Set up ESP32-IDF development environment

1、Introduction to ESP32 development environment

The ESP32 development environment supports Windows, Linux and Mac platforms.

On Linux and Mac platforms, you need to download and install the ESP-IDF development environment from the source code.

There are a variety of development environments available on the Windows platform, including the official ESP-IDF environment, the arduino IDE software development environment, and the VS Code+PlatformIO IDE development method.

Here we take the Ubuntu system as an example to download and install the ESP-IDF development environment from the source code.

2、Install dependencies

Open the Ubuntu system terminal and run the following commands to install related dependencies.

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

3、Download ESP-IDF

Open the Ubuntu system terminal and run the following command to download the esp-idf-v5.1.2 version

```
mkdir -p ~/esp
cd ~/esp
git clone -b v5.1.2 --recursive https://github.com/espressif/esp-idf.git
```

Set the chip esp32s3 supported by the tool.

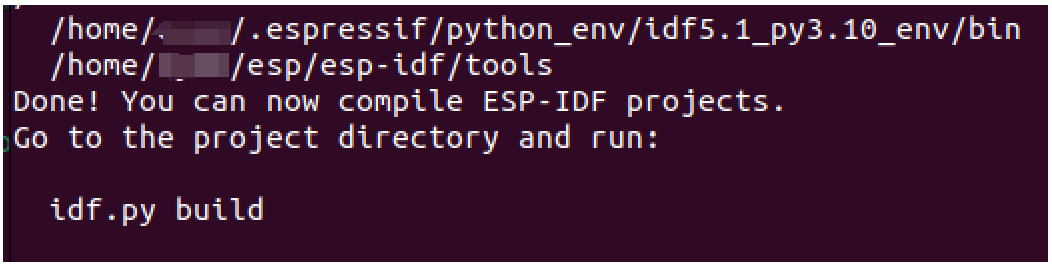
```
cd esp-idf
./install.sh esp32s3
```

4、Activate the ESP-IDF development environment

Run the following command in the esp-idf tool directory

```
source ~/esp/esp-idf/export.sh
```

Note: Every time you open a new terminal, you need to activate the ESP-IDF development environment before you can compile the ESP-IDF project.



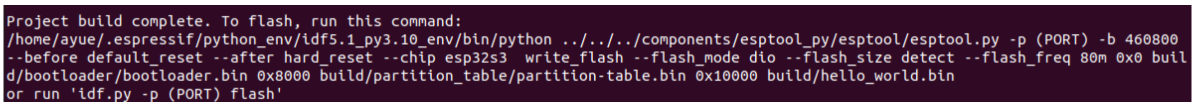
```
  /home/[redacted]/.espressif/python_env/idf5.1_py3.10_env/bin
  /home/[redacted]/esp/esp-idf/tools
Done! You can now compile ESP-IDF projects.
Go to the project directory and run:

  idf.py build
```

5、Compile and flash firmware

Connect the microROS control board to the virtual machine/computer, and then test and compile the hello_world program to generate firmware.

```
cd examples/get-started/hello_world
idf.py set-target esp32s3
idf.py build
```

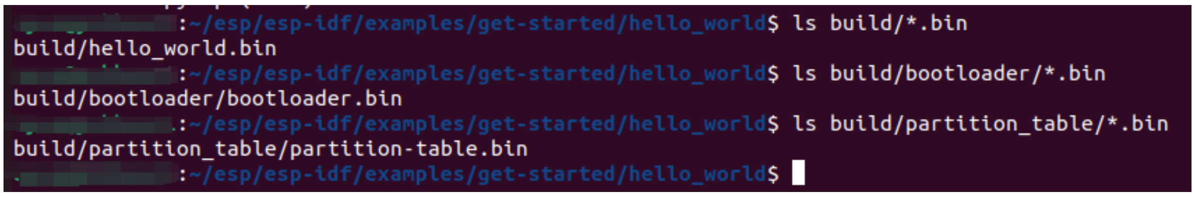
When you see the following prompt, it means the compilation passed.



```
Project build complete. To flash, run this command:
/home/ayue/.espressif/python_env/idf5.1_py3.10_env/bin/python ../../../components/esptool_py/esptool/esptool.py -p (PORT) -b 460800
--before default_reset --after hard_reset --chip esp32s3  write_flash --flash_mode dio --flash_size detect --flash_freq 80m 0x0 buil
d/bootloader/bootloader.bin 0x8000 build/partition_table/partition-table.bin 0x10000 build/hello_world.bin
or run 'idf.py -p (PORT) flash'
```

