



ESP8266 SDK User Manual

Version 1.5

Espressif Systems IOT Team

Copyright © 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© 2015 乐鑫信息科技（上海）有限公司所有。保留所有权利。



Table of Contents

1.	前言.....	5
2.	开发工具	6
2.2.	烧录工具 - FLASH_DOWNLOAD_TOOLS.....	6
2.1.	串口工具	7
3.	SDK 软件包.....	8
4.	编译.....	9
4.1.	编译 esp_iot_sdk_v0.9.5 及之后版本软件	9
4.2.	编译 esp_iot_sdk_v0.9.4 及之前版本软件	11
5.	Flash Map.....	12
5.1.	none boot - 不支持云端升级	12
1.	512KB flash.....	12
2.	1024KB flash.....	13
3.	2048KB flash.....	14
4.	4096KB flash.....	15
5.2.	with boot - 支持云端升级	16
1.	512KB flash.....	16
2.	1024KB flash.....	16
3.	2048KB flash.....	17
4.	4096KB flash.....	18
6.	烧录说明	19
6.1.	不支持云端升级	19
1.	512KB Flash.....	19
2.	1024KB Flash	19
3.	2048KB Flash.....	20
4.	4096KB Flash.....	20
6.2.	支持云端升级(FOTA)	20
1.	512KB Flash.....	21
2.	1024KB Flash.....	21
3.	2048KB Flash.....	22
4.	4096KB Flash.....	22



7. 附录.....	23
------------	----



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/viewtopic.php?f=67&t=821>



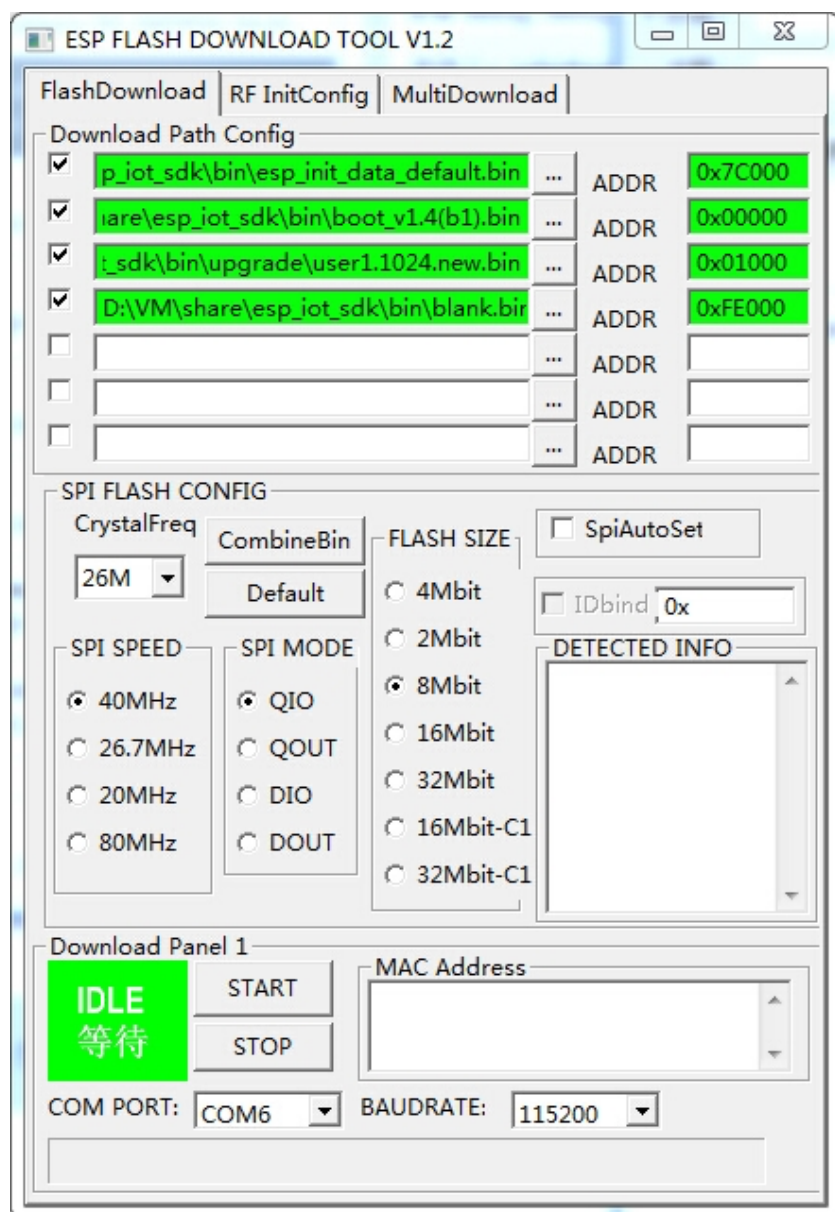
2.

开发工具

2.2. 烧录工具 - FLASH_DOWNLOAD_TOOLS

Espressif 官方烧录工具 “ESP_FLASH_DOWNLOAD_TOOL” 可在 BBS 下载。

下载链接: <http://bbs.espressif.com/viewtopic.php?f=57&t=433>





该工具实现了多个 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_TOOL 的烧录使用说明：

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。

2.1. 串口工具

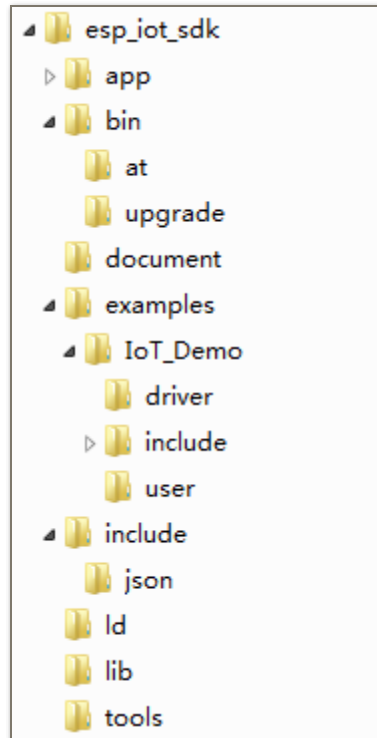
ESP8266 模块默认采用 74880 波特率，请注意在串口工具中设置。



3.

