SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Accelerating DNA Sequence Alignment using Altera DE2-115

Muhammad Hasif Aiman Mohd Sarwar Kamal Helal[1], Fadli Hamidi Rusli[1], Ahmad Hasif Azman[2], Abdul Karimi Halim[1], Abdul Hadi Abdul Razak[1], S. Kaja Mohideen[3], Syed Abdul Mutalib Al Junid[1,4,*]

[1] Electronic Architecture and Application Research Group, School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, Selangor, Malaysia
[2] National Water Research Institute of Malaysia (NAHRIM), Seri Kembangan, Selangor, Malaysia
[3] Department of Electronics and Communication Engineering, B.S. Abdur Rahman Crescent Institute of Science and Technology, Chennai, India
[4] Microwave Research Institute, Shah Alam, Selangor, Malaysia

ARTICLE INFO

Article history:
Received 4 January 2024
Received in revised form 28 August 2024
Accepted 30 August 2024
Available online 1 October 2024

Keywords:
FPGA; Smith-Waterman algorithm; DNA sequence alignment

ABSTRACT

DNA sequence alignment is a technique for discovering information between two base sequences which the Smith-Waterman algorithm is the accurate method that provides a precise result for alignment compared to others. However, the performance was influence by size of dataset and a long DNA base sequence which resulted the time required for the alignment process is much longer in relation to the number of DNA sequence samples. There are many ways to accelerate DNA sequence alignment, and Field Programmable Gate Array (FPGA) is a good choice due to its parallel processing and cost efficiency. Although FPGA acceleration approaches are not new, this work investigates a purely software-based FPGA acceleration using the Altera Cyclone IV EP4CE115F29C7N FPGA as the target device. The SW algorithm was developed using the C language in Quartus II version 18.1 and the Nios II software build tools for Eclipse. The development starts with setting up the QSys architecture before developing the code in Eclipse to determine the computational performance. The result shows the computational timing and speed of the implementation, with the highest speed achieved being 198.76 cells per millisecond. To summarise, the computational performance ultimately depends on the maximum matrix size of the FPGA, which is also influenced by the DNA-based pair length and able to complete using low-cost FPGA.

1. Introduction

The human body stored the information of itself in nucleic acid sequences called the human genome. The genome consists of 23 chromosomes, a dense strand of deoxyribonucleic acid or DNA that coded all the information [1]. Four bases made up DNA; adenine(A), cytosine(C), guanine(G), and thymine(T). A human has billions of bases [2,3], and the sequence of the bases made each person different from each other, even from its close relatives. Based on "The 1000 Genomes Project Consortium" [4], the typical difference between two persons estimated at around 0.6% or 20 million base pairs. The variations in human DNA cause different features in a person like tanned skin,

* *Corresponding author.*
E-mail address: samaljunid@uitm.edu.my

https://doi.org/10.37934/araset.52.1.122131

different eyes, and hair colour. The bases differences can be found by comparing two DNA samples and the process called DNA sequence alignment. In more detail, the DNA sequence alignment is a process of comparing and determining the order of nucleic acid and the bases sequence; adenine, cytosine, guanine, and thymine.

This method is also indispensable in biological research and practical applications such as medical diagnosis, criminal forensics, pharmaceutical development, and virology. In forensics science, it can be used to determine the gender of the criminal and check the similarity between DNA samples. As most of the viruses are small, the virus study can use the approach, and it also allows researchers to predict future virus outbreaks as the viral sequencing can be used by measuring its mutation rate. Other than that, DNA sequencing can provide us with specific antibiotics and reduce the risk of giving antibiotics that are not effective to certain bacteria or bodies [5]. Hence, creating a faster way to align the DNA bases made the applications more effective as it is allowing better discovery of applications, faster research development, and guaranteed precision.

Generally, there are two main methods of DNA sequencing, that is heuristic and dynamic programming as shown in Figure 1. Word or heuristic methods such as FASTA and BLAST produce outcomes that are not guaranteed in optimal alignment, but it is faster than dynamic programming [6–10]. Heuristic methods are suitable to speed up database searching since it is not searching for the details similarity. The dynamic approach is a more precise method of alignment, but the speed decreasing according to the size of data.

