

# OPL1000

ULTRA-LOW POWER 2.4GHZ WI-FI + BLUETOOTH SMART SOC

## 透传应用指导文档



OPULINKS

<http://www.opulinks.com/>

Copyright © 2020, Opulinks. All Rights Reserved.

OPL1000-透传应用指导文档 | Version V04

Date	Version	Contents Updated
2019-04-20	0.1	<ul style="list-style-type: none"><li>Initial Release</li></ul>
2019-04-26	0.2	<ul style="list-style-type: none"><li>Update chapter 3 and 4</li></ul>
2019-06-2	0.3	<ul style="list-style-type: none"><li>Fixed several typos and adjust document format</li></ul>
2020-01-07	0.4	<ul style="list-style-type: none"><li>Add chapter 5,6, and7</li><li>Title change to Chinese</li><li>Modified Fig5/6/11/14/18/19 font=consolas</li><li>Modified chap1.1 description</li><li>Modified chap3.1 description</li><li>Modified chap3.4 description</li><li>Add fig14 comment</li></ul>

## TABLE OF CONTENTS

1. 介绍	1
1.1. 文档应用范围	1
1.2. 缩略语	1
1.3. 参考文献	1
2. 项目构成和工作原理	3
2.1. 工作原理	3
3. 运行透传参考设计应用	4
3.1. 使用 AT 命令连接无线 AP	5
3.2. 使用手机 APP 蓝牙配网连接无线 AP	6
3.3. 连接网站进行数据传输 (标准模式)	10
3.4. 连续传送模式(透传模式)	12
3.5. 返回 IDLE 模式	13
4. 透传工作 AT 指令举例	15
4.1. AT 命令连接 AP 百度完成搜索举例	15
4.2. 蓝牙配网 APP 连接 AP 完成百度搜索举例	15
4.3. 连续传送的方式完成百度搜索举例	16
5. 模式介绍	17
5.1. 系统模式(CWMODE)说明与设定	17
5.1.1. 支持 CWMODE 切换运用	19
5.1.2. BLEWIF MODE (CWMODE=4) 相关指令支持	20
5.2. 工作模式(SYSMODE)	21
5.3. 自动休眠功能(POWER SAVING)	22
6. 透传模式范例	23
6.1. 完成 AP 连接范例-Case1/Case2	23
6.2. 建立 Socket 连接范例-Case3/Case4	30
6.3. 开始透传范例-Case5/Case6/Case7	35
7. BLE 控制范例	44

LIST OF TABLES

Table 1 系统模式介绍 ..... 18

Table 2: 支持 CWMODE 切换 ..... 20

Table 3: BLEMODE mod 支持指令..... 20

Table 4: 工作模式介绍 ..... 22

Table 5: 透传模式步骤说明..... 23

## LIST OF FIGURES

Figure 1: 透传参考设计构成框图 .....	3
Figure 2: 下载透传功能固件 .....	4
Figure 3: Debug 串口输出信息 .....	5
Figure 4: 扫描并连接 AP 的 AT 命令 .....	5
Figure 5: 扫描并连接 AP AT 命令执行结果 .....	5
Figure 6: 进入 blewifi 配网模式 AT 命令 .....	6
Figure 7: 手机 APP 扫描蓝牙设备 .....	6
Figure 8: 连接 OPL1000 设备后扫描 AP .....	8
Figure 9: 连接选择的 AP .....	9
Figure 10: 连接百度搜索“Arduino”关键词 .....	10
Figure 11: 百度搜索“arduino”返回信息 .....	11
Figure 12: 浏览器百度搜索“arduino”返回信息 .....	11
Figure 13: 连续传送 AT 指令举例 .....	12
Figure 14: 结束连续传送执行结果 .....	12
Figure 15: 返回 IDLE AT 指令执行结果 .....	14
Figure 16: AT 命令连接 AP 百度搜索 AT 指令集合 .....	15
Figure 17: 蓝牙配网 APP 连接 AP 百度搜索 AT 指令集合 .....	15
Figure 18: 连续传送方式百度搜索 AT 指令集合 .....	16
Figure 19: 模式介绍架构图 .....	17
Figure 20: CWMODE 状态机切换示意图 .....	19
Figure 21: case1-使用 WIFI mode & AT Command 连接 AP .....	23
Figure 22: case1-其他状况 .....	26
Figure 23: case2-使用 BLEWIFI mode 连接 AP .....	27

Figure 24: case2-其他状况 ..... 29

Figure 25: case3-单点联机 ..... 30

Figure 26: case3-其他状况 ..... 31

Figure 27: case4-多点联机 ..... 32

Figure 28: case4-其他状况 ..... 35

Figure 29: case5-单点联机+一般透传 ..... 36

Figure 30: case5-其他状况 ..... 38

Figure 31: case6-多点联机+一般透传 ..... 39

Figure 32: case6-其他状况 ..... 41

Figure 33: case7-单点联机+进阶透传 ..... 41

Figure 34: case7-其他状况 ..... 42

Figure 35: case1-IDLE mode /WIFI mode 进行 BLE 控制 ..... 44

Figure 36: case1-其他状况 ..... 46

Figure 37: case2-BLEWIFI mode 进行 BLE 控制 ..... 47

## 1. 介绍

### 1.1. 文档应用范围

本文档介绍如何基于 OPL1000 使用透传模式实现网络连接和数据交换。所谓透传(Transparent Transmission)是指 OPL1000 可以连续发送数据，而无需每次发送数据前,再执行相关 AT 命令。在这种模式下 OPL1000 作为从设备受主设备（例如外部 MCU）控制。主设备发送 AT 命令给 OPL1000，完成因特网服务器的 TCP 连接，然后通过发送和接收 TCP 包实现数据的交换。

### 1.2. 缩略语

Abbr.	Explanation
AP	Wireless Access Point 无线访问接入点
APP	APPLication 应用程序
APS	Application Sub-system 应用子系统，在本文中亦指 M3 MCU
Blewifi	BLE config WIFI 蓝牙配网应用
DevKit	Development Kit 开发工具板
OTA	Over-the-Air Technology 空间下载技术
TCP	Transmission Control Protocol 传输控制协议

### 1.3. 参考文献

[1] OPL1000 数据手册 OPL1000-DS-NonNDA.pdf;  
访问链接：<https://github.com/Opulinks-Tech/OPL1000-HDK/OPL1000-DS-NonNDA.pdf>

[2] Download 工具使用指南 OPL1000-patch-download-tool-user-guide.pdf  
访问链接：[https://github.com/Opulinks-Tech/OPL1000A2-SDK/tree/master/Doc/zh\\_CN/OPL1000-patch-download-tool-user-guide.pdf](https://github.com/Opulinks-Tech/OPL1000A2-SDK/tree/master/Doc/zh_CN/OPL1000-patch-download-tool-user-guide.pdf)

[3] AT 指令使用指南文档 OPL1000-AT-instruction-set-and-examples.pdf  
访问链接：

[https://github.com/Opulinks-Tech/OPL1000A2-SDK/tree/master/Doc/zh\\_CN3OPL1000-AT-instruction-set-and-examples.pdf](https://github.com/Opulinks-Tech/OPL1000A2-SDK/tree/master/Doc/zh_CN3OPL1000-AT-instruction-set-and-examples.pdf)

[4] 手机 APP 蓝牙配网操作文档 OPL1000-Demo-BLE-setup-network-guide.pdf

访问链接:

[https://github.com/Opulinks-Tech/OPL1000A2-SDK/tree/master/Demo/BLE\\_Config\\_AP/OPL1000-Demo-BLE-setup-network-and-BLE-OTA-guide.pdf](https://github.com/Opulinks-Tech/OPL1000A2-SDK/tree/master/Demo/BLE_Config_AP/OPL1000-Demo-BLE-setup-network-and-BLE-OTA-guide.pdf)



## 2. 项目构成和工作原理

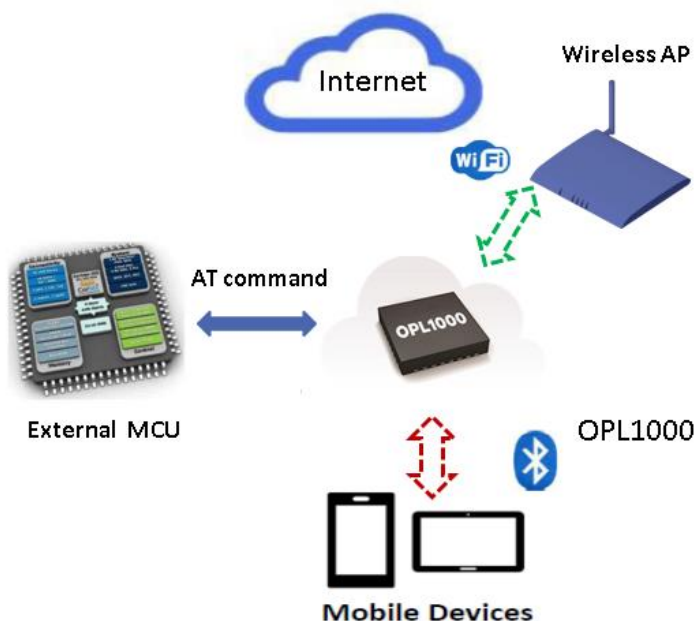
### 2.1. 工作原理

透传参考设计包含 4 个主要部件：物联网模块 OPL1000，移动设备（安装蓝牙配网 APP），云端因特网和外部主控设备（External MCU）。一般地外部主控设备连接若干传感器和外设，采集传感器信息或者控制外设。当数据需要上传到云端时，数据被打包成特定的网络协议包（例如 http 或者 MQTT 数据包），以透传的方式发送；当接受云端控制或者反馈消息时，主控设备通过 OPL1000 获得特定协议数据包，进行解析，然后控制本地外设。

工作过程为：

1. 首先完成 OPL1000 连接到无线 AP 操作。有两种方式实现：第一种是外部主控设备通过 AT 命令控制 OPL1000 完成 AP 扫描和 AP 连接；第二种是使用手机 APP 和 OPL1000 设备通信完成蓝牙配网操作。
2. 主控设备通过 AT 命令连接到因特网特定服务器，建立 TCP 连接。
3. 采用透传方式发送 TCP 包，实现和因特网云端的数据交换。

Figure 1: 透传参考设计构成框图

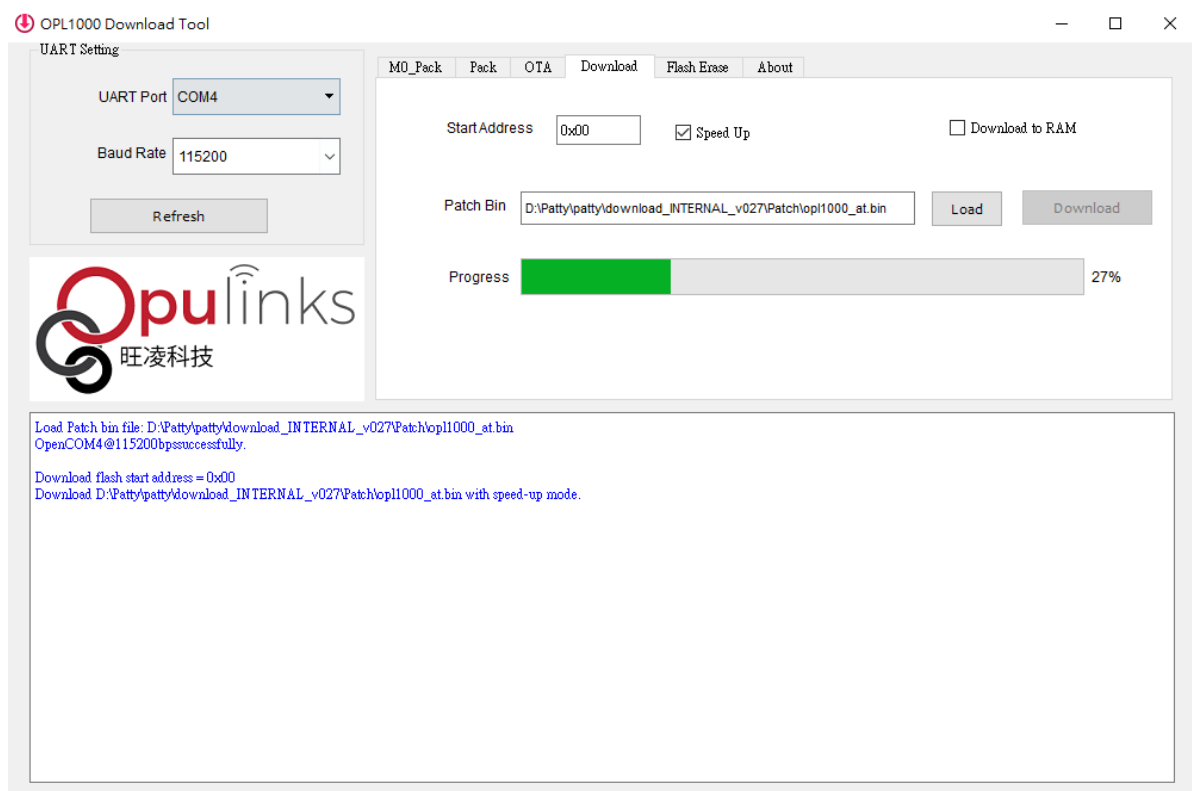


### 3. 运行透传参考设计应用

运行 OPL1000 透传参考设计应用包含 3 个步骤。

- 1 使用固件下载工具，下载位于 FW\_Binary 目录下的 opl1000\_at.bin 固件。
- 2 使用 AT 命令或者蓝牙配网 APP 完成和无线 AP 的连接。
- 3 使用 AT 命令连接云端服务器，建立 TCP 连接。发送和接收 TCP 数据。

