Reconfigurable Hardware-Software Co-Design Methodology for Protein Identification

Venkateshwarlu Y. Gudur, Sandeep Thallada, Abhinay R. Deevi, Venkata Krishna Gande, Amit Acharyya, *Member, IEEE*, Vasundhra Bhandari, Paresh Sharma, and Ganesh R. Naik, *Senior Member, IEEE*, *Saqib Khursheed*

***Abstract*— In this paper we propose an on-the-fly reconfigurable hardware-software co-design based reconfigurable solution for real-time protein identification Reconfigurable string matching is performed in the disciplines of protein identification and biomarkers discovery. With the generation of plethora of sequenced data and number of bio-markers for several diseases, it is becoming necessary to have an accelerated processing and on-the-fly reconfigurable system design methodology to bring flexibility to its usage in the medical science community without the need of changing the entire hardware every time with the advent of new bio-marker or protein. The proteome database of human at UniProtKB (Proteome ID up000005640) comprising of 20192 reviewed proteins and 42132 canonical, and isoform proteins with variable database-size are used for testing the proposed design and the performance of the proposed system has been found to compare favorably with the state-of-the art approaches with the additional advantage of real-time re-configurability.**

***Keywords*— Reconfigurable systems, protein identification, string matching, hardware-software codesign, finite state machines.**

I. Introduction

Recently new technologies and research in computational bioinformatics have revolutionized the rate of biological data generation. A vast amount of proteomics and genomics data is contributed to the life science society by researchers especially in the domain of high-throughput next generation sequencing methods and it is doubling at every 18 months [1]. Protein identification is a fundamental step in protein sequence analysis and it needs efficient solutions to match the data growth. Rapid methods are focused in the quest for faster protein sequence analysis to scan databases and identify a protein accurately [2]. This benefits the discipline of disease biomarker identification and aid disease diagnosis and prognosis [3].

Protein identification using peptide fragments obtained by mass spectrometry involves database searching that is similar to string matching [4]. In string matching a database or text is searched to find locations of one or more strings also called patterns. String matching algorithms like Boyer-Moore [5] and Knuth-Morris-Pratt (KMP) [6] search single pattern strings in a larger string. These approaches have a requirement of high computational complexity [7]. Aho-Corasick algorithm (ACA) is a widely used multi-pattern string matching algorithm that has a linear computational complexity [8][15]. Hardware accelerated solutions for protein identification are used to address the bottlenecks in the computational biology pipeline [4][7][9]. Hardware-Software codesign approach is used for reconfigurable string matching and simultaneously harness the advantages of both hardware and software [10][11]. In [10] the number of proteins in database used for testing is very small and it is not targeted for any real life application and finite state machine (FSM) logic need to be designed for different pattern sets. An effort to test the design in a real life scenario is proposed in [11] and KMP algorithm [6] is used for string matching in the design. Their solution is inherently slower as it uses KMP algorithm over ACA algorithm for string matching.

In this paper we propose an on-the-fly reconfigurable hardware-software co-design based reconfigurable solution for protein identification in real-time. We use on-chip memory based implementation to realize FSM design using ACA algorithm. We demonstrate the way proposed design can be used in real life with plethora of test cases. To the best of our knowledge, the proposed methodology is the first of its kind where implementing an accelerated and reconfigurable multi-pattern string matching platform does not require any step of fixed hardware system design when used in an application making it re-configurable enabling the sequencing with any number of bio-markers for as many diseases as possible. The rest of the paper is organized as follows. Section II presents an overview of hardware accelerated solutions in bioinformatics and applications of string matching in the same. It also includes literature about ACA in various disciplines of bioinformatics, memory based implementation of FSMs and hardware-software codesign approaches. The proposed solution along with implementation is covered in section III. Experimental results obtained during the study are presented in section IV. Finally conclusions are summarized in Section V.

II. Background and Related Work

High performance computing solutions for computational bioinformatics that include multiprocessor, CPU with multicores, cluster, cloud, grid and heterogeneous computing involve huge computational infrastructure and their management is a very costly affair [12]. An alternate cost effective but promising solution is the use of hardware

