An Efficient Countermeasure against Correlation Power-Analysis Attacks with Randomized Montgomery Operations for DF-ECC Processor

Jen-Wei Lee, Szu-Chi Chung, Hsie-Chia Chang, and Chen-Yi Lee

Department of Electronics Engineering and Institute of Electronics,
National Chiao Tung University, Hsinchu, Taiwan
jenweilee@gmail.com, {phonchi,hcchang,cylee}@si2lab.org

Abstract. Correlation power-analysis (CPA) attacks are a serious threat for cryptographic device because the key can be disclosed from data-dependent power consumption. *Hiding* power consumption of encryption circuit can increase the security against CPA attacks, but it results in a large overhead for cost, speed, and energy dissipation. *Masking* processed data such as randomized scalar or primary base point on elliptic curve is another approach to prevent CPA attacks. However, these methods requiring pre-computed data are not suitable for hardware implementation of real-time applications. In this paper, a new CPA countermeasure performing all field operations in a randomized Montgomery domain is proposed to eliminate the correlation between target and reference power traces. After implemented in 90-nm CMOS process, our protected 521-bit dual-field elliptic curve cryptographic (DF-ECC) processor can perform one elliptic curve scalar multiplication (ECSM) in 4.57ms over $GF(p^{521})$ and 2.77ms over $GF(2^{409})$ with 3.6% area and 3.8% power overhead. Experiments from an FPGA evaluation board demonstrate that the private key of unprotected device will be revealed within 10^3 power traces, whereas the same attacks on our proposal cannot successfully extract the key value even after 10^6 measurements.

Keywords: Elliptic curve cryptography (ECC), side-channel attacks, power-analysis attacks, Montgomery algorithm.

1 Introduction

Elliptic curve cryptography (ECC) independently introduced by Koblitz [1] and Miller [2] has been widely applied to provide a confident scheme for information exchange. For the past several years, many previous works [3], [4], [5], [6] have been published for ECC hardware implementation aiming at the performance improvement. However, even the ECC is secure at cryptanalysis, the private data of a unprotected hardware device can be extracted by the physical attacks due to side-channel leakage. The power-analysis attacks, initially presented by Kocher [7], can reveal the key value by analyzing the power information of a cryptographic implementation such as on an ASIC, FPGA or microprocessor.

During the device processing, simple power-analysis (SPA) attacks can distinguish the key value through visual inspection because of the specifically active circuit with direct hardware scheduling. The unified elliptic curve (EC) point calculation [8], [9] is usually used to avoid the variation of power consumption over time. However, the correlation power-analysis (CPA) attacks [10] computing the correlation between target power traces and power model by statistical approach can reveal the key value due to the existence of key-dependent operations in every round of calculation. For ECC primitives specified in IEEE P1363 [11], the CPA attacks can be applied to EC integrated encryption system, single pass EC Diffie-Hellman or single pass EC Menezes-Qu-Vanstone key agreement because the private key is kept invariant for a long time duration.

Hiding technique with algorithm-independent dedicated circuit is a common approach to protect cryptographic processors from attackers collecting the key-dependent characteristics of power traces. In [12], wave dynamic differential logic circuit with regular routing algorithm is exploited to equalize the current between rising and falling transitions. However, at least double hardware latency, area cost, and energy for unprotected encryption engines are required due to precharging for half cycle, and generating complementary logic outputs from divided single ended modules with equivalent power consumption. Switched capacitor [13] is able to isolate the encryption core from the external power supplies, but this approach results in 50% speed loss for replenishing charge every cycle. In order to avoid the throughput degradation, a countermeasure circuit using digital controlled ring oscillators [14] is designed outside of the critical path. The concept is to generate random noise power to dominate the power consumption of arithmetic unit, and then the correlation peak would not be found even matching the correct key value. But this demands extra 100% power overhead for the key-dependent processing element.

At the algorithm level, *masking* the processed data independent of power consumption is another approach to avoid the CPA attacks. Since the scalar K of EC point calculation is periodic with the point order $\#E$, a randomized scalar technique proposed by Coron [15] can be adopted to change the key value by adding $\alpha \cdot \#E$ for every elliptic curve scalar multiplication (ECSM) such as $KP = (K + \alpha \cdot \#E)P$, where α is a random integer and P is a primary base point on EC. However, with this method, the throughput overhead is inevitable due to extending the key length. In [9], the ECSM of 521-bit key extended with a 32-bit random value needs 10% more execution time to be carried out than that of unprotected approach. Another CPA countermeasure also presented in [15] is to mask the primary base point with pre-computed random points R and $S = KR$. Then the ECSM is achieved by computing $K(P + R) = KP'$ and subtracting S before returning such that $KP' - S = KP$. For every next ECSM calculation, the random points R and S are refreshed by performing $R \leftarrow (-1)^{\beta}2R$ and $S \leftarrow (-1)^{\beta}2S$ with a single random bit β. But the time-cost random point generation is not suitable for real-time applications as the EC parameters are various with different users. In [16], the EC isomorphism method can randomize the primary base point by simple finite field operations without pre-computing random points. However, it is limited to be applied in single finite field $GF(p)$.

In this brief, we propose a new efficient countermeasure to overcome the CPA attacks by computing overall dual-field ECC functions in a *randomized Montgomery domain*. The feature of our approach is to mask the intermediate values in not only the arithmetic but also the temporary register. Thus it is unnecessary to extend the key length, customize circuit and modify the routing algorithm in ASIC or FPGA design flow. Since our proposed design adopts simple logic circuit to counteract CPA attacks, the hardware cost overhead could be significantly reduced, and the maximum operating frequency of protected design is the same as that of unprotected design using conventional Montgomery algorithm. Additionally, by reducing the iteration time of the division, which dominates other field operations in computation time, the speed can be improved further.

The remainder of the paper is outlined as follows. CPA attacks applied on the ECC device are introduced in Section 2. The proposed countermeasure method and design architecture are given in Section 3 and Section 4, respectively. Section 5 shows the FPGA power measurement and ASIC implementation results. Section 6 concludes this work.

2 CPA Attacks on ECC Device

Algorithm 1 presented in [8] is a usually adopted approach to counteract SPA attacks by regularly performing the ECSM $KP = P + \cdots + P$, where K is the m-bit private key and P is a point on elliptic curves (ECs). But the intermediate values of elliptic curve point doubling in Step 3 and Step 4 still have dependence on the zero and non-zero bit of the key value. Hence, with a chosen point P, the key value can be distinguished by matching the power trace segment of accessing the memory storage for point coordinates P_1 or P_2.

