内容目录

AGX Xavier 环境配置

1 ibus 中文输入法安装

sudo apt-get install ibus-sunpinyin

2 Qt5 安装及配置

2.1 AGX ubuntu18.04 命令行安装 Qt

sudo apt-get install qt5-default qtcreator -y

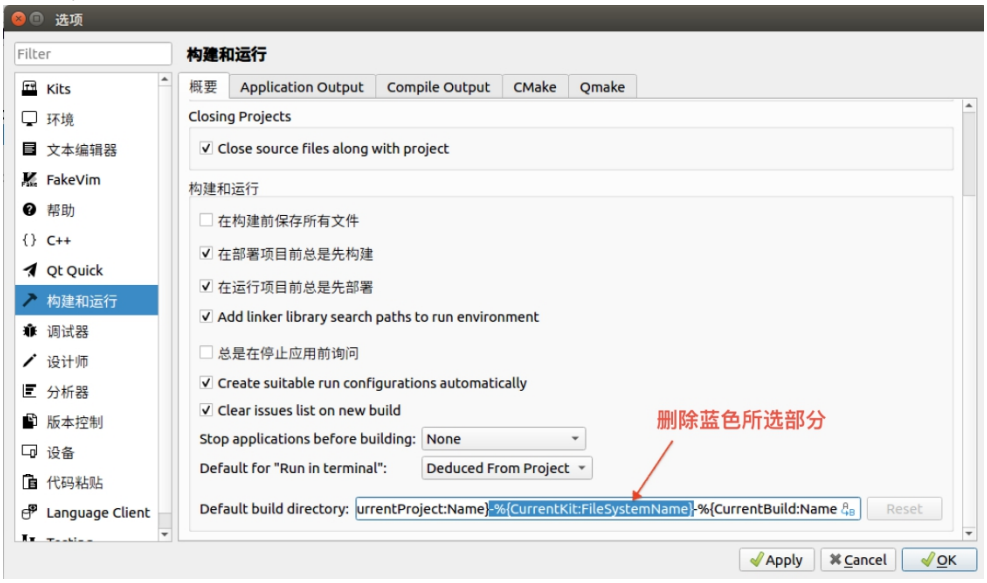
安装对串口类的支持：

sudo apt-get install libqt5serialport5-dev -y

安装 QWebEngineView 类：

sudo apt-get install qtwebengine5-dev qtpositioning5-dev -y

2.2 设置 Qt 的 build 目录名称生成格式，删除图中蓝色所选部分。



2.3 QT Creator 配置多核编译

菜单栏->工具-> 选项->构建与运行->构建套件->点击自动检测内容->在同一页面找到 Environment ->点击 change ->

在弹出的窗口 添加 MAKEFLAGS=-j8

3 ROS 安装

备注：推荐使用脚本一键安装，步骤如下：

$ cd ~

$ git clone https://github.com/jetsonhacks/installROSXavier.git

$ cd installROSXavier

$ gedit installROS.sh

将国外官方源：

sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

更改为国内镜像源：

sudo sh -c 'echo "deb https://mirrors.tuna.tsinghua.edu.cn/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

将 Key：

sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654

更改为：

sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key F42ED6FBAB17C654

$ ./installROS.sh -p ros-melodic-desktop-full

今后以下 3.1 ~ 3.2.7 步骤可以不再使用！！！

3.1 设置-->软件和更新：保证前四个选项都打钩

3.2 终端输入：

3.2.1 设置 sources.list（两个源选一个即可）

国外官方源：

sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

国内镜像源：

```
sudo sh -c 'echo "deb https://mirrors.tuna.tsinghua.edu.cn/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

3.2.2 设置 key

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key F42ED6FBAB17C654
```

3.2.3 更新 package

```
sudo apt-get update
```

3.2.4 安装 ROS 完整版（ubuntu18 对应版本为 melodic，ubuntu16 对应版本为 kinetic）

ubuntu18.04:

```
sudo apt-get install ros-melodic-desktop-full
```

ubuntu16.04:

```
sudo apt-get install ros-kinetic-desktop-full
```

3.2.5 初始化 rosdep （不翻墙可能会失败，不要紧，暂时不影响使用）

```
sudo rosdep init

rosdep update
```

3.2.6 配置 ROS 环境

ubuntu18.04:

```
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
```

ubuntu16.04:

```
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
```

3.2.7 使.bashrc 文件生效

```
source ~/.bashrc
```

4 VNC 远程连接配置

注意：进行 1)~3)步骤前无法打开系统设置 -->桌面共享！！！

1). 在 /usr/share/glib-2.0/schemas/org.gnome.Vino.gschema.xml 中添

加如下键值

```
<key name='enabled' type='b'>
    <summary>Enable remote access to the desktop</summary>
    <description>
        If true, allows remote access to the desktop via the RFB
      protocol. Users on remote machines may then connect to
          the desktop using a VNC viewer.
    </description>
   <default>false</default>
</key>
```

