설치 가이드

#### **Version history**

문제 : 만약 권한 문제로 업데이트가 안되는 경우 apt-get update && apt-get upgrade -y 의 명령어 실행

1. sudo add-apt-repository ppa:graphics-drivers/ppa
2. sudo apt update
3. sudo apt-get install nvidia-driver-버전

-> 문제 : 위의 코드로 설치가 안되는 경우 Driver 버전을 지원하지 않는 경우일 확률이 높음  
-> apt-cache search nvidia 통해서 설치 가능한 Driver 확인  
-> 해결 : sudo apt-get install nvidia-버전2 으로 돌아갔음

**sudo apt-get update**

* **sudo apt-get install build-essential**

**sudo apt install gcc**

#### **IPC BIOS Setup Configuration**

1. BIOS 진입 (F2 key or F???)
2. Advanced 탭 => Fan Control Configuration

Fan Start Trip Point: 30 => **20**으로 변경

Fan Max. Trip Point: 75 => **50**으로 변경

1. Advanced 탭 => System Agent (SA) Configuration => Graphics Configuration  
   Primary Display: AUTO => IPGX **~~PEG~~**로 변경
2. Power 탭 => Power & Performance  
   SKU Power Config: 35W => **MAX. TDP**로 변경
3. Exit 탭 => Exit Saving Changes 눌러 변경사항 저장 후 IPC 재부팅

#### **Ubuntu Installation**

1. Ubuntu 18.04 설치
2. sudo apt update
3. sudo apt install gzip
4. sudo apt-get install build-essential libncurses-dev libssl-dev libelf-dev flex bison

#### **Install Building realtime Linux kernel ( Kernel Ver.: 5.4.84-rt47)**

1.

mkdir ~/kernel

2.

cd ~/kernel

3.

wget https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/linux-5.4.84.tar.gz

4. unpack it with

tar -xzf linux-5.4.84.tar.gz

5. download rt\_preempt patch matching the Kernel version we just downloaded over at <http://cdn.kernel.org/pub/linux/kernel/projects/rt/5.4/>

wget http://cdn.kernel.org/pub/linux/kernel/projects/rt/5.4/older/patch-5.4.84-rt47.patch.gz

6. unpack it with

gunzip patch-5.4.84-rt47.patch.gz

Then switch into the linux directory with

cd linux-5.4.84/

and patch the kernel with the realtime patch

patch -p1 < ../patch-5.4.84-rt47.patch

7.

We simply want to use the config of our Ubuntu installation, so we get the Ubuntu config with

cp /boot/config-5.x.x-xxx-generic .config

ex)

cp /boot/config-5.4.0-42-generic .config

---------------------------------------------------not---------------------------------------------------------------------------

(참고 url

1: <https://docs.ros.org/en/foxy/Tutorials/Building-Realtime-rt_preempt-kernel-for-ROS-2.html>  
 2: <https://chenna.me/blog/2020/02/23/how-to-setup-preempt-rt-on-ubuntu-18-04/>

3: <https://blog.naver.com/PostView.nhn?blogId=wnsgnl01&logNo=221732070855> )

1. make menuconfig

|  |
| --- |
| # Enable CONFIG\_PREEMPT\_RT  -> General Setup  -> Preemption Model (Fully Preemptible Kernel (Real-Time))  (X) Fully Preemptible Kernel (Real-Time)  # Enable CONFIG\_HIGH\_RES\_TIMERS  -> General setup  -> Timers subsystem  [\*] High Resolution Timer Support  # Enable CONFIG\_NO\_HZ\_FULL  -> General setup  -> Timers subsystem  -> Timer tick handling (Full dynticks system (tickless))  (X) Full dynticks system (tickless)  # Set CONFIG\_HZ\_1000 (note: this is no longer in the General Setup menu, go back twice)  -> Processor type and features  -> Timer frequency (1000 HZ)  (X) 1000 HZ  # Set CPU\_FREQ\_DEFAULT\_GOV\_PERFORMANCE [=y]  -> Power management and ACPI options  -> CPU Frequency scaling  -> CPU Frequency scaling (CPU\_FREQ [=y])  -> Default CPUFreq governor (<choice> [=y])  (X) performance |

1. make -j99
2. sudo make modules\_install -j99
3. sudo make install -j99
4. sudo reboot
5. sudo groupadd realtime
6. sudo usermod -aG realtime $USER
7. sudo apt-get install cpufrequtils

* **Install base source ( include NVIDIA GPU Driver & NVIDIA Docker)**

1. <https://github.com/ApolloAuto/apollo/blob/266afbf68d83fa6fac7a812ff8a950223f5ab2c0/docs/howto/how_to_install_apollo_kernel.md>

