Cygnus SDK OTA概要设计

© 2022 Mooresilicon All rights reserved.

本文档版权归TCL摩星半导体所有，受相关法律法规的保护。未经书面许可不得复制传播。

修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 作者 | 说明 |
| 0.1 | 2022-04-14 | 薛鹏宇 | 初稿 |
|  |  |  |  |

目录

[1 引言 4](#_Toc101174508)

[1.1背景 4](#_Toc101174509)

[1.2 约束 4](#_Toc101174510)

[1.3术语 4](#_Toc101174511)

[1.4参考资料 4](#_Toc101174512)

[2总体设计 5](#_Toc101174513)

[2.1 整体框架 5](#_Toc101174514)

[2.2 OTA 组件 5](#_Toc101174515)

[2.3 flash分区 6](#_Toc101174516)

[2.4 安卓手机OTA调试程序 7](#_Toc101174517)

[2.5 Bootloader 7](#_Toc101174518)

[2.6 BT image升级方案 7](#_Toc101174519)

[2.7 软件版本号管理 8](#_Toc101174520)

[3运行设计 10](#_Toc101174521)

[3.1 OTA启动和升级流程 10](#_Toc101174522)

[3.2 系统开机流程（着重于bootloader对OTA的支持） 11](#_Toc101174523)

[4.接口设计 12](#_Toc101174524)

[4.1 OTA over ble 12](#_Toc101174525)

[4.2 OTA 流程管理模块 12](#_Toc101174526)

[4.2 OTA bin可用性管理模块 12](#_Toc101174527)

[5.主要数据结构和逻辑设计 13](#_Toc101174528)

[5.1分区信息表 （partition info） 13](#_Toc101174529)

[5.2 OTA分区信息 （OTA info） 13](#_Toc101174530)

[6. 异常处理设计 15](#_Toc101174531)

[7. 系统扩展 16](#_Toc101174532)

# 1 引言

## 1.1背景

产品销售之后，为了维护便利，需要支持OTA升级。Cygnus SDK OTA升级框架可以让设备在固件正常运行时，通过蓝牙接收数据 进行自我更新。本文以SDK的首产品蓝牙遥控器为例，描述OTA升级框架以及OTA升级涉及到的模块。框架的设计满足后续扩展的要求，可以顺利实现其他产品以及其他途径(WIFI)的升级。

## 1.2 约束

1. OTA升级框架设计基于 ms1008 ble SOC（512KB flash）

1. 首产品蓝牙遥控器的OTA需要和当前TCL遥控器产品基本上兼容，最小化TV端修改
2. Cygnus SDK平台的产品大部分是面向低功耗应用，OTA升级需要考虑电量消耗
3. OTA以组件的方式提供给客户。在客户的产品里面，OTA是可选的
4. OTA组件需要能扩展支持多种OTA空中无线方式，包括蓝牙，wifi等

## 1.3术语

* **OTA:** Over The Air，通过空中无线方式实现设备固件升级。空中无线方式包括BT，WIFI，蜂窝等。
* **GATT：**Generic Attribute Profile, 是一个在蓝牙连接之上的发送和接收很短的数据段的通用规范。
* **OTA over ble**： OTA的固件数据传输连接是建立在 GATT 协议之上，通过GATT service传输数据。
* **GAP：**Generic Access Profile，它在用来控制设备连接和广播。
* **VT表:**函数表 VT(virtual Table)。在实现蓝牙协议栈ROM固化的时候，把 LL 与 Host相互调用的函数都找出来，转化为一个函数指针表。 在正常情况下，这个函数指针表指向原来的ROM函数。当ROM代码需要更新的时候(patch)，这个函数指针表指向FLASH的patch。
* **文件签名**：通过密钥对bin文件的hash进行加密。Bin文件和加密后的hash传给接收端。接收端用公钥解开hash，并和自己计算的bin文件hash对比。如果一样的话，说明这个bin文件是从可信发送方发送的。

