Another Algebraic Decomposition Method for Masked Implementation

Shoichi Hirose

University of Fukui, Fukui, Japan
hrs_shch@u-fukui.ac.jp

Abstract. Side channel attacks are serious concern for implementation of cryptosystems. Masking is an effective countermeasure against them and masked implementation of block ciphers has been attracting active research. It is an obstacle to efficient masked implementation that the complexity of an evaluation of multiplication is quadratic in the order of masking. A direct approach to this problem is to explore methods to reduce the number of multiplications required to represent an S-box. An alternative approach proposed by Carlet et al. in 2015 is to represent an S-box as composition of polynomials with low algebraic degrees. We follow the latter approach and propose to use a special type of polynomials with a low algebraic degree as components, which we call generalized multiplication (GM) polynomials. The masking scheme for multiplication can be applied to a GM polynomial, which is more efficient than the masking scheme for a polynomial with a low algebraic degree. Our experimental results show that, for 4-/6-/8-bit permutations, the proposed decomposition method is more efficient than the method by Carlet et al. in most cases in terms of the number of evaluations of low-algebraic-degree polynomials required by masking.

Keywords: Algebraic decomposition · Boolean function · masking · S-box

1 Introduction

Background. Side channel attacks introduced by Kocher [11] are serious concern for implementation of cryptosystems. Chari et al. [5] proposed a sound approach based on secret sharing [2, 15] against a class of side-channel attacks analyzing power consumption [12]. It is usually called masking [13] in this context. The d-th order masking splits each internal variable into $(d+1)$ shares so that any information of the internal variable cannot be recovered from at most d shares. The complexity of a successful side channel attack against a masked implementation was shown to be exponential in the masking order d [5].

Masked implementation has often been discussed for block ciphers. A block cipher can be manipulated as a function over the finite field $\mathbb{F}_2$ or its extension. For a scalar multiplication or an addition, the number of operations to compute shares of the result from the shares of an input is $O(d)$. For a square, it is also

$O(d)$. For a multiplication, on the other hand, it is $O(d^2)$. Thus, a multiplication is especially called a nonlinear multiplication to refer to the difference from a square.

For efficient masked implementation of block ciphers, it has been actively studied to reduce the number of nonlinear multiplications required to compute an S-box. A similar but different approach is to represent an S-box as composition of polynomials with low algebraic degrees [4].

Our Contribution. We present a method for algebraic decomposition inspired by the method of Carlet et al. [4] and the method to reduce the number of nonlinear multiplications of Goudarzi et al. [8]. The proposed method can be applied to any function from $\{0,1\}^n$ to $\{0,1\}^n$ for even n. It regards a given function $h(x)$ as a pair of bivariate polynomials $(h_0(x_0,x_1), h_1(x_0,x_1))$, where $h_b : \mathbb{F}_{2^{n/2}} \times \mathbb{F}_{2^{n/2}} \to \mathbb{F}_{2^{n/2}}$ for $b \in \{0,1\}$. Then, it represents h as composition of pairs of linear combinations of $x_0^{2^{i_0}} x_1^{2^{i_1}}$, where $0 \leq i_b \leq n/2-1$ for $b \in \{0,1\}$. We call such a pair of linear combinations a generalized multiplication (GM) polynomial. The difference of our proposed method from the method of Carlet et al. [4] is that the former uses GM polynomials instead of polynomials of low algebraic degrees such as 2 or 3. To a GM polynomial, the masking scheme for a multiplication can be applied, which is more efficient than the masking scheme for a polynomial of low algebraic degree presented by Carlet et al. [4]. Due to this property, for masked implementation, in terms of the number of evaluations of nonlinear functions (GM polynomials, polynomials of low algebraic degree or multiplications), the proposed decomposition method is more efficient than the method by Goudarzi et al. [8] for $n = 4, 6, 8$ and than the method by Carlet et al. [4] for $n = 4, 6$ and for $n = 8$ if the masking order is higher than 1.

