

# High Performance Arithmetic for special Hyperelliptic Curve Cryptosystems of Genus Two

Jan Pelzl

Thomas Wollinger

Christof Paar

Communication Security Group (COSY)

Department of Electrical Engineering and Information Sciences

Ruhr-Universitaet Bochum, Germany

Email: {pelzl, wollinger, cpaar}@crypto.rub.de

## Abstract

*Regarding the overall speed and power consumption, cryptographic applications in embedded environments like PDAs or mobile communication devices can benefit from specially designed cryptosystems with fixed parameters. In this contribution, we propose a highly efficient algorithm for a hyperelliptic curve cryptosystem (HECC) of genus two, well suited for these applications on constrained devices. This work presents a major improvement of HECC arithmetic for certain non-supersingular curves defined over fields of characteristic two. We optimized the group doubling operation and managed to speed up the whole cryptosystem by approximately 27% compared to the previously known most efficient case. Furthermore, an actual implementation of the new formulae on an embedded processor shows its practical relevance. A scalar multiplication can be performed in approximately 50ms on an 80MHz embedded device.*

## 1. Introduction

During the last decade, asymmetric cryptosystems based on elliptic curves have become very popular, especially for embedded applications. Elliptic curve cryptosystems (ECC) benefit from shorter operand sizes when compared to RSA or DL based systems. This fact makes ECC particularly well suited for small processors and memory constrained environments. Since their introduction, ECC have been extensively studied by the research community and in industry. Elliptic curves (EC) are a special case of hyperelliptic curves (HEC). In 1988, Koblitz suggested HEC for the use in cryptosystems [9]. In contrast to the EC case, it has only been until recently, that the idea to use HEC for cryptographic applications has been analyzed and implemented both in software [3,20,21,23,25–27,34] and in more

hardware-oriented platforms such as FPGAs [1, 32, 33]. Since the HEC operand size is only a fraction of the EC operand size, HECC is a cryptosystem of choice when targeting embedded environments. In 1999, Smart concluded that there seems to be little practical benefit in using HECC due to the difficulty of finding hyperelliptic curves and the relatively poor performance compared to ECC [28]. However, the efficiency of HEC group operations has been improved recently, and HECC [8, 14, 15, 17, 21, 23, 29] has become an attractive alternative to other cryptosystems such as RSA or ECC. An extensive study of how well HECC performs on different embedded architectures (ARM, PowerPC and ColdFire) was done in [34]. In this contribution, we drastically improved the arithmetic of the group doubling on certain genus-2 HEC, which is the crucial step in the divisor scalar multiplication.

## 2. Previous Work

Since the proposal of HEC for the use in cryptography in 1988 [9], it took several years until the arrival of first improvements in its arithmetic. Nagao [19] proposed several improvements of the polynomial arithmetic of Cantor's algorithm [2]. The same year, Harley came up with the first explicit formulae for Cantor's algorithm for genus-2 HEC [8] resulting in a drastic speed up of the cryptosystem. After Harley's proposal, several contributions containing improvements of the explicit formulae followed. The authors in [14, 15, 17, 29] targeted genus-2 curves whereas [12, 20, 21, 23] deal with the derivation and improvement of explicit formulae for HEC of genus 3 or 4. For more details on previous improvements made to the explicit formulae, the interested reader is referred to [20, 21, 23]. The publication by Lange [13] is the first to address formulae for genus-2 curves of arbitrary characteristic. In particular, curves of characteristic two are important from an implementational point of view.

### 3. Mathematical Background

In this section, we present an elementary introduction to some of 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 [10, 11].

#### 3.1. Hyperelliptic Curves

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 nonsingular 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  $\sigma$  of  $\overline{\mathbb{F}}$  over  $\mathbb{F}$  [16]. Each element of the Jacobian can be represented uniquely by a reduced divisor [2, 4]. This divisor again 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  $\mathbb{F}$ . This notation was first introduced by Mumford [18, page 3.17]. In the remainder of this paper, a divisor  $D$  represented by polynomials will be denoted by  $\text{div}(u, v)$ . Cantor's algorithm describes the group addition of two divisor classes in Mumford's notation. In 2000, Harley proposed the first explicit formulae for a group addition and a group doubling of divisor classes on  $\mathbb{J}_C(\mathbb{F})$  [8].

