京东云AX1800-Pro亚瑟\_AX6600雅典娜USB救砖教程

PS：路由器不像手机，USB可以控制电源输入输出，路由器的USB电源是直接输出的。我多次测试USB救砖，有时还插着USB线启动固件，然后就发现路由器USB的5V不出电压，坏了。USB线接电脑和路由器，两端都输出5V，相当于5V并联，但电源不能简单直接并联使用，会产生反灌电流、输出异常等问题。怀疑反灌电流把路由器USB的5V电源芯片搞坏了，更换之后好了。也许是个例，建议有条件的可以把USB线的5V线剪断（一般是比较粗的红色那根），或者在接口5V的铜片上贴胶带等方式断开5V，地线不用剪，防止反灌电流。

首先明确可以不拆机的情况：

只要变砖后使用双公头USB-A线（有Tpye-C接口的也可以用一头Tpye-C一头USB-A数据线）接路由器和电脑，电脑设备管理器显示QUSB或Qualcomm HS-USB QDLoader 9008的，就可以直接救砖。会导致变砖路由器直接进入高通9008模式的情况：

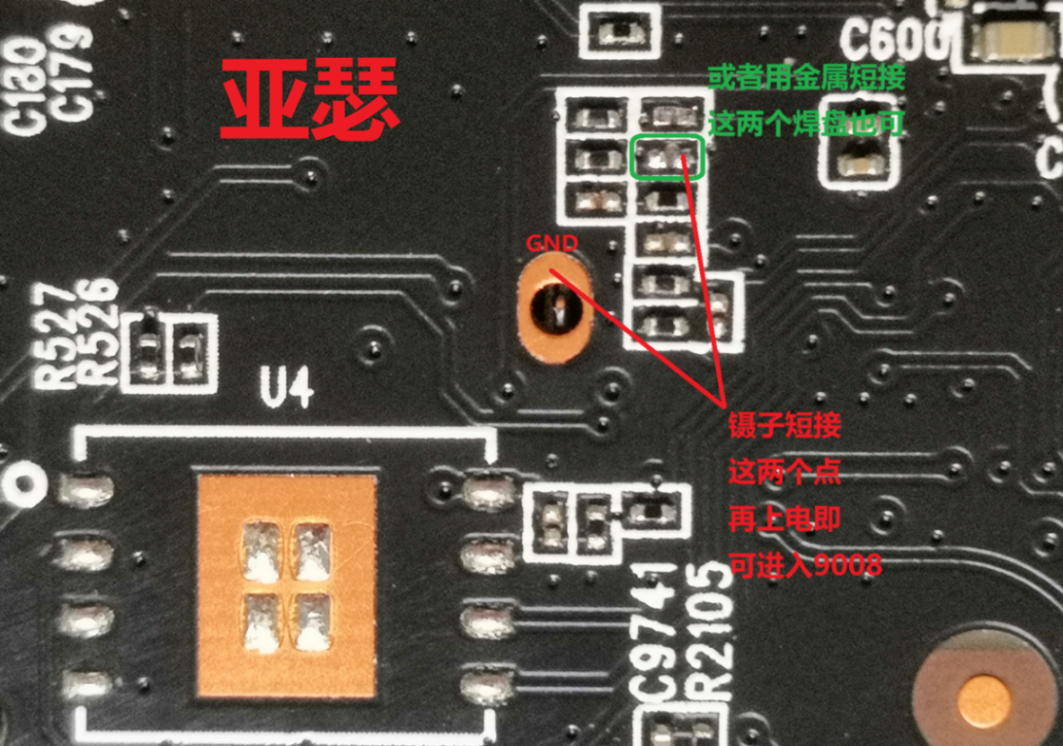
1. 刷坏gpt分区表，导致不能启动SBL1，只擦除gpt的话会使用EMMC最后33个扇区保存的备份gpt启动，不会砖；
2. 擦除或者刷其他文件（非ipq60xx系列SBL1）到SBL1分区，QSEE、DEVCFG、RPM这个分区同理，不过这个几个没什么刷的；
3. 擦除uboot分区（即0:APPSBL分区）或者刷了其他文件（非ipq60xx系列uboot）到该分区；
4. 擦除cdt分区（即0:CDT分区）或者刷了非cdt文件到该分区。

