ECC Processor with Low Die Size for RFID Applications

Franz Fürbass and Johannes Wolkerstorfer, Graz University of Technology
Institute for Applied Information Processing and Communication (IAIK), Graz, Austria
Email: Franz.Fuerbass@austriamicrosystems.com, Johannes.Wolkerstorfer@iaik.tugraz.at

Abstract—This paper presents the design of a special purpose processor with Elliptic Curve Digital Signature Algorithm (ECDSA) functionality. This digital signature generation device (SGD) was developed especially for RFID tags. The design parameters were low energy consumption, small chip area, robustness against cryptographic attacks, and flexibility. The SGD was designed to work as digital processor in an RFID tag requiring the tag to provide the secret key storage and a PRNG. The SHA-1 calculation needs to be included into the SGD to avoid a microcontroller on the tag. The asymmetric cryptosystem allows authentication of the tag to untrusted third parties without revealing the secret key. The ECDSA functionality was implemented using a prime field $GF(p)$ and affine coordinates, an alternative way to reduce the die size and the costs of the tag. The standard-cell based implementation of the device is fully scalable for different prime fields sizes. The $GF(p_{192})$ version will need 23k gate equivalents or 1.3 mm^2 for a $0.35\text{ }\mu\text{m}$ process. 502k clock cycles are used for signature generation. The near-Spice level power simulation with Nanosim estimated the final energy consumption to 0.846 mWs for a generated signature.

I. INTRODUCTION

Radio Frequency Identification (RFID) has settled down in the industry as a bar code replacement and will further proceed to be the “Internet of Things”, meaning an RFID tag attached to everything and communicating to other devices. Today it’s impossible to think of a network without security, the same will be true for RFID networks. The task of an RFID tag is to provide information over the radio channel using minimal hardware components. In many applications it is important that this claimed information can be proven by the tag to the reader device. This work presents an Elliptic Curve Cryptography (ECC) processor for RFID tags which implements an ECDSA signature generation device (SGD). ECC is utilized to gain strong resistance against cryptographic attacks and to reduce the storage requirements. The RFID tag will provide cryptographic authentication and copy protection with the help of the digital signature. Of the many possible ways to implement such an ECC processor, those design parameters were chosen that allow the general use in small embedded devices and especially as digital processor for RFID tags. Figure 1 shows the block diagram of an RFID tag with an integrated SGD.

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932. The information in this paper is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Fig. 1. RFID tag with integrated signature generation device (SGD).

II. REQUIREMENTS AND DESIGN DECISIONS

ECDSA: Authentication should use well known cryptographic methods, which are confirmed to be secure by a huge community. The Elliptic Curve Digital Signature Algorithm (ECDSA) provides authentication utilizing the elliptic curve discrete logarithm problem (ECDLP) (section 1.1 of [2]) as underlying intractable operation. ECC promises the same security level for a 160-bit key as with the RSA method using a 1000-bit key [9]. This makes ECDSA attractive for small devices like RFID tags, where the die size is the major cost factor. The ECDSA signature generation procedure [2] is given by Algorithm 1.

Algorithm 1 ECDSA signature generation

Require: base point $P = (x, y)$, $H(x) := \text{SHA-1}(x)$, message m , base point order n , private key d

Ensure: signature $[r, s]$

- 1: Select $k \in [1, n - 1]$
- 2: $(x_1, y_1) = kP$
- 3: $r = x_1 \bmod n$
- 4: $e = H(m)$
- 5: $s = k^{-1}(e + dr) \bmod n$

Signature generation only: The computation time for ECDSA is known to be high compared to symmetric cryptographic methods like AES. The functionality has to be chosen carefully to hold the energy consumption in feasible bounds. The big gap in computation time between signature generation and verification makes the generation of a signature more suitable for the tag side of the RFID link.

