

# GL-S10 BLE-Gateway User Guide CN

版本	更改项	作者	日期
1.0	第一版	何丰	2020.04.17

## 1. 简介

BLE-Gateway 是一个基于 C 语言开发的蓝牙网关例程，使用的硬件平台是 GL-S10。它实现了收发 BLE 数据并使用 MQTT 协议上传至互联网，打通了 ble 终端设备与云平台之间的数据通道（系统框架图如图 1）。BLE 支持 GAP 和 GATT 规范，即可作为 master 也可作为 slave，用以与其他蓝牙设备连接及交换数据。该固件可以使用 WiFi 作为 station 连接上级路由，也可以通过 wan 口网线上网。它内置了标准的 MQTT 客户端，可以连接至任意标准的 MQTT 服务器。

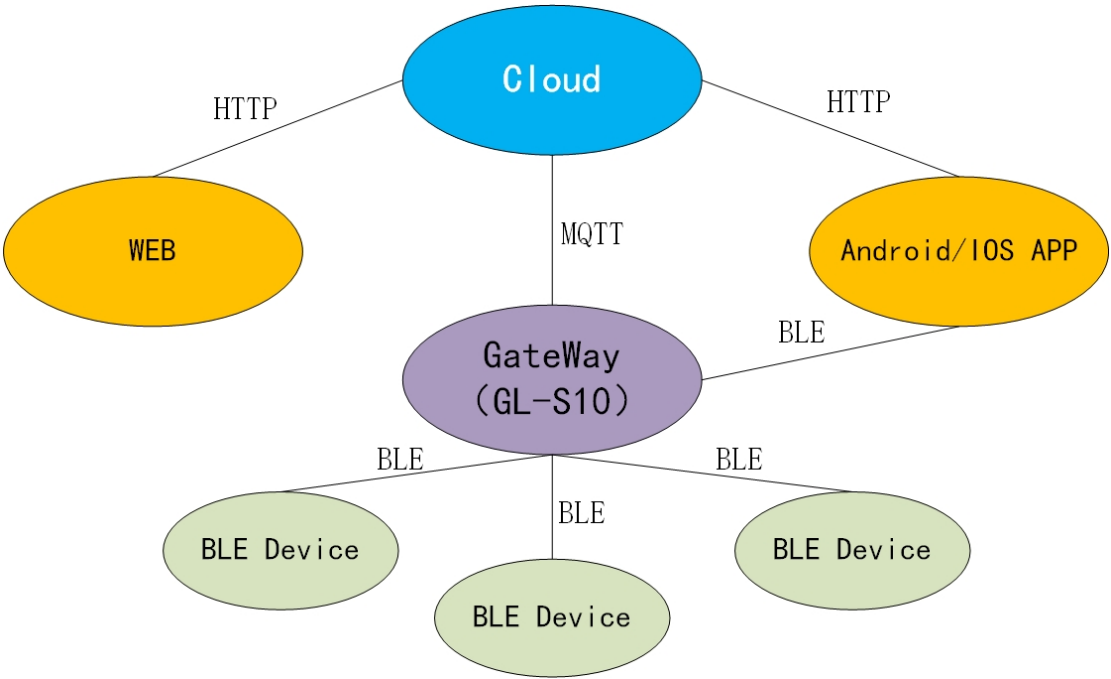


图 0-1.1 系统通信框架

## 2. 功能清单

### 2.1 ble

本例程中 ble 既可作为 master，也可同时作为 slave。

作为 master： 1. 支持发起扫描，扫描结果可传出（见 [5 BLE 扫描并上传至云平台](#)）；  
2. 支持主动发起 ble 连接。

作为 slave： 1. 支持发出广播包；  
2. 支持被动 ble 连接，可以使用 ble notify。

### 2.2 WiFi

本例程中的 WiFi 只能作为 STA，加入上级路由的 WiFi 网络。用户可使用其他的 ble 设备通过 ble API 配置 WiFi 参数（见 [附录 1 – GL-S10 BLE API](#)），也可配套的 APP 直接设置。

### 2.3 网口

本例程可以通过网口连接互联网，但网口连接与 WiFi 连接是互斥的，只能二选一。在连接了网口之后 WiFi 会自动断开；当网口连接断开，WiFi 会自动连接。

### 2.4 LED

GL-S10 共有三个 LED 灯（从右往左分别是 led1、led2、led3），本例程中 LED 灯的显示逻辑见下表：

	常态	按键按住 3-8S 内	按键按住超过 8S	OTA 升级过程中
LED1 电源灯（绿色）	通电亮，断电灭	慢闪	快闪	流水灯
LED2 蓝牙灯（绿色）	有连接时亮， 无连接时灭			
LED3 网络灯（白色）	wifi 连接时闪烁，网线连接时常亮，不联网时灭			

### 2.5 按键

GL-S10 只有一个 reset 按键，本例程中它的功能定义如下：

	按住 3-8S 后松开	按住 8S 以上再松开
--	-------------	-------------

按键功能	清除当前 WiFi 及 MQTT 配置， 但不会断开当前连接	OTA 升级
------	-----------------------------------	--------

## 2.6 MQTT

本例程中内置了一个标准的 MQTT Client。用户可以通过 ble API（见 [GL-S10 BLE API](#)）设置 MQTT 的配置，连上任意的 MQTT 服务器订阅收发数据。相关的使用样例见 [4 MQTT 收发测试](#)

注：本例程只支持基于非加密 tcp 连接的 MQTT

## 2.7 OTA

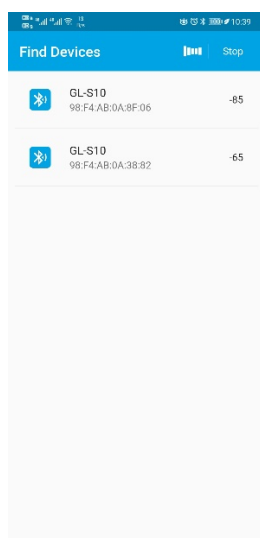
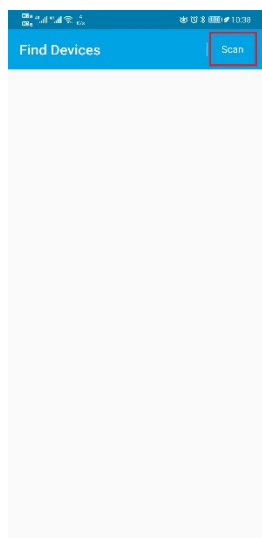
本例程支持 OTA 升级固件，固件中内置了指向本例程的 github 目录下 release 的链接路径。用户可以直接使用按键触发升级（见 [2.5 按键](#)），也可以使用 ble API 修改 OTA 的链接路径（见 [附录 1 – GL-S10 BLE API](#)），使用自己搭建的 http 服务器升级成指定的固件。

