A Complete Divisor Class Halving Algorithm for Hyperelliptic Curve Cryptosystems of Genus Two

Izuru Kitamura[1], Masanobu Katagi[1], and Tsuyoshi Takagi *[2]

[1] Sony Corporation, 6-7-35 Kitashinagawa Shinagawa-ku, Tokyo, 141-0001 Japan
{Izuru.Kitamura, Masanobu.Katagi}@jp.sony.com
[2] Future University - Hakodate, 116-2 Kamedanakano-cho Hakodate, 041-8655, Japan
takagi@fun.ac.jp

Abstract. We deal with a divisor class halving algorithm on hyperelliptic curve cryptosystems (HECC), which can be used for scalar multiplication, instead of a doubling algorithm. It is not obvious how to construct a halving algorithm, due to the complicated addition formula of hyperelliptic curves. In this paper, we propose the first halving algorithm used for HECC of genus 2, which is as efficient as the previously known doubling algorithm. From the explicit formula of the doubling algorithm, we can generate some equations whose common solutions contain the halved value. From these equations we derive four specific equations and show an algorithm that selects the proper halved value using two trace computations in the worst case. If a base point is fixed, we can reduce these extra field operations by using a pre-computed table which shows the correct halving divisor class — the improvement over the previously known fastest doubling algorithm is up to about 10%. This halving algorithm is applicable to DSA and DH scheme based on HECC. Finally, we present the divisor class halving algorithms for not only the most frequent case but also other exceptional cases.

Keywords. hyperelliptic curve cryptosystems, scalar multiplication, divisor class halving, efficient computation

1 Introduction

We know from recent research that hyperelliptic curve cryptosystems (HECC) of small genus are competing with elliptic curve cryptosystems (ECC) [Ava04,Lan02a-c, PWG$^+$03]. With an eye to further improvement of HECC we utilize its abundant algebraic structure to make HECC faster in scalar multiplication than ECC. Lange and Duquesne independently showed that Montgomery scalar multiplication is applicable to HECC [Lan04a,Duq04]. We expect other fast algorithms used for ECC can also be efficiently implemented in HECC.

* This work was carried out when the author was in Technische Universität Darmstadt, Fachbereich Informatik, Hochschulstr.10, D-64289 Darmstadt, Germany

A point halving algorithm is one of the effective algorithms on ECC and the algorithm tries to find a point P such that $2P = Q$ for a given point Q. Knudsen and Schroeppel independently proposed a point halving algorithm for ECC over binary fields $\mathbb{F}_{2^n}$ [Knu99,Sch00]. Their algorithm is faster than a doubling algorithm. Moreover, there has been growing consideration of the point halving algorithm, showing, for instance, a fast implementation [FHL$^+$03], an application for Koblitz curve [ACF04], and an improvement of curves with cofactor 4 [KR04]. The explicit doubling formula of HECC (denote by HECDBL) is more complicated than that of ECC. It is not obvious how the algorithm of Knudsen and Schroeppel can extend to HECC.

In this paper, we propose a divisor class halving algorithm applied to HECC with genus 2 over binary fields. Let $D = (U, V)$ be a reduced divisor, where $U = x^2 + u_1x + u_0$ and $V = v_1x + v_0$. The doubled divisor class $2D$ can be represented as polynomials over $\mathbb{F}_{2^n}$ with coefficients u_1, u_0, v_1, v_0 s and curve parameters $y^2 + h(x)y = f(x)$. We report two crucial quadratic equations which compute some candidates of the halved values. These equations are derived from the property: an equation of degree 6 appeared in the doubling algorithm can be divided by $x^4 + u_1^2x^2 + u_0^2$. We also show a criterion and an algorithm selecting the correct divisor class from two candidates. The correct divisor class can be efficiently found if the polynomial $h(x)$ is irreducible. In order to select the correct halved value, we perform some test calculations, and notice that the number of operations can be reduced if the correct halving value is first found. We developed a divisor class halving algorithm used for not only the most frequent case but also exceptional cases, e.g. the weight of input divisor class is 1. The proposed algorithm can be optimized with careful considerations of the basic operations.

This paper is organized as follows: in Section 2 we review the algorithms of a hyperelliptic curve. In Section 3 we present our proposed divisor class halving algorithm for HECC, and compare it with existing doubling formulae. In Section 4 a complete divisor class halving algorithm is shown. In Section 5 we consider a halving algorithm for a special curve, $\deg h = 1$. In Section 6 is our conclusion.

2 Hyperelliptic Curve

We review the hyperelliptic curve used in this work.

Let $\mathbb{F}_{2^n}$ be a binary finite field with 2^n elements. A hyperelliptic curve C of genus g over $\mathbb{F}_{2^n}$ with one point at infinity is defined by $C : y^2 + h(x)y = f(x)$, where $f(x) \in \mathbb{F}_{2^n}[x]$ is a monic polynomial of degree $2g+1$ and $h(x) \in \mathbb{F}_{2^n}[x]$ is a polynomial of degree at most g, and curve C has no singular point. Let $P_i = (x_i, y_i) \in \overline{\mathbb{F}_{2^n}} \times \overline{\mathbb{F}_{2^n}}$ be a point on curve C and P_∞ be a point at infinity, where $\overline{\mathbb{F}_{2^n}}$ is the algebraic closure of $\mathbb{F}_{2^n}$. The inverse of $P_i = (x_i, y_i)$ is the point $-P_i = (x_i, y_i + h(x_i))$. P is called a ramification point if $P = -P$ holds. A divisor is a formal sum of points: $D = \sum m_iP_i, m_i \in \mathbb{Z}$. A semi-reduced divisor is given by $D = \sum m_iP_i - (\sum m_i)P_\infty$, where $m_i \geq 0$ and $P_i \neq -P_j$ for $i \neq j$, and semi-reduced divisor D is called reduced if $\sum m_i \leq g$ holds. The weight of a reduced divisor D is defined as $\sum m_i$, and we denote it by $w(D)$. Jacobian $\mathbf{J}$ is isomorphic to the divisor class group which forms an additive group. Each divisor class can be represented uniquely by a reduced divisor and so we can identify the set of points on the Jacobian with the set of reduced divisors and assume this identification from now on. The reduced and the semi-reduced divisors are expressed by a pair of polynomials (u, v), which satisfies the following conditions [Mum84]:

$$u(x) = \prod (x + x_i)^{m_i}, v(x_i) = y_i, \deg v < \deg u, v^2 + hv + f \equiv 0 \bmod u.$$

A divisor class is defined over $\mathbb{F}_{2^n}$ if the representing polynomials u, v are defined over this field and the set of $\mathbb{F}_{2^n}$-rational points of the Jacobian is denoted by $\mathbf{J}(\mathbb{F}_{2^n})$. Note that even if $u, v \in \mathbb{F}_{2^n}[x]$, the coordinates x_i and y_i may be in extension field of $\mathbb{F}_{2^n}$. The degree of u equals the weight of the reduced divisor and we represent the zero element by $O = (1, 0)$. To compute the additive group law of $\mathbf{J}(\mathbb{F}_{2^n})$, Cantor gave an addition algorithm as follows:

Algorithm 1 *Cantor Algorithm*

```
Input:  $D_1 = (U_1, V_1)$ and $D_2 = (U_2, V_2)$
Output: $D_3 = (U_3, V_3) = D_1 + D_2$
  $U_i = u_{i2}x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0}$, where $i = 1, 2$ and $u_{i2} \in \mathbb{F}_2$
1. $d \leftarrow \gcd(U_1, U_2, V_1 + V_2 + h) = s_1U_1 + s_2U_2 + s_3(V_1 + V_2 + h)$
2. $U \leftarrow U_1U_2/d^2,\ V \leftarrow (s_1U_1V_2 + s_2U_2V_1 + s_3(V_1V_2 + f))/d \bmod U$
3. while $\deg U > g$
     $U \leftarrow (f + v + V^2)/U,\ V \leftarrow h + V \bmod U$
4. $U_3 \leftarrow \mathrm{MakeMonic}(U), V_3 \leftarrow V$
5. return $(U_3, V_3)$
```

Step 1 and Step 2 are called the composition part and Step 3 is called the reduction part. The composition part computes the semi-reduced divisor $D = (U, V)$ that is equivalent to D_3. The reduction part computes the reduced divisor $D_3 = (U_3, V_3)$.

