Faster Group Operations on Elliptic Curves

Huseyin Hisil, Kenneth Koon-Ho Wong, Gary Carter, Ed Dawson
Information Security Institute, Queensland University of Technology,
{h.hisil, kk.wong, g.carter, e.dawson}@qut.edu.au

Abstract

This paper is on improving implementation techniques of Elliptic Curve Cryptography. We introduce new addition formulae for Jacobi-quartic, Edwards, Hessian forms and new doubling formulae for Jacobi-quartic and Jacobi-intersection forms of elliptic curves. The new formulae speed up the group operations for each of these forms on suitable coordinate systems. To show this, a comparison is made in respect to their performance evaluations with classic point multiplication algorithms using the previous and current operation counts. The most significant outcomes are obtained from the modified Jacobi-quartic coordinates which provide the fastest timings[1] for most point multiplication strategies and the fastest unified[2] addition which costs 7M+3S+1D. The new unified addition formulae can be used to provide a natural way to protect against side channel attacks which are based on simple power analysis (SPA).

Keywords: Efficient elliptic curve arithmetic, unified addition, side channel attack.

1 Introduction

From the advent of elliptic curve cryptosystems, independently by Miller [16] and Koblitz [14] in mid 80's to date, the arithmetic of elliptic curves has drawn wide attention from cryptographic researchers. It is well known that the Weierstrass form provides a general representation for all elliptic curves. In other words, every elliptic curve (over a field K, $\mathrm{char}(K) \neq 2, 3$) can be defined by the set of points (x_i, y_i) satisfying the equation

$$y^2 = x^3 + ax + b, \qquad a, b \in K$$

together with the point at infinity $\mathcal{O}$. These points exhibit a group structure under an explicitly defined additive group law. In other words, two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ can be added to form a third point $R = P + Q = (x_3, y_3)$ on the same curve. The negative of the point P is $(x_1, -y_1)$. The identity element is the point at infinity $\mathcal{O}$. From this we can define scalar multiple S of a point P as

$$S = [k]P = \underbrace{P + P + \ldots + P}_{k \text{ times}}.$$

[1] **M:** The cost of field multiplication, **S:** The cost of field squaring, **D:** The cost of multiplication by a curve constant.

[2] Valid for doubling i.e. addition of a nontrivial point to itself.

Computing k when only P and S are known is believed to be intractable for carefully selected parameters. This forms the basis of the elliptic curve discrete logarithm problem, which is used to provide cryptographic security. One of the main challenges in elliptic curve cryptography is to perform scalar multiplication efficiently under different environment constraints (such as resistance to side channel attacks, bandwidth efficiency, memory limitations). Scalar multiplication is often computed using double-and-add methods and its variants, so the group operations of concern are elliptic curve point addition and doubling.

To obtain faster group operations, other elliptic curve forms have also been considered in the last two decades by researchers. For security considerations, the selected curves should have a small cofactor, usually equal to or less than 4. It is possible to find cryptographically interesting curves which satisfy the security criterion and which can be parameterized by one of the curve models in Section 2. (See [15, 6, 4] for examples). In this context, here is a short outline of previous work on which our paper is built.

- Chudnovsky and Chudnovsky [8] reported the operation counts for inversion-free addition and doubling operations for Weierstrass, Jacobi-quartic, Jacobi-intersection, and Hessian forms. Cohen, Miyaji and Ono [9] provided better operation counts for Weierstrass form. Doche, Icart and Kohel [10] introduced the fastest doubling[3] and tripling in Weierstrass form on two special families of curves.

- In chronological order, Joye and Quisquater [13], Liardet and Smart [15], Brier and Joye [7], Billet and Joye [6] showed ways of doing point multiplication to resist side channel attacks using Hessian, Jacobi-intersection, Weierstrass and Jacobi-quartic forms, respectively.

- Duquesne [11] improved the operation count for the Jacobi-quartic unified addition formulae in [6] by using an alternative coordinate system. Bernstein and Lange [4] provided an extended version of these coordinates with better operation counts for S<M. We extensively use these ideas throughout this paper to obtain faster operation counts for the new formulae.

- Bernstein and Lange [4] showed the importance of Edwards curves for providing fast arithmetic and efficient countermeasure to side channel attacks. Later, Bernstein and Lange [5] introduced the inverted Edwards coordinates which improve timings for Edwards curves and provide the fastest unified addition known to date. They have built a database [2] of explicit formulae that are reported in the literature together with their own optimizations.

Here is a collection of some latest operation counts. The new operation counts that appear in this paper are given in bold. We explain these results in detail in Section 2.

- *Modified Jacobi-quartic coordinates:* (Unified) addition **7M+3S+1D**, readdition **7M+3S+1D**, mixed addition **6M+3S+1D**, doubling **2M+5S+1D** or 3M+4S.

- *Modified Jacobi-intersection coordinates:* (Unified) addition **11M+1S+2D**, readdition **11M+1S+2D**, mixed addition **10M+1S+2D**, doubling **2M+5S+1D** or 3M+4S.

[3] With the improvements of Bernstein, Birkner, Lange and Peters in [1].

- *Standard Edwards coordinates:* (Dedicated) addition **11M**, (unified) addition 10M+1S+1D, readdition **9M+2S**, mixed addition **9M**, doubling 3M+4S.
- *Inverted Edwards coordinates:* (Unified) addition 9M+1S+1D, readdition 9M+1S+1D, mixed addition **9M** or 8M+1S+1D, doubling 3M+4S+1D.
- *Modified Hessian coordinates:* (Unified) addition **6M+6S** or 12M, readdition **6M+6S**, mixed addition **5M+6S**, doubling **3M+6S**.

The paper is organized as follows. We provide new formulae and better operation counts for various elliptic curve forms in Section 2. The exceptional cases are explained in Section 3. We make comparisons of various systems and draw our conclusions in Section 4.

2 Improvements

We omit the operation counts for affine coordinates since these coordinates require field inversions which are relatively expensive compared to the cost of a field multiplication when the field is finite. The derivations of the new addition formulae especially the ones for Edwards and Hessian forms are aided by the use of reduction algorithms for rational expressions on the Maple v.11[4] computer algebra system. Details of the reduction procedure can be found in [17]. We obtain curve definitions and affine versions of various formulae from [2].

2.1 Jacobi-quartic form

The uses of these curves in cryptology are explained by Chudnovsky and Chudnovsky in [8] and Billet and Joye in [6]. We follow the descriptions in [2] for our optimizations. Let K be a field with $\mathrm{char}(K) \neq 2, 3$. An elliptic curve in Jacobi-quartic form is defined by $y^2 = x^4 + 2ax^2 + 1$ where $a \in K$ with $a^2 \neq 1$. The identity element is the point $(0, 1)$. The negative of a point (x, y) is $(-x, y)$. Birational maps between Weierstrass and Jacobi-quartic curves can be found in [6, 2, 3]. The affine unified addition formulae (explained in [6]) are as follows.

$$(x_3, y_3) = \left(\frac{x_1y_2 + y_1x_2}{1 - x_1^2x_2^2}, \frac{(x_1^2x_2^2 + 1)(2ax_1x_2 + y_1y_2) + 2x_1x_2(x_1^2 + x_2^2)}{(1 - x_1^2x_2^2)^2}\right)$$

In this section, we show the derivation of new formulae which produce the same results. In fact, we only need to change the numerator of y_3. If the numerator is designated t then, we have the following.

$$\begin{aligned} t &= (x_1^2x_2^2 + 1)(2ax_1x_2 + y_1y_2) + 2x_1x_2(x_1^2 + x_2^2) \\ &= (x_1^2x_2^2 + 1)(2ax_1x_2 + y_1y_2) + 2x_1x_2(x_1^2 + x_2^2) + x_1^2y_2^2 + 2x_1y_1x_2y_2 + y_1^2x_2^2 - (x_1y_2 + y_1x_2)^2 \end{aligned}$$

Using the curve equation, $y^2 = x^4 + 2ax^2 + 1$, we replace y_1^2 with $x_1^4 + 2ax_1^2 + 1$ and y_2^2 with $x_2^4 + 2ax_2^2 + 1$. Then, we have the following.

$$\begin{aligned} t = {} & (x_1^2x_2^2 + 1)(2ax_1x_2 + y_1y_2) + 2x_1x_2(x_1^2 + x_2^2) + \\ & x_1^2(x_2^4 + 2ax_2^2 + 1) + 2x_1y_1x_2y_2 + (x_1^4 + 2ax_1^2 + 1)x_2^2 - (x_1y_2 + y_1x_2)^2 \end{aligned}$$

