

Fast Arithmetic In Jacobian Of Hyperelliptic Curves Of Genus 2 Over $\text{GF}(p)$

V. Kovtun¹ and J. Pelzl²

¹Senior researcher, Kharkiv Air Force University, Ukraine
E-mail: vladislav.kovtun@gmail.com

²Chief Technology Officer, Escrypt GmbH, Germany
E-mail: jpelzl@escrypt.com

Abstract

In this paper, we suggest a new fast transformation for a divisor addition for hyperelliptic curves. The transformation targets the Jacobian of genus-2 curves over odd characteristic fields in projective representation. Compared to previously published results, the modification reduces the computational complexity and makes hyperelliptic curves more attractive for applications.

Introduction

Cryptographic systems have become an integral element in a wide spectrum of modern information and telecommunication systems. Nowadays, bulks of circulating information have to be processed by efficient ITS clients. With the extensive growth of computational power and mathematical methods for cryptanalysis, the requirements to existing and perspective cryptosystems are challenging. In other words, the main requirement when developing a cryptosystem is the capability to assure a required security level, taking recent development in cryptanalysis into account.

It is widely agreed that new cryptosystems such as elliptic curve cryptosystems (ECC) and hyperelliptic curve cryptosystems (HECC) will very likely dominate applications on constraint devices in the future. This work is another step towards an efficient HECC.

The challenge to develop practical systems is dominated by the difficulty of its efficient implementation on modern hardware such as, e.g. embedded systems. The efficiency of a cryptosystem is directly related to the public key primitives that form its basis.

Important primitives use transformations in rings and fields. More recently, transformations of the group of points on an elliptic curve (EC) have been widely used and are to be found in various international and government standards (ISO 2002, IEEE 2000, DSTU 2002). Such transformations allow for building a cryptosystem with a high security level at quite low computational costs.

Further development of cryptographic transformations using algebraic curves implies applications of even more complex curves – hyperelliptic curves (HEC) – which increases the computational complexity (hereinafter complexity) of such systems. Therefore, papers dealing with the decrease of complexity of prospective cryptosystems are of extreme importance.

Cryptosystems based on HEC are dominated by operations over reduced divisors, in particularly by scalar multiplications of reduced divisors (Koblitz 1989) with its corresponding basic operations of addition and doubling divisors.

The significant decrease in the complexity of calculations based on divisor transformations in the Jacobian of a HEC can be achieved on account of the decrease in complexity of a divisor scalar multiplication algorithm. The earliest publications dedicated to the arithmetic in the Jacobian of HEC are due to (Cantor 1987, Lange 2002a) and are rather of theoretical interest.

Recently, multiple scientists done intense research in the effective implementation of the arithmetic in the Jacobian of HEC (Cantor 1987, Koblitz 1989, Spallek 1994, Harley 2000, Kriger 2001, Miyamoto et al. 2002, Lange 2001, 2002a, 2002b, 2002c, Takahashi 2002, Suguzaki 2002). In these contributions, the attention is particularly paid to the reduction of field operations by applying HEC with fixed genus. This allows various modifications of the arithmetic in the Jacobian of HEC.

Papers (Koblitz 1989, Kriger 2001) deal with methods of addition and doubling of divisors in the Jacobian of genus 2 HEC. The first practical implementation of these methods was described in (Harley 2000). In

(Wollinger 2004), generalized results (Harley 2000) for curves over the even characteristic fields are shown. The development of addition and doubling methods is described in (IEEE 2000, Cantor 1987, Harley 2000). The time required for a scalar multiplication in the Jacobian of genus 2 HEC as the scalar multiplication in the EC curve is main part of discrete logarithm based cryptosystems. Under this circumstance, the challenge of improving the performance of a cryptosystem and, in particular, a divisor scalar multiplication in the Jacobian of HEC becomes of special urgency.

Curves of genus 2 are of our main interest when decreasing the arithmetic complexity in the Jacobian of HEC. Thus, we will take genus-2 curves into our further consideration.

