Public-Key Cryptography on the Top of a Needle

Lejla Batina, Nele Mentens, Kazuo Sakiyama, Bart Preneel, Ingrid Verbauwhede
Katholieke Universiteit Leuven, ESAT/SCD-COSIC, Belgium
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
Email: {lbatina, nmentens, ksakiyam}@esat.kuleuven.be

Abstract—This work describes the smallest known hardware implementation for Elliptic/Hyperelliptic Curve Cryptography (ECC/HECC). We propose two solutions for Public-key Cryptography (PKC), which are based on arithmetic on elliptic/hyperelliptic curves. One solution relies on ECC over binary fields $\mathbb{F}_{2^n}$ where n is a composite number of the form $2p$ (p is a prime) and another on HECC on curves of genus 2 over $\mathbb{F}_{2^p}$. This implies the same arithmetic unit for both cases which supports arithmetic in a field $\mathbb{F}_{2^p}$. Our best solution that still results in a feasible performance features less than 5 k gates with an average power consumption smaller than 10 μW .

I. INTRODUCTION

The field of embedded systems is growing at a rapid rate, as devices such as mobile phones, PDAs, smart cards and more recently RFID tags, sensor nodes and key immobilizers have become unavoidable in our everyday life. Hence, the distinguishing characteristics of embedded security can be divided into two categories: resource-limitation and physical accessibility. The former one specifies severe resource constraints on the security architecture in terms of memory, computational capacity, and energy for embedded devices. The most challenging tasks for embedded security are implementations of Public-key Cryptography (PKC).

RFID tags are small wireless devices for pervasive computing. Despite their rigorous constraints featuring extremely low budget for power and die size they also give rise to serious security and privacy issues. Typical security services include authentication, key management and encryption. Although some experts are a priori giving up on public key solutions, assuming it being too expensive and too power hungry, there exists a line of research exploring the limits of compact public key implementations for low-cost applications such as RFIDs and sensor networks.

In our previous work we investigated standardized low cost solutions for Elliptic Curve Cryptography (ECC) processors supporting security algorithms and protocols for RFID [1]. Namely, in standards it is mainly recommended to use ECC over a field $\mathbb{F}_{2^p}$, where p is a prime. In this work we describe a new solution based on Hyperelliptic Curve Cryptography (HECC) and on ECC over composite fields. HECC has some advantages over ECC because of the possibility to work in a smaller field *e.g.* for HECC (in the case of genus 2 curves) one can work in the field $\mathbb{F}_{2^n}$ whilst obtaining the same level of security as for ECC over fields of bit-lengths that are twice as large. The same holds for ECC over composite fields. This property allows for more compact ALU than in the case of ECC.

We revisit the algorithms for HECC and ECC over composite fields and optimize them on the number of registers, resulting in an area-optimized solution. Our results show that the arithmetic unit for those two cases is smaller than the one for standardized ECC, but memory requirements are slightly bigger. Using RAM for storage, which would result in smaller number of gates than register files, one could have a feasible low cost solution using all kinds of curve-based cryptography.

The paper is organized as follows. Section II lists some related work. In Sect. III we give some background information on curve-based cryptography and supporting arithmetic. In Sect. IV we elaborate on a suitable selection of parameters and algorithms and in Sect. V we outline our architecture and describe our hardware implementation. Our results are discussed in Sect. VI. Section VII concludes the paper.

II. RELATED WORK

As related previous work we mention implementations of Public-key cryptosystems for resource constrained environments such as RFID tags and sensor networks.

Gaubatz *et al.* [5] investigated ECC implementations for wireless sensor networks. The architecture of the ECC processor occupied an area of 18 720 gates and consumed less than 400 μW of power at 500 *kHz*. The field used was a prime field of order $\approx 2^{100}$.

