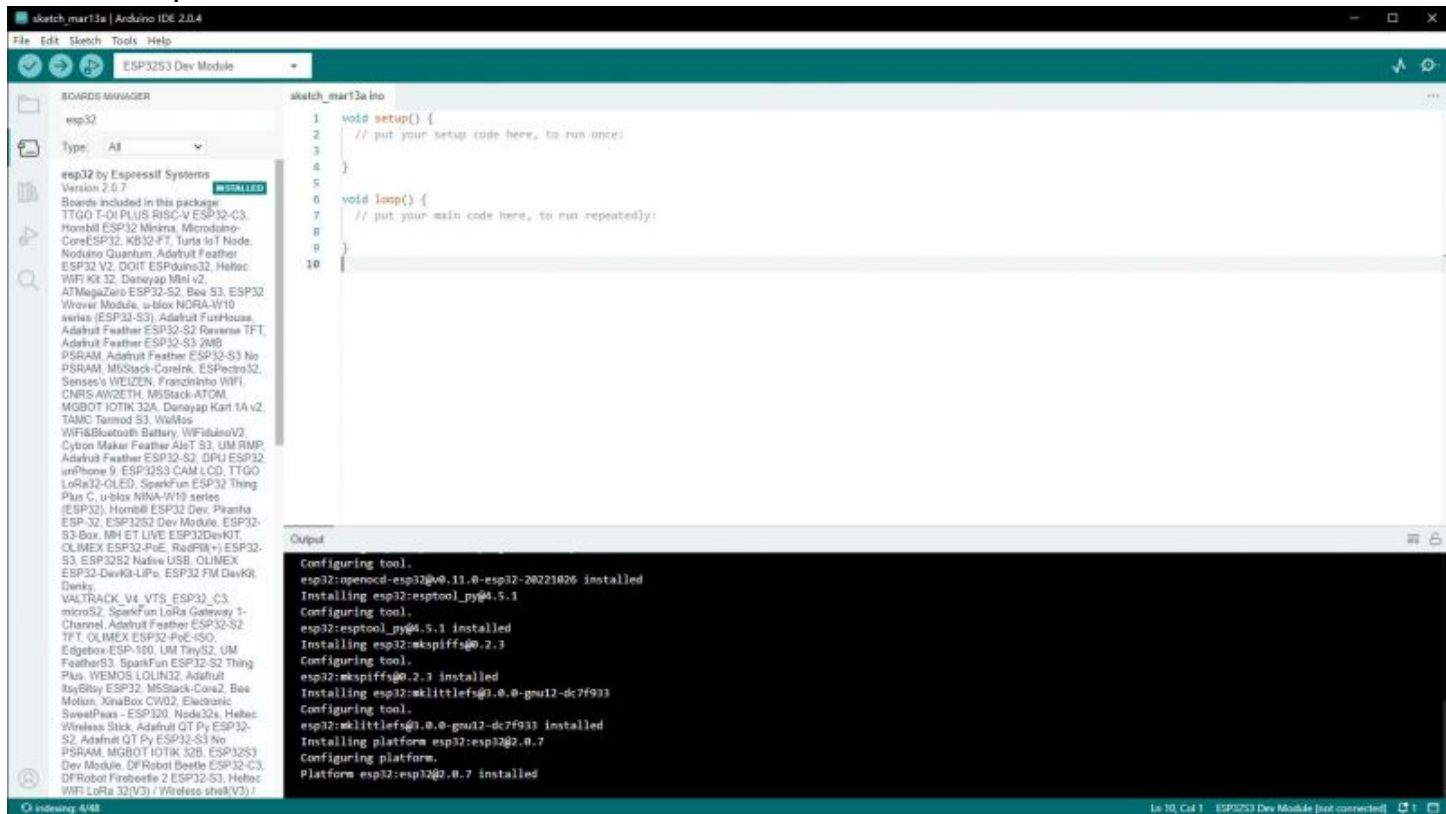
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-06.jpg)

6. 等待下载



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-07.jpg)

7. arduino-esp32下载完成



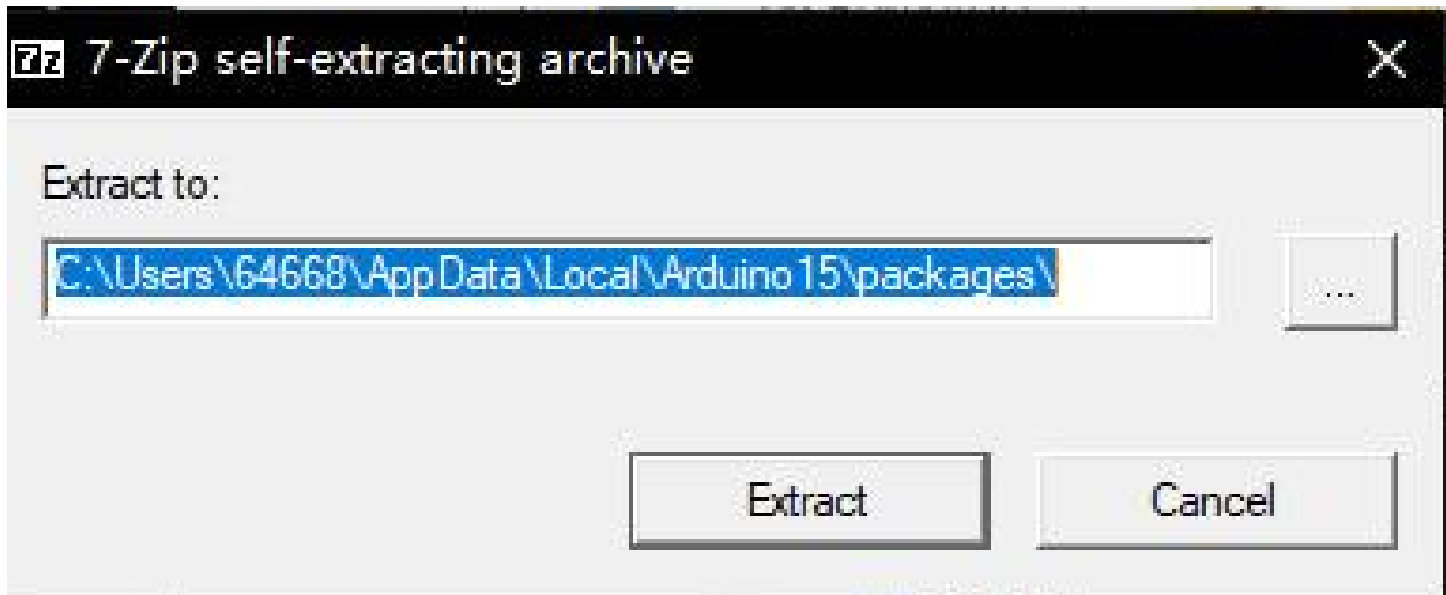
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-08.jpg)

离线安装arduino-esp32（国内推荐）

1. 离线包（提取码：1r4i） (<https://www.aliyundrive.com/s/a1wzmiRkZNL>)

- 提取码：1r4i

2. 将压缩包解压缩



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-18.jpg)

