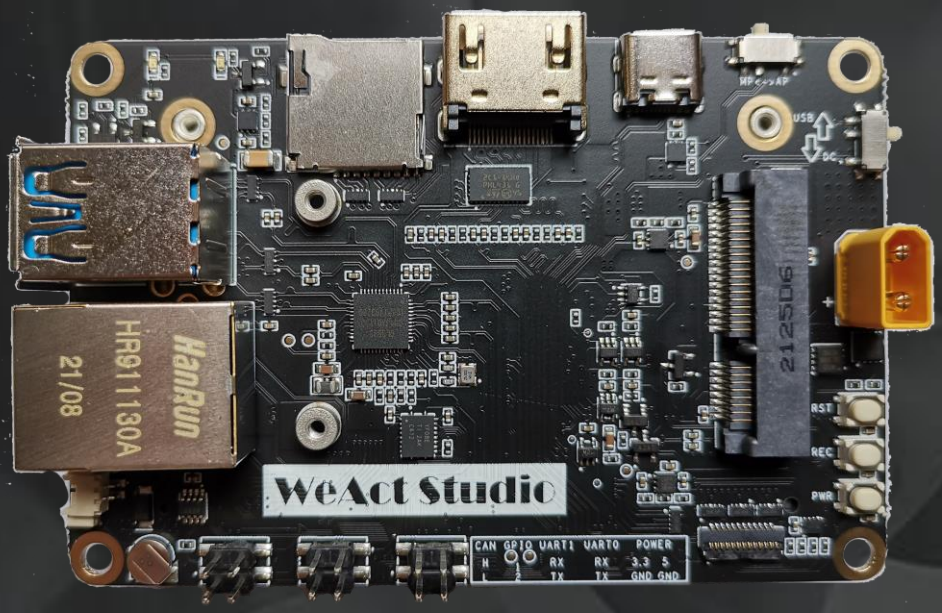


WeAct Studio

NANO&XAVIER TX2 NX 底板

使用教程

WeAct Studio

目录

Revision History 3

1. 搭建烧写环境 4
2. 为 Nano/NX 更新设备树或刷机 7
3. 环境备份及镜像烧写 12
4. 安装 NVIDIA 组件 14
5. 使用 CAN 进行通信 16
6. GPIO&PWM 在 shell 中使用 18
7. 系统迁移至 NVME 固态硬盘 21
8. 系统迁移至 SD 卡 25
9. UART0 开启 DEBUG 信息 28

联系我们 29

REVISION HISTORY

Draft Date	Revision	Description
2021.9.20	V1.0	1. 初始版本
2021.12.26	V1.1	1. 增加系统备份 2. 增加系统迁移至 NVME 固态
2022.1.22	V1.2	1. 增加系统迁移至 SD 卡
2020.04.10	V1.3	1. 增加控制 GPIO 输出 PWM 2. 增加 UART0 输出 debug 信息

1. 搭建烧写环境

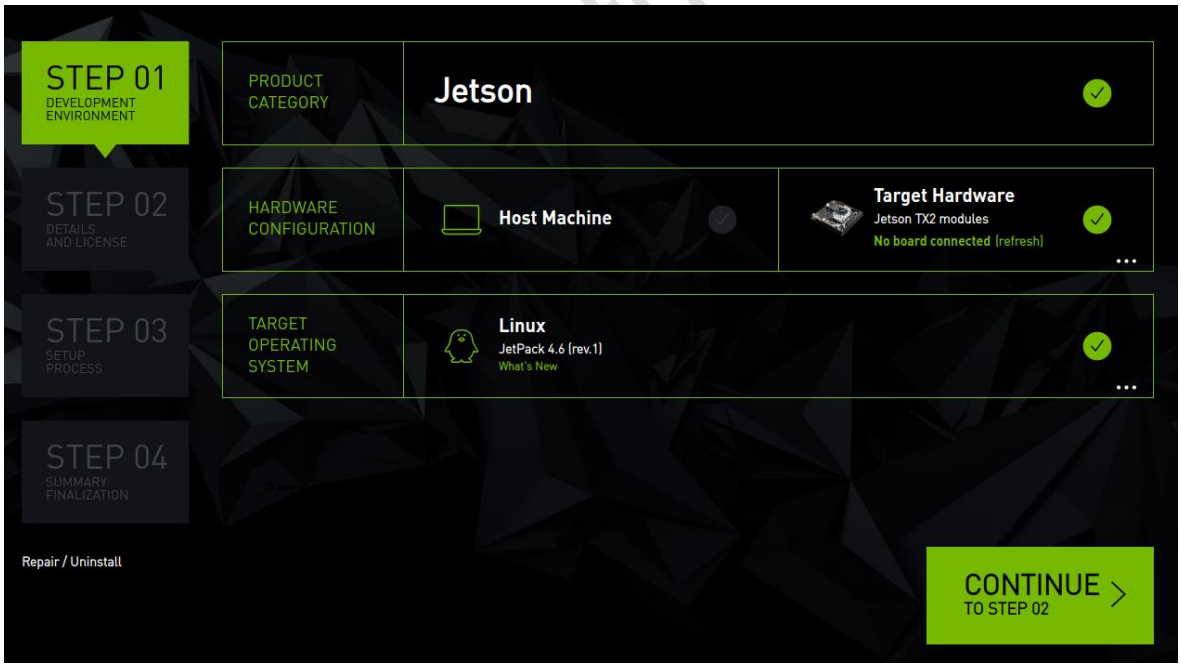
a) 首先，需要一台装有 **Ubuntu16.04** 以上的电脑作为 HOST 端给 Nano/NX 烧写，或者可以在 Windows 上安装 VMware 来实现。

- VMware 上如何安装 Ubuntu18.04：
  https://blog.csdn.net/u012556114/article/details/82751089

