

TYPE-C USB CH340C ESP32 开发板

一、产品介绍

1) 产品描述

ESP32-WROOM-32 是一款通用型 Wi-Fi+BT+BLE MCU 模组，低功耗蓝牙和 Wi-Fi，用途广泛，集成 SPI 闪存，Wi-Fi 支持大范围的无线通信连接，支持通过路由器直接连接互联网；而蓝牙可让用户连接手机便于信号检测，应用场景：WiFi 方案，WiFi Mini 摄像头，Mesh 组网，智能家居，移动物联网等。

2) 产品参数

工作电压：TYPE-C 5V

SPI Flash：默认 32Mbit

串口速率：115200bps

频率范围：2412-2484MHz

蓝牙协议：蓝牙 4.2 BR/EDR 和 BLE 标准

WiFi 协议：802.11 b/g/n

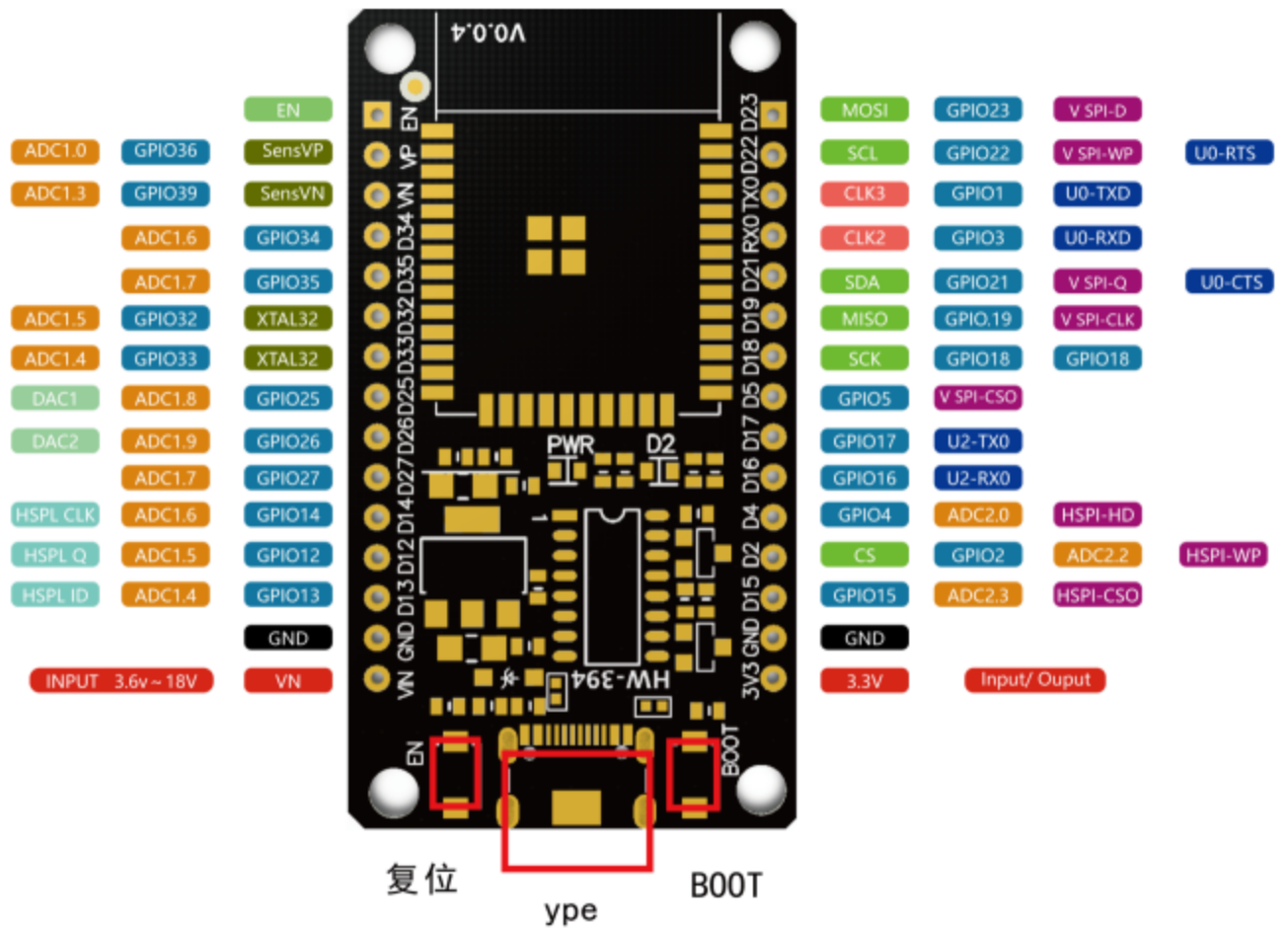
天线形式：板载 PCB 天线，增益 2dBi

支持接口：UART、SPI、SDIO、I2C、PWM、I2S、IR、ADC、DAC

友情提示：

1. 注意按图示接线，切勿接错。
2. 本文档仅代表编辑当时的产品认知和参数，后期如有更改恕不另行通知。

3) 产品实物引脚图



EN: 复位按键

BOOT: 下载按键

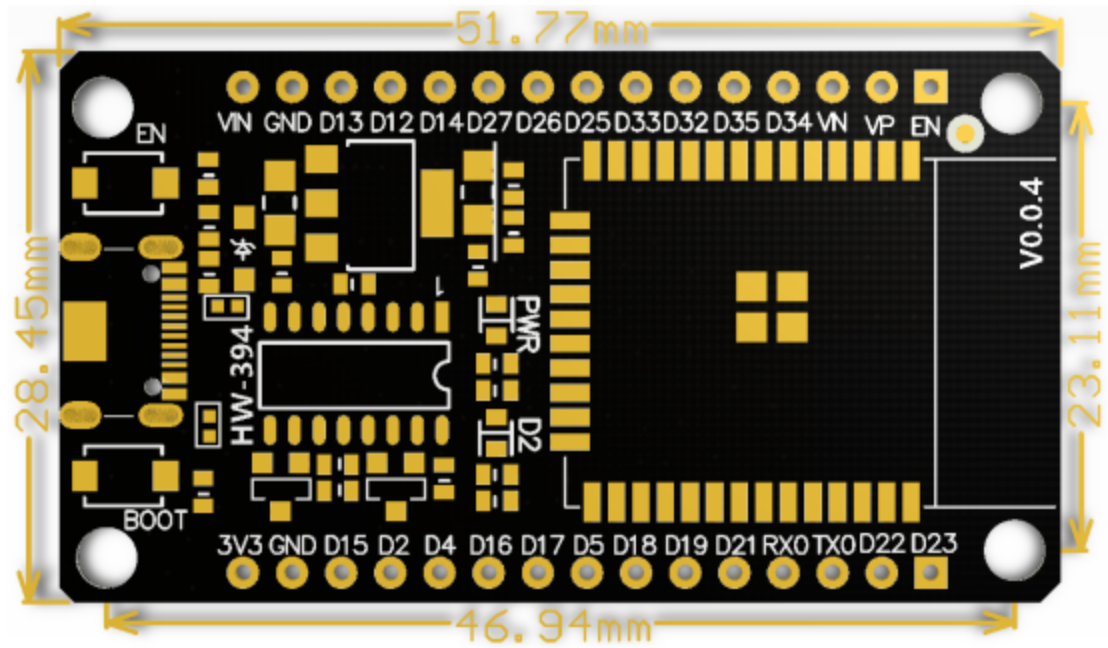
按下 Boot 键并保持(此时不要松开 Boot 键)进入“固件下载”模式，通过串口下载固件