## 3 配置 APP 使用指南

本例程有配套开源的配置 APP，安卓用户可以直接在 XXX 下载 apk 直接安装，ios 用户可以下载源码自行编译调试。

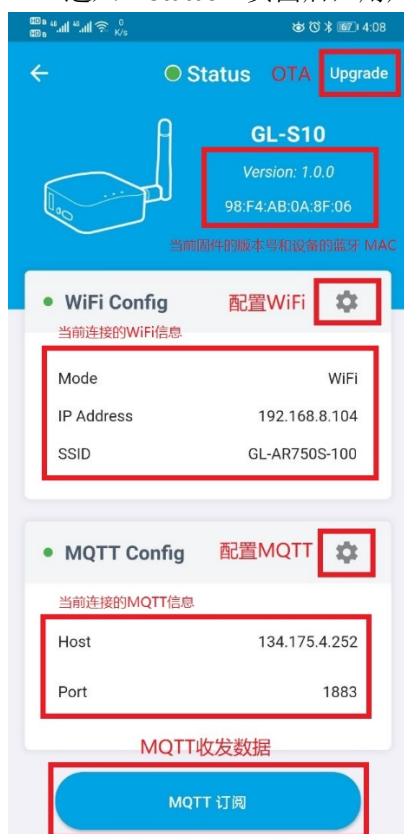
### 3.1 扫描并连接 GL-S10

打开配置 APP，点击右上角的 scan 按钮（如下图）。扫描结果已过滤非 GL-S10 的设备（如下图），如果当前附件有多台 GL-S10，用户可根据外壳的标签二维码获取 mac 地址，从而选定连接哪一台 GL-S10。



## 3.2 查看当前设备信息

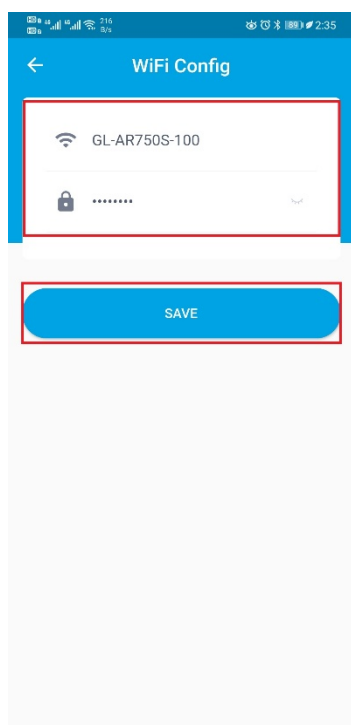
进入“Status”页面后，用户通过下拉刷新，可以获取当前设备的配置信息（如下图）。



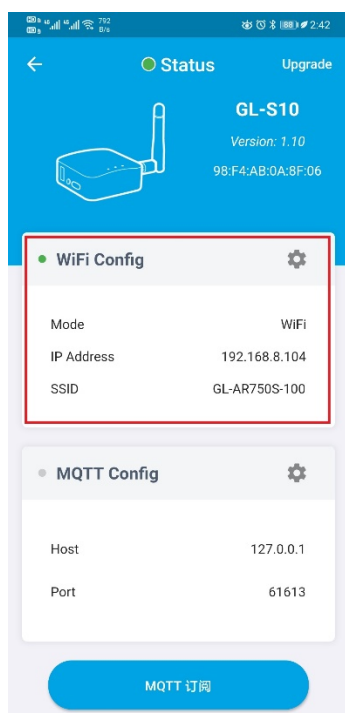
## 3.3 配置 WiFi

点击“Status”页面的设置 wifi 按钮，进入“WiFi Config”页面。

举例：连接 SSID 为：GL-AR750S-100 的 WiFi。如下图所示：



设置完后返回“Status”页面，待 GL-S10 连接上 WiFi 后再次下拉刷新页面，可以看到 WiFi Config 的数据已经更新，如下图所示：

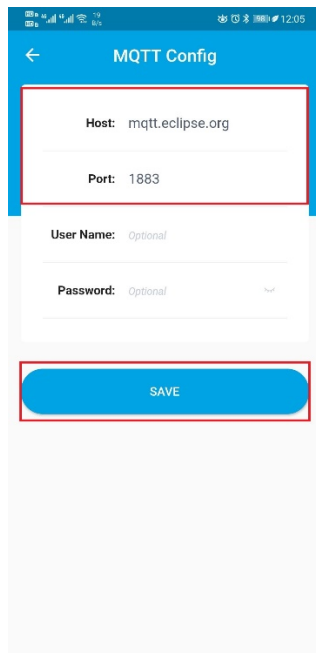


### 3.4 配置 MQTT

点击“Status”页面的设置 MQTT 按钮，进入“MQTT Config”页面。

在“HOST”栏输入需连接的 MQTT Broker 地址，在“PORT”栏输入端口号。如果需要用户名和密码的验证，则在“User Name”和“Password”中输入。

举例：连接地址为“[mqtt.eclipse.org](https://mqtt.eclipse.org)”，端口号为：1883，不需要用户名/密码校验的 MQTT Broker（用户也可使用该服务器做调试）

A screenshot of the 'MQTT Config' screen in a mobile application. The screen has a blue header with a back arrow and the title 'MQTT Config'. Below the header, there are two input fields: 'Host' with the value 'mqtt.eclipse.org' and 'Port' with the value '1883'. These two fields are enclosed in a red rectangular box. Below these fields are two more input fields: 'User Name' with the placeholder 'Optional' and 'Password' with the placeholder 'Optional'. At the bottom of the screen, there is a blue button with the text 'SAVE', which is also enclosed in a red rectangular box.

MQTT Config

Host: mqtt.eclipse.org

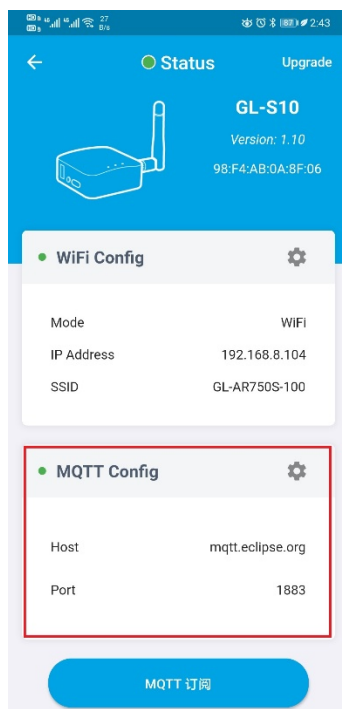
Port: 1883

User Name: Optional

Password: Optional

SAVE

设置完后返回“Status”页面，待 GL-S10 连接上 MQTT Broker 后再次下拉刷新页面，可以看到 MQTT Config 的数据已经更新，如下图所示：

A screenshot of the 'Status' screen in a mobile application. The screen has a blue header with a back arrow, a green status indicator, the title 'Status', and an 'Upgrade' button. Below the header, there is a section for 'GL-S10' with a version of '1.10' and a MAC address '9B:F4:AB:0A:BF:06'. Below this is a 'WiFi Config' section with a gear icon, showing 'Mode' as 'WIFI', 'IP Address' as '192.168.8.104', and 'SSID' as 'GL-AR750S-100'. Below the WiFi Config is an 'MQTT Config' section with a gear icon, showing 'Host' as 'mqtt.eclipse.org' and 'Port' as '1883'. This MQTT Config section is enclosed in a red rectangular box. At the bottom of the screen, there is a blue button with the text 'MQTT 订阅'.

