A Design Methodology for a DPA-Resistant Cryptographic LSI with RSL Techniques

Minoru Saeki[1], Daisuke Suzuki[1,2], Koichi Shimizu[1], and Akashi Satoh[3]

[1] Information Technology R&D Center, Mitsubishi Electric Corporation
Kamakura Kanagawa, Japan
{Saeki.Minoru@db, Suzuki.Daisuke@bx, Shimizu.Koichi@ea}
.MitsubishiElectric.co.jp

[2] Graduate School of Environmental and Information Sciences,
Yokohama National University
Yokohama Kanagawa, Japan

[3] Research Center for Information Security,
National Institute of Advanced Industrial Science and Technology (AIST)
Chiyoda Tokyo, Japan
akashi.satoh@aist.go.jp

Abstract. A design methodology of Random Switching Logic (RSL) using CMOS standard cell libraries is proposed to counter power analysis attacks against cryptographic hardware modules. The original RSL proposed in 2004 requires a unique RSL-gate for random data masking and glitch suppression to prevent secret information leakage through power traces. However, our new methodology enables to use general logic gates supported by standard cell libraries. In order to evaluate its practical performance in hardware size and speed as well as resistance against power analysis attacks, an AES circuit with the RSL technique was implemented as a cryptographic LSI using a 130-nm CMOS standard cell library. From the results of attack experiments that used a million traces, we confirmed that the RSL-AES circuit has very high DPA and CPA resistance thanks to the contributions of both the masking function and the glitch suppressing function. This is the first result demonstrating reduction of the side-channel leakage by glitch suppression quantitatively on real ASIC.

1 Introduction

Since Kocher et al. proposed side-channel attacks [1, 2], many countermeasures such as data masking and power equalization techniques have been studied and implemented as FPGA and ASIC circuits. However, most of them require custom logic gates and/or design tools specialized to their methods, or implementations using standard libraries can be compromised by glitch signals, unbalanced signal delays and power consumptions. A DPA countermeasure called Random Switching Logic (hereafter RSL) [3] is a strong countermeasure that combines a masking function using random numbers and a function that suppresses glitch signals. The original authors further improve RSL in [4] by adding re-mask processing to each RSL gate to counter high-order DPA [5, 6] and such DPAs as identify random numbers for each waveform with the use of filtering [7, 8]. RSL is, however, difficult to develop under a general environment compared with other countermeasures like Wave Dynamic Differential Logic (WDDL) [9] since it uses a logic gate that does not exist in standard CMOS cell libraries. In contract, we propose a design methodology of RSL that enables implementation in a standard CMOS library, and show the design flow using a standard design tool. An AES circuit with the RSL countermeasure is designed and implemented on a cryptographic LSI using a 130-nm CMOS standard cell library [10], and its hardware performances in gate counts and operating speed are evaluated. DPA and CPA experiments are also applied on the LSI, and the effectiveness of our methodology as DPA and CPA countermeasure is demonstrated. In addition, the effects of the RSL function that suppresses glitches is detailed in the evaluation of the power analysis resistance.

2 RSL using Standard Cell and Security Evaluation

RSL is a DPA countermeasure at the transistor level using a custom RSL gate that consists of a majority logic gate with output-control signal shown in Fig. 1. Table 1 shows a pseudo code of NAND operation by RSL. Random numbers are used to mask data and then signal transitions of the RSL gate lose correlation with selection functions of DPA. However, a simple random masking without signal delay control may cause glitches that do have correlation with the selection functions [3, 11]. In order to prevent the glitches, the RSL gate manages the delay relationship of the output-control signal (en), input signals (x_z,y_z), and random mask signals (r_z). Re-mask operation is independently performed in each RSL gate so that high-order DPAs [5, 6] cannot identify the random numbers in each waveform even using filtering techniques [7, 8]. For more details about RSL, refer to [4].

This section proposes a new method to realize RSL using only a standard CMOS cell library, while the original RSL scheme using the custom gate requires high development cost and long design period. Fig. 2 shows an RSL NAND logic using standard CMOS gates, a Majority-Inverter (MAJI) or OR-AND-Inverter (OAI222), and a NOR for output control, which as a whole provide compatible operation with the RSL-NAND gate in Fig. 1. We call this compatible logic “pseudo RSL”. The majority operation needs to be performed at the front step, and the output-control signal is enabled at the final step to suppress glitch signals on the output of the pseudo RSL gate.

According to [12], side-channel leakage occurs in non-linear data operations and is amplified in linear/non-linear operations. We hence estimate side-channel leakage of the MAJI logic in Fig. 2, which operates non-linearly on data inputs, using the leakage models mentioned in [12]. The change of the MAJI gate inputs (x, y, r) at the i-th cycle is denoted as

$$(x_{i-1}, y_{i-1}, r_{i-1}) \rightarrow (x_i, y_i, r_i).$$

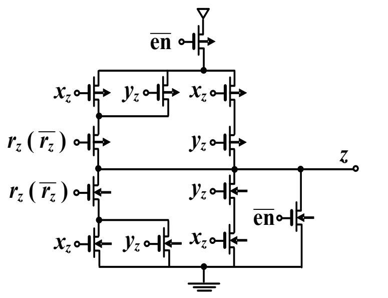


Fig. 1. An RSL-NAND(NOR) gate

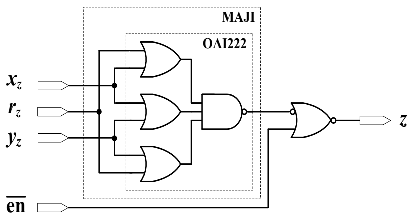


Fig. 2. NAND operation by pseudo RSL

Table 1. Pseudo-code of NAND operation by RSL

```
NAND operation by RSL
input en, $x = a \oplus r_x, y = b \oplus r_y$,
        $r_{xz} = r_x \oplus r_z, r_{yz} = r_y \oplus r_z, r_z$
output $z = \overline{a \cdot b} \oplus r_z$
begin
/*Operation 1: Suppress glitches */
 en <= 0;
/*Operation 2(a): Re-mask $x$ */
 $x_z$ <= $x \oplus r_{xz}$ ($= a \oplus r_z$)
/*Operation 2(b): Re-mask $y$ */
 $y_z$ <= $y \oplus r_{yz}$ ($= b \oplus r_z$)
/*Operation 3: Input data
to the RSL gate */
 $z$ <= RSL-NAND($x_z$, $y_z$, $r_z$, en)
/*Operation 4: The enable signal en
              rise after all other
              input signals are fixed */
 en <= 1 after max_delay($x_z$, $y_z$, $r_z$);
end
```

