ESP8266 SDK

Getting Started Guide

Version 1.6

About This Guide

This document gives instructions for building ESP8266 SDK compiling environment and compiling procedures. Programming references such as binaries and flash map are also provided.

Notes :

- Go to Tutorial for Beginners for instructions on how to use the software development kit.
- Go to Espressif BBS for more product information.

The document is structured as follows.

Chapter	Title	Subject
Chapter 1	Set up Compiling Environment	Provides instructions on how to set up a Linux environment for SDK compilation.
Chapter 2	Compile SDK	Provides introductions for SDK software package and SDK compilation procedures.
Chapter 3	Download Binaries to Flash	Provides instructions for downloading SDK files to flash according to the flash size.
Chapter 4	Flash Map	Provides file allocation information in flash of different sizes.
Appendix	Appendix	Provides configuration information.

Release Notes

Date	Version	Release notes
2015.12	V1.5	First Release.
2016.02	V1.6	Updated Chapter 3.

Table of Contents

1. Set up Compiling Environment

All development tools for ESP8266 Internet of Things (IoT) module have been installed on a virtual machine. Users need to install the virtual machine to set up an Linux compiling environment for SDK compilation.

1.1. Download Virtual Machine Software and Image

VirtualBox is used for Linux virtualization. We recommend version 4.3.12.

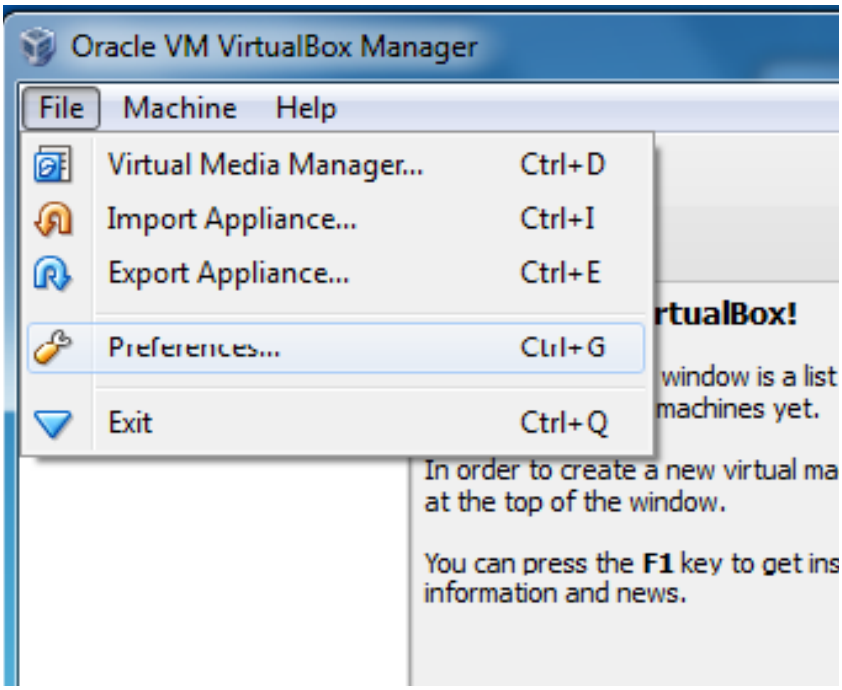
Espressif provides VirtualBox OVA image which can be imported into other virtual machine software.

Password: **qudl**

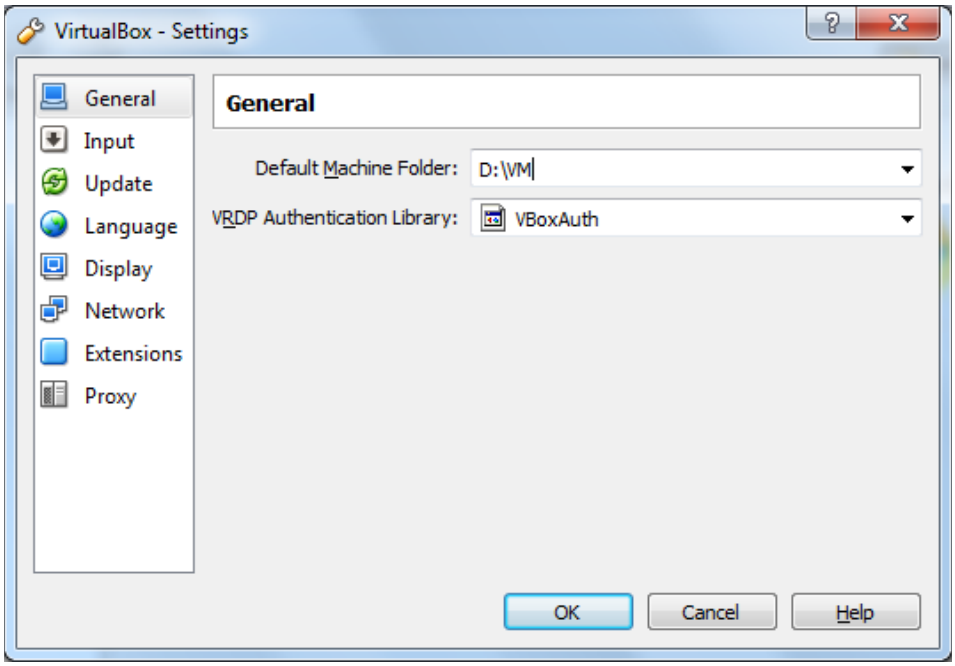
1.2. Install and Import Virtual Computer

Double-click **Virtualbox-4.3.12-xxx.exe** to install VirtualBox. By default, VirtualBox imports virtual computer into system disk and that takes a lot of system space. It is suggested to import virtual computer into a non-system disk. Follow the instructions to import and configure OVA image as below:

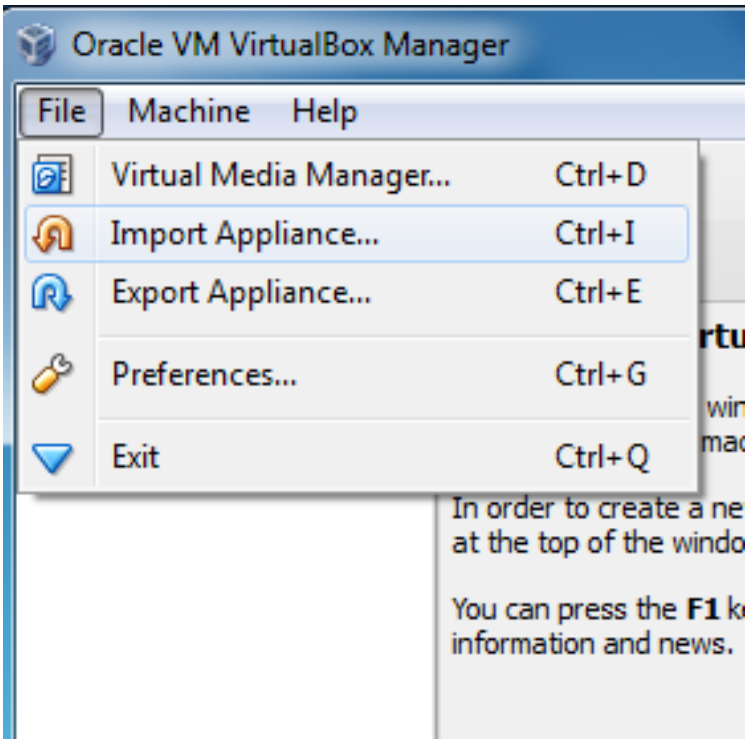
1. Open Virtualbox and select **Preference** from the **File** context menu.

