Fault-propagation Pattern Based DFA on SPN Structure Block Ciphers using Bitwise Permutation, with Application to PRESENT and PRINTcipher

XinJie Zhao[1], Tao Wang[1], ShiZe Guo[2,3]
[1](Department of Computer Engineering, Ordnance Engineering College, Shijiazhuang 050003, China)
[2](The Institute of North Electronic Equipment, Beijing 100083, China)
[3](School of Information Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China)

***Abstract*—This paper proposes a novel fault-propagation pattern based differential fault analysis method - FPP-DFA, and proves its feasibility on SPN structure block ciphers using bitwise permutation, such as PRESENT and PRINTcipher. Simulated experiments demonstrate that, with the fault model of injecting one nibble fault into the r-2^{th} round substitution layer, on average 8 and 16 faulty samples can reduce the master key search space of PRESENT-80/128 to $2^{14.7}$ and $2^{21.1}$ respectively, and 12 and 24 effective faulty samples can reduce the master key search space of PRINTcipher-48/96 to $2^{13.7}$ and $2^{22.8}$ respectively; with the fault model of injecting one nibble fault into the r-3^{th} round substitution layer, 8 samples can reduce the master key search space of PRINTCipher-96 to $2^{18.7}$.**

Fault-propagation pattern; fault-propagation path; differential fault analysis; bitwise permutation; SPN block cipher; PRESENT; PRINTcipher (key words)

I. INTRODUCTION

The new emerging pervasive computing demands have made low-end devices, such as smart cards, RFID tags, IC-printing applications more and more popular. Such tiny computing devices are used in many applications and environments, leading to an ever increasing need of security. This has spurred the development of lightweight cryptography. Many ultra-lightweight block ciphers have been developed such as mCrypton[1], HIGHT[2], SEA[3], DESXL[4], KATAN[5], MIBS[6].

PRESENT[7] and PRINTcipher[8] are two hardware-optimized ultra-lightweight block ciphers presented by Bogdanov, A et al in CHES 2007 and CHES 2010 respectively. Both of the ciphers apply the SPN structure and bitwise permutation, except that the substitution-permutation sequences are different. PRESENT adopts the substitution-permutation sequence, while PRINTcipher adopts the permutation-substitution sequence. In the proposals, the cipher designers analyzed the security of PRESENT and PRINTcipher with respect to the main known cryptanalytic methods, and showed that they were quite secure from the mathematics-based cryptanalysis. However, the cipher designers didn't consider the resilience of them against side-channel attacks. In this paper, we try to fill up this gap through cryptanalysis of the two ciphers against fault-based side channel attacks.

Fault attacks were first introduced by Boneh et al.[9] on RSA public key cryptosystem in 1997. Shortly after, Biham, E. et al[10] proposed an attack on secret key cryptosystems called Differential Fault Analysis (DFA), which combined the ideas of fault attack and differential attack. DFA attacks derived information about the secret key by examining the differences between correct and faulty ciphertexts. After that, various cryptosystems have been attacked by DFA technique, such as ECC[11], 3DES[12], AES[13], Camellia[14][15][16], MIBS[17], RC4[18], GRAIN-128[19],HC-128[20], Rabbit[21] etc. However, to the best of our knowledge, few work except [22] and [23] has been done on DFA against PRESENT, and there is no publication about DFA on PRINTcipher.

Previous Works. Li et al[22] proposed the first DFA on PRESENT-80. Suppose one nibble fault was injected between the r-2^{th} and the r-1^{th} round substitution input, they utilized the S-box output difference set of one bit S-box input difference as a distinguisher to filter the ideal faulty sample, and showed that, on average 40-50 effective faulty samples can recover 64-bit post-whitening key, and reduce the master key search space of PRESENT-80 to 2^{16}. Wang et al[23] proposed the first DFA on PRESENT-80 key schedule. Suppose one nibble fault was injected into the 80-bit intermediate updated key register while generating the 30^{th} and 31^{th} round keys, they used about 64 pairs of faulty samples to obtain 51-bit of the 64-bit post-whitening key, and reduced the master key search space of PRESENT-80 to 2^{29}.

Our Results. In this paper, we present a novel fault analysis method based on DFA, we name it as fault-propagation pattern based differential fault analysis—FPP-DFA. The main idea of FPP-DFA is using the output fault-propagation pattern to infer the fault location, predict the fault-propagation path, and then apply the traditional DFA technique for further key analysis. As to DFA on SPN block ciphers with bitwise permutation, the recovery of the fault-propagation path means the recovery of the S-box input difference, which is the most difficult part of DFA on SPN block ciphers. We show that FPP-DFA is quite efficient on SPN block ciphers with bitwise permutation, and apply it to PRESENT and PRINTcipher successfully.

In FPP-DFA on PRESENT, we adopt the fault model of injecting one nibble fault before the r-2^{th} round substitution layer. Through the analysis of the fault-propagation path and S-box differential pattern two distinguishers, the fault location and the r^{th} round substitution input difference can be deduced. Combing the traditional DFA technique, the post-whitening secret key and the r^{th} round key can be obtained.

As PRINTcipher applies a key-dependent permutation, a part of the secret key is embedded into the algorithm description, so algorithms with different secret keys will be subtly different from others. Thus, FPP-DFA on PRINTcipher is more difficult than PRESENT. In the attack, we adopt the fault model of injecting one nibble fault before

the r-2^{th} or r-3^{th} round substitution layer. Firstly, we compute the faulty index set of the ciphtertext difference, and combine the fault propagating path distinguisher of PRINTcipher to predict the exact fault location. According to the faulty difference before and after the r^{th} round key-dependent permutation, we propose a method to compute the permutation related key. After that, we can take the permutation off, and convert DFA on PRINTcipher to a problem with known cipher algorithm. Combining the recovered permutation and fault location, we can compute the input and output difference of the r-1^{th} round substitution, then apply the traditional DFA technique to deduce the round key.

We analyze the attack complexity of FPP-DFA on PRESENT and PRINTcipher, and verify it through concrete simulation experiments. Experimental results show that, with the fault model of injecting one nibble fault into the r-2^{th} round, on average 8 and 16 faults can reduce the master key search space of PRESENT-80/128 to $2^{14.7}$ and $2^{21.1}$ respectively, which is more efficient than previous DFA works [22][23] on PRESENT-80; on average 12 and 24 effective faults can reduce the master key search space of PRINTcipher-48/96 to $2^{13.7}$ and $2^{22.8}$ respectively. With the fault model of injecting one nibble fault into the r-3^{th} round, on average 8 effective faults can reduce the master key search space of PRINTCipher-96 to $2^{18.7}$.