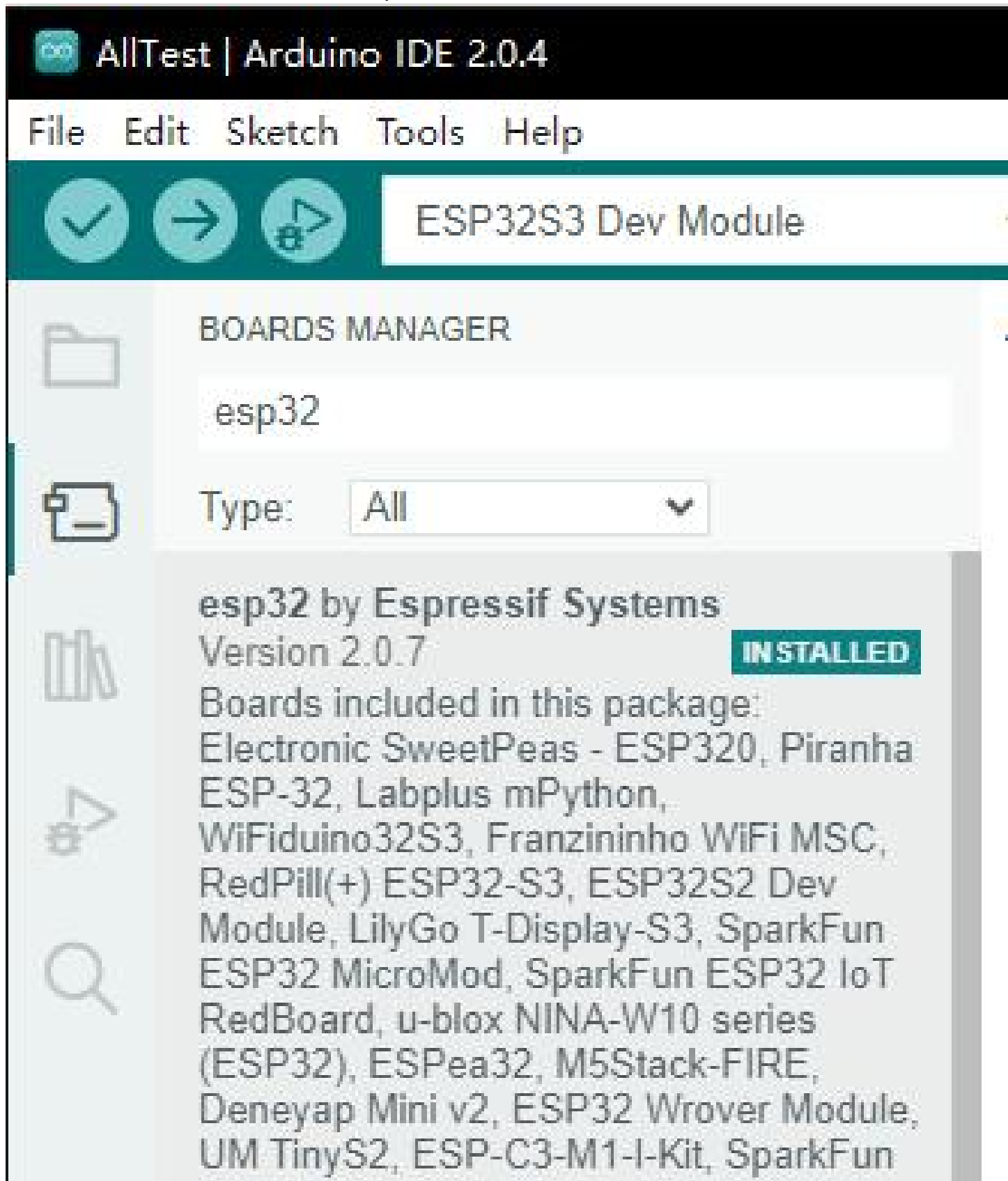
3. 将解压文件放在对应用户的arduino器件包目录

C:\Users\{用户名}\AppData\Local\Arduino15\packages\

以用户名为waveshare为例

C:\Users\waveshare\AppData\Local\Arduino15\packages\

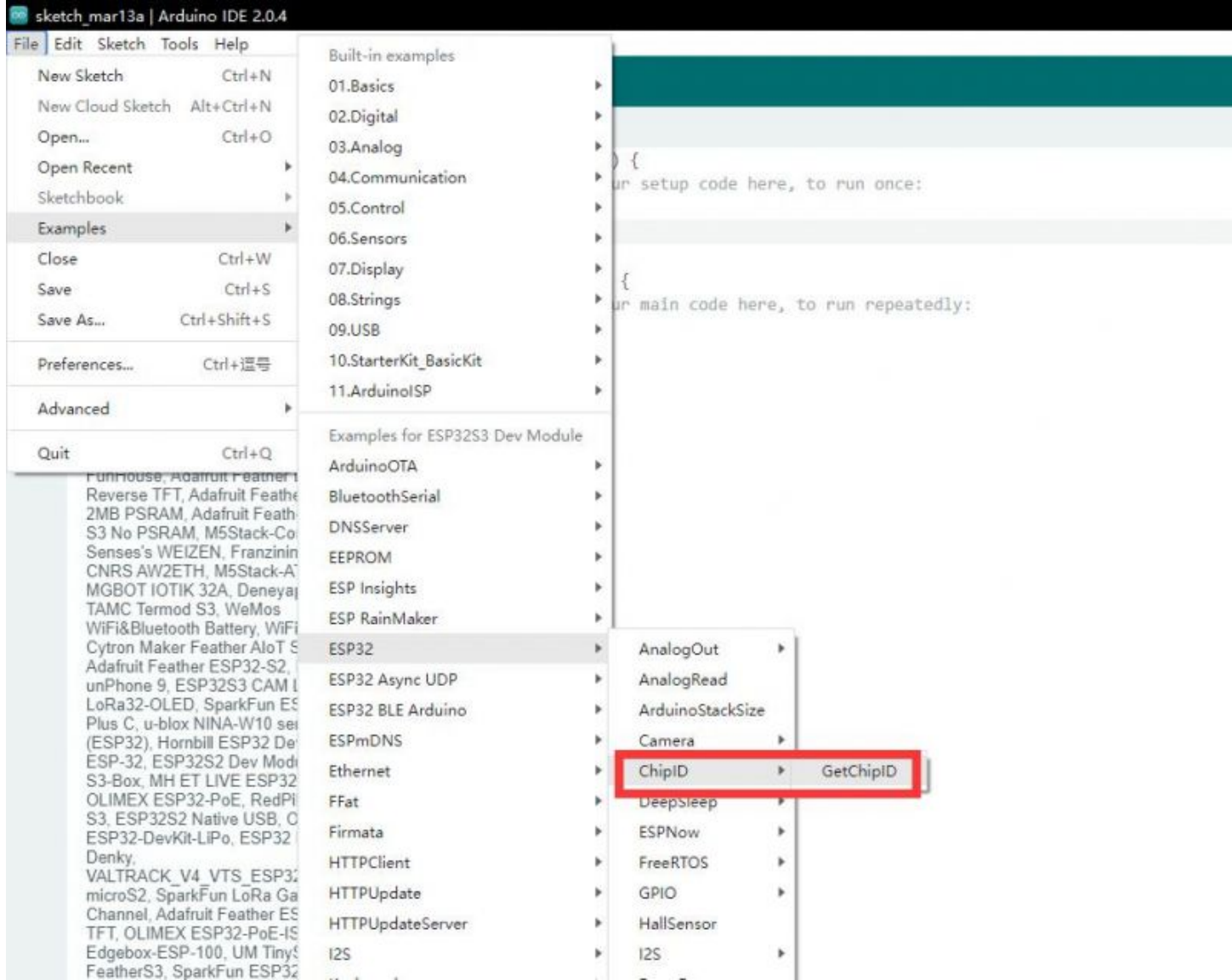
4. 关闭全部arduino窗口，确保arduino关闭
5. 打开arduino，并打开板管理器,看到esp32-arduino已经安装即可



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-19.jpg)

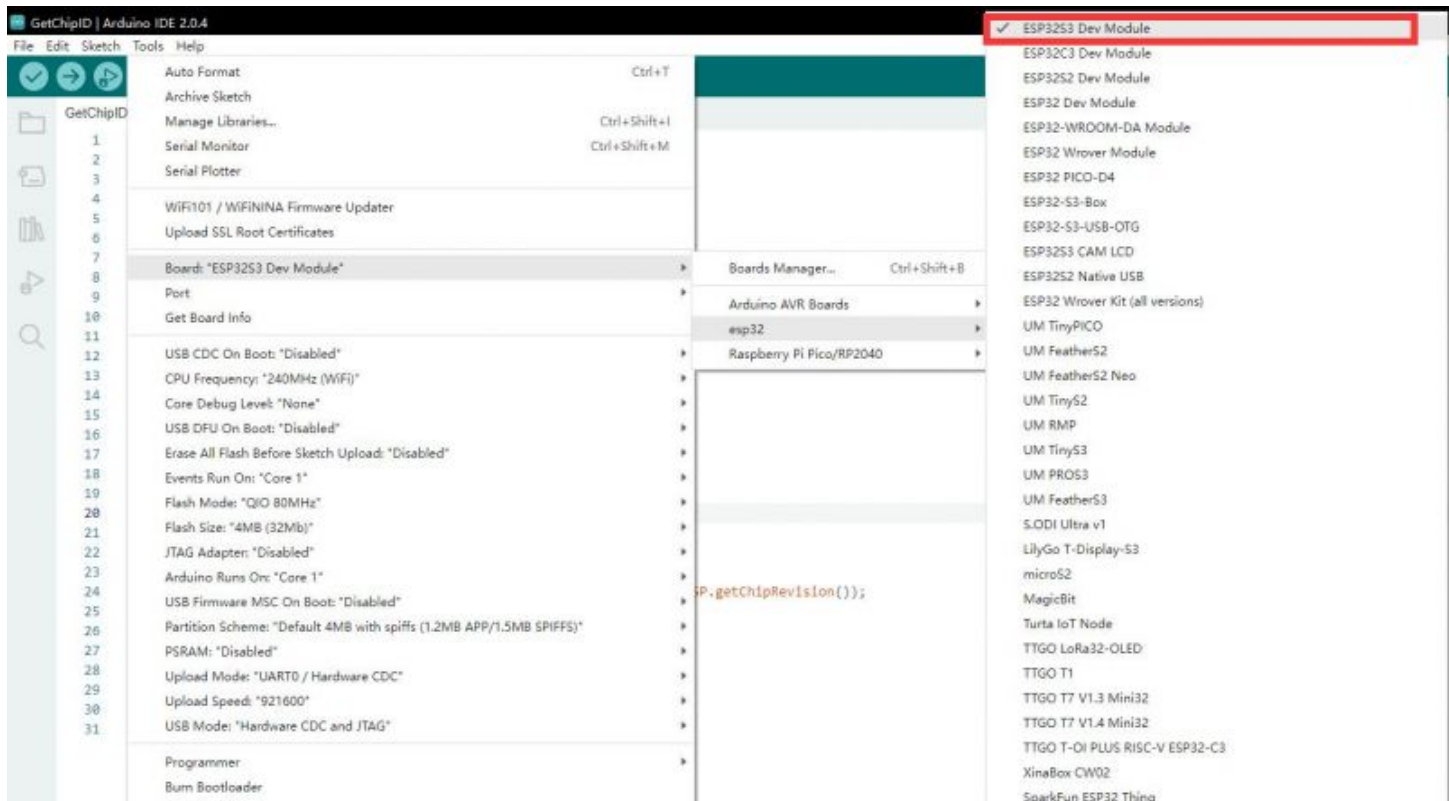
使用Arduino例程

1. 选择例程，这里我们选择获取芯片ID的例程



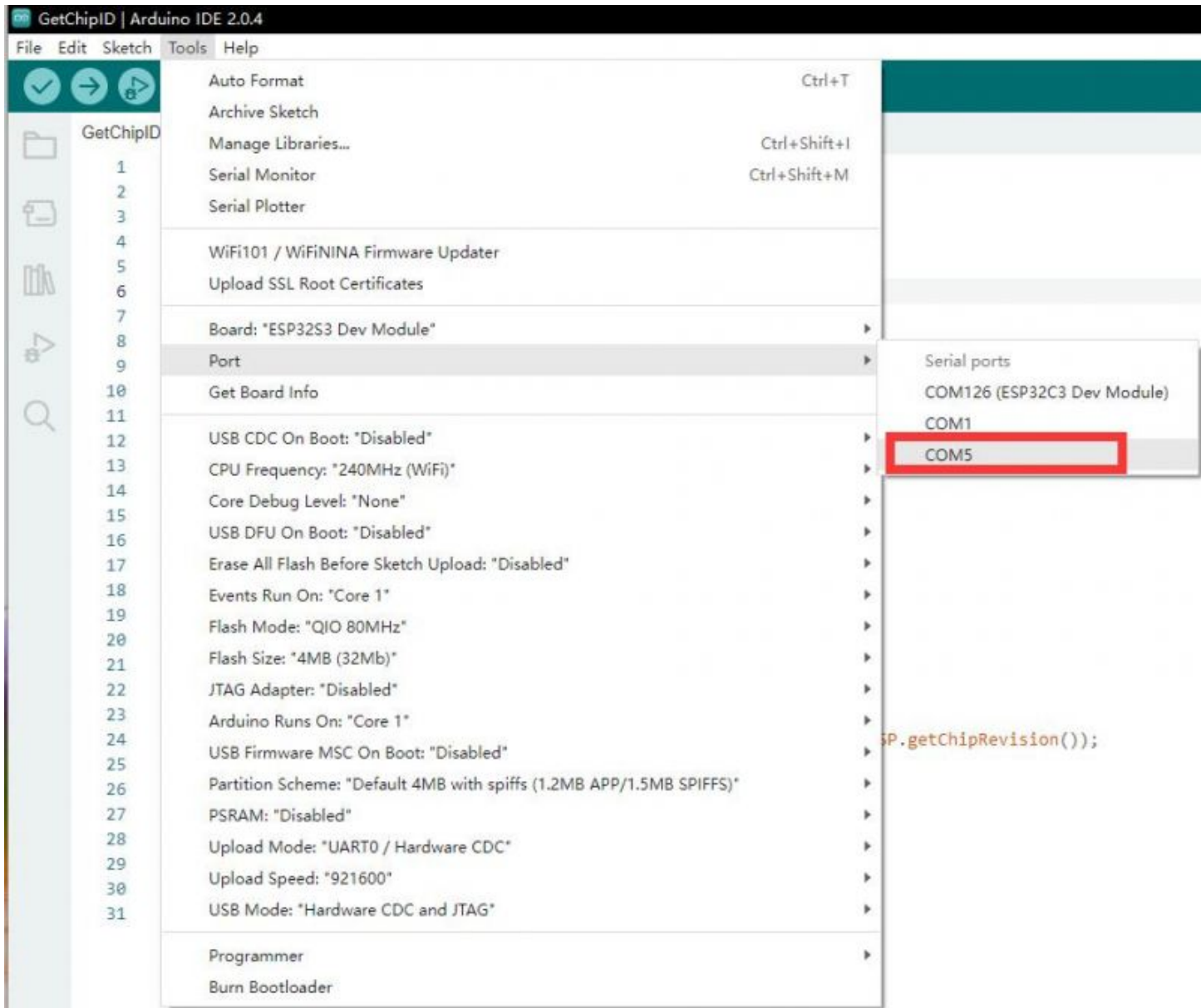
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-09.jpg)

2. 选择我们的板子为 ESP32S3 Dev Moudule



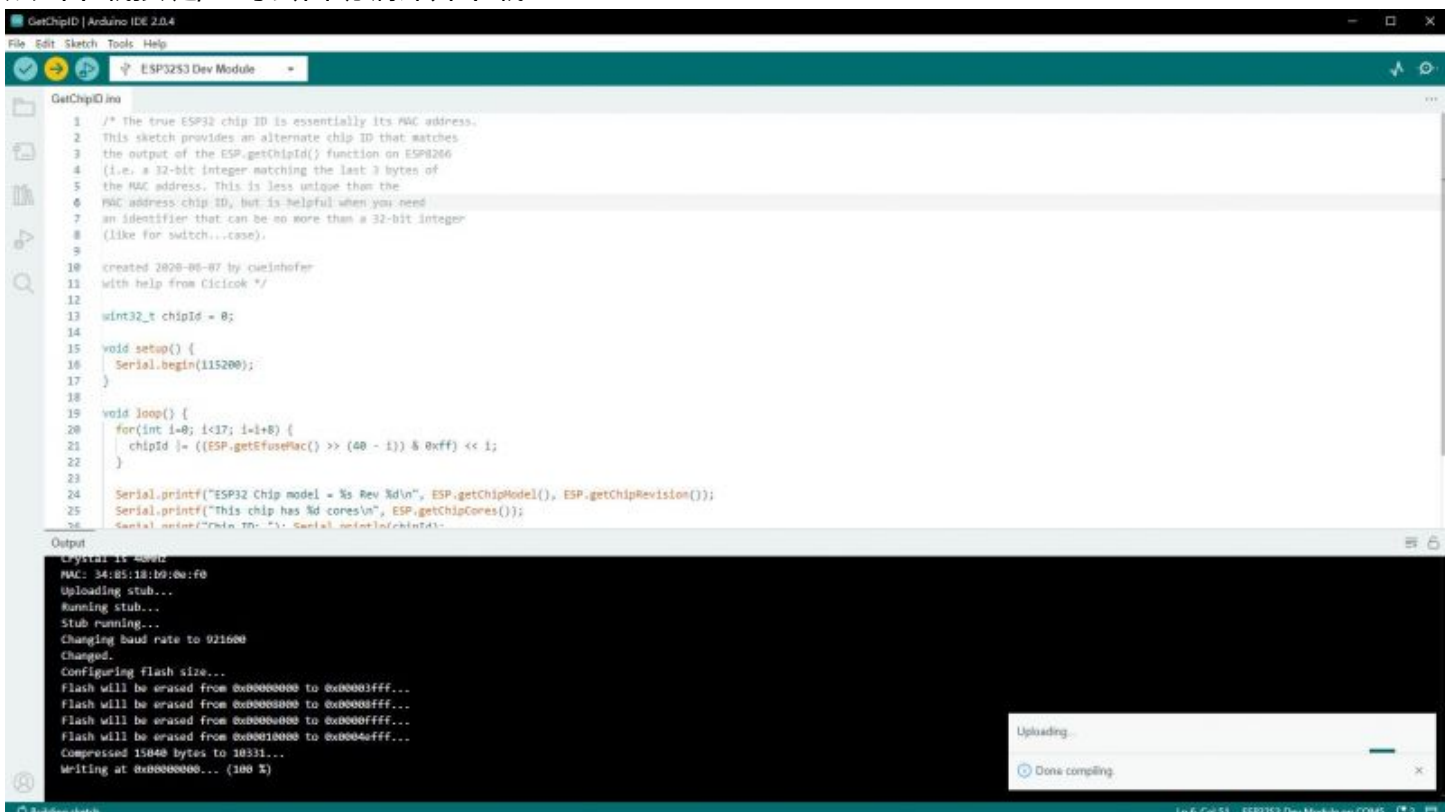
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-12.jpg)