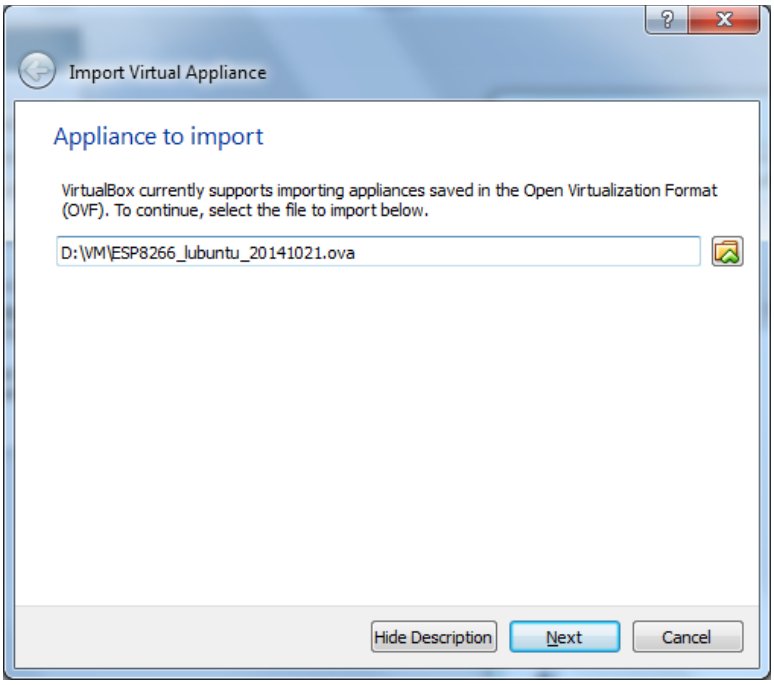


2. Select **General** and set the default machine folder where the virtual computer locates. For example: D:\VM.

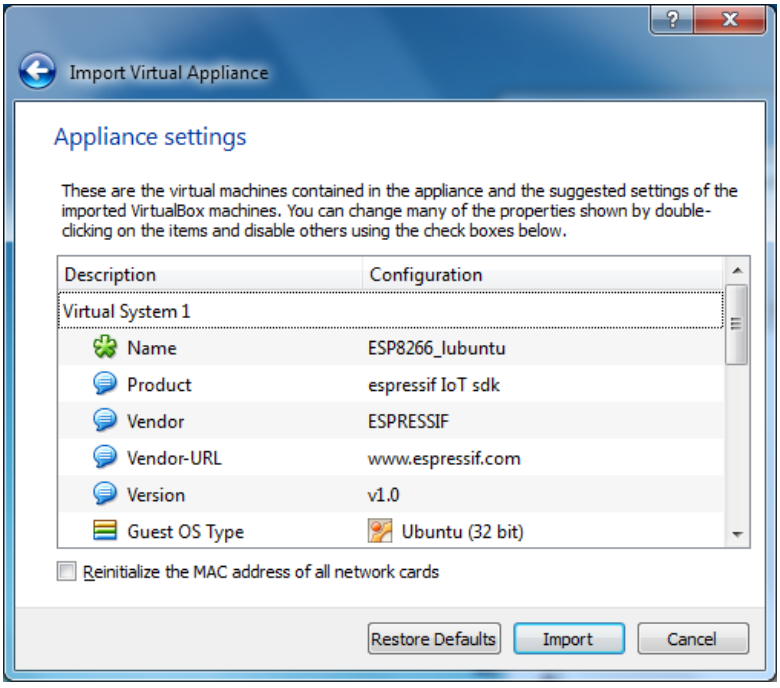


3. Select **Import Appliance** from File context menu. Set the path where the OVA image will be imported. For example: D:\vm\ESP8266_lubuntu_20141021.ova.

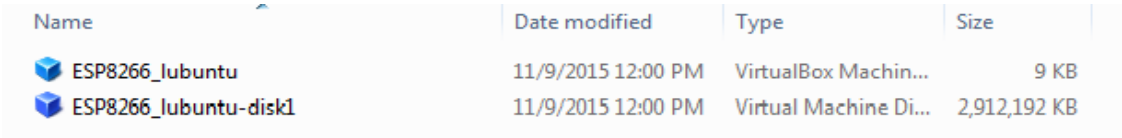




4. Click **Import** to confirm the settings.



5. The following files can be found in the directory where the virtual machine locates. For example, D:\VM\ESP_IOT_SDK.

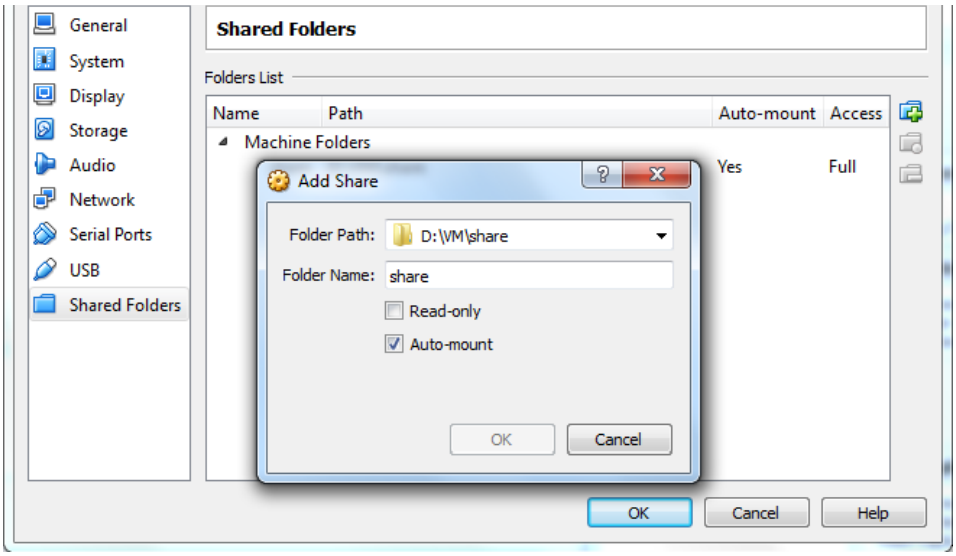


END

1.3. Create Share Folder

Before using the virtual machine, share with it the folders in the host machine.

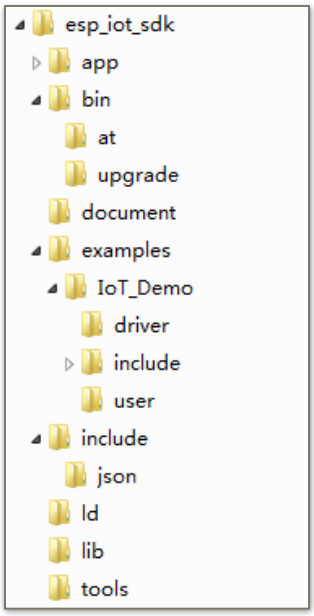
In VM VirtualBox Manager, click **Settings**, select **Machine Folders** in **Shared Folders**, then click on the right side to set up a new share folder.



2. Compile SDK

2.1. SDK Software Package

The SDK software package includes all header files, library files and compilation files needed for secondary development. See the picture below for directory structure:

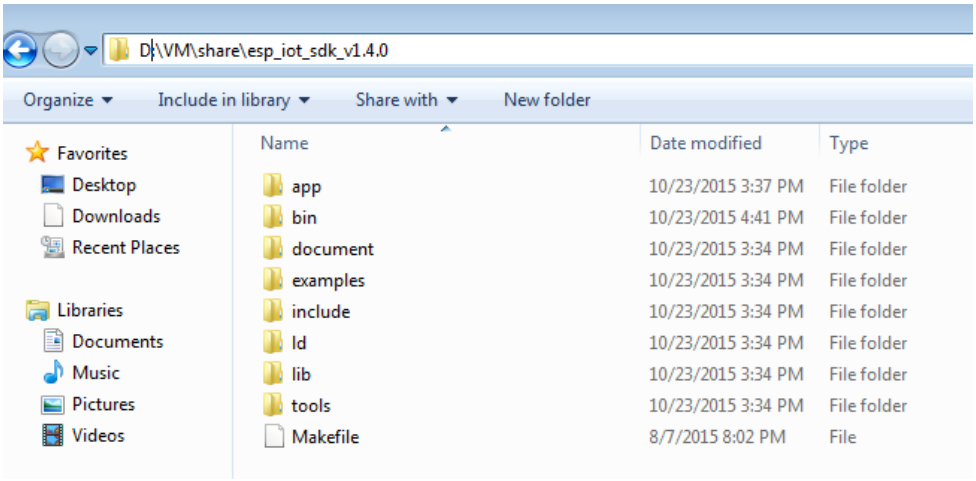


- **app** folder is the main working directory which contains source codes to be compiled.
- **bin** folder stores the bin files to be downloaded into the Flash:
  - **at** folder stores the bin files that support AT commands provided by Espressif Systems.
  - **upgrade** folder stores the compilation-generated bin files which support FOTA (**user1.bin** or **user2.bin**).
  - The root directory stores the compilation-generated bin files which do not support FOTA and other bin files provided by Espressif Systems.
- **examples** folder stores example codes (in IoT_Demo folder for example) which need to be duplicated to app folder if applicable.
- **include** folder stores the header files pre-installed in the SDK. The files contain relevant API functions and other macro definitions. No modifications are needed.
- **ld** folder stores the files needed for SDK software link. No modifications are needed.
- **lib** folder stores the library files needed for SDK compilation.
- **tools** folder stores the tools needed for generating bin files. No modifications are needed.