2). 为 Gnome 编译修改后的式样

```
sudo glib-compile-schemas /usr/share/glib-2.0/schemas
```

3). 禁用 Vino 加密连接

```
gsettings set org.gnome.Vino require-encryption false
```

4) 设置桌面共享
系统设置 -->桌面共享（对应选项打钩，密码设置成 123456）

5) 运行 vino-server （需要设置该文件自启动，请按照下面的“5 设置自启动项” 进行设置）

/usr/lib/vino/vino-server

Ubuntu 下建议使用 Remmina 进行远程登陆访问；
Windows 下建议使用免费开源的 VNC-viewer 进行远程登陆访问。
如果追求访问速度，还可以尝试安装和使用 NoMachine。

5 设置自启动项

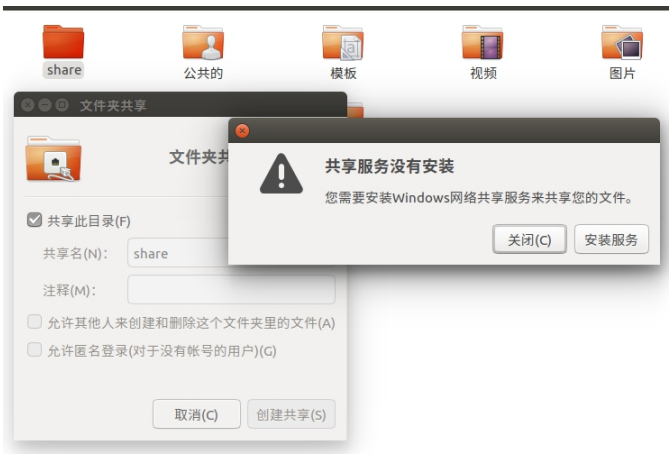
1)终端输入

gnome-session-properties

显示如下界面，点击添加（add），设置名称，浏览选择需要自启动的文件。下次开机就会自启动。（vino-server 需要自启动）

6 建立共享文件夹

新建一个空文件夹（取名 share），鼠标右键点击该文件夹选择“本地网络共享”，出现如下界面，选择安装服务。三个选项均要打钩。



7 设置 Xavier 固定 IP 地址（192.168.1.102）

点击屏幕右上角网络图标，选择编辑连接，编辑有线连接 1，IPv4 设置，方法选择手动。点击增加。填入 IP 信息，选择保存。重新连接网络 IP 设置完成。

8.串口、can 驱动

添加用户

```
sudo gpasswd --add nvidia dialout
```

将 rc.local 文件复制到/etc 目录下实现自启动

```
sudo cp rc.local /etc/
```

添加权限

```
sudo chmod +777 /etc/rc.local
```

rc.local 文件内容：

```
sudo modprobe can
```

```
sudo modprobe can-raw
sudo modprobe can-dev
sudo modprobe mttcan

sudo busybox devmem 0x0c303000 32 0x0000C400
sudo busybox devmem 0x0c303008 32 0x0000C458
sudo busybox devmem 0x0c303010 32 0x0000C400
sudo busybox devmem 0x0c303018 32 0x0000C458

sudo ip link set can0 type can bitrate 500000 dbitrate 2000000 berr-reporting on fd on
sudo ip link set can1 type can bitrate 500000 dbitrate 2000000 berr-reporting on fd on

sudo ip link set up can0
sudo ip link set up can1

sudo modprobe usbserial
sudo modprobe pl2303

#sudo /usr/bin/jetson_clocks
sudo jetson_clocks
sudo nvpmodel -m 0

exit 0
```

9.其他配置项

9.1 安装 busybox

```
sudo apt-get install busybox
```

9.2 安装支持 windows 远程桌面 mstsc 登录的软件

```
sudo apt-get install xrdp
```

9.3 在 home 目录下创建一个 map 目录

用来存放地图文件，因采集地图的软件不会主动生成 map 目录，故需手动创建。

9.4 安装 protoc

sudo apt install protobuf-compiler

9.5 安装 telnet

sudo apt-get install openbsd-inetd
sudo apt-get install telnetd
sudo /etc/init.d/openbsd-inetd restart

