VLSI IMPLEMENTATION FOR ELLIPTIC CURVE CRYPTOGRAPHY OVER BINARY EXTENSION FIELD

[1] Kathi Kondamma, [2]D.Kiran Kumar, [3]R L B R Prasad Reddy
[1]PG Scholar, Dept. Of ECE, Srinivasa Institute of Technology and Science, Kadapa – 516001.
[2]Assistant Professor, Dept. Of ECE, Srinivasa Institute of Technology and Science, Kadapa – 516001.
[3]Associate Professor, Dept. Of ECE, Srinivasa Institute of Technology and Science, Kadapa – 516001.
[1]kathielizabeth1996@gmail.com, [2]kiran.doraboyina@gmail.com, [3]rajendra.409@gmail.com

Abstract

Elliptic curve cryptography plays a crucial role in network and communication security. However, implementation of elliptic curve cryptography, especially the implementation of scalar multiplication on an elliptic curve, faces multiple challenges. One of the main challenges is side channel attacks (SCAs). SCAs pose a real threat to the conventional implementations of scalar multiplication such as binary methods (also called doubling-and-add methods). Several scalar multiplication algorithms with countermeasures against side channel attacks have been proposed. Among them, Montgomery Powering Ladder (MPL) has been shown an effective countermeasure against simple power analysis. However, MPL is still vulnerable to certain more sophisticated side channel attacks. A recently proposed modified MPL utilizes a combination of sequence masking (SM), exponent splitting (ES) and point randomization (PR). And it has shown to be one of the best countermeasure algorithms that are immune to many sophisticated side channel attacks. In this paper, an efficient hardware architecture for this algorithm is proposed.

Keywords: side channel attacks (SCAs), sequence masking (SM), exponent splitting (ES), point randomization (PR), cryptography.

I. INTRODUCTION

The Internet is increasingly important to the people all over the world who use it for personal and business purposes. While the internet brings much convenience to people, there still exist security risks and vulnerabilities in using the internet. For example, various cyber-attacks, including side channel attacks, pose a great danger for the Internet users. Network security, which provides physical and software countermeasures to protect the network from unauthorized access and attacks, becomes a very active research area and industry.

Cryptography plays a critical role in providing essential and unique network security services to the internet. There are two main families of cryptography from the point of view of key generation, symmetric-key cryptography and asymmetric-key cryptography. In symmetric key cryptography system, there is only one key used both for encryption and decryption. This system requires that both parties involved in the communication share one secret key, which has to be pre-arranged in advance in a procedure called key establishment.

This is regarded as a main drawback of symmetric-key cryptography system since it cannot resolve the issue of key establishment without resorting to a third party. Unlike symmetric-key cryptography system, the asymmetric-key cryptography system (more popularly known as public-key system) uses two keys, one for encryption and the other for decryption. The key used for encryption is the public key, which is accessible to the public and can be distributed widely and easily.

The other one used for decryption is the private key, which must be kept secret and is only known to the owner of the cryptosystem. By differentiating the encryption key and decryption key, the asymmetric-cryptography system can provide very important and unique security services such like key exchange and digital signature. A drawback of asymmetric-cryptography systems is that they have higher computational complexity, compared to symmetrical key systems. Since Diffie and Hellman proposed the Diffie-Hellman key exchange scheme as the first asymmetric-cryptography system in 1976 [3], several asymmetric-key cryptography systems have been presented, such like RSA, El Gamel, and Elliptic curve cryptography. All these algorithms are based on some different hard mathematical problems.

