

目录

设备各配网模式对应设置及常见配网问题	1
1. 不同通信类型对应模组设置	1
1) WiFi	1
2) WiFi+BLECombo	1
3) WiFi+BLE+SLE	1
2. 产品配网常见的问题	2
1) 智慧生活 APP 添加设备时扫描不到设备，但可以扫到模组发出的蓝牙广播	2
2) 连接设备时注册失败，提示该设备未经过华为官方认证，该设备 MAC 非法	3
3) 连接设备时提示 0%连接失败	3
4) 连接设备时注册失败，提示 1.确保手机蓝牙已开启	4
5) 设备添加注册成功，点击设备进入 H5 页面显示网络异常没有控制界面	4
6) 双联双控设备稍后配置网络，本地控制界面按钮是灰色，按下没反应	4
7) softap 配网的产品配置 WiFi 信息时 H5 页面出现错误	5
8) 设备添加注册成功后出现间断掉线的问题	5
9) 设备配网上线后，H5 页面显示的属性数值异常	6
SDK API 接口与使用问题	7
1. 修改串口波特率	7
2. PWM 输出 接口的使用	7
3. GPIO 模式配置及常见问题	8
4. 开启蓝牙扫描，以及常见的问题	8
5. 修改蓝牙广播类型及功率	9
6. 修改 WiFi 和蓝牙发射功率	9
7. ntp 服务器获取时间超时	11
8. 设备版本信息和蓝牙广播名称字段修改	11
9. hilink 上报下发数据格式问题导致程序崩溃	12
10. 重置配网接口 HILINK_RestoreFactorySettings()使用注意事项	12
12. 使用 printf 函数无法打印浮点数	12
13. 硬件问题导致异常崩溃	12
14. 在中断处理里面修改中断优先级，偶现系统卡死崩溃等问题	12
15. 版本不一致导致星闪 client 扫不到星闪广播	13
环境搭建和编译常见问题	14
1. 更新 SDK 版本，替换更新文件后编译报错，提示找不到文件路径	14
2. 从主机复制文件到 SDK 中，编译报错提示找不到该文件	14
3. 编译报错缺少关键组件	15
4. 编译报错编解码解析失败	15

设备各配网模式对应设置及常见配网问题

1. 不同通信类型对应模组设置

创建产品时分别有 WiFi, WiFi+BLECombo, WiFi+BLE+SLE 三种通信类型可以选

1) WiFi

通信类型选择 WiFi 时，极简连接方式可选：

The screenshot shows the 'Communication Type' (通信类型) section with 'Wi-Fi' selected. The 'Simplified Connection' (极简连接) dropdown menu is open, showing 'Ultra-fast standard network' (极速常规配网) as the selected option.

2) WiFi+BLECombo

通信类型选择 WiFi+BLECombo 时，极简连接方式可选：

The screenshot shows the 'Communication Type' (通信类型) section with 'Wi-Fi+BLECombo' selected. The 'Simplified Connection' (极简连接) dropdown menu is open, showing 'Ultra-fast standard network' (极速常规配网), 'Bluetooth auxiliary network' (蓝牙辅助配网), and 'Dual-link dual-control (Wi-Fi/BLE+BLE)' (双联双控 (Wi-Fi/蜂窝+BLE)).

3) WiFi+BLE+SLE

通信类型选择 WiFi+BLE+SLE 时，极简连接方式可选：

The screenshot shows the 'Communication Type' (通信类型) section with 'Wi-Fi+BLE+SLE' selected. The 'Simplified Connection' (极简连接) dropdown menu is open, showing 'Ultra-fast standard network' (极速常规配网), 'Bluetooth auxiliary network' (蓝牙辅助配网), 'Auxiliary network (BLE+SLE)' (辅助配网 (BLE+SLE)), 'Dual-link dual-control (Wi-Fi/BLE+BLE)' (双联双控 (Wi-Fi/蜂窝+BLE)), and 'Dual-link dual-control (Wi-Fi/BLE+SLE)' (双联双控 (Wi-Fi/蜂窝+BLE+SLE)).

上述极简连接的类型分别对应不同的 protocolType，可以通过 AT+MODE 命令选择相对应的配网模式

AT+MODE：设置/查询 hilink 配网方式

■ 格式：

◆ 查询 AT+MODE\r\n

+ok=<mode>

◆ 设置 AT+MODE=<mode>\r\n

+ok

■ 参数:

◆ mode: 配网方式

- 0: 蓝牙辅助配网
- 1: 双联双控 (Wi-Fi/蜂窝+BLE)
- 2: 双联双控 (Wi-Fi/蜂窝+BLE+SLE)
- 3: 极速常规配网
- 4: 辅助配网 (BLE+SLE)

