SEMARAK ILMU PUBLISHING

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Enhanced Scalar Multiplication Algorithm over Prime Field Using Elliptic Net

Norliana Muslim[1], Faridah Yunos[2,*], Zuren Razali[3], Nur Idalisa Norddin[4]

[1] Department of Computer and Communication Technology, Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman, 31900, Kampar, Perak, Malaysia
[2] Laboratory of Cryptography, Analysis and Structure, Institute for Mathematical Research, Universiti Putra Malaysia, 43400, Serdang, Selangor, Malaysia
[3] Department of Computing, Faculty of Communication, Visual Art and Computing, Universiti Selangor, 45600, Bestari Jaya, Selangor, Malaysia
[4] Department of Mathematics and Statistics, Faculty of Computer Science and Mathematics, Universiti Teknologi MARA, 23000, Dungun, Terengganu, Malaysia

ARTICLE INFO

Article history:
Received 22 June 2023
Received in revised form 7 November 2023
Accepted 12 November 2023
Available online 28 February 2024

Keywords:

Division polynomials; Elliptic net; Prime field; Scalar multiplication; Twisted Edwards

ABSTRACT

Scalar multiplication in elliptic curve cryptography is the most expensive and time-consuming operation. The elliptic curve cryptography attracted interest due to the development of modern technology since it could offer the equivalent high level of security with a reduced length of key. Therefore, improving elliptic curve scalar multiplication performance has always been the primary goal of cryptography. In this paper, a novel scalar multiplication algorithm based on the modified double and double add via elliptic net with Karatsuba method was proposed in order to enhance the efficiency of scalar multiplication. In the experimental results, the elliptic net equivalence sequence was applied to the Twisted Edwards curve together with safe curves of numsp384t1 and numsp512t1. At the point operational level, the proposed method reduced the cost of multiplication by 46.15% and 42.30% for double and double add, respectively, when compared to elliptic net using eight blocks method. The proposed double lowered the multiplication cost by 12.5% and the squaring cost by 20% when compared to elliptic net using ten temporary variables method. Following this, proposed double add cost reductions of 6.25% and 20% were obtained to multiplication and squaring. At the field operational level, in comparison to the binary method, the eight-block elliptic net method, and the elliptic net method with ten temporary variables for the 384 bits scenario, the developed scalar multiplication algorithm obtained cost reductions of 57.6%, 31.3%, and 13.2%, respectively. On 512 bits with similar comparison, the designed algorithm exhibited better performance by averages 59.2%, 31.0% and 13.2%. The results signified that the designed algorithm over prime field performed better at the point and field operational levels with larger scalar bit size.

* *Corresponding author.*
E-mail address: faridahy@upm.edu.my

https://doi.org/10.37934/araset.40.2.2235

1. Introduction

The Elliptic Curve Cryptography (ECC) was initiated by Miller [1] and Koblitz [2]. ECC provides equal security to Rivest-Shamir-Adleman scheme but with a smaller key size. For instance, RSA requires a key size of 2048 bits to reach a security level of 112 bits, but ECC only needs 224 bits [3]. Besides that, ECC uses less hardware and memory and faster [4]. It is thus appropriate for memory-constrained devices like smart cards [5]. ECC relies on the scalar multiplication (SM) algorithm for efficiency [6].

SM is the operation to compute the n-multiple of points in the EC group [7]. P and Q are points on the EC, where the process is indicated by n times, with n as a positive integer. That means, Q=nP=P+P+P+⋯+P (for n times). Traditionally, SM relies on the points doubling and addition on an elliptic curve (EC) using the binary method (BM). The digits are scanned bits by bits to perform points double add when digits "1" and point double when digits "0". SM algorithm includes Multiplication (M), Squaring (S), and Inversion (I) field operations. The inversion operation required for the computation of SM using BM results in expensive costs in ECC [9]. Projective coordinates are therefore suggested in the previous studies [10-12] in order to prevent the inversion, but doing so incurs additional costs. To eliminate the inversion, some scholars suggested using Jacobi coordinates [13]. Besides the projective coordinates, recent researchers focused on improving the efficiency of SM by reducing the Hamming weight by converting the binary number to a new representation such as non-adjacent form (NAF) [14-15] or {0,1,3)-NAF [16,17].

Another method that can be used to compute the SM is the double and double add via elliptic net (EN) as proposed by [8]. From the EN perspective, reducing the number of operations in double and double add methods can generate faster SM and consequently efficient ECC. These double and double add operations via EN do not involve inverse operation which is suitable for restricted devices [18] and the cost of double is equal to the double add step in each iteration loop of the algorithm [19]. The EN approach can withstand side-channel attacks and does not rely on the scalar's Hamming weight since these two functions are equal [18-20]. The authors in [21] used the division polynomials to compute small scalar multiplications in Affine and Jacobian coordinates. However, the conversion to Affine coordinate slows the computation and only improves the SM algorithm at the point operational level.

The primary purpose of this research is to improve the performance of SM in Affine coordinate over prime field using elliptic net at the point and field operational levels. The research outcomes are intended to confirm the relationships between the Karatsuba method, equivalent sequences on elliptic nets, and division polynomials of Twisted Edwards curve. Specifically, a new SM algorithm using modified double and double add with Karatsuba method [22] is proposed.

1.1 Twisted Edwards Curve

As a generalisation of the Edwards curve [24], the Twisted Edwards curve over prime field was presented by [23], and the curve has been utilised to accelerate the points addition and doubling on the Edwards curve. The general form of Twisted Edwards curve with $a \neq d$ is represented by $ax^2 + y^2 = 1 + dx^2y^2$. A corresponding Edwards curve's quadratic twist is the basis for every Twisted Edwards curve. Given that the addition formula may be utilised for doubling and that it is complete by defined formula for all inputs, the group law of the Twisted Edwards curve is said to be unified [25]. The addition formulas in affine coordinates for Twisted Edwards curve are given as follows. Let $P = (x_1, x_1), Q = (x_2, x_2)$ and $P + Q = (x_3, x_3)$ be points on the EC. If $P = Q$, then the point doubling is denoted by,

$$x_3 = \frac{2x_1y_1}{ax_1^2+ y_1^2}, y_3 = \frac{y_{1-}^2 ax_1^2}{2-ax_1^2+ y_1^2} \quad (1)$$

If $P \neq Q$, then the point addition is generated by,

$$x_3 = \frac{x_1y_2+y_1x_2}{1+dx_1x_2y_1y_2}, y_3 = \frac{y_1y_2+ ax_1x_2}{1-dx_1x_2y_1y_2} \quad (2)$$

From Eq. (1), the cost of addition point is $7M + 2I$ and the cost for double a point is $2M + 2S + 2I$ as Eq. (2).