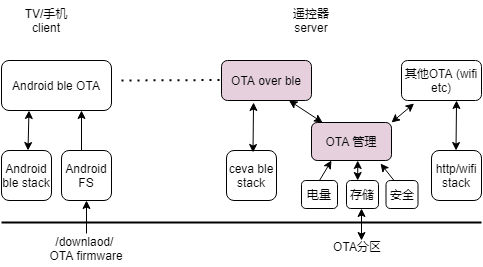
## 1.4参考资料

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 资料名称 | 版本 | 作者 |
| 1. | bt项目-蓝牙遥控器OTA应用.doc | 1.0 | 刘晋 |
| 2. | bt项目-存储模块设计.doc | 1.0 | 江澈 |
| 3. | ROM固化试验与方案v3.pdf | 3.0 | 李晓辉 |
| 4. | 智能硬件所-TCL蓝牙遥控器技术规范V1.5-20210526A | 1.5 | 余力丛、林大煜 |

# 2总体设计

## 2.1 整体框架

以cygnus首产品蓝牙遥控器为例，OTA升级的整体系统框架图如下。浅红色部分是OTA组件。



整体框架说明：

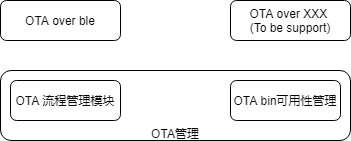
1. 待升级的 OTA image放在 TV端 android download目录下（调试和测试中，我们也会用手机Android代替TV）。
2. OTA升级由TV端发起，发起之前检查遥控器端的电量情况。
3. OTA管理需要 电源管理（电量查询），安全管理（OTA image签名认证），以及存储（OTA分区读写）三个模块提供支撑，对上支持 OTA over ble以及将来 OTA over wifi。
4. OTA bin升级成功需要3个判断：bin完整性判断(Size & CRC) 🡪bin可信判断（签名）🡪bin 可用判断（第一次开机的自检和稳定性检查）
5. OTA over ble是基于ble gatt 的一种service， ble OTA service。

|  |  |  |
| --- | --- | --- |
| Name | UUID | [Property](javascript:;) |
| OTA Server | f000ffc0-0451-4000-b000-000000000000 | - |
| Config Characteristic | f000ffc1-0451-4000-b000-000000000000 | Notify/Write |
| Data Characteristic | f000ffc2-0451-4000-b000-000000000000 | Notify/Write |

1. OTA over wifi 暂时不考虑。

## 2.2 OTA 组件

OTA组件包含以下4个子模块：



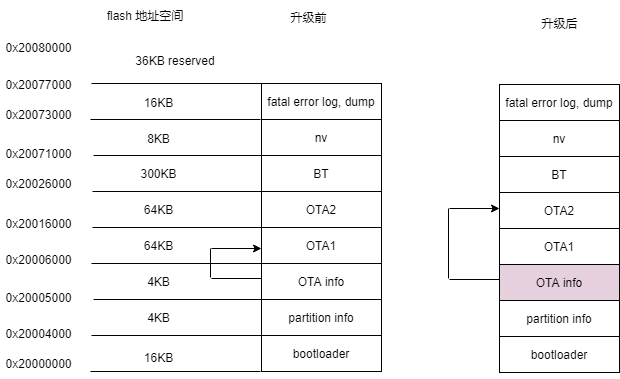
OTA组件子模块功能介绍：

|  |  |  |
| --- | --- | --- |
| 序号 | 模块名 | 主要功能 |
| 1 | OTA over ble | 通过ble OTA service的 characteristic完成firmware版本号，firmware size等信息的交互以及 OTA data的收发。收到的OTA数据交给OTA流程管理模块 处理。在ble OTA service中，TV端是client端，是命令、请求的发起方，遥控器端是Server，是命令、请求的接收方。 client在和server通信时可以通过service的特性的读写，notify、indicate来做数据的交互 |
| 2 | OTA流程管理模块 | 包括获取当前的boot 分区，OTA升级过程管理（start，stop，OTA image完整性，签名检查等），设置当前的boot分区。 |
| 3 | OTA bin可用性管理 | 1. 通过文件大小和CRC判断bin文件完整性 2. 通过签名做可信判断。判断可行之后，更新OTA info区域信息。 3. 升级之后做稳定性自检，判断升级之后首次开机是否有异常。通过bin文件版本号以及升级之后的正确性，判断哪个bin文件可用，并在OTA info区域标记。 |