As it was proved in (Lange 2002b, Hankerson et al. 2000), the divisor addition and doubling operations in the Jacobian of HEC perform a very complex field operation – the field inversion. According to publications Hankerson et al. (2000), Brown et al. (2000), for $\mathbf{GF}(p)$, the complexity of a field inversion I depends on the actual platform and lies in the interval $(80M, 90M)$ Brown et al. (2000), where M is the complexity of field multiplication.

At publication (Miyamoto et al. 2002) proposes for the first time an approach to implement the arithmetic in a Jacobian of HEC of genus 2 without using any field inversion in intermediate computations. The further development of the proposed approach was shown in papers (Lange 2002b, Lange 2002c), where the results were improved and extended to a wider class of HEC over even characteristic fields (Lange 2002b, Lange 2002c). As a prototype for practicable methods under discussion, the results of (Lange 2002b, Lange 2002c) are considered.

Mathematical Background

In this paper, we are considering genus-2 HEC: $v^2 + h(u)v = f(u)$ over field $\mathbf{GF}(p)$ with odd characteristic, where $h(x) = 0$, $f(x) = x^5 + f_3x^3 + f_2x^2 + f_1x + f_0$, $f_i \in \mathbf{GF}(p)$.

The divisor representation in Mumford form is given as $[u, v]$, $u(x) = x^2 + u_1x + u_0$, $v(x) = v_1x + v_0$, $\deg v < \deg u \leq 2$ and will be denominated as affine representation. The representation that does not involve field inversion will be denominated as projective. In our case, the divisor in projective representation is given by $[u, v]$, $u(x) = x^2 + U_1/Z x + U_0/Z$, $v(x) = V_1/Z x + V_0/Z$, and is represented as $[U_1, U_0, V_1, V_0, Z]$ (Miyamoto et al. 2002, Lange 2002b). Where the divisor in weighted representation $[u, v]$, $u(x) = x^2 + U_1/Z_1^2 x + U_0/Z_1^2$, $v(x) = V_1/Z_1^3 Z_2 x + V_0/Z_1^3 Z_2$, is represented as $[U_1, U_0, V_1, V_0, Z_1, Z_2]$ (Lange 2002b).

The aim of this paper is to develop a modified method of the arithmetic in the Jacobian of genus 2 HEC in the projective representation with the purpose to increase the performance of the scalar multiplication in a HECC.

Under the accepted model (Lange 2002a, Harley 2000), a typical addition of divisors is given by the addition of the divisors $[u_1(x), v_1(x)]$ and $[u_2(x), v_2(x)]$, where the resultant $r(u_1(x), u_2(x))$ is not equal to zero, and under doubling of divisor $[u_1(x), v_1(x)]$, where the resultant $r(u_1(x), h(x) + 2v_1(x))$ is not equal to zero.

The proposed modification that provides a reduced complexity is based on Harley's method (Harley 2000) and its modification (Lange 2002a, Wollinger 2004). In this context, we suggest to use the projective representation of divisors in the method being described.

In the algorithms of addition and doubling (Harley 2000, Wollinger 2004), the most difficult parts in terms of computational complexity are operations in the polynomial function ring: division, inversion, reduction, and multiplication.

Proposed Efficient Arithmetic In Jacobians Of HEC Over $\mathbf{GF}(p)$

To decrease these operations, we propose to modify the addition and doubling algorithms as follows:
 pass directly from operations in the polynomial function ring to the field operations using HEC with fixed and small genus (i.e., 2) (Spallek 1994, Harley 2000);
 simplify the arithmetic in the polynomial function ring by polynomial normalizations;

- normalize and minimize the Hamming weight of HEC parameters $h(x)$ and $f(x)$. These parameters set up a special kind of HEC (Miyamoto et al. 2002, Wollinger 2004);
- normalize polynomial function $u(x)$, steps 3 and 4 of algorithms 1, 2; steps 4 and 5 of algorithms 3, 4 (Miyamoto et al. 2002, Wollinger 2004);
- simultaneously invert several field elements with the Montgomery trick (Lange 2002b, Wollinger 2004);
- multiply polynomial functions of different powers with the Karatsuba method, step 5 of algorithms 1, 2; step 6 of algorithms 3, 4 (Wollinger 2004);
- reduce polynomial functions by the Karatsuba method; step 3 of algorithms 1, 2; step 4 of algorithms 3, 4 (Harley 2000);
- exclude multiplicative field inversion by using the projective representation of divisors; step 2 of algorithms 1, 2, 3, 4 (Lange 2002b, Miyamoto et al. 2002).

