Open Access **Baghdad Science Journal** **P-ISSN: 2078-8665**
2020, 17(3) Supplement (September):1029-1040 **E-ISSN: 2411-7986**

DOI: http://dx.doi.org/10.21123/bsj.2020.17.3(Suppl.).1029

Improved throughput of Elliptic Curve Digital Signature Algorithm (ECDSA) processor implementation over Koblitz curve k-163 on Field Programmable Gate Array (FPGA)

Firas.Gh.Tawfeeq [*1] ***Alaa M. Abdul-Hadi*** [2]

[1,2] Department of Computer Engineering University of Baghdad. Iraq.
[1]f.ghanim1205@coeng.uobaghdad.edu.iq , ORCID: 0000-0001-7160-8548
[2] alaa.m.abdulhadi@coeng.uobaghdad.edu.iq , ORCID: 0000-0001-5852-1771
[*]Corresponding author: f.ghanim1205@coeng.uobaghdad.edu.iq

Received 10/9/2019, Accepted 8/12/2019, Published 8/9/2020

Abstract:

The widespread use of the Internet of things (IoT) in different aspects of an individual's life like banking, wireless intelligent devices and smartphones has led to new security and performance challenges under restricted resources. The Elliptic Curve Digital Signature Algorithm (ECDSA) is the most suitable choice for the environments due to the smaller size of the encryption key and changeable security related parameters. However, major performance metrics such as area, power, latency and throughput are still customisable and based on the design requirements of the device.

The present paper puts forward an enhancement for the throughput performance metric by proposing a more efficient design for the hardware implementation of ECDSA. The design raised the throughput to 0.08207 Mbit/s, leading to an increase of 6.95% from the existing design. It also includes the design and implementation of the Universal Asynchronous Receiver Transmitter (UART) module. The present work is based on a 163-bit key-size over Koblitz curve k-163 and secure hash function SHA-1. A serial module for the underlying modular layer, high-speed architecture of Koblitz point addition and Koblitz point multiplication have been considered in this work, in addition to utilising the carry-save-multiplier, modular adder-subtractor and Extended Euclidean module for ECDSA protocols. All modules are designed using VHDL and implemented on the platform Virtex5 xc5vlx155t-3ff1738. Signature generation requires 0.55360ms, while its validation consumes 1.10947288ms. Thus, the total time required to complete both processes is equal to 1.66ms and the maximum frequency is approximately 83.477MHZ, consuming a power of 99mW with the efficiency approaching $3.39 * 10^{-6}$.

Key words: ECDSA, Koblitz-curve, Maximum-frequency, SHA1, UART.

Introduction

There is a dramatically increasing high demand to achieve security objectives due to the growing use of the Internet of things (IoT) in all aspects of daily life, such as commerce and bank transactions (1), (2). Embedded devices utilised in IoT face security and resource availability challenges (3), when using asymmetric encryption to satisfy the required security goals. Public key cryptography relies on a discrete logarithm problem (DLP) that is difficult to be solved (4), (5). Elliptic Curve Cryptography (ECC) was proposed to design an asymmetric cryptographic system and a mathematical model of ECC, based on the Elliptic Curve Discrete Logarithm Problem (EDLP) (6), was adopted. ECC provides the same security level delivered by traditional public key cryptosystem such as RSA (Rivest–Shamir–Aldeman), but with a smaller number of bits i.e., less consumption, smaller area and high speed in computing the encryption key (7).

Elliptic Curve Digital Signature Algorithm (ECDSA), a peer of Digital Signature Algorithm (DSA), is used to generate the signature for the

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

document and allow to verify its originality depending on the owner's private key used to sign it (8), (9). High-speed ECDSA, design and implementation are still a challenge for research studies and developers concerning the reconfigurable circuit environment such as the FPGA, which supplies this feature with flexibility and low cost. This work proposes a throughput enhancement for the hardware implementation of ECDSA over the Galois field GF(2^{163}). The proposed contribution uses partial intermediate registers to increase the throughput and design and implement UART in the signal chip.

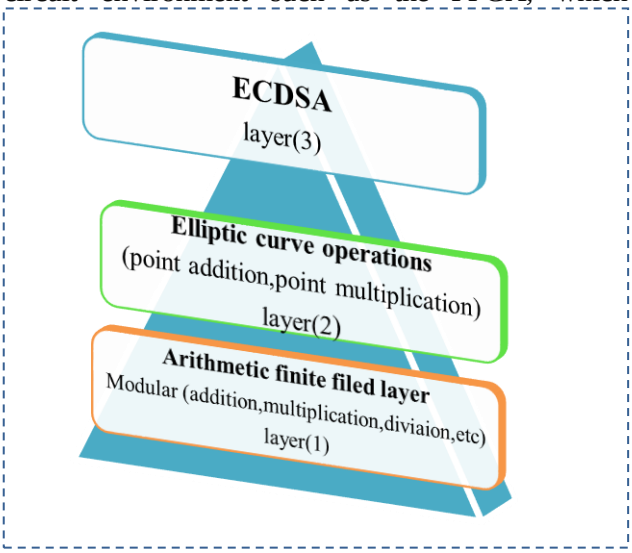

Figure 1. **Hierarchy of ECDSA layers**.

Fig. 1 shows the hierarchy of ECDSA layers (10). The first layer is called the arithmetic field layer, consisting of modular components such as addition, multiplication, division. Elliptic curve crypto-processor layer includes point addition, point doubling and point multiplication. Finally, dedicated components such as inversion and hashing exist to implement the top-most layer.

Related Work

Benselama ZA, Bencherif MA, Khorissi N and Bencherchali, MA (10) presented the implementation of ECDSA over finite field GF(2^{163}), using Virtex-5 platform. The implementation comprises SHA-1 and ECC with control. The satisfying total time of generation is approximately 1.58ms for signature generation, at maximum frequency of 207.097Mhz, with occupation slices 10838 for registers and 28998 for lookup table (LUT) slices. Validation requires 1.953ms with occupied slices 12922 for flip-flop and 54597 for LUT at maximum frequency 195.309Mhz. M Elhadjyoussef, W Benhadjyoussef and N Machhout (11) proposed an architecture, based on the Globally Asynchronous Locally Asynchronous (GALS) design methodology. It is implemented over GF(2^{163}), using the Virtex-6 platform. The proposed system consists of SHA-1, Random Number Generation (RNG), Elliptic curve crypto-processor and memory. The time taken for signature generation and validation was 3.844ms and the power consumption was approximately 127mW.

