International Journal of Networking and Computing – www.ijnc.org, ISSN 2185-2847
Volume 10, Number 2, pages 277–292, July 2020

ECC Atomic Block with NAF against Strong Side-Channel Attacks on Binary Curves

Yusuke Takemura
Interdisciplinary Graduate School of Science and Engineering, Shimane University,
1060 Nishikawatsucho, Matsue, Shimane, 690-8504, Japan

Keisuke Hakuta
Institute of Science and Engineering, Shimane University,
1060 Nishikawatsucho, Matsue, Shimane, 690-8504, Japan

Naoyuki Shinohara
Security Fundamentals Laboratory,
National Institute of Information and Communications Technology,
4-2-1 Nukui-Kitamachi, Koganei, Tokyo, 184-8795, Japan

Received: February 15, 2020
Revised: May 5, 2020
Accepted: June 11, 2020
Communicated by Yasuyuki Nogami

Abstract

Various side-channel attacks against elliptic curve cryptography (ECC) have been proposed so far, including simple power analysis, horizontal collision correlation analysis, improving the Big Mac attack, and differential power analysis. Developing countermeasures against such attacks is considered as an important research task in cryptography. They are executed by analyzing power consumption while a device implemented in a cryptosystem performs cryptographic processing. To address this problem, we propose using three atomic blocks serving as the countermeasures against such attacks on ECC over finite fields of characteristic two. Two of them are basic atomic blocks, while the third one is an improved version of these two, having lower computational cost. In this paper, concerning the possibility of more sophisticated side-channel attacks appearing in the future, we propose a threat model based on the atomic blocks that is constructed to be secure for strong side-channel attacks with more powerful abilities.

Keywords: Elliptic curve cryptography, side-channel analysis, atomic block, non-adjacent form (NAF), scalar multiplication

1 Introduction

Elliptic curve cryptography (ECC) is a public cryptosystem proposed by Koblitz [9] and Miller [12]. ECC is capable of providing a particular security level, such as 128-bit security, using a shorter key length compared to that of RSA, and herefore, deemed suitable for low-specification hardware. The security level of ECC depends on hardness of solving an elliptic curve discrete logarithm problem

[0]This paper is an extended version of a paper from WICS'19 [16].

(ECDLP), which is defined as a problem of computing a secret integer d such that $Q = [d]P$ for given rational points P and Q on a given elliptic curve. The time complexity of the most efficient algorithm to solve ECDLP is exponential; therefore, ECC is considered secure. However, even if ECDLP is difficult to solve, side-channels can be utilized to leak confidential information processed in ECC.

In ECC, computation of a point $[d]P$ is referred to as elliptic curve scalar multiplication (ECSM). Most of the algorithms developed to compute ECSM correspond to two types of algorithms: elliptic curve addition (abbreviated to ECADD) and elliptic curve doubling (abbreviated to ECDBL). In ECSM, $[d]P$ is computed through a sequence of computations corresponding to ECADD and ECDBL, and the information within this sequence has to be hidden, as it contains partial information about a secret integer d. To obtain the information about the sequence, side-channel attackers attempt to identify differences in the side-channel information between ECDBL and ECADD. Kocher *et al.* proposed a method of breaking the hardware implementations of ECC by using simple power analysis (SPA), to obtain the side-channel information regarding changes in computation patterns [10]. Use of atomic blocks, as proposed by Chevallier-Mames *et al.* [4], can serve as a countermeasure against these attacks. However, Kocher *et al.* demonstrated that differential power analysis (DPA) can be applied to fail such atomic blocks [11]. Then, Coron proposed the use of randomization techniques as a countermeasure against DPA [5]. However, it is known that the leakage of additional side-channel information besides computation patterns can lead to ECC failure even under the application of randomization techniques. For example, Walter proposed an improved single-trace attack for RSA protected from SPA and DPA [17]. This idea served as a foundation for implementing horizontal collision correlation analysis (HCCA) [3] and the improved big mac attack (IBMA) [6] for ECC. As mentioned above, side-channel attacks using power consumption on ECC have researched more than 20 years, and are major research topics in cryptography. Countermeasures against SPA and DPA on binary curves treated in this paper have already been proposed [4,5], however, these countermeasures can not defeat HCCA or IBMA. Therefore, it is desirable to construct a countermeasure to defeat all these four attacks (SPA, HCCA, IBMA, DPA).

This paper proposes the implementation of three atomic blocks (represented in Table 1, Table 2 and Table 5) as the countermeasures against SPA, DPA, HCCA, and IBMA attacks. Furthermore, we discuss the computational costs associated with each atomic block, indicating that the computational efficiency of the proposed improved atomic block (Table 5) is 1.625 and 1.8 times better than that of the basic atomic block (Table 1), and another (Table 2), respectively.

This paper is organized as follows: In Section 2, we introduce the mathematical back ground on ECC, ECDBL, ECADD, and *López-Dahab* projective coordinates. The four above side-channel attacks are explained in Section 3. And in that section, we propose two basic atomic blocks and explain that they are countermeasures against those side-channel attacks. We propose more effective atomic block using NAF method in Section 4. The computational costs of our atomic blocks are discussed in Section 5. Section 6 concludes the paper and future work.

2 Elliptic curve arithmetic over $\mathbb{F}_{2^m}$

One of the elliptic curves recommended by the National Institute of Standards and Technology (NIST) for use in ECC is an elliptic curve on a finite field $\mathbb{F}_{2^m}$ [8, Appendix D]. NIST proposed several types of elliptic curves over $\mathbb{F}_{2^m}$. In this paper, we focus on K-curves and B-curves which NIST recommended. A detailed explanation of the minimal mathematical background to understand the algorithms proposed in this paper can be found in [7].

2.1 Elliptic curves over a binary field

An elliptic curve $E/\mathbb{F}_{2^m}$ over a finite field $\mathbb{F}_{2^m}$ is represented by

$$E/\mathbb{F}_{2^m} : y^2 + xy = x^3 + ax^2 + b,$$

where $a \in \mathbb{F}_{2^m}$ and $\Delta := b \in \mathbb{F}_{2^m}^*$ (Δ is the discriminant of the curve [7]). For K-curves and B-curves, $a = 0$ or $a = 1$. An ECDBL is a calculation to compute $[2]P$ for a point $P = (x_1, y_1)$ on $E/\mathbb{F}_{2^m}$.

The coordinates x_3 and y_3 of $[2]P$ are given by

$$\begin{aligned} x_3 &= \lambda^2 + \lambda + a, \\ y_3 &= x_1^2 + (\lambda + 1)x_3, \\ \lambda &= x_1 + \frac{y_1}{x_1}. \end{aligned} \tag{1}$$

Also, an ECADD is a calculation to compute $P + Q$ for two different points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on $E/\mathbb{F}_{2^m}$. The coordinates (x_3, y_3) of $P + Q$ are given by

$$\begin{aligned} x_3 &= \lambda^2 + \lambda + x_1 + x_2 + a, \\ y_3 &= \lambda(x_1 + x_3) + x_3 + y_1, \\ \lambda &= \frac{y_1 + y_2}{x_1 + x_2}. \end{aligned} \tag{2}$$

2.2 Elliptic arithmetic with *López-Dahab* projective coordinates

When performing these ECADD and ECDBL operations, multiplicative inverse operations over a finite field are required to compute λ in Equation (1) and Equation (2). The calculation of multiplicative inverses in a finite field is computationally more expensive than other arithmetic operations in a finite field. However, by using *López-Dahab* projective coordinates (*LD* coordinates), it is possible to perform ECADD and ECDBL calculations without computing multiplicative inverses over a finite field. This section introduces *LD* coordinates to perform ECDBL and ECADD calculations. A point P on an elliptic curve in *LD* coordinates is represented as $(X : Y : Z)$, and its *affine* projective coordinates (x, y) are associated with the *LD* projective coordinates by the relationship $x = X/Z, y = Y/Z^2$ for some $Z \in \mathbb{F}_{2^m}^*$. When introducing LD projective coordinates, an elliptic curve $E/\mathbb{F}_{2^m}$ over a finite field $\mathbb{F}_{2^m}$ is represented as

$$E/\mathbb{F}_{2^m} : Y^2 + XYZ = X^3Z + aX^2Z^2 + bZ^4. \tag{3}$$

For a point $P = (X_1 : Y_1 : Z_1)$ in *LD* projective coordinates, the ECDBL point $[2]P = (X_3 : Y_3 : Z_3)$ can be calculated using Algorithm 1 [7, Algorithm 3.24].

