A high speed implementation counter mode cryptography using hardware parallelism

Samad Najjar-Ghabel
Department of Electrical and Computer Engineering
University of Tabriz,
Tabriz, Iran
samad.najjar92@ms.tabrizu.ac.ir,
samad.najjar@gmail.com

Shamim Yousefi
Department of Electrical and Computer Engineering
University of Tabriz,
Tabriz, Iran
sh.yousefi92@ms.tabrizu.ac.ir,
shamimyousefi75@gmail.com

Mina Zolfy Lighvan
Department of Electrical and Computer Engineering
University of Tabriz,
Tabriz, Iran
mzolfy@tabrizu.ac.ir

***Abstract*—Nowadays, cryptography is one of the common security mechanisms. Cryptography algorithms are used to make secure data transmission over unsecured networks. Vital applications are required to techniques that encrypt/decrypt big data at the appropriate time, because the data should be encrypted/decrypted are variable size and usually the size of them is large. In this paper, for the mentioned requirements, the counter mode cryptography (CTR) algorithm with Data Encryption Standard (DES) core is paralleled by using Graphics Processing Unit (GPU). A secondary part of our work, this parallel CTR algorithm is applied on special network on chip (NoC) architecture that designed by Heracles toolkit. The results of numerical comparison show that GPU-based implementation can be achieved better runtime in comparison to the CPU-based one. Furthermore, our final implementations show that parallel CTR mode cryptography is achieved better runtime by using special NoC that applied on FPGA board in comparison to GPU-based and CPU ones.**

Keywords*— Counter Mode Cryptography (CTR); Data Encryption Standard (DES); FPGA; Grafic Process Unite(GPU); Network on Chip(NoC); Parallel Computing.

I. INTRODUCTION

Nowadays, Internet is used for communication, business, banking, mailing and many other tasks. Internet network is unsecured. So, it is vital to propose the secure mechanisms for using internet's services. Cryptography is one of the security mechanisms that used to make security in data transmission. Cryptography algorithms are used for encryption/decryption of data to archive the security goals (confidentiality, integrity, availability) [1]. The original data is called the "plaintext" and the data that be made by applied encryption algorithm and security key on plaintext is called "ciphertext". The technology that creates cipher text from plain text known as "encryption" and the reverse operation of encryption is called "decryption". It is worth mentioning that the cryptography algorithms are classified into symmetric-key encipherment, asymmetric-key encipherment and hashing [1].

The traditional data encryption algorithms such as Data Encryption Standard (DES) [2], International Data Encryption Algorithm (IDEA) [3] and RSA [4] are only used for encryption/decryption data with a low number of characters. With the development of multi-core systems and parallel architectures, the multiple threads on the central processing unit (CPU) can be created and performed by assigning different tasks independently on multi cores that set on the CPU technologies. However, the evaluation of parallel implementation's performance on data encryption algorithms shows that parallelism using GPU [5] significantly improve the computational performance of data encryption/decryption [6, 7, 8].

Although, software implementations of the cryptography algorithms can easily convert between each others, all of them suffer from few data rates. The fastest implementation of the cryptography algorithms that achieves a high throughput is on the Field Programmable Gate Array (FPGA) [9, 10]. In such implementations, the heart of the encryption/deception algorithm and specific parameters of it are optimized without major restrictions in the system generality.

In real world, confidential data are variable size and usually their size is larger than 64 bits. Modes of operation makes the cryptography algorithms appropriate for encryption/decryption of the data with a large number of characters. Modes of operation involve five kinds: Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB) and Counter (CTR) [1, 11]. However, the traditional implementations have not used the cryptography modes. So, they are not suitable for encryption/decryption data with a high number of characters.

CTR mode cryptography algorithm work only with key security and counter parameters. So, the algorithm does not need to receive all bits of plaintext or cipher text. As a result, CTR mode has highest encryption/decryption speed in comparison with other ones [11]. CTR mode can be work parallel in both of data encryption/decryption. Furthermore, this mode has an excellent performance in a noise channel because it has not any feedback [1].