Algorithm 1. Montgomery ladder ECSM algorithm

```
Input: $K$ and $P$
Output: $KP$
1. Let $P_1 \leftarrow P$, $P_2 \leftarrow 2P$
2. For $i$ from $m-2$ to 0 do
3.     If $K_i = 1$ then $P_1 \leftarrow P_1 + P_2, P_2 \leftarrow 2P_2$
4.     else $P_2 \leftarrow P_1 + P_2, P_1 \leftarrow 2P_1$
5. Return $P_1$
```

Fig. 1 illustrates the scenario of CPA attacks. For ECC primitives, the primary base point is commonly public. Thus the power model can be characterized from the hamming distance of memory storage for key-dependent point coordinates by measuring the device sample before the statistical analysis, which computes the correlation between the measured target power traces and the power model. The correlation value of correct hypothesis will be larger than that of the others due to the same hamming distance of processed data. Through this approach, the overall binary key can be extracted after $m - 1$ rounds in linear time.

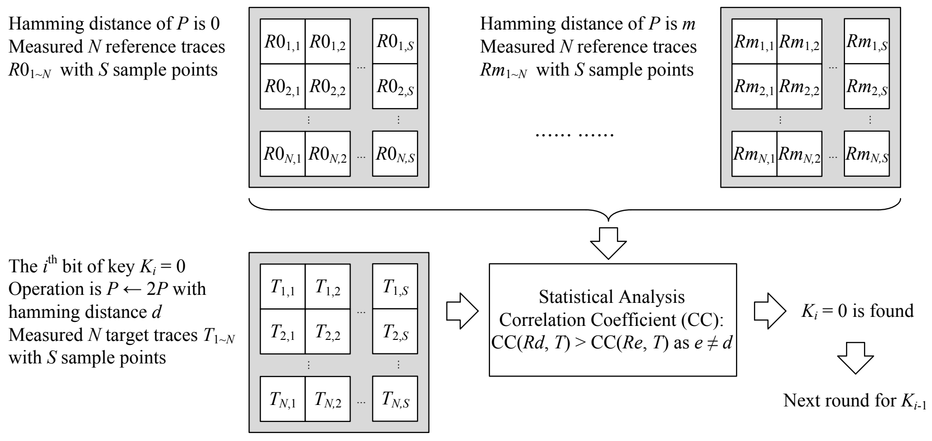

Hamming distance of P is 0
Measured N reference traces R0_{1~N} with S sample points
R0_{1,1} R0_{1,2} R0_{1,S}
R0_{2,1} R0_{2,2} R0_{2,S}
R0_{N,1} R0_{N,2} R0_{N,S}
Hamming distance of P is m
Measured N reference traces Rm_{1~N} with S sample points
Rm_{1,1} Rm_{1,2} Rm_{1,S}
Rm_{2,1} Rm_{2,2} Rm_{2,S}
Rm_{N,1} Rm_{N,2} Rm_{N,S}
The i^th bit of key K_i = 0
Operation is P ← 2P with hamming distance d
Measured N target traces T_{1~N} with S sample points
T_{1,1} T_{1,2} T_{1,S}
T_{2,1} T_{2,2} T_{2,S}
T_{N,1} T_{N,2} T_{N,S}
Statistical Analysis
Correlation Coefficient (CC):
CC(Rd, T) > CC(Re, T) as e ≠ d
K_i = 0 is found
Next round for K_{i-1}

Fig. 1. CPA attacks on an ECC device operating in a specific domain

3 Proposed Algorithm against CPA Attacks

The fundamental concept of CPA countermeasure is to break the dependency between intermediate values and power traces. For achieving the EC point calculation, the well-known Montgomery algorithm [17] is usually adopted to perform the field arithmetic in a specific domain such that $A \equiv a \cdot r \pmod{p}$, where a is in the integer domain and $r \equiv 2^m \pmod{p}$ is the Montgomery constant with m-bit field length. In this work, we introduce an approach to resist the CPA attacks at modular algorithm by calculating the operands in a randomized Montgomery domain $A \equiv a \cdot 2^\lambda \pmod{p}$, where the domain value λ equals the hamming weight (HW) of an n-bit random value α. Note that n is the maximum field length and the bit values of $(\alpha_{n-1}, \alpha_{n-2}, \ldots, \alpha_m)$ are set to zero for preventing λ from exceeding m. By exploiting this approach, the intermediate values can be masked because they are various with different domain values such as $2^g \pmod{p} \neq 2^h \pmod{p}$ when $0 \leq g \neq h < m$. Since the proposed method is to randomize intermediate values in basic modular operations, the SPA resistant ECSM algorithm shown in Algorithm 1 can still be applied without computation overhead from extended scalar length, and there is no need for pre-computed EC points. The overall randomized Montgomery operations for input operands $X \equiv x \cdot 2^\lambda \pmod{p}$ and $Y \equiv y \cdot 2^\lambda \pmod{p}$ are summarized in Table 1.

3.1 Randomized Montgomery Multiplication

Algorithm 2 shows our proposed randomized Montgomery multiplication which contains two operating steps in every iteration to change the intermediate domain value λ', and these steps are determined by the i^{th} bit of random value α. If $\alpha_i = 1$, the domain value of output operand R decreases by one in Step 4 such

Table 1. Operations in Randomized Montgomery Domain

Operation	Arithmetic
Randomized Montgomery multiplication (RMM)	$\mathrm{RMM}(X,Y) \equiv x \cdot y \cdot 2^{\lambda} \pmod{p}$
Randomized Montgomery division (RMD)	$\mathrm{RMD}(X,Y) \equiv x \cdot y^{-1} \cdot 2^{\lambda} \pmod{p}$
Randomized addition (RA)	$\mathrm{RA}(X,Y) \equiv (x+y) \cdot 2^{\lambda} \pmod{p}$
Randomized subtraction (RS)	$\mathrm{RS}(X,Y) \equiv (x-y) \cdot 2^{\lambda} \pmod{p}$

as $R = (R + V_0 \cdot S)/2 \pmod{p}$; the domain value remains the same as $\alpha_i = 0$ in Step 5 such as $R = (R + V_0 \cdot S) \pmod{p}$. The initial values of operands (V, R, S) are set to be $(X, 0, Y)$. In further iterative calculation, the bit value V_0 is equal to the i^{th} bit value of X, and the operand S doubles its value as $\alpha_i = 0$. Base on these, the functionality can be derived as follows:

- For 1^{st} iteration, the intermediate result of R is $(X_0 \cdot Y) \cdot 2^{-\alpha_0} \pmod{p}$.
- For 2^{nd} iteration, R becomes $((X_0 \cdot Y) \cdot 2^{-\alpha_0} \pmod{p} + X_1 \cdot (2^{1-\mathrm{HW}(\alpha_0)} \cdot Y)) \cdot 2^{-\alpha_1} \pmod{p}$.
- Until m^{th} iteration, the final result of R is $(\cdots(((X_0 \cdot Y) \cdot 2^{-\alpha_0} \pmod{p} + X_1 \cdot (2^{1-\mathrm{HW}(\alpha_0)} \cdot Y)) \cdot 2^{-\alpha_1} \pmod{p} + X_2 \cdot (2^{2-\mathrm{HW}(\alpha_1,\alpha_0)} \cdot Y)) \cdot 2^{-\alpha_2} \pmod{p} + \cdots + X_{m-1} \cdot (2^{m-1-\mathrm{HW}(\alpha_{m-2},\cdots,\alpha_1,\alpha_0)} \cdot Y)) \cdot 2^{-\alpha_{m-1}} \pmod{p}$
  $\equiv (X_0 \cdot Y \cdot 2^{-\mathrm{HW}(\alpha_{m-1},\ldots,\alpha_0)}) \pmod{p} + (X_1 Y \cdot 2^{-\mathrm{HW}(\alpha_{m-1},\ldots,\alpha_0)+1}) \pmod{p} + \cdots + (X_{m-1} \cdot Y \cdot 2^{-\mathrm{HW}(\alpha_{m-1},\ldots,\alpha_0)+m-1}) \pmod{p}$
  $\equiv X \cdot Y \cdot 2^{-\mathrm{HW}(\alpha_{m-1},\ldots,\alpha_0)} \pmod{p}$
  $\equiv X \cdot Y \cdot 2^{-\lambda} \pmod{p}$.

Hence, the randomized Montgomery multiplication in Algorithm 2 can be performed in m iterations, the same as those in conventional radix-2 Montgomery multiplication.

Algorithm 2. Radix-2 randomized Montgomery multiplication

```
Input: $X, Y, p$, and $\alpha$
Output: $R = \mathrm{RMM}(X,Y)$
1. Let $V = X$, $R = 0$, $S = Y$
2. For $i$ from 0 to $m-1$ do
3.     $R \equiv R + V_0 \cdot S \pmod{p}, V = V/2$
4.     If $\alpha_i = 1$ then $R \equiv R/2 \pmod{p}$
5.     else $S \equiv 2S \pmod{p}$
6. Return $R$
```

Algorithm 3 shows a radix-4 approach to Algorithm 2 for almost 50% iteration reduction. The domain value of R is determined by the HW of two continuous bits of random value α in Steps 5, 6, and 7. For the case of HW = 2, it is reduced by two through performing quartering operation such as $R \equiv R/4 \pmod{p}$. While halving R and doubling S operations are performed as HW = 1, these are deduced by computing one iteration of radix-2 Montgomery reduction and one iteration of radix-2 modular reduction in single period. For the rest case of HW $= 0$, the operand $S \equiv 4S \pmod p$ is performed due to the unchanged domain value of R.

Algorithm 3. Radix-4 randomized Montgomery multiplication

```
Input: $X, Y, p$, and $\alpha$
Output: $R = \mathrm{RMM}(X, Y)$
1. Let $V = X$, $R = 0$, $S = Y$
2. For $i$ from 0 to $\lceil \frac{m}{2} \rceil - 1$ do
3.     If $m \pmod 2 \equiv 1$ and $i = \lceil \frac{m}{2} \rceil - 1$ then
           $R \equiv R + V_0 \cdot S \pmod p$, $V = \frac{V}{2}$
4.     else
           $R \equiv R + V_0 \cdot S + V_1 \cdot 2S \pmod p$, $V = \frac{V}{4}$
5.     If $(\alpha_{2i+1}, \alpha_{2i}) = (1, 1)$ then
           $R \equiv \frac{R}{4} \pmod p$
6.     else if $(\alpha_{2i+1}, \alpha_{2i}) = (1, 0)$ or $(0, 1)$ then
           $R \equiv \frac{R}{2} \pmod p$, $S \equiv 2S \pmod p$
7.     else
           $S \equiv 4S \pmod p$
8. Return $R$
```

3.2 Randomized Montgomery Division

To achieve the division in Montgomery domain, Kaliski [18] first proposed an iterative algorithm which needs $m \sim 2m$ iterations of successive reduction, $0 \sim m$ iterations for degree recovery (reduce intermediate domain value λ' to be m as $\lambda' > m$), and two additional Montgomery multiplications with a final modular reduction $p - R$. The algorithm presented in [18] is formulated from the identical equations as follows:

$$\begin{cases} Y \cdot R \equiv -U \cdot 2^{\lambda'} \pmod p \\ Y \cdot S \equiv V \cdot 2^{\lambda'} \pmod p. \end{cases}$$

Based on Kaliski's method, we derive a new randomized Montgomery division which is described in Algorithm 4. To directly achieve the division operation without additional multiplication and final modular reduction, our method is to modify the initial values of (U, V, R, S) to be $(p, Y, 0, X)$ in Step 1 and the RS data path with modular subtraction in Steps 10, 11, 13, 14. Then the identities become

$$\begin{cases} X^{-1} \cdot Y \cdot R \equiv U \cdot 2^{\lambda'} \pmod p \\ X^{-1} \cdot Y \cdot S \equiv V \cdot 2^{\lambda'} \pmod p. \end{cases}$$

Similar to RMM, the RS data path between the Montgomery domain and integer domain is determined by the i^{th} bit value of α. The domain value of operands R and S increases by one as $\alpha_i = 1$ and remains the same as $\alpha_i = 0$.

