International Journal of Grid Distribution Computing
Vol.7, no.3 (2014), pp.287-298
http://dx.doi.org/10.14257/ijgdc.2014.7.3.27

Parallelization of Point Operations on Conic Curves over Finite Field GF(2^n)

Yongnan Li[1,2] and Limin Xiao[1,2]

[1]*State Key Laboratory of Software Development Environment, Beihang University, Beijing, 100191, China*
[2]*School of Computer Science and Engineering, Beihang University, Beijing, 100191, China*
liyongnan.buaa@gmail.com

Abstract

It becomes more and more important to design high-speed parallel cryptographic algorithms due to a growing need for information security. Conic curves cryptography is a new developing direction in the field of information security in recent years and there are less works focused on the parallel encryption algorithms for conic curves crypto system. This paper proposes four parallel algorithms for conic curves cryptosystem over finite field GF(2^n). One parallel algorithm of modular-multiplication is designed by analyzing its data dependency and making some modifications of several steps. In order to figure out the average runtime, we consider the probability distributions of different cases to compute the mathematical expectation. The operations of point-addition, point-double and point-multiplication, three fundamental point operations in conic curves cryptosystem over finite field GF(2^n), are paralleled based on this parallel algorithm of modular-multiplication and two parallel algorithms we proposed before. Time complexities and speedup ratios of the parallel algorithms and the sequential algorithms are calculated to make the quantitative comparison. The performance evaluation shows better efficiencies of the proposed parallel algorithms compared to the traditional algorithms.

Keywords: *Parallel algorithm, Conic curves, Finite field GF(2^n),Point-addition, Point-double, Point-multiplication*

1. Introduction

The conic curves cryptosystem [1-3], a new public-key cryptography first proposed by professor Cao [4, 5] based on the discrete logarithm system of conic curves, has developed very quickly over the last few years and it will be widely used in more fields in the future.

A growing need for information security has motivated the field of cryptography both in theoretical research and application development to be more and more important in recent years. However, the process of encryption is time consuming due to the increased secret key length which is used to satisfy the security demands. Therefore, it is important to design high-speed parallel algorithms for both encryption and decryption. We have done some works regarding the parallel algorithms for conic curves cryptosystem over finite field Fp and ring Zn [6-11]. Finite field GF(2^n) is one of the common used mathematical sets for constructing cryptosystems including elliptic curves cryptography[12, 13] and conic curves cryptography. There have been plenty of researches on parallel cryptographic algorithms over finite field GF(2^n), and they were mainly focused on parallel implementation by decreasing computing

ISSN: 2005-4262 IJGDC

units [14, 15], reducing power consumption [16, 17]or choosing best polynomial modular [18]. However, there is less deep study on fast parallel algorithms concerning multiple-precision integers for conic curves cryptosystem over finite field GF(2^n).

In this paper, several parallel algorithms are proposed to accelerate the speed of encryption and decryption for conic curves cryptosystem over finite field GF(2^n). One arithmetic operation over finite field GF(2^n), modular-multiplication operation, is paralleled firstly by analyzing the data dependency. Then we discuss the methods for paralleling point-addition and point-double in conic curves cryptosystem over finite field GF(2^n) based on our previous works about parallelization of two other arithmetic operations over finite field GF(2^n). Our previous work concerning parallel point-multiplication for other mathematical sets is used to parallel point-multiplication over finite field GF(2^n). The performance evaluation demonstrates that our techniques achieve high efficiencies for conic curves cryptosystem over finite field GF(2^n).

The rest of this paper is organized as follows. Next section introduces the definitions of the point operations on conic curves over finite field GF(2^n). Section 3 presents time complexities of two basic arithmetic operations over finite field GF(2^n). Section 4 proposes our parallel algorithms for conic curves cryptosystem over finite field GF(2^n). The last section concludes the whole paper and points out some future works briefly.

