A hardware implementation of public-key cryptographic processor for 233-bit elliptic curve over binary field

Byung Gwan Park[1] and Kyung Wook Shin
School of Electronic Engineering, Kumoh National Institute of Technology
E-mail : [1]bask369@kumoh.ac.kr

Abstract - **This paper describes a design of 233-bit elliptic curve cryptography (ECC) processor that supports the NIST B-233 elliptic curve over binary field. The ECC-B233 processor computes point addition and point doubling operations on elliptic curve as well as scalar multiplication, which are required for ECC. It adopts the modified Montgomery ladder algorithm for scalar multiplication in order to make it strong against simple power analysis attacks. The finite field multiplication, squaring, and division operations over were implemented using shift-and-add serial multiplication and the extended Euclidian algorithm to achieve area-efficient implementation. A prototype chip for the ECC-B233 processor with 51,900 gates was fabricated using a 0.35-um CMOS process. Chip test results show that the ECC-B233 processor can operate with an operating clock up to 55 MHz. The ECC-B233 processor consumes 490,700 clock cycles for a scalar multiplication, which means that it takes about 9 msec at 55 MHz clock frequency. Forty-two chips out of 48 test chips were working correctly, resulting in the yield of 87%.**

***Keywords*— Elliptic curve cryptography (ECC) processor, Modified Montgomery ladder algorithm, Shift-and-add serial multiplication**

I. INTRODUCTION

Based on the continuously evolving Internet of Things (IoT) and the Internet of Everything (IoE), our society is entering a hyper-connected society. In addition, information exchange using networks, electronic finance, and electronic commerce are increasing. Because information must be protected from security attacks, the importance of security and privacy is becoming more important. Information security is to protect information in various aspects such as confidentiality, integrity, and authentication of data. Cryptographic techniques for information security include symmetric key cryptography and public key cryptography.

Symmetric key cryptography is a method that the same key is used for encryption and decryption. AES (Advanced Encryption Standard) [1] and ARIA (Academy, Research Institute, Agency) [2] are standard cryptographic algorithms of America and Korea, respectively, and are widely used in symmetric key cryptosystems.

Public key cryptography is a method that different keys are used for encryption and decryption. It is complementary to symmetric key cryptography and is used for authentication, key exchange and integrity verification between transceivers. Elliptic curve cryptography (ECC) [3] is considered as one of possible solutions implementing public key cryptosystems for IoT devices and networks. Since the introduction of elliptic curves to cryptography by Victor Miller [4] and Neal

Koblitz [5] independently in 1985, ECC has attracted a lot of attention for public key cryptography. ECC has been standardized by National Institute of Standards and Technology (NIST), American National Standards Institute (ANSI) and IEEE [3, 6]. It is well known that ECC requires shorter key size than RSA for achieving similar security level. ECC with 160-bit key offers similar security level to RSA with 1024-bit key [7]. Due to these advantages, ECC is proposed as a public key cryptography suitable for applications having limited resources including IoT and sensor network.

In this paper, we designed ECC-B233 processor for 233-bit elliptic curve over binary field, and was implemented by MPW. A brief introduction to elliptic curve cryptography is given section II. The design of ECC-B233 processor is explained in section III, and results of functional verification are described in section IV. Section V describes chip test results, and conclusions will be followed in section VI.

II. ELLIPTIC CURVE CRYPTOGRAPHY

The elliptic curves (EC) defined in NIST FIPS 186-2 standard are shown in Table Ⅰ [3], which are classified by prime field EC and binary field EC according to the underlying finite field. Five ECs are defined in prime field, and ten ECs are defined in binary field. In the case of binary field, it is classified into Koblitz curves and pseudo-random curves according to the values of coefficients a and b in the equation defining the EC as shown in equation (1).

$$y^2 + xy = x^3 + ax^2 + b \qquad (1)$$

where $a, b \in GF(2^m)$ and $b \neq 0$.

