AN AREA EFFICIENT ARITHMETIC UNIT OF ELLITPIC CURVE CRYPTOGRAPHY PROCESSOR

Anil kumar M*

V. Sridhar**

ABSTRACT

In this paper we propose a new technique to achieve an area efficient inversion block specifically for NIST recommended prime field p521. This is obtained by modifying the Binary Inversion Algorithm. Reduction in area is obtained by replacing addition operation preceded by hard shift right in the architecture of Binary Inversion block by hard rotate right. Most of the earlier architectures on GF(p) have used simple carry propagation binary adders to achieve area saving. Hence the proposed method also assumes similar kind of adders. The new architecture designed has achieved reduction in area in terms of number of reduced logic gates.

Keywords: *Data security, Elliptic Curve Cryptography, Prime field arithmetic, Galois field arithmetic, Binary Inversion Algorithm.*

*Research Scholar, PET Research Foundation, PESCE, Mandya, Karnataka

**Professor, Department of E&C, PET Research Foundation, PESCE, Mandya, Karnataka

I. INTRODUCTION

The data security has become an important and urgent need for health care information, confidential communication, storage and financial services etc. The public key cryptosystem is the most efficient way to secure data transaction and to provide authentication. The challenge to implement the most popular public key cryptosystem, RSA is the increase in key size. Elliptic Curve cryptography (ECC) has been considered an alternative to RSA. The effectiveness of using elliptic curve cryptography is that it provides same security level with shorter keys than in RSA [1-3]. Therefore ECC has been used in smart cards, credit cards and mobile phones where area is a constraint.

The research on different algorithms and hardware accelerations have focussed on efficient implementation of elliptic curve scalar point multiplication Q=k.P. This is the fundamental operation of all elliptic curve cryptosystems. Software based implementations of ECC are flexible but inefficient because general purpose instruction set architecture of the underlying hardware is not optimized for cryptographic computations. An instruction set architecture can be extended to provide partial support for ECC related arithmetic operations. Hence a better approach would be to introduce a special arithmetic unit for accelerating modular operations or even complete scalar multiplications.

Two types of Finite Fields are generally used in ECC. Those are Finite Field over a large prime called as *Galois Field* GF(p) and Extended Binary Field that is known as *Galois Field* GF(2^k). The hardware complexity to implement ECC in GF(p) is little bit higher than that of in GF(2^k) but the advantage is that the k-bit arithmetic unit is capable to process any i-bit data where $1 \leq i \leq k$ [4-10]. Confining designs to binary fields limit the flexibility and may not be used for Elliptic Curve Digital Signature Algorithm. This algorithm in addition to EC point operation is based on normal integer modulo operations. For binary field designs these modulo operations must be done separately by using a processor or in a separate hardware.

The National Institute of Standards and Technology (NIST) has recommended elliptic curves over the five prime fields with moduli:

$p192 = 2^{192} - 2^{64} - 1$

$p224 = 2^{224} - 2^{96} + 1$

$p256 = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$

$p384 = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$

$p521 = 2^{521} - 1$

Except for p521, the powers appearing in these expressions are all multiple of 32. This property yield arithmetic operations especially fast on machines with word size 32. These arithmetic operations include addition, subtraction, multiplication, reduction, squaring and inversion. If all arithmetic operations are to be used, and we wish to represent a full range of integers in n bits, then other NIST recommended primes will not work except NIST recommended prime with modulus $2^{521}-1$.

In some of the earlier hardware accelerators designed, the focus is given to architecture flexibility [11] which reduces the efficiency in terms of area and speed. A cryptography algorithm is secure as long as no effective attack is found. If there is a threat, the algorithm has to be replaced, in particular of switching to a higher key length which restricts operations with lower key length. In this situation, when working with higher key length the architecture can be more specific. This increases its efficiency in terms of area and speed with a compromise on flexibility. So the proposed method has focused on this concept and has been applied to NIST recommended prime field p521 which has the highest key length among other NIST recommended prime fields. Inversion is the costliest operation among all the modular operations. Inversion operation is almost eliminated with projective coordinate systems with the cost of using parallel multipliers [12-13]. But in small devices like smart cards where area is a constraint, adding more multiplier units need more memory and thus increases the cost. Speeding up inversion operation in both fields has been gaining attention because inversion is the most time consuming operation when affine coordinates are selected. Hence the proposed method has aimed to design an area efficient architecture of inversion unit specifically for NIST recommended prime field p521.

