ESP8266 AT Instruction Set

Version 2.0.0

Espressif Systems IoT Team

Copyright © 2016

About This Guide

This document provides AT commands list based on ESP8266_NONOS_SDK.

The document is structured as follows.

Chapter	Title	Subject
Chapter 1	Preambles	Instructions on user-define AT command and downloading.
Chapter 2	Command Description	Basic description of AT commands.
Chapter 3	Basic AT Commands	List basic function AT commands.
Chapter 4	Wi-Fi Related AT Commands	Lists Wi-Fi related AT commands.
Chapter 5	TCP/IP AT Commands	Lists TCP/IP related AT commands.
Chapter 6	Appendix	List the AT commands which will save configuration into flash.
Chapter 7	Q & A	Provides information about AT technical support.

Release Notes

Date	Version	Release notes
2016.04	V1.5.3	First Release
2016.05	V1.5.4	Updated Chapter 5.2.16 and Chapter 5.2.19
2016.07	V2.0.0	Added Chapter 3.2.11, updated Chapter 1.2

Table of Contents

1.   Preambles

AT command set is divided into: Basic AT commands, Wi-Fi AT commands, TCP/IP AT commands.

1.1.   User-defined AT commands

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

AT firmware is based on the ESP8266_NONOS_SDK. Espressif Systems AT commands are provided in libat.a. In the example of \ESP8266_NONOS_SDK\examples\at\user\user_main.c, ways are delivered on how to implement a custom, user-defined AT Command, " AT+TEST".

The structure, **at_funcationType**, is used to define four types of a command, e.g. "AT+TEST".

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 theto folder "app" in ESP8266_NONOS_SDK if users need to compile the AT firmware.
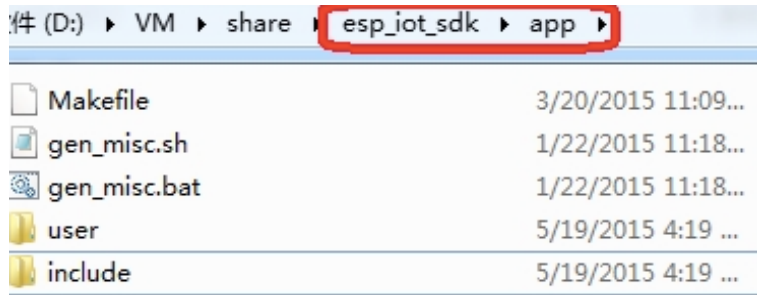
Please refer to ESP8266_Getting_Started_Guide for instructions on compiling applications.

1.2. Downloading AT firmware into Flash

Refer to **ESP8266_NONOS_SDK\bin\at\readme.txt** for instructions on how to download AT firmware into Flash. Please use Espressif official Flash download tool and select corresponding Flash size to download the firmware.

Espressif official Flash download tool: http://bbs.espressif.com/viewtopic.php?f=5&t=433

1.   4Mbit Flash

With the release of ESP8266_NONOS_SDK_V2.0.0, AT_V1.3, AT firmware can use 4Mbit Flash without FOTA (upgrade AT firmware through Wi-Fi) function.

bin	Address	Description
blank.bin	0x78000	Initialize RF_CAL parameter area
esp_init_data_default.bin	0x7C000	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	0x7A000	Initialize Flash user parameter area, more details in Appendix.
blank.bin	0x7E000	Initialize Flash system parameter area, more details in Appendix.
eagle.flash.bin	0x00000	In \bin\at\noboot
eagle.irom0text.bin	0x10000	In \bin\at\noboot

2.   8Mbit Flash

If the Flash size is 8Mbit 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: 8Mbit.

bin	Address	Description
blank.bin	0xFB000	Initialize RF_CAL parameter area
esp_init_data_default.bin	0xFC000	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	Initialize Flash user parameter area, more details in Appendix.
blank.bin	0xFE000	Initialize Flash system parameter area, more details in Appendix.
boot.bin	0x00000	In \bin\at
user1.1024.new.2.bin	0x01000	In \bin\at\512+512

3. 16Mbit Flash, map: 512KB + 512KB

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

bin	Address	Description
blank.bin	0x1FB000	Initialize 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	Initialize Flash user parameter area, more details in Appendix.
blank.bin	0x1FE000	Initialize Flash system parameter area, more details in Appendix.
boot.bin	0x00000	In \bin\at
user1.1024.new.2.bin	0x01000	In \bin\at\512+512

4. 16Mbit Flash, map: 1024KB + 1024KB

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

bin	Address	Description
blank.bin	0x1FB000	Initialize 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	Initialize Flash user parameter area, more details in Appendix.
blank.bin	0x1FE000	Initialize Flash system parameter area, more details in Appendix.
boot.bin	0x00000	In \bin\at
user1.2048.new.5.bin	0x01000	In \bin\at\1024+1024

5. 32Mbit Flash, map: 512KB + 512KB

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

bin	Address	Description
blank.bin	0x3FB000	Initialize 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	Initialize Flash user parameter area, more details in Appendix.
blank.bin	0x3FE000	Initialize Flash system parameter area, more details in Appendix.
boot.bin	0x00000	In \bin\at
user1.1024.new.2.bin	0x01000	In \bin\at\512+512

6. 32Mbit Flash, map: 1024KB + 1024KB

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

bin	Address	Description
blank.bin	0x3FB000	Initialize RF_CAL parameter area

bin	Address	Description
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	Initialize Flash user parameter area, more details in Appendix.
blank.bin	0x3FE000	Initialize Flash system parameter area, more details in Appendix.
boot.bin	0x00000	In \bin\at
user1.2048.new.5.bin	0x01000	In \bin\at\1024+1024

Notes:

- Please make sure that correct BIN (/ESP8266_NONOS_SDK/bin/at) is already in the chip (ESP8266) before the AT commands listed in this documentation can be used.

- 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 Wi-Fi connection of ESP8266 station

  ‣ WiFi CONNECTED - Wi-Fi connected

  ‣ WiFi GOT IP - ESP8266 station got IP from AP

  ‣ WiFi DISCONNECT - Wi-Fi disconnected

2.   Command Description

Each Command set contains four types of AT commands.

Type	Command Format	DescrIPtion
Test	AT+<x>=?	Query the Set command or internal parameters and its range values.
Query	AT+<x>?	Returns the current value of the parameter.
Set	AT+<x>=<…>	Set the value of user-defined parameters in commands and run.
Execute	AT+<x>	Runs commands with no user-defined parameters.

Notes:

1.   Not all AT commands support all the above mentioned four variations.

2.   [] = default value, not required or may not appear

3.   String values require double quotation marks, for example:

AT+CWSAP="ESP756290", "21030826", 1, 4

4.   Default Baudrate = 115200

5.   AT commands have to be capitalized, and must terminate with "/r/n". Terminal programs must be set up accordingly (refer to screenshot below).
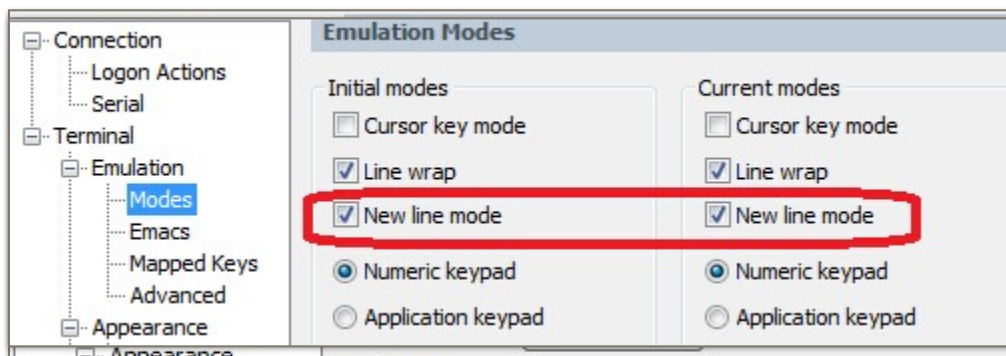
3.   Basic AT Commands

3.1.   Overview

The ESP8266 wireless Wi-Fi modules can be driven via the serial interface using the standard AT commands. Here is a list of some basic AT commands that can be used.

Basic	
Command	Description
AT	Test AT startup
AT+RST	Restart module
AT+GMR	View version info
AT+GSLP	Enter deep-sleep mode
ATE	AT commands echo or not
AT+RESTORE	Factory reset
AT+UART	UART configuration, [@deprecated]
AT+UART_CUR	UART current configuration
AT+UART_DEF	UART default configuration, save to flash
AT+SLEEP	Sleep mode
AT+WAKEUPGPIO	Set a GPIO to wake ESP8266 up from light-sleep mode
AT+RFPOWER	Set maximum value of RF TX Power
AT+RFVDD	Set RF TX Power according to VDD33

3.2. Commands

1. AT – Test AT startup

The type of this command is "executed". It is used to test the setup function of your wireless Wi-Fi module.

AT – Test AT startup	
Response	OK
Parameters	null

2. AT+RST – Restart module

The type of this command is "executed". It's used to restart the module.

AT+RST – Restart module	
Response	OK
Parameters	null

3. AT+GMR – View version info

This AT command is used to check the version of AT commands firmware installed and the SDK that it uses, the type of which is "executed".

AT+GMR – View version info	
Response	<AT version info> <SDK version info> <compile time> OK
Parameters	<AT version info> information about AT version <SDK version info> information about SDK version <compile time> time of the bin was compiled

4. AT+GSLP – Enter deep-sleep mode

This command is used to invoke the deep-sleep mode of the module, the type of which is "set". A minor adjustment has to be made before the module enter this deep sleep mode, i.e., connect XPD_DCDC with EXT_RSTB via **0 ohm resistor**.

AT+GSLP=<time>	
Response	<time> OK
Parameters	The time unit of <time> is ms. ESP8266 will wake up after deep sleep <time> ms.

5. ATE – AT commands echo

This command ATE is an AT 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.

