*IOSR Journal of Computer Engineering (IOSR-JCE)*
*e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 2, Ver. V (Mar-Apr. 2016), PP 31-47*
*www.iosrjournals.org*

# A Survey on Single Scalar Point Multiplication Algorithms forElliptic Curves overPrime Fields

Mohammad Rasmi[1], Ahmad Abu Sokhon[2], Mohammad Sh. Daoud[3], Hani Al-Mimi[4]
[1]*Zarqa University, Jordan*
[2]*Al-Zytoonah University of Jordan*
[3]*Al Ain University of Science and Technology, UAE*
[4]*Al-Zytoonah University of Jordan*

***Abstract:****Elliptic Curve Cryptography (ECC) is an attractive field of research since it requires a shorter key length compared to other public-key cryptosystems such as RSA. A shorter key reduces the required computations, power consumption, and storage. The major time-consuming operation in ECC is the point multiplication, $kP$. Therefore, a lot of research has been carried out to improve the efficiency of ECC implementations. Composite Elliptic Curve (EC) operations and recoding methods are two factors that affect the efficiency of EC scalar multiplication. Deciding which composite EC operation to be used in an ECC system helps to improve the computational efficiency. In addition, finding a method that accelerates the EC computations, which depends on a new recoding method and employing the most efficient composite operations, is considered a pressing need. In this a research, a survey of EC single scalar multiplication methods is introduced. Therefore, a comprehensive information related to EC multiplication methods will be provided in order to facilitate literature review for researchers who would like to conduct a research in this area of science.*

***Keywords:****Elliptic Curve Cryptography, recoding methods, EC multiplication*

## I. Introduction

Computer Security is an important field that covers services, mechanisms and processes that protect computer data from unauthorized access. In the field of computer security, cryptography plays a major role in securing data. Security objectives such as confidentiality, data integrity, data origin authentication, entity authentication and non-repudiation can be achieved by using cryptography (Stallings, 2013). The security of commonly used public-key algorithms relies on three number-theoretic problems that are considered to be hard problems: factorization, discrete logarithm and EC discrete logarithm problems.

Elliptic curve cryptography (ECC), which was proposed independently by Koblitz and Miller (Darrel et al., 2003), is based on the Elliptic CurveDiscrete Logarithm Problem (ECDLP) (Darrel et al., 2003).The ECDLP has exponential time complexity, while the factorization and discrete logarithm problems can be solved in sub-exponential time (Eisenträger et al., 2003). ECC is attractive since it has many criteria that should be considered when designing an encryption system such as performance and security (Darrel et al., 2003).

Table I.1: Key size for ECC and RSA (Stallings, 2013)

| **ECC key size** | **RSA key size** |
|---|---|
| 112 | 512 |
| 160 | 1024 |
| 224 | 2048 |
| 256 | 3072 |

ECC offers equal security as well as RSA with a smaller key size and lower processing power as it can be seen from Table (Stallings, 2013). Portable devices, such as personal digital assistants (PDAs), mobile phones and smart cards, are equipped with limited memory. Therefore, ECC is more suitable than conventional public key cryptosystems for these devices (Tsaur and Chou, 2005). Elliptic Curve operations are performed over a finite field. Thus, its efficiency affects the implementation or design of ECC algorithms. Three types of fields are commonly used in ECC: prime, binary and extension fields (Darrel et al., 2003).Prime fields are considered better than binary fields for software implementation, while binary fields are the best for hardware implementation (Stallings, 2013).

The infrastructure of some public-key cryptosystems is the Abelian group. In Diffie-Hellman key exchange, which depends on the Discrete Logarithm Problem, the exponentiation is defined as repeated multiplication. The exponentiation is computed using the Square-and-Multiply method (Knuth, 1997). While, Double-and-Add method, which is the counterpart of the Square-and-Multiply method, depends on two basic operations: addition and doubling (Darrel et al., 2003, Knuth, 1997).Equation (1) is used to define an Elliptic Curve$E$ over a prime field $p$

$$E(F_q): y^2 mod p = (x^3 + ax + b) mod p \qquad (1)$$
where $a, b, x, y \in F_p$and$\Delta = -16(4a^3 + 27b^2)$

The EC operations, addition$(A)$ and doubling$(D)$, are considered the basic operations.The cost of EC addition in terms of field operations is $[i] + 2[m] + [s]$, while the cost of EC doubling is $[i] + 2[m] + 2[s]$ (where $i$ stands for inversions, $m$ for multiplication, and $s$ for squaring).For the EC $E(F_p)$ , with characteristic$(F_p) > 3$, let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ where $P_1 \neq -P_2, P_1, P_2, P_3 \in E(F_p)$. Let $\infty$ represents the point at infinity, then:

1. $P_1 + \infty = P_1$
2. $P_1 + -P_1 = \infty$, where $-P_1 = (x_1, -y_1)$
3. $P_3 = P_1 + P_2 = (x_3, y_3)$
4. $x_3 = (\lambda^2 - x_1 - x_2) \bmod q$
5. $y_3 = (\lambda(x_1 - x_3) - y_1) \bmod q$
6. $\lambda = \begin{cases} \left(\frac{y_2 - y_1}{x_2 - x_1}\right) mod q if P_1 \neq P_2 \\ \left(\frac{3x_1^2 + a}{2y_1}\right) mod q if P_1 = P_2 \end{cases}$

## II. Point Representation

There are several ways to represent the EC point. The Affine coordinate system is the traditional coordinate system that is used in the Cartesian plane. The point $P$ is represented in Affine coordinates as $P(x_1, y_1)$. On the other hand, the Projective coordinate system is used when the cost of inversion is high compared to the cost of multiplication, especially over prime fields (Darrel et al., 2003, Dimitrov et al., 2008). Therefore, the need for a multiplicative inverse in the EC operations is eliminated by using Projective coordinates. There are some variants of Projective coordinates such as standard Projective coordinates, Jacobian coordinates and Chudnovsky coordinates(Darrel et al., 2003).

## III. Point Multiplication

The dominant operation in EC cryptography is point multiplication. Figure 1.showsthat there aretwo methods of point multiplication: single-scalar and multi-scalar multiplication. Computing$kP$, where $k \in Z$and $P \in E(F_p)$, is called single-scalar multiplication while computing $kP + lQ$ is called mutli-scalar multiplication. Both methods are used in the Elliptic Curve Digital Signature Algorithm (ECDSA).

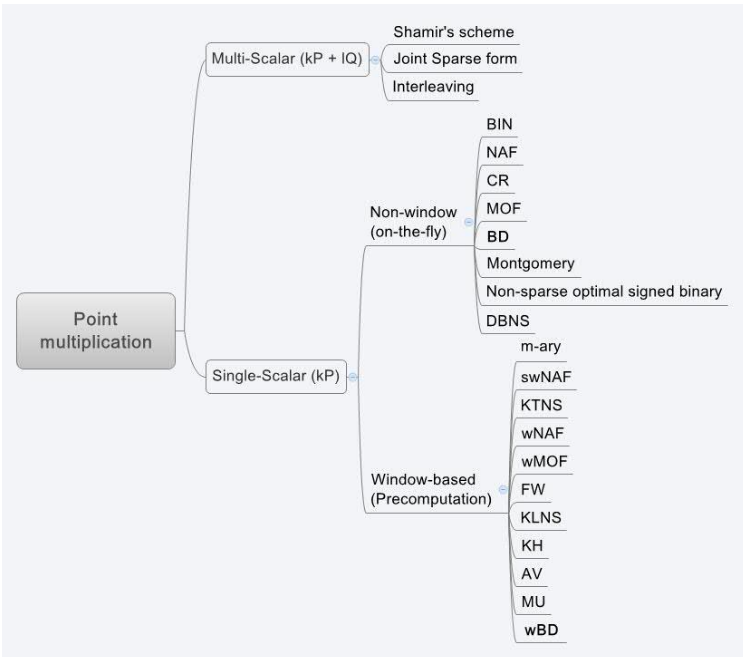

Figure 1. Types of elliptic curve point multiplication

There are three methods for computing EC point multiplication based on the knowledge of point $P$ and exponent $k$. The first uses a generic algorithm when point$P$and exponent $k$are variables. It can also be called the on-the-fly technique. The second method finds the shortest addition subtraction chain when the exponent is fixed. The third method uses precomputations whenever the point is fixed(Cohen and Frey, 2006).

Table summarizes the three methods.