As the selection functions of DPA, we use $a_i (= x_i \oplus r_i), b_i (= y_i \oplus r_i)$, which are unmasked signals of x_i, y_i at the i-th cycle. The average number of signal transitions, $N(i)$ and $N(i+1)$, including glitches caused by the input signals (x, y, r) at MAJI gate, are evaluated. The transitions for two (the i-th and $(i+1)$-th) cycles are used to assess the bias of the numbers based on the cycle when an operation including the DPA selection function takes place. Table 2 shows the transition counts of the MAJI gate for each input pattern in each delay condition. For example, in the delay condition 1, when $(a_i, b_i, r_i) = (1, 0, 0)$, the average number of signal transitions during $(x_{i-1}, y_{i-1}, r_{i-1}) \rightarrow (x_i, y_i, r_i)$ is $N(i) = 1$, and the number during $(x_i, y_i, r_i) \rightarrow (x_{i+1}, y_{i+1}, r_{i+1})$ is $N(i+1) = 1/2$. Next, the amount of leakage is calculated from Table 2 for each delay condition and selection function. The amount of leakage at the i-th cycle, $N_{\text{diff}}(i)$, is defined according to [12] as follows.

$$N_{\text{diff}}(i) = N_{\alpha=1}(i) - N_{\alpha=0}(i)$$

where $N_{\alpha=\delta}$ is the average number with the selection function being α and its value δ.

Table 3 shows the amount of leakage for each delay condition calculated from Table 2. Each condition includes cases where $N_{\text{diff}} \neq 0$, and thus pseudo RSL does not meet the security conditions described in [12] when it is strictly evaluated as a single gate. However, leakage model in [12] is inapplicable to pseudo RSL, which assumes the leakage is amplified by following gates. As stated

Table 2. Signal transitions of a MAJI gate

Delay condition 1			
delay(x) < delay(y) < delay(r)			
a_i, b_i, r_i	x_i, y_i, r_i	$N(i)$	$N(i+1)$
0 0 0	0 0 0	1/2	1/2
0 0 1	1 1 1	1/2	1/2
0 1 0	0 1 0	1	1
0 1 1	1 0 1	1	1
1 0 0	1 0 0	1	1/2
1 0 1	0 1 1	1	1/2
1 1 0	1 1 0	1/2	1
1 1 1	0 0 1	1/2	1

Delay condition 2			
delay(x) < delay(r) < delay(y)			
a_i, b_i, r_i	x_i, y_i, r_i	$N(i)$	$N(i+1)$
0 0 0	0 0 0	1/2	1/2
0 0 1	1 1 1	1/2	1/2
0 1 0	0 1 0	1/2	1
0 1 1	1 0 1	1/2	1
1 0 0	1 0 0	1	1/2
1 0 1	0 1 1	1	1/2
1 1 0	1 1 0	1	1
1 1 1	0 0 1	1	1

Delay condition 3			
delay(r) < delay(x) < delay(y)			
a_i, b_i, r_i	x_i, y_i, r_i	$N(i)$	$N(i+1)$
0 0 0	0 0 0	1/2	1/2
0 0 1	1 1 1	1/2	1/2
0 1 0	0 1 0	1/2	1
0 1 1	1 0 1	1/2	1
1 0 0	1 0 0	1	1
1 0 1	0 1 1	1	1
1 1 0	1 1 0	1	1/2
1 1 1	0 0 1	1	1/2

Table 3. Leakage amount of a MAJI gate

Delay condition 1		
delay(x) < delay(y) < delay(r)		
Cycle	Selection function	N_{diff}
i	a_i	0
i	b_i	0
$i+1$	a_i	0
$i+1$	b_i	1/2

Delay condition 2		
delay(x) < delay(r) < delay(y)		
Cycle	Selection function	N_{diff}
i	a_i	1/2
i	b_i	0
$i+1$	a_i	0
$i+1$	b_i	1/2

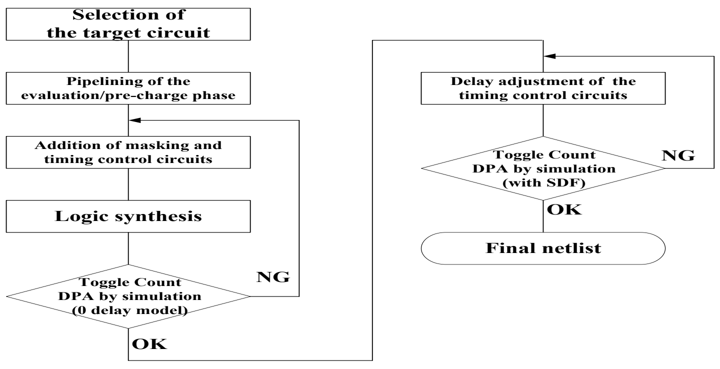
Delay condition 3		
delay(r) < delay(x) < delay(y)		
Cycle	Selection function	N_{diff}
i	a_i	1/2
i	b_i	0
$i+1$	a_i	0
$i+1$	b_i	0

before, pseudo RSL prevents the spread of the glitches after MAJI by the output-control signal, and hence the 1/2 transition bias of the MAJI gate in Table 3 does not propagate. Therefore, the amount of leakage in the entire circuit is at most $k/2$, where k is the the number of MAJI gates whose input bits are used as selection functions.

