A $5.1\mu J$ per Point-Multiplication Elliptic Curve Cryptographic Processor

—authors’ version—

Vladimir Rožić, Oscar Reparaz and Ingrid Verbauwhede

ESAT/COSIC and iMinds, KU Leuven, Kasteelpark Arenberg 10, B-3001
Leuven-Heverlee, Belgium

Abstract. Security features such as privacy and device authentication are required in wireless sensor networks, electronic IDs, RFID tags and many other applications. These features are provided using cryptography. Symmetric key cryptography, where the key is distributed between the communication parties prior to communication, doesn’t provide adequate solution for large scalable systems such as sensor networks. In these cases, public-key cryptography (PKC) should be used. However, public-key algorithms are typically more computationally intensive than their symmetric key counterparts, which creates difficulties in meeting the strict area, power and energy requirements. Elliptic curve cryptography (ECC), due to relatively small operand sizes, can be used to answer the imposed challenges. In this paper we present a processor for ECC over $GF(2^{163})$. This processor can perform elliptic curve (EC) point multiplication as well as general modular operations. The processor is flexible enough to support multiple cryptographic protocols. The chip is fabricated using UMC $.13\mu m$ 1P8M process, resulting in a core area of 0.54 mm^2 . The energy consumption to perform one EC point multiplication is $5.1\mu J$. The design features lightweight countermeasures against side-channel attacks. A security evaluation shows the effectiveness of such countermeasures.

1 Introduction

Strong cryptography is required for many applications on resource-constrained, low-power and low-energy devices, ranging from smart meters, eIDs, Radio Frequency Identification (RFID) tags to pacemakers and body area networks (BAN). Protocols that provide both security and privacy often rely on computationally intensive public-key operations. The design of compact and efficient hardware that performs public-key operations under strict area, power and energy constraints is very challenging. On top, cryptographic implementations on embedded devices are subject to physical attacks such as side-channel attacks (SCA) and fault attacks. Instead of attacking the cryptographic algorithms, these attacks extract the secret key by observing data-dependent execution time, power dissipation or EM radiation of the circuit during normal operation of

the cryptographic function. Countermeasures against these attacks require additional area and/or execution time and thus should be taken into account from the very early design stage to meet the area and energy budget.

From the economic point of view, compact hardware implementations are important to reduce the production costs and the final price of the product. Low power consumption is an essential requirement in wireless sensor networks which rely on the external electromagnetic field for the power supply (RFID tags) or have limited battery capacity (body area networks). Due to high computational complexity of public key algorithms, it is a challenging task to provide a hardware implementation of a PKC which meets the area, power and energy constraints. Elliptic curve cryptography (ECC) is the most suitable public key scheme for constrained devices due to its short operand size and lower computational complexity compared to RSA. Besides, ECC allows for efficient countermeasures against side-channel attacks, which are highly relevant in the embedded setting. 163-bit ECC achieves equivalent security level as 1024-bit RSA which is equivalent to 80-bit symmetric key algorithm.

1.1 Previous work

The ECC processor presented in [1] uses a 16x16 bit multiplier and a 1 Kb RAM to achieve small area. However, this approach resulted in a large execution time of 251 clock cycles for a field multiplication, and 296K clock cycles per PM. The design presented in [2] uses a latch based memory unit and a digit-serial data path to achieve lower execution time. Neither of these two implementations can support the basic Schnorr protocol [3] as they lack instructions for modular arithmetic operations.

1.2 Our contribution

In this paper we present a processor for elliptic curve cryptography. The processor supports instructions for modular arithmetic operations and for elliptic curve point multiplication. Both operations are used in common cryptographic protocols. The modular arithmetic operations are executed on a compact 8-bit data-path in a byte serial manner. The computational effort is reduced by using a redundant 168-bit form to represent 163-bit data. Point multiplication is performed on a separate module which is equipped with six 163-bit registers and a dedicated data path for modular operations on 163-bit operands. By introducing a common-Z coordinate system and register reuse technique, the amount of storage needed for the intermediate results is reduced, which results in a compact design. Our design also features lightweight countermeasures against side-channel attacks. The coordinate system used allows for randomization of the intermediate values. We taped-out the chip and performed a practical side-channel evaluation.

1.3 Paper organization

This paper is organized as follows. The basics of ECC and some generally used authentication protocols are given in Section 2. This chapter also contains the description of the algorithm used for point multiplication. Section 3 presents the general architecture of the security processor. Details on the implementation of the ECP are given in Section 4. In Section 5, an overview of the micro controller implementation and instruction set are presented, and it is demonstrated that most common authentication protocols can be implemented using the proposed instruction set. A summary of the implemented lightweight countermeasures to side-channel attacks is given in section 6. Chip features and measurement results are presented in Section 7 while results of the side-channel security evaluation are given in Section 8. Finally, the conclusion is formed in Section 9.

