

Efficient Explicit Formulae for Genus 2 Hyperelliptic Curves over Prime Fields and Their Implementations

Xinxin Fan and Guang Gong

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada
x5fan@engmail.uwaterloo.ca, G.Gong@ece.uwaterloo.ca

Abstract. We analyze all the cases and propose the corresponding explicit formulae for computing $2D_1 + D_2$ in one step from given divisor classes D_1 and D_2 on genus 2 hyperelliptic curves defined over prime fields. Compared with naive method, the improved formula can save two field multiplications and one field squaring each time when the arithmetic is performed in the most frequent case. Furthermore, we present a variant which trades one field inversion for fourteen field multiplications and two field squarings by using Montgomery's trick to combine the two inversions. Experimental results show that our algorithms can save up to 13% of the time to perform a scalar multiplication on a general genus 2 hyperelliptic curve over a prime field, when compared with the best known general methods.

Keywords: Genus 2 hyperelliptic curves, explicit formulae, Cantor's algorithm, Harley's variant, efficient implementation.

1 Introduction

In 1988, Koblitz proposed for the first time to use the Jacobian of a hyperelliptic curve (HEC) defined over a finite field to implement cryptographic protocols based on the difficulty of the discrete logarithm problem [14]. During the past few years, hyperelliptic curve cryptosystems (HECC) have become increasing popular for use in practice to provide an alternative to the widely used elliptic curve cryptosystems (ECC) because of much shorter operand length than that of ECC. Moreover, recent research has also shown that HECC are well suited for various software and hardware platforms and their performance is compatible to that of ECC [1,4,21,22].

The most important and expensive operation in ECC and HECC is the scalar multiplication by an integer k , i.e., computing a scalar multiple kP of a point P on the points group or kD of a divisor class D on the Jacobian, where k might be 160 bits or more. Various techniques for efficiently computing the scalar multiplication have been proposed [2,13]. For general elliptic curves, Eisenträger *et al.* proposed a very elegant method for accelerating the scalar multiplication [10].

Their improvements are based on the efficient computation of $2P + Q$ in one step from given points P and Q on an elliptic curve. Since the point doubling is slightly more expensive than the point addition in the group operations of ECC, it is more efficient to calculate $2P + Q$ as $P + (P + Q)$ than first doubling P and then adding Q . This trick can save one field multiplication each time the certain sequence of operations occurs. Furthermore, their method finds applications to simultaneous multiple scalar multiplication, the Elliptic Curve Method of factorization, as well as the computation of the Weil and Tate pairings [10]. In the rest of this paper I represents a field inversion, M a field multiplication, and S a field squaring.

This work generalizes Eisenträger *et al.*'s idea to genus 2 HECs over prime fields where the group doubling costs two more field squarings than the group addition [16]. We analyze all the possible cases during the computation procedure of $2D_1 + D_2$ from given divisor classes D_1 and D_2 on a genus 2 HEC over \mathbb{F}_p . For the most frequent case, we propose a basic algorithm and its variant which cost $2I + 42M + 5S$ and $1I + 56M + 7S$, respectively, to compute $2D_1 + D_2$ in one step. Compared to the naive method using two separate group additions, our basic algorithm can save $2M + 1S$. In the variant, which is faster whenever one inversion is more expensive than about sixteen field multiplications, Montgomery's trick [8] is employed to combine the two inversions in the basic algorithm. Furthermore, we implement the proposed algorithms on a Pentium processor to verify the correctness and test the performance of our new explicit formulae. For genus 2 HECs over binary fields, the fastest doubling formula, which requires only half the time of an addition, has been obtained by Lange and Stevens for a special family of curves [17]. We note that the Eisenträger *et al.*'s trick can not be applied to optimize the computation of $2D_1 + D_2$ for the special family of genus 2 curves over binary fields when the group doubling is more efficient than the group addition.

The rest of this paper is organized as follows: Section 2 gives a short introduction to the mathematical background of genus 2 HECs over prime fields. Section 3 makes a thorough case study for the computation of $2D_1 + D_2$, presents the corresponding explicit formulae and analyzes the cost of the NAF scalar multiplication. Section 4 gives the experimental results of our newly derived explicit formulae. Finally, Section 5 ends this contribution.

2 Mathematical Background on Genus 2 Hyperelliptic Curves over Prime Fields

In this section, we present a brief introduction to the theory of genus 2 hyperelliptic curves over prime fields, restricting attention to the material which is relevant to this work. For more details, the reader is referred to [3,6,15,18].

Let \mathbb{F}_q be a finite field of characteristic $p \neq 2$, $q = p^n$, and let $\overline{\mathbb{F}}_q$ denote the algebraic closure of \mathbb{F}_q . Let $\mathbb{F}_q(C)/\mathbb{F}_q$ be a quadratic function field defined via an equation

$$C : Y^2 = F(X) \tag{1}$$

where $F(X) = X^5 + f_4X^4 + f_3X^3 + f_2X^2 + f_1X + f_0 \in \mathbb{F}_q[X]$ is a monic and square-free polynomial of degree 5. The curve C/\mathbb{F}_q associated with this function field is called a *hyperelliptic curve of genus 2 defined over \mathbb{F}_q* . For our purpose it is enough to consider a point P as an ordered pair $P = (x, y) \in \overline{\mathbb{F}}_q^2$ which satisfies $y^2 = F(x)$. Besides these tuples there is one point P_∞ at infinity. The inverse of P is defined as $-P = (x, -y)$. We call a point P that satisfies $P = -P$ a *ramification point*. Note that for $p \neq 5$ the transform $X \rightarrow X - f_4/5$ makes the coefficient of X^4 in $F(X)$ zero.

The divisor class group $\mathcal{J}_C(\mathbb{F}_q)$ of C forms a finite Abelian group and therefore we can construct cryptosystems whose security is based on the difficulty of the discrete logarithm problem on the Jacobian of C . Each element of the Jacobian can be represented uniquely by a so-called reduced divisor [6]. Mumford [19] showed that a reduced divisor can be represented by means of two polynomials $U(X), V(X) \in \mathbb{F}_q[X]$, where $U(X)$ and $V(X)$ satisfy the following three conditions: (i) $U(X)$ is monic, (ii) $\deg V(X) < \deg U(X) \leq 2$, and (iii) $U(X) \mid V(X)^2 - F(X)$. In the remainder of this paper, we will use the notation $[U, V]$ for the divisor class represented by $U(X)$ and $V(X)$. For a genus 2 HEC, we have commonly $[U, V] = [X^2 + u_1X + u_0, v_1X + v_0]$.

Cantor's algorithm [6] describes how to perform the group addition of two divisor classes in Mumford's representation. We review Cantor's algorithm for genus 2 HECs over prime fields in the following Algorithm 1. Cantor's algorithm only involves polynomial arithmetic over the finite field in which the divisor class group is defined. However, there are some redundant computations of the polynomial's coefficients in this classical algorithm. In order to simplify Cantor's algorithm, Harley proposed the first explicit formulae for a group addition and a group doubling of divisor classes on $\mathcal{J}_C(\mathbb{F}_q)$ in 2000. In [11], Gaudry and Harley significantly reduced the computational complexity of the group operations by distinguishing different cases according to the properties of the input divisor classes. They presented a very efficient algorithm, which uses many modern polynomial computation techniques such as Chinese remainder theorem, Newton's iteration, and Karatsuba's multiplication. Algorithm 2 gives a high level description of Harley's variant for adding two reduced divisor classes in the most frequent case for genus 2 HECs over prime fields. The most frequent cases mean that for the addition the inputs are two co-prime polynomials of degree 2, which occur with the overwhelming probability [20], and the remainder cases are called exceptional cases. For more details about Cantor's algorithm and Harley's variant, the reader is referred to [3,16,22].

