International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-3 Issue-1, June 2013

Scalar Multiplication Algorithms of Elliptic Curve Cryptography over GF (2^m)

Praful Kumar Singh, Mrityunjay Kumar Choudhary

Abstract-Since the inception of elliptic curve cryptography by Koblitz [1] and Miller [2] for implementing public-key protocols as the Diffie-Hellman key agreement, elliptic curve cryptography has become one of the most researched area for providing one stop reliable and secure solution in the field of cryptography. The ECC covers all relevant asymmetric cryptographic primitives like digital signature (ECDSA), key exchange and agreement protocols. Point multiplication serves as the basic building block in all ECC primitives and is the computationally most expensive operation and our analysis revolves around this concept. This paper gives an introduction to Elliptic Curve Cryptography and deals with evaluation of fast scalar multiplication with parallelization of field operation and point addition/multiplication. Elliptic curve cryptography offers best optimized solution with minimum resources like Low memory, High Throughput, low power consumption and minimum key length for the same level of security as compared to its counterpart like RSA, DSA etc. in public key cryptography domain. The work is based on the extensive research work done by Julio Lopez, Ricardo Dahab, Montgomery and other pioneer scientists and academicians in the field of elliptic curve cryptography. Given the importance of Scalar multiplication , we focused ourselves on the Fast Multiplication on Elliptic Curves over finite Binary field GF(2^m) without Pre-computation whose background is set by Julio Lopez et al. in [1], because the finite field operations can be implemented very efficiently in hardware and software.

Keywords – Elliptic Curve Cryptography, Scalar Multiplication, Encryption

I. INTRODUCTION

In 1976, Whitfield Diffie and Martin Hellman introduced the concept of public key cryptography (PKC). Since then, many implementations of it have been proposed, and many of these cryptographic hard mathematical problems, namely the integer factorization problem (IFP) and the finite field applications base their security on the intractability of discrete logarithm problem (DLP). Over the years, sub-exponential time algorithms were developed to solve these problems. As a result, key sizes grew to more than 1000 bits, so as to attain a reasonable level of security. In constrained environments where computing power, storage and bandwidth are limited, carrying out thousand-bit operations becomes an impractical approach to providing adequate security. This is most evident in hand-held devices such as the mobile phones, pagers and PDAs that have very limited processing power and battery life. Proposed independently by Neal Koblitz and Victor Miller in 1985, elliptic curve cryptography (ECC) has the special characteristic that to date, the best known algorithm that solves it runs in full exponential time.

Manuscript published on 30 June 2013.
***Correspondence Author(s)**
Mrityunjay Kumar choudhary: is currently pursuing Master degree program (M.Tech) in Information Technology in USICT, GGSIPU, India.
Praful Kumar Singh: is currently pursuing Master degree program (M.Tech) in Information Technology in USICT, GGSIPU, India.

Its security comes from the elliptic curve logarithm, which is the DLP in a group defined by points on an elliptic curve over a finite field.
This result in a dramatic decrease in key size needed to achieve the same level of security offered in conventional PKC schemes.
Elliptic curve cryptography (ECC) has become animportant branch of public key cryptography system [1]. Because of short length of key and high strength of security, ECC has become the substitute for RSA and has been increasingly specified by accredited standards organizations (IEEEP1363, ANSI X9 and ISO/IEC, etc.) as a major public key cryptosystem standards [4]. Currently, a desired security level can be attained with significantly smaller key in elliptic curve systems than what is possible with their RSA counterparts. For example, it is generally accepted that a 160 bit elliptic curve key can provide the same level of security as a 1024bit RSA key.

