Journal of Applied Research and Technology

www.jart.icat.unam.mx

ICAT
Instituto de Ciencias Aplicadas y Tecnología

*Journal of Applied Research and Technology* 22 (2024) 295-307

Original

# Secured V2X communication using optimized prime field ECC architecture

A. Kamaraj

*Department of Electronics and Communication Engineering,*
*Mepco Schlenk Engineering College, Sivakasi*

Received 05 19 2023; accepted 01 16 2024
Available 04 30 2024

**Abstract:** In today's world, vehicles can communicate with one another, pedestrians, roadside infrastructure, and other moving objects using the basic type of vehicular communication known as vehicle-to-everything (V2X) communication. In addition to road safety, security and privacy issues must be taken into consideration in V2X activities. The objective of this research is to ensure an important level of security in various forms of vehicular communication (V2V, V2I, and V2N) and to support vehicles in safely receiving all keys and messages from roadside unit (RSU), other vehicles, or the network with the support of simple cryptographic techniques. This research work develops Elliptic Curve Cryptography (ECC) crypto processor to achieve area efficient, high-speed ECC processor to reach the desired objective. The Koblitz curve secp256k1 is supported by the designed ECC processor for 256-bit point multiplication and point addition. Here, prime fields are incorporated to increase security. A powerful illustration of the "divide and conquer" strategy's ability to accelerate multiplication asymptotically is given by the Karatsuba algorithm. The pipeline technology accelerates multiplication process much faster. For the 256-bit prime field, the proposed pipelined Karatsuba multiplier based ECC processor is implemented on the Xilinx Virtex-7 FPGA. With a maximum clock frequency of 238.40MHz, the proposed Karatsuba-based ECC processor executes 256-bit single point multiplication in 0.937ms, providing 273.21kbps throughput, and taking up 8.42k slices in a Virtex-7 FPGA. Scalar multiplication is extended by incorporating a pipeline by increasing the highest clock frequency by up to 7.97%, which decreases time consumption by 9.90% and boosts throughput by 10.99%. In terms of area, operating frequency, area-delay product, and throughput, the suggested pipelined Karatsuba multiplier based ECC processor performs better than the existing designs.

*Keywords:* Elliptic Curve Cryptography, Point Multiplication, Montgomery, Left to Right algorithm, Karatsuba

*Corresponding author.
*E-mail address:* kamarajvlsi@gmail.com (A. Kamaraj).
Peer Review under the responsibility of Universidad Nacional Autónoma de México.

## 1. Introduction

Under ECC recommendation (08)01 (ECC Recommendation, 2020) and ECC decision (08)01 (ECC Decision, 2020), which were both endorsed by the ECC (CEPT) in March 2020, European authorities have designated the bands 5855-5875 MHz and 5875-5925 MHz, together referred to as the 5.9 GHz spectrum, for use by road Intelligent Transport Systems (ITS).

For this V2X, the security infrastructure needs to conduct the following tasks:

- Verify the sender's identity to ensure that the message came from a reliable source.
- Verify the message's integrity to ensure that its contents have not been changed.
- By making sure that no individual or equipment-identifying information that could be used to observe or track the users is communicated, you can safeguard the privacy of the participants.

To increase the security of information, one of the fastest and most reliable asymmetric key cryptosystems is ECC. The arithmetic operations in a finite field determine how complex the hardware for the ECC is to implement. Addition, subtraction, multiplication, and inversion are the arithmetic operations involved in ECC (Nadikuda & Boppana, 2022). ECC is currently regarded as one of the greatest public key cryptographies (PKC) algorithms and provides significantly more security per bit than (Rivest et al.,) RSA (Rivest et al., 1978). Therefore, its effective hardware implementation is of utmost significance to achieving the speed requirements in real-time applications. The ECC based crypto systems are present over either one of the specific prime field or general prime field category.

Anatoly Karatsuba founded the Karatsuba algorithm in 1960, and it was published in 1962. Divide-and-conquer is the fundamental idea behind Karatsuba's method. In all base systems, Karatsuba can be used to multiply numbers (base-10, base-2, etc.). Introducing pipelining in Karatsuba multiplication doubles the frequency of operation (KS & Elavarasi, 2013) and improves performance.

### *1.1. Elliptic curves over prime fields*

A finite set of elliptic curve points that satisfy the following Equation (1) is known as Elliptic Curve Cryptography (ECC).

$$y^2 = x^3 + ax + b \tag{1}$$

over a finite field $F_p$ of $p$ elements, where $p$ is a prime number greater than 3 and $p \boldsymbol{\Sigma} Z$, with a point at infinity. The formula k = $\log_2 p + 1$ represents the bit size of $p$. For point addition operations, the curve creates a cyclic group.

Algebraic addition of points on an elliptic curve defined over a finite field $GF(p)$ with given points $A(x_A, y_A)$ and $B = (x_B, y_B)$ and calculation of their sum $R = (x_R, y_R)$, requires the following operations: (Bernstein, 2006)

$t_1 = y_A - y_B$ (modular subtraction)
$t^2 = x_A - x_B$ (modular subtraction)
$t^2 = t^{-1}{}_2$ (modular inversion)
$\lambda = t_4 = t_2 * t_1$ (modular multiplication)
$t_1 = t_4 * t_4$ (modular squaring)
$t_3 = x_A + x_B$ (modular addition)
$x_R = t_1 = t_1 - t_3$ (modular subtraction)
$t_2 = x_R - t_1$ (modular subtraction)
$t_3 = t_2 * t_4$ (modular multiplication)
$y_R = t_3 = t_3 - y_A$ (modular subtraction)
where
$x_R = \lambda^2 - (x_A + x_B)$; $y_R = \lambda(x_A - x_R) - y_A$; $\lambda = y_A - y_B/x_A - x_B$

The term "vehicle-to-everything (V2X) communication" refers to a set of apps, services, and recent technologies that enable internal connectivity between equipment already present in the vehicle and/or enable communication of the car to external devices and networks. Therefore, for V2X to begin a secure network/cloud connection, authentication is required (Yoshizawa et al., 2022). The applications require high speed with minimal hardware resource usage along with improved authentication security.

Therefore, in the architectures described in (Asif et al., 2017; Hu et al., 2018; Hossain et al., 2017; Islam et al., 2020; Islam et al., 2019; Javeed & Wang, 2017; Shah et al., 2019), many optimization strategies have been applied to achieve high speed and decrease hardware resources. To reduce the critical route and raise the clock frequency, pipelining is applied (Imran et al., 2019). In Rashidi (2018), Khan and Benaissa (2013), bit/digit serial multipliers are employed to reduce hardware resources while significantly lowering design performance.

Multiple modular multipliers are employed in Khan and Benaissa, (2016) to reduce latency. In Ionita and Simion (2021), Morales-Sandoval et al. (2021), the Co-processor designs for Elliptic-curve Diffie–Hellman (ECDH)-based key agreements are discussed. To accomplish flexibility, a co-processor involves the integration of a field programmable gate array (FPGA) with a host device (such as a microcontroller or processor) without considering the performance factors (Rashid et al., 2019). The only constraint that the designs reported in Khan and Benaissa (2013) are the acceleration or implementation of Elliptic curve point multiplication (ECPM) computations. Like this, the designs reported in (Ionita & Simion, 2021; Morales-Sandoval et al., 2021) are limited by the enormous computation times.