b) 在 NVIDIA 下载最新的 **SDK-Manager** 并在 ubuntu18.04 中安装（需要注册一个 NVIDIA 账号，后面也需要用到）

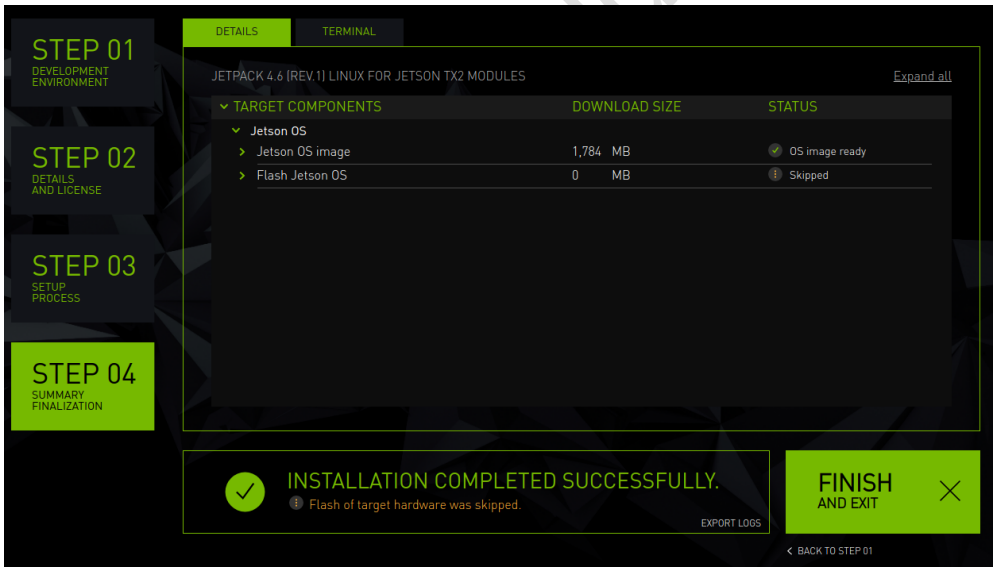
- SDK-Manager 下载地址：https://developer.nvidia.com/nvidia-sdk-manager

c) 选择需要 **Target Hardware** 以及 **JetPack** 版本，**不勾选 HostMachine**，这里以 **TX2NX 为例**选择，点击 **Continue**

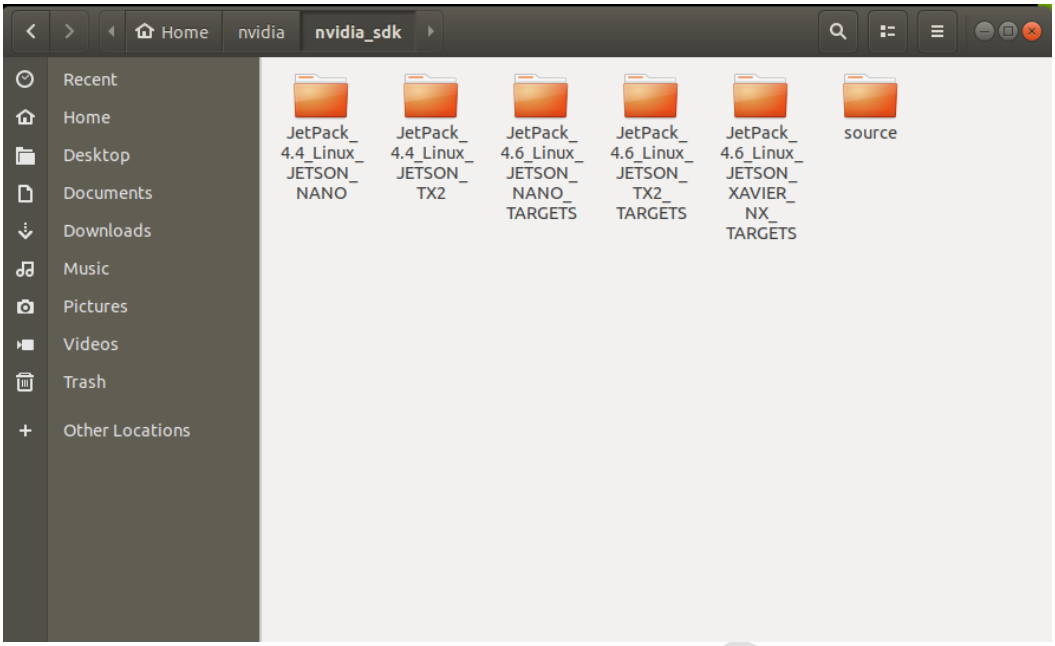


d) 这里需要勾选 **I accept the terms and conditions of the license agreements**，**取消勾选 Jetson SDK Components**，点击 CONTINUE 进行下一步。

e) P.S：请在畅通的网络环境下进行下载以及安装，下载或安装失败时，可点击 **Retry** 继续，直至**全部状态为 Installed 并且显示绿色**，安装过程中会弹出联网烧写的信息，选择**Skip。**



f) 安装成功后，会在**~/nvidia/nvidia_sdk/** 下有相应版本烧写所需的文件



g) 在终端通过 **sudo apt-get install python** 安装 python 支持以便后续烧写环境。

2. 为 NANO/NX 更新设备树或刷机

P.S: WeAct 设备树与官方设备树区别（**其他功能相同**），如果无需求，可以不更新设备树。
！！！ 注意，更新设备树不影响系统任何文件，请放心更新

NVIDIA 与 WeAct 设备树差异

	NVIDIA	WeAct Studio
Nano-SD	相同	相同
Nano-EMMC	无法使用 SD 卡	可以使用 SD 卡
TX2NX	无法使用 **SD 卡&UART1**	可以使用 **SD 卡&UART1**
XavierNX	无法使用 **SD 卡**	可以使用 **SD 卡**

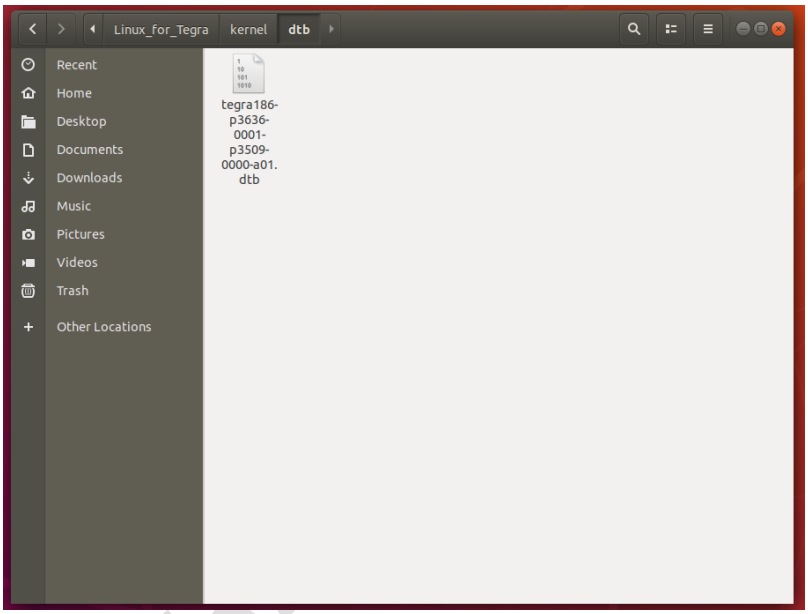
a) 这里**以 TX2NX 为例**，在 WeAct Studio 的 **github** 或者**码云**上下载相应的设备树文件。
- Github：https://github.com/WeActTC/Nano_TX2-Xavier_NX-CB
- 码云：https://gitee.com/WeAct-TC/Nano_TX2-Xavier_NX-CB

