

飞燕模组认证应用使用说明

使用步骤

- 第一步：在飞燕平台上的操作
- 第二步：集成模组认证应用，并替换四元组信息
- 第三步：设备上电连云

补充说明：

- 带有AliOS的版本支持的串口命令
- 设备本地OTA测试
- 功能说明：
- 功能配置：

使用步骤

第一步：在飞燕平台上的操作

1. 需要在飞燕平台创建两个产品，一个是一机一密激活方式同时支持Alink的JSON协议的产品，另外一个支持一型一密使用二进制协议的产品，注意创建产品的配置，选择透传/自定义就是二进制方式：

节点类型

* 节点类型

☒ 设备

☐ 网关

* 是否接入网关

☐ 是

☒ 否

连网与数据

* 连网方式

WiFi

▼

* 数据格式

ICA 标准数据格式 (Alink JSON)

^

☒ ICA 标准数据格式 (Alink JSON)

透传/自定义

二进制方式

更多信息

▼

完成

取消

另外一机一密和一型一密的配置在项目管理页面

项目设置

产品总数
18

项目激活码
20

已激活设备
2

当前在线设备
0

量产管理

产品管理

已在阿里云物联网平台创建过产品？[关联产品](#) [创建新产品](#)

App 管理
平台提供公版 App，同时支持自有品牌 App 控制产品

进入量产管理页面如下，可以切换烧录方式：

量产管理

量产概述
设备激活码
量产记录

项目激活码总数
20

剩余可用激活码
10

已量产激活码
10

使用情况

产品名称	Product Key	通讯方式	状态	已量产(个)	烧录方式	操作
		WiFi	开发中	0	一机一密	批量投产

2. 上面两个产品对应的模组认证应用定义宏：

CT_PRODUCT_DYNAMIC_REGISTER_AND_USE_RAWDATA 在ct_main.h文件中，不定义是第一种产品，定义就是第二种产品，针对一型一密请注意，需要先在飞燕控制台导入DeviceName生成四元组信息，可以批量导入或者手动添加，参考文档：

[https://help.aliyun.com/document_detail/127667.html?](https://help.aliyun.com/document_detail/127667.html?spm=5176.11065259.1996646101.searchclickresult.36806559ybEqdl)

<spm=5176.11065259.1996646101.searchclickresult.36806559ybEqdl>

批量导入，如下图进入量产管理页面找到对应的产品，选择批量投产然后执行批量导入操作

所用激活码类型
设备激活码

激活码规格
标准
日均消息量小于3000条

烧录方式
一机一密(推荐) 一型一密
同一批设备可以烧录相同的ProductKey和ProductSecret，但需要预先批量上传Device Name（如MAC地址、SN、IMEI等）

激活码生成方式
批量上传
单个文件不超过2M，一次最多包含10,000条记录，[下载 csv模板](#)
上传文件

一机一密	批量投产
一机一密	批量投产
一机一密	批量投产
一机一密/一型一密	批量投产
一机一密	批量投产
一机一密	批量投产
一机一密	批量投产
一机一密	批量投产
一机一密	批量投产
一机一密	批量投产
一机一密	批量投产

手动添加调试设备：



3. 第一种产品需要导入模组认证的物模型，参考docs目录下面的ct_model.json文件，请根据你的产品编辑这个文件第四行替换里面的productKey的值，然后在创建产品之后的功能定义界面选择导入物模型，导入该文件即可；



4. 第二种产品需要使用协议转换脚本，参考tools文件夹里面的文件：Alink协议与二进制转换脚本.txt 在创建完成该产品之后的设备管理页面有个脚本导入入口，参考下图



[重新选择](#) [采购](#)

类型：模组 已认证

品牌：iComm


认证类型：阿里云IoT技术认证

[查看详情](#)


通讯类型：Wi-Fi

型号：M169


检测项：常温射频（传导），Wi-Fi兼容，空口...



认证模组
[SDK](#)



嵌入式开发
[设备端开发指南](#)



数据解析
需在云端转为ICA标准数据格式 [编辑脚本](#)

点击编辑脚本，将脚本内容复制进去，然后参考下面内容，复制内容0x010000006400之后点击运行，运行成功之后点击提交就可以了，设备端代码在ct_ut.c里面有函数user_post_raw_data上报开关属性数据。

```
//      return [].slice.call(uint8Array);
78  }
79  function buffer_float32(value) {
80    var uint8Array = new Uint8Array(4);
81    var dv = new DataView(uint8Array.buffer, 0);
82    dv.setFloat32(0, value);
83    return [].slice.call(uint8Array);
84  }
```

模拟输入 输入模拟数据，点击执行，查看解析结果

模拟类型：设备上报数据

i 1

0x010000006400

保存草稿

运行

提交

取消

这里的协议说明：

0x010000006400 0x01是表示设备上报的数据，后面四个字节是id信息，最后一个字节表示灯的开关状态，00表示关闭，01表示开灯。

例如：传入参数 ->

0x010000006400

输出结果 ->

`{"method":"thing.event.property.post","id":"100","params":{"LightSwitch":0},"version":"1.0"}`

另外设备下发数据

传入参数 ->

```
{"method":"thing.service.property.set","id":"100","version":"1.0","params":{"LightSwitch":0}}
```

输出结果 ->

0x0200000006400

0x02是表示云端下发给设备的数据，后面四个字节是id信息，最后一个字节表示灯的开关状态，00表示关闭，01表示开灯。

设备端接收RawData的处理函数是ct_mian.c里面的

```
1 static int user_down_raw_data_arrived_event_handler(const int devid,
  const unsigned char *payload,
2                                     const int payload
   _len)
```

