



---

# ESP8266 智能开关官方例程 使用教程

---



2016-7-19

PillarPeng

# 目录

<b>0. 提前准备:</b> .....	<b>2</b>
<b>1. 下载源码</b> .....	<b>2</b>
1.1 下载官方 RTOS_SDK .....	2
1.2 下载 ESP8266_IOT_PLATFORM 源码.....	2
<b>2. 配置源码</b> .....	<b>2</b>
2.1 放置 RTOS_SDK 源码.....	2
2.2 放置 ESP8266_IOT_PLATFORM 源码.....	3
2.3 确定 RTOS_SDK 的路径和编译后生成的 BIN 文件路径.....	3
<b>3. 编译固件</b> .....	<b>3</b>
3.2 编译可能出现的错误.....	5
<b>4. 获取 mater-device-key.bin</b> .....	<b>7</b>
4.1 注册 IOT 账号.....	7
4.2 登录账号.....	7
4.3 创建一个“开关”设备.....	7
4.4 新建开关设备的数据模型.....	8
4.5 下载 master-device-key.bin.....	8
<b>5. 下载固件</b> .....	<b>9</b>
5.1 配置下载工具.....	9
5.2 下载.....	9
<b>6. 手机 IOT APP</b> .....	<b>10</b>
6.1 下载 IOT APP.....	10
6.2 注册 IOT 账号, 登录 IOT APP.....	10
<b>7. 设备连接上云端</b> .....	<b>11</b>
7.1 APP 配置设备.....	11
7.2 APP 控制设备 .....	12

# ESP8266 智能开关官方例程使用教程

----基于官方 RTOS\_SDK 与 IOT\_PLATFORM

## 0. 提前准备:

0.1 ESP8266EX 开发板一个。(我测试使用的是官方开发板 ESP-LAUNCHER(RMB:150 不包邮), 其实只要是 ESP8266EX 模组都是适用的, 但是 flash 必须是 8Mbit 的或者以上的)

0.2 安卓手机一台。(Smartisan T1)

0.3 电脑一台。(windows 7 系统)

0.4 路由器(联网, 手机热点也可以的, 只要是 AP 都可以)

0.5 编译环境的搭建。(我使用的是 cygwin, windows 下面的 linux 环境, 也可以用虚拟机导入官方的 linux 镜像)

## 1. 下载源码

### 1.1 下载官方 RTOS\_SDK

我们要下载是 RTOS\_SDK, 不是 NONOS\_SDK。

- 方法 1: 直接去 github 上下载, 地址如下:  
[https://github.com/espressif/ESP8266\\_RTOS\\_SDK](https://github.com/espressif/ESP8266_RTOS_SDK)
- 方法 2: 去官方论坛, 找到最新的 RTOS\_SDK 下载。  
官方论坛: <http://bbs.espressif.com/> →  
论坛首页的右边“LASTEST SDK” →  
[“Download the lastest ESP8266 SDK here!”](#)

注: 其实都是到 github 上下载, 论坛最后的导向, 就是到 github。

### 1.2 下载 ESP8266\_IOT\_PLATFORM 源码

ESP8266\_IOT\_PLATFORM 包含多个 IOT demo, 其中有智能开关、智能插座、温湿度检测等 demo, 我们以智能开关为例。

下载 ESP8266\_IOT\_PLATFORM 的方法也有两种, 如下:

- 方法 1: 直接去 github 上下载, 地址如下:  
[https://github.com/espressif/ESP8266\\_IOT\\_PLATFORM](https://github.com/espressif/ESP8266_IOT_PLATFORM)
- 方法 2: 去官方论坛下载  
官方论坛: <http://bbs.espressif.com/> →  
DOWNLOADS →  
Demos →  
[ESP8266\\_IOT\\_PLATFORM](#)