Algorithm 1 ECDBL using *LD* coordinates [7, Algorithm 3.24]

Input: $P = (X_1 : Y_1 : Z_1)$ in *LD* coordinates on $E/\mathbb{F}_{2^m}$.
Output: $[2]P = (X_3 : Y_3 : Z_3)$ in *LD* coordinates.
1: $T_1 \leftarrow Z_1^2$.
2: $T_2 \leftarrow X_1^2$.
3: $Z_3 \leftarrow T_1 \cdot T_2$.
4: $X_3 \leftarrow T_2^2$.
5: $T_1 \leftarrow T_1^2$.
6: $T_2 \leftarrow T_1 \cdot b$.
7: $X_3 \leftarrow X_3 + T_2$.
8: $T_1 \leftarrow Y_1^2$.
9: **If** $a = 1$ **then** : $T_1 \leftarrow T_1 + Z_3$.
10: $T_1 \leftarrow T_1 + T_2$.
11: $Y_3 \leftarrow X_3 \cdot T_1$.
12: $T_1 \leftarrow T_2 \cdot Z_3$.
13: $Y_3 \leftarrow Y_3 + T_1$.
14: **return** $(X_3 : Y_3 : Z_3)$.

When performing an ECADD operation, the *LD* projective and *affine* coordinates can be used to obtain the results in *LD* projective coordinates. That is, for the two different rational points $P = (X_1 : Y_1 : Z_1)$ and $Q = (x_2, y_2)$, the ECADD point $P + Q = (X_3 : Y_3 : Z_3)$ can be calculated using Algorithm 2 [7, Algorithm 3.25].

Algorithm 2 ECADD using *LD* and *affine* coordinates [7, Algorithm 3.25]

Input: $P = (x_2, y_2)$ in *affine* coordinates on $E/\mathbb{F}_{2^m}$,
$Q = (X_1 : Y_1 : Z_1)$ in *LD* coordinates on $E/\mathbb{F}_{2^m}$.
Output: $P + Q = (X_3 : Y_3 : Z_3)$ in *LD* coordinates.

1: $T_1 \leftarrow Z_1 \cdot x_2$.
2: $T_2 \leftarrow Z_1^2$.
3: $X_3 \leftarrow X_1 + T_1$.
4: $T_1 \leftarrow Z_1 \cdot X_3$.
5: $T_3 \leftarrow T_2 \cdot y_2$.
6: $Y_3 \leftarrow Y_1 + T_3$.
7: $Z_3 \leftarrow T_1^2$.
8: $T_3 \leftarrow T_1 \cdot Y_3$.
9: **If** $a = 1$ **then** : $T_1 \leftarrow T_1 + T_2$.
10: $T_2 \leftarrow X_3^2$.
11: $X_3 \leftarrow T_2 \cdot T_1$.
12: $T_2 \leftarrow Y_3^2$.
13: $X_3 \leftarrow X_3 + T_2$.
14: $X_3 \leftarrow X_3 + T_3$.
15: $T_2 \leftarrow x_2 \cdot Z_3$.
16: $T_2 \leftarrow T_2 + X_3$.
17: $T_1 \leftarrow Z_3^2$.
18: $T_3 \leftarrow T_3 + Z_3$.
19: $Y_3 \leftarrow T_3 \cdot T_2$.
20: $T_2 \leftarrow x_2 + y_2$.
21: $T_3 \leftarrow T_1 \cdot T_2$.
22: $Y_3 \leftarrow Y_3 + T_3$.
23: **return** $(X_3 : Y_3 : Z_3)$.

Scalar multiplications of rational points on an elliptic curve (abbreviated to ECSM) appear in ECC operations. For example, a value d, such as a secret key, is used to calculate the product $[d]P$ of d and a point P on the elliptic curve. ECSM can be performed by using the left-to-right algorithm, which is implemented using ECADD and ECDBL operations. Algorithm 3 is an ECSM using left-to-right algorithm with *LD* projective coordinates. Algorithm 3 looks at each bit of d in turn, starting from the bit d_{l-2} and ending with the least significant bit d_0. If bit d_i is zero, only the ECDBL operation is performed; otherwise the ECDBL and ECADD operations are performed in sequence. Therefore, an attacker that has some way of determining the difference between the case when ECADD and ECDBL operations are performed inside the equipment will be able to identify the bits of the secret information d, thereby retrieving the value of d.

Algorithm 3 Left-to-Right Scalar Multiplication Algorithm

Input: $d = (d_{\ell-1}, d_{\ell-2}, \ldots, d_1, d_0)_2$, $P = (x_2, y_2) \in E/\mathbb{F}_{2^m}$, where $d_{\ell-1} = 1$.
Output: $[d]P$.

1: Select $Z \in \mathbb{F}_{2^m}^*$ randomly.
2: $Q \leftarrow (xZ : yZ^2 : Z)$. ($Q$ in *LD* coordinate)
3: **for** $i = \ell - 2 \rightarrow 0$ **do**
4: $Q \leftarrow [2]Q$. (compute ECDBL with Q)
5: **if** $d_i = 1$ **then**
6: $Q \leftarrow Q + P$. (compute ECADD with P and Q)
7: **end if**
8: **end for**
9: **return** Q.

3 Side-channel attacks and countermeasures

One of the possible ways attack ECC is to perform a side-channel attack, a generic term denoting attack methods that are used to gather and analyze information about the data corresponding to electronic hardware, such as its power consumption or processing timing, registered during cryptographic processes. Using the results of such analysis, the confidential information processed on the hardware can be retrieved. Various side-channel attacks on ECC have already been proposed, and therefore, it is important to include side-channel attack countermeasures to operate ECC securely.

In this section, we discuss various ECC side-channel attacks and their countermeasures. Then, we describe how our proposed atomic blocks can be applied to protect against these attacks. In this paper, we assume that side-channel attacks are launched by an attacker having information about the used ECC elliptic curve and corresponding coordinates in the hardware, as well as the algorithms used for ECADD, ECDBL, and scalar multiplication. Recent trends in side-channel attacks are discussed in [1]. Concerning the elliptic curve defined in Equation (3), the ECDBL and ECADD operations are performed differently depending on whether the coefficient a is 0 or 1, according to step 9 in Algorithm 1 and Algorithm 2. If the differences between the ECADD and ECDBL operations are identified by the attacker, the secret information d used in Algorithm 3 can be recovered, as mentioned in Section 2.2. As the algorithms used to perform ECADD and ECDBL should be constructed in such a way to prevent attackers from distinguishing the difference between the ECDBL (Algorithm 1) and ECADD (Algorithm 2) operations for each coefficient $a = 0, 1$, we develop basic atomic blocks (Table 1 and Table 2) considering both possible values of coefficient a.

3.1 Simple power analysis

Simple Power Analysis (SPA) is a side-channel attack proposed by Kocher *et al.* [10] in which secret information d can be identified on the basis of a power consumption waveform corresponding to the hardware that performs cryptographic processing. The underlying concept of SPA is that an attacker can classify the type of field arithmetic processing based on a power trace registered for a cryptographic device.

3.1.1 SPA attack methodology

The computational time and power consumption associated with field multiplication are greater than those associated with field addition. As a result, an attacker can identify whether the hardware executes field multiplication or field addition, by analyzing its power consumption. Therefore, the order in which field multiplication and field addition operations are executed can be determined. Here, when ECC is implemented naïvely using Algorithm 1 and Algorithm 2, the ECDBL computations start with a sequence of six-field multiplication operations, while the ECADD computations start with a sequence of two-field multiplications, followed by a field addition operation. Analyzing such differences, an attacker can distinguish between the ECADD and ECDBL computations, thereby recovering the secret information d.