Figure 2: 下载透传功能固件

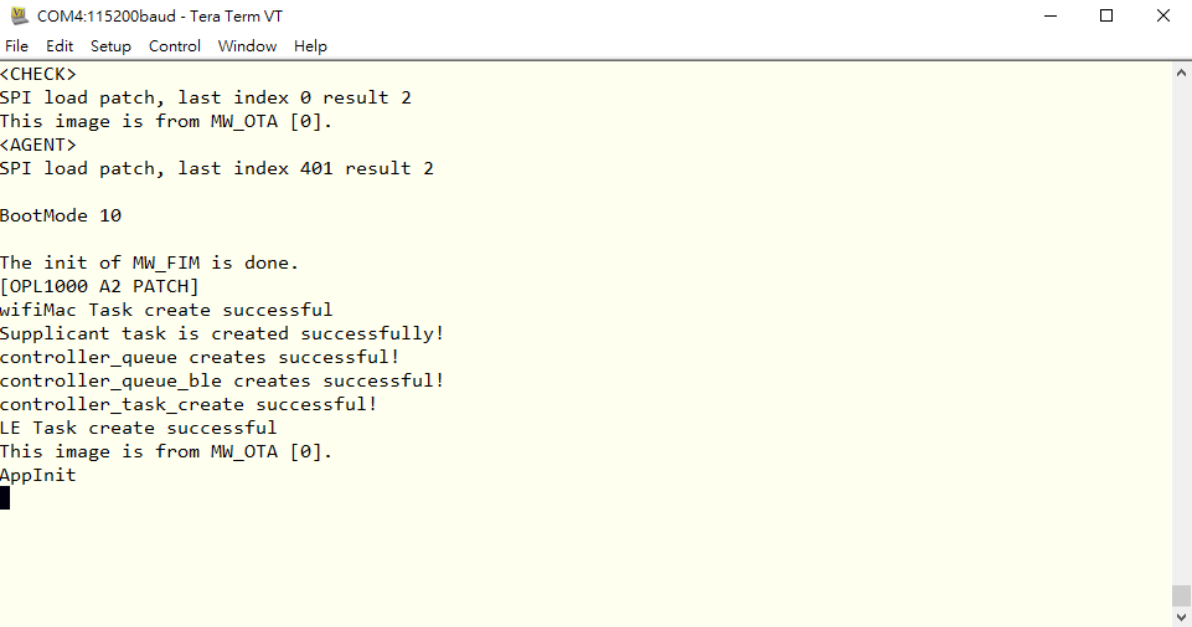


固件下载完成后 IO8/IO9 UART 端口将作为 AT 命令通信通道；IO0/IO1 UART 端口作为 Debug log 信息的输出通道。

如图 Figure 2 所示，COM121 对应于 IO0/IO1 UART 端口。ROM CODE boot loader 进行固件下载时 使用 IO0/IO1 UART 端口；固件载入 RAM 执行后，COM121 是 Debug log 信息的输出端口。显示固件下载成功后输出的调试打印信息。

Figure 3 显示固件下载成功后输出的调试打印信息。

Figure 3:Debug 串口输出信息



### 3.1. 使用 AT 命令连接无线 AP

如 2.2 节所述，OPL1000 使用 AT 命令连接无线 AP。工作原理是主设备使用 IO8/IO9 连接的 UART 端口执行若干 AT 命令。如图 Figure 4 所示。在本例中将 PC 作为主设备，使用串口调试工具执行 AT 命令。

Figure 4:扫描并连接 AP 的 AT 命令

```
at+cwmode=1           // 进入纯 AT Cmd 模式 (要从 mode 0: Idle · 才能进行此切换)
at+cwlap              // wifi scan
at+cwjap="Opulinks-TEST-AP","1234abcd"OK // wifi connect
```

注意 AT 指令的后面要加上回车换行，因此上图中的 “at+cwmode=1” 命令对应的完整字符串为 “at+cwmode=1\r\n”

执行后得到 Figure 5 所示结果。这里我们选择 连接 Opulinks-TEST-AP。AT 指令的使用请参考文档[\[4\]](#) AT 指令使用指南文档 OPL1000-AT-instruction-set-and-examples.pdf

Figure 5: 扫描并连接 AP AT 命令执行结果

```
>at+cwmode=1
OK

>at+cwlap
+CWLAP:3,Opulinks-TEST-AP,-37,44:c3:46:11:41:7f,1
+CWLAP:4,ziroomer,-68,d4:ee:07:52:c8:68,2
+CWLAP:4,ChinaNet-CHca,-58,60:b6:17:50:ab:56,8
+CWLAP:4,UTT-MICHAEL,-45,fc:2f:ef:68:cd:10,9
+CWLAP:4,ChinaNet-eHZU,-67,2c:dd:95:29:3d:e5,9
OK

>at+cwjap="Opulinks-TEST-AP","1234abcd"
WIFI CONNECTED
WIFI GOT IP
OK
```

### 3.2. 使用手机 APP 蓝牙配网连接无线 AP

当使用手机 APP 完成蓝牙配网时，需要先使用 `at+cwmode=0` 将 OPL1000 进入 IDLE 状态（无 WI-FI 模式）。然后使用 `at+cwmode=4` 进入 `blewifi` 配网模式。此模式会保存至 flash，重新启动会自动执行。

Figure 6: 进入 `blewifi` 配网模式 AT 命令

```
at+cwmode=0
at+rst           // 复位设备
at+cwmode=4      // 进入 blewifi 配网模式 (要从 mode 0: Idle · 才能进行此切换)
```

重新启动后，使用 蓝牙配网 APP “`opulinks_iot_app.apk`” 完成蓝牙设备扫描、连接和 AP 连接。具体可参考文献[5] 手机 APP 蓝牙配网操作文档 `OPL1000-Demo-BLE-setup-network-guide.pdf` AP 连接成功后可进入下一环节：连接云端（因特网）服务器或者网站。

Figure 7 展示 APP 扫描蓝牙设备得到的结果。这里 OPL1000 设备为 `OPL_33:44:55:66`

图 Figure9 展示点击 “WIFI Setup” 按钮后得到的 AP 列表。

图 Figure10 展示选择连接 UTT-MICHAEL AP 设备。

AP 连接成功后可进入下一环节：连接云端（因特网）服务器或者网站。

Figure 7: 手机 APP 扫描蓝牙设备

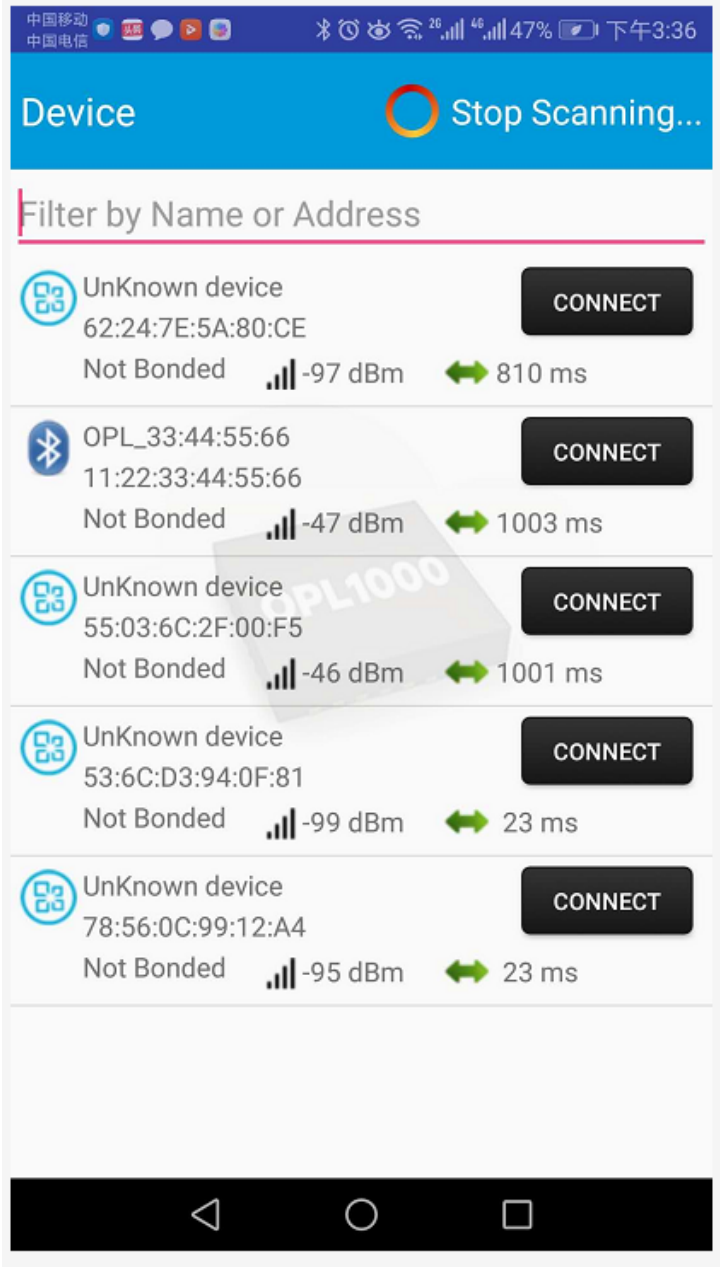
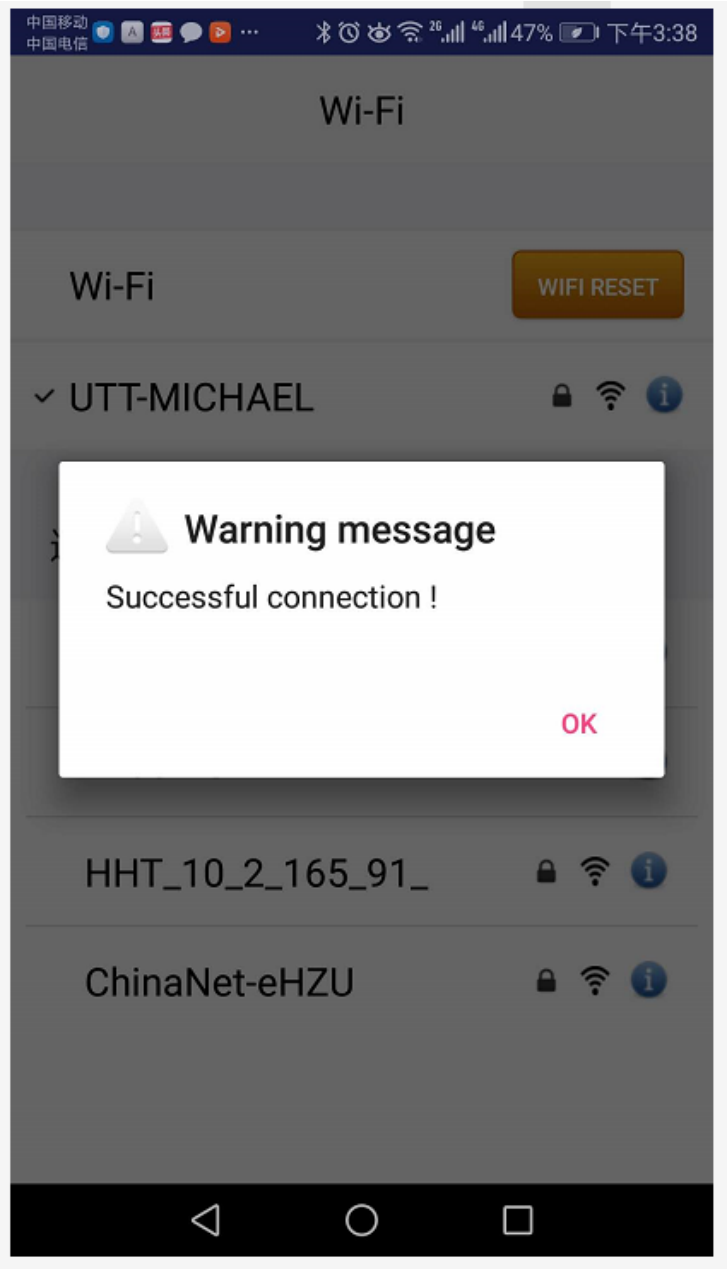