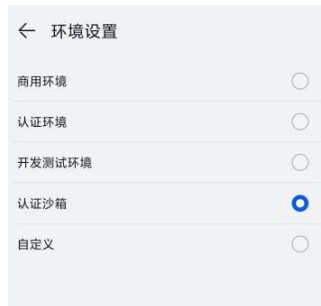
2. 产品配网常见的问题

1) 智慧生活 APP 添加设备时扫描不到设备，但可以扫到模组发出的蓝牙广播

此现象通常有以下四种状况

a. 智慧生活 app 环境设置非认证沙箱环境

设置方法:我的-->设置-->关于-->环境设置



b. 平台信息没同步，刷新网页和清理智慧生活 APP 的缓存之后即可解决，这种情况通常出现在平台新建的产品上。

c. 模组内部有保存上次产品的配网信息，此次发送的蓝牙广播为不可被扫描到的广播类型，这个时候在烧录固件的时候选择 **erase all**，或者发送 **AT+RESET** 重置。

d. 蓝牙广播名称非法，设备无法被识别，需要特别注意 空格 也属于非法字符。

- 2) 连接设备时注册失败，提示该设备未经过华为官方认证，该设备 MAC 非法



此现象为模组的 MAC 未录入到华为的 DP 平台，联系相关人员进行模组 MAC 录入即可解决。

- 3) 连接设备时提示 0%连接失败



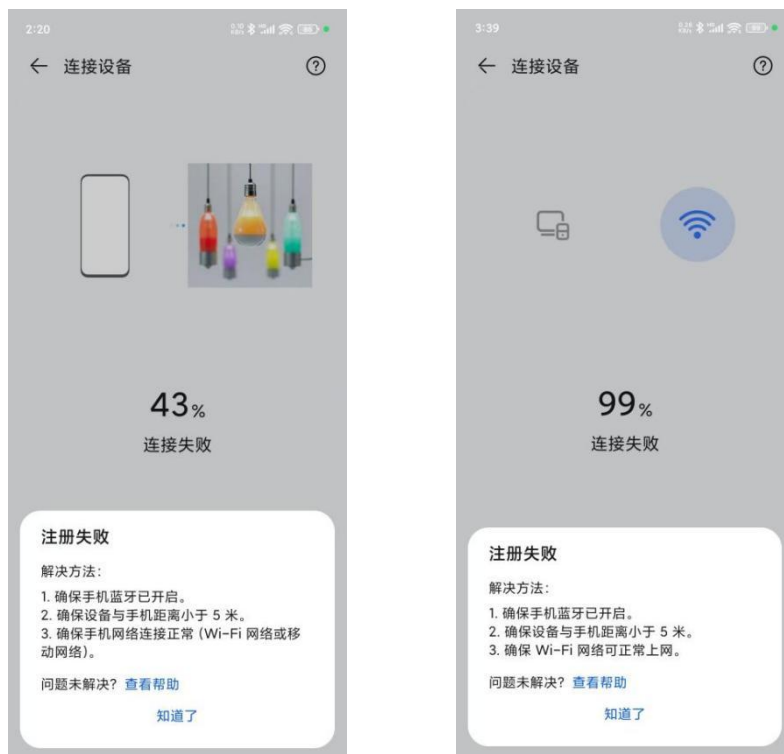
此现象是没有在 DP 勾选自动 Pin 码校验，解决办法：产品开发--->集成开发--->SDK 下载 & 集成开发：勾选 pin 码校验选项

SDK 下载 & 集成开发

蓝牙广播设置



4) 连接设备时注册失败，提示 1.确保手机蓝牙已开启.....



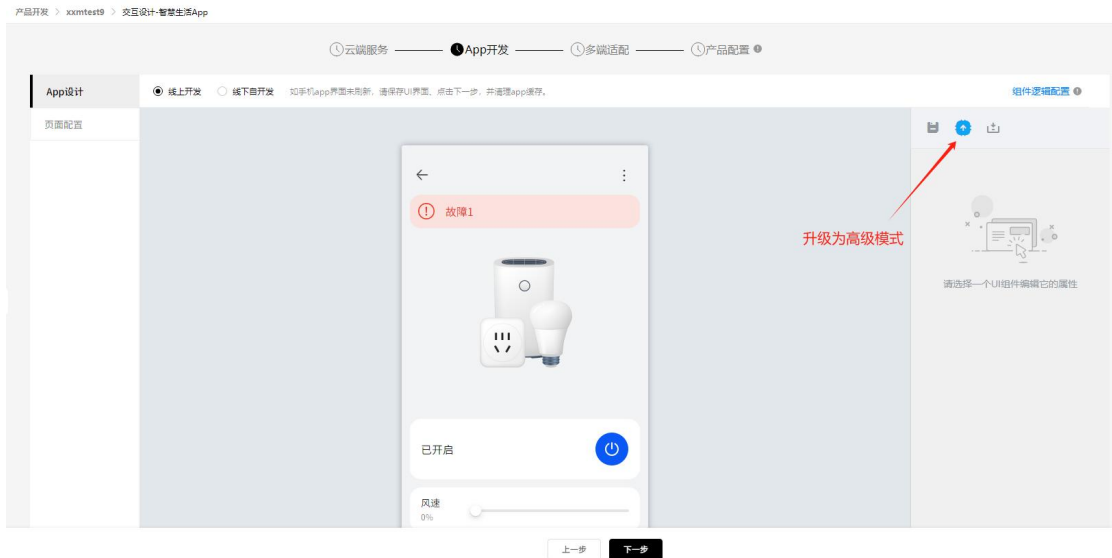
- a. 在 44% / 99% 左右注册失败，此状况常为 MODE 对应的 ProtType 和平台定义不匹配，参考上一章节检查后重新配置;
 - b. BLE 辅助配网产品在 80%注册超时失败;此情况常为开始配网到模组上电的时间的超过了在 hilink_main.c 首次调用 BLE_CfgNetAdvCtrl();设置的广播时间。超过此广播时间后面再开广播 hilink 还会认为配网已经结束，导致实际配网超时，如未修改此处可忽略;
- 5) 设备添加注册成功，点击设备进入 H5 页面显示网络异常没有控制界面



