



Hi3861V100 / Hi3861LV100 升级

开发指南

文档版本 05

发布日期 2020-08-06

版权所有 © 上海海思技术有限公司2020。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HISILICON、海思和其他海思商标均为海思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、服务或特性等应受海思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，海思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。

上海海思技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： <https://www.hisilicon.com/cn/>

客户服务邮箱： support@hisilicon.com



前言

概述

本文档主要介绍了升级接口的使用方法，供用户开发升级使用。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3861	V100
Hi3861L	V100




读者对象

本文档主要适用于以下工程师：



- 技术支持工程师。
- 软件开发工程师。

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 危险	表示如不可避免则将会导致死亡或严重伤害的具有高等级风险的危害。
 警告	表示如不可避免则可能导致死亡或严重伤害的具有中等级风险的危害。
 注意	表示如不可避免则可能导致轻微或中度伤害的具有低等级风险的危害。



符号	说明
 须知	用于传递设备或环境安全警示信息。如不可避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

修改记录

文档版本	发布日期	修改说明
05	2020-08-06	在“ 3 注意事项 ”中新增升级注意兼容场景的说明；新增FlashBoot和Kernel不支持同时升级的说明。
04	2020-07-29	<ul style="list-style-type: none"> 在“2 开发流程”的功能中更新表2-1；开发流程中更新步骤3。 在“3 注意事项”中更新各场景的升级流程中至少需要的内存如下的内容。
03	2020-07-21	<ul style="list-style-type: none"> 在“功能”中更新表2-1。 在“开发流程”中新增“如果是升级文件下载完成后不需要立即进行升级，用户需要升级时再进行升级的场景”步骤。 更新“3 注意事项”的内容。
02	2020-06-28	<ul style="list-style-type: none"> 在“1 概述”中新增关于升级模式的建议。 在“3 注意事项”中新增关于升级模式的建议。
01	2020-04-30	<p>第一次正式版本发布。</p> <ul style="list-style-type: none"> 在“1 概述”中更新关于双分区升级模式、压缩升级模式的描述。 在“2 开发流程”的表2-1中更新hi_upg_transmit、hi_upg_transmit_finish接口的描述。 更新“4 编程实例”的Kernel升级示例、FlashBoot升级示例。



文档版本	发布日期	修改说明
00B03	2020-04-03	<ul style="list-style-type: none">在“1 概述”中新增关于Menuconfig中的“OTA Settings”选择双分区升级模式、压缩升级模式的描述；新增关于注册回调类接口的说明。在“2 开发流程”的表2-1中更新hi_upg_get_file_index接口的描述；新增hi_upg_register_file_verify_fn接口的描述。在“3 注意事项”中新增hi_upg_register_file_verify_fn接口的注意说明。在“4 编程实例”中更新关于双分区升级模式下获取APP升级文件编号的注释说明。
00B02	2020-02-12	<ul style="list-style-type: none">在“表2-1”中新增hi_upg_init接口描述。在“3 注意事项”中新增hi_upg_init接口的注意说明。
00B01	2020-01-15	第一次临时版本发布。



目录

前言.....	i
1 概述.....	1
2 开发流程.....	2
3 注意事项.....	4
4 编程实例.....	5



1 概述

当前Flash分区请参见《Hi3861V100 / Hi3861LV100 SDK 开发指南》的“Flash分区与Flash保护”小节，通过调用升级接口更新Kernel或FlashBoot，从而实现升级。

- Menuconfig中的“OTA Settings”选择“dual-partition ota support”（双分区升级模式），升级Kernel时，如果当前运行在Kernel A，则获取Kernel B的升级文件更新至Kernel B区，升级重启后从Kernel B启动；如果当前运行在Kernel B，则获取Kernel A的升级文件更新至Kernel A区，升级重启后从Kernel A启动。
- Menuconfig中的“OTA Settings”选择“compression ota support”（压缩升级模式），升级Kernel时，将压缩后的升级文件更新至Kernel B，升级重启后，在FlashBoot下，将Kernel B中的压缩文件解压缩后，更新至Kernel A，并从Kernel A启动。
- 升级FlashBoot时，获取FlashBoot升级文件先更新至备份FlashBoot区，校验通过后，再更新至FlashBoot区，升级重启后使用新的FlashBoot。

