Low-Cost Countermeasure against RPA

Jean-Luc Danger[1,2], Sylvain Guilley[1,2], Philippe Hoogvorst[2], Cédric Murdica[1,2], and David Naccache[3]

[1] Secure-IC S.A.S., 80 avenue des Buttes de Coësmes, F-35700 Rennes, France
{jean-luc.danger,sylvain.guilley,cedric.murdica}@secure-ic.com
[2] Département COMELEC, Institut TELECOM, TELECOM ParisTech, CNRS LTCI, Paris, France
{jean-luc.danger,sylvain.guilley,philippe.hoogvorst, cedric.murdica}@telecom-paristech.fr
[3] École normale supérieure, Département d'informatique 45, rue d'Ulm, F-75230, Paris Cedex 05, France
david.naccache@ens.fr

Abstract. On smart-cards, Elliptic Curve Cryptosystems (ECC) can be vulnerable to Side Channel Attacks such as the Refined Power Analysis (RPA). This attack takes advantage of the apparition of special points of the form $(0, y)$. In this paper, we propose a new countermeasure based on co-Z formulæ and an extension of the curve isomorphism countermeasure. It permits to transform the base point $P = (x, y)$ into a base point $P' = (0, y')$, which, with $-P'$, are the only points with a zero X-coordinate. In such case, the RPA cannot be applied. Moreover, the cost of this countermeasure is very low compared to other countermeasures against RPA.

Keywords: Elliptic Curve Cryptosystem, Co-Z formulæ, Differential Power Analysis, Refined Power Analysis.

1 Introduction

The use of elliptic curves for cryptographic applications has been introduced by Koblitz [17] and Miller [23]. Elliptic Curve Cryptosystems (ECC) have gained much importance in smart-cards devices because of their better speed and low memory constraints compared to other asymmetric cryptosystems such as RSA.

The main operation on ECC is the computation of an elliptic curve scalar multiplication (ECSM), that is the computation of $[d]P$ for an integer d and a point P on an elliptic curve. The cryptographic security of ECC is based on the elliptic curve discrete logarithm problem (ECDLP), which asks to compute d given $Q = [d]P$ and P.

An ECSM is generally based on addition and doubling formulæ of points. Meloni points that addition formulæ of two points of an elliptic curve is more efficient if they share the same Z-coordinate [21]. He brought new formulæ, called co-Z formulæ, that can be used to perform an ECSM with addition chains and Zeckendorf representation.

Meloni's formulæ were adapted in [8,10,9] so that they might be usable with traditional ECSM algorithms such as the right-to-left signed-digit method, the Montgomery Powering Ladder [16], or the Joye's double-add method [14].

In this paper, we are interested on the security against Side Channel Attacks. We present alternative co-Z formulæ using an extension of the curve isomorphism countermeasure [15]. The new co-Z formulæ allow to perform an ECSM that can be secured against SPA [18], DPA [19] and RPA [7] attacks. We give a comparison of different countermeasures against RPA. Our countermeasure has a very low cost compared to the other countermeasures.

The rest of the paper is structured as follows. In Section 2, we describe some properties on elliptic curves arithmetic and ECSM algorithms. In Section 3, we recall on the different side channel attacks, especially the RPA. Section 3 also gives countermeasures against RPA. Section 4 describes our countermeasure based on modified co-Z formulæ and an extension of the curve isomorphism countermeasure. We give a comparison of different countermeasures against the RPA in Section 5. Finally, we conclude in Section 6.

2 Elliptic Curve Arithmetic

In this paper, we are interested in elliptic curves based on field with characteristic greater than 3, and the given elliptic curves are in the reduced Weierstraß form.

However, our proposed countermeasure transforms the curve given into another one that is not in its short Weierstraß form. This is why we also give the formulæ for elliptic curve in the general Weierstraß form to understand our modified formulæ of Section 4.

2.1 Elliptic Curve Arithmetic in the Affine Coordinates System

In a finite field $\mathbb{K}$, an elliptic curve can be described by its Weierstraß form:

$$E: \ y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \ .$$

We denote by $E(\mathbb{K})$ the set of points $(x, y) \in \mathbb{K}^2$ satisfying the equation, plus the point at infinity $\mathcal{O}$. $E(\mathbb{K})$ has an Abelian group structure. Let $P = (x_1, y_1) \neq \mathcal{O}$ and $Q = (x_2, y_2) \notin \{\mathcal{O}, -P\}$ two points in $E(\mathbb{K})$. The point $R = (x_3, y_3) = P+Q$ can be computed as:

$$\begin{aligned} x_3 &= \lambda^2 + a_1\lambda - a_2 - x_1 - x_2 \\ y_3 &= \lambda(x_1 - x_3) - y_1 - a_1x_3 - a_3 \end{aligned} \qquad \text{where } \lambda = \begin{cases} \frac{y_1 - y_2}{x_1 - x_2} & \text{if } P \neq Q, \\ \frac{3x_1^2 + 2a_2x_1 + a_4 - a_1y_1}{2y_1 + a_1x_1 + a_3} & \text{if } P = Q. \end{cases}$$

The inverse of the point P is $-P = (x_1, -y_1 - a_1x_1 - a_3)$.

In a finite field $\mathbb{F}_p$, with p a prime such that $p > 3$, an elliptic curve can be described by its short Weierstraß form:

$$E:\ y^2 = x^3 + ax + b\ .$$

The point $R = (x_3, y_3) = P + Q$ can be computed as:

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \\ y_3 &= \lambda(x_1 - x_3) - y_1 \end{aligned} \quad \text{where } \lambda = \begin{cases} \frac{y_1 - y_2}{x_1 - x_2} & \text{if } P \neq Q, \\ \frac{3x_1^2 + a}{2y_1} & \text{if } P = Q. \end{cases}$$

The inverse of the point P is $-P = (x_1, -y_1)$.

2.2 Elliptic Curve Arithmetic in the Jacobian Projective Coordinates System

To avoid costly inversions, one can use the Jacobian projective coordinates system. The equation of an elliptic curve in the Jacobian projective coordinates system in the reduced Weierstraß form is:

$$E^{\mathcal{J}}:\ Y^2 = X^3 + aXZ^4 + bZ^6\ .$$

The point (X, Y, Z) corresponds to the affine point $(X/Z^2, Y/Z^3)$.

We give addition (ECADD) and doubling (ECDBL) formulæ in the Jacobian projective coordinates system. The formulæ are from [3].

