Restore or Upgrade Firmware on ESP8266 (ESP-01) Module Using Arduino UNO

Introduction: Restore or Upgrade Firmware on ESP8266 (ESP-01) Module Using Arduino UNO

The ESP-01 module that i used originally came with older AI Thinker firmware, which limits its capabilities since many useful AT commands are not supported.

It is generally a good idea to upgrade your firmware for bug fixes and also depending on the functionalities needed from the ESP module, a more complete set of supported AT commands might be required. This tutorial also explains how to restore or flash original Espressif firmware if you happen to mess up your ESP firmware, for example, through some wrong AT commands. This is what happened when i tried to change the default baud rate from 115200 to 9600. I used the following command (based on a quick Google search):

```
AT+IPR=9600
```

This bricked the ESP-01 module. No AT commands worked anymore, which prompted me to do more research and after many hours, i finally found a way to restore my ESP-01 and install the latest compatible firmware. So, i decided it was worth sharing the whole process.

Caution: Based on my experience, i would highly advise not to use the command AT+IPR to change your baud rate as it will most likely require you to re-flash your firmware. However, there are many different variants of the ESP-01 module with different flash size and factory firmware version, so your experience might vary.

The ESP-01 module is not 5 V-compliant and requires 3.3 V to power it but also to work properly on the logic level. To communicate with the computer, the module additionally needs a USB to serial converter. So, instead of using a voltage converter and a USB to serial adapter, i decided to opt for a simpler solution. Since i already have an Arduino UNO, i used the latter to power the ESP-01 module and establish communication between the ESP-01 and the computer, effectively operating as a serial bridge.

Supplies

1. ESP-01 module
2. Arduino UNO (with USB cable)
3. Jumper DuPont wires
4. Breadboard

Step 1: Wire Connections

The wire connections with the Arduino UNO and ESP-01 module can be made easily using jumper cables and a breadboard. The connections between the Arduino development board and the ESP module are described in the table. The RX and TX connections are not reversed in this case, since the communication is not happening between the Arduino and ESP module but between the ESP and the computer. So, in this case, the Arduino UNO board is used as a serial bridge via its on-board USB to serial converter.

The RX pins connection between the Arduino board and the ESP module should be made through a voltage divider, since the ESP-01 operates on 3.3 V logic level and receiving 5 V logic level from the Arduino UNO could damage the ESP module. You can build a simple 3.3 V voltage divider using only 2 resistors as shown in the figure. Optionally, you could use a logic level converter.

Caution: For my setup, i just wired the RX pins directly (not advisable!) and everything worked, but if you decide to forego the logic level conversion, proceed at your own risk!

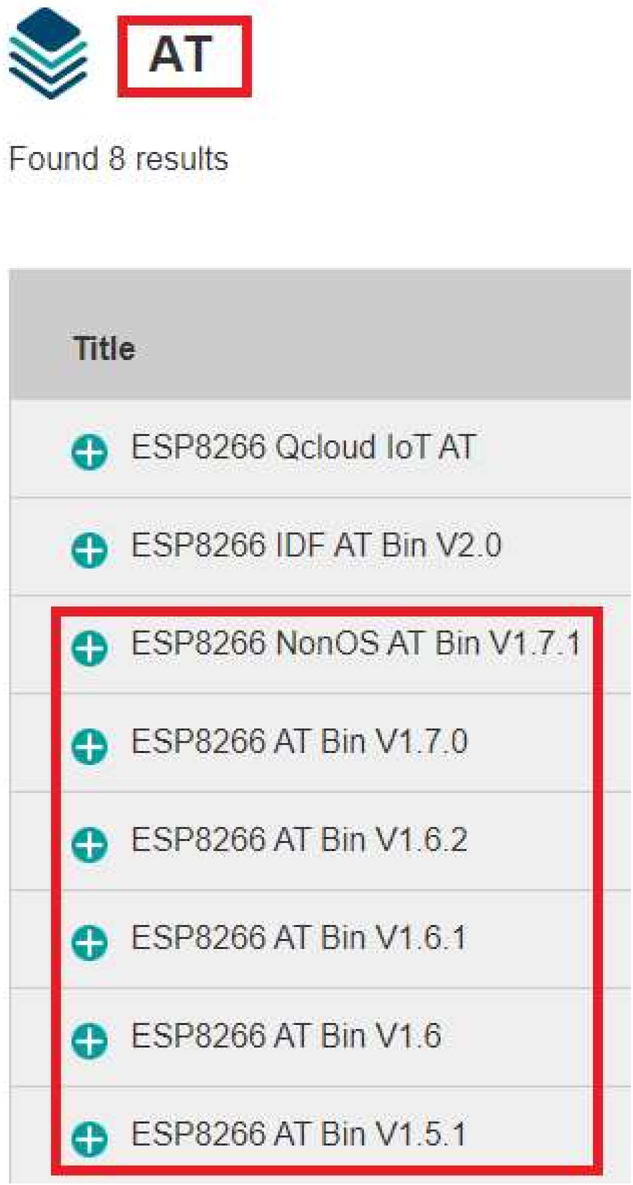
To establish a common Ground connection, the Arduino UNO Ground pin is connected to the ESP module Ground pin.