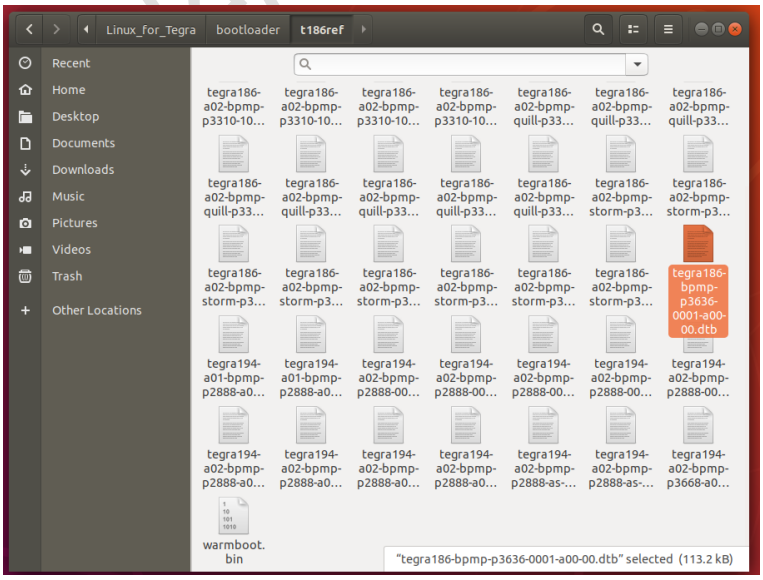
各设备设备树更新路径及设备树名称

	Linux_for_Tegra/kernel/dtb	**Linux_for_Tegra/bootloader/t186ref(t210f)**
Nano-EMMC	**tegra210-p3448-0002-p3449-0000-b00**	**无**
TX2NX	**tegra186-p3636-0001-p3509-0000-a01**	**tegra186-bpmp-p3636-0001-a00-00**
XavierNX	**tegra194-p3668-all-p3509-0000**	**无**

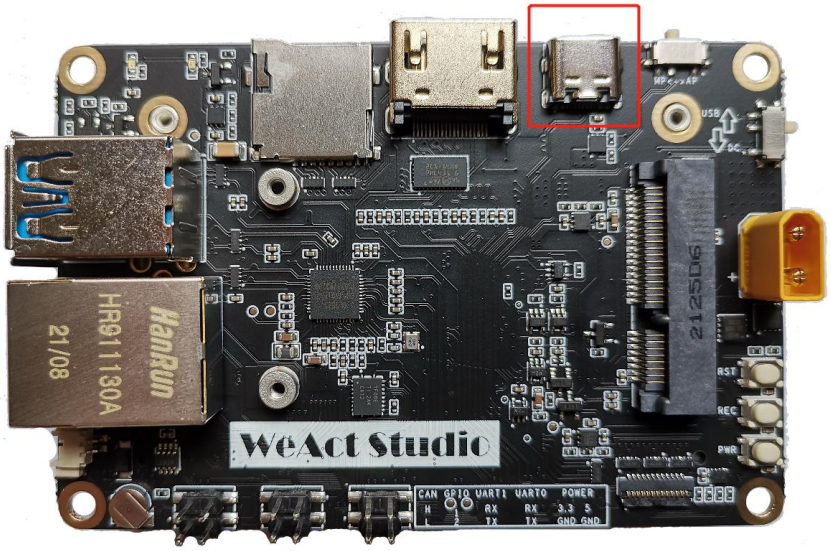
b) 找到相应版本的设备树

1. 进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra/kernel/dtb**，复制提供的设备树 **tegra186-p3636-0001-p3509-0000-a01.dtb** 至该目录
2. 进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra/bootloader/t186ref**，复制提供的设备树 **tegra186-bpmp-p3636-0001-a00-00.dtb** 至该目录【**仅 TX2NX 需要更新**】

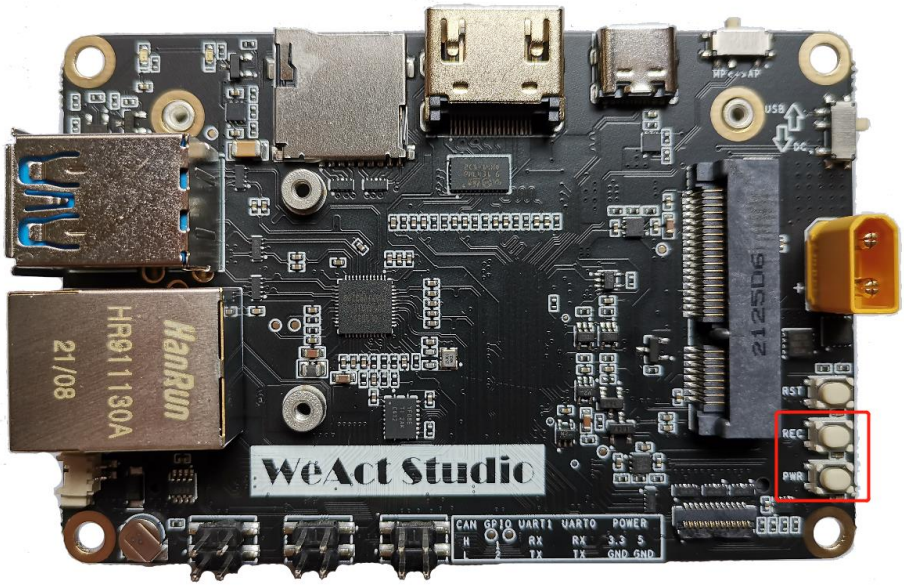




3. 使用 **USB Type-C** 线连接载板上的 **USB OTG** 接口 。



4. 将开机键**拨至 MP（手动开机）**，摁住 **REC** 键，再摁 **PWR** 键开机，松开 **REC** 键进入 Recovery 模式，此时 VMWare 右下角会出现 **NVIDIA 的 USB 驱动标志**，或者打开终端，输入 **lsusb** 命令，会发现 **Nvidia Corp**。



5. 进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**，打开终端：
   a) **如果你没有系统，需要刷机**，请使用刷机命令 **sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**
   b) **如果你有系统，只需要更新设备树**，请使用更新设备树命令 **sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**

