

# ESP8266 AT Instruction Set



Version 2.2  
Copyright © 2018

# About This Guide

This document provides AT commands list based on ESP8266\_NONOS\_SDK.

The document is structured as follows:

Chapter	Title	Content
Chapter 1	Overview	Provides instructions on user-defined AT commands and downloading of AT firmware.
Chapter 2	Command Description	Gives a basic description of AT commands.
Chapter 3	Basic AT Commands	Lists AT commands of basic functions.
Chapter 4	Wi-Fi AT Commands	Lists Wi-Fi-related AT commands.
Chapter 5	TCP/IP-Related AT Commands	Lists TCP/IP-related AT commands.
Chapter 6	Appendix A	Lists the AT commands of which the configuration is saved in the flash.
Chapter 7	Appendix B	Lists messages of AT firmware.
Chapter 8	Q & A	Provides information on where and how to consult questions about ESP8266 AT commands.

## Release Notes

Date	Version	Release notes
2016.04	V1.5.3	First Release.
2016.05	V1.5.4	Updated Section 5.2.16 and Section 5.2.19
2016.07	V2.0.0	Added Section 3.2.11, updated Section 1.2
2017.05	V2.1.0	Updated Section 3.2, Section 4.1 and Section 5.2.
2017.08	V2.1.1	Added Appendix B.
2018.02	V2.2	Added Section 3.2.22, 3.2.23, 4.2.39, 4.2.40, 5.2.15 Updated 4.2.7, 4.2.8, and Appendix B.

## Documentation Change Notification

Espressif provides email notifications to keep customers updated on changes to technical documentation. Please subscribe [here](#).

## Certifications

Please download the product certification(s) [here](#).

# 目录

<b>1. Overview</b>	<b>1</b>
1.1. User-Defined AT Commands	1
1.2. Downloading AT Firmware into the Flash	2
1.2.1. 4 Mbit Flash	2
1.2.2. 8 Mbit Flash	2
1.2.3. 16 Mbit Flash, Map: 512 KB + 512 KB	3
1.2.4. 16 Mbit Flash, Map: 1024 KB + 1024 KB	3
1.2.5. 32 Mbit Flash, Map: 512 KB + 512 KB	4
1.2.6. 32 Mbit Flash, Map: 1024 KB + 1024 KB	4
<b>2. Command Description</b>	<b>6</b>
<b>3. Basic AT Commands</b>	<b>7</b>
3.1. Overview	7
3.2. Commands	8
3.2.1. AT—Tests AT Startup	8
3.2.2. AT+RST—Restarts the Module	8
3.2.3. AT+GMR—Checks Version Information	8
3.2.4. AT+GSLP—Enters Deep-sleep Mode	8
3.2.5. ATE—AT Commands Echoing	9
3.2.6. AT+RESTORE—Restores the Factory Default Settings	9
3.2.7. AT+UART—UART Configuration	9
3.2.8. AT+UART_CUR—Current UART Configuration; Not Saved in the Flash	11
3.2.9. AT+UART_DEF—Default UART Configuration; Saved in the Flash	12
3.2.10. AT+SLEEP—Configures the Sleep Modes	13
3.2.11. AT+WAKEUPGPIO—Configures a GPIO to Wake ESP8266 up from Light-sleep Mode	13
3.2.12. AT+RFPOWER—Sets the Maximum Value of RF TX Power	14
3.2.13. AT+RFVDD—Sets RF TX Power According to VDD33	14
3.2.14. AT+RFAUTOTRACE—Sets RF Frequency Offset Trace	15
3.2.15. AT+SYSRAM—Checks the Remaining Space of RAM	15
3.2.16. AT+SYSADC—Checks the Value of ADC	15
3.2.17. AT+SYSIOSETCFG—Configures IO Working Mode	16

3.2.18. AT+SYSIOGETCFG—Checks the Working Modes of IO Pins.....	16
3.2.19. AT+SYSGPIODIR—Configures the Direction of a GPIO .....	16
3.2.20. AT+SYSGPIOWRITE—Configures the Output Level of a GPIO .....	17
3.2.21. AT+SYSGPIOREAD—Reads the GPIO Input Level.....	17
3.2.22. AT+SYSMSG_CUR—Set Current System Messages .....	18
3.2.23. AT+SYSMSG_DEF—Set Default System Messages.....	18

## 4. Wi-Fi AT Commands .....20

4.1. Overview .....	20
4.2. Commands .....	22
4.2.1. AT+CWMODE—Sets the Wi-Fi Mode (Station/SoftAP/Station+SoftAP).....	22
4.2.2. AT+CWMODE_CUR—Sets the Current Wi-Fi mode; Configuration Not Saved in the Flash.....	22
4.2.3. AT+CWMODE_DEF—Sets the Default Wi-Fi mode; Configuration Saved in the Flash .....	22
4.2.4. AT+CWJAP—Connects to an AP .....	23
4.2.5. AT+CWJAP_CUR—Connects to an AP; Configuration Not Saved in the Flash.....	24
4.2.6. AT+CWJAP_DEF—Connects to an AP; Configuration Saved in the Flash .....	24
4.2.7. AT+CWLAPOPT—Sets the Configuration for the Command AT+CWLAP.....	26
4.2.8. AT+CWLAP—Lists Available APs.....	26
4.2.9. AT+CWQAP—Disconnects from the AP .....	28
4.2.10. AT+CWSAP—Configures the ESP8266 SoftAP .....	28
4.2.11. AT+CWSAP_CUR—Configures the ESP8266 SoftAP; Configuration Not Saved in the Flash....	28
4.2.12. AT+CWSAP_DEF—Configures the ESP8266 SoftAP; Configuration Saved in the Flash .....	29
4.2.13. AT+CWLIF—IP of Stations to Which the ESP8266 SoftAP is Connected.....	30
4.2.14. AT+CWDHCP—Enables/Disables DHCP .....	30
4.2.15. AT+CWDHCP_CUR—Enables/Disables DHCP; Configuration Not Saved in the Flash .....	31
4.2.16. AT+CWDHCP_DEF—Enables/Disables DHCP; Configuration Saved in the Flash.....	31
4.2.17. AT+CWDHCPS_CUR—Sets the IP Address Allocated by ESP8266 SoftAP DHCP; Configuration Not Saved in Flash .....	32
4.2.18. AT+CWDHCPS_DEF—Sets the IP Address Allocated by ESP8266 SoftAP DHCP; Configuration Saved in Flash.....	32
4.2.19. AT+CWAUTOCONN—Auto-Connects to the AP or Not .....	33
4.2.20. AT+CIPSTAMAC—Sets the MAC Address of the ESP8266 Station .....	33
4.2.21. AT+CIPSTAMAC_CUR—Sets the MAC Address of the ESP8266 Station; Configuration Not Saved in the Flash .....	34
4.2.22. AT+CIPSTAMAC_DEF—Sets the MAC Address of the ESP8266 Station; Configuration Saved in the Flash.....	34

4.2.23. AT+CIPAPMAC—Sets the MAC Address of the ESP8266 SoftAP .....	34
4.2.24. AT+CIPAPMAC_CUR—Sets the MAC Address of the ESP8266 SoftAP; Configuration Not Saved in the Flash .....	35
4.2.25. AT+CIPAPMAC_DEF—Sets the MAC Address of the ESP8266 SoftAP; Configuration Saved in Flash.....	36
4.2.26. AT+CIPSTA—Sets the IP Address of the ESP8266 Station .....	36
4.2.27. AT+CIPSTA_CUR—Sets the Current IP Address of the ESP8266 Station; Configuration Not Saved in the Flash .....	37
4.2.28. AT+CIPSTA_DEF—Sets the Default IP Address of the ESP8266 Station; Configuration Saved in the Flash.....	37
4.2.29. AT+CIPAP—Sets the IP Address of the ESP8266 SoftAP .....	38
4.2.30. AT+CIPAP_CUR—Sets the IP Address of the ESP8266 SoftAP; Configuration Not Saved in the Flash.....	38
4.2.31. AT+CIPAP_DEF—Sets the Default IP Address of the ESP8266 SoftAP; Configuration Saved in the Flash.....	39
4.2.32. AT+CWSTARTSMART—Starts SmartConfig.....	39
4.2.33. AT+CWSTOPSMART—Stops SmartConfig .....	40
4.2.34. AT+CWSTARTDISCOVER—Enables the Mode that ESP8266 can be Found by WeChat.....	40
4.2.35. AT+CWSTOPDISCOVER—Disables the Mode that ESP8266 can be Found by WeChat .....	41
4.2.36. AT+WPS—Enables the WPS Function.....	41
4.2.37. AT+MDNS—Configures the MDNS Function .....	41
4.2.38. AT+CWHOSTNAME—Configures the Name of ESP8266 Station .....	42
4.2.39. AT+CWCOUNTRY_CUR—Set ESP8266 WiFi Country Code; Configuration Not Saved in the Flash.....	42
4.2.40. AT+CWCOUNTRY_DEF—Set the default WiFi Country Code of ESP8266; Configuration Saved in the Flash .....	44

## 5. TCP/IP-Related AT Commands .....45

5.1. Overview .....	45
5.2. Commands .....	46
5.2.1. AT+CIPSTATUS—Gets the Connection Status .....	46
5.2.2. AT+CIPDOMAIN—DNS Function .....	46
5.2.3. AT+CIPSTART—Establishes TCP Connection, UDP Transmission or SSL Connection .....	47
5.2.4. AT+CIPSSLSIZE—Sets the Size of SSL Buffer.....	49
5.2.5. AT+CIPSEND—Sends Data.....	49
5.2.6. AT+CIPSENDEX—Sends Data .....	50
5.2.7. AT+CIPSENDBUF—Writes Data into the TCP-Send-Buffer.....	50

5.2.8.	AT+CIPBUFRESET—Resets the Segment ID Count.....	51
5.2.9.	AT+CIPBUFSTATUS—Checks the Status of the TCP-Send-Buffer .....	52
5.2.10.	AT+CIPCHECKSEQ—Checks If a Specific Segment Was Successfully Sent .....	52
5.2.11.	AT+CIPCLOSE—Closes the TCP/UDP/SSL Connection.....	53
5.2.12.	AT+CIFSR—Gets the Local IP Address .....	53
5.2.13.	AT+CIPMUX—Enable or Disable Multiple Connections.....	53
5.2.14.	AT+CIPSERVER—Deletes/Creates TCP Server .....	54
5.2.15.	AT+CIPSERVERMAXCONN—Set the Maximum Connections Allowed by Server .....	54
5.2.16.	AT+CIPMODE—Sets Transmission Mode.....	55
5.2.17.	AT+SAVETRANSLINK—Saves the Transparent Transmission Link in Flash .....	55
5.2.18.	AT+CIPSTO—Sets the TCP Server Timeout .....	56
5.2.19.	AT+PING—Ping Packets.....	56
5.2.20.	AT+CIUPDATE—Updates the Software Through Wi-Fi.....	58
5.2.21.	AT+CIPDINFO—Shows the Remote IP and Port with +IPD.....	58
5.2.22.	+IPD—Receives Network Data .....	58
5.2.23.	AT+CIPSNTPCFG—Sets the Configuration of SNTP.....	59
5.2.24.	AT+CIPSNTPTIME—Checks the SNTP Time.....	59
5.2.25.	AT+CIPDNS_CUR—Sets User-defined DNS Servers; Configuration Not Saved in the Flash ....	59
5.2.26.	AT+CIPDNS_DEF—Sets User-defined DNS Servers; Configuration Saved in the Flash.....	60
<b>6.</b>	<b>Appendix A.....</b>	<b>61</b>
<b>7.</b>	<b>Appendix B.....</b>	<b>62</b>
<b>8.</b>	<b>Q&amp;A.....</b>	<b>63</b>



# 1. Overview

This document provides AT commands based on ESP8266\_NONOS\_SDK and explain how to use them. AT command set is divided into: Basic AT commands, Wi-Fi AT commands, and TCP/IP AT commands.

## 1.1. User-Defined AT Commands

Please use only English letters when naming user-defined AT commands. The AT command name must NOT contain characters or numbers.

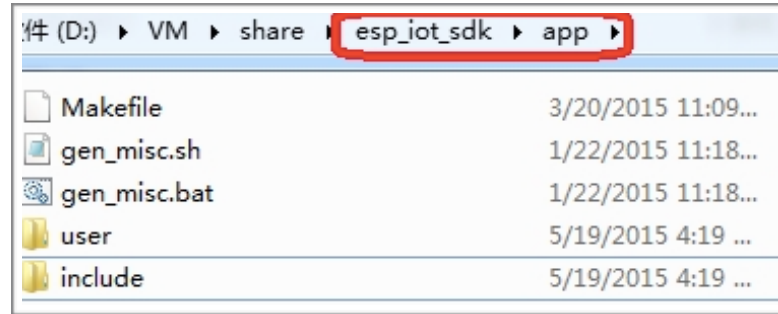
AT firmware is based on ESP8266\_NONOS\_SDK. Espressif Systems' AT commands are provided in *libat.a*, which is included in the AT BIN firmware. Examples of customized, user-defined AT commands are provided in *ESP8266\_NONOS\_SDK/example/at*.

Examples of implementing user-defined AT commands are provided in *ESP8266\_NONOS\_SDK/examples/at/user/user\_main.c*. The structure, *at\_funcationType*, is used to define four types of a command, for details of which please refer to the following table.

Definition	Type	Description	
at_testCmd	Test	AT Command	AT+TEST=?
		Registered Callback In Example	at_testCmdTest
		Function Design	Return the value range of parameters
		If at_testCmd is registered as NULL, there will be no testing command.	
at_queryCmd	Query	AT Command	AT+TEST?
		Registered Callback In Example	at_queryCmdTest
		Function Design	Return the current value
		If at_queryCmd is registered as NULL, there will be no Query Command.	
at_setupCmd	Set	AT Command	AT+TEST=parameter1, parameter2, ...
		Registered Callback In Example	at_setupCmdTest
		Function Design	Set configuration
		If at_setupCmd is registered as NULL, there will be no setup command.	
at_exeCmd	Execute	AT Command	AT+TEST
		Registered Callback In Example	at_exeCmdTest
		Function Design	Execute an action
		If at_exeCmd is registered as NULL, there will be no execution command.	



All the files in folder **at** should be copied to the folder **app** in **ESP8266\_NONOS\_SDK** if users need to compile the AT firmware.



For details please refer to [ESP8266 Getting Started Guide](#).

## 1.2. Downloading AT Firmware into the Flash

Please refer to **ESP8266\_NONOS\_SDK/bin/at/readme.txt** for instructions on how to download AT firmware into flash. Please use Espressif's official Flash Download Tools to download the firmware. Make sure you select the corresponding flash size.

Espressif's official Flash Download Tools:

[http://espressif.com/en/support/download/other-tools?keys=&field\\_type\\_tid%5B%5D=14](http://espressif.com/en/support/download/other-tools?keys=&field_type_tid%5B%5D=14).

### 1.2.1. 4 Mbit Flash

With the release of ESP8266\_NONOS\_SDK\_V2.0.0, AT\_V1.3, AT firmware can use 4-Mbit flash but does not supports FOTA (upgrade AT firmware through Wi-Fi) function.

BIN	Address	Description
blank.bin	0x78000	Initializes the RF_CAL parameter area.
esp_init_data_default.bin	0x7C000	Stores the default RF parameter values; the BIN has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this BIN has to be downloaded too.
blank.bin	0x7A000	Initializes the flash user parameter area; for more details please see <b>Appendix</b> .
blank.bin	0x7E000	Initializes Flash system parameter area; for more details please see <b>Appendix</b> .
eagle.flash.bin	0x00000	In <b>/bin/at/noboot</b> .
eagle.irom0text.bin	0x10000	In <b>/bin/at/noboot</b> .

### 1.2.2. 8 Mbit Flash

If the flash size is 8 Mbit or larger, users can use boot mode which supports AT firmware upgrade feature through Wi-Fi by command **AT+CIUPDATE**. Use Espressif Flash download tool and select flash size: 8 Mbit.





BIN	Address	Description
blank.bin	0xFB000	Initializes the RF_CAL parameter area.
esp_init_data_default.bin	0xFC000	Stores the default RF parameter values; the BIN has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this BIN has to be downloaded too.
blank.bin	0x7E000	Stores the default RF parameter values; the BIN has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this BIN has to be downloaded too.
blank.bin	0xFE000	Initializes the flash user parameter area; for more details please see <b>Appendix</b> .
boot.bin	0x00000	In <b>/bin/at</b>
user1.1024.new.2.bin	0x01000	In <b>/bin/at/512+512</b>

### 1.2.3. 16 Mbit Flash, Map: 512 KB + 512 KB

Use Espressif Flash download tool and select flash size: 16 Mbit.

BIN	Address	Description
blank.bin	0x1FB000	Initializes RF_CAL parameter area.
esp_init_data_default.bin	0x1FC000	Stores default RF parameter values, has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this bin has to be downloaded too.
blank.bin	0x7E000	Initializes Flash user parameter area, more details in <b>Appendix</b> .
blank.bin	0x1FE000	Initializes Flash system parameter area, more details in <b>Appendix</b> .
boot.bin	0x00000	In <b>/bin/at</b> .
user1.1024.new.2.bin	0x01000	In <b>/bin/at/512+512</b> .

### 1.2.4. 16 Mbit Flash, Map: 1024 KB + 1024 KB

Use Espressif Flash download tool and select flash size: 16 Mbit-C1.

BIN	Address	Description
blank.bin	0x1FB000	Initializes RF_CAL parameter area.



BIN	Address	Description
esp_init_data_default.bin	0x1FC000	Stores default RF parameter values, has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this bin has to be downloaded too.
blank.bin	0xFE000	Initializes Flash user parameter area, more details in <b>Appendix</b> .
blank.bin	0x1FE000	Initializes Flash system parameter area, more details in <b>Appendix</b> .
boot.bin	0x00000	In <b>/bin/at</b> .
user1.2048.new.5.bin	0x01000	In <b>/bin/at/1024+1024</b> .