#### 3.2. Group Operations in the Jacobian

In [7], Gaudry and Harley could reduce the number of operations by distinguishing between possible cases according to the properties of the input divisors of the group operation. They described an efficient algorithm (using Karatsuba multiplication, CRT, and Newton Iteration) to reduce the overall complexity of the group operations. Most of the following contributions dealing with explicit formulae, concentrated on the (frequent) case where the two input divisors are coprime and of weight two. In the case of HEC of genus 2 over  $\mathbb{F}_{2^n}$ , this occurs with overwhelming probability of  $P \approx 1 - 2^{-n}$  and, thus, is of main interest for all implementations. The Algorithms for a divisor doubling and a divisor addition can be found in [22, Tables 1,2] respectively.

### 3.3. Security of HECC

To build a secure HECC, criteria to ensure a curve being not supersingular have to be considered [5]. For applications of HEC over binary fields  $\mathbb{F}_{2^n}$ , hyperelliptic curves of the form  $y^2 + h(x)y = f(x)$  with  $1 \leq \deg(h(x)) \leq g + 1$  cannot be supersingular, as shown in [5]. Most cryptographic applications based on EC or HEC require a group order of size of at least  $\approx 2^{160}$ . Thus, for HECC over  $\mathbb{F}_q$  we will need at least  $g \cdot \log_2 q \approx 160$ , where  $g$  is the genus of the curve. Particularly, for a curve of genus two, we will need a field  $\mathbb{F}_q$  with  $|\mathbb{F}_q| \approx 2^{80}$ , i.e., 80-bit long operands. It is well known that the best algorithm to compute the discrete logarithm in generic groups (such as the Jacobian of a HEC) is Pollard's rho method or one of its parallel variants [24, 31]. They solve the DLP with complexity  $O(\sqrt{n})$  in generic groups of order  $n$ . Up to our present knowledge, Jacobians of curves of genus  $\leq 3$  can be considered as generic groups. Recent results by Thériault [30] show progress in attacks against HEC of genus 3 and higher. Thériault 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 the same security level. For the moment, most efficient attacks presented in [30] are not (and will never be) practical because of the high storage usage and required computational power for genus-2 and genus-3 curves. However, we will take this security consideration into account and enlarge the field sizes accordingly.

For comparison, we implemented genus-3 and genus-4 HEC and enlarged the group sizes according to [30] (see *correction factors* given in Table 1).

**Table 1. Comparison of the complexity of the running time of different attacks on HECC**

$g$	1	2	3	4
Index Calculus	$q^2$	$q^2$	$q^2$	$q^2$
Pollard's Rho [6]	$q^{1/2}$	$q$	$q^{3/2}$	$q^2$
Reduced factor base [30]	n.a.	n.a.	$q^{3/2}$	$q^{8/5}$
With large primes [30]	n.a.	n.a.	$q^{10/7}$	$q^{14/9}$
Correction factor for $\log_2  \mathbb{F}_q $	-	-	1.05	1.286

### 4. New Improvements of the Arithmetic

In this section, we briefly outline our improvements on the formulae presented in [14]. All formulae are based on the ideas of Gaudry and Harley [7, 13], see Section 3.2. In this work, we further optimized the formulae for doubling a divisor class for fields  $\mathbb{F}_{2^n}$ . Table 2 presents the optimized

explicit formulae for doubling a divisor class on special binary curves as derived in this paper.

For our improvements we assume non-supersingular curves of the form  $y^2 + xy = x^5 + f_1x + f_0$  where  $f_0, f_1 \in \mathbb{F}_{2^n}$ , which can be achieved according to [14]. Obviously, they form a special class of curves but the free choice of  $f_0$  and  $f_1$  still leads to sufficiently many. To our knowledge, there are no security limitations using these types of curves.