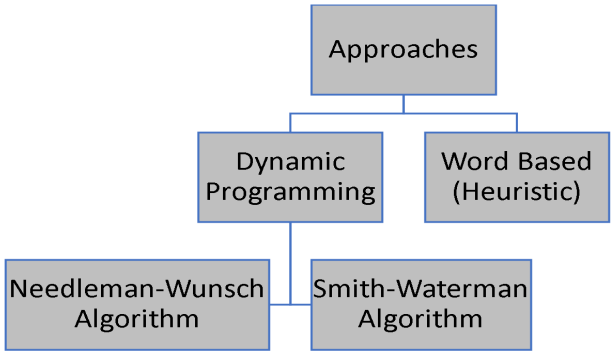

Fig. 1. The approaches used for DNA sequence alignment

Over the past few decades, most research for DNA sequence alignment has emphasized using the general computer as the computational medium. In 1969, Saul B. Needleman and Christian D. Wunsch developed an adaptable computer method that finds the similarities in the amino acid sequences between two proteins [11]. In the study, the algorithm proposed applying the dynamic programming to use global alignment, which executes the sequence comparison from end-to-end of the sample. Later, the Smith-Waterman algorithm[12] was introduced as the previous Needleman-Wunsch algorithm alteration, which implements a local alignment and different scoring methods.

Using a general computer for DNA sequence alignment requires a high cost and hinders DNA sequencing technology development due to computational sequence execution issues. As a result, the FPGA should be more suitable for this proposed DNA sequence alignment application due to faster speed [6,13], low cost [6,13[14]-15], and high cost-effectiveness [15]. In [16], the researcher presents the performance comparison between the FPGA, Graphics Processor Units (GPU), and IBM's Cell Broadband Engine (Cell BE) for the Smith-Waterman pairwise sequence alignment algorithm. The result shows that FPGA has outperformed the GPU and Cell BE in performance per dollar and performance per watts. In conclusion FPGA offers a more cost-effective and reliable solution for this design process. Additionally, FPGA technology has been widely adopted [17]-[19].

In achieving the speed and precision of DNA sequence alignment, it is important to improve the speed of this Smith-Waterman algorithm since it already has good precision. Numerous approaches have been proposed to accelerate this Smith-Waterman algorithm using FPGA, including be it in matrix fill-up, systolic array, FPGA & server, sub-function implementation, affine gap or path reconstruction approach; [2,13,15,20-32]. Based on the previous work proposed on FPGA, in this paper, the focus will be more on implementing the Smith-Waterman algorithm on the FPGA and examining the influence of the design over the frequency and processing runtime.

1.1 Smith-Waterman Algorithm

The Smith-Waterman algorithm is known as local sequence alignment, where it is suitable for more divergent sequencing, while Needleman-Wunsch is a global sequence alignment and is more suitable for closely related sequences. However, both of these dynamic sequences are guaranteed to find the optimal local alignment. The Smith-Waterman algorithm is divided into three basics steps that are:1) Initialization, 2) Matrix filling, and 3) Traceback.

1.1.1 Initialization

In the Smith-Waterman algorithm, the maximum match can be determined by creating an edit-distance matrix [33] that compares the short sample of query sequences (Q) and target sequences (T) while giving information regarding the similarity of compared sequences. In identifying the pairings of segments with a high degree of similarity, matrix H is set up with (M+1)×(N+1) dimensions. The first steps of the Smith Waterman are initialization, which creates a zero region at H_(0,j) and H_(i,0) as shown in Table 1.

