

# OPL1000

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

## 阿里云透传应用说明文档



**OPULINKS**

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

Copyright © 2017-2020, Opulinks. All Rights Reserved.

OPL1000-阿里透传应用说明文档 | Version1.1

版本纪录

Date	Version	Contents Updated
2020-02-09	1.0	初版
2020-04-22	1.1	章节 3.7 · 修正 at+cloudpub 输入指令

## 目录

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. 工程构成和工作原理	2
2.1. 工程构成	2
2.2. 工作原理	3
3. 编译和运行 Ali 透传范例	5
3.1. 在 Ali 云端创建 IoT 设备	5
3.2. 编译 Ali 透传范例	5
3.2.1. Keil C 编译	5
3.2.2. GCC 编译	6
3.3. Pack 成 bin	6
3.4. 下载并执行 Ali 透传固件	7
3.5. 通过 AT 命令完成蓝牙配网	8
3.6. 通过 AT 命令完成阿里云连接	10
3.7. 通过 AT 命令完成数据上报及下发	10
4. 基于 Ali 透传范例开发应用	12
4.1. 修改 Ali IoT 设备模型	12
4.2. 基于已有范例增添功能	12
4.2.1. 增加外设	12
4.2.2. 云端配置	12
4.2.3. 连接阿里云的工作原理	14
4.3. 工程 Heap Size 调整	14
4.4. 低功耗设置和验证	17
4.5. 扩充 AT Cmd	18
4.6. 验证添加的功能	18
5. 进阶：Ali SDK 更新	20
5.1. Ali MQTT 收发 Buffer Size 调整	20
6. FAQ	21

## 图目录

FIGURE 1: 云智能 APP 下载链接 .....	2
FIGURE 2: 项目文件 .....	2
FIGURE 3: 阿里云透传参考设计构成框图 .....	4
FIGURE 4: KEIL C 编译 .....	5
FIGURE 5: GCC 编译 .....	6
FIGURE 6: PACK OPL1000.BIN 固件 .....	6
FIGURE 7: OTA PACK .....	7
FIGURE 8: 下载透传功能固件 .....	7
FIGURE 9: DEBUG 串口输出信息 .....	8
FIGURE 10: 切换 BLEWIFI 模式 .....	9
FIGURE 11: 开启蓝牙广播 .....	9
FIGURE 12: 阿里配网 .....	9
FIGURE 13: 云连接 .....	10
FIGURE 14: 进入睡眠 .....	10
FIGURE 15: 数据上报和下发 .....	11
FIGURE 16: 配置外设 IO .....	12
FIGURE 17: 标准功能定义 .....	13
FIGURE 18: 添加标准功能 .....	13
FIGURE 19: 配置云智能 APP 操作界面 .....	14
FIGURE 20: 连云事件触发 .....	14
FIGURE 21: 从 MAP 文件中获取首末地址 .....	15
FIGURE 22: 更新 SCT FILE 中的 SIZE .....	16
FIGURE 23: 更新 HEAP SIZE 相关的信息 .....	16
FIGURE 24: SMART SLEEP 相关的宏定义 .....	17

FIGURE 25: AT 命令实现函数映射表..... 18

FIGURE 26: 新增功能控件显示..... 18

FIGURE 27: AT 命令验证反馈 ..... 19

FIGURE 28: ALI 收发 BUFFER SIZE 的设置 ..... 20

FIGURE 29: 蓝牙广播间隔配置..... 21

## 1. 介绍

### 1.1. 文档应用范围

本文档介绍了一个基于 OPL1000 A2 芯片通过透传的方式连接 Ali 云实现物联网应用的范例。该范例包含蓝牙配网功能，OPL1000 作为从设备通过串口接收来自主设备的 AT 命令，实现连接 Ali 云，并通过预先定义的 MQTT Topic 实现信息包的订阅和发布。

### 1.2. 缩略语

缩写	说明
AP	Wireless Access Point 无线访问接入点
DevKit	Develop Kit OPL1000 产品板
FW	FirmWare 固件，处理器上运行的嵌入式软件
MQTT	消息队列遥测传输(Message Queuing Telemetry Transport)
UART	Universal Asynchronous Receiver / Transmitter 通用非同步收发传输器

### 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)

## 2. 工程构成和工作原理

透传，即透明传输（transparent transmission），指的是在通讯中不管传输的业务内容如何，只负责将传输的内容由源地址传输到目的地，而不对业务数据内容做任何改变。

### 2.1. 工程构成

阿里云透传参考设计需要下载阿里云智能 APP。云智能 APP 是阿里云物联网云智能手机应用程序，用于 OPL1000 连接阿里云以及设备的数据显示及操作（APP 软件可以扫描下面二维码下载，或者手机应用市场直接搜索‘云智能’）。

