Overcoming Glitches and Dissipation Timing Skews in Design of DPA-Resistant Cryptographic Hardware

Kuan Jen Lin[+], Shan Chien Fang, Shih Hsien Yang and Cheng Chia Lo
Department of Electronic Engineering, Fu Jen Catholic University, Taiwan
[+]kjlin @mail.fju.edu.tw

Abstract

Cryptographic embedded systems are vulnerable to Differential Power Analysis (DPA) attacks. In this paper, we propose a logic design style, called as Pre-charge Masked Reed-Muller Logic (PMRML) to overcome the glitch and Dissipation Timing Skew (DTS) problems in design of DPA-resistant cryptographic hardware. Both problems can significantly reduce the DPA-resistance. To our knowledge, the DTS problem and its countermeasure have not been reported. The PMRML design can be fully realized using common CMOS standard cell libraries. Furthermore, it can be used to implement universal functions since any Boolean function can be represented as the Reed-Muller form. An AES encryption module was implemented with multi-stage PMRML. The results show the efficiency and effectiveness of the PMRML design methodology.

1. Introduction

In 1998, Kocher et al. first reported that the power consumption of a smart card could reveal the secret key of the cryptographic algorithm [6]. The attack, called as *Differential Power Analysis* (DPA), has been considered as the most dangerous attack to the security of cryptographic embedded systems [9]. A recent report stated that the key of an unprotected AES coprocessor can be found in less than three minutes [14].

A lot of research has been conducted on corresponding countermeasures against the DPA attack. Most works can be broadly categorized into two classes: (1) equalizing power consumption or (2) randomizing power consumption. The first approach attempts to make the power consumption constant regardless of what intermediate result being produced. A straightforward method is to use the dual rail logic [7, 10, 12, 14], that represents a data bit by a pair of complementary wires and takes the same number of transitions regardless of what data value it transfers. The logic style is expected to consume constant power for any kind of input transitions. However, it can only be achieved if the complementary wires have the same capacitive load. The requirement is hard to meet in standard semi-custom design environment. Balanced cell design [7] and differential routing technique [14] are proposed to treat the problem. As the transistor size and wiring width continuously shrink, it becomes more difficult to apply such techniques. Besides, current results show that these approaches needs area at least 3 times larger than the standard CMOS implementation of an unprotected AES design [14].

The second countermeasure class aims at randomizing the intermediate results occurring during the computation of the cryptographic algorithm. A common approach is to use *masking*, which makes the output of a circuit unit unpredictable from the leaked information. Early masking approaches consider an arithmetic operation such as addition and multiplication as an atomic gate [1, 5, 16]. It is more realistic to provide protection on the basic gate level. A complete design of AES algorithm with masking at gate-level was presented in [15]. Universal masking for random logic was developed in [4]. However, these masking approaches were shown to be unable to resist DPA attacks in the presence of glitches [8, 13]. To overcome the glitch problem, the *MDPL* (Masking Dual-Rail Pre-charge Logic) design style was proposed [10]. Compared to an unprotected design, the area is increased by 350% and the speed is decreased by 42% in the MDPL design. The *RSL* (Random Switching Logic) [13] exploits timing control to suppress possible glitches. However, it needs to build a new standard cell library.

In this paper, we propose a logic design style, called as Pre-charge Masked Reed-Muller Logic (PMRML) to overcome the glitch and Dissipation Timing Skew (DTS) problems in design of DPA-resistant cryptographic hardware. To our knowledge, the DTS problem and its countermeasure have not been

978-3-9810801-2-4/DATE07 © 2007 EDAA

reported. The PMRML design can be fully realized using common CMOS standard cell libraries. Furthermore, a multi-stage pre-charge scheme is exploited to reduce the performance penalty caused by pre-charging time. An AES encryption module was implemented with the PMRML. The result shows the efficiency and effectiveness of the PMRML design methodology.

