OPTIMIZATION OF THE ARITHMETIC OF THE IDEAL CLASS GROUP FOR GENUS 4 HYPERELLIPTIC CURVES OVER PROJECTIVE COORDINATES

Rodrigo Abarzúa\textsuperscript{1}, Nicolas Thériault\textsuperscript{2}, Roberto Avanzi\textsuperscript{3}, Ismael Soto\textsuperscript{1} and Miguel Alfaro\textsuperscript{1}

\textsuperscript{1}Depto de Ingeniería Industrial
Universidad de Santiago de Chile
Av. Ecuador 3769, Santiago, Chile
\textsuperscript{2}Instituto de Matemática y Física
Universidad de Talca
Casilla 747, Talca, Chile
\textsuperscript{3}Fakultät für Mathematik, Ruhr-Universität Bochum and
Horst Görtz Institut für IT-Sicherheit
Universitätstraße 150, D-44780 Bochum, Germany

(Communicated by Daniel Bernstein)

Abstract. The aim of this paper is to reduce the number of operations in Cantor’s algorithm for the Jacobian group of hyperelliptic curves for genus 4 in projective coordinates. Specifically, we developed explicit doubling and addition formulas for genus 4 hyperelliptic curves over binary fields with $h(x) = 1$. For these curves, we can perform a divisor doubling in $63M + 19S$, while the explicit adding formula requires $203M + 18S$, and the mixed coordinates addition (in which one point is given in affine coordinates) is performed in $165M + 15S$.

These formulas can be useful for public key encryption in some environments where computing the inverse of a field element has a high computational cost (either in time, power consumption or hardware price), in particular with embedded microprocessors.

1. Introduction

The search for more efficient cryptosystems for embedded microprocessors has been of great interest in the last few years. The most promising options for asymmetric key algorithms are based on the public-key exchange presented by Diffie and Hellman in 1976. The security of this type of communication is based on the difficulty in solving the discrete logarithm problem over a finite field, which was later extended to other groups. It can be formulated as follows: Given an additive group $G$ generated by an element $g$ and a second element $h$ in $G$, find some $t \in Z$ with $[t] \cdot g = h$. The computation of scalar multiples of a group element (i.e. given an integer $t$ and a element $g$, compute $[t] \cdot g$) is the fundamental operation of cryptosystems based on the DLP.

2000 Mathematics Subject Classification: Primary: 58F15, 58F17; Secondary: 53C35.

Key words and phrases: Hyperelliptic curves, genus 4, explicit formulas, binary field, projective coordinates.

The authors would like to thank PBCT/CONICYT act 11/04 - Chile (R. Abarzúa, I. Soto, and M. Alfaro) and FONDECYT Chile project #1070242 (N. Thériault) for their financial support.

©2010 AIMS-SDU
In the mid 1980s, Miller and Koblitz independently proposed a system based on the difficulty of the DLP in the group of points of an elliptic curve (EC, curves of genus 1) over a finite field. In 1988, Koblitz, suggested for the first time a generalization of elliptic curves, named hyperelliptic curves (HEC, curves of genus higher than 1).

Due to the work of Cantor (for curves over finite fields of odd characteristic), and Koblitz (generalized to fields of characteristic two), it is possible to perform efficient operations in the group of a hyperelliptic curve (we refer to both cases as Cantor’s Algorithm). The first fully practical algorithms for genus 2 HECC were obtained by Harley. The importance of these theoretical generalizations is mainly due to the different sizes of the fields in which the algorithms work (which decreases, but comes with the prices of having to perform more field operations). This can generate significant advantages in several situations, including low-power implementations on specialized microprocessors. Hyperelliptic curves can supply the same desired security levels for commercial transactions, but with smaller field sizes, so the processor requires much less power to perform the field arithmetic operations.

Given the restrictions on embedded microprocessors (used in the mobile devices), the HECC system has emerged as a promising alternative to give the required security. One line of research is the implementation in software. A second line of research is in platform-oriented hardware implementation, in particular FPGA. A major area of investigation for these cryptosystems, and one that has been getting a lot of attention in the last few years, is the optimization of Cantor’s algorithm via explicit formulas. This research is ongoing in the scientific community. Curves of genus 2 have received the most attention, and curves of genus 3 have also been extensively studied.

After the work of Nagao (for general hyperelliptic curves), the first work to present explicit formulas for genus 4 hyperelliptic curves is the one of Pelzl, Wollinger and Paar. One of the latest work on optimized arithmetic for the Jacobian group of hyperelliptic curves (in this case of genus 3 and 4), is a research carried out by Avanzi, Thériault and Wang. Unlike previous results available in the literature, this work simultaneously addresses field arithmetic enhancements (in software), and the derivation of explicit formulas (as well as the impact of recent attacks), and consider the interplay of these factors producing significant improvements for software implementations.

This latest work demonstrate that hyperelliptic curves of genus 4 can sometimes have a better performance than elliptic curves. They also compare well with HEC of genus 2 and 3, and can therefore be more efficient than previously thought. These works also show that more extensive research is required to know how genus 4 curves could be improved to adjust for the possible applications, in particular for embedded microprocessors, to find out what implementations could be generated.

When considering the algorithms of Cantor, Harley, etc., the operation with the greatest complexity and power requirement for the basic group arithmetic is the computation of the inverse of a field element. For fields of characteristic 2, the inversion of a field element can equivalent (in terms of time) to approximately 7 (and often more) multiplications of random elements in that field. There are environments in which inversion is even more critical in time and/or space, an example of this is the smart cards. It is also known that being able to perform the
group operations without inversion makes it possible to reduce the space used on the microprocessors. For example if the hardware only needs to calculate squares (multiplication of an element by itself), multiplications and sums of the field elements, it is possible to make the devices smaller, implying a reduction in production costs. Another alternative is to use the space of the inversion operation, replacing it by one (or several) multiplication operator(s). In this case, the hardware is of similar size, price, etc, but can compute more multiplications at the same time. That is to say, it is possible to calculate an operation of the Jacobian group in shorter real time.

A technique used to eliminate the inversion of field elements from the group operations is to take advantage of projective coordinates. In this paper we present for the first time inversion-free explicit formulas for genus 4 curves over fields of characteristic two, with $h(x) = 1$.

Genus 4 HECC can be interesting for the implementation of public key cryptosystems in embedded microprocessors in constrained environments. This is especially true for 8-bits microprocessors since the underlying arithmetic is performed in relatively small bit-lengths. In this setting, the hardware cost associated to the field inversion makes projective coordinates very interesting (much more than in software implementations).

The rest of paper is organized as follows. In Section 2, we present a brief overview of the mathematical background related to HECC and explicit formulas (in affine coordinates). The techniques used to obtain inversion-free explicit formulas (in projective coordinates) are presented in Section 3. In Sections 4 and 5, we present our explicit addition and doubling formulas in details. Finally, we draw some conclusions in Section 6.

2. Mathematical background

We first present some background on the theory of hyperelliptic curves over finite fields of characteristic two. An excellent, low brow, introduction to hyperelliptic curves can be found in [18]. A more geometric presentation of the theoretical background is given in [1].

2.1. From Cantor’s algorithm to explicit formulas. We consider a hyperelliptic curve of genus $g \geq 1$ explicitly given by a nonsingular equation of the form

$$C : y^2 + h(x)y = f(x)$$

over the finite field $\mathbb{F}_q$ of characteristic two. The degree of the polynomial $h(x) \in \mathbb{F}[x]$ is at most $g$ and $f(x) \in \mathbb{F}[x]$ is a monic polynomial of degree $2g + 1$. The nonsingularity condition requires $h(x)$ and $f(x)$ to be such that there are no pairs $(x, y) \in \overline{\mathbb{F}} \times \overline{\mathbb{F}}$ which simultaneously satisfy the equation of the curve $C$ and the partial differential equations $2y + h(x) = 0$ and $h'(x)y - f'(x) = 0$.

Let $P_\infty$ be the point at infinity on the curve. In general, the points on a hyperelliptic curve do not form a group (the notable exception being represented by hyperelliptic curves of genus one, i.e. elliptic curves). Instead, the divisor class group of $C$ is used: We briefly recall its main properties. The divisor class group is isomorphic to and sometimes identified with, the algebraic variety called the Jacobian of $C$ which we do not define nor study here.

---

*The latest operation has almost no significant impact on the final cost due to the bitwise nature of binary addition.*
A divisor $D$ is a formal sum of points on the curve, considered with multiplicities. Its degree is the sum of those multiplicities, and its support the set of points with nonzero multiplicity. We are interested in the divisors of degree zero given by sums of the form

$$\sum_{i=1}^{k} m_i P_i - m_P_\infty : P_i \in C \setminus P_\infty$$

where $m = \sum_{i=1}^{k} m_i$. The degree of the associated effective divisor $\sum_{i=1}^{k} m_i P_i$ is the integer $m$. The points $P_i$ form the finite support of $D$. Principal divisors are the divisors of functions i.e. those whose points are the poles and zeros of a rational functions on the curve, the multiplicity of each point being the order of the zero or minus the order of the pole at that point. The divisor class group is the quotient group of the degree zero divisors modulo the principal divisors. In each divisor class there exits a unique element of the form (1) with (effective) degree $m \leq g$. Such an element is called a reduced divisor.

The group elements are these reduced divisors and they can be represented as pairs of polynomials $[u(x), v(x)]$ satisfying:

1. $\deg(u) \leq g$;  
2. $\deg(v) < \deg(u)$;  
3. $u(x)$ is monic;  
4. $u(x)$ divides $v(x)^2 + v(x)h(x) - f(x)$.

This representation is usually attributed to Mumford [20]. If the first degree condition is not satisfied, the divisor is called semi-reduced.†

For computational purposes, the group operation is based on Cantor’s algorithm [4], that operates directly with elements in Mumford’s representation. Cantor’s original version worked only in odd characteristic and was extended for all fields by Koblitz [11]. Algorithm 1 gives the characteristic two case.

Note that at step 6 we are simply computing $v_C(x)$ to be congruent to $v_1(x)$ modulo $u_1(x)/d(x)$ and congruent to $v_2(x)$ modulo $u_2(x)/d(x)$.

The idea behind explicit formulas is to replace the polynomial-based form of Cantor’s algorithm by a coefficient-based approach. These formulas are case-specific, i.e. they depend on whether the divisors are distinct (addition) or equal (doubling), on the degrees of the polynomials involved, etc. (For a detailed case consideration in genus two see for example [14], for genus three see [7]). This approach has a number of advantages which result in a significant speed-up in the computations:

- Coefficients that have no impact on the final result are no longer computed.
- This is quite evident in step 10 where we do not need to compute the coefficients for powers $x$ of degree less than $\deg(u_i)$ in $v_i^2 + hv_i + f$, since we know that the division $v_i^2 + hv_i + f$ is exact and thus has no fractional part.
- In Cantor’s algorithm, some of the partial computations may be done twice, with only the variable names being different. These repetitions are avoided in explicit formulas by keeping those values in memory.
- Parts of the algorithm can be replaced by more efficient techniques that could not be used in a general setting.
- Conditional statements can be reduced to a minimum. Polynomial arithmetic is inherently dependent on conditional loops (mainly on the degree of the polynomial), which cannot be avoided in a general setting. Although checking

†Strictly in terms of divisors (points), to have a reduced divisor one must also require that there can be no pairs of points $(x_0, y_0)$, $(x_0, -y_0 - h(x_0))$ in the finite support of the divisor. However, this condition is implicit in Mumford’s representation.
### Algorithm 1 Group operation for hyperelliptic Jacobian in characteristic two

**Inputs:** Reduced divisors $D_1 = [u_1(x), v_1(x)]$ and $D_2 = [u_2(x), v_2(x)]$

**Outputs:** Reduced divisor $D_3 = [u_3(x), v_3(x)]$, $D_3 = D_1 + D_2$

1. **Composition:** $[u_C, v_C] = D_1 + D_2$ (semi-reduced)
2. $d_1 \leftarrow \text{gcd}(u_1, u_2)$
   
   where $d_1 = s_1 u_1 + s_2 u_2$  
   
   [Extended Euclidean Algorithm]
3. $d \leftarrow \text{gcd}(d_1, v_1 + v_2 + h)$
   
   where $d = t_1 d_1 + t_2 (v_1 + v_2 + h)$  
   
   [Extended Euclidean Algorithm]
4. $r_1 \leftarrow s_1 t_1$, $r_2 \leftarrow s_2 t_1$ and $r_3 \leftarrow t_2$
5. $u_C \leftarrow u_1 u_2 / d^2$
6. $v_C \leftarrow v_2 + \frac{h}{v_2} r_2 (v_1 + v_2) + r_3 (s_2^2 + h v_2 + f)$
7. **Reduction:** $D_3 = [u_3, v_3]$ (reduced)
8. $u_0 \leftarrow u_C$, $v_0 \leftarrow v_C$
9. for $i = 0$ while $\text{deg}(u_i) > g$ do
10.   $u_{i+1} \leftarrow \text{Monic}(\frac{v_i^2 + h v_i + f}{u_i})$
11.   $v_{i+1} \leftarrow v_i + h \mod u_{i+1}$
12.   $i \leftarrow i + 1$
13. end for
14. $u_3 \leftarrow u_1$, $v_3 \leftarrow v_1$

A conditional statement (for example “is k < \text{deg}(u)?”) is not very expensive on its own, the cumulative impact can be significant:

- For almost all reduced divisors $D = [u(x), v(x)]$, $u(x)$ has degree $g$.
- For almost all pairs of polynomials $u$ and $v$ such that $u$ divides $v^2 + h v + f \mod u$, the polynomial $v$ has degree $\text{deg}(u) - 1$
- Almost all randomly chosen polynomials are relatively coprime.