Our work focuses on symmetric-key cryptography that can be used modern block cipher. Each algorithm is designed to encryption/decryption of data in fixed size. In this paper Data Encryption Standard (DES) Algorithm [1, 11, 12] is used that one of the symmetric-key cryptography. This algorithm encrypts/decrypts data with the size of 64 bits. General structure of DES shown in fig. 1. As shown in fig. 1, DES algorithm has two permutation before and after of the process main core. Furthermore, the figure shows that the algorithm has 16 round that use its security key.

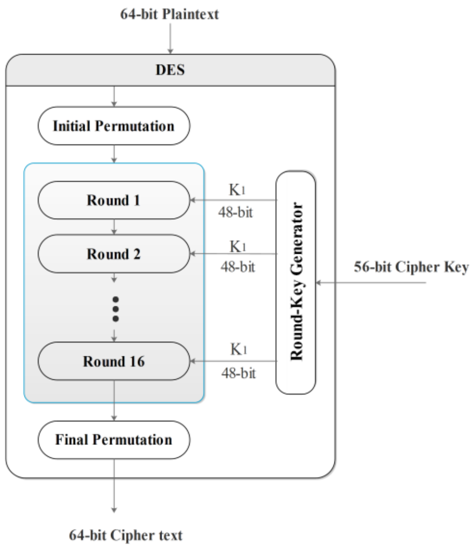

Figure 1. overall structure of DES [1].

In novel cryptography applications, speed is important. In order to increase speed of cryptography modes, hardware and software approaches can be used. In this paper, we apply the Parallel CTR mode with the DES cryptography algorithm on graphic process unit (GPU) [13] to increase speed of the cryptography process. Then, we compare GPU implementation with the CPU one. Our result of the implementations show that the GPU is one of the suitable technologies that significantly improves the computational performance of data encryption/decryption.

Furthermore, main idea of our work is proposed specific hardware over a network on chip (NoC) to increase the speed of data encryption/decryption. Specific hardware that used in this paper calls Heracles [14, 15]. First, the parallel CTR mode with DES algorithm core proposed. Then this algorithm applied on GPU to use the parallelism power of it. Then, the cryptography algorithm is implemented on NoC-based architecture for decrease runtime. Our implementation results show that parallel CTR mode cryptography with the DES core is achieved best performance by using special NoC that applied on FPGA board (Virtex-6 LX550T).

The rest of the paper is organized as follows. In section two, a brief overview of related works is given. In section three, the parallel CTR mode cryptography with the DES core GPU-based and NoC-based implementation are explained. Section 4, includes the simulation setup and implementation of the parallel CTR mode cryptography with the DES core by FPGA and GPU. Then, subsequent results regarding the performance of the method are provided in comparison with the serial one. Finally, the conclusion presented in the last section.

II. RELATED WORK

Despite of several decades' improvement in data encryption algorithms and all the developments made in hardware and technologies, a reasonable performance in data encryption algorithms is not achieved yet. The traditional data encryption algorithms such as Data Encryption Standard (DES) [2], International Data Encryption Algorithm (IDEA) [3] and RSA [4] are only used for encryption/decryption data with a low number of characters.

With the development of multi-core systems and parallel architectures, more computing power has been available. The multiple threads on the central processing unit (CPU) can be created and performed by assigning different tasks independently on multi cores that set on the CPU technologies. Some works used the methods such as OpenMP [16], Pthreads [17] and UPC [18] to apply the ability of multi cores.

However, the evaluation of parallel implementation's performance on data encryption algorithms show that parallelism using GPU [5] significantly improve the computational performance of data encryption/decryption [6, 7]. Szerwinski et al. [19] proposed novel implementations of cryptography techniques based on elliptic curve operations on graphics hardware by using CUDA framework and GPGPU processing of cryptosystems. The simulation results of the authors showed that the approach improves only half of throughput features but provides a faster data response. Therefore, their proposed method seems to be suitable in interactive applications. Venugopal et al. [20] implement target different encryption algorithms in GPU and realize a high level of security and performance, with a reasonable cost. The results of the paper indicated that GPU implementation provides better throughput for all encryption algorithms as compared to serial one. Also, the results show that the GPU is better suited as embedded cryptography co-processors as compared to CPU, in terms of cost and development time. Some other works improve the performance of different cryptography algorithms by using GPU implementations [21, 22, 6, 23, 24,8].