2. Definitions of Point Operations

This section introduces the definitions of point-addition, point-double and point-multiplication in conic curves cryptosystem over finite field GF(2^n) [1]. Conic curves $C_{2^n}(a,b)$ are the solution of the following equation:

$$C_{2^n}(a,b): y^2+xy \equiv ax^2+bx \ mod\ f(x), a,b \in GF(2^n). \tag{1}$$

where module $f(x)$ is a two-element polynomial.

Obviously,$O_{2^n}(0,0) \in C_{2^n}(a,b)$. Let$C_{2^n}(a,b)$ be

$$\begin{aligned} C_{2^n}(a,b) =& \{P(t)=(x,y)=(b(t^2+t+a)^{-1}, bt(t^2+t+a)^{-1}), \\ & t \in GF(2^n), t^2+t \neq a\} \cup \{P(\infty)=(0,0)\} \end{aligned} \tag{2}$$

And defining the operation $\hat{+}$:

(1) $\forall P(t) \in C_{2^n}(a,b)$, such that

$$P(t) \oplus P(\infty) = P(\infty) \oplus P(t) = P(t) \tag{3}$$

(2) $\forall P(t_1), P(t_2) \in C_{2^n}(a,b)$, if $t_1, t_2 \neq \infty$, such that

$$P(t_1) \oplus P(t_2) = P(t_3). \tag{4}$$

where $\begin{cases} t_3 = (t_1 t_2 + a)(t_1+t_2+1)^{-1} mod\ f(x), & t_1+t_2+1 \neq 0 \\ \infty, & t_1+t_2+1=0 \end{cases}$.

(3) $\forall P(t) \in C_{2^n}(a,b)$, negative element

$$-P(t) = P(t+1), -P(\infty) = P(\infty). \tag{5}$$

The operation of point-double is the same with the operation of point-addition except that the parameters t_1 and t_2 are equal in conic curves cryptosystem over finite field GF(2^n).

Point-multiplication signifies summation of many points on conic curves. Parameter *k* and parameter *P* represent coefficient and point on conic curves respectively. Point-multiplication *kP* is defined as:$kP = \overbrace{P \oplus P \oplus \cdots \oplus P}^{count=k}$.

3. Time Complexities of Two Basic Operations

As depicted in [19], our previous works about time complexities of two basic operations over finite field GF(2^n) are listed in Table 1.

Table 1. Time Complexities of Two Operations

Operation	Parallel time complexity	Sequential time complexity
Reduction	$\lceil log_2((n-1)/2) \rceil + 2$	$5(n-1)/2$
Inversion-multiplication	$4 + (n/2)(5 - (1/2)^n)$	$8 + (n/2)(13 - 6(1/2)^n)$

4. Parallel Algorithms

This section discusses four parallel algorithms for conic curves cryptosystem over finite field GF(2^n).We take one time clock as the computation time unit to make the quantitative evaluation of the efficiencies of parallel algorithms and sequential algorithms. To make the quantitative comparison between the parallel algorithms and the sequential algorithms, we assign two parameters (*n*and *t*) in the equations of runtime into different values.

4.1. Parallel Modular-Multiplication

The following algorithm is used for computing the modular-multiplication operation of multiple-precision integer over finite field GF(2^n).

Modular-multiplication over finite field GF(2^n)

Input:module $f(Z)$, $a(Z) = a_{n-1}Z^{n-1} + \cdots + a_1 Z + a_0$,
$b(Z) = b_{n-1}Z^{n-1} + \cdots + b_1 Z + b_0$.
Output:$c(Z) = a(Z) \bullet b(Z) mod\ f(Z)$.
1.if $a_0 = 1$,then $c \leftarrow b$; else, then$c \leftarrow 0$.
2. for *i* from 1 to$n-1$, repeat:
2.1 $b \leftarrow b \bullet Z mod\ f(Z)$.
2.2 if $a_i = 1$, then $c \leftarrow c + b$.
3. return $c(Z)$.

In the first step, only two time clocks are needed to execute an operation of comparison and an operation of assignment, and they have to be computed sequentially. Therefore, the parallel runtime and the sequential runtime of the first step are both 2.

