

# ESP32 系列教程之四： Windows 搭建 esp-idf 环境

Date: 2021.7.20

Version: V1.1

# 关于文档

本文档为 ESP32 教程系列，旨在为客户进行 ESP32 系列芯片开发提供环境搭建、工程示例演示等方面的参考文档及视频演示，降低 ESP32 系列芯片、模组开发的入门难度。

ESP32 教程系列文档主要参考于乐鑫官网提供的 ESP32 入门教程：[https://docs.espressif.com/projects/esp-idf/zh\\_CN/latest/esp32/get-started/index.html](https://docs.espressif.com/projects/esp-idf/zh_CN/latest/esp32/get-started/index.html)。由于个人经验有限，文档编写过程中难免存在失误、错漏之处，欢迎广大开发爱好者对本文档提出批评建议。

版本信息：

日期	版本	作者	说明
2021.6.24	V1.0	Amiliya	首次发布
2021.7.20	V1.1	Amiliya	修改部分描述

文档变更通知：以启明云端官网版本为准，恕不另行通知。

意见提交邮箱：[amiliya@wireless-tag.com](mailto:amiliya@wireless-tag.com)

# 目 录

1 工具链设置.....	1
1.1 安装 git.....	1
1.1.1. 下载.....	1
1.1.2 安装.....	1
1.2 安装 python（版本 3.8 以上，建议 3.8）.....	2
1.2.1 下载.....	2
1.2.2 安装.....	2
1.3 获取 ESP-IDF.....	4
1.4 安装 ESP-IDF 工具安装器（版本 2.5）.....	5
1.4.1 下载.....	5
1.4.2 安装.....	5
2 运行 hello_world 示例.....	8
2.1 克隆工程.....	8
2.2 工程配置.....	8
2.3 编译工程.....	9
2.4 硬件连接.....	9
2.5 查看端口.....	9
2.6 工程烧录.....	10
2.7 监视工程.....	10
3 VS Code 的使用.....	11
3.1 VS Code 的下载与安装.....	11
3.1.1 下载 VS Code.....	11
3.1.2 安装 VS Code.....	11
3.2 VS Code 的使用.....	12
3.2.1 移植脚本.....	12
3.2.2 运行 hello_world 示例.....	13
4 参考视频.....	15
4.1 安装 VS Code.....	15
4.2 安装 git 与 python.....	15
4.3 安装 ESP-IDF 工具.....	15

4.4 VS Code 运行 hello_world 示例.....	15
4.5 Windows 运行 hello_world 示例.....	15
5 后 记.....	16
5.1 注意事项.....	16
5.2 相关建议.....	16

## 1 工具链设置

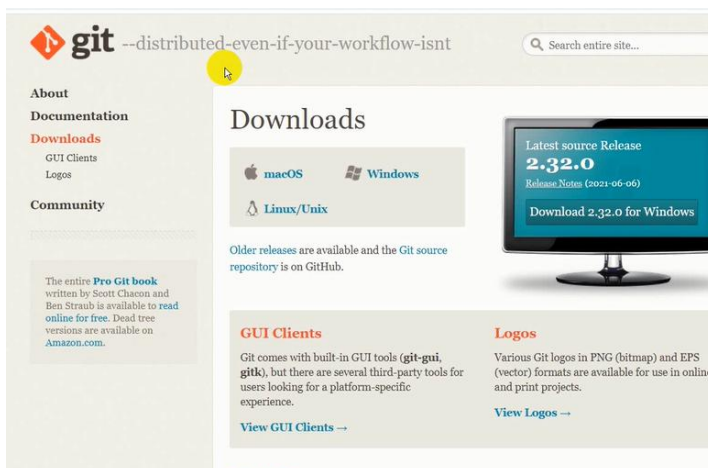
### 1.1 安装 git

#### 1.1.1. 下载

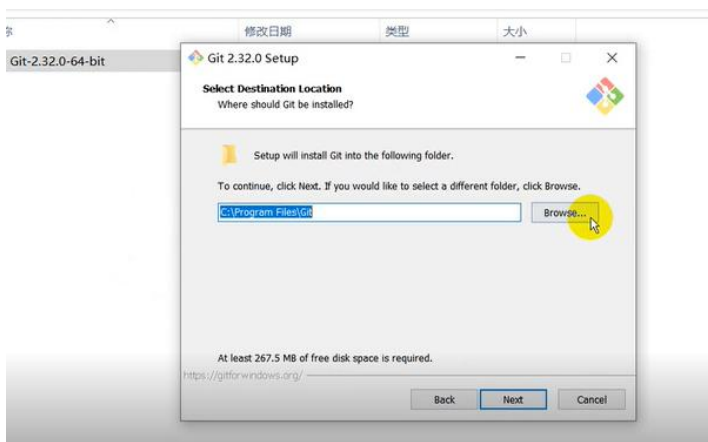
git 下载地址: <https://git-scm.com/downloads>

