Wi-Fi Module AT Commands

Index of all known AT commands

Basic	WiFI layer	TCPIP Layer
AT	AT+CWMODE	AT+CIPSTATUS
AT+RST	AT+CWJAP	AT+CIPSTART
AT+GMR	AT+CWLAP	AT+CIPSEND
AT+GSLP	AT+CWQAP	AT+CIPCLOSE
ATE	AT+CWSAP	AT+CIFSR
	AT+CWLIF	AT+CIPMUX
	AT+CWDHCP	AT+CIPSERVER
	AT+CIPSTAMAC	AT+CIPMODE
	AT+CIPAPMAC	AT+CIPSTO
	AT+CIPSTA	AT+CIUPDATE
	AT+CIPAP	+IPD

Line termination

The module expects `<CR><LF>` or *Carriage Return* and *Line Feed* at the end of each command, but just`<CR>` seems to work.

Command variants

Each command can have up to 4 variants changing the *function* of it. You can chose between them by appending one of four possible values to the end of the root command itself. These four appendices can have the following

values `""`,`=<parameter|[parameters]>`,`"?"`,`=?`

Note:

- Not all AT commands support all 4 variants.
- [] = default value, not required or may not appear.
- String values require double quotation marks, for example:`AT+CWSAP="MOD756190","21030826",1,4`.
- Baud rate = 115200
- AT instruction ends with “\r\n”

Commands

AT - Test AT startup

Variant	Command	Response	Function
Execute	AT	OK	Test if AT system works correctly

AT+RST - Restart module

Variant	Command	Response	Function
Execute	AT+RST	OK	Reset the module

ESP-01 Output after reset:

```
 ets Jan  8 2013,rst cause:4, boot mode:(3,7)

wdt reset
load 0x40100000, len 24444, room 16
tail 12
chksum 0xe0
ho 0 tail 12 room 4
load 0x3ffe8000, len 3168, room 12
tail 4
chksum 0x93
load 0x3ffe8c60, len 4956, room 4
tail 8
chksum 0xbd
csum 0xbd

ready
```

ESP-12 Output after reset:

```
\0x04B1\0x85 \0xff\0x13:'\0xe0;\0xcc;!G\0xfa\0x11\0xa9R\0xc6\0x83\0x01\0xd9\0x81
[Vendor:www.ai-thinker.com Version:0.9.2.4]

ready
```

AT+GMR - View version info

Variant	Command	Response	Function
Execute	AT+GMR	version, OK	Print firmware version

Parameters:

- version: firmware version number

ESP-01 output:

```
00160901
```

ESP-12 output:

```
0018000902-AI03
```

AT+GSLP - Enter deep-sleep mode

Variant	Command	Response	Function
set	AT+GSLP=time	Time OK	Enter deep sleep mode for time milliseconds

parameters:

- time: Time to sleep in milliseconds

Example :

```
AT+GSLP=1500
```

Note:

Hardware has to support deep-sleep wake up (Reset pin has to be High).

ATE - Enable / Disable echo

Variant	Command	Response	Function
Execute	ATE0	OK	Disable echo (Doesn't send back received command)
Execute	ATE1	OK	Enable echo (Sends back received command before response)

AT+CWMODE - WIFI mode（station, AP, station + AP）

Variant	Command	Response	Function
Test	AT+CWMODE=?	+CWMODE:(1-3) OK	List valid modes
Query	AT+CWMODE?	+CWMODE:modeOK	Query AP's info which is connect by ESP8266.
Execute	AT+CWMODE=mode	OK	Set AP's info which will be connect by ESP8266.

Parameters:

- mode ： An integer designating the mode of operation either 1, 2, or 3.
  1 = Station mode (client)
  2 = AP mode (host)
  3 = AP + Station mode (Yes, ESP8266 has a dual mode!)

Notes:
ESP-12 came configured as **host** with ssid set to *ESP_A0A3F2*, no password, channel*1* You can use AT+CWSAP? to find the current settings.

AT+CWJAP - Connect to AP

Variant	Command	Response	Function
Query	AT+CWJAP?	+ CWJAP: ssid OK	Prints the SSID of Access Point ESP8266 is connected to.
Execute	AT+CWJAP=ssid,pwd	OK	Commands ESP8266 to connect a SSID with supplied password.