此现象常为平台数据未同步，多次刷新网页重复保存页面可以解决，清理智慧生活 APP 缓存。个别带有 SLE 的产品由于 UI+侧尚未完全支持 SLE 设备的适配也会出现此现象。

6) 双联双控设备稍后配置网络，本地控制界面按钮是灰色，按下没反应

此现象常为本地控制 H5 界面没有创建。解决方法：DP 平台 产品开发-->交互设计 app 开发，点击升级为高级模式重新保存即可。



7) softap 配网的产品配置 WiFi 信息时 H5 页面出现错误



此现象通常也是 MODE 设置的问题，softap 配网需要设置 MODE 等于 3，开启 WiFi 配网。

8) 设备添加注册成功后出现间断掉线的问题

出现这种现象通常都是在 hilink_device.c 中服务信息定义中参数和平台上定义的物模型参数有出入，检查比对程序与导出的 Profile 文件。

```

application > samples > wifi > ohos_connect > hilink_adapt > product > hilink_device.c > ...
21  /*
22  *
23  * 服务信息定义
24  * 注意: (1) 适配格式{svcType, svcId}
25  *        (2) 与devicepartner 平台物模型定义必须保持一致
26  */
27  static const HILINK_SvcInfo SVC_INFO[] = {
28      { "switch", "switch" },
29  #ifdef CONFIG_SUPPORT_HILINK_INDIE_UPGRADE
30      { "checkSum", "checkSum" },
31  #endif
32  };
33

```

个别品类建的产品没有网络信息属性，设备频繁掉线

现已知目前平台的车载品类新建产品默认没有网络信息属性，引发的现象是设备配网注册之后会出现频繁掉线

9) 设备配网上线后，H5 页面显示的属性数值异常

此情况是该属性的值未进行初始设置或设置的值不在取值范围内，对比检查 HILINK_GetCharState()函数中上报的属性值是否符合 Profile_XXXX.xlsx 文件中的数据类型和取值范围，特别注意 string 类型的数据上报时需要加” ”

设备类型	设备类型(中文)	服务 sid	服务(中文)	服务类型 ServiceType	属性	属性中文名称	属性英文名称	属性类型 Character Type	操作权限	数据类型	数据约束 (IT系统录入时使用)	取值范围	描述
Light Belt 灯带		brightness	亮度	brightness									
				brightness	亮度	Brightness	characteristic.br	GET/REPORT/PUT	int	整数-自定义取值范围	min:0 max:100 步长:1.0		
		colour	RGB#颜色	colour									
				blue	颜色	color	characteristic.co	GET/REPORT/PUT	int	整数-自定义取值范围	min:0 max:255 步长:1.0		
				green	颜色	color	characteristic.co	GET/REPORT/PUT	int	整数-自定义取值范围	min:0 max:255 步长:1.0		
				red	颜色	color	characteristic.co	GET/REPORT/PUT	int	整数-自定义取值范围	min:0 max:255 步长:1.0		
				white	颜色	color	characteristic.co	GET/REPORT/PUT	int	整数-自定义取值范围	min:0 max:255 步长:1.0		
		fadeTime	渐变时长	fadeTime									
		streamer	流光	streamer	value	流光时长	Fade Time	characteristic.ti	GET/REPORT/PUT	string	NA	StringLength:8	

如有设备上电时实际属性值不能及时获取情况，应在 HILINK_GetCharState()函数里直接 return -1；这样页面就会显示合法默认值，避免页面显示错误。

SDK API 接口与使用问题

1. 修改串口波特率

Q: 怎么从代码中修改串口的波特率?

A: 现以提供修改 uart0 波特率的接口: `hfset_uart_parameters()`, 立即生效需在 `hfnet_start_uart()` 前设置, 默认不保存 flash, 如需保存配置参数调用 `hfconfig_file_save()`;

设置 uart0 波特率:9600 数据位:8 停止位:1 无校验 参考代码:

```
hfset_uart_parameters(9600, COMBITS_8, COMSTOPBITS_1, COMPARITY_NONE);
```

串口 1 为 Debug 口不支持修改波特率, 固定 115200。

2. PWM 输出 接口的使用

Q: PWM 输出波形周期计算方法及相位怎么设置?

