

# ESP8266 烧录 MicroPython 固件明细教程

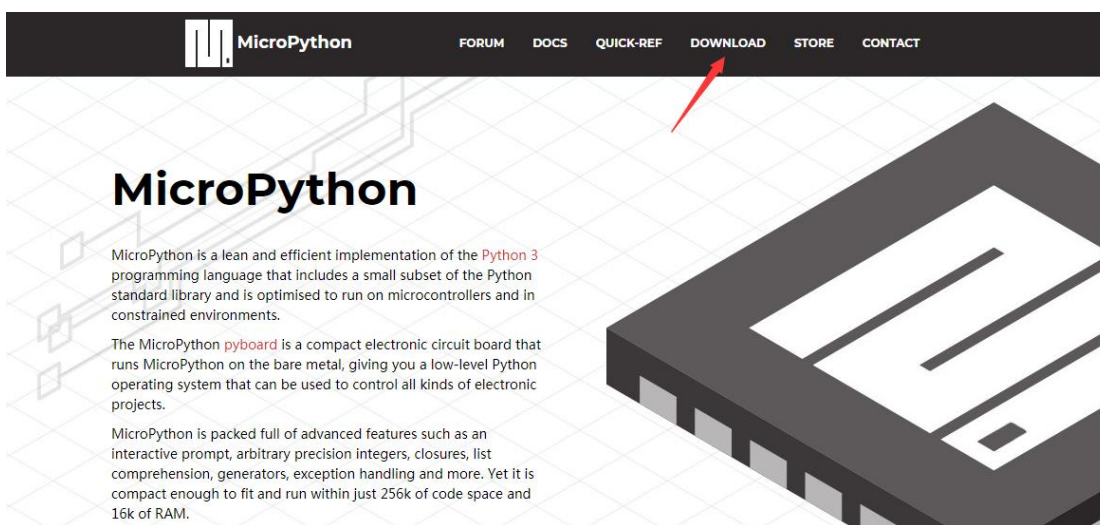
广东职业技术学院 / [www.xmf393.com](http://www.xmf393.com) 欧浩源 [ohy3686@qq.com](mailto:ohy3686@qq.com) 2021-02-06

本明细教程的主要内容有：面向 ESP8266 的 MicroPython 固件获取，利用固件烧录工具 flash\_download\_tool 进行固件烧录，以及烧录过程中常见问题的解决。

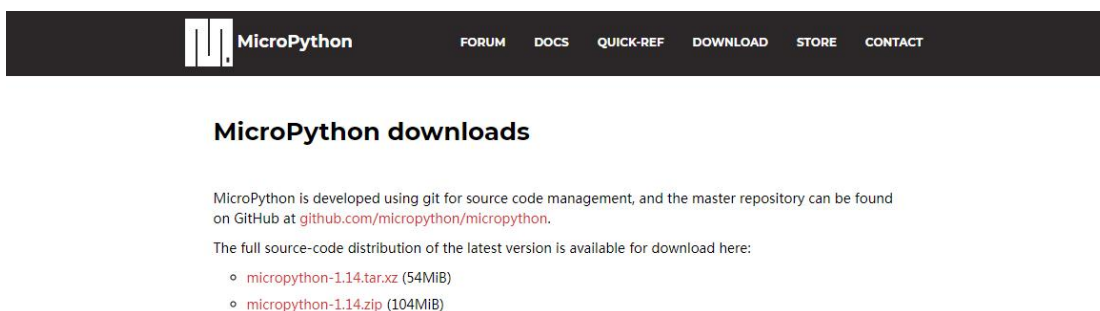
开发常用的资源及相关软件已整理成工具包，共享到网上，有需要可自行下载。

工具包下载地址：<https://www.xmf393.com/2019/09/17/xmf09f/>。

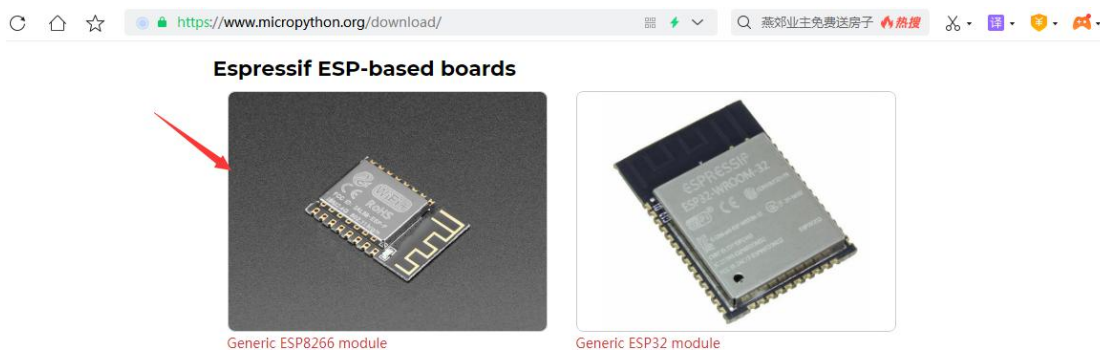
【01】登录小蜜蜂笔记网 ([www.xmf393.com](http://www.xmf393.com)) 或 MicroPython 官网下载 ESP8266 的烧写固件，本人建议登录官网 [www.micropython.org](http://www.micropython.org)，下载最新的版本。



