Espressif ESP8266EX: AT Instruction Set

Status	Released
Current version	V0.22
Author	CG Xu
Completion Date	2015.03.14
Reviewer	Fei Yu
Completion Date	2015.03.14

[√] CONFIDENTIAL
[ ] INTERNAL
[ ] PUBLIC

Version Info

Date	Version	Author	Comments/Changes
2014.9.25	0.18	XuJingjie	1、Added upgrade through network 2、Added CWLAP
2014.11.10	0.19	XuJingjie	Added UDP
2014.11.27	0.20	XuJingjie	1、Added set and get APIP/APMAC/STAIP /STAMAC 2、Added start and stop DHCP
2015.01.23	0.21	CG Xu	1、Added factory reset 2、Added set UART configuration 3、Added set auto-connection 4、Added function ping
2015.03.14	0.22	CG Xu	1、XXX_CUR: set config won't save to flash；XXX_DEF：Set config and save to flash. 2、Add smart config 3、Add command to save transparent transmission link.

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, NONINFRINGEMENT, 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 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.

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

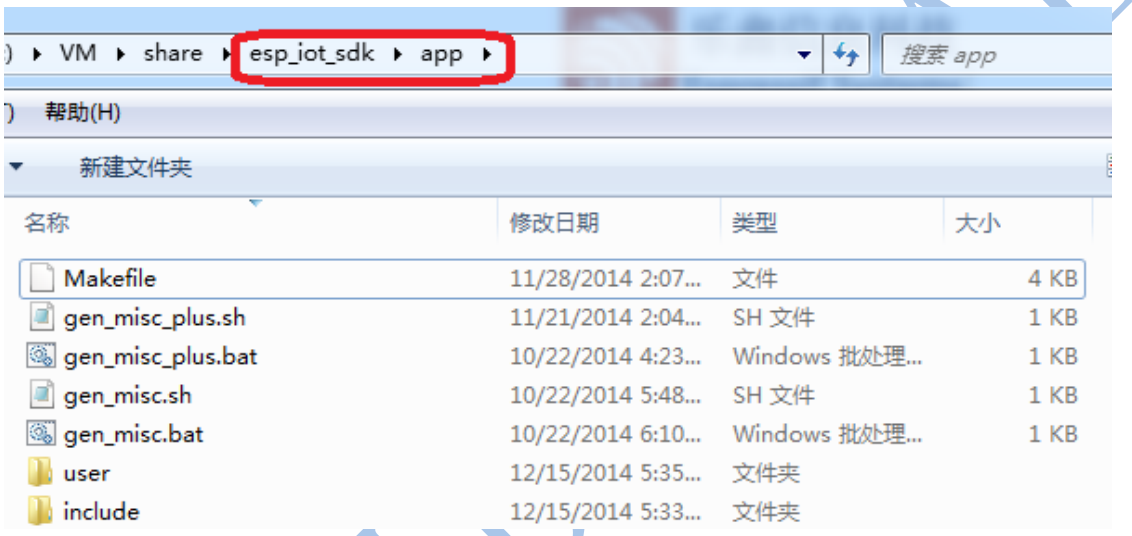
Table of Contents

1 Overview

This is the documentation for Espressif AT command Command set and usage.

Command set is divided into: Basic AT commands, Wifi function, AT commands, TCP / IP Toolbox AT commands.

Copy all files in folder “at” to folder “app” in esp_iot_sdk to compile.



Download:

boot.bin, downloads to flash 0x00000

user1.bin, downloads to flash 0x01000

blank.bin, downloads to flash both 0x3E000 and 0x7E000 to factory initialize

Note: Please make sure that correct BIN(\esp_iot_sdk\bin\at) is already in the chip (ESP8266) before the AT commands listed in this documentation can be used.

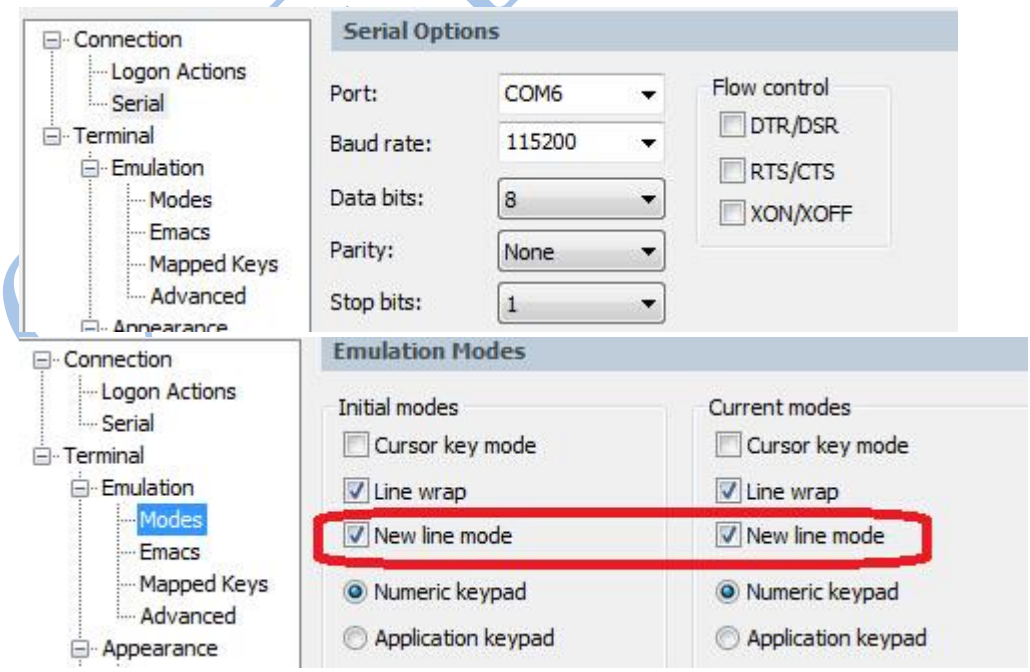
2 Command Description

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.