Table 2. Probability of Every Case

Probability	a_i	b_{n-i}	Sequentialtime clocks	Parallel time clocks
1/4	1	1	4	3
1/4	1	0	3	2
1/4	0	1	3	2
1/4	0	0	2	1

If the highest bit of the coefficient of polynomial $b \bullet Z$ is 1, we could compute one dislocation XOR operation to figure out the coefficients of $b \bullet Z \ mod \ f(Z)$ instead of the reduction over finite field GF(2^n) in the Substep 2.1. We can execute the XOR operation and the comparison operation in one time clock if $b_{n-i}$is 1. In Substep 2.2, the operation of comparison must be calculated and the probability of computing the XOR operation is 0.5. The two operations in Substep 2.2 can be executed simultaneously. For the parallel procedure, the two operations of comparison can be finished in one time clock. As Table 2 describes, there are four cases in the second step. The average sequential runtime of the second step is $3(n-1)$. And the average parallel runtime of the second step is $2(n-1)$.Consequently, the total parallel runtime and sequential runtime of modular-multiplication over finite field GF(2^n) are

$$T_p = 2(n-1) + 2 = 2n \quad (6)$$

$$T_s = 3(n-1) + 2 = 3n - 1 \quad (7)$$

Table 3. Performance Evaluation of Modular-Multiplication

n	Tp_mm	Ts_mm
64	128	191
96	192	287
128	256	383
160	320	479
192	384	575
224	448	671
256	512	767

Therefore, the speedup is

$$S = \frac{3n-1}{2n}. \quad (8)$$

Table 3 and Fig.1 shows the performance improvement of our parallel method for the modular-multiplication operation over finite field GF(2^n).

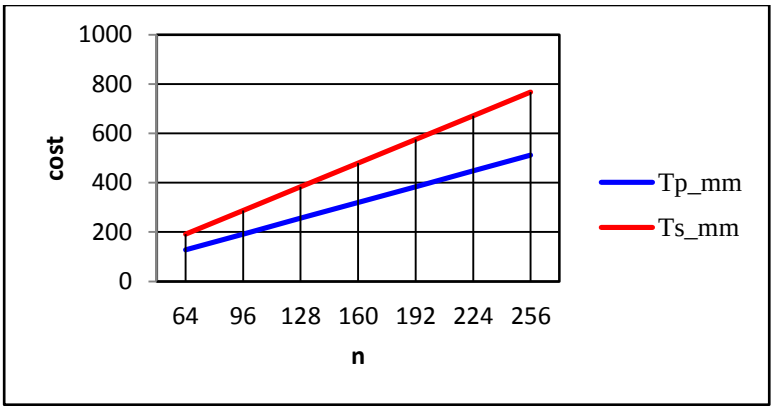

Figure 1. Performance Comparison of Modular-Multiplication

4.2. Parallel Point-Addition

In the definition of point-addition in conic curves cryptosystem over finite field GF(2^n), the operation of XOR can be incorporated into the operation of modular-multiplication in the numerator. The parallel runtime for computing the numerator is the same with modular-multiplication and the two XOR operations in the denominator could be finished while computing numerator. So the parallel runtime of computing numerator and denominator of point-addition is$2n$.

The sequential runtime of computing numerator and denominator of point-addition is the sum of one modular-multiplication and three XOR operations: $3n + 2$. One inversion-multiplication over finite field GF(2^n) is computed after getting the values of numerator and denominator. Then we can get the total sequential runtime and parallel runtime of point-addition:

$$T_p = (n/2)(5 - (1/2)^n) + 2n + 4. \quad (9)$$

$$T_s = (n/2)(13 - 6(1/2)^n) + 3n + 10. \quad (10)$$

Consequently, the speedup is

$$S = \frac{(n/2)(13 - 6(1/2)^n) + 3n + 10}{(n/2)(5 - (1/2)^n) + 2n + 4}. \quad (11)$$

Table 4 and Figure 2 shows the performance evaluation of point-addition in conic curves cryptosystem over finite field GF(2^n).

