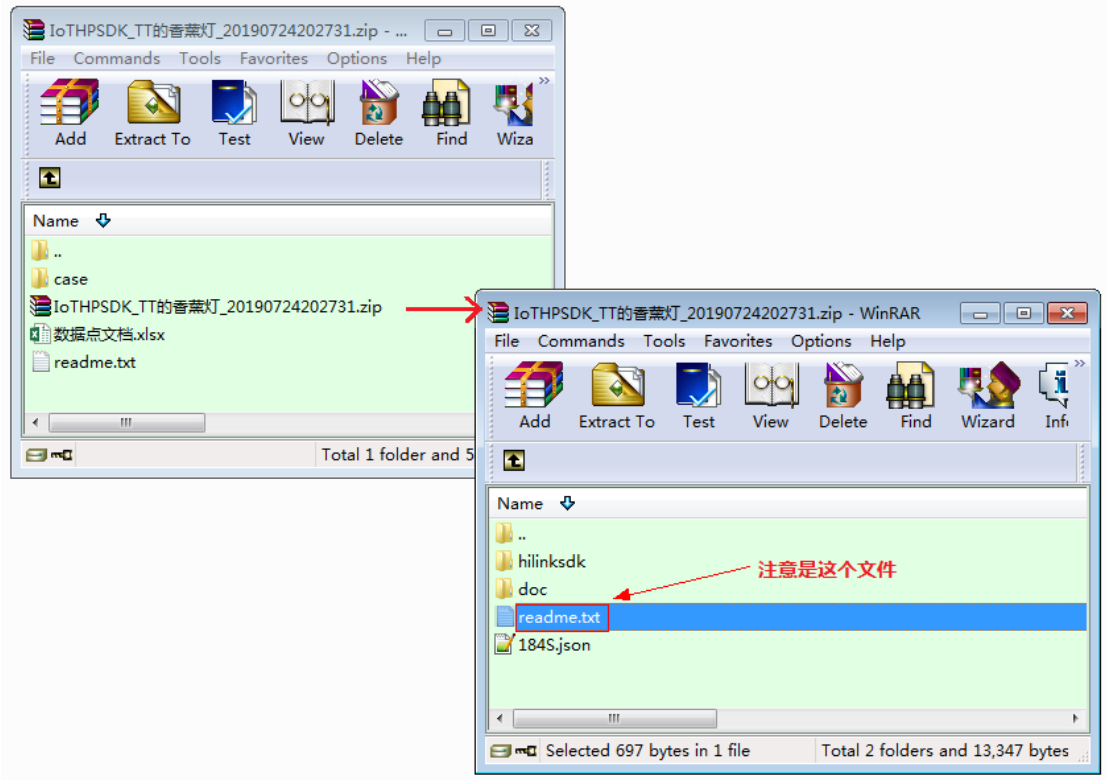
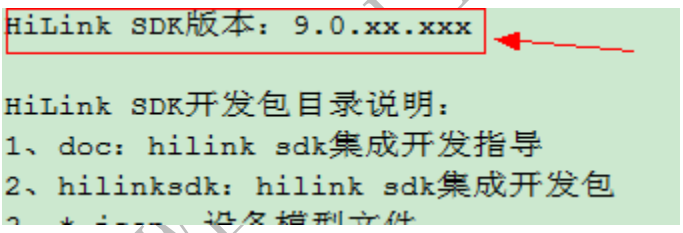
目录

一、如何查看 HiLink SDK 的版本号

开发者从华为开发者平台下载的 HiLink SDK 开发包最内层压缩包内有个 readme.txt 文件。示例如下：



readme 文件中有 HiLink SDK 的版本号。示例如下：



二、HiLink 升级相关问题

1.为什么设备升级成功 APP 一直提示升级失败？

问题原因：

设备升级重启后，需要跟云端重新建立连接，登入云端。APP 只有获取到设备重启登入云端之后，才能判断设备升级并重启成功了，进而判断此次升级成功。如果设备还在重启并登录云端的过程中，而此时 APP 侧已经超时了，就会提示升级失败。

解决方案：

设备镜像升级过程中会同步调用下面的 hilink_ota_rpt_prg 接口上报升级进度 progress，以及等待设备升级完成并重新登录云端的时间 bootTime 给 APP。

int hilink_ota_rpt_prg(int progress, int bootTime);