### 1.2.5. 32 Mbit Flash, Map: 512 KB + 512 KB

Use Espressif Flash download tool and select flash size: 32 Mbit.

BIN	Address	Description
blank.bin	0x3FB000	Initializes RF_CAL parameter area.
esp_init_data_default.bin	0x3FC000	Stores default RF parameter values, has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this bin has to be downloaded too.
blank.bin	0x7E000	Initializes Flash user parameter area, more details in <b>Appendix</b> .
blank.bin	0x3FE000	Initializes Flash system parameter area, more details in <b>Appendix</b> .
boot.bin	0x00000	In <b>/bin/at</b> .
user1.1024.new.2.bin	0x01000	In <b>/bin/at/512+512</b> .

### 1.2.6. 32 Mbit Flash, Map: 1024 KB + 1024 KB

Use Espressif Flash download tool and select flash size: 32 Mbit-C1.

BIN	Address	Description
blank.bin	0x3FB000	Initializes RF_CAL parameter area
esp_init_data_default.bin	0x3FC000	Stores default RF parameter values, has to be downloaded into flash at least once. If the RF_CAL parameter area is initialized, this bin has to be downloaded too.
blank.bin	0xFE000	Initializes Flash user parameter area, more details in <b>Appendix</b> .
blank.bin	0x3FE000	Initializes Flash system parameter area, more details in <b>Appendix</b> .



BIN	Address	Description
boot.bin	0x00000	In <i>/bin/at</i> .
user1.2048.new.5.bin	0x01000	In <i>/bin/at/1024+1024</i> .

**! Notes:**

- Please make sure that correct BIN (*IESP8266\_NONOS\_SDK/bin/at*) is already in the chip (ESP8266) before using the AT commands listed in this document.
- AT firmware uses priority levels 0 and 1 of `system_os_task`, so only one task of priority 2 is allowed to be set up by the user.
- AT returns messages below to show status of the ESP8266 Station's Wi-Fi connection.
  - ▶ Wi-Fi CONNECTED: *Wi-Fi is connected.*
  - ▶ Wi-Fi GOT IP: *the ESP8266 Station has got the IP from the AP.*
  - ▶ Wi-Fi DISCONNECT: *Wi-Fi is disconnected.*



# 2. Command Description

Each command set contains four types of AT commands.

Type	Command Format	Description
Test Command	AT+<x>=?	Queries the Set Commands' internal parameters and their range of values.
Query Command	AT+<x>?	Returns the current value of parameters.
Set Command	AT+<x>=<...>	Sets the value of user-defined parameters in commands, and runs these commands.
Execute Command	AT+<x>	Runs commands with no user-defined parameters.

**⚠ Notice:**

- *Not all AT commands support all four variations mentioned above.*
- *Square brackets [ ] designate the default value; it is either not required or may not appear.*
- *String values need to be included in double quotation marks, for example: AT+CWSAP="ESP756290", "21030826", 1,4.*
- *The default baud rate is 115200.*
- *AT commands have to be capitalized, and must end with a new line (CR LF).*



# 3. Basic AT Commands

## 3.1. Overview

Commands	Description
AT	Tests AT startup.
AT+RST	Restarts the module.
AT+GMR	Checks version information.
AT+GSLP	Enters Deep-sleep mode.
ATE	Configures echoing of AT commands.
AT+RESTORE	Restores the factory default settings of the module.
AT+UART	UART configuration. [ <a href="#">@deprecated</a> ]
AT+UART_CUR	The current UART configuration.
AT+UART_DEF	The default UART configuration, saved in flash.
AT+SLEEP	Configures the sleep modes.
AT+WAKEUPGPIO	Configures a GPIO to wake ESP8266 up from Light-sleep mode.
AT+RFPOWER	Sets the maximum value of the RF TX Power.
AT+RFVDD	Sets the RF TX Power according to VDD33.
AT+RFAUTOTRACE	Sets RF frequency offset trace.
AT+SYSRAM	Checks the available RAM size.
AT+SYSADC	Checks the ADC value.
AT+SYSIOSETCFG	Sets configuration of IO pins.
AT+SYSIOGETCFG	Gets configuration of IO pins.
AT+SYSGPIODIR	Configures the direction of GPIO.
AT+SYSGPIOWRITE	Configures the GPIO output level
AT+SYSGPIOREAD	Checks the GPIO input level.
AT+SYSMSG_CUR	Sets current system messages.



AT+SYMSG\_DEF

Sets default system messages.

## 3.2. Commands

### 3.2.1. AT – Tests AT Startup

Execute Command	AT
Response	OK
Parameters	-

### 3.2.2. AT+RST – Restarts the Module

Execute Command	AT+RST
Response	OK
Parameters	-

### 3.2.3. AT+GMR – Checks Version Information

Execute Command	AT+GMR
Response	<AT version info> <SDK version info> <compile time>  OK
Parameters	<ul style="list-style-type: none"> <li>• &lt;AT version info&gt;: information about the AT version.</li> <li>• &lt;SDK version info&gt;: information about the SDK version.</li> <li>• &lt;compile time&gt;: the duration of time for compiling the BIN.</li> </ul>

### 3.2.4. AT+GSLP – Enters Deep-sleep Mode

Set Command	AT+GSLP=<time>
Response	<time> OK
Parameters	<time>: the duration of ESP8266's sleep. Unit: ms. ESP8266 will wake up after Deep-sleep for as many milliseconds (ms) as <time> indicates.
Note	A minor adjustment has to be made before the module enter the Deep-sleep mode, i.e., connecting XPD_DCDC to EXT_RSTB via a 0-ohm resistor.



### 3.2.5. ATE—AT Commands Echoing

<b>Execute Command</b>	ATE
<b>Response</b>	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• ATE0: Switches echo off.</li> <li>• ATE1: Switches echo on.</li> </ul>
<b>Note</b>	This command ATE is used to trigger command echo. It means that entered commands can be echoed back to the sender when ATE command is used. Two parameters are possible. The command returns OK in normal cases and ERROR when a parameter other than 0 or 1 was specified.

### 3.2.6. AT+RESTORE—Restores the Factory Default Settings

<b>Execute Command</b>	AT+RESTORE
<b>Response</b>	OK
<b>Note</b>	The execution of this command will reset all parameters saved in flash, and restore the factory default settings of the module. The chip will be restarted when this command is executed.

### 3.2.7. AT+UART—UART Configuration

[@deprecated] This command is deprecated. Please use AT+UART\_CUR or AT+UART\_DEF instead.

<b>Command</b>	Query Command: AT+UART?	Set Command: AT+UART=<baudrate>,<databits>,<stopbits>,<parity>,<flow control>
<b>Response</b>	+UART:<baudrate>,<databits>,<stopbits>,<parity>,<flow control> OK	OK
<b>Note</b>	Command AT+UART? will return the actual value of UART configuration parameters, which may have allowable errors compared with the set value.  For example, if the UART baud rate is set as 115200, the baud rate returned by using command AT+UART? could be 115273.	-



<b>Parameters</b>	<ul style="list-style-type: none"><li>• &lt;baudrate&gt;: UART baud rate</li><li>• &lt;databits&gt;: data bits<ul style="list-style-type: none"><li>▶ 5: 5-bit data</li><li>▶ 6: 6-bit data</li><li>▶ 7: 7-bit data</li><li>▶ 8: 8-bit data</li></ul></li><li>• &lt;stopbits&gt;: stop bits<ul style="list-style-type: none"><li>▶ 1: 1-bit stop bit</li><li>▶ 2: 1.5-bit stop bit</li><li>▶ 3: 2-bit stop bit</li></ul></li><li>• &lt;parity&gt;: parity bit<ul style="list-style-type: none"><li>▶ 0: None</li><li>▶ 1: Odd</li><li>▶ 2: Even</li></ul></li><li>• &lt;flow control&gt;: flow control<ul style="list-style-type: none"><li>▶ 0: flow control is not enabled</li><li>▶ 1: enable RTS</li><li>▶ 2: enable CTS</li><li>▶ 3: enable both RTS and CTS</li></ul></li></ul>
<b>Notes</b>	<ol style="list-style-type: none"><li>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.</li><li>2. The use of flow control requires the support of hardware:<ul style="list-style-type: none"><li>▶ MTCK is UART0 CTS</li><li>▶ MTDO is UART0 RTS</li></ul></li><li>3. The range of baud rates supported: 110~115200*40.</li></ol>
<b>Example</b>	AT+UART=115200,8,1,0,3





## 3.2.8. AT+UART\_CUR—Current UART Configuration; Not Saved in the Flash

<b>Command</b>	Query Command: AT+UART_CUR?	Set Command: AT+UART_CUR=<baudrate>,<databits>,<stopbits>,<parity>,<flow control>
<b>Response</b>	+UART_CUR:<baudrate>,<databits>,<stopbits>,<parity>,<flow control> OK	OK
<b>Note</b>	Command AT+UART_CUR? will return the actual value of UART configuration parameters, which may have allowable errors compared with the set value because of the clock division.  For example, if the UART baud rate is set as 115200, the baud rate returned by using command AT+UART_CUR? could be 115273.	-
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;baudrate&gt;: UART baud rate</li> <li>• &lt;databits&gt;: data bits <ul style="list-style-type: none"> <li>▶ 5: 5-bit data</li> <li>▶ 6: 6-bit data</li> <li>▶ 7: 7-bit data</li> <li>▶ 8: 8-bit data</li> </ul> </li> <li>• &lt;stopbits&gt;: stop bits <ul style="list-style-type: none"> <li>▶ 1: 1-bit stop bit</li> <li>▶ 2: 1.5-bit stop bit</li> <li>▶ 3: 2-bit stop bit</li> </ul> </li> <li>• &lt;parity&gt;: parity bit <ul style="list-style-type: none"> <li>▶ 0: None</li> <li>▶ 1: Odd</li> <li>▶ 2: Even</li> </ul> </li> <li>• &lt;flow control&gt;: flow control <ul style="list-style-type: none"> <li>▶ 0: flow control is not enabled</li> <li>▶ 1: enable RTS</li> <li>▶ 2: enable CTS</li> <li>▶ 3: enable both RTS and CTS</li> </ul> </li> </ul>	
<b>Notes</b>	<ol style="list-style-type: none"> <li>1. The configuration changes will NOT be saved in the flash.</li> <li>2. The use of flow control requires the support of hardware: <ul style="list-style-type: none"> <li>▶ MTCK is UART0 CTS</li> <li>▶ MTDO is UART0 RTS</li> </ul> </li> <li>3. The range of baud rates supported: 110~115200*40.</li> </ol>	
<b>Example</b>	AT+UART_CUR=115200,8,1,0,3	



## 3.2.9. AT+UART\_DEF—Default UART Configuration; Saved in the Flash

<b>Command</b>	Query Command: AT+UART_DEF?	Set Command: AT+UART_DEF=<baudrate>,<databits>,<stopbits>,<parity>,<flow control>
<b>Response</b>	+UART_DEF:<baudrate>,<databits>,<stopbits>,<parity>,<flow control> OK	OK
<b>Parameter</b>	<ul style="list-style-type: none"> <li>• &lt;baudrate&gt;: UART baud rate</li> <li>• &lt;databits&gt;: data bits <ul style="list-style-type: none"> <li>▶ 5: 5-bit data</li> <li>▶ 6: 6-bit data</li> <li>▶ 7: 7-bit data</li> <li>▶ 8: 8-bit data</li> </ul> </li> <li>• &lt;stopbits&gt;: stop bits <ul style="list-style-type: none"> <li>▶ 1: 1-bit stop bit</li> <li>▶ 2: 1.5-bit stop bit</li> <li>▶ 3: 2-bit stop bit</li> </ul> </li> <li>• &lt;parity&gt;: parity bit <ul style="list-style-type: none"> <li>▶ 0: None</li> <li>▶ 1: Odd</li> <li>▶ 2: Even</li> </ul> </li> <li>• &lt;flow control&gt;: flow control <ul style="list-style-type: none"> <li>▶ 0: flow control is not enabled</li> <li>▶ 1: enable RTS</li> <li>▶ 2: enable CTS</li> <li>▶ 3: enable both RTS and CTS</li> </ul> </li> </ul>	
<b>Notes</b>	<ol style="list-style-type: none"> <li>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.</li> <li>2. The use of flow control requires the support of hardware: <ul style="list-style-type: none"> <li>▶ MTCK is UART0 CTS</li> <li>▶ MTDO is UART0 RTS</li> </ul> </li> <li>3. The range of baud rates supported: 110~115200*40.</li> </ol>	
<b>Example</b>	AT+UART_DEF=115200,8,1,0,3	



### 3.2.10. AT+SLEEP—Configures the Sleep Modes

<b>Command</b>	Query Command: AT+SLEEP?	Set Command: AT+SLEEP=<sleep mode>
<b>Response</b>	+SLEEP:<sleep mode>  OK	OK
<b>Parameter</b>	<sleep mode>: <ul style="list-style-type: none"> <li>▶ 0: disables sleep mode</li> <li>▶ 1: Light-sleep mode</li> <li>▶ 2: Modem-sleep mode</li> </ul>	
<b>Notes</b>	This command can only be used in Station mode. Modem-sleep is the default sleep mode.	
<b>Example</b>	AT+SLEEP=0	

### 3.2.11. AT+WAKEUPGPIO—Configures a GPIO to Wake ESP8266 up from Light-sleep Mode

<b>Command</b>	AT+WAKEUPGPIO=<enable>,<trigger_GPIO>,<trigger_level>[,<awake_GPIO>,<awake_level>]
<b>Response</b>	OK
<b>Parameter</b>	<ul style="list-style-type: none"> <li>• &lt;enable&gt; <ul style="list-style-type: none"> <li>▶ 0: ESP8266 can NOT be woken up from light-sleep by GPIO.</li> <li>▶ 1: ESP8266 can be woken up from light-sleep by GPIO.</li> </ul> </li> <li>• &lt;trigger_GPIO&gt; <ul style="list-style-type: none"> <li>▶ Sets the GPIO to wake ESP8266 up; range of value: [0, 15].</li> </ul> </li> <li>• &lt;trigger_level&gt; <ul style="list-style-type: none"> <li>▶ 0: The GPIO wakes up ESP8266 on low level.</li> <li>▶ 1: The GPIO wakes up ESP8266 on high level.</li> </ul> </li> <li>• [&lt;awake_GPIO&gt;] <ul style="list-style-type: none"> <li>▶ Optional; this parameter is used to set a GPIO as a flag of ESP8266's being awoken from Light-sleep; range of value: [0, 15].</li> </ul> </li> <li>• [&lt;awake_level&gt;] <ul style="list-style-type: none"> <li>▶ Optional;</li> <li>▶ 0: The GPIO is set to be low level after the wakeup process.</li> <li>▶ 1: The GPIO is set to be high level after the wakeup process.</li> </ul> </li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The value of &lt;trigger_GPIO&gt; and &lt;awake_GPIO&gt; in the command should not be the same.</li> <li>• After being woken up by &lt;trigger_GPIO&gt; from Light-sleep, when the ESP8266 attempts to sleep again, it will check the status of the &lt;trigger_GPIO&gt;: <ul style="list-style-type: none"> <li>▶ if it is still in the wakeup status, the EP8266 will enter Modem-sleep mode instead;</li> <li>▶ if it is NOT in the wakeup status, the ESP8266 will enter Light-sleep mode.</li> </ul> </li> </ul>



<b>Example</b>	<ul style="list-style-type: none"> <li>Set ESP8266 to be woken from Light-sleep, when GPIO0 is on low level: AT+WAKEUPGPIO=1,0,0</li> <li>Set ESP8266 to be woken from Light-sleep, when GPIO0 is on high level. After the waking-up, GPIO13 is set to high level. AT+WAKEUPGPIO=1,0,1,13,1</li> <li>Disable the function that ESP8266 can be woken up from Light-sleep by a GPIO. AT+WAKEUPGPIO=0</li> </ul>
----------------	---

### 3.2.12. AT+RFPOWER—Sets the Maximum Value of RF TX Power

<b>Set Command</b>	AT+RFPOWER=<TX Power>
<b>Response</b>	OK
<b>Parameter</b>	<TX Power>: the maximum value of RF TX power; range: [0, 82]; unit: 0.25 dBm.
<b>Note</b>	This command sets the maximum value of ESP8266 RF TX power; it is not precise. The actual value could be smaller than the set value.
<b>Example</b>	AT+RFPOWER=50

### 3.2.13. AT+RFVDD—Sets RF TX Power According to VDD33

<b>Command</b>	Query Command: AT+RFVDD? Function: Checks the value of ESP8266 VDD33.	Set Command: AT+RFVDD=<VDD33> Function: Sets the RF TX Power according to <VDD33>.	Execute Command: AT+RFVDD Function: Automatically sets the RF TX Power.
<b>Response</b>	+RFVDD:<VDD33> OK	OK	OK
<b>Parameter</b>	<VDD33>: power voltage of ESP8266 VDD33; unit: 1/1024 V.	<VDD33>: power voltage of ESP8266 VDD33 ; range: [1900, 3300].	-
<b>Note</b>	The command should only be used when TOUT pin has to be suspended, or else the returned value would be invalid.	-	TOUT pin has to be suspended in order to measure VDD33.
<b>Example</b>	AT+RFVDD=2800		



### 3.2.14. AT+RFAUTOTRACE—Sets RF Frequency Offset Trace

<b>Command</b>	Query Command: AT+RFAUTOTRACE?	Set Command: AT+RFAUTOTRACE=<enable>
<b>Response</b>	+RFAUTOTRACE:<enable> OK	OK
<b>Parameter</b>	<enable>: <ul style="list-style-type: none"> <li>▶ 0: disables RF frequency offset trace</li> <li>▶ 1: enables RF frequency offset trace</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The RF frequency offset trace function is enabled by default.</li> <li>• This configuration will be saved in the user parameter area in flash, and take effect after the chip restarts.</li> </ul>	
<b>Example</b>	AT+RFAUTOTRACE=0 AT+RST	

### 3.2.15. AT+SYSRAM—Checks the Remaining Space of RAM

<b>Query Command</b>	AT+SYSRAM?
<b>Response</b>	+SYSRAM:<remaining RAM size> OK
<b>Parameter</b>	<remaining RAM size>: remaining space of RAM, unit: byte.

