SUPPLEMENT ISSUE

ARTICLE

# PERFORMANCE EVALUATION OF DIGITAL SIGNATURE USING RNS MONTGOMERY MULTIPLICATION

**Payam Shadman Mehr[1*] and Mohammad Esmaeildoust[2]**

*[1] MSc Student, Science and Research University, Islamic Azad University, Bushehr, IRAN*

*[2]Faculty of Marine Engineering, Khorramshahr University of Marine Science and Technology, Khorramshahr, IRAN*

## ABSTRACT

*Digital signature is one the most important application of cryptography algorithm. Selecting a digital signature algorithm with proper speed and security is crucial. Digital signature standard is based on elliptic curve cryptography (ECC) which provides a security equivalent to other methods such as digital sign based on RSA with much shorter key length. Since this method is composed of point multiplications, improving the multiplication operation may have an influence on the efficiency of whole system and increase the operation speed. In order to reach these goals, Montgomery multiplication algorithm along with residue number system (RNS) are employed. The results show the improvement in the implementation of considered digital signature algorithm.*

**KEY WORDS**
*Residue number system, RSAMontgomery, digital signature algorithm, elliptic curve, standard digital signature*

Published: 10 July 2016

***Corresponding Author**
Email:payamshadman@gmail.com
Tel.: 06153267129
Fax: 06153268999Century

## INTRODUCTION

Today, by moving from traditional to digital era, protecting the information draws the attention of people as a fundamental and important element in exchanging the massages, commercial exchanges, and consequently, issues related to information security including ensuring the absence of eavesdropping the transferred information through public channels by a third unauthorized person, confirming the identity of sender, and ensuring noinformation change by unauthorized persons during sending process. Because of extensive changes in life and activities of individuals, organizations and governments which are resulted from the growth and development of computer network applications especially the Internet and digital computers, information security plays a vital important role in today's world. With regard to the characteristics of network communications which expose the organization information to foreign hosts, organizations need a system which prevents the accessibility of unauthorized individuals to critical and vital information. In order to maintain the correctness and accuracy of information, some methods like digital signature is used [1]. In this paper, in order to achieve high speed implementation of digital signature based on ECC, RNS Montgomery multiplication [1] is employed.

The paper is organized as follows; in the second section, the related background includes elliptic curve digital signature algorithm (ECDSA), Montgomery multiplication and RNS will be discussed. In the third section, implementation of ECDSA using RNS Montgomery multiplication will be detailed. Performance comparison will be discussed in section four, and finally section five concludes the paper.

### Related Background

### Digital Signature

One of the most important responsibilities of digital signature is to confirm the identity of the person signing a document and consequently the originality of received information by using a series of rules and algorithms of cryptography. Digital signature was proposed in 1985 for the first time, and then in 1991 the algorithm of digital signatures was presented, and finally in 1994 it was registered in NIST (National Institute of Standard and Technology) [2] as a digital signature standard (DSS) [3-4]. Elliptic curve digital signature algorithm is a kind of DSA which is established on the basis of elliptic curve cryptography (ECC) [5,6]. According to Table A1, ECC can provide equivalent security with shorter key length compared with the competing methods of cryptography with public key such as RSA [7-10]. Therefore, it is considered as an appropriate method in applications having a limited memory [11].

## MATERIALS AND METHODS

ECDSA algorithm is presented in [Fig. 1]. According to the [Fig. 1], ECDSA requires four stages of point multiplications. On the other hand, point multiplication operation is conducted based on a collection of point doubling and point addition.