In the case of Koblitz curves, the cofactor has a relatively high value of 2 or 4. The cofactor is the value of the order of

a. Corresponding author; kwshin@kumoh.ac.kr

Manuscript Received Sep. 28, 2017, Revised Oct. 26, 2017, Accepted Dec. 14, 2017

elliptic curve divided by the order of the base point and is preferably as small as possible for the safety of elliptic curves in cryptographic applications. The pseudo-random curves have a cofactor of 2 regardless of the coefficient value, which ensures a relatively higher safety than the Koblitz curves.

The main operation in ECC is scalar point multiplication that is represented in the form $Q = k * P$ where k is a secret key, P is a point on the elliptic curve, and Q is the result of scalar point multiplication. The safety of ECC is based on the Elliptic Curve Discrete Logarithmic Problem (ECDLP) which means that it is difficult to know k through inverse operations even if we know the points P and Q.

TABLE Ⅰ. Elliptic curves recommended by NIST

	FIPS 186-2	
	Prime Field $GF(p)$	**Binary Field $GF(2^m)$**
Elliptic Curve	P-192	B-163
		K-163
	P-224	B-233
		K-233
	P-256	B-283
		K-283
	P-384	B-409
		K-409
	P-521	B-571
		K-571

The scalar multiplication can be computed by point addition and point doubling operation. When there are two arbitrary points $A(x_0, y_0)$ and $B(x_1, y_1)$ on elliptic curve, the result $C(x_2, y_2)$ of point operation (PO) is calculated by addition, multiplication, division, and squaring operations on the finite field as shown in Table Ⅱ. The performance of scalar point multiplication and finite field arithmetic operations greatly influences the performance and optimization of ECC processor.

The scalar point multiplication can be implemented using binary method [8], Non-Adjacent Form (NAF) [9], Montgomery ladder [10] algorithms, and so on. The binary method performs point doubling for each loop and point addition by the hamming weight of the given secret key. When the NAF algorithm is used, the amount of computation can be reduced compared to the binary method by reducing the hamming weight of the secret key. However, since the above two algorithms differ in the computation amount according to hamming weight of the secret key, the attacker can obtain information about the secret key through side-channel attack such as simple power analysis. Modified Montgomery ladder (MML) algorithm was adopted to implement scalar multiplication, which makes it strong against simple power analysis attack. The MML algorithm was modified to circumvent the constraint that is the most significant bit of integer k must be a value of one.

Additions on binary field $GF(2^m)$ can be implemented with bitwise XOR gates, and there are many algorithms for implementing multiplication and division operations over binary field. Montgomery multiplier [11] can be used to reduce the clock cycles required for multiplication operation as well as the number of partial products. On the other hand, shift-and-add multiplier [8] takes clock cycles as long as the key length, and simple shift and XOR operations are used, resulting in simple hardware.

The division on binary field $GF(2^m)$ can be computed by multiplying the inverse of multiplication. When Fermat's little theorem [11] is used, the inverse of the divisor is obtained using the same property as $a^{-1}(mod\ p) = a^{p-2}(mod\ p)$, and multiplied by the dividend using multiplier. When using the extended Euclidean algorithm [12], the divisor and dividend can be substituted into initial values, and division result can be obtained directly without performing multiplication operation.

TABLE Ⅱ. Point addition and point doubling operations over binary field using affine coordinate

Point addition (C = A + B)	**Point doubling (C = 2A)**
$\lambda = (y_1 + y_0)/(x_1 + x_0)$	$\lambda = x_0 + y_0/x_0$
$x_2 = \lambda^2 + \lambda + x_0 + x_1 + a$	$x_2 = \lambda^2 + \lambda + a$
$y_2 = \lambda(x_0 + x_2) + x_2 + y_0$	$y_2 = x_0^2 + (\lambda + 1)x_2$