### 3.2.16. AT+SYSADC—Checks the Value of ADC

<b>Query Command</b>	AT+SYSADC?
<b>Response</b>	+SYSADC:<ADC> OK
<b>Parameter</b>	<ADC>: the value of ADC; unit: 1/1024V.



### 3.2.17. AT+SYSIOSETCFG—Configures IO Working Mode

Set Command	AT+SYSIOSETCFG=<pin>,<mode>,<pull-up>
Response	OK
Parameter	<ul style="list-style-type: none"><li>• &lt;pin&gt;: number of an IO pin</li><li>• &lt;mode&gt;: the working mode of the IO pin</li><li>• &lt;pull-up&gt;<ul style="list-style-type: none"><li>▶ 0: disable the pull-up</li><li>▶ 1: enable the pull-up of the IO pin</li></ul></li></ul>
Note	Please refer to <a href="#">ESP8266 Pin List</a> for uses of AT+SYSIO-related commands.
Example	AT+SYSIOSETCFG=12,3,1 //Set GPIO12 to work as a GPIO

### 3.2.18. AT+SYSIOGETCFG—Checks the Working Modes of IO Pins

Set Command	AT+SYSIOGETCFG=<pin>
Response	+SYSIOGETCFG:<pin>,<mode>,<pull-up> OK
Parameter	<ul style="list-style-type: none"><li>• &lt;pin&gt;: number of an IO pin</li><li>• &lt;mode&gt;: the working mode of the IO pin</li><li>• &lt;pull-up&gt;<ul style="list-style-type: none"><li>▶ 0: disable the pull-up</li><li>▶ 1: enable the pull-up of the IO pin</li></ul></li></ul>
Note	Please refer to <a href="#">ESP8266 Pin List</a> for uses of AT+SYSIO-related commands.

### 3.2.19. AT+SYSGPIODIR—Configures the Direction of a GPIO

Set Command	AT+SYSGPIODIR=<pin>,<dir>
Response	<ul style="list-style-type: none"><li>• If the configuration is successful, the command will return: OK</li><li>• If the IO pin is not in GPIO mode, the command will return: NOT GPIO MODE! ERROR</li></ul>
Parameter	<ul style="list-style-type: none"><li>• &lt;pin&gt;: GPIO pin number</li><li>• &lt;dir&gt;:<ul style="list-style-type: none"><li>▶ 0: sets the GPIO as an input</li><li>▶ 1: sets the GPIO as an output</li></ul></li></ul>
Note	Please refer to <a href="#">ESP8266 Pin List</a> for uses of AT+SYSGPIO-related commands.



Example	<pre>AT+SYSIOSETCFG=12,3,1 //Set GPIO12 to work as a GPIO AT+SYSGPIODIR=12,0 //Set GPIO12 to work as an input</pre>
---------	---

### 3.2.20. AT+SYSGPIOWRITE—Configures the Output Level of a GPIO

Set Command	AT+SYSGPIOWRITE=<pin>,<level>
Response	<ul style="list-style-type: none"> <li>If the configuration is successful, the command will return: OK</li> <li>If the IO pin is not in output mode, the command will return: NOT OUTPUT! ERROR</li> </ul>
Parameter	<ul style="list-style-type: none"> <li>&lt;pin&gt;: GPIO pin number</li> <li>&lt;level&gt;: <ul style="list-style-type: none"> <li>▶ 0: low level</li> <li>▶ 1: high level</li> </ul> </li> </ul>
Note	Please refer to <a href="#">ESP8266 Pin List</a> for uses of AT+SYSGPIO-related commands.
Example	<pre>AT+SYSIOSETCFG=12,3,1 //Set GPIO12 to work as a GPIO AT+SYSGPIODIR=12,1 //Set GPIO12 to work as an output AT+SYSGPIOWRITE=12,1 //Set GPIO12 to output high level</pre>

### 3.2.21. AT+SYSGPIOREAD—Reads the GPIO Input Level

Set Command	AT+SYSGPIOREAD=<pin>
Response	<ul style="list-style-type: none"> <li>If the configuration is successful, the command returns: +SYSGPIOREAD:&lt;pin&gt;,&lt;dir&gt;,&lt;level&gt; OK</li> <li>If the IO pin is not in GPIO mode, the command will return: NOT GPIO MODE! ERROR</li> </ul>
Parameter	<ul style="list-style-type: none"> <li>&lt;pin&gt;: GPIO pin number</li> <li>&lt;dir&gt;: <ul style="list-style-type: none"> <li>▶ 0: sets the GPIO as an input</li> <li>▶ 1: sets the GPIO as an output</li> </ul> </li> <li>&lt;level&gt;: <ul style="list-style-type: none"> <li>▶ 0: low level</li> <li>▶ 1: high level</li> </ul> </li> </ul>



<b>Note</b>	Please refer to <a href="#">ESP8266 Pin List</a> for uses of AT+SYSGPIO-related commands.
<b>Example</b>	<pre>AT+SYSIOSETCFG=12,3,1 //Set GPIO12 to work as a GPIO AT+SYSGPIODIR=12,0 //Set GPIO12 to work as an input AT+SYSGPIOREAD=12</pre>

### 3.2.22. AT+SYMSMSG\_CUR—Set Current System Messages

<b>Set Command</b>	AT+SYMSMSG_CUR=<n>
<b>Response</b>	OK
<b>Parameter</b>	<p>&lt;n&gt;:</p> <ul style="list-style-type: none"> <li>• bit0: configure the message of quitting from WiFi-UART passthrough transmission <ul style="list-style-type: none"> <li>▶ if the bit0 is 0, there is no message when quitting from WiFi-UART passthrough transmission; default is 0</li> <li>▶ if the bit0 is 1, when quitting from WiFi-UART passthrough transmission, it will prompt message +QUITTT // Quit transparent transmission</li> </ul> </li> <li>• bit1: configure the message of establishing a network transmission <ul style="list-style-type: none"> <li>▶ if the bit1 is 0, when a network connection is established, it will prompt message &lt;Link_ID&gt;,CONNECT; default is 0</li> <li>▶ if the bit1 is 1, when establishing a network connection, it will prompt message +LINK_CONN:&lt;status_type&gt;,&lt;link_id&gt;,"UDP/TCP/SSL",&lt;c/s&gt;,&lt;remote_ip&gt;,&lt;remote_port&gt;,&lt;local_port&gt;; <ul style="list-style-type: none"> <li>- &lt;status_type&gt; : 0 - the connection is established successfully;</li> <li>1 - fail to establish the connection</li> <li>- &lt;c/s&gt; : 0 - the ESP works as a client; 1 - the ESP works as a server</li> </ul> </li> </ul> </li> </ul>
<b>Note</b>	The configuration changes will NOT be saved in the flash.
<b>Example</b>	AT+SYMSMSG_CUR=3

### 3.2.23. AT+SYMSMSG\_DEF—Set Default System Messages

<b>Set Command</b>	AT+SYMSMSG_DEF=<n>
<b>Response</b>	OK





<b>Parameter</b>	<p>&lt;n&gt;:</p> <ul style="list-style-type: none"><li>• bit0: configure the message of quitting from WiFi-UART passthrough transmission<ul style="list-style-type: none"><li>▶ if the bit0 is 0, there is no message when quitting from WiFi-UART passthrough transmission; default is 0</li><li>▶ if the bit0 is 1, when quitting from WiFi-UART passthrough transmission, it will prompt message +QUIT // Quit transparent transmission</li></ul></li><li>• bit1: configure the message of establishing a network transmission<ul style="list-style-type: none"><li>▶ if the bit1 is 0, when a network connection is established, it will prompt message &lt;Link_ID&gt;,CONNECT; default is 0</li><li>▶ if the bit1 is 1, when establishing a network connection, it will prompt message +LINK_CONN:&lt;status_type&gt;,&lt;link_id&gt;,"UDP/TCP/SSL",&lt;c/s&gt;,&lt;remote_ip&gt;,&lt;remote_port&gt;,&lt;local_port&gt;;<ul style="list-style-type: none"><li>- &lt;status_type&gt; : 0 - the connection is established successfully; 1 - fail to establish the connection</li><li>- &lt;c/s&gt; : 0 - the ESP works as a client; 1 - the ESP works as a server</li></ul></li></ul></li></ul>
<b>Note</b>	The configuration changes will be saved in the flash user parameter area.
<b>Example</b>	AT+SYMSG_DEF=3



# 4. Wi-Fi AT Commands

## 4.1. Overview

Commands	Description
AT+CWMODE	Sets the Wi-Fi mode (Station/AP/Station+AP). <a href="#">[@deprecated]</a>
AT+CWMODE_CUR	Sets the Wi-Fi mode (Station/AP/Station+AP); configuration not saved in the flash.
AT+CWMODE_DEF	Sets the default Wi-Fi mode (Station/AP/Station+AP); configuration saved in the flash.
AT+CWJAP	Connect to an AP. <a href="#">[@deprecated]</a>
AT+CWJAP_CUR	Connects to an AP; configuration not saved in the flash.
AT+CWJAP_DEF	Connects to an AP; configuration saved in the flash.
AT+CWLAPOPT	Sets the configuration of command AT+CWLAP.
AT+CWLAP	Lists available APs.
AT+CWQAP	Disconnects from an AP.
AT+CWSAP	Sets the configuration of the ESP8266 SoftAP. <a href="#">[@deprecated]</a>
AT+CWSAP_CUR	Sets the current configuration of the ESP8266 SoftAP; configuration not saved in the flash.
AT+CWSAP_DEF	Sets the configuration of the ESP8266 SoftAP; configuration saved in the flash.
AT+CWLIF	Gets the Station IP to which the ESP8266 SoftAP is connected.
AT+CWDHCP	Enables/Disables DHCP. <a href="#">[@deprecated]</a>
AT+CWDHCP_CUR	Enables/Disables DHCP; configuration not saved in the flash.
AT+CWDHCP_DEF	Enable/Disable DHCP; configuration saved in the flash.
AT+CWDHCPS_CUR	Sets the IP range of the DHCP server; configuration not saved in the flash.
AT+CWDHCPS_DEF	Sets the IP range of the DHCP server; configuration saved in the flash.
AT+CWAUTOCONN	Connects to an AP automatically on power-up.
AT+CIPSTAMAC	Sets the MAC address of the ESP8266 Station. <a href="#">[@deprecated]</a>
AT+CIPSTAMAC_CUR	Sets the MAC address of the ESP8266 Station; configuration not saved in the flash.
AT+CIPSTAMAC_DEF	Sets the MAC address of ESP8266 station; configuration saved in the flash.



AT+CIPAPMAC	Sets the MAC address of the ESP8266 SoftAP. <a href="#">[@deprecated]</a>
AT+CIPAPMAC_CUR	Sets the MAC address of the ESP8266 SoftAP; configuration not saved in the flash.
AT+CIPAPMAC_DEF	Sets the MAC address of the ESP8266 SoftAP; configuration saved in the flash.
AT+CIPSTA	Sets the IP address of the ESP8266 Station. <a href="#">[@deprecated]</a>
AT+CIPSTA_CUR	Sets the IP address of the ESP8266 Station; configuration not saved in the flash.
AT+CIPSTA_DEF	Sets the IP address of the ESP8266 Station; configuration saved in the flash.
AT+CIPAP	Sets the IP address of ESP8266 SoftAP. <a href="#">[@deprecated]</a>
AT+CIPAP_CUR	Sets the IP address of ESP8266 SoftAP; configuration not saved in the flash.
AT+CIPAP_DEF	Sets the IP address of ESP8266 SoftAP; configuration saved in the flash.
AT+CWSTARTSMART	Starts SmartConfig.
AT+CWSTOPSMART	Stops SmartConfig.
AT+CWSTARTDISCOVER	Enables the mode that ESP8266 can be found by WeChat.
AT+CWSTOPDISCOVER	Disables the mode that ESP8266 can be found by WeChat.
AT+WPS	Sets the WPS function.
AT+MDNS	Sets the MDNS function.
AT+CWHOOSTNAME	Sets the host name of the ESP8266 Station.
AT+CWCOUNTRY_CUR	Sets current WiFi country code
AT+CWCOUNTRY_DEF	Sets default WiFi country code



## 4.2. Commands

### 4.2.1. AT+CWMODE—Sets the Wi-Fi Mode (Station/SoftAP/Station+SoftAP)

[@deprecated] This command is deprecated. Please use AT+CWMODE\_CUR or AT+CWMODE\_DEF instead.

<b>Commands</b>	Test Command: AT+CWMODE=?	Query Command: AT+CWMODE?  Function: to query the current Wi-Fi mode of ESP8266.	Set Command: AT+CWMODE=<mode>  Function: to set the current Wi-Fi mode of ESP8266.
<b>Response</b>	+CWMODE : <mode> OK	+CWMODE : <mode> OK	OK
<b>Parameters</b>	<mode>: <ul style="list-style-type: none"> <li>▶ 1: Station mode</li> <li>▶ 2: SoftAP mode</li> <li>▶ 3: SoftAP+Station mode</li> </ul>		
<b>Note</b>	The configuration changes will be saved in the system parameter area in the flash.		
<b>Example</b>	AT+CWMODE=3		

### 4.2.2. AT+CWMODE\_CUR—Sets the Current Wi-Fi mode; Configuration Not Saved in the Flash

<b>Commands</b>	Test Command: AT+CWMODE_CUR=?	Query Command: AT+CWMODE_CUR?  Function: to query the current Wi-Fi mode of ESP8266.	Set Command: AT+CWMODE_CUR=<mode>  Function: to set the current Wi-Fi mode of ESP8266.
<b>Response</b>	+CWMODE_CUR : <mode> OK	+CWMODE_CUR : <mode> OK	OK
<b>Parameters</b>	<mode>: <ul style="list-style-type: none"> <li>▶ 1: Station mode</li> <li>▶ 2: SoftAP mode</li> <li>▶ 3: SoftAP+Station mode</li> </ul>		
<b>Note</b>	The configuration changes will NOT be saved in the flash.		
<b>Example</b>	AT+CWMODE_CUR=3		

### 4.2.3. AT+CWMODE\_DEF—Sets the Default Wi-Fi mode; Configuration Saved in the Flash

<b>Commands</b>	Test Command: AT+CWMODE_DEF=?	Query Command: AT+CWMODE_DEF?  Function: to query the current Wi-Fi mode of ESP8266.	Set Command: AT+CWMODE_DEF=<mode>  Function: to set the current Wi-Fi mode of ESP8266.
<b>Response</b>	+CWMODE_DEF : <mode> OK	+CWMODE_DEF : <mode> OK	OK



<b>Parameters</b>	<p>&lt;mode&gt;:</p> <ul style="list-style-type: none"> <li>▶ 1: Station mode</li> <li>▶ 2: SoftAP mode</li> <li>▶ 3: SoftAP+Station mode</li> </ul>
<b>Note</b>	The configuration changes will be saved in the system parameter area in the flash.
<b>Example</b>	AT+CWMODE_DEF=3

#### 4.2.4. AT+CWJAP—Connects to an AP

[@deprecated] This command is deprecated. Please use AT+CWJAP\_CUR or AT+CWJAP\_DEF instead.