These are standard assumptions which are made by nearly every author in the development of explicit formulas, beginning with Harley [9]. In the last three statements, *almost-all* can be interpreted as “all but proportion of size $O(g/g)$”. This means that if we concentrate on developing addition formulas which apply to the most general case (i.e. assuming that all polynomials have maximal degree and non-related polynomials are coprime) then only a negligible proportion of all group operations requires a different implementation. From the point of view of efficiency, we can handle all other cases with the general Cantor algorithm without having a noticeable impact on the computation of $[c]D$ (via a double-and-add approach), so only the general case is discussed here.

To reduce computational cost, we restrict ourselves to curves of the form

$$y^2 + y = x^9 + f_7 x^7 + f_5 x^5 + f_1 x + f_0$$

for genus four (the security of curves of these special forms is discussed in Section 5 in [3]). As already mentioned, we consider only the most common case of the addition and doubling formulas, i.e. when the degrees are maximal, and (for the addition formula) when $u_1$ and $u_2$ are coprime.

#### 2.1.1. Addition and doubling formulas

For the details of how the explicit formulas are obtained in affine coordinates, we refer to [3]. We only recall the resulting algorithms:
Algorithm 2 Group addition, most common case

**Inputs:** Reduced divisors $D_1 = [u_1(x), v_1(x)]$ and $D_2 = [u_2(x), v_2(x)]$

**Outputs:** Reduced divisor $D_3 = [u_3(x), v_3(x)]$, $D_3 = D_1 + D_2$

1. Almost inverse, $inv(x) = ru_1(x)^{-1} \mod u_2(x)$, via Cramer’s rule
2. $r = inv(x)u_1(x) \mod u_2(x)$
3. $s'(x) = rs(x)$ where $s(x) = u_1(x)^{-1}(u_2(x) + v_1(x)) \mod u_2(x)$
4. computation of inverses and $\exists (s'(x)$ made monic
5. $u_T(x) = \left[ \frac{\tilde{b}u_2(x)}{u_2(x)} \right] + s_3^2(x + a_3 + b_3)$
6. $z(x) = \tilde{s}(x)u_1(x)$
7. $v_T(x) = s_3z(x) + v_1(x) + 1 \mod u_T(x)$
8. $u_3(x) = Monic\left( \frac{f(x)v_T(x) + v_3^2(x)}{u_T(x)} \right)$
9. $v_3(x) = v_T(x) + 1 \mod u_3(x)$

Algorithm 3 Group doubling, most common case

**Inputs:** Reduced divisors $D_1 = [u_1(x), v_1(x)]$

**Outputs:** Reduced divisor $D_3 = [u_3(x), v_3(x)]$, $D_3 = 2D_1$

1. $u_C(x) = u_1(x)^2$
2. $v_C(x) = v_1(x)^2 + f(x) \mod u_C(x)$
3. computation of inverses
4. $u_T(x) = Monic\left( \frac{f(x) + v_C(x) + v_3^2(x)}{u_C(x)} \right)$
5. $v_T(x) = v_C(x) + 1 \mod u_T(x)$
6. $u_3(x) = Monic\left( \frac{f(x) + v_T(x) + v_3^2(x)}{v_T(x)} \right)$
7. $v_3(x) = v_T(x) + 1 \mod u_3(x)$

3. Projective coordinates

3.1. Inversion-free polynomial arithmetic. Given two polynomials $a(x)$ and $b(x)$, we consider computations of the form $\lfloor a(x)/b(x) \rfloor$, $a(x)^{-1} \mod b(x)$, and $\gcd(a(x), b(x))$. In all cases, the computation relies on a sequence of steps of the form

$r_i(x) = s_i(x)a(x) + t_i(x)b(x)$

For the computation of $\lfloor a(x)/b(x) \rfloor$, we restrict $s_i(x)$ to nonzero constants and proceed until $\deg(r_i(x)) < \deg(b(x))$, the result being $t_j(x)/s_j$. For the computation of $a^{-1}(x) \mod b(x)$, we want the value of $s_j(x)$ when $r_j(x)$ is a nonzero constant. In this case, the result $a^{-1}(x) \mod b(x)$ is given by $s_j(x)/r_j(x)$. For the gcd, we proceed until $r_j(x) = 0$ and the desired result is $\text{Monic}(r_{j-1}(x))$. The last two computations are based on the Euclidean algorithm and can be seen as a repetition of the division.

The idea of doing the group arithmetic in projective coordinates is to avoid field inversions (which can be considered too expensive in some contexts). One integral part of doing this is changing the definitions of various polynomials so they can be computed without using any field divisions.

Some of the polynomial operations are therefore replaced by “inversion-free” algorithms, in particular using the methods of Nagao [21]. Similar methods had previously been presented by Knuth [10] (as Pseudo-division of polynomials), however Nagao was the first to explicitly apply it to Cantor’s algorithm. To denote the results of these computations, which in fact return a (known) multiple of the true output,
we will use the notation $\langle \cdot \rangle$. For example, if $U_1(x) = \sum_{i=0}^{4} a_ix^i$, $U_2(x) = \sum_{i=0}^{4} b_ix^i$, the computation of $\left\langle \left[ \frac{U_1}{U_2} \right] \right\rangle$ would give

$$\left\langle \left[ \frac{U_1}{U_2} \right] \right\rangle = b_4 \left[ \frac{U_1}{U_2} \right]$$

since we do not permit the (necessary) division by $b_4$ (the leading coefficient of $U_2(x)$). We will refer to this multiplicative constant ($b_4$ in this example) as the “projective constant” of the operation.

For the computations of the polynomials $r_i(x)$, $s_i(x)$, and $t_i(x)$ in $r_i(x) = s_i(x)a(x) + t_i(x)b(x)$, we replace these polynomials by the computation of $\tilde{r}_i(x) = L_ir_i(x)$, $\tilde{s}_i(x) = L_is_i(x)$ and $\tilde{t}_i(x) = L_it_i(x)$ where $L_i$ is selected in such a way that the computations can be performed inversion-free. The value of $L_i$ and of the $a_j$ are given by the inversion-free algorithms and usually comes from the leading coefficients of the polynomials involved in that step.

For example, in computing the remainder of $b(x)$ modulo $a(x)$, both of equal degree, we would like to get $r(x) = 1 \cdot a(x) - \frac{\alpha}{\beta}b(x)$ where $\alpha$ is the leading coefficient of $a(x)$ and $\beta$ is the leading coefficient of $b(x)$. To do this inversion-free, we set $L = \beta$ and compute $\tilde{r}(x) = \beta a(x) - \alpha b(x)$. In this case, the result of the computation would be $\tilde{r}(x)$ and the projective constant would simply be $\beta$.

### 3.2. From Affine to Projective Addition Formula.

We now consider how to compute the sum of two divisors $D_1 = [u_1(x), v_1(x)]$ and $D_2 = [u_2(x), v_2(x)]$. Since we want to work in projective coordinates, we cannot assume that $D_1$ and $D_2$ will be obtained in affine coordinates (in general they will be the results of previous computations in projective coordinates), so we will replace the polynomials $u_i(x)$ and $v_i(x)$ given in the form:

$$U_1(x) = a_4u_1(x) = \sum_{i=0}^{4} a_i x^i, \quad V_1(x) = a_4v_1(x) = \sum_{i=0}^{3} c_i x^i,$$

$$U_2(x) = b_4u_2(x) = \sum_{i=0}^{4} b_i x^i, \quad V_2(x) = b_4v_2(x) = \sum_{i=0}^{3} d_i x^i,$$

The reduced divisors will then be presented as $D_1 = [U_1(x), V_1(x)]$ and $D_2 = [U_2(x), V_2(x)]$, and we want to compute the reduced divisor $D_3 = D_1 + D_2$, also in the form $[U_3(x), V_3(x)]$. Note that we could consider using a different multipliers for the polynomials $U_1$ and $V_1$ (i.e. write $V_1(x) = W_1 v_1(x)$ where $W_1$ is not related to $a_4$), however a careful study of the corresponding costs shows that the addition would be 1M more expensive, and the doubling would increase by 5M.

Adapting Algorithm 2 to inversion-free polynomial arithmetic gives us the following steps:

1: We compute $\langle U_1^{-1}(x) \mod U_2(x) \rangle$ via the inversion-free extended GCD.

2: This step is not performed in projective coordinates.

3: We want to compute $s(x) = (u_1^{-1}(x) \mod u_2(x))(v_1(x) + v_2(x)) \mod u_2(x)$.

We first need to define a suitable multiple of $v_1(x) + v_2(x)$ for the projective case, which will be

$$a_4b_4(v_1(x) + v_2(x)) = b_4 V_1(x) + a_4 V_2(x).$$

To compute $S(x) = \left\langle (U_1^{-1}(x) \mod U_2(x))(b_4 V_1(x) + a_4 V_2(x)) \mod U_2(x) \right\rangle$, we apply the inversion-free modular reduction. In Section 4.2, it will be shown that $S(x) = \varphi s(x)$ with $\varphi = b_4^2 (r_5(a_4 b_4))$. 

4. We want to compute a multiple of 
\[ u_T(x) = \text{Monic} \left( \frac{v_2^2(x) + h(x)v_1(x) + f(x)}{u_1(x)u_2(x)} + \frac{s^2(x)u_1(x)}{u_2(x)} + s(x) \right) \]
using the polynomials and operations available in projective coordinates.

5. We have to compute a multiple of 4:

We want to compute \(Adapting the computation of \)
\[ v_1^2(x) + v_1(x) + f(x) \]
\[ U_1(x)U_2(x) \]
using the polynomials and operations available in projective coordinates.

- Given that \(\deg(V_1) = 3\), \(\deg(f) = 9\), and \(\deg(U_1U_2) = 8\) we have 
\[ \left\langle \frac{V_1^2(x) + V_1(x) + f(x)}{U_1(x)U_2(x)} \right\rangle = \left\langle \frac{f(x)}{U_1(x)U_2(x)} \right\rangle \]
\[ = (a_4b_4)x + a_4b_3 + b_4a_3. \]

with projective constant \((a_4b_4)^2\).

- \[ \left[ \frac{S(x)}{U_2(x)} \right] = 0 \], since \(\deg(S(x)) < \deg(U_2(x)).\)

- In the computation of \[ \left[ \frac{S^2(x)U_1(x)}{U_2(x)} \right], \] even though \(S(x)U_1(x)\) will need to be computed completely at the next step, it is more efficient to compute \(S^2(x)\) first and then multiply by \(U_1(x)\). The sparsity of \(S^2(x)\) reduces the computations enough that it costs less than multiplying \(S(x)U_1(x)\) by \(S(x)\). In this argument, we must ignore the cost of computing \(S(x)U_1(x)\) since it will be computed either way.

We now define \(u_T(x)\) for the projective case:

\[ u_T(x) = \left[ \frac{u(x)}{u_1(x)u_2(x)} + \frac{s^2(x)u_1(x)}{u_2(x)} \right] \]
\[ = a_4b_4 \left( \frac{f(x)}{U_1(x)U_2(x)} + \frac{b_4}{a_4b_4} \right) \]
\[ = a_4b_4 \left( \frac{f(x)}{U_1(x)U_2(x)} \right) + \frac{b_4}{a_4b_4}. \]

Therefore, we can define:

\[ U_T(x) = \left\langle u_T(x) \right\rangle = \varphi^{\frac{2b_5}{a_4}} \left\langle \frac{f(x)}{U_1(x)U_2(x)} \right\rangle + \left\langle \frac{S^2(x)U_1(x)}{U_2(x)} \right\rangle. \]

Note that \(u_T(x) = \text{Monic}(U_T(x)),\) so the proportion between the two polynomials can be obtained directly from the leading term of \(U_T(x)\).

