Securing SMS4 Cipher against Differential Power Analysis and Its VLSI Implementation

Xuefei Bai, Yanhua Xu and Li Guo
Department of Electronic Science and Technology
University of Science and Technology of China
Hefei, Anhui 230026, P. R. China
Email: baixf@ustc.edu.cn

Abstract—Differential power analysis is of great concern because it can be used to break implementations of almost any symmetric or asymmetric algorithm, and several countermeasures have been proposed to protect implementations of cryptographic algorithms except SMS4 cipher. In the present paper, we focus on the differential power analysis attack on SMS4 cipher, and suggest a secure masking scheme for SMS4 cipher, which is particularly suited for implementation in dedicated hardware. The masking scheme for the inversion presented in this article is based on composite field arithmetic, in which the inversion is shifted from $GF(2^8)$ down to $GF(2^2)$. In addition, several methods such as module reuse and changing computing order are employed to reduce circuit area and maintain its speed. Using SMIC 0.18μm CMOS technology, the area of this improved SMS4 cipher is only about 25k-gates and the frequency could be up to 50MHz.

I. INTRODUCTION

In January 2006, the Office of State Commercial Cipher Administration of China (OSCCA) declassified and published the specification of SMS4 block cipher [1]. This block cipher is used in the Wireless Authentication and Privacy Infrastructure (WAPI) standard for protecting data packets in wireless network. The release of SMS4 cipher has greatly promoted the research and development of the cipher localization. However, as its extensive application, SMS4 cipher will suffer from more attacks. Since SMS4 cipher was disclosed only a short time ago, few studies have been published on its security.

Power analysis attacks are much more powerful in breaking implementations of ciphers because they do not require expensive resources. In this paper, we focus on differential power analysis (DPA), which was firstly introduced by Kocher et al. [2] in 1999 to attack AES algorithm. Since then, several countermeasures have been proposed, including masking methods. The method suggested by Oswald [3] is suitable for hardware implementation, and this improved AES cipher is resistant to zero value attack and all other DPA. However, no countermeasure for SMS4 cipher resistant to DPA has appeared in the open literatures.

In this study, since SMS4 cipher and AES cipher both have inversion-based mapping in S-Box design, which was suggested by Liu Fen et al. [4], we could adopt Oswald’s $GF(2^8)$ inversion method, but we must adapt it slightly to have it be suitable for SMS4 cipher. Then we proposed a new implementation of the SMS4 algorithm that was secure against the DPA attack.

The rest of this paper is organized as follows. Section II describes the SMS4 block cipher. The principle of DPA on SMS4 cipher is introduced briefly in section III. In section IV, the $GF(2^8)$ inversion method suggested by Oswald is described. In section V, we propose an improved SMS4 algorithm and describe in details. Finally, the simulation results and discussions are presented in section VI. While, conclusions are reported in section VII.

II. SMS4 BLOCK CIPHER

SMS4 is a 32-round iterative algorithm, and both the data block and the key size are fixed to 128 bits [1]. The 128-bit data block is divided into 4 words each with 32-bit length and each word are mapped to a state. Then the input data block is mapped to initial four states, and all the internal iterations operate on former states to generate new states. There are 36 states which are denoted by $X_i$, $i \in \{0,1,...,35\}$. The block diagram of the SMS4 cipher is showed in Fig. 1.

A. Encryption Algorithm

Let $X = (X_0, X_1, X_2, X_3) \in (GF(2^{32}))^4$ be the plaintext and $Y = (Y_0, Y_1, Y_2, Y_3) \in (GF(2^{32}))^4$ be the ciphertext. Let $rk_i \in GF(2^{32})$ denote the $i$-th subkey and
\((X_1, X_{i+1}, X_{i+2}, X_{i+3})\) denote the \((i + 1)\)-th round inputs, \(i \in \{0, 1, \ldots, 31\}\). Then the SMS4 scheme can be written as \(X_{i+4} = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, r_k_i)\)
\(= X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus r_k_i)\)\(^{(1)}\)
and \(Y_0, Y_1, Y_2, Y_3) = (X_{35}, X_{34}, X_{33}, X_{32})\)\(^{(2)}\)
where \(i \in \{0, 1, \ldots, 31\}\), \(F\) is the round function and \(T\) is the composite transformation.