The Cantor Algorithm is applicable to a hyperelliptic curve of any genus. However, this algorithm is relatively slow due to its generality. Harley then proposed an efficient addition and doubling algorithm for a hyperelliptic curve of genus 2 over $\mathbb{F}_p$ [GH00,Har00a,Har00b]. This algorithm achieved speeding up by detailed classification into the most frequent case and some exceptional cases. This classification allows us to avoid extra field operations. Sugizaki et al. expanded the Harley algorithm to HECC over $\mathbb{F}_{2^n}$ [SMC$^+$02], and around the same time Lange expanded the Harley algorithm to HECC over general finite field [Lan02a]. The most frequent case of doubling algorithm HECDBL is defined as follows:

HECDBL: $w(D_1) = w(D_2) = 2$, $D_2 = 2D_1$, and D_1 has no ramification points.

Algorithm 2 HECDBL

```
Input:  $D_1 = (U_1, V_1)$
Output: $D_2 = (U_2, V_2) = 2D_1$
  $U_i = x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0},$ where $i = 1, 2$
1. $U_1' \leftarrow U_1^2$
2. $S \leftarrow (f + hV_1 + V_1^2)/U_1,\ S \leftarrow Sh^{-1} \bmod U_1$
3. $V_1' \leftarrow SU_1 + V_1$
4. $U_2' \leftarrow (f + hV_1' + V_1'^2)/U_1'$
5. $U_2 \leftarrow$ MakeMonic $(U_2')$
6. $V_2 \leftarrow V_1' + h \bmod U_2$
7. return $(U_2, V_2)$
```

In HECDBL, from Step 1 to Step 3 is the composition part and from Step 4 to Step 6 is the reduction part. The composition part computes the semi-reduced divisor $D = (U_1', V_1')$ equivalent to D_2. In Step 2 and Step 3, we compute V_1' such that $f + hV_1' + V_1'^2 \equiv 0 \bmod U_1'$, which can be obtained by $V_1' \equiv V_1 \bmod U_1$ via Newton iteration. The reduction part computes the reduced divisor $D_2 = (U_2, V_2) = 2D_1$. From Algorithm 2, it is clear that the number of field operations depends on the curve parameters. To reduce the number of field operations, in previous works, a transformed curve $y^2 + (x^2 + h_1'x + h_0')y = x^5 + f_3'x^3 + \cdots + f_0'$, via isomorphic transformations: $y \rightarrow h_2^5 y$ and $x \rightarrow h_2^2 x + f_4$, are used. We call this transformed curve a *general curve*. In this paper, our aim is to present the divisor class halving algorithm for the general curve and to prove the correctness of this algorithm. Additionally, we consider a simple polynomial $h(x) = h_1x + h_0$ and we call this curve a *special curve*.

In a cryptographic application, we are only interested in a curve whose order of $\mathbf{J}(\mathbb{F}_{2^n})$ is $2 \times r$, i.e. whose cofactor is two, where r is a large prime number. Note that the cofactor is always divisible by 2 (See Appendix A). Moreover, as inputs and outputs for the halving and doubling algorithm we use the divisor classes whose order is r.

3 Proposed Halving Algorithm for General Curve

In this section we propose a divisor class halving algorithm (HECHLV) on hyperelliptic curve cryptosystems of genus two. We derive HECHLV by inverse computing of HECDBL. For HECHLV, the significance problem is to find the *missing polynomial* k such that $V_1' + h = kU_2 + V_2$ in Algorithm 2. First, we compute k by a reverse operation of the reduction part, then the semi-reduced divisor via k, at last $D_1 = \frac{1}{2}D_2$ by a reverse operation of the composition part.

3.1 Main Idea

We follow the opposite path to HECDBL. From Step 6 of HECDBL, there is a unique polynomial $k = k_1x + k_0$ such that $V_1' + h = (k_1x + k_0)U_2 + V_2$. Substituting V_1' to equation $(f + hV_1' + V_1'^2)$ appeared in Step 4, the following relationship yields:

$$U_2'U_1' = f + h(kU_2 + V_2) + k^2U_2^2 + V_2^2. \tag{1}$$

Because the doubled divisor class (U_2, V_2) is known, we can obtain the relationship between k and U_1'. Note that $U_2' = k_1^2U_2$ from the highest term of equation (1). Recall that $U_1' = U_1^2$ from Step 1, namely, we know

$$U_1' = x^4 + u_{11}^2x^2 + u_{10}^2. \tag{2}$$

In other words, the coefficients of degree 3 and 1 are zero. From this observation, there are polynomials whose solutions includes k_0 and k_1. In our algorithm we try to find k_0 and k_1 by solving the polynomials. Once k_0 and k_1 are calculated, we can easily compute the halved divisor class $D_1 = (U_1, V_1)$ from equation (1). We describe the sketch of the proposed algorithm in the following.

Algorithm 3 *Sketch* HECHLV

```
Input:   $D_2 = (U_2, V_2)$
Output:  $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
  $U_i = x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0},$ where $i = 1, 2$
1. determine $k = k_1x + k_0$ by the reverse operation of the reduction part
   1.1 $V_1' \leftarrow V_2 + h + kU_2,\ k = k_1x + k_0$
   1.2 $U_1' \leftarrow (f + hV_1' + V_1'^2)/(k_1^2U_2)$
   1.3 derive $k_0,\ k_1$ from two equations $\mathrm{coeff}(U_1', 3) = 0$ and $\mathrm{coeff}(U_1', 1) = 0$
2. compute $U_1' = x^4 + u_{11}^2x^2 + u_{10}^2$ in the semi-reduced divisor by using $k_0, k_1$
   2.1 compute $u_{11}^2$ by substituting $k_0, k_1$ in $\mathrm{coeff}(U_1', 2)$
   2.2 compute $u_{10}^2$ by substituting $k_0, k_1$ in $\mathrm{coeff}(U_1', 0)$
3. compute $D_1 = (U_1, V_1) = \frac{1}{2}D_2$ by the reverse operation of the composition part
   3.1 $U_1 \leftarrow \sqrt{U_1'} = x^2 + u_{11}x + u_{10}$
   3.2 $V_1 \leftarrow V_2 + h + kU_2 \bmod U_1$
4. return $(U_1, V_1)$
```

In the following, we explain Algorithm 3 in detail. The coeff(U, i) is the coefficient of x^i in polynomial U. In Step 1.2, we compute polynomial U_1' in equation (1):

$$\begin{aligned}
\mathrm{coeff}(U_1', 3) &= (k_1h_2 + k_1^2u_{21} + 1)/k_1^2 \\
\mathrm{coeff}(U_1', 2) &= (k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2)/k_1^2 \\
\mathrm{coeff}(U_1', 1) &= (k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1)/k_1^2 \\
\mathrm{coeff}(U_1', 0) &= (k_0h_0 + k_0^2u_{20} + c_0)/k_1^2,
\end{aligned}$$

where

$$\begin{aligned}
c_2 &= f_4 + u_{21}, \quad c_1 = f_3 + h_2v_{21} + u_{20} + c_2u_{21}, \\
c_0 &= f_2 + h_2v_{20} + h_1v_{21} + v_{21}^2 + c_2u_{20} + c_1u_{21}.
\end{aligned}$$

Equation (2) yields the explicit relationship related to variables k_0, k_1, u_{11}, and u_{10}:

$$k_1h_2 + k_1^2u_{21} + 1 = 0 \tag{3}$$

$$k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0 \tag{4}$$

$$u_{11} = \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2}/k_1 \tag{5}$$

$$u_{10} = \sqrt{k_0h_0 + k_0^2u_{20} + c_0}/k_1 \tag{6}$$

In the algorithm we used the following lemma in order to uniquely find k_0, k_1. The proof of this lemma is in Appendix B.

Lemma 1. *Let $h(x)$ be an irreducible polynomial of degree 2. There is only one value k_1 which satisfies both equations (3) and (4). Equation (4) has a solution only for the correct k_1. There is only one value k_0 which yields the halved divisor class D_1 in algorithm 3. Equation $xh_2 + x^2u_{11} + 1 = 0$ has a solution only for the correct k_0.*

After calculating k_0, k_1, we can easily compute u_{11}, u_{10}, v_{11}, and v_{10} via equations (5), (6), and $V_1 \leftarrow V_2 + h + (k_1x + k_0)U_2 \bmod U_1$.

3.2 Proposed Algorithm

We make the assumption that the polynomial h has degree two and is irreducible. We present the proposed algorithm in Algorithm 4.