需要拆机短接电阻进入高通9008模式的情况：

1. 刷错了ipq60xx系列uboot（如ax5\_jdc gl-ax1800 uboot）到uboot分区，这种情况会启动SBL1加载uboot，导致不能直接进9008，要拆机用镊子短接启动电阻焊盘，才可以进9008;
2. 刷错了cdt（如ax5\_jdc gl-ax1800 cdt）到cdt分区。

进入高通9008模式除了下面短接启动电阻焊盘，还可以用高通9008工程线，可自制，请自行搜索。原理是短接USB +D数据线和GND，上电开机等四五秒后断开短接，此时相当于正常USB线，机子可直接强制进入9008。这个方法是高通9008通用，不过我没尝试过，需要请自行尝试。

另外一种是短接启动电阻，亚瑟启动电阻焊盘在板子背面，靠近TTL的地方，用镊子短接，然后上电路由器，等两秒再松开即可：



雅典娜启动电阻焊盘在CPU边上，用镊子短接，然后上电路由器，等两秒再松开即可：



运行文件夹中的“打开设备管理器命令.bat”，会打开设备管理器。

USB线一头插路由器，一头插电脑，然后路由器上电。

如正常，此时还没安装驱动，点开设备管理器“其他设备”下有个QUSB设备：



注意：不符合不需拆机情况的，需要短接启动电阻再上电，等两秒再松开短接。

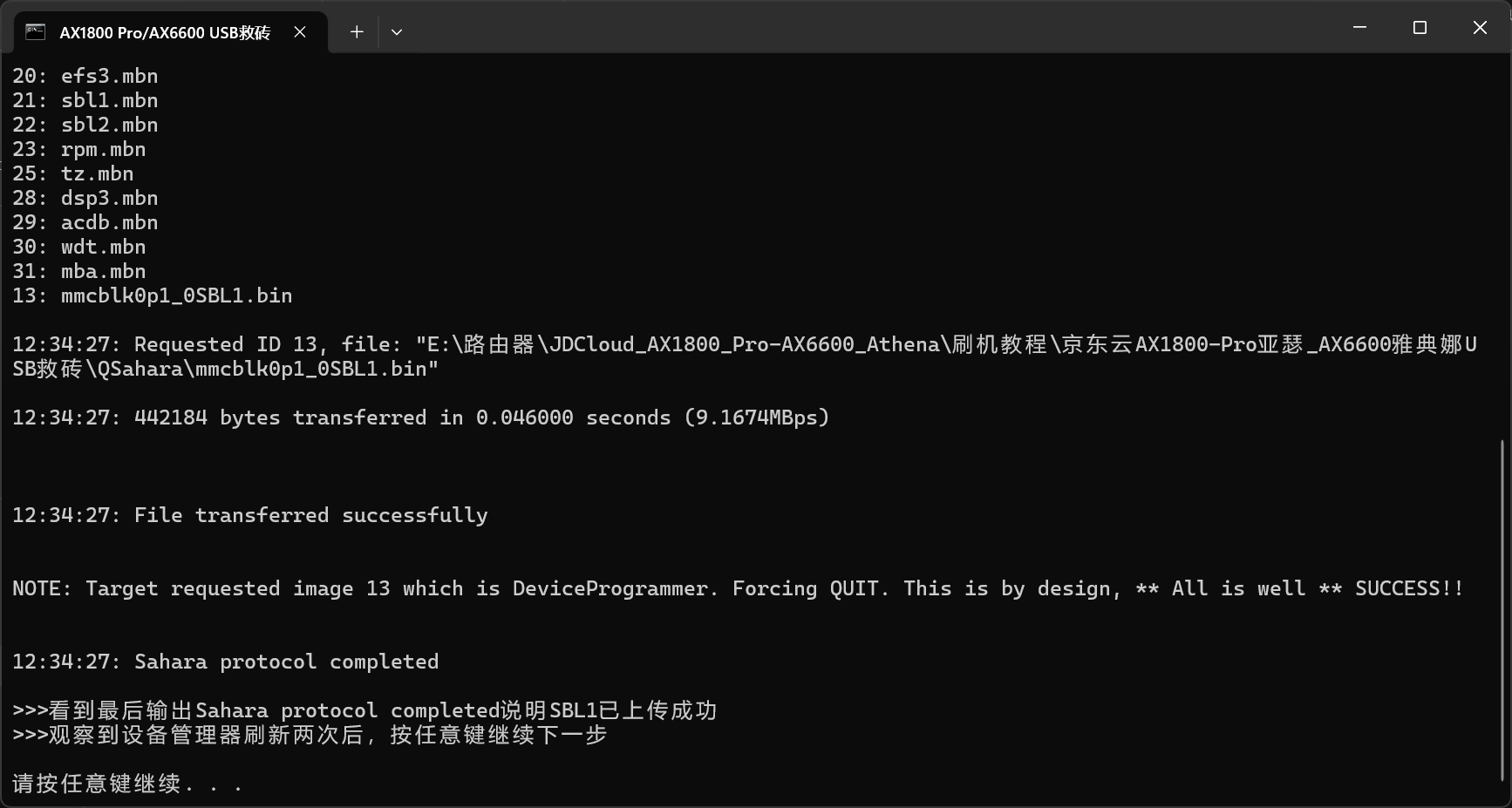
运行文件夹中的Qualcomm USB Driver.exe安装驱动，一路同意允许下一步即可。

