# FPGA Based OpenCL Acceleration of Genome Sequencing Software

Ashish Sirasao, Elliott Delaye, Ravi Sunkavalli, Stephen Neuendorffer
Xilinx Inc. 2100 Logic Drive San Jose, CA 95124
{asirasa, elliott, rsunkav, stephenn}@xilinx.com

## ABSTRACT

**The Smith-Waterman algorithm produces the optimal pairwise alignment between two sequences of proteins or nucleotides and is frequently used as a key component of alignment and variation detection tools for next-generation sequencing data. In this paper an efficient and scalable implementation of the Smith-Waterman algorithm is written in OpenCL and implemented on a Xilinx Virtex-7 FPGA which shows >2x compute performance and 18x-20x performance per watt advantage compared with 12 core and 60 core CPUs. Against a GPU we show a 30% performance advantage and 11.6x better performance per watt. These results were achieved using an off the shelf PCIe accelerator card by optimizing kernel throughput of a systolic array architecture with compiler pipelining of the OpenCL kernels, carefully mapping memories used during computation and adjusting the number of systolic nodes per compute unit to fully utilize the FPGA resources.**

## 1. Introduction

Various software solutions for speeding up Smith-Waterman algorithm have been published on CPUs [3,5] which make explicit use of SIMD assembly instructions. FPGA acceleration results have also been published [4] implementing Smith Waterman in RTL. The work in this paper is the first FPGA solutions that can significantly outperform state of the art SIMD CPU solutions but can do so while using OpenCL. This is also the first paper benchmarking Smith-Waterman with a performance per watt objective on CPUs, FPGAs and GPUs which is important for power budgets in data center applications. The paper also discusses the performance tradeoffs when using an OpenCL based programming environment and how we optimized the architecture to achieving system level performance.

## 2. Smith-Waterman Algorithm

The Smith-Waterman Algorithm [1] is a method for performing local sequence alignment to determine similar regions between two genomic or protein sequences. The algorithm first computes a score matrix followed by dynamic programming based backtracking to find optimal substring alignment. Score matrix computations are the most time consuming part of this algorithm and optimizing this step is the target for computational research. The performance of an algorithm is measured in billion cell updates per second (GCUPS) where the nominal number of score cells for an alignment is *m*n* where *m* and *n* refer to the lengths of the query and reference sequences.

$$H_{i,j} = \max\begin{cases} H_{i-1,j-1} + score(a_i, b_i) & \text{match/mismatch} \\ H_{i-1,j} - penalty & \text{insert/delete error} \\ H_{j-1,i} - penalty & \text{insert/delete error} \end{cases}$$

**Figure 1. Summary of Smith-Waterman score calculation**

## 3. FPGA and OpenCL

FPGA (Field Programmable Gate Array) architectures are efficient in providing highly parallel and customizable compute and customizable memory hierarchies. Combinations of these characteristics along with ability to reconfigure FPGA logic makes FPGAs competitive in applications in terms of performance and performance per watt metrics in various applications. Earlier work [4] configured FPGAs into a systolic array to compute the Smith-Waterman score matrix. This and related FPGA approaches have written systolic arrays in RTL due to the inherent parallelism of RTL languages. The work in this paper demonstrates an approach in OpenCL that can equivalently describe these structures and the productivity gains of OpenCL over RTL allow exploration of various architectures of computation in order to maximize the number of Smith-Waterman GCUPS. The OpenCL kernels were run through Xilinx SDAccel software to compile and implement on the FPGA. The FPGA accelerator card used is the Alpha Data ADM-PCIE-7V3.

## 4. FPGA Implementation

### 4.1 Architecture Exploration

The architectural decisions required for maximizing system performance of Smith-Waterman algorithm on the FPGA were:

1. Use of multiple compute units (kernels) of Smith-Waterman
2. Reduce number of cycles required to execute each kernel.
3. Maximize the clock frequency of each kernel.
4. Maximize the GCUPS/area ratio
5. Use a memory and interconnect scheme with least overhead

It was observed that 1, 2, 3 and 4 above are interrelated and have impact on final application performance. Initial architecture evaluations started with the Striped Smith-Waterman algorithm in OpenCL to generate FPGA logic for the SIMD operations. Performance of this implementation was compared to a more traditional linear systolic array also written in OpenCL. Table 1 shows the performance measured for a single kernel with a single query/reference pair.

**Table 1. SW Architecture GCUPS on FPGA**

| Architecture on FPGA | GCUPS |
|---|---|
| Striped Smith-Waterman | 0.6 |
| Linear Systolic (32 PEs) | 3.2 |

### 4.2 Systolic Architecture in OpenCL

FPGA implementation of Smith-Waterman is a well-researched area. Systolic array approaches typically create a wavefront of computation that starts in one corner of the score matrix and produce successive anti-diagonal scores. The number of systolic cells can vary up to the minimum of the query or the reference length. In OpenCL, one cycle of the systolic array can be described as a *for* loop with an iteration count matching the number of cells. When the loop is unrolled, each function call inside the loop must read and write to separate variables to preserve independence similar to a traditional systolic array. After each iteration of the loop all written variables will become the read variables for the next loop. This is double-buffering approach which if correctly compiled exactly fits semantics of flip-flops. OpenCL attributes were required to efficiently use the memory hierarchy of RAMs and flip-flops. The implementation in this paper does not have any limitation on size of reference string and the results describe alignment of nucleotides but can be enhanced easily to support comparison of protein strings. When the query and reference strings are larger than the number of systolic cells, the anti-diagonal wavefront is segmented and this is also described in OpenCL.