In Imran et al. (2019) a two-stage pipelined design for ECPM computation over Galois field (GF) - ($2^{163}$) was proposed. To minimize the critical path and increase clock frequency, pipelining is used. In (Hossain et al., 2016), an ECC-based crypto-processor for point multiplication over GF ($2^{163}$) is implemented in high-performance hardware. An effective finite-field arithmetic unit, a control unit, and a memory unit make up the design of affine coordinate systems. Key-exchange hardware implementations are designated for use of ECDH cryptographic primitives in an effective architecture for quantum-safe hybrid key exchange (Azarderakhsh et al., 2021). To save size and power consumption, (Ionita & Simion, 2021) offloads the ECDH operations to an FPGA using a high-level synthesis (HLS) technique. A low-cost ECC hardware accelerator architecture on FPGA is given in Morales-Sandoval et al. (2021) for wireless sensor nodes. As a result, it is essential to provide a high-speed, low-resource ECC processor architecture for key agreement.

Bernstein stated that prime fields "have the virtue of minimizing the number of security concerns for Elliptic-Curve Cryptography". There is general agreement that prime fields are the safe, conservative choice for ECC (KS & Elavarasi, 2013). Due to the processing of the Karatsuba algorithm, the bit length rises together with the number of multiplications. Additionally, the number of hardware increases is in direct proportion to the number of multiplications. The Karatsuba algorithm performs fewer multiplications than the traditional multiplication method when they are compared to one another (Eyupoglu, 2015; Mishra & Pradhan, 2012). For LWR-based crypto-processors, the optimized hierarchical Karatsuba algorithm is appropriate. The degree-256 polynomial multiplication is reduced to coefficient-wise multiplication using an 8-level hierarchical Karatsuba structure. The area overheads of the scheduling approach are minimized by using an optimized pre-/post-processing structure and a hardware-efficient Karatsuba scheduling technique (Zhu et al., 2021).

### 1.2. Elliptical Curve Cryptography preliminaries

A solution set of the cubic equation with two variables is an elliptic curve. The elliptic curve over a finite field *k* can be defined by the Weierstrass equation. The equation is expressed as shown in (1) and must satisfy the following Equation (2) condition,

$$4a^3 + 27b^2 \neq 0 \qquad (2)$$

where *a* and *b* are integer coefficients. The points on the elliptic curve are the points over $E_p(a,b)$ equation. The parameters *a*, *b*, and *p* determine the number of rational points over the curve. The set of all points on the elliptic curve with the point at infinity is grouped as an abelian group.

The elliptic curve points are found within the range where the *x* and *y* coordinate of the abelian group points lie within the range (0, *p*). A graph with the integer coefficients as *a = 1* and *b = 2* with the prime number *p*-value as 11 is shown in Figure 1. The points which satisfy this Weierstrass equation are tabulated in Table 1.

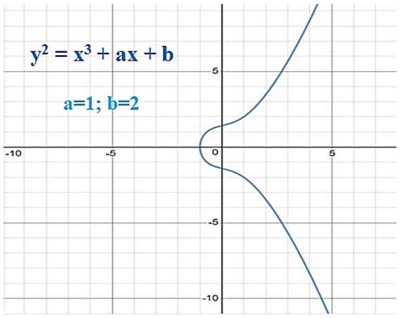


Figure 1. Prime field ECC curve.

Table 1. Look up table co-ordinates for mapping 4-bit binary stream.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0, 0 | 5, 0 | 7, 0 | 10, 0 | 2, 1 | 1, 2 | 4, 2 | 6, 2 |
| 8, 4 | 9, 5 | 9, 6 | 8, 7 | 1, 9 | 4, 9 | 6, 9 | 2, 10 |

### 1.3. Objectives and scope of this research

- With the support of basic cryptography techniques, this research aims to guarantee a high degree of security in the different vehicular communication protocols (V2V, V2I, and V2N) and to enable vehicles to securely receive all keys and messages from roadside units (RSU), other vehicles, or the network.
- Effective security can be attained by incorporating Elliptic Curve Cryptography (ECC), which has the best security features.
- The proposed ECC crypto processor supports the Koblitz curve secp256k1 for 256-bit point addition and multiplication.
- The Karatsuba method follows the "divide and conquer" strategy that can asymptotically speed up multiplication.
- The inclusion of pipelining speeds up the multiplication process of the ECC processor even further.

Key features of this research work ia as follows,

- *secp256k1 Koblitz curve*
- *256-bit prime field*
- *256-bit point addition and multiplication*
- *Karatsuba multiplication*
- *Pipeline to increase speed*
- *Implemented on the Xilinx Virtex-7 FPGA*

## 2. Materials and methods

### 2.1. Materials (algorithms and hardware architectures for ECC implementation)

### 2.1.1. Finite field arithmetic

Modular arithmetic is conducted for ECC which is done over the prime field. This wraps any number greater or equal to the prime number p to the range of 0 to *p*-1. Some of the modular arithmetic operations conducted in the Elliptic Curve Cryptography are addition, subtraction, multiplication, division, and multiplicative inverse. In division, the multiplicative inverse is needed for the computation since *a / b mod p* is defined as $a \times b^{-1} (mod\ p)$. The multiplicative inverse can be found using the following algorithm (Parthasarathy, 2012).

*Algorithm: Multiplicative inverse*

```
$r_{11} = p$
$r_{12} = b$
$t_1 = 0$
$t_2 = 1$
while ( $r_{11}$ != 1 )
        $q = r_{11} / r_{12}$
        $k_1 = t_2 * q$
                    $t = t_1 - k_1$
                    $t_1 = t_2$
                    $t_2 = t$
                    $r = r_{11} \% r_{12}$
                    $r_{11} = r_1$
        $r_{12} = r$
                    return $t_1$
```

To find the modulus value between two numbers a % b, the following algorithm (Rohn, 2012) has been incorporated irrespective of the sign of the numerator value ‘a’.

*Algorithm: To find the modulus*

- Step 1. If number ‘a’ is negative go to 2, else go to Step 6.
- Step 2. If (-a) % b is zero return ‘0’ else go to Step 3.
- Step 3. Calculate c = (-a) // b
- Step 4. Z = [(c + 1) * b] + a
- Step 5. Return Z
- Step 6. Return a % b

### 2.1.2. Left to Right binary method

Assume that the curve has a point *P* and that we must determine the value of *nP* for some number n. Point addition and point doubling participate in this process (Gaur et al., 2019). The flow diagram of the binary Left to Right algorithm is shown in Figure 2.

*Algorithm: Left to Right binary method* (Gaur et al., 2019)

```
• Q = ∞ (Infinite point in ECC curve)
• For i from 0 to [length(k) – 1]
If k[i] = 1,
then Q = Q + P        // Point addition
P = 2P                // Point doubling
• Return Q
```

### 2.1.3. Montgomery algorithm

The conventional algorithms are quite slow to compute multiplications like *xy mod n* because they need to divide the result to determine how many times n needs to be subtracted from the product. The lower bits are simply discarded after adding multiples of n to cancel them out rather than dividing the product and repeatedly removing n.

The modular Montgomery multiplier algorithm is as follows (Gaur et al., 2019).

```
• Q = ∞ (Infinite point in ECC curve)
• For i from [length(k) – 1] to 0
If k[i] = 0,
then P = Q + P and Q = 2Q
   Else,
then Q = Q + P and P = 2P
• Return Q
```

The flow diagram of the Montgomery algorithm is shown in Figure 3. In that, the Q point is set as infinite. i.e., Q = (0, 0) in the ECC curve initially as in binary Left to Right algorithm. k is generated as a binary stream of 3 bits (000 to 111) and is checked from the most significant bit (MSB). The loop is repeated for the length of binary stream ‘k’ times and finally, the Q value is returned as the output point of scalar multiplication (KS & Elavarasi, 2013).