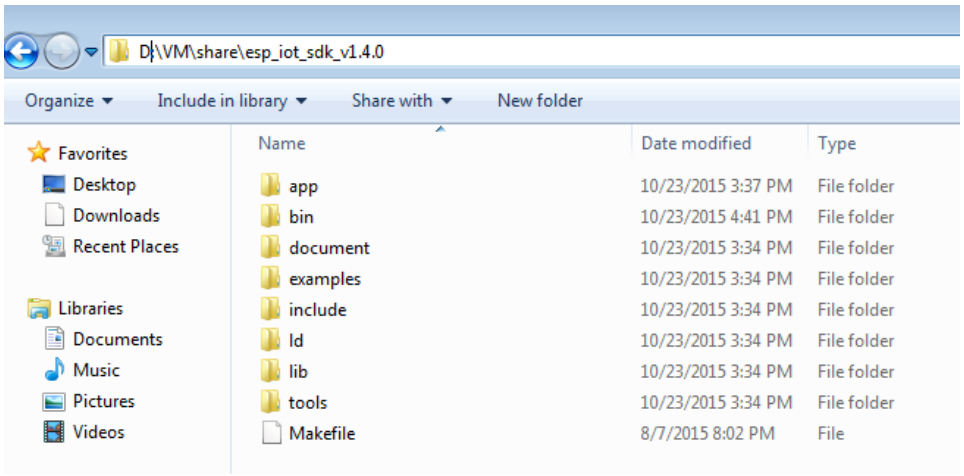
> Note :
>
> The latest SDK software package is provided on Espressif Download.

2.1. Prepare Compilation

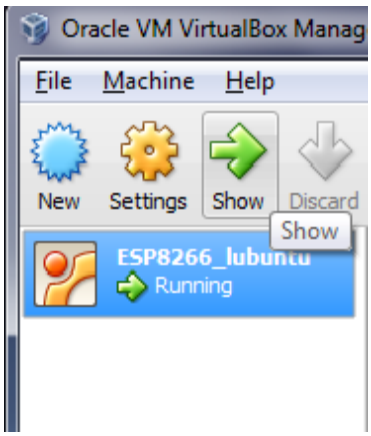
1. Copy **esp_iot_sdk f**older (including sub-folders and files) to **D:\VM\share**.



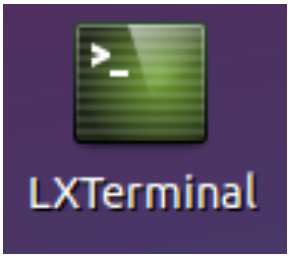
2. Copy all the files and sub-folders in **\esp_iot_sdk\examples\IoT_Demo** to **D:\VM\share \esp_iot_sdk\app**.