Note:

1. Not all AT Command has four commands.
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. Baud rate = 115200
5. AT Command ends with "\r\n"



3 Basic AT Command Set

3.1 Overview

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

3.2 Commands

3.2.1 AT – Test AT startup

AT – Test AT startup	
Type：execute Command： **AT**	Response： OK
	Param description：null

3.2.2 AT+RST – Restart module

AT+RST – Restart module	
Type ： execute Command： **AT+RST**	Response： OK
	Param description：null

3.2.3 AT+GMR – View version info

AT+GMR – View version info	
Type ： execute	Response：

Command： **AT+GMR**	<number> OK
	Param description： < number > version info，length：8 bytes
Note	For example，response is 0017xxxxxx，then 0017 means the AT version.

3.2.4 AT+GSLP – Enter deep-sleep mode

AT+GSLP – Enter deep-sleep mode	
Type ： set Command： **AT+GSLP=<time>**	Response： <time> OK
	Param description： < time > ms , set the sleep time of ESP8266 in ms. ESP8266 will wake up after X ms in deep-sleep.
Note	Hardware has to support deep-sleep wake up (XPD_DCDC connects to EXT_RSTB with 0R).

3.2.5 ATE – AT commands echo

ATE – AT commands echo	
Type ： execute Command： **ATE**	Response： OK
	Param description： ATE0 ： Disable echo ATE1 ： Enable echo

3.2.6 AT+RESTORE – Factory reset

AT+RESTORE – Factory reset	
Type ： execute Command： **AT+RESTORE**	Response： OK
Note	Reset configuration to default factory settings The chip will restart.

3.2.7 AT+UART – UART configuration

AT+UART – UART configuration	
This API is deprecated, please use AT+UART_CUR or AT+UART_DEF instead.	
Type ： set Command： AT+UART=<baudrate>,<databits>,<stopbits>, <parity>,<flow control>	Response： OK
	Param description： <baudrate> UART baudrate <databits> data bits 5：5 bits data 6：6 bits data 7：7 bits data 8：8 bits data <stopbits> stop bits 1：1 bit stop bit 2：1.5 bit stop bit 3：2 bit stop bit <parity> parity 0：None 1：Odd 2：EVEN <flow control> flow control 0：disable flow control 1：enable RTS 2：enable CTS 3：enable both RTS and CTS
Note	1. This configuration will store in Flash user parameter area. 2. To enable flow control hardware need to support it too. MTCK is UART0 CTS , MTDO is UART0 RTS 3. Baudrate range: 110~115200*40
Example	AT+UART=115200,8,1,0,3

3.2.8 AT+UART_CUR – current UART configuration

AT+UART_CUR – UART current configuration, won't save to Flash	
Type ： set Command： AT+UART_CUR=<baudrate>,<databits>,<stopbits>,	Response： OK
	Param description： <baudrate> UART baudrate

<parity>,<flow control>	<databits> data bits 5：5 bits data 6：6 bits data 7：7 bits data 8：8 bits data <stopbits> stop bits 1：1 bit stop bit 2：1.5 bit stop bit 3：2 bit stop bit <parity> parity 0：None 1：Odd 2：EVEN <flow control> flow control 0：disable flow control 1：enable RTS 2：enable CTS 3：enable both RTS and CTS
Note	1. This configuration will not store in Flash. 2. To enable flow control hardware need to support it too. MTCK is UART0 CTS , MTDO is UART0 RTS 3. Baudrate range: 110~115200*40
Example	AT+UART_CUR=115200,8,1,0,3

3.2.9 AT+UART_DEF – default UART configuration

AT+UART_DEF – set UART configuration, and save to flash as default value.	
Type ： set Command： AT+UART_DEF=<baudrate>,<databits>,<stopbits>, <parity>,<flow control>	Response： OK
	Param description： <baudrate> UART baudrate <databits> data bits 5：5 bits data 6：6 bits data 7：7 bits data 8：8 bits data <stopbits> stop bits 1：1 bit stop bit 2：1.5 bit stop bit 3：2 bit stop bit <parity> parity

	0：None 1：Odd 2：EVEN <flow control> flow control 0：disable flow control 1：enable RTS 2：enable CTS 3：enable both RTS and CTS
Note	1. This configuration will store in Flash user parameter area. 2. To enable flow control hardware need to support it too. MTCK is UART0 CTS , MTDO is UART0 RTS 3. Baudrate range: 110~115200*40
Example	AT+UART_DEF=115200,8,1,0,3

4 WIFI functions

4.1 Overview

WIFI	
Command	**Description**
AT+CWMODE	WIFI mode（sta/AP/sta+AP）, @Deprecated
AT+CWMODE_CUR	WIFI mode（sta/AP/sta+AP） Won't save to Flash
AT+CWMODE_DEF	WIFI 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+CWLAP	Lists available APs
AT+CWQAP	Disconnect from AP
AT+CWSAP	Set configuration of ESP8266 softAP @Deprecated
AT+CWSAP_CUR	Set configuration of ESP8266 softAP Won't save to Flash.
AT+CWSAP_DEF	Set configuration of ESP8266 softAP Save to Flash.
AT+CWLIF	Get station's ip which is connected to ESP8266 softAP
AT+CWDHCP	Enable/Disable DHCP, @Deprecated
AT+CWDHCP	Enable/Disable DHCP, won't save to Flash
AT+CWDHCP	Enable/Disable DHCP, save to Flash
AT+CWAUTOCONN	Connect to AP automatically when power on
AT+CIPSTAMAC	Set mac address of ESP8266 station @Deprecated
AT+CIPSTAMAC_CUR	Set mac address of ESP8266 station Won't save to Flash.
AT+CIPSTAMAC_DEF	Set mac address of ESP8266 station Save to Flash.
AT+CIPAPMAC	Set mac address of ESP8266 softAP @Deprecated

AT+CIPAPMAC_CUR	Set mac address of ESP8266 softAP Won't save to Flash.
AT+CIPAPMAC_DEF	Set mac address of ESP8266 softAP Save to Flash.
AT+CIPSTA	Set ip address of ESP8266 station, @Deprecated
AT+CIPSTA_CUR	Set ip address of ESP8266 station Won't save to Flash.
AT+CIPSTA_DEF	Set ip address of ESP8266 station Save to Flash.
AT+CIPAP	Set ip address of ESP8266 softAP, @Deprecated
AT+CIPAP_CUR	Set ip address of ESP8266 softAP Won't save to Flash.
AT+CIPAP_DEF	Set ip address of ESP8266 softAP Save to Flash.