<b>Commands</b>	<p>Query Command:</p> <p>AT+CWJAP?</p> <p>Function: to query the AP to which the ESP8266 Station is already connected.</p>	<p>Set Command:</p> <p>AT+CWJAP=&lt;ssid&gt;,&lt;pwd&gt;[,&lt;bssid&gt;]</p> <p>Function: to set the AP to which the ESP8266 Station needs to be connected.</p>
<b>Response</b>	<p>+CWJAP:&lt;ssid&gt;,&lt;bssid&gt;,&lt;channel&gt;,&lt;rssi&gt;</p> <p>OK</p>	<p>OK</p> <p>or</p> <p>+CWJAP:&lt;error code&gt;</p> <p>FAIL</p>
<b>Parameters</b>	<p>&lt;ssid&gt;: a string parameter showing the SSID of the target AP.</p>	<ul style="list-style-type: none"> <li>• &lt;ssid&gt;: the SSID of the target AP.</li> <li>• &lt;pwd&gt;: password, MAX: 64-byte ASCII.</li> <li>• [&lt;bssid&gt;]: the target AP's MAC address, used when multiple APs have the same SSID.</li> <li>• &lt;error code&gt;: (for reference only) <ul style="list-style-type: none"> <li>▶ 1: connection timeout.</li> <li>▶ 2: wrong password.</li> <li>▶ 3: cannot find the target AP.</li> <li>▶ 4: connection failed.</li> </ul> </li> </ul> <p>This command requires Station mode to be active. Escape character syntax is needed if SSID or password contains any special characters, such as , or " or \.</p>
<b>Note</b>	The configuration changes will be saved in the system parameter area in the flash.	
<b>Examples</b>	<p>AT+CWJAP="abc", "0123456789"</p> <p>For example, if the target AP's SSID is "ab\,c" and the password is "0123456789"\", the command is as follows:</p> <p>AT+CWJAP="ab\\\",c", "0123456789\\""</p> <p>If multiple APs have the same SSID as "abc", the target AP can be found by BSSID:</p> <p>AT+CWJAP="abc", "0123456789", "ca:d7:19:d8:a6:44"</p>	



#### 4.2.5. AT+CWJAP\_CUR—Connects to an AP; Configuration Not Saved in the Flash

<b>Commands</b>	<p>Query Command:</p> <p>AT+CWJAP_CUR?</p> <p>Function: to query the AP to which the ESP8266 Station is already connected.</p>	<p>Set Command:</p> <p>AT+CWJAP_CUR=&lt;ssid&gt;,&lt;pwd&gt;[,&lt;bssid&gt;]</p> <p>Function: to set the AP to which the ESP8266 Station needs to be connected.</p>
<b>Response</b>	<p>+CWJAP_CUR:&lt;ssid&gt;,&lt;bssid&gt;,&lt;channel&gt;,&lt;rssi&gt;</p> <p>OK</p>	<p>OK</p> <p>or</p> <p>+CWJAP_CUR:&lt;error code&gt;</p> <p>FAIL</p>
<b>Parameters</b>	<p>&lt;ssid&gt;: a string parameter showing the SSID of the target AP.</p>	<ul style="list-style-type: none"> <li>• &lt;ssid&gt;: the SSID of the target AP.</li> <li>• &lt;pwd&gt;: password, MAX: 64-byte ASCII.</li> <li>• [&lt;bssid&gt;]: the target AP's MAC address, used when multiple APs have the same SSID.</li> <li>• &lt;error code&gt;: (for reference only) <ul style="list-style-type: none"> <li>▶ 1: connection timeout.</li> <li>▶ 2: wrong password.</li> <li>▶ 3: cannot find the target AP.</li> <li>▶ 4: connection failed.</li> </ul> </li> </ul> <p>This command requires Station mode to be active. Escape character syntax is needed if SSID or password contains any special characters, such as , or " or \.</p>
<b>Note</b>	The configuration changes will NOT be saved in the flash.	
<b>Examples</b>	<p>AT+CWJAP_CUR="abc", "0123456789"</p> <p>For example, if the target AP's SSID is "ab\,c" and the password is "0123456789\", the command is as follows:</p> <p>AT+CWJAP_CUR="ab\\,c", "0123456789\\""</p> <p>If multiple APs have the same SSID as "abc", the target AP can be found by BSSID:</p> <p>AT+CWJAP_CUR="abc", "0123456789", "ca:d7:19:d8:a6:44"</p>	

#### 4.2.6. AT+CWJAP\_DEF—Connects to an AP; Configuration Saved in the Flash

<b>Commands</b>	<p>Query Command:</p> <p>AT+CWJAP_DEF?</p> <p>Function: to query the AP to which the ESP8266 Station is already connected.</p>	<p>Set Command:</p> <p>AT+CWJAP_DEF=&lt;ssid&gt;,&lt;pwd&gt;[,&lt;bssid&gt;]</p> <p>Function: to set the AP to which the ESP8266 Station needs to be connected.</p>
<b>Response</b>	<p>+CWJAP_DEF:&lt;ssid&gt;,&lt;bssid&gt;,&lt;channel&gt;,&lt;rssi&gt;</p> <p>OK</p>	<p>OK</p> <p>or</p> <p>+CWJAP_DEF:&lt;error code&gt;</p> <p>FAIL</p>



<b>Parameters</b>	<p>&lt;ssid&gt;: a string parameter showing the SSID of the target AP.</p>	<ul style="list-style-type: none"><li>• &lt;ssid&gt;: the SSID of the target AP.</li><li>• &lt;pwd&gt;: password, MAX: 64-byte ASCII.</li><li>• [&lt;bssid&gt;]: the target AP's MAC address, used when multiple APs have the same SSID.</li><li>• &lt;error code&gt;: (for reference only)<ul style="list-style-type: none"><li>▶ 1: connection timeout.</li><li>▶ 2: wrong password.</li><li>▶ 3: cannot find the target AP.</li><li>▶ 4: connection failed.</li></ul></li></ul> <p>This command requires Station mode to be active. Escape character syntax is needed if SSID or password contains any special characters, such as , or " or \.</p>
<b>Note</b>	The configuration changes will be saved in the system parameter area in the flash.	
<b>Examples</b>	<pre>AT+CWJAP_DEF="abc", "0123456789"</pre> <p>For example, if the target AP's SSID is "ab\,c" and the password is "0123456789\", the command is as follows:</p> <pre>AT+CWJAP_DEF="ab\\,c", "0123456789\\""</pre> <p>If multiple APs have the same SSID as "abc", the target AP can be found by BSSID:</p> <pre>AT+CWJAP_DEF="abc", "0123456789", "ca:d7:19:d8:a6:44"</pre>	



### 4.2.7. AT+CWLAPOPT—Sets the Configuration for the Command AT+CWLAP

Set Command	AT+CWLAPOPT=<sort_enable>,<mask>
Response	OK or ERROR
Parameters	<ul style="list-style-type: none"> <li>• &lt;sort_enable&gt;: determines whether the result of command AT+CWLAP will be listed according to RSSI: <ul style="list-style-type: none"> <li>▶ 0: the result is ordered according to RSSI.</li> <li>▶ 1: the result is not ordered according to RSSI.</li> </ul> </li> <li>• &lt;mask&gt;: determines the parameters shown in the result of AT+CWLAP; 0 means not showing the parameter corresponding to the bit, and 1 means showing it. <ul style="list-style-type: none"> <li>▶ bit 0: determines whether &lt;ecn&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 1: determines whether &lt;ssid&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 2: determines whether &lt;rssi&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 3: determines whether &lt;mac&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 4: determines whether &lt;ch&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 5: determines whether &lt;freq_offset&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 6: determines whether &lt;freq_calibration&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 7: determines whether &lt;pairwise_cipher&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 8: determines whether &lt;group_cipher&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 9: determines whether &lt;bgn&gt; will be shown in the result of AT+CWLAP.</li> <li>▶ bit 10: determines whether &lt;wps&gt; will be shown in the result of AT+CWLAP.</li> </ul> </li> </ul>
Example	<p>AT+CWLAPOPT=1,2047</p> <p>The first parameter is 1, meaning that the result of the command AT+CWLAP will be ordered according to RSSI;</p> <p>The second parameter is 2047, namely 0x7FF, meaning that the corresponding bits of &lt;mask&gt; are set to 1. All parameters will be shown in the result of AT+CWLAP.</p>

### 4.2.8. AT+CWLAP—Lists Available APs

Commands	Set Command: AT+CWLAP[=<ssid>,<mac>,<channel>,<scan_type>,<scan_time_min>,<scan_time_max>] Function: to query the APs with specific SSID and MAC on a specific channel.	Execute Command: AT+CWLAP Function: to list all available APs.
Response	+CWLAP:<ecn>,<ssid>,<rssi>,<mac>,<channel>,<freq_offset>,<freq cali>,<pairwise_cipher>,<group_cipher>,<bgn>,<wps> OK	+CWLAP:<ecn>,<ssid>,<rssi>,<mac>,<channel>,<freq_offset>,<freq cali>,<pairwise_cipher>,<group_cipher>,<bgn>,<wps> OK





Parameters	<ul style="list-style-type: none"> <li>• [<code>&lt;scan_type&gt;</code>]: optional parameter <ul style="list-style-type: none"> <li>▶ 0: active scan</li> <li>▶ 1: passive scan</li> </ul> </li> <li>• [<code>&lt;scan_time_min&gt;</code>] : optional parameter, unit: ms, range: [0,1500] <ul style="list-style-type: none"> <li>▶ For active scan mode, <code>&lt;scan_time_min&gt;</code> is the minimum scan time for each channel, default is 0</li> <li>▶ For passive scan mode, <code>&lt;scan_time_min&gt;</code> is meaningless, can be omitted</li> </ul> </li> <li>• [<code>&lt;scan_time_max&gt;</code>] : optional parameter, unit: ms, range: [0,1500] <ul style="list-style-type: none"> <li>▶ For active scan mode, <code>&lt;scan_time_max&gt;</code> is the maximum scan time for each channel; if it is set to be 0, a default value, 120ms will be used.</li> <li>▶ For passive scan mode, <code>&lt;scan_time_max&gt;</code> is the scan time for each channel, default is 360ms</li> </ul> </li> <li>• <code>&lt;ecn&gt;</code>: encryption method. <ul style="list-style-type: none"> <li>▶ 0: OPEN</li> <li>▶ 1: WEP</li> <li>▶ 2: WPA_PSK</li> <li>▶ 3: WPA2_PSK</li> <li>▶ 4: WPA_WPA2_PSK</li> <li>▶ 5: WPA2_Enterprise (AT can NOT connect to WPA2_Enterprise AP for now.)</li> </ul> </li> <li>• <code>&lt;ssid&gt;</code>: string parameter, SSID of the AP.</li> <li>• <code>&lt;rssi&gt;</code>: signal strength.</li> <li>• <code>&lt;mac&gt;</code>: string parameter, MAC address of the AP.</li> <li>• <code>&lt;channel&gt;</code>: channel number</li> <li>• <code>&lt;freq_offset&gt;</code>: frequency offset of AP; unit: KHz. The value of ppm is <code>&lt;freq_offset&gt;/2.4</code>.</li> <li>• <code>&lt;freq_calibration&gt;</code>: calibration for frequency offset.</li> <li>• <code>&lt;pairwise_cipher&gt;</code>: <ul style="list-style-type: none"> <li>▶ 0: CIPHER_NONE</li> <li>▶ 1: CIPHER_WEP40</li> <li>▶ 2: CIPHER_WEP104</li> <li>▶ 3: CIPHER_TKIP</li> <li>▶ 4: CIPHER_CCMP</li> <li>▶ 5: CIPHER_TKIP_CCMP</li> <li>▶ 6: CIPHER_UNKNOWN</li> </ul> </li> <li>• <code>&lt;group_cipher&gt;</code>: definitions of cipher types are same as <code>&lt;pairwise_cipher&gt;</code></li> <li>• <code>&lt;bgn&gt;</code>: <ul style="list-style-type: none"> <li>▶ bit0 is for 802.11b mode; bit1 is for 802.11g mode; bit2 is for 802.11n mode;</li> <li>▶ if the value of the bit is 1, the corresponding 802.11 mode is enabled; if the bit value is 0, the mode is disabled.</li> </ul> </li> <li>• <code>&lt;wps&gt;</code>: 0, WPS is disabled; 1, WPS is enabled</li> </ul>
Examples	<pre>AT+CWLAP="Wi-Fi", "ca:d7:19:d8:a6:44", 6</pre> <p>or search for APs with a designated SSID:</p> <pre>AT+CWLAP="Wi-Fi"</pre> <p>or enable passive scan:</p> <pre>AT+CWLAP=,,1,,</pre>



### 4.2.9. AT+CWQAP—Disconnects from the AP

Execute Command	AT+CWQAP
Response	OK
Parameters	-

### 4.2.10. AT+CWSAP—Configures the ESP8266 SoftAP

[@deprecated] This command is deprecated. Please use AT+CWSAP\_CUR or AT+CWSAP\_DEF instead.

Commands	Query Command: AT+CWSAP?  Function: to obtain the configuration parameters of the ESP8266 SoftAP.	Set Command: AT+CWSAP=<ssid>,<pwd>,<chl>,<ecn>[,<max conn>][,<ssid hidden>]  Function: to configure the ESP8266 SoftAP.
Response	+CWSAP:<ssid>,<pwd>,<chl>,<ecn>,<max conn>,<ssid hidden>	OK or ERROR
Parameters	<ul style="list-style-type: none"> <li>• &lt;ssid&gt;: string parameter, SSID of AP.</li> <li>• &lt;pwd&gt;: string parameter, length of password: 8 ~ 64 bytes ASCII.</li> <li>• &lt;chl&gt;: channel ID.</li> <li>• &lt;ecn&gt;: encryption method; WEP is not supported.             <ul style="list-style-type: none"> <li>▶ 0: OPEN</li> <li>▶ 2: WPA_PSK</li> <li>▶ 3: WPA2_PSK</li> <li>▶ 4: WPA_WPA2_PSK</li> </ul> </li> <li>• [&lt;max conn&gt;] (optional): maximum number of Stations to which ESP8266 SoftAP can be connected; within the range of [1, 10].</li> <li>• [&lt;ssid hidden&gt;] (optional):             <ul style="list-style-type: none"> <li>▶ 0: SSID is broadcasted. (the default setting)</li> <li>▶ 1: SSID is not broadcasted.</li> </ul> </li> </ul>	The same as above.  <b>⚠ Notice:</b>  This command is only available when SoftAP is active.
Note	The configuration changes will be saved in the system parameter area in the flash.	
Example	AT+CWSAP="ESP8266", "1234567890", 5, 3	

### 4.2.11. AT+CWSAP\_CUR—Configures the ESP8266 SoftAP; Configuration Not Saved in the Flash

Commands	Query Command: AT+CWSAP_CUR?  Function: to obtain the configuration parameters of the ESP8266 SoftAP.	Set Command: AT+CWSAP_CUR=<ssid>,<pwd>,<chl>,<ecn>[,<max conn>][,<ssid hidden>]  Function: to configure the ESP8266 SoftAP.
----------	--	--



<b>Response</b>	+CWSAP_CUR:<ssid>,<pwd>,<chl>,<ecn>,<max conn>,<ssid hidden>	OK or ERROR
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;ssid&gt;: string parameter, SSID of AP.</li> <li>• &lt;pwd&gt;: string parameter, length of password: 8 ~ 64 bytes ASCII.</li> <li>• &lt;chl&gt;: channel ID.</li> <li>• &lt;ecn&gt;: encryption method; WEP is not supported. <ul style="list-style-type: none"> <li>▸ 0: OPEN</li> <li>▸ 2: WPA_PSK</li> <li>▸ 3: WPA2_PSK</li> <li>▸ 4: WPA_WPA2_PSK</li> </ul> </li> <li>• [&lt;max conn&gt;] (optional): maximum number of Stations to which ESP8266 SoftAP can be connected; within the range of [1, 10].</li> <li>• [&lt;ssid hidden&gt;] (optional): <ul style="list-style-type: none"> <li>▸ 0: SSID is broadcasted. (the default setting)</li> <li>▸ 1: SSID is not broadcasted.</li> </ul> </li> </ul>	<p><b>⚠ Notice:</b></p> <p>This command is only available when SoftAP is active.</p>
<b>Note</b>	The configuration changes will NOT be saved in the flash.	
<b>Example</b>	AT+CWSAP_CUR="ESP8266", "1234567890", 5, 3	

#### 4.2.12. AT+CWSAP\_DEF—Configures the ESP8266 SoftAP; Configuration Saved in the Flash

<b>Commands</b>	Query Command: AT+CWSAP_DEF? Function: to obtain the configuration parameters of the ESP8266 SoftAP.	Set Command: AT+CWSAP_DEF=<ssid>,<pwd>,<chl>,<ecn>[,<max conn>][,<ssid hidden>] Function: to list all available APs.
<b>Response</b>	+CWSAP_DEF:<ssid>,<pwd>,<chl>,<ecn>,<max conn>,<ssid hidden>	OK or ERROR



Parameters	<ul style="list-style-type: none"> <li>• &lt;ssid&gt;: string parameter, SSID of AP.</li> <li>• &lt;pwd&gt;: string parameter, length of password: 8 ~ 64 bytes ASCII.</li> <li>• &lt;chl&gt;: channel ID.</li> <li>• &lt;ecn&gt;: encryption method; WEP is not supported. <ul style="list-style-type: none"> <li>▶ 0: OPEN</li> <li>▶ 2: WPA_PSK</li> <li>▶ 3: WPA2_PSK</li> <li>▶ 4: WPA_WPA2_PSK</li> </ul> </li> <li>• [&lt;max conn&gt;] (optional): maximum number of Stations to which ESP8266 SoftAP can be connected; within the range of [1, 4].</li> <li>• [&lt;ssid hidden&gt;] (optional): <ul style="list-style-type: none"> <li>▶ 0: SSID is broadcasted. (the default setting)</li> <li>▶ 1: SSID is not broadcasted.</li> </ul> </li> </ul>	<p>The same as above.</p> <p><b>⚠ Notice:</b></p> <p>This command is only available when SoftAP is active.</p>
Note	The configuration changes will NOT be saved in the flash.	
Example	AT+CWSAP_DEF="ESP8266", "1234567890", 5, 3	

#### 4.2.13. AT+CWLIF—IP of Stations to Which the ESP8266 SoftAP is Connected

Execute Command	AT+CWLIF	
Response	<ip addr>, <mac> OK	
Parameters	<ul style="list-style-type: none"> <li>• &lt;ip addr&gt;: IP address of Stations to which ESP8266 SoftAP is connected.</li> <li>• &lt;mac&gt;: MAC address of Stations to which ESP8266 SoftAP is connected.</li> </ul>	
Note	This command cannot get a static IP. It only works when both DHCPs of the ESP8266 SoftAP, and of the Station to which ESP8266 is connected, are enabled.	

#### 4.2.14. AT+CWDHCP—Enables/Disables DHCP

[@deprecated] This command is deprecated. Please use AT+CWDHCP\_CUR or AT+CWDHCP\_DEF instead.