#### 1.1.2 安装

##### (1) 下载



##### (2) 设置安装路径



##### (3) 一直 NEXT，直至完成安装



## 1.2 安装 python（版本 3.8 以上，建议 3.8）

### 1.2.1 下载

python 下载地址：<https://www.python.org/downloads/>

### 1.2.2 安装

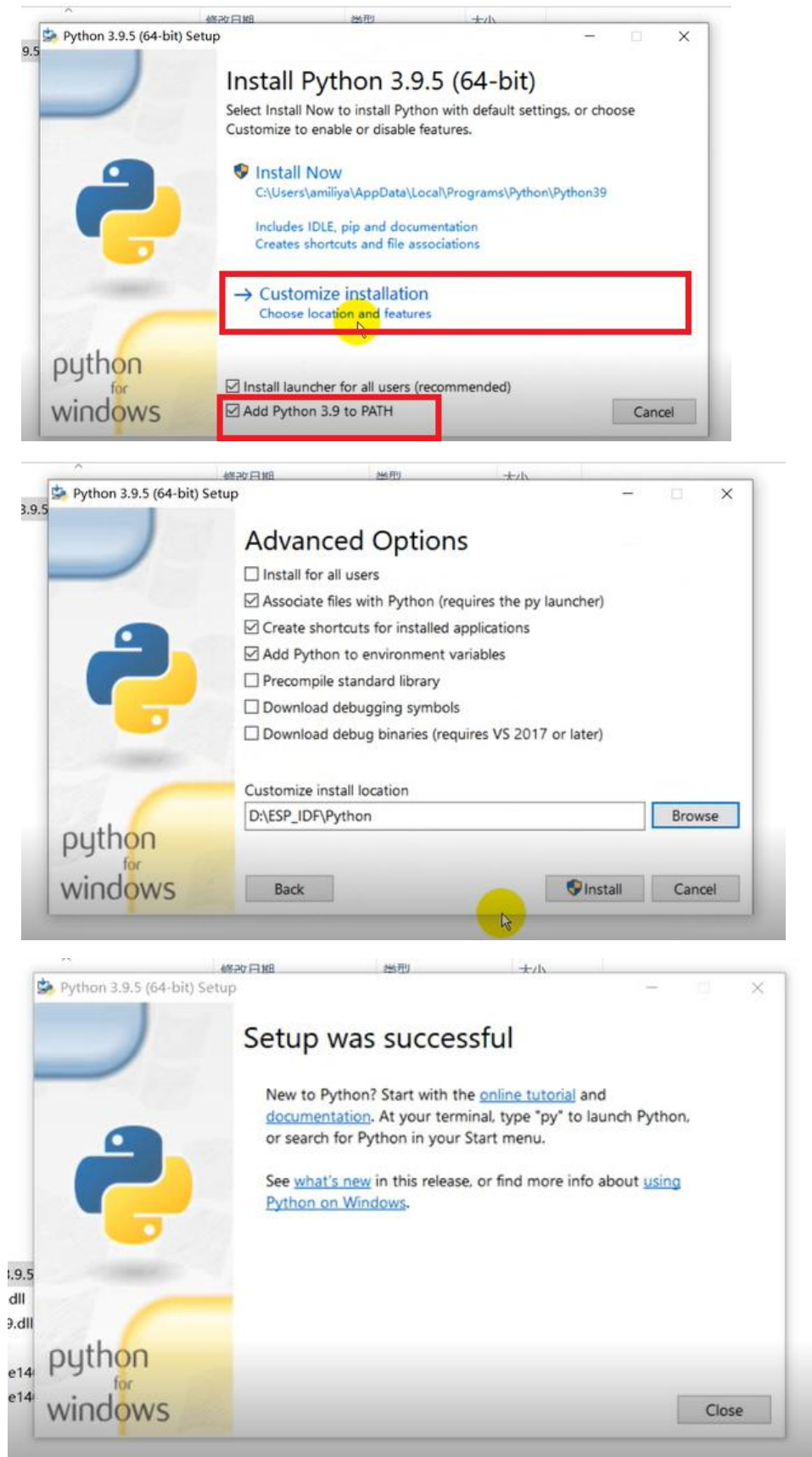
#### （1）下载

3.6	安全	2016-12-23	2021-12-23	PEP 494	Google Translate
2.7	生命尽头	2010-07-03	2020-01-01	PEP 373	

发布版本	发布日期	点击查看更多
蟒蛇 3.9.5	2021 年 5 月 3 日	下载 发行说明
蟒蛇 3.8.10	2021 年 5 月 3 日	下载 发行说明
蟒蛇 3.9.4	2021 年 4 月 4 日	下载 发行说明
蟒蛇 3.8.9	2021 年 4 月 2 日	下载 发行说明
蟒蛇 3.9.2	2021 年 2 月 19 日	下载 发行说明
蟒蛇 3.8.8	2021 年 2 月 19 日	下载 发行说明
蟒蛇 3.6.13	2021 年 2 月 15 日	下载 发行说明

#### （2）安装



### 1.3 获取 ESP-IDF

方式一：在文件夹下打开 git，使用下方命令下载 esp-idf（下载的 esp-idf 位置在当前文件夹）：

```
git clone -b release/v4.3 --recursive https://github.com.cnpmjs.org/espressif/esp-idf.git
```