注: 其实都是到 github 上下载, 论坛最后的导向, 就是到 github。

## 2. 配置源码

### 2.1 放置 RTOS\_SDK 源码

将 RTOS\_SDK 解压后, 复制到自己的工作目录。

我这里是: \cygwin\

如果使用的官方的虚拟镜像的话, 自己选一个工作目录即可, 没有什么特殊性。

## 2.2 放置 ESP8266\_IOT\_PLATFORM 源码

将 ESP8266\_IOT\_PLATFORM 解压后，复制到 RTOS\_SKD 的目录下。

```
\cygwin\ESP8266_RTOS_SDK-master\
```

存放路径最好是放在 RTOS\_SDK 的目录下。

## 2.3 确定 RTOS\_SDK 的路径和编译后生成的 BIN 文件路径

修改 ESP8266\_IOT\_PLATFORM 目录下 gen.misc.sh 文件，其位置在文件的头部，11 行左右

```
export SDK_PATH=/ESP8266_RTOS_SDK-master
```

```
export BIN_PATH=/ESP8266_RTOS_SDK-master/bin
```

注：

SDK\_PATH:：就是你存放 RTOS\_SDK 的路径

BIN\_PATH:：就是你想要编译后生成的 BIN 文件存放的路径，可以自定义，但是建议使用 SDK 目录下的 bin 目录。

# 3. 编译固件

## 3.1 编译

3.1.1 跳转到目录：/ESP8266\_RTOS\_SDK-master/ESP8266\_IOT\_PLATFORM

3.1.2 使用命令：./gen\_misc.sh

根据提示选择编译选项

1) Please check SDK\_PATH & BIN\_PATH, enter (Y/y) to continue:

输入：y

2) STEP 2: choose bin generate

输入：0

3) STEP 3: choose spi speed(0=20MHz, 1=26.7MHz, 2=40MHz, 3=80MHz)

输入：2

4) STEP 4: choose spi mode(0=QIO, 1=QOUT, 2=DIO, 3=DOU)

输入：0

5) STEP 5: choose spi size and map

0= 512KB( 256KB+ 256KB)

2=1024KB( 512KB+ 512KB)

3=2048KB( 512KB+ 512KB)

4=4096KB( 512KB+ 512KB)

5=2048KB(1024KB+1024KB)

6=4096KB(1024KB+1024KB)

输入：2

效果如下图所示：

```
Enqq01Enqq01-PC /ESP8266_RTOS_SDK-master/app
$ ./gen_misc.sh
gen_misc.sh version 20150911

SDK_PATH:
/ESP8266_RTOS_SDK-master

BIN_PATH:
/ESP8266_RTOS_SDK-master/bin

Please check SDK_PATH & BIN_PATH, enter <Y/y> to continue:
y

Please follow below steps<1-5> to generate specific bin(s):
STEP 1: use boot_v1.2+ by default
boot mode: new

STEP 2: choose bin generate<0=eagle.flash.bin+eagle.irom0text.bin 1=user1.bin,
2=user2.bin>
enter <0/1/2, default 0>:
0
ignore boot
generate bin: eagle.flash.bin+eagle.irom0text.bin

STEP 3: choose spi speed<0=20MHz, 1=26.7MHz, 2=40MHz, 3=80MHz>
enter <0/1/2/3, default 2>:
2
spi speed: 40 MHz

STEP 4: choose spi mode<0=QIO, 1=QOUT, 2=DIO, 3=DOUT>
enter <0/1/2/3, default 2>:
0
spi mode: DIO

STEP 5: choose spi size and map
0= 512KB< 256KB+ 256KB>
2=1024KB< 512KB+ 512KB>
3=2048KB< 512KB+ 512KB>
4=4096KB< 512KB+ 512KB>
5=2048KB<1024KB+1024KB>
6=4096KB<1024KB+1024KB>
enter <0/2/3/4/5/6, default 2>:
2
spi size: 1024KB
spi ota map: 512KB + 512KB

start...
```

