

Response: OK

3. Connect to router

AT+CWJAP="ssid", "password" // ssid and password of router

Response: OK 4. Query device's IP

AT+CIFSR

Response: 192.168.3.106 // Device got an IP from router.

5. Using a network tool (eg: "NetAssist.exe") on the computer to create a server.

For example, server ip address:192.168.3.116, port 8080

6. ESP8266EX connect to server as a client

AT+CIPSTART="TCP","192.168.3.116",8080 //protocol、server IP & port

Response: OK

7. Send data

AT+CIPSEND=4 // set date length which will be sent, such as 4 bytes

>DGFY // enter the data, no CR

Response: SEND OK

Note: If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

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

Transparent transmission

1. Set wifi mode:

AT+CWMODE=3 // softAP+station mode

Response: OK

2. Connect to router

AT+CWJAP="ssid","password" // ssid and password of router

Response: OK

3. Query device's IP

AT+CIFSR

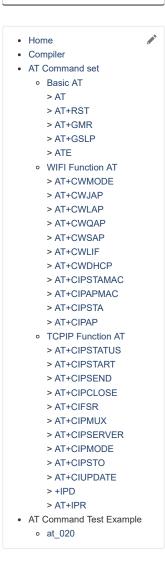
Response: 192.168.101.105 // Device's ip that got from router.

4. Using a network tool (eg: "NetAssist.exe") on the computer to create a server.

For example, server ip address:192.168.101.110, port 8080

5. Device connect to server

AT+CIPSTART="TCP","192.168.101.110",8080 // protocol、server IP & port



Clone this wiki locally

Clone in Desktop

https://github.com/espressi

Response: OK

Linked

6. Enable transparent transmission mode

AT+CIPMODE=1 Response: OK

7. Start send

AT+CIPSEND Response: >

Note: From now on, data received from uart will be transparent transmited to server.

Stop send

Data packet contains only "+++" exits Unvarnished transmission.

Multiple connection

(ESP8266 as TCP server)

1. Set wifi mode:

AT+CWMODE=3 // softAP+station mode

Response: OK

2. Enable multiple connection

AT+CIPMUX=1 Response: OK

3. Setup server

AT+CIPSERVER=1 // default port = 333

Response: OK

4. PC connects to ESP8266EX softAP as station, then PC connects to ESP8266EX server as client.

NOTE: ESP8266EX acting as server has a timeout mechanism. When connection is established and no data is transmitted within a period of time, it will disconnect the client. Please setup a recurring packet transmission every 5s on the computer to ensure connection is maintained.

5. Send data

AT+CIPSEND=4 // set date length which will be sent, such as 4 bytes

>iopd // enter the data, no CR

Response: SEND OK

Note: If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

6. Receive data: +IPD,n:xxxxxxxxxx // received n bytes, data = xxxxxxxxxxx

UDP transmission

1. Set wifi mode:

AT+CWMODE=3 // softAP+station mode

Response: OK
2. Connect to router

AT+CWJAP="ssid","password" // ssid and password of router

Response: OK
3. Query device's IP

AT+CIFSR

Response: +CIFSR:STAIP,"192.168.101.104" // IP address of ESP8266 station

4. PC connects to the same router which ESP8266 connects to. Using a network tool (eg: "NetAssist.exe") on the computer to create UDP.

5. Enable multiple connection

AT+CIPMUX=1 Response: OK

6. Create a UDP transmission, for example, id is 4.

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

Response: 4,CONNECT OK

Note:

"192.168.101.110",8080 here is the remote ip and port of UDP transmission which create on PC in step 4;

1112 is the local port of ESP8266, user-define, if user does not define it, it will be a random value:

0 means destination peer entity of UDP will not change. For example, in this case, if another PC also creates a UDP entity and sends data to ESP8266 port 1112, ESP8266 can receive these data, but when we send data with command "AT+CIPSEND=4,X", it will still be sent to the first PC. If this parameter is not 0, it will send to the new PC.

7. Send data

AT+CIPSEND=4,5 // Send 5 bytes to transmission NO.4

Response > DGFYQ // enter the data, no CR

SEND OK

Note: If the number of bytes sent is bigger than the size defined (n), will reply busy, and after sending n number of bytes, reply SEND OK.

8. Receive data:

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

9. Delete transmission NO.4

AT+CIPCLOSE=4

Response: 4,CLOSED OK

© 2016 GitHub, Inc. Terms Privacy Security Contact Help

Status API Training Shop Blog About