OPL1000

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

MQTT Demo User Guide

OPULINKS

REVISION HISTORY

Date	Version	Contents Updated
2019-07-25	0.1	• Initial Release

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1. Application Scope

This document describes the procedure to set up OPL1000 as MQTT Client using SDK API. This is followed by connecting MQTT Client with MQTT Broker to realize the goal of distributing and subscribing to information.

1.2. Abbreviations

Abbr.	Explanation
AP	Wireless Access Point
APP	APPlication
APS	Application Sub-system, refers to M3 MCU in this document
Blewifi	BLE config WIFI
DevKit	Development Kit
DTIM	Delivery Traffic Indication Message
MQTT	Message Queuing Telemetry Transport

1.3. References

[1] DEVKIT Quick Start Guide OPL1000-DEVKIT-getting-start-guide.pdf

[2] Download Tool User Guide OPL1000-patch-download-tool-user-guide.pdf

[3] SDK Application Development Guide OPL1000-SDK-Development-guide.pdf

2. MQTT PROGRAM DESIGN

2.1. Working Principle

The directory for MQTT sample program is under

SDK\APS_PATCH\examples\protocols\mqtt

Its operating procedure is as follow:

1 Connect to Bluetooth, OPL1000 establishes connection with AP in station mode.
2 Successful connection, establish connection with designated TCP Server and port number after obtaining IP.
3 After successful connection, subscribe and periodically publish and receive information.

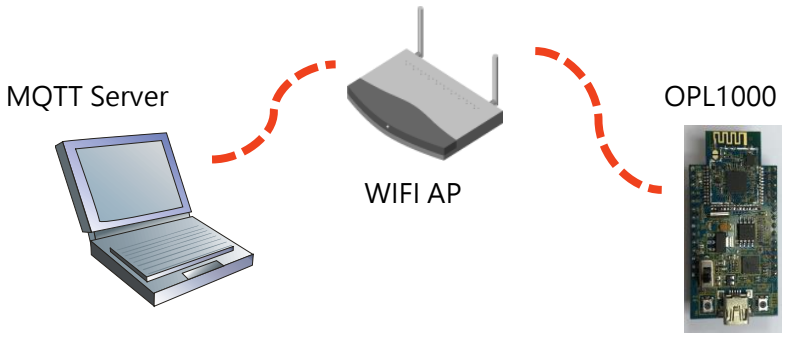
The MQTT server, port number, topic for the information published, topic for the information subscribed, information published and the maximum length for the information published are defined in the mqtt_client.h files as shown below.

```
// MQTT server ip & port.
#define  TCP_SERVER_ADDR     "192.168.1.112"
#define  TCP_SERVER_PORT     1883
// Topic names & msgs.
#define  MQTT_SUB_TOPIC       "sys_M_Sub_T"
#define  MQTT_PUB_TOPIC       "sys_M_Pub_T"
#define  OPL1000_SUB_TOPIC    "OPL1_Sub_T"
#define  OPL1000_PUB_TOPIC    "OPL1_Pub_T"
#define  MQTT_PUB_MSG         "Hello,MQTT!"
#define  OPL1000_PUB_MSG      "Hi,MQTT"
#define  OPL1_PUB_MSG_MAX_LEN  30
```

The MQTT information exchange network topology established between MQTT server and OPL1000 is as shown in Figure 1.

OPL1000 plays the role of Station to connect to WIFI AP and establishes connection with MQTT server to publish and subscribe to information.