Ⅲ. ECC-B233 PROCESSOR DESIGN

A. Overall architecture

We designed an ECC-B233 processor that supports B-233 elliptic curve defined in the NIST FIPS 186-2 standard. The architecture of the ECC-B233 processor is shown in Fig. 1. The ECC-B233 processor consists of Reg_Add block to store intermediate results of scalar multiplication, Alu_GF233 block to perform multiplication, squaring, and division over binary field, control block for controlling

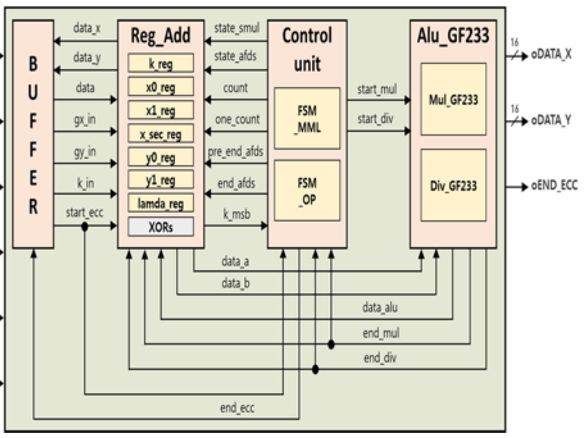


Fig. 1. Architecture of the ECC-B233 processor

scalar point multiplication, and BUFFER block for I/O data conversion. Reg_Add block has seven 233-bit registers to store the coordinates of starting point, secret key k, and intermediate results of scalar multiplication. It also includes XOR gates for addition operations on binary field. The BUFFER block converts 16-bit data entered from outside into 233-bits, then sends them to Reg_Add block. The 233-bit data from Reg_Add are converted into 16-bit data and sent to outside.

B. Alu_GF233 block

Alu_GF233 block is composed of a multiplier and a divider over binary field $GF(2^{233})$. The irreducible polynomial of the elliptic curve B-233 is $f(x) = x^{233} + x^{74} + 1$. In order to reduce hardware complexity, shift-and-add serial multiplier and extended Euclidian algorithm are adopted for binary field arithmetic operations including multiplication, square, and division. The binary field multiplier and divider are implemented by simple shift operations and XORs rather than using multiplication and division operations that require large hardware resources.

Fig. 2 depicts binary field multiplier that consists of three 233-bit registers, MUXs, and XORs. The two input data iDATA_A and iDATA_B are stored in Shift_reg and B_reg, respectively, and iDATA_B or 0 is stored in C_reg according to the least significant bit (LSB) value of iDATA_A. The data stored in Shift_reg is shifted to right by 1 bit per clock cycle and $C\_reg = C\_reg + B\_reg$ operation is performed when the value of Shift_reg[1] is 1. This operation adds the partial product generated in multiplication operation on binary field. AND gates perform modular operation using an irreducible polynomial when the most significant bit (MSB) of the data stored in B_reg is 1. So the intermediate results of multiplication operation are always the element of $GF\ (2^{233})$. The multiplier is designed to output the result from C_reg register with end_mul signal after 233 clock cycles.