Figure 1: 云智能 APP 下载链接

云智能APP 分为用户版和开发版，对于用户版需要产品发布才可以使用，否则无法扫描到蓝牙设备，对于开发版本则无此限制，云智能下载二维码如下：



注：使用的公版APP版本，若使用用户版本，则需要产品发布后才可以使用的，否则无法扫描对应设备的蓝牙，而开发版本则无此限制

Figure 2: 项目文件

src	2020/2/13 15:01
Makefile	2020/2/13 14:57
opl1000_tt_m3.bat	2020/2/13 14:57
opl1000_tt_m3.ini	2020/2/13 14:57
opl1000_tt_m3.sct	2020/2/13 14:57
opl1000_tt_m3.uvoptx	2020/2/13 14:57
opl1000_tt_m3.uvprojx	2020/2/13 14:57
readme.md	2020/2/13 14:57

目录和文件	说明
src	存放 OPL1000 蓝牙配网 · 数据收发相关.c 和.h 文件 · 以及 main 文件
Makefile	用于以 gcc 方式编译的 makefile 文件
opl1000_tt_m3.bat opl1000_tt_m3.ini opl1000_tt_m3.sct opl1000_tt_m3.uvoptx opl1000_tt_m3.uvprojx	编译工程文件

2.2. 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（安装阿里云智能 APP），阿里云和外部主控设备（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。当数据需要上传到阿里云时，数据被打包成特定的网络协议包（例如 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。外部主控设备通过 AT 命令控制 OPL1000 发送蓝牙广播信号，然后透过云智能 APP 连接完成蓝牙配网动作。
2. 主控设备通过 AT 命令连接到阿里云服务器，建立连接。
3. 采用透传方式发送 TCP 包，实现和阿里云端的数据交换。



Figure 3: 阿里云透传参考设计构成框图



## 3. 编译和运行 Ali 透传范例

运行 OPL1000 阿里云透传参考设计应用包含 3 个步骤。

- 1 使用固件下载工具，下载使用工程编译的固件。
- 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 MQTT 连接。发送和接收 TCP 数据。

### 3.1. 在 Ali 云端创建 IOT 设备

如何在阿里云端创建 IOT 设备，可以参考文档 Ali 5 元组和应用程序产生指导文档

OPL1000\_ali\_key&app\_create\_guide.pdf

访问链接：[https://github.com/Opulinks-Tech/OPL1000A2-Sensor-Device-Reference-Code-Ali-Cloud-with-MQTT/tree/master/doc/OPL1000\\_ali\\_key&app\\_create\\_guide.pdf](https://github.com/Opulinks-Tech/OPL1000A2-Sensor-Device-Reference-Code-Ali-Cloud-with-MQTT/tree/master/doc/OPL1000_ali_key&app_create_guide.pdf)

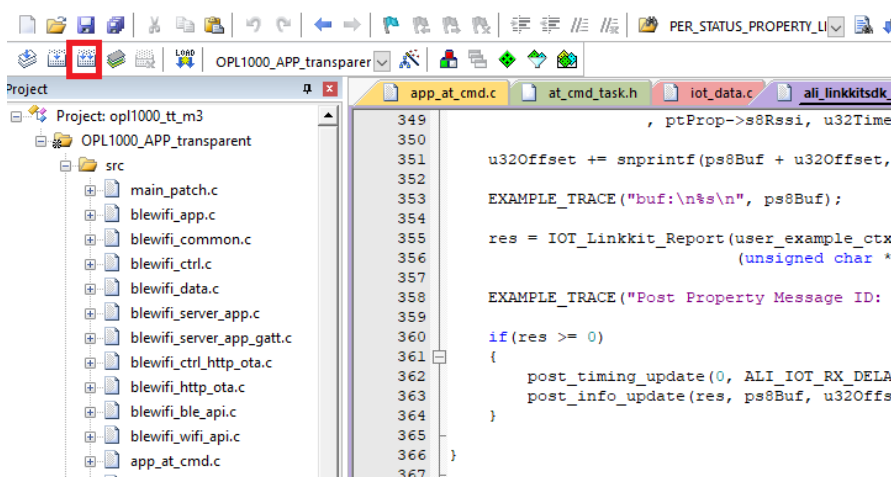
### 3.2. 编译 Ali 透传范例

阿里透传范例支持两种编译方式，一种是 keil c 编译，另一种是 GCC 编译

#### 3.2.1. Keil C 编译

打开 opl1000\_tt\_m3.uvprojx 工程文件，点击 build 按钮即可完成工程的编译，编译完成会在相对目录 Output\Objects 目录下生成对应的 opl1000\_app\_m3\_transparent.bin 文件。

Figure 4: Keil C 编译



Copyright © 2020, Opulinks. All Rights Reserved.  
OPL1000-阿里透传应用说明文档\_UG1-06-1.1

3.2.2. GCC 编译

在 makefile 文件所在目录打开对应的 cmd 界面，输入 make 命令，即可完成编译，产生的编译文件放在相对目录.\Output-GCC\opl1000\_app\_m3\_transparent.bin