ATE – AT commands echo	
Response	OK
Parameters	ATE0: Switch echo off ATE1: Switch echo on

6. AT+RESTORE – Factory reset

This command is used to reset all parameters saved in flash (according to appendix), restore the factory default settings of the module. The chip will be restarted when this command is executed.

AT+RESTORE – Factory reset	
Response	OK
Notes	Restore factory default settings. The chip will restart.

7.   AT+UART – UART configuration

This command sets the UART configuration and writes the new configuration to the flash. It is stored as the default parameter and will also be used as the default baudrate henceforth. [THIS API IS DEPRECATED.]

AT+UART=<baudrate>, <databits>, <stopbits>, <parity>, <flow control>		
This command is deprecated, please use AT+UART_CUR or AT+UART_DEF instead.		
Example	AT+UART=115200, 8, 1, 0, 3	
Response	OK	
Parameters	<baudrate>	Baudrate range: 110 to 115200*40 (4.608 Mega)
	<databits>	5: 5 bits data 6: 6 bits data 7: 7 bits data 8: 8 bits data
	<stopbits>	1: 1 bit stop bit 2: 1.5 bit stop bit 3: 2 bit stop bit
	<parity>	0: None 1: Odd 2: EVEN
	<flow control>	0: disable flow control 1: enable RTS 2: enable CTS 3: enable both RTS and CTS
Notes	• This configuration will also store the baudrate as the default rate in the user parameter area in the flash for boot up. • Flow control needs hardware support: MTCK is UART0 CTS and MTDO is UART0 RTS.	

8. AT+UART_CUR – current UART configuration

This command sets the current UART configuration, it does not write changes to the flash. Hence there is no change in the default baud rate.

AT+UART_CUR=<baudrate>, <databits>, <stopbits>, <parity>, <flow control>		
Example	AT+UART_CUR=115200, 8, 1, 0, 3	
Response	OK	
Parameters	<baudrate>	Baudrate range: 110 to 115200*40 (4.608 Mega)
	<databits>	5: 5 bits data 6: 6 bits data 7: 7 bits data 8: 8 bits data
	<stopbits>	1: 1 bit stop bit 2: 1.5 bit stop bit 3: 2 bit stop bit
	<parity>	0: None 1: Odd 2: EVEN
	<flow control>	0: disable flow control 1: enable RTS 2: enable CTS 3: enable both RTS and CTS
Notes	• This configuration will NOT store in the flash. • Flow control needs hardware support: MTCK is UART0 CTS and MTDO is UART0 RTS.	

9. AT+UART_DEF – default UART configuration

This command sets the UART configuration and saves it to flash. It is stored as the default parameter and will also be used as the default baud rate henceforth.

AT+UART_DEF=<baudrate>, <databits>, <stopbits>, <parity>, <flow control>		
Example	AT+UART_DEF=115200, 8, 1, 0, 3	
Response	OK	
Parameters	<baudrate>	Baudrate range: 110 to 115200*40 (4.608 Mega)
	<databits>	5: 5 bits data 6: 6 bits data 7: 7 bits data 8: 8 bits data
	<stopbits>	1: 1 bit stop bit 2: 1.5 bit stop bit 3: 2 bit stop bit
	<parity>	0: None 1: Odd 2: EVEN
	<flow control>	0: disable flow control 1: enable RTS 2: enable CTS 3: enable both RTS and CTS
Notes	• This configuration will be stored in flash user parameter area for boot up. • Flow control needs hardware support: MTCK is UART0 CTS and MTDO is UART0 RTS.	

10. AT+SLEEP – sleep mode

This command sets ESP8266 sleep mode. It can only be used in station mode. Modem sleep is the default sleep mode.

AT+SLEEP – sleep mode	
Command	AT+SLEEP?
Response	+SLEEP : <sleep mode> OK

Parameters	<sleep mode>
	    0 : disable sleep mode
	    1 : light-sleep mode
	    2 : modem-sleep mode

Command	AT+SLEEP=<sleep mode>
Response	OK
Parameters	The same as above.

11. AT+WAKEUPGPIO – set a GPIO to wake ESP8266 up from light-sleep mode

This command sets a GPIO to wake the ESP8266 up from light-sleep mode. Please note that the <trigger_GPIO> and the <awake_GPIO> in the command cannot use the same GPIO.

After being waken up by <trigger_GPIO>, when the ESP8266 attempts to sleep again, it will check the status of the <trigger_GPIO>:

• if it is still in the wakeup status, the EP8266 will enter modem-sleep mode instead.

• if it is NOT in the wakeup status, the ESP8266 will enter light-sleep mode.

AT+WAKEUPGPIO – set a GPIO to wake ESP8266 up from light-sleep mode	
Example	ESP8266 will be wakened from light-sleep, when the GPIO0 is in low-level. AT+WAKEUPGPIO=1,0,0 ESP8266 will be wakened from light-sleep, when the GPIO0 is in high-level. And after waking up, the GPIO13 will be set to high-level. AT+WAKEUPGPIO=1,0,1,13,1 Disable the function that ESP8266 can be woken up from light-sleep by GPIO. AT+WAKEUPGPIO=0
Command	AT+WAKEUPGPIO=<enable>,<trigger_GPIO>,<trigger_level>, [<awake_GPIO>,<awake_level>]
Response	OK

Parameters	<enable>
	     0 : ESP8266 can NOT be woken up from light-sleep by GPIO
	     1 : ESP8266 can be woken up from light-sleep by GPIO
	 <trigger_GPIO>
	     set a GPIO to wake ESP8266 up, range: [0, 15]
	 <trigger_level>
	     0 : low-level to trigger the wakeup process
	     1 : high-level to trigger the wakeup process
	 [<awake_GPIO>], optional
	     set a GPIO as a flag of awakening
	 [<awake_level>], optional
	     0 : to be low-level after wakeup
	     1 : to be high-level after wakeup

12. AT+RFPOWER – set maximum value of RF TX Power

This command sets the maximum value of ESP8266 RF TX power, it is not precise.

AT+RFPOWER – set RF TX Power	
Example	AT+RFPOWER=50
Command	**AT+RFPOWER=<TX power>**
Response	OK
Parameters	<TX power> maximum value of RF TX power, range: [0, 82], unit: 0.25dBm

13. AT+RFVDD – set RF TX Power according to VDD33

This command sets ESP8266 RF TX power according to VDD33. To get the power voltage of ESP8266 VDD3P3, TOUT pin has to be suspended. TOUT pin has to be suspended in order to measure VDD33.

AT+RFVDD – set RF TX power according to VDD33	
Command	**AT+RFVDD?**
Response	+RFVDD:<VDD33> OK

Parameters	<VDD33>
	    power voltage of ESP8266 VDD33, unit: 1/1024 V
	    TOUT pin has to be suspended to measure VDD33.

Command	**AT+RFVDD=<VDD33>**
Response	OK
Parameters	<VDD33> power voltage of ESP8266 VDD33, range [1900, 3300]
Command	**AT+RFVDD**
Response	OK

Note	"AT+RFVDD" will automatically set RF TX power according to VDD33. TOUT pin has to be suspended.

4.   Wi-Fi AT Commands

Wi-Fi Functions Invoked by AT commands	
Command	**Description**
AT+CWMODE	Wi-Fi mode(sta/AP/sta+AP), [@deprecated]
AT+CWMODE_CUR	Wi-Fi mode(sta/AP/sta+AP) Settings not updated in flash.
AT+CWMODE_DEF	Wi-Fi default mode(sta/AP/sta+AP) Save to flash.
AT+CWJAP	Connect to AP, [@deprecated]
AT+CWJAP_CUR	Connect to AP, won't save to flash
AT+CWJAP_DEF	Connect to AP, save to flash
AT+CWLAPOPT	Set the configuration of command AT+CWLAP
AT+CWLAP	Lists available APs
AT+CWQAP	Disconnect from AP
AT+CWSAP	Set configuration of ESP8266 soft-AP [@deprecated]
AT+CWSAP_CUR	Set current configuration of ESP8266 soft-AP Settings not updated in flash.
AT+CWSAP_DEF	Set configuration of ESP8266 soft-AP Save changes to flash.
AT+CWLIF	Get station IP which is connected to ESP8266 soft-AP
AT+CWDHCP	Enable/Disable DHCP, [@deprecated]
AT+CWDHCP_CUR	Enable/Disable DHCP, changes not save to flash
AT+CWDHCP_DEF	Enable/Disable DHCP, save changes to flash
AT+CWDHCPS_CUR	Set IP range of DHCP server, changes not save to flash
AT+CWDHCPS_DEF	Set IP range of DHCP server, save changes to flash
AT+CWAUTOCONN	Connect to AP automatically on power-up

AT+CIPSTAMAC	Set MAC address of ESP8266 station [@deprecated]
AT+CIPSTAMAC_CUR	Set MAC address of ESP8266 station. Changes not save to flash.
AT+CIPSTAMAC_DEF	Set MAC address of ESP8266 station. Save changes to flash.
AT+CIPAPMAC	Set MAC address of ESP8266 soft-AP [@deprecated]
AT+CIPAPMAC_CUR	Set MAC address of ESP8266 soft-AP. Changes not save to flash.
AT+CIPAPMAC_DEF	Set MAC address of ESP8266 soft-AP Save changes to flash.
AT+CIPSTA	Set IP address of ESP8266 station, [@deprecated]
AT+CIPSTA_CUR	Set IP address of ESP8266 station Changes not save to flash.
AT+CIPSTA_DEF	Set IP address of ESP8266 station Save changes to flash.
AT+CIPAP	Set IP address of ESP8266 soft-AP, [@deprecated]
AT+CIPAP_CUR	Set IP address of ESP8266 soft-AP Changes not save to flash.
AT+CIPAP_DEF	Set IP address of ESP8266 soft-AP Save changes to flash.
AT+CWSTARTSMART	Start SmartConfig
AT+CWSTOPSMART	Stop SmartConfig
AT+CWSTARTDISCOVER	Start the mode that ESP8266 can be found by WeChat
AT+CWSTOPDISCOVER	Stop the mode that ESP8266 can be found by WeChat
AT+WPS	Set WPS function
AT+MDNS	Set MDNS function

4.1. Commands

1. AT+CWMODE – WiFi mode

The function of this AT command is to get the value scope of Wi-Fi mode, including station mode, softAP mode, and station+softAP mode, enquiry about the information of Wi-Fi mode, or set the Wi-Fi mode.