USB: 可用作电路板的供电电源，或连接 PC 和 ESP32-WROOM-32 模组的通信接口

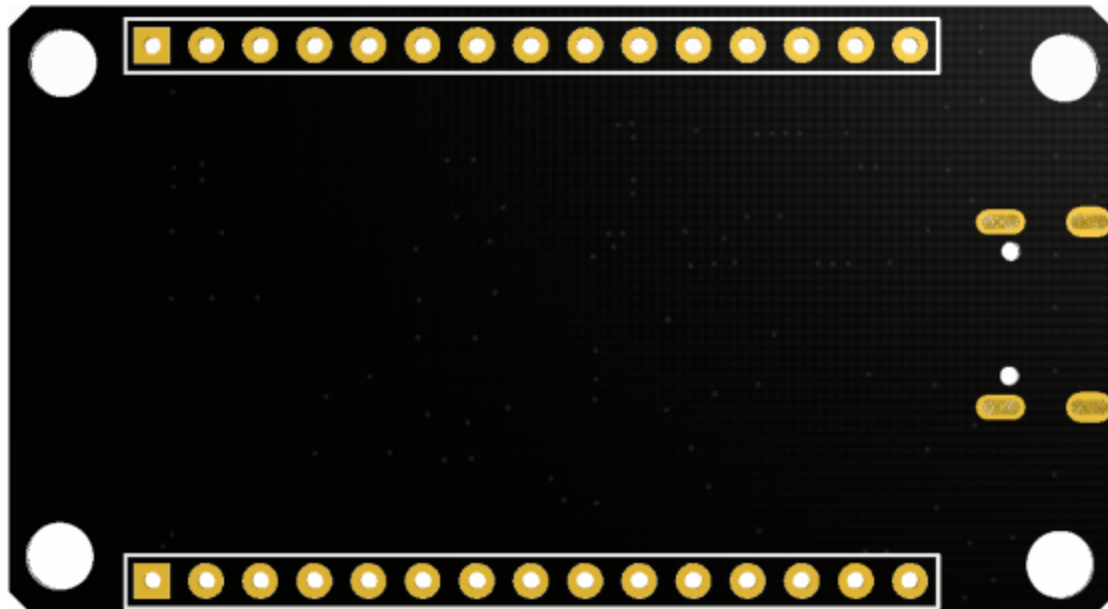
4) 产品实物尺寸图

定位孔尺寸：3.17mm

正面：

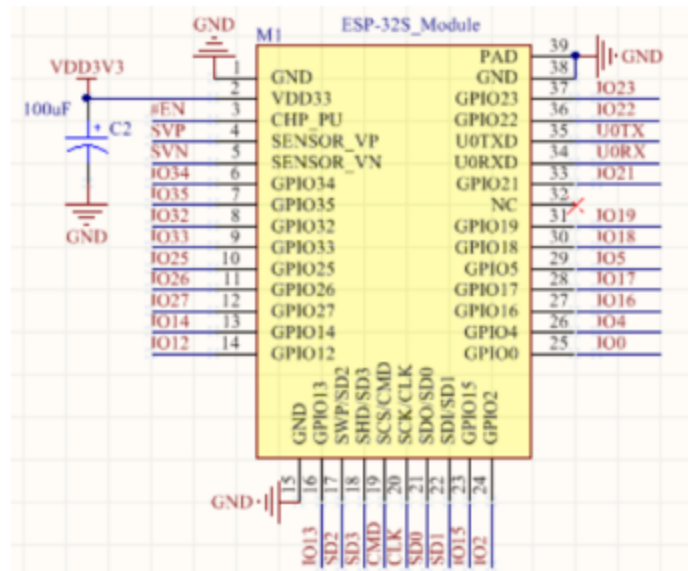


反面：

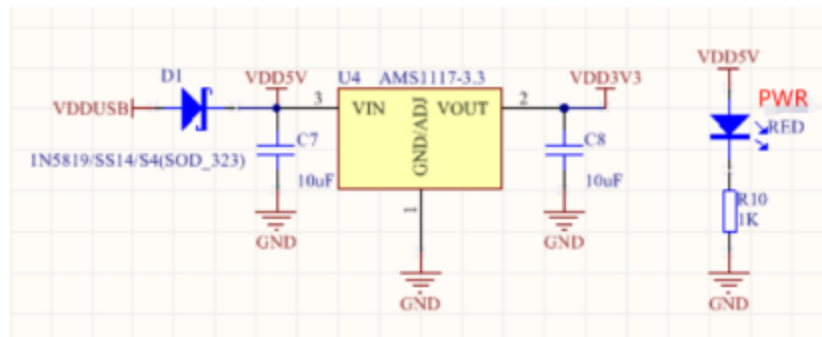


5) 开发板核心原理图

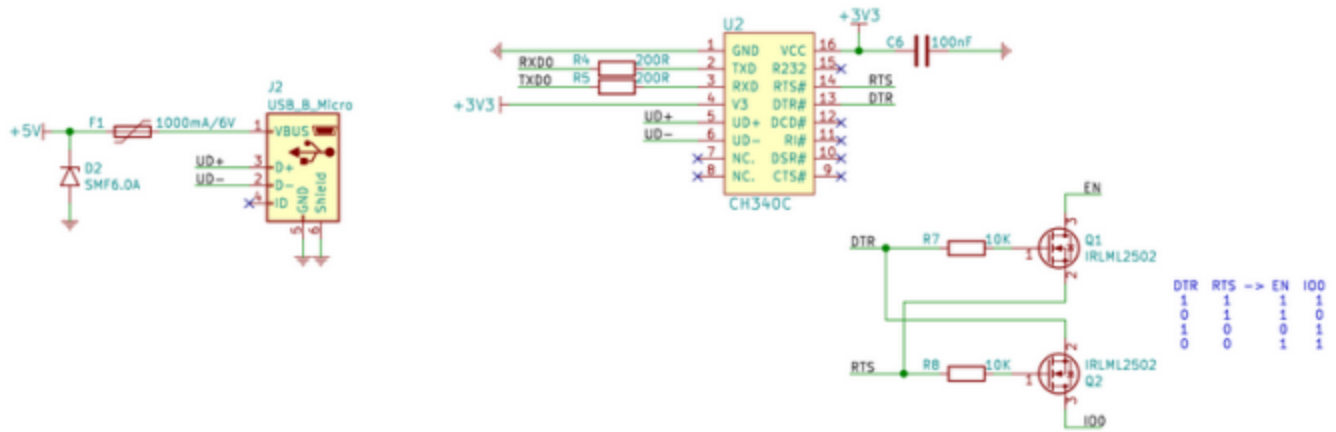
1--核心模组 ESP32-WROOM-32



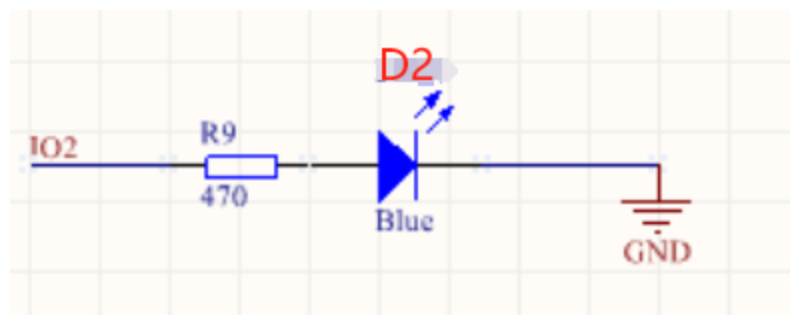
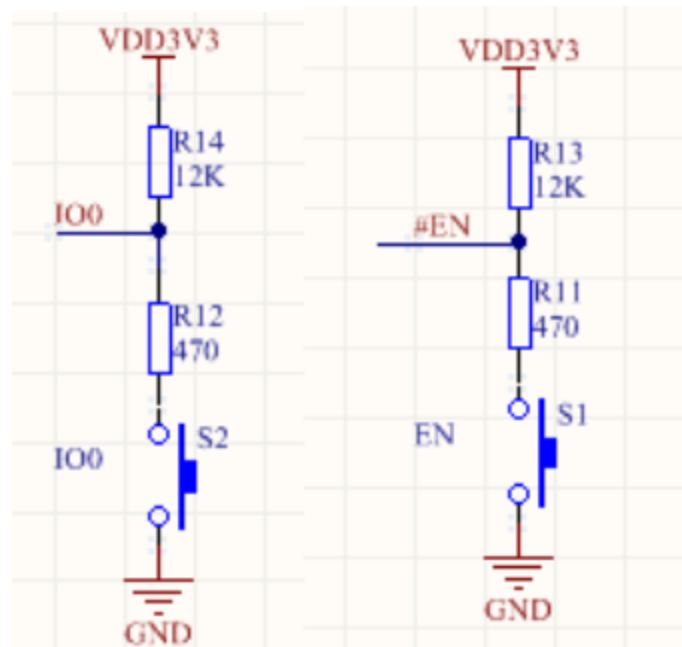
2--电源模块 AMS1117-3.3



3--下载模块 USB (TYPE-C) 与 CH340G 驱动



4--按键与 LED



蓝色 LEDD2 与模组的 IO2 连接，即 GPIO2，与外围的引脚丝印 D2 相对应。

6) 引脚特别说明

1--模组内部的 SPI Flash 引脚

GPIO6 至 GPIO11 引脚控制集成在模组内部的 SPI Flash, 不建议用于其他功能。

SCK/CLK	GPIO6
SDO/SD0	GPIO7
SDI/SD1	GPIO8
SHD/SD2	GPIO9
