ESP8266
Quick Start Guide

Version 1.4

About This Guide

This document is a quick start guide to ESP8266. The document is structured as follows.

Chapter	Title	Content
Chapter 1	Configuring the Development Board ESP-LAUNCHER	Introduction to the ESP8266 development board ESP-LAUNCHER, and how to download firmware to the board and run it.
Chapter 2	Compiling ESP8266_NONOS_SDK	Compiling the ESP8266_NONOS_SDK in the virtual machine.
Chapter 3	Compiling ESP8266_RTOS_SDK	Compiling the ESP8266_RTOS_SDK.
Chapter 4	Common Debug Methods	Common debugging methods and sample codes.
Chapter 5	Downloading Firmware into the ESP-WROOM-02	Instructions on how to download firmware with ESP-WROOM-02.
Appendix A	Learning Resources	ESP8266-related must-read documents and must-have resources.

Release Notes

Date	Version	Release notes
2016.08	V1.0	First release.
2016.11	V1.1	Added Appendix A—Learning Resources.
2017.01	V1.2	Added two Github links to RTOS and non-OS SDK sample code in Appendix A.2—Must-Have Resources.
2017.02	V1.3	Updated the link to the OVA image file in Section 2.1.
2017.05	V1.4	Updated Chapter 1.

Table of Contents

1. Configuring the Development Board ESP-LAUNCHER 1

1.1. Hardware Preparation 1

1.2. Software Preparation 1

1.3. Downloading Firmware into the ESP-LAUNCHER...2

2. Compiling ESP8266_NONOS_SDK..7

2.1. Downloading the Virtual Machine 7

2.2. Setting up Development Environment 7

2.3. Compiling Process...9

3. Compiling ESP8266_RTOS_SDK 12

3.1. Compiling Process...12

3.2. ESP8266_RTOS_SDK Architecture...15

4. Common Debug Methods 16

4.1. Common Debug Methods...16

4.1.1. Add UART Output Logs...16

4.1.2. Debug Fatal Exception 16

5. Downloading Firmware into the ESP-WROOM-02...17

A. Appendix—Learning Resources 19

A.1. Must-Read Documents 19

A.2. Must-Have Resources..20

1. Configuring the Development Board ESP-LAUNCHER

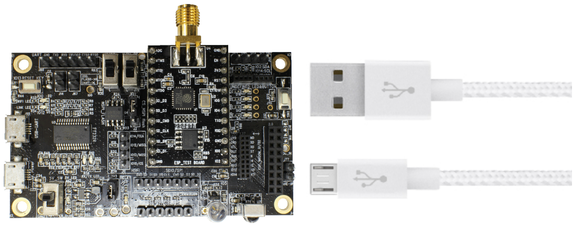
1.1. Hardware Preparation

To start developing applications for the ESP8266, users need the following hardware and its corresponding software tools:

- One of the ESP8266 Hardware Development Kits (HDKs) specified below:
  - ESP8266's development board, ESP-LAUNCHER, as shown in Table 1-1.
  - ESP8266's module, ESP-WROOM-02, as described in ***Chapter 5***.
- PC for programming: Windows XP or Windows 7 OS is recommended, with enough RAM to run a Linux virtual machine.
- Micro-USB cable.

> 📖 ***Notes:***
>
> - *When users deploy third-party development boards or modules that integrate ESP8266, they should use development firmware provided by the corresponding manufacturers.*
> - *To purchase ESP-WROOM-02 or ESP-LAUNCHER, please visit Espressif's official online store at: https://espressif.taobao.com,*

Table 1-1. The ESP-LAUNCHER

• 1 ESP-LAUNCHER • 1 Micro-USB cable	

> ⚠ ***Notice:***
>
> *The ESP8266 Wi-Fi module requires 3.3V power supply capable of delivering a peak current of 500 mA.*

1.2. Software Preparation

- ESP8266's Flash Download Tool
  - Download it from: *ESP8266 Flash Download Tools*
- ESP8266's SDK
  - Download the SDK from: *ESP8266 SDK*

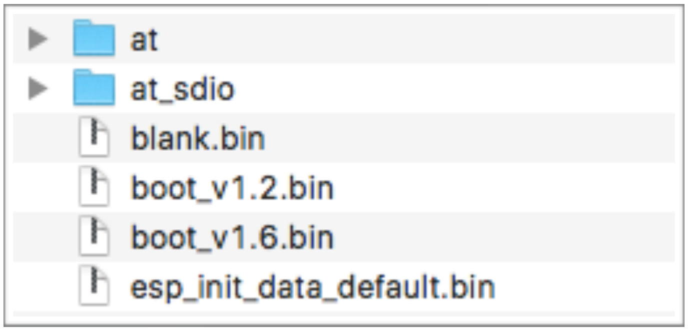
- The official AT firmware (***ESP8266_NONOS_SDK\bin\at***) can be downloaded into the ESP-LAUNCHER by referring to the BIN locations included in the ***Readme*** file which is in the same directory. For instructions on downloading the firmware into ESP-LAUNCHER, please refer to ***Section 1.3***.

- UART terminal emulator tool

  The default baud rate of ESP8266 is74880, therefore, UART tools that can support the default baud rate are recommended. Please note that certain USB-UART converters may not support all baud rates if users use a third-party development board.

1.3. Downloading Firmware into the ESP-LAUNCHER

1. We use the ***ESP8266_NONOS_SDK_V2.0.0_16_07_19*** as an example. The AT firmware binaries are located in ***ESP8266_NONOS_SDK_V2.0.0_16_07_19\ESP8266_NONOS_SDK\bin***.