Bhanu Panjwani and Deval C Mehta (12) demonstrated a proposed ECDSA design with two implementations. The first requires 0.367ms with 11040 slices for signature generation and 0.393ms with 12846 slices for signature verification. Both processes work at a frequency operation of 100MHz. The second one takes 0.615ms with 8773 slices for signature generation and 0.672ms with 9967 slices for verification. Both implementations use the same frequency operation and Virex-5 platform.

Implementation of ECDSA as proposed by Symposium I, Anissa Sghaier, Medien Zeghid and Mohsen Machhout (13) is accomplished over Xilinx Virtex5 ML50 platform. The results show that smaller parameters are a good choice for the ECDSA processor with low storage and low power. Signature generation and validation process in this work is consumed at 1.5ms at maximum frequency of 107MHZ with consumption power approximately of 105.7mW.

Mathematical background

The elliptic curve can be defined over different types of infinite numbers like real numbers (R), complex numbers (C), integer numbers (Z) and Rational numbers (Q), but in the cryptography process, the curve should be defined over the finite field. Realisation of applying the elliptic curve over a finite field requires understanding following terms:

The Group

From mathematical perspective, a group is a set of numbers with binary operation that fulfills the following features:

1. Associativity for any x, y, z ∈ F, $x * (y * z) = (x * y) * z$.
2. The unity is unique c ∈ F, such that w ∈ G,$c * w = w * c = c$.

3. Inverse, for each number a ∈ F, there exists a unique number (a^{-1}), such that a*a1 =c, c=a^{-1}*a. For group F, if any x, y ∈ F have x*y=y*x, it is called abelian or commutative group; else it is called non-abelian or non-commutative group. When the group has only finite set of numbers, it is called finite group, else it is called an infinite group (14).

The Ring

The ring R is a set of numbers with two binary operations, addition and multiplication, achieving the following properties:
Albelian group structure.
Associativity of multiplication operation for x, y, z ∈ F, $x * (y * z) = (x * y) * z$. Compatibility of multiplication and addition operations. For any x, y, z ∈ K, $x * (y + z) = xy + xz, (x + y).z = xz + yz$. If the set of elements are commutative, it is called a commutative ring (15).

The Field

The field (F) is a set of numbers together with two operations (*, +) called Field, if it satisfies the following conditions:
1. (F, +) represent an abelian group with the identity number symbolised by 0.
2. (F, *) represent an abelian group with the identity number symbolised by 1. When F has a finite element within it, it is called a "Finite field". Order is the number of elements the finite field holds. Order is equal to (P^m), where p indicates the prime number (characteristic of the finite field) and (m) represents an integer with value greater than one (dimension). Such field is symbolised by F(p^m). Finite field is distinguished by its orders as shown below:
2.1. When m = 1, the finite field is called prime field and is denoted by F_p.
2.2. For m ≥ 2, finite fields are called Extension Fields. P equal to 2 is a special representative of extension fields denoted by ($F2^m$). Fig. 2 shows the Hierarchy of a finite field (16).

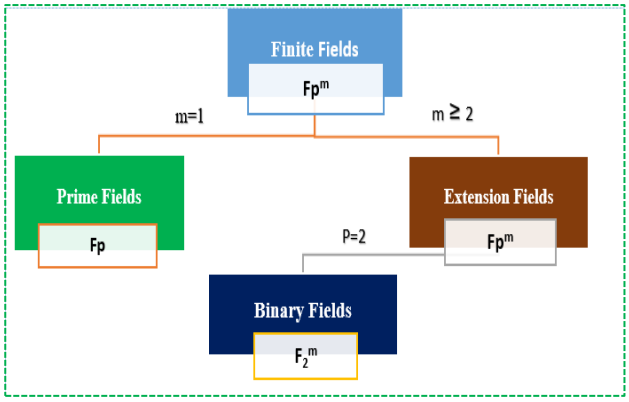

Figure 2. Hierarchy of finite field.

EEC and its arithmetic over Field GF(2^m)

This section provides brief mathematical information about an elliptic curve E. The equation of elliptic curve E over GF(2^m) is cubic with the following general form:

$$y^2 + xy = x^3 + c_1x^2 + c_2 \quad ..1 (17)(18).$$

Where c_1 and c_2 ∈ GF(2^m) and c_2. The curve is a collection of points P over coordinates (x, y). Possible operation points over the curve are point addition and point doubling. Suppose, points N and M are on the curve, where $N(x_1, y_1) \neq M(x_2, y_2)$ then point addition $V(x_3,y_3) = N(x_1,y_1) + M(x_2, y_2)$ and point doubling, when M=N is as below:

$$x_3 = \lambda^2 + \lambda + x_2 + x_1 + c_1 \quad \text{If N} \neq \text{M} \quad .. 2.$$
$$x_3 = \lambda^2 + \lambda + c_1 \quad \text{If N = M}$$
$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1 \quad .. 3.$$

Where $\lambda = \frac{y_2+y_1}{x_2+x_1}$ if N ≠ M (Point Addition)

$\lambda = x_1 + \frac{x_1}{y_1}$ if N= M (Point Multiplication)

There are different kinds of elliptic curves based on a and b values. This paper considers the Koblitz curve, defined over f_2^m. It has the same form of equation(1), where $c_1 \in (0,1)$ *and* $c_2 = 1$. The curve has an advantage depending on the algorithm similar to double and add algorithm (binary algorithm), allowed to replace point doubling by Frobenius endomorphism(∅).

$$\emptyset: (x,y) \rightarrow P(x^2,y^2) \;\&P(\infty) \rightarrow P(\infty) \quad .. 4.$$

Equation (4) shows the endomorphism for point $P(x, y)$ and point at infinity, $P(\infty)$ that needs only two squaring of x and y. It is complicated to replace a point doubling with endomorphism.
If the point multiplication is defined by equation (5)

$$Q = Kp = p + p + p \rightarrow k_{times} \quad .. 5.$$

Where k is a positive integer, it stands for all points on the curve. Manipulation is required on k representation, that is

$\mu\emptyset(p) - \emptyset^2(p) = 2p \;\; where\; \mu = (-1)^{1-a}$ .. 6.