建议优先选择压缩升级模式，其相比双分区升级，镜像可用空间更大，默认多250KB+。如果选择双分区升级，需要考虑后续功能扩展情况，为后续的升级镜像预留足够空间，升级镜像不支持压缩升级和双分区升级混用。

根据功能不同，升级接口可以分为以下几类：

- 流程类接口：用于控制升级流程。此类接口包括：
 - 升级文件传输接口
 - 升级文件传输完成接口
 - 升级结束接口
 - 升级停止接口
- 信息获取类接口：用于获取升级文件相关信息。此类接口包括：
 - 获取升级文件编号接口
 - 获取升级文件大小上限接口
 - 获取升级文件内容接口
- 注册回调类接口：用于用户控制或了解当前执行的升级流程。此类接口包括：
 - 注册用户自定义升级文件合法性校验接口

用户在任务初始化流程中调用注册接口，后续执行对应的升级流程时，调用用户注册的回调函数。



2 开发流程

使用场景

- 升级Kernel
- 升级FlashBoot

功能

升级模块提供的接口如表2-1所示。

表 2-1 升级接口描述

接口名称	描述
hi_upg_init	升级状态初始化。
hi_upg_transmit	升级文件传输。
hi_upg_transmit_finish	升级文件传输完成。与hi_upg_finish配合使用，在hi_upg_transmit_finish的调用与hi_upg_finish接口的调用之间不能出现断电或重启操作，如果有断电或重启操作，则必须重新执行升级文件传输流程下载升级文件。
hi_upg_finish	升级结束。与hi_upg_transmit_finish配合使用，在hi_upg_transmit_finish的调用与hi_upg_finish接口的调用之间不能出现断电或重启操作，如果有断电或重启操作，则必须重新执行升级文件传输流程下载升级文件。
hi_upg_transmit_finish_save_cache	文件传输结束，保存传输流程的关键参数。与hi_upg_finish_with_cache配合使用，适用于升级文件下载完成后不需要立即进行升级的场景，允许在hi_upg_transmit_finish_save_cache和hi_upg_finish_with_cache的调用之间存在掉电或重启操作。



接口名称	描述
hi_upg_finish_with_cache	升级结束，允许在调用此接口前存在断电或重启操作。与hi_upg_transmit_finish_save_cache配合使用，适用于升级文件下载完成后不需要立即进行升级的场景；允许在hi_upg_transmit_finish_save_cache和hi_upg_finish_with_cache的调用之间存在掉电或重启操作。
hi_upg_stop	升级停止。
hi_upg_get_file_index	双分区升级模式，升级Kernel时用于获取Kernel升级文件编号。 1： Kernel A升级文件； 2： Kernel B升级文件。
hi_upg_get_max_file_len	获取允许升级的升级文件长度最大值。
hi_upg_get_content	获取升级文件内容。
hi_upg_register_file_verify_fn	注册用户自定义升级文件合法性校验接口。 升级文件中预留了32byte用户自定义字段，可配合该接口使用。

