



# ESP8266 SDK 使用手册

**Version 1.0.1**

Espressif Systems IOT Team

Copyright (c) 2015



#### 免责声明和版权公告

本文中的信息，包括供参考的URL地址，如有变更，恕不另行通知。

文档“按现状”提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档不负任何责任，包括使用本文档内信息产生的侵犯任何专利权行为的责任。本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

Wi-Fi联盟成员标志归Wi-Fi联盟所有。

文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

版权归© 2014 乐鑫信息技术有限公司所有。保留所有权利。



# Table of Contents

1.	前言.....	4
2.	开发工具 .....	4
2.1.	串口工具 – SecureCRT.....	4
2.2.	烧录工具 - FLASH_DOWNLOAD_TOOLS.....	4
3.	SDK 软件包.....	6
3.1.	目录结构.....	6
2.	编译.....	7
2.1.	编译 esp_iot_sdk_v0.9.5 及之后版本软件.....	8
2.2.	编译 esp_iot_sdk_v0.9.4 及之前版本软件.....	8
3.	烧录说明 .....	9
3.1.	不支持云端升级 .....	9
1.	512KB Flash.....	9
2.	1MB 及以上容量 Flash .....	9
3.2.	支持云端升级(FOTA) .....	9
1.	512KB Flash.....	9
2.	1MB 及以上容量 Flash .....	10



## 1. 前言

本文主要介绍基于ESP8266物联网模块的SDK相关使用方法，包括开发工具使用以及SDK软件包架构等。

更多ESP8266的信息，请访问：<http://bbs.espressif.com/>

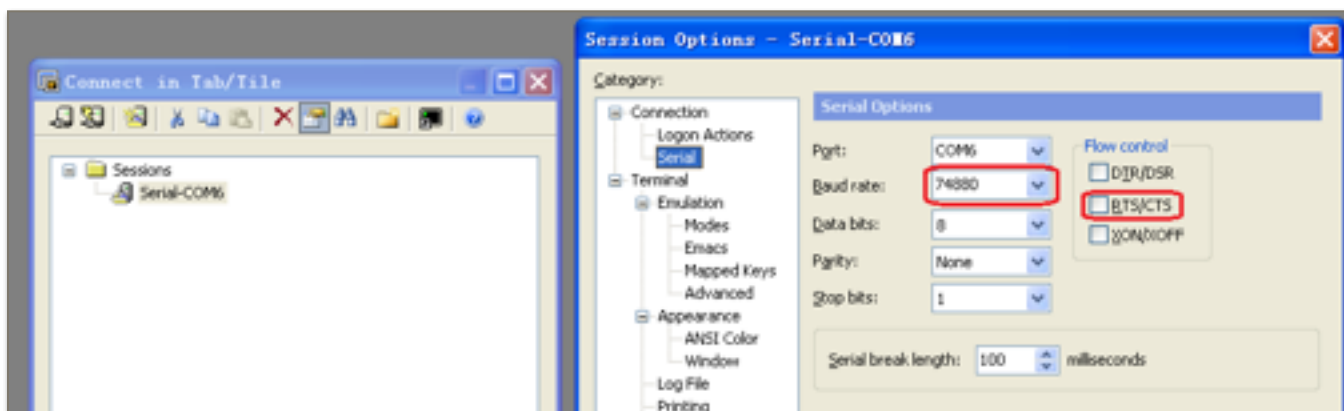
新手指南位于BBS <http://bbs.espressif.com/viewforum.php?f=21>

## 2. 开发工具

以下列出建议使用的串口工具和烧录工具，客户也可以选择使用其他同样功能的工具。串口工具，用于打印信息，进行调试；烧录工具，用于下载软件到 flash 中。

### 2.1. 串口工具 - SecureCRT

ESP8266模块采用74880波特率，需要在SecureCRT中进行设置。



### 2.2. 烧录工具 - FLASH\_DOWNLOAD\_TOOLS

Espressif 提供工具 “ESP\_FLASH\_DOWNLOAD” 实现多个 bin 文件的一键烧录，将编译生成的多个 \*.bin 文件一次性下载到 ESP8266 母板的 SPI Flash 中。