The proposed algorithm requires to solve quadratic equations. It is well known that equation $ax^2 + bx + c = 0$ has roots if and only if $\mathrm{Tr}(ac/b^2) = 0$. Let one root of $ax^2 + bx + c = 0$ be x_0, then the other root be $x_0 + b/a$. If this equation has roots, i.e. $\mathrm{Tr}(ac/b^2) = 0$, then we can solve this equation by using half trace, namely $x_0 = H(ac/b^2), x_0' = x_0 + b/a$. This equation has no root if $\mathrm{Tr}(ac/b^2) = 1$.

We explain the proposed algorithm as follows. The correctness of this algorithm is shown in Lemma 1. In Step 1, we solve two solutions k_1 and k_1' of equation (3). In Step 2, the correct k_1 is selected by checking the trace of equation (4). Then we obtain two solutions k_0 and k_0' of equation (4). In Step 3, the correct k_0 is selected by checking trace of $xh_2 + x^2u_{11} + 1 = 0$. In Steps 4 and 5 we compute the halved divisor class.

Algorithm 4 HECHLV

```
Input:  $D_2 = (U_2, V_2)$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
  $U_i = x^2 + u_{i1}x + u_{i0},\ V_i = v_{i1}x + v_{i0},$ where $i = 1, 2,\ h_2 \neq 0$
step  procedure
1.    Solve $k_1h_2 + k_1^2u_{21} + 1 = 0$
      $\alpha \leftarrow h_2/u_{21}, \gamma \leftarrow u_{21}/h_2^2,\ k_1 \leftarrow H(\gamma)\alpha,\ k_1' \leftarrow k_1 + \alpha$
2.    **Select correct $k_1$ by solving** $k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0$
      $c_2 \leftarrow f_4 + u_{21},\ c_1 \leftarrow f_3 + h_2v_{21} + u_{20} + c_2u_{21},$
      $c_0 \leftarrow f_2 + h_2v_{20} + h_1v_{21} + v_{21}^2 + c_2u_{20} + c_1u_{21}, \alpha \leftarrow h_1/u_{21},$
      $w \leftarrow u_{21}/h_1^2,\ \gamma \leftarrow (c_1 + k_1h_0)w$
      if $\mathrm{Tr}(\gamma) = 1$ then $k_1 \leftarrow k_1',\ \gamma \leftarrow (c_1 + k_1h_0)w$
      $k_0 \leftarrow H(\gamma)\alpha,\ k_0' \leftarrow k_0 + \alpha$
3.    **Select correct $k_0$ by checking trace of** $xh_2 + x^2u_{11} + 1 = 0$
      $u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2}/k_1,\ \gamma \leftarrow u_{11}/h_2^2$
      if $\mathrm{Tr}(\gamma) = 1$ then $k_0 \leftarrow k_0',\ u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2}/k_1$
4.    **Compute** $U_1$
      $u_{10} \leftarrow \sqrt{k_0h_0 + k_0^2u_{20} + c_0}/k_1$
5.    **Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$
      $w \leftarrow h_2 + k_1u_{21} + k_0 + k_1u_{11}$
      $v_{11} \leftarrow v_{21} + h_1 + k_1u_{20} + k_0u_{21} + u_{10}k_1 + u_{11}w,\ v_{10} \leftarrow v_{20} + h_0 + k_0u_{20} + u_{10}w$
6.    $D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}),$ return $D_1$
```

3.3 Complexity and Improvement

In order to estimate the complexity of HECHLV shown in Algorithm 4, we consider four cases with respect to the selection of k_1 and k_0. When we get incorrect k_1 and k_0 (k_1' and k_0' are correct) in Steps 1 and 2, respectively, we have to replace $k_1 \leftarrow k_1'$, $k_0 \leftarrow k_0'$ and compute γ, u_{11} again in Steps 2 and 3, respectively. In the worst case this requires $4M + 1SR$ as additional field operations compared to the best case, and we have another two cases: one is k_0 and k_1' are correct and the other is k_0' and k_1 are correct. Note that a multiplication by M for short and other operations are expressed as follows: a squaring (S), an inversion (I), a square root (SR), a half trace (H), and a trace (T). Our experimental observations found that these four cases occur with almost the same probability. Therefore, we employ the average of these four cases as the average case.

Now we consider how to optimize the field operations in Algorithm 4. We will discuss the optimization under the two topics: choices of the curve parameter and scalar multiplication using a fixed base point.

Choices of the curve parameter. The complexity of HECHLV depends on the coefficients of the curve. If the coefficients are small, one, or zero, we reduce some field operations. Firstly, we reduce some inversion operations to one. If $1/h_1^2$ and $1/h_2^2$ are allowed as inputs, we reduce two inversion operations and we compute $1/k_1 = h_2 + k_1u_{21}$ from equation (3), then Algorithm 4 requires only one inversion operation $1/u_{21}$. Secondly, we use the general curve. When $f_4 = 0$ we reduce $3M$ to $1M + 1S$ by $c_2u_{21} = u_{21}^2$ and $c_2u_{20} + c_1u_{21} = u_{21}(u_{20} + c_1)$. When $h_2 = 1$ two multiplications by h_2 and two multiplications by $1/h_2^2$ are omitted. Thirdly, we use the general curve when $h_1 = 1$. In this case, we change $1M$ to $1S$ by $v_{21}(h_1 + v_{21}) = v_{21} + v_{21}^2$, where $1S$ is faster than $1M$, and two multiplications by h_1 and one multiplication by $1/h_1^2$ are reduced. Finally, we use the general curve when $h_1 = h_0 = 1$ then we skip one multiplication k_1h_0. We summarize these improvements in Algorithm 5.

Algorithm 5 HECHLV $(h_2 = 1, f_4 = 0)$

Input: $D_2 = (U_2, V_2), 1/h_1^2$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
$U_i = x^2 + u_{i1}x + u_{i0},\ V_i = v_{i1}x + v_{i0},$ where $i = 1, 2$

step	*procedure*	*cost*
1.	**Solve** $k_1 + k_1^2u_{21} + 1 = 0$	$1M + 1I + 1H$
	$\alpha \leftarrow 1/u_{21},\ k_1 \leftarrow \mathrm{H}(u_{21})\alpha,\ k_1' \leftarrow k_1 + \alpha$	
2.	**Select correct** k_1 **by solving** $k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0$	$9M + 1S + 1H + 1T$
	$c_1 \leftarrow f_3 + v_{21} + u_{20} + u_{21}^2$	
	$c_0 \leftarrow f_2 + v_{20} + v_{21}(h_1 + v_{21}) + u_{21}(u_{20} + c_1)$	$(h_1 = 1 : v_{21}(h_1 + v_{21}) = v_{21} + v_{21}^2)$
	$w_0 \leftarrow u_{21}/h_1^2,\ \alpha \leftarrow h_1\alpha, \gamma \leftarrow (c_1 + k_1h_0)w_0$	
	if $\mathrm{Tr}(\gamma) = 1$ then $k_1 \leftarrow k_1',\ \gamma \leftarrow (c_1 + k_1h_0)w_0$	$(h_1 = 1 : \gamma \leftarrow \gamma + h_0)$
	$k_0 \leftarrow H(\gamma)\alpha,\ k_0' \leftarrow k_0 + \alpha$	
3.	**Select correct** k_0 **by solving** $x + x^2u_{11} + 1 = 0$	$5M + 1S + 2SR + 1T$
	$w_0 \leftarrow k_1^2,\ w_1 \leftarrow w_0u_{20} + k_1h_1 + u_{21}$	
	$w_2 \leftarrow k_0 + \sqrt{w_1 + k_0},\ w_4 \leftarrow k_1u_{21} + 1,\ u_{11} \leftarrow w_2w_4$	
	if $\mathrm{Tr}(u_{11}) = 1$ then	
	$k_0 \leftarrow k_0',\ w_2 \leftarrow k_0 + \sqrt{w_1 + k_0},\ u_{11} \leftarrow w_2w_4$	
4.	**Compute** U_1	$4M + 1SR$
	$w_1 \leftarrow k_0u_{20},\ w_5 \leftarrow w_4 + 1,\ w_6 \leftarrow (k_0 + k_1)(u_{20} + u_{21})$	
	$u_{10} \leftarrow w_4\sqrt{k_0(w_1 + h_0)} + c_0$	
5.	**Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$	$2M$
	$w_4 \leftarrow w_5 + k_0 + 1,\ w_5 \leftarrow w_1 + w_5 + w_6 + v_{21} + h_1$	
	$w_6 \leftarrow w_1 + v_{20} + h_0,\ w_7 \leftarrow w_2 + w_4$	
	$w_1 \leftarrow w_7u_{10},\ w_3 \leftarrow (k_1 + w_7)(u_{10} + u_{11})$	
	$v_{11} \leftarrow w_1 + w_2 + w_3 + w_5,\ v_{10} \leftarrow w_1 + w_6$	
6.	$D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10})$, return D_1	
total	(k_1, k_0) *is correct*	$18M + 2S + 1I + 2SR + 2H + 2T$
	(k_1, k_0') *is correct*	$19M + 2S + 1I + 3SR + 2H + 2T$
	(k_1', k_0) *is correct*	$20M + 2S + 1I + 2SR + 2H + 2T$
	(k_1', k_0') *is correct*	$21M + 2S + 1I + 3SR + 2H + 2T$
	$h_1 = 1$	
	(k_1, k_0) *or* (k_1', k_0) *is correct*	$14M + 3S + 1I + 2SR + 2H + 2T$
	(k_1, k_0') *or* (k_1', k_0') *is correct*	$15M + 3S + 1I + 3SR + 2H + 2T$
	$h_1 = h_0 = 1$	
	(k_1, k_0) *or* (k_1', k_0) *is correct*	$13M + 3S + 1I + 2SR + 2H + 2T$
	(k_1, k_0') *or* (k_1', k_0') *is correct*	$14M + 3S + 1I + 3SR + 2H + 2T$