Table 1.Types for ECpoint multiplication based on knowledge of the key and the exponent

| **P** | **k** | **Method** |
|---|---|---|
| variable | variable | Generic algorithm (on-the-fly) |
| variable | fixed | Finding the shortest addition subtraction chain |
| fixed | variable | Precomputation |

A pre-computation may give a noticeable improvement to the multiplication methods in terms of speed if there is sufficient memory to store some precomputations. The recoding method that is used to represent the exponent $k$ significantly reduces the computational costof the multiplication method used. Moreover, the common approach is trying to invent a new recoding method that has the minimal Hamming Weight or the shortest addition chains and then invent a single or multi-scalar multiplication method that takes advantage from the proposed recoding method (Darrel et al., 2003, Cohen and Frey, 2006).The single scalar EC point multiplication is represented by $kP$, where $k \in Z$ and $P \in E(F_p)$.This quantity can be computed on-the-fly or using precomputations if sufficient memory is available. The exponent $k$ can be represented using radix 2 positional number system as a summation of the form

$$k = \sum_{0 \leq i < n} a_i 2^i = (a_{n-1} \dots a_3 a_2 a_1 a_0)_2 = a_{n-1}2^{n-1} + \cdots + a_2 2^2 + a_1 2^1 + a_0 2^0, \qquad (2)$$

$where a \in D_2 = \{0,1\} and n = \log_2 k$

If the digit set is extended to$D_w = \{0, \pm 1\}$,it will be the signed binary representation of exponent $k$. Moreover asigned window representation of exponent $k$ can be achieved if the digit set is extended to $D_w = \{0, \pm 1, \pm 3, \pm 5, \dots\}$.

## 2.1 On-the-Fly Elliptic Curve Multiplication

Many algorithms have been proposed to compute $kP$, where $P \in E(F_p)$(Darrel et al., 2003). These algorithms mainly depend on the recoding method of exponent $k$. The most popular method for performing EC point multiplication of the form $kP$ is the Double-and-Add(Binary) method which uses the digit set $D_w = \{0,1\}$ to represent exponent $k$. The efficiency of this method can be further improved using signed binary representations. Figure 2. presents the classification of the on-the-fly EC multiplication methods and some examples. The signed methods use the digit set $D_w = \{0, \pm 1\}$.Also further improvements can be achieved if some precomputations are allowed such as in window recoding and multiplication techniques where the digit set $D_w$ can include more values as $D_w = \{0, \pm 1, \pm 3, \dots\}$.

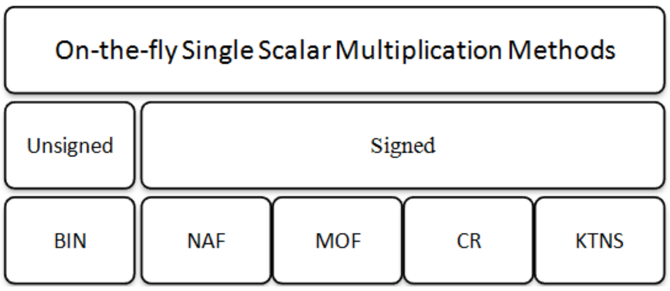

Figure 2. On-the-fly EC schemes

Window methods are considered more efficient than non-window methods if sufficient memory is availableto store the precomputed points. In ECC, left-to-right recoding is the natural choice, because window methods can be used more efficiently. In addition, the multiplication method and recoding method can be merged; therefore, there is no need to store the recoded scalar $k$. The left-to-right methodsare more suitable for limited hardware devices(Okeya et al., 2004).

### 2.1.1 Unsigned Standard Binary Method

The Classical technique that is used to compute $kP$ is the standard binary method (or so-called Double-and-Add method). This method has a linear complexity in terms of the size of the input (Doche and Imbert, 2006) and can be deduced in both directions: left-to-right and right-to-left. Algorithm 1isthe left-to-right variant of binary method(Knuth, 1997, Darrel et al., 2003). The cost of the standard binary multiplication method as stated by Darrel et al. (2003), Dimitrov et al. (2008), and Morain and Olivos (1990) for $E(F_p)$with$char > 3$, is $\frac{n}{2}[A] + n[D]$, where, $n = \log_2 k$and $W(k) = \frac{n}{2}$. The cost of $kP$ in terms of field operations using Affine coordinates is $n(1.5[i] + 3[m] + 2.5[s])$.

```
Algorithm 1: L-to-R standard binary point multiplication
Input            : $k_2, P \in E(F_q)$
Output           : $kP$

$Q \leftarrow \infty$
For $i(\lfloor \log_2 k \rfloor$to 1)
        $\{Q \leftarrow 2Q$
        $\{$If $(k_i = 1)$ then $Q \leftarrow Q + P$
Return $(Q)$
```

### 2.1.2 Signed Binary Schemes

The main idea behind using signed binary representation is to speed up the Classical multiplication method on computers(Booth, 1951). In fact, there are several signed representations of integer $k$ in Radix-2. These representations are suitable for EC scalar multiplication, since the Hamming Weight of some signed binary representations is less than that for unsigned binary. Exponent $k$is represented in signed binary as $k = \sum_{0 \leq i < n} a_i 2^i$, where$a \in D_2 = \{0, \pm 1\}$and$n = \log_2 k$.

The drawback of signed representations is its redundancy due to the use more than 3 digits to represent a binary number. For example, 23 has a unique binary representation while it has many signed representations such as: $1100\bar{1}, 10\bar{1}00\bar{1}, 1\bar{1}1\bar{1}11$. Deciding which representation is the shortest is an open problem (Okeya et al., 2004, Jedwab and Mitchell, 1989). On the other hand, an advantage of signed representations is in finding the additive inverse of an EC point. The inverse of a point $P \neq \infty$ in ECs can be found simply by negation, i.e. the inverse of point $P$ can be calculated for free (Muir and Stinson, 2006). Various techniques of signed binary multiplication methods that have been proposed will be explained in the following sections (Solinas, 1997, Okeya et al., 2004,Koyama and Tsuruoka, 1993).

#### Non Adjacent Form (NAF)

The NAF method was introduced by Reitwiesner in 1960 (Reitwiesner, 1960). Then a generalization of the NAF was introduced bySolinas (2000) which can be used to enhance the efficiency of EC computations. Left-to-right NAF recoding was firstly introduced by Joye and Sung-Ming (2000)(Kong and Li, 2005). Every integer $k$ has a unique NAF representation$k = \sum_{0 \leq i < n} a_i 2^i$, where$a \in D_2 = \{0, \pm 1\}$and$n = \log_2 k$(Darrel et al., 2003). The Hamming Weightof$(k_{NAF}) = \frac{2^n(3n-4)-(-1)^n(6n-4)}{9(n-1)(2^n-(-1)^n)} \approx \frac{1}{3}$, where $n = log_2 k$(Morain and Olivos, 1990, Solinas, 2000).

Even though obtaining the shortest addition sequence is a NP-complete problem, NAF representation has the fewest number of nonzero digits among other signed representations and it can be found in a linear complexity (Gordon, 1998, Solinas, 1997). The bit-length $n$ of $k_{NAF}$ may be one bit longer than the bit-length of its binary expansion; $log_2 k_{NAF} \leq log_2 k + 1$. The signed binary expansion NAF must have at least one zero after each nonzero digit. This is called sparseness or the non-adjacency property (Morain and Olivos, 1990, Solinas, 2000). Algorithm 2is one of the easiest and most elegant methods that are used to compute $k_{NAF}$ from the integer $k$ (Solinas, 2000).

```
Algorithm 2: NAFrecoding method
Input            : $k = \sum_{i=0}^{n-1} k_i 2^i, k_i \in \{0,1\}$
Output           : $k_{NAF} = \sum_{j=0}^{i-1} k_j 2^j, k_j \in \{0,1,-1\}$
$i \leftarrow 0$
While$k \geq 1$
        If odd$(k)$
            $k_i \leftarrow 2 - (k \bmod 4)$
            $k \leftarrow k - k_i$
        Else $k_i \leftarrow 0$
        $k \leftarrow \frac{k}{2}$
        $i \leftarrow i + 1$
Return $(k_{NAF})$
```

Algorithm 3is the left-to-right variant of the NAF multiplication method.It requires $\frac{n}{3}A + nD$ . Consequently, the computational cost of the addition-subtraction method is 12% less than that for the standard binary method (Solinas, 2000).

```
Algorithm 3: L-to-R NAF point multiplication
Input       : $k_{NAF} = \sum_{i=0}^{n-1} k_i 2^i, k_i \in \{0,1,-1\} P \in E(F_p)$
Output      : $k_{NAF} P$

$Q \leftarrow P$
For $i(n-2$ downto $0)$
    $Q \leftarrow 2Qi$
    If $(k_i = 1)\ Q \leftarrow Q + P$
    If $(k_i = -1)\ Q \leftarrow Q - P$
Return ($Q$)
```