3.1.2 Using side-channel atomicity to defeat SPA

As SPA attacks are based on the differences in a sequence of field multiplication operations run on cryptographic hardware, it is possible to take countermeasures against them by performing the ECADD and ECDBL operations, including filed multiplications and additions, in the same order. According to this strategy, the concept of atomic blocks has been proposed as a countermeasure against SPA on ECC [4]. Atomic blocks are the algorithms based on atomic patterns that compute ECADD and ECDBL, such as those represented in Table 1 and Table 2. Furthermore, an atomic pattern has a unified order of field addition and multiplication calculations for executing ECADD and ECDBL on a device.

In this paper, we propose basic atomic blocks, as described in Table 1 and Table 2. For one-time computation of ECDBL in the case of $a = 0$, the column ECDBL of Table 1 is executed one time. For one-time computation of ECADD in the same case, the column ECADD1 is run followed by

Table 1: Our atomic block for $a = 0$

Step	ECDBL	ECADD1	ECADD2	Atomic pattern
1.	$T_1 \leftarrow Z_1^2$	$\star$	$T_{10} \leftarrow T_8^2$	$R_1 \leftarrow R_2^2$
2.	$D_1 \leftarrow T_1 \times x_2$	$T_1 \leftarrow Z_1 \times x_2$	$T_{13} \leftarrow Z_3 \times x_2$	$R_3 \leftarrow R_4 \times x_2$
3.	$T_4 \leftarrow {T_1}^2$	$T_2 \leftarrow {Z_1}^2$	$T_{14} \leftarrow {Z_3}^2$	$R_5 \leftarrow {R_4}^2$
4.	$\star$	$T_3 \leftarrow X_1 + T_1$	$T_{11} \leftarrow T_9 + T_{10}$	$R_6 \leftarrow R_7 + R_8$
5.	$T_2 \leftarrow X_1^2$	$\star$	$\star$	$R_9 \leftarrow R_{10}^2$
6.	$Z_3 \leftarrow T_1 \times T_2$	$T_4 \leftarrow Z_1 \times T_3$	$D_4 \leftarrow Z_3 \times D_5$	$R_{11} \leftarrow R_4 \times R_{12}$
7.	$T_6 \leftarrow {Y_1}^2$	$Z_3 \leftarrow {T_4}^2$	$D_9 \leftarrow {T'_8}^2$	$R_{13} \leftarrow {R_{14}}^2$
8.	$T_3 \leftarrow {T_2}^2$	$T_7 \leftarrow {T_3}^2$	$D_7 \leftarrow {D_5}^2$	$R_{15} \leftarrow {R_{12}}^2$
9.	$D_8 \leftarrow D_8 \times Y_1$	$T_9 \leftarrow T_7 \times T_4$	$T'_{12} \leftarrow T'_4 \times T'_8$	$R_{16} \leftarrow R_{17} \times R_{14}$
10.	$\star$	$T'_4 \leftarrow r_1 + T_4$	$T_{12} \leftarrow T'_{12} + r''_1$	$R_{18} \leftarrow R_{19} + R_{20}$
11.	$T_5 \leftarrow T_4 \times b$	$D_4 \leftarrow D_3 \times b$	$D_6 \leftarrow D_6 \times b$	$R_{21} \leftarrow R_{22} \times b$
12.	$D_2 \leftarrow D_2 \times y_2$	$T_2 \leftarrow T_2 \times y_2$	$D_7 \leftarrow D_7 \times y_2$	$R_{23} \leftarrow R_{24} \times y_2$
13.	$X_3 \leftarrow T_3 + T_5$	$T_{17} \leftarrow x_2 + y_2$	$X_3 \leftarrow T_{11} + T_{12}$	$R_{25} \leftarrow R_{26} + R_{27}$
14.	$\star$	$T_8 \leftarrow Y_1 + T_2$	$T_{15} \leftarrow T_{13} + X_3$	$R_{28} \leftarrow R_{29} + R_{30}$
15.	$T_8 \leftarrow T_6 + T_5$	$r'_1 \leftarrow T'_4 + T_8$	$T_{16} \leftarrow T_{12} + Z_3$	$R_{31} \leftarrow R_{32} + R_{33}$
16.	$T_9 \leftarrow X_3 \times T_8$	$\star$	$T_{18} \leftarrow T_{16} \times T_{15}$	$R_{34} \leftarrow R_{35} \times R_{36}$
17.	$T_{10} \leftarrow T_5 \times Z_3$	$r''_1 \leftarrow r_1 \times r'_1$	$T_{19} \leftarrow T_{14} \times T_{17}$	$R_{37} \leftarrow R_{38} \times R_{39}$
18.	$Y_3 \leftarrow T_9 + T_{10}$	$T'_8 \leftarrow R_1 + T_8$	$Y_3 \leftarrow T_{18} + T_{19}$	$R_{40} \leftarrow R_{41} + R_{42}$

Table 2: Our atomic block for $a = 1$

Step	ECDBL	ECADD1	ECADD2	Atomic pattern
1.	$T_1 \leftarrow Z_1^2$	$\star$	$T_{10} \leftarrow T_8^2$	$R_1 \leftarrow R_2^2$
2.	$T_2 \leftarrow X_1^2$	$\star$	$\star$	$R_3 \leftarrow R_4^2$
3.	$\star$	$\star$	$T'_{12} \leftarrow T'_4 \times T'_8$	$R_5 \leftarrow R_6 \times R_7$
4.	$D_1 \leftarrow T_1 \times x_2$	$T_1 \leftarrow Z_1 \times x_2$	$T_{13} \leftarrow Z_3 \times x_2$	$R_8 \leftarrow R_9 \times x_2$
5.	$\star$	$\star$	$T_{11} \leftarrow T_9 + T_{10}$	$R_{10} \leftarrow R_{11} + R_{12}$
6.	$\star$	$T_3 \leftarrow X_1 + T_1$	$T_{12} \leftarrow T'_{12} + r''_1$	$R_{13} \leftarrow R_{14} + R_{15}$
7.	$T_4 \leftarrow {T_1}^2$	$T_2 \leftarrow {Z_1}^2$	$T_{14} \leftarrow {Z_3}^2$	$R_{16} \leftarrow {R_9}^2$
8.	$Z_3 \leftarrow T_1 \times T_2$	$T_4 \leftarrow Z_1 \times T_3$	$D_4 \leftarrow Z_3 \times D_5$	$R_{17} \leftarrow R_9 \times R_{18}$
9.	$\star$	$T'_4 \leftarrow r_1 + T_4$	$X_3 \leftarrow T_{11} + T_{12}$	$R_{19} \leftarrow R_{20} + R_{21}$
10.	$T_3 \leftarrow {T_2}^2$	$T_7 \leftarrow {T_3}^2$	$D_7 \leftarrow {D_5}^2$	$R_{22} \leftarrow {R_{18}}^2$
11.	$T_5 \leftarrow T_4 \times b$	$D_4 \leftarrow D_3 \times b$	$D_6 \leftarrow D_6 \times b$	$R_{23} \leftarrow R_{24} \times b$
12.	$X_3 \leftarrow T_3 + T_5$	$T_6 \leftarrow T_4 + T_2$	$T_{15} \leftarrow T_{13} + X_3$	$R_{25} \leftarrow R_{26} + R_{27}$
13.	$T_6 \leftarrow Y_1^2$	$Z_3 \leftarrow T_4^2$	$\star$	$R_{28} \leftarrow R_{29}^2$
14.	$D_2 \leftarrow D_2 \times y_2$	$T_2 \leftarrow T_2 \times y_2$	$D_7 \leftarrow D_7 \times y_2$	$R_{30} \leftarrow R_{31} \times y_2$
15.	$T_7 \leftarrow T_6 + Z_3$	$T_8 \leftarrow Y_1 + T_2$	$T_{16} \leftarrow T_{12} + Z_3$	$R_{32} \leftarrow R_{33} + R_{34}$
16.	$T_8 \leftarrow T_7 + T_5$	$r'_1 \leftarrow T'_4 + T_8$	$T_{17} \leftarrow x_2 + y_2$	$R_{35} \leftarrow R_{36} + R_{37}$
17.	$T_9 \leftarrow X_3 \times T_8$	$T_9 \leftarrow T_7 \times T_6$	$T_{18} \leftarrow T_{16} \times T_{15}$	$R_{38} \leftarrow R_{39} \times R_{40}$
18.	$T_{10} \leftarrow T_5 \times Z_3$	$r''_1 \leftarrow r_1 \times r'_1$	$T_{19} \leftarrow T_{14} \times T_{17}$	$R_{41} \leftarrow R_{42} \times R_{43}$
19.	$Y_3 \leftarrow T_9 + T_{10}$	$T'_8 \leftarrow R_1 + T_8$	$Y_3 \leftarrow T_{18} + T_{19}$	$R_{44} \leftarrow R_{45} + R_{46}$

ECADD2. In the case of $a = 1$, a similar process described by Table 2 is executed. A symbol '$\star$' within the atomic blocks represents a dummy operation, which involves executing a computational process using random variables, containing the same number of steps, so that the dummy operation and atomic block entail the same amounts of processing. As the atomic blocks are based on a unified fixed atomic pattern, attackers using SPA cannot determine the difference between ECADD and ECDBL computations; therefore, the proposed basic atomic blocks (Table 1 and Table 2) can be used as countermeasures against SPA.

