Improved Modular Inversion over GF(*p*)

Jonghwa Choi
Dept. of Information Communication Engineering
Chungbuk National University, Chungbuk, Korea

Yong-dae Kim
Institute of Technology
Embedded Solution Co., Ltd, Chungbuk, Korea

Youngil Ahn
Dept. of Information Communication Engineering
Chungbuk National University, Chungbuk, Korea

Younggap You*
School of Electrical & Computer Engineering
Chungbuk National University, Chungbuk, Korea

ABSTRACT

This paper proposed a new modular inverse algorithm based on the right-shifting binary Euclidean algorithm. For an n-bit numbers, the number of operations for the proposed algorithm is reduced about 61.3% less than the classical binary extended Euclidean algorithm. The proposed algorithm implementation shows substantial reduction in computation time over Galois field GF(p).

***Keywords**: Modular arithmetic, Modular Inversion, Cryptography.*

1. INTRODUCTION

The modular arithmetic is an essential operation in many public-key cryptosystems, including RSA and elliptic curve crypto systems (ECC) [1][2]. The modular inversion is one of the slowly arithmetic operations. The modular inversion is the most time-consuming and complicated operation of the field arithmetic operations. The speed of the modular inversion brings a bottleneck in cryptographic process. A high speed inversion is a key to make systems efficient.

There are some well-known inversion algorithms: such as the binary extended Euclidean method [3-5]. This method can quickly compute multiplicative inverses. The binary extended Euclidean method is simple and fast, because it requires only modular additions, subtractions and shifting. This algorithm has some weak points. For example, it has a step comprising comparison of two integers. This paper proposed an implement modular inversion algorithm addressing the speed of comparisons. The Kaliski algorithm achieved speed improvement of Montgomery modular inversion [6]. The proposed inversion algorithm borrowed a part of the Kaliski method yielding faster inversion in the integer domain.

Section 2 explains the previous work on the modular inverse algorithm. Section3 illustrates our algorithm for improvement and analyzes the proposed algorithm. Section 4 evaluates the proposed algorithm, followed by the conclusion in Section 5.

** Corresponding author. E-mail : ygyou@chungbuk.ac.kr*
Manuscript received June 10, 2007 ; accepted June 25, 2007

2. MODULAR INVERSE ALGORITHM

The modular inversion of an integer x over the Galois field GF(p) is defined as the integer y satisfying $xy \equiv 1 \pmod p$.

$$y = \text{Mod\_Inv}(x) = a^{-1} \pmod p \tag{1}$$

The modular inversion takes long operation time due to the division by the modulus in its computations [7]. In this point, researchers have been seeking methods to alter the division and modular inversion less time consuming. The binary extended Euclidean algorithm has been modified to a faster modular

Algorithm 1: Modular inversion for GF(p)
Input: An integer gx, where $x \in (1, p)$, and a modulus p with gcd(p, 2) = 1.
Output: An integer y, such that $y \equiv x^{-1} \pmod p$.

```
Step 1.   u = p, v = x, r = 0, s = 1.
Step 2.   if (u is even) then
              u = u/2, r = r/2 (mod p).
Step 3.   if (v is even) then
              v = v/2, s = s/2 (mod p).
Step 4.   if (u and v are both odd) then
            if (u > v) then
              u = (u - v), r = (r - s) (mod p),
            else
              v = (v - u), s = (s - r) (mod p).
Step 5.   if (u ≠ 1 and v ≠ 1) then goto Step 2.
            else if (u = 1) return y = r.
              else return y = s.
```

Algorithm 1 requires general additions and subtractions; it is relatively simple and fast. However, this algorithm has some shortcoming. After the Step 4, one of the values of u and v must become even while the other remains odd. In the next iteration only one of the Step 2 or the Step 3 will be performed. Moreover, if u or v can be divided by some power of 2, only the Step 2 or 3 will repeat while the other steps will do nothing. These null operations make the computing process inefficient. The Step 4 and the Step 5 see a comparison of large numbers. The comparison degrades the efficiency since it employs the subtraction.

3. IMPROVED BINARY EXTENDED EUCLIDEAN ALGORITHM

The proposed pseudo-code of the improved binary extended Euclidean algorithm is given below.