Parameters:

- ssid ： String, AP's SSID
- pwd ： String, not longer than 64 characters

Example ：

```
AT+CWJAP="my-test-wifi","1234test"
```

Example AT+CWJAP? ：

```
+CWJAP:"my-test-wifi"
```

AT+CWLAP - Lists available APs

Variant	Command	Response	Function
Set	AT+CWLAP=ssid,mac,ch	+CWLAP:ecn,ssid,rssi,macOK	Search available APs with specific c
Execute	AT+CWLAP	AT+CWLAP:ecn,ssid,rssi,macOK	Lists available Access Points.

Parameters:

- 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

Note:

On **ESP-01** I have had no luck with the set version of this command (AT+CWLAP=...). If you know what it does please let me know.
On **ESP-12**, the *Set* version of the command allows to see if a certain SSID, with certain MAC on certain channel exists. If it doesit is returned as one line of the*Execute* version of this command.

Example AT+CWLAP:

```
+CWLAP:(3,"CVBJB",-71,"f8:e4:fb:5b:a9:5a",1)
+CWLAP:(3,"HT_00d02d638ac3",-90,"04:f0:21:0f:1f:61",1)
+CWLAP:(3,"CLDRM",-69,"22:c9:d0:1a:f6:54",1)
+CWLAP:(2,"AllSaints",-88,"c4:01:7c:3b:08:48",1)
+CWLAP:(0,"AllSaints-Guest",-83,"c4:01:7c:7b:08:48",1)
+CWLAP:(0,"AllSaints-Guest",-83,"c4:01:7c:7b:05:08",6)
+CWLAP:(4,"C7FU24",-27,"e8:94:f6:90:f9:d7",6)
+CWLAP:(2,"AllSaints",-82,"c4:01:7c:3b:05:08",6)
+CWLAP:(3,"QGJTL",-87,"f8:e4:fb:b5:6b:b4",6)
+CWLAP:(4,"50EFA8",-78,"74:44:01:50:ef:a7",6)
+CWLAP:(0,"optimumwifi",-78,"76:44:01:50:ef:a8",6)
+CWLAP:(3,"BHQH4",-95,"18:1b:eb:1a:af:5b",6)
+CWLAP:(3,"NETGEAR49",-86,"84:1b:5e:e0:28:03",7)
+CWLAP:(3,"ngHub_319332NW00047",-56,"20:e5:2a:79:b1:2f",11)
+CWLAP:(3,"BFZR4",-73,"18:1b:eb:1d:c3:91",11)
+CWLAP:(1,"5FFVL",-82,"00:26:b8:b5:c0:f2",11)
+CWLAP:(3,"59G6D",-77,"00:7f:28:6d:91:7b",11)
+CWLAP:(3,"N16FU",-53,"20:cf:30:ce:60:fe",11)
+CWLAP:(3,"ITS",-82,"90:72:40:21:5f:76",11)
+CWLAP:(3,"ITS",-79,"24:a2:e1:f0:04:e4",11)
```

Example AT+CWLAP="N16FU","20:cf:30:ce:60:fe",11:

```
+CWLAP:(3,"N16FU",-53,"20:cf:30:ce:60:fe",11)
```

AT+CWQAP - Disconnect from AP

Variant	Command	Response	Function
Execute	AT+CWQAP	OK	Disconnect ESP8266 from the AP is currently connected to.

Note:
After running this command, if you run `AT+CWJAP?` it still shows the AP you were connected to before.

AT+CWSAP - Configuration of softAP mode

Variant	Command	Response	Function
Query	AT+CWSAP?	+CWSAP:ssid,pwd,ch,ecnOK	Query configuration of ESP8266 s
Set	AT+CWSAP=ssid,pwd,ch,ecn	OK	Set configuration of softAP mode.

Parameters:

- `ssid`: String, ESP8266's softAP SSID
- `pwd`: String, Password, no longer than 64 characters
- `ch`: channel id
- `ecn`:
  - **0** = OPEN
  - **2** = WPA_PSK
  - **3** = WPA2_PSK
  - **4** = WPA_WPA2_PSK

Example
`AT+CWSAP="esp_123","1234test",5,3`
`AT+CWSAP?` => `+CWSAP:"esp_123","1234test",5,3`

AT+CWLIF - List clients connected to ESP8266 softAP

Variant	Command	Response	Function
Execute	AT+CWLIF	[ip,other] OK	List information on of connected clients.

