Improving the Randomized Initial Point Countermeasure Against DPA

Kouichi Itoh, Tetsuya Izu, and Masahiko Takenaka

FUJITSU LABORATORIES Ltd.
4-1-1, Kamikodanaka, Nakahara-ku, Kawasaki, 211-8588, Japan
{kito, izu, takenaka}@labs.fujitsu.com

Abstract. DPA-countermeasures are one of the essential technology for implementing elliptic curve cryptosystems (ECC) on smart cards. Not only standard DPA but also recently proposed refined power analysis (RPA) and zero value analysis (ZVA) should be considered. Itoh, Izu and Takenaka proposed a secure and efficient countermeasure (the randomized initial point countermeasure, RIP) in order to resist these attacks. Then, Mamiya, Miyaji and Morimoto improved the efficiency. This paper also aims at improving RIP in another direction. As a result, compared to the original RIP, about 28% improvement can be established. In other words, the proposed countermeasure has almost no penalty from a non DPA-resistant scalar multiplication.

Keywords: Smart card, Elliptic Curve Cryptosystems (ECC), DPA, RPA, ZVA, countermeasure, RIP.

1 Introduction

Smart cards are a new infrastructure in the coming ubiquitous society because of its plenty of applications such as SIM card, ID card, and driving licence. However, side channel attacks are real threats for such applications. When a cryptographic procedure is computed in a smart card with a secret key hidden in the device, the card leaks side channel information such as power consumption. In the side channel attacks, an adversary analyzes the information and tries to detect the secret key. The attacks will be successful if there is a connection between the information and the secret. Among these attacks, the differential power analysis (DPA) [KJJ99, MDS99] is a very strong attack, in which the adversary statistically analyzes the side channel information from some thousands of observations. DPA-countermeasures are essential for smart cards.

Elliptic curve cryptosystems (ECC) are considered as a suitable choice for smart card applications because they achieve high security with shorter key compared to other public-key cryptosystems such as RSA or ElGamal. In 2003, a new variant of DPA, the refined power analysis (RPA), was proposed by Goubin [Gou03]. Soon it is extended to the zero value analysis (ZVA) by Akishita and Takagi [AT03]. Thus ECC requires not only secure against these DPA-attacks but efficient countermeasures. In CARDIS 2004, Itoh, Izu and Takenaka proposed

J. Zhou, M. Yung, and F. Bao (Eds.): ACNS 2006, LNCS 3989, pp. 459–469, 2006.

an efficient countermeasure, the randomized initial point countermeasure (RIP) [IIT04]. Then, in the same year, Mamiya, Miyaji and Morimoto applied RIP to the binary method from the most significant bit (MSB) and window-based methods, and improved the efficiency significantly [MMM04]. However, this approach requires a look-up table. Generally, there is a time-memory trade-off between efficiency of scalar multiplications and available memory. Although recent smart cards have enough memory, it is desirable to compute scalar multiplications with less memory in an efficient way.

This paper is also aimed at applying RIP to the binary method from MSB in a different way, namely by checking effects of changing initial points in some algorithms for scalar multiplications. Especially, we apply RIP to the Montgomery Ladder, a variant of the binary method from MSB, which is known as an efficient algorithm when it is combined to the x-coordinate-only addition formula [IT02], about 28% improvement compared to the original RIP, or reduction of the efficiency or the number of registers can be established.

A rest of the paper is organized as follows: we briefly introduce elliptic curve cryptosystems (ECC) and side channel attacks for ECC in section 2. Then, in section 3, we propose the extended RIP as a DPA-countermeasure. Section 4 compares some secure countermeasures in detail.

2 Preliminaries

This section briefly introduces elliptic curve cryptosystems (ECC). Power analysis attacks and countermeasures for ECC are also reviewed.

2.1 Elliptic Curve Cryptosystems

In this paper, we assume that the characteristic of a definition finite field K is greater than 3 (however, most countermeasures can be applied to finite fields with characteristics 2 or 3).

Elliptic Curve. An elliptic curve over a definition field K is given by an equation

$$E : y^2 = x^3 + ax + b \ (a, b \in K,\ 4a^3 + 27b^2 \neq 0).$$

A set of K-rational points on E is defined by

$$E(K) = \{(x, y) \in E \mid x, y \in K\} \cup \{\mathcal{O}\},$$

where the special point $\mathcal{O}$ is called the *point at infinity*, which is the only point that can not be represented as a pair of two K-elements like (x, y) (conversely, other points can be represented as a pair of K-elements). In this paper, we identify an elliptic curve E and its K-rational point $E(K)$ for simplicity.

