Harnessing FPGAs potential with OpenCL

PhD Student : Maxime MARTELLI

Thesis : Hardware architecture for very fast sampling generation. Application to radar and electromagnetic listening systems certification

Supervisor : Alain MERIGOT. Co-supervisors : Nicolas GAC, Cyrille ENDERLI

Industrial Context

1. With the **end of Moore's Law**, the semi-conductor industry seeks a reliable way to pursue the performance improvements of the last decades, and architecture-algorithm adequacy is a solution for this new landscape.
2. Manufacturers like Intel and Xilinx are pushing for an FPGA resurgence, offering software suites and FPGAs card focused on a **software-like FPGA programming model**[1].

General Thesis Objectives

1. Evaluate and use heterogeneous architectures to accelerate algorithms, specifically **RADAR processing**.
2. Enabling software-like programming for FPGA implementations with architecture and algorithm adequacy in mind.
3. Definition of a methodology for implementing acceleration for general algorithms depending on their inherent specificity.

First use case : 3D X-ray Computed Tomography Reconstruction

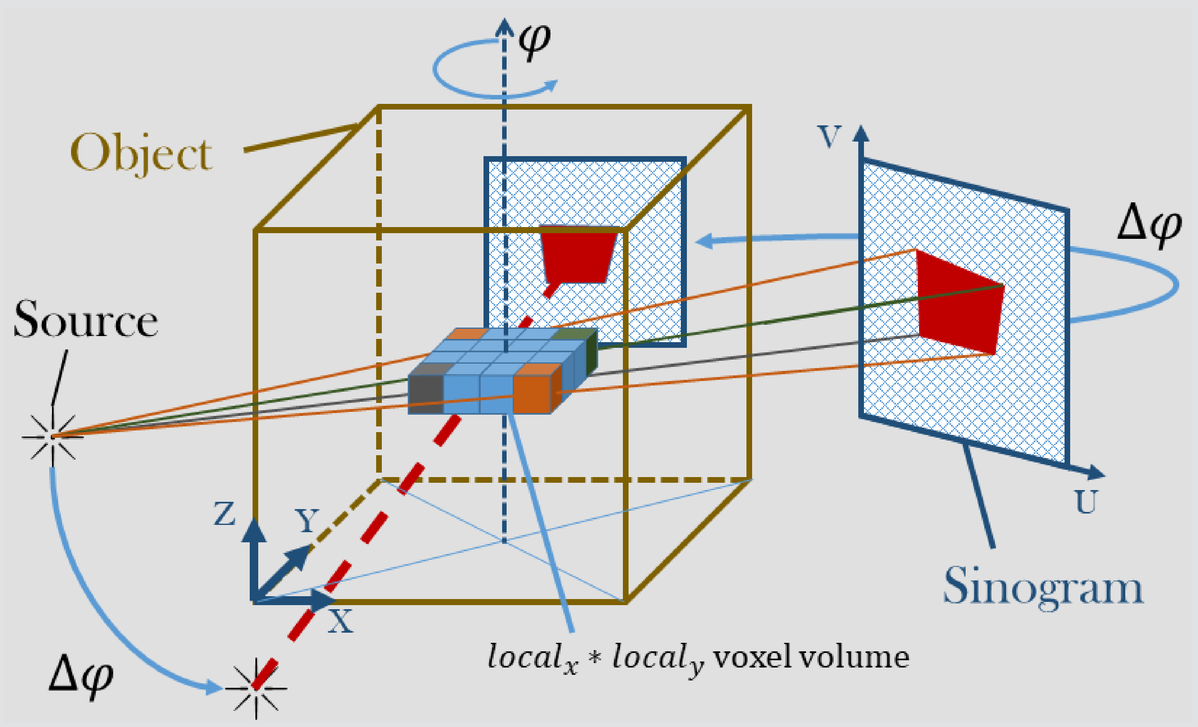


Figure 1: 3D Computed Tomography Back-projection.

