**CMIOT FOTA还原库适配使用说明**

Version: 1.3

Date: 2023-04-14



[[iot.10086.cn](http://www.spreadtrum.com)](http://iot.10086.cn/)

**修订记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编号** | **版本号** | **修订日期** | **修订描述** | **修改人** |
| 1 | V1.0 | 2023-02-01 | 初稿 | OneOS Team |
| 2 | V1.1 | 2023-02-20 | 增加bootloader适配 | OneOS Team |
| 3 | V1.2 | 2023-02-23 | 更新平台使用说明 | OneOS Team |
| 4 | V1.3 | 2023-04-14 | 更新fota\_boot V2.1.0 | OneOS Team |
|  |  |  |  |  |

目 录

[1 背景 3](#_Toc3010)

[2 适配说明 3](#_Toc26284)

[2.1 Bootloader适配 3](#_Toc5996)

[2.1.1 资源 3](#_Toc482)

[2.1.2 数据结构 3](#_Toc18643)

[2.1.3 接口描述 4](#_Toc11189)

[2.1.4 返回码表 6](#_Toc15321)

[2.1.5 适配流程 7](#_Toc319)

[2.1.6 参数配置 9](#_Toc12042)

[3 平台使用说明 10](#_Toc19813)

[3.1 创建产品 10](#_Toc22037)

[3.2 制作版本包 14](#_Toc16365)

[3.3 制作差分包 15](#_Toc10521)

[3.4 删除版本 20](#_Toc8749)

[4 注意事项 20](#_Toc24344)

[5 参考资料 21](#_Toc7700)

[6 FAQ 21](#_Toc30038)

# 背景

CMIOT FOTA 主要包含OTA组件，OTA\_BOOTLOADER和云平台几个部分，各个模块均附有详细的接口及使用说明文档。该文档从初次使用CMIOT FOTA的角度，一步一步介绍全流程操作及注意事项，以及各个模块适配注意事项，旨在帮助用户快速的体验FOTA升级服务。文档目前仅适用于采用自研算法和CMS平台的升级服务场景。

# 适配说明

## Bootloader适配

### 资源

在bootloader阶段，需要依赖的资源有：

①内存，用于存放还原过程中的数据，通常内存空间越大越好；

②堆栈，程序必须运行在堆栈已经完成正确初始化的C环境下；

③flash操作接口，包括下载及升级相关的存储介质的读/写/擦除接口。

### 数据结构

typedef struct ota\_fal\_flash\_ops

{

ota\_int32 (\*init)(void);

ota\_int32 (\*read)(ota\_uint32 addr, ota\_uint8 \*buf, ota\_size\_t size);

ota\_int32 (\*write)(ota\_uint32 addr, const ota\_uint8 \*buf, ota\_size\_t size);

ota\_int32 (\*erase)(ota\_uint32 addr, ota\_size\_t size);

} ota\_fal\_flash\_ops\_t;

typedef struct ota\_fal\_flash

{

ota\_int8 name[FAL\_NAME\_MAX]; /\* flash 名称 \*/

ota\_size\_t size; /\* flash 大小 \*/

ota\_size\_t erase\_size; /\* flash 最小擦除单位 \*/

ota\_fal\_flash\_ops\_t ops; /\* flash 读写擦操作函数 \*/

} ota\_fal\_flash\_t;

typedef struct ota\_fal\_part\_info

{

ota\_int8 name[FAL\_NAME\_MAX]; /\* 分区 名称 \*/

ota\_int8 flash\_name[FAL\_NAME\_MAX]; /\* 分区 所在flash名称 \*/

ota\_uint32 address; /\* 分区 地址 \*/

ota\_uint32 size; /\* 分区 大小 \*/

ota\_uint32 part\_type; /\* 分区 类型 \*/

} ota\_fal\_part\_info\_t;

typedef struct ota\_fal\_map

{

ota\_uint32 logic\_addr; /\* 分区 逻辑地址 \*/

ota\_uint32 phy\_addr; /\* 分区 物理地址 \*/

} ota\_fal\_map\_t;