The two commonly used approaches for computation of ordinary modular inverse are a binary algorithm derived from Montgomery modular inverse and Binary algorithm for ordinary inverse. Both of these approaches are based on the same algorithmic principle, which is the binary right shift greatest common divisor algorithm that calculates the value for two integers using halving and subtraction. Both of these algorithms are suitable for implementation in hardware since halving operation is equal to right shift. Even though the above two algorithms for the computation of inversion can be optimized by the proposed technique in terms of reducing the area with an increase in the speed of computations, proposed method has focussed on the Binary Inversion Algorithm (BIA).The proposed method has designed an area efficient architecture of Binary Inversion Algorithm specific to prime field p521 with a compromise on flexibility.

The proposed technique has reduced the area by removing two 521 bit adder blocks from the architecture of BIA. Reduction in area is obtained by replacing addition operation preceded by hard shift right by hard rotate right. Since the area occupied by hard shift right and area occupied by hard rotate are same, the proposed method saved area occupied by two adder blocks.

The rest of the paper is organised as follows. Section 2 gives the mathematical back ground of ECC, section 3 discuss the methodology and in section 4 results are tabulated. Finally section 5 concludes the paper.

2. ECC BACKGROUND

The elliptic curve arithmetic is defined over Galois field GF(p) where p is a prime number greater than 3. All arithmetic operations are modulo p. The elliptic curve equation E over GF(p) is given by :$y^2 = x^3 + ax + b$; where $p > 3$, $4a^3 + 27b^2 \neq 0$, and x, y, a,b $\in$ GF(p). There is also a single element named the point at infinity or the zero point denoted O, which serves as the additive identity. For any point P(x, y) $\in$E , we have P + O = P .

A Point addition and Point Doubling:

Additions in GF(p) are controlled by the following rules:

$$O = -O$$

$$P(x, y) + O = P(x, y)$$

$$P(x, y) + P(x, -y) = O$$

The addition of two different points on the elliptic curve is computed as shown below.

$$P(x_1 , y_1) + P(x_2 , y_2) = P(x_3 , y_3) \text{ ; where } x_1 \neq x_2$$

$$\lambda = (y_2 - y_1)/(x_2 - x_1)$$

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

The addition of a point to itself (point doubling) on the elliptic curve is computed as shown below

$$P(x_1 , y_1) + P(x_1 , y_1) = P(x_3 , y_3);$$

$$\lambda = (3(x_1)^2 + a) /(2y_1)$$

$$x_3 = \lambda^2 - 2x_1$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

B Point Multiplication:

Scalar multiplication Q=k.P is the result of adding point P to itself (k-1) times

Q = k.P = P + P + ……. + P.

(k-1 Times)

The binary method is the simplest and oldest efficient method for point multiplication. It is based on the binary expansion of k. The corresponding algorithm is shown below.

```
INPUT: A point P and an integer k
OUTPUT: Q = k.P
    1. Q←P
    2. For j = L− 2… 1, 0
        2.1 Q ← 2 Q
    2.2    IF $k_j$ = 1 THEN Q←Q + P
    3. RETURN Q
```

Algorithm 1: Binary scalar multiplication algorithm

3. METHODOLOGY

The reduction in area is obtained by modifying the architecture of Binary Inversion Algorithm (BIA). The two blocks of 'addition and division by 2' are replaced by two 'Rotate Right operations'. The x1= (x1+p)/2 operation of step 2.1.2 and x2= (x2+p)/2 operation of step 2.2.2 of Binary Inversion Algorithm are replaced by Rotate x1 right by 1 bit(RORx1) and Rotate x2 right by 1 bit(RORx2) operations. Since the area occupied by hard shift right and area occupied by hard rotate are same, the proposed method saved area occupied by two adder blocks This replacement is possible only if p is a Mersenne's prime. Algorithms of BIA and modified BIA are shown below.

```
    INPUT: Prime p and b  [1, p-1]
    OUTPUT: $b^{-1}$ mod p
    1.    u=b, v=p, x1=1, x2=0
    2.    while (u !=1 and v!=1 ) do
  while u is even do
            2.1.1 u = u/2
            2.1.2    if x1 is even then
                        x1= x1/2
                     else  x1= (x1+p)/2
              2.1.3    end while
         2.2  while v is even do
            2.2.1 v= v/2
              2.2.2    if x2 is even then
```

```
                    x2= x2/2
                 else x2 = (x2+p)/2
      2.2.3      end while
   2.3 if u≥ v then u=u-v, x1=x1-x2
       else v=v-u, x2=x2-x1
   2.4 end while
   2.5 if(u==1)return x1(modp)
       else return x2(modp)
```

