ESP8266 AT Command Examples

Version 1.3

About This Guide

The document gives some examples of using ESP8266 AT commands.

The document is structured as follows:

Chapter	Title	Content
Chapter 1	Overview	Provides instructions on downloading AT firmware.
Chapter 2	Single Connection as TCP Client	Introduces how to create a TCP client and establish a TCP connection.
Chapter 3	UDP Transmission	Introduces how to create a UDP transmission.
Chapter 4	UART-WiFi Passthrough Mode	Introduces how to create a UART-WiFi passthrough transmission.
Chapter 5	Multiple Connections as TCP Server	Introduces how to create a TCP server.
Chapter 6	Q & A	Provides information on where and how to consult questions about ESP8266 AT commands.

Release Notes

Date	Version	Release notes
2017.08	V1.3	Updated version.

Documentation Change Notification

Espressif provides email notifications to keep customers updated on changes to technical documentation. Please subscribe *here*.

Certifications

Please download the product certification(s) *here*.

Table of Contents

1. Overview 1
2. Single Connection as TCP Client 2
3. UDP Transmission 4
    3.1. UDP (remote IP and port are fixed) 4
    3.2. UDP (remote IP and port can be changed) 6
4. UART-WiFi Passthrough Mode 8
    4.1. TCP client single connection UART-WiFi passthrough 8
    4.2. UDP transmission UART-WiFi passthrough 9
5. Multiple Connections as TCP Server 11
6. Q&A 13

1. Overview

Herein we introduces some specific examples on the usage of ESP8266 AT Commands. For more information about the complete instruction set, please refer to documentation *4A-ESP8266__AT Instruction Set*.

- Download ESP8266 AT Bin: *http://www.espressif.com/en/support/download/at*
  - Users can download the AT bin files refer to the README.md.
- PC UART terminal emulator tool, to send AT commands to the ESP8266.
  - The default baud rate is 115200.
  - PC tool should be set into the "New Line Mode", AT commands end with a new line (CR LF).

> ⚠ ***Notice:***
>
> - *AT commands have to be capitalized.*

2. Single Connection as TCP Client

1. Set WiFi mode.

```
AT+CWMODE=3      // softAP+station mode
```

Response:

```
OK
```

2. Connect to a router.

```
AT+CWJAP="SSID","password"       // SSID and password of router
```

Response:

```
OK
```

3. Query the ESP8266 device's IP address.

```
AT+CIFSR
```

Response:

```
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"1a:fe:34:a5:8d:c6"
+CIFSR:STAIP,"192.168.3.133"
+CIFSR:STAMAC,"18:fe:34:a5:8d:c6"
OK
```

4. Connect PC to the same router that ESP8266 is connected to. Using a network tool on the computer to create a server.
   - For example, the TCP server on PC is 192.168.3.116, port 8080.
5. ESP8266 connects to the server as a TCP client.

```
AT+CIPSTART="TCP","192.168.3.116",8080          //protocol, server IP and port
```

Response:

```
OK
```

6. ESP8266 sends data to the server.

```
AT+CIPSEND=4    // set date length which will be sent,  such as 4 bytes
>test           // enter the data,  no CR
```

Response:

```
Recv 4 bytes
SEND OK
```

> ⚠ ***Notice:***
>
> - *If the number of bytes inputted are more than the size defined (n):*
>   - *the system will reply busy, and send the first n bytes.*
>   - *and after sending the first n bytes, the system will reply SEND OK.*

7. When ESP8266 received data from server, it will prompt message below:

```
+IPD,n:xxxxxxxxxx     // received n bytes,  data=xxxxxxxxxx
```

8. End the TCP connection.

```
AT+CIPCLOSE
```

Response:

```
CLOSED
OK
```

3. UDP Transmission

UDP transmission is established via command AT+CIPSTART.

1. Set WiFi mode.

```
AT+CWMODE=3      // softAP+station mode
```

Response:

```
OK
```

2. Connect to a router.

```
AT+CWJAP="SSID","password"       // SSID and password of router
```

Response:

```
OK
```

3. Query the ESP8266 device's IP address.

```
AT+CIFSR
```

Response:

```
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"1a:fe:34:a5:8d:c6"
+CIFSR:STAIP,"192.168.101.133"
+CIFSR:STAMAC,"18:fe:34:a5:8d:c6"
OK
```

4. Connect PC to the same router as ESP8266 is connected to. Using a network tool on the PC to create a UDP port.
   - For example, the PC's IP address is 192.168.101.116 and the port is 8080.
5. Below is two examples on UDP transmission.

3.1. UDP (remote IP and port are fixed)

In UDP transmission, whether remote IP and port is fixed or not is decided by the last parameter of "AT+CIPSTART". "0" means that the remote IP and port is fixed and cannot be changed. A specific ID is given to such connection, making sure that the data sender and receiver will not be replaced by other devices.