【02】在 MicroPython 官网首页上，点击“DOWNLOAD”进行下载页面。



【03】在下载页面中，向下找到面向 ESP8266 的版本，点击进入。



【04】在 ESP8266 的下载页面中，找到你合适的版本点，点击下载。

Stable firmware, 2M or more of flash

The following files are stable firmware for the ESP8266. Program your board using the esptool.py program as described in the tutorial.

Note: v1.13 of the firmware has a new flash filesystem layout, and uses littlefs as the filesystem by default. When upgrading from older firmware please backup your files first, and either erase all flash before upgrading, or after upgrading execute `uos.VfsLfs2.mkfs(bdev)`. Also note that v1.12 and earlier will work on modules with 1M or more of flash, while v1.13 requires 2M or more.

- esp8266-20210203-unstable-v1.14.bin (elf, map) (latest)
- esp8266-20210202-v1.14.bin (elf, map)
- esp8266-20200911-v1.13.bin (elf, map)
- esp8266-20191220-v1.12.bin (elf, map)

【05】下载到的是 bin 文件，可以进行直接烧录。

新建下载任务

网址: <https://www.micropython.org/resources/firmware/esp8266>

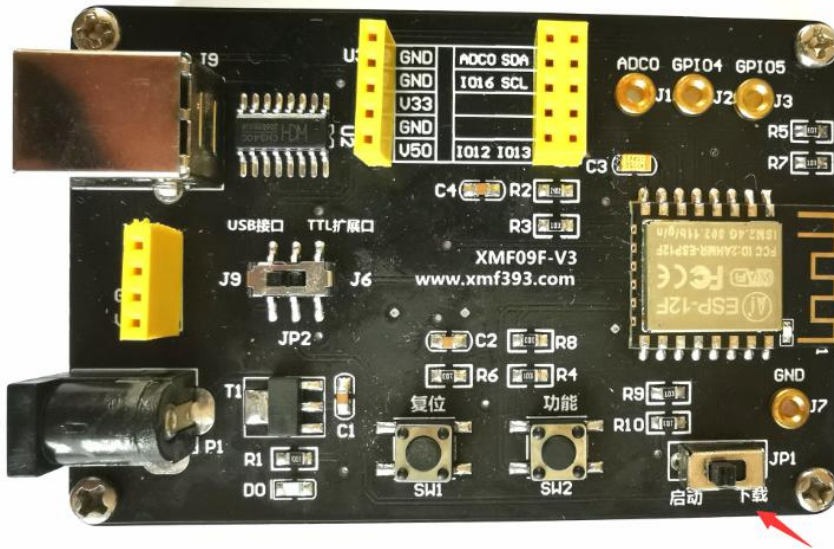
名称: esp8266-20210202-v1.14.bin 617.25 KB