2 Background

2.1 Operations on elliptic curves

Elliptic curves are defined over a finite field. An elliptic curve is a set of all points (x, y) that satisfy the algebraic relation of the form (1) where a and b are curve parameters.

$$y^2 = x^3 + ax + b \quad (1)$$

Addition can be defined on the points on the elliptic curve in such way that all points on the curve form a group with respect to this operation.

Fig. 1. Hierarchy of ECC operations. If projective coordinates are used, finite field inversion is not needed.

Fig. 2. The ID-transfer scheme of Lee *et al.* [4].

Figure 1 shows the hierarchy of ECC operations. The main operation in ECC is multiplication of a point on the curve with a scalar. The security of

ECC-based protocols relies on the difficulty of calculating the inverse of this operation. In other words, if the point P and the product kP are known, it should be difficult for the attacker to calculate the value of the scalar k . Point multiplication can be performed using two basic point operations: point addition and point doubling. These two operations can be computed using arithmetic operations on the finite field: addition, multiplication and inversion. One choice is to represent the points on an elliptic curve using projective coordinates, in which case finite field inversion is not needed to carry out point group operations.

2.2 RFID authentication protocols

The processor presented in this paper supports elliptic-curve point multiplication as the principal cryptographic operation. However, our processor is not limited to performing only point multiplication, but also general modular multiplication and addition. This means that our processor can perform several different ECC-based RFID authentication protocols [3,5,6,4]. A description of the exemplary EC-RAC protocol [4] is shown in Figure 2. The main operations that an RFID tag should perform when executing the EC-RAC protocol are: random number generator, elliptic curve point multiplication, modular multiplication and modular addition. The most complex operation is elliptic curve point multiplication. We discuss the high-level procedure for point multiplication in next subsection. For a detailed description of the EC-RAC protocol, we refer to [4].

2.3 Algorithm for point multiplication: area and power optimizations

In this design we use the elliptic curve K-163 recommended by the National Institute of Standards and Technology (NIST). This curve achieves a balance between security, area and speed. One point multiplication on this curve takes about 1 800 multiplications in $GF(2^{163})$.

We make two design choices to improve efficiency. The first one is to employ the Montgomery ladder exponentiation algorithm. This choice allows to calculate a point multiplication without storing the y coordinate. This reduces the required number of registers and allows for a compact implementation.

The second choice is to use projective coordinates to represent curve points. This reduces the number of cycles needed for a point multiplication, at the cost of storing 3 coordinates per point instead of 2. The execution time is lowered since point addition and doubling can be performed without costly finite field inversions.

The number of registers can be further reduced by using the common Z projective coordinate system (X_1, X_2, Z) where the Z coordinate of both points used in the Montgomery ladder have the same value in every loop run. Equations

Algorithm 1 Montgomery ladder with the López-Dahab's algorithm.

Require: An EC $y^2 + xy = x^3 + ax + b$, a point $P = (x, y)$, a t -bit integer k , $k = (1, k_{t-2}, \dots, k_0)$, $k_i \in \{0, 1\}$

Ensure: $Q = kP$

```

if  $k = 0$  or  $x = 0$  then
    output  $(0, 0)$ 
end if
 $X_1 \leftarrow x, Z_1 \leftarrow 1, X_2 \leftarrow x^4 + b, Z_2 \leftarrow x^2$ 
for  $i = t - 2$  downto 0 do
    if  $k = 1$  then
         $(X_1, Z_1) \leftarrow \text{Madd}(X_1, Z_1, X_2, Z_2)$ 
         $(X_2, Z_2) \leftarrow \text{Mdouble}(X_2, Z_2)$ 
    else
         $(X_1, Z_1) \leftarrow \text{Mdouble}(X_1, Z_1)$ 
         $(X_2, Z_2) \leftarrow \text{Madd}(X_1, Z_1, X_2, Z_2)$ 
    end if
end for
 $Q \leftarrow \text{Mxy}(X_1, Z_1, X_2, Z_2)$ 

```

for the addition and doubling in the López-Dahab algorithm using common Z projective coordinates are the following:

$$\begin{aligned}
 Z_{Add} &= (X_1 \cdot Z_2 + X_2 \cdot Z_1)^2 \\
 &= (X_1 + X_2)^2 \cdot Z^2, \\
 X_{Add} &= x \cdot Z_{Add} + (X_1 \cdot Z_2) \cdot (X_2 \cdot Z_1) \\
 &= x \cdot Z_{Add} + (X_1 \cdot X_2 \cdot Z^2) \\
 Z_{Double} &= (X_2 \cdot Z_2)^2 \\
 X_{Double} &= (X_2^2 + c \cdot Z_2^2)^2
 \end{aligned} \tag{2}$$

At the end of every loop iteration, the results of point addition and doubling have different Z coordinate values. Additional steps shown in (3) are needed to transform the results back into a form with common Z coordinate. These steps are executed at the end of each iteration.

$$\begin{aligned}
 X_1 &= X_{Add} \cdot Z_{Double} \\
 X_2 &= X_{Double} \cdot Z_{Add} \\
 Z &= Z_{Add} \cdot Z_{Double}
 \end{aligned} \tag{3}$$

To sum up, by using Montgomery ladder algorithm with projective coordinates and common Z coordinate, the number of variables needed to store two elliptic curve points is reduced to 3. Two temporary variables are needed, as well as the accumulator used for multiplication, which results in 6 registers in total.