### 接口描述

1. *void ota\_printf(const ota\_int8 \*fmt, ...)*

功能描述：LOG格式化输出接口，升级过程中的LOG信息，以及返回值通过该接口输出。

该接口需用户实现

1. *void ota\_update\_proc\_bar(ota\_int32 percentage)*

功能描述：升级进度条显示

输入参数：ota\_int32 percentage： 当前升级进度：0~100

该接口需用户实现，在升级过程中被调用，不需要应用代码主动调用。通常是将进度条通过屏幕显示，因此需要在该函数中实现屏幕显示进度条的功能。

1. *ota\_int32 ota\_heap\_init(ota\_uint8 \*buf, ota\_size\_t buf\_size, ota\_size\_t block\_size)*

功能描述：堆空间初始化接口

输入参数：ota\_uint8 \*buf：用于OTA还原升级的内存起始地址；

ota\_size\_t buf\_size：用于OTA还原升级的内存大小；

ota\_size\_t block\_size：内存块大小；

返回值： 0：成功，小于0：失败

当bootloader中堆不可用时，可以使用该接口使用内置堆管理。提供给还原升级的内存应尽可能大，除了系统栈所需内存，和预留静态占用的内存外，剩余内存可以都用于还原升级。该内存可以通过定义一个静态数组的方式进行预分配。 为了提高内存的利用率，block\_size不宜过大，buf\_size需与CMIOT\_FOTA\_AVAILABLE\_RAM\_SIZE保持一致，内置堆管理模块会对buf\_size 和block\_size进行合理性检查，当初始化返回失败时，根据提示log进行调整即可。

1. *ota\_int32 ota\_fal\_init(ota\_fal\_flash\_t \*flash\_tab, ota\_size\_t flash\_tab\_size, ota\_fal\_part\_info\_t \*part\_tab, ota\_size\_t part\_tab\_size)*

功能描述：注册分区信息

输入参数：ota\_fal\_flash\_t \*flash\_tab：flash信息，包含flash名称，大小，擦除大小，读写擦操作函数。

ota\_size\_t flash\_tab\_size：flash列表大小

ota\_fal\_part\_info\_t \*part\_tab：分区信息，包含分区名称，分区所在flash名称，分区起始地址，分区大小和分区类型

ota\_size\_t part\_tab\_size：分区信息表大小

返回值： 0：成功，小于0：失败

当需要进行地址映射时，分区表中的地址应当使用逻辑地址。

1. *ota\_int32 ota\_fal\_map\_init(ota\_fal\_map\_t \*map\_table, ota\_size\_t map\_table\_size)*

功能描述：分区地址映射初始化

输入参数：ota\_fal\_map\_t \*map\_table：逻辑地址与物理地址映射表

ota\_size\_t map\_table\_size：逻辑地址与物理地址映射表大小

返回值： 0：成功，小于0：失败

若某些分区使用逻辑地址，某些分区使用物理地址，则使用物理地址的分区，逻辑地址与物理地址保持一致即可。

1. *ota\_int32 ota\_start\_up(ota\_uint32 try\_times)*

功能描述：启动升级

输入参数：ota\_uint32 try\_time：升级重试次数。建议至少重试一次

返回值： 升级结果返回码

### 返回码表

表 1-1 返回码表

|  |  |
| --- | --- |
| \*\*状态码\*\* | \*\*说明\*\* |
| \*\*1000\*\* | 升级成功 |
| \*\*1001\*\* | 不需要升级 |
| \*\*\*1002\*\*\* | 内存不足 |
| \*\*\*1003\*\*\* | 无效升级包 |
| \*\*\*1004\*\*\* | 前置版本校验错误 |
| \*\*\*1005\*\*\* | 升级包类型错误 |
| \*\*\*1006\*\*\* | 升级包不存在 |
| \*\*\*1007\*\*\* | DL分区空间不足 |
| \*\*\*1008\*\*\* | 未找到分区 |
| \*\*\*1009\*\*\* | 分区读取错误 |
| \*\*\*1010\*\*\* | 分区写入错误 |
| \*\*\*1011\*\*\* | 分区擦除错误 |
| \*\*\*1012\*\*\* | 分区数据备份错误 |
| \*\*\*1013\*\*\* | 分区数据还原错误 |
| \*\*\*1014\*\*\* | 流信息初始化错误 |
| \*\*\*1015\*\*\* | 头信息初始化错误 |
| \*\*\*1016\*\*\* | 解压头错误 |
| \*\*\*1017\*\*\* | 解压数据体错误 |
| \*\*\*1018\*\*\* | 还原块长度错误 |
| \*\*\*1019\*\*\* | 差分包数据校验错误 |
| \*\*\*1501\*\*\* | 新APP块数据校验错误 |
| \*\*\*1502\*\*\* | 覆盖线差值计算错误 |
| \*\*\*1503\*\*\* | 构建新数据错误 |
| \*\*1504\*\* | DL备份区域擦除失败 |
| \*\*1505\*\* | 解析块还原编号错误 |
| \*\*1506\*\* | 解析块覆盖线数量错误 |
| \*\*1507\*\* | 新APP分区数据校验错误 |

