

Cantor versus Harley: Optimization and Analysis of Explicit Formulae for Hyperelliptic Curve Cryptosystems

Thomas Wollinger, Jan Pelzl, and Christof Paar, *Member, IEEE*

Abstract—Hyperelliptic curves (HEC) look promising for cryptographic applications, because of their short operand size compared to other public-key schemes. The operand sizes seem well suited for *small* processor architectures, where memory and speed are constrained. However, the group operation has been believed to be too complex and thus, HEC have not been used in this context so far.

In recent years, a lot of effort has been made to speed up group operation of genus-2 HEC. In this contribution, we increase the efficiency of the genus-2 and genus-3 hyperelliptic curve cryptosystems (HECC). For certain genus-3 curves we can gain almost 80% performance for a group doubling. This work not only improves Harley’s algorithm [1], but also improves the original algorithm introduced by Cantor [2]. Contrary to common belief, we show that it is also practical for certain curves to use Cantor’s algorithm to obtain the highest efficiency for the group operation. In addition, we introduce a general reduction method for polynomials according to Karatsuba. We implemented our most efficient group operations on Pentium and ARM microprocessors.

Index Terms—hyperelliptic curves, explicit formulae, Harleys algorithm, Cantor, efficient implementation, embedded implementation

I. INTRODUCTION

IN 1976, Diffie and Hellman [3] revolutionized the field of cryptography by introducing the concept of public-key (PK) cryptography in the open literature. Their key exchange protocol is based on the difficulty of solving the discrete logarithm problem (DLP) over a finite field. Shortly afterwards, a PK algorithm based on the difficulty to solve the integer factorization problem, namely RSA [4], was introduced. RSA is the most widely used public-key encryption in cryptographic applications. In the mid 1980s, a variant of the Diffie-Hellman key exchange was published based on the difficulty of the DLP in the group of points of an elliptic curve (EC) over a finite field [5], [6]. It is important to point out that elliptic curve cryptosystems (ECC) benefit from shorter operand sizes when compared to RSA or discrete logarithm (DL) based systems.

In 1988, Koblitz suggested for the first time the generalization of EC to curves of higher genus for cryptographic use, namely hyperelliptic curves [7]. The operand size of a hyperelliptic curve cryptosystems is even shorter compared to ECC. This fact makes HECC potentially well suited for small

processors and memory constrained environments. It is widely accepted that for most cryptographic applications based on EC or HEC, one needs a group order of at least $\approx 2^{160}$. Thus, for HECC over \mathbb{F}_q we will need at least $g \cdot \log_2 q \approx 2^{160}$, where g is the genus of the curve. For example, a curve of genus two will need a field \mathbb{F}_q with $|\mathbb{F}_q| \approx 2^{80}$, i.e., 80-bit long operands.

Koblitz’s idea to use HEC for cryptographic applications has been analyzed and implemented both in software [8]–[16], and in more hardware-oriented platforms such as field programmable gate arrays (FPGAs) [17]–[22]. Quite recently, the research community put a lot of effort into increasing the efficiency of HEC group operations [1], [14], [15], [23]–[29]. However, most of the improvements concentrate on genus-2 curves and *all* contributions improve the algorithm proposed in [1]. We will refer to this algorithm as Harley’s algorithm. In our contribution we not only improve Harley’s algorithm, but also the original algorithm introduced by Cantor [2] (referred as Cantor’s algorithm). At this point we would like to note, that Harley’s algorithm is derived from Cantor’s algorithm and uses posteriori knowledge of polynomials to determine them. As a major finding we show that it is beneficial for certain curves to use Cantor’s algorithm to achieve higher speed compared to Harley’s algorithm.

Our Main Contributions¹

The group operation of HECC was originally computed using the algorithm presented by Cantor [2]. Cantor’s algorithm is based on elements in the quotient ring using Mumford’s representation [32]. For the first time, we present an explicit notation of the original Cantor algorithm. We were able to speed up Cantor’s group addition and group doubling for genus-2 and 3 HEC compared to Nagao’s implementation [33]. Contrary to common belief, we reached similar or even better performance for the doubling operation for certain curves compared to Harley’s algorithm. For genus-3 HECC and $h(x) = 1$, our group doubling formula saves almost 80% of the field multiplications/squarings compared to [33].

In [1], the authors presented a modified version of the original Cantor algorithm. We present an optimization for the explicit formulae using certain genus-2 HEC. In addition we show the generalized explicit formulae for genus-3 curves including fields of characteristic 2. For certain curves our group doubling formula saves more than 60% of the field multiplications/squarings compared to [15].

In addition, we give an extended description of all the

Manuscript received August ??, 2004; revised ??, 2004.

The authors are with the Department of Electrical Engineering and Information Sciences, Communication Security Group (COSY), Ruhr-Universitaet Bochum, Germany, Universitaetsstrasse 150, 44780 Bochum, Germany, Email: {wollinger, pelzl, cpaar}@crypto.rub.de

¹Part of this research was presented in [30], [31]

techniques used to improve the group operation. Moreover, we generalized Karatsuba's method [34] in order to decrease the complexity of the polynomial reduction. We were able to obtain similar cost reduction for efficient reduction compared to Karatsuba multiplication, namely save one multiplication for the additional cost of three additions.

We implemented all group operations introduced in this contribution on a Pentium and on an ARM microprocessor. One of the reasons to do so was to prove the correctness of our newly derived group formulae. Furthermore, we could show that HECC can reach the performance of ECC and, in some cases, even outperforms ECC. The implementation targeting the ARM microprocessor shows that HECC is one of the cryptographic systems well suited for embedded security applications.

Note, that the complexity of the explicit formulae presented should be viewed as upper bound. One can most probably improve the given group operations further, by finding new improved methods or by restructuring the operations and using the methods given.

The remainder of the paper is organized as follows: Section II summarizes previous contributions dealing with the improvements of the HEC group operation. Section III gives a brief overview of the mathematical background related to HECC. Section IV and IV-G present the methods used to improve the group doubling and addition. Section V summarizes our results whereas Section V-A presents the improved HEC group operations and Section V-B introduces the implementation of HECC. Finally, we end this contribution with a discussion of our results and some conclusions.

II. PREVIOUS WORK

In this section, we summarize previous improvements of group operations of genus-2 and genus-3 curves. In the remainder of the paper I refers to a field inversion, M to a field multiplication, and S to a field squaring. In some references, the authors did not distinguish between multiplications and squarings, that is denoted as M/S .

A. Improving Cantor's Algorithm

The formulae given for the group operation of HEC can be written explicitly, resulting in a more efficient execution. To our knowledge, there exists no publication describing an explicit presentation of Cantor's original algorithm.

Nagao improved the polynomial arithmetic for Cantor's algorithm [33]. He mainly applied the following ideas:

- Division of polynomials without field inversions.
- Computation of the greatest common divisor with one inversion.
- Interleaving superfluous calculations in the reduction part.
- Expressing points on the Jacobian in a different form.

Nagao evaluated the computational cost of the group operations by applying the stated improvements for genus $2 \leq g \leq 10$. The best results for genus two and three HEC are stated in Table IV.

Note, that in [33] the author estimates the cost of the operations using polynomial arithmetic. We were able to

decrease the number of operations needed in all cases by optimizing the explicit notation of these group operations. We can reach similar or even higher performance compared to Harley's algorithms for group doubling using certain curves.

B. Improving Harley's Algorithm

The explicit formulae were first presented in [1]. The authors suggested to reduce the number of operations by making the arithmetic explicit. As a consequence, different cases of the properties of the input divisors have to be distinguished. They described an efficient algorithm to reduce the required number of operations. Using the Karatsuba multiplication algorithm [34], the Chinese remainder theorem, the Newton iteration, and a reordering of operations, the authors further reduced the overall complexity of the group operations.

In [24], Lange developed for the first time explicit formulae for even characteristic fields and genus-2 curves following the approach from [1]. Additionally, in this work the author describes the group operation distinguishing between all the different cases of the properties of the input divisors. For the most frequent case the group addition could be computed using 2 inversion, 24 multiplication, and 3 squarings. The group doubling takes 2 inversion, 26 multiplication, and 6 squarings.

The follow-up implementation based on Harley's algorithm, was done by Matsuo, Chao and Tsuji in [23]. They could save some multiplications compared to Harley's algorithm. A group addition takes 2 inversions and 25 multiplications/squarings and a group doubling takes 2 inversions and 27 multiplications/squarings (see Table V). The authors showed an implementation of both the improved Harley algorithm and the elliptic curve algorithm for comparison. Each one was implemented over optimal extension fields (OEF), a 93-bit OEF for hyperelliptic curves of genus 2 and a 186-bit OEF for elliptic curves. A Pentium III@866MHz with the GNU C++-2.95.2 compiler was used.

A further speed-up for HEC of genus 2 of odd characteristic was achieved in 2002 by Miyamoto, Doi, Matsuo, Chao and Tsuji [25]. The authors suggested Montgomery's trick of simultaneous inversions to compute two inverses by performing only one field inversion and three field multiplications (for more details see Chapter IV). The new algorithm was implemented on a Pentium III@886MHz.

In 2002, Masashi Takahashi further improved the arithmetic of the genus-2 curves of odd characteristic [26]. With the help of a small change in the order of a special operation, he could save one multiplication compared to [25].

The extension of the explicit formulae for arithmetic on genus-2 curves of [25] and [26] to fields of even characteristic and to arbitrary equations of the curve was done, independent from each other and almost at the same time in [35] and [14]. In [35], the authors were able to reduce the number of field operation needed, resulting in one inversion and 25 multiplication for the group addition. The group doubling was performed in one inversion and 27 multiplications. The authors used all of the techniques known from the previous publications, like Karatsuba multiplication and Montgomery's multiple inversion technique. In [14], the author was able to further reduce the

complexity of the group operations: I+22M+3S for group addition and I+22M+5S for group doubling. Timings for the implementation of those formulae are also given. Various libraries for the field arithmetic over prime fields and binary fields on a Pentium IV@1.5GHz were investigated. More recently, Lange gives a thorough comparison of arithmetic on hyperelliptic curves of genus-2 curves [36] containing mainly the material of the previous three papers [14], [27], [28].

Genus-3 HEC group operations using odd characteristic were improved applying Harley's algorithm [29]. The authors adopted the methods from [25], [37] to increase the performance. The proposed algorithm was implemented on an Alpha Workstation 21264@667MHz using a underlying field \mathbb{F}_q , where $q = 2^{61} - 1$. Recently, the authors in [15] published explicit formulae targeting genus-3 curves, whereas addition needs $I + 70M/S$ and doubling $I + 71M/S$.

The work at hand introduces explicit formulae for genus-3 curves of arbitrary characteristic. In addition, we were able to improve the formulae for the group operation of genus-2 and genus-3 HEC defined over fields of characteristic two. The computational complexity of those formulae for genus-2 and genus-3 curves and the corresponding references are given in Table V.

III. MATHEMATICAL BACKGROUND

In this section, we present an elementary introduction to the theory of hyperelliptic curves over finite fields of arbitrary characteristic, restricting attention to material that is relevant for this work. For more details, the reader is referred to [38], [39].

A. HECC and the Jacobian

Let \mathbb{F} be a finite field, and let $\overline{\mathbb{F}}$ be the algebraic closure of \mathbb{F} . A hyperelliptic curve C of genus $g \geq 1$ over \mathbb{F} is the set of solutions $(x, y) \in \mathbb{F} \times \mathbb{F}$ to the equation

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

The polynomial $h(x) \in \mathbb{F}[x]$ is of degree at most g and $f(x) \in \mathbb{F}[x]$ is a monic polynomial of degree $2g + 1$. Such a curve is said to be non-singular if there are no pairs $(x, y) \in \overline{\mathbb{F}} \times \overline{\mathbb{F}}$ which simultaneously satisfy the equation of the curve C and the partial differential equations $2y + h(x) = 0$ and $h'(x)y - f'(x) = 0$. For odd characteristic, it suffices to let $h(x) = 0$ and to have $f(x)$ square free.

If we want to define the Jacobian over \mathbb{F} , denoted by $\mathbb{J}_C(\mathbb{F})$, we say that a divisor $D = \sum m_i P_i$ is defined over \mathbb{F} if $D^\sigma = \sum m_i P_i^\sigma$ is equal to D for all automorphisms σ of $\overline{\mathbb{F}}$ over \mathbb{F} . Notice that this does not mean that each P_i^σ is equal to P_i , σ may permute the points.

Cantor concludes from the Riemann-Roch Theorem that each element of the Jacobian can be represented uniquely by a divisor, denoted as reduced divisors [2]. Mumford [32, page 3.17] shows that the divisors of the Jacobian can be represented as a pair of polynomials $u(x)$ and $v(x)$ with $\deg v(x) < \deg u(x) \leq g$, with $u(x)$ dividing $y^2 + h(x)y - f(x)$ and where the coefficients of $u(x)$ and $v(x)$ are elements of \mathbb{F} . In the remainder of this paper, a divisor D represented by polynomials will be denoted by $\text{div}(x, y)$.

B. Cantor's Group Operations on the Jacobian

This section gives a brief description of the algorithms used for adding and doubling divisors on $\mathbb{J}_C(\mathbb{F})$. These group operations will be performed in two steps. First we have to find a semi-reduced divisor $D' = \text{div}(u', v')$, such that $D' \sim D_1 + D_2 = \text{div}(u_1, v_1) + \text{div}(u_2, v_2)$ in the group $\mathbb{J}_C(\mathbb{F})$. In the second step we have to reduce the semi-reduced divisor $D' = \text{div}(u', v')$ to an equivalent divisor $D = (u, v)$. Algorithm 1 describes the group addition.

Algorithm 1 Group addition

Require: $D_1 = \text{div}(u_1, v_1)$, $D_2 = \text{div}(u_2, v_2)$

Ensure: $D = \text{div}(u_3, v_3) = D_1 + D_2$

- 1: $d = \gcd(u_1, u_2, v_1 + v_2 + h) = s_1 u_1 + s_2 u_2 + s_3 (v_1 + v_2 + h)$
 - 2: $u' = u_1 u_2 d^{-2}$
 - 3: $v' = [s_1 u_1 v_2 + s_2 u_2 v_1 + s_3 (v_1 v_2 + f)] d^{-1} \pmod{u'}$
 - 4: $k = 0$
 - 5: **while** $\deg u'_k > g$ **do**
 - 6: $k = k + 1$
 - 7: $u'_k = \frac{f - v'_{k-1} h - (v'_{k-1})^2}{u'_{k-1}}$
 - 8: $v'_k = (-h - v'_{k-1}) \pmod{u'_k}$
 - 9: **end while**
 - 10: **Output** $(u_3 = u'_k, v_3 = v'_k)$
-

Doubling a divisor is easier than general addition and therefore, Steps 1, 2, and 3 of Algorithm 1 can be simplified as follows:

- 1: $d = \gcd(u, 2v + h) = s_1 u + s_3 (2v + h)$
- 2: $u' = u^2 d^{-2}$
- 3: $v' = [s_1 u v + s_3 (v^2 + f)] d^{-1} \pmod{u'}$

This publication is the first to write Cantor's formulae in an explicit form, see Table VII, VIII, XI, XII, XIII.

C. Harley's Group Operations on a Jacobian

In [1], the authors noticed that one can reduce the number of operations by distinguishing between possible cases according to the properties of the input divisors. They described an efficient algorithm (using Karatsuba multiplication, CRT, and Newton Iteration) to reduce the overall complexity of the group operations.

To determine explicit formulae, it is essential to know the weight of the input divisor. The weight of a divisor is defined as the number of its points [39]. For example, in the case of a hyperelliptic curve of genus 2, a divisor can have weight 0, 1 or 2. For each case, implementations of different explicit formulae are required, thus different functions are called.

Two polynomials have a linear factor in common with probability of $\approx 1/q$. Hence, in the case of a hyperelliptic curve of genus 2, we have no factor in common with the probability $P \approx 1 - 2^{-80}$. Therefore, in most cases the $\gcd(u_1, u_2) = 1$. For the remaining of the paper, we call this the *frequent case* and, for simplicity, we only consider this case in all the optimizations and implementations. An implementation based on explicit formulae can be realized by avoiding the non frequent cases. This can be done on basis of a

protocol which restarts if a non frequent case occurs. Without loss of generality (WLOG) we will consider only the cases for genus-2 curves in the following.

1) *The Frequent Case of Doubling: The frequent case of doubling a divisor occurs if and only if $D = (u_1, v_1)$ is of weight 2 and $\gcd(u_1, \tilde{h}) = 1$ with $\tilde{h} = h + 2v_1$, i.e. both u_1 and \tilde{h} do not have a factor in common.*

According to Cantor's algorithm, $u' = u_1^2$. The v' polynomial can be calculated using the property $u_1(x)|(v_1(x)^2 - f(x))$ and therefore $u'(x)|(v'(x)^2 - f(x))$. Since $v' \equiv v_1 \pmod{u_1}$, $v'(x)$ can be seen as a square root of $f(x)$ modulo $u_1^2(x)$ and $v_1(x)$ as a square root of $f(x)$ modulo $u_1(x)$. Hence, we can obtain $v'(x)$ by performing one step of the Newton Iteration [1]:

$$v' \equiv v_1 - \frac{f - v_1^2}{2v_1} \pmod{u'}. \quad (1)$$

In order to obtain a unique representation of D' , we need to reduce the polynomials $u'(x)$ and $v'(x)$. The polynomials of the reduced divisor $D_2 = (u_2, v_2)$ are

$$\begin{aligned} u_2 &= \frac{f - (v')^2}{u'} & (2) \\ u_2 &= \text{monic}(u_2) \\ v_2 &\equiv -v' \pmod{u_2}. \end{aligned}$$

Equations (1) and (2) can be computed efficiently using following substitutions proposed by Harley:

$$\text{let } k = \frac{f - v^2}{u} \text{ and } s = \frac{k}{2v} \pmod{u},$$

then Equation (1) simplifies to $v' = v - us$ and Equation (2) can be rewritten:

$$u' = \frac{f - (v')^2}{u'} \quad (3)$$

$$= \frac{(v^2 - f) + 2suv + s^2u^2}{u^2} \quad (4)$$

$$= s^2 - \frac{k - 2sv}{u}. \quad (5)$$

Algorithm 2 combines all steps of the most frequent case for arbitrary characteristic including all subexpressions [14].

