What is rabbit?

rabbit.engr.oregonstate.edu

PCIe Bus
2 E5-2630 Xeon Processors
16 Cores total
64 GB of memory
2 TB of disk

NVIDIA Titan Black
PCIe Bus
15 SMs
2880 CUDA cores
6 GB of memory
OpenCL support
CUDA support
What is rabbit?

rabbit lives in a rack in our server room in the Kelley Engineering Center:

rabbit 151% lscpu
Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
CPU(s): 32
On-line CPU(s) list: 0-31
Thread(s) per core: 2
Core(s) per socket: 8
Socket(s): 2
NUMA node(s): 2
Vendor ID: GenuineIntel
CPU family: 6
Model: 63
Stepping: 2
CPU MHz: 2399.982
BogoMIPS: 4799.30
Virtualization: VT-x
L1d cache: 32K
L1i cache: 32K
L2 cache: 256K
L3 cache: 20480K
NUMA node0 CPU(s): 0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30
NUMA node1 CPU(s): 1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31

2 TB of disk
What is *rabbit*?

- **NVIDIA Titan-Black**
- **CPU**
- **PCIe bus and riser**
- **management port**
- **ports**
- **network**

Additional images show the *rabbit* setup with various hardware components labeled.
Getting to *rabbit* and setting up your account

**To login to rabbit:**

```
ssh rabbit.engr.oregonstate.edu -l yourcrawlerusername
```

**Put this in your *rabbit* account’s `.cshrc`:**

```
setenv INTEL_LICENSE_FILE    28518@linlic.engr.oregonstate.edu
setenv ICCPATH /nfs/guille/a2/rh80apps/intel/studio.2013-sp1/composer_xe_2015/bin/
set path= ( $path $ICCPATH )
source /nfs/guille/a2/rh80apps/intel/studio.2013-sp1/bin/iccvars.csh intel64
setenv CUDA_PATH /usr/local/apps/cuda/cuda-9.2
setenv LD_LIBRARY_PATH $CUDA_PATH/lib64:$LD_LIBRARY_PATH
set path = ( $path $CUDA_PATH/bin )
```

**Then activate these values like this:**

```
source .cshrc
```

(These will be activated automatically the next time you login.)

Compiling and running C/C++ on *rabbit*

```
icpc -o try try.cpp -lm -openmp -align -qopt-report=3 -qopt-report-phase=vec
or
g++ -o try try.cpp -lm -fopenmp
```
Compiling for OpenCL

printinfo: printinfo.cpp
icpc -o printinfo printinfo.cpp /usr/lib64/libOpenCL.so -lm -openmp

Compiling for CUDA

arrayMul: arrayMul.cu
nvcc -o arrayMul arrayMul.cu
**The printinfo Program Output**

Number of Platforms = 1
Platform #0:
   Name = 'NVIDIA CUDA'
   Vendor = 'NVIDIA Corporation'
   Version = 'OpenCL 1.1 CUDA 7.0.18'
   Profile = 'FULL_PROFILE'
Device #0:
   Type = 0x0004 = CL_DEVICE_TYPE_GPU
   Device Vendor ID = 0x10de (NVIDIA)
   Device Maximum Compute Units = 15
   Device Maximum Work Item Dimensions = 3
   Device Maximum Work Item Sizes = 1024 x 1024 x 64
   Device Maximum Work Group Size = 1024
   Device Maximum Clock Frequency = 1071 MHz

Device Extensions:
   cl_khr_byte_addressable_store
   cl_khr_icd
   cl_khr_gl_sharing
   cl_nv_compiler_options
   cl_nv_device_attribute_query
   cl_nvPragma_unroll
   cl_nv_copy_opts
   cl_khr_global_int32_base_atomics
   cl_khr_global_int32_extended_atomics
   cl_khr_local_int32_base_atomics
   cl_khr_local_int32_extended_atomics
   cl_khr_fp64

\[15 \times 192 = 2880\] CUDA cores!

**Reservation System – Please use It!!**

https://secure.engr.oregonstate.edu/engr/resources/bailey