SWP/SD3	GPIO10
SCS/CMD	GPIO11

2--Strapping 管脚

ESP32 共有 5 个 Strapping 管脚。复位时, ESP32 会采样 Strapping 管脚, 并锁存。不建议用于其他功能, 除非管脚不够用。

MTDI
GPIO0
GPIO2
MTDO
GPIO5

3--启动时, 引脚状态为高电平的管脚

芯片启动时引脚会变为高电平, 使用不注意可能会出现一些莫名其妙的问题 (比如说高电平有效的继电器会吸合等)。

GPIO1
GPIO3
GPIO6~GPIO11 (SPI)
GPIO5
GPIO14
GPIO15

4--只能用于输入的管脚（无内部上下拉）

GPIO34
GPIO35
GPIO36
GPIO39

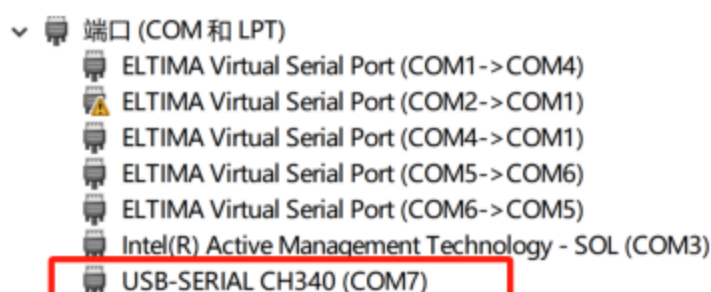
二、使用步骤（以烧录和测试我司提供的 Hellow_world 固件为例）

1) 连接开发板

1--将开发板通过 Type-C 数据线连接到计算机（上电后电源指示灯亮）

2--查看开发板的连接状态

打开我的电脑-管理-设备管理器查看 COM 口识别正常，如下图所示



说明：如果没装驱动显示如下感叹号。本开发板的串口驱动芯片为 CH340G，我们提供的对应的驱动路径和文件名如下图所示，只需将提供的 CH340 驱动软件包打开进行安装即可。



2) 烧录固件程序

1--打开烧录工具“flash_download_tool”