In short, pseudo RSL provides the same level of security as RSL if there exists a lower limit of $|N_{\text{diff}}|$ detectable by DPA, denoted as ϵ, such that $k/2 < \epsilon$. It is hard to give the threshold ϵ value, because it depends on the side-channel evaluation environment and the device characteristics, but it is easy to calculate the maximum of k. For instance, k is 2 for the AES circuit in Section 5. The circuit size is about 30 Kgates and the average number of signal transitions per cycle using virtual delay is approximately 15,000, which implies that if DPA fails to detect the bias of 1/15,000 transitions per DPA trace, pseudo RSL can make the circuit sufficiently secure.

3 Design Flow of Cryptographic Circuits using RSL

Fig. 3 shows the design flow of the RSL and pseudo RSL circuits, and Fig. 4 is the abstract of the hardware architecture. It is assumed that the circuit is



Selection of the target circuit
Pipelining of the evaluation/pre-charge phase
Addition of masking and timing control circuits
Logic synthesis
Toggle Count DPA by simulation (0 delay model)
NG
OK
Delay adjustment of the timing control circuits
Toggle Count DPA by simulation (with SDF)
NG
OK
Final netlist

Fig. 3. Overview chart of design flow

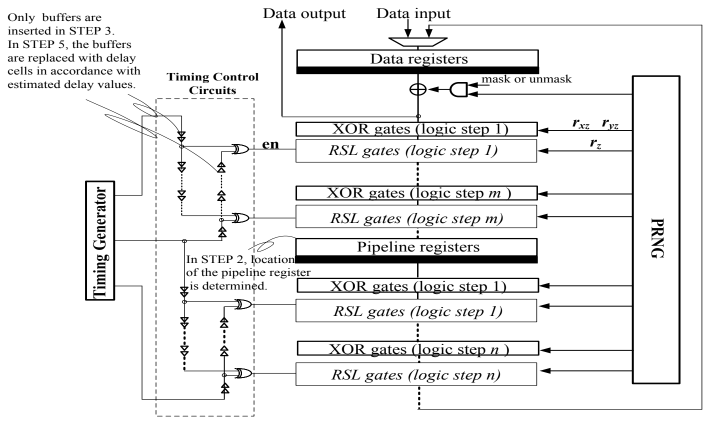


Fig. 4. Cryptographic circuit using RSL

designed in RTL and netlist is generated by a logic synthesis tool. In this flow, RSL or pseudo RSL circuit is generated from an existing cryptographic design without DPA countermeasure. The flow is largely made up of the following five steps.

STEP 1 Selection of the target circuit.
STEP 2 Pipelining of the evaluation/pre-charge phase.
STEP 3 Addition of masking and timing control circuits.
STEP 4 Logic synthesis.
STEP 5 Delay adjustment of the timing control circuit.

In STEP 1, a cryptographic circuit is selected to convert into RSL logics, but circuits that fall into the following two cases cannot be used. The first case is that the circuit data paths contain successive non-linear operations. For example, if SubBytes circuit of AES is implemented by AND-XOR-type 2-step logic such as

Positive Polarity Reed-Muller (PPRM), it contains high-order product terms. Then the precise delay control and random number re-masking for each 2-input NAND is extremely difficult. Integer arithmetic that includes successive non-linear operations such as addition and multiplication is not suitable either. In contrast, SubBytes implementation that employs composite field arithmetic [13] is very suitable because it has a nested structure of AND-XOR operations with a few non-linear steps. The second case is that the circuit is designed as behavior model or using a large look-up table description, because its logic structure depends on synthesis tools.

In STEP 2, a tentative logic synthesis is performed to evaluate delays in the whole circuit chosen in STEP 1. Afterward, a pipeline register is inserted to divide the circuit into two parts and thereby to let the RSL evaluation and pre-charge phases run in parallel. The register location is determined in consideration of the tradeoff between the operating frequency and the circuit area. A high frequency is expected by dividing the critical path at the center, but there may be a large number of intermediate signals that require a large register. In our prototypic LSI described in Section 5, the pipeline register is located at the output of SubBytes to minimize the number.

STEP 3 replaces logic gates for non-linear operations with RSL or pseudo RSL gates, and puts random number generators for data masking. Note that RSL need not be applied to linear operations such as XOR because they do not generate side-channel leakage as long as the inputs to them are masked. The replacement is applied only to the gates related to secret key operations. Fig. 5 shows a conversion example of a NAND gate, in which three random number input ports are added. In the case of a loop architecture, the initial data masking and the final data unmasking need an additional XOR circuit shown in Fig. 4. The above operations determine the number of random number generator and unmasking/re-masking circuits to be added. The timing control logic for output-control signals of each RSL gate is also designed. At this time, timing adjustment to prevent glitches is not required. In the implementation example of AES in Section 5, the number of RSL stages from the data register to the pipeline register is 4, but no RSL stage exists between the pipeline register to data register because there is only a linear transformation. Pseudo-random number generators using two 33-bit LFSR are designed in the AES circuit. Quality and characteristics of random numbers required by a masking method is an open question.

In STEP 4, ordinary logic synthesis is performed while the structures of RSL gates and the timing control circuits are protected not to be modified. In consideration of the increase of the critical path delay for timing adjustment in the next step, logic synthesis must be performed with some margin for performance in speed.