3.2 Horizontal collision correlation analysis

In Section 3.1.2, we outline that atomic blocks can be used as countermeasures against SPA. However, a technique called Horizontal Collision Correlation Analysis (HCCA) [3] has been proposed as an attack on atomic blocks. The main idea of HCCA is that an attacker can distinguish between ECADD and ECDBL through a common operand used in an atomic block. In this section, we describe HCCA and discuss how the proposed basic atomic blocks can be applied to protect from HCCA attacks. The general idea of HCCA is proposed in [2] and its application to ECC is discussed in detail in [3].

3.2.1 HCCA attack methodology

The concept of HCCA is based on the assumption that *an adversary can identify when two field multiplications have at least one operand in common.* In other words, if two multiplications $\mathtt{A} \times \mathtt{B}$ and $\mathtt{C} \times \mathtt{D}$ are performed, an HCCA attack can identify whether any of the equations $\mathtt{A} = \mathtt{C}$, $\mathtt{A} = \mathtt{D}$, $\mathtt{B} = \mathtt{C}$, or $\mathtt{B} = \mathtt{D}$ can be executed. In this paper, we use the term "common operand" to refer to an operand that has the same value in the two field multiplication operations. In general, HCCA can be applied to the operations in which a common operand is used on the same side, for example, $\mathtt{A} \times \mathtt{B}$ and $\mathtt{C} \times \mathtt{B}$. However, in calculations such as $\mathtt{A} \times \mathtt{B}$ and $\mathtt{B} \times \mathtt{C}$, in which the common operand is not on the same side, it cannot be determined whether the common operand $\mathtt{B}$ exists [14, 15]. However, in this paper, considering the possibility of stronger side-channel attacks appearing in the future, we assume a threat model and the proposed basic atomic blocks represented in Table 1 and Table 2 to be constructed in such a way to ensure security under Assumption 1.

Assumption 1 *A side-channel attacker can determine whether common operands exist, even if they are not on the same side.*

Table 3: Description of HCCA

Step		ECDBL		ECADD	Atomic pattern
1.	1.	$T_4 \leftarrow X_1 \times X_1$	1.	$T_4 \leftarrow Z_2 \times Z_2$	$R_1 \leftarrow R_2 \times R_3$
2.		$T_5 \leftarrow T_4 + T_4$		$\star$	$R_4 \leftarrow R_5 + R_6$
3.		$\star$		$\star$	$R_7 \leftarrow -R_8$
4.		$\star$		$\star$	$R_9 \leftarrow R_{10} + R_{11}$
5.	2.	$T_5 \leftarrow T_3 \times T_3$	2.	$T_1 \leftarrow T_1 \times T_4$	$R_1 \leftarrow R_2 \times R_3$
6.		$T_5 \leftarrow T_4 + T_4$		$\star$	$R_4 \leftarrow R_5 + R_6$
7.		$\star$		$\star$	$R_7 \leftarrow -R_8$
8.		$\star$		$\star$	$R_9 \leftarrow R_{10} + R_{11}$
9.	3.	$T_5 \leftarrow T_5 \times T_5 (= Z_1^2 \times Z_1^2)$	3.	$T_4 \leftarrow T_4 \times T_9 (= Z_2^2 \times Z_2)$	$R_1 \leftarrow R_2 \times R_3$
10.		$\star$		$\star$	$R_4 \leftarrow R_5 + R_6$
11.		$\star$		$\star$	$R_7 \leftarrow -R_8$
12.		$\star$		$\star$	$R_9 \leftarrow R_{10} + R_{11}$
$\vdots$		$\vdots$		$\vdots$	$\vdots$

Chevallier-Mames *et al.* proposed an atomic block that can complete ECADD and ECDBL calculations in 10 and 16 iterations, respectively, corresponding to an atomic pattern comprising four

steps [4]. Table 3 represents the first three iterations of an atomic pattern based on the atomic block proposed by Chevallier-Mames *et al.* Here, when executing the ECADD column of Table 3, the first three atomic pattern iterations in the same ECADD contain the common operand Z_2. However, when computing the ECDBL column of Table 3, no common operands in the first three atomic patterns of the ECDBL column are presented. By focusing on this characteristic, an attacker can obtain two power traces: multiplication $\mathcal{P}_1$ occurring at step 1 and multiplication $\mathcal{P}_2$ at step 9 in the first and third atomic patterns, respectively, corresponding to the atomic block presented in Table 3. In this case, if the calculations corresponding to $\mathcal{P}_1$ and $\mathcal{P}_2$ have a common operand, the attacker can determine that the calculations $Z_2 \times Z_2$ and $Z_2^2 \times Z_2$ are executed at $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, and accordingly, that an ECADD operation is performed in this atomic block. Moreover, the calculations $X_1 \times X_1$ and $Z_2^2 \times Z_2$ are performed at $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, when the calculations corresponding to $\mathcal{P}_1$ and $\mathcal{P}_2$ do not have a common operand, implying that an ECDBL operation is performed in this atomic block. As mentioned in the explanation of Algorithm 3, an attacker that can distinguish between ECADD and ECDBL operations during ECSM computation can recover the secret information d accordingly.

3.2.2 Design of countermeasures against HCCA

As mentioned in the description of HCCA, it is possible to determine the operations being executed (ECADD or ECDBL) within the repeated range of a specific atomic pattern due to differences between the steps in which common operands appear. Therefore, as a countermeasure against HCCA, we need to apply an atomic block having the same calculation durations corresponding to the field multiplication operations with common operands while computing ECADD and ECDBL. Table

Table 4: Common operand in Algorithm 1 and Algorithm 2

	$a = 0$	$a = 1$
ECDBL	$T_1 : (3, 5), T_2 : (3, 4)$	$T_1 : (3, 4), T_2 : (3, 5)$
ECADD	$Z_1 : (1, 2, 4), X_3 : (4, 10),$ $T_1 : (7, 8, 11), Y_3 : (8, 12),$ $Z_3 : (15, 17), x_2 : (1, 15)$	$Z_1 : (1, 2, 4), X_3 : (4, 10),$ $T_1 : (7, 8), Y_3 : (8, 12),$ $Z_3 : (15, 17), x_2 : (1, 15)$

4 shows the common operands and their calculation times concerning Algorithm 1 and Algorithm 2 (ECDBL and ECADD) for each constant a. In Table 4, $T_1 : (3, 5)$, indicating the field multiplications in steps 3 and 5, has a common operand T_1. The ECDBL (Algorithm 1) and ECADD (Algorithm 2) operations considered in this paper have different numbers of common operands (Table 4). In this case, the number of field multiplications with the common operands in ECDBL and ECADD can be equalized by improving the atomic block, so as to reduce the number of common operands in the ECADD operations. Specifically, by introducing a random variable R, the calculation result $\mathtt{B} \times \mathtt{C}$ can be obtained without a need to execute the field multiplication $\mathtt{B} \times \mathtt{C}$ using a common operand. For example, when calculating $\mathtt{T_2} \leftarrow \mathtt{B} \times \mathtt{C}$ at a particular step of an atomic pattern, this step can be improved by executing the following six operations:

$$
\begin{aligned}
&\mathtt{B}' \leftarrow \mathtt{B} + \mathtt{R},\\
&\mathtt{C}' \leftarrow \mathtt{C} + \mathtt{R},\\
&\mathtt{B}'' \leftarrow \mathtt{B}' + \mathtt{C}\ (= \mathtt{B} + \mathtt{C} + \mathtt{R}),\\
&\mathtt{R}' \leftarrow \mathtt{B}'' \times \mathtt{R}\ (= \mathtt{R}(\mathtt{B} + \mathtt{C} + \mathtt{R})),\\
&\mathtt{T'}_2 \leftarrow \mathtt{B}' \times \mathtt{C}'\ (= \mathtt{BC} + \mathtt{R}(\mathtt{B} + \mathtt{C} + \mathtt{R})),\\
&\mathtt{T}_2 \leftarrow \mathtt{T}_2' + \mathtt{R}'\ (= \mathtt{BC}).
\end{aligned}
\tag{4}
$$

