Analyzing Side-Channel Leakage of RFID-Suitable Lightweight ECC Hardware

Erich Wenger, Thomas Korak, and Mario Kirschbaum

Graz University of Technology
Institute for Applied Information Processing and Communications
Inffeldgasse 16a, 8010 Graz, Austria
{Erich.Wenger,Thomas.Korak,Mario.Kirschbaum}@iaik.tugraz.at

Abstract. Using RFID tags for security critical applications requires the integration of cryptographic primitives, e.g., Elliptic Curve Cryptography (ECC). It is specially important to consider that RFID tags are easily accessible to perform practical side-channel attacks due to their fields of applications. In this paper, we investigate a practical attack scenario on a randomized ECC hardware implementation suitable for RFID tags. This implementation uses a Montgomery Ladder, Randomized Projective Coordinates (RPC), and a digit-serial hardware multiplier. By using different analysis techniques, we are able to recover the secret scalar while using only a single power trace. One attack correlates two consecutive Montgomery ladder rounds, while another attack directly recovers intermediate operands processed within the digit-serial multiplier. All attacks are verified using a simulated ASIC model and an FPGA implementation.

Keywords: Implementation Attack, Correlation Power Analysis, Simple Power Analysis, Digit-Serial Multiplier, Elliptic Curve Cryptography.

1 Introduction

When it comes to RFID security research, many research groups all around the world investigate the viability of new protocols, new optimized algorithms, and new hardware implementation for RFID tags. Those designs have to cope with the restrictive area, power, and runtime challenges that are mandatory for practically usable RFID tags. Additionally, also power-analysis attacks have to be considered. Those attacks can be used to recover keys, even though the actual protocol or algorithm is mathematically secure.

The most promising public-key protocols are based on Elliptic Curve Cryptography (ECC) as ECC offers comparably small memory and practically useable runtime properties. RSA and ElGamal based public key schemes simply need too much memory or only provide inferior runtimes. Some of the most notable ECC implementations are [5, 12, 22, 23, 33, 39]. Unfortunately, there have been too few practical evaluations of those state-of-the-art hardware implementations. Especially the design of Lee et al [23] raised our interest. They use a digit-serial multiplier with a López and Dahab-like [24] Montgomery Ladder. Further we assume that the design-under-attack performs in constant key-independent runtime, utilizes Randomized Projective Coordinates [9] (RPC), and use the private scalar only once (as it is done for ECDSA signatures and Diffie-Hellman key exchanges). Therefore we have very strong assumptions regarding the actual implementation under investigation and many of the related power analysis techniques [7, 10, 17, 20, 21, 26] simply cannot be mounted.

Our contribution. In this paper, we successfully perform simple and correlation-based power analysis attacks on a protected ECC hardware implementation using only a single power trace. The investigated hardware design utilizes a López and Dahab Montgomery ladder, RPC, ephemeral secret keys, a digit-serial binary field multiplier, and performs in constant runtime. We show the practicability of our attacks using simulated and FPGA-measured power traces. The correlation based attack shows how the hamming distance of consecutive key bits can be used to recover the secret scalar. Additionally, we thoroughly analyze the power consumption of bit-serial and digit-serial binary field multiplier. We are able to recover one of the processed operands and discuss how to utilize this information to perform more advanced attacks.

The paper is structured as follows: Section 2 discusses related work. Sections 3–5 elaborate the design under attack, some attack-related prerequisites, and the basic setup for conducting the attacks, respectively. In Section 6 we assure that there is no key-dependent leakage. Section 7 discusses the correlation of consecutive rounds and Section 8 discusses the recovery of intermediate values. Section 9 concludes the paper.

2 Related Work

There exist many papers that describe hardware implementations of ECC. Most of them make use of digit-serial multipliers over $GF(2^m)$, see for example [1, 4, 5, 11, 14, 23, 30]. The reason for that choice lies in several facts. First, binary-field multipliers significantly improve the performance of ECC since they avoid carry-propagation as opposed to prime-field based implementations. Second, since they are based on binary polynomials, they can be efficiently implemented in hardware which makes them especially attractive for embedded systems and low-area designs. Therefore, they are commonly used and applied in real-world applications such as contactless smart cards, RFIDs, or Java Cards [1, 6, 29].

In view of side-channel attacks, there exist many papers that discuss attacks and countermeasures on ECC, for example presented in [8, 9, 13, 18, 27, 31, 32, 36, 37]. Most of the work exploits the weakness of different scalar-multiplication algorithms such as double-and-add which allows to perform SPA attacks in order to distinguish between a single *double* or *add* operation.

Most notable is the work of Walter [36], who attacked RSA using only a single trace. In a sliding-window RSA multiplication, he correlates the preprocessing step with the per-bit multiplication. He notes that such an attack can even work using a single power trace. The attack on the RSA modular exponentiation by Wittemann *et al.* [38] can be performed even in the presence of the message blinding and the multiply always countermeasures. They take advantage of the fact that multiplications followed by squarings share operands in a key-dependent manner. In order to extract the bits of the secret exponent one power measurement recorded during the exponentiation is sufficient. In 2010, Clavier *et al.* [8] introduced the terms vertical and horizontal power analysis. While a vertical power analysis attacks the same time sample on many curves, a horizontal power analysis correlates parts of a single power trace. They performed a horizontal correlation analysis on RSA and similar to Amiel *et al.* [3] distinguished multiplication from squaring operations. Also in 2010, Homma *et al.* [18] generated collisions between squaring operations by recording only two power traces.

While those attacks work on the group structure of RSA and ECC, there exist only a few papers that demonstrate the susceptibility of underlying finite-field arithmetics. In 2006, Akishita et al. [2] demonstrated an attack on ECC by measuring the difference of modular multiplication and squaring. Since they performed attacks targeting ECC-field operations instead of ECC-group operations, their attack is applicable to countermeasures such as unified-addition formulae or SPA-resistant algorithms like the Montgomery-powering ladder. Recently, Pan et al. [34] presented a correlation power-analysis (CPA) attack on a digit-serial multiplier. They targeted the output register of the multiplier and successfully extracted intermediate values from an FPGA implementation. However, since they applied a CPA attack using 1,000 traces, their attack cannot be applied on ECC implementations that use random scalars.

In the following, we present a power-analysis attack that extracts the secret scalar from an ECC implementation consisting of a Montgomery ladder and RPC using only one single power trace. The attack can therefore be even applied to reveal ephemeral keys such as used in the Elliptic Curve Digital Signature Algorithm (ECDSA) or in Elliptic Curve Diffie Hellman (ECDH) protocols.

