ALIBABA CLOUD

阿里云

IoT物联网操作系统
AT命令WiFi模组固件用户手册

文档版本：20210427

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.基于阿里云直连WiFi固件MCU使用示例

本文介绍如何在MCU侧添加AT连接程序，使用阿里云之连WiFi固件，实现快速入云与数据上报。

获取源码

我们为用户提供了通MCU对接模组的示例代码。在该示例代码中，用户可以了解如何使用模组完成，包括模组配置、联网、属性上报、处理服务等调用。

将示例加入AliOS Things3.0源码中

示例代码基于AliOS Things 3.0源码进行构建，用户下载源码后，需要将源码添加至AliOS Things 3.0源码中才能进行编译使用。

Step 1: 将以上文件解压后，放置于app/example目录下

```
mv wifi_at rel_3.0.0/app/example/
```

wifi_at_app的目录结构如下

```
.
├── aos.mk
├── atcmd_util.c
├── Config.in
├── README.md
├── include
│   ├── atcmd_config.h
│   └── atcmd_util.h
└── wifi_at.c
```

Step 2: 更新app/example/config.in

在app/example/Config.in的choice选项内添加，以上Config.in的source与配置

```
source "app/example/wifi_at/Config.in"
if AOS_APP_WIFI_AT_APP
  config AOS_BUILD_APP
    default "wifi_at_app"
endif
```

Step 3: 编译验证

以stm32f103rb-nucleo为目标版进行编译，命令如下

```
aos make wifi_at@stm32f103rb-nucleo -c config
aos make
```

示例介绍

示例入口为 `wifi_at.c` 的 `application_start` 函数，并在该函数内运行主逻辑。本节按照运行步骤进行介绍。

Step 1：配置MCU与模组

```
int application_start(int argc, char *argv[])
{
  ...
  /* 设置MCU log等级 */
  aos_set_log_level(AOS_LL_DEBUG);
  /* 初始化MCU AT设备驱动，例如UART参数、注册AT前缀处理回调 */
  at_device_init();
  /* 重置模组的连接 */
  at_device_reset();
  /* 等待模组断开 */
  aos_msleep(1000);
  /* 配置模组四元组信息 */
  at_device_config(DEMO_PRODUCT_KEY,
           DEMO_DEVICE_NAME,
           DEMO_DEVICE_SECRET,
           DEMO_PRODUCT_SECRET);
  ...
}
```

Step 2：网络连接

```
int application_start(int argc, char *argv[])
{
  ...
  /*
  * 网络连接
  * 输入参数: WiFi AP的ssid, password
  * 如果ssid与password为空, 使用手机一键配网
  */
  do {
    ret = at_device_connect(NULL, NULL);
    if (ret >= 0) {
      break;
    }
    /* 循环等待网络连接 */
    LOGI(TAG, "AT device connect Cloud failed! retry...\n");
    aos_msleep(5000);
  } while (1);
  ...
}
```

Step 3：主业务循环

```
int application_start(int argc, char *argv[])
{
  ...
  /* 主循环 */
  while (1) {
    count++;
    if (count % 5 == 0) {
      memset(buffer, sizeof(buffer), 0);
      /* 可自行修改上报物模型的内容 */
      len = snprintf(buffer, sizeof(buffer), "{\"Counter\": %d}", count);
      /* 上报物模型 */
      user_send_property(buffer, len);
    }
    /* 处理来自云端消息 */
    user_process_cloud_msg();
    aos_msleep(1000);
  }
  ...
}
```

其中，`user_send_property` 拼接AT指令并等待执行结果，其实现如下。

```
int user_send_property(const char *payload, int payload_len)
{
  int len = 0;
  char cmd_str[100] = { 0 };
  char *ptr = NULL;
  int pkid = -1;
  int devid = 0; /* master device default id */
  /*
   * 拼接AT指令，设置devid
   * e.g. AT+IDMPP=0,
   */
  len = snprintf(cmd_str, sizeof(cmd_str),
          "%s=%d,", "AT+IDMPP", devid);
  ptr = cmd_str + len;
  /*
   * 拼接AT指令,添加payload
   * e.g. {"LightSwitch":1} ==> "{\"LightSwitch\":1}"
   */
  len += atcmd_add_slash_quote(ptr, sizeof(cmd_str) - len, payload, payload_len);
  /*
   * 发送AT指令，等待packet ID
   */
  pkid = atcmd_send_wait_ret_val(cmd_str, len, true, WAIT_PACKET_ID,
                  SEND_WAIT_REPLY_TIMEOUT_MS);
  LOGI(TAG, "Sent Property Message ID: %d", pkid);
  return pkid;
}
```

Step 4：处理云端请求

云端下发的请求，首先由模组以AT字符串形式推送给MCU；MCU收到后，由在 at_device_init() 函数注册的回调 at response callback 进行预处理，并放入缓存队列中。用户在上述主循环中由函数 user_process_cloud_msg 进行处理。

```
void user_process_cloud_msg(void)
{
  ...
  /* 检查消息队列 */
  if(atcmd_tryfetch_msg(&req_msg) != 0) {
    return;
  }
  /* 根据消息类型进行分别处理 */
  switch(req_msg->type) {
    case SERV_SET_MSG:
      /*
       * 用户在此添加业务处理，例如打开灯
       */
      /* 回复云端 */
      res = user_answer_service(req_msg->dev_id, req_msg->serv_id, strlen(req_msg->serv_id),
                answer_payload, strlen(answer_payload), (void *)req_msg->msg_id);
      LOGI(TAG, "Answer Server ret: %d msg: %s", res, req_msg->payload);
      break;
    /* 添加更多消息处理 */
    /* case PROP_SET_MSG: */
    /* case OTA_INFO_MSG: */
    default:
      LOGI(TAG, "Unknown msg type");
      break;
  }
  /* 释放取出的消息 */
  aos_free(req_msg->payload);
  aos_free(req_msg);
}
```

2.阿里云IoT WiFi 固件模组说明书

为了让使用“MCU+WiFi模组”模式接入阿里云IoT的用户更加高效地完成设备接入工作，我们基于AliOS Things实现了一套对阿里云IoT平台接入流程高度抽象的WiFi固件。使得用户最少使用3个指令就可以完成设备的接入和设备数据上报。同时还提供可通过硬件配置进入的透传模式，在用户不便修改端侧设备代码时，也可以快速地将设备接入平台。我们还为用户提供通过OTA本地代理能力，使模组成为本地MCU升级时的OTA服务提供者，帮助本地设备实现固件升级。

快速开始

通过AT命令接入阿里云IoT平台

步骤1：设置设备四元组

```
AT+IDMAU="PRODUCT_KEY","DEVICE_NAME","DEVICE_SECRET","PRODUCT_SECRET"
```

注：四元组信息可以从阿里云物联网平台设备管理页面获取，具体步骤参考通过带有阿里云IoT直连固件的WiFi模组快速完成设备接入

如果设置成功，模组返回OK。

步骤2：一键配网

- 通过发送AT指令启动配网。

```
AT+IWSSTART=0
```

注：该步骤需要配合手机端“云智能”APP使用，具体使用方法参考通过带有阿里云IoT直连固件的WiFi模组快速完成设备接入。

如果模组进入配网状态，则返回OK。

当配网成功后，模组**自动连接阿里云**；连接成功后，模组返回

```
+IDMSTA:2
OK
```

注：可登录阿里云物联网平台设备管理页面查看对应设备在线状态。

- 如果有其他方式可以获得现场路由器的SSID和password也可以通过AT指令完成配网

```
AT+WJAP=ssid,password
```

- 将模组提供的配网按钮拉低3秒钟，也可以使模组直接进入配网模式。

步骤3：上报数据

连云成功后，用户可以根据业务需求向云端上报数据。模组提供以下上报命令：

属性上报

```
AT+IDMPP=0,"{\"PowerSwitch_1\":0}"
```

事件上报

```
AT+IDMEP=0,"Error","{\"ErrorCode\":0}"
```

标签上报

```
AT+IDMTP=0,"[{\"attrKey\":\"welcome\",\"attrValue\":\"hello,world\"}]"
```

透传数据上报

```
AT+IDMRP=0,0
{"hello"}
```

透传方式接入阿里云

模组提供透传接入方式：模组上电后，自动连接阿里云，并根据某引脚输入，在连云成功后自动进入透传模式。在这一过程中，MCU无需向模组发送AT指令。因此，该模式无需添加或修改MCU的发送逻辑。

首次使用

参考以上AT指令步骤，对模组的四元组信息、配网信息进行配置。这些信息将存于flash中，后续使用无需配置。

步骤1：上电时设置引脚输入

根据模组型号，对相关引脚进行配置。例如EM3080，上电时将PWM5拉低，模组连云成功后自动进入透传模式。

步骤2：透传数据上报

进入该模式后，MCU即可以向模组发送自定义格式的数据，模组转发至相应设备的thing/model/up_raw主题。

```
{"hello1"}
//或16进制数
01 02 03 5F AA
```

通过模组给MCU升级固件

模组提供给MCU升级固件的能力，通过两条AT指令与ymodem即可完成。

步骤1：上传MCU固件至云端

MCU的固件需要在生成md5时对magic字段进行修改，用以标识固件类型，脚本见附件。然后，在物联网平台上传MCU的OTA固件，上传步骤参考该链接。

步骤2：下载MCU固件至模组

模组连云成功后，在物联网平台页面选择MCU固件和对应设备，开始升级，操作步骤参考该连接。

下载成功后，模组输出固件版本与长度信息。

```
+OTAINFO:version_1,10240
```

MCU收到该提示信息后，即可以进行升级前的准备。MCU也可以进行主动查询

```
AT+OTAINFO?
```

步骤3：下载MCU固件至模组

MCU准备好升级后，向模组发送开始指令，

```
AT+OTASTART=0
```

模组返回OK后，进入ymodem server等待模式。

MCU启动ymodem client，即进入OTA文件传输模式。

已适配模组及引脚说明

EM3080

引脚说明

PIN	功能	描述
14	透传模式控制	上电时保持低电平，模组进入透传模式
23	配网控制	按键进入配网模式
19	指示灯	1.灯灭表示断网 2.1Hz慢闪指示模组在配网状态 3.常亮表示网络连接成 4.0.5Hz快闪表示收到数据

传输性能指标

参数	值	描述
单次数据上报长度	1KByte	适用属性上报、事件上报、透传上报；透传上报数据到达1KByte，将自动进行一次上报，后续数据将在下次上报中处理
透传超时上报时长	1Sec	超时后对已收到数据包进行一次上报
配网超时	600Sec	配网超时后，将自动退出配网
自动重连次数	3次	Wi-Fi连接成功后，连接云端失败时，将自动重连3次

AT命令列表

物联网平台连接配置参数操作

命令	描述	实现方式
AT+IDMDC	连接域名配置	强制
AT+IDMPA	物联网平台连接参数配置	强制
AT+IDMAU	鉴权配置	强制

设置连接域名和端口 +IDMDC

命令类型	命令格式	响应
测试命令	AT+IDMDC=?	+IDMDC: “mqtt_domain”，“mqtt_port”，“http_domain”，“http_port” OK
查询命令	AT+IDMDC?	+IDMDC:<mqtt_domain>,<mqtt_port>,<http_domain>,<http_port> OK
执行命令	AT+IDMDC=<mqtt_domain>,<mqtt_port>,<http_domain>,<http_port>	OK
参数说明	<mqtt_domain>：mqtt服务器的域名或者IP地址； <mqtt_port>：mqtt服务器的端口，默认1883； <http_domain>：http服务器的域名或者IP地址； <http_port>：http服务器的端口，默认为80	
返回值说明		
示例		
注意事项		

