Algorithm based on one's complement for fast scalar multiplication in ECC for Wireless Sensor Network

Pritam Gajkumar Shah, Xu Huang, Dharmendra Sharma,
Faculty of Information Sciences and Engineering,
University of Canberra, Act 2601-Australia

Abstract: Elliptic curve cryptography (ECC) is having good potential for wireless sensor network security due to its smaller key size and its high strength of security. But there is a room to reduce key calculation time to meet the potential applications in particular for wireless sensor networks. Scalar multiplication is the operation in elliptical curve cryptography which takes 80 % of key calculation time on wireless sensor network motes. This research proposes algorithm based on 1's complement subtraction to represent scalar in scalar multiplication which offer less Hamming weight and will remarkably improve the computational efficiency of scalar multiplication.

Keywords: *Elliptic curve cryptography, Scalar multiplication, Non-adjacent form, Hamming weight, one's complement subtraction, ROM, wireless sensor networks.*

I INTRODUCTION

Rapid growth in the Very Large Scale Integration [VLSI] technology, embedded systems and Micro Electric Mechanical Systems [MEMS] has enabled production of less expensive sensor nodes which can communicate information shorter distances with efficient use of power [1]. Sensor node detects information, processes it with the help of an in-built microcontroller and communicates results to the 'sink or base station'. The base station is a more powerful node linked with central station via satellite or internet communication. Wireless sensor networks can be deployed in various applications namely environmental monitoring e.g. volcano detection [2,3], distributed control system [4] , detection of radioactive sources [5], agricultural and farm management [6], and computing platform for tomorrows' internet[7] .

II CHALLENGES IN DEVELOPING SECURED PROTOCOLS FOR WIRELESS SENSOR NETWORKS [WSN]

Compared to traditional networks, a wireless sensor network has many resource constraints [4]. The MICA2 [Third Generation Mote Module] consists of an 8 bit ATMega 128L microcontroller working on 7.3 MHz .As a result MICA2 motes have limited computational power. Normally radio transceiver of MICA motes can achieve maximum data rate of 250 Kbits/sec which puts a limitation on the communication resources. The flash memory which is available on the MICA mote is only 512 Kbyte. Apart from these the battery which is available on the board is of 3.3.V with 2A-Hr capacity. Due to the above boundaries the current state of art protocols and algorithms are expensive for sensor networks due to their high communication overheads.

III ELLIPTIC CURVE CRYPTOGRAPHY

PRELIMINARIES

Elliptic Curve Cryptography was introduced by Victor Miller [9] and Neal Koblitz [10] independently in the early eighties. The advantage of ECC over other public key cryptography techniques such as RSA, Diffie-Hellman is that the best known algorithm for solving ECDLP the underlying hard mathematical problem in ECC takes the fully exponential time. On the other hand the best algorithm for solving RSA and Diffie-Hellman takes sub exponential time [11]. To sum up the problem of ECC can be solved only in exponential time and so far there is a lack of sub exponential attack on ECC.

An elliptic curve E over $GF(p)$ can be defined by $y^2 = x^3 + ax + b$ where $a, b \in GF(p)$ and $4a^3 + 27b^2 \neq 0$ in the $GF(p)$.

The point (x, y) on the curve satisfies above equation and the point at infinity denoted by ∞ is said to be on the curve.

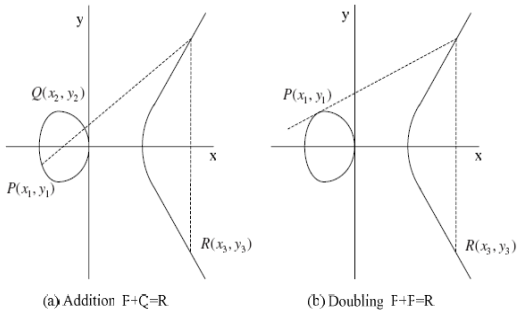

Fig. 1. Point addition and point doubling on elliptic curve

If there are two points on curve namely $P(x_1, y_1)$, $Q(x_2, y_2)$ and their sum given by point $R(x_3, y_3)$ the algebraic formulas for point addition and point doubling are given by following equations,