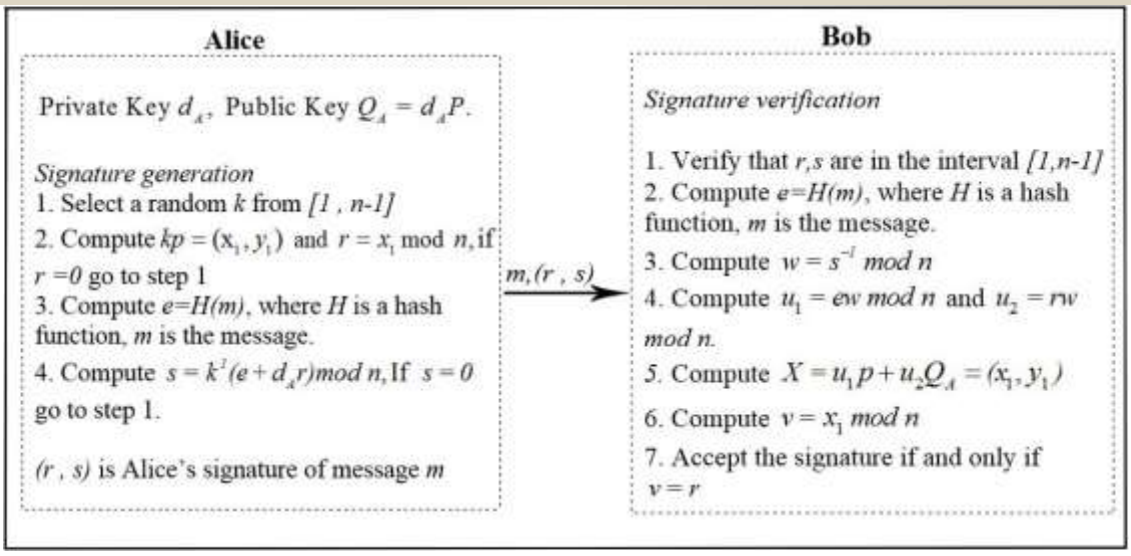


**Fig. 1**: ECDSA algorithm

## Residue Number System

RNS is a non-weighted number system which performs the arithmetic operations like multiplication, subtraction and addition without propagating the partial carry [12]. Since conducting the calculations on the remainders is much faster than using weighted numbers due to the smaller numbers in the arithmetic operations, the speed of conducting the arithmetic operations in this system is much higher and would also decrease the consumed power [7]. In fact, the high speed of calculation due to the un-propagated partial carry, shorter data directions, reducing the operation in parallel channels, are the reasons of high popularity and usage of RNS in calculation systems [12].

## RNS Montgomery Multiplication

One of the most effective methods to increase the speed of modular multiplication is Montgomery algorithm [13, 14, 15, 16]. In this method, the multiplication operation of two numbers in a special modular is conducted without executing the time consuming operation of division. Investigations conducted on this field presented a new method of Montgomery multiplication algorithm on the basis of residue number system in [17-19].

| ***Algorithm 1: RNS Montgomery Multiplication[20]*** |
|---|
| $r \equiv x \times y \times M^{-1} \pmod N$ |
| $1: D = X \times Y \ (d_i = \lvert x_i \times y_i \rvert_{P_i} \text{ in base } \mathrm{B_n} \text{ and } d_i' = \lvert x_i' \times y_i' \rvert_{P_i'} \text{ in base } \mathrm{B_n'})$ |
| $2: q_i = \left\lvert d_i \times \lvert -N \rvert^{-1}_{P_i} \right\rvert_{P_i} \ \mathrm{B_n}\, \textit{base extension}$ |
| $3: q_i \text{ in } \mathrm{B_n} \longrightarrow q_i' \text{ in } \mathrm{B_n'}$ |
| $4: r' = (d_i' + q_i' \times N_i')M^{-1} \ \mathrm{B_n'}\, \textit{base extension}$ |
| $5: r \text{ in } \mathrm{B_n} \longleftarrow r' \text{ in } \mathrm{B_n'}$ |

Algorithm 1 shows the stages of Montgomery multiplication in RNS. In this algorithm, $x_i$ and $y_i$ are the residue representation of numbers in RNS base *B*, and $x'_i$and $y'_i$ are the residue representation in the $B'$auxiliary modular set.

## ECDSA implementation

In this section, ECDSA are implemented by using RNS Montgomery Multiplication and the execution time of the point multiplication is estimated for a 256 bit key size.

[Table 1] shows the recommended value of ECDSA parameters in *GF(P)*. Moreover, according to [Table 1], two values of 256 bit for *X* and *Y* and *P* modular for a 256 bit modular multiplication and also two values of 64 bit for *B*1 and *B*2 for 64 bit multiplication and adding are introduced.