6. We want to compute \(v_T(x) = v_1(x) + s(x)u_1(x) + 1 \mod u_T(x).\) We first re-write the sum \(v_1(x) + s(x)u_1(x) + 1 \) for the projective case:

\[ v_1(x) + s(x)u_1(x) + 1 = \frac{V_1(x)}{a_4} + \frac{S(x)U_1(x)}{a_4} + 1 \]
\[ = \frac{\varphi V_1(x) + S(x)U_1(x)}{\varphi a_4}. \]

Computing \(V_T(x) = \left\langle (\varphi V_1(x) + S(x)U_1(x))^{a_4} \right\rangle \mod U_T(x)\). In Section 4.4, we will show that \(V_T(x) = KV_T(x)\) with \(K = L_0(\varphi a_4).\)

8. Adapting the computation of \(u_3 = \text{Monic}(\frac{f(x) + v_T(x) + v_T^2(x)}{u_T(x)})\) to projective coordinates, we first find 
\[ f(x) + v_T(x) + v_T^2(x) = \frac{K^2f(x) + KV_T(x) + V_T^2(x)}{K^2}. \]
Given that \( \deg(K^2f(x) + KV_T(x) + V_T^2(x)) = \deg(V_T^2(x)) = 10 \), and \( \deg(U_T(x)) = 6 \), we do not need to compute the coefficients of \( x^5 \) and smaller powers in \( x \) in \( K^2f(x) + V_T^2(x) \). We can then compute:

\[
U_3(x) = \langle u_3(x) \rangle = \left\langle \frac{K^2f(x) + V_T^2(x)}{U_T(x)} \right\rangle.
\]

The proportion between \( U_3(x) \) and \( u_3(x) \) is \( P_4 \), the leading term of \( U_3(x) \).

9: We want to compute \( v_3(x) = v_T(x) + 1 \mod u_3(x) \). In projective coordinates, this formula becomes:

\[
v_T(x) + 1 = \frac{V_T(x) + K}{K}.
\]

and we can replace \( v_3(x) \) by \( V_3(x) \) defined as

\[
V_3(x) = W_3v_3(x) = \left\langle V_T(x) + K \mod U_3(x) \right\rangle.
\]

where \( W_3 = P_4K \)

10: We have to adjust the common proportion between the polynomials \( V_3(x) \) and \( U_3(x) \), and we obtain: \( \tilde{U}_3(x) = (P_4K)U_3(x) \).

**Algorithm 4** Group addition projective coordinates, most common case

**Inputs:** Reduced divisors \( D_1 = [U_1(x), V_1(x)] \) and \( D_2 = [U_2(x), V_2(x)] \)

**Outputs:** Reduced divisor \( D_3 = [U_3(x), V_3(x)] \), \( D_3 = D_1 + D_2 \)

1: \( \left\langle U_1^{-1}(x) \mod U_2(x) \right\rangle \)
2: \( S(x) = \left\langle (U_1^{-1}(x) \mod U_2(x))(b_1V_1(x) + a_4V_2(x)) \mod U_2(x) \right\rangle \)
3: \( U_T(x) = v^2b_1^3 \left\langle \frac{f}{U_1(x)U_2(x)} \right\rangle + \left\langle \frac{s^2(x)U_1(x)}{U_2(x)} \right\rangle \)
4: \( V_T(x) = \left\langle (vV_1(x) + SU_1(x) + v^a_4) \mod U_T(x) \right\rangle \)
5: \( U_3(x) = \left\langle \frac{K^2f(x) + V_T^2(x)}{U_T(x)} \right\rangle \)
6: \( V_3(x) = \left\langle V_T(x) + K \mod U_3(x) \right\rangle \)
7: \( \tilde{U}_3(x) = (P_4K)U_3(x) \)

### 3.3. FROM AFFINE TO PROJECTIVE DOUBLING FORMULA.

From the reduced divisor \( D_1 = [U_1(x), V_1(x)] \), we want to compute \( D_3 = [U_3(x), V_3(x)] \) such that \( D_3 = 2D_1 \).

Adapting Algorithm 3 gives us the following steps:

1: We can define \( U_C(x) \) as \( U_1^2(x) = a_2^2 u_C(x) \).

2: For the computation of \( v_C(x) = v_1^2(x) + f(x) \mod u_C(x) \), we first note that \( \deg(v_1^2(x)) = 6 < \deg(U_C(x)) \), so we can perform the computation as:

\[
v_C(x) = (v_1^2(x) \mod U_C(x)) + (f(x) \mod U_C(x))
\]

\[
= v_1^2(x) + \frac{1}{r} \langle f(x) \mod U_C(x) \rangle
\]

\[
= \frac{1}{a_2^2} \langle V_1^2(x) \rangle + \frac{1}{r} \langle f(x) \mod U_C(x) \rangle
\]

Since \( f(x) \) is of the form \( x^9 + f(x)x^7 + \ldots \), and the leading term of \( U_C(x) \) is \( a_2^2 \), it is easily verified that the projective constant \( r \) is equal \( a_2^2 \) in this computation.
It is then natural to combine the polynomials as
\[ V_C(x) = V_1^2(x) + \left( \frac{f(x)}{u_C(x)} \right) \mod U_C(x) \]
with \( V_C(x) = a_4^2 \cdot v_C(x) \).

3: This step is not performed in projective coordinates (we want to eliminate all field inversions).

4: We have to compute \( u_T(x) = \text{Monic} \left( \frac{f(x)+v_C(x)+v_C^2(x)}{u_C(x)} \right) \), from \( V_C(x) \) and \( U_C(x) \) and without using any field inversions. Given that \( \deg(U_C(x)) = 8 \) and \( \deg(V_C(x)) = 7 \), this can be computed as \( \text{Monic} \left( \frac{f(x)+v_C^2(x)}{u_C(x)} \right) \), which becomes (in projective coordinates):
\[
\begin{align*}
u_T(x) &= \text{Monic} \left( \frac{f(x)}{u_C(x)} + \left( \frac{v_C(x)}{u_1(x)} \right)^2 \right) \\
&= \text{Monic} \left( a_4^4 \left( \frac{f(x)}{U_C(x)} \right) + \left( \frac{V_C(x)}{U_1(x)} \right)^2 \right) \\
&= \text{Monic} \left( a_4^4 \left( \frac{f(x)}{U_C(x)} \right) + \left( \frac{V_C(x)}{U_1(x)} \right)^2 \right)
\end{align*}
\]
Therefore, we can define \( U_T(x) \) as
\[ U_T(x) = \alpha_6 u_T(x) = a_4^{10} \left( \frac{f(x)}{U_C(x)} \right) + \left( \frac{V_C(x)}{U_1(x)} \right)^2, \]
where \( \alpha_6 \) is the coefficient of \( x^6 \) in \( U_T(x) \).

5: We want to compute \( v_T(x) = v_C(x) + 1 \mod u_T(x) \) from \( V_C(x) \) and \( U_T(x) \). To do this, we first replace \( v_C(x) + 1 \) by
\[
a_4^2 \left( v_C(x) + 1 \right) = V_C(x) + a_4^2,
\]
and then compute
\[
\begin{align*}
V_T(x) &= \left( V_C(x) + a_4^2 \right) \mod U_T(x) \\
&= \alpha_6 \left( V_C(x) + a_4^2 \mod U_T(x) \right),
\end{align*}
\]
which is \( \delta v_T(x) \) with \( \delta = \alpha_6 a_4^2 \).

6: The computation of \( u_3(x) = \text{Monic} \left( \frac{f(x)+v_T(x)+v_T^2(x)}{u_T(x)} \right) \) form \( V_T(x) \) and \( U_T(x) \) proceeds in much the same way as Step 4:
\[
\begin{align*}
u_3(x) &= \text{Monic} \left( \frac{f(x)}{u_T(x)} + \left( \frac{v_T^2(x)}{u_T(x)} \right) \right) \\
&= \text{Monic} \left( \frac{f(x)}{U_T(x)} + \frac{\alpha_6}{\delta^2} \left( \frac{v_T^2(x)}{U_T(x)} \right) \right) \\
&= \text{Monic} \left( \frac{f(x)}{\alpha_6^6} + \frac{\alpha_6}{\delta^2} \left( \frac{v_T^2(x)}{\alpha_6^6} \right) \right),
\end{align*}
\]
and we define \( U_3(x) = \alpha_6 \delta^2 \left( \left\lfloor \frac{f(x)}{U_T(x)} \right\rfloor \right) + \left( \left\lfloor \frac{V_2(x)}{U_T(x)} \right\rfloor \right) \), so it will be equal to \( \psi_4 v_3(x) \) where \( \psi_4 \) is the leading coefficient of \( U_3(x) \).

7: If we followed the same idea as Step 5, we would define

\[ V_3(x) = \langle V_T(x) + \delta \mod U_3(x) \rangle \]

which is equal to \( W_3 v_3(x) = (\psi_3^2 \delta) v_3(x) \). However, a few simple algebraic manipulations show that we can extract a factor of \( \alpha_3^2 T_5 \) (where \( T_5 \) is the leading coefficient of \( V_T(x) \)), so we will really compute

\[ \tilde{V}_3(x) = \frac{1}{\alpha_3^2 T_5} V_3(x) = (\psi_4 T_5 \delta) v_3(x) \]

(the details for this substitution can be found in Section 5.6).

8: We have to adjust the common proportion between the polynomials \( U_3(x) \) and \( V_3(x) \), and we obtain: \( \tilde{U}_3(x) = (T_5 \delta) U_3(x) \).

**Algorithm 5** Group doubling projective coordinates, most common case

**Inputs:** Reduced divisors \( D_1 = [U_1(x), V_1(x)] \)

**Outputs:** Reduced divisor \( D_3 = [U_3(x), V_3(x)] \), \( D_3 = 2D_1 \)

1: \( U_C(x) = U_1^2(x) \)

2: \( V_C(x) = V_1^2(x) + \left\langle f(x) \mod U_C(x) \right\rangle \)

3: \( U_T(x) = \alpha_4^{10} \left\langle \left\lfloor \frac{f(x)}{U_C(x)} \right\rfloor \right\rangle + \left( \left\lfloor \frac{V_C(x)}{V_1(x)} \right\rfloor \right)^2 \)

4: \( V_T(x) = \langle (V_C(x) + \alpha_4^2 \mod U_T(x) \rangle \)

5: \( U_3(x) = \alpha_6 \delta^2 \left( \left\lfloor \frac{f(x)}{U_T(x)} \right\rfloor \right) + \left( \left\lfloor \frac{V_2(x)}{U_T(x)} \right\rfloor \right) \)

6: \( V_3(x) = V_T(x) + \delta \mod U_3(x) \)

7: \( \tilde{U}_3(x) = (T_5 \delta) U_3(x) \)

3.4. **Reducing the number of multiplications.** Although Karatsuba-like multiplications are most commonly applied to polynomials multiplication, they can also be used when dealing with polynomial divisions (both for the quotient and the remainder). In the detailed description of each step, to keep the equations easier to read (and simplify the operation count), we will denote with curly brackets \{·\} the two products that are combined into one using Karatsuba’s technique and by \( \lfloor · \rfloor \) the two other terms. For example, we would write

\[ (ax + b)(cx + d) = [ac]x^2 + \{ad + bc\}x + [bd] \]

instead of

\[ (ax + b)(cx + d) = acx^2 + ((a + b)(c + d) + ac + bd)x + bd \]

If two layers of Karatsuba multiplication are used, as in the multiplication of two polynomials of degree 3, we use double curly brackets \{{·}\} for the combination of the 4 central terms.

3.5. **Avoiding repeated multiplications.** In the explicit computations, many products (of two or more terms) appear more than once. Obviously, we want to re-use the result of the multiplications that were performed previously, and avoid counting them twice in the number of operations. To differentiate the re-use of previous multiplications from a new one, we usually put the old product between parentheses.
4. Explicit formulas improvement for genus 4 HECs addition case

We consider the divisors \( D_1 = [U_1(x), V_1(x)] \) and \( D_2 = [U_2(x), V_2(x)] \), which are the projective representations of divisors \( D_1 = [u_1(x), v_1(x)] \) and \( D_2 = [u_2(x), v_2(x)] \).

4.1. Computing \( \langle U_1^{-1}(x) \mod U_2(x) \rangle \). Starting from \( r_0 = U_1(x) = s_0(x)u_1(x) + t_0(x)u_2(x) \) and \( r_1 = U_2(x) = s_1(x)u_1(x) + t_1(x)u_2(x) \) (i.e. with \( s_0(x) = a_4 \) and \( s_1(x) = 0 \)), we perform the inversion-free Euclidean algorithm. At the first step, we have

\[
r_2 = b_4 r_0 + a_4 r_1 = L_0 r_0 + \bar{q}_1(x) r_1
\]

which we write as \( r_2 = A_3 x^3 + A_2 x^2 + A_1 x + A_0 \), with

\[
A_3 = b_4 a_3 + a_4 b_3, \quad A_2 = b_4 a_2 + a_4 b_2, \quad A_1 = b_4 a_1 + a_4 b_1, \quad A_0 = b_4 a_0 + a_4 b_0.
\]

Moreover, \( s_2(x) = L_0 s_0(x) + \bar{q}_1(x) s_1(x) = a_4 b_4 \).