**ESP\_FLASH\_DOWNLOAD** 使用说明：

1. 烧录文件勾选区：选择要烧录的bin文件，以及设置对应的烧录地址；
2. SPI FLASH CONFIG 区：配置 SPI Flash 的属性，按键 “CombineBin” 将上述勾选了的 bin 文件合成一个 `target.bin`，按键 “Default” 将 SPI Flash 的配置恢复默认值。
3. Mac 地址: ESP8266 的 MAC 地址。

ESP8266 母板上跳线设置为 **MTDO: 0**，**GPIO0: 0**，**GPIO2: 1**，进入下载模式。  
操作步骤如下：

- 如下绿色显示区域，选择要烧录的 bin 文件 → 填写烧录地址 → 勾选待烧录的选项。



- 设置 COM 口和波特率。
- 点击 "START" 开始下载。
- 下载完成后，将母板断电，修改跳线为运行模式，上电正常运行。  
母板上跳线设置为 **MTDO: 0**, **GPIO0: 1**, **GPIO2: 1**, 可进入运行模式。

注意：进行跳线操作时，请断电操作。



File

<input checked="" type="checkbox"/>	D:\VM\share\esp_iot_sdk\bin\esp_init_data.bin	...	OFFSE"	0x7c000
<input checked="" type="checkbox"/>	D:\VM\share\esp_iot_sdk\bin\blank.bin	...	OFFSE"	0x7e000
<input checked="" type="checkbox"/>	D:\VM\share\esp_iot_sdk\bin\eagle.app.v5	...	OFFSE"	0x00000
<input checked="" type="checkbox"/>	D:\VM\share\esp_iot_sdk\bin\eagle.app.v6	...	OFFSE"	0x40000
<input type="checkbox"/>		...	OFFSE"	
<input type="checkbox"/>		...	OFFSE"	
<input type="checkbox"/>		...	OFFSE"	

SPI FLASH CONFIG

CrystalFreq: 26M

CombineBin: Default

SPI SPEED: ☒ 40MHz, ☐ 26.7MHz, ☐ 20MHz, ☐ 80MHz

SPI MODE: ☒ QIO, ☐ QOUT, ☐ DIO, ☐ DOUT

FLASH SIZE: ☒ 4Mbit, ☐ 2Mbit, ☐ 8Mbit, ☐ 16Mbit, ☐ 32Mbit

Mac Address

AP MAC: 1A-FE-34-97-05-7B  
STA MAC: 18-FE-34-97-05-7B

AP: 1A-FE-34-97-05-7B

ShowMac PRINT

COM: COM6

BAUDRATE: 115200

START STOP Download 下载中...