Organization of the Paper. This paper is organized as follows. The main idea of FPP-DFA is proposed in Section II. The detailed FPP-DFA attacks on PRESENT and PRINTcipher are described in Section III and Section IV respectively, and the conclusions are presented in section V.

II. MAIN IDEA OF FPP-DFA

Next, we present the main idea of the FPP-DFA. It is mainly composed of the following two phases:

1) Fault location analysis

In this phase, the attacker learns the fault-propagation pattern, and then builds the distinguisher for the enumerable faulty locations based on the fault model. As to FPP-DFA on SPN block ciphers with substitution-permutation sequence, such as PRESENT[1], the possible fault locations can be deduced by patterns of the r^{th} round S-box output(or faulty ciphertext) differences. As to FPP-DFA on SPN block cipher with permutation-substitution sequence, such as PRINTCipher[8], the exact injected fault location can be deduced by the faulty ciphertext nibble index set.

2) DFA based Key extraction

As to SPN block ciphers with bitwise permutation, after the fault location is deduced, the predicted fault location can be used to deduce the fault-propagation path, and compute the input difference of the S-box, which is the crucial part of DFA on SPN block ciphers (at most cases, only the S-box output difference can be computed directly from the ciphertext difference, and the input difference is unknown). Finally, combing the traditional DFA method, the secret key can be recovered.

It should be noted that the FPP-DFA technique above is a general framework of DFA on SPN block ciphers using bitwise permutation. In FPP-DFA on specific ciphers, the two phases above can be divided into several steps as needed.

As shown in Section III and Section IV, we take PRESENT and PRINTCipher as two examples, and apply the FPP-DFA technique to prove its feasibility and efficiency.

III. PROPOSED FPP-DFA ATTACK ON PRESENT

A. Description of PRESENT Algorithm

PRESENT is a 31-round SPN type block cipher with block size of 64 bits. It supports 80 and 128-bit secret key. Firstly, the plaintext Xored the subkey K^1 as the input of the 1^{st} round. After 31 rounds iterations, the 31^{th} round output Xored with the subkey K^{32} is the ciphertext. Each encryption round is composed of the following 3 steps.

1) addRoundKey—*AK*. At the beginning of each round, 64 bits output of the last round is Xored with the subkey.

2) sBoxlayer—*SL*. 16 identical 4-bit to 4-bit S-boxes are used in parallel.

3) pLayer—*PL*. The i^{th} bit is moved to bit position $P(i)$ by a constant permutation table P.

PRESENT can take keys of either 80 or 128 bits. Below is the key schedule of 80-bit version. The 80-bit key is stored in a register $K=k_{79}||k_{78}||\ldots||k_0$. At round r ($1\le r\le 31$), the 64-bit round key K^r is equal to the 64 leftmost bits of K. After K^r is extracted, K is rotated by 61 bit positions to the left, then a S-box is applied to the left-most 4 bits of K and finally the round-counter r is Xored with bits $k_{19}||k_{18}||k_{17}||k_{16}||k_{15}$ of K.

B. Notations

Here we introduce some notations in order to make our discussion conveniently. Bits are numbered from zero with bit zero on the right of a block.

We denote plaintext and ciphertext as X, Y, the output of the i^{th} round *AK*, *SL*, *PL* function as A^i,B^i,C^i ($1\le i\le 31$). The faulty variables above can be denoted as X',Y',A^{i}',B^{i}',C^{i}', and the j^{th} bit of the variables above can be denoted as X_j,Y_j,A_j^i,B_j^i,C_j^i ($0\le j\le 63$). The differences between correct and faulty variables above can be denoted as ΔX, ΔY, ΔA^i, ΔB^i, ΔC^i, and the j^{th} nibble of the variables above can be denoted as NX_j, NY_j, NA_j^i, NB_j^i, NC_j^i ($0\le j\le 15$). The nibble difference of the variables above can be denoted as ΔNX_j, ΔNY_j, ΔNA_j^i, ΔNB_j^i, NC_j^i ($0\le j\le 15$), and the related nibble length is 4. The faulty nibble and bit index set of the variables above can be denoted as SX, SY, SA^i, SB^i, SC^i and sx, sy, sa^i, sb^i, sc^i.

C. Fault Model

The fault model of paper is shown as follows.

1) One random nibble fault is induced into the input of the r-2^{th} (29^{th}) round substitution. The attacker knows neither the location nor the concrete value of the fault.

2) For any plaintext adaptively selected, two different ciphertexts under the control of the same secret key are available, the right and the faulty one. How to induce the specific fault is not covered in this paper, since this is not the main concern of our paper and many literatures on fault inductions are available in [24].

3) Only one master key is used in one attack.

Fig.1 shows the fault-propagation path of the above fault model (the faulty nibble index is 15) with the maximal fault propagating width, and the output difference of every faulty S-box is $(1111)_2$.

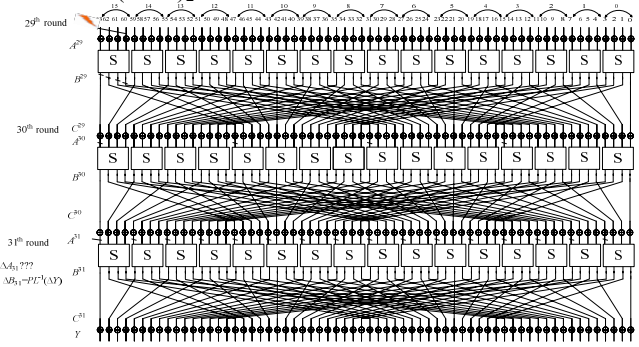


Figure 1. One nibble fault-propagation path of PRESENT

D. FPP-DFA procedure

FPP-DFA on PRESENT is composed of the following steps.

1) Fault location analysis

***a*) Learn the fault-propagation path**

Based on the fault model above, the fault-propagation path of the 16 possible fault locations (i) can be built.

