A Practical Countermeasure against Address-Bit Differential Power Analysis

Kouichi Itoh[1], Tetsuya Izu[2], and Masahiko Takenaka[1]

[1] FUJITSU LABORATORIES Ltd.,
64, Nishiwaki, Okubo-cho, Akashi, 674-8555, Japan
[2] FUJITSU LABORATORIES Ltd.,
4-1-1, Kamikodanaka, Nakahara-ku, Kawasaki, 211-8588, Japan
{kito,izu,takenaka}@labs.fujitsu.com

Abstract. The differential power analysis (DPA) enables an adversary to reveal the secret key hidden in a smart card by observing power consumption. The address-bit DPA is a typical example of DPA which analyzes a correlation between addresses of registers and power consumption. In this paper, we propose a practical countermeasure, the *randomized addressing countermeasure*, against the address-bit DPA which can be applied to the exponentiation part in RSA or ECC with and without pre-computed table. Our countermeasure has almost no overhead for the protection, namely the processing speed is no slower than that without the countermeasure. We also report experimental results of the countermeasure in order to show its effect. Finally, a complete comparison of countermeasures from various view points including the processing speed and the security level is given.

Keywords. Differential Power Analysis (DPA), address-bit DPA, countermeasure, exponentiation, RSA, ECC

1 Introduction

Smart cards are becoming a new infrastructure of the coming IT society for their plenty of attractive applications such as identification cards, telephone cards and electronic tickets. However, the side channel attacks are real threats for them [19,20]. In the attack, an adversary observes side channel information such as computing time and power consumption. The adversary can obtain the secret information if there is a tight relation between the side channel information and the secret information (secret key) hidden in the smart card. Especially, if there is a irregular procedure (short-cuts) in the computation, the adversary can easily detect it. Thus the adversary could reveal the secret key without tampering the device physically. The simple power analysis (SPA) and the differential power analysis (DPA) are typical examples of the side channel attacks. Implementers of cryptographic schemes should take countermeasures against these attacks.

In 1999, Messerges et al. proposed a new powerful attack against the secret key cryptosystems, the *address-bit DPA* (ADPA) (from now on, we call the previous DPA as the *data-bit DPA* (DDPA)), which analyzes a correlation between

C.D. Walter et al. (Eds.): CHES 2003, LNCS 2779, pp. 382–396, 2003.

the secret information and addresses of registers [26]. Then, in 2002, Itoh et al. extended the attack to Elliptic Curve based Cryptosystems (ECC) [11]. These results suggest that implementers should consider the correlation of the secret information to not only data of registers but also addresses of registers. Itoh et al. also gave several countermeasures against the attack, but those countermeasures require at least twice computing time than without them.

In this paper, we propose a practical countermeasure against the address-bit DPA by randomizing addresses of registers for ECC and RSA. Our countermeasure does not change the scalar to be multiplied; an overhead is very small and the processing speed is as fast as before, namely, a scheme resistant against the data-bit DPA can be easily converted to that against the address-bit DPA with almost no penalty. The conversion can be applied not only binary methods but window-based methods. Moreover, we show the concrete security evaluation result of our countermeasure by theoretically and experimentally.

An approach of our countermeasure is similar to Random Register Renaming (RRR), a DPA countermeasure proposed by May et al. [27]. RRR is supposed to be implemented on a processor called NDISC, which can execute instructions in parallel, while our countermeasure does not require special hardware because it can be implemented by only software with very simple program code.

Side channel attack are so powerful that numerous countermeasures have been proposed (a brief overview is in [34]). However, some of them are proved or shown insecure by newer attacks. Finding a good countermeasure, which satisfies a certain security level and requires a compromisable processing speed, is becoming a hard task for implementers. In this paper, we give a complete comparison of countermeasures from several view points including the security level and the processing speed. As a result, our proposed method (combined with other countermeasures) can provide a good practical solution for resisting the side channel attacks.

In the following, we basically deal with scalar exponentiations in ECC, however, most of algorithms, especially proposed countermeasures, can be applied to other exponentiation based cryptosystems such as RSA. The rest of this paper is organized as follows: In section 2, we give a brief overview of side channel attacks and countermeasures. Then section 3 describes our proposed countermeasure and experimental results. A comparison of countermeasures are in section 4.

2 Preliminaries

In this section, we give a brief introduction of Elliptic Curve based Cryptosystems (ECC) and side channel attacks (SCA) against them (however, most attacks can applied to other exponentiation based cryptosystems such as RSA).

2.1 Elliptic Curve Based Cryptosystems (ECC)

Elliptic curve based cryptosystems (ECC) are one of the standard technology in the area of cryptography [10,28,37]. The most advantage of ECC is the key

length; it is currently chosen much shorter than those of existing other cryptosystems (RSA and ElGamal). This feature is quite suitable for implementing on smart cards.

Let K be a finite field with elements a power of a prime. An elliptic curve over K can be represented by the Weierstrass equation

$$E(K) := \{(x, y) \in K \times K \mid y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \mathcal{O}, \tag{1}$$

where $a_i \in K$. A special point $\mathcal{O}$ is called the point of infinity. An elliptic curve $E(K)$ has an additive group structure, in which an neutral element is the point of infinity. We call $P_1 + P_2$ ($P_1 \neq P_2$) the elliptic curve addition (ECADD) and $P_1 + P_2$ ($P_1 = P_2$), that is $2P_1$, the elliptic curve doubling (ECDBL). Let d be an integer and P be a point on the elliptic curve $E(K)$. A *scalar exponentiation* is to compute the point $dP = P + \cdots + P$ ($d-1$ additions). A dominant computation of all encryption/decryption and signature generation/verification algorithms of ECC is the computation of dP, where d is a secret integer and P is a base point. Numerous researches have been dedicated to improve the computing time of this part (see [8] for a survey) by omitting unnecessary computations or by finding various short-cuts.

