Efficient Doubling on Genus 3 Curves over Binary Fields

Xinxin Fan[1], Thomas Wollinger[2], Yumin Wang[1]

[1] State Key Lab of Integrated Service Networks,
Xidian University, Xi'an, P.R.China
xxfan@mail.xidian.edu.cn
ymwang@xidian.edu.cn

[2] Communication Security Group (COSY)
Ruhr-Universitäet Bochum, Germany
wollinger@crypto.rub.de

Abstract. The most important and expensive operation in a hyperelliptic curve cryptosystem (HECC) is scalar multiplication by an integer k, i.e., computing an integer k times a divisor D on the Jacobian. Using some recoding algorithms for scalar k, we can reduce a number of divisor class additions during the process of computing scalar multiplication. So divisor doubling will account for the main part in all kinds of scalar multiplication algorithms. In order to accelerate the genus 3 HECC over binary fields we investigate how to compute faster doubling in this paper.

By constructing birational transformation of variables, we derive explicit doubling formulae for all types of defining equations of the curve. For each type of curve, we analyze how many field operations are needed. So far all proposed curves are secure, though they are more special types. Our results allow to choose curves from a large enough variety which have extremely fast doubling needing only one third the time of an addition in the best case. Furthermore, an actual implementation of the new formulae on a Pentium-M processor shows its practical relevance.

Keywords: Genus 3 Hyperelliptic Curve, Explicit Doubling Formulae, Fast Arithmetic, Binary Fields

1 Introduction

In 1988, Neal Koblitz suggested for the first time the generalization of elliptic curves to curves of higher genus for cryptographic use, namely hyperelliptic curves [Kob88, Kob89]. The operand size of HECC is even shorter compared to elliptic curve cryptosystem (ECC). This fact is advantageous for HECC on any platform. During the last decade, elliptic curve cryptosystems (ECC) [Kob87, Mil86] have been extensively studied from both a pure and applied perspective. However, HECC obtained a lot of attention till recent years. There has been a major effort in improving the group operations and in implementing HECC on different processors. Using explicit formulae instead of Cantor algorithm has reduced sharply the complexity of arithmetic in the ideal class group of hyperelliptic curves and obtained fast implementation in software [MCT01, MDM$^+$02, KMG$^+$02, Tak02, PWP03, WPW$^+$03, Lan03, GMA$^+$04, Ava04, Wol04, FWW05] and hardware platform [BCLW02, Cla02, EMY04, KWC$^+$04].

For all kinds of cryptographic protocols based on ECC or HECC, the computation of scalar multiplication is the main operation. Scalar multiplication algorithms include usually divisor class additions, doublings and perhaps some precomputations. By some recoding methods for scalar, we can reduce a number of divisor class additions. However, we cannot decrease the number of divisor class doubling in general case (for some special curves, it is possible). So divisor class doublings will become the crucial step for the performance of the entire cryptosystem. Improving the arithmetic of doubling has a direct impact on the efficiency of the whole system.

In [LS04, BD04], several authors discuss genus 2 curves over fields of characteristic 2 and doubling formulae for the different types of curves in detail. They give a complete study of all cases of defining equation of the curve and make a trade-off between speed-up and special parameters. Although we can

[1] Supported by the National NKBRSF '973' Program of China (Grant No.G1999035803)

use Koblitz curves to accelerate the computation of scalar multiplication [GLS00, Lan04], there are only 6 and 24 different isogenie classes for genus 2 and 3 binary curves, respectively. So the choice of curves is rather limited. In order to enlarge the range of selecting curves, we will address all kinds of genus 3 curves defined over the extension field in this paper.

For genus 3 curves over $GF(2^n)$, Pelzl et al. [PWGP03] discussed a very special type of curves with $h(x) = 1$ and gave efficient doubling explicit formulae. In [GKP04], the authors proposed efficient algorithms to compute the resultant of two polynomials and of the inverse of one polynomial modulo another, and improved the overall complexity of complexity of the addition and doubling algorithms for both even and odd characteristics. Their explicit formulae are applicable to almost all hyperelliptic curves of genus 3. By using a birational transformation of the form $(x, y) \mapsto (\lambda x + \mu, \nu y)$, they discuss five possible types of curves for even characteristic case.

In this article, we generalize the ideas proposed in [LS04] to genus 3 case and improve the results in [GKP04] further. we construct isomorphic transformations first to achieve as many zero coefficients as possible, and then make strong use of the defining equation of the curve obtain more efficient doubling explicit formulae. We do a complete study of all kinds of curves and analyze which kind of curve can lead to fast computation of doubling a divisor class. Finally, we combine the new doubling explicit formulae with NAF method to compute scalar multiplication fast, and give detailed experiment results.

The remainder of the paper is organized as follows: Section 2 states a brief mathematical background related to genus 3 hyperelliptic curves over binary fields. Section 3 describes Harley's algorithm for doubling a divisor class. In section 4, 5, 6, and 7 we derive the new doubling explicit formulae for genus 3 curves according to the different degree of $h(x)$. Section 8 summarize our contributions. Finally, we present our experimental results in Section 9 and conclude with a discussion of our results in Section 10.

2 Genus 3 Hyperellitpic Curves and Their Divisor Class Groups

In this section we present the representation of the divisor class group elements for genus 3 hyperelliptic curves over finite fields of characteristic two. For mathematical background and more details about hyperelliptic curves, please the interested reader refer to [Can87, Kob89, MWZ96].

Let $GF(q), q = 2^l$ be a finite field of characteristic 2. A non-singular (imaginary quadratic) hyperelliptic curve C of genus 3 over $GF(q)$ is defined by an equation of the form

$$C : Y^2 + h(X)Y = f(X),$$

where $h(X)$ is a polynomial of degree ≤ 3, and $f(X)$ is a monic polynomial of degree 7, i.e.,

$$h(X) = h_3X^3 + h_2X^2 + h_1X + h_0,$$

$$f(X) = X^7 + f_6X^6 + f_5X^5 + f_4X^4 + f_3X^3 + f_2X^2 + f_1X + f_0,$$

with $h_i, f_i \in GF(q)$.
The equation $(*)$ defining a hyperelliptic curves C of genus 3 is unique up to a change of coordinates of the form

$(*)$ $$(x, y) \longrightarrow (\alpha^2 x + \beta, \alpha^7 y + t(x)),$$

Where $\alpha, \beta \in GF(q)$ with $\alpha \neq 0$ and $t(x) \in F_q[x]$ with deg $t \leq 3$ [Lok94]. If an algebraic curve has a singular point then the curve is singular. However a hyperelliptic curve is by definition non-singular. The divisor class group $J_C(GF(q))$ of C forms a finite abelian group and therefore we can construct cryptosystems based on discrete logarithm problems on the Jacobian of C. Any equivalent class D in $J_C(GF(q))$ can be represented by Mumford's representation as follows [Mum84].

Mumford's Representation for Genus 3 Hyperelliptic Curves:
$D = [u(x), v(x)] = [x^3 + u_2x^2 + u_1x + u_0, v_2x^2 + v_1x + v_0], u(x), v(x) \in F_q[X], where$
$u(x) = x^3 + u_2x^2 + u_1x + u_0 = \prod_{i=1}^{3}(x - x_i)^{ord_{P_i}(D)},$
$y_i = v(x_i)$
for $P_i = (x_i, y_i) \in C$ with $ord_{P_i}(D) > 0, (i = 1, 2, 3)$ and $u|v^2 + hv + f$.

The degree of $u(x)$ is called the weight of D and D a reduced divisor, if its weight equals 3. Any class in $J_C(GF(q))$ can be uniquely represented by a reduced divisor.

3 Harley's Algorithm for Divisor Class Doublings

In [GH00], the authors noticed that one can reduced the number of operations by distinguishing between possible cases according to the properties of the input divisors. They proposed an efficient algorithm (using many computational algebra tricks such as Karatsuba multiplication, Chinese Remainder Theory, and Newton Interation) to compute in the Jacobian of hyperelliptic curves. For a complete description about explicit formulae for group operations we refer to [Wol04].