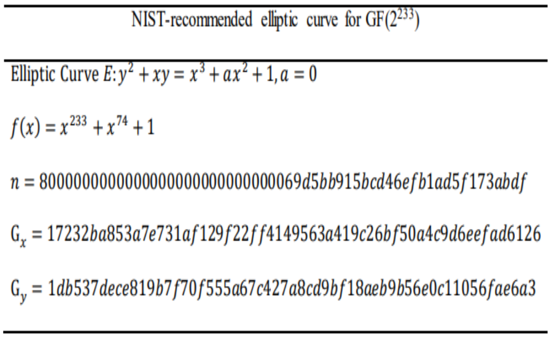
II. EXISTING METHOD

In existing method, they introduce a modified MPL with sequence masking, exponent splitting and point randomization proposed in [11]. A small modification has been done to this algorithm to make it suitable for ECC scalar multiplication since it is originally invented for exponentiation operation. It has been stated that unprotected MPL is still vulnerable to a lot of side channel attacks. To offer protections in algorithm level, He, Huang and Wu proposed a highly secure MPL for exponentiation operation [11]. Several countermeasures had been applied to enhance its security strength.

III. DESIGN METHODLOGY

An efficient hardware architecture is proposed and its FPGA implementation is presented. Very high speed integrated circuit (VHSIC) Hardware Description Language is chosen as the target implementation language. The ECC parameters are NIST - recommended elliptic curve for $GF(2^{233})$ in [39], as shown in Table - 1, where $f(x)$ is the irreducible polynomial, n is the order, G_x and G_y are base point coordinates.

Table – 1: NIST-recommended parameters



NIST-recommended elliptic curve for GF(2^{233})
Elliptic Curve $E{:}\,y^2+xy=x^3+ax^2+1, a=0$
$f(x)=x^{233}+x^{74}+1$
$n = 8000000000000000000000000000069d5bb915bcd46efb1ad5f173abdf$
$G_x = 17232ba853a7e731af129f22ff4149563a419c26bf50a4c9d6eefad6126$
$G_y = 1db537dece819b7f70f555a67c427a8cd9bf18aeb9b56e0c11056fae6a3$

a. Implementation Hierarchy of the ECC operations

The building blocks of computation involved in ECC is illustrated in Fig. Finite field arithmetic such as field addition, subtraction, multiplication, inversion and squaring are the fundamental computations. Both elliptic curve point addition and doubling are based on the finite field computations. As the figure shown, the upper layer computations are constructed by the lower layers. Scalar multiplication is realized by different algorithms based on point addition and doubling. Elliptic curve cryptographic schemes such like ECDSA are on the top.

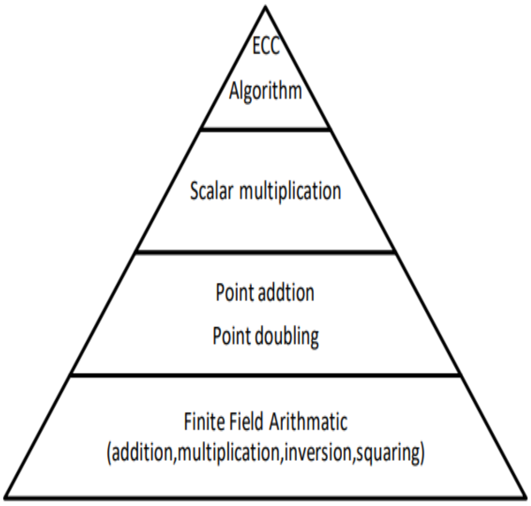


Figure - 1: Hierarchical architecture for computation involved in ECC

b. Random Number Generation

In the pre-computation part, there three random binary sequence need to be generated. This process is implemented with a linear feedback shift register (LFSR). A LFSR is s sequential shift register with combinational logic that causes it to pseudo randomly cycle through a sequence of binary values. It has well-known applications in generating pseudo-random binary sequence. A pseudo-random binary sequence is considered pseudo because it will start to repeat the pattern after a certain number of states. In order to make the generation closer to a real random number, the LFSR need to reach its maximum length. In other words, an n bit LFSR need to generate all 2n -1 states before it starts to repeat itself.

By carefully chosen the positions of the bits feeding back to the next state, a maximum length LFSR can be achieved. For the case of the 233 bit random sequence, the tap value is 233 and 159 [40]. There are two structures of the LFSR. One is one-to-many structure (also known as Galois LFSR). The other is many-to-one structure (also known as Fibonacci LFSR). As Figure, illustrated, a 233-bit Galois LFSR is built. This structure is chosen rather the many-to-one structure, is because that Galois LFSR generates all the feedback bits parallel. In this way, the LFSR runs more efficiently.