Algorithm 1. ecdbl

```
Input: $P = (X_1, Y_1, Z_1) \in E^{\mathcal{J}}(\mathbb{F}_p)$
Output: $2P$

  $A \leftarrow X_1^2;\ B \leftarrow Y_1^2$
  $C \leftarrow B^2;\ D \leftarrow Z_1^2$
  $S \leftarrow 2((X_1 + B)^2 - A - C)$
  $M \leftarrow 3A + aD^2$
  $X_3 \leftarrow M^2 - 2S$
  $Y_3 \leftarrow M(S - X_3) - 8C$
  $Z_3 \leftarrow (Y_1 + Z_1)^2 - B - D$

  return $(X_3, Y_3, Z_3)$
```

Algorithm 2. ecadd

```
Input: $P = (X_1, Y_1, Z_1), Q = (X_2, Y_2, Z_2) \in E^{\mathcal{J}}(\mathbb{F}_p)$
Output: $P + Q$
  $A \leftarrow Z_1^2;\ B \leftarrow Z_2^2$
  $U_1 \leftarrow X_1 B;\ U_2 \leftarrow X_2 A$
  $S_1 \leftarrow Y_1 Z_2 B;\ S_2 \leftarrow Y_2 Z_1 A$
  $H \leftarrow U_2 - U1$
  $I \leftarrow (2H)^2$
  $J \leftarrow HI;\ K \leftarrow 2(S_2 - S_1);\ V \leftarrow U_1 I$
  $X_3 \leftarrow K^2 - J - 2V$
  $Y_3 \leftarrow K(V - X_3) - 2S_1 J$
  $Z_3 \leftarrow ((Z_1 + Z_2)^2 - A - B)H$
  return $(X_3, Y_3, Z_3)$
```

We denote by M, S the cost of field multiplication and field squaring respectively. We neglect the cost of additions and subtractions. ECDBL can be performed in $2M + 8S$ and ECADD can be performed in $11M + 5S$. Mixed addition (mECADD) is the addition of a point in Jacobian coordinates with a point in affine coordinates ($Z_2 = 1$). mECADD can be performed in $7M + 4S$ [3].

2.3 Elliptic Curve Arithmetic Using co-Z Formulæ

We describe here addition formulæ with points sharing the same Z-coordinate. Two procedures are presented. Addition and update in co-Z (ZADDU) is the procedure to compute $P+Q$ and update the point P to get the same Z-coordinate. It was introduced in [21]. Conjugate addition in co-Z (ZADDC) is the procedure to compute $P+Q$ and $P-Q$. It was introduced in [8].

Algorithm 3. co-Z addition and update (ZADDU)

```
Input: $P=(X_1,Y_1,Z), Q=(X_2,Y_2,Z) \in E^{\mathcal{J}}(\mathbb{F}_p)$
Output: $(R,S)$ with $R = P+Q$ and $S = (\lambda^2 X_1, \lambda^3 Y_1, \lambda Z)$ with $\lambda = X_1 - X_2$

  $C \leftarrow (X_1 - X_2)^2$
  $W_1 \leftarrow X_1 C$; $W_2 \leftarrow X_2 C$; $Z_3 \leftarrow Z(X_1 - X_2)$
  $D \leftarrow (Y_1 - Y_2)^2$; $A_1 \leftarrow Y_1(W_1 - W_2)$
  $X_3 \leftarrow D - W_1 - W_2$
  $Y_3 \leftarrow (Y_1 - Y_2)(W_1 - X_3) - A_1$
  $X_4 \leftarrow W_1$
  $Y_4 \leftarrow A_1$
  return $((X_3,Y_3,Z_3),(X_4,Y_4,Z_3))$
```

Algorithm 4. conjugate co-Z addition (ZADDC)

```
Input: $P=(X_1,Y_1,Z), Q=(X_2,Y_2,Z) \in E^{\mathcal{J}}(\mathbb{F}_p)$
Output: $(R,S)$ with $R = P+Q$, $S = P-Q$

  $C \leftarrow (X_1 - X_2)^2$
  $W_1 \leftarrow X_1 C$; $W_2 \leftarrow X_2 C$; $Z_3 \leftarrow Z(X_1 - X_2)$
  $D_1 \leftarrow (Y_1 - Y_2)^2$; $A_1 \leftarrow Y_1(W_1 - W_2)$
  $X_3 \leftarrow D_1 - W_1 - W_2$
  $Y_3 \leftarrow (Y_1 - Y_2)(W_1 - X_3) - A_1$
  $D_2 \leftarrow (Y_1 + Y_2)^2$
  $X_4 \leftarrow D_2 - W_1 - W_2$
  $Y_4 \leftarrow (Y_1 + Y_2)(W_1 - X_4) - A_1$
  return $((X_3,Y_3,Z_3),(X_4,Y_4,Z_3))$
```

Goundar et al. proposed in [9] an optimisation by removing the useless computation of the Z-coordinate. The formulæ are called the (X,Y)-only co-Z formulæ (ZACAU')[1]. ZACAU' (algorithm 15 in appendix) is a procedure computing the point $2P$ and $P+Q$. It can be performed in $8M+6S$ [9].

2.4 Elliptic Curve Scalar Multiplication

In elliptic curve cryptography, one has to compute scalar multiplications, *i.e.* compute $[d]P$, given the point P and a positive integer d.

The Montgomery Ladder is regular since the same operations are performed at each iteration independently of the current bit. Therefore it can be used to prevent the SPA. The Montgomery Ladder can be adapted with co-Z formulæ.

Algorithm 5. Montgomery Ladder

```
Input: $P \in E^{\mathcal{J}}(\mathbb{F}_p), d=(d_{n-1},\dots,d_0)_2, d_{n-1}=1$
Output: $[d]P$
  $R_0 \leftarrow P, R_1 \leftarrow 2P$
  for $i = n-2$ downto $0$ do
     $R_{1-d_i} \leftarrow$ ECADD$(R_{1-d_i}, R_{1-d_i})$
     $R_{d_i} \leftarrow$ ECDBL$(R_{d_i})$
  end for
  return $R_0$
```

Algorithm 6. add only Montgomery Ladder using co-Z formulæ [8]

```
Input: $P \in E^{\mathcal{J}}(\mathbb{F}_p), d=(d_{n-1},\dots,d_0)_2, d_{n-1}=1$
Output: $[d]P$
  $R_0 \leftarrow P, R_1 \leftarrow 2P$
  for $i = n-2$ downto $0$ do
     $(R_{1-d_i}, R_{d_i}) \leftarrow$ ZADDC$(R_{d_i}, R_{1-d_i})$
     $(R_{d_i}, R_{1-d_i}) \leftarrow$ ZADDU$(R_{1-d_i}, R_{d_i})$
  end for
  return $R_0$
```

Remark 1. In algorithm 6, the output point R_{d_i} of ZADDC is always equal to $\pm P$. Indeed, at the end of each iteration of the algorithm, R_0 and R_1 verify $R_1 = R_0 + P$.

[1] We use the same notation as in [9]: (') stands for formulæ that does not involve the Z-coordinate.

Using ZACAU', ZADDC' and ZADDU', the add only Montgomery Ladder using co-Z formulæ can be improved. See [9] for the justification to recover the Z coordinate.

Algorithm 7. Montgomery Ladder with (X, Y)-only co-Z formulæ [9]

```
Input: $P \in E^{\mathcal{J}}(\mathbb{F}_p), d = (d_{n-1}, \ldots, d_0)_2, d_{n-1} = 1$
Output: $[d]P$

  $R_0 \leftarrow P, R_1 \leftarrow 2P$
  $C \leftarrow (X_{R_0} - X_{R_1})^2$
  for $i = n - 2$ downto 1 do
      $(R_{d_i}, R_{1-d_i}, C) \leftarrow$ ZACAU'$(R_{d_i}, R_{1-d_i}, C)$
  end for
  $b \leftarrow d_0$; $(R_{1-b}, R_b) \leftarrow$ ZADDC'$(R_b, R_{1-b})$
  $Z \leftarrow x_P Y_{R_b}(X_{R_0} - X_{R_1})$; $\lambda \leftarrow y_P X_{R_b}$
  $(R_b, R_{1-b}) \leftarrow$ ZADDU'$(R_{1-b}, R_b)$

  return $\left((\frac{\lambda}{Z})^2 X_{R_0}, (\frac{\lambda}{Z})^3 Y_{R_0}\right)$
```

3 Side Channel Attacks

We describe in this section passive attacks such as DPA, RPA and ZPA.