In this paper we concentrate on doublings for genus 3 curves in the most significant case where the input divisor $[u(x), v(x)]$ has full degree and u and h do not have a common factor. Therefore, we assume from now on

$$D = [u(x), v(x)], \quad deg\ u(x) = 3, \quad resutant\ [u(x), h(x)] \neq 0.$$

Let $u(x) = x^3 + u_2x^2 + u_1x + u_0, v(x) = v_2x^2 + v_1x + v_0$. Using the following Harley algorithm, we can double a divisor class on a Jacobian:
Step 1. Compute resultant r of u and h;
Step 2. Compute almost inverse $inv = r/h \bmod u = inv_2x^2 + inv_1x + inv_0$;
Step 3. Compute $z = ((f - hv - v^2)/u) \bmod u = z_2x^2 + z_1x + z_0$;
Step 4. Compute $s' = zinv \bmod u = s'_2x^2 + s'_1x + s'_0$;
Step 5. Compute $s = (s'/r)$ and make s monic: $s = x^2 + s_1x + s_0$;
Step 6. Compute $G = su = x^5 + g_4x^4 + g_3x^3 + g_2x^2 + g_1x + g_0$;
Step 7. Compute $u' = u^{-2}\{[G + (r/s'_2)v]^2 + (r/s'_2)hG + (r/s'_2)^2(hv - f)\} = u'_3x^3 + u'_2x^2 + u'_1x + u'_0$;
Step 8. Compute $v' = -[G(s'_2/r) + h + v] \bmod u' = v'_3x^3 + v'_2x^2 + v'_1x + v'_0$;
Step 9. Reduce u': $u'' = (f - v'h - v'^2)/u' = x^3 + u''_2x^2 + u''_1x + u''_0$;
Step 10. Compute $v'' = -(v' + h) \bmod u'' = v''_2x^2 + v''_1x + v''_0$.

We now study the different expressions for h separately because the actual execution of the Harley's algorithm depends on the coefficients of the curve. We will present explicit formulae for four different case: deg $h = 0$, deg $h = 1$, deg $h = 2$ and deg $h = 3$. In the two latter cases, we try to find special curves which can lead to a significant speedup. The major speedup is obtained by simplify and cancel r in the expressions. For hyperelliptic curves of genus 3 and characteristic two there exist no supersingular cases [RS02]. I.e. for genus 3 HEC we can take special curves with h constant. Using these special curves, we can obtain explicit formulae with low complexity and optimum performance regarding the number of required field operations for the execution of the group operations.

4 Case deg $h = 0$

In this section we assume deg $h = 0$. One can obtain an isomorphic curve where $f_6 = f_4 = f_2 = 0$ and h_0 is divided by any α^7. To improve the efficiency of HEC, we hope that coefficients h_i are 'small' in an isomorphic curve, which allows the multiplication with it to be performed via additions. So we will choose α^7 such that $\frac{h_0}{\alpha^7}$ is 'small' in practical use. If we choose finite fields $GF(2^n)$ with $n \equiv 1(mod\ 3)$ or $n \equiv 2(mod\ 3)$ there are no elements $\alpha \in GF(2^n)$ such that $\alpha^7 = 1$ (the unit element of $GF(2^n)$). Therefore, there is always an α such that $\alpha^7 = h_0$. For $n \equiv 0(mod\ 3)$ this happens with probability 1/7. We obtain the isomorphic curve by using the following birational transformation of variables and dividing the equation by α^{14}:

$$Y \leftarrow \alpha^7\widetilde{Y} + m\widetilde{X}^2 + n\widetilde{X}, X \leftarrow \alpha^2\widetilde{X} + f_6$$

where $m = \alpha^4\sqrt{f_4 + f_5f_6}, n = \alpha^2\sqrt{f_2 + f_3f_6 + h_0m}$. So we obtain a curve of the form $Y^2 + h_0Y = X^7 + f_5X^5 + f_3X^3 + f_1X + f_0$, usually with $h_0 = 1$. Adding a constant term to the substitution of $\widetilde{Y}$ one can achieve $f_0 = 0$ with probability 1/2. Hence, there are only three parameters f_5, f_3, f_1 as opposed to five in the general case showing that the type is indeed special.

With the new curve coefficients the expression r and s will simplify to:

$$r = h_0^3, s_2^{'} = h_0^2 z_2, s_1^{'} = h_0^2 z_1, s_0^{'} = h_0^2 z_0.$$

We note that

$$u_3^{'} = 0, u_2^{'} = s_1^2 = (s_1^{'}/s_2^{'})^2 = (z_1/z_2)^2,$$

$$u_1^{'} = (r/s_2^{'})^2 = h_0^2(z_2^{-1})^2, u_0^{'} = s_0^2 = (s_0^{'}/s_2^{'})^2 = (z_0/z_2)^2,$$

and

$$v_3^{'} = (u_2^{'} + g_3)(s_2^{'}/r) = h_0^{-1}(u_2^{'} z_2 + z_0 + u_2 z_1 + u_1 z_2),$$

$$v_2^{'} = (g_4 u_2^{'} + u_1^{'} + g_2)(s_2^{'}/r) + v_2 = h_0^{-1}[(u_2 z_2 + z_1)u_2^{'} + u_1^{'} + u_2 z_0 + u_1 z_1 + u_0 z_2] + v_2,$$

$$v_1^{'} = (g_4 u_1^{'} + u_0^{'} + g_1)(s_2^{'}/r) + v_1 = h_0^{-1}[(u_2 z_2 + z_1)u_1^{'} + u_0^{'} + u_1 z_0 + u_0 z_1] + v_1,$$

$$v_0^{'} = (g_4 u_0^{'} + g_0)(s_2^{'}/r) + h_0 + v_0 = h_0^{-1}[(u_2 z_2 + z_1)u_0^{'} + u_0 z_0] + h_0 + v_0.$$

Since $f + hv + v^2 = uz + u^2 x$ we also have that

$$f_0 + h_0 v_0 + v_0^2 = u_0 z_0,$$

$$f_1 + h_0 v_1 = u_1 z_0 + u_0 z_1 + u_0^2,$$

$$h_0 v_2 + v_1^2 = u_2 z_0 + u_1 z_1 + u_0 z_2,$$

$$f_3 = z_0 + u_2 z_1 + u_1 z_2 + u_1^2,$$

$$v_2^2 = z_1 + u_2 z_2,$$

$$f_5 = u_2^2 + z_2.$$

Using the equations above, we can calculate cheaply $u_2^{'}, u_0^{'}$ and $v_3^{'}, v_2^{'}, v_1^{'}, v_0^{'}$ as follows:

$$u_2^{'} = (z_1/z_2)^2 = [(v_2^2 + u_2 z_2)/z_2]^2 = [v_2^2 z_2^{-1}]^2 + u_2^2,$$

$$u_0^{'} = (z_0/z_2)^2 = [(f_3 + u_1^2 + u_2 z_1 + u_1 z_2)/z_2]^2 = [(f_3 + u_1^2) z_2^{-1}]^2 + u_1^2 + u_2^2 u_2^{'},$$

$$v_3^{'} = h_0^{-1}(u_2^{'} z_2 + f_3 + u_1^2),$$

$$v_2^{'} = h_0^{-1}(v_2^2 u_2^{'} + u_1^{'} + v_1^2),$$

$$v_1^{'} = h_0^{-1}(v_2^2 u_1^{'} + u_0^{'} + f_1 + u_0^2),$$

$$v_0^{'} = h_0^{-1}(v_2^2 u_0^{'} + f_0 + v_0^2) + h_0.$$

We give the doubling formulae for this case in Table 1. The operations are counted for the case $h_0 = 1, h_0^{-1}$ is 'small' (multiplication with h_0^{-1} are not counted), and arbitrary h_0. Both h_0^2 and h_0^{-1} are precomputed.

