Improved ECC Performance Using NAF Algorithm for Binary Edward and Edward Elliptic Curves

Mohammad Alkhatib[1]

[1]College of Computer and Information Sciences Al Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia

Abstract
Improving the performance of elliptic curve crypto-systems (ECC) to satisfy modern cryptographic applications has become a pressing need. This research presents high-speed ECCs for Binary Edward and Edward curves using Non adjacent Form (NAF) algorithm.
Proposed ECCs use NAF algorithm to perform scaler multiplication, which is the main operation in encryption process. This contributed mainly in minimizing the time delay via reducing the number of point addition operations performed during scaler multiplication. Furthermore, Homogenous projective coordinates were used to avoid time consuming-modular inversion operation. Parallel design implementations were used to accelerate ECC computations and achieve the highest performance level.
Experimental results show that ECC implementations using NAF algorithm overcome corresponding implementations using other methods such as binary method for all possible design choices. A variety of parallel designs in addition to sequential design implementations were examined for both Edward and Binary Edward curves. The Binary Edward ECC using NAF algorithm and 7 parallel multipliers (PM) design accomplished the shortest time delay. So, it represents an attractive choice for security applications that require high-speed crypto-processor. Moreover, ECC sequential design using NAF algorithm scored better performance results in comparison with similar designs that use the binary method. Sequential design is a preferable choice for applications where available resources are limited. Presented ECC designs are implemented using VHDL, and synthesized using the Xilinx tool with target FPGA.
This article also presents performance analysis for the different design choices of the two types of elliptic curves using both NAF and binary algorithms. This helps in developing the most efficient ECC for different security applications.

Key words:
Elliptic curves cryptosystem, Time-consumption, Resources, NAF algorithm, Binary method, Projective Coordinates, Security applications.

1. Introduction

Elliptic Curve Crypto-system (ECC) has gained increasing interest, since it was introduced in 1985 by the scientists Miller and Koblitz. It has been widely used for many security applications such as encryption, digital signature, and key exchange.

In order to satisfy the variety of security requirements for elliptic curves applications, a number of ECC representations over GF(p) were developed in the last few years.

This research introduces a fast and secure ECC implementations using the non-adjacent form (NAF) method and parallel hardware designs. In particular, two famous representations of GF(p) elliptic curves were implemented, which are Edwards and Binary Edwards curves. The study focuses on improving the performance of scaler multiplication operation, which is the key operation in ECC encryption.

The proposed ECC implementations use the projective coordinates to avoid the time-consuming division operation in the scaler multiplication's computations, by replacing it with a number of multiplication operations.

In addition, parallel hardware components; mainly multipliers and adders were used to implement ECC computations in parallel, and thus reduce the time consuming and increase the security against simple time/power attack (STA). Within this context, this research provides several ECC implementations using variable degrees of parallelism to investigate the performance and resources-consumption trade-off at each parallelization degree and help in determining the most efficient ECC design for particular security application.

At the last stage of this research, the NAF method, with digits {-1,0,1}, was used to implement EC scaler multiplication and obtain higher performance level, compared to regular implementations using the binary method.

Results showed that the proposed ECC implementations achieved higher performance level. The speed of the encryption process's computations, represented by the scaler multiplication operation, is considerably improved. Such ECC designs and implementations are highly recommended for security applications that need a high-speed ECC.

The remaining parts of this article are the background and related works, ECCs computations and architectures, Results and analysis, and conclusion.

Manuscript received June 5, 2019
Manuscript revised June 20, 2019

2. Background and Related Work

ECC was first proposed by Miller and Koblitz as a reliable and efficient public-key cryptosystem. Its security level depends on the difficulty of finding a solution for the known discrete logarithm problem for elliptic curves. ECC encrypts a message by converting it into a point on elliptic curve, then applying the scaler multiplication operation on that point to yield another point laying on the elliptic curve. The point obtained after the scaler multiplication represents the encrypted message, which can be converted into a ciphertext as well [1,2].

The most important property of ECC that made researchers prefer using it over other types of asymmetric ciphers for different security applications is the ECC's ability to achieve comparable level of security using much smaller key sizes. This means that ECC can perform regular security operations such as encryption and decryption with higher performance level or speed, as well as, consuming less resources especially in hardware implementations [1, 3].

ECC uses modular arithmetic operations over finite fields to perform those operations accurately. The main finite field in elliptic curve cryptography are the binary field (2m) and the prime field GF(p). The scaler multiplication is the main ECC operation in the encryption process. It consists of two basic operations; point doubling and point addition, which are implemented multiple times to obtain the ciphertext [1, 4].

Point addition and point doubling are widely known as high-level computations for ECC. Each point doubling and addition includes a number of modular multiplication, modular division, and modular addition operations. These operations are known as low-level computations for ECC [4].

The modular division operation is the most time-consuming operation since it requires finding the multiplicative inverse [1]. Researchers proposed using projective coordinates systems to avoid the division by converting it to a number of consecutive multiplication operations. The main projection systems used for ECC are homogenous, López-Dahab, and Jacobean coordinates systems [5, 6].

Several elliptic curves families or representations were developed in previous researches. Among the most significant representations are binary Edward and Edward curves, which have inherent resistance against some side-channel attacks [3-1].

This study aims to speed-up the encryption process for both projective Edward and Binary Edward ECCs. The performance level of ECC encryption depends heavily on the speed of performing the modular arithmetic operations needed for scaler multiplication and its two main operations; point addition and doubling.

According to previous research works, using projective coordinates to implement ECC computations contributes in enhancing the performance level of ECC encryption. However, the use of projective coordinates costs additional multiplication operations. The multiplication is the second most time-consuming operations in elliptic curve cryptography. Therefore, the enhancement achieved, on the performance, by using projective coordinates is still limited if not combined with other factors [5-7].

Studies in this field showed that the homogenous projection system (X/Z, Y/Z) achieved the best performance results when implemented with many GF(p) elliptic curve representations; including Edwards and Binary Edwards, which will be studied in this research [12-21].

It is worth mentioning that there are two types of ECC implementations: software and hardware implementations. It has been found that hardware implementations are more fast and secure since they provide a dedicated environment for ECC operations. Moreover, the security of the crypto-system in software implementations will be affected by threats related to the operating system environment [4, 8-12]. Thus, the current research work focuses on the hardware implementations for ECC over GF (p).

The major hardware components used in hardware implementations for ECC are the multipliers and adders. Several kinds of hardware components were presented in previous literature. Those components have different performance levels and costs [1, 4, 8-12].

Researchers tried to accelerate the speed of ECC operations using high-speed multipliers such as Montgomery multiplier used in [8]. The proposed hardware implementation used sequential design. The performance level achieved for the scalar multiplication operation is estimated by 3 m sec.

authors in [24] developed Weierstrass ECC design that can support both prime and binary fields. However, the results obtained from proposed ECC showed that it needs extra memory resources.