Let $d = d_{n-1}2^{n-1} + \cdots + d_1 2^1 + d_0$ be a binary expression of d with $d_{n-1} = 1$. Then the binary method (from the most significant bit, MSB) for a scalar exponentiation is in Alg. 1. Similar method from the least significant bit (LSB) is easily constructed, but we omit this case in this paper for simplicity.

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

Alg. 1. Binary method (from MSB)

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

Alg. 2. Add-and-double-always method (from MSB)

2.2 Side Channel Attack

The *side channel attacks* (SCA) are powerful attacks against implementations of cryptographic schemes on smart cards. An adversary observes side channel information such as computing time, or power consumption of the device. Then he/she tries to reveal the secret information by analyzing side channel information. The *simple power analysis* (SPA) [19] and the differential power analysis (DPA) [20] are typical examples of SCA. SPA only uses a single observed information, while DPA uses a lot of observed information together with statistic tools.

Simple Power Analysis. The binary method of Alg. 1 computes ECADD only when the bit of the secret key d_i is 1. Therefore an adversary easily detects this irregular procedure by observing side channel information and obtain the bit information of d_i. This is a basic idea of the *simple power analysis* (SPA) [19].

There are three approaches to resist SPA. The first one uses the special addition formula, in which ECDBL and ECADD are computed by same order of operations (the indistinguishable addition formula [6]). Brier-Joye proposed the formula for the Weierstrass form [3], but the security hole was pointed out in [14]. The second one uses so called the *add-and-double-always* method [5], (Alg. 2 for example), in which both ECDBL and ECADD are computed in every bit (in step 3 and step 4), and a pattern of the side channel information is fixed. Thus the adversary cannot obtain the bit information of d by SPA.

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

Alg. 3. Montgomery ladder

```
INPUT: d[], P
OUTPUT: dP
1: T[0] = RPC(P)
2: for i=n-2 downto 0 {
3:   T[0] = ECDBL(T[0])
4:   T[1] = ECADD(T[0],P)
5:   T[0] = T[d[i]]
6: }
7: return invRPC(T[0])
```

Alg. 4. Add-and-double-always method (from MSB) and RPC

The third approach to resist SPA is the Montgomery ladder [24] which essentially computes ECDBL and ECADD repeatedly [3,7,12,13,18,31,32]. This method can be viewed as a variant of the add-and-double-always method, but provides a good processing speed [12,13]. A sample algorithm of the Montgomery ladder is in Alg. 3.

Differential Power Analysis. Even if a scheme is SPA-resistant, it is not always secure, because the *differential power analysis* (DPA) might reveal the secret key by analyzing observed information statistically [20,25]. In DPA, an adversary makes an assumption on d_i ($d_i = 0$, for example) and simulates the computation repeatedly. Then he/she divides the side channel information into two groups depending on the assumption, in order to make the bias of the hamming weights of the internal information between these groups. If the assumption is correct, then a difference of the information of two groups (a *spike*) can be observed in the trace. Countermeasures against DPA aim to make the simulation impossible by using random numbers [5]. By the randomization we are able to enhance an SPA-resistant scheme to be DPA-resistant easily. Earlier DPA (the *data-bit DPA* (DDPA)) only considered a correlation of data of registers to the secret information [20,25], while newer DPA (the *address-bit DPA* (ADPA)) also considers a correlation of addresses of registers [11,26].

Data-bit DPA: DDPA analyzes a relation between the secret key and data of registers. For example, after finishing step 5 in Alg. 2, data of `T[0]` is same as

that of `T[0]` if $d_i = 0$; otherwise it is same as that of `T[1]` if $d_i = 1$. Coron proposed the randomized projective coordinate (RPC) countermeasure [5] in order to resist DDPA. Let $P = (X : Y : Z)$ be a base point represented by the projective coordinate. Then $(X : Y : Z)$ equals to $(rX : rY : rZ)$ for all $r \in K^*$ mathematically; but they all are different data as bit sequences. The side channel information of a scalar exponentiation is randomized if $(X : Y : Z)$ is randomized to $(rX : rY : rZ)$. An example of RPC for the add-and-double-always method (from MSB) is given in Alg. 4, where a function `RPC` outputs the randomized point and a function `invRPC` denotes its inverse map.

Joye-Tymen proposed another countermeasure against DDPA [16], the randomized curve (RC) countermeasure, which uses an isomorphism of elliptic curves with which a curve equation and a base point are transformed with holding the group structure. Two isomorphic curves are same mathematically; but different as bit sequences. Thus the side channel information will be randomized if the curve and the base point is randomized. A sample algorithm is easily obtained similarly to Alg 4 by changing functions `RPC,invRPC` to `RC,invRC`.