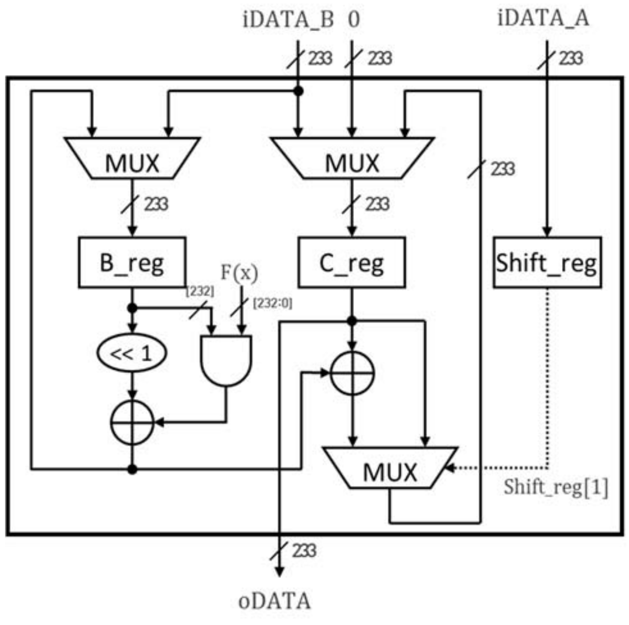


Fig. 2. Binary field multiplier

Division on binary field $GF(2^{233})$ is computed by multiplying the inverse of the multiplication. In this design, the inverse and multiplication operations are simultaneously performed using the extended Euclidean algorithm, and division operation is completed in only 2*m clock cycles. Fig. 3 depicts binary field divider that consists of three 233-bit registers, two 234-bit registers, XOR gates, and so on. The dividend value of division operation is entered to 233-bit input port iDATA_A, and then is stored in the register U_reg. The divisor of division operation is entered to 233-bit input port iDATA_B, and then is stored in the register R_reg. Data stored in T_reg and R_reg are shifted to the left by 1 bit every clock cycle. In the case of data stored in T_reg, the modular operation is performed using an irreducible polynomial after shifting. So the intermediate results of division operation are always the element of $GF\ (2^{233})$. The divider is designed to output the results from V_reg register with end_div signal after 466 clock cycles.

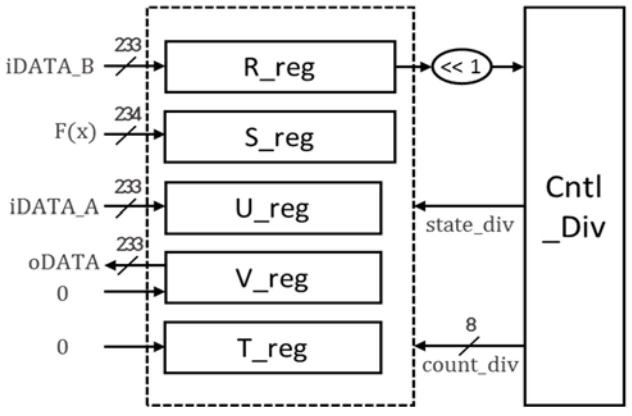


Fig. 3. Binary field divider

C. Control block

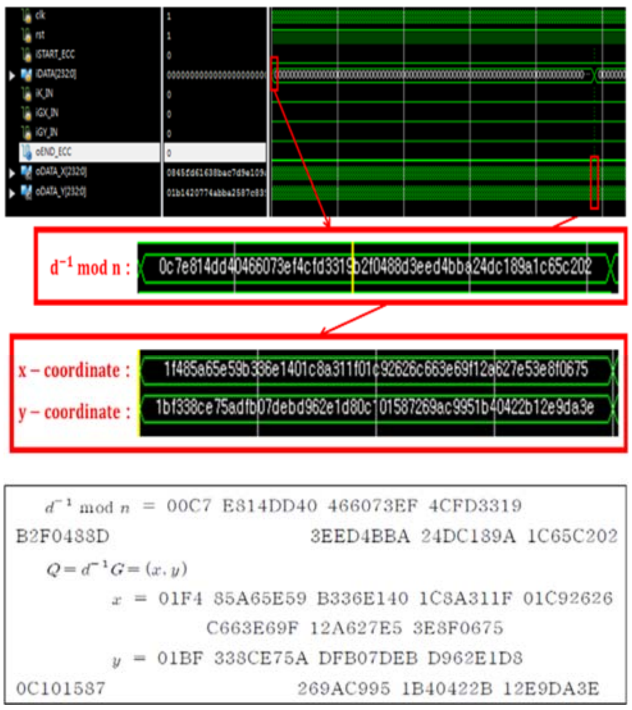
The control block that is a key component for MML algorithm-based scalar point multiplication is composed of finite-state machine as shown in Table Ⅲ. ADDP0_DBLP1 is a state when the bit value of secret key k_i is 1, and ADDP1_DBLP0 is a state when the bit value of secret key k_i is 0. In the above two states, point addition and point doubling operation are performed, and the registers for storing the intermediate results are shared. MAINTAIN state checks the bit value of secret key k_i from MSB to LSB, and does not perform any point operations until the bit value of k_i is 1.