下载到: C:\Users\Administrator\Desktop 剩: 8.49 GB 浏览

使用迅雷下载 直接打开 下载 取消

【06】将 ESP8266 的 **GPI00** 引脚接地，设置成下载模式。

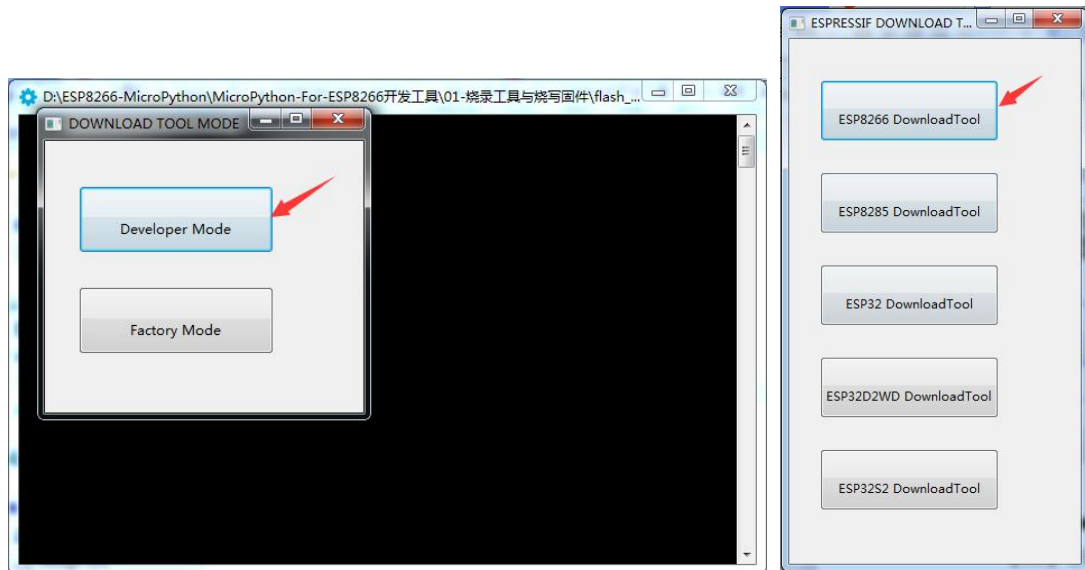
如果您使用的是小蜜蜂出品的 WIFI 开发板 **XMF09F**，将 **JP1** 拨码开关，拨向右边的“**下载**”，将 **JP2** 拨码开关，拨向左边的“**USB 接口**”，再用 USB 线将该开发板接到电脑上。



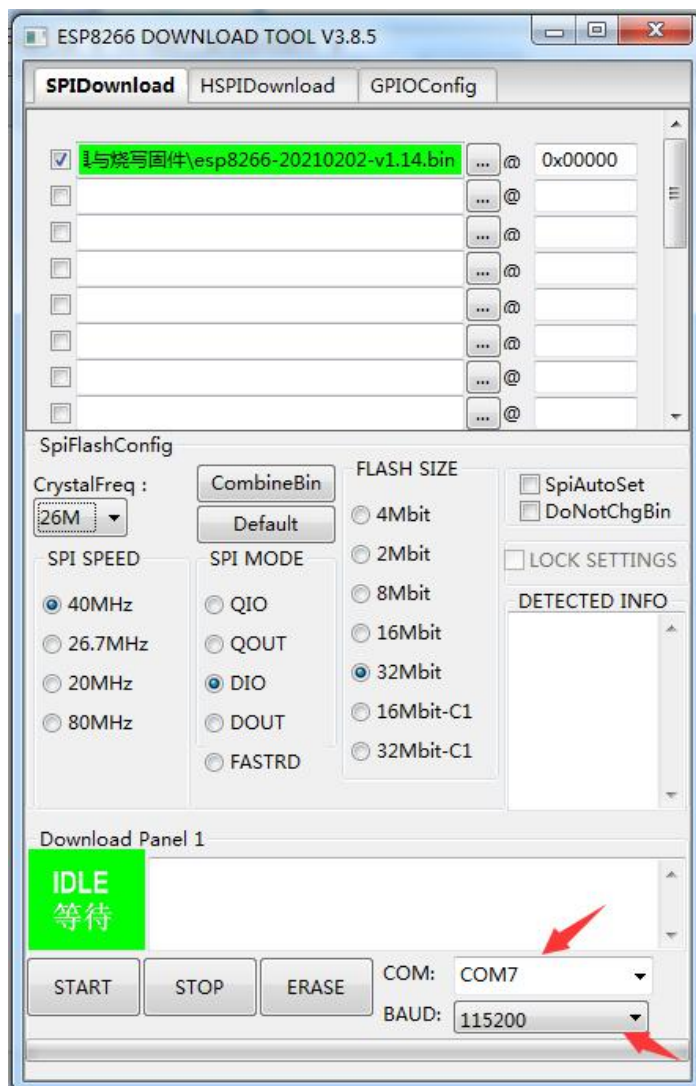
【07】打开烧录工具所在的文件夹“**flash\_download\_tool\_v3.8.5**”，鼠标点击可执行文件“**flash\_download\_tool\_v3.8.5.exe**”

名称	修改日期	类型	大小
bin	2020/4/25 18:57	文件夹	
combine	2021/2/5 17:22	文件夹	
configure	2021/2/5 17:22	文件夹	
dl_temp	2021/2/5 17:22	文件夹	
doc	2021/2/5 17:22	文件夹	
init_data	2021/2/5 17:22	文件夹	
logs	2021/2/5 17:22	文件夹	
RESOURCE	2021/2/5 17:22	文件夹	
flash_download_tool_v3.8.5.exe	2020/5/28 14:36	应用程序	13,834 KB
Readme.pdf	2017/9/22 19:41	WPS PDF 文档	455 KB

