DESIGN AND IMPLEMENTATION OF FPGA-BASED DNA SEQUENCE ALIGNMENT ACCELERATOR

MOHD FIKRI BIN CHE HUSIN

A thesis submitted in
fulfillment of the requirement for the award of the
Degree of Master of Electrical Engineering

Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn Malaysia

DECEMBER 2013

ABSTRACT

Basic Local Alignment Search Tool (BLAST) is a standard computer application that uses in the Bioinformatics field to search for sequence similarity in genomic databases. This project describes a FPGA-based hardware implementation which using a systolic array design architecture for Basic Local Alignment Search Tool (BLAST). Before this designs detect at most one hit in one clock cycle, this design applies a Multiple Hits Detection Module which is a pipelining systolic array to search multiple hits in one clock cycle. This computationally intensive critical segment has been designed and implemented in the FPGA that runs on Cyclone II processor in the Altera DE 2 board. The concepts of portability, scalability and cost-effectiveness of the implementation are also demonstrated from the results obtained.

ABSTRAK

Basic Local Alignment Search Tool (BLAST) ialah satu piawai aplikasi komputer yang digunakan dalam bidang informasi bio untuk mencari kesamaan jujukan dalam pengkalan data genom. Projek ini menerangkan pelaksanaan perkakasan berasaskan FPGA yang menggunakan reka bentuk sistolik pelbagai untuk *Basic Local Alignment Search Tool* (BLAST). Sebelum ini Sebelum reka bentuk ini mengesan paling banyak satu *hit* dalam satu kitaran jam, reka bentuk ini menggunakan Modul Pengesanan Pelbagai *Hit* yang mana reka bentuk sistolik pelbagai digunakan untuk mencari *hit* berganda dalam satu kitaran jam. Pengiraan segmen kritikal intensif telah direka dan dilaksanakan dalam FPGA yang berjalan pada pemproses Cyclone II di dalam Altera DE 2. Konsep mudah alih, berskala dan keberkesanan kos pelaksanaan juga ditunjukkan daripada keputusan yang diperolehi.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

BLAST	-	Basic Local Alignment Search Tool
VHDL	-	Very High Speed Integrated Circuit Hardware Description Language
DNA	-	Deoxyribonucleic acid
FPGA	-	Field-programmable gate array
SDRAM	-	Synchronous dynamic random access memory
HSPs	-	Hit-scoring Sequence Pairs

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

This chapter presents the introduction of the thesis, including with a short overview of the design and implementation of FPGA-BASED DNA sequence alignment accelerator. Furthermore, it details the aims of the project, continuing with the objective as well as the scope of the project and finishing with the outline of the thesis.

1.1 Introduction

The discovery of sequence homology to a known protein or family of proteins often provides the first clues about the function of a newly sequenced gene. As the DNA and amino acid sequence databases continue to grow in size they become increasingly useful in the analysis of newly sequenced genes and proteins because of the greater chance of finding such homologies [1].

With increasingly sophisticated technology in the field of Bioinformatics, scientists and engineers have many options for applying the theories and methods such as algorithms and computational simulations to transform genome into sequence digit numbers.

For example, after sequencing the DNA of the organism, the researchers have a passion for studying the function of genes and compared the gene sequences from different organisms to obtain the similarity between them. The data will help researchers discover new information about the new gene, knowing the function of genes, and

unlock the relationship between the organism. Previously, the researchers did a comparison between the gene (queries) to a database of genes (sequences). This indeed very time consuming process and the probability of errors along the process is very high. Sequences for gene determines the function, so finding similar sequences can help to determine the function of new sequences. Therefore, the compilation of a new sequence comparison process is very dependent on a lot of the same sequence.

Blast (Basic Local Alignment Search Tool) [1] originated at the Washington University in St. Louis, its development was continued by various institutions, whether academic or industry. The blast is a tool used by researchers to find the local similarity between sequences, such comparisons have been described previously. It compares the sequence of nucleotides or protein sequence databases and calculates the statistical significance of matches. The results of BLAST searches have a good statistical interpretation, this makes the correct matching is easier to distinguish from random background hits. Blast using a heuristic algorithm which searches for local rather than global alignment is able to detect the correlation among sequences which share the isolated region of similarity. However, various reports and journals confirm the fact that this result is appropriate to define a significant finding. Since BLAST is based on the heuristic algorithm, then it is not the most accurate tool in search of similarity, but it is still a tool that is widely used by biologists due to the nature of the results produced.

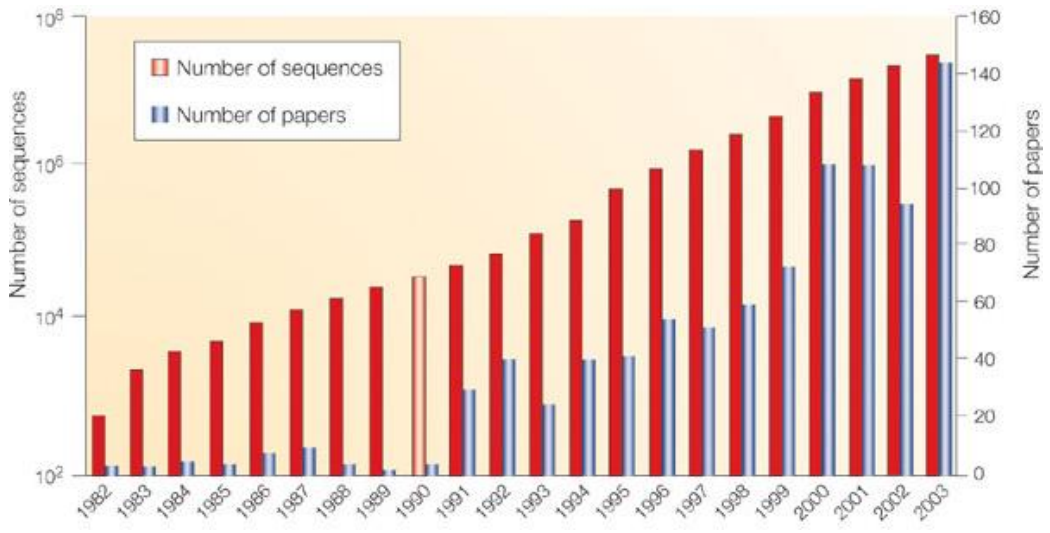

Figure 1.1: Molecular clocks four decades of evolution [2]

To increase the speed of BLAST tools, systolic array architectures have been used in combination with the parallel processing and pipelining in the design. Systolic architecture is compiled and simulated by using Altera Quartus II and Cyclone II devices targeted. Finally, the design is compared with each other to assess the performance of the proposed architecture. Blast using systolic array architecture can increase the speed of the search process better than conventional methods.