Instead of changing the expression of a base point, Coron also proposed the randomized exponent and the randomized base point countermeasures, in which the scalar is randomized. However, Okeya-Sakurai showed the bias of these countermeasures [32]. Messerges et al. proposed the randomized start point countermeasure, in which a scalar is divided to two parts and computed by different methods [25]. Oswald-Aigner proposed another approach to randomize the scalar [30], but security problems are pointed out [33,35,39]. Walter proposed the MIST algorithm [38], which randomizes the intermediate data T (and U) by repeating $T = (d_i \bmod r_i)U + T$, $U = r_i U$ and $d_{i+1} = \lfloor d_i / r_i \rfloor$, where r_i are random numbers and initial values of T, U, d_i are $\mathcal{O}$, P, d respectively. Hasan proposed an approach by randomizing a scalar representation in the Koblitz curve [9].

Address-bit DPA: DDPA analyzes a relation between the secret key and data of registers [20,25]. Messerges et al. proposed the *address-bit DPA* (ADPA) for symmetric-key cryptosystems, which analyzes a relation between the key and addresses of registers [26]. Recently, Itoh et al. extended the analysis to public-key cryptosystems [11]. For example, in step 5 in Alg. 4, a correlation of data are given by $T[0] \leftarrow T[0]$ if $d_i = 0$, and $T[0] \leftarrow T[1]$ if $d_i = 1$; as these operations are same, substituted data are loaded from different registers and ADPA detects this relation. They concluded that only the exponent splitting countermeasure [6], in which the scalar is divided into $d = (d - r) + r$ for a random number r, and the randomized window method [15] are resistant against the attack [11]. But as a drawback, required computing time become at least twice than that of without countermeasures. When a special hardware is available, RRR [27], proposed by May et al., resists ADPA.

2.3 Window-Based Method

In a scalar exponentiation, a pre-computed table sometimes deduces the computing time if extra registers are available (the *window-based method*). The most

```
INPUT: d[], P
OUTPUT: dP
 1: W[0] = 0, W[1] = P
 2: W[2] = ECDBL(W[1])
 3: for i=3 upto 15 {
 4:   W[i] = ECADD(W[i-1],W[1])
 5: }
 6: T = W[d[n-1,n-4]]
 7: for i=n-5 downto 0 step -4 {
 8:   T = ECDBL(T), T = ECDBL(T)
 9:   T = ECDBL(T), T = ECDBL(T)
10:   T = ECADD(T,W[d[i,i-3]])
11: }
12: return T
```

Alg. 5. 4-bit window method

```
INPUT: d[], P
OUTPUT: dP
 1: W[0] = 0, W[1] = RPC(P)
 2: W[2] = ECDBL(W[1])
 3: for i=3 upto 15 {
 4:   W[i] = ECADD(W[i-1],W[1])
 5: }
 6: T = W[d[n-1,n-4]]
 7: for i=n-5 downto 0 step -4 {
 8:   T = ECDBL(T), T = ECDBL(T)
 9:   T = ECDBL(T), T = ECDBL(T)
10:   T = ECADD(T,W[d[i,i-3]])
11: }
12: return invRPC(T)
```

Alg. 6. 4-bit window method and RPC

simplest example is in Alg. 5 with window size 4 just for simplicity (where n is supposed to be a multiple of 4).

Similar to the binary method (Alg. 1), the window method is also vulnerable to SPA, because ECADD in step 10 is not computed if `W[i,i-3]` $= \mathcal{O}$. Möller proposed a method to construct an addition chain in which `W[]` is never equal to $\mathcal{O}$ [22,23], which assures the SPA-resistance. One can combine RPC or RC countermeasures with Möller's method (Alg. 6 for RPC case) in order to resist DDPA. However, Okeya-Sakurai showed the insecureness against the second-order data-bit DPA [34]. Other approach to resist both DDPA and ADPA is proposed by Itoh et al. [15], which randomize the window to be added in step 10 in Alg. 6.

3 Proposed Countermeasure

In this section, we propose a practical countermeasure, the *randomized addressing method*, against the address-bit DPA by randomizing addresses of registers used in a scalar exponentiation. An overhead of the countermeasure is small and the effectiveness will be shown by experimental results described in this section.

An approach of our countermeasure is similar to that of Randomized Register Renaming (RRR), a hardware-based DPA countermeasure proposed by May et al. [27]. However, implementation method is quite different. That is, our countermeasure can be implemented on various processors because it requires no special hardware, and can be implemented by only software with very simple program codes. These specifications make our countermeasure very practical. On the other hand, when a program code is implemented on the processor with RRR, physical registers are randomly chosen by hardware as far as the computation result is unchanged. Because execution timing, instruction order and physical registers are randomly changed, RRR is secure against DPA. However they did not show concrete results of security evaluation.

3.1 Outline

The address-bit DPA is based on the dependency of addresses of registers on the secret key. In other word, addresses of registers are determined by the secret key uniquely, because in Alg. 4, for example, $T[0] \leftarrow T[0]$ if $d_i = 0$ and $T[0] \leftarrow T[1]$ if $d_i = 1$ so that if d_i changes, then registers will change, too.

In order to resist ADPA, previous countermeasures randomize the scalar value [11]. However, the weakness lies on the direct correlation between the secret key and addresses of registers. What we should hide is this relation rather than the scalar value. So we randomize addresses of registers by a one-time random number $r_{n-1}2^{n-1} + \cdots + r_1 2 + r_0$ ($r_i \in \{0,1\}$). We change all parameters d_i to $d_i \oplus r_i$, where $\oplus$ denotes the XOR operation. Then all addresses of registers are randomized so that the side channel information will be randomized for each scalar exponentiation. This is a basic idea of our proposing countermeasure, the *randomized addressing method* (RA), against ADPA.

