Efficient Implementation of Genus Three Hyperelliptic Curve Cryptography over $\mathbb{F}_{2^n}$

Izuru Kitamura and Masanobu Katagi

Sony Corporation, 6-7-35 Kitashinagawa Shinagawa-ku, Tokyo, 141-0001 Japan
{Izuru.Kitamura, Masanobu.Katagi}@jp.sony.com

Abstract. The optimization of the Harley algorithm is an active area of hyperelliptic curve cryptography. We propose an efficient method for software implementation of genus three Harley algorithm over $\mathbb{F}_{2^n}$. Our method is based on fast finite field multiplication using one SIMD operation, SSE2 on Pentium 4, and parallelized Harley algorithm. We demonstrated that software implementation using proposed method is about 11% faster than conventional implementation.

Keywords. hyperelliptic curve arithmetic, scalar multiplication, Harley algorithm, SIMD operation, SSE2

1 Introduction

Hyperelliptic curve cryptography (HECC) was proposed by Koblitz [1]. A practical addition algorithm was proposed by Cantor [2] and Koblitz [1]. HECC has the advantage of a shorter operand length than elliptic curve cryptography (ECC), with the same level of security. The addition algorithm for divisor class groups of hyperelliptic curves, however, is more complicated and therefore its calculation is slower. This algorithm can be applied to arbitrary genus curves. On the other hand, the efficient attacks against curves of genus higher than or equal to four exist [3, 4]. It is advisable therefore to focus on genus two and genus three curves.

Recently, a new addition algorithm for genus two curves over odd prime characteristic was proposed by Harley [5, 6]. This algorithm drastically reduced the cost of calculating divisor addition and doubling by specifying the genus of curves. His study result triggered other researches, which eventually brought improvements and extensions to the algorithm [7–16].

The Harley algorithm has long and involved calculation procedures. In spite of this disadvantage, their parallelization was not enough studied. Most recently, this approach was proposed by Mishra et al [17]. They parallelized the affine and the inversion-free formulae of genus 2 curves. Their work targets at hardware implementation on the assumption of a multiprocessor environment, e.g. 4, 8 and 12 multipliers. However, speeding up the calculations based on a multiprocessor environment requires larger chip area, which is an area/speed tradeoff. We are interested in 2 or 3 parallel-field operations in SIMD (Single Instruction, Multiple Data) styles, which means that multiple data sets are processed at the same time because we also target software implementation. In particular, genus three HECC is suitable for SIMD because of size of its definition field. We propose a fast software implementation technique for genus 3 curve over $\mathbb{F}_{2^n}$ using SIMD-style operations. Many processors have SIMD architecture such as MMX and SSE for Pentium, AltiVec for PowerPC, and VIS for SPARC.

Several studies of the parallel arithmetic with SIMD of ECC have been reported [18–21]. In these works, independent finite field operations of ECC algorithm are executed in same time. These approaches are easily extended to the HECC addition algorithm. In the ECC case, however, the size of the finite field in ECC is larger than the present computer word size. Therefore finite field operations have to be divided several times. On the other hand, finite field operations for genus three HECC can be executed at one time because the definition field of genus three is smaller than 64 bits. In addition, SSE2 instructions provide 2×64bits shift and logical operations. We propose a parallel finite field multiplication, which means that two finite field multiplication AB and AC are executed at one time, over binary fields using SSE2. To apply this parallel finite field multiplication to the Harley algorithm, we manually optimized parallelized sequences of HECADD and HECDBL. In this paper, we report an efficient method for software implementation of genus three Harley algorithm over $\mathbb{F}_{2^n}$. Section 3 proposes the finite field multiplication using SSE2 and Section 4 shows parallel sequences of the procedure of genus three addition algorithm over binary fields. Section 5 presents the software implementation resulted from the proposed method.

2 Background

A genus g hyperelliptic curve C over $\mathbb{F}_{2^n}$ is defined as $C : y^2 + h(x)y = f(x)$, where $h(x),\ f(x) \in \mathbb{F}_{2^n}[x]$, $\deg h \leq g$, $\deg f = 2g + 1$, f is a monic polynomial and C is a non-singular curve. $J_c(\mathbb{F}_{2^n})$ is the Jacobian variety of C over $\mathbb{F}_{2^n}$. Any divisor class of $J_c(\mathbb{F}_{2^n})$ can be represented by a semi-reduced divisor. A semi-reduced divisor can be expressed by two polynomials $a, b \in \mathbb{F}_{2^n}[x]$ which satisfy

1. a is a monic polynomial,
2. $\deg b < \deg a$,
3. $f + hb + b^2 \equiv 0 \bmod a$.

This representation was reported by Mumford[22]. A semi-reduced divisor with $\deg a \leq g$ is called a reduced divisor and any divisor class of $J_c(\mathbb{F}_{2^n})$ is uniquely represented by a reduced divisor. Hereafter we denote $D \in J_c(\mathbb{F}_{2^n})$ by a reduced divisor $D = (a, b)$. $J_c(\mathbb{F}_{2^n})$ has an additive group structure. In [5] and [6], Harley defined a group operation to genus two curves over $\mathbb{F}_p$, where p is an odd prime number greater than three. In [16], Sugizaki et al. showed an extension of the Harley algorithm to curves over $\mathbb{F}_{2^n}$ for computing $D_1 + D_2 = D_3$ (HECADD) and $2D_1 = D_3$ (HECDBL). The extended Harley algorithm over $\mathbb{F}_{2^n}$ is follows.

Algorithm 1 *HECADD*

Input: $D_1 = (u_1, v_1), D_2 = (u_2, v_2), \deg u_1 = \deg u_2 = 2, \gcd(u_1, u_2) = 1$
Output: $D_3 = (u_3, v_3)$

1. $U \leftarrow u_1 u_2$
2. $S \leftarrow (v_2 + v_1)/u_1 \bmod u_2$
3. $V \leftarrow Su_1 + v_1 \bmod U$
4. $U \leftarrow (f + hV + V^2)/U$
5. Make U monic
6. $V \leftarrow V \bmod U$
7. $u_3 \leftarrow U, v_3 \leftarrow U + V + h$
8. return (u_3, v_3)

Algorithm 2 *HECDBL*

Input: $D_1 = (u_1, v_1), \deg u_1 = 2, \gcd(u_1, h) = 1$
Output: $D_3 = (u_3, v_3)$

1. $U \leftarrow u_1^2$
2. $S \leftarrow h^{-1}(f + hv_1 + v_1^2)/u_1 \bmod u_1$
3. $V \leftarrow Su_1 + v_1 \bmod U$
4. $U \leftarrow (f + hV + V^2)/U$
5. Make U monic
6. $V \leftarrow V \bmod U$
7. $u_3 \leftarrow U, v_3 \leftarrow U + V + h$
8. return (u_3, v_3)