at
at_sdio
blank.bin
boot_v1.2.bin
boot_v1.6.bin
esp_init_data_default.bin

Figure 1-1. ESP8266_NONOS_SDK BIN Folder

2. Settings of the development board, ESP-LAUNCHER.

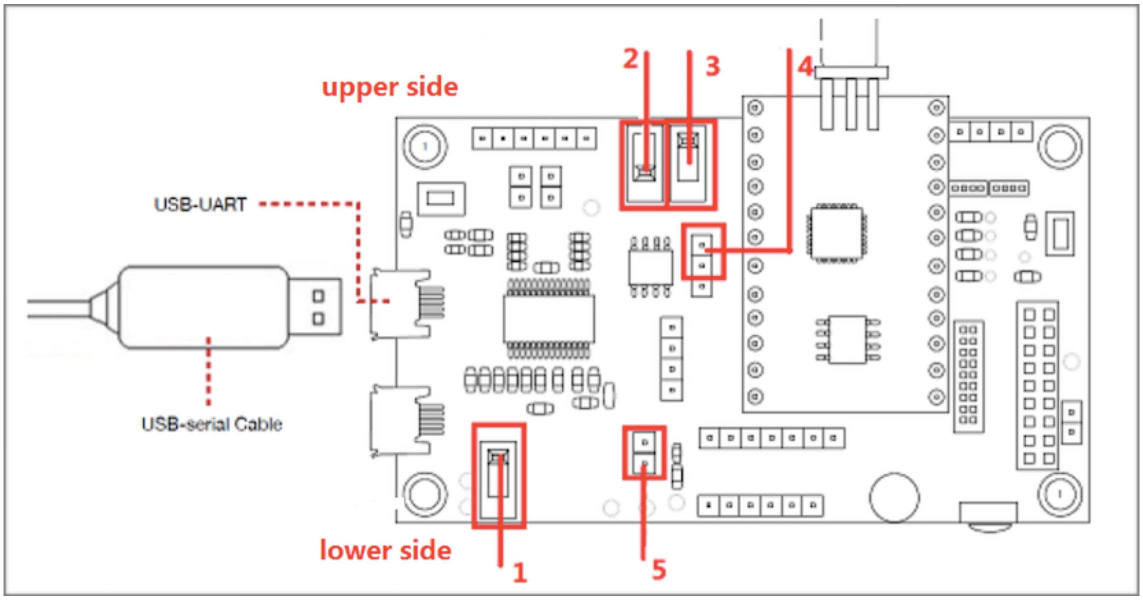


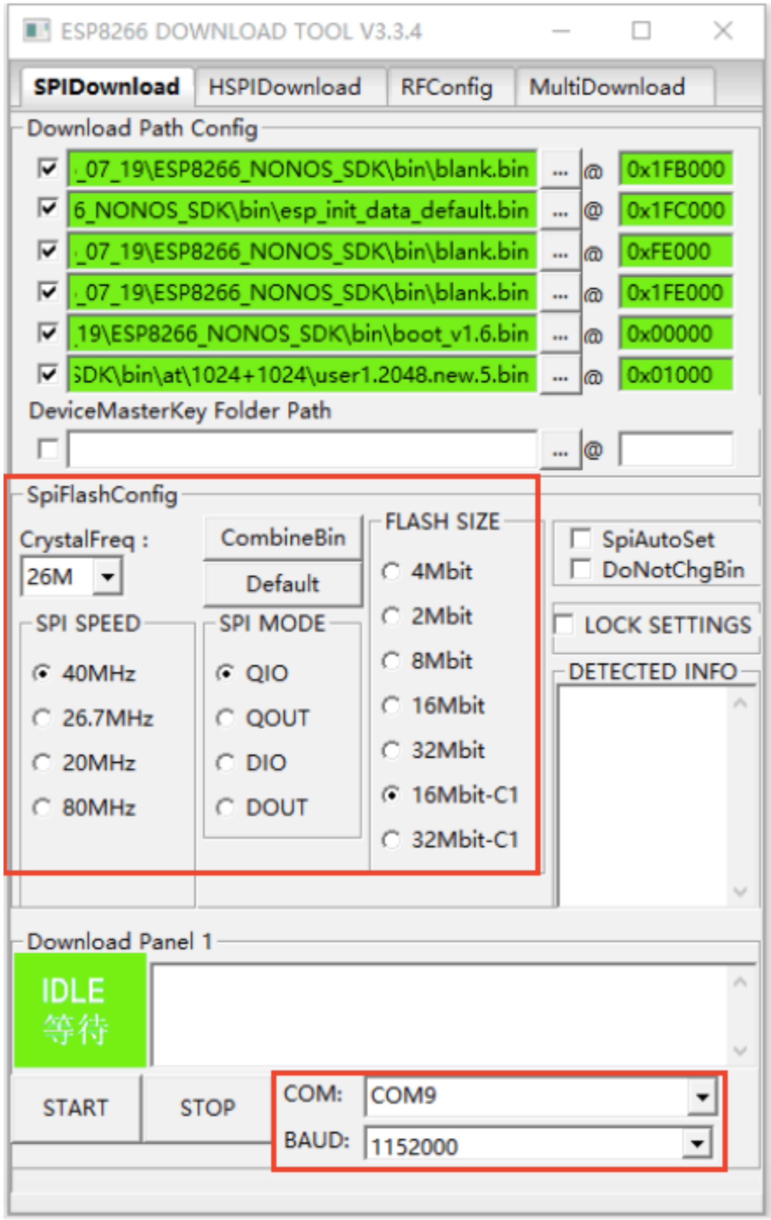
Figure 1-2. The ESP-LAUNCHER

- ***switch 1***: toggle to the lower side;

- ***switch 2***: toggle to the lower side;
- ***switch 3***: toggle to the upper side;
- ***pin4***: put a jumper cap on the two pins above;
- ***pin5***: put a jumper cap on it.

3. Use a Micro-USB cable to connect ESP-LAUNCHER to the PC. The UART driver needs to be installed on the PC.

4. Double-click on ***ESPFlashDownloadTool_v3.3.4.exe*** to run the ESP8266 Flash Download Tool on the PC.



Download Path Config

☑ _07_19\ESP8266_NONOS_SDK\bin\blank.bin ... @ 0x1FB000
☑ 6_NONOS_SDK\bin\esp_init_data_default.bin ... @ 0x1FC000
☑ _07_19\ESP8266_NONOS_SDK\bin\blank.bin ... @ 0xFE000
☑ _07_19\ESP8266_NONOS_SDK\bin\blank.bin ... @ 0x1FE000
☑ 19\ESP8266_NONOS_SDK\bin\boot_v1.6.bin ... @ 0x00000
☑ SDK\bin\at\1024+1024\user1.2048.new.5.bin ... @ 0x01000

DeviceMasterKey Folder Path

☐ ... @

SpiFlashConfig

CrystalFreq : 26M

SPI SPEED: ● 40MHz ○ 26.7MHz ○ 20MHz ○ 80MHz

SPI MODE: ● QIO ○ QOUT ○ DIO ○ DOUT

FLASH SIZE: ○ 4Mbit ○ 2Mbit ○ 8Mbit ○ 16Mbit ○ 32Mbit ● 16Mbit-C1 ○ 32Mbit-C1

☐ SpiAutoSet
☐ DoNotChgBin
☐ LOCK SETTINGS

Download Panel 1

IDLE
等待

COM: COM9
BAUD: 1152000

Figure 1-3. ESP8266 Flash Download Tool

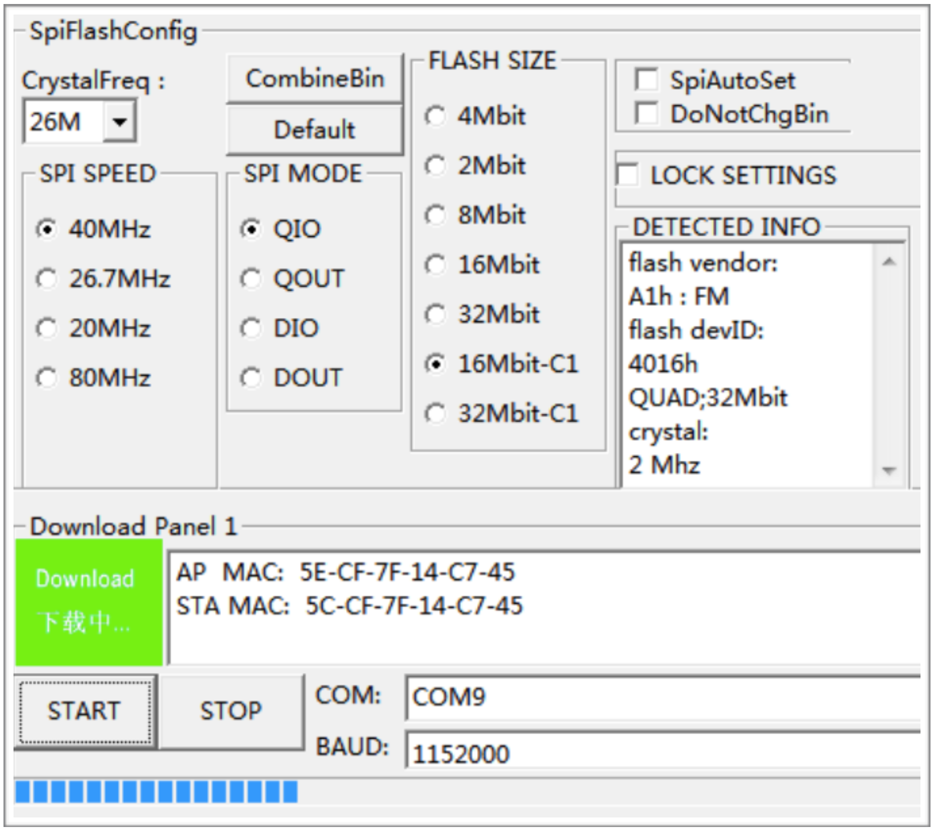
Figure 1-3 uses ***16Mbit-C1*** (1024+1024 map) flash as an example. The download addresses are shown in Table 1-2.