3.1.3 编译成功后，如下图所示：

```
!!!
SDK_PATH: /ESP8266_RTOS_SDK-master
BIN_PATH: /ESP8266_RTOS_SDK-master/bin

No boot needed.
Generate eagle.flash.bin and eagle.irom0text.bin successully in BIN_PATH
eagle.flash.bin----->0x000000
eagle.irom0text.bin---->0x200000
!!!
```

记住这个地址，这就是下载固件时，这两个 bin 文件的 Flash 地址。

## 3.2 编译可能出现的错误

编译错误 1:

wifi/wifi.tpl (55%, heatshrink)

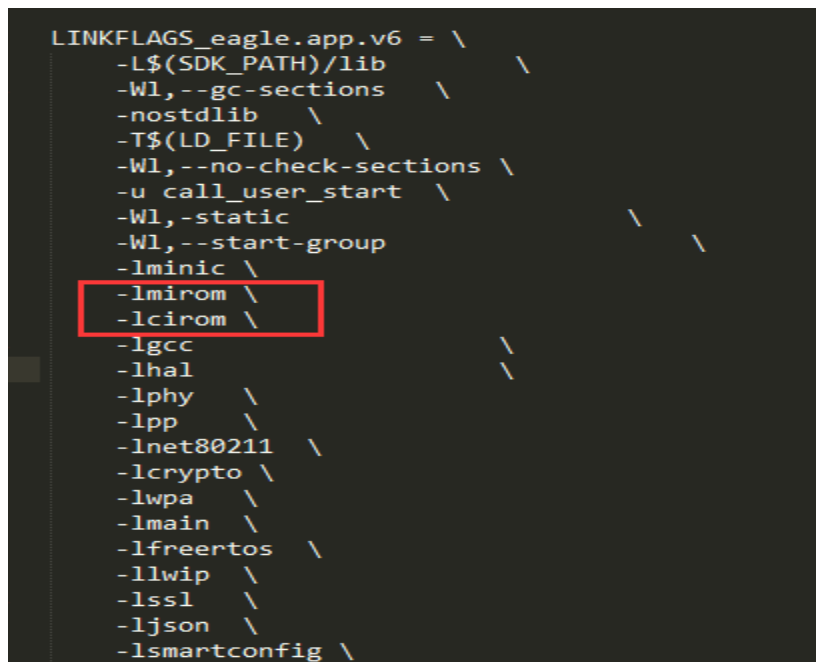
```
make[1]: Leaving directory '/ESP8266_RTOS_SDK-master-1/smartplug/libesphttpd'
xtensa-lx106-elf-gcc -L/ESP8266_RTOS_SDK-master-1/lib -Wl,--gc-sections -nostdlib -T/ESP8266_RTOS_SDK-master-1/ld/eagle.app.v6.ld -Wl,--no-check-sections -u call_user_start -Wl,-static -Wl,--start-group -lminic -lgcc -lhal -lphy -lpp -lnet80211 -lcrypto -lwpa -lmain -lfreertos -llwip -lssl -ljson -lsmartconfig -lpwm -L./libesphttpd -lesphttpd -lwebpages-espfs user/.output/eagle/debug/lib/libuser.a driver/.output/eagle/debug/lib/libdriver.a upgrade/.output/eagle/debug/lib/libupgrade.a -Wl,--end-group -o .output/eagle/debug/image/eagle.app.v6.out /ESP8266_RTOS_SDK-master-1/lib/libjson.a(cJSON.o):(.text.print_number+0x48): undefined reference to `floor'
/ESP8266_RTOS_SDK-master-1/lib/libjson.a(cJSON.o):(.text.print_number+0x148): undefined reference to `floor'
/ESP8266_RTOS_SDK-master-1/lib/libjson.a(cJSON.o):(.text.parse_value+0x14): undefined reference to `pow'
/ESP8266_RTOS_SDK-master-1/lib/libjson.a(cJSON.o):(.text.parse_value+0x270): undefined reference to `pow'
collect2: error: ld returned 1 exit status
/ESP8266_RTOS_SDK-master-1/Makefile:385: recipe for target '.output/eagle/debug/image/eagle.app.v6.out' failed
make: *** [.output/eagle/debug/image/eagle.app.v6.out] Error 1
```

