An approach to Elliptic Curve Cryptography with AOP oriented to Hardware

Luckas A. Farias[1,2], **Bruno C. Albertini**[1], **Paulo S. L. M. Barreto**[1,3]

[1] Escola Politécnica, Universidade de São Paulo, Brazil.
Av. Prof. Luciano Gualberto, trav. 3, no. 158 (Ed. Engenharia Elétrica), s. C2-46.
05508-900 Sã Paulo, Brazil - BR.

luckas.farias@usp.br ; balbertini@usp.br ; pbarreto@usp.br

[2]Departamento de Sistemas e Computação
Universidade Anhembi Morumbi - UAM - São Paulo - SP.

luckas.farias@anhembi.br

[3]Computer Science program committee
Institute of Technology of the University of Washington Tacoma

pbarreto@uw.edu

Abstract. *This work describes a family of binary Edwards curves that admits modular reductions (an operation that can be responsible for up to 30% of the processing time in point arithmetic) twice as fast than the best usual settings, while essentially being as secure as a binary elliptic curve can be (in terms of being rigid and twist-safe). Moreover, we present a hardware architecture with a generic VHDL description that can be synthesized to any FPGA with enough area to support the circuit. For this architecture, we are able to execute a point multiplication by scalar on* $\mathbb{F}_{562}$ *in 2.28ms on Cyclone 4 GX, in 1.23ms on Virtex7 and in 1.01ms on Zynq7020.*

1. Introduction

This work defines a class of elliptic curves that applies the rules of Edwards binary curves and AOP (All-One-Polynomial) as the irreducible polynomial. These curves have operational advantages over binary fields as regards fastening the reduction.

Another advantage for adopting AOP is that quadratic functions have a property that can be calculated into a single clock cycle, regardless of number of repetitions. For example, during an inversion, if a circuit performs one square per clock cycle, if there are 100 cascade squares operations, an inversion can take more than 100 operations. By adopting the AOP, it is possible to design a circuit that performs more than one square per cycle, without lowering the frequency significantly. In fact, AOP needs only one level of XOR to compute any number of square power repetitions, as will be demonstrated herein.

2. Preliminaries

Let $\mathbb{F}$ be a field of characteristic 2. A (complete) binary Edwards curve [4] over $\mathbb{F}$ is an affine curve of form $E_{B,d_1,d_2} : d_1(x+y)+d_2(x^2+y^2) = xy+xy(x+y)+x^2y^2$ where

$d_1, d_2 \in \mathbb{F}$, $d_1 \neq 0$ and the trace of d_2 is 1 ($\mathrm{Tr}(d_2) = 1$). This curve is birationally equivalent to a Weierstrass curve $W_{B,d_1,d_2} : v^2 + uv = u^3 + a_2u^2 + a_6$ where $a_2 = d_1^2 + d_2$ and $a_6 = d_1^4(d_1^4 + d_1^2 + d_2^2)$.

The quadratic twist of the curve W_{B,d_1,d_2} of order $n = 2^m + 1 - t$ (where the trace t of Frobenius satisfies the Hasse bound $|t| \leq 2^{m/2+1}$) is the curve $W'_{B,d_1,d_2} : v^2 + uv = u^3 + (a_2 + d_2)u^2 + a_6$ of order $n' = 2^m + 1 + t$. An efficient special case is $d_1 = d_2 = d$, i.e. $E_{B,d} : d(x + x^2 + y + y^2) = (x + x^2)(y + y^2)$

An elliptic curve is twist-safe if both curve and quadratic twist have near-prime order, i.e. the curve order has a form of $n = hp$ and its quadratic twist order has a form of $n' = h'p'$ where both p and p' are large primes (hence, cofactors h and h' are very small).

The order of a binary elliptic curve is always even, but it is possible to find Edwards curves with order of form $n = 2p$ where p is prime. In this case, it is also possible to find curves with quadratic twist of order $n' = 4p'$ where p' is prime as well. This work considers only the curves satisfying these conditions.

