

ubuntu14.04配置cuda7.5

新手上路，日积月累

1.安装前的准备工作

- 1.1检查自己电脑的**GPU**是否支持
- 1.2检查自己的**linux**的版本属性
- 1.3检查系统**gcc**编译器版本及安装成功与否
- 1.4检查系统是否有与**linux**内核匹配的**Kernel Headers**和**Development Packages**
- 1.5下载合适的**cuda toolkit**和**nvidia-driver-linux**版本
- 1.6安装依赖项
- 1.7卸载残余的驱动

2.先安装英伟达的驱动

- 2.1 进入命令行界面
- 2.2 禁用旧的显卡驱动
- 2.3 安装(注意 参数)

3.安装**cuda toolkit**

- 3.1 运行**run**文件
- 3.2重启电脑，检查**Device Node Verification**
- 3.3设置环境变量
- 3.4重启电脑后查看环境变量

4.检查是否正确的安装

- 4.1检查 **NVIDIA Driver**是否安装成功
- 4.2检查 **CUDA Toolkit**是否安装成功
- 4.3编译**cuda**提供的示例
- 4.4运行编译生成的二进制文件

1.安装前的准备工作

1.1检查自己电脑的**GPU**是否支持

输入指令: `lspci | grep -i nvidia` 检查一下自己电脑的**GPU**的属性，如果没有任何输出则更新**PCI**硬件设备的数据库

输入指令: `sudo update-pciids` 后再运行刚才的命令就会显示信息了。

系统的版本、显卡的类型、**cuda**的版本，最好按照官网的指导书来搭配，并且**nvidia**的驱动版本、**cuda**的版本最好匹配，否则会出现“**cuda driver version is insufficient for cuda runtime version**”这个问题，升级**nvidia**的驱动可以解决这个问题

1.2检查自己的**linux**的版本属性

输入指令: `uname -m && cat /etc/*release` 查看信息

1.3检查系统**gcc**编译器版本及安装成功与否

输入指令: `gcc --version` 查看**gcc**编译器的版本号，如果已经安装成功则会输出版本信息，使用下面方法按照**gcc**

(一般linux系统都已经预装好里gcc)

方法一:

```
sudo apt-get build-dep gcc
```

方法二:

```
sudo apt-get install build-essential
```

参考: <http://www.cnblogs.com/zero1665/archive/2009/11/03/1595510.html>

1.4检查系统是否有与linux内核匹配的Kernel Headers和Development Packages

输入指令: `uname -r` 可以查看当前系统的内核版本

输入指令: `sudo apt-get install linux-headers-$(uname -r)` 安装适合当前内核版本的头文件和开发包

1.5下载合适的cuda toolkit和nvidia-driver-linux版本

最好使用扩展名为.run的包来安装

输入指令: `md5sum <file>` 来检查下载的安装包是否出错, md5号可以在

<http://developer.nvidia.com/cuda-downloads/checksums>

网站上核对, 如果一致则下载的内容无误

1.6安装依赖项

输入下面指令安装依赖项

```
sudo apt-get install freeglut3-dev
sudo apt-get install build-essential
sudo apt-get install libx11-dev
sudo apt-get install libxmu-dev
sudo apt-get install libxi-dev
sudo apt-get install libglu1-mesa
sudo apt-get install libglu1-mesa-dev
```

1.7卸载残余的驱动

如果是第一次安装则可以忽略, 如果遇到循环登陆或者黑屏等问题需要重新安装nvidia-driver和cuda则此步骤是必须的

输入指令: `sudo /usr/local/cuda-X.Y/bin/uninstall_cuda_X.Y.pl` 卸载安装的cuda程序(x, y代表了cuda的版本号, 这里是7.5)

输入指令: `sudo /usr/bin/nvidia-uninstall` 卸载英伟达在linux下的驱动

2.先安装英伟达的驱动

最好先安装linux下的英伟达驱动, 否则会遇到安装完成后循环登陆和桌面侧面栏消失等问题
安装前需要进入text mode模式, 关闭图形界面服务, 禁止nouveau开源驱动

2.1 进入命令行界面

快捷键 **Ctrl+Alt+F1**

输入命令: `sudo stop lightdm`，关闭显示器管理器。否则后面旧的显卡驱动无法禁用，新的显卡驱动无法安装。

输入命令: `sudo apt-get update`，更新apt软件库(尤其是更改软件源后，一定要执行该操作)