PKC processors for RFID tags include the results of Wolkerstorfer [12] and Kumar and Paar [8]. Wolkerstorfer [12] showed that ECC based PKC is feasible on RFID-tags by implementing the ECDSA on a small IC. The chip has an area complexity of around 23 000 gates and it features a latency of 6.67 *ms* for one point multiplication at 68.5 *MHz*. However, it can be used for both types of fields *e.g.* $\mathbb{F}_{2^{191}}$ and $\mathbb{F}_{p^{192}}$. The results of Kumar and Paar [8] include an area complexity of almost 12 k gates and a latency of 18 *ms* for one point multiplication over $\mathbb{F}_{2^{131}}$ at 13.56 *MHz*. The operating frequency is in both cases too high for those applications and therefore the results cannot be properly evaluated. Namely, with such a high frequency the power consumed becomes too large, which has the most crucial impact on the feasibility of the implementations. Our previous work described in [1] considers an ECC processor over a range of fields varying from $\mathbb{F}_{2^{131}}$ till $\mathbb{F}_{2^{163}}$. The best solution features 6718 gates for arithmetic and control unit (data memory not included) in 0.13 μm CMOS technology over the field $\mathbb{F}_{2^{131}}$, which provides a reasonable level of security for the time being. In this case the consumed power is less than 15 μW when operating frequency

is 200 kHz. We compare the previous implementations with our new results in Section VI in more detail.

III. MATHEMATICAL BACKGROUND

In general, ECC relies on a group structure induced on an elliptic curve. A set of points on an elliptic curve together with the point at infinity, denoted ∞ , and with point addition as binary operation has the structure of an abelian group. Here we consider finite fields of characteristic two. A non-supersingular elliptic curve E over $\mathbb{F}_{2^n}$ is defined as the set of solutions $(x, y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$ of the equation: $y^2 + xy = x^3 + ax^2 + b$ where $a, b \in \mathbb{F}_{2^n}$, $b \neq 0$, together with ∞ .

Hyperelliptic Curve Cryptography (HECC) was proposed in 1988 by Koblitz [7] as a generalization of ECC. More details can be found in [2], [10].

The scalar multiplication *i.e.* operation $k * P$ (where k is an integer and P a point or divisor on EC/HEC respectively) is the basic operation for all ECC/HECC protocols. At the next (lower) level are the group operations *i.e.* point/divisor addition and doubling. The lowest level consists of finite field operations such as addition, subtraction, multiplication and inversion required to perform the group operations. The scalar multiplication is easily performed via repeated group operations. The basic scheme is called double-and-add or the binary method [2].

Elliptic curves can be viewed as a special case of hyperelliptic curves *i.e.* an elliptic curve is a hyperelliptic curve of genus $g = 1$. Here we consider a hyperelliptic curve C of genus $g = 2$ over $\text{GF}(2^n)$, which is defined by an equation of the form: $C : y^2 + h(x)y = f(x)$ in $\text{GF}(2^n)[x, y]$, where $h(x) \in \text{GF}(2^n)[x]$ is a polynomial of degree at most g and $f(x)$ is a monic polynomial of degree $2g + 1$. For genus 2 curves, in the general case the following equation is used: $y^2 + (h_2x^2 + h_1x + h_0)y = x^5 + f_4x^4 + f_3x^3 + f_2x^2 + f_1x + f_0$.

For ECC over a composite field we consider $\mathbb{F}_{2^{2p}}$ as a field of quadratic extension over $\mathbb{F}_{2^p}$, so we can write $\mathbb{F}_{2^{2p}} = \mathbb{F}_{2^p}[x]/(f(x))$, where $\deg(f) = 2$. In this case each element from the field $\mathbb{F}_{2^{2p}}$ is represented as $c = c_1t + c_0$ where $c_0, c_1 \in \mathbb{F}_{2^p}$, and all operations in this field can be done by means of operations in $\mathbb{F}_{2^p}$.

In [4], the Weil descent attack is introduced against EC defined over binary fields of composite degree $n = k \cdot m$. This work put some doubt on security of composite field implementations of EC in general. However, further investigations have shown that composite fields with degree $n = 2 \cdot p$ (*i.e.*, extension of degree two), where p is prime, remain secure against Weil Descent attacks and its variants.