[4] http://www.maplesoft.com

We obtain the new formulae for y_3 by organizing the terms.

$$(x_3, y_3) = \left(\frac{x_1y_2 + y_1x_2}{1 - x_1^2x_2^2}, \left(\frac{x_1x_2 + 1}{1 - x_1^2x_2^2}\right)^2 (x_1^2x_2^2 + 2ax_1x_2 + 1 + (x_1 - x_2)^2 + y_1y_2) - x_3^2 - 1\right)$$

The new addition formulae on the Jacobi-quartic coordinates are as follows.

$$\begin{aligned}
X_3 &= X_1Z_1Y_2 + Y_1X_2Z_2\\
Z_3 &= Z_1^2Z_2^2 - X_1^2X_2^2\\
Y_3 &= (X_1X_2 + Z_1Z_2)^2(X_1^2X_2^2 + 2aX_1X_2Z_1Z_2 + Z_1^2Z_2^2 +\\
&\quad (X_1Z_2 - X_2Z_1)^2 + Y_1Y_2) - X_3^2 - Z_3^2
\end{aligned}$$

Note, each point is represented by the triplet $(X_i\colon Y_i\colon Z_i)$ which satisfies the equation $Y^2 = X^4 + 2aX^2Z^2 + Z^4$ and corresponds to the affine point $(X_i/Z_i, Y_i/{Z_i}^2)$ with $Z_i \neq 0$. The identity element is represented by $(0\colon 1\colon 1)$. The negative of $(X_i\colon Y_i\colon Z_i)$ is $(-X_i\colon Y_i\colon Z_i)$. This coordinate system is used in [8, 6]. The new addition formulae are not attractive for the Jacobi-quartic coordinates. On the other hand, they are suitable for a modified version of the Jacobi-quartic coordinates where each point is represented by the sextuplet $(X_i\colon Y_i\colon Z_i\colon X_i^2\colon Z_i^2\colon X_iZ_i)$. The idea behind using such coordinates is explained by Duquesne [11] for the addition formulae in [6]. Regarding the new formulae, $(X_1\colon Y_1\colon Z_1\colon U_1\colon V_1\colon W_1)$ and $(X_2\colon Y_2\colon Z_2\colon U_2\colon V_2\colon W_2)$ with $U_1 = X_1^2$, $V_1 = Z_1^2$, $W_1 = X_1Z_1$, $U_2 = X_2^2$, $V_2 = Z_2^2$, $W_2 = X_2Z_2$ can be added as follows,

$$A \leftarrow U_1U_2, \quad B \leftarrow V_1V_2, \quad C \leftarrow W_1W_2, \quad D \leftarrow Y_1Y_2,$$

$$X_3 \leftarrow (W_1 + Y_1)(W_2 + Y_2) - C - D, \quad Z_3 \leftarrow B - A, \quad U_3 \leftarrow X_3^2, \quad V_3 \leftarrow Z_3^2,$$

$$F \leftarrow A + B + 2C, \quad G \leftarrow (U_1 + V_1)(U_2 + V_2) + kC + D, \quad H \leftarrow U_3 + V_3,$$

$$Y_3 \leftarrow FG - H, \quad W_3 \leftarrow ((X_3 + Z_3)^2 - H)/2$$

where $k = 2(a-1)$. The new unified addition costs **7M+3S+1D** on the modified Jacobi-quartic coordinates. This is faster than the 9M+2S+1D algorithm[5] in [11] and the 8M+3S+1D algorithm in [2]. Assuming that $(X_2\colon Y_2\colon Z_2\colon U_2\colon V_2\colon W_2)$ is cached, the readdition costs **7M+3S+1D**. Then, a **6M+3S+1D** mixed addition can be derived by setting $Z_2 = 1$. We use the names "modified Jacobi-quartic v.1" and "modified Jacobi-quartic v.2b" to refer to this coordinate system in Section 4. Modified Jacobi-quartic v.1 uses the original formulae in [6]. Modified Jacobi-quartic v.2b uses the new formulae. Both systems use 3M+4S doubling algorithm in [12].

To evaluate the new addition formulae, a similar algorithm is beneficial for another version of the modified Jacobi-quartic coordinates using the quintuplet $(X_i\colon Y_i\colon Z_i\colon U_i\colon V_i)$ for representing the points. Then, the unified addition costs **7M+4S+1D** (computing $W_1 = ((X_1 + Z_1)^2 - U_1 - V_1)/2$ and $W_2 = ((X_2 + Z_2)^2 - U_2 - V_2)/2$ on the fly, and not computing W_3). Following this and assuming that $(X_2\colon Y_2\colon Z_2\colon U_2\colon V_2)$ is cached, the readdition costs **7M+3S+1D** (with the extra caching of W_2). Then, a **6M+3S+1D** mixed addition can be derived by setting $Z_2 = 1$. We

[5] Using the Jacobi-quartic curves with $\epsilon = 1$ for the unified addition algorithm in [11].

use the name “modified Jacobi-quartic v.2a” to refer to this system in Section 4. This system also uses 3M+4S doubling algorithm in [12].

The 3M+4S algorithm formulae described in [12] can be easily derived from the new unified addition formulae as follows. First, we input the same points to the new addition formulae and obtain the following,

$$x_3 = \frac{2x_1y_1}{1-x_1^4}, \qquad y_3 = \left(\frac{x_1^2+1}{1-x_1^4}\right)^2 (x_1^4 + 2ax_1^2 + 1 + y_1^2) - x_3^2 - 1.$$

The doubling formulae in [12] can be derived by replacing $x_1^4 + 2ax_1^2 + 1$ with y_1^2.

$$(x_3, y_3) = \left(\frac{2x_1y_1}{1-x_1^4}, 2\left(\frac{y_1(x_1^2+1)}{1-x_1^4}\right)^2 - x_3^2 - 1\right)$$

The doubling formulae on the Jacobi-quartic coordinates are as follows.

$$\begin{aligned} X_3 &= 2X_1Y_1Z_1 \\ Z_3 &= Z_1^4 - X_1^4 \\ Y_3 &= 2(Y_1(X_1^2+Z_1^2))^2 - X_3^2 - Z_3^2 \end{aligned}$$

These formulae are suitable to be used with both versions of the modified Jacobi-quartic coordinates[6]. $(X_1 \colon Y_1 \colon Z_1 \colon U_1 \colon V_1 \colon W_1)$ can be doubled as follows,

$$A \leftarrow U_1 + V_1, \quad X_3 \leftarrow 2Y_1W_1, \quad Z_3 \leftarrow A(V_1 - U_1), \quad U_3 \leftarrow X_3^2,$$
$$V_3 \leftarrow Z_3^2, \quad B \leftarrow U_3 + V_3, \quad W_3 \leftarrow ((X_3+Z_3)^2 - B)/2, \quad Y_3 \leftarrow 2(Y_1A)^2 - B.$$

Doubling costs 3M+4S on both versions of the modified Jacobi-quartic coordinates. Building on similar ideas, it is possible to derive the following doubling formulae.

$$(x_3, y_3) = \left(\frac{2x_1y_1}{1-x_1^4}, 2\left(\frac{y_1^2}{1-x_1^4}\right)^2 - ax_3^2 - 1\right)$$

The new doubling formulae on the Jacobi-quartic coordinates are as follows.

$$\begin{aligned} X_3 &= 2X_1Y_1Z_1 \\ Z_3 &= Z_1^4 - X_1^4 \\ Y_3 &= 2Y_1^4 - aX_3^2 - Z_3^2 \end{aligned}$$

These formulae are again suitable to be used with the modified Jacobi-quartic coordinates. We name two versions of the modified coordinates as “modified Jacobi-quartic v.3a” and “modified Jacobi-quartic v3.b” to emphasize the use of the new doubling formulae. $(X_1 \colon Y_1 \colon Z_1 \colon U_1 \colon V_1 \colon W_1)$ can be doubled as follows,

$$X_3 \leftarrow 2Y_1W_1, \quad Z_3 \leftarrow (V_1 - U_1)(V_1 + U_1), \quad U_3 \leftarrow X_3^2, \quad V_3 \leftarrow Z_3^2,$$
$$W_3 \leftarrow ((X_3+Z_3)^2 - U_3 - V_3)/2, \quad Y_3 \leftarrow 2Y_1^4 - aU_3 - V_3.$$

Doubling costs **2M+5S+1D** on both versions of the modified Jacobi-quartic coordinates.

[6] The adaptation of these formulae to an extended version of the modified Jacobi-quartic coordinates is developed by Bernstein and Lange in EFD [2, 3].