In STEP 5, the delay is adjusted by modifying the timing control logic part of the netlist created in STEP 4. First, the maximum delay before the RSL gates (logic step 1) located at the upper-most of Fig. 4 is extracted. The path of the output-control signal is not included in the delay. Next, temporary buffers in the timing control circuit are replaced to have delays longer than the extracted

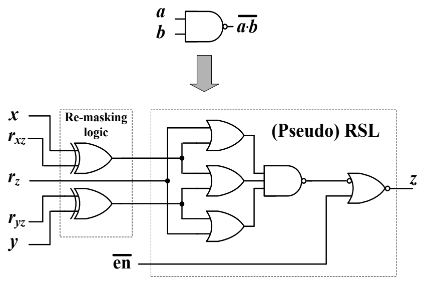


Fig. 5. RSL conversion for NAND Gate

Table 4. Our implementation environment

Process	TSMC 130nm CL013G [10]
Logic synthesis	Design Complier version 2004.12.SP4
Simulator	NC-Verilog version 05.40-p004

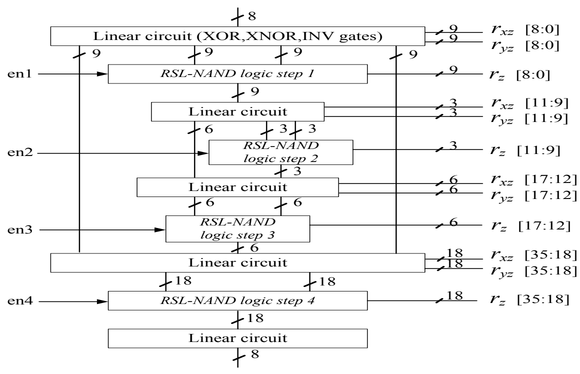


Fig. 6. SubBytes logic with composite field arithmetic

delay. This modification is repeated until timing simulation using the netlist and Standard Delay File (SDF) created in STEP 4 and 5, meets the security conditions of RSL.

4 Performance Evaluation of the Prototype Circuit

Table 4 summarizes our implementation environment applying the pseudo RSL scheme to the Verilog-HDL source code [14] designed with composite field arithmetic for the SubBytes function. It uses one round function block with a 16-byte SubBytes block for the loop architecture, and key expansion is processed on-the-fly using a 4-byte SubBytes block. The pseudo RSL is only applied to the 16-byte SubBytes block, which is the only non-linear function in the round operations. The composite gate OAI222 in the TSMC's 130-nm CMOS standard cell library CL013G is used as the majority logic. The pipeline register is placed at the output of SubBytes and is implemented as negative edge triggered flip-flops to have the same number of clock cycles as that of the original circuit without pseudo RSL. As described in the previous section, the number of RSL stages containing SubBytes before the pipeline register is 4, and numbers of the pseudo RSL

Table 5. Performance evaluation result

Evaluation item	Before applying [14]	After applying
Gate counts	14.5 Kgate	30.5 Kgate
Maximum delay	16.77 ns	14.77 ns
Maximum operation frequency	59.6 MHz	33.8 MHz
Processing performance (at f_{max})	763 Mbps	432 Mbps

Table 6. Our experimental environment

Parameters	Explanation
Target device	TSMC 130-nm cryptographic LSI on SASEBO-R
Operating frequency	24 MHz (standard setting on the board)
Measuring point	Resistance (2.2 Ω) between power supply and ASIC
Oscilloscope	Agilent DSO8104A
Sampling frequency	2 GHz
Number of power traces	1,000,000 traces

gates in each stage are 9, 3, 6, and 18. Therefore three 36-bit random numbers (r_z, r_{xz}, and r_{yz}) are needed for the mask operations. Fig. 6 summarizes this configuration.

Table 5 shows the performances of the AES circuits with and without the pseudo RSL. The experimental cryptographic LSI that contains these AES circuits is operated at a low frequency of 24 MHz, and hence speed constraints for the circuits were not specified to a logic synthesis tool. The throughput of 432 Mbps at 33.8 MHz of the pseudo RSL circuit can be applied to many embedded applications including smart cards. The gate counts are doubled to 30.5 Kgates, including all the components of RSL-AES such as the encryption block, key scheduler and pseudo-random number generator, but this number is small enough for any practical use. A simple performance comparison cannot be made between an AES circuit using pseudo RSL and the one using WDDL in [15] because their design environments are different. However, we believe that our pseudo RSL logic has high advantages over WDDL in both hardware size and speed because the AES circuit with WDDL in [15] requires about three times as many gates as and its operating speed is degraded down to 1/4 in comparison to the original AES circuit without WDDL.

5 Power Analysis Attack against Prototype LSI

Power analysis attacks against the pseudo RSL-AES circuit implemented on the experimental LSI is performed by using a circuit board SASEBO-R (Side-channel Attack Standard Evaluation Board, see Fig. 16 in Appendix A), which is specially designed for the side-channel attack experiments [16]. The evaluation environment is detailed in Table 6. The random masking and glitch suppressing functions in the pseudo RSL-AES circuit can be enabled separately by setting a mode register. The DPA and CPA attacks are performed on four possible combinations of the functions to reveal the final round key.