Although, software implementations of the cryptography algorithms can easily convert between each others, all of them suffer from few data rates. The fastest implementation of the cryptography algorithms that achieves a high throughput is on the Field Programmable Gate Array (FPGA) [9, 10]. McLoone et al. [9] presented a parameter is able and reusable key scheduling core that can be utilized in pipelined private-key encryption algorithms. The DES algorithm that lends itself to pipelining, is used to exemplify presented key scheduling method. The results of the authors' simulation showed that the FPGA implementation is 28 times faster than the software one. Some other works proposed efficient FPGA designs and architectures for different cryptography algorithms [25, 26, 27]. In such implementations, the heart of the encryption/deception algorithm and specific parameters of it are optimized without major restrictions in the system generality. However, the traditional implementations have not used the encryption modes such as Counter-mode encryption (CTR), Ciphertext feedback (CFB) or the Cipher Block Chaining (CBC). So, they are not suitable for encryption/decryption data with a high number of characters.

In this paper, we focus on parallel CTR mode with DES core cryptography. First, the parallel CTR cryptography algorithm is applied on GPU that can be processed big data in acceptable runtime. Then, the parallel CTR algorithm is implemented on special hardware architecture that designed over FPGA using Heracles toolkit [15].

III. Proposed Method

Nowadays, cryptography techniques are used in network-based application to provide a secure communication. In some application, data encryption/decryption runtime is an important parameter. Two approaches can be applied for enhancing data encryption/decryption run time: 1) Using more powerful processors 2) Parallel computing.

In this paper, serial, GPU-based and NoC-based approaches are applied. In the first subsection, we explain every aspect of proposed algorithm and pseudo code. Then, in order to increase speed of data encryption/decryption, we use GPU in the next subsection. Finally, last subsection contains main idea of the paper. In last subsection, we propose a new hardware approach to optimize the encryption/decryption runtime.

A. *Parallel CRT mode cryptography*

Parallel Counter mode with DES cryptography algorithm is the main discussion in our proposed methods. Overall structure counter mode that used in the paper is shown in fig. 2. This mode provides more randomness by the increment value can depend on the block number. It is worth mentioning that the counter mode of cryptography can be work with any kind of symmetric-key cryptography algorithm. Cryptography algorithm only initialization vector (IV) encrypts with security key. Moreover, this encryption part does not need to wait to receive all plaintext. In this work, each block can encrypt or decrypt 64 bits by using DES algorithm. Fig. 2. Shows ~~that~~ counter, security key (Key) and plaintext (p) as input of parallel CTR algorithm and cipher text (C) as output. Each block uses DES encryption algorithm that encrypted counter by security key. The output of DES algorithm is the result of 64 bits plaintext that XORed together.

Algorithm 1 is shown an algorithm that applied in this paper. In the serial format *for* instruction run in the serial form but in the parallel run on GPU or NoC, this instruction can be run parallel. This algorithm use CTR mode in order to decrypt data more than 64 bit. Furthermore, ran parallel and use DES encryption in order to increase runtime. In order to decryption cipher text and achieve plaintext only exchange plaintext (P) with cipher text (C).

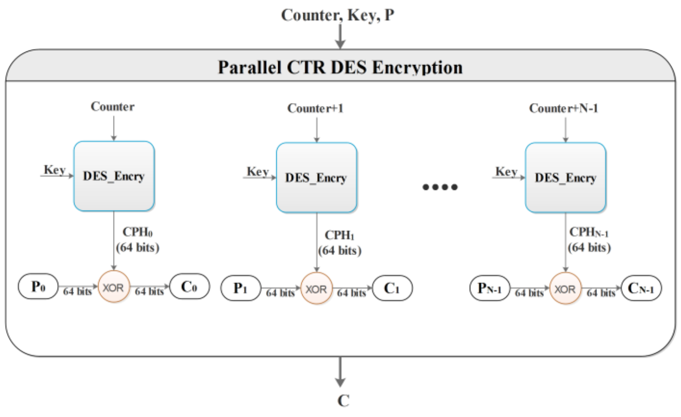

Figure 2. Block Diagram of Parallel CTR DES [1].