9.6 安装 patchelf

sudo apt install patchelf

9.7 安装和配置 SVN

sudo apt install subversion

mkdir -p ~/code/ADS

cd ~/code/ADS

svn checkout https://60.247.58.121:5819/svn/adc_svn/adc_auto_driving/trunk/ADS_modularization/modularization

输入 p 永久接受，然后输入登陆用户的密码、SVN 账号及密码。

9.8 安装串口调试助手

sudo apt-get install cutecom

9.9 安装 eigen-3.3.7

下载解压
wget https://gitlab.com/libeigen/eigen/-/archive/3.3.7/eigen-3.3.7.tar.gz
tar -zxvf eigen-3.3.7.tar.gz
编译安装 eigen3.3.7
cd eigen-3.3.7/
mkdir build
cd build
cmake ..
sudo make
sudo make install

复制 Eigen 库到 /usr/local/include 中

sudo cp -r /usr/local/include/eigen3/Eigen /usr/local/include

9.10 固定内核版本

sudo apt-mark hold linux-image-generic linux-headers-generic

9.11 禁用 ROS 的 sudo apt-get update 更新

sudo rm /etc/apt/sources.list.d/ros-latest.list

到此 AGX Xavier 环境配置即完成！！！！

以下内容如果没有需求可以不用管！！！！

10 AGX 使用过程中遇到问题攻略

10.1 没有公钥

sudo apt-get update 由于没有公钥无法验证下列签名： NO_PUBKEY F42ED6FBAB17C654

解决方法：安装公钥

sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys F42ED6FBAB17C654

10.2 关于 AGX 日志文件过大造成系统无法开机问题解决方法

10.2.1 编辑脚本文件

nvidia@nvidia-desktop:~$ mkdir ～/adc
nvidia@nvidia-desktop:~$ cd ～/adc

```
nvidia@nvidia-desktop:~$ touch auto_del_log.sh
nvidia@nvidia-desktop:~$ chmod +x auto_del_log.sh
nvidia@nvidia-desktop:~$ gedit auto_del_log.sh
```

文件内容：

```
#!/bin/sh
cat /dev/null > /var/log/syslog
cat /dev/null > /var/log/wtmp
```

保存，关闭退出　　注：如果有其他大的日志文件，可以追加脚本文件内容。

10.2.2 添加到自动更新脚本

nvidia@nvidia-desktop:~$ crontab -e　　//编辑 nvidia 用户的 cron 服务

按下 Enter 键后，如果是第一次使用终端将要求您选择一个编辑器以打开此文件。可以输入 2 选择 **nano** 编辑器，如果想每隔 1 小时执行一次脚本，可以追加输入：

```
* */1 * * * ~/adc/auto_del_log.sh
```

使用 nano 的话 Ctrl + O 输入文件名保存，Ctrl + X 退出；使用 vi 的话:wq 保存退出。
然后重启 cron 服务：

```
nvidia@nvidia-desktop:~$ service cron restart
```

10.2.3 备注说明：

crontab 样式注释：

```
# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name  command to be executed
```

以上非命令字段中还可以使用以下特殊字符：
星号（*）：代表所有可能的值
逗号（,）:可以用逗号隔开的值指定一个列表范围，例如：“1,2,3”
中杠（-）：表示一个整数范围，例如："2-5"表示“2,3,4,5”
正斜杠（/）:表示指定时间的间隔频率，例如“0-23/2”表示每两小时执行一次；*/10：在 minute 字段中表示每 10 分钟执行一次。

