

ESP32 系列教程之六： WSL 搭建 esp-idf 环境

Date: 2021.7.10

Version: V1.0

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。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
2021.7.10	V1.0	Amiliya	首次发布

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

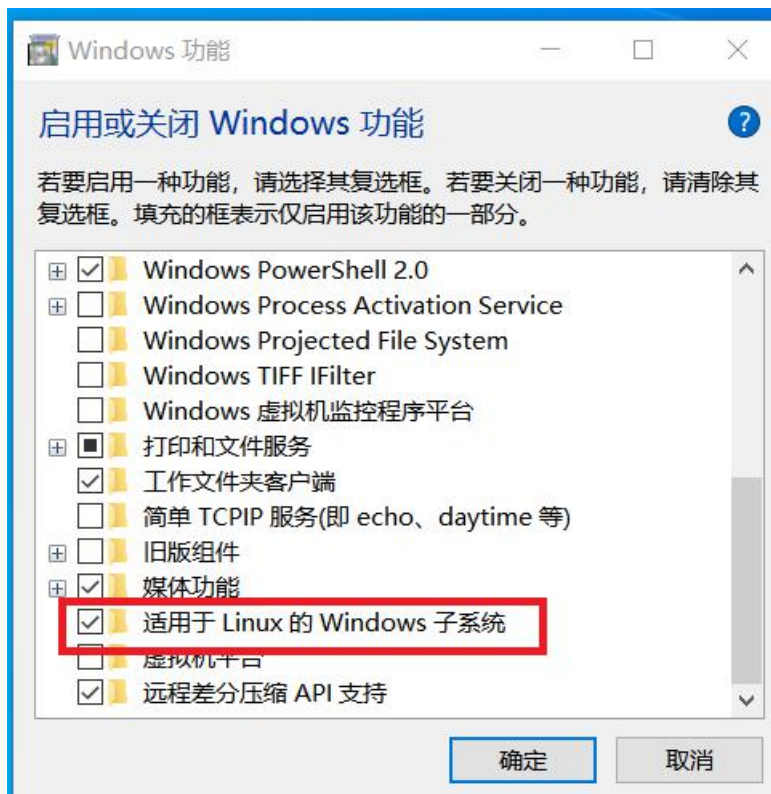
目 录

1 WSL 搭建 esp-idf 环境.....	1
1.1 开启 WSL.....	1
1.2 准备 WSL.....	1
1.3 工具准备.....	3
1.4 获取 ESP-IDF.....	3
1.5 安装工具包.....	3
1.6 将 esp-idf 环境加入环境变量.....	4
2 运行 hello_world 示例.....	5
2.1 拷贝工程.....	5
2.2 工程配置.....	5
2.3 编译工程.....	6
2.4 硬件连接.....	6
2.5 查看端口.....	6
2.6 工程烧录.....	7
2.7 监视工程.....	7
3 参考视频.....	8
3.1 WSL 搭建 esp-idf 环境.....	8
3.2 VS Code 远程连接 WSL.....	8
4 后 记.....	9
4.1 注意事项.....	9
4.2 常见问题.....	9
4.2.1 ESP-IDF 的获取.....	9
4.2.2 子模组缺失.....	10
4.2.3 工具链设置.....	11