Figure 8: 连接 OPL1000 设备后扫描 AP



Figure 9: 连接选择的 AP



### 3.3. 连接网站进行数据传输 (标准模式)

完成 AP 连接后，就可以通过 TCP 连接指定的网站。这里我们以“连接百度网站，搜索 arduino 关键词”为例介绍如何通过 AT 命令完成这个操作。

AT 指令包括两步。第一步通过 AT+CIPSTART 和百度网站建立 TCP 连接。第二步通过 AT+CIPSEND 命令发送 TCP 数据包，即 http get 命令；get 命令指明完成“arduino”关键词搜索。注意 TCP 数据包的字节数包括每一行的字符串加回车换行符（2 个字节）；最后一行需要加回车换行是 http 协议规范要求的。因此 TCP 数据包总共是四行字符串。完整字符串表达形式如下所示。注意在实际输入时前后的引号是没有的。“\r\n”等于回车换行。

```
"GET /s?wd=arduino HTTP/1.1\r\n"  
"Host: www.baidu.com\r\n"  
"Connection: close\r\n"  
"\r\n"
```

完整的建立 TCP 连接，发送 http get 命令 AT 指令如图 Figure 10 所示。

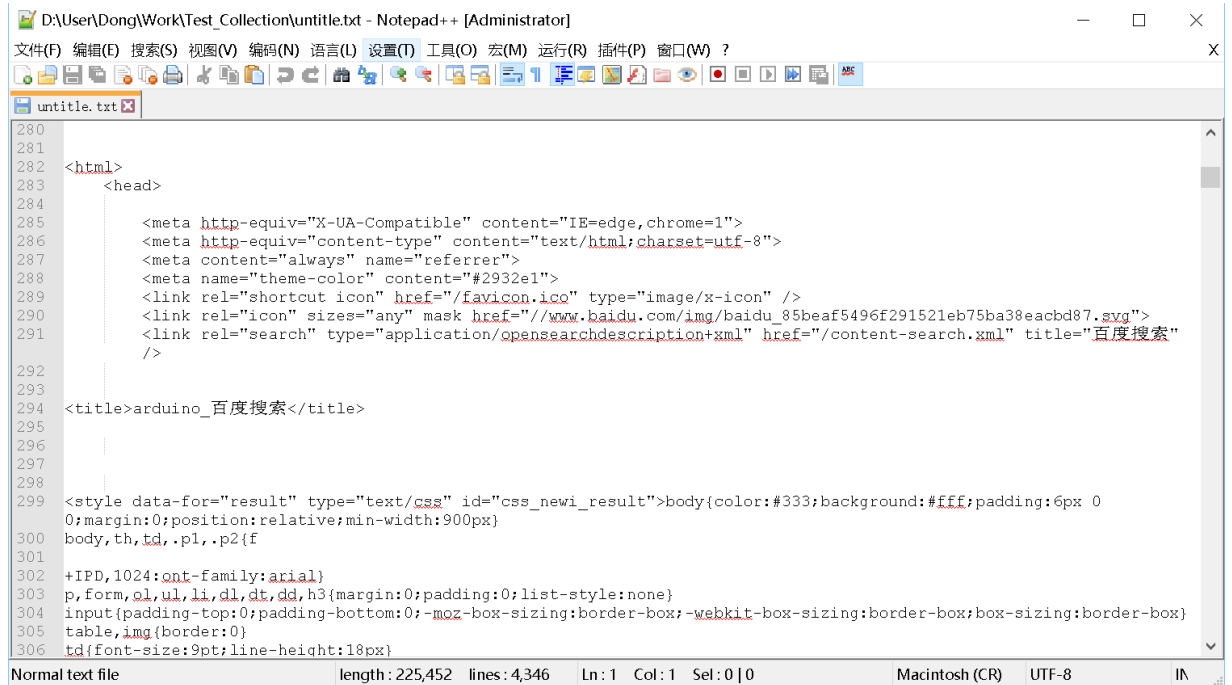
Figure 10: 连接百度搜索“Arduino”关键词

```
AT+CIPSTART="TCP","www.baidu.com",80  
AT+CIPSEND=70  
GET /s?wd=arduino HTTP/1.1  
Host: www.baidu.com  
Connection: close
```

指令正确执行后，在串口上可以得到 Figure12 所示数据，它们是网站返回的 html 格式网页数据。



Figure 11: 百度搜索“arduino” 返回信息



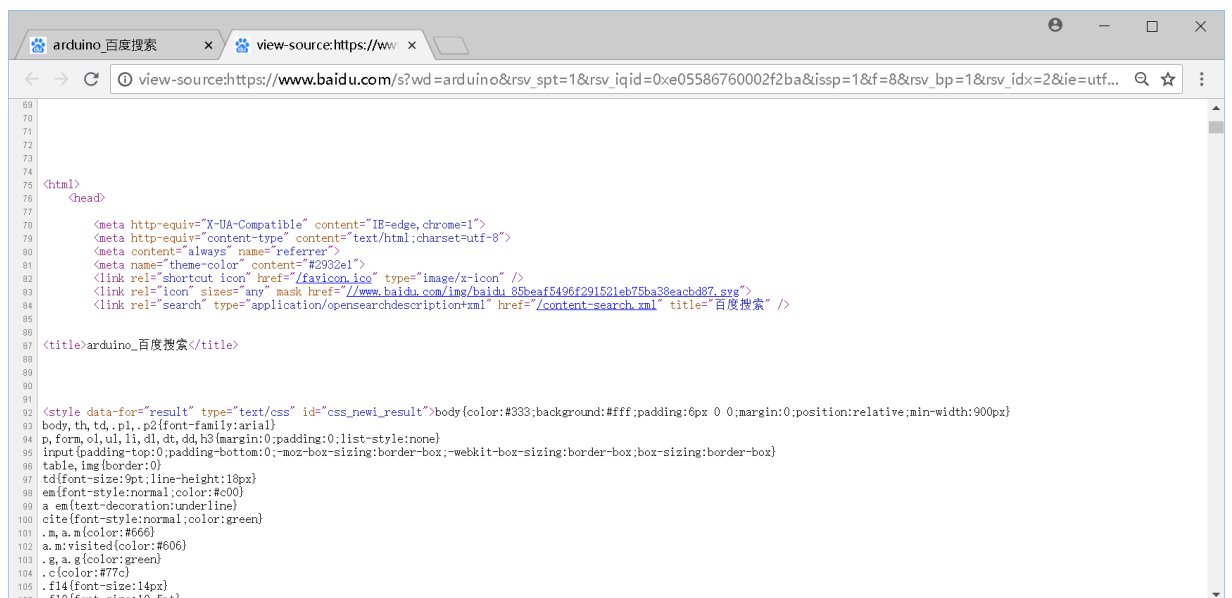
```

280
281
282 <html>
283   <head>
284
285     <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
286     <meta http-equiv="content-type" content="text/html; charset=utf-8">
287     <meta content="always" name="referrer">
288     <meta name="theme-color" content="#2932e1">
289     <link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />
290     <link rel="icon" sizes="any" mask href="//www.baidu.com/img/baidu_85beaf5496f291521eb75ba38eacbd87.svg">
291     <link rel="search" type="application/opensearchdescription+xml" href="/content-search.xml" title="百度搜索" />
292
293
294   <title>arduino_百度搜索</title>
295
296
297
298
299   <style data-for="result" type="text/css" id="css_newi_result">body{color:#333;background:#fff;padding:6px 0 0;margin:0;position:relative;min-width:900px}
300   body,th,td,.p1,.p2{f
301
302   +IPD,1024:ont-family:arial}
303   p,form,ol,ul,li,dl,dt,dd,h3{margin:0;padding:0;list-style:none}
304   input{padding-top:0;padding-bottom:0;-moz-box-sizing:border-box;-webkit-box-sizing:border-box;box-sizing:border-box}
305   table,img{border:0}
306   td{font-size:9pt;line-height:18px}
  
```

Normal text file      length: 225,452    lines: 4,346    Ln: 1    Col: 1    Sel: 0 | 0      Macintosh (CR)    UTF-8    IN

在浏览器百度网址搜寻“arduino”得到的网页源码如 Figure 12 所示。和 Figure 12 对比，它们的结果是相同的。说明通过 AT 指令透传 http 请求返回了正确的结果。

Figure 12: 浏览器百度搜索“arduino” 返回信息



```

69
70
71
72
73
74
75 <html>
76   <head>
77
78     <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
79     <meta http-equiv="content-type" content="text/html; charset=utf-8">
80     <meta content="always" name="referrer">
81     <meta name="theme-color" content="#2932e1">
82     <link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />
83     <link rel="icon" sizes="any" mask href="//www.baidu.com/img/baidu_85beaf5496f291521eb75ba38eacbd87.svg">
84     <link rel="search" type="application/opensearchdescription+xml" href="/content-search.xml" title="百度搜索" />
85
86
87   <title>arduino_百度搜索</title>
88
89
90
91
92
93   <style data-for="result" type="text/css" id="css_newi_result">body{color:#333;background:#fff;padding:6px 0 0;margin:0;position:relative;min-width:900px}
94   body,th,td,.p1,.p2{font-family:arial}
95   p,form,ol,ul,li,dl,dt,dd,h3{margin:0;padding:0;list-style:none}
96   input{padding-top:0;padding-bottom:0;-moz-box-sizing:border-box;-webkit-box-sizing:border-box;box-sizing:border-box}
97   table,img{border:0}
98   td{font-size:9pt;line-height:18px}
99   em{font-style:normal;color:#c00}
100  a em{text-decoration:underline}
101  .m,a.m{color:#666}
102  .a.m.visited{color:#066}
103  .g,a.g{color:green}
104  .c{color:#77c}
105  .f14{font-size:14px}
106  .f10{font-size:10.5pt}
  
```

### 3.4. 连续传送模式(透传模式)

在 3.3 章节发送的 TCP 数据包使用了 AT+CIPSEND=70 指令，数据包的长度是固定的。

在某些应用场合数据包的发送是不固定的，或者是连续的。针对此情况，可使用 [ AT+CIPMODE=1 ] 及 [ AT+CIPSEND ]指令，来取代[ AT+CIPSEND=70 ]，然后持续输入带回车换行的字符串。当需要结束传送时，需要在 20ms 内输入 “+++” 指令完成传送。以百度搜索 “arduino” 为例对应的 AT 指令如 Figure 13 所示。