However, to implement this improvement, it is important to consider the possibility of increasing the computational cost of ECSM by adding an atomic pattern of roughly 5 $(= 6 - 1)$ steps. In our proposed two basic atomic blocks (Table 1 and Table 2), the variables influenced by HCCA applies

are marked in red. As the timings of the common operands in the proposed basic atomic blocks are unified, an HCCA attacker cannot distinguish between ECADD and ECDBL operations. The variable D_j within each proposed basic atomic block has the values for corresponding to the above operations (4), which are intentionally utilized to introduce the common operands.

3.3 Improved HCCA using Big Mac attack on ECC

Improved HCCA using a Big Mac Attack (IBMA) is another side-channel attack relying on common operands [6]. In this section, we provide a general introduction to IBMA and describe how it can be overcome using our basic atomic blocks.

3.3.1 IBMA methodology

As in the HCCA attack, we assume that an attacker can identify the presence of a common operand by examining the power trace obtained while performing two field multiplications. In IBMA, based on the calculation corresponding to the i-th bit d_i of secret information d, it is possible to distinguish between ECADD and ECDBL operations by analyzing the differences in the common operands between the cases when $d_i = 0$ and $d_i = 1$.

Here, we describe a method to recover $d_{\ell-2}$ from the calculations on the elliptic curve defined by (3) corresponding to an algorithm based on the ECADD, ECDBL, and ECSM operations. If we combine the first three atomic blocks of Algorithm 3, we obtain the following three possibilities regarding the value of $d_{\ell-2}$:

- ECDBL $\rightarrow$ ECADD1 $\rightarrow$ ECADD2,
  where $d_{\ell-2} = 1$,
- ECDBL $\rightarrow$ ECDBL $\rightarrow$ ECADD1,
  where $d_{\ell-2} = 0$ (and $d_{\ell-3} = 1$),
- ECDBL $\rightarrow$ ECDBL $\rightarrow$ ECDBL,
  where $d_{\ell-2} = 0$ (and $d_{\ell-3} = 0$).

On the basis of these combinations, we observe that the calculation that starts with the sequence ECDBL, ECADD1 indicates that bit $d_{\ell-2} = 1$, while any other sequence means that $d_{\ell-2} = 0$. Therefore, an attacker can recover bit $d_{\ell-2}$ when there are differences in the timing corresponding to the common operands in an atomic block between the case in which an ECDBL operation is followed by an ECADD1 operation and the case in which an ECDBL operation is followed by another ECDBL operation.

Here, b is the elliptic curve parameter presented in Equation (3) and Algorithm 1. Moreover, the point $P = (x_2, y_2)$ is a fixed point in Algorithm 3, so that the input value P of Algorithm 2 used in Algorithm 3 is also a fixed point.

3.3.2 Design of countermeasures against IBMA

One way of constructing atomic blocks that are resistant to IBMA is to perform field multiplication at the same step using a specified constant corresponding to the common operand in the ECADD and ECDBL parts of an atomic block. In this paper, the fixed values used in the calculations are the elliptic curve parameter b and the *affine* coordinates (x_2, y_2) of the fixed point P used in ECC. In Table 1 and Table 2, the steps using these fixed values in the field multiplications are unified within each atomic block. For example, although x_2 is not needed for the computation of ECDBL, we perform the dummy computation with x_2 at the step 2 of the column of ECDBL in Table 1. Therefore, even though the ECSM operation can be performed in any order within the atomic blocks, there are field multiplications using the same constant in the same step of each atomic block, making it difficult for the attacker to distinguish between ECADD and ECDBL. In the proposed basic atomic blocks (Table 1 and Table 2), the variables to which IBMA can be applied are marked in blue. Furthermore, similarly as in the case of HCCA, countermeasures can be implemented under the assumption of a stronger side-channel attacker compared to the attacker assumed in [6].

3.4 Differential power analysis

Differential Power Analysis (DPA) is a side-channel attack that is based on an approach that differs from those of SPA, HCCA, and IBMA. DPA is an effective attack implying that P is a fixed point on an elliptic curve corresponding to the computation of ECSM ($[d]P$). In this section, we describe the DPA attack method and discuss how the proposed algorithm can be used to protect from DPA attacks. DPA was first proposed by Kocher *et al.* [11], and can be used to recover the secret information from smart cards based on their power traces. In [11], DPA was proposed as an attack against the Data Encryption Standard (DES), which is a symmetric encryption algorithm. In [5], DPA was applied to ECC, implying that it can be used to recover a scalar d when computing $[d]P$.

3.4.1 DPA attack methodology

DPA aims to use power traces during the calculation of $[d]P$ to determine whether a specific point $[r_i]P$ is used in this calculation. Here, the expression corresponding to $P \in E/\mathbb{F}_{2^m}$ is assumed to be an *affine* coordinate. An attacker using DPA relies on the fact that each candidate $r_{\ell-2}, \ldots, r_0$ $(= d)$ has only two possible values. Then, the attacker performs DPA considering each candidate r_i to recover the secret information d. This is explained below, using the calculation process of Algorithm 3, according to which the calculation of $[d]P$ is performed first to derive $[2]P$, and then $[4]P$, if $d_{\ell-2} = 0$. However, when $d_{\ell-2} = 1$, the algorithm calculates $[3]P$ and then $[6]P$; therefore, calculation of $[4]P$ is not performed. Consequently, by using DPA to ascertain whether $r_{\ell-2}$ is equal to 4, it is possible to determine the value of $d_{\ell-2}$. Let us suppose that we know $d_{\ell-2} = 0$. In Algorithm 3, the next rational point to be calculated is $[8]P$ if $d_{\ell-3} = 0$, or $[5]P$ followed by $[10]P$ otherwise. Therefore, DPA can be used to ascertain whether $r_{\ell-3} = 8$, thereby determining the value of $d_{\ell-3}$. Thereafter, by repeating the same operations, the secret information d can be recovered.

3.4.2 Countermeasures against DPA

Three countermeasures against DPA are proposed in [5], one of which involves using randomized projective coordinates. In this method, *affine* coordinates are transformed into projective coordinates when performing ECADD and ECDBL calculations performed separately. In particular, the ECDBL calculations in ECSM are assumed to be performed using *affine* coordinates. The number of values of $[r_i]P$ used when executing a DPA attack in this case is at most $\mathcal{O}(\log_2 d)$ (not exceeding the bit length of d). However, when using projective coordinates, the number of values of $[r_i]P$ used in an attack has the order of $\mathcal{O}$ $(2^k \log_2 d)$. That is, using projective coordinates means that many $[r_i]P$ values have to be utilized to perform a DPA attack, making it difficult to identify whether $[r_i]P$ is included in the calculation of $[d]P$. In this paper, we use *LD* projective coordinates when calculating ECADD and ECDBL, as described in Section 2.1. Therefore, the proposed basic atomic blocks can be used as countermeasures against the DPA attack.

Using atomic blocks in the ECADD and ECDBL elliptic curve operations in a finite field $\mathbb{F}_{2^m}$ has already been proposed in [4], implying that an atomic block can act as a countermeasures against the SPA, HCCA, and IBMA attacks but not against the DPA attack. Moreover, this atomic block includes a finite field inversion. The computational cost of finding a multiplicative inverse in a finite field is very large compared to other ways of processing in a finite field and becomes a bottleneck for a practical implementation. However, the basic atomic blocks (Table 1 and Table 2) proposed in this paper do not include any finite field inversion operation. Furthermore, owing to the possibility of providing countermeasures against DPA, the proposed atomic blocks are considered superior to that proposed in [4].