The transformation \(T : GF(2^{32}) \rightarrow GF(2^{32})\) is composed of the nonlinear transformation \(\tau\) and the linear transformation \(L\):
\[T(\cdot) = L(\tau(\cdot))\]\(^{(3)}\)

The transformation \(\tau\) includes four 8-bit nonlinear S-boxes in parallel. Let denoted by \(A = (a_0, a_1, a_2, a_3) \in (GF(2^8))^4\) the input of \(\tau\) and by \(B = (b_0, b_1, b_2, b_3) \in (GF(2^8))^4\) the output. Then \(\tau\) can be defined as
\[B = (b_0, b_1, b_2, b_3) = \tau(A) = (Sbox(a_0), Sbox(a_1), Sbox(a_2), Sbox(a_3))\]\(^{(4)}\)
where \(Sbox(.)\) is the S-box byte substitution. The S-box table can be found in [1] and the algebraic structure of the S-box was described in Section V.

The output of \(\tau\) is \(B\), which is also the input of the linear transformation \(L\). Let denoted by \(C \in GF(2^{32})\) the output of \(L\). Then \(L\) can be defined as
\[C = L(B) = B \oplus (B \ll 2) \oplus (B \ll 10) \oplus (B \ll 18) \oplus (B \ll 24)\]\(^{(5)}\)
where \(\ll i\) denotes a 32-bit cyclic left shift by \(i\) positions.

**B. Decryption Process**
SMS4 is a symmetric-key cipher, in which both the sender and the receiver use a single key for encryption and decryption. The decryption procedure of SMS4 can be done in the same way as the encryption procedure by reversing the order of the subkeys.

**C. Key Schedule**

The process of the key schedule in SMS4 cipher has the same structure as that in encryption process except for \(L\) function. Let \(MK = (MK_0, MK_1, MK_2, MK_3) \in (GF(2^{32}))^4\) denote the cipher key, \(r_k_i \in GF(2^{32}), i \in \{0, 1, \ldots, 31\}\) denote the round keys, and \(K_i \in GF(2^{32}), i \in \{0, 1, \ldots, 35\}\). Then key schedule algorithm is defined as
\[(K_0, K_1, K_2, K_3) = (MK_0 \oplus FK_0, MK_1 \oplus FK_1, MK_2 \oplus FK_2, MK_3 \oplus FK_3)\]\(^{(6)}\)
and \(r_k_i = K_{i+4} = K_i \oplus T'(K_{i+1} \oplus K_{i+2} \oplus K_{i+3} \oplus CK_i)\)\(^{(7)}\)
where \(FK_i, i \in \{0, 1, 2, 3\}\) is system parameters, \(CK_i, i \in \{0, 1, 2, 3\}\) is key constants, and \(T'(\cdot)\) is a transformation similar to \(T\) in the encryption process.

The only difference between \(T\) and \(T'\) is the linear transformation. Instead of \(L\), the following transformation \(L'\) is used in \(T'\):
\[L'(B) = B \oplus (B \ll 13) \oplus (B \ll 23)\]\(^{(8)}\)
The system parameters \(FK_i, i \in \{0, 1, 2, 3\}\) are defined as \((a3b1bac6, 56a33350, 677d9197)\) and \((b27022dc)\) in hexadecimal, respectively.

The key constants \(CK_i = (ck_{i,0}, ck_{i,1}, ck_{i,2}, ck_{i,3}) \in (GF(2^8))^4\) are computed as follows:
\[ck_{i,j} = (4 \times i + j) \times 7(\text{mod} 256)\]\(^{(9)}\)
where \(i \in \{0, 1, \ldots, 31\}\), and \(j \in \{0, 1, 2, 3\}\).

