Toward More Efficient DPA-Resistant AES Hardware Architecture Based on Threshold Implementation

Rei Ueno(✉), Naofumi Homma, and Takafumi Aoki

Tohoku University, Aramaki Aza Aoba 6–6–05, Aoba-ku, Sendai-shi 980-8579, Japan
ueno@aoki.ecei.tohoku.ac.jp, homma@riec.tohoku.ac.jp

Abstract. This paper presents a highly efficient AES hardware architecture resistant to differential power analyses (DPAs) on the basis of threshold implementation (TI). In contrast to other conventional masking schemes, the major feature of TI is to guarantee DPA-resistance under d-probing condition at the resister-transfer level (RTL). On the other hand, TI utilizes pipelining techniques between the non-linear functions to avoid propagating glitches, which would lead to non-negligible overheads of circuit area and latency. In this paper, we first propose a compact first-order TI-based AES S-box which has a major effect on the performance and DPA-resistance of AES hardware. The proposed S-box exploits a state-of-the-art TI construction with $d + 1$ shares in addition to the algebraic characteristics of AES S-box. We then propose an efficient AES hardware architecture suitable with the above TI-based S-box. The architectural advantage is given by register-retiming and tower-field arithmetic techniques. The performance of the proposed AES hardware was evaluated in comparison with that of conventional best ones. The logic synthesis result suggests that the proposed AES hardware architecture achieves more compact and 11–21% lower-latency than the conventional ones, which indicates that the proposed architecture can perform encryption based on TI with the lowest-energy. We also confirm the DPA-resistance of the proposed AES hardware by the Test Vector Leakage Assessment (TVLA) methodology with its FPGA implementation.

Keywords: Side-channel attacks · AES · Hardware implementation · Threshold implementation · DPA

1 Introduction

Cryptography has been widely used in many systems with secure communications, authentication, and digital signatures. According to the rapid growth of Internet of Things (IoT) applications, many cryptographic algorithms are being required in resource-constraint devices such as smart cards and sensors with limited chip area and power. On the other hand, various implementation attacks are attracting considerable attention because of the increasing applications of

S. Guilley (Ed.): COSADE 2017, LNCS 10348, pp. 50–64, 2017.
DOI: 10.1007/978-3-319-64647-3_4

cryptographic hardware to IoT devices. There is definitely a high demand for efficient tamper-resistant cryptographic hardware securing IoT applications.

Side-channel attack, which is one of the most powerful implementation attacks, retrieves the secret key from operating cryptographic hardware by exploiting side-channel information such as power consumption, EM radiation, and operation timing [9]. Differential power analysis (DPA) is a typical side-channel attack on symmetric ciphers (e.g., AES) which analyzes the relation between side-channel information and the calculated values of cryptographic operations with a statistical means. Since modern ciphers commonly employ GF arithmetic for their components [15], we should design GF arithmetic circuits resistant to DPAs for DPA-resistant cryptographic hardware. Until now, many countermeasures have been developed to defeat DPAs. Masking is considered as one of the predominant countermeasures, which eliminates DPA-leaks using randomness.

Threshold implementation (TI) was presented as a provably-secure countermeasure against DPAs, including advanced DPAs exploiting power consumption caused by glitches [10,11]. While many conventional countermeasures require to use specific tools and/or libraries (e.g., symmetric layout) [21], the usage of TI makes it possible to design DPA-resistant hardware with the standard design tools including standard cells and automatic layout. In recent years, some related works on TI have been reported. They include its extension to higher-order DPAs [2,17], DPA-resistant cryptographic hardware designs based on TI [3,7,12,16], and TI-friendly cryptography where TI can be efficiently applied to the S-box [5].

In this paper, we first present a compact AES S-box based on TI and then propose a more efficient DPA-resistant AES hardware architecture. The proposed TI-based AES S-box is designed with a combination of the state-of-the-art TI construction [17] and the algebraic characteristics of AES S-box. The proposed architecture employs a byte-serial architecture commonly used for TI-based AES in order to tolerate the overheads of circuit area and random number generation [3,7,12]. In such architectures, the latency overhead caused by the pipeline registers of TI is also non-negligible. The conventional works perform SubBytes, ShiftRows, and MixColumns at serial timings despite pipelined SubBytes, which indicates that an extra latency occurs in every round due to the pipelining, and results in the loss of energy. In contrast, the proposed AES hardware architecture exploits a new register-retiming technique to perform the above operations in a partially parallel manner with a modest increase of circuit area. In addition, the proposed architecture performs all the operations over tower field for a further reduction of latency (i.e., pipeline-stage). Furthermore, our architecture saves the cost of TI applied to the key scheduling unit according to the report of [16] that it has no DPA-leaks. With the results of logic synthesis, we confirm that the proposed method has a smaller area and 11–21% lower latency than conventional architectures. In addition, we evaluate the DPA-leakage of the proposed architecture implemented on an FPGA. The t-test result shows that there is no obvious first-order DPA-leak from the proposed architecture within 500,000 traces. While this paper focuses on the first-order security, the concept of the proposed architecture can be applied to the higher-order security.

The rest of this paper is organized as follows: Sect. 2 briefly describes TI. Section 3 proposes a more compact first-order TI-based AES S-box and an efficient byte-serial AES hardware architecture equipped with the S-box. Section 4 provides the performance of the proposed AES hardware architecture with logic synthesis results and evaluates its DPA leakage with an FPGA implementation. Section 5 contains our conclusion.

