本文档用于开发记录

Viqtes< [viqtes@126.com](mailto:viqtes@126.com) >

一、烧录固件

1. 官方固件其是两部份，root filesystem, linux

把这两部份在ubuntu的开发环境下进行组合生成新的固件

1. 系统原生的固件包为：

Tegra\_Linux\_Sample-Root-Filesystem\_R32.4.3\_aarch64.tbz2

Tegra186\_Linux\_R32.4.3\_aarch64.tbz2

1. 生成前的准备： 注意一下其依赖的交叉编译支持包

Sudo apt-get install qemu-user-static

1. 整合两部份生成一个固件的步骤：
2. tar xf Tegra186\_Linux\_R32.4.3\_aarch64.tbz2 //解压linux
3. cd ./Linux\_for\_Tegra/rootfs/
4. sudo tar xpf ../../Tegra\_Linux\_Sample-Root-Filesystem\_R32.4.3\_aarch64.tbz2 //把文件 系统解压到linux的rootfs下
5. $: cd ../

$: sudo ./apply\_binaries.sh //打包成新的固件

1. 有如下提示表示完成

Installing extlinux.conf into /boot/extlinux in target rootfs

Success!

1. 烧录固件到设备
2. 机器连接电脑，短路fc\_rec 和 gnd, 把机器上电，其会自动进入recovery模式(烧录 模式)
3. 连上电脑后，在ubuntu下执行lsusb，其能看到jetson的设备：NVidia Corp.
4. 在Linux\_for\_Tegra目录下执行烧录命令：

Sudo ./flash.sh p3449-0000+p3668-0001-qspi-emmc mmcblk0p1 烧录过程20分钟的 样子

1. 模组分为sd版 和 emmc版本，对应烧录的配置文件也不一样，上面的命令是针对 emmc，如果是sd, 其就是

Sudo ./flash.sh p3449-0000+p3668-0000-qspi-sd mmcblk0p1

p3449-0000+p3668-0000-qspi-sd 这个配置在Linux\_for\_Tegra目录下面有，命令其实 就是调用的这个目录下在的配置，但sd的硬件，烧录后，其开机失败了，不知道 是操作有问题，还是对烧录的命令理解有问题。

Emmc版本的硬件测试 OK

1. 系统的编译
2. 编译前的准备工作， 安装必要的包， 配置交叉编译软件，配置环境变量
3. 安安装必要的软件：sudo apt install build-essential bc
4. 准备交叉编译软件，解压 gcc-linaro-7.3.1-2018.05-x86\_64\_aarch64-linux-gnu.xz

xz -d gcc-linaro-7.3.1-2018.05-x86\_64\_aarch64-linux-gnu.xz

tar -xf gcc-linaro-7.3.1-2018.05-x86\_64\_aarch64-linux-gnu.tar

1. 配置环境变量：

export LOCALVERSION=-tegra

export TEGRA\_KERNEL\_OUT=/media/novasky/workspace/jetson/Linux\_for\_Tegra/source/public/kernel/kernel-4.9/out //这里设置绝对路径，用于保存内核编生成的文件

export CROSS\_COMPILE=/media/novasky/workspace/jetson/tools/gcc-linaro-7.3.1-2018.05-x86\_64\_aarch64-linux-gnu/bin/aarch64-linux-gnu- //这里设置的是交叉编译器的地址

这三条一般把其加入用户名文件的.bashrc下面/home/novasky/.bashrc，添加好后，source .bashrc使其生效

d、生成编译输出路径

mkdir -p $TEGRA\_KERNEL\_OUT

1. 执行make defconfig

在内核代码目录执行如下命令 kernel-4.9目录

$ make ARCH=arm64 O=$TEGRA\_KERNEL\_OUT tegra\_defconfig

1. 执行make

$ make ARCH=arm64 O=$TEGRA\_KERNEL\_OUT -j4

1. 如果之前有编译过而出现的编译出现，按提示make mrproper
2. 生成新的固件并烧录
3. 在内核编译输出目录，source/public/kernel/kernel-4.9/out/arch/arm64/boot

把Image copy到固件的kernel目录， 把dts下面的所有文件copy到固件的kernel/dtb目录下面

cp Image /media/novasky/workspace/jetson/Linux\_for\_Tegra/kernel/

cp -r ./dts/\* /media/novasky/workspace/jetson/Linux\_for\_Tegra/kernel/dtb/

1. 把内核 编译生成的module也加载进入固件的rootfs里面，在kernel-4.9根目录执 行如下命令

$ sudo make ARCH=arm64 O=$TEGRA\_KERNEL\_OUT modules\_install \

INSTALL\_MOD\_PATH=/media/novasky/workspace/jetson/Linux\_for\_Tegra/rootfs/

1. 执行脚本，生成新的固件，在固件根目录：Linux\_for\_Tegra

$: sudo ./apply\_binaries.sh //打包成新的固件

1. 在固件根目录运行烧录命令进行下载固件（机器要连上ubuntu）

//烧录固件

sudo ./flash.sh p3449-0000+p3668-0001-qspi-emmc mmcblk0p1

//烧录系统

//sudo ./flash.sh -r -k DTB p3449-0000+p3668-0001-qspi-emmc mmcblk0p1

sudo ./flash.sh -k kernel-dtb p3449-0000+p3668-0001-qspi-emmc mmcblk0p1

<https://blog.csdn.net/jesse_mx/article/details/53315886>

DTB --》 DTS

./[dtc](https://www.baidu.com/s?wd=dtc&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "https://www.cnblogs.com/hutiann/p/_blank) -I dtb -O dts \*.dtb -o \*.dts

DTS -》 DTB

./dtc -I dts -O dtb -o test.dtb test.dts