设备鉴权配置 +IDMAU

命令类型	命令格式	响应
测试命令	AT+IDMAU=?	+IDMAU:"productKey","deviceName","deviceSecret","productSecret" OK
查询命令	AT+IDMAU?	+IDMAU:<productKey>,<deviceName>,<deviceSecret>,<productSecret> OK

执行命令	AT+IDMAU=<productKey>,<deviceName>,<deviceSecret>,<productSecret>	OK 或者 +CME ERROR:<err>
参数说明	<productKey>,<deviceName>,<deviceSecret>,<productSecret>：设备四元组信息 <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

设置物联网平台连接参数 +IDMPA

命令类型	命令格式	响应
测试命令	AT+IDMPA=?	+IDMPA:“ParaTag”,“ParaValue” OK
查询命令	AT+IDMPA?	+IDMPA:<ParaTag>,<ParaValue> OK
执行命令	AT+IDMPA=<ParaTag>,<ParaValue>	OK 或者 +CME ERROR:<err>
参数说明	<ParaTag>：物联网平台连接参数名称，包含以下参数项； <ParaValue>：物联网平台连接参数值； <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

“REGION”	使用服务器区域: SHANGHAI，SINGAPORE,JAPAN,USSA_WEST,GERMANY,CUSTOM。
“DYN_REG”	动态注册，0: 不启用 1: 启用。
“RECV_PROP_EVENT_REPLY”	接收属性、事件上报回复，0: 不接收 1: 接收。
“SEND_PROP_SET_REPLY”	发送属性设置回复，0: 不发送 1：发送

配网AT指令

命令	描述	实现方式
AT+IWSSTART	开始配网	强制
AT+IWSSTOP	停止配网	强制
AT+WJAP	手动配网	强制
AT+WJAPD	删除配网信息	强制
AT+WJAPQ	查询配网信息	非强制

开始配网 +IWSSTART

命令类型	命令格式	响应	
测试命令	AT+IWSSTART=?	+IWSSTART:"type"	
查询命令	AT+IWSSTART?	+IWSSTART:<type> OK	
执行命令	AT+IWSSTART=<type>	OK 或者 +CME ERROR:<err>	
参数说明	<type>：配网类型：0: 一键配网；1: 其它 <err>：error代码，详见《AT command set for User Equipment (UE)》。		
返回值说明			
示例			AT+WSSTART OK

注：在发起一次新的配网前，需使用AT+IDMCLS命令断开当前模组与路由器的连接。

停止配网 +IWSSTOP

命令类型	命令格式	响应
测试命令	AT+IWSSTOP=?	OK
执行命令	AT+IWSSTOP	OK 或者 +CME ERROR:<err>
参数说明		

返回值说明	<err>：error代码，详见《AT command set for User Equipment (UE)》。
示例	

手动配网 +WJAP

命令类型	命令格式	响应
执行命令	AT+WJAP=<ssid>,<password>	OK 或者 ERROR
参数说明	<ssid>：WiFi AP SSID，最大长度32 <password>：WiFi密码，最大长度64	
返回值说明		
示例		

删除配网信息 +WJAPD

命令类型	命令格式	响应
执行命令	AT+WJAPD	OK 或者 ERROR
参数说明	无	
返回值说明		
示例		

配网信息查询 +WJAPQ

命令类型	命令格式	响应
执行命令	AT+WJAPQ	+WJAPQ:<ssid>,<password> OK 或者 ERROR
参数说明	<ssid>：WiFi AP SSID，最大长度32 <password>：WiFi密码，最大长度64	
返回值说明		
示例		

UART AT指令

命令	描述	实现方式
AT	AT测试	强制
AT+UARTS	UART参数设置	强制

AT测试

命令类型	命令格式	响应
执行命令	AT	OK 或者 +CME ERROR:<err>
参数说明	<err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		

UART参数配置 +UARTS

命令类型	命令格式	响应
测试命令	AT+UARTS=?	+UARTS:"baud_rate","data_width","stop_bits","parity","flow_control" OK
查询命令	AT+UARTS?	+UARTS:<baud_rate>,<data_width>,<stop_bits>,<parity>,<flow_control> OK
执行命令	AT+UARTS=<baud_rate>,<data_width>,<stop_bits>,<parity>,<flow_control>	OK 或者 +CME ERROR:<err>

参数说明 返回值说明 示例 注意事项	<baud_rate>：波特率，有效值：115200，921600，4800，9600，14400，19200，38400，57600，230400，460800； <data_width>：位宽，有效值：5，6，7，8，9； <stop_bit>：停止位，有效值：1，2； <parity>：奇偶校验，有效值：NONE，ODD，EVEN； <flow_control>：流控，有效值：NONE，CTS，RTS，CTSRTS <err>：error代码，详见《AT command set for User Equipment (UE)》。

物联网平台连接操作

命令	描述	实现方式
AT+IDMCON	建立连接	强制
AT+IDMCLS	断开连接	强制
AT+IDMSTA	连接状态	强制

连接 +IDMCON

命令类型	命令格式	响应
测试命令	AT+IDMCON=?	OK
执行命令	AT+IDMCON	OK 或者 +CME ERROR:<err>
参数说明 返回值说明 示例	<err>：error代码，详见《AT command set for User Equipment (UE)》。	

断开连接 +IDMCLS

命令类型	命令格式	响应
测试命令	AT+IDMCLS=?	OK
执行命令	AT+IDMCLS	OK 或者 +CME ERROR:<err>
参数说明		

返回值说明	<err>：error代码，详见《AT command set for User Equipment (UE)》。
示例	

查询连接状态 +IDMSTA

命令类型	命令格式	响应
测试命令	AT+IDMSTA=?	OK
查询命令	AT+IDMSTA?	+IDMSTA:<state> OK 当网络状态发生变化，模组也可以主动上报： +IDMSTA:<state> OK
参数说明	<state>：MQTT连接状态 0：物联网平台连接断开状态。 1：保留。 2：物联网平台连接连接状态。	
返回值说明		
示例		

物模型操作

命令	描述	实现方式
AT+IDMPP	属性上报	强制
+IDMPS	属性设置	强制
+IDMPG	属性获取	非强制
AT+IDMEP	事件上报	强制
+IDMSS	服务请求	强制
AT+IDMRP	透传数据上报	强制
+IDMRS	透传数据下发	强制

属性上报+IDMPP

命令类型	命令格式	响应

测试命令	AT+IDMPP=?	+IDMPP: “device_id”，“message”，“format”，“fragment_id” OK
查询命令	AT+IDMPP?	+IDMPP:<device_id>,<message>,[format],[fragment_id] OK
执行命令	AT+IDMPP=<device_id>,<message>,[format],[fragment_id]	+IDMPP:<packet_id> OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +IDMPP:<device_id>,<packet_id>,<code>,[reply_len],[reply_msg]
参数说明	<device_id>：设备ID； <message>：上报的消息体内容； <packet_id>：发布消息时返回的id； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，支持0。 如果该字段为1时，通信模组需要将HEX字符串转换成二进制数据格式，再执行发布操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 终端模组发布长消息前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果通信模组收到的数据包里面分包计数器大于0，它需要继续接收IDMPP命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行上报属性操作。 <code>：结果代码，详见“阿里云日志服务”； [reply_len]：回复消息长度，可选字段； [reply_msg]：回复消息，可选字段，如果存在，reply_len必须存在； <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

属性设置+IDMPS

命令类型	命令格式	响应

测试命令	AT+IDMPS=?	OK
查询命令	AT+IDMPS?	+IDMPS:<device_id>,<msg_len>,<message>,[format],[fragment_id] OK
		主动上报收到的消息： +IDMPS:<device_id>,<msg_len>,<message>,[format],[fragment_id] OK
参数说明	<device_id>：设备ID； <msg_len>：收到消息体的长度： <message>：收到消息体的内容； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，目前只支持0。 如果该字段为1时，终端模组需要将HEX字符串转换成二进制数据格式，再执行后续操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 通信模组将收到的长消息发给终端模组前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果终端模组收到的数据包里面分包计数器大于0，它需要继续接收IDMPS命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行后续操作。	
返回值说明		
示例		
注意事项		

属性获取+IDMPG

命令类型	命令格式	响应
测试命令	AT+IDMPG=?	OK
查询命令	AT+IDMPG?	+IDMPG:<device_id>,<msg_len>,<message>,[format],[fragment_id] OK

		主动上报收到的消息： +IDMPG:<device_id>,<msg_len>,<message>,[format],[fragment_id] OK
参数说明	<device_id>：设备ID； <msg_len>：收到消息体的长度： <message>：收到消息体的内容； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，0。 如果该字段为1时，终端模组需要将HEX字符串转换成二进制数据格式，再执行后续操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 通信模组将收到的长消息发给终端模组前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果终端模组收到的数据包里面分包计数器大于0，它需要继续接收IDMPG命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行后续操作。	
返回值说明		
示例		
注意事项		

事件上报 +IDMEP

命令类型	命令格式	响应
测试命令	AT+IDMEP=?	+IDMEP:“device_id”,“event_id”,“event_payload”,“format”,“fragment_id” OK
查询命令	AT+IDMEP?	+IDMEP:<device_id>,<event_id>,<event_payload>,[format],[fragment_id] OK

执行命令	AT+IDMEP=<device_id>,<event_id>,<event_payload>,[format],[fragment_id]	+IDMEP:<packet_id> OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +IDMEP:<device_id>,<packet_id>,<code>,<event_id>,[reply_len],[reply_msg]
参数说明	<device_id>：设备ID； <event_id>：事件标识； <event_payload>：事件内容； <packet_id>：发布消息时返回的id； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，目前只支持0。 如果该字段为1时，通信模组需要将HEX字符串转换成二进制数据格式，再执行发布操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 终端模组发布长消息前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果通信模组收到的数据包里面分包计数器大于0，它需要继续接收IDMEP命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行上报事件操作。 <code>：结果代码，详见“阿里云日志服务”； [reply_len]：回复消息长度，可选字段； [reply_msg]：回复消息，可选字段，如果存在，reply_len必须存在； <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

服务请求+IDMSS

命令类型	命令格式	响应
测试命令	AT+IDMSS=?	OK
查询命令	AT+IDMSS?	+IDMSS:<device_id>,<msg_len>,<message>,[format],[fragment_id] OK

		主动上报收到的消息： +IDMSS:<device_id>,<msg_len>,<message>,[format],[fragment_id] OK
参数说明	<device_id>：设备ID； <msg_len>：收到消息体的长度： <message>：收到消息体的内容； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，目前只支持0。 如果该字段为1时，终端模组需要将HEX字符串转换成二进制数据格式，再执行后续操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 通信模组将收到的长消息发给终端模组前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果终端模组收到的数据包里面分包计数器大于0，它需要继续接收IDMSS命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行后续操作。	
返回值说明		
示例		
注意事项		

透传数据上报 +IDMRP

命令类型	命令格式	响应
测试命令	AT+IDMRP=?	+IDMRP:“device_id”,“msg_len” OK “bin_message”
查询命令	AT+IDMRP?	+IDMRP:<device_id>,<msg_len> OK <bin_message>
	AT+IDMRP=<device_id>,<msg_len>	当msg_len大于0时， OK 当msg_len等于0时 >