Algorithm 2: Improved modular inversion for GF(p)
Input: An integer x, where $x \in (1, p)$, and a modulus p with gcd(p, 2) = 1.
Output: An integer y, such that $y \equiv x^{-1} \pmod p$.

```
Step 1.   u = p, v = x, r = 0, s = 1.
Step 2.   while (v is even)
              v = v/2, s = s/2 (mod p), goto Step 4.
Step 3.   while (u is even)
              u = u/2, r = r/2 (mod p), goto Step 5.
Step 4.   if (v = 1) then return y = s.
Step 5.   if (u > v) then
              u = (u - v)/2, r = (r + s)/2 (mod p),
              if (u is even) then goto  Step 3.
              else goto Step 5.
          else
              v = (v - u)/2, s = (s + r)/2 (mod p),
              if (v is even) then goto  Step 2.
              else goto Step 4.
```

inversion algorithm over Galois field GF(p) [4]. The classical binary extended Euclidean algorithm is given below.

Note: The notation "$s/2 \pmod p$" in the pseudo-code means s divided by 2 modulo p.

In the proposed algorithm, both the addition and the subtraction are combined with the halving (right shift). The halving is performed at every step. Since the v value provides with an early termination condition, the proposed modular inversion algorithm demands fewer operations than conventional modular algorithms.

A detail analysis of the proposed algorithm, including its correctness, is given below. During the iterations of the proposed algorithm, the following equations hold:

$$\gcd(p, x) = \gcd(u, v) = 1 \quad (2)$$

$$us + vr \pmod p \equiv 0,\ s \geq 1,\ u \geq 1 \text{ and } 0 < v \leq x \quad (3)$$

When the Step 1 is done, two equations hold:

$$v = x,\ s = 1, \text{ hence } xs \pmod p \equiv v \quad (4)$$

$$u = p,\ r = 0, \text{ hence } xr \pmod p \equiv -u \quad (5)$$

At the Step 2, if v is even, both sides of the equation (4) can be divided by factor 2 simultaneously, because the modulus p is definitely odd. An even number, v, can be right shifted one bit. For the other side of the equation (4), different processing for s is required according to given different values. If s is even, the result of $s/2$ can be obtained by right shifting. If s is odd, the operation of $s/2 \pmod p$ is accomplished by adding the modulus p and then right shifting. As shown in the *Algorithm* 2, the two steps handling odd number can be processed using an adder in one step. Both s and p, which is right shifted one bit, are fed into the adder with '1' into the carry-in. At the Step 3, the operations for both u and r of the equation (5) correspond to the operations of v and s, respectively. At the Step 5, if the values of u and v are odd, the difference of u and v becomes even. Thus, a new equation for the Step 5 can be obtained from the equations (4) and (5) as below

$$[\text{Eq.}(4) + \text{Eq.}(5)]/2 \equiv x(r+s)/2 \equiv (v-u)/2$$
$$\equiv x(r+s)/2 \equiv -(u-v)/2 \quad (6)$$

According to the equation (6), the new values of v and s, and u and r can be computed at the Step 5. Thus, after the last iteration with $v = 1$, $xs \equiv 1 \pmod p$ from the equation (4).

4. VERIFICATION AND EVALUATION

For the verification, we are compared the two algorithms: the binary extended Euclidean algorithm and the proposed algorithm. The number of operations of the two algorithms has been calculated using a C program running in a Pentium-4 3GHz processor with 1Gbytes of main memory. Modular inversions checked the range from 128-bit through 521-bit. The program runs for a full day to check more than three millions cases. Figure 1 and Table 1 illustrate the increase of operations per bit for the two algorithms for comparison purposes.

The number of operations is a typical measure evaluating algorithms. For 128-bits, the proposed algorithm performs 335 operations in Table 1, while the classical algorithm demands 888 operations. For a larger number of bits the number of operations increases nearly proportional to the number of bits. For an n-bit numbers, the number of computations for the proposed algorithm is reduced about 61.3% less than the classical binary extended Euclidean algorithm.