Another research work [25] developed hardware implementations for Weierstrass ECC over GF(2n). The introduced high-speed ECC is suitable for smart card implementations. In [26], researchers proposed hardware implementations for Weierstrass ECC, which uses projective coordinates. In particular, the López-Dahab projective coordinates were used to avoid the costly inversion operation.

The hardware implementations for ECC presented in [7-9, 24-28] used the known sequential design, in which ECC computations are implemented using one multiplier and one adder components.

In despite of the fact that such implementations need less resources, they consume extra time to perform the encryption process. Therefore, they are considered not suitable for security applications that require fast ECC.

Another efficient approach to implement ECC operation is to use parallel hardware designs, in which ECC computations are implemented in parallel using parallel multipliers and adders components. The parallel ECC design utilizes the inherent parallelism in both point doubling and point addition operations. The use of parallel hardware designs to perform ECC encryption was proposed in many research articles [10, 13-23]. Almost all previous works has found that using parallel designs plays crucial role in improving the performance of ECC operations.

One more advantage of using parallel ECC design is to strengthen the security of the cryptosystem against the simple time attack (STA). Moreover, using parallel designs showed much better area×time-consumption2 (AT2) results [13-23].

In [10,14] researchers proposed parallel hardware designs for Weierstrass ECC over GF(p) and GF(2n) respectively. Both homogenous, and Jacobean projective coordinates systems were utilized to avoid division operation. The best performance was achieved using 4 parallel multiplier (4-PM) design when implemented with homogenous projection. It should be mentioned her that the performance level is usually estimated using the number of sequential multiplication (SM) levels needed to implement point addition and doubling operations.

Another design for Weierstrass ECC over GF(2n) was presented in [15]. The proposed design used 4-PM with Jacobean coordinates, and provided a trade-off between performance and power consumption. However, experiments showed that the design has low system utilization level, which is considered essential factor for efficient ECC designs [10].

The majority of the ECC designs introduced in previous research works [10,12,14, 15] uses the standard elliptic curve representation, which is the Weierstrass curve over GF(2n), while there are many elliptic curve representations that have not been sufficiently investigated. Moreover, the proposed parallel ECC designs consume extra resources and area. Therefore, they are considered costly and not efficient for applications with limited resources.

To avoid or mitigate the problem of resources consumption, some research works introduced ECC design schemas with different degrees of parallelism for elliptic curve computations. Those designs showed significant trade-off between time and resources consumption levels. The aim of such researches is to provide variety of ECC design choices that could be suitable for several security applications according to the required performance level and available resources. Each ECC design is supported with comprehensive analysis for the performance and resources-consumption levels [17-23].

Researchers in [17] introduced parallel design schemas for the Weierstrass ECC point addition over GF(p). Variable degrees of parallel hardware designs were implemented. Experiments showed that the 4-PM design achieved the shortest time delay, which is estimated by 4 SMs for each point addition. Other research work [19] studied the use of parallel designs and projective coordinates to improve the performance of point doubling operation. Again, the 4-PM design reported the highest performance level for point doubling. Both researches found that Weierstrass ECC always obtained the best performance when implemented using homogenous coordinates system.

It can be noticed from reviewing related literature that few researches investigated the parallel implementation of other types of elliptic curves. In [22] authors studied the characteristics of Montgomery elliptic curve over GF (p) when implemented using projective coordinates and parallel hardware designs. Several design choices were introduced. Researchers showed that the 2-PM design achieved the best trade-off between area and speed, where the 4-PM ECC obtained the best performance results.

Similar research works studied the performance and resources-consumption features of Tripling Oriented elliptic curve when implemented using variable parallelization levels [23].

In [20] and [21] researchers implemented Binary Edward and Edward elliptic curves respectively using variable hardware designs. Researchers used the same methodology to analyze the performance and resources-consumption characteristics for the proposed ECC designs. Experimental results proved that the 5-PM design obtained the highest performance. On the other side, the best performance for Binary Edward elliptic curve were achieved using 7-PM designs. Both kinds of elliptic curves registered the best performance when implemented using Homogenous projection.

However, the majority of the studies that investigated the implementation of parallel hardware designs for the different elliptic curve representations over GF (p) used the known binary method to perform scaler multiplication operation. Recent research works showed that using NAF method to implement scaler multiplication can speed-up the encryption process considerable. Authors in [29] found that implementing Montgomery elliptic curve scaler multiplication using NAF and parallel design achieves higher performance level for the encryption process, compared to the usual binary method.

In [29], the use of NAF algorithm with both Weirstrass and Montgomery ECCs were investigated. Implementation results illustrated that performance level for both elliptic curve representations was improved when implemented using NAF in comparison with binary method.

The current research work studies the performance and resources-consumption characteristics for two of most secure ECCs, which are Edward and Binary Edward over GF(p) using NAF method and parallel hardware designs. Homogenous projective coordinates system will be used to implement elliptic curve point operations. Different ECC

design choices will be introduced as well, in order to provide useful trade-off between performance and resources.
The next section discusses the computations, and designs for elliptic curve point doubling and addition operations, which represent the main building blocks for the scaler multiplication operation.

3. Computations and Architectures

This section presents computations and equations related to ECC point operations using projective coordinates, as well as the hardware design schemas to implement these operations. This section focuses on the ECC designs that achieve the best performance results for both Edwards and Binary Edwards curves over GF(p).

A. ECC Point Computations Using Projective Coordinates

The computations for ECC point addition and doubling operations are almost similar. The key difference between these computations is the way of calculating the slope (m). In point doubling, the slope is computed via deriving the elliptic curve equation where in point addition, the slope is calculated by dividing the difference between y coordinates on the difference between x coordinates for two points on elliptic curve. Equations (1) and (2) show the calculation of slope (m) for point doubling and addition respectively.
In point doubling, the slope (m) =

$$\frac{dy}{dx} = \frac{3[x^2+2a(x+1)]}{(2y)} \qquad (1)$$

In point addition, the slope (m) =

$$(y_2 - y_1)/(x_2 - x_1) \qquad (2)$$

It can be noticed that computation of slope (m) for point doubling depends on the equation for particular elliptic curve form, so it varies according to the type of elliptic curve. The point addition computation, on the other hand, is similar for all types of elliptic curve.
Projective coordinates system is used to avoid the time-consuming division operation in ECC computations and hence speed-up its operations. The current study implements ECC point operations using the homogeneous coordinates (x/z, y/z) system since it achieves less computations complexity than other projections when applied with both Edwards, and Binary Edwards curves over GF (p). The reader may refer to [17-21] for further details about projective elliptic curve computations for both Edwards and Binary Edwards forms.
Point addition and doubling operations between two points result in third point (x3, y3) such that: P3(x3, y3), where:

$$x_3 = m^2 - x_1 - x_2 \qquad (3),$$

$$y_3 = m(x_1 - x_3) - y_1 \qquad (4)$$

This research studies Edwards, and Binary Edwards curves over GF(p), which are represented by equations (5) and (6) respectively, as follows:

$$X^2 + Y^2 = 1 + d\,X^2\,Y^2 \qquad (5)$$

where a and b belong to GF (p) and 4a2 + 27 and b ≠ 0.