TABLE I. FAULT-PROPAGATION PATH OF PRESENT

i	$sa^{30}(\Delta NA_i^{30})$	SA^{30}	$sa^{31}(\Delta NA_i^{31})$
0	0,16,32,48(0001)$_2$	0,4,8,12	0,16,32,48,4,20,36,52,8,24,40,56,12,28,44,60(0001)$_2$
1	1,17,33,49(0010)$_2$	0,4,8,12	0,16,32,48,4,20,36,52,8,24,40,56,12,28,44,60(0001)$_2$
2	2,18,34,50(0100)$_2$	0,4,8,12	0,16,32,48,4,20,36,52,8,24,40,56,12,28,44,60(0001)$_2$
3	3,19,35,51(1000)$_2$	0,4,8,12	0,16,32,48,4,20,36,52,8,24,40,56,12,28,44,60(0001)$_2$
4	4,20,36,52(0001)$_2$	1,5,9,13	1,17,33,49,5,21,37,53,9,25,41,57,13,29,45,61(0010)$_2$
5	5,21,37,53(0010)$_2$	1,5,9,13	1,17,33,49,5,21,37,53,9,25,41,57,13,29,45,61(0010)$_2$
6	6,22,38,54(0100)$_2$	1,5,9,13	1,17,33,49,5,21,37,53,9,25,41,57,13,29,45,61(0010)$_2$
7	7,23,39,55(1000)$_2$	1,5,9,13	1,17,33,49,5,21,37,53,9,25,41,57,13,29,45,61(0010)$_2$
8	8,24,40,56(0001)$_2$	2,6,10,14	2,18,34,50,6,22,38,54,10,26,42,58,14,30,46,62(0100)$_2$
9	9,25,41,57(0010)$_2$	2,6,10,14	2,18,34,50,6,22,38,54,10,26,42,58,14,30,46,62(0100)$_2$
10	10,26,42,58(0100)$_2$	2,6,10,14	2,18,34,50,6,22,38,54,10,26,42,58,14,30,46,62(0100)$_2$
11	11,27,43,59(1000)$_2$	2,6,10,14	2,18,34,50,6,22,38,54,10,26,42,58,14,30,46,62(0100)$_2$
12	12,28,44,60(0001)$_2$	3,7,11,15	3,19,35,51,7,23,39,55,11,27,43,59,15,31,47,63(1000)$_2$
13	13,29,45,61(0010)$_2$	3,7,11,15	3,19,35,51,7,23,39,55,11,27,43,59,15,31,47,63(1000)$_2$
14	14,30,46,62(0100)$_2$	3,7,11,15	3,19,35,51,7,23,39,55,11,27,43,59,15,31,47,63(1000)$_2$
15	15,31,47,63(1000)$_2$	3,7,11,15	3,19,35,51,7,23,39,55,11,27,43,59,15,31,47,63(1000)$_2$

Table I is the PRESENT faulty nibble and bit index set with maximal fault-propagation width after the 30th, 31th round substitution. It's obvious to see that at most 4 and 16 nibbles after the 30th and 31th round substitution become faulty, and all faulty nibbles of ΔNA^{31}=i/4+1, so there are only 4 possible ΔNA^{31} value for 16 fault locations.

***b*) Learn the patterns of PRESENT differential S-box**

Suppose a denotes the S-box index, $f1$ and $f2$ denotes the S-box input and output difference. a, $f1$, $f2$ are all 4-bit variable satisfying

$$S[A] \oplus S[A \oplus F1]=F2 \quad (1)$$

Table II displays all the possible $f2$ value when only one bit of $f1$ is 1 in PRESENT S-box. It's clear to see that at least 2 bits of $f2$ is 1, which means that after 2 round fault propagation, at least 4 and at most 16 nibbles of the ciphertext become faulty, and $f2$ related column value set can become a distinguisher for $f1$.

TABLE II. PATTERNS OF PRESENT DIFFERENTIAL S-BOX

$f2(f1=(0001)_2)$	$f2(f1=(0010)_2)$	$F2(f1=(0100)_2)$	$F2(f1=(1000)_2)$
$(0011)_2$,$(0111)_2$, $(1001)_2$, $(1101)_2$	$(0011)_2$,$(0101)_2$, $(0110)_2$, $(1010)_2$, $(1100)_2$,$(1101)_2$, $(1110)_2$	$(0101)_2$,$(0110)_2$, $(0111)_2$, $(1001)_2$, $(1010)_2$,$(1100)_2$, $(1110)_2$	$(0011)_2$,$(0111)_2$, $(1001)_2$, $(1011)_2$, $(1101)_2$, $(1111)_2$

***c*) Deduce the fault location**

Firstly, the 31th round substitution output difference can be computed by $\Delta B^{31}=PL^{-1}(\Delta Y^{31})$. For each nonzero ΔNB_i^{31}, according to the 4 distinguishers of Table II, the possible $f1$ candidate set S_i can be deduced, and the intersection set of S_i is the final possible $f1$ candidate set. Then, according to the location distinguishers in Table I, the fault location set can be obtained. The more faulty nibbles in ΔB^{31}, the less candidates of ΔA^{31} can be deduced.

2) DFA based Key extraction

***a*) Deduce the round Key K^{32}**

From above, limited candidates of ΔA^{31} can be computed. As ΔB^{31} can be computed by $\Delta B^{31}=PL^{-1}[Y] \oplus PL^{-1}[Y']$, then for each nonzero nibble of ΔNA_i^{31} and ΔNB_i^{31}, NA_i^{31} satisfying

$$SL[NA_i^{31}] \oplus SL[NA_i^{31} \oplus \Delta NA_i^{31}]= \Delta NB_i^{31} \quad (2)$$

would be the correct candidate.

$$K^{32}_{P^{-1}(4*i+3)} \| K^{32}_{P^{-1}(4*i+2)} \| K^{32}_{P^{-1}(4*i+1)} \| K^{32}_{P^{-1}(4*i)} = SL[NA_i^{31}] \oplus Y^{32}_{P^{-1}(4*i+3)} \| Y^{32}_{P^{-1}(4*i+2)} \| Y^{32}_{P^{-1}(4*i+1)} \| Y^{32}_{P^{-1}(4*i)} \quad (3)$$

Then applying equation (3), 4-bit of K^{32} can be recovered. As about 4-16 nibbles of B^{31} is faulty, applying the technique above, 4-16 nibbles of K^{32} can be obtained, and full 64-bit of K^{32} can be recovered through more sample analysis.