The most advantage of our method is the small overhead; the random number is easily generated and required additional operations are just XORs. However, our countermeasure has no DDPA-resistance. We have to combine other countermeasures to resist all side channel attacks. A DDPA-resistant scheme can be converted to an ADPA-scheme with almost no cost. Moreover, our countermeasure can be easily applied to window methods.

3.2 Description of Algorithms

Example algorithms of our countermeasure combined with the add-and-double-always method, the Montgomery ladder, and a window method (with 4-bit window) are in Alg. 7-9. All sample algorithms are resistant against SCA, namely SPA, DDPA, and ADPA. Here $r_{n-1}2^{n-1}+\cdots+r_1 2+r_0$ ($r_i \in \{0,1\}$) is a random number and r in Alg. 9 is a 4-bit random number. The functions `R,invR` denote `RPC,invRPC` or `RC,invR` described in section 2.2, respectively.

```
INPUT: d[], P
OUTPUT: dP
1: T[2] = R(P)
2: T[r[n-1]] = T[2]
3: for i=n-2 downto 0 {
4:   T[r[i+1]] = ECDBL(T[r[i+1]])
5:   T[1-r[i+1]] =
              ECADD(T[r[i+1]],T[2])
6:   T[r[i]] = T[d[i]⊕r[i+1]]
7: }
8: return invR(T[r[0]])
```

Alg. 7. Proposed countermeasure (add-and-double-always method from MSB)

```
INPUT: d[], P
OUTPUT: dP
1: T[r[n-1]] = R(P)
2: T[1-r[n-1]] = ECDBL(T[r[n-1]])
3: for i=n-2 downto 0 {
4:   T[2] = ECDBL(T[d[i]⊕r[i+1]])
5:   T[1] = ECADD(T[T[0],T[1])
6:   T[0] = T[2-(d[i]⊕r[i])]
7:   T[1] = T[1+(d[i]⊕r[i])]
8: }
9: return invR(T[r[0]])
```

Alg. 8. Proposed countermeasure (Montgomery ladder)

Note 1. In the above algorithms, a random number r can be computed on the fly for efficiency rather than generated and stored in memory at the beginning.

```
INPUT: d[], P
OUTPUT: dP
 1: W[0] = 0, W[1⊕r] = P
 2: W[2⊕r] = ECDBL(W[1⊕r])
 3: for i=3 upto 15 {
 4:   W[i⊕r] = ECADD(W[(i-1)⊕r],W[1⊕r])
 5: }
 6: T = W[d[n-1,n-4]⊕r]
 7: for i=n-5 downto 0 step -4 {
 8:   T = ECDBL(T), T = ECDBL(T)
 9:   T = ECDBL(T), T = ECDBL(T)
10:   T = ECADD(T,W[d[i,i-3]⊕r])
11: }
12: return invR(T)
```

Alg. 9. Proposed countermeasure (4-bit window method)

Note 2. A similar countermeasure by randomizing registers is also proposed by May et al. [27]. Their approach is to construct a specialized hardware, while ours is in the software level.

3.3 Security Analysis

Let us discuss the security of our countermeasure. Basically our scheme is designed to combine other countermeasures to totally resist SCA; we only consider the security against ADPA. In Alg. 7-9, addresses of registers are determined by $d_i \oplus r_i$. ADPA can distinguish whether two addresses corresponding to d_i and d_j are same or not. If these addresses are same, an adversary can know $d_i \oplus r_i = d_j \oplus r_j$. But he/she cannot determine $d_i = d_j$ or not, because r_i, r_j are chosen randomly. Conversely, even if $d_i = d_j$, addresses are not always same by r_i and r_j. Thus addresses of registers are randomized and our countermeasure is secure against ADPA.

3.4 Experimental Results

We performed an experiment for verifying the effect on the security by using our countermeasure. We used the address-bit DPA attack against an implementation of Montgomery ladder using RPC with and without the register randomization. In the experiment, the target processor was run at 10 MHz, the sampling ratio was set to 100 MHz, and we made a differential power trace as

$$(\textit{means of } 10000 \textit{ traces when loading } Q[addr[d_a]]) \\ - (\textit{means of } 10000 \textit{ traces when loading } Q[addr[d_b]])$$

for $d_a \neq d_b$ where $addr[d_x](x = a \text{ or } b)$ represents the address value determined from d_x in each implementation. Fig.1 shows a differential power trace without register randomization, and Fig.2 shows that with register randomization. In Fig.1, some spikes showing the evidence for $d_a \neq d_b$ are observed, but they are not found in Fig.2. Hence we confirmed the effect of our countermeasure for protecting against address-bit DPA attack.