4 Improved atomic block using NAF representation

Compared to Algorithm 3, we can reduce the computational cost associated with ECSM by applying the non-adjacent form (NAF) representation [7, Algorithm 3.31]. In this section, we describe an atomic block for the ECSM algorithm using the NAF representation.

4.1 NAF method

The NAF representation NAF(d) = $(d_{\ell-1}, \ldots, d_0)$ of a positive integer d satisfies the conditions $d = \sum_{i=0}^{\ell-1} d_i 2^i$, $d_i \in \{0, \pm 1\}$, $d_{\ell-1} \neq 0$, and for any $i \geq 0$, the product $d_i d_{i+1}$ is not zero. The NAF representation NAF(d) of a positive integer d can be obtained using Algorithm 4.

Algorithm 4 NAF representation [7, Algorithm 3.30]

Input: A positive integer d.
Output: NAF(d).
1: $i \leftarrow 0$.
2: **while** $d \geq 1$ **do**
3: **if** d is odd **then**
4: $d_i \leftarrow 2 - (d \bmod 4)$.
5: $d \leftarrow d - d_i$.
6: **else**
7: $d_i \leftarrow 0$.
8: **end if**
9: $d \leftarrow d/2$.
10: $i \leftarrow i + 1$.
11: **end while**
12: **return** $(d_{i-1}, d_{i-2}, \ldots, d_1, d_0)$.

NAF is known to possess the following characteristics [7, page 98]:

- d has a unique NAF, denoted by NAF(d).
- NAF(d) has the lowest number of nonzero digits of any signed digit representation of d.
- The length of NAF(d) is at most one more than that of the binary representation of d.
- The average density of nonzero digits in NAF is approximately 1/3.

The computational cost associated with ECSM generally decreases with a decline in the Hamming weight of particular representations (such as the binary representation and the NAF representation) of a scalar. As the average density of nonzero digits in the binary representation is 1/2, NAF is advantageous in terms of computational cost. The ECSM algorithm based on NAF requires executing an inverse operation of ECADD. Therefore, the computational cost associated with elliptic curve subtraction must not be too large, to avoid eliminating the advantage of NAF. In this paper, the subtraction $Q - P$ for rational points P and Q can be calculated as $Q + (-P)$ using an ECADD operation, including the negative point $-P = (x, x + y)$ of $P = (x, y)$, and therefore, has almost the same computational cost as that of ECADD. Algorithm 5 is defined to perform ECSM through NAF so that the functions ECADD Ⅰ and ECADD Ⅱ correspond to the computations of $Q + P$ and $Q - P$, respectively. In Section 4, we outline Table 5 as a new efficient atomic block based on Algorithm 5, defined by improving Table 1 and Table 2.

4.2 Improved atomic block

ECADD and ECDBL may have different calculation specifications depending on the value of the elliptic curve parameter $a \in \{0, 1\}$, and therefore, we propose two basic atomic blocks, Table 1 for the case $a = 0$ and Table 2 for $a = 1$, in Section 3. To reduce the data quantity processed in cryptographic devices, it is preferable to implement fewer atomic blocks implemented in these devices. In this section, we propose an atomic block, as represented in Table 5, that can handle both cases of $a = 0$ and $a = 1$. Moreover, we reduce the computational cost by applying the NAF representation.

The method for unifying the atomic blocks of Table 1 and Table 2 is described below. When we compute ECDBL for $a = 1$ by using Algorithm 1, the calculation $T_1 \leftarrow T_1 + Z_3$ is performed at

Algorithm 5 ECSM using NAF representation [7, Algorithm 3.31]

Input: A positive integer $d, P = (x_2, y_2) \in E/\mathbb{F}_{2^m}$.
Output: $[d]P$.
1: Compute NAF$(d) = \sum_{i=0}^{\ell-1} d_i 2^i, d_i \in \{0, \pm 1\}$.
2: Select $Z \in \mathbb{F}_{2^m}^*$ randomly.
3: $Q \leftarrow (xZ : yZ^2 : Z)$. ($Q$ in LD coordinate)
4: **for** i from $\ell - 1$ down to 0 **do**
5: $Q \leftarrow [2]Q$. (compute ECDBL with Q)
6: **if** $d_i = 1$ **then**
7: $Q \leftarrow Q + P$. (compute ECADD Ⅰ with P and Q)
8: **else if** $d_i = -1$ **then**
9: $Q \leftarrow Q - P$. (compute ECADD Ⅱ with P and Q)
10: **end if**
11: **end for**
12: **return** Q.

step 9, followed by the calculation $T_1 \leftarrow T_1 + T_2$ at step 10. However, when $a = 0$, the calculation $T_1 \leftarrow T_1 + Z_3$ is omitted, and only $T_1 \leftarrow T_1 + T_2$ is executed. Similarly, for computation of ECADD for $a = 1$ using Algorithm 2, calculations $T_1 \leftarrow T_1 + T_2$ at step 9 and $X_3 \leftarrow T_2 \times T_1$ at step 11 are performed; however, when $a = 0$, the calculation $T_1 \leftarrow T_1 + T_2$ is omitted, and only $X_3 \leftarrow T_2 \times T_1$ is calculated. In other words, the field additions required in ECADD and ECDBL when $a = 1$ are not performed when $a = 0$. Here, let a field addition $\mathtt{T_1} \leftarrow \mathtt{T_0} + \mathtt{A}$ be required when $a = 1$, but not when $a = 0$. This can be implemented by performing the following calculation so as to avoid inconsistencies both cases when $a = 0$ and when $a = 1$:

$$\begin{aligned} \mathtt{T_1} &\leftarrow \mathtt{T_0} + \mathtt{A}, \\ \mathtt{B} &\leftarrow (\mathtt{1} - \mathtt{a})\mathtt{T_0} + \mathtt{aT_1}. \end{aligned}$$

After completing this two-step calculation, $\mathtt{B}$ has the value of $\mathtt{T_0} + \mathtt{A}$ if $a = 1$, or of $\mathtt{T_0}$ if $a = 0$. Therefore, by performing the computation with $\mathtt{B}$, it is possible to switch to the required calculation results for $a = 0$ and $a = 1$.

4.2.1 Design of countermeasures against SPA

As mentioned in Section 3.1.1, an SPA attack can be used to distinguish between the ECADD and ECDBL operations, analyzing differences in the order of field multiplications that are processed on a device. It is, therefore, possible to counteract SPA by configuring an atomic block that unifies the calculation specifications of both ECADD and ECDBL calculations, as discussed in Section 3.1.2. The algorithm for computing ECSM by NAF, Algorithm 5, needs three computations: ECDBL, ECADD, and the elliptic curve subtraction (ECSUB). It is, therefore, necessary to construct an atomic block that can perform all these calculations. In Table 5, each atomic block is configured such that ECADD Ⅰ corresponding to ECADD, ECADD Ⅱ for an ECSUB, and ECDBL calculations can be performed within the same atomic pattern. Table 5 can, therefore, serve as a countermeasure against SPA.

