



MCU SDK 移植指南

设备接入 > MCU 开发接入 > BLE 单点通用方案 > 软件开发

文档版本: 20200715

[查看在线版本](#)

目录

1 概述	1
1.1 开发步骤:	1
1.2 文件概览:	1
2 第一步：初始化	2
3 第二步：实现具体用户函数	3
3.1 APP 下发数据处理	3
3.2 数据上报处理	3
4 第三步：完成数据帧的选择	6
5 第四步：配置固件版本信息	8
6 第五步：完善接口	9
7 第六步：完善 MCU OTA 功能	10

1 概述

此 MCU_SDK 是根据涂鸦开发平台上定义的产品功能，自动生成的 MCU 代码。在此基础上进行修改、补充，可快速完成 MCU 程序。

1.1 开发步骤：

- 1、需要根据产品实际情况（重置 bluetooth 按钮和 bluetooth 状态指示灯处理方式、是否支持 MCU 升级等）进行配置，请在 protocol.h 内修改此配置；
- 2、移植此 MCU_SDK，请查看 protocol.c 文件内的移植步骤，并正确完成移植。移植后，请完成数据下发处理、数据上报部分的代码，即可完成全部 bluetooth 功能。

1.2 文件概览：

此 MCU_SDK 包括 9 个文件：

- protocol.h 和 protocol.c 是需要你修改的。protocol.h 和 protocol.c 文件内有详细修改说明，请仔细阅读。
- bluetooth.h 文件为总的.h 文件，如需要调用 bluetooth 内部功能，请 #include "bluetooth.h"。
- system.c 和 system.h 是 bluetooth 功能实现代码，用户无需修改。
- mcu_api.c 和 mcu_api.h 内实现全部此用户需调用函数，用户无需修改。
- mcu_ota_handler.h 和 mcu_ota_handler.c 是你需要修改的。ota 和芯片强相关，需要用户自行调试修改适用自己的芯片平台，这两个文件仅作为一个参考

2 第一步：初始化

- 1、在需要使用到 bluetooth 相关文件的文件中 include “bluetooth.h”
- 2、在 MCU 初始化中调用 mcu_api.c 文件中的 bt_protocol_init() 函数
- 3、将 MCU 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 例如

```
1 void uart_transmit_output(unsigned char value)
2 {
3     uart_send(&value,1);
4 }
```

- 4、在 MCU 串口接收函数中调用 mcu_api.c 文件内的 uart_receive_input 函数, 并将接收到的字节作为参数传入例如

```
1 void uart_isr(void)
2 {
3     uint32_t IntStat;
4
5     IntStat = uart_isr_stat_get();
6
7     if(uart_rx_fifo_need_rd_isr_getf() || uart_rx_end_isr_getf() ||
8        uart_rxd_wakeup_isr_getf())
9     {
10         while((REG_APB3_UART_FIFO_STAT & (0x01 << 21)))
11         {
12             uint8_t ch = UART_READ_BYTE();
13             uart_receive_input(ch);
14             .....
15         }
16         .....
17     }
18 }
```

- 5、单片机进入 while 循环后调用 mcu_api.c 文件内的 bt_uart_service() 函数例如

```
1 void main_loop(void)
2 {
3     while(1)
4     {
5         bt_uart_service();
6         .....
7     }
8 }
```

3 第二步：实现具体用户函数

3.1 APP 下发数据处理

例如

```
1 static unsigned char dp_download_switch_1_handle(const unsigned char
   value[], unsigned short length)
2 {
3     //示例:当前DP类型为BOOL
4     unsigned char ret;
5     //0:关/1:开
6     unsigned char switch_1;
7
8     switch_1 = mcu_get_dp_download_bool(value,length);
9     if(switch_1 == 0)
10    {
11        //开关关
12    }
13    else
14    {
15        //开关开
16    }
17
18    //处理完DP数据后应有反馈
19    ret = mcu_dp_bool_update(DPID_SWITCH_1,switch_1);
20    if(ret == SUCCESS)
21        return SUCCESS;
22    else
23        return ERROR;
24 }
25 .....
```

3.2 数据上报处理

例如

```
1  uint8_t key_onOff_status;
2  uint8_t raw_test[4];
3  uint32_t value_test;
4  uint8_t string_test[16];
5  uint8_t enum_test;
6  void all_data_update(void)
7  {
8      //此代码为平台自动生成，请按照实际数据修改每个可下发可上报函数和只上
      报函数
9      mcu_dp_bool_update(DPID_SWITCH_1, key_onOff_status); //BOOL型数据上报;
10     mcu_dp_raw_update(DPID_RAW_TEST, raw_test, sizeof(raw_test)); //RAW型数
      据上报;
11     mcu_dp_value_update(DPID_INT_TEST, value_test); //VALUE型数据上报;
12     mcu_dp_string_update(DPID_STRING_TEST, string_test, sizeof(string_test)
      ); //STRING型数据上报;
13     mcu_dp_enum_update(DPID_ENUM_TEST, enum_test); //枚举型数据上报;
14 }
```