4.2 Commands

4.2.1 AT+CWMODE – WIFI mode

AT+CWMODE - WIFI mode（station/softAP/station+softAP）	
@Deprecated. Please use AT+CWMODE_CUR or AT+CWMODE_DEF instead.	
Type：test Function: Get value scope of wifi mode. Command： **AT+CWMODE=?**	Response： +CWMODE:(value scope of <mode>) OK
	Param description： <mode>1 means Station mode 2 means AP mode 3 means AP + Station mode
Type：query Function: Query ESP8266's current wifi mode. Command： **AT+CWMODE?**	Response： +CWMODE:<mode> OK
	Param description： The same as above.
Type：set Function: Set ESP8266 wifi mode Command：	Response： OK
	Param description：

AT+CWMODE=<mode>	The same as above.
Note	This configuration will store in Flash system parameter area.
Example	AT+CWMODE=3

4.2.2 AT+CWMODE_CUR – current WIFI mode

AT+CWMODE_CUR – Set WIFI mode（sta/AP/sta+AP）, won't save to Flash	
Type：test Function: Get value scope of wifi mode. Command： **AT+CWMODE_CUR=?**	Response： +CWMODE_CUR:(value scope of <mode>) OK
	Param description： <mode>1 means Station mode 2 means AP mode 3 means AP + Station mode
Type：query Function: Query ESP8266's current wifi mode. Command： **AT+CWMODE_CUR?**	Response： +CWMODE_CUR:<mode> OK
	Param description： The same as above.
Type：set Function: Set ESP8266 wifi mode Command： **AT+CWMODE_CUR= <mode>**	Response： OK
	Param description： The same as above.
Note	This configuration will not store in Flash.
Example	AT+CWMODE_CUR=3

4.2.3 AT+CWMODE_DEF – default WIFI mode

AT+CWMODE_DEF - WIFI mode（sta/AP/sta+AP）,save to Flash

Type：test Function: Get value scope of wifi mode. Command： **AT+CWMODE_DEF=?**	Response： +CWMODE_DEF:(value scope of <mode>) OK
	Param description： <mode>1 means Station mode 2 means AP mode 3 means AP + Station mode
Type：query Function: Query ESP8266's current wifi mode. Command： **AT+CWMODE_DEF?**	Response： +CWMODE_DEF:<mode> OK
	Param description： The same as above.
Type：set Function: Set ESP8266 wifi mode Command： **AT+CWMODE_DEF=** **<mode>**	Response： OK
	Param description： The same as above.
Note	This configuration will store in Flash system parameter area.
Example	AT+CWMODE_DEF=3

4.2.4 AT+CWJAP – Connect to AP

AT+CWJAP – Connect to AP	
@Deprecated. Please use AT+CWJAP_CUR or AT+CWJAP_DEF instead.	
Type：query Function: Query AP's info which is connect by ESP8266. Command： **AT+ CWJAP?**	Response： + CWJAP:<ssid> OK
	Param description： <ssid> string, AP's SSID
Type：set Function: Set AP's info which will be	Response： OK

connect by ESP8266. Command： **AT+ CWJAP =** **<ssid>,< pwd >**	ERROR
	Param description： <ssid> string, AP's SSID <pwd> string, MAX: 64 bytes ASCII This command needs station mode enable. Escape character syntax is needed if "SSID" or "password" contains any special characters (' ,' 、' "' and' \')
Note	This configuration will store in Flash system parameter area.
Example	AT+ CWJAP ="abc"，"0123456789" If SSID is "ab\,c" and password is "0123456789"\" AT+CWJAP ="ab\\\,c"，"0123456789\" \\"

4.2.5 AT+CWJAP_CUR – Connect to AP，for current

AT+CWJAP_CUR – Connect to AP, won't save to Flash	
Type：query Function: Query AP's info which is connect by ESP8266. Command： **AT+CWJAP_CUR?**	Response： + CWJAP_CUR:<ssid> OK
	Param description： <ssid> string, AP's SSID
Type：set Function: Set AP's info which will be connect by ESP8266. Command： **AT+CWJAP_CUR =** **<ssid>,< pwd >**	Response： OK ERROR
	Param description： <ssid> string, AP's SSID <pwd> string, MAX: 64 bytes ASCII This command needs station mode enable. Escape character syntax is needed if "SSID" or "password" contains any special characters (' ,' 、' "' and' \')
Note	This configuration will not store in Flash .
Example	AT+CWJAP_CUR ="abc","0123456789"

	If SSID is “ab\,c” and password is “0123456789”\” AT+CWJAP_CUR=“ab\\\,c”,“0123456789\”\\”

4.2.6 AT+CWJAP_DEF – Connect to AP，save as default

AT+CWJAP_DEF – Connect to AP	
Type：query Function: Query AP’s info which is connect by ESP8266. Command： **AT+CWJAP_DEF?**	Response： + CWJAP_DEF:<ssid> OK Param description： <ssid> string, AP’s SSID
Type：set Function: Set AP’s info which will be connect by ESP8266. Command： **AT+ CWJAP_DEF =** **<ssid>,< pwd >**	Response： OK ERROR Param description： <ssid> string, AP’s SSID <pwd> string, MAX: 64 bytes ASCII This command needs station mode enable. Escape character syntax is needed if "SSID" or "password" contains any special characters (’ ,’ 、’ “’ and’ \’)
Note	This configuration will store in Flash system parameter area.
Example	AT+CWJAP_DEF =“abc”,“0123456789” If SSID is “ab\,c” and password is “0123456789”\” AT+CWJAP_DEF=“ab\\\,c”,“0123456789\”\\”