$$d_1(x + y)+ d_2(x^2 + y^2) = xy + xy(x + y)+ x^2y^2 \qquad (6)$$

where $d_1 \neq 0$ and $d2 \neq d_1^2 + d_1$.
Parallel hardware designs are used in this study to improve the speed of each point addition and point doubling computations as illustrated in the next section. Unlike previous studies, this research implements the scaler multiplication operation for Edwards and Binary Edwards ECC using NAF method in order to improve the performance of the encryption process even further.
This algorithm assumes that the point addition operation happens a one third times (based on the key size) compared to the point doubling operation [1-4].
The next section illustrates the parallel hardware designs that are used to implement the computational schemas for ECCs.

B. ECC Modeling and Architectures

The hardware architectures required to implement ECC point doubling and addition are presented in this section. Proposed ECCs are described and analyzed in order to clarify the improvements that can be achieved in respect to the encryption process's performance and security. Potential drawbacks for these ECCs are discussed as well.
The main hardware components used to implement ECC computations are multipliers (M) and adders (A). They are responsible for performing multiplication and addition operations respectively.
Previous researches [17 - 21] found that the highest performance for Edwards encryption can be accomplished by using a crypto processor with five parallel multipliers (PM). Note that in the hardware design for ECC the focus is usually given to the multipliers, since they consume more time to implement ECC computations, and require more cost in comparison to other components such as adders. The proposed Edwards ECC's architecture is depicted in figure 1. This architecture is used to implement both point doubling and addition operations. Similarly, the Binary Edwards ECC's architecture is presented in figure 2. Note that due to the level of computations complexity for Binary Edwards point doubling operation, seven parallel multipliers are required to implement each operation.

It should be mentioned her that each point doubling and addition operation is performed by calculating the values of x3 and y3 in equations (1) and (2). Note that equations (1) and (2) requires calculating the slope (m). In point doubling operation (3) the slope (m) is computed using equation (1), where in point addition it is computed using equation (2). The reader can find more details regarding ECC point operations' computations in [21-24, 29].

The current research work provides hardware implementations for the ECC architectures that accomplish the best performance levels for both Edwards and Binary Edwards elliptic curves over GF(p). These architectures are shown in figures 1, and 2. In addition, this work implements the scaler multiplication operation for both curves using NAF algorithm to improve the performance of the encryption process. A comparison between proposed ECC implementations and previous implementations that use the binary method is presented as well.

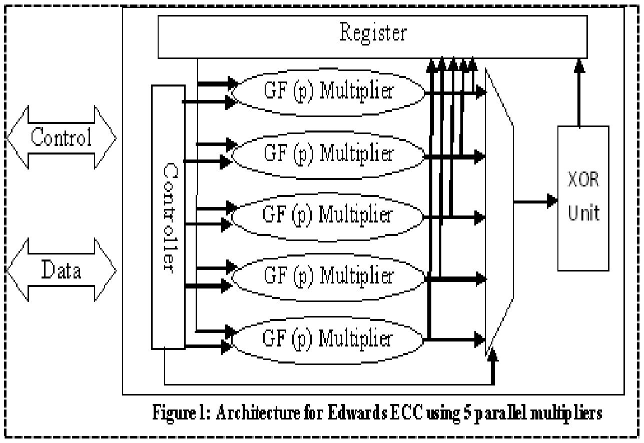

Fig. 1 The 5-PM architecture for Edwards ECC

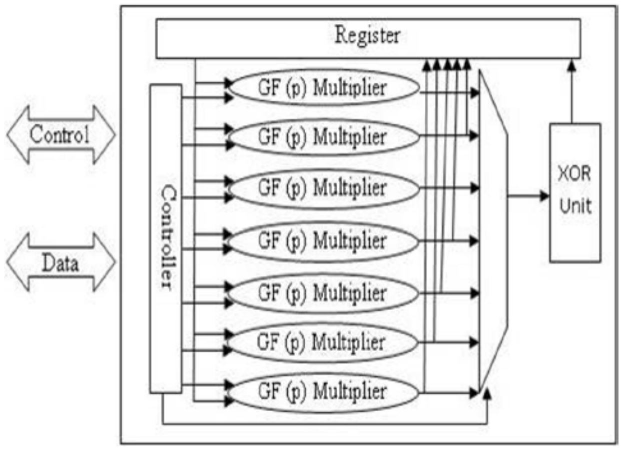

Fig. 2 The 7-PM architecture for Binary Edwards ECC

In order to have a comprehensive understanding for improvement, that can be achieved using NAF algorithm, on Edwards and Binary Edwards ECCs performance, both curves were implemented using varying degrees of parallelism as well as sequential design. In each parallelization level, the performance of ECC was observed. A comparison between NAF and Binary methods implementation were performed for each design as well.

The implementation model, which refers to the main stages followed to achieve high-speed implementation for Edwards and Binary Edwards ECCs, is depicted in figure 3.

It can be noticed that the implementation process start with the use of projective coordinates to calculate each point doubling and point addition operation instead of using the usual affine (x, y) coordinates. In particular, the homogenous projection was found to be the best choice since it has less computational complexity for ECC point computations. In comparison with other projections, homogenous coordinates system requires less number of modulus multiplications. Moreover, the use of projective coordinates saves the time needed to perform the costly modulus division operation, which includes finding the multiplicative inverse.

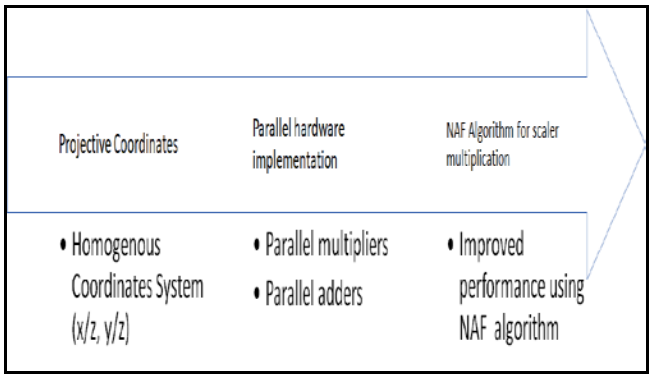
Projective Coordinates
Parallel hardware implementation
NAF Algorithm for scaler multiplication
• Homogenous Coordinates System (x/z, y/z)
• Parallel multipliers
• Parallel adders
• Improved performance using NAF algorithm

Fig. 3 Implementation model for proposed ECCs.