Algorithm 2 Frequent Case for Group Doubling ($g=2$)

Require: $D_1 = \text{div}(u_1, v_1)$

Ensure: $D_2 = \text{div}(u_2, v_2) = 2D_1$

1: $k = \frac{v_1^2 - v_1 h - f}{u_1}$ (exact division)

2: $s \equiv \frac{k}{h + 2v_1} \pmod{u_1}$

3: $u' = s^2 + \frac{k - s(h + 2v_1)}{u_1}$ (exact division)

4: $u_2 = u'$ made monic

5: $v_2 \equiv -(h + s u_1 + v_1) \pmod{u_2}$

In the case of genus-3 curves an extra reduction step is necessary to obtain a reduced divisor. Hence, for genus-3 curves, Steps 6 and 7 have to be added to Algorithm 2 in order to end up with a reduced divisor $D_3 = \text{div}(u_3, v_3) = 2D_1$.

$$6: \quad u_3 = \frac{f - v_2 h - (v_2)^2}{u_2} \quad (\text{exact division})$$

$$7: \quad v_3 \equiv -(v_2 + h) \pmod{u_3}$$

2) *The Frequent Case of Adding: The frequent case of adding a divisor occurs if and only if $D_1 = (u_1, v_1)$ and $D_2 = (u_2, v_2)$ are both of weight 2 and $\gcd(u_1, u_2) = 1$, i.e. both u_1 and u_2 do not have a factor in common.*

u' is given as $u' = u_1 u_2$. Since in the frequent case $d = d_1 = 1$, $s_1 = e_1$, $s_2 = e_2$ and $s_3 = 0$, v' is obtained by $v' = (e_1 u_1 v_2 + e_2 u_2 v_1) \pmod{u'}$. Applying Garner's Algorithm for the Chinese Remainder Theorem [40] results in:

$$v' = \left[\left(\frac{v_2 - v_1}{u_1} \pmod{u_2} \right) u_1 + v_1 \right] \pmod{u'}.$$

Since $D' = (u', v')$ is a semi-reduced divisor, a reduction is necessary. Harley performs a reduction which is optimized by reusing some pre-computed quantities. The reduction of u' can be written as

$$\begin{aligned} u' &= \frac{f - (v')^2}{u'} \\ &= \frac{1}{u_1 u_2} \left\{ f - \left[\left(\frac{v_2 - v_1}{u_1} \pmod{u_2} \right) u_1 \right]^2 \right. \\ &\quad \left. - 2v_1 u_1 \left(\frac{v_2 - v_1}{u_1} \pmod{u_2} \right) - v_1^2 \right\} \\ &= \frac{1}{u_2} \left\{ \frac{f - v_1^2}{u_1} - \left(\frac{v_2 - v_1}{u_1} \pmod{u_2} \right) \right. \\ &\quad \left. \left[\left(\frac{v_2 - v_1}{u_1} \pmod{u_2} \right) u_1 + 2v_1 \right] \right\} \end{aligned}$$

In the next step, u' is made monic and v is reduced, resulting in a reduced divisor $D_3 = D_1 + D_2 = (u_3, v_3)$. All steps for arbitrary characteristic including those for the precomputed quantities are given in Algorithm 3 [14].

Algorithm 3 Frequent Case for Group Addition ($g=2$)

Require: $D_1 = \text{div}(u_1, v_1)$, $D_2 = \text{div}(u_2, v_2)$

Ensure: $D_3 = \text{div}(u_3, v_3) = D_1 + D_2$

1: $k = \frac{f - v_1 h - v_1^2}{u_1}$ (exact division)

2: $s \equiv \frac{v_2 - v_1}{u_1} \pmod{u_2}$

3: $z = s u_1$

4: $u' = \frac{k - s(z + h + 2v_1)}{u_2}$ (exact division)

5: $u_3 = u'$ made monic

6: $v_3 \equiv -(h + z + v_1) \pmod{u_3}$

For genus-3 curves an additional reduction step is necessary to calculate the reduced divisor $D_4 = \text{div}(u_4, v_4) = D_1 + D_2$:

$$6: \quad u_4 = \frac{f - v_3 h - (v_3)^2}{u_3} \quad (\text{exact division})$$

$$7: \quad v_4 \equiv -(v_3 + h) \pmod{u_4}$$

Based on Algorithms 2 and 3, the upcoming explicit formulae for the group doubling on hyperelliptic curves of genera 2 and 3 are derived.

D. Security of HECC

The Diffie-Hellman problem on $\mathbb{J}_C(\mathbb{F})$ is closely related to the well-studied discrete logarithm problem [40]. These problems are significant to public-key cryptography because they form the basis for the security of many cryptographic schemes. The DLP on $\mathbb{J}_C(\mathbb{F})$ can be stated as follows: given two divisors $D_1, D_2 \in \mathbb{J}_C(\mathbb{F})$, determine the smallest integer m such that $D_2 = mD_1$, if such an m exists. The binary algorithm and its variants [40], [41] can be used to efficiently compute mD . The main operations in the algorithm are group additions and group doublings.

Pollard's rho method and its variants [42]–[44] are the most important examples for algorithms solving the DLP with complexity $O(\sqrt{n})$ in groups of order n . However, some special cases of HEC were discovered in [45], [46], which can be attacked with lower complexity than $O(\sqrt{n})$. The first algorithm which computes the DL in subexponential time for sufficiently large genera was published in [47]. The algorithm was improved and implemented, e.g. in [48]–[51]. This algorithm has a lower complexity than Pollard's rho method for $g > 4$.

In [45], the authors described the mapping of the Tate pairing on the divisor class group of a curve C over a finite field \mathbb{F}_q into the multiplicative group $\mathbb{F}_{q^k}^*$. Hence, for small k , the DLP in the divisor class group can be solved with the index-calculus algorithms. In [50] it is shown that index-calculus algorithms in the Jacobian of HEC have a lower complexity than the Pollard rho method for curves of genus greater than 4. However, the author further studied special cases by keeping only a fraction of the divisors in the factor base resulting in a reduction of the base by a factor of n . In this case, the author obtained an overall complexity of $O(q^{\frac{2g}{g+1}})$. Thus, the complexity of the attack of genus-4 curves is lower than the complexity of the rho method. However, no practical comparisons between the two approaches have been made.

Recently, Thériault optimized the algorithm to compute the discrete logarithm in the Jacobian of low genus hyperelliptic curves [52]. He optimized the sub-exponential algorithm to compute the discrete logarithm in the Jacobian of low genus hyperelliptic curves. The underlying field for HEC with genus higher than two might have to be larger than believed in order to achieve a certain security level. Thus, when taking into account the results of Thériault, the underlying field has to be enlarged accordingly, see the correction factor in Table I. For the moment, the most efficient attacks presented in [52] are not practical, because of the high storage usage. Note, we took this security consideration into account and enlarge the field sizes accordingly, see comparison of different implementations in Section V-B.

In order to find secure HECC one has to consider criteria to ensure that a curve is not supersingular [53]. However, there are no hyperelliptic supersingular curves of genus $2^n - 1$ over fields of characteristic 2 for any integer $n \geq 2$ [54].

For our improvements we assume for example genus-2 curves of the form $y^2 + xy = x^5 + f_1x + f_0$ where $f_0, f_1 \in \mathbb{F}_{2^n}$. Similar curves were chosen in the case of genus-3 curves. Obviously, they form a special class of curves, also known as

TABLE I

COMPARISON OF THE COMPLEXITY OF THE RUNNING TIME OF DIFFERENT ATTACKS ON HECC

g	1	2	3	4
Index	q^2	q^2	q^2	q^2
Rho [50]	$q^{1/2}$	q	$q^{3/2}$	q^2
reduced factor base [52]	n.a.	n.a.	$q^{3/2}$	$q^{8/5}$
with large primes [52]	n.a.	n.a.	$q^{10/7}$	$q^{14/9}$
Correction factor for $\log_2 \mathbb{F}_q $	-	-	1.05	1.286

non-ordinary curves [55], but the free choice of f_0 and f_1 still leads to sufficiently many. We are not aware of any security limitation of the curves that we used in this publication.

There is a contribution that dealt with securing practical implementations [56]. In this paper, countermeasures against differential power analysis for HECC were introduced by modeling two techniques used for elliptic curves cryptosystems.

In addition, one should consider the Weil decent attack methodology [57] and especially the GHS Weil decent attack [58]. Consider E to be a non-supersingular elliptic curve defined over a field $K = \mathbb{F}_{2^m}$, and m is composite. The idea of the attack is to reduce the ECDLP in $E(\mathbb{F}_{2^m})$ to the DLP in the jacobian variety of a curve of larger genus defined over a proper subfield $k = \mathbb{F}_l$ of K . Concluding from the above mentioned publications, using fields with composite extension degree can have cryptographic weaknesses which can potentially lead to attacks.

By our choices of fields we tried to get a comparison in group size of ECC and HECC as tight as possible, therefore some of the fields we used are based on m composite. However, all implementation techniques *do not use* the composite field structure, unlike the work in [59], [60]. Hence, all the software implementations should be viewed as example cases to show the efficiency of the different systems and are also applicable to HECC based on prime extension fields.

IV. METHODS TO IMPROVE THE EXPLICIT FORMULAE

In this section, we list all techniques used to improve the efficiency of the explicit formulae and therefore increase the performance of HECC group operation.

A. Montgomery's Trick of Simultaneous Inversions

The idea of Montgomery is to use simultaneous inversions in order to save inversions at the cost of some cheaper operations, e.g., multiplications [61, Algorithm 10.3.4].

WLOG we consider the group addition on genus-2 HECC to illustrate its application. Harley's Algorithm needs two field inversions, see Equation (3), Steps 2 and 5. These steps can be modified. Instead of computing $s(x) = s_1x + s_0 \equiv \frac{v_2 - v_1}{u_1} \pmod{u_2}$, one first computes the resultant r of u_1 and u_2 as: $inv = \frac{r}{u_1} \pmod{u_2}$ (no field inversion needed) and $s'(x) = rs \equiv (v_2 - v_1) \cdot inv \pmod{u_2}$.

In the latter step only one inversion is necessary to obtain both r^{-1} and s_1^{-1} . The variable s_1^{-1} is required in Equation (3)

to make u' monic and r^{-1} is required to calculate $s(x) = s'r^{-1}$. First, one has to compute the inverse $w_1 = (rs'_1)^{-1}$, then s and r^{-1} can be calculated as: $r^{-1} = w_1s'_1$ and $s(x) = s_1x + s_0 = \frac{s'}{r} = \frac{s'_1}{r}x + \frac{s'_0}{r}$. s_1^{-1} is then obtained by $s_1^{-1} = w_1r^2 = \frac{r}{s'_1}$.

B. Reordering of the Normalization Step

This reordering allows for calculating the required monic polynomial u' (Equation (2), Step 4, and Equation (3), Step 5) while saving field operations [26]. The highest order term of u' results from the product of s and z divided by u_2 . Since u_2 is monic, the leading coefficient of u' is s_1^2 . Thus, the step of making u' monic is unnecessary if the polynomial s is already monic (i.e. $s_1 = 1$). In [26], the author shows that this simplification saves one multiplication in total for the case of genus 2. In this contribution, we show that in the case of genus-3 curves one can save even more multiplications. In the following we show, that u' is already monic:

$$\begin{aligned} \text{let } w_4 &= \frac{1}{s_1} \quad \text{and} \quad w_5 = w_4^2 \\ u' &= \frac{1}{u_2} \left[s_{\text{monic}}(s_{\text{monic}}u_1 + w_4(h + 2v_1)) \right. \\ &\quad \left. - w_5 \frac{f - v_1h - v_1^2}{u_1} \right] \\ &= \frac{1}{u_2} \left[sw_4(sw_4u_1 + w_4(h + 2v_1)) \right. \\ &\quad \left. - w_5 \frac{f - v_1h - v_1^2}{u_1} \right] \quad \text{with } s_{\text{monic}} = sw_4 \\ &= \frac{w_4^2}{u_2} \left[s(su_1 + h + 2v_1) - \frac{f - v_1h - v_1^2}{u_1} \right] \\ &= -w_4^2u = -\frac{u}{s_1^2} \end{aligned}$$

Using the reordering, the calculation of v' changes as follows:

$$\begin{aligned} \text{with } w_3 = s_1, \quad v' &= -(w_3s_{\text{monic}}u_1 + h + v_1) \bmod u' \\ &= -(w_3w_4su_1 + h + v_1) \bmod u' \\ &= -(su_1 + h + v_1) \bmod u' \end{aligned}$$

C. Karatsuba Multiplication

In 1962, Karatsuba introduced an algorithm to multiply two polynomials [34]. Compared to the schoolbook method, the Karatsuba Algorithm (KA) saves multiplications of the coefficients at the cost of extra additions. Since its introduction, further work was done to improve the KA and to find bounds of the complexity [62]–[64]. In [65], Bernstein presented a survey of different methods to multiply polynomials. In [66], detailed information on the usage of KA in order to multiply with the least cost is provided.

In [34], the efficient multiplication of two polynomials of degree 1 is introduced. We will briefly develop the KA and refer the interested reader to the given literature. Given are two polynomials $A(x) = a_1x + a_0$ and $B(x) = b_1x + b_0$. Let $D_0 = a_0b_0$, $D_1 = a_1b_1$ and $D_{0,1} = (a_0 + a_1)(b_0 + b_1)$. Hence, the product of two polynomials can be computed as: $C(x) = D_1x^2 + (D_{0,1} - D_0 - D_1)x + D_0$.

The total cost is four additions and three multiplications. Whereas, if using the schoolbook method, four multiplications and one addition are needed. Thus, we save one multiplication at the cost of three extra additions.

D. Efficient Division

In [67], an efficient way to calculate the quotient of two polynomials is presented. The division is based on the observation that the quotient of two polynomials of degree deg_1 and deg_2 , with $deg_1 > deg_2$, depends only on the $deg_1 - deg_2 + 1$ highest coefficients of the dividend and the $deg_1 - deg_2 + 1$ highest coefficients of the divisor. Hence, we do not have to consider all coefficients of the polynomials.

This efficient division is, for example, used in Step 8 in Table IX. One has to compute $u_3 = (f - v'h - v'^2)/u'$ with $deg(f - v'h - v'^2) = 7$ and $deg(u') = 4$. Thus, only the four highest coefficients of the two polynomials are needed to compute u_3 (Notice, that $f_7 = u'_4 = 1$ and therefore does not appear in the formula as a variable).

E. Calculation of the Resultant Using Bezout's Matrix

The calculation of the resultant using Bezout's matrix in the case of genus-3 HEC can be performed very efficiently. Notice, that there is no benefit when applying the Bezout's matrix to genus-2 HEC.

The resultant of two polynomials $a = \prod_{i=1}^n (x - \alpha_i)$ and $b = \prod_{j=1}^m (x - \beta_j)$ is defined by $r(a, b) = \prod_{i=1}^n \prod_{j=1}^m (\beta_j - \alpha_i)$. Let $a = x^3 + ax^2 + bx + c$ and $b = x^3 + dx^2 + ex + f$, then the resultant calculated with the Bezout's matrix is given by

$$\begin{aligned} r(a, b) &= (f + ea - c - bd)[(-c + f)^2 \\ &\quad - (-a + d)(fb - ce)] + (fa - cd) \\ &\quad [(fa - cd)(-a + d) - 2(-b + e)(-c + f)] + \\ &\quad (fb - ce)(-b + e)^2. \end{aligned}$$

In general, the total cost of this operation is 12 multiplications and 2 squarings. For two input polynomials in the case of doubling, the resultant is less complex and can be computed with 6 multiplications and 2 squarings. For more details, see Table IX and Table X.

F. Choice of HEC with Certain Properties

A detailed analysis of the explicit formulae gives rise to certain types of curves with low complexity, i.e. optimum performance regarding the number of required field operations for the execution of the group operations. As a result of this analysis, curves of the form $y^2 + y = f(x)$ over extension fields of characteristic two turn out to be the best option. Unfortunately, this type of curve is supersingular for genus 2 [50], [53]. To our knowledge, curves of this form for genus 3 have no security limitations [54]. Hence, curves of the form $y^2 + h(x)y = f(x)$, with $h(x) = x$ give the best performance for genus-2. In [36], the author suggested some different transformations such that for the case of genus-2 curves, one can neglect most of the coefficients in the f polynomial. We

$$\begin{aligned}
B(x) &= \sum_{l=\frac{d+1}{2}}^{\frac{m+d-2}{2}} \{[a_{2l} - T_0 p_{2l-d} - T_1 p_{2l-d+1}]x + [a_{2l-1} - T_0 p_{2l-1-d} - T_1 p_{2l-d}]\} x^{2l-1} + (a_{d-1} - T_1 p_0) x^{d-1} \\
&+ \sum_{i=0}^{d-2} a_i x^i \tag{6} \\
&= \sum_{l=\frac{d+1}{2}}^{\frac{m+d-2}{2}} \{[a_{2l} - (T_0 + T_1)(p_{2l-d} + p_{2l-d+1}) + T_0 p_{2l-d+1} + T_1 p_{2l-d}]\} x^{2l} + [a_{2l-1} - T_0 p_{2l-1-d} - T_1 p_{2l-d}]\} x^{2l-1} \\
&+ (a_{d-1} - T_1 p_0) x^{d-1} + \sum_{i=0}^{d-2} a_i x^i \tag{7}
\end{aligned}$$

did integrate this technique to speed-up our implementation of HECC. However, all explicit formulae presented in the appendix are for the most general case.

