Received 7 June 2022, accepted 19 June 2022, date of publication 22 June 2022, date of current version 30 June 2022.

*Digital Object Identifier 10.1109/ACCESS.2022.3185209*

# Single-Byte Error-Based Practical Differential Fault Attack on Bit-Sliced Lightweight Block Cipher PIPO

**SEONGHYUCK LIM**[1], **JAESEUNG HAN**[1], **AND DONG-GUK HAN**[1,2]
[1]Department of Financial Information Security, Kookmin University, Seoul 02707, Republic of Korea
[2]Department of Information Security, Cryptology, and Mathematics, Kookmin University, Seoul 02707, Republic of Korea

Corresponding author: Dong-Guk Han (christa@kookmin.ac.kr)

This work was supported in part by the Military Crypto Research Center by the Defense Acquisition Program Administration (DAPA) and Agency for Defense Development (ADD) under Grant UD210027XD.

**ABSTRACT** With the recent development of the Internet of Things (IoT), related device use is increasing rapidly. As a result, accessing and hijacking the devices is an increasing security threat. The challenges of side-channel security of IoT devices are having a way of coming to the surface due to this physical accessibility. Accordingly, there is active research on lightweight block ciphers to provide security even in resource-scarce environments situations such as IoT. The bit-sliced structure increases memory and time efficiency using an implementation method that replaces a lookup table with a bit-wise operation. Therefore, it is a widely-used design technique for lightweight block ciphers. In this paper, we show a differential fault attack study, a type of side-channel analysis, targeting bit-sliced block ciphers. In particular, it proposes a novel attack rationale on the recently proposed lightweight block cipher PIPO and shows that it applies sufficiently to other bit-sliced block ciphers. The proposed attack is based on a more alleviated attacker's assumption than the previously proposed attack, and it shows that less than 32 fewer fault ciphertext may fully recover the 128-bit of the PIPO 64/128 secret key. It proves that the attack is practical by verifying the attack through the actual electromagnetic fault injection. It also discusses the applicability of various bit-sliced block ciphers and shows how redundancy-based countermeasures might improve fault-robustness. Therefore, when using the bit-sliced block ciphers on IoT devices, we recommend applying appropriate countermeasures against fault injection attacks.

**INDEX TERMS** Side-channel analysis, differential fault attack, fault injection attack, lightweight cryptography, bit-sliced cryptography, PIPO.

## I. INTRODUCTION

Side-channel analysis (SCA) is a type of cryptanalysis that uses physical information, such as power consumption, electromagnetic emission, and timing, that can be may occur when cryptographic algorithms run on real hardware [1]. Even if cryptography is mathematically secure, the vulnerability of physical information is a critical issue because cryptographic algorithms always operate on real devices. SCA necessitates direct or indirect device access. With the advancement of IoT and smart devices, the use of these devices has increased, naturally providing attackers with several opportunities to gain access to or hijack devices. In this context, SCA security has become critical factor to consider.

Differential fault attack (DFA) is a type of SCA in which a secret key is analyzed using difference information generated by inducing a fault in the operation of the device [2]. Easy access to the device further highlights the significance of DFA in particular. This is because access to the target device and fault induction problems, which is the critical problem of DFA, is solved. Naturally, various studies on DFA, especially efforts to realize DFA is in progress. The difficulty of DFA depends on the attacker's assumption, and from the attacker's point of view, DFA methods that can analyze the secret key are required even if the attacker's assumption is weak. The weak attacker's assumption we consider implies that there is no fault-inducing capability at the precise time and no

The associate editor coordinating the review of this manuscript and approving it for publication was Mohamad Afendee Mohamed.

single-bit fault-inducing capability. If there is logic to attack in this environment, it can be a fatal threat because the spectrum of attackers becomes wide. Realizing the DFA requires performing a fault injection (FI) attack. Example of FI attacks include laser fault injection (laser-FI) [3], electromagnetic wave fault injection (EM-FI) [4], clock/power glitch [5], [6], and others. Advances in FI technology have accelerated the realization of DFA. Now, when a novel cipher is designed, a security analysis for DFA is inevitable.

With the advent of the lightweight era, various lightweight block ciphers are being designed, and security test for lightweight block ciphers is essential [7], [8]. Plug-In Plug-Out (PIPO) is a bit-sliced lightweight block cipher designed with the concept of SCA resilient-friendly [9]. The PIPO has the advantage of having less overhead when applying statistical SCA countermeasures such as masking, so it can achieve two goals: SCA security and lightweight. The PIPO is being actively studied from the perspective of security analysis, optimization implementation, and countermeasure design [10]–[13]. Bit-sliced implementations have advantages in lightweight environments such as IoT because they reduce the memory burden required to store the substitution table and improve speed through parallelization of the substitution layers. Compared to the interest in the security of lightweight cryptography, the studies on DFA, especially the bit-sliced implementation, are insufficient. Therefore, additional security evaluations are required for DFA against bit-sliced ciphers. Unlike the S-Box of other SPN-based ciphers, the S-Layer of the bit-sliced implementation has a characteristic that when an error occurs in one byte, it propagates to all bytes. We conducted a study on the DFA method that can overcome these difficulties. Therefore, in this paper, we propose the logic of the DFA, which can pose a real threat to the PIPO among bit-sliced block ciphers.

The primary contributions of this paper are as follows:

- **Proposing the practical model-based DFA on the lightweight block cipher PIPO.** The logic we propose is based on single-byte errors at random positions and is a practical attack that requires fewer fault ciphertexts while alleviating the attacker's assumption compared to the previously proposed DFA.
- **Showing the applicability of the proposed DFA against other bit-sliced ciphers.** The proposed DFA is designed to fit the bit-sliced structure. We demonstrate how the proposed DFA can be simply applied to other bit-sliced block ciphers.
- **Validating of the proposed DFA with EM-FI.** We demonstrated the practical applicability of the proposed attack with direct EM-FI attacks. It shows that the proposed attack is a critical threat to the level that can be performed on a real device.
- **Demonstrating the robustness of FI for PIPO to which the countermeasure is applied.** We implemented a redundancy-based countermeasure for the PIPO and demonstrated its effectiveness against FI attacks. This emphasizes the need to devise countermeasures while using PIPO in the real world.

**TABLE 1. Notations for the PIPO cipher.**

| Parameter | Description |
|---|---|
| $r$ | The number of rounds $(= 13, 17)$ |
| $RK_r$ | $r^{th}$ roundkey |
| $c_r$ | $r^{th}$ round constant |
| $R, R^{-1}$ | R-Layer and inverse R-Layer |
| $S$ | S-Layer |
| $Sb$ | S-Box operation |
| $C, C^{\lightning}$ | Normal&fault ciphertexts |
| $m, m^{\lightning}$ | Normal&fault S-Layer output |
| $j, k$ | Index of the matrix representation $j, k \in [0, 7]$ |
| $a_{col}^{j}$ | $j$-column vector of the matrix |
| $a_{row}^{k}$ | $k$-row vector of the matrix |
| $x$ | S-Layer input |
| $i$ | S-Box input difference |
| $b_p$ | 1-bit intermediate value$(p \in [0, 63])$ |
| $diff$ | S-Box difference table |
| $\|\|$ | Concatenation operation |

Section II of the paper begins with a description of PIPO, DFA, and EM-FI. It also introduces a DFA logic that has been previously studied against bit-sliced block ciphers. Section III describes the logic of the novel DFA on PIPO and shows improvements compared to previous studies. Section IV shows how to apply the proposed logic to other bit-sliced block ciphers. Section V shows the EM-FI evaluation results for the real device, and section VI shows the redundancy-based countermeasure and the FI results when applied. Finally, Section VII concludes the paper with a conclusion and future works.

