Design and Implementation of Block Method for Computing NAF

Harsandeep Brar
Centre for Development of Advanced Computing
Mohali, working as Project Associate

Rajpreet Kaur
Baba Banda Singh Bahadur Engineering college
Fatehgarh Sahib, working as Sr. Lecturer

ABSTRACT
Elliptic curve based cryptosystem is an efficient public key cryptosystem, which is more suitable for limited environments. The performance of elliptic curve cryptosystem heavily depends on an operation called point multiplication. It is the multiplication of a scalar with the given point on the curve. Scalar multiplication is faster by using signed binary representation as compared to binary representation. In this paper "Block Method" for computing NAF is proposed. The Proposed method is more efficient as compared to standard method for computing NAF.

The paper presents the comparative study of both standard and block methods for computing NAF. In this paper we have examined that Overall computation for Point Multiplication operation with NAF method can be made more effective by improving speed of calculating the NAF Part.

Keywords
Elliptic curve cryptography, Point multiplication, Binary method, NAF, Block method

1. INTRODUCTION
Elliptic curve cryptography was introduced by Victor Miller and Neal Koblitz [1] in 1985. The popularity of elliptic curve cryptography is due to the determination that is based on a harder mathematical problem than other cryptosystems. It is gaining wide acceptance as an alternative to the conventional public key cryptosystem such as RSA [2], DSA [3].ECC offers the same level of security with smaller key size and it leads to the better performance in limited environments like cellular phones, PDA, sensor networking, etc. For example, ECC with a key size of 160 bits provides the same level of security as RSA with a key size of 1024 bits [14]. Another advantage that makes ECC more attractive is the possibility of optimizing the arithmetic operations in the underlying field [4].

Scalar multiplication is the central operation of elliptic curve cryptosystem. It involves the computation of kP where k is the secret key (scalar) and P a point on the elliptic curve. For any k, the calculation of kP is broken down into a series of additions and doublings. The speed of scalar multiplication plays an important role in the efficiency of whole system. Each value of the 'a' and 'b' gives a different elliptic curve.

The mathematical operations of ECC is defined over the elliptic curve $Y^2=x^3+ax+b$ where $4a^3+27b^2 \neq 0$ [5]. The public key is a point in the curve and the private key is a random number.

The public key is obtained by multiplying the private key with the generator point G in the curve.The generator point G, the curve parameters 'a' and 'b' together with few more constants constitutes the domain parameter of ECC.

2. ECC ON FINTE FIELDS
To make operations on elliptic curve accurate and more efficient, the curve cryptography is defined over two finite fields.

- Prime field F_p
- Binary field F_2^m

The field is chosen with finitely large number of points suited for cryptographic operations [5].

2.1 EC on Prime field
The equation of the elliptic curve on a prime field F_P is given by $y^2 \bmod p = x^3 + ax + b \bmod p$, where $4a^3 + 27b^2 \bmod p \neq 0$. Here the elements of the finite field are integers between 0 and p – 1. All the operations such as addition, substation, division, multiplication involves integers between 0 and p – 1. The prime number p is chosen such that there is finitely large number of points on the elliptic curve to make the cryptosystem secure.

2.2 EC on Binary field
The equation of the elliptic curve on a binary field F_2^m is given by $y^2 + xy = x^3 + ax^2 + b$, where $b \neq 0$ [15]. Here the elements of the finite field are integers of length at most m bits. These numbers can be considered as a binary polynomial of degree m – 1. In binary polynomial the coefficients can only be 0 or 1. All the operation such as addition, substation, division, multiplication involves polynomials of degree m – 1 or lesser [5]. The m is chosen such that there is finitely large number of points on the elliptic curve to make the cryptosystem secure.

3. POINT MULTIPLICATION
In point multiplication a point P on the elliptic curve is multiplied with a scalar k using elliptic curve equation to obtain another point Q on the same elliptic curve i.e. KP=Q

Point multiplication is achieved by two basic elliptic curve operations [5]

- Point addition, adding two points J and K to obtain another point L i.e., L = J + K.
- Point doubling, adding a point J to itself to obtain another point L i.e. L = 2J.