   等更新成功就可以使用了，其他设备命令请参考下面表格。

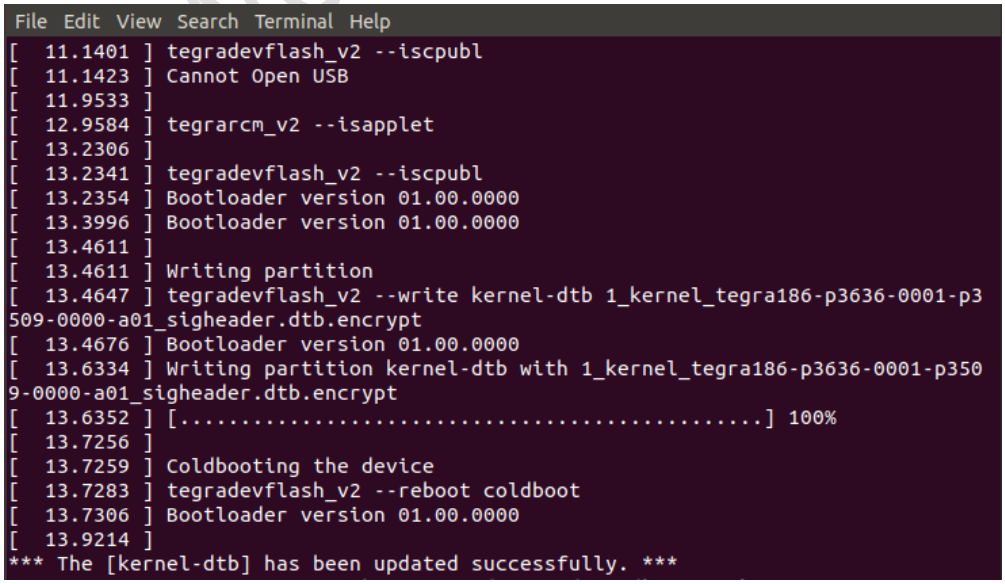
各设备更新设备树命令

设备	设备树更新命令
Nano-SD	sudo ./flash.sh -r -k DTB jetson-nano-qspi-sd mmcblk0p1
Nano-EMMC	sudo ./flash.sh -r -k DTB jetson-nano-emmc mmcblk0p1
TX2-NX	sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-tx2-nx mmcblk0p1
Xavier-SD	sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-qspi mmcblk0p1
Xavier-EMMC	sudo ./flash.sh -r -k kernel-dtb jetson-xavier-nx-devkit-emmc mmcblk0p1

各设备刷机命令

设备	刷机命令
Nano-SD	sudo ./flash.sh jetson-nano-qspi-sd mmcblk0p1
Nano-EMMC	sudo ./flash.sh jetson-nano-emmc mmcblk0p1
TX2-NX	sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx mmcblk0p1
Xavier-SD	sudo ./flash.sh jetson-xavier-nx-devkit-qspi mmcblk0p1
Xavier-EMMC	sudo ./flash.sh jetson-xavier-nx-devkit-emmc mmcblk0p1

更新设备树/刷机后，会有 Successfully!显示，如下图所示。



3. 环境备份及镜像烧写

a) 参考**第 2 章**，无论备份还是镜像烧写，进入 Recovery 模式，注意镜像较大，请保证 Ubuntu 有充足的空间（>40G）。

b) **备份**：这里以 TX2NX 为例（其他设备参考上章内容修改 jetson 名称），对核心板现有环境进行备份。进入~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra，打开终端：

使用镜像备份命令：**sudo ./flash.sh -r -k APP -G backup.img jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**，等待备份完成即可，此时目录下会有 backup.img 的镜像（建议复制一份至其他位置备份），此时**备份已经成功**。

```
greetrix@greetrix-virtual-machine:~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra$ sudo ./flash.sh -r -k APP -G backup.img jetson-xavier-nx-devkit-tx2-nx mmcblk0p1
```

```
[    9.1920 ] tegrarcm_v2 --boot recovery
[    9.1966 ] Applet version 01.00.0000
[    9.3692 ]
[   10.3763 ] tegrarcm_v2 --isapplet
[   10.3793 ] USB communication failed.Check if device is in recovery
[   10.5068 ]
[   10.8536 ] tegradevflash_v2 --iscpubl
[   10.8565 ] Cannot Open USB
[   11.3572 ]
[   12.3617 ] tegrarcm_v2 --isapplet
[   12.5109 ]
[   12.5142 ] tegradevflash_v2 --iscpubl
[   12.5163 ] Bootloader version 01.00.0000
[   12.6843 ] Bootloader version 01.00.0000
[   12.7463 ]
[   12.7464 ] Reading partition
[   12.7492 ] tegradevflash_v2 --read APP /home/greetrix/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra/backup.img
[   12.7511 ] Bootloader version 01.00.0000
[   12.9183 ] [.............................................] 100%
[ 2216.5426 ]
*** The [APP] has been read successfully. ***
        Converting RAW image to Sparse image... greetrix@greetrix-virtual-machin
X2_TARGETS/Linux_for_Tegra$ k 4.6_Linux_JETSON_TX
```

c) **镜像烧写**：进入~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra，将备份好的 backup.img 拷入 Liunx_for_Tegra/bootloader/下，并重命名为 system.img，回到 Linux_for_Tegra 目录下，打开终端：

使用已有镜像烧写命令：**sudo ./flash.sh -r jetson-xavier-nx-devkit-tx2-nx mmcblk0p1**，等待烧写完成即可。

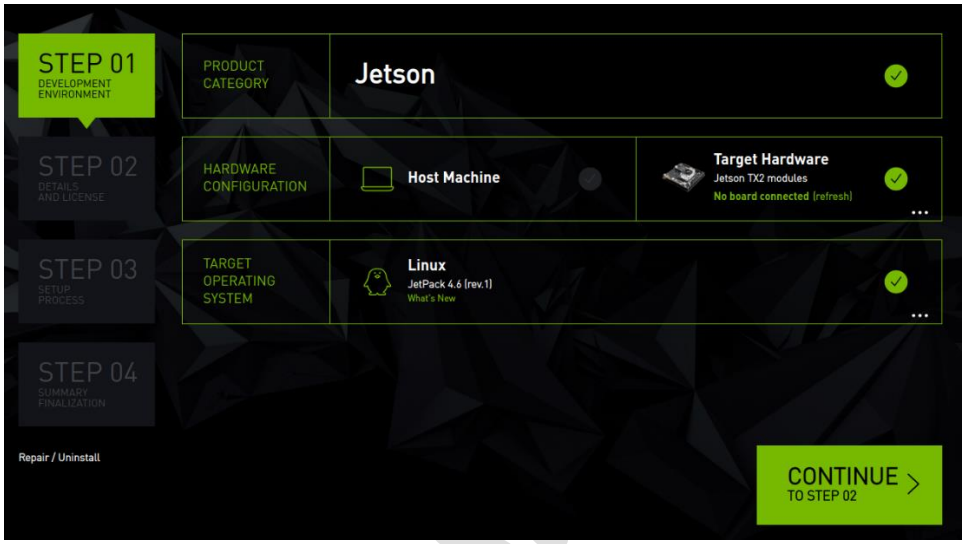
```
[   18.0000 ] Writing partition spe-fw_b with spe_sigheader.bin.encrypt
[   18.0298 ] [.................................................] 100%
[   18.0790 ] Writing partition mb2 with nvtboot_sigheader.bin.encrypt
[   18.1057 ] [.................................................] 100%
[   18.1596 ] Writing partition mb2_b with nvtboot_sigheader.bin.encrypt
[   18.1895 ] [.................................................] 100%
[   18.2416 ] Writing partition mts-preboot with preboot_d15_prod_cr_sigheader.bi
n.encrypt
[   18.2710 ] [.................................................] 100%
[   18.6760 ] Writing partition mts-preboot_b with preboot_d15_prod_cr_sigheader.
bin.encrypt
[   18.7053 ] [.................................................] 100%
[   18.7467 ] Writing partition SMD with slot_metadata.bin
[   18.7744 ] [.................................................] 100%
[   18.9037 ] Writing partition SMD_b with slot_metadata.bin
[   18.9302 ] [.................................................] 100%
[   18.9658 ] Writing partition VER_b with emmc_bootblob_ver.txt
[   18.9922 ] [.................................................] 100%
[   19.0322 ] Writing partition VER with emmc_bootblob_ver.txt
[   19.0592 ] [.................................................] 100%
[   19.0966 ] Writing partition master_boot_record with mbr_1_3.bin
[   19.1194 ] [.................................................] 100%
[   19.1525 ] Writing partition APP with system.img
[   19.1800 ] [........                                        ] 016%
```

```
t.encrypt
[ 1888.6372 ] Bootloader version 01.00.0000
[ 1888.8013 ] Writing partition MB1_BCT with mb1_cold_boot_bct_MB1_sigheader.b
.encrypt
[ 1888.8019 ] [.................................................] 100%
[ 1888.8706 ]
[ 1888.8837 ] tegradevflash_v2 --write MB1_BCT_b mb1_cold_boot_bct_MB1_sigheade
.bct.encrypt
[ 1888.8849 ] Bootloader version 01.00.0000
[ 1889.0452 ] Writing partition MB1_BCT_b with mb1_cold_boot_bct_MB1_sigheader
t.encrypt
[ 1889.0468 ] [.................................................] 100%
[ 1889.1180 ]
[ 1889.1181 ] Flashing completed

[ 1889.1181 ] Coldbooting the device
[ 1889.1436 ] tegradevflash_v2 --reboot coldboot
[ 1889.1449 ] Bootloader version 01.00.0000
[ 1889.3379 ]
*** The target t186ref has been flashed successfully. ***
Reset the board to boot from internal eMMC.
```