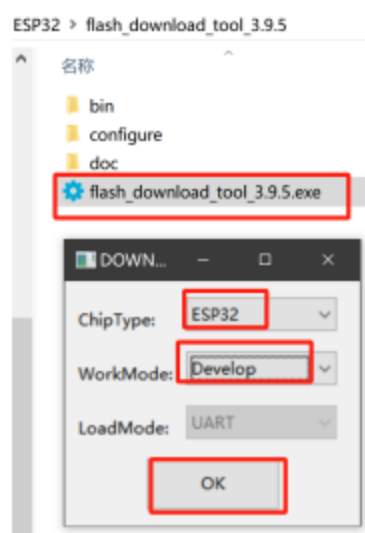
说明：该烧录工具在当前文档中使用的版本为 3.9.5

工具已经提供在软件包中，或者通过如下链接从官方直接获取：

<https://www.espressif.com/zh-hans/support/download/other-tools>

2--选择配置参数

选择“ESP32” 默认“Develop” 如下界面, 点击 OK



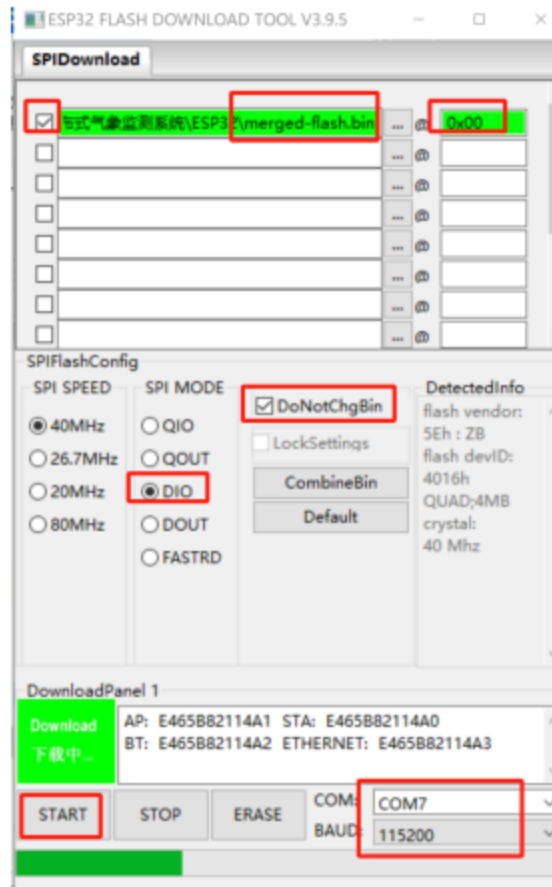
3--加载烧录的 bin 文件

说明：bin 文件实际位置和名称如下图所示。

merged-flash.bin

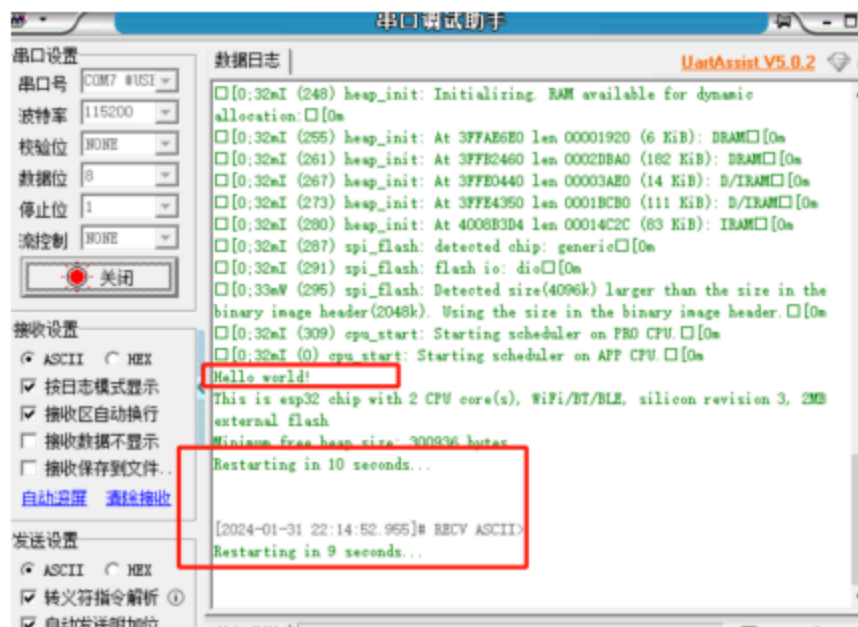
4--烧录下载程序

参数配置好后，开发板对应的端口，单击开始烧录前需**长按 Boot 键 5S** 并点击下载即可。



3) 测试固件程序

烧录成功后，USB 重新拔插，打开串口助手，即可看到如下现象：



三、开发步骤

(一) Arduino IDE

1) 打开 IDE 与配置环境

1--下载并打开 Arduino IDE 工具

说明：Arduino IDE 的获取可通过下面的官网链接获取

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

该软件在本文档中使用的最新的版本为 2.2.1

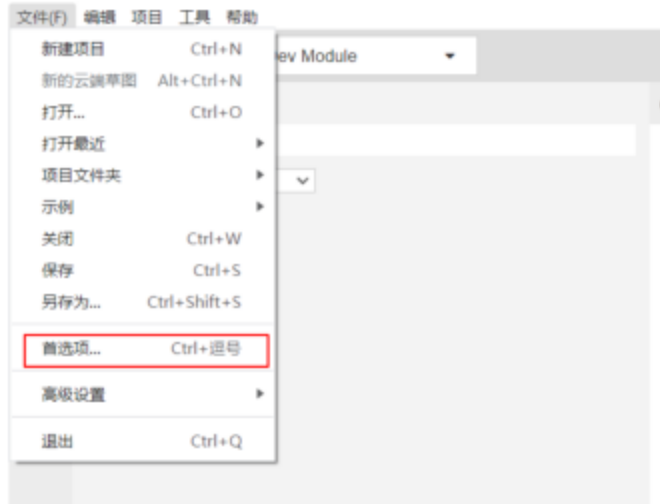
2--配置 ESP 开发板管理器

配置 ESP 开发板管理器地址的目的是能够在 Arduino IDE 中可以搜索到 esp32 的配置包。

打开 Arduino IDE ，找到文件 → 首选项，然后以下链接复制粘贴到首选项附加网址，如红框里，点击“好”

https://dl.espressif.com/dl/package_esp32_index.json

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



3—添加 ESP32 开发环境

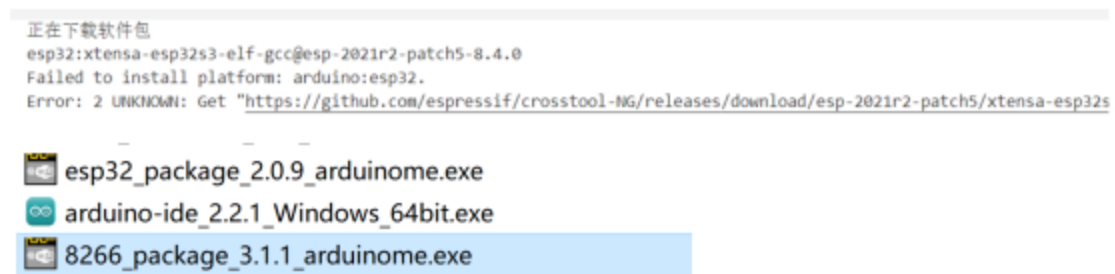
在开发板管理器搜索安装 ESP32 开发板，并点击安装，本文档对应的 ESP32 点击下图所示的第二个。