AT+CWMODE – WiFi mode	
This command is deprecated. Please use AT+CWMODE_CUR or AT+CWMODE_DEF instead.	
Command	AT+CWMODE=?
Response	+CWMODE:(value scope of \<mode>) OK
Parameters	Please refer to AT command settings.
Command	AT+CWMODE?
Response	+CWMODE:\<mode> OK
Parameters	Please refer to AT command settings.
Command	AT+CWMODE=\<mode>
Response	OK

Parameters	\<mode>
	    1 : station mode
	    2 : softAP mode
	    3 : softAP + station mode

Notes	This setting will be stored in the flash system parameter area. It won't be erased even when the power is off and restarted.

2. **AT+CWMODE_CUR – current WiFi mode**

There are three Wi-Fi working modes: Station mode, softAP mode, and the co-existence of Station mode and softAP mode. This command is used to query the current Wi-Fi mode, or to set a desired Wi-Fi mode.

AT+CWMODE_CUR – Set WiFi mode(sta/AP/sta+AP), won't save to flash	
Example	AT+CWMODE_CUR=3
Command	**AT+CWMODE_CUR=?**
Response	+CWMODE_CUR:(value scope of <mode>) OK

Parameters	<mode>
	    1 : station mode
	    2 : softAP mode
	    3 : softAP + station mode

Command	**AT+CWMODE_CUR?**
Response	+CWMODE_CUR:<mode> OK

Parameters	<mode>
	    1 : station mode
	    2 : softAP mode
	    3 : softAP + station mode

Command	**AT+CWMODE_CUR=<mode>**
Response	OK

Parameters	<mode>
	    1 : station mode
	    2 : softAP mode
	    3 : softAP + station mode

Notes	Configuration changes will NOT be stored in flash.

3.　AT+CWMODE_DEF – default WiFi mode

AT+CWMODE_DEF - WiFi mode (sta/AP/sta+AP), save to flash	
Example	AT+CWMODE_DEF=3
Command	**AT+CWMODE_DEF=?**
Response	+CWMODE_DEF:(value scope of <mode>) OK
Parameters	<mode> 　　1 :　station mode 　　2 :　softAP mode 　　3 :　softAP + station mode
Command	**AT+CWMODE_DEF?**
Response	+CWMODE_DEF:<mode> OK
Parameters	<mode> 　　1 :　station mode 　　2 :　softAP mode 　　3 :　softAP + station mode
Command	**AT+CWMODE_DEF=<mode>**
Response	OK
Parameters	<mode> 　　1 :　station mode 　　2 :　softAP mode 　　3 :　softAP + station mode
Notes	Configuration changes will be stored in flash system parameter area.

4.   AT+CWJAP – Connect to AP

AT+CWJAP – Connect to AP	
[@deprecated]. Please use AT+CWJAP_CUR or AT+CWJAP_DEF instead.	

Example	• AT+CWJAP ="abc", "0123456789"
	 • If SSID is "ab\, c" and password is "0123456789"\"
	     AT+CWJAP ="ab\\\, c", "0123456789\"\\"
	 • If several APs have the same SSID as "abc", target AP can be found by bssid:
	     AT+CWJAP ="abc", "0123456789", "ca:d7:19:d8:a6:44"

Command	**AT+CWJAP?**
Response	+CWJAP:\<ssid\>, \<bssid\>, \<channel\>, \<rssi\> OK
Parameters	\<ssid\> string, AP's SSID
Command	**AT+CWJAP=\<ssid\>, \<pwd\>[, \<bssid\>]**
Response	OK or +CWJAP:\<error code\> FAIL

Parameters	\<ssid\> string, AP's SSID
	 \<pwd\> string, MAX: 64 bytes ASCII
	 [\<bssid\>] string, AP's MAC address, for several APs may have the same SSID
	 \<error code\> only for reference, it is not reliable
	     1 : connection timeout
	     2 : wrong password
	     3 : cannot find target AP
	     4 : connection failed
	
	 This command requires station mode to be active.
	 Escape character syntax is needed if "SSID" or "password" contains any special characters (' , ' or ' " 'or ' \ ')

Notes	Configuration changes will be stored in flash system parameter area.

5. AT+CWJAP_CUR – Connect to AP, for current

AT+CWJAP_CUR – Connect to AP, won't save to flash	

Example	• AT+CWJAP_CUR ="abc", "0123456789"
	• If SSID is "ab\, c" and password is "0123456789"\"
	    AT+CWJAP_CUR="ab\\\, c", "0123456789\"\\"
	• If several APs have the same SSID as "abc", target AP can be found by bssid:
	    AT+CWJAP_CUR="abc", "0123456789", "ca:d7:19:d8:a6:44"

Command	**AT+CWJAP_CUR?**
Response	+CWJAP_CUR:<ssid>, <bssid>, <channel>, <rssi> OK
Parameters	<ssid> string, AP's SSID
Command	**AT+CWJAP_CUR=<ssid>, <pwd>[, <bssid>]**
Response	OK or +CWJAP:<error code> FAIL

Parameters	<ssid> string, AP's SSID
	<pwd> string, MAX: 64 bytes ASCII
	[<bssid>] string, AP's MAC address, for several APs may have the same SSID
	<error code> only for reference, it is not reliable
	    1 : connection timeout
	    2 : wrong password
	    3 : cannot find target AP
	    4 : connection failed
	
	This command requires station mode to be active.
	Escape character syntax is needed if "SSID" or "password" contains any special characters (' , ' or ' " 'or ' \ ')

Notes	Configuration changes will NOT be stored in flash

6. **AT+CWJAP_DEF – Connect to AP, save as default**

AT+CWJAP_DEF – Connect to AP and save AP info to flash	

Example	• AT+CWJAP_DEF="abc", "0123456789"
	• If SSID is "ab\, c" and password is "0123456789"\"
	    AT+CWJAP_DEF="ab\\\, c", "0123456789\"\\"
	• If several APs have the same SSID as "abc", target AP can be found by bssid:
	    AT+CWJAP_DEF ="abc", "0123456789", "ca:d7:19:d8:a6:44"

Command	AT+CWJAP_DEF?
Response	+CWJAP_DEF:<ssid>, <bssid>, <channel>, <rssi> OK
Parameters	<ssid> string, AP's SSID
Command	AT+CWJAP_DEF=<ssid>, <pwd>[, <bssid>]
Response	OK or +CWJAP:<error code> FAIL

Parameters	<ssid> string, AP's SSID
	<pwd> string, MAX: 64 bytes ASCII
	[<bssid>] string, AP's bssid(MAC address), for several APs may have the same SSID
	<error code> only for reference, it is not reliable
	    1 : connection timeout
	    2 : wrong password
	    3 : cannot find target AP
	    4 : connection failed
	
	This command requires station mode to be active.
	Escape character syntax is needed if "SSID" or "password" contains any special characters (' , ' or ' " 'or ' \ ')

Notes	Configuration changes will be stored in flash system parameter area.

7. AT+CWLAPOPT – Set configuration for command AT+CWLAP

This command is to set the configuration for command AT+CWLAP, whether the result of AT+CWLAP will be ordered according to <rssi>, and which parameters will be shown in the result of AT+CWLAP.

AT+CWLAPOPT - Set configuration for command AT+CWLAP	
Example	AT+CWLAPOPT=1, 127 The first parameter is 1, means that the result of command AT+CWLAP will be listed according to <rssi> The second parameter 127 which is 0x7F, means that all bits in <mask> are 1, so all parameters in the result of AT+CWLAP will be shown.
Command	**AT+CWLAPOPT =<sort_enable>, <mask>**
Response	OK

Parameters	<sort_enable>
	 whether the result of AT+CWLAP will be ordered according to <rssi>
	    0 : do not order by <rssi>
	    1 : order by <rssi>
	 <mask>
	 which parameters will be shown in the result of AT+CWLAP, 0 means that do not show the parameter corresponding to the bit, 1 means to show it.
	    bit 0 sets whether <ecn> will be shown in the result of AT+CWLAP ,
	    bit 1 sets whether <ssid> will be shown in the result of AT+CWLAP ,
	    bit 2 sets whether <rssi> will be shown in the result of AT+CWLAP ,
	    bit 3 sets whether <mac> will be shown in the result of AT+CWLAP ,
	    bit 4 sets whether <ch> will be shown in the result of AT+CWLAP ,
	    bit 5 sets whether <freq offset> will be shown in the result of AT+CWLAP ,
	    bit 6 sets whether <freq calibration> will be shown in the result of AT+CWLAP .