**Mutual Opposite Form (MOF)**

The Mutual Opposite Form (MOF) was proposed by Okeya et al. (2004). MOF is a bidirectional method.Any positive integer $k$ with a bit-length $n$ has a unique MOF representation, where$\log_2 k_{MOF} \leq \log_2 k + 1$. The signs of any nonzero consecutive digits are opposite. The Hamming Weight, $W(k_{MOF}) = \frac{n}{2}$, is almost the same as binary representation (Okeya et al., 2004). In spite of the fact that the Hamming Weight of MOF is greater than that for NAF, the MOF recoding method is faster than both NAF and binary methods when the cost of recoding is measured together with the performance of the EC multiplication method (Balasubramaniam and Karthikeyan, 2007). The integer $k$ is converted to MOF mathematically by the equation $k_{MOF} = 2k \ominus k$, where $\ominus$ represents a bitwise subtraction. Algorithm 4provides an explicit conversion from $k_2$ to $k_{MOF}$. The previous algorithm, Algorithm 3,can be used to compute the point multiplication.

```
Algorithm 4: Left-to-right MOF recoding
Input       : $k = \sum_{i=0}^{n-1} k_i 2^i, k_i \in \{0,1\}$
Output      : $k_{MOF} = \sum_{i=0}^{n} v_i 2^i, v_i \in \{0,1,-1\}$

$v_n \leftarrow k_{n-1}$
For $i(n-1$ to $1)$
    $\{v_i = k_{i-1} - k_i$
$v_0 = -k_0$

Return ($k_{MOF}$)
```

**Complementary Recoding Method (CR)**

The complementary recoding (CR) was proposed by Chang et al. (2003) to solve the problem of common-multiplicand multiplications. It is supposed that the advantages of complementary methods help increase the speed of large integer multiplications. The integer $k$, with a bit-length $n$, is represented in signed binary using the CR method as follows$k_{CR} = \sum_{i=0}^{n} k_i 2^i$, where$k_i \in \{0, \pm 1\}$. Algorithm 5is used to compute $k_{CR}$.

```
Algorithm 5: Complementary recoding
Input       : $k = \sum_{i=0}^{n-1} k_i 2^i, k_i \in \{0,1\}$
Output      : $k_{CR} = \sum_{i=0}^{n} w_i 2^i, w_i \in \{0,1,-1\}$
Remark  : $\bar{k}$ : one's complement of $k$

$v \leftarrow 2^n$
$kc \leftarrow \bar{k}$
$k_{CR} \leftarrow v - \bar{k} - 1$

Return ($k_{CR}$)
```

The recoding methods, CR and MOF, are faster than NAF since they do not involve expensive operations such as division. They only use simple operations such as bitwise operations, additions and subtractions. Neither Chang et al. (2003) nor the authors Balasubramaniam and Karthikeyan (2007) measured the Hamming Weight of the recoded numbers by the CR method. Therefore, the average $W(k_{CR})$is measured in(Almimi et al., 2015).

Algorithm 6 is the CREC multiplication method (Balasubramaniam and Karthikeyan, 2007). Their method was compared with NAF, MOF and standard binary methods. They measured the performance of the previous EC methods including the recoding process. They showed that their method is the fastest. The recoding method, MOF, does not require expensive operations such as multiplication, division and modulus as claimed by Balasubramaniam and Karthikeyan (2007).

```
Algorithm 6:EC Point multiplication using complementary recoding
Input        : $k_{CR} = \sum_{i=0}^{n} w_i 2^i, w_i \in \{0,1,-1\}, P \in E(F_p)$
Output       : $Q = k_{CR}P$

$Q \leftarrow \infty$
For $i(n-1$ to $0)$
        $Q \leftarrow 2Q$
        If $(w_i = 1) Q \leftarrow Q + P$
        Else if $(w_i = -1) Q \leftarrow Q - P$
Return $(Q)$
```

## 2.2 Window-based Elliptic Curve Multiplication

Elliptic curve operations can be improved by many techniques. Finding a new recoding method which transforms exponent $k$ to $k'$ with less Hamming Weight is one of the techniques that may crucially increase the efficiency of an EC scheme. These recoding methods are classified into window and non-window methods. Window methods are considered a generalization of non-window methods (Solinas, 2000, Koyama and Tsuruoka, 1993, Blake et al., 1999, Okeya et al., 2004, Möller, 2002 Seoul, Korea,, Schmidt-Samoa et al., 2006).Window methods are used when extra memory is available to store some precomputed values (Darrel et al., 2003).It is also of interest to have a left-to-right recoding method since it enhances the efficiency of computing *kP* due to the fact that there is no need to store the recoded exponent *k*.

Figure shows two recoding methods of the integer *k*: unsigned binary representation $k_2 = \sum_0^{n-1} k_i 2^i, k_i \in \{0, 1\}$ and unsigned window representation $k_w = \sum_0^{m-1} k_i 2^{w*i}, k_i \in D_w = \{0,1,\dots,2^{w-1}\}$.The Hamming Weightof $k_w$ is $\frac{n}{w}$, where$n = \log_2 k$. The number of EC doublings relies on the length of the exponent, while the number of EC additions relies on $W(k)$. Hence, processing *w* digits at a time will reduce the number of EC additions with extra memory needed to store the set $D_w$. If only odd values of set $D_w$ are used and zero runs are skipped, then the number of additions will be reduced. Moreover, if signed values are also allowed, the number of precomputations can be reduced. Exponent $k$ can be scanned left-to-right or right-to-left. The former method is preferable for window EC multiplication methods since it can be combined with EC multiplication methods without storing exponent *k*, i.e. left-to-right methods enable us to do recoding and multiplication simultaneously.

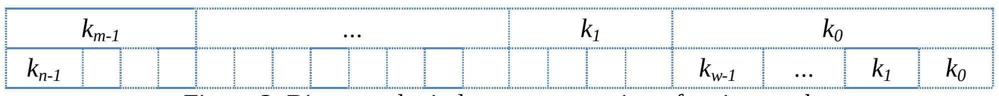

Figure 3. Binary and window representation of an integer *k*

Generally, there are two ways of applying window methods: the first one is a fixed window method like the *m*-ary method, which processes *w* digits at a time without skipping any digit. The second applies a more dynamic technique over the recoded exponent which is a sliding window method (Okeya et al., 2004). Zero runs are skipped while applying the second window technique; therefore only odd window values will be pre-calculated and stored (Gordon, 1998).Moreover, using signed binary representation will reduce the number of precomputed elements.

Morain and Olivos (1990) firstly suggested applying the NAF to construct the addition-subtraction chain for point multiplication (Kong and Li, 2005). The window method, over non-sparse optimal signed binary representation, was firstly proposed by Koyama and Tsuruoka (1993). Miyaji et al. (1997) proposed the wNAF window method for fixed and random EC points. The sliding window method over NAF and wNAFwas also introduced by Solinas (1997). The fractional window method was presented by Möller (2002 Seoul, Korea,) to use the available memory in a more efficient way than the previous methods. Later on, other left-to-right window methods were proposed by Okeya et al. (2004), Avanzi (2005), Muir and Stinson (2005), and Khabbazian et al. (2005). The properties of the proposed methods were either proved in the previous papers or by other papers. For example, the minimality property of the fractional window method was proved by Möller (2004). Some properties of non-sparse optimal signed binary representation and its window method were analyzed by Kong and Li (2005). Muir and Stinson (2006) showed that wNAF has a minimal number of nonzero digits.

**The *m*-ary method**

This method represents the basic technique that uses the fixed window slicing technique where the zero runs are not skipped. The integer $k$ is represented as $k_{m-ary} = \sum_{i=0}^{n-1} k_i m^i$, $m = 2^w, k_i \leq 2^w - 1$, $n = \left\lceil \frac{\log_2 k_2}{w} \right\rceil$. Thus the exponent is divided into $\frac{\log_2 k}{w}$ windows, where $w \geq 2$ and each window valueis in the set $\{0,1,..,2^w - 1\}$.Suppose that $k = (1001100001010110)_2$ and $w = 4$ then $k$is divided into blocks of 4 bits as follows.

| $k$ = | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $m_3$ | | | | $m_2$ | | | | $m_1$ | | | | $m_0$ | | | |

```
Algorithm 7: m-arypoint multiplication
Input          : w,$P \in E(F_p)$, $k = \sum_{j=0}^{n-1} k_j m^j$ , $k_j \in \{0,1,\ldots,m-1\}, m = 2^w$
Output         : $Q = [k]P$

Precomputation
$P_1 = P$
For$i(2$ $ton - 1)$
{$P_i \leftarrow P_{i-1} + P$ (we have $P_i = [i]P$)

$Q \leftarrow \infty$
For$j(n - 1$ to 0)
        {$Q \leftarrow [m]Q$
        {$Q \leftarrow Q + P_{k_j}$
Return ($Q$)
```

Algorithm 7 is the window $m$-ary EC point multiplication method. The cost of precomputations is $n \cdot w[D] + \mathrm{n}[A]$(Blake et al., 1999).