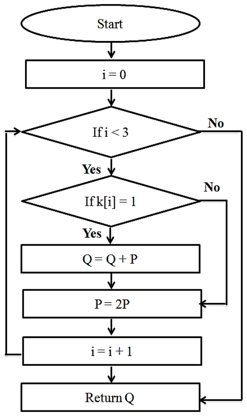


Figure 2. Flow diagram of binary Left to Right algorithm.

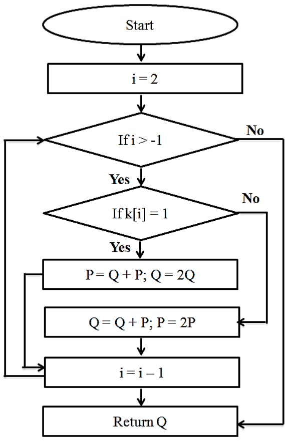


Figure 3. Flow diagram of Montgomery algorithm.

### 2.1.4. Karatsuba algorithm

Divide-and-conquer is the fundamental idea behind Karatsuba's algorithm, which uses a formula to enable one to calculate the product of two large numbers, x and y, using three multiplications of smaller numbers, each with about half as many digits as x or y, along with some additions and digit shifts (Eyupoglu, 2015; Mishra & Pradhan, 2012).

The flow diagram of the Karatsuba Algorithm is shown in Figure 4, employs the divide and conquers strategy. By using addition operations rather than multiplication, this approach reduces the number of multiplier units needed to execute the largest multiplication (Mishra & Pradhan, 2012). The overall speed has increased since an adder operates faster than a multiplier.

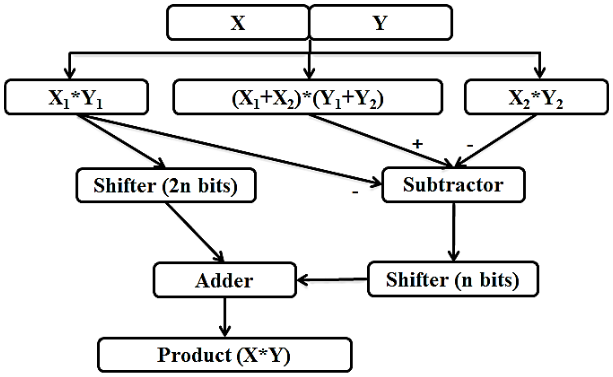


Figure 4. Flow diagram of Karatsuba algorithm.

## 2.2. Proposed methodology

The proposed ECC processor architecture's FPGA implementation methodology is presented in this section. The required curve order, coefficients, and base point coordinates are selected based on the NIST standard. In this research work, a 256-bit ECC processor has been considered. The ECC processor is designed using Verilog HDL, and simulation is performed using HDL Simulator. Xilinx ISE 14.6 was utilized for the synthesis, placement, and routing of the component. The proposed method implemented on the Virtex-7 (XC7VX485T-2FFG1761C) FPGA to attain the optimal speed and area.

Based on IEEE 802.11p, the first V2X technology was standardized. According to the IEEE 1609 series and SAE International (SAE) standard J2735, the V2X system using 802.11 OCB mode is also known as Dedicated Short-Range Communication (DSRC), and its upper layer is known as wireless access in vehicular environment (WAVE) (Houmer et al., 2022). Intelligent transport system G5 (ITS-G5) and C-ITS are the names given to ITS systems based on IEEE 802.11 OCB mode in Europe (Yoshizawa et al., 2022).

Elliptic Curve Cryptography (ECC) is used by ETSI ITS (ETSI, 2021) and IEEE 1609.2 (IEEE1609, 2016) to provide digital signatures and encrypt messages. However, it is well known that public key cryptography techniques now in use, particularly those relying on ECC, are weak in the face of a quantum computer (Shor, 1999). The size of the public key and signature from the ECC-based approach increases if we assume a code-based quantum-resistant signature mechanism. The ECC based crypto processor provides better resilience against the above parameter and the computational cost is also comparatively low (Kamaraj & Marichamy, 2022).

Figure 5 illustrates the procedure of authentication and communication between the vehicle and the roadside unit (RSU). We employ the elliptical curve encryption algorithm to ensure that the message authentication is secure for data transmission between vehicles as shown in Figure 6. The execution time (for key exchange, digital signature, and asymmetric encryption) of the ECC algorithm is comparatively faster than Diffie–Hellman (DH), digital signature algorithm (DSA) and RSA (Houmer et al., 2022).

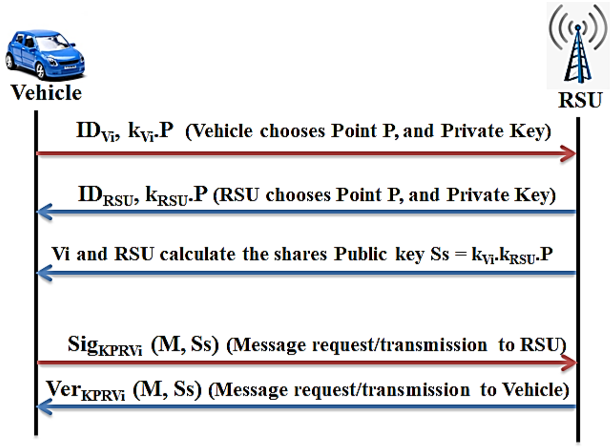

Vehicle
RSU
$ID_{Vi}$, $k_{Vi}.P$ (Vehicle chooses Point P, and Private Key)
$ID_{RSU}$, $k_{RSU}.P$ (RSU chooses Point P, and Private Key)
Vi and RSU calculate the shares Public key Ss = $k_{Vi}.k_{RSU}.P$
$Sig_{KPRVi}$ (M, Ss) (Message request/transmission to RSU)
$Ver_{KPRVi}$ (M, Ss) (Message request/transmission to Vehicle)

Figure 5. Authentication and communication between vehicle and RSU.

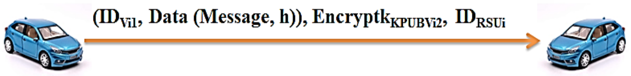

($ID_{Vi1}$, Data (Message, h)), $Encryptk_{KPUBVi2}$, $ID_{RSUi}$

Figure 6. Communication between vehicles.

### 2.2.1. Point multiplication and point doubling

Multiplication of the ordered pair with a random integer is termed as point multiplication or scalar multiplication. The complex area in ECC is scalar multiplication which involves the computation of *kP* where *P* is a point on the abelian group (Kodali et al., 2015). The architectural design flow of scalar multiplication is shown in Figure 7a. Scalar multiplication is performed in the calculation of the public key ($n_b$*G(x, y)), multiplication of the public key with a random integer (k*$P_b$ (x, y)) and in the first pair of the ciphertext (k*G(x, y)). If the coordinates of points P and Q are equal, then point doubling is conducted. When $P_x = Q_x$ and $P_y = Q_y$, then R (x, y) = 2P(x, y). The architecture of point doubling is shown in Figure 7b.

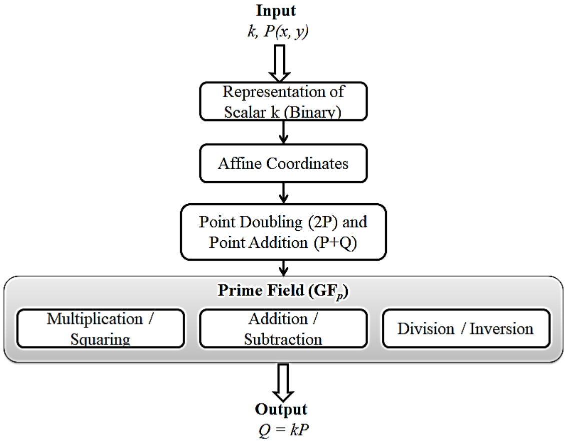


Figure 7a. Scalar multiplication architecture.