Related Work. Ishai, Sahai and Wagner presented a higher-order masking method for multiplication over $\mathbb{F}_2$ in their seminal paper [9]. Rivain and Prouff [14] generalized the method of Ishai et al. [9] to any finite field multiplication and applied it to the AES S-box. Carlet et al. [3] extended the method of Rivain and Prouff [14] and proposed a generic method for masking any S-box based on cyclotomic classes and the Knuth-Eve polynomial evaluation algorithm [7, 10]. Coron, Roy and Vivek [6] improved the method of Carlet et al. [3] and presented a heuristic but generic method for masking any S-box. Goudarzi et al. [8] generalized the approach of Coron, Roy and Vivek [6] and proposed a method to treat any S-box from $\{0,1\}^{w_{\mathrm{i}}\nu}$ to $\{0,1\}^{w_{\mathrm{o}}\nu}$ as a tuple of polynomials over $\mathbb{F}_{2^\nu}$.

Inspired by the work of Coron, Roy and Vivek [6], Carlet et al. [4] introduced a new approach to decompose any S-box using polynomials having low algebraic degrees. They also presented masking methods for polynomials having low algebraic degrees.

Organization. Section 2 introduces some notations and definitions necessary for the discussions. Section 3 presents the proposed algebraic decomposition method using GM polynomials and its experimental results. Section 4 discusses

application of the proposed decomposition to masking. Section 5 gives a brief concluding remark.

2 Preliminaries

Let ν be a positive integer. Let $\mathbb{F}_{2^\nu}$ be the finite field with 2^ν elements.

2.1 Functions over Finite Fields

A function $h : \mathbb{F}_{2^\nu}^{w_\mathrm{i}} \to \mathbb{F}_{2^\nu}^{w_\mathrm{o}}$ is a tuple of functions $(h_0, h_1, \ldots, h_{w_\mathrm{o}-1})$, where $h_j : \mathbb{F}_{2^\nu}^{w_\mathrm{i}} \to \mathbb{F}_{2^\nu}$ for $0 \leq j \leq w_\mathrm{o} - 1$. h_j can be represented as

$$h_j(x_0, x_1, \ldots, x_{w_\mathrm{i}-1}) = \sum_{k_0=0}^{2^\nu-1} \cdots \sum_{k_{w_\mathrm{i}-1}=0}^{2^\nu-1} \alpha_{j,k_0,\ldots,k_{w_\mathrm{i}-1}} x_0^{k_0} \cdots x_{w_\mathrm{i}-1}^{k_{w_\mathrm{i}-1}} \ ,$$

where $\alpha_{j,k_0,\ldots,k_{w_\mathrm{i}-1}} \in \mathbb{F}_{2^\nu}$. We only refer to the cases that $(w_\mathrm{i}, w_\mathrm{o}) \in \{(1,1), (2,1), (2,2)\}$ in the remaining parts.

Definition 1 (Algebraic degree). *For a function $h : \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu}$ such that*

$$h(x) = \sum_{k=0}^{2^\nu-1} \alpha_k x^k \ ,$$

its algebraic degree is the maximum of $\mathrm{HW}(k)$ such that $\alpha_k \neq 0$ for $0 \leq k \leq 2^\nu - 1$, where $\mathrm{HW}(k)$ is the Hamming weight of the binary representation of k.

Definition 2 (Linearized polynomial). *A function $\ell : \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu}$ is called a linearized polynomial if it can be represented as*

$$\ell(x) = \sum_{k=0}^{\nu-1} \alpha_k x^{2^k} \ ,$$

where $\alpha_k \in \mathbb{F}_{2^\nu}$.

For any linearized polynomial $\ell(x)$, its algebraic degree is 1, and it holds that

$$\ell\Big(\sum_{i=0}^{d} x_i\Big) = \sum_{i=0}^{d} \ell(x_i) \ . \tag{1}$$

We introduce generalized multiplication polynomials, which are used in our proposed decomposition method:

Definition 3 (Generalized multiplication polynomial). *We call a function $m : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$ a generalized multiplication (GM) polynomial if it*

can be represented as $m(x) = (m_0(x), m_1(x))$ *such that, for* $b \in \{0,1\}$, $m_b : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu}$ *and*

$$m_b(x) = \sum_{k=0}^{\nu-1} \sum_{l=0}^{\nu-1} \alpha_{b,k,l} x_0^{2^k} x_1^{2^l} \ ,$$

where $x = (x_0, x_1) \in \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$ *and* $\alpha_{b,k,l} \in \mathbb{F}_{2^\nu}$.