Commands	Query Command: AT+CWDHCP?	Set Command: AT+CWDHCP=<<mode>, <en> Function: to enable/disable DHCP.
Response	DHCP disabled or enabled now?	OK
Parameters	<ul style="list-style-type: none"> <li>• Bit0: <ul style="list-style-type: none"> <li>▶ 0: Station DHCP is disabled.</li> <li>▶ 1: Station DHCP is enabled.</li> </ul> </li> <li>• Bit1: <ul style="list-style-type: none"> <li>▶ 0: SoftAP DHCP is disabled.</li> <li>▶ 1: SoftAP DHCP is enabled.</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• &lt;mode&gt;: <ul style="list-style-type: none"> <li>▶ 0: Sets ESP8266 SoftAP</li> <li>▶ 1: Sets ESP8266 Station</li> <li>▶ 2: Sets both SoftAP and Station</li> </ul> </li> <li>• &lt;en&gt;: <ul style="list-style-type: none"> <li>▶ 0: Disables DHCP</li> <li>▶ 1: Enables DHCP</li> </ul> </li> </ul>



<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will be stored in the user parameter area in the flash.</li> <li>• This Set Command interacts with static-IP-related AT commands (AT+CIPSTA-related and AT+CIPA-related commands): <ul style="list-style-type: none"> <li>▶ If DHCP is enabled, static IP will be disabled;</li> <li>▶ If static IP is enabled, DHCP will be disabled;</li> <li>▶ Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>
--------------	--

#### 4.2.15. AT+CWDHCP\_CUR—Enables/Disables DHCP; Configuration Not Saved in the Flash

<b>Commands</b>	Query Command: AT+CWDHCP_CUR?	Set Command: AT+CWDHCP_CUR=<<mode>,<en> Function: to enable/disable DHCP.
<b>Response</b>	DHCP disabled or enabled now?	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• Bit0: <ul style="list-style-type: none"> <li>▶ 0: Station DHCP is disabled.</li> <li>▶ 1: Station DHCP is enabled.</li> </ul> </li> <li>• Bit1: <ul style="list-style-type: none"> <li>▶ 0: SoftAP DHCP is disabled.</li> <li>▶ 1: SoftAP DHCP is enabled.</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• &lt;mode&gt;: <ul style="list-style-type: none"> <li>▶ 0: Sets ESP8266 SoftAP</li> <li>▶ 1: Sets ESP8266 Station</li> <li>▶ 2: Sets both SoftAP and Station</li> </ul> </li> <li>• &lt;en&gt;: <ul style="list-style-type: none"> <li>▶ 0: Disables DHCP</li> <li>▶ 1: Enables DHCP</li> </ul> </li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will be stored in the user parameter area in the flash.</li> <li>• This Set Command interacts with static-IP-related AT commands (AT+CIPSTA-related and AT+CIPA-related commands): <ul style="list-style-type: none"> <li>▶ If DHCP is enabled, static IP will be disabled;</li> <li>▶ If static IP is enabled, DHCP will be disabled;</li> <li>▶ Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>	
<b>Example</b>	AT+CWDHCP_CUR=0,1	

#### 4.2.16. AT+CWDHCP\_DEF—Enables/Disables DHCP; Configuration Saved in the Flash

<b>Commands</b>	Query Command: AT+CWDHCP_DEF?	Set Command: AT+CWDHCP_DEF=<<mode>,<en> Function: to enable/disable DHCP.
<b>Response</b>	DHCP disabled or enabled now?	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• Bit0: <ul style="list-style-type: none"> <li>▶ 0: Station DHCP is disabled.</li> <li>▶ 1: Station DHCP is enabled.</li> </ul> </li> <li>• Bit1: <ul style="list-style-type: none"> <li>▶ 0: SoftAP DHCP is disabled.</li> <li>▶ 1: SoftAP DHCP is enabled.</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• &lt;mode&gt;: <ul style="list-style-type: none"> <li>▶ 0: Sets ESP8266 SoftAP</li> <li>▶ 1: Sets ESP8266 Station</li> <li>▶ 2: Sets both SoftAP and Station</li> </ul> </li> <li>• &lt;en&gt;: <ul style="list-style-type: none"> <li>▶ 0: Disables DHCP</li> <li>▶ 1: Enables DHCP</li> </ul> </li> </ul>



<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will be stored in the user parameter area in the flash.</li> <li>This Set Command interacts with static-IP-related AT commands (AT+CIPSTA-related and AT+CIPA-related commands): <ul style="list-style-type: none"> <li>If DHCP is enabled, static IP will be disabled;</li> <li>If static IP is enabled, DHCP will be disabled;</li> <li>Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>
<b>Example</b>	AT+CWDHCP_DEF=0,1

#### 4.2.17. AT+CWDHCP\_CUR—Sets the IP Address Allocated by ESP8266 SoftAP DHCP; Configuration Not Saved in Flash

<b>Commands</b>	Query Command: AT+CWDHCP_CUR?	Set Command: AT+CWDHCP_CUR=<enable>,<lease time>,<start IP>,<end IP>  Function: sets the IP address range of the ESP8266 SoftAP DHCP server.
<b>Response</b>	+CWDHCP_CUR=<lease time>,<start IP>,<end IP>	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>&lt;enable&gt;: <ul style="list-style-type: none"> <li>0: Disable the settings and use the default IP range.</li> <li>1: Enable setting the IP range, and the parameters below have to be set.</li> </ul> </li> <li>&lt;lease time&gt;: lease time; unit: minute; range [1, 2880].</li> <li>&lt;start IP&gt;: start IP of the IP range that can be obtained from ESP8266 SoftAP DHCP server.</li> <li>&lt;end IP&gt;: end IP of the IP range that can be obtained from ESP8266 SoftAP DHCP server.</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will NOT be saved in the flash.</li> <li>This AT command is enabled when ESP8266 runs as SoftAP, and when DHCP is enabled. The IP address should be in the same network segment as the IP address of ESP8266 SoftAP.</li> </ul>	
<b>Examples</b>	AT+CWDHCP_CUR=1,3,"192.168.4.10","192.168.4.15" or AT+CWDHCP_CUR=0 //Disable the settings and use the default IP range.	

#### 4.2.18. AT+CWDHCP\_DEF—Sets the IP Address Allocated by ESP8266 SoftAP DHCP; Configuration Saved in Flash

<b>Commands</b>	Query Command: AT+CWDHCP_DEF?	Set Command: AT+CWDHCP_DEF=<enable>,<lease time>,<start IP>,<end IP>  Function: sets the IP address range of the ESP8266 SoftAP DHCP server.
<b>Response</b>	+CWDHCP_DEF=<lease time>,<start IP>,<end IP>	OK



<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;enable&gt;: <ul style="list-style-type: none"> <li>▸ 0: Disable the settings and use the default IP range.</li> <li>▸ 1: Enable setting the IP range, and the parameters below have to be set.</li> </ul> </li> <li>• &lt;lease time&gt;: lease time; unit: minute; range [1, 2880].</li> <li>• &lt;start IP&gt;: start IP of the IP range that can be obtained from ESP8266 SoftAP DHCP server.</li> <li>• &lt;end IP&gt;: end IP of the IP range that can be obtained from ESP8266 SoftAP DHCP server.</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will be stored in the user parameter area in the flash.</li> <li>• This AT command is enabled when ESP8266 runs as SoftAP, and when DHCP is enabled. The IP address should be in the same network segment as the IP address of ESP8266 SoftAP.</li> </ul>
<b>Examples</b>	<pre>AT+CWDHCPS_DEF=1,3,"192.168.4.10","192.168.4.15"</pre> <p>or</p> <pre>AT+CWDHCPS_DEF=0 //Disable the settings and use the default IP range.</pre>

#### 4.2.19. AT+CWAUTOCONN—Auto-Connects to the AP or Not

<b>Set Command</b>	AT+CWAUTOCONN=<enable>
<b>Response</b>	OK
<b>Parameters</b>	<enable>: <ul style="list-style-type: none"> <li>▸ 0: does NOT auto-connect to AP on power-up.</li> <li>▸ 1: connects to AP automatically on power-up.</li> </ul> The ESP8266 Station connects to the AP automatically on power-up by default.
<b>Note</b>	The configuration changes will be saved in the system parameter area in the flash.
<b>Example</b>	AT+CWAUTOCONN=1

#### 4.2.20. AT+CIPSTAMAC—Sets the MAC Address of the ESP8266 Station

[@deprecated] This command is deprecated. Please use AT+CIPSTAMAC\_CUR or AT+CIPSTAMAC\_DEF instead.

<b>Commands</b>	Query Command: AT+CIPSTAMAC?	Set Command: AT+CIPSTAMAC=<mac> Function: to set the MAC address of the ESP8266 Station.
<b>Response</b>	+CIPSTAMAC:<mac> OK	OK
<b>Parameters</b>	<mac>: string parameter, MAC address of the ESP8266 Station.	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will be saved in the user parameter area in the flash.</li> <li>• The MAC address of ESP8266 SoftAP is different from that of the ESP8266 Station. Please make sure that you do not set the same MAC address for both of them.</li> <li>• Bit 0 of the ESP8266 MAC address CANNOT be 1. For example, a MAC address can be "18:..." but not "15:..."</li> </ul>	



Example	AT+CIPSTAMAC="18:fe:35:98:d3:7b"
---------	----------------------------------

#### 4.2.21. AT+CIPSTAMAC\_CUR—Sets the MAC Address of the ESP8266 Station; Configuration Not Saved in the Flash

Commands	Query Command: AT+CIPSTAMAC_CUR?	Set Command: AT+CIPSTAMAC_CUR=<mac> Function: to set the MAC address of the ESP8266 Station.
Response	+CIPSTAMAC_CUR: <mac> OK	OK
Parameters	<mac>: string parameter, MAC address of the ESP8266 Station.	
Notes	<ul style="list-style-type: none"> <li>The configuration changes will NOT be saved in the flash.</li> <li>The MAC address of ESP8266 SoftAP is different from that of the ESP8266 Station. Please make sure that you do not set the same MAC address for both of them.</li> <li>Bit 0 of the ESP8266 MAC address CANNOT be 1. For example, a MAC address can be "18:..." but not "15:...".</li> </ul>	
Example	AT+CIPSTAMAC_CUR="18:fe:35:98:d3:7b"	

#### 4.2.22. AT+CIPSTAMAC\_DEF—Sets the MAC Address of the ESP8266 Station; Configuration Saved in the Flash

Commands	Query Command: AT+CIPSTAMAC_DEF?	Set Command: AT+CIPSTAMAC_DEF=<mac> Function: to set the MAC address of the ESP8266 Station.
Response	+CIPSTAMAC_DEF: <mac> OK	OK
Parameters	<mac>: string parameter, MAC address of the ESP8266 Station.	
Notes	<ul style="list-style-type: none"> <li>The configuration changes will be saved in the user parameter area in the flash.</li> <li>The MAC address of ESP8266 SoftAP is different from that of the ESP8266 Station. Please make sure that you do not set the same MAC address for both of them.</li> <li>Bit 0 of the ESP8266 MAC address CANNOT be 1. For example, a MAC address can be "18:..." but not "15:...".</li> </ul>	
Example	AT+CIPSTAMAC_DEF="18:fe:35:98:d3:7b"	

#### 4.2.23. AT+CIPAPMAC—Sets the MAC Address of the ESP8266 SoftAP

[@deprecated] This command is deprecated. Please use AT+CIPAPMAC\_CUR or AT+CIPAPMAC\_DEF instead.





<b>Commands</b>	Query Command: AT+CIPAPMAC?  Function: to obtain the MAC address of the ESP8266 SoftAP.	Set Command: AT+CIPAPMAC=<mac>  Function: to set the MAC address of the ESP8266 SoftAP.
<b>Response</b>	+CIPAPMAC: <mac> OK	OK
<b>Parameters</b>	<mac>: string parameter, MAC address of ESP8266 SoftAP.	
<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will be saved in the user parameter area in the flash.</li> <li>The MAC address of ESP8266 SoftAP is different from that of the ESP8266 Station. Please make sure you do not set the same MAC address for both of them.</li> <li>Bit 0 of the ESP8266 MAC address CANNOT be 1. For example, a MAC address can be "18:..." but not "15:..."</li> </ul>	
<b>Example</b>	AT+CIPAPMAC="1a:fe:36:97:d5:7b"	

#### 4.2.24. AT+CIPAPMAC\_CUR—Sets the MAC Address of the ESP8266 SoftAP; Configuration Not Saved in the Flash

<b>Commands</b>	Query Command: AT+CIPAPMAC_CUR?  Function: to obtain the MAC address of the ESP8266 SoftAP.	Set Command: AT+CIPAPMAC_CUR=<mac>  Function: to set the MAC address of the ESP8266 SoftAP.
<b>Response</b>	+CIPAPMAC_CUR: <mac> OK	OK
<b>Parameters</b>	<mac>: string parameter, MAC address of ESP8266 SoftAP.	
<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will NOT be saved the flash.</li> <li>The MAC address of ESP8266 SoftAP is different from that of the ESP8266 Station. Please make sure you do not set the same MAC address for both of them.</li> <li>Bit 0 of the ESP8266 MAC address CANNOT be 1. For example, a MAC address can be "18:..." but not "15:..."</li> </ul>	
<b>Example</b>	AT+CIPAPMAC_CUR="1a:fe:36:97:d5:7b"	



#### 4.2.25. AT+CIPAPMAC\_DEF—Sets the MAC Address of the ESP8266 SoftAP; Configuration Saved in Flash

<b>Commands</b>	Query Command: AT+CIPAPMAC_DEF?  Function: to obtain the MAC address of the ESP8266 SoftAP.	Set Command: AT+CIPAPMAC_DEF=<mac>  Function: to set the MAC address of the ESP8266 SoftAP.
<b>Response</b>	+CIPAPMAC_DEF: <mac> OK	OK
<b>Parameters</b>	<mac>: string parameter, MAC address of ESP8266 SoftAP.	
<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will be saved in the user parameter area in the flash.</li> <li>The MAC address of ESP8266 SoftAP is different from that of the ESP8266 Station. Please make sure you do not set the same MAC address for both of them.</li> <li>Bit 0 of the ESP8266 MAC address CANNOT be 1. For example, a MAC address can be "18:..." but not "15:...".</li> </ul>	
<b>Example</b>	AT+CIPAPMAC_DEF="1a:fe:36:97:d5:7b"	

#### 4.2.26. AT+CIPSTA—Sets the IP Address of the ESP8266 Station

[@deprecated] This command is deprecated. Please use AT+CIPSTA\_CUR or AT+CIPSTA\_DEF instead.

<b>Commands</b>	Query Command: AT+CIPSTA?  Function: to obtain the IP address of the ESP8266 Station.	Set Command: AT+CIPSTA=<ip>[, <gateway>, <netmask>]  Function: to set the IP address of the ESP8266 Station.
<b>Response</b>	+CIPSTA: <ip> +CIPSTA: <gateway> +CIPSTA: <netmask> OK	OK
<b>Parameters</b>	<p><b>⚠ Notice:</b></p> <p>Only when the ESP8266 Station is connected to an AP can its IP address be queried.</p>	<ul style="list-style-type: none"> <li>&lt;ip&gt;: string parameter, the IP address of the ESP8266 Station.</li> <li>[&lt;gateway&gt;]: gateway.</li> <li>[&lt;netmask&gt;]: netmask.</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will be saved in the user parameter area in the flash.</li> <li>The Set Command interacts with DHCP-related AT commands (AT+CWDHCP-related commands): <ul style="list-style-type: none"> <li>▶ If static IP is enabled, DHCP will be disabled;</li> <li>▶ If DHCP is enabled, static IP will be disabled;</li> <li>▶ Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>	
<b>Example</b>	AT+CIPSTA="192.168.6.100", "192.168.6.1", "255.255.255.0"	



#### 4.2.27. AT+CIPSTA\_CUR—Sets the Current IP Address of the ESP8266 Station; Configuration Not Saved in the Flash

<b>Commands</b>	Query Command: AT+CIPSTA_CUR?  Function: to obtain the current IP address of the ESP8266 Station.	Set Command: AT+CIPSTA_CUR=<ip>[, <gateway>, <netmask>]  Function: to set the current IP address of the ESP8266 Station.
<b>Response</b>	+CIPSTA_CUR:<ip> +CIPSTA_CUR:<gateway> +CIPSTA_CUR:<netmask> OK	OK
<b>Parameters</b>	<b>⚠ Notice:</b> Only when the ESP8266 Station is connected to an AP can its IP address be queried.	<ul style="list-style-type: none"> <li>• &lt;ip&gt;: string parameter, the IP address of the ESP8266 Station.</li> <li>• [&lt;gateway&gt;]: gateway.</li> <li>• [&lt;netmask&gt;]: netmask.</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will NOT be saved in the flash.</li> <li>• The Set Command interacts with DHCP-related AT commands (AT+CWDHCP-related commands):             <ul style="list-style-type: none"> <li>▶ If static IP is enabled, DHCP will be disabled;</li> <li>▶ If DHCP is enabled, static IP will be disabled;</li> <li>▶ Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>	
<b>Example</b>	AT+CIPSTA_CUR="192.168.6.100", "192.168.6.1", "255.255.255.0"	

#### 4.2.28. AT+CIPSTA\_DEF—Sets the Default IP Address of the ESP8266 Station; Configuration Saved in the Flash

<b>Commands</b>	Query Command: AT+CIPSTA_DEF?  Function: to obtain the default IP address of the ESP8266 Station.	Set Command: AT+CIPSTA_DEF=<ip>[, <gateway>, <netmask>]  Function: to set the default IP address of the ESP8266 Station.
<b>Response</b>	+CIPSTA_DEF:<ip> +CIPSTA_DEF:<gateway> +CIPSTA_DEF:<netmask> OK	OK
<b>Parameters</b>	<b>⚠ Notice:</b> Only when the ESP8266 Station is connected to an AP can its IP address be queried.	<ul style="list-style-type: none"> <li>• &lt;ip&gt;: string parameter, the IP address of the ESP8266 Station.</li> <li>• [&lt;gateway&gt;]: gateway.</li> <li>• [&lt;netmask&gt;]: netmask.</li> </ul>



<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will be saved in the user parameter area in the flash.</li> <li>The Set Command interacts with DHCP-related AT commands (AT+CWDHCP-related commands): <ul style="list-style-type: none"> <li>If static IP is enabled, DHCP will be disabled;</li> <li>If DHCP is enabled, static IP will be disabled;</li> <li>Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>
<b>Example</b>	AT+CIPSTA_DEF="192.168.6.100", "192.168.6.1", "255.255.255.0"

#### 4.2.29. AT+CIPAP—Sets the IP Address of the ESP8266 SoftAP

[@deprecated] This command is deprecated. Please use AT+CIPAP\_CUR or AT+CIPAP\_DEF instead.

