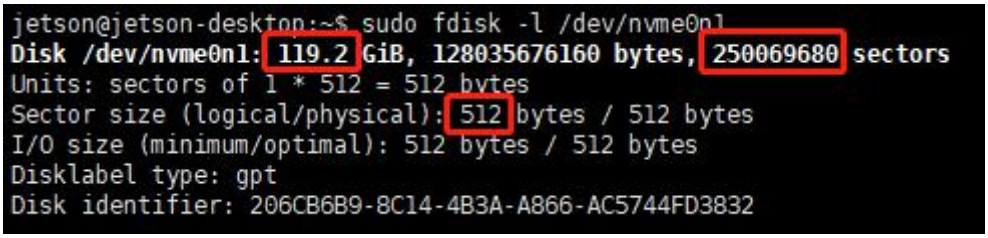
1. View NVME SSD information

Before restoring the system to the solid state drive, you need to check the information of the solid state drive to be restored.

Please insert the NVME solid state drive into the SSD socket of the Jetson Xavier NX, and power on the Jetson Xavier NX, open the terminal, and enter the following command to view the information.

sudo fdisk -l /dev/nvme0n1



Record the three data in the red box in the figure below.

The first data indicates the capacity of the SSD;

The second data indicates how many sectors. (num_sectors: 250069680);

The third data indicates the words occupied by each sector.(sector_size: 512).

2. Prepare L4T programming package

2.1 Download file

Enter this link to download L4T firmware of Jetson Xavier NX system.

https://developer.nvidia.com/embedded/linux-tegra

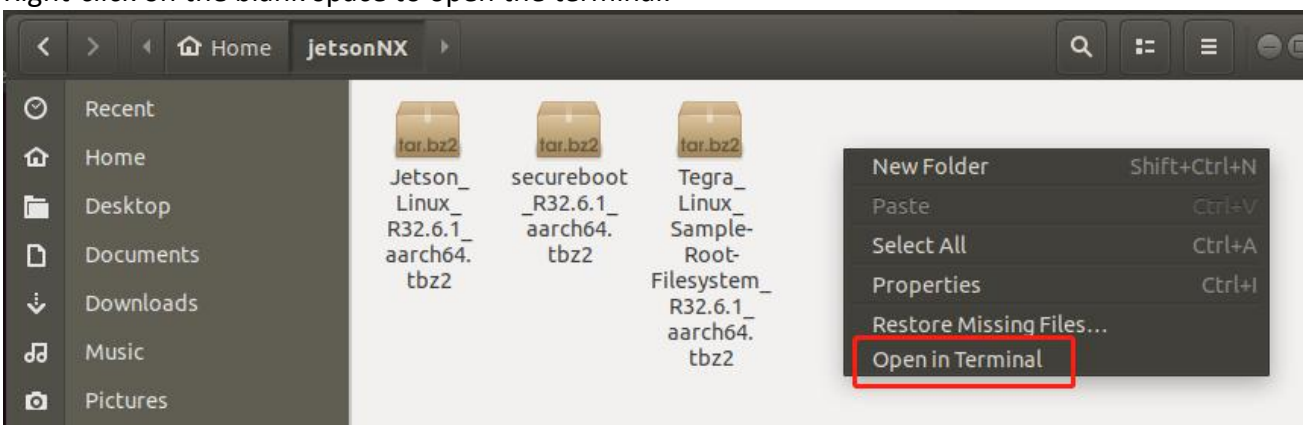
We need to download the three files related to Jetson Xavier NX L4T Driver Package[BSP], Sample Root Filesystem and Jetson Platform Fuse Burning and Secure Boot Documentation and Tools to the local.

32.6.1 Driver Details

	Jetson AGX Xavier Series, Xavier NX and TX2 Series	Jetson Nano, Nano 2GB and TX1
DRIVERS	L4T Driver Package (BSP)	L4T Driver Package (BSP)
	Sample Root Filesystem	Sample Root Filesystem
	NVIDIA Hardware Acceleration in the WebRTC Framework	
SOURCES	L4T Driver Package (BSP) Sources	L4T Driver Package (BSP) Sources
	Cboot Sources T186 Cboot Sources T194	
	Free RTOS Sources	

TOOLS	GCC 7.3.1 for 64 bit BSP and Kernel	
	Sources for the GCC 7.3.1 Tool Chain for 64-bit BSP and Kernel	
	CUDA Tools	
	NVIDIA Nsight Systems	
	NVIDIA Nsight Graphics	
	Jetson Platform Fuse Burning and Secure Boot Documentation and Tools	Jetson Platform Fuse Burning and Secure Boot Documentation and Tools
	Jetson Platform Over-The-Air Update Tools	

For example: I create a new jetsonNX folder in the virtual machine Ubuntu18.04 system, and download the file to the jetsonNX directory here.
Right-click on the blank space to open the terminal.