3 Design Under Attack

In the following, the ECC implementation used for the conducted experiments is presented. The objective of the implementation is to provide resistance against side-channel attacks as well as being flexible in size and runtime. The SCA resistance is achieved by using a constant-runtime Montgomery ladder with randomized projective coordinates and the flexibility in size and runtime is achieved by using a digit-serial multiplier.

3.1 Chosen Top-Level Algorithms

Under the consideration and investigation of related work on timing, power-analysis, and fault attacks applicable on elliptic curve cryptography, we decided to use a left-to-right Montgomery ladder for EC point multiplications $Q \leftarrow k \times P$. The key-independent structure of the Montgomery ladder provides a good foundation against many side-channel attacks. In order to be independent of the most significant bits of the scalar k, the order of the elliptic curve is added to k. Additionally by randomizing the projective coordinates of the base-point, most attacks become infeasible. So a combination of a constant-runtime Montgomery ladder with randomized projective coordinates provides strong resistance against most side-channel attacks[1].

3.2 Hardware Design

Similar to Lee *et al.* [23], our hardware is specially optimized for binary extension fields using the NIST [28] standardized elliptic curve B-163. The hardware design comes with a two-port 163-bit memory, a 163-bit adder, and an MSB-first digit-serial multiplier. In order to save chip area, no dedicated hardware squaring unit is used. For point multiplication we use the formulas by López and Dahab [24] (see Appendix A).

3.3 Digit-Serial Multiplier

Most critical for the size of the hardware design, the runtime of the design and the following attacks is the multiplier used in the design. In the following we give a brief introduction to digit-serial multipliers, which are used in our design.

The term "digit serial" indicates that only a limited number of digits d of operand OpB are processed in each clock cycle. In the special case of $d = 1$, the multiplication approach is also known as bit-serial multiplication approach. A useful property of the digit-serial multiplication approach is that it can be sped up by increasing the digit size d. Thus, the designer can easily change the design to meet the desired area and runtime constraints.

Digit-serial multipliers are used to calculate the product $C \leftarrow OpA \times OpB$, where N bits are required to represent OpA, OpB and C. In the remainder of this paper, we use the notation OpB_i to index a single digit with index i, with $0 \leq i < \lceil N/d \rceil$ and $\lceil N/d \rceil - 1$ indexing the most significant digit. Figure 1 shows two approaches for bit-serial multiplications ($d = 1$) using a fixed reduction polynomial (cf. [15]). Either the most significant bit (MSB) is processed first (shown on the left) or the least significant bit (LSB) is processed first (shown on the right). Since for the LSB-first multiplier, two registers (a_i and c_i) are modified in each cycle, we concentrate on the MSB-first multiplier. The single active (working) register C (respectively c_i) is shifted by d digits to the left and is implicitly reduced using a fixed reduction polynomial. Additionally, OpA (resp. a_i) is multiplied with OpB_i using a simple AND gate. The product of the multiplication is then added to the (by d bits) shifted and reduced work register. After $\lceil N/d \rceil$ cycles, the product ($C \leftarrow OpA \times OpB$) of the finite-field multiplication is stored in register C.

[1] Note that we are aware of fault attacks, but those type of attacks are not subject of this paper.

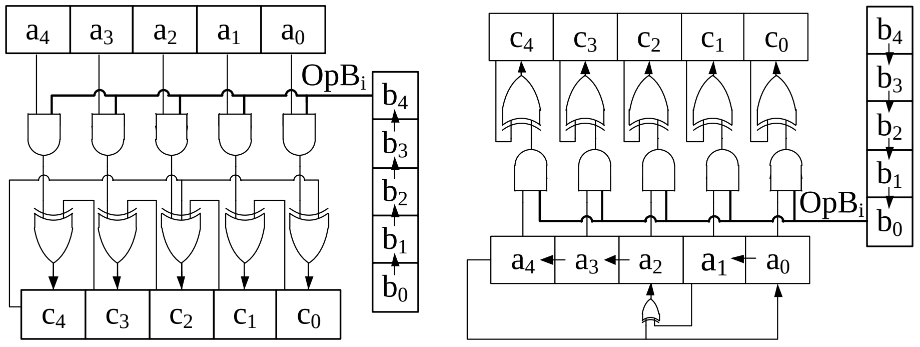

Fig. 1. Finite-field multiplier with fixed reduction polynomial $f(z) = z^5 + z^2 + 1$.

This multiplication approach can be applied to binary-extension fields as well as prime fields. For prime fields, the XOR-gates have to be replaced with full adders. Thus digit-serial multipliers can be used for RSA as well as Elliptic Curve Cryptography.

4 Attack Prerequisites

In order to perform the attacks presented in the following sections, the design has to fulfill some prerequisites which are discussed here.

4.1 Constant Runtime Scalar Multiplication

After recording one power trace while the device under attack performs the scalar multiplication a post-processing step is necessary. The trace T is split in $|k|$ traces after removing the initialization (t_{init}) and final y-recovery phases. Here we take advantage of the fact that all round transformations have an equal runtime. For simplification purposes we assume that $|k| = N$. Sub-traces $R_i = T(t_{init}+t_{round}(N-1-i) \dots t_{init}+t_{round}(N-i))$ represent a López-Dahab double-and-add round operation of the secret scalar k_i, with $i = [0, N-1]$. According to our notation k_{N-1} represents the MSB and k_0 the LSB of k. One method to identify the length of one round (t_{round}) is to perform a cross correlation on the trace T. The result of the cross correlation shows significant, equidistant peaks. The distance corresponds to t_{round}. In the following $R_i(o, o+w)$ denotes a part of the trace of round i with an offset o from the beginning and a length of w samples.