In the next stage of the implementation model, parallel hardware architectures were used to develop elliptic curve crypto processors. Researchers found that the best performance level can be accomplished by using architectures with 7-PM and 5-PM for Binary Edwards and Edwards ECCs respectively.

The use of parallel hardware design to implement ECC operations is crucial factor in improving the speed of the encryption process. Also, it contributes in enhancing the crypto-system immunity against the knows simple power/time attack.

At the final stage of the implementation model, the NAF algorithm was used to perform the scaler multiplication operation. NAF algorithm plays key role in speeding up

ECC encryption. unlike usual implementations using the Binary method, which assumes that point addition happens half times compared to point doubling, the NAF algorithm's implementation requires performing point addition one third times compared to point doubling operation. This represents a considerable improvement since it reduces the time consumed by the scaler multiplication operation, and thus the encryption process.

The next subsection describes the hardware environment used to implement proposed ECC architectures and achieve the performance results.

C. Implementation Environment

This research work uses the popular hardware description language VHDL to implement proposed designs. The code written in VHDL was then simulated using the ModelSim tool for validation purposes.

Researchers used the standard carry save multiplier and carry save adder to perform field operations represented by modular multiplication and addition operations.

In order to simulate and synthesize ECC designs and obtain performance and resources-consumption results, this study uses the Xilinx tool with the target FPGA (Field Programmable Gate Array) chip family chosen to be virtex5 (XC5VLX30).

The next section presents and analyzes the implementation results for proposed ECCs. Results are categorized into theoretical and experimental results.

4. Results and Analysis

This section mainly presents and discusses results and outcomes of the current study. The results are categorized into theoretical results, which are obtained by studying and observing the ECC designs, and implementation results, which obtained from simulating proposed ECCs designs using the Xilinx tool. Proposed ECCs are then evaluated and compared in terms of performance and resources-consumption.

A. Theoretical Results

The current research focuses on two significant factors of ECC designs. In particular, the performance and resources consumption. In the theoretical study, the performance level can be estimated by counting the number of multiplication cycles/levels for elliptic curve point operations. The multiplication level contains a number of modular multiplication operations, which can be performed in parallel at each multiplication level. For example, in the 5-PM design, a maximum of five multiplication operations might be performed in parallel at each multiplication level. The time consumed by addition operations is usually neglected in comparison with the time of multiplication operations.

The resources consumption level, on the other hand, can be theoretically estimated by observing the number of used multipliers and adders hardware components to implement ECC. In the 5-PM design, five multipliers and two adders are needed to design and implement the crypto-system.

In addition to studying the ECC designs that accomplish the highest performance level, this research also studies and implements all possible design choices, including the sequential design, for both Edwards and Binary Edwards curves over GF(p). All proposed ECCs are implemented using NAF algorithm to speed up the encryption process.

Table1 shows a comparison between all possible design choices for the GF (p) Edward ECC using NAF algorithm. It should be mentioned here that elliptic curve point operations were implemented using homogenous projective coordinates to avoid the need for finding the multiplicative inverse; a field arithmetic operation that consumes the major time and processing power in comparison with other field arithmetic for elliptic curve point operations. By using projective coordinates system, each point addition and point multiplication operation is implemented by executing a number of consecutive field multiplications and additions. For more details about Edward ECC computations using projective coordinates, the reader may refer to a previous research published in [21].

This research implemented ECC computations using both sequential and parallel hardware designs. In the sequential design, only one field arithmetic multiplication/addition is implemented at each level, while in the parallel designs a number of multiplications/ additions are performed in parallel. Such parallel implementation is allowed because of the inherited parallelism in ECC computations. Authors studied the point calculations after using projective coordinates and implemented these computations in parallel using designs with different degrees of parallelism. Empirical study showed that the 5-PM design represents the maximum degree of parallelism and achieves the shortest time delay for Edward curve. On the other side, parallel designs consume more resources in terms of the number of required multipliers and adders to perform ECC point operations as shown in table 1.

The table presents theoretical results obtained by observing ECC designs. The time delay for each point doubling or addition is estimated by the number of sequential multiplication (SM) cycles consumed by each ECC design. In each cycle a number of modular multiplications are performed in parallel. The overall time delay for point addition and point doubling operation is computed by using the following formula [23]:

Time-delay = No. SM $_{\text{(point doubling)}}$ + 1/3 × No. SM $_{\text{(point addition)}}$.

Practical experiments showed that point addition happens one-third times when implementing scaler multiplication operation using NAF algorithms. This contributes significantly in improving the performance level in comparison with implementations that use the binary method for performing scaler multiplication operation.
In addition to the performance (time delay) results, table 1 presents the cost factors for each ECC design. Cost factors, which are Area × Time (AT) and Area × Time^2 (AT^2) were widely used to estimate the efficiency of ECC designs [15-23]. Note that AT factor gives similar priority for area and time-consumption where AT^2 factor focuses more on the time-consumption level for ECC design.
Table 2 presents similar results for Binary Edward ECC using NAF algorithm. Performance and cost results in both tables are obtained using same formulas.
It can be seen from performance results in table 1 that 5-PM design for Edward curve obtained the shortest time delay, estimated by 6.6 SMs. As expected, this design achieved the best AT^2 results because it consumes less time compared to other designs. Therefore, the design is perfect for security applications where a high-speed crypto-processor is considered essential component. However, the 5-PM design requires more resources. As a result, it reported the worst AT results. Thus, such design is not recommended for applications with limited resources.
The sequential design requires less resources than other proposed designs, but consumes more time. Other designs show useful trade-off between area and performance. The nature of particular security application and the required performance and available resources determine the most suitable design choice to be implemented.
Results in table 2 shows that the highest performance level, which is 5.6 SMs, for Binary Edward curve can be achieved using the 7-PM design. This design represents the maximum parallelization level for this curve. Moreover, the use of NAF algorithm to perform the scaler multiplication operation has improved the speed of Binary Edward ECC considerable in comparison with implementations that use the usual Binary method for Scaler multiplication operation.
Other proposed designs present a trade-off between area and speed. Again, the sequential design requires few resources and need more SMs in comparison with other designs that use parallel hardware components.
In general, it seems that the Binary Edward ECC can achieve higher performance when implemented using the maximum possible parallelization level (7-PM) compared to Edward ECC with using the maximum parallelization of 5-PM.
Results in tables 1 and 2 are theoretical estimated results obtained by studying proposed ECC designs and analyzing their features. In the following section, the implementation results for proposed designs are presented. It presents the actual performance for each design in mille second as well as the resources-consumption in terms of hardware components required for implementing the ECC design.