执行命令	<bin_message>[+++]	+IDMRP:<packet_id> OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +IDMRP:<device_id>,<packet_id>,<code>,<event_id>,[reply_len],[reply_msg]
参数说明	<device_id>：设备ID; <msg_len>:上报的消息长度; <bin_message>：上报的消息体内容，二进制格式; [+++]:从二进制数据传输模式返回命令模式，仅当msg_len为0时存在; <packet_id>:上报消息时返回的id; <code>：结果代码，详见“阿里云日志服务”； [reply_len]：回复消息长度，可选字段; [reply_msg]：回复消息，可选字段，如果存在，reply_len必须存在; <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

透传数据下发+IDMRS

命令类型	命令格式	响应
测试命令	AT+IDMRS=?	OK
查询命令	AT+IDMRS?	+IDMRS:<device_id>,<msg_len> OK <bin_message>
		主动上报收到的消息： +IDMRS:<device_id>,<msg_len> OK <bin_message>

参数说明	<device_id>：设备ID; <msg_len>：收到消息体的长度： <bin_message>：收到消息体的内容，二进制格式；
返回值说明	
示例	
注意事项	

子设备管理操作

命令	描述	实现方式
AT+ISUBRP	子设备注册	强制
AT+ISUBRD	子设备注销	强制
AT+ISUBLGP	子设备登录	强制
AT+ISUBLGD	子设备登出	强制
+ISUBPS	网关允许添加子设备	强制

子设备注册 +ISUBRP

命令类型	命令格式	响应
测试命令	AT+ISUBRP=?	+ISUBRP:"productKey","deviceName","deviceSecret","productSecret" OK
查询命令	AT+ISUBRP?	+ISUBRP:<productKey>,<deviceName>,<deviceSecret>,<productSecret> OK
执行命令	AT+ISUBRP=<productKey>,<deviceName>,<deviceSecret>,<productSecret>	+ISUBRP:<device_id> OK 或者 +CME ERROR:<err>
参数说明	<productKey>,<deviceName>,<deviceSecret>,<productSecret>：子设备四元组 <device_id>：设备ID <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

子设备注销 +ISUBRD

命令类型	命令格式	响应
测试命令	AT+ISUBRD=?	+ISUBRD:<deviceID> OK
查询命令	AT+ISUBRD?	+ISUBRD:” device_id” OK
执行命令	AT+ISUBRD=<deviceID>	OK 或者 +CME ERROR:<err>
参数说明	<device_id>：设备ID; <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		

子设备登录 +ISUBLGP

命令类型	命令格式	响应
测试命令	AT+ISUBLGP=?	+ISUBLGP:<deviceID> OK
查询命令	AT+ISUBLGP?	+ISUBLGP:<deviceID> OK
执行命令	AT+ISUBLGP=<deviceID>	OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +ISUBLGP:<device_id>,<packet_id>,<code>,<event_id>,[reply_len],[reply_msg]

参数说明	<device_id>：设备ID； <err>：error代码，详见《AT command set for User Equipment (UE)》。 <packet_id>：发布消息时返回的id； <code>：结果代码，详见“阿里云日志服务”； [reply_len]：回复消息长度，可选字段； [reply_msg]：回复消息，可选字段，如果存在，reply_len必须存在；
返回值说明	
示例	

子设备登出 +ISUBLGD

命令类型	命令格式	响应
测试命令	AT+ISUBLGD=?	+ISUBLGD:<deviceID> OK
查询命令	AT+ISUBLGD?	+ISUBLGD:<deviceID> OK
执行命令	AT+ISUBLGD=<deviceID>	OK 或者 +CME ERROR:<err>
参数说明	<device_id>：设备ID； <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		

网关允许添加子设备 +ISUBPS

命令类型	命令格式	响应
测试命令	AT+ISUBPS=?	OK
查询命令	AT+ISUBPS?	+ISUBPS:<productkey>,<timeout> OK
		主动上报收到的消息： +ISUBPS:<productkey>,<timeout> OK

参数说明	<productkey>：产品key值； <timeout>：permit超时时间，单位second；
示例	+ISUBPS=“3a1PcArcq23G”,60 OK

设备标签操作

命令	描述	实现方式
AT+IDMTP	设备标签上报	强制
AT+IDMTD	设备标签删除	强制

设备标签上报 +IDMTP

命令类型	命令格式	响应
测试命令	AT+IDMTP=?	+IDMTP:“device_id”,“message”,“format”,“fragment_id” OK
查询命令	AT+IDMTP?	+IDMTP:<device_id>,<message>,[format],[fragment_id] OK
执行命令	AT+IDMTP=<device_id>,<message>,[format],[fragment_id]	+IDMTP:<packet_id> OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +IDMTP:<device_id>,<packet_id>,<code>,<event_id>,[reply_len],[reply_msg]

参数说明 返回值说明 示例 注意事项	<device_id>：设备ID； <message>：上报的消息体内容； <packet_id>：发布消息时返回的id； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，目前只支持0。 如果该字段为1时，通信模组需要将HEX字符串转换成二进制数据格式，再执行发布操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 终端模组发布长消息前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果通信模组收到的数据包里面分包计数器大于0，它需要继续接收IDMTP命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行发送操作。 <code>：结果代码，详见“阿里云日志服务”； [reply_len]：回复消息长度，可选字段； [reply_msg]：回复消息，可选字段，如果存在，reply_len必须存在； <err>：error代码，详见《AT command set for User Equipment (UE)》。

设备标签删除 +IDMTD

命令类型	命令格式	响应
测试命令	AT+IDMTD=?	+IDMTD:“device_id”，“message”，“format”，“fragment_id” OK
查询命令	AT+IDMTD?	+IDMTD:<device_id>,<message>,[format],[fragment_id] OK
执行命令	AT+IDMTD=<message>,[format],[fragment_id]	+IDMTD:<packet_id> OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +IDMTD:<device_id>,<packet_id>,<code>,<event_id>,[reply_len],[reply_msg]

参数说明 返回值说明 示例 注意事项	<device_id>：设备ID； <message>：上报的消息体内容。 <packet_id>：发布消息时返回的id； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，目前只支持0。 如果该字段为1时，通信模组需要将HEX字符串转换成二进制数据格式，再执行发布操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 终端模组发布长消息前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果通信模组收到的数据包里面分包计数器大于0，它需要继续接收IDMTD命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行发送操作。 <code>：结果代码，详见“阿里云日志服务”； [reply_len]：回复消息长度，可选字段； [reply_msg]：回复消息，可选字段，如果存在，reply_len必须存在； <err>：error代码，详见《AT command set for User Equipment (UE)》。

信息查询操作

命令	描述	实现方式
AT+IDMQ	信息查询	强制

信息查询 +IDMQ

命令类型	命令格式	响应
测试命令	AT+IDMQ=?	+IDMQ: “device_id”，“msg_type”，“message”，“format”，“fragment_id” OK
查询命令	AT+IDMQ?	+IDMQ:<device_id>,<msg_type>,<msg_type>,[message],[format],[fragment_id] OK

执行命令	AT+IDMQ=<device_id>,<msg_type>,[message],[format],[fragment_id]	OK 或者 +CME ERROR:<err> 如果输入合法，首先返回OK，然后主动上报结果。 +IDMQ:<device_id>,<msg_type>,<reply_len>,<reply_msg>
参数说明	<device_id>：设备ID; <msg_type>：消息类型，例如，timestamp、FOTA，由模组厂商事先制定； [message]：消息体内容,可选字段，默认为空； <packet_id>：发送消息时返回的id； [format]：消息体格式，0：一般字符串；1：HEX字符串。可选字段，目前只支持0。 如果该字段为1时，通信模组需要将HEX字符串转换成二进制数据格式，再执行发布操作； [fragment_id]：长消息体分包计数器。可选字段，默认为0。 终端模组发布长消息前，需要先设置分包计数器为最大分包数-1；然后每发送一个新的分包，计数器-1， 0表示这是最后一个分包。如果重发一个分包，分包计数器保持不变。 如果通信模组收到的数据包里面分包计数器大于0，它需要继续接收IDMQ命令，直到收到分包计数器为0，然后将收到的消息按顺序组包，再执行回复操作。 <reply_len>：回复消息长度； <reply_msg>：回复消息； <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		
注意事项		

MCU OTA AT指令

命令	描述	实现方式
AT+OTAINFO	MCU固件信息查询	强制
AT+OTASTART	MCU固件传输开始	强制

MCU固件信息查询

命令类型	命令格式	响应
测试命令	AT+OTAINFO=?	OK

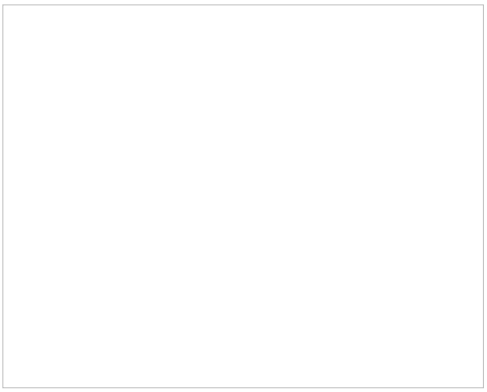
查询命令	AT+OTAINFO?	+OTAINFO:<version>,<image_size> OK 当MCU OTA完成后，模组也可以主动上报： +OTAINFO:<version>,<image_size> OK
参数说明	<version>：MCU 固件版本； <image_size>：MCU 固件大小，单位字节。	
返回值说明		
示例		

MCU固件传输开始

命令类型	命令格式	响应
测试命令	AT+OTASTART=?	+OTASTART=“type” OK
查询命令	AT+OTASTART?	+OTASTART=<type> OK
执行命令	AT+OTASTART=<type>	OK 或者 +CME ERROR:<err>
参数说明	<type>：升级方式，0: ymodem <err>：error代码，详见《AT command set for User Equipment (UE)》。	
返回值说明		
示例		

3.阿里云直连WiFi固件升级方式

本文介绍如何烧录阿里云直连WiFi固件的升级方法。本文以**银尔达emw3080**开发为示例进行介绍，如下图所示。



本文包括：

1.**首次升级**：原厂固件 ==> 阿里云直连WiFi固件

2.**后续升级**：旧阿里云直连WiFi固件 ==> 新阿里云直连WiFi固件

所需工具

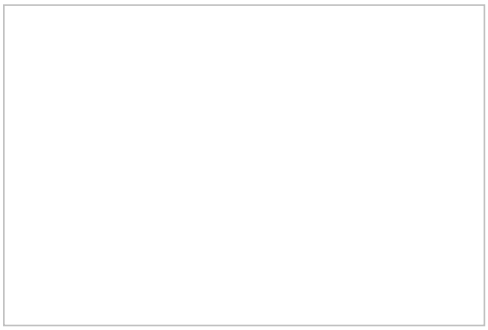
- Windows PC (win7/win10)
- USB连接线
- 串口调试软件