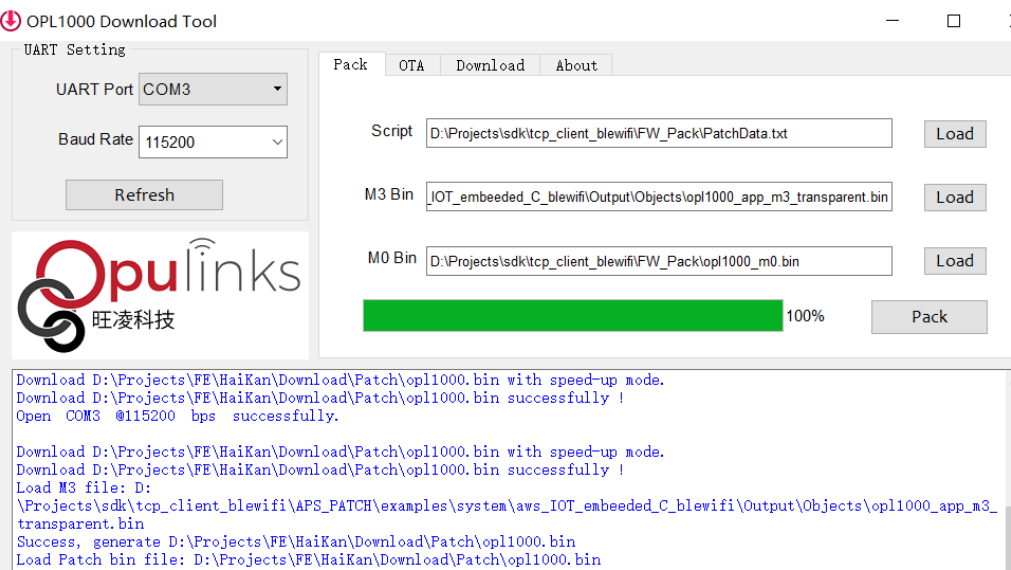
Figure 5: GCC 编译

```
E:\SDK\OPL1000\sh_sdk2\A2\SDK\APS_PATCH\examples\system\ali_blewifi>make
arm-none-eabi-objcopy -O binary ./Output-GCC/opl1000_app_m3_transparent.elf ./Output-GCC/opl1000_app_m3_transparent.bin
```

3.3. Pack 成 bin

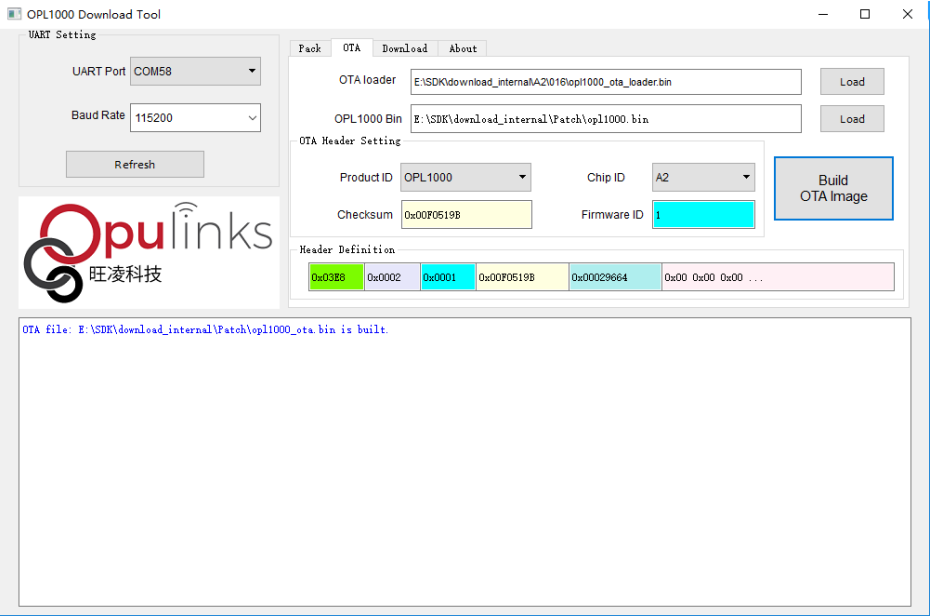
在 download tool 的 pack 界面上，选择相应的 opl1000\_app\_m3\_transparent.bin 文件，将它 pack 成 OPL1000.bin 如下。

Figure 6: pack OPL1000.bin 固件



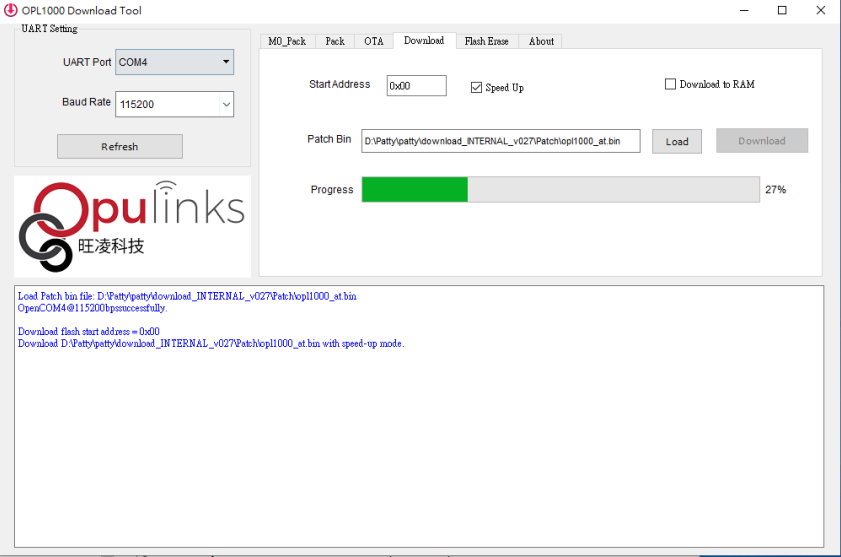
如需 OTA 功能，将产生的 OPL1000.bin 与 opl1000\_ota\_loader.bin 进行 pack 生成完整的固件 opl1000\_ota.bin

