

Accelerating 2D FFTs and Convolutions for Seismic Processing

Oskar Mencer¹ and Robert G. Clapp²

¹ Imperial College London and Maxeler Technologies

² Stanford University

Motivation

As microprocessors are hitting the limits of attainable clock frequencies and acceptable power consumption, we see a significant inflection point in the high performance computing environment. Intel and AMD are scaling up the number of cores per chip and processors per node in order to exploit parallelism for potentially increased performance. Existing software has to be modified to take advantage of only modest speed improvements.

The change in software presents an opportunity to move beyond conventional processors to custom accelerators with the potential of much higher performance. Field-Programmable Gate Array (FPGA) accelerators can speedup highly parallel applications, such as seismic processing, by an order of magnitude.

In seismic processing Downward Continuation based Migration is the current high-end workhorse, and Reverse Time Migration shows promise for the future. Most of the compute time in these two applications is taken up by FFTs and Convolutions. By greatly accelerating these two algorithms turnaround time can be significantly improved. Some applications that benefit include Shot Profile Migration, Common Azimuth Migration, Narrow Azimuth Migration, Reverse-time Migration, Elastic/Acoustic Forward Modelling, 2D/3D SRME, Waveform Inversion and Wave-equation Migration Velocity Analysis.

In this project at the Center for Computational Earth and Environmental Science at Stanford, headed by Professor Jerry Harris, we demonstrate the feasibility of using FPGAs to gain significant speedup in seismic applications. This collaboration between Imperial College, Maxeler Technologies and Stanford shows speedups over a 2.8Ghz AMD Opteron.

Results

We study the potential of FPGAs to accelerate FFT and Convolution. Our target is to accelerate 3D FFT and 3D Convolution, however given the short timescale for this project we have confined ourselves to 2D algorithms. Based on experimental results from this project, we project speed-ups of up to 14x for 2D convolution and 18x for 2D FFT, depending on data precision. Figure 2 shows detailed speed-up potential for 2D convolution and FFT.

An analysis of trends in FPGA capabilities and PC interface bus bandwidths shows that the advantages of using FPGAs are likely to increase over time, compared to single and multi-core processor implementations (Figure 1).

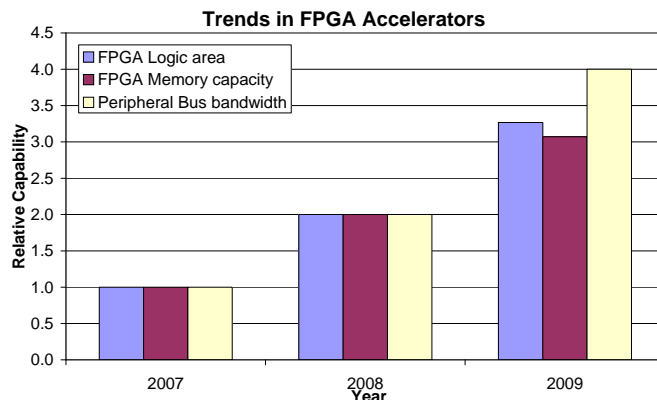


Figure 1: Projected growth in the capabilities of FPGA accelerators based on industry roadmaps.

	MAX-1
FPGA	Xilinx Virtex-4, 40k slices
Interconnect	PCI Express x8
Bus bandwidth	4GB/s
Internal bandwidth	over 1TB/s
Internal memory	over 500KB

Table 1: Characteristics of the MAX-1 FPGA card used in this project.

Stanford-Imperial-Maxeler Collaboration
This project is a joint research effort between the Center for Computational Earth and Environmental Science at Stanford University, Computer Architecture Research Group at Imperial College London, and Maxeler Technologies. The *Center for Computational Earth and Environmental Science* [1] at Stanford, directed by Professor Jerry Harris, brings a partnership between the Earth sciences community and the computer sciences community, each driving development of the other. The computational research at CEES utilizes state-of-the-art hardware and modern computational methods to address complex Earth science problems and the analyses of massive datasets. The *Computer Architecture Research Group* [2] at the Department of Computing, Imperial College London, headed by Dr Oskar Mencer, investigates novel methods of optimizing the performance of state-of-the-art computer hardware, including the use of FPGA hardware accelerators. *Maxeler Technologies* [3] is a technology company that provides complete solutions for maximal acceleration of software via hardware accelerators.

	Memory	Area	Throughput
R2 RAM	n	1	Low
R2 SP	$0.5 \cdot n \cdot \log n$	$\log n$	Moderate
R2 MP	$0.75 \cdot n \cdot \log n$	$2 \cdot \log n$	Moderate
R4 MP	$1.25 \cdot n \cdot \log n$	$3 \cdot \log n$	High

Table 2: The area versus memory design space for computing Fast Fourier Transform on FPGA: Radix-2 RAM-based, Radix-2 single-path streaming, Radix-2 multi-path streaming and Radix-4 multi-path streaming.