Table 1-2. Download AT Binaries for 16 Mbit-C1 Flash Map

BIN	Address	Description
blank.bin	0x1FB000	It initializes the RF_CAL parameter area.
esp_init_data_default.bin	0x1FC000	It stores the default RF parameter values and has to be downloaded into the flash at least once. If the RF_CAL parameter area is initialized, this BIN has to be downloaded too.
blank.bin	0xFE000	It initializes the user parameter area.
blank.bin	0x1FE000	It initializes the system parameter area.
boot.bin	0x00000	It is the main program located in ***\bin\at***.
user1.2048.new.5.bin	0x01000	It is the main program located in ***\bin\at\1024+1024***.

> 📖 ***Note:***
>
> *The **SpiFlashConfig** and the **COM** on the ESP8266 Flash Download Tool should be set according to the actual situation.*

5. Click the ***START*** button and wait for the ESP-LAUNCHER to power up.
6. Power on the ESP-LAUNCHER by toggling ***switch 1*** as is shown in Figure 1-2 to the upper side. Toggle ***switch 2*** as is shown in Figure 1-2 to the lower side to enter the download mode.
7. The ESP8266 Flash Download Tool will start to download AT firmware into the ESP-LAUNCHER. The ***DETECTED INFO*** area will display the information of the flash chip on ESP-LAUNCHER.



SpiFlashConfig

CrystalFreq : 26M

CombineBin

Default

SPI SPEED
● 40MHz
○ 26.7MHz
○ 20MHz
○ 80MHz

SPI MODE
● QIO
○ QOUT
○ DIO
○ DOUT

FLASH SIZE
○ 4Mbit
○ 2Mbit
○ 8Mbit
○ 16Mbit
○ 32Mbit
● 16Mbit-C1
○ 32Mbit-C1

□ SpiAutoSet
□ DoNotChgBin

□ LOCK SETTINGS

DETECTED INFO

```
flash vendor:
A1h : FM
flash devID:
4016h
QUAD;32Mbit
crystal:
2 Mhz
```

Download Panel 1

Download
下载中...

```
AP  MAC:  5E-CF-7F-14-C7-45
STA MAC:  5C-CF-7F-14-C7-45
```

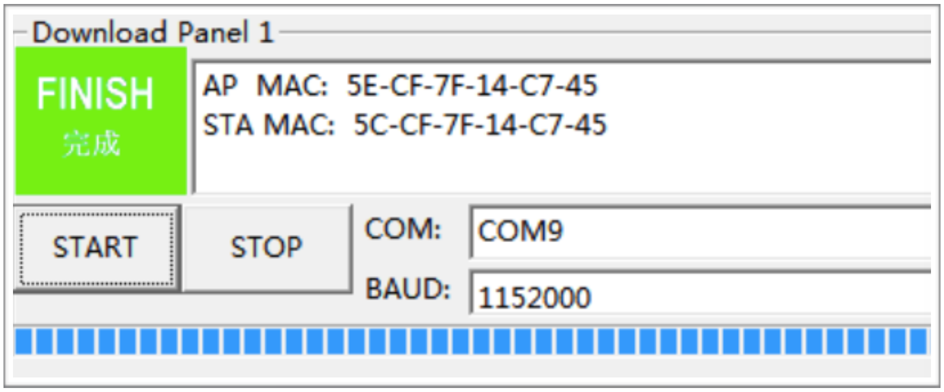
START STOP

COM: COM9

BAUD: 1152000

Figure 1-4. ESP8266 Flash Download Tool - Downloading Firmware

8. After the download is finished, as Figure 1-5 shows, toggle ***switch 1*** to the lower side to power off the ESP-LAUNCHER.



Download Panel 1

FINISH
完成

```
AP  MAC:  5E-CF-7F-14-C7-45
STA MAC:  5C-CF-7F-14-C7-45
```

START STOP

COM: COM9

BAUD: 1152000

Figure 1-5. ESP8266 Flash Download Tool–Finishing Downloading Firmware

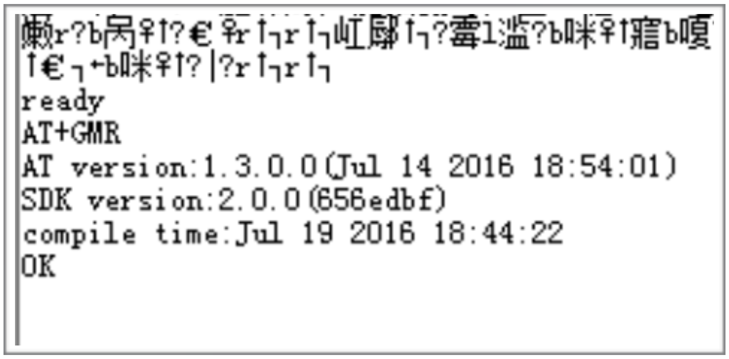
9. Open the UART tool on the PC, set the baud rate to 115200, and configure new line mode.

> ***Note:***
>
> *When using AT commands, then a baud rate of 115200 and new line mode are needed.*

10. Set the ESP8266 to working mode by toggling ***switch 2*** to the upper side. Then toggle ***switch 1*** to the upper side to power on the ESP-LAUNCHER.

The log print may look garbled, as shown in Figure 1-6 (which is normal, because the default power-on baud rate of ESP8266 is 74880). The “ready” message indicates that ESP-LAUNCHER is working successfully.

11. Input command `AT+GMR` and press ***Enter*** button. The information of the AT firmware will appear, as Figure 1-7 shows.



```
懒r?b呙♀↑?€ ♀r↑┐r↑┐屸膠↑┐?毒1滥?b咪♀↑瘖b嗄
↑€┐+b咪♀↑?|?r↑┐r↑┐
ready
AT+GMR
AT version:1.3.0.0(Jul 14 2016 18:54:01)
SDK version:2.0.0(656edbf)
compile time:Jul 19 2016 18:44:22
OK
```

Figure 1-6. AT Logs on the UART Tool

For more AT commands and examples, please refer to documents: *ESP8266 AT Instruction Set* and *ESP8266 AT Command Examples*.