2.2 禁用旧的显卡驱动

禁用旧的显卡驱动；切换到/etc/modprobe.d/，新建文件nvidia-installer-disable-nouveau.conf,输入

`blacklist nouveau`

`options nouveau modeset=0`

输入指令: `sudo update-initramfs -u`，使设置生效

输入指令: `lsmod | grep nouveau`，如果该驱动已经禁用则没有输出

2.3 安装(注意 参数)

输入指令: `sudo chmod a+x NVIDIA-Linux-x86_64-370.28.run` 给驱动run文件赋予执行权限

输入指令: `sudo ./NVIDIA-Linux-x86_64-370.28.run -no-x-check -no-nouveau-check -no-opengl-files`

`-no-x-check` 安装驱动时关闭X服务

`-no-nouveau-check` 安装驱动时禁用nouveau

`-no-opengl-files` 只安装驱动文件，不安装OpenGL文件

重启，并不会出现循环登录的问题

安装完成后，输入 `cat /proc/driver/nvidia/version`，可以查看安装是否成功

参

考: <http://www.linuxdiyf.com/linux/26370.html>, <http://www.itnose.net/detail/6332726.html>

3.安装cuda toolkit

3.1 运行run文件

输入指令: `sudo chmod 755 cuda_7.5.18_linux.run`，给予驱动run文件赋予执行权限

输入指令: `sudo ./cuda_7.5.18_linux.run -no-opengl-libs`，开始安装，后面的-no-opengl-libs是防止安装后出现循环登陆问题

在安装选项中的安装驱动的部分一定要选择**no**

安装完成后输入指令: `sudo start lightdm` 来启动图形界面， 快捷键: **Ctrl+Alt+F7**

如果可以正常登陆，那么cuda的安装只剩下添加环境变量了

3.2重启电脑，检查Device Node Verification

检查路径/dev下有无存在名为nvidia（以nvidia开头）的多个文件(device files)

如果没有的话，可以参考官方文档里的指导步骤，进行添加

```
#!/bin/bash
/sbin/modprobe nvidia
if [ "$?" -eq 0 ]; then
# Count the number of NVIDIA controllers found.
NVDEVS=`lspci | grep -i NVIDIA`
N3D=`echo "$NVDEVS" | grep "3D controller" | wc -l`
NVGA=`echo "$NVDEVS" | grep "VGA compatible controller" | wc -l`
N=`expr $N3D + $NVGA - 1`
for i in `seq 0 $N`; do
mknod -m 666 /dev/nvidia$i c 195 $i
done
mknod -m 666 /dev/nvidiactl c 195 255
else
exit 1
fi
/sbin/modprobe nvidia-uvm
if [ "$?" -eq 0 ]; then
# Find out the major device number used by the nvidia-uvm driver
D=`grep nvidia-uvm /proc/devices | awk '{print $1}'`
mknod -m 666 /dev/nvidia-uvm c $D 0
else
exit 1
fi
```

可以在任意的位置创建一个上述内容的脚本，例如起名为**bash**

输入指令：`chmod 755 bash`，赋予文件的执行权限

输入指令：`sudo ./bash`，执行该脚本，然后再进入**/dev**路径下就可以看到包含**nvidia**的文件了

3.3设置环境变量

输入指令：`sudo vim /etc/profile`，在文件末尾添加两行(适合64位的系统)

```
export PATH=/usr/local/cuda-7.5/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-7.5/lib64:$LD_LIBRARY_PATH
```

