Enhanced Power Analysis Attack Using Chosen Message against RSA Hardware Implementations

Atsushi Miyamoto*, Naofumi Homma*, Takafumi Aoki* and Akashi Satoh†
* Graduate School of Information Sciences, Tohoku University
6-6-05, Aramaki Aza Aoba, Aoba-ku, Sendai-shi 980-8579, Japan
Phone: +81-22-795-7169, Fax: +81-22-263-9308,
E-mail: miyamoto@aoki.ecei.tohoku.ac.jp
† National Institute of Advanced Industrial Science and Technology
1–18–13 Sotokanda, Chiyoda-ku, Tokyo, 101-0021 Japan

***Abstract*— SPA (Simple Power Analysis) attacks against RSA cryptosystems are enhanced by using chosen-message scenarios. One of the most powerful chosen-message SPA attacks was proposed by Yen et. al. in 2005, which can be applied to various algorithms and architectures, and can defeat the most popular SPA countermeasure using dummy multiplication. The special input values of -1 and a pair of $-X$ and X can be used to identify squaring operations performed depending on key bit stream. However, no experimental result on actual implementation was reported. In this paper, we implemented some RSA processors on an FPGA platform and demonstrated that Yen's attack with a signal filtering technique clearly reveal the secret key information in the actual power waveforms.**

I. Introduction

Side-channel attacks using physical information leakage pose a serious threat to cryptographic modules. In order to reveal the secret information, the power dissipation, electromagnetic radiation, or operating times correlated to the internal operations of the cryptographic modules are measured. Simple Power Analysis (SPA) and Differential Power Analysis (DPA) are basic side-channel attacks, and SPA and DPA attacks against RSA were first introduced by Kocher [1] and Messerges [2], respectively.

The basic concept of the SPA on RSA is to distinguish different characteristics of power waveforms for multiplication and squaring operations performed by an RSA in response to the bit pattern of a secret key. Such differences, however, are not always observable by the basic SPA and depend strongly on implementation details. In order to enhance secret information leaks on the power waveforms, chosen-message attacks for RSA using specific input data were proposed [3]–[5]. Yen proposed the use of the input value of -1 for the chosen message [5] and discussed the possibility of an attack to defeat the most popular SPA countermeasures using dummy multiplication [6]. However, no experimental evaluation of actual hardware or software implementation has been reported.

The present paper demonstrates and analyzes the effectiveness and characteristics of the chosen-message attacks against RSA processors implemented on an FPGA platform. Four types of the processors were designed by combining two high-radix Montgomery multiplication algorithms [7] and two types of multipliers. A signal filtering technique was also applied to analyze measured power waveforms, and, in addition to -1, input values of X and $-X$ were used. As a result, the secret key information was clearly revealed in all of the RSA implementations.

ALGORITHM-I
Modular Exponentiation (MSB First)

Input:	X, N, $E = (e_{k-1}, ..., e_1, e_0)_2$
Output:	$Z = X^E \bmod N$

```
1 : Z := 1;
2 : for i = k − 1 downto 0
3 :     Z := Z * Z mod N;        – squaring
4 :     if (e_i = 1) then
5 :         Z := Z * X mod N;    – multiplication
6 :     end if
7 : end for
```

II. RSA cryptosystem

The RSA cryptosystem employs modular exponentiation for encryption and decryption as follows:

$$C = P^E \bmod N, \tag{1}$$

$$P = C^D \bmod N, \tag{2}$$

where P is the plaintext, C is the ciphertext, E and N are the public keys, and D is the secret key.

ALGORITHM-I shows a left-to-right binary method, which is most commonly used for the modular exponentiations. This algorithm scans the bits of the exponent from MSB to LSB and always performs a squaring at Line 3, independently of the scanned bit value. However, the multiply operation at Line 5 is only executed if the scanned bit is '1'. The operation sequence in the binary method is not changed even when the Montgomery multiplication algorithm [8] is employed to speed up the exponentiation.

The concept of the SPA against RSA cryptosystem is to distinguish between multiplication and squaring in the power waveform. Figure 1 shows an image of the SPA against an RSA module using the left-to-right binary method, where the key bit pattern '10100' can be detected. The "square-and-multiply-always" algorithm introduced by Coron [6], which inserts dummy multiplications for the zero bits of the expo-

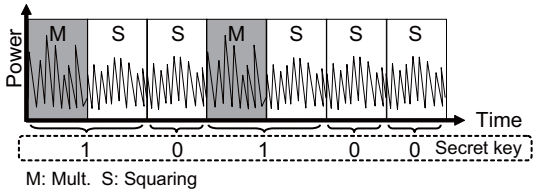


Fig. 1. Conventional SPA against RSA.