1 WSL 搭建 esp-idf 环境

1.1 开启 WSL

快捷键 **win+s** 搜索：“启用或关闭 Windows 功能”，勾选“适用于 Linux 的 Windows 子系统”，确定后重启电脑。



1.2 准备 WSL

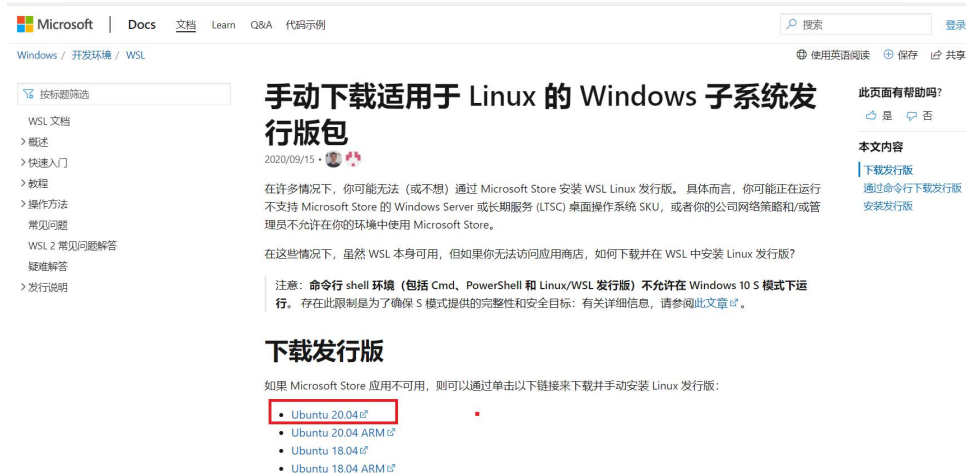
下载 Ubuntu20，以下两种方式任选一种（推荐方式一）。

方式一：快捷键 **win+s** 搜索：“Microsoft Store”，搜索 Ubuntu，下载 Ubuntu20.04



方式二：前往下方网址下载 Ubuntu20.04：（下载很慢，且容易失败）

<https://docs.microsoft.com/zh-cn/windows/wsl/install-manual>



1.3 工具准备

(1) 更新“应用市场”

```
sudo apt-get update
```

(2) 安装 esp-idf 相关应用

```
sudo apt-get install git wget flex bison gperf python3 python3-pip python3-setuptools cmake ninja-build
ccache libffi-dev libssl-dev dfu-util libusb-1.0-0
```

(3) 设置默认 python

```
sudo ln -s /usr/bin/python3 /usr/bin/python
```

(4) 查看当前 python 版本（大于 3.6）

```
python
```

```
amiliya@DESKTOP-LDC9RD8:~$ sudo ln -s /usr/bin/python3 /usr/bin/python
amiliya@DESKTOP-LDC9RD8:~$ python
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
[2]+  Stopped                  python
amiliya@DESKTOP-LDC9RD8:~$
```

1.4 获取 ESP-IDF

使用命令下载 ESP-IDF

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

（注：-b release/v4.3 表示 esp-idf 下载版本，需要下载其它版本或版本介绍请移步至乐鑫官网查看具体介绍）

```
remote: Total 64103 (delta 1131), reused 1853 (delta 1038), pack-reused 62067
Receiving objects: 100% (64103/64103), 17.15 MiB | 71.00 KiB/s, done.
Resolving deltas: 100% (40070/40070), done.
Cloning into '/home/amiliya/esp-idf/components/nghttp/nghttp2/third-party/neverbleed'...
remote: Enumerating objects: 234, done.
remote: Total 234 (delta 0), reused 0 (delta 0), pack-reused 234
Receiving objects: 100% (234/234), 83.16 KiB | 13.86 MiB/s, done.
Resolving deltas: 100% (144/144), done.
Submodule path 'components/nghttp/nghttp2/third-party/mruby': checked out '7c91efc1ffda769a5f1a872e646c82b00698f1b8'
Submodule path 'components/nghttp/nghttp2/third-party/neverbleed': checked out 'b967ca054f48a36f82d8fcd32e54ec5144f2751'
Submodule path 'components/protobuf-c': checked out 'da1a65feac4ad72f612aab99f487056fbcf5c1a'
Submodule path 'components/spiffs/spiffs': checked out 'f5e26c4e933189593a71c6b82cda381a7b21e41c'
Submodule path 'components/tinysub/tinyusb': checked out '334a95fac52a607150157ae5199a19a11f843982'
Submodule path 'components/unity/unity': checked out '7d2bf62b7e6afaf38153041a9d53c21aee9a9e25'
Submodule path 'examples/build_system/cmake/import_lib/main/lib/tinyxml2': checked out '7e8e249990ec491ec15990cf95b6d871a66cf64a'
Submodule path 'examples/peripherals/secure_element/atecc608_ecdsa/components/esp-cryptoauthlib': checked out 'c3d3a69021cfec3236ca2c0b63be4048ec6643a4'
amiliya@DESKTOP-LDC9RD8:~$
```

