FR801xH 错误码参考

Bluetooth Low Energy SOC

www.freqchip.com

Contents

1 综述

本文档旨在介绍 801xH 软件 SDK 各组件涉及到的错误码。

801xH SDK 包含以下组件：

- BLE 5.0 协议栈和常见 Profile
- BLE SIG Mesh 协议栈
- 多个中间件组件
- 非抢占式操作系统
- 保持链接睡眠和关机睡眠调用接口
- 多种外设驱动
- 调试函数和错误处理
- 系统常用辅助函数

下面章节将逐一介绍各组件定义的错误码所代表的含义。

2 错误码解释

2.1 BLE 5.0 协议栈

在调用头文件"gap_api.h" 和"gatt_api.h"定义的协议栈组件函数时，如果打开协议栈 lib 库底层的运行日志，日志运行时会按如下形式给出操作的执行结果。

```
Start advertising...
gapm_cmp_evt_handler: operation = 27, status = 0x00.
gapm_cmp_evt_handler: operation = 167, status = 0x00.
gapm_cmp_evt_handler: operation = 160, status = 0xa2.
```

Fr801x H 协议栈 lib 库运行日志

GAP 操作执行结果日志采用这种格式: gapm_cmp_evt_handler: operation = OP_ID, status = ERR_CODE.

其中 OP_ID 是操作码，ERR_CODE 是执行结果代码。

2.1.1 OP_ID

常见的 OP_ID 如下表所示

OP_ID	操作名称	含义
1	GAPM_RESET	协议栈重启
3	GAPM_SET_DEV_CONFIG	协议栈配置
23	GAPM_RESOLV_ADDR	执行 Resolve addr 解析动作
	…	
160	GAPM_CREATE_ADV_ACTIVITY	底层创建一个广播动作。
161	GAPM_CREATE_SCAN_ACTIVITY	底层创建一个扫描动作。

162	GAPM_CREATE_INIT_ACTIVITY	底层创建一个主动连接动作。
163	GAPM_CREATE_PERIOD_SYNC_ACTIVITY	底层创建一个周期性同步动作。
164	GAPM_START_ACTIVITY	底层启动一个动作
165	GAPM_STOP_ACTIVITY	底层停止一个动作
166	GAPM_STOP_ALL_ACTIVITIES	底层停止所有动作
167	GAPM_DELETE_ACTIVITY	底层删除一个动作
168	GAPM_SET_ADV_DATA	底层设置广播动作的 adv data
169	GAPM_SET_SCAN_RSP_DATA	底层设置广播动作的 adv rsp data
170	GAPM_SET_PERIOD_ADV_DATA	底层设置周期性广播的 adv data

2.1.2 ERR_CODE

ERR_CODE 是 OP_ID 执行的结果反馈。常见的 ERR_CODE 被定义在 components\ble\include\ble_hl_error.h 头文件内，下面逐一解释如下。

ERR_CODE	名称	含义
0x0	GAP_ERR_NO_ERROR	执行成功，无错误

ERR_CODE	名称	含义
0x1	ATT_ERR_INVALID_HANDLE	GATT 的操作的 handler 号无效

解决办法：

1 在 client profile 事件处理回调函数，动作完成分支，如果动作为 GATT_OP_PEER_SVC_REGISTERED，打印扫描到的 UUID 对应的 handler 号是否为非 0，如果为 0，表示该 UUID 对应的 handler 没有扫描到，对该 UUID 对应的属性操作时，会产生上述错误。

示例代码如下

```
case GATTC_MSG_CMP_EVT:
{
    if(p_msg->param.op.operation == GATT_OP_PEER_SVC_REGISTERED)
    {
        uint16_t att_handles[2];      //只定义了 2 个感兴趣的 UUID，缓存它们的 handler 号
        memcpy(att_handles,p_msg->param.op.arg,4);     //拷贝上传的 handler 号
        show_reg((uint8_t *)att_handles,4,1);      //打印 UUID 对应的 handler 号
    }
}
```