Table 2 demonstrates that while the GCUPS per kernel grows with the number of systolic cells, after 32 cells we observed diminishing returns. Table 2 also shows that increasing single kernel performance by using more systolic cells does not translate to maximal system performance. For improving system performance, tradeoffs between optimizing data structures, memory hierarchy and kernel area must all be considered. In our experiments we filled the FPGA with as many kernel compute units as possible and found total GCUPS increase up to 32 systolic cells per kernel and decrease beyond that.

**Table 2. Theoretical GCUPS vs. kernel sizes**

| #Systolic Cells | GCUPS/Kernel | Max Kernels | Total GCUPS |
|---|---|---|---|
| 8 | 0.8 | 153 | 120.0 |
| 16 | 1.6 | 80 | 128.1 |
| 32 | 3.2 | 42 | 135.4 |
| 64 | 5.8 | 21 | 122.3 |
| 128 | 8.7 | 11 | 98.9 |

## 4.3 System Performance and Integration

As genome sequencing is required to process a large data set and any acceleration solution is required to coexist with a broader software framework following optimizations were performed at the system level:

1. The host to FPGA communication is limited by the PCIe bandwidth of the accelerator system. To optimize the communication between the host and the FPGA, the read and reference data was compressed by using 2 bits per base pair.
2. Smith-Waterman requires generation of an *m*n* cost matrix. Instead of storing cost as a number, a running max-score was computed with directions for each cells contributing to the maximum score. This reduces the cost representation to 2 bit per cell. Max scores were represented as 8 bit integers.
3. To enable calling FPGA accelerator in existing sequencing packages, individual calls to Smith-Waterman algorithm are required to be converted to batch mode to reduce roundtrip overhead on PCIe link and all kernels query/reference pairs were combined into a contiguous memory region to allow a single burst read from DDR3 into the FPGA.

## 5. Results and Conclusion

Table 3 shows the performance comparison of OpenCL on the FPGA vs. SSW [4] on an Intel Xeon 12 core processor, SWAPHI [7] on an Intel Xeon Phi 60 core coprocessor and GSWABE on an nVidia Tesla K40 [6]. For this comparison two million samples of query/reference pairs were used with read size as 128 and reference size as 256. These lengths are similar to sizes seen when Smith-Waterman algorithm is used in genome alignment stack for variant calling [3]. FPGA power is reported from the power estimate provided by SDAccel. The ADM-PCIE-7V3 card is allowed 25W per the PCIe specifications however this design does exceed the allowance by a small margin. For the Intel CPUs thermal design power (TDP) was used for estimating power at max load when all cores are utilized. Future work will measure the actual power consumption of the FPGA, CPUs and GPU.

**Table 3. Performance comparison of single device**

| | GCUPS | Watts | GCUPS/Watt |
|---|---|---|---|
| Xilinx Virtex-7 690T | 77.00 | 28.00 | 2.8 |
| Intel® Xeon E5-2697 | 19.7 | 130.00 | 0.15 |
| Intel® Xeon Phi 5110P | 29.5 | 225.00 | 0.13 |
| nVidia® Tesla K40 | 59.1 | 245.00 | 0.24 |
| FPGA vs Xeon E5-2697 | 3.90 | 0.22 | 18.10 |
| FPGA vs Xeon Phi E5110P | 2.57 | 0.12 | 20.63 |
| FPGA vs Tesla K40 | 1.30 | 0.11 | 11.67 |

Based on the results it can be seen that FPGA provides significant performance and performance per watt advantage. The results however are still well below our calculated maximum of 135.4 GCUPS from table 2. Future work will focus on how to design better memory interfaces to more efficiently feed the FPGA kernels. OpenCL on FPGAs is a promising area for compute applications and when FPGA acceleration boards become integrated into data centers, genomics applications will benefit from integration of OpenCL in FPGAs.

## 6. Acknowledgements

The experiments presented in this paper were carried out using the Alpha Data ADM-PCIE-7V3 boards at Xilinx in San Jose, CA. We would like to thank Avinash Thakare, Jeff Lin, Ravi Kurlaganda.

## 7. References

[1] Smith, T.F. and Waterman, M.S. 1981. Identification of common molecular subsequences. *J. Mol. Biol.*, 147, 195-197

[2] McKenna A., et al. 2010. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research*, 20, 1297-303 DOI=10.1101/gr.107524.110

[3] Zhao, M., et al., 2013 SSW library: an SIMD Smith-Waterman C/C++ library for use in genomic applications. *PLoS One*, vol. 8, pp. e82138, 2013. DOI=10.1371/journal.pone.0082138

[4] Benkrid, K., et al, 2009, A Highly Parameterized and Efficient FPGA-Based Skeleton for Pairwise Biological Sequence Alignment, *IEEE Transactions on VLSI*, 17, 4, 561-570, DOI=10.1109/TVLSI.2008.2005314

[5] Liu, Y., Schmidt, B., 2014, SWAPHI: Smith-Waterman Protein Database Search on Xeon Phi Coprocessors, *25th IEEE Intl. Conf. on Application-Specific Systems, Architectures and Processors 2014*, 184-185, DOI=10.1109/ASAP.2014.6868657

[6] Liu, Y., Schmidt, B., 2014, GSWABE: faster GPU-accelerated sequence alignment for optimal alignment retrieval for short DNA sequences, Concurrency and Computat.: Pract. Exper. 2015; 27:958-972, DOI=10.1002/cpe.3371