2.2 Jacobi-intersection form

The uses of these curves in cryptology are explained by Chudnovsky and Chudnovsky in [8] and Liardet and Smart in [15]. Let K be a field with $\text{char}(K) \neq 2, 3$. An elliptic curve in Jacobi-intersection form is the set of points which satisfy the equations $s^2 + c^2 = 1$ and $as^2 + d^2 = 1$ simultaneously where $a \in K$ with $a(1-a) \neq 0$. The identity element is the point $(0, 1, 1)$. The negative of a point (s, c, d) is $(-s, c, d)$. Birational maps to Weierstrass curves can be found in [15, 2, 3]. The affine unified addition formulae are given as follows.

$$(s_3, c_3, d_3) = \left(\frac{s_1c_2d_2 + c_1d_1s_2}{c_2^2 + d_1^2s_2^2}, \frac{c_1c_2 - s_1d_1s_2d_2}{c_2^2 + d_1^2s_2^2}, \frac{d_1d_2 - as_1c_1s_2c_2}{c_2^2 + d_1^2s_2^2}\right)$$

The addition on the standard Jacobi-intersection coordinates is given as follows.

$$\begin{aligned} S_3 &= S_1T_1C_2D_2 + C_1D_1S_2T_2 \\ C_3 &= C_1T_1C_2T_2 - S_1D_1S_2D_2 \\ D_3 &= D_1T_1D_2T_2 - aS_1C_1S_2C_2 \\ T_3 &= D_1^2S_2^2 + T_1^2C_2^2 \end{aligned}$$

Note, each point is represented by the quadruplet $(S_i : C_i : D_i : T_i)$ which satisfies the equations $S^2 + C^2 = T^2$ and $aS^2 + D^2 = T^2$ simultaneously and corresponds to the affine point $(S_i/T_i, C_i/T_i, D_i/T_i)$ with $T_i \neq 0$. The identity element is represented by $(0 : 1 : 1 : 1)$. The negative of $(S_i : C_i : D_i : T_i)$ is $(-S_i : C_i : D_i : T_i)$. We modify the standard Jacobi-intersection coordinates where each point is represented by the sextuplet, $(S_i : C_i : D_i : T_i : S_iC_i : D_iT_i)$. Then, $(S_1 : C_1 : D_1 : T_1 : U_1 : V_1)$ and $(S_2 : C_2 : D_2 : T_2 : U_2 : V_2)$ with $U_1 = S_1C_1$, $V_1 = D_1T_1$, $U_2 = S_2C_2$, $V_2 = D_2T_2$ can be added as follows,

$$E \leftarrow S_1D_2, \quad F \leftarrow C_1T_2, \quad G \leftarrow D_1S_2, \quad H \leftarrow T_1C_2, \quad J \leftarrow U_1V_2, \quad K \leftarrow V_1U_2,$$

$$S_3 \leftarrow (H + F)(E + G) - J - K, \quad C_3 \leftarrow (H + E)(F - G) - J + K,$$

$$D_3 \leftarrow (V_1 - aU_1)(U_2 + V_2) + aJ - K, \quad T_3 \leftarrow (H + G)^2 - 2K, \quad U_3 \leftarrow S_3C_3, \quad V_3 \leftarrow D_3T_3.$$

The unified addition costs **11M+1S+2D** on the modified Jacobi-intersection coordinates. This is faster than the 13M+2S+1D algorithm in [15] for the standard Jacobi-intersection coordinates. Assuming that $(S_2 : C_2 : D_2 : T_2 : U_2 : V_2)$ is cached, the readdition costs **11M+1S+2D**. Then, a **10M+1S+2D** mixed addition can be derived by setting $T_2 = 1$. We use the name "modified Jacobi-intersection" to refer to these results in Section 4.

A similar algorithm can be used for the standard Jacobi-intersection coordinates. Then, the unified addition costs **13M+1S+2D** (computing $U_1 = S_1C_1$, $V_1 = D_1T_1$, $U_2 = S_2C_2$, $V_2 = D_2T_2$ on the fly, and not computing U_3 and V_3). This is also faster than the 13M+2S+1D algorithm in [15] when D<S. Following this and assuming that $(S_2 : C_2 : D_2 : T_2)$ is cached, the readdition costs **11M+1S+2D** (with the extra caching of U_2 and V_2). Then, a **10M+1S+2D** mixed addition can be derived by setting $T_2 = 1$. We use the name "Jacobi-intersection v.2" to refer to these results in Section 4.

Compatible doubling formulae for the modified Jacobi-intersection coordinates can be derived from the unified addition formulae. First, we input the same points to the original addition formulae and obtain the following.

$$(s_3, c_3, d_3) = \left(\frac{2s_1c_1d_1}{c_1^2 + s_1^2d_1^2}, \frac{c_1^2 - s_1^2d_1^2}{c_1^2 + s_1^2d_1^2}, \frac{d_1^2 - as_1^2c_1^2}{c_1^2 + s_1^2d_1^2}\right)$$

Using the defining equations, $s^2 + c^2 = 1$ and $as^2 + d^2 = 1$, we replace c_1^2 with $c_1^2(as_1^2 + d_1^2)$ (only for the denominators) and $s_1^2d_1^2$ with $(1 - c_1^2)d_1^2$.

$$\begin{aligned}
s_3 &= (2s_1c_1d_1)/(c_1^2(as_1^2 + d_1^2) + (1 - c_1^2)d_1^2)\\
c_3 &= (c_1^2(as_1^2 + d_1^2) - (1 - c_1^2)d_1^2)/(c_1^2(as_1^2 + d_1^2) + (1 - c_1^2)d_1^2)\\
d_3 &= (d_1^2 - as_1^2c_1^2)/(c_1^2(as_1^2 + d_1^2) + (1 - c_1^2)d_1^2)
\end{aligned}$$

This gives an intermediate formula for c_3.

$$(s_3, c_3, d_3) = \left(\frac{2s_1c_1d_1}{d_1^2 + as_1^2c_1^2}, \frac{as_1^2c_1^2 + 2c_1^2d_1^2 - d_1^2}{d_1^2 + as_1^2c_1^2}, \frac{d_1^2 - as_1^2c_1^2}{d_1^2 + as_1^2c_1^2}\right)$$

Finally, we replace $2c_1^2d_1^2$ with $2c_1^2(s_1^2 + c_1^2 - as_1^2)$ in c_3.

$$\begin{aligned}
s_3 &= (2s_1c_1d_1)/(as_1^2c_1^2 + d_1^2)\\
c_3 &= (as_1^2c_1^2 + 2c_1^2(s_1^2 + c_1^2 - as_1^2) - d_1^2)/(as_1^2c_1^2 + d_1^2)\\
d_3 &= (d_1^2 - as_1^2c_1^2)/(as_1^2c_1^2 + d_1^2)
\end{aligned}$$

The new doubling formulae are as follows.

$$(s_3, c_3, d_3) = \left(\frac{2s_1c_1d_1}{d_1^2 + as_1^2c_1^2}, \frac{-d_1^2 - as_1^2c_1^2 + 2(s_1^2c_1^2 + c_1^4)}{d_1^2 + as_1^2c_1^2}, \frac{d_1^2 - as_1^2c_1^2}{d_1^2 + as_1^2c_1^2}\right)$$

The new doubling formulae on the standard Jacobi-intersection coordinates are as follows.

$$\begin{aligned}
S_3 &= 2S_1C_1D_1T_1\\
C_3 &= -D_1^2T_1^2 - aS_1^2C_1^2 + 2(S_1^2C_1^2 + C_1^4)\\
D_3 &= D_1^2T_1^2 - aS_1^2C_1^2\\
T_3 &= D_1^2T_1^2 + aS_1^2C_1^2
\end{aligned}$$

$(S_1\colon C_1\colon D_1\colon T_1\colon U_1\colon V_1)$ can be doubled as follows,

$$E \leftarrow V_1^2, \quad F \leftarrow U_1^2, \quad G \leftarrow aF, \quad T_3 \leftarrow E + G, \quad D_3 \leftarrow E - G,$$

$$C_3 \leftarrow 2(F + C_1^4) - T_3, \quad S_3 \leftarrow (U_1 + V_1)^2 - E - F, \quad U_3 \leftarrow S_3C_3, \quad V_3 \leftarrow D_3T_3.$$

It is easy to see that point doubling costs **2M+5S+1D** both on standard and the modified Jacobi-intersection coordinates.