（注：-b release/v4.3 表示当前下载版本为 release/4.3, 需要下载其它版本或版本介绍请移至乐鑫官网查看具体介绍

MINGW64:/d/ESP\_IDF

```
amiliya@DESKTOP-LDC9RD8 MINGW64 /d/ESP_IDF
$ git clone -b release/v4.3 --recursive https://github.com.cnpmjs.org/espressif/esp-idf.git
Cloning into 'esp-idf'...
remote: Enumerating objects: 247239, done.
remote: Counting objects: 100% (8617/8617), done.
remote: Compressing objects: 100% (3610/3610), done.
remote: Total 247239 (delta 4932), reused 8042 (delta 4727), pack-reused 238622
Receiving objects: 100% (247239/247239), 139.31 MiB | 9.02 MiB/s, done.
Resolving deltas: 100% (181747/181747), done.
Updating files: 98% (8117/8282)
```

出现如下错误：

```
Submodule path 'components/unity/unity': checked out '7d2bf62b7e6afaf38153041a9d53c21aeeca9a25'
Submodule path 'examples/build_system/cmake/import_lib/main/lib/tinyxml2': checked out '7e8e249990ec
Submodule path 'examples/peripherals/secure_element/atecc608_ecdsa/components/esp-cryptoauthlib': ch
Failed to recurse into submodule path 'components/nghttp/nghttp2'

amiliya@DESKTOP-LDC9RD8 MINGW64 /d/ESP_IDF
$
```

进入 esp-idf 目录下执行：`git submodule update --init --recursive`

```
amiliya@DESKTOP-LDC9RD8 MINGW64 /d/ESP_IDF/esp-idf (release/v4.3)
$ git submodule update --init --recursive
Cloning into 'D:/ESP_IDF/esp-idf/components/nghttp/nghttp2/third-party/mruby'..
fatal: unable to access 'https://github.com/mruby/mruby/': OpenSSL SSL_read: Co
fatal: clone of 'https://github.com/mruby/mruby' into submodule path 'D:/ESP_ID
Failed to clone 'third-party/mruby'. Retry scheduled
Cloning into 'D:/ESP_IDF/esp-idf/components/nghttp/nghttp2/third-party/neverble
Cloning into 'D:/ESP_IDF/esp-idf/components/nghttp/nghttp2/third-party/mruby'..
fatal: unable to access 'https://github.com/mruby/mruby/': Failed to connect to
```

方式二：前往乐鑫官网下载 esp-idf 压缩包，解压后改名为 esp-idf, 下载地址：

[https://www.espressif.com/zh-hans/support/download/sdks-demos?keys=&field\\_type\\_tid%5B%5D=13](https://www.espressif.com/zh-hans/support/download/sdks-demos?keys=&field_type_tid%5B%5D=13)





## 1.4 安装 ESP-IDF 工具安装器 (版本 2.5)

### 1.4.1 下载

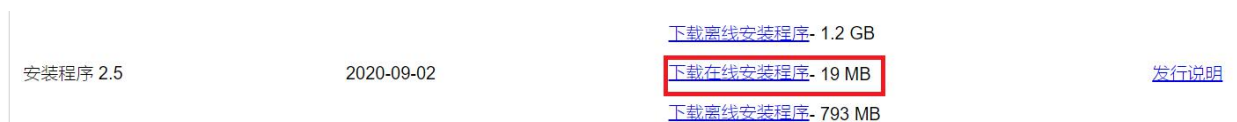
ESP-IDF 工具安装器下载地址: <https://dl.espressif.com/dl/esp-idf/?idf=4.4>

### 1.4.2 安装

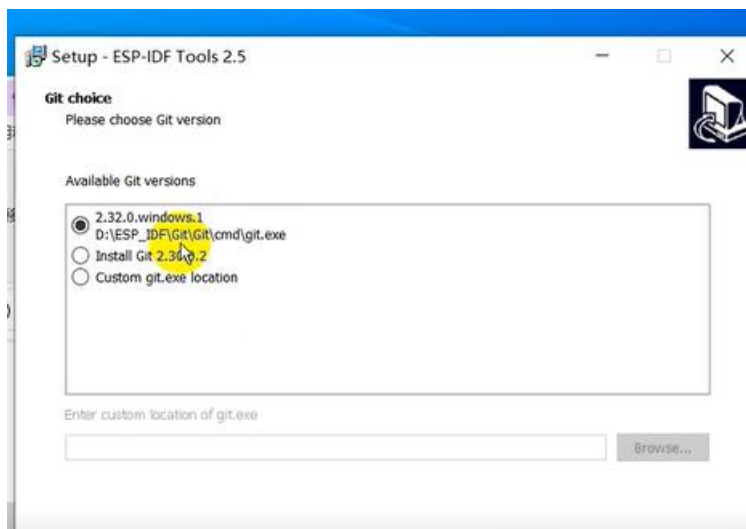
#### (1) 下载

寻找特定版本?