The next section firstly reviews the principles of the DPA attack and its corresponding countermeasures. Then the glitch and DTS problems are described. The PMRML design is described in Section 3. The hardware implementation of AES algorithm based on PMRML and related experimental results are described in Section 4. The final section draws conclusions.

2. Background

2.1 DPA attacks

A DPA attack on a cryptographic device begins by running the algorithm with N random inputs and collecting their power consumption curves (traces). Then an attacker selects a certain bit b, whose value can be calculated based on known plaintext (or cipher-text) and a subset of key, Ks. For example, some bit at the output of an SBOX operation in the first round of the AES algorithm. Given a guess of Ks, the N power traces can be split into two sets, $T_{b=1}$ and $T_{b=0}$, according to b=1 or b=0 at a certain time t. Then the means of the two trace sets are determined, which are referred as $E(T_{b=1})_t$ and $E(T_{b=0})_t$. All the possible values of Ks are used to partition the power traces. The mean difference, $DM_t(b)= |E(T_{b=1})–E(T_{b=0})|_t$, is likely to be maximal for the partition using the correct guess. If the bit width is manageable (|Ks|= 8 in the AES example), an attacker can find the correct Ks in a short time. The attack can be repeated to find remaining key values.

The power consumption of CMOS gates essentially is determined by the switching activity. If the power consumption of a circuit can be determined by observing the output switching, it is said to be an *atomic* gate. That is, we can use a constant ε_{xy} to denote the amount of power consumed during the Δt when $b=x$ at time $t-1$ and $b=y$ at time t. In this paper, we assume that basic gates such as NAND and XOR are atomic.

2.2 Masking on the Gate Level

The countermeasure using masking on the gate level aims at randomizing the intermediate results such that $DM_t(b)= 0$. In the approach, each of probably attacked signal b is represented by $b_m=b \oplus m_b$, where m_b is a uniformly distributed random variable (i.e. $p(m_b=0)=p(m_b=1)=1/2$) and is independent of b. Consequently, the b_m also is a uniformly distributed random variable. In the masking-approach, a circuit is replaced with a masked implementation, as shown in Fig. 1. For example, a 2-input XOR function $g=a \oplus b$ is replaced with $g_m= a_m \oplus b_m$ and $m_g= m_a \oplus m_b$. We refer to the implementation g_m as *M-XOR* gate. The m_g is called its correction mask. Besides of m_a and m_b, other mask signals may be used in the masked circuit. A masked implementation g_m of 2-input AND function $g= c \cdot d$ was proposed by [15] as follows:

$g_m=(c \cdot d) \oplus m_g = c_m \cdot d_m \oplus (m_c \cdot d_m) \oplus ((m_d \cdot c_m) \oplus m_g) \oplus (m_c \cdot m_d)$, where m_g is a new mask and independent of other signals. We refer to the implementation g_m as *M-AND* gate.

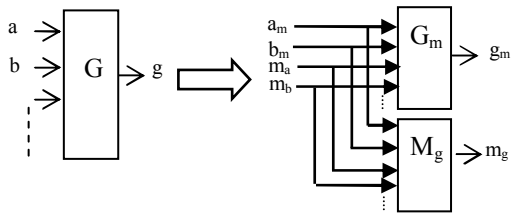

Figure 1: A circuit G is replaced with its corresponding masked circuit G_m and correction-mask circuit M_g, where $a_m=a \oplus m_a$, $b_m= b \oplus m_b$,…, $g_m=g \oplus m_g$.

Let the original circuit G have p different inputs and the masked replacement G_m have q different inputs. There is a mapping function $I: \{0, 1\}^p \rightarrow I_m: \{0, 1\}^q$. Let I^0 (I^1) denote the subset of I that sets G=0 (G=1). The I_m^0 (I_m^1) denote the set mapped from I^0 (I^1). The two set I_m^0 and I_m^1 may intersect. We give the following definition:

Definition 1 (Fully masked): A masked circuit G_m is a fully masked implementation of circuit G, if all possible input vectors to G_m are equally mapped from each of input vector to G. Namely, each $i \in I$ is mapped to each $i_m \in I_m$ with the same probability.