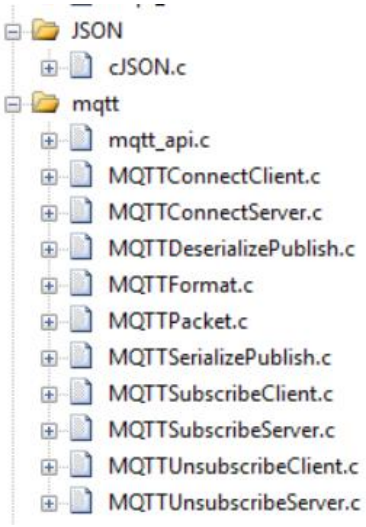
Figure 1：Sample of OPL1000 MQTT client network connection diagram



2.2. Third-Party Library

The MQTT routine uses two third-party libraries, JSON and MQTT as shown below：

Figure 2: The content of a third party library



3. MQTT AUTHENTICATE FUNCTION

3.1. Example of editing mqtt

Step1: Use Keil C to open SDK\APS_PATCH\examples\protocols\mqtt\opl1000_app_m3.uvprojx mqtt project file.

Step2: Depending on the situation, edit the parameters in mqtt_client.h file accordingly as shown below.

```
// MQTT server ip & port.
#define  TCP_SERVER_ADDR     "192.168.1.112"
#define  TCP_SERVER_PORT     1883
// Topic names & msgs.
#define  MQTT_SUB_TOPIC       "sys_M_Sub_T"
#define  MQTT_PUB_TOPIC       "sys_M_Pub_T"
#define  OPL1000_SUB_TOPIC    "OPL1_Sub_T"
#define  OPL1000_PUB_TOPIC    "OPL1_Pub_T"
#define  MQTT_PUB_MSG         "Hello,MQTT!"
#define  OPL1000_PUB_MSG      "Hi,MQTT"
#define  OPL1_PUB_MSG_MAX_LEN  30
```

Step3: Depending on the situation, edit the parameters in blewifi_configurations.h file (whenever the content of the file has been updated, the macro corresponding value for the MW_FIM_VER08_PROJECT should be increased by 1) as shown below.

```
/*
FIM version
*/
#define MW_FIM_VER08_PROJECT         0x04   // 0x00 ~ 0xFF
/* DTIM the times of Interval: ms
*/
#define BLEWIFI_WIFI_DTIM_INTERVAL            (2000)    // ms
```

Step4: Edit mqtt project

Please refer to reference [3] SDK Application Development Guide for the tool setting and editing procedure in Keil C.

3.2. Firmware Download

After a successful edit, opl1000_app_m3.bin file will be produced under the SDK\APS_PATCH\examples\protocols\mqtt\Output\Objects directory. Copy the file to FW_Pack directory and use the download tool to combine it with the m0 bin file. This is followed by

downloading the resultant file into DEVKIT. Please refer to reference [2] Download Tool User Guide for the use of the downloaded tool. Please refer to reference [1] DEVKIT Quick Start Guide for the use of DEVKIT.

3.3. Function to authenticate information subscription and distribution

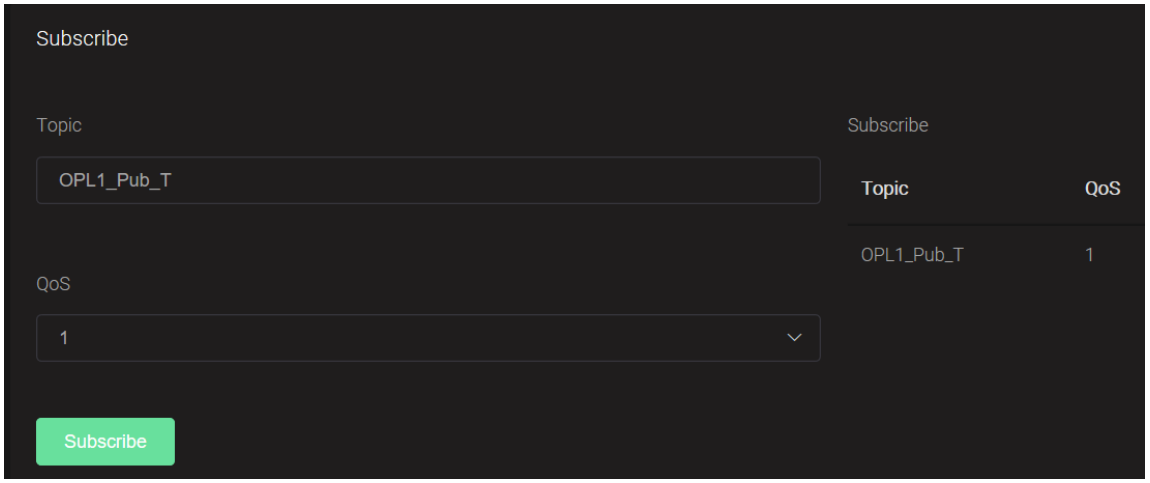
In the MQTT routine, the MQTT Server terminal requires to start EQM as broker. The link to download EMQ is as follow: https://www.emqx.io/cn/downloads