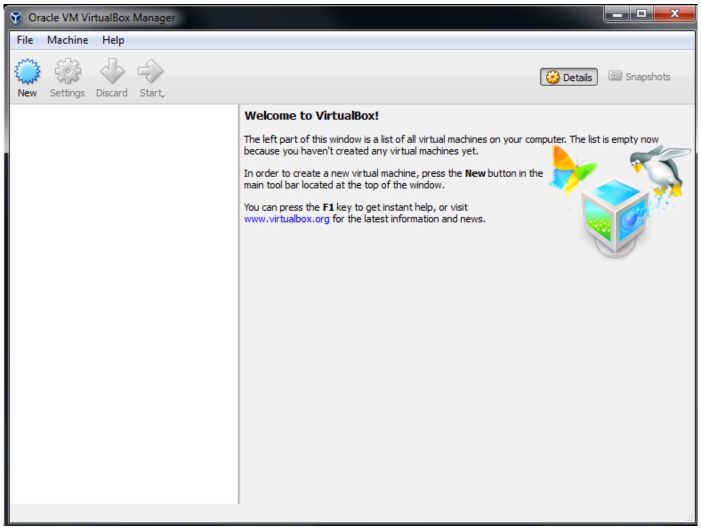
2. Compiling ESP8266_NONOS_SDK

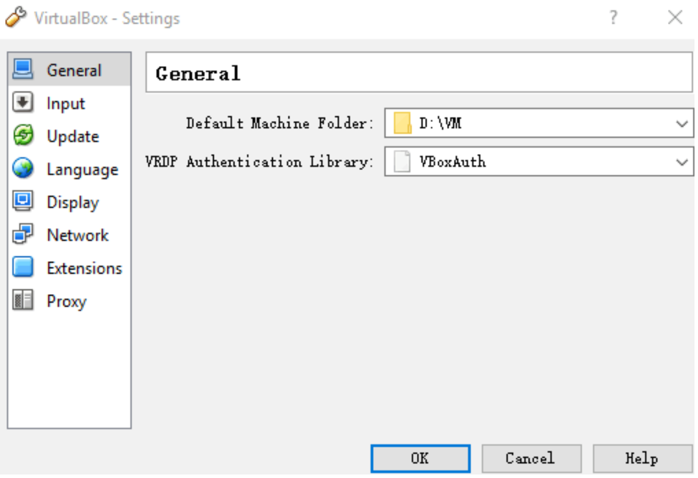
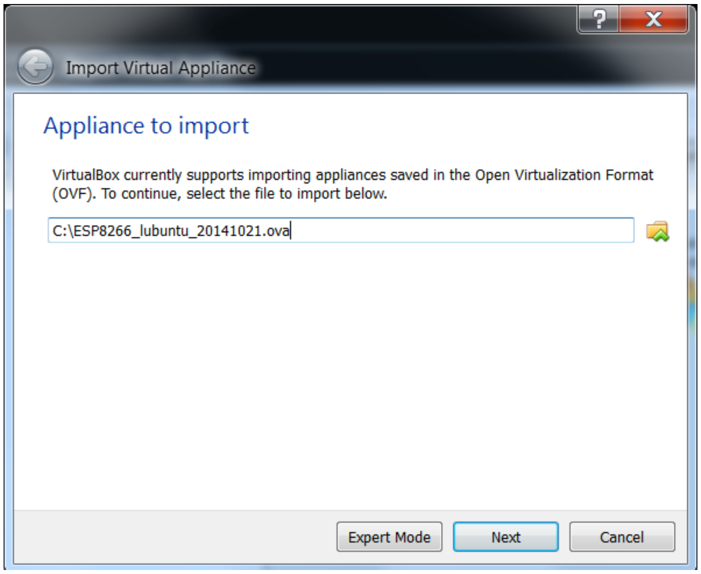
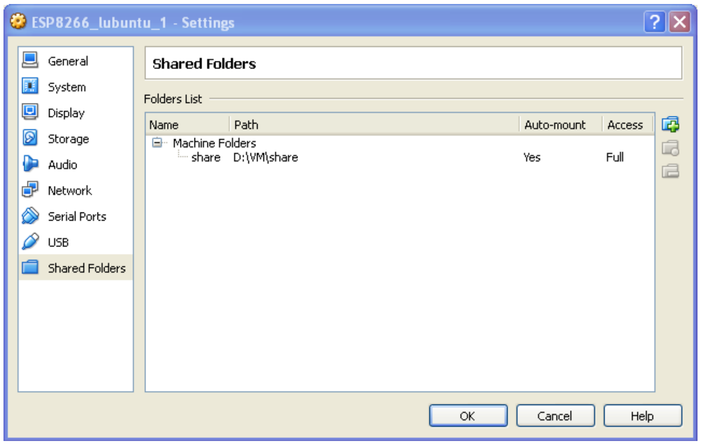
This chapter describes how to compile ***ESP8266_NONOS_SDK*** by using the AT demo as an example.

2.1. Downloading the Virtual Machine

1. PC: Windows XP or Windows 7 OS is recommended.
2. The development environment provided by Espressif Systems is based on Lubuntu. We also provide a virtual image of the development environment that can be run on VirtualBox.
    - Download ***VirtualBox-5.0.16-105871-Win.exe*** from: *https://www.virtualbox.org/wiki/Downloads*
    - Download ***ESP8266_lubuntu_20141021.ova*** from: *http://downloads.espressif.com/FB/ESP8266_GCC.zip*

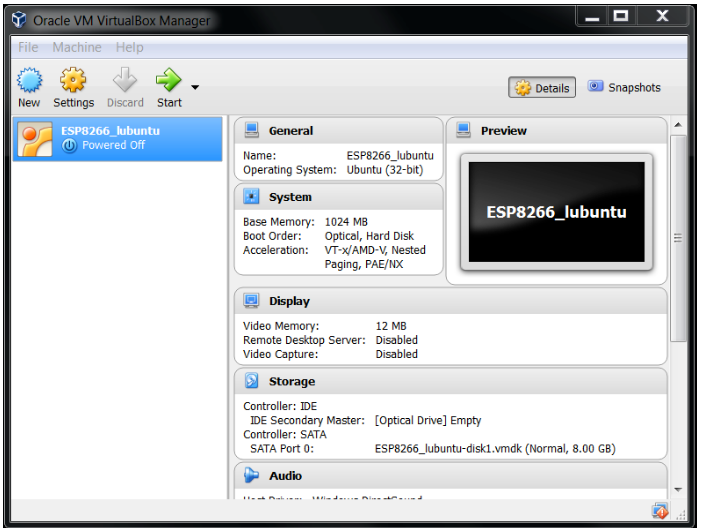
2.2. Setting up Development Environment

Steps	Results
1. Start Windows OS and install the virtual machine.	
• Double-click on ***VirtualBox-5.0.16-105871-Win.exe*** and install VirtualBox. 📖 ***Note:*** *VirtualBox has different versions. We are using Windows V.5.0.16 as an example.* • Double-click on ***Oracle VM VirtualBox.exe*** to run the program and the system will display the main menu 👉. 💬 ***Tip:*** *ESP8266's virtual machine takes up much space (RAM). Please make sure the machine has enough memory to spare.*	 Oracle VM VirtualBox Manager File Machine Help New Settings Discard Start Details Snapshots **Welcome to VirtualBox!** The left part of this window is a list of all virtual machines on your computer. The list is empty now because you haven't created any virtual machines yet. In order to create a new virtual machine, press the **New** button in the main tool bar located at the top of the window. You can press the **F1** key to get instant help, or visit www.virtualbox.org for the latest information and news.
2. Set the default machine folder.	