SDK 软件包

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，如下图：



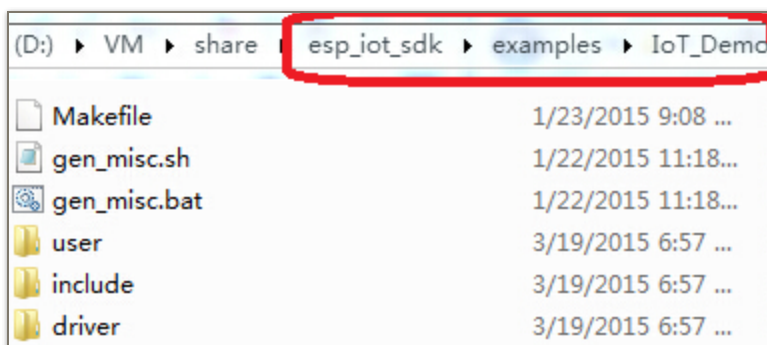
- "app" 目录为用户工作主目录，用户级代码及头文件均放在此目录下编译。
- "bin" 目录存放需下载到 Flash 的 bin 文件，其中：
 - ▶ "at" 文件夹 - Espressif 提供的支持 AT+ 指令的 bin 文件；
 - ▶ "upgrade" 文件夹 - 编译生成的支持云端升级的 bin 文件（user1.bin 或 user2.bin）；
 - ▶ "bin" 文件夹根目录 - 编译生成的不支持云端升级的 bin 文件，和其他 Espressif 提供的 bin 文件。
- "examples" 目录存放 SDK 的上层示例代码，使用时需将子目录（例如 IoT_Demo 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。



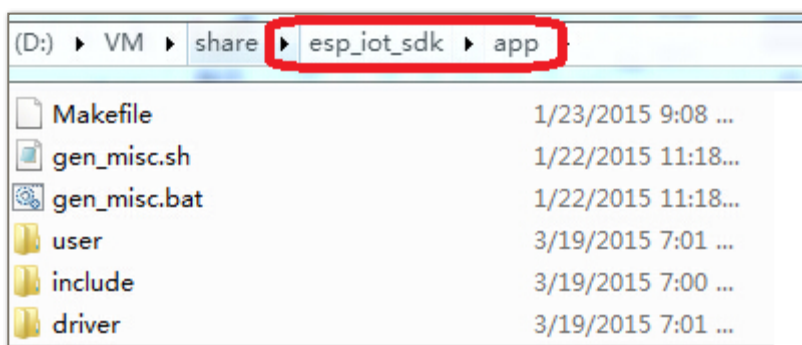
4.

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。



4.1. 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。

STEP 1 选择 boot

0	boot_v1.1, 旧版本 boot, 可支持云端升级
---	------------------------------



STEP 1 选择 boot	
1	boot_v1.2+, 新版本 boot, 始终建议使用最新版本的 boot, 可支持云端升级
2	无 boot, 编译生成 <code>eagle.flash.bin</code> 和 <code>eagle.irom0text.bin</code> , 不支持云端升级
STEP 2 选择生成的 bin 文件	
0	STEP 1 选择 2, 编译生成 <code>eagle.flash.bin</code> 和 <code>eagle.irom0text.bin</code> , 不支持云端升级
1	STEP 1 选择 0 或 1, 编译生成 <code>user1.bin</code> , 可支持云端升级
2	STEP 1 选择 0 或 1, 编译生成 <code>user2.bin</code> , 可支持云端升级
STEP 3 设置 SPI flash 配置 (SPI speed)	
0	SPI speed 20MHz, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
1	SPI speed 26.7MHz, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
2	SPI speed 40MHz, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
3	SPI speed 80MHz, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
STEP 4 设置 SPI flash 配置 (SPI mode)	
0	QIO, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
1	QOUT, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
2	DIO, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
3	DOUT, 烧录时 flash download tool 需同样选择, 与实际硬件 SPI flash 对应
STEP 5 设置 SPI flash 配置 (SPI flash size & map)	
0	flash size 512KB, 代码区对半划分为 256KB + 256KB
2	flash size 1024KB, STEP 1 选择 0 或 1 时, 代码区对半划分为 512KB + 512KB
3	flash size 2048KB, STEP 1 选择 0 或 1 时, 前 1024KB 为代码区, 对半划分为 512KB + 512KB
4	flash size 4096KB, STEP 1 选择 0 或 1 时, 前 1024KB 为代码区, 划分为 512KB + 512KB
5	flash size 2048KB, STEP 1 选择 0 或 1 时, 代码区对半划分为 1024KB + 1024KB sdk_v1.1.0 + boot 1.4 + flash download tool_v1.2 及之后的版本支持
6	flash size 4096KB, STEP 1 选择 0 或 1 时, 前 2048KB 为代码区, 划分为 1024KB + 1024KB sdk_v1.1.0 + boot 1.4 + flash download tool_v1.2 及之后的版本支持