Fig. 3. Top-level system architecture.

Fig. 4. Elliptic-curve co-processor architecture.

3 System architecture

A high-level overview of the system architecture is shown in Figure 3. The main components of the processor are: an elliptic-curve co-processor (ECP), a crypto micro-controller, a bus manager and a set of RAM and ROM memories.

Elliptic-curve co-processor (ECP). The elliptic-curve co-processor (ECP) is used for performing elliptic-curve point multiplication. This co-processor reads a base-point and a scalar value from RAM, performs multiplication and writes the data back to RAM. The ECP is described in detail in Section 4.

Crypto micro-controller. The micro-controller is used to run the authentication protocol. The micro-controller interfaces with the ECP in a master-slave fashion and triggers it to perform point multiplication. This 8-bit micro-controller is used also to perform modular arithmetic operations (such as field addition and multiplication). It reads instructions from a ROM memory and can access the RAM memory. The micro-controller is described in detail in Section 5.

Other components. The ROM memory holds the authentication protocol program, system parameters and the private key, following the architecture proposed in [7]. The RAM is used as scratch memory. The required memory space depends on the implemented protocol. The Schnorr protocol requires 109 bytes of ROM (25B for program and 84B for data) and 107 bytes of RAM. The bus manager orchestrates the memory access of the micro-controller and the ECP as well as the communication with the front end module.

Our prototype chip implements the elliptic-curve co-processor (ECP), the dedicated crypto micro-controller, and the bus manager. We also implemented an on-die test module that enables separate testing of each component of the design. This module consists of multiplexers which provide direct connections between pins of the chip and inputs of different modules, following the strategy of [8]. Our prototype chip is connected to an FPGA that holds the memory

blocks (RAM and ROM) as well as the random number generator. For a final device one should incorporate all the components shown in the figure.

4 Elliptic curve processor: Design Decisions

In this section we describe our processor architecture for ECC over $GF(2^{163})$. The ECP reads an elliptic curve point and a scalar value from memory, performs the scalar multiplication and writes the resulting point back to the memory.

4.1 Control logic

The ECP control logic is organized using a two-level finite state machine (FSM) as shown in Figure 4. Control2 is the higher-level FSM which controls point multiplication using Montgomery ladder with the López-Dahab algorithm [9]. Control1 is the lower-level FSM which performs point addition and doubling using the finite field operations. This FSM receives the data and control signals from Control2 and generates control signals for register management and modular arithmetic logic unit (MALU). The control logic was implemented using standard cells.

4.2 ECP Datapath and registers

In this section we describe our register file and data-path architecture for performing modular operations in a $GF(2^{163})$ field using the polynomial base representation with irreducible polynomial $r(x) = x^{163} + x^7 + x^6 + x^3 + 1$ which is specified by NIST. The presented ECP module contains six 163-bit registers and a 163-bit data path.

Register arrangement. In our architecture, following Lee et al. in [7], we use a careful register arrangement to minimize the impact in area. We save multiplexers and gate area at the price of a limited access to registers.

Registers are arranged in a circular structure such that only *RegA* can be accessed from the input port while other registers can only be updated from the previous register in the chain. The input data is written and read 8 bits at the time. The input is connected to 8 LSBs and the output to 8 MSBs of the *RegA*. Registers *RegA*, *RegB* and *RegC* are connected to the ALU inputs, while the ALU output and intermediate results are always stored in *RegA*.

4.3 Arithmetic Logic Unit

An Arithmetic Logic Unit (ALU) was implemented for performing the arithmetic field operations. During the design of the ALU, we paid special attention to achieving a low area and low energy consumption. We base our design on the work of Sakiyama et al. [10]. Sakiyama et al. proposed an architecture for a scalable and flexible modular arithmetic logic unit (MALU) which enables trade-off between throughput and area.

Table 1. Area and power estimations of EC Processor.

Digit Size	ECP Gate Area	Frequency (KHz)	Power (μW)	Energy (μJ)
1	10106	1130	36.6289	9.16
2	11383	590	21.5480	5.39
3	12236	411	15.7463	3.94
4	12863	323	12.0758	3.02
5	13497	266	11.4063	2.85

Digit size, low area and low energy. A trade-off between throughput and area is made by modifying the digit size. Using a larger digit size enables faster multiplication and increases throughput, but it also increases design area since additional logic cells have to be implemented.