Affine coordinates: Affine coordinates help to reduce the die size as they do not need to store a third coordinate during calculation like projective coordinates. A point is directly represented by the x and y coordinates with a duple of field elements. This method gives an initial advantage when

thinking of the final costs of the tag. Memory is expensive in particular on standard-cell based circuits because a flip-flop has roughly the size of six NAND-2 equivalents.

Free choice of domain parameters: ECDSA uses finite field operations in two fields: one bounded by the prime p , the other by the order n of the base point P . Fast reduction algorithms for Mersenne primes are possible, but not for a device also utilizing the general prime n . To let the controller device calculate field operations modulo n is not recommended, as this approach needs two implementations for the same field operation. Calculation intensive operations with long word lengths should not be requested from a controller device which is heavily optimized for communication and controlling tasks. To further reduce the chip area, to let the lengths of the domain parameters be freely chosen and to gain better utilization of the functional units, the decision was made to implement only general reduction algorithms. All feasible domain parameters for elliptic curves are therefore supported by this device.

III. RESISTANCE AGAINST CRYPTOGRAPHIC ATTACKS

There are several possible ways to attack an ECDSA device:

- Attack on the ECDLP to get the random number k of $(x_1, y_1) = kP$ of Alg. 1.
- Side channel attacks using SPA, DPA or others to attack the random number k or the private key d of Alg. 1.
- Physical attacks on the private key storage.

The best known algorithm to attack the ECDLP is the Pollard's rho attack described in [2] and [7] which takes about $O(\sqrt{n})$ point operations, where n represents the number of points in the used cyclic subgroup. An attack on the ECDLP is therefore computationally infeasible. If the algorithms for the elliptic curve point operations are not carefully chosen, some information about a secret may leak through the side channels like the current consumption or the EM radiation [8]. The ECDSA has two secrets which have to be protected: the private key d and the random number k . If the random number is known to the attacker, the private key could be calculated out of the generated signature with the Equ. 1.

$$d = r^{-1}(ks - e) \mod n \quad (1)$$

The point multiplication $(x_1, y_1) = kP$ can also be attacked with the help of the side-channel information. If the EC point multiplication algorithm behaves differently for ones and zeros in k , the random number can be recovered by looking at the power trace of the device. To avoid such a *simple power analysis (SPA)*, the **Montgomery addition ladder** for scalar point multiplication, described in the paper from Takagi [6] was implemented. Point addition and point doubling is implemented using the x-coordinate only.

To mount a successful differential power analysis (DPA) attack, an off-line model calculating the expected result has to be built. ECDSA uses a secret pseudo random number k as scalar factor during the EC point multiplication $(x_1, y_1) = kP$ which does not allow the creation of a unique model. The signature is finished with the term $k^{-1}(e + dr)$ where the secret key d and the random number k are each combined in one field operation with an unknown value. This is a possible

entry point for a DPA attack, but should also be a hard to solve problem. A successful DPA attack of the ECDSA device is therefore unlikely.

Physical attacks may have a chance to recover the private key, as the storage of the secret key is provided by the control device and not part of the processor.

IV. FINITE FIELD ALGORITHMS

Finite Field Multiplication

The Montgomery multiplication with integrated reduction is commonly used as field multiplication. The Montgomery algorithm can multiply two numbers in the Montgomery domain and avoids trial divisions. The original idea comes from Peter L. Montgomery and was published in [3].

A number in the Montgomery domain $\bar{x}$ is calculated with $\bar{x} = xR \mod p$, where x is the number in integer representation, with $R = 2^{k \cdot n}$, $n = 2 + (\lfloor \log(p) \rfloor + 1)/k$, $k = \#$ of bits to process at once. The Montgomery multiplication (MM) not only multiplies two variables, but also divides through R together with the reduction modulo p as shown in Equ. 2.

$$\bar{c} = \text{MM}(\bar{x}\bar{y}) = \frac{\bar{x}\bar{y}}{R} \mod p \quad (2)$$

The radix-4 version of Orup's Montgomery multiplication [1] arises some problems for this hardware implementation. The prime number dependent value $\widetilde{M}$ needs to be stored in hardware registers or as constant. Partial products need to be computed efficiently in one clock cycle without adding too much hardware resources. The solution for these problems is the presented modified Montgomery multiplication algorithm using Booth and Montgomery recoding.