IV. ALGORITHMS SELECTION AND OPTIMIZATIONS

For the ECC point multiplication we chose the method of Montgomery [11] that maintains the relationship $P_2 - P_1$ as invariant. It uses a representation where computations are performed on the x -coordinate only in affine coordinates (or on the X and Z coordinates in projective representation). That fact allows us to save registers which is one of the main criteria for obtaining a compact solution.

As starting point for our optimizations we use the formulae of Lopez and Dahab [9]. The original formulas in [9] require 2 intermediate registers in normal ECC if the point operations are performed sequentially. In our case we eliminated one more intermediate register, which added a few more steps to the original algorithms. However, mapping this algorithm to the corresponding one suitable for composite fields requires some additional intermediate registers as we translate the field arithmetic to the arithmetic in the subfield as shown in Table I.

TABLE I
BASIC OPERATIONS IN $\mathbb{F}_{(2^p)^2}$ EXPRESSED IN OPERATIONS IN $\mathbb{F}_{2^p}$.

Operation in $\mathbb{F}_{(2^p)^2}$	Addition	Multiplication
$\mathbb{F}_{2^p}$	2ADD	3MUL+4ADD

From the formulae for point operations for projective coordinates [2] it is evident that we need to implement only multiplications and additions. Squaring is considered as a special case of multiplication in order to minimize the area and inversion is avoided by use of projective coordinates. We assume that conversion to affine coordinates can be computed at the reader's side. Note also that, if necessary, the one inversion that is required can be calculated by use of multiplications. In this way the area remains almost intact and some small control logic has to be added.

For our HECC implementation we used so-called type II curves [3], which are defined by $h_2 = 0, h_1 \neq 1$. As a starting point for divisor operations we used formulae from [6]. Those curves allow for faster doublings than for a general curve, while security remains intact. However, we optimized the formulae on the number of intermediate variables which resulted in a small increase in number of multiplications. In this way, we can perform a trade-off between area and performance.

V. CURVE-BASED PROCESSORS FOR LOW-COST APPLICATIONS

Our solution for a curve-based processor is shown in Fig. 1. The architecture consists of the following blocks: a control unit (FSM), a modular arithmetic unit (MALU), and some memory (RAM and ROM). In ROM the ECC/HECC parameters and some constants used in algorithms are stored. On the other hand, RAM contains all input and output variables and therefore it communicates with both, the ROM and the MALU.

The FSMs control the scalar multiplication $k * P$ and the point/divisor operations. In addition, the controller commands the MALU which performs field operations. When the START signal is set, the bits of $k = \sum_{i=0}^{n_k-1} k_i 2^i$, $k_i = \{0, 1\}$, $n_k = \lceil \log_2 k \rceil$, are evaluated from MSB to LSB. The control consists of a number of simple state machines and a counter and its area cost is small.

The datapath of the MALU is an MSB-first bit-serial $\mathbb{F}_{2^n}$ multiplier with digit size d . This arithmetic unit computes $A(x)B(x) \bmod P(x)$ where $A(x) = \sum a_i x^i$, $B(x) = \sum b_i x^i$

Fig. 1. Architecture of our curve-based processor.

and $P(x) = \sum p_i x^i$. Modular addition is also supported by the same hardware logic. This operation requires additional multiplexors and XORs. However the cost of this solution is much cheaper compared to the case of having a separate modular adder. This type of hardware sharing is very important for such low-cost applications. The proposed datapath is scalable in the digit size d which can be determined arbitrary by exploring the best combination of performance and cost. Details of the MALU are given in [1].

VI. RESULTS

Now we give the results for area complexity, power consumption and the latency in the case of ECC/HECC scalar multiplication. The designs were synthesized by Synopsys Design Vision using a $0.13 \mu\text{m}$ CMOS library. We rely on the arithmetic in binary fields $\mathbb{F}_{2^p}$ where p varies from bit-size 67 till 83.

