ECC Is Ready for RFID – A Proof in Silicon

Daniel Hein[1], Johannes Wolkerstorfer[2], and Norbert Felber[1]

[1] Swiss Federal Institute of Technology Zürich, IIS
Gloriastrasse 35, 8092 Zürich, Switzerland
Daniel.Hein@gmx.at, felber@iis.ee.ethz.ch
[2] Graz University of Technology, IAIK
Inffeldgasse 16a, 8010 Graz, Austria
Johannes.Wolkerstorfer@iaik.tugraz.at

Abstract. This paper presents the silicon chip ECCon[1], an Elliptic Curve Cryptography processor for application in Radio-Frequency Identification. The circuit is fabricated on a 180 nm CMOS technology. ECCon features small silicon size (15K GE) and has low power consumption (8.57 μW). It computes 163-bit ECC point-multiplications in 296k cycles and has an ISO 18000-3 RFID interface. ECCon's very low and nearly constant power consumption makes it the first ECC chip that can be powered passively. This major breakthrough is possible because of a radical change in hardware architecture. The ECCon datapath operates on 16-bit words, which is similar to ECC instruction-set extensions. A number of innovations on the algorithmic and on the architectural level substantially increased the efficiency of 163-bit ECC. ECCon is the first demonstration that the proof of origin via electronic signatures can be realized on RFID tags in 180 nm CMOS and below.

Keywords: Radio-Frequency Identification (RFID), Elliptic curve cryptography (ECC), Anti-Counterfeiting, Modular Multiplication.

1 Introduction

Counterfeiting is an increasing problem in the industry. Estimates assume more than 200 bn. US$ of fake products in 2005 [Org07]. In total, counterfeited products might amount up to one tenth of the total industry production. Besides legal measures, technical approaches are required to fight product piracy. This is in particular desirable for goods that affect the health of humans directly like pharmaceuticals or spare parts in aviation.

RFID (Radio-Frequency Identification) technology, which labels products with tiny chips that are powered over an air interface, is able to proof the authenticity of goods. In its simplest form, RFID tags assign a unique number to every item. This ID can be used to track goods from production to the end consumer. Central databases and online applications can verify the e-pedigree of products. The concept of e-pedigree suffers from scalability issues and assumes that unique IDs

[1] A portmanteaux of ECC and economical.

R. Avanzi, L. Keliher, and F. Sica (Eds.): SAC 2008, LNCS 5381, pp. 401–413, 2009.

cannot be copied. The unprotected wireless communication of RFID systems in HF or UHF frequencies allows access to unique IDs without necessitating a line of sight. Also, illegitimate readers can obtain IDs from distances of 1 m (HF) to 15 m (UHF). Thus, the concept of preventing counterfeiting by using RFID technology requires cryptography to guarantee the uniqueness of a tag. A major step to bring cryptography to RFID is the landmark work of Feldhofer et al.: They showed that challenge-response authentication is useful [FDW04] and that symmetric ciphers – in particular the AES (Advanced Encryption Standard) – can be realized on RFID tags [FWR05]. AES can fulfill the hard requirements of small silicon area and minute power consumption.

The use of symmetric cryptography necessitates sophisticated key management. Its main drawback is its limitation to closed systems, where all verifiers have to be trusted and where all potential parties are known in advance. It is not realistic for open-loop systems, like global logistics and supply chain management, where this is not the case.

Asymmetric crypto, where tags possess a private key and verifiers can obtain an authentic public key, can overcome these limitations. Asymmetric crypto allows challenge-response authentication without access to an online database. Verifiers only need the public key of the device claiming a certain identity.

Although asymmetric crypto has very nice properties regarding its applicability in worldwide open-loop systems, it imposes much more effort for realization in silicon. Requirements for silicon area are roughly five times higher; power consumption also quadruples at least. Computation times are several hundred times longer than for symmetric crypto. Therefore, most asymmetric approaches for use in RFID are based on elliptic curve cryptography, which exhibits the lowest hardware requirements among standardized and secure asymmetric algorithms.

In this work, we scrutinize the requirements of passively powered RFID systems and introduce a new hardware architecture for computing ECC operations with the smallest footprint and requiring the smallest power consumption. The architecture processes ECC operations on a word length much smaller than the actual element size of 163 bits. This allows a very compact datapath, which shows excellent power characteristics: On the one hand, the proposed architecture avoids large shares of clocking power and on the other hand, it allows very flat and low power profiles which are desired for passively powered systems.

2 State of the Art

Modular multiplication is the dominating finite-field operation for computing an ECC point multiplication $k \cdot P$. Our work centers on the efficient implementation of ECC over $\mathbb{F}_{2^{163}}$ (Curve B-163 in [Nat00]). Hence, most considerations will be done for this field. Computing the point multiplication $k \cdot P$ over $\mathbb{F}_{2^{163}}$ involves 163 point doublings and additions when using Montgomery's point ladder [Mon87]. This is a widely used algorithm[Wol05, KP06, BMS$^+$06, FW07, BBD$^+$08] – we use it too. It is fast and has good properties to prevent side-channel analysis. In total, roughly 1000 finite-field multiplications have to be computed.