**Sliding Window Non Adjacent Form**

The sliding window technique can be implemented over signed and unsigned binary where the value of each window is odd. In this section, the sliding window techniqueis applied to NAF and called sliding window NAF(swNAF). It can be implemented in both directions (right-to-left or left-to-right). Algorithm 8 is the sliding window over NAF recoded keys. The zero runs are skipped so that each window value is odd(Darrel et al., 2003).The number of precomputed points is $\left(\frac{2^w-(-1)^w}{3}\right)$. It follows that the expected running time for the sliding window methods over $k_{NAF}$including pre-computation cost is $[D] + \left(\frac{2^w-(-1)^w}{3} - 1\right)[A] + \frac{n}{w+v(w)}[A] + n[D]$ where $v(w) = \frac{4}{3} - \frac{(-1)^w}{3.2^{w-2}}$, which represents the average length of zero runs between windows, and $n = \log_2 k$(Darrel et al., 2003).Figure4.shows the process of a moving window over a NAF recoded number right-to-left.

```
Algorithm 8: NAF sliding window multiplication
Input          : $w, k, P \in E(Fq)$
Output         : $kP$

$k_2 \rightarrow k_{NAF}$ using Algorithm 2
Pre-compute the points $1P, 3P, \ldots, \left(\frac{2^w-(-1)^w}{3} - 1\right)P$

$Q \leftarrow \infty, i \leftarrow l-1$
While i ≥ 0
        If ($k_i = 0$) : $t \leftarrow 1, u \leftarrow 0$
        Else: find the largest $t \leq w$ such that $u \leftarrow (k_i, \ldots, k_{i-t+1})$is odd
        $Q \leftarrow 2^t Q$
        If ($u > 0$)$Q \leftarrow Q + P_u$
        Else if ($u < 0$) $Q \leftarrow Q - P_{-u}$
        $i \leftarrow i - t$
Return (Q)
```

| ← direction of processing | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | $\bar{1}$ | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 1 | 0 | 1 | 0 | $\bar{1}$ | 0 |
| 1 | 0 | $\bar{1}$ | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 1 | 0 | 1 | 0 | $\bar{1}$ | 0 |
| 1 | 0 | $\bar{1}$ | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 1 | 0 | 1 | 0 | $\bar{1}$ | 0 |
| 1 | 0 | $\bar{1}$ | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 1 | 0 | 1 | 0 | $\bar{1}$ | 0 |

Figure4. Sliding window technique over NAF

**Non-Sparse Signed Window Method**

This method is a window method proposed by Koyama and Tsuruoka (1993)over Non-Sparse Optimal Signed Binary representation (KTNS). The word optimal means that the Hamming Weight of the signed binary representation is the minimal one among other signed representations(Kong and Li, 2005). Moreover, the sparse output means that for any nonzero digit, its adjacent digit cannot be nonzero. It requires $(2^{w-1}-1)$ precomputed windows. They described their method for fixed points and unknown points. Their average length of zero runs in their representation was 1.42 compared to 1.29 for Morain and Olivos (1990). It was also mentioned byKong and Li (2005) that the average length of zero runs for KTNS recoding was 1.5.

They mentioned that the optimal windows size depends on the bit-length of the EC key. For example, it is 5 for 511 bit-length. They proposed the transformation method, other than the "Morain and Olivos" and "Jedwab and Mitchell" methods (Morain and Olivos, 1990, Jedwab and Mitchell, 1989), which increases the average length of zero runs in the transformed key.The cost of EC multiplication of this method excluding the precomputations is $(n-Z')[D]\ +(\frac{n}{w+Z''})[A]$where $n$ is the length of $k_{KTNS}$ and $Z''$is the average length of zero runs. The number of zeros on the left most side of the most significant window is denoted by $Z'$.

The properties of non-sparse optimal signed binary representationswere investigated by Kong and Li (2005). They also proposed a new Non-Sparse Signed Window Splitting Algorithm (KLNS). The experimental results of the NAF recoding method were better than the results for KLNS. At the end of their paper they concluded that the combination of wNAF and the fractional window method is more efficient than other methods such as binary, their proposed algorithm and KTNSmultiplication method. They had modified the number of precomputed windows for non-sparse signed window representation (KTNS method) from $2^{w-1}-1$ to $\frac{5}{6}\cdot 2^{w-1}-1+\frac{(-1)^w}{3}$. The average zero-run length of their representation was 1.5 for $3 \le w \le 8$. Their algorithm is a 2-pass algorithm, since the input is a NAF representation.The expected number of nonzero windows (Hamming Weight) is$\frac{n}{z}$, where

$$z=\left(w+\frac{4}{3}+\frac{(-1)^w}{3.2^{w-1}}-\left(\frac{1}{2}\right)^{w-3}+(2+(-1)^w)\cdot\left(\frac{1}{2}\right)^{\frac{w}{2}-\frac{3}{4}\cdot(1-(-1)^w}\right).$$

**Window Non Adjacent Form**

In window NAF (wNAF), exponent *k*is directly recoded to $k_{wNAF}$ rather than converting it to NAF and then applying the sliding window technique as in swNAF. The Hamming Weight of wNAFis$\frac{n}{w+1}$.It was proved by Muir and Stinson (2006)that wNAF has the least Hamming Weight of all the recoding methods. In addition, they proved that the number of digits of a window signed representation is at most greater than the length of binary representation by one digit.

The wNAF of positive integer *k*is expressed as $k=\sum_{i=0}^{n-1} k_i 2^i$ for w ≥ 2 where each nonzero digit is followed by at least $w-1$ consecutive zeros. $k_{n-1} \neq 0$, and $k_i$ is either zero or odd, $|k_i| < 2^{w-1}$.Some basic facts about wNAF have been proved by Muir and Stinson (2006) such as: every integer can be represented by at most one wNAF,every integer has a wNAF, and $\lceil \log_2 k \rceil + 1 \le \lceil \log_2 k_{wNAF} \rceil$ .

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_2=$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $k_{w2NAF}$ = | 1 | 0 | 0 | 0 | 1 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 1 | 0 | $\bar{1}$ | 0 | $\bar{1}$ | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ |
| $k_{w3NAF}$ = | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | $\bar{1}$ | 0 | 0 | 1 | 0 | 0 | 3 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ |
| $k_{w4NAF}$ = | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | 0 | 7 |

Figure shows some representations of $k=(1122334455)_{10}$ in wNAF.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_2=$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $k_{w2NAF}$ = | 1 | 0 | 0 | 0 | 1 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 1 | 0 | $\bar{1}$ | 0 | $\bar{1}$ | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ |
| $k_{w3NAF}$ = | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | $\bar{1}$ | 0 | 0 | 1 | 0 | 0 | 3 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ | 0 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | $\bar{1}$ |
| $k_{w4NAF}$ = | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | $\bar{1}$ | 0 | 0 | 0 | 7 |

Figure 5.wNAFexample

Algorithm 9is used to recode $k$ to its wNAF representation. The expression $k mods 2^w$ denotes the integer $u$ satisfying $u \equiv k(mod 2^w)$ and $-2^{w-1} \leq u < 2^{w-1}$

```
Algorithm 9: wNAFrecoding
Input                  : $k \in Z, w \geq 1$
Output                 : $k_{wNAF} = (k_{l-1}, k_{i-2}, \ldots, k_1, k_0)$

$i \leftarrow 0$
While$k \geq 1$
        If ($k$is odd)$k_i \leftarrow k mods 2^w : k \leftarrow k - k_i$
        Else$k_i \leftarrow 0$
        $k \leftarrow k/2 : \; i \leftarrow i+1$
Return ($k_{wNAF}$)
```

Algorithm 10represents the wNAF EC multiplication method. The average length of zero runs for wNAF is 2 while it is 1 for binary representation and $\frac{4}{3} + \frac{(-1)^{w+1}}{3.2^{w-2}}$ for NAF representation(Kong and Li, 2005). The expected running time ofAlgorithm 10including the precomputations is $[D] + (2^{w-2} - 1)[A] + \left(\frac{n}{w+1}[A] + n[D]\right)$(Darrel et al., 2003). The number of precomputed points (windows) that should be stored is $2^{w-2}$, where the set of precmoputed points is $B = \{\pm 1P, \pm 3P, \ldots, (\pm 2^{w-1} - 1)P\}$. The recoding method is implemented right-to-left while the EC multiplication method is implemented in the reverse direction(Solinas, 2000). Thus the recoded exponent should be stored which consumes memory in limited devices.

```
Algorithm 10:wNAFpoint multiplication
Input                  : $w, k_{wNAF}, P \in E(F_p)$
Ooutput    : $kP$

Precompute the points $1P, 3P, \ldots, (2^{w-1} - 1)P$

$Q \leftarrow \infty$
For$i(l-1$ to 0)
            $Q \leftarrow 2Q$
            $Q \leftarrow$ { $Q \leftarrow Q + P_{k_i}$,       $k_i > 0$
                            $Q \leftarrow Q - P_{k_i}$,       $k_i < 0$
Return ($Q$)
```

In Algorithm 10, the window is moving left-to-right, skipping consecutive zero entries after a nonzero digit $k_i$is processed (Darrel et al., 2003). It is also proved thatwNAF method is asymptotically better than swNAF for *w*>3(Blake et al., 1999).The optimal window $w$ for $k$-bit integers varies from 3 to 6 for $k$-160 to $k$-600 (Kong and Li, 2005).