II. ELLIPTIC CURVE CRYPTOGRAPHY(ECC)

1.1 Elliptic Curve Cryptography over GF(p)

An elliptic curve E over GF(p) can be defined by as $y^2=x^3+ax+b$, where $a,b \in GF(p)$, and $4a^3+27b^2 \neq 0$ in the GF(p). Hence, the pair (x,y) will be a point on the curve when (x,y) satisfies the curve equation and the point at infinity is said to be on the curve.
The algebraic formula for point addition and point doubling are given as follows:

$x_3=m^2-x_1-x_2 \pmod p$
$y_3=m(x_1-x_3)-y_1 \pmod p$
$m = (y_2-y_1)/(x_2-x_1), P \neq Q$
$= (3x_1^2+a)/(2y_1), P=Q$

Where, the addition, subtraction, multiplication and division are arithmetic operations over GF(p),

1.2 Elliptic Curve Cryptography over GF(2^m)

Let E be a non-super singular curve defined over a binary field GF(2^m),
E: $y^2+xy=x^3+ax^2+b$ where $a,b \in GF(2^m)$ and $b \neq 0$.
Let $P=(x_1,y_1)$ be a point on E, and $2P=(x_2,y_2)$, where, $\lambda_1=x_1+y_1/x_1$, $x_2= \lambda_1^2+ \lambda_1+a$, $y_2=x_1^2+ \lambda_1 x_2+x_2$.
Let $3P=P+2P=(x_3,y_3)$, where $\lambda_2=(y_2+y_1)/(x_2+x_1)$
$x_3= \lambda_2^2+ \lambda_2+x_1+x_2+a$ and $y_3= \lambda_2(x_1+x_3)+x_3+y_1$.
From the result of 2P and 3P, we can calculate other scalar multiplications on the elliptic curve.

1.3 Hardness of Elliptic Curve Cryptography: ECDLP

Let E be an elliptic curve over some finite field GF(p) and G = E(Fq) be a cyclic additive group, where the group operation is addition modulo p. The elliptic curve discrete logarithm problem is defined as follows:
Given P $\in$ G and an element Q $\in$ <P>, find the integer m, such that Q = [m]P [5].

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

The difference between ECDLP and the Discrete Logarithm Problem (DLP) is that, DLP though a hard problem is known to have a sub exponential time solution, and the solution of the DLP can be computed faster than that to the ECDLP. This property of Elliptic curves makes it favorable for its use in cryptography.

III. SCALAR MULTIPLICATION OF ECC

In all the protocols like ECDH, ECDSA, ECAES, the most time consuming part of the computations is scalar multiplications. That is the calculations of the form Q= k P = P + P + P… k times

Here P is a curve point; k is an integer in the range of order of P (i.e. n). P is a fixed point that generates a large, prime subgroup of E(Fq), or P is an arbitrary point in such a subgroup. Elliptic curves have some properties that allow optimization of scalar multiplications.

ECC Operations: Hierarchy

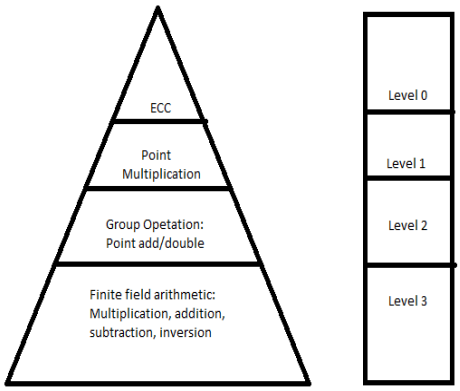

Fig 3.1: The Three Layers for Elliptic Curve Scalar Multiplication

Different algorithms for the scalar multiplications with analysis are given as follows:

1.4 Scalar Multiplication: Most Significant Bit First (MSB First)

1) Require k=(k_{m-1},k_{m-2},…..,k_0), k_{m-1} =1
2) Compute kP
3) Q=P
4) For i=m-2 to 0
5) Q=2P
6) If k_i=1 then
7) Q=Q+P
8) End if
9) End For
10) Return Q

Sequencial Algorithm:
Requires m point doubling and (m-1)/2 point additions on an average.

1.5 Scalar Multiplication: Least Significant Bit First (LSB First)

1) Require k=(k_{m-1},k_{m-2},…..,k_0), k_{m-1} =1
2) Compute kP
3) Q=0, R=P
4) For i=0 to m-1
5) If k_i=1 then
6) Q=Q+R
7) End if
8) R=2R
9) End for
10) Return Q

Can Parallelize
The accumulation and doubling can be stored in separate registers. On the average, m/2 point additions and m/2 point doublings.