Therefore, $\emptyset$ can be seen as a complex number,τ, that satisfies $\mu\tau - \tau^2 = 2$, implying $\tau = (\mu + \sqrt{7})/2$. To ensure fast Frobenius endomorphism, the value k should be given in the $\tau - adic$ representation, as shown in equation (7).

$K = \sum_{i=0}^{l-1} \; k_i \tau^i$ .. 7.

Representing a point (x, y) on the coordinates refers to affine coordinates, but the inversion process is expensive in it. Thus, Lopez Dahab (LD) projective coordinates are used in this paper, where a point (X, Y, Z) represents the affine point in equation (8).

$(x,y) = X/Z \;,\; Y/z^2$ .. 8.

More efficient point addition formula is available, when there is p_1 in LD and p_2 in affine coordinates. Formula for calculating $p_3(X_3,Y_3,Z_3) = (X_2,Y_2,Z_2) + (x_2,y_2)$ is shown as follows (19):

$$A = Y_1 + y_2 Z_1^2 \;; B = X_1 + X_2 Z_1$$
$$C = BZ_1 \;;\; Z_3 = C^2 \;\; ; D = X_2 Z_3$$
$$X_3 = A^2 + C(A + B^2 + c_1 C) \quad .. 9.$$
$$Y_3 = (D + X_3)(AC + Z_3) + (y_2 + x_2) z_3^2$$

Where c_1 is the parameter of the curve, see equation (1)

Applied algorithms:

Polynomial addition

The addition of two polynomials $a(x)$ and $b(x)$ in f_2^m is bitwise exclusive or (XOR).

Polynomial multiplication

Lest-significant-bit (LSB) multiplier is used at the finite level to implement the multiplier. Hence, the process of multiplication of b(x) coefficients in equation (10) begins from the left b_0 and the right b_{m-1} (19).

$C(x) = a(x)b(x) mod\, f(x)$ .. 10.

Algorithm LSB-first multiplier

$INPUT: a = (a_{m-1}, \cdots, a_1, a_0) \in F_2^m$.

$INPUT: b = (b_{m-1} \cdots . b_1, \; b_0) \in F_2^m$.

$Output : c = a.b$.

$1. Set\; c \leftarrow 0.$

$2. for\; i\; from\; 0\; to\; m-1\; do$

$2.1\; c \leftarrow c + \; b_i\; a.$

$2.2\; a \leftarrow leftshift(a) + \; a_{m-1}\; r$

$3. Return(c)$

Fig.3 shows the hardware implementation of LSB multiplier over f_2^m.

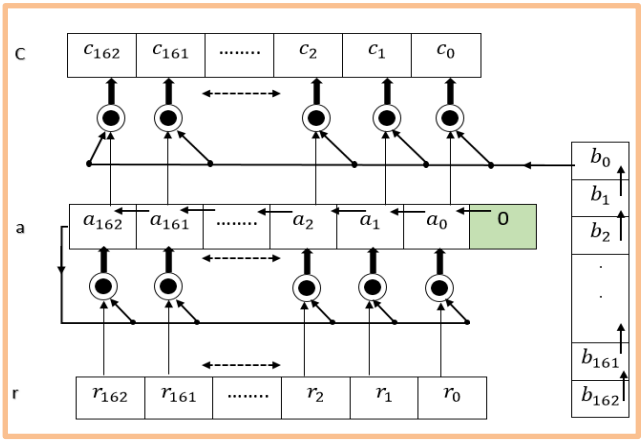

Figure 3. Least significant bit first(LSB) multiplier

Polynomial squaring

Polynomial squaring is considered as a linear operation, as it is faster than multiplication.

Binary representation of $a(z)^2$ in can be calculated by inserting zero bit between bits one after another binary representation of $a(z)$, as shown in fig .4.

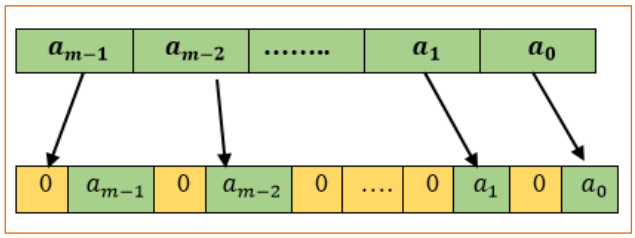

Figure 4. Squaring a binary polynomial $a(x)$.

Polynomial inversion and division

If $a(x)$ is polynomial and $f(x)$ represents an irreducible polynomial over $GF(2^m)$,since the degree of $a(x)$ < degree of $f(x)$, they are relatively prime. Extended Euclidean algorithm could be initiated with $a(x)$ and $f(x)$ and generate polynomials $s(x)$ with degree $t(x) < m$ and t(x) with degree $< m - 1$ and t(x); the polynomials satisfy the following condition:

$a(x)s(x) + f(x)t(x) = GCD(a(x),f(x))$ ..11.
As $f(x)$ represents an irreducible polynomial in $GF(2^m)$, $GCD\big(a(x),f(x)\big) = 1$. Hence $a(x)s(x) + f(x)t(x) = 1$ over finite field $GF(2^m)$ and $f(x)t(x)$=0. Then, the equation (11) could be simplified to $a(x)s(x) = 1$. The inverse element $a(x)^{-1}$ has polynomial representation $s(x)$. Therefore, EEA for inversion in binary field uses polynomial representation (20).

Algorithm binary polynomial division

Input: $\boldsymbol{a(x)\ of\ (m-1)\ degree}$
$\boldsymbol{f(x)\ of\ (m)\ degree}$

1: $a \leftarrow a(x), f \leftarrow f(x), r = 1\ , q = 1$
2: $While\ \big(f_{deg}\ \geq\ a_{deg}\ \big)do$
3: $a \leftarrow \ll (f_{deg} - a_{deg})$
4: $r \leftarrow a\ xor\ f$
5: $if\ \big(r_{deg}\ \geq\ a_{deg}\big)\ then$
6: $q \leftarrow \big(q \ll \big(f_{deg} - r_{deg}\big)\ \big) + 1$
7: $else$
8: $q \leftarrow \Big(\ q \ll \big(f_{deg} - \ a_{deg}\big)\Big)$
9: $end\ if$
10: $f \leftarrow r$
11: $end\ while$

Extended Euclidean Algorithm (EEA)