2 Threshold Implementaion

TI is a state-of-the-art masking countermeasure against DPAs [14,17]. The utilization of TI makes it possible to design any kind of arithmetic circuits over $GF(2^m)$ resistant even to advanced DPAs that utilize power glitches included in observed power consumption [10,11].

The working principle of TI is to represent a secret value $a \in GF(2^m)$ with $a_0 + a_1 + \cdots + a_i + \cdots + a_{s-1}$ by means of (s, s) threshold scheme [20], where $a_i \in GF(2^m)$ $(0 \leq i \leq s-1)$ is initially given by a random mask. Each element a_i is called a share. According to [17], any circuit satisfying the following three properties is secure under the dth-order DPAs. Let a and $a_0, a_1, \ldots, a_i, \ldots, a_{s-1}$ be the input and the shares of a, respectively. Let c and $c_0, c_1, \ldots, c_j, \ldots, c_{s'-1}$ be the output and the shares of c, respectively.

(i) *Correctness*

The first property implies that the sum of shares is equal to the original value at the input and output of the circuit, namely, $a = a_0 + a_1 + \cdots + a_i + \cdots + a_{s-1}$ and $c = c_0 + c_1 + \cdots + c_j + \cdots + c_{s'-1}$, where a and c are the input and output of the original function, respectively, and a_i and c_j are the input and output shares, respectively. This property indicates that the shared circuit correctly performs the original (i.e., nonshared) function.

(ii) *dth-order non-completeness*

The second property implies that the sum of chosen d output shares are independent of at least one input share. The number of input shares (i.e., s) required to meet the dth-order non-completeness is dependent on d and the algebraic degree of the circuit function, namely, the number of serially connected two-input AND gates in the combinational circuit. (Therefore, the number of input shares can be reduced by pipelining the circuit because the number of serially connected two-input AND gates are reduced in the circuits [14].)

(iii) *Uniformity*

The third property indicates that the input and output values of combinational circuits are uniformly distributed. See [3] for details.

While Properties (i) and (ii) can be realized for any GF function, some functions cannot satisfy Property (iii) under the constraints of properties (i) and (ii). However, the uniformity criterion can be satisfied by the addition of fresh mask(s), which is called a remasking scheme, to the non-uniform outputs [14]. If a circuit does not meet all the above properties, a glitch propagation between non-linear functions may leak the secret value [10]. A remasking is always necessary in the

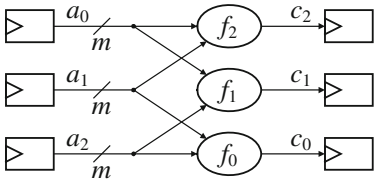


Fig. 1. Overview of circuit of function $t = 2$ meeting first-order non-completeness.

case of more than first-order security even if the output shares are uniformly distributed [17].

There were two known methods to construct circuits satisfying dth-order non-completeness. The difference of the two is the numbers of input shares. The first construction method was proposed in [14], where the number of input shares is given by $td+1$, where t is the algebraic degree of circuit function, and the number of output shares is given by $s' = \binom{s}{t}$ (e.g., $s' = s$ when $d = 1$). For example, Fig. 1 shows an overview of a circuit satisfying the first-order non-completeness when $t = 2$ (e.g., an two-input AND gate and a multiplier), where f_j indicates the function of the circuit that computes c_j. The number of input shares is 3 $(= 1 \times 2 + 1)$, and each f_j has 2 inputs. Given that c_j is independent of a_j in Fig. 1, the circuit is thought to be secure under first-order DPAs from the viewpoint of first-order non-completeness. Note that, in this case, the numbers of input and output shares are the same (i.e., $s = s' = 3$); however, they become different when $d \geq 2$.

It was shown that the above TI with $td + 1$ input shares was useful especially for designing hardware architectures of lightweight ciphers [16] such as PRESENT [4] and LED [8]. This is because such ciphers employ an S-box whose algebraic degree is at most three, which leads to an efficient TI construction with simple one-stage pipelining where the number of input and output are the same. In addition, all the output shares can be satisfied with the uniformity property in the TI-based S-box in PRESENT and LED, which means that any on-the-fly random number generation is not required during one block encryption.

The second construction method was recently proposed in [17], where the number of input shares is given by $d + 1$. To construct TI-based S-box of higher algebraic degree, such as in AES, a multi-stage pipelining and an on-the-fly random number generation are required because it is known that there is no TI construction satisfying the uniformity property [3,12]. The TI with $d + 1$ input shares is more useful for designing compact and efficient hardware architecture for such a cipher. Actually, in [7], a more compact TI-based AES hardware was designed with $d + 1$ input shares. For example, Fig. 2 shows a first-order TI-based $GF(2^m)$ multiplier with 2 $(= d + 1)$ input shares, where a_0, a_1, b_0, and b_1 are the input shares, c_0, c_1, c_2, and c_3 are the output shares, "mult" denotes a nonshared $GF(2^m)$ multiplier, and r_0, r_1, and r_2 are fresh masks for remasking. The multiplier performs $c = a \times b$ under the first-order non-completeness, where $a = a_0 + a_1$, $b = b_0 + b_1$, and $c = c_0 + c_1 + c_2 + c_3$. More precisely, each output

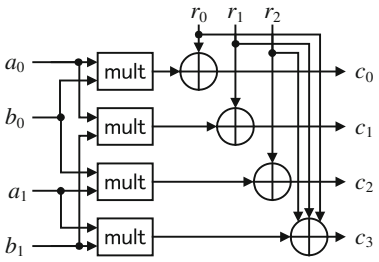


Fig. 2. First-order TI-based $GF(2^m)$ multiplier with two input shares.