首次升级

本部分内容为从**庆科原厂固件**升级至**阿里云直连固件**的操作方式。如果已为阿里云直连固件请参考“后续升级”。

步骤1：进入Boot模式

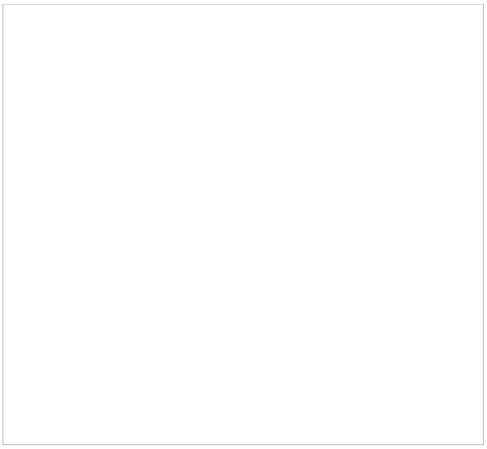
将模块的BOOT引脚拉低，并使用UART 1进行升级。对应于银尔达emw3080，需按以下截图方式，对CONFIG JMP和UART JMP进行调整。



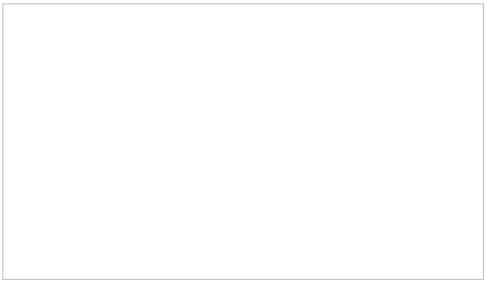
然后，连接USB口至PC。

步骤2：进入烧录等待

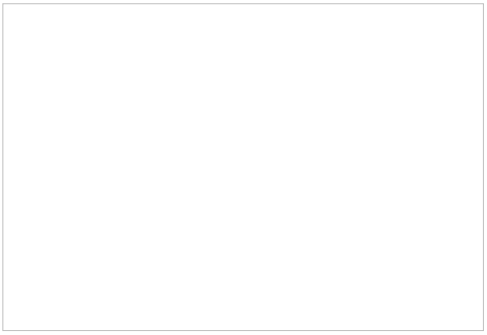
打开串口调试软件工具，以SecureCRT为例。选择对应串口，波特率设置为921600bps。



进入串口窗口，按reset键后进入以下界面。

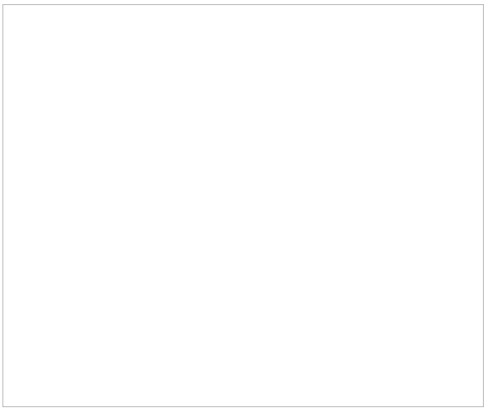


输入 `4 -dev 1 -start 0x0 -end 0x160000` ，设备进入烧录等待。



步骤3: 烧录

选择要烧录的.bin固件文件，打开 Transfer --> Send Ymodem 界面，在 PC 目录中选择要烧录的固件文件后，开始烧录。



步骤4：验证固件

保持USB连接，将Boot引脚恢复，波特率设为115200bps，按reset键。在串口窗口，发送以下“认证测试”命令

```
AT+IDMAU=?
```

预期回复为

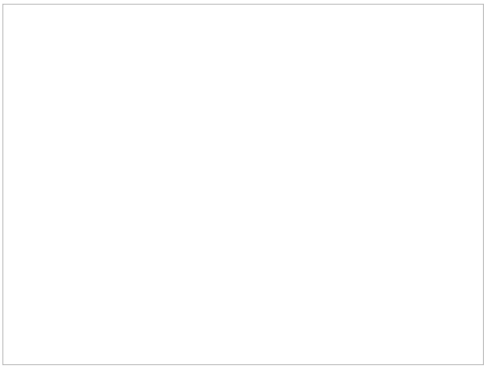
```
AT+IDMAU="PRODUCT_KEY","DEVICE_NAME","DEVICE_SECRET"
```

后续升级

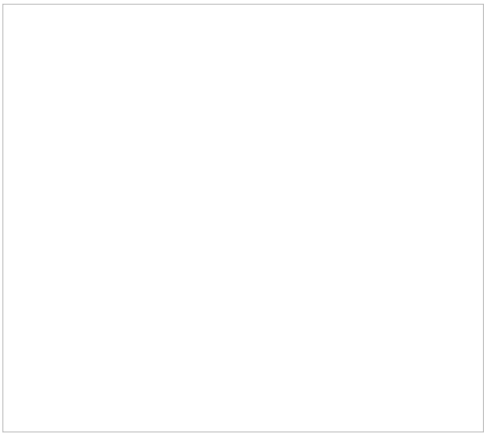
后续升级阿里云直连WiFi固件，有两种方式：使用阿里云物联网平台对模组进行OTA升级；本地使用模组二级Boot进行ymodem进行升级。第一种方式，可以参考阿里云官网文档，此处不再赘述；本文主要介绍第二种方式。

步骤1：进入二级Boot模式

将保持Boot高电平，并使用Debug UART进行升级。对应于银尔达emw3080，需按以下截图方式，对CONFIG JMP和UART JMP进行调整。



打开串口调试软件工具，选择对应串口，波特率设置为115200bps。在输入窗口中，按住W键，按Reset键进入二级Boot模式。



步骤2：进入烧录等待

按“1”进入，进入烧录参数配置。

输入0x19000，进入烧录等待

步骤3: 烧录

选择要烧录的ota.bin固件(非.bin固件)文件，打开 Transfer --> Send Ymodem 界面，在 PC 目录中选择要烧录的固件文件后，开始烧录。

步骤4: 验证固件

保持USB连接，将UART JMP引脚恢复至U1_RX、U1_TX，波特率设为115200bps，按reset键。在串口窗口，发送以下“认证测试”命令

```
AT+IDMAU=?
```

预期回复为

```
AT+IDMAU="PRODUCT_KEY","DEVICE_NAME","DEVICE_SECRET"
```

4.通过带有阿里云IoT直连固件的WiFi模组快速完成设备接入

阿里云IoT直连固件，是运行于生态伙伴WiFi模组中的软件产品，透过该产品可以使希望快速将设备接入阿里云IoT平台的用户，通过简单的几条AT指令即可完成设备连云，数据上报，接收来自云端的数据的功能，结合阿里云IoT平台的其他服务，快速搭建从硬件接入到物联网应用的完整产品。

适应场景

阿里云IoT直连固件，通过串口AT指令为用户提供相应服务，较为适合MCU+WiFi模组的智能设备场景。

接入示例

接下来通过一个实际的例子，完成一个复合灯具类设备接入阿里云IoT智能生活平台，并实现APP对设备状态进行显示及控制的功能。为了快速地讲述整个过程，本示例中不涉及MCU侧的具体编程，主要说明相关业务流程及AT指令，采用PC上的串口调试工具，模拟设备数据的发送和接收，用户可以通过将演示中的命令移植到MCU代码中的方式来实现。后续也将会推出内置AliOS Things操作系统的硬件产品如何使用带有直连固件模组的说明。

step 1：云端创建产品

在阿里云IoT平台中，我们将同一型号的设备定义为一个产品。在进行任何物联网应用开发之前，我们需要根据将要接入的设备的相关特性，在云端设备控制台中先创建一个产品，并且为产品配置相关的功能。

登入iot.aliyun.com 在“特色行业”标签中选择“生活物联网（飞燕）”。

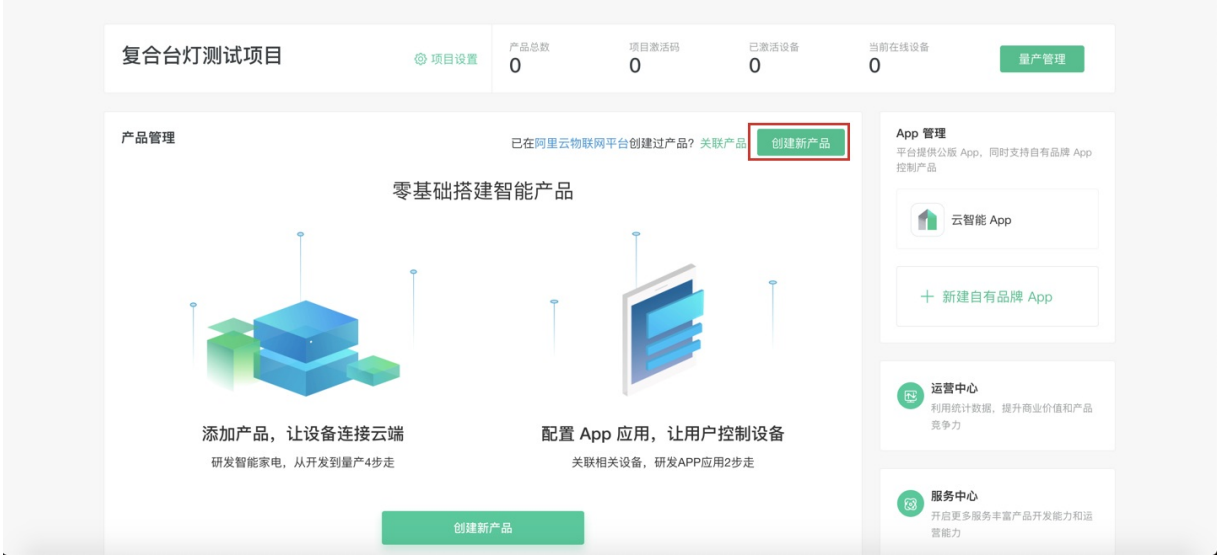
创建项目

登录到生活物联网平台。平台以项目为维度为用户提供不同硬件设备之间的数据隔离，所以我们需要“创建项目”

创建一个叫“复合台灯测试项目”的新项目

创建产品

进入到项目控制台后，我们选择“创建产品”



在弹框中填写要新建的产品的相关信息

1. 给产品命名，可以是产品的型号或者其他规则的名称。
2. 生活物联网平台为设备厂商已经预置了诸多的产品类型模板，用户可以根据实际的情况进行选择，不用从头开始。本示例中选择“灯”这类型。
3. 选择节点类型，设备为直连IoT平台的设备，还是作为其他设备的网关接入平台。
4. 物理连接方式选择，这里我们使用WiFi
5. 数据格式。平台支持ICA联盟所制定的Alink协议规范和透传数据。本演示中采用ICA标准数据格式。