For further reducing the degree recovery phase, the RS data path turns into dividing values by two in Steps 5, 8, 11, 14 to keep the intermediate domain

Algorithm 4. Radix-2 randomized Montgomery division

```
Input: $X, Y, p$, and $\alpha$
Output: $R = \mathrm{RMD}(X, Y)$
1. Let $U = p, V = Y, R = 0, S = X$
2. While ($V > 0$) do
3.    If $U$ is even then $U = U/2$
4.       If $\alpha_i = 1$ then $S \equiv 2S \pmod p$
5.       else $R \equiv R/2 \pmod p$
6.    else if $V$ is even then $V = V/2$
7.       If $\alpha_i = 1$ then $R \equiv 2R \pmod p$
8.       else $S \equiv S/2 \pmod p$
9.    else if $U > V$ then $U = (U - V)/2$
10.      If $\alpha_i = 1$ then $R \equiv R - S \pmod p$, $S \equiv 2S \pmod p$
11.      else $R \equiv (R - S)/2 \pmod p$
12.   else $V = (V - U)/2$
13.      If $\alpha_i = 1$ then $S \equiv S - R \pmod p$, $R \equiv 2R \pmod p$
14.      else $S \equiv (S - R)/2 \pmod p$
15.   If $i < m$ then $i = i + 1$
16. Return $R$
```

value in $\lambda = \mathrm{HW}(\alpha)$ as $i = m$. Thus the identities in Algorithm 4 are given as follows:

$$\text{If } i < m, \text{then} \begin{cases} X^{-1} \cdot Y \cdot R \equiv U \cdot 2^{\lambda'} \pmod p \\ X^{-1} \cdot Y \cdot S \equiv V \cdot 2^{\lambda'} \pmod p \end{cases}$$
$$\text{else} \begin{cases} X^{-1} \cdot Y \cdot R \equiv U \cdot 2^{\lambda} \pmod p \\ X^{-1} \cdot Y \cdot S \equiv V \cdot 2^{\lambda} \pmod p. \end{cases}$$

Before the last iteration, both U and V are 1 because the initial values of U and V are relatively prime. Then after finishing the iterative operations in Step 2, the values of (U, V, R, S) become $(1, 0, X \cdot Y^{-1} \cdot 2^{\lambda} \pmod p, 0)$. As a result, the proposed randomized division algorithm requires at most $2m$ iterations of successive reduction. Table 2 shows the expected operation time and the comparison with related works on modifying radix-2 Montgomery division algorithm. With randomization capability, Algorithm 4 will also benefit the hardware design owing to the low latency.

Table 2. Analysis of Various Division Algorithms

	Algorithm 4	TCAS-I'06 [3]	ESSCIRC'10 [9]
Iteration Time	$m \sim 2m$	$m \sim 2m$	$m \sim 3m$
Multiplication	0	$2 \sim 3$	0
Domain	Random 2^{λ}, $0 \leq \lambda \leq m$	Fixed 2^m	Fixed 2^m

Algorithm 5 shows the radix-4 randomized Montgomery division derived from Algorithm 4, and there are more branches in the algorithm as the radix becomes lager. To remain the domain value of R unpredictable in the flexible range of

$[0, m-1)$, it is determined by the HW of random value α_i or (α_{i+1}, α_i). The values of UV is reduced to at least $UV/4$ except $U \equiv 1 \pmod 4$, $V \equiv 3 \pmod 4$ or $U \equiv 3 \pmod 4$, $V \equiv 1 \pmod 4$ in Steps 17 and 18. With this approach and a radix-4 RMM given in Algorithm 3, the EC point calculation can be carried out faster in affine coordinates than that in projective coordinates [19], where the iteration time ratio RMD/RMM $\cong 1.32$ over $GF(p)$ and 1.44 over $GF(2^m)$.

4 Hardware Architecture of DF-ECC Processor

Fig. 2 shows the block diagram of the proposed dual-field ECC (DF-ECC) processor. For the CPA resistance, all field operations over $GF(p)$ and $GF(2^m)$ are performed by the Galois field arithmetic unit (GFAU) in a randomized Montgomery domain. The operating domain is determined by the value in domain shift register, which is sourced from a 1-bit random number generator (RNG) and refreshed before the next ECSM calculation. For the flexibility, we use an all-digital RNG utilizing the cycle-to-cycle time jitter in free-running oscillators with a synchronous feedback post-processor [20]. The overall architecture of CPA countermeasure circuit is shown in Fig. 3. Besides, to efficiently store the long bit length operands including EC parameters and points, a block memory of register file is exploited.