Algorithm 1: Parallel_CTR_DES Encryption

```
Input: Plaintext, Security Key and Initialization Vector (IV)
Output: Cipher Text Blocks
begin
   N ← calculate number of block from length of plaintext;
   M ← calculate length of last block;
   Counter ← IV;
   Parallel.for (i=0 to N-1)
   {
       $CPH_i$ ← DES_Encry((Counter+i)mod $2^N$, Key);
       if (i<>N-1)
           $C_i$ ← SelectBit(Plaintext, i×64, 64)⊕$CPH_i$;
       else
           $C_i$ ← SelectBit(Plaintext, i×64, M)⊕$CPH_i$;
   }
   Return C;
end
```

B. *GPU-based implementation of CRT mode cryptography*

To reduce encryption/decryption runtime, the Parallel algorithm applied on GPU in this section. In many applications, parallel form of the algorithms runs on the multiprocessor instead of single-thread. So, in this paper, Parallel CTR mode is implemented on the GPU for using the multiprocessing power. Graphic processor of NVidia via VC++ based CUDA Library [13] is used in current implementation. The properties of the system in which algorithm has been implemented on it are:

- Processor: intel corei7-2670Qm
- Memory: 4GB
- Graphic card: NVidia GeForce GT 540m with 96 CUDA Core
- Operation System: Windows 7

The result of implementation shows that GPU cores can reduce run time and increase the performance of cryptography process.

C. *NoC-Based implementation of CRT mode cryptography*

Although, software implementations of the cryptography algorithms can easily convert between each others, all of them suffer from few data rates. So, in many security applications, Hardware approaches are more acceptable. In this subsection, the parallel cryptography processing that be mentioned in previous subsection, is implemented on the special network on chip (NoC). In all NoC implementations and parallel processes, the FPGA processing power is used. In this work, a NoC architecture using the mentioned Heracles toolkit is designed. Then the Parallel CTR mode cryptography is compiled into binary machine code by its compiler tool chain. Finally, the binary codes is loaded on related cores. The overall structure of the proposed architecture is shown in fig. 3

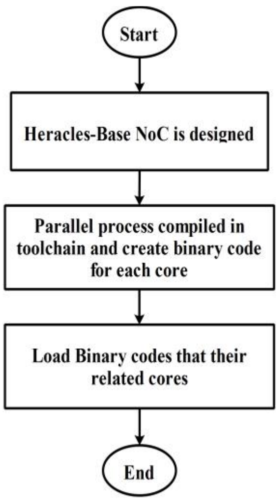
Start
Heracles-Base NoC is designed
Parallel process compiled in toolchain and create binary code for each core
Load Binary codes that their related cores
End

Figure 3. the overal structure of the proposed architecture.

To implement the Parallel CTR mode of cryptography algorithm using NoC, the designed architecture properties include four cores in 2D mesh network. Each core is integer 7-stage 32-bit MIPS and has 256KB local memory [14]. Furthermore, 1KB data/instruction cache memory is given for each core. Dimension-Order Routing (DOR_XY) is applied as routing algorithm. Finally, Xilinx ISE Design Suite 11.5, with Virtex-6 LX550T package ff1760 speed -2, as the targeted FPGA board with 118.86MHZ maximum frequency, synthesizes all properties [14]. In CTR mode each block run independently. Therefore, N/4 block are assigned to a core, when the number of block is N. configurable NoC architecture proposed in [14] calls Heracles [15] is used for our hardware. The result of implementation shows that NoC-based implementation work better than GPU-based. So, our proposed architecture archives best performance.

IV. EXPERIMENTAL RESULT AND DISCUSSION

In this section, the runtime of parallel CTR algorithm by GPU-based and NoC-based implementation is compared in serial. In the serial implementation, encryption algorithm is tested by C++ programming language that run on the CPU. In Cuda C++ implementation, sequential partitions are performed on the CPU and parallel pieces executed on the GPU. It is worth mentioning that the experimental results of serial and Cuda C++ implementations have been conducted in a computer with the properties that mentioned in section 3.