Scalar multiplication with a fixed base point. We describe the scalar multiplication using divisor class halvings. Knudsen and Schroeppel proposed the ECC scalar multiplication algorithm, halve-and-add binary method, which replaces point doublings in double-and-add binary methods with point halvings. Similarly, the halve-and-add binary method can be applied to HECC via the divisor class halving proposed in Algorithm 5. In order to compute the halve-and-add binary method, we have to convert a scalar value from binary representation to half representation. Let r be the order of the underlying base point and $m = \lfloor \log_2 r \rfloor$. For a given integer d we can represent $d \equiv \sum_{i=0}^{m} \hat{d}_i 2^{i-m} \pmod r$ and $\sum_{i=0}^{m} \frac{d_i}{2^i} \leftarrow \sum_{i=0}^{m} \hat{d}_i 2^{i-m}$, where $d_i, \hat{d}_i \in \{0,1\}$. This representation d_i is used for the halve-and-add binary method.

In the case of scalar multiplication with a fixed base point D, we improve a computation method of $\frac{1}{2^i}D$ via pre-computed tables. When we know the correct k_1 and k_0 in advance, we reduce three multiplications, two traces, and one square root in Algorithm 5. We can take the pre-computed tables $\mathbf{t_1} = (t_{1,m}t_{1,m-1}\cdots t_{1,0})_2$ and $\mathbf{t_0} = (t_{0,m}t_{0,m-1}\cdots t_{0,0})_2$ which show whether $k_1(k_0)$ or $k_1'(k_0')$ is the correct value in each halving — $t_{1,i} = 0(t_{0,i} = 0)$ means $k_1(k_0)$ is correct and $t_{1,i} = 1(t_{0,i} = 1)$ means $k_1'(k_0')$ is correct, since whether $k_1(k_0)$ is correct or not depends on D. This improvement can be applied to a right-to-left binary method by adding $\frac{1}{2^i}D$. The divisor class halve-and-add binary method is as follows:

Algorithm 6 *Halve-and-add binary (right-to-left) method.*

Input: $d \in \mathbb{Z}, D \in \mathbf{J}(\mathbb{F}_{2^n}),\ r$: order of $D,\ m = \lfloor \log_2 r \rfloor, \mathbf{t_1}, \mathbf{t_0}$
Output: dD: scalar multiplication with a fixed base point

```
step  procedure
1.    $\sum_{i=0}^{m} \hat{d}_i 2^i \leftarrow 2^m d \pmod r, \quad \hat{d}_i \in \{0,1\}$
2.    $\sum_{i=0}^{m} \frac{d_i}{2^i} \leftarrow \sum_{i=0}^{m} \hat{d}_i 2^{i-m}, \quad d_i \in \{0,1\}$
3.    $Q \leftarrow O, R \leftarrow D$
4.    for $i$ from 0 to $m$ do:
        if $d_i = 1$ then $Q \leftarrow Q + R$.
5.      $R \leftarrow \mathsf{HECHLV}(R, t_{1,i}, t_{0,i})$.
6.    return $Q$.
```

These tables require only the same bit length as D since D needs $4n$ bits while m has length $2n$ and we need two bits to encode the right choices of k_1 and k_0. We adopt this table-lookup method to the general curve and show this in Algorithm 10, which then requires only $18M + 2S + 1I + 2SR + 2H$.

3.4 Comparison of doubling and halving

We compare field operations cost of doubling algorithms to halving algorithms. Table 1 provides a comparison of HECDBL and the above halving algorithms in the average case.

By using the normal basis, we can neglect the computation time of a squaring, a square root, a half trace, and a trace compared to that of a field multiplication or an inversion [Knu99]. Menezes [Men93] showed that an inversion operation requires $\lfloor \log_2(n-1) \rfloor + \#(n-1) - 1$ multiplications, where $\#(n-1)$ is the number of 1's in the binary representation of $n-1$. By neglecting these operations, for the general curve, HECHLV and HECDBL require $19.5M_N + 1I$ and $21M_N + 1I$, respectively, where M_N

Table 1. Comparison of Halving and Doubling

Scheme	HECHLV	HECDBL [LS04]
$h_2 = 1, f_4 = 0$	$19.5M + 2S + 1I + 2.5SR + 2H + 2T$	$21M + 5S + 1I$
random base point	$(27.5M_N, 29.95M_P)$	$(29M_N, 29.5M_P)$
$h_2 = 1, f_4 = 0$	$18M + 2S + 1I + 2SR + 2H$	—
fixed base point	$(26M_N, 28.2M_P)$	—
$h_2 = h_1 = 1, f_4 = 0$	$14.5M + 3S + 1I + 2.5SR + 2H + 2T$	$18M + 7S + 1I$
random base point	$(22.5M_N, 25.05M_P)$	$(26M_N, 26.7M_P)$
$h_2 = h_1 = 1, f_4 = 0$	$14M + 3S + 1I + 2SR + 2H$	—
fixed base point	$(22M_N, 24.3M_P)$	—
$h_2 = h_1 = h_0 = 1, f_4 = 0$	$13.5M + 3S + 1I + 2.5SR + 2H + 2T$	$15M + 7S + 1I$
random base point	$(21.5M_N, 24.05M_P)$	$(23M_N, 23.7M_P)$
$h_1 = h_1 = h_0 = 1, f_4 = 0$	$13M + 3S + 1I + 2SR + 2H$	—
fixed base point	$(21M_N, 23.3M_P)$	—

is a multiplication over the normal basis. When we assume $1I = 8M_N$ HECHLV and HECDBL require $27.5M_N$ and $29M_N$, respectively.

On the other hand, by using the polynomial basis, we cannot ignore the computation time of a squaring, a square root, and a half trace. Assuming that $1S = 0.1M_P$, $1SR = 0.5M_P$, $1H = 0.5M_P$, and $1I = 8M_P$, where M_P is a multiplication over the polynomial basis. For the general curve, HECHLV and HECDBL require $29.95M_P$ and $29.5M_P$, respectively. By selecting the polynomial basis, however, we can compute these arithmetic faster than half the time of multiplication, and there is a possibility to reduce the cost of these operations.

Table 1 shows that when we use the normal basis HECHLV is faster than HECDBL for all the cases. On the contrary by using the polynomial basis, HECHLV is faster than HECDBL when $h_2 = h_1 = 1$ and $f_4 = 0$, especially the improvement by using a fixed base point over HECDBL is up to about 10%.

4 Complete Procedures for Divisor Class Halving Algorithm

In the previous sections, we proposed the halving algorithm, which corresponds to the most frequent case in the doubling algorithm. However, we also have to consider several exceptional procedures for giving complete procedures of the halving algorithm. These cases appear with very low probability, but we cannot ignore them. Therefore, we have to implement these procedures in order to perform the scalar multiplication correctly. In this paper we only deal with a divisor class whose order is r (not order $2 \times r$), and thus the divisor class does not include any ramification points. Therefore, we have to consider four inverse operations of HECDBL$^{2\to1}$, HECDBL$^{1\to2}$, HECDBL$^{2\to2}$, and HECDBL as follows:

HECDBL$^{2\to 1}$: $w(D_1) = 2$, $w(D_2) = 1$, $D_2 = 2D_1$.
HECDBL$^{1\to 2}$: $w(D_1) = 1$, $w(D_2) = 2$, $u_{21} = 0, D_2 = 2D_1$.
HECDBL$^{2\to 2}$: $w(D_1) = 2$, $w(D_2) = 2$, $u_{21} = 0, D_2 = 2D_1$.