3. Run VirtualBox.

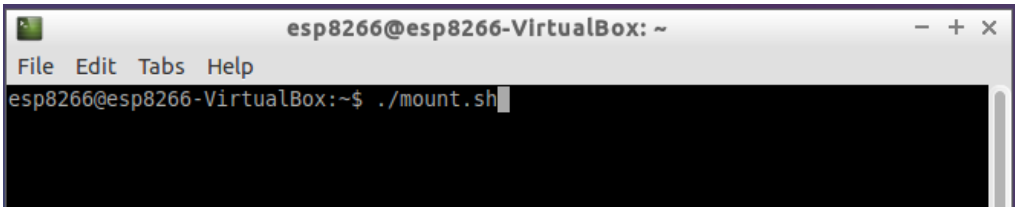


4. Run LXTerminal on the desktop of virtual machine.

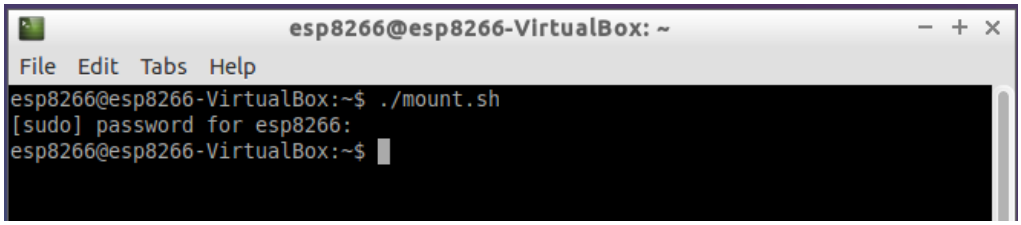


5. Input the following command and press Enter to mount the share folder.

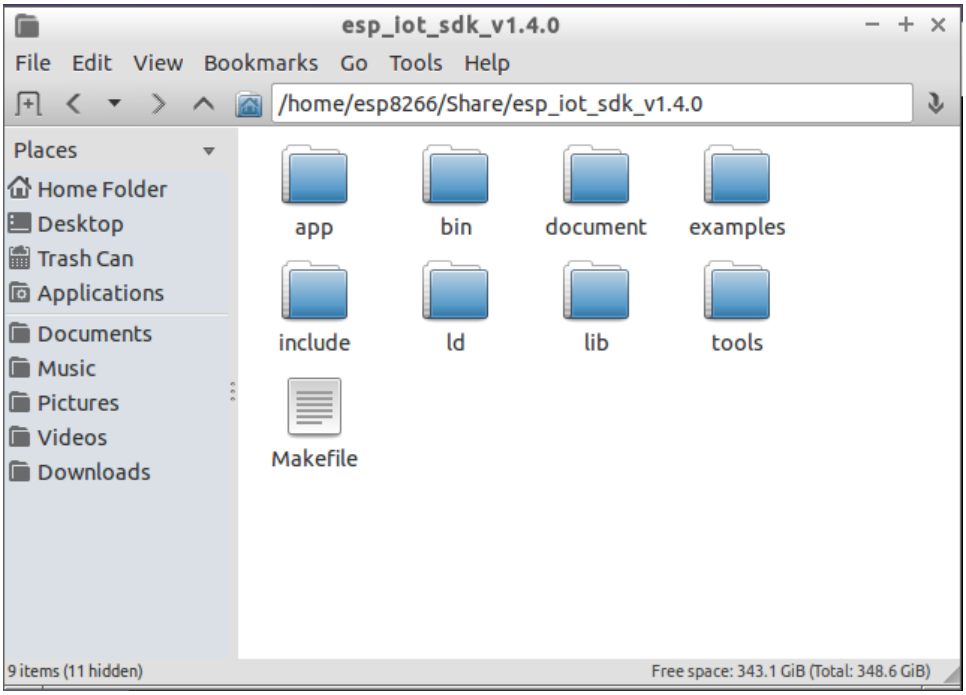
```
./mount.sh
```



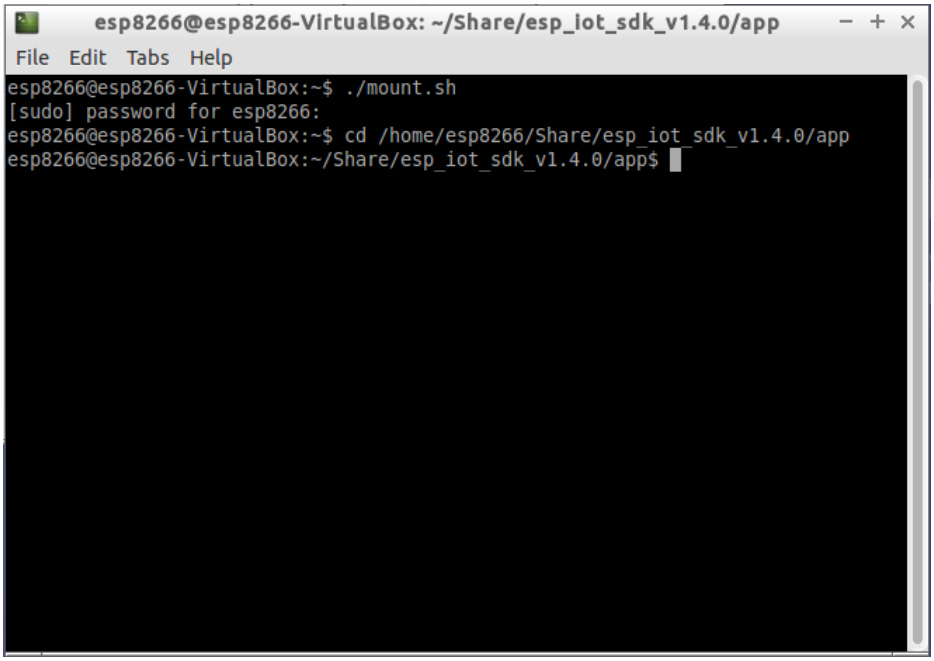
6. Input the password **espressif** and press Enter.



7. Check the shared folder in the VirtualBox.



8. Switch to **/share/esp_iot_sdk/app and** start compiling on the terminal end.



END

2.2. Compile SDK of v0.9.5 and Later Versions

For the SDK release of v0.9.5 and later, the compile process is simplified with a script in the APP folder. Execute the following command:

```
./gen_misc.sh
```

```
esp8266@esp8266-VirtualBox:~/Share/esp_iot_sdk/app$ ./gen_misc.sh
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):
```

STEP 1: Select boot version	
0	boot_v1.1, old version boot, supports FOTA (firmware upgraded through Wi-Fi).
1	boot_v1.2+, new version boot, always recommend to use the latest boot.bin, supports FOTA.
2	none boot, generate **eagle.flash.bin** and **eagle.irom0text.bin**, does not support FOTA.
STEP 2: Select bin to be generated	
0	input 2 in STEP 1, generate **eagle.flash.bin** and **eagle.irom0text.bin**, does not support FOTA.
1	input 0 or 1 in STEP 1, generate **user1.bin**, supports FOTA.
2	input 0 or 1 in STEP 1, generate **user2.bin**, supports FOTA.

STEP 3: SPI flash configuration (SPI speed)	
0	SPI speed 20 MHz, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
1	SPI speed 26.7 MHz, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
2	SPI speed 40 MHz, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
3	SPI speed 80 MHz, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
STEP 4: SPI flash configuration (SPI mode)	
0	QIO, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
1	QOUT, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
2	DIO, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
3	DOUT, according to your actual SPI flash, set same configuration while downloading by ESP flash download tool.
STEP 5: SPI flash configuration (SPI flash size & map)	
0	flash size 512 KB, flash map of program area is 256 KB + 256 KB.
2	flash size 1024 KB, flash map of program area is 512 KB + 512 KB when selecting 0 or 1 at STEP 1.
3	flash size 2048 KB, only the first 1024 KB is program area, flash map of program area is 512KB + 512KB when selecting 0 or 1 at STEP1.
4	flash size 4096 KB, only the first 1024 KB is program area, flash map of program area is 512 KB + 512 KB when selecting 0 or 1 at STEP 1.
5	flash size 2048 KB, flash map of program area is 1024 KB + 1024 KB when selecting 0 or 1 at STEP 1. Only supported by sdk_v1.1.0 + boot 1.4 + flash download tool_v1.2 and later version.
6	flash size 4096 KB, only the first 2048 KB is program area, flash map of program area is 1024 KB + 1024 KB when selecting 0 or 1 at STEP 1. Only supported by sdk_v1.1.0 + boot 1.4 + flash download tool_v1.2 and later version.

Notes :

- After compiling user1.bin, execute `make clean` firstly to clean up the temporary files generated by last compilation, then compile **user2.bin**.
- For detailed flash map, refer to Chapter 4.

When compilation succeeds, it shows the address for the bins to be written to. For example:

```
eagle.app.v6.flash.bin---------->addr:0x00000
eagle.app.v6.irom0text.bin------>addr:0x40000
!!!
esp8266@esp8266-VirtualBox:~/Share/esp_iot_sdk/app$
```

or

```
Generate user1.512.old.bin successully in folder bin/upgrade.
Support boot_v1.1 and +
user1.512.old.bin------>addr:0x1000
!!!
esp8266@esp8266-VirtualBox:~/Share/esp_iot_sdk/app$
```

2.3. Compile SDK of v0.9.4 and Previous Versions

For the SDK of v0.9.4 and previous versions, follow the steps below to enable cloud update (FOTA).

1. Execute the following command, **user1.bin** is generated in **/esp_iot_sdk/bin/upgrade.**

```
./gen_misc_plus.sh 1
```

2. Execute the following command to clean up the previous data.

```
make clean
```

3. Execute the following command, **user2.bin** is generated in **/esp_iot_sdk/bin/upgrade**.

```
./gen_misc_plus.sh 2
```

Notes :

- Refer to document “Firmware update through cloud server" for details about FOTA.
- esp_iot_sdk_v0.7 and previous versions do not support FOTA.
- esp_iot_sdk_v0.8 and later versions support cloud update and are compatible with previous compilation and burning methods.

3. Download Binaries to Flash

3.1. Tools

It is recommended to use minicom for serial port to debug and ESP8266 FLASH TOOL for downloading binaries to flash.

3.1.1. Minicom

When ESP8266 module is initialised, the default baud rate of serial port is 74880 bps with 1 stop bit every 8 data bits and no RTC/CTS. There is no 74880 bps option in minicom by default. Follow the following steps to configure the baud rate:

1. Install setserial using “yum” or “apt get” command. For example,

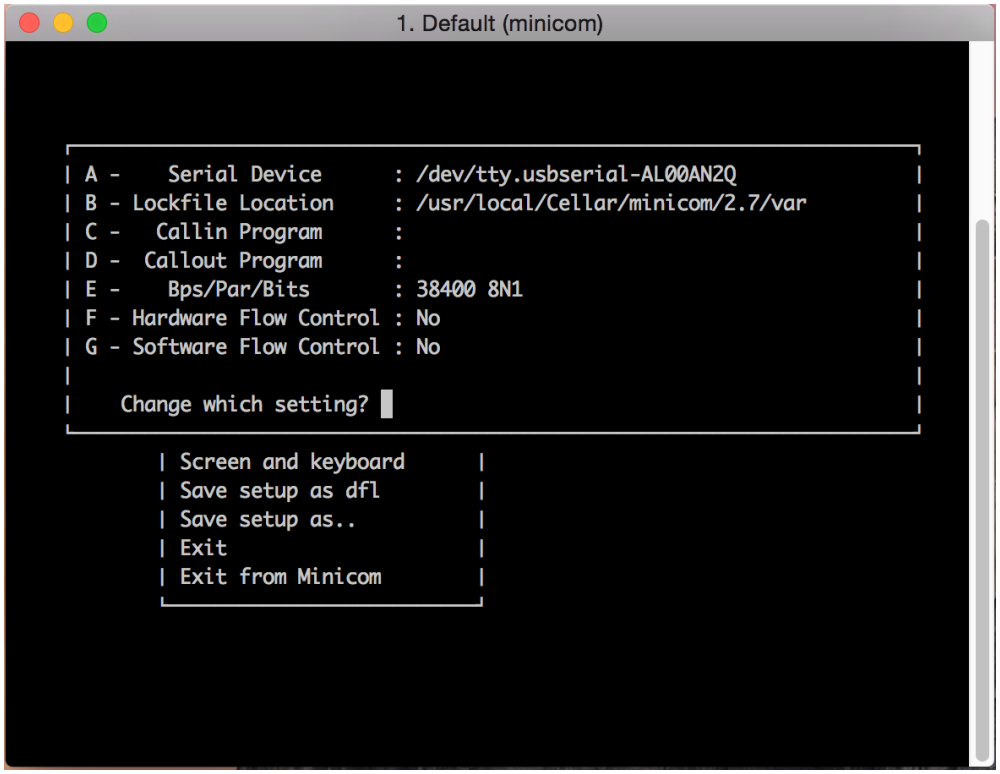
```
apt-get install setserial
```

2. Execute following command to set the division ratio. `ttyUSBx` refers to the serial port which connects to the ESP8266 board.

```
sudo setserial -v /dev/ttyUSBx spd_cust divisor $((24000000/74880))
```

3. Execute following command to configure the baud rate.

```
minicom-s
```



3.1.2. ESP8266 FLASH TOOL

Espressif provides ESP8266 FLASH TOOL to download multiple SDK binaries to the SPI flash of ESP8266 board by simple configuration.