**III. DPA on SMS4 Cipher**

The DPA attack developed by Paul Kocher and Cryptographic Research [2] as an attack on AES cipher is particularly impressive because it uses simple mathematical tools and techniques that are independent of the implementation of the cryptographic algorithm. Moreover, it requires a set of power consumption curves obtained by running the cryptographic algorithm a number of times for the same secret key and different inputs, and then makes use of statistical analysis of the power dissipation during the encryption or decryption process of a cryptographic device.

The DPA attack on SMS4 cipher, which was suggested in [5], can be performed as follows.

Step 1: First we must perform \(M\) encryption operations using \(M\) random plaintexts \(X_m, m \in \{1, \ldots, M\}\) and a same key, and capture the power traces \(T(X_m, t), t \in \{1, \ldots, k\}\) and the ciphertexts \(Y_m, k\) is the sample number.

Step 2: DPA on the round key. The 32-bit round key to be attacked can be separated into 8-bit blocks to be considered respectively.
Let \(r_{k_i,j}\) denote the \(j\)-th block of the \(i\)-th round key and \(W(.)\) denote the Hamming weight of a variable. Moreover, \(r_{k_i,j} \in \{0, 1, \ldots, 255\}\) is a hypothesis of \(r_{k_i,j}\).

Next, we define the selection function for DPA on SMS4 cipher by
\[D(Y_m, j, r_{k_{i,j},s}) = W(b_j) - W(b_j')\]\(^{(10)}\)
where \(Y_m = (X_{i+1}, X_{i+2}, X_{i+3}, X_{i+4})m, m \in \{1, \ldots, M\}\) denote the outputs of the \(M\) encryption operations. Define the power difference by
\[\Delta T(X_m, t) = T(X_m, t) - \overline{T(X_m, t)}\]\(^{(11)}\)
Then we can define the Pearson’s linear correlation coefficients as
\[r(t, j, r_{k_{i,j},s}) = \frac{\sum_m D(Y_m, j, r_{k_{i,j},s}) \Delta T(X_m, t)}{\sqrt{\sum_m D(Y_m, j, r_{k_{i,j},s})^2} \sqrt{\sum_m \Delta T(X_m, t)^2}}\]\(^{(12)}\)
If \(r(t, j, r_{k_{i,j},s})\) is near zero, the hypothesis is wrong. If \(r(t, j, r_{k_{i,j},s})\) indicates a strong correlation at some points in time, the hypothesis is correct. Then we can find out the correct hypothesis of the round key from the curves of \(r(t, j, r_{k_{i,j},s})\) for every \(r_{k_{i,j},s}\) and every \(j\) [6].
Step 3: Encryption key calculation. If the DPA attack on the 32nd round key is performed with success in Step 2, we can get the round key \( r_{k31} \). Suppose that the outputs \((Y_0, Y_1, Y_2, Y_3)\) for every encryption operation are captured during the DPA process. Then \((X_{32}, X_{33}, X_{34}, X_{35})\) will be gotten and we can calculate out \( X_{31} \) from (1). The DPA attack on the 31st round can be performed now and \( r_{k30} \) and \( r_{k30} \) will be found. Similarly, we can get \( r_{k29} \) and \( r_{k28} \).

We can obtain all the round keys \( r_k \) and \((K_0, K_1, K_2, K_3)\) from (7) by the round keys of the last four rounds, \( (r_{k29}, r_{k29}, r_{k30}, r_{k31}) \). Then the encryption key \( MK \) can be calculated out from (6) and the DPA attack is completed.

IV. OSWALD’S \( GF(2^8) \) INVERSION ALGORITHM

Oswald suggested a new masking method [3], which combined additive masks and multiplicative masks. The major advantage of this method is that all the data are additively masked in the advantage of this method is that all the data are additively masked and multiplicative masks. The major advantage of this method is that all the data are additively masked and multiplicative masks. Let \( M(K) \) mask to the composite field \( GF(2^8) \). We can obtain all the round keys \( r_k \) and \((K_0, K_1, K_2, K_3)\) from (7) by the round keys of the last four rounds, \( (r_{k29}, r_{k29}, r_{k30}, r_{k31}) \). Then the encryption key \( MK \) can be calculated out from (6) and the DPA attack is completed.