Parameters:
`ip`: IP address of a client connected to the ESP8266 softAP `other`: Other info, look at example. I don't know what it means yet.

Example (ESP-01):
```
AT+CWLIF

192.168.4.100,3fff50b4:3fff50ba:3fff50c0:3fff50c6:3fff50cc:3fff50d2

OK
```

Example (ESP-12):
```
AT+CWLIF

192.168.4.100,c0:ee:fb:25:33:ec

OK
```

AT+CWDHCP - Enable/Disable DHCP

Variant	Command	Response	Function
Set	AT+CWDHCP=mode,en	OK	Enable or disable DHCP for selected mode

Parameters:

- mode:
  - **0** : set ESP8266 as a softAP
  - **1** : set ESP8266 as a station
  - **2** : set both ESP8266 to both softAP and a station
- en:
  - **0** : Enable DHCP
  - **1** : Disable DHCP

Note:
This command doesn't seem to work on firmware *00160901* (ESP-01) nor*0018000902-AI03* (ESP-12).

AT+CIPSTAMAC - Set MAC address of ESP8266 station

Variant	Command	Response	Function
Query	AT+CIPSTAMAC?	+CIPSTAMAC:macOK	Print current MAC ESP8266's address.
Execute	AT+CIPSTAMAC=mac	OK	Set ESP8266's MAC address.

Parameters:

- mac : String, MAC address of the ESP8266 station.

Example:
AT+CIPSTAMAC="18:aa:35:97:d4:7b"

Note:
This command doesn't seem to work on firmware 00160901

AT+CIPAPMAC - Set MAC address of ESP8266 softAP

Variant	Command	Response	Function
Query	AT+CIPAPMAC?	+CIPAPMAC:macOK	Get MAC address of ESP8266 softAP.
Execute	AT+CIPAPMAC=mac	OK	Set mac of ESP8266 softAP.

Parameters:

- mac : String, MAC address of the ESP8266 softAP.

Example:
AT+CIPAPMAC="2c:aa:35:97:d4:7b"

AT+CIPSTA - Set IP address of Module station

Variant	Command	Response	Function
Query	AT+CIPSTA?	+CIPSTA:ipOK	Get IP address of ESP8266 station.
Execute	AT+CIPSTA=ip	OK	Set ip addr of ESP8266 station.

Parameters:

- ip : String, ip address of the ESP8266 station.

Example:
AT+CIPSTA=”192.168.101.108”

Note:
This command doesn’t seem to work on firmware 00160901

AT+CIPAP - Set ip address of Module softAP

Variant	Command	Response	Function
Query	AT+CIPAP?	+CIPAP:ip OK	Get ip address of ESP8266 softAP.
Execute	AT+CIPAP=ip	OK	Set ip addr of ESP8266 softAP.

Parameters:

- ip : String, ip address of ESP8266 softAP.

Example:

```
AT+CIPAP="192.168.5.1"
```

Note:
This command doesn’t seem to work on firmware 00160901

AT+CIPSTATUS - Information about connection

Variant	Command	Response	Function
Test	AT+CIPSTATUS=?	OK	
Execute	AT+CIPSTATUS	STATUS:status+CIPSTATUS:id,type,addr,port,tetypeOK	Get information about connection.

Parameters:

- status :
  - **2**: Got IP
  - **3**: Connected
  - **4**: Disconnected
- id : id of the connection (0~4), for multi-connect
- type : String, “TCP” or “UDP”
- addr : String, IP address.
- port : port number

- `tetype` :
    - **0** = Module runs as a client
    - **1** = Module runs as a server

Note:
On **ESP-01** this command returns `STATUS:1` instead (no extra info, but status changes)
On **0018000902-AI03** this command returns `STATUS:2` instead (no extra info, but status changes)

AT+CIPSTART - Establish TCP connection or register UDP port and start a connection

Variant	Command	Response	Function
Set	AT+CIPSTART=type,addr,port	OK	Start a connection as client. (Single connection mode)
Set	AT+CIPSTART=id,type,addr,port	OK	Start a connection as client. (Multiple connection mode)
Test	AT+CIPSTART=?	[+CIPSTART:(id)(“type”),(“ip address”),(port)] OK	List possible command variations)

Parameters:

- `id`: 0-4, id of connection
- `type`: String, “TCP” or “UDP”
- `addr`: String, remote IP
- `port`: String, remote port

AT+CIPSEND - Send data

Variant	Command	Response	Function
Test	AT+CIPSEND=?	OK	
Set	AT+CIPSEND=length	SEND OK	Set length of the data that will be sent. For normal send (single connection).
Set	AT+CIPSEND=id,length	SEND OK	Set length of the data that will be sent. For normal send (multiple connection).
Execute	AT+CIPSEND		Send data. For unvarnished transmission mode.