To get more accurate results, the simulations perform 100 times on the plaintexts by the serial and Cuda C++ implementations and average of the numerical results are presented. In the NoC method, the upper bound of clock cycle is considered. The upper bound of the clock cycle is reported by calculating the runtime with applying work frequency of the system. As shown in table I, different data with 2, 4, 16, 32, 64, 128, 256, 512 and 1024 block (each block has 64 bit) were encrypted by using the CPU, the combination of CPU and GPU and special NoC. The results of the table show that the runtime of the CTR algorithm is increased with raising the number of data.

Table I. Numerical result of implementation.

Structure	**Number of block (each block has 64 bit)**	**Average Time(s)**	**Run time (s)**	**Maximum Clock cycle**
Serial execute with C++ (Run on CPU)	2	1.5×10^{-3}	--	--
	4	3.1×10^{-3}	--	--
	8	4.6×10^{-3}	--	--
	16	8.6×10^{-3}	--	--
	32	2.02×10^{-2}	--	--
	64	5.77×10^{-2}	--	--
	128	1.887×10^{-1}	--	--
	256	6.924×10^{-1}	--	--
	512	2.6629	--	--
	1024	10.619	--	--
Parallel Execution Using NoC (4 core) 118.86MHZ	2	--	6.7306×10^{-6}	800clc
	4	--	6.7306×10^{-6}	800clc
	8	--	6.7306×10^{-6}	800clc
	16	--	1.262×10^{-5}	1200clc
	32	--	1.261×10^{-5}	1500clc
	64	--	1.682×10^{-5}	2000clc
	128	--	4.206×10^{-5}	5000clc
	256	--	6.73×10^{-5}	8000clc
	512	--	8.413×10^{-5}	10000clc
	1024	--	1.177×10^{-4}	14000clc
Parallel Execution Using GPU CUDA	2	--	$\sim0.2\times10^{-3}$	--
	4	--	$\sim0.2\times10^{-3}$	--
	8	--	$\sim0.2\times10^{-3}$	--
	16	--	$\sim0.2\times10^{-3}$	--
	32	--	$\sim0.2\times10^{-3}$	--
	64	--	$\sim0.2\times10^{-3}$	--
	128	--	0.247×10^{-3}	--
	256	--	0.839×10^{-3}	--
	512	--	0.3226×10^{-2}	--
	1024	--	0.42488×10^{-1}	--

Furthermore, Compression Serial execute CTR cryptography algorithm with parallel CTR cryptography algorithm implementations by using Cuda C++ are presented in fig. 4. The simulation results show that the runtimes of parallel CTR cryptography algorithm by using the Cuda C++

implementation are almost 83% better than using the serial ones. Also, the results illustrate that the algorithm's runtime in the parallel implementation becomes better when the data size is higher than a threshold. Finally, total results of comparison the CPU and GPU implementations show GPU-based implementation work better than serial one.

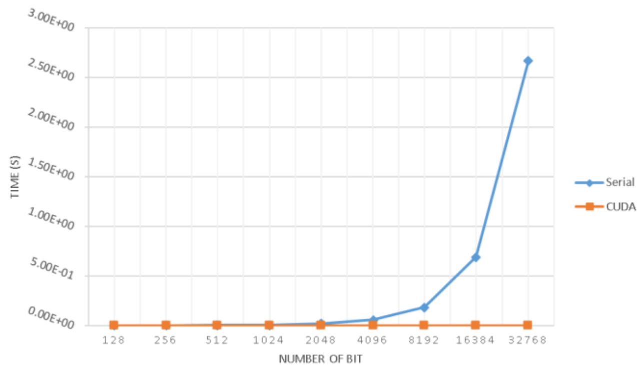

Figure 4. Simulation Result of Serial and Cuda C++ implementation

To get the final results, the average runtime of the parallel CTR cryptography algorithm by using the Cuda C++ and NoC implementations are analyzed in fig. 5. The simulation results show that the runtime of the NoC implementation is 90.91% better than Cuda C++ one.