Standard Addition Formula. An elliptic curve $E(K)$ has an additive group structure by the following rules: a neutral element of the group is the point at infinity $\mathcal{O}$. Inversion of the neutral point is itself ($-\mathcal{O} = \mathcal{O}$). For a point

$P = (x, y) \in E(K)\backslash\{\mathcal{O}\}$, inversion is defined by $-P = (x, -y) \in E(K)\backslash\{\mathcal{O}\}$. For the point at infinity $\mathcal{O}$ and an arbitrary point $P \in E(K)$, we define their addition as $\mathcal{O} + P = P + \mathcal{O} = P$. We also define $P + (-P) = \mathcal{O}$. If two arbitrary points $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2) \in E(K)\backslash\{\mathcal{O}\}$, $P_1 \neq -P_2$ are given, their addition $P_3 = P_1 + P_2 = (x_3, y_3)$ is defined by

$$x_3 = \lambda^2 - x_1 - x_2,\ y_3 = \lambda(x_1 - x_3) - y_1, \tag{1}$$

where

$$\lambda = \begin{cases} (y_2 - y_1)/(x_2 - x_1) & \text{if } P_1 \neq P_2 \\ (3x_1^2 + a)/(2y_1) & \text{if } P_1 = P_2. \end{cases}$$

The above formula is called the *addition formula*, an addition $P_1 + P_2$ ($P_1 \neq P_2$) is called an *elliptic curve addition* (ECADD), and a doubling $P_1 + P_1 = 2P_1$ is called an *elliptic curve doubling* (ECDBL). Note that ECADD and ECDBL are computed by a sequence of fundamental operations in the definition field K such as additions, subtractions, multiplications and divisions. However, the sequences differs for ECADD and ECDBL.

***x*-coordinate-only Addition Formula.** In the standard addition formula (1), both x_3, y_3 are functions on x_1, x_2, y_1 and y_2. The x-coordinate-only addition formula, in which x_3 is computed from x-coordinates, are as follows [Mon87, BJ02, IT02], where $P_3 = (x_3, y_3)$, $P_3' = P_1 - P_2 = (x_3', y_3')$, $P_4 = 2P_1 = (x_4, y_4)$:

$$x_3 = \frac{2(x_1 + x_2)(x_1x_2 + a) + 4b}{(x_1 - x_2)^2} - x_3',$$
$$x_3 = \frac{1}{x_3'}\frac{(x_1x_2 - a)^2 - 4b(x_1 + x_2)}{(x_1 - x_2)^2},$$
$$x_4 = \frac{(x_1^2 - a)^2 - 8bx_1}{4(x_1^3 + ax_1 + b)}.$$

We denote an elliptic addition (doubling) by the x-coordinate-only formula as xECADD (xECDBL), respectively. Note that, x_3' is required to compute xECADD. Also note that there is 2 formulas for xECADD; we may distinguish them by using $\text{xECADD}^{\text{add}}$ (additive xECADD) and $\text{xECADD}^{\text{mul}}$ (multiplicative xECADD).

Scalar Multiplication. When a point $P \in E(K)$ and a scalar $d \in \mathbb{Z}_{>0}$ are given, a *scalar multiplication* of P by d is to compute $dP = P + \cdots + P$. Here, P is called a *base point*. Computing dP from d, P is easy (as described later), while computing d from dP, P is known to be hard in general. This problem is called the *elliptic curve discrete logarithm problem* (ECDLP) and no efficient algorithms are known. In ECC, a base point P and a scalar multiplied point dP are public, while the scalar d is secret. The hardness of ECDLP assures the security of ECC.

Algorithm 1. Montgomery Ladder

```
INPUT: d, P
OUTPUT: d*P
 1: T[0] = P, T[1] = 2*P
 2: for i=n-2 downto 0 {
 3:    T[2] = 2*T[d[i]]
 4:    T[1] = T[0]+T[1]
 5:    T[0] = T[2-d[i]]
 6:    T[1] = T[1+d[i]]
 7:    }
 8: return T[0]
```

Algorithm 2. ADA(from LSB)

```
INPUT: d, P
OUTPUT: d*P
 1: T[0] = 0, T[2] = P
 2: for i=0 upto n-1 {
 3:    T[1] = T[0]+T[2]
 4:    T[2] = 2*T[2]
 5:    T[0] = T[d[i]]
 6: }
 7: return T[0]
```

Since a scalar multiplication is the most time consuming part in ECC, various techniques for speeding-up has been proposed. On the other hand, this computation is a main target of side channel attacks .