Figure 2 shows a comparison of the simulated and measured power consumption with $d = 1$. For the figures the traces $R_0 \dots R_{N-1}$ are plotted overlayed. Apparently both traces show identical characteristics. The left half of the traces (cycles 995-1150) shows a multiplication of pseudo-random OpA with pseudo-random OpB. In the right half (cycles 1150-1320) OpB has been kept unchanged during the N round transformations. In fact a multiplication with the constant

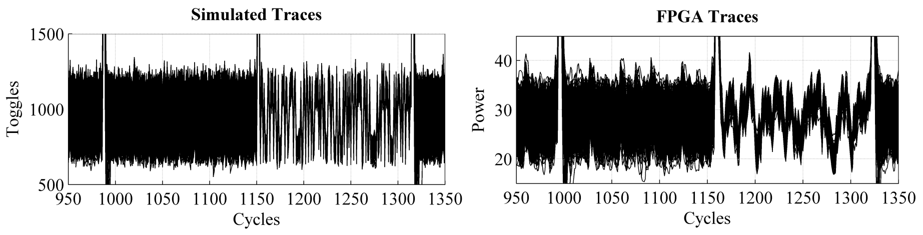

Fig. 2. Comparison of simulated traces (on the left) and traces recorded on the FPGA (on the right).

$c = b^{2^{m-1}} \mod f(z)$ which is used within the López-Dahab formula [24] is performed in this interval. This unchanged operand leads to a similar power consumption in this interval for all round transformations.

4.2 Leakage of the Digit-Serial Multiplier

With the field multiplication being the most time-consuming part of an EC point multiplication and the digit-serial multiplier being the most active part of the circuit, the side-channel vulnerability is highly dependent on the power-consumption of a digit-serial multiplier.

The main source of leakage within the digit-serial multiplier lies within the multiplication of the full word OpA with a single digit OpB_i. In a first model (which was wrong) we theorized that there will be a higher power consumption when $OpB_i = 1$ (with $d = 1$) and a lower power consumption with $OpB_i = 0$. Such a power model may be true for a logic style such as Transistor-Transistor-Logic (TTL), but for a CMOS process our power model had to be refined. In CMOS, the power consumption does not depend on the current state of a signal, but on the transition from the previous state to the next state. So the power consumption is related to the Hamming distance of two consecutive values of OpB_i. Let $w(\cdot)$ denote the Hamming weight and $hd(\cdot,\cdot)$ denote the Hamming distance. Experiments showed that the higher the Hamming distance $hd(OpB_i, OpB_{i+1})$, the higher is the power consumption of the circuit. Let $hd(OpB)$ be a vector with all $hd(OpB_i)$ and $hd(OpB_i)$ be a short form for $hd(OpB_i, OpB_{i+1})$.

Before we discuss the impact of the digit-serial multiplier on the side-channel leakage in Section 8 in detail, we must elaborate our basic side-channel analysis approach and related assumptions.

5 Setup for Conducting the Attacks

Two approaches to record the required power trace for the performed attacks are used. On the one hand the power traces are generated using a simulation of the implementation. On the other hand the design was implemented on an FPGA and the power consumption was measured using an oscilloscope. The achieved results were compared and prove the correctness of the attack assumptions.

5.1 Simulator Toolchain

A simulator toolchain was used in order to enable an attack on the ECC implementation in a noise-free environment. This toolchain consists of four elements: Cadence RTL Compiler v08.10 was used to synthesize the VHDL code to UMC L-130 logic gates; Cadence First Encounter v08.10 was used to place and route the design; NCSim v08.20 was used to simulate the routed design and generate a value-change-dump (VCD) file; and a VCD analyzer was used to generate simulated power traces from the VCD file by applying a toggle-counting technique. By accumulating all toggles within a clock cycle, the resulting trace contained one data value per clock cycle. As shown in Kirschbaum *et al.* [19], simulated power traces derived from toggle counts are well comparable to SPICE power simulations as well as to real power measurements of an integrated circuit.

5.2 FPGA Measurement Setup

We furthermore synthesized the design (with $d = 1$) on an FPGA. This step enables us to record the power consumption during the targeted point multiplication performed on a real-world device. As FPGA a Virtex-II Pro xc2vp7 was used which is part of the SASEBO [35] side-channel evaluation board. As the development environment, Xilinx 10.1.03 was used. For recording the power traces we have used a LeCroy WP725Zi oscilloscope with a sampling rate of 2.5 GS/s and operate the device under test at a clock rate of 25 MHz. Our first measurements showed that for the attacks an accurate clock frequency is beneficial in order to avoid the introduction of noise due to clock jitter. Using an off-the-shelf quartz increases the effort for the attack as additional postprocessing steps on the recorded trace are required. In particular, a fixed number of samples per clock period is required for the attack, that means the ratio between the sampling rate and the clock rate must be integer. If this is not the case an upsampling of the recorded trace is required in order to achieve this ratio. To circumvent this we used an Agilent 33250A signal generator as external clock source. Further the complete point multiplication must be finished before the used oscilloscope runs out of “input buffer”. Our LeCroy WP725Zi oscilloscope is capable of storing 64 million samples per trace.

In order to decrease the calculation and memory effort we performed a downsampling step on the recorded trace. In this preprocessing step, all the sample points within one clock period of the device were summed up. Once again, we want to emphasize that the following attacks work using only *a single power trace* of the scalar multiplication.

6 Assuring Side-Channel Resistance

In order to make sure that our implementation has no key dependent leakage, we performed a basic difference-of-means analysis with a known scalar. By redesigning the hardware, all sources of leakage were eliminated. The difference-of-means trace in Figure 3 shows the leakage of a preliminary version of our hardware design under investigation.

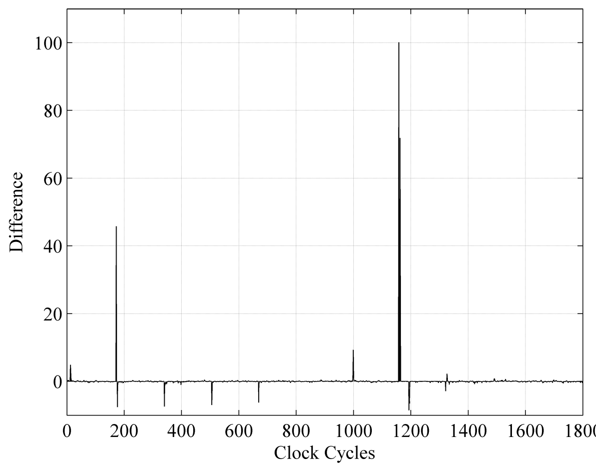

Fig. 3. Difference-of-means of a simulated power trace.

Theory. To launch a difference-of-means attack (cf. [25]), all key-dependent round traces are categorized according to k_i. Next, the average $avg(R_i|_{k_i,\forall i})$ and the difference of the means in the respectable categories $avg(R_i|_{k_i=0,\forall i}) - avg(R_i|_{k_i=1,\forall i})$ are calculated. By investigation of the resulting plot it is possible to find key-dependent operations in the trace.