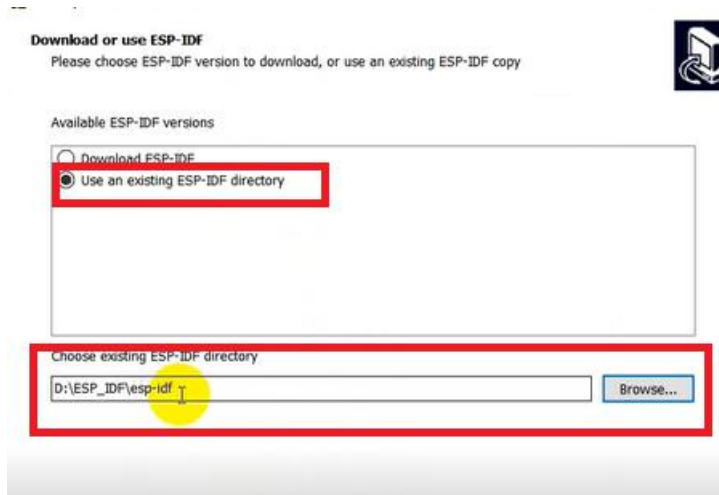
[查看所有可用的下载。](#)



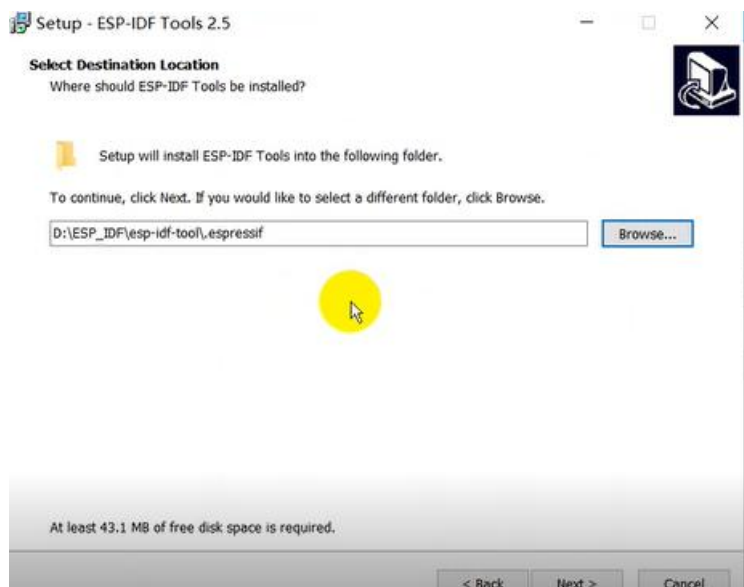
#### (2) 自动检测的 git 路径



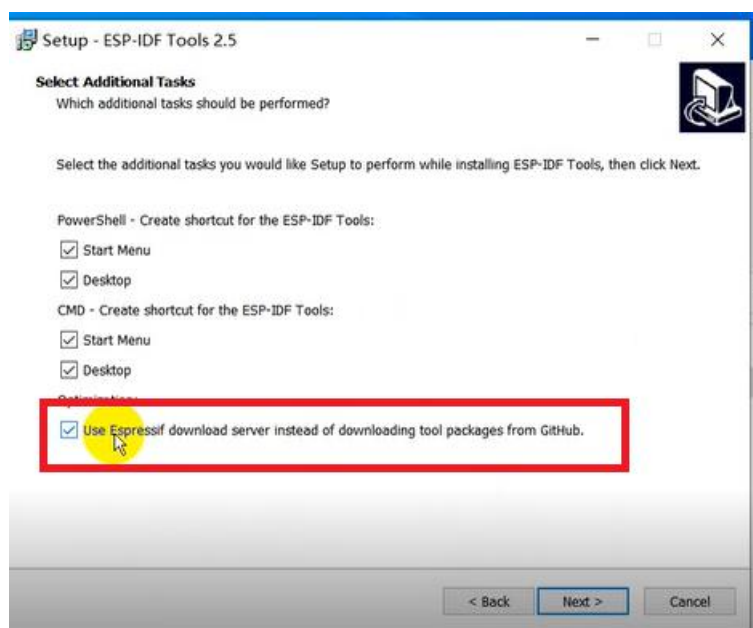
#### (3) 选择自己下载的 esp-idf



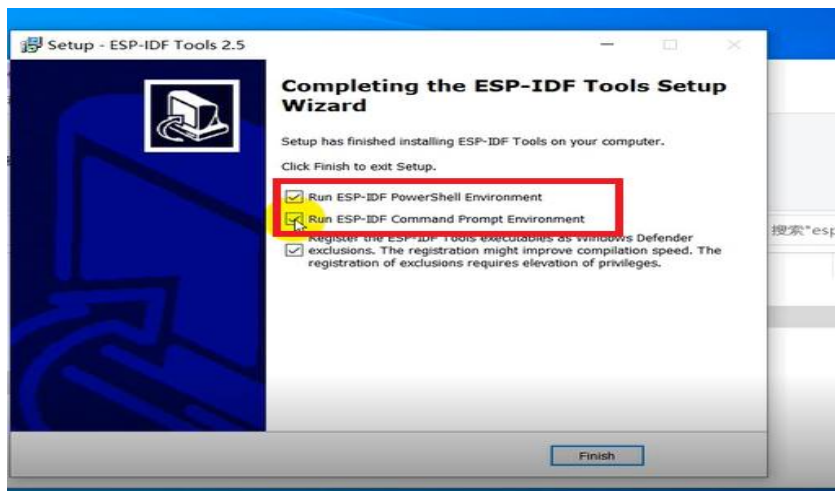
#### (4) 选择安装路径



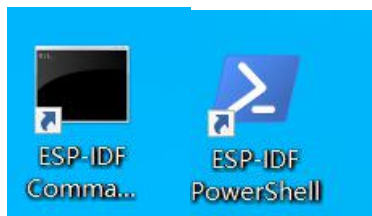
#### (5) 使用 Espressif 作为下载服务器



(6) 选择创建快捷方式



(7) 完成安装



## 2 运行 hello\_world 示例

### 2.1 克隆工程

使用命令克隆 hello\_world 工程：

```
xcopy /e /i %IDF_PATH%\examples\get-started\hello_world hello_world
```

```
D:\ESP_IDF\esp-idf>xcopy /e /i %IDF_PATH%\examples\get-started\hello_world hello_world
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\CMakeLists.txt
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\example_test.py
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\Makefile
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\README.md
覆盖 D:\ESP_IDF\esp-idf\hello_world\main\CMakeLists.txt (Y:是/N:否/A:全部)?a
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\main\CMakeLists.txt
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\main\component.mk
D:\ESP_IDF\esp-idf\examples\get-started\hello_world\main\hello_world_main.c