For any GM polynomial $m(x_0, x_1)$, it holds that

$$m\Big(\sum_{i=0}^{d} x_{0,i}, \sum_{j=0}^{d} x_{1,j}\Big) = \sum_{i=0}^{d} \sum_{j=0}^{d} m(x_{0,i}, x_{1,j}) \tag{2}$$

since

$$\Big(\sum_{i=0}^{d} x_{0,i}\Big)^{2^k} \Big(\sum_{j=0}^{d} x_{1,j}\Big)^{2^l} = \Big(\sum_{i=0}^{d} x_{0,i}^{2^k}\Big)\Big(\sum_{j=0}^{d} x_{1,j}^{2^l}\Big) = \sum_{i=0}^{d} \sum_{j=0}^{d} x_{0,i}^{2^k} x_{1,j}^{2^l} \ . \tag{3}$$

For Eq. (3), multiplication is the case that $k = l = 0$.

3 Algebraic Decomposition

In the remaining parts of the paper, ν is a positive integer and $n = 2\nu$.

3.1 Algebraic decomposition using GM polynomials

Let $h : \{0,1\}^n \to \{0,1\}^n$. Then, h can be seen as $h(x) = (h_0(x_0, x_1), h_1(x_0, x_1))$, where $x = (x_0, x_1)$ and $h_b : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu}$ for $b \in \{0,1\}$. The decomposition of h proceeds as follows:

1. For $1 \le i \le r$, let $f_i : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$ is a GM polynomial chosen uniformly at random.
2. For $1 \le i \le r$, $g_i : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$ is defined as follows:

   $$g_1(x) = f_1(x) \ ,$$

   $$g_2(x) = f_2(g_1(x) + (\ell_{0,0}(x_0) + \ell_{0,1}(x_1), \ell_{1,0}(x_0) + \ell_{1,1}(x_1))) \ ,$$

   where $\ell_{0,0}$, $\ell_{0,1}$, $\ell_{1,0}$ and $\ell_{1,1}$ are linearized polynomials over $\mathbb{F}_{2^\nu}$ chosen uniformly at random, and, for $3 \le i \le r$,

   $$g_i(x) = f_i(g_{i-1}(x)) \ .$$

3. For $1 \le j \le t$, $q_j : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$ is defined as follows:

   $$q_j(x) = (q_{j,0}(x), q_{j,1}(x)) \ ,$$

where, for $b \in \{0,1\}$,

$$q_{j,b}(x) = \sum_{i=1}^{r} \ell_{j,i,b}(g_{i,b}(x)) + \ell_{j,0,b,0}(x_0) + \ell_{j,0,b,1}(x_1) \ , \tag{4}$$

and $\ell_{j,i,b}$, $\ell_{j,0,b,0}$ and $\ell_{j,0,b,1}$ are linearized polynomials over $\mathbb{F}_{2^\nu}$ chosen uniformly at random.

4. For $b \in \{0,1\}$, search GM polynomials $\mu_1, \ldots, \mu_t$ over $\mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$, linearized polynomials $\lambda_{0,b,0}, \lambda_{0,b,1}, \ldots, \lambda_{r,b,0}, \lambda_{r,b,1}$ over $\mathbb{F}_{2^\nu}$ and a constant $\delta_b \in \mathbb{F}_{2^\nu}$ satisfying

$$\begin{aligned} h_b(x) = \sum_{j=1}^{t} \mu_{j,b}(q_j(x)) + \sum_{i=1}^{r} \big(\lambda_{i,b,0}(g_{i,0}(x)) + \lambda_{i,b,1}(g_{i,1}(x))\big) \\ + \lambda_{0,b,0}(x_0) + \lambda_{0,b,1}(x_1) + \delta_b \ . \end{aligned} \tag{5}$$

If the search fails, then return to the first step.