Table 4. Performance Evaluation of Point-Addition

n	Tp_addition	Ts_addition
64	292	618
96	436	922
128	580	1226
160	724	1530
192	868	1834
224	1012	2138
256	1156	2442

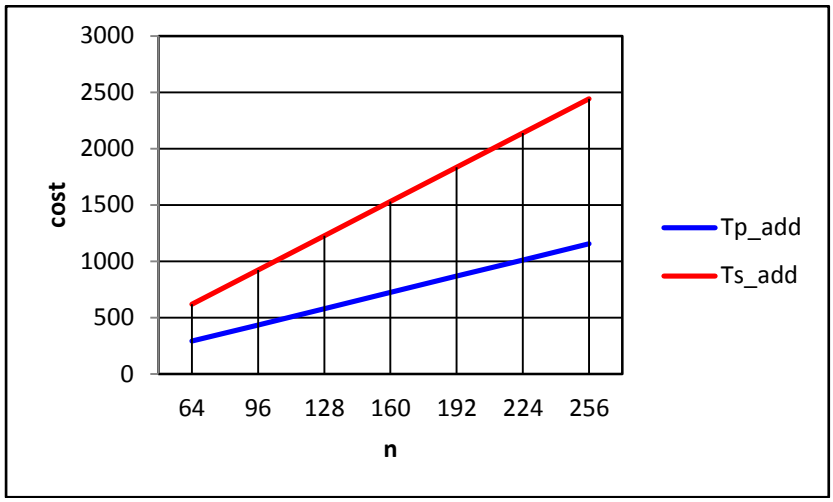

Figure 2. Performance Comparison of Point-Addition

4.3. Parallel Point-Double

We can use two methods to parallel the operation of point-double in conic curves cryptosystem over finite field GF(2^n). One parallel method is the same with the one for paralleling point-addition. The other one takes advantage of the fast operation of square over finite field GF(2^n) and achieves higher efficiency. The operation of square over finite field GF(2^n) has a simplest definition. For $c(Z) = a(Z)^2$, the value is $a_{n-1}Z^{2n-2} + \cdots + a_2Z^4 + a_1Z^2 + a_0$ .Obviously, the coefficient of polynomial $c(Z)$ could be figured out in one assignment. To get the mapping value of $c(Z)$over finite field GF(2^n), we have to compute one reduction over finite field GF(2^n).Then the parallel runtime of calculating numerator is the sum of one operation of assignment and one reduction over finite field GF(2^n): $\lceil log_2((n-1)/2) \rceil + 3$. This value is smaller than *2n*, the value obtained by the method of parallel point-addition, because we often use large integer *n* in this kind of cryptosystem. Consequently, the sequential runtime is the sum of one assignment, one reduction and one XOR operation: $(5n-1)/2$. Other operations are the same with the ones in point-addition, so the parallel runtime and the sequential runtime of point-double are

$$T_p = (n/2)(5-(1/2)^n) + \lceil log_2((n-1)/2) \rceil + 7. \quad (12)$$

$$T_s = (n/2)(13-6(1/2)^n) + (5n+15)/2. \quad (13)$$

Therefore, the speedup is

$$S = \frac{(n/2)(13-6(1/2)^n) + (5n+15)/2}{(n/2)(5-(1/2)^n) + \lceil log_2((n-1)/2) \rceil + 7}. \quad (14)$$

As showed in Table 5 and Figure 3, the method of paralleling point-double could accelerate the speed of point-double significantly.

Table 5. Performance Evaluation of Point-Double

n	Tp_double	Ts_double
64	172	583.5
96	253	871.5
128	333	1159.5
160	414	1447.5
192	494	1735.5
224	574	2023.5
256	654	2311.5