复制了 7 个文件
```

### 2.2 工程配置

进入 hello\_world 示例，进行工程配置：

```
cd hello_world
```

```
idf.py set-target esp32
```

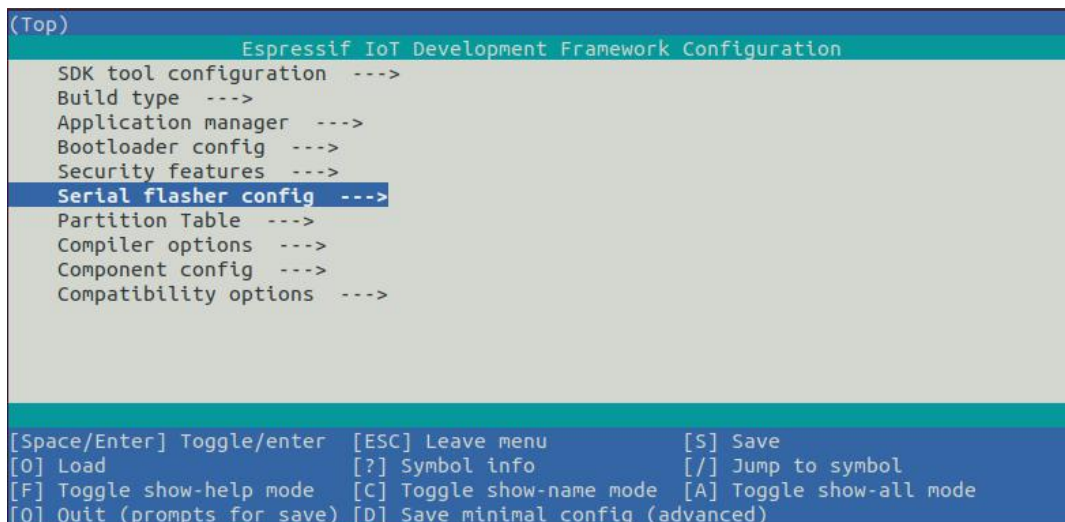
 (使用其它芯片请选择对应芯片类型，如：esp32c3)

(打开一个新项目后，应首先设置“目标”芯片 `idf.py set-target esp32`。注意，此操作将清除并初始化项目之前的编译和配置（如有）。您也可以直接将“目标”配置为环境变量（此时可跳过该步骤）)

```
D:\ESP_IDF\esp-idf>cd hello_world
D:\ESP_IDF\esp-idf\hello_world>idf.py set-target esp32
Adding "set-target"'s dependency "fullclean" to list of commands with default set of options.
Executing action: fullclean
Executing action: set-target
Set Target to: esp32, new sdkconfig created. Existing sdkconfig renamed to sdkconfig.old.
Running cmake in directory d:\esp_idf\esp-idf\hello_world\build
Executing "cmake -G Ninja -DPYTHON_DEPS_CHECKED=1 -DESP_PLATFORM=1 -DIDF_TARGET=esp32 -DCCACHE_ENABLE=1 d:\esp_idf\esp-idf\hello_world"...
-- Found Git: D:/ESP_IDF/Git/Git/cmd/git.exe (found version "2.32.0.windows.1")
```

```
idf.py menuconfig
```

如果之前的步骤都正确，则会显示下面的菜单：



### 工程配置 — 主窗口

您可以通过此菜单设置项目的具体变量，包括 Wi-Fi 网络名称、密码和处理器速度等。hello\_world 示例项目会以默认配置运行，因此可以跳过使用 menuconfig 进行项目配置这一步骤。

