Efficient Pairing Computation on Genus 2 Curves in Projective Coordinates

Xinxin Fan[1,⋆], Guang Gong[1,⋆], and David Jao[2,⋆⋆]

[1] Department of Electrical and Computer Engineering
[2] Department of Combinatorics and Optimization,
University of Waterloo,
Waterloo, Ontario, N2L 3G1, Canada
{x5fan@engmail,ggong@calliope,djao@math}.uwaterloo.ca

Abstract. In recent years there has been much interest in the development and the fast computation of bilinear pairings due to their practical and myriad applications in cryptography. Well known efficient examples are the Weil and Tate pairings and their variants such as the Eta and Ate pairings on the Jacobians of (hyper-)elliptic curves. In this paper, we consider the use of projective coordinates for pairing computations on genus 2 hyperelliptic curves over prime fields. We generalize Chatterjee *et. al.*'s idea of encapsulating the computation of the line function with the group operations to genus 2 hyperelliptic curves, and derive new explicit formulae for the group operations in projective and new coordinates in the context of pairing computations. When applying the encapsulated explicit formulae to pairing computations on supersingular genus 2 curves over prime fields, theoretical analysis shows that our algorithm is faster than previously best known algorithms whenever a field inversion is more expensive than about fifteen field multiplications. We also investigate pairing computations on non-supersingular genus 2 curves over prime fields based on the new formulae, and detail the various techniques required for efficient implementation.

Keywords: Genus 2 hyperelliptic curves, Tate pairing, Miller's algorithm, Projective coordinates, Efficient Implementation.

1 Introduction

Bilinear pairings were first introduced to cryptography by Menezes *et. al.* [24] and Frey and Rück [13] as a tool to attack instances of the discrete logarithm problem (DLP) on (hyper-)elliptic curves. Subsequently, Sakai *et. al.*'s non-interactive key distribution scheme [29] and Joux's tripartite Diffie-Hellman key agreement protocol [19] provided examples of positive usages of pairings. This use of pairings has inspired much research devoted to the design of cryptographic protocols with novel properties and the improvement of existing ones, with some classical

---

examples being Identity Based Encryption [4] and short signatures [5]. Since pairing computations are generally the most important and expensive operation in any pairing-based cryptosystem, improving the speed of pairing computations has become an important issue in pairing-based cryptography.

Miller proposed the first algorithm [26] for computing the Weil pairing on elliptic curves. In practice, the Tate pairing shows better performance than that of the Weil pairing and therefore is widely used. While many important techniques have been proposed to accelerate the computation of the Tate pairing and its variants on elliptic curves [2,3,17], the subject of pairing computations on hyperelliptic curves is also receiving an increasing amount of attention. Choie and Lee [7] investigated the implementation of the Tate pairing on supersingular genus 2 hyperelliptic curves over prime fields. Later on, Ó hÉigeartaigh and Scott [16] improved the implementation of [7] significantly by using a new variant of Miller's algorithm combined with various optimization techniques. Duursma and Lee [9] presented a closed formula for the Tate pairing computation on a very special family of supersingular hyperelliptic curves. Barreto *et. al.* [2] generalized the results of [9] and proposed the Eta pairing approach for efficiently computing the Tate pairing on supersingular genus 2 curves over binary fields. In particular, their algorithm leads to the fastest pairing implementation in the literature. In [23], Lee *et. al.* considered the Eta pairing computation on general divisors on supersingular genus 3 hyperelliptic curves with the form of $y^2 = x^7 - x \pm 1$. Recently, the Ate pairing, which is an extension of the Eta pairing to the setting of ordinary curves, has been generalized to hyperelliptic curves [14] as well. Although the Eta and Ate pairings hold the record for speed at the present time, we will focus our attention on the Tate pairing in this paper. The main reason is that the Tate pairing is uniformly available across a wide range of hyperelliptic curves and subgroups, whereas the Eta pairing is only defined for supersingular curves and the Ate pairing incurs a huge performance penalty in the context of ordinary genus 2 curves [14, Table 6].

Previous work for computing pairings on hyperelliptic curves only considered using affine coordinates. Motivated by Chatterjee *et. al.*'s work [6], we address the efficient implementation of the Tate pairing on genus 2 hyperelliptic curves over large prime fields in projective coordinates in this contribution. We first derive new explicit formulae for the group operations for genus 2 hyperelliptic curves in projective and new (weighted projective) coordinates, respectively. Letting I denote a field inversion, M a field multiplication, and S a field squaring, we find in the context of pairing computations that compared to Lange's formulae [22], our mixed-addition formulae can save $5M$ and $3M$ in projective and new coordinates, respectively, whereas our doubling formulae can save $2M$ in both projective and new coordinates. We then show how to encapsulate the computation of the line function with the mixed addition and doubling formulae in new coordinates, and how to omit some operations which are cancelled by the final exponentiation in the encapsulated method. Our encapsulated explicit formulae can be applied to pairing computations on both supersingular and non-supersingular genus 2 hyperelliptic curves over prime fields. Finally, we

describe an efficient implementation of the Tate pairing on a non-supersingular genus 2 hyperelliptic curve with an embedding degree of 2 over prime fields as a case study. To our knowledge, this is the first concrete implementation of pairing computations on non-supersingular genus 2 curves.

This paper is organized as follows. Section 2 gives an overview of the Tate pairing on hyperelliptic curves and Miller's algorithm for computing the pairing. In Section 3 we describe new explicit formulae which encapsulate group operations and line computations for genus 2 curves over prime fields. Section 4 shows how to apply various techniques from the literature to accelerate the pairing computation on a specific non-supersingular genus 2 curve over prime fields, analyzes the computational complexity of computing the Tate pairings and gives implementation results. Finally, Section 5 concludes this contribution.

2   Mathematical Background

2.1   Tate Pairing on Hyperelliptic Curves

Let $\mathbb{F}_q$ be a finite field with q elements, and $\overline{\mathbb{F}}_q$ be its algebraic closure. Let C be a hyperelliptic curve of genus g over $\mathbb{F}_q$, and let $\mathcal{J}_C$ denote the degree zero divisor class group of C. We say that a subgroup of the divisor class group $\mathcal{J}_C(\mathbb{F}_q)$ has *embedding degree* k if the order n of the subgroup divides $q^k - 1$, but does not divide $q^i - 1$ for any $0 < i < k$. For our purpose, n should be a (large) prime with $n \mid \#\mathcal{J}_C(\mathbb{F}_q)$ and $\gcd(n, q) = 1$. Let $\mathcal{J}_C(\mathbb{F}_{q^k})[n]$ be the n-torsion group and $\mathcal{J}_C(\mathbb{F}_{q^k})/n\mathcal{J}_C(\mathbb{F}_{q^k})$ be the quotient group. Then the Tate pairing is a well defined, non-degenerate, bilinear map [13]:

$$\langle \cdot, \cdot \rangle_n : \mathcal{J}_C(\mathbb{F}_{q^k})[n] \times \mathcal{J}_C(\mathbb{F}_{q^k})/n\mathcal{J}_C(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}^*/(\mathbb{F}_{q^k}^*)^n,$$

defined as follows: let $D_1 \in \mathcal{J}_C(\mathbb{F}_{q^k})[n]$, with $\mathrm{div}(f_{n,D_1}) = nD_1$ for some rational function $f_{n,D_1} \in \mathbb{F}_{q^k}(C)^*$. Let $D_2 \in \mathcal{J}_C(\mathbb{F}_{q^k})/n\mathcal{J}_C(\mathbb{F}_{q^k})$ with $\mathrm{supp}(D_1) \cap \mathrm{supp}(D_2) = \emptyset$ (to ensure a non-trivial pairing value). The Tate pairing of two divisor classes $\overline{D}_1$ and $\overline{D}_2$ is then defined as

$$\langle \overline{D}_1, \overline{D}_2 \rangle_n = f_{n,D_1}(D_2) = \prod_{P \in C(\overline{\mathbb{F}}_q)} f_{n,D_1}(P)^{\mathrm{ord}_P(D_2)}.$$

Note that the Tate pairing as detailed above is only defined up to n-th powers. One can show that if the function f_{n,D_1} is properly normalized, we only need to evaluate the rational function f_{n,D_1} at the effective part of the reduced divisor D_2 in order to compute the Tate pairing [3,14].

