

涂鸦 MCU_SDK 使用指南

SMART LIFE

SMART LINK

1. 下载MCU_SDK

The screenshot shows the Tuya Developer Portal interface. At the top, there is a navigation bar with the Tuya logo and links for '产品列表' (Product List) and '开发资源' (Development Resources). The main content area is titled '三路触摸开关' (Three-Touch Switch). On the left, there is a sidebar menu with options: '产品信息' (Product Information), '数据点' (Data Points), '调试流程' (Debugging Process), and '申请发布' (Apply for Release). The '调试流程' (Debugging Process) option is selected. The main content area displays the '开发流程' (Development Process) section, which includes a red arrow pointing to the '开发流程 (点击下载接入手册)' (Development Process (Click to Download Integration Manual)) link. Below this, there are three steps: 1. Download MCU SDK for fast MCU program development; 2. Download feature point debug files and import the Tuya serial port debug assistant for fast program verification; 3. Connect the Tuya module to the control board, download the Tuya smart app, and add the device for demonstration. Below the steps, there is a section for '开发文件' (Development Files), which states that files are automatically generated based on the product feature points and the Tuya serial port communication protocol. Two cards are shown: 'MCU SDK' and '功能点调试文件' (Feature Point Debug Files). Each card has a '使用说明' (Usage Instructions) button and a download icon.

打开开发者后台网址

<http://developer.cn.tuya.com/>

如已建立完产品，在“调试流程”处 点击下载MCU SDK

如果您没有贵公司的账号，请与贵司相关领导联系

2.将得到的MCU_SDK文件导入单片机工程

文件包括：

1. mcu_api.c mcu_api.h ←客户需要调用的函数
2. protocol.c protocol.h ←客户根据项目需求，需修改这个2个文件
3. system.c system.h ←串口通讯协议的具体实现,可以不做了解
4. wifi.h

3.修改配置（必改）



打开protocol.h

在开发者后台找到“产品Key”，请确保两者一致

```
#define PRODUCT_KEY "tCPzPwLYydfakDIO"
```

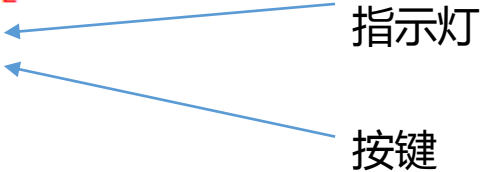
4.修改配置（必改）

1.如果wifi指示灯和按键是**接在wifi模块**上的，那么请开启

```
#define    WIFI_CONTROL_SELF_MODE
```

并且根据实际的硬件连接，将指示灯和按键所连接的GPIO脚位填入下面两行

```
#ifdef    WIFI_CONTROL_SELF_MODE
#define    WF_STATE_KEY    14
#define    WF_RESERT_KEY    0
#endif
```



指示灯

按键

2.如果灯和按键**接在MCU上**，请保持 define 被注释状态

```
// #define    WIFI_CONTROL_SELF_MODE
```

5.修改配置（可选）

1.修改缓冲区大小:仅当传输数据长度特别长时需要修改，一般情况使用默认值即可

```
#ifndef SUPPORT_MCU_FIRM_UPDATE
#define WIFI_UART_RECV_BUF_LMT    64      //根据用户 DP数据大小量定,必须大于 40
#else
#define WIFI_UART_RECV_BUF_LMT    300    //固件升级缓冲区,需大缓存,必须大于 260
#endif

#define WIFIR_UART_SEND_BUF_LMT   64      //根据用户 DP数据大小量定,必须大于 40
```

6.启动wifi服务

- 1:在需要使用到wifi相关文件的文件中include "wifi.h"文件
- 2:在MCU初始化调用mcu_api.c文件中的wifi_protocol_init()函数
- 3:将串口单字节发送函数填入protocol.c文件中uart_transmit_output函数内,并删除#error
- 4:在串口接收函数里面调用mcu_api.c文件内的uart_receive_input函数,并将接收到的字符作为参数传入
- 5:单片机进入while循环后调用mcu_api.c文件内的wifi_uart_service()函数

7.配置验证

端口

串口号: COM5 打开串口 发送

Schema 路径

浏览

开启心跳检测 查询产品信息

查询MCU设定的模块工作模式 状态查询 进入产测模式

1. 连接单片机的串口和电脑的串口
2. 开发者后台下载涂鸦串口调试助手.exe 和功能点调试文件(config.json)
3. 打开涂鸦串口调试助手.exe
4. 设定串口号后点击“打开串口”
5. 点击浏览，打开刚下载的config.json
6. 点击“开启心跳检测”