4.2.7 AT+CWLAP – List available APs

AT+CWLAP - Lists available APs	
Type：set	Response：

Function: Search available APs with specific conditions. Command： **AT+ CWLAP =** **<ssid>,< mac >,<ch>**	+ CWLAP: <ecn>,<ssid>,<rssi>,<mac>,<ch> OK ERROR
	Param description： The same as below.
Type ： execute Function: Lists all available APs. Command： **AT+CWLAP**	Response： + CWLAP: <ecn>,<ssid>,<rssi>,<mac>,<ch> OK ERROR
	Param description： < ecn >0 OPEN 1 WEP 2 WPA_PSK 3 WPA2_PSK 4 WPA_WPA2_PSK <ssid> string, SSID of AP <rssi> signal strength <mac> string, MAC address
Example	AT+CWLAP="wifi","ca:d7:19:d8:a6:44",6 Or find AP with specific ssid： AT+CWLAP="wifi",""

4.2.8 AT+CWQAP – Disconnect from AP

AT+CWQAP - Disconnect from AP	
Type：test Function: Only for test Command： **AT+CWQAP=?**	Response： OK
	Param description：
Type ： execute Function: Disconnect from AP. Command： **AT+ CWQAP**	Response： OK
	Param description：

4.2.9 AT+CWSAP – Configuration of softAP mode

AT+ CWSAP – Configuration of softAP mode	
@Deprecated. Please use AT+CWSAP_CUR or AT+CWSAP_DEF instead.	
Type：Query Function: Query configuration of softAP mode. Command： **AT+ CWSAP?**	Response： + CWSAP:<ssid>,<pwd>,<chl>,<ecn>
	Param description： The same as below.
Type：Set Function: Set configuration of softAP mode. Command： **AT+ CWSAP=** **<ssid>,<pwd>,<chl>,** **<ecn>**	Response： OK ERROR
	Note: This CMD is only available when softAP mode enable, and need to follow by AT+RST to make it works. Param description： <ssid> string, ESP8266 softAP' SSID <pwd> string, MAX: 64 bytes ASCII <chl> channel id < ecn >0 OPEN 2 WPA_PSK 3 WPA2_PSK 4 WPA_WPA2_PSK
Note	This configuration will store in Flash system parameter area.
Example	AT+CWSAP="ESP8266","1234567890",5,3

4.2.10 AT+CWSAP_CUR – Current config of softAP mode

AT+CWSAP_CUR – Current configuration of softAP mode, won't save to Flash	
Type：Query Function: Query configuration of softAP mode. Command： **AT+CWSAP_CUR?**	Response： +CWSAP_CUR:<ssid>,<pwd>,<chl>,<ecn>
	Param description： The same as below.
Type：Set Function:	Response：

Set configuration of softAP mode. Command： **AT+CWSAP_CUR=** **<ssid>,<pwd>,<chl>,** **<ecn>**	OK ERROR
	Note: This CMD is only available when softAP mode enable, and need to follow by AT+RST to make it works. Param description： <ssid> string, ESP8266 softAP' SSID <pwd> string, MAX: 64 bytes ASCII <chl> channel id < ecn >0 OPEN 2 WPA_PSK 3 WPA2_PSK 4 WPA_WPA2_PSK
Note	This configuration will not store in Flash.
Example	AT+CWSAP_CUR="ESP8266","1234567890",5,3

4.2.11 AT+CWSAP_DEF – Default config of softAP mode

AT+ CWSAP_DEF – Default configuration of softAP mode, save to Flash	
Type：Query Function: Query configuration of softAP mode. Command： **AT+ CWSAP_DEF?**	Response： + CWSAP_DEF:<ssid>,<pwd>,<chl>,<ecn>
	Param description： The same as below.
Type：Set Function: Set configuration of softAP mode. Command： **AT+CWSAP_DEF=** **<ssid>,<pwd>,<chl>,** **<ecn>**	Response： OK ERROR
	Note: This CMD is only available when softAP mode enable, and need to follow by AT+RST to make it works. Param description： <ssid> string, ESP8266 softAP' SSID <pwd> string, MAX: 64 bytes ASCII <chl> channel id < ecn >0 OPEN 2 WPA_PSK 3 WPA2_PSK 4 WPA_WPA2_PSK
Note	This configuration will store in Flash system

	parameter area.
Example	AT+CWSAP_DEF="ESP8266","1234567890",5,3

4.2.12 AT+CWLIF – IP of stations

AT+ CWLIF – ip of stations which are connected to ESP8266 softAP	
Type : execute Function: Get ip of stations which are connected to ESP8266 softAP Command: **AT+CWLIF**	Response: <ip addr>,<mac> OK
	Param description: <ip addr> ip address of stations which are connected to ESP8266 softAP <mac> mac address of stations which are connected to ESP8266 softAP

4.2.13 AT+CWDHCP – Enable/Disable DHCP

AT+ CWDHCP – Enable/Disable DHCP	
@Deprecated. Please use AT+CWDHCP_CUR or AT+CWDHCP_DEF instead.	
Type: Query Command: **AT+CWDHCP?**	Response: DHCP disabled or enabled now?
	Description: Bit0: 0 - softap dhcp disable 1 - softap dhcp enable bit1 : 0 - station dhcp disable 1 - station dhcp enable
Type : set Function: Enable/Disable DHCP. Command: **AT+CWDHCP=<mode>,<en>**	Response: OK
	Param description: <mode> 0 : set ESP8266 softAP 1 : set ESP8266 station 2 : set both softAP and station <en> 0 : Enable DHCP 1 : Disable DHCP
Note	This configuration will store in Flash user

	parameter area.