Using the above proposed modifications, we obtain arithmetic algorithms on HEC defined by the equation $v^2 + h(u)v = f(u)$ over field $\mathbf{GF}(p)$ with odd characteristic in projective representation, where

$f(x) = x^5 + f_3x^3 + f_2x^2 + f_1x + f_0$, $f_i \in \mathbf{GF}(p)$, $h(x) = 0$. The proposed addition algorithm is described in Algorithm 1.

Particularly, with the algorithm of addition, there often arises a situation where one of the input divisors is affine (Z is equal to 1), and the other one is projective. The result of addition is obtained in the projective representation. For Algorithm 1, this input data allows for its simplification to Algorithm 2 which comes with a decreased number of field operations and, hence, is of decreased complexity.

Algorithm 1. Addition of divisors

Input: $\text{div}(U_{11}, U_{10}, V_{11}, V_{10}, Z_1)$, $\text{div}(U_{21}, U_{20}, V_{21}, V_{20}, Z_2)$

Output: $\text{div}(U'_1, U'_0, V'_1, V'_2, Z') = \text{div}(U_{11}, U_{10}, V_{11}, V_{10}, Z_1) + \text{div}(U_{21}, U_{20}, V_{21}, V_{20}, Z_2)$

	Operation	Cost
1	Compute resultant $r: Z = Z_1 \cdot Z_2$, $\tilde{U}_{21} = Z_1 \cdot U_{21}$, $\tilde{U}_{20} = Z_1 \cdot U_{20}$, $\tilde{V}_{21} = Z_1 \cdot V_{21}$, $\tilde{V}_{20} = Z_1 \cdot V_{20}$, $y_1 = U_{11} \cdot Z_2 - \tilde{U}_{21}$, $y_2 = \tilde{U}_{20} - U_{10} \cdot Z_2$, $y_3 = U_{11} \cdot y_1 + y_2 \cdot Z_1$, $r = y_2 \cdot y_3 + y_1^2 \cdot U_{10}$.	1S, 11M
2	Compute almost inversion $\text{inv} = r/u_2 \bmod u_1: \text{inv}_1 = y_1$, $\text{inv}_0 = y_3$.	
3	Compute $s = (v_1 - v_2)\text{inv} \bmod u_1: w_0 = V_{10} \cdot Z_2 - \tilde{V}_{20}$, $w_1 = V_{11} \cdot Z_2 - \tilde{V}_{21}$, $w_2 = \text{inv}_0 \cdot w_0$, $w_3 = \text{inv}_1 \cdot w_1$, $s_1 = (\text{inv}_0 + Z_1 \cdot \text{inv}_1) \cdot (w_0 + w_1) - w_2 - w_3 \cdot (Z_1 + U_{11})$, $s_0 = w_2 - U_{10} \cdot w_3$. If $s_1 = 0$ then consider special case.	8M
4	Precomputation: $R = r \cdot Z$, $s_2 = s_0 \cdot Z$, $s_3 = s_1 \cdot Z$, $\tilde{R} = R \cdot s_3$, $w_0 = s_1 \cdot s_0$, $w_1 = s_1 \cdot s_3$, $w_2 = s_0 \cdot s_3$, $w_3 = w_1 \cdot \tilde{U}_{21}$, $w_4 = R \cdot s_1$	9M
5	Compute $l = su_2: l_0 = w_0 \cdot \tilde{U}_{20}$, $l_2 = w_3 + w_2$, $l_1 = (w_1 + w_0) \cdot (\tilde{U}_{21} + \tilde{U}_{20}) - l_0 - w_3$	2M
6	Compute $u' = (s(l + 2v_1) - k)u_1^{-1}$, $k = (f - v_1^2)/u_1: \tilde{U}'_1 = 2w_2 - s_3 \cdot s_1 y_1 - R^2$, $\tilde{U}'_0 = s_2^2 + s_1 \cdot y_1 \cdot (s_1 \cdot \tilde{U}_{11} - 2s_2) + y_2 \cdot w_1 + 2w_4 \cdot \tilde{V}_{21} + R \cdot r \cdot (y_1 + 2\tilde{U}_{21})$	2S, 8M
7	Correction: $U'_0 = \tilde{U}'_0 \cdot \tilde{R}$, $U'_1 = \tilde{U}'_1 \cdot \tilde{R}$, $Z' = s_3^2 \cdot \tilde{R}$	1S, 3M
8	Compute $v' \equiv -(s_1 l + v_2) \bmod u': V'_1 = \tilde{U}'_1 \cdot (l_2 - \tilde{U}'_1) + s_3^2 \cdot (\tilde{U}'_0 - w_4 \tilde{V}_{21} - l_1)$, $V'_0 = \tilde{U}'_0 \cdot (l_2 - \tilde{U}'_1) - s_3^2 \cdot (l_0 + w_4 \cdot \tilde{V}_{20})$	5M
		4S, 46M