Steps	Results
• Create a new folder, for example, ***D:\VM.*** • Select ***File > Preferences***, and the system will show the dialog box 👉. • In the ***General*** tab, set the location for the virtual machine in ***Default Machine Folder***, for example, ***D:\VM***. 💬 ***Tip:*** *ESP8266's virtual machine takes up much space (RAM). Please make sure the machine has enough memory to spare.*	 VirtualBox - Settings General Default Machine Folder: D:\VM VRDP Authentication Library: VBoxAuth General Input Update Language Display Network Extensions Proxy OK Cancel Help
3. Import the image file.	
• Select ***File > Import Appliance...***, and a dialog box will show up 👉. • Select the image file, for example, ***C:\ESP8266_lubuntu_20141021.ova***, and click on ***Next***. • Click ***Import*** to confirm the settings.	 Import Virtual Appliance Appliance to import VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below. C:\ESP8266_lubuntu_20141021.ova
4. Create a shared folder.	
• Create a new folder named ***D:\VM\share***. • Select ***Machine > Settings > Shared Folders...***, and a dialog box will show up 👉. • Select the shared folder in ***Machine Folders***, for example, ***D:\VM\share***.	 ESP8266_lubuntu_1 - Settings Shared Folders Folders List Name / Path / Auto-mount / Access Machine Folders share / D:\VM\share / Yes / Full General System Display Storage Audio Network Serial Ports USB Shared Folders OK Cancel Help

5. Run the virtual machine.

- After importing, a virtual machine named ***ESP8266_lubuntu*** shows up 👉.
- Double-click on ***ESP8266_lubuntu*** or ***Start*** to power on the virtual machine.



General
Name: ESP8266_lubuntu
Operating System: Ubuntu (32-bit)

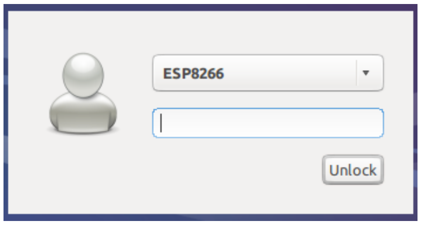
System
Base Memory: 1024 MB
Boot Order: Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

Display
Video Memory: 12 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

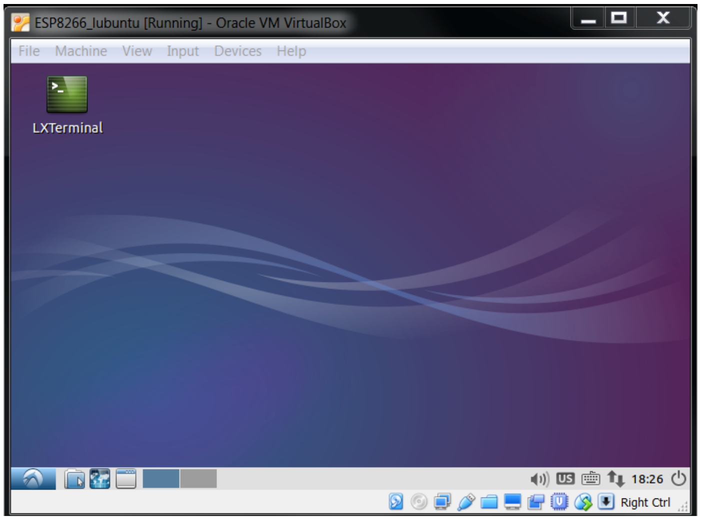
Storage
Controller: IDE
IDE Secondary Master: [Optical Drive] Empty
Controller: SATA
SATA Port 0: ESP8266_lubuntu-disk1.vmdk (Normal, 8.00 GB)

Audio

- The system shows the ESP8266 virtual machine 👉.
- If the virtual machine enters idle mode and is locked, a dialog box like the one below 👇 will show up. This is where the users enter the password: ***espressif***.

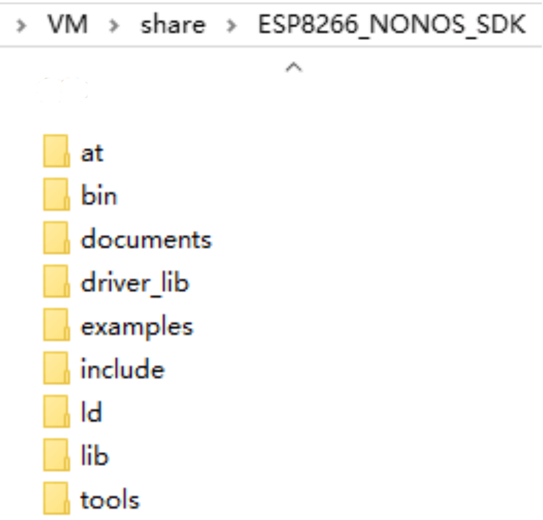


- Double-click on ***LXTerminal*** to start compiling applications. For more details please refer to ***Chapter 2.3***.

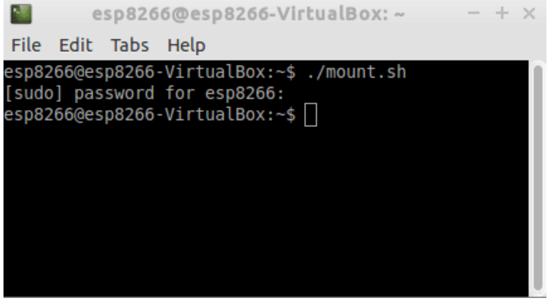
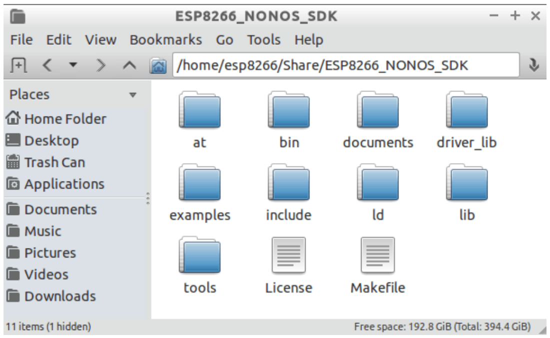

ESP8266_lubuntu [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
LXTerminal
US
18:26
Right Ctrl

2.3. Compiling Process

1. Start the virtual machine. Run ***LXTerminal*** on the desktop.
2. Copy the ***ESP8266_NONOS_SDK*** to the shared folder.

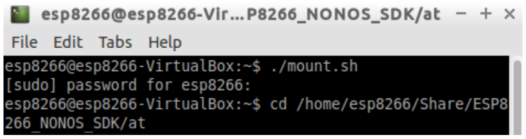
Steps	Result
- Copy the ***ESP8266_NONOS_SDK*** folder to the shared directory, for example, ***D:\VM\share***. - Copy the ***ESP8266_NONOS_SDK\examples\at*** folder to the directory ***D:\VM\share\ESP8266_NONOS_SDK***, as shown in the figure on the right 👉.	 at bin documents driver_lib examples include ld lib tools

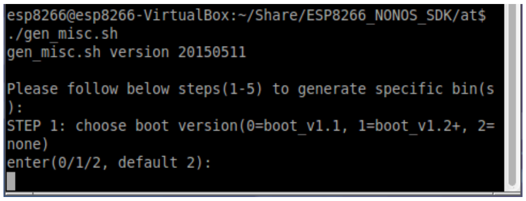
3. Mount the shared directory to the virtual machine.

