New Hybrid Method for Isogeny-based Cryptosystems using Edwards Curves

Suhri Kim[1], Kisoon Yoon[2], Jihoon Kwon[3], Young-Ho Park[3], and Seokhie Hong[1]

[1] Center for Information Security Technologies (CIST), Korea University, Seoul, Republic of Korea
suhrikim@gmail.com, shhong@korea.ac.kr
[2] NSHC Inc., Uiwang, Republic of Korea
kisoon.yoon@gmail.com
[3] Security Algorithm Lab, Samsung SDS, Inc., Seoul, Republic of Korea
jihoon.kwon@samsung.com
[4] Sejong Cyber University, Seoul, Republic of Korea
youngho@sjcu.ac.kr

Abstract. Along with the resistance against quantum computers, isogeny-based cryptography offers attractive cryptosystems due to small key sizes and compatibility with the current elliptic curve primitives. While the state-of-the-art implementation uses Montgomery curves which facilitates efficient elliptic curve arithmetic and isogeny computations, other forms of elliptic curves can be used to produce an efficient result.
In this paper, we present the new hybrid method for isogeny-based cryptosystem using Edwards curves. Unlike the previous hybrid methods, we used Edwards curves for isogeny computations and Montgomery curves for other operations. We demonstrated that our hybrid method outperforms the previously proposed hybrid method, and is as fast as Montgomery-only implementation. We present the implementation results of Supersingular Isogeny Diffie–Hellman (SIDH) key exchange using the proposed hybrid method. Our results show that the use of Edwards curves for isogeny-based cryptosystem can be quite practical.

Keywords: Isogeny, Post-quantum cryptography, Montgomery curves, Edwards curves, SIDH

1 Introduction

The implementation of Shor's algorithm in a quantum computer at a large enough scale would break the intractability assumptions of integer factorization or discrete logarithm problems. Therefore, cryptographic construction based on RSA, DH(Diffie-Hellman key exchange), DSA(Digital Signature Algorithm) or ECC(Elliptic Curve Cryptography) will no longer be secure. These recent concerns have accelerated the field of post-quantum cryptography. Post-quantum cryptography (PQC) refers to classical cryptosystems which remain secure even in the presence of a quantum adversary. The main categories of PQC include

cryptosystems based on multivariate quadratic equations, lattices, error correcting codes, hash functions, and isogenies. The cryptosystems based on these mathematical problems are expected to be quantum-secure since there is no known quantum algorithm that can solve security base problems more efficiently than classical algorithms. Although isogeny-based cryptography is the newest in this field, due to its small key sizes compared to other cryptosystems in PQC, isogeny-based cryptosystems provide a strong candidate for the post-quantum key establishment.

The isogeny-based cryptosystem became increasingly popular after the introduction of SIDH key exchange by Jao, De Feo, and Plût in 2011 [13]. Although the cryptosystem based on isogenies on ordinary curves was first proposed by Couveignes [12] and rediscovered by Stolbunov [21], their algorithm suffered from the quantum sub-exponential attack proposed by Childs et al. [8]. Since SIDH is based on the difficulty of constructing isogenies between supersingular elliptic curves, the endomorphism ring of supersingular curves is non-commutative so that their cryptosystem resists against the attack proposed in [8]. Until now, the best known classical and quantum attacks against the underlying problem are both exponential so that SIDH has positioned itself as a promising candidate for PQC. In 2017, Supersingular Isogeny Key Encapsulation (SIKE) was submitted as one of the candidates to the NIST standardization project [1].

The potential of isogeny-based cryptosystem is that it provides significantly smaller key sizes than other PQC primitives while providing same level of security. However, its state-of-the-art implementation is slower than any other candidates in PQC. Therefore, numerous implementation results on isogeny-based cryptosystem have been proposed to increase the viability as a PQC candidate. In 2016, Azarderakhsh et al. implemented the SIDH key exchange protocol on ARM-NEON and FPGA devices [2, 16]. Costello et al. proposed faster computation methods and library for supersingular isogeny key exchange and their implementation remains state-of-the-art [11]. In 2018, Seo et al. proposed a faster modular multiplication for SIDH and SIKE [20]. Their implementation has resulted in additional speed improvements of SIDH and SIKE on ARM processors.

Regarding the implementation, it is important to choose a model of the elliptic curves that provides efficient elliptic curve arithmetic as well as isogeny computation. Due to the group structure of elliptic curves used in the isogeny-based cryptosystem, either the curve or its twist has a point of order four, which is isomorphic to Montgomery curves or Edwards curves. The state-of-the-art implementation works entirely on Montgomery curves since it provides fast point operations and efficient isogeny computation. However, whether other forms of elliptic curves are efficient as Montgomery curves is still unclear. In [9], Costello et al. also remarked that there could exist savings to be gained in a twisted Edwards version of SIDH, or some hybrid method that exploits the simple relationship between Montgomery and twisted Edwards curves. Meyer et al. proposed the hybrid SIDH scheme which exploits the fact that arithmetic in twisted Edwards curves is efficient in some instances [17]. Their method uses twisted Edwards curves for point operations and Montgomery curves for isogeny

computation. Bos et al. investigated the result of [17] and concluded that using twisted Edwards curves does not result in faster elliptic curve arithmetic in the setting of SIDH [4]. However, recently Kim et al. proposed isogeny formulas on Edwards curves for isogeny-based cryptosystem and concluded that isogenies on Edwards curves are as efficient as on Montgomery curves [15]. Their work suggests that using Edwards curves instead of twisted Edwards curves for hybrid method could result in better performance.

The aim of this work is to demonstrate the optimal combination of the usage of Montgomery curves and Edwards curves. This work is done by first implementing SIDH entirely on Edwards curves. The following list details the main contributions of this work.

- We further optimized the 4-isogeny formula on Edwards curves proposed in [15]. Our optimization of the 4-isogeny formula on Edwards curves requires $6\mathbf{M}+6\mathbf{S}$, where $\mathbf{M}$ (resp. $\mathbf{S}$) refers to field multiplication (resp. a field squaring). Additionally, we analyzed the computational cost of doubling and tripling formula on Edwards curves used in the isogeny-based cryptosystem. We conclude that except for the doubling and differential addition, computational costs on Edwards curves are as fast as on Montgomery curves.
- We optimized the 3- and 4- isogeny formulas to be suitable for our hybrid setting. Naive conversion from an Edwards curve to a Montgomery curve increases the computational costs which results in decreased performance of the cryptosystem. We efficiently combined the isogeny functions on Edwards curves and transformation from an Edwards curve to a corresponding Montgomery curve. Our isogeny functions take points on an Edwards curve as inputs and output points on a corresponding Montgomery curve and its curve coefficients. The computational costs of our functions are detailed in Table 6 on Section 5 and algorithms are listed in the appendix.
- We present the new Montgomery-Edwards hybrid version of SIDH. Previous works on such hybrid method use isogenies on Montgomery curves and elliptic curve arithmetic on twisted Edwards curves [4, 17]. However, we demonstrated that using Montgomery curves for elliptic curve arithmetic and Edwards curves for isogeny computation is faster than the previous hybrid method. Compared with the current state-of-the-art implementation, our hybrid method is faster by 1.18% for the 192-bit security level. Although the result may seem insignificant, since isogeny-based cryptosystem is slower than any other candidates in PQC, small speed improvements are meaningful in this field. The results of our experiments are presented in Section 5.

This paper is organized as follows: In Section 2, we review some special forms of elliptic curves that will be used throughout the paper. Also, the description of the SIDH protocol is presented. In Section 3, we present our optimization of 4-isogeny formula on Edwards curves and analyze the computational cost of the lower-level functions used in SIDH. The implementation result of SIDH using Edwards curves is presented in Section 4, and experiments on hybrid methods are presented in Section 5. We draw our conclusions and future work in Section 6.

2 Preliminaries

In this section, we provide the required background that will be used throughout the paper. First, we introduce the characteristic of Montgomery and Edwards curves and their relations. Then, we introduce the SIDH protocol used in the implementation.