Table 1
Initialization of the matrix

	$H_{i,0}$	Q_1	Q_2	….	Q_M
$H_{0,j}$	0	0	0	….	0
T_1	0				
T_2	0				
….	….				
T_N	0				

The region involved during the initialization process can be summarized by Eq. (1).

$$H_{i,0} = 0;\ 0 \leq i \leq m$$
$$H_{0,j} = 0;\ 0 \leq j \leq n \quad (1)$$

1.1.2 Matrix filling

$$H_{i,j} = max \begin{cases} 0 \\ H_{i-1,j-1} + s_{i,j} \\ H_{i-1,j-d} \\ H_{i,j-1-d} \end{cases} \quad (2)$$

The matrix filling is the next stage where the remaining cell will be filled. Given the sequences Q and T with the length of the sample is M and N respectively, the matrix indicates the number of changes in terms of substitutions, insertions, and deletions that performed to align Q with T. In the matrix filling, the scoring for the sequence will be filled on $a_{i,j}$, $b_{i,j}$, and the rest by comparing the bases on Q_M and T_N. The score was based on the Q_M and T_N either it is a match or not and the scoring for the column on the upper, left and diagonal upper left, shown in Figure 2. The three separate score is calculated and the maximum score between these columns will be assigned into the cell.

	$H_{i,0}$	Q_1	Q_2	....	Q_M
$H_{0,j}$	0	0	0	0	0
T_1	0	$a_{i,j}$	$b_{i,j}$		
T_2	0	$d_{i,j}$	$c_{i,j}$	...	
....	0		...	$e_{i,j}$	
T_N	0				

Fig. 2. Edit-distance matrix; query (q) with *m* length and target (t) with *n* length

The scoring for match, mismatch and gap is +10, -5, -10 respectively which the negative value become 0 when filled on a new column (Eq. (3) and (4)).

$$Matching\ Q\ and\ T; \begin{cases} (top\ cell) - 10 \\ (left\ cell) - 10 \\ (diagonal\ cell) + 10 \end{cases} \quad (3)$$

$$Mismatch\ Q\ and\ T; \begin{cases} (top\ cell) - 10 \\ (left\ cell) - 10 \\ (diagonal\ cell) - 5 \end{cases} \quad (4)$$

1.1.3 Traceback

The traceback is the final step, and it starts after all the cell is being filled. The trace will begin from the highest value of score gain in the matrix table to form an optimal sequence alignment path and end when the similarity score drops. Table 2 shows the traceback pathway in grey.

Table 2
Traceback pathway in grey

	$H_{i,0}$	G	A	T	C
$H_{0,j}$	0	0	0	0	0
T	0	0	0	10	0
A	0	0	10	0	5
T	0	0	0	20	10
C	0	0	0	10	30

In Table 2, the Q sequences (GATC) and T sequences (TATC) are being aligned, and the matrix has been filled with the scoring. This traceback step starts at the highest score cell, usually located at the back of the matrix. Later, it moves to the next highest neighboring cell to form an optimal sequence alignment as shown in Figure 3.

Fig. 3. The optimal sequence alignment that been formed from Q sequences (GATC) and T sequences (TATC)

2. Methodology

The SW algorithm was developed using C language in Quartus II version 18.1, and Nios II Software Build Tools for Eclipse, targeted to Altera Cyclone IV EP4CE115F29C7N FPGA. The results for the timing of the program execution will be compiled as the result of this project. Figure 4 shows in this implementation, the project is created in Quartus Prime 18.1 version, which is used to build an embedded sub-system in SOPC builder. The function of this builder is to integrate the sub-system in the FPGA to become one complete system according to the user's need. The SOPC was created using the Platform Designer, and the Qsys file created interconnected logic will be used to program the FPGA. The Nios II software was later used to run the C codes that have been developed with the input of the DNA sequence data in the text file.

In reading the text file on Nios II, the BSP setting needs to be setting up by including the software packages for HAL's (Hardware Abstraction Layer). This software package provides generic device models for classes of peripherals command in embedded systems like timers and I/O peripherals that allow the user to use specific C command to open and read the file while mounting the text file with the program.

The code reads the text files for the query-sequence and target sequence by character and stores them in arrays. The matrix dimensions are set up, and the initialization step will take up by inserting the value of zero into the first matrix column and row. The other cell will be filled during the matrix

filling process using Eq. (3) and (4) depends on its condition, match, or mismatch. The traceback later will be performed to find the highest scoring matrix from the cells.