Note that HECDBL$^{2\to 2}$ is computed via HECDBL. In the halving algorithm, however, we have to care HECDBL$^{2\to 2}$ because the inverse map of HECDBL$^{2\to 2}$ is indistinguishable from the inverse map of HECDBL$^{1\to 2}$. Therefore, the halving algorithms can be classified into four cases: HECHLV, HECHLV$^{1\to 2}$, HECHLV$^{2\to 2}$, and HECHLV$^{2\to 1}$. These cases are inverse maps of HECDBL, HECDBL$^{2\to 1}$, HECDBL$^{2\to 2}$, and HECDBL$^{1\to 2}$, respectively. The Complete HECHLV is as follows:

Algorithm 7 Complete HECHLV

Input: $D_2 = (U_2, V_2)$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
$U_i = u_{i2}x^2 + u_{i1}x + u_{i0}, V_i = v_{i1}x + v_{i0}, u_{i2} \in \mathbb{F}_2$, where $i = 1, 2$, $h_2 \neq 0$

step	*procedure*
1.	HECHLV$^{1\to 2}$**:** $w(D_2) = 1$, $w(D_1) = 2$
	if $\deg U_2 = 1$ then $D_1 \leftarrow$ HECHLV$^{1\to 2}(D_2)$, return D_1
2.	HECHLV$^{2\to 1}$**:** $w(D_2) = 2, w(D_1) = 1, u_{21} = 0$ or
	HECHLV$^{2\to 2}$**:** $w(D_2) = 2, w(D_1) = 2, u_{21} = 0$
	if $\deg U_2 = 2$ and $u_{21} = 0$ then $D_1 \leftarrow$ HECHLV$^{2\to 2}(D_2)$, return D_1
3.	HECHLV**:** $w(D_2) = w(D_1) = 2$, $u_{21} \neq 0$
	if $\deg U_2 = 2$ and $u_{21} \neq 0$ then $D_1 \leftarrow$ HECHLV(D_2), return D_1

In the following subsection we present explicit algorithms for each exceptional procedure.

4.1 HECHLV$^{1\to 2}$

A divisor class halving algorithm HECHLV$^{1\to 2}$ is similar to HECHLV. The main difference between HECHLV$^{1\to 2}$ and HECHLV is weight of input D_2. For example, in HECHLV$^{1\to 2}$, $f + hV_1' + {V_1'}^2$ is a monic polynomial with degree five because of $\deg(V_1') = 2$ and U_2 is a monic polynomial, so $U_1' \leftarrow (f + hV_1' + {V_1'}^2)/U_2$ not divided by k_1^2 like HECHLV.

We present the proposed algorithm in Algorithm 8. This algorithm is the analogy of HECHLV and the correctness of this algorithm is shown similarly to Lemma 1. Note that, in Step 3, the correct k_0 is selected by checking trace of $xh_2 + x^2u_{11} + 1 = 0$ not $xh_2 + x^2u_{11} + (f_4 + u_{10}) = 0$ because the weight of D_1 is always two and the method to select the correct k_0 is checking whether $\frac{1}{2}D_1 \in \mathbf{J}(\mathbb{F}_{2^n})$ or not, see Appendix B.

Algorithm 8 $\mathsf{HECHLV}^{1\to 2}$

Input: $D_2 = (U_2, V_2) = (x + u_{20}, v_{20})$
Output: $D_1 = (U_1, V_1) = (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}) = \frac{1}{2}D_2,\ h_2 \neq 0$

step	*procedure*
1.	**Solve** $k_1h_2 + k_1^2u_{21} + c_3 = 0$
	$c_3 \leftarrow f_4 + u_{20}, \alpha \leftarrow h_2, \gamma \leftarrow c_3/h_2^2,\ k_1 \leftarrow H(\gamma)\alpha,\ k_1' \leftarrow k_1 + \alpha$
2.	**Select correct** k_1 **by solving** $k_1h_0 + k_0h_1 + k_0^2 + c_1 = 0$
	$c_2 \leftarrow f_3 + c_3u_{20},\ c_1 \leftarrow f_2 + h_2v_{20} + c_2u_{20},\ c_0 \leftarrow f_1 + h_1v_{20} + c_1u_{20}$
	$\alpha \leftarrow h_1, \gamma \leftarrow (c_1 + k_1h_0)/\alpha^2$
	if $\mathrm{Tr}(\gamma) = 1$ then $k_1 \leftarrow k_1',\ \gamma \leftarrow (c_1 + k_1h_0)/\alpha^2$
	$k_0 \leftarrow H(\gamma)\alpha,\ k_0' \leftarrow k_0 + \alpha$
3.	**Select correct** k_0 **by checking trace of** $xh_2 + x^2u_{11} + 1 = 0$
	$u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + c_2},\ \gamma \leftarrow u_{11}/h_2^2$
	if $\mathrm{Tr}(\gamma) = 1$ then $k_0 \leftarrow k_0',\ u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + c_2}$
4.	**Compute** U_1
	$u_{10} \leftarrow \sqrt{k_0h_0 + k_0^2u_{20} + c_0}$
5.	**Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$
	$w \leftarrow h_2 + k_1,\ v_{11} \leftarrow h_1 + k_1u_{20} + k_0 + u_{11}w,\ v_{10} \leftarrow v_{20} + h_0 + k_0u_{20} + u_{10}w$
6.	$D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}),$ return D_1

4.2 $\mathsf{HECHLV}^{2\to 1}$

In this case, $D_1 = (x + u_{10}, v_{10})$ is computed by reverse operation of $\mathsf{HECDBL}^{1\to 2}$. $D_2 = (x^2 + u_{20}, v_{21}x + v_{20}) = 2D_1$ is computed as follows: $x^2 + u_{20} = (x + u_{10})^2$, $v_{21} = (u_{10}^4 + f_3u_{10}^2 + f_1 + h_1v_{10})/h(u_{10})$, and $v_{20} = v_{10} + v_{21}u_{10}$. Then we can easily express u_{10}, v_{10} by u_{20}, v_{21}, v_{20} and curve parameters by $u_{10} = \sqrt{u_{20}}$, $v_{10} = (v_{21}h(u_{10}) + u_{10}^4 + f_3u_{10}^2 + f_1)/h_1$.

4.3 $\mathsf{HECHLV}^{2\to 2}$

The case of $u_{21} = 0$, there are two candidate of $\frac{1}{2}D_2$: $D_1 = (x + \sqrt{u_{20}}, v_2(\sqrt{u_{20}}))$ and $D_1' = (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10})$. If D_1 is a correct divisor class, we use $\mathsf{HECHLV}^{2\to 1}$. On the other hand, if D_1' is a correct one, we use $\mathsf{HECHLV}^{2\to 2}$. We need to select a correct algorithm $\mathsf{HECHLV}^{2\to 1}$ or $\mathsf{HECHLV}^{2\to 2}$ as follows: First, we assume that D_1' is a correct divisor class, second compute u_{11}, then check the trace of $xh_2 + x^2u_{11} + 1 = 0$. If $\mathrm{Tr}(u_{11}/h_2^2) = 0$, D_1' is correct, then select the algorithm $\mathsf{HECHLV}^{2\to 2}$. If $\mathrm{Tr}(u_{11}/h_2^2) = 1$, D_1 is correct, then select the algorithm $\mathsf{HECHLV}^{2\to 1}$. The algorithm $\mathsf{HECHLV}^{2\to 2}$ is as follows:

Algorithm 9 $\mathsf{HECHLV}^{2\to 2}$

```
Input:  $D_2 = (U_2, V_2) = (x^2 + u_{20}, v_{21}x + v_{20})$
Output: $D_1 = (U_1, V_1) = (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10}) = \frac{1}{2}D_2,\ h_2 \neq 0$
step  procedure
1.    Solve $k_1h_2 + 1 = 0$
      $k_1 = 1/h_2$
2.    Solve $k_1h_0 + k_0h_1 + c_1 = 0$
      $c_2 \leftarrow f_4,\ c_1 \leftarrow f_3 + h_2v_{21} + u_{20} + c_2u_{21}$
      $c_0 \leftarrow f_2 + h_2v_{20} + (h_1 + v_{21})v_{21} + c_2u_{20} + c_1u_{21}$
      $k_0 = (k_1h_0 + c_1)/h_1$
3.    Select correct algorithm by checking trace of $xh_2 + x^2u_{11} + 1 = 0$
      $u_{11} \leftarrow \sqrt{k_1h_1 + k_0h_2 + k_1^2u_{20} + k_0^2 + c_2}/k_1,\ \gamma \leftarrow u_{11}/h_2^2$
      if $\mathrm{Tr}(\gamma) = 1$ then $D_1 \leftarrow \mathsf{HECHLV}^{2\to 1}(D_2)$, return $D_1$
3.    Compute $U_1$
      $u_{10} \leftarrow \sqrt{k_0h_0 + k_0^2u_{20} + c_0}/k_1$
4.    Compute $V_1 = V_2 + h + kU_2 \bmod U_1$
      $w \leftarrow h_2 + k_1,\ v_{11} \leftarrow h_1 + k_1u_{20} + k_0 + u_{11}w,\ v_{10} \leftarrow v_{20} + h_0 + k_0u_{20} + u_{10}w$
5.    $D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10})$, return $D_1$
```

5 Halving Algorithm for Other Curves

In this section, we focus on other curves: (1) $h(x)$ is reducible in $\mathbb{F}_{2^n}$ with $\deg h = 2$, and (2)the special curve with $\deg h = 1$, i.e. $h_2 = 0$.