**Fractional Window Method**

The fractional window method (FW), which is provided in Algorithm 11,is a right-to-left recoding method that was introduced and examined by (Möller, 2002 Seoul, Korea,, Möller, 2004). They tried to invest the available memory as much as possible by storing the maximum number of precomputed points in the available memory. For example, the wNAF method requires four points to be stored for *w*=4. If the available memory is sufficient to store up to six elements, then there will be memory waste.
The digit set of the precomputed points for the FW method is $B = \{\pm 1, \pm 3, \ldots, \pm(2^{w-1} + m)\}$, where $m$ is an odd integer such that $1 \leq m \leq 2^{w-1} - 3$ for $w \geq 2$. The average density of nonzero digits of the FW method is $\frac{1}{w+1+\frac{m+1}{2^{w-1}}}$. The number of precomputed windows is $2^{w-2} + \frac{m+1}{2}$for $w \geq 3$.For Algorithm 11,the size of the window is reduced since the original algorithm works for a window size of $w$ + 1. Therefore, to unify all algorithms to use the same value of $w$ in this study, the step $w \leftarrow w - 1$is added to the algorithm.

```
Algorithm 11: Fractional window recoding
Input                  : $w, m: 1 \leq m \leq 2^w - 3, k$
```

```
Output          : $k_{wFrac} = b_{i-1}, b_{i-2}, \ldots, b_0$

$w \leftarrow w - 1$
$d \leftarrow LSB_{w+2}(k)$
$c \leftarrow \left\lfloor \frac{k}{2^{w+2}} \right\rfloor$
$i \leftarrow 0$
While $(d \neq 0 \vee c \neq 0)$
    $b \leftarrow \begin{cases} 0, & \text{even}(d) \\ d, & 0 < d \leq 2^w + m \\ d - 2^{w+1}, & 2^w + m < d < 3 \cdot 2^w - m \\ d - 2^{w+2}, & 3 \cdot 2^w - m \leq d < 2^{w+2} \end{cases}$
    $b_i \leftarrow b$
    $i \leftarrow i + 1$
    $d \leftarrow d - b$
    $d \leftarrow LSB(c) \cdot 2^{w+1} + \frac{d}{2}$
    $c \leftarrow \frac{c}{2}$

Return $(k_{wFrac})$
```

In order to apply the fractional window method, we first convert exponent $k$ to its fractional window format using Algorithm 11, then apply Algorithm 12 to compute the EC multiplication $kP$. Their method is faster than window NAF by 2.3% with $w$=3 and $m$=1 compared to wNAFof $w = 3$, assuming that $\beta = 1$, i.e. squarings and multiplications have the same execution time. If $m$ is increased, then the number of precomputed elements will be increased to the maximum. On the other hand, the average density will be decreased by $\frac{2^{w-1}-2}{2^{w-1}}$.This quantity is ≈ 1 whenever $w$ increases, so the average density will be $\approx \frac{1}{w+2}$ which is less than $\frac{1}{w+1}$ for wNAF. They extended their work by examining left-to-right and right-to-left fractional window method(Möller, 2004). They proved that both representations have the minimal number of nonzero digits among other signed representations.

```
Algorithm 12:Fractional window point multiplication
Input           : $w, k_{wFrac}, m, P \in E(F_p)$
Output          : $kP$

Precompute the points $1P, 3P, \ldots, (2^{w-1} + m)P$
$Q \leftarrow \infty$
For$i(\, l - 1 \; to \; 0\,)$
    $Q \leftarrow 2Q$
    $Q \leftarrow \begin{cases} Q + P_{k_i}, & k_i > 0 \\ Q - P_{k_i}, & k_i < 0 \end{cases}$

Return $(Q)$
```

**Mutual Opposite Form Window Method**

A bidirectional recoding method, called MOF (mutual opposite form) was proposed by Okeya et al. (2004). Their recoding method is more complex than the wNAF method. Window MOF requires two conversions; the first conversion is from unsigned binary representation of exponent $k$ to its MOF equivalent, then MOF is converted to its window MOF (wMOF) equivalent. Each positive integer has a unique wMOF representation that has the same length or is longer by one bit.The Hamming Weight of wMOFis$\frac{n}{w+1}$. However, the minimality property was not proved(Muir and Stinson, 2005). Any nonzero consecutive digits are opposite as it can be seen from the example: $0100\bar{1}01000\bar{1}001\bar{1}0$. Exponent $k$ is converted to MOF by using the bitwise subtraction,$k_{MOF} = 2k \ominus k$.In the recoding phase, only next $w + 1$ bits are required in each step.

**Avanzi Method**

A left-to-right recoding method with the same weight as wNAF for $w > 2$ had been proposed by Avanzi (2005) and Muir and Stinson (2005). The recoding methods have the minimality property as well as wNAF. The minimality property means that the representation has the minimal number of nonzero digits. They proved that their representation has the minimal weight and introduced a left-to-right representation with the same digit set as the methods of Cohen et al. (1998) and Solinas (1997). Their algorithm scans input left-to-right then outputs a recoded key with the same Hamming Weight as the wNAF.

**Khabbazian Method**

An integer representation that has the same average weight as wNAF was introduced by Khabbazian et al. (2005). Their representation can be generated from left-to-right and is called KH. Hence, there is no need for the algorithm to store the recoded exponent whichin turns saves memory. They also introduced a threshold integer$m$, which is the number of points that will be stored in the pre-computation stage. They extended the digit set $D_w = \{\mp 1, \mp 3, \ldots, \mp(2^{w-1} - 1)\}$ to include more windows $D_w = \{\mp 1, \mp 3, \ldots, \mp(2m - 1)\}$. For example, if we have seven memory locations that can hold seven values, the other window methods will use four locations since the suitable window size will be four.Thus, three locations are wasted, whereas this method takes advantage of all available memory locations.Thus, this method utilizes the available memory in a more efficient way than the other window methods and as well as the FW method. Algorithm 13is used to produce the proposed representation. It is a two-pass algorithm; the exponent is converted first to a signed binary representation then to the window representation.

```
Algorithm 13: Left-to-right Khabbazianrecoding
Input     : $m, k = \sum_{i \le i < l} b_i 2^i, b_i \in \{0,1\}$
Output    : a sequence of pairs$\{(k_i, e_i)\}_{i=0}^{d-1}$such that $0 \le e_i \le l$,
            $k_i \in \{\pm 1, \pm 3, \ldots, \pm(2m-1)\}, and k = \sum_{0 \le i < d} k_i 2^{e_i}$

Suppose that$b_l = b_{-1} = 0$ and let$b_i' = b_{i-1} - b_i for\ 0 \le i \le l$,so we have
$b_i' \in \{0,1,-1\}$and$(b_l', b_{l-1}', \ldots, b_0') = (b_{l-1}, b_{l-2}, \ldots, b_0)_2$
$i \leftarrow l, j \leftarrow 0$
While$i \ge 0$
        If$(b_i' = 0)$ $i \leftarrow i - 1$
        Else
        Let$t$ min int such that$|(b_i', b_{i-1}', \ldots, b_t')_2|$ is an odd int less than $2m$
        $k_j \leftarrow (b_i', b_{i-1}', \ldots, b_t')_2, e_j \leftarrow t, i \leftarrow t - 1, j \leftarrow j + 1$

Return $\{(k_0, e_0), (k_1, e_1), \ldots, (k_{d-1}, e_{d-1})\}$
```

**Muir Method**

A new representation that has the same digit set as the wNAF for $w > 2$ was proposed by Muir and Stinson (2005).The algorithmcan be deduced in both directions.In this thesis, this method will be referred to as the MU method.Muir and Stinson also proved the minimality property for wNAF(Avanzi, 2005, Muir and Stinson, 2006).In addition, they showed that any $D_w$ representation of an integer, with the minimality property, is at most one bit longer than its binary representation(Muir and Stinson, 2006). One of the major drawbacks of this method is in determining the closest element to $k$ in the first step inside the while loop of the algorithm. The authors explained a long procedure to do this. But in the third section of their paper they explained another procedure for deducing the key, which depends on a lookup table.

```
Algorithm 14: Left-to-right Muir recoding
Input     : $w \ge 2, k$
Output    :$k' = b_l, b_{l-1}, \ldots, b_0, b_i \in D_w = \{0, \pm 1, \pm 3, \ldots, 2^{w-1} - 1\}$
Remark    : $C_w = \{d \cdot 2^i : d \in D_w\{0\}, i \ge 0\}$

$k' \leftarrow \epsilon$($\epsilon$denotes empty string)
While$k \ne 0$
        $c \leftarrow$ an element$in C_w$closest to$k$
        append digits to$k'$according to the value of$c$
        $k \leftarrow k - c$

Return $(k')$
```