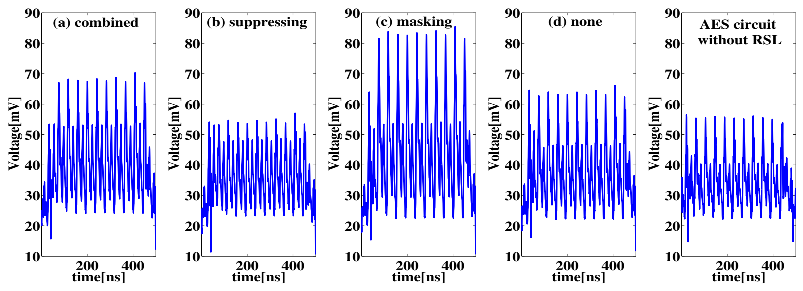


Fig. 7. Comparison of power traces

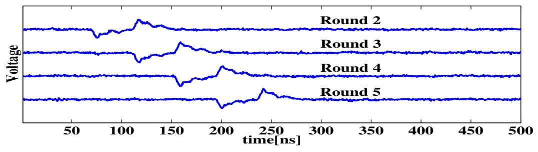


Fig. 8. Example of leakage analysis results

5.1 Comparison of the Power Traces

Fig. 7 shows the mean power traces of 1,000 encryptions in each operation mode: (a) "**combined**" enabling the both functions, (b) "**suppressing**" with only the glitch suppressing, (c) "**masking**" with only the random masking, and (d) "**none**" disabling the both functions. The original AES circuit without the pseudo RSL is also measured as the power trace in Fig. 7.

In all figures, large and relatively small peaks can be observed synchronizing rising and falling edges of system clock, respectively. In the pseudo RSL-AES circuit, by comparing the traces of **none** and **masking**, or those of **suppressing** and **combined**, it is observed that the peaks (power consumption) are increased by the mask operation. The glitch suppression works to reduce the peaks as can be seen in the same way by comparison of **none** and **suppressing**, or **masking** and **combined**.

The glitch suppression enables to reduce the increase of the peak current that the data masking causes. The decrease in peak current of circuits is an important issue for some devices such as contactless smart cards. We thus believe that RSL is suitable for those devices.

5.2 Leakage Analysis

Fig. 8 shows DPA traces in **none** with correct predictions for one input bit to SubBytes circuit in successive four rounds. The spikes in DPA traces appear at appropriate time frame corresponding to the target rounds, and thus they are information leakage but not noise signals. We call this evaluation using correct internal information known by evaluators "leakage analysis". Even if information leakage is confirmed by the leakage analysis, it does not mean the implementation is weak against power analysis attacks because the spikes for wrong keys

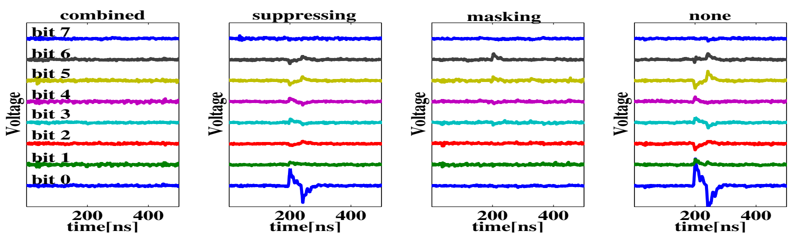


Fig. 9. Leakage analysis results with SubBytes input as selection function

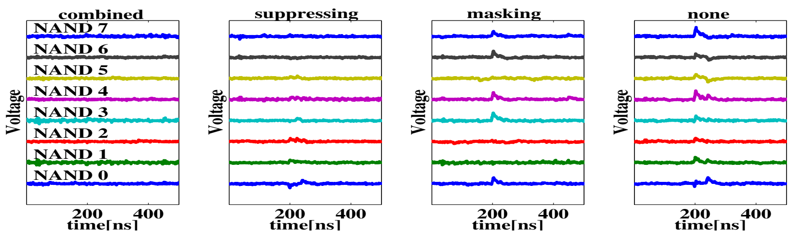


Fig. 10. Leakage analysis results with NAND gate input as selection function

(called ghost peaks) would be larger than correct ones [17], or key search space would be too large to attack while we know the correct key in the leakage analysis. Therefore, the analysis is the most powerful scheme to evaluate security of cryptographic modules against power analysis attacks.

Figs. 9 and 10 show the DPA traces by the leakage analysis of the RSL-AES circuit in four operation modes. Each input bit of one 8-bit S-box and each input bit of eight NAND gates in SubBytes circuit are used as selection functions. The effects of the random mask are not taken into account for the selection functions. In other words, random numbers are assumed to be 0 in the functions. The traces for **suppressing**, **masking** and **none** show spikes of information leakage while no leakage is observed in **combined**. The peak values of **suppressing** and **none** in Fig. 9 are larger than those in Fig. 10, but the values of **masking** in Fig. 9 is smaller than that of Fig. 10. Therefore, a selection function suitable for attacks varies depending on the circuit implementation. Both in Figs. 9 and 10, the peak values for **suppressing** and **masking** are smaller than those for **none** while their shapes are similar. These results clearly show that each countermeasure has some independent effect to reduce the leakage, but the pseudo RSL scheme that is a combination of the two countermeasures provides very high security.

5.3 CPA and DPA Attacks

CPA [17] and DPA attacks are performed using 1,000,000 power traces to reveal the final round key of the pseudo RSL-AES circuit in each mode. In the following discussion, the results for two out of sixteen 8-bit S-box circuits are displayed.

CPA results are shown in Fig. 11, where Hamming distance of a data register is used for the selection function. The vertical axis shows the logarithm plots

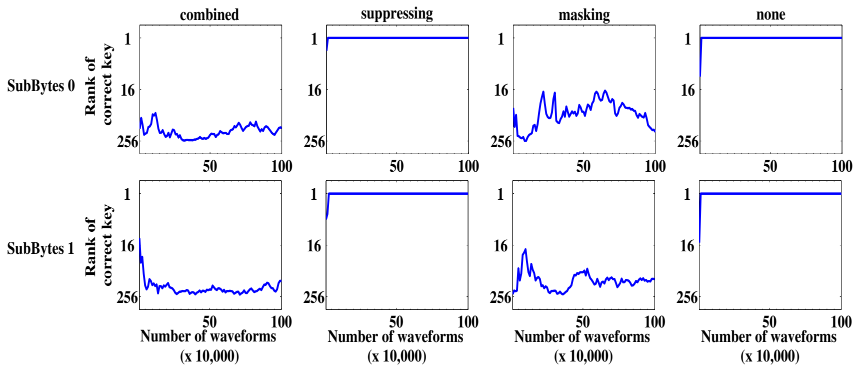


Fig. 11. CPA results of SubBytes 0-1

representing the ranks of a correct 8-bit partial key and the horizontal axis shows the number of traces. The correct keys for the two S-boxes were found with very few number of traces in **suppressing** and **none**, but the attack failed in **combined** and **masking** using the random mask. This is because the Hamming distance of the masked data cannot be estimated from the partial key and a ciphertext.