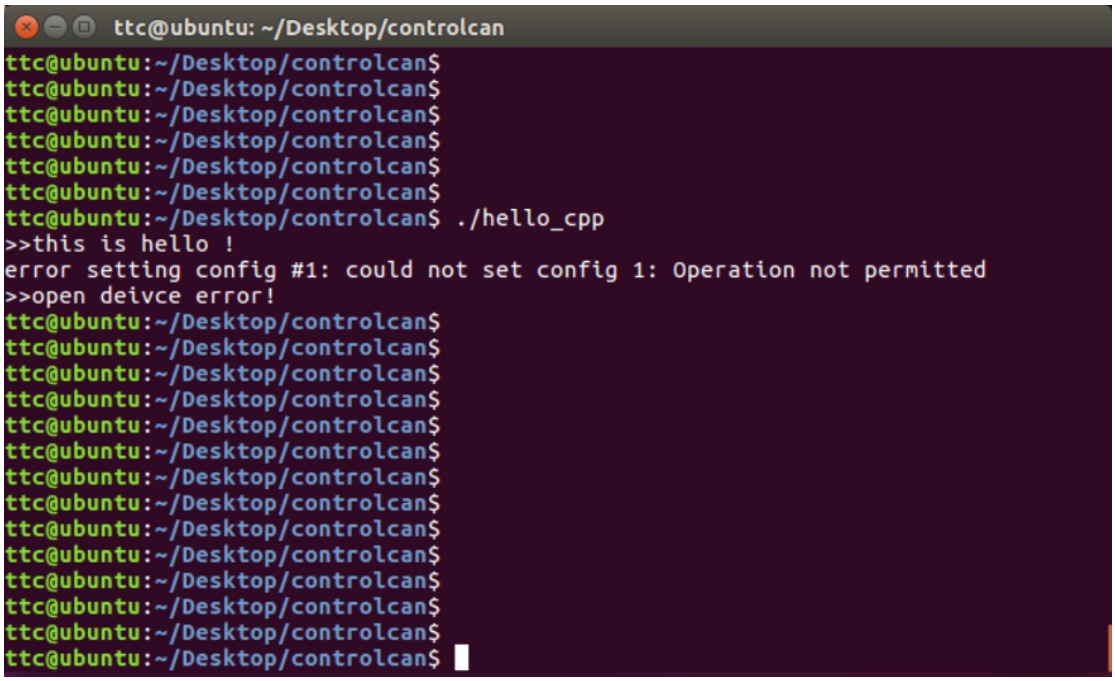
10.3 Kvaser 不需要签名的 Linux 驱动安装

```
# sudo apt-get install build-essential
# sudo apt-get install linux-headers-`uname -r`
# wget --content-disposition "https://www.kvaser.com/downloads-kvaser/?utm_source=software&utm_ean=7330130980754&utm_status=latest"
# tar xvzf linuxcan.tar.gz
# cd linuxcan
# make
# sudo make install
```

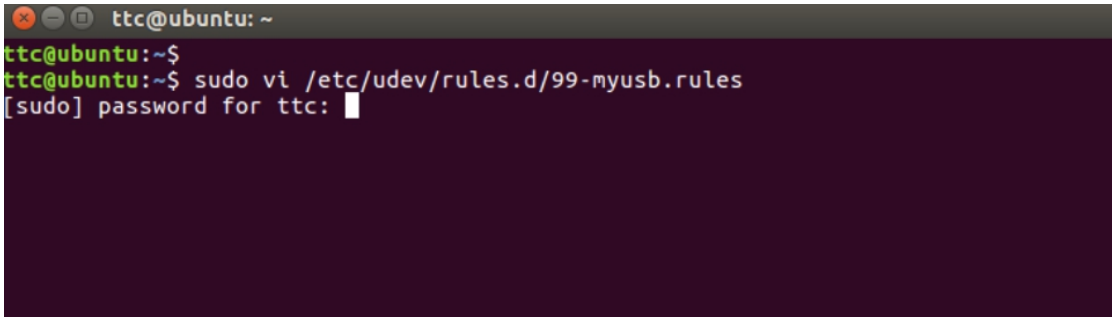
10.4 VCI USB 权限设置

因 Linux 系统下将涉及到 usb 底层驱动的调用,运行时,一定要加 sudo 获取权限运行,否则 USB 设备没有权限操作。 现提供一种 USB 权限设置,配置后,可以不加权限运行。

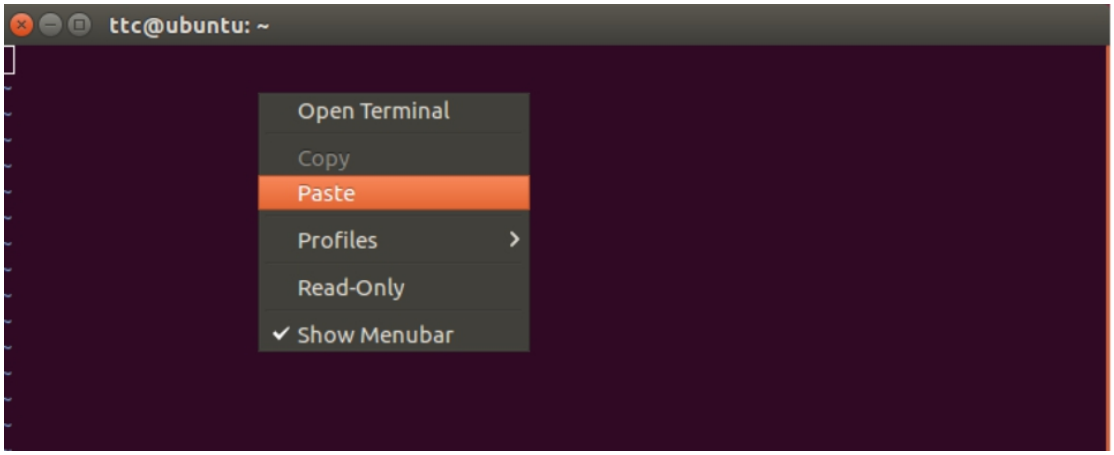
10.4.1 不加 sudo 权限运行,将会报如下错误:



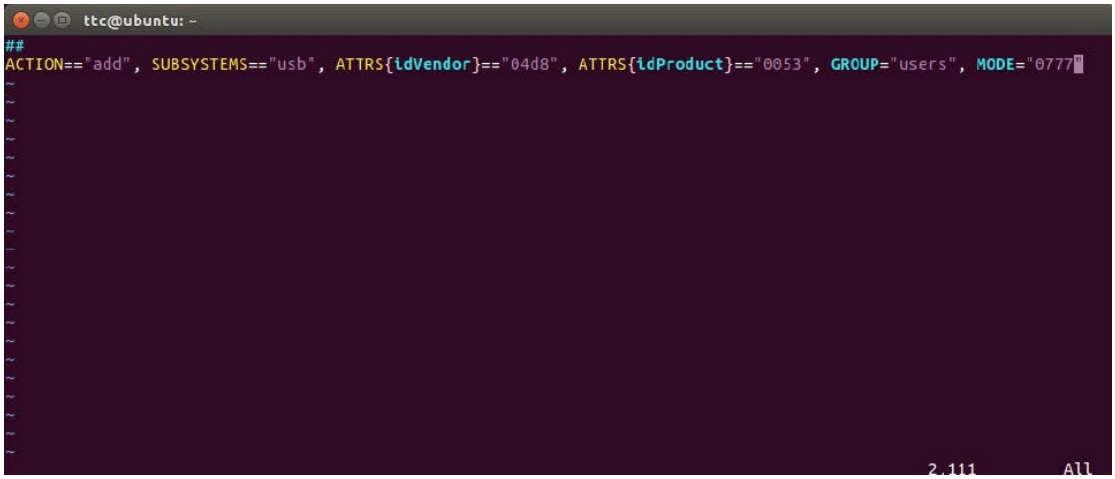
10.4.2 下面创建一个新的 udev 规则。名称取为:99-myusb.rules
sudo vi /etc/udev/rules.d/99-myusb.rules
注意:
1、 数字 99 最好不要改动,否则可能设置失败
2、 要加 sudo



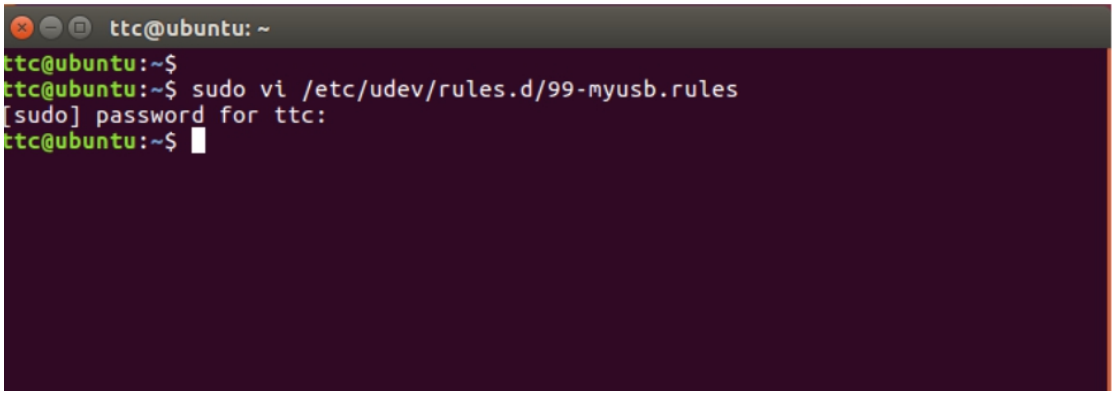
10.4.3 把以下两行代码复制到新建的 99-myusb.rules 文件中
注意:按键盘上 Insert 键切换到“REPLACE”输入模式
##
ACTION=="add",SUBSYSTEMS=="usb", ATTRS{idVendor}=="04d8",GROUP="users", MODE="0777"
ATTRS{idProduct}=="0053",