## II. BACKGROUNDS

### A. PIPO

PIPO is a lightweight block cipher with an SPN structure proposed by Kim *et al.* in 2020 [9]. It is designed for bit-sliced implementation to increase memory and time efficiency by processing the S-Box in parallel as a bit-wise operation. And it encrypts 64-bit blocks and can efficiently apply the SCA countermeasure, which has the advantage of less overhead generated when masking is applied compared to other block ciphers. Depending on the key size, it is divided into the PIPO 64/128 and the PIPO 64/256 and consists of 13 rounds and 17 rounds, respectively. Table 1 shows the notations used to explain the algorithms and attacks in this paper, and Figure 1 shows the overall structure of the PIPO. The PIPO consists of a structure in which S-Layer, R-Layer, and AddRoundKey are repeated after the whitening key is added. The intermediate value 64-bit of the PIPO can be expressed in the $8 \times 8$ matrix, as show in Figure 2. As stated in the following equation, each

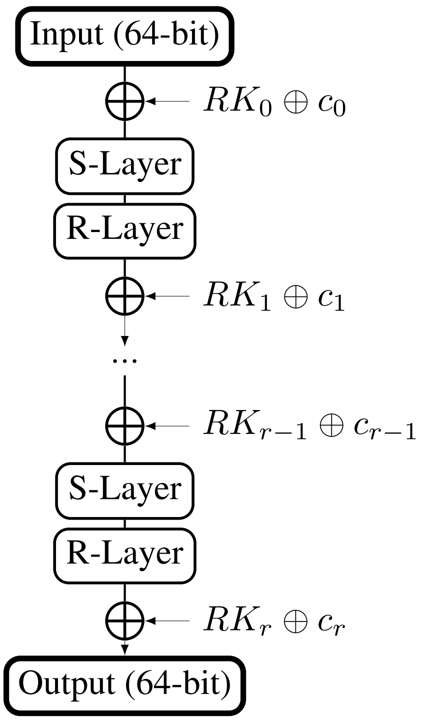


**FIGURE 1.** Overall structure of the PIPO.

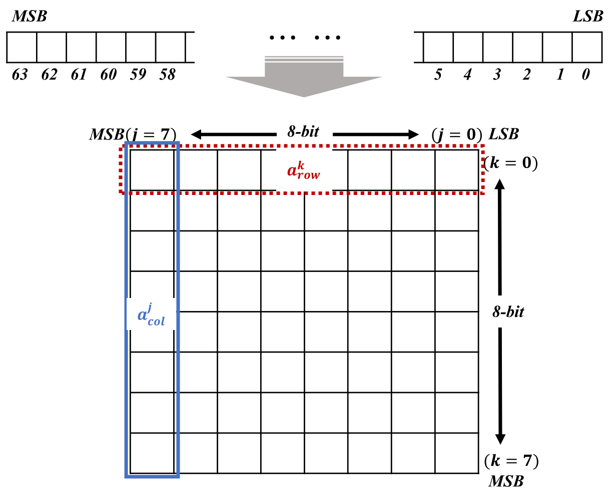


**FIGURE 2.** Operational intermediate value form of the PIPO 64/128.

byte is divided and stored in row units, and the bit-sliced implemented S-Layer is accomplished by bit-wise operation between rows.

$$a^k_{row} = (b_{8\times k+7}||b_{8\times k+6}||\dots||b_{8\times k}),\, k \in [0,7] \quad (1)$$

On the other hand, the S-Box operation is performed in column units from the S-Box perspective. That is, the $a^j_{col}$ value of Figure 2 is used as an input to the S-Box. When a bit-slicing operation that turns over the storage scheme between rows and columns are defined as *BitS*, the S-Layer operation is expressed as follows:

$$S(a) = BitS\left(Sb\left(a^0_{col}\right), Sb\left(a^1_{col}\right), \dots, Sb\left(a^7_{col}\right)\right) \quad (2)$$

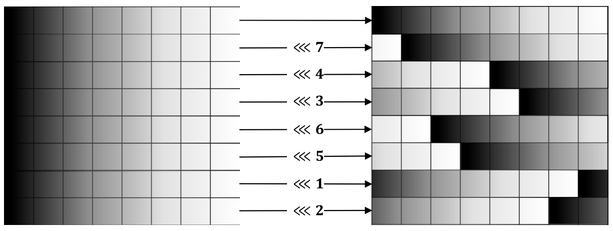


**FIGURE 3.** R-layer configuration of the PIPO 64/128.

Figure 3 shows the R-Layer operation of the PIPO. The R-Layer consists of a rotation operation in row units and can be expressed as follow:

$$R(a) = \Big(a^0_{row} \lll 0, a^1_{row} \lll 7, a^2_{row} \lll 4, a^3_{row} \lll 3, \times a^4_{row} \lll 6, a^5_{row} \lll 5, a^6_{row} \lll 1, a^7_{row} \lll 2\Big) \quad (3)$$

The PIPO's key schedule has a simple structure in which the secret key is divided by 64-bit and repeatedly used, with round constants added for each round. The key schedule of PIPO 64/128 is expressed as follows:

$$K = K_1||K_0, \quad RK_i = K_{i\ mod\ 2} \oplus c_i$$
$$where\ i = 1, 2, \dots, 13\ and\ c_i = i \quad (4)$$

## B. DIFFERENTIAL FAULT ATTACKS

DFA is a type of semi-invasive attack within SCA that combines traditional differential analysis and FI attacks. The DFA mechanism was first proposed in 1997 by Biham and Shamir [2]. Starting with DES, research on the DFA has steadily conducted for various cryptographic algorithms [14]–[17]. In the DFA, the attacker can cause malfunction by injecting artificial faults at specific times during an encryption operation. Additionally, he can observe and collect fault ciphertexts that have occurred. The attacker deduces secret information by comparing and using the difference values between the normal and fault ciphertext. When performing the DFA in a real world, the feasibility of the attack depends on the fault model. According to the FI scale and its positioning capability, the fault model is classified as follows:

- **FI Scale**
  - `Bit flip`: A single-bit of the precise position desired by the attacker belonging to a specific byte/word is flipped.
  - `Byte Error`: A specific byte is altered to a random value that the attacker does not know.
  - `Word Error`: A specific word is altered to a random value that the attacker does not know.
- **Positioning Capability**
  - `Chosen Position`: The attacker can specify where the faults are injected.
  - `Random Position`: The attacker cannot specify where the faults are injected. The faults occur in a random position.

**TABLE 2. Comparison of DFA on bit-sliced block ciphers.**

| Ref. | Target Cipher | Fault Model | Experiment |
|---|---|---|---|
| [18] | RECTANGLE | Forced to 0, Continuous/ Entire Bit Flip, etc. | Simulation |
| [19] | SCREAM | Bit Flip | EM-FI |
| [20] | PIPO | Bit Flip | Simulation |
| This paper | PIPO | Byte Error | EM-FI |

The smaller the FI scale, the stronger the attacker's assumption. Furthermore the attacker's assumption is strong when the fault position must be specified. The DFA can be performed in a real environment using techniques such as a row-hammer attack, laser-FI, and EM-FI, among others. Although EM-FI has difficulties realizing a strong fault model, relatively practical attacks are possible. This paper proposes DFA logic for the PIPO based on a relaxed attacker's assumption at a reproducible level through EM-FI.