<b>Commands</b>	Query Command: AT+CIPAP?  Function: to obtain the IP address of the ESP8266 SoftAP.	Set Command: AT+CIPAP=<ip>[, <gateway>, <netmask>]  Function: to set the IP address of the ESP8266 SoftAP.
<b>Response</b>	+CIPAP:<ip> +CIPAP:<gateway> +CIPAP:<netmask> OK	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>&lt;ip&gt;: string parameter, the IP address of the ESP8266 SoftAP.</li> <li>[&lt;gateway&gt;]: gateway.</li> <li>[&lt;netmask&gt;]: netmask.</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>The configuration changes will be saved in the user parameter area in the flash.</li> <li>Currently, ESP8266 only supports class C IP addresses.</li> <li>The Set Command interacts with DHCP-related AT commands (AT+CWDHCP-related commands): <ul style="list-style-type: none"> <li>If static IP is enabled, DHCP will be disabled;</li> <li>If DHCP is enabled, static IP will be disabled;</li> <li>Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>	
<b>Example</b>	AT+CIPAP="192.168.5.1", "192.168.5.1", "255.255.255.0"	

#### 4.2.30. AT+CIPAP\_CUR—Sets the IP Address of the ESP8266 SoftAP; Configuration Not Saved in the Flash

<b>Commands</b>	Query Command: AT+CIPAP_CUR?  Function: to obtain the current IP address of the ESP8266 SoftAP.	Set Command: AT+CIPAP_CUR=<ip>[, <gateway>, <netmask>]  Function: to set the current IP address of the ESP8266 SoftAP.
<b>Response</b>	+CIPAP_CUR:<ip> +CIPAP_CUR:<gateway> +CIPAP_CUR:<netmask> OK	OK



<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;ip&gt;: string parameter, the IP address of the ESP8266 SoftAP.</li> <li>• [&lt;gateway&gt;]: gateway.</li> <li>• [&lt;netmask&gt;]: netmask.</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will not be saved in the flash.</li> <li>• Currently, ESP8266 only supports class C IP addresses.</li> <li>• The Set Command interacts with DHCP-related AT commands (AT+CWDHCP-related commands): <ul style="list-style-type: none"> <li>▶ If static IP is enabled, DHCP will be disabled;</li> <li>▶ If DHCP is enabled, static IP will be disabled;</li> <li>▶ Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>
<b>Example</b>	AT+CIPAP_CUR="192.168.5.1", "192.168.5.1", "255.255.255.0"

#### 4.2.31. AT+CIPAP\_DEF—Sets the Default IP Address of the ESP8266 SoftAP; Configuration Saved in the Flash

<b>Commands</b>	<p>Query Command: AT+CIPAP_DEF?</p> <p>Function: to obtain the default IP address of the ESP8266 SoftAP.</p>	<p>Set Command: AT+CIPAP_DEF=&lt;ip&gt;[, &lt;gateway&gt;, &lt;netmask&gt;]</p> <p>Function: to set the default IP address of the ESP8266 SoftAP.</p>
<b>Response</b>	<p>+CIPAP_DEF:&lt;ip&gt;</p> <p>+CIPAP_DEF:&lt;gateway&gt;</p> <p>+CIPAP_DEF:&lt;netmask&gt;</p> <p>OK</p>	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;ip&gt;: string parameter, the IP address of the ESP8266 SoftAP.</li> <li>• [&lt;gateway&gt;]: gateway.</li> <li>• [&lt;netmask&gt;]: netmask.</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will be saved in the user parameter area in the flash.</li> <li>• Currently, ESP8266 only supports class C IP addresses.</li> <li>• The Set Command interacts with DHCP-related AT commands (AT+CWDHCP-related commands): <ul style="list-style-type: none"> <li>▶ If static IP is enabled, DHCP will be disabled;</li> <li>▶ If DHCP is enabled, static IP will be disabled;</li> <li>▶ Whether it is DHCP or static IP that is enabled depends on the last configuration.</li> </ul> </li> </ul>	
<b>Example</b>	AT+CIPAP_DEF="192.168.5.1", "192.168.5.1", "255.255.255.0"	

#### 4.2.32. AT+CWSTARTSMART—Starts SmartConfig

<b>Commands</b>	<p>Execute Command: AT+CWSTARTSMART</p> <p>Function: to start SmartConfig. (The type of SmartConfig is ESP-TOUCH + AirKiss.)</p>	<p>Set Command: AT+CWSTARTSMART=&lt;type&gt;</p> <p>Function: to start SmartConfig of a designated type.</p>
<b>Response</b>	OK	



<b>Parameters</b>	<p>&lt;type&gt;:</p> <ul style="list-style-type: none"> <li>▶ 1: ESP-TOUCH</li> <li>▶ 2: AirKiss</li> <li>▶ 3: ESP-TOUCH+AirKiss</li> </ul>
<b>Messages</b>	<p>When smartconfig starts, it will prompt messages as below:</p> <pre>smartconfig type: &lt;type&gt; // AIRKISS or ESPTOUCH smart get wifi info // got SSID and password ssid:&lt;AP's SSID&gt; password:&lt;AP's password&gt; // ESP8266 will try to connect to the AP WIFI CONNECTED WIFI GOT IP smartconfig connected wifi // if the connection failed, it will prompt "smartconfig connect fail"</pre>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• For details on SmartConfig please see <a href="#">ESP-TOUCH User Guide</a>.</li> <li>• SmartConfig is only available in the ESP8266 Station mode.</li> <li>• The message <code>smart get wifi info</code> means that SmartConfig has successfully acquired the AP information. ESP8266 will try to connect to the target AP.</li> <li>• Message <code>smartconfig connected wifi</code> is printed if the connection is successful. Use command <code>AT+CWSTOPSMART</code> to stop SmartConfig before running other commands. Please make sure that you do not execute other commands during SmartConfig.</li> <li>• Starting from AT_v1.0, SmartConfig can get protocol type (AirKiss or ESP-TOUCH) automatically by command <code>AT+CWSTARTSMART</code>.</li> </ul>
<b>Example</b>	<pre>AT+CWMODE=1 AT+CWSTARTSMART</pre>

#### 4.2.33. AT+CWSTOPSMART – Stops SmartConfig

<b>Execute Command</b>	AT+CWSTOPSMART
<b>Response</b>	OK
<b>Parameters</b>	-
<b>Note</b>	Irrespective of whether SmartConfig succeeds or not, before executing any other AT commands, please always call AT+CWSTOPSMART to release the internal memory taken up by SmartConfig.
<b>Example</b>	AT+CWSTOPSMART

#### 4.2.34. AT+CWSTARTDISCOVER – Enables the Mode that ESP8266 can be Found by WeChat

<b>Set Command</b>	AT+CWSTARTDISCOVER=<WeChat number>,<dev_type>,<time>
<b>Response</b>	OK



Parameters	<ul style="list-style-type: none"> <li>• &lt;WeChat number&gt;: WeChat official account, which is to be obtained from WeChat.</li> <li>• &lt;dev_type&gt;: the device type, which is to be obtained from WeChat.</li> <li>• &lt;time&gt;: the interval of time for ESP8266 to send packets; range: 0 ~ 24x3600; unit: second. <ul style="list-style-type: none"> <li>▶ 0: ESP8266 will not take the initiative to send packets; it only makes response to queries from WeChat.</li> <li>▶ Otherwise: the time interval for ESP8266 to send packets regularly in order to be detected by WeChat on the same LAN.</li> </ul> </li> </ul>
Note	<ul style="list-style-type: none"> <li>• For details on detection function of WeChat, please refer to <a href="http://iot.weixin.qq.com">http://iot.weixin.qq.com</a>.</li> <li>• ESP8266 Station should connect to an AP and obtain an IP address first before this command is used.</li> </ul>
Example	AT+CWSTARTDISCOVER="gh_9e2cff3dfa51", "122475", 10

#### 4.2.35. AT+CWSTOPDISCOVER—Disables the Mode that ESP8266 can be Found by WeChat

Execute Command	AT+CWSTOPDISCOVER
Response	OK
Example	AT+CWSTOPDISCOVER

#### 4.2.36. AT+WPS—Enables the WPS Function

Set Command	AT+WPS=<enable>
Response	OK
Parameters	<enable>: <ul style="list-style-type: none"> <li>▶ 1: enables WPS/Wi-Fi Protected Setup</li> <li>▶ 0: disables WPS</li> </ul>
Notes	<ul style="list-style-type: none"> <li>• WPS must be used when the ESP8266 Station is enabled.</li> <li>• WPS does not support WEP/Wired-Equivalent Privacy encryption.</li> </ul>
Example	AT+CWMODE=1 AT+WPS=1

#### 4.2.37. AT+MDNS—Configures the MDNS Function

Set Command	AT+MDNS=<enable>, <hostname>, <server_name>, <server_port>
Response	OK or opmode mismatch when mdns ERROR



<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;enable&gt;: <ul style="list-style-type: none"> <li>▶ 1: enables the MDNS function; the following three parameters need to be set.</li> <li>▶ 0: disables the MDNS function; the following three parameters need not to be set.</li> </ul> </li> <li>• &lt;hostname&gt;: MDNS host name</li> <li>• &lt;server_name&gt;: MDNS server name</li> <li>• &lt;server_port&gt;: MDNS server port</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• Please do not use special characters (such as .) or a protocol name (for example, http) for &lt;hostname&gt; and &lt;server_name&gt;.</li> <li>• ESP8266 SoftAP mode does not support the MDNS function for now.</li> </ul>
<b>Example</b>	AT+MDNS=1,"espressif","iot",8080

#### 4.2.38. AT+CWHOSTNAME—Configures the Name of ESP8266 Station

<b>Commands</b>	Query Command: AT+CWHOSTNAME? Function: Checks the host name of ESP8266 Station.	Set Command: AT+CWHOSTNAME=<hostname> Function: Sets the host name of ESP8266 Station.
<b>Response</b>	+CWHOSTNAME:<host name> OK If the station mode is not enabled, the command will return: +CWHOSTNAME:<null> OK	OK If the Station mode is not enabled, the command will return: ERROR
<b>Parameters</b>	<hostname>: the host name of the ESP8266 Station.	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes are not saved in the flash.</li> <li>• The default host name of the ESP8266 Station is ESP_XXXXXX; XXXXXX is the lower 3 bytes of the MAC address, for example, +CWHOSTNAME:&lt;ESP_A378DA&gt;.</li> </ul>	
<b>Example</b>	AT+CWMODE=3 AT+CWHOSTNAME="my_test"	

#### 4.2.39. AT+CWCOUNTRY\_CUR—Set ESP8266 WiFi Country Code; Configuration Not Saved in the Flash

<b>Commands</b>	Query Command: AT+CWCOUNTRY_CUR? Function: Checks the current WiFi country code of ESP8266.	Set Command: AT+CWCOUNTRY_CUR=<country_policy>,<country_code>,<start_channel>,<total_channel_number> Function: Sets the current WiFi country code of ESP8266.
-----------------	---	---





<b>Response</b>	<code>+CWCOUNTRY_CUR:&lt;country_policy&gt;,&lt;country_code&gt;,&lt;start_channel&gt;,&lt;total_channel_number&gt;</code> OK Command <code>AT+CWCOUNTRY_CUR?</code> will return the actual value of WiFi country code, which may change to be the same as the AP it connected to.	OK
<b>Parameters</b>	<code>&lt;country_policy&gt;</code> : <ul style="list-style-type: none"><li>• 0: will change the county code to be the same as the AP it connected to</li><li>• 1: the country code will not change, always be the one set by command.</li></ul> <code>&lt;country_code&gt;</code> : country code, the length can be 3 characters at most; but the third one is a special character which will not be show when querying by command <code>AT+CWCOUNTRY_CUR?</code> <code>&lt;start_channel&gt;</code> : the channel which starting from <code>&lt;total_channel_number&gt;</code> : total channel number	
<b>Notes</b>	The configuration changes will NOT be saved in the flash.	
<b>Example</b>	<code>AT+CWMODE=3</code> <code>AT+CWCOUNTRY_CUR=1,"CN",1,5</code>	



#### 4.2.40. AT+CWCOUNTRY\_DEF—Set the default WiFi Country Code of ESP8266; Configuration Saved in the Flash

<b>Commands</b>	<p>Query Command:</p> <p>AT+CWCOUNTRY_DEF?</p> <p>Function: Checks the default WiFi country code of ESP8266 which saved in the flash.</p>	<p>Set Command:</p> <p>AT+CWCOUNTRY_DEF=&lt;country_policy&gt;,&lt;country_code&gt;,&lt;start_channel&gt;,&lt;total_channel_number&gt;</p> <p>Function: Sets the default WiFi country code of ESP8266, and saves in the flash.</p>
<b>Response</b>	<p>+CWCOUNTRY_DEF:&lt;country_policy&gt;,&lt;country_code&gt;,&lt;start_channel&gt;,&lt;total_channel_number&gt;</p> <p>OK</p> <p>Command AT+CWCOUNTRY_DEF? will return the default WiFi country code which stores in the flash.</p>	<p>OK</p>
<b>Parameters</b>	<p>&lt;country_policy&gt;:</p> <ul style="list-style-type: none"> <li>• 0: will change the county code to be the same as the AP it connected to</li> <li>• 1: the country code will not change, always be the one set by command.</li> </ul> <p>&lt;country_code&gt;: country code, the length can be 3 characters at most; but the third one is a special character which will not be show when querying by command AT+CWCOUNTRY_DEF?</p> <p>&lt;start_channel&gt; : the channel which starting from</p> <p>&lt;total_channel_number&gt; : total channel number</p>	
<b>Notes</b>	<p>The configuration changes will be saved in the flash user parameter area.</p>	
<b>Example</b>	<p>AT+CWMODE=3</p> <p>AT+CWCOUNTRY_DEF=1,"CN",1,5</p>	



# 5. TCP/IP-Related AT Commands

## 5.1. Overview

Command	Description
AT+CIPSTATUS	Gets the connection status
AT+CIPDOMAIN	DNS function
AT+CIPSTART	Establishes TCP connection, UDP transmission or SSL connection
AT+CIPSSLSIZE	Sets the size of SSL buffer
AT+CIPSEND	Sends data
AT+CIPSENDEX	Sends data when length of data is <Length>, or when \0 appears in the data
AT+CIPSENDERBUF	Writes data into TCP-send-buffer
AT+CIPBUFRESET	Resets the segment ID count
AT+CIPBUFSTATUS	Checks the status of TCP-send-buffer
AT+CIPCHECKSEQ	Checks if a specific segment is sent or not
AT+CIPCLOSE	Closes TCP/UDP/SSL connection
AT+CIFSR	Gets the local IP address
AT+CIPMUX	Configures the multiple connections mode
AT+CIPSERVER	Deletes/Creates a TCP server
AT+CIPSERVERMAXCONN	Set the maximum connections that server allows
AT+CIPMODE	Configures the transmission mode
AT+SAVETRANSLINK	Saves the transparent transmission link in the flash
AT+CIPSTO	Sets timeout when ESP8266 runs as TCP server
AT+PING	Ping packets
AT+CIUPDATE	Upgrades the software through network
AT+CIPDINFO	Shows remote IP and remote port with +IPD
AT+CIPSNTPCFG	Configures the time domain and SNTP server.
AT+CIPSNTPTIME	Queries the SNTP time.
AT+CIPDNS_CUR	Sets user-defined DNS servers; configuration not saved in the flash
AT+CIPDNS_DEF	Sets user-defined DNS servers; configuration saved in the flash



## 5.2. Commands

### 5.2.1. AT+CIPSTATUS—Gets the Connection Status

Execute Command	AT+CIPSTATUS
Response	STATUS:<stat> +CIPSTATUS:<link ID>,<type>,<remote IP>,<remote port>,<local port>,<tetype>
Parameters	<ul style="list-style-type: none"> <li>• &lt;stat&gt;: status of the ESP8266 Station interface. <ul style="list-style-type: none"> <li>▶ 2: The ESP8266 Station is connected to an AP and its IP is obtained.</li> <li>▶ 3: The ESP8266 Station has created a TCP or UDP transmission.</li> <li>▶ 4: The TCP or UDP transmission of ESP8266 Station is disconnected.</li> <li>▶ 5: The ESP8266 Station does NOT connect to an AP.</li> </ul> </li> <li>• &lt;link ID&gt;: ID of the connection (0~4), used for multiple connections.</li> <li>• &lt;type&gt;: string parameter, "TCP" or "UDP".</li> <li>• &lt;remote IP&gt;: string parameter indicating the remote IP address.</li> <li>• &lt;remote port&gt;: the remote port number.</li> <li>• &lt;local port&gt;: ESP8266 local port number.</li> <li>• &lt;tetype&gt;: <ul style="list-style-type: none"> <li>▶ 0: ESP8266 runs as a client.</li> <li>▶ 1: ESP8266 runs as a server.</li> </ul> </li> </ul>

### 5.2.2. AT+CIPDOMAIN—DNS Function

Execute Command	AT+CIPDOMAIN=<domain name>
Response	+CIPDOMAIN:<IP address> OK Or DNS Fail ERROR
Parameter	<domain name>: the domain name, length should be less than 64 bytes.
Example	<pre>AT+CWMODE=1                // set Station mode AT+CWJAP="SSID","password"  // access to the internet AT+CIPDOMAIN="iot.espressif.cn" // DNS function</pre>



## 5.2.3. AT+CIPSTART—Establishes TCP Connection, UDP Transmission or SSL Connection

## Establish TCP Connection

<b>Set Command</b>	Single TCP connection (AT+CIPMUX=0): AT+CIPSTART=<type>,<remote IP>,<remote port>[,<TCP keep alive>]	Multiple TCP Connections (AT+CIPMUX=1): AT+CIPSTART=<link ID>,<type>,<remote IP>,<remote port>[,<TCP keep alive>]
<b>Response</b>	OK or ERROR  If the TCP connection is already established, the response is: ALREADY CONNECTED	
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;link ID&gt;: ID of network connection (0~4), used for multiple connections.</li> <li>• &lt;type&gt;: string parameter indicating the connection type: "TCP", "UDP" or "SSL".</li> <li>• &lt;remote IP&gt;: string parameter indicating the remote IP address.</li> <li>• &lt;remote port&gt;: the remote port number.</li> <li>• [&lt;TCP keep alive&gt;]: detection time interval when TCP is kept alive; this function is disabled by default. <ul style="list-style-type: none"> <li>▶ 0: disable TCP keep-alive.</li> <li>▶ 1 ~ 7200: detection time interval; unit: second (s).</li> </ul> </li> </ul>	
<b>Examples</b>	AT+CIPSTART="TCP","iot.espressif.cn",8000 AT+CIPSTART="TCP","192.168.101.110",1000 For more information please see: <a href="#">ESP8266 AT Command Examples</a> .	