A: PWM 工作时钟频率参考 `uapi_pwm_get_frequency()`; 工作周期为 $T = 1 / \text{uapi_pwm_get_frequency}()$;

实际低电平时间 = $1000 * T$;

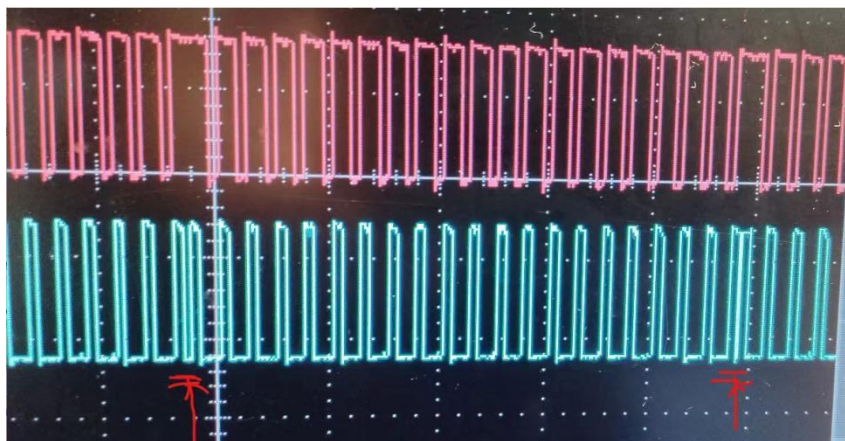
实际高电平时间 = $1000 * T$;

相位可简单理解为延迟输出的时间。

Q: PWM 输出频率小于 1K 时无法输出, 1.5K, 2K 就是正常的

A: PWM 时钟分频寄存器为 16 位寄存器, 最高储存数值 65535, 所以最高可设置频率是 $80M / 65535 = 1220\text{hz}$, 如果有低于该频率的输出需求, 可对 PWM 整体时钟进行分频操作, 详细步骤参考 `PWM_example.c`。

Q: 当更新 PWM 占空比时, 就会破坏当前 PWM 周期, 现象是 LED 出现闪烁, 进行调光时有卡顿感。



A: 占空比动态调整可使用 PWM 预配置接口 `uapi_pwm_config_preload()`; 当上一个 PWM 周期完成时, 此配置会自动加载。

此功能默认关闭, 使用此接口需要添加该配置

```

build > config > target_config > ws63 > menuconfig > acore > ws63_liteos_app_iot.config
291 #
292 # Config PWM
293 #
294 CONFIG_PWM_USING_V151=y
295 # CONFIG_PWM_USING_V150 is not set
296 CONFIG_PWM_GROUP_NUM=4
297 CONFIG_PWM_CHANNEL_NUM=8
298 CONFIG_PWM_PRELOAD=y # 打开PWM预配置功能
299 # end of PWM Configuration
300

```

3. GPIO 模式配置及常见问题

Q: 复制例程中的 GPIO 引脚配置代码，就改了引脚发现配置不生效。

A: 不同引脚的模式的复用信号对应功能不一定相同，配置引脚复用模式请查看

[WS63V100 SoC Wi-Fi、BLE 和 SLE Combo 芯片 用户指南_05.pdf](#) 6.1.2 软用管脚描述

Q: GPIO05 引脚电平中断，按钮电平测试正常，软件只能进一次中断回调。

A: 程序内部没有配置上拉，开发板可以正常使用，用户使用需要配置上拉或加上拉电阻。

4. 开启蓝牙扫描，以及常见的问题

开启蓝牙扫描功能: 在如下路径 config.py 文件中 defined 一栏中打开“NABLE_BLE_SCAN”的宏定义，还需在代码里添加扫描相关接口

```

build > config > target_config > ws63 > config.py
7 target = {
266 'ws63-liteos-app-iot': {
270 'defines': [
308 "SUPPORT_SOFTAP_NETCFG", # softAP配网
309 "SUPPORT_BLE_ANCILLAY_NETCFG", # ble辅助配网
310 # "SUPPORT_QUICK_NETCFG", # 快速配网
311 "CONFIG_SUPPORT_HILINK_INDIE_UPGRADE",
312 "CONFIG_DHCPGW",
313 " HSF ",
314 # "ENABLE_BLE_SCAN" #open ble scan
315 # "HF_MCU_OTA" #open mcu ota
316 ]

```

需要注意的问题: 周围每扫描到一次蓝牙广播都会触发回调，如果在蓝牙扫描回调函数中添加过多操作，如添加打印，串口数据发送等资源消耗的操作，有概率出现系统卡死 崩溃等问题，保证回调函数快进快出。如果必须使用打印的话，可添加添加过滤操作。

Q: 蓝牙扫描遥控器广播有丢包，可以通过那些办法优化？

A: 扫描广播丢包为正常现象，如丢包过多可尝试修改扫描间隔和窗口时间，确保扫描间隔小于遥控器广播的间隔

```

application > ws63 > user_main > app_main.c > ...
128 #ifdef ENABLE_BLE_SCAN
135 int hfble_start_scan(void)
136 {
137     int ret = 0;
138     gap_ble_scan_params_t ble_scan_param = {
139         .scan_interval = 0x40, //30*8/5*2*2,
140         .scan_window = 0x20, //30*8/5/2,
141         .scan_type = 0x00,
142         .scan_phy = GAP_BLE_PHY_1M,
143         .scan_filter_policy = 0x00,
144     };