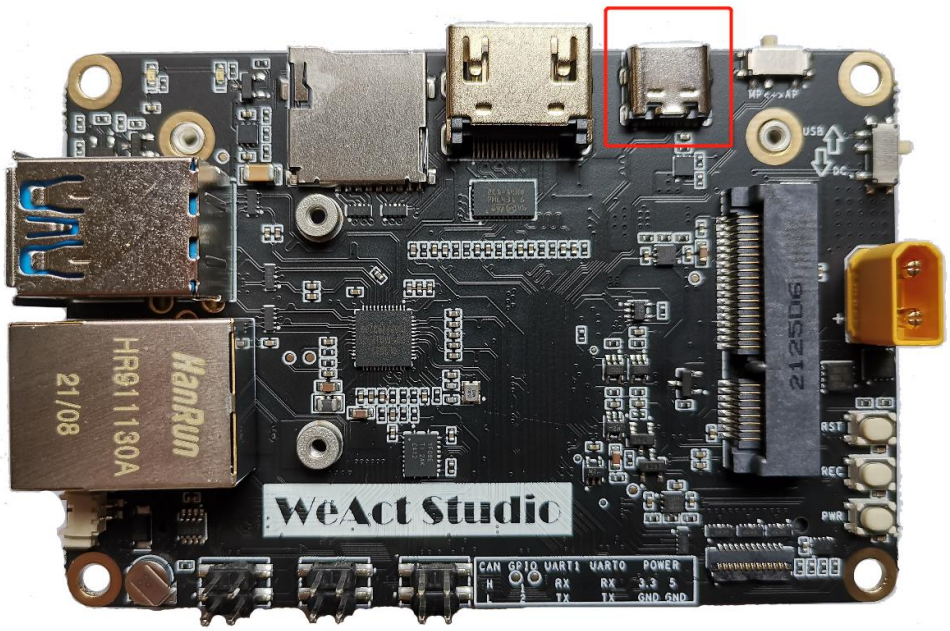
4. 安装 NVIDIA 组件

d) 选择需要 **Target Hardware** 以及 **JetPack** 版本，**不勾选 HostMachine**，这里以 **TX2NX 为例**选择，点击 **Continue**



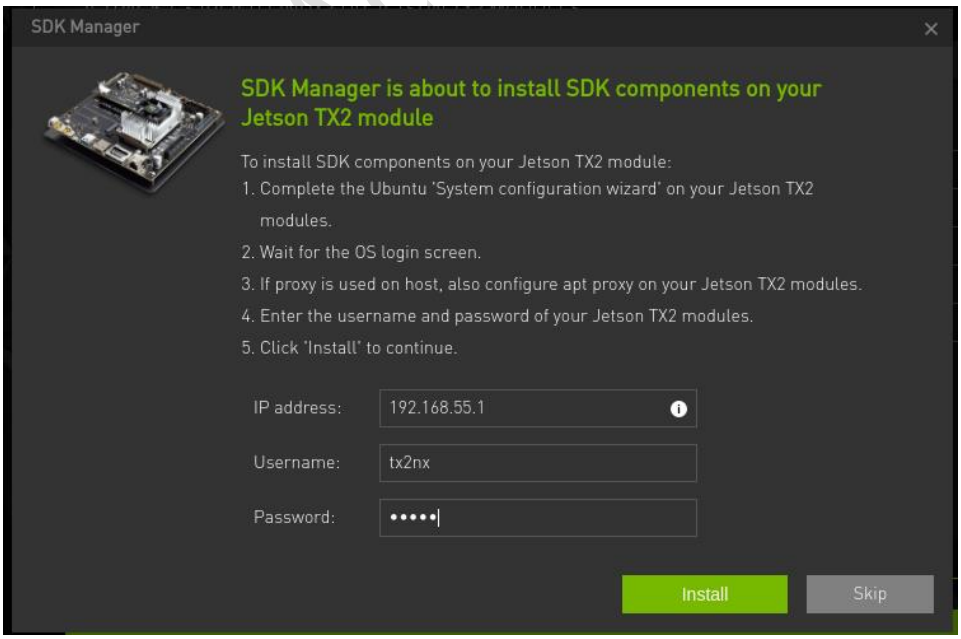
e) 勾选所需要 **SDK 组件**，勾选 **I accept the terms and conditions of the license agreements**，点击 **CONTINUE** 进行下一步。

f) 使用 **USB Type-C** 线连接载板上的 **USB OTG** 接口 。



g) 将开机键**拨至 MP（手动开机）**，摁 **PWR** 键开机，此时 VMWare 右下角会出现 **NVIDIA 的 USB 驱动标志**，或者打开终端，输入 **lsusb** 命令，会发现 **Nvidia Corp**。