### 适配流程

1. Flash操作接口

ota\_int32 (\*init)(void);

flash的初始化一般在bootloader硬件初始化阶段已完成，init接口为NULL；若需要在fal初始化中完成，则可以在init接口中完成flash的硬件初始化。成功返回0，失败返回-1。

ota\_int32 (\*read)(ota\_uint32 addr, ota\_uint8 \*buf, ota\_size\_t size);

数据读取接口，成功返回读取长度，失败返回-1

ota\_int32 (\*write)(ota\_uint32 addr, const ota\_uint8 \*buf, ota\_size\_t size);

数据写入接口，成功返回写入长度，失败返回-1

ota\_int32 (\*erase)(ota\_uint32 addr, ota\_size\_t size);

数据擦除接口，其中addr按block对齐，size不是block的整数倍时，按照最小对齐block擦除，成功返回0，失败返回-1

将操作函数，flash名称，容量，块大小（擦除大小）填写至ota\_fal\_flash\_t类型的表中。如果有多个不同的flash设备，根据实际情况，实现相应的操作函数，并将硬件信息添加到flash表中即可。

1. 填写分区表

ota\_fal\_part\_info\_t类型的分区表中包含了需要升级的“APP”分区以及“download”分区，除了download分区名称是固定的，其他“APP”的名称可以任意命名。每个分区，包含了分区名称，所在flash名称，起始地址（如果使用了逻辑地址需填写逻辑地址），分区大小，分区类型（“APP”必须为FAL\_UPGRADE\_PART类型，“download”必须为FAL\_DOWNLOAD\_PART类型），分区名称不可重复。

1. 配置参数

根据实际情况，修改oneos\_config.h中的配置参数（见2.1.6参数配置章节）

1. 初始化地址映射（必要时）

若使用逻辑地址，需要填写ota\_fal\_map\_t类型的地址映射表，表中必须包含分区表中所有的分区地址映射信息，若某些分区不使用逻辑地址，则逻辑地址与物理地址保持一致。通过ota\_fal\_map\_init接口完成地址映射初始化。

1. 初始化分区信息

将flash表和分区表作为参数，调用ota\_fal\_init接口完成分区信息初始化。

1. 初始化堆空间（必要时）

若使用内置堆管理，则需要调用ota\_heap\_init接口进行堆空间初始化。

1. 检查升级状态

在bootloader启动时，读取download分区最后一个块起始存放的前2个成员数值，判断是否需要进入还原升级：

struct cmiot\_update\_state

{

unsigned char inited;

unsigned short update\_result;

};

update\_result等于100，表示需要进入还原升级

1. 启动升级

调用ota\_start\_up接口，启动升级。对比返回码，可以查看升级结果以及错误码

### 参数配置

在oneos\_config.h中，用户可配置的FOTA相关的几个重要参数如下：

1. 可用内存大小CMIOT\_FOTA\_AVAILABLE\_RAM\_SIZE

示例：#define CMIOT\_FOTA\_AVAILABLE\_RAM\_SIZE 0x80000

支持十进制/十六进制，行尾注释前必须加空格。该宏定义了fota还原升级时真实可用的内存大小，在条件允许的情况下，内存越大越好。

1. Download分区大小CMIOT\_UPDATE\_SIZE

示例：#define CMIOT\_UPDATE\_SIZE 0xA0000

支持十进制/十六进制，行尾注释前必须加空格。该宏定义了差分包下载空间（即download分区）大小，在制作差分包时会对其进行校验，如果差分包的实际大小超过该值定义的大小，会提示制作差分包失败，用户需提前规划好download分区大小，不允许升级版本时再作改变。如果空间有限，可以采取多次少量进行升级。