4 第三步：完成数据帧的选择

```

1  //
   =====
2  //数据帧类型
3  //
   =====

4  #define      HEAT_BEAT_CMD          0
   //心跳包
5  #define      PRODUCT_INFO_CMD       1
   //产品信息
6  #define      WORK_MODE_CMD          2
   //查询MCU 设定的模块工作模式
7  #define      BT_STATE_CMD           3
   //bluetooth工作状态
8  #define      BT_RESET_CMD           4
   //重置bluetooth
9  #define      DATA_QUERT_CMD        6
   //命令下发
10 #define      STATE_UPLOAD_CMD       7
   //状态上报
11 #define      STATE_QUERY_CMD        8
   //状态查询
12 ///////////////////////////////////////////////////////////////////当前MCU SDK版本较上一版本新增支持协议
   接口/////////////////////////////////////////////////////////////////
13 //如果不需要某一条命令接口，直接注释掉命令宏的定义，相关代码将不会被编
   译，以减小代码空间
14 #define TUYA_BCI_UART_COMMON_UNBOUND_REQ          0x09
   //模块解绑
15 #define TUYA_BCI_UART_COMMON_RF_TEST             0x0E
   //rf射频测试
16 #define TUYA_BCI_UART_COMMON_SEND_STORAGE_TYPE    0xE0
   //记录型数据上报（离线缓存）
17 #define TUYA_BCI_UART_COMMON_SEND_TIME_SYNC_TYPE  0xE1
   //获取实时时间
18 #define TUYA_BCI_UART_COMMON_MODIFY_ADV_INTERVAL  0xE2
   //修改休眠模式广播间隔
19 #define TUYA_BCI_UART_COMMON_TURNOFF_SYSTEM_TIME  0xE4
   //关闭系统时钟功能
20 #define TUYA_BCI_UART_COMMON_ENANBLE_LOWER_POWER  0xE5
   //低功耗使能
21 #define TUYA_BCI_UART_COMMON_SEND_ONE_TIME_PASSWORD_TOKEN 0xE6
   //获取一次性动态密码匹配结果
22 #define TUYA_BCI_UART_COMMON_ACTIVE_DISCONNECT    0xE7
   //断开蓝牙连接
23
24 #define TUYA_BCI_UART_COMMON_QUERY_MCU_VERSION    0xE8
   //查询MCU版本号
25 #define TUYA_BCI_UART_COMMON_MCU_SEND_VERSION    0xE9
   //MCU主动发送版本号
26
27 #define TUYA_BCI_UART_COMMON_MCU_OTA_REQUEST      0xEA
   //OTA升级请求
28 #define TUYA_BCI_UART_COMMON_MCU_OTA_FILE_INFO    0xEB
   //OTA升级文件信息
29 #define TUYA_BCI_UART_COMMON_MCU_OTA_FILE_OFFSET  0xEC
   //OTA升级文件偏移请求

```


5 第四步：配置固件版本信息

```
1 //protocol.h
2 #define MCU_VER "1.0.0" //用户的软件版本,用于MCU固件升级,MCU升级版本需修改
3 #define MCU_APP_VER_NUM 0x010000 //用户的软件版本1.0.0
4 #define MCU_HARD_VER_NUM 0x010000 //用户的硬件版本,当前没有实际用处
```

6 第五步：完善接口

protocol.c 有许多接口需要用户自行完善例如

```
1 void bt_rf_test_result(unsigned char result,signed char rssi)
2 {
3     #error "请自行完善该功能,完成后请删除该行"
4     if(result == 0)
5     {
6         //测试失败
7     }
8     else
9     {
10        //测试成功
11        //rssi为信号强度，一般大于-70dbm 蓝牙信号都在正常范围内
12    }
13
14 }
```

7 第六步：完善 MCU OTA 功能

mcu_ota_handler.h 和 mcu_ota_handler.c 是你需要修改的。ota 和芯片强相关，需要用户自行调试修改适用自己的芯片平台，这两个文件仅作为一个参考

```

1  /*
    *****

2  函数名称 : mcu_flash_init
3  功能描述 : flash初始化函数
4  输入参数 :
5
6  返回参数 : 无
7  使用说明 : 用户需要将flash初始化函数在此完善，如果在其他处有flash初始化
    操作，该函数可以不被调用
8  *****
    */
9  uint8_t mcu_flash_init(void)
10 {
11     #error "请自行完善该功能,完成后请删除该行"
12 }
13 /*
    *****

14 函数名称 : mcu_flash_erase
15 功能描述 : flash擦除函数
16 输入参数 : addr 地址 size 大小
17
18 返回参数 : 无
19 使用说明 : 用户自行完善
20 *****
    */
21 uint8_t mcu_flash_erase(uint32_t addr,uint32_t size)
22 {
23     #error "请自行完善该功能,完成后请删除该行"
24 }
25 /*
    *****

26 函数名称 : mcu_flash_write
27 功能描述 : flash写函数
28 输入参数 : addr 地址 size 大小 p_data待写入数据地址
29
30 返回参数 : 无
31 使用说明 : 用户自行完善
32 *****
    */
33
34 uint8_t mcu_flash_write(uint32_t addr, const uint8_t *p_data, uint32_t
    size)
35 {
36     #error "请自行完善该功能,完成后请删除该行"
37 }
38
39 /*
    *****

40 函数名称 : mcu_flash_read
41 功能描述 : flash读函数
42 输入参数 : addr 地址 size 大小 p_data待读出数据地址
43
44 返回参数 : 无
45 使用说明 : 用户自行完善
46 *****
    */

```

如果需要写 boot 标志位，可以添加到 on_dfu_complete 函数中。