During authentication, the user needs to login to EMQ Dashboard through the link http://*<MQTT Server>*:18083/ and establishes a mqtt connection as shown in the figure below.

Figure 3: Establish mqtt Client

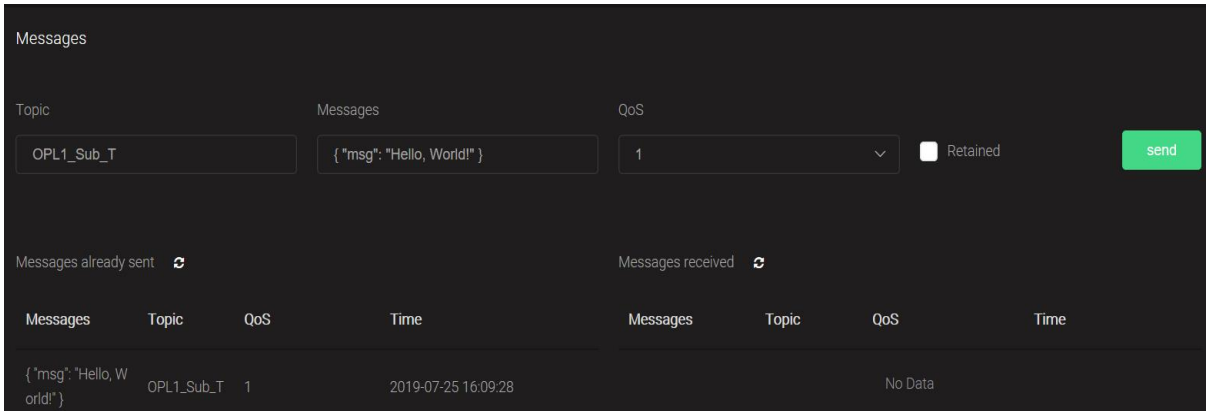
Subscribe to the TOPIC macro defined by OPL1000_PUB_TOPIC in mqtt_client.h file and select 1 as QoS.

Figure 4: mqtt Client information subscription



Publish the TOPIC macro defined by OPL1000_SUB_TOPIC in mqtt_client.h file and select 1 as QoS.

Figure 5: mqtt Client information publication



After publishing or receiving the information mentioned above on OPL1000, the information will be resolved and exported to COM port as shown below.

```
----------start:publish----------------
message : Hi,MQTT 4
..[] rc = 26
mqtt pub ok, topic:OPL1_Pub_T; len:26
32 18 00 0A 4F 50 4C 31 5F 50 75 62 5F 54 00 00 48 69 2C 4D 51 54 54 20 34 0A
topick  : OPL1_Pub_T
message : Hi,MQTT 4
trans is ok,len = 26
Successfully published
----------end----------------
other pack = 4 skip it.
other pack = 4 skip it.
topic:OPL1_Sub_T; message:" { "msg": "Hello, World!" }"   qos:1  msgid:2
..[] rc = 4
send puback ok
other pack = -1 skip it.
----------start:publish----------------
```

CONTACT

sales@Opulinks.com