1.2 Twisted Edwards Curve's Division polynomials

The Twisted Edwards's division polynomials are defined by $\Psi_n(x, y)$ over F_p and the rational functions $\Psi_n(x, y)$ on the function field of elliptic curve for $n \geq 0$ are denoted by,

$$\Psi_0 = 0, \Psi_1 = 1, \Psi_2 = \frac{(a-d)(1+y)}{2x(1-y)} \quad (3)$$

$$\Psi_3 = \frac{(a-d)^3(a+2ay_1-2dy^3-dy^4)}{(2(1-y))^4} \quad (4)$$

$$\Psi_4 = \frac{2(a-d)^6y(1+y)(a-dy^4)}{x(2(1-y))^7} \quad (5)$$

Recursively, Ψ_n can define the division polynomials for $n \geq 5$ using:

$$\Psi_{2n+1} = \Psi_{n+2}\Psi_n^3 - \Psi_{n-1}\Psi_{n+1}^3 \quad (6)$$

$$\Psi_2\Psi_{2n} = \Psi_n(\Psi_{n+1}\Psi_{n-1}^2 - \Psi_{n-2}\Psi_{n+1}^2) \quad (7)$$

The auxiliary polynomials from the equation for the Twisted Edward curve are represented by,

$$\phi_n = \frac{(1+y)\Psi_n^2}{(1-y)} - \frac{4\Psi_{n-1}\Psi_{n+1}}{(a-d)} \quad (8)$$

$$\omega_n = \frac{2\Psi_{2n}}{(a-d)\Psi_n} \quad (9)$$

The set of Twisted Edward's division polynomials from Eq. (8) and Eq. (9) can be expressed as multiple points as follows:

$$nP = (x_n, y_n) = \left(\frac{\phi_n\Psi_n}{\omega_n}, \frac{\phi_n-\Psi_n^2}{\phi_n-\Psi_n^2}\right) \quad (10)$$

The EDS sequence denoted by $h_0, h_1, \ldots, h_n$ must meets that for all $m > n$, then $h_{m+n}h_{m-n} = h_{m+1}h_{m-1}h_n^2 - h_{n+1}h_{n-1}h_m^2$[26]. Another condition that needs to be satisfied is h_n divides h_m whenever *n* divides *m.* The generalisation of EDS to EN using $w_{m+n}w_{m-n} = w_{m+1}w_{m-1}w_n^2 - w_{n+1}w_{n-1}w_m^2$ has been proposed by [19].

1.3 Binary method

This section reviews the traditional method for SM computation namely as BM algorithm [27].

Algorithm 1. BM
Input: An affine point $P \in E(\mathrm{F}_p)$ and $n = (n_{l-1}, \dots, n_0)_2$.
Output: $nP \in E(\mathrm{F}_p)$.
1. For *i* from *l-2* down to 0 do
2. $Q \leftarrow 2Q$
3. If $n_i = 1$ then
   3.1 $Q \leftarrow P + Q$
4. Return *Q*.

Referring to Algorithm 1, Line 2 denotes the process of double (see Eq. (1)) and Line 3 in the process of addition (see Eq. (2)). In BM, the operation involves is double and double add. The cost of double add is the total cost of double and addition. Thus, the cost of double add is $9M + 2S + 4I$. Let h and l be the scalar Hamming weight and the bit length, respectively.

Proposition 1. For a sufficiently large scalar $n = (n_{l-1}, \dots, n_0)_2$, the complexity of BM in terms of double and double add processes is denoted by

$$C_{BM} = (l-h)double + (h-1)double\ add \tag{11}$$

Proof. From BM Algorithm, it is clear that line 2 executed consecutively for every non-zero digit n_i and $I - 2 \leq i \leq 0$. For every $n_i = 0$, line 2 operates as double process and for every $n_i = 1$, lines 3.1 operate as double add process. When line 2 executes for $I - 2 \leq i \leq 0$, this excludes n_{I-1} from $n = (n_{l-1}, \dots, n_0)_2$ and double add is performed as $h - 1$ times. Since double add operates for every non-zero digit n_i and $I - 2 \leq i \leq 0$, then double is executed as $l - h$ times. Therefore, the overall cost of Algorithm 1 is $C_{BM} = (I-h)double + (h-1)double\ add$.

2. Methodology

The Twisted Edwards curve division polynomial sequences satisfy with the non-linear recurrence relation. Let $P = (\mathrm{x}_1, y_1)$ be a point on the Twisted Edwards curve of the form $ax^2 + y^2 = 1 + dx^2y^2$ over prime field F_p, the rational functions $\Psi_n(x, y)$ on the function field of EC are the division polynomials. For developing the new SM algorithm, the following Lemma pertaining to elliptic divisibility sequence (EDS) is required:

i. *Lemma 1:* Let $\{\Psi_n\}$ denotes the proper EDS over F_p with *p* elements with $\Psi_2 \neq 0$. Then, there exists w_n over F_p which is equivalent to the sequence $\{\Psi_n\}$.
ii. *Proof*: Assume $\{\Psi_n\}$ is defined over F_p. The cube root *c* of Ψ_2^{-1} that lies in F_p can be found. Suppose that $\gcd(p-1, 3) = 1$. To change the Ψ_2 to 1, choose *c* such that $c^3 = \Psi_2^{-1}$ then $c^3 = \sqrt[3]{\Psi_2^{-1}}$. To solve the cube root modulo *p,* use the formula $\sqrt[3]{u} = u^{\frac{2p-1}{3}} \mathrm{mod} p$ [28] then $c = (\Psi_2)^{\frac{1-2p}{3}}$. A sequence W_n by $W_n = c^{n^{2-1}}\Psi_n$ for any integer *n* with $w_2 = 1$. The sequence W_n is an EN over F_p since *c* and Ψ_n belongs to F_p. This completes the proof.