It can be easily proven by checking the truth table that a k-input M-XOR gate has such a property. But the 2-input M-AND is not. Note that the masked circuit ensures that no matter what g is, the corresponding masked output g_m has the property $p(g_m=1)=p(g_m=0)=1/2$. Hence, if the circuit G_m is atomic, it is expected to be DPA-resistant.

2.3 Glitches

A glitch is signal transitions before a gate switches to the correct outputs. Mangard et al. [8] and Suzuki et al. [13] showed that the glitches can lead to $DM_t(b) \neq 0$ in the M-AND circuit. Whether a glitch occur or not and the number of transitions it carries during a cycle both depend on the value of inputs and their order

arriving at the gate. This causes that the power consumption is correlated with the unmasked inputs. This offers a possibility to DPA attacks to find the key. The DPA-resistance is reduced.

To overcome the glitch problem, the MDPL [10] design style uses the return-to-zero signalling (achieved by pre-charging) to allow certain gates such as an AND gate having at most one transition during a clock cycle. In the MDPL, the dual rail majority-gate that implements a 2-input masked AND function is considered as an atomic gate.

Definition 2 (Glitch-safe): A masked circuit G_m is glitch-safe if it is glitch-free or a fully masked implementation of the original circuit G.

The second property in the above definition ensures that if an input vector to G_m can cause a glitch, it puts the same impact on each of input vector to G. Hence, the glitch effect is equally distributed to both sets $T_{b=1}$ and $T_{b=0,}$ thus counterbalancing with each other.

Table 1: DTS problem occurs in MDPL AND gate.

c · d	m $c_m d_m$ ($m \rightarrow c_m \rightarrow d_m$)	Masked Output ($(c \cdot d) \oplus m$)
00	000 →000 →000 →000	0 →0 →0 →0
	000 →100 →110 →111	0 →0 →1 →1
01	000 →000 →000 →001	0 →0 →0 →0
	000 →100 →110 →110	0 →0 →1 →1
10	000 →000 →010 →010	0 →0 →0 →0
	000 →100 →100 →101	0 →0 →0 →1
11	000 →000 →010 →011	0 →0 →0 →1
	000 →100 →100 →100	0 →0 →0 →0

2.4 Dissipation Timing Skews (DTS)

So far, we assume that the power dissipation of the attacked circuit can be observed within the interval Δt. However, if the arriving times of inputs vary largely and the time when the power dissipates for $T_{b=1}$ and $T_{b=0}$ is significantly different, this will cause that the $DM_t(b) \neq 0$ even if the amounts of power dissipated during a cycle for the two sets $T_{b=1}$ and $T_{b=0}$ are identical. In the remainder of this document, we refer to this phenomenon as *Dissipation Timing Skews (DTS)*. It can cause that the power consumption is correlated with the unmasked inputs. Let us use the 2-input MDPL-AND (i.e. a 3-input majority) gate to show the DTS effect. The three inputs to the gate are $c_m = c \oplus m$, $d_m = d \oplus m$ and mask m. Suppose that in the evaluation phase the inputs arrive the gate according to the order $m \rightarrow c_m \rightarrow d_m$. As shown in Table 1, the output transition (0→1) occurs at different moment of time for $c \cdot d = 0$ and $c \cdot d = 1$. If the DTS is large enough, it offers a possibility to DPA attacks to find the key. The use of MUX to realize masked circuits for AND (OR) gates in [4] also incur the same problem.

Given an input-arriving order: $i_1 \rightarrow i_2 \rightarrow \ldots \rightarrow i_p$, an output transition is denoted as t_k if it is activated after that i_k arrives. We give the following definition:

Definition 3 (DTS-safe): A masked circuit G_m is DTS safe if (1) it is a fully masked implementation of the original circuit G or (2) for each of possible input-arriving orders, the average number of t_k for I^0 is equal to that for I^1, where k=1,…, p.