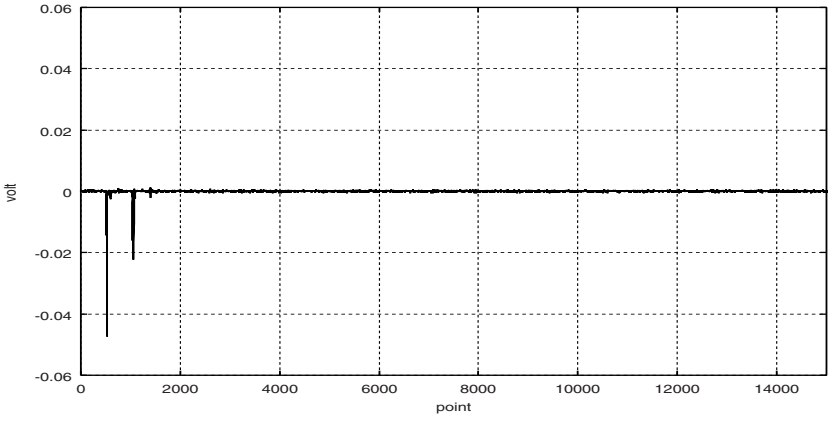

Fig. 1. Differential power traces without register randomization

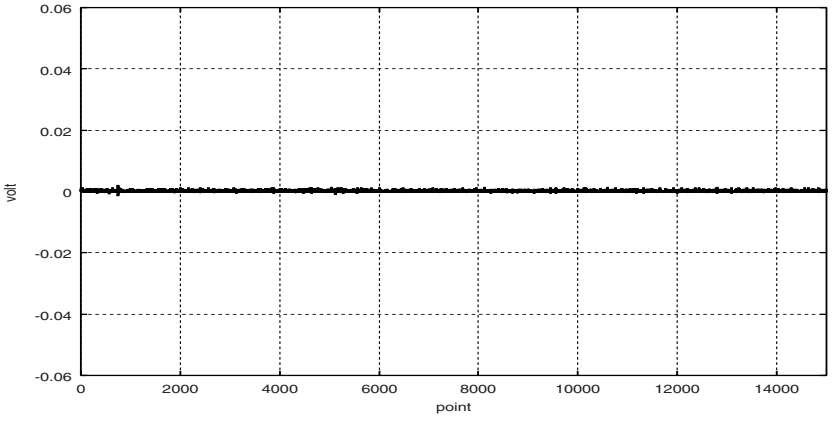

Fig. 2. Differential power traces with register randomization (proposed method)

4 Comparison

In order to resist the side channel attacks, numerous numbers of countermeasures have been proposed. However, finding a good countermeasure which has a certain security level and compromisable processing speed is a hard task for implementers, for there were no standard rule to compare these countermeasures. In this section, we provide a complete comparison of these countermeasures from viewpoints of security level, processing speed, and amount of required registers. Moreover combined countermeasures can be evaluated by the comparison. As a result, our proposed countermeasure can be combined to establish a practical solution for resisting side channel attacks in real world.

4.1 Evaluation Technique

We examine an exponentiation in 160-bit ECC from the security level, processing speed, and amount of required registers. In the followings, the binary methods and the Montgomery Ladder are called the *base algorithms* to which countermeasures are applied. We did not discuss the window based cases and the methods unavailable in the generic elliptic curve for space limitation.

Security Level. The security of a countermeasure against each of SPA, DDPA, and ADPA is measured by the *Attenuation Ratio* (AR)[1] proposed by Itoh et al. [15]. AR is given by a ratio of heights of spikes with and without the countermeasure. We denote the ratio against SPA, DDPA and ADPA by AR_S, AR_d and AR_a respectively. All AR ratios are between 0 and 1 and AR is desired to be smaller. If AR = 0, an adversary cannot observe spikes by any cost and the method is secure; if AR = 1, the adversary can always observe spikes and the method is totally insecure. AR_S takes 0 or 1, while AR_d and AR_a take arbitrary value from the interval $[0, 1]$.

Processing Speed. Processing speed of a scalar exponentiation dP is measured by the numbers of ECADDs denoted N_A and ECDBLs denoted N_D. For the base algorithms, N_D and N_A are given by integers. If a countermeasure requires ECADDs a times than before, we denote it by "$\times a$". We ignore the cost for randomizations or transformations required in the countermeasure because they are relatively small compared to N_A and N_D.

Register. Amount of required registers are measured by the number of points R_P and that of scalars R_s in a scalar exponentiation. We assume all points are represented by the projective coordinate just for simplicity. For the base algorithms, R_P and R_s are given by integers. For countermeasures, R_P and R_s are evaluated as extra points and scalars required in the exponentiation. If a countermeasure requires b extra points than before, R_P is denoted by "$+b$". We ignore the temporary registers for ECADD and ECDBL.

4.2 Countermeasures

In this section, we evaluate these values for each countermeasure. We only deal with countermeasures whose recommended parameters are explicitly given in the original papers, so we exclude RRR and some other countermeasures. We did not deal with the randomized addition-subtraction chain [30] because it is shown to be insecure [33,35,39].

SPA Countermeasure. The add-and-double-always method (Alg. 1) computes ECDBL and ECADD for every bit and is SPA-resistant ($\mathrm{AR}_S = 0$). But data computed in step 5 is predictable ($\mathrm{AR}_d = 1$) and a correlation of addresses

[1] AR is originally introduced to evaluate the security against DDPA, however it is easily to be applied to other attacks.

in step 5 is related to d ($\mathrm{AR}_a = 1$). Required ECDBLs are same ($\mathrm{N}_D = \times 1$) but ECADDs are doubled in average ($\mathrm{N}_A = \times 2$). Extra points and scalars are not required ($\mathrm{R}_P = +0,\ \mathrm{R}_s = +0$). Same discussion can be applied to the LSB case.