3 Efficient Algorithms for Computing $2D_1 + D_2$

In this section we adapt the idea of [10] to genus 2 HECs over prime fields. We obtain $D_3 = 2D_1 + D_2$ by the following two steps: we first compute $D' = [U', V'] = D_1 + D_2$ and omit the computation of the coefficients of V' because V' will not be used in the next phase. And then, we find $D_3 = D' + D_1$. Hence, we use two group additions to form $2D_1 + D_2$ instead of a group addition and

Algorithm 1. Cantor's Algorithm for Group Addition ($g = 2, \mathbb{F}_p$)

Input: $D_1 = [U_1, V_1], D_2 = [U_2, V_2], C : Y^2 = F(X)$

Output: $D_3 = [U_3, V_3]$ reduced with $D_3 \equiv D_1 + D_2$

1. Compute $d_1 = \gcd(U_1, U_2) = e_1U_1 + e_2U_2$
 2. Compute $d = \gcd(d_1, V_1 + V_2) = c_1d_1 + c_2(V_1 + V_2)$
 3. Let $s_1 = c_1e_1, s_2 = c_1e_2, s_3 = c_2$
 4. $U' = \frac{U_1U_2}{d^2}$
 5. $V' = \frac{s_1U_1V_2 + s_2U_2V_1 + s_3(V_1V_2 + F)}{d} \bmod U'$
 6. $U_3 = \frac{F - V'^2}{U'}$, $V_3 = -V' \bmod U_3$
 7. make U_3 monic
-

Algorithm 2. Harley's Variant for Group Addition ($g = 2, \mathbb{F}_p$)

Input: $D_1 = [U_1, V_1], D_2 = [U_2, V_2], C : Y^2 = F(X)$

Output: $D_3 = [U_3, V_3]$ reduced with $D_3 \equiv D_1 + D_2$

1. $K = \frac{F - V_1^2}{U_1}$ (exact division)
 2. $S \equiv \frac{V_2 - V_1}{U_1} \bmod U_2$
 3. $L = SU_1$
 4. $U_3 = \frac{K - S(L + 2V_1)}{U_2}$ (exact division)
 5. make U_3 monic
 6. $V_3 \equiv -(L + V_1) \bmod U_3$
-

a group doubling. To derive explicit formulae, we first study all the exceptional cases during the computation $2D_1 + D_2$ based on the properties of the input divisor classes and the immediate result D' . We then determine how many field operations are required to calculate $2D_1 + D_2$ in one step in the most frequent case. Furthermore, we also propose a variant of our basic algorithm by using Montgomery's trick to compute the two inversions simultaneously at cost of some multiplications, which will be more efficient whenever a field inversion is more expensive than about sixteen field multiplications.

3.1 Explicit Formulae in Exceptional Cases

In this subsection we discuss all the exceptional cases appearing in the procedure of calculating $2D_1 + D_2$. Suppose that $D_1 = [U_1, V_1]$ and $D_2 = [U_2, V_2]$ are two reduced divisor classes as the inputs of the composition step of the Cantor's algorithm. The final output is $D_3 = 2D_1 + D_2 = [U_3, V_3]$. We need to distinguish the following cases:

1. U_1 is of degree zero, this is only possible in the case $[U_1, V_1] = [1, 0]$, i.e. D_1 is the zero element of the divisor class group. The result of $2D_1 + D_2$ is the second class $D_2 = [U_2, V_2]$.

2. U_1 is of degree one and U_2 has degree zero, one or full degree. Let $U_1 = X + u_{10}$ and $V_1 = v_{10} \neq 0$ is a constant.

A. Assume $\deg U_2 = 0$, i.e., D_2 is the zero element of the divisor class group. Therefore, the result of $2D_1 + D_2$ is $2D_1$ and we double the divisor D_1 with $1I + 4M + 1S$ to obtain

$$\begin{aligned} U_3 &= U_1^2 = (X + u_{10})^2, \\ V_3 &= \frac{F'(-u_{10})(X + u_{10})}{2v_{10}} + v_{10}. \end{aligned} \tag{2}$$

B. Assume $\deg U_2 = 1$, i.e., $U_2 = X + u_{20}$ and $V_2 = v_{20} \neq 0$ is a constant.

i. If $U_1 = U_2$ and $V_1 = -V_2$, the result of $D_1 + D_2$ is the zero element $[1, 0]$. Hence, we get $2D_1 + D_2 = [1, 0] + D_1 = D_1$;

ii. If $U_1 = U_2$ and $V_1 = V_2$, the result of $2D_1 + D_2$ is $3D_1$, which can be computed with $1I + 12M + 4S$ (See Table 5 in the appendix).

iii. Otherwise the result of $D_1 + D_2$ is $[U', V']$ where

$$\begin{aligned} U' &= U_1U_2 = (X + u_{10})(X + u_{20}), \\ V' &= \frac{(v_{20} - v_{10})X + v_{20}u_{10} - v_{10}u_{20}}{u_{10} - u_{20}}. \end{aligned} \tag{3}$$

And then we use Table 6 (see the appendix) to obtain $2D_1 + D_2$ in an additional $1I + 18M + 4S$.

C. Assume $\deg U_2 = 2$, i.e. $U_2 = X^2 + u_{21}X + u_{20}$ and $V_2 = v_{21}X + v_{20}$. Then the corresponding divisors are given by $D_1 = (P_1) - (P_\infty)$ and $D_2 = (P_2) + (P_3) - 2(P_\infty)$, with $P_i \neq P_\infty$ ($i = 1, 2, 3$).

i. If $U_2(-u_{10}) \neq 0$ then P_1 and $-P_1$ do not occur in D_2 . This case is dealt with Table 7 (see the appendix). We can obtain $2D_1 + D_2$ at the cost of $I + 28M + 4S$.

ii. Otherwise if $V_2(-u_{10}) = -v_{10}$ the $-P_1$ occurs in D_2 and the result of $D_1 + D_2$ is $D' = [U', V'] = [X + u_{21} - u_{10}, v_{21}(-u_{21} + u_{10}) + v_{20}]$ because $-u_{21}$ equals the sum of the x -coordinates of the points. And then we compute D_3 using (2), unless $D_2 = 2(-P_1) - 2(P_\infty)$ where we can obtain $D_3 = 2D_1 + D_2 = [1, 0]$.

iii. The remaining case is that P_1 occurs in D_2 . If $D_2 = 2(P_1) - 2(P_\infty) = 2D_1$, which holds if $u_{21} = 2u_{10}$ and $u_{20} = u_{10}^2$, then we have $2D_1 + D_2 = 2D_2$. Therefore, we obtain D_3 by doubling a class D_2 of order different from 2 and with first polynomial of full degree as in 3.A. Otherwise we first use Table 6 (see the appendix) to compute $D' = [U', V'] = [X + u'_1X + u'_0, v'_1X + v'_0] = D_1 + D_2$ with $1I + 18M + 4S$ and then differentiate the following three cases to obtain $D_3 = D' + D_1$:

a. If $U'(-u_{10}) \neq 0$ then P_1 and $-P_1$ do not occur in the support set of D' . In this case, D_3 can be calculated with the explicit addition formula of the case of $\deg U_1 = 1$ and $\deg U_2 = 2$ in [16] at the cost of $1I + 10M + S$.