The following modifications are applied to the steps of [22, Table 1] to reduce the required number of multiplications from 17 to 9 as displayed in Table 2:

- Steps 1-3 stay the same: the resultant turns out to be  $r = u_0$  and the inverse is  $inv = x + u_1$ ; the computation of  $k = k_1x + k_0$  costs  $1M$ .
- In the calculation of Step 4,  $s' = s'_1x + s'_0$  costs only two multiplications, since the terms  $k_1inv_1 = k_1 = w_1$  and  $w_1u_1 = k_1$  are already computed in Step 3. Furthermore,  $s'_1 = inv_0k_1 + inv_1k_0 + u_1w_1$  reduces to  $s'_1 = u_1k_1 + k_0 + u_1k_1 = k_0$ . Instead of computing  $s'_0 = k_0inv_0 + u_0k_1inv_1 = k_0u_1 + u_0k_1$  we compute  $s'_0 = (u_0 + u_1)(k_0 + k_1) + u_1k_1 + u_0k_0 = (u_0 + u_1)(k_0 + k_1) + t_1 + t_2$ . Since  $t_1$  is precomputed in Step 3, the calculation of  $(u_0 + u_1)(k_0 + k_1)$  and  $t_2$  has to be carried out ( $2M$ ). Note that  $t_2$  is required in Step 5, where  $1/(rs'_1) = 1/(k_0u_0) = 1/t_2$ .
- In Step 5, we do not normalize the  $s$ -polynomial as suggested by Takahashi [29] and compute  $s_1 = s'_1w_2$ . Instead of computing  $s_0$ , we set  $t_6 = s_0u_1 = u_1^3 + u_1^2s_1 = k_1(u_1 + s_1)$ , which is required in Step 6. Thus, we do not need the former value  $w_2$  from [22, Table 1] and can reduce the cost by two multiplications at the cost of one additional squaring.
- With the previously calculated value  $t_6$ , the computation of  $z = su$  comes for free, reducing the cost by  $2M$  compared to [22, Table 1]. These savings are possible due to  $s$  being *not* monic.
- Contrary to Step 7 in [22, Table 1], we have to normalize the  $u'$  polynomial by multiplying with  $1/s_1^2 = w_4$ . This results in  $u'_1 = w_4$  and  $u'_0 = w_4s_0^2 + k_1 + w_3$ . One can compute  $w_4s_0^2$  as  $w_4s_0^2 = w_4(u_1^2 + s_1^2)u_1^2 = w_4u_1^4 + w_4s_1^2u_1^2 = w_4u_1^4 + u_1^2 = w_4k_1^2 + k_1$  (due to the fact that  $s_0 = k_1 + u_1s_1$ ). Thus, Step 7 consumes only  $1M + 1S$ .
- With  $s$  being not monic,  $v'$  simply is obtained by  $v' = h + z + v \bmod u'$ . Using Karatsuba style reduction and some values computed in previous steps, Step 8 can be carried out with two multiplications only.

**Table 3. Timings for a Divisor Scalar Multiplication on a Genus-2 HEC for different field sizes on an ARM7 (80MHz)**

Field	Group order	Addition in $\mu s$	Doubling in $\mu s$	Scalarmult. in $ms$
$\mathbb{F}_{2^{63}}$	$2^{126}$	433	260	48.35
$\mathbb{F}_{2^{81}}$	$2^{162}$	471	296	69.06
$\mathbb{F}_{2^{83}}$	$2^{166}$	478	301	71.56
$\mathbb{F}_{2^{88}}$	$2^{176}$	484	308	77.19
$\mathbb{F}_{2^{91}}$	$2^{182}$	490	314	81.14
$\mathbb{F}_{2^{95}}$	$2^{190}$	494	319	85.74

## 5. Results

### 5.1. New Formulae

Table 2 presents the new improved doubling algorithm according to the modifications described in Section 4. The total cost of doubling a divisor on a HEC of genus 2 over fields  $\mathbb{F}_{2^n}$  is one field inversion ( $I$ ), 9 field multiplications ( $M$ ), and 6 field squarings ( $S$ ). The saving in multiplications is 47% for the cost of one additional squaring. From a computational point of view, squarings can be neglected in fields  $\mathbb{F}_{2^n}$  since their calculation comes almost for free.