G. Karatsuba Reduction

In our contribution, we used the idea of the algorithm presented by Karatsuba [34] to minimize the computational complexity of polynomial modulo reduction, denoted as Karatsuba reduction. In the context of HECC, this idea was first used for genus-2 curves targeting small polynomials in [29]. All published contribution improving HECC group operation applied this technique in an ‘‘ad hoc’’. In this work, we approached the reduction systematically and generalized the procedure for polynomials of arbitrary degree. The results show, that one can perform polynomial reduction using Karatsuba’s method with the complexity $O(n^{1.58})$. Note that the presented formulas can also be applied if working in finite fields $\text{GF}(p^m)$.

Let polynomial $A(x) = \sum_{i=0}^{m+d} a_i x^i$ be reduced by the polynomial $P(x) = x^m + p(x) = x^m + \sum_{i=0}^{m-1} p_i x^i$ of degree m . Recursive application of $x^m = -p(x) = \sum_{i=0}^{m-1} (-p_i) x^i$ describes a simple way to perform modulo reduction whenever encountering powers of x greater than $m - 1$.

Let

$$\begin{aligned}
B(x) &= A(x) \bmod P(x) \\
&= \sum_{i=0}^{m+d} a_i x^i \bmod x^m + \sum_{i=0}^{m-1} p_i x^i \tag{8}
\end{aligned}$$

with integer $d \geq 1$.

The reduction consists of two substitution steps, that are repeatedly computed $d/2$ or $(d-1)/2$ times for d even or odd, respectively. In the following we describe this step in more detail and summarize them in Algorithm 4.

STEP 1: In the first step, we substitute $x^{m+d} = -x^d \sum_{i=0}^{m-1} p_i x^i$ and the highest coefficient $T_0 = a_{m+d}$ in Equation (8):

$$\begin{aligned}
B(x) &= (a_{m+d-1} - T_0 p_{m-1}) x^{m+d-1} \\
&+ \sum_{i=-d}^{m+d-2} (a_i - T_0 p_{i-d}) x^i + \sum_{i=0}^{d-1} a_i x^i
\end{aligned}$$

STEP 2: In the second step, we substitute $x^{m+d-1} = -x^{d-1} \sum_{i=0}^{m-1} p_i x^i$ and the highest coefficient for simplification $T_1 = (a_{m+d-1} - a_{m+d} p_{m-1})$:

$$\begin{aligned}
B(x) &= \sum_{i=d}^{m+d-2} (a_i - T_0 p_{i-d} - T_1 p_{i-d+1}) x^i \\
&+ (a_{d-1} - T_1 p_0) x^{d-1} + \sum_{i=0}^{d-2} a_i x^i \tag{9}
\end{aligned}$$

Next, we assume WLOG m and d to be odd. We rewrite Equation (9) in order to be able to apply Karatsuba reduction on Equation (6). The result is given in Equation (7). After the second step, we have a polynomial $B(x)$ of degree $m+d-2$, thus, we are able to reduce the degree of the polynomial by 2. Step 1 and Step 2 have to be repeated $d/2$ or $(d+1)/2$ times in order to reduce $A(x)$ for d odd or even, respectively. After applying Step 1 and 2 we set $A(x)_i = B(x)_{i-1}$.

Corollary 1: Let $\#M$ and $\#A$ be the number of multiplications and additions, respectively, to reduce a polynomial with degree $m+d$, where d is a positive integer and where m is the degree of the reduction polynomial. Thus, we need to perform $d+1$ reduction steps in order to replace all terms with occurrences of x^k where $k \geq m$. The total cost of using Karatsuba reduction is shown in Equation 10.

In the case for d even, we have to add an additional step after applying Karatsuba. This last step reduces the polynomial with degree m , by substituting $x^m = \sum_{i=0}^{m-1} (-p_i) x^i$. Hence, we need m additional multiplications and m additions. Applying Karatsuba reduction to the additional step does not result in a more efficient way to reduce the polynomial of degree m . If we use the schoolbook method instead, we need m multiplications and m additions for each reduction step, thus $m(d+1)[M+A]$ in total.

In the above stated consideration we looked only into the cases where the reduction polynomials have all coefficients non-zero. In most cases, reduction polynomials will have only some non-zero coefficients. In this case Corollary 2 should be considered.

Corollary 2: Let $\#M$ and $\#A$ be the number of multiplications and additions, respectively, to reduce a polynomial $A(x)$ with degree $m+d$, where d is a positive integer. The reduction polynomial is given by $P(x) = x^m + x^{t_1} + x^{t_2} + x^{t_3} + \dots + x^{t_i}$,

$$Cost \begin{cases} \lfloor \frac{d+1}{2} \rfloor \left[\lceil 3/2m \rceil M + (8m - 3\lceil 3/2m \rceil)A \right] & \text{for odd } d \\ \lfloor \frac{d+1}{2} \rfloor \left[\lceil 3/2m \rceil M + (8m - 3\lceil 3/2m \rceil)A \right] + mM + mA & \text{for even } d \end{cases} \quad (10)$$

$$Cost \begin{cases} \lfloor \frac{d+1}{2} \rfloor \left[\lceil 3/2i \rceil M + (8i - 3\lceil 3/2i \rceil)A \right] & \text{for odd } d \\ \lfloor \frac{d+1}{2} \rfloor \left[\lceil 3/2i \rceil M + (8i - 3\lceil 3/2i \rceil)A \right] + iM + iA & \text{for even } d \end{cases} \quad (11)$$

where $i + 1$ is the number of non-zero coefficients. Hence, the cost to reduce the polynomial $A(x)$ is given in Equation 11.

Corollary 3: In case of schoolbook or Karatsuba reduction, a polynomial with the smallest possible number of non-zero coefficients results in a minimum of computational cost.

V. RESULTS

This section summarizes our results and is divided in two parts. In the first part we present our derived explicit formulae. The second part introduces and analyzes our implementation of the formulae.

A. Improving the Group Operation

After Koblitz suggested HEC for the use for a cryptosystem in 1989, it took over 10 years until the first improvements were made to its group operations. In [1], the authors proposed explicit formulae for the group operations. For the derivation of the explicit formulae, polynomial calculations are mapped onto field operations. For genus-2 HEC, such a mapping can be calculated quite easily since the degree of the occurring polynomials is low. Deriving explicit formulae for higher genus HEC is more complicated since with increasing genus the degree of the polynomials rises. For higher genera, the problem becomes intractable by hand and the use of additional tools for the mathematical evaluation of all equations is inevitable.

Note, that the explicit formulae given in the appendix are displayed in the most general form. However, the complexity is computed for special curves and therefore some operations are not counted, e.g. multiplication with $h_i \in GF(2)$. In addition, we reuse in some steps previously computed results in order to decrease the complexity.

1) *Improving Cantor's Group Operation:* In this subsection, we considered the group operations presented in [2] and applied the techniques introduced in Chapter IV, resulting in efficient group operation for genus-2 and genus-3 HECC. Table IV summarizes the efforts made to date and our contribution to speed up Cantor's algorithm. The detailed instruction, how to calculate the group operations are given in the appendix.

Our improvements concerning the group operations introduced by Cantor can be summarized as following: This is the first contribution that presents explicit formulae of Cantor's algorithm for genus-2 and 3. We were able to speed up the group operations compared to the results presented in [33]. Our genus-2 HEC group operation saves up to $6M/S$ in the case of adding divisors and $1I$ as well as up to $20M/S$ in the case of doubling a divisor. Using our formulae for

certain genus-3 curves one can save 24% and 78% of the computational cost for adding and doubling, respectively².

2) *Improving Harley's Group Operation:* The approach presented in [1] was also optimized by us using the techniques in Chapter IV. We were able to speed up the group operation of genus-2 and genus-3 HECC. Table V presents a summary of our work, as well as of all the previous work that has been done on improving Harley's algorithm. All the steps necessary to compute the group operations using the explicit formulae are presented in the appendix.

Analyzing our newly derived the group operations based on Harley's considerations lead to following conclusions:

- We could improve the explicit formulae for arbitrary characteristic for genus-3 hyperelliptic curves compared to the previous publications [29].
- For characteristic two fields, we could decrease the costs to calculate the group operations on genus-2 and genus-3 curves.
- Considering certain curve parameters we were able to save 47% of the multiplications for the genus-2 doubling operation compared to [14], [36].
- Our fastest genus-3 curves used $42M/S$ less for the group doubling as presented in [15].

3) *Fastest HECC Group Operation:* The aim of this subsection is to give a summary of the last two subsections, by extracting only our fastest HECC group operations. In the sections above we give a historical survey of the group operations based on the original Cantor algorithm and on the algorithm proposed by Harley. One notices that using specialized the curve parameters the efficiency of the cryptosystem increases.

The bold numbers in Table IV and V indicate our fastest group operations. Contrary to common belief, we could show that explicit formulae for the group doubling based on Cantor's algorithm in the case of genus-3 HEC are faster than the explicit formulae based on Harley. In most of the other cases, the group operations based on Harley's algorithm perform better. However, it is noticeable that the performance of the doubling operation for certain curves based on Harley or Cantor are almost the same.

Following explicit formulae for an efficient implementation of HECC should be used:

<i>genus</i>	<i>addition</i>	<i>doubling</i>
2	[14]	<i>Table VI</i>
3	<i>Table IX</i>	<i>Table XIII</i>

At this point we want to clarify that we applied a sequence of improvements techniques to achieve the presented results.

²assuming one inversion has the same time complexity as eight multiplications

Algorithm 4 Karatsuba reduction**Require:**

Polynomial $P(x) = x^m + \sum_{i=0}^{m-1} p_i x^i$ of degree m ,

Polynomial $B^{(d)}(x) = \sum_{i=0}^{m+d} b_i^{(d)} x^i$ of degree $m+d$

Ensure: $B(x) = B^{(d)}(x) \bmod P(x) = \sum_{i=0}^{m-1} b_i x^i$

```

1: while  $d > 0$  do
2:    $T_0 = b_{m+d}^{(d)}$ 
3:    $T_1 = b_{m+d-1}^{(d)} - T_0 p_{m-1}$ 
4:    $b_{d-1}^{(d-2)} = b_{d-1}^{(d)} - T_1 p_0$ 
5:   if  $d$  odd then
6:     for  $l = \lceil \frac{d+1}{2} \rceil$  to  $\lceil \frac{m+d-2}{2} \rceil$  do
7:        $b_{2l}^{(d-2)} = b_{2l}^{(d)} - (T_0 + T_1)(p_{2l-d} + p_{2l-d+1}) +$ 
          $T_0 p_{2l-d+1} + T_1 p_{2l-d}$ 
8:        $b_{2l-1}^{(d-2)} = b_{2l-1}^{(d)} - T_0 p_{2l-d-1} - T_1 p_{2l-d}$ 
9:     end for
10:  else
11:    for  $l = \lceil \frac{d+1}{2} \rceil$  to  $\lceil \frac{m+d-2}{2} \rceil$  do
12:       $b_{2l-1}^{(d-2)} = b_{2l-1}^{(d)} - (T_0 + T_1)(p_{2l-d-1} + p_{2l-d}) +$ 
         $T_0 p_{2l-d} + T_1 p_{2l-d-1}$ 
13:       $b_{2l-2}^{(d-2)} = b_{2l-2}^{(d)} - T_0 p_{2l-d-2} - T_1 p_{2l-d-1}$ 
14:    end for
15:  end if
16:  if  $m$  even then
17:     $b_{m+d-2}^{(d-2)} = b_{m+d-2}^{(d)} - (T_0 + T_1)(p_{m-1} + p_{m-2}) +$ 
       $T_1 p_{m-2} + T_0 p_{m-1}$ 
18:  end if
19:  for  $j = 0$  to  $d-2$  do
20:     $b_j^{(d-2)} = b_j^{(d)}$ 
21:  end for
22:   $d = d - 2$ 
23: end while
24: if  $d = 0$  then
25:   for  $j = 0$  to  $m-1$  do
26:     $b_j^{(-1)} = b_j^{(d)} - b_m^{(d)} \cdot p_j$ 
27:   end for
28: end if
29: Output  $B(x) = \sum_{j=0}^{m-1} b_j x^j = B^{(-1)}(x)$ 

```

This is somewhat of an ad-hoc approach which does not necessarily lead to optimal results. Rather, the complexity of the explicit formulae presented in this chapter should be viewed as upper bound. New improvements of genus-2 HEC were presented in [68] and is published [69].

B. Implementation

This chapter provides a short overview of the tools and the methodology used. We also present and analyze our implementation results.

1) *Methodology:* The methodology is as follows:

- Test the explicit formulae: An NTL-based [70] implementation of the original Cantor algorithm was used to obtain reliable reference vectors. The implementation of the new explicit formulae was used to check the correctness of the derived formulae. *Microsoft Developer Studio 6* was used for compilation and debugging on a Pentium 4 processor.

- Speeding up the implementation: A fast library for the required field and group operations was developed.
- The code was first tested on a Pentium platform.
- Porting to the ARM: The code was rewritten for the ARM7TDMI@80MHz (ARMulator). Special functions to load the test vectors into the memory of the micro-processor and timing routines were written. As a programming and debugging platform, the *ARM Developer Suite 1.2* was used.
- Running and testing HECC on the ARM7TDMI (ARMulator).
- Detailed timing analysis for different field sizes and curves.

2) *Implementation:* We implemented our explicit formulae for characteristic two fields as summarized in Section V-A.3. In order to be able to compare our results with an elliptic curve cryptosystem (ECC), we implemented ECC with the same methodology. For genus-2 curves we implemented the explicit formula based on Harley's algorithm and $h(x) = x$. In the case of genus-3 HEC, we used the doubling formula based on Cantor Table XIII and the addition formula based on Harley Table IX and $h(x) = 1$. The implementation results on the Pentium and the ARM are summarized in the Table II and Table III, respectively.

Our implementation assists arbitrary fields. Note that the group operations as well as the scalar multiplication could be implemented with higher performance, if fixing the underlying field, e.g. see [71]. Note, that some of the implementations might have potential cryptographical weaknesses according to [57], [58], however, we would like to emphasize that the implementation techniques are also applicable to fields with prime extensions.

3) *Analyzing and Comparing the ECC and HECC Implementation:* It is only useful to compare cryptosystems with the same security level. Hence, we have to take the recent progress by Thériault [52] into consideration. Note, that this is a very pessimistic estimation, since the attacks presented can not be realized because of the high complexity. The results in [52] suggest to enlarge the underlying fields of genus-3 curves (see Table I for more details).

According to [52] an identical security level of 163bit results in the underlying fields $\mathbb{F}_{2^{81}}$ and $\mathbb{F}_{2^{57}}$ for HEC of genus two and three, respectively.

Contrary to common belief, we could show, that the performance of a scalar multiplication of ECC, genus-2 HECC, and genus-3 HECC are in the same range. This fact holds for the Pentium as well as for the ARM implementation. In some cases HECC even outperforms ECC, see Table II and III.

Hyperelliptic curve cryptosystems show very good performance on embedded processors and are therefore well suited for applications in constrained environments.

It is noticeable that with increasing group order the speed of the group operations decreases for certain field sizes. The reason is the dependency of the field operations on the irreducible polynomials. We used trinomials and pentanomials, whereas the groups using trinomials perform relatively better compared with pentanomials.