3.1 DPA Attack and Countermeasures

If the same scalar d is used several times, the implementation can be vulnerable to the DPA [19]. The attacker recursively guesses the bits of the scalar and simulates the computation.

The countermeasures given below can be used to prevent the DPA.

Random Projective Coordinates [6]. A point $P = (X, Y, Z)$ in Jacobian coordinates is equivalent to any point (r^2X, r^3Y, rZ), with $r \in \mathbb{F}_p^*$. One can randomize the base point at the beginning of the ECSM by choosing a random r.

Random Curve Isomorphism [15]. A curve E defined by $E : y^2 = x^3 + ax + b$ in affine coordinates is isomorphic to the curve E' defined by $E' : y^2 = x^3 + a'x + b'$ if and only if there exists $u \in \mathbb{F}_p^*$ such that $u^4a' = a$ and $u^6b' = b$. The isomorphism φ is defined as:

$$\varphi : E \xrightarrow{\sim} E', \begin{cases} \mathcal{O} \to \mathcal{O} \\ (x, y) \to (u^{-2}x, u^{-3}y) \end{cases}$$

The countermeasure consists of computing the ECSM on a random curve E' instead of E.

Scalar Randomization [6]. Randomization of the scalar using $d' = d + r\sharp E$ is effective against DPA. r must be at least 32 bits, because attacks have been pointed out in [24] if r is small. For this reason, the ECSM is 32 iterations longer.

Random Scalar Split [5]. Random scalar splitting, such as computing $Q = [d_1]P + [d_2]P$ with $d = d_1 + d_2$, is effective against DPA. One can also use the euclidian splitting method [5]: compute $Q = [d_1]P + [d_2]S$ with $d_1 = d \bmod r$, $d_2 = \lfloor d/r \rfloor$ and $S = [r]P$ with r a random integer a half size of d. Ciet and Joye proposed in [5] to compute the point Q using a variant of Shamir's trick for efficiency (algorithm 13 in appendix). However, since they use four temporary points, they cannot use the co-Z formulæ. ECDBL and mECADD should be used instead. For the computation of $S = [r]P$, one can also use the variant of Shamir's trick with one of the scalar being zero.

Point Blinding [6]. Computing $Q = [d](P + R)$ instead of $[d]P$, with R a pseudo-random point is effective against DPA. The chip returns $Q - [d]R$. R and $[d]R$ are computed from R_0 and $[d]R_0$ precomputed and stored in the chip, with R_0 a random point.

This countermeasure was improved in [11] and later in [20]. The authors proposed to modify the ECSM for gradually subtract the random point R. The Binary Expansion with Random Initial Point (BRIP) can be found in appendix (algorithm 14). However, since they use three temporary points, they cannot use the co-Z formulæ. ECDBL and mECADD should be used instead.

3.2 RPA Attack

The RPA [7] is based on the apparition of a special point of the form $(0, y)$ during the ECSM[2].

Let $P_0 = (0, y)$ for some y. Suppose that the Montgomery Ladder (algorihm 5) is used to compute an ECSM. Suppose that the attacker already knows the $n - i - 1$ leftmost bits of the fixed scalar $d = (d_{n-1}, d_{n-2} \ldots, d_{i+1}, d_i, d_{i-1}, \ldots, d_0)_2$. He tries to recover the unknown bit d_i.

The attacker computes the point $P = [(d_{n-1}, d_{n-2}, \ldots, d_{i+1}, 0)_2^{-1} \bmod \sharp E]P_0$ and gives P to the targeted chip that computes $[d]P$. If $d_i = 0$, then the point P_0 will appear during the ECSM. If the attacker is able to recognize a zero value in a register, he can then conclude whether his hypothesis ($d_i = 0$) was correct or not.

3.3 ZPA Attack

The Zero-Value Point Attack (ZPA) [1] uses the same approach than the RPA, except that the attack is not only interested in zero values in coordinates but in intermediate registers when computing the double of a point, or during the addition of two points. Such points are defined as zero-value points.

[2] the point $(x, 0)$ can also be used but a point of this form is of order 2. In ECC, the order of the provided base point is checked and points of order 2 never appear during an ECSM.

Finding zero-value points for doubling formulæ consists in resolving polynomial equations in x, y with low degree (less than 4).

Finding zero-value points for addition is more difficult. For the Montgomery Ladder algorithm, suppose the attacker already knows the $n-i-1$ leftmost bits of the fixed scalar $d = (d_{n-1}, d_{n-2}, \dots, d_0)_2$ and try to recover d_i. With $c = (d_{n-1}, d_{n-2} \dots, d_{i+1}, 0)_2$, he has to find a point P_0 such that $[c]P_0$ and $[c+1]P_0$ are zero-value points. The only known procedure is using division polynomials and solve equations in two variables with degree of order $\mathcal{O}(c^2)$ [1]. At this day, when c is large, it is a hard problem. This problem was discussed in [12] and [1].

Remark 2. The random projective coordinates and the random curve isomorphism countermeasures described in the previous subsection fail against RPA and ZPA.

Some countermeasures to prevent RPA and ZPA are given below.

Isogeny Defence [25,2]. Computing an ECSM on a curve E' isogenous to E such that E' does not contain any non-trivial zero-value point is effective against the RPA and the ZPA.

Randomized Linearly Transformed Coordinates (RLC) [11]. This countermeasure consists in modifying the addition and doubling formulæ such that a zero value can never show up. A point $P = (X, Y, Z)$ is transformed into (X_μ, Y, Z, μ) with $X_\mu = X + \mu$, with μ a random field element. The potential zero value X is never manipulated alone. The countermeasure was given in [11] with classical doubling and addition formulæ. We adapted the countermeasure with the co-Z formulæ, because it is more efficient. We only modified the formulæ to prevent RPA because of the remark of the difficulty of the ZPA on addition formulæ. The countermeasure adapted with co-Z formulæ can be found in appendix (algorithm 16).

Remark 3. The scalar randomisation, random scalar split and point blinding countermeasures described in the previous subsection are also effective against RPA and ZPA.

4 Our Proposed Countermeasure

We describe in this section our new co-Z formulæ that we can use to perform a secured ECSM against the DPA and the RPA.

The main idea is to perform the ECSM with a base point $P' = (0, y')$. This point and its opposite $-P'$ are the only points with a zero X-coordinate. Using the ECSM add only Montgomery Ladder using co-Z formulæ (algorithm 6) with P' as the base point, we will show that the inputs of ZADDC are never $\pm P'$ whatever the value of the scalar. So the RPA cannot be performed.

The output point R_{d_i} of ZADDC is always equal to $\pm P'$ (see remark 1). So $\pm P' = (0, \pm y')$ appears at the end of ZADDC and therefore appears at the beginning of ZADDU. Algorithms 3 and 4 can be modified by removing the useless multiplications and additions with the zero-value.