10.4.4 按“Esc”键一次



10.4.5 直接输入“:wq”回车,即保存退出



10.4.6 插拔一下 USBCAN 设备或重启一下电脑后,即可不加 sudo 权限运行程序了。

10.5 远程桌面登陆不允许配置网络

在使用远程桌面操系统过程中，不允许关闭“启用联网”选项，否则远程登陆会立即失

效。

10.6 ubuntu 为 USB 串口绑定固定的设备名

ubuntu USB 设备号为从零开始依次累加，多个设备每次开机后设备号不固定.
udev 的规则，可以参考博客：http://blog.csdn.net/cokewei/article/details/8281239
将端口重映射到固定的名字，并且设置其权限为可读。使用对应的 id 设备映射到固定的名字上。

使用命令 lsusb 查看对应的 usb 端口信息

```
apollo@apollo-ThinkPad-X390:~$ lsusb
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 002: ID 0bda:0411 Realtek Semiconductor Corp.
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 04ca:7070 Lite-On Technology Corp.
Bus 001 Device 007: ID 0bda:8179 Realtek Semiconductor Corp. RTL8188EUS 802.11n
Wireless Network Adapter
Bus 001 Device 011: ID 0403:6001 Future Technology Devices International, Ltd FT
232 USB-Serial (UART) IC
Bus 001 Device 003: ID 0bda:5411 Realtek Semiconductor Corp.
Bus 001 Device 002: ID 046d:c534 Logitech, Inc. Unifying Receiver
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
apollo@apollo-ThinkPad-X390:~$
```

udev 的规则

$kernel, %k：设备的内核设备名称，例如：sda、cdrom。

ID 0403:6001 表示 usb 设备的 ID(这个 ID 由芯片制造商设置，可以唯一表示该设备)
0403 usb_device_descriptor.idVendor --VID
6001 usb_device_descriptor.idProduct --PID

依据上面信息写 udev 文件

串口设备信息

Bus 001 Device 011: ID 0403:6001 Future Technology Devices International, Ltd FT232 USB-Serial (UART) IC

sudo vim /etc/udev/rules.d/p900.rules

KERNEL=="ttyUSB*", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", MODE:="0777", SYMLINK+="p900"

创建生效后**重新插拔 USB**(也可以用命令实现

service udev reload

service udev restart

)

ls -l /dev | grep ttyUSB*

显示

```
apollo@apollo-ThinkPad-X390:~$ ls -l /dev | grep ttyUSB*
lrwxrwxrwx  1 root   root             7 Sep  7 10:40 p900 -> ttyUSB0
crwxrwxrwx  1 root   dialout 188,     0 Sep  7 10:40 ttyUSB0
apollo@apollo-ThinkPad-X390:~$
```

多个不同型号设备可使用这种方法来区分。

打开设备时，用（/dev/p900）即可。

10.7 刷机 Jetpack4.4 后找不到 OpenCV

Xavier刷机完成之后，查看OpenCV版本，会报错No package 'opencv' found。

报这个错是因为没有找到opencv.pc，而刷机时是安装了OpenCV的，不过是OpenCV4，我在/usr/lib/aarch64-linux-gnu/pkgconfig中找到了opencv4.pc，把它复制到/usr/lib/pkgconfig下，并改名为opencv.pc就好了。

11 Jetson AGX Xavier 系统备份与克隆系统到新 Xavier 板

11.1 方法一（DD 方式备份系统）

准备：

- 需要有刷机包，通过刷机教程完成安装之后就有。
- 安装有刷机包的 PC 机

备份：

- xavier 正常进入系统

- 标记系统为只读

```
sudo echo "u" | sudo dd of=/proc/sysrq-trigger
```

- 通过 DD 方式备份系统

```
sudo dd if=/dev/mmcblk0p1 | ssh ubu@192.168.0.85 dd of=/home/ubu/xavier-image.raw
```