The Montgomery Ladder (Alg. 3) computes ECDBL and ECADD for every bit and is SPA-resistant ($\mathrm{AR}_S = 0$). But data computed in step 5 and 6 are predictable ($\mathrm{AR}_d = 1$) and a correlation of addresses of registers in step 3,5,6 are related to d ($\mathrm{AR}_a = 1$). Required ECDBLs and ECADDs are 160 ($\mathrm{N}_D = \mathrm{N}_A = 160$). Required registers are $\mathrm{R}_P = 3,\ \mathrm{R}_s = 1$.

DPA Countermeasure by Data Randomization. In the Randomized Projective Coordinate (RPC) [5], a point (X, Y, Z) is randomized to (rX, rY, rZ) by a 160-bit random number r and it resists DDPA ($\mathrm{AR}_d = 2^{-160}$). As it is vulnerable to SPA ($\mathrm{AR}_S = 1$), it must be used with the add-and-double-always method or the Montgomery Ladder. It is also vulnerable to ADPA ($\mathrm{AR}_a = 1$). Processing speed is unchanged ($\mathrm{N}_A = \mathrm{N}_D = \times 1$). An extra point for the randomized point and an extra scalar for random number are required ($\mathrm{R}_P = +1,\ \mathrm{R}_s = +1$).

In the Randomized Curve (RC) [16], a point (X, Y, Z) is randomized to (r^2X, r^3Y, Z) on an isomorphic curve by a 160-bit random number r and it resists DDPA ($\mathrm{AR}_d = 2^{-160}$). But it is vulnerable to SPA and ADPA ($\mathrm{AR}_S = \mathrm{AR}_a = 1$). Processing speed is unchanged ($\mathrm{N}_A = \mathrm{N}_D = \times 1$). Extra scalars for r and coefficients of isomorphic curves are required ($\mathrm{R}_P = +0,\ \mathrm{R}_s = +3$).

In the Randomized Base Point [5], a scalar exponentiation dP is computed by $d(P + R) - dR$ for a random point R. As R is chosen for each exponentiation, the method is DDPA-resistant ($\mathrm{AR}_d = 2^{-160}$). But it is vulnerable to SPA and ADPA ($\mathrm{AR}_S = \mathrm{AR}_a = 1$). The countermeasure requires two exponentiations and an extra ECADD ($\mathrm{N}_D = \times 2,\ \mathrm{N}_A = \times 2 + 1$). Extra registers for R and $P + R$ are required ($\mathrm{R}_P = +2,\ \mathrm{R}_s = +0$).

DPA Countermeasure by Scalar Randomization. In the Randomized Exponent [5], a scalar d is randomized to $d + r\phi$ for a random number r and the order ϕ of the base point P. In the original paper, the length of r is 20-bit and, thus, the countermeasure is DDPA-resistant ($\mathrm{AR}_d = 2^{-20}$) (There is an analysis which claims the 20-bit randomization is not sufficient [32]. Of course, we can relax the condition to 160-bit, however, the processing speed becomes much slower). It is also ADPA-resistant ($\mathrm{AR}_a = 0$), however it is vulnerable to SPA ($\mathrm{AR}_S = 1$). Processing speed becomes slower for the scalar is 20-bit longer ($\mathrm{N}_D = \mathrm{N}_A = \times 180/160 = \times 1.13$). An extra register for r is required ($\mathrm{R}_P = +0,\ \mathrm{R}_s = +1$).

In the Randomized Start Point [25], a start bit is chosen from a 160-bit scalar and an exponentiation is computed from the chosen bit by MSB for upper bits and by LSB for lower bits. However, the effect is rather small ($\mathrm{AR}_d = 1/160 = 2^{-7.3}$). It is also ADPA-resistant ($\mathrm{AR}_a = 0$), however it is vulnerable to SPA ($\mathrm{AR}_S = 1$). There requires no extra process ($\mathrm{N}_D = \mathrm{N}_A = \times 1$) and register ($\mathrm{R}_P = \mathrm{R}_s = +0$).

In the Exponent Splitting [6], a scalar d is divided into r and $d - r$ for a 160-bit random number r. As the secret information d is randomized, it is resistant

Table 1. Comparison of countermeasures (non-window methods)

No.	Method	AR_S	AR_d	AR_a	N_D	N_A	R_P	R_s
1	*binary method (form MSB)*	1	1	1	160	80	1	0
2	*binary method (from LSB)*	1	1	1	160	80	2	0
3	add-and-double-always method	0	1	1	$\times 1$	$\times 2$	$+1$	$+0$
4	*Montgomery ladder*	0	1	1	160	160	3	0
5	Randomized Projective Coordinate (RPC)	1	2^{-160}	1	$\times 1$	$\times 1$	$+1$	$+1$
6	Randomized Curve (RC)	1	2^{-160}	1	$\times 1$	$\times 1$	$+0$	$+3$
7	Randomized Base Point	1	2^{-160}	1	$\times 2$	$\times 2$	$+2$	$+0$
8	Randomized Exponent ($\lvert r\rvert = 20$)	1	2^{-20}	0	$\times 1.13$	$\times 1.13$	$+0$	$+1$
9	Randomized Start Point	1	$2^{-7.3}$	0	$\times 1$	$\times 1$	$+0$	$+0$
10	Exponent Splitting	1	2^{-160}	0	$\times 2$	$\times 2+1$	$+1$	$+2$
11	Randomized Addressing (RA)	1	1	0	$\times 1$	$\times 1$	$+0$	$+2$
1+3+5+11	Best Combination	0	2^{-160}	0	160	160	3	3
1+3+6+11	Best Combination	0	2^{-160}	0	160	160	2	5
4+5+11	Other Solution	0	2^{-160}	0	160	160	4	3
4+6+11	Other Solution	0	2^{-160}	0	160	160	3	5

against DDPA and ADPA ($AR_a = 0,\ AR_d = 2^{-160}$). However, it is vulnerable to SPA ($AR_S = 1$). The countermeasure requires two exponentiations and an extra ECADD ($N_D = \times 2,\ N_A = \times 2 + 1$). Extra registers for $(d - r)P$, r, d are required ($R_P = +1,\ R_s = +2$).