Practical Results. A difference-of-means calculation was performed using the subtraces R_i extracted from a single measured power trace from an early FPGA implementation and is depicted in Figure 3. The operations at clock cycles with a high difference show some key-dependent behavior. The higher the difference is, the easier it is for an attacker to find the key-dependent parts. This results allow a designer to identify key-dependent operations and improve the design. In our special case we found that at clock cycle 1160, the access to the data memory was dependent on the key bit k_i. For all subsequent experiments this error was obviously fixed.

We redesigned our hardware implementation until the most significant peaks of the difference-of-means analysis vanished. Therefore we covered the basics and assured the significance of the following power analysis attacks.

7 Correlation of Consecutive Rounds

The assumption is that in consecutive rounds R_i and R_{i-1} similarities dependent on the processed bits k_i and k_{i-1} can be found. This holds for our implementation using the formulas of López and Dahab (cf. Appendix A) for point multiplication but is also applicable to other algorithms.

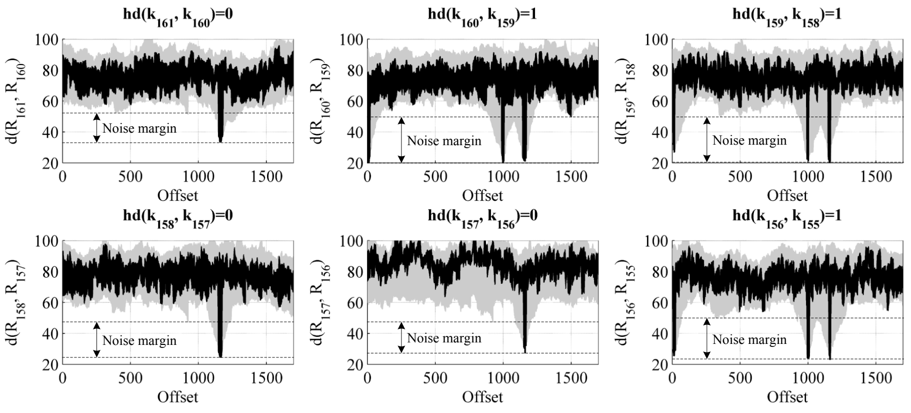

Fig. 4. Euclidean distance traces.

Theory. In this scenario we focus on finding similarities in the power traces R_i and R_{i-1} of two consecutive rounds. Once again it has to be mentioned that these power traces are subtraces of equal length of one power trace recorded during the point multiplication. The idea behind this approach is that the power profile of a digit-serial multiplication is mostly dependent on OpB. If the same operand OpB has been used in two consecutive rounds similarities can be found. Witteman *et al.* [38] have used a similar assumption in their attack on the RSA modular exponentiation. They take advantage of the fact that the square-and-multiply-always algorithm reuses operands in a key-dependent manner. As similarity measure we have tried the correlation coefficient as well as the Euclidean distance. Both approaches lead to comparable results with runtime advantages for the Euclidean distance. Equation 1 shows how to calculate the Euclidean distance of two consecutive rounds with offset o_i, o_{i-1} respectively and a window size w. In Equation 2, the formula to generate the distance matrix for the rounds i and $i-1$ with the given limits for j and l can be found.

$$d(R_i, R_{i-1})|_{o_i, o_{i-1}} = ||R_i(o_i, o_i + w), R_{i-1}(o_{i-1}, o_{i-1} + w)|| \tag{1}$$

$$Dist_l^j(R_i, R_{i-1}) = d(R_i, R_{i-1})|_{j,l} \tag{2}$$

$$|k| \geq i \geq 1; 0 \leq j \leq t_{round} - w; 0 \leq l \leq t_{round} - w;$$

Practical Results. Figure 4 shows the results of a windowed correlation of two consecutive rounds with a window size $w = 163$ using the traces recorded from the FPGA. The length of the subtraces R_i is 1796 samples. Small Euclidean distances are indicators for similar intermediate values. Those traces with small Euclidean distances have been highlighted. Two cases are distinguishable in Figure 4. In the first case (e.g., between rounds 160 and 159) three locations with a small Euclidean distance can be identified. In the second case (e.g., between

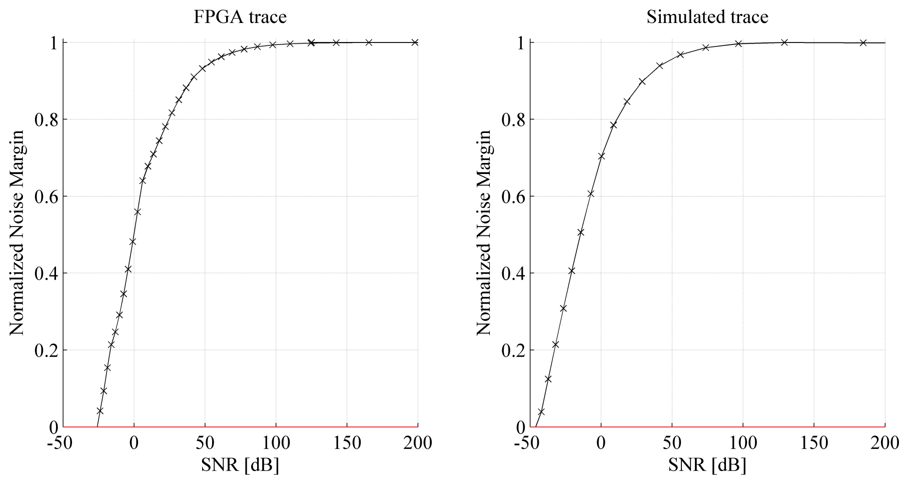

Fig. 5. Influence of noise on the noise margin.

rounds 161 and 160) only one location with a small Euclidean distance can be identified. By investigating the formulas of López and Dahab [24], we identified the peak around offset 1150 as $OpB = c = b^{2^{m-1}} \mod f(z)$. This single peak appears in all correlation figures at the same position. The other two peaks appear, when the Hamming distance $hd(k_i, k_{i-1}) = 1$. In other words: the peaks appear, when the key bit k_i is different from k_{i-1}. In such a case one operand OpB from round i is used again (unchanged) in round $i - 1$. For a better understanding of the underlying problem, we attached the used formulas in Appendix A.