2.3—2.7节 不属于 OTA组件，但是为了支持OTA，需要同步或者提前进行设计开发。

## 2.3 flash分区

为了支持OTA，重新规划flash分区如下。当前MS1008 flash地址空间 0x20000000-0x20080000 共521K。



|  |  |  |
| --- | --- | --- |
| 分区 | 功能 | 说明 |
| Bootloader | 开机读取分区信息表，判断正确的app分区 | 大小16K，OTA升级不更新。用户配置分区表的时候，bootloader最大不超过32K。 |
| Partition table | 分区配置信息表。通过配置工具可以调整，缺省见上表 | 分区配置信息表在编译阶段编译到用户bin。 |
| OTA info | OTA分区信息。两个OTA分区的可用情况，版本号。出厂的时候OTA1分区可用。第一次升级之后，根据升级情况变化 | Bootloader根据OTA info执行对应的bin |
| OTA1， OTA2 | 用户bin（包含遥控器app和所需要的组件。不含蓝牙协议栈，不含单元测试）。出厂的时候OTA1有效（该信息记录在OTA info）。后续升级的数据保存在OTA2，重复进行。 | 目前约50KB。完善所有组件和遥控器app之后，不超过64KB。 |
| BT | BT image,包含ceva bt host和 controller | 大小约300KB, OTA升级不更新。通过VT表 修复 严重问题。 |
| NV | Nv，包含RF校准数据 | ceva bt目前使用4K的 NV空间，预留4KB |
| Log | 记录系统fatal error，以及crash发生时的PC指针 | fatal error在release版本转成flash循环保存 |

## 2.4 安卓手机OTA调试程序

为了支持OTA的调试和稳定性测试，需要在手机端做个OTA的client端

Android代码参考 ： 8\_sw\reference\TCL\_electronics\_doc\OTA

改进点： 需要在手机端支持压力测试。自动升级OTA firmware版本，不断触发升级过程，看遥控器端是否正常响应。

## 2.5 Bootloader

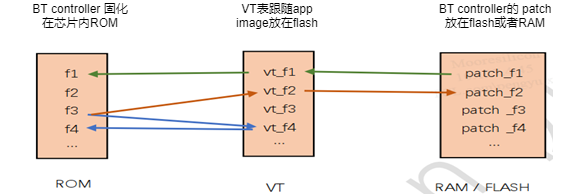
为了支持OTA， 需要设计和实现bootloader。对于OTA功能来讲，bootloader需要支持的开机流程

参考 3.2 支持OTA之后系统开机流程

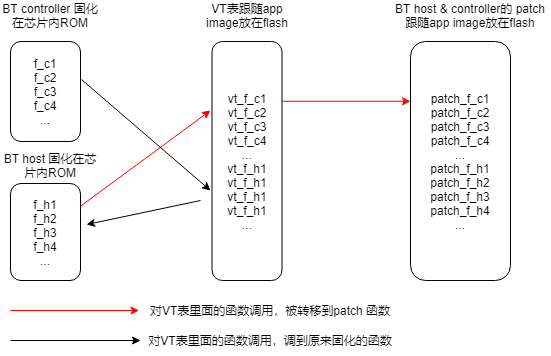
## 2.6 BT image升级方案

如前面2.3 flash分区 所述，因为BT image比较大，不考虑BT image的升级。上市之后如果BT host或者 controller有重大问题需要升级，可以通过VT 表的方式进行 (参考 资料3)。我们可以简单地把BT分区看作固化的分区（和固化ROM一样，不能修改，只能通过VT表转调函数接口）