In the example shown in Table 1, for the order $m \rightarrow c_m \rightarrow d_m$, we have $t_1=0$, $t_2=1/3$ and $t_3=1/6$ for I^0 and $t_1=0$, $t_2=0$ and $t_3=1/2$ for I^1. Hence, it is not DTS-safe. For a fully masked implementation, if an input vector to G_m can cause a DTS problem, it puts the same impact on each of input vector to G. Hence, the DTS effect is equally distributed to I^0 and I^1, thus counterbalancing with each other. The second property also ensures the counterbalance. The M-XOR has the following property.

Theorem 1: A k-input M-XOR gate is glitch-safe and DTS-safe.

We define the DPA-resistant circuit as follows:

Definition 4 (DPA-Resistant): A masked circuit implementation G_m of a circuit G is DPA-resistant if it is glitch-safe and DTS-safe.

3. Pre-charge Masked Reed-Muller Logic (PMRML)

All the Boolean functions can be represented as Fixed Polarity Reed-Muller (FPRM) form [2], in that a function is the XOR sum of cubes (ANDs) in which every variable has either positive or negative polarity. In the PMRML, a combinational logic function is realized as the FPRM form, as shown in Fig. 2. Currently, only 2-input AND gates are used in the AND-part. To provide DPA resistance, each AND gate is one-to-one replaced with its masked implementation, a 4X1 MUX with Dual-rail Selection signals (MUX-DS). We will show that the masked circuit is glitch-safe and DTS-safe under the scheme. In the XOR-part, each XOR gate is one-to-one replaced with the corresponding M-XOR circuit. The corresponding masks to recover the plain data are manipulated in the *correction masks generator*. Initial masks should come from a Random Number Generator (RNG), which is assumed to already be available to our design.

The pre-charge logic is used to ensure at most one transition at an AND (NAND) gate during a cycle. This makes the gates glitch-free. Though the pre-charge method has been used in several previous works [10, 14], there are two improvements in our work. Firstly,

only a subset of data is conveyed by dual-rail signals. Specifically, only the selection signals of MUX-DSs need them. Secondly, a multi-stage pre-charge scheme is exploited to reduce the performance penalty caused by pre-charging time.

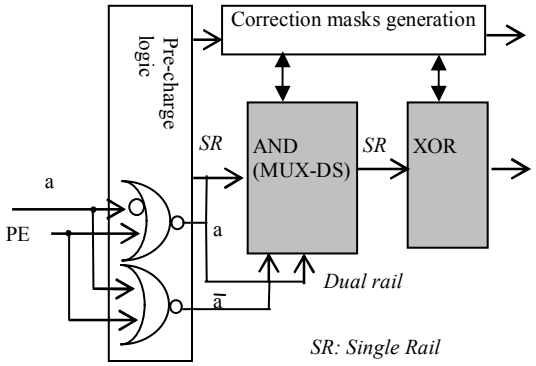

Figure 2: One stage PMRML.

The proposed masked circuit for an AND gate (g= c·d) is shown in Fig. 3, where the 4X1 MUX-DS is implemented by a 2-level NAND network and the selection variable is encoded with dual rail. In this work, NAND gate is assume to be atomic, but the whole MUX-DS circuit is not. The correction-mask circuit is $m_g=m_d m_e'+(d_m m_e)$, where m_e is a new mask. The following theorem provides the theoretical basis to use the 4X1 MUX-DS circuit, whose proof can be found in Appendix.

Theorem 2: The masked implementation of an AND function **g= c·d** shown in Fig. 3 is glitch-safe and DTS-safe when it is used in the PMRML design.