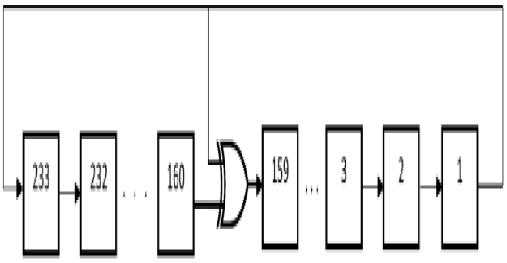


Figure: 233-bit LFSR

The LFSR will generate three random sequences. The first random number will continue generating until it is smaller than order of base point P, then it is assigned to k0 according to Algorithm. The second random sequence will generate and then assigned to s, acting as the switch to determine which scalar multiplication is going to perform. The third random sequence will be r, it will be used in the precomputation to compute rP. Since 49 r is random, rP will be random point. The computation will need the modules introduced in later sections.

c. Scalar Multiplication

As the main computation of the ECC, Algorithm provides a whole new MPL with strong resistance to SCAs. During the pre-computation process, the first scalar random sequence k_0 is generated. The second scalar k_1 is computed by $k{-}k_0$. Another random sequence is produced and finally the random point $R = rP$ is acting as the mask.

There four registers R_0, R_1, R_2 and R_3 holding 4 different intermediate values. They are initialized by the x, y coordinates of the base point P and mask R. R_0 and R_1 are given the value R and $P + R$ at the beginning. R_2 and R_3 are initialized with $-R$ and $P - R$. The anti-mask $-R$ shares the same x coordinate with R. While the y coordinate is calculated simply x-or the coordinates of mask R. Three shift registers to store the value of s, k_0 , and k_1 . The control unit is the core unit to realize the algorithm. The input and the output of the point addition and point doubling module is controlled by this unit. The control unit has 36 different states to decide which value from which register to be given as the input of point operation module, also at the same time, the specific register to store the output is decided. The states are controlled by the three random values in the shift register. Additionally, a counter aiming for m cycles is built in as to tell the whole process when to stop the computation. Fig. illustrates the blocking diagram of the top main computation module.

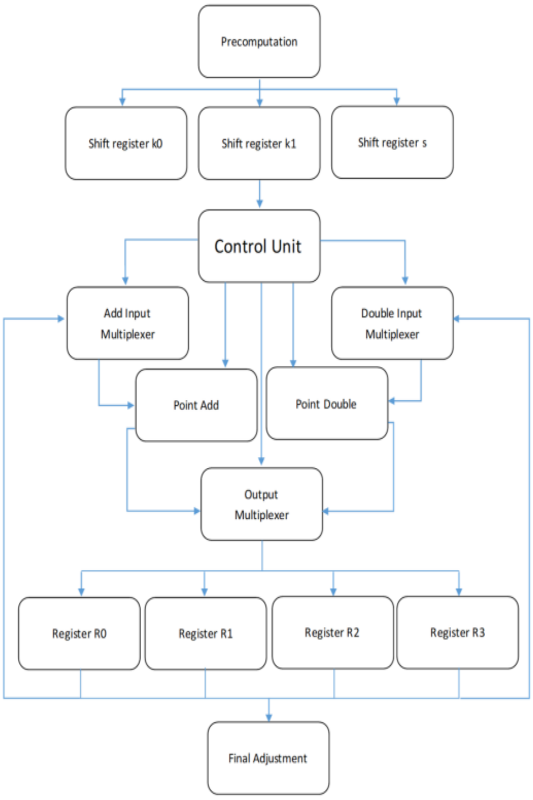
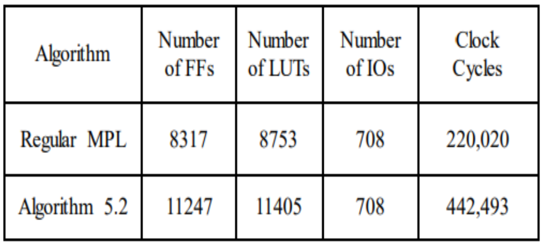


Figure: Architecture of proposed implementation

IV. SIMULATION RESULTS

The VHDL code is synthesized for using Xilinx Vivado 2017. The hardware resource usage is summarized in Table. The computation time at 100 MHz is 4.43ms. We can see Algorithm doubles the computation time needed since it consists of two scalar multiplications. Fig shows the result of the output waveform.