\[
\begin{align*}
\text{Step 3: Encryption key calculation. If the DPA attack on the} \ & \ 32\text{nd round key is performed with success in Step 2, we can get the round key } r_{k31}. \ & \ \text{Suppose that the outputs} \ (Y_0, Y_1, Y_2, Y_3) \text{for every encryption operation are captured during the DPA process. Then} \ (X_{32}, X_{33}, X_{34}, X_{35}) \text{will be gotten and we can calculate out } X_{31} \text{ from (1). The DPA attack on the 31st round can be performed now and } r_{k30} \text{ and } r_{k30} \text{ will be found. Similarly, we can get } r_{k29} \text{ and } r_{k28}. \\
\text{We can obtain all the round keys } r_k \text{ and } (K_0, K_1, K_2, K_3) \text{ from (7) by the round keys of the last four rounds, } (r_{k29}, r_{k29}, r_{k30}, r_{k31}). \text{ Then the encryption key } MK \text{ can be calculated out from (6) and the DPA attack is completed.}
\end{align*}
\]

\[
\begin{align*}
\text{IV. OSWALD’S } GF(2^8) \text{ INVERSION ALGORITHM} & \\
\text{Oswald suggested a new masking method [3], which combined additive masks and multiplicative masks. The major advantage of this method is that all the data are additively masked in the } GF(2^8) \text{ inversion part, which is not only safe under common DPA attacks, but also resistant to the zero-value attack.}
\end{align*}
\]

The \( GF(2^8) \) inversion algorithm of Oswald’s method is unique. The mathematical fundamentals are as follows [3].

Each element of \( GF(2^8) \) is represented as a linear polynomial \( a_h x + a_l \) over \( GF(2^4) \). Then inversion of such a polynomial can be computed using operations in \( GF(2^4) \) as (13) to (16) only:

\[
\begin{align*}
(a_h x + a_l)^{-1} &= a_h' x + a_l' & (13) \\
a_h' &= a_h d^{-1} & (14) \\
a_l' &= (a_h + a_l) d^{-1} & (15) \\
d &= a_h^2 v + a_l^2 + a_h a_l & (16)
\end{align*}
\]

where \( v \) is defined in accordance with the field polynomial which is used to define the quadratic extension of \( GF(2^4) \).

In the masking scheme for inversion, all intermediate values as well as the input and output must be masked additively. Let the input data be \( a + m \), where \( a \) is the original input data and \( m \) is the mask over \( GF(2^8) \). In order to calculate the inversion of a masked input data, we first map the data as well as the mask to the composite field \( GF(2^8) \), the value that needs to be inverted can be represented by \( (a_h + m_h) x + (a_l + m_l) \) over \( GF(2^4) \). Then (13) have to be modified as:

\[
((a_h + m_h) x + (a_l + m_l))^{-1} = (a_h' + m_h') x + (a_l' + m_l') & (17)
\]

In the implementation in [3], the functions as shown in (18) to (20) were derived. By these functions we can calculate out \( a^{-1} + m \) from the inputs \( a + m \) and \( m \).

\[
a_h' + m_h = \left(a_h + m_h\right)\left(d^{-1} + m_l\right) + \left(d^{-1} + m_l\right) m_h \\
+ \left(a_h + m_h\right) m_l + m_h m_l + m_h & (18)
\]

All the calculations including multiplication, square and square with multiplication in (18) to (20) are performed in \( GF(2^4) \). In addition, calculating the inverse \( d^{-1} + m_l \) in \( GF(2^4) \) can be reduced to calculate the inverse in \( GF(2^2) \) by representing \( GF(2^4) \) as a quadratic extension of \( GF(2^2) \), which is that transforming \( d^{-1} + m_l \) from \( GF(2^4) \) to \( GF(2^2) \). Then we can use the same formulae as (18) to (20) to perform the calculations in \( GF(2^2) \). However, the transformation matrix for changing basis from \( GF(2^4) \) to \( GF(2^2) \) is different from that for changing basis from \( GF(2^8) \) to \( GF(2^2) \). Finally, in \( GF(2^2) \), the inversion operation is equivalent to square, that is \((x + m)^{-1} = (x + m)^2 = x^2 + m^2 \), which also makes the cost of the implementation much lower.