4.2.14 AT+CWDHCP_CUR – Enable/Disable DHCP

AT+CWDHCP_CUR – Enable/Disable DHCP, won't save to Flash	
Type ： set Function: Enable/Disable DHCP. Command： **AT+CWDHCP_CUR=** **<mode>,<en>**	Response： OK
	Param description： <mode> 0 : set ESP8266 softAP 1 : set ESP8266 station 2 : set both softAP and station <en> 0 : Enable DHCP 1 : Disable DHCP
Note	This configuration will not store in Flash.
Example	AT+CWDHCP_CUR=0,1

4.2.15 AT+CWDHCP_DEF – Enable/Disable DHCP and save to Flash

AT+CWDHCP_DEF – Enable/Disable DHCP and save to Flash	
Type ： set Function: Enable/Disable DHCP. Command： **AT+CWDHCP_DEF=** **<mode>,<en>**	Response： OK
	Param description： <mode> 0 : set ESP8266 softAP 1 : set ESP8266 station 2 : set both softAP and station <en> 0 : Enable DHCP 1 : Disable DHCP
Note	This configuration will store in Flash user parameter area.
Example	AT+CWDHCP_CUR=0,1

4.2.16 AT+CWAUTOCONN – Auto connect to AP or not

AT+CWAUTOCONN – Connect to AP automatically or not	
Type ： set Function: Connect to AP automatically or not. Command： **AT+CWAUTOCONN=** **<enable>**	Response： OK
	Param description： <enable> 0 : do not auto-connect to AP when power on 1 : connect to AP automatically when power on Default is enable, ESP8266 station will connect to AP automatically when power on.
Note	This configuration will store in Flash system parameter area.
Example	AT+CWAUTOCONN=1

4.2.17 AT+CIPSTAMAC – Set mac address of station

AT+ CIPSTAMAC – Set mac address of ESP8266 station	
@Deprecated. Use AT+CIPSTAMAC_CUR or AT+CIPSTAMAC_DEF instead.	
Type ： query Function: Get mac address of ESP8266 station. Command： **AT+CIPSTAMAC?**	Response： +CIPSTAMAC:<mac> OK
	Param description： <mac> string, mac address of ESP8266 station
Type ： set Function: Set mac address of ESP8266 station. Command： **AT+CIPSTAMAC=<mac>**	Response： OK
	Param description： <mac> string, mac address of ESP8266 station
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPSTAMAC="18:fe:35:98:d3:7b"

4.2.18 AT+CIPSTAMAC_CUR – Set mac address of station

AT+ CIPSTAMAC – Set mac address of ESP8266 station, won't save to Flash

Type ： query Function: Get mac address of ESP8266 station. Command： **AT+CIPSTAMAC_CUR?**	Response： +CIPSTAMAC_CUR:<mac> OK
	Param description： <mac> string, mac address of ESP8266 station
Type ： set Function: Set mac address of ESP8266 station. Command： **AT+CIPSTAMAC_CUR=** **<mac>**	Response： OK
	Param description： <mac> string, mac address of ESP8266 station
Note	This configuration will not store in Flash.
Example	AT+CIPSTAMAC_CUR="18:fe:35:98:d3:7b"

4.2.19 AT+CIPSTAMAC_DEF – Set mac address of station, save as default

AT+ CIPSTAMAC_DEF – Set mac address of ESP8266 station, save to Flash	
Type ： query Function: Get mac address of ESP8266 station. Command： **AT+CIPSTAMAC_DEF?**	Response： +CIPSTAMAC_DEF:<mac> OK
	Param description： <mac> string, mac address of ESP8266 station
Type ： set Function: Set mac address of ESP8266 station. Command： **AT+CIPSTAMAC_DEF** **=<mac>**	Response： OK
	Param description： <mac> string, mac address of ESP8266 station
Note	This configuration will store in Flash user parameter area.

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

4.2.20 AT+CIPAPMAC – Set mac address of softAP

AT+ CIPAPMAC – Set mac address of ESP8266 softAP	
@Deprecated. Use AT+CIPAPMAC_CUR or AT+CIPAPMAC_DEF instead.	
Type ： query Function: Get mac address of ESP8266 softAP. Command： **AT+CIPAPMAC?**	Response： +CIPAPMAC:<mac> OK
	Param description： <mac> string, mac address of ESP8266 softAP
Type ： set Function: Set mac address of ESP8266 softAP. Command： **AT+CIPAPMAC=<mac>**	Response： OK
	Param description： <mac> string, mac address of ESP8266 softAP
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPAPMAC="1a:fe:36:97:d5:7b"

4.2.21 AT+CIPAPMAC_CUR – Set mac address of softAP

AT+CIPAPMAC_CUR – Set mac addr of ESP8266 softAP, won't save to Flash	
Type ： query Function: Get mac address of ESP8266 softAP. Command： **AT+CIPAPMAC_CUR?**	Response： +CIPAPMAC_CUR:<mac> OK
	Param description： <mac> string, mac address of ESP8266 softAP
Type ： set Function: Set mac address of ESP8266 softAP. Command： **AT+CIPAPMAC_CUR=**	Response： OK
	Param description： <mac> string, mac address of ESP8266 softAP

<mac>	
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPAPMAC_CUR="1a:fe:36:97:d5:7b"

4.2.22 AT+CIPAPMAC_DEF – Set mac address of softAP and save as default

AT+ CIPAPMAC_DEF – Set mac address of ESP8266 softAP, save to Flash	
Type ： query Function: Get mac address of ESP8266 softAP. Command： **AT+CIPAPMAC_DEF?**	Response： +CIPAPMAC_DEF:<mac> OK
	Param description： <mac> string, mac address of ESP8266 softAP
Type ： set Function: Set mac address of ESP8266 softAP. Command： **AT+CIPAPMAC_DEF =<mac>**	Response： OK
	Param description： <mac> string, mac address of ESP8266 softAP
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPAPMAC_DEF="1a:fe:36:97:d5:7b"