***b*) Deduce the round Key K^{31}**

After K^{32} was deduced, ΔB^{30} can be computed by

$$\Delta B^{30}=SL^{-1}[PL^{-1}(Y)] \oplus SL^{-1}[PL^{-1}(Y')] \quad (4)$$

Using the distinguishers in Table II, ΔA^{30} can be deduced. And then applying the same DFA technique above, 2-4 nibbles of K^{31} can be obtained. K^{31} can be recovered through

more sample analysis. Noted that, as to the wrong K^{32} candidate, after the DFA analysis with about 4 faulty samples, some bits of K^{31} can always get an empty set, which can be used to eliminate the wrong K^{32} candidates.

c*) Deduce the master Key *K

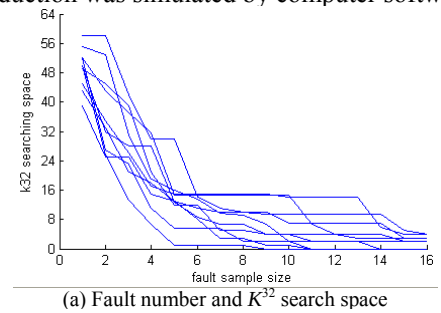
Combining K^{31}, K^{32}, PRESENT key schedule, the master key K can be recovered.

E. Complexity Analysis

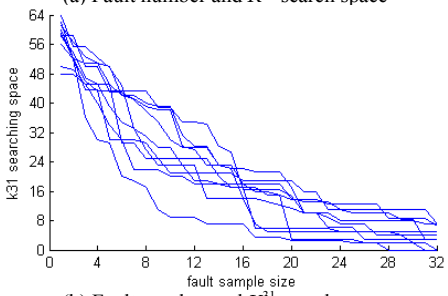
According to the patterns of the PRESENT differential S-box in Table II, at least 2-4 nibbles of B^{30} become faulty, which can be used to deduce 2-4 nibbles of K^{31}, and at least 4-16 nibbles of B^{31} become faulty, which can be used to deduce 4-16 nibbles of K^{32}. As approximately two times analysis of the faulty nibble with the same index can recover one nibble key, about 8 faults can obtain K^{32} and limited candidates of K^{31}, which are enough to recover the PRESENT-80 master key. 16 faults can obtain K^{32} and K^{31}, which are enough to obtain the PRESENT-128 master key.

F. Experimental Results

We have implemented our attack on a PC using Visual C++ 6.0 Compiler on a 1.81 GHz Athlon with 1GB memory, the fault induction was simulated by computer software.



(a) Fault number and K^{32} search space



(b) Fault number and K^{31} search space

Figure 2. Experimental results of 10 times PRESENT attack

In PRESENT-80 on, on average 8 samples can reduce the K^{32} search space to $2^{7.6}$ (Fig.2(a)). Combining the step 5, wrong candidates of K^{32} can be eliminated, and possible candidates of K^{31} can be predicted. Then, the master key search space of PRESENT-80 can be further reduced to $2^{14.7}$. Comparing with the previous PRESENT-80 attacks [22][23], our attacks are much more efficient, and the result is also well agreed with the previous theoretical analysis.

Compared with the DFA on the encrypt procedure of PRESENT-80 by Li et al.[22], they supposed one nibble fault was injected between the r-2th and r-1th round substitution, and judged the effective faulty samples by whose 2-7 nibbles of B^{31} are faulty. Utilizing the S-box output differential distinguisher when only one bit of the S-box input nibble difference is 1, the nonzero nibble of ΔA^{31} was deduced one by one, and K^{32} can be deuced by A^{31} and ΔB^{31}. About 40-50 samples are required to obtain K^{32}. But this may discard those faulty samples which 8-16 nibbles are faulty, but still satisfied the one bit distinguisher and fault model above.

While the attack proposed in this paper can make full use of the faulty samples. We started by learning the fault-propagation path of the model, and utilized the one bit distinguisher of Table II to deduce the possible 4 candidates of 64-bit ΔA^{31}, finally recovered K^{32} by ΔA^{31} and ΔB^{31}. Moreover, after some candidates of K^{32} have been deduced, we use it to compute ΔB^{30}, and apply the one bit distinguisher of Table II again to predict ΔA^{30}. Then the key search space of K^{31} can be reduced, and some wrong K^{32} candidates can also be eliminated. Finally, combing the recovered K^{32} and K^{31} candidates, the search space of PRESENT-80 master key can be further reduced.

Our attack technique can be adjusted to DFA on PRESENT-128 very easily without changing the fault model. In FPP-DFA attack on PRESENT-128, we shown that, on average 16 samples can reduce the K^{32} search space to $2^{1.8}$, the K^{31} search space to $2^{16.3}$(Fig.2(b)) and the PRESENT-128 master key search space to $2^{21.1}$.

IV. THE PROPOSED FPP-DFA METHOD ON PRINTCIPHER

A. Description of PRINTcipher Algorithm

PRINTcipher is a key dependant SPN structure block cipher with b-bit blocks and b rounds, $b \in \{48, 96\}$, and an effective key length of $5b/3$ bits. The $5b/3$ bits user-supplied key sk is consisted of two subkey components $sk = sk1||sk2$ where $sk1$ is a fixed b bits secret key, and $sk2$ is a $2b/3$ bits sub-key to generate the key-dependent permutations and derive an additional of security via the secret algorithm variability. PRINTcipher has two variants decides by b: PRINTcipher-48 and PRINTcipher-96. Each encryption round consists of the following 5 steps.

1) Key xor—*KX*. The b-bit current state of the cipher is Xored with a b-bit subkey $sk1$, $sk1$ is identical in all rounds.

2) Linear diffusion—*LD*. Bit i of the current state is moved to bit position $P(i)$

$$P(i) = \begin{cases} 3 \times i \bmod b - 1 & \text{for } 0 \le i \le b-2 \\ b-1 & \text{for } i = b-1 \end{cases} \quad (5)$$

3) Round counter RC_i xor—*RX*. The least significant n-bit ($n = \log_2 r$) of the cipher state is Xored with a round constant RC_i.

4) Keyed permutation—*KP*. The b-bit cipher state is divided into $b/3$ 3-bit nibbles. $sk2$ are divided into $b/3$ sets of two bits, and each two bits quantity $a_1||a_0$ is used to pick one of four available permutations of the three input bits state nibble. Specifically, the three input bits $c_2||c_1||c_0$ are permuted by $a_1||a_0$.

$$input: c_2 \| c_1 \| c_0 \Rightarrow \begin{cases} case\ a_1 \| a_0 = 00 & output: c_2 \| c_1 \| c_0 \\ case\ a_1 \| a_0 = 01 & output: c_1 \| c_2 \| c_0 \\ case\ a_1 \| a_0 = 10 & output: c_2 \| c_0 \| c_1 \\ case\ a_1 \| a_0 = 11 & output: c_0 \| c_1 \| c_2 \end{cases} \quad (6)$$

5) sBoxLayer—*SL*. *b*/3 identical 3-bit to 3-bit S-boxes are used in parallel.