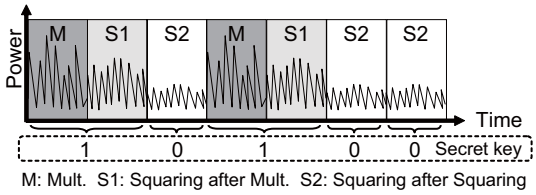


Fig. 2. Chosen-message SPA.

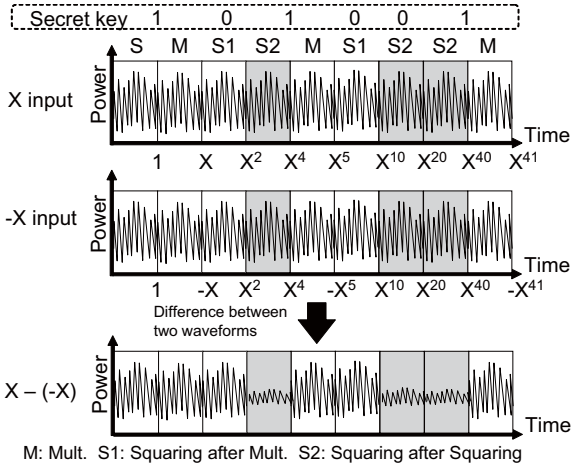


Fig. 3. Extended chosen-message SPA.

nent, is the most popular countermeasure against the SPA on RSA.

III. Chosen-message SPA

In this section, the chosen-message attack proposed by Yen [5] is described. The basic concept is to use a specific input data, -1, to enhance the differences between the multiplication and squaring operations. Using the value of -1, the multiplication and squaring results for the modular exponentiation can be maintained as constants. For example, let $-1 \bmod N (= N-1)$ be the input X in **ALGORITHM-I**. Then, the outputs of the multiplication and squaring operations during exponentiation are $-1 \bmod N$ and $1 \bmod N$, respectively.

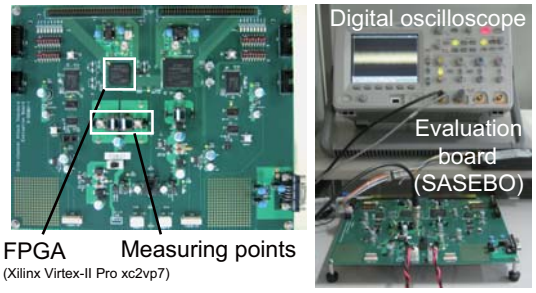
According to this property, the multiplication and squaring operations during exponentiation can be classified into three types: (M) multiplication after squaring, (S1) squaring after multiplication, and (S2) squaring after squaring as follows:

(M) $Z = 1 * (-1) \bmod N = -1 \bmod N$,
(S1) $Z = (-1) * (-1) \bmod N = 1 \bmod N$,
(S2) $Z = 1 * 1 \bmod N = 1 \bmod N$.

Figure 2 shows an image of the above-described SPA. In the binary method, squaring S1 follows multiplication M, and squaring S2 follows squaring S2 or squaring S1. In other words, M is never followed by S2. Thus, the bit pattern of the secret exponent can be obtained if one of the three operations is distinct from the others.

By confining the operation sequences as well as the data values, Yen's method can also defeat the "square-and-multiply-always" algorithms. Dummy multiplications are inserted before the S2 states to hide the key bit pattern. However, the result of dummy multiplication ($= -1$) is discarded, and the value of 1 is used in the following squaring, which becomes S2. Therefore, if the sequence M→S2 is observed, the multiplication is identified as a dummy multiplication, DM.

Another possible attack using two power waveforms obtained from inputs X and $-X (= N - X \bmod N)$ is presented in [5]. In Fig. 3, internal operations for S2 at the same time period are identical between inputs X and $-X$. Then, the S2 operations can be exposed by the difference between the waveforms at the bottom of Fig. 3. This attack is applied to arbitrary inputs, and thus can be available even if specific inputs such as $N-1$ are prohibited in the target module.



