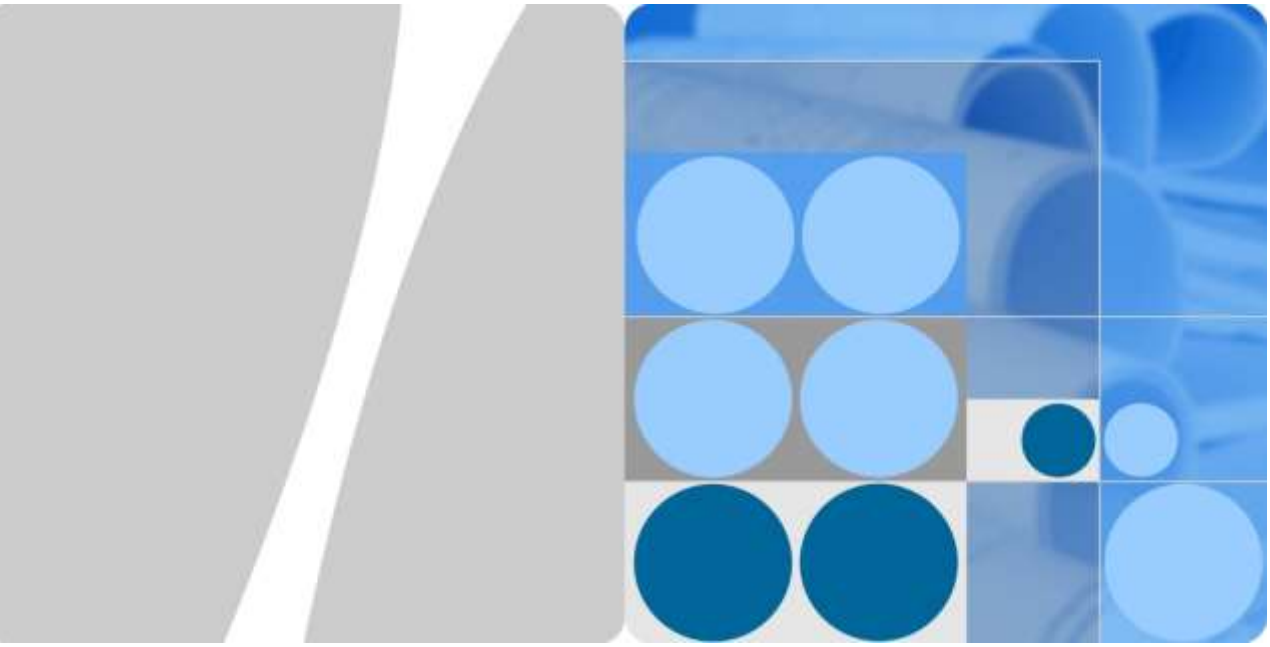


智能家居 HiLink SDK Linux 系统

集成开发调测指导

文档版本 2

发布日期 2020-08-31

华为技术有限公司

HUAWEI

商标声明

 和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

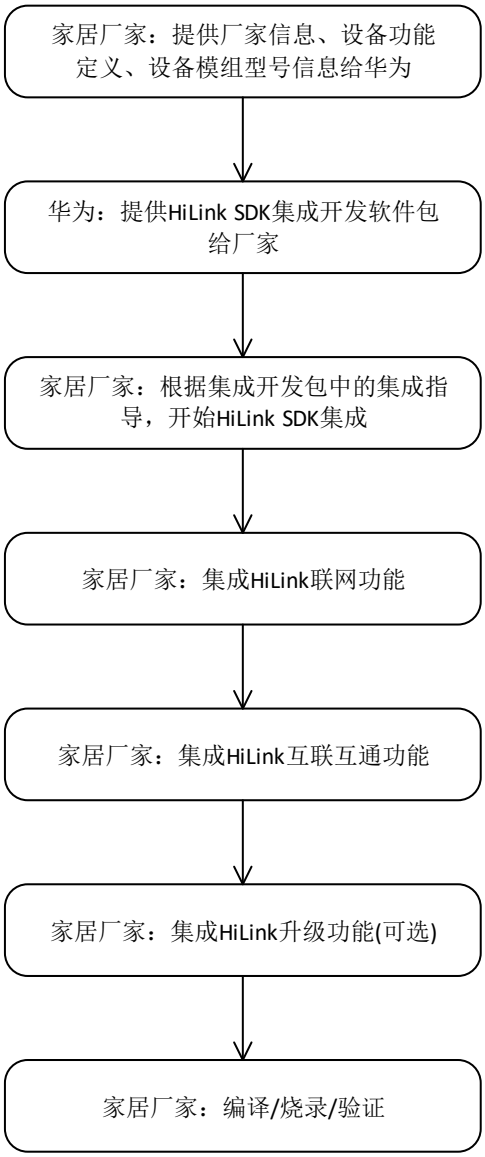
目录

1 概述

HiLink SDK是运行在设备的程序模块，用于实现设备的联网，以及设备与智能家居云和智慧生活App的互联互通。HiLink SDK提供的功能包括设备联网、互联互通、设备升级、本地控制、错误诊断等。集成HiLink SDK可以帮助传统设备快速实现智能化。