解决 1:

修改文件: \ESP8266\_RTOS\_SDK-master\ESP8266\_IOT\_PLATFORM\Makefile

添加编译选项: (注意, 两个选项之间一定不要空行, 否则会报错)

如图所示:



```
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 \  
-lminic \  
-lmirom \  
-lcirom \  
-lgcc \  
-lhal \  
-lphy \  
-lpp \  
-lnet80211 \  
-lcrypto \  
-lwpa \  
-lmain \  
-lfreertos \  
-llwip \  
-lssl \  
-ljson \  
-lsmartconfig \  

```

编译出错 2:

wifi/wifi.tpl (55%, heatshrink)

```
make[1]: Leaving directory '/ESP8266_RTOS_SDK-master-1/smartplug/libesphttpd'
xtensa-lx106-elf-gcc -L/ESP8266_RTOS_SDK-master-1/lib -Wl,--gc-sections -nostdlib -T/ESP8266_RTOS_SDK-master-1/ld/eagle.app.v6.ld -Wl,--no-check-sections -u call_user_start -Wl,-static -Wl,--start-group -lminic -lgcc -lhal -lphy -lpp -lnet80211 -lcrypto -lwpa -lmain -lfreertos -llwip -lssl -ljson -lsmartconfig -lpwm -L./libesphttpd -lesphttpd -lwebpages-espfs -lcirom -lmirom user/.output/eagle/debug/lib/libuser.a driver/.output/eagle/debug/lib/libdriver.a upgrade/.output/eagle/debug/lib/libupgrade.a -Wl,--end-group -o .output/eagle/debug/image/eagle.app.v6.out
/ESP8266_RTOS_SDK-master-1/lib/libminic.a(minic.o): In function `sprintf':
(.irom0.text+0x364): multiple definition of `sprintf'
/ESP8266_RTOS_SDK-master-1/lib/libcirom.a(lib_a-sprintf.o):/home/wjg/Repo/esp-open-sdk/crosstool-NG/.build/src/newlib-2.0.0/newlib/libc/stdio/sprintf.c:624: first defined here
collect2: error: ld returned 1 exit status
/ESP8266_RTOS_SDK-master-1/Makefile:385: recipe for target '.output/eagle/debug/image/eagle.app.v6.out' failed
make: *** [.output/eagle/debug/image/eagle.app.v6.out] Error 1
```

解决 2:

注释掉，并将其放在命令语的末尾。或者或者删除掉，不然可能会报错。

```
2 -lmirom \
3 -lcirom \
4 -lgcc \
5 -lhal \
6 -lphy \
7 -lpp \
8 -lnet80211 \
9 -lcrypto \
0 -lwpa \
1 -lmain \
2 -lfreertos \
3 -llwip \
4 -lssl \
5 -ljson \
6 -lsmartconfig \
7 -lpwm \
8 -L./libesphttpd \
9 -lesphttpd \
0 -lwebpages-espfs \
1 $(DEP_LIBS_eagle.app.v6)
2 -Wl,--end-group
3
4 # -lminic \
5
```

编译出错 3:

Makefile:72: \*\*\* recipe commences before first target。 停止。

Makefile:150: warning: overriding recipe for target 'libwebpages-espfs.a'

Makefile:125: warning: ignoring old recipe for target 'libwebpages-espfs.a'

make[1]: Entering directory '/ESP8266\_RTOS\_SDK-master-1/smartplug/libesphttpd/espfs/mkspfsimage'

rm -f mkspfsimage main.o heatshrink\_encoder.o

make[1]: Leaving directory '/ESP8266\_RTOS\_SDK-master-1/smartplug/libesphttpd/espfs/mkspfsimage'

Makefile:72: \*\*\* recipe commences before first target。 停止。

解决 3:

指令格式错误。

可能是由于在错误 1 添加编译选项时，出现了格式错误，如选项之间的空行等。

## 4. 获取 mater-device-key.bin

Master-device-key.bin 是链接云端必要的密钥，获取他首先需要有一个 [iot.espressif.cn](http://iot.espressif.cn) 的一个账号，然后创建设备，创建后，每个设备都有唯一的一个密钥（Master-device-key.bin），最后从设备里，可以下载此密钥也就是 Master-device-key.bin。

### 4.1 注册 IOT 账号

去官方 IOT 网站：[iot.espressif.cn](http://iot.espressif.cn)，然后注册一个账户

### 4.2 登录账号

注册完后，应该就自动登录了。

### 4.3 创建一个“开关”设备

1) 设备开发 -> 创建



2) 除了产品类型选择开关“”，其他的随便写。



#### 4.4 新建开关设备的数据模型

- 1) 数据模型的名字：必须是 plug-status。
- 2) 维度：必须是一维。
- 3) 其他可暂时不填写。
- 4) 注意：我上面写“必须”，是针对于官方的 APP 和源码的。如果不该成上述内容，当 APP 和设备“云端”通信时，不能成功。  
但是，弄懂原理后，可自己修改 APP 的内容和源码中的内容。

#### 数据模型

创建数据模型 名字，必须

plug-status

维度

一维

单位

符号

标签

Celsius, Kilometre, Kilo

°C, KM, KG...

temperature...

描述

detail description

创建 取消

#### 4.5 下载 master-device-key.bin

如下图所示：

设备 { id: 132417, serial: 97d49529 }

SmartPlug-demo

Private Device

Product { id: 9, name: SmartPlug, serial: 50000000 }

Product Secret 67...

Device Secret 2...

Master Device Key 3...

Not Activated

Last Active 6分钟前

Device Status developing

这两处就是 master-device-key.bin 的具体值

点击此处，下载master-device-key.bin

数据模型

创建数据模型 名字，必须

plug-status

维度

一维

单位

符号

标签

Celsius, Kilometre, Kilo

°C, KM, KG...

temperature...

描述

detail description

请求日志

等待请求...

密钥

device key 3... (master)

device key 825f2c9a90d3f202599013bd06c97729ac327aff (owner)