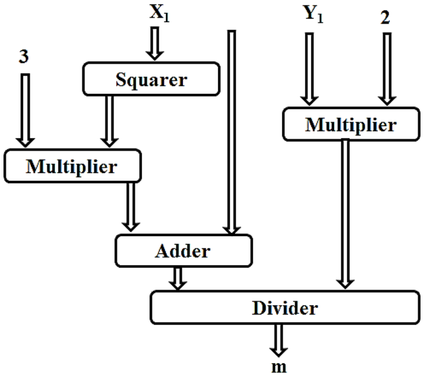


Figure 7b. Point doubling architecture.

### 2.2.2. Point addition

Two points that satisfy the abelian group, when added give a point that also satisfies the abelian group of the elliptic curve. That is when points P $(x_1, y_1)$ and Q $(x_2, y_2)$ are added it gives a point R(x, y) which lies in the ECC curve of the same Weierstrass equation. The architecture of point addition is shown in Figure 8. When the points P and Q are non-negatives of each other, the slope of the line that joins them is given by

$$\Delta = [(y_2 - y_1) / (x_2 - x_1)] \bmod p \qquad (3).$$

The coordinates of the resultant point R are given by,

$$R_x = \Delta^2 - x_2 - x_1 \qquad (4).$$

$$R_y = \Delta (x_1 - R_x) - y_1 \qquad (5).$$

In point addition and point doubling, the $R_x$ value is determined by the equation given below.

$R_x = \Delta^2 - x_2 - x_1$

In point doubling since $x_1$ and $x_2$ are equal, $R_x$ is also given by

$R_x = \Delta^2 - x_2 - x_1$.

$R_x = \Delta^2 - x_1 - x_1$.

$R_x = \Delta^2 - 2x_1$

The architectural design for calculating $R_x$ and $R_y$ is shown in Figure 9a and Figure 9b. In point addition and point doubling, the $R_y$ value is determined by the equation given below.

$R_y = \Delta (x_1 - R_x) - y_1$

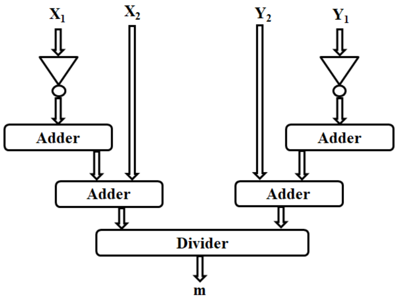


Figure 8. Point addition architecture.

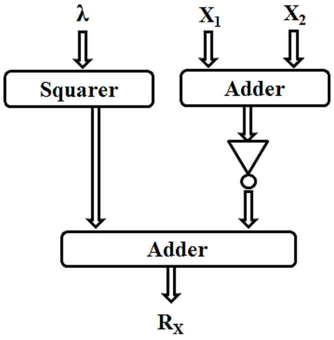


Figure 9a. X-coordinate architecture.

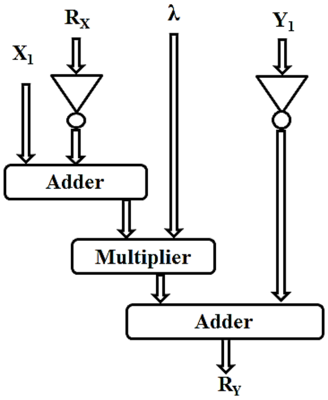


Figure 9b. Y-coordinate Architecture.

## 3. Results and discussion

The proposed ECC processor architecture's FPGA implementation is presented in this section. Based on the NIST standard, the necessary parameters, including curve order, coefficients, and base point coordinates, are chosen. Here, a 256-bit ECC processor has been considered. Verilog HDL is used for the design of the ECC processor, and HDL Simulator is used for simulation. It was synthesized, placed, and routed with the help of Xilinx ISE 14.6. To achieve the optimal speed and area, the Virtex-7 (XC7VX485T-2FFG1761C) FPGA is employed in the proposed method. The performance factors throughput and efficiency are calculated based on Equation (6) (Kamaraj & Marichamy, 2022).

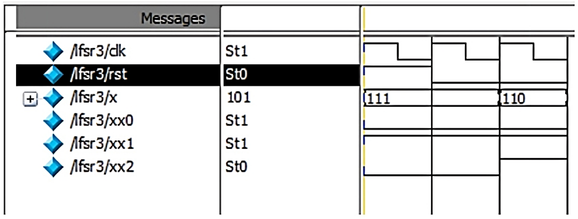
Cycle = Time for one ECPM × maximum frequency

AT/Bit = Area-delay product / number of bits

Throughput = (Maximum frequency × number of bits) / number of clock cycles

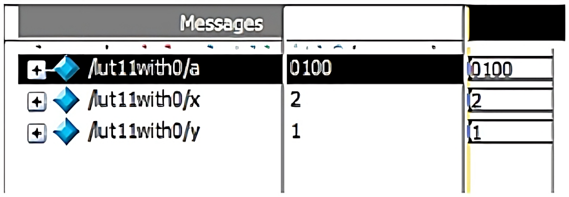
$$\text{Efficiency} = \text{Throughput/area} \tag{6}$$

The simulation results of LFSR, lookup table, inverse lookup table, multiplicative inverse, modulus, point addition, point doubling, point multiplication (for Left to right, Montgomery, Karatsuba), encryption and decryption are shown in Figure 10 (a-l).



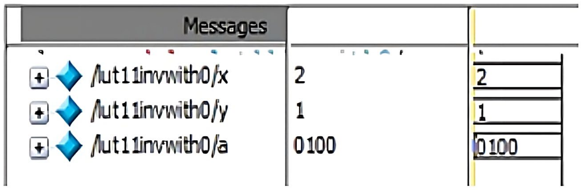
| Messages | |
|---|---|
| /lfsr3/clk | St1 |
| /lfsr3/rst | St0 |
| /lfsr3/x | 101 |
| /lfsr3/xx0 | St1 |
| /lfsr3/xx1 | St1 |
| /lfsr3/xx2 | St0 |

Figure 10a. LFSR.



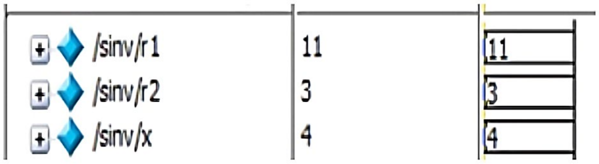
| Messages | |
|---|---|
| /lut11with0/a | 0100 |
| /lut11with0/x | 2 |
| /lut11with0/y | 1 |

Figure 10b. Lookup table.



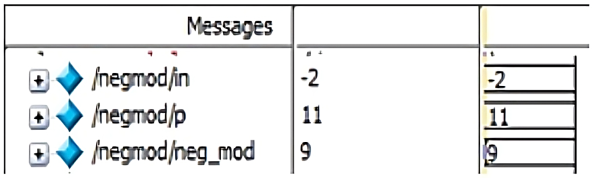
| Messages | |
|---|---|
| /lut11invwith0/x | 2 |
| /lut11invwith0/y | 1 |
| /lut11invwith0/a | 0100 |

Figure 10c. Inverse lookup table.