Table 1. Doubling deg $h = 0$, deg $u = 3$

Input	$[u, v], u = x^3 + u_2x^2 + u_1x + u_0, v = v_2x^2 + v_1x + v_0; h_0^2, h_0^{-1}$
Output	$[u'', v''] = 2[u, v]$

Step	Expression	$h_0 = 1$	h_0^{-1} small	h_0 arbitrary
1	Compute $\tilde{u} = u^2$ and $\tilde{v} = v^2$: $\tilde{u}_2 = u_2^2, \tilde{u}_1 = u_1^2, \tilde{u}_0 = u_0^2, \tilde{v}_2 = v_2^2, \tilde{v}_1 = v_1^2, \tilde{v}_0 = v_0^2;$	$6S$	$6S$	$6S$
2	Compute $u' = x^4 + u_3'x^3 + u_2'x^2 + u_1'x + u_0'$: $z_2 = f_5 + \tilde{u}_2, t_1 = f_3 + \tilde{u}_1, t_2 = f_1 + \tilde{u}_0, t_3 = f_0 + \tilde{v}_0;$ If $z_2 = 0$ then call the Cantor algorithm $invz_2 = z_2^{-1}, u_3' = 0, u_2' = (\tilde{v}_2 invz_2)^2 + \tilde{u}_2;$ $u_1' = h_0^2(invz_2)^2, u_0' = (t_1 invz_2)^2 + \tilde{u}_1 + \tilde{u}_2u_2';$	$1I, 3M, 3S$	$1I, 4M, 3S$	$1I, 4M, 3S$
3	Compute $v' = v_3'x^3 + v_2'x^2 + v_1'x + v_0'$: $v_3' = h_0^{-1}(u_2'z_2 + t_1), v_2' = h_0^{-1}(\tilde{v}_2u_2' + u_1' + \tilde{v}_1);$ $v_1' = h_0^{-1}(\tilde{v}_2u_1' + u_0' + t_2), v_0' = h_0^{-1}(\tilde{v}_2u_0' + t_3);$	$4M$	$4M$	$8M$
4	Reduce u', i.e. $u'' = x^3 + u_2''x^2 + u_1''x + u_0''$: $u_2'' = v_3'^2, u_1'' = f_5 + u_2', u_0'' = u_2''u_2' + v_2'^2 + u_1';$	$1M, 2S$	$1M, 2S$	$1M, 2S$
5	Compute $v'' = v_2''x^2 + v_1''x + v_0''$: $v_2'' = v_2' + v_3'u_2'', v_1'' = v_1' + v_3'u_1'', v_0'' = v_0' + v_3'u_0'';$	$3M$	$3M$	$3M$
Sum		$1I, 11M, 11S$	$1I, 12M, 11S$	$1I, 16M, 11S$

[Remark]: In this case, we obtain the same explicit formula as in [GKP04]. However, we derived it from Harley's algorithm, whereas they from Cantor's algorithm. Compared with the explicit formula in [PWGP03], our formula saves $3M$. For genus 3 hyperellitpic curves defined over binary fields, this is the fastest doubling explicit formula so far.

5 Case deg $h = 1$

In this section we discuss the case of deg $h = 1$. One can obtain an isomorphic curve where $f_6 = h_0 = 0$ and h_1 is divided by any α^5. We will choose α^5 such that $\frac{h_1}{\alpha^5}$ is 'small' in practical use. If we choose finite fields $GF(2^n)$ with n not being divided by 4 there are no elements $\alpha \in GF(2^n)$ such that $\alpha^5 = 1$. Therefore, there is always an α such that $\alpha^5 = h_0$. For $n \equiv 0(mod\ 4)$ this happens with probability 1/5. We obtain the isomorphic curve by using the following birational transformation of variables and dividing the equation by α^{14}:

$$Y \leftarrow \alpha^7\widetilde{Y} + \alpha^6\sqrt{f_6 + \frac{h_0}{h_1}}\widetilde{X}^3, X \leftarrow \alpha^2\widetilde{X} + \frac{h_0}{h_1}$$

So we obtain a curve of the form $Y^2 + h_1XY = X^7 + f_5X^5 + f_4X^4 + f_3X^3 + f_2X^2 + f_1X + f_0$, usually with $h_1 = 1$. Adding a linear factor to the substitution of $\widetilde{Y}$ one can achieve $f_2 = 0$ with probability 1/2. A constant term leads to $f_1 = 0$. Therefore, there are only four free parameters f_5, f_4, f_3, f_0.

With the new curve coefficients the expression r and s will simplify to:

$$r = u_0h_1^3, s_2' = z_0h_1^2, s_1' = (u_2z_0 + u_0z_2)h_1^2, s_0' = (u_1z_0 + u_0z_1)h_1^2,$$

$$rs_2' = u_0z_0h_1^5, s_2 = \frac{s_2'}{r} = \frac{z_0}{u_0h_1}.$$

In this case, we have that

$$u_3' = 0, u_2' = s_1^2 = (s_1'/s_2')^2 = (u_2 + u_0 \cdot \frac{z_2}{z_0})^2,$$

$$u_1' = (r/s_2')^2 = h_0^2u_0^2(z_0^{-1})^2, u_0' = s_0^2 = (s_0'/s_2')^2 = (u_1 + u_0 \cdot \frac{z_1}{z_0})^2,$$

and

$$v_3' = (u_2' + g_3)(s_2'/r) = h_1^{-1}[z_2 \cdot (u_0 \cdot \frac{z_2}{z_0}) + z_1 + u_2z_2],$$

$$v_2' = (g_4u_2' + u_1' + g_2)(s_2'/r) + v_2 = h_1^{-1}[z_2u_2' + \frac{h_1}{s_2} + u_2z_1 + u_1z_2 + z_0] + v_2,$$

$$v_1' = (g_4u_1' + u_0' + g_1)(s_2'/r) + h_1 + v_1 = h_1^{-1}[\frac{1}{h_1s_2}(z_2 \cdot \frac{h_1}{s_2} + z_1^2) + u_2z_0 + u_1z_1 + u_0z_2] + v_1,$$

$$v_0' = (g_4u_0' + g_0)(s_2'/r) + v_0 = h_1^{-1}(z_2u_0' + u_1z_0 + u_0z_1) + v_0.$$

Since $f + hv + v^2 = uz + u^2x$ we also obtain that

$$f_0 + v_0^2 = u_0z_0 \quad (= rs_2'/h_1^5),$$

$$f_1 + h_1v_0 = u_1z_0 + u_0z_1 + u_0^2,$$

$$f_2 + h_1v_1 + v_1^2 = u_2z_0 + u_1z_1 + u_0z_2,$$

$$f_3 + h_1v_2 = z_0 + u_2z_1 + u_1z_2 + u_1^2,$$

$$f_4 + v_2^2 = z_1 + u_2z_2,$$

$$f_5 = u_2^2 + z_2.$$

Using the equations above, we can calculate cheaply u_2', u_0' and v_3', v_2', v_1', v_0' as follows:

$$u_2' = (u_2 + u_0 \cdot \frac{z_2}{z_0})^2 = (u_2 + u_0 \cdot \frac{z_2}{u_0h_1s_2})^2 = (u_2 + \frac{z_2}{h_1s_2})^2,$$

$$u_0' = (u_1 + u_0 \cdot \frac{z_1}{z_0})^2 = (u_1 + u_0 \cdot \frac{z_1}{u_0h_1s_2})^2 = (u_1 + \frac{z_1}{h_1s_2})^2,$$

$$v_3' = h_1^{-1}(\frac{z_2^2}{h_1s_2} + f_4 + v_2^2),$$

$$v_2' = h_1^{-1}(z_2u_2' + \frac{h_1}{s_2} + f_3 + u_1^2),$$

$$v_1' = h_1^{-1}[\frac{1}{h_1s_2}(z_2 \cdot \frac{h_1}{s_2} + z_1^2) + f_2 + v_1^2],$$

$$v_0' = h_1^{-1}(z_2u_0' + f_1 + u_0^2).$$

We note that $f_0 + v_0^2 = u_0z_0 = rs_2'/h_1^5$, so it is very cheap to calculate rs_2' as the exact coefficients of z are not necessary. In Table 2, we present the doubling formula for this case. The operations are counted for the case $h_1 = 1, h_1^{-1}$ is 'small' (multiplication with h_1^{-1} are not counted), and arbitrary h_1. Both h_1^2 and h_1^{-1} are precomputed. In Step 2 the inversion and multiplication with k_0 can also be replaced by a division as the inverse is not used later on.