3. Modular reduction

Modular reduction can be responsible for up to 30% of the processing time in point arithmetic. Conventional wisdom mandates the use of a very sparse polynomial reduction when resorting to polynomial basis representations, typically a trinomial.

Itoh and Tsujii [9] analyzed the properties of reduction by the so-called All-Ones-Polynomial (AOP), i.e. the arithmetic in $\mathbb{F}_2[x]/(x^m + x^{m-1} + \cdots + x + 1)$ assuming that the AOP $x^m + x^{m-1} + \cdots + x + 1$ is irreducible. Unfortunately, this setting has been disregarded in the literature, where m is most typically taken to be prime to avoid Weil descent and related attacks [10, 8, 11], while $x^m + x^{m-1} + \cdots + x + 1$ can only be irreducible if it has an odd number of terms; hence, if m is even.

This work adopts $m = 2q$ where q is prime. This near-prime setting is out of reach from any known conventional attack based on m being composite, except for possibly a very small advantage factor $O(\sqrt{m})$. In this sense, it resembles a situation of curve group orders, which do not need to be prime as long as they are near-prime. As shown by Itoh and Tsujii [9], a necessary and sufficient condition for AOP to be irreducible is that $m + 1$ is prime and that 2 is a generator of $\mathbb{Z}^*_{m+1}$. These stringent requirements severely limit the available choices of m, but there are still plenty of values of interest, adequately covering all practical security levels.

4. Point multiplication by scalar

The main operation in ECC (Elliptic Curve Cryptography) is the multiplication of a point in the curve by a scalar value. The intuitive way is to implement this operation as a series of double-and-add operations. However, doing so means that the number of operations would be key dependent, leaking a potential information.

Montgomery ladder [13] is an algorithm to solve this problem, performing multiplication as a series of double-and-add operations, but always performing the same number of steps for any key. This algorithm iterates over bit k_i of key K, and the operation does not depend on k_i value, preventing information leakage.

Using the Montgomery ladder algorithm, we can define the double-and-add operation based on Edwards [5, 4] equations, proven to be a complete operation[1] for all cases of binary Edwards curves, enabling its use for any point in the curve.

Memory reutilization allows avoiding recalculating the inverse, because it is a function that is calculated with squares and multiplications. Itoh and Tsujii [9] has shown that this operation is singular for each field. An interseting work that made other optimizations is Kim et. al work [12].

4.1. Addition and double optimization

Other works have concerned in presenting new formulas for the arithmetic on the binary Edwards curves which are much faster than the-state-of-the-art [12, 3]. In this work, he concludes that it is possible to perform the complete mixed differential addition and doubling for complete binary Edwards curves with only $5M + D + 4S$, this considering $d_1 = d_2$. This result is the cost of the fastest (but incomplete) formulas among various forms of elliptic curves over finite fields of characteristic 2 in the literar [12]. Here M, S, and D are the cost of a multiplication, a squaring and a multiplication by constant, respectively.

We do not make extra optimization, but opt for using the formula for affine differential addition and doubling that have the cost $I + 4M + 2D + 3S$ in Bernstein-Lange-Farashashi [3] and cost $I + 4M + D + 3S$ in Kin et. al [12]. Here I, M, S, and D are the cost of an inversion, a multiplication, a squaring and a multiplication by constant, respectively. We use these results in our work and derive our results based on this cost per operation of this affine double-and-add operation.

5. Curve choice

For $\mathbb{F}_2[x]/(x^m + x^{m-1} + \cdots + x + 1)$, it is often (always for all cases of practical interest) the case that polynomial x itself has $\mathrm{Tr}(x) = 1$; therefore, it is an obvious candidate for the curve equation. This leaves d_1 as an open choice, limited only by the requirement that both E_{B,d_1,d_2} and its twist E'_{B,d_1,d_2} have orders of $n = 2p$ and $n' = 4p$ (or $n = 4p$ and $n' = 2p$), respectively, where p and p' are prime and by the practical constraint where the multiplication by d_1 is efficient (for instance, by imposing that d_1 has the lowest degree possible).