Pos: (181, 4) Current Pts: 5 Line Count: 6

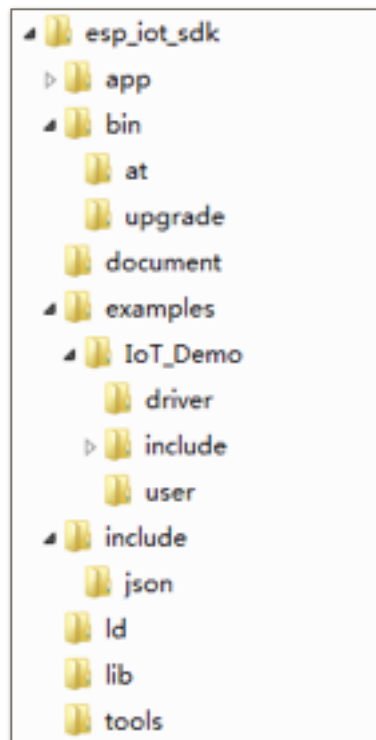


## 3. SDK 软件包

### 3.1. 目录结构

SDK软件包中包含了进行二次开发所需的头文件、库文件以及其他编译所需的文件。

目录结构如下图：



具体说明：

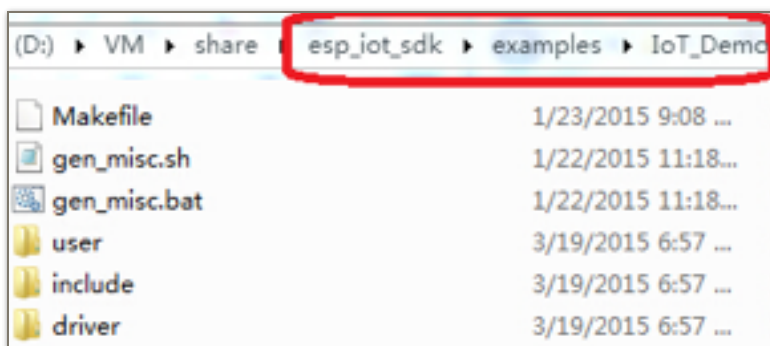
- "app" 目录为用户工作主目录，用户级代码及头文件均放在此目录下编译。
- "bin" 目录存放需下载到 Flash 的 bin 文件，其中：
  - ▶ "at" 文件夹 - Espressif 提供的支持 AT+ 指令的 bin 文件；
  - ▶ "upgrade" 文件夹 - 编译生成的支持云端升级的 bin 文件（user1.bin 或 user2.bin）；
  - ▶ "bin" 文件夹根目录 - 编译生成的不支持云端升级的 bin 文件，和其他 Espressif 提供的 bin 文件。
- "examples" 目录存放 SDK 的上层示例代码，使用时需将子目录（例如 IoT\_Demo 目录）下的所有内容到 "app" 目录下编译；
- "include" 目录为 SDK 自带头文件，包含了用户可使用的相关 API 函数及其他宏定义，用户不需修改；
- "ld" 目录为 SDK 软件编译链接时所需文件，用户不需修改；



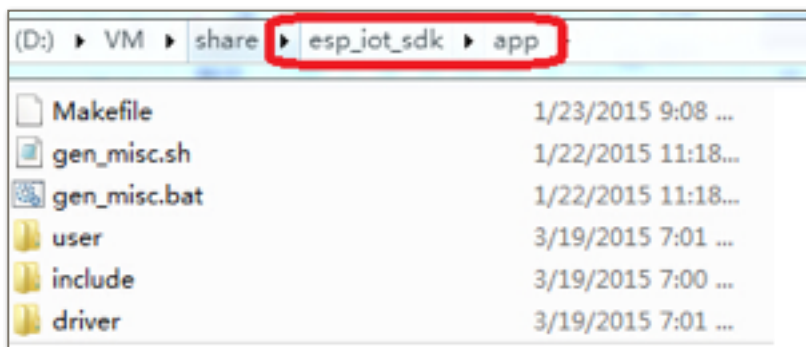
- "lib" 目录为 SDK 编译所需库文件；
- "tools" 目录为编译生成 bin 文件所需的工具，用户不需修改。

## 2. 编译

注意，将 `esp_iot_sdk\examples` 子目录内的文件拷贝到 `esp_iot_sdk\app` 目录下进行编译。  
例如，拷贝编译 IOT\_Demo：



拷贝上图路径所有文件到 `esp_iot_sdk\app` 目录下进行编译。





## 2.1. 编译 esp\_iot\_sdk\_v0.9.5 及之后版本软件

esp\_iot\_sdk\_v0.9.5 及之后版本的软件简化了编译脚本。

编译指令：`./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):
```

根据提示，按用户需求输入编译参数。

注意，

- 不同的编译参数支持的 boot.bin 版本不同，如上图说明；推荐使用更新版本的 boot；
- 每个 bin 编译成功后，会提示该 bin 的烧录位置，例如

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

或者，