These algorithms are similar to the elliptic curve chord-tangent law. We explain Algorithm 1. From step 1 to step 3 is called a composition part and from step 4 to step 7 is called a reduction part. In a composition part, we compute the semi-reduced divisor $D = (U, V)$ such that $D \sim -D_3$ where the symbol $\sim$ indicates to be linearly equivalent. In step 1, we compute $U = u_1 u_2$. In step 2 and step 3, we compute V such that $f + hV + V^2 \equiv 0 \bmod U$. V is obtained by $V \equiv v_1 \bmod u_1$ and $V \equiv v_2 \bmod u_2$ via the Chinese remainder theorem. In a reduction part, we compute a reduced divisor $D_3' = (u_3', v_3')$ such that $D_3' \sim D$. We compute $u_3' = (f + hV + V^2)/u_1 u_2$ and make u_3' a monic polynomial. Then we compute $v_3' \equiv V \bmod u_3'$. Finally, we output a divisor $D_3 = -D_3'$ as: $D_3 = (u_3, v_3) = (u_3', u_3' + v_3' + h)$. HECDBL is similar to HECADD, but the Chinese remainder theorem is replaced by the Newton iteration. In these algorithms, the Karatsuba algorithm is used to reduce the number of multiplications. In algorithms, HECADD takes 25 multiplications and 1 inversion, and HECDBL takes 27 multiplications and 1 inversion.

We deal with the most common situation, i.e., in HECADD, $\deg u_1 = \deg u_2 = 2$, $\gcd(u_1, u_2) = 1$ and in HECDBL, $\deg u_1 = 2, \gcd(u_1, h) = 1$. We will consider only this situation because the probability it will not occur is $O(\frac{1}{2^n})$ [23].

3 Fast Finite Field Multiplication for Genus Three Using SSE2

On the 64-bit architecture, $\mathbb{F}_{2^{59}}$ elements are represented by single-precision. Streaming SIMD Extension 2 (SSE2)[24, 25], which the Intel processor Pentium 4 includes, can deal with two $\mathbb{F}_{2^{59}}$ elements in parallel. In this section we discuss a parallel finite field multiplication algorithm over $\mathbb{F}_{2^n}$, represented by 64-bit single-precision, using SSE2. The following multiplication accelerates addition algorithm for genus three HECC over $\mathbb{F}_{2^{59}}$.

3.1 Genus Three Hyperelliptic Curve over $\mathbb{F}_{2^{59}}$

In this work we choose an example of the genus three hyperelliptic curve over $\mathbb{F}_{2^n}$ in some papers which is suitable for cryptography, i.e. its Jacobian has a large prime order subgroup. We use an isomorphic curve C_1 of the hyperelliptic curve described in Section 4.2 of [26]:

$$y^2 + h(x)y = f(x) \quad \text{over} \quad \mathbb{F}_{2^{59}},$$

$$\mathbb{F}_{2^{59}} \text{ is defined by } t^{59} + t^6 + t^5 + t^4 + t^3 + t + 1 = 0,$$

$$h(x) = x^3 + x^2 + 6723\text{B}8\text{D}13\text{BC}30\text{C}7x + 72\text{D}7\text{EE}15\text{A}5\text{C}9\text{CF}5,$$
$$f(x) = x^7 + x^6 + 6723\text{B}8\text{D}13\text{BC}30\text{C}7x^5 + 72\text{D}7\text{EE}15\text{A}5\text{C}9\text{CF}4x^4 + 24198\text{E}10\text{C}3\text{B}7566x^3 + 1\text{EB}9\text{AF}07\text{BD}3\text{B}303.$$

The order of the Jacobian of C_1 is:

$$2 \times 95780971304118053647396689122057683977359360476125197.$$

The curve has the same level security of 176bit-ECC.

3.2 SMUL: Conventional Finite Field Multiplication over $\mathbb{F}_{2^n}$

Some algorithms of finite field multiplication over $\mathbb{F}_{2^n}$ was proposed [27]. We explain a conventional algorithm of finite field multiplication over $\mathbb{F}_{2^n}$. The multiplication of two finite field elements in $\mathbb{F}_{2^n}$ is accomplished as follows. Let A, $B \in \mathbb{F}_{2^n}$, $F(x)$ be an irreducible binary polynomial of degree n. A and B can be represented by binary polynomial of degree at most $n-1$. Conventional finite field multiplication $AB \bmod F(x)$ is calculated the following 2 steps.

Step1. A fast algorithm for multiplication of two polynomials is presented Algorithm 5 in [28]. We make a table $T_B[i]$ for $0 \leq i < 16$ as:

$$T_B[j] \leftarrow (j_3x^3 + j_2x^2 + j_1x + j_0)B,$$

where $j = (j_3j_2j_1j_0)_2$. Then to calculate AB, this table is referred to by scanning A every 4 bits.

Step2. Reduction $AB \bmod F(x)$ is accomplished as explained Algorithm 6 in [27]. In this work, we use $F(x) = x^{59} + x^6 + x^5 + x^4 + x^3 + x + 1$ as an irreducible polynomial and the following congruence:

$$x^{64} \equiv x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5 \bmod F(x).$$

Using this congruence, AB can be expressed as:

$$\begin{aligned} AB &= AB_{\mathrm{h}} \times x^{64} + AB_{\mathrm{l}} \\ &\equiv AB_{\mathrm{h}} \times (x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5) + AB_{\mathrm{l}} \bmod F(x) \\ &\equiv AB'_{\mathrm{l}} \bmod F(x) \end{aligned}$$

.

A reduction of AB_{h} can be performed by adding $AB_{\mathrm{h}} \times x^i$ six times to AB, and AB'_{l} of degree 63 can be obtained. Then the remaining terms of AB from degree 63 to degree 59 can be reduced by adding AB'_{l}/x^i six times to AB. In this paper, we call this finite field multiplication algorithm composed of Algorithm 5 in [28] and Algorithm 6 in [27] as SMUL.

3.3 PMUL: Parallel Finite Field Multiplication using SSE2

We extend SMUL to finite field multiplication in parallel using SSE2. Before we explain details of the algorithm, we mention SSE2 instructions.

SSE2 Instructions: The Intel processors have SIMD instructions called MMX[24, 25]. The processor introduces four new data types: an 8×8-bit, 4×16-bit, 2×32-bit, and 64-bit data block. Additionally, the Intel Pentium 4 processor includes SSE2 technology that allows MMX instructions to work on a 128-bit data block. Therefore, SSE2 instructions deal with two $\mathbb{F}_q, |q| \leq 64$ elements at once. SSE2 instructions can multiply a 4×16-bit and a 4×16-bit or a 2×32-bit

and a 2×32-bit, however cannot multiply a 64-bit and a 64-bit. On the other hand, it can shift a 2×64-bit and perform 128 bitwise logical operations. $\mathbb{F}_{2^n}$ multiplication only needs shift and logical operations. These SSE2 characteristics show that only $\mathbb{F}_{2^n}$ elements are represented by single-precision.

