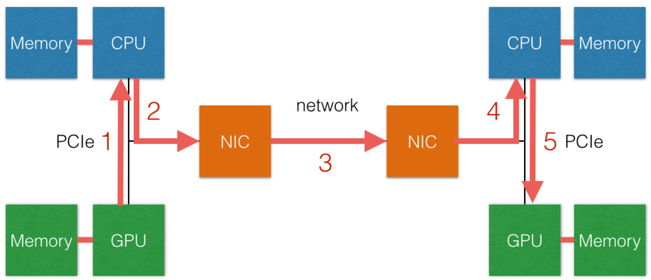
Energy-efficient Distributed GPU Communication

Benjamin Klenk*, Lena Oden†, Holger Fröning*
*University of Heidelberg, Institute of Computer Engineering, {benjamin.klenk, holger.froening}@ziti.uni-heidelberg.de
† Fraunhofer Institute for Industrial Mathematics, oden@fhg.itwm.de

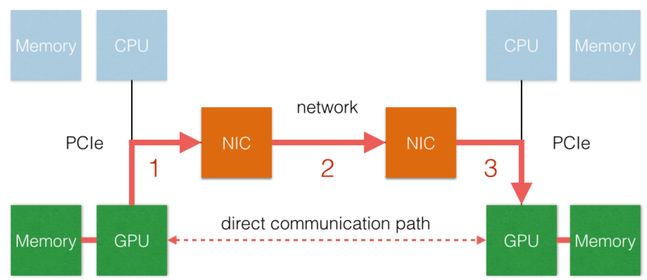
UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

GPU-centric Communication

Cluster systems increasingly deploy accelerators like GPUs to stay within a given power budget, limited by the end of Dennard Scaling. Since then, various applications have been ported to CUDA to make use of the computational power of GPUs. However, communication is still crucial, both in terms of performance but also energy efficiency.
Communication between GPUs is done by traditional CPU-tailored communication methods, such as the Message Passing Interface (MPI). This is shown here:



In order to move data, the GPU has to copy the data to the CPU first (1). Next, the data is sent to the target CPU (2,3,4) and copied to the target GPU memory (5).
We already introduced a direct communication method, avoiding any interference with the CPU. The CPU is entirely bypassed and control flow can stay on the GPU for the whole application. Context switches are no longer necessary, resulting in savings regarding energy and time. The approach is shown here:

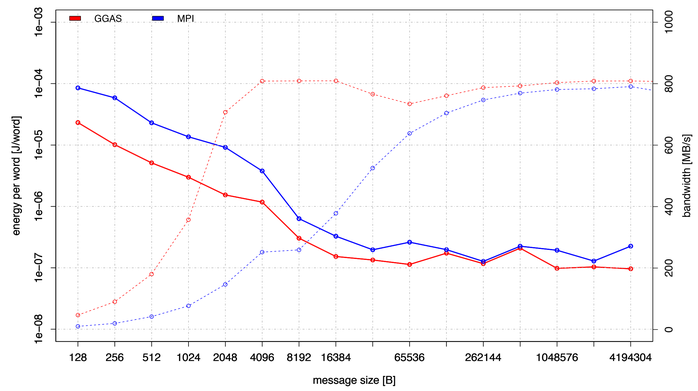


The GPU is able to trigger communication by controlling directly the Network Interface Controller (NIC) (1). On the target side, the NIC writes the data into the GPU memory without any staging copies (2).
Currently, we have implemented two approaches of direct GPU-GPU communication:

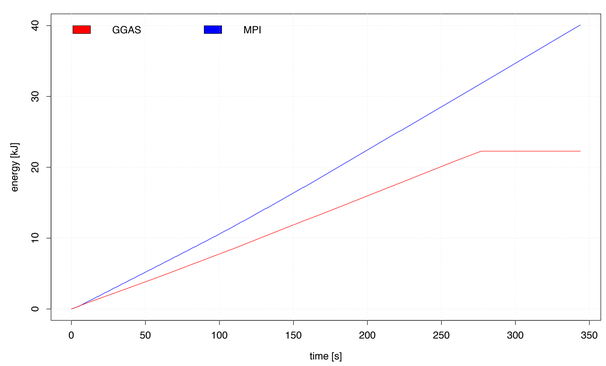
- GGAS: inline with the execution model of GPUs, the data is sent collaboratively by all threads. The NIC forwards memory operations like loads and stores and completes them on the target side. [1]
- GPU RMA: a Put/Get model, whereby one thread creates a Work Request (WR) and the NIC handles the communication. The GPU is released from sourcing the data and can continue with computations. [2]