3. 选择我们的端口号, 这里我们选择CH343的COM5



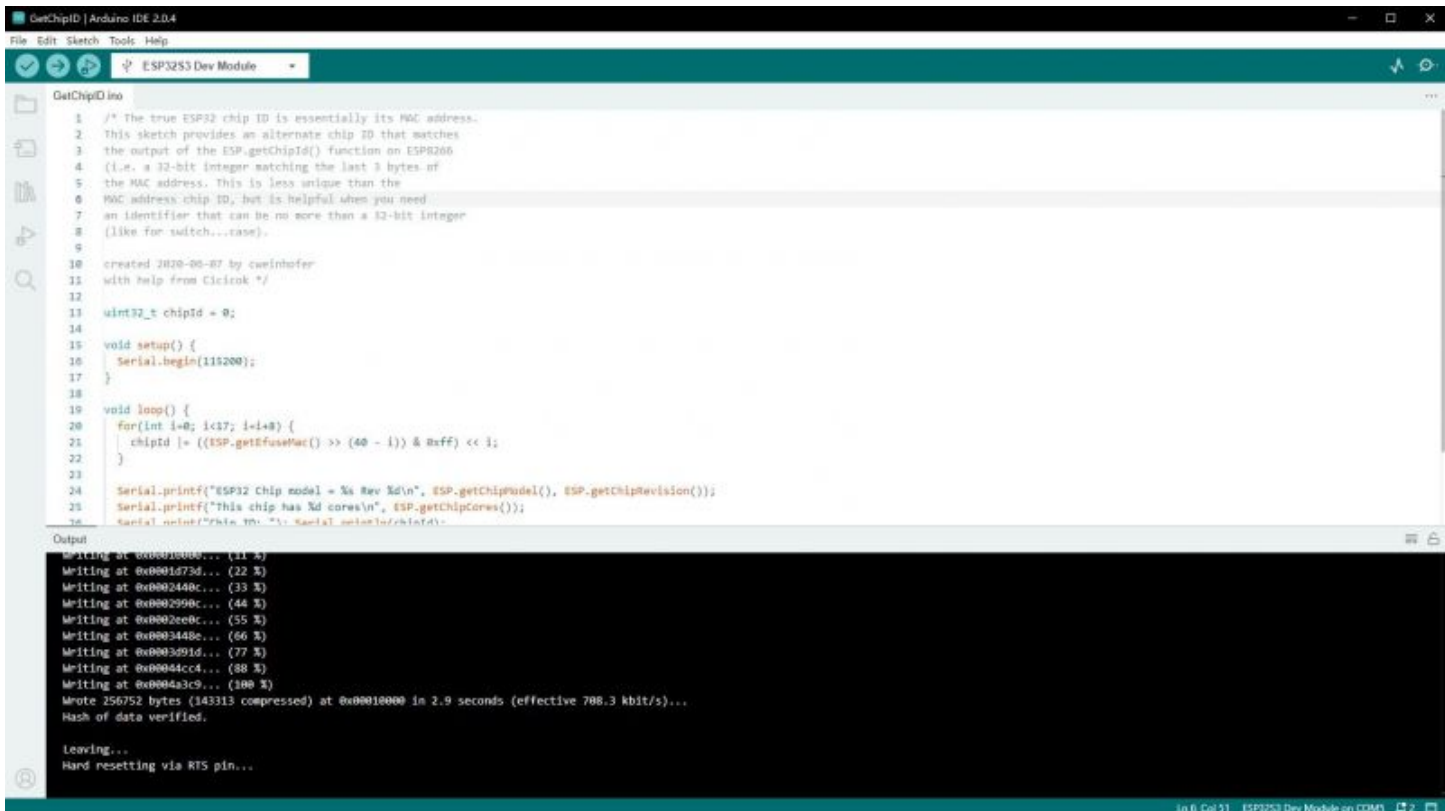
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-13.jpg)

4. 点击下载按钮, 此时会自动编译并下载



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-14.jpg)

5. 下载完成



The screenshot shows the Arduino IDE 2.0.4 interface. The main window displays the 'GetChipID.ino' sketch. The code is as follows:

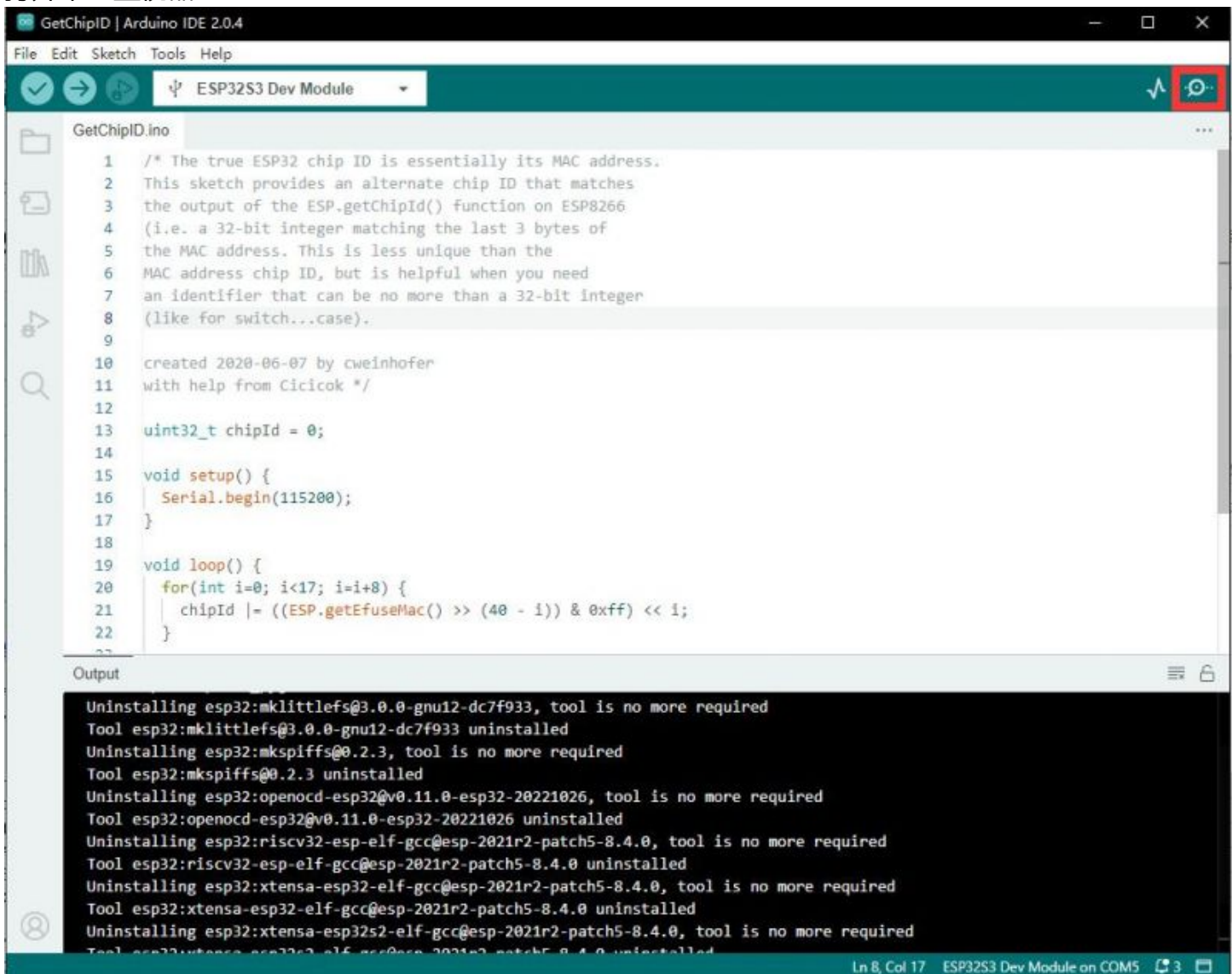
```
1 /* The true ESP32 chip ID is essentially its MAC address.
2 This sketch provides an alternate chip ID that matches
3 the output of the ESP.getChipId() function on ESP8266
4 (i.e. a 32-bit integer matching the last 3 bytes of
5 the MAC address. This is less unique than the
6 MAC address chip ID, but is helpful when you need
7 an identifier that can be no more than a 32-bit integer
8 (like for switch...case).
9
10 created 2020-06-07 by cweinhofer
11 with help from Cicicok */
12
13 uint32_t chipId = 0;
14
15 void setup() {
16   Serial.begin(115200);
17 }
18
19 void loop() {
20   for(int i=0; i<17; i=i+8) {
21     chipId |= ((ESP.getEfuseMac() >> (40 - i)) & 0xff) << i;
22   }
23
24   Serial.printf("ESP32 Chip model = %s Rev %d\n", ESP.getChipModel(), ESP.getChipRevision());
25   Serial.printf("This chip has %d cores\n", ESP.getChipCores());
26   Serial.printf("Pin %d\n", ESP.getPinCount());
27 }
```

The Output window shows the following text:

```
Writing at 0x00010000... (11 B)
Writing at 0x00010730... (22 B)
Writing at 0x0002448c... (33 B)
Writing at 0x0002998c... (44 B)
Writing at 0x0002e08c... (55 B)
Writing at 0x0003448c... (66 B)
Writing at 0x0003991d... (77 B)
Writing at 0x00044c4... (88 B)
Writing at 0x0004a3c9... (100 B)
Wrote 256752 bytes (143313 compressed) at 0x00010000 in 2.9 seconds (effective 788.3 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
```

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-15.jpg)

6. 打开串口监视器



The screenshot shows the Arduino IDE 2.0.4 interface with the serial monitor open. The main window displays the 'GetChipID.ino' sketch. The code is as follows:

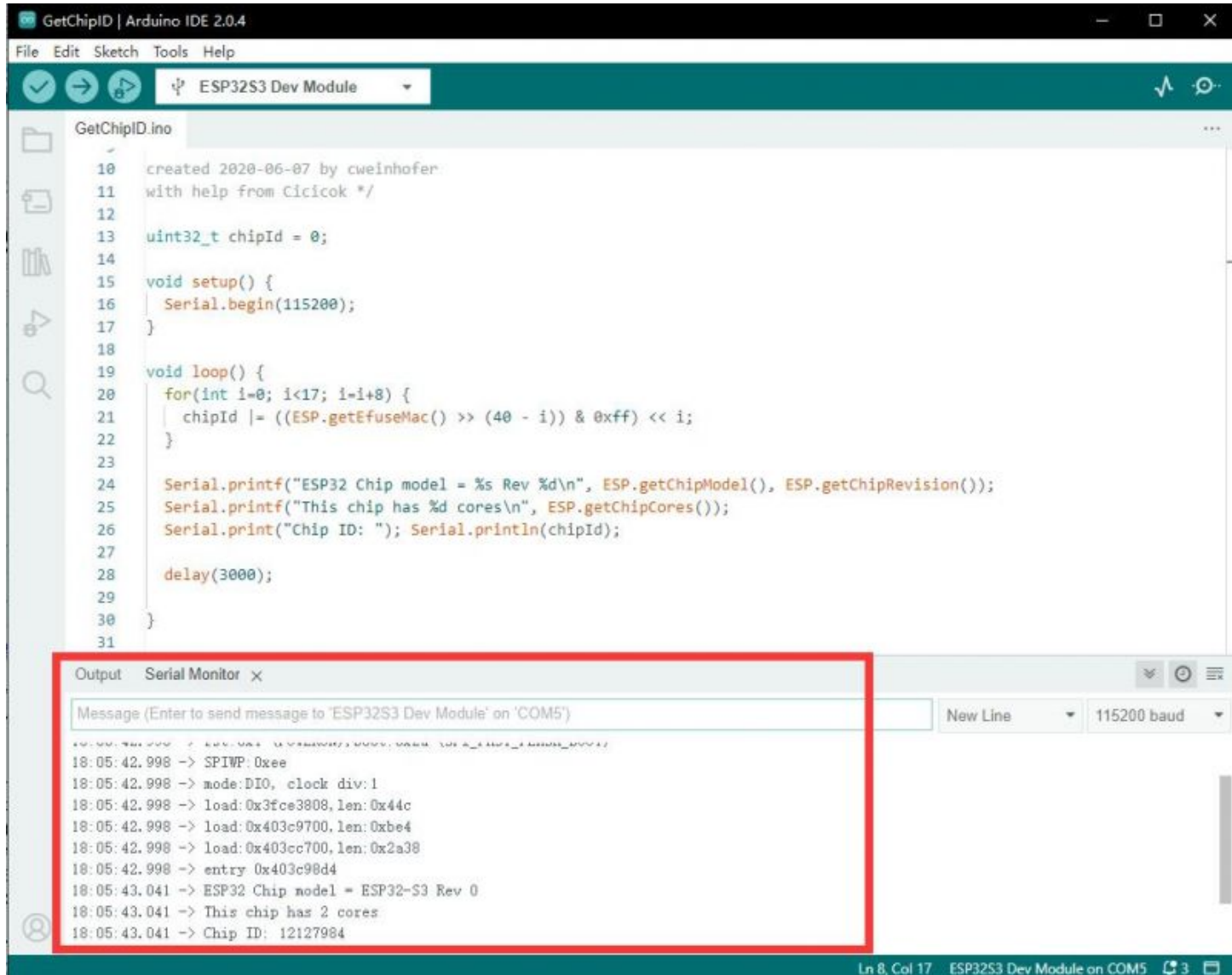
```
1 /* The true ESP32 chip ID is essentially its MAC address.
2 This sketch provides an alternate chip ID that matches
3 the output of the ESP.getChipId() function on ESP8266
4 (i.e. a 32-bit integer matching the last 3 bytes of
5 the MAC address. This is less unique than the
6 MAC address chip ID, but is helpful when you need
7 an identifier that can be no more than a 32-bit integer
8 (like for switch...case).
9
10 created 2020-06-07 by cweinhofer
11 with help from Cicicok */
12
13 uint32_t chipId = 0;
14
15 void setup() {
16   Serial.begin(115200);
17 }
18
19 void loop() {
20   for(int i=0; i<17; i=i+8) {
21     chipId |= ((ESP.getEfuseMac() >> (40 - i)) & 0xff) << i;
22   }
23 }
```