(a) (b)

Fig. 4. (a) Evaluation board (SASEBO), (b) Experimental environment.

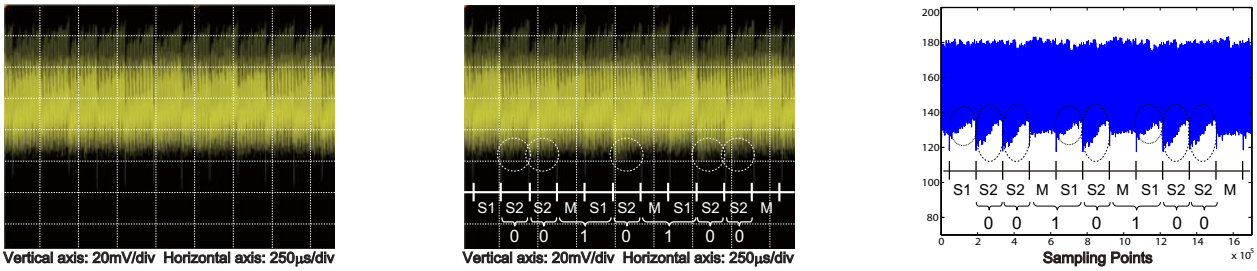
TABLE I
Experimental condition

Experimental FPGA board (SASEBO)	
FPGA	Xilinx Virtex-II Pro xc2vp7
Crystal oscillator	24-MHz
Resistance value	5 Ohm
Power supply voltage	3.3 V
Experimental equipment	
Digital oscilloscope	Agilent MSO6104A
Probe	Coaxial cable (50 Ohm)

IV. Experiments

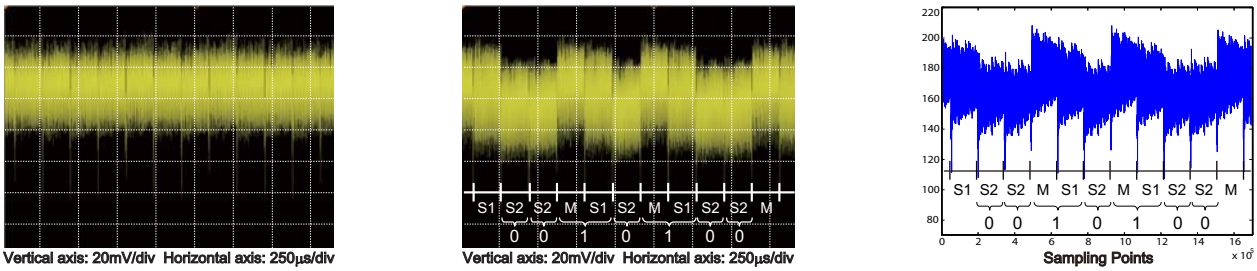
A. Experimental setup

1024-bit RSA processors with high-radix Montgomery multiplication [7] were designed to demonstrate the chosen-message SPA and to investigate its effectiveness depending on hardware architectures. In this experiment, four types of processors were designed by combining two Montgomery multiplication algorithms, Coarsely Integrated Operand Scanning (CIOS) and Finely Integrated Operand Scanning (FIOS) [7],



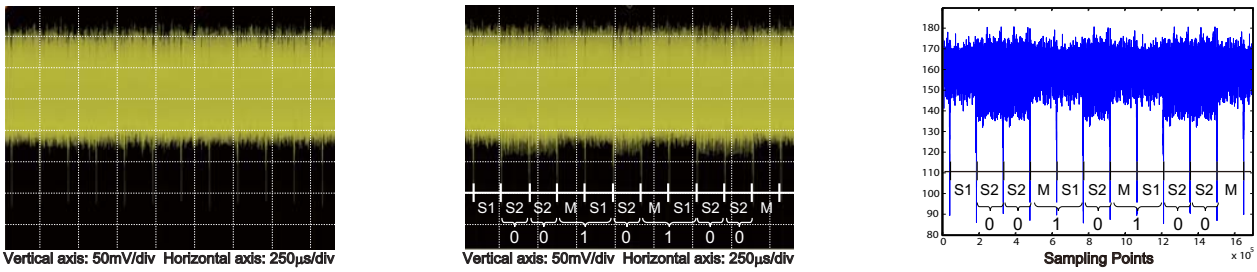
(a) (b) (c)

Fig. 5. Power waveforms of CIOS processor using an embedded multiplier: (a) random value input, (b) $N-1$ value input, (c) $N-1$ value input and filtering.