2.3 Edwards form

The uses of these curves in cryptology are introduced by Bernstein and Lange in [4, 1, 5]. Let K be a field with $\mathrm{char}(K) \neq 2$. An elliptic curve in Edwards form is defined by $x^2 + y^2 = c^2(1 + dx^2y^2)$ where $c, d \in K$ with $cd(1 - c^4d) \neq 0$. The identity element is the point $(0, c)$. The negative of a point (x, y) is $(-x, y)$. Birational maps between Weierstrass and Edwards curves can be found in [4]. The affine unified addition formulae are given as follows.

$$(x_3, y_3) = \left(\frac{x_1y_2 + y_1x_2}{c(1 + dx_1y_1x_2y_2)}, \frac{y_1y_2 - x_1x_2}{c(1 - dx_1y_1x_2y_2)}\right)$$

We first describe how new addition formulae for Edwards curves can be derived from the original addition formulae in [4]. We start with the Edwards curve equation $x^2 + y^2 = c^2(1 + dx^2y^2)$. Suppose we wish to add (x_1, y_1) and (x_2, y_2). Consider the relations obtained by the curve equation at these two points, i.e., $x_1^2 + y_1^2 - c^2(1 + dx_1^2y_1^2) = 0$, $x_2^2 + y_2^2 - c^2(1 + dx_2^2y_2^2) = 0$. From this, we can express c and d in terms of x_1, x_2, y_1, y_2 as follows,

$$c^2 = \frac{x_1^2x_2^2y_1^2 - x_1^2x_2^2y_2^2 + x_1^2y_1^2y_2^2 - x_2^2y_1^2y_2^2}{x_1^2y_1^2 - x_2^2y_2^2}, \qquad d = \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2}{x_1^2x_2^2y_1^2 - x_1^2x_2^2y_2^2 + x_1^2y_1^2y_2^2 - x_2^2y_1^2y_2^2}.$$

Substitutions can be made to the original addition formulae to obtain

$$x_3 = \frac{x_1y_2 + y_1x_2}{\frac{1}{c}\frac{x_1^2x_2^2y_1^2 - x_1^2x_2^2y_2^2 + x_1^2y_1^2y_2^2 - x_2^2y_1^2y_2^2}{x_1^2y_1^2 - x_2^2y_2^2}\left(1 + \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2}{x_1^2x_2^2y_1^2 - x_1^2x_2^2y_2^2 + x_1^2y_1^2y_2^2 - x_2^2y_1^2y_2^2}x_1y_1x_2y_2\right)},$$

$$y_3 = \frac{y_1y_2 - x_1x_2}{\frac{1}{c}\frac{x_1^2x_2^2y_1^2 - x_1^2x_2^2y_2^2 + x_1^2y_1^2y_2^2 - x_2^2y_1^2y_2^2}{x_1^2y_1^2 - x_2^2y_2^2}\left(1 - \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2}{x_1^2x_2^2y_1^2 - x_1^2x_2^2y_2^2 + x_1^2y_1^2y_2^2 - x_2^2y_1^2y_2^2}x_1y_1x_2y_2\right)}.$$

After simplifications, we derive the new addition formulae

$$(x_3, y_3) = \left(\frac{c(x_1y_1 + x_2y_2)}{x_1x_2 + y_1y_2}, \frac{c(x_1y_1 - x_2y_2)}{x_1y_2 - y_1x_2}\right).$$

Note, the formula for computing y_3 is not defined for $(x_1, y_1) = (x_2, y_2)$ and hence not unified. For this reason, we call the new formulae *dedicated* addition for Edwards curves. These new formulae show an interesting fact that dedicated addition on the Edwards curves does not depend on the curve parameter d. Therefore, arbitrary selections of d do not cause any efficiency loss.

To prevent field inversions that appear in the affine formulae, we represent each point in standard Edwards coordinates [4]. Each point is represented by the triplet $(X_i\colon Y_i\colon Z_i)$ which satisfies the projective curve $(X^2 + Y^2)Z^2 = c^2(Z^4 + dX^2Y^2)$ and corresponds to the affine point $(X_i/Z_i, Y_i/Z_i)$ with $Z_i \neq 0$. The identity element is represented by $(0\colon 1\colon 1)$. The negative of $(X_i\colon Y_i\colon Z_i)$ is $(-X_i\colon Y_i\colon Z_i)$. The new addition formulae on the standard Edwards coordinates are as follows,

$$\begin{aligned}
X_3 &= Z_1Z_2(X_1Y_2 - Y_1X_2)(X_1Y_1Z_2^2 + Z_1^2X_2Y_2)\\
Y_3 &= Z_1Z_2(X_1X_2 + Y_1Y_2)(X_1Y_1Z_2^2 - Z_1^2X_2Y_2)\\
Z_3 &= kZ_1^2Z_2^2(X_1X_2 + Y_1Y_2)(X_1Y_2 - Y_1X_2)
\end{aligned}$$

where $k = 1/c$. $(X_1 \colon Y_1 \colon Z_1)$ and $(X_2 \colon Y_2 \colon Z_2)$ can be added as follows,

$$A \leftarrow X_1Z_2, \quad B \leftarrow Y_1Z_2, \quad C \leftarrow Z_1X_2, \quad D \leftarrow Z_1Y_2, \quad E \leftarrow AB, \quad F \leftarrow CD,$$

$$G \leftarrow E+F, \quad H \leftarrow E-F, \quad J \leftarrow (A-C)(B+D)-H, \quad K \leftarrow (A+D)(B+C)-G,$$

$$X_3 \leftarrow GJ, \quad Y_3 \leftarrow HK, \quad Z_3 \leftarrow k\,JK.$$

We also investigate the operation counts for the inverted Edwards coordinates in [5]. The new addition formulae on the inverted Edwards coordinates are as follows.

$$\begin{aligned} X_3 &= Z_1Z_2(X_1X_2+Y_1Y_2)(X_1Y_1Z_2^2 - Z_1^2X_2Y_2) \\ Y_3 &= Z_1Z_2(X_1Y_2-Y_1X_2)(X_1Y_1Z_2^2 + Z_1^2X_2Y_2) \\ Z_3 &= c(X_1Y_1Z_2^2 + Z_1^2X_2Y_2)(X_1Y_1Z_2^2 - Z_1^2X_2Y_2) \end{aligned}$$

Each triplet $(X_i \colon Y_i \colon Z_i)$ satisfies the projective curve $(X^2+Y^2)Z^2 = c^2(X^2Y^2+dZ^4)$ and corresponds to the affine point $(Z_i/X_i, Z_i/Y_i)$ with $X_iY_i \neq 0$. $(X_1 \colon Y_1 \colon Z_1)$ and $(X_2 \colon Y_2 \colon Z_2)$ can be added as follows,

$$A \leftarrow X_1Z_2, \quad B \leftarrow Y_1Z_2, \quad C \leftarrow Z_1X_2, \quad D \leftarrow Z_1Y_2,$$

$$E \leftarrow AB, \quad F \leftarrow CD, \quad G \leftarrow E+F, \quad H \leftarrow E-F,$$

$$X_3 \leftarrow ((A+D)(B+C)-G)H, \quad Y_3 \leftarrow ((A-C)(B+D)-H)G, \quad Z_3 \leftarrow c\,GH.$$

We assume $c = 1$ (see [4, Section 4]). Then, the dedicated addition costs **11M** for both coordinate systems. A **9M** mixed addition can be derived by setting $Z_2 = 1$ again for both coordinate systems. It is more convenient to divide each coordinate of the new formulae by Z_1Z_2 (assuming $Z_1Z_2 \neq 0$) for the readdition on the standard Edwards coordinates. Then, the readdition of $(X_2 \colon Y_2 \colon Z_2)$ can be performed with the cached values $R_1 = X_2Y_2$ and $R_2 = Z_2^2$ as follows,

$$A \leftarrow X_1Y_1, \quad B \leftarrow Z_1^2, \quad C \leftarrow R_2A, \quad D \leftarrow R_1B, \quad E \leftarrow (X_1-X_2)(Y_1+Y_2)-A+R_1,$$

$$F \leftarrow (X_1+Y_2)(X_2+Y_1)-A-R_1, \quad G \leftarrow ((Z_1+Z_2)^2-B-R_2)/2,$$

$$X_3 \leftarrow E(C+D), \quad Y_3 \leftarrow F(C-D), \quad Z_3 \leftarrow k\,EFG.$$

The readdition costs **9M+2S** on the standard Edwards coordinates. (See "Edwards v.2" in Table 1 and Table 2 in the appendix). In fact, the readdition algorithm shows that a modified version of the standard Edwards coordinates in which the points are represented by the quintuplet $(X_i \colon Y_i \colon Z_i \colon Z_i^2 \colon X_iY_i)$ permits an inversion-free addition in 9M+2S using the same algorithm. This is faster than the 11M algorithm that we have just described. However, the 3M+4S doubling formulae/algorithm in [4] costs 5M+2S on this coordinate system and also the mixed addition costs 8M+2S which is slower than the 9M mixed addition given above. Therefore, we do not consider this case. The new addition and its associated readdition on the inverted Edwards coordinates are not attractive as they are for the standard Edwards coordinates. On the other hand, the mixed addition can be used in some cases. (See "Inverted Edwards v.2" in Table 1 and Table 2 in the appendix).