However, low area is not our only requirement. For applications in RFID systems and sensor networks low energy dissipation is the crucial requirement. Table 1 shows the total energy consumed during the execution of the Schnor protocol as a function of digit size. Values in the table are obtained based on the synthesis results and account only for the power consumed by the standard cells.¹

It can be seen that by choosing a larger digit size, the power consumption can be reduced at the price of larger gate area. However, when the digit size is increased from 4 to 5, power consumption drops by less than 10%. In the proposed design, the value of the digit size was chosen to be $d = 4$ to satisfy all the requirements.

Rotation. In order to perform digit-serial multiplication, a circular shift operation of *RegB* data by d bits was implemented, where d is the digit size. Rotation is used instead of shifting because, at the end of the multiplication, data in *RegB* has to be preserved to be used at the later stages of the algorithm. Since the register size 163 is not divisible by the digit size $d = 4$, the datapath has to be implemented in such way that multiplication with both 3-bit and 4-bit digits is supported. The direct solution is shown in Figure 5 a).

Multiplication. Multiplication of values stored in registers *RegB* and *RegC* is performed in a digit-serial manner. *RegA* is used as scratch register. The four most significant bits of *RegB* are connected to the four computational units. By setting appropriate multiplexer control signals, in the first cycle of multiplication 3 computational units are used. In the remaining 40 cycles, all 4 units are used. This solution requires a delay element (flip-flop) in the data path as well as the rotation of data in *RegB* by both 3 and 4 bits. This architecture is improved by extending the *RegB* by one bit as shown in Figure 5 b). This way only a

¹ Area and power associated with the interconnect are not included in this analysis. Frequency values are chosen in such way that the time needed to execute the complete protocol is less than 250ms.

Fig. 5. Two possible architectures of the data path. By moving the flip-flop from the data-path to the register B, multiplexer area and interconnect complexity are reduced.

rotation by 4 places has to be provided which results in lower multiplexer area as well as the simplified interconnect.

4.4 Micro architecture

A full custom layout was used for designing the register file and the 163-bit datapath for finite field operations. The motivation is given below.

Interconnect parasitics. The impact of interconnect parasitics is increasing with the scaling of the integrated circuit technology. Interconnect delay and power do not scale well with the feature size, which leads to a growing ratio of interconnect area and power. A way to cope with these effects is by localizing communication between the logic units to minimize the interconnect complexity.

Partitioning. The presented ECP module contains six large registers and a 163-bit data path. This part of the design is organized into bit slices consisting of exactly one bit of each register, local multiplexers and a segment of the data path. A serial connection of all bit slices is equivalent to the architecture presented in Figures 6 and 7. Data transfer between registers as well as the communication between the registers and the data path is performed locally inside bit slices, while data loading, shifting and communication between different parts of the data path are performed through intra cell connections. The cost of this data transfer is improved through a global placement strategy.

Two types of bit slices are used in this design. Figure 8 shows the type 1 or the regular bit slice consisting of six flip-flops, multiplexers for local connection and a data path segment consisting of four *AND-XOR* branches. Most cells in the ECP are of type 1. Since only two of the registers, namely *RegA* and *RegB*, are

Fig. 6. Architecture of the Register File

Fig. 7. Architecture of the Modular ALU

used for data loading and shifting, the remaining four registers are updated only by internal bit slice operations. For this reason, there are no inputs or outputs connecting directly to flip-flops C, D, E and F. Conversely, flip-flops A and B can be read and updated from the outside. Communication between different parts of the data-path is done through inputs (A_{PRE} , $A1_{PRE}$, $A2_{PRE}$, $A3_{PRE}$) and outputs ($A1$, $A2$, $A3$) which are connected to the neighboring bit slices. Other data path signals as well as the multiplexer control signals are distributed to all bit slices.

Bit slices of type 2 differ from the regular bit slices only in the local data-path in which additional XOR gates are added to perform modular reduction. Type two cells correspond to the bit positions of the coefficients of the used irreducible polynomial $r(x) = x^{163} + x^7 + x^6 + x^3 + 1$, namely positions 7, 6, 3 and 0.

Interconnect. Interconnect complexity is reduced by an efficient placement strategy. The capacitive load of data-path signals which connect adjacent bit slices, is reduced by arranging the cells in a snake-like manner. Figure 9 shows the organization of the design into four blocks as well as the internal organization of each block. Within blocks, cells are arranged into columns of four and data-path signals are routed through the space between the cells. Wires connecting *RegB* bits are routed along the same path. Signal wires used for shifting *RegA*

bytes are needed to store the data in either case. For a detailed description of the modular addition and multiplication algorithms used we refer to [7].

Interfaces. The micro controller interacts with ROM for data, ROM for program, RAM, random number generator, ECP and the I/O interface. The interface is memory mapped: a 13-bit address space is used where 4 most significant bits are used for device identification. The remaining 9 bits are used only when accessing memory modules. Since all data management operations are based on 21-byte blocks, the adopted addressing scheme uses 4 bits to denote the block address and 5 bits denote the byte within the block.

