VOL. 18, NO. 15, AUGUST 2023 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

AN EFFICIENT IMPLEMENTATION OF A HYBRID SIGNCRYPTION PROCESSOR USING FLEXIBLE ENCRYPTION AND SIGNATURE TECHNIQUES

Senthil Murugan M.[1], Bhuvana B. P.[1] and Jayabharathi P[2].
[1]Department of Eelectronics and Communication Engineering, St. Joseph's Institute of Technology, Chennai, India
[2]Department of Computer Science Engineering, St. Joseph's College of Engineering, Chennai, India
E-Mail: senthilmuruganap@gmail.com

ABSTRACT

In today's digital era, with the enormous growth in the Internet of Things, everyday humans utilize IoT in various applications such as banking, online transactions, purchasing data forwarding, etc. Cryptography is all about the design and implementation of protocols that keep the data safe and secure from third parties. The design of the cryptographic processor present in the *Field Programmable Gate Array* (FPGA) incurs more power, area, and throughput. In this article, an *Efficient Hybrid Signcryption Processor* (EHSP) which operates based on the principle of encryption and signature is proposed. Generally, a hybrid signcryption operation comprises of *Data Encapsulation Method* (DEM) and *Key Encapsulation Method* (KEM). and in EHSP, we utilize the improved form of the *Modified Kurosawa and Desmedt* (MKD) hash scheme to encapsulate the key; the *Elliptic Curve Cryptography* (ECC) processor is used to encapsulate the message. The ECC processor is designed in a flexible manner and the flexibility is achieved by the flexible bit-serial multiplier, which reduces the power, and area consumption and maximizes the throughput. The proposed EHSP architecture is designed and simulated in Xilinx tool with various field programmable gate array families such as Virtex4 (XC4VLX60), Virtex5 (XC5VLX50), and Virtex7 (XC7V330T). The simulation result depicts the efficiency of the proposed EHSP design with respect to total power consumption, utilization of hardware, and throughput.

Keywords: hybrid signcryption, flexible encryption, hash function, ECC processor, Xilinx tool, FPGA.

Manuscript Received 28 March 2023; Revised 30 September 2023; Published 10 October 2023

1. INTRODUCTION

Privacy, integrity, non-repudiation, and authentication are major challenges for securing information. In the open key encryption methodology, a message is stamped and thus taken after by an encryption using the "signature-then encryption" method [1]. In signature-then-encryption methodology, there are two drawbacks such as high cost and low efficiency. These drawbacks were overcome by our proposed novel algorithm called Signcryption [2]. Signcryption is a novel technique that achieves the functionalities of both digital signatures as well as encryption methodology. Signcryption is a new approach in public key cryptology. It provides a common structure for various protocols to transmit a message in a confidential and authenticated manner. One of the notable characteristics of Signcryption is the capability of providing both encryption and digital signature at the same time. In other words, by using signcryption we can achieve confidentiality, authentication, integrity, unforgeability, non-repudiation, public verifiability, and forward secrecy of message confidentiality [3] [4]. Signcryption attains very little communication overhead and incurs a small amount of computational cost. In signcryption methodology, before transmitting a message, the sender has to follow a set of protocols as follows [5]:

a) Authenticate it using a digital signature.

b) The data and the signature are encrypted by utilizing a private key encryption algorithm by randomly generated encryption key.

c) With the help of the receiver's public key, the encryption process is carried out by an arbitrary message encryption key.

The major disadvantage of this approach is when a message is digitally signed and encrypted, it incurs additional machine cycles and expands the message bits by introducing redundant bits to it. From this time onward, the process of decryption and verification of message bits exploits a lot of computational power at the receiver's end [6]. Hybrid signcryption is a process that combines the advantages of both symmetric and asymmetric techniques and it is very efficient when compared with all other exiting crypto schemes. In general, by utilizing hybrid signcryption methodology asymmetric encryption schemes were generated, in which encryption is provided by a symmetric encryption scheme. The asymmetric encryption system is used to encrypt randomly generated symmetric keys, which allows the asymmetric encryption scheme to handle long messages. Hence, hybrid signcryption finds its place in applications where large messages will be handled. Hybrid signcryption can be designed using various cryptosystems such as *Key Encapsulation Mechanism* (KEM) and *Data Encapsulation Mechanism* (DEM) [7]. A public-key cryptosystem is utilized in the

design of KEM and the symmetric-key cryptosystem is used to design DEM.

Usually, hybrid signcryption is easy to implement in EDA software, however, it is very slow for real-time applications such as network routers, embedded systems, storage devices, etc. Because of these reasons, it is very important to implement it on the hardware setup [8]. The cryptographic hardware systems should satisfy the self-contradictory requirements such as the implementation of computationally widespread cryptographic functions with high-speed parallel structures. These should coincide with complex sequential structures that help in implementing cryptographic algorithms such as cryptographic protocols, cipher modes, and key management operations [9].

The design of cryptographic protocols and algorithms in the hardware units requires complicated structures which makes the design more vulnerable to side-channel attacks. In recent times, security algorithms were executed in FPGAs consisting of various memory devices such as flash-based devices and SRAMs [10]. Hence, FPGAs have to be reprogrammed and hence hardware updates become cheaper and easier [11].

The design of FPGA with SHA-512 [12] is very tedious and incurs a very high cost. At this point, an error detection system can solve the issues related to either data handling or redundancy errors in hardware. The modified Itoh-Tsujii inverse algorithm (ITA) [13] executed on FPGA needs smaller addition chains. FPGA-based SHA-1 cryptanalysis system [14] is utilized to attain a higher EOC when compared with all other exiting software and hardware solutions.

The keyed-*Hash Message Authentication Code* (HMAC) is a methodology in which [15] fault tolerance is used in the *Secure Hash Standard* (SHS) and also it consumes more area and power. The design of the RC4 algorithm [16] based on the concept of dual-port block RAM in FPGA achieves better utilization of logic and memory resources, which in turn results in improved performance. Implementation of the 128-bit advanced encryption standard (AES) algorithm [17] is based on the conceptualization of C-slow retiming, which offers feedback loops, and this results in rebalancing the registers in the design. Area-efficient, high-throughput multi-mode architectures for the SHA-1 and SHA-2 hash families are designed by a systematic flow for designing multi-mode architectures [18].

Elliptic curve cryptography plays a significant role in the security domain. It provides assured security with more minor key sizes, speedier cryptographic exercises, and running on more straightforward chips. Possible, unnecessary gear executions are open for the ECC process with the humbler key length. Memory anticipated ECC [19] with different evaluated multiplier units are applicable for different point augmentation and duplicating checks over major field apparent FPGAs. The execution time is scaled with the measure of isolated multipliers and demonstrates without any overhead when we relate it with other multipliers. ECC is incomparable and the working of the ECC scalar duplication graph [20] relies upon the Lopez-Dahab estimation over GF (2^{163}). All the operations in the processor are performed by ALU which incorporates addition, squaring, and a multiplication unit.

In this paper, an *Efficient Hybrid Signcryption Processor* (EHSP) is proposed in the FPGA hardware platform, which satisfies the basic security guarantees at the hardware level. The operation of EHSP consists of *Key Encapsulation Method* and *Data Encapsulation Method*. Here, the *Modified Kurosawa and Desmedt* hash is used to perform the Key Encapsulation Method, and the Elliptic Curve Cryptography processor is utilized to perform the Data Encapsulation Method. The main aim of the proposed EHSP hardware architecture is to obtain high reliability and unforgeability with minimal power consumption, effective hardware utilization, and throughput. The rest of this paper is systematized as follows. Recent literature is discussed in Section II. The disadvantages of the previous methodologies and the advantages of the proposed HAS system are described in Section III. The proposed HAS scheme is elaborated in Section IV. The performance of the proposed methodology is analyzed in Section V. The paper is concluded in Section VI.

2. RELATED WORKS

Javeed *et al*. [21] have demonstrated a highly efficient hardware design for ECC which operates for any prime number $p \leq 256$ bits. The design is opted for integrating into various ECC-based cryptosystems for increasing the operation speed of elliptic curve scalar multiplication. This is the major aspect of any such design. The structural design of the hardware accelerator consists of an arithmetic unit. This Arithmetic unit itself comprises four multiplier units and an adder or subtractor unit. The modular multiplier is an important component in the design of elliptic curve scalar multiplication mainly over projective coordinates, and hence radix-4 Booth encoding interleaved modular multiplier is employed in this design. This design has been synthesized and executed on Xilinx Virtex-4 and Virtex-6 FPGA platforms. This utilizes frequencies of 40MHz and 70MHz to execute a single 256-bit elliptic curve scalar multiplication operation, respectively.