### C. ELECTROMAGNETIC FAULT INJECTION ATTACK

EM-FI attacks can equally induce logic timing violations similar to those caused by clock glitch attacks, and can also cause transistor malfunctions that voltage glitchs and laser-FI attacks can cause. The probe tip used for EM-FI has a form in which a copper coil is coiled around a ferrite. The EM field is formed around the coil by allowing an instantaneous current to flow in this coil. The generated EM pulse penetrates the metal layer of the target chip and can induce voltage glitches and eddy currents in circuit loops. As a result, the transistor may malfunction, allowing for attacks such as altering data stored in memory or skipping instruction. In the case of the EM-FI attack, unlike the laser-FI attack, it is possible to inject faults by scanning the surface of the target board without a decapsulation process. Unlike row-hammer attack, it does not necessitate an accurate memory address. Therefore it can be practically used for attack.

### D. PREVIOUS DFA ON THE BIT-SLICED BLOCK CIPHERS

The block ciphers proposed for the purpose of bit-sliced implementation include ROBIN [21], RECTANGLE [22], PRIDE [23], PIPO, etc. The DFA logic on each cipher has been proposed, but it is often targeted at a table lookup implementation of that cipher [24]. DFA research on actual bit-sliced implementation block ciphers is lacking. Sinha and Karmakar proposed the DFA on bit-sliced implemented RECTANGLE-80 in 2018 [18]. They observed the exhaustive searching range according to the fault model based on forcing the bit to be 0, the continuous bits flip, and the entire bit flip, etc. However, the strong fault model with coerciveness and continuity deal is impractical, and the relatively weak fault model has a limitation in that the exhaustive search range is large. In other words, there is no logic to fully analyzing the secret key. Lac *et al.* proposed a bit flip-based DFA for LS-Design-based cipher SCREAM in 2017 [19]. And, Lim *et al.* first proposed the DFA on PIPO in 2021 [20]. Their proposed logic is based on a single-bit flip model. Since FI must be performed for all bits, they presume that the attacker can determine the byte position where the faults are injected. Table 2 shows a comparison of previous DFA studies on bit-sliced block ciphers. Most of the previous studies require precise fault based on bit flip, and only one has carried out an actual FI attack. Therefore, we explored the DFA logic based on the byte error model to perform an easier FI attack based on the alleviated attacker's assumption. In particular, the detailed comparison with [20], which targets the same cipher PIPO, is made in Section III-C.

## III. PROPOSED DFA ON PIPO

In this section, we describe the proposed DFA method for the PIPO. This attack uses the random byte error model. With this fault model, the attacker's assumption is mitigated, compared with previous attack using the single-bit flip model. Using the PIPO 64/128 as an example, the attack strategy is described in detail. The same attack is repeated on additional rounds to fully retrieve the secret key for the PIPO 64/256.

### A. ATTACKER'S ASSUMPTION

The attacker can gain access to or acquire a device that encrypts using the PIPO and can cause a fault with respect to a single-byte when the PIPO operates encryption. However, the attacker cannot specify the position of the byte where the fault occurs, and the target byte changes to an arbitrary value that the attacker is unaware of. He can observe normal and fault ciphertexts and collect as many fault ciphertexts as needed to perform the attack. Faults are injected into the S-Layer input of the lower round, and normal and fault ciphertext occurs. He does not need to know the precise memory location information required for the attack. However, the round operation starting point can be identified through simple power analysis, allowing the approximate FI timing can be regulated.

### B. PROPOSED DFA SCHEME

Figure 4 shows an overview of the proposed DFA on the PIPO 64/128. First, the ***Fault 1*** procedure analyzes $RK_{13}$ by injecting a fault into a random single-byte of the last round S-Layer input. Then, using the secret information obtained from ***Fault 1***, ***Fault 2*** is carried out. This process injects a fault into a random single-byte of the penultimate round S-Layer input and is performed similarly to ***Fault 1***. Each procedure consists of 4 steps, and the detailed attack process is as follows and is described based on ***Fault 1***.

#### 1) [STEP 1] CALCULATE S-LAYER INPUT DIFFERENCE

The S-layer output difference ($\Delta m$) can be calculated without knowing the round key. At this time, R-Layer is a linear

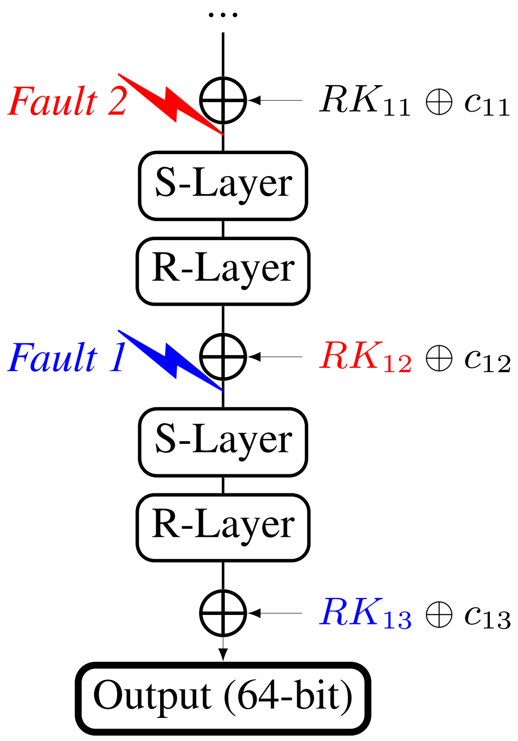


**FIGURE 4.** Fault position of proposed DFA on the PIPO 64/128.

function, so the equation can be developed as follows:

$$\begin{aligned}\Delta m &= R^{-1}\left(C \oplus (RK_{13} \oplus c_{13})\right) \\ &\quad \oplus R^{-1}(C^{\lightning} \oplus (RK_{13} \oplus c_{13})) \\ &= R^{-1}(C) \oplus R^{-1}(RK_{13} \oplus c_{13}) \\ &\quad \oplus R^{-1}(C^{\lightning}) \oplus R^{-1}(RK_{13} \oplus c_{13}) \\ &= R^{-1}(C) \oplus R^{-1}(C^{\lightning}) \end{aligned} \tag{5}$$

Therefore, the S-Layer output difference can only be calculated with information about the pair of normal and fault ciphertexts without round key information. If the difference is caused by injecting a fault into a specific byte of the S-Layer input, the output difference occurs in the column containing the bit in which the difference propagates due to the bit-sliced implementation characteristic. Figure 5 shows an example of the relationship between the S-Layer input and output difference. If a fault is injected into the second byte of the S-Layer input and bit flip occurs at indexes 0, 4, 6, and 7 of that byte, the output difference occurs in each column respectively (marked in red). This is because S-Box operations are conducted in column units (marked in blue). However, there is no difference that occurs in the other columns (marked in green). As a result, while computing the S-Layer output difference in the preceding process using the pair of normal and fault ciphertexts, it is possible to estimate the S-Layer input difference by observing whether each column is 0 or not. In Figure 5, the input difference can be estimated as 0xd1.