We propose a parallel finite field multiplication using SSE2 (PMUL). The algorithm is the following:

Algorithm 3 *PMUL: Parallel Finite Field Multiplication using SSE2*

Input: A, B, C

Output: $D = AB \bmod F$, $E = AC \bmod F$, $F = x^{59} + x^6 + x^5 + x^4 + x^3 + x + 1$

1. $H \leftarrow 0$, $L \leftarrow 0$
2. For $j = 0$ to 15
   $T_{BC}[j] \leftarrow \boxed{(j_3x^3 + j_2x^2 + j_1x + j_0)B \mid (j_3x^3 + j_2x^2 + j_2x + j_0)C}$
3. For $j = 0$ to 14
   $i \leftarrow A/x^{4j} \bmod (x^3 + x^2 + x + 1)$
   $H \leftarrow H \oplus (T_{BC}[i] \gg (64 - 4j))$
   $L \leftarrow L \oplus (T_{BC}[i] \ll 4j)$
4. $L \leftarrow L \oplus (H \ll 11) \oplus (H \ll 10) \oplus (H \ll 9) \oplus (H \ll 8) \oplus (H \ll 6) \oplus (H \ll 5)$
5. $L \leftarrow L \oplus (L \gg 53) \oplus (L \gg 54) \oplus (L \gg 55) \oplus (L \gg 56) \oplus (L \gg 58) \oplus (L \gg 59)$
6. $L \leftarrow L \wedge \boxed{x^{58} + x^{57} + \ldots + x + 1 \mid x^{58} + x^{57} + \ldots + x + 1}$
7. $D \leftarrow L_{\mathrm{h}}$, $E \leftarrow L_{\mathrm{l}}$ /* $L = \boxed{AB \bmod F \mid AC \bmod F}$ */
8. return D, E

Algorithm 3 can compute parallel AB and AC with the table T_{BC} by scanning A. We note that the algorithm cannot compute parallel AB and CD because it is not available to scan A for the table T_{BC}. In Algorithm 3, the symbol $\gg i$ ($\ll i$) indicates that a 2×64-bit is shifted parallel to the right (left) by the i-bit, and $\oplus$ ($\wedge$) indicates a 128 bitwise logical exclusive OR (logical AND). In step 2, we make the table T_{BC} as previously, where $\boxed{***\mid***}$ indicates a 2×64-bit. In step 3, we scan A every 4 bits to select the appropriate polynomial on the table T_{BC}, then H retains the upper 64 bits of AB and AC, and L retains the lower 64 bits, as follows. $H \leftarrow \boxed{AB_{\mathrm{h}} \mid AC_{\mathrm{h}}}$, $L \leftarrow \boxed{AB_{\mathrm{l}} \mid AC_{\mathrm{l}}}$. In step 4, reductions of AB and AC from degree 116 to degree 64 can be accomplished as:

$$L \leftarrow \boxed{AB_{\mathrm{h}} \times (x^{11} + \cdots + x^5) + AB_{\mathrm{l}} \mid AC_{\mathrm{h}} \times (x^{11} + \cdots + x^5) + AC_{\mathrm{l}}}.$$

In step 5, the remaining terms of AB and AC from degree 63 to degree 59 can be reduced, finally to zero in step 6.

3.4 Another Choice of Irreducible Polynomial

In Algorithm 3, we use $\mathbb{F}_{2^{59}}$ defined by the heptanomial $x^{59} + x^6 + x^5 + x^4 + x^3 + x + 1$. The performance of finite field operation generally depends on weight (number of nonzero coefficients) of irreducible polynomial. The lowest weight of irreducible polynomial over $\mathbb{F}_{2^{59}}$ is pentanomial $F(x) = x^{59} + x^7 + x^4 + x^2 + 1$ [29]. Algorithm 3 can be extended to pentanomial case. We note that reduction step in Algorithm 3 depends on th 2nd highest degree t of the irreducible polynomial. If t satisfies the condition $t \geq 7$, 2×64-bit register overflows. Therefore, we need to modify the Algorithm 3.

In the case of $F(x) = x^{59} + x^7 + x^4 + x^2 + 1$, we use the following congruence:

$$x^{64} \equiv x^{12} + x^9 + x^7 + x^5 \bmod F(x).$$

Using this congruence, AB can be expressed as:

$$AB \equiv AB_{\mathrm{h}} \times (x^{12} + x^9 + x^7 + x^5) + AB_{\mathrm{l}} \bmod F(x)$$

AB_{h} is 52 degree and $AB_{\mathrm{h}} \times x^{12}$ is 64 degree at most. Since MSB of $AB_{\mathrm{h}} \times x^{12}$ is overflowed, we should modify Step 4 and 5 in Algorithm 3 as:

$$\begin{aligned} H &\leftarrow H \oplus ((H \wedge x^{52}) \gg 52) \\ L &\leftarrow L \oplus (H \ll 12) \oplus (H \ll 9) \oplus (H \ll 7) \oplus (H \ll 5) \\ L &\leftarrow L \oplus (L \gg 52) \oplus (L \gg 55) \oplus (L \gg 57) \oplus (L \gg 59). \end{aligned}$$

In spite of extra step for the overflow bit, PMUL using pentanomial is faster than heptanomial case because low weight of irreducible polynomial reduces the cost of XOR and Shift operations in Step 4 and 5 in Algorithm 3.

4 Parallel Genus Three Hyperelliptic Curve Arithmetic with SIMD Operations

We reschedule manually long procedures of the Harley algorithm for genus three [30] into parallel sequences to apply the proposed finite field multiplication PMUL. Table 4 and 5 in the Appendices show the two-processor version of affine coordinate HECADD and HECDBL for genus three curves [30]. We assume that these parallel sequences have SIMD-style operations in order to be applicable not only the hardware implementation but also software implementation. In Table 4 and 5, each column corresponds to an instruction. For these pairs which are expressed in the Table with the symbol $*$, we can use the proposed finite field multiplication PMUL, but for other cases, we compute sequentially via SMUL not in parallel via SSE2. We compare the non-parallel formulae [30] with our parallel formulae regarding calculation cost in Table 1. M, and I are the time required for multiplication, inversion of the binary field, respectively. P is the timing of the proposed finite field multiplication PMUL. Note that we estimate multiplication using the coefficient of curve parameter $f(x)$, $h(x)$ as $1M$.

Table 1. Number of PMUL, SMUL, and inversion of HECADD and HECDBL

	HECADD	HECDBL
Sequential Arithmetic	$77M + 1I$	$78M + 1I$
Parallel Arithmetic	$30P + 17M + 1I$	$30P + 18M + 1I$

5 Implementation Results

We present implementation results. All implementation was run under Windows XP on a 1.6GHz Pentium 4 processor, using Microsoft VC++ 6.0 with inline assembler MMX and SSE2. We applied PMUL using SSE2 to the parallelized addition algorithm for genus three HECC over $\mathbb{F}_{2^{59}}$.

First, Table 2 presents timing results for operation in the fields $\mathbb{F}_{2^{59}}$. Irreducible polynomials are pentanomial and heptanomial shown in Sec. 3.4 and Sec. 3.1, respectively. Squaring and inversion are implemented via Algorithm 7

in [27] and Algorithm 10 in [27], respectively. We compare two types of finite field multiplication algorithm, SMUL and PMUL. If irreducible polynomial is pentanomial, PMUL is approximately 14% faster than twice SMUL operations. Note that PMUL using pentanomial irreducible polynomial is faster than that using heptanomial as we mentioned in the previous section. In addition, timing of a successive two instruction SMUL and squaring is 330ns. On the other hand, timing of PMUL is 256ns. Squaring is generally faster than multiplication, but in our experiment, we had better deal with squaring as multiplication because PMUL instructions including squaring, which mean finite field multiplication A^2 and AB at the same time, is fast.