In practice, the fact that the Tate pairing is only defined up to n-th power is usually undesirable, and many pairing-based protocols require a unique pairing value. Hence one defines the *reduced* pairing as

$$\langle \overline{D}_1, \overline{D}_2 \rangle_n^{(q^k-1)/n} = f_{n,D_1}(D_2)^{(q^k-1)/n} \in \mu_n \subset \mathbb{F}_{q^k}^*,$$

where $\mu_n = \{u \in \mathbb{F}_{q^k}^* \mid u^n = 1\}$ is the group of n-th roots of unity. In the rest of this paper we will refer to the extra powering required to compute the reduced

pairing as the *final exponentiation*. Furthermore, we also assume the embedding degree k is greater than 1.

2.2 Miller's Algorithm

The main task involved in the computation of the Tate pairing $\langle \overline{D}_1, \overline{D}_2 \rangle_n$ is to construct a rational function f_{n,D_1} such that $\mathrm{div}(f_{n,D_1}) = nD_1$. In [26], Miller described a polynomial time algorithm, known universally as Miller's algorithm, to construct the function f_{n,D_1} and compute the Weil pairing on elliptic curves. However, the algorithm can be easily adapted to compute the Tate pairing on hyperelliptic curves.

Let $G_{iD_1, jD_1} \in \mathbb{F}_{q^k}(C)^*$ be a rational function with $\mathrm{div}(G_{iD_1, jD_1}) = iD_1 + jD_1 - (iD_1 \oplus jD_1)$ where $\oplus$ is the group law on $\mathcal{J}_C$ and $(iD_1 \oplus jD_1)$ is reduced. Miller's algorithm constructs the rational function f_{n,D_1} based on the following iterative formula:

$$f_{i+j, D_1} = f_{i,D_1} f_{j,D_1} G_{iD_1, jD_1}.$$

Algorithm 1 shows the basic version of Miller's algorithm for computing the reduced Tate pairing on hyperelliptic curves according to the above iterative relation. Essentially, computing the Tate pairing with Miller's algorithm amounts to performing a scalar multiplication of a reduced divisor and evaluating certain intermediate rational functions which appear in the process of the divisor class addition. A more detailed version of Miller's algorithm for hyperelliptic curves can be found in [14].

Algorithm 1. Miller's Algorithm for Hyperelliptic Curves (basic version)

IN: $\overline{D}_1 \in \mathcal{J}_C(\mathbb{F}_{q^k})[n], \overline{D}_2 \in \mathcal{J}_C(\mathbb{F}_{q^k})$, represented by D_1 and D_2
  with $\mathrm{supp}(D_1) \cap \mathrm{supp}(D_2) = \emptyset$
OUT: $\langle D_1, D_2 \rangle_n^{(q^k-1)/n}$
1. $f \leftarrow 1, T \leftarrow D_1$
2. **for** $i \leftarrow \lfloor \log_2(n) \rfloor - 1$ **downto 0 do**
3.   ▷ Compute T' and $G_{T,T}(x,y)$ such that $T' = 2T - \mathrm{div}(G_{T,T})$
4. $f \leftarrow f^2 \cdot G_{T,T}(D_2), \overline{T} \leftarrow [2]\overline{T}$
5. **if** $n_i = 1$ **then**
6.   ▷ Compute T' and $G_{T,D_1}(x,y)$ such that $T' = T + D_1 - \mathrm{div}(G_{T,D_1})$
7. $f \leftarrow f \cdot G_{T,D_1}(D_2), \overline{T} \leftarrow \overline{T} \oplus \overline{D}_1$
8. **Return** $f^{(q^k-1)/n}$

---

3   Encapsulated Computation on Genus 2 Curves

In this section we generalize the idea of encapsulated add-and-line and encapsulated double-and-line proposed in [6] to genus 2 hyperelliptic curves over large prime fields. Note that, in the process of computing Tate pairings, one inversion is required for each divisor class addition and doubling, and the calculation of the inversion of an element in large characteristic is usually quite expensive.

Therefore, to avoid inversions, we need to derive efficient inversion-free explicit formulae for genus 2 hyperelliptic curves in the context of pairing computations.

Lange [22] presented efficient explicit formulae for the group operations on genus 2 curves using various systems of coordinates. In the projective coordinate system, the quintuple $[U_1, U_0, V_1, V_0, Z]$ corresponds to the affine class $[x^2 + U_1/Zx + U_0/Z, V_1/Zx + V_0/Z]$ in Mumford representation [28], whereas the sextuple $[U_1, U_0, V_1, V_0, Z_1, Z_2]$ stands for the affine class $[x^2 + U_1/Z_1^2 x + U_0/Z_1^2, V_1/(Z_1^3 Z_2)x + V_0/(Z_1^3 Z_2)]$ in the new coordinate system. Lange's formulae are designed to be used in the context of computing scalar multiplications, and do not explicitly calculate all of the rational functions required in Miller's algorithm. However, one can extract the rational functions required from the formulae in [22] at the cost of 3 extra field multiplications.

Choie and Lee [7] modified Lange's explicit formulae in affine coordinates to reduce the cost of extracting the rational functions required in Miller's algorithm. The formulae presented in [7] require $1I + 23M + 3S$ and $1I + 23M + 5S$ in $\mathbb{F}_p$ for divisor class addition[1] and doubling, respectively, thereby saving 2 field multiplications over the previous method. Ó hÉigeartaigh and Scott [16] further optimized the doubling formula proposed in [7] for supersingular genus 2 curves over $\mathbb{F}_p$ of the form $y^2 = x^5 + a$ by saving 1 multiplication and 1 squaring.

Based on the above explicit formulae in affine coordinates, we derive new explicit mixed-addition and doubling formulae in the projective and new coordinate systems in the context of pairing computations, respectively. Since the explicit formulae in new coordinates are more efficient than those in projective coordinates, we use new coordinates to represent divisor classes in the main presentation. The mixed-addition and doubling formulae in projective coordinates can be found in the appendix. We will explain how to encapsulate the group operations and the line computations in the following subsections. To increase performance, we also enlarge the set of coordinates to $[U_1, U_0, V_1, V_0, Z_1, Z_2, z_1, z_2]$ as in [22], where $z_1 = Z_1^2$ and $z_2 = Z_2^2$.

3.1   Encapsulated Divisor Addition and Line Computation

In this subsection, we show how to encapsulate the computation of the line function with the divisor class addition in new coordinates. Given two divisor classes $\overline{E}_1 = [U_{11}, U_{10}, V_{11}, V_{10}, 1, 1, 1, 1]$ and $\overline{E}_2 = [U_{21}, U_{20}, V_{21}, V_{20}, Z_{21}, Z_{22}, z_{21}, z_{22}]$ in new coordinates as inputs, Table 1 describes an explicit mixed-addition formula which calculates a divisor class $\overline{E}_3 = [u_3(x), v_3(x)]$ and the rational function $l(x)$ such that $E_1 + E_2 = E_3 + \mathrm{div}\left(\frac{y - l(x)}{u_3(x)}\right)$ in the most common case. Our new explicit formula requires $36M + 5S$ for computing the divisor class addition in new coordinates. Table 2 summarizes the computational cost of calculating the divisor class addition and extracting the line function in various coordinate systems. From Table 2 we note that in the context of pairing computations our mixed-addition formulae can save $5M$ in the projective coordinate system and

---

[1] We note that the addition formula in [7] requires $3S$ instead of $2S$ as claimed. Indeed, each of Steps 1, 4, and 6 in [7, Table 5] requires a separate squaring.