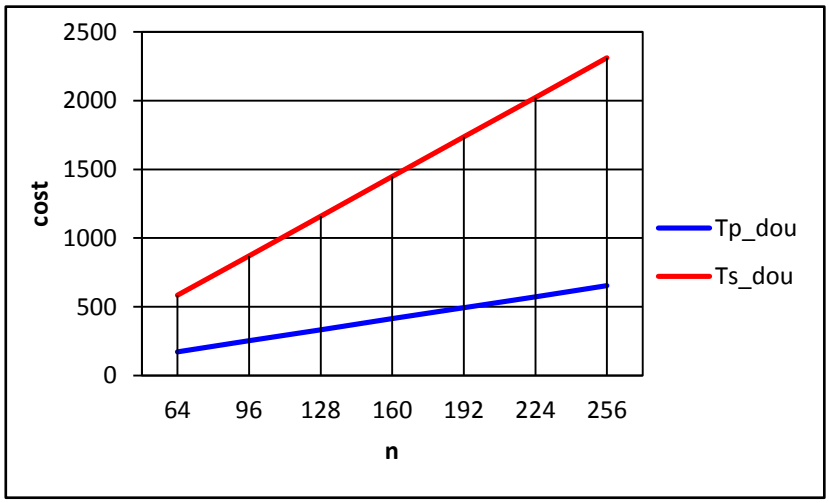

Figure 3. Performance Comparison of Point-Double

4.4. Parallel Point-Multiplication

As our previous research in [8] proposed, we use the algorithm of "square-addition-multiplication"to compute point-multiplication. The parallel runtime of point-multiplicationis

$$T_p = (t-1)T_{p-double} + T_{p-addition} + 1 * T_{communication-unit}. \quad (15)$$

The parameter *t* denotes the numbers of bit in coefficient *k* for computing point-multiplication *kP*. The sequential runtime of point-multiplication over finite field GF(2^n) based on point-addition and point-double using the algorithm of "square-addition-multiplication" is

$$T_s = (t-1)T_{s-double} + ((t-1)/2)T_{s-addition}. \quad (16)$$

Table 6. Performance Evaluation of Point-Multiplication

t	n	Ts	Tp	Speedup
5	32	1810	497	3.641851
15	32	6335	1367	4.634236
25	32	10860	2237	4.854716
35	32	15385	3107	4.951722
5	64	3570	961	3.71488
15	64	12495	2631	4.749145
25	64	21420	4301	4.980237
35	64	30345	5971	5.082063
5	96	5330	1425	3.740351
15	96	18655	3895	4.789474
25	96	31980	6365	5.024352
35	96	45305	8835	5.1279
5	128	7090	1889	3.753309
15	128	24815	5159	4.810041
25	128	42540	8429	5.046862
35	128	60265	11699	5.151295

Then, we obtain

$$T_p = (t-1)\Big((n/2)\big(5-1/2^n\big) + \lceil log_2\big((n-2)/2\big)\rceil + 7\Big) + 9n/2 - n/2^{n+1} + 5 \quad (17)$$

$$T_s = \Big((t-1)/2\Big)\Big(3n/2\Big(13-6(1/2)^n\Big) + 8n + 25\Big). \quad (18)$$

Therefore, the speedup is

$$S = \frac{\Big((t-1)/2\Big)\Big(3n/2\Big(13-6(1/2)^n\Big) + 8n + 25\Big)}{(t-1)\Big((n/2)\big(5-1/2^n\big) + \lceil log_2\big((n-2)/2\big)\rceil + 7\Big) + 9n/2 - n/2^{n+1} + 5}. \quad (19)$$

The performance evaluation and speedup ratio of point-multiplication are depicted in Table 6 and Figure 4, while the parameter *t* varies from 5 to 35 and parameter *n* increases from 32 to 128.