安好驱动后，可以在设备管理器“端口(COM和LPT)”看到9008已连接上，记住COM口号，下面要用到。



运行文件夹中个“USB命令.bat”，选择对应的机型后回车，然后输入上面9008对应的COM口号码后回车，命令会开始上传SBL1。

CMD对话框输出如下信息，同时注意观察“设备管理器”那里会刷新两次，刷新两次之后等5秒则路由器已经重新连接了。



如果报错

ERROR: function: main:287 Could not connect to \\.\COM5

或者

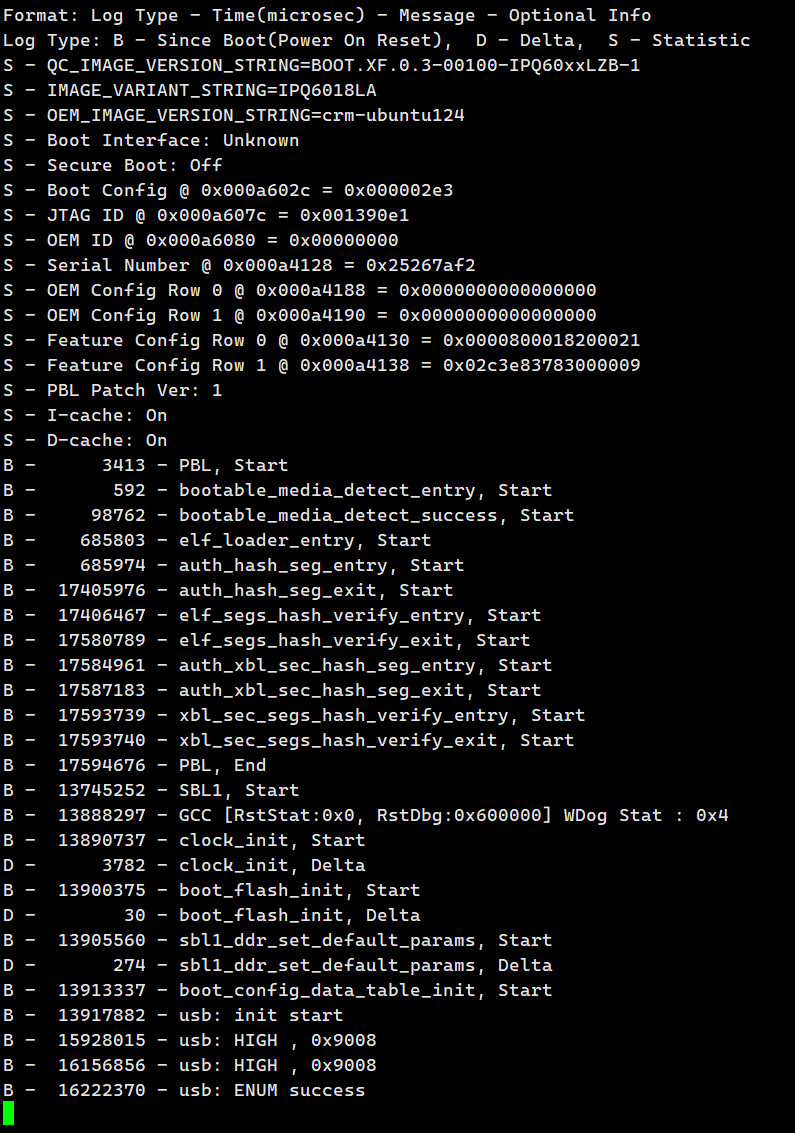
INFO: function: main:273 Port name '\\.\COM5'

