Journal of
*Low Power Electronics and Applications*

MDPI

*Article*

# Completing the Complete ECC Formulae with Countermeasures

**Łukasz Chmielewski [1], Pedro Maat Costa Massolino [2], Jo Vliegen [3], Lejla Batina [2] and Nele Mentens [3,*]**

[1] Riscure BV, 2628 XJ Delft, The Netherlands; Chmielewski@riscure.com
[2] Institute for Computing and Information Sciences (ICIS), Radboud University, 6525 HP Nijmegen, The Netherlands; P.Massolino@cs.ru.nl (P.M.C.M.); lejla@cs.ru.nl (L.B.)
[3] KU Leuven-imec-COSIC, KU Leuven, 3000 Leuven, Belgium; jo.vliegen@esat.kuleuven.be
[*] Correspondence: nele.mentens@kuleuven.be; Tel.: +32-16-325063

Academic Editors: Osnat Keren, Ilia Polian and Sanu Mathew
Received: 30 October 2016; Accepted: 22 January 2017; Published: 30 January 2017

**Abstract:** This work implements and evaluates the recent complete addition formulae for the prime order elliptic curves of Renes, Costello and Batina on an FPGA platform. We implement three different versions: (1) an unprotected architecture; (2) an architecture protected through coordinate randomization; and (3) an architecture with both coordinate randomization and scalar splitting in place. The evaluation is done through timing analysis and test vector leakage assessment (TVLA). The results show that applying an increasing level of countermeasures leads to an increasing resistance against side-channel attacks. This is the first work looking into side-channel security issues of hardware implementations of the complete formulae.

**Keywords:** elliptic curve cryptography; FPGA; Weierstrass curves; complete formulas; side-channel attacks and countermeasures

## 1. Introduction

Public-key cryptography in constrained embedded devices is usually based on elliptic curve cryptography (ECC) because of the relatively low resource and power consumption compared to other public-key systems [1,2]. Since modern use case scenarios in the Internet of Things usually allow possible attackers to be in the vicinity of the embedded device, the leakage of sensitive information through side-channels becomes a realistic threat. The first step in making an implementation side-channel resistant is to protect it against simple power analysis (SPA) attacks [3]. This can be done by making the execution time and the instantaneous power consumption of the operations independent of the processed data and executed instructions. In the case of ECC, this strategy should be applied, e.g., to the point multiplication algorithm, the point addition and point doubling algorithms and the operations in the underlying finite field. More powerful than SPA are side-channel attacks that use statistics to process many measurements and exploit the correlation between (secret) data and physical leakages. This kind of side-channel attack is usually referred to as differential power analysis (DPA) [3]. An effective way of protecting ECC implementations against DPA attacks at the algorithmic level is to apply randomization countermeasures. Examples are scalar blinding and point blinding, which randomize the point representation and the key bits' evaluation [4].

As the point operations are different in principle, there exist different formulae to compute an addition of two different points or a doubling of one point. This has led to insecure implementations, as the two operations feature different power consumption patterns. Recently, there have been efforts to use a single set of formulae to compute both, point addition and doubling [5]. Although this slows down the implementation, it is an important first step in making the implementation side-channel resistant. The formulae derived in [5] are only applicable to a special type of curve, i.e., so-called Edwards curves. Nevertheless, the idea has enabled the balancing of point operations in an intrinsic manner, i.e., without adding dummy operations [6].

In this paper, we evaluate the power analysis resistance of a completely balanced ECC implementation, based on recent advances in the development of complete addition formulae for all prime order Weierstrass curves by Renes et al. [7]. In a second step, we protect the implementation using point randomization techniques, and in a third step, we use random scalar splitting. The architecture is implemented on a SAKURA-GII board containing a Spartan-6 FPGA. This is the first paper that addresses the side-channel security of unprotected and protected hardware implementations of the complete formulas by Renes et al.

The paper is structured as follows. In Section 2, we give background information on the three implementation versions and the type of power analysis attacks we perform. Section 3 gives an overview of related work on protected ECC implementations. In Section 4, the experimental setup is described. Section 5 presents and discusses the measurement results. Finally, Section 6 concludes the paper and gives an outlook on future work.

## 2. Background Information

Here, we give some relevant background information on elliptic curves over a prime field, and we recall the complete formulas of Renes et al. [7]. In addition, we discuss some issues in side-channel analysis that are ECC specific.

### *2.1. Formulas*

Let $\mathbb{F}_q$ be a finite field of characteristic $p$, i.e., $q = p^n$ for some $n$, and assume that $p \neq 2,3$. Typically, the curves used in security applications are defined over $\mathbb{F}_p$, so for $n = 1$, but the formulas for the elliptic curve addition/doubling work for any $n$. For arbitrary $a, b \in \mathbb{F}_q$, an elliptic curve $E$ over $\mathbb{F}_q$ is defined as the set of solutions $(x, y)$ to the curve equation $E : y^2 = x^3 + ax + b$ with an additional point $\mathcal{O}$, called the point at infinity. Those points $(x, y)$ form a group, with $\mathcal{O}$ as its identity element. One has to make sure that $a$ and $b$ are chosen to meet the security requirements as defined by ECC standards.

Elliptic curve cryptography [1,2] relies on the difficulty of the elliptic curve discrete logarithm problem (ECDLP). This means that given two points $P, Q$ on an elliptic curve, it is hard to find a scalar $k \in \mathbb{Z}$ such that $Q = kP$, if it exists. Therefore, the main component of curve-based cryptosystems is the scalar multiplication operation $(k, P) \mapsto kP$. Namely, all ECC protocols are typically based on a few scalar multiplications, i.e., the computations of $kP$ where $k$ is a scalar and $P$ is a known point. The computation of $kP$ is performed via repeated point additions ($P + Q$) and doublings ($P + P$ = 2P). Both operations can be performed by the use of the same sequence of instructions [7] that consist of several finite field operations, i.e., modular multiplications, additions and inversions. The exact counts of field operations depend on the choice of curves and coordinates; see [8]. Modular multiplications are much more expensive than additions in terms of time, area and memory, but the most expensive are inversions. One way to avoid inversions is to work with projective coordinates. In this case, we choose a different point representation, i.e., we represent points with projective coordinates.