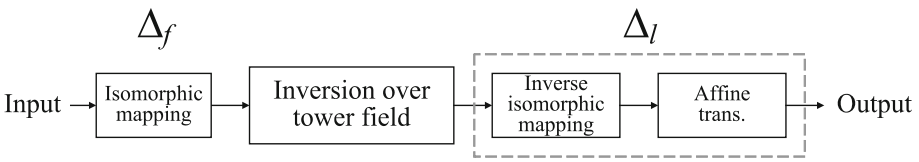


Fig. 3. AES S-box based on tower-field arithmetic.

share c_j is independent of either a_0 or a_1 (b_0 or b_1), which means the multiplier meets the first-order non-completeness. The number of output share for the TI is given by at least $(d+1)^t$.

3 Proposed Architecture

3.1 First-Order TI-Based AES S-Box

An AES S-box consists of $GF(2^8)$ inversion and $GF(2)$ affine transformation. While the inversion is performed by a polynomial basis (PB) based $GF(2^8)$ with an irreducible polynomial $x^8+x^4+x^3+x+1$, the use of tower-field arithmetic is useful for designing compact and efficient inversion circuits over $GF(2^8)$ [6,18]. Figure 3 shows the computation flow of AES S-box utilizing tower-field inversion. The input is initially mapped into a tower field. After the inversion over the tower field, the inverse mapping and affine transformation are applied to the output. Figure 4 shows a typical block diagram of tower-field inversion circuits, which consists of three stages [22]. The algebraic degrees of Stages 1, 2, and 3 are given by two, three, and two, respectively.

Until now, some TI-based inversion (i.e., S-box) circuits were proposed in the literature [3,7,12]. The major difference of the conventional circuits is the numbers of pipeline-stages and input shares. The inversion circuit in [12] was based on the TI with $td+1$ input shares and a four-stage pipeline architecture inserting pipeline registers inside Stage 2 in addition to boundaries between Stages 1, 2, and 3. While the inversion circuit in [3] also employed the TI with $td+1$ input shares, it was based on a three pipeline-stage architecture where

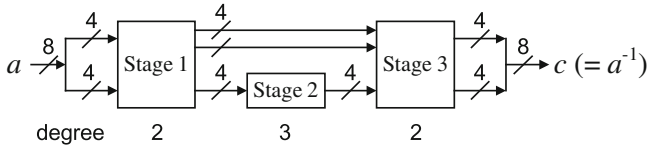


Fig. 4. Tower-field $GF(2^8)$ inversion circuit.

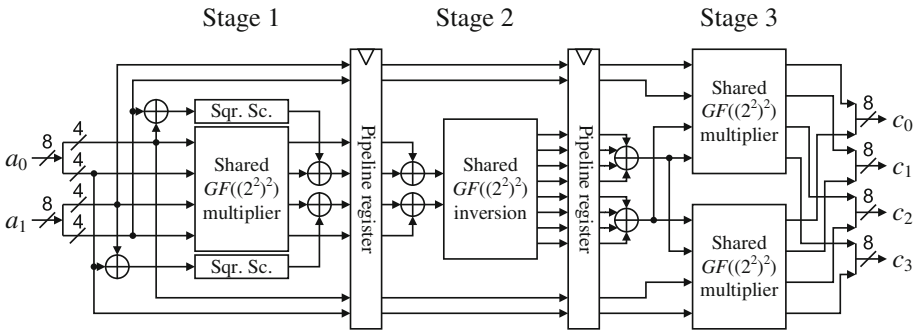


Fig. 5. Proposed first-order TI-based tower-field inversion circuit.

Stage 2 was given as a combinational circuit with algebraic degree three. Since the tower-field inversion is efficiently decomposed to three stages in terms of the algebraic expression [13], such three-stage pipeline architecture also makes the TI-based inversion circuit more efficient. On the other hand, a more compact TI-based inversion circuit with four-stage pipelining was designed in [7] on the basis of TI with $d + 1$ input shares. Note here that two more pipeline stages between the inversion and each mapping (i.e., Δ_f and Δ_l) are inserted for TI-based inversion circuits in order to satisfy dth-order non-completeness.

This paper presents a further compact and efficient TI-based inversion circuit design based on a combination of TI with $d+1$ input shares and the above algebraic characteristics of tower-field inversion. More precisely, the proposed inversion circuit exploits a three-stage pipeline technique similar to [3], and the input of each stage is given by two shares. While the possibility of such a design was mentioned in [7], there was neither description nor evaluation in the literature. Figure 5 illustrates the proposed first-order TI-based S-box which performs $(c_0 + c_1 + c_2 + c_3) = (a_0 + a_1)^{-1}$ using three clock cycles where Stages 1, 2, and 3 correspond to those of Fig. 4, respectively. Note here that paths with fresh masks for remasking are omitted for simplicity, but Stages 1, 2, and 3 require 12-, 28-, and 24-bit random numbers for remasking [7], respectively. The block "Shared $GF((2^2)^2)$" multiplier denotes the TI-based multiplier shown in Fig. 2. The block "Sqr. Sc." performs squaring and scaling operations over $GF((2^2)^2)$ in a nonshared manner. Finally, the block "Shared $GF((2^2)^2)$ inversion" performs $GF((2^2)^2)$ inversion under the first-order non-completeness by a combinational

Table 1. Performance evaluation of first-order TI-based AES S-boxes

	Area [GE]		Clock cycles		Area-Latency product	Randomness [bit]
	`compile`	`_ultra`	S-box	Inversion		
Moradi et al. [12]	No data	4,244	5	4	21,220	44
Bilgin et al. [3]	2,835	2,224	4	3	8,896	32
Cnudde et al. [7]	1,977	1,872	6	4	11,232	54
This work	1,425	1,342	5	3	6,710	64

circuit. As stated in [7], the number of output shares is given by $(d+1)^t$, which would have an impact on the circuit area. However, each subfunction of Shared $GF((2^2)^2)$ inversion can be efficiently factored and implemented using OR (or NOR) gates, which makes Stage 2 smaller. An example of logical expression for Shared $GF((2^2)^2)$ inversion is described in Appendix.