EVENT: STATE <-- SAHARA\_WAIT\_HELLO

关闭CMD对话框，检查COM是否被占用，路由器断电，再重新来。

如果正常上传SBL1，你接了TTL可以看到此时输出停在usb: ENUM success。

不过我们不需要TTL，可以直接启动系统或者进uboot webui救砖就行了。

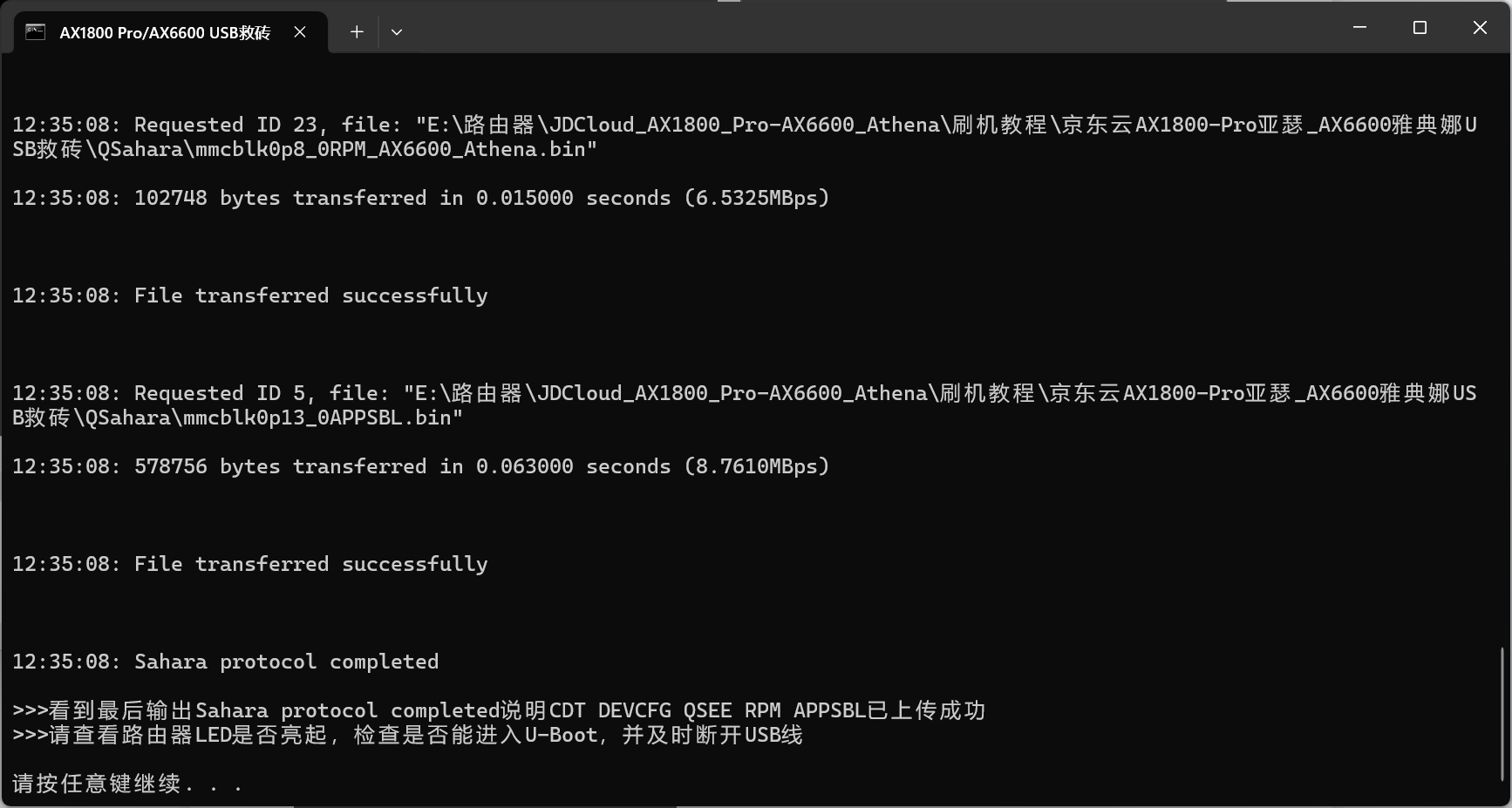


等待5秒后，按任意键继续，如果没问题会输出如下，同时注意看路由器指示灯会亮起，此时马上按住reset按键，等待红灯闪烁，变为蓝灯可以松开reset，这时已经进入uboot webui，可以设置电脑IP 192.168.1.2/24，然后浏览器访问192.168.1.1。建议关闭所有网页，重新打开浏览器去打开192.168.1.1，避免出错。

**uboot正常启动后建议USB线没有断开 5V的，先把USB断开。**

【注意】个别机友反应刷uboot后设置固定ip进不来uboot的webui。这个可能是网卡和uboot的驱动不兼容，此时可尝试将网卡速率的自动协商手动修改为10M全双工，再尝试访问webui。刷好固件后再修改回自动协商。

当然，如果你路由器的系统分区没有坏，建议不按reset按钮，直接加载启动系统，系统启动后再恢复刷砖的分区。