4.2.2 Design of countermeasures against HCCA

As explained in Section 3.2.1, the ECADD and ECDBL calculations can be distinguished by applying an HCCA attack based on analyzing the differences in the step involving field multiplications using common operands in the iterative part of a particular atomic pattern. HCCA can, therefore, be counteracted by an atomic block in which the field multiplications with common operands appear at the same step, as discussed in Section 3.2.2. The computation of ECSM using Algorithm 5 requires executing ECADD Ⅰ, ECADD Ⅱ, and ECDBL. Therefore, the steps involving field multiplications with common operands must be unified in all atomic blocks corresponding to ECADD Ⅰ, ECADD

Table 5: Our improved atomic block for $a \in \{0, 1\}$ based on the NAF method

Step	Pattern	ECDBL	ECADD Ⅰ-1	ECADD Ⅰ-2	ECADD Ⅱ-1	ECADD Ⅱ-2
1	$\times^2$	$T_1 \leftarrow Z_1^2$	$\star$	$\star$	$\star$	$\star$
2	$\times^2$	$T_4 \leftarrow T_1^2$	$T_2 \leftarrow Z_1^2$	$Z_3 \leftarrow T_4^2$	$T_2 \leftarrow Z_1^2$	$Z_3 \leftarrow T_4^2$
3	$+$	$\star$	$Z_1^{'} \leftarrow Z_1 + R_1$	$Z_3^{'} \leftarrow Z_3 + R_2$	$Z_1^{'} \leftarrow Z_1 + R_1$	$Z_3^{'} \leftarrow Z_3 + R_2$
4	$\times$	$D_1 \leftarrow D_1 \times x_2$	$R_1^{'} \leftarrow R_1 \times x_2$	$D_5 \leftarrow D_5 \times x_2$	$R_1^{'} \leftarrow R_1 \times x_2$	$D_5 \leftarrow D_5 \times x_2$
5	$\times$	$D_2 \leftarrow D_2 \times x_2$	$T_1^{'} \leftarrow Z_1^{'} \times x_2$	$T_{13} \leftarrow Z_3 \times x_2$	$T_1^{'} \leftarrow Z_1^{'} \times x_2$	$T_{13} \leftarrow Z_3 \times x_2$
6	$\times^2$	$T_2 \leftarrow X_1^2$	$\star$	$T_{14}^{'} \leftarrow Z_3^{'2}$	$\star$	$T_{14}^{'} \leftarrow Z_3^{'2}$
7	$+$	$\star$	$T_1 \leftarrow T_1^{'} + R_1^{'}$	$T_{14} \leftarrow T_{14}^{'} + R_2^{'}$	$T_1 \leftarrow T_1^{'} + R_1^{'}$	$T_{14} \leftarrow T_{14}^{'} + R_2^{'}$
8	$+$	$\star$	$T_3 \leftarrow X_1 + T_1$	$T_{17}^{'} \leftarrow T_{17} + R_5$	$T_3 \leftarrow X_1 + T_1$	$T_{17}^{'} \leftarrow y_2 + R_5$
9	$\times$	$T_5 \leftarrow T_4 \times b$	$D_4 \leftarrow D_4 \times b$	$D_6 \leftarrow D_6 \times b$	$D_4 \leftarrow D_4 \times b$	$D_6 \leftarrow D_6 \times b$
10	$\times^2$	$T_3 \leftarrow T_2^2$	$T_7 \leftarrow T_3^2$	$T_{10} \leftarrow T_8^2$	$T_7 \leftarrow T_3^2$	$T_{10} \leftarrow T_8^2$
11	$\times$	$Z_3 \leftarrow T_1 \times T_2$	$T_4 \leftarrow Z_1 \times T_3$	$T_{12} \leftarrow T_4 \times T_8$	$T_4 \leftarrow Z_1 \times T_3$	$T_{12} \leftarrow T_4 \times T_8$
12	$\times^2$	$T_6 \leftarrow Y_1^2$	$R_2^{'} \leftarrow R_2^2$	$\star$	$R_2^{'} \leftarrow R_2^2$	$\star$
13	$+$	$\star$	$T_{17} \leftarrow x_2 + y_2$	$T_9 \leftarrow T_9^{'} + R_3^{''}$	$T_{17} \leftarrow x_2 + y_2$	$T_9 \leftarrow T_9^{'} + R_3^{''}$
14	$+$	$\star$	$y_2^{'} \leftarrow y_2 + R_4$	$\star$	$y_2^{'} \leftarrow T_{17} + R_4$	$\star$
15	$\times$	$D_3 \leftarrow D_1 \times D_2$	$T_5^{'} \leftarrow T_2 \times y_2^{'}$	$T_{19}^{'} \leftarrow T_{14} \times T_{17}^{'}$	$T_5^{'} \leftarrow T_2 \times y_2^{'}$	$T_{19}^{'} \leftarrow T_{14} \times T_{17}^{'}$
16	$\times$	$D_2 \leftarrow D_1 \times D_3$	$T_2^{'} \leftarrow T_2 \times R_4$	$T_{14}^{'} \leftarrow T_{14} \times R_5$	$T_2^{'} \leftarrow T_2 \times R_4$	$T_{14}^{'} \leftarrow T_{14} \times R_5$
17	$+$	$\star$	$T_5 \leftarrow T_2^{'} + T_5^{'}$	$T_{19} \leftarrow T_{19}^{'} + T_{14}^{'}$	$T_5 \leftarrow T_2^{'} + T_5^{'}$	$T_{19} \leftarrow T_{19}^{'} + T_{14}^{'}$
18	$+$	$X_3 \leftarrow T_3 + T_5$	$T_4^{'} \leftarrow T_4 + T_2$	$T_{11} \leftarrow T_9 + T_{10}$	$T_4^{'} \leftarrow T_4 + T_2$	$T_{11} \leftarrow T_9 + T_{10}$
19	$+$	$T_7 \leftarrow T_6 + Z_3$	$T_7^{'} \leftarrow T_7 + R_3$	$X_3 \leftarrow T_{11} + T_{12}$	$T_7^{'} \leftarrow T_7 + R_3$	$X_3 \leftarrow T_{11} + T_{12}$
20	Choice	$C_1 \leftarrow (1-a)T_6 + aT_7$	$C_2 \leftarrow (1-a)T_4 + aT_4^{'}$	$C_3 \leftarrow (1-a)D_6 + aD_7$	$C_2 \leftarrow (1-a)T_4 + aT_4^{'}$	$C_3 \leftarrow (1-a)D_6 + aD_7$
21	$+$	$\star$	$T_4^{''} \leftarrow C_7 + R_3$	$T_{15} \leftarrow T_{13} + X_3$	$T_4^{''} \leftarrow C_7 + R_3$	$T_{15} \leftarrow T_{13} + X_3$
22	$+$	$T_8 \leftarrow C_1 + T_5$	$R_3^{'} \leftarrow T_4^{''} + T_7$	$T_{16} \leftarrow T_{12} + Z_3$	$R_3^{'} \leftarrow T_4^{''} + T_7$	$T_{16} \leftarrow T_{12} + Z_3$
23	$\times$	$T_9 \leftarrow X_3 \times T_8$	$T_9^{'} \leftarrow T_4^{''} \times T_7^{'}$	$T_{18} \leftarrow T_{16} \times T_{15}$	$T_9^{'} \leftarrow T_4^{''} \times T_7^{'}$	$T_{18} \leftarrow T_{16} \times T_{15}$
24	$\times$	$T_{10} \leftarrow T_5 \times Z_3$	$R_3^{''} \leftarrow R_3 \times R_3^{'}$	$\star$	$R_3^{''} \leftarrow R_3 \times R_3^{'}$	$\star$
25	$+$	$Y_3 \leftarrow T_9 + T_{10}$	$T_8 \leftarrow Y_1 + T_5$	$Y_3 \leftarrow T_{18} + T_{19}$	$T_8 \leftarrow Y_1 + T_5$	$Y_3 \leftarrow T_{18} + T_{19}$

Ⅱ, and ECDBL, and therefore, we configure the atomic block, Table 5, in such a way to satisfy this condition. Note that the operands corresponding to HCCA are marked in red, in Table 5, and thus, this table can be considered as a countermeasure against HCCA.

4.2.3 Design of countermeasures against IBMA

In Section 3.3.1, it is explained that IBMA can be used if there exists field multiplication with common operands, that only differ between certain atomic block iterations in the overall ECSM computation. A countermeasure against IBMA can be implemented by introducing the steps in which field multiplications using common operands appear, regardless of the order in which the ECADD and ECDBL atomic blocks are calculated in the overall ECSM calculation, as mentioned in Section 3.3.2. In Table 5, regardless of the order according to which the atomic blocks are calculated, the step in which field multiplications are performed using common operands is the same in each atomic block. Note that the operands related to IBMA are marked in blue in Table 5. Table 5 can be, therefore, considered a countermeasure against IBMA.

4.2.4 Design of countermeasures against DPA

As explained in Section 3.4.1, ECSM calculations are performed using rational points represented by specific coordinates, and the information of a scalar can be leaked by DPA. Therefore, a possible countermeasure against DPA is to utilize the randomized projective coordinate method, in which a

rational point is converted into the coordinates associated with random numbers. This countermeasure is described in detail in Section 3.4.2 and is introduced at step 3 of Algorithm 5. Table 5 can, therefore, serve as a countermeasure against DPA.