Booth recoding for radix-4 moves a three bit window over the operand b to get the input bits for the partial field multiplication. It starts at the -1 'th bit, which is defined to be zero. The first two input blocks are then: $\{b_1, b_0, b_{-1}\}$, $\{b_4, b_3, b_2\}$. The recoding table can be found in [11], [10] and [4]. The Montgomery recoding scheme is used to find a multiplier for the modulus p in the range of $\{-2, -1, 0, 1\}$. Montgomery recoding is described in the paper from Hee-Kwan Son and Sang-Geun Oh [4]. The multiplier is chosen to make the two least significant bits of $S + qp$ zero before the right shift. The bit p_1 and the two least significant bits of S are used as input for the recoding function shown in [4].

Algorithm 2 Radix-4 Modified Montgomery multiplication

Require: $0 \leq a, b < 2p$, prime p , $n = 2 + (\lfloor \log(p) \rfloor + 1)/2$

Ensure: $-p \leq (ab)/R < p$

- 1: $S := 0$;
 - 2: **for** $i := 0$ to n **do**
 - 3: $b_{\text{booth}} := \text{booth_recode}(b_1, b_0, b_{-1})$;
 - 4: $q := \text{montgomery_recode}(S \mod 4, p_1)$;
 - 5: $S := (S + qp) \gg 2 + b_{\text{booth}}a$;
 - 6: $b := b \gg 2$
 - 7: **end for**
-

A. Finite Field Division

The division in a finite field is usually implemented with an inversion followed by a multiplication. The inverse x_1 of a finite field element a satisfies the equation $ax_1 \equiv 1 \pmod{p}$. The algorithm of choice is the Extended Euclidean Algorithm (EEA) [2]. The EEA finds the greatest common divisor (gcd) of u, v and the values x_1, x_2 with the help of the two equations $ax_1 + py_1 = u \pmod{p}$ and $ax_2 + py_2 = v \pmod{p}$. The value of y_1 and y_2 are not explicitly calculated and do not matter in an equation modulo p . The gcd is always one because v is a prime. x_1 then represents the inverse of a because the first equation reduces to $ax_1 \equiv 1 \pmod{p}$. The same is true for the second equation if v reaches one. The counting EEA algorithm is based on G. Lai's field inversion in [5].

Algorithm 3 Counting EEA using prime modulus

Require: $0 \leq a, b < p$, prime p

Ensure: $a/b \pmod{p}$

```

1:  $u := a; v := p; x := b; y := 0;$ 
2: while  $u$  not 1 do
3:   while  $(u \bmod 2) == 0$  do
4:      $u := u/2;$ 
5:      $x := (x + (x \bmod 2)p) \ggg 1;$ 
6:      $k := k - 1;$ 
7:   end while
8:   if  $k < 0$  then
9:      $u \leftrightarrow v; x \leftrightarrow y;$ 
10:     $k := -k;$ 
11:   end if
12:    $u = (u + v)/2;$ 
13:    $x = (x + y + (x_0 \otimes y_0)p)/2;$ 
14: end while

```

V. VLSI IMPLEMENTATION

The digital device was constructed involving several design stages, ending up in a synthesizable VHDL description. Cadence tools¹ were used to generate the semi-custom design for the CMOS standard-cell libraries from Austriamicrosystems (C35) and UMC (250 nm and 150 nm). The overall architecture was split into a datapath and a hardwired control unit. The configurable word length of the datapath allows processing of finite field operations in any length and therefore a parameterization of the key lengths on the ECC level. Carry save adders are used throughout the design to minimize and to sustain the critical path for all chosen word lengths. The two processing units of figure 2 and 3 were designed to allow a efficient calculation of the field inversion and field multiplication.