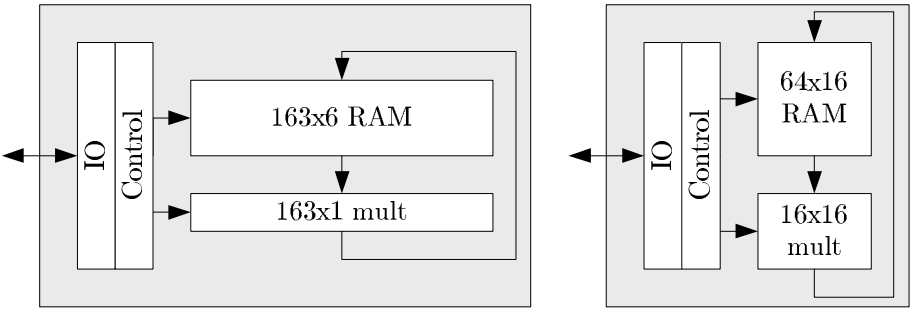

Fig. 1. Classical (bit-serial) vs. proposed architecture

The art of building ECC hardware centers on the efficient implementation of finite-field multiplication in hardware. Most ECC publications for low-resource requirements use bit-serial multipliers [Wol05, SBM+06, LV07, FW07]. Bit-serial multipliers compute $c = a \cdot b$ by scheduling the operand a at full word size and multiplying it with b bit-by-bit: $c = \sum_{i=0}^{162} a \cdot b_i 2^i$.

The classical architecture when using bit-serial multipliers for ECC processors is shown on the left of Figure 1. A datapath unit *163×1 mult* computes the partial products $a \cdot b_i$ and accumulates them. Usually, modular reduction is interleaved to avoid accumulators of double size. Intermediate 163-bit values are stored in a *RAM* unit, which has to hold at least six 163-bit words during an EC point multiplication.

[SBM+06] investigated the impact of the digit size on power consumption and area usage. Generally, bit-serial multipliers are more power efficient, whereas the digit-serial type is more energy efficient. Lee et al. [LV07] present an optimized architecture based on the so-called Modular Arithmetic Logic Unit [BMS+06]. It uses a common-Z coordinate system for representing EC points to minimize memory requirements. Storage of intermediate values is usually the main contributor to area (roughly 66%). Area has a direct impact on the production cost of an integrated circuit.

Fürbaß et al. [FW07] departs from the pure digit-serial multiplier approach and analyzes the ramifications of using an inversion unit in conjunction with affine coordinates for operation over $\mathbb{F}_p$. Inversion is usually avoided because its computation is either much slower than multiplication or it consumes considerable silicon area. Affine coordinates reduce the memory requirements and decrease the number of cycles required to perform a point multiplication at the expense of a more complex datapath and therefore higher power consumption.

3 Architecture

The architecture presented in this article schedules both operands of the multiplication $a \cdot b$ in words of 16 bits – called digits. It thus differs totally from conventional approaches found in literature. A digit size of 16 bits leads to a

datapath of roughly the same area as a bit-serial datapath for 163×1 multiplication. It is shown on the right side of Figure 1. The core component of the datapath is a 16×16 multiplier. The storage requirements for the new approach are the same as for bit-serial approaches. The data width of the RAM circuit is adjusted to fit the 16-bit datapath. RAM is organized as 16-bit wide memory with 64 entries. In comparison, the bit-serial approach and ours require nearly the same hardware resources. Also the timing characteristics are very similar.

The main difference of our digit-level architecture to the classical bit-serial architecture is the power consumption. Bit-serial architectures clock on average $2 \cdot 163 = 326$ registers per clock cycle. Our digit-level architecture clocks roughly $4 \cdot 16 = 64$ registers. Hence, it needs just one fifth of the power for clocking. The power consumption of the combinational logic in our datapath is higher. The longer combinational path causes higher signal activity. Nevertheless, the approach promises a much better power characteristic because it uses a much larger share of the power budget for computation than for clocking.

The second design choice of great impact is the restriction of ECCon to support only a single 163-bit curve over the binary extension field $\mathbb{F}_{2^{163}}$. This allows numerous optimizations, which minimize the required area, improve the running time of ECC operations and decrease the power consumption.

The selection of a binary extension field simplifies the arithmetic unit as addition is equivalent to an XOR operation. The 163-bit elements limit the size of the required memory while providing reasonable security. Support for only one finite field allows efficient modular multiplication. A tailored interleaved reduction algorithm for $\mathbb{F}_{2^{163}}$ is presented in §3.4.

3.1 Word Level Operations in $\mathbb{F}_{2^{163}}$

The small word width of the datapath necessitates splitting up finite-field operations into operations on 16-bit digits. Research with respect to this has been undertaken in the context of $\mathbb{F}_p$[Gro02] and $\mathbb{F}_{2^m}$ [GK03] instruction-set extensions for general-purpose processors. Großschädl et al. propose a Multiply ACumulate (MAC) architecture for a word-level instruction-set extension. The ECCon processor uses the same approach: Main component of the datapath is a 16×16 MAC unit.

The point multiplication requires addition and multiplication in $\mathbb{F}_{2^{163}}$. Both necessitate to split up the 163-bit operands into 11 digits of 16 bits each. In every cycle, one digit can be fetched from memory. As soon, as two digits are available to the Arithmetic Logic Unit (ALU), either an addition, a multiplication or a MAC operation is performed. The result is then stored in an accumulator. The lower 16 bits of the accumulator can be written to memory. Thus, digit for digit a 163-bit operation is performed.