Table: Hardware usage of different algorithms implemented



Algorithm	Number of FFs	Number of LUTs	Number of IOs	Clock Cycles
Regular MPL	8317	8753	708	220,020
Algorithm 5.2	11247	11405	708	442,493

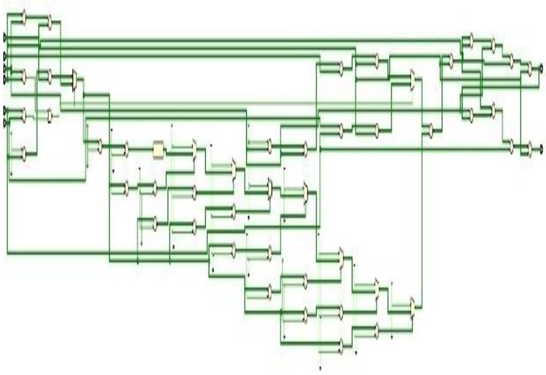
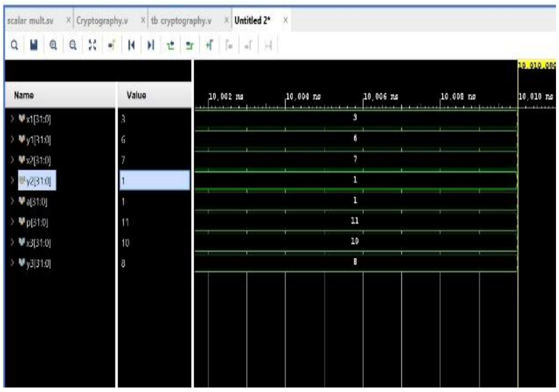


Figure: Output waveform



Name	Value
x1[31:0]	3
y1[31:0]	6
x2[31:0]	7
y2[31:0]	1
a[31:0]	1
p[31:0]	11
x3[31:0]	10
y3[31:0]	8

Figure: Existing simple ECC cryptography

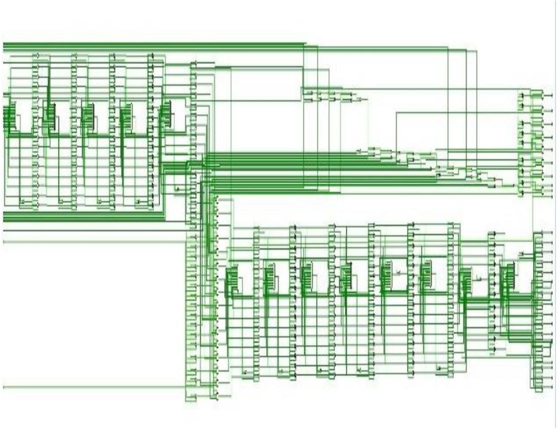
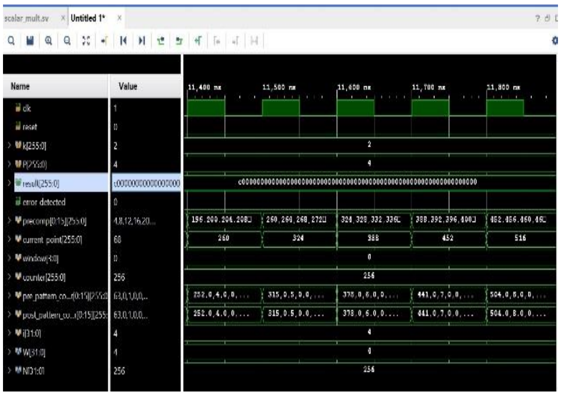