5.1. Sample curves

The underlying finite field is $\mathbb{F}_2[x]/(x^m + x^{m-1} + \cdots + x + 1)$ for the stated value of m, and $d_2 = z$. The value of d_1 is always the first in lexicographical order that yields a curve of order $n = 2p$ or $4p$ for a prime p, whose quadratic twist of order is $n' = 4p'$ or $2p'$ for a prime p', respectively. As an example of curve and parameters that were obtained, the definition for a curve with 562 bits is $d = z^{20} + z^{15} + z^3 + 1$, $n = 4 \times$ 3773962424821541352241554580988268890916921220416440428376206300245624162392148852085424555861285302871699693039890116397569024054345480705562779107921462952194579361881 and $n' = 2 \times$ 7547924849643082704483109161976537781833842440832880856752412600491248324784297704173657788988064464422474114043275146803944290411008209119120178789887858270208422584143

[1] Complete operation: when it is possible to calculate the result with one logical operation, regardless of input value.

6. Hardware implementation

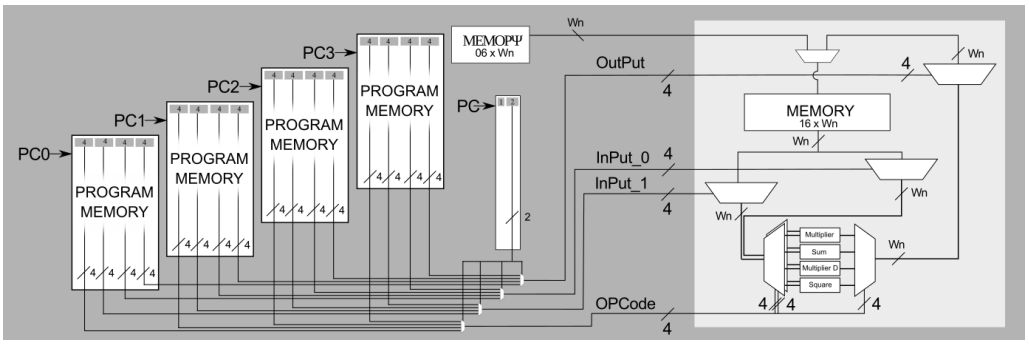

Figure 1. Top level coprocessor block diagram.

A coprocessor was built in a generic VHDL (VHSIC Hardware Description Language), tailored to support all operations (including overload operations) defined over the field.

The top level circuit can be seen in Figure 1, where the program memory is depicted (for 16-bit instructions). This architecture is an abstraction of the architecture that we published in "Cryptographic architecture for co-process on consumer electronics devices" [7], but the main diference is that here the memory is organized as 4 nibbles.

6.1. Binary Field Multiplication

Binary field multiplication is a multiplication between two polynomial elements of the field. If the field multiplication produces a polynomial that is not in the field, it will be necessary to reduce it back to the field. This is obtained reducing the result by an irreducible polynomial (IP), generating an element of the field.

Classical hardware multiplication is implemented by a shift-and-add algorithm, where we have a multiplication of polynomial a by b; $a,b \in \mathbb{F}_m$. Considering "$\oplus$" as the logical operation XOR, "&" as the logical operation AND and "$\ll x$" as a shift operation by x bits.

This method requires m steps to converge into solution, but has the characteristic that it is a complete operation. Moreover, this operation can be parallelized [6]. Knowing this advantage, the next section describes the construction of a parallel architecture to execute the field multiplication.

6.2. Parallel Field Multiplication

The multiplication module is tailored to explore the advantages over the AOP field elements, which is the subject of this work. The first advantage is, since all fields are even, two steps of multiplications can be performed at the same time [6].