B. Implementation Results

The performance level is the most important factor to be considered when evaluating the efficiency of ECC. It is usually assessed through the time required to perform the scalar multiplication (T_{KP}). The time of T_{KP} is computed through performing the following steps of calculations:

Step 1: Calculate the time required to perform one multiplication (T_M) operation. This can be done by using the following equation:

$$\mathbf{T_M = (cycles/bit) \times m \times clockperiod} \quad (7)$$

where **m** is the key size. The current experimental study uses a key size of 256 bits.
Using equation (7), the time of one multiplication operation can be computed as following:
T_M= 1 × 256 × 9.079 = 2324.224 nano sec (n sec)

Note that the time-consumption of one multiplication T_M is equivalent to the time-consumption of one complete sequential multiplication (T_{SM}) level, regardless the number of parallel multiplication operations performed in that level. This represents an important feature of the parallel ECC design that plays crucial role in accelerating elliptic curve computations.
Step 2: Calculate the time needed to perform each elliptic curve point doubling and point addition operation. These point operations are the main building block of the scaler multiplication operation.
For example, the 7-PM Binary Edward ECC design requires performing 4 SMs for each point doubling and 5 SMs for each point addition. Therefore, the time consumed by one point addition (T_{ADD}) and one point doubling (T_{DBLE}) can be computed as following:
T_{ADD}= 5*T_{SM}= 11621.12 n sec, T_{DBLE}= 4*T_{SM}= 9296.896 n sec

Step 3: Calculate the time-consumption of one inversion operation (T_{INV}). To convert the elliptic curve point coordinates back to the affine coordinates system an inversion operation is performed at the end of the scalar multiplication. It is known that the time of one inversion (T_{INV}) is equivalent to the time of three SMs [20-24]. Hence, the T_{INV} can be calculated as following:
T_{INV}= 3 * T_{SM}= 6972.672 n sec

Step 4: Calculate the total time-consumption of scaler multiplication (T_{KP}) operation. Unlike the majority of previous research works, the current research uses the NAF algorithm to perform the scaler multiplication instead of the usual binary method. The NAF algorithm can speed-up ECC computations because it requires performing less number of point addition operations compared to the binary algorithm. The total time-consumption for scalar multiplication, can be computed by using the following equation:

$$\mathbf{T_{KP} = ((0.3 \times 256 \times T_{ADD}) + (256 \times T_{DBLE}) + T_{INV}) \times 10^{-6}} \quad \textbf{(8)}$$

Note that the final result of the equation (8) is multiplied by 10^{-6} to convert the final value from nano to mille seconds.

Using the above equation, the total performance of the proposed 7-PM ECC design for Binary Edward ECC over GF(p) can be calculated as following:

$$\mathbf{T_{KP} = ((0.3 \times 256 \times 11621.12) + (256 \times 9296.896) + 6972.672) \times 10^{-6} = 3.279 \text{ m sec.}}$$

Note that the time-consumption of the 7-PM Binary Edward ECC using binary method equals 3.874 m sec [20]. Thus, the use of NAF algorithm has increased the speed of the scaler multiplication operation and consequently accelerated the encryption process calculations.

Table 3 shows the performance results of Binary Edward ECCs using both NAF and binary algorithms. ECCs were implemented using sequential and parallel designs. The performance level is measured by the time-consumption of the scaler multiplication operation (T_{KP}). The results in tables 3 and 4 were obtained through the Xilinx tool and represented in mille seconds.

The highest performance is accomplished using the NAF algorithm with the 7-PM design. It can be seen that the NAF algorithm performs the scaler multiplication operation in 3.279 m sec. This represents a considerable improvement in comparison with the 3.874 m sec time-consumption registered by previous implementations using the binary method. It can be seen from the table that the use of NAF algorithm has shortened the time delay for all ECC implementations including the sequential design. Note that the resources-consumption level is comparable for NAF and Binary method implementations using the same degree of parallelism. This indicates that NAF algorithm is the most recommended choice for performing ECC scaler multiplication operation the regardless of whether parallel or sequential design was used. both sequential and parallel approaches.

Implementing ECC using different degrees of parallelism provides useful trade-off between speed and resources-consumption. This contributes in developing several design choices that satisfy a variety of cryptographic applications. Required speed level and available resources of a particular application are the main factors that determine which ECC design that should be adopted for that application.

Table 1: Comparison between proposed ECC designs for Edward Curve Using NAF Algorithm

Elliptic Curve Representation over GF (p)	ECC design	Consumed Resources	Time Delay			Cost Factors	
			Point Doubling	Point Addition	Overall Time	AT	AT2
Edward Curve	5-PM	5 M, 2 A	5	5	6.6	33	217.8
	4-PM	4 M, 2 A	6	6	8	32	256
	3-PM	3 M, 2 A	7	7	9.3	27.9	259.47
	2-PM	2 M, 2 A	11	10	14.3	28.6	408.98
	Sequential	1 M, 1 A	21	20	27.6	27.6	761

Table 2: Comparison between proposed ECC designs for Binary Edward Curve Using NAF Algorithm

Elliptic Curve Representation over GF (p)	ECC design	Consumed Resources	Time Delay			Cost Factors	
			Point Doubling	Point Addition	Overall Time	AT	AT2
Binary Edward Curve	7-PM	7 M, 4 A	4	5	5.6	39.2	219.5
	6-PM	6 M, 4 A	5	5	6.6	39.6	261.4
	5-PM	5 M, 4 A	5	5	6.6	33	217.8
	4-PM	4 M, 4A	6	6	8	32	256
	3-PM	3 M, 2 A	8	7	10.3	30.9	319.3
	2-PM	2 M, 2 A	11	10	14.3	28.6	408.98
	Sequential	1 M, 1 A	22	20	28.6	28.6	817.9

Table 4 presents the performance results for Edward ECC implementations using NAF and binary algorithms. As expected, the speed of Edward ECC has been enhanced using NAF algorithm. The 5-PM ECC design scored the shortest time delay, which is 3.874 m sec. NAF algorithm implementations achieved the highest performance results for all design choices. As shown by results obtained from the Xilinx simulation tool, NAF and binary algorithms consume comparable set of resources estimated by the number of required LUTs for each ECC design.

The results were expected and normal with regard to resources-consumption level. ECC designs with higher levels of parallelism tend to use more resources than designs with less penalization levels. The sequential design implementations for both curves achieved the lowest resources consumption levels.