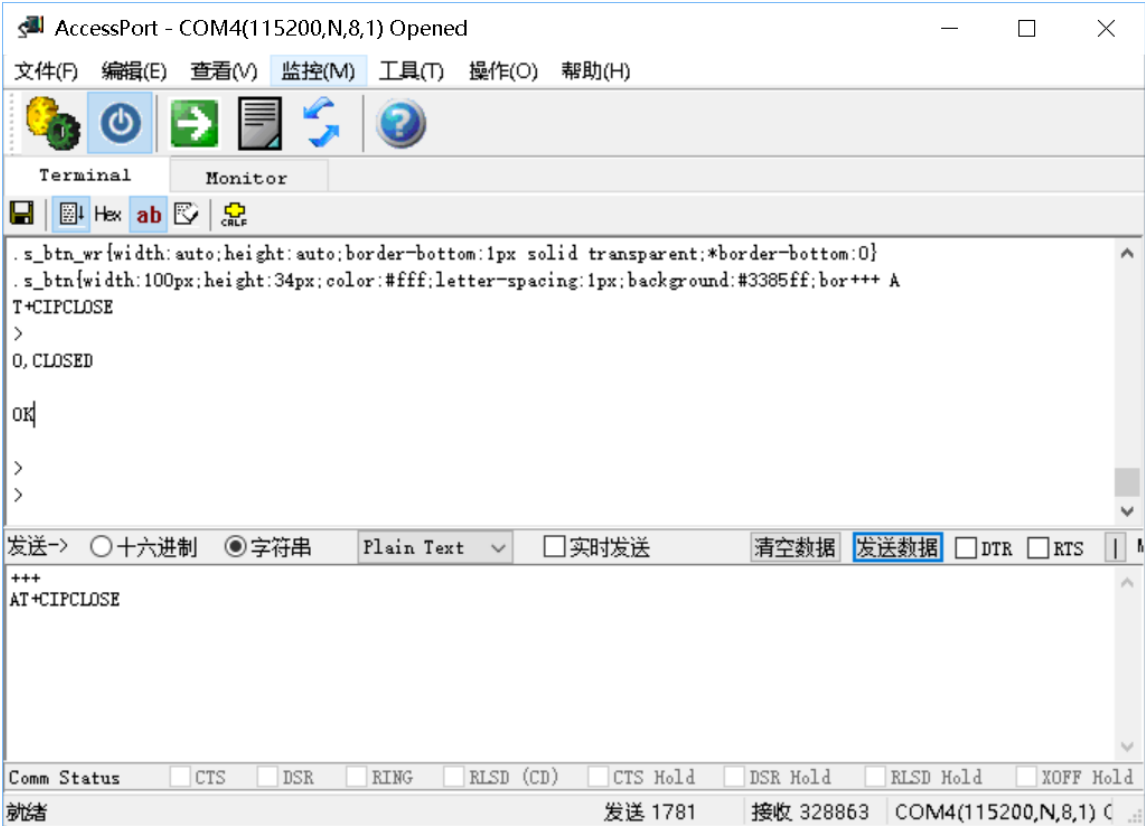
Figure 13: 连续传送 AT 指令举例

```
AT+CIPSTART="TCP","www.baidu.com",80
AT+CIPMODE=1      //開啟透傳模式
AT+CIPSEND
GET /s?wd=arduino HTTP/1.1
Host: www.baidu.com
Connection: close
+++

AT+CIPCLOSE
```

执行结果如 Figure15 所示。

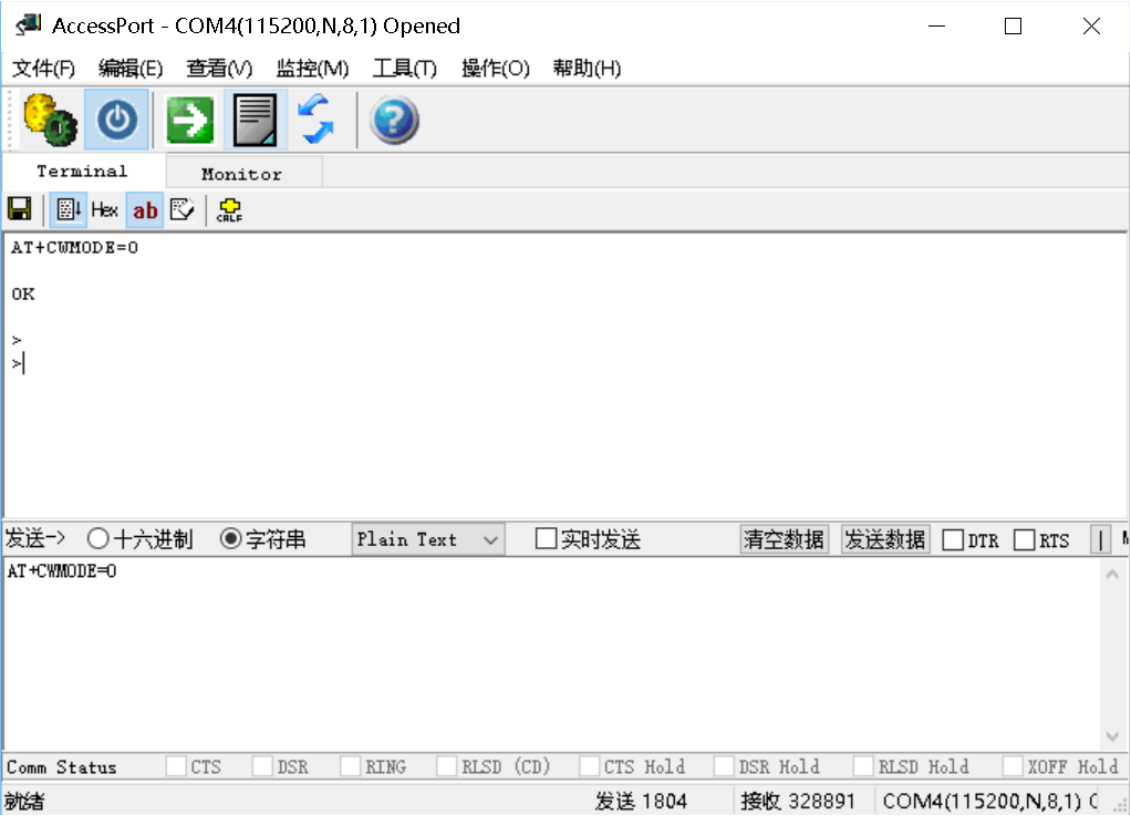
Figure 14: 结束连续传送执行结果



### 3.5. 返回 IDLE 模式

使用 AT+CWMODE=0 回到无 WIFI 模式。执行结果如 Figure16 所示。

Figure 15: 返回 IDLE AT 指令执行结果



## 4. 透传工作 AT 指令举例

本章介绍透传模式下三种工作方式的 AT 指令集合。它们为：

- 1 使用 AT 命令连接 AP，百度网站搜索 “arduino”
- 2 使用蓝牙配网 APP 连接 AP，百度网站搜索 “arduino”
- 3 使用 AT 命令连接 AP，采用连续传送的方式发送 TCP 包，百度网站搜索 “arduino”

### 4.1. AT 命令连接 AP 百度完成搜索举例

AT 指令集合如 Figure 16 所示

Figure 16: AT 命令连接 AP 百度搜索 AT 指令集合

```
at+cwmode=1
at+cwlap
at+cwjap="Opulinks-TEST-AP","1234abcd" // 根据实际情况连接可用的 AP
AT+CIPSTART="TCP","www.baidu.com",80
AT+CIPSEND=70
GET /s?wd=arduino HTTP/1.1
Host: www.baidu.com
Connection: close
```

### 4.2. 蓝牙配网 APP 连接 AP 完成百度搜索举例

AT 指令集合如 Figure 17 所示

Figure 17: 蓝牙配网 APP 连接 AP 百度搜索 AT 指令集合

```
at+cwmode=0
at+rst
at+cwmode=4 // 执行后重启 OPL1000；使用手机 APP 完成蓝牙配网
AT+CIPSTART="TCP","www.baidu.com",80
AT+CIPSEND=70
GET /s?wd=arduino HTTP/1.1
Host: www.baidu.com
Connection: close
```

### 4.3. 连续传送的方式完成百度搜索举例

AT 指令集合如 Figure 18 所示

Figure 18: 连续传送方式百度搜索 AT 指令集合

```
at+cwmode=1
at+cwlap
at+cwjap="Opulinks-TEST-AP","1234abcd" // 根据实际情况连接可用的 AP
AT+CIPSTART="TCP","www.baidu.com",80
AT+CIPMODE=1
AT+CIPSEND
GET /s?wd=arduino HTTP/1.1
Host: www.baidu.com
Connection: close

+++
AT+CIPCLOSE
```

## 5. 模式介绍

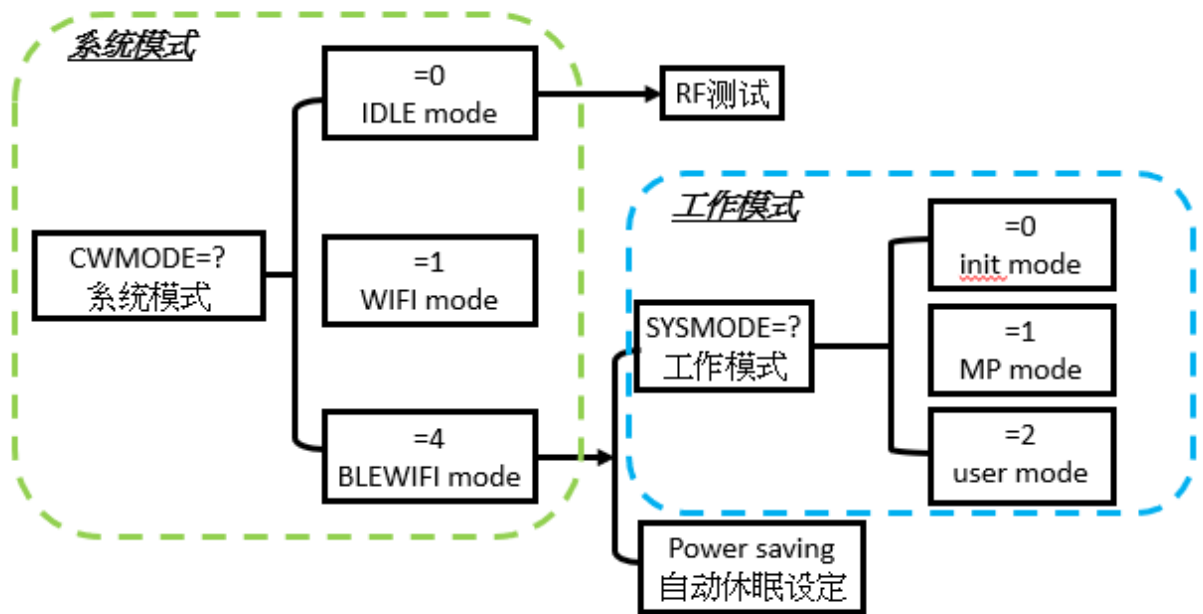


Figure 19: 模式介绍架构图

Note1. 当 BLEWIFI mode (CWMODE=4) 的时候, 系统有两个设定, 皆会保存于 flash

Note2. 两种情况下才能进行 RF 测试:

- 1. 开机在 IDLE mode (CWMODE=0)
- 2. 开机在 BLEWIFI mode (CWMODE=4), 使用 AT+SYSMODE 设定为 MP mode (SYSMODE=1), 重开机后生效。(5-2 章节, 会说明 AT+SYSMODE 指令)

### 5.1. 系统模式(CWMODE)说明与设定

系统模式(CWMODE), 可透过 AT+CWMODE 指令设定及查询。

- 设定指令

AT+CWMODE=<value> //<value>: 0, 1, 4

参数说明: <value>: 0, 1, 4

- 0: IDLE mode

- 1: WIFI mode
- 4: BLEWIFI mode

- 查询指令

AT+CWMODE?

