Compact and Scalable Elliptic Curve Crypto Processor in Binary Field

A.Suhirtha Subha1, Rahila Bilal2
1PG Scholar, Thanthai Periyar Government Institute of Technology,
Vellore, Tamil Nadu, India
suhirthasubha@gmail.com
2Associate Professor of ECE, Thanthai Periyar Government Institute of Technology,
Vellore, Tamil Nadu, India
bilalrahila@yahoo.co.in

Abstract

Elliptic Curve Cryptography plays a vital role in securing the information for the past two decades. Implementing ECC in hardware is more efficient than software. A processor is said to be compact, if its implementation space is small compared to its execution speed and is scalable if this processor is used with different key sizes. Different metrics such as execution time, implementation space and energy consumption are used to quantitatively measure the performance of ECC processor. This ECC processor consumes 87 %, 92 % of the I/O pins from Quartus 10.1 Arria GX namely EP1AGX50DF1152C6, Cyclone III namely EP3C40F780C6 respectively. This also consumes the power of 671.12 mW in 44 ms and 150.39 mW in 40 ms from Quartus 10.1 Arria GX namely EP1AGX50DF1152C6, Cyclone III namely EP3C40F780C6 respectively.

Introduction

Information security is of the greatest importance in a world in which communication over open networks and storage of data in digital form play a key role in daily life. The science of cryptography provides efficient tools to secure information. Elliptic curve cryptography (ECC) has drawn more and more attention due to the fact that its selected key length can be smaller than that in RSA cryptosystems for the same level of security. Literature has shown that ECC is the most widely used public-key cryptosystem, which allows much useful functionality such as digital signature, public-key encryption, and key agreements.

Elliptic curve cryptography (ECC) was independently proposed in 1985 by Neal Koblitz and Victor Miller. ECC is indeed an attractive solution as the public-key scheme. However, the computation involved in the scalar multiplication for an elliptic curve is time consuming and more complex than that in RSA. Due to its many advantages, ECC has been adopted by many standards, such as National Institute of Standards and Technology NIST and Standards for Efficient Cryptography Group SECG.

In many applications ,a software implementation of ECC might be appropriate, but in some cases better performances are required and consequently hardware implementations should be used instead. As the popularity of ECC increases, the need for efficient hardware solutions that accelerate the computation of elliptic curve point multiplications also increases.

Background

Elliptical curve cryptography (ECC) is a public key encryption technique based on elliptic curve theory that can be used to create faster, smaller, and more efficient cryptographic keys. ECC generates keys through the properties of the elliptic curve equation instead of the traditional method of generation as the product of very large prime numbers. The technology can be used in conjunction with most public key encryption methods, such as RSA and Diffie-Hellman. According to some researchers, ECC can yield a level of security with a 164-bit key that other systems require a 1,024-bit key to achieve. Because ECC helps to establish equivalent security with lower computing power and battery resource usage, hence it is becoming widely used for mobile applications.

An elliptic curve is not an ellipse (oval shape), but is represented as a looping line intersecting two axes. ECC is based on properties of a particular type of equation created from the mathematical group (a set of values for which operations can be performed on any two members of the group to produce a third member) derived from points where the line intersects the axes. Multiplying a point on the curve by a number will produce another point on the curve, but it is very difficult to find what number was used, even if

you know the original point and the result. Equations based on elliptic curves have a characteristic that is very valuable for cryptography purposes. They are relatively easy to perform, and extremely difficult to reverse. The industry still has some reservations about the use of elliptic curves.

Galois Field

Galois field arithmetic plays a critical role in elliptic curve cryptography implementation because it's the core of ECC scalar multiplication. Finite Field or Galois Field is a set of finite number of elements, denoted as *GF(q)*. Every element in $GF(2^n)$ can be represented as a polynomial $A(x) = a_n x_{n-1} + \ldots + a_0$ with coefficients a_i €{0,1}.So, more efficient implementation of underlying field operations results more efficient in the overall algorithm. Galois fields suitable for ECC implementation divides into two categories as prime field and binary field. Binary Galois field is preferred in hardware because of free carry propagation property in hardware which make addition operation only done with one n-bit XOR operation (equal to bit wise addition modular 2).