Table 2. Doubling deg $h = 1$, deg $u = 3$

Input	$[u, v], u = x^3 + u_2x^2 + u_1x + u_0, v = v_2x^2 + v_1x + v_0; h_1^2, h_1^{-1}$
Output	$[u^{''}, v^{''}] = 2[u, v]$

Step	Expression	$h_1 = 1$	h_1^{-1} small	h_1 arbitrary
1	Compute $rs_2^{'}$: $k_0 = u_0^2, z_2 = f_5 + u_2^2, t_1 = f_4 + v_2^2$; $z_1 = t_1 + u_2z_2, w_0 = f_0 + v_0^2\ (= rs_2^{'}/h_1^5)$; If $w_0 = 0$ then call the Cantor algorithm	$1M, 4S$	$1M, 4S$	$1M, 4S$
2	Compute $1/h_1s_2$ and s_1, s_0: $w_1 = (1/w_0) \cdot k_0\ (= 1/h_1s_2), k_1 = z_2w_1$; $k_2 = z_1w_1, s_1 = u_2 + k_1, s_0 = u_1 + k_2$;	$1I, 3M$	$1I, 3M$	$1I, 3M$
3	Compute $u^{'} = x^4 + u_3^{'}x^3 + u_2^{'}x^2 + u_1^{'}x + u_0^{'}$: $w_2 = h_1^2w_1\ (= h_1/s_2), u_3^{'} = 0$; $u_2^{'} = s_1^2, u_1^{'} = w_2w_1, u_0^{'} = w_2 + s_0^2$;	$3S$	$2M, 2S$	$2M, 2S$
4	Compute $v^{'} = v_3^{'}x^3 + v_2^{'}x^2 + v_1^{'}x + v_0^{'}$: $v_3^{'} = h_1^{-1}(z_2k_1 + t_1), v_2^{'} = h_1^{-1}(z_2u_2^{'} + w_2 + f_3 + u_1^2)$; $v_1^{'} = h_1^{-1}[w_1(z_2w_2 + z_1^2) + f_2 + v_1^2]$; $v_0^{'} = h_1^{-1}(z_2u_0^{'} + f_1 + u_0^2)$;	$5M, 4S$	$5M, 4S$	$9M, 4S$
5	Reduce $u^{'}$, i.e. $u^{''} = x^3 + u_2^{''}x^2 + u_1^{''}x + u_0^{''}$: $u_2^{''} = v_3^{'2}, u_1^{''} = f_5 + u_2^{'}$; $u_0^{''} = f_4 + u_2^{''}u_2^{'} + v_2^{'2} + u_1^{'} + h_1v_3^{'}$;	$1M, 2S$	$2M, 2S$	$2M, 2S$
6	Compute $v^{''} = v_2^{''}x^2 + v_1^{''}x + v_0^{''}$: $v_2^{''} = v_2^{'} + v_3^{'}u_2^{''}, v_1^{''} = v_1^{'} + v_3^{'}u_1^{''} + h_1, v_0^{''} = v_0^{'} + v_3^{'}u_0^{''}$;	$3M$	$3M$	$3M$
Sum		$1I, 13M, 13S$	$1I, 16M, 12S$	$1I, 20M, 12S$

[Remark]: The algorithm in [GKP04] needs $1I, 44M, 6S$ to compute divisor class doubling. However, our derived explicit formula needs only $1I, 13M, 13S$ in this case. Compared with the explicit formula in [GKP04], our formula save $31M$ at the cost of extra $7S$.

6 Case deg $h = 2$

If h is of degree two then we cannot make any of its coefficients zero in general. In this section we will discuss special curves with $h_1 = 0$, that is, the curves having the form $Y^2 + (h_2X^2 + h_0)Y = X^7 + f_6X^6 + f_5X^5 + f_4X^4 + f_3X^3 + f_2X^2 + f_1X + f_0$, which allows for a significant speedup. By making a change of coordinates we can obtain $f_5 = f_3 = f_2 = h_0 = 0$ and h_2 is divided by any α^3. We will choose α^3 such that $\frac{h_2}{\alpha^3}$ is 'small' in practical use. If, as usual, one choose finite $GF(2^n)$ with n odd there are no non-trivial cube roots of unity. Hence, there is always an α such that $\alpha^3 = h_2$. For even n this happens with probability 1/3. The isomorphic curve is obtained by using the following birational transformation of variables and dividing the equation by α^{14}:

$$Y \leftarrow \alpha^7\widetilde{Y} + m\widetilde{X}^3 + s\widetilde{X} + t, X \leftarrow \alpha^2\widetilde{X} + \beta$$

where $\beta = \sqrt{\frac{h_0}{h_2}}, m = \alpha^6 \cdot \frac{f_5+\beta^2}{h_2}, s = \alpha^2 \cdot \frac{f_3+\beta^4}{h_2}$ and $t = \frac{h_2^2(f_2+f_3\beta+f_6\beta^4+\beta^5)+f_3^2+\beta^8}{h_2^3}$. So we obtain a curve of the form $Y^2 + h_2X^2Y = X^7 + f_6X^6 + f_4X^4 + f_1X + f_0$, usually with $h_2 = 1$. Adding a quadratic factor to the substitution of $\widetilde{Y}$ one can achieve $f_4 = 0$ with probability 1/2. Accordingly, there are only three free parameters f_6, f_1, f_0.

Then the expressions for r and s will simplify to:

$$r = u_0^2h_2^3, s_2^{'} = (u_1z_0 + u_0z_1)h_2^2, s_1^{'} = [u_2(u_1z_0 + u_0z_1) + u_0z_0]h_2^2,$$

$$s_0^{'} = [u_1(u_1z_0 + u_0z_1) + u_0(u_2z_0 + u_0z_2)]h_2^2, s_1 = \frac{s_1^{'}}{s_2^{'}} = u_2 + k_1, s_0 = \frac{s_0^{'}}{s_2^{'}} = u_1 + k_2,$$

where $k_1 = \frac{u_0 z_0}{u_1 z_0 + u_0 z_1}$ and $k_2 = \frac{u_0(u_2 z_0 + u_0 z_2)}{u_1 z_0 + u_0 z_1}$. In this case, we have that

$$u_3^{'} = 0, u_2^{'} = s_1^2, u_1^{'} = \frac{r}{s_2^{'}}(h_2 + \frac{r}{s_2^{'}}) = h_2^2 w_1(1 + w_1),$$

$$u_0^{'} = \frac{r}{s_2^{'}}[h_2(u_2 + s_1) + \frac{r f_6}{s_2^{'}}] + s_0^2 = h_2^2 w_1(k_1 + f_6 w_1) + s_0^2,$$

where $w_1 = \frac{u_0^2}{u_1 z_0 + u_0 z_1}$ and

$$v_3^{'} = (u_2^{'} + g_3)(s_2^{'}/r) = h_2^{-1}[z_2 + \frac{(u_0 z_0)^2}{u_0^2(u_1 z_0 + u_0 z_1)}],$$

$$v_2^{'} = (g_4 u_2^{'} + u_1^{'} + g_2)(s_2^{'}/r) + h_2 + v_2 = h_2^{-1}[z_1 + u_2 z_2 + \frac{(u_0 z_0) k_1^2}{u_0^2}] + h_2 w_1 + v_2,$$

$$v_1^{'} = (g_4 u_1^{'} + u_0^{'} + g_1)(s_2^{'}/r) + v_1 = h_2^{-1}[z_0 + u_1 z_2 + u_2 z_1 + \frac{(u_2 z_0 + u_0 z_2)^2}{u_1 z_0 + u_0 z_1}] + (h_2 w_1)(f_6 + k_1) + v_1,$$

$$v_0^{'} = (g_4 u_0^{'} + g_0)(s_2^{'}/r) + v_0 = h_2^{-1}[u_2 z_0 + u_1 z_1 + u_0 z_2 + \frac{(u_0 z_0) k_2^2}{u_0^2}] + (h_2 k_1)(k_1 + f_6 w_1) + v_0.$$