Notes :

1. Download ESP8266 FLASH TOOL from Espressif Download.
2. For detailed information on the tool , refer to “ESP8266 FLASH TOOL User Manual”.

3.2. Binaries and Addresses

Once compilation succeeds, it shows the address for the binary to be written to the flash. The binaries and addresses vary with compilation mode and flash size. This chapter lists the required binaries and responding addresses for flashes of different capacities (512/1024/2048/4096 KB) in both no cloud update and supporting cloud update (FOTA) scenarios.

3.2.1. No Cloud Update

512 KB Flash

Table 3-1: 512 KB Flash

Bin	Address	Description
master_device_key.bin	0x3E000	Obtained from Espressif Cloud by users to get Espressif Cloud service.
esp_init_data_default.bin	0x7C000	Contains default RF parameters provided in SDK.
blank.bin	0x7E000	Contains default system parameters provided in SDK.
eagle.flash.bin	0x00000	Compiled from SDK.
eagle.irom0text.bin	0x40000	Compiled from SDK.

1024 KB Flash

Table 3-2: 1024 KB Flash

Bin	Address	Description
master_device_key.bin	0x3E000	Obtained from Espressif Cloud by users to get Espressif Cloud service.
esp_init_data_default.bin	0xFC000	Contains default RF parameters provided in SDK.
blank.bin	0xFE000	Contains default system parameters provided in SDK.
eagle.flash.bin	0x00000	Compiled from SDK.
eagle.irom0text.bin	0x40000	Compiled from SDK.

2048 KB Flash

Table 3-3: 2048 KB Flash

bin	Address	Description
master_device_key.bin	0x3E000	Obtained from Espressif Cloud by users to get Espressif Cloud service.
esp_init_data_default.bin	0x1FC000	Contains default RF parameters provided in SDK.
blank.bin	0x1FE000	Contains default system parameters provided in SDK.
eagle.flash.bin	0x00000	Compiled from SDK.
eagle.irom0text.bin	0x40000	Compiled from SDK.

4096 KB Flash

Table 3-4: 4096 KB Flash

bin	Address	Description
master_device_key.bin	0x3E000	Obtained from Espressif Cloud by users to get Espressif Cloud service.
esp_init_data_default.bin	0x3FC000	Contains default RF parameters provided in SDK.
blank.bin	0x3FE000	Contains default system parameters provided in SDK.
eagle.flash.bin	0x00000	Compiled from SDK.
eagle.irom0text.bin	0x40000	Compiled from SDK.

3.2.2. Support Cloud Update

Note :

User2.bin does not need to be burned into Flash, it can be downloaded through Wi-Fi (FOTA). The tables below will provide the address of user2.bin.

512 KB Flash

Table 3-5: 512 KB Flash

bin	Address	Description
master_device_key.bin	0x3E000	Obtained from Espressif Cloud by users to get Espressif Cloud service.
esp_init_data_default.bin	0x7C000	Contains default RF parameters provided in SDK.
blank.bin	0x7E000	Contains default system parameters provided in SDK.
boot.bin	0x00000	Boot loader provided in SDK. Latest version is recommended.
user1.bin	0x01000	Compiled from SDK.
user2.bin	0x41000	Compiled from SDK, does not need burned into Flash.

1024 KB Flash

Table 3-6: 1024 KB Flash

Bin	Address	Description
master_device_key.bin	0x3E000 (suggest to revise)	Obtained from Espressif Cloud by users to get Espressif Cloud service. Located in user parameter area. The default address in IOT_Demo is 0x3E000, which can be changed by the user. When 1 MB Flash is used, it is suggested to change the address to 0x7E000.
esp_init_data_default.bin	0xFC000	Contains default RF parameters provided in SDK.
blank.bin	0xFE000	Contains default system parameters provided in SDK.
boot.bin	0x00000	Boot loader provided in SDK. Latest version is recommended.
user1.bin	0x01000	Compiled from SDK.
user2.bin	0x81000	Compiled from SDK, does not need to be burned into Flash

2048 KB Flash

Table 3-7: 2048 KB Flash

bin	Address	Description
master_device_key.bin	0x3E000 (suggest to revise)	Obtained from Espressif Cloud by users to get Espressif Cloud service. Located in user parameter area. The default address in IOT_Demo is 0x3E000, which can be changed by the user. If selecting 3 in STEP 5 during the compilation, it is suggested to change the address to 0x7E000. If selecting 5 in STEP 5 during the compilation, it is suggested to change the address to 0xFE000. Refer to Table 3-9 for more details.
esp_init_data_default.bin	0x1FC000	Contains default RF parameters provided in SDK.
blank.bin	0x1FE000	Contains default system parameters provided in SDK.
boot.bin	0x00000	Boot loader provided in SDK. Latest version is recommended.
user1.bin	0x01000	Compiled from SDK.
user2.bin	0x81000	Compiled from SDK, does not need to be burned into Flash.

4096 KB Flash

Table 3-8: 4096 KB Flash

bin	Address	Description
master_device_key.bin	0x3E000 (suggest to revise)	Obtained from Espressif Cloud by users to get Espressif Cloud service. Located in user parameter area. The default address in IOT_Demo is 0x3E000, which can be changed by the user. If selecting 4 in STEP 5 during the compilation, it is suggested to change the address to 0x7E000. If selecting 6 in STEP 5 during the compilation, it is suggested to change the address to 0xFE000. Refer to Table 3-9 for more details.
esp_init_data_default.bin	0x3FC000	Contains default RF parameters provided in SDK.
blank.bin	0x3FE000	Contains default system parameters provided in SDK.
boot.bin	0x00000	Boot loader provided in SDK. Latest version is recommended.
user1.bin	0x01000	Compiled from SDK.
user2.bin	0x81000	Compiled from SDK, does not need to be burned into Flash.

Notes :

- System parameter area is the last four sectors of flash, 4 KB per sector, that is the last 16 KB of flash.
- User parameter area depends on user's definition. In IOT_Demo the four sectors starting from `0x3C000` are defined as the user parameter area.
- **master_device_key.bin** is needed if Espressif Cloud is used, and it is only necessary for the initial download. In IOT _Demo, it is located in the third sector of user parameter area.
- **blank.bin** should be downloaded to the second last sector in the flash as the initialisation parameter.
- **esp_init_data_default.bin** stores default RF values and which should be written to the forth sector from the end of flash.
- The flash size applied in IOT_Demo is 512 KB.