4.2.23 AT+CIPSTA – Set ip address of station

AT+ CIPSTA – Set ip address of ESP8266 station	
@Deprecated. Please use AT+CIPSTA_CUR or AT+CIPSTA_DEF instead.	
Type ： query Function: Get ip address of ESP8266 station. Command：	Response： +CIPSTA:<ip> OK
	Param description：

AT+CIPSTA?	<ip> string, ip address of ESP8266 station
Type ： set Function: Set ip address of ESP8266 station. Command： **AT+CIPSTA=<ip>** **[,<gateway>,<netmask>]**	Response： OK
	Param description： <ip> string, ip address of ESP8266 station [<gateway>] gateway [<netmask>] netmask
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPSTA="192.168.6.100","192.168.6.1", "255.255.255.0"

4.2.24 AT+CIPSTA_CUR – Set ip address of station

AT+CIPSTA_CUR – Set ip address of ESP8266 station, won't save to Flash	
Type ： query Function: Get ip address of ESP8266 station. Command： **AT+CIPSTA_CUR?**	Response： +CIPSTA_CUR:<ip> OK
	Param description： <ip> string, ip address of ESP8266 station
Type ： set Function: Set ip address of ESP8266 station. Command： **AT+CIPSTA_CUR** **=<ip>[,<gateway>,** **<netmask>]**	Response： OK
	Param description： <ip> string, ip address of ESP8266 station [<gateway>] gateway [<netmask>] netmask
Note	This configuration will store in Flash user parameter area.

Example	AT+CIPSTA_CUR="192.168.6.100","192.168.6.1","255.255.255.0"

4.2.25 AT+CIPSTA_DEF – Set ip address of station and save as default

AT+CIPSTA_DEF – Set ip address of ESP8266 station, save to Flash	
Type ： query Function: Get ip address of ESP8266 station. Command： **AT+CIPSTA_DEF?**	Response： +CIPSTA:<ip> OK
	Param description： <ip> string, ip address of ESP8266 station
Type ： set Function: Set ip address of ESP8266 station. Command： **AT+CIPSTA_DEF =<ip>[,<gateway>, <netmask>]**	Response： OK
	Param description： <ip> string, ip address of ESP8266 station [<gateway>] gateway [<netmask>] netmask
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPSTA_DEF="192.168.6.100","192.168.6.1","255.255.255.0"

4.2.26 AT+ CIPAP – Set ip address of softAP

AT+ CIPAP – Set ip address of ESP8266 softAP	
@Deprecated. Please use AT+CIPAP_CUR or AT+CIPAP_DEF instead.	
Type ： query Function: Get ip address of ESP8266 softAP.	Response： +CIPAP:<ip> OK

Command： **AT+CIPAP?**	Param description： <ip> string, ip address of ESP8266 softAP
Type ：set Function: Set ip address of ESP8266 softAP. Command： **AT+CIPAP=<ip>**	Response： OK
	Param description： <ip> string, ip address of ESP8266 softAP
Note	This configuration will not store in Flash user parameter area.
Example	AT+CIPAP="192.168.5.1"

4.2.27 AT+CIPAP_CUR – Set ip address of softAP

AT+CIPAP_CUR – Set ip address of ESP8266 softAP, won't save to Flash	
Type ：query Function: Get ip address of ESP8266 softAP. Command： **AT+CIPAP_CUR?**	Response： +CIPAP_CUR:<ip> OK
	Param description： <ip> string, ip address of ESP8266 softAP
Type ：set Function: Set ip address of ESP8266 softAP. Command： **AT+CIPAP_CUR =<ip>**	Response： OK
	Param description： <ip> string, ip address of ESP8266 softAP
Note	This configuration will not store in Flash.
Example	AT+CIPAP_CUR="192.168.5.1"

4.2.28 AT+CIPAP_DEF – Set ip address of softAP, save as default

AT+ CIPAP_DEF – Set ip address of ESP8266 softAP, save to Flash	
Type ： query Function: Get ip address of ESP8266 softAP. Command： **AT+CIPAP_DEF?**	Response： +CIPAP_DEF:<ip> OK
	Param description： <ip> string, ip address of ESP8266 softAP
Type ： set Function: Set ip address of ESP8266 softAP. Command： **AT+CIPAP_DEF** **=<ip>**	Response： OK
	Param description： <ip> string, ip address of ESP8266 softAP
Note	This configuration will store in Flash user parameter area.
Example	AT+CIPAP_DEF="192.168.5.1"

4.2.29 AT+CWSTARTSMART – Start SmartConfig

AT+CWSTARTSMART – Start SmartConfig	
Type ： set Function: Start SmartConfig. Command： **AT+CWSTARTSMART** **=<type>**	Response： OK
	Param description： < type> SmartConfig protocol type 1：ESP_TOUCH 2：AirKiss
Note	1. You can apply for more documents about our SmartConfig from Espressif. 2. ESP8266 station has to be enable 3. Message “Smart get wifi info” means Smart Config succeed, then you can use “AT+CIFSR”

	to check whether it got ip from router or not. 4. ESP8266 can't do anything during SmartConfig so please wait till it succeed or use command "AT+CWSTOPSMART" to stop SmartConfig.
Example	AT+CWMODE=3 AT+CWSTARTSMART=1

4.2.30 AT+CWSTOPSMART – stop SmartConfig

AT+CWSTOPSMART stop SmartConfig	
Type ： Execute Function: stop SmartConfig. Command: **AT+CWSTOPSMART**	Response： OK
Note	No matter SmartConfig succeed or not, please always call "AT+CWSTOPSMART" to release the buffer it took.
Example	AT+CWSTOPSMART

5 TCP/IP Related

5.1 Overview

TCP/IP	
Command	**Description**
AT+ CIPSTATUS	Get connection status
AT+CIPSTART	Establish TCP connection or register UDP port
AT+CIPSEND	Send data
AT+CIPCLOSE	Close TCP/UDP 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