### 2) [STEP 2] SEARCH FAULT-OCCURRING BYTE INDEX

**STEP 1** can be used to determine the S-Layer input difference. However, regardless of the fault-occurring byte position, the input difference affects all bytes of the output difference. The attacker is unsure which byte of the S-Layer input triggers the fault. Therefore, this paper proposes a method of searching for the position of a fault-occurring byte to solve this problem. From the S-Box viewpoint, the input difference of the S-Box has a 1-bit difference regardless of the fault-occurring byte position. In other words, possible S-Box input difference values ($i$) are 1, 2, 4, 8, 16, 32, 64, and 128 for which a 1-bit difference table can be configured. When the fault occurs in column j of the S-Layer intermediate value matrix, the S-Box difference equation is constructed as follows:

$$Sb\left(x^{j}_{col}\right) \oplus Sb\left(x^{j}_{col} \oplus i\right) = \left(m \oplus m^{\lightning}\right)^{j}_{col} \tag{6}$$

Using the 1-bit difference table, verify the existence of $x^{j}_{col}$ with output difference $\left(m \oplus m^{\lightning}\right)$ as the output of the equation. By verifying this for all $i$, candidates for possible S-Box input differences are reduced. When the fault occurs in a single-byte of the S-Layer input, a difference occurs in several bits. that is, the S-Box difference equations can be used for multiple columns. Algorithm 1 is the fault-occurring byte index search algorithm (*FOBISA*) proposed in this research work, and it searches for the exact location of the fault by using the difference information of multiple columns. The input of the proposed algorithm is a non-zero S-Box output difference, and each output difference is defined as $D_0, D_1, \ldots, D_{d-1}$. Furthermore, $d$ denotes the number of

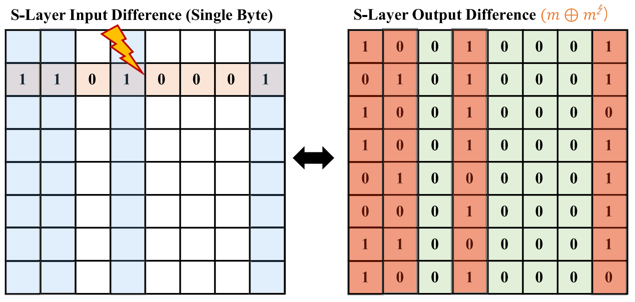


**FIGURE 5.** Relation between S-Layer input difference and output difference.

**Algorithm 1** Fault-Occurring Byte Index Search Algorithm

```
Input  : Non-zero S-Box output differences $D_0, \ldots, D_{d-1}$
         ($d$ : number of columns with non-zero differences)
Output: $A = \{a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$, $a_i = 0$ or 1
1: $A \leftarrow \{1, 1, 1, 1, 1, 1, 1, 1\}$
2: for $j = 0$ to $d - 1$ do
3:   for $k = 0$ to 7 do
4:     if $diff[2^k][D_j] == 0$ then
5:       $A[k] \leftarrow 0$
6:     end if
7:   end for
8: end for
9: Return $A$
```

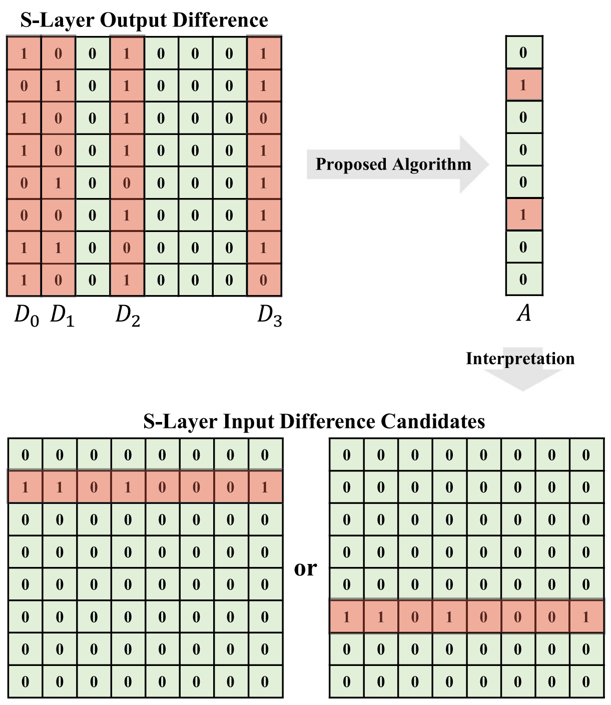

S-Layer Output Difference
Proposed Algorithm
$D_0$ $D_1$ $D_2$ $D_3$
$A$
Interpretation
S-Layer Input Difference Candidates
or

**FIGURE 6.** An operation example of the proposed *FOBISA*.

**TABLE 3.** The probability that the byte position is uniquely determined by the number of flipped bits.

| # of bits | Probability | # of bits | Probability |
|---|---|---|---|
| 1-bit | $\approx 30.1\%$ | 5-bit | $\approx 96.9\%$ |
| 2-bit | $\approx 68.3\%$ | 6-bit | $\approx 98.5\%$ |
| 3-bit | $\approx 85.6\%$ | 7-bit | $> 99.0\%$ |
| 4-bit | $\approx 93.3\%$ | 8-bit | $> 99.0\%$ |

non-zero output differences. The proposed algorithm produces 8 elements array $A$. The index of an element with a value of 1 among elements of $A$ denotes a byte index at which it is determined that a fault has been injected. Figure 6 shows an example of *FOBISA* procedure. When the fault is injected, the bit in which the difference occurs is arbitrary, and the probability that the S-Box input difference is uniquely determined by the algorithm varies according to the number of flipped bits. Table 3 shows the probability that the fault-occurring byte position is uniquely determined based on the number of bits flipped in a single S-Layer input byte. When a single-byte error occurs at a random position, the fault-occurring position for the S-Layer input is uniquely determined with about 88.4% probability.

### 3) [STEP 3] DETERMINE S-LAYER INPUT CANDIDATES

When **STEP 2** is completed, the S-Box input and output difference pair are accurately determined. Candidates for the S-Box input $\left(x^j_{col}\right)$ satisfying equation 6 can be minimized using the pair. Multiple different information in which an error occurs in the same column is required to confirm with only one candidate. That is, it determines the S-Box input value that is commonly satisfied for multiple S-box input and output difference pairs. The S-Box input can be uniquely predicted by 3 pairs with about 89.1% probability and 4 pairs with about 98.8% probability.

### 4) [STEP 4] RECOVER ROUND KEY

**STEP 1** to **STEP 3** should be repeated to analyze the S-Layer input by column unit (S-Box unit). When all columns have been analyzed, all 64-bit of the S-Layer can be confirmed, and the last round key may be recovered as shown in the equation below:

$$RK_{13} = P(S(x)) \oplus C \oplus c_{13} \tag{7}$$

### 5) RECOVER SECRET KEY

The $RK_{13}$ was recovered by following the **STEP 1** to **STEP 4** method on ***Fault 1*** The $RK_{12}$ is recovered by repeating the **STEP 1** to **STEP 4** method on ***Fault 2*** as shown in Figure 4. The recovered $RK_{13}$ is used to calculate the penultimate round S-Layer output difference. In this process, only additional overhead that occurs in the intermediate value calculation is necessary, and no new logic is required. Finally,the inverse key schedule is performed on the recovered $RK_{13}$ and $RK_{12}$ as follows:

$$K = (RK_{13} \oplus 0\text{xd}) \,||\, (RK_{12} \oplus 0\text{xc}) \tag{8}$$

As a result, all 128-bit of the secret key of PIPO 64/128 can be acquired.

## C. ATTACK PERFORMANCE

Analyzing all bytes of the secret key requires analyzing on all columns of the S-Layer input. In other words, the attack needs error information for each column. The flipped bit-position of the fault ciphertext may be reliably recognized using the proposed DFA **STEP 1** method. It is also simple to filter out the fault ciphertext needed for analysis. Since the FI attack is repeatedly performed numerous times, the probability of obtaining information about all columns approaches 1. The random byte error model affects several bits of stored memory. There are variations depending on the device environment, but 3∼4 bits are most likely to change on average. Additionally, because there is a filtering process, a set containing error information for all columns can be collected, and approximately four fault ciphertexts are sufficient to satisfy this. As shown in **STEP 3** of the proposed DFA, when analyzing each column, if more than 3 or 4 errors for different byte positions are used, one column can be uniquely determined by high probability. Finally, the proposed DFA can recover the round key using up to about 16 fault ciphertexts. Because ***Fault 1*** and ***Fault 2*** follow the same method, fewer than 32 fault ciphertexts are sufficient to analyze the secret key. Table 4 compares the performance of the previous and proposed DFA on the PIPO 64/128. The proposed DFA is an efficient and practical attack that uses a

**TABLE 4. Performance comparison of previous and proposed DFA on the PIPO.**

| Methods | Fault Models | Fault Positions | # Fault-Injected Ciphertexts | # Key Candidates |
|---|---|---|---|---|
| [20] | Specific byte position, random single-bit flip | S-Layer input bytes of last round and penultimate round | 64 | 1 |
| Proposed Method | Random byte position, single-byte error | S-Layer input bytes of last round and penultimate round | $\leq$32 | 1 |

mitigated fault model and necessitates few fault ciphertexts compared to the previous attack.