Addition is a good example for this concept. First, a digit of the input a is loaded into the ALU. Then the corresponding digit of b is added (XORed). While the next digit is fetched, the result digit is stored. This is repeated 11 times to execute one 163-bit addition. One addition takes 24 clock cycles (see tab. 1).

Table 1. Performance results

Operations	Cycles	Operations	Cycles
Addition	24	Squaring	49
Multiplication	251	Point multiplication	296,299

The two prevailing integer multiplication algorithms are operand-scanning-form and product-scanning-form multiplication (Comba multiplication). Both depend on an inner multiplication operation $a \cdot b \oplus c$, which is the reason why a MAC architecture was chosen. The ECCon processor employs the Comba algorithm. It computes the result one digit at the time which minimizes memory write cycles, at the expense of requiring an additional 16-bit adder. Furthermore, smart ordering of operand loading allows to decrease the number of memory read operations to a minimum.

A multiplication of two 163-bit elements in $\mathbb{F}_{2^{163}}$ would produce a 325-bit result. Modular reduction $a \cdot b = c \equiv d \, mod \, f(z)$ reduces the product to a 163-bit representation, where $f(z)$ is the irreducible polynomial. Two possible options for reduction exist. The first, interleaved reduction, performs the reduction during multiplication. The alternative is to first compute the 325-bit product and then reduce it. This has the severe drawback that it requires 162 bits of additional storage. An efficient algorithm capable of implementing the interleaved reduction on the 16-bit ALU will be detailed in §3.4.

In conformance with the state-of-the-art implementations, ECCon employs Fermat's little theorem to realize inversion. It requires nine multiplications and 162 square operations. This is not fast enough to allow using affine coordinates, therefore a projective version of the point multiplication requiring only one inversion is applied.

3.2 Choice of Word Size

The choice of a width of 16 bits for the datapath is an outcome of assessing various cost metrics for different datapath sizes. Cost is determined by the area usage, the power consumption and the clock cycles for a point multiplication. The critical path is not considered because the circuit runs well below the maximum clock frequency.

Figure 2 depicts a comparison of the ALU architecture (see also §3.3), synthesized for different bit-widths. The four graphs represent different variants of area, cycle and power products ($A(rea) \cdot C(ycles) \cdot P(ower)$). The values are computed from normalized results to allow a comparison over the different metrics. Bit-widths below 8 and above 31 were not considered because they are either far too slow ($>$ 1 mio. cycles) or too large circuits ($>$ 5500 gate equivalents).

On a first glance, no obvious bit-width presents itself. Considering the $A \cdot C \cdot P$ and $A \cdot C \cdot P^2$ products, starting with the transition between 15 and 16 bits, the values start to increase steeply. The $A \cdot C^2 \cdot P$ and $A \cdot C^2 \cdot P^2$ on the other hand seem to have a local minimum point at 15 bit. As the actual area, cycle and power values for a 15-bit datapath lie well within the aspired limits, this

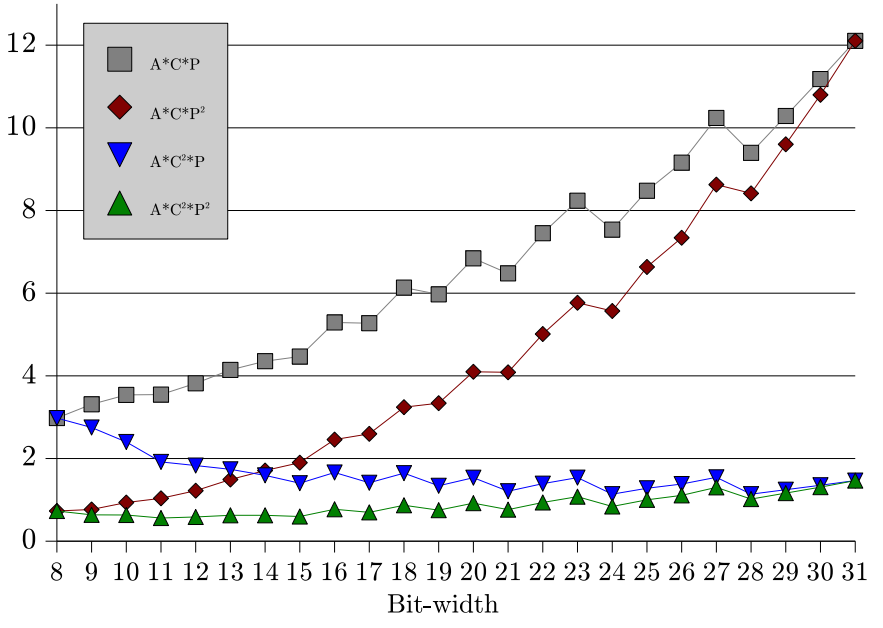

Fig. 2. Datapath bit-width comparison

bit-width was deemed optimal. Due to the fact that the RFID protocol is byte oriented and the difficulties in converting 15-bit digits to this interface, a 16-bit datapath was chosen for implementation.