Algorithm 8. co-Z addition and update with a zero value (ZADDUzero)

```
Input: $P=(X_1,Y_1,Z), Q=(0,Y_2,Z) \in E^{\mathcal{J}}(\mathbb{F}_p)$
Output: $(R,S)$ with $R = P + Q$ and $S = (\lambda^2 X_1, \lambda^3 Y_1, \lambda Z)$ with $\lambda = X_1$

    $C \leftarrow X_1^2$
    $W_1 \leftarrow X_1 C$; $Z_3 = Z X_1$
    $D \leftarrow (Y_1 - Y_2)^2$; $A_1 \leftarrow Y_1 W_1$
    $X_3 \leftarrow D - W_1$
    $Y_3 \leftarrow (Y_1 - Y_2)(W_1 - X_3) - A_1$
    $X_4 \leftarrow W_1$; $Y_4 \leftarrow A_1$

    return $((X_3,Y_3,Z_3),(X_4,Y_4,Z_3))$
```

Algorithm 9. conjugate co-Z addition with a zero value (ZADDCzero)

```
Input: $P=(X_1,Y_1,Z), Q=(X_2,Y_2,Z) \in E^{\mathcal{J}}(\mathbb{F}_p)$, such that $x_{P-Q}=0$
Output: $(R,S)$ with $R = P + Q$, $S = P - Q$

    $C \leftarrow (X_1 - X_2)^2$
    $W_1 \leftarrow X_1 C$; $W_2 \leftarrow X_2 C$; $Z_3 \leftarrow Z(X_1 - X_2)$
    $D \leftarrow (Y_1 - Y_2)^2$; $A_1 = Y_1(W_1 - W_2)$
    $X_3 \leftarrow D - W_1 - W_2$
    $Y_3 \leftarrow (Y_1 - Y_2)(W_1 - X_3) - A_1$
    $Y_4 \leftarrow (Y_1 + Y_2)W_1 - A_1$

    return $((X_3,Y_3,Z_3),(0,Y_4,Z_3))$
```

ZADDUzero and ZADDCzero can be combined without the Z-coordinate: ZACAUzero'. The algorithm is given in appendix (algorithm 17). ZACAUzero' requires one multiplication and one square less compared to ZACAU' (algorithm 15).

The main step is to find a method to transform any base point $P = (x, y)$ of any curve into a base point $P' = (0, y')$.

We present in this paper two methods of such a transformation. The first method uses isogenies and was proposed in [2]. The second method is an extension of the random curve isomorphism countermeasure.

4.1 Transformation of the Base Point Using Isogenies

An isogeny between two elliptic curves E and E' defined over $\mathbb{F}_p$ is a non-constant morphism $\phi : E \to E'$. Every isogeny has a finite kernel and the size of this kernel is called the degree of isogeny.

Brier and Joye introduced in [4] the use of isogenies for efficiency: they transform an elliptic curve $E : y^2 = x^3 + ax + b$ into an elliptic curve $E' : y^2 = x^3 - 3x + b'$. The parameter $a' = -3$ brings better performance for doubling formulæ.

Smart proposed in [25] to use isogenies as a countermeasure against the RPA [7]. ECSMs are performed on an isogenous elliptic curve that does not contain any special point of the form $(0, y)$. Akishita and Takagi extended the isogeny defense in [2] so it can also prevent the ZPA. They also use isogenies for efficiency for binary ECSM methods. The given elliptic curve is transformed into an isogenous curve where the base point G' has the particular form $G' = (0, y')$, for that the addition with the point G' is more efficient because of the zero value.

Finding isogenies for a given elliptic curve is not trivial. Isogenies of standardized curves are precomputed and stored in the chip. The base point given also needs to be mapped in the isogenous curve. This transformation has a non negligible cost which is discussed in [25].

4.2 Transformation of the Base Point Using Isomorphism

We propose here a more practical method to transform the base point that can work to any arbitrary curve. We extend the isomorphic curve countermeasure proposed in [15]. We need the following corollary of the theorem [22, Theorem 2.2].

Corollary 1. *Let $\mathbb{F}_p$ be a finite field with a prime $p > 3$. The elliptic curves given by the Weierstraß equations*

$$\begin{aligned} E &: y^2 = x^3 + a_4 x + a_6 \\ E' &: y^2 = x^3 + a_2' x^2 + a_4' x + a_6' \end{aligned}$$

are isomorphic over $\mathbb{F}_p$ if and only if there exist $u \in \mathbb{F}_p^*$ and $r \in \mathbb{F}_p$ such that the change of variables

$$(x, y) \to (u^{-2}(x - r), u^{-3}y)$$

transforms equation E into equation E'. Such a transformation is referred to as an admissible change of variables. Furthermore,

$$\begin{cases} u^2 a_2' = 3r \\ u^4 a_4' = a_4 + 3r^2 \\ u^6 a_6' = a_6 + r a_4 + r^3 \ . \end{cases}$$

Proof. The corollary is simply a particular case of the theorem [22, Theorem 2.2] with $s = t = a_1 = a_2 = a_3 = a_1' = a_3' = 0$. □

Remark 4. In the isomorphic curve countermeasure [15], the isomorphic curve E' is also in its short short Weierstraß form for efficiency reason. Therefore $a_2' = 0$. This implies $r = 0$. Only u is randomly chosen for the countermeasure.

The ECSM add only Montgomery Ladder using co-Z formulæ (algorithm 6) has the following properties:

- the base point P or its opposite $-P$ appears at each iteration
- a point doubling is never performed in the main loop

The goal is to perform an ECSM with a base point of the form $P' = (0, y_{P'})$. If the base point given is $P = (x_P, y_P)$ on the elliptic curve E, one can choose $r = x_P$ and a random u so that the isomorphism:

$$\varphi : E \xrightarrow{\sim} E', \begin{cases} \mathcal{O} \to \mathcal{O} \\ (x, y) \to (u^{-2}(x - r), u^{-3}y) \end{cases}$$

maps the point $P = (x_P, y_P)$ into the point $P' = (0, u^{-3}y_P)$. However, the elliptic curve E' is not in the short Weierstraß form: the parameter a_2' is non-zero. Thanks to the add only Montgomery Ladder using co-Z formulæ (algorithm 6), a doubling is never performed. Only the addition has to be modified. Using Section 2 to see the modifications due to the non-zero a_2' parameter on co-Z formulæ, we can give the modified co-Z formulæ.