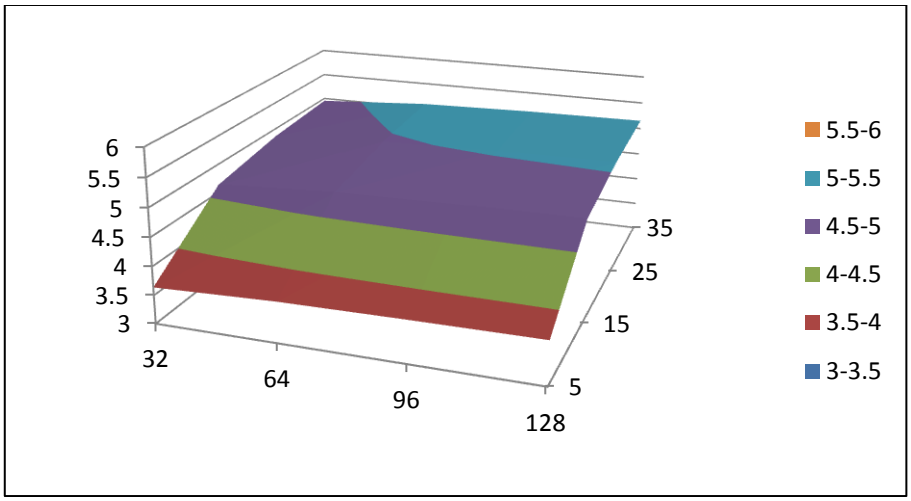

Figure 4. Speedup Ratio of Point-Multiplication

Consequently, the parallel algorithms proposed in this paper could improve the performance of conic curves cryptosystem for the reason that point-addition, point-double and point-multiplication are the basic operations for constructing security protocols in conic curves cryptosystem over finite field GF(2^n).

5. Conclusions

In this paper, we presented four parallel algorithms for conic curves cryptosystem over finite field GF(2^n). Firstly, one mathematical operation was paralleled by analyzing the data dependencies of their traditional algorithms. The probabilities of different cases were considered to compute the runtime of the modular-multiplication algorithm. Then we proposed the parallelization of point-addition and point-double according to this modular-multiplication operation and two other mathematical operations, for which we proposed their parallel algorithms in our previous works. The point-multiplication was paralleled by using the method we proposed in our previous work. We made the quantitative performance comparison to demonstrate that our parallel techniques could accelerate the speed for calculating three point operations in conic curves cryptosystem over finite field GF(2^n).

Only three point operations in conic curves cryptosystem over finite field GF(2^n) were paralleled in this paper. We plan to design the parallel security protocols for conic curves cryptosystem over finite field GF(2^n) based on the proposed parallel algorithms in the future.

Acknowledgments

This study is sponsored by the fund of the State Key Laboratory of Software Development Environment under Grant No. SKLSDE-2012ZX-06, the Hi-tech Research and Development Program of China (863 Program) under Grant No. 2011AA01A205, Beijing Natural Science Foundation under Grant No. 4122042, the National Natural Science Foundation of China under Grant No. 61232009, the National Natural Science Foundation of China under Grant No. 61003015 and the National Natural Science Foundation of China under Grant No. 61370059.

