A High-Throughput Readout Architecture based on PCIe Gen3 and DirectGMA Technology

M. Caselle, M. Balzer, S. Chilingaryan, T. Dritschler, A. Kopmann, L. Rota, M. Vogelgesang, N. Zilio, M. Weber

Karlsruhe Institute of Technology – author email: michele.caselle@kit.edu

TWEPP 2015
Topical Workshop on Electronics for Particle Physics
28 September - 02 October 2015
IST - Lisbon - Portugal

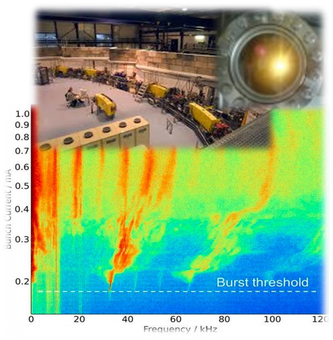
Motivation

The Institute for Data Processing and Electronics (IPE) is involved in the commissioning of several experimental setups that require on-line processing in the range of GB/s. Because of their parallel architecture, GPUs can efficiently process compute-intensive work-loads and are used to achieve real-time data analysis.

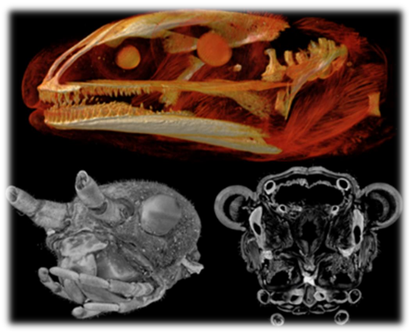
However, the communication between GPUs and other devices limit the performance of these systems. To reduce this constrain, we have developed a custom architecture that enables fast communication between FPGA, CPUs and GPUs via direct memory access (DMA).

Examples for Data-intensive applications:

Particle physics: CMS track trigger



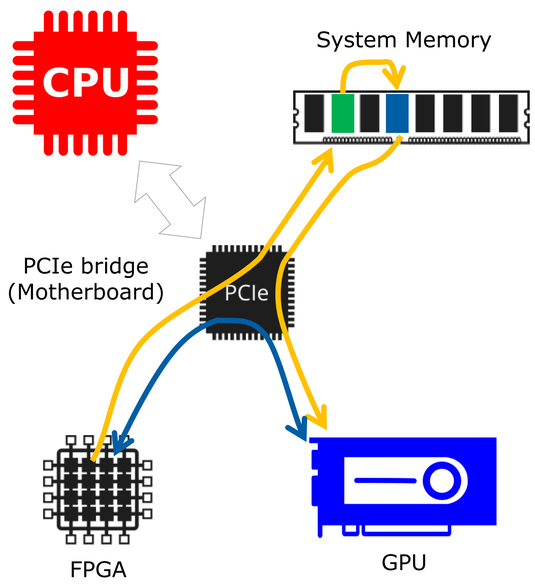
Photon science: on-line beam diagnostics



Life science / medical applications

Basic Concepts

PCI-Express is the standard communication bus for GPUs, with a theoretical bandwidth of up to 15.75 GB/s for Gen 3 x16.



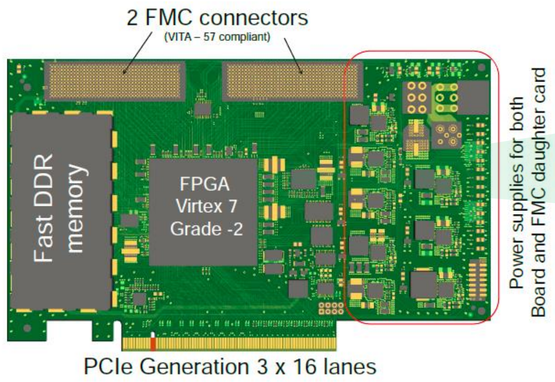
In a traditional DMA architecture, data is first written to the main system memory and then sent to the GPUs for final processing. The main memory is involved in a certain number of read/write operations, depending on the specific HW or SW implementation. The total throughput of the system is therefore limited by the memory bandwidth.

Using direct GPU communication, the DMA engine has direct access to the GPU memory, therefore drastically decreasing the latency and the hardware requirements of the system.

Custom Hardware High-Speed Architecture

High performance readout card

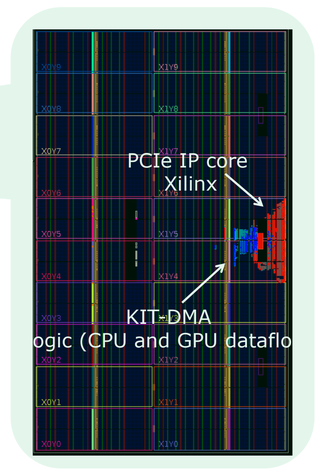
A multi-purposes high-performance readout card is under final assembly. It is designed to interface detectors and GPUs with transfer rates of GB/s.