Steps	Result
- Execute command `./mount.sh` in the LXTerminal. - Input the password ***espressif***. Shared folder mounting is completed.	 esp8266@esp8266-VirtualBox:~$ ./mount.sh [sudo] password for esp8266: esp8266@esp8266-VirtualBox:~$
- Open the shared directory ***ESP8266_NONOS_SDK*** in the virtual machine and check if the mounting has been successful. - If successful, the directory contains files as the figure on the right 👉 shows. - If not, the directory will be empty and users will need to repeat the previous step.	 ESP8266_NONOS_SDK File Edit View Bookmarks Go Tools Help Places Home Folder Desktop Trash Can Applications Documents Music Pictures Videos Downloads /home/esp8266/Share/ESP8266_NONOS_SDK at bin documents driver_lib examples include ld lib tools License Makefile 11 items (1 hidden) Free space: 192.8 GiB (Total: 394.4 GiB)

4. Change the directory to ***/Share/ESP8266_NONOS_SDK/at*** in the LXTerminal.

Steps	Result

• In the LXTerminal, execute the command:
`cd /home/esp8266/Share/ESP8266_NONOS_SDK/at`
	
	 esp8266@esp8266-Vir...P8266_NONOS_SDK/at
	File Edit Tabs Help
	`esp8266@esp8266-VirtualBox:~$ ./mount.sh`
	`[sudo] password for esp8266:`
	`esp8266@esp8266-VirtualBox:~$ cd /home/esp8266/Share/ESP8`
	`266_NONOS_SDK/at`

• Then execute command:
`./gen_misc.sh`
to start compiling.
• For example, the options for STEP 1 ~ 5 can be: 1, 1, 2, 0, 5.
	
	`esp8266@esp8266-VirtualBox:~/Share/ESP8266_NONOS_SDK/at$`
	`./gen_misc.sh`
	`gen_misc.sh version 20150511`
	
	`Please follow below steps(1-5) to generate specific bin(s`
	`):`
	`STEP 1: choose boot version(0=boot_v1.1, 1=boot_v1.2+, 2=`
	`none)`
	`enter(0/1/2, default 2):`

📖 ***Note:***

For more details on compiling, please refer to ESP8266 SDK Getting Started Guide.

5. After compilation, the binaries generated and their corresponding download addresses in the flash memory will be shown as follows:

```
Support boot_v1.4 and +

Generate user1.2048.new.5.bin successfully in folder bin/upgrade.

boot.bin------------>0x00000

user1.2048.new.5.bin--->0x01000

!!!
```

📖 ***Note:***

*Users can open the **/home/esp8266/Share/ESP8266_NONOS_SDK/bin/upgrade** directory and check the binaries compiled.*

6. The AT binaries generated can be downloaded to the ESP-LAUNCHER (refer to ***Section 1.3***) and run.

3. Compiling ESP8266_RTOS_SDK

3.1. Compiling *Process*

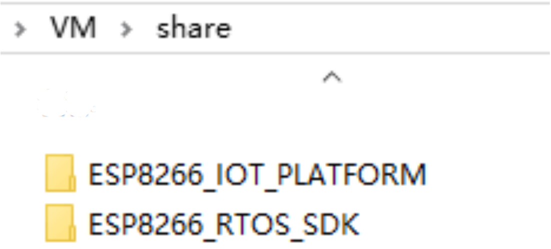
1. Download ***ESP8266_RTOS_SDK*** from:

   https://github.com/espressif/ESP8266_RTOS_SDK

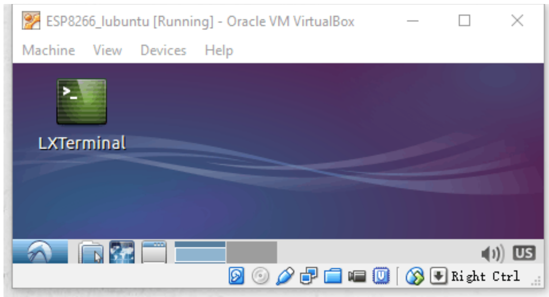
   ESP8266_IOT_PLATFORM is a demo application based on ***ESP8266_RTOS_SDK***.

   Download from: https://github.com/espressif/ESP8266_IOT_PLATFORM

2. Copy ***ESP8266_IOT_PLATFORM*** and ***ESP8266_RTOS_SDK*** to the shared folder.

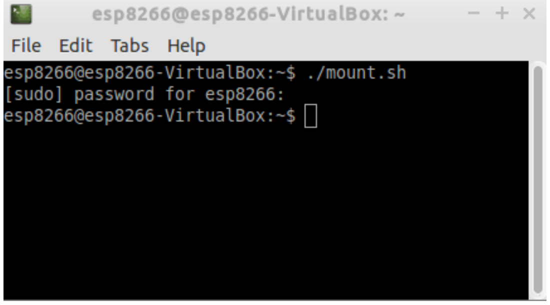
Steps	Result
Copy ***ESP8266_RTOS_SDK*** and ***ESP8266_IOT_PLATFORM*** to the shared folder, for example, ***D:\VM\share*** as shown on the right👉.	 › VM › share ESP8266_IOT_PLATFORM ESP8266_RTOS_SDK

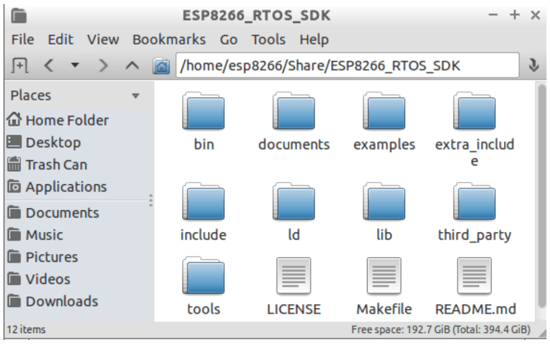
3. Start the virtual machine. Run ***LXTerminal***.

Steps	Result
• Start ESP8266 virtual machine 👉. - If the virtual machine is locked, please enter the password: ***espressif*** to unlock it. • Double-click on ***LXTerminal*** to start the compilation.	 ESP8266_lubuntu [Running] - Oracle VM VirtualBox Machine View Devices Help LXTerminal US Right Ctrl

4. Mount the shared directory to the virtual machine.

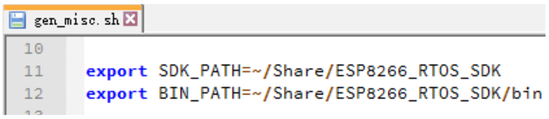
Steps	Result

• Execute command `./mount.sh` in the LXTerminal.
• Input the password: ***espressif***.
Shared folder mounting is completed.
	