References

[1] Y. Cai, L. Zhao and Y. Jin, "A Public-Key Cryptosystem Based on Conic Curve in Finite Field GF(2~n) (in Chinese)", Acta Electronica Sinica, vol. 34, no. 8, **(2006),** pp. 1464–1468.
[2] Z. Chen and X. Song, "A Public-Key Cryptosystem Scheme on Conic Curves over The Ring Zn", Proceedings of the 6th International Conference on Machine Learning and Cybernetics, **(2007)** August 19-22, Hong Kong, China.
[3] B. Wang, W. Zhu and Q. Sun, "Public Key Cryptosystem Based on the Conic Curves Over Zn (in Chinese)", Sichuan Univ. (Engin Sci Ed), vol. 37, no. 5, **(2005),** pp. 112–117.
[4] Z. Cao, "A Public Key Cryptosystem Based On A Conic Over Finite Fields Fp (in Chinese)", Advances in Cryptology: Chinacrypt98, Science Press, **(1998),** pp.45–49.
[5] Z. Cao, "Conic analog of RSA cryptosystem and some improved RSA cryptosystems", Natural Science Journal of Heilongjiang University, vol. 16, no. 4, **(1999),** pp. 5–18.
[6] Y. Li, L. Xiao, Y. Hu, A. Liang and L. Tian, "Parallel Algorithms For Cryptosystem On Conic Curves Over Finite Field Fp", Proceedings of the 9th International Conference on Grid and Cloud Computing, **(2010)** November 1-5, Nanjing, China.
[7] Y. Li, L. Xiao, A. Liang and Z. Wang, "Parallel Point-Addition And Point-Double For Cryptosystem On Conic Curves Over Ring Zn", Proceedings of the 11th International Conference on Parallel and Distributed Computing, Applications and Technologies, **(2010)** December 8-11, Wuhan, China.
[8] Y. Li and L. Xiao, "Parallel Point-Multiplication for Conic Curves Cryptosystem", Proceedings of the 3rd International Symposium on Parallel Architectures, Algorithms and Programming, **(2010)** December 18-20, Dalian, China.
[9] Y. Li, L. Xiao, Z. Wang and H. Tian, "High Performance Point-Multiplication for Conic Curves Cryptosystem Based on Standard NAF Algorithm and Chinese Remainder Theorem", Proceedings of 2011 International Conference on Information Science and Applications, **(2011)** April 26-29, Jeju, Korea.
[10] Y. Li, L. Xiao, G. Qin, X. Li and S. Lei, "Comparison of Three Parallel Point-Multiplication Algorithms on Conic Curves. Proceedings of the 11th International Conference on Algorithms and Architectures for Parallel Processing, **(2011)** October 24-26, Melbourne, Australia.
[11] Y. Li, L. Xiao, S. Chen, H. Tian, L. Ruan and B. Yu, "Parallel Extended Basic Operations for Conic Curves Cryptography over Ring Zn. Proceedings of the 9th IEEE International Symposium on Parallel and Distributed Processing with Applications Workshops, **(2011)** May 26-28, Busan, Korea.
[12] N. Koblitz, "Elliptic Curve Cryptosystems", Mathematics of Computation, vol 48, **(1987),** pp.203-209.
[13] V. S. Miller, "Uses of elliptic curves in cryptography", Advances in Cryptology-Crypto'85. Lecture Notes in Comput. Sci, Springer-Verlag, Berlin, vol. 218, **(1986),** pp. 417–426.
[14] M. N. Hassan, M. Benaissa and A. Kanakis, "Flexible Hardware /Software Co-Design For Scalable Elliptic Curve Cryptography for Low-Resource Applications", Proceedings of the International Conference on Application-Specific Systems, Architectures and Processors, **(2010)** July 7-9, Rennes, France.
[15] M. N. Hassan and M. Benaissa, "A Scalable Hardware/Software Co-Design for Elliptic Curve Cryptography on Picoblaze Microcontroller. Proceedings of 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems, **(2010)** May 30 - June 2, Paris, France.
[16] S. C. Seo, D. Han, H. C. Kim and S. Hong, "TinyECCK: Efficient Elliptic Curve Cryptography Implementation over GF(2m) on 8-bit Micaz Mote", IEICE Transactions on Information and Systems, vol. E91-D, **(2008),** pp. 1338-1347.
[17] S. C. Seo, D. Han and S. Hong, "TinyECCK16: An Efficient Field Multiplication Algorithm on 16-Bit Environment and its Application to Tmote Sky Sensor Motes", IEICE Transactions on Information and Systems, vol. E92-D, **(2009),** pp. 918-928.
[18] A. E. Cohen and K. K. Parhi, "Fast Reconfigurable Elliptic Curve Cryptography Acceleration for GF(2 m) on 32 Bit Processors", Journal of Signal Processing Systems, vol.60, **(2010),** pp. 31-45.
[19] Y. Li and L. Xiao, "Parallelization of Two Arithmetic Operations over Finite Field GF(2n)", International Journal of Security and Its Applications, vol. 6, no. 2, **(2012),** pp.223-228.

Authors

Yongnan Li, he is a Ph.D. student at School of Computer Science and Engineering, Beihang University. His main research areas are computer architecture, cloud computing, parallel computing and information security.

Limin Xiao, he is a professor of the Institute of Computer Architecture, Beihang University. He is also a senior membership of China Computer Federation. His main research areas are computer architecture, computer system software, high performance computing, virtualization and cloud computing.