Algorithm2: Binary Inversion Algorithm

```
INPUT: Prime p and b  [1, p-1]
OUTPUT: $b^{-1}$ mod p
1. u=b, v=p, x1=1, x2=0
2. while  (u !=1 and v!=1 ) do
   2.1while u is even do
       2.1.1     u = u/2
       2.1.2     if x1 is even then
                   x1= x1/2
                 else   x1= ROR x1
         2.1.3     end while
   2.2   while v is even do
         2.2.1     v= v/2
           2.2.2     if x2 is even then
                       x2= x2/2
                     else  x2 = ROR x2
      2.2.3      end while
   2.3 if u≥ v then u=u-v, x1=x1-x2
       else v=v-u, x2=x2-x1
   2.4 end while
     2.5 if(u==1)return x1(modp)
         else return x2(modp)
```

Algorithm3: Modified Binary Inversion Algorithm

Similarly, step2.2 of modified BIA can be implemented in hardware block, replacing u and x1 variables with v and x2 variables respectively. These two hardware blocks work in parallel. Figure 2 shows step2.1 of BIA. The delay due to this adder circuit is eliminated in modified BIA. Detailed architecture of BIA has been reported in [14]. Figure 1 and figure 2 shows the modified architecture and the architecture of the earlier designs which has an additional hardware circuit to perform the operation x1+p and x2 + p.

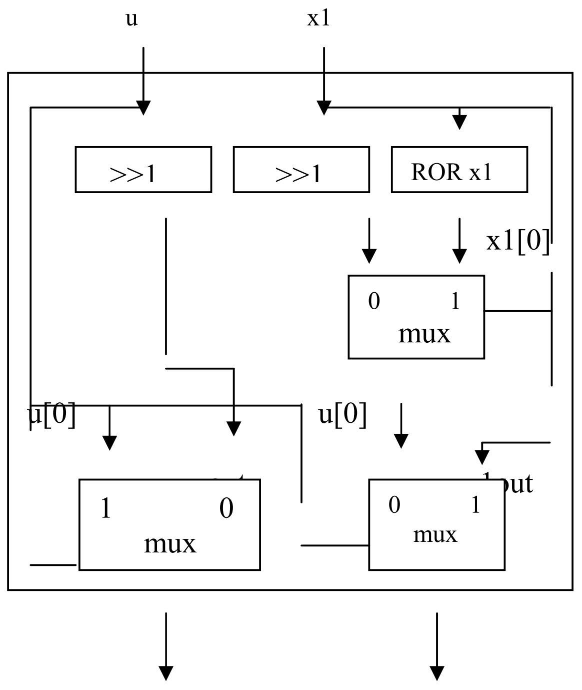

Fig.1 Architecture of the step 2.1 of modified BIA.

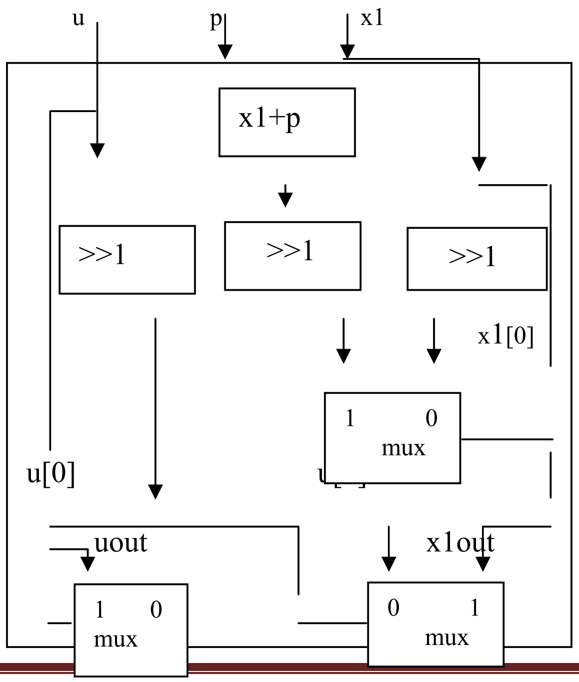

Fig.2 Architecture of the step 2.1 of BIA.

3. RESULTS AND DISCUSSION

In the proposed method, the 'addition of an odd number with a Mersenne's prime preceded by division by 2' has been replaced by rotate right operation. Some of the examples which justifies this is shown in table 1.