The hardwired control unit does not need to add wait states or NOP instructions during signature calculation. This *zero-overhead pipeline* allows a very energy-efficient processing. Figure 4 shows the block diagram of the SGD using 9 registers in the full word length for the overall system.

Fig. 2. Processing unit one (Core 1)

Fig. 3. Processing unit two (Core 2)

VI. RESULTS

Table I gives an overview of the area utilization with a 192-bit word length. The gate equivalents² for 0.35 μm standard-cells are shown. The exact run time mainly depends

SGD area	Cell quantity	Gate equivalents	Area [μm^2]
Datapath registers	1 749	10 417	572 970
Datapath comb.	8 023	11 740	645 736
<i>Datapath</i> Σ		21 769	1 197 341
Control registers	87	525	28 919
Control comb.	787	956	52 598
<i>Control</i> Σ		1 482	81 517
SGD Σ		23 656	1 301 081

TABLE I
COMPONENT DIE AREA FOR A 192-BIT WORD LENGTH

on the word size being used. All algorithms run with linear complexity on the presented hardware except the field inversion which is non-linear because of the redundant-to-binary conversion step in line 9 of Alg. 3. Table III shows the properties of the final circuit with 192-bit domain parameters. All area values are given as component area only, no routing overhead is considered³. The power simulation was done with Nanosim⁴ for the 0.35 μm CMOS standard-cell library from

²NAND gate with two inputs ($55 \mu\text{m}^2$)

³Target of this value is to give a gate count for comparison

⁴<http://www.synopsys.com/products/mixedsignal/nanosim/nanosim.html>

¹<http://www.cadence.com/>

Fig. 4. Overall system block diagram

Austriamicrosystems (C35) with a core voltage of 3.3V⁵. The average signature generation time for a 192-bit word length is 500k clock cycles.

SGD-192	Area	Cycle t.	Latency	Avg. curr.	Energy
Tech.	[mm ²]	[ns]	[ms]	[mA]	[mWs]
350 nm	1.30	12	6.0	42.73	0.846
250 nm	0.69	6	3.0	N/A	N/A
130 nm	0.15	5	2.5	N/A	N/A

TABLE II
RESULTS FOR THE SGD USING ECDSA OVER GF(P₁₉₂)

VII. RELATED WORK

This comparison of the SGD to other ECDSA devices involves chip area (in gate equivalents of 55 μm²), cycle count for signature generation and considerations about side-channel resistance. Column "ECDSA Part" shows which parts of the ECDSA generation are implemented, given by line numbers of Alg. 1.

ECDSA Devices	Gates	Cycles	SCA Considered	Field	ECDSA Parts
SGD-192	23.6k	502k	SPA & DPA	GF(p ₁₉₂)	2, 3, 5
ECC P [15]	23.8k	677k 426k	SPA & DPA	GF(p ₁₉₂) GF(2 ¹⁹²)	2, 3, 5 2
ECC P [14]	19k	527k	SPA & DPA	GF(2 ¹⁹³)	2
8051+ECAU [13]	29k	1 416k	No	GF(2 ¹⁹²)	2
SGD-160	19k	362k	SPA & DPA	GF(p ₁₆₀)	2, 3, 5
PACS [12]	46k	134k	No	GF(2 ¹⁶³)	2
MALU [17]	5.3k	353k	SPA & DPA	GF(2 ¹⁶³)	2

TABLE III
COMPARISON TO OTHER ECDSA DEVICES

J. Wolkerstorfer presented a dual-field arithmetic unit in [15] for GF(p) and GF(2^m). Paar and Kumar showed in [14] a way to calculate the scalar multiplication kP over GF(2^m) with their ECC processor. In [13] an approach for an ECC crypto extension using a 8051 processors was shown. The PACS from [12] for medical applications gives a very fast processor using windowing techniques on EC layer, but utilizing the most gates. The MALU [17] is a very small

circuit for scalar point multiplication over GF(2^m) but will need another microcontroller to finish the digital signature like all GF(2^m) approaches.

