Notes on CUDA practicals on “skynet” cluster

Prof. Mike Giles

1 Head and compute nodes

The head node of the skynet cluster is

- name: skynet.osc.ox.ac.uk
- IP address: 192.168.50.1

This can be accessed from machines on the university network through an SSH command:

```
ssh -X skynet.osc.ox.ac.uk
```

or

```
ssh -X username@skynet.osc.ox.ac.uk
```

if the username on `skynet` is different.

This head node does not have any NVIDIA graphics cards and so it can only be used to compile CUDA codes and run them using the emulation mode. For native execution of CUDA code one has to use one of the eight compute nodes, each of which has

- 2 quad-core Intel Xeons and 8GB main memory
- 2 C1060 cards (one half of a Tesla S1070) each with 240 cores and 4GB of graphics memory
- Gigabit Ethernet and Infiniband networking (for multi-node HPC applications)

To use one of the compute nodes, you can either follow the instructions on the `skynet` help pages to submit a batch job using `qsub`, or you can create an interactive session on one of the compute nodes by issuing the command:

```
    qsub -IVX
```

You can then edit and compile your code as usual, and run it on one of the two GPUs attached to each compute node.

Using an interactive session is appropriate when developing codes using test runs which last no more than a few seconds. When doing "production" runs you should use the batch submission system.

2 CUDA version 2.3

To use CUDA version 2.3 you must enter the commands:

```
module add cuda/2.3
module initadd cuda/2.3
```

You only have to do this once; it will remain set up correctly the next time you log in. Amongst other things, this will set your `PATH` environment variable so that you can use the NVIDIA compiler `nvcc` and access the relevant header files.

The SDK (software development kit) is located at `/opt/cuda/2.3/sdk`. Its `C/src` subdirectory has lots of useful example codes, and the `C/common` subdirectory has various useful utilities, header files and libraries.

3 Editing

`emacs`, `vi` and `gedit` are all available on `skynet`. Windows users who are unused to Linux editors may prefer `gedit` which can be launched in its own window using the command:

```
    gedit &
```

Using `gedit` you can open and save files by clicking on icons, so it's very easy to use, though it doesn't have the advanced features of `emacs`.

4 Makefile

The CUDA SDK comes with a "master" Makefile called `common.mk`. The user's Makefile references this and specifies various files to be compiled, identifying which are CUDA files and which are regular C++ files, and setting various compiler flags.

To execute the Makefile there are 4 options

- `make` creates a standard CUDA executable
- `make dbg=1` creates an executable with error-checking enabled
- `make emu=1` creates an executable to be run under emulation (on the CPU)
- `make emu=1 dbg=1` creates an emulation executable with error-checking

The executables usually get put in subdirectories called `../../bin/linux/release, ../../bin/linux/debug,` etc., but a modification to the user's Makefile puts them instead in `bin/release, bin/debug,` etc., as subdirectories of the directory holding the Makefile and the source files.

Finally, the command

```
make -n
```

(which can be combined, if wanted, with `dbg=1` and/or `emu=1`) is helpful in showing what `make` would do if the `-n` flag were omitted. This shows how it compiled each of the files into object files (using `nvcc` for the CUDA files and, usually, `gcc/g++` for the plain C/C++ files, and then linking them all together with the relevant libraries to form the executable.

5 File transfer and printing

Files should be transferred using `scp` to your home system (which is in the Thom building for the CUDA course) for local printing. You could create a `ssh-print` script on `skynet` to copy the file to your home system and print it.