```

扫描间隔和窗口时间

5. 修改蓝牙广播类型及靠近发现距离

Q: 启用蓝牙靠近发现是在什么地方，如何修改靠近发现的距离？

A: 如图示路径下，通过 BLE_SetAdvType() 函数设置 hilink 配网广播的广播类型，目前已经支持的广播类型有 拉取半模态卡片的靠近发现、蓝牙碰一碰以及常态广播三种

```
application > samples > wifi > oshos_connect > hilink_adapt > entry > hilink_ble_main.c > ...
261 static void HILINK_BT_StateChangeHandler(HILINK_BT_SdkStatus event, const void *param)
265 if (event == HILINK_BT_SDK_STATUS_SVC_RUNNING) {
274     /* 设置蓝牙广播格式，包括靠近发现、碰一碰等，下一次发送广播生效 */
275     BLE_SetAdvType(BLE_ADV_LOCAL_NAME);
276
277     /* BLE配网广播控制：参数代表广播时间，0：停止；0xFFFFFFFF：一直广播，其他：广播指定时间后停止，单位秒 */
278     (void)BLE_CfgNetAdvCtrl(BLE_ADV_TIME);
279 }
280 }
```

```
65 /* BLE广播类型定义 */
66 typedef enum {
67     BLE_ADV_DEFAULT, //新蓝牙靠近发现
68     BLE_ADV_NEARBY_V0, //拉取半模态卡片的靠近发现
69     BLE_ADV_ONEHOP, //蓝牙碰一碰
70     BLE_ADV_LOCAL_NAME, //常态广播
71     BLE_ADV_CUSTOM
72 } BLE_AdvType;
```

靠近发现的距离可通过修改广播的功率来修改，修改位置：上级路径下 device_profile.h 中修改 ADV_TX_POWER 宏的值来设置。

常用距离及功率设置参考：

ADV_TX_POWER	-2	-9	-13	-17
弹窗最远距离(单位：厘米)	30	50	100	300

注：此表仅为使用开发板无遮挡环境下测试结果，实际情况会因测量角度、遮挡等因素产生误差，表中数据仅供参考，产品开发以实际测试结果为准。

6. 修改 WiFi 和蓝牙发射功率

Q: 进行整机 OTA 的时候因设备信号遮挡 OTA 成功率不符合要求，有没有增大 WiFi 功率的方法？

A: 参考如下代码,示例代码中 target_fe_tx_power 为默认值,根据需求增加 target_power 和 limit_power 里成员的值

结构体成员	介绍	备注
hip_max_power	芯片最大发射功率（单位 0.1dB）	不可修改
target_power	分协议速率的目标功率（单位 0.5dB）	适当增大
limit_power	分信道的功率限制（单位 0.5dB）	适当增大
sar_power	比吸收率功率限制值	保持不变

fe_tx_power_type_t 成员介绍

```
#include "nv.h"
#include "nv_common_cfg.h"
#include "string.h"
```

```

// 国家码
typedef enum {
    KEY_ID_NORTH_AMERICA = 0x2053,
    KEY_ID_EUROPE = 0x2054,
    KEY_ID_JAPAN = 0x2055,
    KEY_ID_ASIA_PACIFIC = 0x2056
} KeyIdRegion;

int set_wifi_power_nv_info(void)
{
    int ret = 0;
    fe_tx_power_type_t now_fe_tx_power = {0};
    uint16_t nv_value_len = 0;
    uapi_nv_read(KEY_ID_ASIA_PACIFIC, sizeof(fe_tx_power_type_t), &nv_value_len, (uint8_t
*)&now_fe_tx_power);

    fe_tx_power_type_t target_fe_tx_power = {
        .chip_max_power = {230},
        .target_power = {
            34, 34, 34, 34, // 11b 协议
            30, 30, 30, 30, 30, 30, 30, 30, // 11g 协议
            30, 30, 30, 30, 30, 30, 30, 30, 30, 30, // 11n/ax20M
            30, 30, 30, 30, 30, 30, 30, 30, 30, 30, 22 // 11n/ax20M
        },
        .limit_power = { // 11b、11g、11n/ax20M、11n/ax40M
            34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34,
            34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34,
            34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34, 34,
        },
        .sar_power = {60, 60, 60}
    };
    if(memcmp(&now_fe_tx_power,&target_fe_tx_power,sizeof(fe_tx_power_type_t)) == 0)
        return 0;
    ret = uapi_nv_write(KEY_ID_ASIA_PACIFIC, (uint8_t *)&(target_fe_tx_power),
sizeof(fe_tx_power_type_t));
    printf("ret = %d\r\n",ret);
    return ret;
}

```

Q: 整机测试 app 上扫描设备发现比较慢，有没有增加蓝牙广播功率的方法

A: 参考如下方法

方法 1.路径 application\samples\wifi\ohos_connect\hilink_adapt\product\device_profile.h