TABLE II
TIMINGS OF GROUP OPERATIONS ON THE PENTIUM4@1.8GHZ (EXPLICIT FORMULAE)

Genus	Field	Group order	Group addition in μs	Group doubling in μs	Scalar. mult. in ms
1	$\mathbb{F}_{2^{163}}$	2^{163}	18.3	9.4	2.60
	$\mathbb{F}_{2^{167}}$	2^{167}	19.2	8.5	2.43
	$\mathbb{F}_{2^{179}}$	2^{179}	16.9	9.8	2.80
	$\mathbb{F}_{2^{191}}$	2^{191}	15.4	8.7	2.78
2	$\mathbb{F}_{2^{63}}$	2^{126}	16.5	10.2	1.93
	$\mathbb{F}_{2^{81}}$	2^{162}	18.7	11.7	2.73
	$\mathbb{F}_{2^{83}}$	2^{166}	20.2	12.7	3.01
	$\mathbb{F}_{2^{88}}$	2^{176}	20.5	13.2	3.30
	$\mathbb{F}_{2^{91}}$	2^{182}	21.1	13.7	3.50
	$\mathbb{F}_{2^{95}}$	2^{190}	19.0	12.6	3.41
3	$\mathbb{F}_{2^{43}}$	2^{129}	25.5	9.0	2.15
	$\mathbb{F}_{2^{54}}$	2^{162}	24.8	8.9	2.56
	$\mathbb{F}_{2^{55}}$	2^{165}	25.2	9.0	2.69
	$\mathbb{F}_{2^{57}}$	2^{171}	25.5	9.4	2.95
	$\mathbb{F}_{2^{59}}$	2^{177}	28.5	10.3	3.22
	$\mathbb{F}_{2^{60}}$	2^{180}	24.8	9.2	2.97
	$\mathbb{F}_{2^{61}}$	2^{183}	28.5	10.5	3.34
	$\mathbb{F}_{2^{63}}$	2^{189}	25.3	9.2	3.10

TABLE III
TIMINGS OF GROUP OPERATIONS WITH ARMULATOR ARM7TDMI@80MHZ (EXPLICIT FORMULAE)

Genus	Field	Group order	Group addition in μs	Group doubling in μs	Scalar. mult. in ms
1	$\mathbb{F}_{2^{163}}$	2^{163}	746	406	108
	$\mathbb{F}_{2^{167}}$	2^{167}	579	342	90
	$\mathbb{F}_{2^{179}}$	2^{179}	720	451	120
	$\mathbb{F}_{2^{191}}$	2^{191}	598	358	100
2	$\mathbb{F}_{2^{63}}$	2^{126}	538	326	60
	$\mathbb{F}_{2^{81}}$	2^{162}	600	376	87
	$\mathbb{F}_{2^{83}}$	2^{166}	715	449	105
	$\mathbb{F}_{2^{88}}$	2^{176}	732	463	114
	$\mathbb{F}_{2^{91}}$	2^{182}	736	468	119
	$\mathbb{F}_{2^{95}}$	2^{190}	623	399	107
3	$\mathbb{F}_{2^{43}}$	2^{129}	966	327	75
	$\mathbb{F}_{2^{54}}$	2^{162}	914	317	90
	$\mathbb{F}_{2^{55}}$	2^{165}	917	319	91
	$\mathbb{F}_{2^{57}}$	2^{171}	918	321	94
	$\mathbb{F}_{2^{59}}$	2^{177}	1180	415	126
	$\mathbb{F}_{2^{60}}$	2^{180}	921	324	100
	$\mathbb{F}_{2^{61}}$	2^{183}	1183	417	130
	$\mathbb{F}_{2^{63}}$	2^{189}	925	329	106

VI. CONCLUSIONS

Our contribution is another step to decrease the complexity of the HEC group operations. We showed how far the choice of the algorithm (Harley or Cantor), group order, and curve parameters is crucial in order to achieve the best performance for a given application.

Considering the results presented in this contribution, ex-

PLICIT formulae based on Harley's algorithm should be used when implementing genus-2 curves. In the case of genus-3 curves the doubling operation should be implemented using explicit formulae based on Cantor and for the addition we advise the use of Harley's algorithm.

Considering different group orders, our implementation shows that ECC, genus-2, and genus-3 HECC have similar performance. Contrary to common belief HECC can be the

cryptosystem of choice for general purpose processors as well as for embedded processors.

REFERENCES

- [1] 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, 2000, pp. 297 – 312.
- [2] D. Cantor, "Computing in Jacobian of a Hyperelliptic Curve," in *Mathematics of Computation*, vol. 48(177), January 1987, pp. 95 – 101.
- [3] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, pp. 644–654, 1976.
- [4] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, February 1978.
- [5] N. Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.
- [6] V. Miller, "Uses of elliptic curves in cryptography," in *Advances in Cryptology — CRYPTO '85*, ser. LNCS 218, H. C. Williams, Ed. Berlin, Germany: Springer-Verlag, 1986, pp. 417–426.
- [7] N. Koblitz, "A Family of Jacobians Suitable for Discrete Log Cryptosystems," in *Advances in Cryptology - Crypto '88*, ser. LNCS 403, Shafi Goldwasser, Ed. Berlin, Germany: Springer-Verlag, 1988, pp. 94 – 99.
- [8] U. Krieger, "signature.c," Master's thesis, Mathematik und Informatik, Universität Essen, Fachbereich 6, Essen, Germany, February 1997.
- [9] Y. Sakai and K. Sakurai, "Design of Hyperelliptic Cryptosystems in small Characteristic and a Software Implementation over \mathbb{F}_{2^n} ," in *Advances in Cryptology - ASIACRYPT '98*, ser. LNCS 1514, K. Ohta and D. Pei, Eds. Berlin, Germany: Springer Verlag, 1998, pp. 80 – 94.
- [10] Y. Sakai, K. Sakurai, and H. Ishizuka, "Secure Hyperelliptic Cryptosystems and their Performance," in *Public Key Cryptography: First International Workshop on Practice and Theory in Public Key Cryptography — PKC'98*, ser. LNCS 1431, H. Imai and Y. Zheng, Eds. Berlin, Germany: Springer-Verlag, 1998, pp. 164 – 181.
- [11] N. Smart, "On the Performance of Hyperelliptic Cryptosystems," in *Advances in Cryptology — EUROCRYPT '99*, ser. LNCS 1592, J. Stern, Ed. Berlin, Germany: Springer-Verlag, 1999, pp. 165–175.
- [12] Y. Sakai and K. Sakurai, "On the Practical Performance of Hyperelliptic Curve Cryptosystems in Software Implementation," in *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E83-A NO.4, April 2000, pp. 692 – 703.
- [13] J. Pelzl, "Hyperelliptic Cryptosystems on Embedded Microprocessor," Master's thesis, Department of Electrical Engineering and Information Sciences, Ruhr-Universitaet Bochum, Bochum, Germany, September 2002.
- [14] T. Lange, "Efficient Arithmetic on Genus 2 Hyperelliptic Curves over Finite Fields via Explicit Formulae," Cryptology ePrint Archive, Report 2002/121, 2002, <http://eprint.iacr.org/>.
- [15] M. Goda, K. Matsuo, K. Aoki, J. Chao, and S. Tsujii, "Improvements of Addition Algorithm on Genus 3 Hyperelliptic Curves and their Implementations," in *The 2004 Symposium on Cryptography and Information Security, Japan — SCIS 2004*, January 2004.
- [16] R. M. Avanzi, "Aspects of Hyperelliptic Curves over Large Prime Fields in Software Implementations," in *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2004*, ser. LNCS 3156, M. Joye and J.-J. Quisquater, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 148 – 162.
- [17] T. Wollinger, "Computer Architectures for Cryptosystems Based on Hyperelliptic Curves," Master's thesis, ECE Department, Worcester Polytechnic Institute, Worcester, Massachusetts, USA, May 2001.
- [18] T. Wollinger and C. Paar, "Hardware Architectures Proposed for Cryptosystems Based on Hyperelliptic Curves," in *Proceedings of the 9th IEEE International Conference on Electronics, Circuits and Systems - ICECS 2002*, vol. III, September 2002, pp. 1159 – 1163.
- [19] N. Boston, T. Clancy, Y. Liow, and J. Webster, "Genus Two Hyperelliptic Curve Coprocessor," in *Cryptographic Hardware and Embedded Systems — CHES 2002*, ser. LNCS 2523, B. S. Kaliski, Ç. K. Koç, and C. Paar, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 529–539, updated version available at <http://www.cs.umd.edu/~clancy/docs/hec-ches2002.pdf>.
- [20] T. Clancy, "Analysis of FPGA-based Hyperelliptic Curve Cryptosystems," Master's thesis, University of Illinois Urbana-Champaign, December 2002.
- [21] 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.
- [22] H. Kim, T. Wollinger, Y. Choi, K. Chung, and C. Paar, "Hyperelliptic Curve Coprocessors on a FPGA," in *Workshop on Information Security Applications - WISA*, ser. LNCS. Berlin, Germany: Springer-Verlag, 2004.
- [23] K. Matsuo, J. Chao, and S. Tsujii, "Fast Genus Two Hyperelliptic Curve Cryptosystems," in *ISEC2001-31, IEICE*, 2001.
- [24] T. Lange, "Efficient Arithmetic on Hyperelliptic Curves," Ph.D. dissertation, Institute for Experimental Mathematics, University of Essen, Essen, Germany, 2001.
- [25] Y. Miyamoto, H. Doi, K. Matsuo, J. Chao, and S. Tsuji, "A Fast Addition Algorithm of Genus Two Hyperelliptic Curve," in *The 2002 Symposium on Cryptography and Information Security — SCIS 2002, IEICE Japan*, 2002, pp. 497 – 502, in Japanese.
- [26] M. Takahashi, "Improving Harley Algorithms for Jacobians of Genus 2 Hyperelliptic Curves," in *SCIS, IEICE Japan*, 2002, in Japanese.
- [27] T. Lange, "Inversion-Free Arithmetic on Genus 2 Hyperelliptic Curves," Cryptology ePrint Archive, Report 2002/147, 2002, <http://eprint.iacr.org>.
- [28] ———, "Weighted Coordinates on Genus 2 Hyperelliptic Curves," Cryptology ePrint Archive, Report 2002/153, 2002, <http://eprint.iacr.org>.
- [29] J. Kuroki, M. Gonda, K. Matsuo, J. Chao, and S. Tsujii, "Fast Genus Three Hyperelliptic Curve Cryptosystems," in *The 2002 Symposium on Cryptography and Information Security, Japan — SCIS 2002*, January 2002.
- [30] J. Pelzl, T. Wollinger, J. Guajardo, and C. Paar, "Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves," in *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2003*, ser. LNCS 2779, C. D. Walter, Ç. K. Koç, and C. Paar, Eds. Berlin, Germany: Springer-Verlag, September 2003, pp. 349 – 365.
- [31] J. Pelzl, T. Wollinger, and C. Paar, "High Performance Arithmetic for Special Hyperelliptic Curve Cryptosystems of Genus Two," in *International Conference on Information Technology: Coding and Computing - ITCC 2004*. IEEE Computer Society, April 2004.
- [32] D. Mumford, "Tata Lectures on Theta II," in *Prog. Math.*, vol. 43. Birkhäuser, 1984.
- [33] K. Nagao, "Improving Group Law Algorithms for Jacobians of Hyperelliptic Curves," in *ANTS IV*, ser. LNCS 1838, W. Bosma, Ed. Berlin, Germany: Springer-Verlag, 2000, pp. 439 – 448.
- [34] A. Karatsuba and Y. Ofman, "Multiplication of Multidigit Numbers on Automata," *Sov. Phys. Dokl. (English translation)*, vol. 7, no. 7, pp. 595–596, 1963.
- [35] H. 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, The institute of electronics, information and communication engineers, Tech. Rep., May 2002.
- [36] T. Lange, "Formulae for Arithmetic on Genus 2 Hyperelliptic Curves," J. AAEECC, September 2003.
- [37] R. Harley, "Fast Arithmetic on Genus Two Curves," <http://cristal.inria.fr/harley/hyper/>, 2000, adding.txt and doubling.c.
- [38] N. Koblitz, "Hyperelliptic Cryptosystems," *Journal of Cryptology*, vol. 1, no. 3, pp. 129–150, 1989.
- [39] ———, *Algebraic Aspects of Cryptography*, 1st ed. Berlin, Germany: Springer-Verlag, 1998.
- [40] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, Florida, USA: CRC Press, 1997.
- [41] D. M. Gordon, "A Survey of Fast Exponentiation Methods," *Journal of Algorithms*, vol. 27, pp. 129–146, 1998.
- [42] R. Gallant, R. Lambert, and S. Vanstone, "Improving the Parallelized Pollard Lambda Search on Binary Anomalous Curves," <http://www.certicom.com/chal/download/paper.ps>, 1998.
- [43] J. M. Pollard, "Monte Carlo methods for index computation mod p ," *Mathematics of Computation*, vol. 32, no. 143, pp. 918–924, July 1978.
- [44] D. H. Wiedemann, "Solving Sparse Linear Equations Over Finite Fields," *IEEE Transactions on Information Theory*, vol. IT-32, no. 1, pp. 54–62, January 1986.
- [45] G. Frey and H.-G. Rück, "A Remark Concerning m -Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves," *Mathematics of Computation*, vol. 62, no. 206, pp. 865–874, April 1994.
- [46] H.-G. Rück, "On the Discrete Logarithm in the Divisor Class Group of Curves," *Mathematics of Computation*, vol. 68, no. 226, pp. 805–806, 1999.

- [47] L. Adleman, J. DeMarrais, and M.-D. Huang, "A Subexponential Algorithm for Discrete Logarithms over the Rational Subgroup of the Jacobians of Large Genus Hyperelliptic Curves over Finite Fields," in *Algorithmic Number Theory, First International Symposium, ANTS-I*, ser. LNCS 877, L. Adleman and M.-D. Huang, Eds. Berlin, Germany: Springer-Verlag, May 1994, pp. 28 – 40.
- [48] R. Flassenberg and S. Paulus, "Sieving in Function Fields," <ftp://ftp.informatik.tu-darmstadt.de/pub/TI/TR/TI-97-13.rafla.ps.gz>, 1997, preprint.
- [49] P. Gaudry, "Algorithmique des Courbes Hyperelliptiques et Applications à la Cryptologie, PhD Thesis," Ph.D. dissertation, France, 2000.
- [50] —, "An Algorithm for Solving the Discrete Log Problem on Hyperelliptic Curves," in *Advances in Cryptology — EUROCRYPT 2000*, ser. LNCS 1807, B. Preneel, Ed. Berlin, Germany: Springer-Verlag, 2000, pp. 19–34.
- [51] A. Enge and P. Gaudry, "A General Framework for Subexponential Discrete Logarithm Algorithms," *Acta Arith.*, vol. 102, pp. 83 – 103, 2002.
- [52] N. Thériault, "Index Calculus Attack for Hyperelliptic Curves of Small Genus," in *Advances in Cryptology - ASIACRYPT '03*, ser. LNCS 2894, G. Goos, J. Hartmanis, and J. van Leeuwen, Eds. Berlin, Germany: Springer Verlag, 2003, pp. 79 – 92.
- [53] S. Galbraith, "Supersingular Curves in Cryptography," in *Advances in Cryptology - ASIACRYPT '03*, ser. LNCS 2248, C. Boyd, Ed. Berlin, Germany: Springer Verlag, 2001, pp. 495 – 517.
- [54] J. Scholten and J. Zhu, "Hyperelliptic Curves in Characteristic 2," *International Mathematics Research Notices*, vol. 2002, no. 17, pp. 905 – 917, 2002.
- [55] D. Subrao, "The p-rank of artin-schreier curves," *Manuscripta Math.* 16, pp. 169–193, 1975.
- [56] R. M. Avanzi, "Countermeasures against Differential Power Analysis for Hyperelliptic Curve Cryptosystems," in *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2003*, ser. LNCS 2779, C. D. Walter, Ç. K. Koç, and C. Paar, Eds. Berlin, Germany: Springer-Verlag, 2003, pp. 366 – 381.
- [57] G. Frey, "How to disguise an elliptic curve," Talk at ECC 1998, 1998, <http://cacr.math.uwaterloo.ca/conferences/1998/ecc98/slides.html>.
- [58] P. Gaudry, F. Hess, and N. P. Smart, "Constructive and destructive facets of Weil descent on elliptic curves," *Journal of Cryptology*, vol. 15, no. 1, pp. 19–46, 2002.
- [59] E. D. Win, A. Bosselaers, S. Vandenberghe, P. D. Gerssem, and J. Vandewalle, "A fast software implementation for arithmetic operations in $GF(2^n)$," in *Asiacrypt '96*, ser. LNCS 1233. Berlin, Germany: Springer-Verlag, 1996, pp. 65–76.
- [60] J. Guajardo and C. Paar, "Efficient Algorithms for Elliptic Curve Cryptosystems," in *Advances in Cryptology — CRYPTO '97*, ser. LNCS 1294, B. Kaliski, Ed. Berlin, Germany: Springer-Verlag, August 1997, pp. 342–356.
- [61] H. Cohen, *A Course in Computational Algebraic Number Theory*, ser. Graduate Texts in Math. 138. Berlin, Germany: Springer-Verlag, 1993, third corrected printing 1996.
- [62] D. E. Knuth, *The Art of Computer Programming. Volume 2: Seminumerical Algorithms*, 2nd ed. Reading, Massachusetts, USA: Addison-Wesley, 1981.
- [63] A. Lempel, G. Seroussi, and S. Winograd, "On the Complexity of Multiplication in Finite Fields," *Theoretical Computer Science*, vol. 22, pp. 285–296, 1983.
- [64] S. Winograd, "Some Bilinear Forms Whose Multiplicative Complexity Depends on the Field of Constants," *Mathematical Systems Theory*, vol. 10, pp. 169–180, 1977.
- [65] D. J. Bernstein, "Multidigit Multiplication for Mathematicians," *Advances in Applied Mathematics*, 2001, <http://cr.ypt.to/papers.html>.
- [66] A. Weimerskirch and C. Paar, "Generalizations of the Karatsuba Algorithm for Polynomial Multiplication," Ruhr-University Bochum, Germany, Tech. Rep., 2003, available at <http://www.crypto.rub.de/Publikationen/texte/kaweb.pdf>.
- [67] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*. The Edinburgh Building, Cambridge CB2 2RU, UK: Cambridge University Press, 1999.
- [68] M. Stevens and T. Lange, "Arithmetic on Hyperelliptic curves of genus 1 and 2," <http://www.crypto.rub.de/ge/seminar>, HGI Seminar, 2004.
- [69] —, "Efficient Doubling on Genus Two Curves over Binary Fields," in *11th Annual Workshop on Selected Areas in Cryptography*, ser. LNCS. Berlin, Germany: Springer-Verlag, August 2004.
- [70] V. Shoup, "NTL: A Library for Doing Number Theory (version 5.0c)," 2001, <http://www.shoup.net/ntl/index.html>.
- [71] 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)*, 2004, special Issue on Embedded Systems and Security.