Figure 7: OTA pack



3.4. 下载并执行 Ali 透传固件

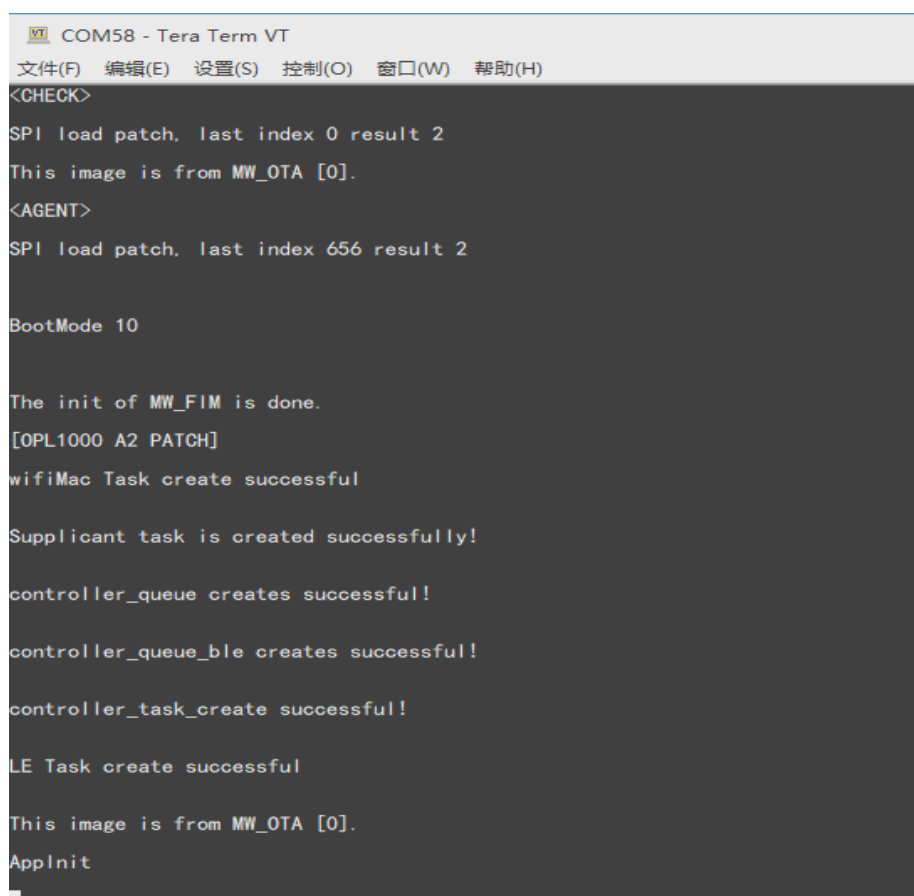
Figure 8: 下载透传功能固件



固件下载完成后 IO8/IO9 UART 端口将作为 AT 命令通信通道；IO0/IO1UART 端口作为 Debug log 信息的输出通道。

如图 Figure 9 所示，COM58 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，COM58 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 9: Debug 串口输出信息



```
COM58 - Tera Term VT
文件(F) 编辑(E) 设置(S) 控制(O) 窗口(W) 帮助(H)
<CHECK>
SPI load patch, last index 0 result 2
This image is from MW_OTA [0].
<AGENT>
SPI load patch, last index 656 result 2

BootMode 10

The init of MW_FIM is done.
[OPL1000 A2 PATCH]
wifiMac Task create successful

Supplicant task is created successfully!

controller_queue creates successful!

controller_queue_ble creates successful!

controller_task_create successful!

LE Task create successful

This image is from MW_OTA [0].
Applinit
```

设备启动后，使用 AT 命令

( at+cloudinfo= <ProductId>,<ProductKey>,<ProductSecret>,<DeviceName>,<DeviceSecret>  
) 写入阿里云五元码，使用 at+cloudinfo?命令查询，写入正确与否，完成后，重启设备，使写入的五元码生效。