8. AT+CWLAP – List available APs

AT+CWLAP - Lists available APs	

Example	• AT+CWLAP
	    List of all available AP's detected by ESP8266
	• AT+CWLAP="WiFi", "ca:d7:19:d8:a6:44", 6
	    Find AP with specific SSID and MAC at specific channel.
	• AT+CWLAP="WiFi"
	    Find AP with specific SSID

Command	**AT+CWLAP=<ssid>[, <mac>, <ch>]**
Response	+CWLAP:<ecn>, <ssid>, <rssi>, <mac>, <ch>, <freq offset>, <freq calibration> OK

Parameters	<ecn>
	    0 : OPEN
	    1 : WEP
	    2 : WPA_PSK
	    3 : WPA2_PSK
	    4 : WPA_WPA2_PSK
	    5 : WPA2_Enterprise (AT can NOT connect to WPA2_Enterprise AP)
	<ssid> string, SSID of AP
	<rssi> signal strength
	<mac> string, MAC address
	<freq offset> frequency offset of AP, unit: KHz. The value of <freq offset> / 2.4 to get the value as ppm
	<freq calibration> calibration for frequency offset

Command	**AT+CWLAP**
Response	+CWLAP:<ecn>, <ssid>, <rssi>, <mac>, <ch>, <freq offset>, <freq calibration> OK
Parameters	The same as above

9. AT+CWQAP – Disconnect from AP

AT+CWQAP - Disconnect from AP	
Command	AT+ CWQAP
Response	OK
Parameters	null

10. AT+CWSAP – Configuration of softAP mode

AT+ CWSAP – Configuration of softAP mode [@deprecated]. Please use AT+CWSAP_CUR or AT+CWSAP_DEF instead.	
Example	AT+CWSAP="ESP8266", "1234567890", 5, 3
Command	AT+CWSAP?
Response	+CWSAP:<ssid>, <pwd>, <chl>, <ecn>, <max conn>, <ssid hidden>

Parameters	<ssid> string, ESP8266 softAP's SSID
	<pwd> string, range: 8 ~ 64 bytes ASCII
	<chl> channel ID
	<ecn>
	    0 : OPEN
	    2 : WPA_PSK
	    3 : WPA2_PSK
	    4 : WPA_WPA2_PSK
	<max conn>
	    maximum count of stations that are allowed to connect to ESP8266 soft-AP
	    range: [1, 4]
	<ssid hidden> Broadcast SSID by default
	    0 : broadcast SSID of ESP8266 soft-AP
	    1 : do not broadcast SSID of ESP8266 soft-AP

Command	AT+CWSAP=<ssid>, <pwd>, <chl>, <ecn>[, <max conn>][, <ssid hidden>]
Response	OK
Parameters	The same as above.
Notes	This CMD is only available when softAP is active. ESP8266 softAP does not support WEP. Configuration changes will be stored in flash system parameter area.

11. AT+CWSAP_CUR – Current config of softAP mode

AT+CWSAP_CUR – Set configuration of softAP mode, won't save to flash	
Example	AT+CWSAP_CUR="ESP8266", "1234567890", 5, 3
Command	AT+CWSAP_CUR?
Response	+CWSAP_CUR:<ssid>, <pwd>, <chl>, <ecn>, <max conn>, <ssid hidden>
Parameters	The same as below.
Command	AT+CWSAP_CUR=<ssid>, <pwd>, <chl>, <ecn>[, <max conn>][, <ssid hidden>]
Response	OK or ERROR // wrong parameter

Parameters	<ssid> string, ESP8266 softAP's SSID
	<pwd> string, range: 8 ~ 64 bytes ASCII
	<chl> channel ID
	<ecn>
	    0 : OPEN
	    2 : WPA_PSK
	    3 : WPA2_PSK
	    4 : WPA_WPA2_PSK
	[<max conn>], optional, default is 4
	    maximum count of stations that are allowed to connect to ESP8266 soft-AP
	    range: [1, 4]
	[<ssid hidden>], optional, broadcast SSID by default
	    0 : broadcast SSID of ESP8266 soft-AP
	    1 : do not broadcast SSID of ESP8266 soft-AP

Notes	This command is only available when softAP is active. ESP8266 softAP does not support WEP. Configuration changes will NOT be stored in flash.

12. AT+CWSAP_DEF – Default config of softAP mode

AT+ CWSAP_DEF – Set configuration of softAP mode, save to flash	
Example	AT+CWSAP_DEF="ESP8266", "1234567890", 5, 3
Command	**AT+CWSAP_DEF?**
Response	+CWSAP_DEF:<ssid>, <pwd>, <chl>, <ecn>, <max conn>, <ssid hidden>
Parameters	The same as below.
Command	**AT+CWSAP_DEF=<ssid>, <pwd>, <chl>, <ecn>[, <max conn>][, <ssid hidden>]**
Response	OK or ERROR // wrong parameter

Parameters	<ssid> string, ESP8266 softAP's SSID
	<pwd> string, range: 8 ~ 64 bytes ASCII
	<chl> channel ID
	<ecn>
	   0 : OPEN
	   2 : WPA_PSK
	   3 : WPA2_PSK
	   4 : WPA_WPA2_PSK
	[<max conn>], optional, default is 4
	   maximum count of stations that are allowed to connect to ESP8266 soft-AP
	   range: [1, 4]
	[<ssid hidden>], optional, broadcast SSID by default
	   0 : broadcast SSID of ESP8266 soft-AP
	   1 : do not broadcast SSID of ESP8266 soft-AP

Notes	This command is only available when softAP is active. ESP8266 softAP does not support WEP. Configuration changes will be stored in flash system parameter area.

13. AT+CWLIF – IP of stations

This command is used to get the IP of stations that are connected to ESP8266 softAP.

AT+ CWLIF– IP of stations which are connected to ESP8266 softAP	
Response	<IP addr>, <mac> OK
Parameters	<IP addr> IP address of stations which are connected to ESP8266 softAP <mac> MAC address of stations which are connected to ESP8266 softAP
Notes	This command cannot get static IP, it is only available if DHCP is enabled.

14. AT+CWDHCP – Enable/Disable DHCP

AT+ CWDHCP – Enable/Disable DHCP	
[@deprecated]. Please use AT+CWDHCP_CUR or AT+CWDHCP_DEF instead.	
Command	AT+CWDHCP?
Response	DHCP disabled or enabled now?

Parameters	Bit0 : 0 - soft-AP DHCP disabled
	        1 - soft-AP DHCP enabled
	bit1 : 0 - station DHCP disabled
	        1 - station DHCP enabled

Command	AT+CWDHCP=\<mode\>, \<en\>
Response	OK

Parameters	\<mode\>
	  0 : set ESP8266 soft-AP
	  1 : set ESP8266 station
	  2 : set both softAP and station
	\<en\>
	  0 : Disable DHCP
	  1 : Enable DHCP

Notes	• Configuration changes will be stored in flash user parameter area.
	• This configuration interacts with static IP related AT commands (AT+CIPSTA related and AT+CIPAP related):
	  ▸ If DHCP is enabled, static IP will be disabled;
	  ▸ If static IP is enabled, DHCP will be disabled;
	  ▸ This will depend on the last configuration.

15. AT+CWDHCP_CUR – Enable/Disable DHCP

AT+ CWDHCP_CUR – Enable/Disable DHCP, won't save to flash	
Command	AT+CWDHCP_CUR?
Response	DHCP disabled or enabled now?
Parameters	Bit0 : 0 - soft-AP DHCP disabled 　　　 1 - soft-AP DHCP enabled bit1 : 0 - station DHCP disabled 　　　 1 - station DHCP enabled
Command	AT+CWDHCP_CUR=<mode>, <en>
Response	OK
Parameters	<mode> 　　0 : set ESP8266 soft-AP 　　1 : set ESP8266 station 　　2 : set both softAP and station <en> 　　0 : Disable DHCP 　　1 : Enable DHCP
Notes	• This configuration will NOT be stored in flash user parameter area. • This configuration interacts with static IP related AT commands (AT+CIPSTA related and AT+CIPAP related): 　▸ If DHCP is enabled, static IP will be disabled; 　▸ If static IP is enabled, DHCP will be disabled; 　▸ This will depend on the last configuration.

16. AT+CWDHCP_DEF – Enable/Disable DHCP and save to flash

AT+ CWDHCP_DEF – Enable/Disable DHCP and save to flash	
Command	**AT+CWDHCP_DEF?**
Response	DHCP disabled or enabled currently?

Parameters	Bit0 : 0 - soft-AP DHCP disabled
	        1 - soft-AP DHCP enabled
	bit1 : 0 - station DHCP disabled
	        1 - station DHCP enabled

Command	**AT+CWDHCP_DEF=<mode>, <en>**
Response	OK

Parameters	<mode>
	   0 : set ESP8266 soft-AP
	   1 : set ESP8266 station
	   2 : set both softAP and station
	<en>
	   0 : Disable DHCP
	   1 : Enable DHCP

Notes	• Configuration changes will be stored in flash user parameter area.
	• This configuration interacts with static IP related AT commands (AT+CIPSTA related and AT+CIPAP related):
	  ▸ If DHCP is enabled, static IP will be disabled;
	  ▸ If static IP is enabled, DHCP will be disabled;
	  ▸ This will depend on the last configuration.

17. AT+CWDHCPS_CUR – Set the IP address allocated by ESP8266 soft-AP DHCP, does not save to flash

This AT command is enabled when ESP8266 runs as soft-AP, and when DHCP server is running normally. The IP address should be on the same network segment with the IP address of ESP8266 soft-AP. Configuration changes will not be stored in flash.

AT+CWDHCPS_CUR – Set the IP address allocated by ESP8266 soft-AP DHCP, not to be stored in flash	
Example	AT+CWDHCPS_CUR=1, 3, "192.168.4.10", "192.168.4.15" or AT+CWDHCPS_CUR=0 // Disable the settings and use the default IP range.
Command	**AT+CWDHCPS_CUR?**
Response	+CWDHCPS_CUR=<lease time>, <start IP>, <end IP>

Parameters	<enable>
	     0 : Disable the settings and use the default IP range.
	     1: Enable setting the IP range, parameters below have to be set.
	 <lease time> the unit of lease time is minute, range [1, 2880]
	 <start IP>
	     start IP of the IP range that can be obtained from ESP8266 soft-AP DHCP server
	 <end IP>
	      end IP of the IP range that can be obtained from ESP8266 soft-AP DHCP server

Command	**AT+CWDHCPS_CUR=<enable>, <lease time>, <start IP>, <end IP>**
Response	OK
Parameters	The same as above.

18. AT+CWDHCPS_DEF – Set the IP address allocated by ESP8266 soft-AP DHCP, save to flash

This AT command is enabled when ESP8266 runs as soft-AP, and when DHCP server is running normally. The IP address should be on the same network segment with the IP address of ESP8266 soft-AP. This configuration will be stored in flash user parameter area.