8.配置成功

```
心跳检测-发送数据：
```

```
55 aa 00 00 00 00 ff
```

```
接收到数据：
```

```
55 aa 00 00 00 01 01 01
```

```
后续心跳返回！
```

串口配置正常的返回数据

恭喜！

MCU SDK移植成功

9.配置异常

心跳检测-发送数据:

```
55 aa 00 00 00 00 ff
```

接收数据超时，设备处于离线状态

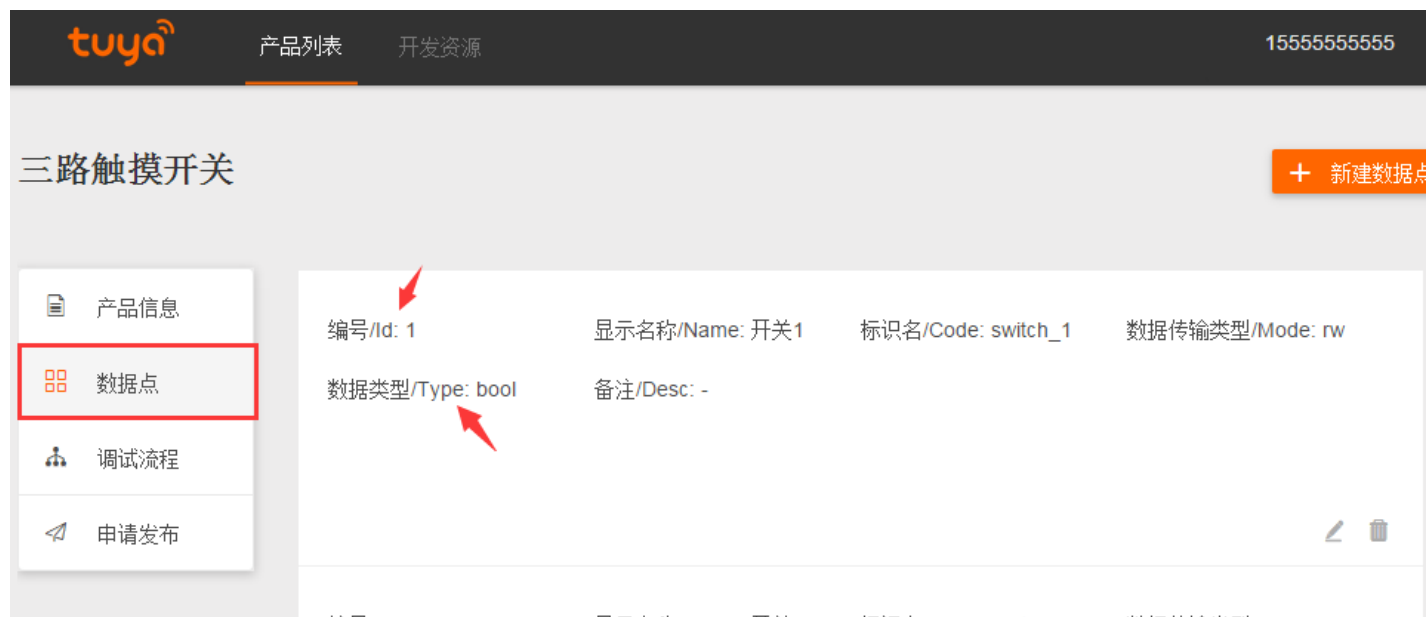
串口配置异常的返回数据

原因可能会有如下几种情况：

1. 电路板未上电
2. 串口号选错
3. 串口tx，rx 接反
4. 代码配置错误，请检查代码
5. 如上述几项均无法解决问题，可与我司联系

10.功能数据处理

1:在protocol.c文件中,确认下发结构体download_cmd内dpID及dp类型



```
#define DPID_SWITCH 1  
#define DPID_TEMPER_SET 2
```

```
const DOWNLOAD_CMD_S download_cmd[] =  
{  
    {DPID_SWITCH, DP_TYPE_BOOL},  
    {DPID_TEMPER_SET, DP_TYPE_VALUE},  
    {DPID_GEAR, DP_TYPE_ENUM},  
    {DPID_DISPLAY, DP_TYPE_STRING},  
    {DPID_RGB, DP_TYPE_RAW},  
};
```