The proposed TI-based S-box was evaluated with Synopsys Design Compiler version D-2010.03 and TSMC 65-nm standard CMOS technology. Table 1 shows the synthesis results of the conventional and proposed first-order TI-based S-boxes, where Area denotes circuit area estimated by two-input NAND equivalent gate size (GE: Gate Equivalents), Clock cycles denotes the number of clock cycles required to perform S-box and inversion operations, Area-Latency product denotes the product of Area and Clock cycles (of S-box), and Randomness denotes the number of random bits required in a clock cycle. The columns `compile` and `_ultra` in Area were obtained by the commands `compile` and `compile_ultra`, respectively. For comparison, the values of conventional methods were derived from a table in [7]. We can confirm that our S-box achieved the smallest area without latency overhead while more randomness is consumed. In other words, our S-box is especially effective if random number generation is not critical.

3.2 Proposed Byte-Serial AES Hardware Architecture

Figure 6 shows the proposed byte-serial AES hardware architecture with the above 1st-order TI-based inversion circuit. The proposed architecture basically has an eight-bit datapath. In Fig. 6, the arrow without bit-width information denotes an eight-bit data flow. The blocks "State array" and "Key array" denote register arrays to store the intermediate values and round keys, respectively. The block "TI-based inversion" denotes the TI-based inversion circuits. Note that paths of random numbers for remasking are omitted. The blocks "Δ_f" and "Δ^{-1}" perform the isomorphic mapping and inverse mapping, respectively. Note that the output of Δ_f should be stored into the register "R" for satisfying non-completeness in the following inversion. Two State arrays are required to store $(d+1)$-shared intermediate value. On the other hand, since it is known that the key scheduling function has no DPA-leaks [16], we do not apply TI to the Key array. SubWord in the key scheduling function is performed by a nonshared S-box in [6]. Here, the S-box should be gated using AND gates to

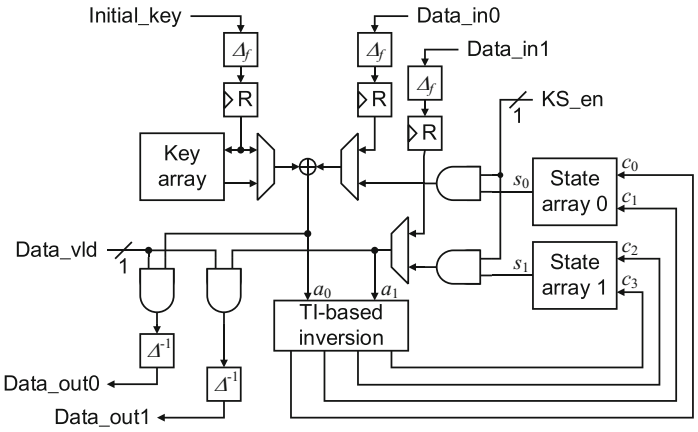


Fig. 6. Proposed byte-serial AES hardware architecture.

reduce dynamic power consumption, where the gating is controlled by one-bit signal "KS_en." Note that though the SubWord can also be performed using TI-based inversion, we use a distinct non-pipelined S-box to suppress the latency due to the pipelined Inversion. Note also that the proposed architecture would be designed with a higher-order TI-based inversion circuit in the similar manner.

Our architecture performs all operations (i.e., AddRoundkey, SubBytes, Shift-Rows, MixColumns, and key scheduling) over the tower field to reduce the number of pipeline stages (i.e., latency). Therefore, the isomorphic mappings are performed only at the input and output of the circuit. In addition, a new register-retiming technique, where the affine transformation in SubBytes is performed in State array, is introduced to further reduce the latency of the pipeline architecture. Consequently, the latency for two clock cycles are reduced by the above architectural design.

Figure 7 shows the internal structure of State array, which mainly consists of eight-bit registers and logic circuits for affine transformation and MixColumns. (Key array is omitted because it can be implemented in the same manner as [12].) We applied the above register-retiming techniques to our State array. The proposed State array is different from that of [12] because of the following three features: (i) the SubBytes of the last byte and ShiftRows are simultaneously performed in one clock cycle, (ii) MixColumns of the second and third colomns and the next round SubBytes are executed in parallel, and (iii) the SubBytes of the last four bytes and MixColumns are simultaneously performed. While the conventional State array has distinct paths only for ShiftRows, our State array performs ShiftRows and one-byte shift simultaneously using a unified path indicated in gray by allowed lines in Fig. 7 thanks to the feature (i). The output of S2 is given to TI-based inversion instead of S0 according to the feature (ii). Finally, by the feature (iii), affine transformation is performed at "Aff" in parallel during the byte shift (for the 0th–11th bytes) or MixColumns (for 12th–15th

