Journal of Applied Research and Technology
ISSN: 1665-6423
jart@aleph.cinstrum.unam.mx
Centro de Ciencias Aplicadas y Desarrollo
Tecnológico
México

Performance versus Power Analysis for Bioinformatics Sequence Alignment

L. Hasan*, H. Zafar

Department of Computer Systems Engineering,
University of Engineering and Technology, Peshawar, Pakistan
* laiqhasan@nwfpuet.edu.pk

ABSTRACT
Due to the utilization of abundant hardware resources, power consumption is becoming an important constraint for bioinformatics sequence alignment applications. In this paper, the dynamic power consumption for such applications and its impact on performance is evaluated. Additionally, resource utilization and performance results are provided for implementation with a number of different FPGA platforms. The results obtained using *Xilinx ISE* tools and *Matlab* demonstrate that the performance per unit Watt increases rapidly when increasing the number of *Processing Elements (PEs)*. Increasing the number of PEs beyond a certain number slows down the performance per unit Watt significantly. This behavior is used for approximating the number of PEs that gives an optimized performance per unit Watt.

Keywords: Sequence alignment algorithms, FPGAs, performance, dynamic power consumption

1. Introduction

Bioinformatics sequence alignment is a power hungry activity [1, 2]. Various sequence alignment methods are available [3]. Methods like BLAST [4], FASTA [5] and HMMER [6] are fast, but they are based on heuristics and do not guarantee an optimal alignment. Based on *dynamic programming (DP)* [7], the *Smith-Waterman (S-W)* algorithm [8] is a method that finds an optimal local sequence alignment between two DNA or protein sequences, i.e. the *query sequence (Nq)* of length N and the *database sequence (Ns)* of length M. To come up with an efficient and fast sequence alignment solution, the S-W algorithm is most often implemented as a linear systolic array [9] on hardware platforms like *field programmable gate arrays (FPGAs)*. The length of the query sequences, aligned in a single pass using such arrays, depends on the number of available PEs. The more the PEs, the longer the query sequences that can be aligned against the database sequences in a specific amount of time. The quantity of PEs that can be placed, routed and utilized, in turn depends on the availability of the amount of hardware resources. Current FPGA technology offers abundant hardware resources, sufficient for fitting a large number of PEs. Hence, longer query sequences can be aligned in one pass against the database sequences. However, the performance per unit Watt is limited by the higher dynamic power consumption for larger designs. Work has been done on accelerating the S-W algorithm in hardware [9–13], but no tangible effort has been made to evaluate the impact of power consumption on the overall performance.

In this paper, an evaluation of dynamic power consumption for sequence alignment applications is presented and the performance per unit Watt for various numbers of PEs is investigated, considering different FPGA platforms for implementations. The resource utilization and performance results are provided. The analysis helps in approximating the number of PEs that gives an optimized performance per unit Watt.

2. Hardware-based S-W design

Semicrystalline PET In this section, we present the implementation of a PE for the hardware-based S-W algorithm and the subsequent designs. Figure 1 shows the block diagram representation of the PE design, used to compute cells in the *H* matrix in a linear systolic array fashion [14].