【08】在弹出的窗口中，鼠标点击“Developer Mode”按钮，并在弹出的窗口中，鼠标点击“ESP8266 DownloadTool”按钮。

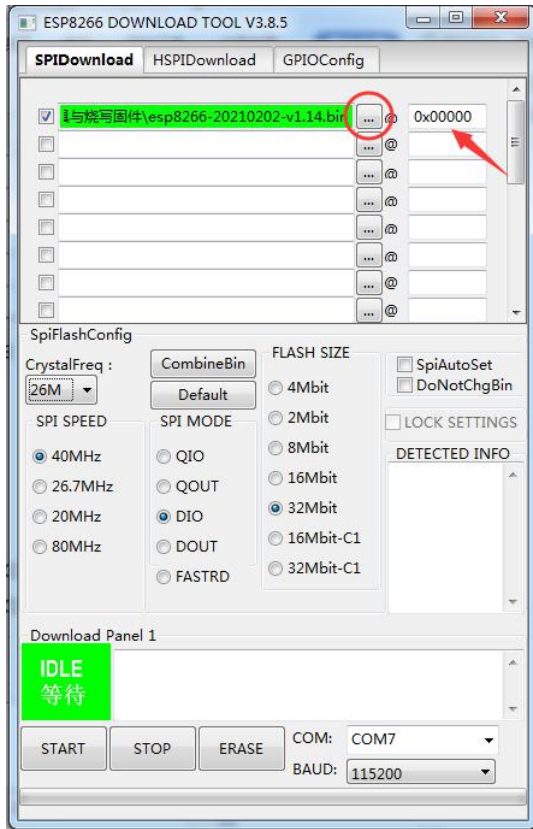


【09】此时，打开了烧录工具的工作界面。在“SPIDownload”界面的右下角，将COM口正确设置为ESP8266或开发板所接入的端口号，波特率设置为115200，其他设置默认即可。

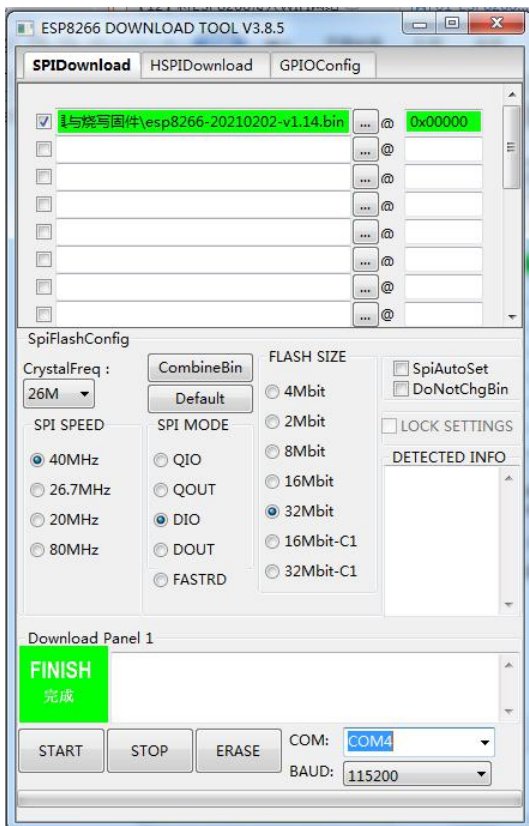
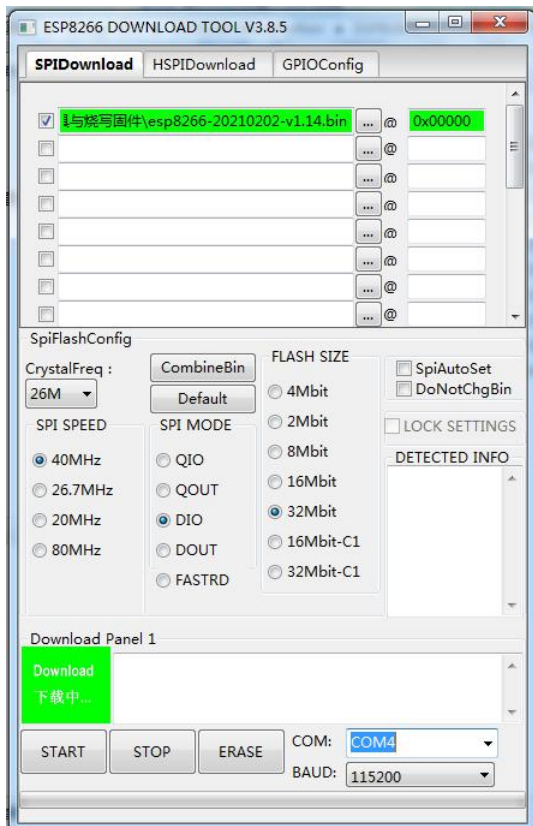