**Table 1:**The Representation of Values Used in the Required Multiplication Operation

| Parameter Name | Value on the basis of 16 | Explanation |
|---|---|---|
| P | FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFEFFFFFC2F | The main modular having a length of 256 bit which is in the formof $2^{256} - 2^{32} - 2^{9} - 2^{8} - 2^{7} - 2^{6} - 2^{4} - 1$ |
| X | FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFEFFF22C2F | A value of 256 bit for modular multiplication |
| Y | FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFF3FFF3FFEFFF22C2F | A value of 256 bit for modular multiplication |
| B1 | 10000000000000000 | A value of 64 bit for 64 bit multiplication and adding |
| B2 | FFFFFFFFFFFFFBFF | A value of 64 bit for 64 bit multiplication and adding |

System specifications for the ECDSA implementation are 32 bit operating system, core i5 CPU with 4 GB RAM and also implemented with Java language in Eclipse environment. [Table 2] shows the final delay in point addition, point doubling and point multiplication. [Table 2] shows the implementation results. It can be seen that by using RNS Montgomery multiplication, noticeable improvement in speed is achieved.

**Table 2:**Estimating the Operation Time (µs)

| | Modular Multiplication | Point Adding | Point Duplication | Point Multiplication |
|---|---|---|---|---|
| **Implementation by Using RNS Montgomery Multiplication** | *8.6* | 137.6 | 86 | 39628.8 |

## RESULTS

### Performance evaluation

[Fig. 2] shows the performance comparison of multiplication operation using RNS Montgomery algorithm considerably improve the operation time of point multiplication compared with a regular modular multiplication.

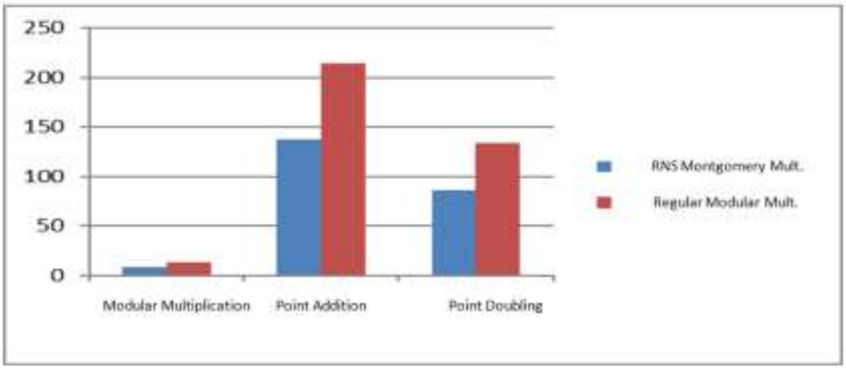


**Fig. 2:**Comparison Point Addition, Doubling and Multiplication (µs)

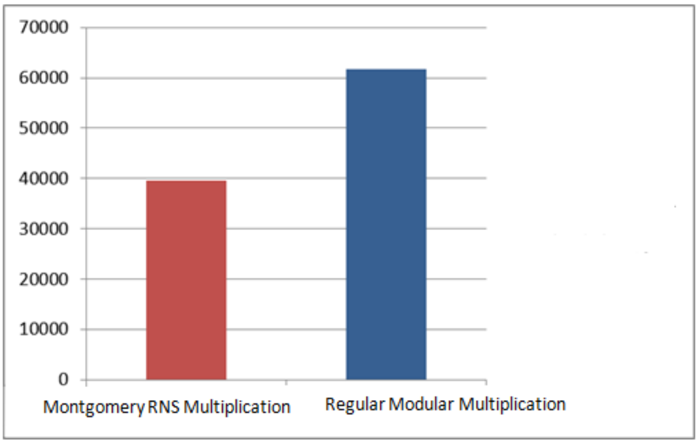


**Fig. 3:** Point Multiplication delay μs)

[Fig. 3] shows the point multiplication implementation with/without using RSA Montgomery multiplication. The result shows the noticeable improvement in speed compared to original point multiplication. As discussed in section2, four point multiplications are required in ECDSA [5-6] implementation. Therefore based on the results in [Fig. 3], it can be concluded that higher speed of ECDSA is achieved compared to original ECDSA [5-6].