An equivalence relation $\sim$ on $\mathbb{F}_q^3$ is defined by letting $(x_0, x_1, x_2) \sim (y_0, y_1, y_2)$ if and only if there exists $\lambda \in \mathbb{F}_q^*$ such that $(x_0, x_1, x_2) = (\lambda y_0, \lambda y_1, \lambda y_2)$.

Then, the projective plane over $\mathbb{F}_q$, denoted $\mathbb{P}^2(\mathbb{F}_q)$, is defined by $\mathbb{F}_q^3 \setminus \{(0,0,0)\}$ modulo the equivalence relation $\sim$. We write $(x_0{:}x_1{:}x_2)$ to emphasize that the tuple belongs to $\mathbb{P}^2(\mathbb{F}_q)$ as opposed to $\mathbb{F}_q^3$. Now, we can define $E(\mathbb{F}_q)$ to be the set of solutions $(X{:}Y{:}Z) \in \mathbb{P}^2(\mathbb{F}_q)$ to the curve equation $E : Y^2 = X^3 + aXZ^2 + bZ^3$. Note that we can easily map between the two representations by $(x, y) \mapsto (x : y : 1)$, $\mathcal{O} \mapsto (0 : 1 : 0)$ and $(X : Y : Z) \mapsto (X/Z, Y/Z)$ (for $Z \neq 0$), (0:1:0) $\mapsto \mathcal{O}$.

Using projective coordinates, there are no inversions, but the number of modular multiplications increases, which makes the design of a hardware multiplier crucial for efficient implementations.

The work of Renes et al. [7] presents addition formulas (to realize the group law) for curves in the short Weierstrass form embedded in the projective plane. They compute the sum of two points $P = (X_1 : Y_1 : Z_1)$ and $Q = (X_2 : Y_2 : Z_2)$ as $P + Q = (X_3 : Y_3 : Z_3)$, where:

$$\begin{aligned} X_3 &= (X_1Y_2 + X_2Y_1)(Y_1Y_2 - a(X_1Z_2 + X_2Z_1) - 3bZ_1Z_2) - \\ &\quad (Y_1Z_2 + Y_2Z_1)(aX_1X_2 + 3b(X_1Z_2 + X_2Z_1) - a^2Z_1Z_2), \\ Y_3 &= (3X_1X_2 + aZ_1Z_2)(aX_1X_2 + 3b(X_1Z_2 + X_2Z_1) - a^2Z_1Z_2) + \\ &\quad (Y_1Y_2 + a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2)(Y_1Y_2 - a(X_1Z_2 + X_2Z_1) - 3bZ_1Z_2), \\ Z_3 &= (Y_1Z_2 + Y_2Z_1)(Y_1Y_2 + a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2) + \\ &\quad (X_1Y_2 + X_2Y_1)(3X_1X_2 + aZ_1Z_2). \end{aligned} \tag{1}$$

### 2.2. *Side-Channel Analysis and Countermeasures*

In many ECC applications that compute $kP$, $k$ is a secret key. This implies that this operation has to be protected against all attacks. In particular, many side-channel attacks [3,9] and countermeasures [4] have been proposed. To ensure protection against SPA attacks, it is important to use regular scalar multiplication algorithms, e.g., Montgomery ladder [10] or double-and-add-always [4], executing both an addition and a doubling operation per scalar bit. On the other hand, the regularity is also important for the group operation, so addition and doubling should preferably be executed via an identical sequence of field operations. This suggests a clear preference for the complete formulas. Our first implementation is based on these formulas.

#### 2.2.1. Countermeasures

In order to protect the implementation against DPA attacks, we use projective coordinate randomization in our second implementation. This countermeasure exploits the fact that the $Z$-coordinate can be chosen randomly [4]. This comes down to choosing a different $Z$-coordinate for each point multiplication during the conversion of the input point $P$ to projective coordinates.

In a third implementation, we implement further protection mechanisms against DPA attacks by adding randomized scalar splitting. In this countermeasure, the scalar multiplication $kP$ is randomly split into two scalar multiplications, namely $rP$ and $(k–r)P$, with $r$ a random number.

#### 2.2.2. Test Vector Leakage Assessment Methodology

We evaluate the SPA and DPA leakage of our hardware architecture by running the test vector leakage assessment methodology (TVLA) [11–13]; we follow and extend the TVLA approach for ECC from [14]. TVLA is a testing methodology for side-channel resistance validation that is based on the following rationale: side-channel attacks, such as SPA and DPA, exploit the presence of information about any sensitive intermediates within the traces collected from a device. The approach uses statistical hypothesis testing to detect if one of a number of sensitive intermediates significantly influences the measurement data.

TVLA consists of two phases. The measurement phase is based on the collection of side-channel traces when standardized test vectors are provided as input to the algorithm being tested and establishes requirements for power measurement equipment and setup, data collection, signal alignment and pre-processing. The analysis phase is based on Welch's $t$-test, which can detect different types of leakages and allows the analyst to identify points in time that deserve further investigation. The testing methodology has so far been applied to AES [11], RSA [12,13] and ECC [13,14].

## 3. Related Work

### *ECC Hardware Implementations*

There are numerous works published on hardware implementations of ECC focusing on various platforms, fields, curves and bases. Even if we narrow our choice to elliptic curves over prime fields only, there are numerous papers in the literature that could be mentioned. To avoid being exhaustive on the topic, we focus on more recent hardware implementations of ECC over prime fields. For those interested more generally in the topic, a recent survey is done by Marzouqi et al. [15].