h) 输入 **TX2NX** 账号密码，**TX2NX** 端请**保持联网状态**



i) 等待安装完成即可。

5. 使用 CAN 进行通信

a) Tx2-NX/XavierNX 上集成了 2 个 CAN 控制器（CAN0/CAN1），另外 WeAct Studio 的载板上设计了 1 个 CAN 收发器（CAN0），可直接挂载 CAN 物理总线使用。

b) Tx2-NX/XavierNX 自带 canbus 的驱动并集成到了镜像中，已经支持 canbus 无需多做处理。我们需要安装 canbus 模块。（在终端输入下面命令或者放入 rc.local 里面开启自启）

```
modprobe can        // 插入 can 总线子系统
modprobe can-raw    //插入 can 协议模块
modprobe can-bcm
modprobe can-gw
modprobe can_dev
modprobe mttcan     //真正的 can 口支持
```

c) 通过 **lsmod** 检查是否安装成功。

```
nvidia@localhost:~$ lsmod
Module                  Size  Used by
fuse                  103841  2
mttcan                 66251  0
can_dev                13306  1 mttcan
can_gw                 10919  0
can_bcm                16471  0
can_raw                10388  0
can                    46600  3 can_raw,can_bcm,can_gw
zram                   26166  6
overlay                48691  0
bcmdhd                934274  0
cfg80211              589351  1 bcmdhd
spidev                 13282  0
nvgpu                1575721  20
bluedroid_pm           13912  0
ip_tables              19441  0
x_tables               28951  1 ip_tables
```

d) 配置 canbus 属性，和串口的波特率设置类似。

sudo ip link set can0 type can bitrate 500000
sudo ip link set up can0

e) 通过 ifconfig 查看是否配置成功。

```
nvidia@localhost:~$ ifconfig
can0: flags=193<UP,RUNNING,NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 131
```

f) 在一个终端通过 cansend can0(can1) ×××命令来发送数据, 另一个终端通过 candump can1(can0)完成实际信号收发测试

```
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$ cansend can0 555#112233445566
nvidia@localhost:~$
```

```
nvidia@linux:~$ candump can1
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
  can1  555   [6]  11 22 33 44 55 66
```

6. GPIO&PWM 在 SHELL 中使用

a) Nano/TX2-NX/Xavier-NX 可直接通过 shell 命令控制 GPIO 输入输出

	GPIO1	GPIO2
Nano	194	38
TX2-NX	338	269
Xavier-NX	196	105

b) **输出 GPIO** - 以 TX2-NX GPIO1 为例

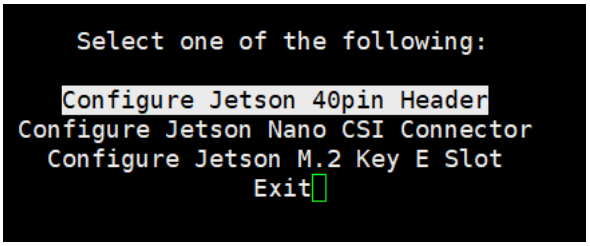
- 先激活 IO: sudo echo 338 > /sys/class/gpio/export
- 设置 IO 方向: echo out > /sys/class/gpio/gpio338/direction
- 设置输出: echo 1 > /sys/class/gpio/gpio338/value

c) 输出 PWM - 以 XavierNX GPIO1 为例

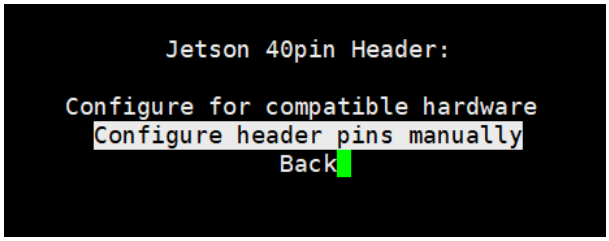
	GPIO1	GPIO2
Xavier-NX	GP_PWM4/0xc340000	GP_PWM1/0x3280000

i. 命令：**cd ~** , 进入 home 目录

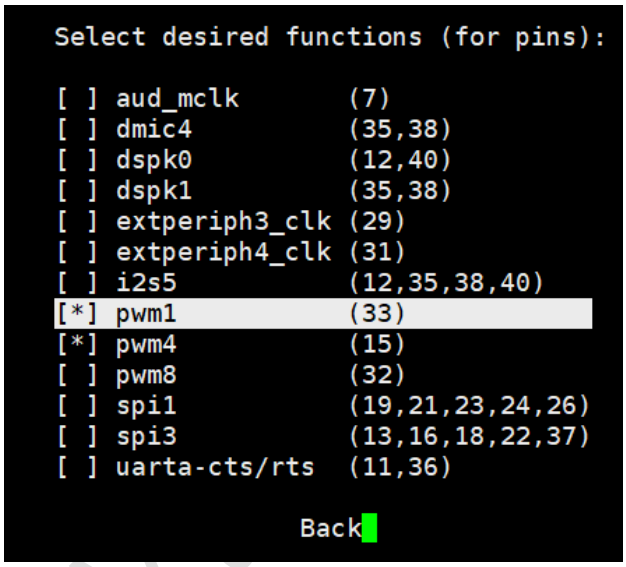
ii. 命令：**sudo /opt/nvidia/jetson-io/jetson-io.py**, 配置 IO



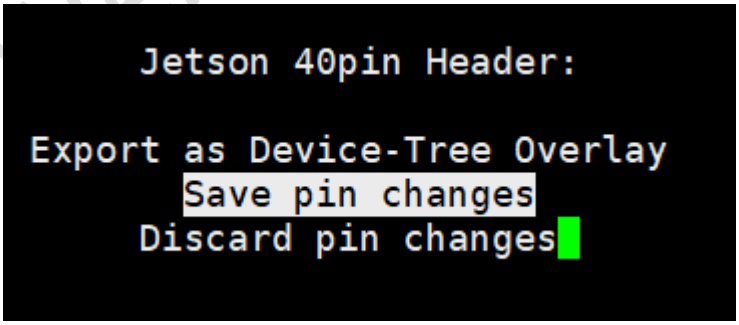
iii. 通过方向键，选择 **Configure Jetson 40Pin Header** 配置 IO