## 2.3 编译工程

idf.py build

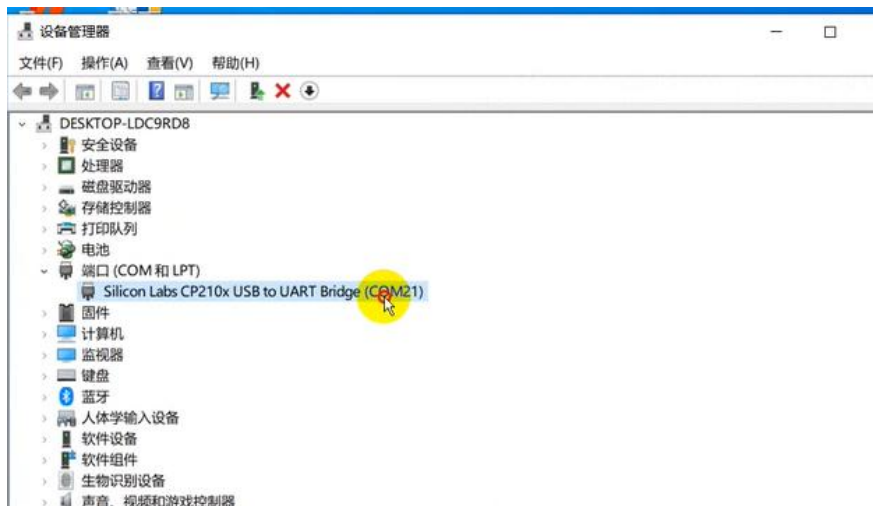
```
D:\ESP_IDF\esp-idf\hello_world>idf.py build
Executing action: all (aliases: build)
Running ninja in directory d:\esp_idf\esp-idf\hello_world\build
Executing "ninja all"...
[75/990] Generating ../../partition_table/partition-table.bin
Partition table binary generated. Contents:
*****
# ESP-IDF Partition Table
# Name, Type, SubType, Offset, Size, Flags
nvs, data, nvs, 0x9000, 24K,
phy_init, data, phy, 0xf000, 4K,
factory, app, factory, 0x10000, 1M,
*****
[351/990] Building C object esp-idf/wpa_supplicant/CMakeFi...wpa_supplicant.dir/src/crypto/crypto_mbedtls-bignum.c.obj
```

## 2.4 硬件连接

用 USB 线将 ESP32 模组与电脑连接

## 2.5 查看端口

在开始菜单中选择设备管理器查看当前端口号：



## 2.6 工程烧录

进行烧录：`idf.py -p COM21 flash`

```
Flash will be erased from 0x00001000 to 0x00007fff...
Flash will be erased from 0x00010000 to 0x00037fff...
Compressed 3072 bytes to 103...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (103 compressed) at 0x00008000 in 0.0 seconds (effective 513.4 kbit/s)...
Hash of data verified.
Compressed 25008 bytes to 15404...
Writing at 0x00001000... (100 %)
Wrote 25008 bytes (15404 compressed) at 0x00001000 in 0.8 seconds (effective 263.6 kbit/s)...
Hash of data verified.
Compressed 161072 bytes to 84558...
Writing at 0x00010000... (16 %)
Writing at 0x0001a9a6... (33 %)
Writing at 0x00020145... (50 %)
Writing at 0x000259e8... (66 %)
Writing at 0x0002ef3f... (83 %)
Writing at 0x0003643e... (100 %)
Wrote 161072 bytes (84558 compressed) at 0x00010000 in 2.2 seconds (effective 585.0 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
Done
```

烧录成功！

## 2.7 监视工程

查看监视器：`idf.py -p COM21 monitor`

```
I (276) heap_init: At 3FFD4350 len 0001BCB0 (111 KiB): D/IRAM
I (282) heap_init: At 4008AA7C len 00015584 (85 KiB): IRAM
I (289) spi_flash: detected chip: generic
I (293) spi_flash: flash io: dio
W (297) spi_flash: Detected size(4096k) larger than the size in the binary image header(2048k). Using
nary image header.
I (311) cpu_start: Starting scheduler on PRO CPU.
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 1, 2MB external flash
Minimum free heap size: 291576 bytes
Restarting in 10 seconds...
Restarting in 9 seconds...
Restarting in 8 seconds...
```

成功打印 `hello_world`.