Here is a simple example of point multiplication.
Let P be a point on an elliptic curve. Let k be a scalar that is multiplied with the point P to obtain another point Q on the curve. i.e. to find Q = kP.

If k = 19 then kP = 19P = 2 (2 (2 (2P)) +P) +P.

Thus point multiplication uses point addition and point doubling repeatedly to find the result. The above method is called 'double and add' method for point multiplication. There are other efficient methods for point multiplication such as NAF (Non – Adjacent Form). In the following sections we review some methods for computing scalar multiplication.

3.1 Binary Method

For computing kp, the simplest method is binary method [7].Binary is the traditional scalar multiplication method based on the binary expansion of the scalar k using (0, 1).The integer k is represented as

$k=k_{n-1}2^{n-1}+k_{n-2}2^{n-2}+\ldots\ldots+k_1+k_0$

Where ki € {0, 1}, n = 0, 1, 2,……., n -1.

That is

$$k=\sum_{j=0}^{l-1} k_j 2^j, \text{ where } k_j \in \{0, 1\}.$$

This method is called binary method [6] which scans the bits of k either from left-to-right or right- to-left. The binary method for the computation of kP is given in the following Algorithm 1.

Algorithm 1: Binary method

Input: Binary representation of k and point P

$k = (k_{n-1}\ldots k_1k_0)_2$

Output: kP

1. R ← P
2. For i = n-2 to 0 do
   - 2.1 R←2R (Doubling)
   - 2.2 If k_i =1 then
     - R = R + P (Addition)
   - 2.3 i ← i −1
3. Return R

The cost of multiplication depends on the length of the binary representation of k and the number of 1s in this representation. If the representation $(k_{n-1}\ldots k_1k_0)_2$ has k_{n-1} # 0 then the number of doubling operation is (n - 1) and the number of addition operations is one less than the number of non-zero digits in $(k_{n-1}\ldots k_1k_0)_2$. The number of non-zero digits is called the Hamming weight of scalar representation. In an average, binary method requires (n -1) doublings and (n -1)/ 2 additions.

Example1. Let integer k = 35 and point P on the elliptic curve. The binary representation of 35 is $(100011)_2$.

So computation of 35P using algorithm1would is as follows:

35P= 2(2(2(2(2P))) +P) +P

It requires 5 doublings and 2 additions. Whenever the bit is 1, two elliptic curve arithmetic operations such as ECDBL and ECADD will be made and if it is 0, only one operation, ECDBL is required.

So if we reduce the number of 1s in the scalar representation or hamming weight, we could speedup the above computation.

3.2 NAF Method

Another method for scalar multiplication was proposed by Booth [8], called signed binary method. The property of this representation is that, of any two consecutive digits, at most one is non-zero.

Before discussing this method we first discuss about non-adjacent form (NAF) that is the basis of this method [7]. An improved method for computing

kP can be obtained from the following facts:

- Every integer *k* has a unique representation of the form

$$k=\sum_{j=0}^{l-1} k_j 2^j,$$

Where each kj € {-1, 0, 1}.such that no two consecutive digit are nonzero. This representation is known as non-adjacent form (NAF).

- The expected weight of a NAF of length *n* is n/3.

A procedure for computing NAF for a positive integer k is described in the algorithm 2.

Algorithm 2: Standard method for computing NAF of an integer

Input: Positive integer *k*

Output: NAF (*k*)

1. i ← 0
2. While $k \geq 1$ do
   - 2.1 If *k* is odd then: *k*i ← 2 - (*k* mod 4),
     - $k \leftarrow k - k_i$
   - 2.2 Else: *k*i ← 0
   - 2.3 $k \leftarrow k/2$, i← i +1
3. Return $(k_{i-1}, k_{i-2},\ldots.., k_1, k_0)$.

The average hamming weight of signed binary representation is n/3 and it has the lower hamming weight than the binary representation. The binary method is revised accordingly and the new algorithm is called addition–subtraction method [10] given in Algorithm 3.

Example2. NAF of k = 996 is

$996= (10000\text{-}100100)_{NAF}$

$= 2^{10}- 2^5 +2^2$

=1024 -32+4

The hamming weight is 3.