Instruction set. The instruction set of the micro-controller includes instructions for moving data around, for performing basic modular arithmetic operations and for triggering the ECP point multiplication. The point multiplication operation is not blocking the micro-controller from performing general modular operations in parallel without waiting for ECP to complete the computation. Instructions are composed of 1, 2 or 3 bytes. The first byte denotes the command and the remaining bytes contain block addresses of the operands.

The provided instruction set is expressive enough to allow concise implementations of multiple authentication protocols. We implemented the whole Schnorr protocol in just 8 instructions, and the EC-RAC protocol in 15 instructions. Timings for these programs are given next in Section 7.

6 Lightweight Countermeasures to Side-channel Attacks

The design is moderately protected against side-channel attacks, featuring affordable light-weight countermeasures that operate at the algorithmic level.

Timing attacks. To prevent timing attacks, the execution time is fixed down to RTL regardless of the value of the key. This is achieved by the choice of the Montgomery ladder for the scalar multiplication operation. The base-point is also fixed.

Power analysis attacks. Power analysis attacks are an important family of side-channel attacks. Power analysis techniques require to measure the instantaneous power consumption of the chip performing the cryptographic operations to infer cryptographic secrets. The prime attack is Differential Power Analysis (DPA) [11].

Elliptic curve cryptography is especially suited for protection against power attacks at an algorithmic level of abstraction. More specifically, our prototype chip implements the *random projective coordinates* countermeasure. This countermeasure consists of randomizing the representation of intermediate points arising in the computation of the scalar multiplication by randomizing the Z coordinate. We perform an evaluation of this countermeasure in Section 8.

Table 2. Chip specifications

Technology	UMC .13 μm 1P8M CMOS
Supply Voltage	Core 1V, I/O 3.3V
Core Area	735 $\mu\text{m} \times 735\mu\text{m}$
Operating Frequency	847.5 kHz
Power Consumption	50.4 μW
Throughput	9.8 point multiplications / s

Balanced layout. In addition, at the layout level, the design was carefully balanced to minimize the impact of data-leakage. Other countermeasures at the cell level, such as WDDL [12], are excessively expensive for the energy requirements (stay within the pJ range).

7 Results

Fig. 10. Prototype chip plugged into the test SASEBO board. **Fig. 11.** Chip micrograph of the ECC processor.

A prototype chip for the proposed ECC processor was fabricated using UMC 0.13 μm process. A chip micrograph is presented in Figure 11. As can be seen, the design is pad limited. In Table 2 we write the core area ($735\mu\text{m} \times 735\mu\text{m}$) and power consumption as measured. The chip is placed in a SASEBO board [13] for testing purposes, as can be seen in Figure 10. This board holds as well an FPGA for interfacing and testing. Table 3 shows a comparison of our design with previous work. We remark that the design presented in this paper offers the lowest energy consumption, and is the only one that supports the complete Schorr protocol.

Power and energy consumption. At the operating frequency of $f_1 = 847.5$ kHz and core voltage $V_{\text{dd}} = 1\text{V}$, the processor consumes 50.4 μW and uses only 5.1 μJ

Table 3. Comparison with the State of the Art

	This work	[1]	[2]
Technology	.13 μ m	.18 μ m	.22 μ m
Short-term storage	FF registers 6x164 bits	SRAM 64x16 bits	Latch-based memory 5x163 bits and FF registers 2x163bits
Clock cycles per field multiplication	41	251	41
Clock cycles per PM	86 224	296 299	80.5k
PM computation time @ 847.5 KHz	102ms	350ms	95ms
Power @ 847.5 kHz	50.4 μ W	67.23 μ W	96.4 μ W
Energy per PM	5.1 μ J	23.5 μ J	9.16 μ J
Obtained by	measurement	measurement	simulation
Curve	NIST K-163	163-bit Binary	NIST B-163
Coordinates (L-D = López-Dahab)	Common-Z L-D		L-D
Return Affine coordinate (Support for inversion)	Yes	Yes	No
Support for integer operations	Yes	No	No
Support for Schnorr protocol	Yes	No	No

for one point multiplication. At this frequency, the throughput is 9.8 point multiplications per second. The average power consumption can be lowered to 13.6 μ W by reducing the clock frequency to $f_2 = 211.9$ kHz. One of the applications of our ECC processor is in passive RFID tags. Passive RFID tags draw energy for operation from the magnetic field. Increasing the distance from the reader causes exponential drop of power density. According to Masui et al [14], for a passive tag (with a typical 54 by 86 mm antenna coil) the tag LSI has a power budget lower than 300 μ W to support a 50 cm range. For a 70 cm communication range, the power budget of the tag drops to 40 μ W. Assuming that a tag without crypto-core needs less than 10 μ W (e.g. using the design from [14]), our ECC processor can support a communication range of 60 cm when running at 847.5 kHz. When running at 211.9 kHz and $V_{dd} = 1$ V, our ECC processor consumes only 13.6 μ W, which is low enough to enable a communication range of 70 cm. (Note that in RFID applications, the clock signal is obtained by dividing the standard communication frequency of $f = 13.56$ MHz. The previous frequencies f_1 and f_2 were selected so that $f_1 = f/16$ and $f_2 = f/64$.)