1. Flash擦除大小CMIOT\_DEFAULT\_SECTOR\_SIZE

示例：#define CMIOT\_DEFAULT\_SECTOR\_SIZE 0x10000

支持十进制/十六进制，行尾注释前必须加空格。该宏定义了FLASH的块擦除大小，必须和实际flash的块擦除大小一致。制作差分包和还原升级，会以此作为计算和存储的基本单位。

1. 固件版本CMIOT\_FIRMWARE\_VERSION

示例：#define CMIOT\_FIRMWARE\_VERSION "1.0"

字符串，行尾注释前必须加空格。该宏定义了固件版本，升级版本时，版本号需同时进行迭代。

1. 下载块大小CMIOT\_DEFAULT\_SEGMENT\_SIZE\_INDEX

示例：#define CMIOT\_DEFAULT\_SEGMENT\_SIZE\_INDEX 3

十进制/十六进制，行尾注释前必须加空格。该宏定义了下载差分包时默认的buff大小，该值越大，单次下载的数据包长度更大，总的下载时间会减少，最大可设置为5。

1. 升级文件CMIOT\_FOTA\_FILENAMEx，CMIOT\_FOTA\_ADDRESSx

示例：#define CMIOT\_FOTA\_FILENAME1 "mmi\_res\_240x240\_child\_circular.bin"

#define CMIOT\_FOTA\_ADDRESS1 0x90000004

CMIOT\_FOTA\_FILENAMEx文件名为字符串，行尾注释前必须加空格。与实际文件名称对应，制作版本包时，会根据该名称检索文件。如果是单bin，则x为空，如果是多bin，x=1，2...，文件名必须是以常见的二进制文件结尾，如’\*.bin’,’\*.img’等，若差分文件为其他后缀，可以在文件末尾添加’.bin’。CMIOT\_FOTA\_ADDRESSx文件地址，必须是十六进制格式，如果使用逻辑地址，也应当符合地址的特征，如长度为8位。

1. 全量/差分升级CMIOT\_FOTA\_USE\_FULLx

示例：#define CMIOT\_FOTA\_USE\_FULL2

CMIOT\_FOTA\_USE\_FULLx，表示对应的CMIOT\_FOTA\_FILENAMEx升级模式为全量升级。默认为差分升级。当目标版本需要做全量升级时，可定义该宏。