修改 ADV_TX_POWER

方法 2.使用 AT+BTPOWER: 设置/查询蓝牙 BLE/SLE 功率，详细参考标准固件操作指南

7. ntp 服务器获取时间超时

Q 模组网络正常连接，使用 NTP 获取时间效果时好时坏，很长一段时间获取失败

A: 网络正常，但获取不到网络时间可尝试切换 ntp 服务器，二次开发工程在调用 `hfnntp_get_time()` 时选择服务器，通用固件可使用 `AT+NTPSER=...` 设置

8. 设备版本信息和蓝牙广播名称字段修改

Q: 设备详情页版本信息怎么改，对应代码什么位置？



A: `HILINK_GetDevInfo()` 中 "`devinfo->fwv`" 字段对应设备详情页版本信息，其中 `FIRMWARE_VER` 为自定义内容

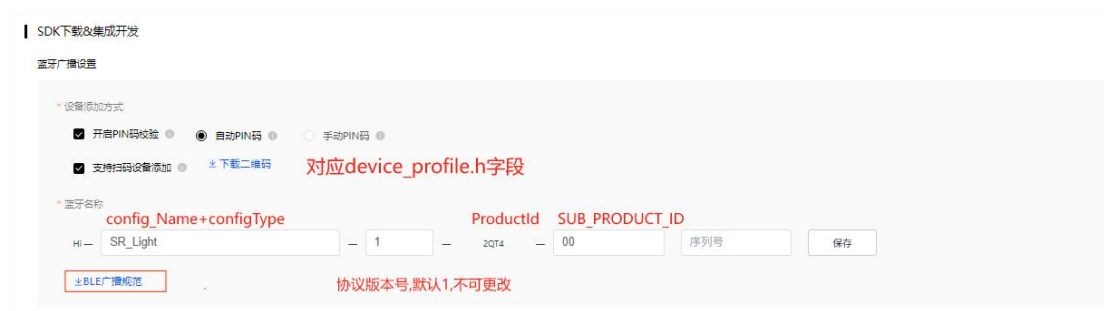
路径: `application\samples\wifi\ohos_connect\hilink_adapt\product\hilink_device.c`

```
application > samples > wifi > ohos_connect > hilink_adapt > product > C hilink_device.c > HILINK_GetDevInfo(HILINK_DevInfo *)
36 int HILINK_GetDevInfo(HILINK_DevInfo *devinfo)
85 if (err != EOK) {
89     HILINK_SAL_DEBUG("HILINK_GetDevInfo manuName: [%s]\r\n", devinfo->manuName);
90 #ifdef CONFIG_SUPPORT_HILINK_INDIE_UPGRADE
91     err = sprintf_s(devinfo->fwv, sizeof(devinfo->fwv), "%s_%s_%s", "103", "1.312", FIRMWARE_VER);
92     if (err <= 0) {
93         HILINK_SAL_ERROR("sprintf_s err:%d\r\n", err);
94         return -1;
95     }
96 #else
97     err = strcpy_s(devinfo->fwv, sizeof(devinfo->fwv), FIRMWARE_VER);
98     if (err != EOK) {
99         HILINK_SAL_ERROR("strcpy_s err:%d\r\n", err);
100         return -1;
101     }
102 #endif
103     HILINK_SAL_DEBUG("HILINK_GetDevInfo fwv: [%s]\r\n", devinfo->fwv);
```

Q: 不同品类设备我们的蓝牙名称有要求，请问这些字段应该在哪改？

A: 蓝牙广播名称对应程序位置:

`application\samples\wifi\ohos_connect\hilink_adapt\product\device_profile.h`



9. hilink 上报下发数据格式问题导致程序崩溃

Q: 设备初次配网后调用 `hilinkgetchar` 所有属性时有概率出现崩溃死机问题

A: `*outlen` 设置的过大, 在完成配网或者周期更新的时候, `hilink` 会 `get` 一次所有 `SID` 的属性状态, 如果 `*outlen` 设置的过大且 `sid` 属性过多会导致短时间 `malloc` 很多大块内存, 资源不足的情况下, 会出现程序崩溃卡死问题。

同时如果 `*outlen` 设置的过小, 组包时溢出开辟的空间, 缓冲区溢出也会导致程序崩溃。

10. 重置配网接口 `HILINK_RestoreFactorySettings()`使用注意事项

Q: 调用此接口之后模组只重启, 没有重置配网, `app` 添加设备扫描不到该产品。

A: 一种情况是调用太早, `hilink` 相关线程还没有起来, 调用会导致程序崩溃重启。

解决方法: 调用之前先获取一下 `hf_hilink_main_is_runing()` 的返回值, 等于 1 代表 `hilink` 相关线程启动完成。

还有一种情况是在硬件中断里面调用了这个接口, 如在 `GPIO` 引脚的中断处理函数和定时器回调函数里调用同样会导致程序崩溃重启。

11. uflash 地址冲突导致 mcu ota 之后, 手机上显示版本号回退 1.0.0 问题

Q: 进行 `MCU OTA` 升级, 升级完成后 `app` 显示的版本信息回退 1.0.0 版本

A: `MCU OTA` 为可选功能, 保存 `MCU OTA` 信息的 `uflash` 区域为 `0x00 ~ 0x1000`(共计 4k), 如果项目中其他地方使用了该地址就会导致数据覆盖, 该块区域保存的内容丢失。