1.5 安装工具包

进入 esp-idf 文件夹，运行 `./install.sh`

若安装工具时速度慢，可在执行 `./install.sh` 命令前执行以下命令，优先选择 Espressif 下载服务器进行下载：`export IDF_GITHUB_ASSETS="dl.espressif.com/github_assets"`

运行脚本激活 esp-idf

```
./export.sh
```

1.6 将 esp-idf 环境加入环境变量

复制并粘贴以下命令到 shell 配置文件中（`.profile`，`.bashrc`，`.zprofile` 等）

```
alias get_idf='. $HOME/esp_4.3/esp-idf/export.sh'
```

（注：文件路径需要正确，如文件夹名称 `esp_4.3`）

以 `.bashrc` 为例：

打开 `.bashrc` 文件

```
vim .bashrc
```

添加命令：（需要不同版本的 IDF 时都可以加入命令，只要变量名 `get_idf` 不同即可）

```
if ! shopt -oq posix; then
if [ -f /usr/share/bash-completion/bash_completion ]; then
. /usr/share/bash-completion/bash_completion
elif [ -f /etc/bash_completion ]; then
. /etc/bash_completion
fi
fi
alias get_idf='. $HOME/esp_4.3/esp-idf/export.sh'
```

保存关闭，重启 Ubuntu 或使用命令刷新配置文件：`source ~/.bashrc`

最终效果如下：执行 `get_idf` 便可以让端口进入 esp-idf 环境：

```
amiliya@ubuntu:~$ get_idf
Setting IDF_PATH to '/home/amiliya/esp_4.3/esp-idf'
Detecting the Python interpreter
Checking "python" ...
Python 3.6.9
"python" has been detected
Adding ESP-IDF tools to PATH...
Using Python interpreter in /home/amiliya/.espressif/python_env/idf4.3_py3.6_env/bin/python
Checking if Python packages are up to date...
Python requirements from /home/amiliya/esp_4.3/esp-idf/requirements.txt are satisfied.
Added the following directories to PATH:
/home/amiliya/esp_4.3/esp-idf/components/esptool_py/esptool
/home/amiliya/esp_4.3/esp-idf/components/espressif/espressif
/home/amiliya/esp_4.3/esp-idf/components/partition_table
/home/amiliya/esp_4.3/esp-idf/components/app_update
/home/amiliya/.espressif/tools/xtensa-esp32-elf/esp-2021r1-8.4.0/xtensa-esp32-elf/bin
/home/amiliya/.espressif/tools/xtensa-esp32s2-elf/esp-2021r1-8.4.0/xtensa-esp32s2-elf/bin
/home/amiliya/.espressif/tools/xtensa-esp32s3-elf/esp-2021r1-8.4.0/xtensa-esp32s3-elf/bin
/home/amiliya/.espressif/tools/riscv32-esp-elf/esp-2021r1-8.4.0/riscv32-esp-elf/bin
/home/amiliya/.espressif/tools/esp32ulp-elf/2.28.51-esp-20191205/esp32ulp-elf-binutils/bin
/home/amiliya/.espressif/tools/esp32s2ulp-elf/2.28.51-esp-20191205/esp32s2ulp-elf-binutils/bin
/home/amiliya/.espressif/tools/openocd-esp32/v0.10.0-esp32-20210401/openocd-esp32/bin
/home/amiliya/.espressif/python_env/idf4.3_py3.6_env/bin
/home/amiliya/esp_4.3/esp-idf/tools
Done! You can now compile ESP-IDF projects.
Go to the project directory and run:

idf.py build

Traceback (most recent call last):
  File "/home/amiliya/esp_4.3/esp-idf/tools/idf.py", line 812, in <module>
    main()
  File "/home/amiliya/esp_4.3/esp-idf/tools/idf.py", line 730, in main
    cli(sys.argv[1:], prog_name=PROG, complete_var='_IDF.PY_COMPLETE')
```