Table 1: Examples illustrating the replacement of (p+x)/2 by Rotate Right x by one bit (x is an odd number and $1 \leq x \leq p-1$)

Mersenne's Prime	Number x ($1 \leq x \leq p-1$)	(p+x)/2 Operation	Rotate Right x by one bit operation
31(2^5-1)	15 (01111)	23 (10111)	10111
127(2^7-1)	43 (0101011)	85 (1010101)	1010101
8191(2^{13}-1)	1023 (0001111 111111)	4607(10001111 11111)	1000111111111

With the above technique the two 521-bit adders are removed from the inversion block. The proposed method has assumed a conventional carry propagation binary adder and the proposed method has eliminated 2 x 521 full adders from the earlier designed architectures. The expression for sum and carry-out of conventional full adder is shown below.

sum=$a \oplus b \oplus c_{in}$

carry_out = $ab + (a\oplus b)c_{in.}$

The number of logic gates reduced in the architecture of inversion block is tabulated in Table2.

Table 2 : The Number of logic gates reduced in the modified architecture of inversion block

Logic Gate	EXOR	2-input AND	2-input OR
Number of gates reduced	4 x521=2084	4 x521=2084	2 x 521 =1042

5. CONCLUSION AND FUTURE SCOPE

A new technique of replacing addition of an odd number with a Mersenne's prime preceded by shift right (division by 2) with rotate right the number by one bit has been applied to the Architecture of Binary Inversion algorithm. Since shift right and rotate right consumed same area, we saved the area occupied by 521-bit adder. But the limitation of this architecture is that it is specific to NIST recommended prime field p521 and hence did not provide the flexibility.

Our future effort will be to develop new techniques to reduce the area of individual computational blocks of ECC processor and the integration of these blocks to achieve better area saving.

5. REFERENCES

1) C. Mclvor, M.McLoone and J.V.McCanny, "Modified Montgomery modular multiplication and RSA exponentiatuin techniques", IEE Proc. Comput.Digit.Tech., Voi.151,N9.6, November 2004
2) QIANG Liu, Fangzhen Ma, Dong Tong, Xu Cheng, "A regular Parallel RSA Processor", The 47th IEEE International Midwest Symposium on Circuits and Systems.
3) Jin Hua Hong, Cheng-Wen Wu, "Cellular-Array Modular Multiplier for fast RSA Public-Key Cryptosystem based on modified Booth's Algorithm", IEEE Transactions on VLSI systems, Vol.11, No.3, June 2003.
4) William N Chelton, Mohammed Benaissa, " Fast Elliptic Curve cryptography on FPGA", IEEE Transactions on VLSI systems, Vol.16, No2. February 2008.
5) William N Chelton, Mohammed Benaissa, "Design of Flexible GF(2^m Elliptic Curve Cryptography Processors", IEEE transactions of VLSI systems, Vol.14, No.6, June 2006.
6) Ray C.C. Cheung, Nicolas Jean-baptiste Telle, Wayne Luk, Peter Y.K. Cheung, " Customizable Elliptic Curve Cryptosystems", IEEE Transactions on VLSI systems, Vol.13, No.9, September 2005.
7) Alireza Hodjat, David D. Hwang, Ingrid Verbauwhede, "A scalable and high performance elliptic curve processor with resistance to timing attacks", ITCC'05.
8) Philip H. W. Leong, Ivan K.H.Leung, "A Microcoded Elliptic Curve Processsor using FPGA Technology",.IEEE Transactions on VLSI systems, Vol.10, No.5, October 2002.

9) Hamid Reza Ahmadi, Ali Afzali-Kusha, “ Low-power flexible GF(p) Elliptic curve cryptography processor”,

10) Ciaran J McIvor, Maire McLoone, John V.McCanny, “ Hardware Elliptic Curve Cryptographic Processor Over GF(p)”, IEEE Transactions on Circuits and Systems , Vol.53, No.9, , September 2006

11) Kendall Ananyi, Hamad Alrimeigh, Daler Rakhmatov, “ Flexible hardware processor for Elliptic curve cryptography”, IEEE transactions on VLSI systems, Vol.17, No.8, August 2009.

12) Jyu-Yuan Lai, Chih-Tsun Huang, “Elixir: High throughput cost effective dual field processors and the design framework for ECC”. IEEE Transactions on VLSI systems, Vol.16, No.11, October 2008

13) Kimmo Jarvinen, Jorma Skytta, “ On parallelization of High-speed processors for Elliptic curve cryptograpgy”, IEEE Transactions on VLSI systems, Vol.16, No.9, September 2008

14) Santhosh Ghosh, Monjur Alam, Indranil Sen Gupta, Dipanwita Roy Chowdhury, “ A robust GF(p) parallel arithmetic unit for public key cryptography”, 10th Euromicro Conference on Digital System Design Architectures, methods and tools(DSD 2007).