3.3 MAC Unit for Interleaved Reduction

Figure 3 illustrates the top level architecture of the ALU of the ECCon processor. **B**, **RC**, $\mathbf{ACC_L}$ and $\mathbf{ACC_H}$ denominate registers. $\mathbf{ACC_L}$ and $\mathbf{ACC_H}$ compose the accumulator **ACC** which also supports a shift right-by-8 and shift right-by-16 operation. **MC**, the multiplication carry register, has the capability to select between two different inputs.

The multiplier unit $\otimes$ computes either the product of the input I times the factor register **B** or $\mathbf{ACC_L}$ times the reduction polynomial $r(z)$ (cf. 3.4). The multiplier unit uses a 16×16 polynomial array multiplier consisting of 256 AND gates and a $\mathbb{F}_2$ tree adder built from 240 XOR gates.

The select-and-add unit **SAA** contains two 16-bit $\mathbb{F}_2$ adders, implemented by two 16-bit XOR gate arrays. The input select of the higher adder (Output **D**H) allows to choose between $\mathbf{ACC_H}$ and **MC** for its first input and **Mul**H and I for its second input. Both inputs are maskable. This is implemented with AND gates. The first input of the lower adder (Output **D**L) can be chosen from **RC**, **Mul**L and I while the second input may be selected from $\mathbf{ACC_H}$, **MC** and $\mathbf{ACC_L}$. Again, both inputs to the adder are maskable.

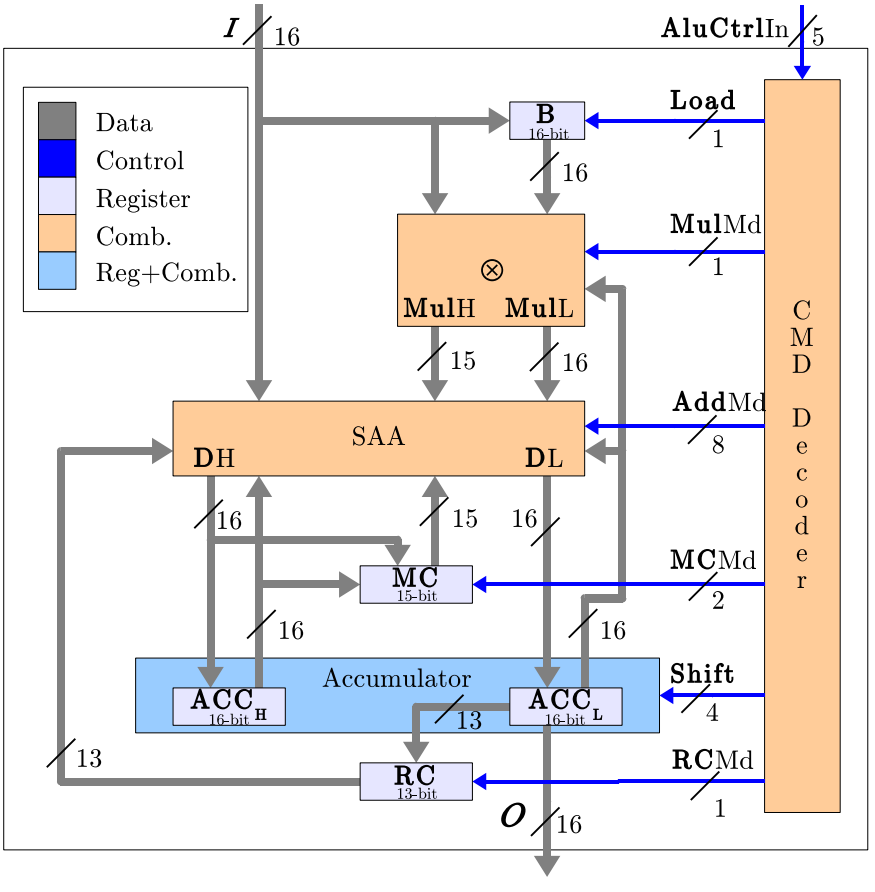

Fig. 3. ALU top level architecture

3.4 Multiplication with Interleaved Reduction

The Comba multiplication algorithm is utilized to perform the multiplication. To execute the modular reduction, the following idea is applied. We split the irreducible polynomial $f(z) = z^{163} + z^7 + z^6 + z^3 + 1$ as $z^{163} + r(z)$ and call $r(z) = z^7 + z^6 + z^3 + 1$ the reduction polynomial. The product polynomial $c(z) = a(z) \cdot b(z) | \, deg\{c(z)\} \leq 2m - 2$ is thus congruent to

$$\begin{aligned} c(z) &= c_{2m-2}z^{2m-2} + \cdots + c_m z^m + c_{m-1}z^{m-1} + \cdots + c_1 z + c_0 \\ &\equiv (c_{2m-2}z^{m-2} + \cdots + c_m)r(z) + c_{m-1}z^{m-1} + \cdots + c_1 z + c_0 \ (mod\ f(z)). \end{aligned}$$

So, by multiplying $r(z)$ and $c_H = (c_{324}z^{161} + \cdots + c_{163})$, and adding the result to $c_L = c_{162}z^{162} + \cdots + c_1 z + c_0$, a new temporary output is derived which only