2 运行 hello_world 示例

2.1 拷贝工程

将 `esp-idf/examples/get-started/` 目录下的 `hello_world` 示例拷贝到 `esp_4.3` 下

```
cp esp-idf/examples/get-started/hello_world/ ./ -r
```

```
amiliya@ubuntu:~/esp_4.3$ cp esp-idf/examples/get-started/hello_world/ ./ -r
amiliya@ubuntu:~/esp_4.3$ ls
esp-idf  hello_world
amiliya@ubuntu:~/esp_4.3$
```

2.2 工程配置

进入 `hello_world` 示例，进行工程配置：

```
cd ~/esp_4.3/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）)

```
idf.py menuconfig
```

如果之前的步骤都正确，则会显示下面的菜单：

```
(Top)
Espressif IoT Development Framework Configuration
SDK tool configuration --->
Build type --->
Application manager --->
Bootloader config --->
Security features --->
Serial flasher config --->
Partition Table --->
Compiler options --->
Component config --->
Compatibility options --->

[Space/Enter] Toggle/enter  [ESC] Leave menu          [S] Save
[O] Load                   [?] Symbol info           [/] Jump to symbol
[F] Toggle show-help mode  [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

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

```
amiliya@ubuntu:~/esp_4.3$ cd hello_world/
amiliya@ubuntu:~/esp_4.3/hello_world$ idf.py build
Executing action: all (aliases: build)
Running cmake in directory /home/amiliya/esp_4.3/hello_world/build
Executing "cmake -G Ninja -DPYTHON_DEPS_CHECKED=1 -DESP_PLATFORM=1 -DCCACHE_ENABLE=0 /home/amiliya/esp_4.3/hello_world"...
-- Found Glt: /usr/bin/glt (found version "2.17.1")
-- IDF_TARGET not set, using default target: esp32
-- The C compiler identification is GNU 8.4.0
-- The CXX compiler identification is GNU 8.4.0
-- The ASM compiler identification is GNU
-- Found assembler: /home/amiliya/.espressif/tools/xtensa-esp32-elf/esp-2021r1-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf-gcc
-- Check for working C compiler: /home/amiliya/.espressif/tools/xtensa-esp32-elf/esp-2021r1-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf-gcc
-- Check for working C compiler: /home/amiliya/.espressif/tools/xtensa-esp32-elf/esp-2021r1-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf-gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /home/amiliya/.espressif/tools/xtensa-esp32-elf/esp-2021r1-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf-g++
-- Check for working CXX compiler: /home/amiliya/.espressif/tools/xtensa-esp32-elf/esp-2021r1-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf-g++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done

merged 1 ELF section
generated /home/amiliya/esp_4.3/hello_world/build/bootloader/bootloader.bin
983/983 Generating binary image from built executable
esptool.py v2.1-dev
merged 2 ELF sections
generated /home/amiliya/esp_4.3/hello_world/build/hello-world.bin

project build complete. To flash, run this command:
/home/amiliya/.espressif/python_env/idf4.3_py3.6_env/bin/python ../esp-idf/components/esptool.py/esptool.py -p (PORT) -b 460800 --before default_reset --after hard_reset --c
ip esp32 --write_flash --flash_mode dio --flash_size detect --flash_freq 40m 0x1000 build/bootloader/bootloader.bin 0x8000 build/partition_table/partition-table.bin 0x10000 build/he
lo-world.bin
r run 'idf.py -p (PORT) flash'
amiliya@ubuntu:~/esp_4.3/hello_world$
```