1.2 Objective of the Project

The intention of this research project is to design and implement a systolic array for DNA sequence using the VHDL language. The requirement of this project is to speed up the performance for the sequences. The objectives of the research are:

i. To explore the use of systolic array to accelerate sequence homology search.
ii. To design Basic Local Alignment Search Tool (BLAST) algorithm using the VHDL language.
iii. To implement the design using Quartus II with Cyclone II processor.

1.3 Problem statement

The evolution of automated DNA sequencing technologies, increasing the rate at which DNA sequence data can be generated; the ability to assemble and analyse finished DNA sequences from sequence fragment data has been identified as a bottleneck in current large scale sequencing design. There is a lot of methods to do the sequencing, but which one is better? The long fragment of data need amount of time to finish the sequence alignment. So this project goal is to design systolic arrays for DNA sequence with reducing the time for DNA sequence alignment and reduce the memory space for analysing.

1.4 Scope of project

The scope of this project is to design and implement FPGA-based DNA sequence alignment accelerator

1.5 Thesis Outline

This report is arranged and distributed into five chapters. Chapter 1 has presented a brief introduction of the project mainly about DNA and Bioinformatics field, the problem statements, the objectives of the project and its scope, and the limitations identified using the proposed approach.

Chapter 2 of the dissertation includes a literature survey related to this project as per referred to previous studies and results obtained by past researchers. It also contains some important findings from past, researchers such as a review of existing database search algorithms presentations. Their respective advantages and disadvantages, with specific reference to the DNA sequence alignment, are discussed.

Together with the literature review carried out in Chapter 2, has helped with the search for systolic array architecture that could potentially improve performance and cost. Chapter 3 provides a methodology in how this project is conducted in sequence. It

also includes the development Multiple Hits Detection Module, Hits Combination Block and Ungapped Extension Block.

Chapter 4 contains the results and findings of the project. A simulation is run on a systolic array architecture for BLAST algorithm and hardware performance are compared with other architecture. Simulation result is analysed and studied.

Lastly is chapter 5 where this chapter concludes the dissertation. It presents a summary of research achievements together with a discussion of their significance. Some recommended future work also presented in this chapter.

CHAPTER 2

LITERATURE REVIEW

1.1 Introduction

Needleman-Wunsch algorithm is an algorithm used in Bioinformatics for protein or nucleotide sequence alignment. The discovery of this algorithm by Saul B. Needleman and Christian D. Wunsch in 1970 [3] was the big breakthrough in the use of dynamic programming, and was first used in dynamic programming for biological sequence comparison. These algorithms provide the optimal outcome of flesh. This algorithm is a global sequence.

In a rare alignment, nucleic acids in the x matched either nucleic acids or with a gap in y and vice versa for example, the sequence GAATTC and Gatta should be aligned according to the following scoring rules. Starting score = 0, match = 2, mismatch = -1, gap = -2. The following equation must be satisfied to get the best score:

$$F(i,j) = \begin{cases} F(i-1,j-1) + s(X_i, Y_i) \\ F(i,j-1) + d \\ F(i-1,j) + d \end{cases}$$

As a result of the global alignment technique introduced by Needleman and Wunsch, a new alignment technique was introduced by Smith and Waterman to identify the same molecules but this technique on local alignment [4]. Smith-Waterman algorithm combines evolutionary insertions and deletions techniques with network functions. After the Gotoh introducing new techniques based on the Smith-

Waterman algorithm to make a few modifications that consider fine gap penalties and techniques are still in use.

Smith-Waterman algorithm is a more detailed and based on a dynamic programming algorithm. It compares the sequence of functions based on local alignment in order to find data matches between the two sequences. Part of the Smith-Waterman algorithm is used to compute the optimal local alignment of two sequences. This procedure consists of two steps, fill in the dynamic programming matrix and find the maximum value (score) and trace back the patch that leads to the maximum score for finding the optimal alignment.

Smith-Waterman search base is the comparison of the two DNA sequences. It used pairwise comparisons between individual characters such as an equation:

$$D_{ij} = max \begin{cases} D\ (i-1), (j-1) + \ Sbt \\ D(i-1), j + d \\ D(i, (j-1) + \ d \end{cases}$$

The BLAST algorithm was used as a popular search algorithm to find the queries against a database of biological sequences [5]. BLAST is used to make comparisons between the biological sequence data, such as the amino acid sequence of each protein or nucleotides of DNA sequences. A BLAST search enables a researcher to compare a query sequence with a database of sequences, and identify library sequences that resemble the query sequence above a certain threshold. Another method for sequencing purpose maybe give optimal alignment such as Smith-Waterman [4] and Needleman- Wunsch [3] algorithms. Both are more sensitive than BLAST but it has taken a long time to search for the pattern, because of that BLAST is introduced to get better speed and reduce the time. BLAST algorithm is heuristic and gives sub-optimal results; it's different from other exhaustive dynamic programming algorithms that have optimal results. But the result from the BLAST algorithm still can be used for the sequencing purposes. Exhaustive algorithms typically have a complexity of O (mn) while heuristic algorithms usually have a complexity of O (n) [6] where *m* is the size of database sequence and *n* is the size of query sequence. The execution times for sequence alignment are very important for the researcher because the size of bio sequence databases has grown

exponentially over the years. BLAST is very popular because of high execution speed.

Based on 3 database search algorithms [3], [4] and [5], BLAST was chosen for this project because of the speed, Needleman-Wunsch and Smith-Waterman algorithms are good for sensitivity but result from BLAST still acceptable for the sequence alignment purpose.

For architectural design Cge, S. etl [8] said special purpose processors designed to speed up compute-intensive sections of applications. Two extreme endpoints in the spectrum of possible accelerators are FPGAs and GPUs, which can often achieve better performance than CPUs on certain workloads. FPGAs are highly customizable, while GPUs provide massive parallel execution resources and high memory bandwidth.

Since systolic model proposed by Kung [9]. Methods of solution for demanding problems and performance potential has attracted great attention. The systolic computation rate is limited by IO operations array, the heart of controlling the flow of blood to the cells because it is a source and a destination for all blood. Systolic arrays have been balanced. Uniform grid such as architecture in which each line shows the communication path and each intersection represents a cell or a systolic element, however, is more of a systolic array processor array that perform systolic algorithms. Systolic arrays composed of elements that take one of the forms; use a special cell with a hardwired function, vector - computer, such as a cell with command decoding and processing unit and processor equipped with a control unit and processing units.

Altera Corporation [10] already present implementations of the Smith-Waterman algorithm for both DNA and protein sequences on the platform. A multistage processing element (PE) design which significantly reduces the FPGA resource usage and allows more parallelism to be exploited; a pipelined control mechanism with uneven stage latencies a key to minimize the overall PE pipeline cycle time.