Table 3-9: Download Address of the Bin File

Bin	Download address					
	512KB	1024KB	2048KB		4096KB	
			512KB+512KB	1024KB +1024KB	512KB+512KB	1024KB +1024KB
esp_init_data_default.bin	0x7C000	0xFC000	0x1FC000	0x1FC000	0x3FC000	0x3FC000
blank.bin	0x7E000	0xFE000	0x1FE000	0x1FE000	0x3FE000	0x3FE000
boot.bin	0x00000	0x00000	0x00000	0x00000	0x00000	0x00000

Bin	Download address					
	512KB	1024KB	2048KB		4096KB	
			512KB+512KB	1024KB +1024KB	512KB+512KB	1024KB +1024KB
user1.bin	0x01000	0x01000	0x01000	0x01000	0x01000	0x01000
user2.bin	0x41000	0x81000	0x81000	0x101000	0x81000	0x101000
master_device_key.bin	0x3E000	0x7E000	0x7E000	0xFE000	0x7E000	0xFE000
Change user_light.h(or user_plug.h) #define PRIV_PARAM_START_SEC from **0x3C** to		0x7C	0x7C	0xFC	0x7C	0xFC
Change user_esp_platform.h #define ESP_PARAM_START_SEC from **0x3D** to		0x7D	0x7D	0xFD	0x7D	0xFD

4. Flash Map

Different settings of STEP 1 and STEP 5 in compilation (see Chapter 2.2) leads to different flash map.

Notes :

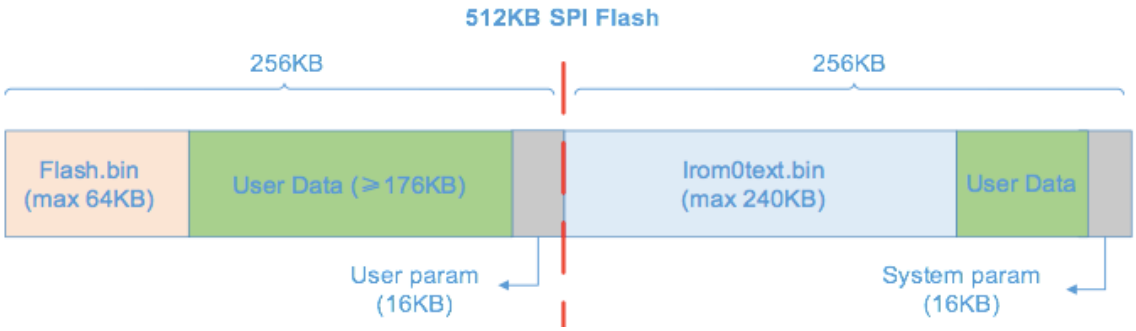
- System param (system parameter area) is always the last 16 KB of flash.
- User param is the user parameter area used by Espressif demo code (IOT_Demo or AT). If users develop their own application, user data can be saved in any flash area available.
- User Data area (green areas in figures below) represents available flash area. If program area does not reach to maximum size, the remaining area can be used to store user data.

4.1. None boot - No cloud Update

Select 2 none boot in STEP 1 of compilation to generate **eagle.flash.bin** (hereinafter referred to as **flash.bin**) and **eagle.irom0text.bin** (hereinafter referred to as **irom0text.bin**). Then select the corresponding flash map in STEP 5 according to the actual SPI flash used.

4.1.1. 512 KB Flash

Select 2 none boot in STEP 1, and then select 0 in STEP 5, the flash map will be as below.

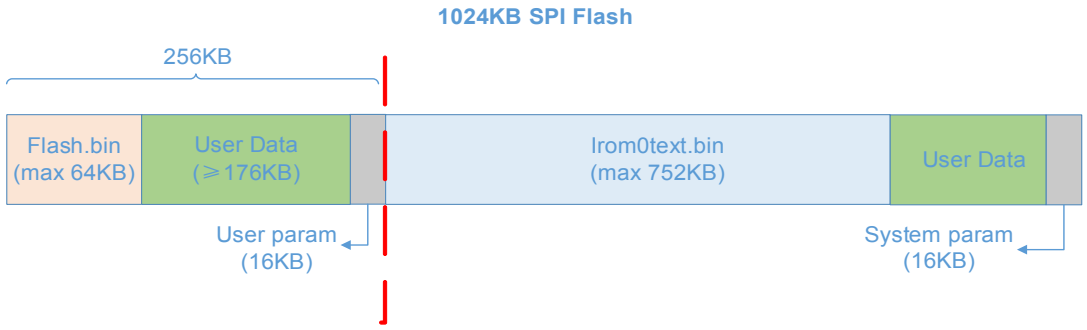


- User Data area: If program area (**flash.bin** and **irom0text.bin**) does not fill up the flash, the remaining area can be used to store user data.
- **irom0text.bin** is defined less than 200 KB by default. For 512 KB flash, users can revise ld file for compilation to realize the maximum size of 256 - 16 = 240 KB.
- In **eagle.app.v6.ld** (**\esp_iot_sdk\ld**), **len** of **irom0_0_seg** refers to the maximum size of **irom0text.bin**. For 512KB flash, it can be revised to 0x3C000 at most with the maximum size of 240 KB.

```
MEMORY
{
  dport0_0_seg :                  org = 0x3FF00000, len = 0x10
  dram0_0_seg :                   org = 0x3FFE8000, len = 0x14000
  iram1_0_seg :                   org = 0x40100000, len = 0x8000
  irom0_0_seg :                   org = 0x40240000, len = 0x32000
}
```

4.1.2. 1024 KB Flash

Select 2 none boot in STEP 1, and then select 2 in STEP 5, the flash map will be as below.

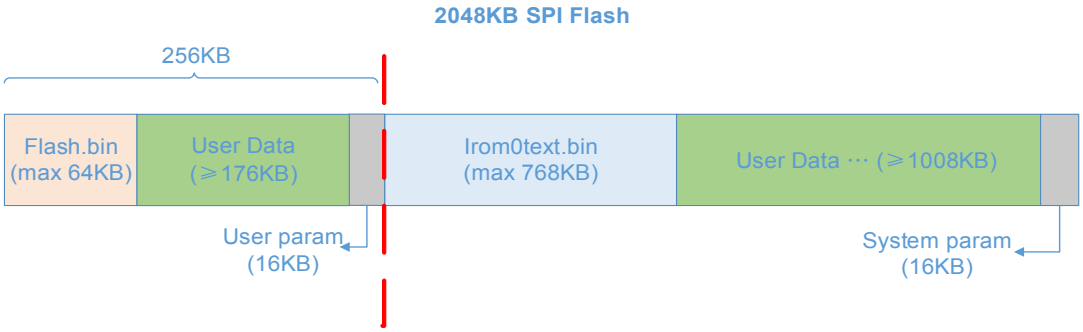


- User Data area: If program area (**flash.bin** and **irom0text.bin**) does not fill up the flash, the remaining area can be used to store user data.
- **irom0text.bin** is defined less than 200 KB by default. For 1024 KB flash, users can revise ld file for compilation to realize the maximum size of 1024 - 256 - 16 = 752 KB.
- In **eagle.app.v6.ld** (**\esp_iot_sdk\ld**), **len** of **irom0_0_seg** refers to the maximum size of **irom0text.bin**. For 1024KB flash, it can be revised to 0xBC000 at most with the maximum size of 752 KB.

```
MEMORY
{
  dport0_0_seg :                  org = 0x3FF00000, len = 0x10
  dram0_0_seg :                   org = 0x3FFE8000, len = 0x14000
  iram1_0_seg :                   org = 0x40100000, len = 0x8000
  irom0_0_seg :                   org = 0x40240000, len = 0x32000
}
```

4.1.3. 2048 KB Flash

Select 2 none boot in STEP 1, and then select 3 in STEP 5, the flash map will be as below.