The Arduino UNO is used to provide 3.3 V power directly to the ESP module's VCC pin. Connect the 3.3 V pin to the breadboard using a jumper wire, since the 3.3 V will be used not only for the ESP's VCC pin but also the ESP's CH_PD pin to enable the ESP chip.

The RESET pin on the Arduino UNO is connected to Ground to bypass any code uploaded via the Arduino so that the code is instead sent from the computer to the ESP-01.

The table figure shows the full set of connections for firmware flashing but at this stage, do not connect the ESP's Reset and GPIO_0 pins since we will only be reading the firmware information in the next step.

Step 2: Check Firmware Version



AT

Found 8 results

Title
ESP8266 Qcloud IoT AT
ESP8266 IDF AT Bin V2.0
ESP8266 NonOS AT Bin V1.7.1
ESP8266 AT Bin V1.7.0
ESP8266 AT Bin V1.6.2
ESP8266 AT Bin V1.6.1
ESP8266 AT Bin V1.6
ESP8266 AT Bin V1.5.1

To find your ESP-01 firmware version, in the Arduino IDE Serial Monitor, type:

```
AT+GMR
```

Note: All AT commands need to be typed in uppercase with no spaces.

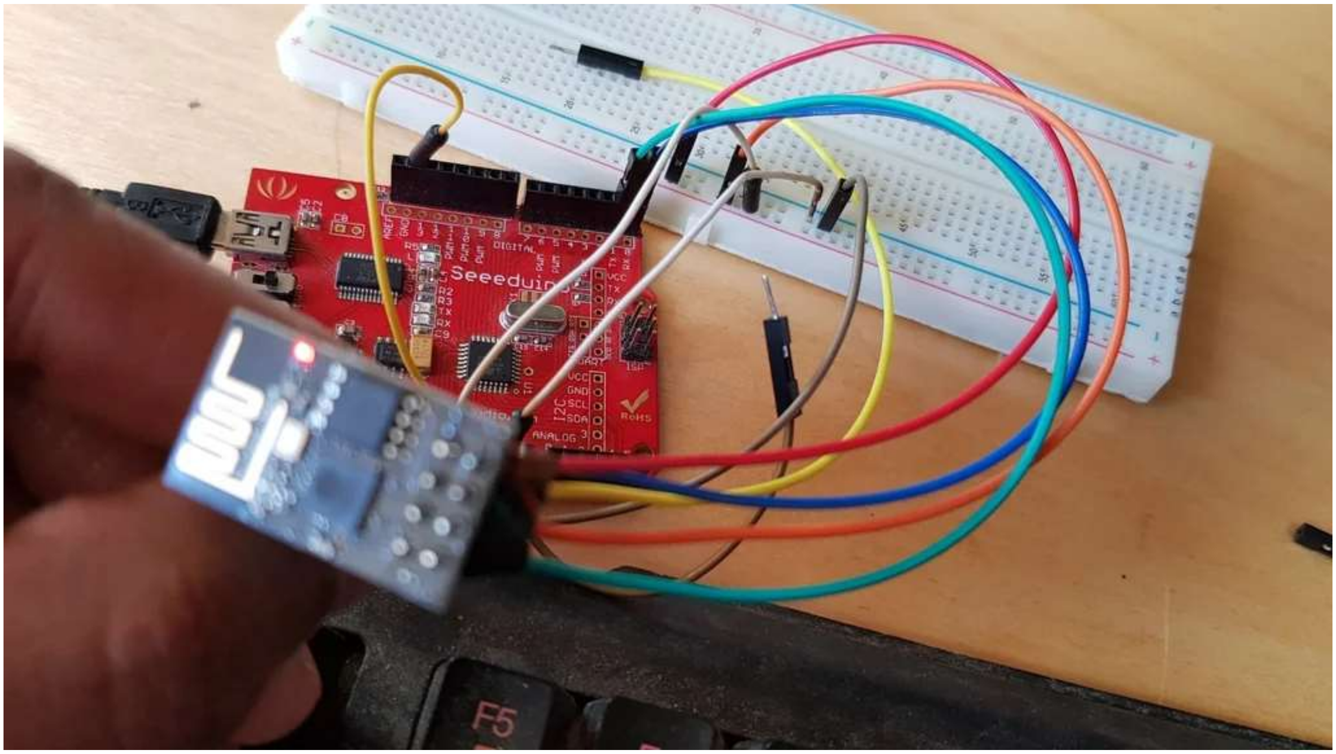
Here is the serial output for my ESP-01 (your ESP module might not show exactly the same information as it depends on the specific model and release date):

```
AT+GMR
AT version:0.25.0.0(Jun  5 2015 16:27:16)
SDK version:1.1.1
Ai-Thinker Technology Co. Ltd.
Jun 23 2015 23:23:50
OK
```