V. IMPROVED SMS4 CIPHER

In order to achieve security, we propose an improved SMS4 cipher by using a combination of additive and multiplicative masking. The main idea of securing SMS4 with masking is following. The message is masked by means of a traditional XOR operation with some random \( M \) at the beginning of the algorithm, and thereafter everything is almost as usual, the mask must be removed in order to reestablish the expected value at the end of the execution. Since the attacker cannot predict random number, the power output will not be estimated by hypothetical model.

A. Algebraic Structure of SMS4 S-box

S-box is usually implemented as a look-up table consisting of 256 entries, and each entry is 8-bit wide. However, for hardware implementation of the masking scheme, one drawback of this method is a significant allocation of hardware resources. In contrast, a new structure of SMS4 S-box suggested by [4], is suitable for hardware implementation. The S-box is of the form

\[
\begin{align*}
\text{Sbox}(a) &= I(a \cdot A_1 + C_1) \cdot A_2 + C_2 & (21)
\end{align*}
\]

where \( I \) is the patched inversion over \( GF(2^8) \), the matrices \( A_1, A_2 \in GL(8, 2) \), and the vectors \( C_1, C_2 \in GF(2)^8 \). The cyclic matrices and the row vectors in the algebraic expression
are as follows:

\[ A_1 = A_2 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \]

\[ C_1 = C_2 = (1, 1, 0, 0, 1, 0, 1, 1) \] (22)

The irreducible polynomial is

\[ M(x) = x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + 1 \] (23)

S-box described in (21) is the composition of two transformations, which is affine transformations and multiplicative inversion in \( GF(2^8) \). The latter is nonlinear part.

The most tricky part when masking SMS4 is to mask its nonlinear part. All other operations are linear and can be masked in a straightforward manner. Next, we will focus on the \( GF(2^8) \) inversion algorithm with masking for SMS4 cipher.

**B. \( GF(2^8) \) inversion algorithm for SMS4 cipher**

The calculation over \( GF(2^8) \) is complicated, so the method suggested by Oswald introduced subfield arithmetic in the crucial step of computing an inverse in \( GF(2^8) \), that is reducing an 8-bit calculation to several 4-bit ones, and then breaking up the 4-bit calculations into 2-bit ones. In our method we adopt Oswald’s method, but must adapt the isomorphic mapping because it is correlated with the cipher and the composite field. In this paper we use the method which was suggested in [7] to construct the isomorphic mapping as follows.

The field polynomial of \( GF(2^8) \) is (23). Let the representation \( GF((2^4)^2) \) generated by \( P(x) = x^2 + x + \omega^{14} \) and the field polynomial of \( GF(2^4) \) be \( Q(x) = x^4 + x + 1 \). Suppose that \( \nu \) in (20) is equal to \( \omega^{14} = \{1001\} \), where \( \omega \in GF(2^4) \). Then the change of basis matrix for \( GF(2^8) \) to \( GF((2^4)^2) \) is given below:

\[ T = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}, T^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \] (24)

Similarly, The field polynomial of \( GF(2^4) \) is \( M(x) = x^4 + x + 1 \). Let the representation \( GF((2^2)^2) \) generated by \( P(x) = x^2 + x + \omega \) and the field polynomial of \( GF(2^2) \) be \( Q(x) = x^2 + x + 1 \). Let \( \nu \) here be \( \omega = \{10\} \), where \( \omega \in GF(2^2) \). Then the change of basis matrix for \( GF(2^4) \) to \( GF((2^2)^2) \) can be described as:

\[ T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, T^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \] (25)

As mentioned in Oswald’s method, the 8-bit inverse in \( GF(2^8) \) is down into 4-bit operations in \( GF(2^4) \), which can further be broken down to still 2-bit operations in \( GF(2^2) \). In \( GF(2^2) \), the inversion operation is equivalent to square. So this modified \( GF(2^8) \) inversion algorithm can reduce the complexity of calculation and then decrease the hardware implementation cost.