AT+CWDHCPS_DEF – Set the IP address allocated by ESP8266 soft-AP DHCP, can be stored in flash	
Example	AT+CWDHCPS_DEF=1, 3, "192.168.4.10", "192.168.4.15" or AT+CWDHCPS_DEF=0 // Disable the settings and use the default IP range.
Command	**AT+CWDHCPS_DEF?**
Response	+CWDHCPS_DEF=\<lease time>, \<start IP>, \<end IP>

Parameters	\<enable>
	    0 : Disable the settings and use the default IP range.
	    1: Enable setting the IP range, parameters below have to be set.
	 \<lease time> the unit of lease time is minute, range [1, 2880]
	 \<start IP>
	    start IP of the IP range that can be obtained from ESP8266 soft-AP DHCP server
	 \<end IP>
	    end IP of the IP range that can be obtained from ESP8266 soft-AP DHCP server

Command	**AT+CWDHCPS_DEF=\<enable>, \<lease time>, \<start IP>, \<end IP>**
Response	OK

19. AT+CWAUTOCONN – Auto connect to AP or not

ESP8266 station will connect to AP automatically on power-up by default.

AT+CWAUTOCONN – Connect to AP automatically or not	
Example	AT+CWAUTOCONN=1
Command	**AT+CWAUTOCONN=\<enable>**
Response	OK

Parameters	\<enable>
	    0 : do NOT auto-connect to AP on power-up
	    1 : connect to AP automatically on power-up

Notes	Configuration changes will be stored in flash system parameter area.

20. AT+CIPSTAMAC – Set MAC address of station

MAC addresses of ESP8266 soft-AP and station are different, please do NOT set them to be the same. Note that the bit 0 of the first byte of ESP8266 MAC address cannot be 1, for example, MAC address can be "18:fe:35:98:d3:7b", but cannot be "15:fe:35:98:d3:7b".

AT+ CIPSTAMAC – Set MAC address of ESP8266 station	
[@deprecated]. Use AT+CIPSTAMAC_CUR or AT+CIPSTAMAC_DEF instead.	
Example	AT+CIPSTAMAC="18:fe:35:98:d3:7b"
Command	**AT+CIPSTAMAC?**
Response	+CIPSTAMAC:<mac> OK
Parameters	<mac> string, MAC address of ESP8266 station
Command	**AT+CIPSTAMAC=<mac>**
Response	OK
Parameters	<mac> string, MAC address of ESP8266 station
Notes	Configuration changes will be stored in flash user parameter area.

21. AT+CIPSTAMAC_CUR – Set MAC address of station

MAC addresses of ESP8266 soft-AP and station are different, please do NOT set them to be the same. Note that the bit 0 of the first byte of ESP8266 MAC address cannot be 1, for example, MAC address can be "18:fe:35:98:d3:7b", but cannot be "15:fe:35:98:d3:7b".

AT+ CIPSTAMAC_CUR - Set MAC address of ESP8266 station, do not save to flash	
Example	AT+CIPSTAMAC_CUR="18:fe:35:98:d3:7b"
Command	**AT+CIPSTAMAC_CUR?**
Response	+CIPSTAMAC_CUR:<mac> OK
Parameters	<mac>　string, MAC address of ESP8266 station
Command	**AT+CIPSTAMAC_CUR=<mac>**
Response	OK
Parameters	<mac>　string, MAC address of ESP8266 station
Notes	This configuration will NOT be stored in flash.

22. AT+CIPSTAMAC_DEF – Set MAC address of station, save as default

MAC addresses of ESP8266 soft-AP and station are different, please do NOT set them to be the same. Note that the bit 0 of the first byte of ESP8266 MAC address cannot be 1, for example, MAC address can be "18:fe:35:98:d3:7b", but cannot be "15:fe:35:98:d3:7b".

AT+ CIPSTAMAC_DEF - Set MAC address of ESP8266 station, save to flash	
Example	AT+CIPSTAMAC_DEF="18:fe:35:98:d3:7b"
Command	**AT+CIPSTAMAC_DEF?**
Response	+CIPSTAMAC_DEF:<mac> OK
Parameters	<mac>　string, MAC address of ESP8266 station
Command	**AT+CIPSTAMAC_DEF=<mac>**
Response	OK
Parameters	<mac>　string, MAC address of ESP8266 station
Notes	Configuration changes will be stored in flash user parameter area.

23. AT+CIPAPMAC – Set MAC address of softAP

MAC addresses of ESP8266 soft-AP and station are different, please do NOT set them to be the same. Note that the bit 0 of the first byte of ESP8266 MAC address cannot be 1, for example, MAC address can be "1a:fe:36:97:d5:7b", but cannot be "15:fe:36:97:d5:7b".

AT+ CIPAPMAC – Set MAC address of ESP8266 softAP	
[@deprecated]. Use AT+CIPAPMAC_CUR or AT+CIPAPMAC_DEF instead.	
Example	AT+CIPAPMAC="1a:fe:36:97:d5:7b"
Command	**AT+CIPAPMAC?**
Response	+CIPAPMAC:<mac> OK
Parameters	<mac> string, MAC address of ESP8266 softAP
Command	**AT+CIPAPMAC=<mac>**
Response	OK
Parameters	<mac> string, MAC address of ESP8266 softAP
Notes	Configuration changes will be stored in flash user parameter area.

24. AT+CIPAPMAC_CUR – Set MAC address of softAP

MAC addresses of ESP8266 soft-AP and station are different, please do NOT set them to be the same. Note that the bit 0 of the first byte of ESP8266 MAC address cannot be 1, for example, MAC address can be "1a:fe:36:97:d5:7b", but cannot be "15:fe:36:97:d5:7b".

AT+CIPAPMAC_CUR – Set MAC addr of ESP8266 softAP, won't save to flash	
Example	AT+CIPAPMAC_CUR="1a:fe:36:97:d5:7b"
Command	**AT+CIPAPMAC_CUR?**
Response	+CIPAPMAC_CUR:<mac> OK
Parameters	<mac> string, MAC address of ESP8266 soft-AP
Command	**AT+CIPAPMAC_CUR=<mac>**
Response	OK
Parameters	<mac> string, MAC address of ESP8266 soft-AP
Notes	This configuration will NOT be stored in flash.

25. AT+CIPAPMAC_DEF – Set MAC address of softAP and save as default

MAC addresses of ESP8266 soft-AP and station are different, please do NOT set them to be the same. Note that the bit 0 of the first byte of ESP8266 MAC address cannot be 1, for example, MAC address can be "1a:fe:36:97:d5:7b", but cannot be "15:fe:36:97:d5:7b".

AT+ CIPAPMAC_DEF – Set MAC address of ESP8266 softAP, save to flash	
Example	AT+CIPAPMAC_DEF="1a:fe:36:97:d5:7b"
Command	**AT+CIPAPMAC_DEF?**
Response	+CIPAPMAC_DEF:\<mac\> OK
Parameters	\<mac\> string, MAC address of ESP8266 soft-AP
Command	**AT+CIPAPMAC_DEF=\<mac\>**
Response	OK
Parameters	\<mac\> string, MAC address of ESP8266 soft-AP
Notes	Configuration changes will be stored in flash user parameter area.

26. AT+CIPSTA – Set IP address of station

Only after ESP8266 station is connected to an AP, station IP can be obtained and inquired.
Configuration changes will be stored in flash user parameter area.

AT+ CIPSTA – Set IP address of ESP8266 station	
[@deprecated]. Please use AT+CIPSTA_CUR or AT+CIPSTA_DEF instead.	
Example	AT+CIPSTA="192.168.6.100", "192.168.6.1", "255.255.255.0"
Command	**AT+CIPSTA?**
Response	+CIPSTA:<IP> OK
Parameters	<IP> string, IP address of ESP8266 station
Command	**AT+CIPSTA=<IP>[, <gateway>, <netmask>]**
Response	OK
Parameters	<IP> string, IP address of ESP8266 station [<gateway>] gateway [<netmask>] netmask
Notes	This configuration interacts with AT+CWDHCP related AT commands: • If static IP is enabled, DHCP will be disabled; • If DHCP is enabled, static IP will be disabled; • This will depend on the last configuration.

27. AT+CIPSTA_CUR – Set IP address of station

Only after ESP8266 station is connected to an AP, station IP can be obtained and inquired. This configuration will NOT be stored in flash.

AT+CIPSTA_CUR – Set IP address of ESP8266 station, do not save to flash	
Example	AT+CIPSTA_CUR="192.168.6.100", "192.168.6.1", "255.255.255.0"
Command	**AT+CIPSTA_CUR?**
Response	+CIPSTA_CUR:\<IP\> OK
Parameters	\<IP\> string, IP address of ESP8266 station
Command	**AT+CIPSTA_CUR=\<IP\>[, \<gateway\>, \<netmask\>]**
Response	OK
Parameters	\<IP\> string, IP address of ESP8266 station [\<gateway\>] gateway [\<netmask\>] netmask
Notes	This configuration interacts with DHCP related AT commands (AT+CWDHCP related): • If static IP is enabled, DHCP will be disabled; • If DHCP is enabled, static IP will be disabled; • This will depend on the last configuration.

28. AT+CIPSTA_DEF – Set IP address of station and save as default

AT+CIPSTA_DEF – Set IP address of ESP8266 station, save to flash	
Example	AT+CIPSTA_DEF="192.168.6.100", "192.168.6.1", "255.255.255.0"
Command	**AT+CIPSTA_DEF?**
Response	+CIPSTA:\<IP\> OK
Parameters	\<IP\> string, IP address of ESP8266 station
Command	**AT+CIPSTA_DEF=\<IP\>[, \<gateway\>, \<netmask\>]**
Response	OK
Parameters	\<IP\> string, IP address of ESP8266 station [\<gateway\>] gateway [\<netmask\>] netmask
Notes	Configuration changes will be stored in flash user parameter area. This configuration interacts with DHCP related AT commands (AT+CWDHCP related): • If static IP is enabled, DHCP will be disabled; • If DHCP is enabled, static IP will be disabled; • This will depend on the last configuration.