- `none boot`：编译生成 `eagle.flash.bin` 和 `eagle.irom0text.bin`，不支持云端升级功能。
- `boot_v1.1` & `boot_v1.2+`，始终推荐使用最新版本的 `boot`，旧版本保留是为了兼容部分客户的旧版本软件；选择使用 `boot`，编译生成 `user1.bin` 或者 `user2.bin`，支持实现云端升级功能。
- 编译生成 `user1.bin` 后，请先运行 `make clean` 清除上次编译生成的临时文件后，再编译生成 `user2.bin`。
- 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$
```

4.2. 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。



5. Flash Map

针对编译时 STEP 1 和 STEP 5 的选择不同，对应的 flash size 和 flash map 不同。

注意

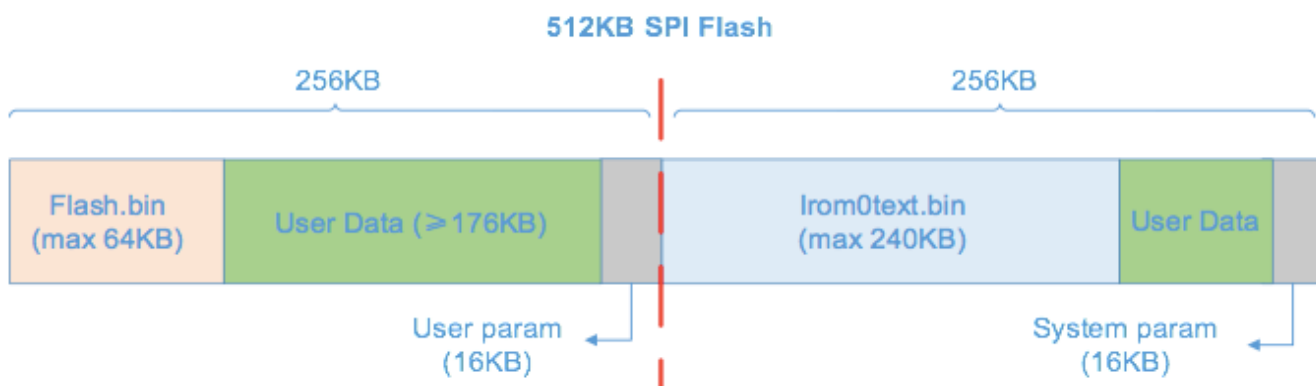
- 系统参数区（System param）始终为 flash 的最后 16KB。
- 用户参数区（User param）指 Espressif 提供的示例软件（IOT_Demo 或 AT）中设定的用户参数区。如果用户自行实现应用程序，则可以将用户参数存放在 flash 任意空闲区域。
- 后文图中 User Data（绿色）区域即表示可能空闲，当程序区未占满 flash 空间时，剩余空间可供用户存储数据的区域。

5.1. none boot - 不支持云端升级

编译时 STEP 1 选择 2 none boot，编译生成 `eagle.flash.bin`（后文简称为 `flash.bin`）和 `eagle.irom0text.bin`（后文简称为 `irom0text.bin`），不支持云端升级功能。则 STEP 5 时选择不同 flash size 对应的布局如下。

1. 512KB flash

若编译时 STEP 1 选择 2，STEP 5 选择 0，则 flash map 如下图



- User Data 区域：当程序区（`flash.bin`和 `irom0text.bin`）未占满整个空间时，空闲区域均可用于存放用户数据。
- 上图 `irom0text.bin` 默认最大值为 200KB；对于 512KB flash，用户可修改编译文件，使其最大支持到 $256 - 16 = 240$ KB

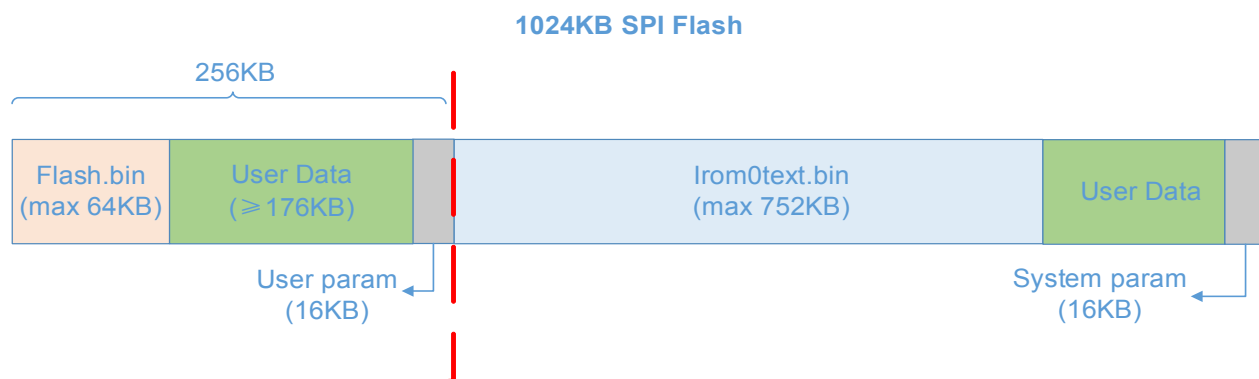


- `\esp_iot_sdk\ld` 路径的“`eagle.app.v6.ld`”文件，其中 `irom0_0_seg` 的 `len` 即设置 `irom0text.bin` 的上限值。对于 512KB flash，此 `len` 最大可修改为 `0x3C000`，`irom0text.bin` 最大支持到 240 KB