# 平台使用说明

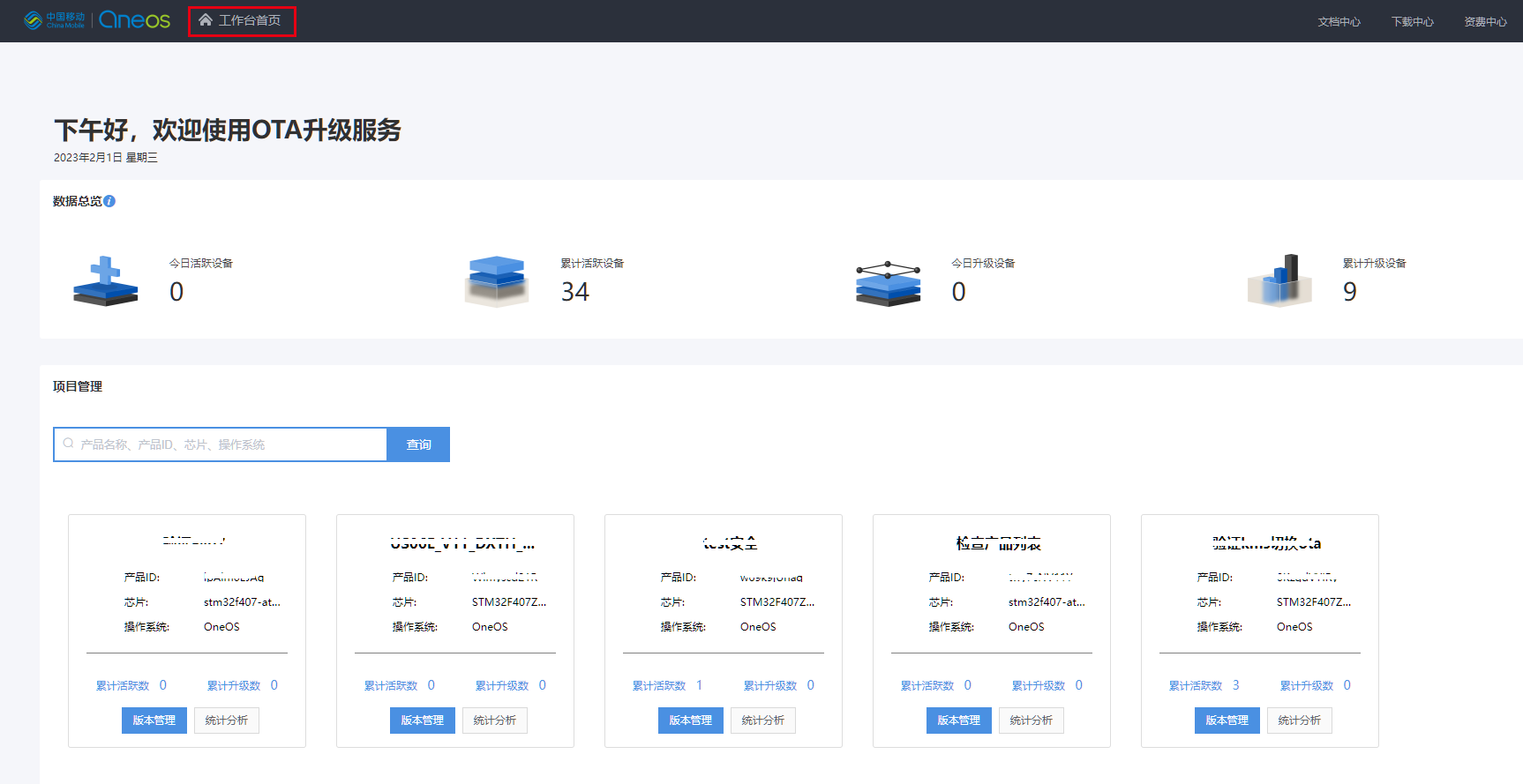
## 创建产品

①登录cms工作台OTA升级服务平台:

https://os.iot.10086.cn/user-center#/user/login



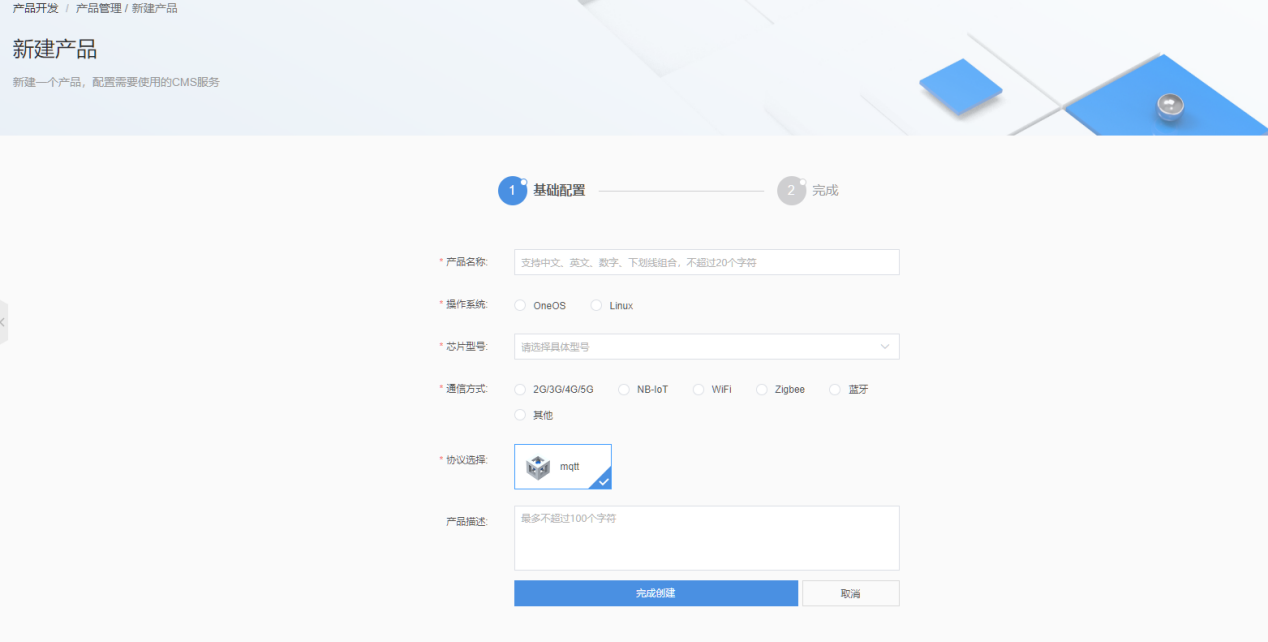
②进入工作台首页



③依次进入【产品开发】【产品管理】选项，点击【新建产品】



④完善产品信息



⑤点击【完成创建】【完成】，即完成产品创建。记录**产品ID**和**产品密钥**备用。



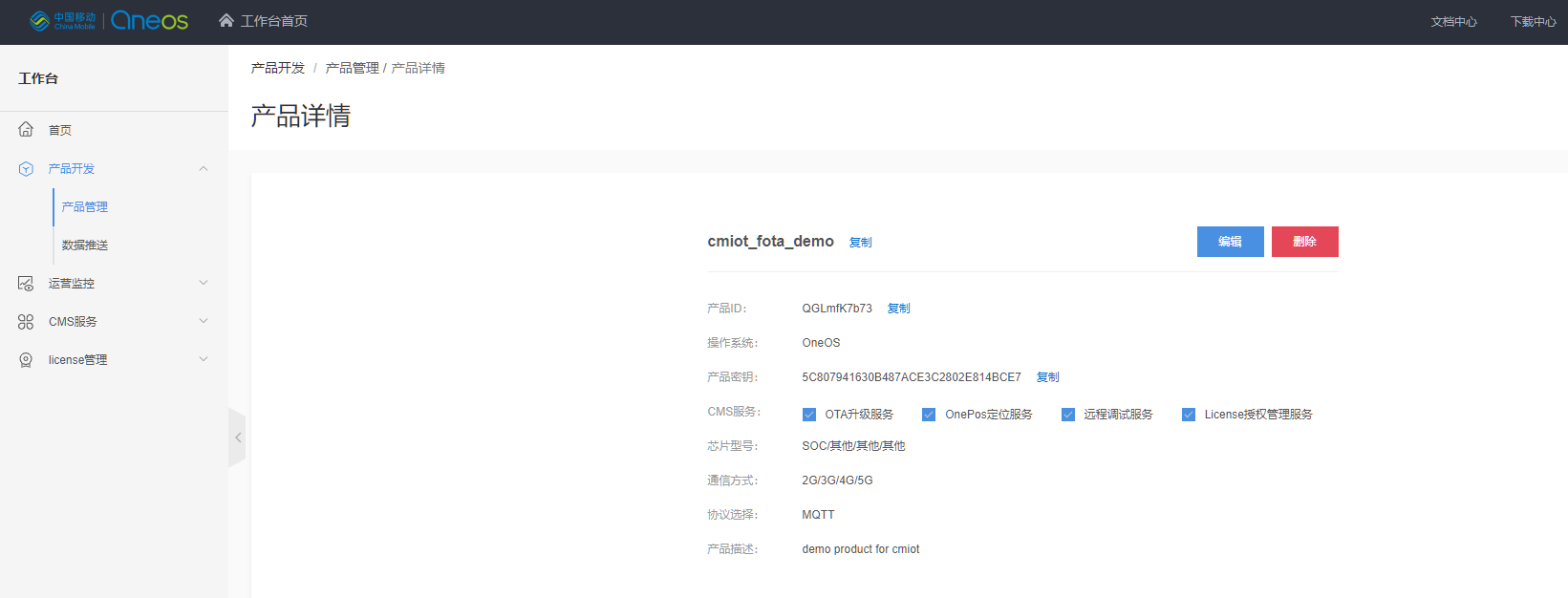


创建完成产品后，可以在【产品开发】【产品管理】选项卡中，查询到产品信息



点击缩略信息可以进入产品详情页





## 制作版本包

①在oneos\_config.h中修改cms产品信息，如果支持OneOS Cube配置，则通过menuconfig进行修改。

/\* CMS ID \*/

#define CMS\_USING\_ID

#define CMS\_ID\_SET\_PID "QGLmfK7b73"/\*产品ID\*/

#define CMS\_ID\_KEY "5C807941630B487ACE3C2802E814BCE7" /\*产品密钥\*/