- C. Assume $\deg U_2 = 2$, i.e. $U_2 = X^2 + u_{21}X + u_{20}$ and $V_2 = v_{21}X + v_{20}$. The corresponding divisor is given by $D_2 = (P_3) + (P_4) - 2(P_\infty)$ with $P_i \neq P_\infty$ ($i = 3, 4$).
- i. Let $U_1 = U_2$. This means that the x -coordinates of P_i and P_{i+2} ($i = 1, 2$) are equal for an appropriate ordering.
 - a. If $V_1 \equiv -V_2 \pmod{U_1}$ then we obtain $2D_1 + D_2 = D_1 + [1, 0] = D_1$.
 - b. If $V_1 = V_2$ then we have $2D_1 + D_2 = 3D_1$. We first double D_1 to get D' based on the two cases in 3.A. If the degree of U' is one, then we need to consider three subcases in 2.C.iii. Otherwise, we differentiate two subcases in 3.B.iii to compute D_3 .
 - c. The remaining case is that $P_i = P_{i+2}$ and $P_j \neq P_{j+2}$ ($i, j \in \{1, 2\}$ and $i \neq j$) is the opposite of P_{j+2} . Without loss of generality, we assume $P_1 = P_3$ and $P_2 \neq P_4$ is the opposite of P_4 . We first calculate $D' = D_1 + D_2 = 2(P_1) - 2(P_\infty)$ by using (1) to double the divisor class $[X - (v_{10} - v_{20}) / (v_{21} - v_{11}), V_1((v_{10} - v_{20}) / (v_{21} - v_{11}))]$. And then we calculate $D_3 = D' + D_1$ by considering two subcases in 3.B.iii.
- ii. For the remainder cases $U_1 \neq U_2$, the following possibilities may appear:
 - a. If $\text{res}(U_1, U_2) \neq 0$ then there is no point in the support of D_1 which is equal to a point or its opposite in the support of D_2 . We first only compute the first part U' of D' with the addition explicit formula of the most frequent case in [16]. And then we require to consider the following three subcases:
 1. If the degree of U' is one, which appears when $s'_1 = 0$ (see Table 1), we first calculate the second part V' of D' with the addition explicit formula of the special case in [16]. And then we need to consider three subcases in 2.C.iii to compute D_3 .
 2. If $\deg U' = 2$ and $\text{res}(U_1, U') = 0$, we first calculate the second part V' of D' with the addition explicit formula of the most frequent case in [16]. And then we compute D_3 with Table 9 (see the appendix).
 3. The remainder case is $\deg U' = 2$ and $\text{res}(U_1, U') \neq 0$. This is the most frequent case and we will deal with this case in the next subsection.
 - b. If $\text{res}(U_1, U_2) = 0$ then we first compute D' with Table 9 (see the appendix). If the degree of U' is one, then we need to consider three subcases in 2.C.iii. Otherwise, we differentiate two subcases in 3.B.iii to compute D_3 .

Although there are many exceptional cases during the computation of $2D_1 + D_2$, most frequently we are in the case of $\gcd(U_1, U_2) = \gcd(U_1, U') = 1$ and U' being quadratic. Therefore, if we can reduce the computational complexity of explicit formulae in the most frequent case, the performance of the whole cyptosystem will be improved on average.

3.2 Explicit Formulae in the Most Frequent Case

In this subsection, we present efficient explicit formulae for computing $2D_1 + D_2$ in the most frequent case where U_1, U_2 and U' are quadratic and $\gcd(U_1, U_2) = \gcd(U_1, U') = 1$. Studying Harley’s variant carefully, we note that the polynomial V' in the intermediate result D' only is used to obtain S in the second group addition (see Step 2 in Algorithm 2). Therefore, when we substitute the expression of V' into S , we find the following important lemma which results in a significant speedup for calculating $2D_1 + D_2$.

Lemma 1. *Let C be a genus 2 HEC over \mathbb{F}_q given by the equation (1). Assume that $D_1 = [U_1, V_1]$, $D_2 = [U_2, V_2]$ and $D' = [U', V'] = D_1 + D_2$ are reduced divisor classes in the Jacobian $\mathcal{J}_C(\mathbb{F}_q)$ of C and satisfy that U_1, U_2 and U' are quadratic, and $\gcd(U_1, U_2) = \gcd(U_1, U') = 1$. Let S and S' satisfy the congruent relations: $S \equiv \frac{V_2 - V_1}{U_1} \pmod{U_2}$ and $S' \equiv \frac{V' - V_1}{U_1} \pmod{U'}$, then we have*

$$S' \equiv -S - \frac{2V_1}{U_1} \pmod{U'}$$

Proof. From Harley’s variant (see Algorithm 2), we know that

$$V' \equiv -(SU_1 + V_1) \pmod{U'}$$

Substitute V' into S' , we obtain

$$S' \equiv \frac{V' - V_1}{U_1} \equiv \frac{-SU_1 - 2V_1}{U_1} \equiv -S - \frac{2V_1}{U_1} \pmod{U'}$$

Lemma 1 suggests that we can eliminate the computation of V' during the procedure of calculating $2D_1 + D_2$. Table 1 presents our new explicit formula (Basic Algorithm) for computing $2D_1 + D_2$ on a genus 2 HEC over \mathbb{F}_p in the most frequent case.

Table 1. Explicit Formula for $2D_1 + D_2$ on a HEC of Genus 2 over \mathbb{F}_p – Basic Version

Input	Genus 2 HEC $C : Y^2 = F(X), F = X^5 + f_3X^3 + f_2X^2 + f_1X + f_0$; Reduced Divisors $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$, $U_1 = X^2 + u_{11}X + u_{10}, V_1 = v_{11}X + v_{10}$; $U_2 = X^2 + u_{21}X + u_{20}, V_2 = v_{21}X + v_{20}$;	
Output	Reduced Divisor $D_3 = (U_3, V_3) = 2D_1 + D_2$; $U_3 = X^2 + u_{31}X + u_{30}, V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute the resultant r of U_1 and U_2: $i_1 = u_{21} - u_{11}, w = u_{10} - u_{20}, i_0 = i_1 u_{21} + w, r = i_0 w + i_1^2 u_{20}$;	$3M, 1S$
2	Compute the pseudo-inverse $I = i_1 X + i_0 \equiv r/U_1 \pmod{U_2}$:	–

Table 1. (continued)

3	Compute $S' = s'_1 X + s'_0 = rS \equiv (V_2 - V_1)I \pmod{U_2}$: $w_0 = v_{20} - v_{10}, w_1 = v_{21} - v_{11}, w_2 = i_0 w_0, w_3 = i_1 w_1, s'_0 = w_2 - u_{20} w_3$; $s'_1 = (i_0 + i_1)(w_0 + w_1) - w_2 - w_3(1 + u_{21})$; If $s'_1 = 0$, see 3.C.ii.a.1.	5M
4	Compute $S'' = X + s_0/s_1 = X + s'_0/s'_1$ and s_1: $w_1 = (rs'_1)^{-1}(= 1/r^2 s_1), w_2 = rw_1(= 1/s'_1), w_3 = rw_2(= 1/s_1)$; $w_4 = w_3^2, w_5 = s'_1 w_1, s''_0 = s'_0 w_2$;	1I, 5M, 1S
5	Compute $U' = (s(l + 2V_1) - k)/U_2 = X^2 + u'_1 X + u'_0$: $u'_0 = (s''_0 - u_{21})(s''_0 - i_1) + u_{11} s''_0 + w + 2v_{11} w_3 + (u_{11} + u_{21}) w_4$; $u'_1 = 2s''_0 - i_1 - w_4$;	4M
6	Compute the resultant \tilde{r} of U_1 and U': $\tilde{i}_1 = u'_1 - u_{11}, \tilde{w} = u_{10} - u'_0, \tilde{i}_0 = \tilde{i}_1 u'_1 + \tilde{w}, \tilde{r} = \tilde{i}_0 \tilde{w} + \tilde{i}_1^2 u'_0$;	4M, 1S
7	Compute the pseudo-inverse $\tilde{I} = \tilde{i}_1 X + \tilde{i}_0 \equiv \tilde{r}/U_1 \pmod{U'}$:	-
8	Compute $\tilde{S}' = \tilde{s}'_1 X + \tilde{s}'_0 = \tilde{r}\tilde{S} \equiv -\tilde{r}S'/r - 2V_1\tilde{I} \pmod{U'}$: $\tilde{r}' = \tilde{r}w_5, \tilde{w}_0 = \tilde{i}_0 v_{10}, \tilde{w}_1 = \tilde{i}_1 v_{11}, \tilde{s}'_0 = -[\tilde{r}' s'_0 + 2(\tilde{w}_0 - u'_0 \tilde{w}_1)]$; $\tilde{s}'_1 = -[\tilde{r}' s'_1 + 2(\tilde{i}_0 + \tilde{i}_1)(v_{10} + v_{11}) - \tilde{w}_0 - \tilde{w}_1(1 + u'_{11})]$; If $\tilde{s}'_1 = 0$, see below	7M
9	Compute $\tilde{S}'' = X + \tilde{s}_0/\tilde{s}_1 = X + \tilde{s}'_0/\tilde{s}'_1$ and \tilde{s}_1: $\tilde{w}_1 = (\tilde{r}\tilde{s}'_1)^{-1}(= 1/\tilde{r}^2 \tilde{s}_1), \tilde{w}_2 = \tilde{r}\tilde{w}_1(= 1/\tilde{s}'_1), \tilde{w}_3 = \tilde{s}'_1{}^2 \tilde{w}_1(= \tilde{s}_1)$; $\tilde{w}_4 = \tilde{r}\tilde{w}_2(= 1/\tilde{s}'_1), \tilde{w}_5 = \tilde{w}_4^2, \tilde{s}''_0 = \tilde{s}'_0 \tilde{w}_2$;	1I, 5M, 2S
10	Compute $\tilde{l}' = \tilde{S}'' u_1 = X^3 + \tilde{l}'_2 X^2 + \tilde{l}'_1 X + \tilde{l}'_0$: $\tilde{l}'_2 = u_{11} + \tilde{s}''_0, \tilde{l}'_1 = u_{11} \tilde{s}''_0 + u_{10}, \tilde{l}'_0 = u_{10} \tilde{s}''_0$;	2M
11	Compute $U_3 = (\tilde{s}(\tilde{l}' + 2V_1) - k)/U' = X^2 + u_{31} X + u_{30}$: $u_{30} = (\tilde{s}''_0 - u_1)(\tilde{s}''_0 - \tilde{i}_1) - u_0 + \tilde{l}'_1 + 2v_{11} \tilde{w}_4 + (u_1 + u_{11}) \tilde{w}_5$; $u_{31} = 2\tilde{s}''_0 - \tilde{i}_1 - \tilde{w}_5$;	3M
12	Compute $V_3 = -(\tilde{l}' + V_1) \pmod{U_3} = v_{31} X + v_{30}$: $w_1 = \tilde{l}'_2 - u_{31}, w_2 = u_{31} w_1 + u_{30} - \tilde{l}'_1, v_{31} = w_2 \tilde{w}_3 - v_{11}$; $w_2 = u_{30} w_1 - \tilde{l}'_0, v_{30} = w_2 \tilde{w}_3 - v_{10}$;	4M
Sum	$\tilde{s}'_1 \neq 0$	2I, 42M, 5S
9'	Compute \tilde{s}_0: $\tilde{w}_1 = \tilde{r}^{-1}, \tilde{s}_0 = \tilde{s}'_0 \tilde{w}_1$;	1I, 1M
10'	Compute $U_3 = (k - \tilde{s}(\tilde{l}' + 2V_1))/U' = X + u_{30}$: $u_{30} = -(u_1 + u_{11} + \tilde{s}''_0)$;	1S
11'	Compute $V_3 = -(\tilde{l}' + V_1) \pmod{U_3} = v_{30}$: $w_1 = \tilde{s}_0(u_1 + u_{30}) + v_{11}, w_2 = \tilde{s}_0 + v_{10}, v_{30} = u'_0 w_1 - w_2$;	2M
Sum	$\tilde{s}'_1 = 0$	2I, 31M, 4S