```
MEMORY
{
    dport0_0_seg :                org = 0x3FF00000, len = 0x10
    dram0_0_seg :                 org = 0x3FFE8000, len = 0x14000
    iraml_0_seg :                 org = 0x40100000, len = 0x8000
    irom0_0_seg :                 org = 0x40240000, len = 0x32000
}
```

2. 1024KB flash

若编译时 STEP 1 选择 2，STEP 5 选择 2，则 flash map 如下



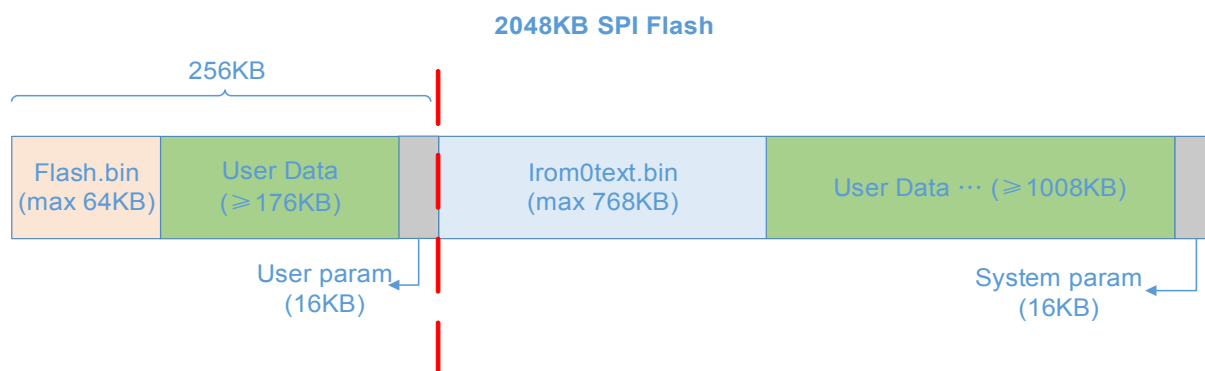
- User Data 区域：当程序区（`flash.bin`和 `irom0text.bin`）未占满整个空间时，空闲区域均可用于存放用户数据。
- 上图 `irom0text.bin` 默认最大值为 200KB；对于 1024KB flash，用户可修改编译文件，使其最大支持到 $1024 - 256 - 16 = 752$ KB
- `\esp_iot_sdk\ld` 路径的“`eagle.app.v6.ld`”文件，其中 `irom0_0_seg` 的 `len` 即设置 `irom0text.bin` 的上限值。对于 1024KB flash，此 `len` 最大可修改为 `0xBC000`，`irom0text.bin` 最大支持到 752 KB



```
MEMORY
{
    dport0_0_seg :                org = 0x3FF00000, len = 0x10
    dram0_0_seg :                  org = 0x3FFE8000, len = 0x14000
    iraml_0_seg :                   org = 0x40100000, len = 0x8000
    irom0_0_seg :                   org = 0x40240000, len = 0x32000
}
```

3. 2048KB flash

若编译时 STEP 1 选择 2，STEP 5 选择 3，则 flash map 如下



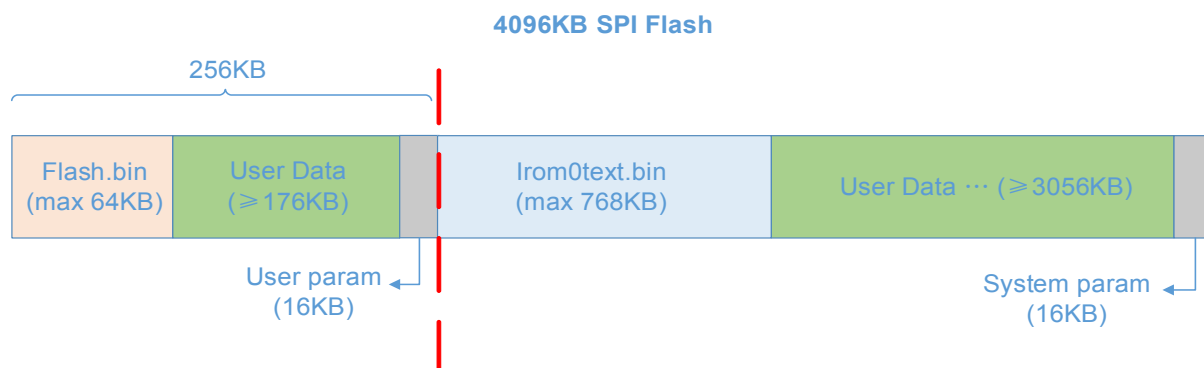
- User Data 区域：当程序区（`flash.bin`和 `irom0text.bin`）未占满整个空间时，空闲区域均可用于存放用户数据。
- 上图 `irom0text.bin` 默认最大值为 200KB；ESP8266 目前程序区最大支持 1024KB，因此对于 2048KB flash，用户可修改编译文件，使其最大支持到 $1024 - 256 = 768$ KB
- `\esp_iot_sdk\ld` 路径的“`eagle.app.v6.ld`”文件，其中 `irom0_0_seg` 的 `len` 即设置 `irom0text.bin` 上限值。对于 2048KB flash，此 `len` 最大可修改为 `0xC0000`，`irom0text.bin` 最大支持到 768 KB