TABLE Ⅲ. Finite-state machine for scalar multiplication

State	Function
IDLE	wait state
MAINTAIN	bit value is checked until it reaches 1 of the secret key k_{i}
ADDP0_DBLP1	bit value of the secret key k_i is 1
ADDP1_DBLP0	bit value of the secret key k_i is 0

Ⅳ. FUNCTIONAL VERIFICATION

Functional verification of the ECC-B233 processor designed in Verilog HDL was carried out using Xilinx ISim. Fig. 4 shows the simulation results of the ECC-B233 processor and reference values of document [13]. The B-233 elliptic curve parameters defined in NIST FIPS 186-2 [3] were used, and a 233-bit secret key of “0c7 e814dd40 466073ef 4cfd3319 b2f0488d 3eed4bba 24dc189a 1c65c202” was multiplied to generation point. After iSTART_ECC signal is activated, a scalar multiplication is performed for 490,699 clock cycles. It can be confirmed that two coordinate values for which the scalar multiplication is completed are output simultaneously with oEND_ECC signal. The x-coordinate of scalar multiplication “1f4 85a65e59 b336e140 1c8a311f 01c92626 c663e69f 12a627e5 3e8f0675” and the y-coordinate “1bf 338ce75a dfb07deb d962e1d8 0c101587 269ac995 1b40422b 12e9da3e” are exactly same as the reference implementation values.


clk
rst
iSTART_ECC
iDATA[232:0]
iK_IN
iGX_IN
iGY_IN
oEND_ECC
oDATA_X[232:0]
oDATA_Y[232:0]
1
1
0
0
0
0
0
0
d^{-1} mod n :
0c7e814dd40466073ef4cfd3319b2f0488d3eed4bba24dc189a1c65c202
x – coordinate :
1f485a65e59b336e1401c8a311f01c92626c663e69f12a627e53e8f0675
y – coordinate :
1bf338ce75adfb07debd962e1d80c101587269ac9951b40422b12e9da3e

$d^{-1} \bmod n$ = 00C7 E814DD40 466073EF 4CFD3319 B2F0488D 3EED4BBA 24DC189A 1C65C202

$Q = d^{-1}G = (x, y)$

x = 01F4 85A65E59 B336E140 1C8A311F 01C92626 C663E69F 12A627E5 3E8F0675

y = 01BF 338CE75A DFB07DEB D962E1D8 0C101587 269AC995 1B40422B 12E9DA3E

Fig. 4. Functional verification results of ECC-B233 processor

Ⅴ. IMPLEMENTATION AND CHIP TEST

A. Layout design

The ECC-B233 processor was implemented with MPW chip using a 0.35μm CMOS process. Table Ⅳ depicts design and verification flow. Xilinx ISim was used for RTL simulation. Synopsys DesignCompiler was used to convert the HDL code modeled at RTL into a gate-level circuit. After logic synthesis was completed, the gate-level netlist was generated and formality was used to verify that the netlist and HDL code were the same. Synopsys PrimeTime was used for both pre-layout and post-layout STAs. Static Timing Analysis (STA) analyzes the operation timing of synthesized circuit by analyzing the violation of setup time and hold time of the register. Mentor Graphics Modelsim was used for both pre- and post-layout simulations. P&R design of ECC-B233 processor was carried out using Astro. The parasitic RC values were extracted using Star-RCXT. Fig. 5 shows the layout of ECC-B233 processor whose area is 2,170 × 2,080μm^2.