+ 创建

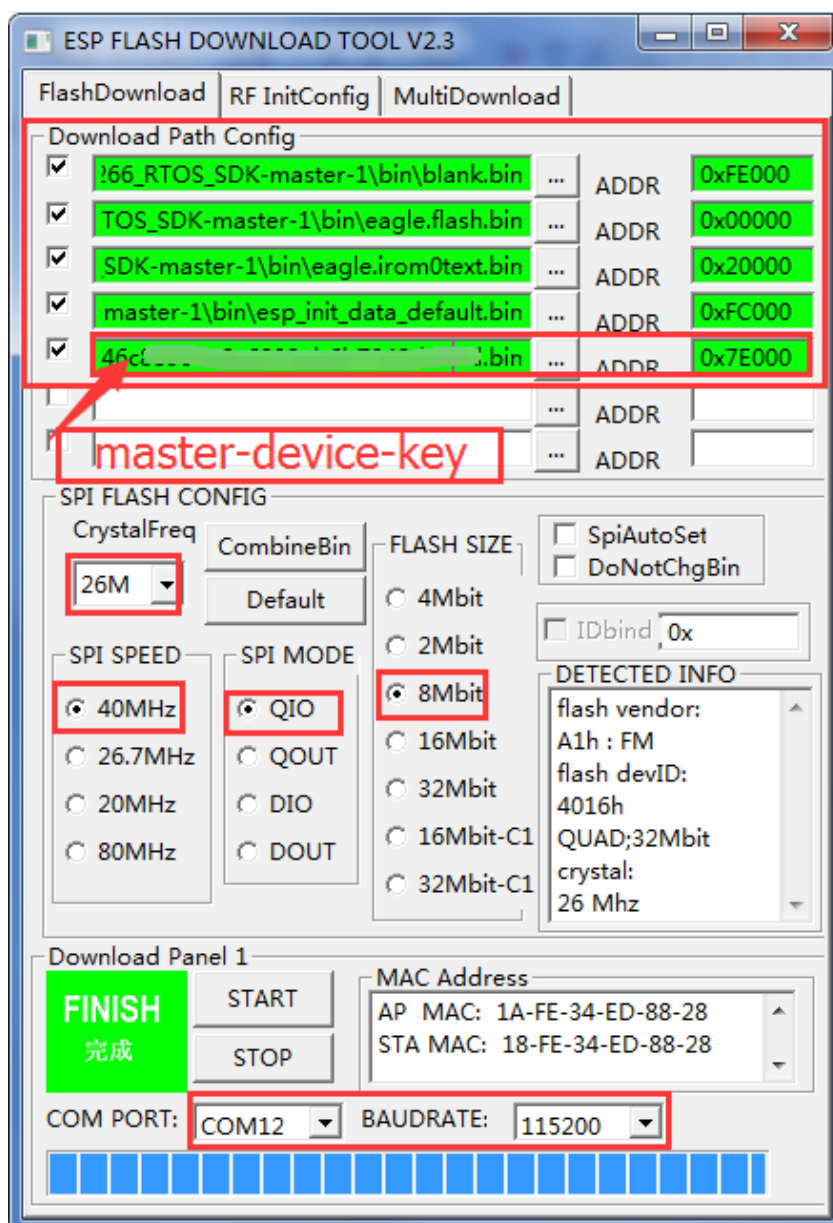
下载该文件后，将其放在之前的 BIN 目录“/ESP8266\_RTOS\_SDK-master/bin”下，方便下载。

## 5. 下载固件

使用官方下载工具：ESP FLASH DOWNLOAD TOOL，下载各个 BIN 文件到开发板。下载时需要特别注意，各个 BIN 文件对应的地址，如果错了的话，是不能正常工作的。