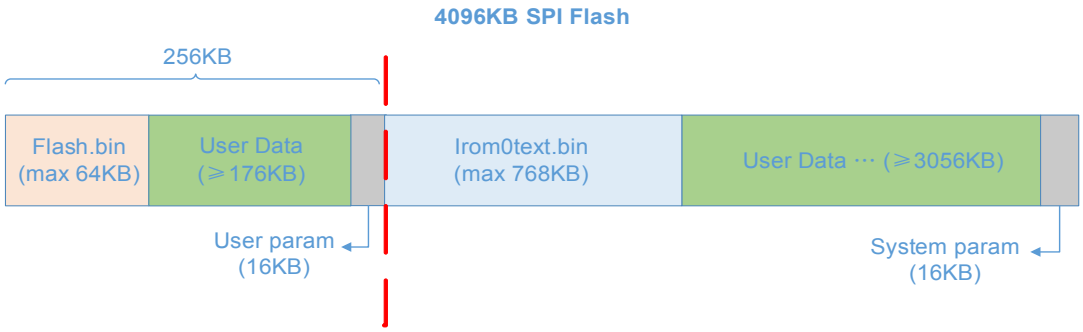


- User Data area: If program area (**flash.bin** and **irom0text.bin**) does not fill up the flash, the remaining area can be used to store user data.
- **irom0text.bin** is defined less than 200 KB by default. Only the first 1024 KB can be program area now, so for 2048 KB flash, user can revise ld file for compilation to realize the maximum size of 1024 - 256 = 768 KB.
- In **eagle.app.v6.ld** (**\esp_iot_sdk\ld**), **len** of **irom0_0_seg** refers to the maximum size of irom0text.bin. For 2048 KB flash, it can be revised to 0xC0000 at most with the maximum size of 768 KB.

```
MEMORY
{
  dport0_0_seg :                    org = 0x3FF00000,  len = 0x10
  dram0_0_seg :                     org = 0x3FFE8000,  len = 0x14000
  iram1_0_seg :                     org = 0x40100000,  len = 0x8000
  irom0_0_seg :                     org = 0x40240000,  len = 0x32000
}
```

4.1.4. 4096 KB Flash

Select 2 none boot in STEP 1, and then select 4 in STEP 5, the flash map will be as below.



- User Data area: If program area (**flash.bin** and **irom0text.bin**) does not fill up the flash, the remaining area can be used to store user data.

- **irom0text.bin** is defined less than 200 KB by default. Only the first 1024 KB can be program area now, so for 4096 KB flash, user can revise ld file for compilation to realize the maximum size of 1024 - 256 = 768 KB.
- In **eagle.app.v6.ld** (**\esp_iot_sdk\ld**), **len** of **irom0_0_seg** refers to the maximum size of irom0text.bin. For 2048 KB flash, it can be revised to 0xC0000 at most with the maximum size of 768 KB.

```
MEMORY
{
  dport0_0_seg :                    org = 0x3FF00000, len = 0x10
  dram0_0_seg :                     org = 0x3FFE8000, len = 0x14000
  iram1_0_seg :                     org = 0x40100000, len = 0x8000
  irom0_0_seg :                     org = 0x40240000, len = 0x32000
}
```

4.2. With boot - Support Cloud Update

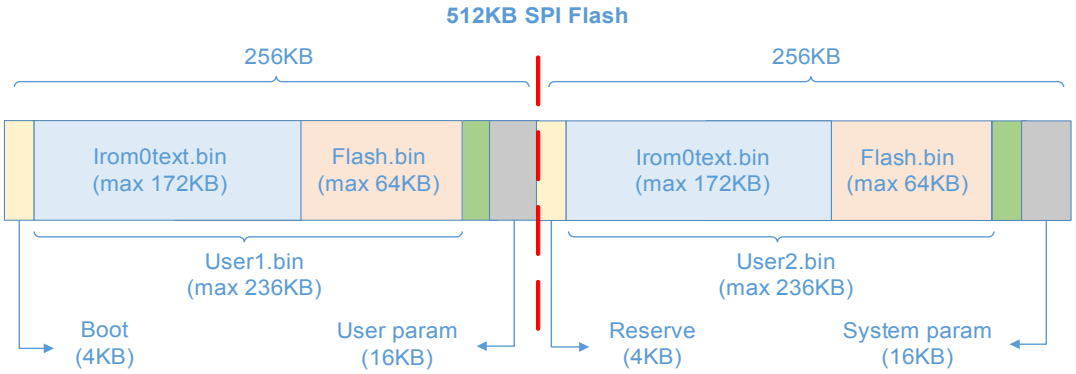
Select 1 boot_v1.2+ in STEP 1 to support FOTA to generate **user1.bin** and **user2.bin**. Then select the corresponding flash map in STEP 5 according to the actual SPI flash used.

Notes :

- After compiling **user1.bin**, execute `make clean` first to clean up the temporary files generated by last compilation, then compile **user2.bin**.
- boot_v1.1 is an old version boot, compilation and downloading are the same as boot_v1.2+, it is recommended to use the latest version of **boot.bin**.
- User Data area (green area in figures below) represents the flash area that may be available. If the program area does not reach the maximum size, the remaining area can be used to store user data.

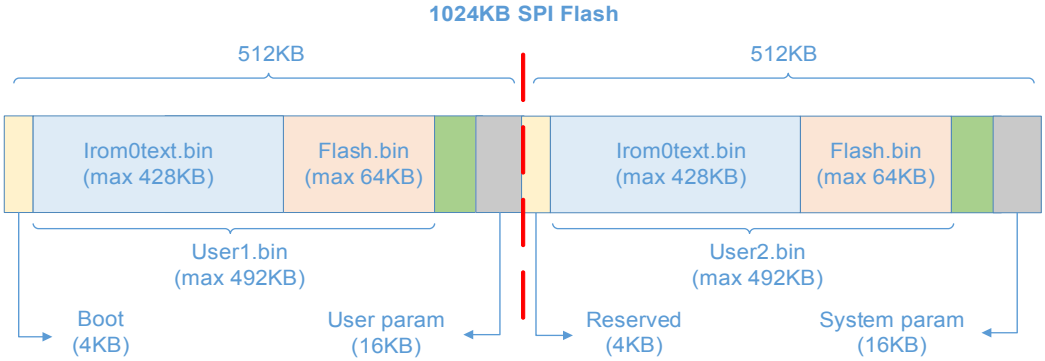
4.2.1. 512 KB Flash

Select 1 boot_v1.2+ in STEP 1, and then select 0 in STEP 5, flash map will be as below.



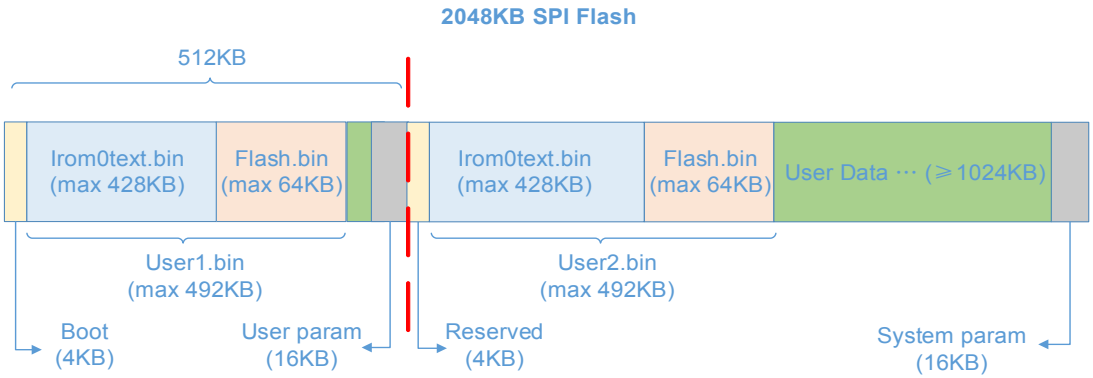
4.2.2. 1024KB Flash

Select 1 boot_v1.2+ in STEP 1, and then select 2 in STEP 5, flash map will be as below.

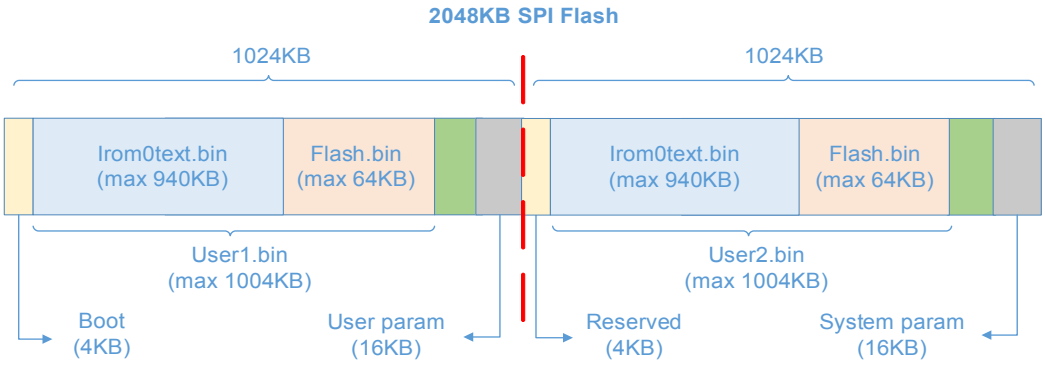


4.2.3. 2048 KB Flash

- Select 1 boot_v1.2+ in STEP 1, and select 3 in STEP 5, only the first 1024KB is the program area (512KB + 512KB), the flash map will be as below.