## IV. APPLYING THE PROPOSED DFA TO OTHER BIT-SLICED BLOCK CIPHERS

This section discusses the applicability of the proposed DFA to various bit-sliced block ciphers such as ROBIN, FANTOMAS, and RECTANGLE, etc. The operating-unit and S-Box size of each cipher is different but can be expressed in the form of a matrix as shown in Figure 2. And the constructed functions operate in row units, and from S-Box viewpoint, they operate in column units. Other bit-sliced block ciphers shown in Figure 7 have a structure in which the substitution layer, permutation layer, and key addition layer all repeat. Accordingly, when executing FI, the attacker can calculate the output difference of the substitution layer using the ciphertext difference. At this point, the output difference occurs in several columns, demonstrating that the relationship as shown in Figure 5 exists equally. Through Algorithm 1, the attacker can analyze the last round key by identifying the fault-occurring byte index. Since the size of the S-Box varies depending on the cipher, the number of fault-occurring columns required to reduce it to one candidate varies. However, various faults occur in FI attacks, and the attackers can sufficiently filter fault ciphertexts that are favorable to them. Consequently, the proposed attack is designed to be suitable for the bit-sliced structure, allowing it to be easily applied to several bit-sliced block ciphers.

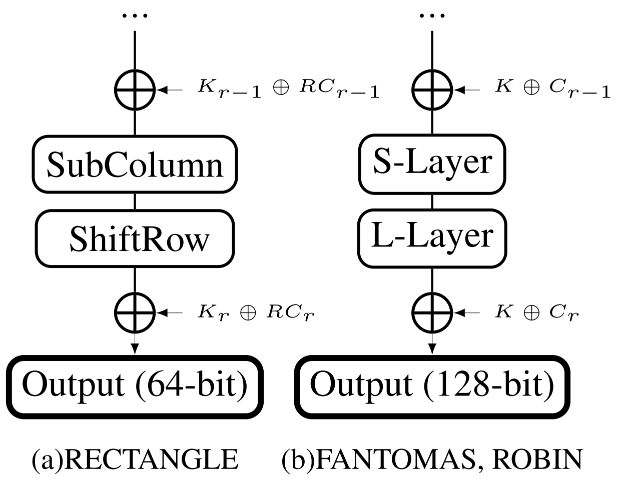


**FIGURE 7. The structure of other bit-sliced block ciphers.**

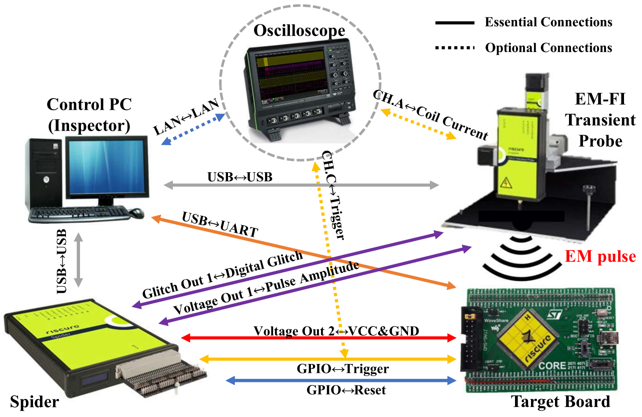


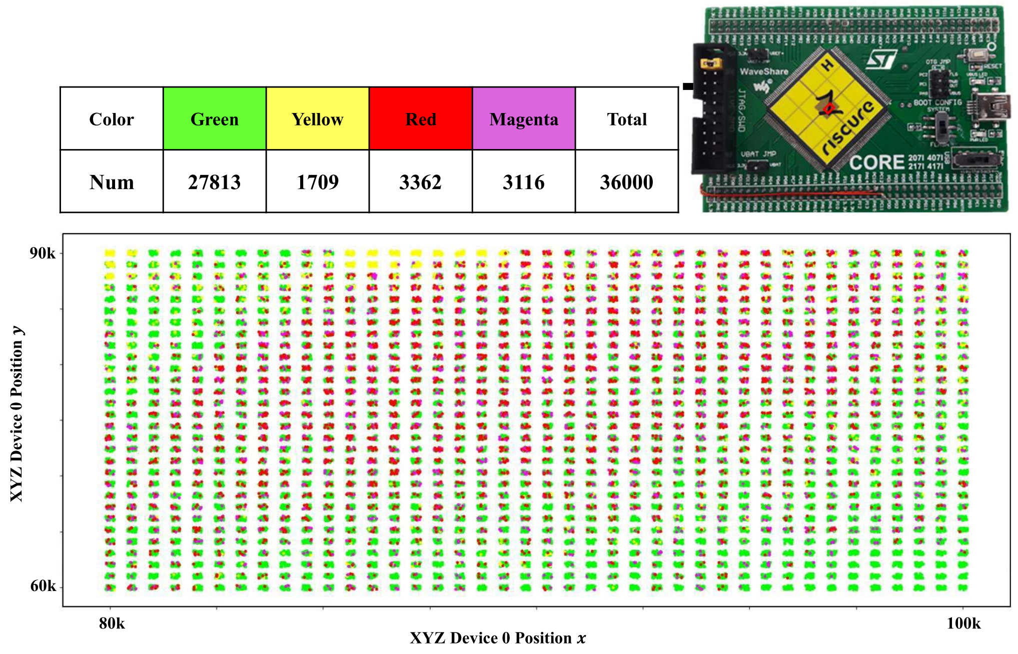
**FIGURE 8. EM-FI experimental environment.**

## V. EVALUATION OF PROPOSED DFA USING EM-FI

This section shows that the proposed DFA can be practically performed in real devices using actual EM-FI attack experiments. First, we identify the time to inject the faults by observing the electromagnetic wave generated when the PIPO 64/128 operates. Then the EM-FI attack was repeatedly performed by setting the parameters, and the fault ciphertexts were collected. More details will be described in the subsections.