The next EN values can be calculated using Eq. (3), Eq. (4) and Eq. (5) respectively,

$W_0 = 0, W_1 = c^0\Psi_1$
$W_2 = c^3\Psi_2, W_3 = c^8\Psi_3, W_2 = c^{15}\Psi_4$
$W_5 = W_4W_2^3 - W_1W_3^3$

The multiple points denoted by nP via EN Twisted Edwards curve can be calculated as stated below:

Theorem 1: Let w_n defined from Lemma 1 and $W_2 = c^3\Psi_2$. If $P = (\mathrm{x}_1, y_1)$ on Twisted Edwards curve over prime field, then $nP = (\mathrm{x}_n, y_n)$ can be computed as follows:

$$x_n = \frac{(a-d)\sigma_n W_c^2 c}{2W_{2n}} \tag{12}$$

$$y_n = \frac{\sigma_n - W_c^2 c^2}{\sigma_n + W_n^2 c^2} \tag{13}$$

Where

$$\sigma_n = \frac{(1+y)W_n^2 c^2}{1-y} - \frac{4W_{n-1}W_{n+1}}{a-d} \tag{14}$$

Proof: Note that $\Psi_n = c^{1-n^2}W_n$ for any integer *i* and $c^3 = \Psi_2^{-1}$. By Eq. (8) and Eq. (9), ϕ_n was obtained, then by Eq. (10), $nP = (\mathrm{x}_n, y_n)$ can be produced. That means,

$$\phi_n = \frac{(1+y)\Psi_n^2}{1-y} - \frac{4W_{n-1}W_{n+1}}{a-d}$$

$$= \frac{(1+y)W_n^2 c^{2(1-n^2)}}{1-y} - \frac{4W_{n-1}c^{2(1-n^2)}W_{n+1}c^{1-(n+1)^2}}{a-d}$$

$$= \frac{(1+y)c^2 c^{-2n^2}W_n^2}{1-y} - \frac{4c^{-2n^2}W_{n-1}W_{n+1}}{a-d}$$

$$= c^{-2n^2}\left[\frac{(1+y)c^2W_n^2}{1-y} - \frac{4W_{n-1}W_{n+1}}{a-d}\right]$$

Let $\sigma_n = \frac{(1+y)W_n^2c^2}{1-y} - \frac{4W_{n-1}W_{n+1}}{a-d}$ and $\phi_n = \sigma c^{-2n^2}$. So,

$$x_n = \frac{(a-d)\phi_n\Psi_n^2}{2\Psi_{2n}}$$

$$= \frac{(a-d)\sigma c^{-2n^2}W_n^2c^{2(1-n^2)}}{2W_{2n}c^{1-4n^2}} - \frac{(a-d)\sigma c^{-2n^2}W_n^2c^2c^{-2n^2}}{2W_{2n}c^{1-4n^2}}$$

$$= \frac{(a-d)\sigma c^{-4n^2} W_n^2 c^2}{2W_{2n} c^{1-4n^2}}$$

$$= \frac{(a-d)\sigma_n W_n^2 c}{2W_{2n}}.$$

$$y_n = \frac{\phi_n - \Psi_n^2}{\phi_n + \Psi_n^2}$$

$$= \frac{\sigma c^{-2n^2} - W_n^2 c^{2^{(1-n^2)}}}{\sigma c^{-2n^2} + W_n^2 c^{2^{(1-n^2)}}} = \frac{\sigma c^{-2n^2} - W_n^2 c^2 c^{-2n^2}}{\sigma c^{-2n^2} + W_n^2 c^2 c^{-2n^2}}$$

$$= \frac{c^{-2n^2}(\sigma - W_n^2 c^2)}{c^{-2n^2}(\sigma + W_n^2 c^2)}$$

$$= \frac{\sigma_n - W_n^2 c^2}{\sigma_n + W_n^2 c^2)}.$$

The performance of new SM over prime field was assessed throughout the analysis and results phase using point and field operational levels. The number of point operations per scalar was determined by the execution of the elliptic curve SM algorithm. The costs of point operations were assessed and contrasted using double and double add costs. An experiment was carried out using 30 samples with various Hamming weights and fixed bit lengths to further study the relationship between the bit length of the scalar and its cost in the SM approaches. The examined safe Twisted Edwards curves namely nums384t1 and nums512t1 had 15 Hamming weights ranging from 192 to 305 and bit lengths that satisfied the equivalence sequences of 384 and 512 bits. The running time of the field operations was computed using Miracl [28]. For 384-bits, the squaring and inversion were converted to multiplication by multiplying with 0.87 and 14.51 respectively. For 512-bits, the multiplication requires 0.89 and 15.26 respectively.

3. Results

This section highlights the modification of the double and double add functions that required on designing a new SM algorithm. Then, the complexity of the proposed algorithm based on the point and field operational levels are discussed.