978-0-7695-4019-1/10 $26.00 © 2010 IEEE
DOI 10.1109/WAINA.2010.48

$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = (x_1 - x_3)\lambda - y_1$$
$$\lambda = (y_2 - y_1) \div (x_2 - x_1), P \neq Q$$

and

$$x_3 = \lambda^2 - 2x_1,$$
$$y_3 = \lambda(x_1 - x_3) - y_1$$
$$\lambda = (3x_1^2 + a) \div 2y_1, P = Q$$

Where the addition, subtraction, multiplication and the inverse are the arithmetic operations over $GF(p)$ which are as shown in Figure 1.

IV ELLIPTIC CURVE DIFFIE-HELLMAN PROTOCOL (ECDH) FOR WSN

As per [13] the original Diffie-Hellman algorithm with RSA requires a key of 1024 bits to achieve sufficient security but *Diffie Hellman based on ECC* can achieve the same security level with only 160 bit key size.

The classical Elliptic Curve Diffie Hellman scheme works as shown in the Fig. 2

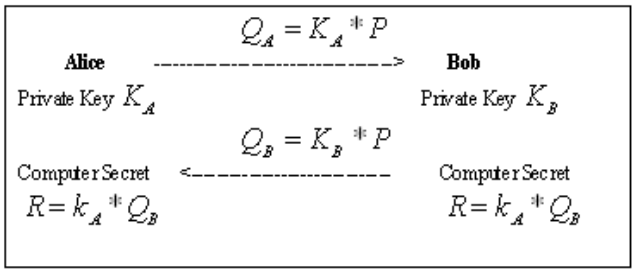

Figure 2 Diffie-Hellman protocol based on ECC

Initially Alice and Bob agree on a particular curve E with base point *P*. They generate their public keys by multiplying P with their private keys namely K_A and K_B . After sharing public keys, they generate a shared secret key by multiplying public keys by their private keys. The secret key is $R = K_A * Q_B = K_B * Q_A$. With the known values of Q_A , Q_B and P it is computationally intractable for an eavesdropper to calculate K_A and K_B which are the private keys of Alice and Bob. As a result, adversaries cannot figure out R which is the shared secret key.

V CHALLENGES INVOLVED IN THE SCALAR MULTIPLICATION

In ECC two heavily used operations are involved namely, scalar multiplication and modular reduction. The Gura et al [14] showed that 85% of execution time is spent on scalar multiplication operation. Scalar Multiplication is the operation of multiplying point P on an elliptic curve E defined over a field $GF(p)$ with positive integer K which involves point addition and point doubling.

$$kP = (P + P + P..... + P) \rightarrow k\ Times$$

Operational efficiency of *K*P is considerably affected by the algorithm used for recoding of integer K in scalar multiplication.

This research paper proposes innovative algorithm based on one's complement subtraction for representation of integer *K* which accelerates the computation of scalar multiplication in wireless sensor networks.

VI RECODING OF INTEGER *K* IN SCALAR MULTIPLICATION

The number of point doubling and point additions in scalar multiplication depends on the recoding of integer k. Expressing integer k in binary format highlight this dependency.

The number of zeros and number of ones in the binary form, their places and the total number of bit affects computational cost of scalar multiplications. The Hamming weight i.e. the number of non zero elements, determines the number of point additions and bit length of integer K determines the number of point doublings operations in scalar multiplication.

One point addition when $P \neq Q$ requires one field inversion and three field multiplications. Squaring is counted as regular multiplications. This cost is denoted by $1I + 3M$ where the I denotes the cost of inversion and M denotes the cost of multiplication.

One point doubling when $P = Q$ requires $1I + 4M$ as we can neglect the cost of field additions as well as the cost of multiplications by small constant 2 and 3 in the above formulae.