如果 bootTime 时间设置太小，会导致设备升级完成重启之后，还未登录到云端，APP 侧已经超时并提示失败。

因此 bootTime 时间建议为 1 分钟，bootTime 时间单位为秒。

2.升级完成调用接口上报 100%后，进度为何没有上报成功？

问题原因：

设备完成镜像升级并调用接口上报 100%进度后，如果马上重启设备，有可能会导致进度无法正常上报。

解决方案：

设备完成镜像升级并调用接口上报 100%进度后，不要马上重启，建议设备可以延迟 2-3 秒后再重启，确保 100%进度发送成功。

3.设备升级失败后是否需要增加重试机制？

问题原因：

智能家居设备升级成功率受网络环境影响较大，升级失败之后需要增加重试机制。

解决方案：

设备升级失败之后需要增加重试机制，由于网络原因导致的下载镜像失败、检测新版本失败，设备需要重试 3 次升级。对于升级失败的，设备要能够上报 SDK 定义的相应的错误码，参照 hilink_ota.h 中 hilink_ota_rpt_prg 接口的使用说明。

4.APP 提示升级超时，没有进度显示，提示升级失败？

问题原因：

APP 点击设备升级后，有 20 秒的超时时间判断，如果连续 20 秒内都没有收到设备上报升级进度的变化，APP 就会提示超时，升级失败。

解决方案：

请按如下指导排查该问题：

1、设备升级进度上报的时间间隔不能大于 20 秒，请排查升级进度上报的时间间隔是否小于 20 秒。

2、请排查设备收到升级命令，启动升级后，是否立即先上报进度 progress =1，表示设备已经开始升级。

三、怎么看 SDK 串口调试打印参考信息

下面以乐鑫 esp8266 模组为例进行说明，其他模组类似。

1.配网调试

(1) 设备进入配网模式：

mode : softAP(hw mac)

dhcp server start:(ip set, mask set, gw set)

add if1

bcn 100

(2) 扫描设备并发送 SSID 和密码到设备

scandone

state: 0 -> 2 (b0)

state: 2 -> 3 (0)

state: 3 -> 5 (10)

add 0

aid 1

pm open phy_2,type:2 0 0

cnt

(3) 设备连接路由器

connected with ap, channel 9

dhcp client start...

ip got, mask ok, gw ok

2.互联互通调试

(1) 设备收到 APP 的控制命令：

Hilink recieve a PUT cmd, on = 0.

(2) 设备收到 APP 的状态查询命令：

Hilink recieve a GET cmd, on = 0.

这里只是示例，具体收到 APP 控制命令打印的日志内容，由开发者在控制函数中实现。

四、模组使用注意事项

1.汉枫 LPB130 模组编译固件时提示错误“region DRAM overflowed with stack”？

问题原因：

汉枫 LPB130 模组编译链接时报错，信息如下：

```
./arm-none-eabi/bin/ld: region DRAM overflowed with stack
```

原因是 DRAM 空间不足。

解决方案：

汉枫 HF-LPX30-HSF-1MB_2MB_20181010.rar 之前的 SDK 版本规划的 DRAM 空间是 64KB，如果固件 DRAM 使用超过 64KB 编译时就会提示上述错误信息。

汉枫在 HF-LPX30-HSF-1MB_2MB_20181010.rar 之后的 SDK 版本规划的 DRAM 空间是 96K，增加了 DRAM，解决了在编译时 DRAM 空间不足的问题。

因此，汉枫 LPB130 模组集成 HiLink SDK 时请使用 HF-LPX30-HSF-1MB_2MB_20181010.rar 版本及之后版本的 SDK。

2.乐鑫 ESP8266 模组添加设备时，设备日志不停打印“send errno [12]”？

问题原因：

乐鑫 ESP8266 模组添加设备时，设备日志不停打印如下信息：

```
connected with ap, channel 9
dhcp client start...
ip got, mask ok, gw ok
send errno [12]

send errno [12]

send errno [12]

send errno [12]

send errno [12]
```