For the second step, we apply the modular reduction, and write

\[
r_3 = A_3^2 r_1 + (A_3 b_4 x + A_6) r_2 = L_1 r_1 + \bar{q}_2(x) r_2
\]

as \( r_3 = A_3 x^2 + A_8 x + A_7 \), where we get:

\[
A_6 = A_3 b_3 + b_4 A_2, \quad A_7 = A_3^2 b_0 + [A_6 A_0],
\]

\[
A_8 = A_3^2 b_1 + \{(A_3 b_4) A_0 + A_6 A_1\}, \quad A_9 = A_3^2 b_2 + [(A_3 b_4) A_1] + A_6 A_2.
\]

We also obtain \( s_3 = L_1 s_1(x) + \bar{q}_2(x) s_2(x) = (a_4 b_4) (A_3 b_4) x + A_6 (a_4 b_4) \), which we will denote \( s_3 = C x + D \).

At the third gcd step, we have

\[
r_4 = A_3^2 r_2 + ((A_3 A_4) x + A_{11}) r_3 = L_2 r_2 + \bar{q}_3(x) r_3,
\]

written as \( r_4 = A_{13} x + A_{12} \), with

\[
A = A_3 A_4, \quad A_{11} = A_4 A_2 + A_3 A_8
\]

\[
A_{12} = A_3^2 A_0 + A_{11} A_7, \quad A_{13} = A_3^2 A_1 + A A_7 + A_{11} A_8.
\]

We also compute \( s_4 = L_2 s_2(x) + \bar{q}_3(x) s_3 = A_3^2 (a_4 b_4) + \{(A x + A_{11}) (C x + D)\} \) (using a \( 2 \times 2 \) Karatsuba multiplication), which will be written as \( s_4 = F x^2 + G x + I \).

The final step of the Euclidean algorithm gives us

\[
r_5 = A_3^2 r_3 + (A_3 A_4 x + A_{15}) r_4 = L_3 r_3 + \bar{q}_4(x) r_4,
\]

where \( A_{15} = A_{13} A_8 + A_0 A_{12} \) and \( r_5 = A_{13} A_7 + A_{15} A_{12} \). Since \( L_3 = A_3^2 \), and \( \bar{q}_4(x) = E x + A_{15} \) (with \( E = A_{13} A_9 \)), we obtain

\[
s_5 = L_3 s_3 + \bar{q}_4 s_4 = A_3^2 (C x + D) + \{(E x + A_{15}) (F x^2 + G x + I)\},
\]

which is computed using a \( 2 \times 3 \) Karatsuba multiplication. Since \( r_5 \equiv s_5 U_1(x) \mod U_2(x) \), we can use the polynomial \( s_5 \) as \( U_1^{-1} \mod U_2 = \sum_{i=0}^{3} b_i x^i \), with a projective constant \( r_5 \), i.e. \( s_5(x) = r_5(u_1^{-1} \mod u_2) \).

4.2. Computing \( S(x) = \left( (U_1^{-1}(x) \mod U_2(x))(b_4 V_1(x) + a_4 V_2(x)) \mod U_2(x) \right) \).

4.2.1. Computing \( v_1(x) + v_2(x) \) in the projective case. We compute \( b_4 V_1(x) + a_4 V_2(x) = \sum_{i=0}^{3} E_i x^i \) with \( E_i = b_4 c_i + a_4 d_i \), for \( i = 0, \ldots, 3 \).
Algorithm 6 Group addition in projective coordinates, Step 1

1: $A_3 \leftarrow b_4 \cdot a_3 + a_4 \cdot b_3$, $A_2 \leftarrow b_4 \cdot a_2 + a_4 \cdot b_2$, $A_1 \leftarrow b_4 \cdot a_1 + a_4 \cdot b_1$,
2: $A_0 \leftarrow b_4 \cdot a_0 + a_4 \cdot b_0$, $ab_4 \leftarrow a_4 \cdot b_4$, $A_6 \leftarrow A_3 \cdot b_3 + b_4 \cdot A_2$, $Aux_1 \leftarrow A_3 \cdot b_4$,
3: $k_{u_1} \leftarrow Aux_1 \cdot A_1$, $k_{u_2} \leftarrow A_6 \cdot A_0$, $k_{v_1} \leftarrow (Aux_1 + A_1) \cdot (A_0 + A_1)$,
4: $A_{32} \leftarrow A_2^2$, $A_6 \leftarrow A_{32} \cdot b_2 + k_{u_1} + A_6 \cdot A_2$, $A_8 \leftarrow A_{32} \cdot b_1 + k_{v_1} + A_{32} + k_{v_1}$,
5: $A_7 \leftarrow A_{32} \cdot b_0 + k_{c_1}$, $C \leftarrow Aux_1 \cdot ab_4$, $D \leftarrow A_6 \cdot ab_4$, $A_{11} \leftarrow A_9 \cdot A_2 + A_3 \cdot A_8$,
6: $A_{92} \leftarrow A_0^2$, $A_{12} \leftarrow A_{92} \cdot A_0 + A_{11} \cdot A_7$, $A \leftarrow A_9 \cdot A_3$,
7: $A_{13} \leftarrow A_{92} \cdot A_1 + A \cdot A_7 + A_{11} \cdot A_8$, $F \leftarrow A \cdot C$, $k_{b_2} \leftarrow (A + A_{11}) \cdot (C + D)$,
8: $k_{c_2} \leftarrow A_{11} \cdot D$, $G \leftarrow F + k_{b_2} + k_{c_2}$, $I \leftarrow A_{92} \cdot ab_4 + k_{c_2}$,
9: $A_{15} \leftarrow A_{13} \cdot A_9 + A_3 \cdot A_{12}$, $L_5 \leftarrow A_{13}^2$, $r_5 \leftarrow L_3 \cdot A_7 + A_{15} \cdot A_{12}$,
10: $E \leftarrow A_{13} \cdot A_9$, $k_{a_3} \leftarrow E \cdot G$, $k_{b_3} \leftarrow (E + A_{15}) \cdot (G + I)$, $k_{c_3} \leftarrow A_{15} \cdot I$,
11: $\delta_3 \leftarrow E \cdot F$, $\delta_2 \leftarrow A_{15} \cdot F + k_{a_3}$, $\delta_1 \leftarrow L_3 \cdot C + k_{b_3} + k_{a_3} + k_{c_3}$,
12: $\delta_0 \leftarrow L_3 \cdot D + k_{c_3}$
13: \[ 45M+3S \]

4.2.2. Computing \((u_1^{-1}(x) \bmod u_2(x))(v_1(x) + v_2(x))\). We want to compute

\[
G(x) = (U_1^{-1} \bmod U_2)(b_4V_1 + a_4V_2)
\]

\[
= (\delta_3 x^3 + \delta_2 x^2 + \delta_1 x + \delta_0)(E_3 x^3 + E_2 x^2 + E_1 x + E_0)
\]

\[
= G_6 x^6 + G_5 x^5 + G_4 x^4 + G_3 x^3 + G_2 x^2 + G_1 x + G_0.
\]

Using Karatsuba’s multiplication method, this can be done in 9 multiplications.

Note that

\[
G(x) = (r_5 a_4 b_4)(u_1^{-1}(x) \bmod u_2(x))(v_1(x) + v_2(x)).
\]

4.2.3. Computing \(s(x) = (u_1^{-1}(x) \bmod u_2(x))(v_1(x) + v_2(x)) \bmod u_2(x)\). Starting from \(r_0(x) = G(x)\) and \(r_1(x) = U_2(x)\), we apply the inversion-free reduction technique, giving us the following iteration steps:

\[
r_2 = b_4 r_0 + G_6 x^2 r_1, \quad r_3 = b_4 r_2 + A_{10} x r_1, \quad r_4 = b_4 r_3 + A_{14} r_1.
\]

The final projective constant is \(D = b_4^2\) and \(S(x) = r_4(x) = \sum_{i=0}^{3} H_i x^i\) with

\[
A_{10} = b_4 G_5 + \lfloor G_6 b_3 \rfloor, \quad A_{14} = b_4^2 G_4 + \lfloor (b_4 b_2) G_6 + A_{10} b_3 \rfloor, \quad H_3 = b_4 G_3 + \lfloor b_4 (G_6 b_1) \rfloor + \lfloor (b_4 b_2) A_{10} \rfloor + A_{14} b_3,
\]

\[
H_2 = b_4^2 G_2 + \lfloor (b_4^2 G_6) b_0 \rfloor + (b_4 A_{10}) b_1 \rfloor + A_{14} b_2, \quad H_1 = b_4 G_1 + \lfloor (b_4 A_{10}) b_0 \rfloor + A_{14} b_1,
\]

\[
H_0 = b_4^2 G_0 + A_{14} b_0,
\]

and the projective constant is calculated as:

\[
S(x) = b_4^2 (r_5 (a_4 b_4))(u_1^{-1}(x) \bmod u_2(x))(v_1(x) + v_2(x)) \bmod u_2(x),
\]

which we denote by \(\phi = b_4^2 (r_5 (a_4 b_4))\).

4.3. Computing \(U_T(x) = \phi^2 b_4^2 \left\langle \frac{f}{U_2^2} \right\rangle + \left\langle \frac{S^2 V_1}{U_2^2} \right\rangle\).

1. At the first step, we need to compute \(\left\langle \frac{f}{U_2^2} \right\rangle = a_4 b_4 x + A_3\), which gives a projective constant of \(D = (a_4 b_4)^2\).
Algorithm 7 Group addition in projective coordinates, Step 2

1: $E_5 \leftarrow b_4 \cdot c_3 + a_4 \cdot d_3$, $E_2 \leftarrow b_4 \cdot c_2 + a_4 \cdot d_2$, $E_1 \leftarrow b_4 \cdot c_1 + a_4 \cdot d_1$, 
2: $E_0 \leftarrow b_4 \cdot c_0 + a_4 \cdot d_0$, $k_{a4} \leftarrow \delta_3 \cdot E_3$, $k_{a5} \leftarrow \delta_0 \cdot E_0$, $k_{c4} \leftarrow \delta_2 \cdot E_2$, 
3: $k_{c5} \leftarrow \delta_1 \cdot E_1$, $k_{a4} \leftarrow (\delta_3 + \delta_2) \cdot (E_2 + E_3)$, $k_{b4} \leftarrow (\delta_3 + \delta_1) \cdot (E_1 + E_3)$, 
4: $k_{b5} \leftarrow (\delta_2 + \delta_0) \cdot (E_0 + E_2)$, $k_{b6} \leftarrow (\delta_1 + \delta_0) \cdot (E_0 + E_1)$, 
5: $k_{d4} \leftarrow (\delta_3 + \delta_2 + \delta_1 + \delta_0) \cdot (E_0 + E_1 + E_2 + E_3)$, $G_5 \leftarrow k_{b4} + k_{a4} + k_{c4}$, 
6: $G_4 \leftarrow k_{b5} + k_{a4} + k_{c5} + k_{c4}$, $G_2 \leftarrow k_{b0} + k_{c5} + k_{a0} + k_{c4}$, $G_1 \leftarrow k_{b7} + k_{c5} + k_{a5}$, 
7: $G_3 \leftarrow k_{d4} + k_{a4} + G_5 + G_4 + G_2 + G_1 + k_{a5}$, $Axu2 \leftarrow b_4 \cdot b_2$, $b_{42} \leftarrow b_{11}$, 
8: $Axu3 \leftarrow b_{42} \cdot k_{a4} + k_{c6} \leftarrow k_{a4} \cdot b_3$, $A_{10} \leftarrow b_4 \cdot G_5 + k_{c0}$, $Axu4 \leftarrow b_4 \cdot A_{10}$, 
9: $k_{a6} \leftarrow Axu2 \cdot A_{10}$, $k_{a7} \leftarrow Axu3 \cdot b_1$, $k_{c7} \leftarrow Axu4 \cdot b_0$, 
10: $k_{b6} \leftarrow (Axu2 + b_3) \cdot (k_{a4} + A_{10})$, $k_{b0} \leftarrow (Axu3 + Axu4) \cdot (b_0 + b_1)$, 
11: $A_{14} \leftarrow b_{42} \cdot G_4 + k_{b6} + k_{a6} + k_{c6}$, $b_{43} \leftarrow b_{42} \cdot b_4$, 
12: $H_3 \leftarrow b_{43} \cdot G_3 + k_{a7} + k_{a6} + A_{14} \cdot b_3$, $H_2 \leftarrow b_{43} \cdot G_2 + k_{b0} + k_{a7} + k_{c7} + A_{14} \cdot b_2$, 
13: $H_1 \leftarrow b_{43} \cdot G_1 + k_{c7} + A_{14} \cdot b_1$, $H_0 \leftarrow b_{43} \cdot k_{a5} + A_{14} \cdot b_0$, $\emptyset \leftarrow b_{43} \cdot r_5 \cdot ab_4$ 
14: $\triangleright 39M+1S$

2. For the second expression $\left[ \frac{S^2U_1}{U_2} \right]$, and since we are working over a field of characteristic 2, we get

$$S^2(x) = S_6 x^6 + S_4 x^4 + S_2 x^2 + S_0 = H_3^2 x^6 + H_2^2 x^4 + H_1^2 x^2 + H_0^2.$$ 

We then define $S^2(x)U_1(x)$ as $\sum_{i=1}^{10} K_i x^i$, where:

- $K_4 = S_6 a_2 + S_4 a_0$, $K_5 = [S_4 a_1] + S_2 a_3$,
- $K_6 = [S_6 a_0] + S_2 a_4$, $K_7 = [S_6 a_1 + S_4 a_3]$,
- $K_8 = [S_6 a_2 + S_4 a_4]$, $K_9 = [S_6 a_3]$,
- $K_{10} = [S_6 a_4]$, $[S_2 a_0]$ (extra mult.).