Point Multiplication on Elliptic Curves

The Elliptic Curve Crypto Processor computes elliptic curve point multiplications for arbitrary curves defined over GF(p). An scalar multiplication *kP* is the result of adding the point $P \in E(K)$ to itself $k-1$ times.

$$kP = P + P + P + \ldots + P \qquad (1)$$

($k-1$ sums)

By scalar multiplication or point multiplication we perform the combination of additions and doublings of points to compute kP for given k and P. There are several methods like the additive variant of repeated squaring or addition-subtraction chains which do this task using O(logm) doublings and additions .Point multiplication is achieved by two basic elliptic curve operations such as

- Point addition, adding two points P and Q to obtain another point,
  i.e., P + Q
- Point doubling, adding a point P to itself to obtain another point R=2P.

Point Addition

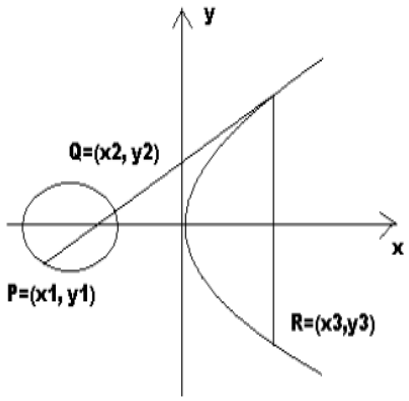

Figure 1 : Point Addition

Point addition is defined geometrically by the "chord-tangent" law of composition. To add two distinct points P and Q on an elliptic curve a chord is drawn between then and the third point of intersection of this line with the curve is reflected through the x axis.

Point Doubling

Adding a point P to itself (doubling P) is performed in a similar manner. In this case the tangent to the curve at P is taken. This line intersects the curve at exactly one other point

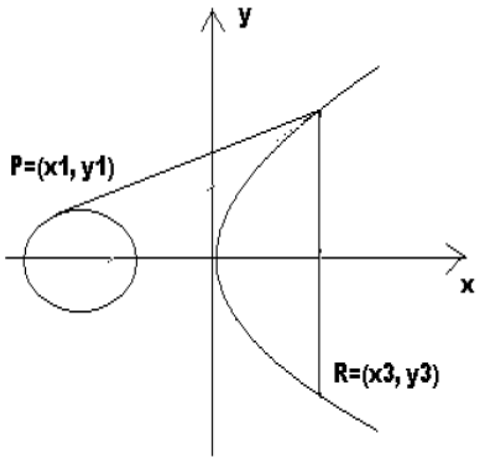

Figure 2 : Point Doubling

which is then reflected through the x axis. This reflection is P + P = 2P. Point scalar multiplication is performed by successive doublings and additions of the base point P.

Algorithm

The main operation in ECC is the elliptic curve point multiplication (ECPM). One of the ways to compute ECPM is proposed by Lopez and Dahab. The main advantage of this algorithm is that it does not have any extra storage requirements; the same operations are performed in every-iteration of the main loop, thereby potentially increasing resistance of timing attacks and power analysis attacks .The Lopez-Dahab Algorithm is given as follows ,

Require: k = $(k_{t-1}, k_1, k_0)$with $k_{t-1} = 1$, P(x, y), b-curve specific coefficient

Ensure: Q(x_0, y_0) = kP

$(X_1, Z_1) \leftarrow (x, 1)$,

$(X_2, Z_2) \leftarrow (x^4 + b, x^2)$

if $k_{t-2} = 1$ then

Swap(X_1, X_2), Swap(Z_1, Z_2)

end if

for i from t − 2 down to 0 do

// Madd

$X_2 \leftarrow X_1Z_2X_2Z_1 + x(X_1Z_2 + X_2Z_1)^2$

$Z_2 \leftarrow (X_1Z_2 + X_2Z_1)^2$

// Mdouble

$X_1 \leftarrow X_1^4 + bZ_1^4$

$Z_1 \leftarrow X_1^2 Z_1^2$

if $(i \neq 0 and k_i \neq k_{i-1})$ or $(i = 0 and k_i = 1)$ then

Swap(X_1, X_2), Swap(Z_1, Z_2)