And since $f + hv + v^2 = uz + u^2(x + f_6)$ we also have that

$$f_0 + v_0^2 = u_0 z_0 + f_6 u_0^2,$$

$$f_1 = u_1 z_0 + u_0 z_1 + u_0^2,$$

$$h_2 v_0 = u_2 z_0 + u_1 z_1 + u_0 z_2 + f_6 u_1^2,$$

$$h_2 v_1 = z_0 + u_2 z_1 + u_1 z_2 + u_1^2,$$

$$f_4 + h_2 v_2 + v_2^2 = z_1 + u_2 z_2 + f_6 u_2^2,$$

$$0 = u_2^2 + z_2.$$

We use the equations above to calculate k_1, k_2, w_1 and $v_3^{'}, v_2^{'}, v_1^{'}, v_0^{'}$ cheaper:

$$k_1 = \frac{f_0 + v_0^2 + f_6 u_0^2}{f_1 + u_0^2}, k_2 = \frac{u_0(h_2 v_0 + u_1 z_1 + f_6 u_1^2)}{f_1 + u_0^2}, w_1 = \frac{u_0^2}{f_1 + u_0^2},$$

$$v_3^{'} = h_2^{-1}[z_2 + \frac{(f_0 + v_0^2 + f_6 u_0^2)^2}{u_0^2(f_1 + u_0^2)}],$$

$$v_2^{'} = h_2^{-1}[f_4 + v_2^2 + f_6 u_2^2 + \frac{(f_0 + v_0^2 + f_6 u_0^2) k_1^2}{u_0^2}] + \frac{h_2 u_0^2}{f_1 + u_0^2},$$

$$v_1^{'} = h_2^{-1}[u_1^2 + \frac{(u_1 z_1 + f_6 u_1^2 + h_2 v_0)^2}{f_1 + u_0^2}] + \frac{h_2 u_0^2 (f_6 + k_1)}{f_1 + u_0^2},$$

$$v_0^{'} = h_2^{-1}[f_6 u_1^2 + \frac{(f_0 + v_0^2 + f_6 u_0^2) k_2^2}{u_0^2}] + (h_2 k_1)(k_1 + \frac{u_0^2 f_6}{f_1 + u_0^2}).$$

We note that $r s_2^{'} = u_0^2(u_1 z_0 + u_0 z_1) h_2^5 = u_0^2(f_1 + u_0^2) h_2^5$, so it is very cheap to calculate $r s_2^{'}$ since we need not know the exact coefficients of z. We describe the doubling formula for this case in Table 3. The operations are counted for the case $h_2 = 1, h_2^{-1}$ is 'small' (multiplication with h_2^{-1} are not counted), and arbitrary h_2. Both h_2^2 and h_2^{-1} are precomputed.

Table 3. Doubling deg $h = 2$, $h_1 = 0$, deg $u = 3$

Input	$[u, v], u = x^3 + u_2x^2 + u_1x + u_0, v = v_2x^2 + v_1x + v_0; h_2^2, h_2^{-1}$			
Output	$[u'', v''] = 2[u, v]$			
Step	Expression	$h_2 = 1$	h_2^{-1} small	h_2 arbitrary
1	Precomputation: $\tilde{u}_2 = u_2^2, \tilde{u}_1 = u_1^2, \tilde{u}_0 = u_0^2, z_2 = f_4 + v_2^2 + f_6\tilde{u}_2$; $z_1 = z_2 + h_2v_2 + \tilde{u}_2u_2, t_1 = f_0 + v_0^2 + f_6\tilde{u}_0$; $t_2 = f_6\tilde{u}_1, t_3 = t_2 + h_2v_0 + u_1z_1, t_4 = f_1 + \tilde{u}_0$; If $t_4 = 0$ then call the Cantor algorithm	$5M, 5S$	$7M, 5S$	$7M, 5S$
2	Compute s_1, s_0: $t_5 = (t_4\tilde{u}_0)^{-1}, t_6 = t_4t_5, t_7 = \tilde{u}_0t_5, t_8 = t_1t_6, k_1 = t_1t_7$; $\tilde{k}_2 = t_3t_7, k_2 = u_0\tilde{k}_2, s_1 = u_2 + k_1, s_0 = u_1 + k_2$;	$1I, 7M$	$1I, 7M$	$1I, 7M$
3	Compute $u' = x^4 + u_3'x^3 + u_2'x^2 + u_1'x + u_0'$: $w_1 = \tilde{u}_0t_7, w_2 = h_2^2w_1, u_2' = s_1^2, u_1' = w_2(1 + w_1)$;	$1M, 2S$	$3M, 1S$	$3M, 1S$
4	Compute $v' = v_3'x^3 + v_2'x^2 + v_1'x + v_0'$: $v_3' = h_2^{-1}(\tilde{u}_2 + t_1^2t_5), v_2' = h_2^{-1}(z_2 + t_8k_1^2) + h_2w_1$; $v_1' = h_2^{-1}(\tilde{u}_1 + \tilde{k}_2t_3) + (h_2w_1)(f_6 + k_1)$; $v_0' = h_2^{-1}(t_2 + t_8k_2^2) + (h_2k_1)(k_1 + f_6w_1)$;	$7M, 3S$	$9M, 3S$	$13M, 3S$
5	Reduce u', i.e. $u'' = x^3 + u_2''x^2 + u_1''x + u_0''$: $u_2'' = f_6 + v_3'^2, u_1'' = u_2' + h_2v_3'$; $u_0'' = f_4 + u_2''u_2' + u_1' + v_2'^2 + h_2v_2'$;	$1M, 2S$	$3M, 1S$	$3M, 1S$
6	Compute $v'' = v_2''x^2 + v_1''x + v_0''$: $v_2'' = v_2' + v_3'u_2'' + h_2, v_1'' = v_1' + v_3'u_1'', v_0'' = v_0' + v_3'u_0''$;	$3M$	$3M$	$3M$
Sum		$1I, 24M, 12S$	$1I, 32M, 10S$	$1I, 36M, 10S$

[Remark]: For general case with $h(x) = h_2x^2 + h_1x + h_0$, the authors using a birational transformation to make the curve's coefficient f_6 zero [GKP04]. Their algorithm needs $1I, 52M, 8S$ to compute divisor class doubling. Using special curves with $h(x) = h_2x^2 + h_0$, our explicit formula needs only $1I, 24M, 12S$ for $h_2 = 1$. Compared with the explicit formula in [GKP04], our formula saves $28M$ at the cost of extra $4S$ in the best case. In the formulae presented in Table 3 there are four counted multiplications with f_6 which are cheaper when f_6 is 'small'.

7 Case deg $h = 3$

When h is of degree three, we cannot also make any of its coefficients zero in general. We will show that special curves with $h_2 = h_1 = h_0 = 0$ can obtain excellent performance in this section. We can construct a change of coordinates to make $f_5 = f_4 = f_3 = 0$ and $h_3 = 1$. The isomorphic curve is obtained by using the following birational transformation of variables and dividing the equation by h_3^{14}:

$$Y \leftarrow h_3^7\widetilde{Y} + h_3^3f_5\widetilde{X}^2 + \frac{f_4h_3^2 + f_5^2}{h_3}\widetilde{X} + \frac{f_3}{h_3}, X \leftarrow h_3^2\widetilde{X}$$

So we obtain a curve of the form $Y^2 + X^3Y = X^7 + f_6X^6 + f_2X^2 + f_1X + f_0$. Adding a cube factor to the substitution of $\widetilde{Y}$ one can achieve $f_6 = 0$ with probability $1/2$. Thereby, there are only three free parameters f_2, f_1, f_0.