The main operation of a hyperelliptic cryptosystem is a scalar multiplication with a divisor. If we assume a scalar of bitsize  $m$  and a windowing method (window size  $w = 4$ ) to perform the multiplication, we have to compute  $m$  group doublings and approximately  $0.2m$  additions. Thus, with the new proposed formulae, the cryptosystem obtains a speed up of  $\approx 27\%$ <sup>1</sup> compared to a system based on the doubling formulae in [13, 14].

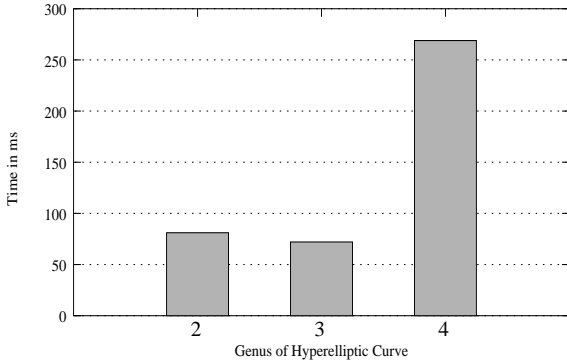
### 5.2. Implementational Results

We implemented our new formulae on an embedded processor — the ARM7TDMI — at a clockrate of 80MHz. Table 3 shows the timings of a divisor scalar multiplication for different field sizes. For most commonly used group orders of  $\approx 2^{128}$  to  $\approx 2^{190}$  a scalar multiplication takes approximately  $48ms$  to  $86ms$ . The numbers clearly indicate that HECC of genus two is a highly efficient cryptosystem since its core operation — the scalar multiplication — takes only a fraction of a second on an embedded device. Thus, using HECC, digital signatures are realizable in a reasonable time on embedded processors. Figure 1 shows a comparison of hyperelliptic cryptosystems of different genera with the same level of security realized over binary fields. As platform, the ARM7 was used. According to [30] and Table 1,

<sup>1</sup>For this approximation, it is assumed that one field inversion is as costly in time as 7 field multiplications. This approximation is based on the observation of several implementations [21].

**Table 2. Optimized explicit formulae for doubling a divisor on a HEC of genus two over  $\mathbb{F}_{2^n}$**

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 $\text{inv} \equiv r/\bar{v} \pmod{u_1}; \text{inv}_1 = 1; \text{inv}_0 = u_1;$	—
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 $s' = k\text{inv} \pmod{u};$ $t_2 = u_0k_0; s'_1 = k_0; g' = (u_0 + u_1)(k_0 + k_1) + t_1 + t_2;$ If $s'_1 = 0$ perform Cantor's Algorithm	$2M$
5	Compute $s_1$ and $s_0u_1;$ $t_3 = t_2^{-1} (= 1/(rs'_1)); w_3 = r^2t_3 (= 1/s_1); w_4 = w_3^2;$ $s_1 = s_1^2t_3; t_6 = t_1 + k_1s_1 (= s_0u_1);$	$I + 3M + 3S$
6	Compute $z = su$ (Karatsuba): $z_0 = s'; z_1 = t_6 + s'_1; z_2 = w_1; z_3 = s_1;$	—
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		$I + 9M + 6S$



**Figure 1. Timings for a Divisor Scalar Multiplication for different HECC with same Security Level (180bit) on an ARM7 (80MHz)**

an identical security level of 180bit results in the underlying fields  $\mathbb{F}_{2^{91}}$ ,  $\mathbb{F}_{2^{63}}$ , and  $\mathbb{F}_{2^{59}}$  for HEC of genus two, three, and four respectively. A scalar multiplication of a genus-2 HEC takes  $81.14ms$ , that of a genus-3 HEC takes  $72.09ms^2$  and on a HEC of genus 4 we need  $268.88ms^3$ .