本文档用于指导开发者在使用Linux OS的智能设备中集成和调测HiLink SDK，实现和验证HiLink设备的远程控制、设备状态上报和OTA升级等功能。

HiLink SDK的开发集成整体流程如下：



2 开发包结构

HiLink根据开发者提供的CPU类型和编译链生成设备的HiLink SDK开发包，其结构及文件说明如下表：

目录	文件名	说明
doc	智能家居 HiLink SDK Linux 系统集成开发调测指导	智能家居设备集成 HiLink SDK 开发和调测指导书
	HiLink 智能家居解决方案 SDK 基本功能测试用例	HiLink 智能家居设备自验用例集
	智能家居 HiLink SDK 集成 FAQ	HiLink SDK 集成中常见问题 FAQ
lib	libhilinkdevicesdk.a	HiLink SDK 静态库文件 Debug 版本用于设备调测时集成 Release 版本用于商用设备发布时集成
	libhilinkota.a	HOTA 升级特性静态库文件 Debug 版本用于设备调测时集成 Release 版本用于商用设备发布时集成 （仅支持使用华为 HOTA 云的产品）
include	hilink.h	HiLink SDK 提供的功能函数定义
	hilink_netconfig_mode_mgt.h	产品配网模式选择相关函数
	hilink_log_manage.h	HiLink SDK日志打印级别管理相关定义
adapter | sdk_adapter	hilink_sdk_adapter.c	HiLink SDK 业务相关待实现函数
	hilink_ota.c	设备 OTA 升级功能待实现函数
	include\ hilink_sdk_adapter.h	HiLink SDK 业务相关待实现函数声明
	include\hilink_ota.h	设备 OTA 升级功能相关定义
adapter | profile_adapter	hilink_profile_adapter.c	产品功能适配源文件，包含设备模型相关待实现函数
	include\ hilink_profile_adapter.h	产品功能适配源文件，包含设备模型相关定义
adapter | network_adapter	hilink_network_adapter.c	设备配网相关待实现函数
	include\ hilink_network_adapter.h	设备配网相关定义
demo	hilink_demo.c	demo 样例
	Makefile	Makefile 样例文件，供开发时参考
plugin	*	HiLink SDK 支持插件列表，具体使用方法请参考 3.12 插件管理功能集成章节

3 HiLink SDK 集成详细步骤

3.1 集成准备

集成HiLink SDK之前，开发者需要先做一些准备工作，以满足HiLink SDK的运行要求。

3.1.1 创建必要文件夹

/usrdata/hilink目录为HiLink SDK运行期间的默认配置文件存储位置。

开发者需要在系统根目录中创建/usrdata/hilink文件夹，并保证HiLink SDK程序对该路径有读写权限。

如果开发者需要自己指定HiLink SDK的配置路径，可使用开发包中include\hilink.h的HILINK_SetConfigInfoPath接口设置该路径。

3.2 添加 HiLink SDK 开发包到主程序工程中

解压HiLink SDK开发包到本地智能设备主程序源代码工程目录。通过修改Makefile文件，将HiLink SDK开发包中的源代码文件（*.c）、头文件（*.h）和静态库文件（*.a）添加到智能设备程序编译中。

3.3 拷贝域名配置文件

开发者需要将开发包中config目录下的hilink_bak.cfg和hilink.cfg域名配置文件拷贝到3.1.1章节创建的HiLink SDK配置路径，并保证HiLink SDK程序对该文件有读写权限。

3.4 联网功能集成

联网功能是指HiLink智慧生活App扫描添加设备、绑定设备到用户账号的功能。设备可以通过集成HiLink SDK的联网功能，实现连接路由网络、注册及登录智能家居云、添加到智慧生活App。

3.4.1 网络相关接口适配

开发者需要先适配adapter\network_adapter路径下的hilink_network_adapter.c文件，完成Linux系统下开关AP、获取网络信息以及HiLink SDK线程重启等接口，供HiLink SDK调用。

3.4.2 设置 HiLink SDK 配置保存路径（可选）

参考include\hilink.h中HILINK_SetConfigInfoPath接口设置HiLink SDK的配置保存路径；

使用HILINK_GetConfigInfoPath接口查询当前HiLink SDK的属性。

3.4.3 设置 HiLink SDK 属性（可选）

参考include\hilink.h中结构体HILINK_SdkAttr的描述，使用HILINK_SetSdkAttrs接口设置HiLink SDK的属性；使用HILINK_GetSdkAttr接口查询当前HiLink SDK的属性。各个属性值的说明如下。

1) monitorTaskStackSize：HiLink SDK监控任务栈大小，开发者根据具体情况调整，默认为1024字节，在异常监控任务中会调用重启前处理接口hilink_process_before_restart、软重启接口rebootSoftware和硬重启接口rebootHardware，若在以上接口的实现中使用了较大的栈空间，调整此接口；

2) deviceMainTaskStackSize：普通设备形态，HiLink SDK运行主任务栈大小，开发者根据具体情况调整，如果当前设备形态是普通设备，可以使用此接口调整栈大小；

3) bridgeMainTaskStackSize：网桥设备形态，HiLink SDK运行主任务栈大小，开发者根据具体情况调整，如果当前设备形态是网桥设备，可以使用此接口调整栈大小；

4) otaCheckTaskStackSize：HiLink OTA检查升级版本任务栈大小，开发者根据具体情况调整，如果使用HiLink提供的OTA升级，可以使用此接口调整栈大小；

5) otaUpdateTaskStackSize：HiLink OTA升级任务栈大小，开发者根据具体情况调整，如果使用HiLink提供的OTA升级，可以使用此接口调整栈大小；