- ubu 替换为自己的主机
- 其中生成的 xavier-image.raw 就是镜像

还原：

- 进入刷机包

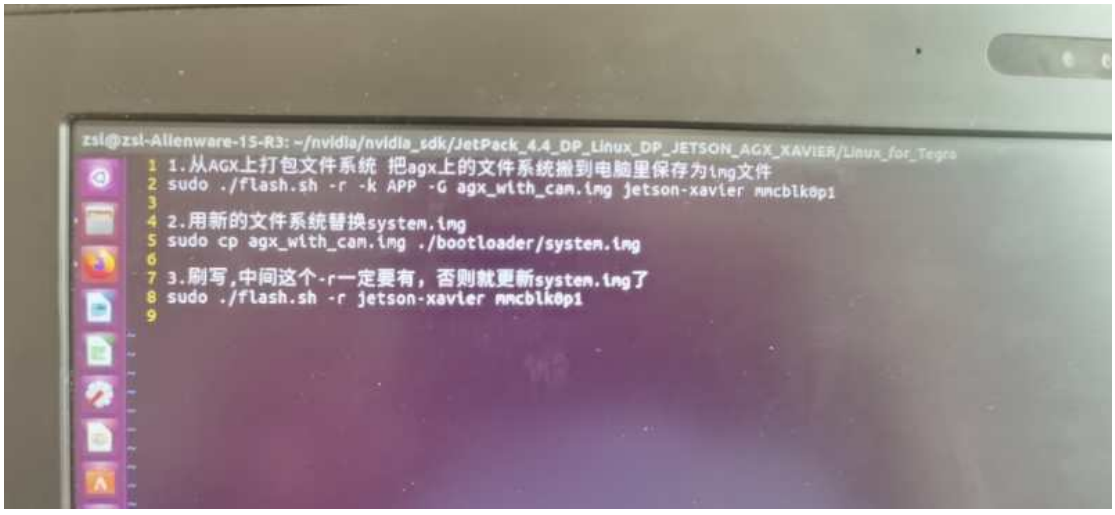
```
cd /home/nvidia/Downloads/Xavier/Linux_for_Tegra/bootloader/
mv system.img system.img.bk
ln -s /home/ubu/xavier-image.raw system.img
```

- 还原

```
cd /home/nvidia/Downloads/Xavier/Linux_for_Tegra/
sudo ./flash.sh -r jetson-xavier mmcblk0p1
```

- 大概要 20 分钟左右完成刷机，即可恢复系统。

11.2 方法二（直接使用 flash.sh 备份系统）



12 查看系统和软件版本参数状态

12.1 Jetpack4.2 版本信息

说明：

- 介绍如何查看 Xavier 系统和软件版本和参数
- 测试 Xavier 版本：Jetpack4.2.3

步骤：

- 查看架构

```
sudo apt-cache show nvidia-jetpack
```

- 查看 Jetson Xavier L4T 版本

```
ubuntu@AiROS:~$ head -n 1 /etc/nv_tegra_release
# R32 (release), REVISION: 2.3, GCID: 17644089, BOARD: t186ref, EABI: aarch64, DATE: Tue Nov  5 21:48:17 UTC 2019
```

- 查看 TensorRT 的版本

```
ubuntu@AiROS:~$ dpkg -l | grep TensorRT
ii  graphsurgeon-tf      5.1.6-1+cuda10.0              arm64        GraphSurgeon for TensorRT package
ii  libnvinfer-dev       5.1.6-1+cuda10.0              arm64        TensorRT development libraries and headers
ii  libnvinfer-samples   5.1.6-1+cuda10.0              all          TensorRT samples and documentation
ii  libnvinfer5          5.1.6-1+cuda10.0              arm64        TensorRT runtime libraries
ii  python-libnvinfer    5.1.6-1+cuda10.0              arm64        Python bindings for TensorRT
ii  python-libnvinfer-dev    5.1.6-1+cuda10.0          arm64        Python development package for TensorRT
ii  python3-libnvinfer       5.1.6-1+cuda10.0          arm64        Python 3 bindings for TensorRT
ii  python3-libnvinfer-dev   5.1.6-1+cuda10.0          arm64        Python 3 development package for TensorRT
ii  tensorrt                 5.1.6.1-1+cuda10.0        arm64        Meta package of TensorRT
ii  uff-converter-tf         5.1.6-1+cuda10.0          arm64        UFF converter for TensorRT package
```

- 查看系统版本

```
ubuntu@AiROS:~$ cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.3 LTS"
```

- 查看系统内核：

```
ubuntu@AiROS:~$ uname -a
Linux AiROS 4.9.140-tegra #1 SMP PREEMPT Tue Nov 5 13:37:19 PST 2019 aarch64 aarch64 aarch64 GNU/Linux
```

- 查看内存：

```
ubuntu@AiROS:~$ free -m
              total        used        free      shared  buff/cache   available
Mem:          15690        2630        3852          82        9207       12921
Swap:          7845           0        7845
```