【10】在烧录之前，需要导入固件所在的路径，地址必须为 **0x00000**。

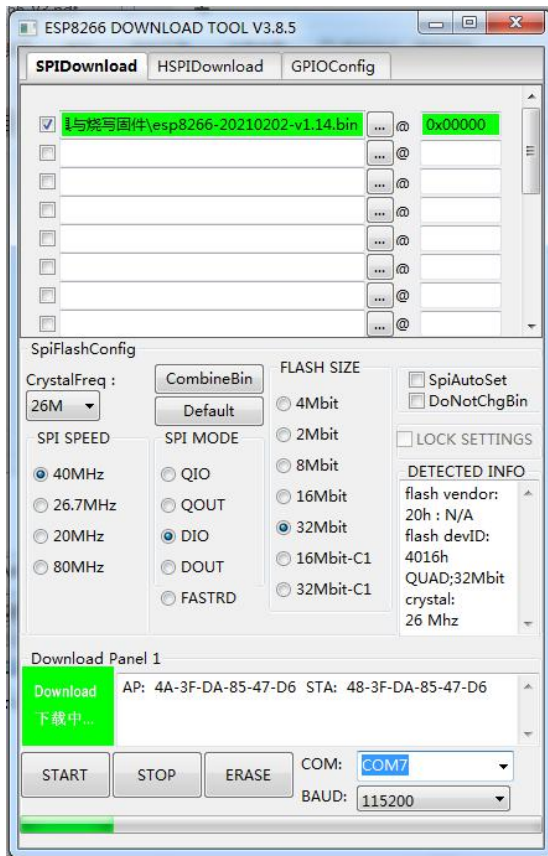
**注意：**在本人的使用过程中，发现下图**圆圈处**的文件浏览按钮不能使用。出现这种情况，可以手动将固件所在的完整路径复制到编辑框中。导入路径正确，该编辑框为**绿色**。



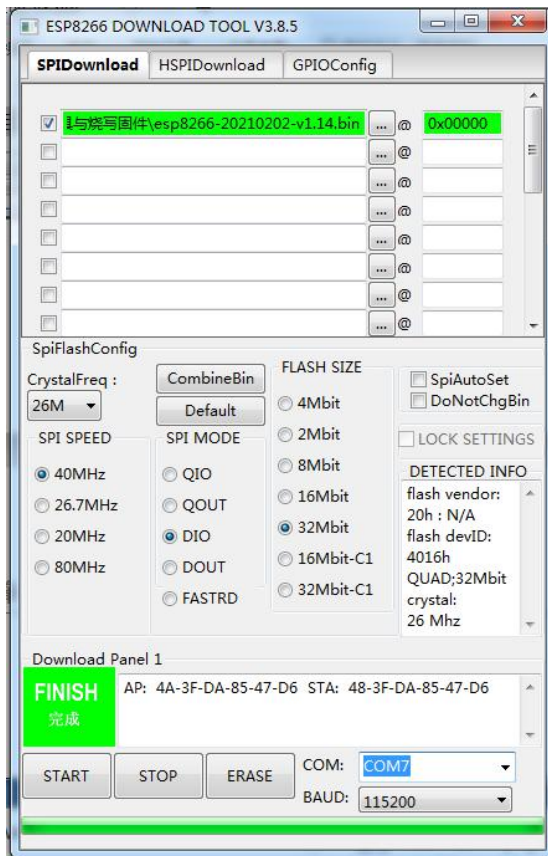
【11】一切准备就绪，点击烧录工具左下角的“**ERASE**”按钮，擦除 ESP8266 上的内容。