2.2 Input following command to extract file.
tar xf Jetson_Linux_R32.6.1_aarch64.tbz2
sudo tar xf Tegra_Linux_Sample-Root-Filesystem_R32.6.1_aarch64.tbz2 -C ./Linux_for_Tegra/rootfs/
sudo xf secureboot_R32.6.1_aarch64.tbz2

```
yahboom@YAB:~/jetsonNX$ ls
Jetson_Linux_R32.6.1_aarch64.tbz2
secureboot_R32.6.1_aarch64.tbz2
Tegra_Linux_Sample-Root-Filesystem_R32.6.1_aarch64.tbz2
yahboom@YAB:~/jetsonNX$ tar xf Jetson_Linux_R32.6.1_aarch64.tbz2
yahboom@YAB:~/jetsonNX$ sudo tar xf Tegra_Linux_Sample-Root-Filesystem_R32.6.1_a
arch64.tbz2 -C ./Linux_for_Tegra/rootfs/
[sudo] password for yahboom:
yahboom@YAB:~/jetsonNX$ sudo tar xf secureboot_R32.6.1_aarch64.tbz2
yahboom@YAB:~/jetsonNX$
```

2.3 Install qemu file, input “y” to confirm the installation.
sudo apt-get install qemu-user-static

```
yahboom@YAB:~/jetsonNX$ sudo apt-get install qemu-user-static
[sudo] password for yahboom:
```

2.4 Generate binary files
First, Input the following command to enter the Linux_for_Tegra folder, and run the command to generate the binary file.

cd Linux_for_Tegra
sudo ./apply_binaries.sh

```
yahboom@YAB:~/jetsonNX$ cd Linux_for_Tegra/
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ sudo ./apply_binaries.sh
```

When you see the final success, it means “OK”.

```
~/jetsonNX/Linux_for_Tegra
Removing QEMU binary from rootfs
Removing stashed Debian packages from rootfs
L4T BSP package installation completed!
Rename ubuntu.desktop --> ux-ubuntu.desktop
Disabling NetworkManager-wait-online.service
Disable the ondemand service by changing the runlevels to 'K'
Success!
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$
```

3. Modify system data
vim ./tools/kernel_flash/flash_l4t_nvme.xml

```
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ vim ./tools/kernel_flash/flash_l4t_nvme.xml
```

Press i to enter the edit mode, and then replace the data of sector_size and num_sectors queried in the first step.

```
<partition_layout version="01.00.0000">
    <device type="nvme" instance="0" sector_size="512" num_sectors="250069680">
        <partition name="master_boot_record" type="protective_master_boot_record">
            <allocation_policy> sequential </allocation_policy>
            <filesystem_type> basic </filesystem_type>
            <size> 512 </size>
            <file_system_attribute> 0 </file_system_attribute>
            <allocation_attribute> 8 </allocation_attribute>
            <percent_reserved> 0 </percent_reserved>
:wq
```

After the modification is completed, press ESC to exit the editing mode, then enter :wq and press Enter to save and exit.

4. Build system image
4.1 Build qspi startup components
Input the following commands in the Linux_for_Tegra directory
sudo ./tools/kernel_flash/l4t_initrd_flash.sh --no-flash jetson-xavier-nx-devkit-qspi internal

```
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ sudo ./tools/kernel_flash/l4t_initrd_fla
sh.sh --no-flash jetson-xavier-nx-devkit-qspi internal
[sudo] password for yahboom:
/home/yahboom/jetsonNX/Linux_for_Tegra/tools/kernel_flash/l4t_initrd_flash_inter
nal.sh --no-flash --no-flash jetson-xavier-nx-devkit-qspi internal
************************************
*                                  *
*  Step 1: Generate flash packages *
*                                  *
************************************
```