在完成产品的初步定义后，平台会引导用户完成接下来的产品定义工作。

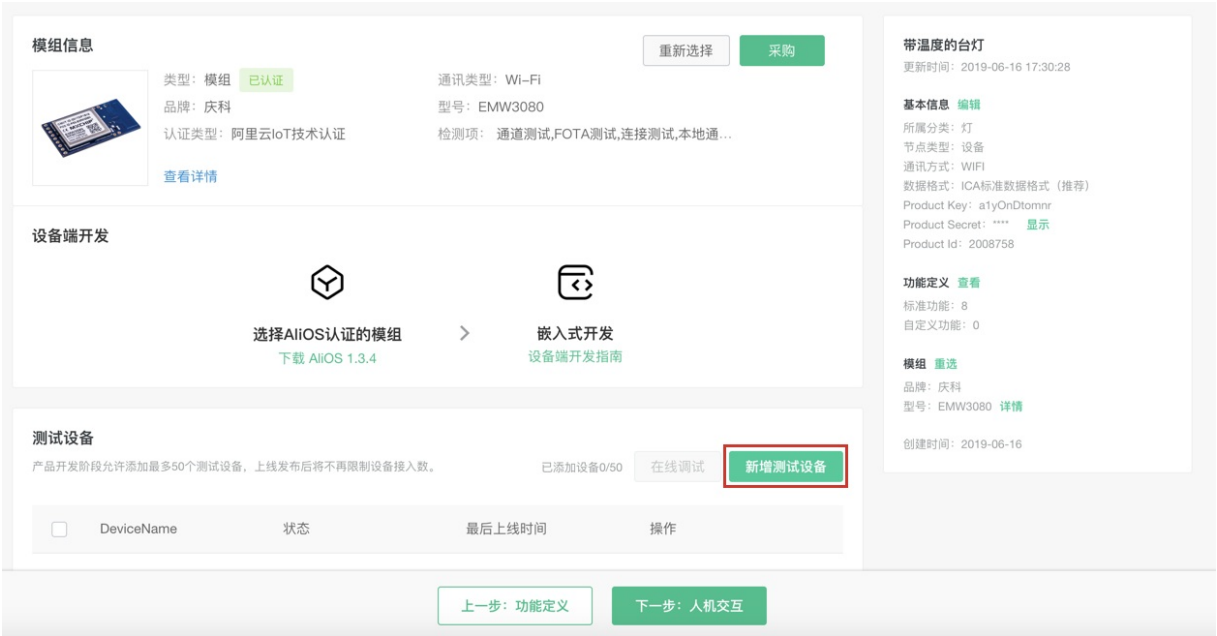
首先需要进行“功能定义”，即该硬件产品有些什么能力需要在云端及APP进行展示，或到控制。在默认情况下，我们可以看到平台已经为设备定义了“主灯开关”属性和“故障上报”事件，由于我们希望演示的是一个复合设备，所以我们可以给他增加一个“室内温度”。我们点击“标准功能”后面的“新增”按钮，在弹出的对话框中选择要新增的属性。

新增完成后，标准功能列表中会增加一个属性：

到此我们的产品定义完成，请留意此处的“标识符”字段，在后续的设备通信中将会使用。我们点击“下一步：设备调试”。平台会列出生态合作伙伴提供的模组供用户进行参考选择，用户选择自己使用的模组即可，对于实际模组并没有在此列表中的情况，任意选择一个类似模组即可。模组的选择将不会影响后续的调试。

创建测试设备

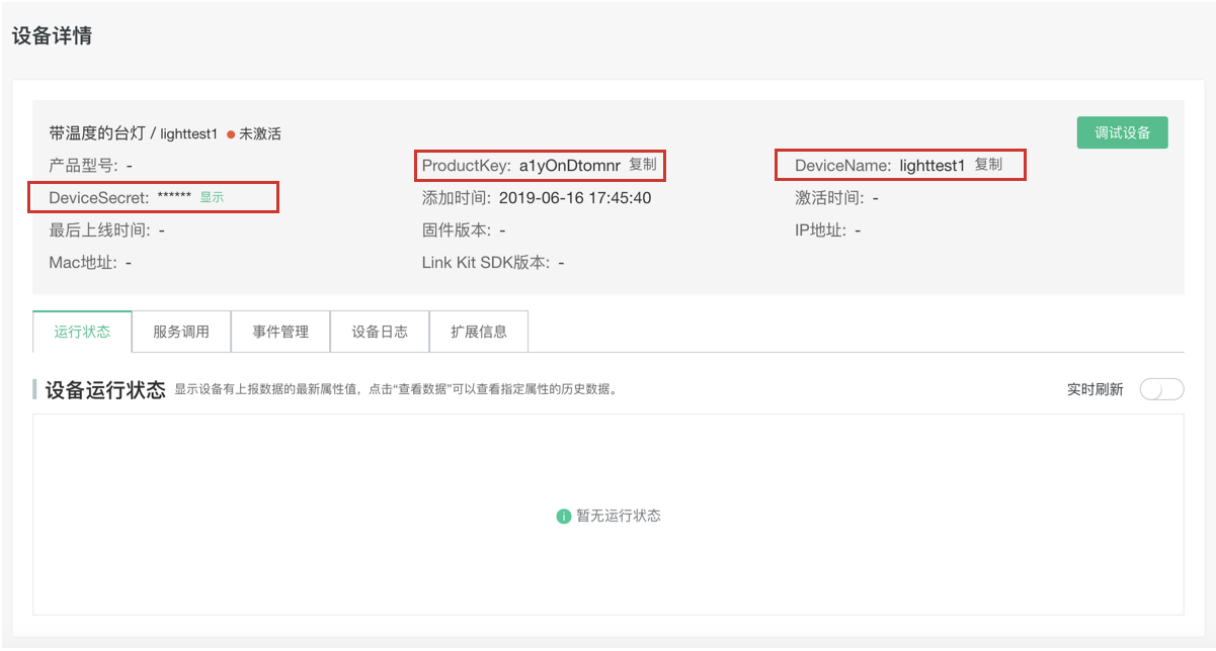
在设备调试页，我们可以真正采购刚刚选择的模组，平台也在用户量产前为用户提供了测试设备额度进行测试开发。我们点击“新增测试设备”按钮。



在弹窗中为设备命名。此名字即为新增的测试设备的名字，如果用户需要添加多个测试设备，请保证测试设备名称间不可相同。

完成新增后，可以看到在测试设备列表中出现了一个未激活的设备

点击查看按钮我们可以看到设备详细信息，其中设备的激活凭证“ProductKey”，“DeviceName”，“DeviceSecret”以及“Product Secret”将在后续的模组连接中用到，请读者注意在此处可以找到相关信息。

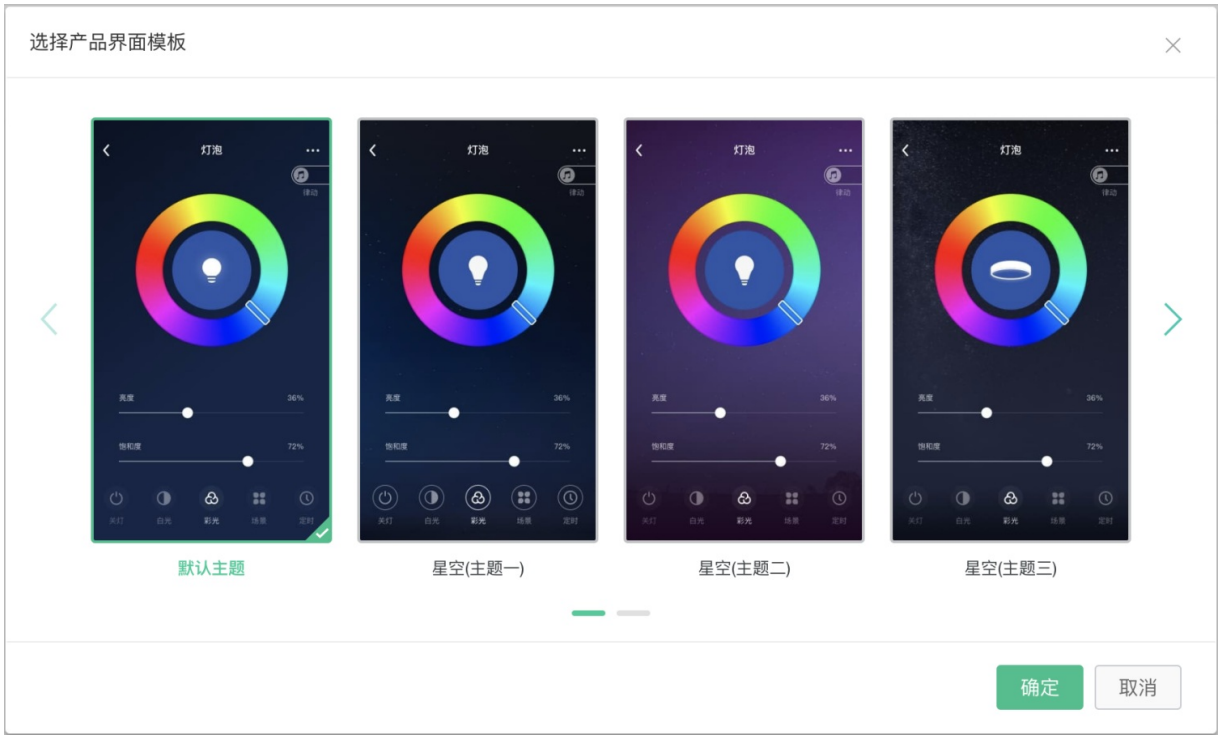


创建APP及面板

接下来我们创建用户展示设备状态和控制设备用的APP面板。生活物联网平台为用户提供了一个“公版APP”容器，使得任何厂商的接入平台的设备都可以通过此APP为终端用户提供相应服务。在本例中，我们也将使用公版APP来完成测试设备的控制。点击“下一步：人机交互”按钮，进入APP配置界面。打开使用公版APP开关。