In the Randomized Addressing (RA) (Alg. 7), proposed in this paper, registers in step 4,5,6 are determined by a random bit r_i. It is ADPA-resistant ($AR_a = 0$), however, it is vulnerable to SPA and DDPA ($AR_S = AR_d = 1$). There requires no extra process ($N_D = N_A = \times 1$). Extra registers for r and $r \oplus d$ are required ($R_P = +0,\ R_s = +2$).

4.3 Combining Countermeasures

As we saw in the previous section, some all countermeasures only resist specific attacks. Implementers should combine them to resist all side channel attacks. In our setting, we can easily evaluate and compare each combined countermeasures. The security level $AR_S,\ AR_d,\ AR_a$ can be evaluated by their product. Similarly, the processing speed is evaluated by their product and amount of registers are evaluated by their sum. For example, if we use the Montgomery Ladder combined with RPC and RA, which is denoted as 4+5+11 in the following Table 1, the security levels, processing speed and amount of registers are given by $AR_S = 0,\ AR_d = 2^{-160},\ AR_a = 0,\ N_D = 160,\ N_A = 160,\ R_P = 4,\ R_s = 3$.

4.4 Comparison

This section provides a complete comparison of countermeasures (Table 1). The base algorithms are described in *italic*. As in the previous section, the security level can be evaluated for combined countermeasures. By this table, we can easily choose the most suitable countermeasure(s) from the security level, processing speed and amount of required registers.

From Table 1, we can conclude that combinations 1+3+5+11 and 1+3+6+11 provide the best combination from security level and processing speed. Other possible solutions are 4+5+11 and 4+6+11. In these combinations, more effective addition formula for ECADD and ECDBL are applicable and faster processing speed and fewer amount of registers can be expected [13]. In these combinations, our proposed countermeasure are used.

Note 3. Similar discussion for window based methods can be established. We do not compare them here for a space limitation.

5 Concluding Remarks

In this paper, a practical countermeasure, the randomized addressing method, against the address-bit DPA, was proposed. As an overhead is quite small, the method provides no slower scalar exponentiation algorithm with improving the security. Although an approach of our proposal is similar to the previous work by May et al. [27], implementational methodology is quite different. Our approach requires no special hardware and can be implemented on various processors with very simple program codes. Moreover, we showed a concrete security evaluation results by theoretically and experimentally.

In order to resist the side channel attacks, considering countermeasures against each attack is an important factor for implementers. However, when they are to establish a total security, combinations of some countermeasures are more important. For this purpose, our comparison table (Table 1) will be a great help.

Acknowledgment. The authors are grateful to Naoya Torii for his supports and many invaluable suggestions. The authors also thank to Colin Walter and anonymous referees for their helpful comments and references.

References

1. M. Akkar, P. Dischamp, and D. Moyart, "Power Analysis, What is Now Possible...", *Asiacrypt 2000*, LNCS 1976, pp. 489–502, Springer-Verlag, 2000.
2. O. Billet, and M. Joye, "The Jacobi Model of an Elliptic Curve and Side-Channel Analysis", Cryptology ePrint Archive, 2002/125, 2002. Available from `http://eprint.iacr.org/2002/125/`
3. E. Brier, and M. Joye, "Weierstraß Elliptic Curves and Side-Channel Attacks", *PKC 2002*, LNCS 2274, pp. 335–345, Springer-Verlag, 2002.
4. I. Blake, G. Seroussi, and N. Smart, *Elliptic Curves in Cryptography*, Cambridge University Press, 1999.
5. J. Coron, "Resistance against differential power analysis for elliptic curve cryptosystem", *CHES'99*, LNCS 1717, pp. 292–302, Springer-Verlag, 1999.
6. C. Clavier, and M. Joye, "Universal exponentiation algorithm – A first step towards provable SPA-resistance –", *CHES 2001*, LNCS 2162, pp. 300–308, Springer-Verlag, 2001.