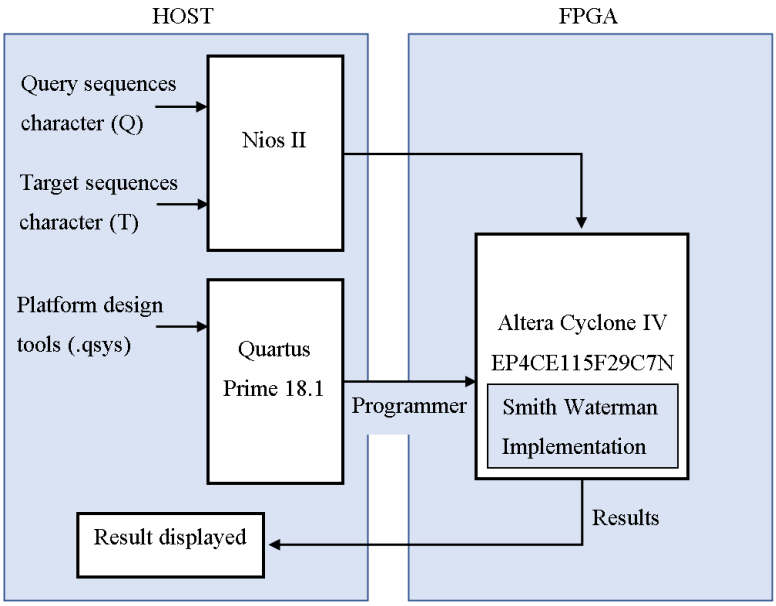

Fig. 4. Block diagram flow of SW implementation

3. Results

In examining the effect of the clock on the runtime speed, the clock speeds 50MHz, 100MHz, and 200MHz are used to aligning two DNA sequences with identical lengths from 1 to 128 DNA sequence characters. The operating system of EP4CE115F29C7N has been compile using Nios II, with the data from the text file is mounted into the program. Each runtime result of the implementation was recorded and summarised in Table 3.

Table 3
The computational time (ticks) for s-w implementation

Numbers of Cells	Runtime (ticks)		
	50 MHz (50M ticks/s)	100 MHz (100M ticks/s)	200 MHz (200M ticks/s)
1	5 227	5 189	5 212
4	6 591	6 384	6 395
8	14 119	13 862	13 211
64	64 557	64 320	64 401
256	417 875	418 669	418 522
1024	3 209 344	3 253 244	3 252 114
4096	26 314 309	26 373 011	26 372 481
16384	211 279 427	217 547 269	217 554 341

The timestamp that is obtained from the Nios II operation is in ticks and ticks per second. To change the value of data into second, Eq. (5) is use.

$$Runtime\ (ms) = \frac{Ticks\ (runtime)}{Ticks\ per\ second}\ x\ 1\ second \quad (5)$$

Table 4 shows the entire computational time for the S-W implementation in a millisecond (ms) interval.

Table 4
The computational time for s-w implementation

Number of Cells	Runtime (ms)		
	50 MHz	100 MHz	200 MHz
1	0.105	0.052	0.026
4	0.132	0.064	0.032
8	0.282	0.139	0.066
64	1.291	0.643	0.322
256	8.358	4.187	2.093
1024	64.19	32.53	16.26
4096	526.28	263.73	131.86
16384	4225.59	2175.47	1087.77

As highlighted in Table 4, the runtime is increasing proportionally to the number of cells, which the number of cells can be determined based on *m+1* and *n+1* dimensions. For example, when the Q sequences and T sequences have 64 bases characters, the dimension of the matrix is (64+1) x (64+1) = 4226. The additional rows are the row that formed during the initialization step.

Figure 5 shows the runtime versus the number of cell graph. Based on the results, when the clock speed is doubled, the runtime for each alignment is 50% less compared to the previous speed. As tabulated in Table 4, when the clock speed increases from 100 MHz to 200MHz for the 4 cells, the runtime decreases by half from 0.064ms to 0.032ms.