- 现阶段支持的系统模式

Table 1 系统模式介绍

参数设定	系统模式说明
CWMODE = 0	<ul style="list-style-type: none"><li>• <b>IDLE mode</b><ul style="list-style-type: none"><li>• WIFI 尚未初始化，故无法使用 AT command 控制 WIFI</li><li>• 蓝芽 尚未初始化，可以透或 AT+BLEINIT 指令进行初始化后开始控制</li><li>• 此模式 会 保存至 flash，重新启动后为 IDLE mode (CWMODE = 0)</li><li>• 可以从 WIFI mode(CWMODE=1) 或 BLEWIFI mode(CWMODE=4) 使用 AT+CWMODE=0 切换过来 (CWMODE= 1-&gt;0, 4-&gt;0)，并且需要重开机后，才能生效</li></ul></li></ul>
CWMODE = 1	<ul style="list-style-type: none"><li>• <b>WIFI mode</b><ul style="list-style-type: none"><li>• 仅能从 IDLE mode (CWMODE=0) 使用 AT+CWMODE=1 切换过来 (CWMODE = 0 -&gt; 1)，实时生效</li><li>• WiFi 已初始化，可使用 AT comand 控制 WiFi 联机，与 WiFi AP 联机成功后，会将此联机成功的 AP 信息记录在系统的自动联机列表(auto connect list)，此自动联机列表会存放在 flash 内</li><li>• 下列情况，会参考自动联机列表，进行 WiFi 自动联机<ul style="list-style-type: none"><li>a. 系统模式初始化：当启动设备切换到 WIFI MODE (CWMODE=1)</li><li>b. WIFI 断线后处理：若 开启自动联机功能 (AUTOCONN=1) 时，WiFi 会自动重连</li></ul></li><li>• 蓝芽尚未初始化，可以透过 AT+BLEINIT 指令进行初始化后开始控制</li><li>• 此模式不会保存至 flash，重新启动后会恢复成 IDLE mode (CWMODE=0)</li></ul></li></ul>
CWMODE = 4	<ul style="list-style-type: none"><li>• <b>BLEWIFI mode</b><ul style="list-style-type: none"><li>• 仅能从 IDLE mode (CWMODE=0) 使用 AT+CWMODE=4 切换过来 (CWMODE = 0 -&gt; 4)，实时生效</li></ul></li></ul>



	<ul style="list-style-type: none"><li>支持蓝牙配网模式，可以透过 手机 APP 操作完成 WiFi 联机，与 WiFi AP 联机成功后，会将此联机成功的 AP 信息记录在系统的自动联机列表(auto connect list)，此自动联机列表会存放在 flash 内</li><li>下列情况，会参考自动联机列表，进行 WiFi 自动联机<ul style="list-style-type: none"><li>系统模式初始化：当启动设备切换到 BLEWIFI MODE (CWMODE=4)</li><li>WIFI 断线后，将会自动重连</li></ul></li><li>此模式 会 保存至 flash，重新启动后仍在 BLEWIFI MODE (CWMODE=4)</li></ul>
--	---

5.1.1. 支持 CWMODE 切换运用

为避免系统出现不稳定，目前仅支持下列 CWMODE 切换应用。

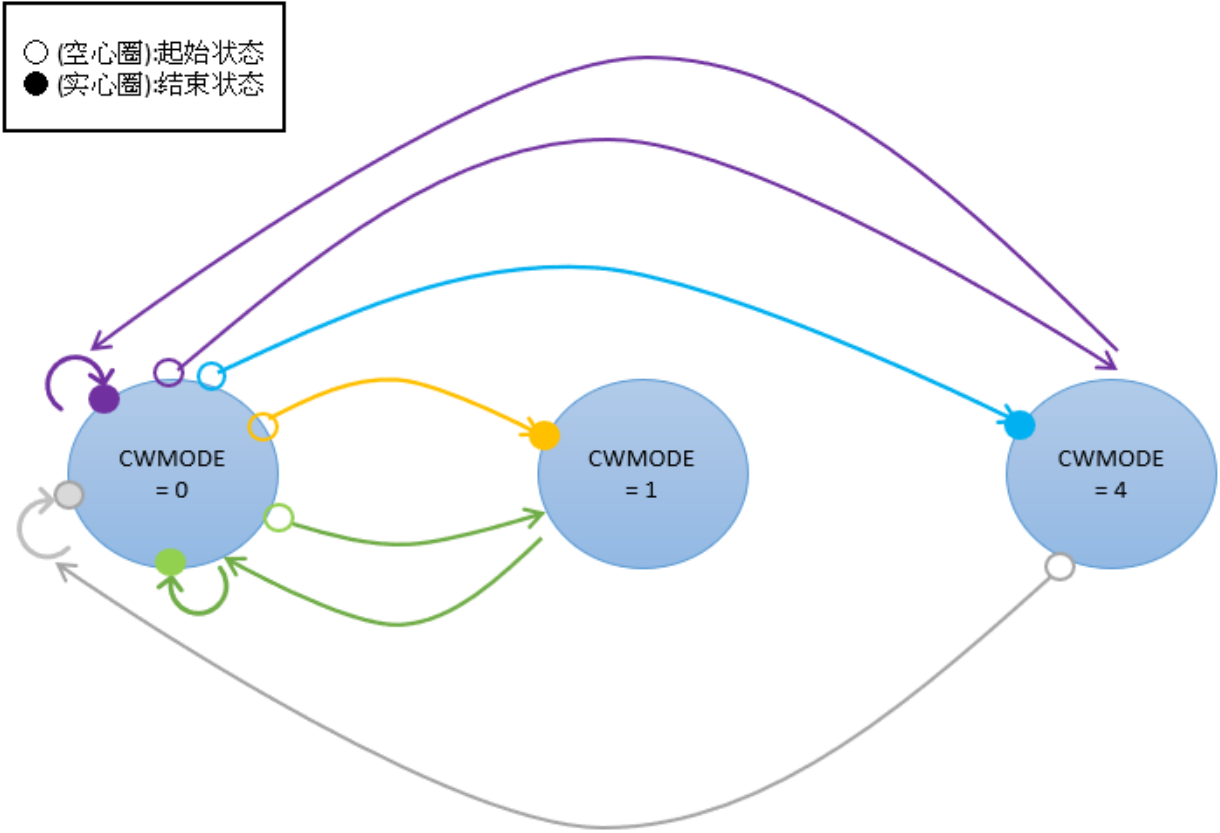


Figure 20: CWMODE 状态机切换示意图

Table 2: 支持 CWMODE 切换

0→1
开机为 IDLE MODE (CWMODE=0)，切换到 WIFI MODE (CWMODE=1)，实时生效。
0→1→0→reboot
开机为 IDLE MODE (CWMODE=0)，切换到 WIFI MODE (CWMODE=1)，再回到 IDLE MODE (CWMODE=0)，重开机生效。
0→4
开机为 IDLE MODE (CWMODE=0)，切换到 BLEWIFI MODE (CWMODE=4)，实时生效。
0→4→0→reboot
开机为 IDLE MODE (CWMODE=0)，切换到 BLEWIFI MODE (CWMODE=4)，再回到 IDLE MODE (CWMODE=0)，重开机生效。
4→0→reboot
开机为 BLEWIFI MODE (CWMODE=4)，切换到 IDLE MODE (CWMODE=0)，重开机生效。

5.1.2. BLEWIF MODE (CWMODE=4) 相关指令支持

Table 3: BLEMODE mod 支持指令

状况	指令	备注
查询 WIFI 联机状况	AT+CWJAP?	
查询 IP address	AT+CIFSR	
查询/设定 WIFI DTIM value	AT+DTIM=<DTIM value> DTIM Time:<DTIM value> <OK ERROR>  - 设定 WIFI DTIM value 在 <i>blewifi_configuration.h</i>  #define BLEWIFI_WIFI_DTIM_INTERVAL (3000) // ms	- 设定后实时生效，此操作不会存于 flash 中，重开机恢复 flash 纪录的值
设定 BLE 广播状态	AT+BLECAST=<enable>[,<time(ms)>] +BLECAST:<enable> <OK ERROR>	

	<div>&lt;enable&gt;</div> <div>0 : 停止广播</div> <div>1 : 开始广播，需要加上&lt;time(ms)&gt;</div> <div>&lt;time(ms)&gt; : 开始广播后，自动停止的时间</div> <div>预设 为 0，范围 0, 1000 ~ 3600000</div> <div>0 : 不自动停止广播</div>	
设定 BLE current 广播参数	<div>AT+BLECASTPARAM=&lt;interval time(ms)&gt;</div> <div>+BLECASTPARAM:&lt;interval time(ms)&gt;</div> <div>&lt;OK ERROR&gt;</div> <div>&lt;interval time(ms)&gt;</div> <div>每次广播的时间间隔 (ms)，范围 32 ~ 10240</div>	<div>- 设定后实时生效，此操作不会存于 flash 中，重开机恢复 flash 纪录的值</div>
查询 BLE 联机状况	<div>AT+BLECONN?</div> <div>+BLECONN:&lt;peer BLE addr&gt;,&lt;addr type&gt;</div> <div>+BLECONN:PEER CONNECTION</div> <div>+BLECONN:PEER DISCONNECTION</div>	

5.2. 工作模式(SYSMODE)

可以透过 AT+SYSMODE 指令设定及查询

- 设定指令

AT+SYSMODE=<value> //<value> : 0, 1, 2

参数说明 : <value> : 0, 1, 2

- 0: Init mode
- 1: MP mode
- 2: User mode

- 查询指令

AT+CWMODE?

-工作模式说明

Table 4: 工作模式介绍

参数设定	工作模式说明
AT+SYSMODE = 0	Init mode, 初始模式。
AT+SYSMODE = 1	MP mode, 工厂模式, 用于工厂进行 MP 测试。
AT+SYSMODE = 2	User mode, 用户模式。

Note, 当 BLEWIFI mode ( CWMODE=4 ) 的时候, 系统设定会保存于 flash 内

5.3. 自动休眠功能(POWER SAVING)

定义 Smart Sleep 功能是 Enable or Disable。

此设定定义在 blewifi\_configuration.h

```
#define BLEWIFI_COM_POWER_SAVE_EN    <value>
```

参数说明 : <value> : 0, 1

- 0: Disable Smart Sleep
- 1: Enable Smart Sleep

Note, 当 BLEWIFI mode ( CWMODE=4 ) 的时候, 系统设定会透过 FIM 的机制保存于 flash 内

- FIM 機制的說明請參考 : [https://github.com/Opulinks-Tech/OPL1000A2-SDK/blob/master/Doc/zh\\_CN/OPL1000-Flash-User-Guide.pdf](https://github.com/Opulinks-Tech/OPL1000A2-SDK/blob/master/Doc/zh_CN/OPL1000-Flash-User-Guide.pdf)

6. 透传模式范例

步骤说明

Table 5: 透传模式步骤说明

STEP1 完成 AP 连接	STEP2 建立 Socket 连接	STEP3 开始透传
使用 WIFI MODE 或 BLEWIFI MODE 连接 AP  case1 / case2 (择一)	建立单点 或 多点联机  case3 / case4 (择一)	不同 Socket 连接方式 搭配不同透传方式  case5/ case6 / case7 (择一)

皆从 IDLE MODE (CWMODE=0) 为起始状态。

6.1. 完成 AP 连接范例-Case1/Case2

- STEP1

联机前，要完成 WIFI 连接 AP 并且 GOT IP

使用 WIFI MODE(CWMODE=1) / BLEWIFI MODE(CWMODE=4) 连接 AP 的方法。

→请参考 Case 1 / Case 2 (择一)

Case 1.