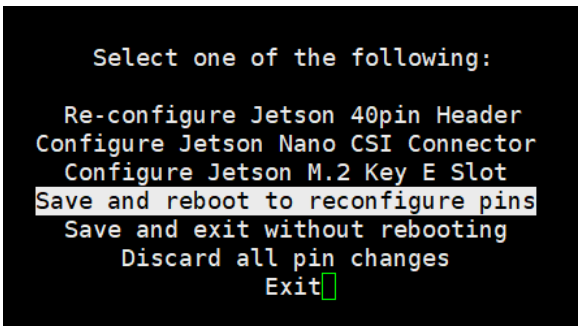
iv. 通过方向键，选择 **Configure header pins manually**



v. **通过方向键移动**至 **pwm1**(GPIO2)、**pwm4**（GPIO1），**回车**勾选（变成*号）



vi. 选择 **Save pin changes**，保存



vii. 选择 **Save and reboot to reconfigure pins**， 重启

viii. 如何使用 **GPIO1**、**GPIO2** 输出 PWM？下面以 GPIO1 为例：

ix. 命令：**cd /sys/devices/c340000.pwm/pwm/pwmchip1**（GPIO2 为：**.../3280000.pwm/pwm/pwmchip0**）

x. 命令：**echo 0 > export**

xi. 命令：**echo 20000000 > period && echo 2000000 > duty_cycle**，配置 PWM 频率及占空比（单位均为 ns，该波形为 50Hz，占空比 10%）

xii. 命令：**echo 1 > enable**，输出波形

7. 系统迁移至 NVME 固态硬盘

a) WeAct-Nano&Xavier-TX2_NX-CB 搭配 WeAct-MiniPCIE2M2 转板，支持 2242/2230 NVME SSD 固态硬盘，最大可达 300M/s 读写速度。

```
tx2nx@tx2nx:/mnt/ssd$ dd if=/dev/zero of=./largefile bs=1M count=1024
dd: failed to open './largefile': Permission denied
tx2nx@tx2nx:/mnt/ssd$ sudo dd if=/dev/zero of=./largefile bs=1M count=1024
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 3.14822 s, 341 MB/s
tx2nx@tx2nx:/mnt/ssd$ sudo sh -c "sync && echo 3 > /proc/sys/vm/drop_caches"
tx2nx@tx2nx:/mnt/ssd$ dd if=./largefile of=/dev/null bs=4k
262144+0 records in
262144+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 3.08641 s, 348 MB/s
tx2nx@tx2nx:/mnt/ssd$
```

b) NVME 固态硬盘配置：

- 1. 配置前确保系统能识别到 NVME 固态硬盘，终端命令：**sudo fdisk -lu**

```
Disk /dev/nvme0n1: 119.2 GiB, 128035676160 bytes, 250069680 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
```

- 2. 将 NVME 设置成 GPT 格式：

i. 终端命令：**sudo parted /dev/nvme0n1** 进入 parted

```
tx2nx@tx2nx:~$ sudo parted /dev/nvme0n1
[sudo] password for tx2nx:
GNU Parted 3.2
Using /dev/nvme0n1
Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted)
```

ii. 终端命令：**mklabel gpt** 将磁盘 label 设置为 gpt 格式

```
(parted) mklabel gpt
Warning: The existing disk label on /dev/nvme0n1 will be destroyed and all data on this disk will be lost. Do you want to
Yes/No? Yes
```

iii. 终端命令：**mkpart logical 0 -1** 将磁盘 part 设置为 gpt 格式

```
(parted) mkpart logic 0 -1
Warning: The resulting partition is not properly aligned for best performance.
Ignore/Cancel? Ignore
```

iv. 终端命令：**print** 查看分区结果

```
(parted) print
Model: KBG40ZNS128G NVMe TOSHIBA 128GB (nvme)
Disk /dev/nvme0n1: 128GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End    Size   File system  Name   Flags
 1      17.4kB  128GB  128GB               logic
```

v. 终端命令：**quit** 退出

vi. 终端命令：**sudo fdisk /dev/nvme0n1**

```
(parted) quit
Information: You may need to update /etc/fstab.

tx2nx@tx2nx:~$ sudo fdisk /dev/nvme0n1

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help):
```

vii. Command（m for help): 输入 **N**，选择增加新分区，后面回车默认即可

```
Command (m for help): n
Partition number (2-128, default 2): 2
First sector (250067728-250069646, default 250068992):
Last sector, +sectors or +size{K,M,G,T,P} (250068992-250069646, default 250069646):

Created a new partition 2 of type 'Linux filesystem' and of size 327.5 KiB.
```

viii. Command（m for help)：输入 **P**，查看分区结果

```
Device             Start       End   Sectors   Size Type
/dev/nvme0n1p1        34 250067727 250067694 119.2G Linux filesystem
/dev/nvme0n1p2 250068992 250069646       655 327.5K Linux filesystem
```

ix. 终端命令：**quit** 退出

x. 终端命令：**sudo mke2fs -t ext4 /dev/nvme0n1p1**，格式化分区

```
tx2nx@tx2nx:~$ sudo mke2fs -t ext4 /dev/nvme0n1p1
mke2fs 1.44.1 (24-Mar-2018)
/dev/nvme0n1p1 contains a ext4 file system
        last mounted on / on Sun Dec 26 11:04:07 2021
Proceed anyway? (y,N)
```

xi. 终端命令：**sudo mount /dev/nvme0n1p1/mnt**，成功 mount 则 NVME 配置成功

```
tx2nx@tx2nx:~$ sudo mount /dev/nvme0n1p1 /mnt
tx2nx@tx2nx:~$
```

c) **NVIDIA Jetson 系统迁移 (!!!迁移前建议参考第 3 章进行系统备份)：**

- ✓ **下面以 TX2NX 为例，其他设备替换命令中间的设备名称即可，设备名称可参考上面命令**
- ➢ 1. 终端命令： **git clone https://github.com/jetsonhacks/rootOnNVMe** 或者 https://github.com/jetsonhacks/rootOnNVMe 下载脚本
- ➢ 2. 进入 **rootOnNVMe** 文件夹，终端命令：**./copy-rootfs-ssd.sh**，复制系统文件至 NVME SSD

```
tx2nx@tx2nx:/home/script/rootOnNVMe-master$ ./copy-rootfs-ssd.sh
mount: /mnt: /dev/nvme0n1p1 already mounted on /mnt.
     17,380,838   0%    2.40MB/s    0:00:06 (xfr#39, ir-chk=1015/44887)
```

- ➢ 3. 终端命令：**./setup-service.sh** 配置启动项

```
tx2nx@tx2nx:/home/script/rootOnNVMe-master$ ./setup-service.sh
==== AUTHENTICATING FOR org.freedesktop.systemd1.reload-daemon ===
Authentication is required to reload the systemd state.
Authenticating as: tx2nx,,, (tx2nx)
Password: Failed to reload daemon: Method call timed out
polkit-agent-helper-1: pam_authenticate failed: Authentication failure
Created symlink /etc/systemd/system/default.target.wants/setssdroot.service → /etc/systemd/system/setssdroot.service.
Service to set the rootfs to the SSD installed.
Make sure that you have copied the rootfs to SSD.
Reboot for changes to take effect.
```