Algorithm 2. Mixed addition of divisors

Input: $\text{div}(U_{11}, U_{10}, V_{11}, V_{10}, 1)$, $\text{div}(U_{21}, U_{20}, V_{21}, V_{20}, Z_2)$

Output: $\text{div}(U'_1, U'_0, V'_1, V'_2, Z') = \text{div}(U_{11}, U_{10}, V_{11}, V_{10}, 1) + \text{div}(U_{21}, U_{20}, V_{21}, V_{20}, Z_2)$

	Operation	Cost
--	-----------	------

1	Compute resultant r for u_1 and u_2 : $\tilde{U}_{11} = Z_2 \cdot U_{11}$, $y_1 = \tilde{U}_{11} - U_{21}$, $y_2 = U_{20} - U_{10} \cdot Z_2$, $y_3 = U_{11} \cdot y_1 + y_2$, $r = y_2 \cdot y_3 + y_1^2 \cdot U_{10}$	1S, 5M
2	Compute almost inversion $inv = r/u_2 \bmod u_1$: $inv_1 = y_1$, $inv_0 = y_3$	
3	Compute $s = (v_1 - v_2)inv \bmod u_1$: $w_0 = V_{10} \cdot Z_2 - V_{20}$, $w_1 = V_{11} \cdot Z_2 - V_{21}$, $w_2 = inv_0 \cdot w_0$, $w_3 = inv_1 \cdot w_1$, $s_0 = w_2 - U_{10} \cdot w_3$, $s_1 = (inv_0 + inv_1) \cdot (w_0 + w_1) - w_2 - w_3 \cdot (U_{11} + 1)$. If $s_1 = 0$ then consider special case	7M
4	Precomputation: $R = r \cdot Z_2$, $s_2 = s_0 \cdot Z_2$, $s_3 = s_1 \cdot Z_2$, $\tilde{R} = R \cdot s_3$, $w_0 = s_1 \cdot s_0$, $w_1 = s_1 \cdot s_3$, $w_2 = s_0 \cdot s_3$, $w_3 = w_1 \cdot U_{21}$, $w_4 = R \cdot s_1$.	9M
5	Compute $l = su_2$: $l_0 = w_0 \cdot U_{20}$, $l_2 = w_3 + w_2$, $l_1 = (w_1 + w_0) \cdot (U_{21} + U_{20}) - l_0 - w_3$	2M
6	Compute $u' = (s(l + 2v_1) - k)u_1^{-1}$, $k = (f - v_1^2)/u_1$: $\tilde{U}'_1 = 2w_2 - s_3 \cdot s_1 y_1 - R^2$, $\tilde{U}'_0 = s_2^2 + s_1 \cdot y_1 \cdot (s_1 \cdot \tilde{U}_{11} - 2s_2) + y_2 \cdot w_1 + 2w_4 \cdot V_{21} + R \cdot r \cdot (y_1 + 2U_{21})$	2S, 8M
7	Correction: $U'_0 = \tilde{U}'_0 \cdot \tilde{R}$, $U'_1 = \tilde{U}'_1 \cdot \tilde{R}$, $Z' = s_3^2 \cdot \tilde{R}$	1S, 3M
8	Compute $v' \equiv -(h + s_1 l + v_2) \bmod u'$: $V'_1 = \tilde{U}'_1 \cdot (l_2 - \tilde{U}'_1) + s_3^2 \cdot (\tilde{U}'_0 - w_4 V_{21} - l_1)$, $V'_0 = \tilde{U}'_0 \cdot (l_2 - \tilde{U}'_1) - s_3^2 \cdot (l_0 + w_4 \cdot V_{20})$	5M
		4S, 39M