Project Details

FPGA based solutions offer a large design space. The absence of tools to adequately explore this space has been a limitation in the past. We use the ASC [4] FPGA programming tool to develop a range of different solutions. ASC, A Stream Compiler, was developed following research at Stanford University and Bell Labs, and is now commercialized by Maxeler Technologies. ASC enables the use of FPGAs as highly parallel stream processors and generates hardware from a C++ input description.

We use a Maxeler MAX-1 FPGA card equipped with a Xilinx Virtex-4 FPGA. Information on this platform’s capabilities is shown in Table 1.

2D FFT We identify a range of different possible circuits for accelerating 1D FFTs, as shown in Table 2. These circuits offer different trade-offs between area and performance, allowing us to select the optimal circuit for a given problem within resource limitations. The first option (R2 RAM) stores the FFT dataset in on-chip memory and loops over it in a similar way to a processor. R2 RAM offers a highly scalable solution to compute large FFTs with reasonable memory and logic area requirements. The other three circuits in Table 2 operate on continuously streaming data, offering higher throughput at the expense of area. 2D FFTs consist of multiple passes through a 1D FFT accelerator circuit, or alternatively the vector-radix method performs small 2D FFTs on the FPGA.

2D Convolution We identify two basic mechanisms for performing 2D Convolution on the FPGA. Firstly, we place the dataset directly into FPGA internal memory and compute the full result locally. In this case control logic directs inputs from memory, through an arithmetic pipeline and back into memory at the end of the computation.

Secondly, an alternative approach streams data through FIFOs and computes the results in a streaming fashion. We implement these relatively small delay FIFOs using FPGA internal memory which is limited by the FPGA memory capacity shown in Figure 1.

Variable Precision All circuits produced with the ASC system are fully parametrizable in terms of their number representation and precision: fixed/floating-point with variable bitwidths. For the FFT low bitwidth fixed-point representations allow more computation to take place in a given FPGA area. FPGAs allow error requirements to be taken into account when accelerating an application, offering a direct trade-off between speed and precision.

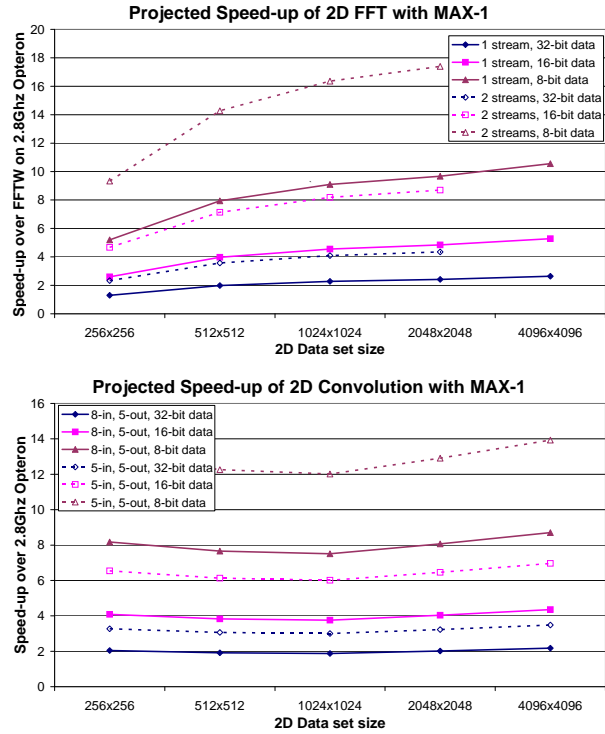


Figure 2: Projected speed-ups for 2D FFT and Convolution.

User Perspective Replacing the FFT and Convolution function calls with FPGA calls makes it easy for programmers to integrate FPGAs and obtain the acceleration while hiding all the complexity of FPGA circuits inside the acceleration library.

Conclusion

We demonstrate that FPGAs accelerate seismic applications. Next-generation FPGA platforms will offer greatly increased parallelism as well as higher bus bandwidth. Trends in FPGA logic area, memory capacities and bus bandwidth (Figure 1) show scaling of the FPGA advantage over time. Achieving high performance with FPGAs remains difficult, requiring careful analysis of application bottlenecks and identification of optimum points in the accelerator design space. In the future as the technology matures we expect the acceleration process to become increasingly automated. At present, for certain problems, acceleration engineers with suitable tools can deliver speedups of an order of magnitude over conventional processors.

References

- [1] Center for Computational Earth and Environmental Science, Stanford University. <http://cees.stanford.edu>
- [2] Computer Architecture Research Group, Imperial College London. <http://www.doc.ic.ac.uk/~oskar>
- [3] Maxeler Technologies. <http://www.maxeler.com>
- [4] O. Mencer. *ASC, A Stream Compiler for computing with FPGAs*, IEEE Transactions on Computer Aided Design, August 2006.