2.4 Hessian form

The uses of these curves in cryptology are explained by Chudnovsky and Chudnovsky in [8], Joye and Quisquater in [13], and Smart in [19]. Let K be a field with $\mathrm{char}(K) \neq 2, 3$. An elliptic curve in Hessian form is defined by $x^3 + y^3 + 1 = 3dxy$ where $d \in K$ with $d^3 \neq 1$. The identity element is the point at infinity. The negative of a point (x, y) is (y, x). Birational maps between Weierstrass and Hessian curves can be found in [19, 13, 2, 3]. The addition formulae attributed to Sylvester in [8, pp.424-425] are as follows.

$$(x_3, y_3) = \left(\frac{y_1^2x_2 - y_2^2x_1}{x_2y_2 - x_1y_1}, \frac{x_1^2y_2 - x_2^2y_1}{x_2y_2 - x_1y_1}\right)$$

The addition formulae on the standard Hessian coordinates are defined as follows (with each coordinate multiplied by 2).

$$\begin{aligned} X_3 &= 2Y_1^2X_2Z_2 - 2X_1Z_1Y_2^2 \\ Y_3 &= 2X_1^2Y_2Z_2 - 2Y_1Z_1X_2^2 \\ Z_3 &= 2Z_1^2X_2Y_2 - 2X_1Y_1Z_2^2 \end{aligned}$$

Note, each point is represented by the triplet $(X_i\colon Y_i\colon Z_i)$ which satisfies the projective curve $X^3 + Y^3 + Z^3 = 3dXYZ$ and corresponds to the affine point $(X_i/Z_i, Y_i/Z_i)$ with $Z_i \neq 0$. The identity element is represented by $(1\colon -1\colon 0)$. The negative of $(X_i\colon Y_i\colon Z_i)$ is $(Y_i\colon X_i\colon Z_i)$. To gain better operation counts, we modify the standard Hessian coordinates with a more redundant representation of points using the nonuplet, $(X_i\colon Y_i\colon Z_i\colon X_i^2\colon Y_i^2\colon Z_i^2\colon 2X_iY_i\colon 2X_iZ_i\colon 2Y_iZ_i)$. $(X_1\colon Y_1\colon Z_1\colon R_1\colon S_1\colon T_1\colon U_1\colon V_1\colon W_1)$ and $(X_2\colon Y_2\colon Z_2\colon R_2\colon S_2\colon T_2\colon U_2\colon V_2\colon W_2)$ with $R_1 = X_1^2$, $S_1 = Y_1^2$, $T_1 = Z_1^2$, $U_1 = 2X_1Y_1$, $V_1 = 2X_1Z_1$, $W_1 = 2Y_1Z_1, R_2 = X_2^2$, $S_2 = Y_2^2$, $T_2 = Z_2^2$, $U_2 = 2X_2Y_2$, $V_2 = 2X_2Z_2$, $W_2 = 2Y_2Z_2$ can be added as follows,

$$X_3 \leftarrow S_1V_2 - V_1S_2, \quad Y_3 \leftarrow R_1W_2 - W_1R_2, \quad Z_3 \leftarrow T_1U_2 - U_1T_2,$$

$$R_3 \leftarrow X_3^2, \quad S_3 \leftarrow Y_3^2, \quad T_3 \leftarrow Z_3^2,$$

$$U_3 \leftarrow (X_3 + Y_3)^2 - R_3 - S_3, \quad V_3 \leftarrow (X_3 + Z_3)^2 - R_3 - T_3, \quad W_3 \leftarrow (Y_3 + Z_3)^2 - S_3 - T_3.$$

The unified addition[7] costs **6M+6S** on the modified Hessian coordinates. If S<M, this strategy improves on the 12M figure reported in [8] at the cost of more space. Assuming that $(X_2\colon Y_2\colon Z_2\colon R_2\colon S_2\colon T_2\colon U_2\colon V_2\colon W_2)$ is cached, the readdition costs **6M+6S**. Then, a **5M+6S** mixed addition can be derived by setting $Z_2 = 1$. We use the name "modified Hessian" to refer to these results in Section 4.

A similar algorithm can be used for the standard Hessian coordinates for the readdition and the mixed addition. Assuming that $(X_2\colon Y_2\colon Z_2)$ is cached, the readdition costs **6M+6S** (with the extra caching of $R_2, S_2, T_2, U_2, V_2, W_2$). Then, a **5M+6S** mixed addition can be derived by setting $Z_2 = 1$. (Also see Hisil, Carter and Dawson [12, pp.146–147]). We use the name "Hessian v.2" to refer to these results in Section 4.

[7] Point doubling can be performed as $(Z_1\colon X_1\colon Y_1\colon T_1\colon R_1\colon S_1\colon V_1\colon W_1\colon U_1)$ + $(Y_1\colon Z_1\colon X_1\colon S_1\colon T_1\colon R_1\colon W_1\colon U_1\colon V_1)$ using the addition formulae on the modified Hessian coordinates. This strategy originates from Joye and Quisquater [13, p.6].

For speed oriented implementations, Sylvester's doubling formulae are as follows.

$$(x_3, y_3) = \left(\frac{y_1(1-x_1^3)}{x_1^3 - y_1^3}, -\frac{x_1(1-y_1^3)}{x_1^3 - y_1^3}\right)$$

When working with the modified coordinates, there exists a doubling strategy which requires no additional effort for generating the new coordinates. Sylvester's doubling formulae can be expressed on the standard Hessian coordinates (with each coordinate multiplied by 4).

$$\begin{aligned} X_3 &= (2X_1Y_1 - 2Y_1Z_1)(2X_1Z_1 + 2(X_1^2 + Z_1^2)) \\ Y_3 &= (2X_1Z_1 - 2X_1Y_1)(2Y_1Z_1 + 2(Y_1^2 + Z_1^2)) \\ Z_3 &= (2Y_1Z_1 - 2X_1Z_1)(2X_1Y_1 + 2(X_1^2 + Y_1^2)) \end{aligned}$$

Then, $(X_1{:}Y_1{:}Z_1{:}R_1{:}S_1{:}T_1{:}U_1{:}V_1{:}W_1)$ can be doubled as follows,

$$X_3 \leftarrow (U_1 - W_1)(V_1 + 2(R_1 + T_1)), \quad Y_3 \leftarrow (V_1 - U_1)(W_1 + 2(S_1 + T_1)),$$

$$Z_3 \leftarrow (W_1 - V_1)(U_1 + 2(R_1 + S_1)), \quad R_3 \leftarrow X_3^2, \quad S_3 \leftarrow Y_3^2, \quad T_3 \leftarrow Z_3^2,$$

$$U_3 \leftarrow (X_3 + Y_3)^2 - R_3 - S_3, \quad V_3 \leftarrow (X_3 + Z_3)^2 - R_3 - T_3, \quad W_3 \leftarrow (Y_3 + Z_3)^2 - S_3 - T_3.$$

Point doubling costs **3M+6S** on both standard and the modified Hessian coordinates. (See [12] for a 7M+1S algorithm on the standard coordinates).

We comment that it is possible to derive unified addition formulae which do not require any permutations of the coordinates to perform doubling. Assuming[8] $x_1x_2 \neq y_1y_2$, we multiply the numerator and the denominator of Sylvester's addition formulae for x_3 by $(x_1^3x_2^3 - y_1^3y_2^3)$ to obtain

$$x_3 = \frac{(x_1^3x_2^3 - y_1^3y_2^3)(y_1^2x_2 - y_2^2x_1)}{(x_1^3x_2^3 - y_1^3y_2^3)(x_2y_2 - x_1y_1)}.$$

This can be rearranged as follows

$$x_3 = \frac{x_1y_1^2(y_2^3 + x_2^3)(y_2^2y_1 + x_1^2x_2) - x_2y_2^2(y_1^3 + x_1^3)(y_1^2y_2 + x_2^2x_1)}{(x_1^3x_2^3 - y_1^3y_2^3)(x_2y_2 - x_1y_1)}.$$

Using the curve equation $x^2 + y^2 + 1 = 3dxy$, the above expression can be rewritten as

$$x_3 = \frac{x_1y_1^2(3dx_2y_2 - 1)(y_2^2y_1 + x_1^2x_2) - x_2y_2^2(3dx_1y_1 - 1)(y_1^2y_2 + x_2^2x_1)}{(x_1^3x_2^3 - y_1^3y_2^3)(x_2y_2 - x_1y_1)}.$$

The numerator can be factorized and cancels with $(x_2y_2 - x_1y_1)$ in the denominator, giving the new addition formulae. The corresponding formula for y_3 can be similarly derived from symmetry.

$$(x_3, y_3) = \left(\frac{x_1x_2(x_1y_1 + x_2y_2 - 3dx_1x_2y_1y_2) + y_1^2y_2^2}{x_1^3x_2^3 - y_1^3y_2^3}, -\frac{y_1y_2(x_1y_1 + x_2y_2 - 3dx_1x_2y_1y_2) + x_1^2x_2^2}{x_1^3x_2^3 - y_1^3y_2^3}\right)$$

[8] This is equivalent to saying $(x_1, y_1) \neq -(x_2, y_2)$. The contrary case should be handled separately as explained in Section 3.