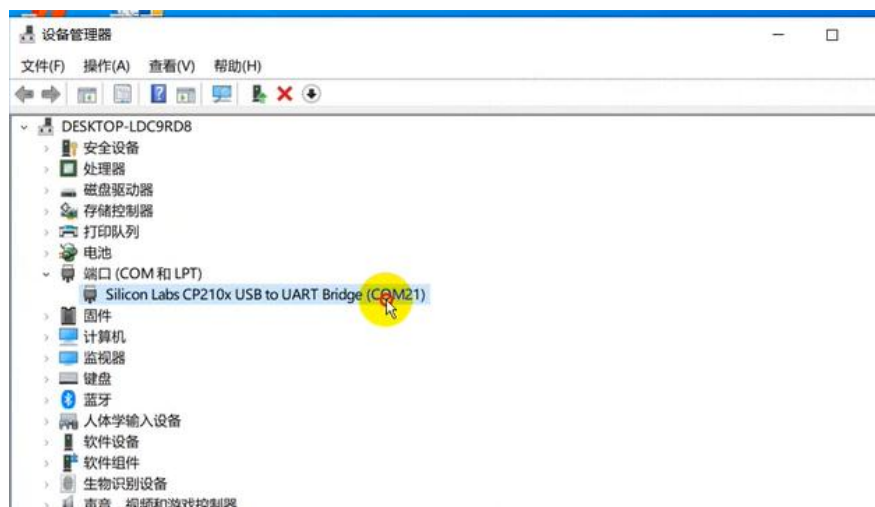
从机中移出或按 Ctrl+Alt,

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 /dev/ttyS21 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 0.1 seconds (effective 429.2 kbit/s)...
Hash of data verified.
Compressed 25152 bytes to 15450...
Writing at 0x00001000... (100 %)
Wrote 25152 bytes (15450 compressed) at 0x00001000 in 0.7 seconds (effective 278.0 kbit/s)...
Hash of data verified.
Compressed 162976 bytes to 85495...
Writing at 0x00010000... (16 %)
Writing at 0x0001acae... (33 %)
Writing at 0x0002048a... (50 %)
Writing at 0x00025c9f... (66 %)
Writing at 0x0002f4e3... (83 %)
Writing at 0x000360b0... (100 %)
Wrote 162976 bytes (85495 compressed) at 0x00010000 in 2.4 seconds (effective 538.9 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done
amiliya@ubuntu:~/esp_4.3/hello_world$
```

烧录成功！

(WSL 中的端口与 Windows 端口的对应关系：COM21 - S21)

2.7 监视工程

查看监视器：`idf.py -p /dev/ttyS21 monitor`

```
W (296) spi_flash: Detected size(4096k) larger than the size in the binary image header(2048k). Using the s
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 cores, WiFi/BT/BLE, silicon revision 1, 2MB external flash
Restarting in 10 seconds...
Restarting in 9 seconds...
Restarting in 8 seconds...
Restarting in 7 seconds...
Restarting in 6 seconds...
Restarting in 5 seconds...
Restarting in 4 seconds...
```

成功打印 `hello_world`.

3 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。）

3.1 WSL 搭建 esp-idf 环境



ESP32系列教程（
10）WSL搭建esp

3.2 VS Code 远程连接 WSL



ESP32系列教程（
11）：VS Code远

4 后 记

4.1 注意事项

- (1) Ubuntu 版本建议在 18.0 以上，搭建 esp-idf 环境需要保证网络通畅
- (2) python 版本必须保证 3.6 以上且为系统默认 python
- (3) 环境变量可添加多个，可通过调用不同变量名选择不同版本的 IDF
- (4) 工程配置中选择目标芯片的过程对某些芯片是必要的，如 ESP32-C3
- (5) 进行 ESP32-C3 的开发必需使用最新版（V4.3）IDF 的支持

4.2 常见问题

由于 esp-idf 环境搭建过程中由于各种原因容易遇到各种问题，此章节整理了一些常见的问题以供大家参考，其它相关问题可提交至意见提交邮箱。

4.2.1 ESP-IDF 的获取

部分开发者由于网络等原因无法从 github 获取 esp-idf，可参考下方步骤从国内的 gitee 获取 esp-idf。

(1) 获取 esp-idf