Caution !!!ubuntu-drivers devices

<http://download.nvidia.com/XFree86/Linux-x86_64/460.91.03/>

1. download

NVIDIA-Linux-x86\_64-460.91.03.run

apt-get install gcc make

* IPC 재부팅 후 esc 눌러 Advanced options for Ubuntu => Ubuntu, with 5.4.0- xxx-generic 접속
* 터미널에 uname -r 입력하여 5.4.0-xxx-generic 출력 확인
* ctrl + alt + f1 입력하여 터미널 모드 전환 후 로그인
* sudo service lightdm stop (GUI 비활성화)

systemctl isolate multi-user.target

chmod +x ./NVIDIA-Linux-x86\_64-460.91.03.run

* sudo ./NVIDIA-Linux-x86\_64-375.39.run --no-opengl-files
* Accept => Continue installation => OK => Yes (nvidix-xconfig utility 관련) => OK

nvidia-smi

* sudo reboot now
* 터미널에 nvidia-smi 입력하여 출력 확인 (그림 참고)

rebooting after rt kernel

nvidis-smi => error

# Change to Nvidia driver source directory

cd /usr/src/nvidia-460.91.03

# Build Nvidia driver with IGNORE\_PREEMPT\_RT\_PRESENCE=1

sudo env NV\_VERBOSE=1 \

make -j8 NV\_EXCLUDE\_BUILD\_MODULES='' \

KERNEL\_UNAME=$(uname -r) \

IGNORE\_XEN\_PRESENCE=1 \

IGNORE\_CC\_MISMATCH=1 \

IGNORE\_PREEMPT\_RT\_PRESENCE=1 \

SYSSRC=/lib/modules/$(uname -r)/build \

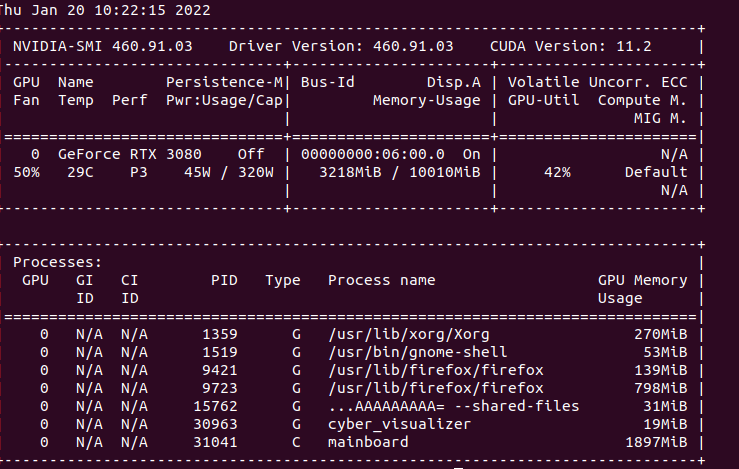
LD=/usr/bin/ld.bfd \

modules

sudo mv \*.ko /lib/modules/$(uname -r)/kernel/drivers/video/

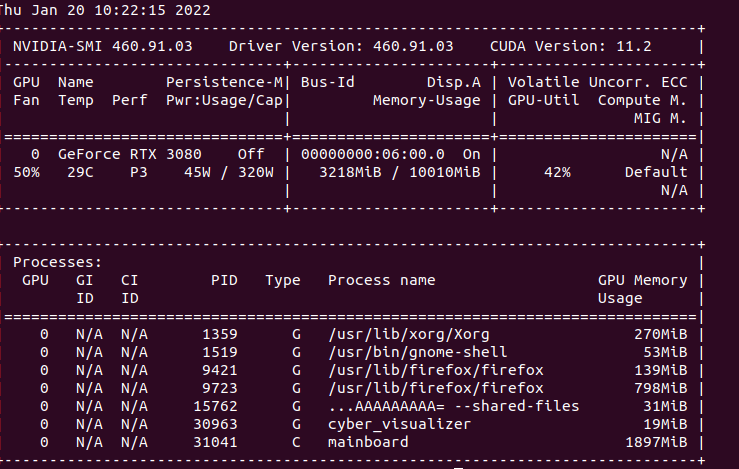
sudo depmod -a

1. sudo reboot now
2. nvidia-smi



1. BIOS 진입 (F2 key)
2. Advanced 탭 => System Agent (SA) Configuration => Graphics Configuration  
   Primary Display: AUTO => PEG로 변경
3. 모니터 연결 케이블을 IPC가 아닌 GPU에 직접 연결 시킨 뒤
4. 터미널에 nvidia-smi 입력하여 출력 확인   
   (드라이버 확인)  
     