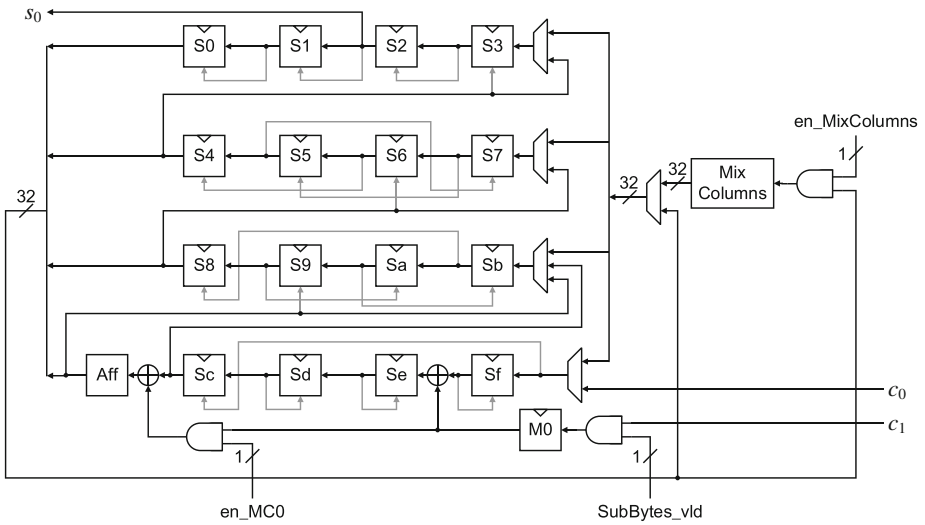


Fig. 7. State array.

bytes). It is possible to unify the affine transformation and the MixColumns in the same manner as [23]; however, we do not apply the unification technique to the proposed architecture it does not contribute to the increase of efficiency (i.e., the product of circuit area and latency). Note here that MixColumns should be gated as well as S-box for SubWord.

Figure 8 shows the timing diagrams of (a) conventional [12] and (b) proposed byte-serial AES hardware architectures, where "SubBytes k," "Inversion k," "Aff k," "SR," "MC l," and "KS l" denote the ith SubBytes, ith byte inversion, ith byte affine transformation, ShiftRows, lth column MixColumns, and lth byte SubWord in key scheduling, respectively. The blocks in gray denote operations of the previous or next round executed in parallel to the round of interest. From Fig. 8, we can confirm that our architecture achieves 20 clock cycles for one round operation while the conventional one requires 25 clock cycles because of the effect of the above resister-retiming and tower-field arithmetic techniques.

4 Evaluation

4.1 Performance Evaluation

To conduct a performance evaluation, we synthesized the proposed hardware architecture with Synopsys Design Compiler and TSMC 65 nm standard CMOS as above. Table 2 shows the synthesis result of the proposed architecture in Fig. 6. For comparison, Table 2 also shows those of the conventional ones derived in the same manner as Table 1. Note that Area of This work includes the area required for all the components in Fig. 6 and a control unit implemented with a 10-bit shift

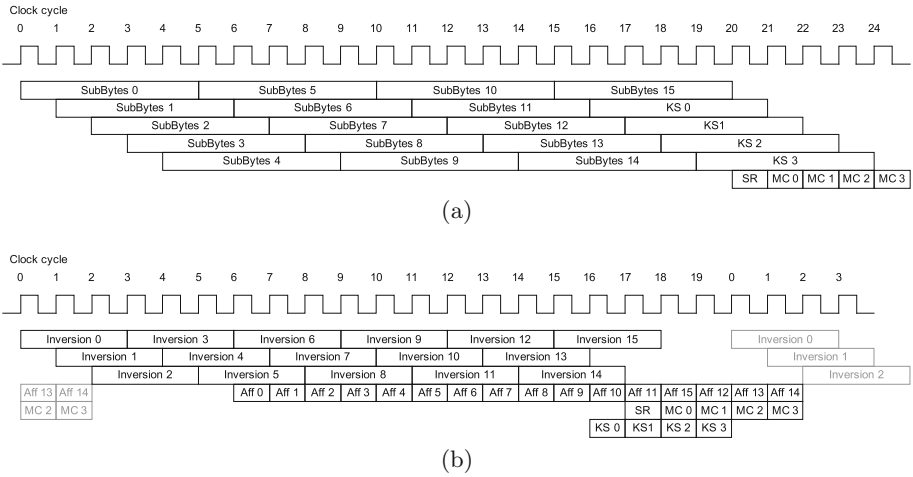


Fig. 8. Timing diagrams of (a) conventional [12] and (b) proposed byte-serial AES hardware architectures.

register and a five-bit counter. From Table 2, we confirmed that our architecture achieved 11–21% lower latency than the conventional ones. Though additional path selectors for register-retiming and MixColumns over tower fields would have an influence on the circuit area [6], our architecture achieved the smallest circuit area because of the proposed S-box and nonshared Key array. This also indicates that the circuit area can be further reduced by performing ShiftRows in a distinct clock cycle and/or replacing the tower-field MixColumns with AES-field one in exchange for increasing 10 clock cycles. Table 2 also shows the estimated power consumption based on gate-level timing simulation, where Power-Latency product indicates the product of Power and Clock cycles. The values of the conventional work was calculated using a table in [12]. The scaled values of Power and Power-Latency product in the parentheses are derived by dividing the original ones by the square of process rate (i.e., $(180/65)^2$). (The architecture in [12] was synthesized with 180 nm standard CMOS.) Note that it is quite difficult to compare power consumption estimation of hardware architectures in a fair manner, which heavily depends on the used technology and estimation method. However, the results roughly indicate that the lower latency would directly lead to lower energy of one block encryption. Thus, we confirmed the effectiveness of the proposed method.

