Technische Universität Kaiserslautern
Fachbereich Informatik
RHRK
Dr. habil. Josef Schüle

# High Performance Computing with GPUs
## Exercise Sheet 2

*Part a: Get information about the used device*
To adapt code to the underlying specific hardware, an exact knowledge of the used GPGPU is mandatory. This knowledge may easily be gained via the function `cudaGetDeviceProperties` in the CUDA API. Search for meaningful properties in the Web and use this function to get information on the hardware used. Use this functionality later in all your exercises.
Try to find out the number of Streaming Multiprocessors the used card has and the number of ALUs per SM. Calculate the peak memory bandwidth according to

$$peak\_Bandwidth = \frac{memory\_clock * bus\_width}{bits\_per\_byte} * 2(\text{DRAM rate})$$

and compare it with information on the specific card. Calculate as well the peak performance for floating point arithmetic (using floats) according to

$$peak\_Performance = clock\_rate_SM * 2\_Ops * number\_SM * cores\_per\_SM \ .$$

*Part b: Performance and Occupancy*
In the file `skeleton.cu` you find 3 kernels and a setup for measuring performance of a GPU. In kernel `kern_A` you may change the number of arithmetic operations (just duplicate the line for the operation and change the Macro `NUMADDS` accordingly, in kernel `kernel_M` the number of memory accesses may be changed together with the Macro `NUMMEM` and finally `kern_C` allows a mix of memory and arithmetic operations (Macro `LNum` for the number of repetitions, `NUMADDS` for the number of arithmetic operations within each repetition and `NUMMEM=1`). Besides of changing the number of operations you may change the number of threads and blocks when calling the kernel.
With help of this skeleton prepare the following sequences and depict the results graphically:

(1) Fixed number of operations. Change the occupancy of the card (A and M).
(2) In between 25% and 50% occupancy. Change the number of operations in the kernels (A and M).
(3) 100% occupancy. Change the number of operations in the kernels (A and M).
(4) 50% occupancy and `kernel_C`. Use `LNum=4` and change `NUMADDS` from 1 to 32.

Any ideas to explain the behavior are welcome. Compare the results with the peak values from part a.
Additional hint: The compiler `nvcc` provides the option `-Xptxas=-v`. It reveals additional information like the number of registers used per kernel.