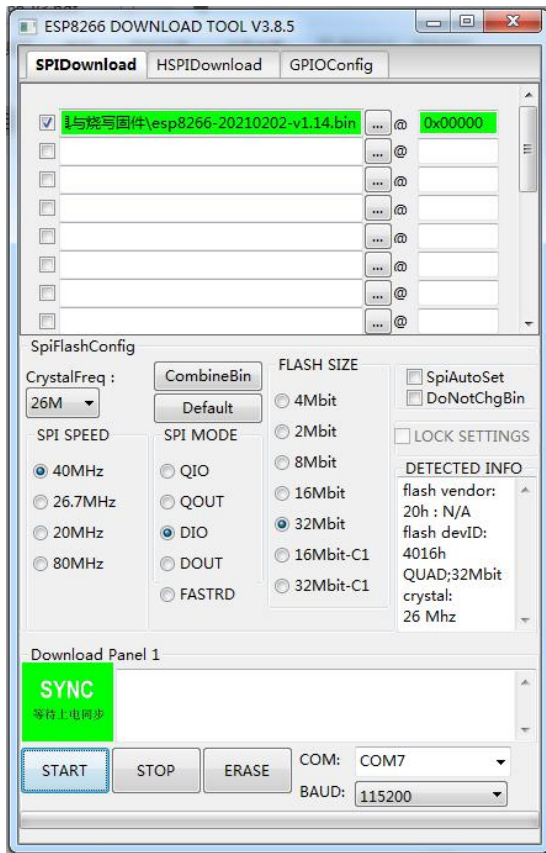
【11】芯片擦除完成之后，点击烧录工具左下角的“START”按钮，即可开始烧录。



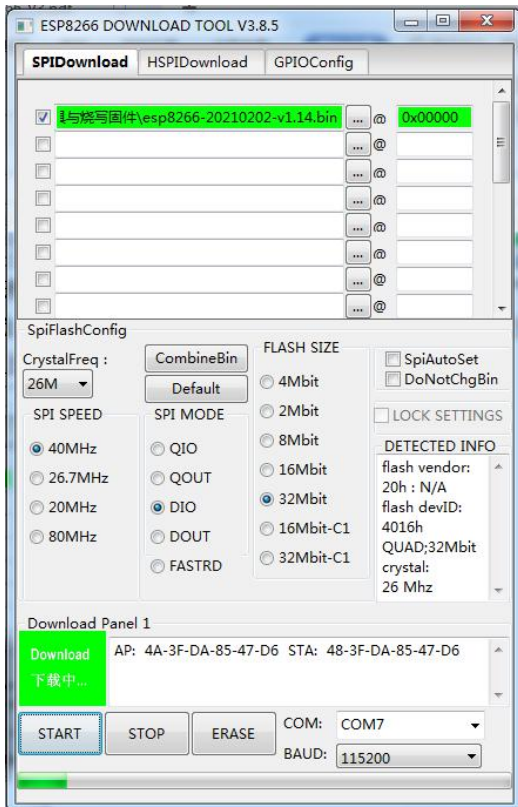
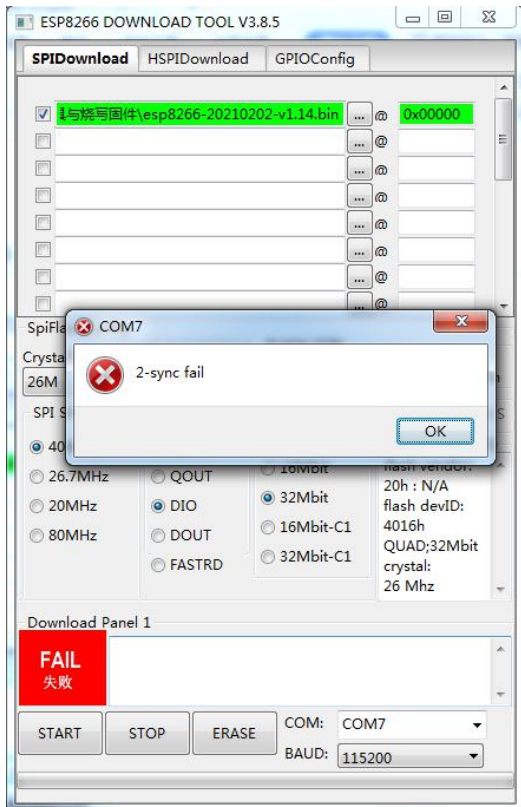
【12】等待几分钟，固件烧录就完成了。