Let $h(x)$ be a reducible polynomial of degree 2, namely $h(x) = (x + x_1)(x + x_2)$ where $x_1, x_2 \in \mathbb{F}_{2^n}$. Assume that $x_1 \neq x_2$, then there are three different divisor classes of order 2, say D_1, D_2, and D_3 (See Appendix A). In this case, Lemma 1 is no longer true, and there are four different candidates of the halved value arisen from equation (3) and equation (4). They are equal to $\frac{1}{2}D$, $\frac{1}{2}D + D_1$, $\frac{1}{2}D + D_2$, and $\frac{1}{2}D + D_3$. In order to determine the proper divisor class, we have to check the trace of both equation (3) and (4). Therefore the halving algorithm for this case requires more number of field operations than that required for the general curve. If $x_1 = x_2$ holds, we know $h_1 = 0$ and there is only one divisor class of degree 2. In this case, equation (4) has a unique root for each solution k_1 of equation (3), namely we have only two candidates of the halved value. It can be distinguished by the trace of equation $xh_2 + x^2h_{11} + 1 = 0$ as we discussed in Lemma 1.

For the special curve of $\deg h = 1$, we have only one value k_1 not two, recall for the general curve, there are two value k_1 and k_1' and we need to select correct one. This is the main difference between the general curve and the special curve. For the special curve, we obtain a system of equations related to variables k_0, k_1, u_{11}, and u_{10} by the same method for the general curve.

$$k_1^2u_{21} + 1 = 0 \tag{7}$$

$$k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0 \tag{8}$$

In the case of the general curve, we select correct k_0 by checking trace of the degree two equation of k_1 in next halving. If this equation has roots (no roots) i.e. trace is zero, k_0 is correct (not correct). In the case of the special curve, on the other hand, we have only one value k_1 from equation (7), so we select correct k_0 by checking a degree two equation (8) of k_0 in next halving, instead of the equation of k_1. If the equation of k_0 in next halving has roots (no roots), k_0 is correct (not correct). We show an example of the algorithm for the special curve $h(x) = x$ in Appendix D.

6 Conclusion

In this paper, we presented the first divisor class halving algorithm for HECC of genus 2, which is as efficient as the previously known doubling algorithm. The proposed formula is an extension of the halving formula for elliptic curves reported by Knudsen [Knu99] and Schroeppel [Sch00], in which the halved divisor classes are computed by solving some special equations that represent the doubled divisor class. Because the doubling formula for HECC is relatively complicated, the underlying halving algorithm is in general less efficient than that for elliptic curves. However, we specified two crucial equations whose common solutions contain the proper halved values, then an algorithm for distinguishing a proper value was presented. Our algorithm's improvement over the previously known fastest doubling algorithm is up to about 10%. Moreover, the proposed algorithm is complete — we investigated the exceptional procedures appeared in the divisor class halving algorithm, for example, operations with divisor classes whose weight is one. The presented algorithm has not been optimized yet, and there is a possibility to enhance its efficiency.

References

[Ava04] R. Avanzi, "Aspects of Hyperelliptic Curves over Large Prime Fields in Software Implementations," CHES 2004, LNCS 3156, pp.148-162, 2004.

[ACF04] R. Avanzi, M. Ciet, and F. Sica, "Faster Scalar Multiplication on Koblitz Curves Combining Point Halving with the Frobenius Endomorphism," PKC 2004, LNCS 2947, pp.28-40, 2004.

[Can87] D. Cantor, "Computing in the Jacobian of a Hyperelliptic Curve," Mathematics of Computation, 48, 177, pp.95-101, 1987.

[Duq04] S. Duquesne, "Montgomery Scalar Multiplication for Genus 2 Curves," ANTS 2004, LNCS 3076, pp.153-168, 2004.

[FHL$^+$03] K. Fong, D. Hankerson, J. López, and A. Menezes, "Field inversion and point halving revised," Technical Report CORR2003-18, http://www.cacr.math.uwaterloo.ca/techreports/2003/corr2003-18.pdf

[GH00] P. Gaudry and R. Harley, "Counting Points on Hyperelliptic Curves over Finite Fields," ANTS 2000, LNCS 1838, pp.313-332, 2000.

[HHM00] D. Hankerson, J. Hernandez, A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields," CHES 2000, LNCS 1965, pp.1-24, 2000.

[Har00a] R. Harley, “Adding.txt,” 2000. http://cristal.inria.fr/˜harley/hyper/

[Har00b] R. Harley, “Doubling.c,” 2000. http://cristal.inria.fr/˜harley/hyper/

[KR04] B. King and B. Rubin, “Improvements to the Point Halving Algorithm,” ACISP 2004, LNCS 3108, pp.262-276, 2004.

[Kob89] N. Koblitz, “Hyperelliptic Cryptosystems,” Journal of Cryptology, Vol.1, pp.139-150, 1989.

[Knu99] E. Knudsen, “Elliptic Scalar Multiplication Using Point Halving,” ASIACRYPT ’99, LNCS 1716, pp.135-149, 1999.

[Lan02a] T. Lange, “Efficient Arithmetic on Genus 2 Hyperelliptic Curves over Finite Fields via Explicit Formulae,” Cryptology ePrint Archive, 2002/121, IACR, 2002.

[Lan02b] T. Lange, “Inversion-Free Arithmetic on Genus 2 Hyperelliptic Curves,” Cryptology ePrint Archive, 2002/147, IACR, 2002.

[Lan02c] T. Lange, “Weighed Coordinate on Genus 2 Hyperelliptic Curve,” Cryptology ePrint Archive, 2002/153, IACR, 2002.

[Lan04a] T. Lange, “Montgomery Addition for Genus Two Curves,” ANTS 2004, LNCS 3076, pp.309-317, 2004.

[Lan04b] T. Lange, “Foumulae for Arithmetic on Genus 2 Hyperelliptic Curves,” J.AAECC Volume 15, Number 5, pp.295-328, 2005.

[LS04] T. Lange, M. Stevens, “Efficient Doubling on Genus Two Curves over Binary Fields,” SAC 2004, pre-proceedings, pp.189-202, 2004.

[Men93] A. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, 1993.

[Mum84] D. Mumford, *Tata Lectures on Theta II*, Progress in Mathematics 43, Birkhäuser, 1984.

[MCT01] K. Matsuo, J. Chao and S. Tsujii, “Fast Genus Two Hyperelliptic Curve Cryptosystems,” Technical Report ISEC2001-31, IEICE Japan, pp.89-96, 2001.

[PWP03] J. Pelzl, T. Wollinger, and C. Paar, “High Performance Arithmetic for Hyperelliptic Curve Cryptosystems of Genus Two,” Cryptology ePrint Archive, 2003/212, IACR, 2003.

[PWG$^+$03] J. Pelzl, T. Wollinger, J. Guajardo and C. Paar, “Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves,” CHES 2003, LNCS 2779, pp.351-365, 2003.

[Sch00] R. Schroeppel, “Elliptic curve point halving wins big. 2nd Midwest Arithmetic Geometry in Cryptography Workshop, Urbana, Illinois, November 2000.

[SMC$^+$02] T. Sugizaki, K. Matsuo, J. Chao, and S. Tsujii, “An Extension of Harley Addition Algorithm for Hyperelliptic Curves over Finite Fields of Characteristic Two,” Technical Report ISEC2002-9, IEICE Japan, pp.49-56, 2002.