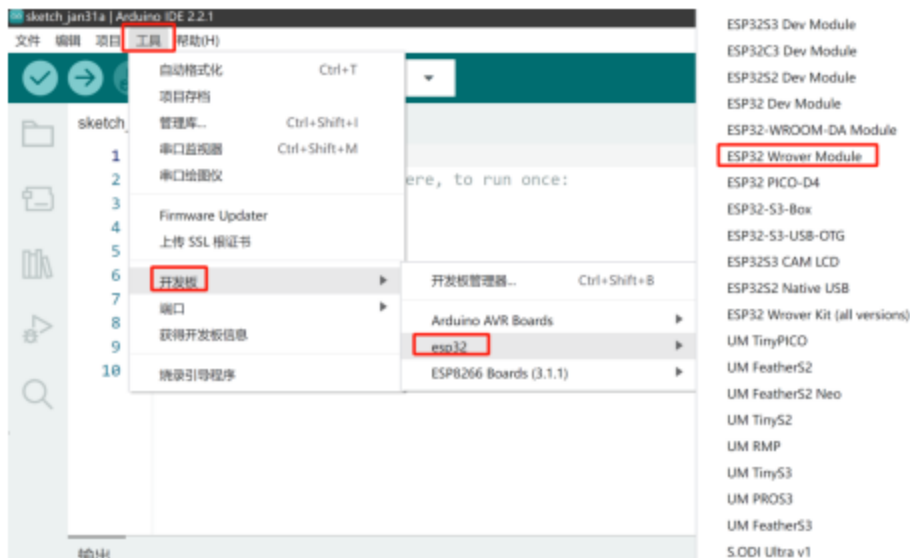
如下图所示，Arudino 正在下载相应的资源



如下图所示，由于网络环境等问题，很可能下载失败，这里我们使用提供的 ESP 开发板安装包



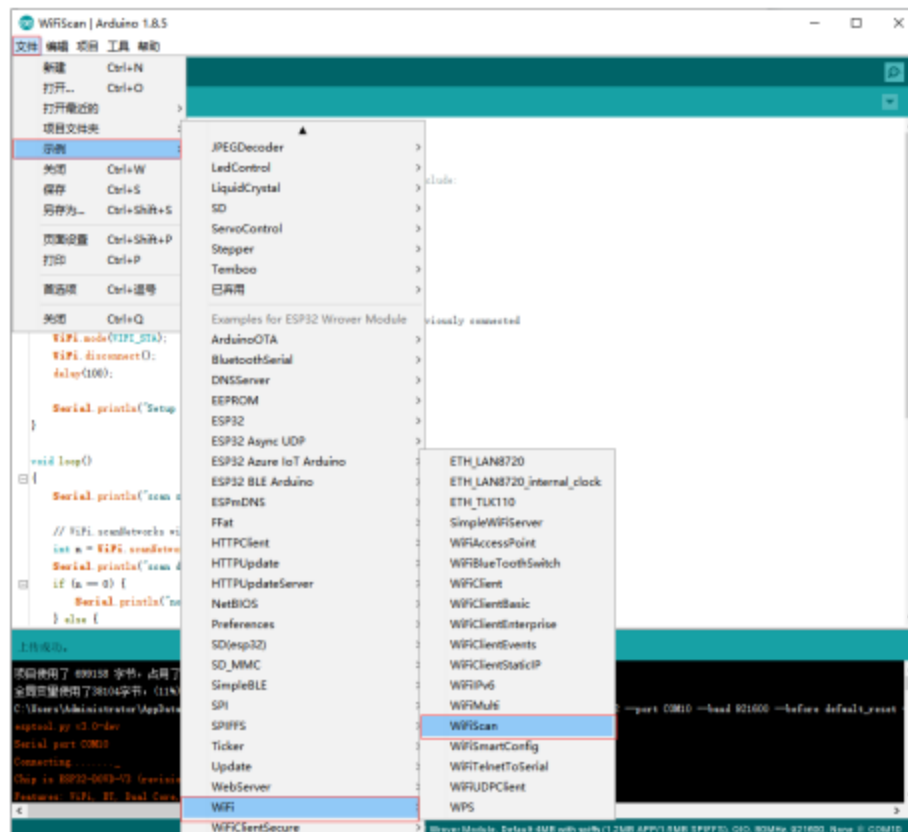
4--重启 Arduino IDE 软件后，配置开发板相应参数，如下图红框





2) 获取代码并烧录程序

1--选择“WiFiSan”例程并上传到开发板上



2--下载时出现.....__..... 这时需要长按 Boot 键 3S 以上

3--下载完成后，打开串口监视串口，波特率 115200

(二) VSCode 开发

说明: VSCode 的基本安装在这里就不做细致说明, 网上有很多, 在下面操作之前, 请确保 VSCode+ESP-IDF 插件已经完成安装

0) ESP-IDF 的导入

为了便于开发, 这里增加上述提到的在 VSCode 中快速导入 ESP-IDF 的方式。

1--自行安装 VSCode 与 ESP-IDF 插件

VSCode 下载链接: <https://code.visualstudio.com/>

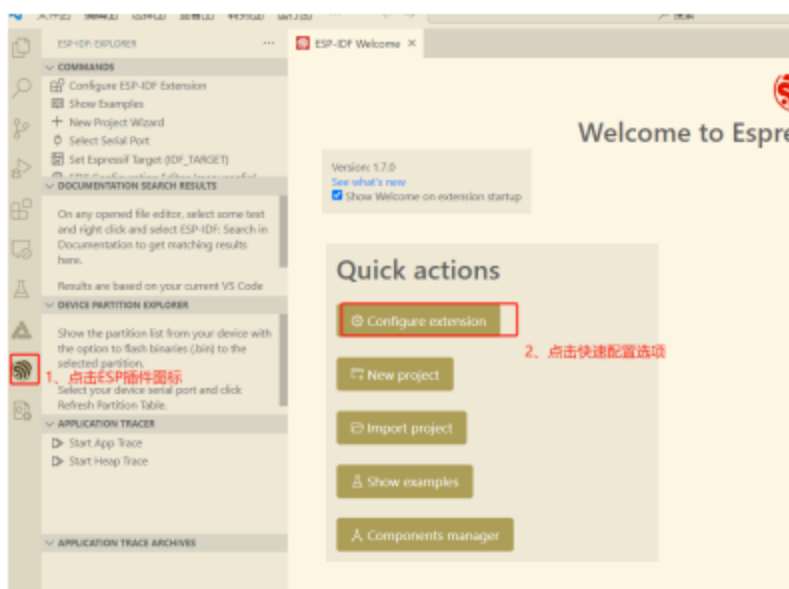
下载完成后, 打开 VSCode, 点击插件, 然后搜索 ESP-IDF 关键字, 进行安装, 如下图所示:



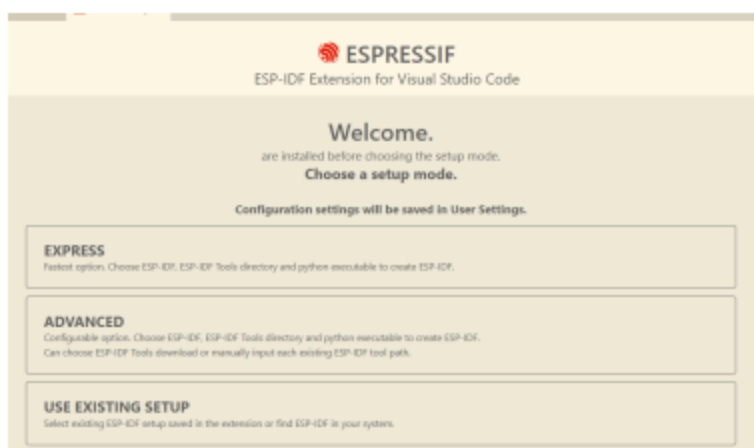
2--安装 ESP-IDF 库

ESP-IDF 库的安装有多种方式, 这里简单介绍两种方式, 一种是在线安装的方式, 另一种方式是导入提供的 ESP-IDF 固件包。

无论采用哪种方式都需要先进入 ESP 插件的界面：

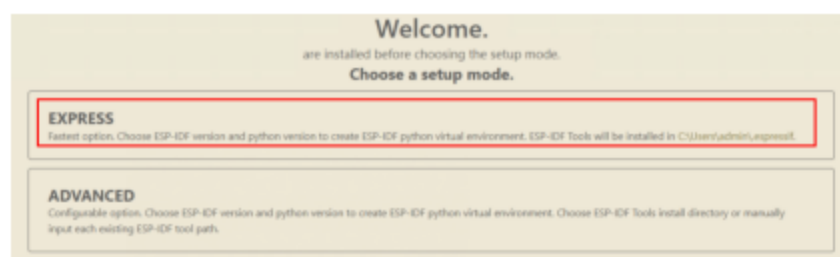


进入到主界面如下图所示：



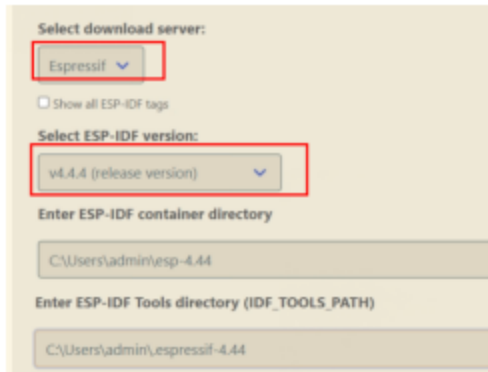
在线安装方式：

对于第一次安装环境的，如下图所示，选择第一个选项：

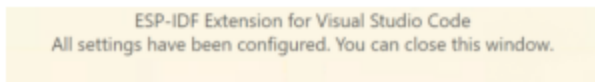


- 选择下载源为 Espressif，如果选择的下载服务器是 GitHub，可能会出现下载失败的情况

➤ 选择版本为 v4.4.4 版本，否则会出现版本不兼容问题

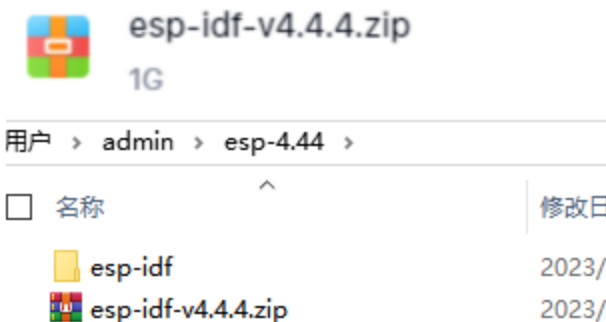


成功后如下图所示

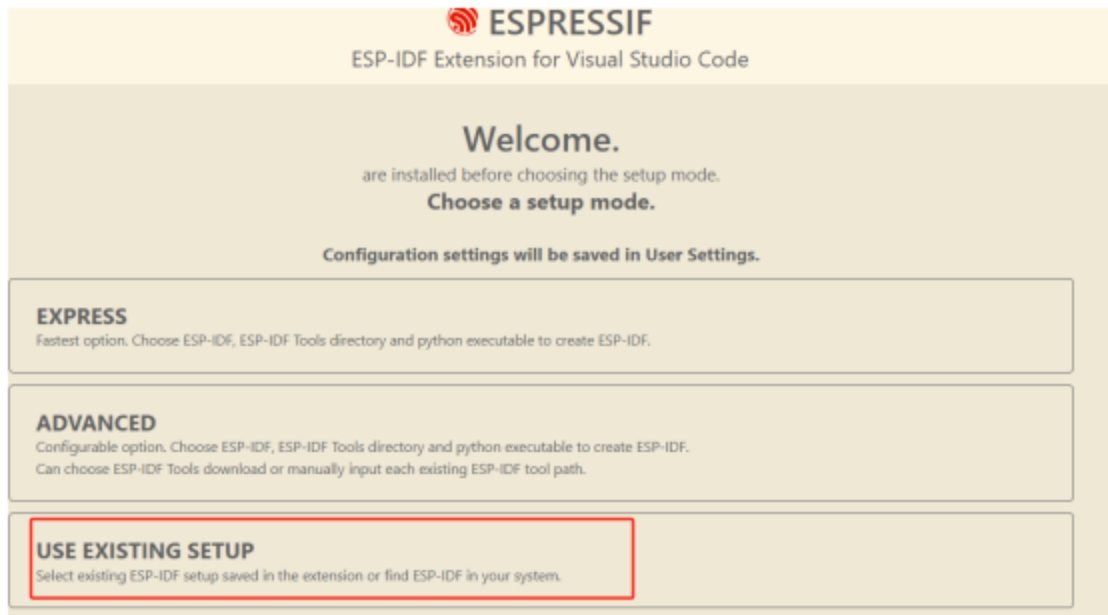


导入本地固件包方式：

固件：将固件准备好至本地，建议放到 c 盘相应的位置，然后解压



对于本地安装固件，如下图所示，选择第三个选项，第三个表示会自动从本地识别然后进行安装：



1) 打开 IDE

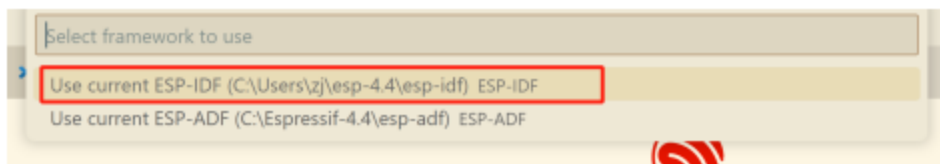
打开 VSCode 编辑器，并点击 ESP-IDF 插件，如下图所示。成功打开后，可以看到 ESP-IDF 在 VSCode 中的可视化界面。