In all cases, the systolic elements or cells adapted to local communication intensive and based parallelism. Because the array is made up of cells only one or, at most, a wide range, it has the usual features and ease. Array usually extensible with minimal difficulty. Three factors have contributed to the evolution of various systolic into the main approach to handle computationally intensive applications: advances in technology, processing concurrency, and demanding scientific applications.

1.2 List of Paper Review

NO	RESEARCHER	TITLE	METHOD/DESCRIPTIONS	ADVANTAGES
1	Stephen F. Ailtshcul, Warren Gish, Webb Miller, Eugene W. Myers, David J. Lipman	Basic local alignment search tool	1. The maximal segment pair measure 2. Rapid approximation of scores 3. Implementation	The concept underlying BLAST is simple and robust and therefore can be implemented in a number of ways and utilized in a variety of contexts.
2	Kasap, S. , Benkrid, K., Ying Liu	High performance FPGA-based core for BLAST sequence alignment with the two-hit method	The design and implementation of a high performance FPGA-based core for BLAST sequence alignment with the two-hit method.	Real hardware implementations show that our core is 52 times faster than equivalent software implementations, on average
3	Needleman, S. and Wunsch, C.	A general method applicable to the search for similarities in the amino acid sequence of two sequences	A computer adaptable method for finding similarities in the amino acid sequences of two proteins has been developed. From these findings it is possible to determine whether significant homology exists between the proteins. This information is used to trace their possible evolutionary development.	Comparisons are made from the smallest unit of significance, a pair of amino acids, one from each protein. All possible pairs are represented by a two-dimensional array, and all possible comparisons are represented by pathways through the array.
4	Smith, T.F. and Waterman, .S.	Identification of Common Molecular Subsequences	A new alignment technique was introduced by Smith and Waterman to identify the same molecules but this technique on local alignment.	Based on Needleman and Wunsch algorithm but just focus on local alignment. Better speed than Needleman and Wunsch.
5	Herbordt, M.C., Model, J., Gu, Y.F., Sukhwani, B. and VanCourt, T.	Single Pass, BLAST-Like, Approximate String Matching on FPGAs	Approximate string matching (ASM) is a fundamental operation of Bioinformatics • High performance ASM is important not only for everyday queries, but also for integration as a subroutine into larger applications • HW BLAST has been challenging because it depends on random access into GB scale databases	More information about BLAST can get from this paper.

6	Euripides Sotiriades, Christos zozanitis, Apostolos Dollas	FPGA based architecture for DNA sequence comparison and database search	This research is about BLAST algorithm. The new architecture was fully designed, placed and routed. The post place-and-route cycle-accurate simulation, accounting for the I/O, shows a better performance than a cluster of workstations running highly optimized code over identical datasets.	More specifically this implementation contains several memories which have been all implemented using BRAMs.
7	F. Zhang, X-Z. Qiao, Z-Y. Liu	A parallel Smith-Waterman algorithm based on divide and conquer	Develop a new parallel Smith-Waterman algorithm using the method of divide and conquer, named PSW-DC	Memory space required in the new parallel algorithm is reduced significantly in comparison with existing ones. A key technique, named the C&E method, is developed for implementation of the new parallel Smith-Waterman algorithm.
8	Cge, S., Li, J., Sheadder, J.W., Skadron, K. And Lach, J.,	Accelerating Computer-Intensive Applications With GPUs and FPGAs	Special purpose processors designed to speed up compute-intensive sections of applications. Two extreme endpoints in the spectrum of possible accelerators are FPGAs and GPUs, which can often achieve better performance than CPUs on certain workloads. FPGAs are highly customizable, while GPUs provide massive parallel execution resources and high memory bandwidth.	Perform a comparative study of application behaviour on accelerators considering performance and code complexity. Based on results, present an application characteristic to accelerate platform mapping, which can aid developers in selecting an appropriate target architecture for their chosen application.

9	Xia, F., Dou, Y. and Xu, J.Q.,	Families of FPGA-Based Accelerators for BLAST Algorithm with Multi-seeds Detection and Parallel Extension	Propose a systolic array approach to detect string matches without using lookup tables. The pipelining systolic array is implemented as a multi-seeds detection and parallel extension pipeline engine to accelerate the first two stages of NCBI BLAST famil y algorithms.	Achieves superior performance results in both of processing element number and clock frequency over related works in the area of FPGA BLAST accelerators.
10	Altera Corporation	Implementation of the Smith-Waterman Algorithm on a Reconfigurable Supercomputin g Platform	Present implementations of the Smith-Waterman algorithm for both DNA and protein sequences on the platform.	A multistage processing element (PE) design which significantly reduces the FPGA resource usage and allows more parallelism to be exploited; a pipelined control mechanism with uneven stage latencies—a key to minimize the overall PE pipeline cycle time; and a compressed substitution matrix storage structure, resulting

CHAPTER 3

METHODOLOGY

3.1 Introduction

It is a well known fact that the most important step in the research process is to define the problem as well as discussed the methods of study. Research methods are referring to how the researcher gets the information and analysed the result based on the research objective. At this point, the information and details of the flowchart sequence on how the project is performed and an explanation of each step is also provided.

For this project, Altera Quartus II 9.1 is used to perform the simulations in order to apply the design architecture for the systolic array architecture. ModelSim- Altera 10.1d provided a waveform based on simulation results from Quartus II. Quartus II also model of block diagrams can be created. This project focuses on the design and implementation of FPGA-BASED DNA sequence alignment accelerator. The DNA sequence alignment in this project uses systolic array architecture to increase the speed of sequential processes.