The new addition formulae on the standard Hessian coordinates are defined as follows.

$$\begin{aligned} X_3 &= X_1X_2(X_1Y_1Z_2^2 + X_2Y_2Z_1^2 - 3dX_1Y_1X_2Y_2) + Y_1^2Z_1Y_2^2Z_2 \\ Y_3 &= -Y_1Y_2(X_1Y_1Z_2^2 + X_2Y_2Z_1^2 - 3dX_1Y_1X_2Y_2) - X_1^2Z_1X_2^2Z_2 \\ Z_3 &= X_1^3X_2^3 - Y_1^3Y_2^3 \end{aligned}$$

We again use a modified version of the standard coordinates. Two points $(X_1{:}\,Y_1{:}\,Z_1{:}\,V_1{:}\,W_1)$ and $(X_2{:}\,Y_2{:}\,Z_2{:}\,V_2{:}\,W_2)$ with $V_1 = X_1Y_1$, $W_1 = Z_1^2$, $V_2 = X_2Y_2$, $W_2 = Z_2^2$ can be added as follows,

$$A \leftarrow X_1X_2, \quad B \leftarrow Y_1Y_2, \quad C \leftarrow ((Z_1+Z_2)^2 - W_1 - W_2)/2, \quad D \leftarrow A^2, \quad E \leftarrow B^2,$$

$$F \leftarrow D+E, \quad G \leftarrow ((A+B)^2 - F)/2, \quad H \leftarrow (V_1+W_1)(V_2+W_2) - (3d+1)G - C^2,$$

$$X_3 \leftarrow AH + EC, \quad Y_3 \leftarrow -BH - DC, \quad Z_3 \leftarrow (A-B)(G+F), \quad V_3 \leftarrow X_3Y_3, \quad W_3 \leftarrow Z_3^2.$$

This strategy costs **9M+6S+1D** which is faster than the unified addition in Weierstrass form in [7, 2]. However, it is slower than all other unified additions considered in this paper. In addition, doubling, readdition and mixed addition formulae that can be derived from these formulae are not attractive. Therefore, we omit these formulae from further comparison with other systems.

3 Handling Exceptional Cases

An elliptic curve which can be written in one of these forms always has points of small order (other than the identity) and the arithmetic of these points can cause division by zero exceptions depending on the formulae and the coordinate system in use. Cryptographic applications use a large prime order subgroup in which these points (except the identity element, $\mathcal{O}$) do not exist. At this stage, an implementer only needs to be careful about the identity element. When the points P and Q are to be added, a general strategy to handle the exceptional cases is as follows. Let R be the sum of P and Q. Then, $R = Q$ if $P = \mathcal{O}$; $R = P$ if $Q = \mathcal{O}$; $R = \mathcal{O}$ if $P = -Q$. For all other inputs, the sum can be computed with the relevant formulae given in Section 2. In this context, there are some formulae and coordinate system combinations which do not cause exceptions. These are Edwards v.1a, v.1b, v.2, Jacobi-quartic v.1, v.2, Jacobi-intersection v.1, v.2, modified Jacobi-quartic v.1, v.2a, v.2b, v.3a, v.3b and modified Jacobi-intersection. The ones which need exception handling are inverted Edwards (as explained in [5]) v.1, v.2, Hessian v.1, v.2, and modified Hessian. The descriptions and the references for the systems which are not defined so far can be found in the appendix.

4 Comparison and Conclusion

There are several point multiplication algorithms which can benefit from the optimizations in this paper. We only make comparisons for the popular point multiplication strategies between known elliptic curve forms/families. We exclude the cost of the final inversion to affine coordinates for point multiplication.

Resource limited environments. In memory limited environments (such as smartcards), there is not enough space for storing precomputation tables. For these environments, point multiplication with the *"Non-adjacent form without precomputation"* algorithm is a convenient selection. This algorithm requires 1 doubling, 1/3 mixed addition per bit. The cost estimates are depicted in Table 1. *For example, the best timings for 256-bit scalar multiplication (S/M=0.8, D/M≈0) are obtained by the modified Jacobi-quartic v.3a and v.3b which costs 2246M. The previous best was set by the inverted Edwards v.1 [5] which requires 2331M for the same example.*

Speed implementations. This is the most difficult case in which to state a fair comparison because the optimum speeds are somewhat dependent on the choice of the scalar multiplication algorithm. For instance, Doche/Icart/Kohel-3 curves [10] have very fast tripling formulae which can highly benefit from double base number system based point multiplication. For double-and-add type point multiplication algorithms, one might expect to gain the best timing with the system which has the fastest doubling operation since point doubling is the dominating operation. However, the readdition and the mixed addition costs also play important roles in the overall timings. We can *roughly* state that the fast systems (S/M=0.8, D/M≈0) are the modified Jacobi-quartics v.1, v.2a, v.2b, v.3a, v.3b, inverted Edwards v.1a, v.1b, Edwards v.2, and modified Jacobi-intersection. At least, these systems can be faster than the Montgomery ladder [18] which has the fixed cost of 4M+5S+1D per key bit. To make the comparison easier, we fix the algorithm to the *"signed 4-bit sliding windows"* scalar multiplication algorithm analyzed in [4]. The algorithm requires 0.98 doublings, 0.17 readditions, 0.025 mixed additions and 0.0035 additions per bit (for 256-bit scalars). We use this analysis to report current rankings between different systems in Table 2. With our improvements, the modified Jacobi-quartic v.3a, v.3b provides the fastest timings for almost all S/M and D/M values. *For example, 256-bit scalar multiplication (S/M=0.8, D/M≈0) costs around 1970M for the modified Jacobi-quartic v.3a, v.3b. The previous best was set by the inverted Edwards v.1 which require 2040M for the same example.*

Side channel attacks. Targeting the embedded implementations, we take the *"Non-adjacent form without precomputation with SPA protection"* scalar multiplication algorithm into our consideration. This is almost the same as using the *"Non-adjacent form without precomputation"* algorithm with the difference that unified addition formulae is used for both point doubling and point addition. This strategy hides the side channel information from the attacker who needs more samplings and statistical tools for a successful attack. This algorithm invokes 4/3 unified additions per bit. The modified coordinates for Hessian and Jacobi-intersection forms are only useful here. The **7M+3S+1D** unified addition of the modified Jacobi-quartic v.2b, v.3b is the fastest among all other unified additions. The cost estimates for various systems are depicted in Table 3. *For example, 256-bit scalar multiplication (S/M=0.8, D/M≈0) costs 3208M for the modified Jacobi-quartic v.2b, v.3b. The same operation requires 3345M for the inverted Edwards v.1, v.2 (previous fastest) and 5256M for the Weierstrass form (a=-3) using the standard projective coordinates. Modified Jacobi-quartic v.2b and v3.b are 64% faster than the Weierstrass form in this context. The speedup varies between 45% and 67% depending on the S/M and D/M values present.* We should note that the Montgomery ladder [18] is still the fastest for defeating the SPA attacks.

Future directions. Many of these operation counts may be subject to further development.

For instance, the lazy reduction possibilities and the total memory requirements of the new algorithms have not been determined yet. In addition, there are curve models which are not studied in this paper, which may provide improvements. Furthermore, similar ideas might also apply to the low characteristic and/or higher genus curves. Therefore, there is still much room for research on this topic.

References

[1] Daniel J. Bernstein, Peter Birkner, Tanja Lange, and Christiane Peters. Optimizing double-base elliptic-curve single-scalar multiplication. In *INDOCRYPT 2007*, volume 4859 of *LNCS*, pages 167–182. Springer, 2007.