Table 1. Mixed-Addition Formula on a Genus 2 Curve over $\mathbb{F}_p$ (New Coordinates)

Input	Genus 2 HEC $C : y^2 = x^5 + f_3 x^3 + f_2 x^2 + f_1 x + f_0$	
	$\overline{E}_1 = [U_{11}, U_{10}, V_{11}, V_{10}, 1, 1, 1, 1]$ and	
	$\overline{E}_2 = [U_{21}, U_{20}, V_{21}, V_{20}, Z_{21}, Z_{22}, z_{21}, z_{22}]$	
Output	$\overline{E}_3 = [U_{31}, U_{30}, V_{31}, V_{30}, Z_{31}, Z_{32}, z_{31}, z_{32}] = \overline{E}_1 \oplus \overline{E}_2$	
	$l(x)$ such that $E_1 + E_2 = E_3 + \text{div}\left(\frac{y - l(x)}{u_3(x)}\right)$	
Step	Expression	Cost
1	**Compute resultant and precomputations:**	$7M, 1S$
	$z_{23} = Z_{21}Z_{22}, z_{24} = z_{21}z_{23}, \tilde{U}_{11} = U_{11}z_{21}, y_1 = \tilde{U}_{11} - U_{21}$	
	$y_2 = U_{20} - U_{10}z_{21}, y_3 = U_{11}y_1, y_4 = y_2 + y_3, r = y_2 y_4 + y_1^2 U_{10}$	
2	**Compute almost inverse of u_2 mod u_1:**	–
	$inv_1 = y_1, inv_0 = y_4$	
3	**Compute s':**	$7M$
	$w_0 = V_{10}z_{24} - V_{20}, w_1 = V_{11}z_{24} - V_{21}, w_2 = inv_0 w_0$	
	$w_3 = inv_1 w_1, s_1' = y_1 w_0 + y_2 w_1, s_0' = w_2 - U_{10}w_3$	
4	**Precomputations:**	$4M, 3S$
	$\tilde{r} = rz_{23}, R = \tilde{r}^2, Z_{31} = s_1' Z_{21}, Z_{32} = \tilde{r}Z_{21}$	
	$z_{31} = Z_{31}^2, z_{32} = Z_{32}^2, \tilde{s}_0' = s_0' z_{21}$	
5	**Compute l:**	$5M$
	$l_2 = s_1' U_{21} + \tilde{s}_0', l_0 = s_0' U_{20} + rV_{20}$	
	$l_1 = (s_1' + s_0')(U_{21} + U_{20}) - s_1' U_{21} - s_0' U_{20} + rV_{21}$	
6	**Compute U_3:**	$7M, 1S$
	$w_1 = \tilde{U}_{11} + U_{21}, U_{31} = s_1'(2\tilde{s}_0' - s_1' y_1) - z_{32}, l_1' = l_1 s_1'$	
	$U_{30} = \tilde{s}_0'(s_0' - 2s_1' U_{11}) + s_1'^2(y_3 - \tilde{U}_{10} - U_{20}) + 2l_1' + Rw_1$	
7	**Compute V_3:**	$6M$
	$w_1 = l_2 s_1' - U_{31}, V_{30} = U_{30}w_1 - z_{31}(l_0 s_1')$	
	$V_{31} = U_{31}w_1 + z_{31}(U_{30} - l_1')$	
Sum		$36M, 5S$

Table 2. Divisor Class Addition in Different Systems and in Odd Characteristic

Reference	Coordinate Type	Addition	Mixed Addition	Extracting Line Function $l(x)$
Miyamoto *et al.* [27]	Affine	$1I, 24M, 2S$	–	no cost
	Projective	$54M$	–	no cost
Lange [22]	Affine	$1I, 22M, 3S$	–	$3M$
	Projective	$47M, 4S$	$40M, 3S$	$3M$
	New	$47M, 7S$	$36M, 5S$	$3M$
Choie and Lee [7]	Affine	$1I, 23M, 3S$	–	no cost
Our work	Projective	–	**$38M, 3S$** **Table 9**	**no cost**
	New	–	**$36M, 5S$** **Table 1**	**no cost**

Table 3. Doubling Formula on a Genus 2 Curve over $\mathbb{F}_p$ (New Coordinates)

Input	Genus 2 HEC $C: y^2 = x^5 + f_3 x^3 + f_2 x^2 + f_1 x + f_0$	
	$\overline{E}_1 = [U_{11}, U_{10}, V_{11}, V_{10}, Z_{11}, Z_{12}, z_{11}, z_{12}]$	
Output	$E_3 = [U_{31}, U_{30}, V_{31}, V_{30}, Z_{31}, Z_{32}, z_{31}, z_{32}] = [2]\overline{E}_1$	
	$l(x)$ such that $2E_1 = E_3 + \text{div}\left(\frac{y - l(x)}{u_3(x)}\right)$	
Step	Expression	Cost
1	**Compute resultant:**	$4M, 2S$
	$w_0 = V_{11}^2, w_1 = U_{11}^2, w_2 = V_{10} z_{11}$	
	$w_3 = w_2 - U_{11} V_{11}, r = U_{10} w_0 + V_{10} w_3$	
2	**Compute almost inverse:**	–
	$inv_1' = -V_{11}, inv_0' = w_3$	
3	**Compute k':**	$7M, 1S$
	$z_{11}' = z_{11}^2, w_3 = f_3 z_{11}' + w_1, \tilde{U}_{10} = U_{10} z_{11}$	
	$k_1' = z_{12}(2(w_1 - \tilde{U}_{10}) + w_3), z_{11}'' = z_{11} z_{11}'$	
	$k_0' = z_{12}(U_{11}(4\tilde{U}_{10} - w_3) + f_2 z_{11}'') - w_0$	
4	**Compute s':**	$5M$
	$w_0 = k_0' inv_0', w_1 = k_1' inv_1', s_1' = w_2 k_1' - V_{11} k_0', s_0' = w_0 - \tilde{U}_{10} w_1$	
5	**Precomputations:**	$8M, 4S$
	$Z_{31} = s_1' z_{11}, z_{31} = Z_{31}^2, w_0 = r z_{11}, w_1 = w_0 Z_{12}$	
	$Z_{32} = 2w_1 Z_{11}, z_{32} = Z_{32}^2, w_2 = w_1^2, R = r Z_{31}$	
	$S_0 = s_0'^2, S = s_0' Z_{31}, s_0 = s_0' s_1', s_1 = s_1' Z_{31}$	
6	**Compute l:**	$6M$
	$l_2 = s_1 U_{11} + s_0 z_{11}, V_{10}' = R V_{10}, l_0 = s_0 U_{10} + 2V_{10}'$	
	$V_{11}' = R V_{11}, l_1 = (s_1 + s_0)(U_{11} + U_{10}) - s_1 U_{11} - s_0 U_{10} + 2V_{11}'$	
7	**Compute U_3:**	$1M$
	$U_{30} = S_0 + 4(V_{11}' + 2w_2 U_{11}), U_{31} = 2S - z_{32}$	
8	**Compute V_3:**	$4M$
	$w_0 = l_2 - U_{31}, w_1 = w_0 U_{30}, w_2 = w_0 U_{31}$	
	$V_{31} = w_2 + z_{31}(U_{30} - l_1), V_{30} = w_1 - z_{31} l_0$	
Sum		$35M, 7S$

$3M$ in the new coordinate system, respectively, when compared to the formulae given by Lange [22].