Hardware implementations allow for different trade-offs of resources (in terms of silicon area, configurable look-up tables, embedded mathematical and memory blocks), operational speed (in terms of latency or throughput) and power or energy consumption. In this work, we concentrate on a low-resource implementation. Some publications that also use the same approach are by Roy et al. [16], Pöpper et al. [17] and Vliegen et al. [18], where the last one is the basis of this work. All of these implementations use the same approach of having a very small datapath on which the algorithmic operations are executed in a sequential way. Because of the very restricted amount of resources, the number of cycles can increase quadratically in time for certain multiplication algorithms. Since all ECC operations are constructed on top of field arithmetic, the latency of an ECC point multiplication easily grows. To compensate this loss in time, resource-constrained architectures are usually carefully optimized to try to do all operations in the least amount of cycles. For example, Roy et al. [16] optimize the reduction for NIST curves, while Vliegen et al. [18] optimize for Montgomery multiplication.

On the other side of the spectrum, there is the work of Alrimeih and Rakhmatov [19] and Guillermin [20], which focus on a higher speed by utilizing a large amount of resources. This trade-off between area and time resources can only be optimized with an application scenario in mind. Some applications, like servers, will need high throughput, while entrance authorization and vehicle communication need low latency. In IoT use cases, very inexpensive and low power devices are required.

A different field arithmetic implementation is presented by Guillermin [20] and Esmaeildoust et al. [21], based on the residue-number-system (RNS). Usually, implementations represent the values in one basis, thus requiring a multiplier of the same size of the basis or iterating several steps in a smaller multiplier. RNS-based implementations work with different small bases. Therefore, one can split the one multiplication into several smaller and faster multiplications and combine all results afterwards. Because of working in a parallel way, RNS implementations require many resources, but also have very good timing results.

Research based on different types of curves is performed by Sasdrich and Güneysu [22], Baldwin et al. [23] and Järvinen et al. [24]. These implementations are not based on standard Weierstrass curves, but on curves with more efficient and faster arithmetic. Unfortunately, some applications do not allow the use of those new curves. Nevertheless, given the high efficiency, this scenario is expected to change.

In terms of adding side-channel protection, there is the work of Ghosh et al. [25] and Pöpper et al. [17]. These solutions add both simple and differential power analysis protection. They show that even though most ECC implementations tend to focus on scalar multiplication only, side-channel protection needs to be evaluated on a higher level for real-life applications.

## 4. Experimental Setup

This section first elaborates on the hardware architecture, which is calculating the point operations. Subsequently, the setup to perform the actual measurements is discussed.

### *4.1. Hardware Architecture*

The hardware architecture is a standalone 32-bit elliptic curve processor as presented in [18]. A block diagram of this processor is depicted in Figure 1 and consists of the typical components: an instruction memory, a data memory, a modular arithmetic and logic unit (ALU) and a control unit. The curve on which the processor operates is configurable trough the initialization of the data memory and is set to the NIST standardized P-256 curve.

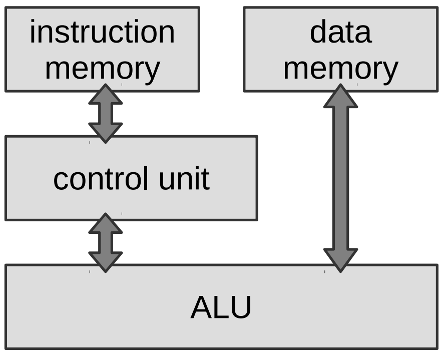

**Figure 1.** Block diagram of the elliptic curve processor.

The ALU has two internal operations: a modular addition/subtraction (MAS) and a Montgomery multiplier (MM) that executes $A \times B \times R^{-1}$. The architectures of the MAS and MM are shown in Figure 2. A conversion to Montgomery form (and back) is required: $A = a \times R$, with $A$ the Montgomery form of $a$. As originally publish by Montgomery [26], the Montgomery multiplication requires a final subtraction. This can be avoided at the cost of an additional word in the datapath, as presented by Walter in [27]. Therefore, the data memory has a width of $256 + w$ bits, which, for a 32-bit processor, rounds up to a width of 288 bits.

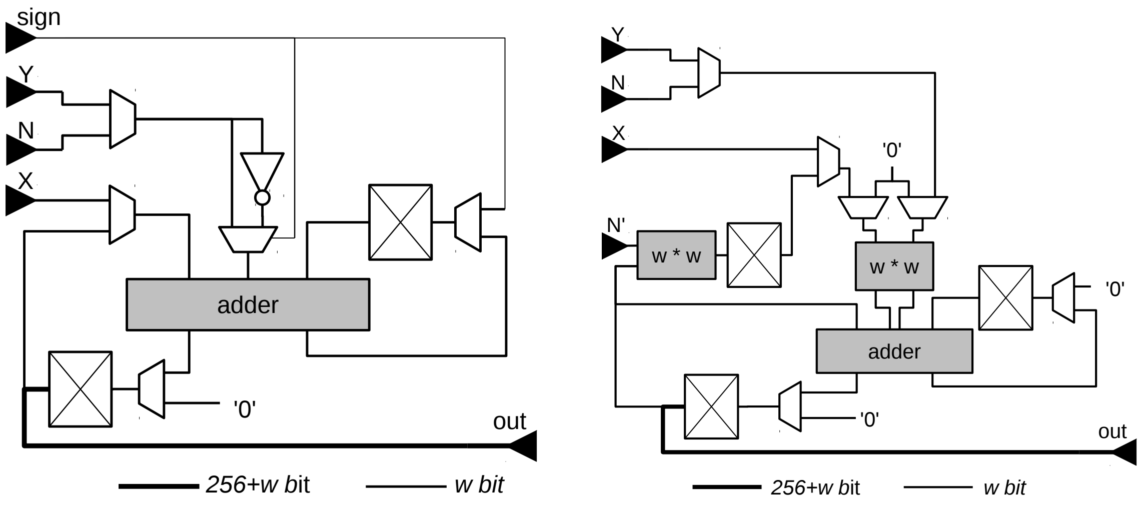

**Figure 2.** The architectures of the modular addition/subtraction (MAS) (left) and Montgomery multiplier (MM) (right) operations.