### 5.1 配置下载工具

根据下图配置，就可以了。

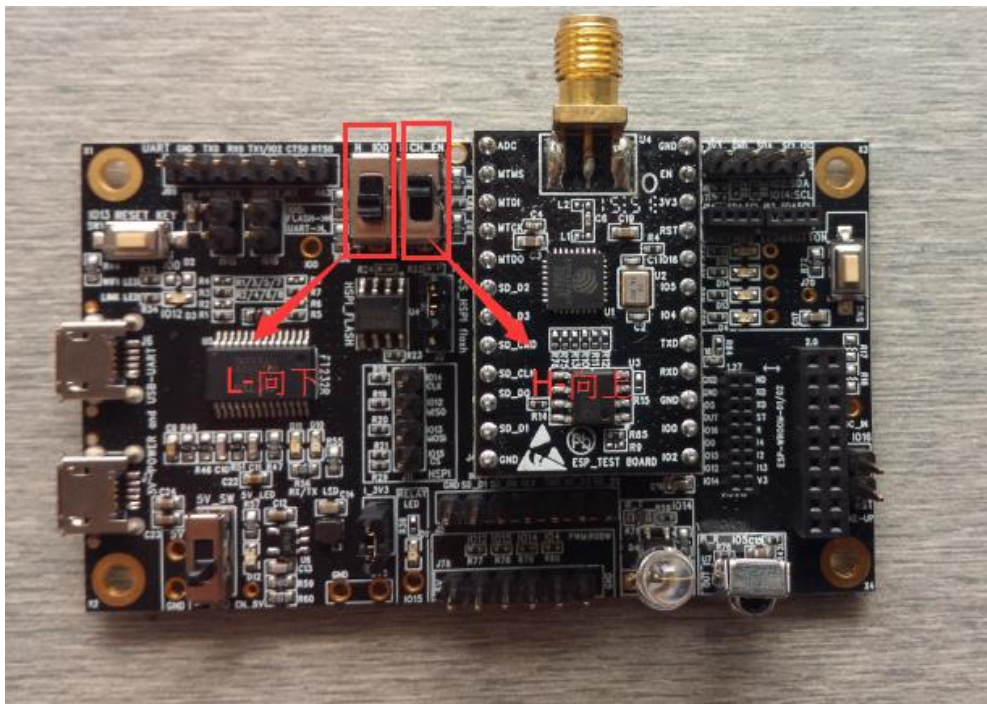


### 5.2 下载

下载前，需要将开发板置于下载模式，然后重新上电，才能正常下载。下载模式对应引脚的电平如下：

CH\_EN = H  
IO0 = L  
IO13 = H  
IO15 = L

我是使用的官方出的开发板 ESP-LAUNCHER，其只需要拨动两个拨码开关即可。



下载完成后，将板子切换到 Flash 启动模式,其对应引脚电平如下：

CH\_EN = H

I/O0 = H

I/O13 = H

I/O15 = L

暂时不上电，等待 APP 配置好后，再上电。

## 6. 手机 IOT APP

官方出的 IOT APP，目前只有安卓版的。

### 6.1 下载 IOT APP

IOT APP 的名字：IOT Espressif

下载地址：<http://apk.91.com/Soft/Android/com.espressif.iot.html>

使用版本：1.20

### 6.2 注册 IOT 账号，登录 IOT APP

我们在上面的获取 master-device-key.bin 时，已经注册过 IOT 账号了，直接使用该账号登录即可。

如果，不登录的话，是不能连接到云端，就只能使用本地模式。

本地模式：只有手机和设备在同一个局域网下，才可以通信，才可以使用 APP 控制设备。

云端模式：只要手机有网络，不管是不是同一局域网下，也不管你那里，登录后，你就可以远程控制设备了。

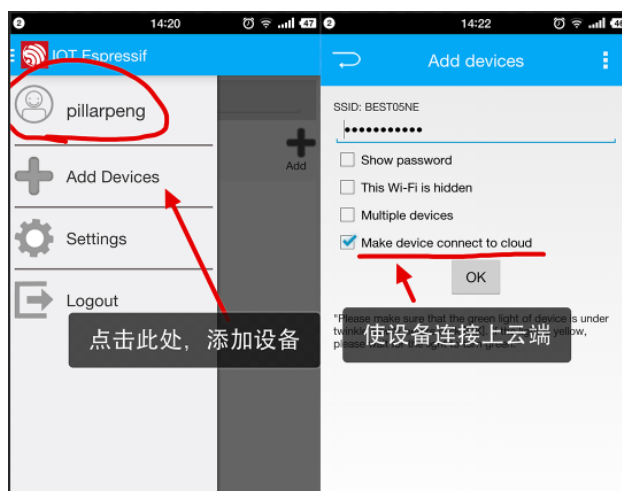
这里的云端服务器是：乐鑫官方的云服务器