## Establish UDP Transmission

<b>Set Command</b>	Single connection (AT+CIPMUX=0): AT+CIPSTART=<type>,<remote IP>,<remote port>[,<UDP local port>],[<UDP mode>]	Multiple connections (AT+CIPMUX=1): AT+CIPSTART=<link ID>,<type>,<remote IP>,<remote port>[,<UDP local port>],[<UDP mode>]
<b>Response</b>	OK or ERROR  If the UDP transmission is already established, the response is: ALREADY CONNECTED	



<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;link ID&gt;: ID of network connection (0~4), used for multiple connections.</li> <li>• &lt;type&gt;: string parameter indicating the connection type: "TCP", "UDP" or "SSL".</li> <li>• &lt;remote IP&gt;: string parameter indicating the remote IP address.</li> <li>• &lt;remote port&gt;: remote port number.</li> <li>• [&lt;UDP local port&gt;]: optional; UDP port of ESP8266.</li> <li>• [&lt;UDP mode&gt;]: optional. In the UDP transparent transmission, the value of this parameter has to be 0. <ul style="list-style-type: none"> <li>▶ 0: the destination peer entity of UDP will not change; this is the default setting.</li> <li>▶ 1: the destination peer entity of UDP can change once.</li> <li>▶ 2: the destination peer entity of UDP is allowed to change.</li> </ul> </li> </ul> <p><b>⚠ Notice:</b></p> <p>To use &lt;UDP mode&gt; , &lt;UDP local port&gt; must be set first.</p>
<b>Example</b>	AT+CIPSTART="UDP", "192.168.101.110", 1000, 1002, 2 For more information please see: <a href="#">ESP8266 AT Command Examples</a> .

### Establish SSL Connection

<b>Set Command</b>	AT+CIPSTART=[<link ID>,]<type>,<remote IP>,<remote port>[,<TCP keep alive>]
<b>Response</b>	OK or ERROR If the TCP connection is already established, the response is: ALREADY CONNECTED
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;link ID&gt;: ID of network connection (0~4), used for multiple connections.</li> <li>• &lt;type&gt;: string parameter indicating the connection type: "TCP", "UDP" or "SSL".</li> <li>• &lt;remote IP&gt;: string parameter indicating the remote IP address.</li> <li>• &lt;remote port&gt;: the remote port number.</li> <li>• [&lt;TCP keep alive&gt;]: detection time interval when TCP is kept alive; this function is disabled by default. <ul style="list-style-type: none"> <li>▶ 0: disable the TCP keep-alive function.</li> <li>▶ 1 ~ 7200: detection time interval, unit: second (s).</li> </ul> </li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• ESP8266 can only set one SSL connection at most.</li> <li>• SSL connection does not support UART-Wi-Fi passthrough mode (transparent transmission).</li> <li>• SSL connection needs a large amount of memory; otherwise, it may cause system reboot. The command AT+CIPSSLSIZE=&lt;size&gt; can be used to enlarge the SSL buffer size.</li> </ul>
<b>Example</b>	AT+CIPSTART="SSL", "iot.espressif.cn", 8443



### 5.2.4. AT+CIPSSLSIZE—Sets the Size of SSL Buffer

Set Command	AT+CIPSSLSIZE=<size>
Response	OK
Parameters	<size>: the size of the SSL buffer; range of value: [2048, 4096].
Example	AT+CIPSSLSIZE=4096

### 5.2.5. AT+CIPSEND—Sends Data

Commands	<p>Set Command:</p> <ol style="list-style-type: none"> <li>Single connection: (+CIPMUX=0) AT+CIPSEND=&lt;length&gt;</li> <li>Multiple connections: (+CIPMUX=1) AT+CIPSEND=&lt;link ID&gt;,&lt;length&gt;</li> <li>Remote IP and ports can be set in UDP transmission: AT+CIPSEND=[&lt;link ID&gt;,&lt;length&gt; [&lt;remote IP&gt;,&lt;remote port&gt;]</li> </ol> <p>Function: to configure the data length in normal transmission mode.</p>	<p>Execute Command:</p> <p>AT+CIPSEND</p> <p>Function: to start sending data in transparent transmission mode.</p>
Response	<p>Send data of designated length.</p> <p>Wrap return &gt; after the Set Command. Begin receiving serial data. When data length defined by &lt;length&gt; is met, the transmission of data starts.</p> <p>If the connection cannot be established or gets disrupted during data transmission, the system returns:</p> <p>ERROR</p> <p>If data is transmitted successfully, the system returns:</p> <p>SEND OK</p> <p>If it failed, the system returns:</p> <p>SEND FAIL</p>	<p>Wrap return &gt; after executing this command.</p> <p>Enter transparent transmission, with a 20-ms interval between each packet, and a maximum of 2048 bytes per packet.</p> <p>When a single packet containing +++ is received, ESP8266 returns to normal command mode. Please wait for at least one second before sending the next AT command.</p> <p>This command can only be used in transparent transmission mode which requires single connection.</p> <p>For UDP transparent transmission, the value of &lt;UDP mode&gt; has to be 0 when using AT+CIPSTART.</p>
Parameters	<ul style="list-style-type: none"> <li>&lt;link ID&gt;: ID of the connection (0~4), for multiple connections.</li> <li>&lt;length&gt;: data length, MAX: 2048 bytes.</li> <li>[&lt;remote IP&gt;]: remote IP can be set in UDP transmission.</li> <li>[&lt;remote port&gt;]: remote port can be set in UDP transmission.</li> </ul>	-
Example	For more information please see: <a href="#">ESP8266 AT Command Examples</a> .	



### 5.2.6. AT+CIPSENDEX—Sends Data

<b>Set Command</b>	<p>1. Single connection: (+CIPMUX=0) AT+CIPSENDEX=&lt;length&gt;</p> <p>2. Multiple connections: (+CIPMUX=1) AT+CIPSENDEX=&lt;link ID&gt;,&lt;length&gt;</p> <p>3. Remote IP and ports can be set in UDP transmission: AT+CIPSENDEX=[&lt;link ID&gt;,&lt;length&gt;[,&lt;remote IP&gt;,&lt;remote port&gt;]</p> <p>Function: to configure the data length in normal transmission mode.</p>
<b>Response</b>	<p>Send data of designated length.</p> <p>Wrap return &gt; after the Set Command. Begin receiving serial data. When the requirement of data length, determined by &lt;length&gt;, is met, or when \0 appears in the data, the transmission starts.</p> <p>If connection cannot be established or gets disconnected during transmission, the system returns:</p> <p>ERROR</p> <p>If data are successfully transmitted, the system returns:</p> <p>SEND OK</p> <p>If it failed, the system returns:</p> <p>SEND FAIL</p>
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;link ID&gt;: ID of the connection (0~4), for multiple connections.</li> <li>• &lt;length&gt;: data length, MAX: 2048 bytes.</li> <li>• When the requirement of data length, determined by &lt;length&gt;, is met, or when \0 appears, the transmission of data starts. Go back to the normal command mode and wait for the next AT command.</li> <li>• When sending \0, please send it as \\0.</li> </ul>

### 5.2.7. AT+CIPSENBUFF—Writes Data into the TCP-Send-Buffer

<b>Set Command</b>	<p>1. Single connection: (+CIPMUX=0) AT+CIPSENBUFF=&lt;length&gt;</p> <p>2. Multiple connections: (+CIPMUX=1) AT+CIPSENBUFF=&lt;link ID&gt;,&lt;length&gt;</p>
--------------------	--





<b>Response</b>	<p>&lt;current segment ID&gt;,&lt;segment ID of which sent successfully&gt;</p> <p>OK</p> <p>&gt;</p> <ul style="list-style-type: none"> <li>• Wrap return &gt; begins receiving serial data; when the length of data defined by the parameter &lt;length&gt; is met, the data is sent; if the data length over the value of &lt;length&gt;, the data will be discarded, and the command returns <b>busy</b>.</li> <li>• If the connection cannot be established, or if it is not a TCP connection, or if the buffer is full, or some other error occurs, the command returns</li> </ul> <p>ERROR</p> <ul style="list-style-type: none"> <li>• If data is transmitted successfully, <ul style="list-style-type: none"> <li>▸ for single connection, the response is: <p>&lt;segment ID&gt;,SEND OK</p> </li> <li>▸ for multiple connections, the response is: <p>&lt;link ID&gt;,&lt;segment ID&gt;,SEND OK</p> </li> </ul> </li> <li>• If it failed, the system returns: <p>SEND FAIL</p> </li> </ul>
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;link ID&gt;: ID of the connection (0~4), for multiple connections.</li> <li>• &lt;segment ID&gt;: uint32; the ID assigned to each data packet, starting from 1; the ID number increases by 1 every time a data packet is written into the buffer.</li> <li>• &lt;length&gt;: data length; MAX: 2048 bytes.</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• This command only writes data into the TCP-send-buffer, so it can be called continually, and the user need not wait for SEND OK; if a TCP segment is sent successfully, it will return &lt;segment ID&gt;,SEND OK.</li> <li>• Before data length reaches the value defined by &lt;length&gt;, input +++ can switch back from data mode to command mode, and discard the data received before.</li> <li>• This command can NOT be used for SSL connections.</li> </ul>

### 5.2.8. AT+CIPBUFRESET – Resets the Segment ID Count

<b>Set Command</b>	<ol style="list-style-type: none"> <li>1. Single connection: (+CIPMUX=0) AT+CIPBUFRESET</li> <li>2. Multiple connections: (+CIPMUX=1) AT+CIPBUFRESET=&lt;link ID&gt;</li> </ol>
<b>Response</b>	<p>OK</p> <p>If the connection is not established or there is still TCP data waiting to be sent, the response will be:</p> <p>ERROR</p>
<b>Parameter</b>	<link ID>: ID of the connection (0~4), for multiple connections.
<b>Note</b>	This command can only be used when AT+CIPSENDER is used.



### 5.2.9. AT+CIPBUFSTATUS—Checks the Status of the TCP-Send-Buffer

Set Command	<ol style="list-style-type: none"> <li>Single connection: (+CIPMUX=0) AT+CIPBUFSTATUS</li> <li>Multiple connections: (+CIPMUX=1) AT+CIPBUFSTATUS=&lt;link ID&gt;</li> </ol>
Response	<next segment ID>,<segment ID sent >,<segment ID successfully sent>,<remain buffer size>,<queue number> OK
Parameters	<ul style="list-style-type: none"> <li>&lt;next segment ID&gt;: the next segment ID obtained by AT+CIPSENDERBUF;</li> <li>&lt;segment ID sent&gt;: the ID of the TCP segment last sent;</li> <li>Only when &lt;next segment ID&gt; - &lt;segment ID sent&gt; = 1, can AT+CIPBUFRESET be called to reset the counting.</li> <li>&lt;segment ID successfully sent&gt;: the ID of the last successfully sent TCP segment;</li> <li>&lt;remain buffer size&gt;: the remaining size of the TCP-send-buffer;</li> <li>&lt;queue number&gt;: available TCP queue number; it's not reliable and should be used as a reference only.</li> </ul>
Notes	This command can not be used for SSL connection.
Example	For example, in single connection, the command AT+CIPBUFSTATUS returns: <b>20,15,10,200,7</b> Description: <ul style="list-style-type: none"> <li><b>20</b>: means that the latest segment ID is 19; so when calling AT+CIPSENDERBUF the next time, the segment ID returned is 20;</li> <li><b>15</b>: means that the TCP segment with the ID 15 is the last segment sent, but the segment may not be successfully sent;</li> <li><b>10</b>: means that the TCP segment with the ID 10 was sent successfully;</li> <li><b>200</b>: means that the remaining size of the TCP-send-buffer is 200 bytes;</li> <li><b>7</b>: the available TCP queue number; it is not reliable and should be used as a reference only; when the queue number is 0, no TCP data can be sent.</li> </ul>

### 5.2.10. AT+CIPCHECKSEQ—Checks If a Specific Segment Was Successfully Sent

Set Command	<ol style="list-style-type: none"> <li>Single connection: (+CIPMUX=0) AT+CIPCHECKSEQ=&lt;segment ID&gt;</li> <li>multiple connections: (+CIPMUX=1) AT+CIPCHECKSEQ=&lt;link ID&gt;,&lt;segment ID&gt;</li> </ol>
Response	[<link ID>,<segment ID>,<status> OK



<b>Parameters</b>	<ul style="list-style-type: none"> <li>• The command can only be used to record the status of the last 32 segments at most.</li> <li>• [<b>&lt;link ID&gt;</b>]: ID of the connection (0~4), for multiple connection;</li> <li>• <b>&lt;segment ID&gt;</b>: the segment ID obtained by calling AT+CIPSENBUFF;</li> <li>• <b>&lt;status&gt;</b>: <ul style="list-style-type: none"> <li>▸ FALSE: the segment-sending failed;</li> <li>▸ TRUE: the segment was sent successfully.</li> </ul> </li> </ul>
<b>Notes</b>	This command can only be used when AT+CIPSENBUFF is used.

### 5.2.11. AT+CIPCLOSE—Closes the TCP/UDP/SSL Connection

<b>Commands</b>	Set Command (used in multiple connections): AT+CIPCLOSE=<link ID> Function: closes the TCP/UDP Connection.	Execute Command (used in multiple connections): AT+CIPCLOSE
<b>Response</b>	OK	
<b>Parameters</b>	<link ID>: ID of the connection to be closed. When ID is 5, all connection will be closed. (In server mode, the ID 5 has no effect.)	-

### 5.2.12. AT+CIFSR—Gets the Local IP Address

<b>Execute Command</b>	AT+CIFSR	
<b>Response</b>	+CIFSR:APIP,<SoftAP IP address> +CIFSR:APMAC,<SoftAP MAC address> +CIFSR:STAIP,<Station IP address> +CIFSR:STAMAC,<Station MAC address> OK	
<b>Parameters</b>	<IP address>: IP address of the ESP8266 SoftAP; IP address of the ESP8266 Station. <MAC address>: MAC address of the ESP8266 SoftAP; MAC address of the ESP8266 Station.	
<b>Notes</b>	Only when the ESP8266 Station is connected to an AP can the Station IP can be queried.	

### 5.2.13. AT+CIPMUX—Enable or Disable Multiple Connections

<b>Commands</b>	Query Command: AT+CIPMUX?	Set Command: AT+CIPMUX=<mode> Function: to set the connection type.
-----------------	------------------------------	---



<b>Response</b>	+CIPMUX: <mode> OK	OK
<b>Parameters</b>	<mode>: <ul style="list-style-type: none"> <li>▶ 0: single connection</li> <li>▶ 1: multiple connections</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The default mode is single connection mode.</li> <li>• Multiple connections can only be set when transparent transmission is disabled (AT+CIPMODE=0).</li> <li>• This mode can only be changed after all connections are disconnected.</li> <li>• If the TCP server is running, it must be deleted (AT+CIPSERVER=0) before the single connection mode is activated.</li> </ul>	
<b>Example</b>	AT+CIPMUX=1	

#### 5.2.14. AT+CIPSERVER—Deletes/Creates TCP Server

<b>Set Command</b>	AT+CIPSERVER=<mode>[,<port>]	
<b>Response</b>	OK	
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;mode&gt;:  <ul style="list-style-type: none"> <li>▶ 0: deletes server.</li> <li>▶ 1: creates server.</li> </ul> </li> <li>• &lt;port&gt;: port number; 333 by default.</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• A TCP server can only be created when multiple connections are activated (AT+CIPMUX=1).</li> <li>• A server monitor will automatically be created when the TCP server is created.</li> <li>• When a client is connected to the server, it will take up one connection and be assigned an ID.</li> </ul>	
<b>Example</b>	AT+CIPMUX=1 AT+CIPSERVER=1,1001	

#### 5.2.15. AT+CIPSERVERMAXCONN—Set the Maximum Connections Allowed by Server

<b>Commands</b>	Query Command: AT+CIPSERVERMAXCONN?  Function: obtain the maximum number of clients allowed to connect to the TCP or SSL server.	Set Command: AT+CIPSERVERMAXCONN=<num>  Function: set the maximum number of clients allowed to connect to the TCP or SSL server.
<b>Response</b>	+CIPSERVERMAXCONN: <num> OK	OK
<b>Parameters</b>	<num>: the maximum number of clients allowed to connect to the TCP or SSL server, range: [1, 5]	
<b>Notes</b>	To set this configuration, you should call the command AT+CIPSERVERMAXCONN=<num> before creating a server.	