Algorithm 10. co-Z addition and update with a zero value and a_2' parameter (ZADDUazero)

```
Input: $P = (X_1, Y_1, Z), Q = (0, Y_2, Z) \in E'^{\mathcal{J}}(\mathbb{F}_p)$ and $T_a = a_2' Z^2$
Output: $(R, S, T_a)$ with $R = P + Q$, $S = (\lambda^2 X_1, \lambda^3 Y_1, \lambda Z)$ with $\lambda = X_1$ and $T_a = a_2' Z_3^2$

  $C \leftarrow X_1^2$
  $W_1 \leftarrow X_1 C;\ Z_3 \leftarrow Z X_1;\ T_a \leftarrow T_a C$
  $D \leftarrow (Y_1 - Y_2)^2;\ A_1 \leftarrow Y_1 W_1$
  $X_3 \leftarrow D - W_1 - T_a$
  $Y_3 \leftarrow (Y_1 - Y_2)(W_1 - X_3) - A_1$
  $X_4 \leftarrow W_1;\ Y_4 \leftarrow A_1$
  return $((X_3, Y_3, Z_3), (X_4, Y_4, Z_3), T_a)$
```

Algorithm 11. conjugate co-Z addition with a zero value and a_2' parameter (ZADDCazero)

```
Input: $P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z) \in E'^{\mathcal{J}}(\mathbb{F}_p)$, such that $x_{P-Q} = 0$ and $T_a = a_2' Z^2$
Output: $(R, S, T_a)$ with $R = P + Q$, $S = P - Q$ and $T_a = a_2' Z_3^2$

  $C \leftarrow (X_1 - X_2)^2$
  $W_1 \leftarrow X_1 C;\ W_2 \leftarrow X_2 C;\ Z_3 \leftarrow Z(X_1 - X_2)$
  $T_a \leftarrow T_a C$
  $D \leftarrow (Y_1 - Y_2)^2;\ A_1 \leftarrow Y_1(W_1 - W_2)$
  $X_3 \leftarrow D - W_1 - W_2 - T_a$
  $Y_3 \leftarrow (Y_1 - Y_2)(W_1 - X_3) - A_1$
  $Y_4 \leftarrow (Y_1 + Y_2)W_1 - A_1$
  return $((X_3, Y_3, Z_3), (0, Y_4, Z_3), T_a)$
```

The combination of the two formulæ without the Z-coordinate is given in appendix (algorithm 18). We called it ZACAUazero'. ZACAUazero' requires $9M + 5S$. That is one multiplication more and one square less than ZACAU'.

The complete ECSM using the countermeasure and the modified co-Z formulæ is given below.

Algorithm 12. (X, Y)-only add only Montgomery Ladder using modified co-Z formulæ

```
Input: $P \in E^{\mathcal{J}}(\mathbb{F}_p), d = (d_{n-1}, \ldots, d_0)_2, d_{n-1} = 1$
Output: $[d]P$

  $u \overset{R}{\leftarrow} \mathbb{F}_p^*$
  $P' \leftarrow (0, u^{-3} y_P, 1)$                                   ▷ isomorphism
  $T_a \leftarrow 3 x_P u^{-2}$                    ▷ $T_a$ is the parameter $a_2'$ of the isomorphic curve $E'$
  $R_0 \leftarrow P', R_1 \leftarrow 2P'$                ▷ $R_0$ and $R_1$ must share the same $Z$-coordinate
  $C \leftarrow (X_{R_0} - X_{R_1})^2 = X_{R_1}^2$
  for $i = n - 2$ downto 1 do
      $(R_{d_i}, R_{1-d_i}, C, T_a) \leftarrow$ ZACAUazero'$(R_{d_i}, R_{1-d_i}, C, T_a)$
  end for
  $b \leftarrow d_0$;  $(R_{1-b}, R_b, T_a) \leftarrow$ ZADDCazero'$(R_b, R_{1-b}, T_a)$
  $Z \leftarrow 3 x_P u^{-2} Y_{R_b}(X_{R_0} - X_{R_1});\ \lambda \leftarrow u^{-3} y_P T_a$     ▷ $Z = a_2' Y_{R_b}(X_{R_0} - X_{R_1}),\ \lambda = u^{-3} y_P a_2' Z_{R_0}^2$
  $(R_b, R_{1-b}, T_a) \leftarrow$ ZADDUazero'$(R_{1-b}, R_b, T_a)$
  $(x', y') \leftarrow \left( \left(\frac{\lambda}{Z}\right)^2 X_{R_0}, \left(\frac{\lambda}{Z}\right)^3 Y_{R_0} \right)$
  $(x, y) \leftarrow (u^2 x' + x_P, u^3 y)$                               ▷ isomorphism inverse

  return $(x, y)$
```

4.3 Security Analysis of our Countermeasure

The security against RPA is based on the fact that the base point P' has a x-zero coordinate. The only possible points having a x-zero coordinate are P' and $-P'$. These two points can never appear as inputs of ZACAUazero'. Joye was the first to propose in [13] an extension of the random curve isomorphism countermeasure to prevent the RPA. His countermeasure can be applied for elliptic curves on binary fields of the form $y^2 + xy = x^3 + a_2x^2 + a_6$, so choosing a random r for the isomorphism of theorem [22, Theorem 2.2] does not affect the efficiency. In this paper, we introduced the extension of the random isomorphic curve countermeasure on elliptic curve over field of large characteristic without any efficiency loss.

SPA Security. Our ECSM is regular: the same operation ZACAUazero' is performed whatever the value of the current bit. The classical SPA where an attacker is able to distinguish different patterns depending on the value of the current bit cannot be applied.

DPA Security. The random parameter u gives the security against DPA. All values and intermediates values in ZACAUazero' (algorithm 18) are multiplicatively randomized by u.

RPA Security. The RPA security is provided by the following lemma.

Lemma 1. *Suppose d satisfies $1 < d < ord(P)$. The points R_0 and R_1 at the beginning of each iteration $1 \leq i \leq n-3$ in algorithm 12 cannot take the values $\pm P'$.*

Proof. Suppose the ECSM is performed with the scalar $d = (d_{n-1}, d_{n-2}, \ldots, d_0)_2$ and the base point P'. Let $c_i = (d_{n-1}, d_{n-2} \ldots, d_{i+1})_2$. At the beginning of iteration i with $1 \leq i \leq n-3$, the points R_0, R_1 verify $R_0 = c_iP'$ and $R_1 = [c_i + 1]P'$.

– if $R_0 = [c_i]P' = P'$, then $[c_i - 1]P' = \mathcal{O}$ so the order of P' and P is $(c_i - 1)$, which is impossible by the condition of d.
– if $R_0 = [c_i]P' = -P'$, then $[c_i + 1]P' = \mathcal{O}$ so the order of P' and P is $(c_i + 1)$, which is impossible by the condition of d.
– if $R_1 = [c_i + 1]P' = P'$, then $[c_i]P' = \mathcal{O}$ so the order of P' and P is c_i, which is impossible by the condition of d.
– if $R_1 = [c_i + 1]P' = -P'$, then $[c_i + 2]P' = \mathcal{O}$ so the order of P' and P is $(c_i + 2)$, which is impossible by the condition of d.

By contradiction, we prove that the points R_0, R_1 cannot take the values P' or $-P'$. □

An elliptic curve $E' : y^2 = x^3 + a'_2x^2 + a'_4x + a'_6$ contains at most two points of the form $(0, y')$. Those points are $P' = (0, \sqrt{a'_6})$ and $-P' = (0, -\sqrt{a'_6})$. The points P' and $-P'$ are the only points with a zero x-coordinate. With the lemma, we can state that an attacker is not able to perform a RPA attack because the zero-value points never appear in outputs of ZACAUazero'.

ZPA Security. The ZPA security is not guaranteed. However, the ZPA remains difficult because no doubling is performed during the ECSM. The attacker has to find zero-value points for addition which is a difficult problem [12,1]. This is discussed in Section 3.

5 Comparison with Prior RPA Countermeasures

In this section, we compare different countermeasures against RPA described in Section 3.