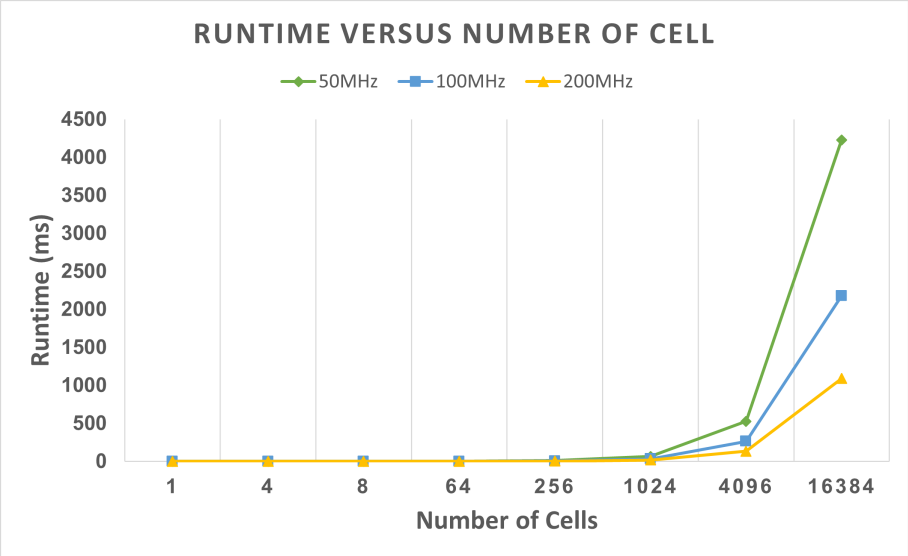

Fig. 5. Runtime versus the number of cell graph

In Figure 5, the line graph shows the runtime for each Smith-Waterman alignment that varies for each number of cells and clock speed. This graph was plotted based on the result recorded in Table 4. This graph clearly shows that the increment of the runtime is proportional to the increment number of cells. Meanwhile, the size of the memory been shown in Table 5.

Table 5
The size of memory for every implementation

Number of Cells	Memory (kB)
1	77
4	77
8	77
64	77
256	78
1024	82
4096	93
16384	178

The increase in the runtime is cause by the increase size of the memory for the program after the compile process. This shows that the increase of memory for the DNA sequence alignment, increase the runtime for the S-W implementation. Figure 6 shows the speed that each alignment obtained from the time taken to align a cell.

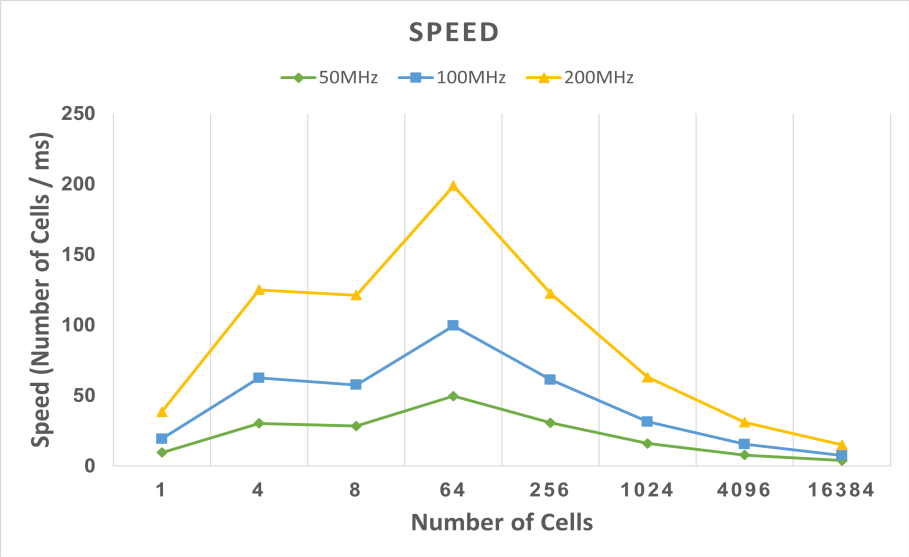

Fig. 6. Speed of each SW alignment completion

Moreover, the peak runtime speed was recorded when aligning 64 number cells compared to the rest with the runtime speed per cell trend increasing for single cells until 64 cells before decreasing after that. The 200MHz clock speed has recorded the fastest average speed compared to the others with the highest during 64 cells with 198.76 cells per milliseconds.