Noted that PRINTCipher has no key schedule, the master key *sk* is used in every round.

B. Notations

We denote the plaintext and ciphertext as X, Y, the output of the i^{th} round *KX*, *LD*, *KP*, *SL* function as A^i,B^i,C^i,$D^i(0\le i\le b\text{-}1)$. The faulty variants above can be denoted as X',Y',$A^{i'}$,$B^{i'}$,$C^{i'}$,$D^{i'}$, the j^{th} bit of the variants above can be denoted as X_j,Y_j,A_j^i,B_j^i,C_j^i,$D_j^i(0\le i,j\le b\text{-}1)$. The difference between correct and faulty variants above can be denoted as ΔX_j, ΔY_j, ΔA_j^i, ΔB_j^i, ΔC_j^i, ΔD_j^i, the j^{th} nibble of the variants above can be denoted as NX_j, NY_j, NA_j^i, NB_j^i, NC_j^i, $ND_j^i(0\le j\le b\text{-}1/a)$, and the related nibble length a is 3. The j^{th} nibble difference of the variants above can be denoted as ΔNX_j, ΔNY_j, ΔNA_j^i, ΔNB_j^i, NC_j^i, ND_j^i, and the nonzero faulty nibble and bit index set of the variant above can be denoted as *SX*, *SY*, SA^i, SB^i, SC^i, SD^i.and *sx*, *sy*, sa^i, sb^i, sc^i, sd^i.

C. Fault Model

The fault model of FPP-DFA on PRINTCipher is the same with PRESENT, except that the injected faulty round m has subtle difference. As to PRINTCipher-48, m= 46, and as to PRINTCipher-96, m= 94 or 93. Next we take FPP-DFA on PRINTCipher-48 as an example.

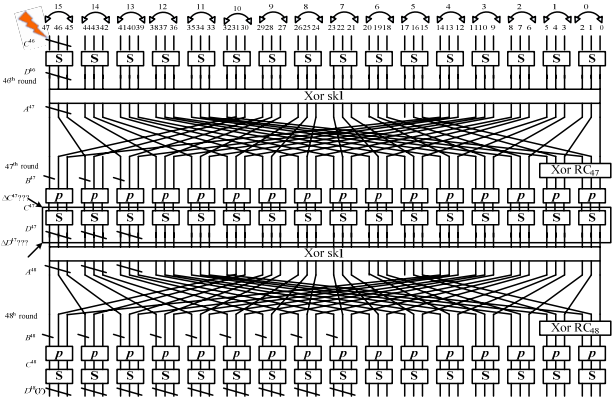


Figure 3. One nibble fault-propagation path of PRINTCipher-48

Fig.3 is the fault propagating procedure of one nibble fault injecting into the 15th nibble before the 46th round substitution with the maximized width, in which all of the S-box output difference is $(111)_2$.

D. FPP-DFA procedure

FPP-DFA on PRINTCipher is shown as follows.

1) Fault location analysis

***a*) Learn the fault-propagation path**

Based on the fault model above,

Table III is the faulty nibble index set of PRINTCipher-48 with maximal fault-propagation width after the 47th, 48th round substitution, and at most 3 and 9 nibbles after the 47th and 48th round substitution become faulty.

***b*) Deduce the fault location**

Take injecting fault into the 15th nibble before the 46th round substitution as an example. According to the ciphertext difference, a distinct faulty nibble index set SD^{48} can be computed. Suppose SD^{48}={7,…,15}, according to Table III, we can deduce that the faulty nibble index is 15. And we store it as an effective fault sample. Interestingly, if the exact injected fault location i of the 46th round can be deduced, either all 3 nibbles of ΔD^{47} with consecutive indices (SD^{47}={13,14,15}) become faulty, or 2 nibbles of ΔD^{47} with nonconsecutive indices (SD^{47}={13,15}) become faulty, which also means that ΔND_i^{46} can be either $(101)_2$ or $(111)_2$. After the fault location is deduced, both ΔB^{47} and ΔB^{48} can be obtained.

TABLE III. FAULT-PROPAGATION PATH OF PRINTCIPHER-48

i	SD^{47}	SD^{48}	*i*	SD^{47}	SD^{48}
0	0,1,2	0,1,2;3,4,5;6,7,8	8	8,9,10	8,9,10;11,12,13;14,15,0
1	3,4,5	9,10,11;12,13,14;15,0,1	9	11,12,13	1,2,3;4,5,6;7,8,9
2	6,7,8	2,3,4;5,6,7;8,9,10	10	14,15,0	10,11,12;13,14,15;0,1,2
3	9,10,11	11,12,13;14,15,0;1,2,3	11	1,2,3	3,4,5;6,7,8;9,10,11
4	12,13,14	4,5,6;7,8,9;10,11,12	12	4,5,6	12,13,14;15,0,1;2,3,4
5	15,0,1	13,14,15;0,1,2;3,4,5	13	7,8,9	5,6,7;8,9,10;11,12,13
6	2,3,4	6,7,8;9,10,11;12,13,14	14	10,11,12	14,15,0;1,2,3;4,5,6
7	5,6,7	15,0,1;2,3,4;5,6,7	15	13,14,15	7,8,9;10,11,12;13,14,15

2) DFA based Key extraction

***a*) Deduce the permutation related key *sk*2**

After the exact fault location is deduced, the permutation related key *sk*2 can be obtained by following algorithm.