A The Divisor Class of Order 2

We show that the order of Jacobian $\mathbf{J}$ over $\mathbb{F}_{2^n}$ of genus 2 is always divisible by 2.

Let T_2 be the divisor represented by $T_2 = (g, v_T)$, where g is a divisor of MakeMonic(h) with $\deg g > 0$ and v_T is uniquely determined from h due to Mumford representation. It is easy to check $2T_2 = O$ via Cantor Algorithm. Indeed, we know $d =$ MakeMonic(h) in Step 1, and thus $(U_3, V_3) = (1, 0)$ holds in Step 4.

Next, we prove that T_2 is always in $\mathbf{J}(\mathbb{F}_{\mathbf{2^n}})$. First assume that h is degree 1, namely $h(x) = h_1x + h_0$. Then v_T is y_1 such that $y_1^2 = f(h_0/h_1)$ due to $h|(v_T^2 + f)$.

If h is reducible of degree 2, we can represent MakeMonic$(h(x)) = (x+x_1)(x+x_2)$, where $x_1, x_2 \in \mathbb{F}_{2^n}$ are x-coordinate of points on the definition curve. The corresponding y-coordinate can be computed by solving $y_i^2 = f(x_i)$ for $i = 1, 2$, respectively. From Mumford representation, we obtain $v_T = v_1x + v_0 \in \mathbb{F}_{2^n}[x]$ as follows:

$$v_0 = \frac{y_1x_2 + y_2x_1}{x_1 + x_2}, \quad v_1 = \frac{y_1 + y_2}{x_1 + x_2}, \tag{9}$$

except the case of $x_1 = x_2$. Consequently, we can calculate $v_T \in \mathbb{F}_{2^n}[x]$. Set $D_3 = ((x + x_2)(x + x_2), v_1x + v_0)$. Similarly, $D_i = (x + x_i, y_i)$ is a divisor of order 2, where $y_i = f(x_1)$ for $i = 1, 2$. We notice that $\{O, D_1, D_2, D_3\}$ is a quaternion group of Klein with multiplication rules $D_1 + D_2 = D_3, D_2 + D_3 = D_1$, and $D_3 + D_1 = D_2$. If $x_1 = x_2$ holds, the order of the divisor $(x + x_1, y_1)$ with $y_1^2 = f(x_1)$ is divisible by 2.

In the following we assume that h is irreducible of degree 2. Set $h(x) = h_2x^2+h_1x+h_0$. The solutions x_1, x_2 of $h(x) = 0$ is not in $\mathbb{F}_{2^n}$, and we can not apply the algorithm used for the reducible h of degree 2. We show how to construct $v_T = v_1x+v_0 \in \mathbb{F}_{2^n}[x]$ explicitly. We have

$$\frac{h_1}{h_2} = x_1 + x_2, \quad \frac{h_0}{h_2} = x_1x_2, \quad y_i = \sqrt{x_i^5 + f_4x_i^4 + f_3x_i^3 + f_2x_i^2 + f_1x_i + f_0}, \tag{10}$$

where $i = 1, 2$ for the definition polynomial f of the underlying curve. From these equations we can explicitly write down v_0, v_1 in the following.

$$\begin{aligned} y_1x_2 + y_2x_1 &= \sqrt{x_1^5}x_2 + \sqrt{f_4x_1^4}x_2 + \sqrt{f_3x_1^3}x_2 + \sqrt{f_2x_1^2}x_2 + \sqrt{f_1x_1}x_2 + \sqrt{f_0}x_2 \\ &\quad + \sqrt{x_2^5}x_1 + \sqrt{f_4x_2^4}x_1 + \sqrt{f_3x_2^3}x_1 + \sqrt{f_2x_2^2}x_1 + \sqrt{f_1x_2}x_1 + \sqrt{f_0}x_1 \\ &= x_1x_2\sqrt{x_1^3 + x_2^3} + x_1x_2\sqrt{f_4}(x_1 + x_2) + x_1x_2\sqrt{f_3}\sqrt{x_1 + x_2} + 2x_1x_2\sqrt{f_2} \\ &\quad + \sqrt{f_1}\sqrt{x_1x_2}\sqrt{x_1 + x_2} + \sqrt{f_0}(x_1 + x_2) \\ &= \frac{h_0}{h_2}\sqrt{\left(\frac{h_1}{h_2}\right)^3 + \frac{h_0h_1}{h_2^2}} + \sqrt{f_4}\frac{h_0h_1}{h_2^2} + \sqrt{f_3}\frac{h_0}{h_2}\sqrt{\frac{h_1}{h_2}} + \sqrt{f_1}\sqrt{\frac{h_0h_1}{h_2^2}} + \sqrt{f_0}\frac{h_1}{h_2} \end{aligned}$$

This equation contains only coefficients of h and f. In the transformation above we used the relationship:

$$x_1^3 + x_2^3 = (x_1 + x_2)^3 + x_1x_2(x_1 + x_2) = \left(\frac{h_1}{h_2}\right)^3 + \frac{h_0h_1}{h_2^2}$$

Therefore we have obtained the explicit formula for calculating $v_0 \in \mathbb{F}_{2^n}$.

$$v_0 = \frac{y_1x_2 + y_2x_1}{x_1 + x_2} = \frac{h_0}{h_1}\sqrt{\left(\frac{h_1}{h_2}\right)^3 + \frac{h_0h_1}{h_2^2}} + \sqrt{f_4}\frac{h_0}{h_2} + \sqrt{f_3}\frac{h_0}{h_1}\sqrt{\frac{h_1}{h_2}} + \sqrt{f_1}\sqrt{\frac{h_0}{h_1}} + \sqrt{f_0}$$

Similarly $v_1 \in \mathbb{F}_{2^n}$ can be calculated as follows:

$$v_1 = \frac{y_1 + y_2}{x_1 + x_2} = \sqrt{\left(\frac{h_1}{h_2}\right)^3 + \frac{h_0}{h_2}\left(\frac{h_1}{h_2} + \frac{h_0}{h_1}\right)} + \sqrt{f_4}\frac{h_1}{h_2} + \sqrt{f_3}\sqrt{\frac{h_1}{h_2} + \frac{h_0}{h_1}} + \sqrt{f_2} + \sqrt{f_1}\sqrt{\frac{h_2}{h_1}}$$

B Proof of Lemma

Lemma 1. *Let $h(x)$ be an irreducible polynomial of degree 2. There is only one value k_1 which satisfies both equations (3) and (4). Equation (4) has a solution only for the correct k_1. There is only one value k_0 which yields the halved divisor D_1 in algorithm* HECHLV. *Equation $xh_2 + x^2u_{11} + 1 = 0$ has a solution only for the correct k_0.*

Proof. In this paper we assume that the order of $\mathbf{J}(\mathbb{F}_{2^n})$ is $2 \times r$, where r is a large prime number. For a given divisor D_2, there are two points whose doubled reduced divisor is equal to D_2. The halved divisor equals either $\frac{1}{2}D_2$ or $\frac{1}{2}D_2 + T_2$, where T_2 is an element in the kernel of the multiplication-by-two in $\mathbf{J}(\mathbb{F}_{2^n})$. We call $\frac{1}{2}D_2$ the proper halved divisor.

From the condition of coeff$(U', 3) = 0$, equation (3) is always solvable. For each solution of equation (3), there exist two solutions of equation (4) due to coeff$(U', 1) = 0$. From these values we obtain four different divisors using equations (5),(6), and one of them is the proper halved value $\frac{1}{2}D$. In the following, we discuss how to select the proper divisor.

At first we prove that if $h(x)$ is an irreducible polynomial, equation (4) is solvable only for one solution of equation (3). It is well known that equation $ax^2 + bx + c = 0$ has roots if and only if $\mathrm{Tr}(ac/b^2) = 0$. Let one root of $ax^2 + bx + c = 0$ be x_0 and the other be $x_0 + b/a$. Let the two roots of equation (3) be k_1 and $k_1' = k_1 + h_2/u_{21}$. Equation (4) with k_1' substituted is as follows:

$$(k_1 + h_2/u_{21})h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0. \quad (11)$$

Now we compute $\mathrm{Tr}(ac/b^2)$ of the equation (11):

$$\mathrm{Tr}(((k_1 + h_2/u_{21})h_0 + c_1)u_{21}/h_1^2) = \mathrm{Tr}((k_1h_0 + c_1)u_{21}/h_1^2) + \mathrm{Tr}((h_0h_2)/h_1^2) \quad (12)$$

Because $\mathrm{Tr}(ac/b^2)$ of the equation (4) is $\mathrm{Tr}((k_1h_0 + c_1)u_{21}/h_1^2)$, if $\mathrm{Tr}((h_0h_2)/h_1^2) = 1$ i.e. $h(x) = h_2x^2 + h_1x + h_0$ is irreducible, one equation has two roots and the other equation has no roots. This leads to the uniqueness of k_1. Therefore, we can select a proper value from $\{k_1, k_1'\}$ by checking the trace of equation (4).