2.1 Montgomery curves and Edwards curves

Montgomery curves Let K be a field with the characteristic not equal to 2 or 3. The Montgomery elliptic curves over K are given by the equation of the form

$$M_{a,b} : by^2 = x^3 + ax^2 + x, \tag{1}$$

where $b(a^2 - 4) \neq 0$. The j-invariant of Montgomery curves is defined as $j(M_{a,b}) = 256(a^2-3)^3/(a^2-4)$. When evaluating the isogenous curve coefficients using Vélu's formula, it is efficient to work with both projective coordinates and projective curve coefficients to avoid field inversions [11]. Let $(A : B : C) \in \mathbb{P}^2(K)$ with $C \in \bar{K}^\times$ suth that $a = A/C$ and $b = B/C$. Then $M_{a,b}$ can be expressed as

$$M_{A:B:C} : By^2 = Cx^3 + Ax^2 + Cx.$$

Arithmetic on Montgomery Curves Montgomery curves are known for fast elliptic curve arithmetic. In [18], Montgomery simplified the computations by dropping the y-coordinate. For example, let $P = (x, y)$ be a point on Montgomery curve $M_{a,b}$, where $x = X/Z$ and $y = Y/Z$. The doubling $[2]P$ can be obtained using only XZ-coordinate as described below.

$$\begin{aligned} X' &= (X - Z)^2(X + Z)^2 \\ Z' &= ((X + Z)^2 - (X - Z)^2)\left((X + Z)^2 + \frac{a-2}{4}((X + Z)^2 - (X - Z)^2)\right) \end{aligned}$$

Additionally, the Montgomery ladder is a method of computing scalar multiples of points on various forms of elliptic curves. This method is only efficient when used on a Montgomery curve [5].

Edwards Curves Edwards elliptic curves over K are defined by the equation,

$$E_d : x^2 + y^2 = 1 + dx^2y^2, \tag{2}$$

where $d \neq 0, 1$. The E_d has singular points $(1 : 0 : 0)$ and $(0 : 1 : 0)$ at infinity. In Edwards curves, the point $(0, 1)$ is the identity element, and the point $(0, -1)$ has order two. The points $(1, 0)$ and $(-1, 0)$ have order four. Since the condition that E_d always has a rational point of order four restrict the use of elliptic curves

in the Edwards model. Twisted Edwards curves are a generalization of Edwards curves proposed by Bernstein et al. in [3], to overcome such deficiency. Twisted Edwards curves are defined by the equation,

$$E_{a,d} : ax^2 + y^2 = 1 + dx^2y^2, \tag{3}$$

for distinct nonzero elements $a, d \in K$ [3]. Clearly, $E_{a,d}$ is isomorphic to an Edwards curve over $K(\sqrt{a})$. The j-invariant of Edwards curves is defined as $j(E_d) = 16(1 + 14d + d^2)^3/d(1 - d)^4$. For the same reason as in Montgomery curves, we use projective curve coefficients on Edwards curves to avoid inversion. Let $(C, D) \in \mathbb{P}^2(K)$ where $C \in \bar{K}^\times$ such that $d = D/C$. Then E_d can be expressed as

$$E_{C:D} : Cx^2 + Cy^2 = C + Dx^2y^2.$$

Arithmetic on Edwards Curves For points (x_1, y_1) and (x_2, y_2) on Edwards curves E_d, the addition of two points is defined as below, and doubling can be performed with exactly the same formula.

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2} \right).$$

In general, projective coordinates $(X : Y : Z) \in \mathbb{P}^2$ where $x = X/Z$ and $y = Y/Z$ are used for the corresponding affine point (x, y) on E_d to avoid inversion during point operations. There are many coordinate systems for Edwards curves but similar to XZ-only Montgomery arithmetic, Castryck et al. proposed YZ-only doubling formulas on Edwards curves [7]. Let $P = (x, y)$ be a point on an Edwards curve E_d. The doubling of P on Edwards curves is given by,

$$[2]P = \left(\frac{2xy}{1 + dx^2y^2}, \frac{y^2 - x^2}{1 - dx^2y^2} \right).$$

Substituting the $x^2 = (1 - y^2)/(1 - dy^2)$ on y-coordinate of $[2]P$ using the curve equation, we have,

$$\frac{y^2 - x^2}{1 - dx^2y^2} = \frac{y^2(1 - dy^2) - (1 - y^2)}{(1 - dy^2) - dy^2(1 - y^2)} = \frac{-dy^4 + 2y^2 - 1}{dy^4 - 2dy^2 + 1}.$$

Therefore, the second coordinate of $[2]P$ is expressed using only y-coordinate. Now, let $P = (X : Y : Z)$ be the projective representation of P such that $x = X/Z$ and $y = Y/Z$. Then $[2]P = (Y' : Z')$ is given by,

$$Y' = -dY^4 + 2Y^2Z^2 - Z^4$$
$$Z' = dY^4 - 2dY^2Z^2 + Z^4,$$

which is expressed using only YZ-coordinate. The tripling on Edwards curves can also be expressed in YZ-coordinate. In our implementation of SIDH, we use YZ-coordinate system on Edwards curves for computational efficiency and compatibility with XZ-coordinate on Montgomery curves.

Relation between Montgomery Curves and Edwards Curves Generally, every twisted Edwards curve over K is birationally equivalent over K to a Montgomery curve [3]. In [3], Bernstein et al. demonstrated that for a field K with $\#K \equiv 3 \pmod 4$, then every Montgomery curve over K is birationally equivalent over K to an Edwards curve. Therefore, to exploit the birationality of Montgomery and Edwards curves, we shall define K with $\#K \equiv 3 \bmod 4$ in the remainder of this paper, unless otherwise specified.

Let d be a nonzero element in K. Then every Edwards curve E_d is birationally equivalent to a Montgomery form $M_{A,B}$ via

$$(x, y) \to (u, v) = \left(\frac{1+y}{1-y}, \frac{1+y}{(1-y)x}\right), \tag{4}$$

where $A = 2(1+d)/(1-d)$ and $B = 4/(1-d)$. The inverse of the map from $M_{A,B}$ to E_d, is defined as

$$(u, v) \to (x, y) = \left(\frac{u}{v\sqrt{a}}, \frac{u-1}{u+1}\right). \tag{5}$$

where $a = (A+2)/B$ and $d = (A-2)/(A+2)$. The first coordinate in map (4) is computed by using only y-coordinate and the second coordinate in map (5) uses only u-coordinate. In projective coordinates, this map becomes remarkably simple [7]. A point $(X_M : Z_M)$ on a Montgomery curve can be transformed to the corresponding Edwards YZ-coordinates $(Y_E : Z_E)$ and vice versa:

$$(X_M : Z_M) \to (Y_E : Z_E) = (X_M - Z_M : X_M + Z_M),$$
$$(Y_E : Z_E) \to (X_M : Z_M) = (Y_E + Z_E : Z_E - Y_E).$$

Therefore, the point conversion between these two curves costs only two additions.

2.2 Supersingular Isogeny Diffie-Hellman

We recall the SIDH key exchange protocol proposed by Jao, De Feo, and Plût [13]. For more information, please refer to [13]. The notations used in this section will continue to be used throughout the paper.

SIDH protocol Fix two small prime numbers ℓ_A and ℓ_B. Let p be prime of the form $p = \ell_A^{e_A}\ell_B^{e_B} f \pm 1$ for some integer cofactor f, and e_A and e_B are positive integers such that $\ell_A^{e_A} \approx \ell_B^{e_B}$. Then we can easily construct a supersingular elliptic curve E over $\mathbb{F}_{p^2}$ of order $(\ell_A^{e_A}\ell_B^{e_B} f)^2$ [6]. We have full ℓ^e-torsion subgroup on E over $\mathbb{F}_{p^2}$ for $\ell \in \{\ell_A, \ell_B\}$ and $e \in \{e_A, e_B\}$. Choose basis $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ for the $\ell_A^{e_A}$- and $\ell_B^{e_B}$-torsion subgroups, respectively.

Suppose Alice and Bob want to exchange a secret key. Let $\{P_A, Q_A\}$ be the basis for Alice and $\{P_B, Q_B\}$ be the basis for Bob. For key generation, Alice chooses random elements $m_A, n_A \in \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$, not both divisible by ℓ_A, and computes the subgroup $\langle R_A \rangle = \langle [m_A]P_A + [n_A]Q_A \rangle$. Then using Velu's formula, Alice

computes a curve $E_A = E/\langle R_A \rangle$ and an isogeny $\phi_A : E \to E_A$ of degree $\ell_A^{e_A}$, where $ker\phi_A = \langle R_A \rangle$. Alice computes and sends $(E_A, \phi_A(P_B), \phi_A(Q_B))$ to Bob. Bob repeats the same operation as Alice so that Alice receives $(E_B, \phi_B(P_A), \phi_B(Q_A))$.