Liu *et al*. [22] have described a processor that consumes less area for resource-constrained applications. In this paper, the author employed a 16 bit-processor for processing information and it has a general silicon zone of 5821 GE once joined along with a CMOS 130nm standard-cell library. The core was designed using parallel processing methodology utilizing two ALUs and RAMs and it results in compact and fast FPGA implementation in the double scalar multiplication. This methodology accomplishes good throughputs for verification of signature and it verifies the server-side activities with the help of IoT by one FPGA device.

Liu *et al*. [23] have shown a twofold field ECC processor, which aids field length up to 576 bits. The programmable ECC processor controls some essential measures to play out the timetables of various ECC multiplications (ECSM) checks. A detailed *Modular*

Arithmetic Logic Unit (MALU) is utilized to enhance the effectiveness of hardware, which helps the design to operate at high speed.

Mitra *et al*. [24] have indicated high throughput low latency CRC-32 with reconfigurable parameters utilizing expandable data bus lines. The low latency is achieved by parallelizing the execution of various logics. The way this problem is dealt with is clearly described with the help of block diagrams and mathematical expressions. The exceptionality in this design is the capability of the design to operate on the same clock cycle in which the code word is generated and the results are generated in the next clock cycle. This CRC design is much more desirable concerning very high throughput, single clock latency, and reconfigurability.

Chellam *et al*. [25] have proposed high-throughput and resource-enhanced execution of 128-bit AES processors. The partitioning of memory is done so that several ports are assigned for accessing the information parallelly. All activities in advanced encryption Standard can be started with a single clock cycle and it can receive input at each clock cycle. The design is synthesized and executed in the upgraded XC7VX690T device which gives a throughput of 104.06 Gbps at a frequency of 813 MHz and 1.23-ns path delay.

Sugier [26] proposed the BLAKE figuring utilizing hash limits, which made as a likelihood for the SHA-3 challenge where it efficaciously skilled the last round even though it lost to KECCAK. Particular changes of hardware executions that utilize RAM modules in contemporary FPGA. The alteration is utilized to remove the distribution of message bits among the ciper rounds to diminish the use of array resources and enhances the performance metrics. This is verified on four various structures (i.e.) one standard iterative and three high-speed loop-unrolled organizations with 2, 4, and 5 rounds instantiated in hardware.

At *et al.* [27] have proposed the approach of an 8-bit coprocessor for the AES with the encryption, decryption, and key change of cryptographic hash functionality Grøstl. This ALU has a single instruction that permits for the implementation of AES encryption, AES key extension, AES decryption, and Grøstl at all levels of security. A key plan joins the Add Round Constant advances and the bitwise XOR activities of the weight work. An automatic utilization of Grøstl and AES on a Virtex-6 FPGA requires 169 slices and a particular 36k memory block and accomplishes a high throughput.

Asif *et al.* [28] proposed the framework of an ECC processor which is based on a concept called *Residue Number System* (RNS). The multiplication in Jacobian coordinates is executed by the processor by using point addition and point doubling. Architecture can be reduced in terms of time and area for better performance. Optimized modular reduction architecture is a methodology that consumes very little area by partitioning the RNS module into tiny groups and these groups were processed individually.

A combined architecture of the ECPD and ECPA is used to reduce the required number of modular operations on the critical path and improve the number of clock cycles for one ECPM by approximately 17%. The design was synthesized, and implemented on the Virtex-7 and Virtex-6 FPGAs and shows performance to the state-of-the-art binary and RNS based ECC processors. The combined arrangement of ECPD and ECPA helps in reducing the number of assignments needed for the focal way and it upgrades the number of clock cycles needed by approximately 17%. This system is simulated and implemented on Virtex-7 and Virtex-6 FPGAs and this implementation is efficient in RNS based ECC processors. Shahbazi *et al*. [29] have designed a novel 32-bit crypto processor based on *Application Specific Instruction set Processor* (ASIP) for the applications of *International Data Encryption Algorithm* (IDEA), *Message Digest Method 5* (MD5) and AES. This system has two data buses as well as nine function units. This processor consists of 32-bit instruction sets for the execution of input/ output reference (I/OR) instructions, memory reference, and register reference. The output of the encryption process in 128-bit input modules is obtained after 122, 146, and 170 clock cycles for AES-128, AES-192, and AES-256, respectively. It utilizes 95 clock cycles to encode a 64-bit input module when we exploit the IDEA system. Whereas MD5 system (system or algorithm??). Hash requires 469 clock cycles to encode 512-bit of data. The simulation of the proposed processor is compared with some past systems and the efficiency of the proposed system is proved by comparing the throughput, speed, latency, and flexibility.

Lee *et al*. [30] have proposed heterogeneous *dual-processing Element* (dual-PE) architecture and priority-oriented scheduling of *EC Scalar Multiplication* (ECSM) with random techniques. The performance of hardware is good when the ECC systems are secure in cryptanalysis. The information stored in the non-protected hardware device will be extracted by various physical security threats. It attains a *Power-Analysis-Resistant Dual-Field ECC* (DF-ECC) processor. Memory synchronization is utilized in the memory hierarchy to achieve increased bandwidth of data. This architecture is executed over GF(p^{160}) and GF(2^{160}) using a UMC 90-nm CMOS process with a 0.41 mm^2 core area.

3. EXITING METHODOLOGIES

3.1 Problem Methodology

Khan *et al.* [31] have developed a high-speed ECC processor that works based on *Point Multiplication* (PM) on FPGA. A divided pipelined full-accuracy multiplier is utilized for lessening the latency, and the Lopez-Dahab Montgomery PM process is altered for focused planning to eliminate data dependency which results in decreasing the total number of clock cycles (CCs) essential in the system. The high-speed ECC processor is designed with very little latency to attain a good performance limit. These are realized with a new technique based on pipelining that aids in accomplishing high clock frequencies and a detailed study of field multipliers along with various arrangements it. The ECC

system has been designed using Xilinx Field Programmable Gate Arrays such as Virtex4, Virtex5, and Virtex7 families. This ECC processor realizes high speed along with the efficient area-time performance on Virtex4, Virtex5, and Virtex7. For GF (2^{163}) operation with a single multiplier in Virtex4 design consumes 12964 slices, 3077 flip flops (FFs), 23468 lookup tables (LUTs), and 210 MHz frequency; single multiplier in Virtex5 design consumes 4393 slices, 3090 flip flops (FFs), 16090 lookup tables (LUTs) and 228 MHz frequency; three multipliers in Virtex5 design consumes 11777 slices, 3403 flip flops (FFs), 42192 lookup tables (LUTs) and 113 MHz frequency; single multiplier in Virtex7 design consumes 4150 slices, 3747 flip flops (FFs), 14202 lookup tables (LUTs) and 352 MHz frequency; three multiplier in Virtex7 design consumes 11657 slices, 7969 flip flops (FFs), 41090 lookup tables (LUTs) and 159 MHz frequency. For GF (2^{571}) operation with a single multiplier in Virtex7 design consumes 50336 slices, 29217 flip flops (FFs), 141078 look-up tables (LUTs), and 111 MHz frequency.

From the above reviews [21]-[31], the prevailing cryptographic processors are mainly designed to provide strong models for blind digital signatures to fulfill the security guarantees, and their processes mainly focus on masking data. While hardware-based encryption provides more security than software-based approaches, it does add to the expense and complexity of the solution. Because it is unlikely that every application can benefit enough to justify the additional cost and effort, organizations should deploy hardware-based solutions only for applications with sufficiently high requirements for security and performance. Moreover, case of hardware design [21], In the literature [23]-[27], authors design systems in such a way that they maintain high throughput along with reliability and robustness. However, these incur high costs. Hardware design that is application specific [28]-[30] can attain extraordinary power and offer high reliability however these exploits more cost because of the additional hardware employed. In recent times, high-speed ECC processors have utilized multipliers for the optimization of throughput. However, the area occupied by the hardware is very high as it utilizes three multipliers. These multipliers decrease the latency and increase the path delay in the system.

