ESP8266EX 启动的提示信息与上下拉跳线的解释和问题分析

问：为什么在我的模块刚才还执行得好好的，但是重启后就执行不了了，启动信息如下，这代表什么意思？

ets Jan 8 2013,rst cause:4, boot mode:(1,3)
wdt reset

答：这是因为在看门狗复位的时候，可能继续沿用了上次采集到的 boot_mode 参数，而进入了 UART BOOTING 模式，而非正常的从 LASH 启动。

rst cause 的含义：后面的数字代表着本次重启的原因。

rst cause:1 表示上电复位，含 CHIP_EN/CHIP_PD 管脚造成的复位
rst cause:2 表示管脚复位，即通过 nRESET 管脚拉低造成的复位
rst cause:4 表示看门狗带来的复位

在本列中，本次重启的原因是看门狗复位。

boot mode 后面括号里数据的来源：boot mode 后面括号的数值，表次最近一次上电重启(含 CHIP_EN/CHIP_PD 重启)、或通过 nRESET 管脚拉低造成的复位时，芯片从一些管脚采集到的电平逻辑。

其对应关系如下：

逗号前的数值 = b(GPIO15 GPIO0 GPIO2)
逗号后的数值 = b(SD3 SD2 SD0)

在本例中，boot mode:(1,3)表示，最近一次上电重启(含 CHIP_EN/CHIP_PD 重启)、或通过 nRESET 管脚拉低造成的复位时，芯片从一些管脚采集到的电平逻辑对应关系为：

GPIO15 = 低电平逻辑 = 0
GPIO0 = 低电平逻辑 = 0
GPIO2 = 高电平逻辑 = 1 b001 = 1 = 逗号前的数值

SD3 = 低电平逻辑 = 0
SD2 = 高电平逻辑 = 1
SD0 = 高电平逻辑 = 1 b011 = 3 = 逗号后的数值

特别地，

1. 在看门狗复位时，芯片可能不会去重新采集这些管脚的电平，因此看到的数据仍然会是上一次上电重启或 reset 管脚复位时的数值。
2. 在上电重启或 nRESET 管脚复位时候，这个提示信息一般只在 74880bps 波特率（对应外部晶体频率为 26MHz）或 115200bps 对应外部晶体频率为 40MHz）波特率下才

可以看到；而在看门狗复位时，在当前设置的波特率下可以看到。

boot_mode 的数值和启动模式的对应关系如下：

0, x(!=2) Remapping
1, x(!=2) UART0 Booting 烧写程序时会用这种启动模式
2, x(!=2) Jump Booting
3, x(!=2) Flash Boot， 目前大多数模块的启动都用到这种从 FLASH 启动的方式
4, x(!=2) SDIO Low Speed V2 IO
5, x(!=2) SDIO High Speed V1 IO
6, x(!=2) SDIO Low Speed V1 IO
4-7, 2 UART1 Booting

常见的启动配置的解释和建议：

如果你希望进入烧写程序而需要进入 UART0 Booting 模式，那么在上电复位或通过 nRESET 管脚拉低造成的复位之前，你需要将

GPIO15 = 低电平逻辑 = 0 = 下拉
GPIO0 = 低电平逻辑 = 0 = 下拉
GPIO2 = 高电平逻辑 = 1 = 上拉 或悬空 **烧写启动**

而如果你希望在启动时，能进入正常的 FLASH 启动，那么在上电复位或通过 nRESET 管脚拉低造成的复位之前，你需要将

GPIO15 = 低电平逻辑 = 0 = 下拉
GPIO0 = 低电平逻辑 = 1 = 上拉 或悬空
GPIO2 = 高电平逻辑 = 1 = 上拉 或悬空 **正常地从 FLASH 启动**

悬空是因为 GPIO15、GPIO0 和 GPIO2 在上电复位或 nRESET 管脚复时都有内部弱上拉。因此，在这里，悬空实际就等于上拉。

但是有一点值得指出，SD3、SD2 和 SD0 管脚在上电复位或 nRESET 管脚复位时内部是没有弱上拉的。因此，在某些模块上，大家会看到 boot mode 后面括号里逗号后面的数值经常会不同，这可能是因为在这些模块的设计上，它们将这些管脚悬空了而出现了随机数。

类似地，nRESET 管脚也有内部上拉，而 CHIP_EN/CHIP_PD 管脚则没有内部上拉。

因此，在**实际的设计或连线**的时候，如果只用到了烧写启动/UART0 BOOTING 和正常的从 FLASH 启动两种启动模式，应该按照下面的方法连接或设计：