### 3.5. 通过 AT 命令完成蓝牙配网

通过 AT 命令完成蓝牙配网，可以参考下面步骤完成

- 使用 AT+CWMODE =4 命令切换 BLEWIFI mode ,此模式下，才支持手机 APP 进行 WIFI 配网

Figure 10: 切换 BLEWIFI 模式

```
>at+cwmode=4  
  
OK
```

- 发送命令 AT+BLECAST=1 开启蓝牙广播

Figure 11: 开启蓝牙广播

```
>at+blecast=1  
+BLECAST:1,0  
  
OK  
+BLECAST:BLE ENTER ADVERTISING
```

- 使用云智能 APP 扫描蓝牙，完成蓝牙配网，如图

Figure 12: 阿里配网



### 3.6. 通过 AT 命令完成阿里云连接

- 配网完成后(注意：在 log 口出现 WIFI CONNECTION · WIFI GOT IP 时进行阿里云连接，在上图中进行连接网后还没有到达百分百，就要发送命令 `at+cloudconn` 连接云)，发送 `at+cloudconn` 命令完成阿里云连接，可以通过 `at+cloudconnstatus` 查看阿里云的连线状态，同时也可以云智能 APP 上查看是否跳转的自定义的应用操作界面，若是则表示云连接成功。

Figure 13: 云连接

```
COM160 - Tera Term VT
File Edit Setup Control Window Help
Switch: AT UART
>
>at+blecast=1
+BLECAST:1,0
OK
+BLECAST:BLE ENTER ADVERTISING
+BLECONN:PEER CONNECTION
WIFI CONNECTED
WIFI GOT IP
>
>at+cloudconn
OK
>
>+BLECONN:PEER DISCONNECTION
+BLECAST:BLE EXIT ADVERTISING
```

- 如需进入睡眠，并使用 GPIO 脚位高电平唤醒，可以通过 `at+sleep=1,23` (23 代表 GPIO #23 脚位) 命令实现操作。

Figure 14: 进入睡眠

```
>at+sleep=1,23
OK
<
```

### 3.7. 通过 AT 命令完成数据上报及下发

如需要设备往云端上报数据，可以使用 AT 命令 `at+cloudpub=0,0,"<data>"` 实现，注意：

`<data>`：传送给 ALI 云 Server 的 message data。字符，前面要加上 \

`at+cloudpub=0,0,"{"ContactState":1,\"BatteryPercentage":100,\"RSSI":-57}"`

当服务器有数据下发时，设备端则会收到下发的数据，格式为

+RCVFROMSRV:<recv type>,<data>

其中，<recv type>=PUB：收到的资料是属于由 Server 收到 PUBLISH 后 Reply 过来的。

Figure 15: 数据上报和下发

```
>+RCVFROMSRV:PUB,{"LedLightSwitch":0}
+RCVFROMSRV:PUB,{"LedLightSwitch":1}
+RCVFROMSRV:PUB,{"LedLightSwitch":0}
+RCVFROMSRV:PUB,{"LedLightSwitch":1}

>

>at+cloudpub=0,0,"{"ContactState":1\,"BatteryPercentage":100\,"RSSI":-57}"
OK
+RCVFROMSRV:REPLY,2,200,{}
```



## 4. 基于 ALI 透传范例开发应用

### 4.1. 修改 Ali IOT 设备模型

ALI IOT 设备模型可以根据需要定义，同时在阿里服务器选择对应的产品类型，如门磁传感器，LED 等  
详细参考链接：[https://github.com/Opulinks-Tech/OPL1000A2-Sensor-Device-Reference-Code-Ali-Cloud-with-MQTT/tree/master/doc/OPL1000\\_ali\\_key&app\\_create\\_guide.pdf](https://github.com/Opulinks-Tech/OPL1000A2-Sensor-Device-Reference-Code-Ali-Cloud-with-MQTT/tree/master/doc/OPL1000_ali_key&app_create_guide.pdf)

### 4.2. 基于已有范例增添功能

#### 4.2.1. 增加外设

用户可以根据需要添加需要的功能，以开关为例。  
在文件 hal\_pin\_config\_project\_transparent.h 定义使用的 IO 口，如图，并在 ali\_linkkitsdk\_impl.c 文件中定义函数功能。

Figure 16: 配置外设 IO

```
#define HAL_PIN_TYPE_IO_21  PIN_TYPE_GPIO_OUTPUT_LOW           // PIN_TYPE_NONE
// PIN_TYPE_GPIO_INPUT
// PIN_TYPE_GPIO_OUTPUT_LOW
// PIN_TYPE_GPIO_OUTPUT_HIGH
// PIN_TYPE_UART0_RX
// PIN_TYPE_UART1_RTS
// PIN_TYPE_UART1_TX
// PIN_TYPE_I2C_SDA
// PIN_TYPE_SPI2_IO_1
// PIN_TYPE_SPI2_IO_MISO
// PIN_TYPE_PWM_2
// PIN_TYPE_UART_APS_RX
// PIN_TYPE_UART_MSQ_TX
// PIN_TYPE_ICE_M3_CLK
// PIN_TYPE_ICE_M0_DAT
```

#### 4.2.2. 云端配置

登录阿里云，进入对应项目的功能定义界面，在功能定义界面添加功能，如为标准功能，则在标准功能下直接点击添加功能按键，然后进入其他类型搜索是否有需要的功能。若无标准功能，则在自定义功能下添加功能。

