APP Data Tool Quick Start Guide

V1.0

2023/11/01

修订历史（Revision History）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 日期 | 版本 | 修改 | 作者 | Reviewer |
| **2023/11/01** | V1.0 | First Release | Arthur |  |

目录

[修订历史（Revision History） 2](#_Toc149741716)

[目录 3](#_Toc149741717)

[1 概述 4](#_Toc149741718)

[2 操作步骤 5](#_Toc149741719)

[2.1 APP Data1 5](#_Toc149741720)

[2.2 APP Data2 5](#_Toc149741721)

[2.3 APP Data3~6 6](#_Toc149741722)

[3 APP Data的版本管理 7](#_Toc149741723)

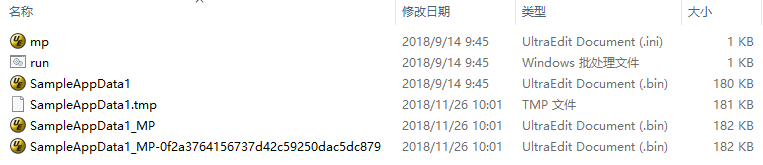
# 概述

APP Data Tool是将原始的APP Data的bin文件转换成可通过RTL87x2G MPTool烧录到External Flash中的APP Data类型的image。APP Data Tool处理后，会在原始的APP data前面添加一个512 bytes MP header 和1280 bytes image header和，分别用于MPTool烧录和Bootloader校验。假设原始的APP Data的bin文件的大小是120KB，最后烧录到flash中将是1280 bytes image header加上120KB的APP data payload。

# 操作步骤

## APP Data1

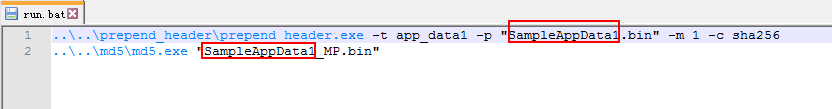
1. sdk\tools\AppData\AppData1目录如下：



SampleAppData1.bin是原始的APP Data的bin文件。

SampleAppData1\_MP-0f2a3764156737d42c59250dac5dc879.bin是由tool生成的用于烧录的APPData1的bin文件。

1. 打开“run.bat”，将以下两部分替换成需要处理的APPdata的bin文件名。注意第二行中的必须是“xxx\_MP.bin”，**后缀“\_MP”不可更改**。



1. 运行“run.bat”，将会额外生成3个文件：
2. SampleAppData1.tmp：是在原始的APP Data的bin文件上添加了1280 bytes image header的临时文件；
3. SampleAppData1\_MP.bin：是在临时文件上再添加了512 bytes的MP Header的bin文件；
4. SampleAppData1\_MP-0f2a3764156737d42c59250dac5dc879：在“xxx\_MP.bin”的基础上计算了MD5的校验值，**注意只有同时带有\_MP和MD5校验的bin文件才可以最后用于烧录和OTA。**

## APP Data2

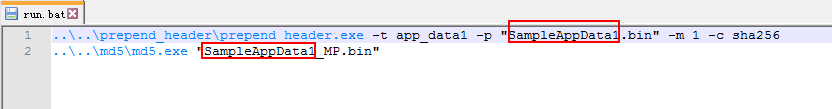
1. sdk\tools\AppData\AppData2目录如下：



SampleAppData2.bin是原始的APP Data的bin文件。

SampleAppData2\_MP-d873c53e6ba57dfc476defeba1794f97.bin是由tool生成的用于烧录的APPData2的bin文件。

1. 打开“run.bat”，将以下两部分替换成需要处理的APPdata的bin文件名。注意第二行中的必须是“xxx\_MP.bin”，**后缀“\_MP”不可更改。**



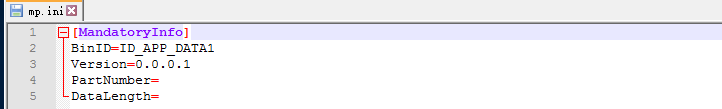
1. 运行“run.bat”，也会额外生成3个文件。
2. SampleAppData1.tmp：是在原始的APP Data的bin文件上添加了1280 bytes image header的临时文件；
3. SampleAppData1\_MP.bin：是在临时文件上再添加了512 bytes的MP Header的bin文件；
4. SampleAppData1\_MP-0f2a3764156737d42c59250dac5dc879：在“xxx\_MP.bin”的基础上计算了MD5的校验值。**同样地，只有带有\_MP和MD5校验的bin文件才可以最后用于烧录和OTA。**

## APP Data3~6

操作类似上面APP data1和APP data2的处理，在此不做赘述。

# APP Data的版本管理

在sdk\tools\AppData\AppDataX目录下均有一个“mp.ini”的配置文件如下图所示，通过修改该配置文件即可生成不同版本的APP data的bin文件。



其中“Version=0.0.0.1”即表示生成的APP data的版本。该点分的四段式版本号表示的版本信息是**“Major. Minor.Revision.Reserve”**，总共占32 bit，对应如下的结构体。

typedef struct

{

union

{

uint32\_t version;

struct

{

uint32\_t \_version\_major: 4; //!< major version

uint32\_t \_version\_minor: 8; //!< minor version

uint32\_t \_version\_revision: 15; //!< revision version

uint32\_t \_version\_reserve: 5; //!< reserved

} img\_sub\_version; //!< other image sub version including patch, app, app data1-6

} ver\_info;

} T\_IMAGE\_VERSION;

该版本信息会保存在前1280 bytes的image header中git\_ver字段，偏移0x1fe。

typedef union \_IMG\_HEADER\_FORMAT

{

uint8\_t bytes[DEFAULT\_HEADER\_SIZE];

struct

{

T\_AUTH\_HEADER\_FORMAT auth;

T\_IMG\_CTRL\_HEADER\_FORMAT ctrl\_header;

uint8\_t uuid[16];

uint32\_t exe\_base;

uint32\_t load\_src;

uint32\_t load\_len;

uint32\_t image\_base;

uint16\_t dev\_id; // for AMZN

uint16\_t flash\_layout\_size\_4k; // ex. 1MB flash used size => flash\_layout\_size\_4k = 1MB/4KB = 256

uint32\_t magic\_pattern;

uint8\_t dec\_key[16];

uint32\_t load\_dst;

T\_EXTRA\_INFO\_FORMAT ex\_info;

T\_VERSION\_FORMAT git\_ver;

PUBLIC\_KEY PubKey;

T\_FLASH\_SEC\_FORMAT flash\_sec\_cfg;

union

{

uint8\_t reserved\_data[344];

struct

{

uint32\_t ver\_val;

uint32\_t image\_info[(IMG\_MAX - IMG\_OTA - 1) \* 2];

uint32\_t reserved0[45];

};

struct \_\_attribute\_\_((packed))

{

uint32\_t tool\_version;

uint32\_t timestamp;

uint16\_t reserved1[168];

};

};

};

} T\_IMG\_HEADER\_FORMAT;