2 检查 client_att_table 定义的 UUID，是否存在于对端的服务中。

ERR_CODE	名称	含义
0x2	ATT_ERR_READ_NOT_PERMITTED	GATT 的读操作禁止

解决办法：

1 检查进行读操作的 UUID，在对端服务中，有没有定义读操作的权限。

ERR_CODE	名称	含义
0x3	ATT_ERR_WRITE_NOT_PERMITTED	GATT 的写操作禁止

解决办法：

1 检查进行读操作的 UUID，在对端服务中，有没有定义写操作的权限。

ERR_CODE	名称	含义
0x40	GAP_ERR_INVALID_PARAM	GAP 操作输入参数非法

解决办法：

1 查找 GAP 操作对应的 OP_ID，找到执行该操作的 API 函数，检查输入的参数是否合法。

ERR_CODE	名称	含义
0x42	GAP_ERR_NOT_SUPPORTED	GAP 操作因为协议栈配置导致不支持

解决办法：

1 查找 GAP 操作对应的 OP_ID，询问 FAE 人员，协议栈底层 lib 库协议栈配置是否支持该 GAP 操作。

ERR_CODE	名称	含义
0x43	GAP_ERR_COMMAND_DISALLOWED	GAP 操作不被允许

解决办法：

1 通常该错误原因是，有同一个 GAP 操作正在执行，此时应用层再次调用该 GAP 操作，导致后面的 GAP 操作执行结果报这个错误。需要应用层检查是否存在前一个操作未完成的情况下，再次调用相同的操作。

ERR_CODE	名称	含义
0x44	GAP_ERR_CANCELED	GAP 操作被取消

解决办法：

1 该 ERR_CODE 不代表错误，代表某个正在执行的 GAP 操作被提前终止掉，比如调用 gap_scan_stop()函数终止正在进行的扫描动作，就会上传该 err_code。

ERR_CODE	名称	含义
0x45	GAP_ERR_TIMEOUT	GAP 操作超时导致被终止

解决办法：

1 找到该操作 OP_ID，查找调用该操作的 gap API 函数，是否有设置执行时间，比如启动广播的函数 gap_start_advertising(uint8_t)输入参数即为动作执行时间，如果非 0，广播时间到后，广播动作会终止，然后上传该 ERR_CODE

ERR_CODE	名称	含义
0x46	GAP_ERR_DISCONNECTED	GAP 操作被终止因为链接已经断开

解决办法：

1 该 ERR_CODE 不需要做进一步的纠错动作，这是底层通知某个 GAP 或 GATT 操作因为链接断开而被提前终止，比如调用 gap_conn_param_update(…) API 进行链接参数更新时，如果在参数更新动作完成之前，链接断开，底层会打印该 ERR_CODE 的日志。

ERR_CODE	名称	含义
0x48	GAP_ERR_REJECTED	GAP 操作被对端拒绝

解决办法：

1 查找产生该 ERR_CODE 的 OP_ID，找到调用该操作的 API 函数，某些操作被拒绝后会上传到 GAP 或 GATT 的回调函数内，应用层需要做进一步的处理。比如调用 gap_conn_param_update(…) API 进行链接参数更新时，如果在参数被对端拒绝，底层会上传链接参数被拒绝的事件，应用层需要用新的链接参数进行更新或终止参数更新。

示例

```
case GAP_EVT_LINK_PARAM_REJECT:
        gap_evt_link_param_reject_t *cs = &(event->param.link_reject);
        if(cs->status == GAP_ERR_COMMAND_DISALLOWED)
                break;
      if(cs->status != GAP_ERR_REJECTED && cs->status != LL_ERR_UNACCEPTABLE_CONN_INT
          && cs->status != GAP_ERR_INVALID_PARAM)
            os_timer_start(&update_param_timer,3000,0);
      else
            os_timer_stop(&update_param_timer);
break;
```

该示例中，如果链接参数更新失败，原因是 GAP_ERR_REJECTED 或 GAP_ERR_INVALID_PARAM，则停止更新参数的定时器。否则启动更细参数定时器，继续更新链接参数。