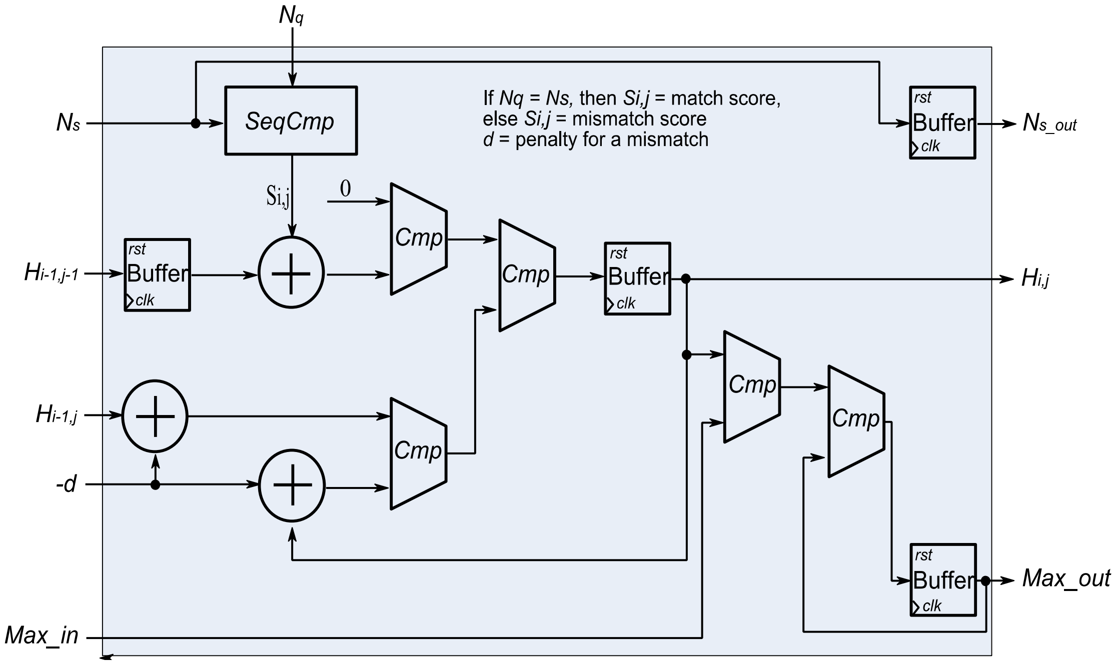
Nq
Ns
SeqCmp
If Nq = Ns, then Si,j = match score,
else Si,j = mismatch score
d = penalty for a mismatch
rst
Buffer
clk
Ns_out
Si,j
0
Hi-1,j-1
Cmp
Hi,j
Hi-1,j
-d
Max_out
Max_in

Figure 1. Processing element for hardware based S-W design.

In the PE design of figure 1, *SeqCmp* compares the corresponding characters of the two input sequences and generates a similarity score. If the corresponding characters are similar, the similarity score is equal to a specific match score, otherwise it is equal to a mismatched score. The diagonal input from the element ($H_{i-1,j-1}$) is delayed by a buffer for one clock cycle, as it is the output of the preceding element in the array. The similarity score is added with the delayed diagonal element using an adder, the output out of which is compared with a 0 using a comparator. The comparator returns 0 if the output of the adder is negative, otherwise it returns the output of the adder.

The left element ($H_{i-1,j}$) and the upper element (which is the current value of the PE) are added with the gap penalty using adders, the outputs out of which are compared using a comparator that returns the maximum of the two values. This value is then compared with the value of the previous comparator (the one that compared the sum of the diagonal element and the similarity score with a 0) and the maximum of the two values is returned. The value of the PE, hold in a buffer, is compared with the Max in value from the preceding PE to find the global maximum. The current maximum value of the PE, hold in another buffer, is compared with the global maximum. The values of the current and global maximums are compared and the greatest of the two values is returned and stored in a buffer.

Another buffer is used to delay the database sequence (N_s) by one clock cycle for the succeeding element of the array. The external clock and reset lines are connected with the *clk* and *rst* inputs of all the buffers. The described PE is used to implement an FPGA based 4-PE linear systolic array design, as shown in figure 2. The array design uses *Block RAM (BRAM)* for intermediate data storage before transmitting the resultant data to the PC, since BRAM is the nearest and fastest available on-chip memory. In addition, there are two BRAMs for the input sequences, i.e. a BRAM for N_q and a BRAM for N_s. These two BRAMs are initialized with the values of the two input sequences. The input sequences are applied to the PEs in such a way that the N_q values stay fixed in their corresponding PEs, whereas the N_s values are propagated through the array in synchronism with the clock. The PE itself and the subsequent array are both implemented in *very high speed IC Hardware Description Language (VHDL)* to verify the correctness of the design. The design is scalable and it is used to implement arrays of various sizes for performance and power analysis, as presented in the succeeding sections.