4. Conclusions

This paper presents the DNA sequence alignment accelerator using Quartus II version 18.1, and Nios II Software Build Tools for Eclipse, targeted to Altera Cyclone IV EP4CE115F29C7N Field Programmable Gate Array (FPGA) as to overcome the performance and computational complexity issue. Twenty-four series of tests comprising different DNA sequence-based pair data ranging from single cells up to 16384 cells and three different computational clock speeds have been used to measure the implementation performance. The highest speed recorded is during the 200MHz clock speed with 64 cells, which is 198.76 cells per milliseconds. Not only the computational clock speed but the number of memories also influenced the overall computational runtime. Finally, it can be

concluded that the computational performance ultimately depends on the DNA-based pair's length and computational clock speed.

Acknowledgement
This study was supported by the Ministry of Higher Education Malaysia (MOHE) through the Fundamental Research Grant Scheme FRGS/1/2021/TK0/UITM/02/66. The authors would like to thank Universiti Teknologi MARA for providing the facilities to carry out the work.

References

[1] M. W. Karaman, "Genomes, 2nd edition," *Journal of Heredity* 94, no. 5 (2003): 432-a-433. https://doi.org/10.1093/jhered/esg082
[2] Abou El-Wafa, Wael, Asmaa G. Seliem, and Hesham FA Hamed. "Hardware acceleration of Smith-Waterman algorithm for short read DNA alignment using FPGA." In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, pp. 604-605. IEEE, 2016. https://doi.org/10.1109/COMPSAC.2016.127
[3] Olson, Corey B., Maria Kim, Cooper Clauson, Boris Kogon, Carl Ebeling, Scott Hauck, and Walter L. Ruzzo. "Hardware acceleration of short read mapping." In *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, pp. 161-168. IEEE, 2012. https://doi.org/10.1109/FCCM.2012.36
[4] Consortium, Genomes Project, A. Auton, L. D. Brooks, R. M. Durbin, E. P. Garrison, and H. M. Kang. "A global reference for human genetic variation." *Nature* 526, no. 7571 (2015): 68-74. https://doi.org/10.1038/nature15393
[5] "Antimicrobial resistance," *World Health Organization*, Oct. 13, 2020. https://www.who.int/en/news-room/fact-sheets/detail/antimicrobial-resistance (accessed Dec. 18, 2020).
[6] Al Junid, Syed Abdul Mutalib, Zulkifli Abd Majid, and Abdul Karimi Halim. "High speed DNA sequencing accelerator using FPGA." In *2008 International Conference on Electronic Design*, pp. 1-4. IEEE, 2008. https://doi.org/10.1109/ICED.2008.4786759
[7] Al Junid, Syed Abdul Mutalib, Zulkifli Abd Majid, and Abdul Karimi Halim. "Development of DNA sequencing accelerator based on Smith Waterman algorithm with heuristic divide and conquer technique for FPGA implementation." In *2008 International Conference on Computer and Communication Engineering*, pp. 994-996. IEEE, 2008. https://doi.org/10.1109/ICCCE.2008.4580756
[8] Al Junid, S. A. M., M. A. Haron, Z. Abd Majid, A. K. Halim, F. N. Osman, and Hadzli Hashim. "Development of Novel Data Compression Technique for Accelerate DNA Sequence Alignment Based on Smith–Waterman Algorithm." In *2009 Third UKSim European Symposium on Computer Modeling and Simulation*, pp. 181-186. IEEE, 2009. https://doi.org/10.1109/EMS.2009.93
[9] Al Junid, S. A. M., M. A. Haron, Z. Abd Majid, F. N. Osman, H. Hashim, M. F. M. Idros, and M. R. Dohad. "Optimization of DNA sequences data to accelerate DNA sequence alignment on FPGA." In *2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation*, pp. 231-236. IEEE, 2010. https://doi.org/10.1109/AMS.2010.54
[10] Seliem, Asmaa G., Wael Abou El-Wafa, A. I. A. Galal, and Hesham FA Hamed. "Parallel Smith-Waterman algorithm hardware implementation for ancestors and offspring gene tracer." In *2016 World Symposium on Computer Applications & Research (WSCAR)*, pp. 116-121. IEEE, 2016. https://doi.org/10.1109/WSCAR.2016.8
[11] Needleman, Saul B., and Christian D. Wunsch. "A general method applicable to the search for similarities in the amino acid sequence of two proteins." *Journal of molecular biology* 48, no. 3 (1970): 443-453. https://doi.org/10.1016/0022-2836(70)90057-4
[12] Smith, Temple F., and Michael S. Waterman. "Identification of common molecular subsequences." *Journal of molecular biology* 147, no. 1 (1981): 195-197.
[13] Mensah, Patrick Kwabena, Edem K. Bankas, and Mohammed Muniru Iddrisu. "RNS Smith-Waterman Accelerator based on the moduli set 2 n, 2 n-1, 2 n-1-1." In *2018 IEEE 7th International Conference on Adaptive Science & Technology (ICAST)*, pp. 1-8. IEEE, 2018. https://doi.org/10.1109/ICASTECH.2018.8506912
[14] Khairudin, N., N. Mahmod, AK Abdul Halim, S. A. M. Al Junid, MF Md Idros, SL Mohd Hassan, and Z. Abdul Majid. "Design and Analysis of High Performance Matrix Filling for DNA Sequence Alignment Accelerator Using ASIC Design Flow." In *2010 Fourth UKSim European Symposium on Computer Modeling and Simulation*, pp. 108-114. IEEE, 2010. https://doi.org/10.1109/EMS.2010.29
[15] Shah, Hurmat Ali, Laiq Hasan, and Nasir Ahmad. "An optimized and low-cost FPGA-based DNA sequence alignment—A step towards personal genomics." In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2696-2699. IEEE, 2013. https://doi.org/10.1109/EMBC.2013.6610096