1. Enable multiple connection.

```
AT+CIPMUX=1
```

Response:

```
OK
```

2. Create a UDP transmission, for example, ID is 4

```
AT+CIPSTART=4,"UDP","192.168.101.110",8080,1112,0
```

Response:

```
4,CONNNECT
OK
```

> 📖 ***Note:***
>
> *"192.168.101.110", 8080 here is the remote IP and port of UDP transmission of the opposite side, i.e., the configuration set by PC.*
>
> *1112 is the local port of ESP8266. User can self-define this port. The value of this port will be random if it's not defined beforehand.*
>
> *0 means that remote IP and port is fixed and cannot be changed. For example, if another PC also creates a UDP entity and sends data to ESP8266 port 1112. ESP8266 can receive data sent from UDP port 1112, but when data is sent using AT command "AT+CIPSEND=4, X", it will still be sent to the first PC end. If this parameter is not 0, it will send to a new PC.*

3. Send data.

```
AT+CIPSEND=4,7          // Send 7 bytes to transmission NO.4
>UDPtest                // enter the data,  no CR
```

Response:

```
Recv 7 bytes
SEND OK
```

> ⚠️ ***Notice:***
>
> - *If the number of bytes inputted are more than the size defined (n):*
>   - *the system will reply busy, and send the first n bytes.*
>   - *and after sending the first n bytes, the system will reply SEND OK.*

4. When ESP8266 received data, it will prompt message below:

```
+IPD,4,n:xxxxxxxxxx     // received n bytes,  data=xxxxxxxxxx
```

5. End the UDP transmission.

```
AT+CIPCLOSE=4
```

Response:

```
4,CLOSED
OK
```

3.2. UDP (remote IP and port can be changed)

When creating a UDP transmission, set the last parameter of "AT+CIPSTART" to be "2". It means that the remote IP and port can be changed.

1. Create a UDP transmission.

```
AT+CIPSTART="UDP","192.168.101.110",8080,1112,2
```

Response:

```
CONNNECT
OK
```

> 📖 ***Note:***
>
> *"192.168.101.110", 8080 here is the remote IP and port of UDP transmission of the opposite side, i.e., the configuration set by PC.*
>
> *1112 is the local port of ESP8266. User can self-define this port. The value of this port will be random if it's not defined beforehand.*
>
> *2 means the remote IP and port will change to be the latest one that has been communicating with ESP8266. For example, if another PC also creates a UDP entity and sends data to ESP8266 port 1112. ESP8266 can receive data sent from UDP port 1112, and when data is sent using AT command "AT+CIPSEND=4, X", it will still be sent to the new one.*

2. Send data.

```
AT+CIPSEND=7            // Send 7 bytes
>UDPtest                // enter the data,  no CR
```

Response:

```
Recv 7 bytes
SEND OK
```

> ⚠ ***Notice:***
>
> - *If the number of bytes inputted are more than the size defined (n):*
>   - *the system will reply busy, and send the first n bytes.*
>   - *and after sending the first n bytes, the system will reply SEND OK.*

3. To send data to other UDP terminals, you can set the target IP and port when sending data. For example, send 7 bytes to the 192.168.101.111, port 1000.

```
AT+CIPSEND=7,"192.168.101.111",1000            // Send 7 bytes
>UDPtest                                       // enter the data,  no CR
```

Response:

```
Recv 7 bytes
SEND OK
```

4. When ESP8266 received data, it will prompt message below:

```
+IPD,n:xxxxxxxxxx    // received n bytes,  data=xxxxxxxxxx
```

5. End the UDP transmission.

```
AT+CIPCLOSE
```

Response:

```
CLOSED
OK
```

4. UART-WiFi Passthrough Mode

UART-WiFi passthrough mode can only be enabled when ESP8266 is working as TCP client creating a single connection, or in UDP transmission.

4.1. TCP client single connection UART-WiFi passthrough

Here is an example that ESP8266 station as TCP client to create a single connection and execute UART-WiFi passthrough transmission. For ESP8266 softAP, it can be set into UART-WiFi passthrough mode in the similar way.

1. Set WiFi mode.

```
AT+CWMODE=3      // softAP+station mode
```

Response:

```
OK
```

2. Connect to a router.

```
AT+CWJAP="SSID","password"       // SSID and password of router
```

Response:

```
OK
```

3. Query the ESP8266 device's IP address.

```
AT+CIFSR
```

Response:

```
+CIFSR:APIP,"192.168.4.1"
+CIFSR:APMAC,"1a:fe:34:a5:8d:c6"
+CIFSR:STAIP,"192.168.101.133"
+CIFSR:STAMAC,"18:fe:34:a5:8d:c6"
OK
```

4. Connect PC to the same router as ESP8266 is connected to. Using a network tool on the PC to create a TCP server.
   - For example, the PC's IP address is 192.168.101.110 and the port is 8080.
5. ESP8266 connects to the server as a TCP client.

```
AT+CIPSTART="TCP","192.168.101.110",8080        //protocol, server IP and port
```

Response:

```
OK
```

6. Enable UART-WiFi passthrough mode.

```
AT+CIPMODE=1
```

Response:

```
OK
```

7. ESP8266 starts sending data to the server.

```
AT+CIPSEND
```

Response:

```
>    //From now on, data received from UART will be transmitted to the server automatically.
```

8. Stop sending data

When receiving a packet that contains only "+++", the UART-WiFi passthrough transmission process will be stopped. Then please wait at least 1 second before sending next AT command.

