Jump Test Project of ESP8266 2nd Boot (V1.6+)

Version 1.0

About This Guide

The document is structured as follows.

Chapter	Title	Content
Chapter 1	Overview	Introduces the secondary boot V1.6 + (***SDK Non-OS 2.0.0*** and later versions) of ESP8266.
Chapter 2	Configuration method of enabling	Introduces configuration method of enabling.

Release Notes

Date	Version	Release notes
2016.09	V1.0	Initial release.

Table of Contents

1. Overview

The secondary boot V1.6 + (***SDK Non-OS 2.0.0*** and later versions) of ESP8266 supports jump test. That is to say, boot can trigger jump test by checking on the status of a configured and enabled GPIO at the moment (within 100 ms of power up) when the system is powered on. When a configured GPIO is pulled to a low level, system will jump to a specified test bin file and run. This bin file can be downloaded to flash memory together with the firmware that needs to be downloaded before production. In this way, users can save time for downloading test bin during production.

> ⚠ ***Notice:***
>
> - *Applicable to boot V1.6 + (for both **SDK Non-OS SDK** and **RTOS SDK**, unrelated to SDK version).*
> - *The bin file for jump test should be a specified test file of Espressif. Users cannot download other binaries which are for secondary development to this bin file.*

2. Configuration Method of Enabling

2.1. Configuring GPIO

In this project, the GPIO configuration information for enabling jump test is located in ***esp_init_data_default.bin***. ***esp_init_data_default.bin*** is 128 bytes in size, in which ***byte[119]*** defines which GPIO port controls jumping on boot. ***byte[119]*** is 0x00 by default, which disables the jump test. If configuration is in accordance with the corresponding relations shown below, boot BIN enables jump test by reading on the status of the specified GPIOs.

The corresponding relations are:

0xA5 ———> GPIO5

0xAC ———> GPIO12

0xAD ———> GPIO13

0xAE ———> GPIO14

> ⚠ ***Notice:***
> *The GPIO state is evaluated only at the moment (within 100 ms of power up) when the system is powered on. When system boots up, the state of this GPIO is not evaluated, thus making it available to the user application firmware.*

Example:

As the figure shows, users can modify ***byte[119]*** to 0xAC, which enables jump test when GPIO12 is pulled to a low level on power up.

```
00000000 05 00 04 02 05 05 05 02 05 00 04 05 05 04 05 05 ................
00000010 04 fe fd ff f0 f0 f0 e0 e0 e0 e1 0a ff ff f8 00 .    癈疣嚕?
00000020 f8 f8 52 4e 4a 44 40 38 00 00 01 01 02 03 04 05   RNJD@8........
00000030 01 00 00 00 00 00 02 00 00 00 00 00 00 00 00 00 ................
00000040 e1 0a 00 00 00 00 00 00 00 00 01 93 43 00 00 00 ?.........擘...[
00000050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000070 03 00 02 00 00 00 00 ac 00 00 00 00 00 00 00 00 .......?.......[
```

2.2. Configuration of the Address of Test Bin File

Test bin file should be located in ***blank.bin*** area. Users need to use a script named ***gen_test_blank.py*** and input the address where bin file jumps and runs in flash. ***gen_test_blank.py*** can generate a specified ***test_blank.bin***. When downloading flash firmware, users should burn ***test_blank.bin*** to the initial blank area according to flash map in the document *ESP8266 SDK Getting Started Guide*.

Example:

Generate ***test_blank.bin***,

Run ***python gen_test_blank.py***, the system will ask users to input the address where bin file locates.

```
[genmisc@Ubuntu bin]$python gen_test_blank.py
Enter you test bin addr(eg. 0x101000):
```

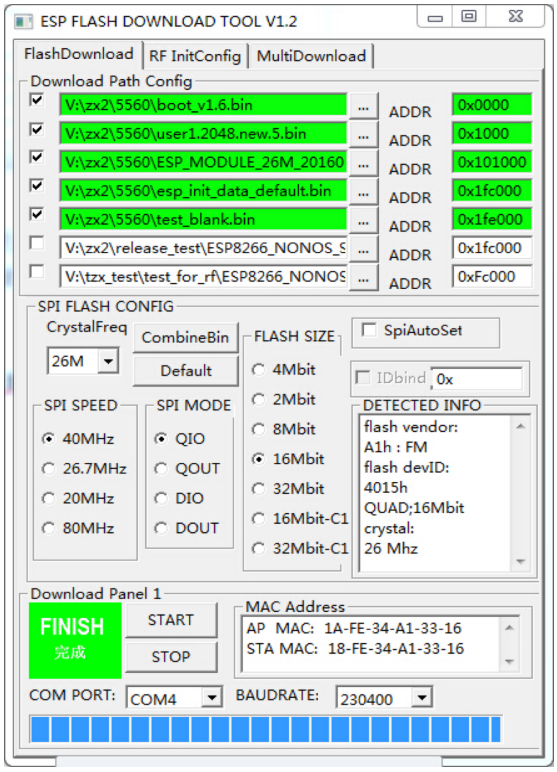
Users should input the actual bin file address for downloading according to the used flash size (please consult the corresponding flash memory map, avoiding conflict to SDK flash map). ***test_blank.bin*** will be generated after running ***gen_test_blank.py***.

2.3. Downloading Application Firmware

The figure below shows how to configure the official flash download tool to download application firmware. If ***ESP_MODULE_26M_20160520.bin*** is a test bin file, it should be downloaded in the address 0x101000 as flash map in the example below does.

When download is finished and the system runs normally, boot will jump to the address 0x101000 to run test bin file if GPIO12 is pulled to a low level on system power-up. Users can accomplish testing by this method.

If GPIO12 is not pulled to a low level on system power-up, system will jump to the address 0x1000 to run user application firmware.



Espressif IOT Team

www.espressif.com