Second, Table 3 presents timing results of addition algorithm and scalar multiplication using binary method for genus three hyperelliptic curve C_1 shown in Sec. 3.1. The proposed method using PMUL is 11% faster than the conventional method.

Table 2. Timings of finite field operations over $\mathbb{F}_{2^{59}}$

Irreducible Polynomial	SMUL $\times 2$	PMUL	Squaring	Inversion
pentanomial	290ns	256ns	158ns	1.29μs
heptanomial	288ns	249ns	158ns	1.18μs

Table 3. Timings of addition algorithm and scalar multiplication

	Addition	Doubling	Scalar Multiplication (176 bit, binary method)
conventional	15.9μs	14.7μs	4.37ms
proposed	13.5μs	13.2μs	3.91ms

6 Conclusion

We propose a software optimization method of the genus three addition algorithm of HECC by combining parallelized Harley Algorithm and the parallel finite field multiplication using SSE2. We achieved 11% faster scalar multiplication than the usual implementation. Our idea is based on one of good feature of HECC. Its smaller definition field than ECC has comparable size with recent CPU register size.

However, another major CPU, e.g. ARM processor, does not have the 2×64 SIMD instruction. It is a further work to optimize the Harley algorithm using SIMD instructions.

Acknowledgements

The authors would like to thank Toru Akishita for valuable comments and suggestions.

References

1. Koblitz, N.: Hyperelliptic Cryptosystems. J. Cryptology **1** (1989) 139–150
2. Cantor, D.: Computing in the Jacobian of a Hyperelliptic Curve. Mathematics of Computation **48** (1987) 95–101
3. Adelman, L.M., DeMarrais, J., Huang, M.D.: A Subexponential Alogrithm for Discrete Logarithms over the Rational Subgroup of the Jacobian of Large Genus Hyperelliptic Curves over Finite Fields. In: ANTS-I, LNCS 877, Springer-Verlag (1994) 28–40
4. Gaudry, P.: An Algorithm for Solving the Discrete Log Problem on Hyperelliptic Curves. In: EUROCRYPT2000, LNCS 1807, Springer-Verlag (2000) 19–34
5. Harley, R.: Adding.text. http://cristal.inria.fr/~harley/hyper/ (2000)
6. Harley, R.: Doubling.c. http://cristal.inria.fr/~harley/hyper/ (2000)
7. Matsuo, K., Chao, J., Tsuji, S.: Fast Genus Two Hyperelliptic Curve Cryptosystems. Technical Report ISEC2001-31, IEICE Japan (2001) 89–96
8. Pelzl, J., Wollinger, T., Guajardo, J., Paar, C.: Hyperelliptic Curve Cryptosystems: Closing the Performance Gap to Elliptic Curves. Cryptology ePrint Archive, 2003/06, IACR (2003)
9. Lange, T.: Efficient Arithmetic on Genus 2 Hyperelliptic Curves over Finite Fields via Explicit Formulae. Cryptology ePrint Archive, 2002/121, IACR (2002)
10. Lange, T.: Inversion-free Arithmetic on Genus 2 Hyperelliptic Curves. Cryptology ePrint Archive, 2002/147, IACR (2002)
11. Lange, T.: Weighed Coordinate on Genus 2 Hyperellipitc Curve. Cryptology ePrint Archive, 2002/153, IACR (2002)
12. Takahashi, N., Morimoto, H., Miyaji, A.: Efficient Exponentiation on Genus Two Hyperelliptic Curves (ii). Technical Report ISEC2002-145, IEICE Japan (2003) in Japanese.
13. Takahashi, M.: Improving Harley Algorithms for Jacobians of Genus 2 Hyperelliptic Curves. In: Proc. of SCIS2002. (2002) in Japanese.
14. Miyamoto, Y., Doi, H., Matsuo, K., Chao, J., Tsujii, S.: A Fast Addition Algorithm of Genus Two Hyperelliptic Curves. In: Proc. of SCIS2002. (2002) in Japanese.
15. Kuroki, J., Gonda, M., Matsuo, K., Chao, J., Tsujii, S.: Fast Genus Three Hyperelliptic Curve Cryptosystems. In: Proc. of SCIS2002. (2002)
16. Sugizaki, T., Matsuo, K., Chao, J., Tsujii, S.: An Extension of Harley Addition Algorithm for Hyperelliptic Curves over Finite Fields of Characteristic Two. Technical Report ISEC2002-9, IEICE Japan (2002) 49–56
17. Mishra, P.K., Sarkar, P.: Parallelizing Explicit Formula for Arithmetic in the Jacobian of Hyperelliptic Curves. Cryptology ePrint Archive, Report 2003/180 (2003) http://eprint.iacr.org/.
18. Aoki, K., Hoshino, F., Kobayashi, T.: The Fastest ECC Implementations. Proc. of SCIS2000 (2000) in Japanese.
19. Smart, N.: The Hessian Form of an Elliptic Curve. In: CHES2001, LNCS 2162, Springer-Verlag (2001) 118–125
20. Aoki, K., Hoshino, F., Kobayashi, T., Oguro, H.: Elliptic Curve Arithmetic Using simd. In: ISC2001, LNCS 2200, Springer-Verlag (2001) 235–247
21. Izu, T., Takagi, T.: Fast Elliptic Curve Multiplications with SIMD Operations. In: ICICS 2002, LNCS 2513, Springer-Verlag (2002) 217–230
22. Mumford, D.: Tata lectures on theta ii. In: Progress in Mathematics. Number 43, Birkhäuser (1984)
23. Nagao, N.: Improving Group Law Algorithms for Jacobians of Hyperelliptic Curves. In: ANTS-IV, LNCS 1838, Springer-Verlag (2000) 439–448
24. Intel Corporation: IA-32 Intel Architecture Software Developer's Manual, volume 1: Basic Architecture. (2003)
25. Intel Corporation: IA-32 Intel Architecture Software Developer's Manual, volume 2: Instruction Set Reference. (2003)
26. Hess, F., Seroussi, G., Smart, N.: Two Topics in Hyperelliptic Cryptography. In: http://www.hpl.hp.com/techreports/2000/HPL-2000-118.html. (2000) 181–189

27. Hankerson, D., Hernandez, J.L., Menezes, A.: Software Implementation of Elliptic Curve Cryptography over Binary Fields. In: CHES2000, LNCS 1965, Springer-Verlag (2000) 1–24
28. Lopéz, J., Dahab, R.: High-speed Software Multiplication in $\mathbb{F}_{2^m}$. In: Indocrypt2000, LNCS 1977, Springer-Verlag (2000) 203–212
29. Seroussi, G.: Table of low-weight binary irreducible polynomials. http://www.hpl.hp.com/techreports/98/HPL-98-135.html (1998)
30. Pelzl, J.: Hyperelliptic Cryptosystems on Embedded Microprocessors. Communication Security Group, Rühr-Universität Bochum (2002) diploma thesis.