#define UNILATERALISM\_AUTH

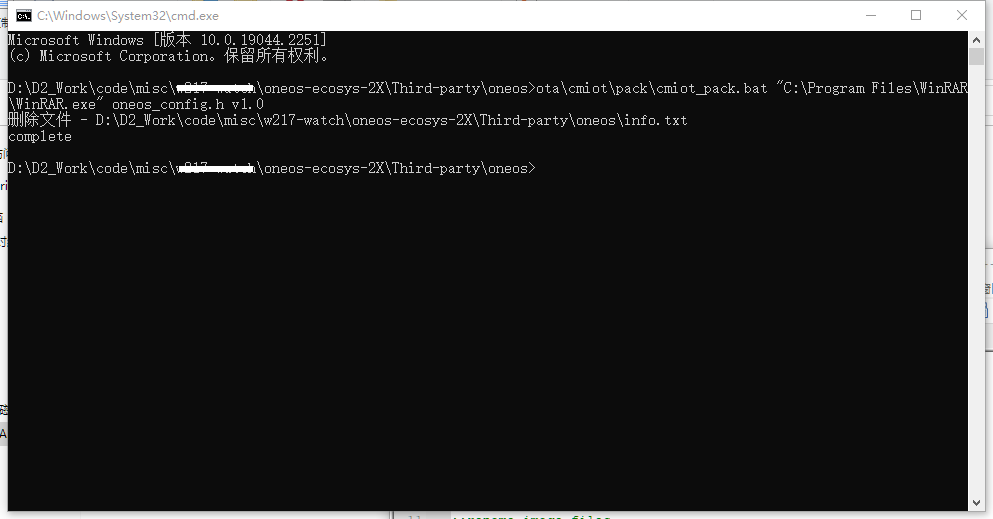
/\* end of CMS ID \*/

②在oneos\_config.h中修改固件版本，如果支持OneOS Cube配置，则通过menuconfig进行修改。

#define CMIOT\_FIRMWARE\_VERSION "1.0"

③PC上安装压缩软件WinRAR，通过cmiot\_pack.bat脚本完成版本包制作，在oneos\_config.h所在目录，打开cmd，执行如下指令

ota\cmiot\pack\cmiot\_pack.bat "C:\Program Files\WinRAR\WinRAR.exe" oneos\_config.h v1.0



生成的版本包名称的前缀为版本和日期



④修改源码，同时在oneos\_config.h中修改版本号：例如修改为2.0，并执行步骤③中的指令，生成2.0的版本包

#define CMIOT\_FIRMWARE\_VERSION "2.0"