We can apply the inversion-free division algorithm to calculate $\left[ \frac{S^2U_1}{U_2} \right]$, and we obtain:

$$\frac{S^2U_1}{U_2} = b_4^3 K_{10} x^6 + b_4^2 E_3 x^5 + b_4^2 E_7 x^4 + b_4^2 E_{11} x^3 + b_4^2 E_{15} x^2 + b_4 E_{19} x + E_{23},$$

with a projective constant of $D = (b_4)^7$, and:

- $E_3 = b_4 K_9 + [K_{10} b_3]$,
- $E_7 = b_4 K_8 + \{K_{10} (b_4 b_2) + E_3 b_3\}$,
- $E_{11} = b_4^2 K_7 + \{K_{10} (b_4^2 b_1) + E_7 b_3\} + [E_3 (b_4 b_2)]$,
- $E_{15} = b_4^3 K_6 + \{[K_{10} (b_4^3 b_0) + E_3 (b_4^2 b_1) + E_7 (b_4 b_2) + E_{11} b_3]\}$,
- $E_{19} = b_4^2 K_5 + \{E_3 (b_4^2 b_0) + E_{11} (b_4 b_2)\} + [E_7 (b_4 b_1)] + E_{15} b_3$,
- $E_{23} = b_4^3 K_4 + \{E_7 (b_4^3 b_0) + E_{11} (b_4^2 b_1)\} + E_{15} (b_4 b_2) + E_{19} b_3$.

In Step 5 of 3.2, we defined $U_T(x)$ as $\left( \emptyset \cdot b_4^5 \left[ \frac{S^2U_1}{U_2} \right] \right) + \left( \frac{S^2U_1}{U_2} \right) = \sum_{i=0}^{10} L_i x^i$, where the $L_i$'s are computed as:

- $L_0 = (\emptyset \cdot b_4^5) A_3 + E_{23}$,
- $L_1 = b_4 (E_{19} + a_4 (\emptyset \cdot b_4^5))$,
- $L_2 = b_4^2 E_{15}$,
- $L_3 = b_4^2 E_{11}$,
- $L_4 = b_4^2 E_7$,
- $L_5 = b_4^2 E_3$,
- $L_6 = b_4^2 K_{10}$.

Advances in Mathematics of Communications Volume 4, No. 2 (2010), 115–139
Algorithm 8 Group addition in projective coordinates, Step 3

1: \(S_0 \leftarrow H_3^2, S_1 \leftarrow H_3^2, S_2 \leftarrow H_3^2, S_0 \leftarrow H_0^5, a_k \leftarrow S_6 \cdot a_4, a_{k_0} \leftarrow S_6 \cdot a_3,\)
2: \(k_{c_a} \leftarrow S_4 \cdot a_2, k_{c_b} \leftarrow S_4 \cdot a_1, k_{c_0} \leftarrow S_4 \cdot a_0, k_{k_0} \leftarrow (S_6 + S_4) \cdot (a_2 + a_4),\)
3: \(k_{b_1} \leftarrow (S_6 + S_4) \cdot (a_1 + a_3), k_{b_2} \leftarrow (S_6 + S_2) \cdot (a_0 + a_4),\)
4: \(k_{b_3} \leftarrow (S_2 + S_1) \cdot (a_2 + a_0), K_8 \leftarrow k_{b_0} + k_{a_0} + k_{c_0}, K_7 \leftarrow k_{b_1} + k_{a_0} + k_{c_0},\)
5: \(K_2 \leftarrow k_{b_1} + k_{a_0} + k_{c_10} + k_{c_0}, K_5 \leftarrow k_{c_0} + S_2 \cdot a_3, K_4 \leftarrow S_0 \cdot a_4 + k_{b_13} + k_{c_10} + k_{c_0},\)

6: \(k_{a_{10}} \leftarrow k_{a_0} \cdot b_0, E_5 \leftarrow b_4 \cdot k_{a_0} + k_{a_{10}}, Aux_7 \leftarrow b_{42} \cdot b_1, Aux_8 \leftarrow b_{43} \cdot b_0,\)
7: \(c_{11} \leftarrow E_3 \cdot Aux_2, k_{b_14} \leftarrow (k_{a_0} + E_3) \cdot (Aux_2 + b_3),\)
8: \(E_7 \leftarrow b_{42} \cdot K_8 + k_{b_14} + k_{a_{10}} + k_{c_{11}}, k_{a_{11}} \leftarrow E_7 \cdot Aux_7,\)
9: \(k_{b_{15}} \leftarrow (k_{a_0} + E_7) \cdot (Aux_7 + b_3), E_{11} \leftarrow b_{43} \cdot K_7 + k_{b_{15}} + k_{a_{10}} + k_{a_{11}} + k_{c_{11}},\)
10: \(k_{a_{12}} \leftarrow E_{11} \cdot Aux_8, k_{b_{16}} \leftarrow (E_3 + E_{11}) \cdot (Aux_8 + Aux_7),\)
11: \(E_7 \leftarrow E_7 + E_{11} \cdot (Aux_8 + Aux_7),\)
12: \(k_{d_2} \leftarrow (k_{a_0} + E_7 + E_{11}) \cdot (Aux_8 + Aux_7 + Aux_2 + b_3), b_{44} \leftarrow b_{42}^2,\)
13: \(E_{15} \leftarrow b_{44} \cdot K_6 + k_{d_2} + k_{b_{14}} + k_{c_{11}} + k_{b_{15}} + k_{a_{10}} + k_{a_{11}} + k_{b_{16}} + k_{b_{17}} + k_{b_{18}},\)
14: \(b_{45} \leftarrow b_{44} \cdot b_1, E_{19} \leftarrow b_{45} \cdot K_5 + k_{b_{16}} + k_{c_{11}} + k_{b_{15}} + k_{a_{11}} + E_{15} \cdot b_3, b_{46} \leftarrow b_{43},\)
15: \(E_{23} \leftarrow b_{46} \cdot K_4 + k_{b_{17}} + k_{a_{12}} + k_{b_{18}} + E_{15} \cdot Aux_2 + E_{19} \cdot b_3, Aux_{18} \leftarrow \varphi \cdot b_{45},\)
16: \(L_6 \leftarrow b_{46} \cdot k_{a_0}, L_5 \leftarrow b_{45} \cdot E_{17}, L_4 \leftarrow b_{44} \cdot E_7, L_3 \leftarrow b_{43} \cdot E_{11}, L_2 \leftarrow b_{42} \cdot E_{15},\)
17: \(L_1 \leftarrow b_6 \cdot (E_{19} + a_4 \cdot Aux_{18}), L_0 \leftarrow Aux_{18} \cdot A_3 + E_{23}\)
18: \(\triangleright 41M+7S\)

4.4. Computing \(V_T(x) = \left( (\varphi V_1(x) + S(x)U_1(x) + \varphi a_4) \mod U_T(x) \right)^\prime.\) We first compute \(\varphi V_1(x) + S(x)U_1(x) + \varphi a_4\) as

\[
\varphi \left( \sum_{i=0}^{3} c_i x^i \right) + \left( \sum_{i=0}^{3} H_i x^i \right) \left( \sum_{i=0}^{4} a_i x^i \right) + \varphi a_4
\]

where we use Karatsuba techniques for the polynomial multiplications. Writing \(\varphi V_1 + SU_1 + \varphi a_4 = \sum_{i=0}^{7} M_i x^i\), we now want

\[
\sum_{i=0}^{7} M_i x^i \mod U_T.
\]

For this computation we apply the inversion-free modular reduction, with iterative steps:

1. \(r_2(x) = L_6(x) + \sum_{i=0}^{7} M_i x^i + M_T x U_T(x)\),
2. \(\hat{V}_T(x) = L_6 r_2(x) + \eta_6 U_T(x),\)

where

\[
\eta_6 = L_6 M_6 + M_T L_5.
\]

This reduction has a projective constant of \(D = L_6^2\). We write \(V_T(x)\) as \(\sum_{i=0}^{5} N_i x^i\) where the coefficients \(N_i\) are:

\[
\begin{align*}
N_5 &= L_6^2 M_5 + (L_6 M_T) L_4 + \eta_6 L_5, \\
N_4 &= L_6^2 M_4 + [(L_6 M_T) L_3] + \eta_6 L_4, \\
N_3 &= L_6^2 M_3 + \{(L_6 M_T) L_2 + \eta_6 L_3\}, \\
N_2 &= L_6^2 M_2 + [(L_6 M_T) L_1] + [\eta_6 L_2], \\
N_1 &= L_6^2 M_1 + \{(L_6 M_T) L_0 + \eta_6 L_1\}, \\
N_0 &= L_6^2 M_0 + [\eta_6 L_0].
\end{align*}
\]

The proportion between \(V_T(x)\) and \(v_T(x)\) is therefore \(L_6^2 (\varphi a_4)\).
Algorithm 9 Group addition in projective/mixed coordinates, Step 4

1: \( k_{a13} \leftarrow H_3 \cdot a_3, \quad k_{a14} \leftarrow H_0 \cdot a_0, \quad k_{c12} \leftarrow H_2 \cdot a_2, \quad k_{c13} \leftarrow H_1 \cdot a_1, \)
2: \( k_{b18} \leftarrow (H_3 + H_2) \cdot (a_3 + a_2), \quad k_{b19} \leftarrow (H_3 + H_1) \cdot (a_3 + a_1), \)
3: \( k_{b20} \leftarrow (H_2 + H_0) \cdot (a_2 + a_0), \quad k_{b21} \leftarrow (H_1 + H_0) \cdot (a_1 + a_0), \)
4: \( k_5 \leftarrow (H_3 + H_2 + H_1 + H_0) \cdot (a_0 + a_1 + a_2 + a_3), \quad \text{Aux13} \leftarrow \phi \cdot a_4, \)
5: \( M_7 \leftarrow H_3 \cdot a_4, \quad M_6 \leftarrow H_2 \cdot a_4 + k_{a13}, \quad M_5 \leftarrow H_1 \cdot a_4 + k_{b18} + k_{a13} + k_{c12}, \)
6: \( M_4 \leftarrow H_0 \cdot a_4 + k_{c12} + k_{b19} + k_{a13} + k_{c13}, \)
7: \( M_3 \leftarrow \phi \cdot c_3 + k_{d3} + k_{b18} + k_{b19} + k_{a13} + k_{b20} + k_{c12} + k_{a14} + k_{b21} + k_{c13}, \)
8: \( M_2 \leftarrow \phi \cdot c_2 + k_{c13} + k_{b20} + k_{c12} + k_{a14}, \quad M_1 \leftarrow \phi \cdot c_1 + k_{b21} + k_{c13} + k_{a14}, \)
9: \( M_0 \leftarrow \phi \cdot c_0 + k_{a14} + \text{Aux13}, \quad \text{Aux14} \leftarrow L_6 \cdot M_7, \quad \eta_6 \leftarrow L_6 \cdot M_6 + M_7 + L_5, \)
10: \( k_{a15} \leftarrow \text{Aux14} \cdot L_3, \quad k_{a16} \leftarrow \text{Aux14} \cdot L_1, \quad k_{c14} \leftarrow \eta_6 \cdot L_2, \quad k_{c15} \leftarrow \eta_6 \cdot L_0, \)
11: \( k_{b22} \leftarrow (\text{Aux14} + \eta_6) \cdot (L_2 + L_4), \quad k_{b23} \leftarrow (\text{Aux14} + \eta_6) \cdot (L_0 + L_1), \quad L_{62} \leftarrow L_{60}, \)
12: \( N_5 \leftarrow L_{62} \cdot M_5 + \text{Aux14} \cdot L_4 + \eta_6 \cdot L_5, \quad N_4 \leftarrow L_{62} \cdot M_4 + k_{a15} + \eta_6 \cdot L_4, \)
13: \( N_3 \leftarrow L_{62} \cdot M_{3} + k_{b22} + k_{a16} + k_{c14}, \quad N_2 \leftarrow L_{62} \cdot M_{2} + k_{a16} + k_{c14}, \)
14: \( N_1 \leftarrow L_{62} \cdot M_{1} + k_{b23} + k_{a16} + k_{c15}, \quad N_0 \leftarrow L_{62} \cdot M_{0} + k_{c15}, \quad K \leftarrow L_{62} \cdot \text{Aux13} \)
15: \( \triangleright 37M + 18 \)

4.5. Computing \( U_3(x) = \left\langle \frac{K^2f(x) + V_2(x)}{U_T(x)} \right\rangle \). First we need to compute \( K^2f(x) + V_2(x) = N_5^2x^{10} + K^2x^9 + N_4^2x^8 + K^2f_7x^7 + N_2^2x^6 \).