解决方法: 如果用到 `MCU OTA`, 使用 `uflash` 时跳过 `0x00 ~ 0x1000` 区域

12. 使用 `printf` 函数无法打印浮点数

由于 `liteos` 底层适配性问题, `printf` 函数无法正常打印浮点类型参数, 如有相关需求可以使用 `sprintf` 函数将要打印的浮点数拼接转换成字符串再使用 `%s` 打印。

13. 硬件问题导致异常崩溃

Q: 同一批设备和固件, 在测试阶段配网后长稳挂机时, 某个特定设备出现崩溃问题

A: 测试阶段该设备的模组 `debug` 串口的 `rx` 口有外接出去, 后续挂机测试该口悬空, 导致模组串口收到垃圾数据, 引起的设备重启。

解决方法 1: 调试完成后硬件上给 `debug` 串口的 `rx` 口接高电平屏蔽干扰信号。

解决方法 2: 发送 `AT+NDBGL` 指令软件上关闭 `debug` 串口的 `rx` 口接收, 指令详细介绍参考标准固件操作指南。

Q: 使用开发板调试 同一份固件有块板子会一直重启

A: `LPT262` 模组在刚启动的时候, 功耗会比较高, 电源供电有波动的时候, 可能会把 3.3V 电压拉低, 低于 2.93V, 复位芯片就会触发模组复位, 导致模组不停的重启, 特别是通过电脑 `usb` 供电且拓展坞外接设备时。外接 5v 供电即可解决

14. 在中断处理里面修改中断优先级, 偶现系统卡死崩溃等问题

如果在中断执行过程中, 存在以下场景: 当前中断或者嵌套中断的执行过程中, 提高了当前或者嵌套中断优先级, 再来相同中断时, 不会置位 `pending`, 而是直接二次重入中断。

问题现象: 概率出现系统卡死重启问题

解决方法: 首先不建议系统运行过程中调整中断优先级。如果业务无法避免, 不要将调整中断优先级的操作放在中断中, 建议放在用户的线程中进行。

15. 版本不一致导致星闪 client 扫不到星闪广播

新版本星闪协议层升级(主要是 service 端广播格式有调整),当 server 设备升级到新版本时,旧版本 client 设备接收新广播解析失败,从而表现 1.1.102 版本星闪 client 无法扫描到 1.10.103 版本发出的星闪广播,更新 SDK 至新版本解决。

新旧版本区分日期为 25 年 6 月份,可在 debug 串口发送 AT+SYSINFO\r\n 查询当前 SDK 版本

```
601  » AT+SYSINFO
602  « SDK Version:1.10.103
```

