Montgomery Multiplier In Galois Field

Aakriti Kumari[1], Ishita Mishra[2], Manasa KV[3], Punarvasu R K[4], Vasudeva G[5]
[5]Assistant Professor
[1, 2, 3, 4, 5] AIT, Bengaluru

Abstract- *Finite Field arithmetic is becoming increasingly a very prominent solution for calculations in many applications. The most demanding Finite Field arithmetic operation is multiplication. In this paper two Finite Field multiplier architectures and VLSI implementations are proposed using the Montgomery Multiplication Algorithm. The first architecture (Folded) is optimized in order to minimize the silicon covered area (gate count) and the second (Pipelined) is optimized in order to reduce the multiplication time delay. Both architectures are measured in terms of gate count-chip covered area and multiplication time delay and have more than adequate results in comparison with other known multipliers.*

Keywords- Montgomery multiplier, ECC, Galois field.

I. INTRODUCTION

In mathematics, a finite field or Galois field, so-named in honor of Évariste Galois, is a field that contains a finite number of elements. As with any field, a finite field is a set on which the operations of multiplication, addition, subtraction and division are defined and satisfy certain basic rules. The most common examples of finite fields are given by the integers of *p* when *p* is a prime number.

Elliptic Curve Cryptography, or ECC, is a powerful approach to cryptography and an alternative method from the well-known RSA. It is an approach used for public key encryption by utilizing the mathematics behind elliptic curves in order to generate security between key pairs.

II. LITERATURE SURVEY

1. Low complexity Montgomery multiplication architecture for Elliptical curve cryptography over GF(p^m)

In this paper**,** Somsubhra Talapatra proposed a scalable VLSI multiplication architecture based on Montgomery multiplication (MM)algorithm for elliptic curve cryptography (ECC) over GF (*pm*), where *p* is a positive prime and *m* is the degree of extension of the base field GF(*p*), is presented. The elements of the GF (*pm*) are in polynomial basis (PB) representation. The coefficients of the polynomials are represented in Montgomery residue format to simplify the multiplications over GF(*p*).

2. Hardware implementation of Montgomery multiplier in a Systolic array

In this paper, Sıddıka Berna describes a hardware architecture for modular multiplication operation which is efficient for bit-lengths suitable for both commonly used types of Public Key cryptography (PKC) i.e. ECC and RSA Cryptosystems. The challenge of current PKC implementations is to deal with long numbers (160-2048 bits) in order to achieve system’s efficiency, as well as security.

3. Montgomery's Multiplication Technique: How to Make it Smaller and Faster

Colin D Walter states that, Montgomery's modular multiplication algorithm has enabled considerable progress to be made in the speeding up of RSA cryptosystems. Perhaps the systolic array implementation stands out most in the history of its success. This article gives a brief history of its implementation in hardware, taking a broad view of the many aspects which need to be considered in chip design.

4. Precise Bounds for Montgomery Modular Multiplication and Some Potentially Insecure RSA Moduli

Colin D. Walter states that, An optimal upper bound for the number of iterations and precise bounds for the output are established for the version of Montgomery Modular Multiplication from which conditional statements have been eliminated. The removal of such statements is done to avoid timing attacks on embedded cryptosystems but it can mean greater execution time. Unfortunately, this inefficiency is close to its maximal for standard RSA key lengths such as 512 or 1024 bits. Certain such keys are then potentially subject to attack using differential power analysis. These keys are identified, but they are rare and the danger is minimal. The improved bounds, however, lead to consequent savings in hardware.

5. Efficient Implementation of an Elliptic Curve Cryptosystem Over Binary Galois Fields in normal and Polynomial Bases

Matthew Estes and Philip Hines, states that, Elliptic Curve Cryptography has many features
that distinguish it from other cryptosystems, one of which is that it is still relatively new cryptosystem. As such, many improvements in performance have been discovered during the last few years for Galois Field operations both in Polynomial Basis and in Normal Basis. However, there is still some confusion to the relative performance of these new algorithms and very little examples of practical implementations of these new algorithms. The purpose of this project is to implement these high-performance algorithms in a library, test their relative performance, and provide a flexible framework for integrating
future algorithms.