## 6. Conclusions

With this contribution, we introduce a major speed-up of the arithmetic on certain genus-2 hyperelliptic curves over fields of characteristic two. We were able to reduce

the number of multiplications for doubling a divisor class by approximately 47%. Since doubling is the crucial step for performing a divisor scalar multiplication, this improvement results in a performance gain of approximately 27% for HECC of genus 2.

We implemented both the new algorithm for genus-2 HECC and the currently best formulae for HECC of genera 3 and 4 on an embedded device. This paper is the first contribution to address a thorough comparison of HECC of different genera while taking recent progress in cryptanalysis of HECC [30] into account.

From the results of our implementation of the proposed HECC on an embedded device, it clearly turns out that, contrary to common belief, digital signatures are feasible within an acceptable time. The comparison of HECC of genus 2, 3 and 4 shows, that genus-2 and genus-3 HECC over binary fields nearly have equal performance, if the formulae proposed in this paper for the arithmetic on genus-2 curves are used. Despite the proposed attack in [30], certain curves of genus three still perform slightly better. Due to their relatively high complexity, it takes a factor of approximately 3 more time for genus-4 curves to perform a scalar multiple of a divisor class compared to genus 2 and 3.

## References

- [1] N. Boston, T. Clancy, Y. Liow, and J. Webster. Genus Two Hyperelliptic Curve Coprocessor. In Kaliski, Koç, Paar, editor, *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2002*, New York, 2002. Springer Verlag. LNCS 2523.

<sup>2</sup>Implementation of curves with  $h(x)=1$ , formulae see [21]

<sup>3</sup>Implementation according to formulae in [23]