Appendices

Table 4. Parallel arithmetic of HECADD for genus three

HECADD 44M + 58A + I with 16 var.
Input: $D_1 = (u_1, v_1)$ and $D_2 = (u_2, v_2)$ with $u_1 = x^3 + u_{12}x^2 + u_{11}x + u_{10}$; $u_2 = x^3 + u_{22}x^2 + u_{21}x + u_{20}$; $v_1 = v_{12}x^2 + v_{11}x + v_{10}$; $v_2 = v_{22}x^2 + v_{21}x + v_{20}$; $h(x) = x^3 + h_2x^2 + h_1x + h_0$; $f(x) = x^7 + f_6x^6 + f_5x^5 + f_4x^4$ $+f_3x^3 + f_2x^2 + f_1x + f_0$;
Output: $D_3 = (u_3, v_3) = D_1 + D_2$ with $u_3 = x^3 + u_{32}x^2 + u_{31}x + u_{30}$; $v_3 = v_{32}x^2 + v_{31}x + v_{30}$;
$R_{00} \leftarrow u_{12}$
$R_{01} \leftarrow u_{11}$
$R_{02} \leftarrow u_{10}$
$R_{03} \leftarrow u_{22}$
$R_{04} \leftarrow u_{21}$
$R_{05} \leftarrow u_{20}$
$R_{06} \leftarrow R_{04} \times R_{00}$ $R_{07} \leftarrow R_{04} \times R_{02}$ *
$R_{08} \leftarrow R_{03} \times R_{01}$ $R_{09} \leftarrow R_{03} \times R_{02}$ *
$R_{10} \leftarrow R_{05} \times R_{01}$ $R_{11} \leftarrow R_{05} \times R_{00}$ *
$R_{05} \leftarrow R_{05} + R_{02}$ $R_{04} \leftarrow R_{04} + R_{01}$
$R_{02} \leftarrow R_{05} \times R_{05}$ $R_{12} \leftarrow R_{05} \times R_{04}$ *
$R_{04} \leftarrow R_{04} \times R_{04}$
$R_{03} \leftarrow R_{03} + R_{00}$ $R_{10} \leftarrow R_{10} + R_{07}$
$R_{13} \leftarrow R_{10} \times R_{04}$ $R_{10} \leftarrow R_{10} \times R_{03}$ *
$R_{05} \leftarrow R_{05} + R_{06}$ $R_{11} \leftarrow R_{11} + R_{09}$
$R_{05} \leftarrow R_{05} + R_{08}$ $R_{02} \leftarrow R_{02} + R_{10}$
$R_{08} \leftarrow R_{03} \times R_{11}$
$R_{11} \leftarrow R_{08} \times R_{11}$
$R_{03} \leftarrow R_{05} \times R_{03}$ $R_{05} \leftarrow R_{05} \times R_{02}$ *
$R_{05} \leftarrow R_{05} + R_{11}$ $R_{04} \leftarrow R_{04} + R_{03}$
$R_{05} \leftarrow R_{05} + R_{13}$ $R_{08} \leftarrow R_{08} + R_{12}$
$R_{00} \leftarrow u_{22}$
$R_{01} \leftarrow u_{21}$
$R_{03} \leftarrow R_{04} \times R_{01}$ $R_{01} \leftarrow R_{04} \times R_{00}$ *
$R_{01} \leftarrow R_{01} + R_{08}$ $R_{03} \leftarrow R_{03} + R_{02}$
$R_{08} \leftarrow R_{08} \times R_{00}$
$R_{03} \leftarrow R_{03} + R_{08}$
$R_{00} \leftarrow v_{12}$
$R_{02} \leftarrow v_{11}$
$R_{06} \leftarrow v_{10}$
$R_{07} \leftarrow v_{22}$
$R_{08} \leftarrow v_{21}$
$R_{09} \leftarrow v_{20}$
$R_{00} \leftarrow R_{00} + R_{07}$ $R_{02} \leftarrow R_{02} + R_{08}$
$R_{06} \leftarrow R_{06} + R_{09}$ $R_{07} \leftarrow R_{01} + R_{04}$
$R_{08} \leftarrow R_{03} + R_{04}$ $R_{09} \leftarrow R_{03} + R_{01}$
$R_{01} \leftarrow R_{01} \times R_{02}$ $R_{03} \leftarrow R_{03} \times R_{06}$
$R_{02} \leftarrow R_{02} + R_{00}$ $R_{06} \leftarrow R_{06} + R_{00}$
$R_{07} \leftarrow R_{07} \times R_{02}$ $R_{08} \leftarrow R_{08} \times R_{06}$
$R_{02} \leftarrow R_{02} + R_{06}$ $R_{07} \leftarrow R_{07} + R_{01}$
$R_{02} \leftarrow R_{02} \times R_{09}$ $R_{00} \leftarrow R_{00} \times R_{04}$
$R_{02} \leftarrow R_{02} + R_{01}$ $R_{09} \leftarrow R_{01} + R_{08}$
$R_{02} \leftarrow R_{02} + R_{03}$ $R_{09} \leftarrow R_{09} + R_{03}$
$R_{07} \leftarrow R_{07} + R_{00}$ $R_{09} \leftarrow R_{09} + R_{00}$
$R_{04} \leftarrow u_{22}$
$R_{10} \leftarrow u_{21}$
$R_{11} \leftarrow u_{20}$
$R_{12} \leftarrow R_{00} \times R_{04}$ $R_{13} \leftarrow R_{00} \times R_{10}$ *
$R_{12} \leftarrow R_{12} + R_{07}$ $R_{09} \leftarrow R_{09} + R_{13}$
$R_{10} \leftarrow R_{10} + R_{11}$ $R_{02} \leftarrow R_{02} + R_{13}$
$R_{13} \leftarrow R_{12} \times R_{11}$ $R_{04} \leftarrow R_{12} \times R_{04}$ *
$R_{03} \leftarrow R_{03} + R_{13}$ $R_{00} \leftarrow R_{00} + R_{12}$
$R_{02} \leftarrow R_{02} + R_{13}$ $R_{09} \leftarrow R_{09} + R_{04}$
$R_{10} \leftarrow R_{10} \times R_{00}$
$R_{02} \leftarrow R_{02} + R_{10}$
$R_{01} \leftarrow R_{09} \times R_{05}$ $R_{06} \leftarrow R_{09} \times R_{09}$ *
$R_{01} \leftarrow 1/R_{01}$
$R_{04} \leftarrow R_{01} \times R_{05}$ $R_{06} \leftarrow R_{01} \times R_{06}$ *
$R_{05} \leftarrow R_{05} \times R_{04}$
$R_{12} \leftarrow R_{05} \times R_{05}$
$R_{00} \leftarrow R_{04} \times R_{03}$ $R_{01} \leftarrow R_{04} \times R_{02}$ *
$R_{08} \leftarrow R_{02} \times R_{00}$ $R_{09} \leftarrow R_{02} \times R_{01}$ *
$R_{02} \leftarrow R_{02} + R_{01}$ $R_{09} \leftarrow R_{09} + R_{04}$
$R_{07} \leftarrow R_{04} \times R_{00}$ $R_{04} \leftarrow R_{04} \times R_{01}$ *
$R_{08} \leftarrow R_{08} + R_{04}$ $R_{09} \leftarrow R_{09} + R_{00}$
$R_{11} \leftarrow R_{10} \times R_{00}$ $R_{13} \leftarrow R_{10} \times R_{01}$ *
$R_{07} \leftarrow R_{07} + R_{13}$ $R_{08} \leftarrow R_{08} + R_{10}$
$R_{04} \leftarrow u_{22}$
$R_{14} \leftarrow u_{21}$
$R_{10} \leftarrow R_{02} + R_{01}$ $R_{03} \leftarrow R_{14} + R_{09}$
$R_{10} \leftarrow R_{10} + R_{04}$ $R_{03} \leftarrow R_{03} + R_{00}$
$R_{13} \leftarrow R_{10} \times R_{04}$ $R_{15} \leftarrow R_{10} \times R_{14}$ *
$R_{13} \leftarrow R_{13} + R_{03}$ $R_{15} \leftarrow R_{15} + R_{12}$
$R_{04} \leftarrow R_{01} \times R_{02}$ $R_{14} \leftarrow R_{01} \times R_{09}$ *
$R_{13} \leftarrow R_{13} + R_{04}$ $R_{15} \leftarrow R_{15} + R_{14}$
$R_{03} \leftarrow R_{01} \times h_2$ $R_{04} \leftarrow R_{01} \times R_{08}$ *
$R_{13} \leftarrow R_{13} + R_{05}$ $R_{03} \leftarrow R_{03} + h_1$
$R_{01} \leftarrow R_{01} + h_2$ $R_{03} \leftarrow R_{03} + R_{00}$
$R_{01} \leftarrow R_{05} \times R_{01}$ $R_{03} \leftarrow R_{05} \times R_{03}$ *
$R_{15} \leftarrow R_{15} + R_{01}$ $R_{04} \leftarrow R_{04} + R_{03}$
$R_{01} \leftarrow R_{00} \times R_{02}$ $R_{03} \leftarrow R_{00} \times R_{09}$ *
$R_{15} \leftarrow R_{15} + R_{01}$ $R_{04} \leftarrow R_{04} + R_{03}$
$R_{05} \leftarrow u_{22}$
$R_{14} \leftarrow u_{21}$
$R_{01} \leftarrow R_{13} \times R_{05}$ $R_{03} \leftarrow R_{13} \times R_{14}$ *
$R_{15} \leftarrow R_{15} + R_{01}$ $R_{04} \leftarrow R_{04} + R_{03}$
$R_{15} \leftarrow R_{15} + R_{08}$ $R_{04} \leftarrow R_{04} + R_{07}$
$R_{00} \leftarrow u_{20}$
$R_{03} \leftarrow R_{00} \times R_{10}$
$R_{15} \leftarrow R_{15} + R_{00}$ $R_{04} \leftarrow R_{04} + R_{03}$
$R_{03} \leftarrow R_{05} \times R_{15}$
$R_{04} \leftarrow R_{04} + R_{03}$
$R_{00} \leftarrow u_{12}$
$R_{03} \leftarrow R_{00} + f_6$