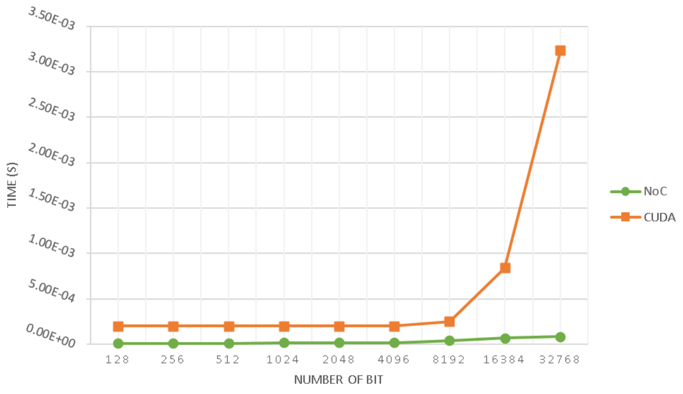

Figure 5. Simulation Result of NoC and Cuda C++ implementation.

The overall comparison of the figures is shown in fig. 6. The result of comparing figures shows that the average runtime of the NoC implementation is 99.98% better than serial one. Furthermore, the results illustrate that the execution time of the parallel CTR cryptography algorithm by using NoC becomes better when the data size is higher than a threshold.

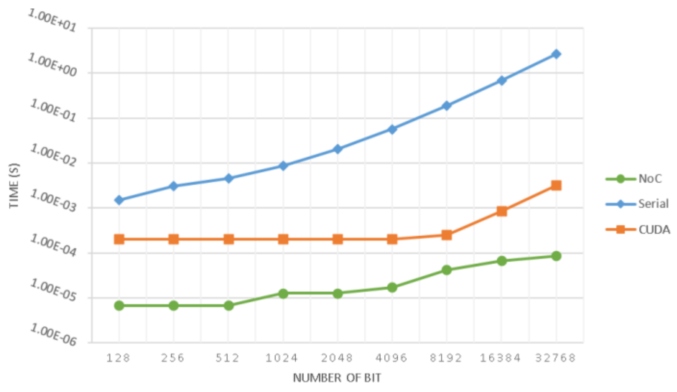

Figure 6. Simulation Result of NoC, Cuda C++ and Serial Implementation

V. CONCLUSION

Vital data usually is transmitted over unsecure channel. So, the security of the transmitted information is one of the important challenges in today's networks. Cryptography algorithms are used for data encryption/decryption to archive the security goals. Vital applications are required to techniques that encrypt/decrypt big data at the appropriate time, because the data should be encrypted/decrypted are variable size and usually the size of them is large. In this paper, we proposed two special parallelism approaches to achieve a reasonable runtime. First, we implanted the parallel CTR algorithm on GPU. Then, the parallel CTR algorithm is applied on a special NoC. The results of implementations show that the runtime of the parallel CTR algorithm by using the NoC implementation is 95.91% and 99.98% better than Cuda C++ and serial ones, respectively.

References

[1] B. A. Forouzan, Cryptography & Network Security, Indian: McGraw-Hill, 2007.

[2] V. V. Thampi and R. K. Gopal., "A review on different encryption algorithms for a wellness tracking system," in *Communication Technologies (GCCT), 2015 Global Conference on IEEE*, 2015.

[3] A. Hora and N. Sahayam, "Implementation of Efficient Probabilistic Symmetric Key Encryption Method," *Asian Journal of Current Engineering and Maths*, vol. 4, no. 3, pp. 33-36, 2015.

[4] S. Khatarkar and R. Kamble, "A Survey and Performance Analysis of Various RSA based Encryption Techniques," *International Journal of Computer Applications*, vol. 114, no. 7, pp. 30-34, 2015.

[5] W.-K. Lee , H.-S. Cheong and R. C.-W. Phan, "Fast implementation of block ciphers and PRNGs in Maxwell GPU architecture," *Cluster Computing*, vol. 19, no. 1, pp. 335-347, 2016.

[6] J. V. Tembhurne and S. R. Sathe, "RSA Public Key Acceleration on CUDA GPU," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*, India, Springer, 2016, pp. 365-375.

[7] E. Niewiadomska-Szynkiewicz, M. Marks, J. Jantura, M. Podbielski and P. Strzelczyk, "Comparative study of massively parallel cryptalysis and cryptography on CPU-GPU cluster," in *In Military Communications and Information Systems Conference (MCC)*, St.-Malo, 2013.