7. W. Fischer, C. Giraud, E. Knudsen, and J. Seifert, "Parallel Scalar Multiplication on General Elliptic Curves over F_p Hedged Against Non-Differential Side-Channel Attacks", Cryptology ePrint Archive, 2002/007, 2002. Available from `http://eprint.iacr.org/2002/007/`
8. D. Gordon, "A Survey of fast exponentiation methods", *J. Algorithms*, vol.27, pp. 129–146, 1998.
9. M. Hasan, "Power Analysis Attacks and Algorithmic Approaches to Their Countermeasures for Koblitz Curve Crypto-systems", *IEEE Trans. Computers*, pp. 1071–1083, October 2001.
10. IEEE P1363, Standard Specifications for Public-Key Cryptography, 2000.
11. K. Itoh, T. Izu, and 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.
12. T. Izu, B. Möller, and T. Takagi, "Improved Elliptic Curve Multiplication Methods Resistant against Side Channel Attacks", *Indocrypt 2002*, LNCS 2551, pp. 296–313, Springer-Verlag, 2002.
13. T. Izu, and T. Takagi, "A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks", *PKC 2002*, LNCS 2274, pp. 280–296, Springer-Verlag, 2002.
14. T. Izu, and T. Takagi, "Exceptional Procedure Attack on Elliptic Curve Cryptosystems", *PKC 2003*, LNCS 2567, pp. 224–239, Springer-Verlag, 2003.
15. K. Itoh, J. Yajima, M. Takenaka, and N. Torii, "DPA Countermeasures by Improving the Window Method", *CHES 2002*, LNCS 2523, pp. 303–317, Springer-Verlag, 2003.
16. M. Joye, C. Tymen, "Protections against Differential Analysis for Elliptic Curve Cryptography", *CHES 2001*, LNCS 2162, pp. 377–390, Springer-Verlag, 2001.
17. M. Joye, J. Quisquater, "Hessian Elliptic Curves and Side-Channel Attacks", *CHES 2001*, LNCS 2162, pp. 402–410, Springer-Verlag, 2001.
18. M. Joye, and S-M. Yen, "The Montgomery Powering Ladder", *CHES 2002*, LNCS 2523, pp. 291–302, Springer-Verlag, 2003.
19. C. Kocher, "Timing attacks on Implementations of Diffie-Hellman, RSA, DSS, and other systems", *Crypto'96*, LNCS 1109, pp. 104–113, Springer-Verlag, 1996.
20. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis", *Crypto'99*, LNCS 1666, pp. 388–397, Springer-Verlag, 1999.
21. P. Liardet, N. Smart, "Preventing SPA/DPA in ECC Systems Using the Jacobi From", *CHES 2001*, LNCS 2162, pp. 391–401, Springer-Verlag, 2001.
22. B. Möller, "Securing Elliptic Curve Point Multiplication against Side-Channel Attacks", *ISC 2001*, LNCS 2200, pp. 324–334, Springer-Verlag, 2001.
23. B. Möller, "Parallelizable Elliptic Curve Point Multiplication Method with Resistance against Side-Channel Attacks", *ISC 2002*, LNCS 2433, pp. 402–413, Springer-Verlag, 2002.
24. P. Montgomery, "Speeding the Pollard and elliptic curve methods for factorizations", *Math. of Comp*, vol.48, pp. 243–264, 1987.
25. T. Messerges, E. Dabbish, and R. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards", *CHES'99*, LNCS 1717, pp. 144–157, Springer-Verlag, 1999.
26. T. Messerges, E. Dabbish, and R. Sloan, "Investigations of Power Analysis Attacks on Smartcards", preprint, USENIX Workshop on Smartcard Technology, 1999.
27. D. May, H.L. Muller, and N.P. Smart, "Random Register Renaming to Foil DPA", *CHES 2001*, LNCS 2162, pp. 28–38, Springer-Verlag, 2001.

28. National Institute of Standards and Technology, Recommended Elliptic Curves for Federal Government Use, in the appendix of FIPS 186-2.
29. E. Oswald, "Enhancing Simple Power-Analysis Attacks on Elliptic Curve Cryptosystems", *CHES 2002*, LNCS 2523, pp. 82–97, Springer-Verlag, 2003.
30. E. Oswald, and M. Aigner, "Randomized Addition-Subtraction Chains as a Countermeasure against Power Attacks", *CHES 2001*, LNCS 2162, pp. 39–50, Springer-Verlag, 2001.
31. K. Okeya, H. Kurumatani, and K. Sakurai, "Elliptic curves with the Montgomery form and their cryptographic applications", *PKC 2000*, LNCS 1751, pp. 446–465, Springer-Verlag, 2000.
32. K. Okeya, and K. Sakurai, "Power analysis breaks elliptic curve cryptosystem even secure against the timing attack", *Indocrypt 2000*, LNCS 1977, pp. 178–190, Springer-Verlag, 2000.
33. K. Okeya, and K. Sakurai, "On Insecurity of the Side Channel Attack Countermeasure Using Addition-Subtraction Chains under Distinguishability between Addition and Doubling", *ACISP 2002*, LNCS 2384, pp. 420–435, Springer-Verlag, 2002.
34. K. Okeya, and K. Sakurai, "A Second-Order DPA Attack Breaks a Window-method based Countermeasure against Side Channel Attacks", *ISC 2002*, LNCS 2443, pp. 389–401, Springer-Verlag, 2002.
35. K. Okeya, and K. Sakurai, "A Multiple Power Analysis Breaks the Advanced Version of the Randomized Addition-Subtraction Chains Countermeasure against Side Channel Attacks", to appear in the proceedings of 2003 IEEE Information Theory Workshop.
36. N. Smart, "The Hessian Form of an Elliptic Curve", *CHES 2001*, LNCS 2162, pp. 118–125, Springer-Verlag, 2001.
37. Standards for Efficient Cryptography Group (SECG), Specification of Standards for Efficient Cryptography.
38. C. Walter, "MIST: An Efficient, Randomized Exponentiation Algorithm for Resisting Power Analysis", *CT-RSA 2002*, LNCS 2271, pp. 53–66, Springer-Verlag, 2002.
39. C. Walter, "Security Constraints on the Oswald-Aigner Exponentiation Algorithm", Cryptology ePrint Archive, Report 2003/013, 2003. Available from `http://eprint.iacr.org/2003/013/`