In order to prevent unmasking the operands, we must introduce another additive mask before all the other mask correction terms are added. This mask could be new or reused bits from the original mask \( M \), here we choose the reused bits from \( M \), which can be represented by \( n_h \cdot x + n_l \) over \( GF(2^4) \), where \( n_h \) and \( n_l \) are values mapped from \( GF(2^8) \), then (18) and (19) can be modified into (26) and (27) as follows:

\[ a_h' + m_h = (dm_4 + c_7 + n_h) + (c_1 + c_5 + c_6 + n_h) \] (26)

\[ a_l' + m_l = (a_h' + m_h + dm_5 + c_8 + n_l) + (c_2 + c_5 + c_6 + c_9 + n_l) \] (27)

Next we go through the calculation of S-box. If we assume the input data as \( a \) and the mapping matrix as \( T \). Then for a non-masked input we have:

1. \( a \cdot A_1 + C_1 \)
2. \( T \cdot (a \cdot A_1 + C_1) \)
3. \( T \cdot (a \cdot A_1 + C_1) \rightarrow I(T \cdot (a \cdot A_1 + C_1)) \rightarrow I(a \cdot A_1 + C_1) \cdot I(T) \)
4. \( I(a \cdot A_1 + C_1) \cdot I(T) \cdot T \cdot A_2 + C_2 \)
   \[ = I(a \cdot A_1 + C_1) \cdot A_2 + C_2 \]
   \[ = Sbox(a) \]

The calculations for a masked input are changed as follows:

1. \( (a + M) \cdot A_1 + C_1 \)
2. \( T \cdot ((a + M) \cdot A_1 + C_1) \)
3. \( T \cdot ((a + M) \cdot A_1 + C_1) \rightarrow I(T \cdot ((a + M) \cdot A_1 + C_1)) \rightarrow I(T \cdot ((a + M) \cdot A_1 + C_1) + M \cdot A_1 \cdot I(T)) \rightarrow I(a \cdot A_1 + C_1) \cdot I(T) + M \cdot A_1 \cdot I(T) \)
4. \( I(a \cdot A_1 + C_1) \cdot I(T) + M \cdot A_1 \cdot I(T) \cdot T \cdot A_2 + C_2 \)
   \[ = I(a \cdot A_1 + C_1) \cdot A_2 + C_2 + MA_1A_2 \]
   \[ = Sbox(a) + MA_1A_2 \]

where \( M \) is a 32-bit random mask. The result consists of the S-box of input added by a function of the random mask. This part can be removed by a separate mask removal generator part which generates the \( MA_1A_2 \).

Moreover, the corresponding nonlinear transformation \( \tau \) changes because the calculation of S-box for the masked data is modified. We define the new nonlinear transformation as \( \tau_M \).
C. Operation Flow of the Improved SMS4 Cipher

The hardware implementation of SMS4 cipher can be performed in two ways, one is that the encryption process and key expansion are synchronous; the other is that the encryption process and key expansion are asynchronous. The former means round keys are generated at the same time of enciphering process. While, the latter means when the encryption key is updated, the round keys can be generated in 32 clock cycles and be stored in some registers. Then the round keys can be read from the storage cells during the encryption or decryption process. In order to reuse the round keys module, we choose the latter, and the encryption process and key expansion are synchronous; the other is that the encryption process and key expansion are asynchronous. The former means round keys are generated at the same time of enciphering process. While, the latter means when the encryption key is updated, the round keys can be generated in 32 clock cycles and be stored in some registers. Then the round keys can be read from the storage cells during the encryption or decryption process. In order to reuse the round keys module, we choose the latter, and the encryption process and key expansion are synchronous.