We define \( r_0 = K^2f(x) + V_2(x) \), and \( r_1 = U_T(x) \), and compute \( U_3(x) = \left\langle \frac{r_i}{r_{i-1}} \right\rangle \) iteratively, using the steps:

\[
\begin{align*}
r_2 &= L_6r_0 + N_5^2x^4r_1, \\
r_3 &= L_6r_2 + \Theta_4x^3r_1, \\
r_4 &= L_6r_3 + \Theta_9x^2r_1, \\
r_5 &= L_6r_4 + \Theta_{14}xr_1, \\
r_6 &= L_6r_5 + \Theta_{19}r_1.
\end{align*}
\]

We obtain \( U_3(x) = \sum_{i=0}^{4} P_i x^i = L_6^4N_5^2x^4 + L_6^3\Theta_4x^3 + L_6^2\Theta_9x^2 + L_6\Theta_{14}x + \Theta_{19} \),

where the \( \Theta_i \) are given by:

\[
\begin{align*}
\Theta_4 &= L_6K^2 + N_5^2L_5, \\
\Theta_9 &= L_6^2N_5^2 + (L_6L_4)N_5^2 + \Theta_4L_5, \\
\Theta_{14} &= L_6^3K^2f_7 + \{(L_6L_3)N_5^2 + (L_6L_4)\Theta_4\} + \Theta_9L_5, \\
\Theta_{19} &= L_6^4N_5^2 + L_6^3L_3N_5^2 + [(L_6L_4)\Theta_4] + (L_6L_4)\Theta_9 + \Theta_{14}L_5.
\end{align*}
\]

4.6. Computing \( V_3(x) = \left\langle V_T(x) + K \mod U_3(x) \right\rangle \). We need to compute \( V_3(x) = (V_T(x) + K \mod U_3(x)) \).

Defining \( r_0 = \sum_{i=0}^{5} N_i x^i + K \), and \( r_1 = U_3(x) \), we have the iteration steps:

\[
r_2 = \sum_{i=0}^{4} Q_i x^i = P_4r_0(x) + N_5xr_1(x),
\]

and

\[
V_3(x) = r_3(x) = P_4r_2(x) + Q_4r_1(x).
\]
Group addition in projective/mixed coordinates, Step 6

The details of the computations are as follows:

\[ Q_4 = P_4 \cdot N_4 + N_5 \cdot P_3, \]
\[ P_4Q_3 + Q_4P_3 = P_4^2N_3 + \{ (P_4 \cdot N_5)P_2 \} + Q_4P_3, \]
\[ P_4Q_2 + Q_4P_2 = P_4^2N_2 + \{ (P_4 \cdot N_5)P_1 \} + Q_4P_2, \]
\[ P_4Q_1 + Q_4P_1 = P_4^2N_1 + (P_4 \cdot N_5)P_0 + [Q_4P_1], \]
\[ P_4Q_0 + Q_4P_0 = P_4^2(N_0 + K) + Q_4P_0. \]

For the ratio between \( V_3(x) \) and \( v_3(x) \), we find

\[ V_3(x) = P_4^2 \left( K(v_T(x) + 1) \mod u_3(x) \right) = (P_4^2K)v_3(x). \]

Algorithm 11 Group addition in projective/mixed coordinates, Step 6

1: \( Q_4 \leftarrow P_4 \cdot N_4 + N_5 \cdot P_3, \quad Aux_{17} \leftarrow P_4 \cdot N_5, \quad k_{b_{25}} \leftarrow (Aux_{17} + Q_4) \cdot (P_1 + P_2), \)
2: \( k_{a_{18}} \leftarrow Aux_{17} \cdot P_2, \quad k_{c_{17}} \leftarrow Q_4 \cdot P_1, \quad P_{42} \leftarrow P_4^2, \quad Q_8 \leftarrow P_{42} \cdot N_3 + k_{a_{18}} + Q_4 \cdot P_3, \)
3: \( Q_7 \leftarrow P_{42} \cdot N_2 + k_{b_{25}} + k_{a_{18}} + k_{c_{17}}, \quad Q_6 \leftarrow P_{42} \cdot N_1 + Aux_{17} \cdot \Theta_{19} + k_{c_{17}}, \)
4: \( Q_5 \leftarrow P_{42} \cdot (N_0 + K) + Q_4 \cdot \Theta_{19}, \quad P_{42k} \leftarrow P_{42} \cdot K \)
5: \( \text{\triangleright 14M+1S} \)

4.7. Computing \( \hat{U}_3(x) = (P_4K)U_3(x) \). We compute

\[ \hat{U}_3(x) = (P_4^2K)x^4 + (P_4K) \sum_{i=0}^{3} P_i x^i \]

TOTAL \( \hat{U}_3(x) \) 5M

Algorithm 12 Group addition in projective/mixed coordinates, Step 7

1: \( P_{4k} \leftarrow P_4 \cdot K, \quad a_3 \leftarrow P_{4k} \cdot P_3, \quad a_2 \leftarrow P_{4k} \cdot P_2, \quad a_1 \leftarrow P_{4k} \cdot P_1, \quad a_0 \leftarrow P_{4k} \cdot \Theta_{19} \)
2: \( \text{\triangleright 5M} \)
4.8. Mixed additions. A common practice when working with divisors in projective coordinates to compute $[e]D$, is to use a mix of affine and projective coordinates for the group addition. This is done for example by keeping $D$ in affine coordinates and performing the scalar multiplication in projective coordinates, but taking advantage that the extra coordinate(s) of $D$ when seeing it in projective form will have value 1. Since multiplications by 1 do not require any computations, we can re-write the addition formula to save a (large) number of multiplications. This approach is often referred to as “mixed coordinates additions”\(^\dagger\). It can also be applied to scalar multiplications that use more non-zero digits than \pm 1, for example in a w-NAF, in which case the small multiples of $D$ are recorded in affine coordinates.

Although Cantor’s addition algorithm is symmetric in the sense that the order of the two divisors inputted do not affect the outcome, that order does affect the intermediate computations in the explicit formulas. This is due to computation of the modular inverse computation (Step 1), so the choice of which of the two inputs should be in affine coordinates and which should be in projective coordinates is quite relevant on the cost of the group operation. From the formulas in the previous subsections, it is easy to see that there are much more multiplications by $a_4$ than $a_2$, so it is natural to choose the second divisor ($D_2$) to be the one in affine coordinates (forcing $a_4 = 1$). Obtaining the mixed-coordinates addition formula from the projective one is quite straightforward, so we give the corresponding formulas without further details.

**Algorithm 13** Group addition in mixed coordinates, Step 1

1. $A_3 \leftarrow a_3 + a_4 \cdot b_3, A_2 \leftarrow a_2 + a_4 \cdot b_2, A_1 \leftarrow a_1 + a_4 \cdot b_1, A_0 \leftarrow a_0 + a_4 \cdot b_0$
2. $A_6 \leftarrow A_3 \cdot b_6 + A_2, k_{c1}\leftarrow A_3 \cdot A_1, k_{c1} \leftarrow A_6 \cdot A_0, k_{b1} \leftarrow (A_3 + A_6) \cdot (A_0 + A_1),$
3. $A_{3+} \leftarrow A_3^2, A_9 \leftarrow A_{3+} \cdot b_2 + k_{c1} + A_6 \cdot A_2, A_8 \leftarrow A_{3+} \cdot b_1 + k_{b1} + k_{c1} + k_{c1},$
4. $A_7 \leftarrow A_{3+} \cdot b_0 + k_{c1}, C \leftarrow A_3 \cdot a_4, D \leftarrow A_6 \cdot a_4, A_{11} \leftarrow A_9 \cdot A_2 + A_3 \cdot A_8,$
5. $A_{9+} \leftarrow A_3^2, A_{12} \leftarrow A_{9+} \cdot A_0 + A_{11} \cdot A_7, A \leftarrow A_9 \cdot A_3,$
6. $A_{13} \leftarrow A_{9+} \cdot A_1 + A_7 \cdot A_3 + A_{11} \cdot A_8, F \leftarrow A \cdot C, k_{b2} \leftarrow (A + A_{11}) \cdot (C + D),$ 7. $k_{c2} \leftarrow A_{11} \cdot D, G \leftarrow F + k_{b2} + k_{c2}, I \leftarrow A_{9+} \cdot A_4 + k_{c2},$
8. $A_{15} \leftarrow A_{13} \cdot A_4 + A_9 \cdot A_{12}, L_2 \leftarrow A_{15}^2, r_5 \leftarrow L_2 \cdot A_7 + A_{15} \cdot A_{12},$
9. $E \leftarrow A_{12} \cdot A_9, k_{c3} \leftarrow E \cdot G, k_{c3} \leftarrow (E + A_{15}) \cdot (G + I), k_{c3} \leftarrow A_{15} \cdot I,$
10. $\delta_1 \leftarrow E \cdot F, \delta_2 \leftarrow A_{15} \cdot F + k_{c3}, \delta_1 \leftarrow E_3 \cdot C + k_{b4} + k_{c3} + k_{c3},$
11. $\delta_0 \leftarrow L_3 \cdot D + k_{c3}$
12. ▶ 38M+3S

**Algorithm 14** Group addition in mixed coordinates, Step 2

1. $E_3 \leftarrow c_3 + a_4 \cdot d_3, E_2 \leftarrow c_2 + a_4 \cdot d_2, E_1 \leftarrow c_1 + a_4 \cdot d_1,$
2. $E_0 \leftarrow c_0 + a_4 \cdot d_0, k_{a4} \leftarrow d_3 \cdot E_3, k_{a4} \leftarrow b_0 \cdot E_0, k_{c4} \leftarrow b_2 \cdot E_2,$
3. $k_{c5} \leftarrow b_1 \cdot E_1, k_{a5} \leftarrow (d_3 + d_2) \cdot (E_2 + E_3), k_{b5} \leftarrow (d_3 + d_1) \cdot (E_1 + E_3),$ 4. $k_{b6} \leftarrow (d_3 + d_0) \cdot (E_0 + E_2), k_{a6} \leftarrow (d_1 + d_0) \cdot (E_0 + E_1),$ 5. $k_{c7} \leftarrow (d_3 + d_2 + d_1 \cdot d_0) \cdot (E_0 + E_1 + E_2 + E_3),$ $G_5 \leftarrow k_{b6} + k_{c4} + k_{c7}, G_2 \leftarrow k_{b6} + k_{c4} + k_{a5} + k_{c5}, G_1 \leftarrow k_{b7} + k_{c5} + k_{a4},$
6. $G_7 \leftarrow k_{a5} + k_{a4} + G_5 + G_4 + G_2 + G_1 + k_{a4},$
8. $k_{a6} \leftarrow k_{a4} \cdot b_3, A_{10} \leftarrow G_5 + k_{a6}, k_{b6} \leftarrow b_2 \cdot A_{10}, k_{a7} \leftarrow k_{a5} \cdot b_1,$
9. $k_{c7} \leftarrow A_{10} \cdot b_0, k_{b7} \leftarrow (b_2 + b_1) \cdot (k_{a6} + A_{10}), k_{b7} \leftarrow (k_{a6} + A_{10}) \cdot (b_0 + b_1),$ 10. $A_{14} \leftarrow G_4 + k_{a6} + k_{a6} + k_{c6}, H_3 \leftarrow G_3 + k_{a7} + k_{a6} + A_{14} \cdot b_3,$
11. $H_2 \leftarrow G_2 + k_{b6} + k_{a6} + E_3 \cdot b_2, H_1 \leftarrow G_1 + k_{c7} + A_{14} \cdot b_1,$
12. $H_0 \leftarrow k_{a5} + A_{14} \cdot b_0, \rho \leftarrow r_5 \cdot a_4$
13. ▶ 24M

\(^\dagger\)In some cases, the mix may involve more than one type of projective coordinates, although this is not the case here
Algorithm 15 Group addition in mixed coordinates, Step 3

1: $S_0 \leftarrow H_2^3$, $S_1 \leftarrow H_2^3$, $S_2 \leftarrow H_2^3$, $S_0 \leftarrow H_2^3$, $k_{8a} \leftarrow S_6 \cdot a_4$, $k_{9a} \leftarrow S_6 \cdot a_3$.
2: $k_{10} \leftarrow (S_0 + S_1) \cdot (a_1 + a_3)$, $k_{12} \leftarrow (S_0 + S_2) \cdot (a_0 + a_4)$.
3: $k_{13} \leftarrow (S_2 + S_4) \cdot (a_0 + a_2)$, $K_8 \leftarrow k_{10} + k_{8a} + k_{8b}$, $K_7 \leftarrow k_{10} + k_{8a} + k_{8b}$.
4: $K_5 \leftarrow k_{10} + k_{8a} + k_{8b}$, $K_5 \leftarrow k_{10} + k_{8a} + k_{8b}$, $K_4 \leftarrow S_0 \cdot a_4 + k_{13} + k_{10} + k_{8b}$.
5: $k_{10} \leftarrow k_{8a} \cdot b_3$, $E_3 \leftarrow k_{8a} + k_{10}$, $k_{11} \leftarrow E_3 \cdot b_2$, $k_{14} \leftarrow (k_{8a} + E_3) \cdot (b_2 + b_3)$.
6: $S_7 \leftarrow K_8 + k_{14} + k_{10} + k_{11}$, $k_{14} \leftarrow S_7 \cdot b_1$, $k_{14} \leftarrow S_7 \cdot b_1$.
7: $E_{11} \leftarrow k_{15} + k_{14} + k_{10} + k_{11} + k_{14}$, $k_{11} \leftarrow E_3 \cdot b_2$, $k_{14} \leftarrow k_{8a} + E_3) \cdot (b_2 + b_3)$.
8: $k_{16} \leftarrow (E_3 + E_11) \cdot (b_0 + b_2)$, $k_{16} \leftarrow (E_3 + E_11) \cdot (b_0 + b_2)$.
9: $k_{16} \leftarrow (E_3 + E_11) \cdot (b_0 + b_2)$, $k_{16} \leftarrow (E_3 + E_11) \cdot (b_0 + b_2)$.
10: $k_{16} \leftarrow (E_3 + E_11) \cdot (b_0 + b_2)$, $k_{16} \leftarrow (E_3 + E_11) \cdot (b_0 + b_2)$.

