

# RKupgrade

## 二次开发库用户手册

发布版本:1.0

日期:2017.11

# 前言

## 概述

RKUpgrade.dll 二次开发库, 是基于 VS2008 开发, 支持 Ansi 和 Unicode 编码. 提供读写读序列号|蓝牙地址|网卡地址等接口, 方便客户定制工具.

## 支持产品

芯片名称	
RK3399 RK3368	
RK3288 RK3228 RK3229	
RK3188 RK3126 RK3128	
RK3066	

## 读者对象

工具开发工程师

## 修订记录

日期	版本	作者	修改说明
2017.11.06	1.0	LY	初稿

# 目录

1.二次开发步骤.....	2
1.1 导入库和头文件.....	2
1.2 初始化 RKUpgrade 库 .....	2
1.3 扫描设备.....	2
1.4 操作设备(以写序列号为例).....	2
1.5 反初始化 RKUpgrade.dll 库.....	2
2.操作接口 .....	3
2.1 读写自定义数据.....	3
2.2 读写序列号 .....	3
2.3 读写网卡地址.....	3
2.4 读写 WifiMac 地址.....	3
2.5 读写蓝牙地址 .....	3
2.6 清空 Sector3 数据.....	4
2.7 读写 Vendor 数据 .....	4
2.8 读写 Provision 数据 .....	4
2.9 读写 KeyHash 数据 .....	4
2.10 读写 Efuse 数据.....	4
2.11 重启 rockusb 设备 .....	4
3.常见问题 .....	5
3.1 日志文件提示" ERROR:CheckUsbDevice->Usb type mismatch".....	5
3.2 日志文件提示" ERROR:WriteSN-->SN Size is Wrong" .....	5
3.3 日志文件提示" ERROR:WriteSN-->CheckIDBData failed" .....	5
3.4 日志文件提示" ERROR:TestDevice-->RKU_TestDeviceReady failed,Total is zero" .....	5
3.5 日志文件提示" ERROR:PrepareIDB-->No Found 1st Flash CS" .....	5

# 1.二次开发步骤

## 1.1 导入库和头文件

采用 Vs2008 开发环境, 请按以下步骤:

步骤 1: 包含头文件(`#include "RKUpgradeDll.h"`)

步骤 2: 导入库文件(`#pragma comment(lib, "RKUpgrade.lib")`)

采用其他 windows 开发平台, 请按以下步骤:

步骤 1: 参考 RKUpgradeDll.h 文件, 声明使用到的数据类型和函数

步骤 2: 调用系统的 LoadLibrary 函数, 加载 RKUpgrade.dll

步骤 3: 调用系统的 GetProcAddress 函数, 引入使用到的函数指针

## 1.2 初始化 RKUpgrade 库

步骤 1: 初始化 INIT\_DEV\_INFO 变量为全零, bScan4FsUsb 成员和 uiRockusbTimeout 成员根据实际情况设置

步骤 2: 初始化 InitLogInfo 变量, 设置是否要记录日志和日志保存位置

步骤 3: 初始化 InitCallbackInfo 变量 为全零

步骤 4: 调用 RK\_Initialize 初始化函数

注: 在程序初始化时调用

## 1.3 扫描设备

步骤 1: 调用 RK\_ScanDevice 函数, 扫描设备

步骤 2: 判断返回值, 0 没有发现设备, 1 发现 1 台设备, >1 发现多台设备默认只操作最前面的那台

步骤 3: 判断设备的 emUsbType 值, loader 设备可以直接进行操作; Msc 设备需要先调用 RK\_SwitchToRockusb 函数切换到 rockusb; adb 设备, 需要先调用外部工具 adb.exe 执行 adb reboot loader

## 1.4 操作设备(以写序列号为例)

步骤 1: 调用 RK\_WriteSN 函数

## 1.5 反初始化 RKUpgrade.dll 库

步骤 1: 所有调用 RK\_Uninitialize 函数

注: 在程序退出时调用

以上部分请参考 TS\_DllSample 代码

## 2.操作接口

### 2.1 读写自定义数据

说明:自定义数据保存在 IDBLOCK 的扇区 3 中,有 512 个字节空间

函数:RK\_WriteCustomData 和 RK\_ReadCustomData

参数:

pCustomData:分配 512 字节 buffer

nCustomDataOffset:自定义数据在 512 空间中的偏移

nCustomDataLen:自定义数据的长度,字节单位

注:读取成功后,返回的是整个 sector3 数据,要通过 nCustomDataOffset 偏移到自定义数据.