Figure: RTL Schematic for Scalar multiplication with detection



Name	Value
clk	1
reset	0
k[255:0]	2
P[255:0]	4
result[255:0]	c000000000000000
error detected	0
precomp[0:15][255:0]	4,8,12,16,20...
current point[255:0]	68
window[3:0]	0
counter[255:0]	256
pre_pattern_co...r[0:15][255:0]	63,0,1,0,0,...
post_pattern_co...r[0:15][255:	63,0,1,0,0...
i[31:0]	4
W[31:0]	4
N[31:0]	256

Figure: RTL Schematic for Scalar multiplication with detection

CONCLUSION

In this thesis, an efficient architecture for the scalar multiplication algorithm [11] is proposed. A FPGA implementation of the algorithm [11] is presented. It is the first time that this algorithm is implemented in hardware. This implementation resistant to most existing side channel attacks such as doubling attack [8], relative doubling attack [30], comparative power analysis [31], m-safe error attack [19], c-safe error attack [9], high-order [36] and template attack [37]. As shown in Table 6.4, compared to the existing related works, the proposed implementation offers the best countermeasures to SCAs. As a pseudo-random number generator, LFSR is simple and fast but its output does not have the property of very good randomness. It follows a pattern that can repeat after a certain number of states. Those sequences of numbers are random-like in some aspects. If the attacker knows the seed and also the tap values, the randomness of the generated sequence maybe compromised. A better random generator such like mentioned in [43] can further protect the implementation.

In addition, since our design is implemented using affine coordinates, projective coordinates [29] can be adopted in the design. The advantage of using projective coordinates is that the amount of finite field inversion operation can be greatly reduced with proper pre-computation. Finite field inversion operation is considered as the most time-consuming module in ECC scalar multiplication. So, the computation time may be shortened if projective coordinate systems is adopted.

REFERENCE

[1] V.S. Miller, "Use of Elliptic Curves in Cryptography," Advances in Cryptology Proc. (CRYPTO'85), Springer-Verlag, LNCS 218, pp. 417-426, 1985.

[2] N. Koblitz, "Elliptic Curve Cryptosystems," Mathematics of Computation, vol. 148, pp. 203-209, 1987.

[3] W. Diffie and M.E. Hellman, "New directions in cryptography," IEEE Transaction of Information Theory, vol. 22, pp. 444-454, 1976.

[4] G. Sutter, J. Deschamps, and J. Imana, "Efficient elliptic curve point multiplication using digit-serial binary field operations," IEEE Transactions on Industrial Electronics, vol. 60, pp. 217-225, Jan. 2013.

[5] P. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," Advances in Cryptology Proc. (CRYPTO'96), Springer-Verlag, LNCS 1109, pp. 104-113, 1996.

[6] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," Advances in Cryptology Proc. (CRYPTO'99), Springer-Verlag, LNCS 1666, pp. 388-397, 1999.

[7] J.S. Coron, "Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems," Proc. Int'l Cryptographic Hardware and Embedded Systems (CHES '99), pp. 192-302, Aug. 1999.

[8] A.P Fouque and F. Valette, "The Doubling Attack: Why Upwards is Better than Downwards," Proc. Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES'03), pp.269-280, Sept. 2003.

[9] M. Joye and S.M. Yen, "The Montgomery Powering Ladder," Cryptographic Hardware and Embedded Systems (CHES'02), pp. 291-302, 2002.

[10] P. Kocher, J.Jaffe, and B. Jun, "Differential Power Analysis," Advances in Cryptology Proc. (CRYPTO '99), Springer-Verlag, LNCS 1666, pp. 388-397, 1999.

[11] Y. He, "Highly Secure Cryptographic Computations against Side-Channel Attacks," Master thesis, University of Windsor, 2012.

[12] M. Abdalla, M. Bellare, and P. Rogaway, "DHAES: An encryption scheme based on the Diffie-Hellman problem," IEEE P1363a, 1998.