In practice, a large number of PEs is required to align long sequences. The larger the number of PEs, the longer the query sequences that can be aligned against the database sequences in a single pass. When all the PEs are simultaneously active, the dynamic power consumption by the design increases when increasing array length. Also, the on-chip local BRAM becomes very limited for storing all the intermediate values and can only be used as a buffer that transfers the data to an off-chip main memory, e.g. the double data rate (DDR) RAM. Figure 3 gives a block diagram description of such an extended system. Thus, an evaluation of the dynamic power consumption and its impact on performance becomes very critical. The following section evaluates the dynamic power consumption of the hardware-based S-W design.

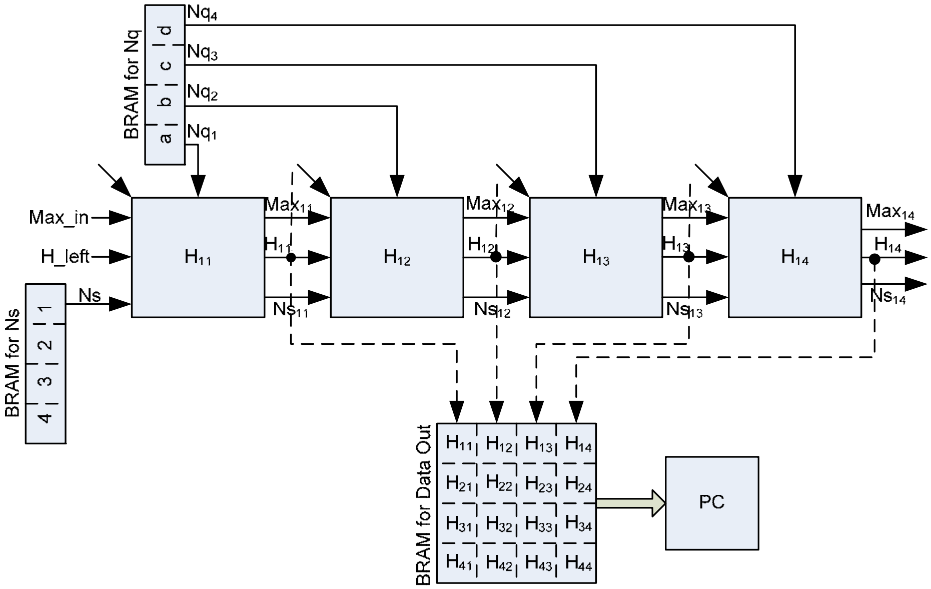

Figure 2. FPGA-based linear systolic array design.

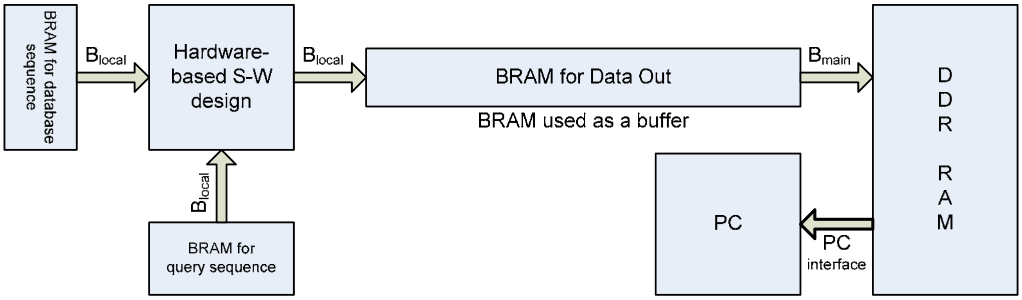

Figure 3. Block diagram description of an FPGA-based design for aligning long sequences.

3. Dynamic power consumption evaluation

The dynamic power consumed by an FPGA is largely due to the charging and discharging activities of the capacitive elements, such as logic resources and the interconnecting fabric [15]. This can be modeled as,

$$P_i = \sum C_i V_i^2 f_i$$

where C_i, V_i and f_i are the capacitance, supply voltage and operating frequency of resource *i*, respectively [16].