(a) (b) (c)

Fig. 6. Power waveforms of FIOS processor using an embedded multiplier: (a) random value input, (b) $N-1$ value input, (c) $N-1$ value input and filtering.



(a) (b) (c)

Fig. 7. Power waveforms of CIOS processor using a custom array multiplier with booth encoder: (a) random value input, (b) $N-1$ value input, (c) $N-1$ value input and filtering.

and two types of 32-bit multipliers, an embedded multiplier block in an FPGA and a custom array multiplier with a Booth encoder.

Figure 4 (a) shows the experimental FPGA (Xilinx Virtex-II Pro xc2vp7) board SASEBO (Side-channel Attack Standard Evaluation BOard)[1] and the measurement point at which a resistor is inserted between the FPGA ground pin and the ground plane of the board. The RSA processors on the FPGA were synthesized using Xilinx ISE 9.1. The power traces were monitored by an oscilloscope as voltage drops caused by the resistor (Fig. 4 (b)). The RSA operations were performed at a clock frequency of 24 MHz, and the sampling rate of the oscilloscope was 800 MSamples/sec. The modulus N was chosen randomly in this experiment. Table II summarizes the experimental conditions.

[1]SASEBO was developed by the National Institute of Advanced Industrial Science and Technology (AIST) and Tohoku University under a research project to develop security evaluation methodologies for cryptographic modules funded by the Ministry of Economy, Trade and Industry (METI) of Japan.

B. Experimental results and discussions

Figure 5 shows the power traces generated by the CIOS architecture using an embedded multiplier in the FPGA, where (a) and (b) indicate the waveforms for input data with random values and $N-1$, respectively. Figure 6 also shows the power traces of the FIOS architecture corresponding to Fig. 5. The conventional SPA is performed using the waveforms (a), but no relationship between the waveform patterns and the operations was observed. In contrast, multiplication and squaring can easily be distinguished in (b). These differences between CIOS and FIOS were caused by the sequence of multiplication. The CIOS algorithm does not always change two operands every cycle and thus differences in transistor switching are smaller than the FIOS algorithm that always changes the operands. In this experiment, a signal processing technique is applied for the waveforms (b) to emphasize the differences, and the waveforms (c) are obtained using a low-path filter (LPF). The cut-off frequency is set to 0.02 radians, which was determined by the frequency characteristics of waveforms. The figures

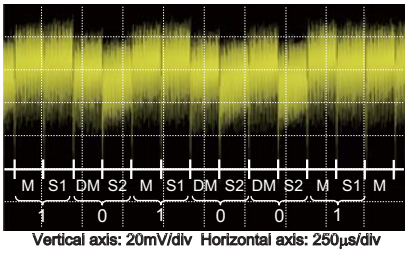
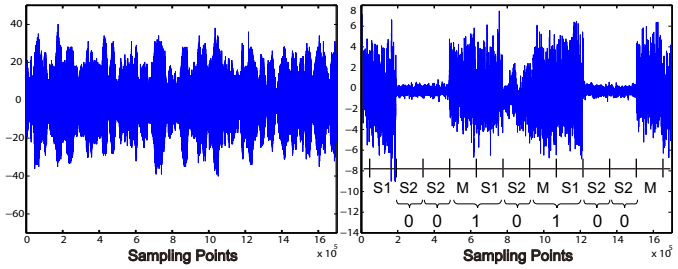
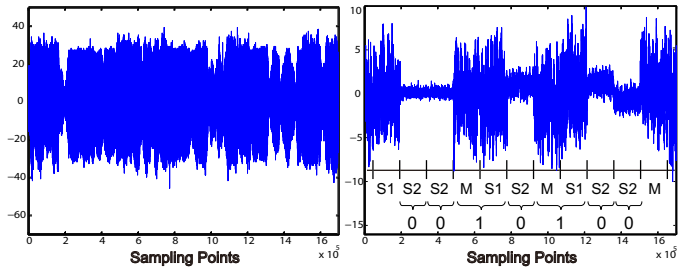


Fig. 8. Chosen-message SPA against squaring-and-multiply-always method.



(a) (b)

Fig. 9. SPA using X and $-X$ value inputs (CIOS): (a) difference waveform, (b) filtered difference waveform.