### A. EM-FI ATTACK ENVIRONMENT

Figure 8 shows the experimental environment, which used EM-FI equipment provided by Riscure. In the figure, solid lines represent essential configurations and dotted lines indicate optional configurations. The experimental equipment consists of EM-FI Transient Probe [25], Spider [26], PC, oscilloscope, and target device. The Control PC uses Inspector software [27] to control the experimental environment and to process and analyze collected data. It controls the target board's encryption operation through UART communication and receives ciphertexts. The Spider and EM-FI Transient Probe receive operating parameters from the PC through USB communication, and the operation is controlled based on the received information. The Spider controls the reset signal and trigger signal to the target board and controls the signal and power required for the EM-FI Transient Probe to inject a fault. The EM-FI Transient Probe moves along the coordinates of the XYZ-table and injects EM faults. An oscilloscope is used to observe trigger signals, observe EM traces that



| Color | Green | Yellow | Red | Magenta | Total |
|---|---|---|---|---|---|
| Num | 27813 | 1709 | 3362 | 3116 | 36000 |

**FIGURE 9.** EM-FI result against piñata board operating the PIPO 64/128.

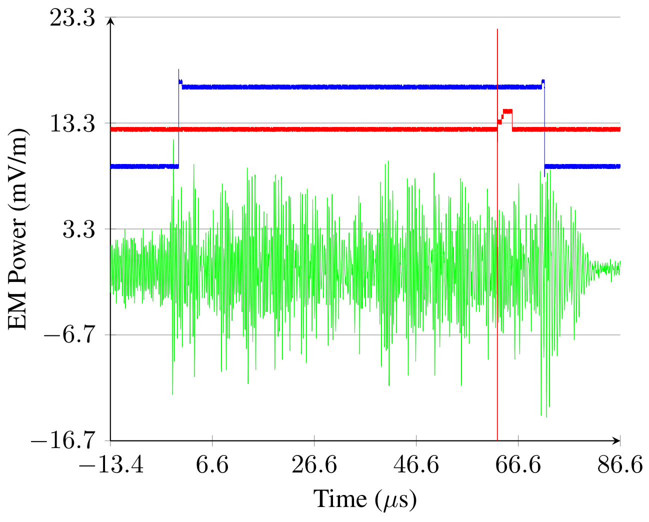


**FIGURE 10.** EM traces of PIPO 64/128 observed by oscilloscope.

occur when the encryption algorithm operates and determine when faults are injected. We experimented with Riscure's Piñata board [28], which uses an Arm Cortex-M4F microcontroller [29]. The PIPO 64/128 was implemented on the Piñata board based on 8-bit variables, and was compiled using the GNU Arm Embedded Toolchain version 4.8 [30].

### B. EM-FI ATTACK RESULTS

Figure 10 shows the EM traces generated when PIPO 64/128 is operated in an I/O trigger environment. The green signal in the figure represents PIPO 64/128 encryption, the blue signal represents the trigger, and the red signal represents the FI. Simple EM analysis [1] can be used to distinguish each round of PIPO 64/128. We could predict the expected start times of the last and penultimate rounds of the S-Layer. We then set delays to inject faults at the identified times. When attacking the last round, a random delay of roughly 66,000 *ns* was applied, and when attacking the penultimate round, roughly 62,000 *ns* was applied. When a fault is injected into a specific byte of the S-Layer input, the error is propagated to all columns containing the modified bits. Therefore, the greater the Hamming weight of the S-Layer input difference, the higher the probability of errors propagating to all bytes of the ciphertext. Figure 9 shows the results of the EM-FI attack on PIPO 64/128. The red square box area of the target device was scanned and the fault ciphertexts were filtered into four types. The green type is the normal ciphertexts, the yellow type is the abnormal behaviors, the red type is when the difference occurs in some bytes of the ciphertexts, and the magenta type is when the difference occurs in all bytes of the ciphertexts. The experiment resulted in 3,362 red types, and 3,116 magenta types being filtered. However, both types of fault ciphertexts contain unidentified causes. Therefore, before performing the proposed DFA, it is important to filter the fault ciphertexts in which a single-byte error occurred. In this experiment, we built Algorithm 2 to filter ciphertexts in which the fault- occurring byte was determined to be merely one. Algorithm 1 was utilized for this. Except for duplicates, 164 types were among the 3,362 red type fault ciphertexts, and 110 fault ciphertexts were validated as single-byte errors after filtering. For the magenta type, 241 of the 3,116 fault ciphertexts existed except for duplicates and 139 after the filtering. Table 5 shows examples of fault ciphertexts and

**Algorithm 2** Fault Ciphertext Filtering Algorithm

```
Input  : Normal Ciphertext($C$),
         Fault Ciphertexts($C_0, C_1, \ldots, C_{n-1}$)
Output: $X$−Fault Ciphertexts for analysis($T$)
 1: $T[X][2] \leftarrow NULL, k \leftarrow 0$
 2: for $i = 0$ to $n-1$ do
 3:     $N = NonZeroColumnSet(R^{-1}(C \oplus C_i))$
 4:     $A = FOBISA(N)$                    # Using Algorithm 1.
 5:     $cnt \leftarrow 0$
 6:     for $j = 0$ to 7 do
 7:         if $A[j] == 1$ then
 8:             $T[k][1] \leftarrow j$        # Store Fault-Occurring Byte Index.
 9:             $cnt \leftarrow cnt + 1$
10:         end if
11:         if $cnt == 1$ then
12:             $T[k][0] \leftarrow C_i$      # Index Confirmed → Store Fault Data.
13:             $k \leftarrow k + 1$
14:         end if
15:     end for
16: end for
17: Return $T$
```

**TABLE 5.** Fault ciphertexts filtered through Algorithm 2.

| Error byte position | Fault ciphertexts |
|---|---|
| Normal Cipher | 27 03 5D AD 81 29 6B 6B |
| 0th byte error | 6F 31 D9 EC 82 A4 7B DA |
| | 05 02 7F AD 09 69 2B 6B |
| 1st byte error | 23 41 11 89 B0 31 E3 7B |
| | 25 02 7D AD 01 69 6B 6B |
| 2nd byte error | F7 8A 65 BF 31 31 CC 6A |
| | 37 0B 1F AD 88 AD 03 EB |
| 3rd byte error | 1D 1E 7E A4 0F 0C 69 27 |
| | 27 06 BF BD 08 AD 37 CB |
| 4th byte error | 2F 31 1F CD 80 21 23 5A |
| | 07 03 D0 2B 89 3C 6B A8 |
| 5th byte error | 06 83 4F A4 C1 8D 61 FF |
| | 07 03 59 AF 89 2D 6B EA |
| 6th byte error | 06 8B 4E 24 C5 8D 61 BF |
| | 05 12 5F BC 81 6D 6B 6B |
| 7th byte error | 01 50 33 88 38 35 E3 73 |
| | 0D 02 7F ED 0B 69 3B 6B |

fault-occurring byte positions derived by filtering the two classifications. Table 6 shows the number of fault ciphertexts obtained based on the error type. Since the degree of vulnerability for each register varies based on the target board, there is a difference in the success rate of fault induction for each byte. However, more than 10 fault ciphertexts were obtained for each byte.

### C. PROPOSED DFA RESULTS

The proposed DFA was carried out using the fault ciphertexts obtained from the Section V-B experiment. The

**TABLE 6.** Number of ciphertexts obtained according to fault type.

| Error byte position | # of faults (Red) | # of faults (Magenta) |
|---|---|---|
| 0th | 2 | 9 |
| 1st | 12 | 22 |
| 2nd | 15 | 11 |
| 3rd | 6 | 22 |
| 4th | 15 | 23 |
| 5th | 8 | 22 |
| 6th | 41 | 5 |
| 7th | 11 | 25 |

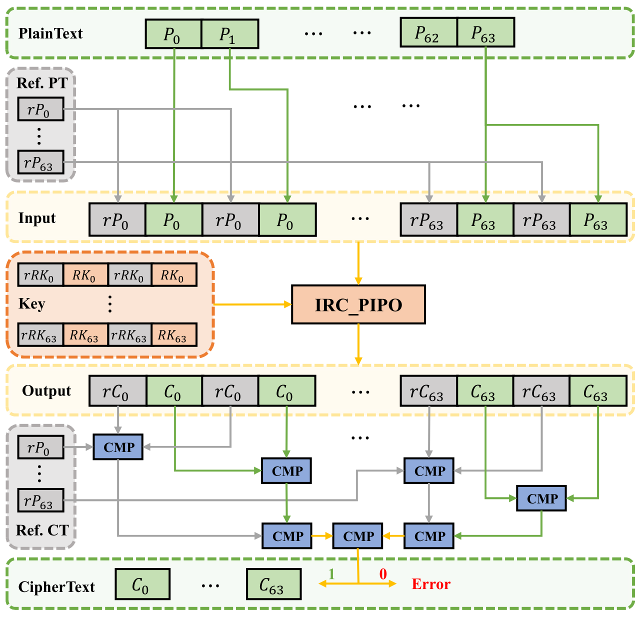

PlainText
$P_0$
$P_1$
$P_{62}$
$P_{63}$
Ref. PT
$rP_0$
$rP_{63}$
Input
$rP_0$ $P_0$ $rP_0$ $P_0$
$rP_{63}$ $P_{63}$ $rP_{63}$ $P_{63}$
$rRK_0$ $RK_0$ $rRK_0$ $RK_0$
Key
$rRK_{63}$ $RK_{63}$ $rRK_{63}$ $RK_{63}$
IRC_PIPO
Output
$rC_0$ $C_0$ $rC_0$ $C_0$
$rC_{63}$ $C_{63}$ $rC_{63}$ $C_{63}$
CMP
Ref. CT
CipherText
$C_0$
$C_{63}$
1
0
Error