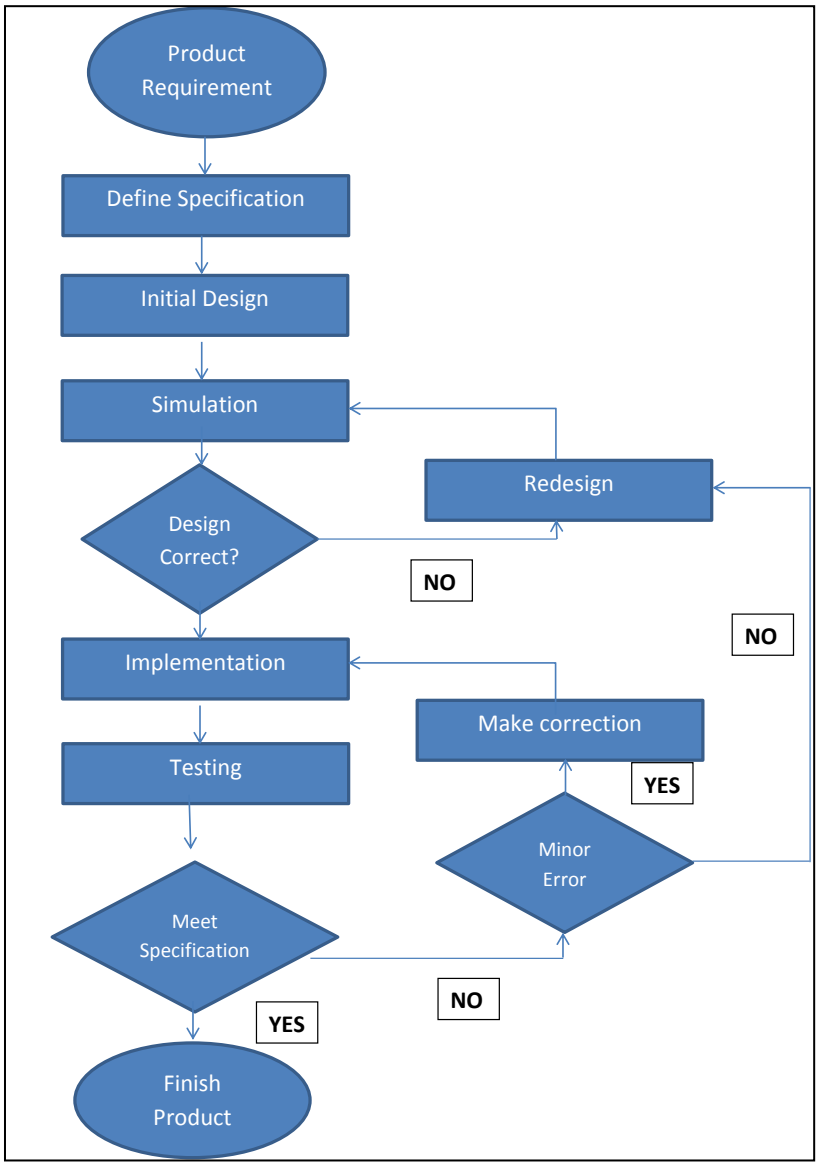

Figure 3.1: Project Flowchart

Figure 3.1 shows that at the early stage, the project starts by gathering information or literature research of the project. It starts by searching about the sequencing method, type of architecture to get optimal results and software that to be used in order to gain more knowledge about the project. As a result of this research, a suitable method and architecture is obtained to get better speed for sequencing the DNA from the database. All of the activities are organized in the literature review part.

The next stage is writing the VHDL code and performing the compilation using the Quartus II and doing the simulations in the ModelSim-Altera working platform. Several different types of sequencing method are simulated with their own architecture. If there is an error, the coding needs to recheck and make the correction in order to produce good output. Finally, the system is applied to the analysis of DNA databases and compared it with a previous sequencing method to see the differences. The final stage is a report writing for this project.

3.2 System Block Diagram

A block diagram as shown in Figure 3.1 is basically the flow of this project. It consist of three main parts, writing VHDL code for BLAST algorithm, designing systolic array architecture and implementation of Altera DE2-70 FPGA board. These three parts are the main systems that will be focusing on this project.

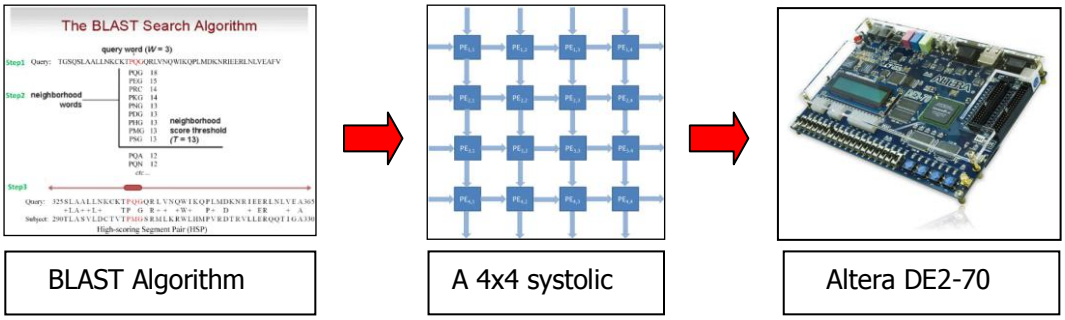

Figure 3.2: BLAST algorithm with systolic array architecture

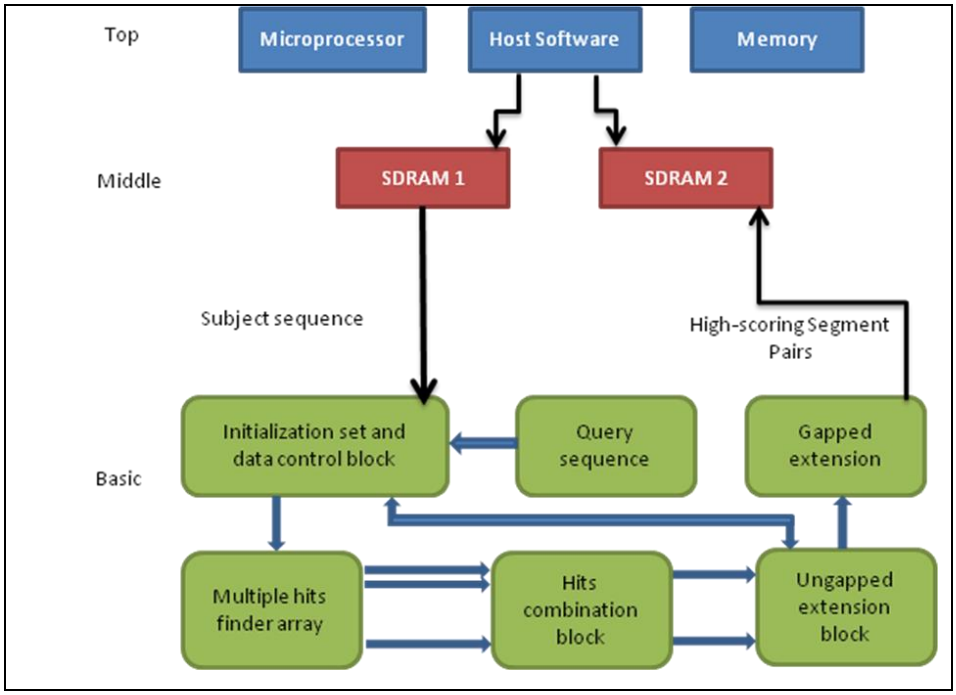

Figure 3.3: Block diagram of the simulated system

From Figure 3.2 shows DNA sequence alignment system with a systolic array architecture consists of three layers. The top layer is a host computer which runs Altera Quartus II and ModelSim for design and simulation purpose. The middle layer consists of two SDRAM. The basic layer is the systolic array-based parallel architecture fitted on FPGA. The main part of this system is the basic layer because all sequencing and comparing process occurs in this layer.

For the top layer of the system, an HP Compaq 8200 Elite desktop computer with Intel Core i5 microprocessor and 4GB RAM to run the host software. In the middle later, two 1 GB SDRAMs are attached to the FPGA where the basic layer is located.