5 Computational cost of the proposed atomic block

In this section, we estimate the computational costs associated with the two ECSM algorithms (Algorithm 3 and Algorithm 5) based on ECADD/ECDBL algorithms. **EA** and **ED** denote the computational costs relying on ECADD and ECDBL, respectively. The computational cost of ECSM based on Algorithm 3 is defined as follows:

$$\frac{\ell}{2} \cdot \mathbf{EA} + \ell \cdot \mathbf{ED}, \tag{5}$$

where ℓ indicates the bit length of secret information d. Similarly, the computational cost of ECSM based on Algorithm 5 is defined as follows:

$$\frac{l}{3} \cdot \mathbf{EA} + l \cdot \mathbf{ED}, \tag{6}$$

where l indicates the length of NAF representation NAF(d) of secret information d. As the length of NAF(d) is at most one larger than that of the binary representation of d, we assume that $\ell \approx l$. Symbols $\mathcal{A}$ and $\mathcal{M}$ denote the computational costs associated with field addition and field multiplication, respectively. It is well known that the computational cost $\mathcal{A}$ is negligible compared to that of $\mathcal{M}$ ($\mathcal{A} \ll \mathcal{M}$). Therefore, it is sufficient to estimate the number of field multiplications in Algorithm 1, Algorithm 2, as well as Table 1, Table 2, and Table 5; the corresponding numbers are represented in Table 6. Moreover, Table 6 represents the total computational costs of the two considered ECSM algorithms, Algorithm 3 and Algorithm 5, based on Algorithm 1, Algorithm 2, and Table 1, Table 2, and Table 5, respectively.

Table 6: Computational cost of each atomic block and the overall computational cost of ECSM

		Computational cost		ECSM algorithm	
		EA	**ED**	Algorithm 3	Algorithm 5
ECADD and ECDBL algorithms	Algorithm 1 Algorithm 2	$13\mathcal{M}$	$9\mathcal{M}$	$\frac{31}{2}\ell\mathcal{M}$	$\frac{40}{3}\ell\mathcal{M}$
	Table 1	$24\mathcal{M}$	$12\mathcal{M}$	$24\ell\mathcal{M}$	N/A
	Table 2	$24\mathcal{M}$	$12\mathcal{M}$	$24\ell\mathcal{M}$	N/A
	Table 5	$26\mathcal{M}$	$13\mathcal{M}$	N/A	$\frac{65}{3}\ell\mathcal{M}$

In the case of combing Algorithm 3 with those described in Table 1 and Table 2, the computational cost is approximately 1.8 times larger than that incurred when combining Algorithm 5 with Algorithm 1 and Algorithm 2. However, when we merge Algorithm 5 with that presented in Table 5, the computational cost is approximately 1.625 times larger than that of combining Algorithm 5 with Algorithm 1 and Algorithm 2. In other words, the atomic block provided in Table 5 can

execute ECSM calculations more efficiently compared to those corresponding to Table 1 and Table 2. When we apply Algorithm 5 to the proposed atomic blocks (Table 1, Table 2, and Table 5), the computational cost increases from approximately 1.625 to 1.8 times compared to the case when Algorithm 5 is merged with Algorithm 1 and Algorithm 2. However, the proposed atomic blocks can protect the four types of side-channel attacks: SPA, HCCA, IBMA, and DPA.

6 Future work

This paper proposed implementing three atomic blocks (Table 1, Table 2, Table 5) as the countermeasures against side-channel attacks using power consumption against ECC on a finite field $\mathbb{F}_{2^m}$. Furthermore, the proposed algorithms are effective even when assuming the powerful side-channel attacks specified in this paper. Among the proposed atomic blocks, the atomic block represented in Table 5 has the lowest computational cost of associated with ECSM.

Our future works are as follows:

- As a step toward the practical use of the atomic block in Table 5, it is desirable to evaluate the computational cost of that atomic block by a simulation or by using a software or hardware implementation.
- When we compute the ECSM for a secret information d used in ECC by Algorithm 5 and Table 5, the Hamming weight of d can be computed from the number of iterations of performing Table 5, namely, that Hamming weight might leak. Since there is a width-w NAF method proposed by Okeya and Takagi [13] as a countermeasure against this problem, it is desirable to construct atomic blocks that do not leak the Hamming weight of d by incorporating the method of [13] into the atomic block in Table 5.
- In this paper, we proposed atomic blocks which are countermeasures against four side-channel attacks (SPA, HCCA, IBMA, DPA) using power consumption. However, in addition to power consumption, there are many kinds of side-channel attacks like using electromagnetic waves or processing time. Therefore, new countermeasures that combine our proposed atomic block with countermeasures against these side-channel attacks are needed.

References

[1] Rodrigo Abarzúa, Claudio Valencia, and Julio López. Survey for performance & security problems of passive side-channel attacks countermeasures in ecc. Cryptology ePrint Archive, Report 2019/010, 2019. `https://eprint.iacr.org/2019/010`.

[2] Aurélie Bauer, Eliane Jaulmes, Emmanuel Prouff, and Justine Wild. Horizontal and vertical side-channel attacks against secure rsa implementations. In Ed Dawson, editor, *Topics in Cryptology – CT-RSA 2013*, pages 1–17, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[3] Aurélie Bauer, Eliane Jaulmes, Emmanuel Prouff, and Justine Wild. Horizontal collision correlation attack on elliptic curves. In Tanja Lange, Kristin Lauter, and Petr Lisoněk, editors, *Selected Areas in Cryptography – SAC 2013*, pages 553–570, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[4] B. Chevallier-Mames, M. Ciet, and M. Joye. Low-cost solutions for preventing simple side-channel analysis: side-channel atomicity. *IEEE Transactions on Computers*, 53(6):760–768, June 2004.

[5] Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Çetin K. Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 1999*, pages 292–302, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[6] Jean-Luc Danger, Sylvain Guilley, Philippe Hoogvorst, Cédric Murdica, and David Naccache. Improving the big mac attack on elliptic curve cryptography. In Peter Y. A. Ryan, David Naccache, and Jean-Jacques Quisquater, editors, *The New Codebreakers*, pages 374–386, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[7] Darrel Hankerson, Alfred J. Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag, Berlin, Heidelberg, 2003.

[8] Cameron F. Kerry, Acting Secretary, and Charles Romine Director. Fips pub 186-4 federal information processing standards publication digital signature standard (DSS), 2013.

[9] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, January 1987.

[10] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO' 99*, pages 388–397, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[11] Paul Kocher, Joshua Jaffe, Benjamin Jun, and Pankaj Rohatgi. Introduction to differential power analysis. *Journal of Cryptographic Engineering*, 1(1):5–27, April 2011.

[12] Victor S. Miller. Use of elliptic curves in cryptography. In Hugh C. Williams, editor, *Advances in Cryptology – CRYPTO '85 Proceedings*, pages 417–426, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

[13] Katsuyuki Okeya and Tsuyoshi Takagi. The width-w NAF method provides small memory and fast elliptic scalar multiplications secure against side channel attacks. In Marc Joye, editor, *Topics in Cryptology – CT-RSA 2003*, pages 328–343, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[14] Debdeep Mukhopadhyay Poulami Das, Debapriya Basu Roy. Exploiting the order of multiplier operands: A low cost approach for hcca resistance. Cryptology ePrint Archive, Report 2015/925, 2015. `https://eprint.iacr.org/2015/925`.

[15] Takeshi Sugawara, Daisuke Suzuki, and Minoru Saeki. Two operands of multipliers in side-channel attack. In *Revised Selected Papers of the 6th International Workshop on Constructive Side-Channel Analysis and Secure Design*, volume 9064, Berlin, Heidelberg, 2015. Springer-Verlag.

[16] Yusuke Takemura, Keisuke Hakuta, and Naoyuki Shinohara. ECC atomic block against strong side-channel attacks using binary curves. In *2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW)*, pages 387–393, November 2019.

[17] C. D. Walter. Sliding windows succumbs to big mac attack. In Çetin K. Koç, David Naccache, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, pages 286–299, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.