[8] S. Najjar Ghabel, A. Mohammad Baklou, M. Zolfy Lighvan and L. Mohammad Khanli, "Search Acceleration in Preprocessing Mechanism of Network Intrusion Detection Systems Using Graphics Processors," in *Information and Knowledge Technology (IKT), 2015 7th Conference on IEEE*, Urmia, Iran, 2015.

[9] M. McLoone and J. V. McCanny, "High-performance FPGA implementation of DES using a novel method for implementing the key

schedule," *IEE Proceedings - Circuits, Devices and Systems,* vol. 150, no. 5, pp. 373-378, 2003.

[10] H. Michail, G. Athanasiou, G. Theodoridis and C. Goutis, "On the development of high-throughput and area-efficient," *Integration, the VLSI Journal,* vol. 47, no. 4, pp. 387-407, 2014.

[11] W. Stallings, Cryptography and Network Security Principles and Practices, F. Edition, Ed., United States of America: Prentice Hall, 2005.

[12] C. Paar and J. Pelzl, Understanding Cryptography, London New York: Springer, 2010.

[13] "CUDA Parallel Computing Platform," NVIDIA, [Online]. Available: http://www.nvidia.com/object/cuda_home_new.html. [Accessed 2 2 2016].

[14] M. A. Kinsy, S. Devadas and M. Pellauer, "Heracles: a tool for fast RTL-based design space exploration of multicore processors," in *the ACM/SIGDA international symposium on Field programmable gate arrays*, California, USA, 2013.

[15] Computation Structures Group, "Heracles Project," Computer Science and Artificial Intelligence Laboratory, [Online]. Available: http://projects.csail.mit.edu/heracles/. [Accessed 21 5 2015].

[16] M. Joppich , D. Schmidl, A. M. Bolger, T. Kuhlen and B. Usadel, "PAGANtec: OpenMP parallel error correction for next-generation sequencing data," *Heterogenous Execution and Data Movements,* vol. 9342, no. 1, pp. 3-17, 2015.

[17] Y. Fei, H. Zhu, X. Wu and H. Fang, "Comparative Modeling and Verification of Pthreads and Dthreads," in *2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE)*, Orlando, FL, 2016.

[18] A. Bakker and M. L. Allen, "Methods and system for using a signal universal product code (UPC) to start discount injection," *United States Patent Application,* vol. 20, no. 4, pp. 698-740, 2016.

[19] R. Szerwinski and T. Güneysu, ""Exploiting the power of GPUs for asymmetric cryptograph," *Cryptographic Hardware and Embedded Systems*, vol. 5154 , no. 1, pp. 79-99, 2008.

[20] V. Venugopal and D. Manikantan Shila, "High Throughput Implementations of Cryptography," in *IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, Minneapolis, MN, 2013.

[21] T. Mahajan and S. Masih, "Enhancing Blowfish File Encryption Algorithm," in *Computer, Communication and Control (IC4)*, Indore, 2015.

[22] D. Nilesh and M. Nagle, "The New Cryptography Algorithm with High," in *Computer Communication and Informatics (ICCCI), 2014 International Conference on*, Coimbatore, 2014.

[23] M. Yoon, A. Cho, M. Jang and J.-W. Chang, "A data encryption scheme and GPU-based query processing algorithm for spatial data outsourcing," in *International Conference on Big Data and Smart Computing (BIGCOMP)*, Jeju, 2015.

[24] L. Habibpour, S. Yousefi, M. Zolfy Lighvan and H. S. Aghdasi, "1D Chaos-based Image Encryption Acceleration by using GPU," *Indian Journal of Science and Technology,* vol. 9, no. 6, pp. 19-25, 2016.

[25] H. I. Shahadi, R. Jidin and W. Hun, "Concurrent hardware architecture for dual-mode audio steganography processor-based FPGA."," *Computers & Electrical Engineering*, vol. 49, no. 1, pp. 95-116, 2016.

[26] D.-S. Kundi, A. Aziz and N. Ikram, "A high performance ST-Box based unified AES encryption/decryption architecture on FPGA," *Microprocessors and Microsystems*, vol. 41, no. 1, pp. 37-46, 2015.

[27] K. Järvinen, "Optimized FPGA-based elliptic curve cryptography processor for high-speed applications," *INTEGRATION, the VLSI journal*, vol. 44, no. 4, pp. 270-279, 2011.