(a) (b)

Fig. 10. SPA using X and $-X$ value inputs (FIOS): (a) difference waveform, (b) filtered difference waveform.

show that the LPF processing technique significantly enhanced the differences for both architectures.

Figure 7 shows the power traces of the CIOS processor with a custom array multiplier. The power consumption was very high, and the boundaries of operations were not clear, but the LPF enhanced the differences in this case. The FIOS processor with a custom array multiplier also showed similar results to those of Fig. 6, but the waveforms are omitted due to space limitation. These results suggest that the chosen-message SPA would be applied to other platforms such as ASICs, in which the power consumed by an arithmetic core is dominant.

Figure 8 shows the power trace of the FIOS processor with the square-and-multiply-always method for the input value of -1. The true multiplications M and the dummy multiplications DM can be distinguished by checking whether the following squaring is S1 (which means M) or S2 (which means DM).

As shown above, the SPA using the chosen message -1 is a very powerful attack, but, as a countermeasure, an RSA module can check the value to remove it from operation. Therefore, an advanced attack using an arbitrary pair of X and $-X$ was investigated. In order to find squaring S2, in which the same internal operations were performed by the RSA processor for X and $-X$, a difference waveform was generated by subtracting the waveform of $-X$ from that of X. Then, the waveforms of Figs. 9 (a) and 10 (a) were obtained from those of Figs. 5 (a) and 6 (a) for CIOS and FIOS, respectively. The differential power for the S2 operation should ideally be zero, but the actual waveforms contain a lot of noise signals disturbing the power analysis. However, these signals can also be eliminated by using the LPF technique, after which the clear waveforms of Figs. 9 (b) and 10 (b) were obtained.

V. Conclusions

The present paper demonstrated and analyzed the effectiveness of SPA attacks using chosen messages against RSA hardware designs on an FPGA platform. The experimental results show that the differences between multiplication and squaring operations for all of the implementations were visible to the eye, while the power traces using random inputs did not expose the secret information. Chosen-message attacks can also defeat the most popular SPA countermeasure based on the square-and-multiply-always method. The extended attack using a message pair of X and $-X$ showed significant improvements to enhance the difference waveform for squaring. In addition to these algorithm-based attacks, a signal processing technique was introduced, and a low-pass filter optimized for the experimental RSA circuits on the FPGA successfully eliminated noise signals and strongly emphasized the differences of the operations depending on the secret key stream. Through this experiment, it was observed that waveform patterns are strongly influenced by the multiplication algorithms and the processor architectures, as well as the SPA attacks. Therefore, we are developing a cryptographic ASIC chip including an RSA processor to investigate the different characteristics between FPGA and ASIC with respect to side-channel attacks. The experimental results will be reported in the future.

References

[1] P. Kocher, J. Jaffe and B. Jun: “Differential power analysis”, CRYPTO 1999, Lecture Notes in Computer Science, **1666**, pp. 388 – 397 (1999).

[2] T. S. Messerges, E. A. Dabbish and R. H. Sloan: “Power analysis attacks of modular exponentiation in smartcards”, CHES 1999, Lecture Notes in Computer Science, **1717**, pp. 144–157 (1999).

[3] R. Novak: “SPA-based adaptive chosen-ciphertext attack on RSA implementation”, PKC 2002, Lecture Notes in Computer Science, **2274**, pp. 252–262 (2002).

[4] B. D. Boer, K. Lemke and G. Wicke: “A DPA attack against the modular reduction within a CRT implementation of RSA”, CHES 2002, Lecture Notes in Computer Science, **2523**, pp. 228–243 (2002).

[5] S. M. Yen, W. C. Lien, S. J. Moon and J. C. Ha: “Power analysis by exploiting chosen message and internal collisions - vulnerability of checking mechanism for RSA-decryption.”, Mycrypt 2005, Lecture Notes in Computer Science, **3715**, pp. 183–195 (2005).

[6] J. S. Coron: “Resistance against differential power analysis for elliptic curve cryptosystems”, CHES 1999, Lecture Notes in Computer Science, **1717**, pp. 192–302 (1999).

[7] C. K. Koc, T. Acar and B. S. Kaliski: “Analyzing and comparing montgomery multiplication algorithms”, IEEE Micro, **16**, 3, pp. 26–33 (1996).

[8] P. L. Montgomery: “Modular multiplication without trial division”, Math. Comp., **44**, 170, pp. 519–521 (1985).