打开后，用户可以看到如下界面：

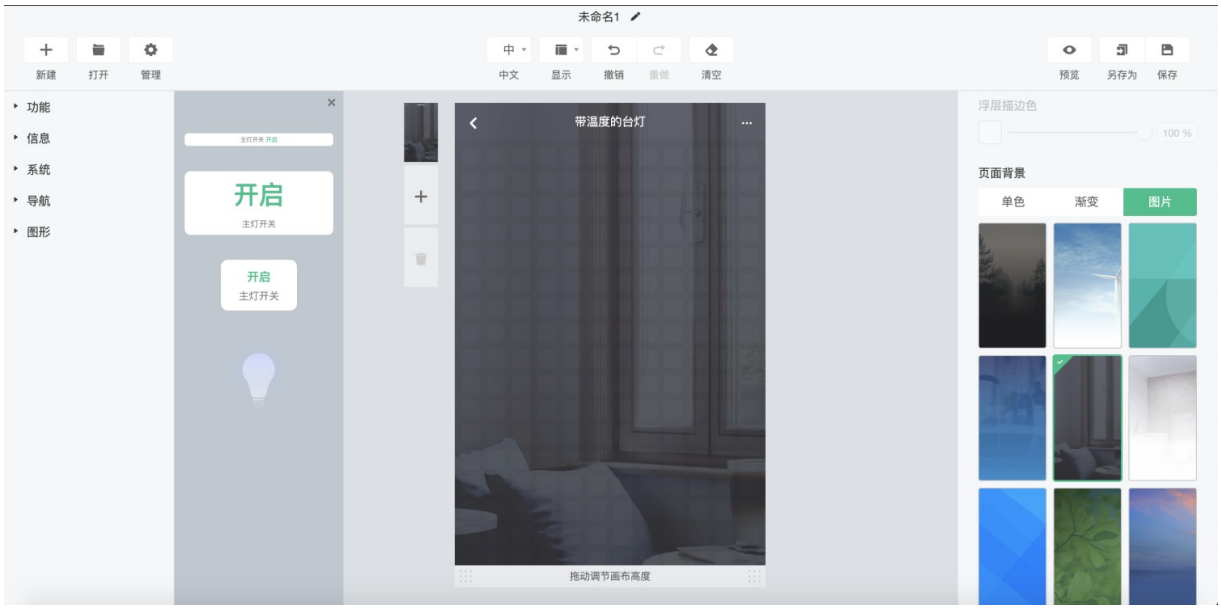
平台为用户提供了一些预置面板，点击“更换面板”可以进行选择



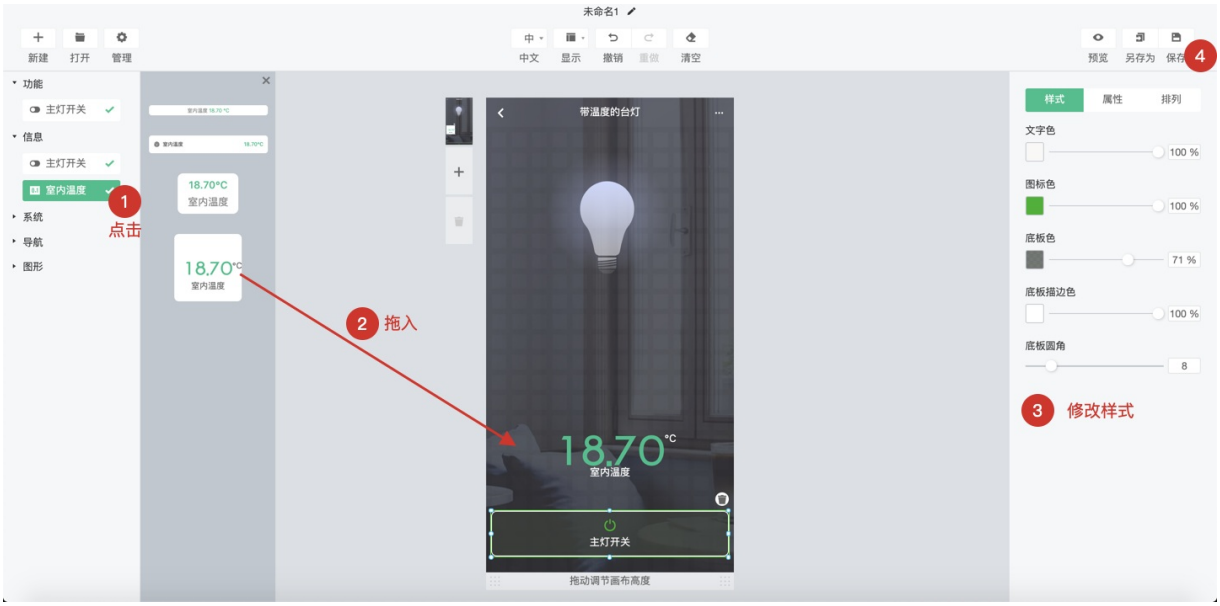
本例由于是一个复合设备，没有标准面板满足，所以我们选择第二页的自定义界面2.0进行开发