In this section, dynamic power consumption is evaluated for the hardware-based S-W design, described in the previous section. Randomly selected input sequences from *ssearch* class-c benchmark of BioPerf are used for simulations. The BioPerf suite [17] includes benchmark source codes (e.g. *ssearch* for the S-W algorithm), input datasets of various sizes, and information for compiling and using the benchmarks. It contains codes from highly popular bioinformatics packages [18] and covers the major fields of study in computational molecular biology, such as sequence comparison, phylogenetic reconstruction, protein structure prediction, and sequence homology & gene finding. The benchmark considered for simulations represents the complete genome. The number of PEs is scaled according to the lengths of the input biological sequences, randomly selected from the benchmark for the evaluation of dynamic power consumption. However, sequences of lengths larger than the maximum available PEs are aligned by partitioning the query sequences [19]. For each selected length, a variety of input sequences are considered for simulations and the average dynamic power consumption is recorded. Power analyzer tool *XPower* of Xilinx *ISE 10.1* Design suite is used for the power analysis, whereas the devices used for implementations are Xilinx *Virtex2P (XC2VP30)*, *Virtex4 (XC4VFX12)* and *Virtex5 (XC5VTX240T)* FPGAs.

Table 1 presents an evaluation of the dynamic power consumption for the hardware-based S-W design, considering varying number of PEs. The device used for implementation is XC2VP30. The 1st column represents the number of PEs. The 2nd column shows the power consumed by clock transitions, which increases with the increasing number of PEs. The 3rd column gives the power consumed by logic. Again, the power consumption increases with the increasing number of PEs except for the 1st row, where more power is consumed than that for the succeeding higher number of PEs. The reason for this is that the memories are also implemented as logic by the Xilinx ISE tool and no BRAMs are instantiated. The 4th column provides the power consumed by the signals, i.e. the dynamic power consumption due to the switching activity along the wires. The 5th column represents the combined power consumed by IOs and BRAMs. The last column presents the total dynamic power consumption, which is the sum of power consumed by clocks, logic, signals, IOs and BRAMs, i.e.

Tables 2 and 3 present dynamic power consumption results for implementations using XC4VFX12 and XC5VTX240T devices, where similar trends are observed, as for XC2VP30 device in table 1. The maximum number of PEs in table 2 is limited due to the reduced amount of resources offered by the XC4VFX12 device.

PEs	Clocks	Logic	Signals	IOs + BRAMs	Total
4	2.07	1.19	1.50	0.10	4.85
6	2.23	0.69	2.23	0.10	5.25
8	2.75	0.84	2.33	0.11	6.02
20	5.34	0.89	5.17	0.12	11.51
44	8.18	1.86	11.38	0.40	21.81
72	10.15	2.99	19.63	0.43	33.18
108	10.87	5.43	33.64	0.53	50.46

Table 1. Dynamic power consumption in milliwatts for XC2VP30 implementation.

PEs	Clocks	Logic	Signals	IOs + BRAMs	Total
4	28.11	0.36	0.33	0.03	28.82
6	29.00	0.19	0.34	2.34	31.87
8	32.92	0.21	0.58	2.48	36.19
20	37.24	0.26	0.85	7.71	46.06
44	41.26	0.57	2.68	17.61	62.12
48	41.21	0.68	4.53	19.15	65.57

Table 2. Dynamic power consumption in milliwatts for XC4VFX12 implementation.

PEs	Clocks	Logic	Signals	IOs + BRAMs	Total
4	9.32	0.15	0.16	0.03	9.66
6	11.10	0.10	0.21	0.78	12.19
8	12.03	0.12	0.23	1.02	13.39
20	22.05	0.19	0.47	2.42	25.12
44	37.82	0.41	1.38	5.25	44.85
72	81.93	0.63	2.39	8.84	93.79
108	118.22	1.14	4.51	13.04	136.90

Table 3. Dynamic power consumption in milliwatts for XC5VTX240 implementation.

4. Resource utilization

In this section, resource utilization and performance results are presented for implementations with various numbers of PEs, considering the input biological sequences from ***ssearch*** [20] class-c benchmark of BioPerf. Xilinx *ISE 10.1* simulator is used for synthesis and post place and route simulations. The devices considered for implementations are XC2VP30, XC4VFX12 and XC5VTX240T FPGAs.