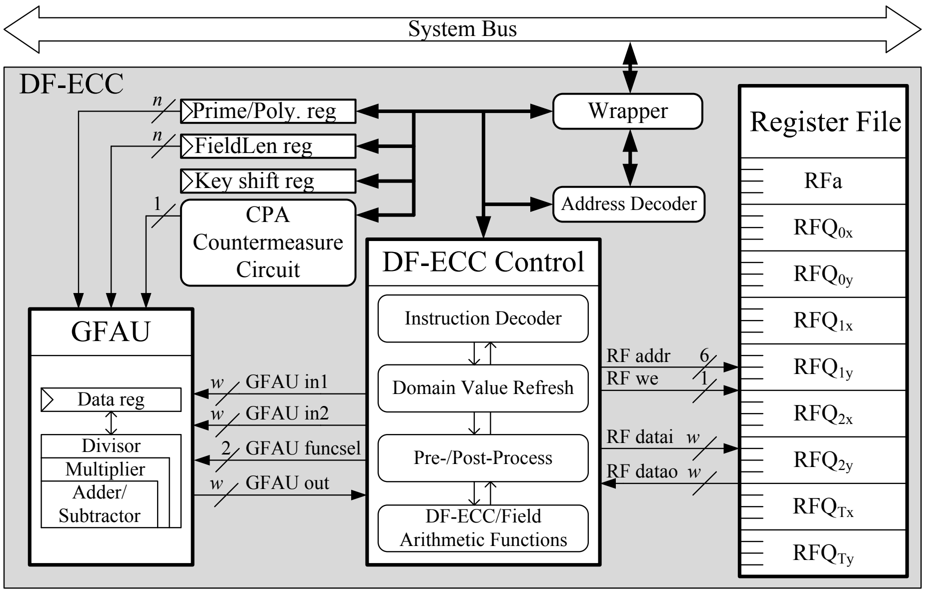


Fig. 2. Overall diagram for the DF-ECC processor

Algorithm 5. Radix-4 randomized Montgomery division

```
Input: $X, Y, p$, and $\alpha$
Output: $R = \mathrm{RMD}(X, Y)$
1. Let $U = p, V = Y, R = 0, S = X, i = 0$
2. While $(V > 0)$ do
3.     $c \equiv U \pmod 4, d \equiv V \pmod 4, t = 2$
4.     If $i = m - 1$ then
           $R \equiv 2R \pmod p, S \equiv 2S \pmod p, t = 1$
5.     else if $c = 0$ then $U = \frac{U}{4}, S \equiv 4S \pmod p$
6.     else if $d = 0$ then $V = \frac{V}{4}, R \equiv 4R \pmod p$
7.     else if $c = d$ then
8.         If $U > V$ then $U = \frac{U-V}{4}$,
             $R \equiv R - S \pmod p, S \equiv 4S \pmod p$
9.         else $V = \frac{V-U}{4}$,
             $S \equiv S - R \pmod p, R \equiv 4R \pmod p$
10.    else if $c = 2$ then
11.        If $\frac{U}{2} > V$ then $U = \frac{\frac{U}{2}-V}{2}$,
             $R \equiv R - 2S \pmod p, S \equiv 4S \pmod p$
12.        else $V = \frac{V-\frac{U}{2}}{2}, U = \frac{U}{2}$,
             $S \equiv 2S - R \pmod p, R \equiv 2R \pmod p$
13.    else if $d = 2$ then
14.        If $U > \frac{V}{2}$ then $U = \frac{U-\frac{V}{2}}{2}, V = \frac{V}{2}$,
             $R \equiv 2R - S \pmod p, S \equiv 2S \pmod p$
15.        else $V = \frac{\frac{V}{2}-U}{2}$,
             $S \equiv S - 2R \pmod p, R \equiv 4R \pmod p$
16.    else
17.        If $U > V$ then $U = \frac{U-V}{2}$,
             $R \equiv R - S \pmod p, S \equiv 2S \pmod p, t = 1$
18.        else $V = \frac{V-U}{2}$,
             $S \equiv S - R \pmod p, R \equiv 2R \pmod p, t = 1$
19.    If $i < m$ then
20.        If $i = m - 1$ or $t = 1$ then
21.            If $\alpha_i = 1$ then $R \equiv R \pmod p, S \equiv S \pmod p$
22.            else $R \equiv \frac{R}{2} \pmod p, S \equiv \frac{S}{2} \pmod p$
23.        else
24.            If $(\alpha_{i+1}, \alpha_i) = (1, 1)$ then
                 $R \equiv R \pmod p, S \equiv S \pmod p$
25.            else if $(\alpha_{i+1}, \alpha_i) = (1, 0)$ or $(0, 1)$ then
                 $R \equiv \frac{R}{2} \pmod p, S \equiv \frac{S}{2} \pmod p$
26.            else
                 $R \equiv \frac{R}{4} \pmod p, S \equiv \frac{S}{4} \pmod p$
27.        $i = i + t$
28.    else $R \equiv \frac{R}{2^t} \pmod p, S \equiv \frac{S}{2^t} \pmod p$
29. Return $R$
```

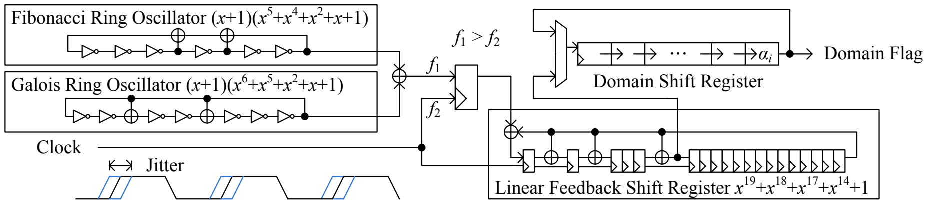

Fibonacci Ring Oscillator $(x+1)(x^5+x^4+x^2+x+1)$
Galois Ring Oscillator $(x+1)(x^6+x^5+x^2+x+1)$
Clock
Jitter
$f_1 > f_2$
f_1
f_2
α_i
Domain Flag
Domain Shift Register
Linear Feedback Shift Register $x^{19}+x^{18}+x^{17}+x^{14}+1$

Fig. 3. The domain flag is to randomly assign operating domain for GFAU

As the iterative operations in Algorithm 4 and Algorithm 5 are performed in one cycle, the critical path is to calculate the results of R or S consisting of the UV comparison with modular operations. For the modular division by 2 or 4 in Steps 5, 8, 11, 14 of Algorithm 4 and Steps 22, 25, 26, 28 of Algorithm 5, multiples of the prime p are added to enable the lowest part of R or S is zero so that they can be carried out by simple shift logic operator. Further, since the results of R, S are irrelevant to the results of operands U or V, a fully-pipelined stage can be inserted between the UV and RS data path to moderate the critical path. As the UV data path is determined, then the next cycle is to set the values of the operands R, S and simultaneously determine the next case until $V = 0$. Although one additional cycle is needed after pipelining, this is negligible as the operation takes hundreds or thousands of cycles. The timing flow of pipelined scheme is shown in Fig. 4. Besides, to reduce the hardware cost, symmetric modular operations such as $R \equiv (R - S)/2 \pmod p$ and $S \equiv (S-R)/2 \pmod p$ in Algorithm 4, $R \equiv (R-S)/4 \pmod p$ and $S \equiv (S-R)/4 \pmod p$ in Algorithm 5 can be executed by the same computational unit with a swap logic circuit, which is to switch the input operands of RS data path. In Algorithm 4, the RS data path can be classified into two groups: the first group includes Steps 4, 5 and Steps 10, 11; the second one consists of Steps 7, 8 and Steps 13, 14. In Algorithm 5, the two groups of RS data path are classified as follows: Steps 6, 9, 12, 15, and 18 belong in the first group; the second one consists of the others. The data flows of R and S are switched as the processing group is different from the group in previous cycle. Moreover, since the EC point calculation is a serial field operation, both of the temporary registers and modular operations can be shared for the operands V, S, R in Algorithm 2 and Algorithm 4 (or Algorithm 3 and Algorithm 5). These multiple modular operations in the iterative calculation can be effectively implemented by using a programmable data path of bit-level architecture, which consists of the carry-save adders with a carry-lookahead adder at last stage. The detailed radix-2 and radix-4 GFAU architecture is shown in Fig. 5 and Fig. 6, respectively.