Status Upgrade

GL-S10  
Version: 1.10  
9B:F4:AB:0A:BF:06

WiFi Config

Mode: WIFI  
IP Address: 192.168.8.104  
SSID: GL-AR750S-100

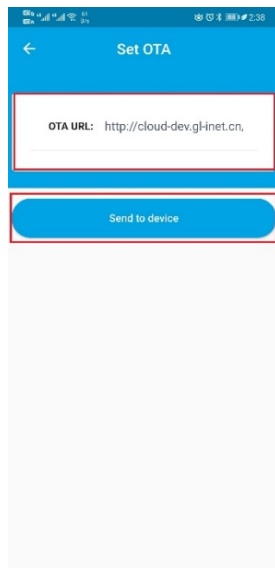
MQTT Config

Host: mqtt.eclipse.org  
Port: 1883

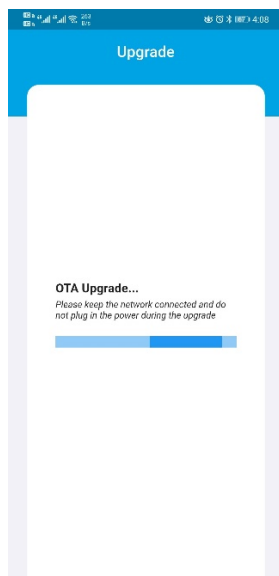
MQTT 订阅

### 3.5 设置 OTA

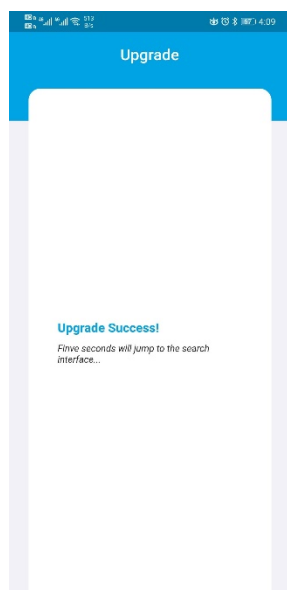
点击“Status”页面的“Upgrade”按钮，进入“Set OTA”页面。  
在“OTA URL”栏中输入 http 服务器的地址，然后点击“Send to device”。



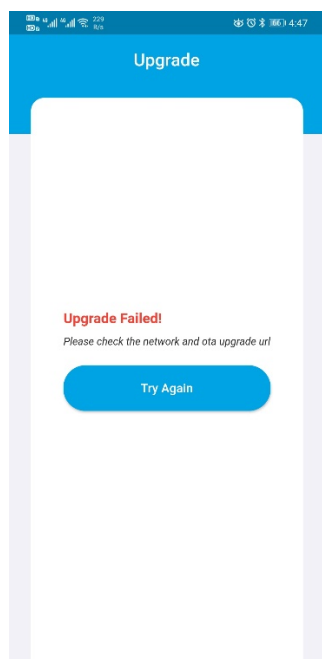
GL-S10 收到该指令后会自动开始 OTA 升级，如下图所示：



升级完成后会返回升级结果：



成功时：



失败时：

## 4 MQTT 收发测试

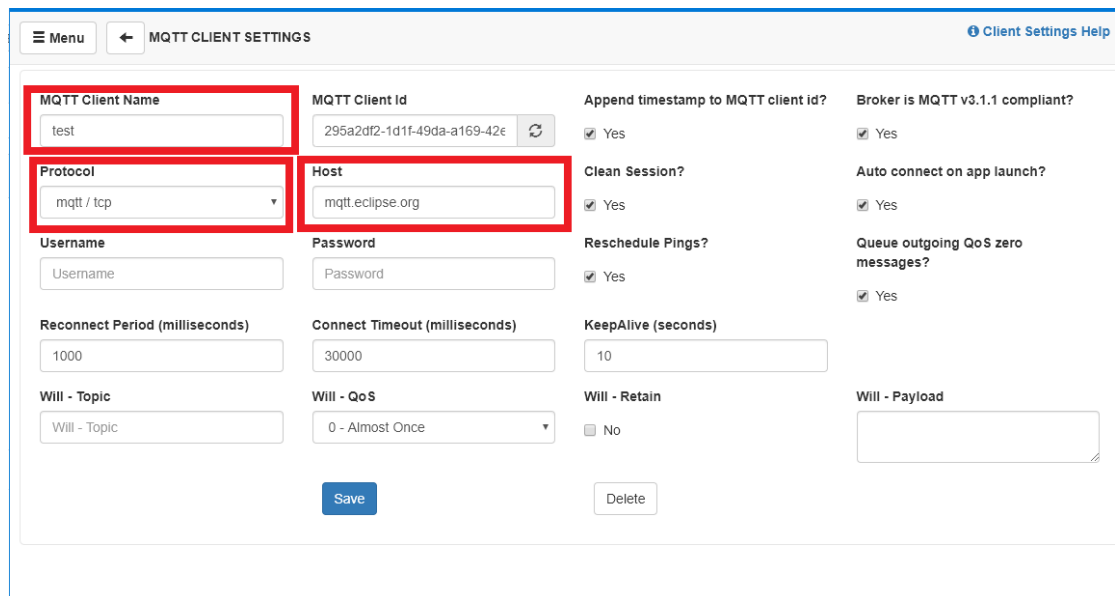
GL-S10 上打通了 ble 子设备<==>网关<==>云平台的数据链路。用户可以调用 ble API 往指定的 MQTT 服务器订阅及发送数据，在以下的演示中手机充当该数据链路中的 ble 子设备部分。

当 GL-S10 已连接到 MQTT 服务器后（配置 MQTT 的步骤见 [3.4 配置 MQTT](#)），用户可以使用另外的 MQTT Client 来进行收发测试。本文中使用的配测 MQTT Client 是 MQTTBox，可在 <http://workswithweb.com/html/mqttbox/downloads.html> 下载。