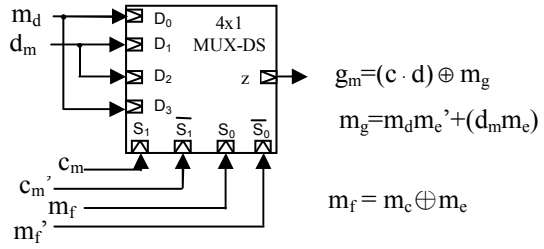

Figure 3: A masked implementation of an AND gate (g= c·d), where g_m =z= $(G_0' \cdot G_1' \cdot G_2' \cdot G_3')'$, in that $G_0=m_d c_m' m_f'$, $G_1=d_m c_m' m_f$, $G_2=d_m c_m m_f'$ and $G_3=m_d c_m m_f$.

Multi-stage PMRML design

The Fig. 4 (a) shows a 4-satge PMAXL structure. Each stage is controlled by separate PE (Pre-charge Enable) signals. All stages start pre-charge (PE_k=1) simultaneously, while disable pre-charge (PE_k=1), i.e. start evaluation, at different time. The timing diagram of PE signals is shown in Fig. 4(b). Let D_k denote the maximal circuit delay through the *stage k* logic, including the delays through pre-charge circuit (NOR + INV) and AND-XOR circuits. The duration of PE_k=1, T_k, must meet the following timing constraints:

(1) $\forall k \geq 1$, $T_k \geq D_k$.

(2) $\forall k > 1$, $T_k \geq T_{k-1} + D_{k-1}$.

The condition (1) ensures that all nodes in the stage *K* have returned to zero before starting next evaluation. The condition (2) ensures that before starting next evaluation, all input signals from stage *K-1* have set stable. Consequently, we have the minimal cycle time equals $T_1 + D_1 + ... + D_k + R_t$, where R_t is the delay of the register. Currently, the partition and timing control are done manually.

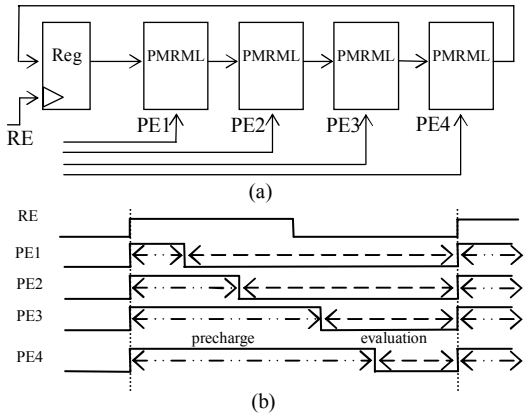

Figure 4: (a) A multi-stage PMRML network and (b) the timing diagram of its PE signals.

4. AES Hardware Implementation

The *SubBytes* transformation (SBOX) involving multiplicative inverse in $GF(2^8)$ is the most complicated operation in the AES algorithm. There have been many works that used composite field $GF((2^4)^2)$ arithmetic to reduce the hardware complexity, [11, 15, 17]. We use similar logic functions as proposed in these works. Then the PMRML is applied to make the circuit DPA-resistant. The Fig. 5 shows the PMRML-based SBOX implementation. On the left-hand part, the circuit is for correction-mask generation. The right-hand part is the data path for the masked data. The δ(x) means the mapping function from a ∈ $GF(2^8)$ to a polynomial $a_h x+a_l$ with coefficients in $GF(2^4)$. The masks m_{e*} are used for MUX-DS circuits, which are different from the current correction masks,. The SBOX is split into 3 stages. The whole design for AES encryption with 128-bit key length and 10-round computation is depicted in Fig. 6. Furthermore, 4 SBoxes are used in the encryption data path.

The proposed AES encryption hardware design was successfully synthesized by Synopsys DC with

conservative wire load model under UMC 0.18um technology. Compared to the unprotected design, as shown in Table 2, the area is increased by 100% and the speed is decreased by 29% in the PMRML design. Comparison with other approaches is shown in Table 3. Because the technology and the architecture (e.g. the number of SBOXs) used in those approaches vary, the comparison of the gate count and the performance can not fairly indicate the advantages. Therefore, we show the ratio of those DPA-resistant designs to the unprotected one in each own design environment. Although pre-charge scheme is used in the PMAXL design, the multi-stage schemes make the speed not halved.