Single-bit DPA results are shown in Fig. 12, where each input bit of the 8-bit S-box is used as the selection function. Therefore, eight estimations for one correct partial key are shown in each graph. The correct key was identified by some selection functions in **suppressing** and **none** but the attack was completely failed in **combined**. There are some selection functions that lead relatively high rank for the correct key, but it would be difficult to identify the answer. In contrast, The DPA using NAND gate input as the selection function successfully revealed the correct key as shown in Fig. 13. These results are consistent with the leakage analysis in Figs. 9 and 10.

Multi-bit DPAs, extended version of DPA, are more powerful attacks [18, 19]. We explain the attack results by the multi-bit DPA in [19], which worked fine as an attack to our circuit. This is an attack method that uses the sum of absolute values of differential power of multi-bits. Fig. 14 shows the results of 8-bit DPA using the 8-bit input to the S-box circuit in the final round. In the same way as the single-bit DPA, the correct key was identified in **suppressing** and **none**, but not in **combined** and **masking**. The multi-bit DPA that generates only one DPA trace for each partial key makes the key estimation easier than the single-bit DPA that creates eight DPA traces that contain wrong keys. In Fig. 15, each input to eight NAND gates of the S-box circuit is used as a selection function. The results also show the correct key estimation more clearly than the single-bit DPA. However, the attack in **combined** still failed. It would be difficult to form a special selection function such as the input to the specific NAND gate without design information about the target device. However, the method presented in [20] enables us to seek an efficient selection function. Therefore, the glitch suppressing function in the pseudo RSL is important for pursuing sufficient security for power analysis attacks.

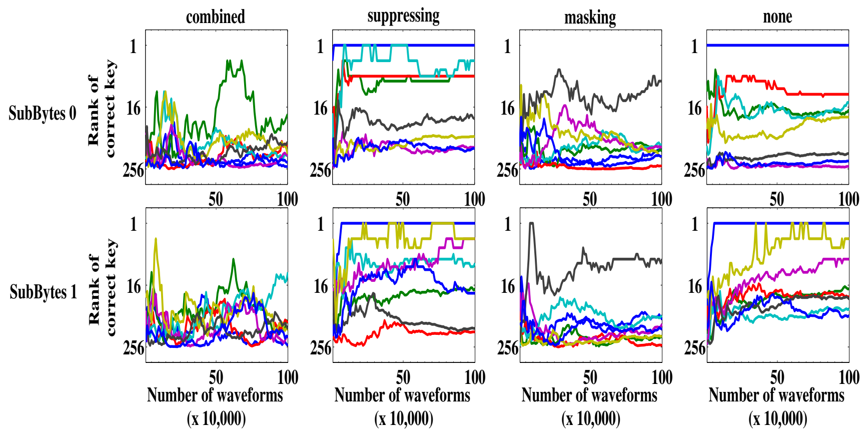


Fig. 12. 1-bit DPA results of SubBytes 0-1 with SubBytes input as selection function

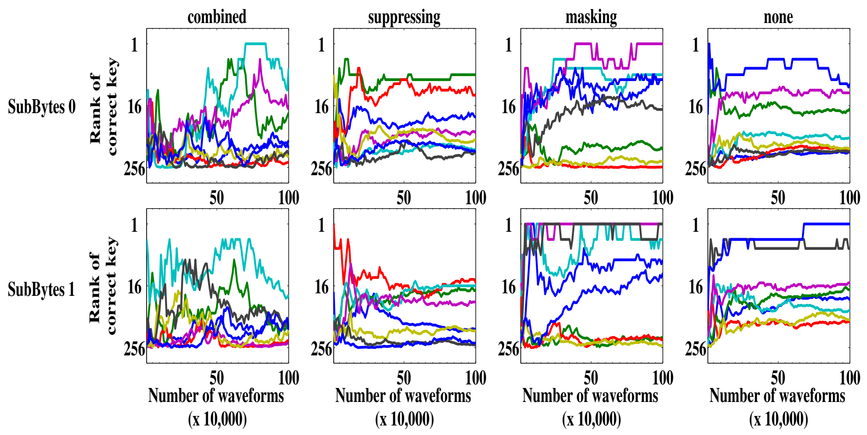


Fig. 13. 1-bit DPA results of SubBytes 0-1 with NAND gate input as selection function

6 Conclusion

In this paper, we proposed design methodology of RSL using CMOS standard libraries, which we call “pseudo RSL”. Conditions to guarantee security of the pseudo RSL are also discussed. AES circuits with (and without) the pseudo RSL gates are implemented on an experimental LSI using a 130-nm CMOS standard cell library according to the proposed design flow. The pseudo RSL scheme doubles the hardware resource and halves the operating speed in comparison with original AES circuit. However, this result shows high advantages in hardware performance over the WDDL scheme that needs three times as many gates and reduces the operating frequency down to 1/4. Various CPA and DPA attacks on the LSI controlling data masking and glitch suppressing function demonstrated that the pseudo RSL-AES circuit has a very high security against the attacks. We also confirmed that the AES circuit with the pseudo RSL has a very high