5.2 TCP/IP

5.2.1 AT+CIPSTATUS – Information about connection

AT+ CIPSTATUS – Information about connection	
Type ： execute Function: Get information about connection. Command: **AT+** **CIPSTATUS**	Response： STATUS:<stat> + CIPSTATUS:<id>,<type>,<remote_ip>,<remote_port>,<local_port>,<tetype> OK
	Param description： <stat> 2: Got IP 3: Connected 4: Disconnected 5: Didn‘t connect to AP <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

5.2.2 AT+CIPSTART – Start connection

AT+CIPSTART – Establish TCP connection or register UDP port, start connection	
Type ： test Function: Get the information of param. Command: **AT+CIPSTART=?**	Response： 1) If AT+CIPMUX=0 +CIPSTART:(<type>),(<IP address>),(<port>)[,(<local port>),(<mode>)] +CIPSTART:(<type>),(<domain name>),(<port>)[,(<local port>),(<mode>)]

	OK 2) If AT+CIPMUX=1 +CIPSTART:(id),(<type>),(<IP address>),(<port>)[,(<local port>),(<mode>)] +CIPSTART: (id), (<type>),(<domain name>),(<port>)[,(<local port>),(<mode>)]
	Param description：null
Type ： Set Function: Start a connection as client. Command: 1)Single connection **(+CIPMUX=0)** **AT+CIPSTART=** **<type>,<addr>,<port>** **[,(<local port>),(<mode>)]** 2)Multiple connection (+CIPMUX=1) **AT+CIPSTART=** **<id><type>,<addr>,<port>** **[,(<local port>),(<mode>)]**	Response： OK or ERROR If connection already exists, returns ALREAY CONNECT
	Param description： <id> 0-4 , id of connection <type> string, “TCP” or “UDP” <addr> string, remote ip <port> string, remote port [<local port>] for UDP only [<mode>] for UDP only 0 : destination peer entity of UDP will not change. 1 : destination peer entity of UDP can change once. 2 : destination peer entity of UDP is allowed to change. Note: [<mode>] can only be used when [<local port>] is set.
Example	AT+CIPSTART="TCP","192.168.101.110",1000 Refer to “Espressif AT Command Examples”

5.2.3 AT+CIPSEND – Send data

AT+CIPSEND – Send data	
Type ： test Function:	Response：

Only for test. Command: **AT+CIPSEND=?**	OK Param description： null
Type ： Set Function: Set length of the data that will be sent. For normal send. Command: 1)For single connection: (+CIPMUX=0) **AT+CIPSEND=<length>** 2) For multiple connection: (+CIPMUX=1) **AT+CIPSEND=** **<id>,<length>** 3) For UDP transmission, remote ip & port can be set: **AT+CIPSEND=** **[<id>,]<length>** **[,<ip>,<port>]**	Wrap return “>” after set command. Begins receive of serial data, when data length is met, starts transmission of data. If connection cannot be established or gets disconnected during send, returns ERROR If data is transmitted successfully, returns SEND OK Note: This CMD Param description： <id> ID no. of transmit connection <length> data length, MAX 2048 bytes
Type ： execute Function: Send data. For unvarnished transmission mode. Command: **AT+CIPSEND**	Response： Wrap return “>” after execute command. Enters unvarnished transmission, 20ms interval between each packet, maximum 2048 bytes per packet. When single packet containing “+++” is received, it returns to command mode. This command can only be used in unvarnished

	transmission mode which require to be single connection mode.
Example	Refer to “Espressif AT Command Examples”

5.2.4 AT+CIPCLOSE – Close TCP or UDP connection

AT+CIPCLOSE – Close TCP or UDP connection	
Type : test Function: Only for test. Command: **AT+CIPCLOSE=?**	Response: OK
Type : Set Function: Close TCP or UDP connection. Command: For multiply connection mode **AT+CIPCLOSE=<id>**	Response: No errors, returns OK If connection <id> is disconnected, returns Link is not Param description: <id> ID no. of connection to close, when id=5, all connections will be closed. (id=5 has no effect in server mode)
Type : execute Command: For single connection mode **AT+CIPCLOSE**	Response: OK or If no such connection, returns ERROR Prints UNLINK when there is no connection

5.2.5 AT+CIFSR – Get local IP address

AT+CIFSR – Get local IP address	
Type : Test Function: Only for test. Command: **AT+CIFSR=?**	Response: OK
Type : Execute	Response:

Function: Get local IP address. Command: **AT+ CIFSR**	+ CIFSR:<IP address> + CIFSR:<IP address> OK ERROR
	Param description： <IP address> IP address of ESP8266 softAP IP address of ESP8266 station

5.2.6 AT+CIPMUX – Enable multiple connections

AT+ CIPMUX – Enable multiple connections or not	
Type ： Query Function: Get param config. Command: **AT+ CIPMUX?**	Response： + CIPMUX:<mode> OK
	Param description： The same as below.
Type ： Set Function: Set connection mode. Command: **AT+ CIPMUX=<mode>**	Response： OK If already connected, returns Link is builded
	Param description： <mode>0 single connection 1 multiple connection
Note	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. If server is started, reboot is required.
Example	AT+CIPMUX=1

5.2.7 AT+CIPSERVER – Configure as TCP server

AT+ CIPSERVER – Configure as TCP server	
Type ： Set Function: Set TCP server. Command:	Response： OK

AT+ CIPSERVER= <mode>[,<port>]	Param description： <mode> 0 Delete server (need to follow by restart) 1 Create server <port> port number, default is 333
Note	1、Server can only be created when AT+CIPMUX=1 2、Server monitor will automatically be created when Server is created. 3、When a client is connected to the server, it will take up one connection，be gave an id.
Example	AT+ CIPMUX=1 AT+ CIPSERVER=1,1001