To compute the density of a given VOlume piXEL (voxel) $\vec{c} = (x, y, z)$, we sum up the contribution of every elementary detector (u, v) in line with the source and the considered voxel for every φ value.

Using OpenCL on FPGAs

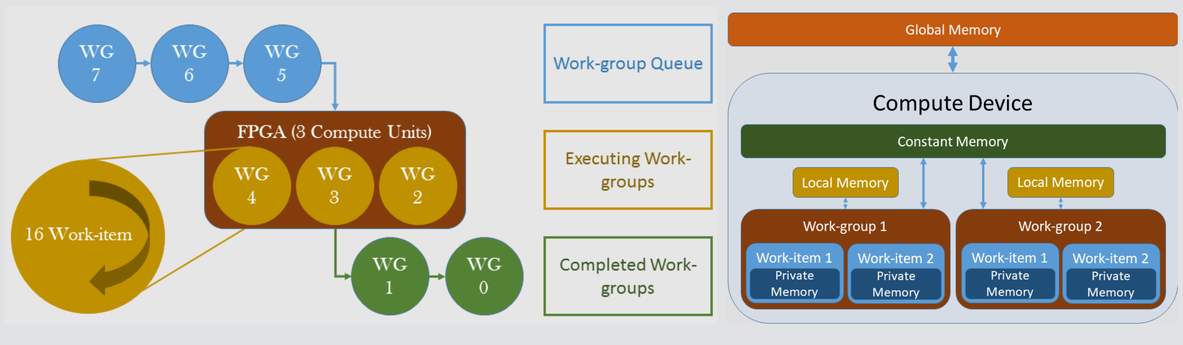


Figure 2: Work-group queueing mechanism and OpenCL Memory Architecture.

There are two OpenCL kernel programming models : **NDRange** (NDR) that is Data Parallelism and **Single Work Item** (SWI), that enables task parallelism. OpenCL allows FPGAs programming, and one contribution of this work is to evaluate its effectiveness compared to VHDL and GPU implementations [2].

Memory Benchmark

For benchmarking purposes, we implemented a custom routine program to measure the mean latency of each memory type on an Altera Cyclone V FPGA.

Memory structure	Kernel Frequency (MHz)	Mean latency (cycles)
Global	137.4	164
Constant	150.4	45
Local	137.1	12
Private	161.3	1

Table 1: Memory structure latency on an Altera Cyclone V.

Implemented OpenCL optimizations

- **SWI Naive :** CPU like naive version.
- **SWI+SRP :** Shift Register Pattern (FIFO queue) to reduce logic utilization.
- **NDR+Naive :** GPU like naive version (shared local memory).
- **NDR+2CU :** Kernel replication of the NDR+Naive version.
- **NDR+MF :** Prefetching mechanism bound to the algorithm specificity. The goal is to access the pattern needed by a group of voxels in one call.