6) rebootSoftware：HiLink SDK异常、OTA升级、设备删除等场景软重启接口，不影响设备硬件状态，如果用户注册此接口，首先使用；

7) rebootHardware：HiLink SDK异常、OTA升级、设备删除等场景硬重启接口，影响硬件状态，如果开发者未注册软重启接口，使用此接口重启。

示例：如果要调整HiLink SDK监控任务栈大小，由默认1024字节改为2048字节，该如何处理？

(1) 在调整该栈大小前先调用HILINK_SdkAttr *HILINK_GetSdkAttr(void)函数接口获取当前各任务栈的大小；

(2) 调用int HILINK_SetSdkAttr(HILINK_SdkAttr sdkAttr)函数接口来设置要修改的任务栈的大小。

注：调用(1) (2)中两个函数都必须在hilink_main()函数之前，否则不会生效。

```
void main(void)
{
    HILINK_SdkAttr *sdkAttr = NULL;

    /*  获取HiLink SDK当前属性值  */
    sdkAttr = HILINK_GetSdkAttr();
    if (sdkAttr == NULL) {
        printf("sdk attr is null\r\n");
```

```
        return;
    }

    /* 设置新的属性值 */
    sdkAttr->monitorTaskStackSize = 2048;
    HILINK_SetSdkAttr(*sdkAttr);

    /* 启动HiLink任务 */
    hilink_main();

    return;
}
```

3.4.4 设置配网方式（可选）

参考include\hilink_netconfig_mode_mgt.h中enum HILINK_NetConfigMode的描述，使用HILINK_SetNetConfigMode接口通知HiLink SDK。

3.4.5 启动 HiLink SDK 主程序

在设备程序中直接调用hilink_main()函数，示例代码如下：

```
include "hilink.h"
...
void main(void)
{
    ...
    hilink_main();
    ...
}
```

3.5 互联互通功能集成

互联互通功能是HiLink SDK提供的设备与智慧生活App之间的命令交互、状态同步的功能。功能包括：智能设备响应华为智慧生活App的服务状态控制命令；智能设备上报服务状态给华为智慧生活App；智能设备处理华为智慧生活App的服务状态查询命令。

集成互联互通功能，请参考如下步骤：

3.5.1 修改设备信息

1. 版本信息

请根据实际情况修改adapter\profile_adapter\include\hilink_profile_adapter.h中设备的固件版本、软件版本、硬件版本。

```
#define FIRMWARE_VER "20000"
#define SOFTWARE_VER "1.0.3"
#define HARDWARE_VER "20000"
```

2. 厂商及设备名称修改(可选)

请根据实际情况修改adapter\profile_adapter\include\hilink_profile_adapter.h中的厂商和设备的名称。

```
#define DEVICE_TYPE_NAME "Switch"     // 设备类型名称
#define MANUAFACTURER_NAME "Huawei"     // 设备厂商名称
```

HiLink SDK会使用到厂商名称和设备类型名称来组装SoftAP的SSID，开发者需要确保这两个字符串的长度之和不能超过17字符。建议名称字符串使用简短单词或单词缩写来表示。

3. 设备SN录入(可选)

HiLink SDK默认使用MAC地址作为SN号，如果开发者需要根据实际情况录入SN，可以实现adapter\profile_adapter\hilink_profile_adapter.c中的如下接口，将SN数据传递给HiLink SDK。

```
void HilinkGetDeviceSn(unsigned int len, char *sn);
```

参数说明：

(1) unsigned int len: SN限制的最大长度，39字节；

(2) char *sn: 录入的SN信息。

4. 设备子型号适配(可选)

如果设备定义有子型号，则HiLink SDK在组装SoftAp的SSID阶段、向云端注册阶段、登录录后设备信息同步阶段，三个阶段均会使用设备子型号，开发者需要适配adapter\profile_adapter\hilink_profile_adapter.c 中的如下接口，将设备的子型号ID传递给HiLink SDK。

```
int HILINK_GetSubProdId(char *subProdId, int len);
```

参数说明:

(1) char *subProdId: 保存子型号的缓冲区;

(2) int len: 缓冲区的长度;

如果设备定义有子型号，则子型号长度固定为2字节，取值为十六进制的字符串表示，范围00 ~FF，其中有字母则字母大写，并以'\0'结束，函数返回0。

如果未定义有设备子型号，函数返回-1。

示例如下：

```
int HILINK_GetSubProdId(char *subProdId, int len)
{
    strcpy(subProdId, "0A");
    return 0;
}
```

3.5.2 向 HiLink SDK 上报设备 PIN 码

每台设备都有一个独自享有的8位数的PIN码，量产时可以存储到FLASH并将其贴到产

品外包装。开发者需要通过如下适配接口，将产品的PIN码传递给HiLink SDK。

adapter\profile_adapter\hilink_profile_adapter.c文件中的int HiLinkGetPinCode(void)函数，返回值是8位数字的整型值，返回-1时HiLink SDK使用默认内置PIN码进行配网。

```
int HiLinkGetPinCode(void)

{

    /* 测试时，这个数字可以随便改，只要是8位数字即可 */

    return 12345678;

}
```