下图为当前为了把BT controller部分固化在芯片内ROM的方案。



下图为修正之后的方案：



说明：

1. 整体的固化方案不需要修改。VT表扩展成为同时支持 BT controller和BT Host即可。
2. 当前的编译脚本需要修改，把BT Host部分独立出来，剩余的部分编译成app image
3. 如果支持蓝牙controller ROM固化，那么编译生成 bt\_controller.bin，bt\_host.bin，app.bin。如果不需要ROM固化，那么编译生成 bt\_.bin， app.bin

## 2.7 软件版本号管理

在OTA升级之前，我们需要对设备端的bin⽂文件版本与TV本地bin⽂文件的版本进⾏比较；在比较之前，需要先获取设备端bin⽂文件版本、TV本地bin⽂文件的版本。为了支持OTA 功能并和之前TCL的产品兼容，需要确定 软件版本号命名规则，设备名，Android如何读取bin文件信息 以及 设备端如何读取bin文件信息。注意这个是平台首产品 蓝牙遥控器的规则。

1. 软件版本号命名规则 （参考资料4）

分为两类，cygnus首产品蓝牙遥控器按照第一类进行。

第一类：

部分升级--名称包含\_app\_v\_ -只升级APP部分的文件名：

TCL\_RC\*\*\*\_app \_v\_\*\*\*\*\_\*\*\*\*.bin

第一段\*\*\*\_ :表示型号

第二段\*\*\*\*\_: 表示ROM版本号 ,范围从（1~9999）

第三段\*\*\*\*: 表示APP版本号，范围从（1~9999）

例如802D3型号的遥控器,只升级APP，ROM版本号为0003，APP版本号为0011:

TCL\_RC802D3\_app \_v\_0003\_0011.bin

第二类 ：

全部升级-名称包含\_ app\_stack \_v\_-升级APP和协议栈两个部分的文件名：

TCL\_RC\*\*\*\_ app\_stack \_v\_\*\*\*\*\_\*\*\*\*.bin

第一段\*\*\*\_ :表示型号

第二段\*\*\*\*\_: 表示ROM版本号 ,范围从（1~9999）

第三段\*\*\*\*: 表示APP版本号，范围从（1~9999）

例如802D3型号的遥控器，全部升级，ROM版本号为0004，APP版本号为0011:

TCL\_RC802D3\_ app\_stack \_v\_0004\_0011.bin

升级规则：

部分升级：需要保证ROM版本号一致，APP版本号不同。

全部升级：仅需保证ROM版本号不同。

版本号迭代：

ROM版本号 以0001 开始，APP版本号 以 0001开始，当ROM版本号+1后，APP版本号需要从0001重新开始累加。

1. 设备名 （RC902? TBD）
2. Android如何读取bin文件信息

开始OTA升级之前，TV端需要从OTA bin获取当前这个bin支持的设备号，版本号，bin文件大小，签名等各种信息。这些信息在和设备端交互的时候需要用到。

OTA bin文件格式参考（文件是否需要加密？ TBD）：

|  |  |  |
| --- | --- | --- |
| 字节顺序 | 用途 | 说明 |
| 0x1-0x4 | OTA bin 文件标志 |  |
| 0x5-0x100 | OTA bin 签名信息 | 用于设备端做签名认证。OTA数据传输结束后，设备端需要确认这个是TCL遥控器的bin文件。 |
| 0x101-0x10F | 文件大小，checksum | OTA开始的时候，发给设备端。设备端根据这个值判断文件大小和完整性。 |
| 0x110-0x11B | OTA bin 设备名 (12字节) | OTA开始的时候，发给设备端。设备端根据这个值判断是否推送错。 |
| 0x11C-0x11F | OTA bin 版本号 （4字节） | OTA开始的时候，发给设备端。设备端根据这个值判断是否需要升级。 |
| 0x120- | OTA bin 数据 | 待升级的OTA bin 数据 |