In the new coordinate system, the rational function $c(x, y) = y - l(x)$ that is required in Miller's algorithm has the following form:

$$c(x, y) = y - \left(\frac{s_1'}{r z_{23}} x^3 + \frac{l_2}{r z_{24}} x^2 + \frac{l_1}{r z_{24}} x + \frac{l_0}{r z_{24}}\right),$$

where $s_1', l_2, l_1, l_0, r, z_{23}$ and $z_{24} = z_{21} z_{23}$ are computed in Table 1. By defining the auxiliary rational function $c'(x, y) = (r z_{24}) c(x, y)$, we obtain

$$c'(x, y) = (r z_{24}) y - ((s_1' z_{21}) x^3 + l_2 x^2 + l_1 x + l_0).$$

Note that the result of evaluating the function $c(x, y)$ at an image divisor D_2 will be raised to the power $(q^k - 1)/n$ $(k > 1)$ in the last step of Miller's algorithm.

Table 4. Divisor Class Doubling in Different Systems and in Odd Characteristic

Reference	Coordinate Type	Doubling	Extracting Line Function $l(x)$
Miyamoto *et al.* [27]	Affine	$1I, 23M, 4S$	no cost
	Projective	$53M$	no cost
Lange [22]	Affine	$1I, 22M, 5S$	$3M$
	Projective	$38M, 6S$	$3M$
	New	$34M, 7S$	$3M$
Choie and Lee [7]	Affine	$1I, 23M, 5S$	no cost
Ó hÉigeartaigh and Scott [16]	Affine	$1I, 22M, 4S$	no cost
Our work	Projective	**$39M, 6S$** **Table 10**	**no cost**
	New	**$35M, 7S$** **Table 3**	**no cost**

For efficiency reasons, the first input to the Tate pairing is usually restricted to the 1-eigenspace of the Frobenius endomorphism on $\mathcal{J}_C[n]$. Therefore, we have the following relation

$$c(D_2)^{(q^k-1)/n} = ((c'(D_2)/(rz_{24}))^{q-1})^{(q^{k-1}+q^{k-2}+\ldots+1)/n} = c'(D_2)^{(q^k-1)/n}.$$

The above relation means that in new coordinates we can work with the rational function $c'(x, y)$ instead of $c(x, y)$ without altering the value of the resulting Tate pairing. For the same reason we also work with the rational function $u_3'(x) = z_{31}x^2 + U_{31}x + U_{30}$ instead of $u_3(x) = x^2 + \frac{U_{31}}{z_{31}}x + \frac{U_{30}}{z_{31}}$ for both divisor addition and divisor doubling.

3.2 Encapsulated Divisor Doubling and Line Computation

In this subsection, we describe how to encapsulate the computation of the line function with the divisor class doubling in new coordinates. Given a divisor class $\overline{E}_1 = [U_{11}, U_{10}, V_{11}, V_{10}, Z_{11}, Z_{12}, z_{11}, z_{12}]$ in new coordinates as an input, Table 3 describes an explicit doubling formula which calculates a divisor class $\overline{E}_3 = [u_3(x), v_3(x)]$ and the rational function $l(x)$ such that $2E_1 = E_3 + \operatorname{div}\left(\frac{y-l(x)}{u_3(x)}\right)$ in the most common case. Our new explicit formula needs $35M + 7S$ to double a divisor class in new coordinates. Table 4 summarizes the computational cost of doubling a divisor class and extracting the line function in various coordinate systems. From Table 4 we note that in the context of pairing computations our doubling formulae can save $2M$ in both projective and new coordinates, when compared to the formulae given by Lange [22].

In the new coordinate system, the rational function $c(x, y) = y - l(x)$ that is required in Miller's algorithm has the following form:

$$c(x, y) = y - \left(\frac{s_1}{s_1' Z_{32}}x^3 + \frac{l_2}{Z_{31}Z_{32}}x^2 + \frac{l_1}{Z_{31}Z_{32}}x + \frac{l_0}{Z_{31}Z_{32}}\right),$$

where $s_1, s_1', l_2, l_1, l_0, Z_{31}$ and Z_{32} are available in Table 3. By defining the auxiliary rational function $c'(x,y) = (Z_{31}Z_{32})c(x,y)$, we obtain

$$c'(x,y) = (Z_{31}Z_{32})y - ((s_1z_{11})x^3 + l_2x^2 + l_1x + l_0),$$

where z_{11} is also available in Table 3. With the same argument as the case of the mixed-addition, we have the relation $c(D_2)^{(q^k-1)/n} = c'(D_2)^{(q^k-1)/n}$ for an image divisor D_2. Therefore, we can simply work with the rational function $c'(x,y)$ instead of $c(x,y)$ without altering the value of the resulting Tate pairing in the new coordinate system.

4 Implementing the Tate Pairing

4.1 The Non-supersingular Pairing-Friendly Genus 2 Curve

There are only a few techniques that have been proposed for constructing non-supersingular curves of genus $g \geq 2$ and low embedding degree for pairing-based cryptography — see [10,11,18,21] for example. By modeling on the Cocks-Pinch method for constructing pairing-friendly elliptic curves [8], Freeman generated the first examples of non-supersingular pairing-friendly genus 2 curves [11]. In our implementations, we will use an example from [11], which gives a genus 2 curve whose Jacobian has embedding degree 2 with respect to the prime $n = 2^{160} + 7$. The curve is given by the equation

$$C : y^2 = x^5 + f_3x^3 + f_2x^2 + f_1x + f_0$$

over $\mathbb{F}_p$. The curve coefficients f_3, f_2, f_1 and f_0, the subgroup order n, and the characteristic p of the prime field can be found in Appendix A of [11]. Although Freeman [11] also provides examples of non-supersingular genus 2 curves with larger embedding degree, those curves are defined over prime fields with very large characteristics and therefore are not suitable for efficient implementations. Generating non-supersingular pairing-friendly genus 2 curves defined over small prime fields ($|p| \sim 80$) with large embedding degree ($k \geq 12$) remains an open problem.

Let $c \in \mathbb{F}_p$ be a quadratic non-residue over $\mathbb{F}_p$. A quadratic twist of C, denoted by C_t, over $\mathbb{F}_p$ is defined by the following equation

$$C_t : y^2 = x^5 + c^2f_3x^3 + c^3f_2x^2 + c^4f_1x + c^5f_0.$$

Let $\mathcal{J}_{C_t}(\mathbb{F}_{p^2})$ be the Jacobian of C_t when considering C_t as a curve defined over $\mathbb{F}_{p^2}$, and $\overline{D}_t = [u_t, v_t]$ be an element of $\mathcal{J}_{C_t}(\mathbb{F}_{p^2})$ in Mumford representation. It is known [1] that $C_t(\mathbb{F}_{p^2})$ is isomorphic to $C(\mathbb{F}_{p^2})$. Therefore, we can construct the isomorphism ψ of Jacobians $\mathcal{J}_{C_t}(\mathbb{F}_{p^2})$ and $\mathcal{J}_C(\mathbb{F}_{p^2})$ by applying the isomorphism ϕ to each point $P = (x_t, y_t)$ in the support of the divisor D_t as shown in the following figure.

$$\psi\colon \mathcal{J}_{C_t}(\mathbb{F}_{p^2}) \longrightarrow \mathcal{J}_C(\mathbb{F}_{p^2}) \qquad \phi\colon C_t(\mathbb{F}_{p^2}) \longrightarrow C(\mathbb{F}_{p^2})$$
$$\overline{D_t} \longmapsto \overline{D} \qquad\qquad (x_t, y_t) \longmapsto (x, y)$$
$$[u_t, v_t] \longmapsto [u, v] \qquad\qquad (x_t, y_t) \longmapsto (c^{-1} x_t, c^{-5/2} y_t)$$