```
MEMORY
{
    dport0_0_seg :                org = 0x3FF00000, len = 0x10
    dram0_0_seg :                  org = 0x3FFE8000, len = 0x14000
    iraml_0_seg :                   org = 0x40100000, len = 0x8000
    irom0_0_seg :                   org = 0x40240000, len = 0x32000
}
```



4. 4096KB flash

若编译时 STEP 1 选择 2，STEP 5 选择 4，则 flash map 如下



- User Data 区域：当程序区（`flash.bin`和 `irom0text.bin`）未占满整个空间时，空闲区域均可用于存放用户数据。
- 上图 `irom0text.bin` 默认最大值为 200KB；ESP8266 目前程序区最大支持 1024KB，因此对于 4096KB flash，用户可修改编译文件，使其最大支持到 $1024 - 256 = 768$ KB
- `\esp_iot_sdk\ld` 路径的“`eagle.app.v6.ld`”文件，其中 `irom0_0_seg` 的 `len` 即设置 `irom0text.bin` 上限值。对于 4096KB flash，此 `len` 最大可修改为 `0xC0000`，`irom0text.bin` 最大支持到 768 KB

```
MEMORY
{
    dport0_0_seg :                org = 0x3FF00000, len = 0x10
    dram0_0_seg :                  org = 0x3FFE8000, len = 0x14000
    iraml_0_seg :                  org = 0x40100000, len = 0x8000
    irom0_0_seg :                  org = 0x40240000, len = 0x32000
}
```



5.2. with boot - 支持云端升级

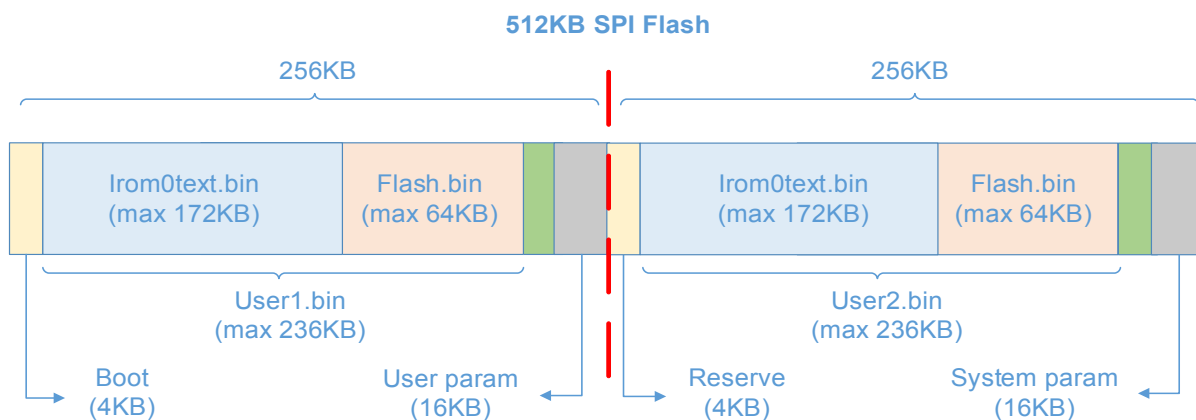
编译时 STEP 1 选择 1 boot_v1.2+, 编译两次, 分别生成 `user1.bin` 和 `user2.bin`, 支持云端升级功能。则 STEP 5 时选择不同 flash size 对应的布局如下。

注意:

- 编译生成 `user1.bin` 之后, 先 `make clean`, 清除上次编译生成的临时文件, 再编译同样配置的 `user2.bin`
- boot_v1.1 为旧版本 boot, 仅为兼容部分使用旧版本软件的用户, 编译烧录基本相同, 不另做详细说明。
- 后文图中绿色区域, 均表示 User Data 区域; 当程序区 (`user1.bin` 或 `user2.bin`) 未占满整个空间时, 空闲区域均可用于存放用户数据。

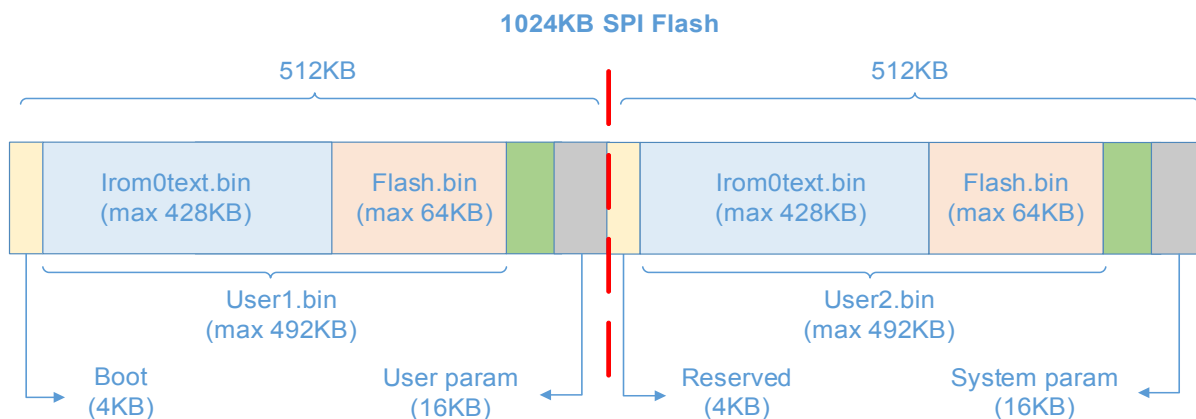
1. 512KB flash

若编译时 STEP 1 选择 1, STEP 5 选择 0, 则 flash map 如下



2. 1024KB flash

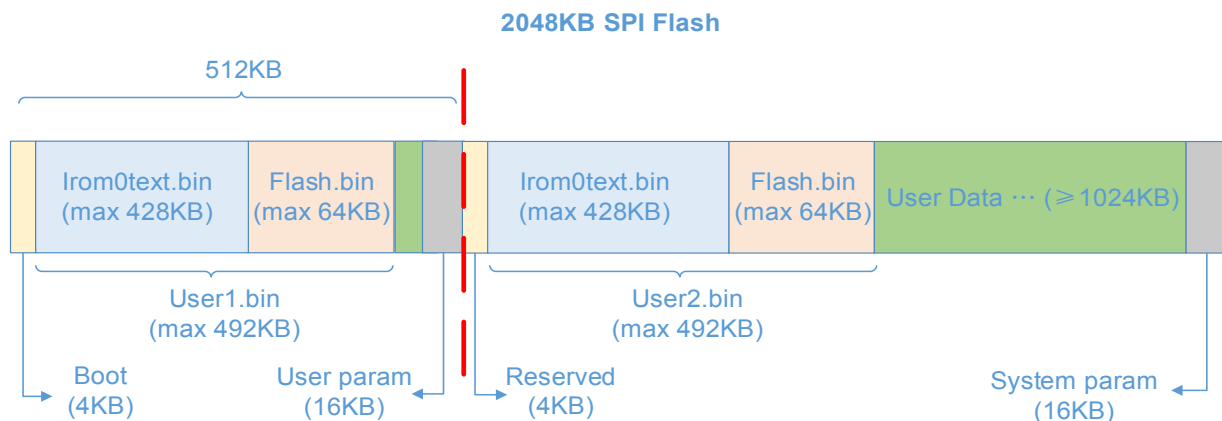
若编译时 STEP 1 选择 1, STEP 5 选择 2, 则 flash map 如下



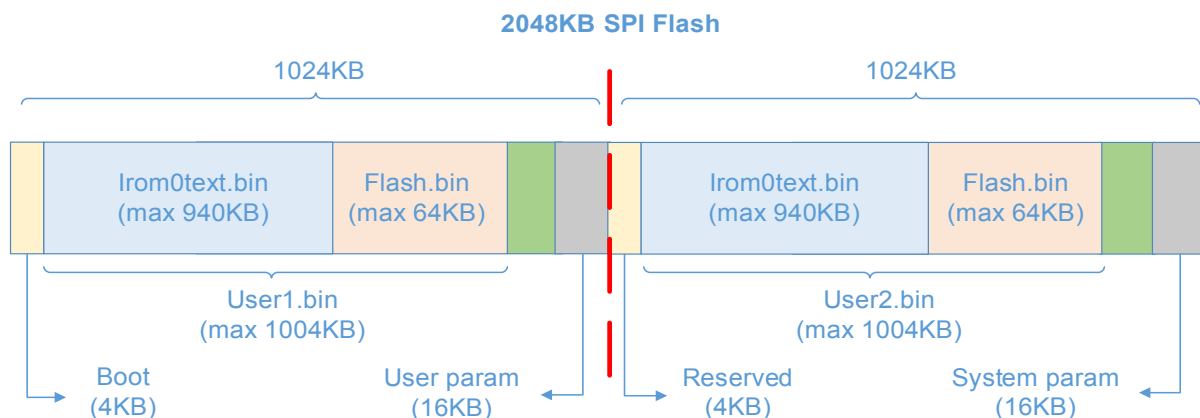


3. 2048KB flash

若编译时 STEP 1 选择 1，STEP 5 选择 3，仅前 1024KB 为代码区，对半划分为 512KB + 512KB，则 flash map 如下



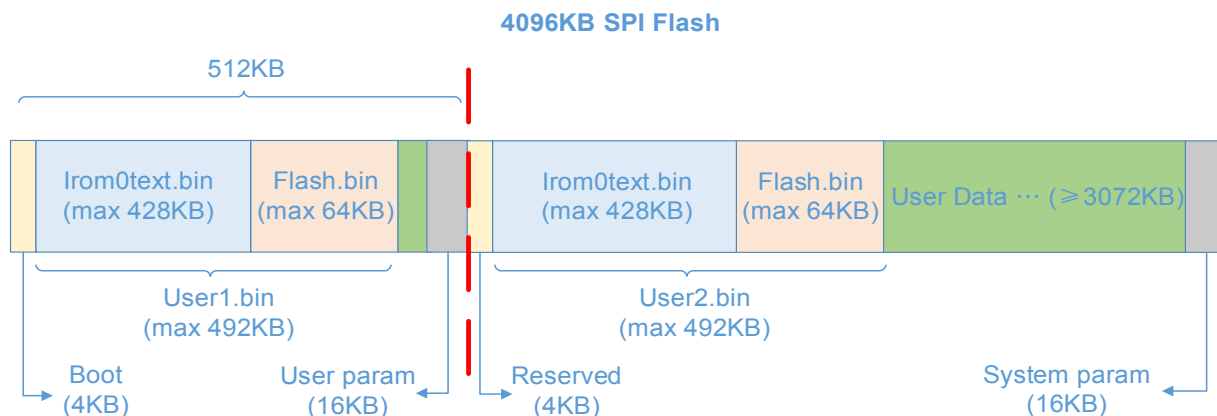
若编译时 STEP 1 选择 1，STEP 5 选择 5，代码区对半划分为 1024KB + 1024KB，sdk_v1.1.0 + boot 1.4 + flash download tool v1.2 及之后的版本支持，Flash map 如下