The basic layer is the systolic array-based parallel architecture that implements the BLAST algorithm. This layer mapped to the Cyclone II chip on the Altera DE2 FPGA board. This layer contains six blocks with various functions. The Initialization Set & Data Control Block is used to initialize the status of the registers for the Hit Finder Array and to forward subject sequence as well as a query sequence to the Hit Finder Array. The Query Sequence Memory storing query sequence is constructed from

internal block RAMs. The four other blocks implement the three steps of the BLAST algorithm in parallel. Hits Finder Arrays can detect multiple hits per clock cycle. The Hits Combination Blocks can find combinable overlapping hits and combine them. For example, it can find adjoined overlapping hits “TGA” and “GAC” then combine them into “TGAC”. Ungapped Extension Blocks extend ungapped hits in both directions until the score meets a thread value T. Gapped Extension Block extension gapped on Hit-scoring Sequence Pairs (HSPs) through an exhaustive algorithm. The block outputs HSPs and alignment scores.

3.3 Multiple Hits Detection Module

The first approach towards a parallel implementation of a Multiple Hits Detection Module is by using two-dimensional systolic arrays. With some analysis of the data and the way systolic array synchronization process data, a two-dimensional implementation was designed. A common systolic array structure is the rectangle to be more appropriate based on way data has flowed and the operations to process computationally intensive tasks. The two-dimensional architecture of systolic array as shown in Figure 3.3.

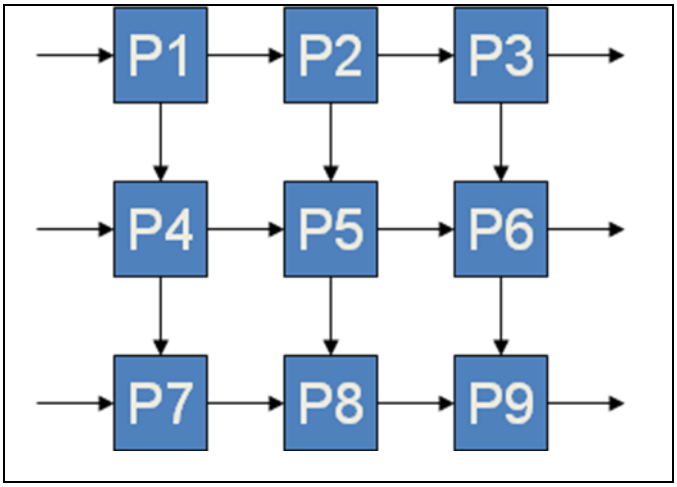

Figure 3.4: Two-dimensional systolic array

Arrows represent directions of data flowing. In this architecture, data flow synchronously across two-dimensional network, usually with different data flowing in different directions. Each unit at each step takes in data from one or more neighbour (up and right), processes the data, and in the next step outputs results in the opposite direction (down and left).

3.3.1 Multiple Hits Detection Architecture

Multiple Hits Detection Module is used to detect 3-word hits and record hits addresses on the query and the subject sequence. This design can detect more than one hits in only one cycle, that is why less time is used. The architecture of Multiple Hits Detection Module is shown in Figure 3.4.

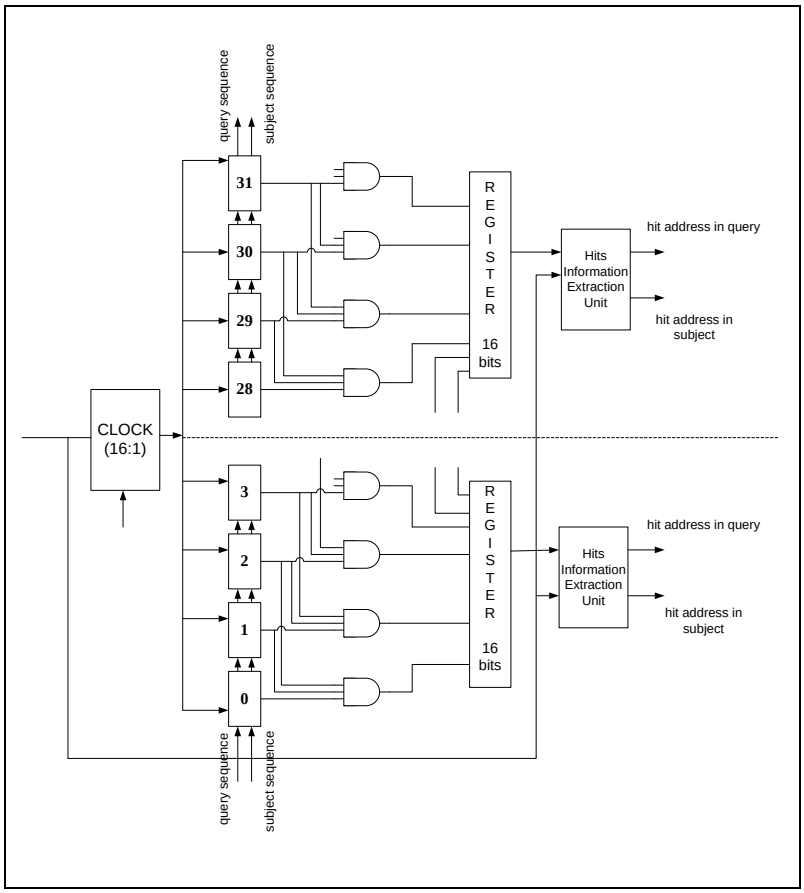

Figure 3.5: Architecture of the Multiple Hits Detection Module

From the Figure 3.4, the systolic array has 32 processing units, every three units are connected to one 3-input AND gates. Every 16 gates outputs are connected to 16-bit register. The value of both registers is sent to the corresponding Hits Information Extraction Units for recording the hits address in the query and the subject sequence. For this system a Clock Division Block coordinates the systolic array and the Hits Information Extraction Units.

A whole system start when a query sequence with 32 characters is forwarded into the systolic array that means each processing unit holds a character from the query sequence. Then the subject sequence drove into the systolic array by each internal clock rising edge. Meanwhile, the incoming subject character and the query character which is held by the unit are compared, if they are identical, the logic '1' would be generated and otherwise, and the logic '0' would be generated. The comparison result is an input of a 3-input AND gate. Every three processing units, maps to an AND gate. A hit is detected when the logic '1' is generated from its output. So, the systolic array with 3-input AND gates can detect multiple hits at one internal clock rising edge. The architecture of the processing unit in the systolic array is illustrated in Figure 3.5.