ERR_CODE	名称	含义
0x4A	GAP_ERR_ADV_DATA_INVALID	设置广播 Adv data 或 adv response data 非法

解决办法：

1 检查调用 void gap_set_advertising_data(uint8_t *p_adv_data, uint8_t adv_data_len); 和 void gap_set_advertising_rsp_data(uint8_t *p_rsp_data, uint8_t rsp_data_len);设置广播数据和广播回复数据时，数据格式是否有重复的字段，数据的总长度是否超过限制。

针对经典广播，adv 的数据长度最长为 0x1F - 3。 Adv rsp data 的数据长度最长为 0x1F。

ERR_CODE	名称	含义
0x75	SMP_ERROR_REM_PAIRING_NOT_SUPP	对端不支持绑定操作。

解决办法：

1 该 ERR_CODE 表明，对端设备不支持绑定操作

ERR_CODE	名称	含义
0x77	SMP_ERROR_REM_CMD_NOT_SUPPORTED	本机不支持某个绑定的子动作。

解决办法：

1 该 ERR_CODE 表明，本地设备不支持某个绑定子操作

ERR_CODE	名称	含义
0x78	SMP_ERROR_REM_UNSPECIFIED_REASON	因为某个原因绑定动作失败。

解决办法：

1 该 ERR_CODE 表明，因为某个原因绑定动作失败，进一步的处理，请咨询 FAE 人员。

ERR_CODE	名称	含义
0x9B	LL_ERR_ACL_CON_EXISTS	针对同一个设备的链接已经存在了。

解决办法：

1 该 ERR_CODE 表明，应用层发起主动连接动作去连一个已经建立链接的对端设备。应用层需要检查做主机时主动连接的函数是否在连接一个已经链接上的设备。设备由 mac 地址区分。

ERR_CODE	名称	含义

0x9D	LL_ERR_CONN_REJ_LIMITED_RESOURCES	因为内存不足，导致主动连接动作失败。

解决办法：

1 该 ERR_CODE 表明，内存不够了，导致主机去主动连接时，内部分配内存资源失败。应用层可以参考《FR8010X H 内存泄漏调试》来查看是否有内存泄漏的问题。如果没有内存泄漏，应用层通过调用操作系统组件"os_mem.h"定义的 uint16_t os_get_free_heap_size(void) 打印系统剩余的内存大小，查看是否存在剩余内存过小的问题。

ERR_CODE	名称	含义
0xA2	LL_ERR_INVALID_HCI_PARAM	发送到协议栈链路层的 hci 命令参数非法。

解决办法：

1 该 ERR_CODE 表明，协议栈 host 层发送到链路层的 hci 命令的参数不合法。具体的原因请咨询 FAE 人员

ERR_CODE	名称	含义
0xA8	LL_ERR_PAIRING_NOT_ALLOWED	绑定终止因为链路层不允许进行绑定操作。

解决办法：

1 该 ERR_CODE 表明，绑定操作结束，因为本地设备链路层配置不支持绑定

2.1.3 链接断开错误码

Fr801xH SDK 框架下应用程序会遇到各种断开链接的原因，在设置了 GAP 事件回调函数后，链接断开的错误码信息会被上传。该错误码是单独编码，不属于上面介绍的 GAP 和 GATT 的错误码集合。

示例代码

```
case GAP_EVT_DISCONNECT:
{
        co_printf("Link[%d] disconnect,reason:0x%02X\r\n",p_event->param.disconnect.conidx
                                ,p_event->param.disconnect.reason);
}
break;
```

打印的 log 信息示例：Link[0],disconnect,reason:0x08. 其中 Link[0]表示链接号为 0 的链接断开，断开的错误码是 0x08.