这样可以永久设置环境变量，如果在终端里运行上述语句，则重启电脑后环境变量会失效

3.4重启电脑后查看环境变量

输入指令：`env | grep cuda`，如果输出中显示了刚才添加的环境变量，则设置成功

（如果不重启电脑，输入：`source /etc/profile`，也可以使环境变量立即生效）

至此**cuda**的安装暂时完成，但一定要做最后的检查，查看**cuda**是否完整安装

4.检查是否正确的安装

4.1检查 NVIDIA Driver是否安装成功

终端输入：`cat /proc/driver/nvidia/version`，会输出**NVIDIA Driver**的版本号

前面在安装**nvidia**驱动时已经检查过，所以这里可以跳过

4.2检查 CUDA Toolkit是否安装成功

终端输入：`nvcc -V` 会输出**CUDA**的版本信息

4.3编译cuda提供的示例

切换到例子存放的路径，默认路径是 `~/NVIDIA_CUDA-7.5_Samples`

输入指令：`make`，开始编译，如果编译不出错则会在该目录下生成一个bin文件夹，用来存放编译后创建的二进制文件

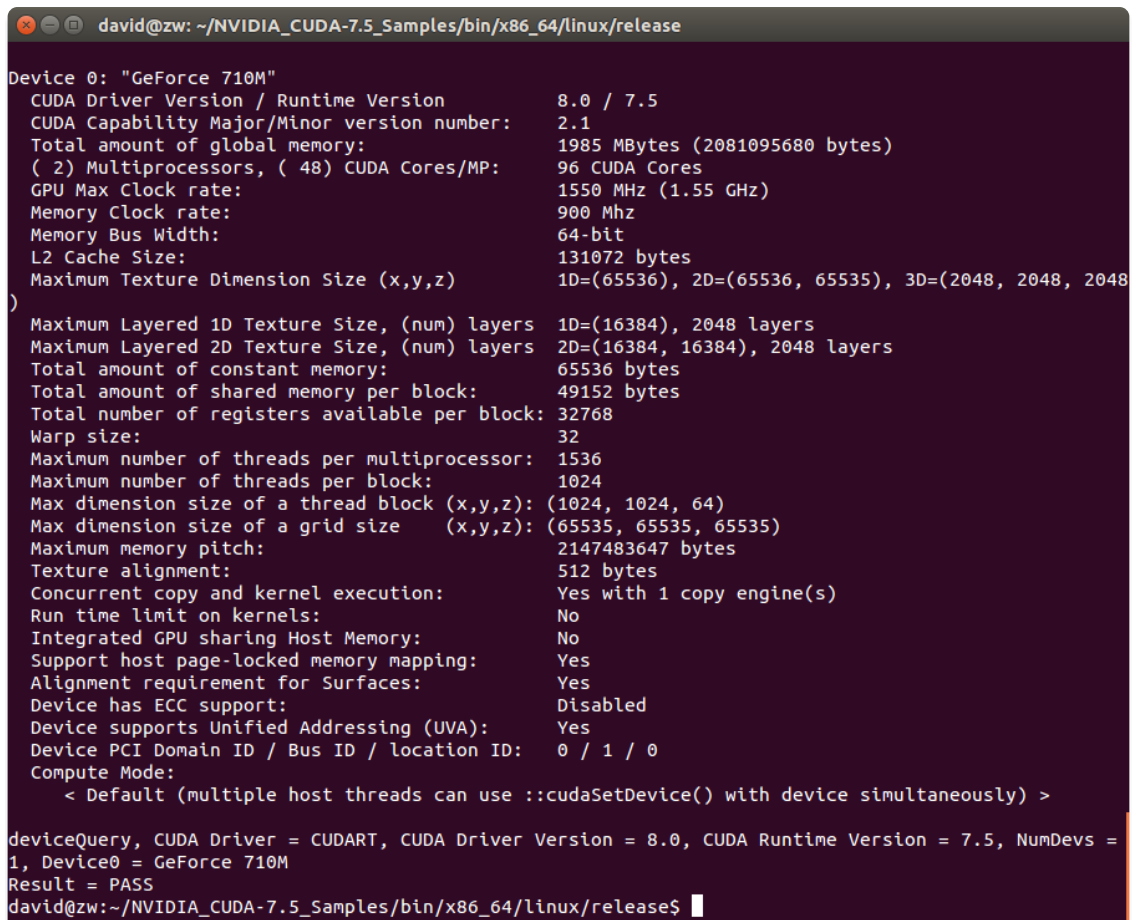
4.4运行编译生成的二进制文件

编译后的二进制文件默认存放在 `~/NVIDIA_CUDA-7.5_Samples/bin` 中

切换路径：`cd ~/NVIDIA_CUDA-7.5_Samples/bin/x86_64/linux/release`

终端输入：`./deviceQuery`

看到类似如下图片中的显示，则代表CUDA安装且配置成功



```
david@zw: ~/NVIDIA_CUDA-7.5_Samples/bin/x86_64/linux/release
Device 0: "GeForce 710M"
  CUDA Driver Version / Runtime Version      8.0 / 7.5
  CUDA Capability Major/Minor version number: 2.1
  Total amount of global memory:             1985 MBytes (2081095680 bytes)
  ( 2) Multiprocessors, ( 48) CUDA Cores/MP: 96 CUDA Cores
  GPU Max Clock rate:                       1550 MHz (1.55 GHz)
  Memory Clock rate:                         900 Mhz
  Memory Bus Width:                         64-bit
  L2 Cache Size:                            131072 bytes
  Maximum Texture Dimension Size (x,y,z)     1D=(65536), 2D=(65536, 65535), 3D=(2048, 2048, 2048)
  Maximum Layered 1D Texture Size, (num) layers 1D=(16384), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers 2D=(16384, 16384), 2048 layers
  Total amount of constant memory:           65536 bytes
  Total amount of shared memory per block:    49152 bytes
  Total number of registers available per block: 32768
  Warp size:                                 32
  Maximum number of threads per multiprocessor: 1536
  Maximum number of threads per block:       1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size (x,y,z): (65535, 65535, 65535)
  Maximum memory pitch:                      2147483647 bytes
  Texture alignment:                         512 bytes
  Concurrent copy and kernel execution:      Yes with 1 copy engine(s)
  Run time limit on kernels:                  No
  Integrated GPU sharing Host Memory:         No
  Support host page-locked memory mapping:    Yes
  Alignment requirement for Surfaces:         Yes