Both the instruction and the data memory use the internal block RAM (BRAM) of the FPGA. They are initialized with the configuration of the FPGA. The processor first initializes all internal registers in the control unit with values from the data memory. Subsequently, it starts the execution of the core operations, which are:

1. conversion of the coordinates of the point to Montgomery form and to homogeneous coordinates: $P(x, y) \rightarrow Q(QX, QY, QZ)$
2. initialization of $S$: $2Q \rightarrow S(SX, SY, SZ)$
3. execution of the Montgomery ladder [28], performing a scalar multiplication of a hard coded scalar with $P$
4. calculation of the modular inverse of $QZ^{-1}$

5. conversion of $Q$ back to affine coordinates and undoing the Montgomery form $Q(QX, QY, QZ) \rightarrow scalarP(x, y)$

The modular inverse is calculated by using Fermat's little theorem, which states: $m^p \equiv m$ mod $p$ when gcd($m$, $p$) = 1. Because $p$ is a prime number, the latter condition is met. The modular inverse can hence be computed with $m^{-1} \equiv m^{p-2}$ mod $p$. This exponentiation is achieved by a square-and-multiply operation, using the modular multiplication unit in the ALU.

The ECPcomponent is wrapped into a top level component. This wrapping provides an interface for the measurement setup, discussed in Section 4.2. Table 1 summarizes the required resources on a Xilinx Spartan-6 LX75 FPGA. This table reports both the occupied resources for the standalone ECP and for the wrapped ECP.

**Table 1.** The FPGA resource usage of the experimental setup.

| FPGA Resource | ECPOnly | Full |
|---|---|---|
| Number of slice registers | 2274 | 2892 |
| Number of slice LUTs | 2421 | 2752 |
| Number of RAMB16 | 9 | 9 |
| Number of DSP48A1 | 7 | 7 |

Adding countermeasures to the ECP component does not have an impact on the resource occupation, because these modifications only touch the content of the instruction memory in BRAM. The reported number of nine BRAMs in Table 1 breaks down into eight BRAMs for the data memory and one BRAM for the instruction memory. For all versions of the ECP, the instructions fit into one BRAM.

## 4.2. Measurement Setup

The power measurement setup, shown in Figure 3, consists of a oscilloscope, an FPGA board and a commodity PC. The devices are arranged in such a way that the PC can coordinate and store all power measurements/traces.

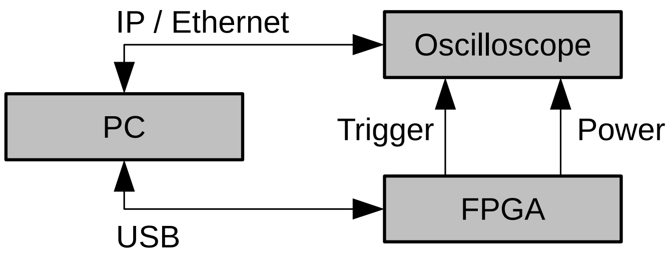

**Figure 3.** The measurement setup.

The oscilloscope in our experiment is the Teledyne Lecroy Waverunner 610Zi. It was configured with $10^8$ samples/s and at most $32 \times 10^6$ samples. This is enough in terms of sample rate and sample size, since the cryptographic core is running at 6 MHz and takes about 300 ms to complete one point multiplication. Furthermore, the oscilloscope has TCP/IP support for both controlling and downloading the measurements, which helps to automatize the entire process.

The FPGA board in Figure 3 is the SAKURA-GII board [29]. The board has two FPGAs, one USB/serial controller and two separate power regulators. For our setup, one FPGA was configured to act as an interface [29] between the serial inputs and the other FPGA that contains the ECC core. Although the ECC core can handle a larger clock frequency, we let it operate at a 6-MHz clock, given the 48-MHz frequency of the USB/serial communication. The separation of the two FPGAs, with different power regulators, lowers the amount of noise in the measurements introduced by the USB communication.

The traces were acquired with the following procedure. First, the PC configures the oscilloscope in terms of the number of channels, the trigger event and the number of samples. Then, the elliptic curve parameters are sent via USB to the FPGA board SAKURA-GII. After verifying the correct reception of the parameters, the PC signalizes the FPGA to start. When the FPGA receives the start signal, it automatically sends a trigger signal to the oscilloscope. After finishing both the power acquisition and the FPGA computation, the PC verifies if the computed value matches the correct output and downloads the power measurements from the oscilloscope. This entire process is repeated until all measurements have been done. Subsequently, the acquired traces are analyzed using Riscure's Inspector software package (http://www.riscure.com/).

## 5. Side-Channel Analysis

### *5.1. Application of the TVLA Methodology to ECC*

We apply the TVLA methodology [12,13] to our implementation using the aforementioned measurement setup, following the approach from [14] (In [14], the authors apply the TVLA methodology to evaluate an implementation of ECDH-Curve25519.). Specifically, we select a set of test vectors to be used for the power measurement phase, which cover normal and special cases for the chosen implementation, as shown in Table 2. Table 3 shows the categories of special values used in Sets 4 and 5. We use a notation that is similar to [14].