Now, to upgrade to the latest official Espressif ESP8266EX firmware, go to the resources section of its website: https://www.espressif.com/en/products/hardware/esp8266ex/resources

Note: The list of available firmware might not all be compatible with your ESP-01 model. Check the Troubleshooting section at the end of this tutorial for more information.

Step 3: Preparing the Setup: Before Firmware Flashing



Now, we will prepare the Arduino UNO for the ESP-01 firmware flashing procedure.

Make sure that the Arduino RESET wire is disconnected from Ground. Also, the TX and RX wires from the Arduino UNO need to be disconnected from the ESP-01 module.

Open Arduino IDE and from the top menu, go to File > Examples > 01.Basics > BareMinimum. Upload the sketch to the Arduino UNO. This blank sketch will ensure that no communication interference occurs with the ESP module.

Reconnect the RX and TX cables between the UNO and ESP-01. Also, connect the UNO's RESET pin to Ground.

The CH_PD or CH_EN pin stands for 'Chip Power-Down' or 'Chip Enable' and it needs to be pulled HIGH or connected to 3.3 V in order to enable the ESP chip.

Two additional jumper cables are needed for the following ESP pins: GPIO_0 and RESET.

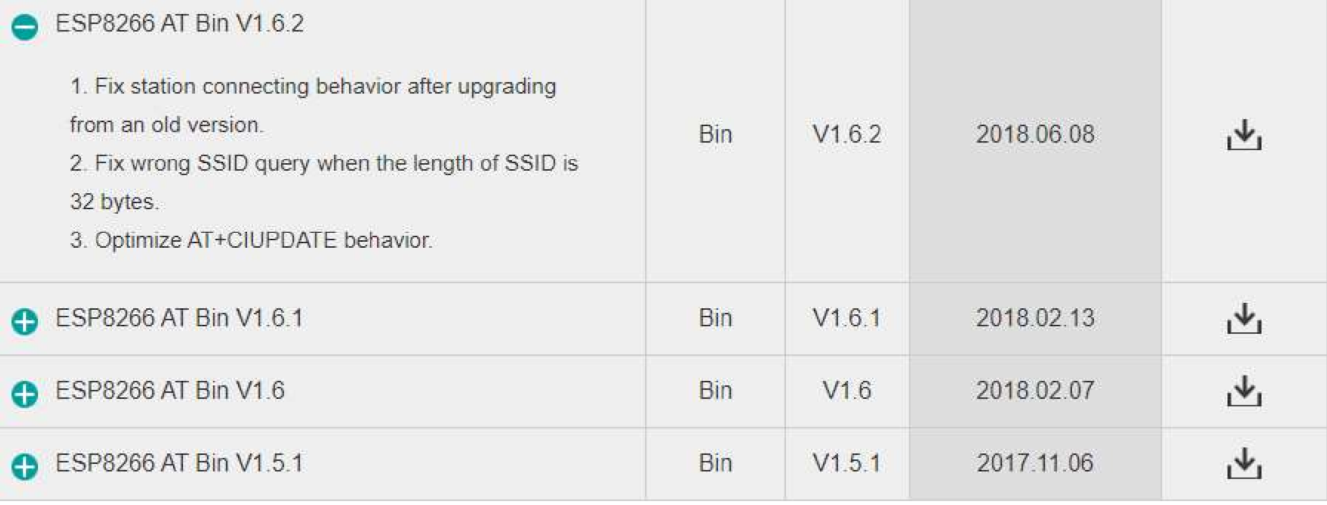
GPIO_2 is not used and left disconnected.