For the key establishment, Alice computes the subgroup $\langle R'_A \rangle = \langle [m_A]\phi_B(P_A) + [n_A]\phi_B(Q_A) \rangle$. By using Velu's formula, Alice computes a curve $E_{AB} = E_B/\langle R'_A \rangle$. Bob repeats the same operation as Alice and computes a curve $E_{BA} = E_A/\langle R'_B \rangle$. The shared secret between Alice and Bob is the j-invariant of E_{AB}, i.e. $j(E_{AB}) = j(E_{BA})$.

Computing large degree isogenies In the isogeny-based cryptosystem, one has to compute ℓ^e-degree isogeny ϕ. The complexity of Vélu's formulas scales linearly with the size of the kernel subgroup. Therefore, we decompose an isogeny of degree ℓ^e into e isogenies of degree ℓ for computational efficiency.

Given a cyclic subgroup $\langle R \rangle \in E[\ell^e]$ of order ℓ^e, let ϕ be the isogeny from E to $E/\langle R \rangle$, with $ker\phi = \langle R \rangle \in E[\ell^e]$. The isogeny ϕ is computed as the composition of e isogenies of degree ℓ by Velu's formula [22]. Starting with by setting $E = E_0$ and $R = R_0$, compute $E_{i+1} = E_i/\langle \ell^{e-i-1} R_i \rangle$ for $0 \leq i < e$ [13]. For each iteration, compute ℓ-isogeny $\phi_i : E_i \to E_{i+1}$, with $ker\phi_i = \langle \ell^{e-i-1} R_i \rangle$ of order ℓ, and set $R_{i+1} = \phi_i(R_i)$. The point R_i is an ℓ^{e-i} torsion point so that $[\ell^{e-i-1}]R_i$ has order ℓ. Therefore, combining $\phi = \phi_{e-1} \circ \cdots \circ \phi_0$ we can obtain isogeny ϕ of degree ℓ^e having $\langle R \rangle$ as a kernel.

3 Formulas for Constructing Isogeny-based Cryptosystems

In this section, we present the computational cost of lower-level functions which are used as building blocks of SIDH. In order to avoid inversions during the computation, not only projective coordinates but projective curve coefficients are also used [11]. The formulas presented in this sections are the result of considering such circumstances. Additionally, since $\ell_A = 2$ and $\ell_B = 3$ are the common choice for implementing isogeny-based systems, we consider formulas focusing on the doubling (resp. tripling) and 4-isogeny (resp. 3-isogeny). For projective 4-isogeny formula on Edwards curves, we optimized the formula presented in [15].

3.1 Elliptic Curves Arithmetic in SIDH

The elliptic curves arithmetic on Edwards curves using projective curve coefficients is similar to the case when using twisted Edwards curves. Unlike the currently used ECC, elliptic curves used in the isogeny-based cryptosystem are not fixed but changes as moving around an isogeny class. Therefore, the formula used in SIDH cannot be optimized for specific curve coefficients.

Doubling Let $P = (x, y)$ be a point on an Edwards curve E_d defined as in equation (2). Let $d = D/C$, $x = X/Z$, and $y = Y/Z$. For $P = (Y : Z)$ in

projective coordinates, the doubling of P gives $[2]P = (Y' : Z')$, where Y' and Z' are defined as

$$Y' = -DY^4 + 2CY^2Z^2 - CZ^4$$
$$Z' = DY^4 - 2DY^2Z^2 + CZ^4.$$

The above equations can be computed as,

$$t_0 = Y^2, \quad t_1 = Z^2, \quad t_2 = D \cdot Y^2, \quad t_3 = C \cdot Z^2,$$
$$t_4 = t_0 - t_1, \quad t_5 = t_2 - t_3, \quad t_4 = t_4 \cdot t_5,$$
$$t_5 = C - D, \quad t_5 = t_5 \cdot t_0 \cdot t_1$$
$$Y' = t_5 - t_4, \quad Z' = t_5 - t_4,$$

The cost of this computation is $5\mathbf{M}+2\mathbf{S}$.

Tripling For $P = (Y : Z)$ on an Edwards curve E_d represented in projective coordinates, the tripling of P gives $[3]P = (Y' : Z')$, where Y' and Z' are defined as

$$Y' = Y(D^2Y^8 - 6CDY^4Z^4 + 4C^2Y^2Z^6 + 4CDY^2Z^6 - 3C^2Z^8)$$
$$Z' = Z(C^2Z^8 - 6CDY^4Z^4 + 4D^2Y^6Z^2 + 4CDY^6Z^2 - 3D^2Y^8).$$

Let $F = Y' + Z'$ and $G = Y' - Z'$. Then F and G can be written as,

$$F = Y' + Z' = (DY^2(Y^2 - 2YZ) + CZ^2(2YZ - Z^2))^2(Y + Z)$$
$$G = Y' - Z' = (DY^2(Y^2 + 2YZ) - CZ^2(2YZ + Z^2))^2(Y - Z).$$

After computing F and G, the results Y' and Z' can be obtained by computing $F + G$ and $F - G$. The cost of tripling is $7\mathbf{M}+5\mathbf{S}$.

Differential addition The SIDH starts by computing $R = [m]P + [n]Q$ for chosen basis P and Q and a secret key (m, n). Without loss of generality, we may assume that m is invertible, and compute $R = P + [m^{-1}n]Q$. This can be done by using the Montgomery ladder which requires computing differential additions as a subroutine. In the proof of [14], Edwards curves have the similar formula, and we briefly introduce here.

Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be two different points on the Edwards curve E_d. Let $P_1 + P_2 = (x_3, y_3)$ and $P_1 - P_2 = (x_4, y_4)$. The addition formula on Edwards curves gives

$$y_3(1 - dx_1x_2y_1y_2) = y_1y_2 - x_1x_2,$$
$$y_4(1 + dx_1x_2y_1y_2) = y_1y_2 + x_1x_2.$$

By multiplying the two equations above, we obtain

$$y_3y_4(1 - d^2x_1^2x_2^2y_1^2y_2^2) = y_1^2y_2^2 - x_1^2x_2^2. \tag{6}$$

Substitute $x_1^2 = \frac{1-y_1^2}{1-dy_1^2}$ and $x_2^2 = \frac{1-y_2^2}{1-dy_2^2}$, and let $y_i = Y_i/Z_i$ for $i = 1, 2, 3$. Then the equation (6) can be rewritten as,

$$\frac{Y_3}{Z_3} = -\frac{(DY_1^2Y_2^2 - CY_2^2Z_1^2 - CY_1^2Z_2^2 + CZ_1^2Z_2^2)Z_4}{(DY_1^2Y_2^2 - DY_2^2Z_1^2 - DY_1^2Z_2^2 + CZ_1^2Z_2^2)Y_4} \tag{7}$$

where $d = D/C$. Using the equation (7), the doubling-and-addition for ladder computation on an Edwards curve costs 13**M**+4**S**.

The *j*-invariant In SIDH, the j-invariant of the image curve E_{AB} is used as a shared secret between two parties. The j-invariant of an Edwards curve E_d is defined as

$$j(E_d) = \frac{16(C^2 + 14CD + D^2)^3}{CD(C - D)^4}$$

where $d = D/C$. The computational cost of the j-invariant is 3**M**+4**S**+1**I**, where **I** refers to field inversion.