Figure 17: 标准功能定义

标准功能 ●

导入物模型 查看物模型 添加功能

功能类型	功能名称	标识符	数据类型	数据定义	操作
属性	门磁状态 <span>必选</span>	ContactState	bool (布尔型)	布尔值: 0 - 关闭 1 - 打开	编辑

Figure 18: 添加标准功能

选择功能: 全选添加

门磁传感器 其他类型

请输入功能名称或适用类别 搜索

AB线电压 属性

标识符: Uab  
适用类别: MultifunctionElectricityMeter

ADAS开关 属性

标识符: ADASSwitch  
适用类别: SmartRearviewMirror

AI事件 事件

标识符: AIEvent  
适用类别: HIKVISIONEdgeServer

功能添加完成后，进入人机交互界面，添加手机 APP 端显示或者控制的界面

Figure 19: 配置云智能 APP 操作界面



### 4.2.3. 连接阿里云的工作原理

在配网成功并获得 IP ( 触发 BLEWIFI\_CTRL\_EVENT\_BIT\_GOT\_IP 事件 ) 后 , OPL1000 会进入连云命令等待状态 ( 注意手机 APP 端超时 ) , 当连接 AT 命令时间触发后 , 此时会初始化云操作 , 完成云连接状态的更新。

Figure 20: 连云事件触发

```
{
#ifdef __BLEWIFI_TRANSPARENT__
    if (true == BleWifi_Ctrl_EventStatusGet(BLEWIFI_CTRL_EVENT_BIT_CLOUD_ENABLE))
    {
#endif
}
```

### 4.3. 工程 Heap Size 调整

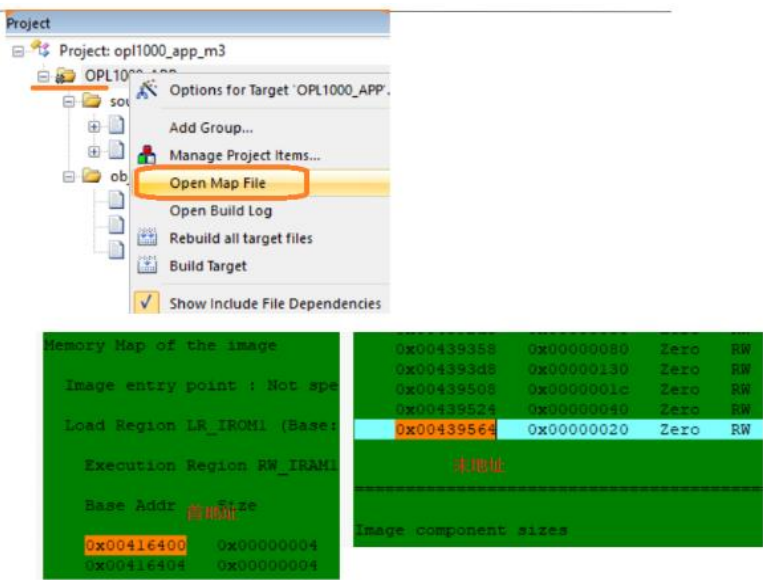
随着功能的不断加入 , 应用程序可能会碰到由于代码的不断增加 , 导致空间不够而引起的宕机问题.用户可以参考本节的内容 , 对 scatter file, heap size 做适当调整 , 尝试着解决内存不够问题。

( 1 ) 在调整 scatter file 之前，需要在 map 文件获取首末地址，方法如下：

- 用 keil 打开相应的工程
- 右键点击工程,选择'Open Map File'选项
- 在打开的.map 文件中获取 image 在 Memory Map 中的首地址和末地址。

在该例子中，首地址是： 0x00416400 ，末地址是： 0x00439564 。如下图所示。对末地址以 4k 或其它值(1k,512B 等)为单位向上取整，为 0x0043a000.