3.5.3 获取设备在线状态（可选）

开发者可以通过接口获取设备当前状态，当前支持状态列表如下：

```
/* 设备与云端连接断开(版本向前兼容) */
#define HILINK_M2M_CLOUD_OFFLINE  0
/* 设备连接云端成功，处于正常工作态(版本向前兼容) */
#define HILINK_M2M_CLOUD_ONLINE  1
/* 设备与云端连接长时间断开(版本向前兼容) */
#define HILINK_M2M_LONG_OFFLINE  2
/* 设备与云端连接长时间断开后进行重启(版本向前兼容) */
#define HILINK_M2M_LONG_OFFLINE_REBOOT  3
/* HiLink线程未启动 */
#define HILINK_UNINITIALIZED  4
/* 设备处于配网模式 */
#define HILINK_LINK_UNDER_AUTO_CONFIG  5
/* 设备处于10分钟超时状态 */
#define HILINK_LINK_CONFIG_TIMEOUT  6
/* 设备正在连接路由器 */
#define HILINK_LINK_CONNECTTING_WIFI  7
/* 设备已经连上路由器 */
#define HILINK_LINK_CONNECTED_WIFI  8
/* 设备正在连接云端 */
#define HILINK_M2M_CONNECTTING_CLOUD  9
/* 设备与路由器的连接断开 */
#define HILINK_M2M_CLOUD_DISCONNECT  10
/* 设备被注册 */
#define HILINK_DEVICE_REGISTERED  11
/* 设备被解绑 */
#define HILINK_DEVICE_UNREGISTER  12
```

1. 查询接口

调用include\hilink.h中声明的int hilink_get_devstatus(void)函数，返回值对应上面不同的状态。

2. 状态变化回调函数

实现adapter\sdk_adapter\hilink_sdk_adapter.c中的虚函数hilink_notify_devstatus(int

status)，添加设备在线状态变化时的响应处理：

```
void hilink_notify_devstatus(int status)
{
    switch(status) {
        case HILINK_M2M_CLOUD_OFFLINE:
            /* 设备与云端连接断开，请在此处添加实现 */
            break;
        case HILINK_M2M_CLOUD_ONLINE:
            /* 设备连接云端成功，请在此处添加实现 */
            break;
        …
        default:
            break;

    }
}
```

3.5.4 实现恢复出厂设置（可选）

恢复出厂设置，设备会清除Flash和华为智能家居云记录的设备绑定信息，并重新进入配网状态。

如果智能设备有恢复出厂设置按钮或接口，可在按钮/接口触发时调用include\hilink.h下的hilink_restore_factory_settings()函数。

3.5.5 实现设备服务状态控制功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_adapter.c中实现hilink_put_char_state函数。

3.5.6 实现设备服务状态查询功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_adapter.c中实现hilink_get_char_state函数。

3.5.7 实现设备服务状态上报功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_adapter.c中添加实现某些服务的状态上报接口。在设备状态主动发生变化，智慧生活App控制下发后设备状态没有立刻改变、过段时间完成更改后主动上报设备的状态。请开发者调用include\hilink.h中的hilink_report_char_state接口实现。

3.5.8 实现设备重启预处理功能

开发者需要根据实际情况实现adapter\profile_adapter\hilink_profile_adapter.c中的接口hilink_process_before_restart(int flag)。在异常、OTA升级、设备删除等情况下，HiLink SDK会请求重启模组，调用该接口通知开发者进行数据备份等必要操作，以保证模组重启后的运行状态与重启前一致。返回0表示处理成功，系统可以重启，使用硬重启接口重启；返回1表示处理成功，系统可以重启，使用软重启；返回负值表示处理失败，系统不能重启，HiLink SDK继续等待。

这里的flag表示了不同的重启类型，当flag为0时，表示HiLink SDK线程看门狗触发即将重启，此时函数返回值既可以为0或1(允许重启)，也可以为负值(不允许重启)，需要开发者根据具体业务状态决定；当flag为1时，表示App侧删除设备即将重启，此时函数返回值必须为0或1(允许重启)，否则将导致删除设备功能异常。

注意：请将该接口实现为非阻塞函数。

3.5.9 设备离线场景在 App 上删除设备 SDK 释放清除注册信息功能适配（可选）

设备离线时，如果在App上删除了该设备，设备下次上线后云端会给设备下发Errcode=5(登录错误)或Errcode=6(设备已被删除)错误码。此时设备接收到这两个错误码后，会根据接口HILINK_EnableProcessDelErrCode (enable)的设置结果来判断是否需要清除注册信息进入配网。

enable为0表示SDK不处理云端下发的Errcode=5或Errcode=6错误码，此时SDK不会清除设备端注册信息，需要用户手动硬件恢复出厂设置，设备才能重新进入配网状态。