The proposed architecture is based on a basic module that performs two shift-and-add operations. We call it Double Step Module (DSM) and its logical equation is:

$$c_{out} := \{[(c_{in} \oplus (a\&b_i)) \ll 1] \oplus [(a\&b_{i-1}) \ll 1]\}\ mod(IP)$$

The number of times that we are going to replicate this module is proportional to the field size and parallelism level, assuming that DSM_u has u levels. E.g. F_{562} need one

DSM instance 231 times. Replicating the *DSM* reduces the number of times by half, but also doubles[2] the area used by this module.

Figure 2, depicts our proposed parallel architecture, composed by the replicated *DSM* module and output multiplexor.

The interface is straightforward, being composed of two m bits inputs and one output of the same size. All inputs are field elements, as well the output. Control signals are limited to clock, start flag and end flag. Using this start-end synchronization, control unit hides multiplication parallelism from upper modules.

For more results over this architecture of parallel field multiplication, including some comparatives between this performance and the area usage, the work "Parallelism Level Analysis of Binary Field Multiplication on FPGAs" [6] conducted a deeper study into this architecture and measured the throughput, the area consumption and made an index using throughput/area.

6.3. Square

The field square operation, considering AOP and the little Fermat theorem (which proves $a^p = (a) mod (p)$), can perform an optimization that improves the inverse calculation [9]

[2] Approximate value due to synthesis optimizations.

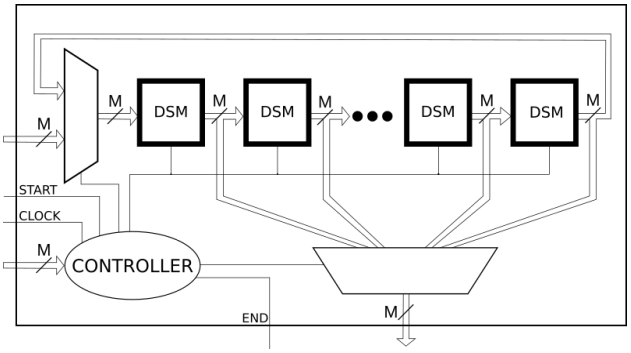

Figure 2. Top Level of parallelism in multiplication

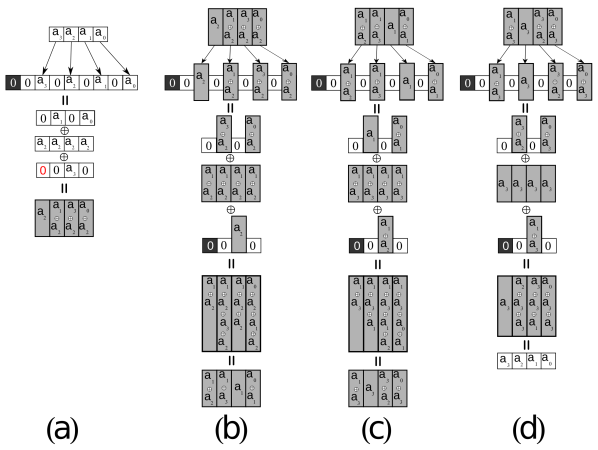

Figure 3. Sequential squares in the field $n = 4$ using AOP as irreducible polynomial

where we are searching for $a^{-1} = \frac{a^{2^n}}{a} = a^{2^n-1}$.

Figure 3 depicts a sequential square operation set over a binary field with $n = 4$ and where (a) is the first square , (b) is the square of result (a), (c) is the square of result (b) and (d) is the square of result (c), returning to the initial polynomial as predicted by the little fermat theorem. As observed, all the results of squares are, at most, a single logical XOR with two permuted vectors of initial value. In addition, although it is not certain that it is a property of all fields with AOP, it is a valid property of all fields used herein[3].