4.2 Experimental Evaluation of DPA-Leakage

The DPA-resistance capability of our S-box was evaluated with an experiment using an FPGA implementation.

Figure 9 shows the experimental setup consisting of a Side-Channel Attack Standard Evaluation Board (SASEBO-G) [1] and an oscilloscope Tektronix

Table 2. Performance of AES hardware architecture based on first-order TI

	Area [GE]		Clock cycles	Area-Latency product	Power [μW]	Power-Latency product
	`compile`	`_ultra`				
Moradi et al. [12]	11,114	11,031	266	2,956 K	24.12 (3.14)	6,415 (835)
Bilgin et al. [3]	8,119	7,282	246	1,997 K	No data	
Cnudde et al. [7]	6,681	6,340	276	1,844 K	No data	
This work	6,321	6,053	219	1,376 K	3.06	670

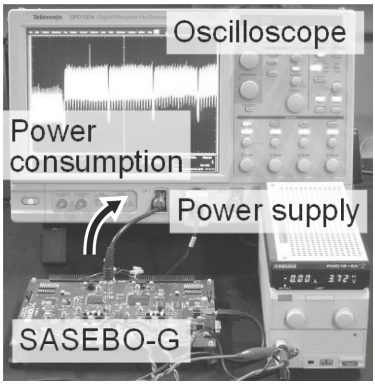


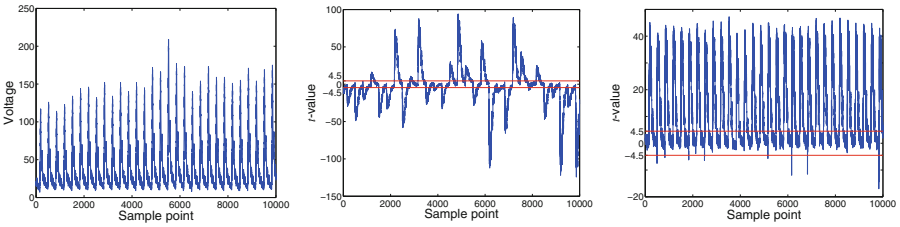
Fig. 9. Experimental setup.

DPO7254. The proposed AES hardware architecture with the proposed S-box was implemented on an FPGA (Xilinx Virtex II Pro) on the SASEBO-G, and the power variation was sampled with the sampling rate of 1 GS/s.

We evaluated the resistance and vulnerability of the AES hardware architecture by Test Vector Leakage Assessment (TVLA) based on Welch's t-test (a.k.a. non-specific t-test) [19]. The TVLA examines t-values which indicate the existence of dth-order DPA-leakage exploitable by the attackers.

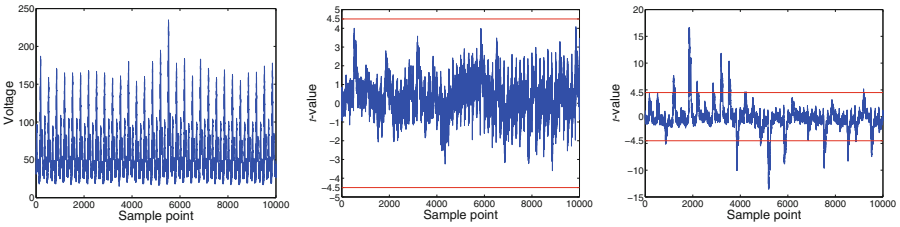
Figures 10(a) and 11(a) show examples of power traces at around the nineth with and without a pseudo random number generator (PRNG) implemented on the FPGA, respectively. When the PRNG is turned on, the TI works. We can find the small spikes between the big spikes in Fig. 11(a) because AES and PRNG are asynchronously active. Thus, the PRNG would not have a significant impact on the following TVLA result.

Figures 10 and 11 show the (b) first-order and (c) second-order TVLA results. We used 10,000 and 500,000 traces for Figs. 10 and 11, respectively. It is known that the absolute t-value of more than 4.5 indicates a high confidence in the existence of exploitable DPA-leakage. The results suggest that our design is resistant to the first-order DPAs under the condition of 500,000 traces by means



(a) Raw trace. (b) First-order. (c) Second-order.

Fig. 10. Measurement and TVLA results without PRNG.



(a) Raw trace. (b) First-order. (c) Second-order.

Fig. 11. Measurement and TVLA results with PRNG.

of the first-order TI. On the other hand, we can see the second-order leakage in both Figs. 10 and 11 due to the limitation of the first-order TI. Thus, we could validate the DPA resistance of the proposed hardware architecture in the experimental condition with 500,000 traces.

5 Conclusion

This paper presented an efficient DPA-resistant AES hardware architecture based on the 1st-order TI. We first described the most compact first-order TI-based S-box design by combining the TI with $d + 1$ input shares and the algebraic characteristics of S-box. We then proposed a more efficient AES hardware architecture based on register-retiming and tower-field arithmetic in addition to the proposed S-box. The logic synthesis result showed that the proposed architecture achieved 11–21% lower latency and smaller area than the conventional ones, which would lead to the lowest-energy encryption secure against first-order DPAs. The DPA-resistance was validated through an experimental evaluation based on TVLA with 500,000 traces.