	`esp8266@esp8266-VirtualBox:~$ ./mount.sh`
	`[sudo] password for esp8266:`
	`esp8266@esp8266-VirtualBox:~$`

• Open the shared directory ***ESP8266_RTOS_SDK*** in the virtual machine and confirm if the mounting has been successful. - If successful, the directory contains files as shown in the figure on the right 👉. - If not, the directory will be empty, and users will need to repeat the previous step.	 ESP8266_RTOS_SDK File Edit View Bookmarks Go Tools Help /home/esp8266/Share/ESP8266_RTOS_SDK Places Home Folder Desktop Trash Can Applications Documents Music Pictures Videos Downloads bin documents examples extra_include include ld lib third_party tools LICENSE Makefile README.md 12 items Free space: 192.7 GiB (Total: 394.4 GiB)

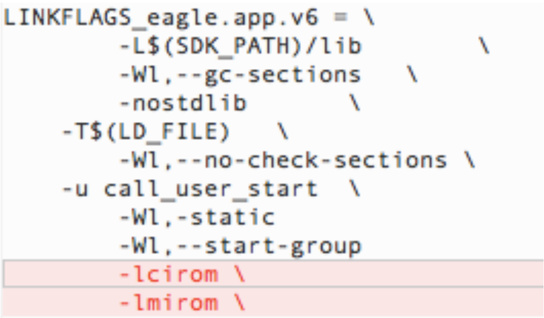
5. Modify the file ***ESP8266_IOT_PLATFORM/gen_misc.sh***. Set the variable PATH to point to the SDK and binaries.

Steps	Result

• ***SDK_PATH***: the path of ***ESP8266_RTOS_SDK***
• ***BIN_PATH***: the path where the generated binaries are stored after compilation
• Please see the figure on the right 👉.
	
	`export SDK_PATH=~/Share/ESP8266_RTOS_SDK`
	`export BIN_PATH=~/Share/ESP8266_RTOS_SDK/bin`

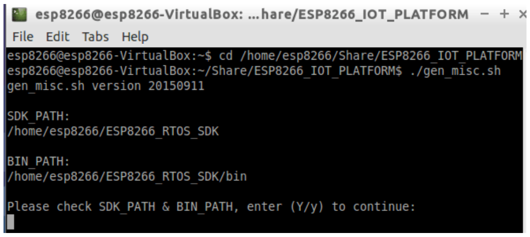
6. Modify ***ESP8266_IOT_PLATFORM/makefile***.

Steps	Result

Modify the `LINKFLAGS_eagle.app.v6` area in the ***ESP8266_IOT_PLATFORM/makefile*** by:
- Deleting `-lminic`
- Adding `-lcirom` and `-lmirom`
	
	`LINKFLAGS_eagle.app.v6 = \`
	`        -L$(SDK_PATH)/lib        \`
	`        -Wl,--gc-sections    \`
	`        -nostdlib        \`
	`    -T$(LD_FILE)    \`
	`        -Wl,--no-check-sections \`
	`    -u call_user_start    \`
	`        -Wl,-static`
	`        -Wl,--start-group`
	`        -lcirom \`
	`        -lmirom \`

7. Switch to ***/Share/ESP8266_IOT_PLATFORM*** in the ***LXTerminal***.

Steps	Result

• Execute the command:
`cd /home/esp8266/Share/ESP8266_IOT_PLATFORM`
• Then execute the command:
`./gen_misc.sh`
to start compiling.
	

```
esp8266@esp8266-VirtualBox:~$ cd /home/esp8266/Share/ESP8266_IOT_PLATFORM
esp8266@esp8266-VirtualBox:~/Share/ESP8266_IOT_PLATFORM$ ./gen_misc.sh
gen_misc.sh version 20150911

SDK_PATH:
/home/esp8266/ESP8266_RTOS_SDK

BIN_PATH:
/home/esp8266/ESP8266_RTOS_SDK/bin

Please check SDK_PATH & BIN_PATH, enter (Y/y) to continue:
```

📖 ***Note:***

For more details on compilation, please refer to ESP8266 SDK Getting Started Guide.

8. After compilation, the binaries generated and their download addresses in the flash memory will be shown as follows:

```
Support boot_v1.4 and +

Generate user1.1024.new.2.bin successfully in BIN_PATH

boot.bin------------>0x00000

user1.1024.new.2.bin--->0x01000

!!!
```

📖 ***Note:***

*Users can open the **/home/esp8266/Share/ESP8266_RTOS_SDK/bin** directory and check the binaries compiled.*

9. Download the generated binaries to the ESP-LAUNCHER and run.

📖 ***Note:***

The default baud rate of ESP8266 is 74880.

3.2. *ESP8266_RTOS_SDK* Architecture

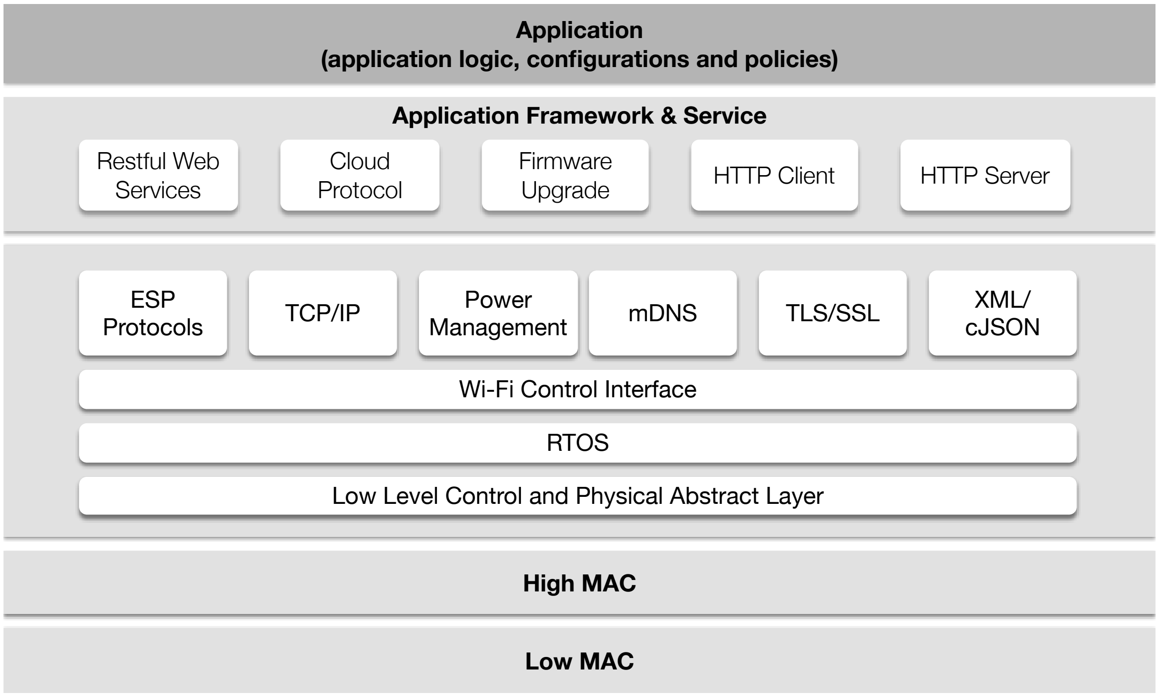


Figure 3-1. *ESP8266_RTOS_SDK* Architecture

4. Common Debug Methods

4.1. Common Debug Methods

4.1.1. Add UART Output Logs

For ***ESP8266_NONOS_SDK***, users can add debug logs as below:

```
os_printf("SDK version:%s\n", system_get_sdk_version());
```

For ***ESP8266_RTOS_SDK***, users can add debug logs as below:

```
printf("SDK version:%s\n", system_get_sdk_version());
```

4.1.2. Debug Fatal Exception

If a fatal exception occurs, the UART output logs will be shown as below:

```
Fatal exception (28):

epc1=0x4025bfa6, epc2=0x00000000, epc3=0x00000000, excvaddr=0x0000000f, depc=0x00000000
```

The debugging steps are as follows:

1. Find the ***.s*** file that corresponds to the firmware that is currently running.

   For example, if ***eagle.flash.bin*** and ***eagle.irom0text.bin*** are running, then the corresponding file is ***eagle .s***.

