瑞芯微RK3399K固件备份三种方式

文章目录

瑞芯微RK3399K固件备份三种方式

1. 前言

2. 获取分区表

3. 方式一：开发工具直接导出镜像

3.1 分区简单介绍及导出分区镜像

3.2 注意点以及可能的问题

5. 方式二：fireflydev生成镜像

6. 方式三：dd命令直接制作

6.1 配置RK3399设备ssh

6.2 同步根目录到远端PC

6.3 PC上制作rootfs镜像

6.4 烧写rootfs

6.5 解包及打包update.img后烧写

6.6 烧写后/dev/root大小不正确的问题解决

1. 前言

前面我们介绍了RK3399K以及其镜像烧写（https://blog.csdn.net/weixin\_39510813/article/details/117261824），然后我们在开发完程序后，为了方便生产，我们直接备份整个固件方便批量生产。

这期间折腾了不少方法，最后总算是成功备份了整个rootfs，这里总结一下。

2. 获取分区表

连接设备并通过设备分区表功能获取设备分区，如下（关于如何连接设备请查看之前总结的镜像烧写部分）：

3. 方式一：开发工具直接导出镜像

3.1 分区简单介绍及导出分区镜像

根据我们的需求通过导出镜像功能设置扇区后导出即可，这里再简单介绍下各个分区的功能，然后我们根据自己的情况导入不同镜像来备份升级即可。

https://wiki.t-firefly.com/Core-3328-JD4/linux\_compile.html#da-bao-gu-jian

uboot: 程序，一个主要用于系统的引导加载程序，uboot分区简单理解就是运行存储uboot程序的磁盘分区，该程序虽然也需要开发但是很少动，所以基本上备份一次即可

trust: 是 U-Boot 作为二级 loader 的打包

misc: 用来控制是正常启动，还是进入急救模式（Recovery Mode）

boot: 内核的内存启动盘 (initrd)，是内核启动后最先加载的根文件系统，包含重要的初始化动作，一般不需要改动。

recovery: 急救模式的映像，内含内核和急救模式的根文件系统。

backup: 预留,暂时没有用。后续跟 android 一样作为 recovery 的 backup 使用.（可省略）

oem: 给厂家使用,存放厂家的 app 或数据。只读。代替原来音箱的 data 分区。挂载在/oem 目录.（可省略）

rootfs: 存放 buildroot 或者 debian 编出来的 rootfs.img,只读

userdata: 存 放 app 临 时 生 成 的 文 件 或 者 是 给 最 终 用 户 使 用 。 可 读 写 , 挂 载 在/userdata 目录下.（可省略）

这样子看下来我目前最多备份rootfs就可以了，其它的基本不会动。

然后我们切换到高级功能根据获取到的分区表中扇区信息导出对应的镜像即可：

导出成功后的img在开发工具的同级目录的output文件夹下（注意：最好是将所有分区全部导出后合并成总的update.img后升级，因为像我的板子实际上有16G大小，其中rootfs分区占了14G，但是实际上我们烧写的rootfs分区只烧写了4G左右，其它的是剩余的，而导出各个分区后合并的update.img只有5G左右）

3.2 注意点以及可能的问题

我在使用2.61和2.65版本的工具发现导出都立刻失败了，是立刻失败，不是一段时间后失败，然后在论坛里找到了解决思路（https://dev.t-firefly.com/thread-99487-1-1.html

“嗨，我也遇到了同样的问题，我的解决办法是使用低版本AndroidTool，我的版本是V2.42，我成功的将uboot 、trust、kernel等备份出来了”）我这里是换了低版本2.38可以导出镜像，但是官方下载速度慢的。。。(可以使用迅雷加速一下会好一些)

2.38版本的没有设备分区表功能，对应位置是低格功能。还有2.3x默认为中文，2.6x版本默认为英文，更换中文需要安装目录下修改config.ini文件，里面有切换中英文的说明。2.58的默认也是中文，而且分区表一次读取成功，是匹配Ubuntu系列的分区表的。

这个是给到的几个版本的开发工具，我估计各个版本对应不同产品，具体对应关系我目前还不是很清楚，只发现最适合rk系列的分区的似乎是2.58版本（通过这个文章尝试2.58的：https://blog.csdn.net/oxp7085915/article/details/80291057）。

这个是目前的开发工具的各个版本（https://www.t-firefly.com/doc/download/page/id/3.html）：

不明白为何隐去了2.58版本？（http://download.t-firefly.com/product/RK3328/Tools/AndroidTool/AndroidTool\_Release\_V2.58.zip）

可以看到2.58版本的下面有很多rk系列的配置文件，这在其它版本开发工具下面是没有的，默认的分区也是和rk系列匹配的，所以最后我是使用的2.58版本的获取分区信息，然后使用2.38的获取固件，然后再使用2.58进行升级，所以如果你使用其它版本的开发工具升级固件失败之类的也可以换一下其它版本的试一下，很可能是开发工具版本不匹配。

所以，这里也可以导入不同的配置：

2.38:

2.65:

5. 方式二：fireflydev生成镜像

上述方式如果行不通的话可以试下如下方式，主要来自：https://wiki.t-firefly.com/zh\_CN/Firefly-RK3399/ubuntu\_manual.html