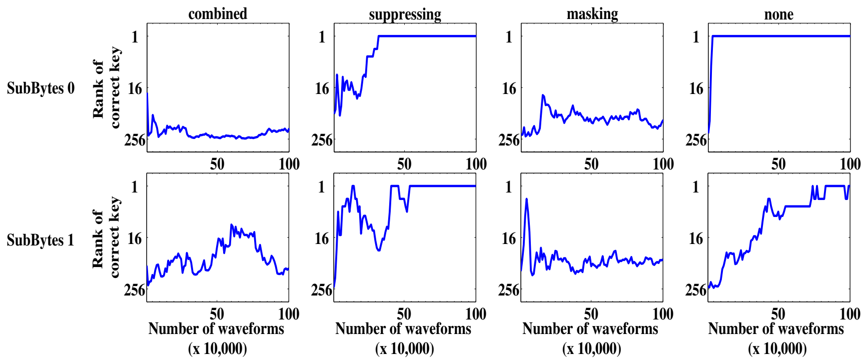


Fig. 14. 8-bit DPA results of SubBytes 0-1 with with SubBytes input as selection function

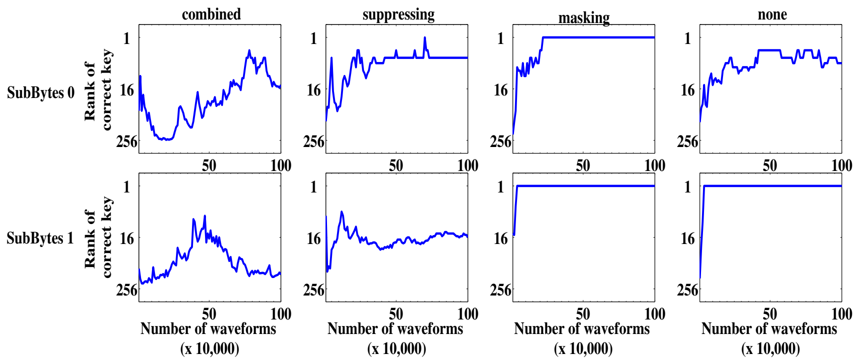


Fig. 15. 8-bit DPA results of SubBytes 0-1 with NAND gate input as selection function

power analysis resistance thanks to the contributions of the masking and the glitch suppressing functions. This is the first result demonstrating reduction of the side-channel leakage by glitch suppression quantitatively on real ASIC.

However, in the pseudo RSL-AES circuit, a specific 1-bit intermediate signal was found to be used as a selection function to identify a specific 1-byte partial key (See Appendix A). We have confirmed that some signal paths do not satisfy the delay conditions for pseudo RSL by static timing analysis using design tools. Therefore, new LSIs have been developed, where the detailed delay information to satisfy the security conditions were fed back to their layouts. We have just started evaluation of the new LSIs, and no information leakage has been found so far. Detailed experimental results will soon be reported.

Acknowledgments

SASEBO boards were developed by AIST and Tohoku University in undertaking projects sponsored by METI (Ministry of Economy, Trade and Industry, Japan).

References

1. Kocher, P.C. : Timing Attacks on Implementations of Diffe-Hellmann, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104-113. Springer, Heidelberg (1996)
2. Kocher, P.C., Jaffe, J., Jun, B. : Differential Power Analysis. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388-397. Springer, Heidelberg (1999)
3. Suzuki, D., Saeki, M., Ichikawa, T. : Random Switching Logic: A Countermeasure against DPA based on Transition Probability. Cryptology ePrint Archive, Report 2004/346, 2004.
4. Suzuki, D., Saeki, M., Ichikawa, T. : Random Switching Logic: A New Countermeasure against DPA and Second-Order DPA at the Logic Level. IEICE Trans. Fundamentals E90-A(1), 160-168 (2007)
5. Messerges, T.S. : Using Second-Order Power Analysis to Attack DPA Resistant Software. In: Paar, C., Koç, Ç.K. (eds.) CHES 2000. LNCS, vol. 1965, pp. 238-251. Springer, Heidelberg (2000)
6. Waddle, J., Wagner, D. : Towards Efficient Second-Order Power Analysis. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 1-15. Springer, Heidelberg (2004)
7. Tiri, K., Schaumont, P. : Changing the Odds against Masked Logic. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 134-146. Springer, Heidelberg (2007)
8. Tiri, K., Schaumont, P. : Masking and Dual-Rail Logic Don't Add Up. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 95-106. Springer, Heidelberg (2007)
9. Tiri, K., Verbauwhede I. : A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In: Design, Automation and Test in Europe Conference (DATE2004), pp. 246-251. (2004)
10. TSMC Webpage, `http://www.tsmc.com/english/default.htm`
11. Mangard, S., Popp, T., Gammel, B.M. : Side-Channel Leakage of Masked CMOS Gates. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 351-365. Springer, Heidelberg (2005)
12. Suzuki, D., Saeki, M., Ichikawa, T. : DPA Lekage Models for CMOS Logic Circuits. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 366-382. Springer, Heidelberg (2005)
13. Satoh, A., Morioka, S., Takano, K., Munetoh, S. : A Compact Rijndael Hardware Architecture with S-Box Optimization. ASIACRYPT 2001, LNCS 2248, pp. 239-254. Springer, Heidelberg (2001)
14. Cryptographic Hardware Project : Project Webpage, `http://www.aoki.ecei.tohoku.ac.jp/crypto/web/cores.html`
15. Tiri, K., Hwang, D., Hojat A., Lai, B., Yang, S., Schaumont, P., Verbauwhede I. : Prototype IC with WDDL and Differential Routing - DPA Resistance Assessment. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 354-365. Springer, Heidelberg (2005)
16. Side-channel Attack Standard Evaluation Board (SASEBO) Webpage, `http://www.rcis.aist.go.jp/special/SASEBO/index-en.html`
17. 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)