Similarly, with the algorithm of doubling, there often arises a situation when the input divisor is affine (Z is equal to 1). The result of doubling is produced in projective representation. For Algorithm 3, this input data allows for its simplification to Algorithm 4 which incorporates a decreased number of field operations and which is of decreased complexity.

Algorithm 3. Doubling of divisor

Input: $\text{div}(U_1, U_0, V_1, V_0, Z)$

Output: $\text{div}(U'_1, U'_0, V'_1, V'_2, Z') = 2 \text{div}(U_1, U_0, V_1, V_0, Z)$

	Operation	Cost
1	Compute resultant r for u and $2v$ (where $\tilde{v} \equiv (2v) \bmod u$): $Z_2 = Z^2$, $\tilde{V}_1 = 2V_1$, $\tilde{V}_0 = 2V_0$, $w_0 = V_1^2$, $w_1 = U_1^2$, $w_2 = \tilde{V}_1^2 = 4w_0$, $w_3 = \tilde{V}_0 \cdot Z - U_1 \cdot \tilde{V}_1$, $r = \tilde{V}_0 \cdot w_3 + w_2 \cdot U_0$.	3S, 4M
2	Compute almost inversion $inv \equiv r/\tilde{v} \bmod u$: $inv_1 = -\tilde{V}_1$, $inv_0 = w_3$.	
3	Compute $k \equiv [(f - v^2)/u] \bmod u$: $w_3 = f_3 \cdot Z + w_1$, $k_1 = 2w_1 + w_3 - Z \cdot 2U_0$, $k_0 = U_1 \cdot (4ZU_0 - w_3) + Z \cdot (f_2 \cdot Z - w_0)$.	5M
4	Compute $s = k \cdot inv \bmod u$: $w_0 = k_0 \cdot inv_0$, $w_1 = k_1 \cdot inv_1$, $s_0 = w_0 - ZU_0 \cdot w_1$, $s_3 = (inv_0 + inv_1) \cdot (k_0 + k_1) - w_0 - w_1 \cdot (1 + U_1)$, $s_1 = s_3 \cdot Z$. If $s_1 = 0$ then consider special case.	6M
5	Precomputation: $R = r \cdot Z_2$, $\tilde{R} = R \cdot s_1$, $w_0 = s_1 \cdot s_3$, $w_1 = s_0 \cdot s_3$, $w_3 = w_1 \cdot Z$, $w_4 = R \cdot s_3$.	6M
6	Compute $l = su$, $l = l_2 x^2 + l_1 x + l_0$: $l_0 = U_0 \cdot w_1$, $l_2 = U_1 \cdot w_0$, $l_1 = (w_1 + w_0) \cdot (U_1 + U_0) - l_0 - l_2$.	3M
7	Compute $u' = [l^2 + \frac{1}{s} l 2v - \frac{1}{s^2} (f - v^2)]/u^2$: $\tilde{U}'_0 = s_0^2 + 2w_4 \cdot V_1 + R \cdot r \cdot 2U_1$, $\tilde{U}'_1 = 2w_3 - R^2$.	2S, 2M
8	Correction: $U'_0 = \tilde{U}'_0 \cdot \tilde{R}$, $U'_1 = \tilde{U}'_1 \cdot \tilde{R}$, $Z' = s_1^2 \cdot \tilde{R}$.	1S, 3M
9	Compute $v' \equiv -(s_1 l + v_2) \bmod u'$: $V'_1 = \tilde{U}'_1 \cdot (l_2 - \tilde{U}'_1 + w_3) + s_1^2 \cdot (\tilde{U}'_0 - w_4 V_1 - l_1)$, $V'_0 = \tilde{U}'_0 \cdot (l_2 - \tilde{U}'_1 + w_3) - s_1^2 \cdot (l_0 + w_4 \cdot V_0)$.	5M
		6S, 35M