The results of the area complexity for various architectures with respect to the choice of fields and the size of d for the MALU are given in Table II. The results for the complete architecture in gates are given in Table III.

TABLE II
THE AREA COMPLEXITY OF MALU IN GATES.

Field size	$d=1$	$d=2$	$d=4$	$d=6$	$d=8$
67	2295	2551	3041	3574	4056
71	2426	2694	3201	3709	4217
79	2705	2953	3558	4157	4676
83	2844	3154	3739	4328	4956

TABLE III
THE COMPLETE AREA COMPLEXITY IN GATES.

Field size	$d=1$	$d=2$	$d=4$	$d=6$	$d=8$
ECC: 67	4345	4600	5089	5612	6103
HECC: 67	5893	6147	6635	7166	7652
ECC: 71	4523	4787	5293	5801	6309
HECC: 71	6073	6337	6845	7352	7861
ECC: 79	4884	5180	5740	6340	6864
HECC: 79	6436	6729	7290	7894	8413
ECC: 83	5071	5375	5971	6558	7193
HECC: 83	6622	6930	7513	8112	8747

The graphical representations of our results for area in μm^2 and for the total power consumed are shown in Fig. 2, Fig. 3,

Fig. 4 and Fig. 5 for ECC and HECC respectively. The power estimates were made assuming the operating frequency of 200 kHz . With this frequency the power stays between 8 and $16 \mu\text{W}$ which is assumed to be acceptable for these applications.

Fig. 2. Results for area complexity in μm^2 (ECC-comp).

Fig. 3. Results for the power consumed (ECC-comp).

Next we give the numbers for the performance. For the point multiplication we used Montgomery ladder and for divisor scalar multiplication the binary method. We calculate the total number of cycles for each field operation by use of the following formulae for field operations. The total number of cycles for one field multiplication is $\lceil \frac{n}{d} \rceil + 3$ where n and d are the bit size of an arbitrary element from the field in which we are working and the bit size respectively. On the other hand, one field addition takes 4 cycles. The number of cycles required for one point multiplication in the case of ECC over a composite field $\mathbb{F}_{2^{2p}}$ is: $(p-1)[36(\lceil \frac{p}{d} \rceil + 324)]$. For HECC over $\mathbb{F}_{2^p}$ we get the following number of cycles for one scalar multiplication: $(p-1)[115(\lceil \frac{p}{d} \rceil + 621)]$.

The results for the total number of cycles of one point multiplication for ECC are given in Table IV. To calculate the time for one point multiplication we need an operating frequency. However, the frequency that can be used is strictly influenced by the total power. We assumed an operating frequency of 200 kHz in order to estimate the actual timing as our power results showed to be reasonable for RFID applications. We get

Fig. 4. Results for area complexity in μm^2 (HECC).

Fig. 5. Results for the power consumed (HECC).

TABLE IV

THE NUMBER OF CYCLES REQUIRED FOR ONE POINT MULTIPLICATION FOR ECC.

Field size	$d=1$	$d=2$	$d=4$	$d=8$
67	180 576	102 168	61 766	42 768
71	201 600	113 400	68 040	45 360
79	244 296	137 592	81 432	53 352
83	271 584	150 552	88 560	59 040

210 ms for the best case of ECC over $\mathbb{F}_{2^{67 \times 2}}$ ($d = 8$) and 546 ms for the best case of HECC ($\mathbb{F}_{2^{67}}$ and $d = 8$). Our results are compared with other related work in Table V.

TABLE V

COMPARISON WITH OTHER RELATED WORK.

Ref.	bits	A [gates]	Tech. [μm]	f [kHz]	t [ms]	P [μW]
[8]	131	11 969.93	0.35	13 560	18	-
[5]	100	18 720	0.13	500	410.45	< 400
[12]	192	23 000	0.35	68 500	9.89	n.a.
[1]	131	8104	0.13	500	115	< 30
ECC	134	6103	0.13	200	210	< 13
HECC	134	7652	0.13	200	546	< 17

We underline again that our results for the area complexity

do not include RAM. The amount of storage that is required for our implementation is to store $13n$ and $28n$ bits for ECC and HECC respectively, where n is the number of bits of elements in a subfield. We can conclude that our architecture presents the smallest known curve-based crypto processor for low-cost applications.