**Table 2.** Sets of test vectors for test vector leakage assessment (TVLA) leakage analysis, where $k$ is the secret scalar and $P$ is the point.

| Set # | Properties | Rationale |
|---|---|---|
| 1 | constant $k$, constant $P$ | This is the baseline. The tests compare power consumption from the other sets against it. |
| 2 | constant $k$, varying $P$ | The goal is to detect systematic relationships between the power consumption and the $P$ value. |
| 3 | varying $k$, constant $P$ | The goal is to detect systematic relationships between the power consumption and the $k$ value. |
| 4 | constant k, special P | Edge cases of the algorithms used. |
| 5 | special k, constant P | Edge cases of the algorithms used. |

**Table 3.** Categories of special values for $k$, $x$ and $y$, where $x$ and $y$ are coordinates of the input point, $k$ is the scalar and $l$ is the subgroup order.

| Set # | Category # | Properties |
|---|---|---|
| 4 | 1x | $x \in \{0, 1024\}$. |
| 4 | 1y | $y \in \{0, 1024\}$. |
| 4 | 2x | $x \in \{p_{p256} - 1024, \ldots p_{p256} - 1\}$. |
| 4 | 2y | $y \in \{p_{p256} - 1024, \ldots p_{p256} - 1\}$. |
| 5 | 3 | $k \in \{0, 1024\}$. |
| 5 | 4 | $k \in \{l - 1024, \ldots l - 1\}$. |
| 4 | 5x | $x$ has a low Hamming Weight ($\leq$25). |
| 4 | 5y | $y$ has a low Hamming Weight ($\leq$25). |
| 4 | 6x | $x$ has a high Hamming Weight ($\geq$230). |
| 4 | 6y | $y$ has a high Hamming Weight ($\geq$230). |

### 5.2. Leakage Analysis

The leakage analysis of our implementation adapts the approach from [14] and is conducted in the following way. Let $\{DS_1, \ldots DS_5\}$ be the set of power traces corresponding to the selected test vectors. The full test consists of running the (pairwise) tests described in [12,13] for each of the following pairs of datasets: $\{(DS_1, DS_2), \ldots (DS_1, DS_5)\}$. If any of the previous tests fail, then the top-level test fails, and the implementation is deemed to have failed; otherwise, it is deemed to have passed the tests. We chose the confidence threshold $C = 4.5$, the same as in [12–14].

We assume that $T = |DS_1| = |DS_2| = |DS_3|$, $|DS_4| = 8T$ (because $DS_4$ corresponds to eight categories) and $|DS_5| = 2T$ (because $DS_5$ corresponds to two categories).

### 5.3. Analysis Results

#### 5.3.1. Timing Analysis

Timing analysis of the implementation (with and without coordinate randomization) was performed based on the power measurements. We have analyzed 200 measurements with different private keys. The results show that the implementations with and without coordinate randomization are constant time, with respect to the private key.

If the private key is randomized, then the execution time only differs if some of the most significant bits of the scalar are zeroes; this is as expected, since the multiplication iterations are not performed for the most significant zero bits of the scalar.

#### 5.3.2. TVLA Analysis

The leakage analysis methodology was applied to our implementation in three different settings:

- with no countermeasures applied;
- with point randomization;
- with point randomization and scalar randomization.

##### Unprotected Implementation

First, we test the implementation without any countermeasure implemented. For this test, we assume that $T = 200$. We only performed the tests for $DS_1, DS_2, DS_3$, because we detected a significant leakage already at this stage; we chose $T$ relatively low since we expected to find a significant leakage. Figure 4 shows the t-statistics for a small range of sample indices (time instants), for one run of Welch's t-test for group $A_3$ $(S_{A,1}, S_{A,3})$ of vectors selected from $DS_1$ and $DS_3$ and the same test run over a random grouping $R_3$ $(S_{R,1}, S_{R,3})$. Groups $A_j$ and $R_j$ are a partition of test vector sets $DS_i$ and $DS_j$ (where $i = 1$ and $j = 3$):

- $S_{A,i} = DS_i, S_{A,j} = DS_j$,
- $S_{R,i}$ = randomly selected subset of $DS_i \cup DS_j$ of size $T$ and
  $S_{R,j} = (DS_i \cup DS_j) \setminus S_{R,i}$.

We compute a $t$-statistics trace using the following formula:

$$\frac{|\mu_{S_{A,i}} - \mu_{S_{A,j}}|}{\sqrt{\frac{\sigma^2_{S_{A,i}}}{N_{S_{A,i}}} + \frac{\sigma^2_{S_{A,j}}}{N_{S_{A,j}}}}}, \tag{2}$$

where $\mu_x$, $\sigma_x$ and $N_x$ denote, respectively: the average of all the traces, the standard deviation and the number of traces in the partition $x$.

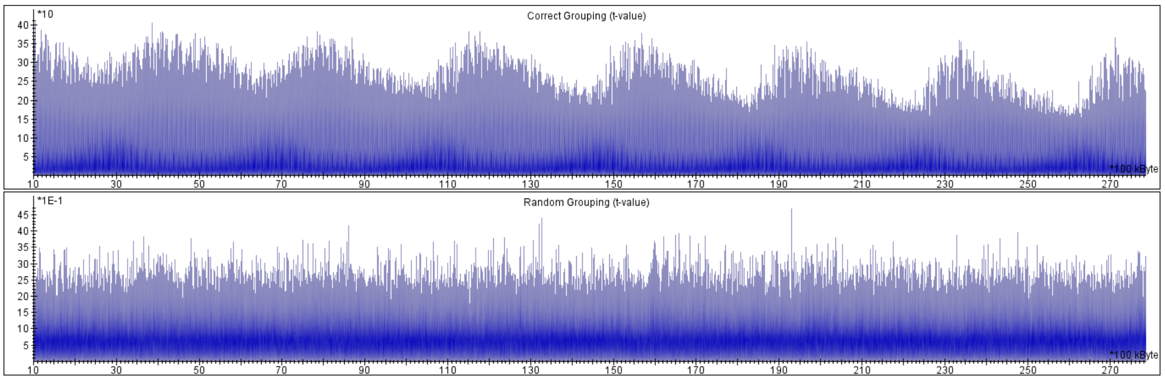

**Figure 4.** $T$-statistics versus sample index for comparing $DS_1$ and $DS_3$, for two independent groups of traces for group $A_3$ (top) and group $R_3$ (bottom).

The $t$-statistics for the group $A_3$ is way above $C = 4.5$; it even reaches 400. The $t$-statistics for the group $R_3$ rarely reaches $C = 4.5$. We do not need to consider negative values because we compute the absolute value of $t$-statistics.

We consider values around $C = 4.5$ to be ghost peaks. These results show that the implementation is vulnerable to DPA, as expected, since no countermeasure against DPA is employed.