### 4.1 配置连接 MQTT Broker

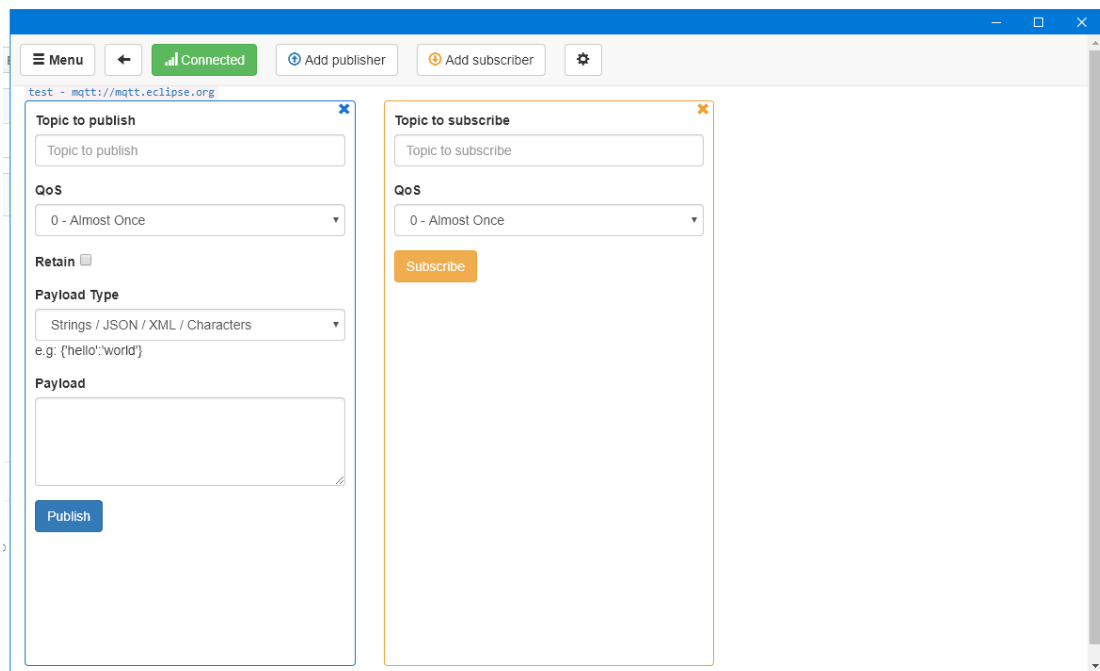
首先使用配置 APP 设置 GL-S10（步骤见 [3.4 配置 MQTT](#)），然后设置 MQTTBox 连接到同一个 MQTT Broker（配置如下图）





The screenshot shows the 'MQTT CLIENT SETTINGS' window. The 'MQTT Client Name' is set to 'test'. The 'MQTT Client Id' is '295a2df2-1d1f-49da-a169-42e'. The 'Protocol' is set to 'mqtt / tcp'. The 'Host' is 'mqtt.eclipse.org'. The 'Append timestamp to MQTT client id?' checkbox is checked. The 'Broker is MQTT v3.1.1 compliant?' checkbox is checked. The 'Clean Session?' checkbox is checked. The 'Auto connect on app launch?' checkbox is checked. The 'Queue outgoing QoS zero messages?' checkbox is checked. The 'Reconnect Period (milliseconds)' is '1000'. The 'Connect Timeout (milliseconds)' is '30000'. The 'KeepAlive (seconds)' is '10'. The 'Will - Topic' is 'Will - Topic'. The 'Will - QoS' is '0 - Almost Once'. The 'Will - Retain' checkbox is unchecked. The 'Will - Payload' is empty. The 'Save' button is highlighted.

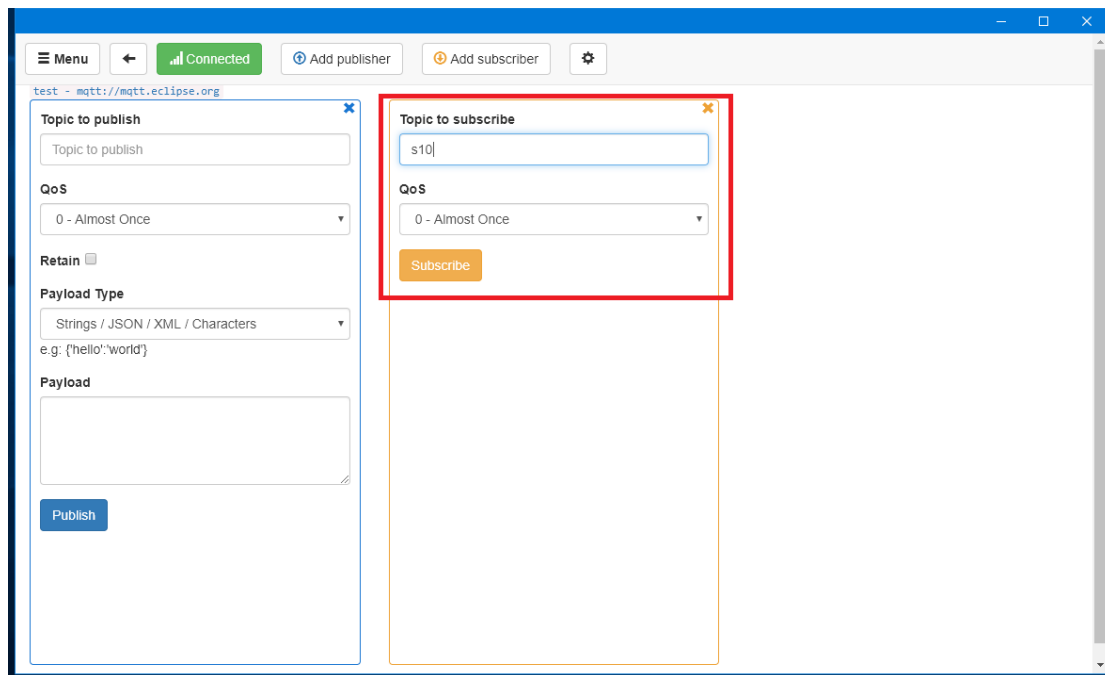
配置成功后会自动连接，连接成功如下图：



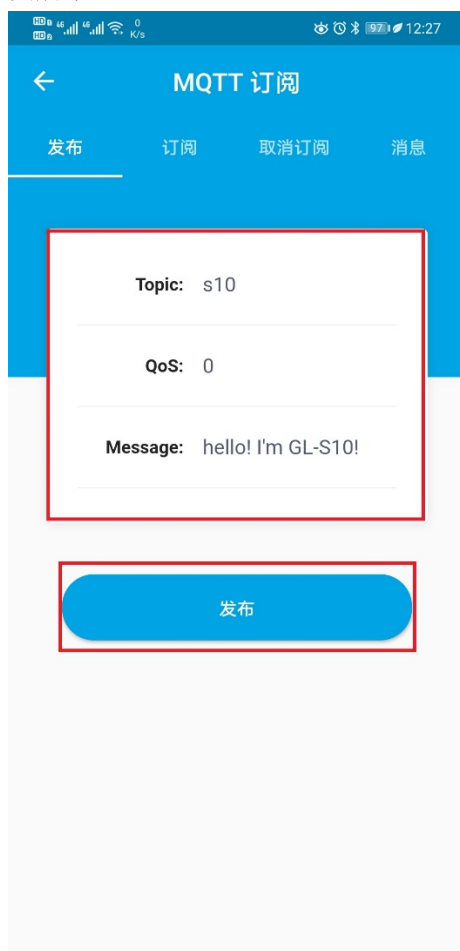
The screenshot shows the MQTTBox interface. The status bar at the top indicates 'Connected'. The 'Topic to publish' field is empty. The 'QoS' is set to '0 - Almost Once'. The 'Retain' checkbox is unchecked. The 'Payload Type' is set to 'Strings / JSON / XML / Characters'. The 'Payload' field is empty. The 'Publish' button is highlighted. The 'Topic to subscribe' field is empty. The 'QoS' is set to '0 - Almost Once'. The 'Subscribe' button is highlighted.