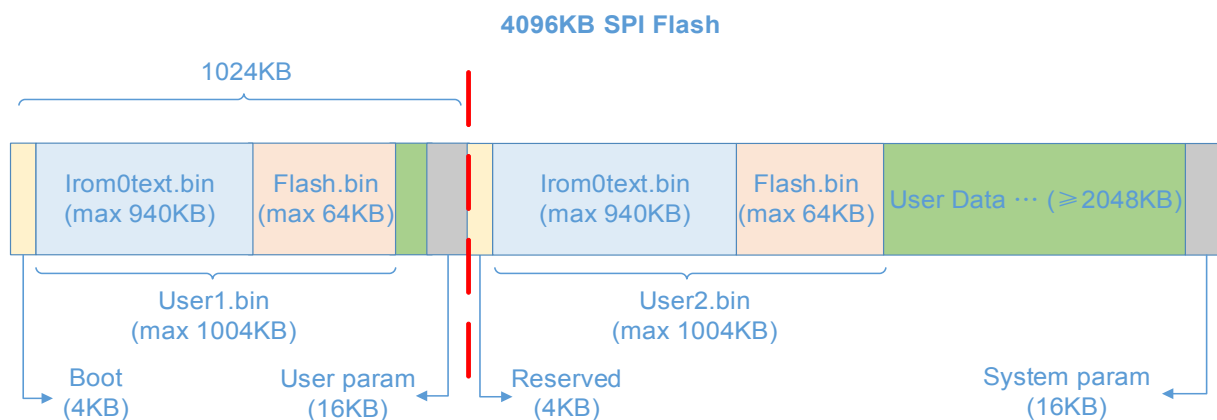


4. 4096KB flash

若编译时 STEP 1 选择 1，STEP 5 选择 4，仅前 1024KB 为代码区，对半划分为 512KB + 512KB，则 flash map 如下



若编译时 STEP 1 选择 1，STEP 5 选择 6，前 2048KB 为代码区，对半划分为 1024KB + 1024KB，在 sdk_v1.1.0 + boot 1.4 + flash download tool v1.2 及之后的版本支持，flash map 如下





6.

烧录说明

根据实际编译方式和 flash 容量，选择烧录方法，可以根据编译完成时的提示地址烧录。

注意:

- 系统参数区固定为 flash 的最后四个扇区，每扇区 4KBytes，即 flash 最后 16KB；

用户参数区地址由用户自定义，IOT_Demo 中设置为 0x3C000 开始的四个扇区，用户可以设置为任意未占用的地址。

- `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；烧录地址在 IOT_Demo 中设置为用户参数区的第三个扇区；
- `blank.bin` 初始化系统参数，烧录地址为 flash 的倒数第二个扇区；
- `esp_init_data_default.bin` 初始化射频相关参数，烧录地址为 flash 的倒数第四个扇区；
- ESP8266_NONOS_SDK 提供的示例 IoT_Demo 默认为 512KB flash map，IoT_Demo 详细介绍请参考文档“2B-ESP8266__SDK__IoT_Demo”。

6.1. 不支持云端升级

1. 512KB Flash

bin	烧录地址	说明
<code>master_device_key.bin</code>	0x3E000	用户在 Espressif Cloud 申请，依此享受 Espressif 云端服务
<code>esp_init_data_default.bin</code>	0x7C000	初始化射频参数，由 Espressif 在 SDK 中提供
<code>blank.bin</code>	0x7E000	初始化系统参数，由 Espressif 在 SDK 中提供
<code>eagle.flash.bin</code>	0x00000	主程序，编译代码生成
<code>eagle.irom0text.bin</code>	0x40000	主程序，编译代码生成

2. 1024KB Flash

bin	烧录地址	说明
<code>master_device_key.bin</code>	0x3E000	用户在 Espressif Cloud 申请，依此享受 Espressif 云端服务
<code>esp_init_data_default.bin</code>	0xFC000	初始化射频参数，由 Espressif 在 SDK 中提供



bin	烧录地址	说明
blank.bin	0xFE000	初始化系统参数，由 Espressif 在 SDK 中提供
eagle.flash.bin	0x00000	主程序，编译代码生成
eagle.irom0text.bin	0x40000	主程序，编译代码生成

3. 2048KB Flash

bin	烧录地址	说明
master_device_key.bin	0x3E000	用户在 Espressif Cloud 申请，依此享受 Espressif 云端服务
esp_init_data_default.bin	0x1FC000	初始化射频参数，由 Espressif 在 SDK 中提供
blank.bin	0x1FE000	初始化系统参数，由 Espressif 在 SDK 中提供
eagle.flash.bin	0x00000	主程序，编译代码生成
eagle.irom0text.bin	0x40000	主程序，编译代码生成

4. 4096KB Flash

bin	烧录地址	说明
master_device_key.bin	0x3E000	用户在 Espressif Cloud 申请，依此享受 Espressif 云端服务
esp_init_data_default.bin	0x3FC000	初始化射频参数，由 Espressif 在 SDK 中提供