Our architecture can also be easily extended to higher-order TI-based S-boxes. On the other hand, the proposed S-box is not necessarily useful for compact implementation for the case of higher orders because the number of Stage 2 outputs increases by the cubic of TI-order, that is, by $(d + 1)^3$. A further evaluation of the proposed architecture and S-box for higher-order security

would be required in the future. It is also demanded to consider the (partially) uniform sharing of TI-based $GF((2^2)^2)$ inversion for a further efficient first-order DPA-resistant implementation.

Acknowledgments. This research has been supported by JSPS KAKENHI Grants No. 16K12436 and No. 16J05711.

Appendix: First-Order TI-Based $GF((2^2)^2)$ Inversion with $d+1$ Input Shares

Let $a^{(n)}$ and $a_i^{(n)}$ $(0 \le n \le 3, 0 \le i \le 1)$ be the nth bit of input and its shares, respectively. Let $c^{(n)}$ and $c_j^{(n)}$ $(0 \le j \le 7)$ be the nth bit of output and its shares, respectively. Here, the least-significant bits correspond to $a_i^{(0)}$ and $c_j^{(0)}$.

$$c_0^{(0)} = a_0^{(1)}a_0^{(2)}a_0^{(3)} + a_0^{(1)}a_0^{(3)} + a_0^{(2)}, \tag{1}$$
$$c_0^{(1)} = a_0^{(0)}a_0^{(2)}a_0^{(3)} + a_0^{(0)}a_0^{(3)} + a_0^{(2)}, \tag{2}$$
$$c_0^{(2)} = a_0^{(0)}a_0^{(1)}a_0^{(3)} + a_0^{(1)}a_0^{(3)} + a_0^{(0)}, \tag{3}$$
$$c_0^{(3)} = a_0^{(0)}a_0^{(1)}a_0^{(2)} + a_0^{(1)}a_0^{(2)} + a_0^{(0)}, \tag{4}$$
$$c_1^{(0)} = a_0^{(1)}a_0^{(2)}a_1^{(3)} + a_0^{(1)}a_0^{(2)} + a_0^{(0)}a_1^{(3)}, \tag{5}$$
$$c_1^{(1)} = a_0^{(0)}a_0^{(2)}a_1^{(3)} + a_0^{(0)}a_1^{(3)}, \tag{6}$$
$$c_1^{(2)} = a_0^{(0)}a_0^{(1)}a_1^{(3)} + a_0^{(1)}a_1^{(3)}, \tag{7}$$
$$c_1^{(3)} = a_0^{(0)}a_0^{(1)}a_1^{(2)} + a_0^{(1)}a_1^{(2)}, \tag{8}$$
$$c_2^{(0)} = a_0^{(1)}a_1^{(2)}a_0^{(3)} + a_0^{(0)}a_1^{(2)} + a_0^{(0)}a_0^{(3)}, \tag{9}$$
$$c_2^{(1)} = a_0^{(0)}a_1^{(2)}a_0^{(3)} + a_0^{(1)}a_0^{(3)} + a_0^{(3)}, \tag{10}$$
$$c_2^{(2)} = a_0^{(0)}a_1^{(1)}a_0^{(3)} + a_1^{(1)}a_0^{(2)} + a_0^{(0)}a_0^{(2)}, \tag{11}$$
$$c_2^{(3)} = a_0^{(0)}a_1^{(1)}a_0^{(2)} + a_1^{(1)}a_1^{(3)} + a_1^{(1)}, \tag{12}$$
$$c_3^{(0)} = a_0^{(1)}a_1^{(2)}a_1^{(3)} + a_0^{(0)}a_1^{(3)}, \tag{13}$$
$$c_3^{(1)} = a_0^{(0)}a_1^{(2)}a_1^{(3)} + a_0^{(1)}a_1^{(3)}, \tag{14}$$
$$c_3^{(2)} = a_0^{(0)}a_1^{(1)}a_1^{(3)} + a_0^{(0)}a_1^{(2)} + a_1^{(1)}a_1^{(2)}, \tag{15}$$
$$c_3^{(3)} = a_0^{(0)}a_1^{(1)}a_1^{(2)} + a_1^{(1)}a_0^{(3)}, \tag{16}$$
$$c_4^{(0)} = a_1^{(1)}a_0^{(2)}a_0^{(3)} + a_1^{(1)}a_0^{(3)}, \tag{17}$$
$$c_4^{(1)} = a_1^{(0)}a_0^{(2)}a_0^{(3)} + a_1^{(1)}a_0^{(3)}, \tag{18}$$
$$c_4^{(2)} = a_1^{(0)}a_0^{(1)}a_0^{(3)} + a_1^{(0)}a_0^{(2)} + a_0^{(1)}a_0^{(2)}, \tag{19}$$
$$c_4^{(3)} = a_1^{(0)}a_0^{(1)}a_0^{(2)} + a_0^{(1)}a_1^{(3)}, \tag{20}$$