Our contributions. These issues are rectified by an Efficient Hybrid Signcryption Processor (EHSP), which works based on the techniques called flexible configuration of encryption and signature. To the best of our knowledge, the proposed EHSP technique is the first hardware architecture for signcryption. The EHSP uses the MKD hashing scheme for key encapsulation; and the ECC processor for data encapsulation. The key can be encrypted by the known public key and the message encrypted by the already encrypted key. The cipher text of the EHSP design maintains the meaningless information and the control parameters are active parameters to avoid the decryption point of view. The ECC processor contains three units an arithmetic unit, a memory unit, and a control unit. Here, our aim is the optimization of the arithmetic unit by utilizing a flexible multiplier as an alternative to single and three multipliers as presented in [31]. This flexible design provides support in all suggested field sizes of various GFs without the help of reconfigurable hardware devices, which is suitable for efficient hardware design. Major contributions of our proposed EHSP design are described as follows:

a) In EHSP design, the MKD hashing scheme is used to encapsulate the key, and the ECC processor is utilized for encapsulation of the data.

b) The ECC processor comprises three parts. They are the arithmetic unit, memory, and control unit; the Arithmetic unit is enhanced as it incurs high hardware costs when compared with all other units.

c) The proposed EHSP design is designed, synthesized, and simulated on various FPGAs such as the Virtex4, Virtex5, and Virtex7. The performance is then compared with the exiting topologies concerning utilization of hardware, throughput, and energy consumption.

3.2 Proposed EHSP Design

The Enhanced Hybrid Signcryption Processor (EHSP) design is a novel cryptography mechanism proposed for applications where high-level security is needed. It enables the transmitter to encrypt a confidential message to a receiver. The system model of the proposed EHSP design is shown in Figure-1. The key is encapsulated by the MKD hashing technique using a public key, and the plain text or images are encrypted by our proposed ECC processor by utilizing an encrypted key. And then, the data or the messages would be signed with the aggregated signature. This creates an effective aggregate signature without full control of all the valid individual signatures. The proposed HAS technique requires more computational time to complete the signature verification and decryption.

4. EFFICIENT HYBRID SIGNCRYPTION PROCESSOR (EHSP)

The operation and the mathematical modelling of the EHSP scheme along with an ECC processor and modified Kurosawa and Desmedt hashing are explained in detail as follows.

4.1 Data Encapsulation Process by Utilizing ECC Processor

A data encapsulation mechanism uses the symmetric key from signcryption KEM to encrypt the data of any random size. Elliptical curve cryptography is an open key encryption method that is based on the theory of elliptic curve for the generation of efficient cryptographic keys with high speed and smaller length. Elliptical curve cryptography creates keys abiding the characteristics of the elliptic curve mathematical equation, which also acts as a substitute for the conventional technique of generating

the product of very large prime numbers. Elliptical curve cryptography can attain a level of security with a key of 164-bit, whereas other structures need a 1024-bit key to attain the same level of security. Elliptical curve cryptography is commonly used nowadays for its efficient security features along with low computational power and less usage of battery. Also, it is capable of delivering a similar cryptographic strength as the Rivest-Shamir-Adleman (RSA)-based system with the added advantage of having smaller key sizes. Ex. A 256-bit Elliptical curve cryptography key is equal to a 3072-bit Rivest-Shamir-Adleman keys. Most commonly employed symmetric algorithms utilize at least 128-bit keys and hence asymmetric keys must have at least a security of this level.

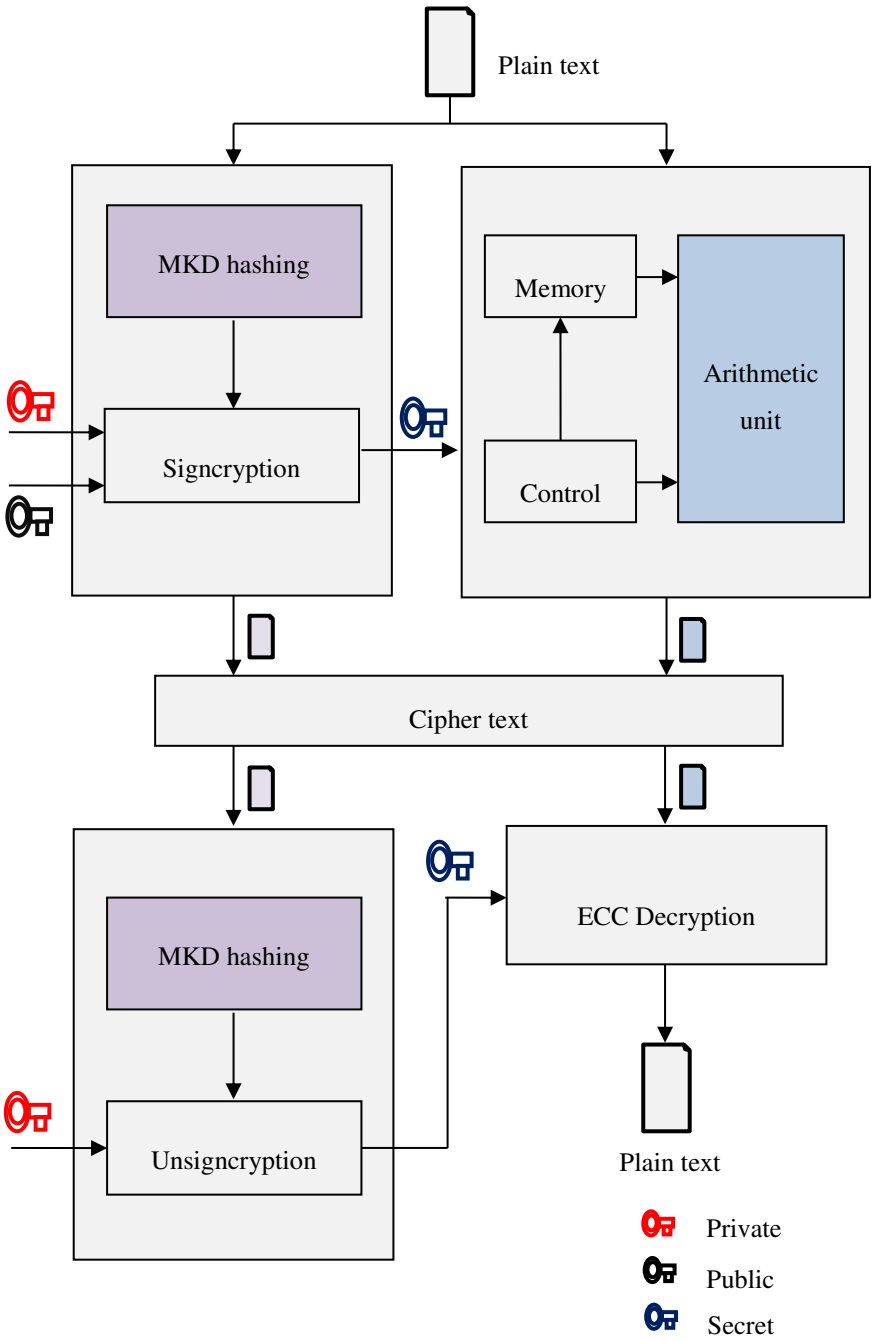


Figure-1. Structure of proposed Efficient Hybrid Signcryption Processor (EHSP).

4.1.1 Background mathematic

A field is a group of elements with various operations specified for the elements of a particular set that is associated with distinct operations such as addition, multiplication, subtraction, and division. Few conditions have to be satisfied for an ideal field. They are

- In the addition and multiplication operation, the set is closed. (E.x) If x and y are present in a particular set then, they are x + y and x × y

- Associative law is applicable for addition and multiplication,

Addition: x + (y + z) = (x + y) + z
Multiplication: x × (y × z) = (x × y) × z

- Commutative law is valid for addition and multiplication,

Addition: x + y = y + z
Multiplication: x × y = y × x

- Addition and multiplication should satisfy the law of identity

Ex. Addition: x + 0 = x
Multiplication: x × 1 = x

- Additive and multiplicative inverses should be there for any particular element in the set.

- Distributive law should be satisfied. Consider the inputs x, y and z are present in a set, then x × (y + z) = (x × y) + (x × z)

A finite field (FF) or *Galois Field* (GF) is generally a field where the set has a fixed number of elements. All the integers present in a set cannot be used in a fixed field as there are several integers present in it. In Elliptical curve cryptography, the commonly employed curve is a generic curve and in the Galois field, various arithmetic operations are performed. The Galois field is an extension binary field GF (2m) or prime field GF (p). In hardware, the GF (2m) design is simpler to implement.

$$y^2 \bmod m = x^3 + ax^2 + b \bmod m \quad (1)$$

Generally, two algorithms are employed for the design of scalar point multiplication. They are the Montgomery ladder methodology and the Binary Method. Here, the elementary method of scalar point multiplication is a binary methodology and it is also commonly referred to as the double and adds method. For every k^{th}-bit, the scalar point multiplication is iterated and for each such iteration, a point doubling operation is executed. If the k^{th}-bit is unity, a point addition operation is performed. The second methodology is the most commonly employed technique called the Montgomery ladder method for the execution of scalar point multiplication. For point doubling and point addition x -x-coordinate is utilized and this is the key advantage of this methodology. Therefore, for each iteration, registered variables and the number of field operations are decreased. In most of the projective coordinates, inversion operation is completely avoided instead X and Z coordinates are utilized. Inversion operation is avoided as it requires 6 m-bit register variables for calculating the number of iterations needed. This is the same as the number of m-bit registers used in the projective coordinate methodology.