Thomas Wollinger After he obtained his B.S. from the University of Dieburg, he worked at secunet Inc., the largest consulting firm in Germany, in the area of IT-Security. Between 1999 and 2001, he worked as a research assistant in the department of Electrical and Computer Engineering at Worcester Polytechnic Institute, MA, USA. In May 2001, Thomas Wollinger obtained a Master of Science from the WPI. In July 2004, he received his PhD with specialization in applied data security from the University of Bochum. Thomas Wollinger is currently working as

researcher at this university in the field of communication security. He was a Fulbright fellow (academic year 1999/2000) and he was awarded a corporate graduate fellowship (academic year 2000/2001) and Quadrille Ball Grant in the USA (academic year 2000/2001). His PhD was awarded with 'summa cum laude', the 'Gerd Massenbergs Preis' award for the best dissertation of all the engineering and natural science faculties of the University of Bochum (2004), and a fellowship for a book publication. Thomas Wollinger currently teaches introduction to data security and he organizes seminars in the same area. Mr. Wollinger also teaches industry courses covering topics like data security and cryptology, e.g. at Motorola labs, TÜV Rheinland, and GITS AG. His research interests range from fast software and hardware implementations of cryptographic algorithms to power analysis attacks against smart cards. He is also interested in cryptosystems that are particularly suited for embedded application, e.g. hyperelliptic curve cryptosystem, and their realizations on different platforms.



Jan Pelzl From 1994 to 1997, Jan Pelzl worked as an IT specialist for Bosch Telecom in Dortmund. In 2002, he obtained his Diploma from the Ruhr University of Bochum in Electrical Engineering with specialization in cryptology, digital communications, and digital signal processing. From 1997 on, he has been working as a research assistant and tutor at the Department of Electrical Engineering. Jan Pelzl was awarded a DAAD fellowship for the academic year 2000/2001 for graduate studies at the Purdue University, IN, USA. He obtained awards from the

"Prof. Koepfchen Foundation", the "Gerd und Ruth Massenbergs Foundation", and is a fellow of the Siemens Student Program. Currently, Mr. Pelzl is tutoring classes in the field of cryptology and data security and organizes seminars. His research interests include fast algorithms of asymmetric cryptographic algorithms especially suited for constrained environments as well as factorization methods on the basis of special purpose hardware.



Christof Paar Prof. Dr.-Ing. Christof Paar is head of the chair for communication security at the EURO-BITS center for IT-security in Bochum (Germany). From 1994 to 2001 he was professor at Worcester Polytechnic Institute (USA), where he headed the Cryptography and Information Security lab. Together with Çetin K. Koç he founded the CHES (Cryptographic Hardware and Embedded Systems) workshop series, which is one of the leading international forums for applied cryptography. Prof. Paar's research interests cover fast software- and hardware-

realizations of cryptographic algorithms, side-channel attacks and most other aspects of embedded security. He has over 60 peer-reviewed publications in applied cryptography, is editor of 6 conference proceedings and special issues, and holds several patents in his area.

TABLE IV
SPEEDING UP GROUP OPERATIONS OF CANTORS ALGORITHM.

genus		field charac.	curve properties	cost	
				addition	doubling
2	Cantor/Koblitz [33], [38]	general		$3I + 70M/S$	$3I + 76M/S$
	Nagao [33]	odd	$h(x) = 0, f_i \in \mathbb{F}_2$	$2I + 52M/S$	$2I + 49M/S$
	our work	general	$h_i \in \mathbb{F}_2, f_4 = 0$	$2I + 44M + 4S$	$2I + 42M + 8S$
		two	$h_i \in \mathbb{F}_2, f_4 = 0$	$2I + 42M + 4S$	$2I + 40M + 8S$
two		$h(x) = x, f_4 = 0$	$2I + 42M + 4S$ Table VII	$I + 23M + 6S$ Table VIII	
3	Cantor/Koblitz [33], [38]	general		$4I + 200M/S$	$4I + 207M/S$
	Nagao [33]	odd	$h(x) = 0, f_i \in \mathbb{F}_2$	$2I + 154M/S$	$2I + 132M/S$
	our work	general	$h_i \in \mathbb{F}_2, f_6 = 0$	$2I + 118M + 4S$	$2I + 106M + 19S$
		two	$h_i \in \mathbb{F}_2, f_6 = 0$	$2I + 110M + 4S$	$2I + 98M + 13S$
two		$h(x) = 1, f_6 = 0$	$2I + 110M + 4S$ Table XI	$I + 14M + 11S$ Table XII	

TABLE V
SPEEDING UP GROUP OPERATIONS OF HARLEY'S ALGORITHM.

genus		field charac.	curve properties	cost	
				addition	doubling
2	Harley [37]	odd	$h(x) = 0$	$2I + 27M/S$	$2I + 30M/S$
	Lange [24]	odd		$2I + 24M + 3S$	$2I + 26M + 6S$
	Matsuo et al. [23]	odd	$h(x) = 0$	$2I + 25M/S$	$2I + 27M/S$
	Miyamoto et al. [25]	odd	$h(x) = 0, f_4 = 0$	$I + 26M/S$	$I + 27M/S$
	Takahashi [26]	odd	$h(x) = 0$	$I + 25M/S$	$I + 29M/S$
	Sugizaki et al. [35]	even	$h_i \in \mathbb{F}_{2^n}$	$I + 27M/S$	$I + 26M/S$
	Lange [14], [36]	general	$h_i \in \mathbb{F}_{2^n}, f_4 = 0$	$I + 22M + 3S$	$I + 22M + 5S$
		two	$h_i \in \mathbb{F}_{2^n}, f_4 = 0$	$I + 21M + 3S$	$I + 20M + 5S$
two		$h_2 = 0, h_i \in \mathbb{F}_{2^n}, f_4 = 0$	I+21M+3S	$I + 17M + 5S$	
our work	two	$h(x) = x, f_4 = f_3 = f_2 = 0$	–	I+9M+6S Table VI	
3	Kuroki et al. [29]	odd	$h(x) = 0, f_6 = 0$	$I + 81M/S$	$I + 74M/S$
	Gonda et al. [15]	odd	$h(x) = 0, f_6 = 0$	$I + 70M/S$	$I + 71M/S$
our work	general	$h_i \in \mathbb{F}_2, f_6 = 0$	$I + 70M + 6S$	$I + 62M + 10S$	
	two	$h_i \in \mathbb{F}_2, f_6 = 0$	$I + 65M + 6S$	$I + 53M + 10S$	
	two	$h(x) = 1, f_6 = 0$	I+65M+6S Table IX	$I + 22M + 7S$ Table X	

TABLE VI

EXPLICIT FORMULAE FOR DOUBLING A DIVISOR ON SPECIAL CURVES OF GENUS TWO OVER \mathbb{F}_{2^n} WITH $h(x) = x$ (HARLEY)

Input	Weight two reduced divisors $D = \text{div}(u, v)$ with $u = x^2 + u_1x + u_0$ $v = v_1x + v_0$ furthermore: $h = x$ and $f = x^5 + f_1x + f_0$	
Output	A weight two reduced divisor $D' = \text{div}(u', v') = [2]D$ with $u' = x^2 + u'_1x + u'_0$; $v' = v'_1x + v'_0$;	
Step	Procedure	Cost
1	Compute resultant r of u and $h + 2v$: $r = u_0$;	—
2	Compute almost inverse $\text{inv} \equiv r/\tilde{v} \pmod{u_1}$: $\text{inv}_1 = 1$; $\text{inv}_0 = u_1$;	—
3	Compute $k \equiv [(f - hv - v^2)/u] \pmod{u}$: $w_0 = v_1^2$; $w_1 = u_1^2$; $k_1 = w_1$; $t_1 = u_1k_1$; $k_0 = t_1 + w_0 + v_1$;	$1M + 2S$
4	Compute $s' = k\text{inv} \pmod{u}$: $t_2 = u_0k_0$; $s'_1 = k_0$; $s'_0 = (u_0 + u_1)(k_0 + k_1) + t_1 + t_2$; If $s'_1 = 0$ perform Cantor's Algorithm	$2M$
5	Compute s_1 and s_0u_1 : $t_3 = t_2^{-1} (= 1/(rs'_1))$; $w_3 = r^2t_3 (= 1/s_1)$; $w_4 = w_3^2$; $s_1 = s'_1t_3$; $t_6 = t_1 + k_1s_1 (= s_0u_1)$;	$I + 3M + 3S$
6	Compute $z = su$ (Karatsuba): $z_0 = s'_0$; $z_1 = t_6 + s'_1$; $z_2 = w_1$; $z_3 = s_1$;	—
7	Compute $u' = 1/s_1^2((su + h + v)^2 + f)/u^2$: $u'_2 = 1$; $u'_1 = w_4$; $u'_0 = w_4k_1^2 + k_1 + w_3$;	$M + S$
8	Compute $v' \equiv h + z + v \pmod{u'}$ (Karatsuba): $t_4 = w_3$; $t_7 = t_4 + z_2$; $t_5 = t_7u'_0$; $v'_1 = (z_3 + t_7)(u'_0 + u'_1) + t_4 + t_5 + 1 + z_1 + v_1$; $v'_0 = t_5 + z_0 + v_0$;	$2M$
Total		$I + 9M + 6S$

TABLE VII
EXPLICIT FORMULAE FOR ADDING ON A HEC OF GENUS TWO (CANTOR)

Input	Weight two reduced divisors $D_1 = (u_1, v_1)$ and $D_2 = (u_2, v_2)$ with $u_1 = x^2 + ax + b$; $u_2 = x^2 + cx + d$; $v_1 = ix + j$; $v_2 = kx + l$; furthermore: $h = h_2x^2 + h_1x + h_0$; where $h_i \in \{0, 1\}$; $f = x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$; where $f_i \in \{0, 1\}$;	
Output	A weight two reduced divisor $D' = (u', v') = D_1 + D_2$ with $u' = x^2 + a_3x + b_3$; $v' = i_3x + j_3$;	
Step	Procedure	Cost
1	Compute $\gcd(u_1, u_2) = 1 = s_1u_1 + s_2u_2$: $r_{21} = a - c$; $r_{20} = b - d$; $t_{20} = -1$; $r_{31} = r_{21} \cdot c - r_{20}$; $r_{30} = r_{21} \cdot d$; $t_{30} = r_{21}$; $t_{31} = 1$; $r_{40} = r_{31} \cdot r_{20} - r_{21} \cdot r_{30}$; $s_{20} = -r_{31} - r_{21}^2$; $s_{21} = -r_{21}$; $t_0 = r_{40}^{-1}$; $s_{20} = s_{20} \cdot t_0$; $s_{21} = s_{21} \cdot t_0$; $s_{11} = -s_{21}$; $s_{10} = -s_{21} \cdot c - s_{20} - a \cdot s_{11}$;	$I + 8M + 2S$
2	Compute $u' = u_1u_2 = x^4 + u'_3x^3 + u'_2x^2 + u'_1x + u'_0$ (Karatsuba): $t_0 = b \cdot d$; $t_1 = a \cdot c$; $t_2 = (a + b) \cdot (c + d)$; $u'_0 = t_0$; $u'_1 = t_2 - t_0 - t_1$; $u'_2 = t_1 + b + d$; $u'_3 = a + c$;	$2M$
3	Compute $v' = s_1u_1v_2 + s_2u_2v_1 \bmod u'$: $t_0 = b \cdot s_{10}$; $t_1 = a \cdot s_{11}$; $t_2 = (a + b) \cdot (s_{11} + s_{10})$; $su_{10} = t_0$; $su_{11} = t_2 - t_0 - t_1$; $su_{12} = t_1 + s_{10}$; $su_{13} = s_{11}$; $t_0 = l \cdot su_{10}$; $t_1 = k \cdot su_{11}$; $t_2 = (k + l) \cdot (su_{10} + su_{11})$; $d_{00} = t_0$; $d_{01} = t_2 - t_0 - t_1$; $d_{02} = t_1$; $t_0 = l \cdot (su_{10} + su_{12})$; $t_1 = k \cdot (su_{11} + su_{13})$; $t_2 = (k + l) \cdot (su_{10} + su_{11} + su_{12} + su_{13})$; $d_{20} = t_0$; $d_{21} = t_2 - t_0 - t_1$; $d_{22} = t_1$; $sv_{10} = d_{00}$; $sv_{11} = d_{01}$; $sv_{12} = d_{02} + d_{20} - d_{00}$; $sv_{13} = d_{21} - d_{01}$; $sv_{14} = d_{22} - d_{02}$; $t_0 = d \cdot s_{20}$; $t_1 = c \cdot s_{21}$; $t_2 = (c + d) \cdot (s_{21} + s_{20})$; $su_{20} = t_0$; $su_{21} = t_2 - t_0 - t_1$; $su_{22} = t_1 + s_{20}$; $su_{23} = s_{21}$; $t_0 = j \cdot su_{20}$; $t_1 = i \cdot su_{21}$; $t_2 = (i + j) \cdot (su_{20} + su_{21})$; $d_{00} = t_0$; $d_{01} = t_2 - t_0 - t_1$; $d_{02} = t_1$; $t_0 = j \cdot (su_{20} + su_{22})$; $t_1 = i \cdot (su_{21} + su_{23})$; $t_2 = (i + j) \cdot (su_{20} + su_{21} + su_{22} + su_{23})$; $d_{20} = t_0$; $d_{21} = t_2 - t_0 - t_1$; $d_{22} = t_1$; $sv_{20} = d_{00}$; $sv_{21} = d_{01}$; $sv_{22} = d_{02} + d_{20} - d_{00}$; $sv_{23} = d_{21} - d_{01}$; $sv_{24} = d_{22} - d_{02}$; $v'_3 = sv_{13} + sv_{23} - u'_3 \cdot (sv_{14} + sv_{24})$; $v'_2 = sv_{12} + sv_{22} - u'_2 \cdot (sv_{14} + sv_{24})$; $v'_1 = sv_{11} + sv_{21} - u'_1 \cdot (sv_{14} + sv_{24})$; $v'_0 = sv_{10} + sv_{20} - u'_0 \cdot (sv_{14} + sv_{24})$;	$22M$
4	Compute monic $u_3 = (f - v'h - v'^2)/u' = x^2 + a_3x + b_3$: $t_1 = -v'_3{}^2$; $a_3 = -2 \cdot v'_3 \cdot v'_2 + 1 - v'_3 \cdot h_2 - t_1 \cdot u'_3$; $b_3 = -v'_2 \cdot h_2 - v'_3 \cdot h_1 - 2 \cdot v'_3 \cdot v'_1 + f_4 - v'_2{}^2 - t_1 \cdot u'_2 - a_3 \cdot u'_3$; $t_0 = t_1^{-1}$; $a_3 = a_3 \cdot t_0$; $b_3 = b_3 \cdot t_0$;	$I + 7M + 2S$
5	Compute $v_3 = -(h + v') \bmod u_3 = i_3x + j_3$: $t_0 = -v'_3$; $t_1 = -(v'_2 + h_2) - t_0 \cdot a_3$; $i_3 = -(v'_1 + h_1) - (t_0 + t_1) \cdot (b_3 + a_3) + t_1 \cdot b_3 + t_0 \cdot a_3$; $j_3 = -(v'_0 + h_0) - t_1 \cdot b_3$;	$5M$
Total	fields of arbitrary characteristic, $h_i \in \mathbb{F}_2$, $f_4 = 0$ fields of characteristic two, $h_i \in \mathbb{F}_2$, $f_4 = 0$	$2I + 44M + 4S$ $2I + 42M + 4S$

TABLE VIII
EXPLICIT FORMULAE FOR DOUBLING ON A HEC OF GENUS TWO (CANTOR)