The amount of computation for an evaluation of h based on the decomposition is summarized in Table 1.

Table 1. The amount of computation based on the proposed decomposition. Both the linearized polynomials and the additions are over $\mathbb{F}_\nu$.

The number of evaluations of GM polynomials	$r+t$
The number of evaluations of linearized polynomials	$2(r+2)(t+2)$
The number of evaluations of additions	$2(r+2)(t+2)$

Similar to the decomposition in [4], the search in the 4th step above can be done by solving a system of linear equations over $\mathbb{F}_{2^\nu}$:

$$A \cdot \boldsymbol{v}_b = \boldsymbol{c}_b \tag{6}$$

for $b \in \{0,1\}$. $\boldsymbol{c}_b$ is a 2^n-dimensional vector over $\mathbb{F}_{2^\nu}$ such that

$$\boldsymbol{c}_b = (h_b(e_1), h_b(e_2), \ldots, h_b(e_{2^n}))^{\mathrm{T}} \ ,$$

where $e_i = (e_{i,0}, e_{i,1}) \in \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$ and $e_{i_1} \neq e_{i_2}$ if $i_1 \neq i_2$. $\boldsymbol{v}_b$ is the vector of unknowns representing the coefficients of GM polynomials $\mu_{1,b}, \ldots, \mu_{t,b}$, linearized polynomials $\lambda_{0,b,0}, \lambda_{0,b,1}, \ldots, \lambda_{r,b,0}, \lambda_{r,b,1}$ and δ_b. The matrix A, which does not depend on the value of b, is defined as follows:

$$A = (A_{q_1} \ A_{q_2} \ \cdots \ A_{q_t} \ A_{g_{1,0}} \ \cdots \ A_{g_{r,0}} \ A_{g_{1,1}} \ \cdots \ A_{g_{r,1}} \ A_{e_{*,0}} \ A_{e_{*,1}} \ \mathbf{1}) \ .$$

$$A_{q_j} = \begin{pmatrix} Q_{j,1}^{(0,0)} & \cdots & Q_{j,1}^{(0,\nu-1)} & Q_{j,1}^{(1,0)} & \cdots & Q_{j,1}^{(1,\nu-1)} & \cdots & Q_{j,1}^{(\nu-1,0)} & \cdots & Q_{j,1}^{(\nu-1,\nu-1)} \\ Q_{j,2}^{(0,0)} & \cdots & Q_{j,2}^{(0,\nu-1)} & Q_{j,2}^{(1,0)} & \cdots & Q_{j,2}^{(1,\nu-1)} & \cdots & Q_{j,2}^{(\nu-1,0)} & \cdots & Q_{j,2}^{(\nu-1,\nu-1)} \\ & \cdots & & & \cdots & & \cdots & & \cdots & \\ Q_{j,2^n}^{(0,0)} & \cdots & Q_{j,2^n}^{(0,\nu-1)} & Q_{j,2^n}^{(1,0)} & \cdots & Q_{j,2^n}^{(1,\nu-1)} & \cdots & Q_{j,2^n}^{(\nu-1,0)} & \cdots & Q_{j,2^n}^{(\nu-1,\nu-1)} \end{pmatrix}$$

is a $2^n \times \nu^2$ matrix, where $Q_{j,i}^{(k,l)} = q_{j,0}(e_i)^{2^k} q_{j,1}(e_i)^{2^l}$ for $1 \le i \le 2^n$, $1 \le k \le \nu-1$ and $1 \le l \le \nu - 1$.