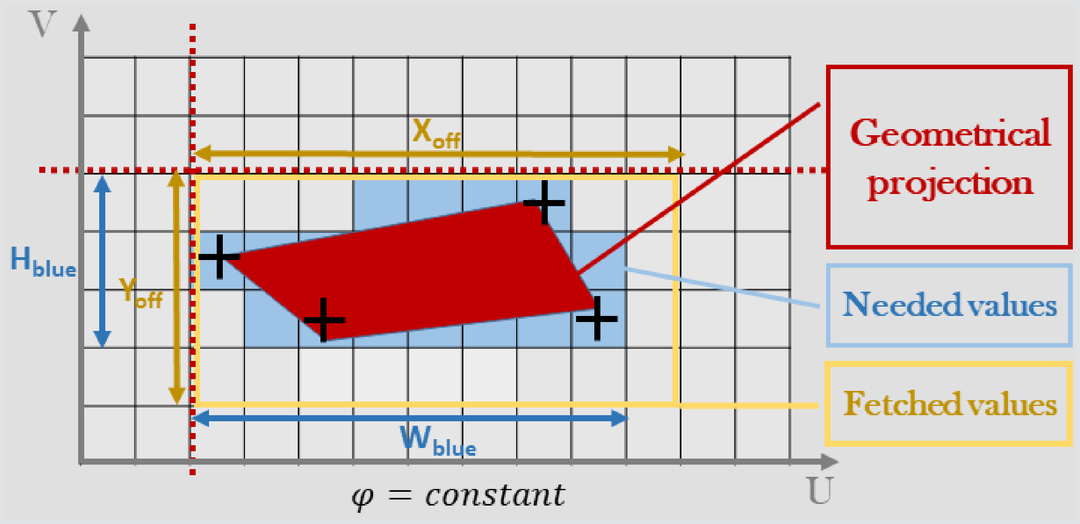
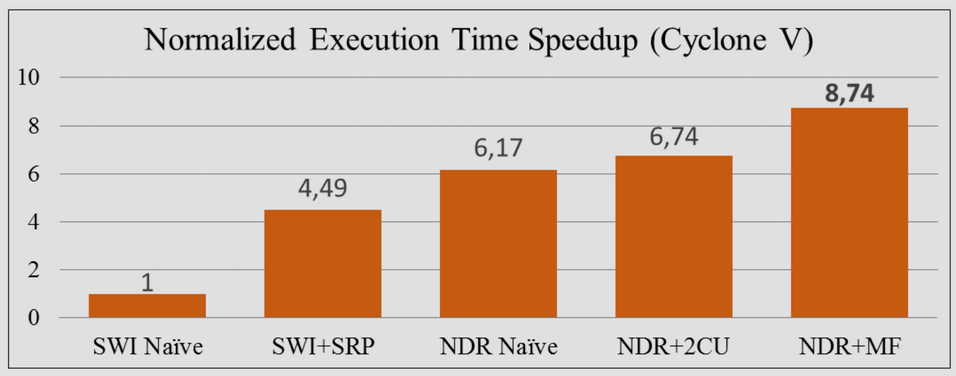


Figure 3: Sinogram memory fetching pattern optimization.

FPGA obtained speedup and GPU comparison



- The Altera Offline Compiler is effective in guaranteeing little kernel stall.
- Optimizing a SWI kernel lies with **optimizing memory handling** and data streaming effectiveness in order to increase kernel frequency.
- Optimizing a NDRange kernel coincide with **reducing the logical footprint** to allow more kernel replication within the same chip.

Device	NET (ms)	Energy (mWh)	Efficiency (cycles)
Titan X Pascal	12	0.83	12.16
Jetson TK2	253	1.054	19.6
Intel Arria 10	991	0.63	1.02

Even with the extrapolated results an the improvements obtained on an Aria 10, the FPGA has a much longer execution time than the GPUs, mostly due to the back-projection algorithm being adapted to SIMD architectures.

Conclusion

1. With an overall **8.74 speedup** [3] between naive and optimized kernels, there is room for more improvements closely related to memory handling.
2. With OpenCL, FPGAs have a good pipeline efficiency. But, the algorithm being well suited for SIMD execution, FPGAs frequency and limited pipeline replication explains their limited performance compared to GPUs.

Current works and perspectives

1. **Accelerating radar processing algorithms on FPGAs.**
2. Benchmarking FPGAs using OpenCL with 1D/2D filtering algorithms.
3. Comparing various implementations on heterogeneous architectures (CUDA, OpenACC, OpenMP, OpenCL).

References

[1] Kavya Shagrithaya, Krzysztof Kepa, and Peter Athanas. Enabling Development of OpenCL Applications on FPGA platforms. *ASAP*, 2013.

[2] Nicolas Gac, Stephane Mancini, Michel Desvignes, and Dominique Houzet. High Speed 3D Tomography on CPU, GPU, and FPGA. *EURASIP journal on Embedded Systems*, 2008.

[3] Maxime Martelli, Nicolas Gac, Alain Mérigot, and Cyrille Enderli. 3D Tomography parallelization on FPGAs via HLS tools. In *DASIP*, Dresden, Germany, September 2017.