1. 设备端如何读取bin文件信息

开始OTA升级之前，设备端需要获取本地的版本号和设备名。这些信息存放在OTA分区的最后面16个字节。

以OTA1分区为例：

0x20015ff0-0x20015ffd 设备名 12字节

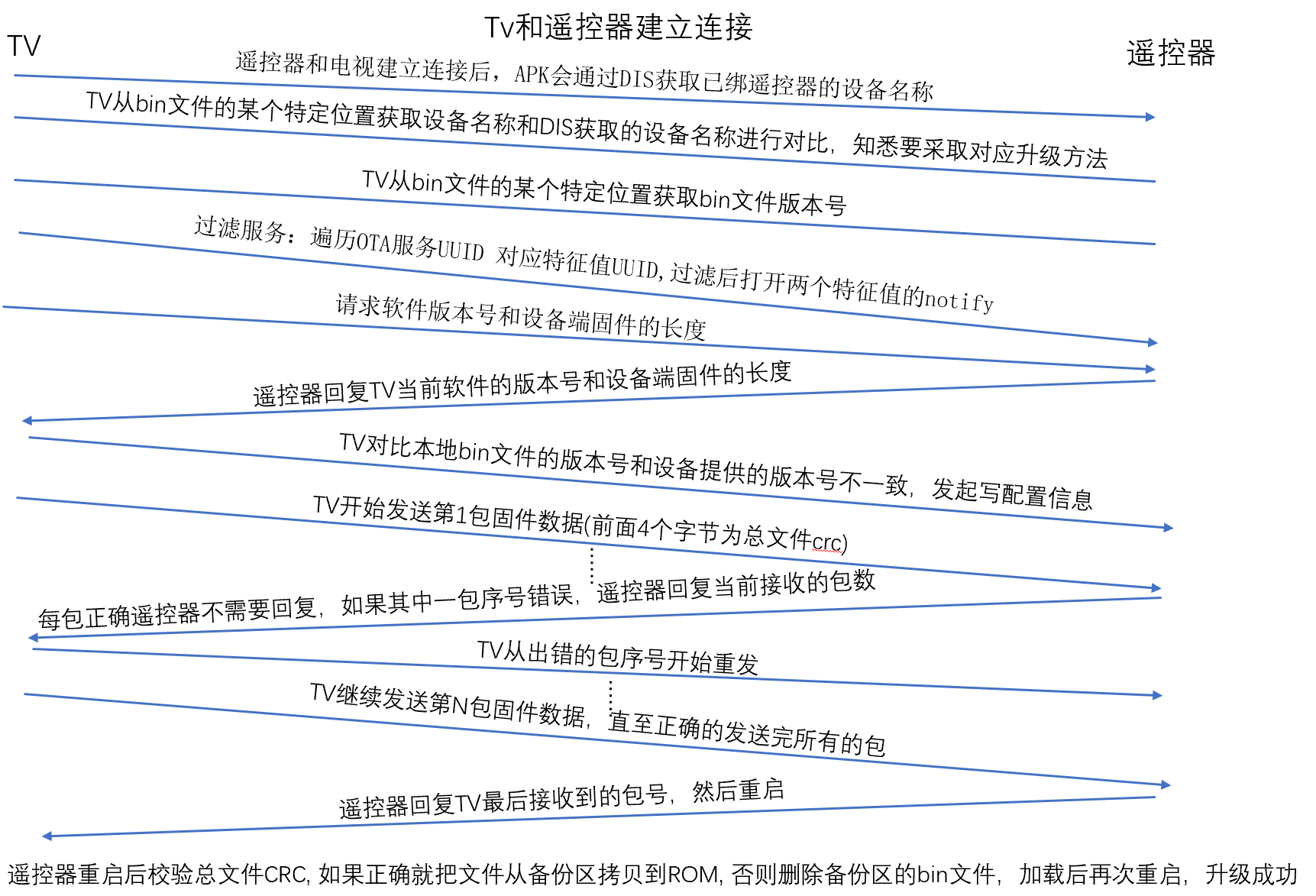
0x20015ffe-0x20015fff 版本号 4字节

# 3运行设计

典型流程的运行过程：

## OTA启动和升级流程

OTA升级的流程，更详细的内容请 参考资料1和资料4：



OTA升级的基本流程分为五大步骤:

连接设备、过滤OTA服务、过滤特征值并打开Notify、版本⽐较、发送bin⽂件；

1. **连接设备:**连接设备采⽤用Android 或 iOS 系统标准的BLE接口进⾏连接，具体操作请参考Android或iOS的操作⼿手册。
2. **过滤OTA服务：**成功连接上设备以后，我们需要发现设备的所有服务，并判断设备是否存在OTA服务(⼀个自定义的Service)。如果不存在OTA服务，则不能进⾏升级，通过服务的UUID进行判断。
3. **过滤特征值并打开Notify：**过滤出OTA服务以后，我们需要过滤出两个关键的特征值，并打开这两个特征值的Notfiy。这两个特征值一个是⽤用来升级的时候写⼊bin文件，另外一个是写，用来写入或读取设备端bin⽂件的配置信息
4. **版本比较:**在升级之前，APK会通过DIS服务去获取已绑遥控器的设备名称，然后获取APK本地bin文件某个位置存储的设备名称进行比较；

然后获取遥控器bin文件的版本号的方式可以通过向遥控器的OTA服务下面的特征值先后写入0、1，即可获取存放在bin文件头部第3-4个字节的文件版本号。然后APK获取本地bin文件某个位置存储的文件版本号进行对比决定是否需要升级。

1. **发送bin文件：**bin文件发送分为两个过程:发送文件的配置信息，发送文件；配置信息与发送文件分别对应两个不同的特征值；

写配置信息:在发送bin文件之前我们需要先对特征值(f000ffc1-0451-4000-b000-000000000000)写入12个字节的文件信息，具体写入内容请查看2.2.5小结的协议

写bin文件:写入bin文件其实就是向特征值(f000ffc2-0451-4000-b000-000000000000)写入数据的过程。每次发送18字节的有效数据,其中前两个字节表示发送的序号，后面16个字节表示文件数据；注意写这个特征值的时候，是从第0个字节开始发送;

重传方案：发送每帧OTA包时候，要考虑某一包发送不成功的情况。为避免这种情况发生，有两种参考思路：

第一种：尽可能快的发送(比如每5ms发送一包)，并监听发送结果的回调函数(比如Android监听onCharacteristicWrite回调函数)，监听到回调函数后再发送下一包，如果没有收到回调，就不断的重发,直到超时或收到回调.

第二种：尽可能低的降低发送速率(比如20ms发一包)，每发完一包再休息一定的时间。

性能评估： 为了缩短OTA升级时间，采用5ms一包的方式进行。OTA ble service 每个包160--200字节左右，有效载荷120字节。凑齐256字节，写一个扇区flash，以优化flash写速度。升级之前先擦除flash，因为擦除flash时间比较慢，擦除4K约400ms，因此不能在OTA升级过程中做擦除动作。 假定升级的bin是64KB，整个升级过程的数据部分在3秒左右完成。

## 3.2 系统开机流程（着重于bootloader对OTA的支持）

Remap

N

Y

APP image启动

APP image启动

OTA info判断

OTA支持持？

分区表信息读取

Bootloader启动（初始化）

OTA info判断选择当前需要使用的OTA分区。这个需要判断两个OTA分区的版本号以及OTA bin的状态。具体参考 5.2 OTA分区数据结构。

# 4.接口设计

## 4.1 OTA over ble

注册GAP回调函数，实现包括设备信息获取，连接，绑定，加密等消息的处理部分代码。

注册GATT 回调函数，实现包括OTA service的读写消息等部分的处理部分代码等。

注册OTA回调函数，实现包括OTA开始，停止，失败等状态处理部分代码。

详细请 参考资料1

## 4.2 OTA 流程管理模块

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| partinfo ota\_get\_boot\_partition() | 从**OTA info区**获取当前正在使用的OTA分区。Current working partition. |
| partinfo ota\_get\_update\_partition() | 从OTA info区获取本次要升级的OTA分区。To be updated partition. |
| appinfo ota\_get\_appimage\_info() | 从OTA区获取当前正在使用的app image的信息(版本号，设备名) |
| bool ota\_start(version, device) | 根据TV端的信息，进行是否开始OTA的判断。如果开始的话，进行OTA组件初始化 |
| bool ota\_write（data, length） | 接收到OTA数据包之后，往flash写操作。每次接收到新数据包之后，顺序写入。写入地址由流程管理模块统一控制。 |
| bool ota\_finish() | 完整收到所有OTA数据之后，OTA组件释放资源 |
| bool ota\_set\_boot\_partition() | OTA升级成功（完整的OTA数据写入OTA partition）之后，在OTA info区设置下一次上电时 bootloader选定的 OTA分区。 |
| bool ota\_abort() | 异常处理，OTA流程中止，OTA组件释放资源 |