4.2 Build system image

Input the following commands in the Linux_for_Tegra directory

sudo ./tools/kernel_flash/l4t_initrd_flash.sh --no-flash --external-device nvme0n1p1 -c ./tools/kernel_flash/flash_l4t_nvme.xml -S 118GiB --showlogs jetson-xavier-nx-devkit-emmc nvme0n1p1

```
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ sudo ./tools/kernel_flash/l4t_initrd_flash.sh --no-flash --external-devi
ce nvme0n1p1 -c ./tools/kernel_flash/flash_l4t_nvme.xml -S 118GiB --showlogs jetson-xavier-nx-devkit-emmc nvme0n
1p1
```

Explanation of the above parameters:

--no-flash: Indicates that the system is only compiled and not burned.

--external-device nvme0n1p1: Indicates burning to the /dev/nvme0n1p1 device, that is, the APP partition of the SSD.

-c ./tools/kernel_flash/flash_l4t_nvme.xml: Specify the burned xml file, that is, the file modified in the third step.

-S 118GiB: Indicates the size of the space occupied by the system APP partition. This value is the SSD capacity queried in the first step -1. Since the actual capacity of the SSD used this time is only 119.2GiB, the system also needs to reserve 1GiB space for other partitions, so the APP can occupy 118GiB.

--showlogs: Indicates that LOG information is displayed.

jetson-xavier-nx-devkit-emmc nvme0n1p1: Indicates that the Jetson Xavier NX device and the nvme0n1p1 partition are burned.

```
/tmp/tmp.nbTSzp2f1R ~/jetsonNX/Linux_for_Tegra
writing boot image config in bootimg.cfg
extracting kernel in zImage
extracting ramdisk in initrd.img
/tmp/tmp.nbTSzp2f1R/initrd /tmp/tmp.nbTSzp2f1R ~/jetsonNX/Linux_for_Tegra
66117 blocks
80916 blocks
/tmp/tmp.nbTSzp2f1R ~/jetsonNX/Linux_for_Tegra
flashimg0=boot0.img
~/jetsonNX/Linux_for_Tegra
Success
Cleaning up...
Finish generating flash package.
Put device in recovery mode, run with option --flash-only to flash device.
```

Finally, when you see "Success" it means the system was successfully built.

5.Replace the IMG file of the system

5.1 Copy the previously backed up nx_rootfs.raw file to the bootloader

sudo cp nx_rootfs.raw jetsonNX/Linux_for_Tegra/bootloader/

```
yahboom@YAB:~$ sudo cp nx_rootfs.raw jetsonNX/Linux_for_Tegra/bootloader/
```

5.2 Enter bootloader directory

cd jetsonNX/Linux_for_Tegra/bootloader/

```
yahboom@YAB:~$ cd jetsonNX/Linux_for_Tegra/bootloader/
yahboom@YAB:~/jetsonNX/Linux_for_Tegra/bootloader$
```

5.3 Convert raw files to img files and view the generated files

./mksparse --v --fillpattern=0 nx_rootfs.raw nx_rootfs.img

```
yahboom@YAB:~/jetsonNX/Linux_for_Tegra/bootloader$ ./mksparse --v --fillpattern=0 nx_rootfs.raw nx_rootfs.img

---- Raw to Sparse Image Converter v1.0 ----------------------------
  0: RAW:    34025472(   8307 blks) ==>          28:34025484
  1: SKP:     3919872(    957 blks) ==>    34025512:12
  2: RAW:    96337920(  23520 blks) ==>    34025524:96337932
  3: SKP:     4141056(   1011 blks) ==>   130363456:12
  4: RAW:    12849152(   3137 blks) ==>   130363468:12849164
  5: SKP:        4096(      1 blks) ==>   143212632:12
```

ls *.img

```
5861: SKP:       98304(     24 blks) ==>  6145336024:12
5862: RAW:    12849152(   3137 blks) ==>  6145336036:12849164
5863: SKP:   782999552( 191162 blks) ==>  6158185200:12
-- Total: ----------------------------------------------------
5864 CHUNK  9987665920(2438395 blks) ==>  6158185212(1503446 blks)

yahboom@YAB:~/jetsonNX/Linux_for_Tegra/bootloader$ ls *.img
boot0.img             camera-rtcpu-sce.img  nx_rootfs.img  tos.img               tos_t194.img
boot.img              eks.img               recovery.img   tos-mon-only.img      tos-trusty.img
camera-rtcpu-rce.img  l4t_initrd.img        system.img     tos-mon-only_t194.img tos-trusty_t194.img
yahboom@YAB:~/jetsonNX/Linux_for_Tegra/bootloader$
```

5.4 Return to the Linux_for_Tegra directory

cd ..