## CONCLUSION

In this paper, Point multiplication as the main operation in ECDSA is implemented by using RSA Montgomery multiplication. In order to implement point multiplication, point addition and doubling are implemented with Java language on eclipse environment. The result shows that noticeable improvement is achieved by employing RNS Montgomery multiplication compared to original implementation.

### CONFLICT OF INTEREST

There is no conflict of interest

### ACKNOWLEDGEMENTS

None

### FINANCIAL DISCLOSURE

None

## REFERENCES

[1] Diffie W, Hellman ME.[1976] New directions in cryptography. IEEE Trans.Information Theory, IT-22(6):644–654.

[2] 10NSA .The case for elliptic curve cryptography. http://www.nsa.gov/business/programs/elliptic curve.shtml. [2009].

[3] 85NBS–SHS. Secure Hash Standard (SHS). FIPS Publication 180-2, NationalBureau of Standards, [2003] http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf.

[4] U.S. Department of Commernce / National Institute of Standards and Technolgoy, Digital Signature Standard (DSS), http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf.

[5] Koblitz N.[1987]Elliptic Curve Cryptosystem, Mathematics of Computation.48:203–209.

[6] Miller V. [1985] Uses of Elliptic Curves in Cryptography, Advances in Cryptology- CRYPTO, Williams HC(ed.). 218 :417–426.

[7] Posch KC, Posch R. [1992] Residue Number System: a Key to Parallelism in Public Key Cryptography, IEEE Symposium on parallel distributed processing. 432-435.

[8] Bajard JC, Imbert L. [2004] A Full RNS Implementation of RSA, IEEE Transactions on Computers. 53(6): 769-774.

[9] Skavantzos A, Abdallah M. [1999] Implementation Issues of the Two-level Residue Number System with Pairs of Conjugate Moduli, IEEE Trans. on Signal Processing. 47(3):826- 838.

[10] Rivest RL, Shamir A, Adleman LM. [1978] A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Commun. ACM. 21 (2):120–126.

[11] Arjen K, Lenstra, Eric R. [1999] Verheul. Selecting cryptographic key sizes.*Journal of Cryptology.* 14:255-293.

[12] Navi K, Molahosseini ASM.[2011] Esmaeildoust, How to teach residue number system to computer scientists and engineers?IEEETrans. Edu.54 (1):156–163.

[13] HanserCh.[2010] New Trends in Elliptic Curve Cryptography, Master Thesis,Graz University of Technology.

[14] Montgomery PL. [1985] Modular Multiplication without Trial Division. Math. Computation. 44:519-521.

[15] Hamano T, Takagi N, Yajima S,Preparata FP. [1995] O(n)-Depth Circuit Algorithm for Modular Exponentiation, Proc. 12th IEEE Symp. Computer Arithmetic, Knowles S and McAllister WH, eds.188-192.

[16] Koc CK,Acar T. [1997] Fast Software Exponentiation in GF (2k), Proc. 13th IEEE Symp. Computer Arithmetic, Lang TJ, Muller M, Takagi N, eds.225-231.

[17] Bajard J, Didier L ,Kornerup P. [1998] An RNS Montgomery's Modular Multiplication Algorithm, IEEE Trans. Computers. 47(2):167-178.

[18] Bajard J, Didier L,Kornerup P. [2001] Modular Multiplication and Base Extensions in Residue Number Systems, Proc. 15th IEEE Symp. Computer Arithmetic (ARITH '01).59-65.

[19] Wang W, Swamy MNS, Ahmad MO, Wang Y.[2000] A high-speed residue-to-binary converter and a scheme of its

VLSI implementation, IEEE Transactions on Circuits and Systems-II. 47(12):1576–1581.
[20] Nozaki H, Motoyama M, Shimbo A, Kawamura S. [2001] Implementation of RSA Algorithm Based on RNS Montgomery Multiplication, Proc. Cryptographic Hardware and Embedded Systems (CHES 2001). 364-376.