How to download AT commands firmware to ESP8266 (ESP12) board

This loading procedure has been carried out for the Adafruit Huzzah board, although these instructions should theoretically be valid for the ESP-12 board in general. The firmware installed for AT-command interaction with the device is the ESP8266\_NONOS\_SDK, provided by the manufacturer (Espressif). In this case, I have installed the version 2.2.0 (AT version 1.6.0.0).

# Wiring

The wiring to be used is shown in the next image (source: <http://flower-platform.com/2015/12/16/esp8266-with-at-commands-flashingupdating-the-firmware-step-by-step/>)

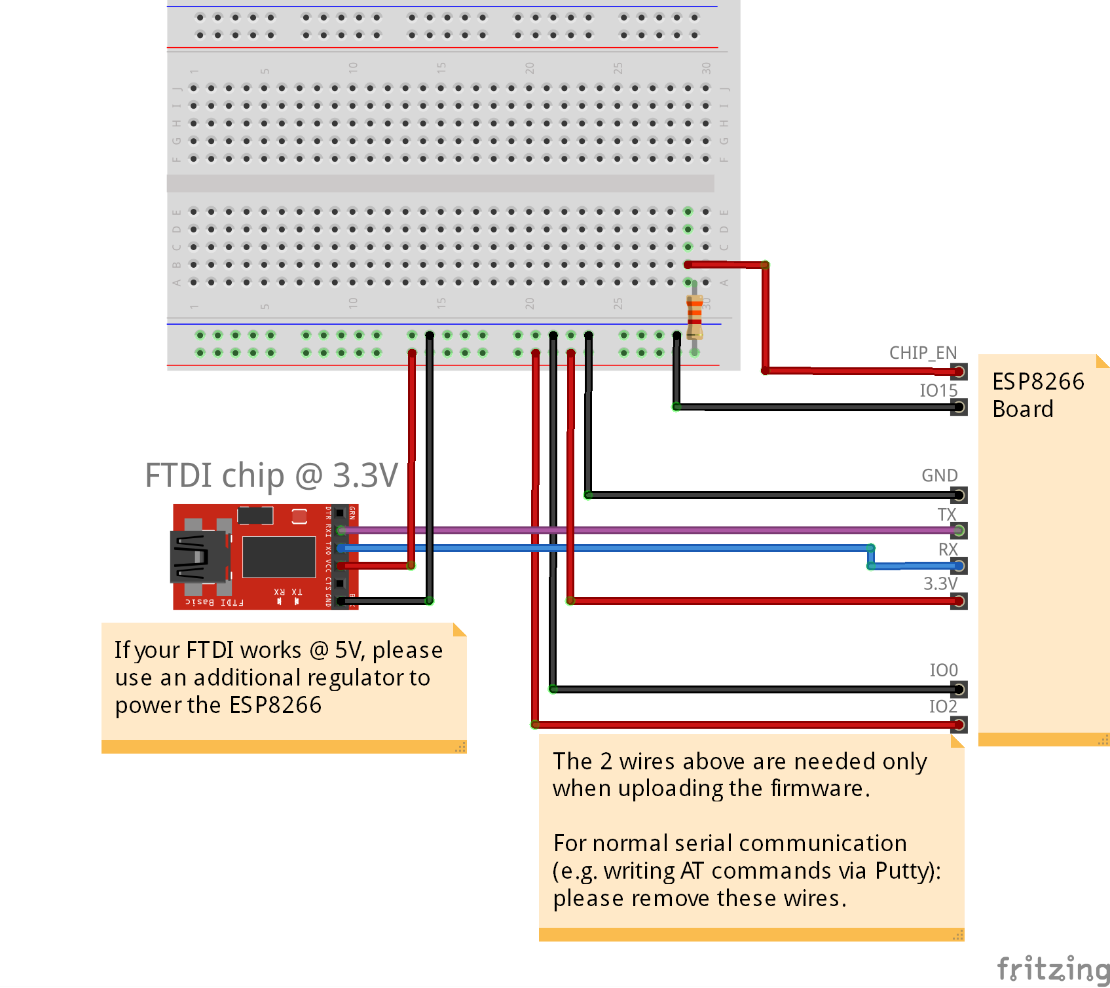


Figure 1. Wiring diagram for uploading firmware to ESP8266

As it is indicated in the note at the bottom of the image, the wiring configuration for normal serial communication is the same except leaving GPIO0 and GPIO2 disconnected. As it will be explained later, GPIO0 is held low for entering bootloader mode, while GPIO2 must be held high to select the UART loading configuration[[1]](#footnote-1).