blank.bin	0x3FE000	初始化系统参数，由 Espressif 在 SDK 中提供
eagle.flash.bin	0x00000	主程序，编译代码生成
eagle.irom0text.bin	0x40000	主程序，编译代码生成

6.2. 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。



1. 512KB Flash

bin	烧录地址	说明
master_device_key.bin	0x3E000	用户在 Espressif Cloud 申请，依此享受 Espressif 云端服务
esp_init_data_default.bin	0x7C000	初始化射频参数，由 Espressif 在 SDK 中提供
blank.bin	0x7E000	初始化系统参数，由 Espressif 在 SDK 中提供
boot.bin	0x00000	启动程序，由 Espressif 在 SDK 中提供，建议使用最新版本
user1.bin	0x01000	主程序，编译代码生成
user2.bin	0x41000	主程序，编译代码生成，无需烧录

2. 1024KB Flash

bin	烧录地址	说明
master_device_key.bin	0x3E000 (建议更改)	用户在 Espressif Cloud 申请，依此享受 Espressif 云端服务，存放于用户参数区，IOT_Demo 中设置为 0x3E000，用户可自行更改。建议使用 1MB flash 时，参考 BBS 修改，烧录到 0x7E000 http://bbs.espressif.com/viewtopic.php?f=10&t=305
esp_init_data_default.bin	0xFC000	初始化射频参数，由 Espressif 在 SDK 中提供
blank.bin	0xFE000	初始化系统参数，由 Espressif 在 SDK 中提供
boot.bin	0x00000	启动程序，由 Espressif 在 SDK 中提供，建议使用最新版本
user1.bin	0x01000	主程序，编译代码生成
user2.bin	0x81000	主程序，编译代码生成，无需烧录



3. 2048KB Flash

bin	烧录地址	说明
master_device_key.bin	0x3E000 (建议更改)	用户在 Espressif Cloud 申请, 依此享受 Espressif 云端服务, 存放于用户参数区, IOT_Demo 中设置为 0x3E000, 用户可自行更改。 建议如果编译时 STEP 5 选择 3, 则参考 BBS 修改, 烧录到 0x7E000 http://bbs.espressif.com/viewtopic.php?f=10&t=305 如果编译时 STEP 5 选择 5, 则同理修改代码地址为 0xFE000, 并烧录到 0xFE000
esp_init_data_default.bin	0x1FC000	初始化射频参数, 由 Espressif 在 SDK 中提供
blank.bin	0x1FE000	初始化系统参数, 由 Espressif 在 SDK 中提供
boot.bin	0x00000	启动程序, 由 Espressif 在 SDK 中提供, 建议使用最新版本
user1.bin	0x01000	主程序, 编译代码生成
user2.bin	0x81000	主程序, 编译代码生成, 无需烧录

4. 4096KB Flash

bin	烧录地址	说明
master_device_key.bin	0x3E000 (建议更改)	用户在 Espressif Cloud 申请, 依此享受 Espressif 云端服务, 存放于用户参数区, IOT_Demo 中设置为 0x3E000, 用户可自行更改。 建议如果编译时 STEP 5 选择 4, 则参考 BBS 修改, 烧录到 0x7E000 http://bbs.espressif.com/viewtopic.php?f=10&t=305 如果编译时 STEP 5 选择 6, 则同修修改代码地址为 0xFE000, 并烧录到 0xFE000
esp_init_data_default.bin	0x3FC000	初始化射频参数, 由 Espressif 在 SDK 中提供