1.6 Scalar Multiplication: Non Adjacent Form (NAF)

This is a much efficient method used in the computation of kP. Here, the integer k is represented as $k = \sum_{j=0}^{l-1} k_j 2^j$, where each kj ∈ {–1, 0, 1}. The weight of NAF representation of a number of length *l* is *l*/3. Given below is an algorithm for finding NAF of a number.

```
NAF(k)
Comment: Returns u[] which contains the NAF representation of k
Begin
        c ← k
        l ← 0
        While (c > 0)
        BeginWhile
                If (c is odd)
                BeginIf
                        u[l] ← 2 – (c mod 4)
                        c ←c – u[l]
                Else
                        u[l] ← 0
                EndIf

                c ← c/2
                l ← l + 1
        EndWhile
        Return u
End
```

Figure 3.2: Computation of the NAF of a scalar

The generation of NAF for k = 7 = $(111)_2$ is as shown below

No of iterations	c	*L*	u
1	7	0	-1
2	4	1	0
3	2	2	0
4	1	3	1

Therefore, the value of 7 in NAF form is (1 0 0 –1). (Note that no two consecutive digits are non-zero)

The algorithm [6] for scalar multiplication using NAF is given as

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Algorithm: Scalar Multiplication (NAF)
Input: k, P ϵ $E(F_p)$
Output: kP

1) Compute NAF(k)= $\sum_{j=0}^{l-1} k_j 2^j$
2) Q← Infinity
3) i from t-1 to 0, repeat
   a. Q←2Q
   b. If(k_i=1) then Q←Q+P
   c. If(k_i= -1) then Q←Q - P
4) Return Q

1.7 Scalar Multiplication: Montgomery Method

1) Input : k>0, P
2) Output : Q=kP
3) Set k= $(k_{m-1},k_{m-2},\ldots..,k_0)_2$
4) Set P_1=P, P_2=2P
5) For i from m-2 to 0
   a) If k_i=0
      Set $P_1=P_1+P_2$, $P_2=2P_2$
   b) Else
      Set $P_2=P_1+P_2$, $P_1=2P_1$
6) Return Q=P_1

Invariant Property: $P=P_2-P_1$

This brings us to the topic of fast multiplication of EC without pre-computations.

This is based on invariant property where we say that there are two points P2 and P1 and we want to compute kP. We generate P2 and P1 at each step or at each iteration such that the difference of P_2 and P_1 are always maintained to be P.
The extension of Montgomery's method [7] to elliptic curves over $GF(2^m)$ requires formulas for implementing Step 5 of the Algorithm. In what follows we give efficient formulas that use only the x-coordinates of P_1, P_2 and P for performing the arithmetic operations needed in above algorithm. At the end of the lth iteration of Algorithm, we obtain the x-coordinates of kP and (k+1)P. We also provide a simple formula for recovering the y-coordinate of kP.

Weierstrass Point Addition

$y^2+xy=x^3+ax^2+b$, $(x,y)\epsilon GF(2^m)*GF(2^m)$

1) Let P=(x_1, y_1) be a point on the curve.
2) -P=(x_1, x_1+y_1)
3) Let Q=(x_2,y_2) and R=P+Q=(x_3, y_3)

$$x_3=\begin{cases} m^2+m+a+x_2+x_1 & P\neq Q \text{ where } m=(y_2+y_1)/(x_2+x_1) \\ x_1^2+b/x_1^2 & P=Q \end{cases}$$

$$y3=\begin{cases} m(x_1+x_3)+x_3+y_1 & P\neq Q \text{ where } m=(y_2+y_1)/(x_2+x_1) \\ x_1^2+(x_1+y_1/x_1)x_3+x_3 & P=Q \end{cases}$$

1) Point addition and doubling each requires 1 inversion and 2 multiplications.
2) We neglect the cost of constant multiplication, squaring and addition
3) Montgomery noticed that the x-coordinate of 2P does not depend on the y-coordinate of P in point doubling i.e. 2P

Result1:

Let $P_1=(x_1,y_1)$ and $P_2=(x_2,y_2)$ be elliptic curve points. Then, the x-coordinate of P_1+P_2, x_3 can be computed as:
$x_3=(x_1y_2+x_2y_1+x_1x_2^2+x_2x_1^2)/(x_1+x_2)^2$

Derivation:

Given that the field has characteristics =2 and that P_1 and P_2 are points on the curve $y^2+xy=x^3+ax^2+b$. Now $x_3=[(y_2+y_1)/(x_2+x_1)]^2 + (y_2+y_1)/(x_2+x_1) +x_2+x_1+a$

$=[(y_2+y_1)^2+(y_2+y_1) (x_2+x_1)+ (x_2+x_1)^3+ a((x_2+x_1)^2]/ (x_2+x_1)^2$
$=[(x_1y_2+x_2y_1)+(y_1^2+x_1y_1+x_1^3+ax_1^2+b)+(y_2^2+x_2y_2+x_2^3+ax_2^2+b)+(x_1^2x_2+x_2^2x_1)]/ (x_2+x_1)^2$
$=[x_1y_2+x_2y_1+x_1x_2^2+x_2x_1^2]/(x_1+x_2)^2$
Further, for the Montgomery's method, $P=P_2-P_1$. Here
$P_1=(x_1,y_1)$ => - $P_1=(x_1, x_1+y_1)$
$P=P_2-P_1$

➔ $(x,y)= (x_2,y_2)+ (x_1, x_1+y_1)$
➔ $x= [x_1y_2+x_2(y_1+x_1)+x_1x_2^2+x_2x_1^2]/(x_1+x_2)^2$
➔ $x+x_3=[x_1y_2+x_2(y_1+x_1)+x_1x_2^2+x_2x_1^2]/(x_1+x_2)^2 + [x_1y_2+x_2y_1+x_1x_2^2+x_2x_1^2]/(x_1+x_2)^2$
➔ $x+x_3=x_1x_2/(x_1+x_2)^2$
➔ $x_3=x + x_1x_2/(x_1+x_2)^2$
➔ $x_3=x+x_1^2/(x_1+x_2)^2 + x_1/(x_1+x_2)$

Result2:

If P=(x,y), $P_1=(x_1,y_1)$ and $P_2=(x_2,y_2)$ be elliptic points. Let $P=P_2-P_1$is an invariant. Then x-coordinate of P_1+P_2, x_3 can be computed in terms of x-coordinates as

$$x_3=\begin{cases} x+[x_1/(x_1+x_2)]^2+ x_1/(x_1+x_2), & P_1\neq P_2 \\ x_1^2+b/x_1^2, & P_1=P_2 \end{cases}$$

Result3:

Let P=(x,y), $P_1=(x_1,y_1)$ and $P_2=(x_2,y_2)$ be elliptic points. Assume that $P=P_2-P_1$ and x is not zero. Then y-coordinate of P1 can be expressed in terms of P, and x-coordinates of P_1 and P_2 as follows:
$y_1=(x_1+x)[(x_1+x)(x_2+x)+x^2+y]/x + y$

Derivation:

$P=P_2-P_1$ => $P_2=P+P_1$

➔ $x_2=[x_1y+xy_1+x_1x^2+xx_1^2]/(x_1+x)^2$
➔ $xy_1=x_2(x_1+x)^2 + x_1y+ x_1x^2+xx_1^2$
➔ $xy_1=x_1(x_1x_2+x_1x+x^2+y) + x(xx_2)$
➔ $xy_1=x_1(x_1x_2+x_1x+x^2+y+xx_2+x^2)+ x(xx_2+x_2x_1+xx_1+y)+xy$
➔ $xy_1=(x_1+x)[(x_1+x)(x_2+x)+x^2+y] + xy$
➔ $y_1=(x_1+x)[(x_1+x)(x_2+x)+x^2+y]/x + y$

Fast multiplication on Elliptic Curve over $GF(2^m)$ without precomputation[3] is as:

Scalar Multiplication: Final Montgomery Algorithm for Affine Coordinate

Input: k>0, P=(x,y)
Output: Q=kP

1) If k=0 or x=0 then Output (0,0)
2) Set k=$(k_{l-1},k_{l-2},\ldots,k_0)2$
3) Set $x_1=x$, $x_2=x^2+b/x^2$

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

4) For i from l-2 to 0
   a) Set $t=x_1/(x_1+x_2)$
   b) If $k_i=1$ then $x_1=x+t^2+t$, $x_2= x_2^2 +b/ x_2^2$
   c)Else $x_1=x_1^2+b/ x_1^2$, $x_2=x+t^2+t$
5) $r_1=x_1+x$, $r_2=x_2+x$
6) $y_1=r_1(r_1r_2+x^2+y)/x + y$
7) return $Q=(x_1,y_1)$

IV. SUMMARRY & DISCUSSION

We have summarized the comparison among different algorithm used for scalar multiplication based on different properties. We choose both sequential as well as parallelizable algorithms for our comparative work as shown below in Table 4.1. All four chosen algorithms have different style of working but yield same result that is Scalar Multiplication.

TABLE 4.1Comparison of Algorithms: MSB First, LSB First and NAF

SNO	Properties	MSB	LSB	Addition /Subtraction Method	Montgomery's Method
1	Used for	Scalar Multiplication	Scalar Multiplier	Scalar Multiplier	Scalar Multiplication
2	Type of Algorithm	Sequential	Parallelizable	Sequential	Parallelizable
3	Digits used	0,1(Binary)	0,1(Binary)	0,1,-1(NAF)	0,1(Binary)
4	No. of point addition	(m-1)/2	(m-1)/2	(m-1)/3	m-1
5	No. of point doubling	M	m/2	m-1	m-1
6	No. of register used	1	2	1	----------
7	Hamming Weight	Higher	Higher	Lower	----------
8	Speed of scalar Multiplication	Lower	Lower (if not parallelized)	Higher	Higher
9	Pre-computation	YES	YES	YES	NO
10	Invariant Property	No	No	No	YES

Comments:

How to reduce inversions?

1) The affine coordinate inversions are very expensive.
2) For n>=128, each inversion requires around 7 multipliers (in hardware design).
3) Lopez Dehab Projective Coordinates
   a) $(x,y,z),z\neq0$, maps to $(x/z, y/z^2)$
   b) Motivation is to replace inversion by the multiplication operations and then perform one inversion at the end (to obtain back the affine coordinate.)

V. PROJECTIVE COORDINATES

In the situation where inversion in GF(2^m) is expensive relative to multiplication, it may be advantageous to represent points using projective coordinates of which several types have been proposed. In standard projective coordinates, the projective point (X:Y:Z), $Z\neq0$, corresponds to affine coordinate (X/Z, Y/Z). The projective equation of the elliptic curve is $Y^2Z+XYZ=X^3+aX^2Z+bZ^3$. In Jacobian [8] projective coordinates the projective points (X:Y:Z), $Z\neq0$, corresponds to affine coordinates $(X/Z^2, Y/Z^3)$ and the projective equation of the curve is $Y^2+XYZ=X^3+aX^2Z^2+bZ^6$. In [9], a new set of projective coordinates was introduced. Here, a projective point (X:Y:Z), $Z\neq0$, corresponds to the affine point $(X/Z, Y/Z^2)$, and the projective equation of the curve is $Y^2+XYZ=X^3Z+aX^2Z^2+bZ^4$.

a. Comparison of point addition and doubling in Lopez Dehab and Affine Coordinate

In Lopez Dehab Projective Coordinates point doubling and point addition is given as:

$P_1=P_2$, $X_3=X_1^4+bZ_1^4$ and $Z_3=Z_1^2.X_1^2$

$P_1\neq P_2$, $Z3= (X_1Z=+X_2Z_1)^2$ and $X_3=x.Z_3+(X_1Z_2)(X_2Z_1)$

In Affine Coordinates point doubling and point addition is given as:

$P_1\neq P2$, $x3=x+[x_1/(x_1+x_2)]^2+ x_1/(x_1+x_2)$,

$P_1=P_2$, $x_3=x_1^2+b/x_1^2$.