We also performed the same attack on the simulated traces and achieved comparable results. As the simulated traces do not contain measurement noise, the minimum Euclidean distances are even smaller. The presented attack worked for $d = 1$, $d = 2$, and $d = 4$ (used window sizes: $w = 163$, $w = 82$, and $w = 41$ respectively).

In order to visualize the influence of noise (e.g. introduced by the measurement environment or active noise countermeasures), simulated gaussian noise with different power levels has been added to the simulated as well as the measured trace. Figure 5 shows the normalized noise margin as a function of the signal to noise ratio SNR. The SNR has been calculated according to Equation 3. The evolution of the noise margin for simulation and FPGA experiment is similar, only the value of the SNR where the noise margin comes below zero is different. For the FPGA experiment the noise margin comes below zero at -24 dB and for the simulation at -43 dB. The difference can be explained by the fact that the trace recorded with the FPGA already contains some noise. The trace extracted using the simulation on the other hand can be seen as noise free.

$$\mathrm{SNR} = 20 \cdot \log \frac{U_{\mathrm{eff,Signal}}}{U_{\mathrm{eff,Noise}}} \tag{3}$$

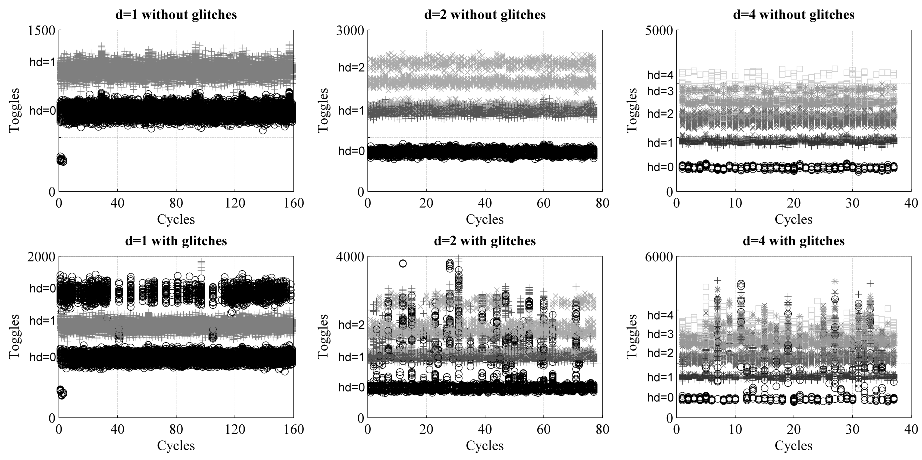

Fig. 6. Simulated power profiles for different d in the presence of glitches.

8 Revealing Intermediate Operands

In this section, we want to uniquely identify the intermediate finite-field polynomials used as operand OpB and discuss some more advanced attack scenarios. For a better understanding of the associated challenges, let us first investigate practical simulations.

Assumptions & Transferability. For this scenario we need to assume that a digit-serial multiplier is used. It is applicable to all designs based on digit-serial multipliers.

Practical Results. Figure 6 shows simulated toggle counts for $|k|$ overlayed N-bit multiplications. Using reference simulations, we marked the different Hamming distances $hd(OpB_i)$ with respective symbols. The following three characteristics are eye-catching: The different Hamming distances are distinguishable. The larger the digit size d is, the less distinguishable are the different levels of power consumption. And the more glitches occur on OpB_i, the less separable are those power levels. In our hardware design, we used an N-bit wide 5:1 multiplexer and a N:d multiplexer to select OpB_i. The resulting glitches are clearly visible in the lower row of Figure 6. The upper row is the result of a d bit register being added after the multiplexers to suppress those glitches[2]. Also if the multiplexers are replaced by shift registers, there are virtually no glitches.

Interestingly, in the case of $d = 1$, glitches only occurred in a $1 \rightarrow 1$ transition. Obviously, the AOI-gates used in our design suppress glitches in $0 \rightarrow 0$ transitions.

[2] For high-performance ECC implementations those registers are necessary in order to achieve the desired timings.

Table 1. Average number of solutions for OpB assuming $hd(OpB)$ is given.

Parameter	**N = 163**	**N = 256**
$d = 1$	$2^1 = 2^1$	$2^1 = 2^1$
$d = 2$	$2^2 2^{0.5\times 81} = 2^{42.5}$	$2^2 2^{0.5\times 127} = 2^{65.5}$
$d = 3$	$2^3 3^{0.75\times 54} = 2^{67.2}$	$2^3 3^{0.75\times 85} = 2^{104}$
$d = 4$	$2^4 4^{0.5\times 40} 6^{0.375\times 40} = 2^{82.8}$	$2^4 4^{0.5\times 63} 6^{0.375\times 63} = 2^{128.1}$

Identifying the Intermediates. At this point, the attacker wants to identify the intermediate values processed during the field multiplications by using the given power traces. The identification of the operand(s) OpB_i has to be done in two phases:

At first, the leaking Hamming distances in the power trace have to be classified. By overlaying all $|k|$ rounds, as it has been done in Figure 6, it is possible to distinguish clusters of different Hamming distances. This classification can for instance be performed using a k-means algorithm [16]. We performed this classification on our simulated power traces and were able to detect the correct Hamming distances with the following error rates: 0.33% for $d = 1$ without and with glitches, 4.13% for $d = 2$ without glitches, 9.31% for $d = 2$ with glitches, 4.96% for $d = 4$ without glitches and 9.04% for $d = 4$ with glitches.

During the second phase, we used the Hamming distances $hd(OpB)$ to iterate through all possible solutions. Several possible solutions for OpB result in the same $hd(OpB)$. Equation 4 gives an average number of possible solutions $N_{solutions}$ in dependence of N and d, when $hd(OpB)$ is given. $\#(hd = h)$ is the number of possible solutions for a fixed Hamming distance h and $p(hd = h)$ is the probability of the Hamming distance h. Since it is necessary to guess OpB_{MSB}, 2^d is a multiplicand factor within the equation. Table 1 gives exemplary numbers for $N = 163$, $N = 256$, and $d = 1...4$.

$$N_{solutions} = 2^d \cdot \left(\prod_{h=0}^{d} \#(hd = h)^{p(hd=h)} \right)^{\lceil \frac{N}{d} \rceil - 1} \tag{4}$$

The following short example should clarify the problematic: When $d = 2$, $hd(OpB_i, OpB_{i+1})$ is either 0, 1, or 2. Whereas $hd = 0$ and $hd = 2$ uniquely map OpB_i to a single OpB_{i+1}, $hd = 1$ does not. Let us assume $OpB_i = 00_b$. Then there are two solutions $hd(00_b, 01_b) = hd(00_b, 10_b) = 1$. With our power model[3], those two solutions cannot be distinguished in the power trace. In average there are $2^{42.5}$ possible solutions (cf. Table 1) for $d = 2$ and $N = 163$. Using brute-force, breaking $2^{42.5}$ is practicable.