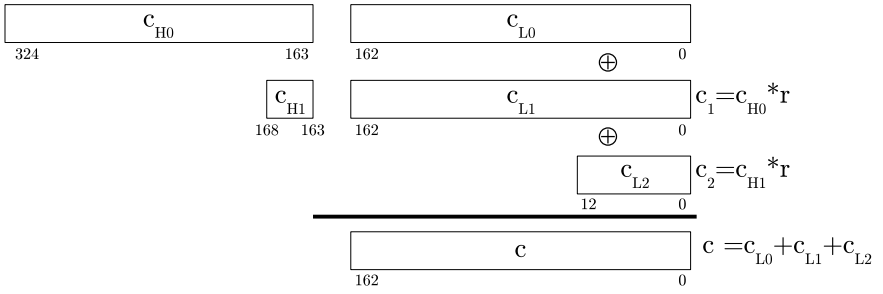

Fig. 4. Modular reduction

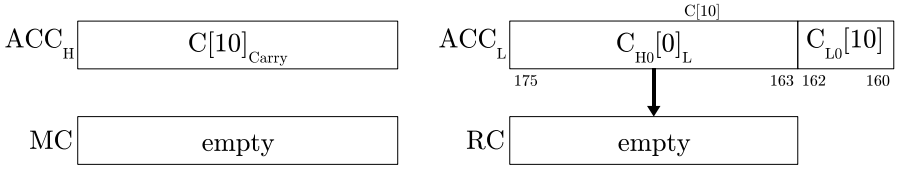

Fig. 5. Saving the first 13-bit of C_{H^0}

has a degree of $m = 168$. The process is then repeated once more and the fully reduced result is computed. Figure 4 illustrates the procedure.

Algorithm 1 presents the necessary steps to perform a multiplication with interleaved reduction. A line in the algorithm illustrates all the operations executed in one clock cycle. Line number five will serve as example to facilitate understanding of the nomenclature.

$T[i-1] \leftarrow \mathbf{ACC_L}$ signifies that the output of the ALU is to be stored in the (i-1)th digit of memory element T. $\mathbf{ACC} \leftarrow \mathrm{i} - \mathcal{INDEX}(\mathrm{i,j}) \cdot \mathbf{B} \oplus \mathbf{ACC_H}$ denotes that the result of the MAC operation is stored in the accumulator. In this case the ALU input is B[i $- \mathcal{INDEX}(\mathrm{i,j})$]. The function $\mathcal{INDEX}(i,j)$ computes the optimal loading sequence for the input operands, thus minimizing memory read operations. $\mathcal{INDEX}(i,j)$ is implemented by a look-up table.

For a multiplication $c = a \cdot b$, $a, b \in \mathbb{F}_{2^{163}}$, the 11^{th} partial product computed with the product scanning form contains the polynomial $c_{175}z^{175} + \ldots + c_{160}z^{160}$. In accordance with the nomenclature used in Figure 4 the polynomial $c_{162}z^{162} + \ldots + c_{160}z^{160}$ is the last digit of C_L^0 ($C_L^0[10]$), while $c_{175}z^{175} + \ldots + c_{163}z^{163}$ is the first part of C_{H^0}.

The lower three bits representing $C_L^0[10]$ are stored in the last digit of the memory element that will contain the final product. The lower part of C_{H^0} is saved in temporary variable **RC**. This is illustrated in Figure 5.

The next partial product ($C[11]$) that is ascertained will again be split at the 3-bit boundary. This time the higher 13 bits ($c_{191}z^{191} + \ldots + c_{176}z^{176}$) are written

Algorithm 1. Comba multiplication with interleaved reduction

Input: $a, b \in \mathbb{F}_{2^m}$
Output: $c = a \cdot b \, mod \, f, c \in \mathbb{F}_{2^m}$