开发流程

升级典型场景的开发流程：

步骤1 调用hi_upg_transmit，分包传输升级文件。如果该接口返错，则停止升级流程。

步骤2 调用hi_upg_transmit_finish，通知UPG模块传输完成。如果该接口返错，则停止升级流程。

步骤3 调用hi_upg_finish，实现升级重启。

----结束

如果是升级文件下载完成后不需要立即进行升级，用户需要升级时再进行升级的场景：

步骤1 调用hi_upg_transmit，分包传输升级文件。如果该接口返错，则停止升级流程。

步骤2 调用hi_upg_transmit_finish_save_cache，通知UPG模块传输完成。如果该接口返错，则停止升级流程。

步骤3 调用hi_upg_finish_with_cache，实现升级重启，在此步骤前，允许存在断电或重启操作。

----结束



3 注意事项

- hi_upg_transmit传输第1包的长度 ≥ 96 byte。
- hi_upg_transmit分包传输升级文件且按从前到后的顺序传输。
- hi_upg_transmit相同包不能重复传输。
- hi_upg_transmit_finish必须在传输文件后调用，不能重复调用。
- FlashBoot升级过程中不能断电，FlashBoot升级不支持停止升级。
- hi_upg_get_content需要在调用hi_upg_transmit传输文件之后，且调用hi_upg_transmit_finish之前调用。
- 升级接口不支持在中断中调用；不支持多个APP同时调用。
- hi_upg_init必须在hi_nv_init之后调用。SDK包中默认APP已调用此接口，具体请参见app_main函数。
- hi_upg_register_file_verify_fn必须在启动升级流程前调用。
- 建议优先选择压缩升级模式，其相比双分区升级，镜像可用空间更大：默认多250KB+。如果选择双分区升级，需要考虑后续功能扩展情况，为后续的升级镜像预留足够空间，升级镜像不支持压缩升级和双分区升级混用。
- hi_upg_transmit_finish需要和hi_upg_finish配合使用；
hi_upg_transmit_finish_save_cache需要和hi_upg_finish_with_cache配合使用，以适配不同的使用场景，不能相互混用。
- 如果存在升级不兼容场景，APP必须使用升级文件头中的自定义字段提前增加校验处理，避免升级后单板损坏。
- FlashBoot和Kernel不支持同时升级，即不支持在一次升级流程中完成FlashBoot和Kernel的升级。
- 各场景的升级流程中至少需要的内存如下：
 - Flashboot加密场景下，升级加密Kernel： 4KBytes
 - Flashboot加密场景下，升级非加密Kernel： 4KBytes
 - Flashboot加密场景下，升级加密Flashboot： 等于Flashboot的大小
 - Flashboot非加密场景下，升级加密Kernel： 4KBytes
 - Flashboot非加密场景下，升级非加密Kernel： 4KBytes
 - Flashboot非加密场景下，升级非加密Flashboot： 4KBytes



4 编程实例

Kernel 升级示例

```
hi_u32 file_size = 0x2000; /* 升级文件大小（实际大小由APP获取） */
hi_u32 max_len;
hi_u8 file_index;

/* 1.获取APP升级文件大小上限. */
hi_u32 ret = hi_upg_get_max_file_len(HI_UPG_FILE_KERNEL, &max_len);
if ((ret != HI_ERR_SUCCESS) || (file_size > max_len)) {
    return HI_ERR_UPG_FILE_LEN;
}

/* 2.双分区升级模式，获取APP升级文件编号以确定需要传输的升级文件。压缩升级模式，跳过此步骤即可. */
ret = hi_upg_get_file_index(&file_index);
if (ret != HI_ERR_SUCCESS) {
    return ret;
}

/* 3.用户自行实现：通过网口或串口等方式加载对应升级文件。
并调用接口hi_upg_transmit将升级文件传输给UPG模块. */

/* 4.传输完成hi_upg_transmit_finish.如果该接口返错，则停止升级流程. */
ret = hi_upg_transmit_finish();
if (ret != HI_ERR_SUCCESS) {
    /* 停止升级流程 */
    hi_upg_stop();
    .....
}

/* 5.升级结束hi_upg_finish. */
hi_upg_finish();
```

FlashBoot 升级示例

```
hi_u32 file_size = 0x2000; /* 升级文件实际大小（实际大小由APP获取） */
hi_u32 max_len;

/* 1.获取APP升级文件大小上限. */
hi_u32 ret = hi_upg_get_max_file_len(HI_UPG_FILE_BOOT, &max_len);
if ((ret != HI_ERR_SUCCESS) || (file_size > max_len)) {
    return HI_ERR_UPG_FILE_LEN;
}

/* 2.用户自行实现：通过网口或串口等方式加载对应升级文件。
并调用接口hi_upg_transmit将升级文件传输给UPG模块. */
```



```
/* 3.传输完成hi_upg_transmit_finish.如果该接口返错，则停止升级流程. */  
ret = hi_upg_transmit_finish();  
if (ret != HI_ERR_SUCCESS) {  
    /* 停止升级流程 */  
    hi_upg_stop();  
    .....  
}  
/* 4.升级结束hi_upg_finish. */  
hi_upg_finish();
```