5.2.8 AT+CIPMODE – Set transfer mode

AT+ CIPMODE – Set transfer mode	
Type ： Query Function: Query transfer mode. Command: **AT+ CIPMODE?**	Response： + CIPMODE:<mode> OK
	Param description： The same as below.
Type ： Set Function: Set transfer mode. Command: **AT+CIPMODE=<mode>**	Response： OK If already connected, returns Link is builded
	Param description： <mode>0 normal mode 1 unvarnished transmission mode
Note	This configuration would not save into Flash.
Example	AT+CIPMODE=1

5.2.9 AT+SAVETRANSLINK – Save transparent transmission link to Flash

AT+SAVETRANSLINK – Save transparent transmission link to Flash	
Type ： Set Function:	Response：

Save transparent transmission link to Flash. Command: **AT+SAVETRANSLINK** **=<mode>,<ip>,<port>**	OK or ERROR
	Param description： <mode>0 normal mode 1 transparent transmission mode <ip> remote ip <port> remote port
Note	1. This command will save the transparent transmission mode and its TCP link into Flash user parameter area. 2. As long as the ip, port numerical conformance to specification, we will save them to Flash
Example	AT+SAVETRANSLINK=1,”192.168.6.110”,1002

5.2.10 AT+CIPSTO – Set TCP server timeout

AT+ CIPSTO – Set TCP server timeout	
Type ： Query Function: Query server timeout. Command: **AT+CIPSTO?**	Response： + CIPSTO:<time> OK
	Param description： The same as below.
Type ： Set Function: Set server timeout. Command: **AT+CIPSTO=<time>**	Response： OK
	Param description： < time> TCP server timeout, range 0~7200 seconds
Note	ESP8266 as TCP server, will disconnect to TCP client that didn’t communicate with it even if timeout. If AT+CIPSTO=0, it will never timeout. We don’t recommend that.
Example	AT+ CIPMUX=1 AT+ CIPSERVER=1,1001 AT+CIPSTO=10

5.2.11 AT+CIUPDATE – Update through network

AT+ CIUPDATE – update through network	
Type ： execute Function: Start upgrade. Command: **AT+ CIUPDATE**	Response： +CIPUPDATE:<n> OK
	Param description： <n> 1 found server 2 connect server 3 got edition 4 start update
Note	Firmware upgrade depends on network condition. It will return ERROR if upgrade fail，please wait a while.

5.2.12 AT+PING – Function Ping

AT+PING – Function Ping	
Type ： set Function: Start upgrade. Command: **AT+PING=<ip>**	Response： +<time> OK Or ERROR // means ping fail
	Param description： <ip> : string, host ip or domain name <time> : response time of ping
Example	AT+PING=”192.168.1.1” AT+PING=”www.baidu.com”

5.2.13 +IPD – Receive network data

+IPD – Receive network data	
1)Single connection: **(+CIPMUX=0)** **+IPD,<len>:<data>** 2) Multiple connection	NOTE: When the module receives network data, it will send the data through the serial port using +IPD command Param description： <id> id no. of connection <len> data length

(+CIPMUX=1) **+IPD,<id>,<len>:<data>**	<data> data received

6 Appendix

ESP8266 AT commands below will save configuration into Flash:

Command	*Example*
Save in 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+SAVETRANSLINK	AT+SAVETRANSLINK =1,"192.168.6.10",1001
Save in 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

NOTE:

(1) We will check the new setting with original configuration from flash first，only if the configuration changes，we will write it to flash.

(2) To 512KB flash, default setting:

user parameter area is 0x3C000 ~ 0x40000, 16KB;

system parameter area is 0x7C000~0x80000, 16KB

To 1MB flash (or larger than 1MB), default setting:

user parameter area is 0x7C000 ~ 0x80000, 16KB;

system parameter area is the last 16KB of flash.

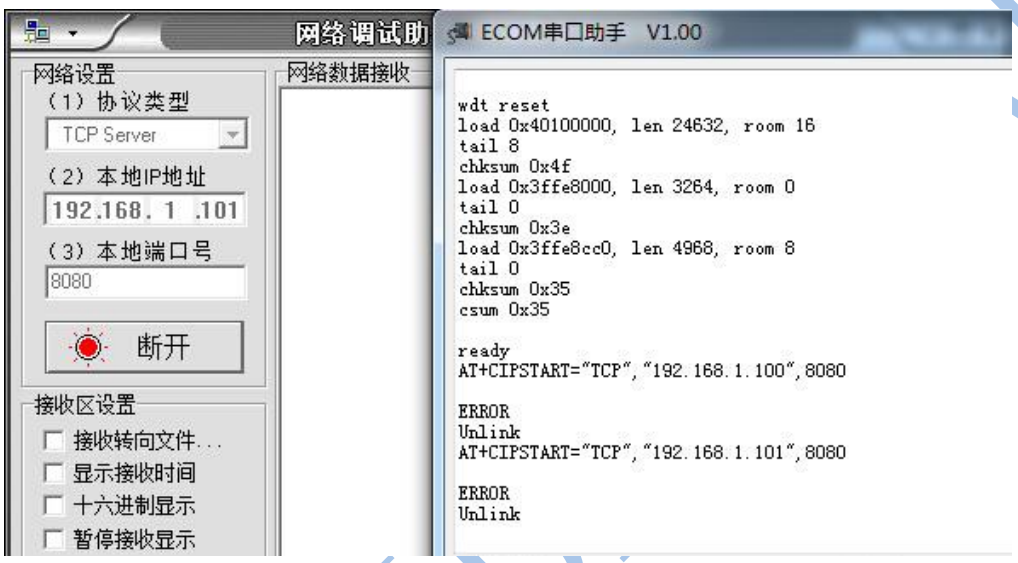
7 Q&A

If you have any question about AT Commands, please contact us (support-at@espressif.com) with information as follows:

(1) Version info of AT : Using “AT+GMR” to get the version info.

Hardware Module info：example AITHINK ESP-01

(2) Screenshot or steps of the test steps, for example:



(3) 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)
```