Implementation protected with coordinate randomization:

Second, we test the implementation with the coordinate randomization enabled. This countermeasure is implemented in the following way: instead of initializing $Z$ to one, we initialize $Z$ randomly; then, we update $X$ and $Y$ by multiplying it by the new random $Z$.

We perform the TVLA analysis similarly to the unprotected implementation, but we set $T = 1000$ and perform the tests for all $DS_1, \ldots DS_5$. Note that one can argue that $T = 1000$ is a relatively low number of traces for a $t$-test. However, we need to acquire the whole execution of the scalar multiplication (i.e., 32,000,000 samples) for $13T = 13{,}000$ traces; this acquisition results in a trace set of approximately 300 gigabytes. Therefore, for the sake of efficiency, we decided not to acquire larger trace sets.

The results of the TVLA analysis are as follows:

- the $t$-statistics values for the groups $A_3$ and $A_5$ (both categories) are way above $C = 4.5$, as presented in Figure 5; the values reach 30.0 for $A3$, 9.0 for $A_5$ and Category 3 and 17.0 for $A_5$ and Category 4.
- the $t$-statistics for the group $A_2$ and the group $A_4$ are almost always less than $C = 4.5$, and they rarely reach the threshold (never significantly).

We consider the values around 4.5 to be ghost peaks, because the same levels of values are achieved in the corresponding random groupings. Additionally, we perform the analysis for two equal and disjoint parts of $A_2$ that contain $T/2$ elements each. The spikes that are above 4.5 do not occur at the same time for both parts. Therefore, we consider these spikes to be false positives.

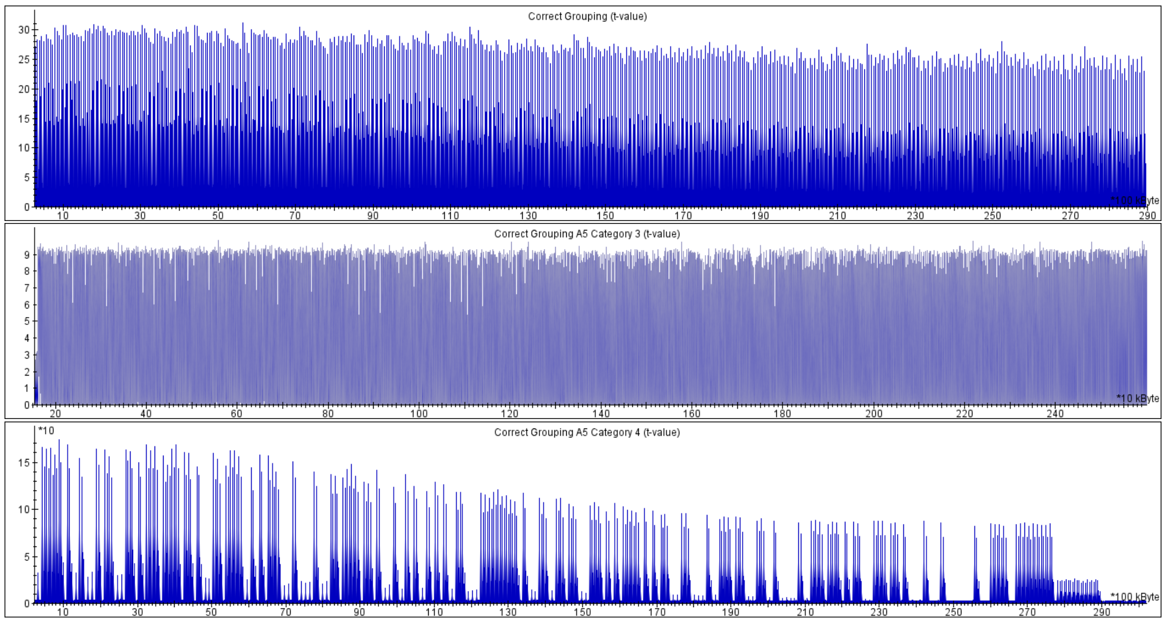

**Figure 5.** $T$-statistics versus sample index for the group $A_3$ (top), the group $A_5$ for Category 3 (center) and the group $A_5$ for Category 4 (bottom).

Based on the results presented above, we conclude that the implementation protected with coordinate randomization does not leak the intermediate point values during the scalar multiplication. However, the implementation seems to leak the key bit values; therefore, we suspect that the implementation might be susceptible to attacks similar to address-based DPA [30] or address-based template attacks [31].

Implementation protected with coordinate randomization and scalar splitting:

Since the coordinate randomization does not protect the scalar itself, we consider an additional countermeasure that hides the scalar: scalar splitting [32,33]. We consider the additive version of splitting: the scalar $k$ is split into two values $r$ and $k - r$, where $r$ is a random value of the size of $k$. Subsequently, for an input point $P$, two scalar multiplications are performed: (1) $[r]P$ and (2) $[k - r]P$. Then, the two resulting points are added to obtain $[k]P$.

Observe that for each splitting execution, the random value $r$ is chosen independently at random from the previous random choices. As a result, all scalars used in the first multiplication are independent of each other; the same holds also for the second one. Therefore, since we test for first order leakage, it is sufficient to test a single scalar multiplication $[r]P$ using TVLA.

We perform the TVLA analysis for two groups, $DS_3$ and $DS_5$, acquired during the previous analysis in the following way: we divide each group into two equal non-intersecting sets of size $T/2 = 500$ and compute the $t$-value between the sets. The $t$-statistics values for two sets from $DS_3$ are presented in Figure 6; we obtained similar results for $DS_5$. Again, we consider a single value that is slightly above 4.5 to be a ghost peak, because we achieve a similar value for a random grouping.