end if

end for

// Mxy

$x_0 \leftarrow X1/Z1$

$y_0 \leftarrow (xZ_1+X_1)(Z_1Z_2(x^2+y)+(xZ_1+X_1)(xZ_2+X_2))+xyZ_1^2 Z_2/(xZ_1^2Z_2)$

return Q(x_0, y_0)

Architecture of ECC Processor

The functional units of the ECC processor are Modular Multiplier, a Squarer unit, Point Addition Unit, Point Multiplication unit, RAM and Control units. This design utilizes carry-free adders (carry-save adders) as the fundamental arithmetic unit in all field operations. Carry-free adders that can be used to implement all field operations are very fast since carries do not propagate. Finite field arithmetic unit based on carry-free adders were utilized in different cryptographic applications

Figure 3 shows the implementation of the finite field arithmetic unit by carry-free adder units.

Family	Arria GX	Cyclone IV E	Cyclone III	Cyclone II
Device	EP1AGX90EF1152C6	EP4CE30F29C6	EP3C40F780C6	EP2C70F896C6
Thermal Power Dissipation	843.20 mW	199.45 mW	150.39 mW	251.70 mW
Total I/O Resource used	81 %	92 %	84 %	79 %
Total CPU time	00:00:55	00:00:43	00:00:40	00:00:41

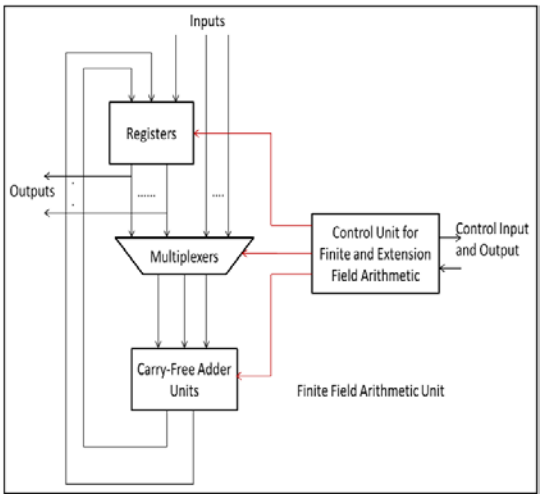

Figure 3 : Finite Field Arithmetic Unit

.

Finite field (FF) operations are the fundamental building blocks in implementing ECC operations such as FF addition ,FF squaring , FF multiplication and FF inversion . Among these

operations, FFADD can be implemented using a bit-wise exclusive-OR (XOR) operation.

Experimental and Synthesis Result

The proposed scalable ECP has been implemented using the Quartus 10.1 software. The target FPGAs selected were Arria GX EP1AGX90EF1152C6, Cyclone II EP2C70F896C6, Cyclone III EP3C40F780C6, Cyclone IV EP4CE30F29C6, in order to compare to other designs in the current literature. These devices were chosen instead from the family of devices because there are not
enough resources to fully implement this proposed design. However, the FPGAs are in the same family, so they can still be compared. The implementation results of the proposed design are shown in Table I along with performance of other scalable ECP designs in the current literature.