4.9. Cost of the addition formulas.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Projective</th>
<th>Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\langle U_1^{-1} \mod U_2 \rangle$</td>
<td>45M + 3S</td>
<td>38M + 3S</td>
</tr>
<tr>
<td>$S(x) = \langle (U_1^{-1} \mod U_2)(b_4V_1 + a_4V_2) \mod U_2 \rangle$</td>
<td>39M + 1S</td>
<td>24M</td>
</tr>
<tr>
<td>$U_T(x) = \phi^2b^2 \left\langle \frac{f}{U_1^2} \right\rangle + \left\langle \frac{s^2U_1}{U_2} \right\rangle$</td>
<td>41M + 7S</td>
<td>25M + 5S</td>
</tr>
<tr>
<td>$V_T(x) = \left\langle (6V_1 + U_1 + \phi a_4) \mod U_T \right\rangle$</td>
<td>37M + 1S</td>
<td>37M + 1S</td>
</tr>
<tr>
<td>$U_3(x) = \left\langle \frac{K \cdot f + V_1}{U_2} \right\rangle$</td>
<td>22M + 5S</td>
<td>22M + 5S</td>
</tr>
<tr>
<td>$V_3(x) = \left\langle V_T + K \mod U_3 \right\rangle$</td>
<td>14M + 1S</td>
<td>14M + 1S</td>
</tr>
<tr>
<td>$U_3(x) = (P_4K)U_3(x)$</td>
<td>5M</td>
<td>5M</td>
</tr>
<tr>
<td>Total Cost</td>
<td>203M + 18S</td>
<td>165M + 15S</td>
</tr>
</tbody>
</table>

5. Explicit formulas improvement for genus 4 HECs doubling case

We consider the divisor $D_1 = [U_1(x), V_1(x)]$, and compute $D_3 = 2D_1$.

5.1. Computing $U_C(x) = \langle U_1^2(x) \rangle$. This is a direct computation, and we obtain $U_C(x) = a_4^2x^8 + a_3^2x^6 + a_2^2x^4 + a_1^2x^2 + a_0^2$.

Algorithm 16 Group doubling in projective coordinates, Step 1

1: $A_{u1} \leftarrow a_4^2$, $A_{u3} \leftarrow a_3^2$, $A_{u2} \leftarrow a_2^2$, $A_{u1} \leftarrow a_1^2$, $A_{u0} \leftarrow a_0^2$.

5.2. Computing $V_C(x) = V_3^2(x) + \left\langle f(x) \mod U_C(x) \right\rangle$. We first compute $f(x) \mod U_C(x) = \eta_4x^7 + \eta_3x^5 + \eta_2x^3 + \eta_1x + \eta_0$, where

$\eta_4 = a_4^2f_7 + a_3^2$, $\eta_3 = a_2^2$, $\eta_2 = a_1^2f_3 + a_1^2$. 

Advances in Mathematics of Communications  Volume 4, No. 2 (2010), 115–139
\[ \eta_1 = a_4^2 f_1 + a_6^2, \quad \eta_0 = a_4^2 f_0, \]
and the projective constant for this operation is \( D = a_4^2 \). For the other half of the sum, we have \( V_t^2(x) = c_3^2 x^6 + c_2^2 x^4 + c_1^2 x^2 + c_0^2 \).

Now \( V_C(x) = V_t^2(x) + \left( f(x) \mod U_C(x) \right) \), and we let \( V_C(x) = \sum_{i=0}^{7} R_i x^i \) with

\[
\begin{align*}
R_7 &= \eta_4, & R_6 &= c_3^2, & R_5 &= \eta_3, & R_4 &= c_2^2, \\
R_3 &= \eta_2, & R_2 &= c_1^2, & R_1 &= \eta_1, & R_0 &= c_0^2 + \eta_0,
\end{align*}
\]

and \( V_C(x) = a_4^2 v_C(x) \).

**Algorithm 17** Group doubling in projective coordinates, Step 2

1. \( \eta_4 \leftarrow A_4 \cdot f_7 + A_{u_5}, \quad \eta_2 \leftarrow A_4 \cdot f_3 + A_{u_4}, \quad \eta_1 \leftarrow A_4 \cdot f_1 + A_0, \)
2. \( \eta_0 \leftarrow A_4 \cdot f_0, \quad A_{u_5} \Leftarrow c_0^2, \quad A_{u_6} \Leftarrow c_1^2, \quad A_{u_7} \Leftarrow c_2^2, \quad A_{u_8} \Leftarrow c_3^3, \)
3. \( R_0 \Leftarrow A_{u_5} + \eta_0 \)
4. \( \triangleright 4M+4S \)

5.3. **Computing** \( U_T(x) = a_4^{10} \left( \left\langle \frac{f(x)}{U_C(x)} \right\rangle + \left\langle \frac{V_C(x)}{U_1(x)} \right\rangle \right)^2 \). We first compute \( \left\langle \frac{f(x)}{U_C(x)} \right\rangle \)
and \( \left\langle \frac{V_C(x)}{U_1(x)} \right\rangle \) as follows:

- \( \left\langle \frac{f(x)}{U_C(x)} \right\rangle = x \), with a projective constant of \( a_4^2 \).
- \( \left\langle \frac{V_C(x)}{U_1(x)} \right\rangle \) is given by \( a_4^3 R_7 x^3 + a_4^2 T_3 x^2 + a_4 T_7 x + T_{11} \), with

\[
T_3 = a_4 R_6 + [R_7 a_3],
T_7 = a_4^2 R_5 + \{(a_4 a_2) R_7 + T_3 a_3\},
T_{11} = (a_4(a_4^2)) R_4 + (a_4^2 R_7) a_1 + [(a_4 a_2) T_3] + T_7 a_3,
\]

and a projective constant of \( a_4^3 \). From these we easily obtain

\[
\left\langle \frac{V_C(x)}{U_1(x)} \right\rangle^2 = (a_4^3 R_7)^2 x^6 + (a_4^2 T_3)^2 x^4 + (a_4 T_7)^2 x^2 + T_{11}^2,
\]

and a projective constant of \( a_4^3 \).

Combining the two, we find

\[
U_T(x) = a_4^{10} \left( \left\langle \frac{f(x)}{U_C(x)} \right\rangle + \left\langle \frac{V_C(x)}{U_1(x)} \right\rangle \right)^2 = (a_4^3(a_4^2))^2 x + (a_4^3 R_7)^2 x^6 + (a_4^2 T_3)^2 x^4 + (a_4 T_7)^2 x^2 + T_{11}^2,
\]

and define \( U_T(x) = a_6 x^6 + a_4 x^4 + a_2 x^2 + a_1 x + a_0 \).

5.4. **Computing** \( V_T(x) = \left\langle (V_C(x) + a_4^2) \mod U_T(x) \right\rangle \). We define

\[
V_C(x) + a_4^2 = \left( \sum_{i=0}^{7} R_i x^i \right) + a_4^2.
\]
Algorithm 18 Group doubling in projective coordinates, Step 3
1. \( A_{a_1} \leftarrow a_1 \cdot a_2 \), \( k_{a_1} \leftarrow \eta_4 \cdot a_3 \), \( T_3 \leftarrow A_4 \cdot A_{a_3} + k_{a_1} \), \( k_{b_1} \leftarrow (A_{u_9} + a_3) \cdot (\eta_4 + T_3) \),
2. \( k_{a_2} \leftarrow A_{u_9} \cdot T_3 \), \( T_7 \leftarrow A_{u_4} \cdot A_{u_2} + k_{b_1} + k_{c_2} + k_{a_1} \), \( A_{u_{10}} \leftarrow A_{u_4} \cdot \eta_4 \),
3. \( Aux_{1} \leftarrow a_4 \cdot A_{u_{14}} \), \( T_{11} \leftarrow Aux_{1} \cdot A_{u_{15}} + A_{u_{19}} \cdot a_1 + k_{c_1} + T_7 \cdot a_3 \), \( A_{u_{11}} \leftarrow a_4 \cdot T_7 \),
4. \( A_{u_{12}} \leftarrow A_{u_4} \cdot T_3 \), \( A_{u_{13}} \leftarrow Aux_{1} \cdot \eta_4 \), \( \alpha_6 \leftarrow A^2_{u_{13}} \), \( \alpha_4 \leftarrow A^2_{u_{12}} \), \( \alpha_2 \leftarrow A^2_{u_{11}} \),
5. \( \alpha_1 \leftarrow (A_{u_4} \cdot Aux_{1})^2 \), \( \alpha_0 \leftarrow T^2_{11} \).
6. \( \triangleright 15M+5S \)

We compute \( V_T(x) = \left\langle (V_C(x) + a_2^2) \mod U_T(x) \right\rangle = \sum_{i=0}^{5} T_i x^i \), as:
\[
\begin{align*}
T_5 &= \alpha_6 R_5 + R_7 \alpha_4, \\
T_3 &= \alpha_6 R_3 + [R_7 \alpha_2], \\
T_1 &= \alpha_6 R_1 + R_7 a_0 + [R_6 \alpha_1],
\end{align*}
\]
\( T_4 = \alpha_6 R_4 + R_6 \alpha_4 \), \( T_2 = \alpha_6 R_2 + \{R_7 \alpha_1 + R_6 \alpha_2 \} \).

The modular reduction has a projective constant of \( \alpha_6 \), and we find that \( V_T = \delta v_T \) with \( \delta = \alpha_6 a_2^2 \).

Algorithm 19 Group doubling in projective coordinates, Step 4
1. \( k_{u_2} \leftarrow \eta_4 \cdot a_2 \), \( k_{u_2} \leftarrow (\eta_4 + A_{u_8}) \cdot (a_1 + a_2) \), \( k_{c_2} \leftarrow A_{u_8} \cdot \alpha_1 \),
2. \( T_5 \leftarrow \alpha_6 \cdot A_{u_2} + \eta_4 \cdot a_4 \), \( T_4 \leftarrow \alpha_6 \cdot A_{u_7} + A_{u_8} \cdot \alpha_4 \), \( T_3 \leftarrow \alpha_6 \cdot \eta_2 + k_{u_2} \),
3. \( T_2 \leftarrow \alpha_6 \cdot A_{u_6} + k_{b_2} + k_{a_2} + k_{c_2} \), \( T_1 \leftarrow \alpha_6 \cdot \eta_1 + \eta_4 \cdot \alpha_0 + k_{c_2} \),
4. \( T_0 \leftarrow \alpha_6 \cdot (R_0 + A_{u_4}) + A_{u_8} \cdot \alpha_0 \), \( \delta \leftarrow \alpha_6 \cdot A_{u_4} \).
5. \( \triangleright 14M \)

5.5. Computing \( U_3(x) = \alpha_6 \delta^2 \left\langle \left| \frac{f(x)}{U_T(x)} \right| \right\rangle + \left\langle \left| \frac{V_T^2(x)}{U_T(x)} \right| \right\rangle \).

- First, we compute \( \left\langle \left| \frac{f(x)}{U_T(x)} \right| \right\rangle = \alpha_6 x^3 + \xi_5 x \) where \( \xi_5 = \alpha_6 f_7 + \alpha_4 \), and the projective constant is \( \alpha_6^2 \).
- Secondly, we obtain \( \left\langle \left| \frac{V_T^2(x)}{U_T(x)} \right| \right\rangle = \alpha_6^2 T_5^2 x^4 + \alpha_6 \varphi_8 x^2 + \varphi_{14} \) with \( \varphi_6 = \alpha_6 T_5^2 + T_5^2 \alpha_2 \), \( \varphi_8 = \alpha_6 T_4^2 + T_5^2 \alpha_4 \), \( \varphi_{14} = \alpha_6 \varphi_6 + \varphi_8 \alpha_4 \), and as a result:
\[
U_3(x) = \alpha_6 \delta^2 \left\langle \left| \frac{f(x)}{U_T(x)} \right| \right\rangle + \left\langle \left| \frac{V_T^2(x)}{U_T(x)} \right| \right\rangle = (\alpha_6^2 T_5^2) x^4 + (\alpha_6^2 \delta^2) x^3 + (\alpha_6 \varphi_8) x^2 + (\delta^2 \alpha_6 \xi_5) x + \varphi_{14},
\]
which we write as \( U_3(x) = \sum_{i=0}^{4} \psi_i x^i \).