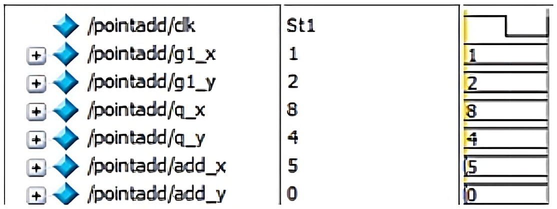
| | |
|---|---|
| /sinv/r1 | 11 |
| /sinv/r2 | 3 |
| /sinv/x | 4 |

Figure 10d. Multiplicative inverse.



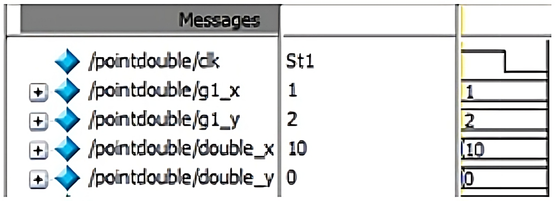
| Messages | |
|---|---|
| /negmod/in | -2 |
| /negmod/p | 11 |
| /negmod/neg_mod | 9 |

Figure 10e. Modulus.



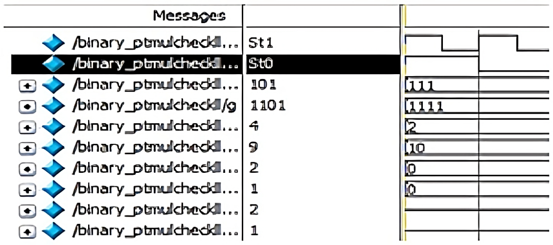
| | |
|---|---|
| /pointadd/clk | St1 |
| /pointadd/g1_x | 1 |
| /pointadd/g1_y | 2 |
| /pointadd/q_x | 8 |
| /pointadd/q_y | 4 |
| /pointadd/add_x | 5 |
| /pointadd/add_y | 0 |

Figure 10f. Point addition.



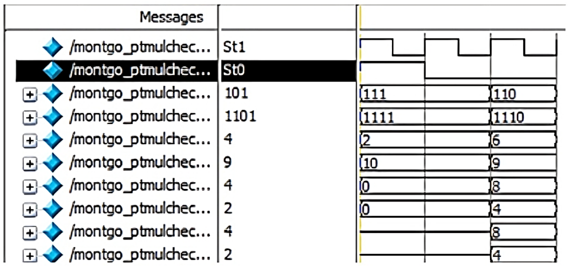
| Messages | |
|---|---|
| /pointdouble/clk | St1 |
| /pointdouble/g1_x | 1 |
| /pointdouble/g1_y | 2 |
| /pointdouble/double_x | 10 |
| /pointdouble/double_y | 0 |

Figure 10g. Point doubling.



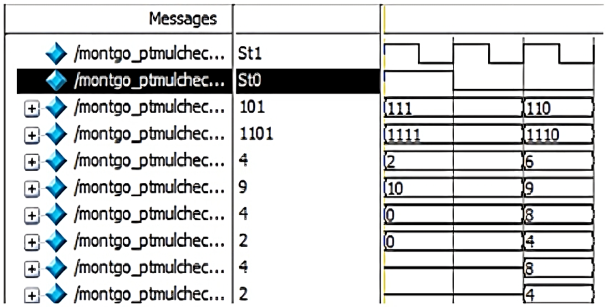
| Messages | |
|---|---|
| /binary_ptmulcheckl... | St1 |
| /binary_ptmulcheckl... | St0 |
| /binary_ptmulcheckl... | 101 |
| /binary_ptmulcheckl/g | 1101 |
| /binary_ptmulcheckl... | 4 |
| /binary_ptmulcheckl... | 9 |
| /binary_ptmulcheckl... | 2 |
| /binary_ptmulcheckl... | 1 |
| /binary_ptmulcheckl... | 2 |
| /binary_ptmulcheckl... | 1 |

Figure 10h. Binary Left to Right point multiplication.



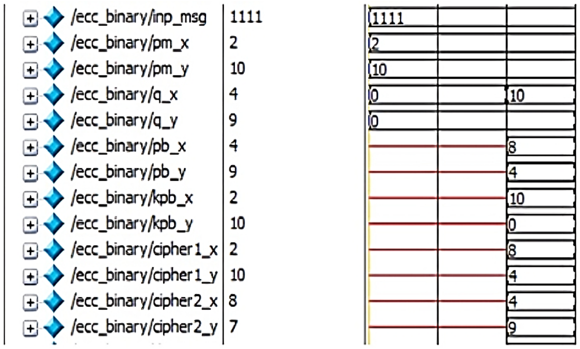
| Messages | |
|---|---|
| /montgo_ptmulchec... | St1 |
| /montgo_ptmulchec... | St0 |
| /montgo_ptmulchec... | 101 |
| /montgo_ptmulchec... | 1101 |
| /montgo_ptmulchec... | 4 |
| /montgo_ptmulchec... | 9 |
| /montgo_ptmulchec... | 4 |
| /montgo_ptmulchec... | 2 |
| /montgo_ptmulchec... | 4 |
| /montgo_ptmulchec... | 2 |

Figure 10i. Montgomery point multiplication.



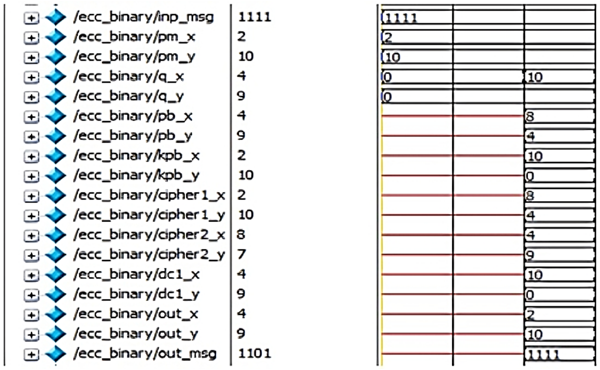
```
Messages
/montgo_ptmulchec... St1
/montgo_ptmulchec... St0
/montgo_ptmulchec... 101
/montgo_ptmulchec... 1101
/montgo_ptmulchec... 4
/montgo_ptmulchec... 9
/montgo_ptmulchec... 4
/montgo_ptmulchec... 2
/montgo_ptmulchec... 4
/montgo_ptmulchec... 2
```

Figure 10j. Karatsuba point multiplication.



```
/ecc_binary/inp_msg      1111
/ecc_binary/pm_x         2
/ecc_binary/pm_y         10
/ecc_binary/q_x          4
/ecc_binary/q_y          9
/ecc_binary/pb_x         4
/ecc_binary/pb_y         9
/ecc_binary/kpb_x        2
/ecc_binary/kpb_y        10
/ecc_binary/cipher1_x    2
/ecc_binary/cipher1_y    10
/ecc_binary/cipher2_x    8
/ecc_binary/cipher2_y    7
```

Figure 10k. ECC encryption.



```
/ecc_binary/inp_msg      1111
/ecc_binary/pm_x         2
/ecc_binary/pm_y         10
/ecc_binary/q_x          4
/ecc_binary/q_y          9
/ecc_binary/pb_x         4
/ecc_binary/pb_y         9
/ecc_binary/kpb_x        2
/ecc_binary/kpb_y        10
/ecc_binary/cipher1_x    2
/ecc_binary/cipher1_y    10
/ecc_binary/cipher2_x    8
/ecc_binary/cipher2_y    7
/ecc_binary/dc1_x        4
/ecc_binary/dc1_y        9
/ecc_binary/out_x        4
/ecc_binary/out_y        9
/ecc_binary/out_msg      1101
```

Figure 10l. ECC decryption.