TABLE Ⅳ. Design and verification flow of ECC-B233 processor

Process	Tool
1. RTL Modeling & Functional Verification	ISim
2. Logic Synthesis	DesignCompiler
3. Equivalence Check	Formality
4. Pre-layout STA	Primetime
5. Pre-layout Simulation	Modelsim
6. Place & Routing	Astro
7. Equivalence Check	Formality
8. RC Extraction	Star-RCXT
9. Post-layout STA	Primetime
10. Post-layout Simulation	Modelsim

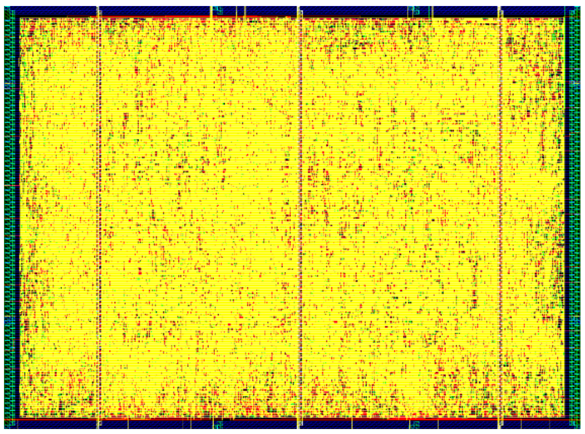


Fig. 5. Layout of ECC-B233 processor

B. Chip Test Result

Fig. 6 shows the chip test setup that consists of a test board and GUI software on PC. The test board contains a socket for mounting a chip under test, a FPGA device of Xilinx Spartan3 XC3S1000, and some switches. In order to send test vectors from PC to test board, and to receive test responses from test board, UART and wrapper circuits are implemented in FPGA device. Elliptic curve parameters and secret key are sent to the wrapper implemented in FPGA, then are moved into the test chip (ECC-B233 processor). In the ECC-B233 processor, scalar-multiplied data are sent back to PC via FPGA.

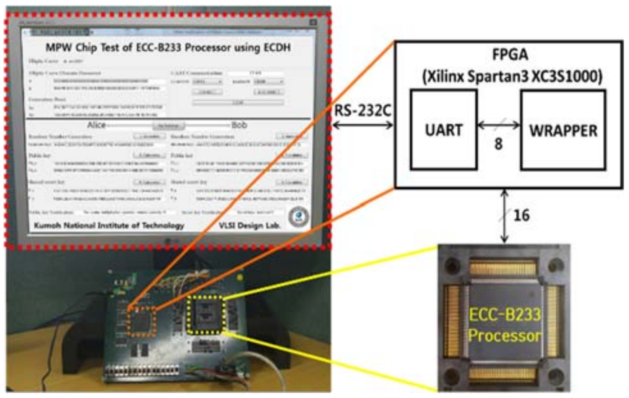
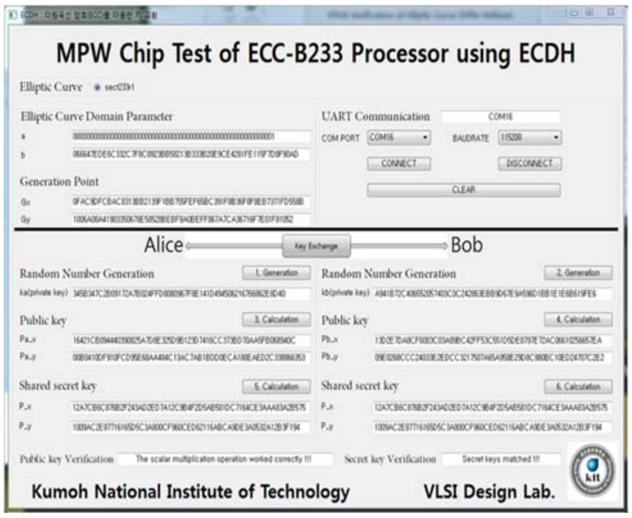

FPGA
(Xilinx Spartan3 XC3S1000)
RS-232C
UART
8
WRAPPER
16
ECC-B233 Processor
MPW Chip Test of ECC-B233 Processor using ECDH
Kumoh National Institute of Technology
VLSI Design Lab.