Our explicit formula of the basic version requires $2I + 42M + 5S$ to calculate $2D_1 + D_2$ for genus 2 HECs over \mathbb{F}_p . However, the naive method which separately computes the two divisor class additions will cost $2I + 44M + 6S$ [16]. Therefore, our improvements can save $2M + 1S$ each time the operation $2D_1 + D_2$ is performed.

We note that there exist two inversions in the above explicit formula of the basic version. Therefore, we propose a variant of the basic algorithm where we delay the inversion in Step 4 of Table 1 and combine it with the inversion in Step 6 of Table 1 using Montgomery’s trick of simultaneous inversions [8]. Table 2 presents the explicit formula for this variant of the basic algorithm.

Table 2. Explicit Formula for $2D_1 + D_2$ on a HEC of Genus 2 over \mathbb{F}_p - Variant

Input	Genus 2 HEC $C : Y^2 = F(X), F = X^5 + f_3X^3 + f_2X^2 + f_1X + f_0$; Reduced Divisors $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$, $U_1 = X^2 + u_{11}X + u_{10}, V_1 = v_{11}X + v_{10}$; $U_2 = X^2 + u_{21}X + u_{20}, V_2 = v_{21}X + v_{20}$;	
Output	Reduced Divisor $D_3 = (U_3, V_3) = 2D_1 + D_2$, $U_3 = X^2 + u_{31}X + u_{30}, V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute the resultant r of U_1 and U_2: $i_1 = u_{21} - u_{11}, w = u_{10} - u_{20}, i_0 = i_1u_{21} + w, r = i_0w + i_1^2u_{20}$;	$3M, 1S$
2	Compute the pseudo-inverse $I = i_1X + i_0 \equiv r/U_1 \pmod{U_2}$:	-
3	Compute $S' = s'_1X + s'_0 = rS \equiv (V_2 - V_1)I \pmod{U_2}$: $w_0 = v_{20} - v_{10}, w_1 = v_{21} - v_{11}, w_2 = i_0w_0, w_3 = i_1w_1, s'_0 = w_2 - u_{20}w_3$; $s'_1 = (i_0 + i_1)(w_0 + w_1) - w_2 - w_3(1 + u_{21})$; If $s'_1 = 0$, see 3.C.ii.a.1.	$5M$
4	Monic $S'' = X + s_0/s_1 = X + s'_0/s'_1$:	-
5	Compute $U' = (s(l + 2V_1) - k)/U_2 = X^2 + u'_1X + u'_0$: $s_q = s_1'^2, w_1 = i_1s'_1, w_2 = s'_0 - w_1, R = r^2, u'_1 = s'_1(s'_0 + w_2) - R$; $u'_0 = s'_0(w_2 - w_1) + i_0s_q + 2rv_{11}s'_1 + R(u_{11} + u_{21})$;	$7M, 2S$
6	Compute the resultant \tilde{r} of U_1 and U': $\tilde{i}_1 = u'_1 - u_{11}s_q, \tilde{w} = u_{10}s_q - u_{10}, \tilde{i}_0 = u_1\tilde{i}_1 + \tilde{w}s_q, \tilde{r} = \tilde{i}_0\tilde{w} + \tilde{i}_1^2u'_0$;	$6M, 1S$
7	Compute the pseudo-inverse $\tilde{I} = \tilde{i}_1X + \tilde{i}_0 \equiv \tilde{r}/U_1 \pmod{U'}$:	-
8	Compute $\tilde{S}' = \tilde{s}'_1X + \tilde{s}'_0 = \tilde{r}\tilde{S} \equiv -\tilde{r}S'/r - 2V_1\tilde{I} \pmod{U'}$: $\tilde{w}_0 = \tilde{i}_0v_{10}, \tilde{w}_1 = \tilde{i}_1v_{11}, \tilde{w}_2 = rs_q, \tilde{s}'_0 = -[\tilde{r}s'_0 + 2\tilde{w}_2(\tilde{w}_0 - u'_0\tilde{w}_1)]$; $\tilde{s}'_1 = -[\tilde{r}s'_1 + 2\tilde{w}_2(\tilde{i}_0 + \tilde{i}_1s_q)(v_{10} + v_{11}) - \tilde{w}_0 - \tilde{w}_1(s_q + u'_1)]$; If $\tilde{s}'_1 = 0$, see below	$11M$
9	Compute $\tilde{S}'' = X + \tilde{s}_0/\tilde{s}_1 = X + \tilde{s}'_0/\tilde{s}'_1$ and \tilde{s}_1: $t_1 = \tilde{r}\tilde{s}'_1, t_2 = (t_1\tilde{w}_2)^{-1}, t_3 = \tilde{w}_2t_2, t_4 = t_1t_2, t_5 = rt_4, t_6 = s_q t_4$; $\tilde{w}_1 = rt_3, t_7 = (t_6\tilde{s}'_1)^2, \tilde{w}_3 = t_7\tilde{w}_1, \tilde{w}_4 = \tilde{r}^2\tilde{w}_1, \tilde{w}_5 = \tilde{w}_4^2, \tilde{s}''_0 = \tilde{r}\tilde{s}'_0 t_3$;	$1I, 12M, 3S$
10	Adjust: $u'_1 = u_1 t_5, u'_0 = u'_0 t_5, \tilde{i}_1 = \tilde{i}_1 t_5$;	$3M$
11	Compute $\tilde{l} = \tilde{S}''u_1 = X^3 + \tilde{l}_2X^2 + \tilde{l}_1X + \tilde{l}_0$: $\tilde{l}_2 = u_{11} + \tilde{s}''_0, \tilde{l}_1 = u_{11}\tilde{s}''_0 + u_{10}, \tilde{l}_0 = u_{10}\tilde{s}''_0$;	$2M$
12	Compute $U_3 = (\tilde{s}(\tilde{l} + 2V_1) - k)/U' = X^2 + u_{31}X + u_{30}$: $u_{30} = (\tilde{s}''_0 - u'_1)(\tilde{s}''_0 - \tilde{i}_1) - u'_0 + \tilde{l}_1 + 2v_{11}\tilde{w}_4 + (u'_1 + u_{11})\tilde{w}_5$; $u_{31} = 2\tilde{s}''_0 - \tilde{i}_1 - \tilde{w}_5$;	$3M$
13	Compute $V_3 = -(\tilde{l} + V_1) \pmod{U_3} = v_{31}X + v_{30}$: $w_1 = \tilde{l}_2 - u_{31}, w_2 = u_{31}w_1 + u_{30} - \tilde{l}_1, v_{31} = w_2\tilde{w}_3 - v_{11}$; $w_2 = u_{30}w_1 - \tilde{l}_0, v_{30} = w_2\tilde{w}_3 - v_{10}$;	$4M$
Sum	$\tilde{s}_1 \neq 0$	$I, 56M, 7S$
9'	Compute \tilde{s}_0 and Adjust: $\tilde{w}_1 = (\tilde{r}s_q)^{-1}, t_1 = s_q\tilde{w}_1, t_2 = \tilde{r}\tilde{w}_1, \tilde{s}_0 = \tilde{s}'_0 t_1 s_q, u'_1 = u'_1 t_2, u'_0 = u'_0 t_2$;	$1I, 7M$
10'	Compute $U_3 = (k - \tilde{s}(\tilde{l} + 2V_1))/U' = X + u_{30}$: $u_{30} = -(u'_1 + u_{11} + \tilde{s}_0^2)$;	$1S$
11'	Compute $V_3 = -(\tilde{l} + V_1) \pmod{U_3} = v_{30}$: $w_1 = \tilde{s}_0(u'_1 + u_{30}) + v_{11}, w_2 = \tilde{s}_0 + v_{10}, v_{30} = u'_0 w_1 - w_2$;	$2M$
Sum	$\tilde{s}'_1 = 0$	$I, 38M, 6S$

