
```
NVIDIA CUDA
Linux Release Notes
Version 1.1
```

On some Linux releases, due to a GRUB bug in the handling of upper memory and a default vmalloc too small on 32-bit systems, it may be necessary to pass this information to the bootloader:

```
vmalloc=256MB, uppermem=524288
```

Example of grub conf:

```
title Red Hat Desktop (2.6.9-42.ELsmp)
root (hd0,0)
uppermem 524288
kernel /vmlinuz-2.6.9-42.ELsmp ro root=LABEL=/1 rhgb quiet vmalloc=256MB
pci=nommmconf
initrd /initrd-2.6.9-42.ELsmp.img
```

New Features

Platform Support

- o New distributions supported
 - Fedora 7
 - Red Hat Enterprise Linux 3.9
 - Red Hat Enterprise Linux 4.5
 - SUSE Linux Enterprise Desktop - Service Pack 1
 - Ubuntu-7.04

Hardware Support

- o Additional hardware support:
 - Tesla C870
 - Tesla D870
 - Tesla S870

 - Quadro FX 1700
 - Quadro FX 570
 - Quadro FX 370
 - Quadro NVS 130M
 - Quadro NVS 135M
 - Quadro NVS 140M
 - Quadro NVS 290
 - Quadro NVS 320M
 - Quadro FX 1600M
 - Quadro FX 570M
 - Quadro FX 360M
 - Quadro Plex 1000 Model IV
 - Quadro Plex 1000 Model S4

 - GeForce 8800 GT
 - GeForce 8400 GS
 - GeForce 8800M GTX
 - GeForce 8800M GTS
 - GeForce 8700M GT
 - GeForce 8600M GT
 - GeForce 8600M GS
 - GeForce 8400M GT
 - GeForce 8400M GS
 - GeForce 8400M G

Compiler and Toolchain

- o `__global__` functions are now always mangled. When using the driver API developers must declare these with `'extern "C"'`.
- o device emulation mode no longer requires `libcuda.so`.
- o C++ language support

- host code only, alpha quality
- developers must "opt in" with the --host-compilation=C++ option
- o Added device functions:
 - __clz
 - __clzll
 - __ffs
 - __ffsll
 - __sad
 - __float2ll_rz
 - __float2ll_rn
 - __float2ull_rz
 - __ll2float_rn
 - __ull2float_rn
 - __usad
- o Removed the float-valued atomicCAS device function

Mathematical Functions

- o Added math functions:
 - llrintf()
 - llroundf()
- o Improved accuracy of the functions:
 - cosf
 - powf
 - sinf
 - sincosf
 - tanf

Asynchronous Memcpy Support

- o Streams and Asynchronous memcpy functions
 - CPU/GPU concurrency and compute/host<->device memcpy concurrency
- o Events (enable high-precision timing and CPU/GPU synchronization)

Miscellaneous

- o New cuDeviceGetAttribute function obsoletes cuDeviceGetProperties
 - Also enables applications to query whether the hardware can process and copy data concurrently

CUFFT Library

- o Performance improvements

CUBLAS Library

- o Performance improvements
- o Added the functions:
 - cublasCrot()
 - cublasCrotg()
 - cublasCsrot()

Major Bug Fixes

- o Fixed OpenGL interoperability on multi-GPU hardware configurations

Known Issues

- o Individual GPU program launches are limited to a run time of less than 5 seconds on a GPU with a display attached. Exceeding this time limit causes a launch failure reported through the CUDA driver or the CUDA runtime. GPUs without a display attached are not subject to the 5 second run time restriction. For this reason it is recommended that CUDA is run on a GPU that is NOT attached to an X display.
- o In order to run CUDA applications, the CUDA module must be loaded and the entries in /dev created. This may be achieved by initializing X Windows, or by creating a script to load the kernel module and create the entries.

An example script (to be run at boot time):

```
#!/bin/bash

modprobe nvidia

if [ "$?" -eq 0 ]; then

# Count the number of NVIDIA controllers found.
N3D=`/sbin/lspci | grep -i NVIDIA | grep "3D controller" | wc -l`
NVGA=`/sbin/lspci | grep -i NVIDIA | 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
```

- o When compiling GCC, special care must be taken for structs that contain 64-bit integers. This is because GCC aligns long longs to a 4 byte boundary by default, while NVCC aligns long longs to an 8 byte boundary by default. Thus, when using GCC to compile a file that has a struct/union, users must give the `-malign-double` option to GCC. When using NVCC, this option is automatically passed to GCC.
- o On systems with multiple GPUs installed or systems with multiple monitors connected to a single GPU, OpenGL interoperability always copies shared buffers through host memory.
- o Current hardware limits the number of asynchronous memcopies that can be overlapped with kernel execution. Overlap is also limited to kernels executing for less than 1 second. These limitations are expected to improve on future hardware.
- o The following APIs exhibit high CPU utilization if they wait for the hardware for a significant amount of time. As a workaround, apps may use `cu(da)StreamQuery` and/or `cu(da)EventQuery` to check whether the GPU is busy and yield the thread as desired.
 - `cuCtxSynchronize`
 - `cuEventSynchronize`
 - `cuStreamSynchronize`
 - `cudaThreadSynchronize`
 - `cudaEventSynchronize`
 - `cudaStreamSynchronize`
- o When the profiler gathers performance signals on G80-based products, the driver reduces the clock rate on the device. If the CUDA app crashes or otherwise exits uncleanly, the clocks will not be reset to their previous values. The system must be rebooted to restore the original clock rate.

Open64 Sources

The Open64 source files are controlled under terms of the GPL license. Current and previously released versions are located via anonymous ftp at download.nvidia.com in the CUDAOpen64 directory.

Revision History

11/2007 - Version 1.1
06/2007 - Version 1.0

06/2007 - Version 0.9
02/2007 - Version 0.8 - Initial public Beta

More Information

For more information and help with CUDA, please visit
<http://www.nvidia.com/cuda>