The Isomorphism of $\mathcal{J}_{C_t}(\mathbb{F}_{p^2})$ and $\mathcal{J}_C(\mathbb{F}_{p^2})$ $\qquad$ The Isomorphism of $C_t(\mathbb{F}_{p^2})$ and $C(\mathbb{F}_{p^2})$

4.2   Finite Field Arithmetic

In our implementation, the curve C has embedding degree $k = 2$. Therefore we first need to construct the quadratic extension field $\mathbb{F}_{p^2}$. Since the prime p in this paper is congruent to 5 modulo 12, the quadratic extension field $\mathbb{F}_{p^2}$ can be constructed by the irreducible binomial $x^2 + 3$. Letting β denote -3, the elements of the field $\mathbb{F}_{p^2}$ can be represented as $a + b\sqrt{\beta}$, where $a, b \in \mathbb{F}_p$. By using the Karatsuba multiplication technique [20], a multiplication of two elements in $\mathbb{F}_{p^2}$ costs 3 multiplications in $\mathbb{F}_p$.

4.3   Using Degenerate Divisors and Denominator Elimination

Degenerate divisors have been widely used in the literature to speed up pairing computations on supersingular hyperelliptic curves [2,9,16]. Frey and Lange [12] have shown that the value of the Tate pairing is non-trivial if one restricts the second input to the embedding of $C(\mathbb{F}_{q^k})$ into $\mathcal{J}_C(\mathbb{F}_{q^k})$. In particular, when the embedding degree k is even, we can use a degenerate divisor class $\overline{P - P_\infty} \in \mathcal{J}_C(\mathbb{F}_{q^k})$ as the second argument of the Tate pairing where the coordinates of $P = (x, y) \in C(\mathbb{F}_{q^k})$ satisfy $x \in \mathbb{F}_{q^{k/2}}$ but $y \notin \mathbb{F}_{q^{k/2}}$. Therefore, in our implementation we first generate a degenerate divisor class $\overline{D_t} = [x - x_t, y_t] \in \mathcal{J}_{C_t}(\mathbb{F}_p)$ on the twisted curve $C_t/\mathbb{F}_p$. We then use the isomorphism ψ given above to obtain the degenerate divisor class $\overline{D} = \psi(\overline{D_t}) = [x - c^{-1} x_t, c^{-5/2} y_t] \in \mathcal{J}_C(\mathbb{F}_{p^2})$ on the curve C defined over $\mathbb{F}_{p^2}$. Hence, $\overline{D}$ can be used as the second argument to Miller's algorithm. Note that the first part of $\overline{D}$, namely $x - c^{-1} x_t$, is defined over $\mathbb{F}_p$, and thus the denominator elimination technique [3] applies in this case.

4.4   Evaluating Line Functions

At each iteration of the loop, we extract the rational functions $y - l(x)$ and $u_3(x)$ from the group operations and evaluate these functions at the second argument D_2. When new coordinates are used, we can work with $c'(x, y) = (r z_{24})y - ((s'_1 z_{21})x^3 + l_2 x^2 + l_1 x + l_0)$ and $u'_3(x) = z_{31} x^2 + U_{31} x + U_{30}$ for group addition, and $c'(x, y) = (Z_{31} Z_{32})y - ((s_1 z_{11})x^3 + l_2 x^2 + l_1 x + l_0)$ and $u'_3(x) = z_{31} x^2 + U_{31} x + U_{30}$ for group doubling as described in Section 3, where $\tilde{r}, z_{11}, z_{21}, Z_{31}, Z_{32}, s_1, s'_1, l_2, l_1$ and l_0 are from Table 1 and Table 3. We consider the following two cases for evaluating line functions at the divisor D_2:

1. D_2 is a degenerate divisor generated by the method in Section 4.3. In this case, D_2 can be represented by $[x - x_2, y_2] \in \mathcal{J}_C(\mathbb{F}_{p^2})$ for which $x_2 \in \mathbb{F}_p$ and $y_2 \notin \mathbb{F}_p$. Since x_2^2 and x_2^3 can be precomputed, this leaves $6M$ over $\mathbb{F}_p$ to be computed each time the function $c'(x, y)$ is evaluated. In particular, denominator elimination is applicable in this case and we do not need to

Table 5. Evaluating $c'(x,y)$ and $u'_3(x)$ at a General Divisor D_2 in New Coordinates

Input	$c'(x,y) = (\tilde{r}z_{21})y - ((s'_1 z_{21})x^3 + l_2 x^2 + l_1 x + l_0) \in \mathbb{F}_p[x,y]$	
	$u'_3(x) = z_{31}x^2 + U_{31}x + U_{30} \in \mathbb{F}_p[x]$	
	$D_2 = [x^2 + u_{21}x + u_{20}, v_{21}x + v_{20}] \in \mathcal{J}_C(\mathbb{F}_{p^2})$	
Output	$c'(D_2), u'_3(D_2) \in \mathbb{F}_{p^2}$	

Precomputations	Cost
$t_1 = u_{20}v_{21}, t_2 = u_{21}v_{20}, t_3 = t_1 - t_2, t_4 = v_{21}t_3, t_5 = v_{20}^2$	$13M, 3S$
$t_6 = t_4 + t_5, t_7 = u_{21}v_{21}, t_8 = 2v_{20} - t_7, t_9 = t_1 + t_3, t_{10} = u_{21}t_3$	in $\mathbb{F}_{p^2}$
$t_{11} = u_{20}v_{20}, t_{12} = t_{10} + 2t_{11}, t_{13} = u_{21}^2, t_{14} = t_3 t_{13}, t_{15} = 2t_3 - t_2$	
$t_{16} = u_{20}t_{15}, t_{17} = t_{14} - t_{16}, t_{18} = u_{20}u_{21}, t_{19} = u_{20}^2, t_{20} = t_{19}u_{20}$	
$t_{21} = t_{19}u_{21}, t_{22} = t_{13} - 2u_{20}, t_{23} = u_{20}t_{22}, t_{24} = t_{22} - u_{20}, t_{25} = u_{21}t_{24}$	

Computing $c'(D_2)$	Cost
$w_1 = \tilde{r}z_{21}, w_2 = s'_1 z_{21}, w_3 = w_1 t_6, w_4 = w_2 t_{17}, w_5 = l_2 t_{12}, w_6 = l_1 t_9$	$38M, 1S$
$w_7 = l_0 t_8, w_8 = w_3 - w_4 + w_5 - w_6 - w_7, w_9 = w_1 w_8, w_{10} = w_2 t_{20}$	in $\mathbb{F}_p$
$w_{11} = l_2 t_{21}, w_{12} = l_1 t_{23}, w_{13} = l_0 t_{25}, w_{14} = w_{10} - w_{11} + w_{12} - w_{13}$	
$w_{15} = w_2 w_{14}, w_{16} = l_2 t_{19}, w_{17} = l_1 t_{18}, w_{18} = l_0 t_{22}, w_{19} = w_{16} - w_{17} + w_{18}$	
$w_{20} = l_2 w_{19}, w_{21} = l_1 u_{20}, w_{22} = l_0 u_{21}, w_{23} = w_{21} - w_{22}, w_{24} = l_1 w_{23}$	
$w_{25} = l_0^2, c'(D_2) = w_9 + w_{15} + w_{20} + w_{24} + w_{25}$	

Computing $u'_3(D_2)$	Cost
$i_1 = z_{31}^2, i_2 = z_{31}U_{30}, i_3 = U_{30}^2, i_4 = i_1 t_{19}, i_5 = U_{31}u_{20}, i_6 = z_{31}t_{18}, i_7 = U_{30}u_{21}$	$13M, 2S$
$i_8 = i_5 - i_6 - i_7, i_9 = U_{31}i_8, i_{10} = i_2 t_{22}, u'_3(D_2) = i_3 + i_4 + i_9 + i_{10}$	in $\mathbb{F}_p$

evaluate the function $u'_3(x)$ at D_2. Therefore, the total cost of evaluating the rational functions at a degenerate divisor D_2 is given as $6M$ in $\mathbb{F}_p$ per iteration of the loop, with a precomputation of $1M + 1S$ in $\mathbb{F}_p$.