Fig. 6. Chip test setup



MPW Chip Test of ECC-B233 Processor using ECDH

Elliptic Curve ● sect233r1

Elliptic Curve Domain Parameter

a 000000000000000000000000000000000000000000000000000000000001

b 066647EDE6C332C7F8C0923BB58213B333B20E9CE4281FE115F7D8F90AD

Generation Point

Gx 0FAC9DFCBAC8313BB2139F1BB755FEF65BC391F8B36F8F8EB7371FD558B

Gy 1006A08A41903350678E58528BEBF8A0BEFF867A7CA36716F7E01F81052

UART Communication COM16

COM PORT COM16 BAUDRATE 115200

CONNECT DISCONNECT CLEAR

Alice Key Exchange Bob

Random Number Generation 1. Generation

ka(private key) [illegible]

Public key 3. Calculation

Pa.x [illegible]

Pa.y [illegible]

Shared secret key 5. Calculation

P.x [illegible]

P.y [illegible]

Random Number Generation 2. Generation

kb(private key) [illegible]

Public key 4. Calculation

Pb.x [illegible]

Pb.y [illegible]

Shared secret key 6. Calculation

P.x [illegible]

P.y [illegible]

Public key Verification The scalar multiplication operation worked correctly !!!

Secret key Verification Secret keys matched !!!

Kumoh National Institute of Technology VLSI Design Lab.

Fig. 7. Test result of ECC-B233 processor

Fig. 7 shows chip test result using Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol that uses scalar point multiplications in EC. The x-coordinate G_x of generation point on EC is "0fa c9dfcbac 8313bb21 39f1bb75 5fef65bc 391f8b36 f8f8eb73 71fd558b" and the y-coordinate G_y is "100 6a08a419 03350678 e58528be bf8a0bef f867a7ca 36716f7e 01f81052". Alice and Bob share the two coordinates G_x and G_y of generation point, and then generate random constants k_a and k_b that are greater than 0 and less than the order of generation point. The constant k_a is used as Alice's secret key, and k_b is used as Bob's secret key. Two secret keys k_a, k_b and two coordinates G_x, G_y are sent to the wrapper implemented in FPGA, and the data are sent from the wrapper to the chip. Alice and Bob compute scalar multiplications of $P_a = k_a * G$ and $P_b = k_b * G$ using the ECC-B233 processor implemented in chip, respectively. After Alice and Bob exchange P_a and P_b each other, another scalar multiplications of $P_{alice} = k_a * P_b$ and $P_{bob} = k_b * P_a$ are calculated by the ECC-B233 processor. After key exchange is completed, the software verifies whether the ECC-B233 chip works correctly by comparing the calculated final values P_{alice} and P_{bob}, as well as they are included in the B-233 elliptic curve. Total 48 chips were tested with operating clock of 55 MHz and 42 chips work correctly, resulting in 87% yield.

VI. CONCLUSIONS

An ECC-B233 processor supporting B-233 elliptic curve defined in FIPS 186-2 was designed and fabricated in $0.35\mu m$ CMOS process. The designed ECC processor was implemented using modified Montgomery ladder algorithm to be safe for simple power analysis. The multiplication and division operators on binary field were implemented using simple shift operation and XOR gates to minimize hardware resource consumption.

The ECC-B233 processor integrates 51,900 GEs on the area of $2{,}170 \times 2{,}080\mu m^2$. Chip test was performed with operating clock of 55 MHz, and it requires 8.9 msec for a scalar point multiplication. Among 48 chips tested, 42 chips were operated correctly, resulting in yield of 87%. The ECC-B233 processor can operate complex operations of public key cryptosystem ECC at a high speed, so it can be used to implement security system where code size or memory is limited.