## 4.2 APP 发布消息，MQTTBox 接收

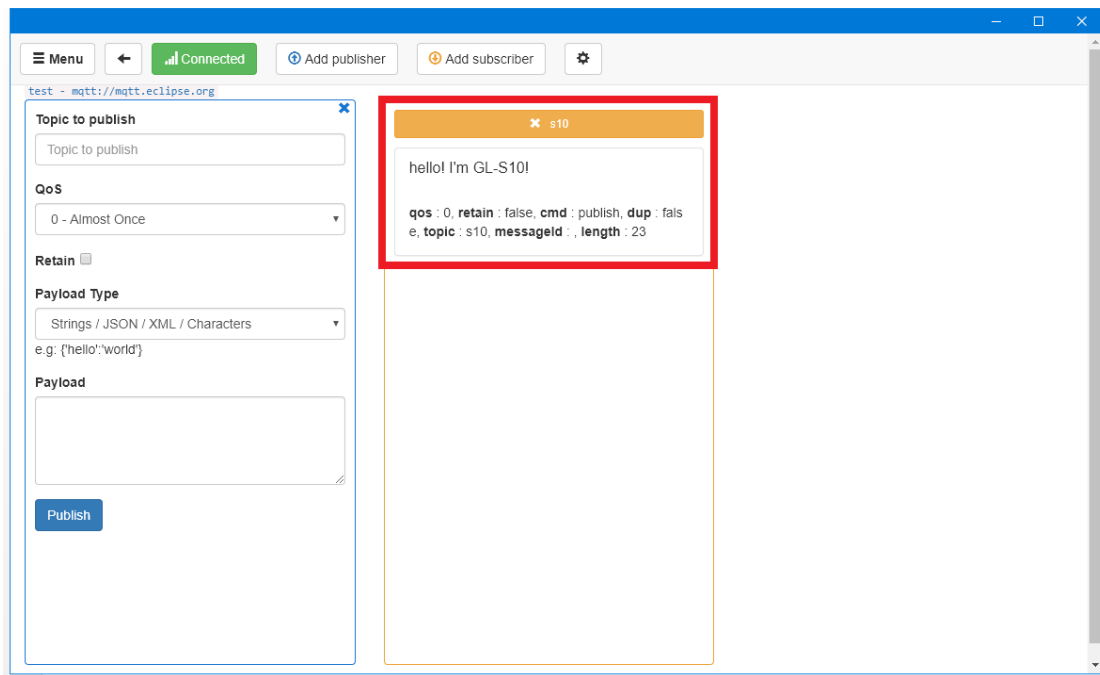
首先设置 MQTTBox 订阅 “s10” 主题，如下图：



使用配套 APP，在“MQTT 订阅页面”往“s10”主题发送“hello! I’m GL-S10!”，如下图所示：

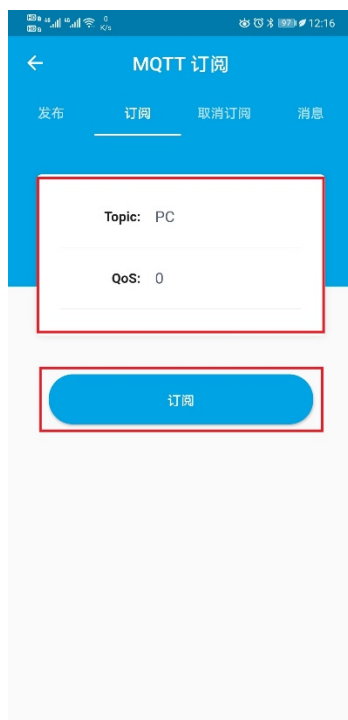


MQTTBox 接收到数据，如下图所示：

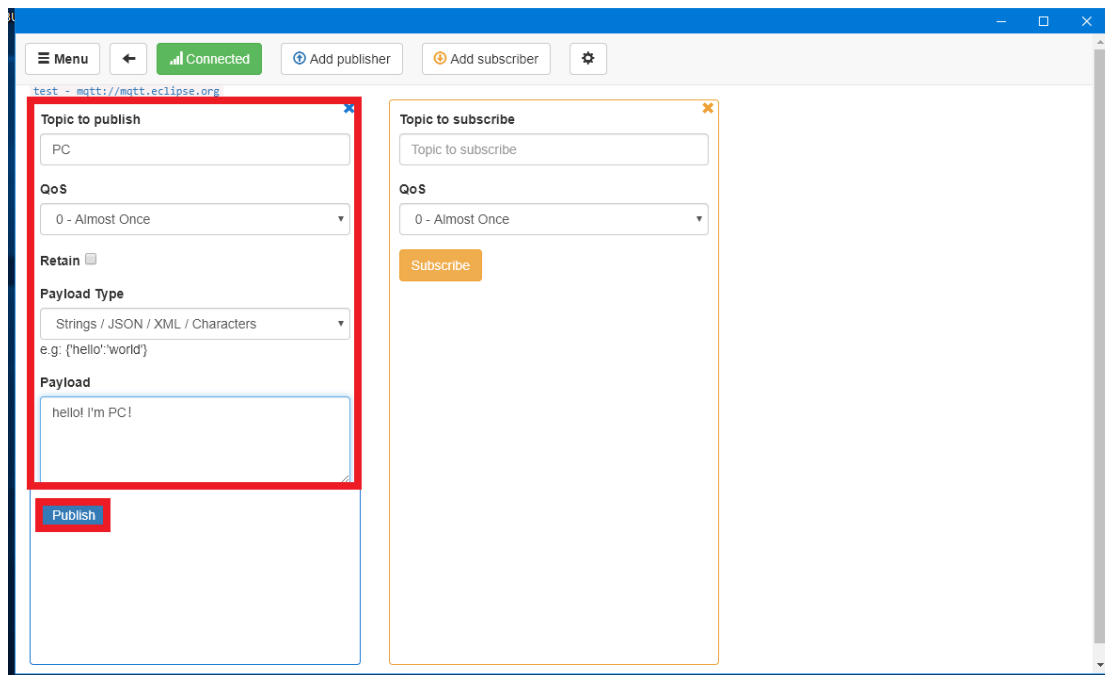


## 4.3 MQTTBox 发布消息，APP 接收

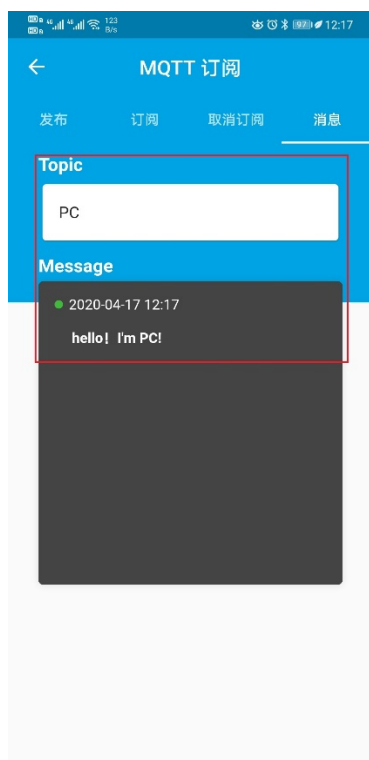
首先使用配套 APP，在 GL-S10 上订阅“PC”主题，如下图所示：



MQTTBox 往“PC”主题发送“hello! I'm PC!”