2. D_2 is a general divisor in Mumford representation, namely $D_2 = [u_2(x), v_2(x)] = [x^2 + u_{21}x + u_{20}, v_{21}x + v_{20}] \in \mathcal{J}_C(\mathbb{F}_{p^2})$. Note that the Mumford representation of divisors essentially gives the symmetric functions of the coordinates of the points in the support of the divisor. Hence, we can use these symmetric functions to obtain an explicit formula for evaluating the rational functions $c'(x,y)$ and $u'_3(x)$ at D_2, which only uses the coefficients of $u_2(x)$ and $v_2(x)$. Table 5 describes the efficient explicit formula for computing $c'(D_2)$ and $u'_3(D_2)$.

Note that, in many of the multiplications in Table 5, one of the operands is in $\mathbb{F}_p$. Hence, a multiplication in $\mathbb{F}_{p^2}$ only needs $2M$ in $\mathbb{F}_p$ in this case. The total cost of evaluating the rational functions at a general divisor D_2 is given as $51M + 3S$ in $\mathbb{F}_p$ per iteration of the loop, with a precomputation of $13M + 3S$ in $\mathbb{F}_{p^2}$.

4.5   Final Exponentiation

For a genus 2 curve with an embedding degree of $k = 2$, the output of Miller's algorithm must be exponentiated to the power of $(p^2 - 1)/n$. The final exponentiation can be expressed in terms of operations in the base field $\mathbb{F}_p$. Letting

$f = a + b\sqrt{\beta} \in \mathbb{F}_{p^2}$ denote the output of Miller's algorithm, we can compute the final exponentiation as follows:

$$f^{\frac{p^2-1}{n}} = \left(\frac{a - b\sqrt{\beta}}{a + b\sqrt{\beta}}\right)^{\frac{p+1}{n}} = \left(\frac{a^2 - 3b^2 + \sqrt{\beta}\left((a-b)^2 - (a^2 + b^2)\right)}{a^2 + 3b^2}\right)^{\frac{p+1}{n}}.$$

We first calculate the expression in the parenthesis with $1I + 2M + 3S$ in $\mathbb{F}_p$, followed by an expensive exponentiation by $(p+1)/n$ which is executed in the arithmetic in $\mathbb{F}_{p^2}$.

4.6 Efficiency Comparison and Analysis

Since our encapsulated explicit formulae are applicable to pairing computations on both supersingular and non-supersingular genus 2 curves, we first show how our method can be used to improve previous implementations on supersingular genus 2 curves. We then analyze the case of non-supersingular genus 2 curves.

In [7] and [16], the authors considered the pairing computation on a family of supersingular genus 2 hyperelliptic curves with embedding degree 4 in affine coordinates. The curves are defined by the equation $y^2 = x^5 + a$, where $a \in \mathbb{F}_p^*$ and $p \equiv 2, 3 \bmod 5$. Note that our explicit doubling formulae only need $37M + 6S$ and $32M + 6S$ in projective and new coordinates, respectively, for this family of curves since the curve coefficients f_2 and f_3 are zero. Assume that the order n of the subgroup is about 160 bits. Following the same analysis as in [7] and [16], we compare the cost of computing the Tate pairing on this family of curves in different coordinate systems (without including the cost of the final exponentiation) in Table 6.

Table 6. Theoretical Complexity of Miller's Algorithm in Different Systems

Reference	Coordinate Type	Subgroup Order	Cost
Choie and Lee [7]	Affine	Random	$240I, 17688M, 2163S$
Ó hÉigeartaigh and Scott [16]	Affine	Solinas Prime	$162I, 10375M, 645S$
Our work	Projective	Random	$20017M, 1201S$
		Solinas Prime	$13129M, 967S$
	New	Random	$19297M, 1361S$
		Solinas Prime	$12487M, 971S$

We assume that field squarings have cost $S = 0.8M$. Then our encapsulated method is faster than that of [7] whenever $I/M > 4.03$. Moreover, our algorithm can achieve better performance than that of [16] whenever $I/M > 14.65$. These conditions usually hold for large prime field arithmetic on modern processors [25]. Therefore, for sufficiently large I/M, our method based on the encapsulated explicit formulae will be superior.

Next, we analyze the computational complexity of computing the Tate pairing using the non-supersingular genus 2 curve with embedding degree 2 (see Section

4.1). Note that the group order $n = 2^{160} + 7 = 2^{160} + 2^3 - 1$ is a Solinas prime [32]. The encapsulated explicit formulae for performing the group operations and extracting the rational functions in new coordinates are used (See Section 3). Furthermore, the degenerate divisor generated by using the technique in Section 4.3 is used as the second argument to Miller's algorithm. Based on these optimizations, the theoretical cost for computing the Tate pairing is given as (again without including the cost of the final exponentiation)

$$(\log_2 n)(T_D + T_c + T_{sk} + T_{mk}) + 2(T_A + T_c + T_{mk}),$$

where $T_A = 36M + 5S$ is the cost of adding two general divisors and extracting the rational functions with the formula in Table 1, $T_D = 35M + 7S$ is the cost of doubling a general divisor and extracting the rational functions with the formula in Table 3, $T_c = 6M$ (with a precomputation of $1M + 1S$) is cost of evaluating the rational function $c'(x, y)$ at the degenerate divisor D_2, and $T_{sk} = 3S$ and $T_{mk} = 3M$ are respectively the cost of squaring and multiplication in $\mathbb{F}_{p^2}$. Hence, the total cost of computing the Tate pairing with our optimizations is given as $7175M + 1621S$ in $\mathbb{F}_p$, whereas $163I + 4243M + 975S$ are required when using affine coordinates.

4.7   Experimental Results

In this section, experimental results are given for computing the Tate pairing using the techniques detailed in this paper for the non-supersingular genus 2 curve defined over $\mathbb{F}_p$ with embedding degree 2. All experiments were conducted on a Core 2 Duo™ processor with a clock frequency of 2.67 GHz. The code was written in C and complied and debugged using *Microsoft Developer Studio 6*. The implementation of $\mathbb{F}_p$-arithmetic is based on various efficient algorithms in [15], where p is a 651-bit prime. Table 7 shows the timings of our finite field library and the corresponding IM-ratio. From Table 7, we note that the IM-ratio is 45.8 and $S = 0.89M$ in the target processor. Therefore, using our encapsulated explicit formulae, we can obtain a 31.5% performance improvement over working with affine coordinates.

Table 7. Timings of Prime Field $\mathbb{F}_p$ Library

# of bits of p	Multiplication (M)	Squaring (S)	Inversion (I)	IM-ratio
651	$4.59\mu s$	$4.08\mu s$	$210\mu s$	45.8

Table 8 gives experimental results for the implementation of the Tate pairing for the (160/1024) security level. All of the timings are given in milliseconds and three cases are included in the Table. The first case is the time taken to compute the Tate pairing when a degenerate divisor is used as the second argument to Miller's algorithm and the denominator technique is applied. The second case

Table 8. Experimental Results – (160/1024) Security Level

Case	Description	Running Time (ms)
1	Evaluating at a degenerate divisor	46.5
2	Evaluating using Mumford Representation	85.2
3	Elliptic curve pairing ($k = 2$ and $\log_2 p \approx 512$) [31]	8.9

gives the time when a general divisor with Mumford representation is used as the second input to the algorithm. The third case is the timing for computing the Tate pairing on non-supersingular elliptic curves with embedding degree 2 over $\mathbb{F}_p$ given by Scott [31].