Input	Weight two reduced divisors $D = (u, v)$ with $u = x^2 + ax + b$; $v = ix + j$; furthermore: $h = h_2x^2 + h_1x + h_0$; where $h_i \in \{0, 1\}$; $f = x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$; where $f_4 \in \{0, 1\}$;	
Output	A weight two reduced divisor $D' = (u', v') = [2]D$ with $u' = x^2 + a_2x + b_2$; $v' = i_2x + j_2$;	
Step	Procedure	Cost
1	Compute $\gcd(u_1, h + 2v_1) = 1 = s_1u_1 + s_3(h + 2v_1)$; $r_{11} = h_1 + 2 \cdot i - h_2 \cdot a$; $r_{10} = h_0 + 2 \cdot j - h_2 \cdot b$; $r_{21} = r_{11} \cdot a - r_{10}$; $r_{20} = r_{11} \cdot b$; $r_{30} = r_{21} \cdot r_{10} - r_{11} \cdot r_{20}$; $s_{31} = r_{11}$; $s_{30} = r_{21}$; $t_0 = r_{30}^{-1}$; $s_{31} = s_{31} \cdot t_0$; $s_{30} = s_{30} \cdot t_0$; $s_{11} = -s_{31} \cdot h_2$; $s_{10} = -2 \cdot s_{31} \cdot i - s_{31} \cdot h_1 - s_{30} \cdot h_2 - a \cdot s_{11}$;	$I + 8M$
2	Compute $u' = u_1^2 = x^4 + u'_3x^3 + u'_2x^2 + u'_1x + u'_0$; $t_0 = b^2$; $t_1 = a^2$; $t_2 = (a + b)^2$; $u'_0 = t_0$; $u'_1 = t_2 - t_0 - t_1$; $u'_2 = 2 \cdot b + t_1$; $u'_3 = 2 \cdot a$;	$3S$
3	Compute $v' = s_1u_1v_1 + s_3(v_1^2 + f) \bmod u'_1$; $t_0 = j \cdot s_{10}$; $t_1 = i \cdot s_{11}$; $t_2 = (i + j) \cdot (s_{10} + s_{11})$; $sv_0 = t_0$; $sv_1 = t_2 - t_1 - t_0$; $sv_2 = t_1$; $t_0 = b \cdot sv_0$; $t_1 = a \cdot sv_1$; $t_2 = (a + b) \cdot (sv_0 + sv_1)$; $d_{00} = t_0$; $d_{01} = t_2 - t_0 - t_1$; $d_{02} = t_1$; $d_{10} = sv_2$; $t_0 = (b + 1) \cdot (sv_0 + sv_2)$; $t_1 = a \cdot sv_1$; $t_2 = (a + b + 1) \cdot (sv_0 + sv_1 + sv_2)$; $d_{20} = t_0$; $d_{21} = t_2 - t_0 - t_1$; $d_{22} = t_1$; $svv_0 = d_{00}$; $svv_1 = d_{01}$; $svv_2 = d_{02} + d_{20} - d_{00} - d_{10}$; $svv_3 = d_{21} - d_{01}$; $svv_4 = d_{10} + d_{22} - d_{02}$; $t_0 = j^2$; $t_1 = i^2$; $t_2 = (i + j)^2$; $vq_0 = t_0$; $vq_1 = t_2 - t_0 - t_1$; $vsq_2 = t_1$; $vsf_3 = f_3 - u'_2 - f_4 \cdot u'_3 + u'_3^2$; $vsf_2 =$ $vsq_2 + f_2 - u'_1 - f_4 \cdot u'_2 + u'_3 \cdot u'_2$; $vsf_1 = vsq_1 + f_1 - u'_0 - f_4 \cdot u'_1 + u'_3 \cdot u'_1$; $vsf_0 = vsq_0 + f_0 - f_4 \cdot u'_0 + u'_3 \cdot u'_0$; $t_0 = s_{30} \cdot vsf_0$; $t_1 = s_{31} \cdot vsf_1$; $t_2 = (s_{30} + s_{31}) \cdot (vsf_0 + vsf_1)$; $d_{00} = t_0$; $d_{01} = t_2 - t_0 - t_1$; $d_{02} = t_1$; $t_0 = (vsf_2 + vsf_0) \cdot s_{30}$; $t_1 = (vsf_3 + vsf_1) \cdot s_{31}$; $t_2 = (vsf_3 + vsf_2 +$ $vsf_1 + vsf_0) \cdot (s_{30} + s_{31})$; $d_{20} = t_0$; $d_{21} = t_2 - t_0 - t_1$; $d_{22} = t_1$; $svf_0 = d_{00}$; $svf_1 = d_{01}$; $svf_2 = d_{02} + d_{20} - d_{00}$; $svf_3 = d_{21} - d_{01}$; $svf_4 = d_{22} - d_{02}$; $v'_3 = svv_3 + svf_3 - u'_3 \cdot (svv_4 + svf_4)$; $v'_2 = svv_2 + svf_2 - u'_2 \cdot (svv_4 + svf_4)$; $v'_1 = svv_1 + svf_1 - u'_1 \cdot (svv_4 + svf_4)$; $v'_0 = svv_0 + svf_0 - u'_0 \cdot (svv_4 + svf_4)$;	$22M + 4S$
4	Compute monic $u_3 = (f - v'h - v'^2)/u' = x^2 + a_2x + b_2$; $t_1 = -v'_3$; $a_2 = -2 \cdot v'_3 \cdot v'_2 + 1 - v'_3 \cdot h_2 - t_1 \cdot u'_3$; $b_2 = -v'_2 \cdot h_2 - v'_3 \cdot h_1 - 2 \cdot v'_3 \cdot v'_1 + f_4 - v'_2^2 - t_1 \cdot u'_2 - a_2 \cdot u'_3$; $t_0 = t_1^{-1}$; $a_2 = a_2 \cdot t_0$; $b_2 = b_2 \cdot t_0$;	$I + 7M + S$
5	Compute $v_3 = -(h + v')$ mod $u_3 = i_2x + j_2$; $t_0 = -v_3$; $t_1 = -(v'_2 + h_2) - t_0 \cdot a_2$; $i_2 = -(v'_1 + h_1) - (t_0 + t_1) \cdot (b_2 +$ $a_2) + t_1 \cdot b_2 + t_0 \cdot a_2$; $j_2 = -(v'_0 + h_0) - t_1 \cdot b_2$;	$5M$
Total	fields of arbitrary characteristic, $h_i \in \mathbb{F}_2$, $f_4 = 0$ fields of characteristic two, $h_i \in \mathbb{F}_2$, $f_4 = 0$ fields of characteristic two, $h(x) = x$, $f_4 = 0$	$2I + 42M + 8S$ $2I + 40M + 8S$ $I + 23M + 6S$

TABLE IX
EXPLICIT FORMULAE FOR ADDING ON A HEC OF GENUS THREE (HARLEY)

Input	Weight three reduced divisors $D_1 = (u_1, v_1)$ and $D_2 = (u_2, v_2)$ with $u_1 = x^3 + ax^2 + bx + c$; $u_2 = x^3 + dx^2 + ex + f$; $v_1 = kx^2 + lx + m$; $v_2 = nx^2 + ox + p$; $h = h_3x^3 + h_2x^2 + h_1x + h_0$ where $h_i \in \{0, 1\}$; $f = x^7 + f_6x^6 + f_5x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$ where $f_6 \in \{0, 1\}$;	
Output	A weight three reduced divisor $D_3 = (u_3, v_3) = D_1 + D_2$ with $u_3 = x^3 + a_3x^2 + b_3x + c_3$; $v_3 = k_3x^2 + l_3x + m_3$;	
Step	Procedure	Cost
1	Compute resultant r of u_1 and u_2 (Bezout): $t_1 = ae$; $t_2 = bd$; $t_3 = bf$; $t_4 = ce$; $t_5 = af$; $t_6 = cd$; $t_7 = (f - c)^2$; $t_8 = (e - b)^2$; $t_9 = (d - a)(t_3 - t_4)$; $t_{10} = (d - a)(t_5 - t_6)$; $t_{11} = (e - b)(f - c)$; $r = (f - c + t_1 - t_2)(t_7 - t_9) + (t_5 - t_6)(t_{10} - 2t_{11}) + t_8(t_3 - t_4)$;	$12M + 2S$
2	Compute almost inverse $inv = r/u_1 \bmod u_2$: $inv_2 = (t_1 - t_2 - c + f)(d - a) - t_8$; $inv_1 = inv_2d - t_{10} + t_{11}$; $inv_0 = inv_2e - d(t_{10} - t_{11}) + t_9 - t_7$	$4M$
3	Compute $s' = rs \equiv (v_2 - v_1)inv \bmod u_2$ (Karatsuba): $t_{12} = (inv_1 + inv_2)(n - k + o - l)$; $t_{13} = (o - l)inv_1$; $t_{14} = (inv_0 + inv_2)(n - k + p - m)$; $t_{15} = (p - m)inv_0$; $t_{16} = (inv_0 + inv_1)(o - l + p - m)$; $t_{17} = (n - k)inv_2$; $r'_0 = t_{15}$; $r'_1 = t_{16} - t_{13} - t_{15}$; $r'_2 = t_{13} + t_{14} - t_{15} - t_{17}$; $r'_3 = t_{12} - t_{13} - t_{17}$; $r'_4 = t_{17}$; $t_{18} = dr'_4 - r'_3$; $s'_0 = r'_0 + ft_{18}$; $s'_1 = r'_1 - (e + f)(r'_4 - t_{18}) + er'_4 - ft_{18}$; $s'_2 = r'_2 - er'_4 + dt_{18}$; If $s'_2 = 0$ perform Cantor	$11M$
4	Compute $s = (s'/r)$ and make s monic: $w_1 = (rs'_2)^{-1}$; $w_2 = rw_1$; $w_3 = w_1s'_2{}^2$; $w_4 = rw_2$; $w_5 = w_4^2$; $s_0 = w_2s'_0$; $s_1 = w_2s'_1$;	$I + 6M + 2S$
5	Compute $z = su_1$: $z_0 = s_0c$; $z_1 = s_1c + s_0b$; $z_2 = s_0a + s_1b + c$; $z_3 = s_1a + s_0 + b$; $z_4 = a + s_1$;	$6M$
6	Compute $u' = [s(z + w_4(h + 2v_1)) - w_5((f - v_1h - v_1^2)/u_1)]/u_2$: $u'_3 = z_4 + s_1 - d$; $u'_2 = -du'_3 - e + z_3 + s_0 + w_4h_3 + s_1z_4$; $u'_1 = w_4(h_2 + 2k + s_1h_3) + s_1z_3 + s_0z_4 + z_2 - w_5 - du'_2 - eu'_3 - f$; $u'_0 = w_4(s_1h_2 + h_1 + 2l + 2s_1k + s_0h_3) + s_1z_2 + z_1 + s_0z_3 + w_5(a - f_6) - du'_1 - eu'_2 - fu'_3$	$15M$
7	Compute $v' = -(w_3z + h + v_1) \bmod u'$: $t_1 = u'_3 - z_4$; $v'_0 = -w_3(u'_0t_1 + z_0) - h_0 - m$; $v'_1 = -w_3(u'_1t_1 - u'_0 + z_1) - h_1 - l$; $v'_2 = -w_3(u'_2t_1 - u'_1 + z_2) - h_2 - k$; $v'_3 = -w_3(u'_3t_1 - u'_2 + z_3) - h_3$;	$8M$
8	Reduce u' , i.e. $u_3 = (f - v'h - v'^2)/u'$: $a_3 = f_6 - u'_3 - v'_3{}^2 - v'_3h_3$; $b_3 = -u'_2 - a_3u'_3 + f_5 - 2v'_2v'_3 - v'_3h_2 - v'_2h_3$; $c_3 = -u'_1 - a_3u'_2 - b_3u'_3 + f_4 - 2v'_1v'_3 - v'_2{}^2 - v'_2h_2 - v'_3h_1 - v'_1h_3$;	$5M + 2S$
9	Compute $v_3 = -(v' + h) \bmod u_3$: $k_3 = -v'_2 + (v'_3 + h_3)a_3 - h_2$; $l_3 = -v'_1 + (v'_3 + h_3)b_3 - h_1$; $m_3 = -v'_0 + (v'_3 + h_3)c_3 - h_0$;	$3M$
Total	fields of arbitrary characteristic, $h_i \in \mathbb{F}_2$, $f_6 = 0$ fields of characteristic two, $h_i \in \mathbb{F}_2$, $f_6 = 0$	$I + 70M + 6S$ $I + 65M + 6S$

TABLE X
EXPLICIT FORMULAE FOR DOUBLING ON A HEC OF GENUS THREE (HARLEY)

Input	A weight three reduced divisors $D_1 = (u_1, v_1)$ with $u_1 = x^3 + ax^2 + bx + c$; $v_1 = kx^2 + lx + m$; furthermore: $h = h_3x^3 + h_2x^2 + h_1x + h_0$ where $h_i \in \{0, 1\}$; $f = x^7 + f_6x^6 + f_5x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$ where $f_6 \in \{0, 1\}$;	
Output	A weight three reduced divisor $D_2 = (u_2, v_2) = [2]D_1$ with $u_2 = x^3 + a_2x^2 + b_2x + c_2$; $v_2 = k_2x^2 + l_2x + m_2$;	
Step	Procedure	Cost
1	Compute resultant r of u_1 and $h + 2v_1$ (Bezout): let $\tilde{h} = h + 2v_1$; $t_1 = a\tilde{h}_1$; $t_2 = b\tilde{h}_2$; $t_3 = b\tilde{h}_0$; $t_4 = c\tilde{h}_1$; $t_5 = a\tilde{h}_0$; $t_6 = c\tilde{h}_2$; $t_7 = (\tilde{h}_0 - h_3c)^2$; $t_8 = (\tilde{h}_1 - h_3b)^2$; $t_9 = (\tilde{h}_2 - h_3a)(t_3 - t_4)$; $t_{10} = (\tilde{h}_2 - h_3a)(t_5 - t_6)$; $t_{11} = (\tilde{h}_1 - h_3b)(\tilde{h}_0 - h_3c)$; $r = (\tilde{h}_0 - h_3c + t_1 - t_2)(t_7 - t_9) + (t_5 - t_6)[(t_5 - t_6)(\tilde{h}_2 - h_3a) - 2t_{11}] + t_8(t_3 - t_4)$;	$6M + 2S$
2	Compute almost inverse $inv = r/(h + 2v_1) \bmod u_1$: $inv_2 = -(t_1 - t_2 - h_3c + \tilde{h}_0)(\tilde{h}_2 - h_3a) + t_8$; $inv_1 = inv_2a + t_{10} - t_{11}$; $inv_0 = inv_2b + a(t_{10} - t_{11}) - t_9 + t_7$	$4M$
3	Compute $z = ((f - hv_1 - v_1^2)/u_1) \bmod u_1$: $t_{12} = k^2$; $z'_3 = f_6 - a$; $t_{13} = z'_3b$; $z'_2 = f_5 - h_3k - b - az'_3$; $z'_1 = f_4 - h_2k - h_3l - t_{12} - c - t_{13} - z'_2a$; $z_2 = f_5 - h_3k - 2b + a(a - 2z'_3)$; $z_1 = z'_1 - t_{13} + ab - c$; $z_0 = f_3 - h_2l - h_1k - 2kl - h_3m + c(a - 2z'_3) - z'_2b - z'_1a$;	$7M + 2S$
4	Compute $s' = zinv \bmod u_1$ (Karatsuba): $t_{12} = (inv_1 + inv_2)(z_1 + z_2)$; $t_{13} = z_1inv_1$; $t_{14} = (inv_0 + inv_2)(z_0 + z_2)$; $t_{15} = z_0inv_0$; $t_{16} = (inv_0 + inv_1)(z_0 + z_1)$; $t_{17} = z_2inv_2$; $r'_0 = t_{15}$; $r'_1 = t_{16} - t_{13} - t_{15}$; $r'_2 = t_{13} + t_{14} - t_{15} - t_{17}$; $r'_3 = t_{12} - t_{13} + t_{17}$; $r'_4 = t_{17}$; $t_{18} = ar'_4 - r'_3$; $s'_0 = r'_0 + ct_{18}$; $s'_1 = r'_1 - (b+c)(r'_4 - t_{18}) + br'_4 - ct_{18}$; $s'_2 = r'_2 - br'_4 + at_{18}$; If $s'_2 = 0$ perform Cantor	$11M$
5	Compute $s = (s'/r)$ and make s monic: $w_1 = (rs'_2)^{-1}$; $w_2 = w_1r$; $w_3 = w_1(s'_2)^2$; $w_4 = w_2r$; $(= r/s'_2)$; $w_5 = w_4^2$ $s_0 = w_2s'_0$; $s_1 = w_2s'_1$;	$I + 6M + 2S$
6	Compute $G = su_1$: $g_0 = s_0c$; $g_1 = s_1c + s_0b$; $g_2 = s_0a + s_1b + c$; $g_3 = s_1a + s_0b$; $g_4 = a + s_1$;	$6M$
7	Compute $u' = u_1^{-2}[(G + w_4v_1)^2 + w_4hG + w_5(hv_1 - f)]$: $u'_3 = 2s_1$; $u'_2 = s_1^2 + 2s_0 + w_4h_3$; $u'_1 = 2s_0s_1 + w_4(2k + h_3s_1 + h_2 - h_3a) - w_5$; $u'_0 = w_4[2l + h_1 + h_3s_0 - h_3b + 2ks_1 + a(ah_3 - 2k - h_2 - s_1h_3) + h_2s_1] + w_5(-f_6 + 2a) + s_0^2$;	$6M + 2S$

8	<p>Compute $v' = -(Gw_3 + h + v_1) \bmod u'$:</p> $\begin{aligned} t_1 &= u'_3 - g_4; \\ v'_3 &= -(t_1 u'_3 - u'_2 + g_3)w_3 - h_3; \\ v'_2 &= -(t_1 u'_2 - u'_1 + g_2)w_3 - h_2 - k; \\ v'_1 &= -(t_1 u'_1 - u'_0 + g_1)w_3 - h_1 - l; \\ v'_0 &= -(t_1 u'_0 + g_0)w_3 - h_0 - m; \end{aligned}$	$8M$
9	<p>Reduce u', i.e. $u_2 = (f - v'h - v'^2)/u'$:</p> $\begin{aligned} a_2 &= f_6 - u'_3 - v'^2_3 - v'_3 h_3; \\ b_2 &= -u'_2 - a_2 u'_3 + f_5 - 2v'_2 v'_3 - v'_3 h_2 - v'_2 h_3; \\ c_2 &= -u'_1 - a_2 u'_2 - b_2 u'_3 + f_4 - 2v'_1 v'_3 - v'^2_2 - v'_2 h_2 - v'_3 h_1 - v'_1 h_3; \end{aligned}$	$5M + 2S$
10	<p>Compute $v_2 = -(v' + h) \bmod u_2$:</p> $\begin{aligned} k_2 &= -v'_2 + (v'_3 + h_3)a_2 - h_2; \\ l_2 &= -v'_1 + (v'_3 + h_3)b_2 - h_1; \\ m_2 &= -v'_0 + (v'_3 + h_3)c_2 - h_0; \end{aligned}$	$3M$
Total	<p>fields of arbitrary characteristic, $h_i \in \mathbb{F}$, $f_6 = 0$ fields of characteristic two, $h_i \in \mathbb{F}$, $f_6 = 0$ fields of characteristic two, $h(x) = 1$, $f_6 = 0$</p>	$I + 62M + 10S$ $I + 53M + 10S$ $I + 22M + 7S$