Please be noted that if you input "+++" directly by typing, the "+++", may not be recognised as three consecutive "+" because of the Prolonged time when typing.

> 📖 ***Note:***
>
> *The aim of inputting "+++" is to exit UART-WiFi passthrough transmission and turn back to accept normal AT command, while the TCP connection still remains connected. So, we can also use command "AT+CIPSEND" to turn back into UART-WiFi passthrough transmission.*

9. Disable UART-WiFi passthrough mode.

```
AT+CIPMODE=0
```

Response:

```
OK
```

10.End the TCP connection.

```
AT+CIPCLOSE
```

Response:

```
CLOSED
OK
```

4.2. UDP transmission UART-WiFi passthrough

Here is an example that ESP8266 softAP create a UDP UART-WiFi passthrough transmission. For ESP8266 station, it can execute UDP UART-WiFi passthrough transmission in the similar way.

1. Set WiFi mode.

```
AT+CWMODE=3      // softAP+station mode
```

Response:

```
OK
```

2. Connect PC to ESP8266 softAP. Then using a network tool on PC to create a UDP port.

- For example, the PC's IP address is 192.168.4.2 and the port is 1001.

3. ESP8266 create a UDP transmission of which remote IP and port are fixed.

```
AT+CIPSTART="UDP","192.168.4.2",1001,2233,0
```

Response:

```
OK
```

4. Enable UART-WiFi passthrough mode.

```
AT+CIPMODE=1
```

Response:

```
OK
```

5. ESP8266 starts sending data.

```
AT+CIPSEND
```

Response:

```
>    //From now on, data received from UART will be transmitted automatically.
```

6. Stop sending data

When receiving a packet that contains only "+++", the UART-WiFi passthrough transmission process will be stopped. Then please wait at least 1 second before sending next AT command.

Please be noted that if you input "+++" directly by typing, the "+++", may not be recognized as three consecutive "+" because of the Prolonged time when typing.

> 📖 ***Note:***
>
> *The aim of inputting "+++" is to exit UART-WiFi passthrough transmission and turn back to accept normal AT command, while the TCP connection still remains connected. So, we can also use command "AT+CIPSEND" to turn back into UART-WiFi passthrough transmission.*

7. Disable UART-WiFi passthrough mode.

```
AT+CIPMODE=0
```

Response:

```
OK
```

8. Delete the UDP transmission.

```
AT+CIPCLOSE
```

Response:

```
CLOSED
OK
```

5. Multiple Connections as TCP Server

When ESP8266 is working as a TCP server, the multiple connections should be allowed, since there may be more than one client connecting to the ESP8266.

Here is an example showing how TCP server is realized when ESP8266 is working in softAP mode.

1. Set WiFi mode.

```
AT+CWMODE=3      // softAP+station mode
```

Response:

```
OK
```

2. Enable multiple connections.

```
AT+CIPMUX=1
```

Response:

```
OK
```

3. Create a TCP server.

```
AT+CIPSERVER=1       // default port = 333
```

Response:

```
OK
```

4. Connect PC to ESP8266 softAP. Then using a network tool on PC to create a TCP client and connect to the TCP server that ESP8266 created.

> 📖 ***Note:***
>
> *There is a timeout mechanism when ESP8266 working as a TCP server. If a TCP client has connected to the ESP8266 TCP server, but there is no data transmission for a period of time, then the server will stop the connection when timeout. To avoid such problems, please set up a data transmission circulation every 5 seconds.*

5. Send data.

```
AT+CIPSEND=0,4          // set date length which will be sent,  such as 4 bytes
>test           // enter the data,  no CR
```

Response:

```
Recv 4 bytes
SEND OK
```

> ⚠ ***Notice:***
>
> - *If the number of bytes inputted are more than the size defined (n):*
>   - *the system will reply busy, and send the first n bytes.*
>   - *and after sending the first n bytes, the system will reply SEND OK.*

6. When ESP8266 received data from server, it will prompt message below:

```
+IPD,0,n:xxxxxxxxxx    // received n bytes,  data=xxxxxxxxxx
```

7. End the TCP connection.

```
AT+CIPCLOSE=0
```

Response:

```
0,CLOSED
OK
```

6. Q&A

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

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

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

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

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

Espressif IoT Team

www.espressif.com