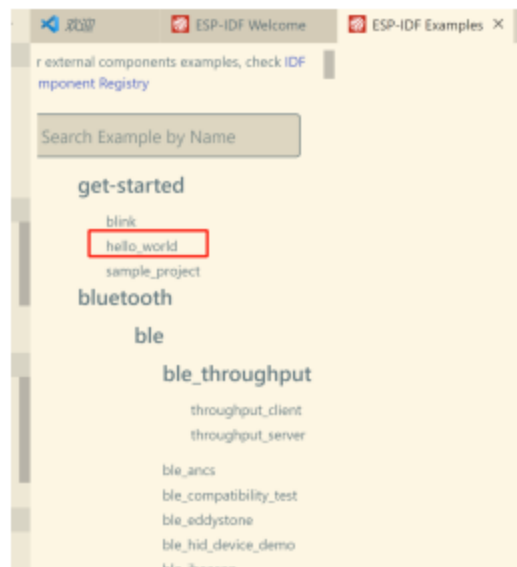


2) 导入样例工程

点击上图所示中的任意一处“Show examples”，会提示选择你已经安装的官方的 ESP-IDF 库，如下图所示

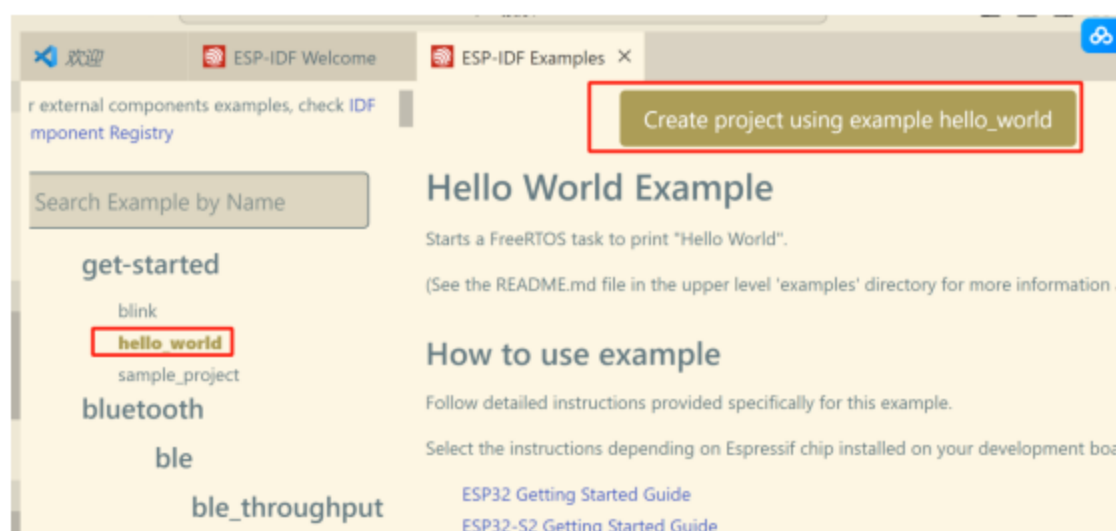


选择后即可看到官方为 IDF 库提供的样例工程，如下图所示，这里我们可以先选择“hello_world”，作为测试，这个样例工程是通过板子的串口可以输出“Hello World”字符串，并且倒数重启。

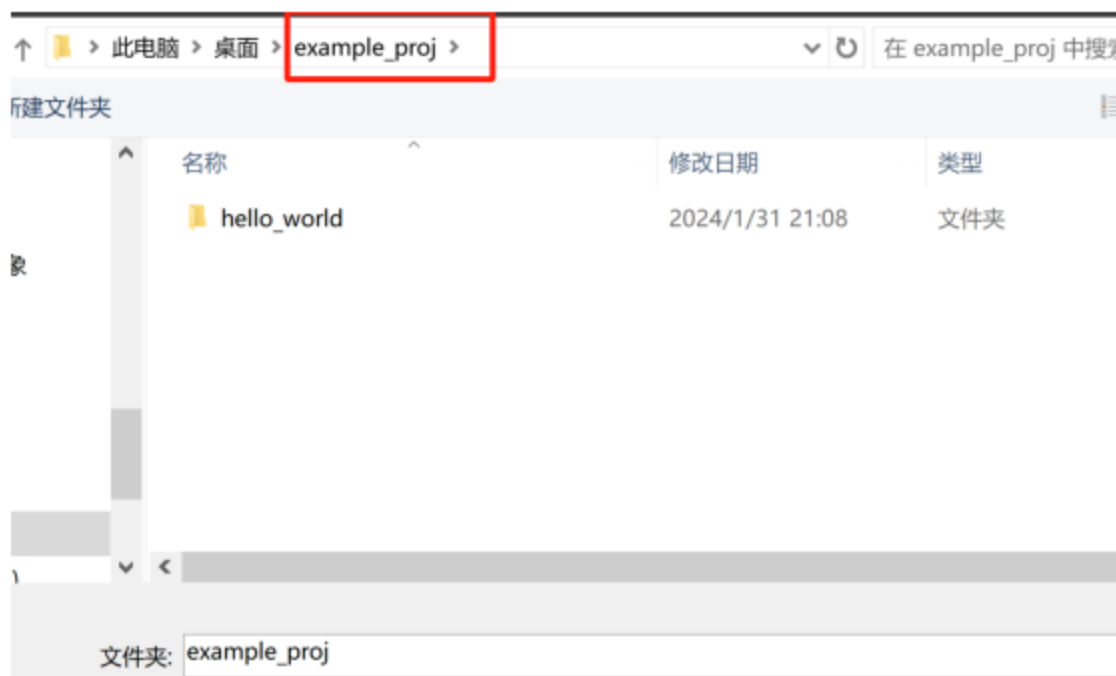


选择该样例工程后，在界面的左侧会出现相应的工程的介绍，这里我们直

接点击最上方的“使用样例工程创建工程”的按钮，然后会提示你将样例工程保存在什么位置，**这里要注意，请在一个没有中文字符的路径下新建一个英文名的文件夹**，我们可以将样例工程保存在相应的位置即可，如果路径或者工程名有中文，可能会导致后面的工程编译无法正常进行。