   (Nvidia Docker Ref. url: [https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html#docker](https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/install-guide.html" \l "docker))
5. sudo apt-get install curl
6. curl https://get.docker.com | sh
7. sudo systemctl start docker && sudo systemctl enable docker
8. distribution=$(. /etc/os-release;echo $ID$VERSION\_ID)
9. curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -
10. curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list
11. sudo apt-get update
12. sudo apt-get install -y nvidia-docker2
13. sudo systemctl restart docker
14. sudo docker run --rm --gpus all nvidia/cuda:11.0-base nvidia-smi
15. nvidia version 확인

1. cd source
2. cd docker/setup\_host
3. sudo ./setup\_host.sh
4. sudo usermod -aG docker $USER
5. sudo reboot now
6. bash docker/scripts/dev\_start.sh
7. bash docker/scripts/dev\_into.sh
8. nvidia-smi

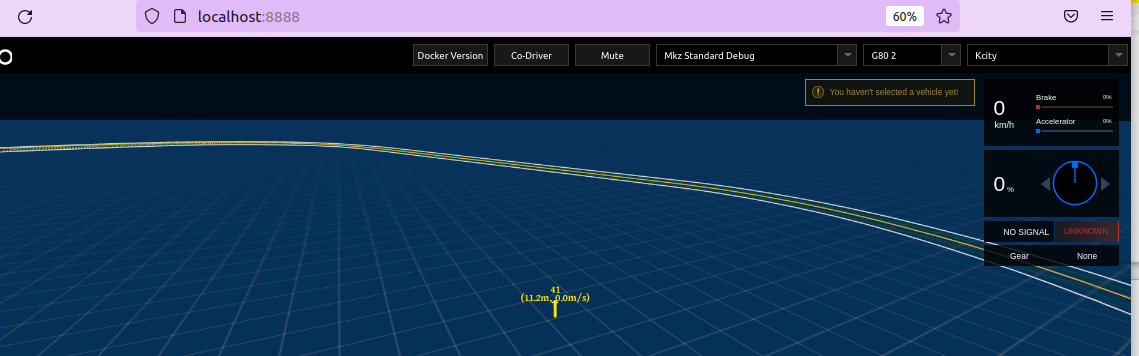


1. sudo chown -R 1000:1000 /apollo
2. bash apollo.sh build\_opt\_gpu

37. execute in docker

bash scripts/bootstrap.sh

<http://localhost:8888/>



swm G801 / map - > seoul

//local perception test

cyber\_launch start ./modules/transform/launch/static\_transform.launch

//gps driver

cyber\_launch start ./modules/drivers/gnss/launch/gnss.launch

//rtk localization

cyber\_launch start ./modules/localization/launch/localization.launch

issue)

1. dev\_into.sh 시 문제 발생
   1. 증상
      1. dev\_start.sh 스크립트 실행 후, dev\_start.sh 스크립트 실행 시 다음과 같은 오류 발생
         1. apollo\_dev\_유저명 의 컨테이너를 찾을 수 없음



* + 1. dev\_start.sh 스크립트 실행 결과 중 다음과 같은 에러메시지가 발생함



* 1. 원인
     1. Install\_nvidia\_docker.sh 스크립트 설치 후 Nvidia-docker2 를 별도로 설치할 경우 설치된 runtime 경로가 변경된것이 적용되지 않아서 생기는 문제
  2. 해결방법
     1. Docker 재시작
        1. sudo systemctl daemon-reload
        2. sudo systemctl restart docker

#### 

#### **※ Appendix**

##### **rc.local activation (Ubuntu 18.04)**

1. .18.04는 rc.local이 활성화 되어 있지 않아 부팅 이후 특정 드라이버를 실행할 필요가
2. 있을 경우 rc.local를 활성화해야 됨(구글에 찾아 보면 진행 절차가 나옴)
3. ..

##### **Novatel Driver & Library**

1. <https://www.novatel.com/support/info/documents/809> 접속
2. USB Drivers => Linux USB Driver (Ver 1.1.0) 다운로드
3. cd Downloads

tar -zxvf

1. cd ngpsusbpackage
2. sudo apt-get install dkms
3. sudo dpkg -i ngpsusb.deb
4. IPC 재부팅 후 터미널에 ls /dev/novatel\* 입력 후 출력 확인

* **Setting Dual IPCs**

git clone https://github.com/richardcochran/linuxptp.git

cd linuxptp

make

# on IPC1: percepiton IPC

sudo ./ptp4l -i eth0 -m &

# on IPC2: planning IPC

sudo ./ptp4l -i eth0 -m -s &

sudo ./phc2sys -a -r &