Note that the implementations in [16] and [31] use special assembly language routines in MIRACL [30] for field operations. Therefore, it is hard to compare our implementation with those in [16,31]. The timings in Table 8 indicate that the Tate pairing on the non-supersingular genus 2 curve over $\mathbb{F}_p$ is a valid candidate for practical applications. However, elliptic curve pairings are faster than those in the genus 2 case for a (160/1024) bit security level, due to the more complicated group operations and larger Jacobian sizes of genus 2 curves.

5   Conclusion

In this paper, we have described how to efficiently implement pairing computations on genus 2 hyperelliptic curves over prime fields in projective coordinates. We generalize Chatterjee *et. al.*'s idea of encapsulated double-and-line computation and add-and-line computation to genus 2 curves in projective and new coordinates, respectively. We also show that some of the operations in the encapsulated method do not need to be computed since they are eliminated by the final exponentiation. Our new explicit formulae are applicable to pairing computations on both supersingular and non-supersingular genus 2 curves. Theoretical analysis shows that for pairing computations on supersingular genus 2 curves with embedding degree 4 over prime fields, our encapsulated method is faster than previously best known algorithms whenever $I/M > 14.65$. Furthermore, we also report the first efficient implementation of pairing computations on a non-supersingular genus 2 curve with embedding degree 2 over prime fields using the encapsulated explicit formulae and various known optimization techniques.

References

1. 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 (2006)
2. Barreto, P.L.S.M., Galbraith, S., Ó hÉigeartaigh, C., Scott, M.: Efficient Pairing Computation on Supersingular Abelian Varieties. Design, Codes and Cryptography 42, 239–271 (2007)

3. Barreto, P.L.S.M., Kim, H.Y., Lynn, B., Scott, M.: Efficient Algorithm for Pairing-Based Cryptosystems. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, p. 354. Springer, Heidelberg (2002)
4. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. SIAM Journal of Computing 32(3), 586–615 (2003)
5. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
6. Chatterjee, S., Sarkar, P., Barua, R.: Efficient Computation of Tate Pairing in Projective Coordinate over General Characteristic Fields. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 168–181. Springer, Heidelberg (2005)
7. Choie, Y., Lee, E.: Implementation of Tate Pairing on Hyperelliptic Curve of Genus 2. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 97–111. Springer, Heidelberg (2004)
8. Cocks, C., Pinch, R.G.E.: Identity-based Cryptosystems Based on the Weil Pairing (Unpublished manuscript) (2001)
9. Duursma, I.M., Lee, H.-S.: Tate pairing implementation for hyperelliptic curves $y^2 = x^p − x + d$. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 111–123. Springer, Heidelberg (2003)
10. Galbraith, S.D., McKee, J.F., Valença, P.C.: Ordinary Abelian Varieties Having Small Embedding Degree. Finite Fields and Their Applications 13(4), 800–814 (2007)
11. Freeman, D.: Constructing Pairing-Friendly Genus 2 Curves over Prime Fields with Ordinary Jacobians. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 152–176. Springer, Heidelberg (2007)
12. Frey, G., Lange, T.: Fast Bilinear Maps from The Tate-Lichtenbaum Pairing on Hyperelliptic Curves. In: Hess, F., Pauli, S., Pohst, M. (eds.) ANTS 2006. LNCS, vol. 4076, pp. 466–479. Springer, Heidelberg (2006)
13. Frey, G., Rück, H.-G.: A Remark Concerning m-Divisibility and the Discrete Logarithm Problem in the Divisor Class Group of Curves. Mathematics of Computation 62(206), 865–874 (1994)
14. Granger, R., Hess, F., Oyono, R., Thériault, N., Vercauteren, F.: Ate Pairing on Hyperelliptic Curves. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 430–447. Springer, Heidelberg (2007)
15. Hankerson, D., Menezes, A., Vanstone, S.: Guide to Elliptic Curve Cryptography. Springer, New York (2004)
16. Ó hÉigeartaigh, C., Scott, M.: Pairing Calculation on Supersingular Genus 2 Curves. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 302–316. Springer, Heidelberg (2007)
17. Hess, F., Smart, N.P., Vercauteren, F.: The Eta Pairing Revisited. IEEE Transactions on Information Theory 52(10), 4595–4602 (2006)
18. Hitt, L.: Families of Genus 2 Curves with Small Embedding Degree, Cryptology ePrint Archive, Report 2007/001 (2007), http://eprint.iacr.org/2007/001
19. Joux, A.: A One-Round Protocol for Tripartite Diffie-Hellman. In: Bosma, W. (ed.) ANTS 2000. LNCS 1838, vol. 1838, pp. 385–394. Springer, Heidelberg (2000)
20. Karatsuba, A., Ofman, Y.: Multiplication of Multidigit Numbers on Automata. Soviet Physics Doklady (English Translation) 7(7), 595–596 (1963)
21. Kawazoe, M., Takahashi, T.: Pairing-friendly Hyperelliptic Curves of Type $y^2 = x^5 + ax$, Cryptology ePrint Archive, Report 2008/026 (2008), http://eprint.iacr.org/2008/026

22. Lange, T.: Formulae for Arithmetic on Genus 2 Hyperelliptic Curves. Applicable Algebra in Engineering, Communication and Computing 15(5), 295–328 (2005)
23. Lee, E., Lee, H.-S., Lee, Y.: Eta Pairing Computation on General Divisors over Hyperelliptic Curves $y^2 = x^7 - x \pm 1$. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 349–366. Springer, Heidelberg (2007)
24. Menezes, A., Okamoto, T., Vanstone, S.A.: Reducing Elliptic Curve Logarithms to a Finite Field. IEEE Transactions on Information Theory 39(5), 1639–1646 (1993)
25. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. Chapman & Hall/CRC, Boca Raton (1997)
26. Miller, V.S.: Short Programs for Functions on Curves (Unpublished manuscript) (1986), `http://crypto.stanford.edu/miller/miller.pdf`
27. Miyamoto, Y., Doi, H., Matsuo, K., Chao, J., Tsujii, S.: A Fast Addition Algorithm of Genus Two Hyperelliptic Curve. In: The 2002 Symposium on Cryptography and Information Security - SCIS 2002, pp. 497–502 (2002) (in Japanese)
28. Mumford, D.: Tata Lectures on Theta II. In: Prog. Math., vol. 43. Birkhäuser (1984)
29. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems Based on Pairings. In: Proceedings of the 2000 Symposium on Cryptography and Information Security - SCIS 2002, Okinawa, Japan, pp. 26–28 (2000)
30. Scott, M.: MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library), `http://www.shamus.ie/`
31. Scott, M.: Scaling Security in Pairing-based Protocols, Cryptology ePrint Archive, Report 2005/139 (2005), `http://eprint.iacr.org/2005/139`
32. Solinas, J.: Generalized Mersenne Primes, Centre for Applied Cryptographic Research (CACR) Technical Reports, CORR 99-39, `http://www.cacr.math.uwaterloo.ca/techreprots/1999/corr99-39.pdf`