Table I : Performance Analysis for 163 bit

Elliptic Curve Point Multiplication

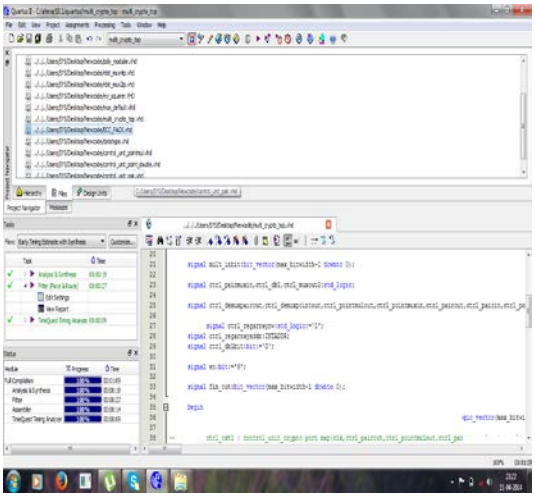

Figure 4 : Synthesis Result for 163 bit Elliptic

Curve Point Multiplication

Conclusion and Future Work

Hardware implementation of Elliptic Curve Cryptography encryption engine has been shown in this paper. The system is designed using VHDL and synthesized using Quartus. For 163 bits key length, the system consumes around 80% of the device resources and performs the point multiplication operation in 40□□s. This is much faster than the software implementation, where about 120 ms is required for the same operation. In summary, it is shown that elliptic curve cryptosystem can be efficiently implemented on a commercial FPGA, resulting in very flexible implementation with increased speed performance over the software solution.

References

[1]. "A scalable hardware/software co-design for elliptic curve cryptography on PicoBlaze microcontroller," in Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS), June 2010, pp. 2111–2114.

[2]. "Flexible Hardware/Software Co-design for Scalable Elliptic Curve Cryptography for Low-Resource Applications," in 21st IEEE International Conference on Application-specific Systems Architectures and Processors (ASAP), July 2010, pp. 285–288.

[3]. Ansari and M. Hasan, "High-Performance Architecture on Elliptic Curve Scalar Multiplication," IEEE Transactions on Computers, vol. 57, no. 11, pp. 1443–1453, November 2008.

[4]. Hankerson, A. Menezes, and S. Vanstone, Guide to Elliptic Curve Cryptography. New York, NY, USA: Springer-Verlag, 2004.

[5]. J. Lopez and R. Dahab, "Fast multiplication on elliptic curves over GF(2^m) without precomputation," in CHES99: Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems. Springer-Verlag, 1999, pp. 316–327.

[6]. Karatsuba and Y. Ofman, "Multiplication of multi-digit numbers on automata," Soviet Physics Doklady, vol. 7, pp. 595–596, 1963.

[7]. M. Benaissa and W. M. Lim, "Design of flexible GF(2^m) elliptic curve cryptography processors," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 14, no. 6, pp. 659–662, 2006.

[8]. M. Hassan and M. Benaissa, “Low Area - Scalable Hardware/Software Co-design for Elliptic Curve Cryptography,” in 3rd International Conference on New Technologies, Mobility and Security (NTMS), December 2009, pp. 1–5.

[9]. M. Morales-Sandoval, C. Feregrino-Uribe, R. Complido, and I. Algredo-Badillo, “A reconfigurable GF(2^m) elliptic curve cryptographic coprocessor,” in 2011 VII Southern Conference on Programmable Logic (SPL), April 2011, pp. 209–214.

[10]. N. I. of Standards and Technology, “Recommended Elliptic Curves for Federal Government Use,” July 1999.

[11]. N. Koblitz, “Elliptic curve cryptosystems,” Mathematics of Computation, vol. 48, no. 177, pp. 203–209, 1987.

[12]. P. G. Comba, “Exponentiation cryptosystems on the IBM PC,” IBM Systems Journal, vol. 29, no. 4, pp. 526–538, 1990.

[13]. R. Rivest, A. Shamir, and L. Adleman, “A method for obtaining Digital Signatures and Public-Key Cryptosystems,” Communications of the ACM, vol. 21, no. 2, pp. 120–126, February 1978.

[14]. S. for Efficient Cryptography, “SEC 2: Recommended Elliptic Curve Domain Parameters,” July 2000.

[15]. T. Itoh and S. Tsujii, “A Fast Algorithm for Computing Multiplicative Inverses in GF(2^m) Using Normal Bases,” Information and Computation, vol. 78, no. 3, pp. 171–177, 1988.

[16]. V. Miller, “Use of elliptic curves in cryptography,” in CRYPTO85: Proceedings of the Advances in Cryptology. Springer–Verlag, 1986, pp. 417–426.