Addition Chain. Let $d = d_{n-1}2^{n-1} + \cdots + d_1 2^1 + d_0$ $(d_i \in \{0,1\}, d_{n-1} = 1)$ be a binary representation of an n-bit scalar d. Then the binary method from the least significant bit (LSB), the binary method from the most significant bit (MSB), and the Montgomery Ladder (Algorithm 1) computes scalar multiplications efficiently. Standard addition formula can be combined with all of the methods, while the x-coordinate-only formula can be combined only to the Montgomery Ladder.

Note that some algorithms use precomputed tables in order to improve the efficiency. Since such methods require rather a large amount of registers and are expensive for low-end smart cards, we do not consider such strategies in the followings.

Coordinate System. In the previous section, an elliptic curve is given by the affine coordinate system, in which addition formulas require inversions in K. In most environments (especially, in our supposed environments), computing inversions is sometimes a hard task. Thus following coordinate systems are widely used to avoid inversions. In the projective coordinate system, a point on an elliptic curve is represented as a tuple of K-elements (X, Y, Z), and two points (X, Y, Z) and $(\lambda X, \lambda Y, \lambda Z)$ $(\lambda \in K \backslash \{0\})$ are identified. The elliptic curve equation is obtained by substituting $x = X/Z,\ y = Y/Z$ into the affine equation. The point at infinity is represented as points whose Z-coordinate value being 0.

Table 1. Computing amounts of ECADD/ECDBL

	Coordinate System	ECADD		ECDBL
		$Z \neq 1$	$Z = 1$	$a \neq -3$
Standard Formula	$\mathcal{P}$	$12M + 2S$	$9M + 2S$	$7M + 5S$
	$\mathcal{J}$	$12M + 4S$	$8M + 3S$	$4M + 6S$
x-coordinate Formula	$\mathcal{P}$	$8M + 2S$	$8M + 3S$	$6M + 3S$
			$13M + 4S$	

The Jacobian coordinate system is also used. However, a detailed description is omitted here (see [CMO98]).

Table 1 summarizes amounts of computations of ECADD and ECDBL in the projective coordinate system $\mathcal{P}$ and the Jacobian system $\mathcal{J}$, where M, S, I denote the computing times of a multiplication, a squaring and an inversion in the definition field K, respectively. The last row of Table 1 describes amounts of computations of a merged function xECADDDBL, which computes xECADD and xECDBL together [IT02].

2.2 Power Analysis

Power analysis attack is a powerful side channel attack in which power traces of smart cards are observed and analyzed. The following SPA and DPA are typical examples of the power analysis. For elliptic curve cryptosystems, the following RPA and ZVA, variants of DPA, should be also considered.

Simple Power Analysis (SPA). Binary methods compute an ECADD when d_i, the i-th bit of d, equals to 1. Since power traces of ECADD and ECDBL have different patterns, an adversary easily detect the corresponding value of d_i by analyzing power traces. This is a main idea of the simple power analysis (SPA) proposed by Kocher [Koc96], which can be applied to other exponentiation-based cryptosystems such as RSA.

The simplest way to resist SPA is the add-and-double-always method (ADA) proposed by Coron [Cor99], in which an ECADD and an ECDBL are computed for every bit independent from a value d_i (Algorithm 2, 3). Since power traces of these algorithm become fixed, the adversary cannot detect d_i any more. However, the efficiency of a scalar multiplication is reduced because of dummy operations. On the other hand, the Montgomery Ladder substantially resists SPA since it computes ECADD and ECDBL for each bit.

Differential Power Analysis (DPA). Patterns of power traces depend on not only operations but operands. Assume an adversary is able to simulate the target computation and obtain arbitrary intermediate status. By assuming $d_i = 1$ (for example), the adversary can collect simulated power traces and classify them into two groups depending on the hamming weight of intermediate values. If the assumptions is correct, there appears some differences between these power traces, and the adversary can confirm the correctness of his/her assumption. This is a basic strategy of the differential power analysis (DPA) [KJJ99, MDS99]. DPA can be applied to ECC and RSA.