The serial monitor output shows the following text:

```
Uninstalling esp32:mklittlefs@3.0.0-gnu12-dc7f933, tool is no more required
Tool esp32:mklittlefs@3.0.0-gnu12-dc7f933 uninstalled
Uninstalling esp32:mkspliffs@0.2.3, tool is no more required
Tool esp32:mkspliffs@0.2.3 uninstalled
Uninstalling esp32:openocd-esp32@v0.11.0-esp32-20221026, tool is no more required
Tool esp32:openocd-esp32@v0.11.0-esp32-20221026 uninstalled
Uninstalling esp32:riscv32-esp-elf-gcc@esp-2021r2-patch5-8.4.0, tool is no more required
Tool esp32:riscv32-esp-elf-gcc@esp-2021r2-patch5-8.4.0 uninstalled
Uninstalling esp32:xtensa-esp32-elf-gcc@esp-2021r2-patch5-8.4.0, tool is no more required
Tool esp32:xtensa-esp32-elf-gcc@esp-2021r2-patch5-8.4.0 uninstalled
Uninstalling esp32:xtensa-esp32s2-elf-gcc@esp-2021r2-patch5-8.4.0, tool is no more required
Tool esp32:xtensa-esp32s2-elf-gcc@esp-2021r2-patch5-8.4.0 uninstalled
```

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-16.jpg)

7. 看到循环输出的芯片ID



The screenshot shows the Arduino IDE interface with the 'GetChipID.ino' file open. The code defines a chip ID by reading the ESP32's efuse MAC address and printing it along with the chip model and revision. The serial monitor shows the output of the program, including the chip ID: 12127984.

```
GetChipID.ino
10  created 2020-06-07 by cweinhofer
11  with help from Cicicok */
12
13  uint32_t chipId = 0;
14
15  void setup() {
16    Serial.begin(115200);
17  }
18
19  void loop() {
20    for(int i=0; i<17; i=i+8) {
21      chipId |= ((ESP.getEfuseMac() >> (40 - i)) & 0xff) << i;
22    }
23
24    Serial.printf("ESP32 Chip model = %s Rev %d\n", ESP.getChipModel(), ESP.getChipRevision());
25    Serial.printf("This chip has %d cores\n", ESP.getChipCores());
26    Serial.print("Chip ID: "); Serial.println(chipId);
27
28    delay(3000);
29  }
30 }
31
```

Serial Monitor Output:

```
18:05:42.998 -> SPIWP: 0xee
18:05:42.998 -> mode: DIO, clock div: 1
18:05:42.998 -> load: 0x3fce3808, len: 0x44c
18:05:42.998 -> load: 0x403c9700, len: 0xbe4
18:05:42.998 -> load: 0x403cc700, len: 0x2a38
18:05:42.998 -> entry 0x403c98d4
18:05:43.041 -> ESP32 Chip model = ESP32-S3 Rev 0
18:05:43.041 -> This chip has 2 cores
18:05:43.041 -> Chip ID: 12127984
```

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-Ar-17.jpg)

MicroPython

- 如果没有MicroPython基础，可以查看MicroPython官方文档 (<https://docs.micropython.org/en/latest/>)进行学习

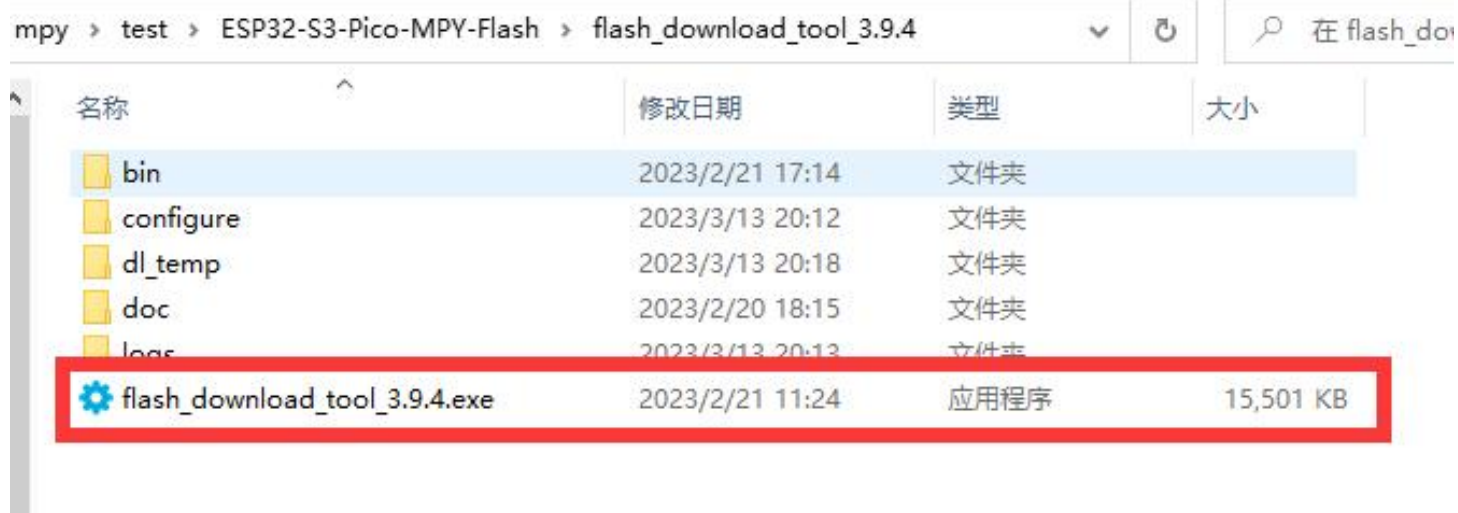
烧录固件

- 点击此处下载烧录器与固件 (/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-Flash.zip)
- 将刚刚下载的解压包解压并进入

| 名称 | 修改日期 | 类型 | 大小 |
|---------------------------|-----------------|--------|----------|
| flash_download_tool_3.9.4 | 2023/3/13 20:18 | 文件夹 | |
| micropython.bin | 2023/3/13 20:09 | BIN 文件 | 1,414 KB |

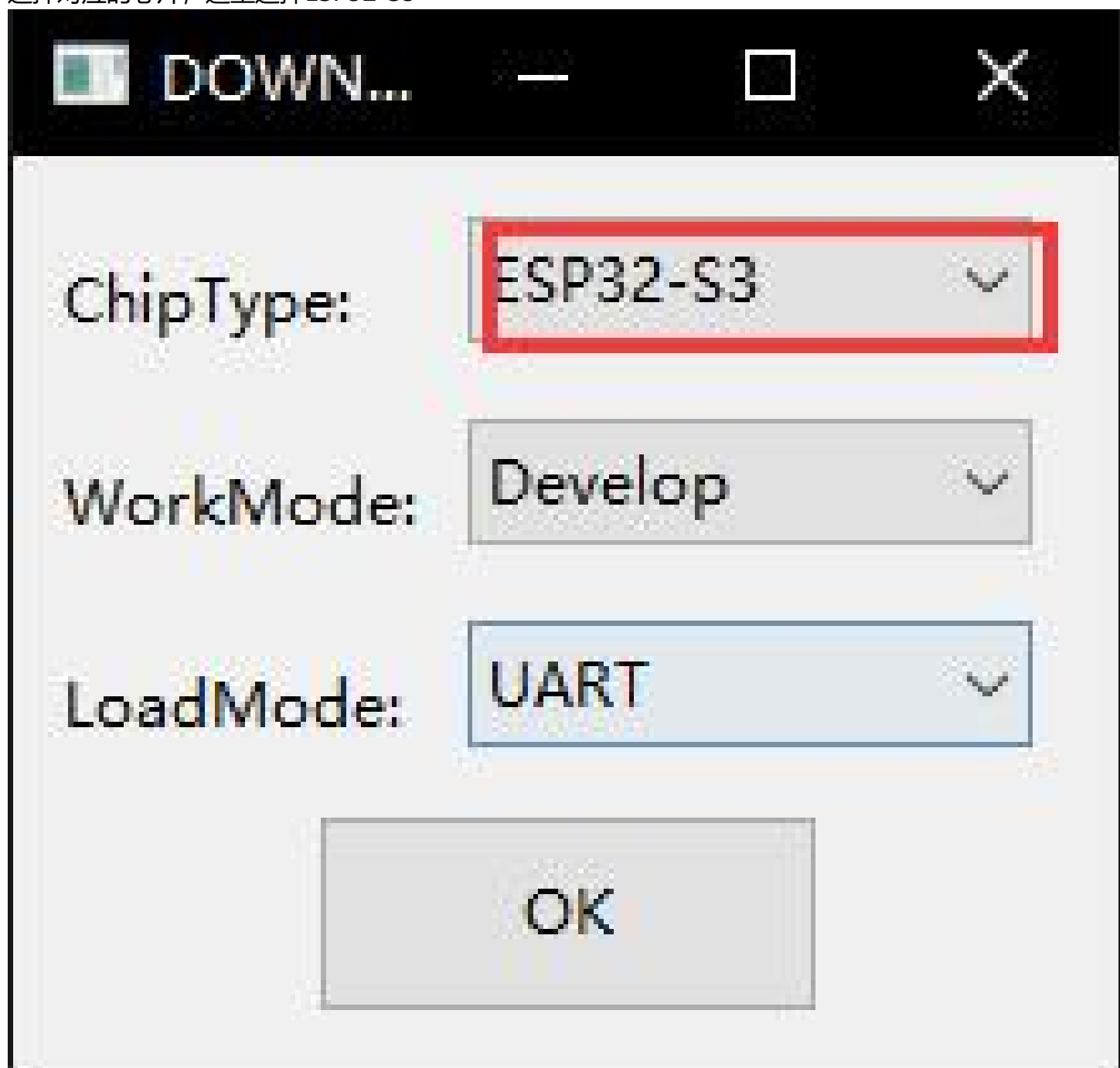
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-01.jpg)

3. 进入flash_download_tool_3.9.4文件夹，打开flash_download_tool_3.9.4.exe



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-02.jpg)

4. 选择对应的芯片，这里选择ESP32-S3



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-03.jpg)

5. 选择对应的COM口, 我们已经为你配置好其他信息, 点击Start开始下载即可

SPIDownload

| | | | | |
|-------------------------------------|--------------------|-----|---|---|
| <input checked="" type="checkbox"/> | ..\micropython.bin | ... | @ | 0 |
| <input type="checkbox"/> | | ... | @ | |
| <input type="checkbox"/> | | ... | @ | |
| <input type="checkbox"/> | | ... | @ | |
| <input type="checkbox"/> | | ... | @ | |
| <input type="checkbox"/> | | ... | @ | |
| <input type="checkbox"/> | | ... | @ | |
| <input type="checkbox"/> | | ... | @ | |

