Jump Test of ESP8266's Secondary Bootloader (V1.6+)

Version 2.0

About This Guide

The document is structured as follows:

Chapter	Title	Content
Chapter 1	Overview	Introduces ESP8266's Secondary bootloader V1.6+ which supports ***SDK Non-OS 2.0.0*** and later versions.
Chapter 2	Jump Test Configuration	Presents the method of enabling GPIOs, configuring BIN files and setting up the Flash download tool for the jump test mode.

Release Notes

Date	Version	Release notes
2016.09	V1.0	Initial release.
2017.05	V2.0	Major revisions.

Table of Contents

1. Overview

ESP8266's Secondary Bootloader V1.6+ (suited for the ***SDK Non-OS 2.0.0*** and later versions) supports the so-called jump test mode, which means that the bootloader can determine whether to trigger a jump test or not by checking on the status of an enabled GPIO when the system is powered on, that is, within 100 ms of power-up. When the enabled GPIO is pulled to a low level, the system will jump to a specified test bin file and run it; and when the enabled GPIO is not pulled to a low level, the system will run user-application firmware. Here, the enabled GPIO can be configured by users. Also, the test bin file can be downloaded to the flash memory together with the firmware that needs to be downloaded before the SMT production. In this way, users can save the time they would need for downloading this test bin file during the test that comes after the SMT production.

> ⚠ ***Notice:***
>
> - *This document is applicable to ESP8266's Secondary Bootloader V1.6+ (supports both the **Non-OS SDK** and **RTOS SDK**).*
> - *The bin file for the jump test should be a specified test file provided by Espressif, while any secondary development of this bin file by users is not supported in this case.*

2. Jump Test Configuration

2.1. Configuring the GPIO that Enables the Jump Test Mode

Users can choose the GPIO that enables the jump test mode by configuring ***byte[119]*** of ***esp_init_data_default.bin***, which is a 128-bit file. By default, ***byte[119]*** is initialized to `0x00`, which disables the jump test mode. However, if ***byte[119]*** is configured to `0xA5`, `0xAC`, `0xAD` or `0xAE`, then the bootloader will check on the status of GPIO5, GPIO12, GPIO13 and GPIO14, respectively, to determine if a jump test should be initiated.

The above-mentioned correlations are:

`0xA5` ———> GPIO5

`0xAC` ———> GPIO12

`0xAD` ———> GPIO13

`0xAE` ———> GPIO14

> ⚠ ***Notice:***
>
> - ***Byte[119]*** *can only be configured to* `0x00`, `0xA5`, `0xAC`, `0xAD` *or* `0xAE`, *otherwise malfunctions may occur.*
> - *The enabled GPIO is only checked when the system is powered on, that is, within 100 ms of power-up. This makes it available for user-application firmware afterwards.*

Example:

As the figure below shows, users can modify ***byte[119]*** to `0xAC`. Subsequently, upon power-up, the Bootloader will check on the status of GPIO12 to determine whether a jump test will be triggered or not.

```
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...C
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 .......?.......C
```

2.2. Configuring the Address of the Configuration Bin File

Users can configure the ***test_blank.bin*** file in the following way:

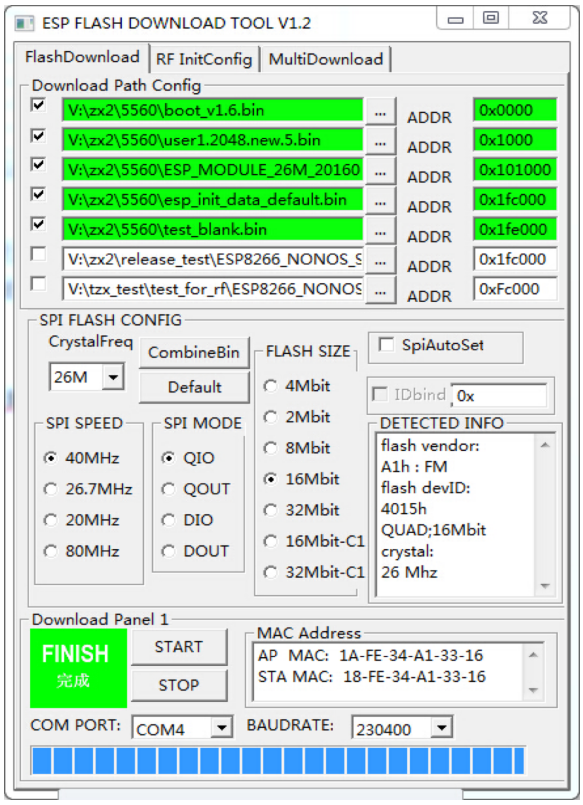
- Users should run ***python gen_test_blank.py*** and input the address of the flash memory that is reserved for the test bin file, as requested. This step is shown in the following screenshot:

- Then, the ***python gen_test_blank.py*** will generate a specified ***test_blank.bin*** in accordance with the user input.
- Subsequently, users should burn the address of the generated ***test_blank.bin*** to the initial blank area, i.e. the ***blank.bin*** area.

> ⚠ ***Notice:***
>
> *In an effort to avoid any discrepancies with the SDK flash map, users should consult the corresponding flash memory map, when allocating the address of the test bin file.*

2.3. Configuring the Flash Download Tool

Users can refer to the figure below to configure the Flash Download Tool (To download our latest tool, please click *here*):



where,

- ***ESP_MODULE_26M_20160520.bin*** is a test bin file and its address is `0x101000`;
- ***user1.2048.new.5.bin*** is a user application firmware and its address is `0x1000`;
- ***esp_init_data_default.bin*** is an initialization bin file;
- ***test_blank.bin*** is the configuration bin file, which is generated by script.

When downloading is finished and the system runs normally, the bootloader will check on the status of the enabled GPIO upon power-up. In this case:

- If the enabled GPIO is pulled to a low level, the system will jump to the address `0x101000` to run the test bin file;
- If the enabled GPIO is not pulled to a low level, the system will jump to the address `0x1000` to run user-application firmware.

Espressif IOT Team

www.espressif.com