GPIO15 = 低电平逻辑 = 0 = 下拉， **必须下拉**
GPIO0 = 可控逻辑 = 0/1 = **可悬空，但须留出一个控制点，可下拉以便烧写**
GPIO2 = 高电平逻辑 = 1 = 上拉，**可悬空**
nRESET = 高电平逻辑 = 1 = **可悬空**，但最好留出一个可以手工复位的控制点

CH_EN/CH_PD = 1 = 上拉，**必须上拉**

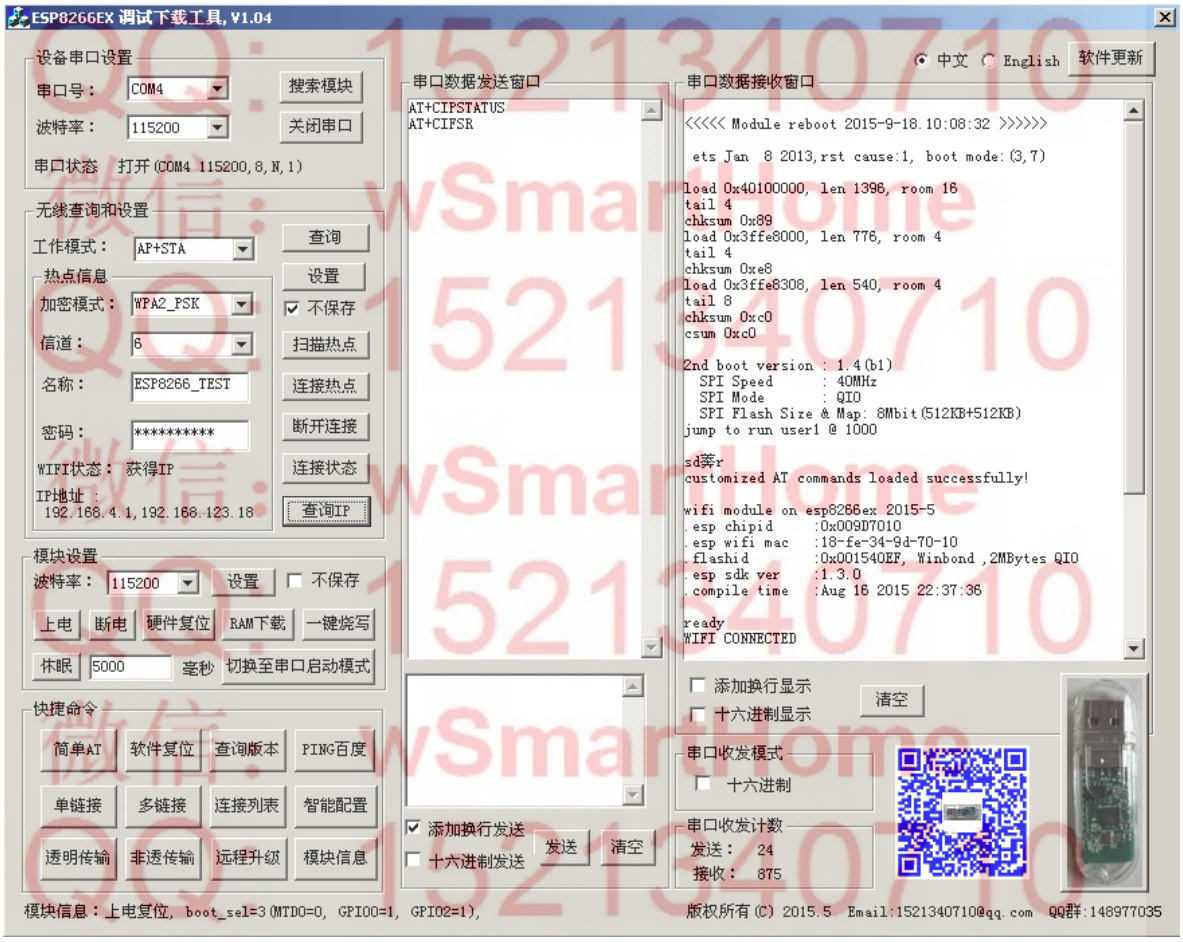
SD3 SD2 和 SD0 里面须有一个是上拉或下拉的，以确保其数值不等于 b010