enable为非0表示SDK处理云端下发的Errcode=5或Errcode=6错误码，此时SDK会清除设备端注册信息，并重新进入配网状态。

如果不设置，默认enable为0。

对于配网跟注册一体的设备无需设置此接口。

对于配网跟注册分离的设备，如果具备单独的本地清除注册的功能，无需设置此接口。

对于配网跟注册分离的设备，如果不具备单独的本地清除注册的功能，需要设置enable为非0。

3.5.10 设备读写 license 功能适配（可选）

如果设备支持License认证机制，需要完成如下两个接口的适配。

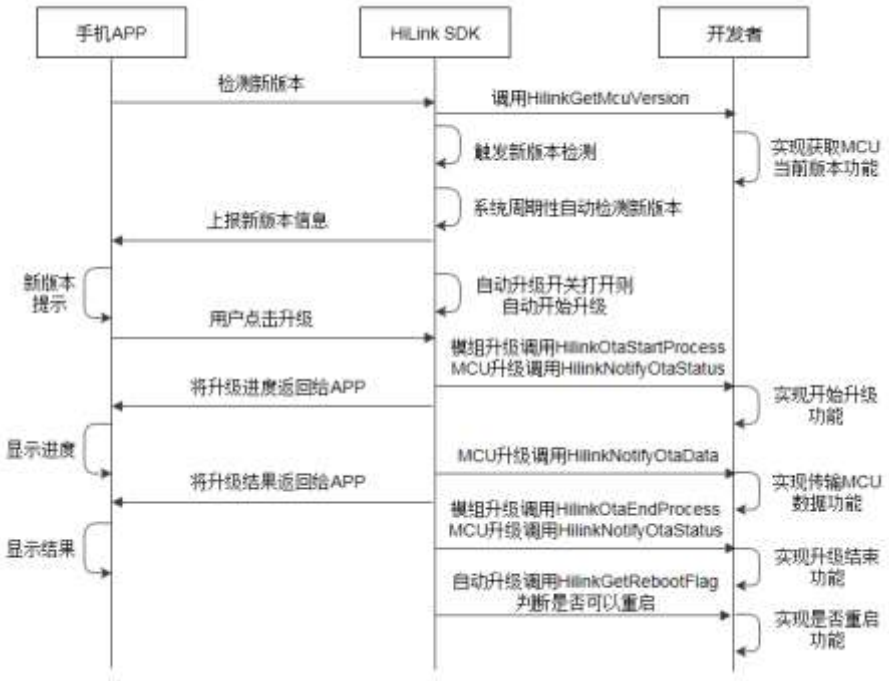
int HILINK_WriteLicense(const unsigned char *license, unsigned int len)：此接口实现License信息的存储，写入flash位置或者文件由厂家决定。厂家需要保证备份机制，防止突然断电导致license信息丢失，如果license信息丢失，将无法继续绑定设备，设备将不能再使用。执行成功返回0，执行失败返回-1。

int HILINK_ReadLicense(unsigned char *license, unsigned int len)：此接口实现License信息的读取，读取flash位置或者文件由厂家决定。执行成功返回0，执行失败返回-1。

3.6 HOTA 功能集成（可选）(使用华为 HOTA 云)

HiLink SDK实现了基于华为HOTA云的设备和MCU的OTA升级功能，需要开发者决定是否将固件存在华为HOTA云服务器，使用该方式则由HiLink SDK完成升级包的检测、下载功能，否则由开发者实现。

HiLink设备的升级通过华为智慧生活App触发或者用户打开自动升级功能自动触发，由HiLink SDK实现设备和MCU（如果有）升级业务，开发者只需要实现几个接口，整体流程如下图所示：



3.6.1 实现升级接口函数

实现adapter\sdk_adapter\hilink_ota.c中升级接口函数：

1. 读取新固件大小函数（必须）

实现获取固件大小的函数HILINK_GetMaxUpdateFileSize(void)，HiLink SDK下载文件前需要保证所下载的文件不能超过分区的大小，所以需要获取下载文件需要占用的最大空间。

2. 获取新固件存放路径（必须）

实现升级文件的存放路径接口HILINK_GetUpdateFilePath(char *filePath, unsigned int len)，HiLink SDK会将下载的升级文件保存在开发者指定的路径。

3. 设备开始升级函数（必须）

实现设备开始升级函数HilinkOtaStartProcess (int type)，开发者可在此函数中实现升级开始时需要添加的功能。

在手动升级场景下，HiLink SDK在接收到用户发出的升级指令后，将直接调用此接口；在自动升级场景下，当HiLink SDK在调用HilinkGetRebootFlag接口返回值是

MODULE_CAN_REBOOT时，HiLink SDK将调用此接口。厂商可在此接口中完成和升级流程相关的处理。自动升级流程在凌晨进行，因此厂商在实现升级流程相关功能时，确保在升级的下载安装固件和重启设备时避免对用户产生影响，比如发出声音、光亮等。如果处理正常就返回RETURN_OK，处理异常请返回RETURN_ERROR。

4. 设备结束升级函数（必须）

实现设备结束升级函数HilinkOtaEndProcess (int status)，开发者可在此函数中实现在升级结束时需要添加的功能。

HiLink SDK在将固件写入到OTA升级区后，且完整性校验通过后，将调用厂商适配的此接口；厂商可在此接口中完成和升级流程相关的处理。自动升级流程在凌晨进行，因此厂商在实现升级流程相关功能时，确保在升级的下载安装固件和重启设备时避免对用户产生影响，比如发出声音、光亮等；升级类型是否为自动升级可参考接口HilinkOtaStartProcess的参数type的描述。如果处理正常就返回RETURN_OK，处理异常请返回RETURN_ERROR。

5. 安装新固件函数（必须）

实现安装新固件函数HILINK_StartSoftwareIntall(void)，HiLink SDK负责新版本的检测和下载，由开发者来实现软件的安装。

HiLink SDK调用此函数后便会退出升级，交由开发者的安装程序来完成软件的安装。返回HILINK_OK启动成功，HILINK_ERROR启动失败。

6. 判断设备是否能重启函数（必须）

实现判断设备是否能重启函数HilinkGetRebootFlag(void)，开发者可在此函数中实现重启前保存数据之类的功能。

在用户同意设备可以自动升级的情况下，HiLink SDK调用此接口获取设备当前业务状态下，模组是否可以立即升级并重启的标志。只有当设备处于业务空闲状态时，接口才可以返回MODULE_CAN_REBOOT。当设备处于业务非空闲状态时，接口返回MODULE_CANNOT_REBOOT。