[16] Benkrid, Khaled, Ali Akoglu, Cheng Ling, Yang Song, Ying Liu, and Xiang Tian. "High performance biological pairwise sequence alignment: FPGA versus GPU versus cell BE versus GPP." *International Journal of Reconfigurable Computing* 2012 (2012): 7-7. https://doi.org/10.1155/2012/752910

[17] Wan Rahiman, "Electronic mailbox designed with fpga platform", *Journal of Advanced Research in Computing and Applications* 22, no. 1 (2024): 1–10. https://www.akademiabaru.com/submit/index.php/arca/article/view/5373

[18] Wan Rahiman, "Infrared sensor and pwm integration using fpga platform for ground vehicle navigation and speed control", *Journal of Advanced Research in Computing and Applications* 25, no. 1 (2024): 1-8. https://www.akademiabaru.com/submit/index.php/arca/article/view/5375

[19] Wan Rahiman, "Prototype of water dam system by using fpga platform", *Journal of Advanced Research in Computing and Applications* 24, no.1 (2024): 1–13. https://www.akademiabaru.com/submit/index.php/arca/article/view/5374

[20] Rucci, Enzo, Carlos Garcia, Guillermo Botella, Armando E. De Giusti, Marcelo Naiouf, and Manuel Prieto-Matias. "Oswald: opencl smith–waterman on altera's fpga for large protein databases." *The International Journal of High Performance Computing Applications* 32, no. 3 (2018): 337-350. https://doi.org/10.1177/1094342016654215

[21] Isa, M. N., D. S. Nurdin, M. I. Ahmad, S. A. Z. Murad, S. N. Mohyar, M. Mahyiddin, A. Harun, R. C. Ismail, S. S. Jamuar, and A. Bahari. "DNA sequence alignment: From algorithm to systolic architecture." In *2016 3rd International Conference on Electronic Design (ICED)*, pp. 304-308. IEEE, 2016. https://doi.org/10.1109/ICED.2016.7804657

[22] Fei, Xia, Zou Dan, Lu Lina, Man Xin, and Zhang Chunlei. "FPGASW: accelerating large-scale Smith–Waterman sequence alignment application with backtracking on FPGA linear systolic array." *Interdisciplinary Sciences: Computational Life Sciences* 10 (2018): 176-188. https://doi.org/10.1007/s12539-017-0225-8