[2] Daniel J. Bernstein and Tanja Lange. Explicit-formulas database. `http://www.hyperelliptic.org/EFD`.

[3] Daniel J. Bernstein and Tanja Lange. Analysis and optimization of elliptic-curve single-scalar multiplication. Cryptology ePrint Archive, Report 2007/455, 2007. `http://eprint.iacr.org/`.

[4] Daniel J. Bernstein and Tanja Lange. Faster addition and doubling on elliptic curves. In *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 29–50. Springer, 2007.

[5] Daniel J. Bernstein and Tanja Lange. Inverted Edwards coordinates. In *AAECC-17*, volume 4851 of *LNCS*, pages 20–27. Springer, 2007.

[6] Olivier Billet and Marc Joye. The Jacobi model of an elliptic curve and side-channel analysis. In *AAECC-15*, volume 2643 of *LNCS*, pages 34–42. Springer, 2003.

[7] Eric Brier and Marc Joye. Weierstraß elliptic curves and side-channel attacks. In *PKC 2002*, pages 335–345. Springer, 2002.

[8] David V. Chudnovsky and Gregory V. Chudnovsky. Sequences of numbers generated by addition in formal groups and new primality and factorization tests. *Advances in Applied Mathematics*, 7(4):385–434, 1986.

[9] Henri Cohen, Atsuko Miyaji, and Takatoshi Ono. Efficient elliptic curve exponentiation using mixed coordinates. In *ASIACRYPT'98*, pages 51–65. Springer, 1998.

[10] Christophe Doche, Thomas Icart, and David R. Kohel. Efficient scalar multiplication by isogeny decompositions. In *PKC 2006*, volume 3958 of *LNCS*, pages 191–206. Springer, 2006.

[11] Sylvain Duquesne. Improving the arithmetic of elliptic curves in the Jacobi model. *Information Processing Letters*, 104(3):101–105, 2007.

[12] Huseyin Hisil, Gary Carter, and Ed Dawson. New formulae for efficient elliptic curve arithmetic. In *INDOCRYPT 2007*, volume 4859 of *LNCS*, pages 138–151. Springer, 2007.

[13] Marc Joye and Jean Jacques Quisquater. Hessian elliptic curves and side-channel attacks. In *CHES 2001*, volume 2162 of *LNCS*, pages 402–410. Springer, 2001.

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

[15] Pierre Yvan Liardet and Nigel P. Smart. Preventing SPA/DPA in ECC systems using the Jacobi form. In *CHES 2001*, volume 2162 of *LNCS*, pages 391–401. Springer, 2001.

[16] Victor S. Miller. Use of elliptic curves in cryptography. In *CRYPTO'85*, volume 218 of *LNCS*, pages 417–426. Springer, 1986.

[17] Michael Monagan and Roman Pearce. Rational simplification modulo a polynomial ideal. In *ISSAC'06*, pages 239–245. ACM, 2006.

[18] Peter L. Montgomery. Speeding the pollard and elliptic curve methods of factorization. *Mathematics of Computation*, 48(177):243–264, 1987.

[19] Nigel P. Smart. The Hessian form of an elliptic curve. In *CHES 2001*, volume 2162 of *LNCS*, pages 118–125. Springer, 2001.

A Appendix

The appendix is composed of three tables. The underlined values are the fastest timings in that column. The rows are sorted with respect to the column (D=0, S=0.8M) in descending order. "REG" stands for the number of coordinates in each system. "DBL", "mADD", "reADD", "ADD", and "uADD" stands for the costs of doubling, mixed addition, readdition, addition and unified addition, respectively. Some forms have alternative versions due to alternative operation counts for different S/M and D/M values. It is possible to include more versions due to the richness of current formulae and algorithms. On the other hand, this will decrease readability of the tables. Therefore, we only provide the most significant cases. The references for the comparisons are;

- Doche/Icart/Kohel-2; all operations from [10, 2]. The appearance of [2] is to emphasize that better operation counts are available and is obtained from this database. This is the same for other dot items,
- Edwards; all operations for v.1a, v.1b, and doubling for v.2 from [4],
- Hessian; doubling for v.1, v.2 from [12], readdition, mixed addition, and addition for v.1, addition for v.2 from [8],
- Inverted Edwards; all operations for v.1 and doubling, readdition and addition for v.2 from [5],
- Jacobian ($a = -3$) and Jacobian; all operations from [8, 9, 2],
- Jacobi-intersection; doubling, addition, readdition, from [15, 2], mixed addition from [12],

- Jacobi-quartic; doubling[9], readdition, mixed addition[10] and addition for v.1a, v.1b from [6, 11, 2],
- Modified Jacobi-quartic doubling for v.1, v.2a, v.2b [12, 2], readdition, mixed-addition, and addition for v.1 from [11, 2],
- Projective ($a = -3$) and Projective doubling, readdition, mixed addition and addition for [8, 2], unified addition from [7, 2].

The rest are from this paper and they are highlighted in the tables.

[9] The 2M+6S+2D doubling formulae/algorithm by Hisil, Dawson and Carter reported in [2] cost 1M+7S+2D if the coordinate X_3 is computed as $(X_1Z_1 + Y_1)^2 - (X_1Z_1)^2 - Y_1^2$.

[10] The mixed addition costs 7M+3S+1D on the Jacobi-quartic coordinates when $Z_2 = 1$ for the 8M+3S+1D readdition algorithm in [11, 2].

Table 1: Point multiplication cost estimates per bit for "Non-adjacent form without precomputation" method.

System	REG	DBL			mADD			1 DBL, 0.33 mADD per bit								
		M	S	D	M	S	D	D=M S=M	D=M S=0.8M	D=M S=0.67M	D=0.5M S=M	D=0.5M S=0.8M	D=0.5M S=0.67M	D=0 S=M	D=0 S=0.8M	D=0 S=0.67M
Projective	3	5	6	1	9	2	0	15.667	14.333	13.467	15.167	13.833	12.967	14.667	13.333	12.467
Projective (a=-3)	3	7	3	0	9	2	0	13.667	12.933	12.457	13.667	12.933	12.457	13.667	12.933	12.457
Jacobi-quartic v.1a	3	1	9	0	7	3	1	13.667	11.667	10.367	13.500	11.500	10.200	13.333	11.333	10.033
Hessian v.1	3	7	1	0	10	0	0	11.333	11.133	11.003	11.333	11.133	11.003	11.333	11.133	11.003
Hessian v.2	3	3	6	0	**5**	**6**	**0**	12.667	11.067	10.027	12.667	11.067	10.027	12.667	11.067	10.027
Modified Hessian	9	**3**	**6**	**0**	**5**	**6**	**0**	12.667	11.067	10.027	12.667	11.067	10.027	12.667	11.067	10.027
Jacobian	3	1	8	1	7	4	0	13.667	11.800	10.587	13.167	11.300	10.087	12.667	10.800	9.587
Jacobian (a=-3)	3	3	5	0	7	4	0	11.667	10.400	9.577	11.667	10.400	9.577	11.667	10.400	9.577
Jacobi-intersection v.1	4	3	4	0	10	2	1	11.333	10.400	9.793	11.167	10.233	9.627	11.000	10.067	9.460
Jacobi-quartic v.1b	3	1	7	2	7	3	1	13.667	12.067	11.027	12.500	10.900	9.860	11.333	9.733	8.693
Doche/Icart/Kohel-2	4	2	5	2	8	4	1	13.333	12.067	11.243	12.167	10.900	10.077	11.000	9.733	8.910
Jacobi-intersection v.2	4	**2**	**5**	**1**	**10**	**1**	**2**	12.333	11.267	10.573	11.500	10.433	9.740	10.667	9.600	8.907
Modified Jacobi-intersection	6	**2**	**5**	**1**	**10**	**1**	**2**	12.333	11.267	10.573	11.500	10.433	9.740	10.667	9.600	8.907
Edwards v.1b	3	3	4	0	6	5	1	11.000	9.867	9.130	10.833	9.700	8.963	10.667	9.533	8.797
Edwards v.1a	3	3	4	0	9	1	1	10.667	9.800	9.237	10.500	9.633	9.070	10.333	9.467	8.903
Modified Jacobi-quartic v.1	6	3	4	0	7	3	1	10.667	9.667	9.017	10.500	9.500	8.850	10.333	9.333	8.683
Inverted Edwards v.2	3	3	4	1	**9**	**0**	**0**	11.000	10.200	9.680	10.500	9.700	9.180	10.000	9.200	8.680
Edwards v.2	3	3	4	0	**9**	**0**	**0**	10.000	9.200	8.680	10.000	9.200	8.680	10.000	9.200	8.680
Inverted Edwards v.1	3	3	4	1	8	1	1	11.333	10.467	9.903	10.667	9.800	9.237	10.000	9.133	8.570
Modified Jacobi-quartic v.2a	5	3	4	0	**6**	**3**	**1**	10.333	9.333	8.683	10.167	9.167	8.517	10.000	9.000	8.350
Modified Jacobi-quartic v.2b	6	3	4	0	**6**	**3**	**1**	10.333	9.333	8.683	10.167	9.167	8.517	10.000	9.000	8.350
Modified Jacobi-quartic v.3a	5	**2**	**5**	**1**	**6**	**3**	**1**	11.333	10.133	9.353	10.667	9.467	8.687	10.000	8.800	8.020
Modified Jacobi-quartic v.3b	6	**2**	**5**	**1**	**6**	**3**	**1**	11.333	10.133	9.353	10.667	9.467	8.687	10.000	8.800	8.020

Table 2: Point multiplication cost estimates per bit for “Signed 4-bit Sliding Windows” method.