Input*: Polynomail* $a(x) of\ (m-1)\ degree$
$Polynomail\ f(x) of\ (m)\ degree$

Output: $a(x)^{-1}\ mod\ f(x)$
1: $a \leftarrow a(x), f \leftarrow f(x), 0, s \leftarrow 1$
2: $\boldsymbol{While}\ (r\ \neq 0)\ (\text{gcd}(\neq 0)\ \boldsymbol{do}$
3: $\boldsymbol{Perform\ PolyDivide\ to\ find\ r\ \&\ q}$
$(f = a\ q + r\ )$
4: $f \leftarrow a\ , a\ \leftarrow r$
5: $t\ \leftarrow s\ , s \leftarrow t - q\ s$
6: $\boldsymbol{end\ while}$

The loop in the binary division computes the remainder and quotient, when $f(x)$ is divided by $a(x)$. In line 3, the polynomial $a(x)$ is shifted by the amount of $f_{deg} - a_{deg}$ and uses the new values of $a(x)$ to do xor with $f(x)$ and obtain the remainder $r(x)$. In line 5, the degree $r(x)$ is checked whether it is greater than that of d. If the condition is true, the quotient polynomial $q(x)$ by amount of $f_{deg} - r_{deg}$ and add 1 to the last bit of q(x). If the degree of $r(x)$ is smaller than d_a, then the $q(x)$ is left shifted by the amount of $f_{deg} - a_{deg}$.

Algorithms for mode operation of ECDSA

.And then assign the value of $r(x)$ to $f(x)$ in line 10. This procedure is repeated until the degree of $f(x)$ is smaller than d_a.

Signature generation (Mode 0)

The first step to operate ECDSA is to initialise all necessary parameters. Values of coefficients a and b of domain parameters specify the shape of the elliptic curve, while G represents its generating point. The signature process starts by sending the message and zero value to the ECDSA, in order to get 160-bit message digest. The message composed of 48-bits is forwarded into a secure hash function SHA-1 process. After the hashing process, the message digest is padded to get 163-bit, directly before storing it in the register. The next step is to calculate the public key through a point-multiplication process over the curve to obtain r value, which represents the first part of the signature. In a parallel manner, the inverse (K-1) is also prepared using the inversion process to calculate the second part of the signature s. Finally, the generation of signature process delivers two values r and s.

Signature validation (Mode 1)

After the ECDSA component receives the values of the parameters of signature r, s and value one to determine the mode operation type, the validation process starts by calculating the digest of the message first, in addition to the padding process. Then the receiver uses his private key to decrypt the signature. Next, the sender uses his public key to make sure that the message is coming from the origin sender. Finally, the output represents a point on the curve with coordinates (X1, Y1). The value of X1 being equal to r implies that the signature is valid and coming from the origin sender. It is not valid when X1 is not equal to r. As a result of signature validation, the process ends.

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

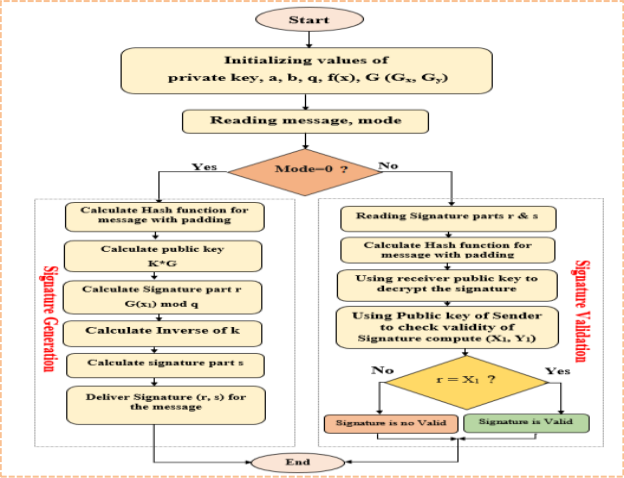

Figure 5. Flowchart of ECDSA operations.

ECDSA architecture for throughput improvement

Throughput of the proposed work is increased by using a number of intermediate registers used to save the data during internal processes. This work is based on Koblitz point addition and point multiplication. Since the point multiplication on the Koblitz curve can be calculated using classic squaring and point addition components, the technique of adding an intermediate register for storing operation results at each step should increase the throughput and reduce the number of clock cycles. Fig. 6 illustrates the proposed architecture of Koblitz point addition.

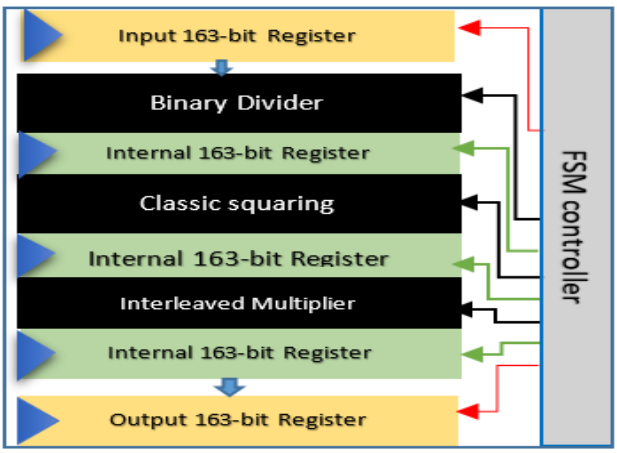

Figure 6. Proposed Koblitz Point addition.

The proposed ECDSA processor consists of two major components, UART and Core_unit. UART is constructed using three sub-modules; UART_receiver, UART_transmitter and Clock divider. UART_receiver gets the input data (message, mode, r and s) from the pc through the serial communication port R232 and stores the values inside serial-input-parallel-output-registers (SIPO) before starting any mode operations. ECDSA in its turn takes the input value and starts either the signature or validation process, based on the mode flag value. The output result is stored in parallel_input_serial_output (PISO) registers to be ready for transmission through UART_transmitter. The clock divider is responsible to generate a continuous signal for a baud rate of 9600bit/s. The total time required for sending 48byte (r, s, m, mode byte) is 49999ns with clock period of 10ns. ECDSA architecture consists of SHA-1, signature generation and validation components.

Fig. 7 shows internal components of UART multiplexer receiver and transmitter, with its intermediate registers and finite state machine controllers.