VII THE EXISTING METHODS OF SCALAR MULTIPLICATION:

1. BINARY METHOD

Scalar multiplication is the computation of the form $Q = kP$ where P, Q are the elliptic curve points and k is positive integer. This is achieved by repeated elliptic curve point addition and doubling operations. In binary method the integer k is represented in binary form,

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

The Binary method 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: Left to right binary method for point multiplication

Input: Point $P \in E$, an ℓ bit integer

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

Output: $Q = kP$

1. $Q \leftarrow \infty$
2. For $j = \ell - 1$ to 0 do
   2.1 $Q \leftarrow 2Q$,
   2.2 *if* $k_j = 1$ then $Q \leftarrow Q + P$
3. Return Q

The cost of multiplication in binary method depends on the number of non zero elements and length of the binary representation of k .If the representation has $k_{l-1} \neq 0$ then Binary method require $\ell - 1$ point doublings and $W - 1$ where ℓ is the length of the binary expansion of k and W is the Hamming weight of the k that is the number of non zero elements in expansion of k .

For example if $k = 629 = (1001110101)_2$, it will require $W - 1 = 6 - 1 = 5$ point additions and $\ell - 1 = 10 - 1 = 9$ point doublings operations.

2. SIGNED DIGIT REPRESENTATION METHOD

The subtraction has virtually same cost as addition in the elliptic curve group. The negative of point (x, y) is (x,-y) in odd characteristics. This leads to scalar multiplication methods based on addition –subtraction chains, which help to reduce the number of curve operations. When integer k is represented with the following form, it is called *as binary signed digit representations*.

$k = \sum_{j=0}^{\ell} S_j 2^j, S_j \in \{1,0,-1\}$

When signed digit representation has no adjacent non zero digits, i.e. $S_j S_{j+1} = 0$ for all $j \geq 0$, it is called non-adjacent from (NAF).

The following Algorithm 2 computes the NAF of a positive integer given in binary representation.

Algorithm 2: Conversion from Binary to NAF

Input: An integer $k = \sum_{j=0}^{\ell-1} K_J 2^J, K_j \in \{0,1\}$

Output: NAF $k = \sum_{j=0}^{\ell} S_j 2^j, S_j \in \{1,0,-1\}$

$1. C_0 \leftarrow 0$

$2.$ *For* $j = 0$ to ℓ do

$3. C_{j+1} \leftarrow \lfloor (K_j + K_{j+1} + C_j)/2 \rfloor$

$4. S_j \leftarrow K_j + C_J - 2C_{j+1}$

$5.$ Re *turn* $(S_\ell \ldots\ldots\ldots S_0)$

NAF has usually fewer non zero digits than binary representations. The average hamming weight for NAF form is $(n-1)/3$ .So generally it requires $(n-1)$ point doublings and $(n-1)/3$ point additions. The Binary method can be revised accordingly and is given in Algorithm 3 for NAF form. This modified method is called as *Addition Subtraction* method.

VIII PROPOSED ALGORITHM BASED ON ONE'S COMPLEMENT FOR RECODING OF SCALAR K

A subtraction by utilization of the 1's complement is most common in binary arithmetic. The 1's complement of any binary number may be found by the following *equation I*.

$C_1 = (2^a - 1) - N$ ……(I)

Where,

$N = Binary\ Number$

$C_1 = ones$ complement *of binary number*.

a = *Number of bits in N in its binary form.*

A close observation of the *equation I* reveal fact that any positive integer can be represented by using minimal non zero bits in its 1's complement form provided that it is having minimum of 50% Hamming Weight. The minimal non zero bits in positive integer scalar are very important to reduce number of intermediate operations of multiplication, squaring and inverse in elliptical curve cryptography as we have seen in the previous sections.