System	REG	DBL			reADD			mADD			ADD			0.98 DBL, 0.17 reADD, 0.025 mADD, 0.0035 ADD per bit								
		M	S	D	M	S	D	M	S	D	M	S	D	D=M S=M	D=M S=0.8M	D=M S=0.67M	D=0.5M S=M	D=0.5M S=0.8M	D=0.5M S=0.67M	D=0 S=M	D=0 S=0.8M	D=0 S=0.67M
Projective	3	5	6	1	12	2	0	9	2	0	12	2	0	14.433	13.177	12.360	13.942	12.685	11.869	13.451	12.194	11.377
Projective (a=-3)	3	7	3	0	12	2	0	9	2	0	12	2	0	12.468	11.801	11.368	12.468	11.801	11.368	12.468	11.801	11.368
Jacobi-quartic v.1a	3	1	9	0	8	3	1	7	3	1	10	3	1	12.136	10.251	9.026	12.039	10.154	8.929	11.942	10.057	8.832
Hessian v.1	3	7	1	0	12	0	0	10	0	0	12	0	0	10.140	9.943	9.816	10.140	9.943	9.816	10.140	9.943	9.816
Jacobian	3	1	8	1	10	4	0	7	4	0	11	5	0	12.475	10.748	9.624	11.984	10.256	9.133	11.493	9.765	8.642
Hessian v.2	3	3	6	0	**6**	**6**	**0**	**5**	**6**	**0**	12	0	0	11.147	9.739	8.824	11.147	9.739	8.824	11.147	9.739	8.824
Modified Hessian	9	**3**	**6**	**0**	**6**	**6**	**0**	**5**	**6**	**0**	**6**	**6**	**0**	11.147	9.735	8.817	11.147	9.735	8.817	11.147	9.735	8.817
Jacobian (a=-3)	3	3	5	0	10	4	0	7	4	0	11	5	0	10.511	9.372	8.632	10.511	9.372	8.632	10.511	9.372	8.632
Doche/Icart/Kohel-2	4	2	5	2	12	5	1	8	4	1	12	5	1	12.213	11.042	10.280	11.134	9.962	9.201	10.054	8.883	8.121
Jacobi-intersection v.1	4	3	4	0	11	2	1	10	2	1	13	2	1	9.577	8.714	8.152	9.480	8.617	8.055	9.383	8.520	7.958
Jacobi-quartic v.1b	3	1	7	2	8	3	1	7	3	1	10	3	1	12.136	10.644	9.675	11.057	9.565	8.595	9.977	8.485	7.516
Edwards v.1b	3	3	4	0	7	5	1	6	5	1	7	5	1	9.376	8.396	7.759	9.279	8.299	7.662	9.182	8.202	7.565
Jacobi-intersection v.2	4	**2**	**5**	**1**	**11**	**1**	**2**	**10**	**1**	**2**	**13**	**1**	**2**	10.560	9.539	8.875	9.874	8.853	8.189	9.189	8.168	7.504
Edwards v.1a	3	3	4	0	10	1	1	9	1	1	10	1	1	9.182	8.357	7.821	9.085	8.260	7.724	8.988	8.163	7.627
Modified Jacobi-intersection	6	**2**	**5**	**1**	**11**	**1**	**2**	**10**	**1**	**2**	**11**	**1**	**2**	10.553	9.531	8.868	9.867	8.846	8.182	9.182	8.161	7.497
Edwards v.2	3	3	4	0	**9**	**2**	**0**	**9**	**0**	**0**	**11**	**0**	**0**	8.963	8.111	7.557	8.963	8.111	7.557	8.963	8.111	7.557
Modified Jacobi-quartic v.1	6	3	4	0	8	3	1	7	3	1	8	3	1	9.182	8.280	7.693	9.085	8.183	7.596	8.988	8.085	7.499
Inverted Edwards v.2	3	3	4	1	9	1	1	**9**	**0**	**0**	9	1	1	9.946	9.126	8.593	9.370	8.550	8.017	8.794	7.974	7.441
Inverted Edwards v.1	3	3	4	1	9	1	1	8	1	1	9	1	1	9.970	9.146	8.609	9.382	8.557	8.021	8.794	7.969	7.433
Modified Jacobi-quartic v.2a	5	3	4	0	**7**	**3**	**1**	**6**	**3**	**1**	**7**	**4**	**1**	8.991	8.088	7.501	8.894	7.991	7.404	8.797	7.894	7.307
Modified Jacobi-quartic v.2b	6	3	4	0	**7**	**3**	**1**	**6**	**3**	**1**	**7**	**3**	**1**	8.988	8.085	7.499	8.891	7.988	7.402	8.794	7.891	7.305
Modified Jacobi-quartic v.3a	5	**2**	**5**	**1**	**7**	**3**	**1**	**6**	**3**	**1**	**7**	**4**	**1**	9.974	8.874	8.159	9.386	8.286	7.571	8.797	7.698	6.983
Modified Jacobi-quartic v.3b	6	**2**	**5**	**1**	**7**	**3**	**1**	**6**	**3**	**1**	**7**	**3**	**1**	9.970	8.871	8.157	9.382	8.283	7.569	8.794	7.695	6.981

Table 3: Point multiplication cost estimates per bit for “Non-adjacent form without precomputation with SPA protection” algorithm.

System	REG	uADD			1.33 uADD per bit								
		M	S	D	D=M S=M	D=M S=0.8M	D=M S=0.67M	D=0.5M S=M	D=0.5M S=0.8M	D=0.5M S=0.67M	D=0 S=M	D=0 S=0.8M	D=0 S=0.67M
Projective	3	11	6	1	24.000	22.400	21.360	23.333	21.733	20.693	22.667	21.067	20.027
Projective (a=-1)	3	13	3	0	21.333	20.533	20.013	21.333	20.533	20.013	21.333	20.533	20.013
Jacobi-intersection v.1	4	13	2	1	21.333	20.800	20.453	20.667	20.133	19.787	20.000	19.467	19.120
Jacobi-intersection v.2	4	**13**	**1**	**2**	21.333	21.067	20.893	20.000	19.733	19.560	18.667	18.400	18.227
Jacobi-quartic v.1a, v.1b	3	10	3	1	18.667	17.867	17.347	18.000	17.200	16.680	17.333	16.533	16.013
Hessian v.1, v.2	3	12	0	0	16.000	16.000	16.000	16.000	16.000	16.000	16.000	16.000	16.000
Modified Jacobi-intersection	6	**11**	**1**	**2**	18.667	18.400	18.227	17.333	17.067	16.893	16.000	15.733	15.560
Edwards v.1b	3	7	5	1	17.333	16.000	15.133	16.667	15.333	14.467	16.000	14.667	13.800
Edwards v.1a	3	10	1	1	16.000	15.733	15.560	15.333	15.067	14.893	14.667	14.400	14.227
Modified Hessian	9	**6**	**6**	**0**	16.000	14.400	13.360	16.000	14.400	13.360	16.000	14.400	13.360
Modified Jacobi-quartic v.1	6	8	3	1	16.000	15.200	14.680	15.333	14.533	14.013	14.667	13.867	13.347
Modified Jacobi-quartic v.2a, v.3a	5	**7**	**4**	**1**	16.000	14.933	14.240	15.333	14.267	13.573	14.667	13.600	12.907
Inverted Edwards v.1	3	9	1	1	14.667	14.400	14.227	14.000	13.733	13.560	13.333	13.067	12.893
Modified Jacobi-quartic v.2b, v.3b	6	**7**	**3**	**1**	14.667	13.867	13.347	14.000	13.200	12.680	13.333	12.533	12.013