## 4.2 OTA bin可用性管理模块

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| bool ota\_image\_verify() | 完整收到所有OTA数据之后，OTA组件做CRC检查，和TV端OTA之前发过来的CRC对比。对比失败的话，OTA升级中止。 |
| bool ota\_mark\_image\_version() | ota\_image\_verify成功之后，在OTAinfo区域更新当前最新的版本号。这个标记位会影响 ota\_get\_boot\_partition以及ota\_get\_update\_partition函数返回的结果。 |
| bool ota\_mark\_image\_invalid() | OTA升级成功首次开机，需要确认首次开机是否正常。如果不正常，在OTAinfo区域标记该版本为invalid。这个标记位会影响 ota\_get\_boot\_partition以及ota\_get\_update\_partition函数返回的结果。 |
| bool ota\_image\_check\_siga() | 收到完整的OTA bin数据之后，遥控器端通过签名确认这是可行的TCL遥控器的bin文件 |

数据全部收到再算hash和签名，还是每次收到256字节之后计算hash和签名。 综合考虑CPU计算量再确定。

首次开机的判断自检方案TBD。有两个方向可以判断：

* + - 1. 首次开机简单的自检（同生产自检）
      2. 首次开机不出现exception

# 5.主要数据结构和逻辑设计

## 5.1分区信息表 （partition info）

1. 数据结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分区名字 | 分区类型 | 开始地址 | 大小 | 属性 |
| Bootloader | app | 0x20000000 | 16KB | read |
| Partition info | data | 0x20004000 | 4KB | read |
| OTA info | data | 0x20005000 | 4KB | read/write |
| OTA1 | app | 0x20006000 | 64KB | read/write |
| OTA2 | app | 0x20016000 | 64KB | read/write |
| BT | app | 0x20026000 | 300KB | read |
| NV | data | 0x20071000 | 8KB | read/write |
| Log | data | 0x20073000 | 16KB | read/write |

typedef struct

{

char name[20];

int partitiontype;

int startaddress

int size;

int attribute;

} partitioninfo;

Partitioninfo ms\_partition[n];

属性可以用来做分区的写保护。

1. 分区信息导入逻辑

开发下载：

开机之后，app image将0x20004000强制转成 ms\_partition[n]

烧入工具 烧入配置文件到0x20004000

配置工具配置后生成bin

如果是生产线烧入或者烧入器烧入，需要把 bootloader，partition info，ota info， ota1，bt等各个bin都合并到一起生成一个完整的512K烧入image.

1. 客户配置

分区表支持用户配置。配置工具会对配置做检查。Bootloader，partition info，BT，nv，log分区都必须有，且大小不能修改。支持ota升级的话，必须有ota info分区。不支持ota升级的话，只有一个ota分区。多余的空间客户可以创建新的分区。

## 5.2 OTA分区信息 （OTA info）

当支持多个ota分区的时候，必须有ota info分区。

typedef struct

{

int index; // ota info record index. Growing from 1 to oxffff

char otapartitionname[20]; // partition name OTA1, OTA2 etc

short version; // 对应ota 分区bin版本号

short invalid; // 当前ota分区是否可用 0xffff 表示首次烧入完 未测试，下一次开机需要用。0x0:valid。0x1: invalid

char otapartitionname[20]; // partition name OTA1, OTA2 etc

short version; // 对应ota 分区bin版本号

short invalid; // 当前ota分区是否可用 0xffff 表示首次烧入完 未测试，下一次开机需要用。0x0:valid。0x1: invalid

} otapartinfo;