Normal Mode

Parameters:

- `id`: ID no. of transmit connection
- `length`: data length, MAX 2048 bytes

Unvarnished Transmission Mode
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.

AT+CIPCLOSE - Close TCP or UDP connection

Variant	Command	Response	Function
Test	AT+CIPCLOSE=?	OK	
Set	AT+CIPCLOSE=id	OK	Close TCP or UDP connection.For multiply connection mode
Execute	AT+CIPCLOSE	OK	Close TCP or UDP connection.For single connection mode

Parameters:

- id : ID no. of connection to close, when id=5, all connections will be closed.

Note:
In server mode, id = 5 has no effect!

AT+CIFSR - Get local IP address

Variant	Command	Response	Function
Test	AT+CIFSR=?	OK	
Execute	AT+CIFSR	+CIFSR:ip OK	Get local IP address.

Parameters:

- ip: IP address of the ESP8266 as an client.

Example AT+CIFSR:
```
10.101.10.134
```

AT+CIPMUX - Enable multiple connections or not

Variant	Command	Response	Function
Set	AT+CIPMUX=mode	OK	Enable / disable multiplex mode (up to 4 conenctions)
Query	AT+CIPMUX?	+CIPMUX:modeOK	Print current multiplex mode.

Parameters:

- mode:
    - **0**: Single connection
    - **1**: Multiple connections (MAX 4)

NOTE:

This mode can only be changed after all connections are disconnected. If server is started, reboot is required.

AT+CIPSERVER - Configure as server

Variant	Command	Response	Function
Set	AT+CIPSERVER=mode[,port]	OK	Configure ESP8266 as server

Parameters:

- 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.

AT+CIPMODE - Set transfer mode

Variant	Command	Response	Function
Query	AT+CIPMODE?	+CIPMODE:modeOK	Set transfer mode,normal or transparent transmission.
Set	AT+CIPMODE=mode	OK	Set transfer mode,normal or transparent transmission.

Parameters:

- mode:
- 0: normal mode
- 1: unvarnished transmission mode

AT+CIPSTO - Set server timeout

Variant	Command	Response	Function
Query	AT+CIPSTO?	+CIPSTO:time	Query server timeout.
Set	AT+CIPSTO=time	OK	Set server timeout.

Parameters:

- time: server timeout, range 0~7200 seconds

AT+CIUPDATE - update through network

!!! Don't run this unless you know what you're doing !!!
!!! It will likely brick your device !!! Attempts to self-update from the internet.

Variant	Command	Response	Function
Execute	AT+CIUPDATE	+CIPUPDATE:n OK	Start update through network

Parameters:
-n:
- 1: found server
- 2: connect server
- 3: got edition
- 4: start update

Example:
```
AT+CIUPDATE

+CIUPDATE: 1
+CIUPDATE: 2
+CIUPDATE: 3
+CIUPDATE: 4

\0x02\0x8cl\0x8el\0x8e\0x1cp\0x0c\0x8c\0xf2nn\0xee\0x00l\0x8c\0x8el`
\0x02\0x90\0x12\0x12nnl\0x8cl`\0x02\0x0e\0x02nr\0x8e\0x92\0x92n\0x0c\0x0c
\0x02\0x8c\0x92`\0x02`
\0xf2n\0x0c\0x0c\0x0c\0x9e\0xe0b\0x82nl\0x8c\0x0c\0x8c
\0xf2nn\0xee\0x00\0x0c\0x8e\0x0elp\0xf2n\0xe0\0x10\0x02\0x0c
\0x0cr\0x8c\0x9c\0x9c\0xe2\0xe0\0x0c\0x0c\0x0c
\0x0cb\0x0cn\0xe2|\0x02\0xec\0xecl\0x8c\0x0cb\0x8c\0xf2nn
...forever
```

+IPD - Receive network data

Variant	Command	Response	Function
Execute		+IP,len:data	Receive network data from single connection.
Execute		+IPD,id,len:data	Receive network data from multiple connection.

Parameters:

- id: id no. of connection
- len: data length
- data: data received