Now, we compare the field operations in both the coordinate system as:

	Affine Coordinates	Lopez Dehab Projective Coordinates
Inverses	2	0
Field Multiplication	1	4
Addition	4	3
Squaring	2	5

1.8 Scalar Multiplication: Montgomery Algorithm in Projective Version

1) Input k>0, P=(x,y)
2) Output: kP
3) Set $K\leftarrow(k_{l-1}, \dots , k_2,k_1,k_0)$
4) Set $X_1=x$, $Z_1=1$, $X_2=x^4+b$, $Z_2=x^2$
5) For i from l-2 to 0
   If $k_i=1$ then

   $Madd(X_1,Z_1,X_2,Z_2)$,
   $Mdouble(X_2,Z_2)$

   Else

   $Madd(X_2,Z_2,X_1,Z_1)$,
   $Mdouble(X_1,Z_1)$
6) Return $Q=(M_{XY}(X_1,Z_1,X_2,Z_2))$

Projective to Affine

$x_3=X_1/Z_1$ and $y_3=(x+X_1/Z_1)[(X_1+xZ_1)(X_2+xZ_2)+(x^2+y)(Z_1Z_2)](xZ_1Z_2)^{-1} + y$

1) The result requires 10 multiplications and 1 inversion.

Proof:

$y_1=(x_1+x)[(x_1+x)(x_2+x)+x^2+y]/x + y$

(Here $x=X/Z$ and $y=Y/Z^2$)

$= (x+X_1/Z_1)[(X_1/Z_1+x) +(X_2/Z_2+x)+x^2+y]/x + y$

$=(x+X_1/Z_1)[(X_1+xZ_1)(X_2+xZ_2)+(x^2+y)(Z_1Z_2)](xZ_1Z_2)^{-1} + y$

Final Comparison of Affine and Projective Coordinates

Affine Coordinates	Projective Coordinates
Inv: 2logk+1	Inv: 1

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Mult 2logk+4	Mult: 6logk+10
Add: 4logk+6	Add: 3logk+7
Sqr: 2logk+2	Sqr: 5logk+3

Inversion has been reduced but multiplication, squaring and addition have been increased in the projective coordinates. Hence, final decision depends upon the I:M ratio of the finite field operators. i.e. it depends upon the M(Number of multipliers) and I(Number of inverse finder) that shows whether affine coordinates is better or projective.

1.9 Parallelizing strategies for Scalar Multiplication

1) Point Doubling
   a) Cycle 1: $T=X_1^2$, $M=cZ_1^2$, $Z_2=T.Z_1^2$
   b) Cycle 1a: $X_2=T^2+M^2$
2) Point Addition
   a) Cycle 1: $t_1= (X_1.Z_2)$, $t_2= (X_2.Z_1)$
   c) Cycle 1a: $M=(t_1+t_2)$, $Z_1=M^2$
   d) Cycle 2: $N=t_1.t_2$, $M=xZ_1$
   e) Cycle 2a: X1=M+N

[We assume that squaring and multiplication with constants can be performed without multipliers.]

For doubling operation we need only one multiplier for $T.Z_1^2$. Now in the point addition there are two multiplications in t_1 and t_2 and two multiplications in N and M. Thus, if we want to parallelize then we require two multipliers one for doubling and the other for addition. Therefore, depending upon the resource constraints, we can parallelize the point addition. Thus, we see that in the design hierarchy, we can parallelize the point multiplication (kP) which is at the level 1 as well as group operations: point add/double which is at level 2.