3.1 Modified Double and Double Add Algorithm

In order to generate new V_i in double and double add blocks, the Karatsuba-Ofman method [29] was deployed over prime field by setting $S_i = V_{i+1}^2$, $P_i = ((V_i + V_{i+2})^2 - S_i - S_{i+2})/2$, and $R_i = S_i P_i$ for $1 \leq i \leq 4$, in which the outcomes are $V_0 = (S_0 - S_1)(P_0 + P_1) - R_0 + R_1$, $V_1 = (S_0 - S_2)(P_0 + P_2) - R_0 + R_2)\alpha$, $V_2(S_1 - S_2)(P_1 + P_2) - R_1 + R_2$, $V_3((S_1 - S_3)(P_1 + P_3) - R_1 + R_3\alpha$, $V_2 = V_4 = (S_2 - S_3)(P_2 + P_3) - R_2 + R_3)$, $V_5((S_2 - S_4)(P_2 + P_4) - R_2 + R_4)\alpha$ and $V_6 = (S_3 - S_4)(P_3 + P_4) - R_3 + R_4)$. This new V_i had 1M each with an overall cost of 7M in double block. Similar R_i was used for double add block, except that the latter block was set at $\beta = \hat{W}(1)\hat{W}(3), \varepsilon = \hat{W}(2)^2, t_1 = V_3 V_5$, and , $t_2 = V_4^2$ for V_6. Then, the new terms are $V_0 = ((S_0 - S_2)(P_0 + P_2) - R_0 + R_2)\tilde{\alpha}$, $V_2 = ((S_1 - S_3)(P_1 + P_3) - R_1 + R_3)\tilde{\alpha}$,$V_3(S_2 - S_3)(P_2 + P_3) - R_2 + R_3$,$V_4 = ((S_2 - S_4)(P_2 + P_4) -$

$R_2 + R_4)\tilde{\alpha}$,$V_5(S_3 - S_4)(P_3 + P_4) - R_3 + R_4)$ and $V_6 = (t_1\varepsilon - \beta t_2)/V_2$. Each value of V_0 until V_5 required $1M$, but $2M$ was noted in V_6. Hence, the total cost of 8*M* was obtained for double add block. Algorithm 3 shows the modification of the double and double add algorithm using Karatsuba-Ofman Algorithm.

Algorithm 3. New double and double add
Input: A block $V = [V_0, V_1, \dots, V_7]$ centered at k, Boolean add, $\alpha = 1, \beta = W_1W_3, \delta = W_2^2$.
Output: Block V centred at 2k if add=0 and centred at 2k+1 if add=1.
1. For i from 0 to 4 do
   1.1 $S_i = V_{i+1}^2$
2. For i from 0 to 1 do
   2.1 $P_{5i} = V_{5i}V_{5i+2}$
3. For i from 0 to 2 do
   3.1 $P_{1+i} = ((V_{i+1} + V_{i+3})^2 - S_i - S_{i+2})/2$
4. For i from 0 to 4 do
   4.1 $R_i = S_iP_i$
5. For i from 0 to 2 do
   5.1 If $add = 0$ then
      $V_{2i} \leftarrow (S_i - S_{i+1})(P_i + P_{i+1}) - R_i + R_{i+1}$
      $V_{2i+1} \leftarrow ((S_i - S_{i+2})(P_i + P_{i+2}) - R_i + R_{i+2})\alpha$
   5.2 $V_6 \leftarrow (S_3 - S_4)(P_3 + P_4) - R_3 + R_4$
   5.3 Else
      $V_{2i} \leftarrow ((S_i - S_{i+2})(P_i + P_{i+2}) - R_i + R_{i+2})\alpha$
      $V_{2i+1} \leftarrow ((S_{i+1} - S_{i+2})(P_{i+1} + P_{i+2}) - R_{i+1} + R_{i+2})$
   5.4 $t_1 \leftarrow V_3V_5; t_2 \leftarrow V_4^2; t_3 \leftarrow inverse(V_2)$
   5.5 $V_6 \leftarrow (t_3\delta - t_2\beta)t_3$
6. Return V.

Lines 1.1, 2.1, and 3.1 in Algorithm 3 are the temporary variables that are the same as [18] and [30]. This research adds line 4.1 which contains five temporary variables, $5M$ that are used for both double and double add compared to [30] which contains 10 different temporary variables for double and double add. Compared to Algorithm 1 (line 1.2), the cost is reduced from $6M$ to $2M + 4S$ by applying the squaring and multiplication trade-offs inspired by [18] by replacing it with $P_{1+i} = ((V_i + V_{i+2})^2 - S_i - S_{i+2})/2$ for $1 \leq i \leq 4$. Lines 5.1 and 5.2 utilized the new formula (see Eq. (12) and Eq. (13)). Line 5.1 denotes the process of double which produces a block centred at 2K while line 5.2 represents the double add process produces a block centred at 2K+1.

In comparison to Algorithm 1, a reduced cost was obtained from 20*M* to 10*M* with five temporary variables. Furthermore, as α=1 existed in double and double add processes, this reduces 3*M* in each iteration. Lines 5.2 and 5.5 is the last terms in the EN generated with a cost 1*M* for double and double add respectively. The cost of Line 6.2 could be ignored because t_1 and t_2 must be calculated in the next iteration to update the block and become S_3 and P_3, while the cost of t_3 could be removed with binary inverse algorithm. Thus, the cost of Algorithm 3 is $14M + 8S$ for double and $15M + 8S$ for double add operation. In comparison to [8], the proposed method obtained 46.15% and 42.30% multiplication cost reductions for double and double add, respectively. In comparison to [30], the proposed double reduced the multiplication cost by 12.5% and squaring cost by 20%. This was followed by 6.25% and 20% cost reductions in multiplication and squaring via proposed double add.

Now, the proposed double and double add is used to create a new design for the SM algorithm as depicted in Algorithm 4.

Algorithm 4: New SM algorithm via EN

Input: $n = (n_{l-1}, \dots, n_0)_2$, *with* $n_l = 1$, $P \in E(\mathrm{F}_p)$, $a, d \in E$, $K = \hat{\mathrm{W}}(2), L = \hat{\mathrm{W}}(3), M = \hat{\mathrm{W}}(4), \tilde{\alpha} = \hat{\mathrm{W}}(2)^{-1}, A = (1-y)^{-1}, B = (a-d)^{-1}$, *c such that* $C^3 = \tilde{\alpha}, c^{2(1-n^2)}, S = c^{-2n^2}$ and $T = c^{1-4n^2)}$.

Output: (x_n, y_n).