$\Delta C^{48} \leftarrow SL^{-1}[D^{48}] \oplus SL^{-1}[D^{48'}]$
ΔB^{48} can be deduced by ΔD^{48}
For each index i in SD^{48}
{
  j←faulty index bit of ΔNB_i^{48}
  If(j == 2)
  {
  $\Delta NB_i^{48} \leftarrow (100)_2$
  If ($\Delta NB_i^{48} == (100)_2$)
    $sk2_{2*i}=(0)_2$
  else if ($\Delta NB_i^{48} == (010)_2$)
    $sk2_{2*i+1} \| sk2_{2*i}=(01)_2$
  else if ($\Delta NB_i^{48} == (000)_2$)
    $sk2_{2*i+1} \| sk2_{2*i}=(11)_2$
  }
  If(j == 1)
  {
  $\Delta NB_i^{48} \leftarrow (010)_2$
  If ($\Delta NB_i^{48} == (010)_2$)
    $sk2_{2*i+1} \| sk2_{2*i}=\{(00)_2,(11)_2\}$
  else if ($\Delta NB_i^{48} == (100)_2$)
    $sk2_{2*i+1} \| sk2_{2*i}=\{01\}_2$

```
        else if (ΔNB_i^48 == (001)_2)
            sk2_{2*i+1}||sk2_{2*i}={10}_2
    }
    If(j==0)
      {
        ΔNB_i^48←(001)_2
        If (ΔNB_i^48 == (001)_2)
            sk2_{2*i+1}={0}_2
        else if (ΔNB_i^48 == (100)_2)
            sk2_{2*i+1}||sk2_{2*i}={11}_2
        else if (ΔNB_i^48 == (010)_2)
            sk2_{2*i+1}||sk2_{2*i}={10}_2
    }
}
```

Applying the algorithm above, after one sample analysis, at most 18 bits of *sk*2 can be recovered, and after more sample analysis, full 32 bits of *sk*2 can be obtained.

***b*) Deduce the secret key *sk*1**

Once *sk*2 was recovered, ΔD^{47} can be computed by

$$\Delta D^{47} = LD^{-1}(KP^{-1}(SL^{-1}[D^{48}] \oplus SL^{-1}[D^{48'}])) \quad (7)$$

Take recovering $sk1_{47}||sk1_{46}||sk1_{45}$ as an example, according to Fig.3, ΔNB^{47} can be computed, and $\Delta NB_{15}^{47}=(100)_2$, then ΔNC_{15}^{47} can be deduced by $KP(\Delta NB_{15}^{47})$. Candidates of NC_{15}^{47} satisfying

$$SL[NC_{15}^{47}] \oplus SL[NC_{15}^{47} \oplus \Delta NC_{15}^{47}] = \Delta ND_{15}^{47} \quad (8)$$

would be the possible one. Then

$$sk1_{47}||sk1_{46}||sk1_{45}=SL[NC_{15}^{47}] \oplus LD^{-1}(RX^{-1}(KP^{-1}(SL^{-1}[D^{48}]))) \quad (9)$$

Applying the same method above, after one sample analysis, at most 9 bits of *sk*1 can be recovered, and after more sample analysis, full 48 bits of *sk*1can be obtained.

c*) Deduce the master key *sk

The master key *sk* equals to *sk*1||*sk*2.

E. Complexity Analysis

1) Probability of effective fault location deducing

Suppose the S-box index is *a*, the input and output difference is *f*1and *f*2, *a*, *f*1, *f*2 are all 3-bit variable and satisfy equation 1.

Table IV displays all the possible candidates of *f*2 when only one bit of *f*1 is 1. According to sub-section *D* and Table III, if the fault location *i* was deduced, ΔND_i^{46} can be either $(101)_2$ or $(111)_2$, the probability is 2/7. The 1st bit of the output difference ΔND_j^{47} propagated by the 1st bit of ΔND_i^{46} should be 1, and its probability is 16/24. The 3rd output difference ΔND_j^{47} propagated by the 3rd bit of ΔND_i^{46} should be 1, and its probability is 16/24. So the overall fault location deducing probability is 12.7% (2/7*((16/24)*(16/24))).

TABLE IV. PATTERNS OF PRINTCIPHER DIFFERENTIAL S-BOX

a	***f*2(*f*1=(001)$_2$)**	***f*2(*f*1=(010)$_2$)**	***f*2(*f*1=(100)$_2$)**
$(000)_2$	$(001)_2$	$(011)_2$	$(111)_2$
$(001)_2$	$(001)_2$	$(111)_2$	$(101)_2$
$(010)_2$	$(101)_2$	$(011)_2$	$(110)_2$
$(011)_2$	$(101)_2$	$(111)_2$	$(100)_2$
$(100)_2$	$(011)_2$	$(010)_2$	$(111)_2$
$(101)_2$	$(011)_2$	$(110)_2$	$(101)_2$
$(110)_2$	$(111)_2$	$(010)_2$	$(110)_2$
$(111)_2$	$(111)_2$	$(110)_2$	$(100)_2$

2) Complexity analysis of *sk*2 deducing

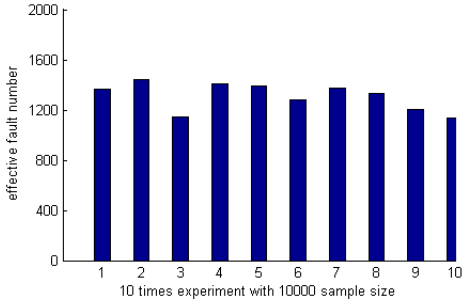
Once the fault location is exactly located, at least 2 nibbles, at most 9 nibbles, on average 5.5 nibbles of B^{48} have become faulty. According sub-section *D*, if one nibble faulty difference after the *KP* layer changed, 2-bit of *sk*2 can be deduced directly. Else, either one bit of *sk*2 can be deduced directly, or 2 bits of *sk*2 are equal. So on average, 5/3 bits of *sk*2 can be deduced for each faulty B^{48} nibble. Then about 16/(5.5*(5/3))=3.5 faulty samples are needed to recover *sk*2.

3) Complexity analysis of *sk*1 deducing

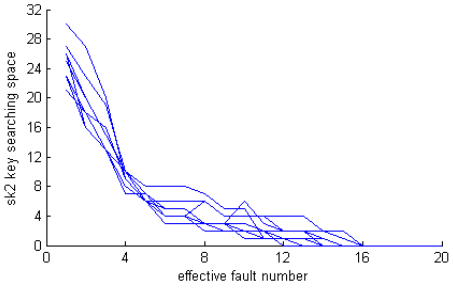
According sub-section *D*, once the fault location is exactly located, on average 2.5 nibbles of ΔNB^{47} become faulty and be used to deduce *sk*1. As shown in Table IV, if only one bit of ΔNB^{47} is 1, on average 2 candidates of *a* can be deduced, the key search space of one *sk*1 nibble can be reduced from 8 to 2, and two times analysis of the nibble with the same index can recover the related *sk*1 nibble. So 1.25 nibbles of *sk*1 can be deduced and about 12.8 (16/1.25) samples are required to obtain *sk*1.