The binary representation of 996 is $(1111100100)_2$ $=2^9+2^8+2^7+2^6+2^5+2^2$

=512+256+128+64+32+4

The hamming weight is 6. By NAF hamming weight of k is reduced from 6 to 3 which improve the speed of the scalar multiplication.

Algorithm 3: NAF Method for Point Multiplication
Input: NAF of a Positive integer *k* and P

Output: kP

1. R ← P

2. For i= n −2 to 0 do

2.1 R←2R

2.2 if k_i = 1 then R←R+P

2.3 if k_i = −1 then R←R-P

2.4 i ← i −1

3. Return R

The addition-subtraction method is analogue of the binary method, performs an addition or subtraction depending on the sign of digit of non-adjacent form (NAF) of , scanned from left to right This algorithm performs (n-1) doublings and (n-1)/3 additions in an average[9].

3.3 Proposed Block Method for computing NAF

As we have examined above that the NAF method for point multiplication is more efficient than the binary method as it reduces the hamming weight for a given input and thus improves the speed of multiplication.

NAF method includes two parts for performing Point multiplication operations.

- First is calculation of NAF for given input
- Second is computation of Point multiplication operation using NAF obtained.

Overall computation for Point Multiplication operation with NAF method can be made more effective by improving speed of calculating the NAF Part.

Our proposed Block method improves the speed for computing NAF over the standard method. According to our proposed method the procedure for converting the scalar k into signed binary representation is as follows:

a) The first step is to partition the input into blocks of binary of equal size.

b) If there is an odd block at the end, sufficient padding bit will be appended to the left of this block to make all blocks equal in size.

c) Then get the index for each block of binary to extract the NAF value for each block from look up table

d) Perform the combine part for the blocks to get Final result in NAF

In our proposed method, look up table is used to store NAF values for the blocks. In our case block size used r=8bits and NAF values are stored using 9bytes.Look up table contains 256 values of NAF for the present case with 8bits block size as 2^8=256.Few such NAF values are shown in Table 1.

Table 1. Look up Table

Index	NAF Value
0	000000000
1	000000001
2	000000010
3	00000010-1
4	000000100
5	000000101
6	0000010-10
7	00000100-1
8	000001000
9	000001001
10	000001010
.	.
.	.
.	.
255	10000000-1

Algorithm 4: Block method for computing NAF of an integer
Input: A positive Integer k

Output: NAF (k)

Step1:

1.1 Define input size to be m bits

1.2 Divide the input into n blocks of binary with each block

Having equal that is r bit size

n=m/r

//always gives equal no of blocks by making each input to m bits size

1.3 Choose the least significant (rightmost) block as lower block

And all other blocks towards left are upper blocks

Step2:

Obtain NAF for each block from look up table

Step3:

3.1 Do the combine of blocks that already in NAF starting from right to left

3.2 Perform the boundary addition with MSB of lower block and LSB of upper block

3.3 Get the final combine result in NAF

Step 4:

Return $(k_m, k_{m-1}, \ldots, k_1, k_0)_{NAF}$

Point multiplication operation using proposed method can be performed with algorithm 3.

Example3. Let k=4570

The binary representation of 4570 is $(1000111011010)_2$

There are two blocks of 8bits as

Block 1=11011010

Block 2=00010001

Above blocks after making to NAF are

Block 1=100-10-1010

Block 2=000010001

Now combining the blocks:

000010001

100-10-1010

= 00001001000-10-1010

$=2^{12}+2^{9}-2^{5}-2^{3}+2^{1}$

=4096+512-32-8+2

4. COMPARISON

Our proposed method has less number of computations than the standard method of NAF .Block method computes NAF faster than the standard method for a given input. The following Table2 shows the comparison of number of iterations observed with both the standard and block method.

Table 2. Iterations

Input	Standard Method	Block method
29	35	9
235	44	9
849	50	29
2990	56	31
59837	68	33

From table 2, it is clear that the number of iterations observed with standard method is more than block method for different size of inputs. It can be seen that as the input size increases the performance of the proposed block method becomes more efficient than the standard method. Figure1 describes the performance of both methods graphically.