Table 4 presents device utilization in terms of slices and BRAMs, considering XC2VP30 implementation. Furthermore, it provides the maximum frequency in Mega Hertz (MHz) and performance in *Giga Cell Updates per Second (GCUPS)* for the hardware-based S-W design. The 1st column in the table represents the number of PEs. The 2nd column provides the number of slices consumed for all given numbers of PEs. The 3rd column presents the BRAMs utilization. The reason for having no BRAMs in the 1st row is that when a limited number of memories needs to be instantiated then the Xilinx *ISE* synthesizer puts them in *Look Up Tables (LUTs)* instead of BRAMs during the synthesis process, to avoid any wastage of BRAM resources. The on-chip BRAM in FPGAs is a limited commodity and this approach saves it for other applications. The 4th column gives the maximum post place and route frequency in MHz. The last column presents the performance in GCUPs, calculated as follows:

$$\text{Performance} = N_{PE} \times f$$

where N_{PE} is the number of PEs and f is the maximum operating frequency.

Similarly, tables 5 and 6 present device utilization and performance results for implementations with XC4VFX12 and XC5VTX240T devices. Tables 4, 5 and 6 indicate an increase in the performance for a higher number of PEs. However, a decreasing trend is observed for the maximum operating frequency due to the higher latency for larger designs.

In the following section, performance optimization is presented to approximate the number of PEs that gives an optimized performance per unit Watt.

PEs	Slices	BRAMs	Frequency (MHz)	Performance (GCUPS)
4	646	---	110.26	0.441
6	723	3	110.00	0.660
8	975	4	109.80	0.878
20	2307	10	109.00	2.180
44	4897	24	107.20	4.717
72	7762	38	105.50	7.596
108	11737	56	103.70	11.908

Table 4. Device utilization and performance results for XC2VP30 implementation.

PEs	Slices	BRAMs	Frequency (MHz)	Performance (GCUPS)
4	670	---	140.64	0.563
6	816	3	140.00	0.840
8	1072	4	139.44	1.115
20	2478	10	136.32	2.726
44	4943	24	129.79	5.711
48	5359	26	128.63	6.174

Table .5. Device utilization and performance results for XC4VFX12 implementation.

PEs	Slices	BRAMs	Frequency (MHz)	Performance (GCUPS)
4	317	---	198.63	0.794
6	429	3	197.38	1.184
8	552	4	196.13	1.569
20	1461	10	192.31	3.846
44	3343	21	189.13	8.322
72	5479	35	186.42	13.422
108	8286	52	181.56	19.608

Table 6. Device utilization and performance results for XC5VTX240T implementation.

5. Performance per unit Watt

Power consumption is becoming an important constraint for modern day computationally intensive applications, such as biological sequence alignment. Detailed power analysis is important, since this helps in optimizing the performance and power efficiency of hardware designs for the aforementioned applications.

In this section, an optimized performance per unit Watt for the hardware-based S-W design is investigated. This is done by scaling the linear systolic array design for various numbers of PEs and measuring the dynamic power and performance values. The scaling criterion is based on the lengths of multiple input biological sequences, randomly selected from BioPerf Benchmark Suite to have a realistic measure of the dynamic power consumption. Figure 4 depicts the results of performance per unit Watt for various number of PEs, considering various FPGA platforms like XC2VP30, XC4VFX12 and XC5VTX240T for implementations.

The results in figure 4 demonstrate that the performance per unit Watt increases when increasing the number of PEs initially. It stabilizes after increasing the number of PEs beyond a certain point and eventually starts to decrease. The curve for XC4VFX12 is shorter than the other two curves due to a limited amount of resources offered by the device. The results are influenced by the following two factors.

1) The sub-linear increase in performance with the increasing number of PEs. The reason for this is that the maximum operating frequency decreases due to the increasing latency for larger designs.
2) The slightly super-linear increase in dynamic power consumption with the increasing number of PEs. The reason for this is that larger designs generate higher switching activity and hence consume more dynamic power.