Appendix: Explicit Formulae for Genus 2 Curves over $\mathbb{F}_p$

In this appendix, we give efficient explicit formulae for group operations on genus 2 curves over $\mathbb{F}_p$ in projective coordinates in the context of pairing computations. Table 9 and Table 10 address the cases of projective coordinates. Given two divisor classes $\overline{E}_1$ and $\overline{E}_2$, Table 9 computes the divisor class $\overline{E}_3 = [u_3(x), v_3(x)]$ and the rational function $l(x)$ such that $E_1 + E_2 = E_3 + \operatorname{div}\left(\frac{y - l(x)}{u_3(x)}\right)$ in the projective coordinate system, where $l(x) = \frac{s_1'}{r}x^3 + \frac{l_2}{rZ_2}x^2 + \frac{l_1}{rZ_2}x + \frac{l_0}{rZ_2}$. For doubling a reduced divisor class E_1, Table 10 calculates the divisor class $\overline{E}_3 = [u_3(x), v_3(x)]$ and the rational function $l(x)$ such that $2E_1 = E_3 + \operatorname{div}\left(\frac{y - l(x)}{u_3(x)}\right)$ in projective coordinates, where $l(x) = \frac{s_1}{R'}x^3 + \frac{l_2}{R'Z_1}x^2 + \frac{l_1}{R'Z_1}x + \frac{l_0}{R'Z_1}$.

Table 9. Mixed-Addition Formula on a Genus 2 curve over $\mathbb{F}_p$ (Projective Coordinates)

Input	Genus 2 HEC $C: y^2 = x^5 + f_3 x^3 + f_2 x^2 + f_1 x + f_0$	
	$\overline{E}_1 = [U_{11}, U_{10}, V_{11}, V_{10}, 1]$ and $\overline{E}_2 = [U_{21}, U_{20}, V_{21}, V_{20}, Z_2]$	
Output	$E_3 = [U_{31}, U_{30}, V_{31}, V_{30}, Z_3] = \overline{E}_1 \oplus \overline{E}_2$	
	$l(x)$ such that $E_1 + E_2 = E_3 + \mathrm{div}\left(\frac{y - l(x)}{u_3(x)} \right)$	
Step	**Expression**	**Cost**
1	**Compute resultant $r = \mathrm{Res}(u_1, u_2)$:**	$5M, 1S$
	$\tilde{U}_{11} = U_{11} Z_2, \tilde{U}_{10} = U_{10} Z_2, z_1 = \tilde{U}_{11} - U_{21}, z_2 = U_{20} - \tilde{U}_{10}$	
	$z_3 = U_{11} z_1, z_4 = z_2 + z_3, r = z_2 z_4 + z_1^2 U_{10}$	
2	**Compute almost inverse of $u_2 \bmod u_1$:**	$-$
	$inv_1 = z_1, inv_0 = z_4$	
3	**Compute s':**	$7M$
	$w_0 = V_{10} Z_2 - V_{20}, w_1 = V_{11} Z_2 - V_{21}, w_2 = inv_0 w_0$	
	$w_3 = inv_1 w_1, s_1' = z_1 w_0 + z_2 w_1, s_0' = w_2 - U_{10} w_3$	
4	**Precomputations:**	$4M, 1S$
	$R = r^2, \tilde{s}_0' = s_0' Z_2, \tilde{s}_1' = s_1' Z_2, S = s_1' \tilde{s}_1', \tilde{r} = r \tilde{s}_1'$	
5	**Compute l:**	$5M$
	$l_2 = s_1' U_{21} + \tilde{s}_0', l_0 = s_0' U_{20} + r V_{20}$	
	$l_1 = (s_1' + s_0')(U_{21} + U_{20}) - s_1' U_{21} - s_0' U_{20} + r V_{21}$	
6	**Compute U_3:**	$8M, 1S$
	$w_1 = \tilde{U}_{11} + U_{21}, U_{31} = s_1'(2\tilde{s}_0' - s_1' z_1) - R Z_2, l_1' = l_1 s_1'$	
	$U_{30} = \tilde{s}_0'(s_0' - 2s_1' U_{11}) + s_1'^2(z_3 - \tilde{U}_{10} - U_{20}) + 2l_1' + R w_1$	
7	**Compute V_3:**	$6M$
	$w_1 = l_2 s_1' - U_{31}, V_{30} = U_{30} w_1 - S(l_0 s_1'), V_{31} = U_{31} w_1 + S(U_{30} - l_1')$	
8	**Adjust:**	$3M$
	$Z_3 = \tilde{r} S, U_{31} = \tilde{r} U_{31}, U_{30} = \tilde{r} U_{30}$	
Sum		$38M, 3S$

Table 10. Doubling Formula on a Genus 2 Curve over $\mathbb{F}_p$ (Projective Coordinates)

Input	Genus 2 HEC $C: y^2 = x^5 + f_3 x^3 + f_2 x^2 + f_1 x + f_0$	
	$\overline{E}_1 = [U_{11}, U_{10}, V_{11}, V_{10}, Z_1]$	
Output	$E_3 = [U_{31}, U_{30}, V_{31}, V_{30}, Z_3] = [2]\overline{E}_1$	
	$l(x)$ such that $2E_1 = E_3 + \mathrm{div}\left(\frac{y - l(x)}{u_3(x)} \right)$	
Step	**Expression**	**Cost**
1	**Compute resultant and precomputations:**	$4M, 3S$
	$Z_2 = Z_1^2, \tilde{V}_{11} = 2V_{11}, \tilde{V}_{10} = 2V_{10}, w_0 = V_{11}^2, w_1 = U_{11}^2, w_2 = \tilde{V}_{10} Z_1$	
	$w_3 = 4w_0, w_4 = w_2 - U_{11} \tilde{V}_{11}, r = U_{10} w_3 + \tilde{V}_{10} w_4$	
2	**Compute almost inverse:**	$-$
	$inv_1' = -\tilde{V}_{11}, inv_0' = w_4$	
3	**Compute k':**	$5M$
	$w_3 = f_3 Z_2 + w_1, w_4 = 2U_{10}, \tilde{w}_4 = w_4 Z_1, k_1' = 2w_1 + w_3 - \tilde{w}_4$	
	$k_0' = U_{11}(2\tilde{w}_4 - w_3) + Z_1(f_2 Z_2 - w_0)$	
4	**Compute s':**	$7M$
	$w_0 = k_0' inv_0', w_1 = k_1' inv_1', s_2 = w_2 k_1' - \tilde{V}_{11} k_0'$	
	$s_1' = s_2 Z_1, s_0' = w_0 - Z_1 U_{10} w_1$	
5	**Precomputations:**	$6M, 2S$
	$R = r Z_2, \tilde{R} = R s_1', R' = R s_2, S_0 = s_0'^2, S_1 = s_1'^2, S = s_0' s_1', s_0 = s_0' s_2, s_1 = s_1' s_2$	
6	**Compute l:**	$6M$
	$l_2 = s_1 U_{11} + s_0 Z_1, l_0 = s_0 U_{10} + R' V_{10}$	
	$l_1 = (s_1 + s_0)(U_{11} + U_{10}) - s_1 U_{11} - s_0 U_{10} + R' V_{11}$	
7	**Compute U_3:**	$4M, 1S$
	$U_{30} = S_0 + R(s_2 \tilde{V}_{11} + 2r Z_1 U_{11}), U_{31} = 2S - R^2$	
8	**Compute V_3:**	$4M$
	$w_1 = l_2 - U_{31}, w_2 = U_{30} w_1, w_3 = U_{31} w_1$	
	$V_{31} = w_3 + S_1(U_{30} - l_1), V_{30} = w_2 - S_1 l_0$	
9	**Adjust:**	$3M$
	$Z_3 = S_1 \tilde{R}, U_{31} = U_{31} \tilde{R}, U_{30} = U_{30} \tilde{R}$	
Sum		$39M, 6S$