7. 获取MCU当前版本信息函数（如果有MCU）

实现获取MCU当前版本号HilinkGetMcuVersion (char *version, unsigned int inLen, unsigned int *outLen)，开发者实现获取MCU的当前版本号。

如果获取不到MCU的版本，则不对MCU进行升级。建议开发者在MCU正常启动后，或升级启动后，就将MCU的版本号传递给设备，确保设备可以获取到MCU的版本。如果没有MCU的返回RETURN_ERROR_NO_MCU，获取成功则返回RETURN_OK，获取失败则返回RETURN_ERROR。

8. 通知MCU升级状态函数（如果有MCU）

实现通知MCU升级状态函数HilinkNotifyOtaStatus (int flag, unsigned int len, unsigned int type)，开发者可在此函数中实现MCU升级状态改变时需要添加的功能。没有MCU可不用实现此函数。

HiLink SDK调用开发者适配的此接口通知MCU固件传输的状态。当flag是STOP_SEND_DATA时，此接口需返回MCU侧固件升级的结果；当flag是其它值时，需返回接口接收到此消息后的处理结果。自动升级流程在凌晨进行，因此厂商在实现升级流程相关功能时，确保在升级的下载安装固件和重启设备时避免对用户产生影响，比如发出声音、光亮等。

9. MCU数据传输函数（如果有MCU）

实现MCU数据传输函数HilinkNotifyOtaData(unsigned char *data, unsigned int len, unsigned int offset)，开发者可在此函数中实现传输MCU固件数据给MCU的功能。没有MCU可不用实现此函数。

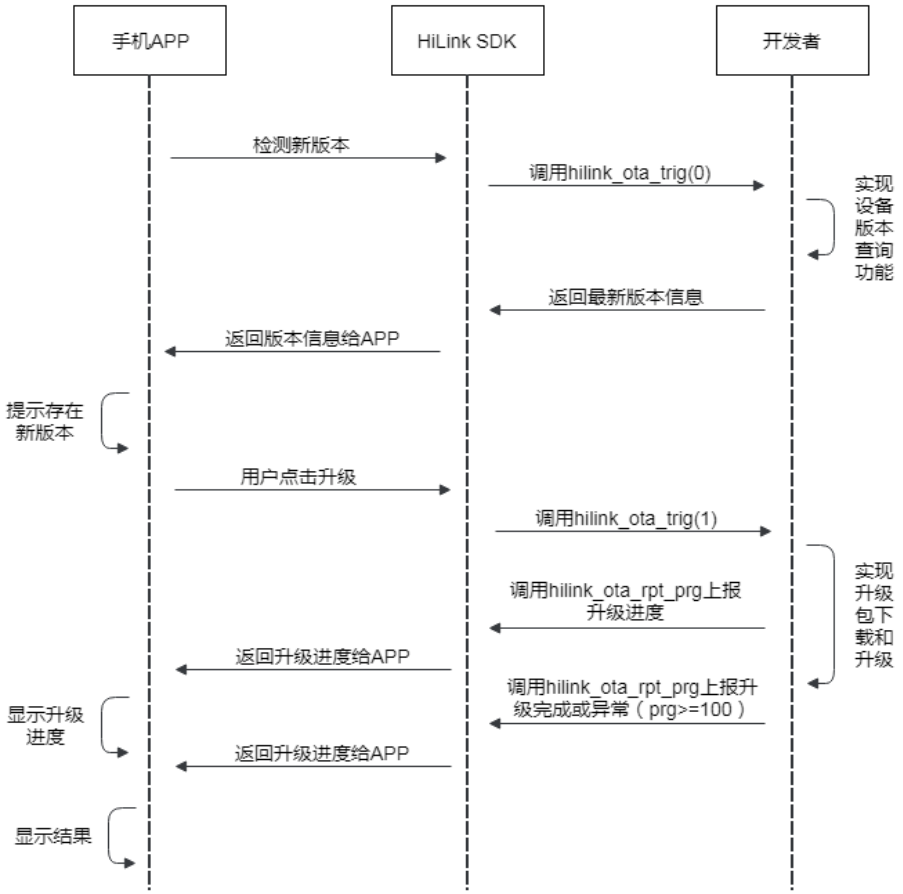
HiLink SDK调用开发者适配的此接口通知开发者发送MCU固件文件数据。HiLink SDK通知发送MCU固件文件时，将MCU固件文件拆分成若干个数据包通知给设备。开发者适配的此接口需要返回MCU接收这部分数据的处理结果。当此接口返回

RETURN_OK时，HiLink SDK继续正常处理后续流程；当此接口返回RETURN_ERROR时，HiLink SDK将终止此次的MCU固件升级。如果处理正常就返回RETURN_OK，处理异常请返回RETURN_ERROR。