SPIFlashConfig

SPI SPEED

- 40MHz
 26.7MHz
 20MHz
 80MHz

SPI MODE

- QIO
 QOUT
 DIO
 DOUT
 FASTRD

 DoNotChgBin LockSettings

CombineBin

Default

DetectedInfo

DownloadPanel 1

IDLE
等待START ²

STOP

ERASE

COM

COM5 ¹

BAUD:

1152000

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-04.jpg)

- ① 选择COM口
- ② 下载按键

安装Thonny

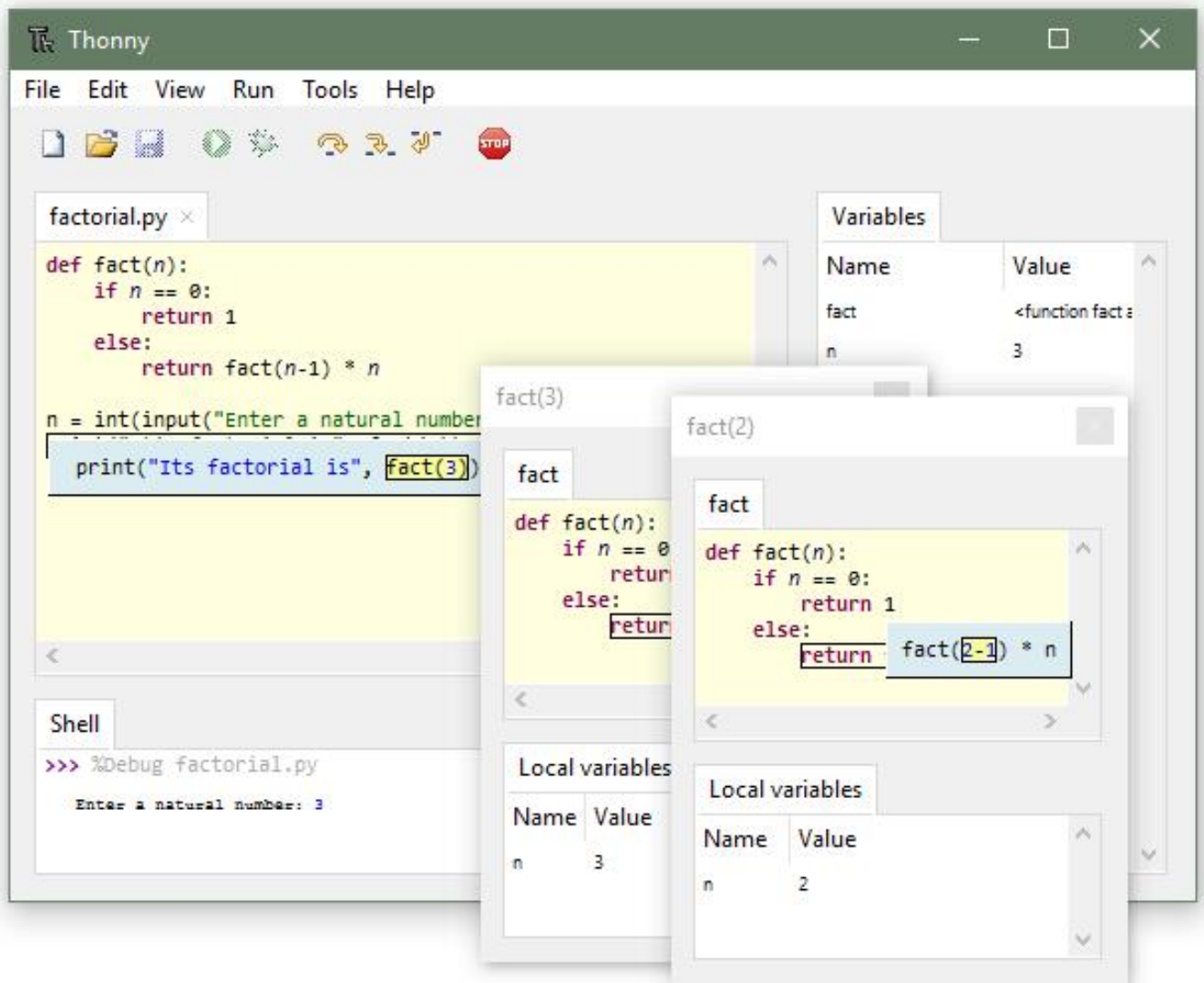
1. 打开Thonny官网 (<https://thonny.org/>)

Thonny

Python IDE for beginners



Download version [4.0.2](#) for
[Windows](#) • [Mac](#) • [Linux](#)



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-05.jpg)

2. 选择对应的系统和版本，这里我选择windows，鼠标要移动到windows处，才会显示对应的信息

Thonny



Download version **4.0.2** for
Windows • Mac • Linux

Official downloads for Windows

Installer with 64-bit Python 3.10, requires 64-bit Windows 8.1 / 10 / 11
[thonny-4.0.2.exe \(20.4 MB\)](#) ← *recommended for you*

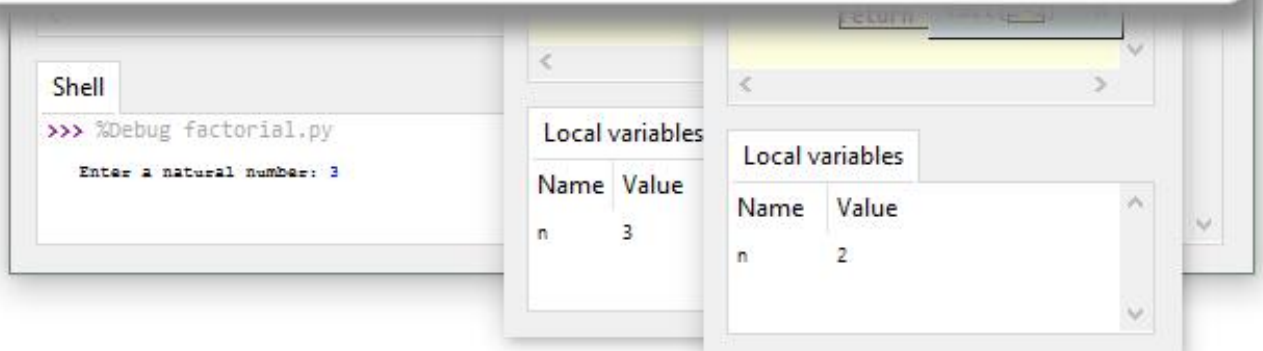
Installer with 32-bit Python 3.8, suitable for all Windows versions since 7
[thonny-py38-4.0.2.exe \(18.9 MB\)](#)

Portable variant with 64-bit Python 3.10
[thonny-4.0.2-windows-portable.zip \(30.5 MB\)](#)

Portable variant with 32-bit Python 3.8
[thonny-py38-4.0.2-windows-portable.zip \(28.6 MB\)](#)

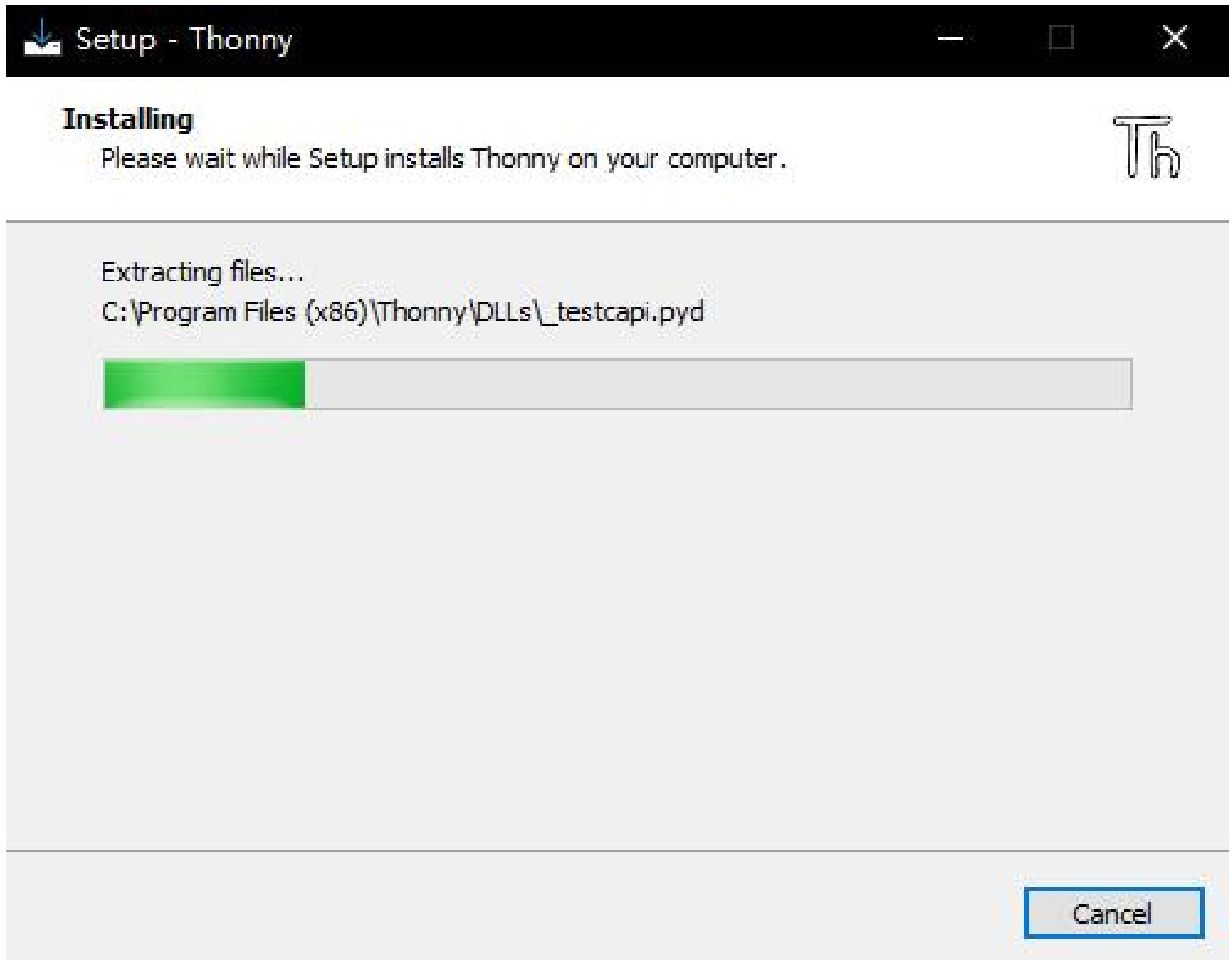
Re-using an existing Python installation (for advanced users)

```
pip install thonny
```



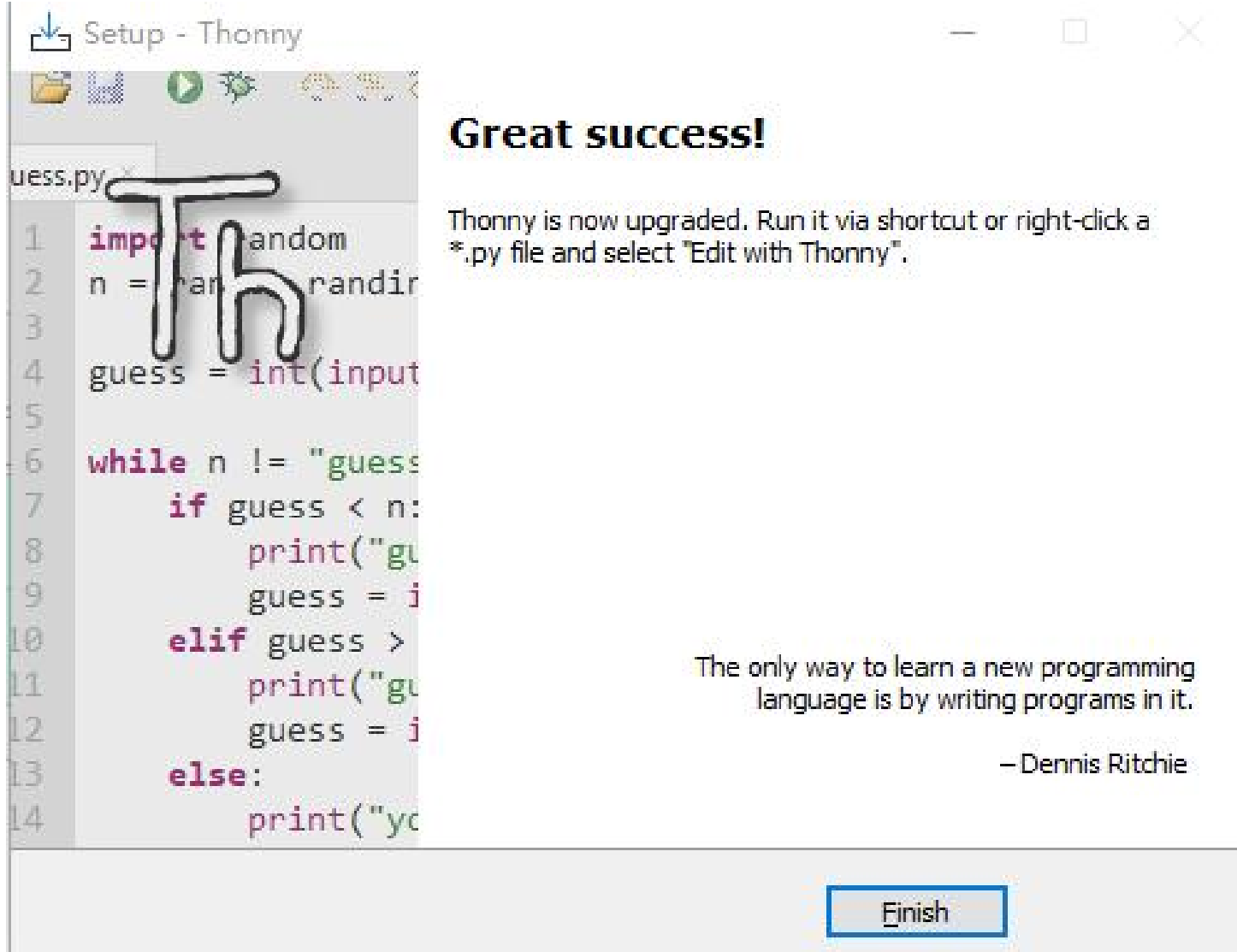
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-06.jpg)