blank.bin	0x3FE000	初始化系统参数, 由 Espressif 在 SDK 中提供
boot.bin	0x00000	启动程序, 由 Espressif 在 SDK 中提供, 建议使用最新版本
user1.bin	0x01000	主程序, 编译代码生成
user2.bin	0x81000	主程序, 编译代码生成, 无需烧录



7.

附录

从 **esp_iot_sdk_v1.4.0** 版本起，开发者可以通过设置 `esp_init_data_default.bin` (0~127 byte) 的 114 byte 控制上电时 RF 初始化的行为，具体如下：

- 114 byte 默认值为 0：RF 初始化仅做 VDD33 校准，耗时约 2 ms，初始电流最小；
- 114 byte 若修改为 1：RF 初始化做 VDD33 和 TX power CAL，耗时约 18 ms，初始电流较小；
- 114 byte 若修改为 2：RF 初始化仅做 VDD33 校准，耗时约 2 ms，初始电流最小；与 0 值时一致；
- 114 byte 若修改为 3：RF 初始化进行全部 RF CAL，耗时约 200 ms，初始电流较大；

如需修改 114 byte，则步骤如下：

(1) 打开工具“ESP FLASH DOWNLOAD TOOL”，选择“RF InitConfig”

ESP FLASH DOWNLOAD TOOL V2.3

FlashDownload **RF InitConfig** MultiDownload

RF Settings

TxTargetPowerConfig

MCS	Value	Unit
MCS0-1	20.5	dBm
MCS2-3	19.5	dBm
MCS4	18.5	dBm
MCS5	17	dBm
MCS6	16	dBm
MCS7	14	dBm

LowPowerMode

☐ LowPowerEn: 0 dB

☐ BackOffEn: 0 dB

☐ PowerLimitEn: 20.5 dBm

Buttons

Default

GenInitBin

LoadInitBin

CrystalFreq

☒ 40Mhz

☐ 26Mhz

☐ 24Mhz

TOUT PinConf

☒ TOUT_ADC_EN

VDD: 3.3 V

☐ TOUT_VDD_EN

FreqOffset

☐ SetFreqEnable

☒ AutoCalEn

PracticalFreqOffset: 50 KHz

RFInit mode

☐ LoadRFCalParam



☒ TxPwrCtrl in init

☐ FullRFCal in RFInit

	A	B	C	D	E
109	108	Reserved	Reserved	unsigned	0
110	109	Reserved	Reserved	unsigned	0
111	110	Reserved	Reserved	unsigned	0
112	111	Reserved	Reserved	unsigned	0
113	112	tx_param42	freq_correct_en	unsigned	3
114	113	tx_param43	force_freq_offset	unsigned	0
115	114	tx_param44	rf_cal_use_flash	unsigned	1
116	115	Reserved	Reserved	unsigned	0



- (2) 在 RFInit mode 一栏可以设置 RF 初始化的行为，例如，设置为选项 TxPwrCtrl in init，最下方的表格指示 esp_init_data_default.bin (0~127 byte)，记录 RF 相关的各项参数，可以看到其中的 Byte 114 对应修改为 1，如上图。
- (3) 点击 “GenInitBin” 生成 esp_init_data_setting.bin，将它代替 esp_init_data_default.bin 烧录到 Flash 中即可。

 esp_init_data_setting.bin	8/7/2015 1:45 PM	VLC media file (....	1 KB
 ESP8266_RF_init.xls	5/7/2015 5:15 PM	Microsoft Excel ...	48 KB