Execution time. The Schnorr protocol consists of one point multiplication (86k cycles), one integer addition (574 cycles) and one integer multiplication (25k cycles). This means that the Schnorr protocol executes in 135 ms at the operating frequency of 847.5 kHz. The EC-RAC protocol performs 3 point multiplications and takes 286 876 cycles (or 339 ms) to execute.

8 Security Analysis

In this section, we report on the prototype IC’s resistance against power analysis attacks. Two widely used techniques were applied: Differential Power Analysis (DPA) and Simple Power Analysis (SPA) [11].

8.1 Measurement setup and measurements

For the purpose of the security evaluation, the prototype chip was placed in a SASEBO-R board [13], which contains a control FPGA that interacts with the chip. The power consumption was measured as the voltage drop across a $1\ \Omega$ shunt resistor inserted in the core GND path using an Agilent DSO6104 oscilloscope with 1 GHz bandwidth, 8-bit resolution and a sampling rate of 2 GS/s. The clock signal of 12.5 MHz was provided by a waveform generator. The amplitude of the clock signal was carefully adjusted to the bare minimum that allowed the chip to work. Reducing the clock amplitude reduced the amount of clock signal coupled to the GND path which in turn substantially improved the quality of the measurements.

The following analyses were executed in a white-box scenario, that is, as evaluators we had access to the clock and trigger signals (allowing an accurate alignment of the power consumption measurements) and to the full RTL description of the chip (therefore we knew exactly what the chip computes at every clock cycle). If this is not the case in a real attack, the attacker has first to guess and test the necessary information by trial and error, slowing down the attack process.

8.2 DPA Methodology

DPA is a statistical technique used to recover the key of a cryptographic chip from its instantaneous power consumption, provided that the power consumption is related to the intermediate data processed. Informally, DPA recovers the key in a divide-and-conquer fashion by comparing the measured power consumption with several hypothesized power consumptions, one for each sub-key hypothesis. It is generally expected that the similarity between the measured power consumption and the predicted power consumption will be high only for the correct key hypothesis. More precisely, the attack consists of two steps: (a) prediction of the instantaneous power consumption at one specific clock cycle, for several key hypotheses and (b) correlation of the predictions against the measurements to distinguish the correct hypothesis.

For (a) the attacker normally has to build a model describing the dependency between data processed and power consumption. In our evaluation, we assume that the instantaneous power consumption is proportional to the number of bits flipped in all registers (the so-called Hamming distance model [15].) This assumption neglects dependency between data and static leakage (which may no longer be negligible in smaller feature sizes). Nevertheless, the approximation turned out to be valid for the purpose of the security evaluation. Register-transfer level

Fig. 12. DPA attack. Top: mean trace. Center: correlation trace for the correct subkey hypothesis. Bottom: correlation trace for the incorrect subkey hypothesis.

simulations were executed to generate the predictions of the instantaneous power consumption using the Hamming distance model at clock cycle 1458 from the beginning of the ECP operation. At this specific clock cycle only *RegA* updates its value, regardless of the value of the processed key bit. Up to this point, only one key bit has been processed, and thus the number of possible sub-keys is two.

For (b), a metric to quantify similarity between the predicted power consumption and the actual power consumptions needs to be defined. We chose the widely-employed Pearson’s linear correlation coefficient [16].

8.3 DPA Measurements and Results

For the purpose of this first evaluation, all countermeasures were switched off by fixing the random number generator output to a specific byte known to the evaluator. A total of 1500 power consumption traces corresponding to the processing of the first two key bits of the scalar multiplication of known, varying base points $P_1, P_2, \dots, P_{1500}$ with a fixed key were recorded. The mean trace is depicted in Figure 12, top. The first key bit is processed from time samples 0.9×10^5 to 1.7×10^5 , and the second one from 1.7×10^5 to 2.5×10^5 . For the analysis in this section, we focus on the processing of the first key bit and we aim at recovering it. Other bits can be recovered in the same way. Figure 12,

Fig. 13. Maximum correlation coefficient over one the processing of one key bit, as a function of the number of traces available for the attack. The correct subkey is in blue. The incorrect subkey is in green.

Fig. 14. Effect of the swapping on the power consumption. Green: no switch, red: switch. Same base point, no z -coordinate randomization

center (bottom), shows the correlation trace for the correct (incorrect) key hypothesis, respectively, when 1500 traces are used. The correlation for the correct key hypothesis achieves a maximum around time sample 1.5×10^5 of $\rho \approx 0.29$ over a noise floor of $\rho \approx 0.1$, effectively distinguishing between the correct and incorrect subkey hypothesis.