1. $V \leftarrow [-K, -1, 0, 1, L, L, M]$
2. *For i from* $I-1$ *down to 0 do*
   - *2.1 If* $n_i = 0$ *then*
     - $V_{2i} \leftarrow (S_i - S_{i+1})(P_i + P_{i+1}) - R_i + R_{i+1}$
     - $V_{2i+1} \leftarrow ((S_i - S_{i+2})(P_i + P_{i+2}) - R_i + R_{i+2})\alpha$
     - $V_6 \leftarrow (S_3 - S_4)(P_3 + P_4) - R_3 + R_4$.
   - *2.2 Else*
     - $V_{2i} \leftarrow ((S_i - S_{i+2})(P_i + P_{i+2}) - R_i + R_{i+2})\tilde{\alpha}$
     - $V_{2i+1} \leftarrow (S_{i+1} - S_{i+2})(P_{i+1} + P_{i+2}) - R_{i+1} + R_{i+2}$
   - $t_1 \leftarrow V_3 V_5$
   - $t_1 \leftarrow V_4^2$
   - $t_3 \leftarrow inverse(V_2)$
   - $V_6 \leftarrow (t_3\delta - t_2\beta)t_3$
3. $V \leftarrow [V_0, V_1, V_2, V_3, V_4, V_5, V_6, V_7]\, V \leftarrow [V_0, V_1, V_2, V_3, V_4, V_5, V_6, V_7]$
4. $S_0 \leftarrow V_2^2; S_1 \leftarrow V_3^2; S_2 \leftarrow V_4^2$
5. $J \leftarrow V_3(V_5 S_0 - V_1 S_2)$
6. $K \leftarrow [(V_2 + V_4)^2 - S_0 - S_2]/2$
7. $L \leftarrow ABS_1 R - 4KC$
8. $M \leftarrow 2J$
9. $N \leftarrow L + S_1 R$
10. $O \leftarrow inverse(MN)$
11. $x_n \leftarrow (a-d)gLS_1 ON$
12. $y_n \leftarrow (L - S_1 R)OM$
13. *Return* (x_n, y_n).

From Algorithm 4, the generation of EN block is computed with double or double add of sequences in lines 3 or 4. This process takes $l-1$ time based on the Boolean values '0' or '1'. The process of lines 3 and 4 are based on Algorithm 2 which means for every zero Boolean processes double is executed and the nonzero Boolean process double add is computed. The double add is executed for $h-1$ times while the double add is for $l-h$ times. Lines 5 until 11 are the variables calculated using the EN value from the last block generated n_0 based on the Eq. (14) to Eq. (16) with cost 7*M*+4*S*. The inversion in line 11 is computed using the binary inversion algorithm [31] which replaces the inversion with cheaper shifts divisions by 2 and subtraction [32]. Thus, the overall cost of Algorithm 5 uses double to illustrate the double process in line 3 and double add to illustrate the double add process in line 4 and CL_1 to illustrate the cost of multiple points (see lines 5-13). The point operational cost of the proposed algorithm is stated in the following proposition:

Proposition 1. For a sufficiently large scalar $n = (n_{l-1}, \dots, n_0)_2$, the complexity of Algorithm 4 in terms of double, double add processes is denoted by

$$C_{Proposed} = (l-h)\ \text{double}\ +\ (h-1)\ \text{double add} + CL_1 \quad (15)$$

Proof. From Algorithm 4, the required number of doubles add process depends on Hamming weight of the scalar denoted by h. Since the first digit of the scalar n_{l-1} will not be counted then the double add process is $h-1$ times while the remainder of the bit length after subtracting by the hamming weight is a double process. Thus, the double process is executed for $l-h$ times. The cost of multiple points formula using Twisted Edwards curve is denoted by CL_1. Therefore, the cost of the proposed algorithm is the total of double, double add process, and SM where the cost is $C_{Proposed} = (l-h)\text{double}\ +\ (h-1)\ \text{double add} + CL_1$. Recall back the complexity of ENPM algorithm proposed by [30] is $C_{[30]} = (l-1)double + CL_0$ with $1\ double = 16M + 10S$ and $CL_0 = 8M + 3S + 1I$.

An experimental data is developed using 30 sets of data with 15 different Hamming weights for 384 and 512-bit sizes. The following is a description of the running environment required for this research: 8 GB of memory and a 1.80 GHz Intel Core i-7 8565 CPU. The converted running time for one multiplication for 384-bits is $5.8\times10^{-7}s$, while for 512-bits is $8.5\times10^{-7}s$.

3.2 Point Operation Cost

The cost of point operation for the proposed algorithm is assessed using Eq. (15). Then, the cost was compared to other proposed methods in the literature. Table 1 summarizes the cost of point operations using various SM methods.

Table 1
Point operational cost

Method	384-bits	512-bits
[27]	191 double add +192 double	255 double add +256
[8]	383 double + CL_0	511 double + CL_0
[30]	383 double + CL_0	511 double + CL_0
This work	191 double add +192 double + CL_1	255 double add + 256 double+CL_1

Referring to Table 1, on average cases, for 384-bits, the Hamming weight is 192 while for 512-bits, the Hamming weight is 256. Note that, the point operational cost for [26] was computed using Eq. (11), while the cost for [8] and [29] was evaluated based on $C = (l-1)\ double + CL_0$ where $CL_0 = 8M + 3S + 1I$ is the cost of multiple points formulas on a short Weierstrass curve.

3.3 Field Operational Cost

Referring to Table 1, the costs of field operations over prime field for 384 and 512 bits denoted by CF_{384} and CF_{512}, respectively, were computed by plugging in $1\ double = 14M + 8S$, $1\ double\ add = 15M + 8MS$, and $CL_1 = 14M + 4S$. That means,

$$\begin{aligned} CF_{384} &= 192DBL + 191DBLADD + CL_1 \\ &= 192(14M + 8S) + 191(15M + 8S) + 14M + 4S \\ &= 2688M + 1536S + 2865M + 1528S + 14M + 4s \\ &= 5567M + 3068S. \end{aligned}$$

Similar computation can be conducted for 512 bits and Table 2 summarises the costs of field operations over prime field using various SM methods. Notably, the proposed method over prime field decreased the number of field inversions by 100% for 384- and 512-bit lengths.