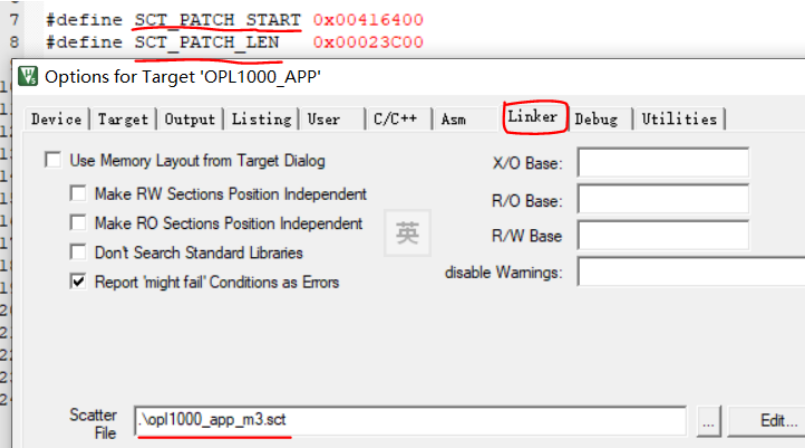
Figure 21: 从 map 文件中获取首末地址



( 2 ) 在 Memory Map 中获取首，末地址后，就可根据它们对 scatter file 做相应调整，方法如下：

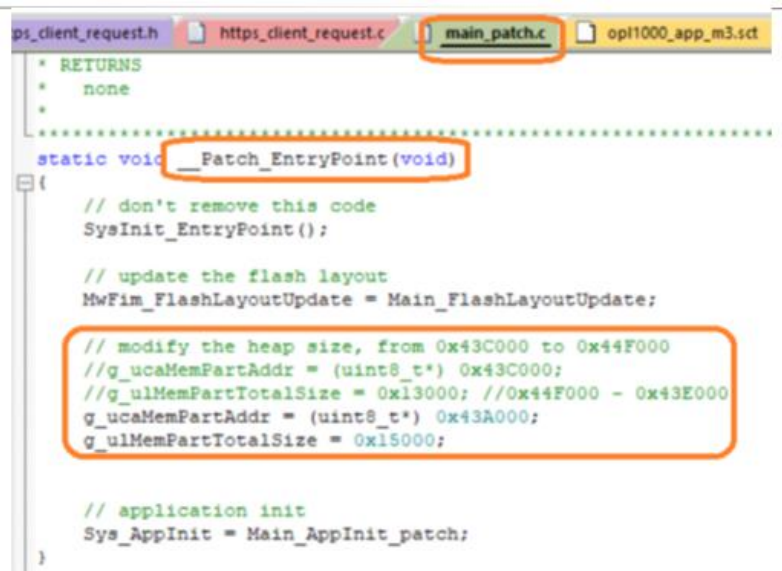
- 右键点击工程,选择 'Options' 选项,选择 'Linker' ，再点击 'Edit' scatter file,如下图所示
- 保持 SCT\_PATCH\_START 的值不变，而 SCT\_PATCH\_LEN 则可以改为末地址 - 首地址的值，如在该例中，大小 = 0x0043a000 - 0x00416400 = 0x00023C00 。

Figure 22: 更新 SCT file 中的 size



- ( 3 ) 在修改 scatter file 文件后，就可根据它设置 Heap 的起始地址，并计算出它的大小，方法如下：
- Heap 的设置通常都放在工程的主文件 main\_patch.c 文件的 \_\_Patch\_EntryPoint() 中进行；
  - Heap 的起始地址可以设置为末地址，在该例中也就是的 0x0043a000。大小 = 0x44F000 - 末地址，对于下图中的例子，就是，大小 = 0x44F000 - 0x0043a000 = 0x15000

Figure 23: 更新 heap size 相关的信息



4.4. 低功耗设置和验证

在 blewifi\_configuration.h 文件中定义了两个跟低功耗有关的宏： BLEWIFI\_WIFI\_DTIM\_INTERVAL 和 BLEWIFI\_COM\_RF\_POWER\_SETTINGS 宏。

其中，可以通过控制是否使能 smart sleep 模式，它的默认值为 1，即使能的。

Figure 24: smart sleep 相关的宏定义

```

}/*
RF Power


|                       | BLE Low Power | BLE High Power |
|-----------------------|---------------|----------------|
| WIFI Low power        | 0x00          | 0x0F           |
| WIFI Low power + 2 DB | 0x20          |                |
| WIFI High power       | 0xB0          | 0xFF           |
| WIFI High power +2 db | 0xD0          |                |
| WIFI High power +3 DB | 0xE0          |                |


*/
#define BLEWIFI_COM_RF_POWER_SETTINGS (0x20)

```

用户可根据需要自行调用以下接口函数(在 ps\_public.h 文件中定义)：

```
void ps_smart_sleep(int enable);
```

BLEWIFI\_WIFI\_DTIM\_INTERVAL 用于设置 DTIM 的默认值，该值越大，功耗越低，响应时间也相应加长。用户可根据需要自行调用以下接口函数(在 blewifi\_wifi\_api.h 文件中定义)：

```
int BleWifi_Wifi_SetDTIM(uint32_t value);
```

BLEWIFI\_COM\_RF\_POWER\_SETTINGS 用于设置 BLE 和 WIFI 模块的功耗模式的默认值，默认值为 0x00，用户可根据需要自行调用以下接口函数(在 blewifi\_common.h 文件中定义)：

```
void BleWifi_RFPowerSetting(uint8_t level);
```

4.5. 扩充 AT Cmd

OPL1000 阿里云透传主要由主控 MCU 通过 AT 命令完成数据指令，用户可以根据需要定义 AT 命令。用户修改的 AT 命令在 app\_at\_cmd.c 文件中实现，在 g\_taAppAtCmd 变量中定义，每个条目对应一个 AT 命令。