The *equation I* can be modified as per below-

$N = (2^a - C_1 - 1)$ …….. (II)

For example let us take N =1788

N= $(11011111100)_2$ in its binary form

C_1 = 1's Complement of the number of N= $(00100000011)_2$

$a = number$ of bits in N when it is in Binary form = 11

After putting all the above values in the *equation II* we will get,

$1788 = 2^{11} - 001\,000\,000\,11 - 1$, this can be reduced to,

$1788 = 1\,000\,000\,000\,00 - 001\,000\,000\,11 - 1$ …..*equation III*

1788= 2048 – 256– 2 – 1 –1

As evident from *equation III* the Hamming Weight of scalar N has reduced from 8 to 5 which will save 3 elliptic curve addition operations. One addition operation requires 2 Squaring, 2 Multiplication and 1 inverse operation. In this case total 6 Squaring, 6 Multiplication and 3 Inverse operations will be saved.

The above recoding method based on one's complement subtraction combined with sliding window method gives very good optimization results.

Algorithm for sliding window scalar multiplication on elliptic curves.

1. $Q \leftarrow P_{\infty}$ *and* i $\leftarrow l$ - 1
2. while i $\geq$ 0 do
3. if $n_i = 0$ then Q $\leftarrow$ [2]Q and i $\leftarrow$ i - 1
4. else
5. s $\leftarrow \max(i - k + 1, 0)$
6. *while* $n_s = 0$ do s $\leftarrow$ s + 1
7. for h = 1 to i - s + 1 do Q $\leftarrow$ [2]Q
8. u $\leftarrow (n_i \ldots\ldots n_s)_2$ $[n_i = n_s = 1$ and i - s + 1 $\leq$ k]
9. Q $\leftarrow$ Q $\oplus$ [u]P [u is odd so that [u]P is precompute d]
10. i $\leftarrow$ s - 1
11. return Q

Let us compute [763] P with sliding window algorithm with K recoded in Binary form with different window sizes ranging from 2 to 10.It is observed that as the window size increases the number of pre computations also increases geometrically. At the same time number of additions and doubling operations decreases.

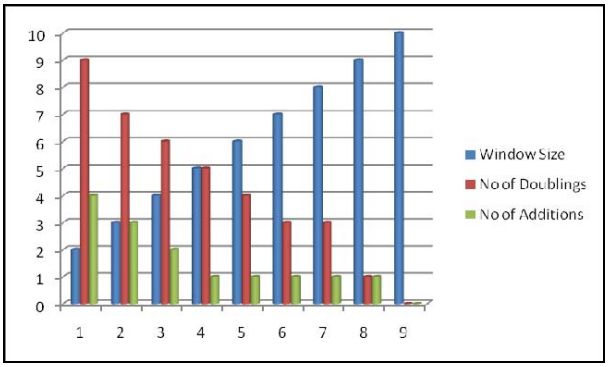

Figure 3 Trade off between window size and computational cost

TABLE 1: WINDOW SIZE VS NO OF DOUBLINGS, ADDITIONS AND PRE COMPUTATIONS

Window Size	No of Doublings	No of Additions	No of Pre computation
2	9	4	1
3	7	3	3
4	6	2	7
5	5	1	15
6	4	1	31
7	3	1	61
8	3	1	127
9	1	1	251
10	0	0	501

Window Size w = 2

$763 = (1011111011)_2$

No of Pr *ecomputati ons* $= 2^w - 1 = 2^2 - 1 = [3]P$

763 = 10 11 11 10 11

The intermedia te values of Q are

P,2P,4P,8P ,11P,22P,4 4P,47P,94P ,

95P,190P,3 80P,760P,7 63P

Computatio nal Cost = 9 doublings, 4additions ,

1 precomputaion.

Window Size w = 3

No of Precomputa tions $= 2^k - 1 = 2^3 - 1 = [7]P$

Soall odd values [3]P, [5]P, [7]P

763 = 101 111 101 1

= [5]P [7]P [5]P [1]P

The intermedia te values of Q are

5P,10P,20P ,40P,47P,9 4P,188P,

376P,381P, 762P,763P

Computatio nal Cost = 7doublings ,3 additions,

3 precomputaions.