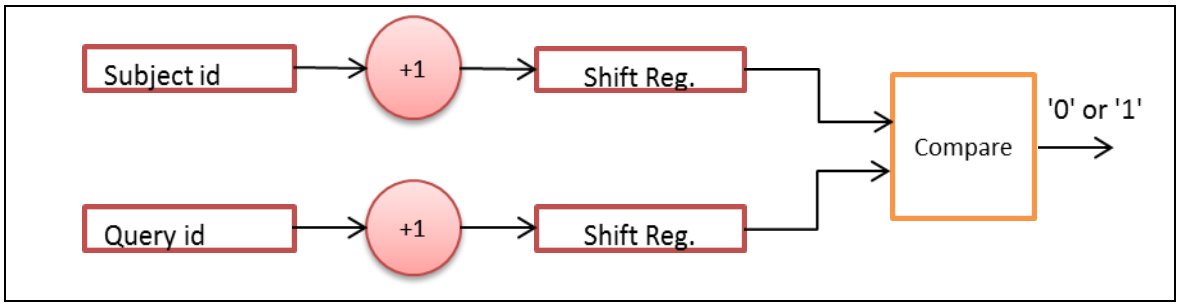

Figure 3.6: Architecture of Processing Unit of Multiple Hits Detection Module

In implementation, 20 3-bit long binary numbers ("001" to "100") are defined to present 4 nucleotides in a DNA. Shift registers are able to shift characters in the adjacent processing unit so that the query and the subject sequence can go through the systolic array. When a new character from the query or the subject sequence is shifted into the unit, the corresponding id register would increase its value by 1. Each processing unit of the systolic array will compare two characters in it at each internal clock rising.

The Multiple Hits Detection Module is a parallel, pipelined architecture. The systolic array with 32 processing units cooperates with 32 3-inputs AND gates to detect hits in both sequences. The responsibility of Hits Information Extraction Block is recording those hits' locations. The two processes in the module are driven by two clocks.

3.3.2 Behaviour Simulation Multiple Hits Detection Module

Several experiments were made to the behaviour simulation of Multiple Hits Detection Module. 3-bit binary represents one kind of nuclide acid in a DNA. Table 3.1 describes the names of nucleic acid and their 3-bit binary.

Table 3.1: 3-bit binary description for nucleic acids in a DNA

Name of the nucleotide acid	Acronym	3-bit binary
Guanine	G	001
Adenine	A	010
Thymine	T	011
Cytosine	C	100

Multiple Hits Detection Module is actually a systolic array-architecture. Several experiments were designed to show the correctness of the behaviour function of the architecture. For example:

CTATGATGATAAATGATTATATAATCATTACG-> Query sequence
ATTAATTGAACTCATATTTATATAATAAGCGA-> Subject sequence

In order to show hits positions clearly, three systolic array statuses in which hits are existing are listed as below. Each character in query sequence is in one processing unit of the systolic array, each character in subject sequence is being forwarded into the systolic array.

Systolic array status 1:

Query: GCATTACTAATATATTAGTAAATCAGAGTAGT
Subject: TTA

Hit address in a query sequence is 16 and subject sequence is 2.

Systolic array status 2:

Query: GCATTACTAATATATTAGTAAATAGTAGTATC
Subject: AGCGAATAATATATTTATACTCAAGTTAATTA

Hit address in a query and subject sequence is 25.

3.4 Hits Combination Block

The Multiple Hits Detection Module may output a large amount of hits per clock cycle if there is high similarity between query and subject sequence. Ungapped Extension Block is relatively slower than Multiple Hits Detection Module and the systolic array in this situation. Hence, the speed between both stages is unbalanced.

From the previous research [10] [11] suggested that in many cases there is a high similarity between the query sequence and the subject sequence. Overlapping occurred will waste execution cycle for Multiple Hits Detection Module and consumes additional extension time for Ungapped Extension Block because one hits extended twice. For example, two adjacent hits “ATC” and “GAT” are found, but they are actually only one hit “GATC”. This hit will be recorded twice in Multiple Hits Detection and extended twice in Ungapped Extension Block. That is why the implementation of Hits Combination Block is needed because it can detect the overlapping hits and merge them to reduce verbose its and maintain the sensitivity of BLAST.

This block contains an Hits (First In First Out) FIFO buffer which is used to store hits location addresses from both query and subject sequence. The algorithm implementation engine executes hits combination algorithm the forward the hit addresses and hit length of the Ungapped Extension Block. The data flow in this block is shown in Figure 3.6.

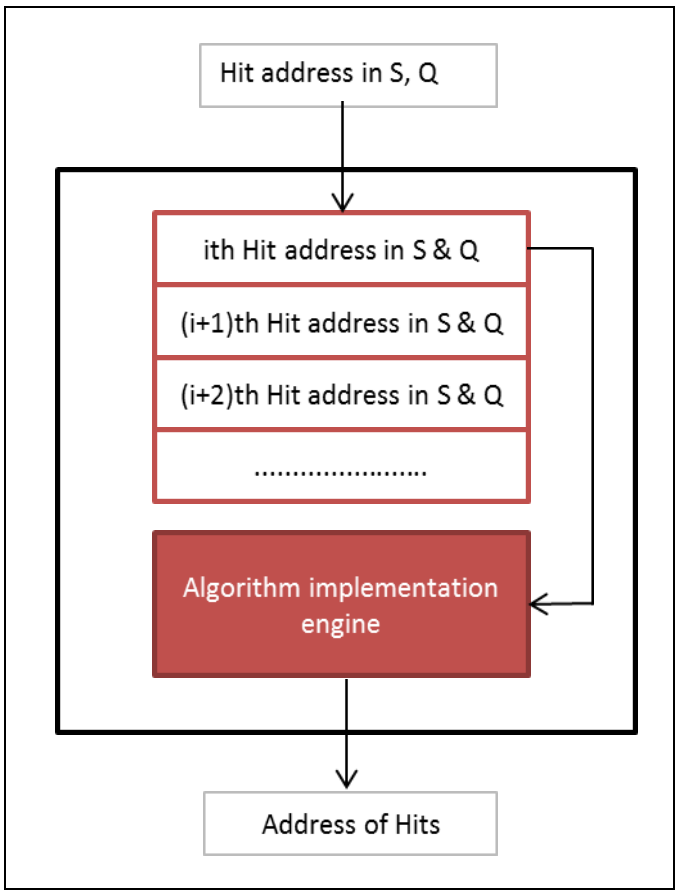

Figure 3.6: Hits Combination Block

The arrows show how the order of the Hit addresses in the Hits FIFO buffer. For example, the address “ATC” should be stored in the FIFO buffers first, then the “GAT”. The combination algorithm is triggered when there is more than one record in the Hits FIFO buffer. Here, the hit addresses in subject sequence are recorded and forwarded to the following operations. First, the algorithm implementation engine calculates the ending address of the earth and (i+1) the hit in the subject sequence. For this project the

starting address is the address which is recorded minus 1, because the recorded address is the address of the middle word in a 3-word hit. The ending address is the address of the last character in a sequence, so (Ending address = Starting address + (sequence length -1)). Here, the sequence length is equal to 3. Then a comparison between ith hit's ending address and (i+1)th hit's starting address is performed. If ith hit's ending address is greater than or equal to the (i+1)th hit's starting address, there will be an overlap between two hits. The algorithm can determine new addresses in both subject sequence and query sequence of the merged hit and calculate its length. Then the algorithm records the new address and hit length at the ith place in Hit's FIFO buffer. If ith hit's ending address is less than the (i+1)th hit's starting address, the (i+1)th hit address will be chosen to be compared with (i+1)th hit address. This operation continues until the Hits FIFO buffer is empty.