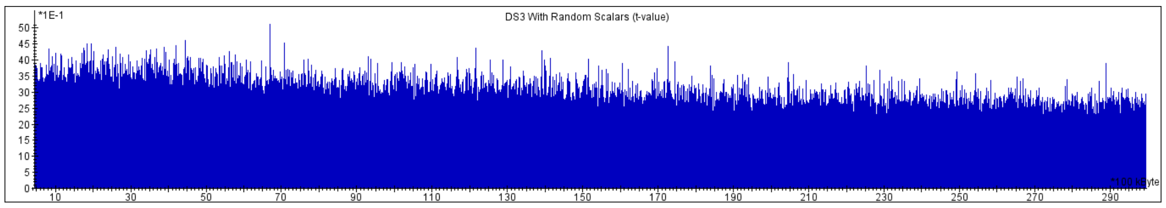

**Figure 6.** $T$-statistics versus two sample index groups coming from $DS_3$.

## 6. Results and Discussion

As expected, the unprotected implementation is time constant and resistant against SPA attacks. The implementations protected with coordinate randomization and scalar splitting are resistant against first order attacks, like SPA and DPA. Only applying the coordinate randomization as a countermeasure, however, is not enough to protect the implementation.

Observe that our analysis is aimed at detecting first order leakage. We have not evaluated the implementation against higher order attacks, like cross-correlation [34] (Observe that the scalar splitting should mitigate the cross-correlation attack.), horizontal cross-correlation [35], single trace template attacks [31] and horizontal cluster attacks [36]. We leave evaluating the implementation against these attacks as future work.

## 7. Conclusions

This paper presented the FPGA implementation and side-channel evaluation of three algorithms for ECC point multiplication, all based on the complete addition formulae introduced by Renes, Costello and Batina, with an increasing level of side-channel protection. Based on the TVLA method, the results indicate that only the implementation with all countermeasures in place is resistant against first-order DPA. Further improvements include the integration of countermeasures against higher order attacks.

**Acknowledgments:** This work was supported in part by the Technology Foundation STW (Project 13499 TYPHOON and ASPASIA), from the Dutch government. In addition, this work was supported in part by the Research Council KU Leuven (C16/15/058), CORNET project DynamIA (IWT 140389), Projects G.0130.13N and G.0876.14N of the Flemish government and by the Hercules Foundation through Project AKUL/11/19.

**Author Contributions:** For this work, the authors of KU Leuven were mainly responsible for the low-level architecture, i.e. ECC arithmetic on the FPGA, while the authors of Riscure and Radboud mainly worked on the high-level architecture (including countermeasures) and the side-channel evaluation.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209.
2. Miller, V. Use of Elliptic Curves in Cryptography. In *Advances in Cryptology—CRYPTO 85 Proceedings*; Springer: Berlin/Heidelberg, Germany, 1986; Volume 218, pp. 417–426.
3. Kocher, P.; Jaffe, J.; Jun, B. Differential Power Analysis. In *Advances in Cryptology—CRYPTO' 99*; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1666, pp. 388–397.
4. Coron, J. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems, CHES'99, Worcester, MA, USA, 12–13 August 1999; pp. 292–302.
5. Bernstein, D.J.; Lange, T. Faster Addition and Doubling on Elliptic Curves. In Proceedings of the 13th International Conference on the Theory and Application of Cryptology and Information Security, Advances in Cryptology—ASIACRYPT 2007, Kuching, Malaysia, 2–6 December 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 29–50.
6. Batina, L.; Mentens, N.; Preneel, B.; Verbauwhede, I. Side-Channel Aware Design: Algorithms and Architectures for Elliptic Curve Cryptography over GF($2^n$). In Proceedings of the 16th IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP 2005), Samos, Greece, 23–25 July 2005; pp. 350–355.
7. Renes, J.; Costello, C.; Batina, L. Complete Addition Formulas for Prime Order Elliptic Curves. In Proceedings of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Advances in Cryptology—EUROCRYPT 2016, Vienna, Austria, 8–12 May 2016.
8. Bernstein, D.J.; Lange, T. Explicit-Formulas Database. Available online: http://hyperelliptic.org/EFD/index.html (accessed on 21 February 2015).