Flash加密，读出来的数据可能不是0xffffffff. 第一次开机写oxffffffff 加密后写入？

OTA info分区判断逻辑：从flash地址0x20005000（OTA info分区起始地址）开始， 每隔52字节读取 record index。 如果下一个record index为0xffffffff(flash未写入状态)，则上一个为最新的record。

OTA info区域数据内容的变化如下表所示：

|  |  |
| --- | --- |
| 步骤 | 分区内容 (从 0x20005000起) |
| OTA前 (首次烧入之后) | 0x00000001  “ota1”  0x0001 //假定第一个版本 版本号是 01  0x0000  “ota2”  0xffff //无效版本号 ota2分区不可用  0xffff |
| 第一次OTA升级完  （OTA image CRC check pass, 更新OTA info区域） | 0x00000002 //增加新的记录。方法：在最近的一次记录基础上修改增加  “ota1”  0x0001  0x0000  “ota2”  0x0002 //假定第2个版本版本号是02。因为版本号高，bootloader选择ota2  0xffff //升级完，未验证bin是否可用状态。下一次开机要测试。 |
| 第一次OTA升级完， 开机OK | 0x00000002  “ota1”  0x0001  0x0000  “ota2”  0x0002  0x0000 //升级完，下一次开机正常，invlaid置为0。 |
| 第一次OTA升级完， 开机失败 | 0x00000002  “ota1”  0x0001  0x0000  “ota2”  0x0002  0x0001 //升级完，下一次开机不正常，invalid置为1。 |
| 第二次OTA升级完  （OTA image CRC check pass, 更新OTA info区域） | 0x00000003 //增加新的记录。方法：在最近的一次记录基础上修改增加  “ota1”  0x0003 //假定第三个版本 版本号是 03  0xffff //升级完，未验证bin是否可用状态。下一次开机要测试。  “ota2”  0x0002  0x0000 |

注： Bootloader在读取OTAinfo区信息的时候，关注两个标记位。一个是版本号，一个是invalid。

两个分区invalid均为0，取版本号高的分区。

两个分区有一个分区invalid为oxffff， 且版本号高， 下次开机采用该分区，并且判断稳定性。

两个分区有一个分区invalid为ox1，不论版本号，该分区不可用，为带升级分区

# 6. 异常处理设计

|  |  |
| --- | --- |
| 异常类型 | 处理方案 |
| OTA升级耗电 | 1.只升级app， 减少升级的bin 大小，缩短升级时间  2. TV在OTA前会主动下发查询电量指令。遥控器电量小于 xx%不能升级。OTA升级过程中持续判断电量  3. 开展OTA升级过程中的优化 (PM模块确认升级时CPU最低频率等) |
| 升级中异常断电 | 升级过程按照 下载ota image🡪image完整性CRC校验🡪image可用性验证🡪更新ota info信息。如果中途掉电，则ota info的标志位没有置位，那么开机还是使用旧的ota分区。 |
| 版本可用性 | 升级ota版本之后，首次开机做稳定性判断。如果不出现死机，异常重启，则该版本可用。 |
| 用户分区配置异常 | 通过配置工具管控。OTA info, log, NV,BT等分区大小不能修改。Bootloader最大32KB。 |
| Flash加密 | 因为flash加密，读写NV, OTA info这些区域的操作，需要使用 flash\_stig\_read/write接口。这些接口里面要做地址空间的判断，只支持对对 NV, OTA info区域的操作。 |

# 7. 系统扩展

1. 不同的产品采用的OTA升级服务器不同。蓝牙遥控器是由TV（或者手机来模拟）进行升级，待升级的bin文件放在TV端。如果有其他的产品形态，比如像手机的OTA升级一样，直接搭建OTA升级服务器给所有终端升级。不同的升级服务器应该对这个OTA升级框架没有影响。
2. 当前我们只支持ble，所以ota通过ble进行。后续cygnus平台支持WiFi，需要在不改变当前设计的前提下，可以增加ota over wifi（http下载）。