$$A_{g_{i,b'}} = \begin{pmatrix} g_{i,b'}(e_1)^{2^0} & g_{i,b'}(e_1)^{2^1} & \cdots & g_{i,b'}(e_1)^{2^{\nu-1}} \\ g_{i,b'}(e_2)^{2^0} & g_{i,b'}(e_2)^{2^1} & \cdots & g_{i,b'}(e_2)^{2^{\nu-1}} \\ & \cdots & & \\ g_{i,b'}(e_{2^n})^{2^0} & g_{i,b'}(e_{2^n})^{2^1} & \cdots & g_{i,b'}(e_{2^n})^{2^{\nu-1}} \end{pmatrix}$$

is a $2^n \times \nu$ matrix for $1 \le i \le r$ and $b' \in \{0, 1\}$.

$$A_{e_{*,b'}} = \begin{pmatrix} e_{1,b'}^{2^0} & e_{1,b'}^{2^1} & \cdots & e_{1,b'}^{2^{\nu-1}} \\ e_{2,b'}^{2^0} & e_{2,b'}^{2^1} & \cdots & e_{2,b'}^{2^{\nu-1}} \\ & \cdots & & \\ e_{2^n,b'}^{2^0} & e_{2^n,b'}^{2^1} & \cdots & e_{2^n,b'}^{2^{\nu-1}} \end{pmatrix}$$

is a $2^n \times \nu$ matrix. $\mathbf{1}$ is the column vector whose 2^n coordinates equal 1.

The matrix A has 2^n rows and $t \cdot \nu^2 + 2(r+1)\nu + 1$ columns. In order for the system of linear equations Eq.(6) to have a solution for any $\boldsymbol{c}_b$, the rank of A must be 2^n and it is required that

$$t \cdot n^2/4 + (r+1)n + 1 \ge 2^n \ . \tag{7}$$

It is also required that the algebraic degree of the polynomial in the right side of Eq.(5) is $n/2$ with respect to each of x_0 and x_1. Thus,

$$2^r \ge n/2 \ . \tag{8}$$

Once we obtain a matrix A with its rank 2^n from some $g_1, \ldots, g_r$ and $q_1, \ldots, q_t$, we can use it to decompose any function $h : \{0,1\}^n \to \{0,1\}^n$.

3.2 Experimental Result

Table 2 shows the values of parameters of successful decomposition minimizing the number of GM polynomials, that is, $r + t$. For $n = 4, 6$, all optimal values satisfying inequalities (7) and (8) and minimizing $r + t$ are achieved. For $n = 8$, optimal values are also achieved. On the other hand, though $(r, t) = (2, 15)$ are also optimal, they cannot be achieved with one hundred trials. For $n = 10$, all optimal values $(r, t) = (3, 40), (4, 39)$ as well as $(r, t) = (3, 41), (4, 40)$ cannot be achieved with one hundred trials.

In Table 2, for $n = 4$ and $(r, t) = (1, 2)$, four linearized polynomials used to compose g_2 is not necessary for the proposed algebraic decomposition algorithm in Sect. 3.

As an example, a decomposition of the 4-bit S-box of the tweakable block cipher SKINNY is presented in Appendix B.

Table 3 shows the smallest number of nonlinear functions achieved by our decomposition and the decomposition methods by Carlet et al. [4] and by Goudarzi

Table 2. Achievable parameters minimizing $r+t$. #GMP represents the number of GM polynomials. #LinP represents the number of linearized polynomials. #Add represents the number of additions.

n	(r,t)	#GMP	#LinP	#Add
4	$(1,2)$	3	20	20
	$(2,1)$	3	24	24
6	$(2,5)$	7	56	56
8	$(3,14)$	17	160	160
10	$(5,39)$	44	574	574

et al. [8]. For algebraic decomposition by Carlet et al., methods using polynomials of algebraic degrees 2 and/or 3 were presented, and the most efficient method was shown to be the method using only quadratic polynomials (polynomials of algebraic degree 2), which is mentioned in Table 3. The decomposition method to reduce the number of multiplications by Goudarzi et al. [8] is able to process any function over $\{0,1\}^n$ by regarding it as a function over $\mathbb{F}_\xi^{n/\xi}$ for any ξ such that $\xi \mid n$. Table 3 mentions only the case that $\xi = n/2$.