原因是设备和云端 TCP 通信时内存不足，TCP 发送返回的错误码 12 表示 Out of memory。

解决方案：

可以通过修改链接文件配置，节省 HiLink SDK 占用的 RAM 内存。优化方法如下：

在 ESP8266_RTOS_SDK-2.0.0/ld 目录下找到链接文件 eagle.app.v6.common.ld。
把 HiLink SDK 的静态库添加到.irom0.text 段中，参考如下：

```
106   .irom0.text : ALIGN(4)
107   {
108     _irom0_text_start = ABSOLUTE(.);
109     *libuser.a:(.rodata.* .rodata)
110     *libcirom.a:(.rodata.* .rodata)
111     *libmbedtls.a:(.rodata.* .rodata)
112     *libssl.a:(.rodata.* .rodata)
113     *libopenssl.a:(.rodata.* .rodata)
114     *libplatforms.a:(.rodata.* .rodata)
115     *libhilinkdevicesdk.a:(.rodata.* .rodata)
116     *(.irom0.literal .irom.literal .irom.text.literal .irom0.text .irom.text)
117     *(.literal.* .text.*)
118     _irom0_text_end = ABSOLUTE(.);
119   } >irom0_0_seg :irom0_0_phdr
```

3.LiteOS 模组编译固件时提示错误“region ‘BD_RAM’ overflowed by xxx bytes”？

问题原因：
LiteOS 模组编译链接时报错，信息如下：

```
./arm-none-eabi/bin/ld: region `BD_RAM' overflowed by 345 bytes
```

固件的静态 RAM 不足，超出 xxx 字节。
解决方案：
通过调节 LiteOS 动态内存的大小，匀出部分 RAM 空间作为静态 RAM 区；方法如下：
/component/soc/realtek/8711b/cmsis/device/app_start.c

```
20  __attribute__ ((aligned (4)))
21  #endif
22  #endif
23  /* All heap memory in data section, when less than 140k, system maybe abnormal*/
24  #define TOTAL_HEAP_SIZE (145 * 1024)
25  static unsigned char g_sysHeap[TOTAL_HEAP_SIZE];
26
27  #if (defined(CONFIG_POST_SIM))
28  void Simulation_Init(void);
```

修改 TOTAL_HEAP_SIZE 的大小即可，单位 byte；
TOTAL_HEAP_SIZE 的值越小，则 LiteOS 可分配的动态内存越少，静态内存就越大；
注意：此特性仅 LiteOS 版本为 B337 及以上版本支持。

五、HiLink SDK 提供的功能

1.联网功能

HiLink SDK 提供的基本功能之一就是联网功能。联网功能实现设备连接路由网络、注册及登录智能家居云、添加到智能家居 APP。联网功能的实现已封装到 HiLink SDK 静态库内部，集成方法参见《智能家居 HiLink SDK 集成开发调测指导》。

2.互联互通功能

互联互通功能是 HiLink SDK 提供的设备与智能家居 APP 之间命令交互、状态同步的功能，包括智能设备响应智能家居 APP 的服务状态控制命令和状态查询命令，以及智能设备上报服务状态给智能家居 APP。互联互通功能的集成方法参见《智能家居 HiLink SDK 集成开发调测指导》。

3.升级功能

针对部分模组，HiLink SDK 默认实现了模组和 MCU 的 HOTA 升级功能。目前支持的模组有乐鑫 ESP8266 和汉枫 LPB130 模组。开发者只需实现几个接口，即可完成集成，集成方法参见《智能家居 HiLink SDK 集成开发调测指导》。

除了支持 HOTA 升级的模组外，使用其他模组的设备，升级功能需要开发者实现。HiLink SDK 提供了适配接口和示例实现流程，帮助开发者实现和集成 OTA 功能。具体方法参见《智能家居 HiLink SDK 集成开发调测指导》。

4.时间管理功能

时间管理功能指的是智能家居 APP 通过定时或倒计时的方式进行设备控制的功能。时间管理功能已经在 HiLink SDK 内部默认实现了，不需要用户适配集成。

5.设备网络信息功能

设备网络信息功能指的在智能家居 APP 上可以查看设备连接的网络信息。网络信息包括设备连接的 WiFi 热点名称、WiFi 强度、RSSI、设备 IP 地址、BSSID。设备网络信息功能已经在 HiLink SDK 内部默认实现了，不需要用户适配集成。

6.本地控制功能

本地控制实现智能家居 APP 和设备间的直接互联互通操作，而不经过智能家居云。当设备添加到 APP 和智能家居云后，在出现网络状态不好或者外网断连的情况下，智能家居

APP 依然可以通过本地路由器实现对设备的控制，增强用户的使用体验。本地控制功能是 HiLink SDK 内部默认功能，不需要用户适配集成。

六、开关插排类多路设备共用代码的实现

1.背景说明

对于开关、插排类产品，会存在同类产品不同路数的情况(比如一路开关、二路开关)。这些不同路数产品对应的 HiLink SDK 集成代码的差异很小，因此有些开发者希望对于不同路数的产品是否可以共用一套代码。