1.10 Parallelizing Montgomery Strategies

1) Input k>0, P=(x,y)
2) Output: kP
3) Set K←$(k_{l-1}, \dots, k_2,k_1,k_0)$
4) Set $X_1=x$, $Z_1=1$, $X_2=x^4+b$, $Z_2=x^2$
5) For i from l-2 to 0
   If $k_i=1$ then
   5a)Madd(X_1,Z_1,X_2,Z_2),
   Mdouble(X_2,Z_2)
   Else
   5b)Madd(X_2,Z_2,X_1,Z_1),
   Mdouble(X_1,Z_1)
6) Return $Q=(M_{XY}(X_1,Z_1,X_2,Z_2))$

Parallelizing Strategies

1) Parallelize level 1: if we allocate one multiplier to each of Madd and Mdouble, then we can parallelize 5a) and 5b). Thus, four clock cycles are required for each iteration. Total time is nearly 4l.
2) Parallelize level 2: if we can parallelize the Madd and Mdouble then we cannot parallelize level 1, if we have only 2 multipliers. So, we have sequential steps for 5a and 5 b. Therefore, total time is 3l.
3) Parallelizing both the levels: Total time is 2l clock cycles. Requiring 3 multipliers one for the point doubling and two for the point addition.
4) Thus, Montgomery Algorithm is highly parallelizable.
5) Helpful in high performance designs (low power, high throughput etc.).

VI. CONCLUSION

In this paper, we have presented an efficient method for computing elliptic scalar multiplications. The method performs exactly 6Floor[log2k]+10 field multiplication for computing kP on elliptic curves selected at random, is easy to implement in both hard wareand software, requires no pre-computations, works for any implementation of GF(2n), is faster than the addition-subtraction method on average, and uses fewer registers than methods based on projective schemes. Therefore, the method appears useful for applications of elliptic curves in constraint environments such as mobile devices and smart cards.

REFERENCES

1. N. Koblitz, “Elliptic Curve Cryptosystems", Mathematics of Computation, 48, pp. 203-209, 1987
2. V. Miller, “Uses of elliptic curves in cryptography", Advances in Cryptology: proceedings of Crypto'85, Lecture Notes in Computer Science, vol. 218. New York: Springer-Verlag, 1986, pp. 417-426.
3. Julio Lopez, Ricardo Dahab,” Fast Multiplication on Elliptic Curves over GF(2^m) without Precomputation”, Cryptographic Hardware and Embedded Systems Lecture Notes in Computer Science Volume 1717, 1999, pp 316-327.
4. A. Menezes, Elliptic Curve Public Key Cryptosystems,Kluwer Academic Publishers, 1993.
5. Blake, I., Seroussi, G., and Smart, N. Elliptic Curves in Cryptography, Cambridge University Press, 1999.
6. Xiaopeng Zhang*, Shuguo Li “A High Performance ASIC Based Elliptic CurveCryptographic Processor over GF(p) ” Design and Test Workshop, 2007, IDT 2007. 2nd International, pp. 182 – 186.
7. P. Montgomery, “Speeding the Pollard and elliptic curve methods of factorization",Mathematics of Computation, vol 48, pp. 243-264, 1987.
8. D. Chudnovsky and G. Chudnovsky, \Sequences of numbers generated by additionin formal groups and new primality and factoring tests", Advances in AppliedMathematics, 7 (1987), 385-434.
9. J. Lopez and R. Dahab, “Improved algorithms for elliptic curve arithmetic inGF(2n)", Selected Areas in Cryptography - SAC '98, LNCS 1556, 1999, 201-212.

AUTHOR PROFILE

Mrityunjay Kumar Choudhary received his BCA and MCA degrees from IGNOU in 2007 and 2009, respectively. After MCA he joined GGSIPU for M.Tech in Information Technology. During M.Tech, he as a Research Scholar works on Cryptography with specializing in Elliptic Curve Cryptography under guidance of Prof. C.S Rai. His research areas are Cryptography, ECC, and Natural Language Processing etc.

Praful Kumar Singh received his B.Tech degree in I.T from GGSIPU in 2010. After B.Tech he joined GGSIPU for M.Tech in Information Technology. During M.Tech, he as a Research Scholar works on Image Encryption using Chaos theory under guidance Assistant Professor of USICT Dr. RavinderPurwar. He did his Dissertation on Quality Issues in Data Warehouse under the guidance of Assistant Professor of USICT Mrs. Jaspreeti Singh. His Research Area includes Cryptography, Data Warehouse, Image Encryption, Network Security, Software Engineering, Data Mining etc.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)