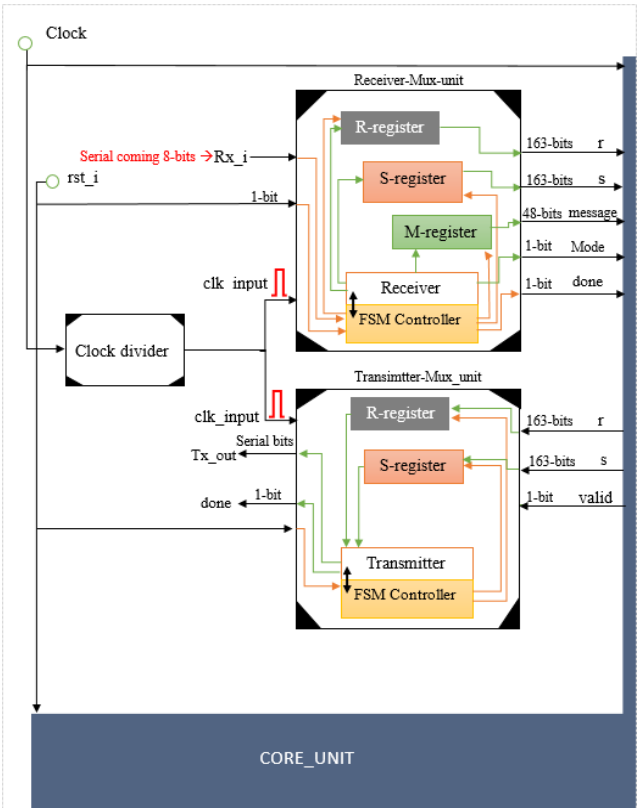

Figure 7. Proposed ECDSA architecture.

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

Simulation

Simulation of the proposed ECDSA processor was based on the digital signature specification FIPS 186-4. The sampling data published in 2013 by RFC for information purpose, ISSN: 2070-1721, is shown in table.1.

Simulation of signature process

Fig. 8 illustrates the result of the signature process. The generation process takes approximately 0.55360ms to end the task. Ready_out signal gets a value 1 to give identification for ending the process. The clock period is 10 ns and the frequency of operation is 100Mhz.

Simulation of validation process

Fig. 9 shows the result of the validation process. Time required for validation is approximately 1.10947289ms. Values r, s and the message should be initialised before starting to run the process. The value of validation control signal determines the validity of the signature. When it is one the signature is valid, else it is rejected. Mod_i signal is set to 1 to let the ECDSA processor work in validation mode.

Table 1. NIST recommendation Koblitz Parameters.

ECDSA, 163 Bits (Binary Field, Koblitz Curve) Curve: NIST K-163
q = 4000000000000000000020108A2E0CC0D99F8A5EF
Private key: x = 09A4D6792295A7F730FC3F2B49CBC0F62E862272F
Public key = x * G
Ux = 79AEE090DB05EC252D5CB4452F356BE198A4FF96F
Uy = 782E29634DDC9A31EF40386E896BAA18B53AFA5A3
Generation Point(Gx,Gy)
Gx: 2FE14C0537BBC11ACAA07D793DE4E6D5E5C94EEE8
Gy: 289080FB05D38FF58321F2E800536D538CCDAA3D9
Signatures: With SHA-1, message = "sample": 01110011011000010110110101110000011011000110010 1
k = 09744429FA741D12DE2BE8316E35E84DB9E5DF1CD
r = 30C45B80BA0E1406C4EFBBB7000D6DE4FA465D505
s = 38D87DF89493522FC4CD7DE1553BD9DBBA2123011

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

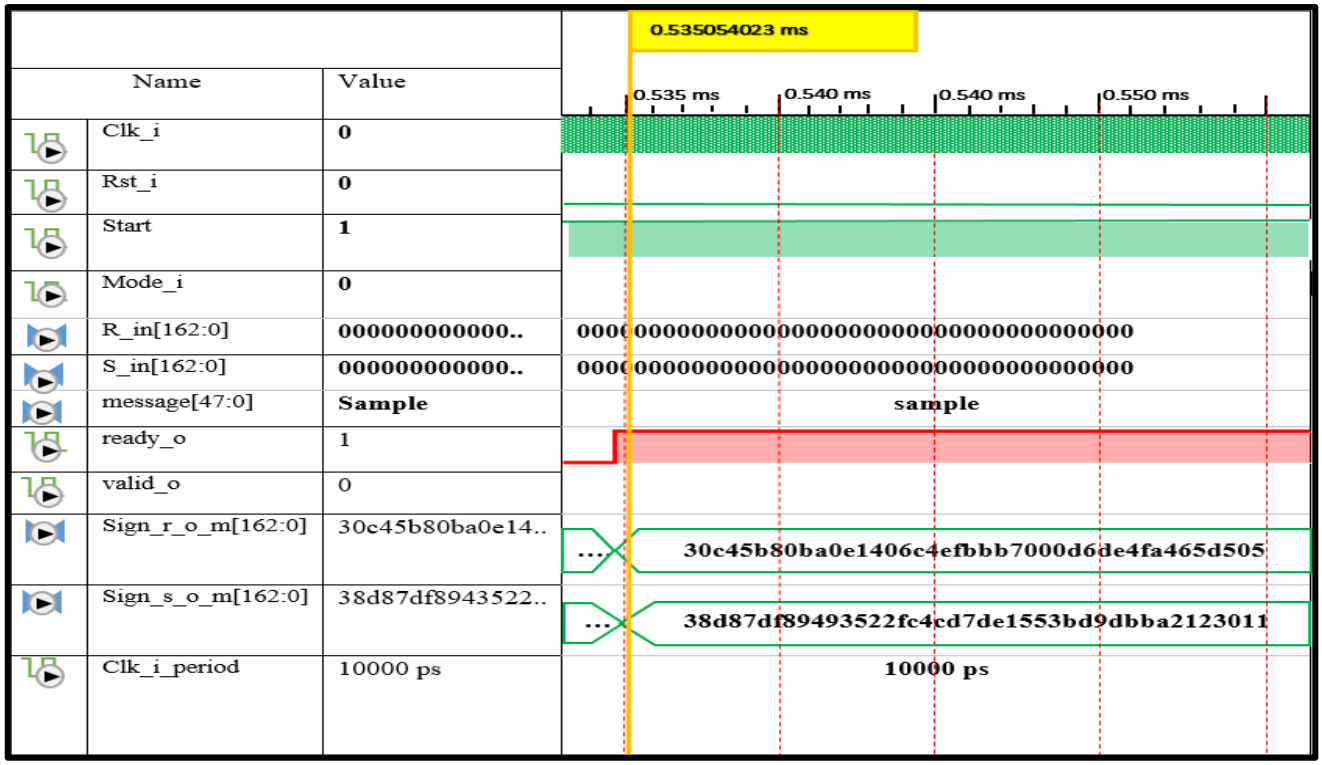

Figure 8. Simulation of signature generation.