2. Locate the address of epc1 (in the form of 0x40XXXXXX) in the ***.s*** file to find the target function.

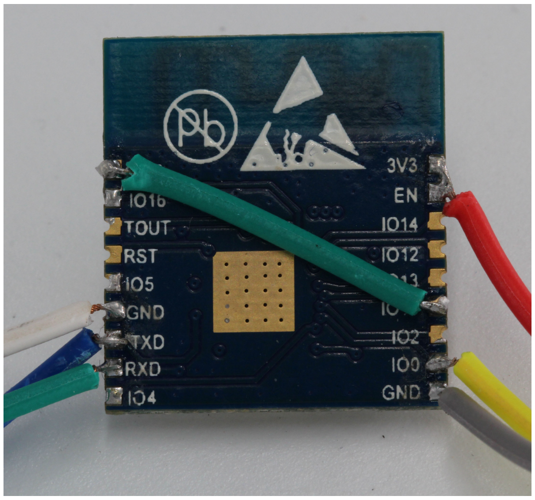
3. Add the UART logs before and after the target function to debug the fatal exception problem.

5. Downloading Firmware into the ESP-WROOM-02

Please follow the steps below to download firmware into ESP-WROOM-02.

1. ESP-WROOM-02 is the official ESP8266 module provided by Espressif Systems. Lead out the pins of ESP-WROOM-02, as shown in Table 5-1.

Table 5-1. ESP-WROOM-02 Pins

Pin	Pin status	Figure
EN	Pull up	 3V3 EN IO14 IO12 IO16 TOUT RST IO5 GND TXD RXD IO4 IO2 IO0 GND
3V3	3.3V power supply (VDD)	
IO15	Pull down	
IO0	UART Download mode: pull down; FLASH Boot mode: floating/pull up	
GND	GND	
RXD	UART Download Rx	
TXD	UART Download Tx, floating/pull up	

2. Connect ESP-WROOM-02 to the USB-to-TTL converter, using Dupont lines, as shown in Figure 5-1.

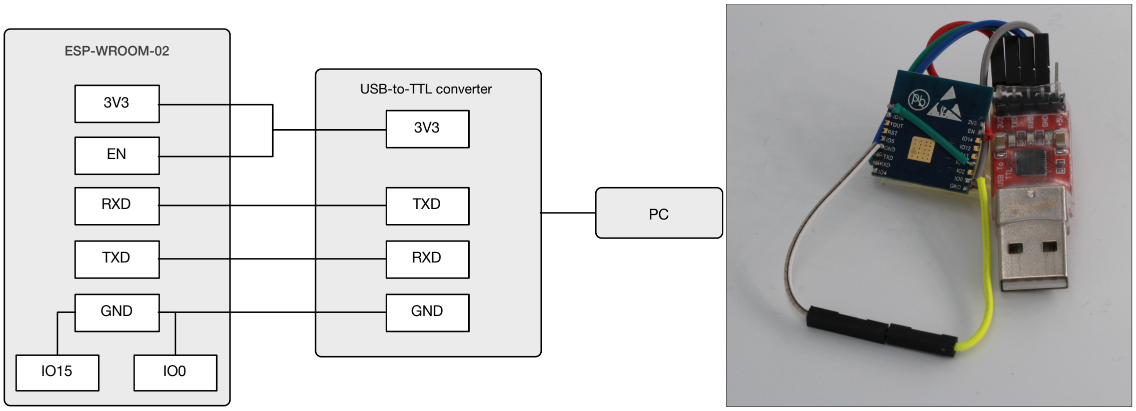


Figure 5-1. ESP-WROOM-02 Download Mode

3. Connect the USB-to-TTL converter to the PC.
4. Download firmware to flash with the ESP8266 Flash Download Tool.

> 📖 ***Note:***
>
> *On how to download firmware, please refer to* ***Section 1.3****.*

5. After downloading, set `IO0` as floating or pull up and switch ESP-WROOM-02 to working mode.
6. Power on ESP-LAUNCHER again and the chip will read and run programs from the flash.

> 📖 ***Note:***
>
> `IO0` *is an internal pull-up pin. For more information on the ESP-WROOM-02 hardware, please refer to ESP8266 System Description and ESP-WROOM-02 Datasheet.*

A. Appendix—Learning Resources

A.1. Must-Read Documents

- *ESP8266EX Datasheet*

  Description: This document introduces the specifications of ESP8266EX, including an overview of its features, protocols, technical parameters and applications. It also provides ESP8266EX's pin layout and the relevant description, as well as major functional modules and protocols applied on ESP8266EX (CPU, flash and memory, clock, radio, Wi-Fi, and low-power management). Furthermore, it provides a description of peripheral interfaces integrated on ESP8266EX, lists the electrical data of ESP8266EX and illustrates the package details for ESP8266EX.
- *ESP8266 Hardware Resources*

  Description: This zip package includes the manufacturing specifications of the ESP8266 board and the modules, manufacturing bill of materials (BOM) and schematics.
- *ESP8266 Non-OS SDK IoT_Demo Guide*

  Description: This document provides simple demo implementations of three types of smart devices: Smart Lights, Smart Power Plugs, and Sensor Devices. It also introduces the readers to curl toolkits, functions in LAN and WAN.
- *ESP8266 RTOS SDK Programming Guide*

  Description: This document provides sample codes based on ESP8266_RTOS_SDK, including basic examples, networking protocol examples and advanced examples.
- *ESP8266 AT Command Examples*

  Description: This document introduces some specific examples of using Espressif's AT commands, including setting up single connection using ESP8266 as a TCP client, setting up UDP transmission and transparent transmission, and setting up multiple connection using ESP8266 as a TCP server.
- *ESP8266 AT Instruction Set*

  Description: This document provides lists of AT commands based on ESP8266_NONOS_SDK, including user-defined AT commands, basic AT commands, Wi-Fi AT commands and TCP/IP-related AT commands. It also introduces the downloading of AT firmware into flash.
- *ESP8266 Non-OS SDK API Reference*

  Description: This document lists ESP8266_NONOS_SDK APIs, and provides an overview of the ESP8266_NONOS_SDK. It also introduces the readers to system APIs, TCP/UDP APIs, mesh APIs, application-specific APIs, definitions and data structures, as well as APIs for peripheral interfacing.

- *ESP8266 RTOS SDK API Reference*

  Description: This document lists ESP8266_RTOS_SDK APIs, including functions for Wi-Fi-related APIs, and boot APIs, etc.
- *FAQ*

A.2. Must-Have Resources

- *ESP8266 SDKs*

  Description: This webpage provides links to the latest version of the ESP8266 SDK as well as the older ones.
- *RTOS Sample Code*

  Description: This webpage provides sample code for the commonly-used functions.
- *Non-OS Sample Code*

  Description: This webpage provides sample code for the commonly used functions.
- *ESP8266 Tools*

  Description: This webpage provides links to the ESP8266 flash download tools and ESP8266 performance evaluation tools.
- *ESP8266 APK*
- *ESP8266 Certification and Test Guide*
- *ESP8266 BBS*
- *ESP8266 Resources*

Espressif IOT Team

www.espressif.com

Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice.

THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to use of information in this document is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG.

All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.