Figure 25: AT 命令实现函数映射表

```
at_command_t g_taAppAtCmd[] =
|{
|  { "at+readfim",      app_at_cmd_sys_read_fim,      "Read FIM data" },
|  { "at+writefim",     app_at_cmd_sys_write_fim,     "Write FIM data" },
|  { "at+dtim",         app_at_cmd_sys_dtim_time,     "Wifi DTIM" },
|  { "at+cloudinfo",    app_at_cmd_sys_cloud_info,    "Cloud Info" },
|  #if (WIFI_OTA_FUNCTION_EN == 1)
|  { "at+ota",          app_at_cmd_sys_do_wifi_ota,     "Do Wifi OTA" },
|  #endif
|  #ifdef BLEWIFI_TRANSPARENT
|  { "at+blecast",      app_at_cmd_sys_ble_cast,      "Ble Advertise" },
|  { "at+blecastparam", app_at_cmd_sys_ble_cast_param, "Ble Advertise Param" },
|  { "at+blests",       app_at_cmd_sys_ble_sts,       "Ble Status" },
|  { "at+wifists",      app_at_cmd_sys_wifi_sts,      "Wifi Status" },
|  { "at+cloudconn",    app_at_cmd_sys_cloudconn,    "Cloud Do Connect" },
|  { "at+clouddisconn", app_at_cmd_sys_clouddisconn, "Cloud Do Disconnect" },
|  { "at+cloudconnstatus", app_at_cmd_sys_cloudconnstatus, "Cloud connect status" },
|  { "at+cloudpub",     app_at_cmd_sys_cloudpub,     "Cloud Publish Topic" },
|  #endif
|  { NULL,             NULL,             NULL},
|};
```

4.6. 验证添加的功能

打开串口工具，在串口工具输入已经添加的 AT 命令，看是否符合预期，若能返回 OK，则说明 AT 命令没有问题。

1. 手机云智能 APP 验证：

用户可以根据自己的功能定义，将所需功能添加到云智能 APP 界面，根据手机 APP 上对应控件变化，判断功能，如下图添加的小夜灯开关功能

Figure 26: 新增功能控件显示



2. 通过 Debug 串口打印信息验证：

OPL1000 模块的 IO01 分别接上 UART 转板的 Rx,Tx，打开串口工具，连接云成功后，串口会打印云端上报以及下发的数据信息，如下

Figure 27: AT 命令验证反馈

```
>at+cloudconnstatus  
+CLOUDCONNSATUS:CLOUD DISCONNECTED  
  
OK
```



## 5. 进阶：ALI SDK 更新

### 5.1. Ali MQTT 收发 Buffer Size 调整

本例程使用的阿里 SDK 库的版本是 3.0.1.它所在的目录是

APS\_PATCH\middleware\third\_party\aliyun\_3.0.1。在 src\mqtt\impl\iotx\_mqtt\_config.h 文件中有定义了 IOTX\_MC\_TX\_MAX\_LEN 和 IOTX\_MC\_RX\_MAX\_LEN 两个宏，分别用于指定发送和接收消息的 buffer 的最大长度。它们的默认值为 512.

Figure 28: Ali 收发 buffer size 的设置

```
77  #ifndef IOTX_MC_TX_MAX_LEN
78      #define IOTX_MC_TX_MAX_LEN          (512)
79  #endif
80
81  #ifndef IOTX_MC_RX_MAX_LEN
82      #define IOTX_MC_RX_MAX_LEN          (512)
83  #endif
```

用户在做二次开发，为设备规划好收发消息的主题和数据包的格式时，可根据实际情况调整这两个 buffer 的大小。每次修改过 buffer 大小后，需要重新编译阿里库，以使改动生效。

## 6. FAQ

- i. 运行 ALI 云透传例程，云智能 APP 端找不到设备？
- 确认 OPL1000 设备有发送蓝牙广播信号,即成功执行 at+blecast=1 命令
  - 保证连接的 WIFI 信号已经接入外网
  - 确认写入阿里五元码后，有无重启 OPL1000 设备，使其生效
  - 检查工程文件 blewifi\_configuration.h 中蓝牙广播间隔

Figure 29: 蓝牙广播间隔配置

```
/* Advertisement Interval Calculation Method:
1000 (ms) / 0.625 (ms) = 1600 = 0x640
*/
#define BLEWIFI_BLE_ADVERTISEMENT_INTERVAL_MIN 0x64 // For 100 ms
#define BLEWIFI_BLE_ADVERTISEMENT_INTERVAL_MAX 0x64 // For 100 ms

/* For power saving
0xFFFF is deifined 30 min in dirver part
*/
#define BLEWIFI_BLE_ADVERTISEMENT_INTERVAL_PS_MIN 0xFFFF // 30 min
#define BLEWIFI_BLE_ADVERTISEMENT_INTERVAL_PS_MAX 0xFFFF // 30 min
```

- ii. 云智能 APP 成功配网后，连接阿里云超时？
- 检查设备端是否发送连云请求，即 AT 命令 at+cloudconn
- iii. 云连接成功后，无法收到云端下发及上报的数据
- 确保工程中写入的标识符（如 ContactState）是否与阿里云端功能定义的标识符同步
  - 确认传送阿里云的消息数据，特殊符号有无加上转义符号\

## CONTACT

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