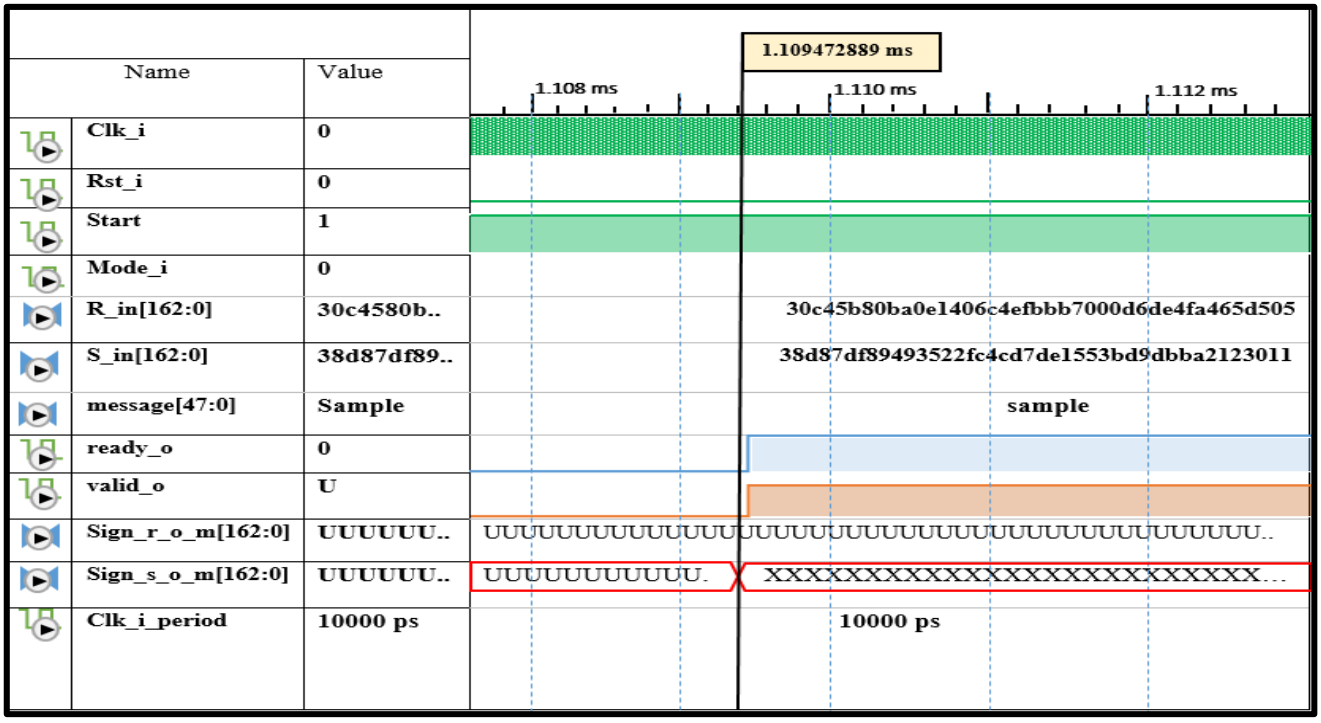

Figure 9. Simulation of signature validation.

Synthesis results

The proposed ECDSA processor implementation is more suitable for applications requiring low-bandwidth communication environment. It is based on three components: UART consisting of receiver, transmitter and clock-divider and cryptographic hash function SHA-1, in addition to ECDSA-Core-unit. ECDSA over Koblitz curve is based on Finite Field GF (2^{163}), using Virtex-5 xc5vlx155t-3ff1738 as a platform. The serial the component is designed to build, the underlying arithmetic finite field and Koblitz point addition and point multiplication comprise the cornerstone of ECC operations, while ECDSA operations depend on CAS_Multiplier, extended Euclidean Algorithm (EEA) and Adder-Subtractor. The execution time of signature generation and validation process is close to 1.66ms. The number of slices on it is approximately 24,132 shown in fig.10. 83.477MHZ is the maximum frequency after timing optimization process. Fig. 11 shows the implementation of the top-most component of ECDSA processor, while fig. 12 illustrates its Register Transfer Level (RTL).

Time and clock cycles

Clock cycles of signature and validation

Table 2 shows the number of clock cycles for signature and validation components. It clearly illustrates that point multiplication consumes more clock cycles, compared to other components.

Table 2. Clock cycles for signature and validation.

Component	Signature		Validation	
	Qty.	Clocks	Qty.	Clocks
Interleaved Multiplier	1	12060	1	11760
Divider	1	3287	1	3287
Squaring	1	1	1	1
XOR (bitwise)	1	1	1	1
Point Multiplication	1	52646	2	52646
CAS Multiplier	1	409	2	295
Inversion	1	3295	1	3295
SHA-1	1	3630	1	3630
Add-Sub	1	224	-	-

Slice logic utilisation

Fig. 10 shows the number of registers and LUT slices and its distribution required for the design of the proposed system.

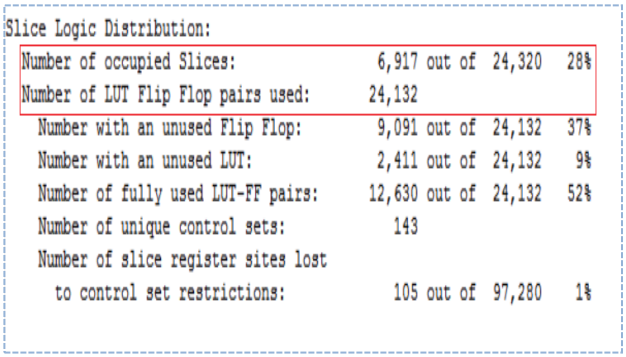

Figure 10. Number of Slices of the ECDSA design.

Throughput and efficiency

Obtained results show that the difference between the arrival input time and output is 0.311ns. The combinational path delay is 3.763ns, while the maximum frequency m is 83.477MHz. Throughput and efficiency values of the proposed ECDSA system can be computed based on equations mentioned in (21).