- ➢ 4. 参考第 2 章，进入 **Recovery 模式。**
- ➢ 5. （**烧录环境的 Ubuntu**，参考**前面章节**）进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**，打开终端：**sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx nvme0n1p1** 更新 EMMC 内部引导

```
greetrix@greetrix-virtual-machine:~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra$ sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx nvme0n1p1
```

```
[   37.3739 ] Coldbooting the device
[   37.3775 ] tegradevflash_v2 --reboot coldboot
[   37.3788 ] Bootloader version 01.00.0000
[   37.5711 ]
*** The target t186ref has been flashed successfully. ***
Make the target filesystem available to the device and reset the board to boot from external nvme0n1p1.
```

- 6. 重启 TX2NX，终端命令：**df -l**， 此时系统盘已经变为 NVME SSD，并且原有 EMMC 上系统已经成功迁移。

```
tx2nx@tx2nx:~$ df -l
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/nvme0n1p1 122547172 11949920 104329176  11% /
none             1578060        0   1578060   0% /dev
tmpfs            1962748       52   1962696   1% /dev/shm
tmpfs            1962748    20764   1941984   2% /run
tmpfs               5120        4      5116   1% /run/lock
tmpfs            1962748        0   1962748   0% /sys/fs/cgroup
tmpfs             392548       12    392536   1% /run/user/120
tmpfs             392548        0    392548   0% /run/user/1000
```

8. 系统迁移至 SD 卡

a) SD 卡配置：

1. 配置前确保系统能识别到 SD 卡，终端命令：**sudo fdisk -lu**

```
Disk /dev/mmcblk1: 59.5 GiB, 63864569856 bytes, 124735488 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

2. 将 SD 卡设置成 GPT 格式：

i. 终端命令：**sudo fdisk /dev/mmcblk1** ，进入 sd 卡配置

```
tx2nx@tx2nx:~/Desktop$ sudo fdisk /dev/mmcblk1

Welcome to fdisk (util-linux 2.31.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

ii. 终端命令：**g**，新建 gpt 分区表

```
Command (m for help): g
Created a new GPT disklabel (GUID: E39DF30E-48FE-B041-A6FA-5EFAEC223CEA).
```

iii. 终端命令：**n**，新建分区

```
Command (m for help): n
Partition number (1-128, default 1):
First sector (2048-124735454, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-124735454, default 124735454):
```

iv. 终端命令：**w**， 保存分区信息

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

v. 终端命令：**sudo mke2fs -t ext4 /dev/mmcblk1p1，格式化分区**

```
tx2nx@tx2nx:~/Desktop$ sudo mke2fs -t ext4 /dev/mmcblk1p1
mke2fs 1.44.1 (24-Mar-2018)
Discarding device blocks: done
Creating filesystem with 15591675 4k blocks and 3899392 inodes
Filesystem UUID: 3c6ce310-2a86-4385-8dd0-86ab0089767e
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (65536 blocks): done
Writing superblocks and filesystem accounting information: done
```

vi. 终端命令：**sudo mount /dev/mmcblk1p1/mnt**，成功 mount 则 SD 卡配置成功

```
tx2nx@tx2nx:~/Desktop$ sudo mount /dev/mmcblk1p1 /mnt
```

b) NVIDIA Jetson 系统迁移 (!!!迁移前建议参考第 3 章进行系统备份)：

✓ **下面以 TX2NX 为例，其他设备替换命令中间的设备名称即可，设备名称可参考上面命令**

➢ 1. 终端命令： **git clone https://github.com/jetsonhacks/rootOnNVMe** 或者 https://github.com/jetsonhacks/rootOnNVMe 下载脚本

➢ 2. 修改 **copy-rootfs-ssd.sh** 文件，注释掉 mount 命令

```
#!/bin/bash
# Mount the SSD as /mnt
# sudo mount /dev/nvme0n1p1 /mnt
# Copy over the rootfs from the SD card to the SSD
sudo rsync -axHAWX --numeric-ids --info=progress2 --exclude={"/dev/","/proc/","/sys/","/tmp/"
,"/run/","/mnt/","/media/*","/lost+found"} / /mnt
# We want to keep the SSD mounted for further operations
# So we do not unmount the SSD
```

➢ 3. 进入 **rootOnNVMe** 文件夹，终端命令：**./copy-rootfs-ssd.sh**，复制系统文件至 SD 卡

```
tx2nx@tx2nx:~/Desktop/rootOnNVMe$ sudo ./copy-rootfs-ssd.sh
  1,149,753,593  71%   23.18MB/s    0:00:19  xfr#6703, ir-chk=2715/12064)
```

➢ 4. 参考第 2 章，进入 **Recovery 模式。**

➢ 5. （**烧录环境的 Ubuntu**，参考**前面章节**）进入**~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra**，打开终端：**sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx mmcblk1p1** 更新 EMMC 内部引导

```
greetrix@greetrix-virtual-machine:~/nvidia/nvidia_sdk/JetPack_4.6_Linux_JETSON_TX2_TARGETS/Linux_for_Tegra$ sudo ./flash.sh jetson-xavier-nx-devkit-tx2-nx mmcblk1p1
```

```
[   30.4511 ] Coldbooting the device
[   30.4521 ] tegradevflash_v2 --reboot coldboot
[   30.4531 ] Bootloader version 01.00.0000
[   30.6253 ]
*** The target t186ref has been flashed successfully. ***
Make the target filesystem available to the device and reset the board to boot f
rom external mmcblk1p1.
```

- 6. 重启 TX2NX，终端命令：**df -l**， 此时系统盘已经变为 SD 卡，并且原有 EMMC 上系统已经成功迁移。

```
tx2nx@tx2nx:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk1p1   59G   12G   44G  21% /
devtmpfs        1.6G     0  1.6G   0% /dev
tmpfs           1.9G   52K  1.9G   1% /dev/shm
tmpfs           1.9G   21M  1.9G   2% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           1.9G     0  1.9G   0% /sys/fs/cgroup
tmpfs           384M   12K  384M   1% /run/user/120
tmpfs           384M     0  384M   0% /run/user/1000
```

9. UART0 开启 DEBUG 信息

a) **Jetson XavierNX：**

1. 替换 **p3668.conf.common** 到 **Linux_for_Tegra** 根目录
2. 替换 **tegra194-a02-bpmp-p3668-a00.dtb** 到 **/Linux_for_Tegra/bootloader/t186ref**
3. 替换 **tegra194-mb1-bct-misc-flash.cfg** 到 **/Linux_for_Tegra/bootloader/t186ref/BCT**
4. 替换 **tegra194-mb1-bct-misc-l4t.cfg** 到 **/Linux_for_Tegra/bootloader/t186ref/BCT**
5. 替换 **tegra194-p3668-all-p3509-0000.dtb** 到 **/Linux_for_Tegra/kernel/dtb**
6. 参考章节二进入 Recovery 模式，使用 xavierNX 命令进行刷机

联系我们

- Github：https://github.com/WeActTC
- 码云：https://gitee.com/WeAct-TC
- 网站：https://www.weact-tc.cn/
- 淘宝：https://weactstudio.taobao.com/

WeAct Studio
官方淘宝店