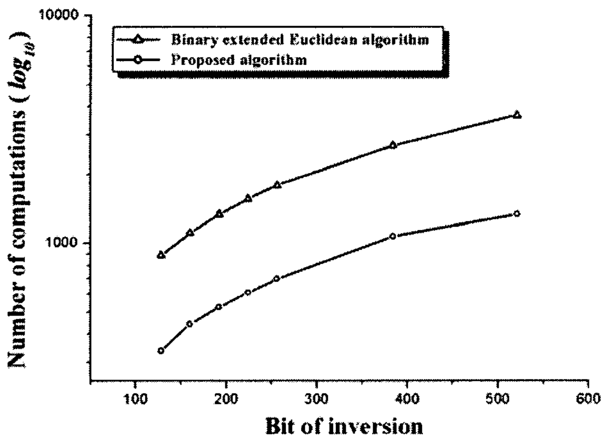

Fig. 1. Number of operations of modular inversion algorithms

Table 1. Performance Comparison of Modular Inversion

Bits	Computation cycles for one inversion		Improvement rate (%)
	Classical algorithm	Proposed algorithm	
128	888	335	62.3
160	1114	442	60.3
192	1340	524	60.9
224	1566	606	61.3
256	1792	697	61.1
384	2695	1074	60.1
521	3665	1339	63.4

5. CONCLUSION

This paper proposes a modular inversion for fast computation over Galois field GF(p). The proposed algorithm is based on the right-shifting binary Euclidean algorithm. For an n-bit numbers, the number of operations for the proposed algorithm is reduced about 61.3% less than the classical binary extended Euclidean algorithm. The proposed algorithm implementation shows substantial reduction in computation time over a Galois field GF(p).

REFERENCES

[1] A. Menezes, **Elliptic Curve Public Key Cryptosystems,** Massachusetts, Kluwer Academic Publishers, 1993.

[2] C. J. McIvor, M. M. McLoone and J. V. McCanny, "Hardware Elliptic Curve Cryptographic Processor Over *GF(p)*," **IEEE Transactions on Circuits and Systems,** September 2006, pp. 1946-1957.

[3] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, **Handbook of Applied Cryptography**, CRC Press, 1997.

[4] D. Sun, "A Computer Algorithm for the Computation of Inverse in Finite Field GF(p)," **Communications and Security**, vol. 4, 1997, pp.57-59.

[5] T. Zhou, X. Wu, G. Bai and H. Chen, "New Algorithm and Fast VLSI Implementation for Modular Inversion in Galois Field GF(p)," **Proc. Int. Conf. on Comm., Circuits and Systems and West Sino Expositions**, vol. 2, July 2002, pp. 1491-1495.

[6] B. S. Kaliski, "The Montgomery Inverse and its Applications," **IEEE Transactions on Computers**, vol. 44, no. 8, August 1995, pp. 1064-1065.

[7] R. L. Rivest, A. Shamir, and L. Adle, "A Method for Obtaining Digital Signatures and Public-key Crypto systems," **Communications of the ACM**, vol. 22, no. 2, February 1978, pp. 120-126.

Jonghwa Choi
received the B.S. degree in Semiconductor Engineering from Chungbuk National University, Cheongju, Korea in 2002, and the M.S. degrees in Computer and Communication Engineering from Chungbuk National University, Cheongju, Korea in 2004. From 2003 to 2006, he was with engineer of EmbeddedSolution Company in Korea. He is currently pursuing the Ph.D. degree at the graduate school of the Chungbuk National University, Korea. His current research interests include algorithm and architecture for computations, VLSI system design, and cryptographic system.

Yong-dae Kim
received the B.S., M.S. and Ph.D. degrees in Computer and Communication Engineering from Chungbuk National University, Cheongju, Korea in 1991, 1993, and 2007, respectively. He is currently a teaching staff in the School of Electrical and Computer Engineering at Chungbuk National University, Cheongju, Korea. His current research interests include computer arithmetic, computer architecture, embedded system, and cryptographic system.

Youngil Ahn
received the B.S. degrees in Electronic Engineering from Hanbat University, Daejeon, Korea in 2006. He is currently pursuing the M.S. degree at the graduate school of the Chungbuk National University, Korea. His current research interests include cryptographic system, embedded system.

Younggap You
received the B.S. degree in Electronic Engineering from the Sogang Jesuit University, Seoul, Korea and the M.S. and Ph.D. degrees in Electrical Engineering from the University of Michigan, Ann Arbor, U.S.A., in 1981 and 1986, respectively. From 1975 to 1979, he was with the Agency for Defense Development, Korea. He worked as a principal engineer at LG Semiconductor, inc., Seoul, Korea, from 1986 to 1988. He is currently a Professor in the School of Electrical and Computer Engineering at Chungbuk National University, Cheongju, Korea. His research interests are fault tolerant computing, computer architecture, cryptography, cellular system design, and frequency synthesis technology.