TABLE XI

EXPLICIT FORMULAE FOR ADDING ON A HEC OF GENUS THREE (CANTOR)

Input	<p>Weight three reduced divisors $D_1 = (u_1, v_1)$ and $D_2 = (u_2, v_2)$ with</p> $\begin{aligned} u_1 &= x^3 + ax^2 + bx + c; \\ u_2 &= x^3 + dx^2 + ex + f; \\ v_1 &= kx^2 + lx + m; \\ v_2 &= nx^2 + ox + p; \end{aligned}$ <p>furthermore:</p> $\begin{aligned} h &= h_3x^3 + h_2x^2 + h_1x + h_0 \text{ where } h_i \in \{0, 1\}; \\ f &= x^7 + f_6x^6 + f_5x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0 \text{ where } f_6 \in \{0, 1\}; \end{aligned}$	
Output	<p>A weight three reduced divisor $D_3 = (u_3, v_3) = D_1 + D_2$ with</p> $\begin{aligned} u_3 &= x^3 + a_3x^2 + b_3x + c_3; \\ v_3 &= k_3x^2 + l_3x + m_3; \end{aligned}$	
Step	Procedure	Cost
1	<p>Compute $\gcd(u_1, u_2) = s_1u_1 + s_2u_2$ (EEA):</p> $\begin{aligned} r_{22} &= a - d; r_{21} = b - e; r_{20} = c - f; t_{20} = -1; r_{32} = r_{22} \cdot d - r_{21}; \\ r_{31} &= r_{22} \cdot e - r_{20}; r_{30} = r_{22} \cdot f; t_{31} = 1; t_{30} = r_{22}; r_{41} = r_{32} \cdot r_{21} - r_{22} \cdot r_{31}; \\ r_{40} &= r_{32} \cdot r_{20} - r_{22} \cdot r_{30}; t_{41} = -r_{22} \cdot t_{31}; t_{40} = r_{32} \cdot t_{20} - r_{22}^2; r_{51} = \\ r_{41} \cdot r_{31} &- r_{32} \cdot r_{40}; r_{50} = r_{41} \cdot r_{30}; t_{52} = -r_{32} \cdot t_{41}; t_{51} = r_{41} \cdot t_{31} - r_{32} \cdot t_{40}; \\ t_{50} &= r_{41} \cdot t_{30}; r_{60} = r_{51} \cdot r_{40} - r_{41} \cdot r_{50}; s_{22} = r_{51} \cdot t_{42} - r_{41} \cdot t_{52}; \\ s_{21} &= r_{51} \cdot t_{41} - r_{41} \cdot t_{51}; s_{20} = r_{51} \cdot t_{40} - r_{41} \cdot t_{50}; t_0 = r_{60}^{-1}; s_{22} = s_{22} \cdot t_0; \\ s_{21} &= s_{21} \cdot t_0; s_{20} = s_{20} \cdot t_0; s_{12} = -s_{22}; s_{11} = -s_{22} \cdot d - s_{21} - a \cdot s_{12}; \\ s_{10} &= -s_{22} \cdot e - s_{21} \cdot d - s_{20} - a \cdot s_{11} - b \cdot s_{12}; \end{aligned}$	$I + 33M + S$
2	<p>Compute $u' = u_1u_2/d^2 = u_1u_2$ (Karatsuba):</p> $\begin{aligned} t_0 &= c \cdot f; t_1 = b \cdot e; t_2 = (c + b) \cdot (e + f); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0; \\ d_{02} &= t_1; d_{10} = a \cdot d; d_{11} = 0; d_{12} = 0; t_0 = (a + c) \cdot (d + f); t_1 = d_{02}; \\ t_2 &= (a + b + c) \cdot (d + e + f); d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1; \\ u'_0 &= d_{00}; u'_1 = d_{01}; u'_2 = d_{20} - d_{10} - d_{00} + d_{02}; u'_3 = d_{21} - d_{01} + c + f; \\ u'_4 &= d_{10} + d_{22} - d_{02} + b + e; u'_5 = a + d; \end{aligned}$	$6M$

3	<p>Compute $v' = s_1 u_1 v_2 + s_2 u_2 v_1 \bmod u'$:</p> <p>$t_0 = c \cdot s_{10}; t_1 = b \cdot s_{11}; t_2 = (c+b) \cdot (s_{10} + s_{11}); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0;$ $d_{02} = t_1; d_{10} = a \cdot s_{12}; d_{11} = 0; d_{12} = 0; t_0 = (a+c) \cdot (s_{12} + s_{10}); t_1 = d_{02};$ $t_2 = (a+b+c) \cdot (s_{12} + s_{11} + s_{10}); d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1;$ $su_{10} = d_{00}; su_{11} = d_{01}; su_{12} = d_{20} - d_{10} - d_{00} + d_{02}; su_{13} = d_{21} - d_{01} + s_{10};$ $su_{14} = d_{10} + d_{22} - d_{02} + s_{11}; su_{15} = s_{12}; t_0 = su_{10} \cdot p; t_1 = su_{11} \cdot o;$ $t_2 = (su_{10} + su_{11}) \cdot (o + p); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0; d_{02} = t_1;$ $d_{10} = su_{12} \cdot n; d_{11} = 0; d_{12} = 0; t_0 = (p+n) \cdot (su_{10} + su_{12}); t_1 = d_{02};$ $t_2 = (n+o+p) \cdot (su_{12} + su_{11} + su_{10}); d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1;$ $d_{100} = d_{00}; d_{101} = d_{01}; d_{102} = d_{02} + d_{20} - d_{10} - d_{00}; d_{103} = d_{21} - d_{01};$ $d_{104} = d_{10} + d_{22} - d_{02}; t_0 = (su_{10} + su_{13}) \cdot p; t_1 = (su_{11} + su_{14}) \cdot o;$ $t_2 = (su_{10} + su_{13} + su_{11} + su_{14}) \cdot (o+p); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0; d_{02} = t_1;$ $d_{10} = (su_{12} + su_{15}) \cdot n; d_{11} = 0; d_{12} = 0; t_0 = (su_{10} + su_{13} + su_{12} + su_{15}) \cdot$ $(n+p); t_1 = d_{02}; t_2 = (su_{10} + su_{13} + su_{11} + su_{14} + su_{12} + su_{15}) \cdot (n+o+p);$ $d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1; d_{120} = d_{00}; d_{121} = d_{01}; d_{122} =$ $d_{02} + d_{20} - d_{10} - d_{00}; d_{123} = d_{21} - d_{01}; d_{124} = d_{10} + d_{22} - d_{02}; suv_{10} = d_{100};$ $suv_{11} = d_{101}; suv_{12} = d_{102}; suv_{13} = d_{120} - d_{100} + d_{103}; suv_{14} = d_{121} -$ $d_{101} + d_{104}; suv_{15} = d_{122} - d_{102}; suv_{16} = d_{123} - d_{103}; suv_{17} = d_{124} - d_{104};$ $t_0 = f \cdot s_{20}; t_1 = e \cdot s_{21}; t_2 = (f+e) \cdot (s_{20} + s_{21}); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0;$ $d_{02} = t_1; d_{10} = d \cdot s_{22}; d_{11} = 0; d_{12} = 0; t_0 = (d+f) \cdot (s_{22} + s_{20}); t_1 = d_{02};$ $t_2 = (d+e+f) \cdot (s_{22} + s_{21} + s_{20}); d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1;$ $su_{20} = d_{00}; su_{21} = d_{01}; su_{22} = d_{20} - d_{10} - d_{00} + d_{02}; su_{23} = d_{21} - d_{01} + s_{20};$ $su_{24} = d_{10} + d_{22} - d_{02} + s_{21}; su_{25} = s_{22}; t_0 = su_{20} \cdot m; t_1 = su_{21} \cdot l;$ $t_2 = (su_{20} + su_{21}) \cdot (l + m); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0; d_{02} = t_1;$ $d_{10} = su_{22} \cdot k; d_{11} = 0; d_{12} = 0; t_0 = (m+k) \cdot (su_{20} + su_{22}); t_1 = d_{02};$ $t_2 = (k+l+m) \cdot (su_{22} + su_{21} + su_{20}); d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1;$ $d_{100} = d_{00}; d_{101} = d_{01}; d_{102} = d_{02} + d_{20} - d_{10} - d_{00}; d_{103} = d_{21} - d_{01};$ $d_{104} = d_{10} + d_{22} - d_{02}; t_0 = (su_{20} + su_{23}) \cdot m; t_1 = (su_{21} + su_{24}) \cdot l;$ $t_2 = (su_{20} + su_{23} + su_{21} + su_{24}) \cdot (l + m); d_{00} = t_0; d_{01} = t_2 - t_1 - t_0;$ $d_{02} = t_1; d_{10} = (su_{22} + su_{25}) \cdot k; d_{11} = 0; d_{12} = 0; t_0 = (su_{20} + su_{23} +$ $su_{22} + su_{25}) \cdot (k+m); t_1 = d_{02}; t_2 = (su_{20} + su_{23} + su_{21} + su_{24} + su_{22} + su_{25}) \cdot$ $(k+l+m); d_{20} = t_0; d_{21} = t_2 - t_1 - t_0; d_{22} = t_1; d_{120} = d_{00}; d_{121} = d_{01};$ $d_{122} = d_{02} + d_{20} - d_{10} - d_{00}; d_{123} = d_{21} - d_{01}; d_{124} = d_{10} + d_{22} - d_{02};$ $suv_{20} = d_{100}; suv_{21} = d_{101}; suv_{22} = d_{102}; suv_{23} = d_{120} - d_{100} + d_{103};$ $suv_{24} = d_{121} - d_{101} + d_{104}; suv_{25} = d_{122} - d_{102}; suv_{26} = d_{123} - d_{103};$ $suv_{27} = d_{124} - d_{104};$ $c_0 = suv_{10} + suv_{20}; c_1 = suv_{11} + suv_{21}; c_2 = suv_{12} + suv_{22}; c_3 =$ $suv_{13} + suv_{23}; c_4 = suv_{14} + suv_{24}; c_5 = suv_{15} + suv_{25}; c_6 = suv_{16} + suv_{26};$ $c_7 = suv_{17} + suv_{27}; t_0 = c_7; t_2 = t_0 \cdot u'_5; t_1 = c_6 - t_2; t_3 = t_1 \cdot u'_4; t_4 = t_0 \cdot u'_3;$ $t_5 = t_1 \cdot u'_2; t_6 = t_0 \cdot u'_1; t_7 = t_1 \cdot u'_0; v'_5 = c_5 - (t_0 + t_1) \cdot (u'_4 + u'_5) + t_3 + t_2;$ $v'_4 = c_4 - t_4 - t_3; v'_3 = c_3 - (t_0 + t_1) \cdot (u'_2 + u'_3) + t_5 + t_4; v'_2 = c_2 - t_6 - t_5;$ $v'_1 = c_1 - (t_0 + t_1) \cdot (u'_0 + u'_1) + t_7 + t_6; v'_0 = c_0 - t_7;$</p>	45M
4	<p>Compute $u_3 = (f - v'h - v'^2)/u' = u_{34}x^4 + u_{33}x^3 + u_{32}x^2 + u_{31}x + u_{30}$ and make u_3 monic:</p> <p>$u_{34} = -v'_5{}^2; u_{33} = -2 \cdot v'_5 \cdot v'_4 - u_{34} \cdot u'_5;$ $u_{32} = -v'_4{}^2 - v'_5 \cdot h_3 - 2 \cdot v'_5 \cdot v'_3 - u_{34} \cdot u'_4 - u_{33} \cdot u'_5;$ $u_{31} = -2 \cdot v'_5 \cdot v'_2 + 1 - 2 \cdot v'_4 \cdot v'_3 - v'_4 \cdot h_3 - v'_5 \cdot h_2 - u_{33} \cdot u'_4 - u_{32} \cdot u'_5 - u_{34} \cdot u'_3;$ $u_{30} = -v'_3 - 2 \cdot v'_5 \cdot v'_1 - v'_3 \cdot h_3 - v'_5 \cdot h_1 + f_6 - 2 \cdot v'_4 \cdot v'_2 - v'_4 \cdot h_2 - u_{34} \cdot$ $u'_2 - u_{32} \cdot u'_4 - u_{31} \cdot u'_5 - u_{33} \cdot u'_3; t_0 = u_{34}^{-1}; u_{33} = u_{33} \cdot t_0;$ $u_{32} = u_{32} \cdot t_0; u_{31} = u_{31} \cdot t_0; u_{30} = u_{30} \cdot t_0;$</p>	$I + 20M + 2S$

5	Compute $v_3 = -(v' + h) \bmod u_3 = v_{33}x^3 + v_{32}x^2 + v_{31}x + v_{30}$: $t_0 = -v'_5; t_2 = t_0 \cdot u_{33}; t_1 = -v'_4 - t_2; t_3 = t_1 \cdot u_{32}; t_4 = t_0 \cdot u_{31}; t_5 = t_1 \cdot u_{30};$ $v_{33} = -(v'_3 + h_3) - (t_0 + t_1) \cdot (u_{32} + u_{33}) + t_3 + t_2; v_{32} = -(v'_2 + h_2) - t_4 - t_3;$ $v_{31} = -(v'_1 + h_1) - (t_0 + t_1) \cdot (u_{30} + u_{31}) + t_5 + t_4; v_{30} = -(v'_0 + h_0) - t_5;$	$6M$
6	Compute $u_4 = (f - v_3h - v_3^2)/u_3 = x^3 + a_3x^2 + b_3x + c_3$: $a_3 = -v_{33} \cdot h_3 - v_{33} + f_6 - u_{33}; b_3 = -v_{32} \cdot h_3 - v_{33} \cdot h_2 - 2 \cdot v_{33} \cdot v_{32} +$ $f_5 - u_{32} - a_3 \cdot u_{33}; c_3 = -v_{32} \cdot h_2 - 2 \cdot v_{33} \cdot v_{31} - v_{33} \cdot h_1 - v_{31} \cdot h_3 + f_4 -$ $v_{32}^2 - u_{31} - a_3 \cdot u_{32} - b_3 \cdot u_{33};$	$5M + S$
7	Compute $v_4 = -(v_3 + h) \bmod u_4 = k_3x^2 + l_3x + m_3$: $t_0 = -(v_{33} + h_3); k_3 = -(v_{32} + h_2) - t_0 \cdot a_3;$ $l_3 = -(v_{31} + h_1) - t_0 \cdot b_3; m_3 = -(v_{30} + h_0) - t_0 \cdot c_3;$	$3M$
Total	fields of arbitrary characteristic, $h_i \in \mathbb{F}_2, f_6 = 0$ fields of characteristic two, $h_i \in \mathbb{F}_2, f_6 = 0$	$2I + 118M + 4S$ $2I + 110M + 4S$

TABLE XII
EXPLICIT FORMULAE FOR DOUBLING ON A HEC OF GENUS THREE (CANTOR)

Input	A weight three reduced divisors $D_1 = (u_1, v_1)$ with $u_1 = x^3 + ax^2 + bx + c;$ $v_1 = kx^2 + lx + m;$ furthermore: $h = h_3x^3 + h_2x^2 + h_1x + h_0$ where $h_i \in \{0, 1\};$ $f = x^7 + f_6x^6 + f_5x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$ where $f_6 \in \{0, 1\};$	
Output	A weight three reduced divisor $D_2 = (u_2, v_2) = [2]D_1$ with $u_2 = x^3 + a_2x^2 + b_2x + c_2;$ $v_2 = k_2x^2 + l_2x + m_2;$	
Step	Procedure	Cost
1	Compute $\gcd(u_1, 2v_1 + h) = s_1u_1 + s_3(2v_1 + h)$: $r_{12} = h_2 + 2 \cdot k - h_3 \cdot a; r_{11} = h_1 + 2 \cdot l - h_3 \cdot b; r_{10} = h_0 + 2 \cdot m - h_3 \cdot c;$ $r_{22} = r_{12} \cdot a - r_{11}; r_{21} = r_{12} \cdot b - r_{10}; r_{20} = r_{12} \cdot c; t_{20} = 0; t_{21} = -1;$ $r_{31} = r_{22} \cdot r_{11} - r_{12} \cdot r_{21}; r_{30} = r_{22} \cdot r_{10} - r_{12} \cdot r_{20}; t_{31} = r_{12}; t_{30} = r_{22};$ $r_{41} = r_{31} \cdot r_{21} - r_{22} \cdot r_{30}; r_{40} = r_{31} \cdot r_{20}; t_{42} = -r_{22} \cdot t_{31}; t_{41} = -r_{31} - r_{22} \cdot t_{30};$ $t_{40} = 0; r_{50} = r_{41} \cdot r_{30} - r_{31} \cdot r_{40}; s_{32} = -r_{31} \cdot t_{42}; s_{31} = r_{41} \cdot t_{31} - r_{31} \cdot t_{41};$ $s_{30} = r_{41} \cdot t_{30}; t_0 = r_{50}^{-1}; s_{32} = s_{32} \cdot t_0; s_{31} = s_{31} \cdot t_0; s_{30} = s_{30} \cdot t_0;$ $s_{12} = -s_{32} \cdot h_3; s_{11} = -2 \cdot s_{32} \cdot k - s_{32} \cdot h_2 - s_{31} \cdot h_3 - a \cdot s_{12}; s_{10} =$ $-2 \cdot s_{32} \cdot l - s_{32} \cdot h_1 - 2 \cdot s_{31} \cdot k - s_{31} \cdot h_2 - s_{30} \cdot h_3 - a \cdot s_{11} - b \cdot s_{12};$	$I + 27M$
2	Compute $u_1 = u^2 = x^6 + u_{15}x^5 + u_{14}x^4 + u_{13}x^3 + u_{12}x^2 + u_{11}x + u_{10}$: $d_{00} = c^2; d_{02} = b^2; d_{01} = (b + c)^2 - d_{00} - d_{02}; d_{10} = a^2; d_{12} = 1;$ $d_{11} = (a + 1)^2 - d_{10} - d_{12}; d_{20} = (a + c)^2; d_{22} = (b + 1)^2; d_{21} =$ $(a + b + c + 1)^2 - d_{20} - d_{22}; u_{10} = d_{00}; u_{11} = d_{01}; u_{12} = d_{02} + d_{20} - d_{10} - d_{00};$ $u_{13} = d_{21} - d_{11} - d_{01};$ $u_{14} = d_{22} - d_{12} - d_{02} + d_{10}; u_{15} = d_{11};$	$8S$