In Table 2, the variant of the basic algorithm needs $I + 56M + 7S$ to calculate $2D_1 + D_2$ for genus 2 HECs over \mathbb{F}_p . Compared to our explicit formula of the basic version, we trade I for $14M + 2S$. Therefore, when we implement genus 2 HECC on some application environments where a field inversion is more expensive than fourteen field multiplications and two field squarings, the variant in Table 2 will be faster than the basic algorithm in Table 1.

3.3 Cost of the NAF Scalar Multiplication

The above trick of efficiently computing $2D_1 + D_2$ has found important applications in some scalar multiplication algorithms such as NAF, JSF and so on [7]. In this subsection, we only compare the average cost per bit scalar when implementing NAF scalar multiplication algorithm with the naive method and our newly derived formulae, respectively, because the NAF scalar multiplication algorithm will be used in our implementation in the next section. The results of comparisons are listed in the following Table 3 (The pre- and post-computations are neglected as in [7]).

Table 3. Average Cost Per Bit for NAF on Genus 2 HECs over \mathbb{F}_p

Method	Cost of $2D_1 + D_2$	Cost per bit scalar	$S = 0.8M$
Naive	$2I + 44M + 6S$	$\frac{4}{3}I + \frac{88}{3}M + \frac{16}{3}S$	$1.33I + 33.6M$
Basic Algorithm (Table 1)	$2I + 42M + 5S$	$\frac{4}{3}I + \frac{86}{3}M + 5S$	$1.33I + 32.67M$
Variant (Table 2)	$I + 56M + 7S$	$I + \frac{100}{3}M + \frac{17}{3}S$	$I + 37.87M$

From Table 3, we can see clearly that our basic algorithm saves about 2.8% cost for per bit scalar compared to the naive method and the break-even point of the performance between the basic algorithm and the variant is still when one inversion is equivalent to about sixteen field multiplications.

4 Implementation Results

We implement the proposed algorithms on a Pentium-4 @2.8GHz processor and with C programming language in order to check the correctness and test the performance of our explicit formulae. *Microsoft Developer Studio 6* is used for compilation and debugging. For genus 2 HECC over \mathbb{F}_q , the most efficient attack is Pollard’s Rho algorithm which takes $O(\sqrt{\#\mathcal{J}_C(\mathbb{F}_q)})$ group operations. This means that for genus 2 HECC a 80-bit finite field is enough to achieve the same security level as 160-bit ECC. Considering the security and efficiency of the implementation, we choose a Mersenne prime $p = 2^{89} - 1$ as the characteristic of the prime field \mathbb{F}_p and develop a fast library for the required field and group operations. The implementation of \mathbb{F}_p -arithmetic is basically due to [5,9] and further optimized by using the idea in [12] to yield a fast modular reduction

Table 4. Timings of Group Operation on Genus 2 HECs over $\mathbb{F}_{2^{89}-1}$

Method	$2D_1 + D_2$ in μs	Scalar Multiplication in ms	Performance Improvement
Naive	23.1	2.85	–
Basic Algorithm (Table 1)	21.7	2.78	2.46%
Variant (Table 2)	16.4	2.48	12.98%

procedure. Table 4 summarizes our implementation results and comparisons for the group operation $2D_1 + D_2$ and the NAF scalar multiplication algorithm.

The experimental results of Table 4 show that when compared to the implementation with the naive method the performance of genus 2 HECC can be improved by 2.46% and 12.98% with our basic algorithm and the variant, respectively. Furthermore, due to the high *MI*-ratio (the ratio of the timing of one inversion to one multiplication) in the target processor, the variant is about 10% faster than the basic algorithm.

5 Conclusion

In this paper, we propose the efficient algorithms for computing $2D_1 + D_2$ in one step for genus 2 HECs over prime fields. Our basic algorithm is the direct generalization of Eisenträger *et al.*'s idea, which can save $2M + 1S$ compared with the naive method in the most frequent case. The performance of the variant will be better than that of the basic algorithm whenever a field inversion is more expensive than about sixteen field multiplications. Based our new explicit formulae, we analyze the average cost of per bit scalar in the NAF scalar multiplication algorithm and implement fast genus 2 HECC over $\mathbb{F}_{2^{89}-1}$. The experimental results show that we can obtain up to 13% performance gain when implementing genus 2 HECC with our newly derived explicit formulae.

References

1. Avanzi, R.M.: Aspects of Hyperelliptic Curves over Large Prime Fields in Software Implementations. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 148–162. Springer, Heidelberg (2004)
2. Avanzi, R.M.: The Complexity of Certain Multi-Exponentiation Techniques in Cryptography. *Journal of Cryptology* 18(4), 357–373 (2005)
3. Avanzi, R.M., Cohen, H., Doche, C., Frey, G., Lange, T., Nguyen, K., Vercauteren, F.: *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Chapman & Hall/CRC, Boca Raton, Florida, USA (2006)
4. Avanzi, R.M., Thériault, N., Wang, Z.: Rethinking Low Genus Hyperelliptic Jacobian Arithmetic over Binary Fields: Interplay of Field Arithmetic and Explicit Formulae, Centre for Applied Cryptographic Research (CACR) Technical Reports, CACR 2006-07, available at <http://www.cacr.math.uwaterloo.ca/>