5 Power Measurement and Implementation Results

Based on our proposed architecture using Montgomery ladder ECSM method, four different 160-bit DF-ECC processors with radix-2 and radix-4 algorithms are

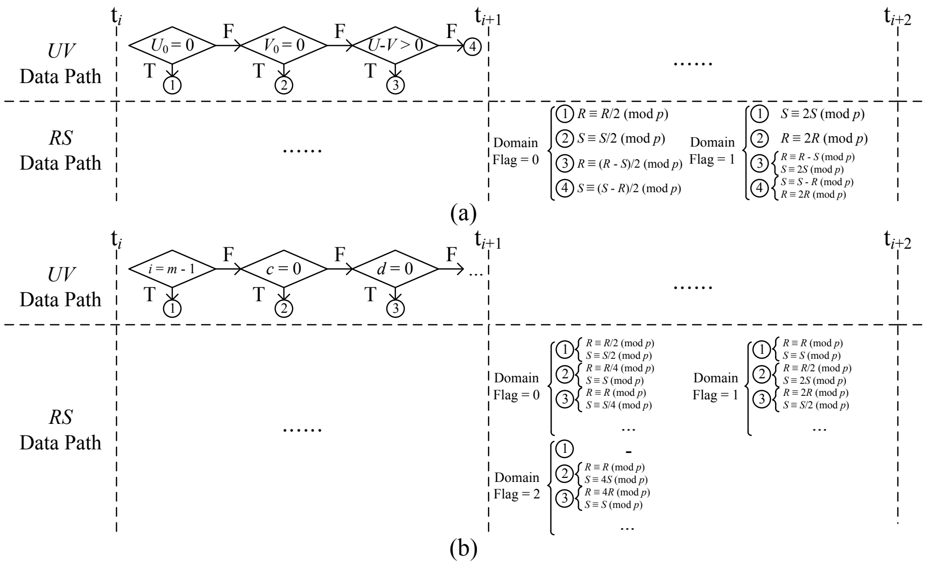


Fig. 4. Fully-pipelined scheme for the (a) radix-2 (b) radix-4 randomized Montgomery division

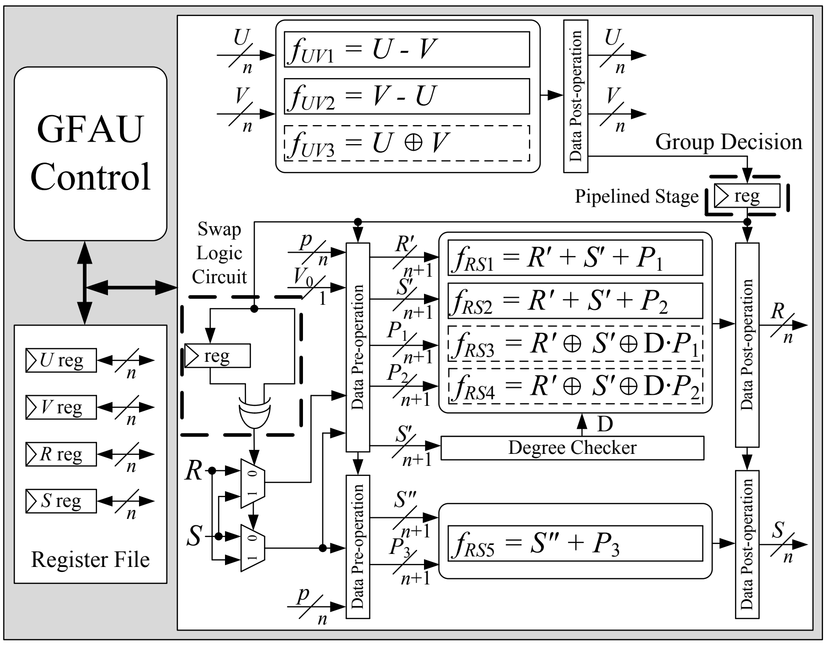


Fig. 5. Radix-2 GFAU

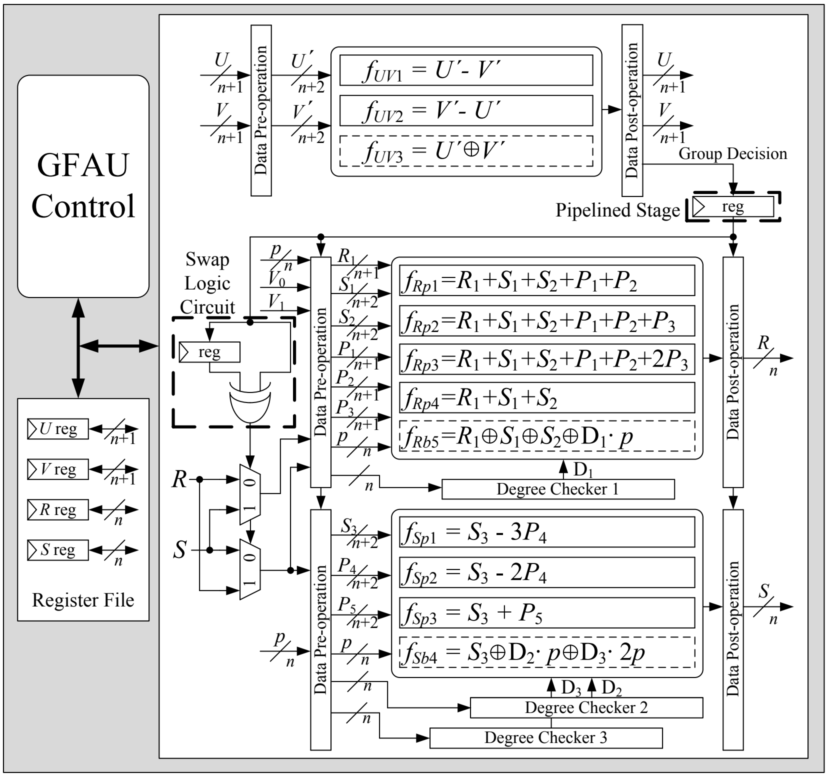


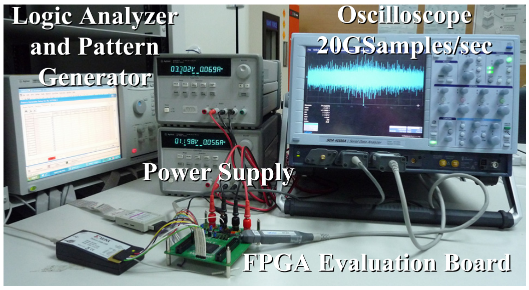
Fig. 6. Radix-4 GFAU

independently designed on an FPGA platform to evaluate the CPA resistance. The performance results are given in Table 3, and the verification environment is shown in Fig. 7.

