ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

High Performance Computing for
Science and Engineering II
Spring semester 2015

M. Troyer
ETH Zürich, HIT G 31.8
CH-8093 Zürich

Set 9 - GPUs II

Issued: May 11, 2015
Hand in: May 18, 2015

Question 1: Diffusion on GPUs - Part II

In this exercise we will further improve the 2D diffusion code on GPUs.

a) Write a kernel for `GetMoment()` to further reduce the memory transfer needed between GPU and CPU by calculating (at least) paritial sums on the GPU.
Hint: An important part of the operation is a reduction.
While you can spend a full talk on how to optimize reductions[1], a simple GPU reduction will do here. It is called only every 100th update and will not dominate the execution time of our code. A simple reduction scheme within a block using the shared memory can be achieved by selecting only specific thread indices:

```
__shared__ float data[8];

...
if(threadIdx.x < 4)
    data[threadIdx.x] += data[threadIdx.x + 4];
__syncthreads();
if(threadIdx.x < 2)
    data[threadIdx.x] += data[threadIdx.x + 2];
__syncthreads();
if(threadIdx.x < 1)
    data[threadIdx.x] += data[threadIdx.x + 1];
```

b) Think about good choices for blocksPerGrid and threadsPerBlock of your kernels and explain your choice.

Summary

Summarize your answers, results and plots into a short PDF document. Furthermore, elucidate the main structure of the code and report possible code details that are relevant in terms of accuracy or performance. Send the PDF document and source code to your assigned teaching assistant.

[1] M. Harris, Optimizing Parallel Reduction in CUDA, 2007,
`http://docs.nvidia.com/cuda/samples/6_Advanced/reduction/doc/reduction.pdf`