# Flash tools

## ESPFlashDownloadTool

There is a flash tool available for Windows, which includes a GUI to make the flashing procedure a bit more intuitive. However, I have not been able to connect to the ESP board with it. It can be downloaded from the manufacturer webpage:

<https://www.espressif.com/en/products/hardware/esp8266ex/resources>

under the ‘Tools’ tab:

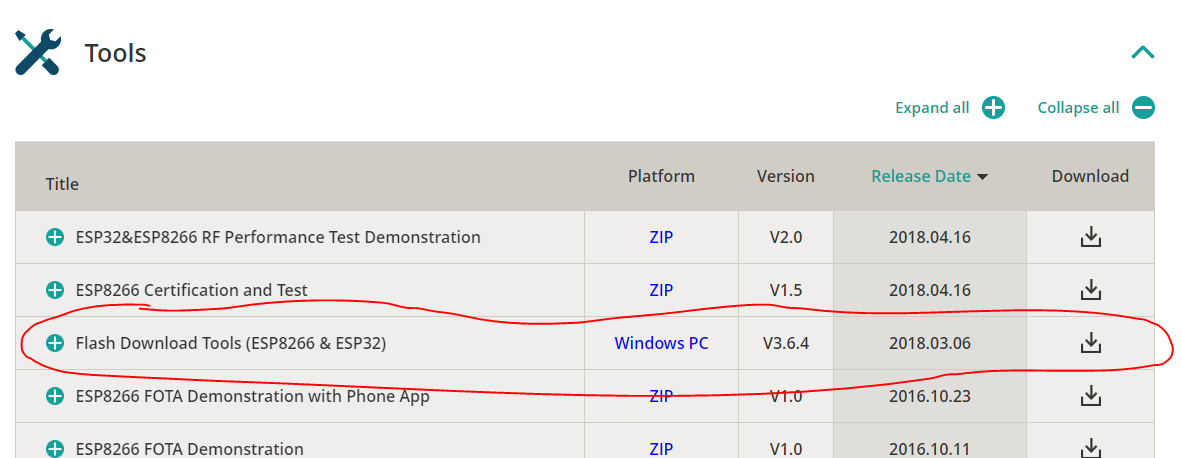


Figure 2. Flash Download Tools for Microsoft Windows, in the Espressif webpage

**For future tests, it would be interesting to guess how it works, because the alternative method proposed in 2.2. requires Linux OS.**

## esptool

This tool started as an unofficial community project but is now supported by Espressif. This is the method I have successfully used to flash the AT firmware into the ESP8266 Huzzah board.

The easiest way to use this tool is installing it from a Linux Terminal. The complete instructions for installing it and further information can be found in: <https://github.com/espressif/esptool>

*Summary of installation:*

1. Python 2.7 or Python 3.4 or newer is needed.
   1. If Python is already installed, the version can be found out by typing:

$ python –-version

* 1. Another version can be installed with the command (changing the version to the desired one):

$ sudo apt-get install python3.4

1. The latest stable esptool release can be installed with:

$ pip install esptool

In some cases, this may not work. Then, use instead:

$ python -m pip install esptool or

$ pip2 install esptool

1. After installation, the tool will be available under the command esptool.py. As usual, all available commands can be checked with $ esptool.py -h

# Flashing procedure

Some explanations of this part are very basic for a person with some knowledge about the Linux terminal. However, in case the reader is a beginner in this topic, they will hopefully be useful.

I have used Ubuntu 16.04 running in a VM VirtualBox (in Windows 10).

**IMPORTANT: for downloading firmware into the ESP8266 board, it must be in bootloader mode. This is achieved by holding GPIO0 low (connect to GND) during startup.** For the Adafruit Huzzah board, this is done in the following way:

Press and hold the GPIO0 button 🡪 an orange led should glow

Without releasing GPIO0, press and release the RESET button

Release GPIO0 🡪 the orange led should remain on but glowing a little more dimly.

This means that the board is now in bootloader mode.

## Configuring serial port from Linux

First, after connecting the board to a USB port of the PC through the FTDI chip, the serial port must be configured from the Linux terminal. In case a Virtual Machine is used, enable it to access the USB port (in VirtualBox: Devices -> USB ports -> tick desired device).