雅典娜原厂是两颗512M DDR4内存，QSahara文件夹的雅典娜CDT是原厂双内存CDT，如果雅典娜改了内存，只用单颗内存，用原厂CDT是启动不了的，可以用文件夹中的单内存cdt-JDC\_AX6600\_Athena\_1024M16\_DDR4\_2G.bin改名替换mmcblk0p10\_0CDT\_AX6600\_Athena.bin后再尝试。或者用自己能用的CDT。

进入uboot webui更新固件：支持kernel为6MB大小的factory.bin和官方原厂固件

http://192.168.1.1

更新ART：

http://192.168.1.1/art.html

更新CDT：上传CDT文件需要大于10KB

http://192.168.1.1/cdt.html

更新IMG：可更新GPT分区表或者EMMC IMG镜像

http://192.168.1.1/img.html

更新U-BOOT：

http://192.168.1.1/uboot.html

注意：上传文件点击更新后，蓝灯会闪烁，然后更新，更新完后出现UPDATE IN PROGRESS页面，同时绿灯会亮3秒，没亮绿灯的是不成功，刷新浏览器192.168.1.1重新上传并更新。

此时如果是uboot刷砖了，此时可直接uboot.html更新uboot即可。

如果是cdt刷砖了，此时可直接cdt.html更新cdt即可。

如果是gpt分区表刷砖了，可在img.html更新gpt分区表文件，然后系统启动后SSH保存下分区表，使用命令echo -e 'w' | fdisk /dev/mmcblk0。如果刷分区表后在系统中用分区表工具新建了分区，也会保存分区表，可以不用上面fdisk保存分区表了。