Other exemplary cases such as $d > 2$ or $N = 256$ are theoretically possible but impractical. However Table 1 depicts an *average* case. In practice the number of possibilities $N_{solutions}$ for a certain OpB depends on $hw(OpB)$. Hence, it is clever to only attack OpB with a small search space. An approximation of the necessary runtime can be performed in constant time. Furthermore we are

[3] Note that this might be possible using more advanced power models.

confident that by investigating the different arrival times of single bits in OpB_i and by using more detailed timing information, our leakage model can be refined.

For many practical implementations without point randomization, finding those intermediate values would be sufficient. However, in the context of this paper (with point randomization) finding the intermediate values is only a preparational step for the following attack scenarios.

8.1 Correlate with an Arithmetic Combination of Intermediates.

In Section 7, a direct correlation of the power trace on two consecutive rounds was performed. By changing the order of the operands of the multiplication this attack can be prevented. In many cases there is still a link between consecutive rounds when the result of an arithmetical combination of several operands $OpB^1, OpB^2, \ldots$ in round i is used as operand in round $i+1$. If the arithmetical combination $f(\cdot)$ as well as a set of operands $OpB^1, OpB^2, \ldots$ for round i is known, a set of used operands $F = f(OpB^1, OpB^2, \ldots)$ for round $i+1$ can be calculated. Feeding $f(\cdot)$ with every possible combination of $OpB^1, OpB^2, \ldots$ and performing the correlation $d(hd(F), R_{i-1})$ the size of the possible key candidates can be decreased. The complexity of this attack highly depends on the number of operands N_{OpB} used as arguments for $f(\cdot)$ as well as on d. N_{OpB} as well as d have an influence on the number of possible results for F. If e.g. $f(\cdot)$ is a squaring function ($N_{OpB} = 1$) and $d \leq 2$ the attack can be performed within acceptable bounds. Furthermore it has to be considered that if some bits in F are wrong there might still be a significant peak in the correlation plot, so there is no need to find the exact value of F. This fact also decreases the attack complexity.

8.2 Attack Several Intermediates Simultaneously.

An enhancement of the previous scenario is to attack $(OpB^1, OpB^2, \ldots)$ as well as $F = f(OpB^1, OpB^2, \ldots)$ simultaneously. Let us assume the Hamming distances of F and $OpB^1, OpB^2, \ldots$ are known, so only a limited number of combinations for F and $OpB^1, OpB^2, \ldots$ fulfill the equation $F = f(OpB^1, OpB^2, \ldots)$. Although we did not test it, we are confident that by attacking different intermediates simultaneously the search space $N_{solutions}$ can be reduced by a certain degree.

8.3 Find the x-Coordinate.

If an attacker can reveal the exact values for X_i (projective X coordinate in round i) and Z_i, she can calculate the x-coordinate of the currently processed point. Even if X_i and Z_i have been randomized and multiplied with a random λ this attack works. In the case of $X_r = X_i \cdot \lambda$ and $Z_r = Z_i \cdot \lambda$,

$$x_i = X_r \cdot Z_r^{-1} = (\lambda X_i) \cdot (\lambda Z_i)^{-1} = X \cdot Z^{-1} \quad (5)$$

can be recovered. Assuming a scalar multiplication $Q = k \times P$ is performed, small multiples of P can be precalculated and compared with x_i which is revealed in each round. Thus step by step all bits of k can be revealed. Even if there is an error in round i and x_i cannot be matched, x_{i+1} can be used to identify more key bits at once.

8.4 Undo the Projective Coordinate Randomization.

In order to perform a projective coordinate randomization, each coordinate is multiplied with a random number λ. For that operation usually the same finite-field multiplier is used as the finite-field multiplier used during a point multiplication. So in the case of $X_r = X \cdot \lambda$ and λ is used as OpB, the randomization factor can be found. By knowing λ many attack scenarios on a device doing a point multiplication become feasible.

9 Conclusion

Nowadays the community knows that no implementation is believed to be secure as long as it has not been attacked and investigated in sufficient detail. In this paper, we attacked a protected ECC implementation by using only a single power trace. So even the popularly used ECDSA and the Diffie-Hellman key exchange algorithms are vulnerable. The corollary one should take away from this paper is that a designer must not only consider a simple difference-of-means attack but also be aware of more advanced and unexpected attack scenarios such as the here shown custom correlation attack or the attack on all processed intermediate values. But how should any designer be aware of future, currently unknown attacks?

Acknowledgments

The research described in this paper has been supported, in parts, by the European Commission through the ICT Program under contract ICT-SEC-2009-5-258754 TAMPRES, and by the Austrian Research Promotion Agency (FFG) and the Styrian Business Promotion Agency (SFG) under grant number 836628 (SeCoS).

References

1. H. Aigner, H. Bock, M. Hütter, and J. Wolkerstorfer. A Low-Cost ECC Coprocessor for Smartcards. In M. Joye and J.-J. Quisquater, editors, *CHES 2004, Proceedings*, volume 3156 of *LNCS*, pages 107–118. Springer, 2004.
2. T. Akishita and T. Takagi. Power Analysis to ECC Using Differential Power Between Multiplication and Squaring. In *CARDIS 2006*, volume 3928 of *LNCS*, pages 151–164, April 2006.
3. F. Amiel, B. Feix, M. Tunstall, C. Whelan, and W. Marnane. Distinguishing Multiplications from Squaring Operations. In R. Avanzi, L. Keliher, and F. Sica, editors, *Selected Areas in Cryptography*, volume 5381 of *LNCS*, pages 346–360. Springer, 2009.