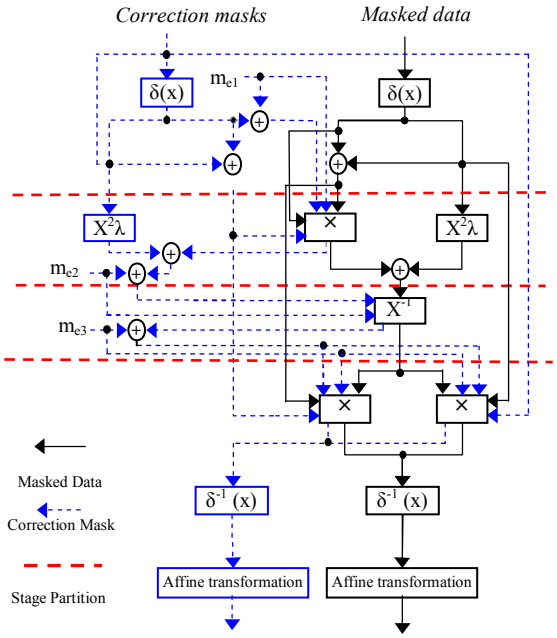

Figure 5: The SBOX implementation.

5. Conclusion

In this paper, we have proposed the PMRML design style to overcome the glitch and Dissipation Timing Skew (DTS) problems in design of DPA-resistant cryptographic hardware. Both problems can significantly reduce the DPA-resistance. To our knowledge, the DTS problem and its countermeasure have not been reported. The PMRML design can be fully realized using common CMOS standard cell libraries. Furthermore, a multi-stage pre-charge scheme is exploited to reduce the performance penalty caused by pre-charging time. The PMRML can be used to implement universal functions since any Boolean function can be represented as the Reed-Muller form. An AES encryption module was implemented with the PMRML. The results have shown the efficiency and effectiveness of the PMRML design methodology.

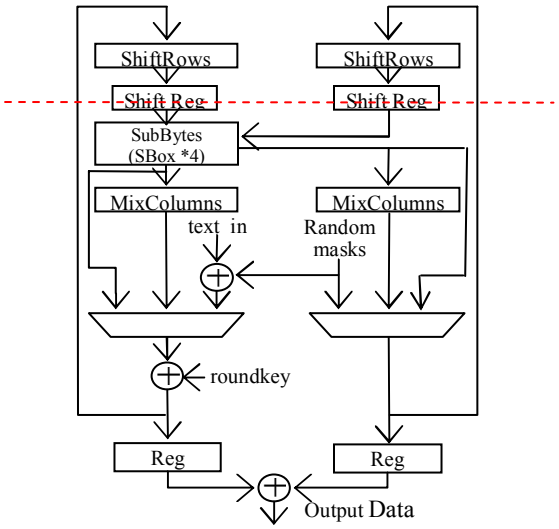

Figure 6: The PMRML design of the AES encryption module using 4 SBoxes.

Table 2: Comparison between the PMRML design and unprotected design.

	Area (gate count)	Performance* (Mbps)
unprotected	10.1k	156
PMRML	20.1 k[+]	111
PMRML/unprotected	2	0.71

[+] Does not include RNG circuit.

* Mbps = 128 / (cycle time x 41)

Table 3: Comparison of area (performance) ratio of various DPA-resistant designs to the unprotectedone.

	Area	Performance
Unprotected	1	1
M-AND [15]*	1.84	0.93
WDDL [14][+]	3	0.26
MDPL [10][+]	4.54	0.58
PMRML	2	0.71

*Does not deal with both glitch and DTS problems.

[+] Does not deal with the DTS problem.