如果是SBL1、QSEE、DEVCFG或RPM刷砖了，可在img.html上刷对应的EMMC IMG合成镜像。

如RPM分区刷砖了，用备份合成一个GPT到mmcblk0p9\_0RPM\_1.bin的IMG镜像，按下列顺序合成：

mmcblk0\_GPT.bin

mmcblk0p1\_0SBL1.bin

mmcblk0p2\_0BOOTCONFIG.bin

mmcblk0p3\_0BOOTCONFIG1.bin

mmcblk0p4\_0QSEE.bin

mmcblk0p5\_0QSEE\_1.bin

mmcblk0p6\_0DEVCFG.bin

mmcblk0p7\_0DEVCFG\_1.bin

mmcblk0p8\_0RPM.bin

mmcblk0p9\_0RPM\_1.bin

使用合并，或者使用下面命令合并（自行修改文件名，注意文件顺序）。

Linux：

cat file1.bin file2.bin file3.bin > merged\_file.bin

Windows：

copy /B file1.bin + file2.bin + file3.bin merged\_file.bin

其他镜像依次类推。

恢复IMG镜像后，要进入OP恢复自己的各个分区数据，特别是mmcblk0p12\_0APPSBLENV.bin和mmcblk0p15\_0ART.bin分区保存着本机的MAC地址信息，如果被覆盖了，要恢复一下，如果系统没坏，不用刷固件，系统坏了再进uboot刷下固件。

刷分区表建议使用这个命令，fdisk检查分区表错误。最后还是fdisk -l /dev/mmcblk0自己检查下分区表

dd if=/tmp/mmcblk0\_GPT.bin of=/dev/mmcblk0 bs=512 count=34 conv=fsync && echo -e 'w' | fdisk /dev/mmcblk0

恢复其他分区

dd if=/tmp/mmcblk0p1\_0SBL1.bin of=$(blkid -t PARTLABEL=0:SBL1 -o device) conv=fsync

dd if=/tmp/mmcblk0p2\_0BOOTCONFIG.bin of=$(blkid -t PARTLABEL=0:BOOTCONFIG -o device) conv=fsync

dd if=/tmp/mmcblk0p3\_0BOOTCONFIG1.bin of=$(blkid -t PARTLABEL=0:BOOTCONFIG1 -o device) conv=fsync

dd if=/tmp/mmcblk0p4\_0QSEE.bin of=$(blkid -t PARTLABEL=0:QSEE -o device) conv=fsync

dd if=/tmp/mmcblk0p5\_0QSEE\_1.bin of=$(blkid -t PARTLABEL=0:QSEE\_1 -o device) conv=fsync

dd if=/tmp/mmcblk0p6\_0DEVCFG.bin of=$(blkid -t PARTLABEL=0:DEVCFG -o device) conv=fsync

dd if=/tmp/mmcblk0p7\_0DEVCFG\_1.bin of=$(blkid -t PARTLABEL=0:DEVCFG\_1 -o device) conv=fsync

dd if=/tmp/mmcblk0p8\_0RPM.bin of=$(blkid -t PARTLABEL=0:RPM -o device) conv=fsync

dd if=/tmp/mmcblk0p9\_0RPM\_1.bin of=$(blkid -t PARTLABEL=0:RPM\_1 -o device) conv=fsync

dd if=/tmp/mmcblk0p10\_0CDT.bin of=$(blkid -t PARTLABEL=0:CDT -o device) conv=fsync

dd if=/tmp/mmcblk0p11\_0CDT\_1.bin of=$(blkid -t PARTLABEL=0:CDT\_1 -o device) conv=fsync

dd if=/tmp/mmcblk0p12\_0APPSBLENV.bin of=$(blkid -t PARTLABEL=0:APPSBLENV -o device) conv=fsync

dd if=/tmp/mmcblk0p15\_0ART.bin of=$(blkid -t PARTLABEL=0:ART -o device) conv=fsync

QSaharaServer中的img id：

<https://gitee.com/sususususu/Qualcomm-IPQ8074/blob/master/meta-tools/scripts/mbn_tools.py#L202>

class ImageType:

NONE\_IMG = 0

OEM\_SBL\_IMG = 1

AMSS\_IMG = 2

QCSBL\_IMG = 3

HASH\_IMG = 4

APPSBL\_IMG = 5

APPS\_IMG = 6

HOSTDL\_IMG = 7

DSP1\_IMG = 8

FSBL\_IMG = 9

DBL\_IMG = 10

OSBL\_IMG = 11

DSP2\_IMG = 12

EHOSTDL\_IMG = 13 # If the board was bricked (flash does not have a valid booting image)

NANDPRG\_IMG = 14

NORPRG\_IMG = 15

RAMFS1\_IMG = 16

RAMFS2\_IMG = 17

ADSP\_Q5\_IMG = 18

APPS\_KERNEL\_IMG = 19

BACKUP\_RAMFS\_IMG = 20

SBL1\_IMG = 21 # If the boot from USB option is selected

SBL2\_IMG = 22

RPM\_IMG = 23

SBL3\_IMG = 24

TZ\_IMG = 25

PSI\_IMG = 32

<https://github.com/openpst/libopenpst/blob/master/include/qualcomm/mbn.h>

enum MbnImageId {

kMbnImageNone = 0x00, //0

kMbnImageOemSbl = 0x01, //1

kMbnImageAmss = 0x02, //2

kMbnImageOcbl = 0x03, //3

kMbnImageHash = 0x04, //4

kMbnImageAppbl = 0x05, //5

kMbnImageApps = 0x06, //6

kMbnImageHostDl = 0x07, //7

kMbnImageDsp1 = 0x08, //8

kMbnImageFsb = 0x09, //9

kMbnImageDbl = 0x0A, //10

kMbnImageOsb = 0x0B, //11

kMbnImageDsp2 = 0x0C, //12

kMbnImageEhostdl = 0x0D, //13

kMbnImageNandprg = 0x0E, //14

kMbnImageNorprg = 0x0F, //15

kMbnImageRamfs1 = 0x10, //16

kMbnImageRamfs2 = 0x11, //17

kMbnImageAdspQ5 = 0x12, //18

kMbnImageAppsKernel = 0x13, //19

kMbnImageBackupRamfs = 0x14, //20

kMbnImageSbl1 = 0x15, //21

kMbnImageSbl2 = 0x16, //22

kMbnImageRpm = 0x17, //23

kMbnImageSbl3 = 0x18, //24

kMbnImageTz = 0x19, //25

kMbnImageSsdKeys = 0x1A, //26

kMbnImageGen = 0x1B, //27

kMbnImageDsp3 = 0x1C, //28

kMbnImageAcdb = 0x1D, //29

kMbnImageWdt = 0x1E, //30

kMbnImageMba = 0x1F, //31

kMbnImageLast = kMbnImageMba

};

<https://gitee.com/sususususu/Qualcomm-IPQ8074/blob/master/BOOT.BF.3.3.1/boot_images/core/storage/tools/QSaharaServer/src/sahara_protocol.c#L108>

bool init\_sahara\_mapping\_list(void) {

add\_sahara\_mapping(2, "amss.mbn");

add\_sahara\_mapping(6, "apps.mbn");

add\_sahara\_mapping(8, "dsp1.mbn");

add\_sahara\_mapping(10, "dbl.mbn");

add\_sahara\_mapping(11, "osbl.mbn");

add\_sahara\_mapping(12, "dsp2.mbn");

add\_sahara\_mapping(16, "efs1.mbn");

add\_sahara\_mapping(17, "efs2.mbn");

add\_sahara\_mapping(20, "efs3.mbn");

add\_sahara\_mapping(21, "sbl1.mbn");

add\_sahara\_mapping(22, "sbl2.mbn");

add\_sahara\_mapping(23, "rpm.mbn");

add\_sahara\_mapping(25, "tz.mbn");

add\_sahara\_mapping(28, "dsp3.mbn");

add\_sahara\_mapping(29, "acdb.mbn");

add\_sahara\_mapping(30, "wdt.mbn");

return add\_sahara\_mapping(31, "mba.mbn");

};