3.2 Isogenies on Edwards Curves

Projective 3-isogenies The formulas for odd-degree isogenies on Edwards curves were first proposed by Moody and Shumow in [19]. They proposed a 'multiplicative' isogeny formula on Edwards curves which resulted in better algebraic complexity than 'additive' form of Vélu's formula. Let $P = (\alpha, \beta)$ be a 3-torsion point on an Edwards curve E_d. Then ϕ is a 3-isogeny from E_d to E'_d given by the equation,

$$\psi(x, y) = \left(\frac{x}{\beta^2}\frac{\beta^2x^3 - \alpha^2y^2}{1 - d^2\alpha^2\beta^2x^2y^2}, \frac{y}{\beta^2}\frac{\beta^2y^2 - \alpha^2x^2}{1 - d^2\alpha^2\beta^2x^2y^2}\right),$$

where $d' = \beta^8d^3$ and having $\langle P \rangle$ as a kernel. Later, Kim et al. optimized the isogeny formula presented in [19] using projective coordinates, projective curve

coefficients, and division polynomial, for the use in the isogeny-based cryptosystem [15]. Let $P = (\alpha, \beta)$ be a 3-torsion point on Edwards curve E_d, where $\beta = Y_3/Z_3$. Let $\phi : E_d \to E_{d'}$ be a 3-isogeny generated by a kernel $\langle P \rangle$, so that $E_{d'} = E_d/\langle P \rangle$. Let $(Y : Z)$ be the additional input and $(Y' : Z')$ be its corresponding image such that $\phi(Y : Z) = (Y' : Z')$. The projective version of 3-isogeny on Edwards curves is given as

$$(Y' : Z') = (Y(Z^2Y_3^2 + 2Z^2Y_3Z_3 + Y^2Z_3^2) : Z(Z^2Y_3^2 + 2Y^2Y_3Z_3 + Y^2Z_3^2)). \quad (8)$$

The curve coefficients of the isogenous curve $E_{d'}$ are,

$$D' = (Z_3 + 2Y_3)^3 Z_3, \qquad C' = (2Z_3 + Y_3)^3 Y_3, \quad (9)$$

where $d' = D'/C'$. The computational cost for evaluating 3-isogeny and curve coefficients is $6\mathbf{M}+5\mathbf{S}$ [15].

Projective 4-isogenies In [15], Kim et al. proposed 4-isogeny formula on Edwards curves by exploiting the efficiency of transforming Edwards curves and Montgomery curves. They combined the transformation between the two curves and 4-isogeny on Montgomery curves.

Let $P = (\alpha, \beta)$ be a 4-torsion point on an Edwards curve E_d, where $\beta = Y_4/Z_4$. Let $\phi : E_d \to E_{d'}$ be a 4-isogeny generated by a kernel $\langle P \rangle$, so that $E_{d'} = E_d/\langle P \rangle$. Let $(Y : Z)$ be the additional input and $(Y' : Z')$ be its corresponding image such that $\phi(Y : Z) = (Y' : Z')$. The projective version of 4-isogeny on Edwards curves is given as

$$\begin{aligned} Y' &= (Z^2Y_4^2 + Y^2Z_4^2)YZ(Y_4 + Z_4)^2, \\ Z' &= (Z^2Y_4^2 + Y^2Z_4^2)^2 + 2Y^2Z^2Y_4Z_4(Y_4^2 + Z_4^2). \end{aligned} \quad (10)$$

The curve coefficients of the isogenous curve $E_{d'}$ are,

$$\begin{aligned} D' &= 8Y_4Z_4(Y_4^2 + Z_4^2), \\ C' &= (Y_4 + Z_4)^4. \end{aligned} \quad (11)$$

where $d' = D'/C'$. For the computational cost, we further optimized the result presented in [15]. Algorithm 1 shows an efficient way to compute 4-isogeny and its corresponding curve coefficients. The computational cost for Algorithm 1 is $6\mathbf{M}+6\mathbf{S}$. The algorithm for computing 4-isogeny is presented in the appendix.

3.3 Summary on the Lower-level Functions

We compared the computational cost of point and isogeny operations on Montgomery and Edwards curves in Table 1. The computational costs of 4-isogeny

(resp. 3-isogeny) represented in Table 1 is the combined computational costs of `get_4_isog` and `eval_4_isog` (resp. `get_3_isog` and `get_3_isog`). As shown in Table 1, except for the doubling and differential addition for computing Montgomery ladder, operations on Edwards curves are as efficient as on Montgomery curves.

Table 1: Computational costs of lower-level functions on Montgomery and Edwards curves

	Montgomery Curves	Edwards Curves
Differential Addition	6**M**+4**S**	13**M**+4**S**
Doubling	4**M**+2**S**	5**M**+2**S**
4-isogeny	6**M**+6**S**	6**M**+6**S**
Tripling	7**M**+5**S**	7**M**+5**S**
3-isogeny	6**M**+5**S**	6**M**+5**S**
j-invariant	3**M**+4**S**+1**I**	3**M**+4**S**+1**I**

4 SIDH on Edwards Curves

In this section, we present the implementation result of SIDH entirely on Edwards curves. From this section, the field K is fixed as $K = \mathbb{F}_{p^2}$, where p is prime, and $\mathbb{F}_{p^2} = \mathbb{F}_{p^2}[X]/(X^2+1)$. For the prime p, we used the 503-bit prime $p_{503} = 2^{250} \cdot 3^{159} - 1$ and the 751-bit prime $p_{751} = 2^{372} \cdot 3^{239} - 1$, presented in [1,10], which aims at 128-bit and 192-bit security level, respectively.

4.1 Parameters for SIDH on Edwards curves

The starting curve of our implementation is the supersingular Edwards curve,

$$E_{-1}/F_{p^2} : x^2 + y^2 = 1 - x^2y^2.$$

This curve is birationally equivalent to the Montgomery curve $M_{0,1} : y^2 = x^3+x$ over $\mathbb{F}_{p^2}$, which are used as a public curve in [1,10]. The reason for choosing this curve is to compare the exact differences caused by isogeny computations and elliptic curve arithmetic when using Edwards curves. As a result, the public keys we used on the Edwards curve correspond to the public key used in [1,10].

4.2 Strategies on Edwards curves

The most expensive part when computing isogeny is the computation of the point $[\ell^{e-i-1}]R$ for each iteration. Computing $[\ell^{e-i-1}]R$ costs at most e multiplication by the scalar ℓ, which is repeated e times. Therefore, naive implementation requires a total of $O(e^2)$ operations. In [13], Jao, De Feo, and Plût proposed an efficient way that decomposes isogeny computation which reduces to $O(e \log e)$ multiplication by the scalar ℓ and ℓ-isogeny evaluations.

Recall that computation of ℓ^e-isogeny is composed of e number of isogenies of degree ℓ. Computation of an ℓ^e-isogeny can be guided by a full binary tree on $e-1$ nodes, as depicted in Figure 2 in [13]. The goal is to compute all leaves of a tree, namely, $[\ell^{e-i-1}]R$ for $0 \leq i < e$. Depending on the relative costs of multiplication-by-ℓ and ℓ-isogeny evaluation, the algorithm moves along the left edge or right edge and selects the optimal strategy. In [11], Costello et al. stored a strategy in a list L of integers of length equal to the total number of leaves. The i-th entry $L[i]$ corresponds to the number of scalar-multiplication steps along the left edge that are needed in a strategy of i leaves to reach the root of the next sub-strategy. In [1], they stored a strategy by first labeling the node the number of leaves to its right. Then, they stored the labels as they encountered when walking the tree in depth-first left-first order.

To obtain the strategies on Edwards curves, we first computed the relative costs of scalar multiplication-by-ℓ and ℓ-isogeny evaluation. The ratio between 4-isogeny evaluation and multiplication-by-4 was $(15,494/2 \cdot 10,470)$, and the ratio between 3-isogeny evaluation and multiplication-by-3 was $(8,856/17281)$. The results were obtained on an Intel Core i7-6700 (Skylake) at 3.40 GHz, running Ubuntu 16.04 LTS, averaging 10^6 executions. The results are represented in the number of clock cycles. From these ratios, we obtained the strategies by using the magma script provided in [10].

4.3 Computing isogenies

When computing the ℓ^e-degree isogeny, the public key of the opponents are also evaluated for each iteration. In order to compute the ladder efficiently in the key establishment stage of SIDH, not only the basis (P, Q) but their differences $P-Q$ are also considered and stored as a public key [1,10]. The implementations in [1,10] first compute the coefficients of the isogenous curve and evaluate isogeny on the public key $(P, Q, P-Q)$ for $0 \leq i < e$. Therefore, it is efficient to split the computation of ℓ-degree isogeny in [15] and Algorithm 1 into the function where it computes the coefficients, i.e. `get_ℓ_isog`, and the function where it evaluates, i.e. `eval_ℓ_isog` as in [1,10]. Since `eval_ℓ_isog` is called more than `get_ℓ_isog` for each round, it is important to reduce the cost of the evaluation function for a better performance.