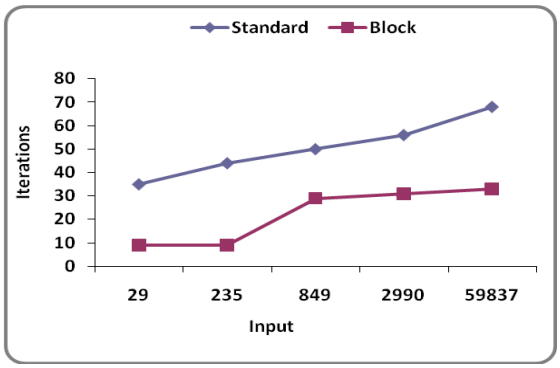

Figure-1 comparison plot of number of iterations for standard and block method with different input values

5. CONCLUSION AND FUTURE WORK

In the implementation of ECC scalar multiplication is not only the basic computation but also the most time consuming operation. Its operational efficiency directly determines the performance of ECC.

In this Paper we have proposed an efficient Block method which can be used for scalar multiplication for computing NAF of an integer more efficiently. Theoretical tasks and numerical tests reveal that this algorithm can remarkably improve the speed of computing NAF as compared to standard method. We have computed results for proposed method for input size of 16bits as shown in table 2.

Therefore in the future the block method can be implemented with large input sizes and even with larger block size in order to obtain more efficient results.

6. REFERENCES

[1] N. Koblitz, "Elliptic curve cryptosystem", Mathematics of Computation 48 (1987)203–209.

[2] R.L. Rivest, A. Shamir, L.M. Adleman, "A Method for obtaining digital signatures and public key cryptosystem", Communications of the ACM 21 (1978) 120–126.

[3] T. ElGamal, "Public key cryptosystem and a signature scheme based on discrete logarithms", IEEE Transactions on Information Theory 31 (4) (1985) 469–472.

[4] E. Al-Daoud, R, mahmod, Md. Rushdan, A. Kilicman (2002), "A new addition formula for Elliptic curves over GF (2n)", IEEE Transactions on Computers, vol. 51, no. 8, August 2002, pp. 972-975.

[5] Anoop MS, "Elliptic curve Cryptography", available at http://www.infosecwriters.com/text_resources/pdf/Elliptic_Curve_AnnopMS.pdf, 5 Jan 2007.

[6] Standard Specifications for Public Key Cryptography, IEEE Standard 1363, 2000.

[7] J. Lopez, R. Dahab (2000), "An overview of elliptic curve cryptography", Technical report, IC-00-10, May 22. Available at

http://www.dcc.unicamp.br/ic-main/public-cation - e.html.

[8] A.D. Booth, "A signed binary multiplication technique", Journal of Applied Mathematics 4 (2) (1951) 236–240.

[9] P. Balasubramaniam, E. Karthikeyan, "Elliptic Curve scalar multiplication algorithm using complementary recoding", available at www.sciencedirect.com.

[10] F. Morain, J. Olivos, "Speeding up the computations on an elliptic curve using addition–subtraction chains", RAIRO Theoretical Informatics and Applications 24 (1990), pp. 531–543.

[11] Rahim Ali, "Elliptic Curve Cryptography A new way for Encryption", IEEE 2008.

[12] Alessandro Cilardo, Luigi Coppolino, Nicola Mazzocca and Luigi Romano, "Elliptic Curve Cryptography Engineering", Proceedings of the IEEE, Vol. 94- No. 2, February 2006.

[13] G.V.S Raju, Rehan Akbani, "Elliptic Curve Cryptosystem and its applications", IEEE 2003, pp 1540-1543.

[14] Qizhi Qiu and Qianxing Xiong, "Research on Elliptic Curve Cryptography", the 8th international conference on computer supported cooperative work in design proceedings", IEEE 2003, pp 698-701.

[15] G.N.Purohit, Asmita Singh Rawat, "Efficient implementation of Arithmetic operations in ECC over Binary fields", International Journal of Computer Applications, Vol.6-No.2, September 2010.

[16] O.Srinivasa Rao, S.Pallam Setty, "Efficient mapping methods for Elliptic Curve Cryptosystems", International Journal of engineering Science and Technology Vol. 2(8), 2010, pp 3651-3656.

[17] Kristin Lauter, "The advantages of Elliptic Curve Cryptography for Wireless Security", IEEE Wireless Communications, February 2004, pp 62-67.