### 2.2.1 ZOT Method

Two positional numbering systems; redundant big-digit numbering system BDNS and non-redundant ZOT-binary numbering system were proposed by (Jahani, 2009). The hamming weight of ZOT-binary numbering system is 21.8% which is $\frac{m}{4.6}$ compared to $\frac{m}{2}$ for binary and $\frac{m}{3}$ for NAF(Jahani, 2009).Any positive binary integer $k$can be converted to ZOT-binary representationright-to-left or left-to-right. The length of an integer $k$ is greater than or equal to the length of its ZOT-binary representation; i.e. $\log_2 k_{ZOT} \le \log_2 k_2$.
The ZOT-binary numbering systemhad been slightly modifiedbyAlmimi et al. (2015)so that it can be used in elliptic curve multiplication.Given $k_2 = (k_{n-1}, k_{l-2}, \ldots, k_1, k_0)_2$, $k_{ZOT} = (\hat{b}_{m-1}, \hat{b}_{m-2}, \ldots, \hat{b}_1, \hat{b}_0)$is called the ZOT form of the integer $k$, where $\hat{b}_i = (t_i, g_i)$, $g_i$ is the bit-length of the big-digit, and the type of big-digit $t_i \in \{0,1,2\}$. The big-digits, big-zero ($Z_n$), big-one ($O_n$) and big-two ($T_n$) are denoted by 0,1, and 2 consequently, and $n$ represents the bit-length of each big-digit. Algorithm 15is the pseudo code for the previous procedure.

Algorithm 15ZOT recoding method

```
$Input k = \sum_{i=0}^{n-1} k_i 2^i, k_i \in \{0,1\}$
$Output k_{ZOT} = \{\hat{b}_{m-1}, \hat{b}_{m-2}, \dots, \hat{b}_1, \hat{b}_0\}$
$e = 0$
$while i < n$
        $if the sequence(k_i, \dots, k_j) = bigOne, where j \geq i$
                $k_{ZOT}[e].t = 1$
        $else if the sequence = bigTwo$
                $k_{ZOT}[e].t = 2$
        $else if the sequence = bigZero$
                $k_{ZOT}[e].t = 0$
                $k_{ZOT}[e].g = k_{ZOT}[e-1].g$
        $k_{ZOT}[e].l = j - i + 1$
        $Increment(e, i)$
$return k_{ZOT}$
```

**Single scalar multiplication using ZOT recoding method (ZOTEC)**

Algorithm 16isbasic version of the single scalar multiplication method that is based on ZOT recoding methodto calculate the quantity $k_{ZOT}P$, where $P \in E(F_p)$(Almimi et al., 2015).

Algorithm 16: Scalar multiplication using ZOT recoding (ZOTEC)

```
$Input: G, k_{ZOT} = \{\hat{b}_{m-1}, \hat{b}_{m-2}, \dots, \hat{b}_1, \hat{b}_0\}$
$Output: Q = k_{ZOT}G$
$Q = \infty$
$For i(m-1 \; to \; 0)$
      $If t_i \neq zero then$
            $Q = Q + \hat{b}_i(G)$
      $Else$
            $Q = 2^{g_i}(Q)$
$return Q$
```

**Window method of ZOT**

Usually, a radix-2 representation of $k$ is called window representation if $w \geq 2$ and the window values are in the digit set $D_w = \{\mp 1, \mp 3, \dots, \mp 2^w - 1\}$. A new window-based single scalar multiplication method over prime fields and using affine coordinates is proposed(Mimi et al., 2013).Let $k_2 = \sum_{i=0}^{n-1} a_i 2^i, a_i \in \{0,1\}$ be the binary representation of an integer $k$. Then $k_w = \sum_{i=0}^{l} b_i 2^i, b_i \in \{0\} \cup D_w, D_w = \{\mp 1, \mp 3, \dots, \mp(2^{w-1} - 1)\}$, is the window NAF representation of $k$.

```
Algorithm 17: Window Big-Digit Recoding
Input: $k_2 = \sum_{i=0}^{n-1} a_i 2^i, a_i \in \{0,1\}$
Output: $k_{wBD} = \sum_{i=0}^{m-1} \hat{b}_i, \hat{b}_i = (t_i, l_i), t_i \in \{0,1,2\}, l_i \le w for t_i \ne 0$
Remark: $\epsilon$ denotes empty string

$e \leftarrow 0$
while $(i < n)$
  $i \leftarrow j$
  find the largest j ≤ w such that $O \leftarrow (a_i, \ldots, a_{i+j}), a_u = 1, \forall i \le u \le i+j$
  $if (O \ne \epsilon) k_{wBD}[e].t = 1$
  find the largest j ≤ w such that $T \leftarrow (a_i, \ldots, a_{i+j}), a_u * a_{u+1} = 0, \forall i \le u < i+j$
  $if (T \ne \epsilon) k_{wBD}[e].t = 2$
  find the largest j such that $Z \leftarrow (a_i, \ldots, a_{i+j}), a_u = 0, \forall i \le u \le i+j$
  $if (Z \ne \epsilon)$
    $k_{BD}[e].t \leftarrow 0$
    $k_{BD}[e].l \leftarrow j + 1$
    $k_{wBD}[e].l \leftarrow k_{wBD}[e].l + k_{wBD}[e-1].l$
  Increment$(e, i)$

Return $k_{wBD}$
```

Algorithm 17is used to convert a key into its wBDrepresentation$k_{wBD} = \sum_{0 \le i < m} \hat{b}_i, \hat{b}_i = (t_i, l_i), t_i \in \{0,1,2\}, l_i \le w$ for$t_i \ne 0, l_i \in N^+$. The length of big-zero is equal to the big-zero digit length in addition to the previous big-digit length. This consideration helps in improving the window method and repeating doublings.

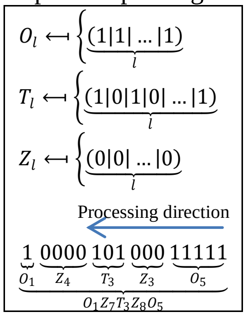
$O_l \leftarrow \{\underbrace{(1|1| \ldots |1)}_{l}$
$T_l \leftarrow \{\underbrace{(1|0|1|0| \ldots |1)}_{l}$
$Z_l \leftarrow \{\underbrace{(0|0| \ldots |0)}_{l}$
Processing direction
$\underbrace{\underbrace{1}_{O_1}\ \underbrace{0000}_{Z_4}\ \underbrace{101}_{T_3}\ \underbrace{000}_{Z_3}\ \underbrace{11111}_{O_5}}_{O_1 Z_7 T_3 Z_8 O_5}$

Figure 6. Big-digits Processing and Recognition

The conversion from binary to window BD representation is described in Figure . A contiguous sequence of nonzero digits is converted to either big-one or big-two, while the contiguous sequence of zero digits is converted to big-zero. The hamming weight of ZOT-binary number system is $\frac{n}{4.6}$ compared to $\frac{n}{2}$ for binary , $\frac{n}{3}$ for NAF and $\frac{n}{w+1}$ for wNAF (Jahani, 2009), where $n$ is the bit-length of $k$. The wBD multiplication method is provided in Algorithm 18.

```
Algorithm 18: Window BD Single Scalar EC  Point Multiplication
Input: $G \in E(F_p), w \ge 2, k_{wBD} = \sum_{0 \le i < n} \hat{b}_i, \hat{b}_i = (t_i, l_i), t_i \in \{0,1,2\}, l_i \le w$
Output: $Q = k_{wBD} G$

Precompute the points:
    $D_w = \{x: x = 2^i - 1, 1 \le i \le w\} \cup \{y: y = \frac{1}{3}(2^{i+1} - 1), 3 \le odd(i) \le w\}$

$Q \leftarrow \infty$
$For i (n - 1 \ to \ 0)$
    $Q \leftarrow \begin{cases} Q + \hat{b}_i(G), & t_i \ne 0 \\ 2^{g_i}(Q), & t_i = 0 \end{cases}$

Return $(Q)$
```

The average length of zero runs in binary, ZOT-binary, NAF, or wNAFrecoded keys is computed using the formula with a key length of $n = \log_2 k$:

$$Z'(k) = \frac{1}{C} \sum_{i=0}^{n-1} 1 - |k_i|, \quad C = \sum_{i=1}^{n-1} z(i),$$

$$z(i) = \begin{cases} 1, & b_i \ne 0, b_{i-1} = 0 \\ 0, & b_i = 0 \end{cases}$$

.
Whereas average length of zero runs for BD recoded keys is computed using the following formula:

$$Z^{'}(k) = \frac{\sum_{i=0}^{n-1} l_i z(i)}{\sum_{i=0}^{n-1} z(i)}, z(i) = \begin{cases} 1, & t_i = 0 \\ 0, & t_i \neq 0 \end{cases}$$

