Technische Universität Berlin

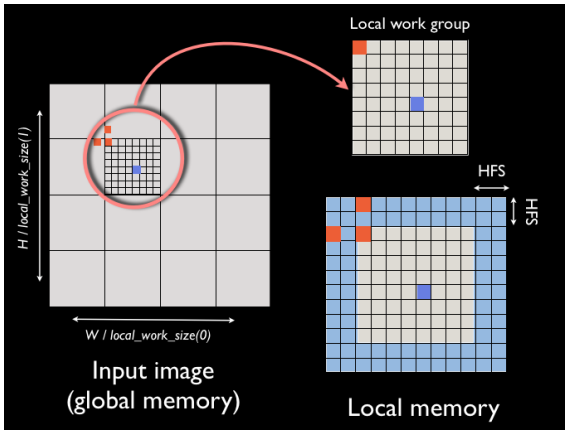
AES Embedded Systems Architecture

Master Project

Compiler-based Exploitation of Local Memory

Description

GPU programming models distinguish between different types of memory that are exposed to the programmer. *Local Memory* is one of them. Because it is on-chip it can be accessed with a much lower latency compared to the off-chip global memory. A distinct property of local memory is that the memory is shared among all threads in a work group. If there is *data reuse* across threads, this type of memory can be used to accelerate the execution of a program.



Loading data from global memory to local memory (`https://www.evl.uic.edu/kreda/gpu/image-convolution/`)

Threads in a work group can use local memory for *cooperative* loading of data from global memory to local memory. This is a well-known optimization technique and allows subsequent processing steps to work on the faster local memory instead on the slower global memory. However, usage and partitioning of local memory is up to the programmer and often a non-trivial task.

Objective

In this master project possibilities of compiler-based automatic utilization of local memory shall be explored. There are two major objectives:

1. Exploration of possibilities to recognize input data that is eligible to be prefetched to local memory in order to accelerate the subsequent execution of the kernel.
2. Transformation of kernels to make use of local memory, and exploration of local memory configurations.

Required Skills

- profound C++ experience
- intermediate compiler construction skills
- basic knowledge of the OpenCL/CUDA programming model
- preferable LLVM development experience

References

Fang et al. Grover: Looking for Performance Improvement by Disabling Local Memory Usage in OpenCL Kernels. 2014.

Contact Person

Daniel Maier <daniel.maier@tu-berlin.de>

January 2018