A $n_b$ and k are random integers that are used during the encryption and decryption of data. A Look-up table has the points having x and y-coordinates within a range [0, *p*-1] that satisfies the chosen ECC curve. Similarly, for each binary input, the data is mapped to the lookup table that is initialized and the data of the x and y-coordinates are stored to the respective points that are considered. Initially for the input message the s and u coordinate are mapped to the $P_m(x,y)$. In Figure 10f, the input points are (a,b) (c,d) and the resultant point addition is available in $(x_3, y_3)$.

During transmission of the encrypted data and during decryption the point must be mapped again to the binary stream. This is done using the inverse look-up module. Using the extended Euclidean algorithm, the multiplicative inverse is found. Point multiplication calculates the value of *kP* where k is of the binary form $(k_{t-1}, k_{t-2},\ldots k_0)_2$ and P is a point on the ECC curve. To encrypt a message of binary form, it is mapped to a point in the elliptic curve; this point is considered as $P_m$. A random generator point G is considered for encryption as shown in Figure 10k according to C1 (x, y) = k * G (x, y); C2 (x, y) = Pm (x, y) + [k * Pb (x, y)]. The two-cipher text point calculated as C1 and C2 form the encrypted data. Reconstructed plaintext is found in the simulation shown in Figure 10l according to $P_m = C_2$ (x, y) – [$n_b$ * $C_1$ (x, y)].

This yields an ordered pair of (x, y) which is further mapped with the inverse lookup table to generate the binary stream of data. The complete ECPM is completed in 229.63k cycles for Karatsuba based point multiplication without pipeline and 223.38k cycles for pipelined version. The Karatsuba pipelined ECC implemented on Virtex-7 occupies 8.42k slices, takes 223.38k clock cycles for process completion, the maximum operating frequency is 232.60MHz, and the process takes 0.937ms to complete as shown in Table 2. The area-delay product and throughput are 7.89 and 262.30kbps, respectively. The pipelined ECC has 32.45% efficiency. The pipelined ECC has 10.99% and 8.36% improvement in throughput and efficiency concerning non-pipelined implementation. This is due to the increased frequency of operation (238.40MHz) of pipelined architecture. It leads to an additional 2.43% overhead on the area, but the area-delay overhead is reduced by 7.71%.

Table 3 provides the performance comparison improvement of Karatsuba PM based ECC concerning Montgomery and binary Left to Right algorithm. It is observed that the performance has improved in the range of 0.47-2.84%, 1.60-4.08%, 2.49-11.23%, 4.45-16.21%, 4.16-15.96% and 4.66-19.35% in number of slices, clock cycles, maximum frequency, area delay product (AT/B), Throughput and efficiency, respectively.

Table 4 shows the quality attributes of the ECC implementation in FPGA. For implementation considerations, various FPGA families including Virtex-4, 5, 6, 7 and Kintex-7 are employed in this research. Except for (Shor, 1999; Kamaraj & Marichamy, 2022), where 192 and 193 bits, respectively, are taken into consideration, most researchers created their designs for the 256-bit size ECC. The key major performance parameters considered for the analysis are area, throughput, efficiency, maximum operating frequency, number of clock cycles, and area-delay product.

Table 4 proves that higher operating frequency reduces the number of clock cycles needed, speeds up completion times, and improves throughput. The area, speed, and throughput of the proposed pipelined Karatsuba multiplier are all optimized. In comparison to all the earlier works, the efficiency of the suggested method is better. Furthermore, the efficiency of the multiplication process has increased up to 32.45% through pipelining. Table 3 does not include the older

FPGA research employing the Virtex-I Pro, Virtex-E, and Spartan 4 processors for comparison purposes because of their high-power consumption and constrained input/output capabilities. The better the communication the smooth intelligent transport systems are possible (Kamaraj et al., 2016).

## 4. Conclusions

The establishment of vehicle communication networks currently is a pertinent solution to guarantee the security of road users and improve traffic flow. Sharing information between vehicles (V2V) and between vehicles and infrastructure (V2I) and networks (V2N) is consequently important. The authentication and secrecy of data exchanged between entities are one of the issues that have developed in these networks. This paper develops an area efficient, high-speed ECC processor based on the secp256k1 Koblitz curve. It provides prime field-based 256-bit point addition and multiplication. In this research, Karatsuba multiplication is used to enhance performance. The pipeline technology accelerates multiplication much more. For the 256-bit prime field, the proposed pipelined Karatsuba multiplier based ECC processor is implemented on the Xilinx Virtex-7 FPGA. The implemented pipelined processor performs a single 256-bit multiplication in 0.937ms with a maximum clock frequency of 238.40MHz, which provides 273.21kbps throughput and occupies 8.42k slices in Virtex-7 FPGA. Incorporating pipeline in scalar multiplication improves the maximum clock frequency up to 7.97%, and reduces time by 9.90%, which in turn increases the throughput by 10.99%. The pipeline has an additional area overhead of 2.43% in Virtex-7. Also, the Karatsuba multiplier based ECC is providing better than the Montgomery and Left to Right algorithm based ECC. In terms of area, clock cycle count, operating frequency, time, area-delay product, throughput, efficiency, and security, the pipelined Karatsuba multiplier based ECC processor performs better than the existing designs. Based on the proposed ECC processor's overall performance, it is possible to conclude that both resource-constrained applications and V2X communication can utilize it with reliability.

Table 2. Performance of proposed ECC processor.

| Algorithm | Platform | Number of bits | Number of Slices (k) | Clock Cycles (k) | Maximum Frequency (MHz) | | Time (ms) | Area Delay Product | AT/B | Throughput (kbps) | Efficiency |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Montgomery | Virtex-7 (woP) | 256 | 8.46 | 239.39 | 198.50 | | 1.21 | 10.20 | 0.0399 | 212.27 | 25.09 |
| Binary Left to Right | | | 8.26 | 234.14 | 216.80 | | 1.08 | 8.92 | 0.0348 | 237.04 | 28.70 |
| Karatsuba | | | 8.22 | 229.63 | 220.80 | | 1.04 | 8.55 | 0.0334 | 246.15 | 29.95 |
| Montgomery | Virtex-7 (wP) | | 8.62 | 232.97 | 228.40 | | 1.02 | 8.79 | 0.0343 | 250.98 | 29.12 |
| Binary Left to Right | | | 8.46 | 227.02 | 232.60 | | 0.98 | 8.26 | 0.0323 | 262.30 | 31.00 |
| Karatsuba | | | 8.42 | 223.38 | 238.40 | | 0.94 | 7.89 | 0.0308 | 273.21 | 32.45 |
| Percentage of Improvement | woP Vs wP | | 1.89 | -2.68 | 15.06 | | 15.42 | 13.82 | 13.82 | 18.24 | 16.04 |
| | | | 2.42 | 3.04 | 7.29 | | 9.63 | 7.44 | 7.44 | 10.66 | 8.04 |
| | | | 2.43 | 2.72 | 7.97 | | 9.90 | 7.71 | 7.71 | 10.99 | 8.36 |
| Karatsuba vs Montgomery | Virtex-7 (woP) | | 2.84 | 4.08 | 11.23 | | 13.76 | 16.21 | 16.21 | 15.96 | 19.35 |
| Karatsuba vs Binary | | | 0.48 | 1.93 | 1.85 | | 3.70 | 4.17 | 4.17 | 3.85 | 4.35 |
| Karatsuba vs Montgomery | Virtex-7 (wP) | | 2.32 | 4.12 | 4.38 | | 8.14 | 10.27 | 10.27 | 8.86 | 11.44 |
| Karatsuba vs Binary | | | 0.47 | 1.60 | 2.49 | | 4.00 | 4.45 | 4.45 | 4.16 | 4.66 |