V.Y.G., S.T., A.R.D., V.K.G., and A.A. are with Department of Electrical Engineering, Indian Institute of Technology Hyderabad, 502285, Telangana, India (e-mail: ee15resch02009, ee13m1028, ee13b1010, ee13b1011, amit_acharyya)@iith.ac.in). P.S. is with National Institute of Animal Biotechnology (NIAB), Miyapur, Hyderabad, 500049, Telangana, India (e-mail: paresh@niab.org.in). V.B. is with Department of Animal Biology, School of Life Sciences, University of Hyderabad, 500019, Telangana, India (e-mail: vasundhra23@gmail.com). G.R.N is with Faculty of Engineering and Information Technology, University of Technology, Sydney, Broadway NSW 2007, Australia (e-mail: Ganesh.Naik@uts.edu.au). s.khursheed@liverpool.ac.uk

accelerators [13]. In the use of hardware accelerators most of the existing work is inclined towards graphical processing units (GPUs) and field programmable gate arrays (FPGAs) than application specific integrated circuit (ASIC), System on a chip (SoC) and multiprocessor SoC [9][13]. The reconfigurable and parallelism nature of FPGAs make them best suited for acceleration of bioinformatics disciplines that require modification of system as the application demands [13]. For acceleration in the domains of computational bioinformatics like pairwise sequence alignment, multiple sequence alignment, resequencing, gene-sequence analysis, DNA sequencing algorithms, database searching, genome assembly and study of homologous sequences, FPGAs are used [13]. Many disciplines in bioinformatics are profited by the research in faster string matching solutions [2][3][14][15]. Biomarker discovery using MS-based proteomics, basic local alignment search tool, proteogenomic mapping in which genome annotation is performed using proteomics, homologous series detection, sequence alignment and sequence similarity search are few of the disciplines that are benefitted.

Aho-Corasick algorithm is the most widely used multiple string matching algorithm in computational biology. The linear processing time of search has popularized its use. In this algorithm a finite state machine is constructed for a set of pattern strings to be searched and then in a single pass a text string is processed [8]. A hardware accelerated solution using FPGAs is proposed for searching MS/MS-derived query peptides in genome database [4]. For fast alignment of large genomic sequences, a simplified version of ACA is presented in [16]. The SITEBLAST algorithm of [17] uses ACA for local alignment of genomics sequence using prior knowledge. ACA performs the best of many string matching algorithms used in [15] to locate unique oligonucleotides from DNA databases. ACA algorithm is implemented using bit split architecture in [7] and it is used for matching peptides obtained by mass spectrometry against a genome translated in all six reading frames. A method is proposed for reducing the memory size and multiplexer complexity using don't care values of the inputs in [18] for on-chip memory based FSM implementation in FPGAs. RAM-based implementation of FSMs in FPGAs is proposed in [19] and performance of FSMs for various characteristics of FPGAs is studied in it. Though FSMs are present at the heart of ACA, memory based ACA-FSM implementations are limited by large memories needed to store FSMs tables. An attempt to implement ACA at a reduced memory usage is presented in [7][10]. It is limited by the resources available on FPGA. Hardware-software codesign is the design of a system where hardware and software modules interact with each other to perform a complete task [20]. It gives the advantages of both hardware and software such as speed, power and parallelism of former and flexibility, modularity of later [20]. Design constraints such as cost, performance, complexity, and power of the system are optimized and time-to-market is reduced by combining the rewards offered by hardware and software [21]. FPGAs have an advantage of including both hardware and software resources and more recently, FPGAs like Zynq-7000 family include system on a chip that have the capability to implement a complete hardware-software system on a single platform [22]. A hardware-software codesign approach for implementing ACA algorithm is presented in [10] and the advantages of software being flexible and hardware being speedy are employed. A small set of peptides and proteins are taken for verification. No real life application is targeted. For different pattern sets hardware for FSM logic need to be created by following steps for hardware system design. A hardware-software based Knuth Morris Pratt (KMP) algorithm [6] for searching a biomarker against a protein database is presented in [11]. Data transfer between host processor and reconfigurable logic is optimized. The KMP algorithm used here is slower than ACA [15]. It also uses direct memory access (DMA) components that add up additional hardware and related power.

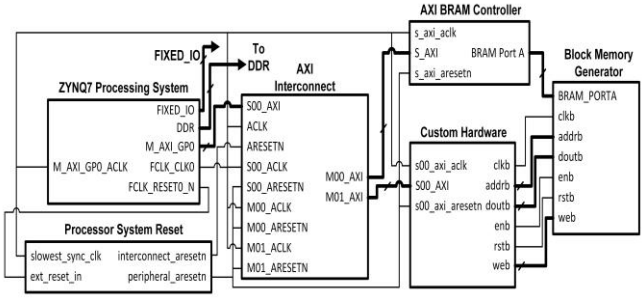

Figure 1. Architecture of the proposed system

III. Proposed System Design and Implementation

A. *Motivation:*

Reconfigurable string matching is required at bioinformatics disciplines where patterns to be searched are changed, for example, a newly discovered biomarker is added into the database of known biomarkers. Multiple biomarkers can be searched in a given sample and many diseases can be found simultaneously. Pure hardware solution is not feasible in this scenario where patterns to be searched are updated continuously. All the steps of a hardware system design are necessarily run in pure hardware solutions. In a hardware system design steps like writing programs in hardware description language (HDL), synthesis, translation, mapping, place and route, programming file generation and configuring FPGA using bitstream file are run. In the scenario of only hardware solution, these steps are repetitively carried that add substantial amount design time. These require dedicated computer systems with sophisticated proprietary tools. Repetitive running of these steps can be avoided by employing reconfigurable systems with intelligent hardware-software partitioning and codesign. We use hardware-software codesign approach in our design.