```
git clone -b release/v4.3 https://gitee.com/EspressifSystems/esp-idf.git
```

```
amiliya@DESKTOP-LDC9RDS:~/esp$ git clone -b release/v4.3 https://gitee.com/EspressifSystems/esp-idf.git
正克隆到 'esp-idf'...
remote: Enumerating objects: 30044, done.
remote: Counting objects: 100% (30044/30044), done.
remote: Compressing objects: 100% (11812/11812), done.
remote: Total 250881 (delta 20441), reused 24664 (delta 17190), pack-reused 220837
接收对象中: 100% (250881/250881), 139.96 MiB | 5.57 MiB/s, 完成。
处理 delta 中: 100% (185486/185486), 完成。
正在更新文件: 100% (8306/8306), 完成。
amiliya@DESKTOP-LDC9RDS:~/esp$ cd esp-idf/
```

(2) 下载工具

```
git clone https://gitee.com/EspressifSystems/esp-gitee-tools.git
```

```
amiliya@DESKTOP-LDC9RDS:~/esp$ git clone https://gitee.com/EspressifSystems/esp-gitee-tools.git
正克隆到 'esp-gitee-tools'...
remote: Enumerating objects: 48, done.
remote: Counting objects: 100% (48/48), done.
remote: Compressing objects: 100% (47/47), done.
remote: Total 48 (delta 19), reused 0 (delta 0), pack-reused 0
展开对象中: 100% (48/48), 15.36 KiB | 249.00 KiB/s, 完成。
amiliya@DESKTOP-LDC9RDS:~/esp$ ls
esp-gitee-tools  esp-idf
amiliya@DESKTOP-LDC9RDS:~/esp$
```

(3) 下载子模块和工具链

进入工具文件夹，保存路径

```
cd esp-gitee-tools
```

```
export EGT_PATH=$(pwd)
```

进入 esp-idf 文件夹，下载子模块和工具链

```
cd ~/esp-idf/
```

```
$EGT_PATH/submodule-update.sh
```

```
amiliya@DESKTOP-LDC9RD8:~$ cd esp/esp-gitee-tools/
amiliya@DESKTOP-LDC9RD8:~/esp/esp-gitee-tools$ export EGT_PATH=$(pwd)
amiliya@DESKTOP-LDC9RD8:~/esp/esp-gitee-tools$ cd ../esp-idf/
amiliya@DESKTOP-LDC9RD8:~/esp/esp-idf$ $EGT_PATH/submodule-update.sh
子模块 'components/asio/asio' (https://gitee.com/espressif/asio.git) 已对路径 'components/asio/asio' 注册
子模块 'components/bootloader/subproject/components/micro-ecc/micro-ecc' (https://gitee.com/kmackay/micro-ecc.git) 已对
路径 'components/bootloader/subproject/components/micro-ecc/micro-ecc' 注册
子模块 'components/bt/controller/lib_esp32' (https://gitee.com/espressif/esp32-bt-lib.git) 已对路径 'components/bt/controller/lib_esp32' 注册
子模块 'components/bt/controller/lib_esp32c3_family' (https://gitee.com/espressif/esp32c3-bt-lib.git) 已对路径 'components/bt/controller/lib_esp32c3_family' 注册
子模块 'components/bt/host/nimble/nimble' (https://gitee.com/espressif/esp-nimble.git) 已对路径 'components/bt/host/nimble/nimble' 注册
子模块 'components/cbor/tinychor' (https://gitee.com/intel/tinychor.git) 已对路径 'components/cbor/tinychor' 注册
子模块 'components/cmocker/CMock' (https://gitee.com/ThrowTheSwitch/CMock.git) 已对路径 'components/cmocker/CMock' 注册
子模块 'components/coap/libcoap' (https://gitee.com/obgm/libcoap.git) 已对路径 'components/coap/libcoap' 注册
子模块 'components/esp_wifi/lib' (https://gitee.com/espressif/esp32-wifi-lib.git) 已对路径 'components/esp_wifi/lib' 注册
子模块 'components/esptool_py/esptool' (https://gitee.com/espressif/esptool.git) 已对路径 'components/esptool_py/esptool' 注册
```