As signature generation requires 0.553604023ms and validation process needs 1.109472889ms, the entire ECDSA system needs 1.663072889ms or 1663072.889ns, implying that the number of clock cycles is equal to (1663073). Throughput can be calculated using equation (10).

$$\text{Throughput} = \frac{83 \cdot 477 * 163}{166307} = 0 \cdot 081817 \text{ Mbit/s} \quad ..10$$

In the same context, the efficiency of the ECDSA system can be computed by applying equation (11), as

$$\text{Efficiency} = \frac{0 \cdot 081817}{24132} = 3 \cdot 39 * 10^{-6} \quad .. 11$$

Table 3 and fig. 10 clearly show that time of signature generation and verification in this work, during the simulation process, is equal to 1. 66ms. When the results are compared with designs presented in (12) and (13), their time is less than the time of the proposed solution by 0.9m and 0.1ms respectively, while the difference in time between the suggested solution and designs presented in (10), (11) show that their design is greater than by 1.873ms and 2.184ms,

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

sequentially. The maximum frequency of this design is the smallest value among other previous designs.

Table 3. Time Comparison for ECDSA implementations using FPGA.

Design	Platform	Finite Field	Max-Freq. MHz	Area(Slice)	Time (Gen+Ver) ms
[10]	Virtex 5	GF(2^{163})	195.3099	23,760	3.533
[11]	Virtex 5	GF(2^{163})	148.963	20,628	3.844
[12]	Virtex 6	GF(2^{163})	100	Sign -23,886 Gen-27791	1.287
[13]	Virtex 5 150 M	GF(2^{163})	107.4	18504	1.5
This Work	Virtex 5 155T	GF(2^{163})	83.477	24,132	1.66

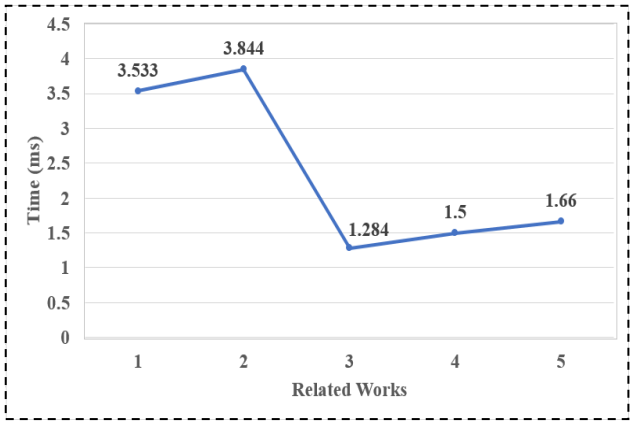

Figure 11. Relative time comparison for ECDSA implementation

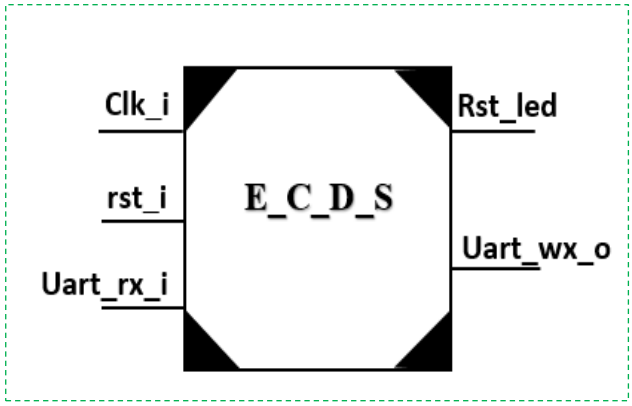

Figure 12. Main Component of ECDSA.

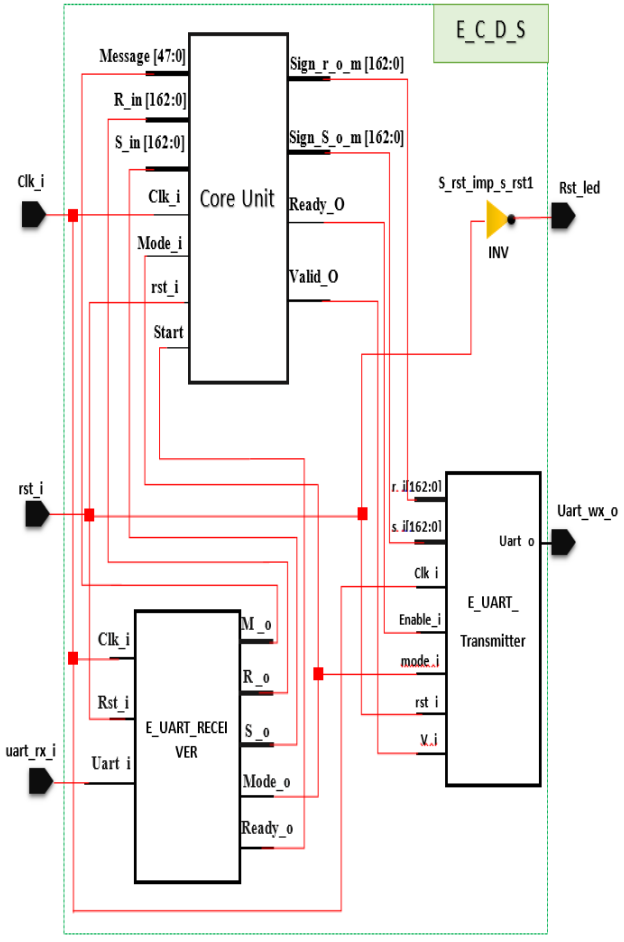

Figure 13. RTL Scheme of proposed ECDSA.

Conclusion

The paper presented improved throughput of ECDSA processor implementation for a binary finite field over Koblitz curve k-163, due to the importance of performance utilisation in a modern FGPA device. The number of clock cycles in work (13) was 167494. In the proposed work, intermediate registers store intermediate values between Koblitz point addition components, reducing the number of clock cycles for signature generation and validation to 166307, according to equation 6. Throughput is raised to 0.098011 Mbit/s, leading to an increase of 6.95% from the existing work, but the area increased from 18504 to 24,132 that also affects the consuming power. The design is based on a serial component for building the underlying arithmetic finite field and high-speed architecture using Koblitz point addition and point multiplication for ECC operations. Further, considering the architecture Virtex5-xc5vlx155t-3ff1738, the execution time of signature generation and validation process is approximately 1.66ms with a maximum frequency of 83.477MHz. UART design and implementation was included in the work, while it was not implemented in previous works.