[23] Chiang, Jason, Michael Studniberg, Jack Shaw, Stephen Seto, and Kevin Truong. "Hardware accelerator for genomic sequence alignment." In *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 5787-5789. IEEE, 2006. https://doi.org/10.1109/IEMBS.2006.260286

[24] Buhagiar, Karl, Owen Casha, Ivan Grech, Edward Gatt, and Joseph Micallef. "Hardware implementation of efficient path reconstruction for the Smith-Waterman algorithm." In *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 1113-1116. IEEE, 2017. https://doi.org/10.1109/CoDIT.2017.8102748

[25] Al Junid, Syed Abdul Mutalib, Mohd Faizul Md Idros, Abdul Hadi Abdul Razak, Fairul Nazmie Osman, and Nooritawati Md Tahir. "Parallel processing cell score design of linear gap penalty smith-waterman algorithm." In *2017 IEEE 13th International Colloquium on Signal Processing & its Applications (CSPA)*, pp. 299-302. IEEE, 2017. https://doi.org/10.1109/CSPA.2017.8064969

[26] Zahid, Saad Khan, Laiq Hasan, Asif Ali Khan, and Salini Ullah. "A novel structure of the Smith-Waterman Algorithm for efficient sequence alignment." In *2015 Third International Conference on Digital Information, Networking, and Wireless Communications (DINWC)*, pp. 6-9. IEEE, 2015. https://doi.org/10.1109/DINWC.2015.7054208

[27] Khairudin, N., M. A. Haron, S. A. M. Al Junid, AK Abdul Halim, MF Md Idros, and NF Abdul Razak. "Design and Analysis of High Performance and Low Power Matrix Filling for DNA Sequence Alignment Accelerator Using ASIC Design Flow." In *2011 UKSim 5th European Symposium on Computer Modeling and Simulation*, pp. 123-128. IEEE, 2011. https://doi.org/10.1109/EMS.2011.9

[28] Di Tucci, Lorenzo, Davide Conficconi, Alessandro Comodi, Steven Hofmeyr, David Donofrio, and Marco D. Santambrogio. "A parallel, energy efficient hardware architecture for the meraligner on fpga using chisel hcl." In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 214-217. IEEE, 2018. https://doi.org/10.1109/IPDPSW.2018.00041

[29] Nurdin, D. S., M. N. Isa, and S. H. Goh. "DNA sequence alignment: A review of hardware accelerators and a new core architecture." In *2016 3rd International Conference on Electronic Design (ICED)*, pp. 264-268. IEEE, 2016. https://doi.org/10.1109/ICED.2016.7804650

[30] Razip, Mansur Iylia Mohamed, Syed Abdul Mutalib Al Junid, Abdul Karimi Halim, and Zulkifli Abd Majid. "Sequence alignment using systolic array for an accelerator." In *2014 IEEE 8th International Power Engineering and Optimization Conference (PEOCO2014)*, pp. 663-665. IEEE, 2014. https://doi.org/10.1109/PEOCO.2014.6814510

[31] Liu, Ying, Khaled Benkrid, AbdSamad Benkrid, and Server Kasap. "An fpga-based web server for high performance biological sequence alignment." In *2009 NASA/ESA Conference on Adaptive Hardware and Systems*, pp. 361-368. IEEE, 2009. https://doi.org/10.1109/AHS.2009.59

[32] Houtgast, Ernst, Vlad-Mihai Sima, and Zaid Al-Ars. "High performance streaming Smith-Waterman implementation with implicit synchronization on intel FPGA using OpenCL." In *2017 IEEE 17th International Conference on Bioinformatics and Bioengineering (BIBE)*, pp. 492-496. IEEE, 2017. https://doi.org/10.1109/BIBE.2017.000-6

[33] Caffarena, Gabriel, Carlos Pedreira, Carlos Carreras, Slobodan Bojanic, and Octavio Nieto-Taladriz. "FPGA acceleration for DNA sequence alignment." *Journal of Circuits, Systems, and Computers* 16, no. 02 (2007): 245-266. https://doi.org/10.1142/S0218126607003575