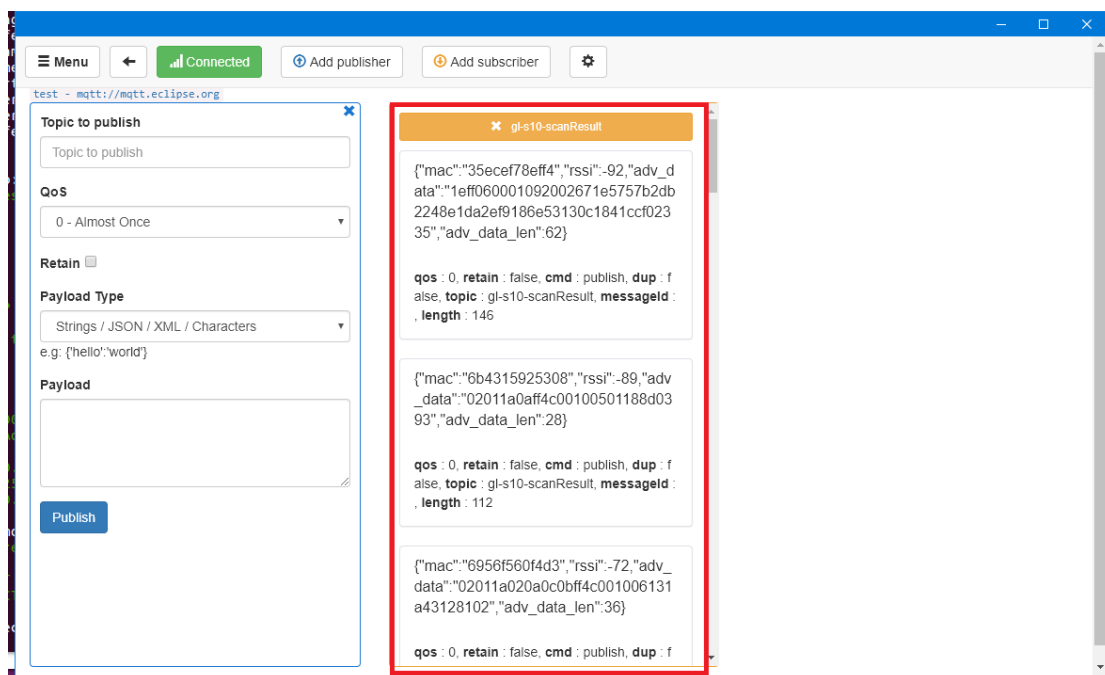
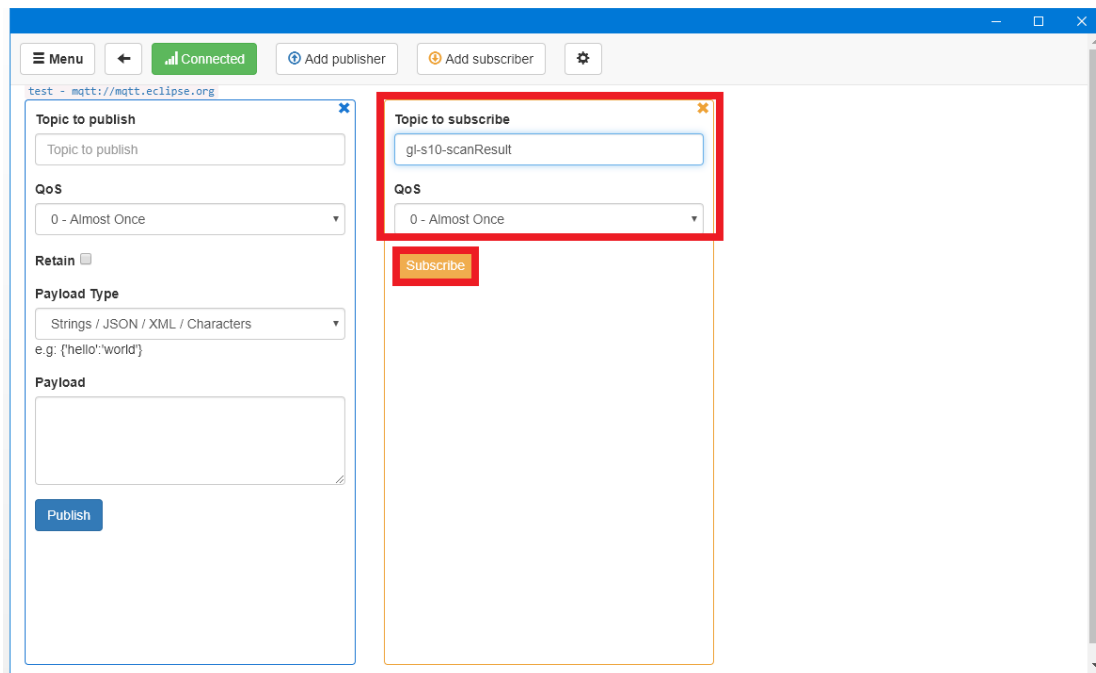


APP 上接收到 GL-S10 传来的数据，如下图所示：



## 5 BLE 扫描并上传至云平台

本例程中，在用户联网后默认开启 ble 扫描（每间隔 9 秒扫描 1 秒），如果连接了 MQTT 服务器，扫描的结果会以 JSON 的格式打包上传到"gl-s10-scanResult"主题。用户可以使用另外的 MQTT Client（如 MQTTBox）订阅该主题，获取数据。



# 附录 1 -- GL-S10 BLE API

GL-S10 包含两个 GATT 服务。第一个是为了接受指令和消息，服务的 UUID 是 000000FF-0000-1000-8000-00805F9B34FB，它包含一个特征值，特征值的 UUID 是 0000FF01-0000-1000-8000-00805F9B34FB，该特征值的读写权限只有可写。第二个是为了发送通知，服务的 UUID 是 000000EE-0000-1000-8000-00805F9B34FB，他也只有一个特征值，特征值的 UUID 是 0000EE01-0000-1000-8000-00805F9B34FB。该特征值的权限是可读可通知 (read | notify)。使能了第二个服务的 notify 功能，才可以获取通知。

## 1.1 指令格式

指令必须通过第一个 GATT 服务来发送，将要发送的指令写到第一个 GATT 的特征值里(服务 UUID 为 000000FF-0000-1000-8000-00805F9B34FB，特征值 UUID 为 0000FF01-0000-1000-8000-00805F9B34FB)即实现指令的发送。指令有特定的格式。每条指令有四个部分构成，开始符，指令码，指令内容，结束符。其中指令内容长度不定，可以为空。