VII. CONCLUSIONS

This work presents architectures for low-cost applications of curve-based cryptography for RFID tags. Several solutions for various field sizes are given. Our results show that PKC for RFIDs is an option but further investigations are necessary with respect to the memory requirements and more precise power estimates.

ACKNOWLEDGMENTS

Lejla Batina, Nele Mentens and Kazuo Sakiyama are funded by FWO projects (G.0450.04, G.0475.05) and FP6 project SESOC. This research has been also partially supported by the EU IST FP6 project ECRYPT and by IBBT, and K.U. Leuven (OT).

REFERENCES

- [1] L. Batina, N. Mentens, K. Sakiyama, B. Preneel, and I. Verbauwhede. Low-cost Elliptic Curve Cryptography for wireless sensor networks. In L. Buttyan, V. Gligor, and D. Westhoff, editors, *In Proceedings of Third European Workshop on Security and Privacy in Ad hoc and Sensor Networks*, volume 4357 of *LNCS*, pages 6–17. Springer-Verlag, 2006.
- [2] I. Blake, G. Seroussi, and N. P. Smart. *Elliptic Curves in Cryptography*. London Mathematical Society Lecture Note Series. Cambridge University Press, 1999.
- [3] B. Byramjee and S. Duquesne. Classification of genus 2 curves over F_{2^n} and optimization of their arithmetic. Cryptology ePrint Archive: Report 2004/107.
- [4] G. Frey. How to disguise an elliptic curve (Weil descent). Presentation given at the 2nd Elliptic Curve Cryptography Workshop (ECC '98). Slides available at <http://www.cacr.math.uwaterloo.ca/>, September 14–16, 1998.
- [5] G. Gaubatz, J.-P. Kaps, E. Öztürk, and B. Sunar. State of the Art in Ultra-Low Power Public Key Cryptography for Wireless Sensor Networks. In *2nd IEEE International Workshop on Pervasive Computing and Communication Security (PerSec 2005)*, Kauai Island, Hawaii, March 2005.
- [6] A. Hodjat, L. Batina, D. Hwang, and I. Verbauwhede. HW/SW Co-design of a Hyperelliptic Curve Cryptosystem using a μ Code Instruction Set Coprocessor. *Elsevier Science Integration the VLSI Journal*, 40(1):45–51, 2006.
- [7] N. Koblitz. A family of Jacobians suitable for Discrete Log Cryptosystems. In S. Goldwasser, editor, *Advances in Cryptology: Proceedings of CRYPTO'88*, number 403 in *Lecture Notes in Computer Science*, pages 94–99. Springer-Verlag, 1988.
- [8] S. Kumar and C. Paar. Are standards compliant elliptic curve cryptosystems feasible on RFID? In *Proceedings of Workshop on RFID Security*, page 19, Graz, Austria, July 2006.
- [9] J. López and R. Dahab. Fast multiplication on elliptic curves over $GF(2^m)$. In Ç. K. Koç and C. Paar, editors, *Proceedings of 1st International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 1717 of *Lecture Notes in Computer Science*, pages 316–327. Springer-Verlag, 1999.
- [10] A. Menezes, Y.-H. Wu, and R. Zuccherato. *An Elementary Introduction to Hyperelliptic Curves - Appendix*, pages 155–178. Springer-Verlag, 1998. N. Koblitz: Algebraic Aspects of Cryptography.
- [11] P. Montgomery. Speeding the Pollard and Elliptic Curve Methods of Factorization. *Mathematics of Computation*, Vol. 48:243–264, 1987.
- [12] J. Wölkerstorfer. Scaling ECC Hardware to a Minimum. In ECRYPT workshop - Cryptographic Advances in Secure Hardware - CRASH 2005, September 6–7 2005. invited talk.