29. AT+ CIPAP – Set IP address of softAP

ESP8266 only supports class C IP addresses. This configuration will be stored in flash user parameter area.

AT+ CIPAP – Set IP address of ESP8266 softAP	
[@deprecated]. Please use AT+CIPAP_CUR or AT+CIPAP_DEF instead.	
Example	AT+CIPAP="192.168.5.1", "192.168.5.1", "255.255.255.0"
Command	**AT+CIPAP?**
Response	+CIPAP:<IP> OK
Parameters	<IP> string, IP address of ESP8266 softAP
Command	**AT+CIPAP=<IP>[, <gateway>, <netmask>]**
Response	OK
Parameters	<IP> string, IP address of ESP8266 softAP [<gateway>] gateway [<netmask>] netmask
Notes	This configuration interacts with DHCP related AT commands (AT+CWDHCP related): • If static IP is enabled, DHCP will be disabled; • If DHCP is enabled, static IP will be disabled; • This will depend on the last configuration.

30. AT+CIPAP_CUR – Set IP address of softAP

ESP8266 only support class C IP address. This configuration will NOT store in flash.

AT+CIPAP_CUR – Set IP address of ESP8266 softAP, won't save to flash	
Example	AT+CIPAP_CUR="192.168.5.1", "192.168.5.1", "255.255.255.0"
Command	**AT+CIPAP_CUR?**
Response	+CIPAP_CUR:<IP>

OK |
Parameters	<IP> string, IP address of ESP8266 softAP
Command	**AT+CIPAP_CUR=<IP>[, <gateway>, <netmask>]**
Response	OK
Parameters	<IP> string, IP address of ESP8266 softAP [<gateway>] gateway [<netmask>] netmask
Notes	This configuration interacts with DHCP related AT commands (AT+CWDHCP related): • If enable static IP, DHCP will be disabled; • If enable DHCP, static IP will be disabled; • This will depend on the last configuration.

31. AT+CIPAP_DEF – Set IP address of softAP, save as default

ESP8266 only supports class C IP addresses.

AT+ CIPAP_DEF – Set IP address of ESP8266 softAP, save to flash	
Example	AT+CIPAP_DEF="192.168.5.1", "192.168.5.1", "255.255.255.0"
Command	AT+CIPAP_DEF?
Response	+CIPAP_DEF:<IP> OK
Parameters	<IP> string, IP address of ESP8266 softAP
Command	AT+CIPAP_DEF=<IP>[, <gateway>, <netmask>]
Response	OK
Parameters	<IP> string, IP address of ESP8266 softAP [<gateway>] gateway [<netmask>] netmask
Notes	Configuration changes will be stored in flash user parameter area. This configuration interacts with DHCP related AT commands (AT+CWDHCP related): • If static IP is enabled, DHCP will be disabled; • If DHCP is enabled, static IP will be disabled; • This will depend on the last configuration.

32. AT+CWSTARTSMART – Start SmartConfig

SmartConfig is only available in station mode. SmartConfig can get protocol type (AirKiss or ESP-TOUCH) automatically by command "AT+CWSTARTSMART". Alternatively, users may command "AT+CWSTARTSMART=<type>" to set a specific protocol type.

AT+CWSTARTSMART - Start SmartConfig	
Example	AT+CWMODE=1 AT+CWSTARTSMART
Command	**AT+CWSTARTSMART**
Response	OK
Parameters	none
Notes	The type of SmartConfig will be ESP-Touch + AirKiss, if the command is "AT+CWSTARTSMART"
Command	**AT+CWSTARTSMART=<type>**
Response	OK

Parameters	<type>
	    1 : ESP-Touch
	    2 : AirKiss
	    3 : ESP-Touch + AirKiss

Notes	• Message "Smart get WiFi info" means Smart Config acquired AP information successfully, then ESP8266 will try to connect to target AP. "Smartconfig connected WiFi" is printed if connection is successful. • ESP8266 cannot do anything during SmartConfig. Please wait for the message "Smartconfig connected WiFi" and then use command "AT+CWSTOPSMART" to stop SmartConfig.

33. AT+CWSTOPSMART – stop SmartConfig

AT+CWSTOPSMART - stop SmartConfig	
Command	**AT+CWSTOPSMART**
Response	OK
Notes	Irrespective of whether SmartConfig succeeded or not, before any other AT commands please always call "AT+CWSTOPSMART" to release the internal memory taken up by SmartConfig.

34. AT+CWSTARTDISCOVER – Start the mode that ESP8266 can be found by WeChat

The parameter of this command needs to be obtained from Espressif Cloud. After connecting to an AP and obtaining an IP address, ESP8266 can be found by WeChat which is on the same LAN.
WeChat : http://iot.weixin.qq.com

AT+CWSTARTDISCOVER	
– Start the mode that ESP8266 can be found by WeChat which is on the same LAN	
Example	AT+CWSTARTDISCOVER="gh_9e2cff3dfa51", "122475", 10
Command	AT+CWSTARTDISCOVER=<WeChat number>, <dev_type>, <time>
Response	OK

Parameters	<WeChat number> got from WeChat
	<dev_type> got from WeChat
	<time> time intervals after which ESP8266 sends packet, range: 0 ~ 24x3600, unit : second.
	    0 : ESP8266 will not send packet, waiting for WeChat's detection.
	    Otherwise, it is the time interval that ESP8266 sends packet, so the WeChat may find it easily.

Notes	ESP8266 station needs to connect to an AP and obtain an IP address first. This command should be used after that so WeChat can find this ESP8266 device.

35. AT+CWSTOPDISCOVER – Stop the mode that ESP8266 can be found by WeChat

AT+CWSTOPDISCOVER	
– Stop the mode that ESP8266 can be found by WeChat which is on the same LAN	
Command	AT+CWSTOPDISCOVER
Response	OK

36. AT+WPS – Set WPS function

Noe that WPS function can only be used when ESP8266 station is enabled. WPS function does not support WEP encryption.

AT+WPS – Set WPS function	
Example	AT+CWMODE=1 AT+WPS=1
Command	**AT+WPS=<enable>**
Response	OK
Parameter	<enable> 　1 : start WPS function 　0 : stop WPS function

37. AT+MDNS – Set MDNS function

ESP8266 softAP mode cannot support MDNS. Please do not use special characters (for example, "." character), or use a protocol name (for example, "http"), when defining "host_name" and "server_name" for MDNS.

AT+MDNS – Set MDNS function	
Example	AT+MDNS=1, "espressif", "iot", 8080
Command	**AT+MDNS=<enable>, <hostname>, <server_name>, <server_port>**
Response	OK
Parameter	<enable> 　1 : enable MDNS function 　0 : disable MDNS function <hostname>　MDNS host name <server_name> MDNS server name <server_port>　MDNS server port

5.　　TCP/IP Related AT Commands

5.1.　　Overview

TCP/IP	
Command	**Description**
AT+ CIPSTATUS	Get connection status
AT+ CIPDOMAIN	DNS function
AT+CIPSTART	Establish TCP connection, UDP transmission or SSL connection
AT+CIPSSLSIZE	Set the size of SSL buffer
AT+CIPSEND	Send data
AT+CIPSENDEX	Send data, if <length> or "\0" is met, data will be sent
AT+CIPSENDBUF	Write data into TCP-send-buffer
AT+CIPBUFRESET	Reset segment ID count
AT+CIPBUFSTATUS	Check status of TCP-send-buffer
AT+CIPCHECKSEQ	Check if a specific segment is sent or not
AT+CIPCLOSE	Close TCP/UDP/SSL connection
AT+CIFSR	Get local IP address
AT+CIPMUX	Set multiple connections mode
AT+CIPSERVER	Configure as server
AT+CIPMODE	Set transmission mode
AT+SAVETRANSLINK	Save transparent transmission link to flash
AT+CIPSTO	Set timeout when ESP8266 runs as TCP server
AT+CIUPDATE	Upgrade firmware through network
AT+PING	Function PING
AT+CIPDINFO	Show remote IP and remote port with "+IPD"

5.2. TCP/IP

1. AT+CIPSTATUS – Check network connection status

AT+CIPSTATUS – Check network connection status	
Command	**AT+CIPSTATUS**
Response	STATUS:<stat> +CIPSTATUS:<link ID>, <type>, <remote_IP>, <remote_port>, <local_port>, <tetype>

Parameters	<stat> status of ESP8266 station interface
	    2 : ESP8266 station connected to an AP and has obtained IP
	    3 : ESP8266 station created a TCP or UDP transmission
	    4 : the TCP or UDP transmission of ESP8266 station disconnected
	    5 : ESP8266 station did NOT connect to an AP
	<link ID> ID of the connection (0~4), for multi-connect
	<type> string, "TCP" or "UDP"
	<remote_IP> string, remote IP address
	<remote_port> remote port number
	<local_port> ESP8266 local port number
	<tetype>
	    0 : ESP8266 runs as client
	    1 : ESP8266 runs as server

2. AT+CIPDOMAIN – DNS function

ESP8266 needs to connect to a router to access to the internet first, then it is able to call the DNS function.

AT+CIPDOMAIN – DNS function	
Example	AT+CWMODE=1 // Set station mode AT+CWJAP="SSID","password" // access to the internet AT+CIPDOMAIN="iot.espressif.cn" // DNS function
Command	**AT+CIPDOMAIN=<domain name>**
Response	+CIPDOMAIN:<IP address>
Parameters	<domain name> the domain name

3.   AT+CIPSTART – Establish TCP connection, UDP transmission or SSL connection

Refer to documentation "4B-ESP8266__AT Command Examples" for more on how to use this command.