F. Experimental Results

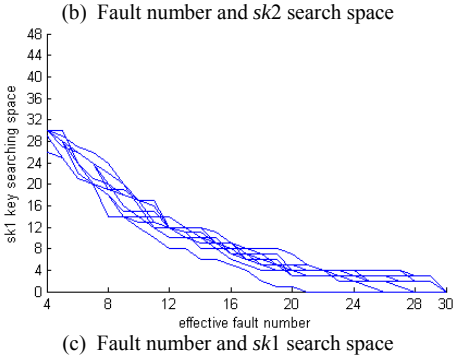
Fig.4-(a) is the effective fault number of 10 times FPP-DFA experiment on PRINTCipher-48 with 10000 samples. The average fault location deducing probability is 13.1%, which is close to the theoretical value 12.7%.



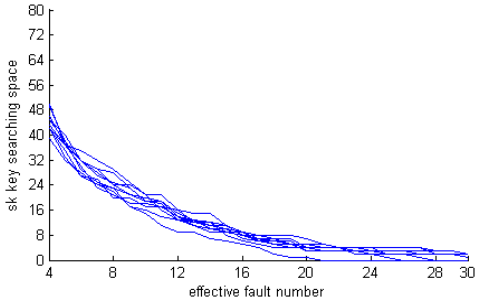
(a) Effective fault number statistics



(b) Fault number and *sk*2 search space



(c) Fault number and *sk*1 search space



(d) Fault number and *sk* search space

Figure 4. Experimental results of PRINTCipher-48 attack

Fig 4-(b),(c),(d) show the relationships between the fault number and the key search space of *sk*2, *sk*1, *sk*. It's clear to see that, 4 effective faults can reduce the *sk*2 key search space from 2^{32} to $2^{9.1}$, 12 effective faults can reduce the *sk*1 search space from 2^{48} to $2^{2.1}$, 12 effective faults can reduce the master key search space from 2^{80} to $2^{13.7}$, the results are well agreed with previous theoretical analysis.

Under the *r*-2^{th} round fault model, we have applied the FPP-DFA on PRINTCipher-96. The fault-propagation path is shown in appendix Table V, on average 24 effective faults can reduce the 160-bit master key search space to $2^{22.8}$.

Meanwhile, based on the fault model of injecting one nibble into the *r*-3^{th} round substitution, we have also applied the FPP-DFA on PRINTCipher-96. The fault-propagation path is shown in appendix Table VI, on average 8 effective faults can reduce the 160-bit master key search space to $2^{18.7}$.

V. CONCLUSION

We have proposed a novel fault-propagation pattern based DFA techniques on SPN structure block ciphers with bitwise permutation, applied it to PRESENT and PRINTcipher, and proved its efficiency through concrete simulated experiments. We show that, implementing SPN structure block ciphers with bitwise permutation should take certain countermeasures of fault-based attacks; at least last 3 rounds of PRESENT-80/128, PRINTCipher-48, and at least last 4 rounds of PRINTCipher-96 should be protected.

REFERENCES

[1] Lim, C., Korkishko, T.: mCrypton - A Lightweight Block Cipher for Security of Low-cost RFID Tags and Sensors. In: Song, J., Kwon, T., Yung, M. (eds.) WISA 2005. LNCS, vol. 3786, pp. 243–258, Springer, Heidelberg(2006).

[2] Hong, D., Sung, J., Hong, S., Lim, J., Lee, S., Koo, B.-S., Lee, C., Chang, D., Lee, J., Jeong, K., Kim, H., Kim, J., Chee, S.: HIGHT: A New Block Cipher Suitable for Low-Resource Device. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 46–59. Springer, Heidelberg (2006)

[3] Standaert, F.-X., Piret, G., Gershenfeld, N., Quisquater, J.-J.: SEA: A Scalable Encryption Algorithm for Small Embedded Applications. In: Domingo-Ferrer, J.,Posegga, J., Schreckling, D. (eds.) CARDIS 2006. LNCS, vol. 3928, pp. 222–236. Springer, Heidelberg (2006)

[4] Leander, G., Paar, C., Poschmann, A., Schramm, K.: New Lightweight DES Variants. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 196–210. Springer, Heidelberg (2007)

[5] de Canniѐe, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN−A Family of Small and Efficient Hardware-Oriented Block Ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272−288. Springer, Heidelberg (2009)

[6] Maryam Izadi, Babak Sadeghiyan, Seyed Saeed Sadeghian, et al.: MIBS: A New Lightweight Block Cipher[A]. J.A. Garay, A. Miyaji, and A. Otsuka (Eds.): CANS 2009[C], LNCS, vol.5888, Springer, pp. 334–348.(2009)

[7] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., et al.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)

[8] Knudsen, L., Leander, G., Poschmann, A., and Robshaw, M.J.B.: PRINTcipher: A Block Cipher for IC-Printing. In: S. Mangard and F.-X. Standaert (Eds.): CHES 2010, LNCS, vol. 6225, pp. 16–32, Springer, Heidelberg (2010)

[9] Boneh, D., DeMillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg. (1997)

[10] Biham, E., Shamir, A.: Differential Fault Analysis of Secret Key Cryptosystem. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)

[11] Biehl, I., Meyer, B., Muller, V. Differential fault analysis on elliptic curve cryptosystems. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 131−146. Springer, Heidelberg. (2000)

[12] Hemme, L.: A differential fault attack against early rounds of (Triple-) DES. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 254–267. Springer, Heidelberg. (2004)

[13] Piret, G., Quisquater, J.J.: A Differential Fault Attack Technique against SPN Structures, with Application to the AES and Khazad. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 77–88. Springer, Heidelberg. (2003)

[14] Zhou, Y.B., Wu, W.L., Xu, N.N, e tal.: Differential Fault Attack on Camellia. Chinese Journal of Electronics, Vol.18, No.1, pp. 13–19. (2009)

[15] Zhao, X.J., Wang, T.: An Improved Differential Fault Attack on Camellia. Cryptology ePrint Archive, Report 2009/585 (2009)

[16] Zhao, X.J., Wang, T.: Further Improved Differential Fault Analysis on Camellia by Exploring Fault Width and Depth. Cryptology ePrint Archive, Report 2010/026 (2010)

[17] Zhao, X.J., Wang, T., Wang, S.Z., et al: Research on deep differential fault analysis against MIBS. Journal on Communications, Vol. 31, No.12, pp: 89-98. (2010)

[18] Hoch, J.J., Shamir, A.: Fault analysis of stream ciphers. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 240–253. Springer, Heidelberg. (2004)