VIII. CONCLUSION

This paper presented an RFID processor implementing an ECDSA generator using elliptic curves defined over GF(p). Points on the curve are represented with affine coordinates. The RFID tag further needs a secure key storage, a one-bit PRNG and SHA-1 functionality to implement copy protection and authentication. This makes this processor a nearly complete signature generation device missing only a way to calculate SHA-1 in the circuit. The target of this approach is to avoid an additional microcontroller on the tag which is only possible with a GF(p) device. The security of the device including possible side-channel attacks was analyzed. The GF(p) architecture only uses 9 full word length registers and a hardwired control unit without pipelining overhead. The energy and current consumption was estimated with the Nanosim tool using the fastest possible clock cycle time and the 0.35 μm CMOS standard-cell process.

REFERENCES

- [1] Holger Orup, *Simplifying Quotient Determination in High-Radix Modular Multiplication*, 1995
- [2] D.Hankerson, A.Menezes, S.Vanstone, *Guide to Elliptic Curve Cryptography*, Springer 2004
- [3] Peter L. Montgomery, *Modular multiplication without trial division*, Mathematics of Computation, vol. 44, no. 170, pp 519-521, 1985
- [4] Hee-Kwan Son, Sang-Geun Oh, *Design and Implementation of Scalable Low-Power Montgomery Multiplier*, 2004
- [5] Gerald Lai, *Analysis of Modular Inverse GF(p) Implementations*, <http://islab.oregonstate.edu/koc/ece679/project/2004/lai.pdf>, 2004
- [6] Tetsuya Izu Bodo Moeller Tsuyoshi Takagi, *Improved Elliptic Curve Multiplication Methods*, Resistant against Side Channel Attacks, IN-DOCRYPT 2002
- [7] Ian F.Blake, Gadiel Seroussi, Nigel P.Smart, *Elliptic Curves in Cryptography*, Cambridge University Press 1999
- [8] Ian F.Blake, Gadiel Seroussi, Nigel P.Smart, *Advances in Elliptic Curve Cryptography*, Cambridge University Press 2005
- [9] Wenbo Mao, *Modern Cryptography*, Prentice Hall PTR, 2nd Printing 2004
- [10] Johannes Wolkerstorfer, *Hardware Aspects of Elliptic Curve Cryptography*, PhD, GRAZ UNIVERSITY OF TECHNOLOGY 2004
- [11] Franz Fürbass, *ECC Processor with Low Die Size for RFID Applications*, Master's Thesis, GRAZ UNIVERSITY OF TECHNOLOGY 2006
- [12] Jin Park, Jeong-Tae Hwang, Young-Chul Kim, *FPGA and ASIC Implementation of ECC Processor for Security on Medical Embedded System*, IEEE Computer Society, ICITA'05
- [13] M.Koschuch, J.Lechner, A.Weitzer, J. Großschädl, A.Szekely, S.Tillich, and J.Wolkerstorfer, *Hardware/Software Co-Design of Elliptic Curve Cryptography on an 8051 Microcontroller*, CHES 2006
- [14] Sandeep S.Kumar, Christof Paar, *Are standards compliant Elliptic Curve Cryptosystems feasible for RFID?*, RFIDSec 2006
- [15] J. Wolkerstorfer, *Is Elliptic-Curve Cryptography Suitable to Secure RFID Tags?*, 2005
- [16] L.Batina, J.Guajardo, T.Kerins, N.Mentens, P.Tuyts, and I.Verbaauwhede, *Public-Key Cryptography for RFID-Tags*, RFIDSec 2006
- [17] Kazuo Sakiyama, Lejla Batina, Nele Mentens, Bart Preneel and Ingrid Verbaauwhede, *Small-footprint ALU for public-key processors for pervasive security*, RFIDSec 2006

⁵All power values include routing overhead from a test layout