In terms of the number of multiplications or GM polynomials, our decomposition is slightly more efficient than the decomposition by Goudarzi et al. [8]. On the other hand, if implementaion adopts table lookup for evaluation of multiplications or GM polynomials, then our decomposition needs a lookup table for each GM polynomial, while the decomposition by Goudarzi et al. needs just a single lookup table for multiplication. Thus, the total table size for our decomposition is $2(r+t)$ times as large as that for the decomposition by Goudarzi et al. For example, for $n = 8$, the total table size of GM polynomials for our decomposition is $4352 (= 17 \times 256)$ Bytes.

In terms of the number of quadratic polynomials or GM polynomials, our decomposition does not seem so good as decomposition by Carlet et al. [4] apparently. We will see in the next section, however, our decomposition is more effective than the decomposition by Carlet et al. [4] for masked implementation.

Table 3. Comparison of best achievable parameters

	$n=4$	$n=6$	$n=8$
# quadratic polynomials [4]	3	5	11
# multiplications [8]	4	9	18
# GM polynomials (Ours)	3	7	17

4 Application to Masking

Algorithm 1 presents an algorithm of d-th order masking for a GM polynomial. Due to the property of GM polynomials shown by Eq.(3), it is similar to d-th order masking for multiplication. For reference, the algorithm of d-th order masking for a quadratic polynomial [4] is shown in Appendix A.

Algorithm 1: d-th order masking for a GM polynomial $m : \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu} \to \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$, where $\nu = n/2$

input : Shares $(a_0, a_1, \ldots, a_d)$ of a and $(b_0, b_1, \ldots, b_d)$ of b
output: Shares $(c_0, c_1, \ldots, c_d)$ of $c = m(a, b)$
for $i = 0$ **to** d **do**
  for $j = i + 1$ **to** d **do**
    $r_{i,j} \twoheadleftarrow \mathbb{F}_{2^\nu} \times \mathbb{F}_{2^\nu}$;
    $r_{j,i} \leftarrow (r_{i,j} + m(a_i, b_j)) + m(a_j, b_i)$;
for $i = 0$ **to** d **do**
  $c_i \leftarrow m(a_i, b_i)$;
  for $j = 0$ **to** d **do**
    if $j \neq i$ **then**
      $c_i \leftarrow c_i + r_{i,j}$;
return $(c_0, c_1, \ldots, c_d)$

Table 4 shows complexity of d-th order masking for an evaluation of a quadratic polynomial or a GM polynomial. Roughly, d-th order masking for a GM polynomial is twice as efficient as that for a quadratic polynomial. From Tables 3 and 4, in terms of the number of evaluations of nonlinear functions (quadratic polynomials or GM polynomials), the proposed decomposition yields more efficient masking for n-bit S-boxes than the decomposition using quadratic polynomials by Carlet et al. [4] for $n = 4, 6$, and for $n = 8$ if $d \geq 2$.

Table 4. Complexity of d-th order masking. "# eval," "# rand" and "# add," represent the required number of evaluations of a nonlinear function (a quadratic polynomial or a GM polynomial), random sequences and additions, respectively.

	# eval	# rand	# add
Quadratic poly. eval.	$(d+1)(2d+1)$	$d(d+1)$	$9d(d+1)/2+1$
GP poly. eval. (Algorithm 1)	$(d+1)^2$	$d(d+1)/2$	$2d(d+1)$

5 Conclusion

We have presented an algebraic decomposition method for masked implementation of any S-box. Essentially, our proposal is to use GM polynomials instead of polynomials with low algebraic degrees for decomposition. Future work is performance evaluation of masked implementaion of S-boxes using the proposed decomposition method.

Acknowledgements

The author was supported in part by JSPS KAKENHI Grant Number JP18H05289.

A Masking for Quadratic Polynomial

Algorithm 2 presents an algorithm of d-th order masking for a quadratic polynomial [4].