3.5 Parallel Architecture for Multiple Hits Detection Module and Hits Combination Block

The speed for hit detection can be increased if Multiple Hits Detection Module can work together with Hits Combination Block in parallel architecture that shown in Figure 3.7.

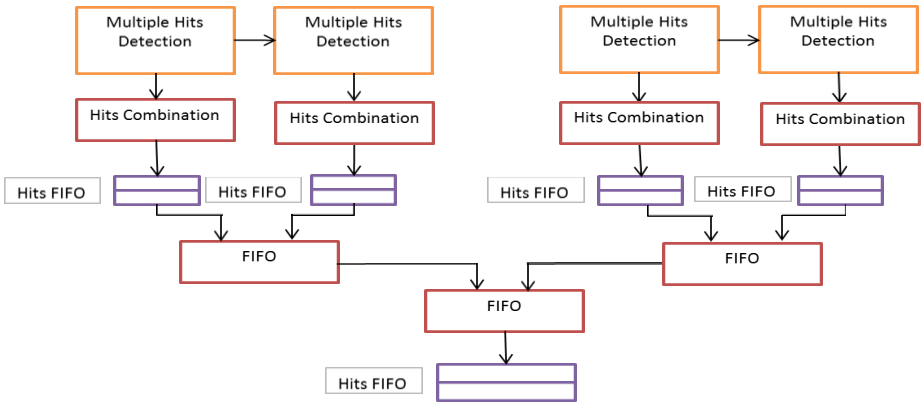

Figure 3.7: Parallel Architecture Multiple Hits Finder Array and Hits Combination Block

The large systolic array contains many Multiple Hits Detection Modules. Each module matches one Hits Combination Block, each of which records and combines hits detected by its mapped module. After that, it sends the hits information to a local hits FIFO. Then FIFO Combination modules combine hits FIFOs and deliver hits information to larger hits FIFOs. In this architecture, Multiple Hits Detection Modules are connected in a serial way to extend the array comparison size. Each Hits Combination Block map for a Multiple Hits Detection Module so that each block can detect overlapping hits and merge them simultaneously. The parallel implementation trades on-chip resource for speed.

3.6 Ungapped Extension Block

This block implements the ungapped extension step of the BLAST algorithm. Hits information will be stored by FIFO. The score matrix BLAST Nucleotide Matrix is applied to find the alignment score for a pair of words. If this block detects a hit in a FIFO, the address of the hit word in query sequence and subject sequence and the hit length would be extracted from the FIFO to compute both extension points. After this step, the Ungapped Extension Algorithm Implementation Engine can finish the ungapped extension step. The data flow in the Ungapped Extension Block is shown in Figure 3.8.

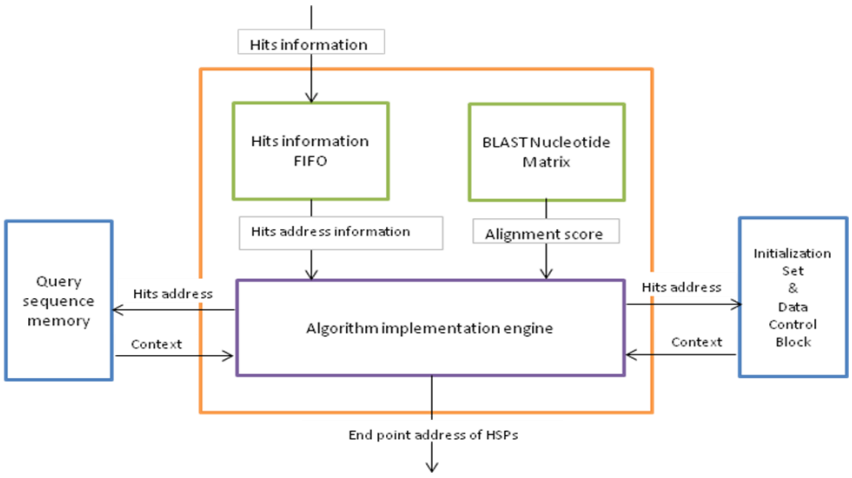

Figure 3.8: Ungapped Extension Block

From the figure, the hits information is entered the FIFO. Addresses of hits in query sequence and subject sequences included in this information. The algorithm implementation engine will calculate the start points of the extension in the query sequence and the subject sequence.