The ESP-01 needs to be set in programming mode so that code can be uploaded to it. But the ESP-01 does not have the necessary on-board additional hardware to achieve this directly, so it all needs to be connected separately. For my own use, i did not bother using switches, but instead simply used two male-female jumper cables connected to the GPIO_0 and RESET pins of the ESP-01 module and plugged them into the breadboard's common Ground pins from the Arduino UNO's Ground.

Before starting the flashing procedure, GPIO_0 is connected to Ground for the entire flashing procedure to enable programming mode.

The RESET pin is connected to Ground for a second and then removed. This allows the new firmware to be uploaded.

Step 4: Download Flashing Tool and Firmware



⊖ ESP8266 AT Bin V1.6.2 1. Fix station connecting behavior after upgrading from an old version. 2. Fix wrong SSID query when the length of SSID is 32 bytes. 3. Optimize AT+CIUPDATE behavior.	Bin	V1.6.2	2018.06.08	⤓
⊕ ESP8266 AT Bin V1.6.1	Bin	V1.6.1	2018.02.13	⤓
⊕ ESP8266 AT Bin V1.6	Bin	V1.6	2018.02.07	⤓
⊕ ESP8266 AT Bin V1.5.1	Bin	V1.5.1	2017.11.06	⤓

The firmware used is from Espressif which is the original manufacturer of the ESP8266 chip.

Go to: https://www.espressif.com/en/products/hardware/esp8266ex/resources to access the official tools and AT firmware files.

Under the 'Tools' tab, download the Flash Download Tools (ESP8266 & ESP32), currently the latest one is V3.6.8.

Under the 'AT' tab, download the latest compatible AT firmware, which is ESP8266 AT Bin V1.6.2 for my ESP-01 model. It depends on the ESP-01 model since it can have different flash memory size. You will find more information about this in the **DETECTED INFO** section of the Flash Download Tools program after clicking on **START** to begin the flashing procedure. So, if you are not sure about the flash size or which firmware files to select, just run the program to get the correct information about your ESP module.

After flashing, you can use AT commands to test and work with the ESP-01. Download the official ESP8266 AT instruction set: https://www.espressif.com/sites/default/files/documentation/4a-esp8266_at_instruction_set_en.pdf

Step 5: Flashing Procedure

Extract the Flash Download Tools zip file and open the exe file. Make sure to run it as administrator if you're using Windows. A DOS window will first open, followed by a pop-up window. Select ESP8266 Download Tool. This will open a window with several configurable options.

At this stage, refer to the screenshot to see exactly what options to select and what to type in the hex code boxes next to each of the selected firmware files.

Note: Select the firmware files in the same order since the flash procedure is done sequentially. The blank.bin file needs to be selected three times as shown in the AT instruction set screenshot.

To find the correct hex codes or addresses, check the official AT instruction set document. I have attached a screenshot of the table which i used for flashing my ESP module.

Then, just press on the **START** button and watch the magic happen. The button text will show **SYNC** and the **DETECTED INFO** section will display the specifications of your ESP module. Then, the text will show **DOWNLOAD** and the progress bar will activate as the firmware files are uploaded to the ESP flash memory. After the firmware flashing procedure is done, you will see: **FINISH**.