Table 2
Field operation cost

Method	384-bits	512-bits
[27]	2103*M* + 766*S* + 1148*I*	2807*M* + 1022*S* + 1532*I*
[8]	9966*M* + 2301*S* + 1*I*	13294*M* + 3069*S* + 1*I*
[30]	6136*M* + 3833*S* + 1*I*	8184*M* + 5113*S* + 1*I*
This work	5567*M* + 3068*S*	7423*M* + 4092*S*

From Table 2, the proposed SM optimized the cost by $14M + 8S$ using equivalent sequence properties, multiplication-squaring trade-offs, and seven terms of EN block with the Karatsuba method. Recall back that the bit lengths that satisfied Lemma 1 were 384 and 512 bits. The calculated number of multiplications relied on Tables 1 and 2. Consider a scalar n with $l - 384$ and $h - 192$. The total number of multiplications for 384 bits in the proposed SM denoted by TM_{384} was obtained using $1S = 0.87M$, as listed in the following:

$$\begin{aligned} TM_{384} &= 5567M + 3068S \\ &= \big(5567M + 3068(0.87)\big)M \\ &= 8236.16M. \end{aligned}$$

Similar computation was made for 512 bits but using $1S = 0.89M$ and these values were compared with [27,8,30] as shown in Figure 1.

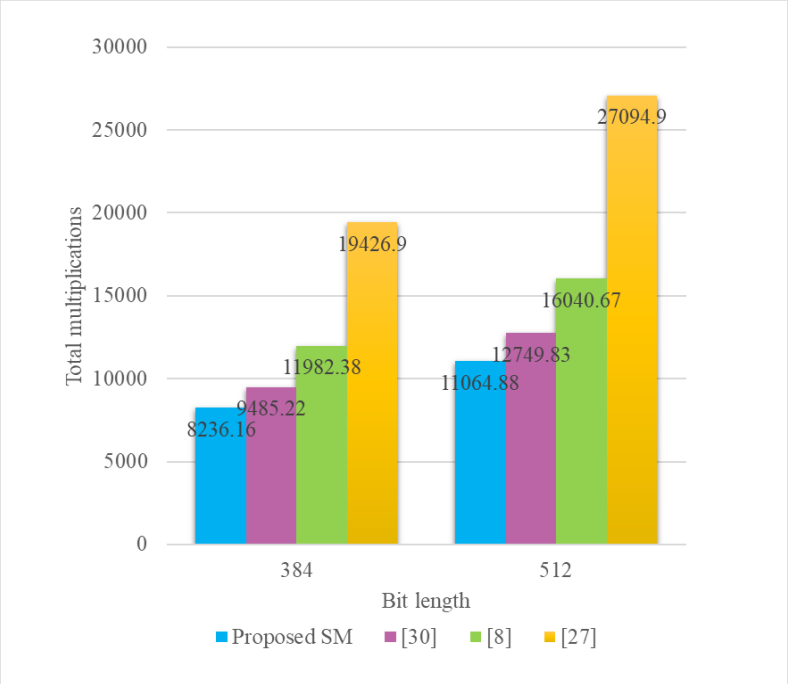

Fig. 1. Estimated number of multiplications

According to Figure 1, the proposed SM is far more efficient than [27,8,30]. For 384-bits, [27,8,30] need 19426.90*M*, 11982.38*M*, and 9485.22*M* respectively, while the proposed SM needs 8236.22*M*.

When the bit size increased to 512-bits, the proposed SM only needs 11064.88M compared to 27094.9*M* of [27], 16040.67*M* of [8], and 12749.83*M* of [30]. Furthermore, the cost for [27] and the proposed SM is increased when the Hamming weight and bit size increase. For the EN method, the cost does not change when the Hamming increase in the same bit size but the cost increase as the bit size increase. Overall, the SM cost for all methods is increased when the bit size increased. The running time for 384 bits denoted by RT_{P384} was determined by converting the total multiplications in Fig. 1 using $1M = 5.8x10^{-7}s$. That means,

$$RT_{P384} = 8236M(5.8x10 - 7)s$$
$$= 0.00478s.$$

Similar computations can be conducted for 512 bits using $1M = 8.5x10^{-7}s$ for related SM approaches and Table 3 compares the SM complexity in terms of running time.

Table 3
Running time for SM methods in seconds (s)

Method	384 bits	512 bits
[27]	0.01127	0.02303
[8]	0.00695	0.01363
[30]	0.00550	0.01084
This work	0.00478	0.00941

From Table 3, at 384-bits, the proposed method speeded up the running times by 57.6% compared to [27], 31.3% compared to [8], and 13.2% compared to [30]. On similar comparison, the percentages of speed up for 512-bits were computed to 59.2%, 31.0%, and 13.2%, respectively. The cost of the proposed algorithm was then compared with method in [27] in order to determine the minimal Hamming weight necessary to make the proposed method more effective than BM over prime field [27]. However, no comparison was made between [8] and [30].

By substituting $1\,double = 14M + 8S$, $1\,double\,add = 15M + 8MS$, and $CL_1 = 14M + 4S$ into Eq. (15) gives the following:

$$C_{Proposed} = (l - h)\,(14M + 8S)\; +\; (h - 1)\,(15M + 8MS) + 14M + 4S \tag{16}$$

Plugging in $1\,double = 2M + 2S + 2I$ and $1\,double\,add = 9M + 2S + 4I$ into Eq. (11), the following is obtained:

$$C_{BM} = (l - h)(2M + 2S + 2I\,) + (h - 1)(9M + 2S + 4I) \tag{17}$$

By comparing Eq. (16) and Eq. (17), it can be shown that the proposed algorithm over the prime field is more cost-effective than [27] in terms of operational cost. Suppose that $C_{Proposed} < C_{BM}$. For l = 384 with $1M = 5.8x10^{-7}s$, the average case was computed such that

$$(l - h)\,(20.96M) +\; (h - 1)\,(21.96M) + 17.48M < (l - h)(32.76M) + (h - 1)(68.78M)$$
$$12.1568\,l + 0.58\,h - 2.5984 < 19.0008\,l + 20.8916\,h - 39.8924$$
$$h > -127.$$

Similar average case can be computed for l = 512 with $1M = 8.5\text{x}10^{-7}s$ and $h > -182$ was obtained. Since the required minimum Hamming weight is negative, then this depicted that the proposed SM algorithm over the prime field outperforms [27] in the average scenario. For the worst-case scenario that means $h = l$ and when substituting l = 384 and l = 512, the proposed over prime field also obtained better cost than [27]. The results of this study indicated that in each different bit lengths, the proposed SM attained better cost than [27] when $h > -127$ and $h > -182$. Surprisingly, the best performance was obtained by the proposed SM method when $h = l$.