B. *Proposed methodology*

The architectural diagram of the proposed system is depicted in Fig. 1. The architecture has a Zynq-7000 All Programmable SoC (AP SoC) ZYNQ7 processing system as the master. The FPGA device on board is XC7Z020 and it has an ARM Cortex-A9 MPCore CPU. A processor system reset module is used to reset and synchronize all modules in the design by resetting them as per the reset conditions given by master at its input. Advanced eXtensible Interface (AXI) interconnect is used to interface the memory-mapped ZYNQ7 master with the memory-mapped slaves. Block memory generator module is used for generating block memory. Block memory is used for storing data during

program run time. AXI block RAM (BRAM) controller acts as a bridge to communicate between ZYNQ7 via AXI interconnect and BRAM created by block memory generator. Custom hardware module is the necessary hardware logic required to realize FSM using memory. A detailed diagram of the interface between block memory and hardware logic is depicted in Fig. 2 (a). The custom hardware is a generic hardware and along with memory implements a memory based FSM. Block memory is stored with contents that realize Aho-Corasick algorithm with the hardware logic. Depending on an input character value the corresponding input lines at the multiplexer (MUX) are activated and are available at the output of multiplexer. A state register is used to latch the multiplexer output and it feeds block memory generator with the necessary address.

We use Xilinx Vivado Design Suite software tools to design our system. Vivado Design Suite has all the features of a hardware system design along with system on a chip development. We built the design in Vivado as depicted in Fig. 1. We use Creating and Packaging Custom IP utility available in Vivado to design the hardware logic as depicted in Fig. 2 (a) and interface it to the ZYNQ7 processing system. A hardware description file (hdf) is generated in Vivado and exported along with bitstream file to Xilinx Software Development Kit (SDK) tool. Complete hardware of the system is ready before begin SDK. We write a C language application program in SDK to run on the ZYNQ7 processing system. A flowchart describing the application program execution is depicted in Fig. 2 (b). A brief idea of all the steps performed in the system and its working is described as follows.

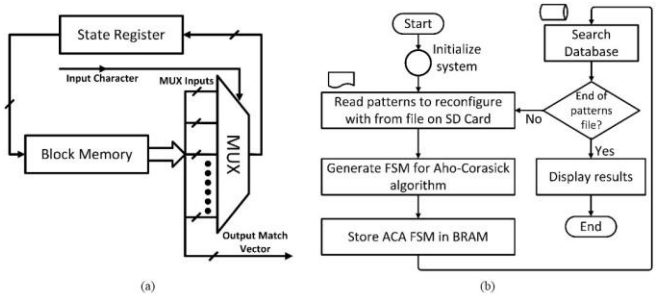

Figure 2. Proposed (a) Memory based FSM design (b) Flow diagram describing steps followed in the application program

A file containing patterns which are to be searched is stored on a portable Secure Digital High Capacity (SDHC) card with FAT32 file system. The database in which search is performed is also residing on the SDHC card. The FPGA device on board is configured with the bitstream file along with the application program. Patterns are read from patterns file and ACA algorithm is run for searching these patterns. An ACA-FSM is designed on the run as per the given patterns. We use the memory based architecture for implementing FSM proposed in [18][19] and generate block RAM contents for the corresponding FSM and store them in block RAM. The database file in which patterns are searched is read from the SDHC card and characters are received by the custom hardware. These characters act as addresses to block memory and data stored at these addresses is made available out of the block memory. Multiplexer inputs are connected to this data and part of this data is selected depending on the character fed to custom hardware. Output match vector is also a part of the data and it gives the information about the pattern found after receiving characters. The system work till the end of the database file. UniProt (Universal Protein Resource) identifier for the protein in which the patterns are found along with their corresponding location in that protein is displayed on a console window.