Then the expressions for r and s will simplify to:

$$r = u_0^3, s_2' = u_0(u_2z_0 + u_1z_1 + u_0z_2) + u_1^2z_0, s_1' = u_2[u_0(u_2z_0 + u_1z_1 + u_0z_2)] + u_0(u_1z_0 + u_0z_1),$$

$$s_0' = u_1[u_0(u_2z_0 + u_1z_1 + u_0z_2)] + u_0[u_2(u_1z_0 + u_0z_1) + u_0z_0], s_1 = \frac{s_1'}{s_2'} = u_2 + k_1, s_0 = \frac{s_0'}{s_2'} = u_1 + k_2,$$

where $k_1 = \frac{u_0(u_1z_0+u_0z_1)}{u_0(u_2z_0+u_1z_1+u_0z_2)+u_1^2z_0}$ and $k_2 = \frac{u_0[u_2(u_1z_0+u_0z_1)+u_0z_0]}{u_0(u_2z_0+u_1z_1+u_0z_2)+u_1^2z_0}$. In this case, we have that

$$u_3^{'} = 0, u_2^{'} = s_1^2 + \frac{r}{s_2^{'}}, u_1^{'} = \frac{r}{s_2^{'}}(k_1 + \frac{r}{s_2^{'}}), u_0^{'} = \frac{r}{s_2^{'}}(k_2 + u_2k_1 + \frac{rf_6}{s_2^{'}}) + s_0^2,$$

$$v_3^{'} = (u_2^{'} + g_3)(s_2^{'}/r) + 1 = \frac{u_0z_0}{u_0^2} + \frac{(u_1z_0+u_0z_1)^2}{u_0^2(u_2z_0+u_1z_1+u_0z_2)+u_1^2(u_0z_0)},$$

$$v_2^{'} = (g_4u_2^{'} + u_1^{'} + g_2)(s_2^{'}/r) + v_2 = \frac{(u_1z_0+u_0z_1)(u_2^{'}+u_2^2)}{u_0^2} + z_2 + k_1 + \frac{r}{s_2^{'}} + v_2,$$

$$v_1^{'} = (g_4u_1^{'} + u_0^{'} + g_1)(s_2^{'}/r) + v_1 = \frac{k_2[u_2(u_1z_0+u_0z_1)+u_0z_0] + u_2^2(u_0z_0)}{u_0^2} +$$

$$k_1(k_1 + \tfrac{r}{s_2^{'}}) + (k_2 + u_2k_1 + \tfrac{rf_6}{s_2^{'}}) + (z_1 + u_2z_2) + v_1,$$

$$v_0^{'} = (g_4u_0^{'} + g_0)(s_2^{'}/r) + v_0 = \frac{(u_1z_0+u_0z_1)(u_0^{'}+u_1^2)}{u_0^2} + (z_0 + u_2z_1 + u_1z_2) + v_0.$$

And since $f + hv + v^2 = uz + u^2(x + f_6)$ we also have that

$$f_0 + v_0^2 = u_0z_0 + f_6u_0^2,$$
$$f_1 = u_1z_0 + u_0z_1 + u_0^2,$$
$$f_2 + v_1^2 = u_2z_0 + u_1z_1 + u_0z_2 + f_6u_1^2,$$
$$v_0 = z_0 + u_2z_1 + u_1z_2 + u_1^2,$$
$$v_1 + v_2^2 = z_1 + u_2z_2 + f_6u_2^2,$$
$$v_2 = u_2^2 + z_2.$$

Using the equations above, we can calculate $k_1, k_2, \frac{r}{s_2^{'}}$ and $v_3^{'}, v_2^{'}, v_1^{'}, v_0^{'}$ cheaper:

$$k_1 = \frac{u_0^2(f_1+u_0^2)}{u_0^2(f_2+v_1^2+f_6u_1^2)+u_1^2(f_0+v_0^2+f_6u_0^2)},$$

$$k_2 = \frac{u_0^2[u_2(f_1+u_0^2)+(f_0+v_0^2+f_6u_0^2)]}{u_0^2(f_2+v_1^2+f_6u_1^2)+u_1^2(f_0+v_0^2+f_6u_0^2)},$$

$$\frac{r}{s_2^{'}} = \frac{u_0^4}{u_0^2(f_2+v_1^2+f_6u_1^2)+u_1^2(f_0+v_0^2+f_6u_0^2)},$$

$$v_3^{'} = \frac{f_0+v_0^2+f_6u_0^2}{u_0^2} + \frac{(f_1+u_0^2)^2}{u_0^2(f_2+v_1^2+f_6u_1^2)+u_1^2(f_0+v_0^2+f_6u_0^2)},$$

$$v_2^{'} = \frac{(f_1+u_0^2)(u_2^{'}+u_2^2)}{u_0^2} + k_1 + \frac{r}{s_2^{'}} + u_2^2,$$

$$v_1^{'} = \frac{k_2[u_2(f_1+u_0^2)+(f_0+v_0^2+f_6u_0^2)] + u_2^2(f_0+v_0^2+f_6u_0^2)}{u_0^2} +$$

$$k_1(k_1 + \tfrac{r}{s_2^{'}}) + (k_2 + u_2k_1 + \tfrac{rf_6}{s_2^{'}}) + (v_2^2 + f_6u_2^2),$$

$$v_0^{'} = \frac{(f_1+u_0^2)(u_0^{'}+u_1^2)}{u_0^2} + u_1^2.$$

We note that $rs_2^{'} = u_0^2[u_0^2(u_2z_0+u_1z_1+u_0z_2)+u_1^2(u_0z_0)] = u_0^2[u_0^2(f_2+v_1^2+f_6u_1^2)+u_1^2(f_0+v_0^2+f_6u_0^2)]$, so we can calculate $rs_2^{'}$ cheaply without knowing the exact coefficients of z. We present the explicit formula for this case in Table 4.

Table 4. Doubling deg $h = 3$, $h_2 = h_1 = h_0 = 0$, deg $u = 3$

Input	$[u, v], u = x^3 + u_2x^2 + u_1x + u_0, v = v_2x^2 + v_1x + v_0;$
Output	$[u'', v''] = 2[u, v]$

Step	Expression	Cost
1	Precomputation: $\tilde{u}_2 = u_2^2, \tilde{u}_1 = u_1^2, \tilde{u}_0 = u_0^2, \tilde{v}_2 = v_2^2, \tilde{v}_1 = v_1^2, \tilde{v}_0 = v_0^2, t_1 = f_0 + \tilde{v}_0 + f_6\tilde{u}_0;$ $t_2 = f_2 + \tilde{v}_1 + f_6\tilde{u}_1, t_3 = \tilde{u}_0t_2 + \tilde{u}_1t_1, t_4 = f_1 + \tilde{u}_0, t_5 = u_2t_4 + t_1;$ If $t_3 = 0$ then call the Cantor algorithm	$5M, 6S$
2	Compute s_1, s_0: $t_6 = (t_3\tilde{u}_0)^{-1}, t_7 = t_3t_6, t_8 = \tilde{u}_0t_6, t_9 = \tilde{u}_0t_8;$ $k_1 = t_4t_9, k_2 = t_5t_9, s_1 = u_2 + k_1, s_0 = u_1 + k_2;$	$1I, 6M$
3	Compute $u' = x^4 + u'_3x^3 + u'_2x^2 + u'_1x + u'_0$: $w_4 = \tilde{u}_0t_9, u'_2 = s_1^2 + w_4, t_{10} = k_1 + w_4, u'_1 = w_4t_{10};$ $t_{11} = k_2 + u_2k_1 + f_6w_4, u'_0 = s_0^2 + w_4t_{11};$	$5M, 2S$
4	Compute $v' = v'_3x^3 + v'_2x^2 + v'_1x + v'_0$: $v'_3 = t_1t_7 + t_4^2t_8, t_{12} = t_4t_7, v'_2 = t_{12}(u'_2 + \tilde{u}_2) + k_1 + w_4 + \tilde{u}_2;$ $v'_1 = k_1t_{10} + t_{11} + (k_2t_5 + \tilde{u}_2t_1)t_7 + \tilde{v}_2 + f_6\tilde{u}_2, v'_0 = t_{12}(u'_0 + \tilde{u}_1) + \tilde{u}_1;$	$10M, 1S$
5	Reduce u', i.e. $u'' = x^3 + u''_2x^2 + u''_1x + u''_0$: $u''_2 = f_6 + v'_3 + v'^2_3, u''_1 = u'_2 + v'_2, u''_0 = u''_2u'_2 + u'_1 + v'^2_2 + v'_1;$	$1M, 2S$
6	Compute $v'' = v''_2x^2 + v''_1x + v''_0$: $v''_2 = v'_2 + (v'_3 + 1)u''_2, v''_1 = v'_1 + (v'_3 + 1)u''_1, v''_0 = v'_0 + (v'_3 + 1)u''_0;$	$3M$
Sum		$1I, 30M, 11S$

[Remark]: In [GKP04], the authors discuss two types of curves with $h_2 = 0$ and $f_6 = 0$, respectively. Their doubling formulae cost $1I, 63M, 9S$ and $1I, 64M, 5S$ for this two different cases. We note that using special curves with $h(x) = h_3x^3$ can lead to fast computation of a divisor class doubling. We derive the new explicit doubling formula which needs only $1I, 30M, 11S$. Compared with the algorithms in [GKP04], our formula saves $33M$ at the cost of extra $2S$. In addition, there are four counted multiplications with f_6 which can be computed cheaply when f_6 is 'small' in the formulae.