**FIGURE 11.** Overall configuration of the IRC PIPO.

fault ciphertexts classified in magenta differ in all bytes, so when the fault is injected, many bits are flipped in the fault-occurring byte of the S-Layer input. Therefore, we first selected the magenta type fault ciphertexts and used them for analysis. However, in the filtering through Algorithm 2, it was considered a single-byte error, but there were fault ciphertexts that prevented analysis as a false positive. Therefore, when the proposed DFA was applied among the filtered fault ciphertexts we searched for the optimal set determined by only one key and were able to uniquely determine the correct last round key through 7 fault ciphertexts. After that, the same process was performed for the penultimate round. As a result, we could fully recover 128-bit of the secret key of PIPO 64/128 with fewer than 32 fault ciphertexts.

## VI. COUNTERMEASURE

Various countermeasures have been proposed for robustness against FI attacks. There are hardware-level countermeasures, such as using abnormal EM or power detection

**TABLE 7. Performance comparison for encrypting.**

| ARM Cortex-M4 | Time (clock cycle) | Size (bytes) |
|---|---|---|
| Normal PIPO | 3,694 | 403 |
| IRC PIPO | 6,291 | 727 |

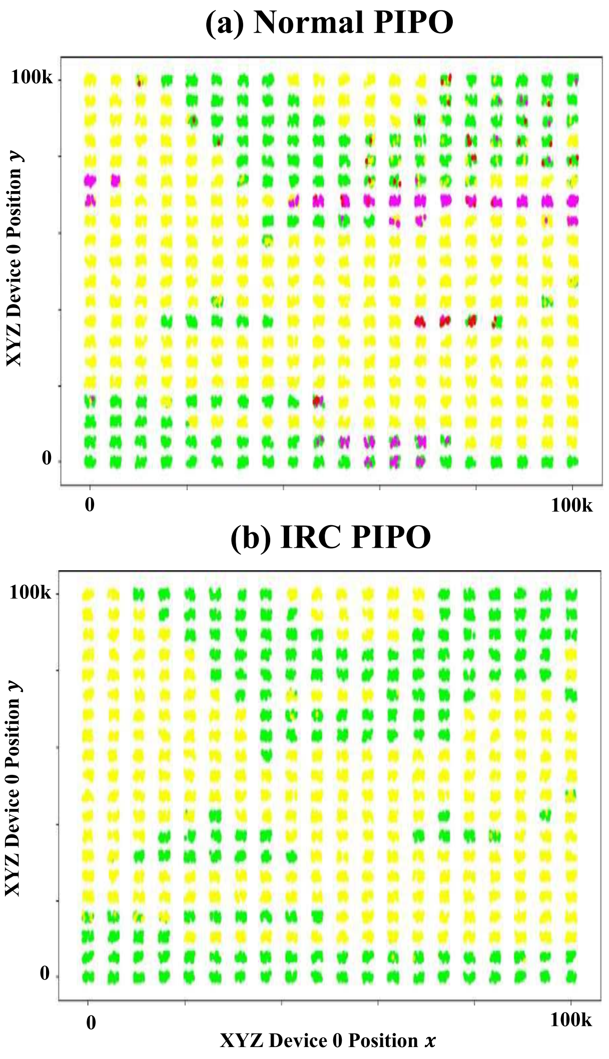


**FIGURE 12. Comparison of EM-FI attack results based on the application of countermeasure.**

modules [31], and software-level countermeasures, such as the redundancy technique, which repeats and verifies the same operations [32]–[34]. Lac *et al.* proposed an efficient FI countermeasure for lightweight cryptography based on SIMD instructions in 2018 [35]. They proposed an internal redundancy countermeasure (IRC) that encrypts and compares two bytes from the reference block and two bytes from the data block in conjunction. Additionally, using SIMD instructions, ciphers based on an 8-bit process were parallelized to 32-bit process operation types. In this paper, the fault-robustness of the IRC PIPO applied the aforementioned countermeasure to PIPO was validated. Figure 11 shows the overall layout of the IRC PIPO. SIMD instructions offer simple parallelization of the Addroundkey and S-Layer, but do not support rotation operations. Therefore, we performed the parallelization of the R-Layer using masking. The reconstructed R-layer is shown in the following equation:

$$
\begin{aligned}
&R_{IRC}((a_0, a_1, \ldots, a_7))\\
&= a_1 \leftarrow ((a_1\&0\text{xfefefefe}) \gg 1)\\
&\quad \oplus ((a_1\&0 \times 01010101) \ll 7)\\
&\quad a_2 \leftarrow ((a_2\&0\text{xf0f0f0f0}) \gg 4)\\
&\quad \oplus ((a_2\&0 \times 0\text{f0f0f0f}) \ll 4)\\
&\quad a_3 \leftarrow ((a_3\&0\text{xe0e0e0e0}) \gg 5)\\
&\quad \oplus ((a_3\&0 \times 1\text{f1f1f1f}) \ll 3)\\
&\quad a_4 \leftarrow ((a_4\&0\text{xfcfcfcfc}) \gg 2)\\
&\quad \oplus ((a_4\&0 \times 03030303) \ll 6)\\
&\quad a_5 \leftarrow ((a_5\&0\text{xf8f8f8f8}) \gg 3)\\
&\quad \oplus ((a_5\&0 \times 07070707) \ll 5)\\
&\quad a_6 \leftarrow ((a_6\&0 \times 80808080) \gg 7)\\
&\quad \oplus ((a_6\&0 \times 7\text{f7f7f7f}) \ll 1)\\
&\quad a_7 \leftarrow ((a_7\&0\text{xc0c0c0c0}) \gg 6)\\
&\quad \oplus ((a_7\&0 \times 3\text{f3f3f3f}) \ll 2)
\end{aligned} \tag{9}
$$

The use of 32-bit processes is inevitable and increases the overall memory overhead, but the time overhead mainly occurs in the operation of comparing redundancy. Table 7 shows an encryption performance comparison between the normal PIPO and the PIPO to which a countermeasure is applied. The comparison was carried out on ARM Cortex-M4 boards that have been subject to FI. Figure 12 shows the results of performing FI attacks on each. The normal PIPO was 446 out of 8,000 times when performed on the same area, and faults occurred in approximately 5.6% of the cases. On the other hand, it was confirmed that IRC PIPO did not create any faults and was completely filtered as abnormal behaviors (yellow type) when faults occurred. In conclusion, it can be seen that the countermeasure used improved the FI-robustness. To attack this redundancy-based countermeasure, not only FI for key acquisition is necessary, but also FI for skipping the fault checking area. Therefore, the attacker needs high-level techniques to inject multiple faults during single encryption.