从起始状态为 IDLE MODE (CWMODE=0)，切换到 WIFI MODE(CWMODE=1)，使用 AT command 连接 AP

Figure 21: case1-使用 WIFI mode &AT Command 连接 AP

+CWMODE:0  OK	//开机为 IDLE MODE (CWMODE=0)
---------------------	----------------------------

>>at+cwmode=1 OK	// 进入 WIFI MODE (CWMODE=1) (要从 IDLE MODE [CWMODE=0], 才能进行此切换)
>>at+cwlap? OK	// wifi scan, 会依序列出结果, 格式为 +CWLAP:<channel>,<SSID>,<RSSI>,<BSSID> +CWLAP:4,Xiaomi_opulinks,- 67,ec:41:18:0e:3b:55,1 +CWLAP:4,netis_3B10DA,- 57,00:72:63:3b:10:da,11
>>at+cwjap="netis_3B10DA","12345678" WIFI CONNECTED WIFI GOT IP OK	// wifi connect with correct password // 联机成功并取得 IP, 此 AP 联机信息会记录到自动联机列表
>>at+cwjap? +CWJAP:netis_3B10DA, 00:72:63:3b:10:da,11,-57 OK	// 查询联机上的 AP // 回传联机的 AP 信息, 格式为 +CWJAP:<SSID>,<BSSID>,<channel>,<RSSI>
>>at+cifsr +CIFSR:STAIP,"10.10.10.103" +CIFSR:STAMAC,"00:38:03:76:57:02" OK	// 查询 STA 的 IP 和 MAC address 信息
>>at+wifimaccfg? +WIFIMACCFG:0,30 OK	// 查询 skip DTIM // 格式为 +WIFIMACCFG:<cfg_id>,<value> // <cfg_id> : 0, 表示 skip DTIM 数量, 此结果为 skip 30 个 DTIM
>>at+wifimaccfg=0,40 OK	// 设定 skip DTIM, 格式为 AT+WIFIMACCFG:<cfg_id>,<value> // <cfg_id> : 0, 表示 skip DTIM 数量, 此设定为 skip 40 个 DTIM // <value> 范围 0~255

>>at+wifimaccfg? +WIFIMACCFG:0,40 OK	// 查询 skip DTIM // 格式为 +WIFIMACCFG:<cfg_id>,<value> // <cfg_id> : 0, 表示 skip DTIM 数量, 此 结果为 skip 40 个 DTIM
>>at+cwjap="Xiaomi_opulinks","12345678" WIFI DISCONNECT WIFI CONNECTED WIFI GOT IP OK	// 改连其他 AP // 会先与前一个 AP 断线 // 再与下一个 AP 进行联机 // 联机成功并取得 IP, 此 AP 联机信息会记录到 自动联机列表
>>at+cwjap? +CWJAP:Xiaomi_opulinks, ec:41:18:0e:3b:55,1,-67 OK	// 查询连接上的 AP // 回传连接的 AP 信息
>>at+cifsr +CIFSR:STAIP,"192.168.1.33" +CIFSR:STAMAC,"00:38:03:76:57:02" OK	// 查询 STA 的 IP 和 MAC address 信息
>>at+wifimaccfg? +WIFIMACCFG:0,40 OK	// 查询 skip DTIM, 连上其他 AP , DTIM 设 定没变
>>at+cwqap OK WIFI DISCONNECT	// 断开与 AP 的连接
>>at+rst OK	// 重启设备
>>at+cwmode? +CWMODE:0 OK	// 重启后, 在 IDLE MODE (CWMODE=0)

>>at+cwmode=1 OK WIFI CONNECTED WIFI GOT IP OK	// 切换到 WIFI MODE (CWMODE=1) // 因为自动联机有 AP 信息，设备自动与 AP 做 WiFi 重连，且联机成功取得 IP
>>at+cwjap? +CWJAP:netis_3B10DA,00:72:63:3b:10:da,1 1,-57 OK	// 查询与哪个 AP 联机成功
>>at+cifsr +CIFSR:STAIP,"10.10.10.103" +CIFSR:STAMAC,"00:38:03:76:57:02" OK	// 查询 STA 的 IP 和 MAC address 信息
>>at+wifimaccfg? +WIFIMACCFG:0,40 OK	// 查询 skip DTIM，重开机后，DTIM 依旧是设定值，非默认值

- 可能遇到的其他状况

Figure 22: case1-其他状况

>>at+cwjap? No AP connected OK	// 查询 WiFi connect 状态 // 没有与 AP 联机
>>at+cifsr +CIFSR:STAIP,"0.0.0.0" +CIFSR:STAMAC,"00:38:03:76:57:02" OK	// 查询 STA 的 IP 和 MAC address 信息 // 尚未连上 AP 且取得 IP 前，IP 为 0.0.0.0



>>at+cwjap="netis_3B10DA","12345678" +CWJAP:3 ERROR	// wifi connect // 错误讯息，表示尚未 wifi scan，找不到目标 AP
>>at+cwjap="netis_3B10DA","12341234" +CWJAP:1 ERROR	// wifi connect with wrong password // 错误讯息，表示联机超时
>>at+cwjap="netis_3B10DA","12341234" +CWJAP:2 ERROR	// wifi connect with wrong password // 错误讯息，表示密码错误
>>at+wifimaccfg=1,40 +CWWIFIMACCFG:1 ERROR	// cfg_id = 1，不支援
>>at+wifimaccfg=0,300 +CWWIFIMACCFG:1 ERROR	// value = 300，超过范围 0~255

Case 2.

从起始状态为 IDLE MODE (CWMODE=0)，切换到 BLEWIFI MODE (CWMODE=4)，使用 AT Cmd 查询 + 手机蓝牙配网。

Figure 23: case2-使用 BLEWIFI mode 连接 AP

>at+cwmode? +CWMODE:0 OK	// 开机为 IDLE MODE (CWMODE=0)
>at+cwmode=4 OK	// 进入 BLEWIFI MODE (CWMODE=4) (要从 IDLE MODE [CWMODE=0]，才能进行此切换)  // 此模式会保存至 flash，重新启动会自动执行

<pre>&gt;at+blecast=1 +BLECAST:1,0 OK  WIFI CONNECTED WIFI GOT IP  &gt;at+cwjap? +CWJAP:netis_3B10DA,00:72:63:3b:10:da, 11,-57 OK  &gt;at+cifsr +CIFSR:STAIP,"10.10.10.103" +CIFSR:STAMAC,"00:38:03:76:57:02" OK  &gt;at+dtim? DTIM Time: 3000 OK  &gt;at+dtim=4000 OK  &gt;at+dtim? DTIM Time: 4000 OK  WIFI DISCONNECT</pre>	<pre>// 自动联机列表没有 AP 信息，没有自动进行 WiFi 重连  // 开启蓝芽广播，后续会有详细说明  // 使用手机 APP 进行配网，且配网成功，此 AP 联机信息会记录到自动联机列表  // 查询连接上的 AP // 回传连接的 AP 信息  // 查询 STA 的 IP 和 MAC address 信息  // 查询 DTIM 设定，格式 DTIM Time: &lt;time(ms)&gt; // &lt;time(ms)&gt; : 表示间隔多久收一次 beacon，如 beacon 间隔为 100 ms，skip DTIM 的数量为 30 个  // 设定 DTIM 设定 // 格式 AT+DTIM=&lt;time(ms)&gt; // &lt;time(ms)&gt; : 表示间隔多久收一次 beacon，如 beacon 间隔为 100 ms，skip DTIM 的数量为 40 个  // 查询 DTIM 设定，格式 DTIM Time: &lt;time(ms)&gt; // &lt;time(ms)&gt; : 表示间隔多久收一次 beacon，如 beacon 间隔为 100 ms，skip DTIM 的数量为 40 个  // 与 AP 断线</pre>
--	--

WIFI CONNECTED WIFI GOT IP	// 断线后会与 AP 自动联机
>at+rst OK	// 重启设备
>at+cwmode? +CWMODE:4 OK	// 重启后，依然在 BLEWIFI MODE (CWMODE=4)
WIFI CONNECTED WIFI GOT IP	// 因自动联机列表有 AP 信息，进行自动重连动作
>at+cwjap? +CWJAP:netis_3B10DA,00:72:63:3b:10:da,11,-57 OK	// 查询连接上的 AP // 回传连接的 AP 信息
>at+cifsr +CIFSR:STAIP,"10.10.10.103" +CIFSR:STAMAC,"00:38:03:76:57:02" OK	// 查询 STA 的 IP 和 MAC address 信息
>at+dtim? DTIM Time: 3000 OK	// 查询 DTIM 设定，恢复为 3000 (ms)

- 可能遇到的其他状况

Figure 24: case2-其他状况

>at+cwjap? No AP connected OK	// 查询 WiFi connect 状态 // 没有与 AP 联机
-------------------------------------	---------------------------------------

<pre>&gt;at+cifsr +CIFSR:STAIP,"0.0.0.0" +CIFSR:STAMAC,"00:38:03:76:57:02" OK</pre>	<pre>// 查询 STA 的 IP 和 MAC address 信息 // 尚未连上 AP 且取得 IP 前, IP 为 0.0.0.0</pre>
---	--

6.2. 建立 Socket 连接范例-Case3/Case4

- STEP2. 建立 Socket 连接

使用 TCP / UDP 建立单点或多点联机。

→请参考 [Case 3](#) / [Case 4](#) (择一)

Case3.

建立单点联机 (联机前, 要完成 WIFI 连接 AP 并且 GOT IP, 请参考 STEP1, 以下不再赘述)。

Figure 25: case3-单点联机

<pre>&gt;at+cipmux? +CIPMUX:0 OK</pre>	<pre>// 查询单点或多点连接 // CIPMUX:0, 默认值, 支持单点连接 // CIPMUX:1, 支持多点连接</pre>
<pre>&gt;at+cipstart="TCP","www.google.com",80  CONNECT OK</pre>	<pre>// 建立 TCP socket, //格式为 AT+CIPSTART="&lt;proto&gt;","&lt;domain name or IP&gt;",&lt;port num&gt; // &lt;proto&gt; : TCP or UDP // 联机成功</pre>
<pre>&gt;at+cipstatus? STATUS:3  +CIPSTATUS:0,"TCP","172.217.160.68",80 ,62512,0  OK</pre>	<pre>// 查询已联机 socket 状态 // 显示状态代 3, 设备已建立 TCP 或 UDP 联机 // 格式为 +CIPSTATUS:&lt;link ID&gt;,&lt;type&gt;,&lt;remote IP&gt;,&lt;remote port&gt;,&lt;local port&gt;,&lt;tetype&gt; // &lt;tetype&gt;=0, 设备为 Client // &lt;tetype&gt;=1, 设备为 Server</pre>

<pre>&gt;at+cipclose 0,CLOSED OK  /////** 建立 UDP socket **///// //&gt;at+cipstart="UDP","180.76.76.76",53 // CONNECT // OK //&gt;at+cipstatus? // STATUS:3 //+CIPSTATUS:0,"UDP","180.76.76.76",53,62512,0 // OK  //&gt;at+cipclose // 0,CLOSED // OK</pre>	<pre>// 关闭联机的 socket  // 建立 UDP socket // 联机成功 // 查询已联机 socket 状态 // 显示状态代 3，设备已建立 TCP 或 UDP 联机 // 格式为 +CIPSTATUS:&lt;link ID&gt;,&lt;type&gt;,&lt;remote IP&gt;,&lt;remote port&gt;,&lt;local port&gt;,&lt;tetype&gt; // &lt;tetype&gt;=0，设备为 Client // &lt;tetype&gt;=1，设备为 Server // 关闭 socket</pre>
--	---

- 可能遇到的其他状况

Figure 26: case3-其他状况

<pre>&gt;at+cipstart="TCP","www.google.com",80 DNS Fail CONNECT FAIL ERROR</pre>	<pre>// 未重连成功前，进行 TCP socket 联机 // 无法解析 DNS // 建立 TCP socket 联机失败</pre>
<pre>&gt;at+cipstatus? STATUS:5 OK</pre>	<pre>// 查询已联机 socket 状态 // 显示状态代码 5，设备未连接 AP</pre>
<pre>&gt;at+cipstatus? STATUS:2 OK</pre>	<pre>// 查询已联机 socket 状态 // 显示状态代码 2，设备已连接 AP，获得 IP address</pre>

>at+cipstart="UDP","8.8.8.8",53 ALREAY CONNECT ERROR	// 已经联机，不能联机到其他的地方
>at+cipstatus? STATUS:4 OK	// 查询已联机 socket 状态 // 显示状态代码 4，设备断开网络连接

Case4.