Algorithm 2: d-th order masking for a quadratic polynomial $f : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$

input : Shares $(a_0, a_1, \ldots, a_d)$ of a
output: Shares $(b_0, b_1, \ldots, b_d)$ of $b = f(a)$
for $i = 0$ **to** d **do**
  for $j = i + 1$ **to** d **do**
    $r_{i,j} \twoheadleftarrow \mathbb{F}_{2^n}$; $r'_{i,j} \twoheadleftarrow \mathbb{F}_{2^n}$;
    $r_{j,i} \leftarrow r_{i,j} + f(a_i + r'_{i,j}) + f(a_j + r'_{i,j}) + f((a_i + r'_{i,j}) + a_j) + f(r'_{i,j})$;
for $i = 0$ **to** d **do**
  $b_i \leftarrow f(a_i)$;
  for $j = 0$ **to** d **do**
    if $j \neq i$ **then**
      $b_i \leftarrow b_i + r_{i,j}$;
if d *is odd* **then**
  $b_1 \leftarrow b_1 + f(0)$;
return $(b_0, b_1, \ldots, b_d)$

B Decomposition of SKINNY 4-Bit S-box

A decomposition of the 4-bit S-box of the tweakable block cipher SKINNY [1], which is given in Table 5, is presented.

Table 5. The 4-bit S-box of SKINNY

input	0 1 2 3 4 5 6 7 8 9 a b c d e f
output	c 6 9 0 1 a 2 b 3 8 5 d 4 e 7 f

For the algebraic decomposition algorithm in Sect. 3, let $(r,t) = (1,2)$. For $f_1(x_0,x_1) = (f_{1,0}(x_0,x_1), f_{1,1}(x_0,x_1))$,

$$f_{1,0}(x_0,x_1) = 0x_0x_1 + 2x_0x_1^2 + 0x_0^2x_1 + 3x_0^2x_1^2 \ ,$$
$$f_{1,1}(x_0,x_1) = 0x_0x_1 + 1x_0x_1^2 + 2x_0^2x_1 + 3x_0^2x_1^2 \ .$$

For Eq.(4),

$$q_{1,b}(x_0,x_1) = \ell_{1,1,b}(g_{1,b}(x_0,x_1)) + \ell_{1,0,b,0}(x_0) + \ell_{1,0,b,1}(x_1) \ ,$$
$$q_{2,b}(x_0,x_1) = \ell_{2,1,b}(g_{1,b}(x_0,x_1)) + \ell_{2,0,b,0}(x_0) + \ell_{2,0,b,1}(x_1) \ ,$$

where

$$\ell_{1,1,0}(z) = 0z + 1z^2 \quad \ell_{1,0,0,0}(x_0) = 3x_0 + 0x_0^2 \quad \ell_{1,0,0,1}(x_1) = 0x_1 + 1x_1^2 \ ,$$
$$\ell_{1,1,1}(z) = 1z + 1z^2 \quad \ell_{1,0,1,0}(x_0) = 0x_0 + 2x_0^2 \quad \ell_{1,0,1,1}(x_1) = 0x_1 + 1x_1^2 \ ,$$

and

$$\ell_{2,1,0}(z) = 2z + 2z^2 \quad \ell_{2,0,0,0}(x_0) = 1x_0 + 3x_0^2 \quad \ell_{2,0,0,1}(x_1) = 3x_1 + 3x_1^2 \ ,$$
$$\ell_{2,1,1}(z) = 2z + 1z^2 \quad \ell_{2,0,1,0}(x_0) = 1x_0 + 3x_0^2 \quad \ell_{2,0,1,1}(x_1) = 2x_1 + 0x_1^2 \ .$$