We can see that if the cost of a multiplication and a square is the same, our countermeasure does not bring any additional cost. The isogeny defence does not bring any additional cost as well but it does not work to any curve: the isogenous curves have to be precomputed and stored in the chip. Moreover, the base point given has to be mapped to the isogenous curve, so the countermeasure has an extra cost. The cost of isogeny is approximatively $3l$ multiplications with l the degree of isogeny [25].

ECSM	Countermeasure	Cost per bit	Cost of ECSM	works to any curve
(X,Y)-only co-Z Montgomery Ladder	none (reference) [9]	$8M+6S$	$n(8M+6S)$	✓
(X,Y)-only co-Z Montgomery Ladder	$d' = d + r\sharp E$ [6]	$8M+6S$	$(n+32)(8M+6S)$	✓
Regular Shamir's trick	random scalar split [5]	$8M+12S$	$n(8M+12S)$	✓
(X,Y)-only co-Z Montgomery Ladder	isogeny defense [25,2]	$8M+6S$	$n(8M+6S)$	× (isogenous curves precomputed)
BRIP algorithm 14	BRIP [20]	$8M+12S$	$n(8M+12S)$	× (initial random point precomputed)
(X,Y)-only co-Z Montgomery Ladder with RLC	Random Linear Coordinates [11]	$10M+6S$	$n(10M+6S)$	✓
(X,Y)-only co-Z Montgomery Ladder with ZACAUazero'	this paper	$9M+5S$	$n(9M+5S)$	✓

6 Conclusion

We presented in this paper a secured ECSM where the base point is of the form $P' = (0, y')$. The base point P given is transformed into P' using an extension of the isomorphic curve countermeasure [15]. The ECSM is secured against DPA [19] and RPA [7]. Moreover, thanks to co-Z formulæ, a doubling is never performed during the main loop, so the ZPA [1] remains a hard problem. A comparison of different countermeasure against RPA is also given. Using modified co-Z formulæ, the loss of efficiency is negligible, and our countermeasure is the most efficient.

Further work is to guarantee the security against the ZPA with either finding formulæ with no zero-value point or calculating the cost of finding zero-value points for addition. Also, a comparison of the memory cost of countermeasures against RPA is missing.

References

1. Akishita, T., Takagi, T.: Zero-Value Point Attacks on Elliptic Curve Cryptosystem. In: Boyd, C., Mao, W. (eds.) ISC 2003. LNCS, vol. 2851, pp. 218–233. Springer, Heidelberg (2003)
2. Akishita, T., Takagi, T.: On the Optimal Parameter Choice for Elliptic Curve Cryptosystems Using Isogeny. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 346–359. Springer, Heidelberg (2004)
3. Bernstein, D.J., Lange, T.: Explicit-formulas database (2004), `http://hyperelliptic.org/EFD`
4. Brier, E., Joye, M.: Fast Point Multiplication on Elliptic Curves through Isogenies. In: Fossorier, M.P.C., Høholdt, T., Poli, A. (eds.) AAECC 2003. LNCS, vol. 2643, pp. 43–50. Springer, Heidelberg (2003)
5. Ciet, M., Joye, M.: (Virtually) Free Randomization Techniques for Elliptic Curve Cryptography. In: Qing, S., Gollmann, D., Zhou, J. (eds.) ICICS 2003. LNCS, vol. 2836, pp. 348–359. Springer, Heidelberg (2003)
6. Coron, J.-S.: Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 292–302. Springer, Heidelberg (1999)
7. Goubin, L.: A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 199–211. Springer, Heidelberg (2003)
8. Goundar, R.R., Joye, M., Miyaji, A.: Co-Z Addition Formulæ and Binary Ladders on Elliptic Curves - (Extended Abstract). In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 65–79. Springer, Heidelberg (2010)
9. Goundar, R.R., Joye, M., Miyaji, A., Rivain, M., Venelli, A.: Scalar multiplication on Weierstraß elliptic curves from Co-Z arithmetic. Journal of Cryptographic Engineering 1, 161–176 (2011)
10. Hutter, M., Joye, M., Sierra, Y.: Memory-Constrained Implementations of Elliptic Curve Cryptography in Co-Z Coordinate Representation. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 170–187. Springer, Heidelberg (2011)
11. Itoh, K., Izu, T., Takenaka, M.: Efficient Countermeasures against Power Analysis for Elliptic Curve Cryptosystems. In: Proceedings of CARDIS 2004, pp. 99–114. Kluwer Academic Publishers (2004)
12. Izu, T., Takagi, T.: Exceptional Procedure Attackon Elliptic Curve Cryptosystems. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 224–239. Springer, Heidelberg (2003)
13. Joye, M.: Smart-Card Implementation of Elliptic Curve Cryptography and DPA-type Attacks. In: Proceedings of CARDIS 2004, pp. 115–126. Kluwer Academic Publisher (2004)
14. Joye, M.: Highly Regular Right-to-Left Algorithms for Scalar Multiplication. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 135–147. Springer, Heidelberg (2007)

15. Joye, M., Tymen, C.: Protections against Differential Analysis for Elliptic Curve Cryptography. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 377–390. Springer, Heidelberg (2001)
16. Joye, M., Yen, S.-M.: The Montgomery Powering Ladder. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 291–302. Springer, Heidelberg (2003)
17. Koblitz, N.: Elliptic Curve Cryptosystems. J. Mathematics of Computation 48, 203–209 (1987)
18. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
19. Kocher, P.C., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
20. Mamiya, H., Miyaji, A., Morimoto, H.: Efficient Countermeasures against RPA, DPA, and SPA. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 343–356. Springer, Heidelberg (2004)
21. Meloni, N.: New Point Addition Formulae for ECC Applications. In: Carlet, C., Sunar, B. (eds.) WAIFI 2007. LNCS, vol. 4547, pp. 189–201. Springer, Heidelberg (2007)
22. Menezes, A.J.: Elliptic Curve Public Key Cryptosystems. Kluwer Academic Publishers (1993)
23. Miller, V.S.: Use of elliptic curves in cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
24. Okeya, K., Sakurai, K.: Power Analysis Breaks Elliptic Curve Cryptosystems Even Secure against the Timing Attack. In: Roy, B., Okamoto, E. (eds.) INDOCRYPT 2000. LNCS, vol. 1977, pp. 178–190. Springer, Heidelberg (2000)
25. Smart, N.P.: An Analysis of Goubin's Refined Power Analysis Attack. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 281–290. Springer, Heidelberg (2003)

A Elliptic Curve Scalar Multiplication Algorithms

Algorithm 13. Variant of Shamir's trick [5]

```
Input: $P, S \in E^{\mathcal{J}}(\mathbb{F}_p), k = (k_{n-1}, \ldots, k_0)_2,$
  $d = (d_{n-1}, \ldots, d_0)_2$ with $(k_{n-1}, d_{n-1}) \neq (0, 0)$
Output: $[k]P + [d]S$
  $R_1 \leftarrow P;\ R_2 \leftarrow S;\ R_3 \leftarrow P + S;\ R_4 \leftarrow P + S$
  $c \leftarrow 2d_{n-1} + k_{n-1};\ R_0 \leftarrow R_c$
  for $i = n - 2$ downto 0 do
      $R_0 \leftarrow \textsc{ecdbl}(R_0)$
      $b \leftarrow \neg(k_i \vee d_i);\ c \leftarrow 2d_i + k_i$
      $R_{4b} \leftarrow \textsc{mecadd}(R_{4b}, R_c)$
  end for
  return $R_0$
```