```
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$
```

## 2.2. 编译 esp\_iot\_sdk\_v0.9.4 及之前版本软件

不支持云端升级的编译指令为：`./gen_misc.sh`。

支持云端升级（FOTA）的编译步骤如下：

- (1) 运行 `./gen_misc_plus.sh 1`，在 `/esp_iot_sdk/bin/upgrade` 路径下生成 `user1.bin`
- (2) 运行 `make clean`，清除之前的编译信息；
- (3) 运行 `./gen_misc_plus.sh 2`，在 `/esp_iot_sdk/bin/upgrade` 路径下生成 `user2.bin`

注意：

- 1) 详细的云端升级功能说明，请参见文档“云端升级实现方案”。
- 2) `esp_iot_sdk_v0.7` 及以前的版本，不支持云端升级。
- 3) `esp_iot_sdk_v0.8` 及之后的版本，支持云端升级，同时也兼容之前的编译及烧录方式。





### 3. 烧录说明

以下两种方式，根据实际需求，选择一种烧录方法即可，可以根据编译完成时的提示地址烧录。

#### 3.1. 不支持云端升级

注意:

- `master_device_key.bin` 是 ESP8266 设备享受 Espressif 云端服务的身份证明，如不使用 Espressif Cloud 可以不烧录，否则，仅烧录一次即可；
- 主程序 bin 文件为: `eagle.app.v6.flash.bin` 和 `eagle.app.v6.irom0text.bin`。
- 烧录 `blank.bin` 作为初始化

##### 1. 512KB Flash

- `blank.bin`: 由 Espressif 在 SDK 中提供，烧录到 `0x7E000`
- `eagle.app.v6.flash.bin`: 如上编译生成，烧录到 `0x00000`
- `master_device_key.bin`: 向 Espressif 云端服务器申请，烧录到 `0x3E000`
- `eagle.app.v6.irom0text.bin`: 如上编译生成，烧录到 `0x40000`
- `esp_init_data_default.bin`: 由 Espressif 提供，存储射频相关参数的初始值，烧录到 `0x7C000`。

##### 2. 1MB 及以上容量 Flash

- `blank.bin`: 由 Espressif 在 SDK 中提供，烧录到 Flash 的倒数第二个扇区，例如 1MB Flash 烧到 `0xFE000`, 4MB Flash 烧到 `0x3FE000`
- `eagle.app.v6.flash.bin`: 如上编译生成，烧录到 `0x00000`
- `master_device_key.bin`: 向 Espressif 云端服务器申请，烧录到 `0x7E000`
- `eagle.app.v6.irom0text.bin`: 如上编译生成，烧录到 `0x80000`
- `esp_init_data_default.bin`: 由 Espressif 提供，存储射频相关参数的初始值，烧录到 Flash 的倒数第四个扇区，例如 1MB Flash 烧到 `0xFC000`, 4MB Flash 烧到 `0x3FC000`

#### 3.2. 支持云端升级(FOTA)

注意:

- 支持云端升级 (FOTA) 的软件无需烧录 `user2.bin`，可以通过网络升级下载 `user2.bin` 到 Flash 并重启运行，后文仅作为说明 `user2.bin` 的实际存放位置。
- `master_device_key.bin` 是 ESP8266 设备享受 Espressif 云端服务的身份证明，如不使用 Espressif Cloud 可以不烧录，否则，仅烧录一次即可；

##### 1. 512KB Flash

- `blank.bin`: 由 Espressif 在 SDK 中提供，烧录到 Flash `0x7E000`;
- `esp_init_data_default.bin`: 由 Espressif 提供，存储射频相关参数的初始值，烧录到 `0x7C000`。
- `boot.bin`: 由 Espressif 在 SDK 中提供，烧录到 `0x00000`;



- `user1.bin`: 如上编译生成, 烧录到 `0x01000`;
- `user2.bin`: 如上编译生成, 烧录到 `0x41000`;
- `master_device_key.bin`: 向 Espressif 云端服务器申请, 烧录到 `0x3E000`;

### 2. 1MB 及以上容量 Flash

- `blank.bin`: 由 Espressif 在 SDK 中提供, 烧录到 Flash 的倒数第二个扇区, 例如 1MB Flash 烧到 `0xFE000`, 4MB Flash 烧到 `0x3FE000`
- `boot.bin`: 由 Espressif 在 SDK 中提供, 烧录到 `0x00000`;
- `user1.bin`: 如上编译生成, 烧录到 `0x01000`;
- `user2.bin`: 如上编译生成, 烧录到 `0x81000`;
- `master_device_key.bin`: 向 Espressif 云端服务器申请, 烧录到 `0x7E000`
- `esp_init_data_default.bin`: 由 Espressif 提供, 存储射频相关参数的初始值, 烧录到 Flash 的倒数第四个扇区, 例如 1MB Flash 烧到 `0xFC000`, 4MB Flash 烧到 `0x3FC000`