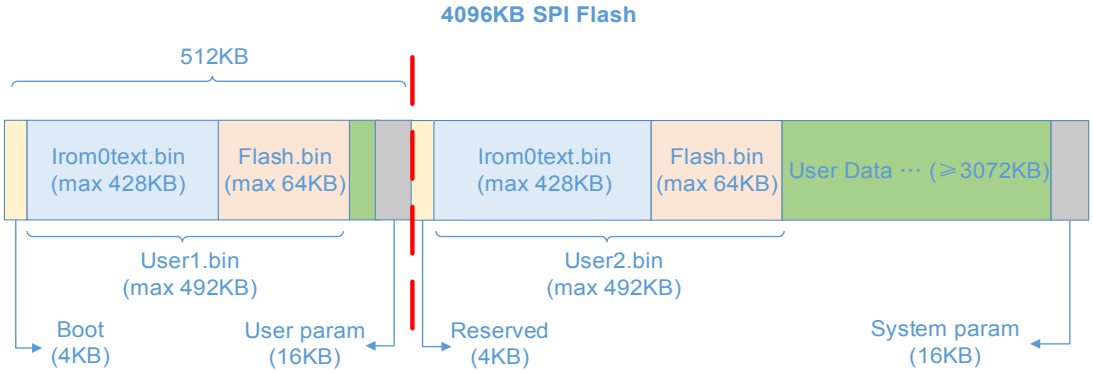


- Select 1 boot_v1.2+ in STEP 1, and then select 5 in STEP 5 (only supported by sdk_v1.1.0 + boot v1.4 + flash download tool v1.2 and later version), the program area is 1024KB + 1024KB, the flash map will be as below.

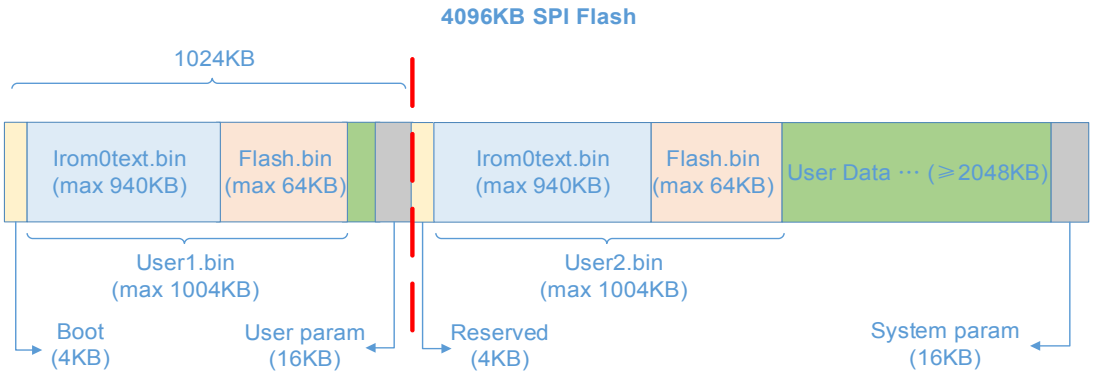


4.2.4. 4096 KB Flash

- Select 1 boot_v1.2+ in STEP 1, and then select 4 in STEP 5, only the first 1024KB is the program area (512KB + 512KB), the flash map will be as below.

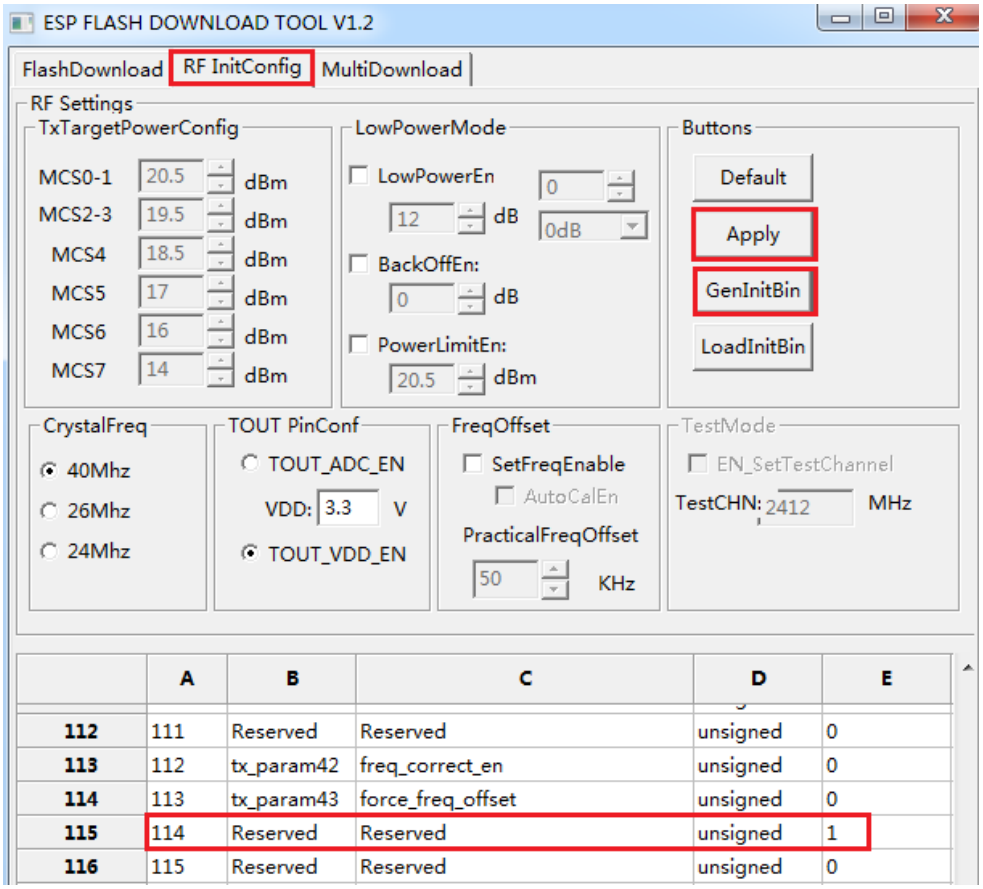


- Select 1 boot_v1.2+ in STEP 1, and then select 6 in STEP 5 (only supported by sdk_v1.1.0 + boot 1.4 + flash download tool v1.2 and later version), the first 2048KB is the program area (1024KB + 1024KB), the flash map will be as below.

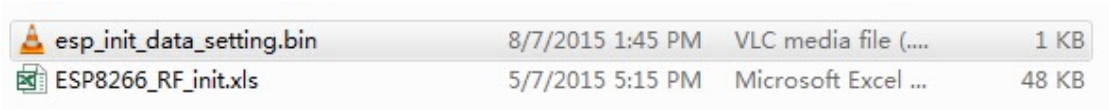


I. Appendix

By default, RF calibration will be performed every time the system is initialized and this will take some time. From esp_iot_sdk_v1.3.0 and later releases, the user can set RF calibration to perform only when the system is powered on at the first time. The calibration data will be saved in flash and no more calibrations are needed afterwards. Follow the steps of configuration below to reduce the system initialization time (including deep-sleep wake-up time):



1. Open ESP8266 FLASH TOOL and click **RF InitConfig** tab. The table at the bottom contains RF configuration parameters (0 ~ 127 bytes) in **esp_init_data_default.bin**. Click byte 114, and change the parameter to 1.
2. Click **Apply** button to confirm the configuration.
3. Click **GenInitBin** to generate **esp_init_data_setting.bin** which will be downloaded to Flash instead of **esp_init_data_default.bin**.



END

Espressif IOT Team

http://bbs.espressif.com