1 $\mathbf{B} \leftarrow$ A[0];
2 $\mathbf{ACC} \leftarrow$ B[0] $\cdot \mathbf{B} \oplus \mathbf{ACC}$;
3 $i \leftarrow 0$, $j \leftarrow 0$;
4 **for** $i \leftarrow 1$ **to** $t-1$ **do**
5 $\quad T[i-1] \leftarrow \mathbf{ACC_L}$; $\mathbf{ACC} \leftarrow$ B[i $- \mathcal{INDEX}$(i, j)] $\cdot \mathbf{B} \oplus \mathbf{ACC_H}$;
6 $\quad$ **for** $j \leftarrow 1$ **to** i **do**
7 $\quad\quad \mathbf{B} \leftarrow$ A[$\mathcal{INDEX}$(i, j)];
8 $\quad\quad \mathbf{ACC} \leftarrow$ B[i $- \mathcal{INDEX}$(i, j)] $\cdot \mathbf{B} \oplus \mathbf{ACC}$;
9 $T[t-1] \leftarrow \mathbf{ACC_L}$; $\mathbf{ACC} \leftarrow \mathbf{ACC} \gg 16$; $\mathbf{MC} \leftarrow 0$; $\mathbf{RC} \leftarrow \mathbf{ACC_L} \gg S_R$;
10 $i \leftarrow 0$, $j \leftarrow 0$;
11 $\mathbf{B} \leftarrow$ A[$\mathcal{INDEX}$(i, j)];
12 **for** $i \leftarrow 0$ **to** $t-2$ **do**
13 $\quad$ **if** $i \neq 0$ **then**
14 $\quad\quad C[i-1] \leftarrow \mathbf{ACC_L}$; $\mathbf{ACC} \leftarrow$ B[(t + i) $- \mathcal{INDEX}$(i, j)] $\cdot \mathbf{B} \oplus \mathbf{MC}$; $\mathbf{MC} \leftarrow \mathbf{ACC_H}$;
15 $\quad$ **else**
16 $\quad\quad \mathbf{ACC} \leftarrow$ B[(t + 1) $- \mathcal{INDEX}$(i, j)] $\cdot \mathbf{B} \oplus \mathbf{ACC}$;
17 $\quad$ **for** $j \leftarrow 1$ **to** $(t-2) - i$ **do**
18 $\quad\quad \mathbf{B} \leftarrow$ A[$\mathcal{INDEX}$(i, j)];
19 $\quad\quad \mathbf{ACC} \leftarrow$ B[(t + 1) $- \mathcal{INDEX}$(i, j)] $\cdot \mathbf{B} \oplus \mathbf{ACC}$;
20 $\quad \mathbf{ACC_H} \leftarrow$ T[i] $\oplus \mathbf{MC}$; $\mathbf{ACC_L} \leftarrow \{(\mathbf{ACC_L} \ll S_L) | \mathbf{RC}\}$; $\mathbf{MC} \leftarrow \mathbf{ACC_H}$; $\mathbf{RC} \leftarrow \mathbf{ACC_L} \gg S_R$;
21 $\quad \mathbf{ACC} \leftarrow \mathbf{ACC_L} \cdot r(z) \oplus \mathbf{ACC_H}$;
22 $C[t-2] \leftarrow \mathbf{ACC_L}$; $\mathbf{ACC_H} \leftarrow 0$; $\mathbf{ACC_L} \leftarrow \mathbf{MC}$; $\mathbf{MC} \leftarrow \mathbf{ACC_H}$;
23 $\mathbf{ACC_H} \leftarrow 0 \oplus \mathbf{MC}$; $\mathbf{ACC_L} \leftarrow \{(\mathbf{ACC_L} \ll S_L) | \mathbf{RC}\}$; $\mathbf{MC} \leftarrow \mathbf{ACC_H}$; $\mathbf{RC} \leftarrow \mathbf{ACC_L} \gg S_R$;
24 $\mathbf{ACC} \leftarrow \mathbf{ACC_L} \cdot r(z) \oplus \mathbf{ACC_H}$;
25 $\mathbf{ACC_H} \leftarrow \mathbf{MC}$; $\mathbf{ACC_L} \leftarrow$ T[i] $\oplus \mathbf{ACC_L}$; $\mathbf{MC} \leftarrow \mathbf{ACC_H}$; $\mathbf{RC} \leftarrow \mathbf{ACC} \gg S_R$;
26 $C[i] \leftarrow \mathbf{ACC_L}$; $\mathbf{ACC_H} \leftarrow$ C[0]; $\mathbf{ACC_L} \leftarrow \mathbf{RC}$;
27 $\mathbf{ACC} \leftarrow \mathbf{ACC_L} \cdot r(z) \oplus \mathbf{ACC_H}$;
28 $C[0] \leftarrow \mathbf{ACC_L}$; $\mathbf{ACC} \leftarrow \mathbf{ACC} \gg 16$;

to a second temporary space **MC**. The lower three bits are then combined with the 13 bits carried over from the last round which are currently stored in **RC**. Together they constitute the 16-bit digit $c_{178}z^{178} + \ldots + c_{163}z^{163}$, the first digit of C_H^0. This digit is restored in the accumulator (cf. Figure 6).

In the next step it is multiplied with $r(z)$. The lower 16 bit of the result in $\mathbf{ACC_L}$ are added to $C_L^0[0]$. The carry of the multiplication in $\mathbf{ACC_H}$ is than swapped with the content of **MC**. Thus, **MC** alternately stores the carry of the normal multiplication and the one of the reduction multiplication. Figure 7 corresponds to this state. Then the next partial product ($C[12]$) is computed, and the process is repeated. Thus, the interleaved multiplication and reduction