4. Conclusions

The choice of SM algorithm directly impacts the security of cryptographic systems, particularly in ECC. The proposed SM is one of robust algorithm that resists to side-channel attack and its vulnerability is imperative to ensure the confidentiality and integrity of sensitive information in engineering system. The enhanced SM algorithm has been designed using modified double and double add with seven terms of EN. At the point operational level, the cost of the new double and double add was evaluated to 14*M*+8*S* and 15*M*+8*S* which saved 2*M*+2*S* in double and saved 1*M*+2*S* in double add compared to the latest reported in the literature. On the other hand, the proposed double obtained the cost reduction by 12.5% in multiplication, 20% in squaring, while double add attained 6.25% in multiplication and 20% cost reductions in squaring. The improvement started by implementing the equivalent sequence into Twisted Edwards division polynomials. The equivalent sequence where the scalar had binary representation was applied to create the new explicit formulas upon Twisted Edwards curve in the designed SM algorithm.

The experimental data for the prime field with 30 samples were developed under the assumption that the 384 and 512 bits were fixed, and that the average Hamming weight varied between 192 and 305. The performance of the designed algorithm by bit length, Hamming weight, and running time was evaluated using the data. The experimental results from the field operation analysis upon the Twisted Edwards curve of numsp384t1 indicated that the designed algorithm enhanced the SM computation by 57.6%, 31.3% and 13.2% compared to [27,8,30]. While nums512t1 reduced the cost of SM by 59.2%, 31.0% and 13.2% for [27,8,30], respectively. The results also showed that the enhanced SM algorithm over prime field worked better with bigger scalar bit size. The percentage of speed up in the enhanced algorithm may be higher if tested on the different running environments. As the enhanced SM algorithm does not contain inverse operation, this research will lead to other advancements such as [33] in developing the SM algorithm in ECC.

Acknowledgement

This research was not funded by any grant. This research receives facility support from Universiti Tunku Abdul Rahman and the Institute for Mathematical Research, Universiti Putra Malaysia.

References

[1] Miller, Victor S. "Use of elliptic curves in cryptography." In *Conference on the theory and application of cryptographic techniques*, pp. 417-426. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985. https://doi.org/10.1007/3-540-39799-X_31

[2] Koblitz, Neal. "Elliptic curve cryptosystems." *Mathematics of computation* 48, no. 177 (1987): 203-209. https://doi.org/10.1090/S0025-5718-1987-0866109-5

[3] Mahto, Dindayal, and Dilip Kumar Yadav. "RSA and ECC: A comparative analysis." *International journal of applied engineering research* 12, no. 19 (2017): 9053-9061.

[4] Gaur, Vimal, Bineet Singh, Megha Agrawal Deepak, and Nimisha Mishra. "Estimation of Various Scalar Multiplication Algorithms in ECC." *International Journal of Innovative Technology and Exploring Engineering (IJITEE), SSN*: 2278-3075.

[5] Shaikh, Javed R., Maria Nenova, Georgi Iliev, and Zlatka Valkova-Jarvis. "Analysis of standard elliptic curves for the implementation of elliptic curve cryptography in resource-constrained E-commerce applications." In *2017 IEEE international conference on microwaves, antennas, communications and electronic systems (COMCAS)*, pp. 1-4. IEEE, 2017. https://doi.org/10.1109/COMCAS.2017.8244805

[6] Kadu, Rakesh K., and Dattatraya S. Adane. "A novel efficient hardware implementation of elliptic curve cryptography scalar multiplication using vedic multiplier." *International Journal of Simulation: Systems, Science and Technology* 19, no. 6 (2018): 38-1. https://doi.org/10.5013/IJSSST.a.19.06.38

[7] Ajeena, Ruma Kareem K., and Sarah Jabbar Yaqoob. "The integer sub-decomposition method to improve the elliptic elgamal digital signature algorithm." In *2017 International Conference on Current Research in Computer Science and Information Technology (ICCIT)*, pp. 14-20. IEEE, 2017. https://doi.org/10.1109/CRCSIT.2017.7965554

[8] Kanayama, Naoki, Yang Liu, Eiji Okamoto, Kazutaka Saito, Tadanori Teruya, and Shigenori Uchiyama. "Implementation of an elliptic curve scalar multiplication method using division polynomials." *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 97, no. 1 (2014): 300-302. https://doi.org/10.1587/transfun.E97.A.300

[9] Javeed, Khalid, Xiaojun Wang, and Mike Scott. "High performance hardware support for elliptic curve cryptography over general prime field." *Microprocessors and Microsystems* 51 (2017): 331-342. https://doi.org/10.1016/j.micpro.2016.12.005

[10] Bernstein, Daniel J., Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. "Twisted edwards curves." In *Progress in Cryptology–AFRICACRYPT 2008: First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings 1*, pp. 389-405. Springer Berlin Heidelberg, 2008. https://doi.org/10.1007/978-3-540-68164-9_26

[11] Al Saffar, Najlae Falah Hameed, and Mohamad Rushdan Md Said. "Improved Arithmetic on Elliptic Curves over Prime Field." (2014).

[12] Li, Xingran, Wei Yu, and Bao Li. "Parallel and Regular Algorithm of Elliptic Curve Scalar Multiplication over Binary Fields." *Security and Communication Networks* 2020 (2020): 1-10. https://doi.org/10.1155/2020/4087873

[13] Liu, Shuang-Gen, Xin Heng, and Yuan-Meng Li. "Anti-SPA scalar multiplication algorithm on Twisted Edwards elliptic curve." *International Journal of Network Security* 22, no. 6 (2020): 1015-1021.