Table 3. Performance comparison of Karatsuba ECC and other ECC.

| Improvement of Karatsuba w.r.t. | Pipeline | Number of Slices (%) | Clock Cycles (%) | Maximum Frequency (%) | Time (%) | Area Delay Product | AT/B | Throughput (%) | Efficiency |
|---|---|---|---|---|---|---|---|---|---|
| Montgomery | No Pipeline | 2.84 | 4.08 | 11.23 | 13.76 | 16.21 | 16.21 | 15.96 | 19.35 |
| Binary Left to Right | | 0.48 | 1.93 | 1.85 | 3.70 | 4.17 | 4.17 | 3.85 | 4.35 |
| Montgomery | Pipeline | 2.32 | 4.12 | 4.38 | 8.14 | 10.27 | 10.27 | 8.86 | 11.44 |
| Binary Left to Right | | 0.47 | 1.60 | 2.49 | 4.00 | 4.45 | 4.45 | 4.16 | 4.66 |

Table 4. Performance comparison of proposed ECC.

| Ref | Year | Platform | Number of bits | Number of Slices (k) | Clock Cycles (k) | Maximum Frequency (MHz) | Time (ms) | Area Delay Product | $^{\beta}$AT/B | Throughput (kbps) | $^{\beta}$Efficiency |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Ours | 2023 | Virtex-7 (woP) | 256 | 8.46 | 239.39 | 198.50 | 1.21 | 10.20 | 0.0399 | 212.27 | 25.09 |
| | | | | 8.26 | 234.14 | 216.80 | 1.08 | 8.92 | 0.0348 | 237.04 | 28.70 |
| | | | | 8.22 | 229.63 | 220.80 | 1.04 | 8.55 | 0.0334 | 246.15 | 29.95 |
| | | Virtex-7 (wP) | | 8.62 | 232.97 | 228.40 | 1.02 | 8.79 | 0.0343 | 250.98 | 29.12 |
| | | | | 8.46 | 227.02 | 232.60 | 0.98 | 8.26 | 0.0323 | 262.30 | 31.00 |
| | | | | 8.42 | 223.38 | 238.40 | 0.94 | 7.89 | 0.0308 | 273.21 | 32.45 |
| (Kamaraj & Marichamy, 2022) | 2023 | Virtex-7 | 256 | 8.23 | 232.2 | 192.5 | 1.2062 | 9.927 | 0.03878 | 212.23 | 25.79 |
| | | | | 8.46 | 212.4 | 226.8 | 0.9365 | 7.922 | 0.03095 | 273.36 | 32.31 |
| | | Virtex-6 | | 8.82 | 232.2 | 186.2 | 1.247 | 10.99 | 0.04296 | 205.29 | 23.27 |
| | | | | 9.12 | 212.4 | 216.8 | 0.98 | 8.93 | 0.03491 | 261.30 | 28.65 |
| (Islam et al., 2020) | 2020 | Virtex-7 | 256 | 6.5 | 198.7 | 104.39 | 1.9 | 12.35 | 0.0482 | 134.49 | 20.69 |
| | 2020 | Virtex-6 | 256 | 6.6 | 198.7 | 93.23 | 2.13 | 14.05 | 0.0549 | 120.12 | 18.20 |
| (Islam et al., 2019) | 2019 | Virtex-7 | 256 | 8.9 | 262.7 | 177.7 | 1.48 | 13.17 | 0.051 | 173.2 | 19.46 |
| | 2019 | Virtex-6 | 256 | 9.2 | 262.7 | 161.1 | 1.63 | 15.00 | 0.059 | 157.00 | 17.06 |
| (Shah et al., 2019) | 2018 | Virtex-6 | 256 | 65.6 | 153.2 | 327 | 0.47 | 30.83 | 0.120 | 546.42 | 8.33 |
| (Hu et al., 2018) | 2018 | Virtex-4 | 256 | 9.4 + 14DSPs | 610 | 20.44 | 29.84 | 280.5 | 1.096 | 8.58 | 0.91 |
| (Hossain et al., 2017) | 2017 | Kintex-7 | 256 | 11.3 | 397.3 | 121.5 | 3.27 | 63.95 | 0.144 | 78.28 | 6.92 |
| (Asif et al., 2017) | 2017 | Virtex-7 | 256 | 24.2 | 215.9 | 72.9 | 2.96 | 71.63 | 0.280 | 1816.20 | 3.57 |
| (Kodali et al., 2015) | 2017 | Virtex-4 | 193 | 12 | 459.9 | 36.5 | 12.6 | 151.2 | 0.783 | 20.32 | 1.69 |
| (Javeed et al., 2017) | 2017 | Virtex-4 | 256 | 20.6 | 191.6 | 49 | 3.91 | 80.55 | 0.315 | 65.47 | 3.18 |
| (Javeed & Wang, 2017) | 2016 | Virtex-4 | 256 | 13.2 | 200 | 40 | 5 | 66 | 0.258 | 51 | 3.88 |
| (Javeed & Wang, 2016) | 2016 | Virtex-4 | 192 | 35.7 | 207.1 | 70 | 2.96 | 105.67 | 0.550 | 86.53 | 2.42 |
| (Marzouqi et al., 2015) | 2016 | Virtex-5 | 256 | 8.7 | 361.6 | 160 | 2.26 | 19.66 | 0.077 | 113.27 | 13.02 |

## Conflict of interest

The author does not have any types of conflict of interest to declare.

## Acknowledgements

Author acknowledged the support provided Mepco Schlenk Engineering College, Sivakasi, Tamilnadu.

## Funding

The author did not receive any sponsorship to carry out the research reported in the present manuscript.

## References

Asif, S., Hossain, M. S., & Kong, Y. (2017). High-throughput multi-key elliptic curve cryptosystem based on residue number system. *IET Computers & Digital Techniques*, *11*(5), 165-172.
https://doi.org/10.1049/iet-cdt.2016.0141

Arunachalam, K., & Perumalsamy, M. (2022). FPGA implementation of time-area-efficient Elliptic Curve Cryptography for entity authentication. *Informacije MIDEM*, *52*(2), 89-103.
https://doi.org/10.33180/InfMIDEM2022.203

Azarderakhsh, R., Elkhatib, R., Koziel, B., & Langenberg, B. (2021). Hardware deployment of hybrid PQC: SIKE+ ECDH. In *Security and Privacy in Communication Networks: 17th EAI International Conference, SecureComm 2021, Virtual Event, September 6–9, 2021, Proceedings, Part II 17* (pp. 475-491). Springer International Publishing.
https://doi.org/10.1007/978-3-030-90022-9_26

Bernstein, D. J. (2006). Curve25519: new Diffie-Hellman speed records. In *Public Key Cryptography-PKC 2006: 9th International Conference on Theory and Practice in Public-Key Cryptography, New York, NY, USA, April 24-26, 2006. Proceedings 9* (pp. 207-228). Springer Berlin Heidelberg.
https://doi.org/10.1007/11745853_14

ECC Decision (08)01. (2020). The harmonised use of safety-related intelligent transport systems (ITS) in the 5875~5935MHz frequency band. Approved 14 March 2008, latest amendment on 06 March 2020.

ECC Recommendation (08)01 (2020). Use of the band 5855~5875MHz for intelligent transport systems (ITS). Approved 21 February 2008, latest amendment on 06 March 2020.

ETSI (2021). Intelligent Transport Systems (ITS); Security; Security header and certificate formats. Technical Specification (TS) TS 103 097. European Telecommunications Standard Institute (ETSI). Version 2.1.1.