By knowing this advantage, modules were implemented for exploring this "roll effect" to execute the square operations required by the inverse. For example, for fields 346, 446 and 562 we need to implement (1, 2, 5, 10, 21, 43, 86, 172), (1, 3, 7, 14, 29, 58, 116, 232) and (1, 2, 4, 8, 17, 35, 70, 140, 280) squares, respectively, to make use of this optimization.

7. Experimental Results

Table 1 shows the resources used by the synthesis tool without any specific optimization. Any resource tied to FPGA model was also avoided. For each field size, the required hardware is synthesized to perform the instructions for which the coprocessor was designed (i.e. all the cryptography operations over the field) without any pipeline. Minimum delay for each unit, which directly implies the maximum clock frequency, is depicted in Table 2.

Board	Field	Parallelism	Slices	Slices FF	LUT's
Zynq 7020	Available	—-	106,400	35,104	53,200
Zynq 7020	346	24	11123	7624	42352
Zynq 7020	466	24	9,991	7,046	36,797
Zynq 7020	562	24	11,123	7,624	42,352

Table 1. Synthesis results by Xilinx ISE [14] for Zynq 7020

Board	Field	Parallelism	General		Multiplication	
			Minimum Period (ns)	Frequency (MHz)	Minimum Period (ns)	Frequency (MHz)
DE2i-150	346	36	13.749	72.730	10.300	97.090
DE2i-150	466	36	13.615	73.450	10.035	99.650
DE2i-150	562	36	14.888	67.170	12.034	83.100
Zynq 7020	346	24	4.147	241.138	4.792	208.681
Zynq 7020	466	24	4.357	229.516	5.418	184.570
Zynq 7020	562	24	4.690	213.220	5.435	183.993

Table 2. Time results by QuartusII[2] for DE2i-150 and Xilinx ISE[14] for Zynq7020

Table 3 illustrates the number of clock cycles used in the elliptic cryptographic operation, from the start signal until the finish flag for each bit on Montgomery ladder. This table is made using the cost $I + 4M + 2D + 3S$ for Bernstein et. al [3] and $I + 4M + 1D + 3S$ for Kim et. al [12]. We are not considering the mixed coordinate that uses Z value. Normal Clocks (NC) are for the general process, and Multiplication Clock (MC) are the clocks used by multiplication. ECC operations are sum, doubling, and the combination Sum+Double operation.

With those results, we estimated how many point multiplications by scalar operations the synthesized hardware can execute. Table 4 is derived from the same data,

[3]This has been extensively tested manually to the fields of interest

Field	*DSM*	Bernstein et. al [3]		Kim et. al [12]	
		NC	MC	NC	MC
346	1	39	4176	36	4176
346	12	39	372	36	372
346	24	39	204	36	204
346	30	39	168	36	168
346	36	39	144	36	144
466	1	39	5616	36	5616
466	12	39	492	36	492
466	24	39	264	36	264
466	30	39	216	36	216
466	36	39	180	36	180
562	1	42	7332	39	7332
562	12	42	637	39	637
562	24	42	338	39	338
562	30	42	273	39	273
562	36	42	234	39	234

Table 3. Clock cycles required by each operation

showing how many multiplication operations the synthesized hardware can accomplish per second, considering each field size.

Board × Field	82	106	178	226	346	466	562
Zynq 7020	29,371.65	19,495.66	7,336.41	5,233.57	1,875.18	1,462.80	984.93
Virtex-7 690T	29,371.65	17,626.98	6,623.95	4,501.74	1,636.20	1,266.70	810.26
DE2i-150	10,978.32	7,280.19	2,941.38	2,422.57	1,167.79	729.71	437.80

Table 4. Point multiplications by scalar per second

8. Publications

We published one paper about the field multiplication architecture and the analysis performed on its parallelism level. This paper is the **Parallelism Level Analysis of Binary Field Multiplication on FPGAs [6]** whose authors are Luckas Farias, Bruno Albertini and Paulo Barreto. It was published at the *V Brazilian Symposium on Computing Systems Engineering - SBESC*.