3. 一路默认安装即可



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-07.jpg)

4. 安装完成



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-08.jpg)

获取例程

1. 点击此处下载例程 (/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-Example.zip)

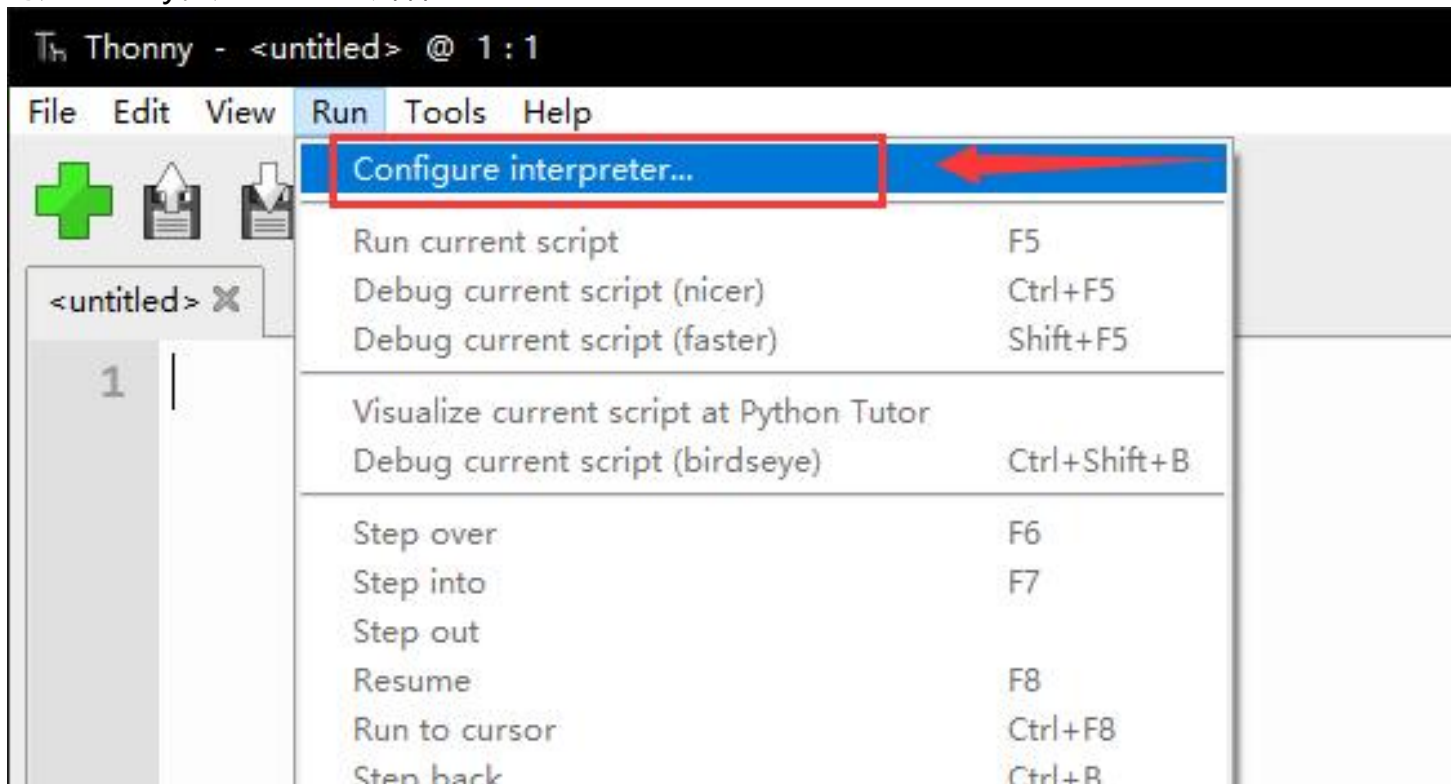
2. 解压例程压缩包

| | | | |
|------------|-----------------|--------------|------|
| 0-Firmware | 2023/3/15 19:01 | 文件夹 | |
| 1-GPIO | 2023/3/13 21:06 | 文件夹 | |
| 2-PWM | 2023/3/14 19:13 | 文件夹 | |
| 3-UART | 2023/3/15 17:30 | 文件夹 | |
| 4-I2C | 2023/3/15 16:07 | 文件夹 | |
| 5-SPI | 2023/3/15 17:58 | 文件夹 | |
| 6-ADC | 2023/3/15 16:26 | 文件夹 | |
| 7-RGB-LED | 2023/3/13 21:00 | 文件夹 | |
| 8-SYS | 2023/3/15 11:35 | 文件夹 | |
| 9-WIFI | 2023/3/15 17:50 | 文件夹 | |
| LICENSE | 2023/3/15 18:44 | 文件 | 2 KB |
| README.md | 2023/3/15 19:03 | Markdown 源文件 | 1 KB |

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-09.jpg)

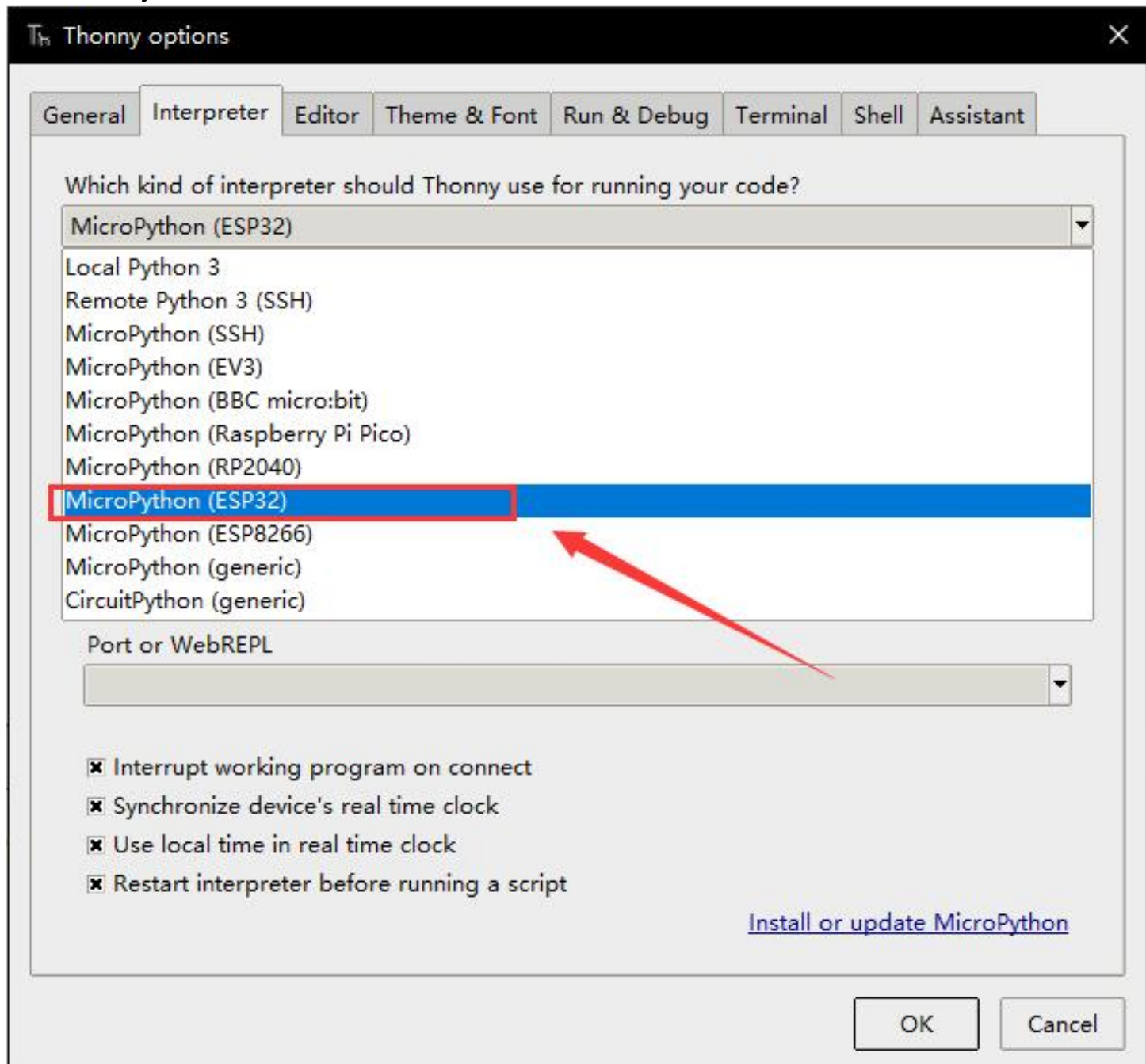
运行例程

1. 打开thonny, 并选择配置解释器



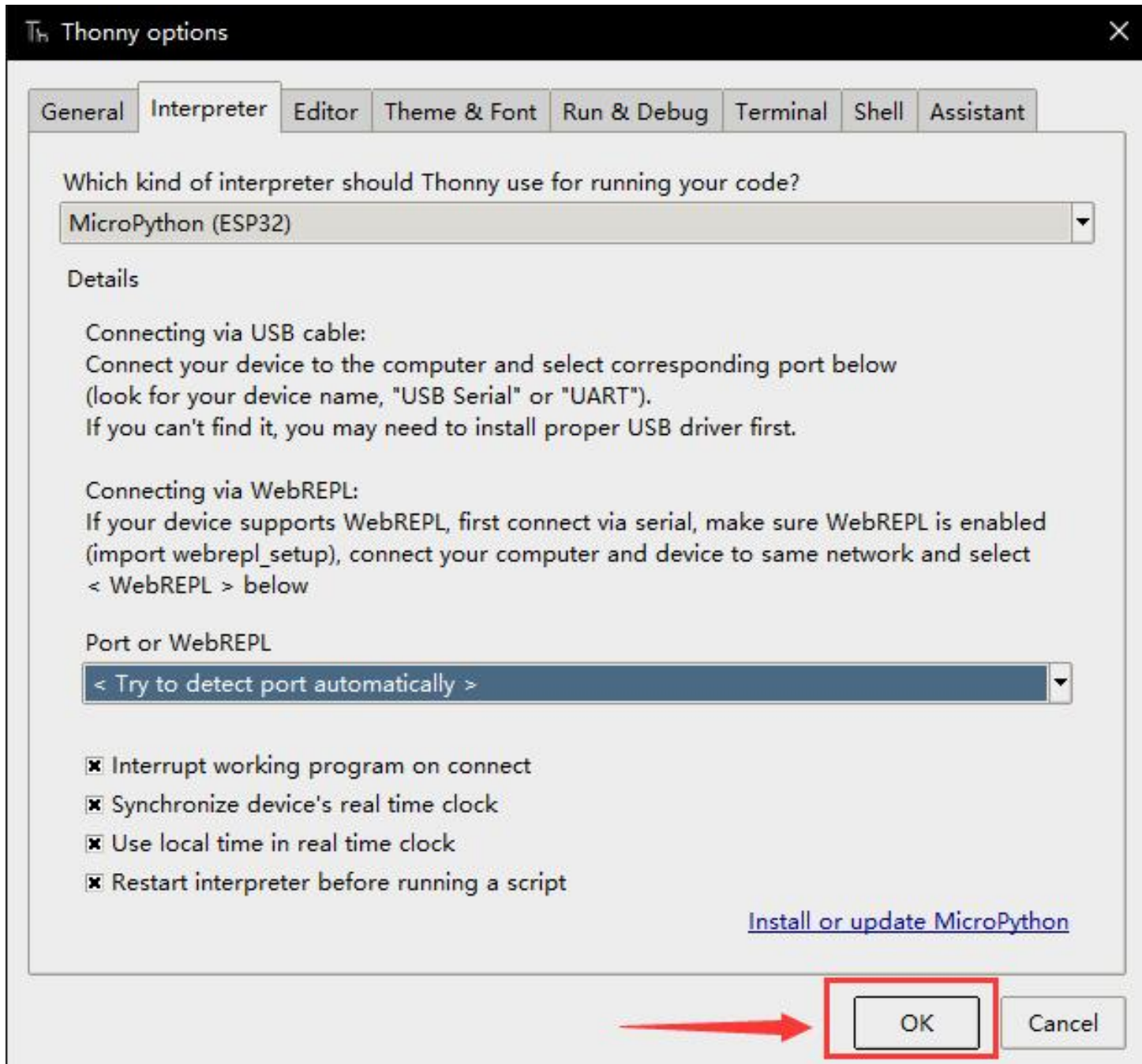
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-10.jpg)