In the structure as Fig. 2 showed, $M'$ which is equivalent to $MA_1A_2$ must be computed every round, it will greatly increase the calculation. In order to avoid this situation, module of $M'$ can be reused to improve computing speed. Let the masked data labeled by $dm1$ to $dm5$ and the so-called correction terms labeled by $C1$ to $C9$ in (18)−(20), this makes it easier to see how many additional operations are introduced by masking scheme. Because all terms labeled by $dm1$ to $dm5$ have to be calculated in the original S-box design. However, all terms labeled by $C1$ to $C9$ are the additional operations, which are introduced by the masking. In addition, we can reuse several of these correction terms, and this also reduces the area required for our implementation. Finally, we could use the methods such as module reuse and changing computing order which were suggested by Canright [8] to get a lower cost of implementation.

VI. EXPERIMENT RESULTS

In this study, we have some simulation experiments to verify that the improved SMS4 cipher presented in section V can be resistant to the DPA attack described in [5]. The experiments were performed using method mentioned in [5] and the results are described in this section.

A. Experiment

Using SMIC 0.18 μm CMOS technology, we completed the logic synthesis and physical design of both the ordinary SMS4 module and the improved SMS4 module, where the S-box of the ordinary SMS4 module is implemented as a look-up table. Then we got two circuit netlists.

Let the encryption key be the same as used in [5], then

$$MK = (0x01234567, 0x89ABCDEF, 0xFEDCBA98, 0x76543210)$$

Use 128-bit and 160-bit random numbers as the plaintext inputs respectively, two sets of 2000 transistor-level simulations were performed and the current traces and ciphertext outputs were captured. As the supply voltage is constant, we could get the same results by expressing the power tracks as the currents.

According to the DPA attack method described in section III, DPA attacks on the four bytes of the round key of the 32-nd round were performed. The results of two algorithms were performed as follows.

![Fig. 2. Improved SMS4 cipher encryption process.](image)

![Fig. 3. Maximums of the absolute values of the correlation coefficients along sample time versus all key hypotheses for ordinary SMS4 algorithm.](image)

![Fig. 4. Maximums of the absolute values of the correlation coefficients along sample time versus all key hypotheses for improved SMS4 algorithm.](image)
TABLE I
COMPARISON OF DIFFERENT DESIGNS

<table>
<thead>
<tr>
<th>Performances</th>
<th>Ordinary SMS4</th>
<th>Improved SMS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>SMIC 0.18 μm</td>
<td>SMIC 0.18 μm</td>
</tr>
<tr>
<td>Frequency (MHz)</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Throughput (Mbps)</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>Area (k-gates)</td>
<td>23</td>
<td>25</td>
</tr>
<tr>
<td>Throughput/Area (Mbps/k-gates)</td>
<td>8.7</td>
<td>8</td>
</tr>
<tr>
<td>Resistant to DPA</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Resistant to zero-value attack</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

B. Results

Fig. 3 and Fig. 4 show the maximums of the absolute values of the correlation coefficients along sample time for every hypothesis. As shown in Fig. 3, the correct assumption is \( r_{k_i,0,s} = 0 \times 91 \) (145) because the maximum correlation coefficient at this value is far greater than the one of other values. This means that attacks on ordinary SMS4 cipher is a success. However, the improved SMS4 cipher we suggested in this paper is not appeared above, there are no obvious fluctuations in Fig. 4 to get the correct assumption. So this design can be resistant to the DPA attack.

The other three bytes of \( r_{k_{31}} \) can be discussed the same as \( r_{k_{31,0}} \), then the DPA attack on 32-nd round is a complete.

C. Discuss

This study has presented a unique method for securing SMS4 cipher resistant to the DPA attack. The comparison of the ordinary SMS4 cipher and the improved SMS4 cipher is shown in Table I. For the same frequency and throughout, the design we suggested is resistant to DPA attack, although it is increase of hardware implementation area in a little extent. Moreover, it is also resistant to zero-value attack because all the data are additively masked.

VII. Conclusions

This paper analyzed the structure of SMS4 cipher and suggested an improved SMS4 algorithm with masking. This design can be resistant to the DPA attack especially zero-value attack while other designs can not. In addition, we performed module reuse and composite field computation to reduce circuit area and maintain its speed, so this design also adapt to hardware implementation. The design performed in this paper is suitable for the field that requires low hardware cost and high-level security.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China under grant No. 60606005.

REFERENCES