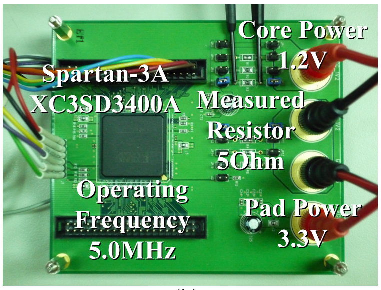
Table 3. FPGA Implementation Results

Design	Area (Slices)	$f_{\max}$ (MHz)	Field Arithmetic
I	7,573 (32%)	27.7	Radix-2 Montgomery
II	8,158 (34%)	27.7	Radix-2 Randomized Montgomery
III	9,828 (41%)	20.2	Radix-4 Montgomery
IV	10,460 (43%)	20.2	Radix-4 Randomized Montgomery

As shown in Algorithm 1, the point coordinate value P_2 is dependent on the bit value of the key in every iteration. Fig. 8(a) and Fig. 8(b) illustrate the CPA attacks on the unprotected Design-I and Design-III, respectively, using conventional Montgomery algorithm [21] to reveal the key value. The correlation


Logic Analyzer and Pattern Generator
Oscilloscope 20GSamples/sec
Power Supply
FPGA Evaluation Board

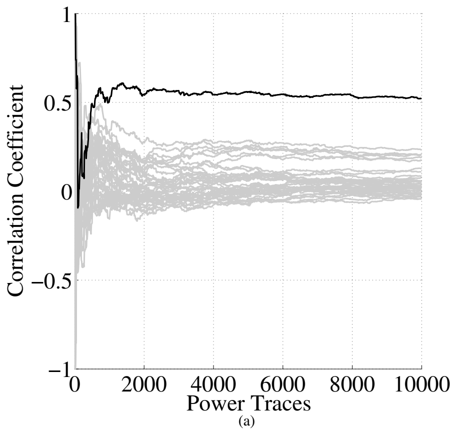
(a)



(b)

Fig. 7. (a) Environment of power measurement. (b) Current running through the DF-ECC processor recorded by measuring the voltage drop via a resistor in series with the board power pin and FPGA power pin.

coefficients for all possible hamming distances of the point coordinate P_2 are plotted over power traces, and that of the correct key hypothesis is plotted in black. In this case, as more than 10^3 power traces are used, the correlation of the correct key is the highest one among that of all the other key hypotheses, and then the key value can be found easily. However, even after collecting 10^6 power measurements from the Design-II and Design-IV using randomized Montgomery operations, the correlation coefficients of correct and incorrect hypothesis shown in Fig. 9 cannot be scattered, and they are near zero because the processed data are uncorrelated to power model. This means that there is no information bias of the key value extracted by the CPA attacks.



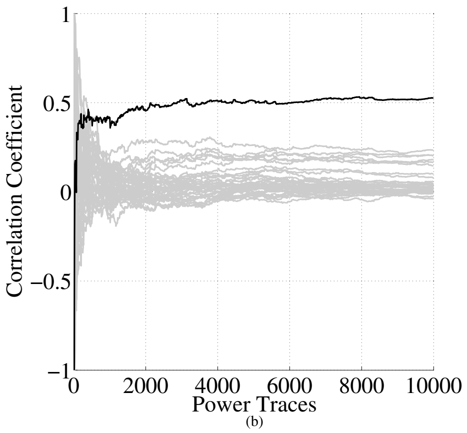


Fig. 8. Correlation coefficients of the target traces and power model over power traces obtained from the (a) Design-I (b) Design-III performing arithmetic in a fixed domain

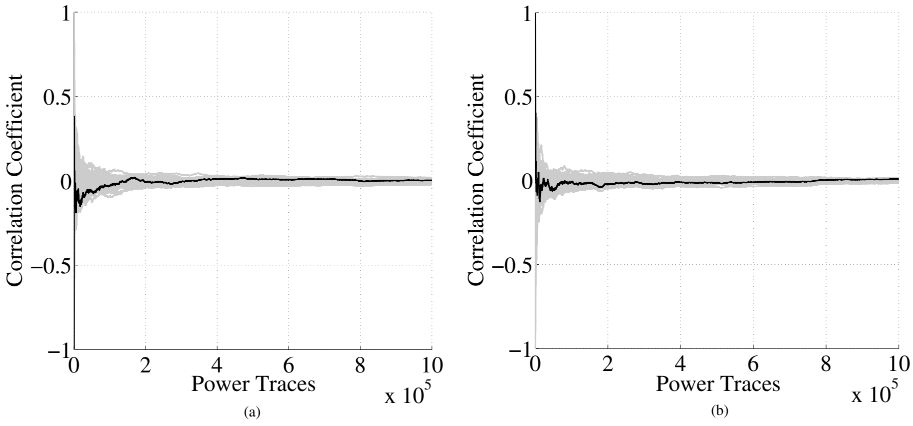


Fig. 9. Correlation coefficients of the target traces and power model over power traces obtained from the (a) Design-II (b) Design-IV performing arithmetic in a randomized domain.

Table 4. Implementation Results Compared with Related Works

	Technology	Field Length	Area (mm^2)	KGates	Galois Field	$f_{\max}$ (MHz)	Time (ms/ECSM)	Energy (μJ/ECSM)	AT Product
Ours (Radix-2)	90-nm	160	0.21	61.3	$GF(p_{160})$	277	0.71	11.9	1
					$GF(2^{160})$	277	0.61	9.6	1
Ours (Radix-4)	90-nm	160	0.29	83.2	$GF(p_{160})$	238	0.43	11.2	0.82
					$GF(2^{160})$	238	0.39	8.97	0.87
TCAS-II'09 [5]	0.13-μm	160	1.44	169	$GF(p_{160})$	121	0.61	42.6	1.63*
					$GF(2^{160})$	146	0.37	30.5	1.16*
Ours (Radix-2)	90-nm	521	0.58	168	$GF(p_{521})$	250	8.08	452	1
					$GF(2^{409})$	263	4.65	246	1
Ours (Radix-4)	90-nm	521	0.93	265	$GF(p_{521})$	232	4.57	435	0.89
					$GF(2^{409})$	238	2.77	238	0.94
ESSCIRC'10 [9]	90-nm	521	0.55	170	$GF(p_{521})$	132	19.2	1,123	2.40
					$GF(2^{409})$	166	8.2	480	1.78

* Technology scaled area-time product = Gates $\times$ (Time $\times$ t), where t = 90-nm/0.13-μm.