3.7 OTA 功能集成（可选）(不使用华为 HOTA 云)

针对不使用华为HOTA云服务器的产品，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。

HiLink设备的升级通过华为智慧生活App触发，由开发者实现具体升级业务，整体流程如下图所示：



根据业务需要，修改实现adapter\sdk_adapter\hilink_ota.c中如下升级配置：

3.7.1 修改升级配置

1. 版本号字符串长度

修改宏定义VERSION_LEN，调整设备版本字符串长度，范围为[0, 64)。代码示例如下：

```
#define VERSION_LEN (16)
```

2. 版本描述字符串长度

修改宏定义VERSION_INTRO_LEN，调整设备版本描述字符串长度，范围为[0, 512)。代码示例如下：

```
#define VERSION_INTRO_LEN (64)
```

3. 设备重启延迟时间

修改宏定义OTA_REBOOT_TIME，调整设备升级后重启延迟时间，单位为秒。示例代码如下：

```
#define OTA_REBOOT_TIME (60)
```

4. 升级任务优先级和栈空间大小

修改hilink_ota_trig函数中创建升级任务hilink_ota_task时指定的优先级和栈空间大小，示例如下：

```
ret = hilink_task_create(&deviceOtaTask, "hilink_ota_task",
                         4,     /* task优先级参考值，可根据实际情况调整 */
                         1024, /* task栈大小参考值，可根据实际情况调整 */
                         device_ota_task,
                         NULL);
```

3.7.2 实现升级业务函数

1. 获取设备最新版本信息函数

实现获取设备最新版本号和版本描述信息函数get_latest_version_info，示例代码如下：

```
int get_latest_version_info(char* latest_version, char* latest_ver_intro) {
    /* 实现逻辑:
        1、连接升级服务器;
        2、获取最新固件版本信息，输出到latest_version和latest_ver_intro。
    */
    strncpy(latest_version, HOTA_dev_ver, strlen(HOTA_dev_ver));
    strncpy(latest_ver_intro, HOTA_ver_intro, strlen(HOTA_ver_intro));
    return 0;
}
```

2. 获取设备当前版本号

实现获取设备当前版本号函数get_current_version，将当前设备版本号赋值全局变量g_currentVersion，示例代码如下：

```
int get_current_version(void) {
```

```
    /* 获取当前设备版本号到全局变量g_currentVersion */
    strncpy(g_currentVersion, dev_ver, strlen(dev_ver));
    return 0;
}
```

3. 实现定时器函数

实现定时器函数device_ota_timer，在升级任务超时后，由定时器函数终止任务。示例代码如下：

```
void device_ota_timer(int action){
    if (action == 1) {
        /* 在此处实现启动定时器 */
        start_ota_timeout_timer();
    } else if (action == 0) {
        /* 在此处实现关闭定时器 */
        stop_ota_timeout_timer();
    } else {
        /* 非法参数 */
    }
}
```

4. 实现升级任务函数

实现升级任务函数device_ota_task，实现设备软件版本下载和本地升级、重启。示例代码如下：

```
int device_ota_task(void){
    /* 设备升级流程由开发者实现，实现逻辑大体分为以下几个步骤供参考:
        1. 连接升级服务器;
        2. 下载升级包;
        3. 升级包校验;
        4. 升级包存入flash;
        5. 设备重启，运行新版本固件.
    升级过程中同步调用hilink_ota_rpt_prg接口上报实时升级进度.
```

```
    */
    hilink_task_delete(&deviceOtaTask, NULL);
    device_ota_timer(0); /* 关闭定时器 */
    return 0;
}
```

5. 上报升级进度

在升级任务函数device_ota_task中，调用hilink_ota_rpt_prg函数，上报设备升级进度。示例代码如下：

```
hilink_ota_rpt_prg(55, OTA_REBOOT_TIME);
```

3.8 DHCP Option 60 功能实现

DHCP Option 60用于智能设备向DHCP服务器上报自身厂商以及配置信息，设备开发者需要自行实现DHCP Option 60上报功能。HiLink要求在DHCP的discover报文中上报DHCP Option60信息，上报格式为：“厂商标识”:“终端类型”:“终端型号”。

厂商标识”、“终端类型”、“终端型号”三个属性可以通过adapter\profile_adapter\include\hilink_profile_adapter.h文件中的“MANUAFACTURER_NAME”、“DEVICE_TYPE_NAME”、“DEVICE_MODEL”三个宏定义获取。

3.9 网桥下挂子设备

网桥功能需要适配的功能如下：

3.9.1 上报子设备设备状态

子设备状态参考adapter\profile_adapter\include\hilink_profile_bridge.h中enum DevOnlineStatus的描述，使用HilinkSyncBrgDevStatus接口通知HiLink SDK。

3.9.2 实现子设备信息查询功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_bridge.c中实现HilinkGetBrgDevInfo函数。

3.9.3 实现子设备服务信息查询功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_bridge.c中实现HilinkGetBrgSvcInfo函数。

3.9.4 同步上报子设备服务状态

开发者需要根据产品功能定义，需要同步子设备服务状态时，调用adapter\profile_adapter\include\hilink_profile_bridge.h中声明的接口函数HilinkReportBrgDevCharState上报子设备的服务状态。

3.9.5 异步上报子设备服务状态

开发者需要根据产品功能定义，需要同步子设备服务状态时，调用adapter\profile_adapter\include\hilink_profile_bridge.h中声明的接口函数HilinkUploadBrgDevCharState上报子设备的服务状态。

3.9.6 实现子设备服务状态控制功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_bridge.c中实现HilinkPutBrgDevCharState函数。