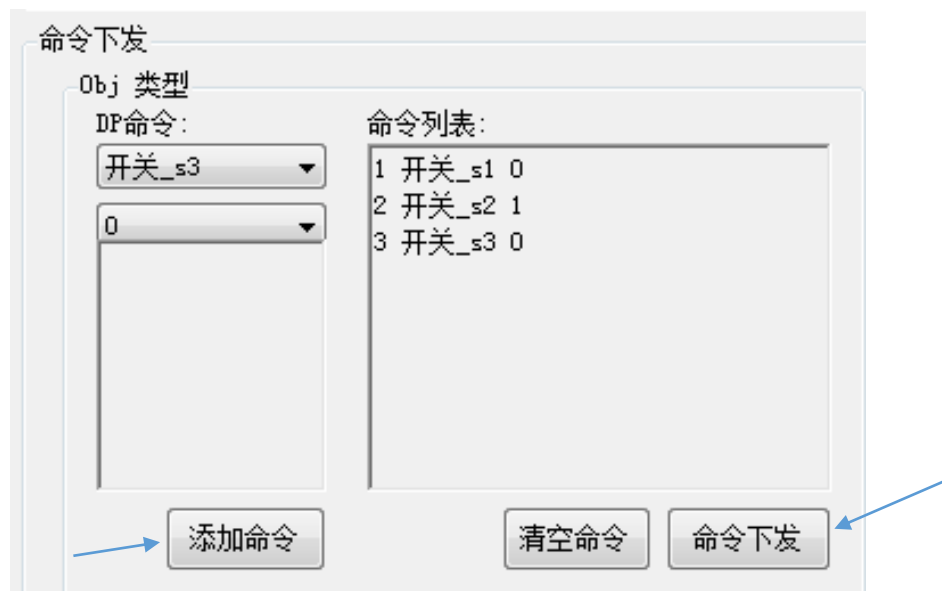
11.功能数据处理-接收

2:在protocol.c文件dp_download_handle函数内填入dp对应处理函数

```
unsigned char dp_download_handle(unsigned char dpid,  
{  
    /*****  
    当前函数处理可下发/可上报数据调用  
    具体函数内需要实现下发数据处理  
    完成用需要将处理结果反馈至APP端,否  
    *****/  
    unsigned char ret;  
    switch(dpid)  
    {  
    case DPID_SWITCH:  
        //童锁处理函数  
        ret = dp_download_look_handle(value,length);  
        break;
```

```
static unsigned char dp_download_look_handle(const un:  
{  
    unsigned char ret;  
    //示例:当前DP类型为BOOL  
    unsigned char look;  
    look = mcu_get_dp_download_bool(value,length);  
    if(look == 0)  
    {  
        //开关关  
    }  
    else  
    {  
        //开关开  
    }  
  
    //处理完DP数据后应有反馈  
    ret = mcu_dp_bool_update(DPID_LOOK,look);  
    if(ret == SUCCESS)  
        return SUCCESS;  
    else  
        return ERROR;
```

12.功能数据处理-接收调试



1.使用涂鸦串口调试助手，完成“7.配置验证”后，点击“添加命令”，可实现同时发送多条命令

2.命令下发

3. 上一页添加的函数将会收到刚才下发的数据

13.数据上报

```
void all_data_update(void)
{
/*****
当前函数处理全部数据上报(包括可下发/可上报和只上报)
需要用用户按照实际情况实现
1:需要实现可下发/可上报数据点上报
2:需要实现只上报数据点上报
此函数为MCU内部必须调用
用户也可调用此函数实现全部数据上报
*****/
//本示例数据为模拟数据,请按照实际数据填入
//可下发可上报数据

mcu_dp_bool_update(DPID_LOOK,1); //BOOL型数据上报
mcu_dp_value_update(DPID_TEMPER_SET,25); //VALUE型数据上报
// mcu_dp_enum_update(DPID_GEAR,4); //枚举型数据上报
mcu_dp_string_update(DPID_DISPLAY,"1234",4); //STRING型数据上报
mcu_dp_raw_update(DPID_RGB,"123",3); //RAW型数据上报
//只上报数据示例
mcu_dp_fault_update(DPID_ERROR,25); //故障型数据上报
mcu_dp_value_update(DPID_TEMPER_CURRENT,25); //VALUE型数据上报
} ? end all_data_update ?
```

1.打开protocol.c 找到函数all_data_update(void)

2.把所有需要上报的数据点在这个函数中罗列出 (wifi_sdk会在特定时刻 (如开机) 调用这个函数 , 以获取各个功能当前状态)