8 Summary

Depending on the degree of h, we derived the corresponding explicit formulae which can compute doublings fast in the previous sections. For h of degree 0 and 1 the case f_6 not small does not apply since we make it zero by isomorphic transformations. We also find fast doubling formulae for special curves when the degree of h is 2 and 3. All results are summarized in Table 5.

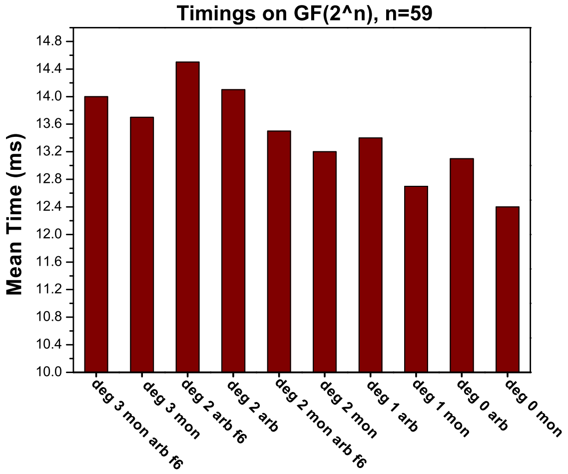
Table 5. Overview

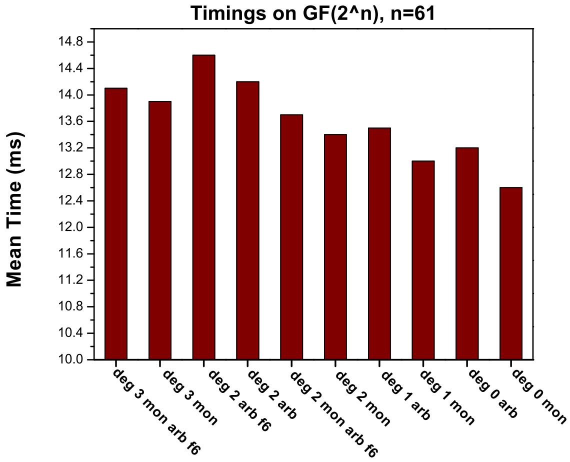
$h(x)$	$h(x) = h_0$			$h(x) = h_1x$		
h_i	$h_0 = 1$	h_0^{-1} small	h_0 arb.	$h_1 = 1$	h_1^{-1} small	h_1 arb.
cost	$1I, 11M, 11S$	$1I, 12M, 11S$	$1I, 16M, 11S$	$1I, 13M, 13S$	$1I, 16M, 12S$	$1I, 20M, 12S$
$h(x)$	$h(x) = h_2x^2$			$h(x) = x^3$		
h_i	$h_2 = 1$	h_2^{-1} small	h_2 arb.	—		
f_6 small	$1I, 20M, 12S$	$1I, 28M, 10S$	$1I, 32M, 10S$	$1I, 26M, 11S$		
f_6 arb.	$1I, 24M, 12S$	$1I, 32M, 10S$	$1I, 36M, 10S$	$1I, 30M, 11S$		

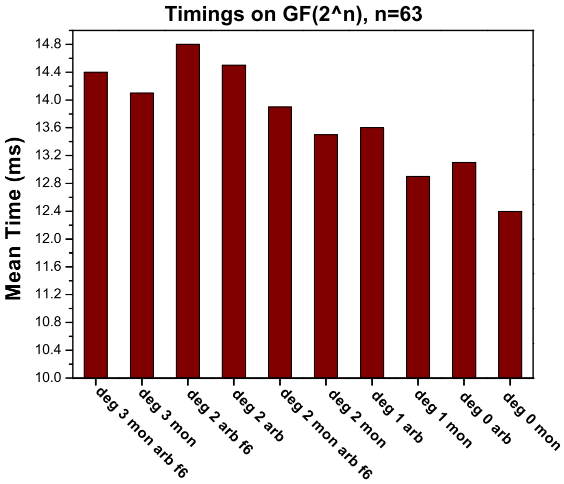
9 Experimental Results

In order to test the performance of our new doubling formulae, we implemented genus 3 HECC over 3 binary fields. Due to the attack proposed by Thériault [Thé03], we should select at least 56-bit finite fields in order to obtain the same security as a 160-bit elliptic curve cryptosystem. So we used binary fields

$GF(2^{59})$, $GF(2^{61})$ and $GF(2^{63})$. For $GF(2^{59})$ and $GF(2^{61})$, we used the minimal weight irreducible pentanomial $x^{59} + x^7 + x^4 + x^2 + 1$ and $x^{61} + x^5 + x^2 + x + 1$ to construct finite fields, respectively. However, for $GF(2^{63})$, we used the minimal weight irreducible trinomial $x^{63} + x + 1$ as field extension. Efficient algorithms summarized in [Pel02] were used to perform the field arithmetic. In addition, We used a NAF method to perform the scalar multiplication. All tests are implemented on a Pentium-M @1.5 GHz processor and with C programming language. The experimental results were depicted in three bar graphs.







In the graphs above we include the following ten cases respectively:

- deg 3 mon arb f_6: The case where deg $h = 3, h_2 = h_1 = h_0 = 0, f_6 \neq 0$ and $h_3 = 1$;
- deg 3 mon: The case where deg $h = 3, h_2 = h_1 = h_0 = 0, f_6 = 0$ and $h_3 = 1$;
- deg 2 arb f_6: The case where deg $h = 2, h_1 = h_0 = 0, f_6 \neq 0$;
- deg 2 arb: The case where deg $h = 2, h_1 = h_0 = 0, f_6 = 0$;
- deg 2 mon arb f_6: The case where deg $h = 2, h_1 = h_0 = 0, f_6 \neq 0$ and $h_2 = 1$;

- deg 2 mon: The case where deg $h = 2, h_1 = h_0 = 0, f_6 = 0$ and $h_2 = 1$;
- deg 1 arb: The case where deg $h = 1, h_0 = 0$;
- deg 1 mon: The case where deg $h = 1, h_0 = 0$ and $h_1 = 1$;
- deg 0 arb: The case where deg $h = 0$;
- deg 0 mon: The case where deg $h = 1$ and $h_0 = 1$;

10 Conclusion and Outlook

We have discussed how to accelerate the computation of divisor class doublings for genus 3 hyperelliptic curves defined over binary fields and given explicit formulae for all types of curves. Compared with the results in [PWGP03, GKP04], our explicit formulae have reduced some field operations further and shown excellent performance on a Pentium-M processor. The divisor addition formulae depend far less on the coefficients of h. Several authors have improved the corresponding addition explicit formulae according to the degree of h in [GKP04].

Side channel attacks (SCA) are powerful attacks which use a priori innocuous information such as power consumption or timings to break implementations of cryptosystem [Koc96, KJJ99]. On lightweight crypto-devices such as smartcards and PDAs, side channel attacks are a major threat. In order to implement genus 3 HECC securely on all kinds of embedded processors, we need consider countermeasures to thwart SCA because of the remarkable difference of the operation counts for addition and doubling. For example, in the case of $h(x) = 1$, the operation number of doubling is only one third that of addition. In [Thé04], the author presented two integer recoding algorithms which are resistant to SPA attacks and suitable especially to our case where the doubling operations are significantly fast than a group addition. To avoid DPA we can employ the countermeasures proposed in [Ava03].

In this paper we restricted our attentions to affine coordinate system. However, Xinxin Fan et. al has obtained inversion-free explicit formulae for genus 3 hyperelliptic curves [FWW05]. For genus 3 HECC defined over binary fields, the authors gave only inversion-free explicit formulae for special curves with $h(x) = 1$. So how to extended the idea of this paper to projective coordinate system and improve the formulae in [FWW05] will be the next logical step to accelerate the implementation for genus 3 HECC.

References

[Ava03] R. M. Avanzi. Countermeasures Against Differential Power Analysis for Hyperelliptic Curve Cryptosystems. In *Workshop on Cryptographic Hardware and Embedded Systems - CHES 2003*, volume LNCS 2779, pp. 366-381, Springer-Verlag, 2003.