Authors' declaration:

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are mine ours. Besides, the Figures and images, which are not mine ours, have been given the permission for re-publication attached with the manuscript.
- The author has signed an animal welfare statement.
- Ethical Clearance: The project was approved by the local ethical committee in University of Baghdad.

References

1. Riahi A, Challal Y, Natalizio E, Chtourou Z, Bouabdallah A. A systemic approach for IoT security. Proceedings of the IEEE Int Conf Distrib Comput Sens Syst DCoSS; 2013; pp. 351–5.
2. Siegel M, Jalali MS, Kaiser JP, Siegel M, Madnick S. The Internet of Things (IoT) Promises New Benefits – and Risks : A Systematic Analysis of Adoption Dynamics of IoT Products. 2017;(August).
3. Yousuf T, Mahmoud R, Aloul F, Zualkernan I. Internet of Things (IoT) Security: Current Status, Challenges and Countermeasures. Int J Inf Secur Res. 2016;5(4):608–16.
4. Koblitz N. An elliptic curve implementation of the finite field digital signature algorithm. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 1998;1462:327–37.
5. Oswald E, Fischlin M. Advances in cryptology – EUROCRYPT 2015 34th annual international conference on the theory and applications of cryptographic techniques Sofia, Bulgaria, april 26–30, 2015 proceedings, part I. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 2015;9056:129–55.
6. Gallagher PD, Romine C. FIPS PUB 186–4 Digital Signature Standard (DSS) Category: Computer Security. Sub-category: Cryptography. 2013 July. 186–4 pp. Available from: http://dx.doi.org/10.6028/NIST.FIPS.
7. Wander AS, Gura N, Eberle H, Gupta V, Shantz SC. Energy Analysis of Public-Key Cryptography on Small Wireless Devices. Pervasive Comput [Internet]. 2005;324–8. Available from: IEEE
8. Guneysu T, Paar C. Ultra high performance ECC over NIST primes on commercial FPGAs. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 2008;5154 LNCS:62–78.
9. Foundation F.Patent Application Publication (10) Pub . No: US 2019/0059733 A1 (43). 2019;1.
10. Benselama ZA, Bencherif MA, Khorissi N, Bencherchali MA. Low cost reconfigurable Elliptic Crypto-hardware. Proceedings of the IEEE/ACS Int Conf Comput Syst Appl AICCSA; 2014; pp.788–92.
11. Elhadjyoussef, W.;Benhadjyoussef, N.; Machhout M. Low Power Elliptic Curve Digital Signature Design for Constrained Devices. Int J Secur. 2012;6(2):1–14.
12. Panjwani B, Mehta DC. Hardware-software co-design of elliptic curve digital signature algorithm over binary fields. Int Conf Adv Comput Commun Informatics, ICACCI; 2015. pp. 1101–6.
13. Symposium I. Anissa Sghaier, Medien Zeghid, Mohsen Machhout University of Monastir, Faculty of Sciences, EfJELab, Monastir 5019, 2016;343–8.
14. Thangaraj A, Dihidar S, Calderbank AR, McLaughlin SW, Merolla JM. Applications of LDPC codes to the wiretap channel. IEEE Trans Inf Theory. 2007;53(8):2933–45.
15. Booker AR, Sijsling J, Sutherland A V., Voight J, Yasaki D. A database of genus-2 curves over the rational numbers. LMS J Comput Math. 2016;19(A):235–54.
16. Aung TM, Hla NN. Implementation of Elliptic Curve Arithmetic Operations for Prime Field and Binary Field using Java BigInteger Class. SSRN Electron J. 2018;
17. Johnson D, Menezes A, Vanstone S. The Elliptic Curve Digital Signature Algorithm Validation System (ECDSAVS). Certicom. 2004;
18. Imran M, Rashid M, Jafri AR, Kashif M. Throughput/area optimised pipelined architecture for elliptic curve crypto processor. IET Comput Digit Tech. 2019;1–8.

19. Hazmi I, Gebali F, Ibrahim A. High Speed and Low Area Complexity Extended Euclidean Inversion over Binary Fields. IEEE Trans Consum Electron. 2019; 4:1.
20. Sghaier A, Zeghid M, Massoud C, Mahchout M. Design And Implementation of Low Area/Power Elliptic Curve Digital Signature Hardware Core. Electronics. 2017; 6: 46.

تحسين معدل الإنتاجية لتنفيذ معالج التوقيع الالكتروني على المنحني الاهليجي من نوع كوبلتز باستخدام البوابات المنطقية القابلة للبرمجة

فراس غانم توفيق[1] ***علاء محمد عبد الهادي[2]***

[2،1] قسم هندسة الحاسبات، جامعة بغداد، بغداد، العراق.

الخلاصة:

يقدم البحث دراسة عن تصميم وتنفيذ دائرة الكترونية لتوليد التوقيع الالكتروني والتاكد من صحته ,بالاعتماد على مواصفات المنحني الاهليجي الموصى بها من قبل المعهد الوطني للمعايير والتكنولوجيا(NIST) .حيث أرتكز العمل على إختيار منحني كوبلتز وتطبيقه على الحقول المنتهية أو ما تسمى بحقول غالو($GF(2^{163})$، ونظراً لأهمية تحسين الأداء في المعالجات الحديثة المبنية في بيئة البوابات المنطقية القابلة للبرمجة (FPGA)، فقد أظهرت نتائج المحاكاة والتنفيذ للتصميم المقترح على الجهاز نوع Virtex5-xc5vlx155t-3ff1738 زيادة في معدل البيانات التي يتم معالجتها اثناء عمليتي توليد التوقيع واثبات صحته الى 0.08187 Mbit/s وبنسبة تصل الى 6.95% ,بالمقارنة مع التصميمات السابقة ، كما أستغرقت مدة تنفيذ العمليتين 1.66 ملي ثانية وبتردد أقصاه 83.477 ميكاهرتز. تم الاخذ بنظرالاعتبار تصميم المنفذ التسلسلي غير المتزامن (UART) والمستخدم في عملية نقل البيانات بين الحاسبة وFPGA .

الكلمات المفتاحية :التوقيع الالكتروني، المنفذ التسلسلي غير المتزامن، حقل غالو، منحني كوبلتز.