For Eq.(5),

$$h_b(x) = \mu_{1,b}(q_1(x)) + \mu_{2,b}(q_2(x)) + \lambda_{1,b,0}(g_{1,0}(x)) + \lambda_{1,b,1}(g_{1,1}(x)) + \lambda_{0,b,0}(x_0) + \lambda_{0,b,1}(x_1) + \delta_b \ ,$$

where

$$\mu_{1,0}(x_0,x_1) = 0x_0x_1 + 0x_0x_1^2 + 1x_0^2x_1 + 2x_0^2x_1^2 \ ,$$
$$\mu_{1,1}(x_0,x_1) = 0x_0x_1 + 0x_0x_1^2 + 1x_0^2x_1 + 2x_0^2x_1^2 \ .$$

$$\mu_{2,0}(x_0,x_1) = 2x_0x_1 + 2x_0x_1^2 + 3x_0^2x_1 + 0x_0^2x_1^2 \ ,$$
$$\mu_{2,1}(x_0,x_1) = 2x_0x_1 + 1x_0x_1^2 + 3x_0^2x_1 + 0x_0^2x_1^2 \ .$$

$$\lambda_{1,0,0}(z) = 3z + 3z^2 \qquad \lambda_{1,1,0}(z) = 0z + 0z^2$$
$$\lambda_{1,0,1}(z) = 0z + 0z^2 \qquad \lambda_{1,1,1}(z) = 1z + 0z^2$$

$$\lambda_{0,0,0}(x_0) = 0z + 0z^2 \qquad \lambda_{0,1,0}(x_0) = 3x_0 + 2x_0^2$$
$$\lambda_{0,0,1}(x_1) = 2z + 3z^2 \qquad \lambda_{0,1,1}(x_1) = 1x_0 + 0x_0^2$$

$\delta_0 = 3$ and $\delta_1 = 0$.

References

1. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 123–153. Springer (2016). https://doi.org/10.1007/978-3-662-53008-5_5,
2. Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of AFIPS 1979 National Computer Conference. vol. 48, pp. 313–317 (1979)
3. Carlet, C., Goubin, L., Prouff, E., Quisquater, M., Rivain, M.: Higher-order masking schemes for S-boxes. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 366–384. Springer (2012). https://doi.org/10.1007/978-3-642-34047-5_21
4. Carlet, C., Prouff, E., Rivain, M., Roche, T.: Algebraic decomposition for probing security. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 742–763. Springer (2015). https://doi.org/10.1007/978-3-662-47989-6_36
5. Chari, S., Jutla, C.S., Rao, J.R., Rohatgi, P.: Towards sound approaches to counteract power-analysis attacks. In: Wiener, M.J. (ed.) CRYPTO '99. LNCS, vol. 1666, pp. 398–412. Springer (1999). https://doi.org/10.1007/3-540-48405-1_26
6. Coron, J., Roy, A., Vivek, S.: Fast evaluation of polynomials over binary finite fields and application to side-channel countermeasures. In: Batina, L., Robshaw, M. (eds.) CHES 2014. LNCS, vol. 8731, pp. 170–187. Springer (2014). https://doi.org/10.1007/978-3-662-44709-3_10
7. Eve, J.: The evaluation of polynomials. Numerische Mathematik **6**, 17–21 (1964)
8. Goudarzi, D., Rivain, M., Vergnaud, D., Vivek, S.: Generalized polynomial decomposition for S-boxes with application to side-channel countermeasures. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 154–171. Springer (2017). https://doi.org/10.1007/978-3-319-66787-4_8
9. Ishai, Y., Sahai, A., Wagner, D.A.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer (2003). https://doi.org/10.1007/978-3-540-45146-4_27
10. Knuth, D.E.: Evaluation of polynomials by computer. Commun. ACM **5**(12), 595–599 (1962). https://doi.org/10.1145/355580.369074
11. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO '96. LNCS, vol. 1109, pp. 104–113. Springer (1996). https://doi.org/10.1007/3-540-68697-5_9
12. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M.J. (ed.) CRYPTO '99. LNCS, vol. 1666, pp. 388–397. Springer (1999). https://doi.org/10.1007/3-540-48405-1_25
13. Messerges, T.S.: Securing the AES finalists against power analysis attacks. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 150–164. Springer (2000). https://doi.org/10.1007/3-540-44706-7_11
14. Rivain, M., Prouff, E.: Provably secure higher-order masking of AES. In: Mangard, S., Standaert, F. (eds.) CHES 2010. LNCS, vol. 6225, pp. 413–427. Springer (2010). https://doi.org/10.1007/978-3-642-15031-9_28
15. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)