AT+CIPSTART – Function 1: Establish TCP connection	
Example	AT+CIPSTART="TCP", "iot.espressif.cn", 8000 AT+CIPSTART="TCP", "192.168.101.110", 1000
Single connection (AT+CIPMUX=0)	**AT+CIPSTART=** **<type>, <remote IP>, <remote port>[, <TCP keep alive>]**
Multiple connection (AT+CIPMUX=1)	**AT+CIPSTART=<link ID>,** **<type>, <remote IP>, <remote port>[, <TCP keep alive>]**
Response	OK or ERROR If TCP is already connected, returns ALREADY CONNECT

Parameters	<link ID> ID of network connection (0~4), used for multi-connection
	 <type> string, "TCP" or "UDP"
	 <remote IP> string, remote IP address
	 <remote port> string, remote port number
	 [<TCP keep alive>] optional, detection time interval when TCP is kept alive, this function is closed by default.
	    0 : disable TCP keep-alive
	    1 ~ 7200 : detection time interval, unit: second

AT+CIPSTART – Function 2: Register UDP port	
Example	AT+CIPSTART="UDP", "192.168.101.110", 1000, 1002, 2
Single connection (AT+CIPMUX=0)	**AT+CIPSTART=<type>, <remote IP>, <remote port>[, <UDP local port>, <UDP mode>]**
Multiple connection (AT+CIPMUX=1)	**AT+CIPSTART=<link ID>, <type>, <remote IP>, <remote port>[, <UDP local port>, <UDP mode>]**
Response	OK or ERROR If connection already exists, returns ALREADY CONNECT

Parameters	<link ID> ID of network connection (0~4), used for multi-connection
	<type> string, "TCP" or "UDP"
	<remote IP> string, remote IP
	<remote port> string, remote port
	[<UDP local port>] optional, UDP port of ESP8266
	[<UDP mode>] optional. In UDP transparent transmission, it has to be 0.
	    0 : destination peer entity of UDP will not change, default option.
	    1 : destination peer entity of UDP can change once.
	    2 : destination peer entity of UDP is allowed to change.
	
	Note: [<UDP mode>] can only be used when [<UDP local port>] is set.

AT+CIPSTART – Function 3: Establish SSL connection

Example	AT+CIPSSLSIZE=4096 AT+CIPSTART="SSL", "iot.espressif.cn", 8443
Single connection (AT+CIPMUX=0)	**AT+CIPSTART=** **<type>, <remote IP>, <remote port>[, <TCP keep alive>]**
Multiple connection (AT+CIPMUX=1)	**AT+CIPSTART=<link ID>,** **<type>, <remote IP>, <remote port>[, <TCP keep alive>]**
Response	OK or ERROR If TCP is already connected, returns ALREADY CONNECT

Parameters	<link ID> ID of network connection (0~4), used for multi-connection
	<type> string, "SSL"
	<remote IP> string, remote IP address
	<remote port> string, remote port number
	[<TCP keep alive>] optional, detection time interval when TCP is kept alive, this function is closed by default.
	    0 : disable TCP keep-alive
	    1 ~ 7200 : detection time interval, unit: second

Note	1. ESP8266 can only set one SSL connection at most.
	2. SSL connection does not support UART-WiFi passthrough mode (transparent transmission).
	3. SSL connection needs a lot of memory, otherwise, it may cause system reboot. Users can try command "**AT+CIPSSLSIZE=<size>**" to enlarge the buffer size.

4. AT+CIPSSLSIZE – Set the size of SSL buffer

AT+CIPSSLSIZE – Set the size of SSL buffer	
Command	**AT+CIPSSLSIZE=<size>**
Response	OK
Parameters	<size> the size of SSL buffer, range: [2048, 4096]

5. AT+CIPSEND – Send data

Please refer to documentation "4B-ESP8266__AT Command Examples" for more examples.

AT+CIPSEND – Send data	
Single connection	(+CIPMUX=0) **AT+CIPSEND=\<length\>**
Multiple connection	(+CIPMUX=1) **AT+CIPSEND=\<link ID\>, \<length\>**
UDP Transmission	**AT+CIPSEND=[\<link ID\>,]\<length\>[, \<remote IP\>, \<remote port\>]**
Response	Wrap return ">" after set command. Begins receiving serial data, when data length is met, starts transmission of data. If connection cannot be established or gets disconnected during data transfer, returns ERROR If data is transmitted successfully, returns SEND OK
Parameters	\<link ID\> ID of the connection (0~4), for multi-connect \<length\> data length, MAX 2048 bytes [\<remote IP\>] optional, UDP transmission can set remote IP when sending data [\<remote port\>] optional, UDP transmission can set remote port when sending data
Command	**AT+CIPSEND**
Response	Wrap return ">" after execute command. Enters transparent transmission, 20ms interval between each packet, maximum 2048 bytes per packet. When single packet containing "+++" is received, it returns to normal command mode. Please wait at least 1 second before sending next AT command. This command can only be used in transparent transmission mode which requires single connection. For UDP transparent transmission, \<UDP mode\> has to be 0 in command "AT+CIPSTART"

6.   AT+CIPSENDEX – Send data

AT+CIPSENDEX – Send data	
Single connection	(+CIPMUX=0) **AT+CIPSENDEX=\<length\>**
Multiple connection	(+CIPMUX=1) **AT+CIPSENDEX=\<link ID\>, \<length\>**
UDP Transmission	**AT+CIPSENDEX=[\<link ID\>,]\<length\>[, \<remote IP\>, \<remote port\>]**
Response	Wrap return ">" after set command. Begins receiving serial data, when data length or "\0" is met, starts transmission of data. Therefore, for sending "\0", please send it as "\\0" If connection cannot be established or gets disconnected during transmission, returns ERROR If data is transmitted successfully, returns SEND OK
Parameters	\<link ID\> ID of the connection (0~4), for multi-connect \<length\> data length, MAX 2048 bytes

7.　AT+CIPSENDBUF – Write data into TCP-send-buffer

This command only writes data into 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 , SEND OK. Before data <length> is met, input "+++" can switch back from data mode to command mode, and discard the data received before, cancel the "AT+CIPSENDBUF". This command can NOT be used on SSL connections.

AT+CIPSENDBUF – Write data into TCP-send-buffer	
Single connection	(+CIPMUX=0) **AT+CIPSENDBUF=<length>**
Multiple connection	(+CIPMUX=1) **AT+CIPSENDBUF=<link ID>, <length>**
Response	<current segment ID>, OK > Wrap return ">" begins receiving serial data, when data <length> is met, sends it; data more than <length> will be discarded, and returns "busy" If connection cannot be established, or if it is not a TCP connection, or if buffer is full, or some other error occurred, returns ERROR If data is transmitted successfully, (1) for single connection, returns , SEND OK (2) for multiple connection, returns <link ID>, , SEND OK
Parameters	<link ID> ID of the connection (0~4), for multi-connect uint32, starts from 1, add 1 every time be called; <length> data length, data more than <length> will be discarded, MAX 2048 bytes

8.   AT+CIPBUFSTATUS – Check status of TCP-send-buffer

Please do not use this command on SSL connections.

AT+CIPBUFSTATUS – Check status of TCP-send-buffer	
Example	Single connection: AT+CIPBUFSTATUS returns 20, 15, 10, 200, 7 20 : means the latest segment ID is 19, next time AT+CIPSENDBUF is called, the segment ID returned will be 20; 15: means that the TCP segment having ID 15 is the latest segment that was sent(may not succeed); 10: means TCP segment having ID 10 was sent successfully; 200: TCP-send-buffer has 200 bytes available; 7: available TCP queue number, it is not reliable; when queue number is 0, no more TCP data can be sent.
Single connection	(+CIPMUX=0) **AT+CIPBUFSTATUS**
Multiple connection	(+CIPMUX=1) **AT+CIPBUFSTATUS=<link ID>**
Response	<next segment ID>, , , <remain buffer size>, <queue number> OK If connection is not established, returns ERROR
Parameters	<link ID> ID of the connection (0~4), for multi-connect <next segment ID> next segment ID will be obtained by AT+CIPSENDBUF; the latest segment that was sent (may not succeed); the latest segment that was sent successfully; <remain buffer size> TCP-send-buffer remaining buffer size; <queue number> available TCP queue number, it's not reliable;when queue number is 0, no more TCP data can be sent.

9. AT+CIPCHECKSEQ – Check if a specific segment was sent successfully

AT+CIPCHECKSEQ – Check if a specific segment was sent successfully	
Single connection	(+CIPMUX=0) **AT+CIPCHECKSEQ=**
Multiple connection	(+CIPMUX=1) **AT+CIPCHECKSEQ=<link ID>, **
Response	[<link ID>,] , <status> OK If connection is not established, returns ERROR
Parameters	<link ID> ID of the connection (0~4), for multi-connect segment ID got by AT+CIPSENDBUF; <status> TRUE, sent successfully; FALSE, failed to send
Note	1. This command can only be used if AT+CIPSENDBUF is used. 2. Only keeps record of the status of the latest 32 segments at most.

10. AT+CIPBUFRESET – Reset segment ID count

AT+CIPBUFRESET – Reset segment ID count	
Single connection	(+CIPMUX=0) **AT+CIPBUFRESET**
Multiple connection	(+CIPMUX=1) **AT+CIPBUFRESET=<link ID>**
Response	OK If connection is not established or there is still TCP data waiting to be sent, returns ERROR
Parameters	<link ID> ID of the connection (0~4), for multi-connect
Note	This command can only be used if AT+CIPSENDBUF is used.

11. AT+CIPCLOSE – Close TCP, UDP or SSL connection

AT+CIPCLOSE – Close TCP, UDP or SSL connection	
Multiple connection	AT+CIPCLOSE=<link ID>
Response	OK or If no such connection, returns ERROR
Parameters	<link ID> ID number of connection to close, when ID=5, all connections will be closed. (ID=5 has no effect in server mode)
Single connection	AT+CIPCLOSE
Response	OK or If no such connection is active, returns ERROR

12. AT+CIFSR – Get local IP address

AT+CIFSR – Get local IP address	
Command	AT+ CIFSR
Response	+ CIFSR:<IP address> OK
Parameters	<IP address> IP address of ESP8266 softAP IP address of ESP8266 station
Note	Only after ESP8266 station is connected to an AP, station IP can be obtained and inquired.