The set of precomputed windows is $D_w = \{x: x = 2^i - 1, 1 \leq i \leq w\} \cup \{y: y = \frac{1}{3}(2^{i+1} - 1), 3 \leq \text{odd}(i) \leq w\}$. Whereas the number of precomputed points are $\left(w + \left\lfloor \frac{w-1}{2} \right\rfloor\right)$. Since the BD is a bidirectional recoding method, the memory required to store the recoded exponent is $w$ whenever left-to-right method used. The cost of wBD multiplication method is computed using the formula $\text{n}[D] + \frac{n}{w(k_{wBD})}[A], n = \log_2 k$. Finally, only one pass is required to transform the exponent $k$ from its binary format to its wBD formant.
Table 2.lists window methods and the required memory for storing the exponent $k$ in addition to the recoding direction and the number of passes required by an algorithm to convert the exponent $k$ to its new format. Bidirectional methods in addition to left-to-right methods require less memory to store the recoded exponent than right-to-left methods. Number of passes means that the conversion requires two sub-conversions for the recoding process. For example if method $A$ converts $k$ to singed $k$' next it converts singed $k'$ to window singed $k$'' then this method requires two passes. As it can be seen from the table, the conversion of the exponent $k$ from its binary representation to its wBD representation requires one pass which is considered one of the advantages of wBD method. Moreover, it is a bidirectional method which also does not store the whole recoded key and can be computed in any direction depending on the available amount of memory.

Table 2. Window Methods Summary

| Method | Year | Reference | Required memory | Recoding direction | n-Pass |
|---|---|---|---|---|---|
| m-ary | 1939 | (Blake et al., 1999) | $O(w)$ | BI | 1 |
| swNAF | NA | (Darrel et al., 2003) | $O(n)$ | RTL | 1 |
| KTNS | 1993 | (Koyama and Tsuruoka, 1993) | $O(n)$ | RTL | 2 |
| wNAF | 1997 | (Solinas, 1997) | $O(n)$ | RTL | 1 |
| wMOF | 2004 | (Okeya et al., 2004) | $O(w)$ | BI | 2 |
| FW | 2002 | (Möller, 2002 Seoul, Korea,), (Möller, 2004) | $O(n)$ | RTL | 1 |
| KLNS | 2005 | (Kong and Li, 2005) | $O(n)$ | RTL | 2 |
| KH | 2005 | (Khabbazian et al., 2005) | $O(w)$ | LTR | 2 |
| MU | 2005 | (Muir and Stinson, 2005) | $O(w)$ | LTR | 1 |
| wBD | 2012 | (Mimi et al., 2013) | $O(w)$ | BI | 1 |

The mathematical representation of $k_{wBD}$ in addition to other window methods is depictedin

Table 3.

Table 3. Mathematical Key Representation for Some Window Methods

| Method | Representation |
|---|---|
| m-ary | $k_{mary} = \sum_{i=0}^{l-1} k_i m^i, m = 2^w, k_i < 2^w, l = \left\lceil \frac{\log_2 k}{w} \right\rceil$ |
| swNAF | $k_{swNAF} = \sum_{i=0}^{l-1} k_i 2^i, k_i isodd, \lvert k_i \rvert < 2^{w-1}, k_{l-1} \neq 0, \lceil \log_2 k \rceil \leq l \leq \lceil \log_2 k \rceil + 1$ |
| KTNS | $k_{KTNS} = \sum_{i=0}^{l} k_i 2^i, k_i isodd, \lvert k_i \rvert \leq 2^w - 3, l \leq \lceil \log_2 k \rceil + 1$ |
| wNAF | $k_{wNAF} = \sum_{i=0}^{l-1} k_i 2^i, kisodd, \lvert k_i \rvert < 2^{w-1}, k_{l-1} \neq 0, l \leq \lceil \log_2 k \rceil + 1$ |
| wMOF | $k_{wMOF} = \sum_{i=0}^{l-1} k_i 2^i, kisodd, \lvert k_i \rvert < 2^{w-1}, l \leq \lceil \log_2 k \rceil + 1$ |
| FW | $k_{FW} = \sum_{i=0}^{l-1} k_i 2^i, kisodd, \lvert k_i \rvert \leq 2^w - 3, l \leq \lceil \log_2 k \rceil + 1$ |
| KLNS | $k_{KLNS} = \sum_{i=0}^{l-1} k_i 2^i, kisodd, \lvert k_i \rvert \leq \frac{5}{6} \cdot 2^w - \frac{1}{3}, l \leq \lceil \log_2 k \rceil + 1$ |
| KH | $k_{KH} = \sum_{i=0}^{l-1} k_i 2^i, kisodd, \lvert k_i \rvert \leq (2m - 1), l \leq \lceil \log_2 k \rceil + 1,$ where$m \leq C$, and $C$ is the maximum number of points that can be stored |

| | |
|---|---|
| MU | $k_{MU} = \sum_{i=0}^{l-1} k_i 2^i, kisodd, \lvert k_i \rvert < 2^{w-1}, l \le \lceil \log_2 k \rceil + 1$ |
| wBD | $k_{wBD} = \sum_{i=0}^{i<m} \hat{b}_i, \hat{b}_i = (t_i, l_i), t_i \in \{0,1,2\}, l_i \le w$ for$t_i \neq 0, l_i \in N^{+}$. |

## IV. Discussion

In the current chapter, the basic concepts of ECC are introduced. ECC is based on finite field arithmetic. Therefore, an introduction about finite field arithmetic is given. Basic and composite EC operations are explained. Affine coordinates, which are used in this study, are also explained in addition to other point representation systems. Single EC multiplication methods are classified into on-the-fly and pre-computation methods. Some of the on-the-fly single scalar EC multiplication methods have been introduced in the previous sections. Signed methods are more efficient than the Classical unsigned binary method. If memory is not limited, precomputationscan be done to speed up the EC methods. In the following sections, window-based methods that employ pre-computation in their techniqueswill be explored.

Window-based methods were introduced in the previous sections. These techniques require memory to store the precomputed points (windows). These methods can be classified according to two criteria: flexibility of memory usage and direction. Left-to-right recoding is preferred since it can be merged with the EC multiplication method. Thus it is considered a memory efficient method since it requires only $w$ digits to be known when applying the multiplication technique. The memory flexible methods are those which can limit their number of precomputed points according to the available memory such as FW and KH.

Table IV.lists window methods and the required memory for storing exponent $k$ in addition to the recoding direction and the number of passes required by an algorithm to convert exponent $k$into its new format. Bidirectional methods in addition to left-to-right methods require less memory to store the recoded exponent than right-to-left methods. The number of passes means that the conversion requires two sub-conversions for the recoding process. For example, if method $A$ converts $k$ to signed $k$' then converts signed $k$' to window signed $k$'' then this method requires two passes.

Table IV. Summary of the window schemes

| Method | Year | Reference | Required Memory | Recoding Direction | n-Pass |
|---|---|---|---|---|---|
| $m$-ary | 1939 | (Blake et al., 1999) | $O(w)$ | BI | 1 |
| swNAF | NA | (Darrel et al., 2003) | $O(n)$ | RTL | 1 |
| KTNS | 1993 | (Koyama and Tsuruoka, 1993) | $O(n)$ | RTL | 2 |
| wNAF | 1997 | (Solinas, 1997) | $O(n)$ | RTL | 1 |
| wMOF | 2004 | (Okeya et al., 2004) | $O(w)$ | BI | 2 |
| FW | 2002 | (Möller, 2002 Seoul, Korea,), (Möller, 2004) | $O(n)$ | RTL | 1 |
| KLNS | 2005 | (Kong and Li, 2005) | $O(n)$ | RTL | 2 |
| KH | 2005 | (Khabbazian et al., 2005) | $O(w)$ | LTR | 2 |
| MU | 2005 | (Muir and Stinson, 2005) | $O(w)$ | LTR | 1 |

The ZOT-number system is a new number system proposed by Jahani and Samsudin (2013). It was introduced since the proposed methods rely on the Big-Digit recoding method which was originally based on the ZOT-number system.

During this research, it is also found that $\alpha$that was considered byEisenträger et al. (2003) was too small since they did not rely on Montgomery enhancement. Thus their improvement will be reduced if more reliable values for the previous ratio are considered. Although the field complexity of the Dahmen method is better than that for the Classical method for computing the points $3P$, $5P$ and $7P$, its time complexity is worse than that of theClassical method. Thus, the Dahmen method is not suitable for on-the-fly computation. It is only suitable when precomputations are allowed according to the available memory. On the other hand, Ciet methods for computing $2P + Q, 3P + Q$ are the best known methods.