4.4 Implementation Result

To evaluate the performance, the algorithms are implemented in C language. We used the SIDH library version 3.0 for SIDH on Montgomery curves. To make an exact comparison, the field operations implemented in the SIDH library were used for both curves which are written in x64 assembly. As a result, the difference in performance lies purely in the choice of the model of an elliptic curve. All cycle counts were obtained on one core of an Intel Core i7-6700 (Skylake) at 3.40 GHz, running Ubuntu 16.04 LTS. For compilation, we used GNU GCC version 5.4.0.

We first measure the field operations over $\mathbb{F}_{p^2}$ to examine the ratio between each operation. To this end, each field operations were repeated 10^8 times for

each prime field. Table 2 summarizes the average cycle counts of field operations over $\mathbb{F}_{p^2}$.

Table 2: Cycle counts of the field operations over $\mathbb{F}_{p^2}$.

Field size	Addition	Subtraction	Multiplication	Squaring	Inversion
p_{503}	48	40	530	448	119,389
p_{751}	79	63	930	759	332,926

As in Table 2, 1**S** equals approximately 0.8**M**, for both 503-bit prime and 751-bit prime. Base on the above result, Table 3 shows the computational costs and corresponding cycle counts of ℓ-isogeny and multiplication-by-ℓ, when using Montgomery curves and Edwards curves. For each operation, we report the average cycles of 10^8 times. Note that the number of field multiplications and squarings are identical on both curves except for the multiplication-by-2 map (doubling). Therefore, to better represent the results, we also counted field additions and subtractions. In Table 3, **a** (resp. **s**) refers to field addition (resp. subtraction).

Table 3: Implementation results of operations on Montgomery curves and Edwards curves (cc represents the number of clock cycles).

	Montgomery Curve		Edwards Curve (This Work)	
	p_{503}	p_{751}	p_{503}	p_{751}
Doubling	4**M**+2**S**		5**M**+2**S**	
	3,085 cc	5,594 cc	3,614 cc	6,541 cc
`get_4_isog`	4**S**+4**a**+1**s**		4**S**+2**a**+2**s**	
	1,926 cc	3,396 cc	1,888 cc	3,355 cc
`eval_4_isog`	6**M**+2**S**+3**a**+3**s**		6**M**+2**S**+4**a**+3**s**	
	4,145	7,498 cc	4,210 cc	7,670 cc
Tripling	7**M**+5**S**+3**a**+7**s**		7**M**+5**S**+4**a**+8**s**	
	6,043 cc	11,078 cc	6,326 cc	11,056 cc
`get_3_isog`	2**M**+3**S**+12**a**+3**s**		2**M**+3**S**+7**a**+4**s**	
	2,955 cc	5,157 cc	2,930 cc	4,890 cc
`eval_3_isog`	4**M**+2**S**+2**a**+2**s**		4**M**+2**S**+3**a**+3**s**	
	3,131 cc	5,478 cc	3,234 cc	5,602 cc

The `get_4_isog` and `get_3_isog` are a function that computes the coefficients of the isogeneous curve. The `eval_4_isog` and `eval_3_isog` are a function that evaluates the isogeny on a given input point. Combining the results from Table 2 and Table 3, Table 4 compares the performance of SIDH when using Edwards curves and Montgomery curves.

Table 4: Performance results of SIDH implementation using Edwards and Montgomery Curves. The results were rounded to the nearest 10^3 clock cycles.

	Montgomery Curve [1]		Edwards Curve (This Work)	
	p_{503}	p_{751}	p_{503}	p_{751}
Alice's Keygen	7,818	21,792	9,023	24,671
Bob's Keygen	8,762	24,780	9,427	26,397
Alice's Shared Key	6,413	17,870	7,672	21,990
Bob's Shared Key	7,396	21,322	8,033	22,664
Total	30,389	85,764	34,155	95,668

Remark. The SIDH implementation on Edwards curves might be improved when using the addition chain proposed in [4] to compute the kernels instead of the Montgomery ladder.

5 Montgomery-Edwards Hybrid SIDH

Due to the differential addition and doubling on Edwards curves, implementing SIDH using Edwards curves does not result in faster computation. However, as demonstrated in Section 3, some combination of the use of Montgomery and Edwards curves might speed-up the current result. In this section, we present the most optimal hybrid SIDH method that efficiently combines the usage of both curves. We deduce our result by measuring the performance of various combinations. As denoted in Table 1 and Table 3, Montgomery curves provides fast elliptic curve arithmetic while Edwards curves provide fast isogeny computation.

5.1 Switching between Montgomery and Edwards curves

In this subsection, we analyze the additional cost required during the transformation process. When converting between two curves, the conversion of the curve coefficients should also be considered. Let A_M, B_M, and C_M be the projective curve coefficients of the Montgomery curve $M_{A_M:B_M:C_M}$ and D_E and C_E be the projective curve coefficients of the corresponding Edwards curve $E_{C_E:D_E x}$. Fortunately, the arithmetic on Montgomery curve uses only the curve coefficients A_M and C_M, which correspond to Edwards curve coefficients C_E and D_E. Instead of storing A_M and C_M, the implementation in [1,10] stores $A_M + 2C_M$ and $4C_M$ for doubling and $A_M + 2C_M$ and $A_M - 2C_M$ for tripling, for computational efficiency.

Montgomery to Edwards Since the conversion of a Montgomery curve to an Edward curve only occurs after elliptic curve point operations in our setting, we solely consider the case. Let $(X_M : Z_M)$ be the projective point on a Montgomery curve $M_{A_M:B_M:C_M}$ and $(Y_E : Z_E)$ be the projective point on an Edwards

curve $E_{C_E:D_E}$. The transformation from Montgomery curve to Edwards curve on Alice's side is as follows:

$$\begin{aligned}(X_M : Z_M) \to (Y_E : Z_E) &= (X_M - Z_M : X_M + Z_M)\\ (A' : C') \to (C_E : D_E) &= (A' : A' - C')\end{aligned}$$

where $A' = A_M + 2C_M$ and $C' = 4C_M$.

The transformation from Montgomery curve to Edwards curve on Bob's side is as follows:

$$\begin{aligned}(X_M : Z_M) \to (Y_E : Z_E) &= (X_M - Z_M : X_M + Z_M)\\ (A' : C') \to (C_E : D_E) &= (A' : C')\end{aligned}$$

where $A' = A_M + 2C_M$ and $C' = A_M - 2C_M$. There is no additional cost in the conversion of the curve coefficients.

Edwards to Montgomery Since the conversion of an Edwards curve to a Montgomery curve only occurs after evaluating isogenies in our setting, we solely consider the case. Let $(X_M : Z_M)$ be the projective point on a Montgomery curve $M_{A_M:B_M:C_M}$ and $(Y_E : Z_E)$ be the projective point on an Edwards curve $E_{C_E:D_E}$. The transformation from an Edwards curve to a Montgomery curve on Alice's side is as follows:

$$\begin{aligned}(Y_E : Z_E) \to (X_M : Z_M) &= (Y_E - Z_E : Z_E - Y_E)\\ (C_E : D_E) \to (A' : C') &= (4C_E : 4(C_E - D_E))\end{aligned}$$

where $A' = A_M + 2C_M$ and $C' = 4C_M$.

The transformation from an Edwards curve to a Montgomery curve on Bob's side is as follows:

$$\begin{aligned}(Y_E : Z_E) \to (X_M : Z_M) &= (Y_E - Z_E : Z_E - Y_E)\\ (C_E : D_E) \to (A' : C') &= (4C_E : 4D_E)\end{aligned}$$

where $A' = A_M + 2C_M$ and $C' = A_M - 2C_M$. However, converting points on Edwards curves to the corresponding Montgomery curves after evaluating isogenies requires no additional cost due to the computational structure presented in [15] and in Algorithm 1. Rather, it reduces the computational cost of isogeny functions. For example, in Algorithm 1, to compute the image $(Y' : Z')$ on a isogenous curve, the algorithms computes $F = Y' + Z'$ and $G = Y' - Z'$ and obtain Y' and Z'. By omitting the Step 22 and 23 and changing the Step 19 into $t_6 - t_2$, the outputs are the image of an input point on a corresponding Montgomery curve. Therefore, the cost of computing isogenies is reduced by 1**a**+1**s** on Edwards curves in the hybrid setting.