3	<p>Compute $v' = s_1 u_1 v_1 + s_3 (v_1^2 + f) \bmod u_1$:</p> <p>$t_0 = m \cdot s_{10}; t_1 = l \cdot s_{11}; t_2 = (s_{11} + s_{10}) \cdot (l + m); d_{00} = t_0; d_{01} = t_2 - t_0 - t_1;$ $d_{02} = t_1; t_0 = k \cdot s_{12}; t_1 = 0; t_2 = t_0; d_{10} = t_0; d_{11} = 0; d_{12} = 0;$ $t_0 = (s_{10} + s_{12}) \cdot (m + k); t_1 = d_{02}; t_2 = (s_{12} + s_{11} + s_{10}) \cdot (k + l + m);$ $d_{20} = t_0; d_{21} = t_2 - t_0 - t_1; d_{22} = t_1; sv_0 = d_{00}; sv_1 = d_{01}; sv_2 =$ $d_{02} + d_{20} - d_{10} - d_{00}; sv_3 = d_{21} - d_{01}; sv_4 = d_{10} + d_{22} - d_{12} - d_{02}; t_0 = c \cdot sv_0;$ $t_1 = b \cdot sv_1; t_2 = (sv_1 + sv_0) \cdot (b + c); d_{00} = t_0; d_{01} = t_2 - t_0 - t_1; d_{02} = t_1;$ $d_{10} = a \cdot sv_2; d_{11} = 0; d_{12} = 0; t_0 = (sv_0 + sv_2) \cdot (a + c); t_1 = sv_1 \cdot b;$ $t_2 = (sv_0 + sv_1 + sv_2) \cdot (a + b + c); d_{20} = t_0; d_{21} = t_2 - t_0 - t_1; d_{22} = t_1;$ $d_{100} = d_{00}; d_{101} = d_{01}; d_{102} = d_{02} + d_{20} - d_{10} - d_{00}; d_{103} = d_{21} - d_{01};$ $d_{104} = d_{10} + d_{22} - d_{12} - d_{02}; t_0 = c \cdot (sv_0 + sv_3); t_1 = b \cdot (sv_1 + sv_4);$ $t_2 = (sv_1 + sv_4 + sv_0 + sv_3) \cdot (c + b); d_{00} = t_0; d_{01} = t_2 - t_0 - t_1;$ $d_{02} = t_1; d_{10} = d_{10}; d_{11} = 0; d_{12} = 0; t_0 = (sv_0 + sv_3 + sv_2) \cdot (a + c);$ $t_1 = d_{02}; t_2 = (sv_2 + sv_1 + sv_4 + sv_0 + sv_3) \cdot (a + b + c); d_{20} = t_0; d_{21} =$ $t_2 - t_0 - t_1; d_{22} = t_1; d_{120} = d_{00}; d_{121} = d_{01}; d_{122} = d_{02} + d_{20} - d_{10} - d_{00};$ $d_{123} = d_{21} - d_{01}; d_{124} = d_{10} + d_{22} - d_{12} - d_{02}; suv_0 = d_{100}; suv_1 = d_{101};$ $suv_2 = d_{102}; suv_3 = d_{103} + d_{120} - d_{100} + sv_0; suv_4 = d_{104} + d_{121} - d_{101} + sv_1;$ $suv_5 = d_{122} - d_{102} + sv_2; suv_6 = d_{123} - d_{103} + sv_3; suv_7 = d_{124} - d_{104} + sv_4;$ $d_{00} = m^2; d_{02} = l^2; d_{01} = (l + m)^2 - d_{00} - d_{02}; d_{10} = k^2; d_{12} = 0;$ $d_{11} = 0; d_{20} = (m + k)^2; d_{22} = d_{02}; d_{21} = (k + l + m)^2 - d_{20} - d_{22};$ $vsq_0 = d_{00}; vsq_1 = d_{01}; vsq_2 = d_{02} + d_{20} - d_{10} - d_{00}; vsq_3 = d_{21} - d_{01};$ $vsq_4 = d_{10} + d_{22} - d_{02}; vf_5 = f_5 - u_{14} - (f_6 - u_{15}) \cdot u_{15}; vf_4 = (vsq_4 +$ $f_4) - u_{13} - (f_6 - u_{15}) \cdot u_{14}; vf_3 = (vsq_3 + f_3) - u_{12} - (f_6 - u_{15}) \cdot u_{13}; vf_2 =$ $(vsq_2 + f_2) - u_{11} - (f_6 - u_{15}) \cdot u_{12}; vf_1 = (vsq_1 + f_1) - u_{10} - (f_6 - u_{15}) \cdot u_{11};$ $vf_0 = (vsq_0 + f_0) - (f_6 - u_{15}) \cdot u_{10}; t_0 = s_{30} \cdot vf_0; t_1 = s_{31} \cdot vf_1;$ $t_2 = (vf_0 + vf_1) \cdot (s_{31} + s_{30}); d_{00} = t_0; d_{01} = t_2 - t_0 - t_1; d_{02} = t_1;$ $d_{10} = s_{32} \cdot vf_2; d_{11} = 0; d_{12} = 0; t_0 = (vf_0 + vf_2) \cdot (s_{30} + s_{32}); t_1 = d_{02};$ $t_2 = (vf_0 + vf_1 + vf_2) \cdot (s_{30} + s_{31} + s_{32}); d_{20} = t_0; d_{21} = t_2 - t_0 - t_1; d_{22} = t_1;$ $d_{100} = d_{00}; d_{101} = d_{01}; d_{102} = d_{02} + d_{20} - d_{10} - d_{00}; d_{103} = d_{21} - d_{01};$ $d_{104} = d_{10} + d_{22} - d_{12} - d_{02}; t_0 = s_{30} \cdot (vf_0 + vf_3); t_1 = s_{31} \cdot (vf_1 + vf_4);$ $t_2 = (s_{30} + s_{31}) \cdot (vf_0 + vf_3 + vf_1 + vf_4); d_{00} = t_0; d_{01} = t_2 - t_0 - t_1;$ $d_{02} = t_1; d_{10} = s_{32} \cdot (vf_2 + vf_5); d_{11} = 0; d_{12} = 0; t_0 = (vf_0 + vf_3 +$ $vf_2 + vf_5) \cdot (s_{32} + s_{30}); t_1 = (vf_1 + vf_4) \cdot s_{31}; t_2 = (vf_0 + vf_1 + vf_2 +$ $vf_3 + vf_4 + vf_5) \cdot (s_{30} + s_{31} + s_{32}); d_{20} = t_0; d_{21} = t_2 - t_0 - t_1; d_{22} = t_1;$ $d_{120} = d_{00}; d_{121} = d_{01}; d_{122} = d_{02} + d_{20} - d_{10} - d_{00}; d_{123} = d_{21} - d_{01};$ $d_{124} = d_{10} + d_{22} - d_{12} - d_{02}; svf_0 = d_{100}; svf_1 = d_{101}; svf_2 = d_{102};$ $svf_3 = d_{103} + d_{120} - d_{100}; svf_4 = d_{104} + d_{121} - d_{101}; svf_5 = d_{122} - d_{102};$ $svf_6 = d_{123} - d_{103}; svf_7 = d_{124} - d_{104}; t_0 = suv_7 + svf_7; t_2 = t_0 \cdot u_{15};$ $t_1 = (suv_6 + svf_6) - t_2; t_3 = t_1 \cdot u_{14}; t_4 = t_0 \cdot u_{13}; t_5 = t_1 \cdot u_{12}; t_6 = t_0 \cdot u_{11};$ $t_7 = t_1 \cdot u_{10}; v_{15} = (suv_5 + svf_5) - (t_0 + t_1) \cdot (u_{14} + u_{15}) + t_2 + t_3; v_{14} =$ $(suv_4 + svf_4) - t_4 - t_3; v_{13} = (suv_3 + svf_3) - (t_0 + t_1) \cdot (u_{12} + u_{13}) + t_4 + t_5;$ $v_{12} = (suv_2 + svf_2) - t_6 - t_5; v_{11} = (suv_1 + svf_1) - (t_0 + t_1) \cdot (u_{11} +$ $u_{10}) + t_6 + t_7; v_{10} = (suv_0 + svf_0) - t_7;$</p>	44M + 7S
---	---	----------

4	<p>Compute $u_3 = (f - v_1h - v_1^2)/u_1$ and make u_3 monic, $u_3 = x^4 + u_{33}x^3 + u_{32}x^2 + u_{31}x + u_{30}$; $u_{34} = -v_{15}^2$; $u_{33} = -2 \cdot v_{15} \cdot v_{14} - u_{34} \cdot u_{15}$; $u_{32} = -v_{14}^2 - v_{15} \cdot h_3 - 2 \cdot v_{15} \cdot v_{13} - u_{34} \cdot u_{14} - u_{33} \cdot u_{15}$; $u_{31} = -2 \cdot v_{15} \cdot v_{12} + 1 - 2 \cdot v_{14} \cdot v_{13} - v_{14} \cdot h_3 - v_{15} \cdot h_2 - u_{33} \cdot u_{14} - u_{32} \cdot u_{15} - u_{34} \cdot u_{13}$; $u_{30} = -v_{13} - 2 \cdot v_{15} \cdot v_{11} - v_{13} \cdot h_3 - v_{15} \cdot h_1 + f_6 - 2 \cdot v_{14} \cdot v_{12} - v_{14} \cdot h_2 - u_{34} \cdot u_{12} - u_{32} \cdot u_{14} - u_{31} \cdot u_{15} - u_{33} \cdot u_{13}$; $t_0 = u_{34}^{-1}$; $u_{33} = u_{33} \cdot t_0$; $u_{32} = u_{32} \cdot t_0$; $u_{31} = u_{31} \cdot t_0$; $u_{30} = u_{30} \cdot t_0$;</p>	$I + 20M + 2S$
5	<p>Compute $v_3 = -(v' + h) \bmod u_3 = v_{33}x^3 + v_{32}x^2 + v_{31}x + v_{30}$; $t_0 = -v_{15}$; $t_2 = t_0 \cdot u_{33}$; $t_1 = -v_{14} - t_2$; $t_3 = t_1 \cdot u_{32}$; $t_4 = t_0 \cdot u_{31}$; $t_5 = t_1 \cdot u_{30}$; $v_{33} = -(v_{13} + h_3) - (t_0 + t_1) \cdot (u_{32} + u_{33}) + t_3 + t_2$; $v_{32} = -(v_{12} + h_2) - t_4 - t_3$; $v_{31} = -(v_{11} + h_1) - (t_0 + t_1) \cdot (u_{30} + u_{31}) + t_5 + t_4$; $v_{30} = -(v_{10} + h_0) - t_5$;</p>	$6M$
6	<p>Compute $u_4 = (f - v_3h - v_3^2)/u_3 = x^3 + a_2x^2 + b_2x + c_2$; $a_2 = -v_{33} \cdot h_3 - v_{33}^2 + f_6 - u_{33}$; $b_2 = -v_{32} \cdot h_3 - v_{33} \cdot h_2 - 2 \cdot v_{33} \cdot v_{32} + f_5 - u_{32} - a_2 \cdot u_{33}$; $c_2 = -v_{32} \cdot h_2 - 2 \cdot v_{33} \cdot v_{31} - v_{33} \cdot h_1 - v_{31} \cdot h_3 + f_4 - v_{32}^2 - u_{31} - a_2 \cdot u_{32} - b_2 \cdot u_{33}$;</p>	$6M + 2S$
7	<p>Compute $v_4 = -(v_3 + h) \bmod u_4 = k_2x^2 + l_2x + m_2$; $t_0 = -(v_{33} + h_3)$; $k_2 = -(v_{32} + h_2) - t_0 \cdot a_2$; $l_2 = -(v_{31} + h - 1) - t_0 \cdot b_2$; $m_2 = -(v_{30} + h - 0) - t_0 \cdot c_2$;</p>	$3M$
Total	<p>fields of arbitrary characteristic, $h_i \in \mathbb{F}_2$, $f_6 = 0$ fields of characteristic two, $h_i \in \mathbb{F}_2$, $f_6 = 0$ fields of characteristic two, $h(x) = 1$, $f_6 = 0$, see Table XIII</p>	<p>$2I + 106M + 19S$ $2I + 98M + 13S$ $I + 14M + 11S$</p>

TABLE XIII

EXPLICIT FORMULAE FOR DOUBLING ON SPECIAL CURVES OF GENUS THREE OVER \mathbb{F}_{2^n} WITH $h(x) = 1$ (CANTOR)

Input	<p>A weight three reduced divisors $D_1 = (u_1, v_1)$ with $u_1 = x^3 + ax^2 + bx + c$; $v_1 = kx^2 + lx + m$; $h = h_0 = 1$ $f = x^7 + f_6x^6 + f_5x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$ where $f_6 \in \{0, 1\}$;</p>	
Output	<p>A weight three reduced divisor $D_2 = (u_2, v_2) = [2]D_1$ with $u_2 = x^3 + a_2x^2 + b_2x + c_2$; $v_2 = k_2x^2 + l_2x + m_2$;</p>	
Step	Procedure	Cost
1	<p>Compute $d = \gcd(u_1, 1) = 1 = s_1a + s_3h$ ($s_3 = 1, s_1 = 0$): $s_3 = 1$; $s_1 = 0$;</p>	—
2	<p>Compute $u' = u_1^2$; $t_1 = a^2$; $t_2 = b^2$; $t_3 = c^2$;</p>	$3S$
3	<p>Compute $v' = v_1^2 + f \bmod u'$; $t_4 = k^2$; $t_5 = l^2$; $t_6 = m^2$; $v'_5 = f_5 + t_1$; $v'_4 = f_4 + t_4$; $v'_3 = f_3 + t_2$; $v'_2 = f_2 + t_5$; $v'_1 = f_1 + t_3$; $v'_0 = f_0 + t_6$;</p>	$3S$
4	<p>Compute $u'' = ((f - hv' - v'^2)/u')$; $u'_4 = v'_5^2$; $u'_3 = 0$; $u'_2 = v'_4^2 + t_1 \cdot u'_4$; $u'_1 = 1$; $u'_0 = v'_3 + t_2 \cdot u'_4 + t_1 \cdot u'_2$;</p>	$3M + 3S$
5	<p>Compute $u_2 = u''$ made monic $= x^3 + u_{22}x^2 + u_{21}x + u_{20}$; $u_{21} = u'_4^{-1}$; $u_{22} = u'_2 \cdot u_{21}$; $u_{20} = u'_0 \cdot u_{21}$;</p>	$1I + 2M$
6	<p>Compute $v_2 = -(v' + h) \bmod u_2 = v_{23}x^3 + v_{22}x^2 + v_{21}x + v_{20}$; $t_1 = v'_5 \cdot u_{22}$; $t_2 = v'_4 \cdot u_{21}$; $t_3 = (v'_5 + v'_4) \cdot (u_{22} + u_{21})$; $v_{23} = v'_3 + t_1$; $v_{22} = t_3 + t_1 + t_2 + v'_2$; $v_{21} = t_2 + v'_5 \cdot u_{20} + v'_1$; $v_{20} = v'_4 \cdot u_{20} + v'_0 + 1$;</p>	$5M$
7	<p>Compute $u_3 := (f - v_2h - v_2^2)/u_2 = x^3 + u_{32}x^2 + u_{31}x + u_{30}$; $a_2 = v_{23}^2$; $b_2 = u_{22} + f_5$; $c_2 = u_{22} \cdot a_2 + f_4 + v_{22}^2 + u_{21}$;</p>	$1M + 2S$
8	<p>Compute $v_3 := -(v_2 + h) \bmod u_3 = v_{32}x^2 + v_{31}x + v_{30}$; $k_2 = v_{22} + v_{23} \cdot a_2$; $l_2 = v_{21} + v_{23} \cdot b_2$; $m_2 = v_{20} + v_{23} \cdot c_2 + 1$;</p>	$3M$
Total		$I + 14M + 11S$