Another accepted paper is about the architecture for a high level coprocessor, trying to explain the idea of a future standard architecture on consumer electronics. The paper **Cryptographic architecture for co-process on consumer electronics devices [7]**, whose authors are Luckas Farias, Bruno Albertini and Paulo Barreto, was published in 2016 at the *IEEE International Symposium on Consumer Electronics*.

The most recent publication is about the architecture and the new Edwards curves (which is presented in this work). The paper **A class of safe and efficient binary Edwards curves [1]**, whose authors are Luckas Farias, Bruno Albertini and Paulo Barreto, was published at the *Journal of Cryptography Engineering - JCEN*.

8.1. Talks on events

In some non academic events, we presented some talks on this subject. This occurred during a master's, whose subject is part of this research. In total, we presented 38 talks (7 of them in international events) and 4 workshops.

9. Conclusion

In conclusion, a new approach over binary Edwards curves is proposed by applying optimization with AOP. This method was not found in the literature, but it is possible to

apply in actual scenarios, since it was based on secure concepts such as Montgomery ladder, Edwards curve and secure hardware implementation. The resulting coprocessor can be used for different security levels and it is just as efficient as the ones proposed in the literature.

Therefore, it is considered a good approach for binary ECC, updating the technology and opening new possibilities of research that include other curves with this optimizations employing this approach, other methods and hardware optimizations.

References

[1] Luckas A. Farias, Bruno C. Albertini, and Paulo S. L. M. Barreto. A class of safe and efficient binary edwards curves. *Journal of Cryptographic Engineering*, Jan 2018.

[2] Altera. Quartus ii web edition. dl.altera.com?edition=web, 2015.

[3] Daniel J. Bernstein and Tanja Lange. *Faster Addition and Doubling on Elliptic Curves*, pages 29–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[4] DanielJ. Bernstein, Tanja Lange, and Rezaeian Farashahi. Binary edwards curves. *Cryptographic hardware and embedded systems – CHES 2008*, 5154:244–265, 2008.

[5] Harold M. Edwards. A normal form for elliptic curves. *Bull. Amer. Math. Soc.*, pages 393–422, 2007.

[6] L. A. Farias, B. C. Albertini, and P. S. L. M. Barreto. Parallelism level analysis of binary field multiplication on fpgas. In *2015 Brazilian Symposium on Computing Systems Engineering (SBESC)*, pages 64–69, Nov 2015.

[7] L. A. Farias, B. C. Albertini, and P. S. L. M. Barreto. Cryptographic architecture for coprocess on consumer electronics devices. In *2016 IEEE International Symposium on Consumer Electronics (ISCE)*, pages 3–4, Sept 2016.

[8] S.D. Galbraith, F. Hess, and N.P. Smart. Extending the ghs weil descent attack. Cryptology ePrint Archive, Report 2001/054, 2001. `http://eprint.iacr.org/2001/054`.

[9] Toshiya Itoh and Shigeo Tsujii. Structure of parallel multipliers for a class of fields gf (2 m). *Information and computation*, 83(1):21–40, 1989.

[10] Rivera J. and Meulen R. D. V. Weil descent page. `www.cs.bris.ac.uk/~nigel/weil_descent.html`, 2017.

[11] Michael Jacobson, Alfred Menezes, and Andreas Stein. Solving elliptic curve discrete logarithm problems using weil descent. Cryptology ePrint Archive, Report 2001/041, 2001. `http://eprint.iacr.org/2001/041`.

[12] Kwang Ho Kim, Chol Ok Lee, and Christophe Negre. *Binary Edwards Curves Revisited*, pages 393–408. Springer International Publishing, Cham, 2014.

[13] Peter L. Montgomery. Speeding the pollard and elliptic curve methods of factorization. *Mathematics of Computation*, (48):243–264, 1987.

[14] Xilinx. Xilinx ise webpack 14.7. www.xilinx.comproductsdesign-toolsise-design-suiteise-webpack.html, 2015.