The Algorithm 2 in appendix, describes ways to compute curve coefficients of the 3-isogenous image curve, given 3-torsion points on an Edwards curve. Let $P = (Y_3 : Z_3)$ be a 3-torsion point on an Edwards curve E_d and $\phi : E_d \to E_{d'}$, where $ker\phi = \langle P \rangle$. The Algorithm 2 outputs curve coefficients of a Montgomery curve $M_{a,b}$, where $M_{a,b}$ is birationally equivalent to $E_{d'}$. For additional curve point $Q = (Y : Z)$ on an Edwards curve E_d, The Algorithm 3 outputs $Q' = (X' : Z')$ on a Montgomery curve $M_{a,b}$. Similarily, the Algorithm 4 describes ways to compute curve coefficients of the 4-isogenous image curve, given 4-torsion points on an Edwards curve. Let $P = (Y_4 : Z_4)$ be a 4-torsion point on an Edwards curve E_d and $\phi : E_d \to E_{d'}$, where $ker\phi = \langle P \rangle$. The Algorithm 4 outputs curve coefficients of a Montgomery curve $M_{a,b}$, where $M_{a,b}$ is birationally equivalent to $E_{d'}$. For additional curve point $Q = (Y : Z)$ on an Edwards curve E_d, The Algorithm 5 outputs $Q' = (X' : Z')$ on a Montgomery curve $M_{a,b}$.

5.2 Implementation results of hybrid SIDH

As analyzed in Table 1 and Table 3, Montgomery curves provide fast elliptic curve arithmetic while Edwards curves provide fast isogeny computation. Additionally, the conversion between Montgomery curve and Edwards curves is almost cost-free and has a particular advantage when transforming Edwards curves to Montgomery curves. Therefore, we can think of a hybrid method that exploits the merits of each curve. In order to evaluate the best combination, we implemented the SIDH in 2 additional ways.

Montgomery-only-for-ladder This implementation computes the kernel $\langle P + [m]Q \rangle$ on Montgomery curves and other operations on Edwards curves. Since Bob repeats the same operation as Alice in the SIDH protocol, we shall describe the implementation on Alice's side.

On the choice of her secret key m_A, Alice first computes the kernel $\langle P_A + [m_A]Q_A \rangle$ on a Montgomery curve. Alice then converts her kernel as well as Bob's public key $(P_B, Q_B, P_B - Q_B)$ to points on an Edwards curve. The $\ell_A^{e_A}$-isogeny is computed on Edwards curves. The results are converted back to points on a Montgomery curve. Upon the receiving of $(E_B, \phi_B(P_A), \phi_B(Q_A), \phi_B(P_A - Q_A))$ from Bob, Alice again computes $\langle \phi_B(P_A) + [m_A]\phi_B(Q_A) \rangle$ on a Montgomery curve and converts to point on the corresponding Edwards curves. As the j-invariant is used as a shared secret, converting back to Montgomery curves is not required. The shared secret is the j-invariant of E_{AB} in Edwards form.

Our implementation used $E_{-1} : x^2 + y^2 = 1 - x^2y^2$ and $M_{0,1} : y^2 = x^3 + x$ over $\mathbb{F}_{p^2}$ as the base curves. For the efficiency, Alice's and Bob's public key $(P_A, Q_A, P_A - Q_A)$ and $(P_B, Q_B, P_B - Q_B)$ may already be converted into points on the Edwards curve, $E_{-1} : x^2 + y^2 = 1 - x^2y^2$.

Edwards-only-for-isogeny This implementation exploits the efficiency of isogeny computation on Edwards curves. The Edwards-only-for-isogeny method computes the entire isogenies on Edwards curves and the entire points operations on

Montgomery curves. Since Bob repeats the same operation as Alice in the SIDH protocol, we shall describe the implementation on Alice's side.

On the choice of her secret key m_A, Alice first computes the kernel $\langle R\rangle = \langle P_A + [m_A]Q_A\rangle$ on a Montgomery curve. Alice then converts Bob's public key $(P_B, Q_B, P_B - Q_B)$ to points on an Edwards curve. The computation of $[\ell^{e-i-1}]R$ is obtained using Montgomery curves. The results are converted to points on Edwards curves and isogeny computations on $[\ell^{e-i-1}]R, P_B, Q_B$, and $P_B - Q_B$ for each i are followed. After isogeny computations, the points are converted back to points on a Montgomery curve. Upon receiving points from Bob, Alice follows the identical conversion step and computes the shared secret.

Identical to the previous implementation, we used $E_{-1} : x^2 + y^2 = 1 - x^2y^2$ and $M_{0,1} : y^2 = x^3 + x$ over $\mathbb{F}_{p^2}$ as the base curves. For the efficiency, Alice's and Bob's public key $(P_A, Q_A, P_A - Q_A)$ and $(P_B, Q_B, P_B - Q_B)$ may already be converted into points on the Edwards curve, $E_{-1} : x^2 + y^2 = 1 - x^2y^2$.

Implementation results In the Edwards-only-for-isogeny implementation, the multiplication-by-ℓ is computed on Montgomery curves and isogeny is evaluated on Edwards curves. Therefore, a new strategy is needed for Edwards-only-for-isogeny implementation. The Table 5 compares the relative costs of scalar multiplication-by-ℓ and ℓ-isogeny evaluation. The `Alice` in Table 5 represents the ratio between 4-isogeny evaluation and multiplication-by-4. The `Bob` in Table 5 represents the ratio between 3-isogeny evaluation and multiplication-by-3. The results were obtained by averaging 10^6 executions. Base on this ratio, the strategy for the hybrid SIDH was obtained by running the magma script provided in [10].

Table 5: Relative cost of scalar multiplication and isogeny evaluation on various circumstances.

	Montgomery Curve		Edwards Curve		Edwards-to-Montgomery	
	p_{503}	p_{751}	p_{503}	p_{751}	p_{503}	p_{751}
`Alice`	0.59	0.67	0.57	0.58	0.60	0.67
`Bob`	0.51	0.49	0.52	0.49	0.52	0.50

The Table 6 shows the computational costs of key generation stage in SIDH on Montgomery curves presented in [10], and the hybrid methods described in Section 5.2.1 and Section 5.2.2. The computational costs and the form of the elliptic curve used in the computation are analyzed in Table 6. The `Mont.` and `Ed.` represent Montgomery curves and Edwards curves, respectively. In Table 6, the computational cost of `Kernel` means the computational cost of differential addition.

In Montgomery-only-for-ladder implementation (Section 5.2.1), the transformation from a Montgomery curve to an Edwards curve only occurs after comput-

ing the kernel, $\langle P_A + [m_A]Q_A \rangle$ (or $\langle P_B + [m_B]Q_B \rangle$ for Bob). In Edwards-only-for-isogeny implementation (section 5.2.2), the transformation from a Montgomery curve to an Edwards curve occurs after doubling or tripling. The transformation from an Edwards curve to a Montgomery curve occurs after isogeny evaluation in order to perform doubling or tripling on next iteration. The computational cost of curve conversion is included when computing isogenies and coefficients of the image curve and its method is detailed in Algorithm 2 to Algorithm 5.

Table 6: Computational cost of key generation stage in SIDH on various circumstances.

		[10]	5.2.1	5.2.2
Alice	Kernel	6**M**+4**S**	13**M**+4**S**	6**M**+4**S**
		Mont.	Mont.	Mont.
	Curve Conversion	-	1**s**	-
	Point Conversion	-	1**a**+1**s**	-
	Doubling	4**M**+2**S**	5**M**+2**S**	4**M**+2**S**
		Mont.	Ed.	Mont.
	Curve Conversion	-	-	1**s**
	Point Conversion	-	-	1**a**+1**s**
	get_4_isog	4**S**+4**a**+1**s**	4**S**+2**a**+2**s**	4**S**+4**a**+1**s**
		Mont.	Ed.	Ed.
	eval_4_isog	6**M**+2**S**+3**a**+3**s**	6**M**+2**S**+4**a**+3**s**	6**M**+2**S**+3**a**+2**s**
		Mont.	Ed.	Ed.
Bob	Kernel	6**M**+4**S**	13**M**+4**S**	6**M**+4**S**
		Mont.	Mont.	Mont.
	Curve Conversion	-	-	-
	Point Conversion	-	1**a**+1**s**	-
	Tripling	7**M**+5**S**+3**a**+7**s**	7**M**+5**S**+4**a**+8**s**	7**M**+5**S**+3**a**+7**s**
		Mont.	Ed.	Mont.
	Curve Conversion	-	-	-
	Point Conversion	-	-	1**a**+1**s**
	get_3_isog	2**M**+3**S**+12**a**+3**s**	2**M**+3**S**+7**a**+4**s**	2**M**+3**S**+8**a**+4**s**
		Mont.	Ed.	Ed.
	eval_3_isog	4**M**+2**S**+2**a**+2**s**	4**M**+2**S**+3**a**+3**s**	4**M**+2**S**+2**a**+2**s**
		Mont.	Ed.	Ed.