Finite field or Galois field

It is particularly useful in translating computer data as they are represented in binary forms. That is, computer data consist of combination of two numbers, 0 and 1, which are the components in Galois field whose number of elements is two. Representing data as a vector in a Galois Field allows mathematical operations to scramble data easily and effectively.

Expression for Galois Field

The elements of Galois Field Gf (p^n) is defined as,

Gf $(p^n) = (0, 1, , , , , , , , , , , , , , ,p - 1)$ U
$(p, p + 1, p + 2, , , , , , , , , , , , , , , p + p - 1)$ U
$(^{p2}, ^{p2} + 1, p^2 + 2, , , , , , , , , , , , ,p^2 + p - 1)$ U , , , , , U
$(p^{(n-1)}, p^{(n-1)}+1 ,\ p^{(n-1)} + 2, , , , , , p(n-1) + p - 1)$

Where, p belongs to P and n belongs Z.

The order of the field is given by p^n while p is called the characteristic of the field. On the other hand, gf, as one may have guessed it, stands for Galois Field. Also note that the degree of polynomial of each element is at most n - 1.

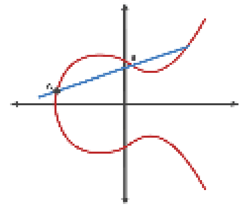
Elliptic Curve Cryptography

ECC has been slowly gaining in popularity over the past few years due to its ability to provide the same level of security as RSA with a much smaller key size. The resources available to crack encrypted keys continues to expand, meaning the size of encrypted keys must continue to grow in order to remain secure. This can prove to be a burden to certain devices, particularly mobile, that do not have as much available computational power. However, Elliptic Curve Cryptography helps to solve that problem.

Working

An elliptical curve can simply illustrate as a set of points defined by the following equation:

$$\mathbf{y^2 = x^3 + ax + b}$$



Based on the values given to *a* and *b*, this will determine the shape of the curve. Elliptical curve cryptography uses these curves over finite fields to create a secret that only the private key holder is able to unlock. The larger the key size, the larger the curve, and the harder the problem is to solve.

Why is ECC Important?

As noted in the previous section, size is a major factor in the importance of elliptic curve cryptography. For keys of the same size, solving for an elliptic curve discrete logarithm is significantly harder than factoring, which is how RSA encrypts keys.

To put things into perspective, according a Universal Security study, breaking a 228-bit RSA key would take less energy than what is needed to boil a teaspoon of water. Alternatively, breaking a 228-bit ECC key would require more energy than it would take to boil all the water on earth.

Therefore, having the ability to significantly reduce the size of these keys can serve very useful for devices which have less computational power.

III. MONTGOMERY'S MODULAR MULTIPLICATION

Digit Multipliers

Before Montgomery Multiplication digit multipliers were used. Since digit multipliers are complex, we will use the radix multiplier. These rxr multipliers form the core of an RSA processor, forming the digit-by-digit products. The cross-over point is greater than the size of the digits here. So classical multiplication methods are preferable.

Speed is most easily obtained by using at least n multipliers to perform a full-length multiplication ai×B (or equivalent) in one clock cycle. In this realistic measure of the speed required for real-time decryption is provided by an assumption that the internal bus speed is in the order of one k-bit digit per clock cycle. If the k-bit multiplier operates in one cycle with no internal pipelining then computing A×B takes n cycles using n multipliers in parallel in order to compute ai×B in one cycle. The throughput is therefore one digit per cycle for a multiplication.

Modular Reduction

Modular arithmetic has the problem that a division is usually needed in order to get the remainder. Division is a complex, time consuming operation in GF(2k) fields. Thus, methods for bypassing the division obstacle have been devised. So, to overcome this we go for Montgomery Multiplication Algorithm.

The Montgomery Multiplication Algorithm for GF(2k) fields

One of the most popular algorithms, taken from standard arithmetic, is the Montgomery Multiplication Algorithm. It performs modular multiplication without division.
Instead of a(x) b(x) mod f(x) the algorithm calculates a(x) b(x) r-1(x) mod f(x) where r(x) is a precomputed value. It is required that gcd (r(x), f(x)) = 1. By choosing r(x) = x k, for the gcd(r(x), f(x)) = 1 assumption to hold, it suffices f(x) not be divisible by x, which is always the case since f(x) is defined over the field GF (2). Through the correct choosing of the value r(x), the algorithm becomes less complex and can give efficient hardware implementations.