ACKNOWLEDGMENT

This work was supported by Industrial Core Technology Development Program (10049009, Development of Main IPs for IoT and Image- Based Security Low-Power SoC) funded by the Ministry of Trade, Industry & Energy.

This work was supported by the Korea Institute for Advancement of Technology (KIAT) grant funded by the Korean government (Motie:Ministry of Trade, Industry&Energy, HRD Program for Software-SoC convergence) (No. N0001883)

The authors are thankful to IDEC for supporting EDA software.

REFERENCES

[1] FIPS-197, Advanced Encryption Standard, National Institute of Standard and Technology (NIST), Nov. 2001.

[2] KS X 1213:2004, 128 bit Block Encryption Algorithm ARIA, Korean Agency for Technology and Standard (KATS), 2004.

[3] NIST Std. FIPS PUB 186-2: Digital Signature Standard (DSS), NIST, Jan. 2000.

[4] V. Miller, "Uses of Elliptic Curve cryptography," *Advances of Cryptography*, LNCS 218, pp. 417-426, 1985.

[5] N. Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203-209, 1987.

[6] ANSI Std. X9.63: Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography, ANSI, Nov. 2000.

[7] H.A. Selma and H. M'hamed, "Elliptic curve cryptographic processor design using FPGAs," *IEEE 2015 International Conf. on Engineering & Information Technology (CEIT)*, pp. 1-6, 2015.

[8] M. Amara and A. Siad, "Hardware implementation of Elliptic Curve Point Multiplication over GF(2^m) for ECC protocols," *International Journal for Information*

Security Research (IJISR), vol. 2, no. 1, pp. 106-112, March. 2012.

[9] V.R. Venkatasubramani, G.R. Kumar, and K. Vignesh, “Fast computation of scalar multiplication over binary edwards curve processor against side channel attack,” *IEEE 2014 International Conf. on Electronics and Communication Systems (ICECS)*, pp. 1-7, 2014.

[10] C. Rebeiro, SS. Roy, and D. Mukhopadhyay. "Pushing the limits of high-speed GF (2^m) elliptic curve scalar multiplication on FPGAs." *International Workshop on Cryptographic Hardware and Embedded Systems. Springer Berlin Heidelberg*, pp. 494-511, 2012.

[11] D. Amiet, A. Curiger, and P. Zbinden, “Flexible FPGA-Based Architectures for Curve Point Multiplication over GF(p),” *IEEE 2016 Euromicro Conference on Digital System Design (DSD)*, pp. 107-114, 2016.

[12] J. Park, J.T. Hwang, and Y.C. Kim, “FPGA and ASIC implementation of ECC processor for security on medical embedded system,” *IEEE Third International Conference on Information Technology and Applications (ICITA 2005)*, vol. 2, pp. 547-551, 2005.

[13] TTA Std. TTAK.KO-12.0015/R1, Digital Signature Mechanism with Appendix(Part 3) Korean Certificate-based Digital Signature Algorithm using Elliptic Curves, 2012.

Byung Gwan Park received the B.S. degree in electronic engineering from Kumoh National Institute of Technology, Gumi, Korea, in 2015 and is currently working toward the M.S. degree.

His main interests include semiconductor IP design for communications and signal process, semiconductor IP design for information security.

Kyung Wook Shin received the B.S. degree in electronic engineering from Korea Aerospace University, Goyang, Korea, 1984. He received the M.S. degree and the Ph.D. degree, both in electronic engineering, from Yonsei University, Seoul, Korea, in 1990 and 1990.

He worked at Electronics and Telecommunications Research Institute (ETRI) from 1990 to 1991. He is currently a professor of school of electronic engineering at Kumoh National Institute of Technology since 1991. He spent a sabbatical year as visiting professor at University of Illinois at Urbana-Champaign during 1995-1996, and at University of California at San Diego during 2003-2004 and at Georgia Institute of Technology during 2013-2014. His research interests include SoC design for communications and signal processing, SoC design for information security, and semiconductor IP design**.**