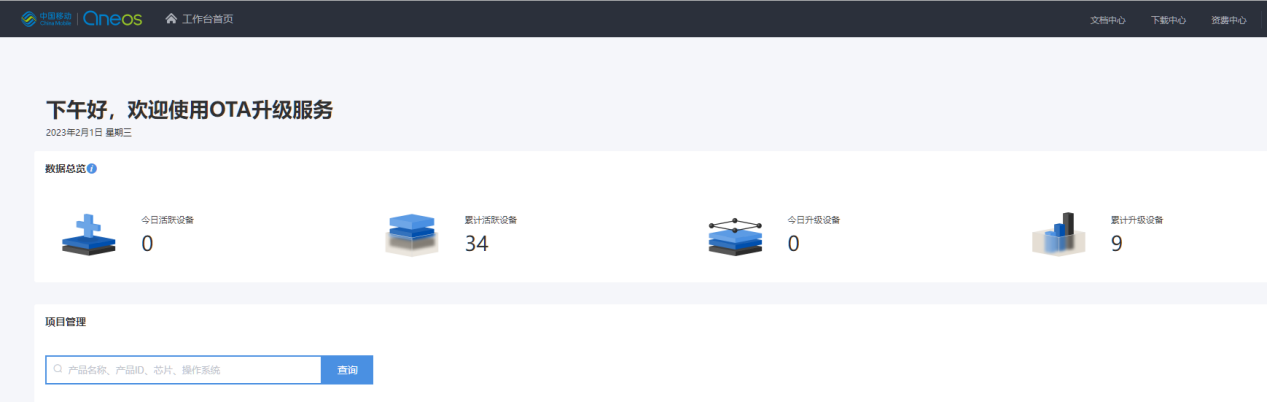


## 制作差分包

版本包制作完成后，通过平台生成差分包并发布。

①登录cms工作台OTA升级服务平台，并选择【CMS服务】【OTA升级服务】选项卡，进入OTA页面





②找到产品选项卡，并点击【版本管理】



③点击【添加版本】



④完善版本信息，并上传版本包



⑤按照步骤④的操作，完成2.0版本的固件上传，即可看到两个版本的信息。



⑥点击【配置升级任务】【配置】，进入配置页面，点击【确定】。







⑦稍作等待，即可完成差分包的制作接下来进入发布测试发布流程，依次点击【开始测试】【测试通过】【发布】







注意，新版本发布需等待几分钟后，设备端方可检测到新的版本。



## 删除版本

在【版本管理】页面，点击【更多操作】可进行已发布版本的删除。



# 注意事项

①压缩软件推荐使用WinRAR；

②download分区除差分包存放区域外，还另外需要两个区域用于备份和存放信息，所以download分区大小至少需要三个扇区；

③flash操作接口适配时，当读写成功时返回读写数据的长度，读写失败时，返回-1；擦除失败时返回-1，擦除成功返回0；

④在没有完善堆可使用ota\_heap\_init初始化升级使用内存，有完善堆空间时可不使用。

# 参考资料

[1]CMS工作台：[文档中心 (10086.cn)](https://os.iot.10086.cn/v2/doc/detailPage/documentHtml?proId=2000000002&proName=CMS%E5%B7%A5%E4%BD%9C%E5%8F%B0&idss=177&versionName=v1.0&versionId=3000000002)

[2]FOTA服务：[文档中心 (10086.cn)](https://os.iot.10086.cn/v2/doc/detailPage/documentHtml?proId=2000000004&proName=FOTA%E6%9C%8D%E5%8A%A1&idss=199&versionName=v1.0&versionId=3000000003)

[3]视频教程：[OneOS学院 (10086.cn)](https://os.iot.10086.cn/college/)

# FAQ

Q：如何合理的配置download分区大小？

download分区配置为需要升级的尺寸总和的10~50%之间，若空间非常紧张，建议设置为最大应用尺寸的20%。

Q：可用RAM大小会对性能有哪些影响？

ram空间不足时，第一，可能导致还原过程中内存分配失败，从而增加升级失败的风险；第二，不会启用二次压缩，相同的版本，制作的差分包会更大，分配给算法ram空间配置尽可能大，最小不低于3个block（最小擦除）大小。

Q：升级过程中断电会有什么后果？

升级过程中断电，重新上电后会继续升级。但是升级过程中断电有可能带来其他风险，因此推荐在升级在点亮充足的条件下进行，并且过程中明确的提示用户不要断电。

Q：哪些因素会影响升级的时间？

第一，版本差异大小。升级前后版本差异越小，需要更新的flash越少，擦写flash的总次数越少，耗时也就越少；第二，存储介质擦写耗时，升级相同的版本，存储介质擦写耗时越少，总的耗时越少。

Q：在一个平台上适配的还原库，可以直接应用到其他项目上吗？

主要取决于以下几个方面：1.芯片架构，2.编译工具，3.算法配置；如果这几项都一样即可可以直接应用到其他项目上。