Algorithm 4. Mixed doubling of divisor

Input: $\text{div}(U_1, U_0, V_1, V_0, 1)$

Output: $\text{div}(U'_1, U'_0, V'_1, V'_2, Z') = 2 \text{div}(U_1, U_0, V_1, V_0, 1)$

	Operation	Cost
1	Compute resultant r for u and $2v$ (where $\tilde{v} \equiv (2v) \bmod u$): $\tilde{V}_1 = 2V_1, \tilde{V}_0 = 2V_0, w_0 = V_1^2, w_1 = U_1^2, w_2 = \tilde{V}_1^2 = 4w_0, w_3 = \tilde{V}_0 - U_1 \cdot \tilde{V}_1, r = \tilde{V}_0 \cdot w_3 + w_2 \cdot U_0$.	2S, 3M
2	Compute almost inversion $\text{inv} \equiv r/\tilde{v} \bmod u$: $\text{inv}_1 = -\tilde{V}_1, \text{inv}_0 = w_3$.	
3	Compute $k \equiv [(f - v^2)/u] \bmod u$: $w_3 = f_3 + w_1, w_4 = 2U_0, k_1 = 2w_1 + w_3 - w_4, k_0 = U_1 \cdot (2w_4 - w_3) + f_2 - w_0$.	1M
4	Compute $s = k \cdot \text{inv} \bmod u$: $w_0 = k_0 \cdot \text{inv}_0, w_1 = k_1 \cdot \text{inv}_1, s_0 = w_0 - U_0 \cdot w_1, s_1 = (\text{inv}_0 + \text{inv}_1) \cdot (k_0 + k_1) - w_0 - w_1 \cdot (1 + U_1)$. If $s_1 = 0$ then consider special case.	5M
5	Precomputation: $\tilde{R} = r \cdot s_1, w_0 = s_1^2, w_1 = s_0 \cdot s_1$.	1S, 2M
6	Compute $l = su$: $l_0 = U_0 \cdot w_1, l_2 = U_1 \cdot w_0, l_1 = (w_1 + w_0) \cdot (U_1 + U_0) - l_0 - l_2$.	3M
7	Compute $u' = [l^2 + \frac{1}{s}l2v - \frac{1}{s^2}(f - v^2)]/u^2$: $\tilde{U}'_0 = s_0^2 + 2\tilde{R} \cdot V_1 + r^2 \cdot 2U_1, \tilde{U}'_1 = 2w_1 - r^2$.	2S, 2M
8	Correction: $U'_0 = \tilde{U}'_0 \cdot \tilde{R}, U'_1 = \tilde{U}'_1 \cdot \tilde{R}, Z' = w_0 \cdot \tilde{R}$.	3M
9	Compute $v' \equiv -(s_1l + v_2) \bmod u'$: $V'_1 = \tilde{U}'_1 \cdot (l_2 - \tilde{U}'_1 + w_1) + w_0 \cdot (\tilde{U}'_0 - \tilde{R}V_1 - l_1), V'_0 = \tilde{U}'_0 \cdot (l_2 - \tilde{U}'_1 + w_1) - w_0 \cdot (l_0 + \tilde{R} \cdot V_0)$.	5M
		5S, 24M

Computational Complexity

Table 1 summarizes estimates of the computational complexity for the known arithmetic (Miyamoto et al. 2002, Lange 2002a, Lange 2002b, Wollinger 2004) and the proposed Algorithm 1 to Algorithm 4 in the Jacobian of genus-2 HEC for common cases. The computational complexity is evaluated in terms of field operations.