$$c_5^{(0)} = a_1^{(1)} a_0^{(2)} a_1^{(3)} + a_1^{(0)} a_0^{(2)}, \tag{21}$$
$$c_5^{(1)} = a_1^{(0)} a_0^{(2)} a_1^{(3)} + a_1^{(1)} a_1^{(3)} + a_1^{(3)}, \tag{22}$$
$$c_5^{(2)} = a_1^{(0)} a_0^{(1)} a_1^{(3)} + a_0^{(1)} a_1^{(2)} + a_1^{(0)} a_1^{(2)}, \tag{23}$$
$$c_5^{(3)} = a_1^{(0)} a_0^{(1)} a_1^{(2)} + a_0^{(1)} a_0^{(3)} + a_0^{(3)}, \tag{24}$$
$$c_6^{(0)} = a_1^{(1)} a_1^{(2)} a_0^{(3)} + a_1^{(0)} a_1^{(2)} + a_1^{(0)} a_0^{(3)}, \tag{25}$$
$$c_6^{(1)} = a_1^{(0)} a_1^{(2)} a_0^{(3)} + a_1^{(0)} a_0^{(3)}, \tag{26}$$
$$c_6^{(2)} = a_1^{(0)} a_1^{(1)} a_0^{(3)} + a_1^{(1)} a_0^{(3)}, \tag{27}$$
$$c_6^{(3)} = a_1^{(0)} a_1^{(1)} a_0^{(2)} + a_1^{(1)} a_0^{(2)}, \tag{28}$$
$$c_7^{(0)} = a_1^{(1)} a_1^{(2)} a_1^{(3)} + a_1^{(1)} a_1^{(3)} + a_1^{(2)}, \tag{29}$$
$$c_7^{(1)} = a_1^{(0)} a_1^{(2)} a_1^{(3)} + a_1^{(0)} a_1^{(3)} + a_1^{(2)}, \tag{30}$$
$$c_7^{(2)} = a_1^{(0)} a_1^{(1)} a_1^{(3)} + a_1^{(1)} a_1^{(3)} + a_1^{(0)}, \tag{31}$$
$$c_7^{(3)} = a_1^{(0)} a_1^{(1)} a_1^{(2)} + a_1^{(1)} a_1^{(2)} + a_1^{(0)}. \tag{32}$$

References

1. Side-channel attack standard evaluation board (SASEBO). http://www.rcis.aist.go.jp/special/SASEBO
2. Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V., Rijmen, V.: Higher-order threshold implementations. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 326–343. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45608-8_18
3. Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V., Rijmen, V.: Trade-offs for threshold implementations illustrated on AES. IEEE Trans. Comput. Aided Des. Integr. Syst. **34**(7), 1188–1200 (2015)
4. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74735-2_31
5. Boss, E., Grosso, V., Güneysu, T., Leander, G., Moradi, A., Schneider, T.: Strong 8-bit Sboxes with efficient masking in hardware. In: Gierlichs, B., Poschmann, A.Y. (eds.) CHES 2016. LNCS, vol. 9813, pp. 171–193. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53140-2_9
6. Canright, D.: A very compact S-Box for AES. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 441–455. Springer, Heidelberg (2005). doi:10.1007/11545262_32

7. De Cnudde, T., Reparaz, O., Bilgin, B., Nikova, S., Nikov, V., Rijmen, V.: Masking AES with $d + 1$ shares in hardware. In: Gierlichs, B., Poschmann, A.Y. (eds.) CHES 2016. LNCS, vol. 9813, pp. 194–212. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53140-2_10
8. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.: The LED block cipher. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 326–341. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23951-9_22
9. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999). doi:10.1007/3-540-48405-1_25
10. Mangard, S., Pramstaller, N., Oswald, E.: Successfully attacking masked AES hardware implementations. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 157–171. Springer, Heidelberg (2005). doi:10.1007/11545262_12
11. Moradi, A., Mischke, O., Eisenbarth, T.: Correlation-enhanced power analysis collision attack. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 125–139. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15031-9_9
12. Moradi, A., Poschmann, A., Ling, S., Paar, C., Wang, H.: Pushing the limits: a very compact and a threshold implementation of AES. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 69–88. Springer, Heidelberg (2011). doi:10.1007/978-3-642-20465-4_6
13. Morioka, S., Satoh, A.: An optimized S-Box circuit architecture for low power AES design. In: Kaliski, B.S., Koç, K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 172–186. Springer, Heidelberg (2003). doi:10.1007/3-540-36400-5_14
14. Nikova, S., Rijmen, V., Schläffer, M.: Secure hardware implementation of nonlinear functions in the presence of glitches. J. Cryptol. **24**, 292–321 (2011)
15. Nyberg, K.: Differentially uniform mappings for cryptography. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 55–64. Springer, Heidelberg (1994). doi:10.1007/3-540-48285-7_6
16. Poschmann, A., Moradi, A., Khoo, K., Lim, C.W., Wang, H., Ling, S.: Side-channel resistant crypto for less than 2,300 GE. J. Cryptol. **24**, 322–334 (2011)
17. Reparaz, O., Bilgin, B., Nikova, S., Gierlichs, B., Verbauwhede, I.: Consolidating masking schemes. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 764–783. Springer, Heidelberg (2015). doi:10.1007/978-3-662-47989-6_37
18. Satoh, A., Morioka, S., Takano, K., Munetoh, S.: A compact Rijndael hardware architecture with S-Box optimization. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 239–254. Springer, Heidelberg (2001). doi:10.1007/3-540-45682-1_15
19. Schneider, T., Moradi, A.: Leakage assessment methodology. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 495–513. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48324-4_25
20. Shamir, A.: How to share a secret. Commun. ACM **22**, 612–613 (1979)
21. 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 and Exhibition (DATE), vol. 1, pp. 246–251 (2004)
22. Ueno, R., Homma, N., Sugawara, Y., Nogami, Y., Aoki, T.: Highly efficient $GF(2^8)$ inversion circuit based on redundant GF arithmetic and its application to AES design. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 63–80. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48324-4_4
23. Ueno, R., Morioka, S., Homma, N., Aoki, T.: A high throughput/gate AES hardware architecture by compressing encryption and decryption datapaths. In: Gierlichs, B., Poschmann, A.Y. (eds.) CHES 2016. LNCS, vol. 9813, pp. 538–558. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53140-2_26