3.9.7 实现子设备服务状态查询功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_bridge.c中实现HilinkGetBrgDevCharState函数。

3.9.8 实现子设备删除功能

开发者需要根据产品功能定义，在adapter\profile_adapter\hilink_profile_bridge.c中实现HilinkDelBrgDev函数。

3.9.9 实现子设备复位功能

开发者需要根据产品功能定义，在子设备复位时调用adapter\profile_adapter\include\hilink_profile_bridge.h中声明的接口函数HilinkResetBrgDev通知HiLink SDK。

3.10 PIN 码配网功能集成

在SOFTAP配网时，也需要用到PIN码配网功能，在3.5.2中已经适配PIN，其他涉及到需要开发者适配的接口如下：

3.10.1 设置 SoftAp 配网时的最大接入 station 数

建议开发者设置SoftAP热点的最大允许接入STA数为2，适配函数见adapter\network_adapter\hilink_network_adapter.c中的

```
void HILINK_SetStationNumLimit(void)
{
    return;
}
```

3.10.2 配网超时后踢除掉连接的设备

开发者需要适配adapter\network_adapter\hilink_network_adapter.c中的HILINK_DisconnectStation函数，根据HiLink SDK传入的IP地址，将对应的station踢除掉。

```
void HILINK_DisconnectStation(const char *ip)
{
    return;
}
```

3.11 建立语音 SDK 通道（语音生态单品使用）

语音生态产品需要使用语音SDK时，开发者在启动HiLink主程序之前调用hilinksdk/include/hilink.h中的接口函数HILINK_GetVoiceContext获取一个void*类型的变量，再通过语音SDK接口将该变量设置到语音SDK，语音SDK接口设置方法请参考语音SDK开发指导。

3.12 日志级别配置功能（可选）

HiLink SDK在include\hilink_log_manage.h中提供了日志级别的枚举类型HiLinkLogLevel，默认情况下，release版本的日志打印级别是HILINK_LOG_ERR，debug版本的日志打印级别是HILINK_LOG_DEBUG。如果开发者希望自己配置日志级别，可以参考本章节设置。

3.12.1 配置 HiLink SDK 日志打印级别

开发者可以调用include\hilink_log_manage.h中的HILINK_SetLogLevel函数，配置HiLink SDK内部日志打印最高级别。例如配置的日志级别为HILINK_LOG_ERR, 表示级别在HILINK_LOG_ERR以下(HILINK_LOG_WARN等)的日志不会打印出来。

3.12.2 获取 HiLink SDK 日志打印级别

开发者可以调用include\hilink_log_manage.h中的HiLinkLogLevel HILINK_GetLogLevel函数，查询HiLink SDK内部日志打印最高级别。例如查询到的日志级别为HILINK_LOG_ERR, 表示级别在HILINK_LOG_ERR以下(HILINK_LOG_WARN等)的日志不会打印出来。

3.13 插件管理功能集成（可选）

HiLink SDK以插件的形式集成其他组件，为插件提供安全的控制通道，开发者可根据需求和设备能力去集成。目前仅支持DV-Kit。若选择集成某插件，将该插件lib目录下的库文件与hilink库文件放在同一目录下即可。具体集成方法和适配接口可参考插件doc目录下对应文档。

4 功能验证

4.1 概述

根据集成开发包中的认证测试用例《HiLink智能家居解决方案SDK基本功能测试用例》验证智能设备基本功能。

根据产品定义的功能，验证业务功能。使用配套的华为智慧生活App（实现该设备对应的添加、注册、控制功能）与设备共同验证。

特别注意，请关闭手机应用市场中华为智慧生活App的自动升级功能。该App为测试专用App，请勿升级至其他版本，否则App无法进行正常调试工作。

4.2 App 调试环境设置

1. 运行华为智慧生活App，选择服务器，选择“厂家认证云”，点击“确定”按钮。

2. 选择音箱云服务器，任选一个即可，点击“确认”按钮。

3. 阅读相关用户协议和隐私声明，勾选加入用户体验计划（也可后面在App中打开），点击“同意”按钮。

4. 允许App获取的权限，点击“始终允许”按钮。

5. 登录华为账号（若之前手机已登录过，App会自动登录），观看使用引导。

4.3 搜索添加待测设备

1. 在智慧生活App“家居”页面，点击右上角的“+”号按钮，选择“添加设备”。

2. 华为智慧生活App开始自动扫描周围设备。

3. 选择扫描出的对应设备，点击“连接”按钮，进入配网流程。

4. 在“连接设备”页面输入手机当前连接的WLAN热点的密码，单击“下一步”，设备开始自动连网注册。

说明

1、如果App已绑定了华为HiLink路由器，密码将自动填充，不需输入。

2、如果设备实现了定制PIN码配网，输入密码后还需要手动输入PIN码。

5. 在“设备设置”页面，设置设备名称及选择设备所属房间。如果设备需要使用自动升级功能，可以打开自动升级开关。点击“完成”按钮，设备即添加成功。

6. 设备添加完成后，可在华为智慧生活App“家居”页面查看到已添加设备。点击设备图标进入设备页面，即可完成设备功能的控制等操作。

4.4 验证设备控制功能

1. 进入设备页面，操作相关控件，控制设备功能。
2. 查看设备状态是否按预期改变，以及App状态与设备侧状态是否一致。

具体页面和操作，与具体设备类型和功能相关，此处不再赘述。