写入的数据是从 pCustomData + nCustomDataOffset 开始的 nCustomDataLen 数据

### 2.2 读写序列号

说明:序列号在 sector3 中 2-61 位置,0-1 是序列号长度

函数:RK\_WriteSN 和 RK\_ReadSN

参数:

pSN:序列号,字符串数据

nSNLen:序列号长度,字节单位

### 2.3 读写网卡地址

说明:网卡地址在 sector3 中 506-511 位置,每 4 位代表一个字符,一共表示 12 个字符网卡地址,

函数:RK\_WriteMAC 和 RK\_ReadMAC

参数:

pMac:6 个字节转换后的地址

nMacLen:长度为 6

### 2.4 读写 WifiMac 地址

说明:WifiMac 地址在 sector3 中 445-450 位置,每 4 位代表一个字符,一共表示 12 个字符网卡地址,

函数:RK\_WriteWifi 和 RK\_ReadWifi

参数:

pWifi:6 个字节转换后的地址

nWifiLen:长度为 6

### 2.5 读写蓝牙地址

说明:蓝牙地址在 sector3 中 499-504 位置,每 4 位代表一个字符,一共表示 12 个字符网卡地址,

函数:RK\_WriteBT 和 RK\_ReadBT

参数:

pBT:6 个字节转换后的地址

nBTLen:长度为 6

## 2.6 清空 Sector3 数据

说明:sector3 中全部 512 字节清零

函数:RK\_ClearAllInfo

## 2.7 读写 Vendor 数据

说明:有两个 Vendor 区, 分别是 vendor0 和 vendor1, 每个区 504 个字节, 这个区域的性质是升级后数据不会丢失, 设备端可读可写

函数:RK\_WriteVendorInfo 和 RK\_ReadVendorInfo

参数:

pVendorBuffer:504 为单位的 buffer  
sectorOffset:指定 vendor 号, 只有 0 或者 1  
sectorCount: 指定读写访问的 vendor 数

## 2.8 读写 Provision 数据

说明:Provision 区, 大概 1-1.5M 大小的空间, 按 ID 来访问每个读写项, 每个项数据不能超过 62K. 目前只有新的芯片方案有这个接口, 请与系统工程师确认后使用

函数: RK\_WriteProvisioningData 和 RK\_ReadProvisioningData

参数:

pDataBuffer:数据项的访问 buffer  
nBufferSize:数据项 buffer 大小, 字节单位  
nID:数据项 ID

## 2.9 读写 KeyHash 数据

说明:芯片内部有一块 efuse 存储空间, 里面有块区域保存的是公钥的 hash. 这部分空间只能写一次. 写入公钥 hash 后, 芯片激活安全机制.

函数:RK\_WriteKeyHashToEfuse 和 RK\_ReadKeyHashFromEfuse

参数:

pKeyHash:32 字节内存空间  
usKeyHashSize:读取到的 keyhash 长度

注:调用 RK\_WriteKeyHashToEfuse 前, 要先调用 RK\_SetFirmware 设置签名后的 update.img 固件

## 2.10 读写 Efuse 数据

说明:芯片内部有一块 efuse 存储空间, 去掉被占用的空间外还有一些空间是开放给客户使用. 这部分空间只能写一次. 具体每个芯片开放的空间大小都不同, 请与系统工程师确认后使用.

函数: RK\_WriteDataToEfuse 和 RK\_ReadDataFromEfuse

参数:

pBuffer:内存空间, 每个 bit 占用一个字节, 最多读写 512 比特  
usPos:读写的起始比特  
usWriteSize:写入的比特数  
usReadSize:读取的比特数

## 2.11 重启 rockusb 设备

说明:重启 rockusb 设备

函数: RK\_ResetRockusb

参数:

Subcode:0 为正常重启,3 为重启进入 maskrom

## 3.常见问题

### 3.1 日志文件提示" ERROR:CheckUsbDevice->Usb type mismatch "

原因:上面除 efuse 相关的操作外,都需要在 loader 状态下进行.

注:maskrom 和 loader 都属于 rockusb,maskrom 下的操作有限.当通过 RK\_ScanDevice 扫描到 Rockusb 设备后,可以通过调用 RK\_GetDeviceInfo 函数中 pUsbtypeArray 参数来判断,值为 1 是 maskrom,值为 2 是 loader

### 3.2 日志文件提示" ERROR:WriteSN-->SN Size is Wrong "

原因:SN 超过 60 个字节

### 3.3 日志文件提示" ERROR:WriteSN-->CheckIDBData failed "

原因:IDBLock 数据被破坏,校验失败,需要重新升级固件后才能再写

### 3.4 日志文件提示" ERROR:TestDevice-->RKU\_TestDeviceReady failed,Total is zero "

原因:设备安全机制被 Enable,无法读写 sector3.需要将签名后的固件发给我司系统工程师,生成授权证书,再在所有读写操作前调用 RK\_OpenChannel 函数

### 3.5 日志文件提示" ERROR:PrepareIDB-->No Found 1st Flash CS "

原因:loader 上报没有找到 flash,请跟系统工程师确认 flash 型号是否在支持列表中,硬件检查 flash 有没有存在虚焊

### 3.6 RK\_ScanDevice 找不到设备

1. 原因:
2. 先打开设备管理器,确认是不是有 rockusb 设备
3. 存在 rockusb 设备,那么检查库初始化时是不是有指定 bScan4FsUsb 为 TRUE
4. 存在未知设备,查看此设备硬件 ID,我们的 rockusb 设备 vid 是 0x2207,pid 是 0x3xxx,0x2xxx,0x1xxx
5. 如果属于上面范围,使用 DriverAssistant 工具安装驱动
6. 未知设备(获取描述符失败),请更新产品最新 loader