An Introduction to HPC and Scientific Computing

CWM, Department of Engineering Science

University of Oxford

Karel Adámek

Practical 6: Introduction to the CUDA programming language

This practical will review examples showed in the lecture. First example is 'Hello world' example which is then expanded to demonstrate scheduling of threads, warps and blocks on the GPU. Second example it kernel for vector addition, where importance of correct memory access is demonstrated. The practical ends by writing a code which calculates decimation of a series of integer numbers.

The learning outcomes of this practical are:

- To be become familiar with code structure when using GPUs.
- To know how to write kernel and configure them.
- To understand scheduling of thread, warp and block execution of GPUs.

All practicals for this course will be carried out on the Universities ARCUS-B computer. To understand how to use ARCUS-B see the slides from lecture 3. As a reminder log in using ssh as follows:

ssh –CX teachingXY@arcus-b.arc.ox.ac.uk

Where teachingXY is the account that we have issued you with.

When logged into the Arcus-B head node, you can create an interactive session on one of the K80 GPU compute nodes by issuing the following command:

export SALLOC_RESERVATION=cuda-openmp-thu

salloc -pgpu --ntasks-per-node=1 srun --pty --x11 --preserve-env /bin/bash -l

and once you are then put onto one of the K80 nodes, issue the commands

module load gpu/cuda

and

export CUDA_VISIBLE_DEVICES=0,1,2,3

If you have not done so clone the github repo for this CWM. To do this, at the command prompt type:

$ git clone https://github.com/wesarmour/CWM-in-HPC-and-Scientific-Computing.git

Or

```
$git pull
```

To update your local repo.

Instructions for this practical

The first part of this practical is to review examples given on the lecture. These are in 'code' directory of this practical. These examples are

1. helloworld,
2. helloworld_scheduling,
3. vector_addition,
4. vector_addition_memory.

Please follow instructions contained in the code files themselves.

Decimation

Second part of the practical is to write a code which calculates decimation of a series of integer numbers.

Decimation is where we divide a series of numbers into distinct series of sets which contain two neighbouring numbers. These number are then added together to create a new series. The algorithm is shown in the figure 1.

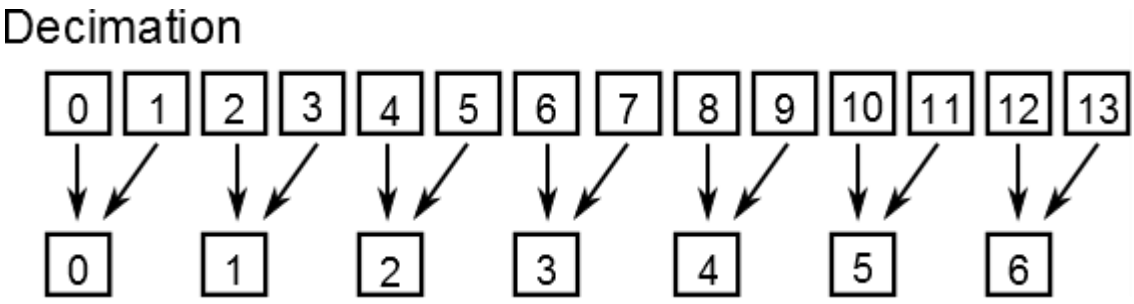

Figure 1: Decimation, two neighbouring numbers are added together to create new series of numbers.

For example if we have a series of numbers {1,2,5,7,3,8} the decimation will perform {1+2, 5+7, 3+8}

$$\{1+2\,,5+7\,,3+8\} => \{3,12,11\}.$$

The code with description and specific tasks which needs to be completed are describe in the code itself.