When we compare the binary method with the Montgomery ladder method, the Montgomery ladder method accomplishes point addition and point doubling in each iteration irrespective of the value of ki. Generally, the scalar point multiplication can be categorized with four computing elements in GF (2^m). These are addition, multiplication, inversion, and squaring operations. An inversion operation is executed by the combination of multiplication and squaring. The requirements of a high throughput ECC operation are optimizing the area overhead, critical path delay of any logic, and number of clock cycles utilized in the design. Throughput can be enhanced by parallel operation of multiplications and by utilizing large size multiplication. The structure of the proposed ECC processor is shown in Figure-2. It comprises three key components, such as memory unit, control unit, and arithmetic unit. Our focus is on the arithmetic unit as it consumes a lot more hardware when compared with other units.

4.2 Arithmetic Logic Unit

Consider $C = (c_{m-1}, c_{m-2}, \dots\dots\dots c_0) \in GF(2^m)$ and $D = (d_{m-1}, d_{m-2}, \dots\dots\dots d_0) \in GF(2^m)$ are two domains and field addition in the binary extension field of GF (2^m) can be obtained by bit wise XOR arithmetic operation as,

$$C + D = \sum_{i=0}^{m-1} (c_i \oplus d_i) x^i \quad (2)$$

Thus, the arithmetic unit is designed using m XOR portions with a single clock cycle latency. Fast squarer is simulated with single latency. The process of squaring involves expansion with the help of interleaving techniques with 0's. When it is implemented in a hardware device, it is interleaving of 0-bit lines to expand it to 2n bits. Further minimization of this polynomial is reasonable as polynomial reduction utilizes a trinomial function. The reduced polynomial is so small that it requires no reduction for $\left\lfloor \frac{n}{2} \right\rfloor$. The simulated squarer utilizes a wired XOR circuit. $C(x) \in GF(2^m)$ represents the polynomial as:

$$C(x) = \sum_{i=0}^{m-1} c_i x^i \text{ , where } c_i \in GF(2^m) \quad (3)$$

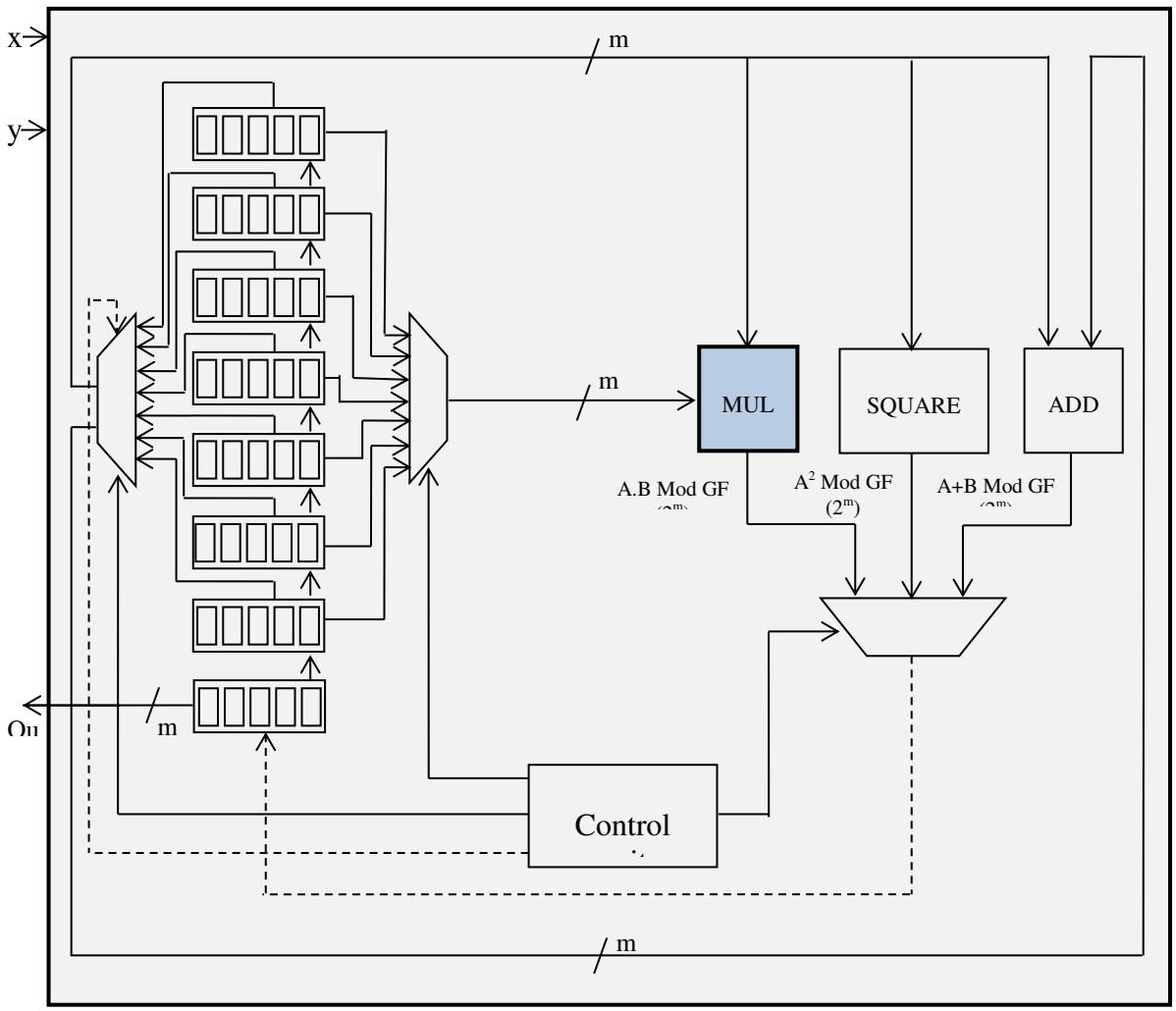


Figure-2. Hardware architecture of ECC processor.

Squaring operation in GF (2^m) can be modelled as shown below:

$$C^2(x) = c_{m-1}x^{2m-2} \dots\dots\dots + c_2x^4 + c_1x^2 + c_0 \bmod F(x) \quad (4)$$

Multipliers are unavoidable with the development of communication. Less delay, low power dissipation, minimized area utilization and regularity in layout are major things that have to be taken care of while we design a multiplier. In a fixed width multiplier i.e. for a N×N multiplication, the output is a product of 2N. Hence, it is understood that the output is of fixed width and the bit length cannot be controlled. According to the need of various applications, to change the length of the bits the fixed width multipliers should be converted into flexible multipliers.

The proposed architecture of a flexible multiplier over the Galois field distributes a typical gain in energy consumption and area-delay product. The multiplier over GF(2^m) with a polynomial basis design is based on the field element expressed in the polynomial form p(x) and field multiplication can be performed. The multiplication is represented as follows:

$$C(x) = a_i \times b_i (mod\ p(x)) \quad (5)$$

where p(x) represents the irreducible polynomial over GF(2^m), then calculate the parameter m_{ij} for i = 0, 1, …, 2(n - 1) and j = 0, 1, …, n -1 using,

$$x^i \bmod p(x) = \sum_{i=0}^{m-1} m_{ij}\, x^j \quad (6)$$

AND logic is used to compute the parameter S_i by,

$$S_i = \sum_{j+k=i} a_j\, b_k \quad (7)$$

XOR logic is used to compute the parameter c_i by,

$$c_i = \sum_{i=0}^{2(m-1)} m_{ij}\, S_i \quad (8)$$

Finally, the multiplication is represented as follows:

$$C(x) = \sum_{i=0}^{m-1} c_i \, x^i \quad (9)$$

The architecture of the flexible multiplier is shown in Figure-3. The array of AND gates is connected with a register (A) which is the multiplicand and a polynomial register (P) for aiding the multiplier bits. These multiplier bits are exchanged with appropriate tri-state buffers/registers. In the multiplier design, the efficiency is measured in terms of area overhead when an array of AND gate and Tristate buffer logic are introduced.