环境搭建和编译常见问题

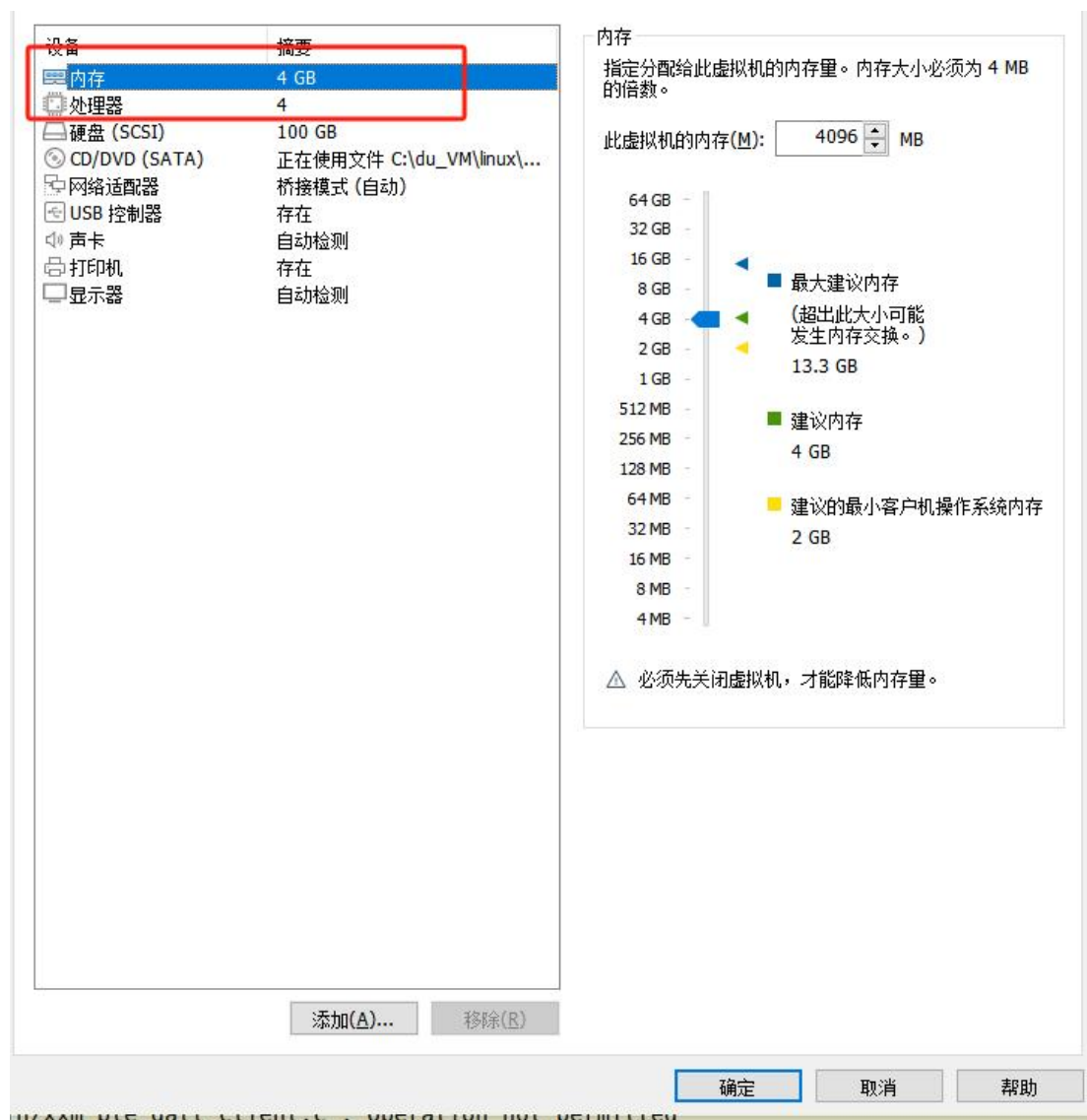
1. 更新 SDK 版本，替换更新文件后编译报错，提示找不到文件路径

Q: 下载新版本的 SDK，对比替换了有改动的文件夹再编译出现找不到文件的问题

A: 此现象因为在 `output` 文件夹下有一些中间文件保存了编译路径，移动 SDK 目录或者替换某些文件夹会出现这样的编译问题。

解决办法:删除 `output` 文件夹后，使用 `./indie_build.py all` 指令重新编译即可

另外首次编译比较慢也是因为要创建 `output` 文件，后续编译速度能相对快一些，如果追求更快的编译速度，可在虚拟机设置中调整虚拟机的内存大小和处理器内核数量



2. 从主机复制文件到 SDK 中，编译报错提示找不到该文件

Q: 从 windows 系统下复制一些库到工程里，修改了 `cmake` 和 `config.py` 文件但是还报错提示找不到文件。

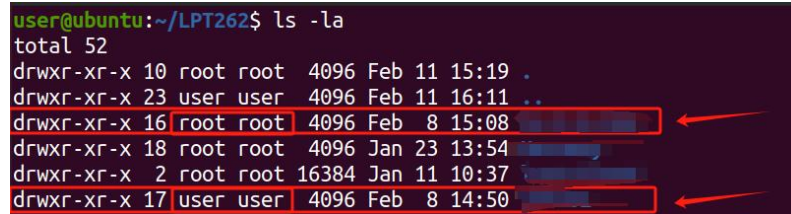
A: 一些虚拟机环境因为文件夹权限原因，复制进去的文件没有操作权限，在虚拟机终端中使用 `chmod` 指令修改该文件的权限。

例：修改当前路径下所有的文件权限为所有人都有读、写、执行权限：

```
chmod 777 -R ./*
```

还有一种特殊情况是 sdk 所在文件夹权限是 root 用户，这些组件没有访问权限，需修改文件夹权限为 user 用户访问

ls -la 查看文件夹的权限如图示：



```
user@ubuntu:~/LPT262$ ls -la
total 52
drwxr-xr-x 10 root root 4096 Feb 11 15:19 .
drwxr-xr-x 23 user user 4096 Feb 11 16:11 ..
drwxr-xr-x 16 root root 4096 Feb 8 15:08
drwxr-xr-x 18 root root 4096 Jan 23 13:54
drwxr-xr-x 2 root root 16384 Jan 11 10:37
drwxr-xr-x 17 user user 4096 Feb 8 14:50
```

修改文件夹用户指令: `sudo chown -R user:user ./file_name` (PS:该虚拟机的用户名为 user，实际以自己用户名为准)

3. 编译报错缺少关键组件

Q: 编译报错 `ModuleNotFoundError: No module named 'pycparser'`

A: 执行指令 `pip3 show pycparser` 查看安装信息。

情况 1:未正确安装 'pycparser'

复制 `pycparser-2.22-py3-none-any.whl` 文件到虚拟机

安装: `pip3 install pycparser-2.22-py3-none-any.whl`

情况 2:显示已安装

卸载指令 `pip3 uninstall pycparser`，之后重新安装

4. 编译报错编解码解析失败

Q: 编译报错 `UnicodeDecodeError: 'utf-8' codec can't decode byte...`

A: 出现此问题通常是虚拟机用户名或语言是中文引起

修改语言环境为英文后重启虚拟机。该方法无效时，可尝试下面两个方法

方法 1:

终端命令行输入: `sudo gedit ~/.bashrc`

最后一行输入 `export LC_ALL=C.UTF-8`

保存退出，重启虚拟机

方法 2:

终端命令行输入:

`sudo apt-get update`

`sudo apt-get install locales`

安装/修复完成后输入 `locale -a` 查看安装结果

重启虚拟机