For the computation of direct doublings, the Sakai method will be used as the most efficient method to be implemented and merged with the proposed methods. The break-even point of the Sakai method is represented by $\frac{3.6\mathrm{d}+1.8}{\mathrm{d}-1}$. The Hamming Weight of the complementary recoding method is not determined in the literature. Thus, it is experimentally found in this study. It was said by Chang et al. (2003) that if $\mathrm{W(k)} = \frac{\log_2 \mathrm{k}}{2}$ then the possible value of the complementary recoded key will be $\mathrm{W(k_{CR})} = \frac{\log_2 \mathrm{k}}{4}$. It is found that their assumption is not correct.

Let $x = (a_{n-1}a_{n-2} \dots a_1a_0)_2$ , where $a_i = 1 \; \forall \; 0 \leq i < n$.Morain and Olivos (1990) observed that they could save one addition in EC computation of the form *xP*. It is easy to see that the exponent is represented as follows: $x = 2^n - 1$. Thus the exponent is replaced by $(y - 1)$, where $y = 2^n$ . Thus the quantity $xP$ is calculated as follows: $xP = yP - P$. The cost of computing *xP* using the Classical method is $(n - 1)[D] + (n - 1)[A]$. On the other hand, the cost will be $(n[D] + [A])$ if the previous enhancement is employed. The relation between both costs is expressed by the inequality $n[D] + [A] < (n - 1)([D] + [A])$. It is proved that a = 4 is the minimum length of nonzero sequence of bits that can take advantage of this enhancement. Therefore, the Morain method is suitable whenever the nonzero sequence has a length of 4 bits or more if basic EC operations are used.

## Acknowledgment
This research is funded by the Deanship of Research and Graduate Studies in Zarqa University /Jordan.

## References

[1]. ALMIMI, H., SAMSUDIN, A. & JAHANI, S. 2015. Elliptic-curve scalar multiplication algorithm using ZOT structure. Security and Communication Networks, 8, 1141-1154.

[2]. AVANZI, R. 2005. A Note on the Signed Sliding Window Integer Recoding and a Left-to-Right Analogue. Selected Areas in Cryptography, vol. 3357, pp. 130-143.

[3]. BALASUBRAMANIAM, P. & KARTHIKEYAN, E. 2007. Elliptic curve scalar multiplication algorithm using complementary recoding. Applied Mathematics and Computation, vol. 190, pp. 51-56.

[4]. BLAKE, I. F., SEROUSSI, G. & SMART, N. P. 1999. Elliptic Curves in Cryptography, Cambridge University Press.

[5]. BOOTH, A. D. 1951. A Signed Binary Multiplication Technique. Quarterly J. Mechanics and Applied Mathematics, vol. 4, pp. 236-240.

[6]. CHANG, C.-C., KUO, Y.-T. & LIN, C.-H. 2003. Fast algorithms for common-multiplicand multiplication and exponentiation by performing complements. 17th International Conference on Advanced Information Networking and Applications, pp. 807-811.

[7]. COHEN, H. & FREY, G. 2006. Handbook of Elliptic and Hyperelliptic Curve Cryptography, Chapman & Hall/CRC.

[8]. COHEN, H., MIYAJI, A. & ONO, T. 1998. Efficient elliptic curve exponentiation using mixed coordinates. Proceedings of the International Conference on the Theory and Applications of Cryptology and Information Security: Advances in Cryptology, vol. 1514, pp. 51-65.

[9]. DARREL, H., ALFRED, J. M. & SCOTT, V. 2003. Guide to Elliptic Curve Cryptography, Springer-Verlag New York, Inc.

[10]. DIMITROV, V., IMBERT, L. & MISHRA, P. K. 2008. The double-base number system and its application to elliptic curve cryptography. Mathematics of Computation, vol. 77, pp. 1075-1104.

[11]. DOCHE, C. & IMBERT, L. 2006. Extended Double-Base Number system with applications to elliptic curve cryptography. Progress in Cryptology - INDOCRYPT, vol. 4329, pp. 335-348.

[12]. EISENTRÄGER, K., LAUTER, K. & MONTGOMERY, P. L. 2003. Fast elliptic curve arithmetic and improved Weil pairing evaluation. Topics in Cryptology, Lecture Notes in Computer Science, vol. 2612, pp. 343-354.

[13]. GORDON, D. M. 1998. A survey of fast exponentiation methods. Journal of Algorithms, vol. 27, pp. 129-146.

[14]. JAHANI, S. 2009. ZOT-MK: A New Algorithm for Big Integer Multiplication. MSc MSc, Universiti Sains Malaysia.

[15]. JAHANI, S. & SAMSUDIN, A. 2013. ZOT-Binary: A New Numbering System with an Application on Big-Integer Multiplication. Journal of Theoretical and Applied Information Technology (JATIT), vol. 48, pp. 029 - 040.

[16]. JEDWAB, J. & MITCHELL, C. J. 1989. Minimum weight modified signed-digit representations and fast exponentiation. Electronics Letters, vol. 25, pp. 1171-1172.

[17]. JOYE, M. & SUNG-MING, Y. 2000. Optimal left-to-right binary signed-digit recoding. IEEE Transactions on Computers, vol. 49, pp. 740-748.

[18]. KHABBAZIAN, M., GULLIVER, T. A. & BHARGAVA, V. K. 2005. A new minimal average weight representation for left-to-right point multiplication methods. IEEE Transactions on Computers, vol. 54, pp. 1454-1459.

[19]. KNUTH, D. E. 1997. The art of computer programming, Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc.

[20]. KONG, F. Y. & LI, D. X. 2005. A note on signed binary window algorithm for elliptic curve cryptosystems. Cryptology and Network Security, Proceedings, vol. 3810, pp. 223-235.

[21]. KOYAMA, K. & TSURUOKA, Y. 1993. Speeding up Elliptic Cryptosystems by Using a Signed Binary Window Method. Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, pp. 345-357.

[22]. MIMI, H., SAMSUDIN, A. & JAHANI, S. 2013. Elliptic Curve Point Multiplication Algorithm Using Precomputation. WSEAS Transactions on Computers, 12, .

[23]. MIYAJI, A., ONO, T. & COHEN, H. 1997. Efficient elliptic curve exponentiation. ICICS '97: Proceedings of the First International Conference on Information and Communication Security, vol. 1334, pp. 282-290.

[24]. MÖLLER, B. Improved techniques for fast exponentiation. ICISC'02: Proceedings of the 5th international conference on Information security and cryptology, 2002 Seoul, Korea, . Springer-Verlag, pp. 298-312.

[25]. MÖLLER, B. 2004. Fractional windows revisited: Improved signed-digit representations for efficient exponentiation. ICISC'04: Proceedings of the 7th international conference on Information Security and Cryptology, vol. 3506, pp. 137-153.

[26]. MORAIN, F. & OLIVOS, J. 1990. Speeding up the Computations on an Elliptic Curve Using Addition-Subtraction Chains. Rairo-Informatique Theorique Et Applications-Theoretical Informatics and Applications, vol. 24, pp. 531-544.

[27]. MUIR, J. A. & STINSON, D. R. 2005. New Minimal Weight Representations for Left-to-Right Window Methods. CT-RSA'05: Proceedings of the 2005 international conference on Topics in Cryptology, vol. 3376, pp. 366-383.

[28]. MUIR, J. A. & STINSON, D. R. 2006. Minimality and other properties of the width-w nonadjacent form. Mathematics of Computation, vol. 75, pp. 369-384.

[29]. OKEYA, K., SCHMIDT-SAMOA, K., SPAHN, C. & TAKAGI, T. 2004. Signed binary representations revisited. Proceedings of CRYPTO’04, vol. 3152, pp. 123-139.

[30]. REITWIESNER, G. W. 1960. Binary Arithmetic. Advances in Computers, vol. 1, pp. 231-308.

[31]. SCHMIDT-SAMOA, K., SEMAY, O. & TAKAGI, T. 2006. Analysis of fractional window recoding methods and their application to elliptic curve cryptosystems. IEEE Transactions on Computers, vol. 55, pp. 48-57.

[32]. SOLINAS, J. A. 1997. An improved algorithm for arithmetic on a family of elliptic curves. CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology, vol. 1294, pp. 357-371.

[33]. SOLINAS, J. A. 2000. Efficient Arithmetic on Koblitz Curves. Designs, Codes and Cryptography, vol. 19, pp. 195-249.

[34]. STALLINGS, W. Cryptography and Network Security: Principles and Practice. 2013. Prentice Hall.

[35]. TSAUR, W.-J. & CHOU, C.-H. 2005. Efficient algorithms for speeding up the computations of elliptic curve cryptosystems. Applied Mathematics and Computation, vol. 168, pp. 1045-1064.