4.2.1 Memory and control units

The memory registers X and Y are the fixed registers utilized for creating our public key. These public keys aid in the creation of four more new public keys. Let M_1, M_2, M_3, and M_4 be the newly created public keys that are stockpiled in registers R_M_1, R_M_2, R_M_3, and R_M_4, respectively. Temporary variables that are created in any calculation are stored in R_Temp. Registers R_M_1 and R_M_3 are used for loading the constant parameters required in the execution of a program in the ECC processor. The performance of the processor can be improved by increasing the speed and suitable design of the memory unit. RAM used in this processor is optimized as well as distributed. M-bit input can be loaded at any location of the register and two M-bit output registers can access data from any location of a register. Execution of various operations such as data handling, arithmetic operations, doubling, and multiplication conferring to Algorithm 1, is carried out with the help of a control unit that works based on the FSM model.

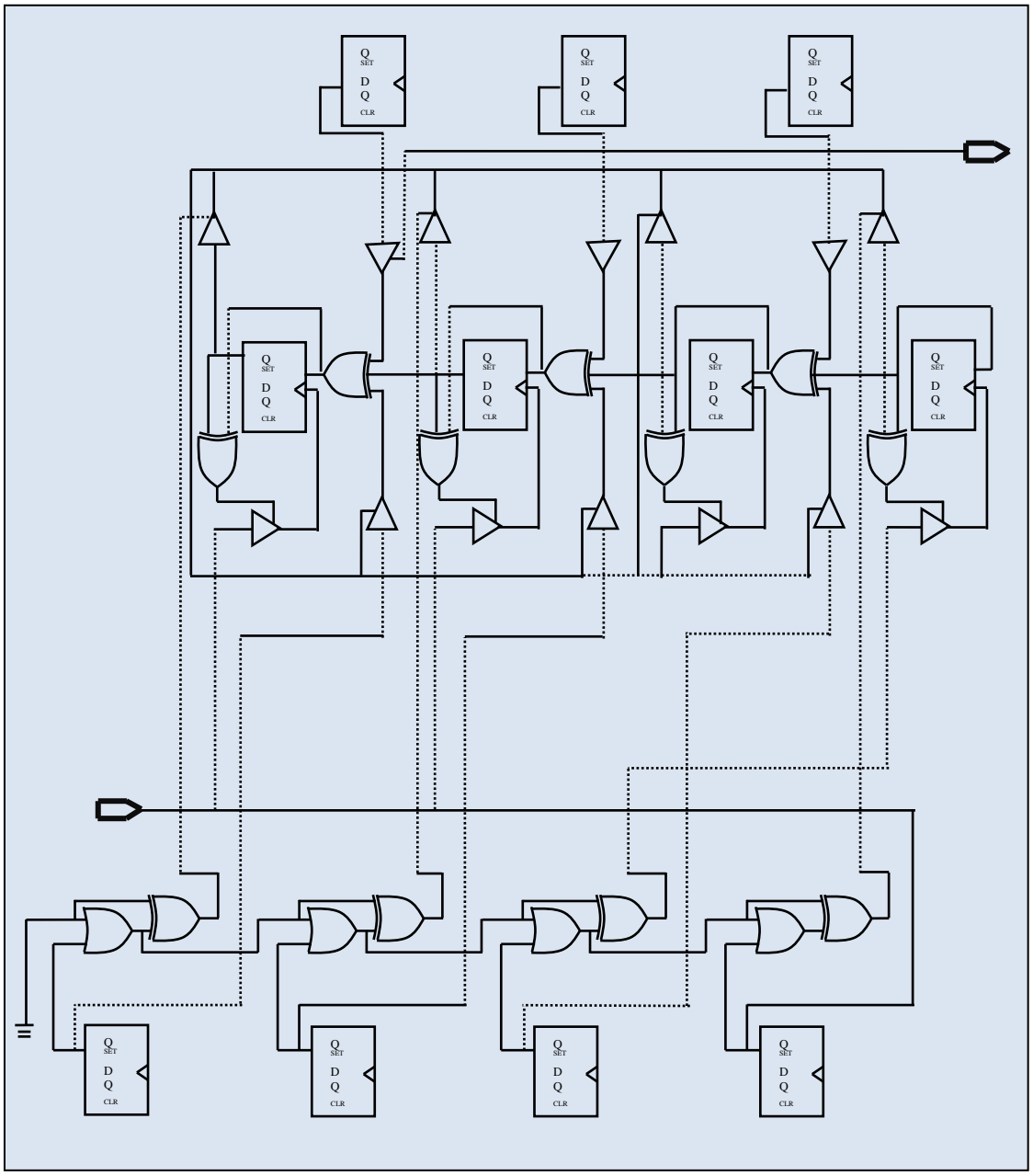


Figure-3. Structure of Flexible multiplier for ECC processor.

Algorithm 1 ECC processor using flexible multiplier	
Input: sk = (k_0, k_1, … k_{t-1}) ϵ GF (2^m) and p = (x, y) ϵ GF (2^m) Output: Q (x, y) = k × p	
1.	if sk = 0 or x = 0, then
2.	return Q = (0, 0) and stop
3.	set $M_1 \leftarrow x$, $M_3 \leftarrow 1$, $M_2 \leftarrow x^4+b$, $M_4 \leftarrow x^2$
4.	for i = t - 2 decrease to 0, do
5.	if $sk_i = 1$,
6.	set (M_1, M_3) = add | M_1, M_2, M_3, M_4 |, set (M_2, M_4) = square | M_2, M_4 |
7.	else if
8.	set (M_2, M_4) = add | M_1, M_2, M_3, M_4 |, set (M_1, M_3) = square | M_1, M_3 |
9.	end
10.	set Q (x, y) = mul | M_1, M_2, M_3, M_4|
Return: Q (x, y)	

4.3 Modified Kurosawa and Desmedt (MKD) Hashing Scheme

In the first phase, the m-bit input message modules are XORed with the initial m bits of the state, and inserted with various CT applications. This stage is done when all message modules are prepared. In the second stage, the initial m bits of the state are returned as hash bits, interleaved with CT applications. This stage is completed when the anticipated length of the hash is generated. Each round contains five stages as stated in Algorithm 2. The entire process on the indices is completed under GF (2^m), and the spin amounts are constant. Entire rounds of MKD hash achieve similar operations except for various CT. MKD hash is planned such that CPU word length is fixed, which leads to effective usage of CPU resources across a wide range of processors. The ideal bit size for MKD hash permutation (1,600) is chosen in favor of 163-bit architectures. Figure-4 shows the Hardware architecture of the MKD hash function.

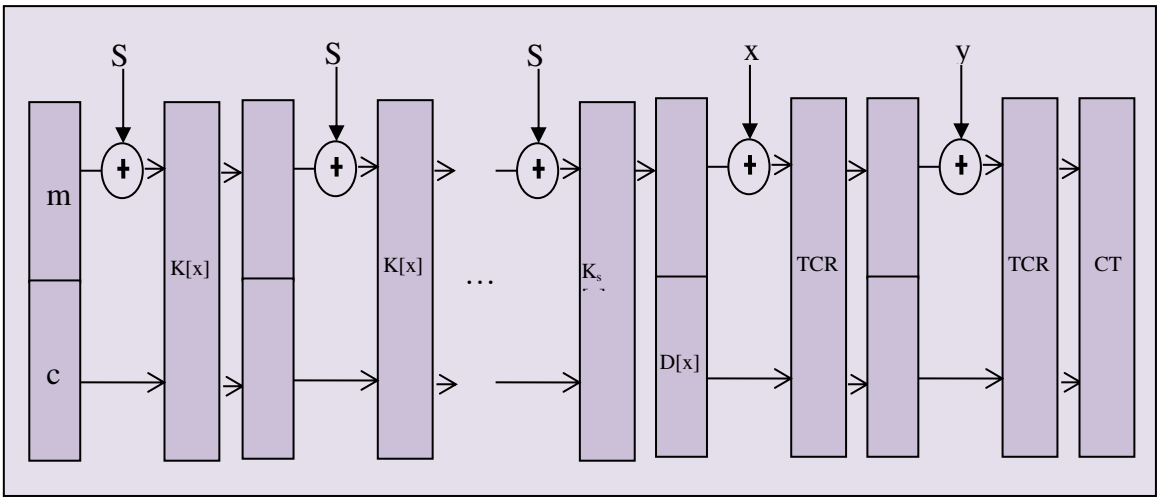


Figure-4. Hardware architecture of MKD hash function.