2. 选择MicroPython (ESP32) 为解释器



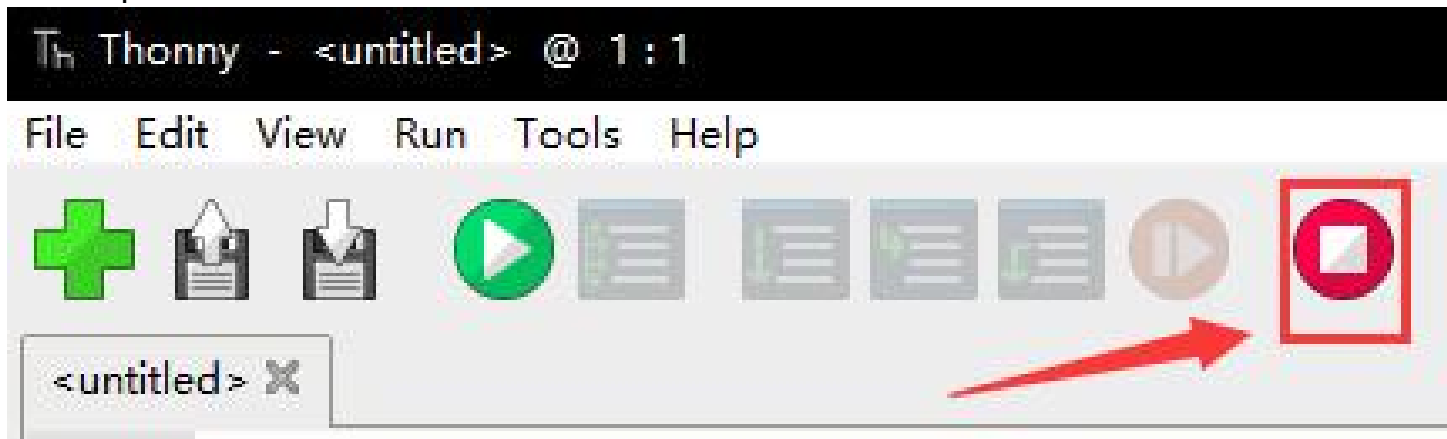
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-11.jpg)

3. 点击OK进行保存



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-12.jpg)

4. 点击Stop按键，或者快捷键 Ctrl+F2



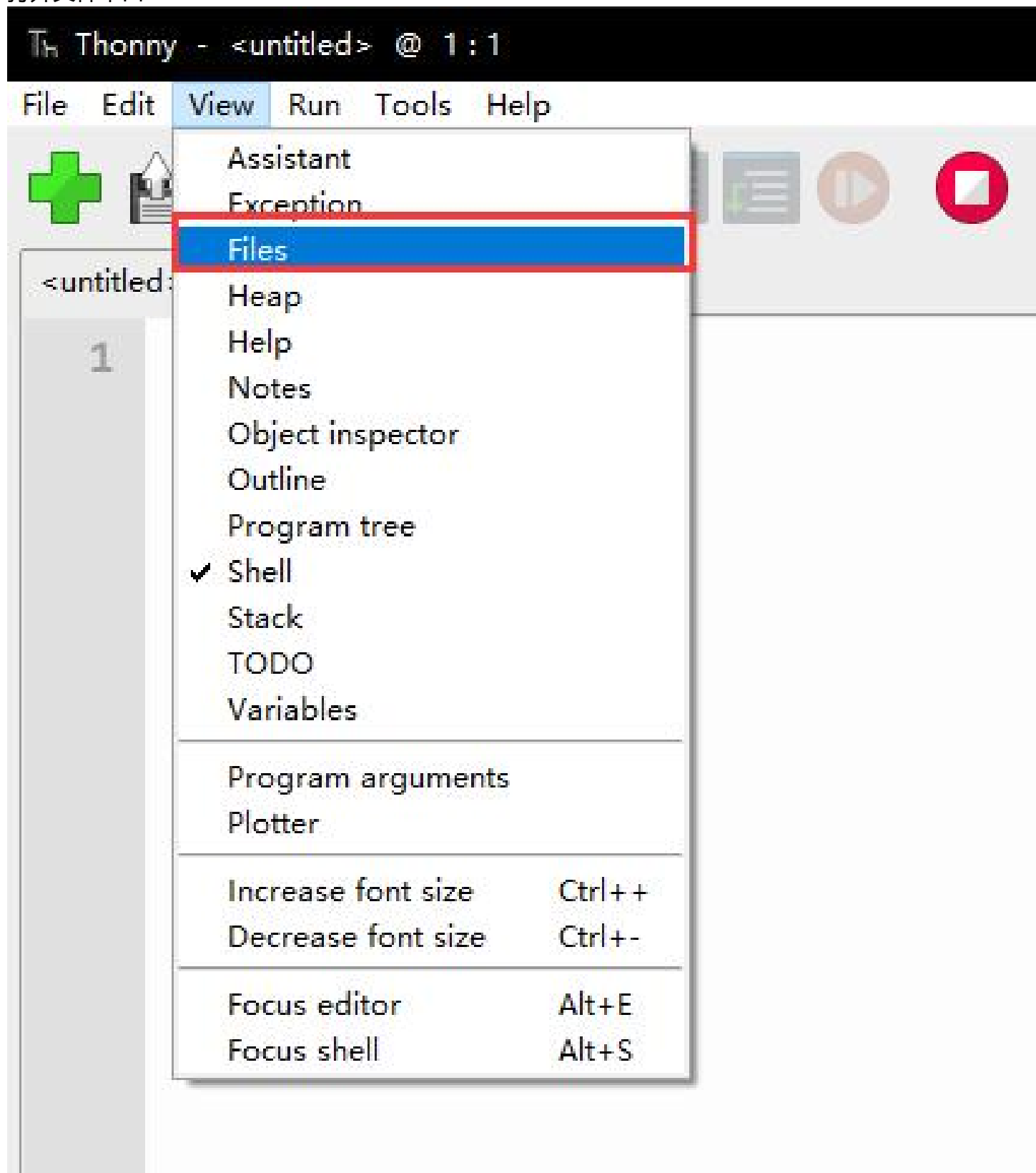
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-13.jpg)

5. 此时命令行可以看到 MicroPython 的版本信息与板名

```
Shell X
MicroPython v1.19.1-746-gf2de289ef-dirty on 2023-03-14; ESP32-S3-Pico with ESP32S3
Type "help()" for more information.
>>>
```

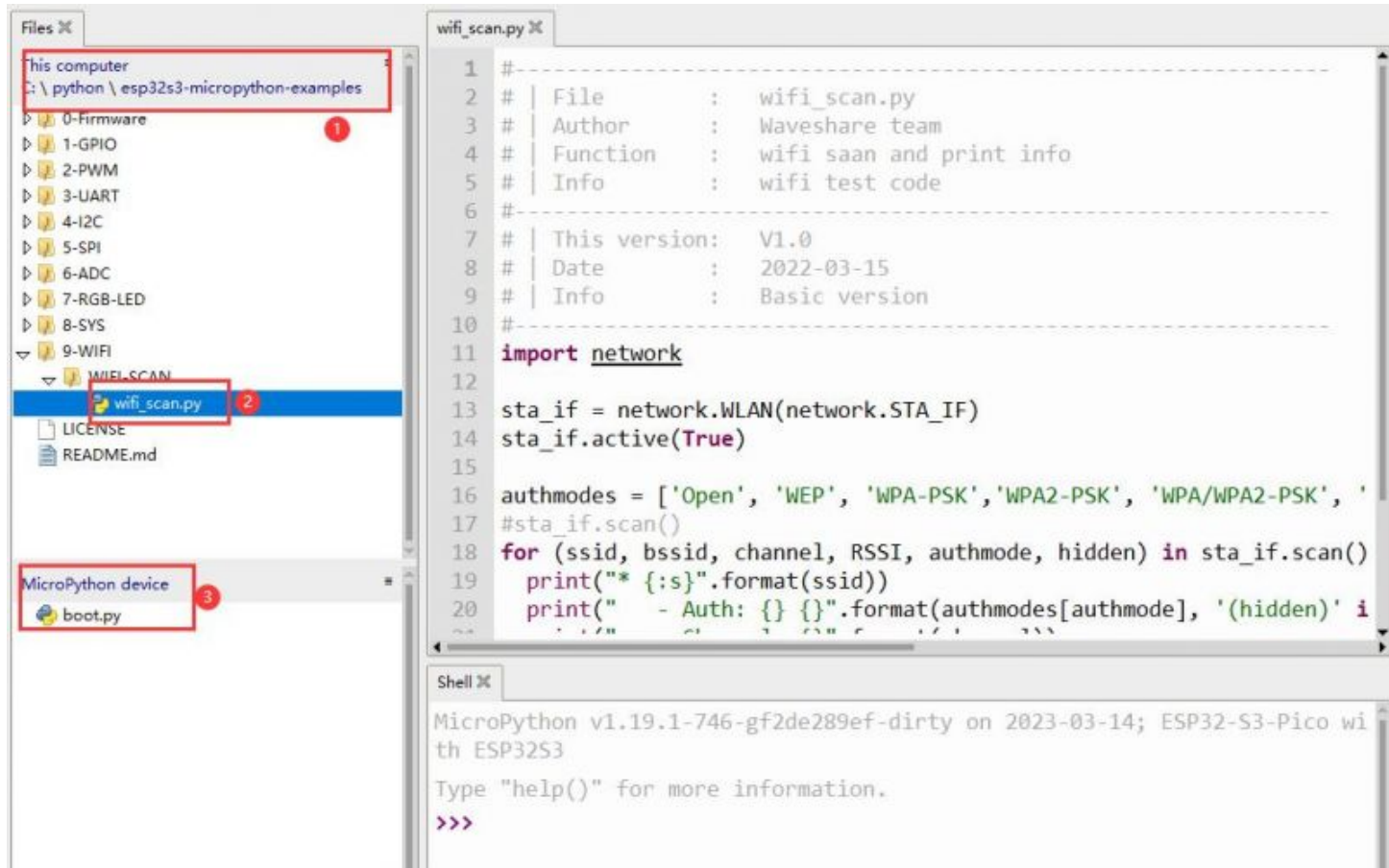
(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-17.jpg)

6. 打开文件串口



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-14.jpg)