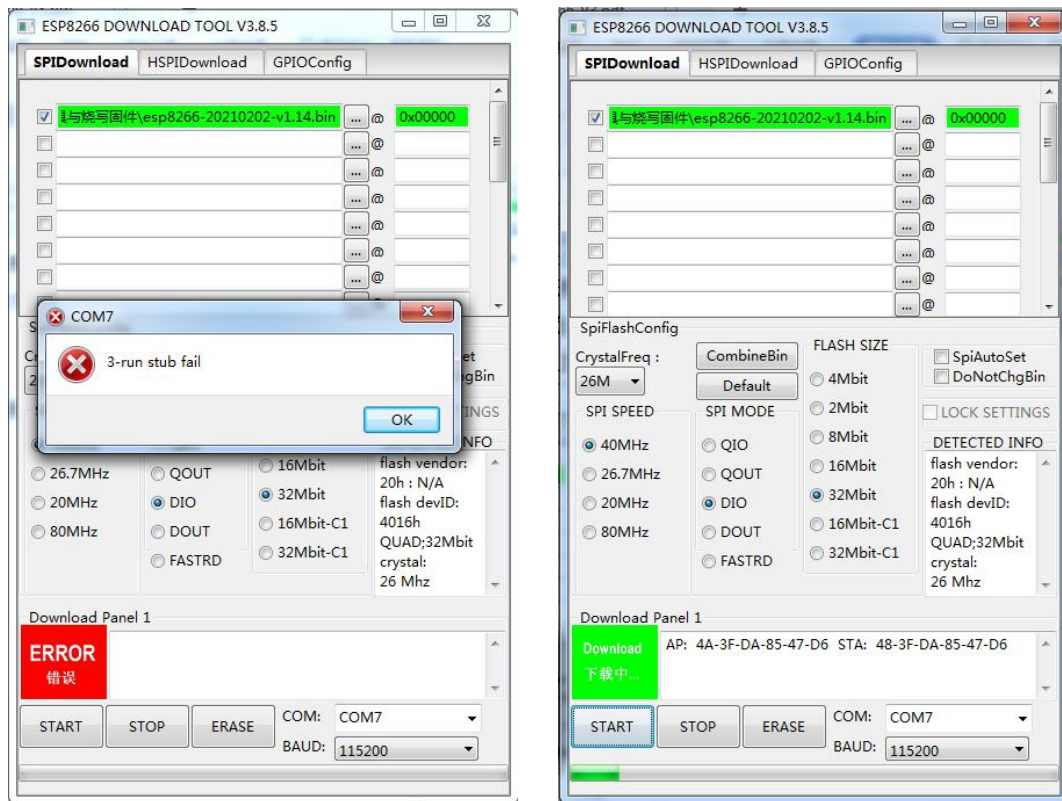
**【特别注意】**：有时候，在点击“**START**”按钮后，一直处于“等待上电同步”状态。这时有2种解决办法：重新上电或手动硬复位。



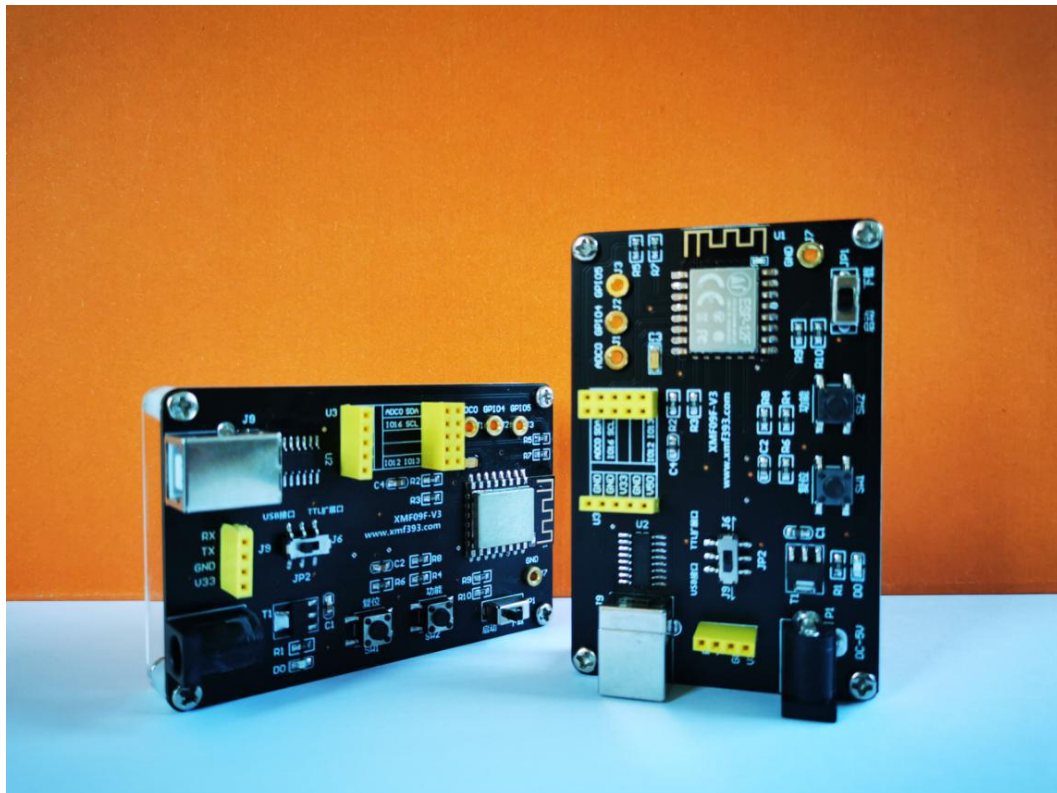
**【解决方法一】**先把 USB 线从电脑断开，烧录界面弹出同步错误，点击“OK”，然后重新接入 USB 线上电，点击“**START**”按钮即可正常烧录。