In order to assess the minimum required amount of traces that are necessary for the attack to succeed, the attack was repeated with a different number of traces. Figure 13 shows the results of the attacks. When more than ≈ 350 power consumption traces are available to the attacker, the correct key hypothesis is distinguishable.

8.4 DPA with countermeasures enabled

The results from the previous section show that when the countermeasures are disabled the chip is insecure against DPA attacks if one knows the implementation. Our chip, however, features some lightweight countermeasures against power analysis attacks. Mainly, our design can randomize the internal projective point representation by randomizing the Z coordinate. As a consequence, DPA attacks cannot be mounted because the intermediate values can no longer be predicted. (Randomness is supposed to be generated on-chip and be unknown to the attacker.)

To evaluate the effectiveness of the countermeasure, a total of 30 000 measurements were taken with the randomization enabled. We supplied the random numbers to the chip, thus, they were known to us. Using this fact, we could predict the intermediate values and perform a DPA attack in the same lines as the previous section. The correlation coefficient dropped slightly to $\rho \approx 0.1$. When the random numbers were assumed to be unknown (a scenario more likely to match real world operation) no significant correlation for the correct sub-key hypothesis was found. This gives some evidence that a DPA attack would not succeed when the random projective coordinate countermeasure is activated.

8.5 SPA Methodology

SPA is a family of attacks that exploit some specific characteristic in the power consumption trace of the device. When successful, SPA attacks typically require few traces (even a single one) to recover the full key from the device. However, contrary to what the name suggests, SPA attacks are usually very specific to the device, and therefore harder to mount.

An important class of SPA attacks are timing attacks. These attacks exploit the timing variance with different inputs. Our chip implements the Montgomery powering ladder to perform a scalar multiplication, which runs in constant time independently of the data and key (In a prototype version, if the first n bits of the scalar are 0, n operations will be skipped. Hence, an attacker can learn n bits of the scalar with probability 2^{-n} . This can pose a problem for ECDSA signatures: an adversary could first learn a few bits of each ephemeral key and then mount a lattice-based attack to recover the long-term key. This is avoided

by not skipping operations and thus forcing constant-time execution.) Hence, it is intrinsically resistant to timing attacks.

However, in our evaluation a source of SPA leakage was identified in our prototype chip. When the chip is performing a scalar multiplication, the only key-dependent operation is a swapping of the contents of *RegA* and *RegB* (see Algorithm 1). More precisely, if the key bit being processed equals the following key bit, the swapping occurs, else the swapping is omitted. (This step is executed in constant time.) Thus, the power consumption will be slightly higher when the swapping occurs than when it does not. This difference is perceptible in the measured power traces. Figure 14 shows a superposition of 100 traces where no swapping occurs (green) and 100 traces where the swapping occurs (red). At time samples around 6885 it is possible to distinguish whether the swapping occurs or not, leading to knowledge if two consecutive bits are equal or not.

This observation can be turned into a key-recovery attack that requires one trace and succeeds both when the projective coordinate randomization is enabled and disabled. The base point is not required to be known. The attack can be reproduced as follows:

1. Take a reference pattern p from the portion t_r of the power trace corresponding to the processing of bit r of the key. The pattern p is several cycles long and contains the cycle where the swap does/does not occur.
2. For each portion of the trace where the key bit i is being processed, $i \neq r$, correlate the reference pattern p against the portion t_i that contains the cycle where the swap does/does not occur. If similarity between t_i and p is found, then $k_r \oplus k_{r+1} = k_i \oplus k_{i+1}$.

Note that precise information about the device is required to launch this attack. Namely, one needs to know when the switching operation occurs. The attack outputs two key candidates k and k^* since all bit xor differences are known (but not the actual bit values). They can be easily tested against a (P, kP) pair to select the actual key if such a pair is available.

8.6 SPA Measurements and Results

The attack described in the previous section was performed on the prototype chip. For this section we focus on recovering all key bits from a single power trace. Figure 15, top, shows one single trace of the chip performing a scalar multiplication with a key of 32 effective bits. Superposed in red is the chosen pattern p . In the center there are zoomed version. Bottom shows the similarity between the pattern and the whole trace at the relevant time samples, showing the process of key-recovery. A horizontal ad-hoc threshold around $\rho \approx 0.994$ shows the partition of the two sets: high and low similarity with respect to the pattern. All the xor differences between consecutive key bits can be recovered in this way. (Note that since we chose $r = 4$, for the fourth bit the similarity is 1, perfect match. As expected, the similarity between the pattern and itself is 1.)

Fig. 15. SPA attack. Top: one single trace with 32 effective key bits. The extracted pattern is superposed in red. The result of the correlation of the pattern against the trace at the relevant time samples is depicted in the bottom plot.