## 3 VS Code 的使用

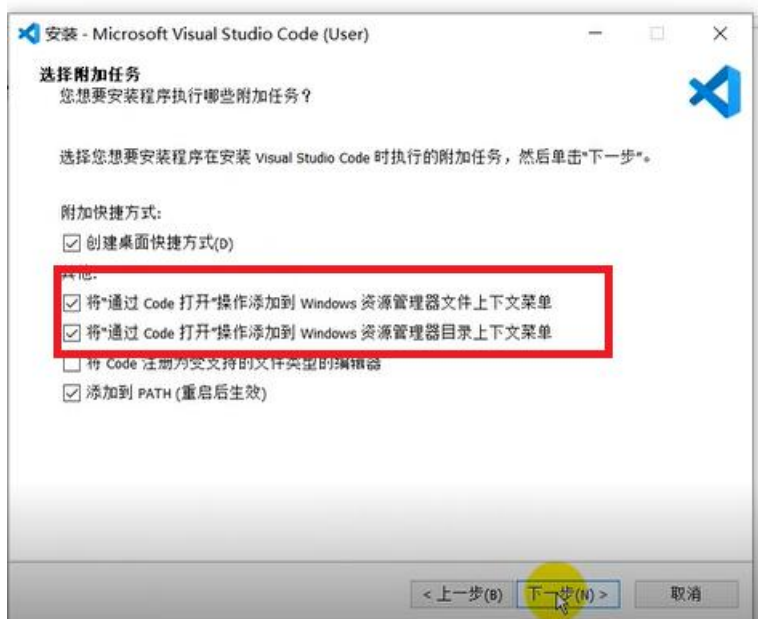
### 3.1 VS Code 的下载与安装

#### 3.1.1 下载 VS Code

官网下载地址：<https://code.visualstudio.com/Download#>

#### 3.1.2 安装 VS Code

勾选这两项可将 VS Code 添加至右键菜单





## 3.2 VS Code 的使用

### 3.2.1 移植脚本

(1) 创建一个文本文档，将下列内容复制粘贴（需适当修改）

```
@echo off

Set home=%cd%

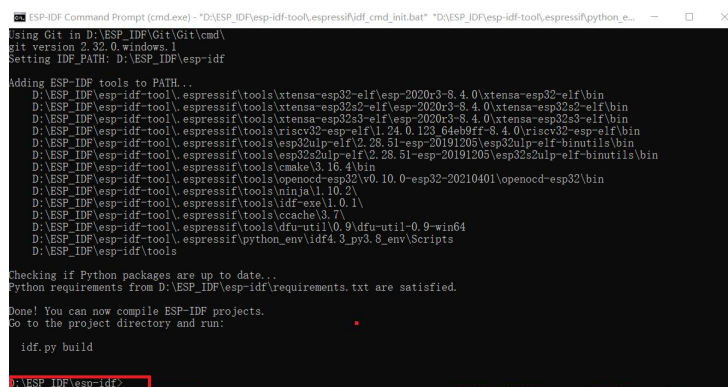
cd D:\ESP_IDF\esp-idf

C:\Windows\system32\cmd.exe /k "D:\ESP_IDF\esp-idf-tool\espressif\idf_cmd_init.bat" "D:\ESP_IDF\esp-idf-tool\espressif\python_env\idf_py3.8_env\Scripts" "D:\ESP_IDF\Git\Git\cmd\""
```

/\*

home 为储存当前路径的变量名，可改可不改。

cd 后的路径为 esp-idf 路径



```
ESP-IDF Command Prompt (cmd.exe) - "D:\ESP_IDF\esp-idf-tool\espressif\idf_cmd_init.bat" "D:\ESP_IDF\esp-idf-tool\espressif\python_e...
Using Git in D:\ESP_IDF\Git\Git\cmd\
git version 2.32.0.windows.1
Setting IDF_PATH: D:\ESP_IDF\esp-idf

Adding ESP-IDF tools to PATH...
D:\ESP_IDF\esp-idf-tool\espressif\tools\xtensa-esp32-elf\esp-2020r3-8.4.0\xtensa-esp32-elf\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\xtensa-esp32s2-elf\esp-2020r3-8.4.0\xtensa-esp32s2-elf\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\xtensa-esp32s3-elf\esp-2020r3-8.4.0\xtensa-esp32s3-elf\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\riscv32-esp-elf\1.24.0.123.64eb9ff-8.4.0\riscv32-esp-elf\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\esp32ulp-elf\2.28.51-esp-20191205\esp32ulp-elf\binutils\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\esp32s2ulp-elf\2.28.51-esp-20191205\esp32s2ulp-elf\binutils\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\cmake\3.16.4\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\openocd-esp32\v0.10.0-esp32-20210401\openocd-esp32\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\ninja\1.10.2\
D:\ESP_IDF\esp-idf-tool\espressif\tools\idf-exe\1.0.1\
D:\ESP_IDF\esp-idf-tool\espressif\tools\ccache\3.7\
D:\ESP_IDF\esp-idf-tool\espressif\tools\dfu-util\0.9\dfu-util-0.9-win64
D:\ESP_IDF\esp-idf-tool\espressif\python_env\idf4.3_py3.8_env\Scripts
D:\ESP_IDF\esp-idf\tools

Checking if Python packages are up to date...
Python requirements from D:\ESP_IDF\esp-idf\requirements.txt are satisfied.

Done! You can now compile ESP-IDF projects.
Go to the project directory and run:

idf.py build

D:\ESP_IDF\esp-idf>
```

C:\Windows\s..... 这一句指令替换为 ESP-IDF Command Prompt (cmd.exe) 的目标

右键点击 ESP-IDF Command Prompt (cmd.exe)，选择属性，将目标中语句进行复制替换。

完成后将文档改名为 get\_idf.bat，将其放入 ESP-IDF Command Prompt (cmd.exe) 所在位置。





\*/

(2) 创建另一个文本文档，复制粘贴以下内容：

```
@echo off
cd %home%
```

/\* home 变量应与第一个脚本保持一致 \*/

将文档改名为 go\_home.bat, 将其放入 ESP-IDF Command Prompt (cmd.exe) 所在位置。

### 3.2.2 运行 hello\_world 示例

(1) 使用 VS Code 打开 hello\_world 工程文件夹, 打开一个新终端, 运行 get\_idf:

```
PS D:\ESP_IDF\esp32\hello_world> get_idf
Setting PYTHONNOUSERSITE, was not set
Using Python in D:\ESP_IDF\esp-idf-tool\espressif\python_env\idf_py3.8_env\Scripts
Python 3.8.7
Using Git in D:\ESP_IDF\Git\Git\cmd\
git version 2.32.0.windows.1
Setting IDF_PATH: D:\ESP_IDF\esp-idf
Adding ESP-IDF tools to PATH...
D:\ESP_IDF\esp-idf-tool\espressif\tools\xtensa-esp32-elf\esp-2020r3-8.4.0\xtensa-esp32-elf\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\xtensa-esp32s2-elf\esp-2020r3-8.4.0\xtensa-esp32s2-elf\bin
D:\ESP_IDF\esp-idf-tool\espressif\tools\xtensa-esp32s3-elf\esp-2020r3-8.4.0\xtensa-esp32s3-elf\bin
```