Window Size = 4

763 = 1011 111 0 11

= 11P 7P 3P

No of precomputa tions = 3P,5P,7P,9 P,11P,13P, 15P

The intermedia te Values of Q are

11P,22P,44 P,88P,95P, 190P,380P, 760P,763P

Computatio nal Cost = 6doublings ,2 additions,

7 precomputa tions.

Window Size = 5

763 = 10111 11011

Precomputa tions = 3P,5P,7P,9 P,11P,13P, 15P,

17P,19P,21 P,23P,25P, 27P,29P,31 P

Theinterme diate values of Q are

23P,46P,92 P,184P,368 P,736P,763 P

Computatio nal Cost = 5 doublings, 1 addition,

15 precomputa tions

window Size = 6

763 = 101111 1011

Precomputa tions = $3P,5P,7P,9P,11P,13P,15P,17P,19P,$

$21P,23P,25P,27P,29P,31P,33P,35P,37P,39P,41P,43P,$

$45P,47P,49P,51P,53P,55P,57P,59P,61P$

The intermedia te values of Q are

47P,94P,18 8P,376P,75 2P,763P

Computatio nal Cost = 4 doublings, 1 addition, 31 precomputa tions.

window size = 7

763 = 1011111 011

= 95P 3P

Precomputations = 3P,5P,7P,9P,11P,13P,15P,17P,19P,21P, 23P,25P,27P,29P,31P,33P,35P,37P,39P,41P,43P,45P,47P, 49P,51P,53P,55P,57P,59P,61P,63P,65P,67P,69P,69P,71P, 73P,75P,77P,79P,81P,83P,85P,87P,89P,91P,93P,95P,97P, 99P,101P,103P,105P,107P,109P,111P,113P,115P,117P,119P, 121P

The intermediate values of Q are

95P,190P,380P,760P,763P

Computational Cost = 3 Doublings, 1 addition,

61 precomputations

Window Size = 8

763 = 1011111 0 11

= 95P 3P

Precomputations = 3P,5P,7P,9P,11P,13P,15P,17P,19P,21P, 23P,25P,27P,29P,31P,33P,35P,37P,39P,41P,43P,45P,47P, 49P,51P,53P,55P,57P,59P,61P,63P,65P,67P,69P,69P,71P, 73P,75P,77P,79P,81P,83P,85P,87P,89P,91P,93P,95P,97P, 99P,101P,103P,105P,107P,109P,111P,113P,115P,117P,119P, 121P,123P,125P,127P,129P,131P,133P,135P,137P,139P,141P, 143P,145P,147P,149P,151P,153P,155P,157P,159P,161P,163P, 165P,167P,169P,171P,173P,175P,177P,179P,181P,183P,185P, 187P,189P,191P,193P,195P,197P,199P,201P,203P,205P,207P, 209P,211P,213P,215P,217P,219P,221P,223P,225P,227P,229P, 231P,233P,25P,237P,241P,243P,245P,247P,249P,251P,253P, 255P.

The intermediate values of Q are

95P,190P,380P,760P,763P

Computational Cost = 3 Doublings, 1 addition,

127 precomputations

Window size = 9

763 = 101111101 1

= 381P 1P

The intermediate values of Q are

381P,762P,763P

Cost of Computation is 1 doubling, 1 addition,

aprroximtely 251 precomputations.

Window Size = 10

763 = 1011111011

= 763P

Cost of Computation is 0 doublings, 0 additions and

aprroximately 510 precomputations.

The trade-off between the computational cost and the window size is shown in the figure 3 and 4 and in Table no1.