建立多点联机 (联机前，要完成 WIFI 连接 AP 并且 GOT IP，请参考 STEP1，以下不再赘述)。

Figure 27: case4-多点联机

>at+cipmux? +CIPMUX:0 OK	// 查询单点或多点连接 // CIPMUX:0，默认值，支持单点连接 // CIPMUX:1，支持多点连接
>at+cipmux= 1 OK	// 设定为多点连接 (CIPMUX=1)
>at+cipmux? +CIPMUX:1 OK	// 查询单点或多点连接 // 已切换为支持多点连接(CIPMUX=1)
>at+cipstart=0,"TCP","www.chinabiz.org .tw",80 0,CONNECT OK  /** TCP/UDP 可以混用 **/ /** Iink ID 可为 0, 1, 2, 3, 4 **/ >at+cipstart=1,"UDP","180.76.76.76",53 1,CONNECT	// 建立 TCP/UDP socket, //注意格式与单点连接 (CIPMUX=0) 不同。 AT+CIPSTART=<link ID>,<socket type>,<IP or domain>,<port> // <link ID> : 0~4  // TCP/UDP 可以混用

<pre>OK  &gt;at+cipstart=2,"TCP","people.com.cn",8 0     2,CONNECT     OK  &gt;at+cipstart=3,"UDP","114.114.114.114" ,53     3,CONNECT     OK  &gt;at+cipstart=4,"TCP","www.cccdevelopment.com",80     4,CONNECT     OK  &gt;at+cipstatus?     STATUS:3     +CIPSTATUS:0,"TCP","202.39.150.7", 80,57423,0     +CIPSTATUS:1,"UDP","180.76.76.76", 53,57424,0     +CIPSTATUS:2,"TCP","209.177.81.229 ",80,57425,0     +CIPSTATUS:3,"UDP","114.114.114.11 4",53,57426,0     +CIPSTATUS:4,"TCP","123.51.184.122 ",80,57427,0     OK  &gt;at+cipclose=0     0,CLOSED     OK</pre>	
	<pre>// 查询已联机 socket 状态</pre>
	<pre>// 关闭 socket, 需指定&lt;link ID&gt;, 注意格式 与单点连接 (CIPMUX=0) 不同 // AT+CIPCLOSE=&lt;link ID&gt; // &lt;link ID&gt; : 0~5 // 5 : 同时关闭所有 socket (link ID:0~4)</pre>

<pre>&gt;at+cipclose=1 1,CLOSED OK</pre>	<pre>// 关闭 socket</pre>
<pre>&gt;at+cipstatus? STATUS:3 +CIPSTATUS:2,"TCP","209.177.81.229",80,57425,0 +CIPSTATUS:3,"UDP","114.114.114.114",53,57426,0 +CIPSTATUS:4,"TCP","123.51.184.122",80,57427,0 OK</pre>	<pre>// 查询已联机 socket 状态</pre>
<pre>&gt;at+cipstart=0,"UDP","223.6.6.6",53 0,CONNECT OK</pre>	<pre>// link ID = 0 已经先关闭，可以建立新的 UDP socket</pre>
<pre>&gt;at+cipstatus? STATUS:3 +CIPSTATUS:0,"UDP","223.6.6.6",53,58360,0 +CIPSTATUS:2,"TCP","209.177.81.229",80,57425,0 +CIPSTATUS:3,"UDP","114.114.114.114",53,57426,0 +CIPSTATUS:4,"TCP","123.51.184.122",80,57427,0 OK</pre>	<pre>// 查询已联机 socket 状态</pre>
<pre>&gt;at+cipclose=5 0,CLOSED 2,CLOSED 3,CLOSED 4,CLOSED OK</pre>	<pre>// 同时关闭所有 socket (link ID:0~4)</pre>



<pre>&gt;at+cipstatus?  STATUS:4  OK</pre>	<pre>// 查询已联机 socket 状态</pre>
--	-------------------------------

-可能遇到的其他状况

Figure 28: case4-其他状况

<pre>&gt;at+cipstart="UDP","180.76.76.76",53 Link type ERROR ERROR</pre>	<pre>// 使用单点连接(CIPMUX=0)指令格式做联机 时, 因为格式不同, 会显示 parser 错误</pre>
<pre>&gt;at+cipstart=5,"UDP","223.6.6.6",53 ID ERROR ERROR</pre>	<pre>// link ID = 5 超出范围出现 ID ERROR</pre>
<pre>&gt;at+cipstart=0,"UDP","223.6.6.6",53 ALREAY CONNECT ERROR</pre>	<pre>// 已经联机的不能再拿来使用</pre>
<pre>&gt;at+cipclose=6 ERROR</pre>	<pre>// link ID 超出范围, 显示 ERROR</pre>
<pre>&gt;at+cipclose MUX=1 ERROR</pre>	<pre>// 使用单点模式(CIPMUX=0)的指令格式会出现 ERROR  // 告知现在为多点连接模式(CIPMUX=1)</pre>

6.3. 开始透传范例-Case5/Case6/Case7

- STEP3. 透传资料

建立连接后, 透传数据。不同透传数据, 搭配不同连接方式。

→请参考 Case5 / Case6 / Case7 (择一), 皆为独立

Case5.

建立单点联机(CIPMUX=0)后，进行一般透传(CIPMODE=0)。(建立单点模式请参考 case3)  
(透传数据前，请先确认以连接 IP(case1 or 2)，且建立 Socket 连接完成(case3 or 4)，以下不再赘述)。

Figure 29: case5-单点联机+一般透传

<pre>&gt;at+cipmux? +CIPMUX:0 OK</pre>	<pre>// 设定为单点联机(CIPMUX=0)</pre>
<pre>&gt;at+cipstatus? STATUS:3 +CIPSTATUS:0,"TCP","202.39.150.7",80,57423,0 OK</pre>	<pre>// 查询状态: Socket 已联机 // 显示状态代 3, 设备已建立 TCP 或 UDP 联机</pre>
<pre>&gt;at+cipmode? +CIPMODE:0 OK</pre>	<pre>// 查询透传模式 // +CIPMODE=0 : 一般透传模式 // +CIPMODE=1 : 进阶透传模式, 透传模式可以连续传送数据直到输入+++ (20ms 内) 为止</pre>
	<pre>// 下面的 TX / RX 属于独立事件发生</pre>
	<pre>// 当收到对方资料会 show 出来 // 格式为 - +IPD,&lt;recv bytes&gt;:&lt;recv content&gt;</pre>
<pre>+IPD,5:Hello +IPD,12:I am server.</pre>	<pre>// ***** 收到资料(1) ***** // ***** 收到资料(2) *****</pre>
	<pre>// ***** 传送开始(1) *****</pre>
<pre>&gt;at+cipsend=20 OK</pre>	<pre>// 先设定传送数据长度, 传送指令格式为 AT+CIPSEND=&lt;send bytes&gt; // 此为预传送 20 bytes 给 Server</pre>

> Hello, I am client.	// 输入要传送数据内容，长度同 at+cipsend 设定的 bytes 数，未达 bytes 数会一直等待到数量正确为止 // 注意：换行也算是一个字符 (byte) !!!
Recv 20 bytes SEND OK	// system 收到从 AT 串口输入的 20 bytes // 传送成功
+IPD,17:Nice to meet you.	// ***** 传送结束(1) *****
	// ***** 收到资料(3) *****
	// ***** 传送开始(2) *****
>at+cipsend=23 OK	// 先设定传送数据长度
> Nice to meet you,too Recv 23 bytes SEND OK	// 输入与长度相同的数据内容
	// ***** 传送结束(2) *****
	// ***** 传送开始(3) *****
>at+cipsend=9 OK	// 先设定传送数据长度
> Goodbye.. Recv 9 bytes SEND OK	// 输入与长度相同的数据内容
	// ***** 传送结束(3) *****
+IPD,8:Goodbye.	// ***** 收到资料(4) *****

>at+cipclose 0,CLOSED	// 关闭 socket
--------------------------	--------------

-可能遇到的其他状况

Figure 30: case5-其他状况

>at+cipsend=74 OK > GET /search?q=arduino HTTP/1.1 Host: <a href="http://www.google.de">www.google.de</a> Connection: close	// 当进行 HTTP protocol "请求-应答" 的过程中  // HTTP Header 的 Connection 用来告知对方, 此次应答后, Socket 是否启用 Keep-Alive  // HTTP 1.0 中默认是关闭的, 需要在 HTTP Header 加入 "Connection: Keep-Alive", 才能启用 Keep-Alive  // HTTP 1.1 中默认启用 Keep-Alive, 如果加入"Connection: close", 才关闭。  // Header 的 Connection 为 close 时, 告知对方应答后关闭 Keep-Alive
Recv 74 bytes SEND OK	
+IPD,1024:HTTP/1.1 200 OK +IPD,1024:... +IPD,1024:... +IPD,128:... CLOSED	// 从对方收完响应后, 由对方触发 Socket 断线
>at+cipsend=10 link is not connected ERROR	// 传送 10 bytes , 传送指令格式为 AT+CIPSEND=<send bytes>  // 显示 ERROR 讯息, 表示 socket link 还没有连接

Case6.

建立多点联机(CIPMUX=1)后, 进行一般透传(CIPMODE=0)。(建立多点模式请参考 case4)  
(透传数据前, 请先确认以连接 IP(case1/2), 且建立 Socket 连接完成(case3/4), 以下不再赘述)。

Figure 31: case6-多点联机+一般透传

<pre>&gt;at+cipmux? +CIPMUX:1 OK</pre>	<pre>// 设定为多点联机(CIPMUX=1)</pre>
<pre>at+cipstatus? STATUS:3 +CIPSTATUS:0,"TCP","202.39.150.7",80,57423,0 +CIPSTATUS:1,"UDP","180.76.76.76",53,57424,0 +CIPSTATUS:2,"TCP","103.235.46.39",80,57425,0 +CIPSTATUS:3,"UDP","114.114.114.114",53,57426,0 +CIPSTATUS:4,"TCP","123.51.184.122",80,57427,0  OK</pre>	<pre>// Socket 已联机</pre>
<pre>&gt;at+cipmode? +CIPMODE:0 OK</pre>	<pre>// 查询透传模式 // +CIPMODE=0 : 一般透传模式 // +CIPMODE=1 : 进阶透传模式, 透传模式可以连续传送数据直到输入+++ (20ms 内) 为止</pre>
	<pre>// 下面的 TX / RX 属于独立事件发生, 并没有顺序之分</pre>
	<pre>// 当收到对方数据会 show 出来, 格式为 +IPD,&lt;link_ID&gt;,&lt;recv_bytes&gt;:&lt;recv_content&gt;</pre>
<pre>+IPD,1,24:Hello, this is server 1.  +IPD,3,24:Hello, this is server 3.</pre>	<pre>// ***** 收到数据(1) ***** 从 Link ID = 1 传送过来</pre>

<pre>&gt;at+cipsend=0,10  &gt;1234567890 Recv 10 bytes SEND OK  +IPD,0,23:What are you doing now.  &gt;at+cipsend=2,15 OK  &gt;this is client. Recv 15 bytes SEND OK  &gt;at+cipclose=5 0,CLOSED 1,CLOSED 2,CLOSED 3,CLOSED 4,CLOSED OK</pre>	// ***** 收到数据(2) ***** 从 Link ID = 3 传送过来
	// ***** 传送开始(1) *****
	// 先设定指定的 LINK ID 及 传送数据长度, 多点连接(CIPMUX=1)需指定 LINK ID, 格式为 AT+CIPSEND=<link id>,<send bytes> // 此为送 10 bytes 给 LINK ID = 0
	// 输入与长度相同的数据内容
	// ***** 传送结束(3) *****
	// ***** 收到数据(3) ***** 从 Link ID = 0 传送过来
	// ***** 传送开始(2) *****
	// 先设定指定的 LINK ID 及 传送数据长度
	// 输入与长度相同的数据内容
	// ***** 传送结束(2) *****
	// 同时关闭所有 socket (link ID:0~4)

- 可能遇到的其他状况

Figure 32: case6-其他状况

<pre>&gt;at+cipsend=0,10 link is not connected ERROR</pre>	// 显示 ERROR 讯息，表示 socket link 还没有连接
<pre>&gt;at+cipsend=10 ERROR</pre>	// 使用单点链接指令，直接显示错误

Case7.