Algorithm 2 MKD hashing function	
Input: S = (s_1, s_2 … s_n) ϵ GF (2^m) and (x, y) ϵ GF (2^m) Output: CT	
1.	K_s [x] = A [x, s_1] © A [x, s_2] © A [x, s_3] © A [x, s_4]
2.	D [x] = K_s [x - 1] © (K_s [x + 1] <<< 1)
3.	TCR [g_1^s, g_2^s] = A [x, y] © D [x]
4.	B [y, 2x+3y] = A [x, y] <<< D [x]
5.	TCR [g_1^s, g_2^s] = B [x, y] © (B [x+1, y] ^ B [x+2, y])
6.	CT = TCR [g_1^s, g_2^s] © K_s [x]
Return: CT	

Note © is modulo addition

5. SIMULATION RESULTS

Our proposed processor EHSP is programmed and simulated using Verilog Hardware Description Language. Xilinx ISE 14.5 design suite is used for the simulation, synthesis, mapping, and routing process. For the comparison purpose, the design was synthesized in various Xilinx families such as Virtex4 (XC4VLX60), Virtex5 (XC5VLX50), and Virtex7 (XC7V330T). The FPGA chosen was the smallest in the Virtex family which can easily accomplish our design concerning area and pin count. The designs were synthesized and mapped using XST tools from Xilinx. The verification process of the EHSP processor is carried out with the ISIM simulator. All the simulations are done in a personal computer with an operating system of Windows 7 with a RAM of 4GB and an Intel processor of Core i3. Figure-5a shows the RTL Schematic of the proposed EHSP processor with GF (2^{163}), executed in the Virtex7 (XC7V330T) family.

Figure-5b shows the detailed architecture of the EHSP Processor. Figure-6 shows the device utilization of the EHSP processor design of GF (2^{163}) with the Virtex7 FPGA family.

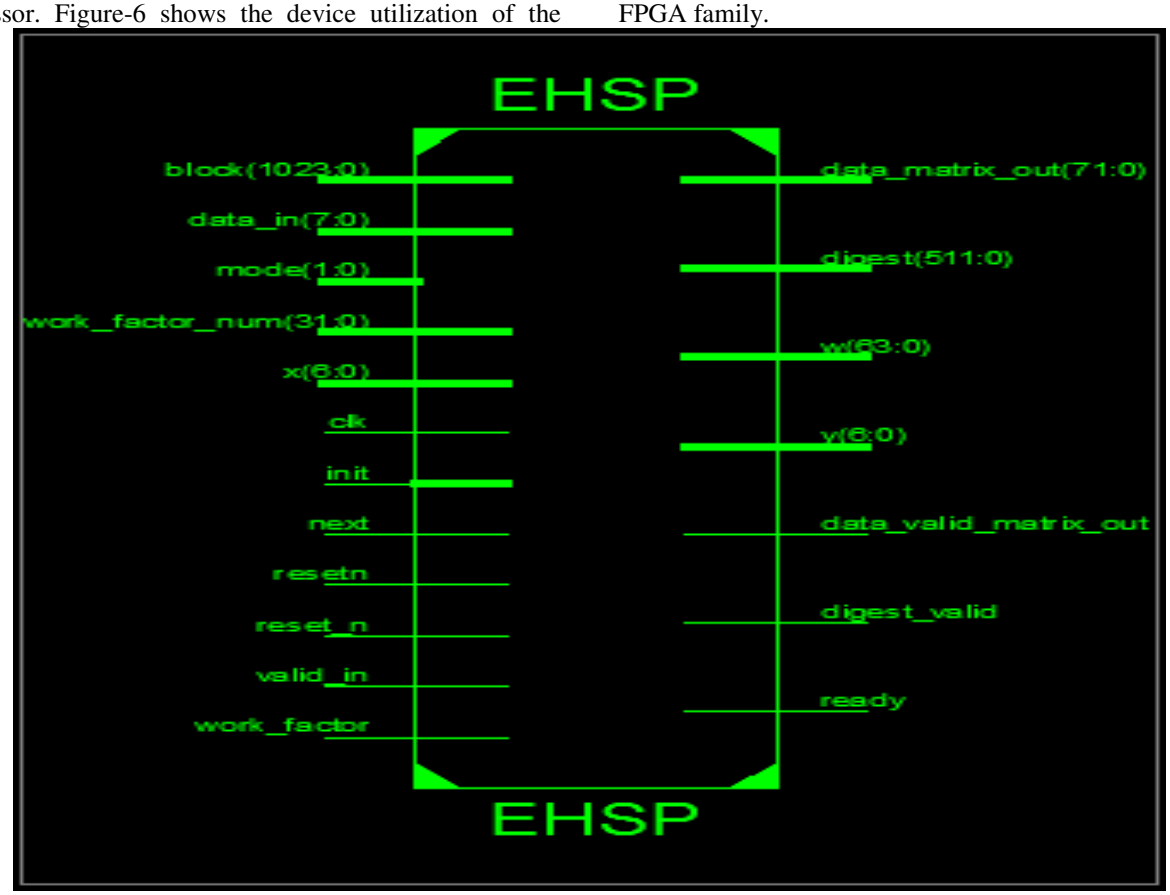


Figure-5a. RTL schematic proposed EHSP design (Overview architecture).

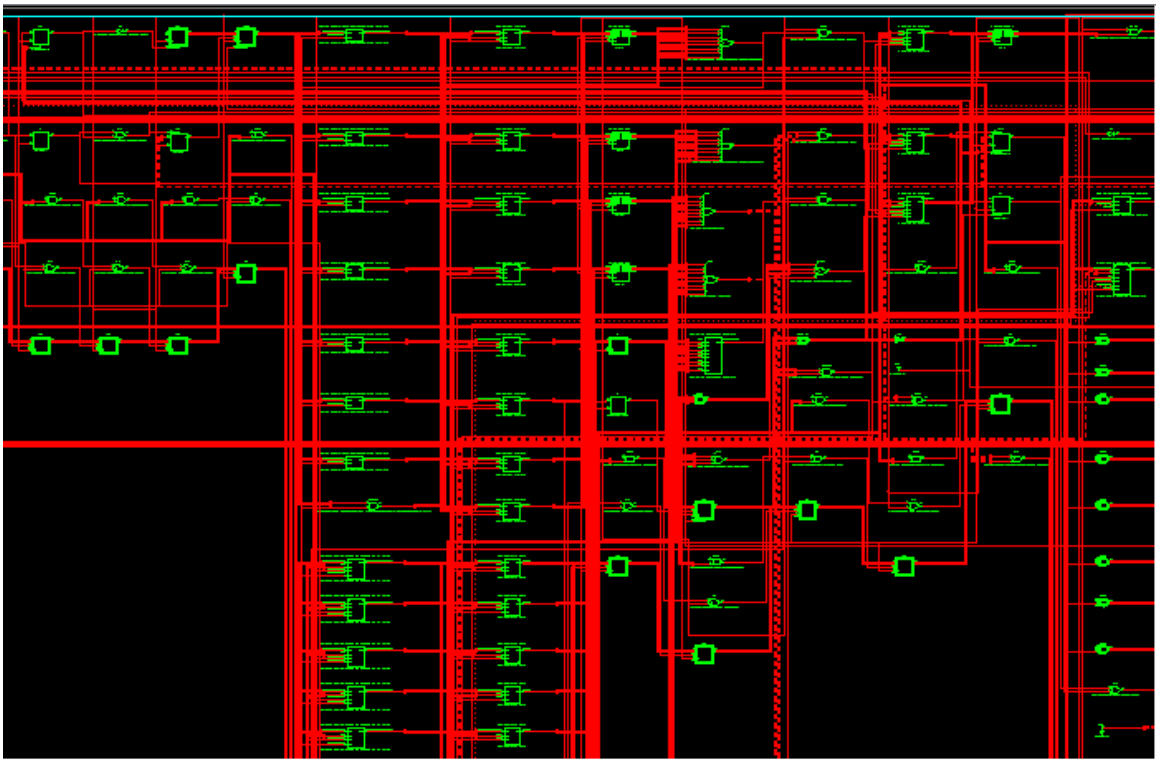


Figure-5b. RTL schematic proposed EHSP design (Detailed architecture).

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	4553	26624	17%
Number of Slice Flip Flops	6077	53248	11%
Number of 4 input LUTs	14502	53248	9%
Number of bonded IOBs	77	448	17%
Number of FIFO16/RAMB16s	2	160	1%
Number of GCLKs	1	32	3%

Figure-6. Device utilization of proposed EHSP design of GF (2^{163}) with Virtex7 FPGA family.