点击“配置界面2.0”按钮，可以直接进入配置。选择空白面板后，进入配置工作台

我们点击中间的空白面板，先给APP更换一下背景



然后在左侧的控件区将呈现和控制的控件加入到面板上。按步骤选择内容，拖入面板，修改样式，保存。



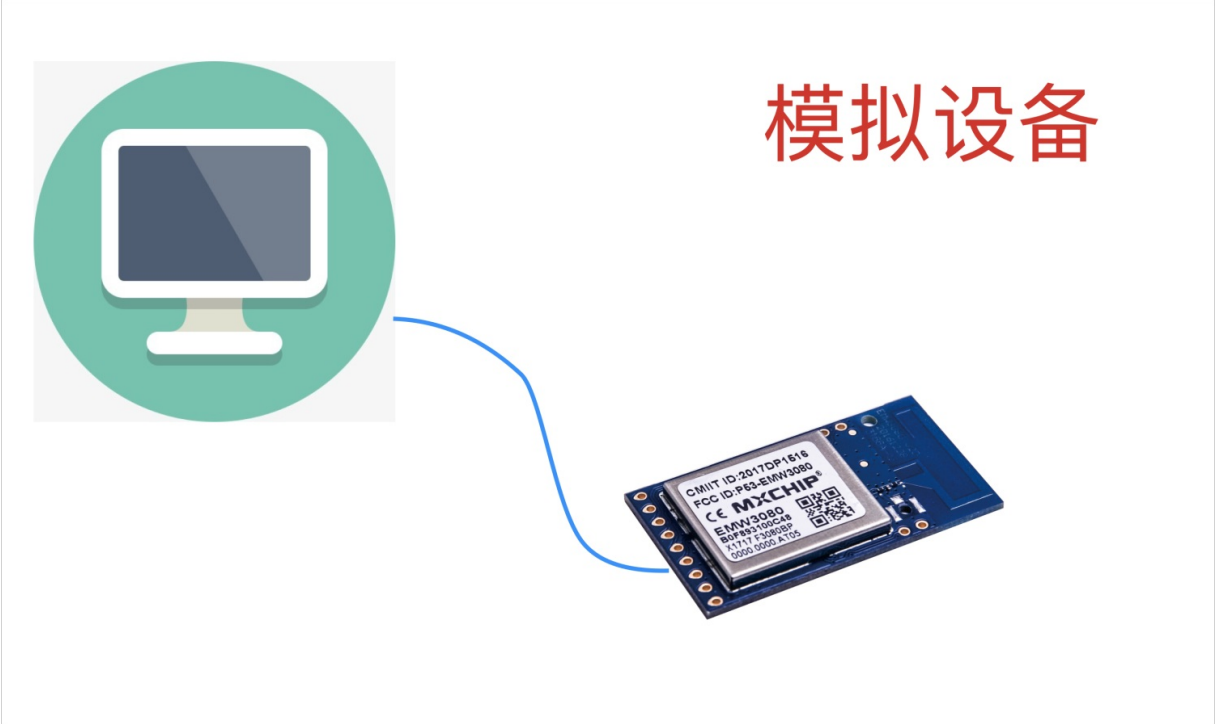
选择“保存”时，会询问是否发布到公版APP，选择确定：

到此，云端的工作告一段落，接下来我们要演示如何通过带有直连固件的模组，快速将设备接入刚刚配置好的智能设备项目。

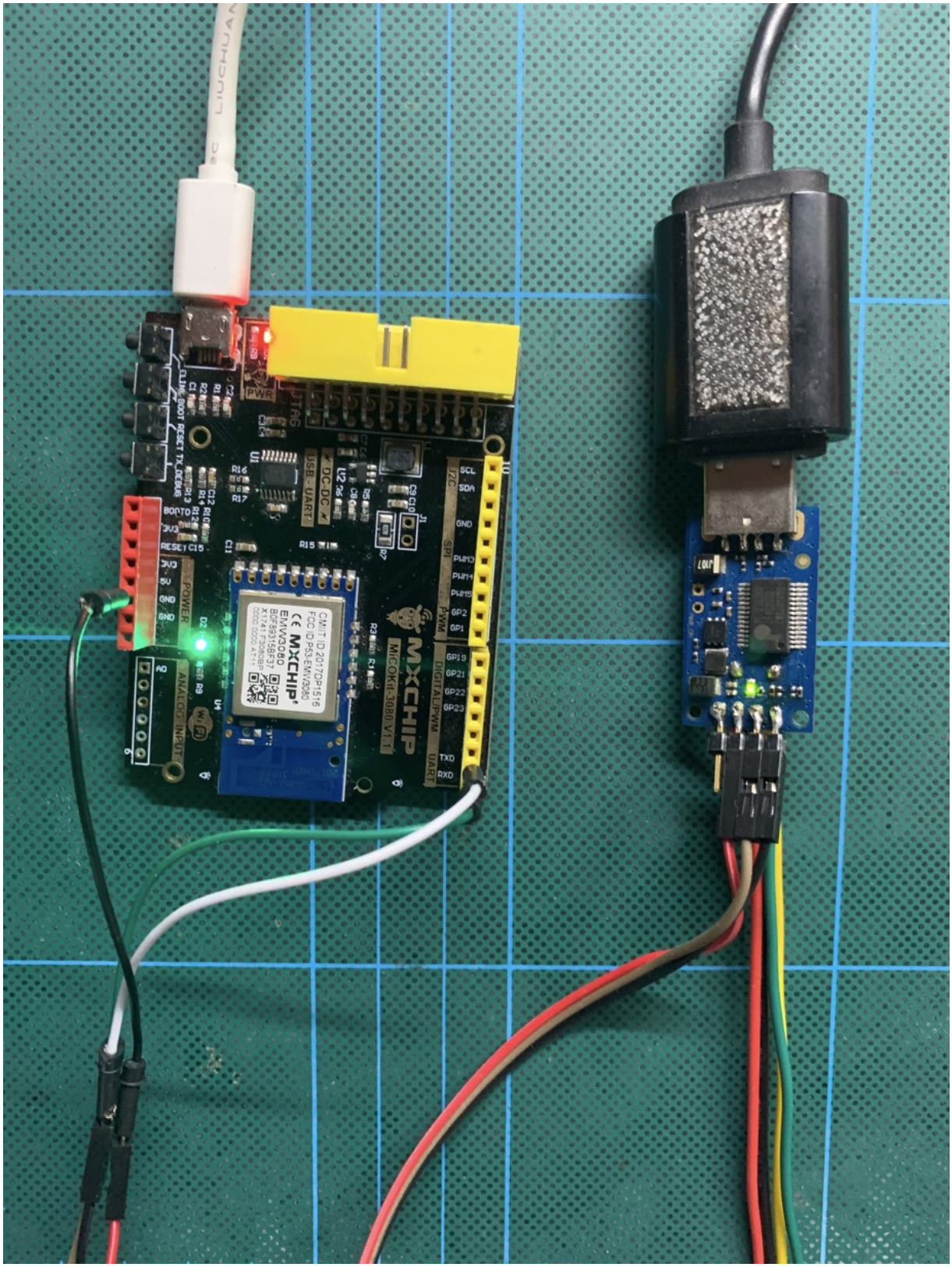
step2：设备端接入

将PC串口（用于模拟MCU）通过杜邦线与模组进行连接，确认连接可靠，并为模组可靠供电。

模拟连接图，白框中的PC+模组组成了待模拟的MCU+模组形式的智能设备。



模拟连接图



实际连接图

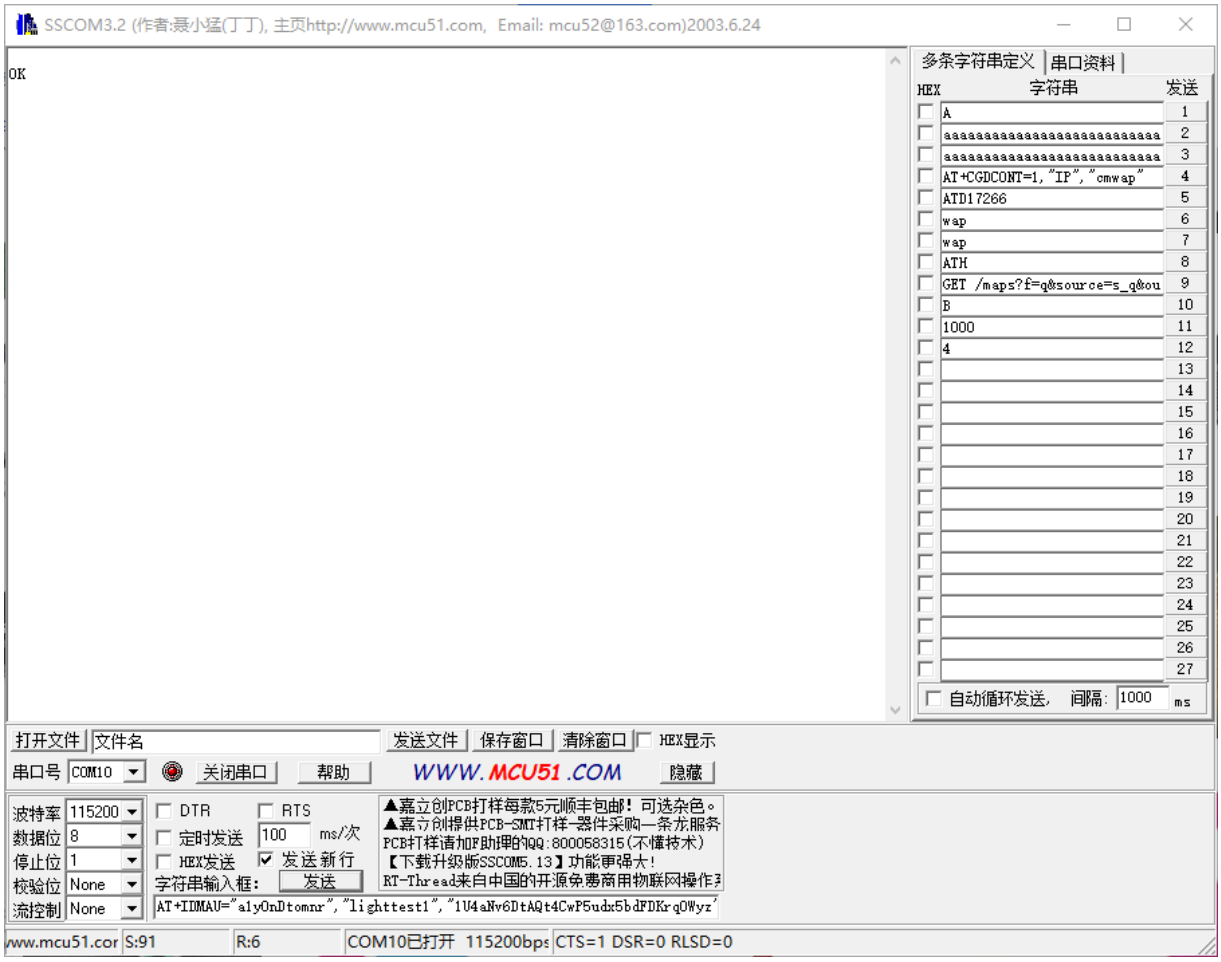
在PC端打开串口调试工具，用于模拟MCU向模组发送数据。

配置设备信息

通过 AT+IDMAU="PK","DN,"DS","PS"命令，可以将当前设备的相关激活信息传递给模组，模组拿到这些信息后，将会代表设备进行连云。我们打开测试设备详情页，获取这些设备信息。并组织出一个完整的AT命令：

```
AT+IDMAU="a1yOnDtomnr","lighttest1","1U4aNv6DtAQt4CwP5udx5bdFDKrqOWyz","QtKQAjyNmJzMHxdq"
```

