GPU Programming 2018/19

Assignment 3

Due: 25/01/2019, 22:00

In this assignment, we implement the Mergesort algorithm in CUDA.

Your tasks are as follows:

1.) Download the skeleton code and generate the build system using `cmake`.

2.) Implement the function `input()` so that it generate an array of random integers with size 2^k and computes a reference solution. (1 Point)

3.) Complete `mergeSort1()` and `merge_device1()` so that they implement the Mergesort algorithm using only global memory for up to $k = 20$. Your implementation should use a number of threads that is appropriate for the chosen input size and that exploits the available parallelism on the device. (3 Points)

4.) Complete `mergeSort2()` and `merge_device2()` so that they implement the Mergesort algorithm using shared memory. You might need to use `merge_device1()` due to shared memory limitations. Document the speed up that this version provides compared to the global memory one with a graph where the performance of both is plotted as a function of n. (3 Points)

Bonus: Improve the performance of the implementation so that one obtains a speed up of (at least) 50%. (3 Points)

Submission: Please submit your implementation as well as the graphs before the deadline to gpucourse@isg.cs.ovgu.de. Your code has to compile and run with the given CMake file.

Plagarism policy: You are free to discuss the assignment with your classmates and consult available resources (books, websites, ...). However, every student writes her / his own implementation. Violations are handled according to the study regulations.