<b>Example</b>	<pre>AT+CIPMUX=1 AT+CIPSERVERMAXCONN=2 AT+CIPSERVER=1,80</pre>
----------------	--

### 5.2.16. AT+CIPMODE—Sets Transmission Mode

<b>Commands</b>	Query Command: AT+CIPMODE? Function: to obtain information about transmission mode.	Set Command: AT+CIPMODE=<mode> Function: to set the transmission mode.
<b>Response</b>	<pre>+CIPMODE:&lt;mode&gt; OK</pre>	OK
<b>Parameters</b>	<mode>: <ul style="list-style-type: none"> <li>▶ 0: normal transmission mode.</li> <li>▶ 1: UART-Wi-Fi passthrough mode (transparent transmission), which can only be enabled in TCP single connection mode or in UDP mode when the remote IP and port do not change.</li> </ul>	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• The configuration changes will NOT be saved in flash.</li> <li>• During the UART-Wi-Fi passthrough transmission, if the TCP connection breaks, ESP8266 will keep trying to reconnect until +++ is input to exit the transmission. If it is a normal TCP transmission and the TCP connection breaks, ESP8266 will give a prompt and will not attempt to reconnect.</li> </ul>	
<b>Example</b>	AT+CIPMODE=1	

### 5.2.17. AT+SAVETRANSLINK—Saves the Transparent Transmission Link in Flash

#### Save TCP Single Connection in Flash

<b>Set Command</b>	AT+SAVETRANSLINK=<mode>,<remote IP or domain name>,<remote port>[,<type>,<TCP keep alive>]
<b>Response</b>	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;mode&gt;:             <ul style="list-style-type: none"> <li>▶ 0: ESP8266 will NOT enter UART-Wi-Fi passthrough mode on power-up.</li> <li>▶ 1: ESP8266 will enter UART-Wi-Fi passthrough mode on power-up.</li> </ul> </li> <li>• &lt;remote IP&gt;: remote IP or domain name.</li> <li>• &lt;remote port&gt;: remote port.</li> <li>• [&lt;type&gt;] (optional): TCP or UDP, TCP by default.</li> <li>• [&lt;TCP keep alive&gt;] (optional): TCP is kept alive. This function is disabled by default.             <ul style="list-style-type: none"> <li>▶ 0: disables the TCP keep-alive function.</li> <li>▶ 1 ~ 7200: keep-alive detection time interval; unit: second (s).</li> </ul> </li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• This command will save the UART-Wi-Fi passthrough mode and its link in the flash. ESP8266 will enter the UART-Wi-Fi passthrough mode on any subsequent power cycles.</li> <li>• As long as the remote IP (or domain name) and port are valid, the configuration will be saved in the flash.</li> </ul>



<b>Example</b>	AT+SAVETRANSLINK=1, "192.168.6.110", 1002, "TCP"
----------------	--

#### Save UDP Transmission in Flash

<b>Set Command</b>	AT+SAVETRANSLINK=<mode>,<remote IP>,<remote port>,<type>[,<UDP local port>]
<b>Response</b>	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;mode&gt;: <ul style="list-style-type: none"> <li>▸ 0: normal mode; ESP8266 will NOT enter UART-Wi-Fi passthrough mode on power-up.</li> <li>▸ 1: ESP8266 enters UART-Wi-Fi passthrough mode on power-up.</li> </ul> </li> <li>• &lt;remote IP&gt;: remote IP or domain name.</li> <li>• &lt;remote port&gt;: remote port.</li> <li>• [&lt;type&gt;] (optional): UDP; TCP by default.</li> <li>• [&lt;UDP local port&gt;] (optional): local port when UDP transparent transmission is enabled on power-up.</li> </ul>
<b>Notes</b>	<ul style="list-style-type: none"> <li>• This command will save the UART-Wi-Fi passthrough mode and its link in the flash. ESP8266 will enter the UART-Wi-Fi passthrough mode on any subsequent power cycles.</li> <li>• As long as the remote IP (or domain name) and port are valid, the configuration will be saved in the user parameter area in the flash.</li> </ul>
<b>Example</b>	AT+SAVETRANSLINK=1, "192.168.6.110", 1002, "UDP", 1005

#### 5.2.18. AT+CIPSTO—Sets the TCP Server Timeout

<b>Commands</b>	Query Command: AT+CIPSTO? Function: to check the TCP server timeout.	Set Command: AT+CIPSTO=<time> Function: to set the TCP server timeout.
<b>Response</b>	+CIPSTO:<time> OK	OK
<b>Parameter</b>	<time>: TCP server timeout within the range of 0 ~ 7200s.	
<b>Notes</b>	<ul style="list-style-type: none"> <li>• ESP8266 configured as a TCP server will disconnect from the TCP client that does not communicate with it until timeout.</li> <li>• If AT+CIPSTO=0, the connection will never time out. This configuration is not recommended.</li> </ul>	
<b>Example</b>	AT+CIPMUX=1 AT+CIPSERVER=1, 1001 AT+CIPSTO=10	

#### 5.2.19. AT+PING—Ping Packets

<b>Set Command</b>	AT+PING=<IP> Function: Ping packets.
--------------------	---



<b>Response</b>	<code>+&lt;time&gt;</code> OK or <code>+timeout</code> ERROR
<b>Parameters</b>	<ul style="list-style-type: none"><li>• <code>&lt;IP&gt;</code>: string; host IP or domain name</li><li>• <code>&lt;time&gt;</code>: the response time of ping</li></ul>
<b>Notes</b>	<code>AT+PING="192.168.1.1"</code> <code>AT+PING="www.baidu.com"</code>



### 5.2.20. AT+CIUPDATE—Updates the Software Through Wi-Fi

Execute Command	AT+CIUPDATE Function: updates software.
Response	+CIUPDATE:<n> OK
Parameters	<ul style="list-style-type: none"> <li>• &lt;n&gt;: <ul style="list-style-type: none"> <li>▶ 1: find the server.</li> <li>▶ 2: connect to server.</li> <li>▶ 3: get the software version.</li> <li>▶ 4: start updating.</li> </ul> </li> </ul>
Notes	<ul style="list-style-type: none"> <li>• The speed of the upgrade is susceptible to the connectivity of the network.</li> <li>• ERROR will be returned if the upgrade fails due to unfavourable network conditions. Please wait for some time before retrying.</li> </ul>
Notes	<ul style="list-style-type: none"> <li>• If using Espressif's AT BIN (<i>/ESP8266_NONOS_SDK/bin/at</i>), AT+CIUPDATE will download a new AT BIN from the Espressif Cloud.</li> <li>• If using a user-compiled AT BIN, users need to make their own AT+CIUPDATE upgrade. Espressif provides a demo as a reference for local upgrade (<i>/ESP8266_NONOS_SDK/example/at</i>).</li> <li>• It is suggested that users call AT+RESTORE to restore the factory default settings after upgrading the AT firmware.</li> </ul>

### 5.2.21. AT+CIPDINFO—Shows the Remote IP and Port with +IPD

Set Command	AT+CIPDINFO=<mode>
Response	OK
Parameters	<mode>: <ul style="list-style-type: none"> <li>▶ 0: does not show the remote IP and port with +IPD.</li> <li>▶ 1: shows the remote IP and port with +IPD.</li> </ul>
Example	AT+CIPDINFO=1

### 5.2.22. +IPD—Receives Network Data

Command	Single connection: (+CIPMUX=0)+IPD,<len>[,<remote IP>,<remote port>]:<data>	multiple connections: (+CIPMUX=1)+IPD,<link ID>,<len>[,<remote IP>,<remote port>]:<data>
Parameters	The command is valid in normal command mode. When the module receives network data, it will send the data through the serial port using the +IPD command. <ul style="list-style-type: none"> <li>• [&lt;remote IP&gt;]: remote IP, enabled by command AT+CIPDINFO=1.</li> <li>• [&lt;remote port&gt;]: remote port, enabled by command AT+CIPDINFO=1.</li> <li>• &lt;link ID&gt;: ID number of connection.</li> <li>• &lt;len&gt;: data length.</li> <li>• &lt;data&gt;: data received.</li> </ul>	





### 5.2.23. AT+CIPSNTPCFG—Sets the Configuration of SNTP

<b>Commands</b>	Query Command: AT+CIPSNTPCFG?	Set Command: AT+CIPSNTPCFG=<enable>[,<timezone>][,<SNTP server0>,<SNTP server1>,<SNTP server2>]
<b>Response</b>	+CIPSNTPCFG:<enable>,<timezone>,<SNTP server1>[,<SNTP server2>,<SNTP server3>] OK	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;enable&gt;: <ul style="list-style-type: none"> <li>▸ 0: SNTP is disabled;</li> <li>▸ 1: SNTP is enabled.</li> </ul> </li> <li>• &lt;timezone&gt;: time zone; range: [-11,13]; if SNTP is enabled, the &lt;timezone&gt; has to be set;</li> <li>• &lt;SNTP server0&gt;: optional parameter indicating the first SNTP server;</li> <li>• &lt;SNTP server1&gt;: optional parameter indicating the second SNTP server;</li> <li>• &lt;SNTP server2&gt;: optional parameter indicating the third SNTP server.</li> </ul>	
<b>Example</b>	AT+CIPSNTPCFG=1,8,"cn.ntp.org.cn","ntp.sjtu.edu.cn","us.pool.ntp.org"	
<b>Note</b>	If the <SNTP server> parameters are not set, servers "cn.ntp.org.cn", "ntp.sjtu.edu.cn", "us.pool.ntp.org" will be used by default.	

### 5.2.24. AT+CIPSNTPTIME—Checks the SNTP Time

<b>Query Command</b>	AT+CIPSNTPTIME?
<b>Response</b>	+CIPSNTPTIME:<time> OK
<b>Parameters</b>	<time>: SNTP time For example, +CIPSNTPTIME:Thu Aug 04 14:48:05 2016 OK
<b>Example</b>	AT+CWMODE=1 //set station mode AT+CWJAP="DemoAP","password" //connect to router, access the internet AT+CIPSNTPCFG=1,8 //set time zone AT+CIPSNTPTIME? //get time

### 5.2.25. AT+CIPDNS\_CUR—Sets User-defined DNS Servers; Configuration Not Saved in the Flash

<b>Commands</b>	Query Command: AT+CIPDNS_CUR?  Function: Get the current DNS server.	Set Command: AT+CIPDNS_CUR=<enable>[,<DNS server0>,<DNS server1>]  Function: Set user-defined DNS servers.
-----------------	---	---



<b>Response</b>	[+CIPDNS_CUR:<DNS server0>] [+CIPDNS_CUR:<DNS server1>] OK	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;enable&gt;: <ul style="list-style-type: none"> <li>▸ 0: disable to use user-defined DNS servers;</li> <li>▸ 1: enable to use user-defined DNS servers.</li> </ul> </li> <li>• &lt;DNS server0&gt;: optional parameter indicating the first DNS server;</li> <li>• &lt;DNS server1&gt;: optional parameter indicating the second DNS serve.</li> </ul>	
<b>Example</b>	AT+CIPDNS_CUR=1, "208.67.220.220"	
<b>Note</b>	<ul style="list-style-type: none"> <li>• For command: AT+CIPDNS_CUR=0 (disable to use user-defined DNS servers), "208.67.222.222" will be used as DNS server by default. And the DNS server may change according to the configuration of the router which the chip connected to.</li> <li>• For command: AT+CIPDNS_CUR=1 (enable to use user-defined DNS servers, but the &lt;DNS server&gt; parameters are not set), servers "208.67.222.222" will be used as DNS server by default.</li> </ul>	

### 5.2.26. AT+CIPDNS\_DEF—Sets User-defined DNS Servers; Configuration Saved in the Flash

<b>Commands</b>	Query Command: AT+CIPDNS_DEF?  Function: Get the user-defined DNS servers which saved in flash.	Set Command: AT+CIPDNS_DEF=<enable>[,<DNS server0>,<DNS server1>]  Function: Set user-defined DNS servers.
<b>Response</b>	[+CIPDNS_DEF:<DNS server0>] [+CIPDNS_DEF:<DNS server1>] OK	OK
<b>Parameters</b>	<ul style="list-style-type: none"> <li>• &lt;enable&gt;: <ul style="list-style-type: none"> <li>▸ 0: disable to use a user-defined DNS server;</li> <li>▸ 1: enable to use a user-defined DNS server.</li> </ul> </li> <li>• &lt;DNS server0&gt;: optional parameter indicating the first DNS server;</li> <li>• &lt;DNS server1&gt;: optional parameter indicating the second DNS serve.</li> </ul>	
<b>Example</b>	AT+CIPDNS_DEF=1, "208.67.220.220"	
<b>Note</b>	<ul style="list-style-type: none"> <li>• This configuration will be saved in the user parameter area of flash.</li> <li>• For command: AT+CIPDNS_DEF=0 (disable to use user-defined DNS servers), "208.67.222.222" will be used as DNS server by default. And the DNS server may change according to the configuration of the router which the chip connected to.</li> <li>• For command: AT+CIPDNS_DEF=1 (enable to use user-defined DNS servers, but the &lt;DNS server&gt; parameters are not set), servers "208.67.222.222" will be used as DNS server by default.</li> </ul>	



## 6.

## Appendix A

ESP8266 AT commands below will save the configuration changes in flash:

AT Command	Examples
<b>Configuration Saved in the User Parameter Area in the Flash</b>	
AT+UART_DEF	AT+UART_DEF=115200,8,1,0,3
AT+CWDHCP_DEF	AT+CWDHCP_DEF=1,1
AT+CIPSTAMAC_DEF	AT+CIPSTAMAC_DEF="18:fe:35:98:d3:7b"
AT+CIPAPMAC_DEF	AT+CIPAPMAC_DEF="1a:fe:36:97:d5:7b"
AT+CIPSTA_DEF	AT+CIPSTA_DEF="192.168.6.100"
AT+CIPAP_DEF	AT+CIPAP_DEF="192.168.5.1"
AT+CWDHCPS_DEF	AT+CWDHCPS_DEF=1,3,"192.168.4.10","192.168.4.15"
AT+SAVETRANSLINK	AT+SAVETRANSLINK_DEF=1,"192.168.6.10",1001
AT+CIPDNS_DEF	AT+CIPDNS_DEF=1,"208.67.220.220"
AT+RFAUTOTRACE	AT+RFAUTOTRACE=0
AT+SYSMSG_DEF	AT+SYSMSG_DEF=3
AT+CWCOUNTRY_DEF	AT+CWCOUNTRY_DEF=1,"CN",1,5
<b>Configuration Saved in the System Parameter Area in the Flash</b>	
AT+CWMODE_DEF	AT+CWMODE_DEF=3
AT+CWJAP_DEF	AT+CWJAP_DEF="abc","0123456789"
AT+CWSAP_DEF	AT+CWSAP_DEF="ESP8266","12345678",5,3
AT+CWAUTOCONN	AT+CWAUTOCONN=1

**⚠ Notice:**

- Only when the configuration changes will the AT firmware write the new configuration into the flash. Therefore, users need not be concerned about wearing out the flash on repeated application of commands that set the same default configurations over and over again.
- For 512 KB + 512 KB Flash Map, the user parameter area is 0x7C000 ~ 0x80000, 16 KB;
- For 1024 KB + 1024 KB Flash Map: the user parameter area is 0xFC000 ~ 0x100000, 16 KB;
- The system parameter area is always the last 16 KB in the flash.



# 7. Appendix B

Messages of ESP8266 AT are as below:

Messages	Description
ready	The AT firmware is ready.
ERROR	AT command error, or error occurred during execution.
WIFI CONNECTED	ESP8266 station connected to an AP.
WIFI GOT IP	ESP8266 station got IP address.
WIFI DISCONNECT	ESP8266 station disconnected from an AP.
busy s...	Busy sending. The system is sending data now, cannot accept the newly input.
busy p...	Busy processing. The system is in process of handling the previous command, cannot accept the newly input.
<conn_id>,CONNECT	A network connection of which ID is <conn_id> is established.
<conn_id>,CLOSED	A network connection of which ID is <conn_id> ends.
+IPD	Received network data.
+STA_CONNECTED:<sta_mac>	A station connects to the ESP8266 softAP.
+DIST_STA_IP:<sta_mac>,<sta_ip>	ESP8266 softAP distributes an IP address to the station connected.
+STA_DISCONNECTED:<sta_mac>	A station disconnects from the ESP8266 softAP.



## 8.

## Q&A

If you have any questions about the execution of AT commands, please contact us via [Espressif Technical Inquiries](#). Please describe the issues that you might encounter, including any relevant details, as follows:

- AT Version information or AT Command: You can use command `AT+GMR` to acquire information on your current AT command version.
- Hardware Module information: for example, ESP-WROOM-02.
- Details of the test steps, for example:

```
AT+CWMODE_CUR=1
OK
AT+GMR
AT version:0.23.0.0(Apr 24 2015 21:11:01)
SDK version:1.0.1
compile time:Apr 24 2015 21:19:31
OK
AT+CIPSTAMAC_DEF="14:CF:11:22:33:05"
OK
```

- If possible, please provide the printed log information, such as:

```
ets Jan 8 2013,rst cause: 1, boot mode: (3,3)
load 0x40100000, len 26336, room 16
tail 0
chksum 0xde
load 0x3ffe8000, len 5672, room 8
tail 0
chksum 0x69
load 0x3ffe9630, len 8348, room 8
tail 4
chksum 0xcb
csum 0xcb
SDK version: 0.9.1
addr not ack when tx write cmd
mode : sta(18: fe: 34: 97: d5: 7b) + softAP(1a: fe: 34: 97: d5: 7b)
```



Espressif IOT Team  
[www.espressif.com](http://www.espressif.com)

#### **Disclaimer and Copyright Notice**

Information in this document, including URL references, is subject to change without notice.

THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to the use of information in this document, is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG.

All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.

**Copyright © 2018 Espressif Inc. All rights reserved.**