5.5 Copy the original system files as a backup (in order to save space, no backup can be made)

sudo cp tools/kernel_flash/images/external/system.img tools/kernel_flash/images/external/system.img.bak

5.6 Copy and overwrite the system.img system file, copy the nx_rootfs.img file generated in the previous step to the external/ directory, and overwrite the system.img file.

sudo cp bootloader/nx_rootfs.img tools/kernel_flash/images/external/system.img

```
yahboom@YAB:~/jetsonNX/Linux_for_Tegra/bootloader$ cd ..
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ sudo cp tools/kernel_flash/images/external/system.img tools/kernel_flash
/images/external/system.img.bak
[sudo] password for yahboom:
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ sudo cp bootloader/nx_rootfs.img tools/kernel_flash/images/external/syst
em.img
```

6. Enter flash mode

6.1 Jetson Xavier NX enters system REC flashing mode.

Connect the jumper caps to the FC REC and GND pins, that is, the second and third pins of the carrier board, as shown below:

6.2 Connect the HDMI display, mouse, keyboard and microUSB cable to the Jetson Nano, and finally plug in the power.

Since the jumper cap has been connected to the FC REC and GND pins in the previous step, after turning on the power switch, the Jetson NANO will automatically enter the REC flashing mode.

If a prompt appears on the computer desktop, we need to choose to connect the device to the virtual machine.

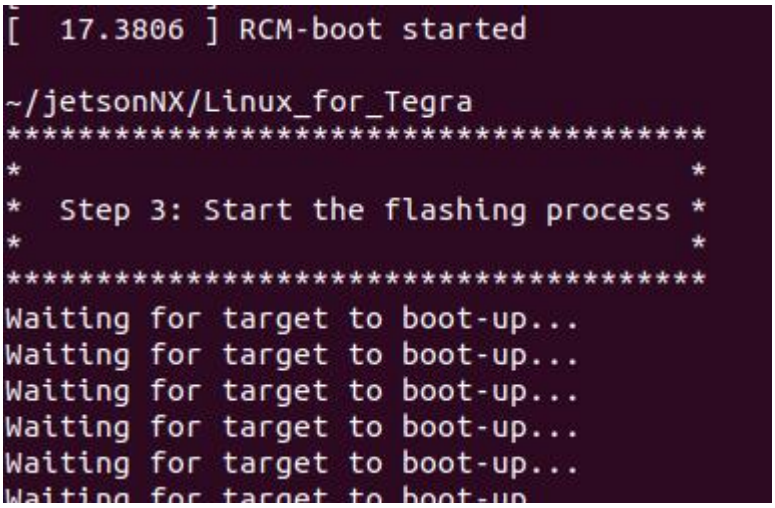
7. Start flashing

Open a terminal and execute the following commands in the Linux_for_Tegra directory:

sudo ./tools/kernel_flash/l4t_initrd_flash.sh --flash-only --external-device nvme0n1p1 -c ./tools/kernel_flash/flash_l4t_nvme.xml -S 118GiB --showlogs jetson-xavier-nx-devkit-emmc nvme0n1p1

```
yahboom@YAB:~/jetsonNX/Linux_for_Tegra$ sudo ./tools/kernel_flash/l4t_initrd_flash.sh --flash-only --external-de
vice nvme0n1p1 -c ./tools/kernel_flash/flash_l4t_nvme.xml -S 118GiB --showlogs jetson-xavier-nx-devkit-emmc nvme
0n1p1
```

!!!Note:
After starting to restore the system, Jetson Xavier NX will restart again. If you are using a virtual machine, you need to manually connect Jetson Xavier NX to the virtual machine, otherwise it will fail to connect and exit.



After the flashing is complete, the Jetson Xavier NX will automatically power on.
Note: After programming the system, please unplug the jumper caps of FC REC and GND.