- FPGA: Virtex 7, speed grade -2
- PCIe: Gen 3 x16 lanes (up to 15 GB/s)
- Connections: 2 FMC-VITA 57 connectors
- Memory: DDR3 (119 Gbit/s)

FPGA DMA architecture

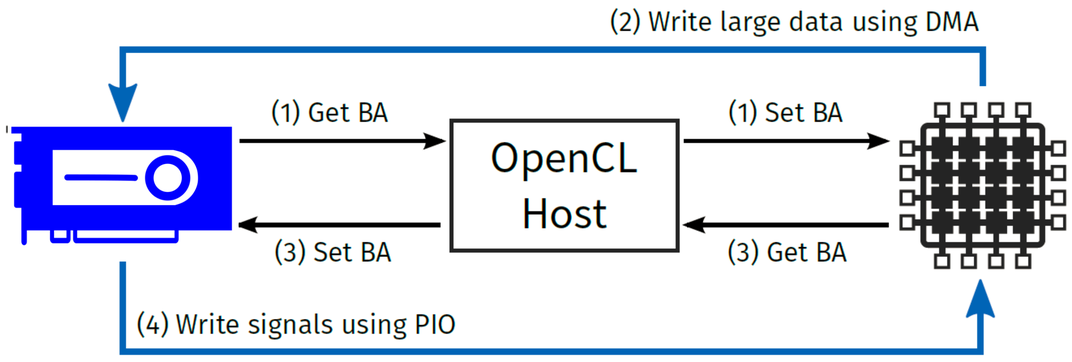
A new very light-weight and high data throughput Direct Memory Access (DMA) engine has been designed. Compatible: Xilinx family 6/7



Floorplan and resource utilization - Virtex 7 (XC7VX690T-2FFG1761C)

Software Implementation

To transfer data from the FPGA to the GPU and send back signals from the GPU to the FPGA, we use AMD's bus-addressable memory extension for OpenCL (DirectGMA). This extension allows the declaration of GPU buffers. Their physical bus addresses are retrieved and configured at the FPGA (1) and used with DMA (2). It also allows mapping known bus addresses, i.e. for the configuration space of the FPGA.



These mechanisms were integrated into a "DirectGMA" data generator plugin for our computing framework that sets up and executes data processing chains on single and multi GPU nodes. The plugin inserts raw data into the data stream which is then processed accordingly by subsequent filters. The following command shows a decoding and Fourier transform pipeline and the result written to disk:

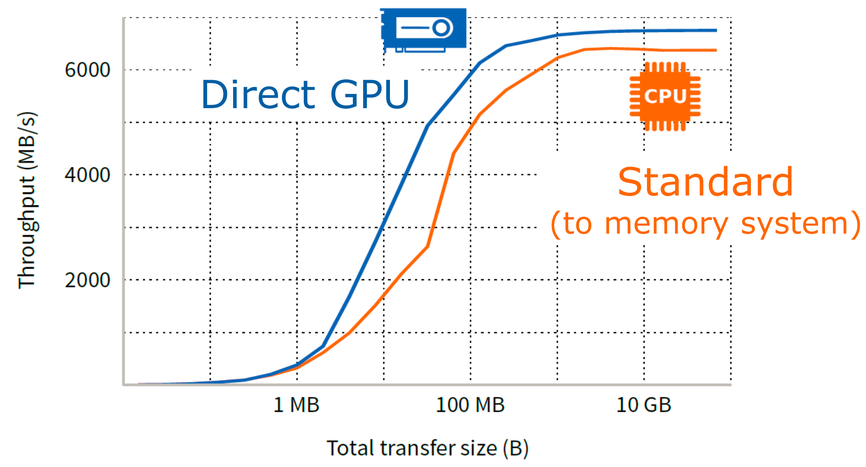
```
$ ufo-launch direct-gma ! decode  fft ! write filename = out.tif
```

At run-time data is pushed to the GPU. Decoding and Fourier transform is performed in parallel.

Results

FPGA Write Performance

DMA can be operated in both configurations: "standard" where the data is transferred from FPGA → System memory and "Direct GPU" where the date is transferred from FPGA → GPU.



The DMA is capable to sustain very fast data transfer in both configurations that exceed 6.3 GB/s. Until 2 MB transfer size, the performance is almost the same, after that the Direct GPU transfer shows a slightly better slope.

CPU vs GPU benchmark

Decoding of a complex data format with 5120×3840 words from high frame rate camera requires 23.17 ms on a Xeon E5-1630 CPU and only **640 µs** on a AMD W9100 FirePro GPU resulting in a processing bandwidth of 1.6 GB/s and 6.5 GB/s. So the CPU bandwidth even with this simple operation is dominated by the decoding while the GPU is still limited by the PCI memory transfer. *The data latency measurements are ongoing, preliminary measurements show a very low latency in the order of some microseconds for Direct GPU access versus a few millisecond for CPU access.*