Refined Power Analysis (RPA) and Zero Value Analysis (ZVA). Goubin enhanced DPA to the refined power analysis (RPA) applicable to ECC only [Gou03]. Points with 0-coordinate values are called the *special points*. These points are easily observed by DPA since power traces with regard to the special points are so characteristic. In RPA, an adversary adaptively chooses the base point and tries to detect such characteristic patterns in power traces. RPA can be applied to all of described addition chains.

Algorithm 3. ADA (from MSB)

```
INPUT: d, P
OUTPUT: d*P
 1: T[0] = 0, T[1] = P
 2: for i=n-1 downto 0
 3:    T[0] = 2*T[0]
 4:    T[1] = T[0]+T[2]
 5:    T[0] = T[d[i]]
 6: }
 7: return T[0]
```

Algorithm 4. MMM method

```
INPUT: d, P
OUTPUT: d*P
 1: T[0] = randompoint()
 2: T[1] = -T[0]
 3: T[2] = P+T[1]
 4: for i=n-1 downto 0 {
 5:    T[0] = 2*T[0]
 6:    T[0] = T[0]+T[1+d[i]]
 7: }
 8: return T[0]
```

Akishita and Takagi extended RPA to the zero value analysis (ZVA) [AT03], in which an adversary detects 0-values in ECADD and ECDBL, rather than 0-values in intermediate points. ZVA is also applicable to ECC only.

Countermeasures. Since SPA and DPA are independent attacks, the Montgomery Ladder (Algorithm 1) or add-and-double-always methods (Algorithm 2,3) are required to resist SPA. We assume to use these algorithms in the followings.

In order to resist DPA, RPA and ZVA, randomization is a common technique [Cor99]. Clavier and Joye split a scalar randomly, in which dP is computed by $dP = rP + (d - r)P$ for a random integer r (exponent splitting, ES) [CJ01]. ES requires at least twice amount of computation than without it. Ciet and Joye proposed another splitting method, in which dP is computed by $\lfloor d/r \rfloor (rP) + (d \bmod r)P$ for a random integer r (improved exponent splitting, iES) [CJ03]. The size of r can be as long as half of the size of d, and, with the Shamir's trick, it provides an efficient computation without reducing the security.

On the other hand, the randomized linearly-transformed coordinates countermeasure (RLC) [IIT04] is an extension of Coron's randomized projective coordinates countermeasure (RPC) [Cor99]. While RPC is vulnerable to RPA/ZVA, RLC resists all of DPA/RPA/ZVA. Itoh et al. also proposed the randomized initial point countermeasure (RIP) in the same paper [IIT04]. Since RIP is a main topic of this paper, we proceed to the next section.

3 Randomized Initial Point Countermeasure (RIP)

The randomized initial point countermeasure (RIP), which is a main topic of this paper, is a DPA/RPA/ZVA-countermeasure proposed by Itoh et al. [IIT04]. When a scalar multiplication is computed by the binary method from LSB, or the add-and-double-always method from LSB, compute $R + dP$ for a randomly generated point R in the addition chain and output dP by subtracting R from the above result. RIP is very similar to Coron's 2nd countermeasure [Cor99] which computes a scalar multiplication dP by $d(P + Q) - R$ for $R = dQ$. Since R should be equal to dQ, R is not a random point. In fact, Okeya and Sakurai

showed the security problem of the Coron's 2nd countermeasure [OS00]. In a sense, RIP is an extension of the Coron's 2nd countermeasure.

In the proposal paper, Itoh et al. claimed that RIP can be combined only to LSB methods [IIT04] (in spite of the naming). However, a principle of RIP, namely

> Add a random point to initial points in the beginning. After finishing a main scalar multiplication, exclude an unnecessary point from the result

is not dependent from addition chains. Thus, in the followings, we discuss the possibility of the application of this principle to add-and-double-always method from LSB (Algorithm 2), add-and-double-always method from MSB (Algorithm 3), and the Montgomery Ladder (Algorithm 1).