Table 3: Performance results for Binary Edward ECC using NAF and Binary algorithms

ECC Scaler Multiplication Methods	7-PM Design	6-PM Design	5-PM Design	4-PM Design	3-PM Design	2-PM Design	Sequential Design
NAF Algorithm	3.279	3.874	3.874	4.648	6.016	8.337	16.667
Binary Method	3.874	4.469	4.469	5.362	6.849	9.527	19.047

TABLE 4: PERFORMANCE RESULTS FOR EDWARD ECC USING NAF AND BINARY ALGORITHMS

ECC Scaler Multiplication Methods	5-PM Design	4-PM Design	3-PM Design	2-PM Design	Sequential Design
NAF Algorithm	3.874	4.648	5.421	8.337	16.072
Binary Method	4.469	5.362	6.254	9.527	18.452

C. Performance Analysis for NAF and Binary Algorithms

This section presents a comparative analysis for Edward and Binary Edward ECCs using NAF and Binary algorithms. This research work studied the speed or time-consumption for both types of elliptic curve when implemented using NAF and Binary algorithms. Experimental results showed that using NAF algorithm has improved the speed of elliptic curve scaler multiplication operation for all proposed ECC implementations. The time-consumption of ECC encryption process is represented in terms of the time delay of the scaler multiplication, which is the key operation in elliptic curve encryption. The use of NAF algorithm accelerated ECC computations for both parallel and sequential design approaches.

An important feature of NAF algorithm is that its ability to reduce the number of point addition operations that happen during the encryption process in comparison with Binary method. This feature played significant role in reducing the total time-consumption of ECC encryption for the two representations of elliptic curve that have been studied in this research.

Figure 4 shows performance comparison between Binary Edward ECC implementations using NAF and Binary algorithms.

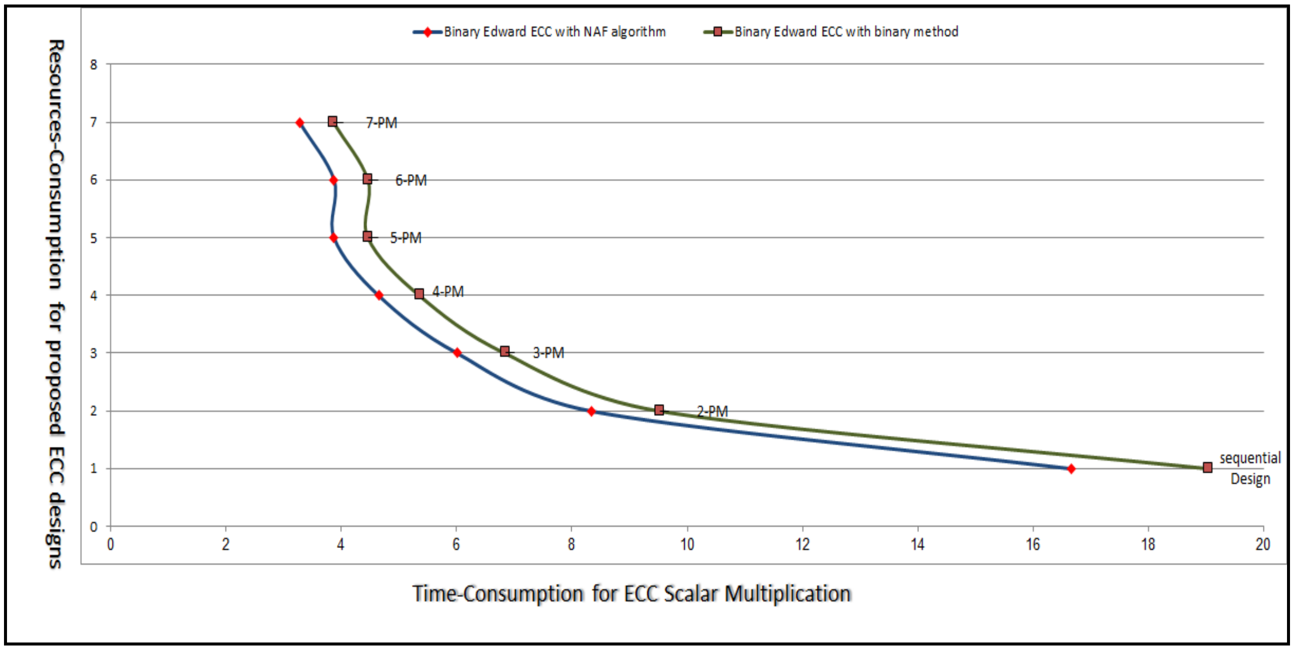

Fig. 4 Comparison between Binary Edward ECC implementations using NAF and Binary Algorithms

It can be seen from the figure that the highest speed level or shortest time delay was scored by the 7-PM ECC implementation using NAF algorithm. Also, NAF algorithm has achieved better performance for sequential ECC design.

It be concluded from analyzing ECC performance that implementation using NAF algorithm represents the most efficient choice regardless level of available resources. That is, ECC using NAF algorithm always scores higher performance level for all possible parallel design choices as well as the sequential design.

Smiler trend for performance results was shown by experiments on Edward ECC designs. Again, the use of NAF algorithm has uplifted the speed of encryption process for all design choices including the sequential design. Figure 5 presents a comparison between NAF and binary algorithms implementations for Edward curve. The highest speed level for Edward curve can be achieved using 5-PM design. For this and all other design choices, the performance of ECCs using NAF algorithm overcomes the performance level obtained using Binary method. Therefore, the use of NAF algorithm to perform the scaler multiplication operation represents the most efficient choice for all cryptographic applications no matter the level of resources that is available for that application.

An important observation from experiments conducted in this study is that the most important improvement on the performance level was achieved for sequential ECC design implementation using NAF algorithm in comparison with the corresponding sequential implementation using the Binary algorithm. This observation appears in both elliptic curve representations that were examined in this article. The difference in sequential ECC performance using NAF and binary algorithms for Binary Edward and Edward curves is depicted in figures 4 and 5 respectively.