This analysis helps in approximating the number of PEs that gives an optimized performance per unit Watt. It is observed from figure 4 that for achieving an optimized performance per unit Watt, the number of PEs can be approximated between 40 and 60 for XC4VFX12 and XC5VTX240T FPGA devices. Similarly, it can be approximated between 70 and 80 for XC2VP30 device. Beyond these numbers, the performance per unit Watt decreases with any further increase in the number of PEs.

The results are approximated by using the MATLAB curve fitting tool and selecting a 4th degree polynomial for the curve fit, as it better resembles the experimental curves and gives a minimum *root mean square error (RMSE)*.

$$f(x) = c_1 \times x^4 + c_2 \times x^3 + c_3 \times x^2 + c_4 \times x + c_5$$

The above equation gives an approximated model where, $x = N_{PE}$. The values of the polynomial coefficients and RMSE for various FPGA platforms as determined by the curve fitting tool are given in Table 7.

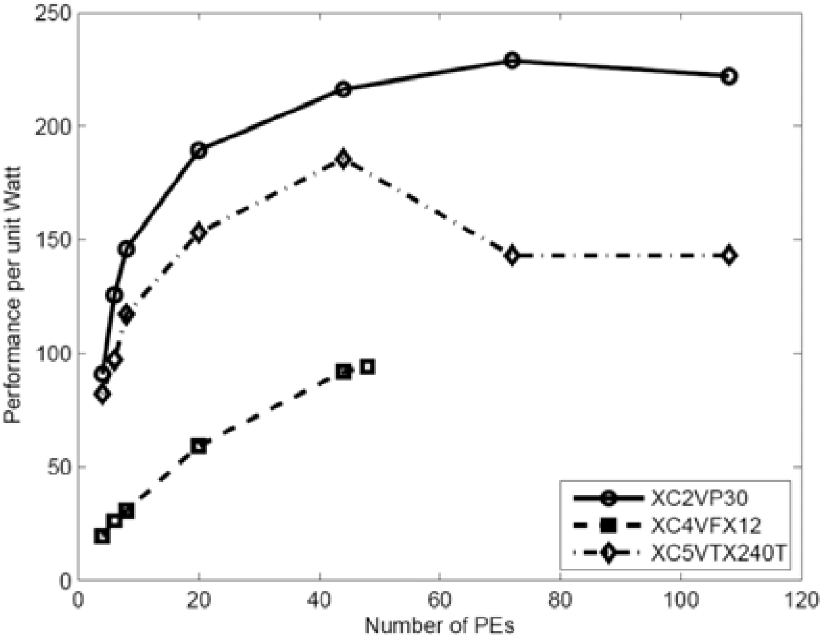

Figure 4. Performance per unit Watt on various FPGA platforms.

Coefficients	XC2VP30	XC4VFX12	XC5VTX240T
c_1	-1.87×10^{-005}	-5.302×10^{-006}	3.212×10^{-006}
c_2	0.004467	0.0005742	3.455×10^{-005}
c_3	-0.368	-0.04786	-0.09594
c_4	12.73	3.376	6.668
c_5	55.93	7.03	61.48
RMSE	11.78	0.8504	7.464

Table 7. Modeling coefficients for various FPGA platforms.

6. Conclusion

This paper presented an evaluation of dynamic power consumption for bioinformatics sequence alignment. It gave resource utilization in terms of slices and BRAMs. Furthermore, the paper presented the performance in terms of GCUPS for various numbers of PEs, using different FPGA platforms for implementations. The results demonstrated an initial rapid increase in the performance per unit Watt when increasing the number of PEs. It stabilized after increasing the number of PEs to a certain limit. Beyond that limit, a decreasing trend is observed for the performance per unit Watt. The experimental data is used for approximating the number of PEs such that an optimized performance per unit Watt is achieved.

References

[1] M. Vingron and M.S. Waterman, “Sequence Alignment and Penalty Choice: Review of Concepts, Case Studies and Implications”, Journal of Molecular Biology, vol. 235, pp. 1-12, 1994.

[2] A. YarKhan and J.J. Dongarra, “Biological Sequence Alignment on the Computational Grid Using the GrADS Framework”, Future Generation Computer Systems, vol. 21, no. 6, pp. 980-986, June 2005.