Start Code	Length	CmdID	Payload
开始符（1 字节）	包长（2 字节）	指令码（1 字节）	数据段

开始符是固定的 1 个字节 0xFE ；  
包长是该数据包的长度（包含开始符）占 2 字节；  
指令码的详细内容见“1.3.1 指令汇总表”占 1 字节；  
数据段是指令码的参数，部分指令不需要参数，这个区域就为空，详细内容见下表“1.3.2 带参数的指令内容格式”。

**注意，整条指令必须转换成十六进制的 uint8 数组才可以写入 GATT 特征值里**

## 1.2 回应格式

每条指令执行后都会返回一个执行结果，来表明其执行是否成功。这个执行结果是通过第二个 GATT 服务的通知功能传递出来的（服务 UUID 000000EE-0000-1000-8000-00805F9B34FB，特征值 UUID ： 0000EE01-0000-1000-8000-00805F9B34FB）。

Start Code	Length	CmdID	result	Payload
开始符（1 字节）	包长（2 字节）	指令码（1 字节）	执行结果	数据段

GL-S10 的回应格式和指令格式在大体上一致的，执行结果“00”代表执行成功；“01”代表执行失败。

## 1.3 指令码

### 1.3.1 指令汇总表

指令码	指令	需要参数?	描述
0x01	<a href="#">SET_WIFI_CONFIG</a>	Y	设置 wifi 的 SSID 和密码
0x02	<a href="#">START_WIFI</a>	N	连接 wifi
0x03	<a href="#">STOP_WIFI</a>	N	断开 wifi
0x04	<a href="#">RESTART_WIFI</a>	N	重连 WiFi
0x05	<a href="#">GET_NETWORK_STATE</a>	N	获取当前网络状态
0x06	<a href="#">CLEAR_WIFI_CONFIG</a>	N	清空 wifi 信息
0x65	<a href="#">SET_MQTT_URI</a>	Y	设置 MQTT uri
0x66	<a href="#">START_MQTT</a>	N	连接 MQTT
0x67	<a href="#">STOP_MQTT</a>	N	断开 MQTT
0x68	<a href="#">RESTART_MQTT</a>	N	重连 MQTT
0x69	<a href="#">GET_MQTT_CONFIG</a>	N	获取当前 MQTT uri
0x6A	<a href="#">CLEAR_MQTT_CONFIG</a>	N	清除 MQTT 配置信息
0x6B	<a href="#">MQTT_SUBSCRIBE</a>	Y	订阅 MQTT 主题
0x6C	<a href="#">MQTT_UNSUBSCRIBE</a>	Y	取消订阅 MQTT 主题
0x6D	<a href="#">MQTT_PUBLISH</a>	Y	向 MQTT 服务器发布消息
0x6E	<a href="#">MQTT_SUBSCRIBE_MES</a>	/	订阅的 MQTT 主题有消息通知
0xF0	<a href="#">GET_VERSION</a>	N	获取当前固件版本号
0xF1	<a href="#">SET_OTA_URL</a>	Y	设置 OTA 服务器的 url
0xF2	<a href="#">OTA_STATUS</a>	/	OTA 升级状态通知
0xFA	<a href="#">RESTART</a>	N	重启设备

### 1.3.2 带参数的指令内容格式

指令码	操作	内容格式
0x01	SET_WIFI_CONFIG	ssid@password
0x65	SET_MQTT_URI	username:password@host:port
0x6B	MQTT_SUBSCRIBE	qos@topic
0x6C	MQTT_UNSUBSCRIBE	topic
0x6D	MQTT_PUBLISH	qos@topic@data
0XF1	SET_OTA_URL	updateNow@url

## 1.4 API 详情

### 1.4.1 SET\_WIFI\_CONFIG

SET\_WIFI\_CONFIG 用于设置 wifi 配置，指令码 0x01。

指令码	指令	内容格式
0x01	SET_WIFI_CONFIG	ssid@password

举例：要连接 wifi 的 SSID：GL-S1300-9cc，密码：12345678

**fe 00 16 01 47 4c 2d 53 31 33 30 30 2d 39 63 63 40 31 32 33 34 35 36 37 38**

回应：

若连接成功：**fe 00 05 01 00**

若 20s 内未连接成功：**fe 00 05 01 01**

### 1.4.2 START\_WIFI

START\_WIFI 用于启动连接 wifi，指令码 0x02。

指令码	指令	内容格式
0x02	START_WIFI	无

举例：

**fe 00 04 02**

回应：

若启动成功：**fe 00 05 02 00**

若启动失败：**fe 00 05 02 01**

### 1.4.3 STOP\_WIFI

STOP\_WIFI 用于关闭 wifi 连接，指令码 0x03。

指令码	指令	内容格式
0x03	STOP_WIFI	无

举例：

**fe 00 04 03**

回应：

若关闭成功：**fe 00 05 03 00**

若关闭失败：**fe 00 05 03 01**



#### 1.4.4 RESTART\_WIFI

RESTART\_WIFI 用于重启 wifi 连接，指令码 0x04。

指令码	指令	内容格式
0x04	RESTART_WIFI	无

举例:

**fe 00 04 00**

回应:

若重启成功: **fe 00 05 04 00**

若重启失败: **fe 00 05 04 01**

#### 1.4.5 GET\_NETWORK\_STATE

GET\_NETWORK\_STATE 用于获取当前网络状态，指令码 0x05。

指令码	指令	内容格式
0x05	GET_NETWORK_STATE	无

举例:

**fe 00 04 05**

回应:

data 段格式	例子
state@mode@IP_address@ssid	1@0@192.168.1.111@testwifi

“state” 表示连接的状态（网络连接或者是 MQTT 服务器连接），1 表示连接，0 表示连接断开。“mode” 是表示连接方式，0 表示通过 wifi 连接，1 表示通过网线连接。

**fe 00 1f 05 00 31 40 30 40 31 39 32 2e 31 36 38 2e 31 2e 31 31 31 40 74 65 73 74 77 69 66 69**  
====> 获取到当前网络状态为：网络已连接，通过 wifi，ip 地址为 192.168.1.111，连接到的 wifi ssid 为 testwifi。

#### 1.4.6 CLEAR\_WIFI\_CONFIG

CLEAR\_WIFI\_CONFIG 用于清除当前 wifi 配置，指令码 0x06。

指令码	指令	内容格式
0x06	CLEAR_WIFI_CONFIG	无

举例:

**fe 00 04 06**

回应:

若成功: **fe 00 05 06 00**

若失败: **fe 00 05 06 01**

## 1.4.7 SET\_MQTT\_URI

SET\_MQTT\_URI 用于设置 MQTT 配置, 指令码 0x65。

指令码	指令	内容格式
0x65	SET_MQTT_URI	mqtt://username:password@host:port

举例: 要连接的 mqtt 服务端 host: iot.cloud.com; port: 1883; 用户名: admin; 密码: password

**fe 00 2c 65 6d 71 74 74 3a 2f 2f 61 64 6d 69 6e 3a 70 61 73 73 77 6f 72 64 40 69 6f 74 2e 63 6c 6f 75 64 2e 63 6f 6d 3a 31 38 38 33**