6. Acknowledgments

The authors would like to thank the anonymous reviewers for their comments to improve the quality of this paper. This work was supported by Taiwan NSC under Contract No. NSC 95-2221-E-030-023.

7. References

[1] M. L. Akkar and C. Giraud, “An Implementation of DES and AES, Secure against Some Attacks,”

CHES2001, LNCS, vol. 2162, pp. 309-318, 2001.

[2] M. Davio, J.P. Deschamps, and A. Thayse, "*Discrete and Switching Functions*," McGraw-Hill Int'l, 1978.

[3] W. Fischer and B. M. Gammel: Masking at Gate Level in the Presence of Glitches, CHES 2005, LNCS 3659, pp. 187-200, 2005.

[4] J. D. Golić and R. Menicocci, "Universal Masking on Logic Gate Level," *Electronics Letters* 40(9), pp. 526–527, 2004.

[5] J. D. Golic and C. Tymen: Multiplicative Masking and Power Analysis of AES, CHES002, LNCS, vol. 2523, pp. 198-212, 2003.

[6] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," *Advances in Cryptology – CRYPTO '99, LNCS,* vol. 1666, pp. 388-397, 1999.

[7] K. J. Kulikowski, M. Su, A. B. Smirnov, A. Taubin, M. G. Karpovsky and D. MacDonald, "Delay Insensitive Encoding and Power Analysis: A Balancing Act," *ASYNC 2005*, pp. 116-125, 2005.

[8] S. Mangard, T. Popp, and B. Gammel, "Side-Channel Leakage of Masked CMOS Gates", *CT-RSA, LNCS 3376*, pp. 351-365, 2005.

[9] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the thread of power analysis attacks," *IEEE TC*, vol. 51, no. 5, pp. 541-552, 2002.

[10] T. Popp and S. Mangard, "Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints," *CHES 2005*, pp.172-186, 2005.

[11] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. R. Rao, and P. Rohatgi, "Efficient Rijndael Encryption Implementation with Composite Field Arithmetic," *CHES 2001, LNCS*, vol. 2162 , pp. 171-184, 2001.

[12] H. Saputra, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, R. Brooks, S. Kim and W. Zhang: Masking the Energy Behavior of DES Encryption, Design, Automation and Test in Europe Conference and Exhibition, pp. 84-89, 2003.

[13] D. Suzuki, M. Saeki, and T. Ichikaw: Random Switching Logic: A Countermeasure against DPA based on Transition Probability, Cryptology ePrint Archive (http://eprint.iacr.org/), Report 2004/346, 2004.

[14] K. Tiri, D. Hwang, A. Hodjat, B. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "A Side-Channel Leakage Free Coprocessor IC in 0.18µm CMOS for Embedded AES-based Cryptographic and Biometric Processing", *DAC*, June 2005.

[15] E. Trichina and T. Korkishko, "Secure AES Hardware Module for Resource Constrained Devices," *ESAS 2004, LNCS,* vol. 3313, pp. 215-229, 2005.

[16] E. Trichina, D. D. Seta, and L. Germani: Simplified Adaptive Multiplicative Masking for AES, CHES 2002, LNCS, vol. 2523 , pp. 187-197, 2003.

[17] X. Zhang and K. K. Parhi: High-Speed VLSI Architectures for the AES Algorithm, IEEE Transactions on VLSI Systems, vol. 12, Issue 9, pp. 957-967, Sept. 2004.

Appendix

Proof (Theorem 2): The circuit $g_m = z = (G_0' \cdot G_1' \cdot G_2' \cdot G_3')'$, where $G_0 = m_d c_m' m_f'$, $G_1 = d_m c_m' m_f$, $G_2 = d_m c_m m_f'$, $G_3 = m_d c_m m_f$. It is implemented with a 2-level NAND network. According to the truth table as shown in Table 4, the circuit is proven to be a masked implementation of $g = c \cdot d$ function. Namely, $g = g_m \oplus m_g$.