Table 1. Summarized estimates of computational complexity of arithmetic in the Jacobian of HEC

Parameters of genus 2 HEC	Algorithm											
	Addition			Mixed addition			Doubling			Mixed doubling		
	$()^{-1}$	\wedge^2	*	$()^{-1}$	\wedge^2	*	$()^{-1}$	\wedge^2	*	$()^{-1}$	\wedge^2	*
Odd characteristic fields												
Affine coordinates												
$f_4 = 0$ (Lange 2002a)	1	3	22				1	5	22			
Projective coordinates $[U_1, U_0, V_1, V_0, Z]$												
$\deg(h) = 2, h_i \in \mathbf{F}_2$ (Lange 2002b)		4	47		3	40		6	40		5	25
$\deg(h) = 2, h_i \in \mathbf{F}_2$ (Kovtun, Wollinger 2007)		4	46		4	39		6	39		5	25
$h(x) = 0, f_4 = 0$ [proposed]		4	46		4	39		6	35		5	24
Weighted coordinates $[U_1, U_0, V_1, V_0, Z_1, Z_2, Z_1^2, Z_2^2]$												
$h(x) = 0, f_4 = 0$ (Lange 2002c)		7	47		5	36		7	34		5	21
Even characteristic fields												
Affine coordinates												
$f_4 = 0$ (Lange 2002a)	1	3	21				1	5	20			

$h_2 = 0, f_4 = 0$ (Lange 2002a)	1	3	21				1	5	17			
$h(x) = x, f_4 = 0, f_3 = f_2 = 0$ (Wollinger 2004)							1	6	9			
Projective coordinates $[U_1, U_0, V_1, V_0, Z]$												
$h(x) = x, f_4 = 0, f_3 = f_2 = 0$ (Wollinger 2004)		5	45		5	38		6	31		5	18
$h(x) = x, f_4 = 0, f_3 = f_2 = 0$ (Kovtun, Wollinger 2004)		4	44		4	37		6	30		4	17
Weighted coordinates $[U_1, U_0, V_1, V_0, Z_1, Z_2, Z_1^2, Z_2^2, Z_1Z_2, Z_1^3Z_2]$												
$f_4 = 0, h_2 \neq 0$ (Lange 200c)		4	46		5	35		6	35		5	24
$f_4 = 0, h_2 = 0$ (Lange 200c)		6	44		6	34		6	29		6	19

From Table 1, one can see that for $\mathbf{GF}(p)$ the proposed algorithms show a decreased complexity in comparison to (Lange 2002b, Wollinger 2004).

Conclusions

We developed a modification of the arithmetic in the Jacobian of genus-2 HEC in projective coordinates. The new formulas have a lower complexity in comparison to the existing algorithms (Lange 2002a, Wollinger 2004). The modification is characterized as follows:

- a decreased number of recomputed values due to pre-computations and reordering in comparison to (Harley 2000, Lange 2002b, Wollinger 2004);
- an increased number of pre-computed values due to reordering of field operations in comparison with (Harley 2000, Lange 2002b, Wollinger 2004);
- an increased number of pre-computed values due to the use of dependencies among resulting polynomial functions.
- a decreased Hamming weight of HEC parameters in analogy to (Lange 2002b).

The proposed modifications of the arithmetic in the Jacobian of genus 2 HEC allows for a 2% decrease in complexity compared to the best previous results (Lange 2002a, Lange 2002b, Kovtun, Wollinger 2007). From (Hankerson et al. 2000) it is known that the complexity of a simple scalar multiplication is, on average:

$$DSM = \frac{1}{2}t \cdot DA + t \cdot DD,$$

where t is the bitlength of the scalar, DSM is the complexity of the divisor scalar multiplication, DA is the complexity of the divisor addition, DD is the complexity of a divisor doubling. Thus, using the proposed algorithms, we decrease the complexity of the scalar multiplication by $\frac{1}{2}t$ field multiplications in comparison with (Kovtun, Wollinger 2007) and by $7t$ field multiplications in comparison with (Lange 2002a).

For the implementation formulas from (Lange 2002a, Kovtun, Wollinger 2007) and proposed are used Microsoft Visual C++ 2005 without assembler. For the performance benchmark is used workstation with AMD AthlonXP (Barton core) 2500+ MHz CPU and Microsoft Windows XP OS.

Jacobian arithmetic based on the own fast finite field library tuned to the latest x86 processors family.

Table 2. Experimental timings for the simple scalar multiplication of weight 2 divisor [ms]

Results / Base field	$\mathbf{GF}(p_{80})$	$\mathbf{GF}(p_{84})$	$\mathbf{GF}(p_{161})$
$\deg(h) = 2, h_i \in \mathbf{F}_2$ (Lange 2002b)	11352.6	12209.4	66288
$\deg(h) = 2, h_i \in \mathbf{F}_2$ (Kovtun, Wollinger 2007)	11156.24	11998.22	65268.8
$h(x) = 0, f_4 = 0$ [proposed]	10463	11252.66	61313.6

When using the proposed formulas, we can reduce the time for a scalar multiplication by approximately 6.2%.