## 7. 设备连接上云端

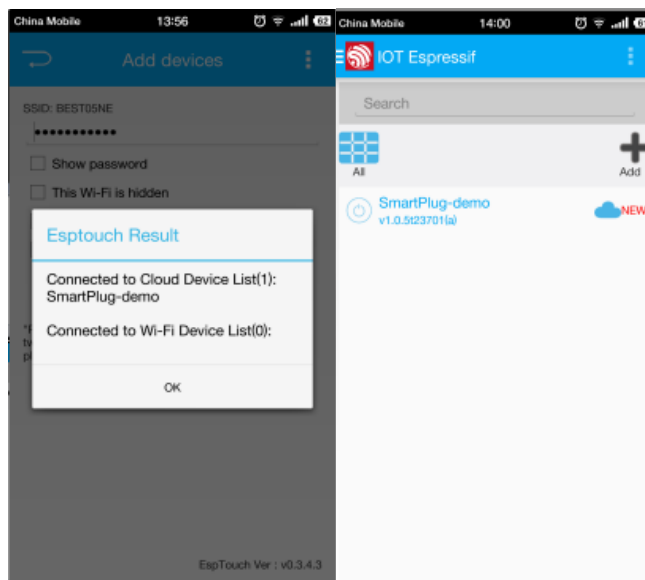
接下来使用 APP 将设备与乐鑫的云服务器连接起来，这样就可以远程控制了。

### 7.1 APP 配置设备

- 1) 一定得先打开路由器，并且是路由器链接上网络。（手机开热点也可以）
- 2) 将设备（ESP 开发板）设置为 Flash 启动模式，然后上电，直到蓝色的 link\_LED 由亮变为慢闪。说明已经扫描完周围 AP（路由器或者热点）的信息，等待链接。
- 3) 使用 APP，添加设备（记得登录先）



- 4) 点击 OK 后，蓝色的 link-LED 灯，会由慢闪编程快闪，直到常亮。这样就表示，已经添加好了设备。同时，手机 APP 上也会出现“已连接上云端设备：(1)”这样的提示。如下图所示：



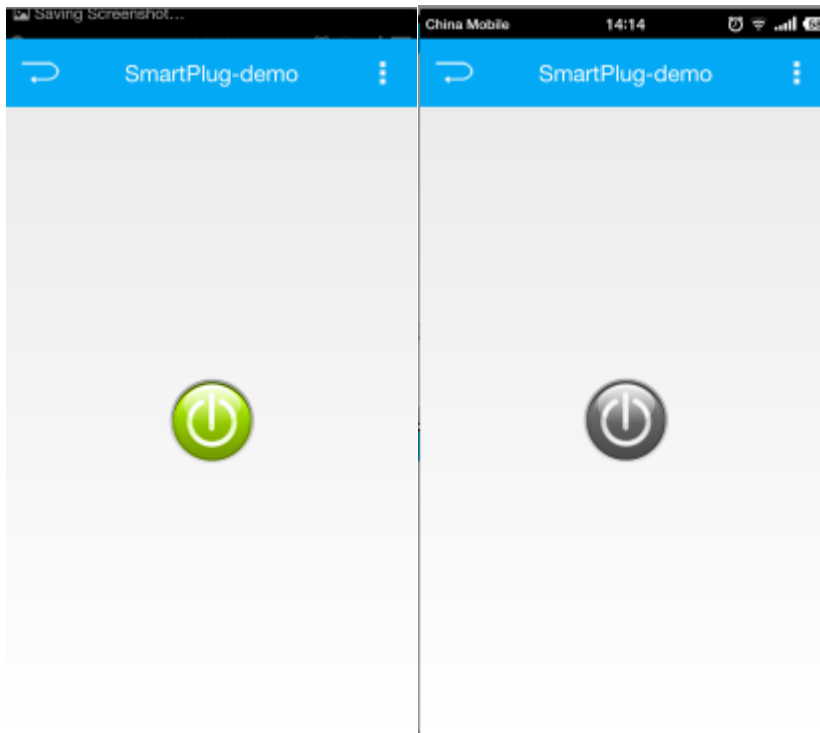
也有可能是“已连接上 Wi-Fi 设备：(1)”，这就表示设备没有连接上云端，只是本地模式。如果不在同一个局域网内，是不能控制设备的。造成这种的原因可能有以下几种：

- 路由器没有连接到外网
- Master-device-key.bin 文件中的 key 值不正确
- 烧写 Master-device-key.bin 文件时，地址弄错。

## 7.2 APP 控制设备

App 上连接的设备，就可以通过手机 APP 控制了。

下图是，开与关。



Ps:

1. 如有错误和不当之处，请不吝指出。
2. 如有不理解或者不清楚之处，可以留言或者 Email 我，如能解答，一定尽快回复。

Pillarpeng

2016.7.19 14:18

Pillarpeng@outlook.com