[14] Хлебородов, Д. С. "Эффективный алгоритм скалярного умножения точки эллиптической кривой на основе NAF-метода." *Программная инженерия* 7, no. 1 (2016): 21-28. https://doi.org/10.17587/prin.7.21-28

[15] Takemura, Yusuke, Keisuke Hakuta, and Naoyuki Shinohara. "ECC Atomic Block with NAF against Strong Side-Channel Attacks on Binary Curves." *International Journal of Networking and Computing* 10, no. 2 (2020): 277-292. https://doi.org/10.15803/ijnc.10.2_277

[16] Yasin, Sharifah Md. "New signed-digit {0, 1, 3}-NAF scalar multiplication algorithm for elliptic curve over binary field." PhD diss., Universiti Putra Malaysia, 2011.

[17] AbdulRaheem, Waleed K., Sharifah Bte Md Yasin, Nur Izura Binti Udzir, and Muhammad Rezal bin Kamel Ariffin. "Improving the Performance of {0, 1, 3}-NAF Recoding Algorithm for Elliptic Curve Scalar Multiplication." *International Journal of Advanced Computer Science and Applications* 10, no. 4 (2019). https://doi.org/10.14569/IJACSA.2019.0100432

[18] Chen, Binglong, Chuangqiang Hu, and Chang-An Zhao. "Note on scalar multiplication using division polynomials." *IET Information Security* 11, no. 4 (2017): 195-198. https://doi.org/10.1049/iet-ifs.2015.0119

[19] Stange, Katherine E. "The Tate pairing via elliptic nets." In *Pairing-Based Cryptography–Pairing 2007: First International Conference, Tokyo, Japan, July 2-4, 2007. Proceedings 1*, pp. 329-348. Springer Berlin Heidelberg, 2007. https://doi.org/10.1007/978-3-540-73489-5_19

[20] Muslim, N., F. Yunos, Z. Razali, and M. R. M. Said. "Elliptic Net Scalar Multiplication upon Koblitz Curves." *Malaysian Journal of Mathematical Sciences* 14, no. 3 (2020).

[21] Agievich, Sergei Valer'evich, Stanislav Vyacheslavovich Poruchnik, and Vladislav Igorevich Semenov. "Small scalar multiplication on Weierstrass curves using division polynomials." *Математические вопросы криптографии* 13, no. 2 (2022): 17-35. https://doi.org/10.4213/mvk406

[22] Lee, Chiou-Yng, Chia-Chen Fan, Jiafeng Xie, and Shyan-Ming Yuan. "Efficient implementation of karatsuba algorithm based three-operand multiplication over binary extension field." *IEEE Access* 6 (2018): 38234-38242. https://doi.org/10.1109/ACCESS.2018.2851662

[23] Bessalov, A. V., and L. V. Kovalchuk. "Supersingular twisted Edwards curves over prime fields. I. Supersingular twisted Edwards curves with j-invariants equal to zero and 12 3." *Cybernetics and Systems Analysis* 55 (2019): 347-353. https://doi.org/10.1007/s10559-019-00140-9

[24] Bernstein, Daniel J., and Tanja Lange. "Inverted edwards coordinates." In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes: 17th International Symposium, AAECC-17, Bangalore, India, December 16-20, 2007. Proceedings 17*, pp. 20-27. Springer Berlin Heidelberg, 2007. https://doi.org/10.1007/978-3-540-77224-8_4

[25] Hisil, Huseyin, Kenneth Koon-Ho Wong, Gary Carter, and Ed Dawson. "Twisted Edwards curves revisited." In *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 326-343. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. https://doi.org/10.1007/978-3-540-89255-7_20

[26] Ward, Morgan. "Memoir on elliptic divisibility sequences." *American Journal of Mathematics* 70, no. 1 (1948): 31-74. https://doi.org/10.2307/2371930

[27] Elmegaard-Fessel, Lars. "Efficient scalar multiplication and security against power analysis in cryptosystems based on the NIST elliptic curves over prime fields." *Cryptology ePrint Archive* (2006).

[28] Eide, Olav Wegner. "Elliptic Curve Cryptography-Implementation and Performance Testing of Curve Representations." Master's thesis, 2017.

[29] Cuevas-Farfan, Eduardo, Miguel Morales-Sandoval, Alicia Morales-Reyes, Claudia Feregrino-Uribe, Ignacio Algredo-Badillo, Paris Kitsos, and Rene Cumplido. "Karatsuba-Ofman Multiplier with Integrated Modular Reduction for (2) m GF." *Advances in Electrical and Computer Engineering* 13, no. 2 (2013): 1-5. https://doi.org/10.4316/AECE.2013.02001

[30] SubramanyaRao, SrinivasaRao, Zhi Hu, and Chang-An Zhao. "Division polynomial-based elliptic curve scalar multiplication revisited." *IET Information Security* 13, no. 6 (2019): 614-617. https://doi.org/10.1049/iet-ifs.2018.5361

[31] Kudithi, T. "An efficient hardware implementation of finite field inversion for elliptic curve cryptography." *International Journal of Innovative Technology and Exploring Engineering* 8, no. 9 (2019): 827-832. https://doi.org/10.35940/ijitee.I7605.078919

[32] Hankerson, Darrel, and Alfred Menezes. "Elliptic curve cryptography." In *Encyclopedia of Cryptography, Security and Privacy*, pp. 1-2. Berlin, Heidelberg: Springer Berlin Heidelberg, 2021. https://doi.org/10.1007/978-3-642-27739-9_245-2

[33] Jagadeesh, Selvaraj, Sabna Machinchery Ali, Soundara Pandian Gnana Selvan, Mohammad Aljanabi, and Manimaran Gopianand. "Hybrid AES-Modified ECC Algorithm for Improved Data Security over Cloud Storage." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 32, no. 1 (2023): 46-56. https://doi.org/10.37934/araset.32.1.4656