[Ava04] R. M. Avanzi. Aspects of Hyperelliptic Curves over Large Prime Fields in Software Implementations. In M. Joye and J.-J. Quisquater, editors, *Workshop on Cryptographic Hardware and Embedded Systems - CHES 2004*, volume LNCS 3156, pp. 148-162, Springer-Verlag, 2004.

[BCLW02] N. Boston, T. Clancy, Y. Liow, and J. Webster. Genus Two Hyperelliptic Curve Coprocessor. In B. S. Kaliski, Ç. K. Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume LNCS 2523, pp. 529 - 539. Springer-Verlag, 2002. Updated version available at http://www.cs.umd.edu/ clancy/docs/hec-ches2002.pdf.

[BD04] B. Byramjee and S. Duqesne. Classification of genus 2 curves over F_{2^n} and optimazation of their arithmetic. *Cryptology ePrint Archieve*, Report 2004/107, 2004. http://eprint.iacr.org/.

[Can87] D.G.Cantor. Computing In The Jacobian Of A Hyperelliptic Curve. *Math. Comp.* 48:95-101, 1987.

[Cla02] T. Clancy. *Analysis of FPGA-based Hyperelliptic Curve Cryptosystems.* Master's thesis, University of Illinois Urbana-Champaign, December 2002.

[EMY04] G. Elias, A. Miri and T.H. Yeap. High-Performance, FPGA-Based Hyperelliptic Curve Cryptosystems. In *The Proceeding of the 22nd Biennial Symposium on Communications*, May 2004, Queen's University, Kingston, Ontario, Canada.

[FWW05] X. Fan, T. Wollinger, and Y. Wang. Inversion-Free Arithmetic on Genus 3 Hyperelliptic Curves and Its Implementations. *International Conference on Information Technology: Coding and Computing - ITCC*, April 11 - 13, 2005, Las Vegas, USA.

[GH00] P.Gaudry and R. Harley. Counting Points on Hyperelliptic Curves over Finite Fields. In *ANTS-IV*, ser. LNCS 1838, W.Bosma, Ed. Berlin: Springer-Verlag, pp. 297 – 312, 2000.

[GKP04] C. Guyot, K. Kaveh, V.M. Patankar. Explicit Algorithm for The Arithmetic on The Hyperelliptic Jacobians of Genus 3. *Journal of Ramanujan Mathematical Society*, 19 (2004), No.2, 119-159.

[GLS00] C. Günther, T. Lange, and A. Stein. Speeding up the Arithmetic on Koblitz Curves of Genus Two. In *Selected Areas in Cryptography - SAC 2000*, Volume 2012, Lecture Notes in Computer Science, pp.106-117, Springer-Verlag, 2000.

[GMA+04] M. Gonda, K. Matsuo, K. Aoki, J. Chao and S. Tsujii. Improvements Of Addition Algorithm On Genus 3 Hyperelliptic Curves And Their Implementations. In Proc. of SCIS 2004, Japan, 2004

[KJJ99] P. Kocher, J. Jaffe and B. Jun. Differential power analysis. In *Advances in Cryptology - Crypto'99*, LNCS 1666, pages 388 - 397, Berlin, 1999. Springer-Verlag.

[Kob87] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48:203-209, 1987.

[Kob88] N. Koblitz. A Family of Jacobians Suitable for Discrete Log Cryptosystems. In Shafi Goldwasser, editor, *Advances in Cryptology - Crypto '88*, LNCS 403, pages 94-99, Berlin, 1988. Springer-Verlag.

[Kob89] N. Koblitz. Hyperelliptic Cryptosystems. In Ernest F.Brickell, editor, *Journal of Cryptology*, pp.139-150, 1989.

[Koc96] P. Kocher. Timing attack on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Advance in cryptology - Crypto'96*, volume LNCS 1109, pages 104 - 113, Springer-Verlag, 1996.

[KWC+04] H. Kim, T. Wollinger, Y. Choi, K. Chung and C. Paar. Hyperelliptic Curve Coprocessors on a FPGA, In *Workshop on Information Security Applications - WISA*, volume LNCS, Springer-Verlag, 2004.

[KMG+02] J. Kuroki, M. Gonda, K. Matsuo, J. Chao and S.Tsujii.: Fast Genus Three Hyperelliptic Curve Cryptosystems. In *Proc. of SCIS 2002*, IEICE Japan, pp.503-507, 2002.

[Lan03] T. Lange. Formulae for Arithmetic on Genus 2 Hyperelliptic Curves. *Jounal of AAECC*, Septemper 2003.

[Lan04] T. Lange. Koblitz Curve Cryptosystems. *Finite Fields and Their Applications*, 2004. to appear.

[Loc94] P. Lockhart. On the discriminant of a hyperelliptic curve. In *Tran. Amer. Math. Soc.* 342, 2, 729-752, 1994.

[LS04] T. Lange and M. Stevens. Efficient Doubling on Genus Two Curves over Binary Fields. In H.Handschuh and A.Hasan, editors, *Eleventh Annual Workshop on Selected Areas in Cryptography - SAC 2004*, volume LNCS 3357, pp. 170-181, Springer-Verlag, 2005.

[MCT01] K. Matsuo, J. Chao, and S. Tsujii. Fast Genus Two Hyperelliptic Curve Cryptosystems. In *ISEC2001-31*, IEICE, 2001.

[MDM+02] Y. Miyamoto, H. Doi, K. Matsuo, J. Chao, and S. Tsuji. A Fast Addition Algorithm of Genus Two Hyperelliptic Curve. In *SCIS*, IEICE Japan, pp. 497 - 502, 2002. in Japanese.

[Mil86] V. Miller. Uses of Elliptic Curves in Cryptography. In H. C. Williams, editor, *Advances in Cryptology - CRYPTO '85*, LNCS 218, pages 417-426, Berlin, Germany, 1986. Springer-Verlag.

[MWZ96] A.Menezes, Y.Wu and R.Zuccherato. An Elementary Introduction to Hyperelliptic Curve. *Technical Report CORR 96-19*, University of Waterloo, 1996, Canada. Available at http://www.cacr.math.uwaterloo.ca

[Pel02] J.Pelzl. *Hyperelliptic Cryptosystems on Embedded Microprocessor*. Master's thesis, Department of Electronical Engineering and Information Sciences, Ruhr-Universitaet Bochum, Bochum, Germany, September 2002.

[PWGP03] J. pelzl, T. Wollinger, J. Guajardo and C. Paar. Hyperelliptic Curve Cryptosystems: Closing The Performance Gap To elliptic Curve (Update), *Cryptology ePrint Archieve*, Report 2003/026, http://eprint.iacr.org/, 2003

[PWP03] J. Pelzl, T. Wollinger, and C. Paar. Low Cost Security: Explicit Formulae for Genus-4 Hyperelliptic Curves. In M. Matsui and R. Zuccherato, editors, *Tenth Annual Workshop on Selected Areas in Cryptography - SAC 2003*, volume LNCS 3006, pp. 1-16, Springer-Verlag, 2003.

[RS02] K.Rubin and A.Silverberg. Supersingular abelian varieties in cryptology. In *Advance in cryptology - Crypto'2002*, volume 2442 of Lecture Notes in Computer Science, pages 336-353, Springer-Verlag, 2002.

[Tak02] M. Takahashi. Improving Harley Algorithms for Jacobians of Genus 2 Hyperelliptic Curves. In SCIS, IEICE Japan, 2002. in Japanese.

[Thé03] N.Thériault. Index calculus attack for hyperelliptic curves of small genus. *Advances in Cryptology - ASIACRYPT'03*, G.Goos, J.Hartmanis, and J.van Leeuwen, Eds. Berlin: Springer Verlag, 2003, pp.79 - 92, LNCS 2894.

[Wol04] T. Wollinger. *Software and Hardware Implementation of Hyperelliptic Curve Cryptosystems*. Europäischer Universitätsverlag, 3-86515-025-X, 2004.

[WPW+03] T. Wollinger, J. Pelzl, V. Wittelsberger, C. Paar, G. Saldamli, and Ç. K. Koç. Elliptic & hyperelliptic curves on embedded μp. *ACM Transactions in Embedded Computing Systems (TECS)*, 2003. Special Issue on Embedded Systems and Security.