```
$EGT_PATH/install.sh
```

```
amiliya@DESKTOP-LDC9RD8:~/esp/esp-idf$ $EGT_PATH/install.sh
Installing ESP-IDF tools
Installing tools: xtensa-esp32-elf, xtensa-esp32s2-elf, xtensa-esp32s3-elf, riscv32-esp-elf, esp32ulp-elf, esp32s2ulp-elf,
f, openocd-esp32
Skipping xtensa-esp32-elf@esp-2021r1-8.4.0 (already installed)
Skipping xtensa-esp32s2-elf@esp-2021r1-8.4.0 (already installed)
Skipping xtensa-esp32s3-elf@esp-2021r1-8.4.0 (already installed)
Skipping riscv32-esp-elf@esp-2021r1-8.4.0 (already installed)
Skipping esp32ulp-elf@2.28.51-esp-20191205 (already installed)
Skipping esp32s2ulp-elf@2.28.51-esp-20191205 (already installed)
Skipping openocd-esp32@v0.10.0-esp32-20210401 (already installed)
Installing Python environment and packages
Python 3.8.10
pip 21.1.3 from /home/amiliya/.espressif/python_env/idf4.3_py3.8_env/lib/python3.8/site-packages/pip (python 3.8)
Installing Python dependencies from /home/amiliya/.espressif/python_env/idf4.3_py3.8_env/lib/python3.8/site-packages/requirements.txt
Requirement already satisfied: brotli in /home/amiliya/.espressif/python_env/idf4.3_py3.8_env/lib/python3.8/site-packages
Requirement already satisfied: greenlet==0.4.14 in /home/amiliya/.espressif/python_env/idf4.3_py3.8_env/lib/python3.8/site-packages
Requirement already satisfied: MarkupSafe==2.0 in /home/amiliya/.espressif/python_env/idf4.3_py3.8_env/lib/python3.8/site-packages
Requirement already satisfied: Jinja2==2.4 in /home/amiliya/.espressif/python_env/idf4.3_py3.8_env/lib/python3.8/site-packages
All done! You can now run:
. /home/amiliya/esp/esp-idf/export.sh
amiliya@DESKTOP-LDC9RD8:~/esp/esp-idf$
```

4.2.2 子模组缺失

绝大多数开发者所遇到的无法启动菜单、工程编译失败等问题都源于 ESP-IDF 子模组未下载完全，因此保证子模组的完全下载对完整搭建 esp-idf 环境至关重要。

```
-- Found assembler: /home/lu/.espressif/tools/xtensa-esp32-elf/esp-2020r3-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf
-- Check for working C compiler: /home/lu/.espressif/tools/xtensa-esp32-elf/esp-2020r3-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf
-- Check for working C compiler: /home/lu/.espressif/tools/xtensa-esp32-elf/esp-2020r3-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /home/lu/.espressif/tools/xtensa-esp32-elf/esp-2020r3-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf
-- Check for working CXX compiler: /home/lu/.espressif/tools/xtensa-esp32-elf/esp-2020r3-8.4.0/xtensa-esp32-elf/bin/xtensa-esp32-elf
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Project is not inside a git repository, or git repository has no commits; will not use 'git describe' to determine project version
-- Project version: 1
-- Building ESP-IDF components for target esp32
-- Project sdkconfig file /home/lu/esp_4.1/hello_world/sdkconfig
CMake Error at /home/lu/esp_4.1/esp-idf/tools/cmake/component.cmake:302 (message):
  Include directory
  "/home/lu/esp_4.1/esp-idf/components/nbdtls/nbdtls/include" is not a
  directory.
Call Stack (most recent call first):
  /home/lu/esp_4.1/esp-idf/tools/cmake/component.cmake:478 (_component_add_include_dirs)
  /home/lu/esp_4.1/esp-idf/components/nbdtls/CMakeLists.txt:3 (idf_component_register)

-- Configuring incomplete, errors occurred!
See also "/home/lu/esp_4.1/hello_world/build/CMakeFiles/CMakeOutput.log".
cmake failed with exit code 1
lu@lu-ThinkPad-T430:~/esp_4.1/hello_world$
```

缺少相应子模块时的错误 log

在 esp-idf 文件夹下使用以下命令下载缺失子模组