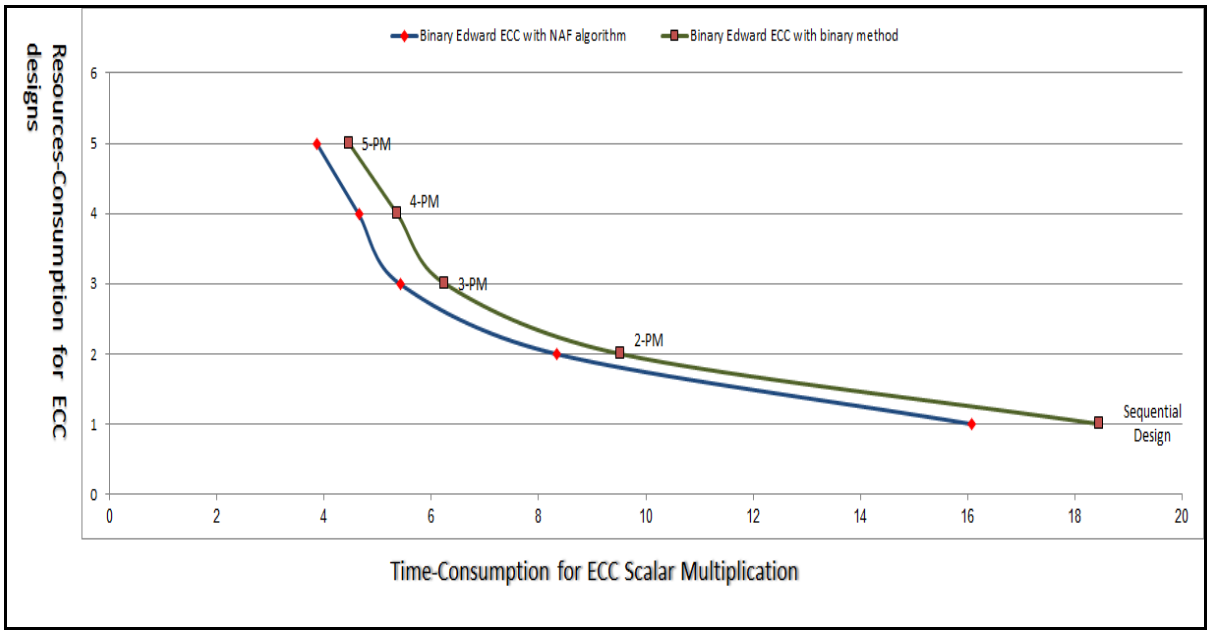

Fig. 5 Comparison between Edward ECC implementations using NAF and Binary Algorithms Performance Comparison with Previous Research Works

This section presents performance comparison between proposed Edward and Binary Edward ECC implementations using NAF algorithm and ECC implementations found in previous studies. It can be noticed from table 5 that proposed Binary Edward ECC using the 7-PM design overcomes all previous ECC implementations as well as proposed Edward 5-PM ECC implementation in the current research. The 7-PM ECC using Binary Edward curve and NAF algorithms achieved the shortest time delay for scaler multiplication operation. The best performance level for Binary Edward ECC that was achieved in previous studies equals 3.874 m sec where current Binary Edward ECC implementation scored a performance level equals 3.279 m sec using similar resources and a key size of 256 bits. This represents considerable improvement on the performance level taking into account that using longer key sizes such as 1024 bits will increase the gap in terms of time-consumption between proposed crypto-system and previously used ECCs. This gives the advantage to proposed ECC for cryptographic applications that require using longer key sizes to improve the security level.

Furthermore, proposed 5-PM Edward ECC using NAF algorithm accomplished the shortest time delay compared to previous Edward curve implementations using the same level of resources. Current Edward curve implementations consumes 3.874 m sec to perform scaler multiplication which exceeds the latest highest-speed level estimated by 4.469 m sec.

For certain crypto application, where the high performance level is a key priority, the proposed 7-PM Binary Edward ECC using NAF algorithm represents the most efficient choice to be considered.

Eventually, experiments have showed that NAF algorithm can reduce the time-consumption of scaler multiplication operation, and hence increase the speed of the encryption process in comparison with binary method.

Table 5: PERFORMANCE COMPARISON with previous ECC implementations in literature

No	Name, Ref.	Finite Field, Key size	KP time (m sec)	Device
1	Lo'ai et al.[12] (Edwards)	GF (p), 512-bit	33.301	XC5VLX30
2	Kerins et al. [25]	GF (2n), 233-bit	13.2	XCV2000E
3	Nele et al. [26]	GF (2n), 160-bit	3.8	XCV800-4
4	Kocabas et al. [28] (Binary Edwards)	GF (2n), 163-bit	149.50	ASIC
5	Alkhatib [18] (Binary Edward curve)	GF (p), 256-bit	3.874	XC5VLX30
6	Alkhatib [21] (Edward curve)	GF (p), 256-bit	4.469	XC5VLX30
7	Proposed Edward ECC implementation using NAF algorithm	GF (p), 256-bit	3.874	XC5VLX30
8	Proposed Binary Edward ECC implementation using NAF algorithm	GF (p), 256-bit	3.279	XC5VLX30

5. Conclusion and Future Work

This research presented high-performance ECCs for both Binary Edward and Edward representations using NAF algorithm. Proposed ECCs achieved the shortest time delay for scaler multiplication operation which contributes significantly in increasing the speed of encryption process. The use of NAF algorithm played the key role in reducing the time-consumption of elliptic curve point operations that represent the main building blocks of scaler multiplication. In particular NAF algorithm has the ability to reduce the number of elliptic curve point addition operations performed during encryption process.

In order to improve the performance even further, ECC computations where implemented using homogenous coordinates system in order to avoid the time-consuming inversion operation. Moreover, adopting parallel ECC designs has significantly accelerated the cryptosystem computations.

In comparison with previous ECC implementations using binary method, current implementations using NAF algorithm has accomplished better performance level for all possible design choices. In this study, Edward and Binary Edward ECCs were implemented using both NAF and binary algorithms with different degrees of parallelism. The proposed 7-PM Binary Edward ECC accomplished the highest performance level. On the other side, Edward ECC using 5-PM achieved the best performance results for Edward curve.

Experiments illustrated that ECC implementation NAF algorithm increases the performance level no matter the level of resources consumed.

The major enhancement of performance is noticed when using NAF algorithm for sequential design implementation in comparison whith previouse implementations that used the known binary method.

In future, several research dimensions can be explored. The use of NAF algorithm for different elliptic curve representations and its effect on the performance level. In addition, investigating other algorithms to perform the scaler multiplication operation such as the Montgomery ladder algorithm and their potential improvement on ECC performance seem interesting research topics.

References

[1] Wade Trappe, Lawrence. c, Introduction to Cryptography with Coding Theory. Washington, Pearson Prentice Hall, 2002.

[2] N. Koblitz, "Elliptic curve cryptosystem", Mathematics of Computation, Vol. 48, pp. 203-209, 1987.

[3] V. Miller, "Uses of elliptic curves in cryptography", Lecture Notes in Computer Science, Vol. 218, pp. 417-426, 1986.

[4] Blake, Seroussi, and Smart. Elliptic Curves in Cryptography. Cambridge University Press: New York, 1999

[5] Tanja Lange, "A note on L´opez-Dahab coordinates", Faculty of Mathematics, Technical University of Denmark, 2006.

[6] David, Nigel, and Jacques, "Projective coordinates Leak", Applied research and security center, France.