View the generated firmware

```
ls build/*.bin
ls build/bootloader/*.bin
ls build/partition_table/*.bin
```



```
[redacted]:~/esp/esp-idf/examples/get-started/hello_world$ ls build/*.bin
build/hello_world.bin
[redacted]:~/esp/esp-idf/examples/get-started/hello_world$ ls build/bootloader/*.bin
build/bootloader/bootloader.bin
[redacted]:~/esp/esp-idf/examples/get-started/hello_world$ ls build/partition_table/*.bin
build/partition_table/partition-table.bin
[redacted]:~/esp/esp-idf/examples/get-started/hello_world$ 
```

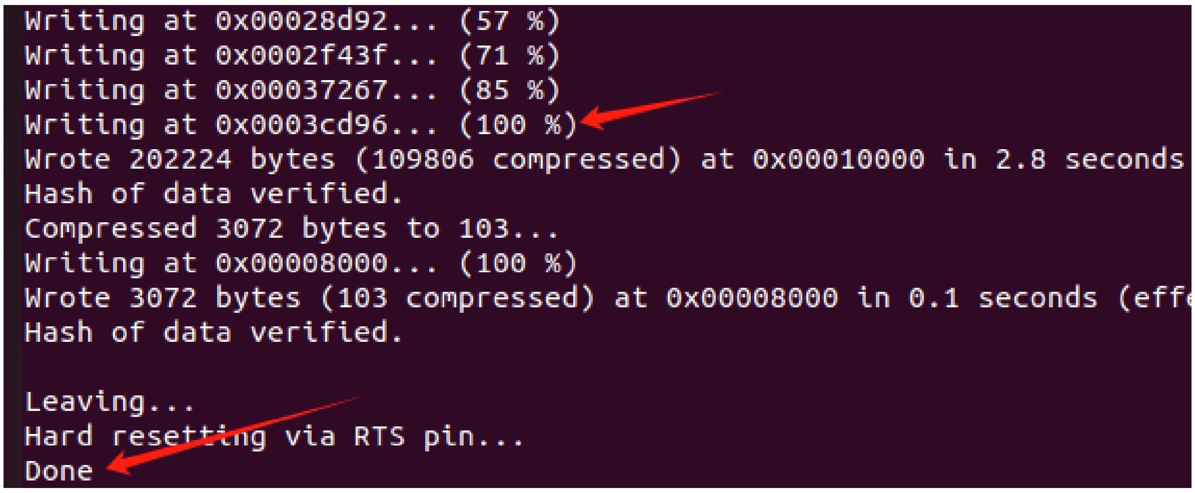
Run the following command to burn the generated firmware to the microROS control board.

```
idf.py flash
```

Before using this command to burn, you need to confirm that the computer is only connected to one serial port device. If multiple serial port devices are connected, you can manually specify the serial port number in order to distinguish it. For example, specify the serial port number for burning as /dev/ttyUSB0

```
idf.py flash -p /dev/ttyUSB0
```

When the download progress reaches 100% and Done is displayed, the burning is completed.



```
Writing at 0x00028d92... (57 %)
Writing at 0x0002f43f... (71 %)
Writing at 0x00037267... (85 %)
Writing at 0x0003cd96... (100 %)
Wrote 202224 bytes (109806 compressed) at 0x00010000 in 2.8 seconds
Hash of data verified.
Compressed 3072 bytes to 103...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (103 compressed) at 0x00008000 in 0.1 seconds (effe
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done
```

Open the serial port simulator

```
idf.py monitor
```

If you need to exit the serial port simulator, please press **Ctrl+]** to exit.

6、Shortcut command

If you need to compile, burn, and open the serial port simulator, please enter the following commands

```
idf.py build flash monitor
```

7、ESP-IDF API reference

Open the following link to view the ESP32S3 official API reference content

```
https://docs.espressif.com/projects/esp-idf/en/v5.1.2/esp32s3/api-
reference/index.html
```