```
git submodule update --init --recursive
```

```
amiliya@ubuntu:~/esp_4.1/esp-idf$ git submodule update --init --recursive
正克隆到 '/home/amiliya/esp_4.1/esp-idf/components/nghttp/nghttp2/third-party/mruby'...
子模组路径 'components/nghttp/nghttp2/third-party/mruby': 检出 '22464fe5a0a10f2b077eaba109ce1e912e4a77de'
子模组路径 'components/nghttp/nghttp2/third-party/neverbleed': 检出 'da5c2ab419a3bb8a4cc6c37a6c7f3e4bd4b41134'
amiliya@ubuntu:~/esp_4.1/esp-idf$
```

若命令长时间无响应可取消后重试，此命令也可用于检测子模组是否下载完全（下载完全则直接返回命令行），也可进入缺失子模组的目录下单独克隆该子模组。

4.2.3 工具链设置

少部分开发者在搭建环境时会遇到此类问题，可依据错误提示 log 做出相应处理，以下仅供参考。

```
Requirement already satisfied: Werkzeug<1.0, >=0.7 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from python-so
idf/requirements.txt (line 21)) (0.16.0)
Requirement already satisfied: python-engineio<4, >=3.13.0 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from python-so
idf/requirements.txt (line 21)) (3.14.2)
Requirement already satisfied: preparser in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from cffi>=1.12->cryptography>
requirements.txt (line 11)) (2.20)
Requirement already satisfied: Werkzeug<1.0, >=0.7 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from Flask<1.0, >=0.12.
ilya/esp-idf/requirements.txt (line 15)) (0.16.1)
Requirement already satisfied: itsdangerous<=0.21 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from Flask<1.0, >=0.12.
ilya/esp-idf/requirements.txt (line 15)) (2.0.1)
Requirement already satisfied: Jinja2<=2.4 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from Flask<1.0, >=0.12.2->gdbgc
sp-idf/requirements.txt (line 15)) (3.0.1)
Requirement already satisfied: brotli in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from Flask-Compress<2.0, >=1.4.0->g
idf/requirements.txt (line 15)) (1.0.9)
Requirement already satisfied: greenlet<=0.4.14 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from gevent<2.0, >=1.2.2->g
ilya/esp-idf/requirements.txt (line 15)) (1.1.0)
Requirement already satisfied: MarkupSafe<=2.0 in /home/amiliya/.espressif/python_env/idf4.4_py3.8_env/lib/python3.8/site-packages (from Jinja2<=2.4->Flask<1
/home/amiliya/.espressif/python_env/idf4.4_py3.8_env/bin/python -m pip install --upgrade pip command.
WARNING: You are using pip version 21.1.2, however version 21.3.1 is available.
You should consider upgrading via the '/home/amiliya/.espressif/python_env/idf4.4_py3.8_env/bin/python -m pip install --upgrade pip' command.
./export.sh
```

按照提示执行相应命令可消除警告（pip 版本问题）

4.2.3 烧录问题

在 WSL 下进行烧录时出现以下错误：

```
cd /home/amiliya/esp-idf/components/esptool.py && /usr/bin/cmake -D IDF_PATH="/home/amiliya/esp-idf" -D SERIAL_TOOL="/home/amiliya/.espressif/python_env/idf4.4_py3.8_env/bin/python
/home/amiliya/esp-idf/components/esptool.py --chip esp32 -D SERIAL_TOOL_ARGS="--before=before_defmt_flash --after=after_defmt_flash --write_flash &flash_args" -D WORKING_DIRECTORY
"/home/amiliya/esp-idf/examples/get-started/hello_world/build" -P /home/amiliya/esp-idf/components/esptool.py/serial_tool.cmake
inja: build stopped: subcommand failed.
inja failed with exit code 1
amiliya@ubuntu:~/esp-idf/examples/get-started/hello_world$
```

按照提示执行相应命令或更改波特率可正常进行烧录

```
idf.py -p /dev/ttyS21 -b 115200 flash
```