Algorithm 20 Group doubling in projective coordinates, Step 5
1. \( \xi_5 \leftarrow \alpha_6 \cdot f_7 + \alpha_4 \), \( \sigma \leftarrow T_5^2 \), \( \varphi_8 \leftarrow \alpha_6 \cdot T_3^2 + \sigma \cdot a_2 \), \( \varphi_8 \leftarrow \alpha_6 \cdot T_4^2 + \sigma \cdot a_4 \),
2. \( \varphi_{14} \leftarrow \alpha_6 \cdot \varphi_6 + \varphi_8 \cdot \alpha_4 \), \( \delta_2 \leftarrow \delta^2 \), \( Aux_{2} \leftarrow \alpha_6^2 \), \( \psi_4 \leftarrow Aux_{2} \cdot \sigma \),
3. \( \psi_3 \leftarrow Aux_{2} \cdot \delta_2 \), \( \psi_2 \leftarrow \alpha_6 \cdot \varphi_8 \), \( \psi_1 \leftarrow \alpha_6 \cdot \xi_5 \cdot \delta_2 \)
4. \( \triangleright 12M+5S \)
5.6. Computing \( V_3(x) = \langle V_T(x) + \delta \mod U_3(x) \rangle \). We find an equivalent multiple of \( v_3(x) \) by factoring out a common multiple from the coefficients. We first have:

\[
V_3(x) = x^3(\psi_4 \varphi_8 + \varphi_9 \psi_3) + x^2(\psi_4 \varphi_7 + \varphi_9 \psi_2) + x(\psi_4 \varphi_6 + \varphi_9 \psi_1) + \psi_4 \varphi_5 + \varphi_9 \psi_0
\]

where \( \varphi_9 \) satisfies

\[
\varphi_9 = \psi_4 T_4 + T_5 \psi_3 = \alpha_6^2 T_5^2 T_4 + T_5 \alpha_6^2 \delta^2 = \alpha_6^2 T_5 (T_5 T_4 + \delta) = (\alpha_6^2 T_5) \varphi_9.
\]

The other \( \varphi_i \) are given by

\[
\varphi_8 = \psi_4 T_3 + T_5 \psi_2, \quad \varphi_7 = \psi_4 T_2 + T_3 \psi_1, \quad \varphi_6 = \psi_4 T_1 + T_5 \psi_0, \quad \varphi_5 = \psi_4 (T_0 + \delta).
\]

Defining \( \Gamma = \psi_4 T_5, \sigma = T_5^2 \), and \( \psi = \alpha_6^2 T_5 \), we can re-write the coefficients of \( V_3(x) \) as

\[
\psi_4 \varphi_8 + \varphi_9 \psi_3 = \alpha_6^2 T_5 (\Gamma T_3 + \sigma \psi_2 + \widehat{\varphi}_9 \psi_3), \quad \psi_4 \varphi_7 + \varphi_9 \psi_2 = \alpha_6^2 T_5 (\Gamma T_2 + \sigma \psi_1 + \widehat{\varphi}_9 \psi_2),
\]

\[
\psi_4 \varphi_6 + \varphi_9 \psi_1 = \alpha_6^2 T_5 (\Gamma T_1 + \sigma \psi_0 + \widehat{\varphi}_9 \psi_1), \quad \psi_4 \varphi_5 + \varphi_9 \psi_0 = \alpha_6^2 T_5 (\Gamma (T_0 + \delta) + \widehat{\varphi}_9 \psi_0).
\]

Since \( V_3 = (\psi_3^2 \delta) v_3 = (\alpha_6^2 T_5) \psi_4 T_5 \delta v_3 \), we can remove a factor of \( \alpha_6^2 T_5 \), and compute \( \widehat{V}_3 = (\Gamma \delta) v_3 \) with coefficients

\[
\Gamma T_3 + \sigma \psi_2 + \widehat{\varphi}_9 \psi_3, \quad \Gamma T_2 + [\sigma \psi_1] + \widehat{\varphi}_9 \psi_2,
\]

\[
\Gamma T_1 + \{\sigma \psi_0 + \widehat{\varphi}_9 \psi_1\}, \quad \Gamma (T_0 + \delta) + [\widehat{\varphi}_9 \psi_0].
\]

**Algorithm 21** Group doubling in projective coordinates, Step 6

1: \( \widehat{\varphi}_9 \leftarrow T_5 \cdot T_4 + \delta, \quad \Gamma \leftarrow \psi_4 \cdot T_5, \quad k_{a_3} \leftarrow \sigma \cdot \psi_1, \quad k_{b_3} \leftarrow (\sigma + \widehat{\varphi}_9) \cdot (\varphi_{14} + \psi_1), \)

2: \( k_{c_3} \leftarrow \widehat{\varphi}_9 \cdot \varphi_{14}, \quad v_3 \leftarrow \Gamma \cdot T_3 + \sigma \cdot \psi_2 + \widehat{\varphi}_9 \cdot \psi_3, \quad v_2 \leftarrow \Gamma \cdot T_2 + k_{a_3} + \widehat{\varphi}_9 \cdot \psi_2, \quad v_1 \leftarrow \Gamma \cdot T_1 + k_{b_3} + k_{c_3} + k_{a_3}, \quad v_0 \leftarrow \Gamma \cdot (T_0 + \delta) + k_{c_3}, \quad \chi \leftarrow \Gamma \cdot \delta \)

4: \( \triangleright 13M \)

5.7. Computing \( \widehat{U}_3(x) = (T_5 \delta) U_3(x) \). We compute

\[
\widehat{U}_3(x) = (\Gamma \delta) x^4 + (T_5 \delta) \sum_{i=0}^{3} \psi_i x^i.
\]

**Algorithm 22** Group doubling in projective coordinates, Step 7

1: \( \omega \leftarrow T_5 \cdot \delta, \quad u_3 \leftarrow \omega \cdot \psi_3, \quad u_2 \leftarrow \omega \cdot \psi_2, \quad u_1 \leftarrow \omega \cdot \psi_1, \quad u_0 \leftarrow \omega \cdot \varphi_{14} \)

2: \( \triangleright 5M \)
5.8. Cost of the Doubling Formula.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_C(x) = \langle U_C^2 \rangle$</td>
<td>$5S$</td>
</tr>
<tr>
<td>$V_C(x) = V_C^2 + \left\langle f \pmod{U_C} \right\rangle$</td>
<td>$4M + 4S$</td>
</tr>
<tr>
<td>$U_T(x) = a_4^0 \left\langle \frac{U_T}{V_T} \right\rangle + \left\langle \frac{V_T}{U_T} \right\rangle^2$</td>
<td>$15M + 5S$</td>
</tr>
<tr>
<td>$V_T(x) = \left\langle (V_C + a_3^2) \pmod{U_T} \right\rangle$</td>
<td>$14M$</td>
</tr>
<tr>
<td>$U_3(x) = \alpha_6 \delta^2 \left\langle \frac{U_T}{V_T} \right\rangle + \left\langle \frac{V_T}{U_T} \right\rangle$</td>
<td>$12M + 5S$</td>
</tr>
<tr>
<td>$V_3(x)$</td>
<td>$13M$</td>
</tr>
<tr>
<td>$U_3(x) = (T_5 \delta) U_3(x)$</td>
<td>$5M$</td>
</tr>
<tr>
<td>Total Cost</td>
<td>$63M + 19S$</td>
</tr>
</tbody>
</table>

6. Conclusion

We presented the first explicit formulas for genus four hyperelliptic curves in projective coordinates. These formulas allow inversion-free group arithmetic for those curves. Such group operations are particularly interesting for hardware implementations since they remove the need for an inverter unit, which can produce significant savings in the cost of the processor (reduced size), or allow to have more multiplier units (for faster group arithmetic). Such an implementation, including protection against side-channel attacks will be the subject of future work.

In Tables 1 and 2, we compare our operation counts with those for hyperelliptic curves of other genus and with hyperelliptic curves of genus four using affine coordinates. Note that for elliptic curves (genus one) the costs are those of López-Dahab coordinates and for genus 2 curves the projection is in “recent” coordinates [1]. In Table 1, the curves of genus 1 and 2 are of the form $h(x) = x$ and those of genus 3 and 4 are of the form $h(x) = 1$ (for genus 1 and 2, curves with $h(x) = 1$ are supersingular, which is not the case in genus 3, and can be avoided in genus 4 by requiring that $f_7 \neq 0$ [25]). We also indicate the proportion between field sizes (in bits) to obtain the same level of security for the different genera (see [3], Section 5, for a detailed discussion on the security aspects).

Table 1. Cost of group arithmetic in projective coordinates

<table>
<thead>
<tr>
<th>Reference</th>
<th>Genus</th>
<th>Field Size</th>
<th>Addition</th>
<th>Mixed Add.</th>
<th>Doubling</th>
</tr>
</thead>
<tbody>
<tr>
<td>[16, 1]</td>
<td>1</td>
<td>$n$</td>
<td>$13M + 4S$</td>
<td>$8M + 5S$</td>
<td>$5M + 4S$</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>$(1/2)n$</td>
<td>$44M + 6S$</td>
<td>$36M + 4S$</td>
<td>$20M + 8S$</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>$(3/8)n$</td>
<td>$119M + 9S$</td>
<td>$102M + 8S$</td>
<td>$42M + 15S$</td>
</tr>
<tr>
<td>This work</td>
<td>4</td>
<td>$(1/3)n$</td>
<td>$203M + 18S$</td>
<td>$165M + 15S$</td>
<td>$63M + 19S$</td>
</tr>
</tbody>
</table>

Table 2. Cost of group arithmetic in genus four hyperelliptic curves

<table>
<thead>
<tr>
<th>Reference</th>
<th>Coordinates</th>
<th>Addition</th>
<th>Mixed Add.</th>
<th>Doubling</th>
</tr>
</thead>
<tbody>
<tr>
<td>[3]</td>
<td>affine</td>
<td>$1I + 119M + 10S$</td>
<td>–</td>
<td>$1I + 28M + 16S$</td>
</tr>
<tr>
<td>This work</td>
<td>projective</td>
<td>$203M + 18S$</td>
<td>$165M + 15S$</td>
<td>$63M + 19S$</td>
</tr>
</tbody>
</table>

The formulas presented in this paper have all been implemented in Magma and thoroughly verified, then transferred to C to be timed with a specialized field arithmetic library [2]. The Magma files for the formulas can be obtained on request (by email to the first author). The resulting timings can be found in Table 3. These timings are compared to the genus 4 affine formulas of [3]. It is easy to see that affine
coordinates formulas outperform the projective formulas if only timing is considered (as could be expected from the operation counts).

Table 3. Timings for the group operations in genus four hyperelliptic curves on a 2.8 GHz Core 2 Duo computer

<table>
<thead>
<tr>
<th>Field Size (in bits)</th>
<th>Affine coordinates ([3])</th>
<th>Projective coordinates (this work)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Addition</td>
<td>Doubling</td>
</tr>
<tr>
<td>47</td>
<td>5.6946</td>
<td>1.9665</td>
</tr>
<tr>
<td>53</td>
<td>6.7143</td>
<td>2.3816</td>
</tr>
<tr>
<td>59</td>
<td>7.2209</td>
<td>2.5724</td>
</tr>
<tr>
<td>67</td>
<td>11.2299</td>
<td>3.4740</td>
</tr>
<tr>
<td>71</td>
<td>11.2299</td>
<td>3.6368</td>
</tr>
<tr>
<td>79</td>
<td>11.5277</td>
<td>3.8348</td>
</tr>
</tbody>
</table>

However, the relative difference between affine and projective coordinates is in the order of 45% to 69% for the doublings and 24% to 44% for the additions (affine versus mixed). These ratios could still make projective formulas interesting in hardware as the processor no longer requires a field inversion unit.

In that setting, it is common to take into account the area of the processor (which affects both the cost and the amount of computations the processor can perform). The reduction in area due to removing the inversion circuit could then be sufficient to offset the increased number of multiplications. Unfortunately, a full comparison would require actual implementations in hardware of all the formulas and field arithmetic, which is beyond the scope of this work.

A similar situation can be found when comparing curves of different genera. There, curves of genus four require a smaller finite field than curves of lower genus, which has a significant impact on the cost of the field arithmetic. For hardware implementations, this is true not only for the speed of the field arithmetic, but also for the area (size) of the processor and its power consumption.

References

Genus 4 arithmetic in projective coordinates


Received May 2009; revised April 2010.

E-mail address: rodrigo.abarzua@usach.cl
E-mail address: ntheriau@inst-mat.utalca.cl
E-mail address: roberto.avanzi@ruhr-uni-bochum.de