  Device has ECC support:                     Disabled
  Device supports Unified Addressing (UVA):   Yes
  Device PCI Domain ID / Bus ID / location ID: 0 / 1 / 0
  Compute Mode:
    < Default (multiple host threads can use ::cudaSetDevice() with device simultaneously) >
deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 8.0, CUDA Runtime Version = 7.5, NumDevs = 1, Device0 = GeForce 710M
Result = PASS
david@zw:~/NVIDIA_CUDA-7.5_Samples/bin/x86_64/linux/release$
```

再检查一下系统和CUDA-Capable device的连接情况

输入指令：`./bandwidthTest`

看到类似如下图片中的显示，则代表成功

```

david@zw: ~/NVIDIA_CUDA-7.5_Samples/bin/x86_64/linux/release
cdpSimpleQuicksort      histEqualizationNPP    oceanFFT                simpleHyperQ
clock                   histogram              p2pBandwidthLatencyTest simpleIPC
clock_nvrtc             HSOpticalFlow         particles               simpleLayeredTexture
concurrentKernels       imageDenoising         postProcessGL           simpleMultiCopy
conjugateGradient       inlinePTX              ptxjit                 simpleMultiGPU
conjugateGradientPrecond inlinePTX_nvrtc         quasirandomGenerator    simpleOccupancy
conjugateGradientUM      interval              quasirandomGenerator_nvrtc simpleP2P
convolutionFFT2D         jpegNPP                radixSortThrust         simplePitchLinearText
convolutionSeparable     libcuhook.so.1        randomFog               simplePrintf
convolutionTexture       lineOfSight            recursiveGaussian        simpleSeparateCompila
cppIntegration           Mandelbrot             reduction               simpleStreams
david@zw:~/NVIDIA_CUDA-7.5_Samples/bin/x86_64/linux/release$ ./bandwidthTest
[CUDA Bandwidth Test] - Starting...
Running on...

Device 0: GeForce 710M
Quick Mode

Host to Device Bandwidth, 1 Device(s)
PINNED Memory Transfers
  Transfer Size (Bytes)      Bandwidth(MB/s)
  33554432                   2841.7

Device to Host Bandwidth, 1 Device(s)
PINNED Memory Transfers
  Transfer Size (Bytes)      Bandwidth(MB/s)
  33554432                   3282.8

Device to Device Bandwidth, 1 Device(s)
PINNED Memory Transfers
  Transfer Size (Bytes)      Bandwidth(MB/s)
  33554432                   12537.1

Result = PASS

NOTE: The CUDA Samples are not meant for performance measurements. Results may vary when GPU Boost i
david@zw:~/NVIDIA_CUDA-7.5_Samples/bin/x86_64/linux/release$

```

到此为止，**cuda7.5**安装完成