3.1 Binary Methods from LSB

Algorithm 2 uses 3 registers `T[0]`, `T[1]`, `T[2]`. When an initial point of a register `T[0]` is replaced from $\mathcal{O}$ to an arbitrary (random) point R, since the algorithm outputs $R+dP$, desired dP is obtained by subtracting R from the above output. This is the original RIP proposed by Itoh et al. [IIT04]. We denote this algorithm as RIP(LSB,0). For generating a random point R, Itoh et al. proposed some concrete algorithms. For example, R can be kept inside of a smart card (as a global register).

Since a register `T[1]` is initialized in step 3, changing an initial point has no effect on a scalar multiplication.

On the other hand, when an initial point of a register `T[2]` is replaced from P to $P + R$, (and if an initial point of `T[0]` is not changed) since the algorithm outputs $d(P + R)$, desired dP is contained by subtracting dR from the above output. This is the randomized base-point countermeasure by Coron [Cor99] (RIP(LSB,2)), however, there requires to compute dR in turn. In addition, a security problem is pointed out in [OS00].

It is possible to replace initial points of registers `T[0]`, `T[2]` at the same time (RIP(LSB,0+2)). However, the efficiency is not competitive since it requires another scalar multiplication.

3.2 Binary Methods from MSB

Algorithm 3 uses 3 registers `T[0]`, `T[1]`, `T[2]`. When an initial point of a variable `T[0]` is replaced from $\mathcal{O}$ to a random point R (and if an initial point of a register `T[2]` is not changed), the algorithm outputs $dP + 2^nR$ (RIP(MSB,0)). By subtracting 2^nR from the output, we obtain a desired dP. If we update R by $R \leftarrow 2R$ after step 5, 2^nR is obtained when step 6 finishes. Or keeping a pair $(R, 2^nR)$ inside a smart card would be another solution. Since 2^n is independent from the secret d, there are no leakage of d in the computation of 2^nR.

Since a register `T[1]` is initialized in step 4, changing an initial point has no effect on a scalar multiplication.

On the other hand, when an initial point of a register `T[2]` is replaced from P to $P + R$ (and if an initial point of a register `T[0]` is not changed), the algorithm outputs $d(P + R)$ (RIP(MSB,2)). This is again identical to Coron's countermeasure. Similar to the LSB case, it is possible to replace initial points of registers `T[0]`, `T[2]` at the same time (RIP(MSB,0+2)).

Let us denote -1 as $\overline{1}$. Using a fact that an arbitrary point R is represented as $R = (1\overline{1}\ldots\overline{1})_2 R$, Mamiya et al. proposed a countermeasure (Algorithm 4, RIP(MSB,MMM)) [MMM04], which is a variant of RIP(MSB,0+2). Here a function `randompoint()` generates a random point on an elliptic curve. See [IIT04] for concrete constructions.

3.3 Montgomery Ladder

Algorithm 1 uses 3 registers `T[0]`, `T[1]`, `T[2]`. Since a register `T[2]` is initialized in step 3, changing an initial point has no effect on a scalar multiplication.

When an initial point of one of registers `T[0], T[1]` is added by a random point R, Algorithm 1 outputs $dP + d'R$ (RIP(Mon,0), RIP(Mon,1)). Then computing $d'P$ is required in turn. Since a scalar d' is dependent on d, computing $d'R$ might introduce the leakage of the secret d. This strategy is not suitable as a countermeasure.

On the other hand, when initial points of both of `T[0]`, `T[1]` are added by a random point R, Algorithm 1 outputs $dP + 2^{n-1}R$ (RIP(Mon,0+1)). Here computing $2^{n-1}R$ is required again, but it is obtained easily similar to RIP(MSB,0). Since 2^{n-1} is independent form the secret d, there are no leakage of d in the computation of $2^{n-1}R$.

Strongly note that in step 6 of the Montgomery Ladder (Algorithm 1), a difference of two registers `T[0]` and `T[1]` is kept always to be P. Therefore, the x-coordinate-only method can be applied to the algorithm and the efficiency will be considerably improved.

4 Comparison

In this section, we compare amounts of computation for a scalar multiplication and of intermediate registers of the exponent splitting countermeasure (ES) [CJ01], the improved exponent splitting countermeasure (iES) [CJ03], the randomized linearly-transformed coordinates countermeasure (RLC) [IIT04] and the randomized initial point countermeasure (RIP) [IIT04] discussed in this paper. Note that all of above algorithms resist all of DPA, RPA, ZVA. A comparison is summarized in Table 2.