9. Batina, L.; Chmielewski, L.; Papachristodoulou, L.; Schwabe, P.; Tunstall, M. Online Template Attacks. In Proceedings of the 15th International Conference on Cryptology in India, Progress in Cryptology—INDOCRYPT 2014, New Delhi, India, 14–17 December 2014; pp. 21–36.
10. Joye, M.; Yen, S. The Montgomery Powering Ladder. In Proceedings of the 4th International Workshop, Cryptographic Hardware and Embedded Systems—CHES 2002, Redwood Shores, CA, USA, 13–15 August 2002; pp. 291–302.
11. Goodwill, G.; Jun, B.; Jaffe, J.; Rohatgi, P. *A Testing Methodology for Side-Channel Resistance Validation*; NIST Non-Invasive Attack Testing Workshop: Nara, Japan, 2011.
12. Jaffe, J.; Rohatgi, P.; Witteman, M. Efficient Side-Channel Testing for Public Key Algorithms: RSA Case Study. Available online: http://csrc.nist.gov/news_events/non-invasive-attack-testing-workshop/papers/09_Jaffe.pdf (accessed on 22 January 2017).
13. Tunstall, M.; Goodwill, G. Applying TVLA to Public Key Cryptographic Algorithms. Cryptology ePrint Archive, Report 2016/513, 2016. Available online: http://eprint.iacr.org/2016/513 (accessed on 24 May 2016).
14. Nascimento, E.; López, J.; Dahab, R. Efficient and Secure Elliptic Curve Cryptography for 8-bit AVR Microcontrollers. In Proceedings of the 5th International Conference, Security, Privacy, and Applied Cryptography Engineering, SPACE 2015, Jaipur, India, 3–7 October 2015; Springer International Publishing: Cham, Switzerland, 2015; pp. 289–309.
15. Marzouqi, H.; Al-Qutayri, M.; Salah, K. Review of Elliptic Curve Cryptography processor designs. *Microprocess. Microsyst.* **2015**, *39*, 97–112.
16. Roy, D.B.; Das, P.; Mukhopadhyay, D. ECC on Your Fingertips: A Single Instruction Approach for Lightweight ECC Design in GF (p). In Proceedings of the 22nd International Conference, Selected Areas in Cryptography, Sackville, NB, Canada, 12–14 August 2015; Springer International Publishing: Cham, Switzerland, 2015.
17. Pöpper, C.; Mischke, O.; Güneysu, T. MicroACP—A Fast and Secure Reconfigurable Asymmetric Crypto-Processor. In *Reconfigurable Computing: Architectures, Tools, and Applications*; Springer International Publishing: Cham, Switzerland, 2014; Volume 8405, pp. 240–247.
18. Vliegen, J.; Mentens, N.; Genoe, J.; Braeken, A.; Kubera, S.; Touhafi, A.; Verbauwhede, I. A Compact FPGA-Based Architecture for Elliptic Curve Cryptography over Prime Fields. In Proceedings of the 2010 21st IEEE International Conference on Application-Specific Systems Architectures and Processors (ASAP), Rennes, France, 7–9 July 2010; pp. 313–316.
19. Alrimeih, H.; Rakhmatov, D. Fast and Flexible Hardware Support for ECC Over Multiple Standard Prime Fields. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2014**, *22*, 2661–2674.
20. Guillermin, N. A High Speed Coprocessor for Elliptic Curve Scalar Multiplications over $\mathbb{F}_p$. In Proceedings of the 12th International Workshop, Cryptographic Hardware and Embedded Systems, CHES 2010, Santa Barbara, CA, USA, 17–20 August 2010; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6225, pp. 48–64.
21. Esmaeildoust, M.; Schinianakis, D.; Javashi, H.; Stouraitis, T.; Navi, K. Efficient RNS Implementation of Elliptic Curve Point Multiplication Over GF(p). *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2013**, *21*, 1545–1549.
22. Sasdrich, P.; Güneysu, T. Efficient Elliptic-Curve Cryptography Using Curve25519 on Reconfigurable Devices. In *Reconfigurable Computing: Architectures, Tools, and Applications*; Springer International Publishing: Cham, Switzerland, 2014; Volume 8405, pp. 25–36.
23. Baldwin, B.; Moloney, R.; Byrne, A.; McGuire, G.; Marnane, W.P. A Hardware Analysis of Twisted Edwards Curves for an Elliptic Curve Cryptosystem. In *Reconfigurable Computing: Architectures, Tools and Applications*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5453, pp. 355–361.
24. Järvinen, K.; Miele, A.; Azarderakhsh, R.; Longa, P. Four $\mathbb{Q}$ on FPGA: New Hardware Speed Records for Elliptic Curve Cryptography over Large Prime Characteristic Fields. In Proceedings of the 18th International Conference on Cryptographic Hardware and Embedded Systems—CHES 2016, Santa Barbara, CA, USA, 17–19 August 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 517–537.
25. Ghosh, S.; Mukhopadhyay, D.; Roychowdhury, D. Petrel: Power and Timing Attack Resistant Elliptic Curve Scalar Multiplier Based on Programmable GF(p) Arithmetic Unit. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2011**, *58*, 1798–1812.

26. Montgomery, P.L. Modular Multiplication without Trial Division. *Math. Comput.* **1985**, *44*, 519–521.
27. Walter, C.D. Montgomery Exponentiation Needs No Final Subtraction. *Electron. Lett.* **1999**, *35*, 1831–1832.
28. Montgomery, P.L. Speeding the Pollard and Elliptic Curve Methods of Factorization. *Math. Comput.* **1987**, *48*, 243–264.
29. Guntur, H.; Ishii, J.; Satoh, A. Side-channel Attack User Reference Architecture Board SAKURA-G. In Proceedings of the 2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE), Tokyo, Japan, 7–10 October 2014; pp. 271–274.
30. Itoh, K.; Izu, T.; Takenaka, M. A Practical Countermeasure against Address-Bit Differential Power Analysis. In *Cryptographic Hardware and Embedded Systems—CHES 2003*; Walter, C.D., Koç, Ç.K., Paar, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2779, pp. 382–396.
31. Nascimento, E.; Chmielewski, Ł.; Oswald, D.; Schwabe, P. Attacking Embedded ECC Implementations through cmov Side Channels. In *Selected Areas in Cryptology—SAC 2016*; Avanzi, R., Heys, H., Eds.; Springer: Berlin/Heidelberg, Germany, 2016.
32. Chari, S.; Jutla, C.S.; Rao, J.R.; Rohatgi, P., Towards Sound Approaches to Counteract Power-Analysis Attacks. In Proceedings of the 19th Annual International Cryptology Conference, Advances in Cryptology—CRYPTO' 99, Santa Barbara, CA, USA, 15–19 August 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 398–412.
33. Clavier, C.; Joye, M. Universal Exponentiation Algorithm A First Step towards Provable SPA-Resistance. In Proceedings of the Third International Workshop on Cryptographic Hardware and Embedded Systems—CHES 2001, Paris, France, 14–16 May 2001; Springer: Berlin/Heidelberg, Germany, 2001; pp. 300–308.
34. Witteman, M.; van Woudenberg, J.; Menarini, F. Defeating RSA Multiply-Always and Message Blinding Countermeasures. In Proceedings of the Topics in Cryptology—CT-RSA 2011, San Francisco, CA, USA, 14–18 February 2011; Kiayias, A., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6558, pp. 77–88.
35. Hanley, N.; Kim, H.; Tunstall, M. Exploiting Collisions in Addition Chain-Based Exponentiation Algorithms Using a Single Trace. In Proceedings of the Topics in Cryptology–CT-RSA 2015, San Francisco, CA, USA, 20–24 April 2015; Nyberg, K., Ed.; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9048, pp. 431–448.
36. Perin, G.; Chmielewski, Ł. A Semi-Parametric Approach for Side-Channel Attacks on Protected RSA Implementations. In *Smart Card Research and Advanced Application*; Homma, N., Medwed, M., Eds.; Springer: Cham, Switzerland, 2015; Volume 9514, pp. 102–114.