Next we show how to choose the proper k_0. Equation (4) has two roots k_0 and k_0' for the proper k_1 described above. Two different halved divisor $\frac{1}{2}D_2$ and $\frac{1}{2}D_2 + T_2$ can be obtained by k_0 and k_0'. We will distinguish the proper divisor by applying the above halving algorithm again. Let $D_1 = \frac{1}{2}D_2$. The halving algorithm for D_1 yields two divisors $\frac{1}{2}D_1 \in \mathbf{J}(\mathbb{F}_{2^n})$ and $\frac{1}{2}D_1 + T_2 \in \mathbf{J}(\mathbb{F}_{2^n})$. On the other hand, for $D_1' = \frac{1}{2}D_2 + T_2$, there are two halved divisors: $\frac{1}{2}D_1' + T_4$ and $\frac{1}{2}D_1' + 3T_4$, where T_4 and $3T_4$ are two divisors of order four in $\mathbf{J}$ not in $\mathbf{J}(\mathbb{F}_{2^n})$, namely $\frac{1}{2}D_1' + T_4 \notin \mathbf{J}(\mathbb{F}_{2^n})$ and $\frac{1}{2}D_1' + 3T_4 \notin \mathbf{J}(\mathbb{F}_{2^n})$. Therefore, the proper k_0 should satisfy halved D_1 in $\mathbf{J}(\mathbb{F}_{2^n})$. In the other words, if and only if k_0 (or k_0') is proper, equation $xh_2 + x^2u_{11} + 1 = 0$ has two roots over $\mathbb{F}_{2^n}$, where u_{11} is computed from k_0 (or k_0') using equation (5). Consequently, we can select the proper k_0 by checking the trace of equation $xh_2 + x^2u_{11} + 1 = 0$ for $u_{11} \neq 0$. The case of $u_{11} = 0$ occurs with negligible probability, but we can select the proper k_0 as follows: Let u_{11} and u_{11}' be the coefficient of equation (5) for two candidates k_0 and k_0', respectively. Note that if $u_{11} = 0$ holds, then $u_{11}' \neq 0$ for $h_1/h_2 \neq u_{21}$. Therefore, the proper one can be selected by checking $\mathrm{Tr}(u_{11}'/h_2^2) = 1$. If $h_1/h_2 = u_{21}$ holds, we can use the formula $\mathsf{HECHLV}^{2\to 2}$ described in Section 5. $\square$

C Improved Algorithm with fixed base point

Algorithm 10 HECHLV*($h_2 = 1, f_4 = 0$, fixed base point)*

Input: $D_2 = (U_2, V_2),\ 1/h_1^2,\ t_0,\ t_1$
Output: $D_1 = (U_1, V_1) = \frac{1}{2}D_2$
$U_i = x^2 + u_{i1}x + u_{i0},\ V_i = v_{i1}x + v_{i0},$ where $i = 1, 2$

step	*procedure*	*cost*
1.	**Solve** $k_1 + k_1^2u_{21} + 1 = 0$ $\alpha \leftarrow 1/u_{21}$ if $(t_1 = 0)$ then $k_1 \leftarrow H(u_{21})\alpha$ else $k_1 \leftarrow (H(u_{21}) + 1)\alpha$	$1M + 1I + 1H$
2.	**Solve** $k_1h_0 + k_0h_1 + k_0^2u_{21} + c_1 = 0$ $c_1 \leftarrow f_3 + v_{21} + u_{20} + u_{21}^2$ $c_0 \leftarrow f_2 + v_{20} + v_{21}(h_1 + v_{21}) + u_{21}(u_{20} + c_1)$ $w_0 \leftarrow u_{21}/h_1^2,\ \alpha \leftarrow h_1\alpha, \gamma \leftarrow (c_1 + k_1h_0)w_0$ if $(t_0 = 0)$ then $k_0 \leftarrow H(\gamma)\alpha$ else $k_0 \leftarrow (H(\gamma) + 1)\alpha$	$7M + 1S + 1H$ $(h_1 = 1 : v_{21}(h_1 + v_{21}) = v_{21} + v_{21}^2)$
3.	**Compute** U_1 $w_0 \leftarrow k_1^2,\ w_1 \leftarrow w_0u_{20} + k_1h_1 + u_{21},\ w_2 \leftarrow k_0 + \sqrt{w_1 + k_0}$ $w_4 \leftarrow k_1u_{21} + 1,\ u_{11} \leftarrow w_2w_4$ $w_1 \leftarrow k_0u_{20},\ w_5 \leftarrow w_4 + 1,\ w_6 \leftarrow (k_0 + k_1)(u_{20} + u_{21})$ $u_{10} \leftarrow w_4\sqrt{k_0(w_1 + h_0) + c_0}$	$8M + 1S + 2SR$
4.	**Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$ $w_4 \leftarrow w_5 + k_0 + 1,\ w_5 \leftarrow w_1 + w_5 + w_6 + v_{21} + h_1$ $w_6 \leftarrow w_1 + v_{20} + h_0,\ w_7 \leftarrow w_2 + w_4$ $w_1 \leftarrow w_7u_{10}\ w_3 \leftarrow (k_1 + w_7)(u_{10} + u_{11})$ $v_{11} \leftarrow w_1 + w_2 + w_3 + w_5,\ v_{10} \leftarrow w_1 + w_6$	$2M$
5.	$D_1 \leftarrow (x^2 + u_{11}x + u_{10}, v_{11}x + v_{10})$, return D_1	
total		$18M + 2S + 1I + 2SR + 2H$
	$h_1 = 1$	$14M + 3S + 1I + 2SR + 2H$
	$h_1 = h_0 = 1$	$13M + 3S + 1I + 2SR + 2H$

D Halving Algorithm for the Special Curve: $h(x) = x$

Algorithm 11 HEC_HLV$(y^2 + xy = x^5 + f_1 x + f_0)$

Input: $D_2 = (U_2, V_2)$
Output: $D_1 = (U_1, V_1)$, $U_i(x) = x^2 + u_{i1}x + u_{i0},\ V_i = v_{i1}x + v_{i0},\ \gcd(U_i, h) = 1$

step	*procedure*	*cost*
1.	**Solve** $k_1^2 u_{21} + 1 = 0$	$1I + 1SR$
	$w_0 \leftarrow 1/u_{21},\ k_1 \leftarrow \sqrt{w_0}$	
2.	**Solve** $k_0 + k_0^2 u_{21} + c_1 = 0$	$3M + 2S + 1SR + 1H$
	$c_1 \leftarrow u_{20} + u_{21}^2,\ w_1 \leftarrow c_1 u_{21},\ c_0 \leftarrow v_{21} + v_{21}^2 + u_{21}u_{20} + w_1$	
	$invk_1 \leftarrow \sqrt{u_{21}},\ w_2 \leftarrow H(w_1),\ w_3 \leftarrow w_2 + 1$	
	$k_0 \leftarrow w_0 w_2,\ k_0' \leftarrow k_0 + w_0$	
3.	**Compute** U_1	$4M + 3SR + 1T$
	$u_{11} \leftarrow \sqrt{invk_1 + k_0}, u_{10} \leftarrow \sqrt{(k_0 + c_1)u_{20} + c_0 u_{21}}$	
	if $Tr(u_{11}(u_{10} + invk_1 + k_0)) = 1$ *then*	
	$k_0 \leftarrow k_0',\ w_2 \leftarrow w_3,\ u_{11} \leftarrow u_{11} + k_1,\ u_{10} \leftarrow u_{10} + \sqrt{w_0 u_{20}}$	
5.	**Compute** $V_1 = V_2 + h + kU_2 \bmod U_1$	$5M$
	$w_1 \leftarrow k_1(u_{21} + u_{11}) + k_0$	
	$v_{11} \leftarrow k_1(u_{20} + u_{10}) + w_2 + v_{21} + 1 + u_{11}w_1$	
	$v_{10} \leftarrow k_0 u_{20} + v_{20} + u_{10}w_1$	
total	k_0 *is correct*	$11M + 2S + 1I + 4SR + 1H + 1T$
	k_0' *is correct*	$12M + 2S + 1I + 5SR + 1H + 1T$