[7] GuerricMeurice de Dormale, Jean-Jacques Quisquater. High-speed hardware implementations of Elliptic Curve Cryptography: A survey. Journal of Systems Architecture 53 (2007) 72-84, by Elsevier.

[8] A. Satoh, K. Takano, "A scalable dual-field elliptic curve cryptographic processor," IEEE Transactions Computers 52 (4) (2003) 449–460.

[9] G. Orlando, and C. Paar, "A scalable GF (p) elliptic curve processor architecture for programmable hardware," Cryptographic Hardware and Embedded Systems - CHES 2001, Paris, France, May 14-15, 2001

[10] Adnan Gutub and Mohammad K. Ibrahim., "High Radix Parallel Architecture For GF(p) Elliptic Curve Processor", IEEE Conference on Acoustics, Speech, and Signal Processing, ICASSP 2003, Pages: 625- 628, Hong Kong, April 6-10.

[11] Adnan Gutub. Efficient Utilization of Scalable Multipliers in Parallel to Compute GF (p) Elliptic Curve Cryptographic Operations. Kuwait Journal of Science & Engineering (KJSE) 2007, 34(2): 165-182.

[12] L. Tawalbeh and Q. Abu Al-Haija. Speeding up Elliptic Curve Cryptography Computations by Adopting Edwards Curves over GF (P). International Journal of Security (IJS) 2009, CSC Journals, Malaysia, Vol.3, Issue.4, IJS-19.

[13] Q. Abu Al-Haija and Mohammad Al-Khatib. Parallel Hardware Algorithms & Designs for Elliptic Curves Cryptography to Improve Point Operations Computations Using New Projective Coordinates. Journal of Information Assurance and Security (JIAS) 2010, By Dynamic Publishers Inc., USA, Vol.4, No.1, Paper 6: (588-594).

[14] Adnan Abdul-Aziz Gutub and Mohammad K. Ibrahim, "High performance elliptic curve GF (2k) crypto-processor architecture for multimedia", IEEE International Conference

on Multimedia & Expo, ICME 2003, pages 81- 84, Baltimore, Maryland, USA, July 6-9, 2003.

[15] Adnan Gutub, “High Speed Low Power GF (2k) Elliptic Curve Cryptography Processor Architecture”, IEEE 10th Annual Technical Exchange Meeting, KFUPM, Dhahran, Saudi Arabia, March 23-24, 2003.

[16] L. Tawalbeh and Q. Abu Al-Haija. Enhanced FPGA Implementations for Doubling Oriented and Jacobi-Quartics Elliptic Curves Cryptography. Journal of Information Assurance and Security (JIAS), 2010, Volume 6, pp. 167-175

[17] Mohammad Al-khatib, Qacem, and AzmiJaafar. Hardware Architecture & Designs for Projective Elliptic Curves Point Addition Operation using Variable Levels of Parallelism. International Review on Computers and Software 2011 Vol. 6 N. 2, pp. 237-243.

[18] Mohammad Al-Khatib, Q. Abu Al-Haija, and Ramlan Mahmud. Performance Evaluation of Projective Binary Edwards Elliptic Curve Computations with Parallel Architectures. Journal of Information Assurance and Security (JIAS) 2011, By Dynamic Publishers Inc., USA, Vol.6, No.1, Paper1: (001-009).

[19] Mohammad Al-khatib, AzmiJaafar, and Q. Sbu Al-Haija. Choices on Designing GF (p) Elliptic Curve Coprocessor Benefiting from Mapping Homogeneous Curves in Parallel Multiplications. International Journal on computer science and engineering 2011, Vol.3, No.2, Paper 2: (467-480).

[20] Mohammad Alkhatib, Azmi B. Jaafar, ZuriatiZukarnain, and Mohammad Rushdan. On the Design of Projective Binary Edwards Elliptic Curves over GF (p) Benefiting from Mapping Elliptic Curves Computations to Variable Degree of Parallel Design. International Journal on computer science and engineering 2011, Vol.3, No.4, Paper 44: (1697-1712).

[21] Mohammad Alkhatib, Azmi B. Jaafar, ZuriatiZukarnain, and Mohammad Rushdan, "Trade-off between Area and Speed for Projective Edwards Elliptic Curves Crypto-system over GF (p) using Parallel Hardware Designs and Architectures", International Review on Computers and Software, July 2011 Vol. 6 N. 4, pp. 163-173.

[22] Mohammad Al-khatib, AzmiJaafar, Zuriati Ahmad Zukarnain, and MohamadRushdanMd Said, "Hardware Designs and Architectures for Projective Montgomery ECC over GF (p) benefiting from mapping elliptic curve computations to different degrees of parallelism", International Review on Computers and Software, Vol. 6, N. 6, November 2011.

[23] Mohammad Al-khatib, and Adel Al-Salem. Efficient hardware implementations for Tripling Oriented Elliptic Curve Crypto-system. International Review on Computers and Software. 2014, Vol. 9, N. 4.

[24] A. Satoh, K. Takano, “A scalable dual-field elliptic curve cryptographic processor,” IEEE Transactions Computers 52 (4) (2003) 449–460.

[25] T. Kerins, E. M. Popovici and W. P. Marnane. “An FPGA Implementation of a Flexible, Secure Elliptic Curve Cryptography Processor”, International Workshop on Applied Reconfigurable Computing-ARC 2005, IADIS press, pp.22-30.

[26] Nele Mentens, Siddika Berna Ors, Bart Preneel, “An FPGA Implementation of an Elliptic Curve Processor over GF (2m)”, In Proceedings of the 2004 ACM Great lakes Symposium on VLSI, GLSVLSI 2004: VLSI in the Nanometer Era. pp. 454-457.

[27] Turki F. Al-Somani. Performance Evaluation of Elliptic Curve Projective Coordinates with Parallel GF (p) Field Operations and Side-Channel Atomicity. Journal of Computers 2010. JCP.5.1.99-109.

[28] Kocabas, U., J. Fan, and I. Verbauwhede, “Implementation of Binary Edwards curves for very-constrained devices,” In 21st IEEE International Conference on Application-specific Systems Architectures and Processors, ASAP 2010, Rennes, France, pages 185 –191.

[29] Mohammad Alkhatib, “Cost-Effective Implementations for Weirstrass and Montgomery Elliptic Curve Crypto-systems”, International Journal of Computer Science and Information Security, 2016, 14 (9), 363.

Mohammad Al-khatibe is an assistant Professor in faculty of Computer and Information Sciences at Imam university. He received the bachelor degree in Computer Science. His Master degree was obtained from Depaul University in the field of Information Systems. He also achieved PhD in Computer Science (Security in Computing) from University PUTRA Malaysia (UPM). His research interest includes: information security, cryptography, and Elliptic Curve algorithm.