5. Bailey, D.V., Paar, C.: Optimal Extension Fields for Fast Arithmetic in Public-Key Algorithms. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 472–485. Springer, Heidelberg (1998)
6. Cantor, D.: Computing in Jacobian of a Hyperelliptic Curve. *Mathematics of Computation* 48(177), 95–101 (1987)
7. Ciet, M., Joye, M., Lauter, K., Montgomery, L.: Trading Inversions for Multiplications in Elliptic Curve Cryptography. *Design, Codes and Cryptography* 39, 189–206 (2006)
8. Cohen, H.: *A Course in Computational Algebraic Number Theory*. Graduate Texts in Math., vol. 138. Springer, Berlin (1993) (fourth corrected printing, 2000)
9. Hankerson, D., Menezes, A., Vanstone, S.: *Guide to Elliptic Curve Cryptography*. Springer, New York (2004)
10. Eisenträger, K., Lauter, K., Montgomery, P.L.: Fast Elliptic Arithmetic and Improved Weil Pairing Evaluation. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 343–354. Springer, Heidelberg (2003)
11. Gaudry, P., Harley, R.: Counting Points on Hyperelliptic Curves over Finite Fields. In: Bosma, W. (ed.) *Algorithmic Number Theory*. LNCS, vol. 1838, pp. 297–312. Springer, Heidelberg (2000)
12. Gonda, M., Matsuo, K., Aoki, K., Chao, J., Tsujii, S.: Improvements of Addition Algorithm on Genus 3 Hyperelliptic Curves and Their Implementation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science* E88-A(1), 89–96 (2005)
13. Gordon, D.M.: A Survey of Fast Exponentiation Methods. *Journal of Algorithms* 27(1), 129–146 (1998)
14. Kobitz, N.: A Family of Jacobian Suitable for Discrete Log Cryptosystems. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 94–99. Springer, Heidelberg (1990)
15. Kobitz, N.: Hyperelliptic Cryptosystems. *Journal of Cryptology* 1(3), 129–150 (1989)
16. Lange, T.: Formulae for Arithmetic on Genus 2 Hyperelliptic Curves. *Applicable Algebra in Engineering, Communication and Computing* 15(5), 295–328 (2005)
17. Lange, T., Stevens, M.: Efficient Doubling for Genus Two Curves over Binary Fields. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, pp. 170–181. Springer, Heidelberg (2004)
18. Menezes, A., Wu, Y., Zuccherato, R.: An Elementary Introduction to Hyperelliptic Curve, Centre for Applied Cryptographic Research (CACR) Technical Reports, CORR 1996-19, available at <http://www.cacr.math.uwaterloo.ca/>
19. Mumford, D.: *Tata Lectures on Theta II*. Prog. Math. 43, Birkhäuser (1984)
20. Nagao, K.: Improving Group Law Algorithms for Jacobians of Hyperelliptic Curves. In: Bosma, W. (ed.) *Algorithmic Number Theory*. LNCS, vol. 1838, pp. 439–448. Springer, Heidelberg (2000)
21. Wollinger, T.: *Software and Hardware Implementation of Hyperelliptic Curve Cryptosystems*, PhD. thesis, Department of Electrical Engineering and Information Sciences, Ruhr-Universität Bochum, Bochum, Germany (2004)
22. Wollinger, T., Pelzl, J., Paar, C.: Cantor versus Harley: Optimization and Analysis of Explicit Formulae for Hyperelliptic Curve Cryptosystems. *IEEE Transactions on Computers* 54(7), 861–872 (2005)

Appendix: Explicit Formulae in Exceptional Cases

In this appendix, we give the explicit addition formulae for the exceptional cases during the computation procedure of $2D_1 + D_2$, which have been discussed in detail in subsection 3.1. These cases usually appear with a very low probability and therefore have not important influence on the performance of genus 2 HECC. Tables 5 to 9 list the detailed steps and the corresponding cost of the group addition in the exceptional cases. In Tables 5 to 9, $\text{ADD}^{i+j \rightarrow k}$ denotes the divisor class addition $D_3 = [U_3, V_3] = D_1 + D_2 = [U_1, V_1] + [U_2, V_2]$, and $\text{TRI}^{i \rightarrow k}$ denotes the divisor class tripling $D_3 = [U_3, V_3] = 3D_1 = 3[U_1, V_1]$, where i, j and k are the degrees of U_1, U_2 and U_3 , respectively.

Table 5. Explicit Formula for $3D_1$ on a HEC of Genus 2 over \mathbb{F}_p : $\text{TRI}^{1 \rightarrow 2}$

Input	Genus 2 HEC $C : Y^2 = F(X), F = X^5 + f_3 X^3 + f_2 X^2 + f_1 X + f_0$; Reduced Divisors $D_1 = (U_1, V_1), U_1 = X + u_{10}, V_1 = v_{10}$,	
Output	Reduced Divisor $D_3 = (U_3, V_3) = 3D_1$, $U_3 = X^2 + u_{31}X + u_{30}, V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute $V_2 = v_{21}X + v_{20}$ (See Equation (1)): $\tilde{u}_{10} = u_{10}^2, \tilde{v}_{10} = v_{10}^2, t_1 = 5\tilde{u}_{10}, t_2 = t_1 + 3f_3, t_3 = u_{10}t_2$; $t_4 = t_3 - 2f_2, t_5 = u_{10}t_4, t_6 = t_5 + f_1, t_7 = (2v_{10})^{-1}$; $v_{21} = t_6t_7, v_{20} = u_{10}v_{21} + v_{10}$;	$1I, 4M, 2S$
2	Compute $d_1 = \text{gcd}(U_1, U_2) = X + u_{10} = e_1U_1 + e_2U_2$: $e_1 = 1, e_2 = 0$;	-
3	Compute $d = \text{gcd}(d_1, V_1 + V_2) = 1 = c_1d_1 + c_2(V_1 + V_2)$: $s_1 = c_1e_1 = c_1, s_2 = c_2e_2 = 0, s_3 = c_2 = t_7$;	-
4	Compute $U' = U_1^3 d^{-2} = (X + u_{10})^3$:	-
5	Compute $v' = v_2'X^2 + v_1'X + v_0' \equiv [s_1U_1V_2 + s_3(V_1V_2 + F)]d^{-1} \text{ mod } U'$: $\tilde{v}_{21} = v_{21}^2, v_2' = t_7(f_2 - \tilde{v}_{21} - u_{10}(t_1 + t_2))$; $v_1' = v_{21} + 2u_{10}v_2', v_0' = v_{20} + \tilde{u}_{10}v_2'$;	$4M, 1S$
6	Compute $U_3 = X^2 + u_{31}X + u_{30} = (F - V'^2)/U'$: $u_{31} = -(v_2'^2 + 3u_{10}), u_{30} = f_3 + t_1 + \tilde{u}_{10} + v_2'(3u_{10}v_2' - 2v_1')$;	$2M, 1S$
7	Compute $V_3 = v_{31}X + v_{30} = -V' \text{ mod } U_3$: $v_{31} = u_{31}v_2' - v_1', v_{30} = u_{30}v_2' - v_0'$;	$2M$
Sum		$1I, 12M, 4S$

Table 6. Explicit Formula for $D_1 + D_2$ on a HEC of Genus 2 over \mathbb{F}_p : $\text{ADD}^{1+2 \rightarrow 2}$

Input	Genus 2 HEC $C : Y^2 = F(X), F = X^5 + f_3X^3 + f_2X^2 + f_1X + f_0$; Reduced Divisors $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$, $U_1 = X + u_{10}, V_1 = v_{10}, U_2 = (X + u_{10})(X + u_{20}), V_2 = v_{21}X + v_{20}$	
Output	Reduced Divisor $D_3 = (U_3, V_3) = D_1 + D_2$, $U_3 = X^2 + u_{31}X + u_{30}, V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute $V_2 = v_{21}X + v_{20}$ (See Equation (2)): $t_1 = u_{10} - u_{20}, t_2 = v_{20} - v_{10}, t_3 = 2v_{10}, t_4 = v_{20}u_{10}, t_5 = v_{10}u_{20}$; $t_6 = (t_1t_3)^{-1}, t_7 = t_3t_6, t_8 = t_1t_6, v_{21} = t_2t_7, v_{20} = (t_4 - t_5)t_7$;	1I, 7M
2	Compute $d_1 = \text{gcd}(U_1, U_2) = X + u_{10} = e_1U_1 + e_2U_2$: $e_1 = 1, e_2 = 0$;	-
3	Compute $d = \text{gcd}(d_1, V_1 + V_2) = 1 = c_1d_1 + c_2(V_1 + V_2)$: $s_1 = c_1e_1 = c_1, s_2 = c_2e_2 = 0, s_3 = c_2 = t_8$;	-
4	Compute $U' = U_1U_2d^{-2} = (X + u_{10})^2(X + u_{20})$:	-
5	Compute $v' = v_2X^2 + v_1X + v_0 \equiv [s_1U_1V_2 + s_3(V_1V_2 + F)]d^{-1} \pmod{U'}$: $\tilde{u}_{10} = u_{10}^2, \tilde{u}_{20} = u_{20}^2, \tilde{v}_{21} = v_{21}^2, w_1 = u_{10} + u_{20}, w_2 = u_{10} + w_1$; $w_3 = f_2 - \tilde{v}_{21} - w_2(f_3 + \tilde{u}_{10} + \tilde{u}_{20}) - 2\tilde{u}_{10}w_1, v_2' = w_3t_8$; $v_1' = v_2'w_1 + v_{21}, w_4 = u_{10}u_{20}, v_0' = v_2'w_4 + v_{20}$;	6M, 3S
6	Compute $U_3 = X^2 + u_{31}X + u_{30} = (F - V_2')/U'$: $u_{31} = -(v_2'^2 + w_2), w_5 = u_{10}(w_1 + u_{20}), u_{30} = f_3 - 2v_1'v_2' - w_5(u_{31} + 1)$;	3M, 1S
7	Compute $V_3 = v_{31}X + v_{30} = -V_3' \pmod{U_3}$: $v_{31} = u_{31}v_2' - v_1', v_{30} = u_{30}v_2' - v_0'$;	2M
Sum		1I, 18M, 4S