IV. Results and Discussion

We implemented the proposed system using Avnet Zedboard development board. From the available on-chip resources for the XC7Z020 FPGA device 9.18% FF, 12.42% LUT, 1.23% Memory LUT and 35.71% BRAM resources are used for the system. The above figure for BRAM is maximum allotted and actual BRAM utilization depends on the number of patterns and their size. Total on-chip power comprising of static and dynamic is 1.891 watt for the complete board.

For testing the system with real world data we use the UniProt Knowledgebase (UniProtKB) [24]. The proteome database of human at UniProtKB (Proteome ID up000005640) is made up of 20192 reviewed proteins. This database, made of 42132 canonical & isoform proteins, is chosen in FASTA format for testing our system. A set of well referred and standard disease biomarker proteins is selected [24]. PeptideMass, an online tool for enzymatic cleavage of proteins, is used to digest these selected disease biomarker proteins. The peptides obtained after proteins digestion act as patterns and stored in different patterns file with identifier as their file names. The proposed system is run with these peptides and the time taken for searching the complete database is noted. We also studied the effect of database size in searching these proteins. The whole human proteome database is divided into four databases of 8404, 16830, 25256 and 33682 proteins. We perform search for the previously mentioned ten pattern sets. Table I shows both the results. We see that time taken for searching database is independent of the number of patterns. We take a maximum of 32 patterns obtained after protein digestion and perform search. In case of patterns more than 32, the next 32 proteins are selected for searching. As a result of this the time taken for searching patterns greater than 32 and less than 64 in number is nearly doubled and for greater than 64 in number it is tripled. Fig. 3 shows the comparison bar chart. It is evident that time taken for search is linear and is independent of the number of patterns which is a characteristic of Aho-Corasick algorithm [8]. Table II compares the features available in the proposed design with few other designs available in literature. The proposed system does not have any limitation on the number of patterns to be searched but at the cost of time. It is a multi-pattern searching system designed intelligently with hardware-software codesign. Table I and Fig. 3 shows that the proposed system is reconfigurable with the desired patterns to be searched. This can be done during run time by selecting the respective pattern file.

TABLE I. IMPACT OF DATABASE LENGTH ON TIME FOR SEARCH

UniProt Identifier	Number of peptide	Time taken for searching (seconds)				
		DB (1)	*DB (2)*	*DB (3)*	*DB (4)*	*Full DB*
P01258	4	81.97	162.62	247.45	330.66	415.30
P61278	5	82.06	162.45	248.24	328.57	415.64
P01350	8	82.48	163.17	249.57	327.66	415.36
P15692	12	81.62	160.91	247.99	327.47	416.25
P07288	14	81.88	161.57	244.55	329.86	416.51
P01236	17	82.27	160.25	245.66	329.47	415.45
P06731	30	81.99	160.92	242.37	332.99	416.02
P03372	32	81.79	161.46	247.57	326.54	416.22
P02771	43	163.81	321.12	497.66	654.24	838.46
P02768	49	163.99	323.67	499.38	656.51	838.81
P00533	81	249.31	484.47	748.90	990.13	1264.61

**DB: Database*

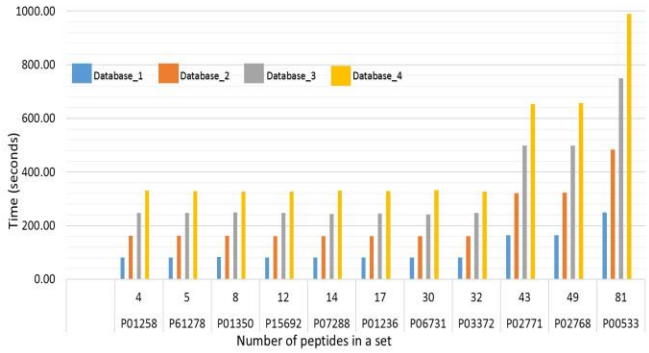

Figure 3. Effect of varying the length of database

TABLE II. FEATURES AVAILABLE IN PROPOSED SYSTEM

System Design	SMV [10]	GB [11]	Proposed design
Scheme of string matching algorithm	ACA	KMP	ACA
Simultaneous single pattern matching	✓	✓	✓
Simultaneous multi-pattern matching	✓	×	✓
Real world data verification	×	✓	✓
Hardware modules in system	FSM Logic	KMP and DMA core	Generic custom hardware
Need to run hardware system design flow repeatedly	×	✓	✓

KMP: Knuth-Morris-Pratt DMA: direct memory access

V. CONCLUSION

We presented a real-time reconfigurable string matching solution using hardware-software codesign that does not require to carry the steps of hardware system design repeatedly. The proposed method has also been validated against the variable human proteomic data-bases. The proposed system can search multiple strings in a given text in quick-time and can be employed for applications real world proteomic/genomic variable data-size for emerging bioinformatics applications.