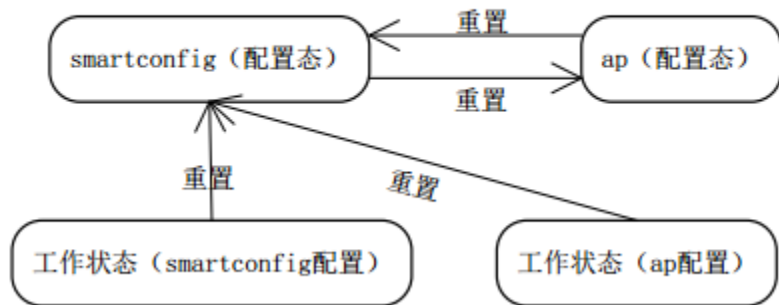
4.在需要上报处 (如按钮按下) 根据dp类型调用上图中所罗列的 : mcu_dp_xxxx_updata(DPID_X,n)

14.重置WIFI（重置按钮在mcu上）

当wifi模块需要连接新的路由器，需要重置WIFI,有如下两个函数可供调用：

```
1. void mcu_reset_wifi(void)
{
    reset_wifi_handle();
}
```

1) 重置 WIFI 状态转化如下图所示：



2.直接进入指定的配置态

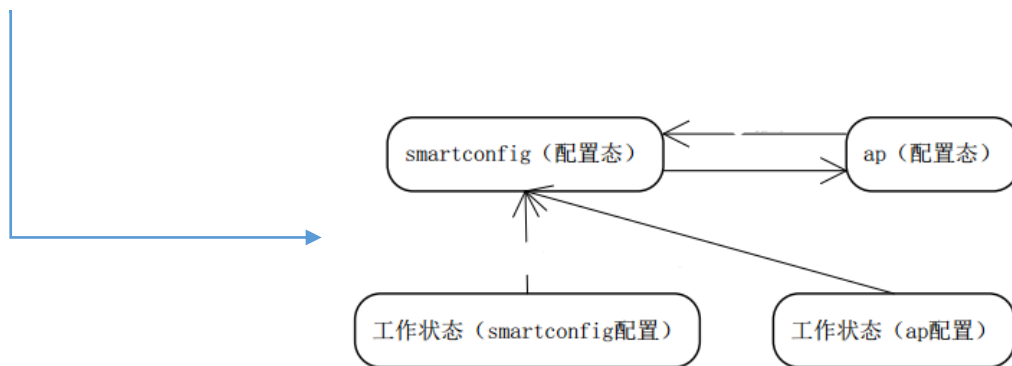
```
mcu_set_wifi_mode(AP_CONFIG);
mcu_set_wifi_mode(SMART_CONFIG);
```

15.重置WIFI-获取状态

1.查询重置结果 `unsigned char mcu_get_reset_wifi_flag(void)`

2.查询配置结果 `unsigned char mcu_get_wificonfig_state(void)`

3.获取wifi连接状态 `unsigned char mcu_get_wifi_work_state(void)`



16.MCU在线升级

1.开启宏 `64: #define SUPPORT_MCU_FIRM_UPDATE`
`#define MCU_VER "1.0.0"`

```
/*-----*/  
函数名称 : mcu_firm_update_handle  
功能描述 : MCU进入固件升级模式  
输入参数 : value:固件缓冲区  
           position:当前数据包在于固件位置  
           length:当前固件包长度(固件包长度为0时,表示固件包发送完成)  
返回参数 : 无  
使用说明 : MCU需要自行实现该功能  
/*-----*/  
unsigned char mcu_firm_update_handle(const unsigned char value[], unsigned short position, unsigned short length)  
{  
// #error "请自行完成MCU固件升级代码,完成后请删除该行"  
if(length == 0)  
{  
//固件数据发送完成  
}  
else  
{  
//固件数据处理  
}  
  
return SUCCESS;  
}
```

注：升级过程由手机发起，调试时可使用涂鸦串口调试助手点击“升级启动”

17.获取在线时间

1.MCU需要获取网络时间，开启下面的define

```
#define SUPPORT_MCU_RTC_CHECK
```

2.调用mcu_get_system_time(void)启动获取流程

```
void mcu_get_system_time(void)
```

3.获取到的时间会被送到函数 mcu_write_rtctime (protocol.c) ，请自行完成获取时间后存入MCU的操作

```
void mcu_write_rtctime(unsigned char time[])
```

```
{  
    /*  
    time[0]为 是否获取时间成功标志，为 0表示失败  
    time[1]为 年份，0x00表示 2000年  
    time[2]为 月份，从 1开始到 12结束  
    time[3]为 日期，从 1开始到 31结束  
    time[4]为 时钟，从 0开始到 23结束  
    time[5]为 分钟，从 0开始到 59结束  
    time[6]为 秒钟，从 0开始到 59结束  
    */  
    .....
```

18.产测模式

功能完善中...

技术支持

support@tuya.com