[19] Alexandre Berzati, C´ecile Canovas, Guilhem Castagnos, et al.: Fault Analysis of GRAIN-128. HOST 2009, pp.7-14.(2009)

[20] Aleksandar Kircanski and Amr M. Youssef.: Differential Fault Analysis of HC-128. In D.J. Bernstein and T. Lange (Eds.): AFRICACRYPT 2010[C], LNCS, vol. 6055, pp. 261–278. Springer, Heidelberg. (2010)

[21] Alexandre Berzati, C´ecile Canovas-Dumas, and Louis Goubin.: Fault Analysis of Rabbit: Toward a Secret Key Leakage[A]. B. Roy and N. Sendrier (Eds.): INDOCRYPT 2009[C], LNCS, vol. 5922, pp. 72–87. Springer, Heidelberg. (2009)

[22] Li, J.R., Gu, D.W.: Differential Fault Analysis on PRESENT. CHINACRYPT 2009(in chinese), pp.3-13. (2009)

[23] Wang, G.L., Wang, S.S.: Differential Fault Analysis on PRESENT Key Schedule. In Proc of International Conference on Computational Intelligence and Security (CIS 2010), pp.362-366. IEEE Computer society. (2010)

[24] Giraud, C., Thiebeauld, H.: A survey on fault attacks. In Proc of 6th International Conference on Smart Card Research and Advanced Applications (CARDIS'O4), France, pp. 22–27. (2004)

TABLE V. FAULT-PROPAGATION PATH OF PRINTCIPHER-96 (94TH ROUND NIBBLE FAULT MODEL)

i	SD^{95}	SD^{96}	*i*	SD^{95}	SD^{96}
0	0,1,2	0,1,2,3,4,5,6,7,8	16	16,17,18	16,17,18,19,20,21,22,23,24
1	3,4,5	9,10,11,12,13,14,15,16,17	17	19,20,21	25,26,27,28,29,30,31,0,1
2	6,7,8	18,19,20,21,22,23,24,25,26	18	22,23,24	2,3,4,5,6,7,8,9,10
3	9,10,11	27,28,29,30,31,0,1,2,3	19	25,26,27	11,12,13,14,15,16,17,18,19
4	12,13,14	4,5,6,7,8,9,10,11,12	20	28,29,30	20,21,22,23,24,25,26,27,28
5	15,16,17	13,14,15,16,17,18,19,20,21	21	31,0,1	29,30,31, 0,1,2, 3,4,5
6	18,19,20	22,23,24,25,26,27,28,29,30	22	2,3,4	6,7,8, 9,10,11, 12,13,14
7	21,22,23	31,0,1,2,3,4,5,6,7	23	5,6,7	15,16,17, 18,19,20, 21,22,23
8	24,25,26	8,9,10,11,12,13,14,15,16	24	8,9,10	24,25,26, 27,28,29, 30,31,0
9	27,28,29	17,18,19,20,21,22,23,24,25	25	11,12,13	1,2,3, 4,5,6, 7,8,9
10	30,31,0	26,27,28,29,30,31,0,1,2	26	14,15,16	10,11,12, 13,14,15, 16,17,18
11	1,2,3	3,4,5,6,7,8,9,10,11	27	17,18,19	19,20,21, 22,23,24, 25,26,27
12	4,5,6	12,13,14,15,16,17,18,19,20	28	20,21,22	28,29,30, 31,0,1, 2,3,4
13	7,8,9	21,22,23,24,25,26,27,28,29	29	23,24,25	5,6,7, 8,9,10, 11,12,13
14	10,11,12	30,31,0,1,2,3,4,5,6	30	26,27,28	14,15,16, 17,18,19, 20,21,22
15	13,14,15	7,8,9,10,11,12,13,14,15	31	29,30,31	23,24,25, 26,27,28, 29,30,31

TABLE VI. FAULT-PROPAGATION PATH OF PRINTCIPHER-96 (93TH ROUND NIBBLE FAULT MODEL

i	SD^{94}	SD^{95}	SD^{96}
0	0,1,2	0,1,2,3,4,5,6,7,8	0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26
1	3,4,5	9,10,11,12,13,14,15,16,17	27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21
2	6,7,8	18,19,20,21,22,23,24,25,26	22,23,24,25,26,27, 28,29,30, 31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
3	9,10,11	27,28,29,30,31,0,1,2,3	17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11
4	12,13,14	4,5,6,7,8,9,10,11,12	12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6
5	15,16,17	13,14,15,16,17,18,19,20,21	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1
6	18,19,20	22,23,24,25,26,27,28,29,30	2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28
7	21,22,23	31,0,1,2,3,4,5,6,7	29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23
8	24,25,26	8,9,10,11,12,13,14,15,16	24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18
9	27,28,29	17,18,19,20,21,22,23,24,25	19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13
10	30,31,0	26,27,28,29,30,31,0,1,2	14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8
11	1,2,3	3,4,5,6,7,8,9,10,11	9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3
12	4,5,6	12,13,14,15,16,17,18,19,20	4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30
13	7,8,9	21,22,23,24,25,26,27,28,29	31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25
14	10,11,12	30,31,0,1,2,3,4,5,6	26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
15	13,14,15	7,8,9,10,11,12,13,14,15	21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
16	16,17,18	16,17,18,19,20,21,22,23,24	16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10
17	19,20,21	25,26,27,28,29,30,31,0,1	11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5
18	22,23,24	2,3,4,5,6,7,8,9,10	6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0
19	25,26,27	11,12,13,14,15,16,17,18,19	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27
20	28,29,30	20,21,22,23,24,25,26,27,28	28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22
21	31,0,1	29,30,31,0,1,2,3,4,5	23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17
22	2,3,4	6,7,8,9,10,11,12,13,14	18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12
23	5,6,7	15,16,17,18,19,20,1,22,23	13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7
24	8,9,10	24,25,26,27,28,29,30,31,0	8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2
25	11,12,13	1,2,3,4,5,6,7,8,9	3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29
26	14,15,16	10,11,12,13,14,15,16,17,18	30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24

i	SD^{94}	SD^{95}	SD^{96}
27	17,18,19	19,20,21,22,23,24,25,26,27	25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19
28	20,21,22	28,29,30,31,0,1,2,3,4	20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14
29	23,24,25	5,6,7,8,9,10,11,12,13	15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4,5,6,7,8,9
30	26,27,28	14,15,16,17,18,19,20,21,22	10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,0,1,2,3,4
31	29,30,31	23,24,25,26,27,28,29,30,31	5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31