REFERENCES

[1] Clark, K., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., & Sayers, E. W. (2016). GenBank. Nucleic Acids Research, 44(Database issue), D67–D72.

[2] W.J. Henzela, C. Watanabea and J.T. Stults, "Protein identification: the origins of peptide mass fingerprinting," Journal of the American Society for Mass Spectrometry, Vol. 14, no. 9, pp. 931-942. 2003.

[3] Sahab, Z. J., Semaan, S. M., & Qing-Xiang, A. S. (2007). Methodology and applications of disease biomarker identification in human serum. Biomarker Insights, 2, 21.

[4] T. A. Anish , M. Dumontier , J. S. Rose and C. W. V. Hogue, "Hardware-accelerated protein identification for mass spectrometry", Rapid Communi. Mass Spectrom., vol. 19, pp. 833-837, 2005.

[5] Boyer RS, Moore JS: A Fast String Searching Algorithm. Communications of the ACM 1977, 20:762-772.

[6] Knuth DE, Morris JH, Pratt VB: Fast pattern matching in strings, SIAM Journal of Computing 1977, 6:323-350.

[7] Y. Dandass, S. Burgess, M. Lawrence, and Susan Bridges. Accelerating String Set Matching in FPGA Hardware for Bioinformatic Research. BMC Bioinformatics, April, Vol. 9, No. 197, pp1471-2105, 2008.

[8] A. V. Aho and M. J. Corasick. Efficient string matching: An aid to bibliographic search. Communications of the ACM, 18(6): 33 3-340, 1975.

[9] Bogdan 2008 database search paper database search paper database search paper database search paper

[10] Vidanagamachchi, S.M.; Dewasurendra, S.D.; Ragel, R.G., "Hardware software co-design of the Aho-Corasick algorithm: Scalable for protein identification?," in Industrial and Information Systems (ICIIS), 2013 8th IEEE International Conference on , vol., no., pp.321-325, 17-20 Dec. 2013.

[11] Bianchi, G.; Casasopra, F.; Durelli, G.C.; Santambrogio, M.D., "A hardware approach to protein identification," in Biomedical Circuits and Systems Conference (BioCAS), 2015 IEEE , vol., no., pp.1-4, 22-24 Oct. 2015.

[12] E. E. Schadt, M. D. Linderman, J. Sorenson, L. Lee and G. P. Nolan, "Computational Solutions to Large-scale Data Management and Analysis," Nat Rev Genet, vol. 11, no. 9, pp.647 -657, 2010.

[13] Aluru, S.; Jammula, N., "A Review of Hardware Acceleration for Computational Genomics," in Design & Test, IEEE , vol.31, no.1, pp.19-30, Feb. 2014.

[14] Koonin EV, Galperin MY. Sequence - Evolution - Function: Computational Approaches in Comparative Genomics. Boston: Kluwer Academic; 2003.

[15] H. Hyyrö, M. Juhola, M. Vihinen, "On exact string matching of unique oligonucleotides". Comput Biol Med, vol. 35, no. 2, pp. 173-81, 2005.

[16] M. Brudno and B. Morgenstern, Fast and sensitive multiple alignment of large genomic sequences. BMC Bioinformatics, vol 4, 2003, pp 66.

[17] Michael M, Dieterich C, Vingron M: SITEBLAST--rapid and sensitive local alignment of genomic sequences employing motif anchors. Bioinformatics 2005, 21(9):2093–2094.

[18] I. Garcia-Vargas et al., "Rom-based finite state machine implementation in low cost fpgas," in Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on, June 2007, pp. 2342-2347.

[19] R. Senhadji-Navarro, I. Garcia-Vargas and J. L. Guisado , "Performance evaluation of RAM-based implementation of finite state machines in FPGAs," Proc. 19th IEEE Int. Conf. Electron. Circuits Syst. (ICECS) , pp.225 -228.

[20] P.R. Schaumont, "A Practical Introduction to Hardware/Software Codesign", Springer, 1st Edition, 2010.

[21] J. Teich , "Hardware/software codesign: The past, the present, and predicting the future" , Proc. IEEE, vol. 100 , pp.1411 -1430 , 2012.

[22] M. Santarini, "Zynq-7000 EPP Sets Stage for New Era of Innovations," Xcell journal, issue 75, second quarter, 2011.

[23] M. Polanski and NL Anderson, "A List of Candidate Cancer Biomarkers for Targeted Proteomics", Biomarker Insights, pp. 1-48, 2006.

[24] "UniProt: a hub for protein information", Nucleic Acids Res., vol. 43, pp. D204-D212, 2015.