Then, in the Linux terminal, use the command:

$ dmesg | grep tty

Available serial ports will be displayed, with an associated serial device name of the form ttySX for standard serial ports, and ttyUSBX for USB serial ports (for example, in my case, the ESP8266 board device name was /dev/ttyUSB0). This is the identifier of the serial port to be used for all commands (configuring the serial connection, sending and receiving messages, specifying the port for a read/write command…).

Then, the serial port can be configured by means of the stty command. The baud rate generally must be set to 115200 bps for the bootloader mode of the ESP-12 (although it will vary in some cases, for instance, if someone has change the baud rate of the device before). The command syntax for setting the baud rate of the serial port is: **stty -F <port> <baudrate>**. For instance:

$ stty -F /dev/ttyUSB0 115200

If the following steps don’t work, it may be due to an incorrect baud rate, so a possible solution would be trying different values. Changing other configuration parameters, as the number of data bits, stop bits, or parity shouldn’t be necessary.

The first step for flashing the firmware is identifying the flash characteristics (including the size). For this purpose, use the flash\_id command:

$ esptool.py flash\_id

Upon executing the command, a “Connecting………” message is shown in the terminal. In this moment, restart the ESP8266 in *bootloader mode* (see the square at the top of the page), and the flash identification should be executed right away.

A report is shown with a lot of fields. Example of some of them:

Device name: ESP8266

Device type: ESP8266EX

...

Manufacturer: e0

Device: 4016

Detected flash size: 4MB

Once the identification is successful, the device is ready for downloading the firmware. Depending on the detected flash size, the firmware binary files must be written to different addresses. These addresses can be looked up in the README file of the NONOS\_SDK, in <https://github.com/espressif/ESP8266_NONOS_SDK/tree/master/bin/at>.

Finally, the binary data is written to flash with the write\_flash command[[2]](#footnote-2), which has the syntax: **esptool.py --port <port> write\_flash <address> <file> [<address> <file> …]**.

Multiple files can be written at once (fields between brackets). An example of the command is:

$ esptool.py –port /dev/ttyUSB0 write\_flash 0x00000 boot\_v1.2.bin

Like the previous command, this one may take some time to load. Restarting the device again in *bootloader mode* after launching the write command will probably run the firmware download immediately.

# Testing the firmware

Once the 4 required binary files (plus 1 optional) have been downloaded to the device, the next step is testing whether the firmware is working properly.

As indicated before, since the device is going to be used in normal operation mode, the connections of GPIO0 and GPIO2 must be removed. After that, a restart of the module can be executed to enter normal operation mode. For the Adafruit Huzzah, just press the RESET button: the GPIO0 orange led should turn off. If the firmware downloading was correct, the ESP8266 will now respond to AT commands through the UART.

To test it, some AT commands can be sent from the serial port to check the answers of the module. This can be done either from the Linux terminal itself, or from a serial port terminal application like CoolTerm.

Two easy commands which can be tested are AT (check communication) and AT+GMR (check version information). The expected output of this commands is shown in Figure 3. As it can be also seen, sending an incorrect command (AT+GMT) makes the device to return ERROR.

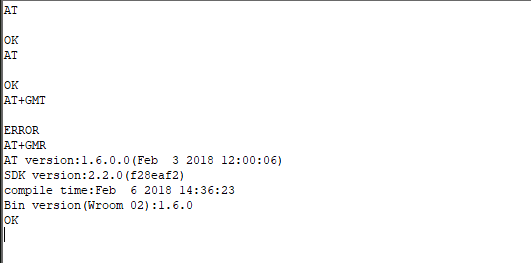


Figure 3. Test of some AT commands to ESP8266 (using CoolTerm)

# References and repositories

1. Online information about the ESP8266 – manufacturer webpage

<https://www.espressif.com/en/products/hardware/esp8266ex/overview>

1. Another tutorial with additional information

<http://flower-platform.com/2015/12/16/esp8266-with-at-commands-flashingupdating-the-firmware-step-by-step/>

1. AT Firmware binary files and addresses to download – GitHub repository

<https://github.com/espressif/ESP8266_NONOS_SDK/tree/master/bin/at>

1. esptool – GitHub repository

<https://github.com/espressif/esptool>

1. Sending and receiving serial data from Linux terminal

<https://unix.stackexchange.com/questions/117037/how-to-send-data-to-a-serial-port-and-see-any-answer>

1. I’m not sure of this affirmation, but GPIO2 must be held high for sure for the flashing to work properly. [↑](#footnote-ref-1)
2. Before executing the command, the directory that contains the files must be set as the working directory, by means of the cd command. [↑](#footnote-ref-2)