下面分别介绍常见的断开链接错误码以及对应的处理措施。

ERR_CODE	名称	含义
0x08	LL_ERR_CON_TIMEOUT	链接因为握手超时导致断开。

解决办法：

1 该 ERR_CODE 表明，链接因为握手连续失败，达到链接超时断开时间，底层主动断开链接。握手失败的可能原因是：a 链接参数设置的链接间隔时间太长，超过 1 秒，应用层软件通过链接参数更新 API 减少握手间隔。b 硬件天线的频偏与匹配参数没有调试过，需要咨询 FAE 人员如何调试这两个天线性能参数。c SDK 版本太旧，射频参数没有更新。需要从 git hub 上下载最新 Fr801x H SDK。

ERR_CODE	名称	含义
0x13	LL_ERR_REMOTE_USER_TERM_CON	链接被对端设备断开。

解决办法：

1 该 ERR_CODE 表明，某个链接被对端设备主动断开。

ERR_CODE	名称	含义
0x16	LL_ERR_CON_TERM_BY_LOCAL_HOST	链接被对本地设备自己主动断开。

解决办法：

1 该 ERR_CODE 表明，某个链接被本地设备主动断开，在应用层调用了 void gap_disconnect_req(uint8_t conidx) API 断开某个链接后，会打印该 log 信息。

ERR_CODE	名称	含义
0x1F	LL_ERR_UNSPECIFIED_ERROR	握手接收失败，导致开窗过大，底层主动断开链接。

解决办法：

1 该 ERR_CODE 表明，链接被底层主动断开，原因是握手失败导致开窗过大。解决办法参考 ERR_CODE:0x08 的解决办法

ERR_CODE	名称	含义
0x22	LL_ERROR_LMP_RSP_TIMEOUT	发送的控制包未收到对端回复超时(默认是 40 秒)，主动断开。 或发送断开链接包之后，链接超时时间内没有收到对端的 ack。

解决办法：

1 该 ERR_CODE 表明，对端在规定时间 40s 内没有回复控制包，或者链接超时时间没有 ack 本地主动断开链接的包。

ERR_CODE	名称	含义
0x3D	LL_ERROR_TERMINATED_MIC_FAILURE	执行加密操作时，发现密码不对,主动断开

解决办法：

1 该 ERR_CODE 表明，对链接进行加密操作时，对端的密码不对，底层主动断开链接。查看

a) 做为主机有没有针对未绑定设备直接进行加密链接操作，做为主机有没有调用绑定管理的绑定信息删除函数，删掉绑定信息。有没有更改绑定信息存储 flash 的起始地址。

b) 做为从机时，有没有调用绑定管理的绑定信息删除函数，删掉绑定信息。有没有更改绑定信息存储 flash 的起始地址。

ERR_CODE	名称	含义
0x3E	LL_ERROR_CONN_FAILED_TO_BE_EST	建立链接时，没有收到第一包数据，建立链接失败，主动断开。

解决办法：

1 该 ERR_CODE 表明，在链接建立时，未收到对端设备第一包的数据，导致链接没有建立起来。此种问题很可能跟天线射频性能有关系，参考 ERR_CODE:0x08 的处理办法，调试天线频偏和匹配参数，同时更新 SDK 到最新版本。

2.1.4 创建 profile 失败错误码

应用层调用 proifile 创建函数 uint8_t gatt_add_service(gatt_service_t *p_service) 和 uint8_t gatt_add_client(gatt_client_t *p_client)后，返回值定义如下

返回值	名称	含义
0xff	PRF_ID_INVALID	Profile 创建失败。

解决办法：

1 检查应用层定义的 Profile 总数有没有超过 8 个，Profile 创建个数超过 8 个后，再创建 Profile 就会返回该值。

返回值	名称	含义
0~7	PRF_ID	Profile 创建成功，返回被分配的 profile ID 号

2.2 操作系统组件

该组件在调用任务创建函数 uint16_t os_task_create(os_task_func_t task_func)后，返回值定义如下

返回值	名称	含义
0xff	TASK_ID_FAIL	任务创建失败。

解决办法：

1 检查应用层定义的任务总数有没有超过 50 个，任务数超过 50 个后，再创建任务就会返回该值。

2 系统剩余内存不足，导致创建任务失败。调用 uint16_t os_get_free_heap_size(void) 查看系统剩余的内存大小。

返回值	名称	含义
0~50	TASK_ID	任务成功，返回被分配的任务 ID 号