Algorithm 14. BRIP [20]

```
Input: $d = (d_{n-1}, \ldots, d_0)_2, P$
Output: $[d]P$

  $R \leftarrow randompoint()$
  $R_0 \leftarrow R,\ R_1 \leftarrow -R, R_0 = P - R$
  for $i = n - 1$ downto 0 do
      $R_0 \leftarrow \textsc{ecdbl}(R_0)$
      $R_0 \leftarrow \textsc{mecadd}(R_0, R_{d_i+1})$
  end for

  return $R_0 + R_1$
```

B co-Z Formulæ

Algorithm 15. (X, Y)-only co-Z conjugate-addition-addition with update (ZACAU') [9]

```
Input: $(X_1, Y_1), (X_2, Y_2), C$ with $P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z) \in E^{\mathcal{J}}(\mathbb{F}_p)$ and $C = (X_1 - X_2)^2$
Output: $(X_3, Y_3), (X_4, Y_4), C$ with $R = (X_3, Y_3, Z_3)$, $S = (X_4, Y_4, Z_3)$ such that $R = 2P, S = P + Q$ and
    $C = (X_3 - X_4)^2$

  $W_1 \leftarrow X_1 C;\ W_2 \leftarrow X_2 C$
  $D_1 \leftarrow (Y_1 - Y_2)^2;\ A_1 \leftarrow Y_1(W_1 - W_2)$
  $X'_1 \leftarrow D_1 - W_1 - W_2;\ Y'_1 \leftarrow (Y_1 - Y_2)(W_1 - X'_1) - A_1$
  $D_2 \leftarrow (Y_1 + Y_2)^2$
  $X'_2 \leftarrow D_2 - W_1 - W_2;\ Y'_2 \leftarrow (Y_1 + Y_2)(W_1 - X'_2) - A_1$
  $C' \leftarrow (X'_1 - X'_2)^2$
  $X_4 \leftarrow X'_1 C';\ W'_2 \leftarrow X'_2 C'$
  $D' \leftarrow (Y'_1 - Y'_2)^2;\ Y_4 \leftarrow Y'_1(X_4 - W'_2)$
  $X_3 \leftarrow D' - X_4 - W'_2$
  $C \leftarrow (X_3 - X_4)^2$
  $Y_3 \leftarrow (Y'_1 - Y'_2 + X_4 - X_3)^2 - D' - C - 2Y_4$
  $X_3 \leftarrow 4X_3;\ Y_3 \leftarrow 4Y_3;\ X_4 \leftarrow 4X_4$
  $Y_4 \leftarrow 8Y_4;\ C \leftarrow 16C$
  return $(X_3, Y_3), (X_4, Y_4), C$
```

Algorithm 16. (X, Y)-only co-Z conjugate-addition-addition with update using RLC

```
Input: $(X_{1,\mu}, Y_1), (X_{2,\mu}, Y_2), C, \mu$ with $P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z) \in E^{\mathcal{J}}(\mathbb{F}_p)$
    with $X_{1,\mu} = X_1 + \mu, X_{2,\mu} = X_2 + \mu$ and $C = (X_1 - X_2)^2$
Output: $(X_{3,\mu}, Y_3), (X_{4,\mu}, Y_4), C$ with $R = (X_3, Y_3, Z_3)$, $S = (X_4, Y_4, Z_3)$ such that $R = 2P, S = P + Q$
    with $X_{3,\mu} = X_3 + \mu, X_{4,\mu} = X_4 + \mu$ and $C = (X_3 - X_4)^2$

  $W_1 \leftarrow X_{1,\mu} C;\ W_2 \leftarrow X_{2,\mu} C$
  $C_\mu = \mu C$
  $D \leftarrow (Y_1 - Y_2)^2;\ A_1 \leftarrow Y_1(W_1 - W_2)$
  $X'_{1,\mu} \leftarrow D - W_1 - W_2 + 2C_\mu + \mu;\ Y'_1 \leftarrow (Y_1 - Y_2)(W_1 - X'_{1,\mu} + \mu - 2C_\mu) - A_1$
  $\bar{D} \leftarrow (Y_1 + Y_2)^2$
  $X'_{2,\mu} \leftarrow \bar{D} - W_1 - W_2 + 2C_\mu + \mu;\ Y'_2 \leftarrow (Y_1 + Y_2)(W_1 - X'_{2,\mu} + \mu - 2C_\mu) - A_1$
  $C' \leftarrow (X'_{1,\mu} - X'_{2,\mu})^2$
  $C'_\mu = \mu C'$
  $X_{4,\mu} \leftarrow X'_{1,\mu} C';\ W'_2 \leftarrow X'_{2,\mu} C'$
  $D' \leftarrow (Y'_1 - Y'_2)^2;\ Y_4 \leftarrow Y'_1(X_{4,\mu} - W'_2 + \mu - 2C'_\mu)$
  $X_{3,\mu} \leftarrow D' - X_{4,\mu} - W'_2 + 2C'_\mu + \mu$
  $X_{4,\mu} \leftarrow X_{4,\mu} - C'_\mu + \mu$
  $C \leftarrow (X_{3,\mu} - X_{4,\mu})^2$
  $Y_3 \leftarrow (Y'_1 - Y'_2 + X_{4,\mu} - X_{3,\mu})^2 - D' - C - 2Y_4$
  $X_{3,\mu} \leftarrow 4X_{3,\mu} - 3\mu;\ Y_3 \leftarrow 4Y_3;\ X_{4,\mu} \leftarrow 4X_{4,\mu} - 3\mu$
  $Y_4 \leftarrow 8Y_4;\ C \leftarrow 16C$
  return $(X_{3,\mu}, Y_3), (X_{4,\mu}, Y_4), C$
```

Algorithm 17. (X, Y)-only co-Z conjugate-addition-addition with a zero value (ZACAUzero')

Input: $(X_1, Y_1), (X_2, Y_2), C$ with $P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z) \in E'^{\mathcal{J}}(\mathbb{F}_p)$ such that $x_{P-Q} = 0$ and $C = (X_1 - X_2)^2$
Output: $(X_3, Y_3), (X_4, Y_4), C$ with $R = (X_3, Y_3, Z_3)$, $S = (X_4, Y_4, Z_3)$ such that $R = 2P, S = P + Q$ and $C = (X_3 - X_4)^2$

```
  $W_1 \leftarrow X_1 C;\ W_2 \leftarrow X_2 C$
  $D \leftarrow (Y_1 - Y_2)^2;\ A_1 \leftarrow Y_1(W_1 - W_2)$
  $X_1' \leftarrow D - W_1 - W_2;\ Y_1' \leftarrow (Y_1 - Y_2)(W_1 - X_1') - A_1$
  $Y_2' \leftarrow (Y_1 + Y_2)W_1 - A_1$
  $C' \leftarrow (X_1' - X_2')^2$
  $X_4 \leftarrow X_1' C'$
  $D' \leftarrow (Y_1' - Y_2')^2;\ Y_4 \leftarrow Y_1' X_4$
  $X_3 \leftarrow D' - X_4$
  $C \leftarrow (X_3 - X_4)^2$
  $Y_3 \leftarrow (Y_1' - Y_2' + X_4 - X_3)^2 - D' - C - 2Y_4$
  $X_3 \leftarrow 4X_3;\ Y_3 \leftarrow 4Y_3;\ X_4 \leftarrow 4X_4$
  $Y_4 \leftarrow 8Y_4;\ C \leftarrow 16C$
  return $(X_3, Y_3), (X_4, Y_4), C$
```