4. L. Batina, N. Mentens, S. B. Örs, and B. Preneel. Serial Multiplier Architectures over GF(2^n) for Elliptic Curve Cryptosystems. In *IEEE Mediterranean Electronical Conference – MELECON 2004*, pages 779–782. IEEE, May 2004.
5. L. Batina, N. Mentens, K. Sakiyama, B. Preneel, and I. Verbauwhede. Low-Cost Elliptic Curve Cryptography for Wireless Sensor Networks. In L. Buttyán, V. Gligor, and D. Westhoff, editors, *Security and Privacy in Ad-Hoc and Sensor Networks – ESAS 2006, Revised Selected Papers*, volume 4357, pages 6–17, Berlin Heidelberg, 2006. Springer-Verlag.
6. H. Bock, M. Braun, M. Dichtl, E. Hess, J. Heyszl, W. Kargl, H. Koroschetz, B. Meyer, and H. Seuschek. A Milestone Towards RFID Products Offering Asymmetric Authentication Based on Elliptic Curve Cryptography. Invited talk at RFIDsec 2008, July 2008.
7. D. Brumley and D. Boneh. Remote timing attacks are practical. *Computer Networks*, 48(5):701–716, 2005.
8. C. Clavier, B. Feix, G. Gagnerot, M. Roussellet, and V. Verneuil. Horizontal Correlation Analysis on Exponentiation. In M. Soriano, S. Qing, and J. López, editors, *Information and Communications Security*, volume 6476 of *LNCS*, pages 46–61. Springer, 2010.
9. J.-S. Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Ç. K. Koç and C. Paar, editors, *CHES 1999, Proceedings*, volume 1717 of *LNCS*, pages 292–302. Springer, 1999.
10. J.-F. Dhem, F. Kœune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, and J.-L. Willems. A Practical Implementation of the Timing Attack. In J.-J. Quisquater and B. Schneier, editors, *CARDIS 1998, Proceedings*, number 1820 in LNCS, pages 167–182. Springer, 1998.
11. H. Eberle, N. Gura, S. C. Shantz, and V. Gupta. A Cryptographic Processor for Arbitrary Elliptic Curves over GF(2^m). In E. Deprettere, S. Bhattacharyya, J. Cavallaro, A. Darte, and L. Thiele, editors, *Application-Specific Systems, Architectures, and Processors – ASAP 2003*, pages 444–454, June 2003.
12. F. Fürbass and J. Wolkerstorfer. ECC Processor with Low Die Size for RFID Applications. In *Proceedings of 2007 IEEE International Symposium on Circuits and Systems*. IEEE, IEEE, May 2007.
13. C. H. Gebotys and R. J. Gebotys. Secure Elliptic Curve Implementations: An Analysis of Resistance to Power-Attacks in a DSP Processor. In B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, editors, *CHES 2002, Revised Papers*, volume 2523 of *LNCS*, pages 114–128. Springer, 2003.
14. J. Großschädl. A Bit-Serial Unified Multiplier Architecture for Finite Fields GF(p) and GF(2^m). In Ç. K. Koç, D. Naccache, and C. Paar, editors, *CHES 2001, Proceedings*, volume 2162, pages 202–219. Springer Berlin / Heidelberg, May 2001.
15. D. Hankerson, A. J. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography.* Springer, Berlin, Germany / Heidelberg, Germany / London, UK / etc., 2004.
16. J. A. Hartigan and M. A. Wong. Algorithm AS 136: A K-Means Clustering Algorithm. volume 28, pages 100–108. Blackwell Publishing for the Royal Statistical Society, 1979.
17. C. Herbst and M. Medwed. Using Templates to Attack Masked Montgomery Ladder Implementations of Modular Exponentiation. In K.-I. Chung, M. Yung, and K. Sohn, editors, *Workshop on Information Security Applications – WISA 2008, Proceedings*, volume 5379 of *LNCS*, pages 1–13. Springer, Februar 2008.

18. N. Homma, A. Miyamoto, T. Aoki, A. Satoh, and A. Shamir. Comparative Power Analysis of Modular Exponentiation Algorithms. *IEEE Transactions on Computers*, 59(6):795–807, 2010.
19. M. Kirschbaum and T. Popp. Evaluation of Power Estimation Methods Based on Logic Simulations. In K. C. Posch and J. Wolkerstorfer, editors, *Proceedings of Austrochip 2007, October 11, 2007, Graz, Austria*, pages 45–51. Verlag der Technischen Universität Graz, October 2007. ISBN 978-3-902465-87-0.
20. P. C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In N. Koblitz, editor, *Advances in Cryptology - CRYPTO 1996, Proceedings*, number 1109 in LNCS, pages 104–113. Springer, 1996.
21. P. C. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis. In *CRYPTO*, pages 388–397, 1999.
22. S. S. Kumar and C. Paar. Are standards compliant Elliptic Curve Cryptosystems feasible on RFID? In *Workshop on RFID Security – RFIDSec 2006*, 2006.
23. Y. K. Lee, K. Sakiyama, L. Batina, and I. Verbauwhede. Elliptic-Curve-Based Security Processor for RFID. *IEEE Transactions on Computers*, 57(11):1514–1527, November 2008.
24. J. López and R. Dahab. Fast Multiplication on Elliptic Curves over GF(2^{m}) without Precomputation. In Ç. K. Koç and C. Paar, editors, *CHES 1999, Proceedings*, volume 1717 of *LNCS*, pages 316–327. Springer, 1999.
25. S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks – Revealing the Secrets of Smart Cards*. Springer, 2007. ISBN 978-0-387-30857-9.
26. M. Medwed and E. Oswald. Template Attacks on ECDSA. In K.-I. Chung, M. Yung, and K. Sohn, editors, *Workshop on Information Security Applications – WISA 2008, Pre-Proceedings*, pages 14–27, 2008.
27. B. Möller. Securing Elliptic Curve Point Multiplication against Side-Channel Attacks. In G. I. Davida and Y. Frankel, editors, *Information Security Conference – ISC 2001, Proceedings*, volume 2200 of *LNCS*, pages 324–334. Springer, 2001.
28. National Institute of Standards and Technology (NIST). FIPS-186-3: Digital Signature Standard (DSS), 2009. Available online at `http://www.itl.nist.gov/fipspubs/`.
29. NXP. Jcop 41 v2.3.1 java card, 2007.
30. G. Orlando and C. Paar. A High-Performance Reconfigurable Elliptic Curve Processor for GF(2^m). In Ç. K. Koç and C. Paar, editors, *CHES 2000, Proceedings*, volume 1965 of *0302-9743*, pages 41–56, London, UK, August 2000. Springer.
31. S. B. Örs, E. Oswald, and B. Preneel. Power-Analysis Attacks on FPGAs – First Experimental Results. In C. D. Walter, Ç. K. Koç, and C. Paar, editors, *CHES 2003, Proceedings*, volume 2779 of *LNCS*, pages 35–50. Springer, 2003.
32. E. Oswald. Enhancing Simple Power-Analysis Attacks on Elliptic Curve Cryptosystems. In B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, editors, *CHES 2002, Revised Papers*, volume 2523 of *LNCS*, pages 82–97. Springer, 2003.
33. E. Öztürk, B. Sunar, and E. Savas. Low-Power Elliptic Curve Cryptography Using Scaled Modular Arithmetic. In M. Joye and J.-J. Quisquater, editors, *CHES 2004, Proceedings*, volume 3156 of *LNCS*, pages 92–106. Springer, August 2004.
34. W. Pan and W. P. Marnane. A Correlation Power Analysis Attack against Tate Pairing on FPGA. In *Reconfigurable Computing: Architectures, Tools and Applications – ARC 2011, Proceedings*, volume 6578 of *LNCS*, pages 340–349. Springer-Verlag, 2011.
35. Side-channel attack standard evaluation board. The SASEBO Website. `http://staff.aist.go.jp/akashi.satoh/SASEBO/en/index.html`.