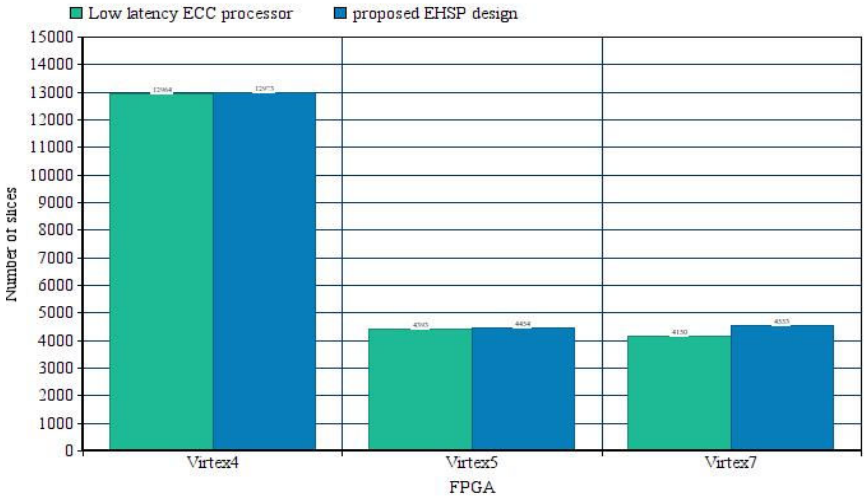
Table-1. Performance comparison with exiting FPGA implementations.

Design	Field size (bits)	Platform	Device utilization (Number of …)		
			Slices	Flip-flops	Lookup tables (LUTs)
[21]	256	Virtex 4 Virtex 6	13158 11104	- -	- -
[22]	256	Virtex 7	377	992	995
[23]	256	Virtex 2	-	-	12425
[25]	128	Virtex 5 Virtex 6 Virtex 7	2940 2537 2617	- - -	- - -
[27]	128&256	Virtex 6	102	-	
[28]	256	Virtex 6 Virtex 7	1620 1558	- -	3360 3370
[31]	163 571	Virtex 4 Virtex 5 Virtex 7 Virtex 7	12964 4393 4150 50336	3077 3090 3747 29217	23468 16090 14202 141078
EHSP*	163 571	Virtex 4 Virtex 5 Virtex 7 Virtex 7	12973 4454 4553 50349	3145 3569 6077 29571	23470 16214 14502 141587

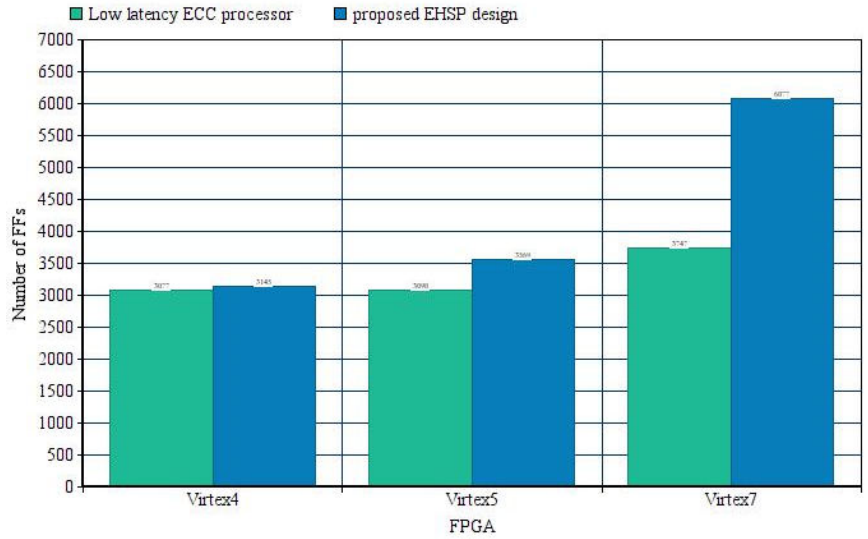
proposed design

5.1 Hardware Utilization Comparison

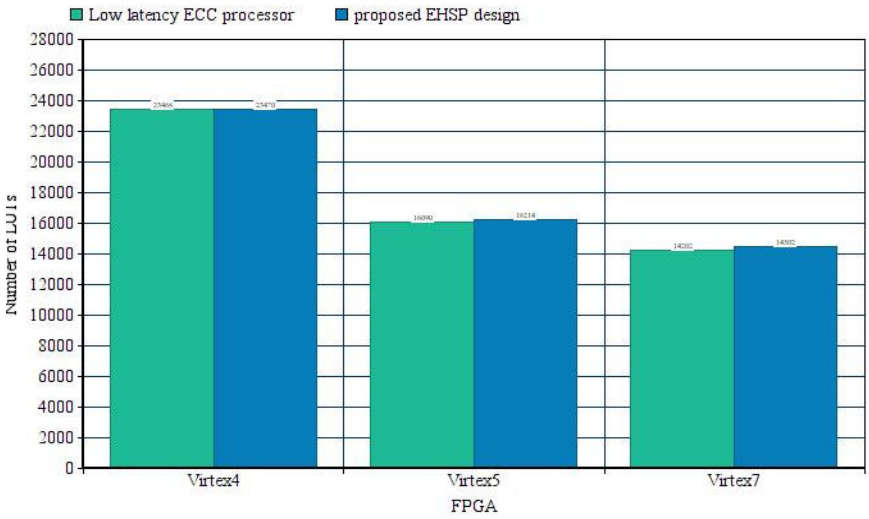
After complete synthesize, the device utilization can be verified by studying the generated design summary. Comparison is made between our novel EHSP design and prevailing cryptography processors such as ECC processors utilizing radix-4 booth encoding which is based on interleaved modular multiplier [21], ECC processors using double scalar multiplier [22], dual field ECC processors [23], 128 bit AES [25], hybrid AES + Grøstl hash [27], RNS based ECC [28] and low latency ECC processor [31]. Performance is compared concerning the utilization of resources for the proposed processor design and exiting designs. The total number of slices utilized by EHSP in Virtex7 is 4553, which is much less when compared with other designs. Figures 7a to Figure-7c exhibit the comparison of the proposed EHSP design and low latency ECC processor [31] over GF (2^{163}) concerning the total number of slices, Flip Flops used, and LUTs. For the value of m = 163, the proposed EHSP design incurs the same amount of hardware unit, however, it provides dual security.



(a)



(b)



(c)

Figure-7. Comparison of device utilization (a) number of slices (b) FFs (c) LUTs achieved by different designs on different FPGAs.

Table-2. Maximum frequency comparison with exiting FPGA implementations.

Design	Field size (bits)	Platform	Maximum frequency (MHz)
[21]	256	Virtex 4 Virtex 6	40 70
[22]	256	Virtex 7	205.634
[23]	256	Virtex 2	55.7
[25]	128	Virtex 5 Virtex 6 Virtex 7	704.7 740.7 81.3
[27]	128&256	Virtex 6	413
[31]	163 571	Virtex 4 Virtex 5 Virtex 7 Virtex 7	210 228 352 111
EHSP	163 571	Virtex 4 Virtex 5 Virtex 7 Virtex 7	221 231 373.756 121.584

5.2 Comparison of Maximum Clock Frequency

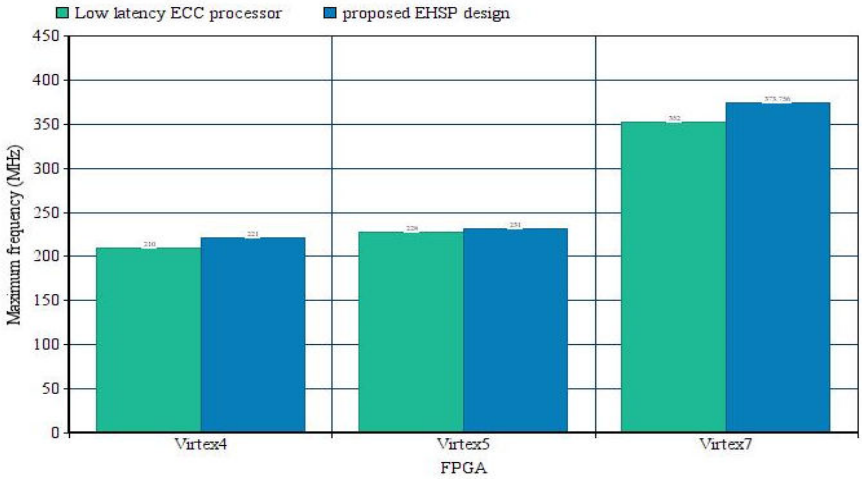


Figure-8. Comparison of frequencies achieved by proposed and exiting designs on different FPGAs.

The clock frequency of operation of proposed EHSP design and exiting cryptography processors such as interleaved modular multiplier [21], ECC processor using double scalar multiplier [22], dual field ECC processor [23], 128 bit AES [25], hybrid AES + Grøstl hash [27], and low latency ECC processor [31]. Table-2 displays the comparison of performance concerning the maximum frequency attained between the EHSP designs and exiting schemes. Figure-8 labels the maximum frequency comparison of the proposed design with low latency ECC processor [31] over GF (2^{163}) with various FPGA families. For the value of m=163, the EHSP (hybrid) design achieves a maximum clock operating frequency than all other prevailing processor designs [31].