- 查看 CPU

```
ubuntu@AiROS:~$ lscpu
Architecture:        aarch64
Byte Order:          Little Endian
CPU(s):              8
On-line CPU(s) list: 0-3
Off-line CPU(s) list: 4-7
Thread(s) per core:  1
Core(s) per socket:  2
Socket(s):           2
Vendor ID:           Nvidia
Model:               0
Model name:          ARMv8 Processor rev 0 (v8l)
Stepping:            0x0
CPU max MHz:         2265.6001
CPU min MHz:         115.2000
BogoMIPS:            62.50
L1d cache:           64K
L1i cache:           128K
L2 cache:            2048K
L3 cache:            4096K
Flags:               fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp
```

- 查看硬盘空间

```
ubuntu@AiROS:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk0p1   28G   12G   15G  44% /
none            7.7G     0  7.7G   0% /dev
tmpfs           7.7G   10M  7.7G   1% /dev/shm
tmpfs           7.7G   56M  7.7G   1% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           7.7G     0  7.7G   0% /sys/fs/cgroup
tmpfs           1.6G  148K  1.6G   1% /run/user/1000
```

- 查看 cuda 版本

```
ubuntu@AiROS:~$ /usr/local/cuda/bin/nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Mon_Mar_11_22:13:24_CDT_2019
Cuda compilation tools, release 10.0, V10.0.326
```

- 查看 cudnn 版本

ubuntu@AiROS:~$ cat /usr/include/cudnn.h | grep CUDNN_MAJOR -A 2

#define CUDNN_MAJOR 7

#define CUDNN_MINOR 5

#define CUDNN_PATCHLEVEL 0

--

#define CUDNN_VERSION (CUDNN_MAJOR * 1000 + CUDNN_MINOR * 100 + CUDNN_PATCHLEVEL)

#include "driver_types.h"

- 查看 opencv 版本

ubuntu@AiROS:~$ pkg-config --modversion opencv

3.3.1

- 查看 python 版本

ubuntu@AiROS:~$ python -V

Python 2.7.15+

- 查看 python3 版本

ubuntu@AiROS:~$ python3 -V

Python 3.6.9

12.2 JetPack 4.4 版本信息

JetPack 4.4 Highlights:

- Support for the new Jetson Xavier NX module [1] and Jetson Xavier NX Developer Kit
- Support for CUDA 10.2, and TensorRT 7.1.3 and cuDNN 8.0.0
- Support for Vulkan 1.2 [1] and VPI 0.3 [1] (Developer Preview)
- Support for upgrading JetPack and L4T using Debian package management tool
- Support for Generic Timestamping Engine (GTE) for Jetson AGX Xavier and Jetson Xavier NX
- Support for Dynamic Frequency Scaling (DFS) for Video Image Compositor (VIC) using actmon
- SE (Security Engine) samples to demonstrate hardware backed authentication and encryption capabilities of Jetson TX2 series, Jetson AGX Xavier and Jetson Xavier NX modules.
- Utility to fuse multiple Jetson modules simultaneously
- Option to specify APP partition size on the microSD card during initial configuration at first boot of Jetson Nano Developer Kit

JetPack 4.4 components:

- L4T R32.4.3
- CUDA 10.2
- cuDNN 8.0.0
- TensorRT 7.1.3
- VisionWorks 1.6
- OpenCV 4.1
- Vulkan 1.2
- VPI 0.3 (Developer Preview)
- Nsight Systems 2020.2
- Nsight Graphics 2020.1
- Nsight Compute 2019.3

13 系统使用小技巧

13.1 查看系统对外 IP

#curl cip.cc

13.2 安装和使用比 tree 更好用的文件查看工具

#sudo apt-get install ranger
#ranger

13.3 监视指定网络接口的数据包

#sudo apt-get install tcpdump
#tcpdump -i eth0

13.4 监视系统端口

#sudo apt-get install lsof
#lsof -i

13.5 每次上电同步系统时间防止编译报错

#sudo apt-get install ntpdate
然后将如下命令添加到/etc/rc.local 最后一行

sudo ntpdate time.windows.com

13.6 使用向日葵远程控制软件

软件安装包放在了 SVN 云端，需要同步到本地
#svn checkout https://60.247.58.121:5819/svn/adc_svn/adc_auto_driving/trunk/ADS_modularization/modularization/thirdpartylib/Sunlogin
切换到 Sunlogin 目录下，执行安装命令
#sudo dpkg -i sunlogin.deb
安装并切换 lightdm 图形界面,否则回向日葵软件会闪退。
$ sudo apt install lightdm
$ sudo dpkg-reconfigure lightdm

13.7 安装*.deb 包时缺少依赖

```
sudo dpkg -i *.deb
```

如果出现依赖问题，执行：

```
sudo apt install -f
```

然后再次安装*.deb。