The results of the implementations are organized in Table 7. For each implementation, we report the average cycles of 10^8 times. The roman number I refers to SIDH implementation entirely on Montgomery curves and II refers to SIDH implementation entirely on Edwards curves. The roman number III and IV refers to Montgomery-only-for-ladder implementation and Edwards-only-for-isogeny implementation, respectively.

Table 7: Performance results of the hybrid SIDH implementation. The results were rounded to the nearest 10^3 clock cycles.

	I		II		III		IV	
	p_{503}	p_{751}	p_{503}	p_{751}	p_{503}	p_{751}	p_{503}	p_{751}
Alice's Keygen	7,712	21,820	8,878	24,769	8,245	23,002	7,660	21,583
Bob's Keygen	8,610	24,755	9,305	26,495	8,658	24,634	8,569	24,331
Alice's Shared Key	6,302	17,965	7,401	20,912	6,809	19,358	6,252	17,947
Bob's Shared Key	7,498	21,129	8,169	22,599	7,398	20,947	7,389	20,794
Total	30,122	85,669	33,753	94,775	32,110	87,941	29,870	84,655

As shown in Table 7, IV which uses Montgomery curves for elliptic curve arithmetic and Edwards curves for isogeny computation is faster than any other hybrid method. The implementation IV is 0.83% and 1.18% faster than the implementation using only Montgomery curve (i.e. I) on p_{503} and p_{751}, respectively. The reason is that computational costs of `get_3_isog` and `eval_4_isog` is reduced in IV. Although the reduction is minimal, since the functions are called multiple times this resulted in an increase in the overall speed. Therefore, for a better performance in hybrid SIDH setting, it is recommended to use Montgomery curves for point operations and Edwards curves for computing isogenies.

6 Conclusion and Future Work

In this paper, we proposed the new hybrid method for SIDH implementation. Although using Edwards curves does not result in better SIDH performance, we noticed that Edwards curves have an advantage on isogeny computation and examined the optimal combination of using Montgomery and Edwards curves through various experiments. We demonstrated that using Montgomery curve for elliptic curve arithmetic and Edwards curves for isogeny is faster than currently proposed hybrid method. We believe that we were the first to propose such a hybrid SIDH method which uses Edwards curves. Moreover, compared with the standard SIDH, our Montgomery-Edwards hybrid SIDH is faster than the standard SIDH by 0.83% and 1.18% for 128-bit and 192-bit security level, respectively. The proposed hybrid method in this paper is meaningful in two ways – i) its speed is as fast as the current state-of-the-art implementation, ii) it is faster than previously proposed Montgomery-twisted Edwards version of hybrid SIDH. We emphasize the fact that using Edwards curves on isogeny-based cryptosystem can be quite practical. Additionally, since currently proposed isogeny-based cryptosystems have the same structure as in SIDH, our implementation results can also be applied.

References

1. Azarderakhsh, R., Campagna, M., Costello, C., De Feo, L., Hess, B., Jalali, A., Jao, D., Koziel, B., LaMacchia, B., Longa, P., et al.: Supersingular isogeny key

encapsulation. submission to the nist post-quantum standardization project, 2017
2. Azarderakhsh, R., Koziel, B., Langroudi, S.H.F., Kermani, M.M.: Fpga-sidh: High-performance implementation of supersingular isogeny diffie-hellman key-exchange protocol on fpga. IACR Cryptology ePrint Archive 2016, 672 (2016)
3. Bernstein, D.J., Birkner, P., Joye, M., Lange, T., Peters, C.: Twisted edwards curves. In: International Conference on Cryptology in Africa. pp. 389–405. Springer (2008)
4. Bos, J., Friedberger, S.: Arithmetic considerations for isogeny based cryptography. IEEE Transactions on Computers (2018)
5. Bos, J.W., Costello, C., Longa, P., Naehrig, M.: Selecting elliptic curves for cryptography: An efficiency and security analysis. Journal of Cryptographic Engineering 6(4), 259–286 (2016)
6. Bröker, R.: Constructing supersingular elliptic curves. J. Comb. Number Theory 1(3), 269–273 (2009)
7. Castryck, W., Galbraith, S.D., Farashahi, R.R.: Efficient arithmetic on elliptic curves using a mixed edwards-montgomery representation. IACR Cryptology ePrint Archive 2008, 218 (2008)
8. Childs, A., Jao, D., Soukharev, V.: Constructing elliptic curve isogenies in quantum subexponential time. Journal of Mathematical Cryptology 8(1), 1–29 (2014)
9. Costello, C., Hisil, H.: A simple and compact algorithm for sidh with arbitrary degree isogenies. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 303–329. Springer (2017)
10. Costello, C., Longa, P., Naehrig, M.: Sidh library (2016)
11. Costello, C., Longa, P., Naehrig, M.: Efficient algorithms for supersingular isogeny diffie-hellman. In: Annual Cryptology Conference. pp. 572–601. Springer (2016)
12. Couveignes, J.M.: Hard homogeneous spaces. IACR Cryptology ePrint Archive 2006, 291 (2006)
13. Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: International Workshop on Post-Quantum Cryptography. pp. 19–34. Springer (2011)
14. Justus, B., Loebenberger, D.: Differential addition in generalized edwards coordinates. In: International Workshop on Security. pp. 316–325. Springer (2010)
15. Kim, S., Yoon, K., Kwon, J., Hong, S., Park, Y.H.: Efficient isogeny computations on twisted edwards curves. Security and Communication Networks 2018 (2018)
16. Koziel, B., Jalali, A., Azarderakhsh, R., Jao, D., Mozaffari-Kermani, M.: Neon-sidh: efficient implementation of supersingular isogeny diffie-hellman key exchange protocol on arm. In: International Conference on Cryptology and Network Security. pp. 88–103. Springer (2016)
17. Meyer, M., Reith, S., Campos, F.: On hybrid sidh schemes using edwards and montgomery curve arithmetic
18. Montgomery, P.L.: Speeding the pollard and elliptic curve methods of factorization. Mathematics of computation 48(177), 243–264 (1987)
19. Moody, D., Shumow, D.: Analogues of vélu's formulas for isogenies on alternate models of elliptic curves. Mathematics of Computation 85(300), 1929–1951 (2016)
20. Seo, H., Liu, Z., Longa, P., Hu, Z.: Sidh on arm: Faster modular multiplications for faster post-quantum supersingular isogeny key exchange
21. Stolbunov, A.: Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. Adv. in Math. of Comm. 4(2), 215–235 (2010)
22. Vélu, J.: Isogénies entre courbes elliptiques. CR Acad. Sc. Paris. 273, 238–241 (1971)

I Appendix

Algorithm 1 Computing 4-isogeny on Edwards curves

Require: 4-torsion point $P = (Y_4 : Z_4)$ and a curve point $Q = (Y : Z)$ on E_d
Ensure: Image point $Q' = (Y' : Z')$ on $E_{d'}$ and curve coefficients C', D' of the image curve $E_{d'}$ where $d' = D'/C'$