Energy Efficiency

Energy consumption has become a major aspect of today's system, reasoning the rise of GPUs in the HPC world. With our communication method, power can be saved by enabling the CPU to be idle during the communication. With MPI, the CPU is still needed to handle communication, avoiding entering power saving states. Furthermore, time is also saved by achieving higher bandwidth. Following graph shows the energy per transferred word using GGAS compared to MPI.



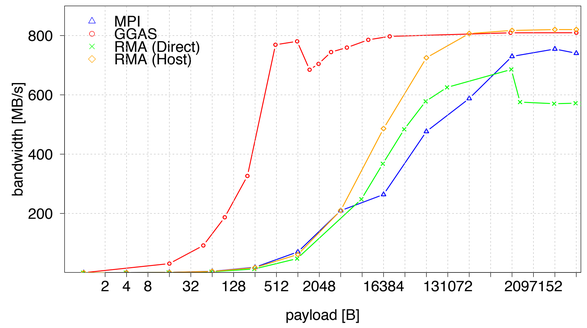
As can be seen, the energy per word is always superior for GGAS because of two reasons: less power consumption while being also faster.
In another work, we implemented the reduce/allreduce operation with GGAS. Using this, we showed that the energy consumption for the global sum benchmark, which calculates the global sum of a given array, is significantly lower than using MPI as a general-purpose communication model. The energy results are presented in the following figure.



For this benchmark, only small messages are exchanged. Therefore, GGAS is superior in performance, while saving power at the same time. This leads to remarkable energy savings. More on this can be found in [3].

Performance Results

The following graph shows the bandwidth results of direct GPU-GPU communication methods, compared to state-of-the-art MPI+CUDA communication.



Note that we use the EXTOLL interconnect based on an FPGA implementation (157 MHz, 64 Bit data paths).
As can be seen, GGAS achieves a remarkable bandwidth, even for small message sizes. RMA (Direct) performs superior than MPI, but because of the GPUDirect RDMA issue [5], the bandwidth drops significantly for larger messages.
In addition to bandwidth, we implemented a barrier and collective reduce operation with GGAS. More information on this can be found in [1][3].

Future Work

We are going to analyze performance and energy consumption on application level, using different communication methods. Furthermore, we plan to implement a communication library that provides suitable abstractions.

References

[1] L. Oden and H. Fröning, "GGAS: Global GPU address spaces for efficient communication in heterogeneous clusters," in IEEE Cluster, 2013.
[2] B. Klenk, L. Oden, and H. Fröning, "Analyzing put/get apis for thread-collaborative processors," in *HUCAA Workshop in conjunction with ICPP, Minneapolis, MN, USA*, 2014.
[3] L. Oden, B. Klenk and H. Fröning, "Energy-efficient collective reduce and allreduce operations on distributed gpus," in International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2014.
[4] B. Klenk, L.Oden and H. Fröning, "GPU-centric communication for improved efficiency", in International Workshop on Green Programming, Computing and Data Processing (GPCDP) in conjunction with International Green Computing Conference (IGCC), Dallas, TX, USA, 2014
[5] R. A. et al., "GPU peer-to-peer techniques applied to a cluster interconnect," in Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2013 IEEE International Symposium on, 2013.
[6] B. Klenk, L. Oden, H. Fröning, "Analyzing Communication Models for Distributed Thread-Collaborative Processors in Terms of Energy and Time", in International Symposium on Performance Analysis of Systems and Software (ISPASS), Philadelphia, PA, 2015

Questions?