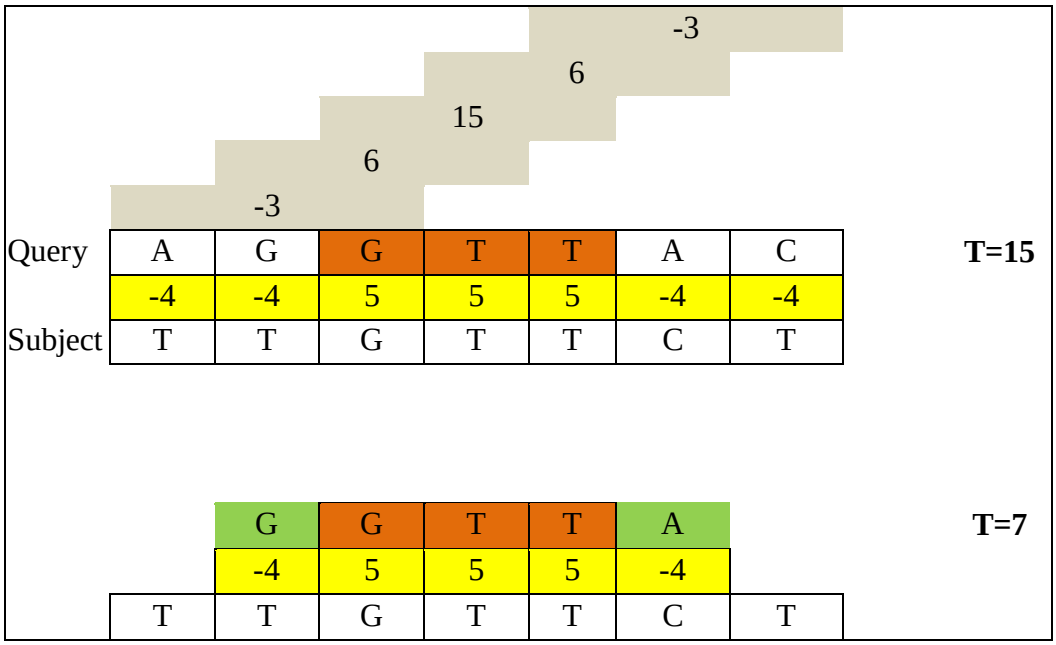

Figure 3.9: Extension procedure

REFERENCES

1. Stephen F. Ailtshcul, Warren Gish, Webb Miller, Eugene W. Myers, David J. Lipman (1990), Basic local alignment search tool, *Journal of Molecular Biology, Volume 215, Issue 3,* pp. 403-410.
2. Sudhir Kumar (2005), Molecular clocks: four decades of evolution, *Nature Reviews Genetics, Volume 6,* pp. 654-662.
3. Needleman, S. and Wunsch, C. (1970), A general method applicable to the search for similarities in the amino acid sequence of two sequences", *Journal of Molecular Biology, Volume 48(3),* pp. 443-453.
4. Smith, T.F. and Waterman, M.S.," Identification of Common Molecular Subsequences", *Journal of Molecular Biology, Volume 147,* pp 195-197.
5. Altschul, S. and Madden T. (1981), "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs", *Nucleic Acids Res, Volume 25 (17),* pp. 3389-402.
6. Herbordt, M.C., Model, J., Gu, Y.F., Sukhwani, B. and VanCourt, T. (2006), " Single Pass, BLAST-Like, Approximate String Matching on FPGAs", *14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines,* pp. 217-226.
7. F. Zhang, X-Z. Qiao, Z-Y. Liu (2002), "A parallel Smith-Waterman algorithm based on divide and conquer," *ICA3PP '02*.
8. Needleman, Saul B. and Wunsch, Christian D. (1970), " A general method applicable to the search for similarities in the amino acid sequence of two proteins". *Journal of Molecular Biology, Volume 48,* pp. 443-453
9. H.T. Kung. (1982), "Why Systolic Architectures?"*Cotnpurer, Volume 15. No. 1,* pp. 37-46.
10. Rao, A.A. And Visakhapatanam (2007), " A Tool for 8m, kilo Pairwise Alignment Algorithm", *Management Review: An International Journal, Volume 2 (1),* pp. 28-40.
11. Cge, S., Li, J., Sheadder, J.W., Skadron, K. And Lach, J. (2008), " Accelerating Computer-Intensive Applications With GPUs and FPGAs", *Symposium on Application Specific Processors,* pp. 101-107.

12. Xia, F., Dou, Y. and Xu, J.Q. (2008), "Families of FPGA-Based Accelerators for BLAST Algorithm with Multi-seeds Detection and Parallel Extension", *CCIS 13*, pp. 43-57.
13. M.C. Herbordt, J. Model, Gu Yongfeng, B. Sukhwani and T. VanCourt (2006), "Single Pass, BLAST-Like, Approximate String Matching on FPGAs", *Proc. 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines* , pp. 217–226.
14. S.A.M Al Junid, Z. Abed Majid, A.K Halim (2008), "Development of DNA Sequencing Accelerator based on Smith-Waterman Algorithm with Heuristic Divide and Conquer Technique for FPGA Implementation". ICA3 23.
15. F. Zhang, X-Z. Qiao, Z-Y. Liu (2002), " A parallel Smith-Waterman Algorithm based on divide and conquer", ICA3 02.
16. T. Oliver T Oliver, B Schmidt, D Maskelll. (2005) "Hyper Customized Processors for Bio-Sequence Database Scanning on FPGAs", *Proceedings of the ACM/SIGDA 13th international symposium on Field programmable gate arrays (FPGA)*, pp 229 – 237.
17. Sidhu, R.; Prasanna, V.K. (2001) "Fast Regular Expression Matching Using FPGAs", *Proceedings of the 9th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp 227 – 238.
18. Hutchings, B.L.; Franklin, R.; Carver, D. (2002)," Assisting network intrusion detection with reconfigurable hardware", *Proceedings of the 10th Annual IEEE Symposium on Field- Programmable Custom Computing Machines*, pp 111-120.
19. S. Guccione and Eric Keller.(2002), "Gene Matching Using JBits". *Proceedings of the 12th International Conference on Field- Programmable Logic and Applications, Lecture Notes In Computer Science; Volume 2438*, pp 1168-1171.
20. K. Muriki, K.D Underwood, R. Sass (2005). "RC-BLAST: Towards a Portable, Cost-Effective Open Source Hardware Implementation", *Proceedings 19th IEEE International Symposium Parallel and Distributed Processing (IPDPS)*, pp 196b-196b.
21. Sotiriades, E.,Dollas (2007),'A General Reconfigurable Architecture for the BLAST Algorithm', *Journal of VLSI Signal Processing; Volume 48*, pp 189–208.

22. D.A. Benson, I. Karsch-Mizrachi, D.J. Lipman, J. Ostell, D.L. Wheeler,(2005), "GenBank, Nucleic Acids Res.", Jan 1; *Volume 33* (Database issue):D34-8,.

23. Jacobi, R.P., Ayala-Rinc´on, M., Carvalho, L.G.A., Llanos, C.H., Hartenstein, R.W., (2005) "Reconfigurable systems for sequence alignment and for general Dynamic programming." *Genetics and Molecular Research, 4. Volume 3,* pp. 543–552.
24. G. Tempesti, D. Mange, A. Stauffer, and C. Teuscher., (2002)," The BioWall: An electronic tissue for prototyping bio-inspired systems." *Proceedings of 2002 NASA/DoD Conference on Evolvable Hardware, IEEE,* pp. 221–230.
25. Kuon, I. and Rose, J. (2007), "Measuring the gap between FPGAs and ASICs." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 26.Volume 2*, pp. 203–215.
26. Malhis, N., Butterfield, Y.S.N., Ester, M., and Jones, S.J.M., (2009), "Slider maximum use of probability information for alignment of short sequence reads and SNP detection." *Bioinformatics Volume 25*, pp.1- 6.
27. Trapnell, C. and Salzberg, S.L., (2009), "How to map billions of short reads onto genomes." *Nature biotechnology Volume 27*. pp. 5- 455.
28. T.Oliver, B. Schmidt, Y. Jakop and D.L. Maskell, (2006), "Accelerating the Viterbi algorithm for profile Hidden Markov Models using reconfigurable hardware". *Lecture Notes in Computer Science, Springer-Verlag, Volume 3991*, pp. 522–529.
29. Y. Yamaguchi, Y. Miyajima, T. Maruyama and A. Konagaya, (2002), "High speed homology search with FPGAs". In *Pacific Symposium on Biocomputing*, pp. 271– 282.
30. D. J. Lipman and W. R. Pearson, (1985), "*Rapid and Sensitive Protein Similarity Searches*," *Science*, Volume 227, pp. 1435-41.