Eyupoglu, C. (2015). Performance analysis of karatsuba multiplication algorithm for different bit lengths. *Procedia-Social and Behavioral Sciences*, *195*, 1860-1864.
https://doi.org/10.1016/j.sbspro.2015.06.420

Gaur, V., Singh, B., Deepak, M. A., & Mishra, N. (2019). Estimation of Various Scalar Multiplication Algorithms in ECC. International Journal of Innovative Technology and Exploring Engineering (IJITEE), 8(6S4): 183-186.

Hossain, M. S., Kong, Y., Saeedi, E., & Vayalil, N. C. (2017). High-performance elliptic curve cryptography processor over NIST prime fields. *IET Computers & Digital Techniques*, *11*(1), 33-42.
https://doi.org/10.1049/iet-cdt.2016.0033

Hossain, M. S., Saeedi, E., & Kong, Y. (2016). High-Performance FPGA Implementation of Elliptic Curve Cryptography Processor over Binary Field GF (2^ 163). In ICISSP (pp. 415-422).

Houmer, M., Ouaissa, M., & Ouaissa, M. (2022). Secure authentication scheme for 5g-based v2x communications. *Procedia Computer Science*, *198*, 276-281.
https://doi.org/10.1016/j.procs.2021.12.240

Hu, X., Zheng, X., Zhang, S., Cai, S., & Xiong, X. (2018). A low hardware consumption elliptic curve cryptographic architecture over GF (p) in embedded application. *Electronics*, *7*(7), 104.
https://doi.org/10.3390/electronics7070104

IEEE 1609 Working Group. (2016). IEEE standard for wireless access in vehicular environments-security services for applications and management messages. IEEE Std, 1609.

Imran, M., Rashid, M., Jafri, A. R., & Kashif, M. (2019). Throughput/area optimised pipelined architecture for elliptic curve crypto processor. IET Computers & Digital Techniques, 13(5), 361-368.
https://doi.org/10.1049/iet-cdt.2018.5056

Ionita, D. M., & Simion, E. (2021). FPGA Offloading for Diffie-Hellman Key Exchangeusing Elliptic Curves. *Cryptology ePrint Archive*.
https://eprint.iacr.org/2021/065

Islam, M. M., Hossain, M. S., Hasan, M. K., Shahjalal, M., & Jang, Y. M. (2019). FPGA implementation of high-speed area-efficient processor for elliptic curve point multiplication over prime field. *IEEE Access*, *7*, 178811-178826.
https://doi.org/10.1109/ACCESS.2019.2958491

Islam, M. M., Hossain, M. S., Hasan, M. K., Shahjalal, M., & Jang, Y. M. (2020). Design and implementation of high-performance ECC processor with unified point addition on twisted Edwards curve. *Sensors*, *20*(18), 5148.
https://doi.org/10.3390/s20185148

Javeed, K., & Wang, X. (2016). FPGA based high speed SPA resistant elliptic curve scalar multiplier architecture. International Journal of Reconfigurable Computing, 2016.

Javeed, K., & Wang, X. (2017). Low latency flexible FPGA implementation of point multiplication on elliptic curves over GF (p). *International Journal of Circuit Theory and Applications*, *45*(2), 214-228.
https://doi.org/10.1002/cta.2295

Javeed, K., Wang, X., & Scott, M. (2017). High performance hardware support for elliptic curve cryptography over general prime field. *Microprocessors and Microsystems*, *51*, 331-342.
https://doi.org/10.1016/j.micpro.2016.12.005

Kamaraj, A., Radha, K., Priyanka, M., & Punitha, M. (2016). Intelligent transport system using integrated GPS optimized reader. In *2016 Second International Conference on Science Technology Engineering and Management (ICONSTEM)* (pp. 332-336). IEEE.
https://doi.org/10.1109/ICONSTEM.2016.7560972

Khan, Z. U. A., & Benaissa, M. (2013). Low area ECC implementation on FPGA. In *2013 IEEE 20th International Conference on Electronics, Circuits, and Systems (ICECS)* (pp. 581-584). IEEE.
https://doi.org/10.1109/ICECS.2013.6815481

Khan, Z. U., & Benaissa, M. (2016). 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.
https://doi.org/10.1109/TVLSI.2016.2574620

Kodali, R. K., Boppana, L., Saikiran, A. V., & Amanchi, C. N. (2015). FPGA implementation of multiplication algorithms for ECC. In *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 549-554). IEEE.
https://doi.org/10.1109/ICACCI.2015.7275666

KS, T., & Elavarasi, E. (2013), High Speed Efficient Karatsuba-Ofman Pipelined Multiplier for Low Contrast Image Enhancement, International Journal of Engineering and Advanced Technology, 2(5): 476-478.

Marzouqi, H., Al-Qutayri, M., Salah, K., Schinianakis, D., & Stouraitis, T. (2015). A high-speed FPGA implementation of an RSD-based ECC processor. *IEEE Transactions on very large scale integration (vlsi) systems*, *24*(1), 151-164.
https://doi.org/10.1109/TVLSI.2015.2391274

Mishra, S., & Pradhan, M. (2012). Implementation of karatsuba algorithm using polynomial multiplication. *Indian Journal of Computer Science and Engineering, ISSN*, *976*(5166), 88-93.

Morales-Sandoval, M., Flores, L. A. R., Cumplido, R., Garcia-Hernandez, J. J., Feregrino, C., & Algredo, I. (2021). A compact fpga-based accelerator for curve-based cryptography in wireless sensor networks. *Journal of Sensors*, *2021*, 1-13.
https://doi.org/10.1155/2021/8860413

Nadikuda, P. K. G., & Boppana, L. (2022). An area-efficient architecture for finite field inversion over GF (2m) using polynomial basis. *Microprocessors and Microsystems*, *89*, 104439.
https://doi.org/10.1016/j.micpro.2022.104439

Parthasarathy, S. (2012). Multiplicative inverse in mod (m). Algologic Technical Report, (1), 1-3.

Rashid, M., Imran, M., Jafri, A. R., & Al-Somani, T. F. (2019). Flexible architectures for cryptographic algorithms—A systematic literature review. *Journal of Circuits, Systems and Computers*, *28*(03), 1930003.
https://doi.org/10.1142/S0218126619300034

Rashidi, B. (2018). Low-cost and fast hardware implementations of point multiplication on binary edwards curves. In *Electrical Engineering (ICEE), Iranian Conference on* (pp. 17-22). IEEE.
https://doi.org/10.1109/ICEE.2018.8472703

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*(2), 120-126. https://doi.org/10.1145/359340.359342

Rohn, J. (2012). An algorithm for computing all solutions of an absolute value equation. *Optimization Letters*, *6*, 851-856. https://doi.org/10.1007/s11590-011-0305-3

Shah, Y. A., Javeed, K., Azmat, S., & Wang, X. (2019). Redundant-signed-digit-based high speed elliptic curve cryptographic processor. *Journal of Circuits, Systems and Computers*, *28*(05), 1950081. https://doi.org/10.1142/S0218126619500816

Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, *41*(2), 303-332. https://doi.org/10.1137/S0036144598347011

Yoshizawa, T., Singelée, D., Muehlberg, J. T., Delbruel, S., Taherkordi, A., Hughes, D., & Preneel, B. (2023). A survey of security and privacy issues in v2x communication systems. *ACM Computing Surveys*, *55*(9), 1-36. https://doi.org/10.1145/3558052

Zhu, Y., Zhu, M., Yang, B., Zhu, W., Deng, C., Chen, C., ... & Liu, L. (2021). LWRpro: An energy-efficient configurable crypto-processor for module-LWR. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *68*(3), 1146-1159. https://doi.org/10.1109/TCSI.2020.3048395