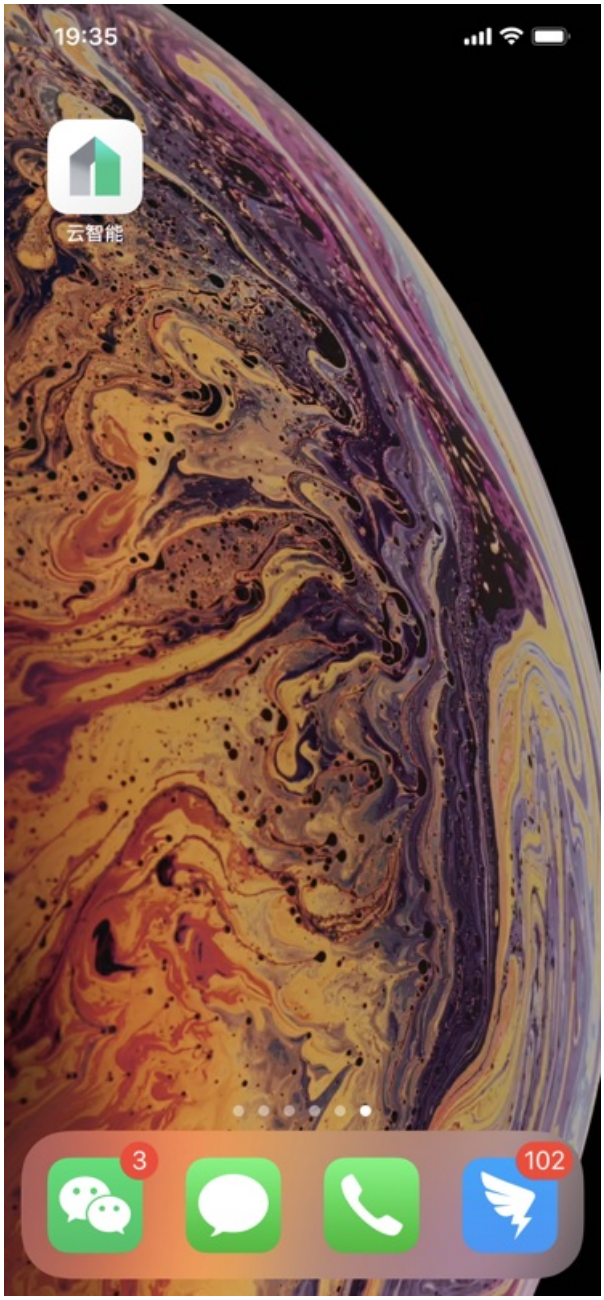
将此命令通过串口调试助手发送给模组，模组将返回“OK”。



自动配网

在配置好激活信息后，就可以用前面提到的公版APP为设备配网，并绑定设备到用户账户下。APP可通过扫描下面二维码安装，安装后需要注册用户。

安装APP用



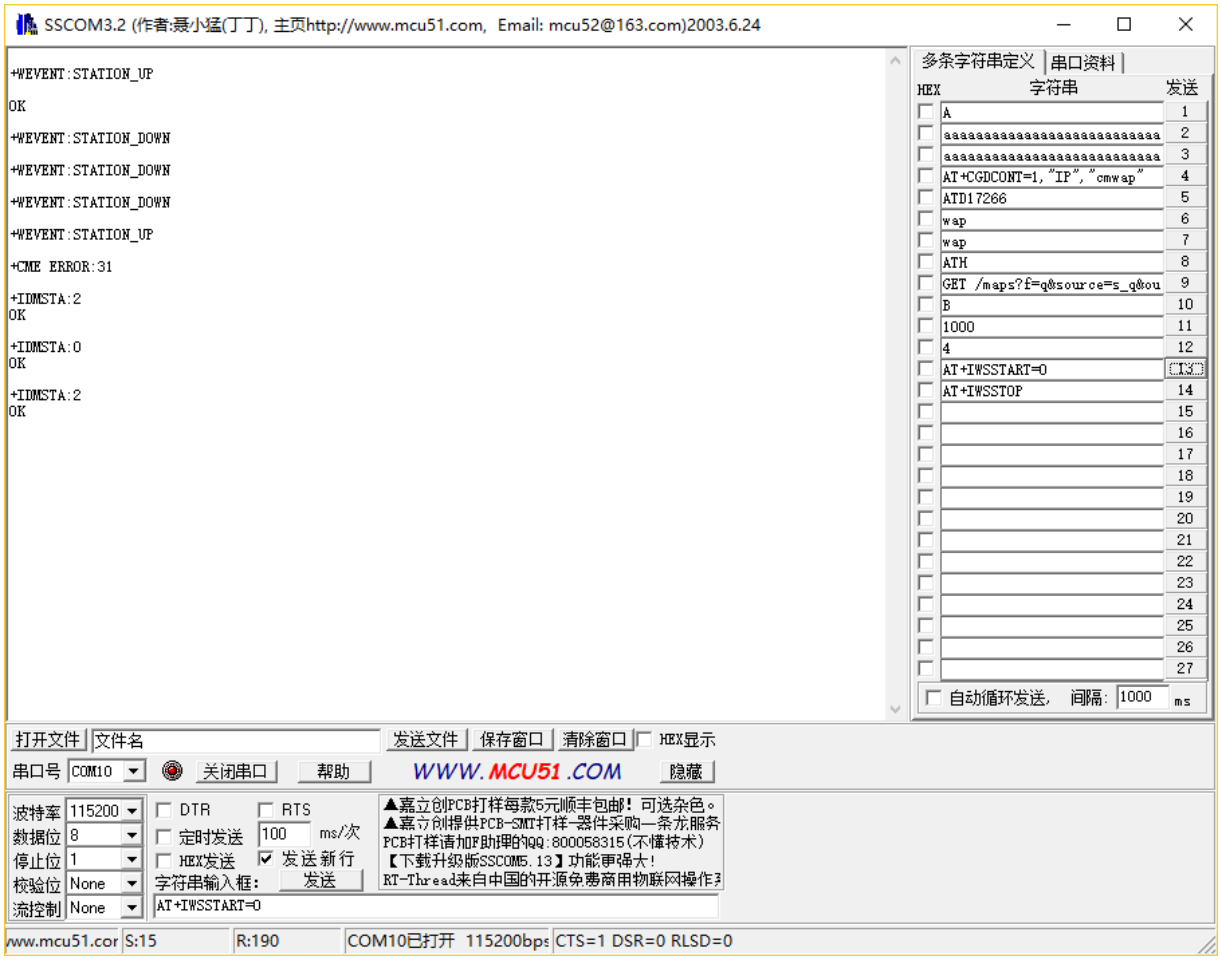
安装完成的公版APP

接下来我们需要准备用于配网的设备二维码，公版APP扫描该二维码以后，就会搜索当前网络下的待配网设备进行配网。配网二维码位于云端控制台开发工作台的“人机交互”页下

可以打开后保存至本地备用。接下来，我们通过AT指令，使模组进入配网模式。进入自动配网模式的AT指令如下

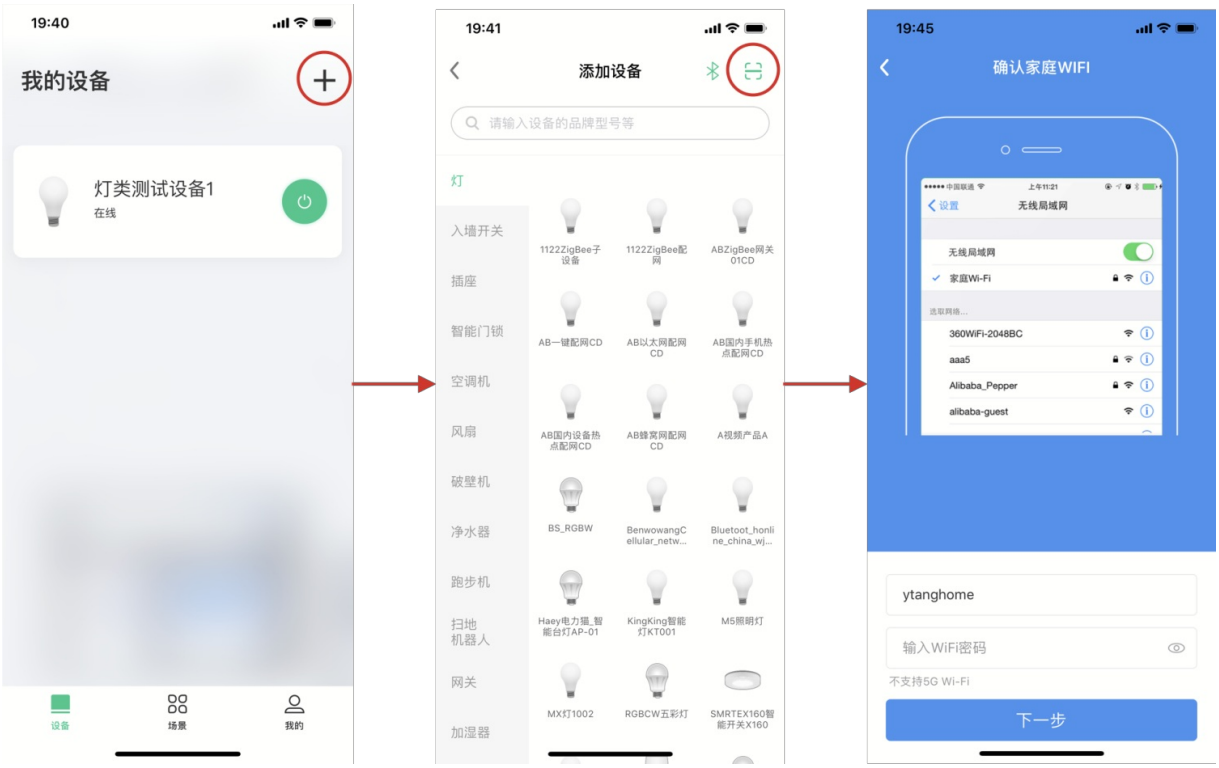
```
AT+IWSSTART=0
```

通过串口调试助手将命令发送给模组，模组将进入配网模式，配网指示灯开始闪烁。



同时，打开“云智能”APP，点击右上角的“+”号，开始添加设备。在添加设备页面中选择扫描二维码，对之前保存的

配网二维码进行扫描。接下来将待加入的wifi网络信息填写完整，“下一步”。注意：手机必须也要在此网络中。



接下来APP和模组之间会进行相关通信，自行完成配网过程。

很快就可以成功完成设备的添加，并进入到在step1中我们配置出的设备面板界面中


室内温度
主灯开关

此时，模组配网灯也停止闪烁。

模拟数据上报

接下来我们来模拟设备进行数据上报，改变APP上的状态。设备数据上报的AT指令如下

```
AT+IDMPP=0,"{\"LightSwitch\":1}"
```

其中第一个参数0表示是否分段，第二个参数则是我们需要上报的设备属性“标识符”以及其值。此处的属性标识符就是在step1中提到的需要读者注意的部分。端侧上传的标识符必须与云端定义的一致。在本例中，我们用到的两个标识符分别是

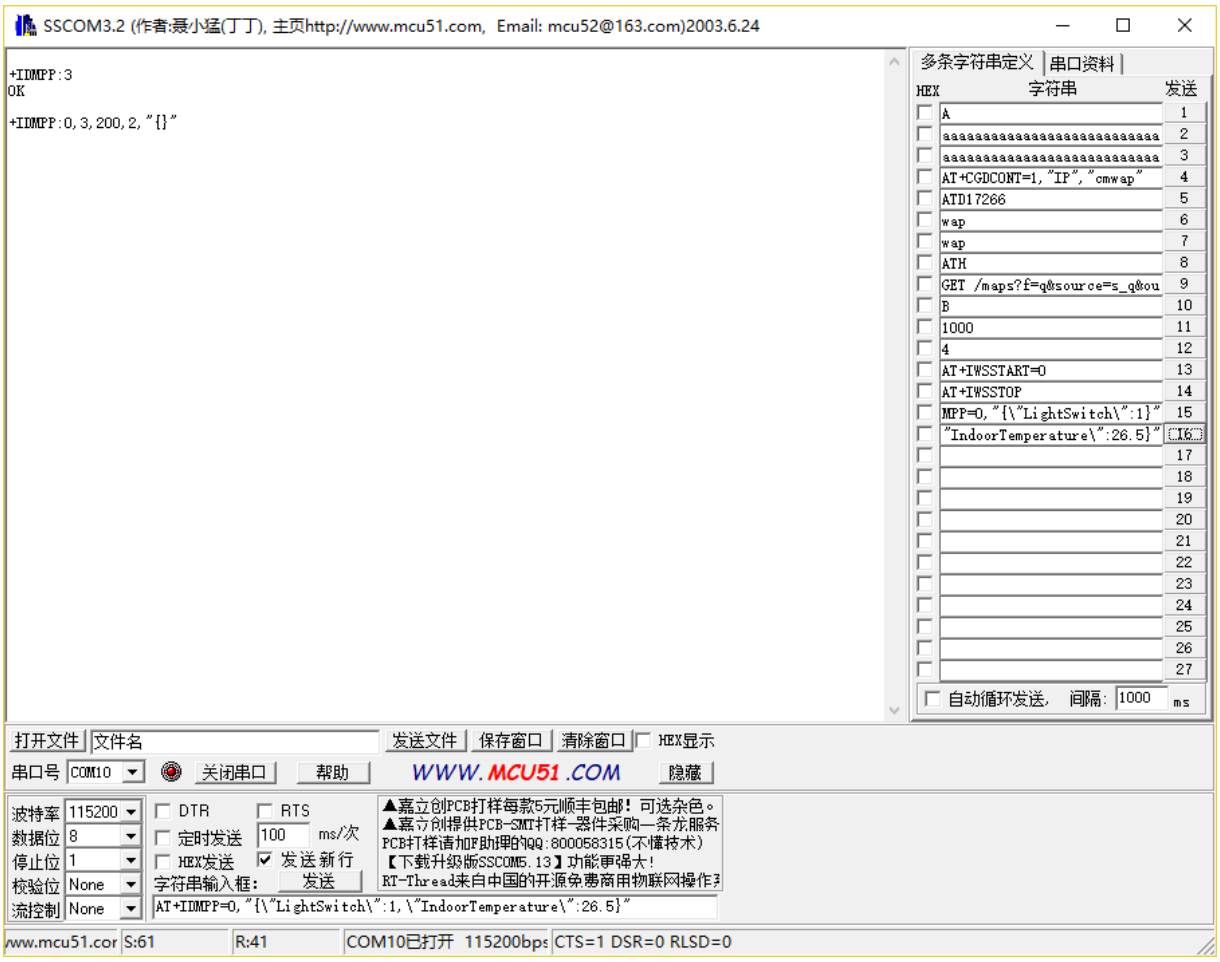
标识符名称	类型
LightSwitch	布尔型
IndoorTemperature	浮点型

注：

- 务必保证上传属性标识符与云端一致
- 务必保证上传属性类型与云端定义一致

设备属性采用json格式进行上报，可以单次报一个属性，也可以单次报多个属性。但是单个命令长度不得超过模组极限。且注意发送时引号等特殊字符的转义处理。下面以上报“主灯状态”为开，“室内温度”为26.5度为例，构建一个AT指令，并通过串口助手发送给模组。

```
AT+IDMPP=0,"{\"LightSwitch\":1,\"IndoorTemperature\":26.5}"
```

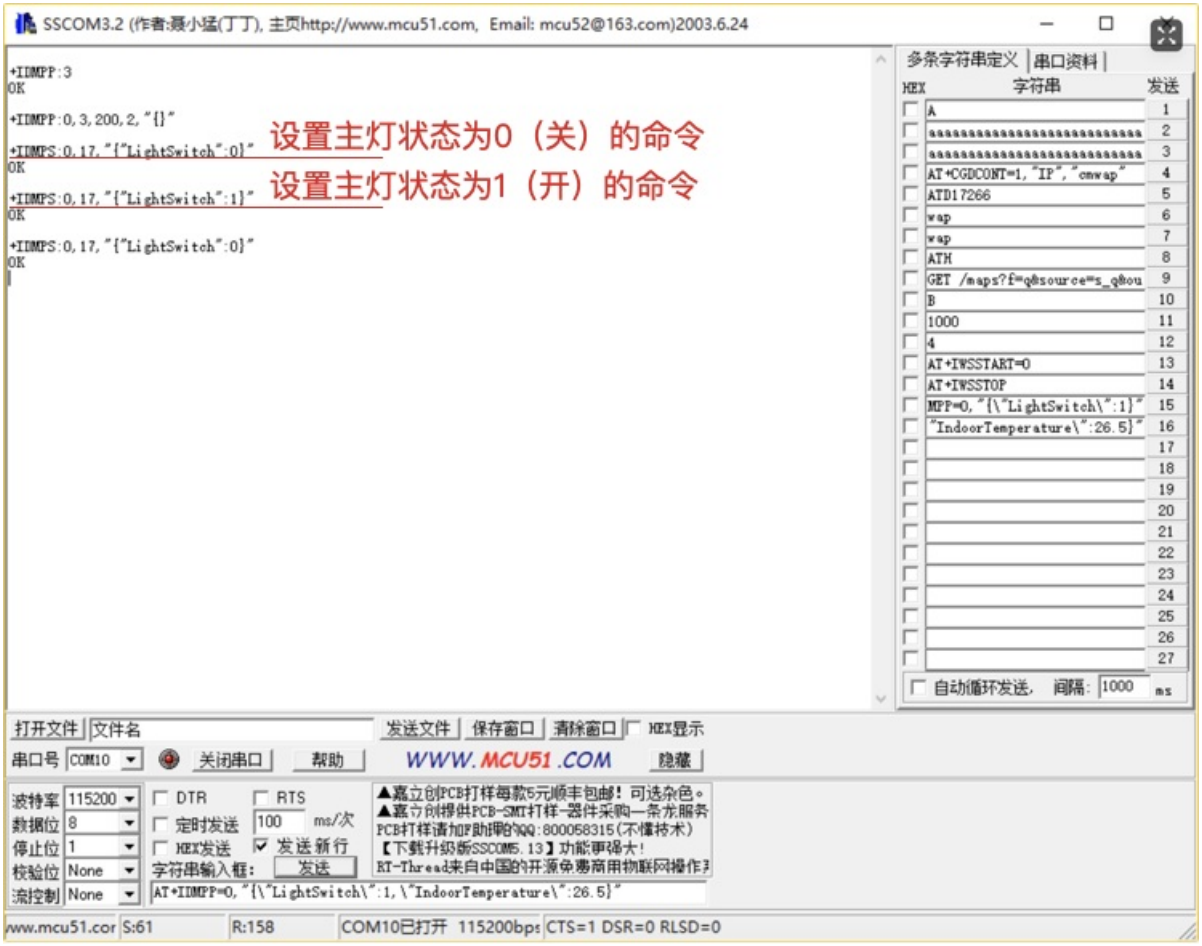


模组收到相关指令后，会帮助用户完成Alink协议的打包，并上报至云端。同时，将云端的返回结果通过串口返回给MCU侧（PC串口调试助手）。确认数据上报成功后，我们可以查看APP侧面板变化。

我们可以看见，面板中灯变为打开状态，同事室内温度也更新为了26.50度，与刚才上报的数据一致。

APP反控设备

接下来，我们将尝试通过APP上的按钮对设备进行控制。我们点击APP界面上的“主灯开关”按钮。使设备进入关闭状态，此时APP上的灯泡图标也回相应变暗。同时，我们观察与模组相连的PC端串口调试助手。在串口调试助手的接收框中，展现出了来自云端的控制命令如下图：



在实际的设备收到改命令后，可以通过收到的命令的相关值对设备进行状态更改。实现云端对设备的反向控制操作。

总结

在以上的演示中，我们在设备侧通过简单的4个AT命令完成了设备与生活物联网平台的对接，实现了设备的自动配网，数据的上报以及平台对设备控制指令的下发，完成了一个完整的智能家居设备的开发过程。在设备侧大大简化了用户对接阿里云IoT平台工作量。