Assumption. In each countermeasure, add-and-double-always method (including the Montgomery Ladder) is used as an SPA-countermeasure. The projective or Jacobian coordinates is used. We assumed $a \neq 3$ and $1S = 0.8M$ in the estimation, here M, S denotes computing times for a multiplication and a squaring in the definition field. In addition, we assumed that pre-computations and post-computations of a scalar multiplication are negligible. Especially, we assumed

Table 2. A comparison of SPA/DPA/RPA/ZVA/ADPA-resistant countermeasures

Countermeasure	# of global reg.	# of local reg.	Computing time per bit
ES + ADA (from MSB)	–	5	$24M + 18S$ $(38.4M)$
iES + ADA (from MSB) + Shamir's trick	–	4	$12M + 9S$ $(19.2M)$
RLC + ADA (from MSB) + RA	–	3	$20M + 10S$ $(28.0M)$
RLC + ADA (from LSB) + RA	–	4	$17M + 10S$ $(25.0M)$
RLC + ADA (from LSB) + iECDBL + RA	–	4	$17M + 8S$ $(23.4M)$
RIP + ADA (from MSB) + RA	0/1	5/4	$16M + 10S$ $(24.0M)$
RIP + ADA (from MSB) + SSM($t = 2$) + RA	0/1	7/6	$10.3M + 8.1S$ $(16.8M)$
RIP + ADA (from MSB) + SSM($t = 3$) + RA	0/1	11/10	$8.7M + 7.6S$ $(14.7M)$
RIP + ADA (from MSB) + SSM($t = 4$) + RA	0/1	19/18	$8.2M + 7.4S$ $(14.1M)$
RIP(LSB,0) + RA	0/1	4/3	$16M + 10S$ $(24.0M)$
RIP(LSB,0) + iECDBL + RA	0/1	–[1]	$16M + 8S$ $(22.4M)$
RIP(LSB,0+2) + RA	0/1	5/4	$32M + 20S$ $(48.0M)$
RIP(MSB,0) + RA	0/1	4/3	$16M + 15S$ $(28.0M)$
	2	3	$12M + 9S$ $(19.2M)$
RIP(MSB,0+2) + RA	0/1	5/4	$32M + 30S$ $(56.0M)$
	2	4	$24M + 18S$ $(38.4M)$
RIP(Mon,0+1) + RA	0/1	4/3	$13M + 4S$ $(16.2M)$
	2	3	$13M + 4S$ $(16.2M)$

a processing time for the function `randompoint()`. We used the estimation in [IIT04] for ES, iES, RLC, ADA (from MSB)+RIP. Here RIP (from MSB) is identical to a countermeasure proposed by Mamiya et al. [MMM04] and SSM denotes a simultaneous scalar multiplication.

Address-bit DPA. In addition to SPA/DPA/RPA/ZVA, we also consider the address-bit DPA (ADPA) proposed by Itoh et al. [IIT02], since RLC and RIP are vulnerable to this attack. However, the randomized addressing countermeasure (RA) [IIT03] is an efficient countermeasure for these algorithms.

The Number of Intermediate Registers. For a random point generation in RIP, we used methods described in [IIT04]. Since the number of intermediate (global or local) registers vary from the generating algorithms, we list up the possible values. Here we assumed that 1 register holds coordinate for 1 point.

In Table 2, registers outside a scalar multiplication are called as the *global*, registers inside are as *local*. For example, for RIP(LSB,0), there is a description 0/1 for global registers and 4/3 for local registers. This means that 0 global registers are required if 4 local registers are used, and 1 global register is required if 3 local registers are used.

Discussion. As in Table 2, the most efficient countermeasure is a combination of RIP (from MSB) and a simultaneous scalar multiplication (ADA(from

[1] Since iterated ECDBL (iECDBL) computes ECDBLs in the addition chain layer, rather than as the function, the number of (apparently) required registers grows larger. Thus we omit the number because this situation is quite different from others.

MSB)+RIP +SSM($t = 4$)+RA). On the other hand, when we consider the number of local registers, a combination of RIP and the Montgomery Ladder (RIP(Mon,0+1)+RA) provides relatively efficient countermeasure. Especially, this method establishes about 28% improvement from the original RIP countermeasure [IIT04]. Thus this countermeasures can be regarded as a leading alternative for low-end smart card applications.