Each NAND gate (i.e. G_0',…, G_3')in the circuit produces at most one transition in the evaluation phase. Hence, all are glitch-free.

Let $I_m(c, d)$ denote the input vector set to the masked circuit mapped from the plain input vector (c, d) to the original circuit, where $(c, d) \in \{00, 01, 10, 11\}$. All the $I_m(0, 0)$, $I_m(0, 1)$, $I_m(1, 0)$ and $I_m(1, 1)$ set one transition ($1 \rightarrow 0$) on G_0, G_1, G_2 and G_3. This means that for any possible input-arriving order, the average number of transition t_k (as stated in Definition 3) is the same for all the $I_m(0, 0)$, $I_m(0, 1)$, $I_m(1, 0)$ and $I_m(1, 1)$. Therefore, the masked circuit is DFS-safe.

The output of G_i has the same probability: $p(G_i=0=1/8)$ and $p(G_i=0=7/8)$ for the $I_m(0, 0)$, $I_m(0, 1)$, $I_m(1, 0)$ and $I_m(1, 1)$. Hence, it does not correlated to any plain input. Each is also glitch-free and DTS-safe.

In summary, the circuit is glitch-safe and DTS-safe.

Table 4: The truth table of the masked circuit of an AND gate shown in Fig. 3

c d	m_c m_d m_e	c_m d_m m_f	G_0	G_1	G_2	G_3	g_m	m_g	g
0 0	0 0 0	0 0 0	1	1	1	1	0	0	0
0 0	0 0 1	0 0 1	1	1	1	1	0	0	0
0 0	0 1 0	0 1 0	0	1	1	1	1	1	0
0 0	0 1 1	0 1 1	1	0	1	1	1	1	0
0 0	1 0 0	1 0 1	1	1	1	1	0	0	0
0 0	1 0 1	1 0 0	1	1	1	1	0	0	0
0 0	1 1 0	1 1 1	1	1	1	0	1	1	0
0 0	1 1 1	1 1 0	1	1	0	1	1	1	0
0 1	0 0 0	0 1 0	1	1	1	1	0	0	0
0 1	0 0 1	0 1 1	1	0	1	1	1	1	0
0 1	0 1 0	0 0 0	0	1	1	1	1	1	0
0 1	0 1 1	0 0 1	1	1	1	1	0	0	0
0 1	1 0 0	1 1 1	1	1	1	1	0	0	0
0 1	1 0 1	1 1 0	1	1	0	1	1	1	0
0 1	1 1 0	1 0 1	1	1	1	0	1	1	0
0 1	1 1 1	1 0 0	1	1	1	1	0	0	0
1 0	0 0 0	1 0 0	1	1	1	1	0	0	0
1 0	0 0 1	1 0 1	1	1	1	1	0	0	0
1 0	0 1 0	1 1 0	1	1	0	1	1	1	0
1 0	0 1 1	1 1 1	1	1	1	0	1	1	0
1 0	1 0 0	0 0 1	1	1	1	1	0	0	0
1 0	1 0 1	0 0 0	1	1	1	1	0	0	0
1 0	1 1 0	0 1 1	1	0	1	1	1	1	0
1 0	1 1 1	0 1 0	0	1	1	1	1	1	0
1 1	0 0 0	1 1 0	1	1	1	0	1	0	1
1 1	0 0 1	1 1 1	1	1	1	1	0	1	1
1 1	0 1 0	1 0 0	1	1	1	1	0	1	1
1 1	0 1 1	1 0 1	1	0	1	1	1	0	1
1 1	1 0 0	0 1 1	1	1	0	1	1	0	1
1 1	1 0 1	0 1 0	1	1	1	1	0	1	1
1 1	1 1 0	0 0 1	1	1	1	1	0	1	1
1 1	1 1 1	0 0 0	0	1	1	1	1	0	1

$m_f = m_c \oplus m_e$