1:	$t_0 \leftarrow Y \cdot Z$	// $t_0 = YZ$
2:	$t_1 \leftarrow Y_4 + Z_4$	// $t_1 = Y_4 + Z_4$
3:	$t_1 \leftarrow t_1^2$	// $t_1 = (Y_4 + Z_4)^2$
4:	$c_1 \leftarrow Y_4 - Z_4$	// $c_1 = Y_4 - Z_4$
5:	$c_1 \leftarrow c_1^2$	// $c_1 = (Y_4 - Z_4)^2$
6:	$t_2 \leftarrow t_1 + c_1$	// $t_2 = 2(Y_4^2 + Z_4^2)$
7:	$C' \leftarrow t_1^2$	// $C' = (Y_4 + Z_4)^4$
8:	$D' \leftarrow c_1^2$	// $D' = (Y_4 - Z_4)^4$
9:	$D' \leftarrow C' - D'$	// $D' = 8Y_4Z_4(Y_4^2 + Z_4^2)$
10:	$t_2 \leftarrow t_0 \cdot t_2$	// $t_2 = 2YZ(Y_4^2 + Z_4^2)$
11:	$t_0 \leftarrow Y \cdot Z_4$	// $t_0 = YZ_4$
12:	$t_1 \leftarrow Z \cdot Y_4$	// $t_1 = ZY_4$
13:	$t_3 \leftarrow t_0 + t_1$	// $t_3 = YZ_4 + ZY_4$
14:	$t_4 \leftarrow t_0 - t_1$	// $t_4 = YZ_4 - ZY_4$
15:	$t_3 \leftarrow t_3^2$	// $t_3 = (YZ_4 + ZY_4)^2$
16:	$t_4 \leftarrow t_4^2$	// $t_4 = (YZ_4 - ZY_4)^2$
17:	$t_6 \leftarrow t_3 + t_4$	// $t_6 = 2(Y^2Z_4^2 + Z^2Y_4^2)$
18:	$t_0 \leftarrow t_2 + t_6$	// $t_0 = 2YZ(Y_4^2 + Z_4^2) + 2(Y^2Z_4^2 + Z^2Y_4^2)$
19:	$t_1 \leftarrow t_2 - t_6$	// $t_1 = 2YZ(Y_4^2 + Z_4^2) - 2(Y^2Z_4^2 + Z^2Y_4^2)$
20:	$t_0 \leftarrow t_0 \cdot t_3$	
21:	$t_1 \leftarrow t_1 \cdot t_4$	
22:	$Y' \leftarrow t_0 + t_1$	
23:	$Z' \leftarrow t_0 - t_1$	
24:	return Y', Z', C', D'	

Algorithm 2 Computing 3-isogeny on Edwards curves

Require: 3-torsion point $P = (Y_3 : Z_3)$ on an Edwards curve E_d
Ensure: The 3-isogenous Montgomery curve with projective curve coefficients C_M/D_M where $C_M = A' + 2C'$ and $D_M = A' - 2C'$.

1: $t_0 \leftarrow Y_3 + Z_3$ // $t_0 = Y_3 + Z_3$
2: $t_0 \leftarrow t_0^2$ // $t_0 = (Y_3 + Z_3)^2$
3: $t_0 \leftarrow t_0 + t_0$ // $t_0 = 2(Y_3 + Z_3)^2$
4: $t_1 \leftarrow Y_3^2$ // $t_1 = Y_3^2$
5: $t_2 \leftarrow Z_3^2$ // $t_2 = Z_3^2$
6: $t_3 \leftarrow t_1 + t_1$ // $t_3 = 2Y_3^2$
7: $t_4 \leftarrow t_2 + t_2$ // $t_4 = 2Z_3^2$
8: $t_6 \leftarrow t_0 - t_3$ // $t_6 = 2Z_3^2 + 4Y_3Z_3$
9: $t_5 \leftarrow t_0 - t_4$ // $t_5 = 2Y_3^2 + 4Y_3Z_3$
10: $D_M \leftarrow t_0 + t_3 - t_2$ // $D_M = (Z_3 + 2Y_3)^2$
11: $C_M \leftarrow t_0 + t_4 - t_1$ // $C_M = (Y_3 + 2Z_3)^2$
12: $D_M \leftarrow D_M \cdot t_6$ // $D_M = 2(Z_3 + 2Y_3)^3 Z_3$
13: $C_M \leftarrow C_M \cdot t_5$ // $C_M = 2(Y_3 + 2Z_3)^3 Y_3$
14: $C_M \leftarrow C_M + C_M$
15: $D_M \leftarrow D_M + D_M$
16: return C_M, D_M

Algorithm 3 Evaluating 3-isogeny on Edwards curves

Require: 3-torsion point $P = (Y_3 : Z_3)$ and a curve point $Q = (Y : Z)$ on E_d
Ensure: Image point $Q' = (X' : Z')$ on the image curve $M_{a,b}$ birationally equivalent to $\phi(E_d)$

1: $t_0 \leftarrow Y_3 \cdot Z$ // $t_0 = YZ_3$
2: $t_1 \leftarrow Z_3 \cdot Y$ // $t_1 = ZY_3$
3: $t_2 \leftarrow t_0 + t_1$ // $t_2 = YZ_3 + ZY_3$
4: $t_3 \leftarrow Y + Z$ // $t_3 = Y + Z$
5: $t_2 \leftarrow t_2^2$ // $t_2 = (YZ_3 + ZY_3)^2$
6: $X' \leftarrow t_2 \cdot t_3$ // $X' = (Y + Z)(ZY_3 + YZ_3)^2$
7: $t_0 \leftarrow t_0 - t_1$ // $t_0 = YZ_3 - ZY_3$
8: $t_0 \leftarrow t_0^2$ // $t_0 = (YZ_3 - ZY_3)^2$
9: $t_1 \leftarrow Z - Y$
10: $Z' \leftarrow t_0 \cdot t_1$ // $Z' = (Z - Y)(ZY_3 - YZ_3)^2$
11: return X', Z'

Algorithm 4 Computing 4-isogeny on Edwards curves

Require: 4-torsion point $P = (Y_4 : Z_4)$ on an Edwards curve E_d

Ensure: The 4-isogenous Montgomery curve with projective curve coefficients C_M/D_M where $C_M = A' + 2C'$ and $D_M = 4C'$ with coefficients c_0 that are used to evaluate the 4-isogeny

1: $C_M \leftarrow Y_4 + Z_4$ // $C_M = Y_4 + Z_4$
2: $C_M \leftarrow C_M^2$ // $C_M = (Y_4 + Z_4)^2$
3: $t_1 \leftarrow Y_4 - Z_4$ // $t_1 = Y_4 - Z_4$
4: $t_1 \leftarrow t_1^2$ // $t_1 = (Y_4 - Z_4)^2$
5: $c_0 \leftarrow C_M + t_1$ // $c_0 = 2(Y_4^2 + Z_4^2)$
6: $C_M \leftarrow C_M + C_M$ // $C_M = 2(Y_4 + Z_4)^2$
7: $t_1 \leftarrow t_1 + t_1$ // $t_1 = 2(Y_4 - Z_4)^2$
8: $C_M \leftarrow C_M^2$ // $C_M = 4(Y_4 + Z_4)^4$
9: $D_M \leftarrow t_1^2$ // $D_M = 4(Y_4 - Z_4)^4$
10: return C_M, D_M, c_0

Algorithm 5 Evaluating 4-isogeny on Edwards curves

Require: 4-torsion point $P = (Y_4 : Z_4)$, a curve point $Q = (Y : Z)$ on E_d and c_0 computed from Algorithm 4.

Ensure: Image point $Q' = (X' : Z')$ on the image curve $M_{a,b}$ birationally equivalent to $\phi(E_d)$

1: $t_0 \leftarrow Y \cdot Z$ // $t_0 = YZ$
2: $t_2 \leftarrow c_0 \cdot t_0$ // $t_2 = 2YZ(Y_4^2 + Z_4^2)$
3: $t_0 \leftarrow Y \cdot Z_4$ // $t_0 = YZ_4$
4: $t_1 \leftarrow Z \cdot Y_4$ // $t_1 = ZY_4$
5: $t_3 \leftarrow t_0 + t_1$ // $t_3 = YZ_4 + ZY_4$
6: $t_4 \leftarrow t_0 - t_1$ // $t_4 = YZ_4 - ZY_4$
7: $t_3 \leftarrow t_3^2$ // $t_3 = (YZ_4 + ZY_4)^2$
8: $t_4 \leftarrow t_4^2$ // $t_4 = (YZ_4 - ZY_4)^2$
9: $t_5 \leftarrow t_3 + t_4$ // $t_5 = 2(Y^2Z_4^2 + Z^2Y_4^2)$
10: $t_0 \leftarrow t_2 + t_5$ // $t_0 = 2YZ(Y_4^2 + Z_4^2) + 2(Y^2Z_4^2 + Z^2Y_4^2)$
11: $t_1 \leftarrow t_5 - t_2$ // $t_1 = 2(Y^2Z_4^2 + Z^2Y_4^2) - 2YZ(Y_4^2 + Z_4^2)$
12: $X' \leftarrow t_0 \cdot t_3$
13: $Z' \leftarrow t_1 \cdot t_4$
14: return X', Z'