Algorithm 18. (X, Y)-only co-Z conjugate-add-add with a zero value and a_2' (ZACAUazero')

Input: X_1, Y_1, X_2, T_a, C with $P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z) \in E'^{\mathcal{J}}(\mathbb{F}_p)$ such that $x_{P-Q} = 0$, $T_a = a_2' Z^2$ and $C = (X_1 - X_2)^2$
Output: $(X_3, Y_3), (X_4, Y_4), T_a, C$ with $R = (X_3, Y_3, Z_3)$, $S = (X_4, Y_4, Z_3)$ such that $R = 2P, S = P + Q$, $T_a = a_2' Z_3^2$ and $C = (X_3 - X_4)^2$

```
  $W_1 \leftarrow X_1 C;\ W_2 \leftarrow X_2 C;\ T_a \leftarrow T_a C$
  $D \leftarrow (Y_1 - Y_2)^2;\ A_1 \leftarrow Y_1(W_1 - W_2)$
  $X_1' \leftarrow D - W_1 - W_2 - T_a$
  $Y_1' \leftarrow (Y_1 - Y_2)(W_1 - X_1') - A_1;\ Y_2' \leftarrow (Y_1 + Y_2)W_1 - A_1$
  $C' \leftarrow (X_1' - X_2')^2$
  $X_4 \leftarrow X_1' C';\ T_a \leftarrow T_a C'$
  $D' \leftarrow (Y_1' - Y_2')^2;\ Y_4 \leftarrow Y_1' X_4$
  $X_3 \leftarrow D' - X_4 - T_a$
  $C \leftarrow (X_3 - X_4)^2$
  $Y_3 \leftarrow (Y_1' - Y_2' + X_4 - X_3)^2 - D - C - 2Y_4$
  $X_3 \leftarrow 4X_3;\ Y_3 \leftarrow 4Y_3;\ X_4 \leftarrow 4X_4;\ T_a \leftarrow 4T_a;\ Y_4 \leftarrow 8Y_4;\ C \leftarrow 16C$
  return $(X_3, Y_3), (X_4, Y_4), T_a, C$
```

Algorithm 19. (X, Y)-only co-Z conjugate-add-add with a zero value and a'_2 (ZACAUzero') (register allocation)

```
Input: $(X_1, Y_1), (X_2, Y_2), T_a, C$ with $P = (X_1, Y_1, Z), Q = (X_2, Y_2, Z)$ such that $x_{P-Q} = 0$, $T_a = a'_2 Z^2$ and
  $C = (X_1 - X_2)^2$
Output: $(X_3, Y_3), (X_4, Y_4), T_a, C$ with $R = (X_3, Y_3, Z_3)$, $S = (X_4, Y_4, Z_3)$ such that $R = 2P, S = P + Q$,
  $T_a = a'_2 Z_3^2$ and $C = (X_3 - X_4)^2$

  $T_1 \leftarrow X_1, T_2 \leftarrow Y_1, T_3 \leftarrow C, T_4 \leftarrow X_2, T_5 \leftarrow Y_2$

   1: $T_a \leftarrow T_a \times T_3$        $\{a'_2 Z_{P+Q}^2\}$
   2: $T_6 \leftarrow T_3 \times T_4$        $\{W_2\}$
   3: $T_3 \leftarrow T_3 \times T_1$        $\{W_1\}$
   4: $T_1 \leftarrow T_2 - T_5$        $\{Y_1 - Y_2\}$
   5: $T_1 \leftarrow T_1^2$        $\{D\}$
   6: $T_1 \leftarrow T_1 - T_a$        $\{D - a'_2 Z'^2\}$
   7: $T_1 \leftarrow T_1 - T_3$        $\{D - a'_2 Z'^2 - W_1\}$
   8: $T_1 \leftarrow T_1 - T_6$        $\{X'_1\}$
   9: $T_6 \leftarrow T_6 - T_3$        $\{W_2 - W_1\}$
  10: $T_6 \leftarrow T_6 \times T_2$        $\{-A_1\}$
  11: $T_2 \leftarrow T_2 - T_5$        $\{Y_1 - Y_2\}$
  12: $T_5 \leftarrow 2T_5$        $\{2Y_2\}$
  13: $T_5 \leftarrow T_5 + T_2$        $\{Y_1 + Y_2\}$
  14: $T_5 \leftarrow T_5 \times T_3$        $\{Y'_2 + A_1\}$
  15: $T_5 \leftarrow T_5 + T_6$        $\{Y'_2\}$
  16: $T_3 \leftarrow T_3 - T_1$        $\{W_1 - X'_1\}$
  17: $T_2 \leftarrow T_2 \times T_3$        $\{Y'_1 + A_1\}$
  18: $T_2 \leftarrow T_2 + T_6$        $\{Y'_1\}$
  19: $T_3 \leftarrow T_1^2$        $\{C'\}$
  20: $T_a \leftarrow T_a \times T_3$        $\{a'_2 Z_R^2\}$
  21: $T_4 \leftarrow T_1 \times T_3$        $\{X_4\}$
  22: $T_3 \leftarrow T_2 - T_5$        $\{Y'_1 - Y'_2\}$
  23: $T_5 \leftarrow T_2 \times T_4$        $\{Y_4\}$
  24: $T_2 \leftarrow T_3^2$        $\{D'\}$
  25: $T_1 \leftarrow T_2 - T_a$        $\{D' - a'_2 Z'^2\}$
  26: $T_1 \leftarrow T_1 - T_4$        $\{X_3\}$
  27: $T_6 \leftarrow T_1 - T_4$        $\{X_3 - X_4\}$
  28: $T_3 \leftarrow T_3 - T_6$        $\{Y'_1 - Y'_2 + X_4 - X_3\}$
  29: $T_3 \leftarrow T_3^2$        $\{(Y'_1 - Y'_2 + X_4 - X_3)^2\}$
  30: $T_2 \leftarrow T_3 - T_2$        $\{(Y'_1 - Y'_2 + X_4 - X_3)^2 - D'\}$
  31: $T_3 \leftarrow T_6^2$        $\{C\}$
  32: $T_2 \leftarrow T_2 - T_3$        $\{(Y'_1 - Y'_2 + X_4 - X_3)^2 - D' - C\}$
  33: $T_5 \leftarrow 2T_5$        $\{2Y_4\}$
  34: $T_2 \leftarrow T_2 - T_5$        $\{Y_3\}$
  35: $T_1 \leftarrow 4T_1$        $\{4X_3\}$
  36: $T_2 \leftarrow 4T_2$        $\{4Y_3\}$
  37: $T_3 \leftarrow 16T_3$        $\{16C\}$
  38: $T_4 \leftarrow 4T_4$        $\{4X_4\}$
  39: $T_5 \leftarrow 4T_5$        $\{8Y_3\}$
  40: $T_a \leftarrow 4T_a$        $\{4a'_2 Z_3^2\}$

  return $((T_1, T_2), (T_4, T_5), T_a, T_3)$
```