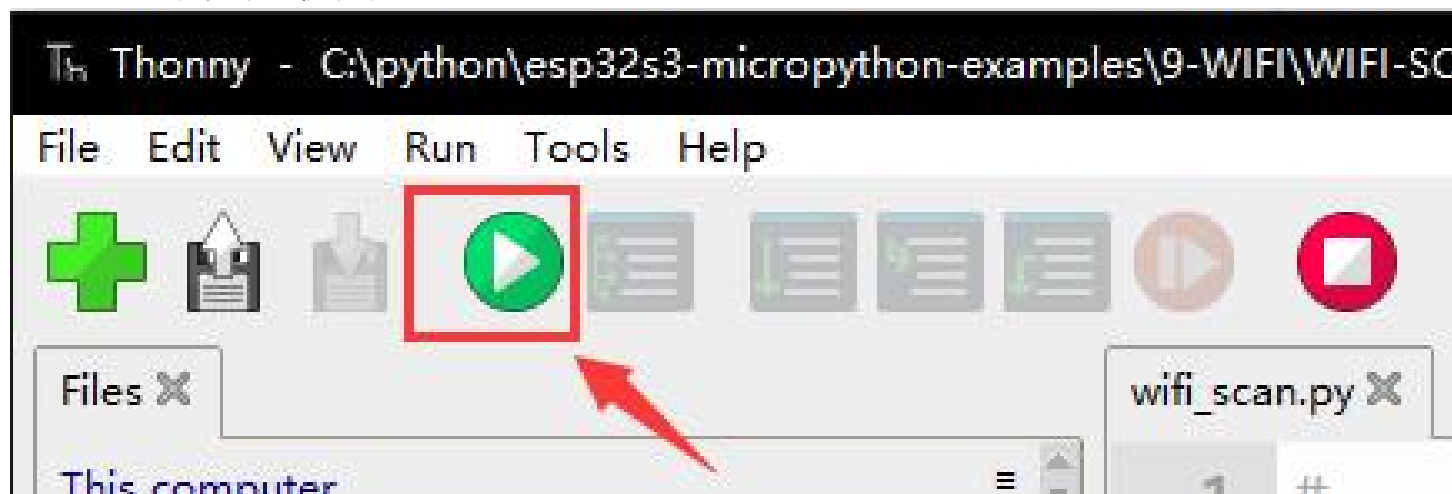
7. 选择我们的例程文件，并打开一个例程，这里我们选的是wifi-scan例程



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-15.jpg)

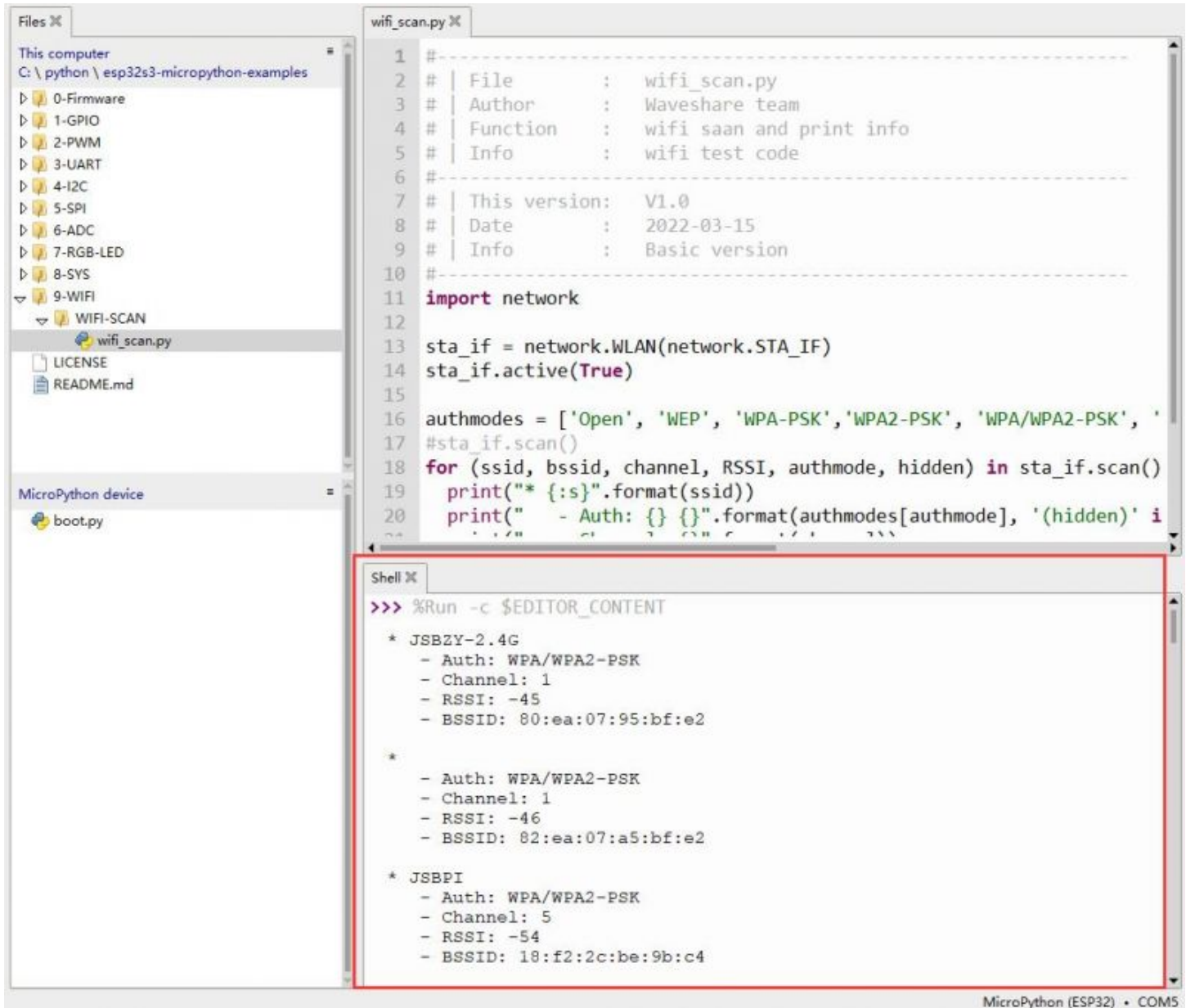
- ①当前本机文件目录
- ②我们要打开的例程 wifi-scan.py
- ③ESP32-S3-PICO的目录

8. 点击运行按键，或者快捷键 F5



(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-16.jpg)

9. 稍等片刻，可以看到esp32-s3-pico输出的wifi信息



```
Files X
This computer
C:\python\esp32s3-micropython-examples
  0-Firmware
  1-GPIO
  2-PWM
  3-UART
  4-I2C
  5-SPI
  6-ADC
  7-RGB-LED
  8-SYS
  9-WIFI
    WIFI-SCAN
      wifi_scan.py
      LICENSE
      README.md
MicroPython device
boot.py

wifi_scan.py X
1 #-----
2 # | File       : wifi_scan.py
3 # | Author    : Waveshare team
4 # | Function  : wifi scan and print info
5 # | Info     : wifi test code
6 #-----
7 # | This version: V1.0
8 # | Date       : 2022-03-15
9 # | Info     : Basic version
10 #-----
11 import network
12
13 sta_if = network.WLAN(network.STA_IF)
14 sta_if.active(True)
15
16 authmodes = ['Open', 'WEP', 'WPA-PSK', 'WPA2-PSK', 'WPA/WPA2-PSK', '
17 #sta_if.scan()
18 for (ssid, bssid, channel, RSSI, authmode, hidden) in sta_if.scan()
19 print("* {:s}".format(ssid))
20 print("  - Auth: {} {}".format(authmodes[authmode], '(hidden)' i
21 ..

Shell X
>>> %Run -c $EDITOR_CONTENT
* JSBZY-2.4G
  - Auth: WPA/WPA2-PSK
  - Channel: 1
  - RSSI: -45
  - BSSID: 80:ea:07:95:bf:e2
*
  - Auth: WPA/WPA2-PSK
  - Channel: 1
  - RSSI: -46
  - BSSID: 82:ea:07:a5:bf:e2
* JSBPI
  - Auth: WPA/WPA2-PSK
  - Channel: 5
  - RSSI: -54
  - BSSID: 18:f2:2c:be:9b:c4
```

(/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-18.jpg)

MicroPython例程讲解

- 例程大部分基于Pico-Sensor-Kit-B (<https://www.waveshare.net/shop/Pico-Sensor-Kit-B.htm>)进行编写

为了更好的使用体验我们的固件内部集成了一个py文件，用于将零散的GPIO包装成按照pico引脚顺序排列。源码如下：

```
from micropython import const
D0 = const(11)
D1 = const(12)
D2 = const(13)
D3 = const(14)
D4 = const(15)
D5 = const(16)
D6 = const(17)
D7 = const(18)
D8 = const(33)
D9 = const(34)
D10= const(35)
D11= const(36)
D12= const(37)
D13= const(38)
D14= const(39)
D15= const(40)
D16= const(42)
D17= const(41)
D18= const(1)
D19= const(2)
D20= const(4)
D21= const(5)
D22= const(6)
D26= const(7)
D27= const(8)
D28= const(9)

A1= const(7)
A2= const(8)
A3= const(9)

RGB_PIN= const(21)
USB_ADC= const(3)
```

1-GPIO

Blink效果讲解

- 将对应树莓派PICO的GPIO10(对应ESP32-S3的GP35)以1秒的频率闪烁

Key效果讲解

- 读取对应树莓派PICO的GPIO3(对应ESP32-S3的GP13)电平

- 若为低电平（按键按下）则将LED的电平翻转

2-PWM

fade效果讲解

- 将对应树莓派PICO的GPIO12(对应ESP32-S3的GP37)输出PWM频率在1000Hz, 占空比为0%-100%之间递增或者递减

Siren效果讲解

- 将对应树莓派PICO的GPIO12(对应ESP32-S3的GP37)输出PWM占空比为30%,频率在600-1400Hz之间递增或者递减

3-UART

uart效果讲解

- 以树莓派PICO的GPIO0(对应ESP32-S3的GP11)和GPIO1(对应ESP32-S3的GP12)做为UART的TX和RX引脚, 做UART回响

4-I2C

I2C-OLED效果讲解

- 以树莓派PICO的GPIO6(对应ESP32-S3的GP17)和GPIO7(对应ESP32-S3的GP18)做为I2C的SDA和SCL引脚,驱动1.5inch OLED模块

I2C-SCAN效果讲解

- 以树莓派PICO的GPIO8(对应ESP32-S3的GP33)和GPIO9(对应ESP32-S3的GP34)做为I2C的SDA和SCL

引脚，扫描I2C总线下的设备地址

5-SPI

Pico-LCD-1.3效果讲解

- 可驱动PICO-LCD-1.3模块

6-ADC

adc效果讲解

- 依次读取树莓派PICO的GPIO26-29引脚的电压值并输出信息

7-RGB

rgb效果讲解

- 点亮板载RGB-LED并不停转化颜色

8-SYS

FLASH-SIZE效果讲解

- 输出剩余FLASH空间

RAM-SIZE效果讲解

- 输出剩余和使用的RAM空间

RTC效果讲解

- 设置RTC时间并循环读取RTC数值

SLEEP效果讲解

- 将ESP32-S3设置为深度睡眠10秒，到达睡眠时间会自动复位并重启

WDT效果讲解

- 设置看门狗并等待看门狗复位

9-WIFI

scan效果讲解

- 扫描周围的wifi信号并显示相关信息

资料

原理图

- ESP32-S3-Pico原理图 (/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-SCH.pdf)

程序

- MicroPython例程 (/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-Example.zip)

软件

ESP32-S3

- 烧录软件与Micropython固件 (/wiki/%E6%96%87%E4%BB%B6:ESP32-S3-Pico-MPY-Flash.zip)
- Thonny Python IDE (Windows V3.3.3) (<https://www.waveshare.net/w/upload/7/73/Thonny-3.3.3.zip>)

Arduino

- 离线器件包 (提取码: 1r4i) (<https://www.aliyundrive.com/s/a1wzmiRkZNL>)

CH343

- Window VCP驱动 (<https://www.waveshare.net/w/upload/f/f1/CH343SER.7z>)
- 安卓APP (https://www.waveshare.net/w/upload/2/22/WCHUARTDemo_V1.3.7z)
- MAC驱动 (https://www.waveshare.net/w/upload/0/04/CH34XSER_MAC.7z)

串口

- 串口和网络调试助手 (<https://www.waveshare.net/w/upload/b/b3/Sscom5.13.1.zip>)

数据手册

ESP32-S3

- ESP32-S3数据手册 (中文) (/wiki/%E6%96%87%E4%BB%B6:Esp32-s3_datasheet_cn.pdf)
- ESP32-S3技术参考手册 (中文) (/wiki/%E6%96%87%E4%BB%B6:Esp32-s3_technical_reference_manual_cn.pdf)
- ESP32-S3数据手册 (英文) (/wiki/%E6%96%87%E4%BB%B6:Esp32-s3_datasheet_en.pdf)

- ESP32-S3技术参考手册 (英文) (/wiki/%E6%96%87%E4%BB%B6:Esp32-s3_technical_reference_manual_en.pdf)

CH343

- CH343数据手册(中文) (<https://www.waveshare.net/w/upload/b/b0/CH343DS1.PDF>)
- CH343数据手册 (英文) (<https://www.waveshare.com/w/upload/a/a3/CH343DS1-en.PDF>)

官方文档

ESP32官方文档

- ESP32-Arduino官方文档 (<https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html>)
- ESP-IDF官方文档(中文) (https://docs.espressif.com/projects/esp-idf/zh_CN/release-v5.0/esp32s3/get-started/index.html)
- ESP-IDF官方文档(英文) (<https://docs.espressif.com/projects/esp-idf/en/release-v5.0/esp32s3/index.html>)

MicroPython官方文档

- MicroPython官方文档 (<https://docs.micropython.org/en/latest/>)