Close the Flash Download Tools program. This is necessary to free the serial port to allow the Arduino IDE Serial Monitor to take over.

Disconnect the ESP GPIO_0 pin from Ground connection. This will disable programming mode.

Connect RESET to Ground for 1 second and then disconnect. This will reset the module.

Open Arduino IDE. From the top menu, select Tools > Port > select the correct COM port. Open Serial Monitor and choose "Both NL & CR" and select baud rate of 115200 which is the default one.

Type:

```
AT
```

If the flashing procedure went correctly, the response will be:

```
OK
```

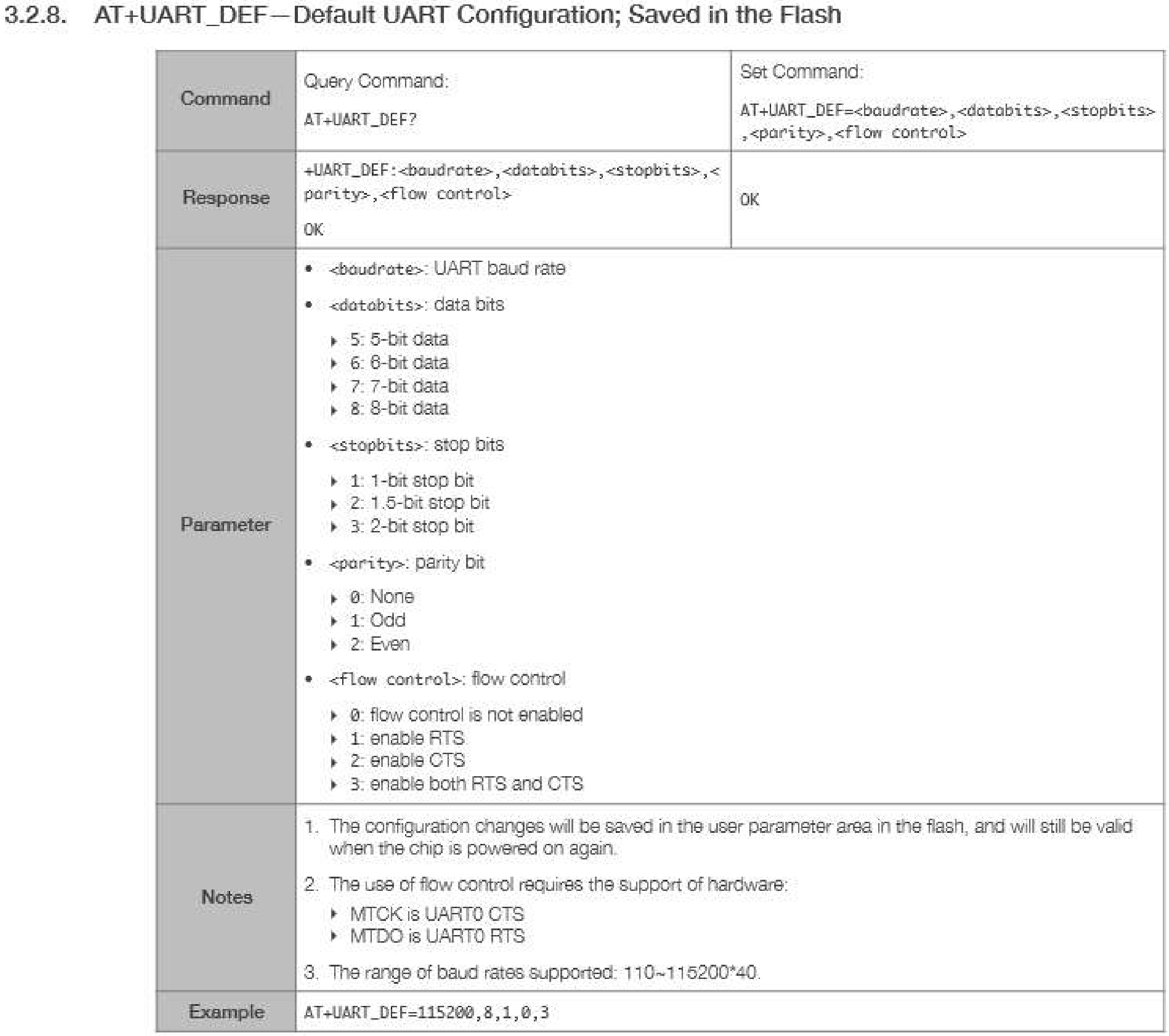
To verify your new firmware, type:

```
AT+GMR
```