Table-3. Power consumption comparison with exiting FPGA implementations.

Design	Field size (bits)	Platform	Power consumption (mW)
[21]	256	Virtex 4 Virtex 6	173 192
[22]	256	Virtex 7	208
EHSP*	163 571	Virtex 4 Virtex 5 Virtex 7 Virtex 7	102 109 127 171

5.3 Power Comparison

Comparison of the power dissipation of EHSP processors with the prevailing cryptography processors such as ECC processors utilizing radix-4 booth encoding based interleaved modular multiplier [21] and ECC processors using double scalar multiplier [22]. Table 3 illustrates the performance comparison concerning power dissipation attained from proposed and prevailing methodologies. Table 3 analyses the power dissipation of the proposed design and exiting designs [21], [22]. For m=256 bits, the proposed EHSP (hybrid) design consumes low power in comparison with all other processors.

6. CONCLUSIONS

In this paper, an efficient hybrid signcryption processor (EHSP) is proposed with the help of the data encapsulation method (DEM) and Flexible key Encapsulation Method (KEM). The modified Kurosawa and Desmedt (MKD) hashing scheme is utilized for encapsulating the key and the elliptic curve cryptography (ECC) processor which aids in encapsulating the data. The flexible data encryption is realized by the flexible multiplier over GF (2^{751}) without any reconfigurable design. The simulation results demonstrate the efficiency of the proposed EHSP design, which consumes very less utilization of hardware, lower power consumption, and maximum frequency in various FPGAs Virtex 4 (XC4VLX60), Virtex 5 (XC5VLX50) and Virtex 7 (XC7V330T). To conclude, the proposed EHSP design provides good security by maximizing the time of the key.

REFERENCES

[1] H. Petersen and M. Michels. 1988. Cryptanalysis and improvement of signcryption schemes. IEE Proceedings - Computers and Digital Techniques. 145(2): 149.

[2] W. He and T. Wu. 1999. Cryptanalysis and improvement of Petersen-Michels signcryption scheme. IEE Proceedings - Computers and Digital Techniques. 146(2): 123.

[3] Q. Huang, D. Wong and G. Yang. 2011. Heterogeneous Signcryption with Key Privacy. The Computer Journal. 54(4): 525-536.

[4] Bhuvana B. P. and Bhaaskaran VS Kanchana. 2018. Positive feedback symmetric adiabatic logic against differential power attack. In 2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID), pp. 149-154. IEEE.

[5] M. Senthil murugan and T. Sasilatha. 2019. Hardware Implementation of Hybrid Model Encryption Algorithm for Secure Transmission of Medical Images. International Journal of Engineering and Advanced Technology, ISSN No 2249-8958.

[6] M. Senthil murugan and T. Sasilatha. 2019. Area-Efficient and High-Speed Hardware Structure of Hybrid Cryptosystem (AES-RC4) For Maximizing Key Lifetime Using Parallel Sub Pipeline Architecture. Journal of Concurrency and Computation. Practice and Experience (Web of Science), ISSN No 1532-0634.

[7] Y. Zhang, X. Chen, and H. Li. 2012. Key-Evolving Hierarchical ID-Based Signcryption. The Computer Journal. 56(10): 1228-1248.

[8] Yi Wang, J. Leiwo, T. Srikanthan and Yu Yu. 2006. FPGA based DPA-resistant Unified Architecture for Signcryption. Third International Conference on Information Technology: New Generations (ITNG'06).

[9] M. Senthil murugan and T. Sasilatha. 2019. Implementation of Advanced Encryption Standard Algorithm on Steganography. Journal of Recent Technology and Engineering, ISSN No 2277-3878

[10] A. Elbirt, W. Yip, B. Chetwynd and C. Paar. 2001. An FPGA-based performance evaluation of the AES block cipher candidate algorithm finalists. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 9(4): 545-557.

[11] Y. Xiaohui, D. Zibin, L. Yuanfeng and W. Ting. 2007. Researching and implementation of reconfigurable Hash chip based on FPGA. Journal of Systems Engineering and Electronics. 18(1): 183-187.

[12] I. Ahmad and A. Das. 2007. Analysis and Detection of Errors in Implementation of SHA-512 Algorithms on FPGAs. The Computer Journal. 50(6): 728-738.

[13] M. Senthil murugan and T. Sasilatha. 2017. Design of Hybrid Model Cryptographic Algorithm for Wireless Sensor Network. Journal of Pure and Applied Mathematics , ISSN No 1314-3395

[14] A. Cilardo and N. Mazzocca. 2013. Exploiting Vulnerabilities in Cryptographic Hash Functions Based on Reconfigurable Hardware. IEEE Transactions on Information Forensics and Security. 8(5): 810-820.

[15] M. Juliato and C. Gebotys. 2013. A Quantitative Analysis of a Novel SEU-Resistant SHA-2 and

HMAC Architecture for Space Missions Security. IEEE Transactions on Aerospace and Electronic Systems. 49(3): 1536-1554.

[16] E. Taqieddin, O. Abu-Rjei, K. Mhaidat and R. Bani-Hani. 2015. Efficient FPGA Implementation of the RC4 Stream Cipher using Block RAM and Pipelining. Procedia Computer Science. 63: 8-15.

[17] R. Farashahi, B. Rashidi and S. Sayedi. 2014. FPGA based fast and high-throughput 2-slow retiming 128-bit AES encryption algorithm. Microelectronics Journal. 45(8): 1014-1025.

[18] H. Michail, G. Athanasiou, G. Theodoridis and C. Goutis. 2014. On the development of high-throughput and area-efficient multi-mode cryptographic hash designs in FPGAs. Integration, the VLSI Journal. 47(4): 387-407.

[19] R. Laue and S. Huss. 2007. Parallel Memory Architecture for Elliptic Curve Cryptography over GF (P) Aimed at Efficient FPGA Implementation. Journal of Signal Processing Systems. 51(1): 39-55.

[20] Y. Dan, X. Zou, Z. Liu, Y. Han and L. Yi. 2009. High-performance hardware architecture of elliptic curve cryptography processor over GF (2163). Journal of Zhejiang University-SCIENCE A. 10(2): 301-310.

[21] K. Javeed and X. Wang. 2018. FPGA Based High Speed SPA Resistant Elliptic Curve Scalar Multiplier Architecture.

[22] Z. Liu, J. Gro Bschadl, Z. Hu, K. Jarvinen, H. Wang and I. Verbauwhede. 2017. Elliptic Curve Cryptography with Efficiently Computable Endomorphisms and Its Hardware Implementations for the Internet of Things. IEEE Transactions on Computers. 66(5): 773-785.

[23] Z. Liu, D. Liu and X. Zou. 2017. An Efficient and Flexible Hardware Implementation of the Dual-Field Elliptic Curve Cryptographic Processor. IEEE Transactions on Industrial Electronics. 64(3): 2353-2362.

[24] J. Mitra and T. Nayak. 2017. Reconfigurable very high throughput low latency VLSI (FPGA) design architecture of CRC 32. Integration, the VLSI Journal. 56: 1-14.

[25] M. Chellam and R. Natarajan. 2017. AES Hardware Accelerator on FPGA with Improved Throughput and Resource Efficiency. Arabian Journal for Science and Engineering.

[26] J. Sugier. 2017. Simplifying FPGA Implementations of BLAKE Hash Algorithm with Block Memory Resources. Procedia Engineering. 178: 33-41.

[27] N. At, J. Beuchat, E. Okamoto, I. San and T. Yamazaki. 2017. A low-area unified hardware architecture for the AES and the cryptographic hash function Grøstl. Journal of Parallel and Distributed Computing. 106: 106-120.

[28] S. Asif, M. Hossain, Y. Kong and W. Abdul. 2017. A Fully RNS based ECC Processor. Integration, the VLSI Journal.

[29] K. Shahbazi, M. Eshghi and R. Faghih Mirzaee. 2017. Design and implementation of an ASIP-based cryptography processor for AES, IDEA, and MD5. Engineering Science and Technology, an International Journal. 20(4): 1308-1317.

[30] J. Lee, S. Chung, H. Chang and C. Lee. 2014. Efficient Power-Analysis-Resistant Dual-Field Elliptic Curve Cryptographic Processor Using Heterogeneous Dual-Processing-Element Architecture. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 22(1): 49-61.

[31] Z. Khan and M. Benaissa. 2017. High-Speed and Low-Latency ECC Processor Implementation Over GF ($2^{m})$ on FPGA. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 25(1): 165-176.