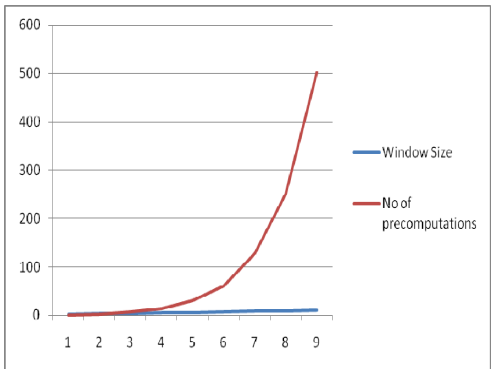

Figure4: Trade off between window size and number of pre computations

Now let us apply the proposed algorithm to the same number 763 to show the effectiveness of algorithm with window size of 3.

$763 = (1011111011)_2$

Let us recode 763 with equation II in one's complemnt substraction form and we will get,

$763 = 10000000000 - 0100000100 - 1$

$= 10\bar{1}0000\bar{1}0\bar{1}$

With window size of 3,

$763 = 10\bar{1}\ \ 000000\ \ \bar{1}0\bar{1}$

The intermediate values of Q are,

3P,6P,12P,24P,48P,96P,192P,384P,768P,763P

Computational Cost = 8 doublings, 1 addition and 3 pre computations

With the equation II the computational cost has been reduced from 3 additions as in binary method to only 1 addition in one's complement subtraction form. The number of pre computations remained same. This can be proved for different window sizes.

IX CONCLUSION

The positive integer in point multiplication may be recoded with *one' complement subtraction* to reduce the computational cost involved in this heavy mathematical operation for wireless sensor network platforms. The window size may be a subject of trade off between the available RAM and ROM at that particular instance on sensor node. As NAF method involves modular inversion operation to get the NAF of binary number, the one's complement subtraction can provide a very simple way of recoding integer.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks,* vol. 38, pp. 393-422, 2002.

[2] C. Chung-Kuo, J. M. Overhage, and J. Huang, "An application of sensor networks for syndromic surveillance," 2005, pp. 191-196.

[3] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing,* vol. 10, pp. 18-25, 2006.

[4] B. Sinopoli, C. Sharp, L. Schenato, S. Schaffert, and S. S. Sastry, "Distributed control applications within sensor networks," *Proceedings of the IEEE,* vol. 91, pp. 1235-1246, 2003.

[5] D. L. Stephens, Jr. and A. J. Peurrung, "Detection of moving radioactive sources using sensor networks," *Nuclear Science, IEEE Transactions on,* vol. 51, pp. 2273-2278, 2004.

[6] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, "Wireless ad hoc sensor and actuator networks on the farm," 2006, pp. 492-499.

[7] Z. Feng, "Wireless sensor networks: a new computing platform for tomorrow's Internet," 2004, pp. I-27 Vol.1.

[8] I. F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks" in *IEEE Communication Magazine*. vol. 40, August 2002, pp. 102-116.

[9] V. S. Miller, "Use of Elliptic Curves in Cryptography," in *Advances in Cryptology - CRYPTO '85: Proceedings*. vol. 218: Springer-Verlag, 1986, pp. 417-426.

[10] N.Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation,* vol. 48, pp. 203-209, 1987.

[11] J. Lopez and R. Dahab., " An overview of elliptic curve cryptography," Technical report ,Institute of Computing, Sate University of Campinas, Sao Paulo, Brazil, May 2000.

[12] K. Lauter, "The advantages of elliptic curve cryptography for wireless security," *Wireless Communications, IEEE [see also IEEE Personal Communications],* vol. 11, pp. 62-67, 2004.

[13] H. Wang, B. Sheng, and Q. Li, "Elliptic curve cryptography-based access control in sensor networks," *Int. J. Security and Networks,,* vol. 1, pp. 127-137, 2006.

[14] N. Gura, A. Patel, and A. Wander, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in *Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES)* August 2004.

[15] http://csrc.nist.gov/CryptoToolkit/dss/ecdsa/NISTReCur.pdf."

[16] D.J Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography," in *2nd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, pp. 71-80.

[17] I. Blake, G. Seroussi, and N. Smart, *Elliptic Curves in Cryptography* vol. 265, 1999.

[18] D. Hankerson, J. L. Hernandez, and A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields, CHES," 2000.