5 Concluding Remarks

This paper discussed the randomized initial point method (RIP) as a DPA/RPA/ZVA-countermeasure, and propose some countermeasures. Especially, a combination of RIP and the Montgomery Ladder establishes about 28% improvement on the original countermeasure.

It is taken for granted that there is no choice to sacrifice the efficiency for implementing power analysis countermeasures. However, RIP countermeasure has almost no penalty from scalar multiplication algorithms without such countermeasures. [2]. On this point, RIP randomizes without sacrifice the efficiency. But RIP even requires dummy operations for SPA-resistance. Reducing dummy operations without reducing the security will be the future work.

Acknowledgements

The authors would like to thank Naoya Torii and anonymous reviewers for their helpful comments.

References

[AT03] T. Akishita, T. Takagi, "Zero-value Point Attacks on Elliptic Curve Cryptosystem", *ISC 2003*, LNCS 2851, pp.218-233, Springer-Verlag, 2003.

[BJ02] E. Brier, M. Joye, "Weierstraß Elliptic Curvs and Side-Channel Attacks", *PKC 2002*, LNCS 2274, pp.335-345, Springer-Verlag, 2002.

[Cor99] J. Coron, "Resistance against differential power analysis for elliptic curve cryptosystem", *CHES'99*, LNCS 1717, pp.292-302, Springer-Verlag, 1999.

[CJ01] C. Clavier, M. Joye, "Universal exponentiation algorithm – A first step towards provable SPA-resistance –", *CHES 2001*, LNCS 2162, pp. 300-308, Springer-Verlag, 2001.

[CJ03] M. Ciet, M. Joye, "(Virtually) Free Randomization Technique for Elliptic Curve Cryptography", *ICICS 2003*, LNCS 2836, pp. 348-359, Springer-Verlag, 2003.

[CMO98] H. Cohen, A. Miyaji, T. Ono, "Efficient Elliptic Curve Exponentiation using Mixed Coordinates", *Asiacrypt'98*, LNCS 1514, pp.51-65, Springer-Verlag, 1998.

[Gou03] L. Goubin, "A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems", *PKC 2003*, LNCS 2567, pp.199-210, Springer-Verlag, 2003.

[2] Of course some overheads are required for pre-computations and post-computations. However, these computations can be negligible from a total computation.

[IIT02] K. Itoh, T. Izu, M. Takenaka, "Address-bit Differential Power Analysis of Cryptographic Schemes OK-ECDH and OK-ECDSA", *CHES 2002*, LNCS 2523, pp.129-143, Springer-Verlag, 2003.

[IIT03] K. Itoh, T. Izu, M. Takenaka, "A Practical Countermeasure against Address-bit Differential Power Analysis", *CHES 2003*, LNCS 2779, pp. 382-396, Springer-Verlag, 2003.

[IIT04] K. Itoh, T. Izu, M. Takenaka, "Efficient Countermeasure against Power Analysis for Elliptic Curve Cryptosystems", *CARDIS 2004*, pp.99-114, Kluwer, 2004.

[IT02] T. Izu and T. Takagi, "A Fast Parallel Elliptic Curve Multiplication Resisntant against Elliptic Curve Cryptosystems", *PKC 2002*, LNCS 2274, pp.280-296, Springer-Verlag, 2002.

[Koc96] C. Kocher, "Timing attacks on Implementations of Diffie-Hellman, RSA, DSS, and other systems", *Crypto'96*, LNCS 1109, pp.104-113, Springer-Verlag, 1996.

[KJJ99] C. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis", *Crypto'99*, LNCS 1666, pp.388-397, Springer-Verlag, 1999.

[Mon87] P. Montgomery, "Speeding the Pollard and Elliptic Curve Methods for Factorizations", *Mathematics of Computation*, vol.48, pp.243-264, 1987.

[MDS99] T. Messerges, E. Dabbish, R. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards", *CHES'99*, LNCS 1717, pp. 144-157, Springer-Verlag, 1999.

[MMM04] H. Mamiya, A. Miyaji, H. Morimoto, "Efficient Countermeasures against RPA, DPA, and SPA", *CHES 2004*, LNCS 3156, pp. 343-356, Springer-Verlag, 2004.

[OS00] K. Okeya, K. Sakurai, "Power Analysis Breaks Elliptic Curve Cryptosystems even Secure Against the Timing Attack", *Indocrypt 2000*, LNCS 1977, pp.178-190, Springer-Verlag, 2000.