Table 4. Parallel arithmetic of HECADD for genus three – continued

$R_{03} \leftarrow R_{03} \times R_{12}$				
$R_{04} \leftarrow R_{04} + R_{03}$				
$R_{00} \leftarrow v_{12}$				
$R_{01} \leftarrow v_{11}$				
$R_{03} \leftarrow v_{10}$				
$R_{02} \leftarrow R_{02} + R_{10}$				
$R_{05} \leftarrow R_{02} \times R_{04}$	$R_{12} \leftarrow R_{02} \times R_{15}$	*		
$R_{05} \leftarrow R_{05} + R_{11}$	$R_{12} \leftarrow R_{12} + R_{07}$			
$R_{12} \leftarrow R_{12} + R_{04}$				
$R_{05} \leftarrow R_{06} \times R_{05}$	$R_{12} \leftarrow R_{06} \times R_{12}$	*		
$R_{05} \leftarrow R_{05} + h_0$	$R_{12} \leftarrow R_{12} + h_1$			
$R_{05} \leftarrow R_{05} + R_{03}$	$R_{12} \leftarrow R_{12} + R_{01}$			
$R_{01} \leftarrow R_{02} \times R_{13}$	$R_{03} \leftarrow R_{02} \times R_{10}$	*		
$R_{01} \leftarrow R_{01} + R_{15}$	$R_{03} \leftarrow R_{03} + R_{13}$			
$R_{01} \leftarrow R_{01} + R_{08}$	$R_{03} \leftarrow R_{03} + R_{09}$			
$R_{01} \leftarrow R_{06} \times R_{01}$	$R_{03} \leftarrow R_{06} \times R_{03}$	*		
$R_{01} \leftarrow R_{01} + h_2$	$R_{03} \leftarrow R_{03} + 1$			
$R_{01} \leftarrow R_{01} + R_{00}$				
$R_{00} \leftarrow R_{03} \times R_{03}$	$R_{02} \leftarrow R_{03} \times h_1$	*		
$R_{06} \leftarrow R_{00} + f_6$	$R_{07} \leftarrow R_{15} + R_{02}$			
$R_{00} \leftarrow h_2 \times R_{03}$	$R_{02} \leftarrow h_2 \times R_{01}$	*		
$R_{08} \leftarrow R_{13} + R_{00}$	$R_{07} \leftarrow R_{07} + R_{02}$			
$R_{06} \leftarrow R_{06} + R_{10}$	$R_{08} \leftarrow R_{08} + f_5$			
$R_{06} \leftarrow R_{06} + R_{03}$	$R_{08} \leftarrow R_{08} + R_{01}$			
$R_{00} \leftarrow R_{06} \times R_{10}$	$R_{02} \leftarrow R_{06} \times R_{13}$	*		
$R_{08} \leftarrow R_{08} + R_{00}$	$R_{07} \leftarrow R_{07} + R_{02}$			
$R_{00} \leftarrow R_{01} \times R_{01}$				
$R_{00} \leftarrow R_{00} + f_4$	$R_{07} \leftarrow R_{07} + R_{12}$			
$R_{07} \leftarrow R_{07} + R_{00}$	$R_{03} \leftarrow R_{03} + 1$			
$R_{00} \leftarrow R_{08} \times R_{10}$	$R_{02} \leftarrow R_{08} \times R_{03}$	*		
$R_{07} \leftarrow R_{07} + R_{00}$	$R_{12} \leftarrow R_{12} + R_{02}$			
$R_{00} \leftarrow R_{03} \times R_{06}$	$R_{02} \leftarrow R_{03} \times R_{07}$	*		
$R_{01} \leftarrow R_{01} + R_{00}$	$R_{05} \leftarrow R_{05} + R_{02}$			
$R_{01} \leftarrow R_{01} + h_2$	$R_{12} \leftarrow R_{12} + h_1$			
$R_{05} \leftarrow R_{05} + h_0$				
$u_{32} \leftarrow R_{06}$				
$u_{31} \leftarrow R_{08}$				
$u_{30} \leftarrow R_{07}$				
$v_{32} \leftarrow R_{01}$				
$v_{31} \leftarrow R_{12}$				
$v_{30} \leftarrow R_{05}$				