建立单点联机(CIPMUX=0)后，进行进阶透传(CIPMODE=1)。(建立单点模式请参考 case3)

(透传数据前，请先确认以连接 IP(case1/2)，且建立 Socket 连接完成(case3/4)，以下不再赘述)

Figure 33: case7-单点联机+进阶透传

<pre>&gt;at+cipmux? +CIPMUX:0 OK</pre>	// 现在设定为单点联机 (CIPMUX=0)
<pre>&gt;at+cipstatus? STATUS:3 +CIPSTATUS:0,"TCP","202.39.150.7",80,5 7423,0 OK</pre>	// 查询已联机 socket 状态
<pre>&gt;at+cipmode? +CIPMODE:0 OK</pre>	// 查询透传模式 // +CIPMODE=0 : 一般透传模式 // +CIPMODE=1 : 进阶透传模式，透传模式可以连续传送数据直到输入+++ (20ms 内) 为止
<pre>&gt;at+cipmode=1 OK</pre>	// 设定为 进阶透传模式
<pre>&gt;at+cipmode? +CIPMODE:0</pre>	// 查询：已设定为 进阶透传模式

OK	
>at+cipsend	// 进行连续传送，在 AT 串口输入透传的封包内容，透传模式下的指令格式 AT+CIPSEND
OK	// 在透传模式下， 20ms 输入的内容才会以一个封包传出去
	// ***** 传送开始(1) *****
/>>Hello, I am Client	//输入的内容不会显示在画面上 // 直接输入传送数据
	// ***** 传送结束(1) *****
Hello, I am Server.	// ***** 收到数据(1) ***** 会显示在画面上
Please login with account and password	// ***** 收到数据(2) ***** 会显示在画面上
	// ***** 传送开始(2) *****
/>>I don't account and password	//输入的内容不会显示在画面上
/>>byebye	
	// ***** 传送结束(2) *****
/>>+++	// +++ 不会显示在画面上 // 结束进阶透传，+++ 20ms 之内输入完毕（手敲键盘来不及，一定要用 copy and paste）
>at+cipclose	// 关闭 TCP 联机
0,CLOSED	
OK	

- 可能遇到的其他状况

Figure 34: case7-其他状况



<code>&gt;at+cipmode=2</code> <code>ERROR</code>	// 不在预期内的 value, 显示 ERROR
<code>&gt;at+cipsend</code> <code>ERROR</code>	// 显示 ERROR 讯息, 表示 socket link 还没有连接

7. BLE 控制范例

- 范例为不同的系统模式进行 BLE 控制。
- 初始化 BLE 并进行广播、停止广播及修改广播参数
  - 查询 BLE 连接状态，中断已联机的 BLE PEER

Case 1. 系统模式为 IDLE MODE (CWMODE=0)或 WIFI mode (CWMODE=1)，初始化 BLE 进行 BLE 控制

Figure 35: case1-IDLE mode /WIFI mode 进行 BLE 控制

<pre>&gt;at+cwmode? +CWMODE:0 OK  // &gt;at+cwmode=1 // OK  &gt;at+bleinit? +BLEINIT:0 OK  &gt;at+bleinit=2 OK  &gt;at+bleaddr? +BLEADDR:"01:57:76:03:38:00" OK</pre>	<pre>// 开机为 IDLE MODE (CWMODE=0) // 可于此系统模式初始化 BLE 进行 BLE 控制  // 或于 WIFI mode 初始化 BLE 进行 BLE 控制，切换到 WIFI mode (CWMODE=1)  // BLE 初始化状态，初始化后的状态不会记录在 Flash，故每次开机都需要重新初始化 // +BLEINIT:0 : 尚未初始化 // +BLEINIT:1 : 已初始化为 Client // +BLEINIT:2 : 已初始化为 Client + Server  // 初始化 BLE 为 Client + Server // AT+BLEINIT=1 : Client // AT+BLEINIT=2 : Client + Server  // 查询 BLE MAC Address 时，与实际生效后的 BLE MAC Address 看到是反向的。 // 显示格式为 +BLEADDR:&lt;BLE PUB ADDR&gt; // 实际生效后的 BLE MAC Address 为 "00:38:03:75:67:01"</pre>
---	---

<pre>&gt;at+bleaddr=0,"66:57:76:03:38:00" OK</pre>	<pre>// 设置 BLE MAC 时，与实际生效后的 BLE MAC 看到是反向的。实际生效后的 BLE MAC address 为 "00:38:03:75:67:66"  // 设置格式为 AT+BLEADDR=&lt;addr type&gt;,&lt;ble mac address&gt;  // &lt;addr type&gt; = 0 : Public MAC address  // &lt;addr type&gt; = 1 : Random MAC address</pre>
<pre>&gt;at+rst OK</pre>	<pre>// 需要重新启动让 BLE MAC Address 生效</pre>
<pre>&gt;at+cwmode? +CWMODE:0 OK</pre>	<pre>// 开机为 IDLE MODE (CWMODE=0) // 可于此系统模式初始化 BLE 进行 BLE 控制</pre>
<pre>// &gt;at+cwmode=1 // OK</pre>	<pre>// 或于 WIFI mode 初始化 BLE 进行 BLE 控 制，切换到 WIFI mode (CWMODE=1)</pre>
<pre>&gt;at+bleinit? +BLEINIT:0 OK</pre>	<pre>// 重启后，BLE 恢复为尚未初始化，需重新初 始化</pre>
<pre>&gt;at+bleinit=2 OK</pre>	<pre>// 初始化 BLE 为 Client + Server</pre>
<pre>&gt;at+bleadvstart OK</pre>	<pre>// BLE 开始广播</pre>
<pre>&gt;at+bleadvstop OK</pre>	<pre>// BLE 停止广播</pre>
<pre>&gt;at+bleadvstart OK +BLECONN:0,"4B:20:BE:8B:E9:62",1 +BLECONNPARAM:0,0 +BLECONNPARAM:0,0</pre>	<pre>// BLE 开始广播  // 当有 Peer 连上后，BLE 就自动关闭广播</pre>

<pre>&gt;at+bleconnparam? OK</pre>	<pre>// 查询连上的 Peer 信息 // 格式为 +BLECONNPARAM=&lt;conn index&gt;,&lt;current interval&gt;,&lt;latency&gt;,&lt;timeout&gt; +BLECONNPARAM:0,36,0,2000</pre>
<pre>&gt;at+bledisconn=0 OK +BLEDISCONN:0,"4B:20:BE:8B:E9:62"</pre>	<pre>// 对 Peer 断线, 格式为 AT+BLEDISCONN=&lt;conn index&gt;  // 显示与 client 断线成功, 格式为 +BLEDISCONN:&lt;conn index&gt;,&lt;Peer BLE addr&gt;</pre>
<pre>&gt;at+bleadvparam? +BLEADVPARAM:32,32,0,0,7,0,0,"00:00: 00:00:00:00" OK</pre>	<pre>// 查询广播参数, 格式为 +BLEADVPARAM:&lt;adv_int_min&gt;,&lt;adv_int_ma x&gt;,&lt;adv_type&gt;,&lt;own_addr_type&gt;,&lt;channel _map&gt;,&lt;adv_filter_policy&gt;,&lt;peer_addr_t ype&gt;,&lt;peer_addr&gt;</pre>
<pre>&gt;at+bleadvparam=100,100,0,0,4 OK</pre>	<pre>// 设定广播参数, 格式为 AT+BLEADVPARAM=&lt;adv_int_min&gt;,&lt;adv_int_ max&gt;,&lt;adv_type&gt;,&lt;own_addr_type&gt;,&lt;chann el_map&gt;</pre>
<pre>&gt;at+bleadvparam? +BLEADVPARAM:100,100,0,0,4,0,0,"00: 00:00:00:00:00" OK</pre>	<pre>// 查询是否设定成功广播参数</pre>

- 可能遇到的其他状况

Figure 36: case1-其他状况

<pre>&gt;at+bleaddr? +BLEADDR:1 ERROR</pre>	<pre>// 查询 BLE MAC Address // 尚未初始化, Show ERROR</pre>
<pre>&gt;at+bleadvstart +BLEADVSTART:1 ERROR</pre>	<pre>// BLE 开始广播 // 尚未初始化, Show ERROR</pre>

<pre>&gt;at+bleadvstop +BLEADVSTOP:1 ERROR</pre>	<pre>// BLE 停止广播 // 尚未初始化, Show ERROR</pre>
--	---

Case 2. 系统模式为 BLEWIFI MODE (CWMODE=4), 进行 BLE 控制

Figure 37: case2-BLEWIFI mode 进行 BLE 控制

<pre>&gt;at+cwmode? +CWMODE:4 OK</pre>	<pre>// 已设定为 BLEWIFI MODE (CWMODE=4)</pre>
<pre>&gt;at+blecastparam? +BLECASTPARAM:100 OK</pre>	<pre>// 查询广播参数, 格式为 +BLECASTPARAM:&lt;interval time(ms)&gt; // 每次广播间隔 100 ms</pre>
<pre>&gt;at+blecastparam=10000 +BLECASTPARAM:10000 OK</pre>	<pre>// 设定广播参数, 格式为 AT+BLECASTPARAM=&lt;interval time(ms)&gt;, &lt;interval time(ms)&gt; 范围 32 ~ 10240 // 每次广播间隔 10000 ms ( 10 s )</pre>
<pre>&gt;at+blecast=1,1800000 +BLECAST:1,1800000 OK</pre>	<pre>// 设定开始广播, 30 min (1800000 ms) 自动 停 止 , 格 式 为 AT+BLECAST=&lt;enable&gt;[,&lt;timeout(ms)&gt;] // &lt;enable&gt; // 0 : 停止广播 // 1 : 开始广播, 需要加上&lt;time(ms)&gt; // &lt;timeout(ms)&gt; : 时间到停止广播, 预设为 0, 范围 0, 1000 ~ 3600000 // 0 : 维持广播, 直到输入关闭指令 AT+BLECAST=0</pre>
<pre>+BLECAST:BLE ENTER ADVERTISING +BLECAST:BLE EXIT ADVERTISING</pre>	<pre>// 开始广播消息 // 经过 30 min (1800000 ms) 后自动关闭广播</pre>
<pre>&gt;at+blecast=1</pre>	<pre>// 设定开始广播</pre>

<div>+BLECAST:1</div> <div>OK</div>	
<div>+BLECAST:BLE ENTER ADVERTISING</div> <div>+BLECONN:PEER CONNECTION</div>	<div>// 开始广播消息</div> <div>// 当有 Peer 连上，显示讯息</div>
<div>&gt;at+bleconn?</div> <div>+BLECONN:"00:01:02:03:04:05",1</div>	<div>// 查询 BLE 联机状态，格式为</div> <div>+BLECONN:&lt;peer BLE addr&gt;,&lt;addr type&gt;</div> <div>// &lt;addr_type&gt; = 0 : public address</div> <div>// &lt;addr_type&gt; = 1 : random address</div>
<div>+BLEDISCONN:PEER DISCONNECTION</div>	<div>// Peer 断线后，显示讯息</div>
<div>&gt;at+blecast=0</div> <div>+BLECAST:0</div> <div>OK</div> <div>+BLECAST:BLE EXIT ADVERTISING</div>	<div>// 设定停止广播</div>

## CONTACT

[sales@Opulinks.com](mailto:sales@Opulinks.com)