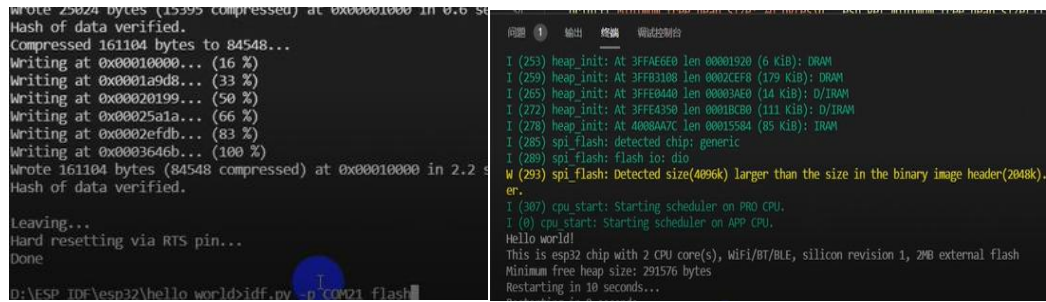
(2) 运行 go\_home, 返回 hello\_world 目录下:

```
D:\ESP_IDF\esp-idf>go_home
D:\ESP_IDF\esp32\hello_world>
```

(3) 工程编译

```
D:\ESP_IDF\esp-idf>go_home
D:\ESP_IDF\esp32\hello_world>idf.py build
Executing action: all (aliases: build)
Running cmake in directory d:\esp_idf\esp32\hello_world\build
Executing "cmake"
CKED=1 -DESP_PLATFORM=1 -DIDF_TARGET=esp32 -DCCACHE_ENABLE=1 d:\esp_idf\esp32\hello_world\build\idf.py build
```

#### (4) 工程烧录与监视



```
wrote 25024 bytes (15395 compressed) at 0x00001000 in 0.6 s
Hash of data verified.
Compressed 161104 bytes to 84548...
Writing at 0x00010000... (16 %)
Writing at 0x0001a9d8... (33 %)
Writing at 0x00020199... (50 %)
Writing at 0x00025a1a... (66 %)
Writing at 0x0002efdb... (83 %)
Writing at 0x0003646b... (100 %)
wrote 161104 bytes (84548 compressed) at 0x00010000 in 2.2 s
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
Done
D:\ESP-IDF\esp32\hello_world\idf.py -c C:\21_flash
```

```
I (253) heap_init: At 3FFAE6E0 len 00001920 (6 KiB): DRAM
I (259) heap_init: At 3FFB3108 len 0002CEF8 (179 KiB): DRAM
I (265) heap_init: At 3FFE0440 len 00003AE0 (14 KiB): D/IRAM
I (272) heap_init: At 3FFE4350 len 0001BCB0 (111 KiB): D/IRAM
I (278) heap_init: At 40080A7C len 00015584 (85 KiB): IRAM
I (285) spi_flash: detected chip: generic
I (289) spi_flash: flash io: dio
W (293) spi_flash: Detected size(4096k) larger than the size in the binary image header(2048k).
er.
I (307) cpu_start: Starting scheduler on PRO CPU.
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 1, 2M external flash
Minimum free heap size: 291576 bytes
Restarting in 10 seconds...
```

成功打印 hello\_world.

## 4 参考视频

(注：为保证视频质量，演示视频并未录制声音，同时对于一些长时间下载过程进行了缩略录制，视频仅作为本文档参考文件。)

### 4.1 安装 VS Code



ESP32系列教程 (01) : VS Code的

### 4.2 安装 git 与 python



ESP32系列教程 (02) : git与pythc

### 4.3 安装 ESP-IDF 工具



ESP32系列教程 (03) : ESP-IDF工:

### 4.4 VS Code 运行 hello\_world 示例



ESP32系列教程 (04) : VS Code运

### 4.5 Windows 运行 hello\_world 示例



ESP32系列教程 (05) : Windows~

## 5 后 记

### 5.1 注意事项

- (1) git 版本没有具体要求
- (2) python 版本建议为 3.8（至少在 3.8 之上）
- (3) 本文档适用的 ESP-IDF 工具安装器版本为 2.5，其它版本仅供参考
- (4) 脚本移植时要进行相应的替换，命令间的空格最好重新设置，防止因字符格式的转变而插入其它字符

### 5.2 相关建议

本文档使用 ESP-IDF 工具安装器的方式在 Windows 下构建 esp-idf 环境，同时为了更方便于编写代码而通过移植脚本的方式引入 VS Code 的使用，将代码编写与编译一体化，消除频繁切换界面的麻烦，该方式适用于习惯在 Windows 环境下使用命令行进行开发的开发者。

除了 ESP-IDF 工具安装器的方式外，乐鑫在 VS Code 上面提供了一个 Espressif IDF 插件用来在 VS Code 上构建 esp-idf 环境，但在实际安装中发现该插件仍存在一些缺陷，其工具链设置较为繁琐，对该插件有兴趣的开发者可参考下方博客：

<https://my.oschina.net/u/4364212/blog/4948170>