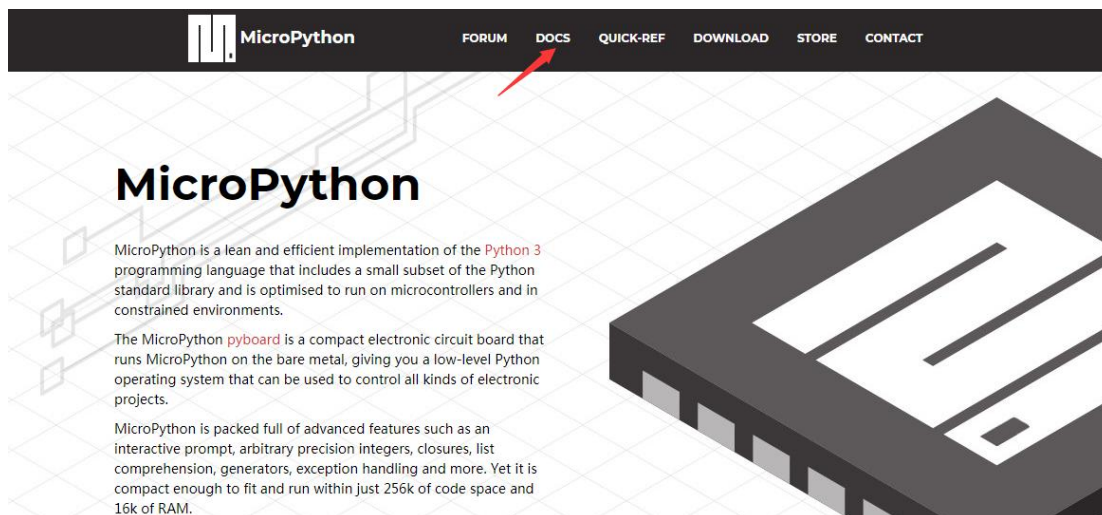
**【解决方法二】**如果您使用的是 XMF09F 开发模块，或者是带有复位按键的开发板。在“等待上电同步”状态时，反复多按几次复位键，信号同步后，便开始正常烧录。在这个过程中，可能会出现错误，点击“OK”，重复上述过程，直至正常烧录即可。



**【友情推荐】**：小蜜蜂出品的 XMF09F 开发套件  
详见：<https://www.xmf393.com/2019/09/17/xmf09f/>



**【温馨提示】**：在 MicroPython 的官网中，有一个“DOC”的菜单，点击进去，可以找到最权威的参考文档，是你应用开发过程中必不可少的工具书。



进入“DOC”页面后，在右侧导航栏中，选择“Quick reference for the ESP8266”便可查阅 ESP8266 开发中涉及的 MicroPython 相关知识，最大的缺点就是它是 English 的。

**Pins and GPIO**

Use the `machine.Pin` class:

```
from machine import Pin

p0 = Pin(0, Pin.OUT) # create output pin on GPIO0
p0.on()              # set pin to "on" (high) level
p0.off()             # set pin to "off" (low) level
p0.value(1)         # set pin to on/high

p2 = Pin(2, Pin.IN)  # create input pin on GPIO2
print(p2.value())   # get value, 0 or 1

p4 = Pin(4, Pin.IN, Pin.PULL_UP) # enable internal pull-up resistor
p5 = Pin(5, Pin.OUT, value=1) # set pin high on creation
```

Available pins are: 0, 1, 2, 3, 4, 5, 12, 13, 14, 15, 16, which correspond to the actual GPIO pin numbers of ESP8266 chip. Note that many end-user boards use their own adhoc pin numbering (marked e.g. D0, D1, ...). As MicroPython supports different boards and modules, physical pin numbering was chosen as the lowest common denominator. For mapping between board logical pins and physical chip pins, consult your board documentation.

Note that Pin(1) and Pin(3) are REPL UART TX and RX respectively. Also note that Pin(16) is a special pin (used for wakeup from deepsleep mode) and may be not available for use with higher-level classes like `Neopixel`.

### UART (serial bus)