- [2] D. Cantor. Computing in the Jacobian of a Hyperelliptic Curve. In *Mathematics of Computation*, volume 48(177), pages 95 – 101, 1987.
- [3] A. Enge. The extended Euclidean algorithm on polynomials, and the computational efficiency of hyperelliptic cryptosystems, November 1999. Preprint.
- [4] W. Fulton. *Algebraic Curves - An Introduction to Algebraic Geometry*. W. A. Benjamin, Inc., Reading, Massachusetts, 1969.
- [5] S. Galbraith. Supersingular curves in cryptography. LNCS 2248:495–517, 2001.
- [6] P. Gaudry. An algorithm for solving the discrete log problem on hyperelliptic curves. In B. Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 19–34, Berlin, Germany, 2000. Springer-Verlag. LNCS 1807.
- [7] P. Gaudry and R. Harley. Counting Points on Hyperelliptic Curves over Finite Fields. In W. Bosma, editor, *The 4th Algorithmic Number Theory Symposium — ANTS IV*, pages 297 – 312, Berlin, 2000. Springer Verlag. LNCS 1838.
- [8] R. Harley. Fast Arithmetic on Genus Two Curves. Available at <http://cristal.inria.fr/~harley/hyper/>, 2000.
- [9] N. Kobitz. A Family of Jacobians Suitable for Discrete Log Cryptosystems. In Shafi Goldwasser, editor, *Advances in Cryptology — CRYPTO '88*, pages 94 – 99, Berlin, 1988. Springer-Verlag. LNCS 403.
- [10] N. Kobitz. Hyperelliptic Cryptosystems. In Ernest F. Brickell, editor, *Journal of Cryptology*, pages 139 – 150, 1989.
- [11] N. Kobitz. *Algebraic Aspects of Cryptography*. Algorithms and Computation in Mathematics. Springer-Verlag, 1998.
- [12] 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*, Jan.29-Feb.1 2002.
- [13] 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/>.
- [14] T. Lange. Formulae for Arithmetic on Genus 2 Hyperelliptic Curves, 2003. Available at <http://www.ruhr-uni-bochum.de/itsc/tanja/preprints.html>.
- [15] K. Matsuo, J. Chao, and S. Tsujii. Fast Genus Two Hyperelliptic Curve Cryptosystems. In *ISEC2001-31, IEICE*, 2001.
- [16] A. Menezes, Y.-H. Wu, and R. Zuccherato. An elementary introduction to hyperelliptic curves. In N. Kobitz, editor, *Algebraic Aspects of Cryptography*, Berlin, Heidelberg, 1996. Springer Verlag.
- [17] Y. Miyamoto, H. Doi, K. Matsuo, J. Chao, and S. Tsuji. A Fast Addition Algorithm of Genus Two Hyperelliptic Curve. In *SCIS, IEICE Japan*, pages 497 – 502, 2002. in Japanese.
- [18] D. Mumford. Tata lectures on theta II. In *Prog. Math.*, volume 43. Birkhäuser, 1984.
- [19] K. Nagao. Improving group law algorithms for Jacobians of hyperelliptic curves. In W. Bosma, editor, *ANTS IV*, volume LNCS 1838, pages 439 – 448, Berlin, 2000. Springer Verlag.
- [20] J. Pelzl. Hyperelliptic Cryptosystems on Embedded Microprocessors, September 2002. Diplomarbeit, Fakultät für Elektrotechnik und Informationstechnik, Ruhr-Uni-Bochum.
- [21] J. Pelzl, T. Wollinger, J. Guajardo, and C. Paar. Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves. In C. Walter, C. Koc, and C. Paar, editors, *Workshop on Cryptographic Hardware and Embedded Systems — CHES 2003*, pages 351–365, Berlin. Springer Verlag. LNCS 2779.
- [22] J. Pelzl, T. Wollinger, and C. Paar. High performance arithmetic for hyperelliptic curve cryptosystems of genus two. Cryptology ePrint Archive, Report 2003/212, 2003. <http://eprint.iacr.org/>.
- [23] J. Pelzl, T. Wollinger, and C. Paar. Low Cost Security: Explicit Formulae for Genus-4 Hyperelliptic Curves. In *Tenth Annual Workshop on Selected Areas in Cryptography — SAC 2003*. Springer-Verlag, 2003. LNCS.
- [24] J. M. Pollard. Monte carlo methods for index computation mod  $p$ . *Mathematics of Computation*, 32(143):918–924, July 1978.
- [25] 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*, pages 80 – 94, Berlin, 1998. Springer Verlag. LNCS 1514.
- [26] 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*, volume E83-A NO.4, pages 692 – 703, April 2000. IEICE Trans.
- [27] Y. Sakai, K. Sakurai, and H. Ishizuka. Secure Hyperelliptic Cryptosystems and their Performance. In *Public Key Cryptography*, pages 164 – 181, Berlin, 1998. Springer-Verlag. LNCS 1431.
- [28] N. Smart. On the Performance of Hyperelliptic Cryptosystems. In *Advances in Cryptology - EUROCRYPT '99*, volume 1592, pages 165 – 175, Berlin, 1999. Springer-Verlag. LNCS 1592.
- [29] M. Takahashi. Improving Harley Algorithms for Jacobians of Genus 2 Hyperelliptic Curves. In *SCIS, IEICE Japan*, 2002. in Japanese.
- [30] N. Thériault. Index calculus attack for hyperelliptic curves of small genus. In *Advances in Cryptology - ASIACRYPT '03*, Berlin, 2003. Springer Verlag. LNCS.
- [31] P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, Winter 1999.
- [32] T. Wollinger. Computer Architectures for Cryptosystems Based on Hyperelliptic Curves. Master's thesis, Worcester Polytechnic Institute, 2001.
- [33] 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*, volume III, pages 1159 – 1163, September 15-18 2002.
- [34] T. Wollinger, J. Pelzl, V. Wittelsberger, C. Paar, G. Saldamli, and Ç. K. Koç. Elliptic & hyperelliptic curves on embedded  $\mu p$ . *ACM Transactions in Embedded Computing Systems (TECS)*, 2003. Special Issue on Embedded Systems and Security.