36. C. D. Walter. Sliding Windows Succumbs to Big Mac Attack. In Ç. K. Koç, D. Naccache, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems CHES 2001*, volume 2162 of *LNCS*, pages 286–299. Springer, 2001.
37. C. D. Walter. Simple Power Analysis of Unified Code for ECC Double and Add. In M. Joye and J.-J. Quisquater, editors, *CHES 2004, Proceedings*, volume 3156 of *LNCS*, pages 191–204. Springer, 2004.
38. M. F. Witteman, J. G. van Woudenberg, and F. Menarini. Defeating RSA Multiply-Always and Message Blinding Countermeasures. In S. B. Heidelberg, editor, *Topics in Cryptology CT-RSA*, pages 77–88, 2011.
39. J. Wolkerstorfer. Is Elliptic-Curve Cryptography Suitable for Small Devices? In *Workshop on RFID and Lightweight Crypto, July 13-15, 2005, Graz, Austria*, pages 78–91, 2005.

A Used Double-And-Add Formula

In this paper we used a slight modification of the Montgomery Ladder by López and Dahab. Algorithm 1 shows three iterations of the used double-and-add algorithm for three consecutive key bits. The modification from the original formula can be found in line 6. Here we swapped the order of x and Z_1, which is allowed according to the law of commutativity. During the first two iterations (left and middle column), an identical key bit is handled. So there is only a correlation when the constant c is used. During the second and third iteration, in which the key bit differs, Z_1 is used multiple times. Consequently in Figure 4 three easily distinguishable peaks occur. A correlation of c and Z_1 can be observed.

Algorithm 1 López and Dahab round operations with key bits (0-0-1).

```
Ensure: $P_1' \leftarrow P_1 + P_2$.
Ensure: $P_2' \leftarrow 2 \cdot P_2$.
    Point Addition
 1: $X_1 \leftarrow X_1 \cdot Z_2$
 2: $Z_1 \leftarrow Z_1 \cdot X_2$
 3: $T_1 \leftarrow X_1 \cdot Z_1$
 4: $Z_1 \leftarrow Z_1 + X_1$
 5: $Z_1 \leftarrow Z_1 \cdot Z_1$
 6: $X_1 \leftarrow x \cdot Z_1$
 7: $X_1 \leftarrow X_1 + T_1$
    Point Doubling
 8: $X_2 \leftarrow X_2 \cdot X_2$
 9: $Z_2 \leftarrow Z_2 \cdot Z_2$
10: $T_1 \leftarrow Z_2 \cdot \boxed{\mathbf{c}}$
11: $Z_2 \leftarrow Z_2 \cdot X_2$
12: $T_1 \leftarrow T_1 \cdot T_1$
13: $X_2 \leftarrow X_2 \cdot X_2$
14: $X_2 \leftarrow X_2 + T_1$
```

```
Ensure: $P_1' \leftarrow P_1 + P_2$.
Ensure: $P_2' \leftarrow 2 \cdot P_2$.
    Point Addition
 1: $X_1 \leftarrow X_1 \cdot Z_2$
 2: $Z_1 \leftarrow Z_1 \cdot X_2$
 3: $T_1 \leftarrow X_1 \cdot Z_1$
 4: $Z_1 \leftarrow Z_1 + X_1$
 5: $Z_1 \leftarrow Z_1 \cdot Z_1$
 6: $X_1 \leftarrow x \cdot \boxed{\mathbf{Z_1}}$
 7: $X_1 \leftarrow X_1 + T_1$
    Point Doubling
 8: $X_2 \leftarrow X_2 \cdot X_2$
 9: $Z_2 \leftarrow Z_2 \cdot Z_2$
10: $T_1 \leftarrow Z_2 \cdot \boxed{\mathbf{c}}$
11: $Z_2 \leftarrow Z_2 \cdot X_2$
12: $T_1 \leftarrow T_1 \cdot T_1$
13: $X_2 \leftarrow X_2 \cdot X_2$
14: $X_2 \leftarrow X_2 + T_1$
```

```
Ensure: $P_2' \leftarrow P_2 + P_1$.
Ensure: $P_1' \leftarrow 2 \cdot P_1$.
    Point Addition
 1: $X_2 \leftarrow X_2 \cdot \boxed{\mathbf{Z_1}}$
 2: $Z_2 \leftarrow Z_2 \cdot X_1$
 3: $T_1 \leftarrow X_2 \cdot Z_2$
 4: $Z_2 \leftarrow Z_2 + X_2$
 5: $Z_2 \leftarrow Z_2 \cdot Z_2$
 6: $X_2 \leftarrow x \cdot Z_2$
 7: $X_2 \leftarrow X_2 + T_1$
    Point Doubling
 8: $X_1 \leftarrow X_1 \cdot X_1$
 9: $Z_1 \leftarrow Z_1 \cdot \boxed{\mathbf{Z_1}}$
10: $T_1 \leftarrow Z_1 \cdot \boxed{\mathbf{c}}$
11: $Z_1 \leftarrow Z_1 \cdot X_1$
12: $T_1 \leftarrow T_1 \cdot T_1$
13: $X_1 \leftarrow X_1 \cdot X_1$
14: $X_1 \leftarrow X_1 + T_1$
```