13. AT+CIPMUX – Enable multiple connections

AT+ CIPMUX – Enable or disable multiple connections	
Example	AT+CIPMUX=1
Command	**AT+CIPMUX?**
Response	+CIPMUX:<mode> OK
Parameters	<mode> 0 : single connection 1 : multiple connection
Command	**AT+CIPMUX=<mode>**
Response	OK If already connected, returns Link is builded
Parameters	The same as above.
Notes	1. "AT+CIPMUX=1" can only be set when transparent transmission disabled ("AT+CIPMODE=0") 2. This mode can only be changed after all connections are disconnected. 3. If TCP server is running, it must be deleted before single connection mode can be activated.

14. AT+CIPSERVER – Configure as TCP server

Server monitor will automatically be created when Server is created. When a client is connected to the server, it will take up one connection and be assigned an ID.

AT+CIPSERVER – Configure as TCP server	
Example	AT+CIPMUX=1 AT+CIPSERVER=1,1001
Command	**AT+CIPSERVER=<mode>[, <port>]**
Response	OK
Parameters	<mode> 　0 : Delete server 　1 : Create server <port> port number, default is 333
Notes	Server can only be created when AT+CIPMUX=1

15. AT+CIPMODE – Set transfer mode

UART-WiFi passthrough mode (transparent transmission) can only be enabled in TCP single connection mode or UDP of which remote IP and port won't change (parameter <UDP mode> is 0 when using command "AT+CIPSTART" to create a UDP transmission) .

During UART-WiFi passthrough transmission, if it is TCP connection and the TCP connection breaks, ESP8266 will keep trying to reconnect until "+++" is input to quit from transmission. After sending the "+++" token, please wait for at least 1 second before sending the next AT command.

If it is a normal TCP transmission and TCP connection breaks, ESP8266 will prompt " [<link ID>,] CLOSED" , and will not attempt to reconnect. Users can call "AT+CIPSTART" to create a connection again if it is needed.

AT+ CIPMODE – Set transfer mode	
Example	AT+CIPMODE=1
Command	**AT+CIPMODE?**
Response	+CIPMODE:<mode> OK
Parameters	same as below
Command	**AT+CIPMODE=<mode>**
Response	OK If already connected, returns Link is builded

Parameters	<mode>
	    0 : normal mode
	    1 : UART-WiFi passthrough mode

Notes	Configuration changes will NOT be stored in flash.

16. AT+SAVETRANSLINK – Save transparent transmission link to flash

AT+SAVETRANSLINK – Save transparent transmission link to flash	
Example	AT+SAVETRANSLINK=1, "192.168.6.110", 1002, "TCP"
Command	For TCP connection: **AT+SAVETRANSLINK =<mode>, <remote IP or domain name>, <remote port>[, <type>][, <TCP keep alive>]** For UDP transmission: **AT+SAVETRANSLINK =<mode>, <remote IP>, <remote port>[, <type>][, <UDP local port>]**
Response	OK

Parameters	<mode>
	   0 : normal mode, will NOT enter UART-WiFi passthrough mode on power-up
	   1 : save UART-WiFi passthrough mode
	<remote IP> remote IP or domain name
	<remote port> remote port
	[<type>] optional, TCP or UDP, default to be "TCP"
	[<TCP keep alive>] optional, TCP keep alive, default to be disabled
	   0: disable TCP keep alive
	   1 ~ 7200: keep-alive detect time interval, unit: second
	[<UDP local port>] optional, local port if UDP transparent transmission is to be started on power-up.

Notes	• This command will save the UART-WiFi passthrough mode and its link into Flash user parameter area, ESP8266 will enter UART-WiFi passthrough mode on any subsequent power cycles. • As long as the IP (or domain name), port conform to the specification, they will be saved to flash

17. AT+CIPSTO – Set TCP server timeout

AT+ CIPSTO – Set TCP server timeout	
Example	AT+CIPMUX=1 AT+CIPSERVER=1, 1001 AT+CIPSTO=10
Command	**AT+CIPSTO?**
Response	+ CIPSTO:<time> OK
Parameters	The same as below.
Command	**AT+CIPSTO=<time>**
Response	OK
Parameters	<time> TCP server timeout, range 0~7200 seconds
Notes	ESP8266 configured as TCP server will disconnect to TCP client that did not communicate with it until timeout. If AT+CIPSTO=0, the connection will never time out. It is not recommended.

18. AT+PING – Function Ping

AT+PING – Function Ping	
Example	AT+PING="192.168.1.1" AT+PING="www.baidu.com"
Command	**AT+PING=<IP>**
Response	+<time> OK Or ERROR // wrong parameter, or ping fail
Parameters	<IP> : string, host IP or domain name <time> : response time of ping

19. AT+CIUPDATE – Update through network

Firmware upgrade depends on network condition. It will return ERROR if upgrade failed due to unfavorable network conditions. Please wait for some time before retrying.

- If using Espressif AT BIN (\ESP8266_NONOS_SDK\bin\at), "AT+CIUPDATE" will download new AT BIN from Espressif Cloud.
- If using user-compiled AT BIN, users need to make their own "AT+CIUPDATE" to upgrade, Espressif provide a demo in \ESP8266_NONOS_SDK\example\at. AT BINs on server have to be named as "user1.bin" and "user2.bin".
- It is suggested that users should call "AT+RESTORE" to restore the factory default settings after upgrading the AT firmware.

AT+ CIUPDATE – update through network	
Command	AT+CIUPDATE
Response	+CIUPDATE:\<n> OK

Parameters	\<n>
	  1 : found server
	  2 : connected to server
	  3 : got edition
	  4 : starting update

20. AT+CIPDINFO – Show remote IP and port with "+IPD"

AT+CIPDINFO – Show remote IP and port with "+IPD" (received data from network)	
Example	AT+CIPDINFO=1
Command	AT+CIPDINFO=\<mode>
Response	OK

Parameters	\<mode>
	  0 : will not show remote IP and port with "+IPD"
	  1 : show remote IP and port with "+IPD"

21. +IPD – Receive network data

+IPD – Receive network data	
Single connection	(+CIPMUX=0) +IPD, <len>[, <remote IP>, <remote port>]:<data>
Multiple connection	(+CIPMUX=1) +IPD, <ID>, <len>[, <remote IP>, <remote port>]:<data>
Parameters	<remote IP> remote IP, enabled by command "AT+CIPDINFO=1" <remote port> remote port, enabled by command "AT+CIPDINFO=1" <ID> ID number of connection <len> data length <data> data received
Notes	When the module receives network data, it will send the data through the serial port using +IPD command.

6. Appendix

ESP8266 AT commands below will save configuration parameters into flash:

| AT Command	Example
Save into flash user parameter area	
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 =1, "192.168.6.10", 1001
Save into flash system parameter area	
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

- Only if the configuration changes, the AT firmware will write the new configuration into 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.
- To 512KB+512KB Flash Map:

    user parameter area is 0x7C000 ~ 0x80000, 16KB;
- To 1024KB+1024KB Flash Map:

    user parameter area is 0xFC000 ~ 0x100000, 16KB;
- System parameter area is always the last 16KB of Flash.

7. Q&A

If you have any questions about AT Commands, please contact us (feedback@espressif.com) with information as follows:

- **Version info of AT** : Using "AT+GMR" to get the version info.

    Hardware Module info: example Ai-thinker ESP-01

- **Screenshot or steps of the test steps,** for example:
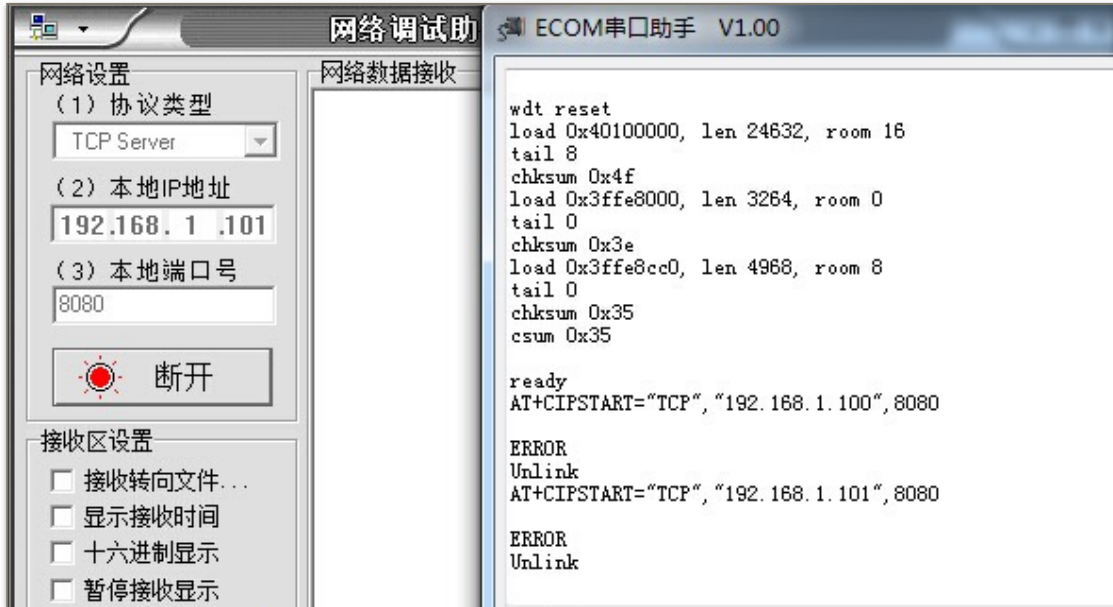
- **Log**:

    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)