Table 5. Parallel arithmetic of HECDBL for genus three

HECDBL 43M + 55A + I with 16 var.
Input: $D_1 = (u_1, v_1)$ with $u_1 = x^3 + u_{12}x^2 + u_{11}x + u_{10}$; $v_1 = v_{12}x^2 + v_{11}x + v_{10}$; $h(x) = x^3 + h_2x^2 + h_1x + h_0$; $f(x) = x^7 + f_6x^6 + f_5x^5 + f_4x^4$ $+f_3x^3 + f_2x^2 + f_1x + f_0$;
Output: $D_3 = (u_3, v_3) = 2D_1$ with $u_3 = x^3 + u_{32}x^2 + u_{31}x + u_{30}$; $v_3 = v_{32}x^2 + v_{31}x + v_{30}$;
$R_{00} \leftarrow u_{12}$
$R_{01} \leftarrow u_{11}$
$R_{02} \leftarrow u_{10}$
$R_{06} \leftarrow h_1 \times R_{00}$ $R_{07} \leftarrow h_1 \times R_{02}$ *
$R_{08} \leftarrow h_2 \times R_{01}$ $R_{09} \leftarrow h_2 \times R_{02}$ *
$R_{10} \leftarrow h_0 \times R_{01}$ $R_{11} \leftarrow h_0 \times R_{00}$ *
$R_{05} \leftarrow R_{02} + h_0$ $R_{04} \leftarrow R_{01} + h_1$
$R_{02} \leftarrow R_{05} \times R_{05}$ $R_{12} \leftarrow R_{05} \times R_{04}$ *
$R_{04} \leftarrow R_{04} \times R_{04}$
$R_{03} \leftarrow R_{00} + h_2$ $R_{10} \leftarrow R_{10} + R_{07}$
$R_{13} \leftarrow R_{10} \times R_{04}$ $R_{10} \leftarrow R_{10} \times R_{03}$ *
$R_{05} \leftarrow R_{05} + R_{06}$ $R_{11} \leftarrow R_{11} + R_{09}$
$R_{05} \leftarrow R_{05} + R_{08}$ $R_{02} \leftarrow R_{02} + R_{10}$
$R_{08} \leftarrow R_{03} \times R_{11}$
$R_{11} \leftarrow R_{08} \times R_{11}$
$R_{03} \leftarrow R_{05} \times R_{03}$ $R_{05} \leftarrow R_{05} \times R_{02}$ *
$R_{05} \leftarrow R_{05} + R_{11}$ $R_{04} \leftarrow R_{04} + R_{03}$
$R_{05} \leftarrow R_{05} + R_{13}$ $R_{08} \leftarrow R_{08} + R_{12}$
$R_{03} \leftarrow R_{04} \times R_{01}$ $R_{01} \leftarrow R_{04} \times R_{00}$ *
$R_{01} \leftarrow R_{01} + R_{08}$ $R_{03} \leftarrow R_{03} + R_{02}$
$R_{08} \leftarrow R_{08} \times R_{00}$
$R_{03} \leftarrow R_{03} + R_{08}$
$R_{00} \leftarrow u_{12}$
$R_{02} \leftarrow u_{11}$
$R_{06} \leftarrow u_{10}$
$R_{07} \leftarrow v_{12}$
$R_{08} \leftarrow v_{11}$
$R_{09} \leftarrow v_{10}$
$R_{10} \leftarrow R_{07} \times R_{07}$ $R_{11} \leftarrow R_{07} \times h_1$ *
$R_{12} \leftarrow R_{00} + f_6$ $R_{13} \leftarrow R_{07} + f_5$
$R_{14} \leftarrow R_{12} \times R_{02}$ $R_{15} \leftarrow R_{12} \times R_{00}$ *
$R_{13} \leftarrow R_{13} + R_{15}$ $R_{15} \leftarrow R_{10} + f_4$
$R_{13} \leftarrow R_{13} + R_{02}$ $R_{15} \leftarrow R_{15} + R_{08}$
$R_{08} \leftarrow h_2 \times R_{08}$ $R_{10} \leftarrow h_2 \times R_{07}$ *
$R_{14} \leftarrow R_{14} + R_{06}$ $R_{08} \leftarrow R_{08} + R_{09}$
$R_{15} \leftarrow R_{15} + R_{14}$ $R_{07} \leftarrow R_{07} + f_5$
$R_{09} \leftarrow R_{13} \times R_{00}$ $R_{13} \leftarrow R_{13} \times R_{02}$ *
$R_{15} \leftarrow R_{15} + R_{10}$
$R_{15} \leftarrow R_{15} + R_{09}$ $R_{08} \leftarrow R_{08} + R_{13}$
$R_{14} \leftarrow R_{14} + R_{15}$ $R_{08} \leftarrow R_{08} + f_3$
$R_{12} \leftarrow R_{12} \times R_{06}$ $R_{15} \leftarrow R_{15} \times R_{00}$
$R_{02} \leftarrow R_{00} \times R_{02}$ $R_{00} \leftarrow R_{00} \times R_{00}$ *
$R_{08} \leftarrow R_{08} + R_{12}$ $R_{11} \leftarrow R_{11} + R_{15}$
$R_{07} \leftarrow R_{07} + R_{00}$ $R_{14} \leftarrow R_{14} + R_{02}$
$R_{08} \leftarrow R_{08} + R_{11}$
$R_{00} \leftarrow u_{12}$
$R_{02} \leftarrow u_{11}$
$R_{06} \leftarrow u_{10}$
$R_{09} \leftarrow R_{01} + R_{04}$ $R_{10} \leftarrow R_{14} + R_{07}$
$R_{09} \leftarrow R_{09} \times R_{10}$ $R_{14} \leftarrow R_{14} \times R_{01}$
$R_{11} \leftarrow R_{03} + R_{04}$ $R_{12} \leftarrow R_{08} + R_{07}$
$R_{11} \leftarrow R_{11} \times R_{12}$ $R_{08} \leftarrow R_{08} \times R_{03}$
$R_{13} \leftarrow R_{03} + R_{01}$ $R_{10} \leftarrow R_{10} + R_{12}$
$R_{13} \leftarrow R_{13} \times R_{10}$ $R_{07} \leftarrow R_{07} \times R_{04}$
$R_{13} \leftarrow R_{13} + R_{14}$ $R_{09} \leftarrow R_{09} + R_{14}$
$R_{11} \leftarrow R_{11} + R_{14}$ $R_{12} \leftarrow R_{08} + R_{07}$
$R_{13} \leftarrow R_{13} + R_{08}$ $R_{09} \leftarrow R_{09} + R_{07}$
$R_{01} \leftarrow R_{07} \times R_{00}$ $R_{03} \leftarrow R_{07} \times R_{02}$ *
$R_{01} \leftarrow R_{01} + R_{09}$ $R_{13} \leftarrow R_{13} + R_{03}$
$R_{07} \leftarrow R_{07} + R_{01}$ $R_{02} \leftarrow R_{02} + R_{06}$
$R_{06} \leftarrow R_{01} \times R_{06}$ $R_{00} \leftarrow R_{01} \times R_{00}$ *
$R_{08} \leftarrow R_{08} + R_{06}$ $R_{00} \leftarrow R_{00} + R_{03}$
$R_{02} \leftarrow R_{02} \times R_{07}$
$R_{11} \leftarrow R_{11} + R_{12}$ $R_{13} \leftarrow R_{13} + R_{02}$
$R_{13} \leftarrow R_{13} + R_{06}$ $R_{11} \leftarrow R_{11} + R_{00}$
$R_{01} \leftarrow R_{11} \times R_{05}$ $R_{06} \leftarrow R_{11} \times R_{11}$ *
$R_{01} \leftarrow 1/R_{01}$
$R_{03} \leftarrow R_{01} \times R_{05}$ $R_{06} \leftarrow R_{01} \times R_{06}$ *
$R_{05} \leftarrow R_{05} \times R_{03}$
$R_{00} \leftarrow R_{03} \times R_{08}$ $R_{01} \leftarrow R_{03} \times R_{13}$ *
$R_{02} \leftarrow u_{12}$
$R_{04} \leftarrow u_{11}$
$R_{10} \leftarrow u_{10}$
$R_{08} \leftarrow R_{02} \times R_{00}$ $R_{09} \leftarrow R_{02} \times R_{01}$ *
$R_{02} \leftarrow R_{02} + R_{01}$ $R_{09} \leftarrow R_{09} + R_{04}$
$R_{07} \leftarrow R_{04} \times R_{00}$ $R_{04} \leftarrow R_{04} \times R_{01}$ *
$R_{08} \leftarrow R_{08} + R_{04}$ $R_{09} \leftarrow R_{09} + R_{00}$
$R_{11} \leftarrow R_{10} \times R_{00}$ $R_{12} \leftarrow R_{10} \times R_{01}$ *
$R_{07} \leftarrow R_{07} + R_{12}$ $R_{08} \leftarrow R_{08} + R_{10}$
$R_{04} \leftarrow u_{12}$
$R_{14} \leftarrow u_{11}$
$R_{10} \leftarrow R_{04} + h_2$ $R_{12} \leftarrow R_{00} + h_1$
$R_{10} \leftarrow R_{10} + R_{01}$ $R_{12} \leftarrow R_{12} + R_{14}$
$R_{04} \leftarrow R_{10} \times R_{04}$ $R_{10} \leftarrow R_{10} \times R_{05}$ *
$R_{13} \leftarrow R_{01} \times h_2$ $R_{01} \leftarrow R_{01} \times R_{01}$ *
$R_{12} \leftarrow R_{12} + R_{04}$ $R_{01} \leftarrow R_{01} + R_{05}$
$R_{12} \leftarrow R_{12} + R_{13}$
$R_{12} \leftarrow R_{05} \times R_{12}$ $R_{05} \leftarrow R_{05} \times R_{05}$ *
$R_{03} \leftarrow R_{05} \times f_6$ $R_{00} \leftarrow R_{00} \times R_{00}$
$R_{10} \leftarrow R_{10} + R_{05}$ $R_{03} \leftarrow R_{03} + R_{00}$