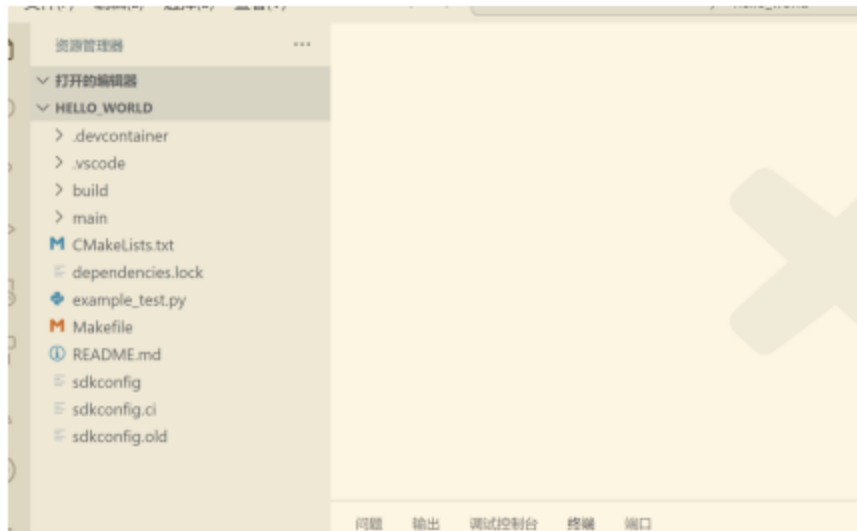
接例如下图所示，我这里将样例工程保存到我在桌面新建的一个“example_proj”文件夹下。



3) 配置、编译、烧录工程

完成导入样例工程后，VSCode 会自动打开一个包含样例工程的新窗口，

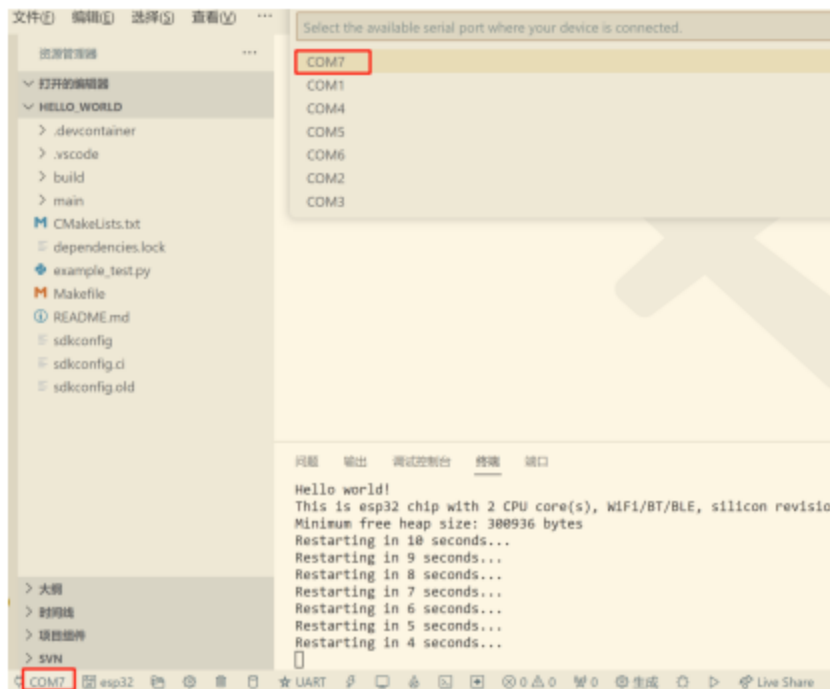
如下图所示：



1--选择输出的串口 COM7

因为我的设备在设备管理器上的 COM 口为 COM7，所以这里我也选择 COM7，

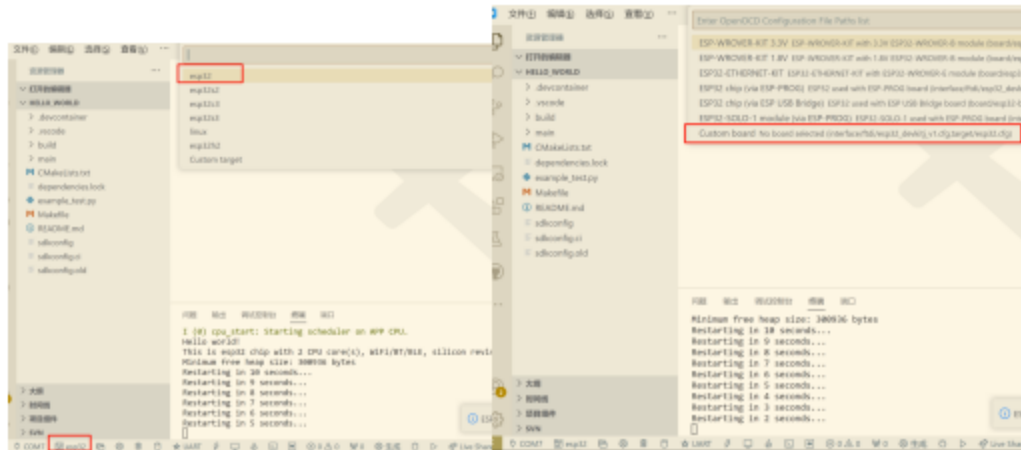
如下图所示：





2--选择设备型号 esp32


设备型号我们点击后，首先选择 ESP32-->然后选择 Custom Board 即可，

我们要选择正确的型号，否则会导致编译出来的可执行程序正常的进行烧录或运行。如下图所示，我们需要等待工程帮忙我们完成配置后才能进入下一步骤。



3--选择编译符号 

4--选择烧录符号 

烧录完成后，即可打开 VSCode 自带的监视器 （即串口工具），或者使用第三方的串口工具也可以，即可看到现象如下：

```
问题 输出 调试控制台 终端 端口

I (0) cpu_start: Starting scheduler on APP CPU.
Hello world!
This is esp32 chip with 2 CPU core(s), WiFi/BT/BLE, silicon revision 3, 4MB external
Minimum free heap size: 300936 bytes
Restarting in 10 seconds...
Restarting in 9 seconds...
Restarting in 8 seconds...
Restarting in 7 seconds...
Restarting in 6 seconds...
Restarting in 5 seconds...
█
```

C:\Espressif-4.4\python_env\idf4.4_py3.8_env\Scripts\python.exe

C:\Users\zj\esp-4.4\esp-idf\components\esptool_py\esptool\esptool.py --chip ESP32 merge_bin -o merged-flash.bin @flash_args