[3] L. Hasan, Z. Al-Ars and S. Vassiliadis, “Hardware Acceleration of Sequence Alignment Algorithms – An Overview”, Proc. International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS’07), pp. 96-101, Rabat, Morocco, September 2-5, 2007.

[4] S.F. Altschul, W. Gish, W. Miller, E.W. Myers and D.J. Lipman, “A Basic Local Alignment Search Tool”, Journal of Molecular Biology, vol. 215, pp. 403-410, 1990.

[5] W.R. Pearson and D.J. Lipman, “Rapid and Sensitive Protein Similarity Searches”, Science, vol. 227, pp. 1435-1441, 1985.

[6] S.R. Eddy, “Profile Hidden Morkov Models”, Bioinformatics Review, vol. 14, no. 9, pp. 755-763, July 1998.

[7] R. Giegerich, “A Systematic Approach to Dynamic Programming in Bioinformatics”, Bioinformatics, vol. 16, pp. 665-677, 2000.

[8] T.F. Smith and M.S. Waterman, “Identification of Common Molecular Subsequences”, Journal of Molecular Biology, vol. 147, pp. 195-197, 1981.

[9] L. Hasan and Z. Al-Ars, “An Efficient and High Performance Linear Recursive Variable Expansion Implementation of the Smith-Waterman Algorithm”, Proc. 31st Annual International Conference of the IEEE EMBS, pp. 3845-3848, Minneapolis, Minnesota, USA, September 2009.

[10] A.B. Buyukkur and W. Najjar, “Compiler Generated Systolic Arrays for Wavefront Algorithm Acceleration on FPGAs”, Proc. International Conference on Field Programmable Logic and Applications (FPL08), Heidelberg, Germany, September 2008.

[11] A.D. Blas, D.M. Dahle, M. Diekhans, L. Grate, J. Hirschberg, K. Karplus, H. Keller, M. Kendrick, F.J. Mesa-Martinez, D. Pease, E. Rice, A. Schultz, D. Speck and R. Hughey, “The UCSC Kestrel Parallel Processor”, IEEE Transactions on Parallel and Distributed Systems, vol. 16, no. 1, pp. 80-92, 2005.

[12] W. Liu, B. Schmidt, G. Voss, A. Schroder and W. Muller-Wittig, “Bio-Sequence Database Scanning on a GPU” HICOMB, 2006.

[13] L. Hasan, Z. Al-Ars, Z. Nawaz and K. L. M. Bertels, “Hardware Implementation of the Smith-Waterman Algorithm Using Recursive Variable Expansion”, Proc. 3rd International Design and Test Workshop IDT’08, Monastir, Tunisia, December 2008.

[14] L. Hasan, Y.M. Khawaja and A. Bais, “A Systolic Array Architecture for The Smith-Waterman Algorithm With High Performance Cell Design”, Proc. IADIS European Conference on Data Mining, pp. 35-42, Amsterdam, The Netherlands, July 2008.

[15] L. Shang, A.S. Kaviani and K. Bathala, “Dynamic Power Consumption in VirtexTM-II FPGA Family”, FPGA’02, Monterey, CA, USA, February 24-26, 2002.

[16] G. Yeap, Practical Low Power Digital VLSI Design, Kluwer Academic Publishers, 1998.

[17] BioPerf, http://www.bioperf.org/.

[18] Y. Yu, L.A. Santat and S. Choi, “Bioinformatics packages for sequence analysis”, Bioinformatics, vol. 6, pages 143-160, 2006.

[19] L. Hasan, Z. Al-Ars and M. Taouil, “High Performance and Resource Efficient Biological Sequence Alignment”, Proc. 32nd IEEE EMBS, Buenos Aires, Argentina, Aug 31-Sep 4, 2010.

[20] L. Hasan, M. Kentie and Z. Al-Ars, “DOPA: GPU-based Protein Alignment Using Database and Memory Access Optimizations”, Submitted to BMC Bioinformatics, 2011, ISSN 1471-2105.