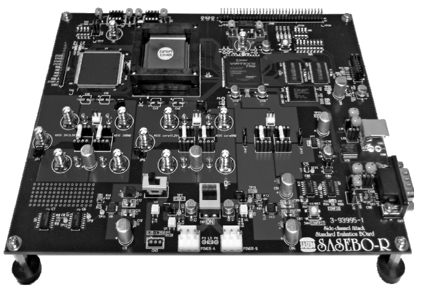


Fig. 16. Experimental board SASEBO-R

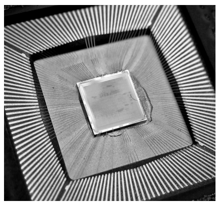


Fig. 17. Our prototype LSI

18. Messerges, T. S., Dabbish, E. A., Sloan R. H. :Investigations of Power Analysis Attacks on Smartcards. USENIX 1999, `http://www.usenix.org/`, 1999.
19. Bevan, R., Knudsen, R. :Ways to enhance differential power analysis. In: Lee, P. J., Lim, C. H. (eds.) ICISC 2002, LNCS 2587, pp. 327-342, Springer-Verlag, 2002.
20. Suzuki, D., Saeki, M., Matsumoto T. :Self-Contained Template Attack: How to Detect Weak Bits for Power Analysis without Reference Devices. SCIS2009, 1A1-2(In Japanese).

Appendix A: Weakness detected in our prototype LSI

A photograph of an evaluation board used in our experiment is shown in Fig. 16. Our pseudo RSL-AES co-processor can toggle on/off independently the two functions to prevent leakage, the random masking and glitch suppressing functions by setting the values of the mode register implemented in the prototype LSI shown in Fig. 17.

As for **combined**, Fig. 18 shows the results of attacks by 1 bit DPA that uses all input data to the SubBytes circuit as the selection function. The figure shows that there is a bit where the correct answer key ranks 1 with a million traces. Similar tendency is also observed in the experimental result shown in Fig. 19 changing selection functions.

This could be caused by some input signals not satisfying the delay conditions of the pseudo RSL. We have already confirmed some timing violations in a report of static timing analysis. Therefore, new LSIs have been developed, where the detailed delay information to satisfy the security conditions were fed back to their layouts. We have just started evaluation of the new LSIs, and no information leakage has been found so far. Detailed experimental results will soon be reported.

Also, in Fig. 18, as the number of trace increases, the correct key goes higher in rank and stays there for a while, and then goes down. This is caused by the bias of pseudo-random number for the masking. This trend is changed if the order of selecting trace data changes in the statistical processing. This case shows that it is difficult to know how many waveforms justify the end of attacks.

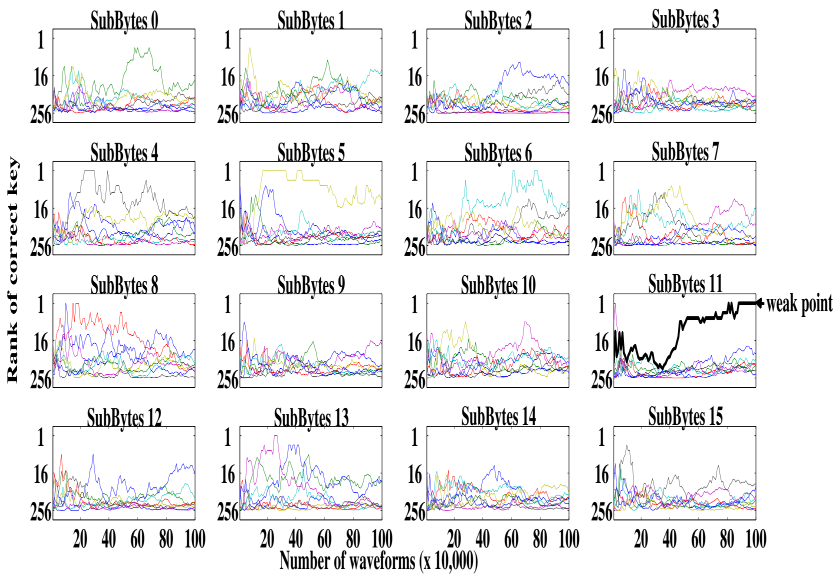


Fig. 18. 1-bit DPA result of all on **combined** with SubBytes input as selection function

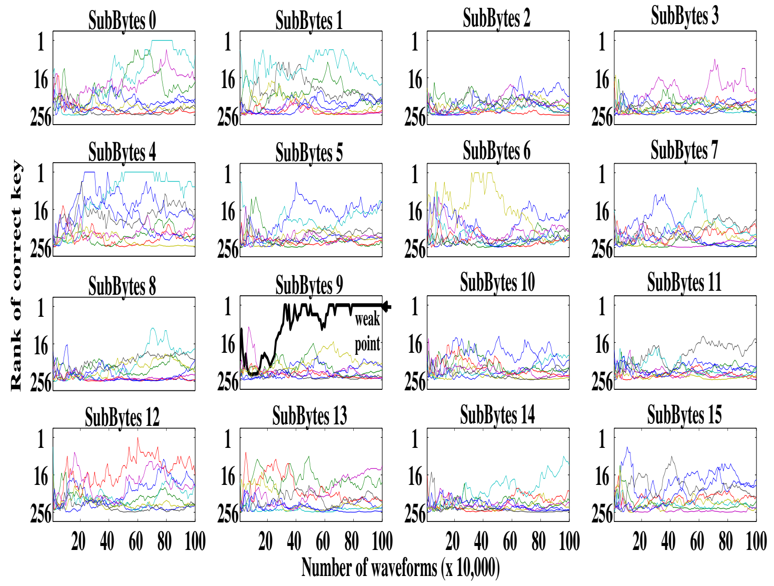


Fig. 19. 1-bit DPA result of all on **combined** with NAND gate input as selection function