回应:

若成功: **fe 00 05 65 00**

若失败: **fe 00 05 65 01**

## 1.4.8 START\_MQTT

START\_MQTT 用于启动 MQTT 连接, 指令码 0x66。

指令码	指令	内容格式
0x66	START_MQTT	无

举例:

**fe 00 04 66**

回应:

若成功: **fe 00 05 66 00**

若失败: **fe 00 05 66 01**

## 1.4.9 STOP\_MQTT

STOP\_MQTT 用于中断 MQTT 连接, 指令码 0x67。

指令码	指令	内容格式
0x67	STOP_MQTT	无

举例:

**fe 00 04 67**

回应:

若成功: **fe 00 05 67 00**

若失败: **fe 00 05 67 01**

### 1.4.10 RESTART\_MQTT

RESTART\_MQTT 用于重新连接 MQTT 服务器，指令码 0x68。

指令码	指令	内容格式
0x68	RESTART_MQTT	无

举例:

**fe 00 04 68**

回应:

若成功: **fe 00 05 68 00**

若失败: **fe 00 05 68 01**

### 1.4.11 GET\_MQTT\_CONFIG

GET\_MQTT\_CONFIG 用于获取当前 MQTT 配置信息，指令码 0x69。

指令码	指令	内容格式
0x69	GET_MQTT_CONFIG	无

举例:

**fe 00 04 69**

回应:

数据段格式	例子
state@user:password@host:port	1@admin:password@192.168.8.1:1107

“state” 表示 MQTT 服务器连接的状态，1 表示连接，0 表示连接断开。“user” 是用户名（没有可即为空），“password” 是密码（没有即为空），“host” 是 MQTT 服务器的域名或 IP，“port” 是 MQTT 服务器的端口。

**fe 00 26 69 00 31 40 61 64 6d 69 6e 3a 70 61 73 73 77 6f 72 64 40 31 39 32 2e 31 36 38 2e 38 2e 31 3a 31 31 30 37** ==> 当前 MQTT 配置为: 已连接到服务器，用户名是“admin”，密码是“password”，服务器 IP 是“192.168.8.1”，端口是“1107”。

### 1.4.12 CLEAR\_MQTT\_CONFIG

CLEAR\_MQTT\_CONFIG 用于清除当前 MQTT 配置，指令码 0x6a。

指令码	指令	内容格式
0x6a	CLEAR_MQTT_CONFIG	无

举例:

**fe 00 04 6a**

回应:

若成功: **fe 00 05 6a 00**

若失败: **fe 00 05 6a 01**

### 1.4.13 MQTT\_SUBSCRIBE

MQTT\_SUBSCRIBE 用于 MQTT 订阅主题消息, 指令码 0x6b。

指令码	指令	内容格式
0x6b	MQTT_SUBSCRIBE	qos@topic

举例: 订阅一个 “test” 主题, qos 为 0

**fe 00 0a 6b 30 40 74 65 73 74**

回应:

若成功: **fe 00 05 6b 00**

若失败: **fe 00 05 6a 01**

### 1.4.14 MQTT\_UNSUBSCRIBE

MQTT\_UNSUBSCRIBE 用于 MQTT 取消订阅主题信息, 指令码 0x6c。

指令码	指令	内容格式
0x6c	MQTT_UNSUBSCRIBE	topic

举例: 取消订阅一个 “test” 主题

**fe 00 08 6c 74 65 73 74**

回应:

若成功: **fe 00 05 6c 00**

若失败: **fe 00 05 6c 01**

### 1.4.15 MQTT\_PUBLISH

MQTT\_PUBLISH 用于 MQTT 发布主题消息, 指令码 0x6d。

指令码	指令	内容格式
0x6d	MQTT_PUBLISH	qos@topic@data

举例: 往 “test” 主题发送一个 “hello” 的消息, qos 为 0

**fe 00 12 6c 30 40 74 65 73 74 40 74 6f 70 69 63**

回应:

若成功: **fe 00 05 6d 00**

若失败: **fe 00 05 6d 01**

### 1.4.16 GET\_VERSION

GET\_VERSION 用于获取当前固件的版本号。

指令码	指令	内容格式
0XF0	GET_VERSION	无

举例：

**fe 00 04 f0**

回应：

数据段格式	例子
version	1.00

“version”即是当前固件的版本号。**fe 00 00 f0 00 31 2e 30 30** ==> 当前固件版本号为 1.00

### 1.4.17 SET\_OTA\_URL

SET\_OTA\_URL 用于设置 OTA 升级的服务器链接，并可设置是否立即 OTA 升级。

指令码	指令	内容格式
0XF1	SET_OTA_URL	updateNow@url

“updateNow”即为是否立即 OTA 升级，占 1 字节，0x00 为立即升级，0x01 为不立即升级。

举例：设置 OTA 升级的服务器链接是：[http://cloud-dev.gl-inet.cn/gl-s10\\_app.bin](http://cloud-dev.gl-inet.cn/gl-s10_app.bin)，且立即 OTA 升级。

**fe 00 2e f1 68 74 74 70 3a 2f 2f 63 6c 6f 75 64 2d 64 65 76 2e 67 6c 2d 69 6e 65 74 2e 63 6e 2f 67 6c 2d 73 31 30 5f 61 70 70 2e 62 69 6e**

回应：

若成功：**fe 00 05 f1 00**

若失败：**fe 00 05 f1 01**

### 1.4.18 RESTART

RESTART 用于立即重启设备，该指令没有返回，**一般情况下不推荐使用**。

指令码	指令	内容格式
0xF2	RESTART	无

举例：

**fe 00 04 f2**

## 1.5 主动信息通知

除了返回指令执行结果之外，还有在特定场景下触发的 ble 信息，也是通过 ble notify 的形式下发的。

### 1.5.1 订阅 MQTT 信息通知

当 MQTT 订阅的主题有数据通知时，会通过 ble 传出。

Start Code	Length	CmdID	Payload
开始符（1 字节）	包长（2 字节）	指令码（0x6E）	数据段（topic@data）

例如，收到通知 **fe 00 15 6e 74 65 73 74 40 48 65 6c 6c 6f 20 77 6f 72 6c 64 21** 的意思是收到 MQTT 服务器推送的一条来自于订阅主题“test”的消息“Hello world!”

### 1.5.2 OTA 状态通知

当 OTA 升级通过 http 拉取固件并烧录结束或者该过程中出现失败，都会有 OTA 状态通知通过 ble 传出。

Start Code	Length	CmdID	Payload
开始符（1 字节）	包长（2 字节）	指令码（0xF2）	数据段（1 字节）

Payload 段 0x00 表示成功，0x01 表示失败

例如：

OTA 升级成功，即将马上重启：**fe 00 05 f2 00**

OTA 失败：**fe 00 05 f2 01**