不同路数产品的差异在于产品信息和服务信息等，因此若要共用一套代码，需要使用最大路数的产品 HiLink SDK 代码为基准修改，使用宏隔离的方式将其他路数的产品信息集成进去。

2.不同路数产品差异点

(1) 版本号信息，包括固件版本号、软件版本号、硬件版本号。
(2) 设备基本信息，包括产品 ID、产品型号。
(3) 设备服务信息，包括服务个数、服务列表。
(4) 设备 AC 值
(5) 设备 BI 值

3.隔离方法

以多路开关为例。定义宏 SWITCH_WAYS 表示开关的路数，取值 1 表示一路开关，2 表示二路开关，3 表示三路开关，那么隔离的方法如下：

```
#if (SWITCH_WAYS == 1)
    // 一路开关的相关定义
#elif (SWITCH_WAYS == 2)
    // 二路开关的相关定义
#else // (SWITCH_WAYS == 3)
    // 三路开关的相关定义
#endif
```

4.操作步骤(以一至三路开关为例)

(1) 获取一路开关、二路开关和三路开关的 HiLink SDK 集成开发包。以最大路数产品(此处就是三路开关)为基准，在其 HiLink SDK 集成开发包代码上进行修改。

(2) 定义区分不同路数开关的宏，可以定义在 hilink_device.h 文件中。例如:

```
#define SWITCH_WAYS 2    // 表示二路开关产品
```

(3) 将一路、二路开关的版本号定义和设备信息定义拷贝到 hilink_device.h 文件对应位置，并用宏隔开。示例如下：

```
#if (SWITCH_WAYS == 1)

// 适配产品根据实际情况自行修改
#define FIRMWARE_VER "1.0.0"
#define SOFTWARE_VER "1.0.0"
#define HARDWARE_VER "1.0.0"

//设备基本信息，请勿修改
#define PRODUCT_ID "0001"
#define DEVICE_MODEL "SWITCH-01"

#elif (SWITCH_WAYS == 2)

// 适配产品根据实际情况自行修改
#define FIRMWARE_VER "1.0.0"
#define SOFTWARE_VER "1.0.0"
#define HARDWARE_VER "1.0.0"

//设备基本信息，请勿修改
#define PRODUCT_ID "0002"
#define DEVICE_MODEL "SWITCH-02"

#else // (SWITCH_WAYS == 3)

// 适配产品根据实际情况自行修改
#define FIRMWARE_VER "1.0.0"
#define SOFTWARE_VER "1.0.0"
#define HARDWARE_VER "1.0.0"

//设备基本信息，请勿修改
#define PRODUCT_ID "0003"
#define DEVICE_MODEL "SWITCH-03"

#endif
```

一路开关信息

二路开关信息

三路开关信息

(4) 将一路、二路开关的服务信息、AC 值和 BI 值拷贝到 hilink_device_sdk.c 文件对应位置，并用宏隔开。示例如下：

```
#if (SWITCH_WAYS == 1)
// 设备服务信息
int gSvcNum = 1;
svc_info_t gSvcInfo[] =
{
    {"binarySwitch", "switch1"}
};

// AC值
unsigned char A_C[48] = {}; // 内容省略

// BI值
char* bi_rsacipher = ""; // 内容省略
```

一路开关信息

```
#elif (SWITCH_WAYS == 2)
// 设备服务信息
int gSvcNum = 2;
svc_info_t gSvcInfo[] =
{
    {"binarySwitch", "switch1"},
    {"binarySwitch", "switch2"}
};

// AC值
unsigned char A_C[48] = {}; // 内容省略

// BI值
char* bi_rsacipher = ""; // 内容省略
```

二路开关信息

```
#else // (SWITCH_WAYS == 3)
// 设备服务信息
int gSvcNum = 3;
svc_info_t gSvcInfo[] =
{
    {"binarySwitch", "switch1"},
    {"binarySwitch", "switch2"},
    {"binarySwitch", "switch3"}
};

// AC值
unsigned char A_C[48] = {}; // 内容省略

// BI值
char* bi_rsacipher = ""; // 内容省略

#endif
```

三路开关信息

(5) 实际使用时，对于不同路数的产品，只需在固件编译时，将宏的值定义成对应路数即可。
例如：-DSWITCH_WAYS=3

(6) 开发者在实现设备功能时，如果存在需要对不同路数产品进行隔离的情况时，也可使用前面定义的宏进行隔离。