Table 7. Explicit Formula for $2D_1 + D_2$ on a HEC of Genus 2 over \mathbb{F}_p : $\text{ADD}^{1+2 \rightarrow 2}$

Input	Genus 2 HEC $C : Y^2 = F(X), F = X^5 + f_3X^3 + f_2X^2 + f_1X + f_0$; Reduced Divisors $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$, $U_1 = X + u_{10}, V_1 = v_{10}, U_2 = X^2 + u_{21}X + u_{20}, V_2 = v_{21}X + v_{20}$	
Output	Reduced Divisor $D_3 = (U_3, V_3) = 2D_1 + D_2$, $U_3 = X^2 + u_{31}X + u_{30}, V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute the resultant r of U_1 and U_2: $i_0 = u_{10} - u_{21}, r = i_0u_{10} + u_{20}$;	$1M$
2	Compute the pseudo-inverse $I = -X + i_0 \equiv r/U_1 \pmod{U_2}$:	-
3	Compute $S' = s'_1X + s'_0 = rS \equiv (V_2 - V_1)I \pmod{U_2}$: $w_0 = v_{20} - v_{10}, s'_1 = u_{10}v_{21} - w_0, s'_0 = i_0w_0 + u_{20}v_{21}$;	$3M$
4	Compute $S = s_1X + s_0 = (s'_1/r)X + (s'_0/r)$:	-
5	Compute $U' = (k - S(l + 2V_1))/U_2 = X^2 + u'_1X + u'_0$: $R = r^2, w_0 = u_{10} + u_{21}, w_1 = f_3 + u_{10}^2, u'_1 = -(s_1^2 + Rw_0)$; $u'_0 = R(w_1 - u_{20} + u_{21}w_0) - s_1(s_1i_0 + 2s_0)$;	$5M, 3S$
6	Compute the resultant \tilde{r} of U_1 and U': $\tilde{i}_0 = Ru_{10} - u'_1, \tilde{r} = \tilde{i}_0u_{10} + u'_0$; If $\tilde{r} = 0$ then factor $U' = (X + u_{10})(X + u'_{20})$ and see Table 6	$2M$
7	Compute the pseudo-inverse $\tilde{I} = -X + \tilde{i}_0 \equiv \tilde{r}/U_1 \pmod{U'}$:	-
8	Compute $\tilde{S}' = \tilde{s}'_1X + \tilde{s}'_0 = \tilde{r}\tilde{S} \equiv -S' - 2V_1\tilde{I} \pmod{U'}$: $\tilde{s}'_1 = 2v_{10} - s_1, \tilde{s}'_0 = -(Rs_0 + 2v_{10}\tilde{i}_0)$;	$2M$
9	Compute $\tilde{S} = \tilde{s}_1X + \tilde{s}_0$: $\tilde{w} = (\tilde{r}R)^{-1}, t_1 = R\tilde{w}, t_2 = \tilde{r}\tilde{w}, \tilde{s}_1 = R\tilde{s}'_1t_1, \tilde{s}_0 = \tilde{s}'_0t_1$;	$1I, 6M$
10	Adjust: $u'_1 = u_1t_2, u'_0 = u'_0t_2, \tilde{i}_0 = \tilde{i}_0t_2$;	$3M$
11	Compute $U_3 = (k - \tilde{S}(\tilde{l} + 2V_1))/U' = X^2 + u_{31}X + u_{30}$: $\tilde{w}_0 = u_{10} + u'_1, u_{31} = -(\tilde{s}_1^2 + \tilde{w}_0), u_{30} = w_1 - u'_0 + u'_1\tilde{w}_0 - \tilde{s}_1(\tilde{s}_1\tilde{i}_0 + 2\tilde{s}_0)$;	$3M, 1S$
12	Compute $V_3 = -(\tilde{l} + V_1) \pmod{U_3} = v_{31}X + v_{30}$: $v_{31} = \tilde{s}_1(u_{31} - u_{10}) - \tilde{s}_0, v_{30} = \tilde{s}_1u_{30} - \tilde{s}_0u_{10} - v_{10}$;	$3M$
Sum		$I, 28M, 4S$

Table 8. Explicit Formula for $2D_1 + D_2$ on a HEC of Genus 2 over \mathbb{F}_p : $\text{ADD}^{2+1 \rightarrow 2}$