Our proposed DF-ECC processor was also implemented by UMC 90-nm CMOS technology, and the post-layout simulations for ASIC implementation with comparisons are given in Table 4. In terms of area-time product, our DF-ECC processor outperforms other approaches. By reducing the division iteration time and randomizing intermediate values in field arithmetic without increasing the key size, our work using radix-2 approach is at least 44% faster than the previous 521-bit design [9] with comparable hardware complexity. Compared with a four

Table 5. Overhead for CPA Resistance

	Ours (Radix-2)	Ours (Radix-4)	ESSCIRC'10 [9]	JSSC'06 [12]	JSSC'10 [13]
Design	521 DF-ECC	521 DF-ECC	521 DF-ECC	128 AES	128 AES
Area	4.3%	3.6%	10%	210%	7.2%
Time	0	0	14.0%[a]	288%	100%
Energy	5.2%	3.8%	20.8%[b]	270%	33%

Overhead $= \frac{\text{Result differences between protected and unprotected circuit}}{\text{Results of unprotected circuit}} \times 100\%$.

a. Estimated by cycle count×clock period.

b. Estimated by operation time×average power.

multipliers based ECC processor without power-analysis protection [5], our fully-pipelined and highly-integrated radix-4 GFAU architecture achieves competitive speed with 51% less gate counts.

For the CPA resistance, our approach is to mask the processed data uncorrelated with power traces without lengthening the hardware latency and without dominating the power consumption of key-dependent operations. From the comparison given in Table 5, our proposed countermeasure is superior to others not only in operation time but also in energy dissipation.

6 Conclusion

In this paper, we introduced a randomized dual-field Montgomery algorithm which is suitable for ECC hardware implementation against the CPA attacks. Without modifying logic circuit and without pre-computing data from host system, the relationship between target power traces and power model can be broken by performing the field arithmetic in a unpredictable operating domain. The proposed CPA countermeasure approach has been analyzed on an FPGA platform. Attacks on the unprotected designs reveal the private key within one thousand power traces, while the key value of the protected core cannot be extracted after one million power traces. Circuit overhead for randomly determining the operating domain can be integrated into the system without speed degradation. Implemented by an UMC 90-nm technology, our protected 521-bit DF-ECC processor using radix-4 randomized Montgomery operations, with 3.6% area and 3.8% average power overhead, can perform one $GF(p_{521})$ ECSM in 4.57ms and one $GF(2^{409})$ ECSM in 2.77ms. We believe that both high performance and efficient CPA countermeasure are achieved in our proposed DF-ECC processor.

Acknowledgments. The work described in this paper was supported by Taiwan's National Science Council (NSC) and Ministry of Economic Affairs (MOEA) under Grants NSC 100-2220-E-009-016, NSC 101-2220-E-009-060, and MOEA 100-EC-17-A-01-S1-180. The authors are grateful to United Microelectronics Corporation (UMC), Taiwan, for technology support. The authors would also like to thank Po-Ming Tu for his suggestion of FPGA power measurement, and anonymous reviewers for their constructive comments to improve the quality of this paper.

References

1. Koblitz, N.: Elliptic Curve Cryptosystems. Math. Comp. 48, 203–209 (2001)
2. Miller, V.S.: Use of Elliptic Curves in Cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
3. McIvor, C.J., McLoone, M., McCanny, J.V.: Hardware Elliptic Curve Cryptographic Processor over $GF(p)$. IEEE Trans. Circuits Syst. I 53(9), 1946–1957 (2006)
4. Sakiyama, K., Batina, L., Preneel, B., Verbauwhede, I.: Multicore Curve-Based Cryptoprocessor With Reconfigurable Modular Arithmetic Logic Units over $GF(2^n)$. IEEE Trans. Comput. 56(9), 1269–1282 (2007)
5. Lai, J.-Y., Huang, C.-T.: A Highly Efficient Cipher Processor for Dual-Field Elliptic Curve Cryptography. IEEE Trans. Circuits Syst. II 56(5), 394–398 (2009)
6. Chen, J.-H., Shieh, M.-D., Lin, W.-C.: A High-Performance Unified-Field Reconfigurable Cryptographic Processor. IEEE Trans. VLSI Syst. 18(8), 1145–1158 (2010)
7. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
8. Montgomery, P.: Speeding the Pollard and Elliptic Curve Methods of Factorization. Math. Comp. 48, 243–264 (1987)
9. Lee, J.-W., Chen, Y.-L., Tseng, C.-Y., Chang, H.-C., Lee, C.-Y.: A 521-bit Dual-Field Elliptic Curve Cryptographic Processor With Power Analysis Resistance. In: European Solid-State Circuits Conference (ESSCIRC 2010), pp. 206–209 (2010)
10. Brier, E., Clavier, C., Olivier, F.: Correlation Power Analysis with a Leakage Model. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 16–29. Springer, Heidelberg (2004)
11. IEEE: Standard Specifications or Public-Key Cryptography. IEEE Std. 1363 (2000)
12. Hwang, D., Tiri, K., Hodjat, A., Lai, B.-C., Yang, S., Schaumont, P., Verbauwhede, I.: AES-Based Security Coprocessor IC in 0.18-μm CMOS With Resistance to Differential Power Analysis Side-Channel Attacks. IEEE J. Solid-State Circuits 41(4), 781–792 (2006)
13. Tokunaga, C., Blaauw, D.: Securing Encryption Systems With a Switched Capacitor Current Equalizer. IEEE J. Solid-State Circuits 45(1), 23–31 (2010)
14. Liu, P.-C., Chang, H.-C., Lee, C.-Y.: A True Random-Based Differential Power Analysis Countermeasure Circuit for an AES Engine. IEEE Trans. Circuits Syst. II 59(2), 103–107 (2012)
15. Coron, J.-S.: Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 292–302. Springer, Heidelberg (1999)
16. Joye, M., Tymen, C.: Protections against Differential Analysis for Elliptic Curve Cryptography. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 377–390. Springer, Heidelberg (2001)
17. Montgomery, P.: Modular Multiplication Without Trial Division. Math. Comp. 44, 519–521 (1985)
18. Kaliski, B.: The Montgomery Inverse and Its Applications. IEEE Trans. Comput. 44(8), 1064–1065 (1995)

19. Cohen, H., Miyaji, A., Ono, T.: Efficient Elliptic Curve Exponentiation Using Mixed Coordinates. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 51–65. Springer, Heidelberg (1998)
20. Golic, J.D.: New Methods for Digital Generation and Postprocessing of Random Data. IEEE Trans. Comp. 55, 1217–1229 (2006)
21. Chen, Y.-L., Lee, J.-W., Liu, P.-C., Chang, H.-C., Lee, C.-Y.: A Dual-Field Elliptic Curve Cryptographic Processor With a Radix-4 Unified Division Unit. In: IEEE Int. Symp. on Circuits Syst. (ISCAS 2011), pp. 713–716 (2011)