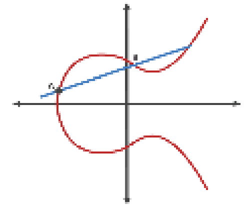
Elliptic Curve Cryptography

ECC has been slowly gaining in popularity over the past few years due to its ability to provide the same level of security as RSA with a much smaller key size. The resources available to crack encrypted keys continues to expand, meaning the size of encrypted keys must continue to grow in order to remain secure. This can prove to be a burden to certain devices, particularly mobile, that do not have as much available computational power. However, Elliptic Curve Cryptography helps to solve that problem.

Working

An elliptical curve can simply illustrate as a set of points defined by the following equation:

$$\mathbf{y^2 = x^3 + ax + b}$$



Based on the values given to *a* and *b*, this will determine the shape of the curve. Elliptical curve cryptography uses these curves over finite fields to create a secret that only the private key holder is able to unlock. The larger the key size, the larger the curve, and the harder the problem is to solve.

Why is ECC Important?

As noted in the previous section, size is a major factor in the importance of elliptic curve cryptography. For keys of the same size, solving for an elliptic curve discrete logarithm is significantly harder than factoring, which is how RSA encrypts keys.

To put things into perspective, according a Universal Security study, breaking a 228-bit RSA key would take less energy than what is needed to boil a teaspoon of water. Alternatively, breaking a 228-bit ECC key would require more energy than it would take to boil all the water on earth.
Therefore, having the ability to significantly reduce the size of these keys can serve very useful for devices which have less computational power.

REFRENCES

[1] Hardware Implementation of a Montgomery Modular Multiplier in a Systolic Array-
Sıddıka Berna Ors, LejlaBatina, Bart Prenee, JoosVandewalle Katholieke University Leuven, ESAT/SCD-COSIC Kasteelpark Arenberg 10 B-3001 Leuven-Heverlee, Belgium.

[2] Low Complexity Montgomery Multiplication Architecture for Elliptic Curve Cryptography overGF(*pm*)-
Somsubhra Talapatra1, and Hafizur Rahaman, Department of Information Technology, Bengal Engineering and Science University (BESU), Shibpur, Howrah-711103, West Bengal, India.

[3] R. Lidl, and H. Niederreiter, “Introduction to Finite Fields and Their Applications”, Cambridge Univ. Press,1994.

[4] P. L. Montgomery, “Modular Multiplication without Trial Division”,
Math. Computation, vol. 44, pp:519-521, 1985

[5] C. K. Koc and T. Acar, “Montgomery Multiplication in GF(2*k*),” *Proc. of 3rd Annu. Work. on Selected Areas in*

Cryptography, Queen's University, Kingston, Ontario, Canada, pp:95-106, Aug. 1996.
[6] A. F. Tenca, and C. K. Koc, "A Scalable Architecture for Modular Multiplication Based on Montgomery's Algorithm," *IEEE Trans. On Computers*, vol. 52, no. 9, pp:1215-1220, Sep. 2003.
[7] R.V. Kamala, M. Sudhakar and M.B. Srinivas, "An Efficient Reconfigurable Montgomery Multiplier Architecture for GF(*n*)," *9th EUROMICRO Conf. Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006*. pp:155 – 159, 2006.
[8] Jan M. Rabaey, Anantha Chandrakasan, and Borivoje Nikoli□, "Digital Integrated Circuits: A Design Perspective," Prentice Hall (India), 2nd ed., New Delhi, 2006.
[9] Jorge Guajardo Merchan, "Arithmetic Architectures for Finite Fields GF (*pm*) with Cryptographic Applications," *PhD Thesis*, Ruhr-Universität Bochum, Germany, 2004.
[10] Huapeng Wu, Anawar Hasan, and Ian F. Blake, "Low Complexity Parallel Multiplier in *Fq n* Over *Fq*," *IEEE Trans. Circ. & Syst.-I*. vol. 49, no. 7, pp: 1009-1013, Jul. 2002.