Input	Genus 2 HEC $C : Y^2 = F(X)$, $F = X^5 + f_3X^3 + f_2X^2 + f_1X + f_0$; Reduced Divisors $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$, $U_1 = X^2 + u_{11}X + u_{10}$, $V_1 = v_{11}X + v_{10}$, $U_2 = X + u_{20}$, $V_2 = v_{20}$	
Output	Reduced Divisor $D_3 = (U_3, V_3) = 2D_1 + D_2$, $U_3 = X^2 + u_{31}X + u_{30}$, $V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute the resultant $r = U_1 \bmod U_2$: $r = u_{10} - (u_{11} - u_{20})u_{20}$;	1M
2	Compute the inverse $i \equiv 1/U_1 \bmod U_2$:	–
3	Compute $S = s_0 \equiv (V_2 - V_1)i \bmod U_2$: $s_0 = v_{20} - v_{10} - v_{11}u_{20}$;	1M
4	Compute $K = (F - V_1^2)/U_1 = X^3 + k_2X^2 + k_1X + k_0$: $k_1 = f_3 + u_{11}^2 - u_{10}$;	1S
5	Compute $U' = (k - S(l + 2V_1))/U_2 = X^2 + u'_1X + u'_0$: $R = r^2$, $u'_1 = -(s_0^2 + R(u_{11} + u_{20}))$, $u'_0 = Rk_1 - s_0(s_0u_{11} + 2rv_{11}) - u_{20}u'_1$;	6M, 2S
6	Compute the resultant \tilde{r} of U_1 and U': $\tilde{i}_1 = u'_1 - Ru_{11}$, $\tilde{w} = Ru_{10} - u'_0$, $\tilde{i}_0 = \tilde{i}_1u'_1 + R\tilde{w}$, $\tilde{r} = \tilde{i}_0\tilde{w} + \tilde{i}_1^2u'_0$; If $\tilde{r} = 0$ then see Table	6M, 1S
7	Compute the pseudo-inverse $\tilde{I} = \tilde{i}_1X + \tilde{i}_0 \equiv \tilde{r}/U_1 \bmod U'$:	–
8	Compute $\tilde{S}' = \tilde{s}'_1X + \tilde{s}'_0 = \tilde{r}\tilde{S} \equiv -S - 2V_1\tilde{I} \bmod U'$: $\tilde{w}_0 = \tilde{i}_0v_{10}$, $\tilde{w}_1 = \tilde{i}_1v_{11}$, $\tilde{s}'_0 = -(Rrs_0 + 2(\tilde{w}_0 - u'_0\tilde{w}_1))$; $\tilde{s}'_1 = -2((\tilde{i}_0 + R\tilde{i}_1)(v_{10} + v_{11}) - \tilde{w}_0 - \tilde{w}_1(R + u'_1))$; If $\tilde{s}'_1 = 0$ See Below	8M
9	Compute $\tilde{S}'' = X + \tilde{s}_0/\tilde{s}'_1 = X + \tilde{s}'_0/\tilde{s}'_1$ and \tilde{s}_1: $\tilde{w}_0 = \tilde{r}\tilde{s}'_1$, $\tilde{w}_1 = (R\tilde{w}_0)^{-1}$, $R_1 = \tilde{w}_0\tilde{w}_1$, $R_2 = R_1^2$, $R_3 = R_1R_2$, $\tilde{w}_2 = \tilde{r}\tilde{w}_1R_2$; $\tilde{w}_3 = \tilde{s}'_1{}^2\tilde{w}_1R_3$, $\tilde{w}_4 = \tilde{r}\tilde{w}_2R_3$, $\tilde{w}_5 = \tilde{w}_4^2$, $\tilde{s}''_0 = \tilde{s}'_0\tilde{w}_2R_2$;	1I, 12M, 3S
10	Adjust: $u'_1 = u'_1R_1$, $u'_0 = u'_0R_1$, $\tilde{i}_1 = \tilde{i}_1R_1$;	3M
11	Compute $\tilde{l}' = \tilde{S}''u_1 = X^3 + \tilde{l}'_2X^2 + \tilde{l}'_1X + \tilde{l}'_0$: $\tilde{l}'_2 = u_{11} + \tilde{s}''_0$, $\tilde{l}'_1 = u_{11}\tilde{s}''_0 + u_{10}$, $\tilde{l}'_0 = u_{10}\tilde{s}''_0$;	2M
12	Compute $U_3 = (\tilde{s}(\tilde{l}' + 2V_1) - k)/U' = X^2 + u_{31}X + u_{30}$: $u_{30} = (\tilde{s}''_0 - u'_1)(\tilde{s}''_0 - \tilde{i}_1) - u'_0 + \tilde{l}'_1 + 2v_{11}\tilde{w}_4 + (u'_1 + u_{11})\tilde{w}_5$; $u_{31} = 2\tilde{s}''_0 - \tilde{i}_1 - \tilde{w}_5$;	3M
13	Compute $V_3 = -(\tilde{l}' + V_1) \bmod U_3 = v_{31}X + v_{30}$: $w_1 = \tilde{l}'_2 - u_{31}$, $w_2 = u_{31}w_1 + u_{30} - \tilde{l}'_1$, $v_{31} = w_2\tilde{w}_3 - v_{11}$; $w_2 = u_{30}w_1 - \tilde{l}'_0$, $v_{30} = w_2\tilde{w}_3 - v_{10}$;	4M
Sum	$\tilde{s}'_1 \neq 0$	I, 46M, 7S
9'	Compute \tilde{s}_0: $\tilde{w}_1 = (\tilde{r}R)^{-1}$, $t_1 = \tilde{r}\tilde{w}_1$, $t_2 = R\tilde{w}_1$, $\tilde{s}_0 = R\tilde{s}'_0t_2$;	1I, 5M
10'	Adjust: $u'_1 = u'_1t_1$, $u'_0 = u'_0t_1$;	2M
11'	Compute $U_3 = (k - \tilde{s}(\tilde{l}' + 2V_1))/U' = X + u_{30}$: $u_{30} = -(u'_1 + u_{11} + \tilde{s}_0^2)$;	1S
12'	Compute $V_3 = -(\tilde{l}' + V_1) \bmod U_3 = v_{30}$: $w_1 = \tilde{s}_0(u'_1 + u_{30}) + v_{11}$, $w_2 = \tilde{s}_0 + v_{10}$, $v_{30} = u'_0w_1 - w_2$;	2M
Sum	$\tilde{s}'_1 = 0$	I, 31M, 5S

Table 9. Explicit Formula for $D_1 + D_2$ on a HEC of Genus 2 over \mathbb{F}_p : $\text{ADD}^{2+2 \rightarrow 2}$

Input	Genus 2 HEC $C : Y^2 = F(X), F = X^5 + f_3X^3 + f_2X^2 + f_1X + f_0$; Reduced Divisors $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$, $U_1 = X^2 + u_{11}X + u_{10} = (X + u_{p1})(X + u_{p2}), V_1 = v_{11}X + v_{10}$, $U_2 = X^2 + u_{21}X + u_{20} = (X + u_{p1})(X + u_{p3}), V_2 = v_{21}X + v_{20}$;	
Output	Reduced Divisor $D_3 = (U_3, V_3) = D_1 + D_2$, $U_3 = X^2 + u_{31}X + u_{30}, V_3 = v_{31}X + v_{30}$;	
Step	Expression	Cost
1	Compute $d_1 = \gcd(U_1, U_2) = X + u_{p1} = e_1U_1 + e_2U_2$: $e_1 = 1, e_2 = 0$;	-
2	Compute $d = \gcd(d_1, V_1 + V_2) = c_1d_1 + c_2(V_1 + V_2)$: If $d = X + u_{10}$ then see below, else $d = 1$ and we have $s_1 = c_1e_1 = c_1, s_2 = c_2e_2 = 0, s_3 = c_2$;	-
3	Compute $U' = U_1U_2d^{-2} = X^4 + u_3X^3 + u_2X^2 + u_1X + u_0$: $u_3 = u_{11} + u_{21}, t_0 = u_{11}u_{21}, u_2 = u_{10} + u_{20} + t_0, u_0 = u_{10}u_{20}$; $u_1 = (u_{11} + u_{10})(u_{20} + u_{21}) - t_0 - u_0$;	3M
4	Compute $V' = v_3X^3 + v_2X^2 + v_1X + v_0 \equiv [s_1U_1V_2 + s_3(V_1V_2 + F)]d^{-1} \pmod{U'}$: $t_1 = v_{11} + v_{21}, t_2 = u_{11}v_{21}, t_3 = u_{21}v_{20}, t_4 = v_{11}v_{21}, t_5 = v_{10}v_{20}$; $t_6 = (v_{10} + v_{11})(v_{20} + v_{21}) - t_4 - t_5, t_7 = (u_{11} + u_{21})(v_{20} + v_{21}) - t_2 - t_3$; $t_8 = v_{10} + v_{20} - t_1u_{p1}, t_9 = f_3 - t_1v_{21} - u_2 + u_3, t_{10} = (t_8t_9)^{-1}$; $c_2 = t_9t_{10}, v_3 = c_2t_9, v_2 = c_2(f_2 + t_4 - t_1(t_2 + v_{20}) - u_1 + u_2u_3)$; $v_1 = c_2(f_1 + t_6 - t_1t_7 - u_0 + u_1u_3), v_0 = c_2(f_0 + t_5 - t_1t_3 + u_0u_3)$;	1I, 20M, 1S
5	Compute $U_3 = X^2 + u_{31}X + u_{30} = (V'^2 - F)/U'$: $t_1 = t_8^2t_{10}, u_{31} = t_1(2v_2 - t_1) - u_3, u_{30} = (v_2t_1)^2 + 2v_1t_1 - u_2 - u_3u_{31}$;	5M, 2S
6	Compute $V_3 = v_{31}X + v_{30} = -V' \pmod{U_3}$: $t_2 = u_{31}v_3 - v_2, v_{31} = u_{30}v_3 - v_1 - u_{31}t_2, v_{30} = -(u_{30}t_2 + v_0)$;	4M
Sum	$d = 1$	1I, 32M, 3S
3'	Compute $U_3 = U_1U_2d^{-2} = (X + u_{p2})(X + u_{p3})$: $u_{31} = u_{p2} + u_{p3}, u_{30} = u_{p2}u_{p3}$;	1M
4'	Compute $V' = v_{31}X + v_{30}$: $t_0 = (u_{p2} - u_{p3})^{-1}, t_1 = v_{11}u_{p2} + v_{10}, t_2 = v_{21}u_{p3} + v_{20}$; $t_3 = t_2 - t_1, v_{31} = t_0t_3, t_4 = t_2u_{p2} - t_1u_{p3}, v_{30} = t_0t_4$;	1I, 6M
Sum	$d = X + u_{10}$	I, 7M