8.7 Discussion

From the results from previous sections, we conclude that the chip offers an adequate security against DPA attacks when the randomization is enabled. The identified SPA leakage is harder to exploit since it requires considerable knowledge of the internals of the chip to find the precise clock cycles where the swapping between registers occurs. The source of the DPA leakage (neutralized by the randomization countermeasure) and SPA leakage are of different nature. They occur at different clock cycles and exploit different phenomena: while the SPA attack exploits the fact that power consumption is proportional to the xor of each pair of consecutive key bits, the DPA attack (with countermeasures disabled) exploits the slight dependence between power consumption and the number of flips in the registers. Future work includes investigation on strategies to minimize the SPA leakage, as it was not expected during design. More complex

attack strategies, such as Refined Power Analysis [17], are out of the scope of this paper.

9 Conclusion

In this paper, we have demonstrated that ECC-based authentication protocols can be carried out on low-cost hardware in less than 500ms. By using instructions for the modular addition and multiplication, our ECC processor can support a wide range of ECC-based protocols such as Schnorr protocol and EC-RAC protocol. Our chip also features lightweight countermeasures against side-channel attacks. A practical side-channel evaluation was carried to show the effectiveness of those countermeasures.

References

1. Hein D, Wolkerstorfer J, Felber N. ECC Is Ready for RFID – A Proof in Silicon. *Selected Areas in Cryptography, Lecture Notes in Computer Science*, vol. 5381, Avanzi R, Keliher L, Sica F (eds.). Springer Berlin / Heidelberg, 2009; 401–413.
2. Block et al H. RFID Products Offering Asymmetric Authentication Based on Elliptic Curve Cryptography. *RFIDsec*. 2008.
3. Schnorr CP. Efficient Identification and Signatures for Smart Cards. *Advances in Cryptology - CRYPTO'89, Lecture Notes in Computer Science*, vol. 435, Brassard G (ed.), Springer-Verlag, 1989; 239–252.
4. Lee Y, Batina L, Verbauwhede I. EC-RAC (ECDLP Based Randomized Access Control): Provably Secure RFID authentication protocol. *IEEE International Conference on RFID*, 2008; 97–104.
5. Bringer J, Chabanne H, Icart T. Cryptanalysis of EC-RAC, a RFID Identification Protocol. *International Conference on Cryptology and Network Security – CANS'08, Lecture Notes in Computer Science*, vol. 5339, Springer-Verlag, 2008; 149–161.
6. Okamoto T. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. *Advances in Cryptology – CRYPTO'92, Lecture Notes in Computer Science*, vol. 740, Brickell E (ed.), Springer-Verlag, 1992; 31–53.
7. Lee YK, Sakiyama K, Batina L, Verbauwhede I. Elliptic-curve-based security processor for RFID. *IEEE Trans. Computers* 2008; **57**(11):1514–1527.
8. Fan J, Knežević M, Karaklajić D, Maes R, Rozić V, Batina L, Verbauwhede I. FPGA-based testing strategy for cryptographic chips: A case study on elliptic curve processor for RFID tags. *On-Line Testing Symposium, 2009. IOLTS 2009. 15th IEEE International*, 2009; 189 –191.
9. López J, Dahab R. Improved Algorithms for Elliptic Curve Arithmetic in $GF(2^n)$. *Selected Areas in Cryptography, Lecture Notes in Computer Science*, vol. 1556, Tavares S, Meijer H (eds.). Springer Berlin / Heidelberg, 1999; 632–632.
10. Sakiyama K, Batina L, Mentens N, Preneel B, Verbauwhede I. Small-footprint ALU for public-key processors for pervasive security. *Workshop on RFID Security – RFIDSec'06, Lecture Notes in Computer Science*, 2006.
11. Kocher PC, Jaffe J, Jun B. Differential power analysis. *Advances in Cryptology – CRYPTO'99, Lecture Notes in Computer Science*, vol. 1666, Wiener MJ (ed.), Santa Barbara, CA, USA, 1999; 388–397.

12. Tiri K, Verbauwhede I. A logic level design methodology for a secure dpa resistant asic or fpga implementation. *Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings*, vol. 1, 2004; 246 – 251 Vol.1.
13. Side-channel attack standard evaluation board
<http://www.rcis.aist.go.jp/special/SASEBO/>.
14. Masui S, Teramoto T. A 13.56 MHz CMOS RF Identification Passive Tag LSI with Ferroelectric Random Access Memory. *IEICE Transactions* 2005; **88-C**(4):601–607.
15. Messerges T, Dabbish E, Sloan R. Examining smart-card security under the threat of power analysis attacks. *Computers, IEEE Transactions on* may 2002; **51**(5):541–552.
16. Brier E, Clavier C, Olivier F. Correlation power analysis with a leakage model. *Cryptographic Hardware and Embedded Systems - CHES 2004, Lecture Notes in Computer Science*, vol. 3156, Joye M, Quisquater JJ (eds.), Springer Berlin / Heidelberg, 2004; 135–152.
17. Goubin L. A refined power-analysis attack on elliptic curve cryptosystems. *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography: Public Key Cryptography, PKC '03*, Springer-Verlag: London, UK, UK, 2003; 199–210.