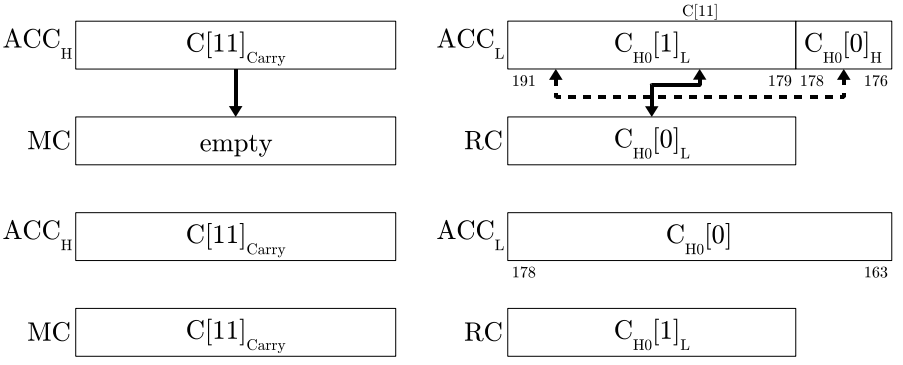

Fig. 6. Restoring C_{H^0} and saving the multiplication carry

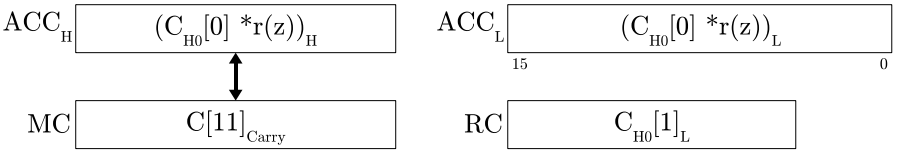

Fig. 7. Result of the first reduction multiplication

operation is performed step by step as the partial products of C_{H^0} and after that C_{H^1} become available.

4 Results and Comparison

The ECCon processor was fabricated using the UMC L180 GII 1P/6M 1.8V/3.3V CMOS technology. The ECC core consists of a 7×163-bit memory, the ALU introduced in §3 and a control unit that is capable of performing an EC point multiplication. It employs a fully registered, two phase handshake enabled 8-bit bus interface to connect to an ISO 18000-3-1 [ISO04] compatible RFID front-end.

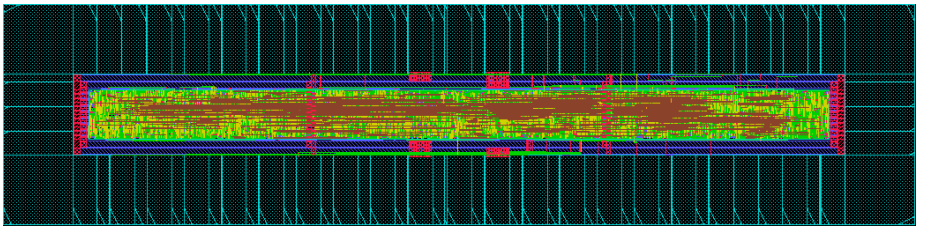

Fig. 8. The tape-out ready layout of the ECCon processor

Table 2. Synthesis results of the ECC core

Technology	Area [μm^2]	Area [GE]	Max. Frequency [MHz]	Power @ 106 KHz [μW]	Voltage [V]
UMC L180	128,098	13,250	46	8.57	1.8V
AMS C35	727,090	13,320	20	54.7	2.5V

Table 3. Area and cycle comparison

	Area [GE]	Runtime [Cycles]	Field	Bits	Technology
[BBD+08]	12,876	80K	$\mathbb{F}_{2^{163}}$	163	INF 220nm
ECCon w/o k	11,904	296K	$\mathbb{F}_{2^{163}}$	163	UMC 180nm
[LV07]	13,182	314K	$\mathbb{F}_{2^{163}}$	163	TSMC 180nm
[KP06]	15,094	430K	$\mathbb{F}_{2^{163}}$	163	AMI 350nm
[Wol05]	23,000	426K	$\mathbb{F}_{2^{191}}$	191	AMS 350nm
[FW07]	23,656	502K	$\mathbb{F}_{p_{192}}$	192	AMS 350nm
[OScE04]	30,333	545K	$\mathbb{F}_{p_{(2^{167}+1)/3}}$	165	TSMC 130nm

The integrated RFID front-end is purely digital in nature and lacks an analog air interface.

The chip has a core area of 219,897 μm^2 with a utilization of approximately 70%. The minimum core area amounts to 151,126 μm^2 or 15,630 GE. An overhead of 654 GE is incurred by circuits for production testing, which reduces the core area required for the RFID front-end and the ECC processor to 14,976 GE. The circuit achieves a maximum frequency of 46 MHz and the ECC core has a power consumption of 8.57 μW at 106 kHz. This frequency was chosen because it is $\frac{1}{128}$ of the 13.56 MHz carrier signal used by the ISO-18000-3-1 RFID standard.

Table 2 presents the synthesis results of the ECC core without the RFID front-end and the test hardware. The power consumption value of the UMC 180 nm variant was obtained by measuring the fabricated IC.

ECCon was also synthesized, placed and routed in the AMS c35b4 0.35 μm CMOS technology. A power simulation with Synopsys© NanoSim was performed on the placed & routed layout. The simulated mean current, the determinative factor for an RFID application is 21.88 μA at $2.5V$. During a whole EC point multiplication it varies only by 10%. This is important because high fluctuations disrupt the communication channel.

Table 3 tries to compare the ECCon processor to different architectures. The area for the ECCon processor in this table includes only a 6×163 bits memory. This was done to allow a fair comparison, because most related work does not include storage for the key k. [LV07], [KP06] and [OScE04] are capable to perform the point multiplication, whereas [Wol05] and [FW07] implement all operations of the ECDSA standard, except hashing and random number generation. Bock et al. ([BBD+08]) implement an ISO-18000-3-1 compatible RFID tag and a Diffie-Hellman key exchange based authentication protocol employing a digit-serial multiplier.