Here is the Serial Monitor output for my ESP-01:

```
AT+GMR
AT version:1.6.2.0(Apr 13 2018 11:10:59)
SDK version:2.2.1(6ab97e9)
compile time:Jun  7 2018 19:34:26
Bin version(Wroom 02):1.6.2
OK
```

Step 6: Change the Baud Rate Permanently



3.2.8. AT+UART_DEF—Default UART Configuration; Saved in the Flash

Command	Query Command: `AT+UART_DEF?`	Set Command: `AT+UART_DEF=<baudrate>,<databits>,<stopbits>,<parity>,<flow control>`

Response	`+UART_DEF:<baudrate>,<databits>,<stopbits>,<parity>,<flow control>`
	`OK`
		`OK`

Parameter	• `<baudrate>`: UART baud rate • `<databits>`: data bits ‣ 5: 5-bit data ‣ 6: 6-bit data ‣ 7: 7-bit data ‣ 8: 8-bit data • `<stopbits>`: stop bits ‣ 1: 1-bit stop bit ‣ 2: 1.5-bit stop bit ‣ 3: 2-bit stop bit • `<parity>`: parity bit ‣ 0: None ‣ 1: Odd ‣ 2: Even • `<flow control>`: flow control ‣ 0: flow control is not enabled ‣ 1: enable RTS ‣ 2: enable CTS ‣ 3: enable both RTS and CTS	
Notes	1. The configuration changes will be saved in the user parameter area in the flash, and will still be valid when the chip is powered on again. 2. The use of flow control requires the support of hardware: ‣ MTCK is UART0 CTS ‣ MTDO is UART0 RTS 3. The range of baud rates supported: 110~115200*40.	
Example	`AT+UART_DEF=115200,8,1,0,3`	

This part is optional. The default baud rate is 115200 but if you want to change it to another baud speed, then you can simply type the following command in the Arduino Serial Monitor.

For example, if you want to change to 9600 baud rate / 8 data bits / 1 stop bits / no parity bit / no flow control.
Type in:

```
AT+UART_DEF=9600,8,1,0,0
```

Response should be:

```
OK
```

The command above changes the baud rate permanently to 9600 but you can choose any of the standard baud speeds.

Step 7: Troubleshooting

If there are any anomalies after successful flashing of the firmware, for example, in the Serial Monitor, after you select the default baud rate of 115200 and type: AT but do not see any response or if you type AT+GMR and receive some other kind of information other than the firmware version information, then you might have flashed the wrong firmware. In that case, in the Flash Download Tools program, after you press the **START** button to begin the firmware flashing procedure, the status green button reads **SYNC** and that's the stage where the ESP module information is extracted and made available under **DETECTED INFO**. This information is crucial to determine the right firmware and correct files to flash.

The following is for my ESP-01 module:

```
flash vendor:
E0h : N/A
flash devID:4014h
QUAD;8Mbit
crystal:26 Mhz
```

Your **DETECTED INFO** will depend on the ESP-01 model. But this will allow you to deduce the correct flash size. In my ESP, it is 8Mbit which is equivalent to 1 MB. So, this means that i should choose 512 KB + 512 KB from the firmware files. And this also means that firmware 1.7.0 or 1.7.1 will not work properly for my ESP module, which i also confirmed by flashing these firmware and testing. Also, in the description of the 1.7.0 and 1.7.1 firmware on the official Espressif resource website, the following information is listed: "Limited by the size of the AT bin file, only `1024+1024 flash map` is supported by default." 1024+1024 means that it's for an ESP module with a total flash size of 2048 KB or 2 MB.

Note: In case you're wondering, you can downgrade or upgrade between different firmware versions without any restrictions. If anything goes wrong, you can just flash again with different parameters or try a different firmware.