也就是说如果通过瑞芯微的开发工具导出rootfs及相关分区后不成功的话我们可以直接在我们已经部署好工作环境的设备上通过firefly提供的工具进行以下两步操作：

1、fireflydev导出设备rootfs

2、firefly-linux-repack二次打包完整固件，将rootfs与其它固件组成部分二次打包，生成完整固件

这种方式应该是最简单的，但是可惜需要firefly的设备，目前市面上大多都是基于开源的东西拼凑的，很多类似的工具都没有集成进去，所以很可能你的设备是不支持这种方式，没关系，我们还有第三种方式。

6. 方式三：dd命令直接制作

https://blog.csdn.net/RedKeyer/article/details/100032252

我们的目标一直没变，就是导出rootfs然后打包成可升级的固件，所以外部开发工具以及内部开发工具导出rootfs.img升级都行不通的情况下，我们直接将根文件系统/整个全部命令打包，采用最暴力的方式拷贝根目录后直接制作镜像，如果你的板子空间足够的话(未压缩的情况下Ubuntu18.04的大概4G左右)直接在板子上打包(不推荐)，如果你的板子空间不足的话则可以使用rsync同步设备根目录到远端PC上来打包，这种方式应该是最通用的了。

6.1 配置RK3399设备ssh

默认root用户没有密码，且ssh无法远程登录，这个对部分文件上传有影响，需要修改一下。

$ sudo -s

$ passwd root

Enter new UNIX password:

Retype new UNIX password:

passwd: password updated successfully

1

2

3

4

5

再设置sshd\_config:

sudo vim /etc/ssh/sshd\_config

1

修改其中的Authenation下的PermitRootLogin为yes即可：

...

# Authentication:

#LoginGraceTime 2m

PermitRootLogin yes

#StrictModes yes

#MaxAuthTries 6

#MaxSessions 10

...

1

2

3

4

5

6

7

8

9

然后重启生效：

/etc/init.d/ssh restart

1

6.2 同步根目录到远端PC

# 保证板子root用户可建立ssh链接，PC可远程链接开发板

# rsync -avx root@设备ip:/ 存储目录

rsync -avx root@192.168.1.239:/ ubuntuBoard

1

2

3

6.3 PC上制作rootfs镜像

# 1. dd创建镜像文件，大小按照你的板子的/大小，通过df -h查看，我的是14G，所以ubuntu.img创建了14000

dd if=/dev/zero of=ubuntu.img bs=1M count=14000

# 2. 格式化镜像文件并加入卷标

sudo mkfs.ext4 -F -L linuxroot ubuntu.img

# 3. 挂载镜像并拷贝分区内容

mkdir ubuntu-mount

sudo mount ubuntu.img ubuntu-mount

sudo cp -rfp ubuntuBoard/\* ubuntu-mount

缷载镜像：

sudo umount ubuntu-mount

# 4. e2fsck检查并修复镜像文件

e2fsck -p -f ubuntu.img

# 5. 减小镜像文件大小

resize2fs -M ubuntu.img

1

2

3

4

5

6

7

8

9

10

11

12

13

14

6.4 烧写rootfs

使用5.8版本的瑞芯微开发工具再一次读取分区后烧写rootfs分区：

经测试后发现烧写的镜像是正常的，我配置进去的系统配置和软件开发环境仍然保留了下来（但注意单独烧写rootfs时要包含parameter.txt文件，这个文件需要解压原始镜像获取，否则很可能会出问题）。

6.5 解包及打包update.img后烧写

最近又尝试了第二种方法，其中解包和打包的工具可以使用，结合第二种打包和解包的方式我们可以将原本的镜像解包生成各个img，然后将第三种方法生成的rootfs分区镜像再替换进去，最后再通过第二种方法的打包方法打包成update.img后整体烧写。

6.6 烧写后/dev/root大小不正确的问题解决

烧写后我们通过：

df -h

1

查看分区情况，发现rootfs似乎不太对：

rootfs的大小缩水了，上面显示只有6.2G，而我们通过parted查看理论上我们的根分区大小应该是14G大小才对，通过以下命令查看（建议使用parted，MBR和GPT分区表都能识别，目前由于磁盘容量的扩大，都在逐渐过渡到GPT分区表了）：

lsblk

df -h

parted /dev/mmcblk1

(parted) p

1

2

3

4

莫慌，这个是由于我们的文件系统大小没有设置成分区大小引起的，还记得我们烧写的时候对rootfs.img做的廋身操作吗，在那个位置我们对镜像文件结合内容进行了搜身，所以这里默认显示的是系统的大小，那么我们将其重新设置为分区大小即可，参考这里的命令：

root@bionic:~# resize2fs /dev/mmcblk1p8

resize2fs 1.44.1 (24-Mar-2018)

Filesystem at /dev/mmcblk1p8 is mounted on /; on-line resizing required

old\_desc\_blocks = 1, new\_desc\_blocks = 1

The filesystem on /dev/mmcblk1p8 is now 3670016 (4k) blocks long.

1

2

3

4

5

ok，完成了最后一步：

如果你发现单独烧写rootfs.img后根文件系统的使用为100%，无法上传文件，那么就是少了这一步，我们烧写的根文件系统没有重新调整为整个分区的大小导致没有多余的空间了（制作update.img时也需要注意各个分区设置的大小）。

————————————————

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

原文链接：https://blog.csdn.net/weixin\_39510813/article/details/117824965