References

- Brown, M., Hankerson, D., Lopez, D., Menezes, A. (2000) Software implementation of the NIST elliptic curves over prime fields. Research Report CORR 2000–55. Department of Combinatorics and Optimization, University of Waterloo. –Canada: Waterloo, Ontario, 21p.
- Cantor, D.G. (1987) Computing in the Jacobian of hyperelliptic curve. *Math. Comp.*, No 48. pp. 95–101.
- Hankerson, D., Lopez, J., Menezes, A. (2000) Software implementation of elliptic curve cryptography over binary fields / In Cetin K. Koc and C. Paar editors. *Workshop and embedded systems. –CHES'99. – LNCS 1717. –Berlin: Springer–Verlag. pp.1–24.*
- Harley, R. (2000) Fast arithmetic on genus 2 curves. Available at: <http://cristal.infra.fr/~harley/hyper>.
- Institute of Electrical and Electronics Engineers. (2000) IEEE P1363–2000: Standard Specifications for Public Key Cryptography. 206p.
- International Organization for Standardization. (2002) ISO/IEC FCD 15946-2: Information technology - Security techniques - Cryptographic techniques based on elliptic curves - Part 2: Digital signatures, Final Committee.
- Koblitz, N. (1989) Hyperelliptic cryptosystems. *Journal of cryptology*, No 1. pp.139–150.
- Kovtun, V., Wollinger, T. (2007) Fast explicit formulae for genus 2 hyperelliptic curves using projective coordinates. In *pro. 4th International conference on Information Technology (ITNG'2007)*. pp. 893–897.
- Kruger, U. (2001) Anwendung hyperelliptischer kurven in der kryptographie. Master's thesis, Universitat Gesamthochschule Essen.
- Lange, T. (2001) Efficient arithmetic on hyperelliptic curves. PhD thesis, Universitat Gesamthochschule Essen.
- Lange, T. (2002a) Efficient arithmetic on genus 2 hyperelliptic curves over finite fields via explicit formulae. *Cryptology ePrint Archive. Report 2002/121*. Available <http://eprint.iacr.org>.
- Lange T. (2002b) Inversion-free arithmetic on genus 2 hyperelliptic curves. *Cryptology ePrint Archive. Report 2002/147*. Available <http://eprint.iacr.org>
- Lange, T. (2002c) Weighted coordinates on genus 2 hyperelliptic curves. *Cryptology ePrint Archive. Report 2002/153*. Available <http://eprint.iacr.org>.
- Miyamoto, Y., Doi, H., Matsuo, K., Chao, J., Tsujii, S. (2002) A fast addition algorithm of genus two hyperelliptic curve. In the 2002 Symposium on cryptography and information security. – SCIS 2002, IEICE Japan, pp.497–502. In Japanese.
- Spallek, A.M. (1994) Kurven vom geschlecht 2 und ihre anwendung in public-key-kryptosystemen. PhD thesis, Universitat Gesamthochschule Essen.
- State Standards of Ukraine (Derzhavni Standarty Ukrainy, DSTU) (2002) DSTU 4145-2002. Information technologies. Cryptographic information security. Digital signature, what based on elliptic curves. Generation and verification. –K.: Derzhstandart Ukrainy, 40p.
- Suguzaki, H., Matsuo, K., Chao, J., Tsujii, S. (2002) An extension of Harley algorithm addition algorithm for hyperelliptic curves over finite fields of characteristic two. Technical report ISEC2002-9 (2002-5), IEICE Japan, pp. 49–56.
- Takahashi, M. (2002) Improving Harley algorithms for jacobians of genus 2 hyperelliptic curves. In *Proc. of SCIS2002, IEICE Japan*. in Japanese.
- Wollinger, T. (2004) Software and hardware implementation of hyperelliptic curve cryptosystems. PhD dissertation. Bochum, Germany, May 2004.