Table 5. Parallel arithmetic of HECDBL for genus three – continued

$R_{12} \leftarrow R_{12} + R_{03}$		
$R_{03} \leftarrow v_{12}$		
$R_{04} \leftarrow v_{11}$		
$R_{05} \leftarrow v_{10}$		
$R_{09} \leftarrow R_{09} + R_{01}$	$R_{08} \leftarrow R_{08} + R_{10}$	
$R_{00} \leftarrow R_{02} \times R_{01}$	$R_{13} \leftarrow R_{02} \times R_{10}$	*
$R_{14} \leftarrow R_{02} \times R_{12}$		
$R_{12} \leftarrow R_{12} + R_{07}$	$R_{14} \leftarrow R_{14} + R_{11}$	
$R_{00} \leftarrow R_{00} + R_{08}$	$R_{13} \leftarrow R_{13} + R_{12}$	
$R_{00} \leftarrow R_{06} \times R_{00}$	$R_{13} \leftarrow R_{06} \times R_{13}$	*
$R_{14} \leftarrow R_{06} \times R_{14}$	$R_{15} \leftarrow R_{06} \times R_{09}$	*
$R_{15} \leftarrow R_{15} + 1$	$R_{00} \leftarrow R_{00} + h_2$	
$R_{13} \leftarrow R_{13} + h_1$	$R_{14} \leftarrow R_{14} + h_0$	
$R_{00} \leftarrow R_{00} + R_{03}$	$R_{13} \leftarrow R_{13} + R_{04}$	
$R_{14} \leftarrow R_{14} + R_{05}$	$R_{04} \leftarrow R_{15} + f_6$	
$R_{02} \leftarrow R_{15} \times R_{15}$	$R_{03} \leftarrow R_{15} \times h_1$	*
$R_{03} \leftarrow R_{03} + R_{10}$	$R_{02} \leftarrow R_{02} + R_{04}$	
$R_{04} \leftarrow R_{00} \times R_{00}$	$R_{05} \leftarrow R_{00} \times h_2$	*
$R_{03} \leftarrow R_{03} + R_{04}$	$R_{06} \leftarrow R_{01} + f_5$	
$R_{03} \leftarrow R_{03} + f_4$	$R_{05} \leftarrow R_{05} + R_{13}$	
$R_{07} \leftarrow R_{15} \times h_2$		
$R_{06} \leftarrow R_{06} + R_{07}$	$R_{03} \leftarrow R_{03} + R_{05}$	
$R_{15} \leftarrow R_{15} + 1$	$R_{06} \leftarrow R_{06} + R_{00}$	
$R_{01} \leftarrow R_{02} \times R_{01}$	$R_{04} \leftarrow R_{02} \times R_{15}$	*
$R_{03} \leftarrow R_{03} + R_{01}$	$R_{04} \leftarrow R_{04} + R_{00}$	
$R_{05} \leftarrow R_{15} \times R_{06}$	$R_{07} \leftarrow R_{15} \times R_{03}$	*
$R_{05} \leftarrow R_{05} + R_{13}$	$R_{07} \leftarrow R_{07} + R_{14}$	
$R_{04} \leftarrow R_{04} + h_2$	$R_{05} \leftarrow R_{05} + h_1$	
$R_{07} \leftarrow R_{07} + h_0$		
$u_{32} \leftarrow R_{02}$		
$u_{31} \leftarrow R_{06}$		
$u_{30} \leftarrow R_{03}$		
$v_{32} \leftarrow R_{04}$		
$v_{31} \leftarrow R_{05}$		
$v_{30} \leftarrow R_{07}$		