第二步：集成模组认证应用，并替换四元组信息

1. 模组认证的应用有两个版本，一个是AliOS版本，一个是飞燕C-SDK的版本，两个版本的差异是AliOS版本多一个ct_cmds.c文件，可以支持写命令行的操作，参考应用的src文件夹，里面有fy_aos_sdk和fy_c_sdk两个文件夹。
- 如果是飞燕AOS的SDK版本，请将certification文件夹拷贝到example目录或者 **Living_SDK/example文件夹（大于等于1.3.0的版本）** 下面；
- 注意在ct_main.h文件里面有几个宏定义如下，根据你使用的SDK版本进行定义：

```
1 //if your fy sdk is V1.0.0 or V 1.1.0
2 //define CT_FY_SDK_VERSION_1_0_OR_1_1
3
4 //if your fy sdk is V1.3.0 or V 1.4.0
5 #define CT_FY_SDK_VERSION_1_3_OR_1_4
6
7 //if your fy sdk is V1.5.0
8 //define CT_FY_SDK_VERSION_1_5
```

- 如果是飞燕C-SDK的版本，请合并fy_c_sdk文件夹里面的内容，注意合并下examples/iot.mk文件里面的certification相关修改；
- 如果需要在C-SDK版本设备本地OTA功能，需要合并下面几个文件的修改，增加两个ota相关的函数：dm_ota_download和iotx_ota_download

```
1 ./src/services/linkkit
2 ./src/services/linkkit/dm
```

```

3 ./src/services/linkkit/dm/dm_ota.c
4 ./src/services/linkkit/dm/dm_ota.h
5 ./src/services/ota
6 ./src/services/ota/iotx_ota.c
7 ./src/services/ota/iotx_ota.h

```

2. 替换四元组有三种方法，参考代码ct_entry.c里面的函数load_ct_meta_info

- 第一种方法是从kv里面读取
- 第二种方法是如果KV没有就会使用ct_ut.h里面定义的四元组
- 第三种方法就是厂家自己的适配函数，适配下面几个函数即可：

在类似这样的c文件里面src/ref-impl/hal/os/ubuntu/HAL_OS_linux.c

```

1 HAL_SetProductKey();
2 HAL_SetProductSecret();
3 HAL_SetDeviceName();
4 HAL_SetDeviceSecret();
5 HAL_GetProductKey();
6 HAL_GetProductSecret();
7 HAL_GetDeviceName();
8 HAL_GetDeviceSecret();

```

第三步：设备上电连云

补充说明：

带有AliOS的版本支持的串口命令

命令名称	用法	备注
reset	串口执行reset	产品恢复出厂设置，需要删掉所有用户数据，例如ssid等信息
netmgr	netmgr connect ssid pw	串口命令给设备配网，ssid和pw是连接路由器的AP名称和密码（ 需要先执行reset命令 ）
active_awss	串口执行active_awss	启动一键配网功能（ 需要先执行reset命令 ）
dev_ap	串口执行dev_ap	启动设备热点配网（ 需要先执行reset命令 ）
kv list	串口执行kv list	查看所有用户保存的信息（ 必须编译Debug版本 ）

备注：不带AliOS版本需要支持所有五个命令的功能，实现方法可以不一样

设备本地OTA测试

功能说明：

这个功能可以在测试OTA功能的时候使用，开发者不需要反复去操作升级推送，可以配置一个版本号大于实际固件版本号就可以反复升级测试

功能配置：

本功能可以参考ct_ota.h里面的宏定义：CT_DOWNLOAD_OTA_WHEN_CONNECT_CLOUD，可以通过KV配置升级信息到flash中，信息如下：

```
1 #define OTA_DESC_KV_KEY_SIZE "ct_ota_size"
2 #define OTA_DESC_KV_KEY_VERSION "ct_ota_ver"
3 #define OTA_DESC_KV_KEY_MD5 "ct_ota_md5"
4 #define OTA_DESC_KV_KEY_URL "ct_ota_url"
5
6 //例如串口执行kv命令设置参数
7 kv set ct_ota_size 609758
8 kv set ct_ota_ver ct-1.5.0-20200319.180352
9 kv set ct_ota_md5 137367c97efaf3635d8754a4a67bc8ef
10 kv set ct_ota_url https://iotx-ota.oss-cn-shanghai.aliyuncs.com/ota/
```

ct_ota_size：OTA固件包的大小，单位为字节

ct_ota_ver：固件版本号

ct_ota_md5：固件的MD5值

ct_ota_url：固件下载的URL地址（可以从飞燕平台固件升级中创建升级任务之后获取，也可以是你自己的下载地址），这个地址太长的话可能不能使用kv set命令设置

参考下图，签名算法必须是Md5，固件签名就是MD5，右键点击下载复制链接地址就是固件下载的URL地址

运营中心

概览

设备运维

设备列表

固件升级

告警中心

激活数据

活跃数据

用户运营

数据大屏

固件列表 > 固件详情

XJ_OTA_V132 已验证

固件类型: 整包

固件签名: 552bb8701c98736bfe36b8c091d7d950

下载

签名算法: Md5

目标设备总数
1

目标成功数
0

目标失败数
1

固件信息 批次管理

固件基本信息

固件名称	XJ_OTA_V132	所属产品	模组认证测试一	固件签名	552bb8701c98736bfe3...
固件版本号	V1.3.2	添加时间	2020/03/09 12:09:02	签名算法	Md5
固件状态	已验证	验证进度	100%		
固件描述	-				