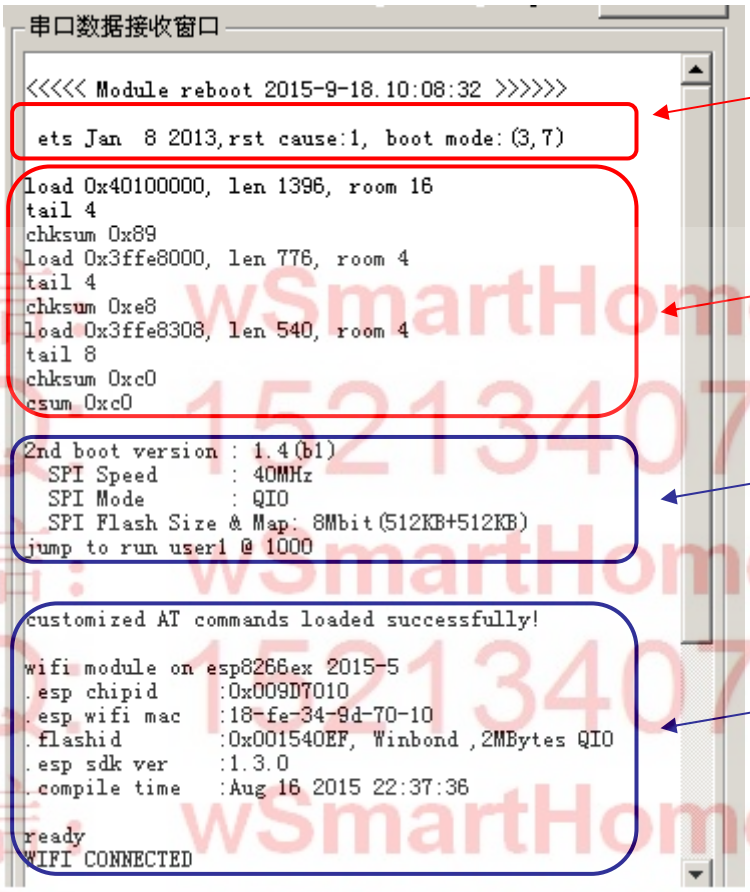
通过 boot_mode 的启动模式来做第一步的问题定位

“ets Jan 8 2013,rst cause:4, boot mode:(1,3)”启动提示信息来源芯片内部，与外部是否焊接了 FLASH 无关。一般说来，只要 8266 主芯片工作正常（包括供电正常、外部晶体正常）、串口连接正确电平逻辑兼容（主要是 8266 模块的 TXD 串口线），就可以打印出该信息。所以，一般可以用这种方法来初步地缩小定位一些第一步遇到的问题。

查看启动配置信息的串口输出，需要设置正确的串口波特率。如果外部晶体的频率是 26MHz，那么这里的波特率是 74880bps。如果外部晶体的频率是 40MHz，那么这里的串口波特率是 115200bps。目前市面上大多数模块都使用 26MHz 的晶体，**可以去这个地址下载软件支持 74880bps 的串口波特率**

ESP8266 调试下载工具下载地址 http://pan.baidu.com/s/1pJ2xTJd





回到最初问题的解释

1. 这里的 rst cause:4, boot mode:(1,3) 表示在看门狗复位的时候，进入了 UART0 Booting，非正常地从 FLASH 启动了

2. 但是刚才执行很正常，现在却不行了，一种可能的原因和解决办法：

   之前的上电/复位管脚重启的时候，可能是去烧写程序，在烧写完毕后，一些烧写软件会通过其后的烧写重启参数而直接跳到程序里开始执行，因此这个时候，无论是是否释放 GPIO0 或继续拉低着 GPIO0，都可以进入正常程序执行。

   但是一旦在程序里，你执行了看门狗复位，或任何其他可能造成看门狗复位的事件，你都可能出现本问题所示的现象，即使在看门狗复位之前，你可能已经将 GPIO 拉高或释放悬空。因为看门狗复位不会去重新读取外部管脚而直接继续沿用上一次的 boot_mode。

3. 解决办法：
   这个时候，你只需要将 GPIO0 拉高或释放悬空，重新对模块通过上电或通过复位管脚进行一次复位，就不再出现这里的问题了。

欢迎大家试用我们的一键烧写的 **USBWIFI ESP8266** 模块、排针版 **ESP8266** 模块、调试下载软件工具，特别方便于二次开发和程序烧写，有空去踩踩吧。在下面的链接里有相关的产品说明书简介（详细说明书可以和我们联系索取）。

ESP8266EX 模块，USB 版

http://item.taobao.com/item.htm?id=522160552246

ESP8266EX 模块，插针版

http://item.taobao.com/item.htm?id=522158628730