Table 4. Power consumption comparison

	Power [μW]	I_{mean} [μA]	f [kHz]	Tech.	Obtained by
ECCon	54.7	21.88	106	AMS350	NanoSim power simulation
[BBD+08]	96.4	n.a.	847.5	INF 220	PowerTheater power simulation
ECCon	67.23	37.35	847.5	UMC180	measured
[FW07]	141	42.73	106	AMS350	NanoSim power simulation
[BMS+06]	20 - 30	n.a.	500	CMOS130	estimated, MALU only
[OScE04]	990	n.a.	20000	TSMC130	plain synthesis results assumed

Table 4 compares the power consumption of different ECC implementations. It is important to grade the quality of these values. In this table measurement and power simulation are the most accurate, followed by synthesis results. Estimations are the most inaccurate. The power consumption of [BBD+08] is similar to that of ECCon due to their efficient latch based memory unit. They dissipate 88% of the total power in the ALU, whereas the ECCon ALU requires only 48%. Their ALU is also approximately three and a halve times larger.

5 Conclusions

This article presented a fully functional implementation of ECC over $\mathbb{F}_{2^{163}}$ on a 180 nm CMOS process. The so-called ECCon processor has a footprint of 15k GE, which is the smallest reported ECC solution so far, considering that this number also includes overhead for place-and-route, a hardwired state machine for controlling the roughly 300,000 clock cycles of one ECC operation, and a digital interface for communicating via the ISO-18000-3-1 standard. The power consumption obtained by measurement is 8.57 μW.

The ECCon processor is one of the first complete hardware solutions that can compute ECC point multiplications while fulfilling the harsh requirements of passively powered RFID tags. It will help to realize a secure Internet of things, where goods have an electronic identity that can be proved.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments on an earlier version of this paper. For their help in fabricating the ECCon processor the authors also thank the staff of the Microelectronics Design Center at the Department of Information Technology and Electrical Engineering of the Swiss Federal Institute of Technology Zürich.

This research has been partially supported by the European Union Information Society Technologies (EU IST) FP6 project Collaboration @ Rural (IST-FP6-34921).

References

[BBD+08] Bock, H., Braun, M., Dichtl, M., Hess, E., Heyszl, J., Kargl, W., Koroschetz, H., Meyer, B., Seuschek, H.: A milestone towards RFID products offering asymmetric authentication based on elliptic curve cryptography. In: Workshop on RFID Security, RFIDsec (2008)

[BMS+06] Batina, L., Mentens, N., Sakiyama, K., Preneel, B., Verbauwhede, I.: Low-cost elliptic curve cryptography for wireless sensor networks. In: Proceedings of Third European Workshop on Security and Privacy in Ad hoc and Sensor Networks (2006)

[FDW04] Feldhofer, M., Dominikus, S., Wolkerstorfer, J.: Strong authentication for RFID systems using the AES algorithm. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 357–370. Springer, Heidelberg (2004)

[FW07] Fürbass, F., Wolkerstorfer, J.: ECC processor with low die size for RFID applications. In: Proc. IEEE International Symposium on Circuits and Systems, ISCAS (2007)

[FWR05] Feldhofer, M., Wolkerstorfer, J., Rijmen, V.: AES implementation on a grain of sand. IEEE Proceedings on Information Security 152, 13–20 (2005)

[GK03] Großschädl, J., Kamendje, G.-A.: Instruction set extension for fast elliptic curve cryptography over binary finite fields GF(2^m). In: Proc. IEEE International Conference on Application-Specific Systems, Architectures, and Processors (2003)

[Gro02] Großschädl, J.: Instruction set extension for long integer modulo arithmetic on RISC-based smart cards. In: Proc. 14th Symposium on Computer Architecture and High Performance Computing (2002)

[ISO04] ISO/IEC/JTC 1/SC 31. ISO/IEC 18000 Information technology – Radio frequency identification for item management. Technical report, International Organization for Standardization, Geneva, Switzerland (2004)

[KP06] Kumar, S., Paar, C.: Are standards compliant elliptic curve cryptosystems feasible on RFID? Printed handout of Workshop on RFID Security (RFIDSec) (July 2006)

[LV07] Lee, Y.K., Verbauwhede, I.: A compact architecture for montgomery elliptic curve scalar multiplication processor. In: Kim, S., Yung, M., Lee, H.-W. (eds.) WISA 2007. LNCS, vol. 4867, pp. 115–127. Springer, Heidelberg (2008)

[Mon87] Montgomery, P.L.: Speeding the Pollard and elliptic curve methods of factorization. Mathematics of Computation 48, 243–264 (1987)

[Nat00] National Institute for Standards and Technology. Digital Signature Standard (DSS). Technical report, NIST (January 2000)

[Org07] Organisation for Economic Co-operation and Development. The economic impact of counterfeiting and piracy. Technical report, OECD (2007)

[OScE04] Öztürk, E., Sunar, B., Savaş, E.: Low-power elliptic curve cryptography using scaled modular arithmetic. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 92–106. Springer, Heidelberg (2004)

[SBM+06] Sakiyama, K., Batina, L., Mentens, N., Preneel, B., Verbauwhede, I.: Small-footprint ALU for public-key processors for pervasive security. In: Workshop on RFID Security (2006)

[Wol05] Wolkerstorfer, J.: Is elliptic-curve cryptography suitable to secure RFID tags? In: Workshop on RFID and Lightweight Crypto (2005)