## VII. CONCLUSION

This paper proposed a DFA on the PIPO based on a single-byte error model in a random position. The proposed DFA could fully recover 128-bit of the secret key of the PIPO 64/128 with overwhelming probability using less than 16 fault ciphertexts each in two rounds. Compared to the previously proposed DFA, our DFA is a practical attack that requires half the fault ciphertext while also alleviating the attacker's assumption. Additionally, the proposed attack logic is designed to be suitable for the bit-sliced structure-agnostic; thus, it may be simply applied to bit-sliced block ciphers other than the PIPO. We used EM-FI to evaluate the proposed DFA. By proposing an algorithm capable of filtering the fault ciphertexts, it was possible to obtain the fault ciphertexts required to analyze the secret key. In reality, we derived the result of accurately recovering the last round key using 7 fault ciphertexts. Consequently, the proposed attack demonstrates to be a practical attack that may be used in the real world. This paper contributes to raising awareness about FI security for bit-sliced block ciphers that are being considered in the IoT environment. Various countermeasures are being applied to construct a safe environment against FI attacks. As an example, there is a countermeasure based on redundancy to defend FI. We applied a redundancy-based countermeasure to the PIPO and demonstrated its FI-robustness. This paper experimentally showed that IRC PIPO prevents attackers from acquiring the necessary fault ciphertexts for a single-FI attack. Consequently, our experimental results emphasize the need for countermeasures when operating the PIPO in the real world. In the future, we intend to apply the proposed DFA to various bit-sliced block ciphers and upload the PIPO to various test boards to undertake EM-FI attacks. In particular, we plan to perform FI attacks on lightweight block ciphers in the hardware environment. This plan will show that the proposed DFA is a highly scalable attack applicable to various target bit-sliced ciphers and environments. Additionally, we plan to launch double-FI attacks on IRC PIPO. Double-FI attack strategies are necessary to attack the block cipher against which countermeasures are applied. For a double-FI attack to be successful, it must bypass checking for redundancy at the last point, as well as the induce faults in the true encrypt operation. In other words, continuous FI at different time points is required, and a more precise attack execution capability is required. Therefore, in the future, we plan to study attack methods against IRC PIPO by trying various methods such as FI using heterogeneous fault sources [36] as well as basic multiple-FI.

## REFERENCES

[1] P. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," in *Proc. IACR CRYPTO*, in Lecture Notes in Computer Science, vol. 1109. Santa Barbara, CA, USA: Springer, Aug. 1996, pp. 104–113, doi: 10.1007/3-540-68697-5_9.

[2] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 1294. Santa Barbara, CA, USA: Springer, 1997, pp. 513–525, doi: 10.1007/BFb0052259.

[3] S. P. Skorobogatov and R. J. Anderson, "Optical fault induction attacks," in *Cryptographic Hardware and Embedded Systems* (Lecture Notes in Computer Science), vol. 2523. Berlin, Germany: Springer, Aug. 2002, pp. 2–12, doi: 10.1007/3-540-36400-5_2.

[4] J.-M. Schmidt and M. Hutter, "Optical and EM fault-attacks on CRT-based RSA: Concrete results," in *Proc. 15th Austrian Workhop Microelectron. Austrochip*, Graz, Austria: Verlag der Technischen Universität Graz, Oct. 2007, pp. 61–67.

[5] S. Endo, T. Sugawara, N. Homma, T. Aoki, and A. Satoh, "An on-chip glitchy-clock generator for testing fault injection attacks," *J. Cryptograph. Eng.*, vol. 1, no. 4, pp. 265–270, 2011, doi: 10.1007/s13389-011-0022-y.

[6] L. Zussa, J.-M. Dutertre, J. Clediere, and A. Tria, "Power supply glitch induced faults on FPGA: An in-depth analysis of the injection mechanism," in *Proc. IEEE 19th Int. Line Test. Symp. (IOLTS)*, Jul. 2013, pp. 110–115, doi: 10.1109/IOLTS.2013.6604060.

[7] A. D. Dwivedi, "Security analysis of lightweight IoT cipher: Chaskey," *Cryptography*, vol. 4, no. 3, p. 22, Aug. 2020, doi: 10.3390/cryptography4030022.

[8] S. S. Dhanda, B. Singh, and P. Jindal, "Lightweight cryptography: A solution to secure IoT," *Wireless Pers. Commun.*, vol. 112, no. 3, pp. 1947–1980, Jun. 2020, doi: 10.1007/s11277-020-07134-3.

[9] H. Kim, Y. Jeon, G. Kim, J. Kim, B. Sim, D. Han, H. Seo, S. Kim, S. Hong, J. Sung, and D. Hong, "PIPO: A lightweight block cipher with efficient higher-order masking software implementations," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, vol. 12593. Seoul South Korea: Springer, Dec. 2020, pp. 99–122, doi: 10.1007/978-3-030-68890-5_6.

[10] H. Kim, M. Sim, S. Eum, K. Jang, G. Song, H. Kim, H. Kwon, W. Lee, and H. Seo, "Masked implementation of PIPO block cipher on 8-bit AVR microcontrollers," in *Proc. Int. Conf. Inf. Secur. Appl.*, vol. 13009. Jeju Island, South Korea: Springer, Aug. 2021, pp. 171–182, doi: 10.1007/978-3-030-89432-0_14.

[11] Y. Kwak, Y. Kim, and S. C. Seo, "Parallel implementation of PIPO block cipher on 32-bit RISC-V processor," in *Proc. Int. Conf. Inf. Secur. Appl.*, vol. 13009. Jeju Island, South Korea: Springer, Aug. 2021, pp. 183–193, doi: 10.1007/978-3-030-89432-0_15.

[12] J. Kim, S. Kim, S. Kim, D. Hong, J. Sung, and S. Hong, "MILP-aided division property and integral attack on lightweight block cipher PIPO," in *J. Korea Inst. Inf. Secur. Cryptol.*, vol. 31, no. 5, pp. 875–888, 2021, doi: 10.13089/JKIISC.2021.31.5.875.

[13] J. Woo, J. Han, Y. Kim, H. Mun, S. Lim, T. Lee, S. An, S. Kim, and D. Han, "Deep learning-based side-channel analysis on PIPO," in *Proc. Int. Conf. Inf. Secur. Cryptol. (ICISC)*, 2021, pp. 303–316. [Online]. Available: https://link.springer.com/book/9783031088957

[14] J. Blömer and J. Seifert, "Fault based cryptanalysis of the advanced encryption standard (AES)," in *Proc. Int. Conf. Financial Cryptogr.*, vol. 2742. Guadeloupe, French West Indies: Springer, Jan. 2003, pp. 162–181, doi: 10.1007/978-3-540-45126-6_12.

[15] L. Hemme, "A differential fault attack against early rounds of (triple-)DES," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, vol. 3156. Cambridge, MA, USA: Springer, Aug. 2004, pp. 254–267, doi: 10.1007/978-3-540-28632-5_19.

[16] G. Wang and S. Wang, "Differential fault analysis on PRESENT key schedule," in *Proc. Int. Conf. Comput. Intell. Secur.*, Dec. 2010, pp. 362–366, doi: 10.1109/CIS.2010.84.

[17] S. Lim, J. Lee, and D.-G. Han, "Improved differential fault attack on LEA by algebraic representation of modular addition," *IEEE Access*, vol. 8, pp. 212794–212802, 2020, doi: 10.1109/ACCESS.2020.3039805.

[18] S. Sinha and S. Karmakar, "Differential fault analysis of rectangle-80," *IACR Cryptol. ePrint Arch.*, to be published. [Online]. Available: https://eprint.iacr.org/2018/428

[19] B. Lac, A. Canteaut, J. Fournier, and R. Sirdey, "DFA on LS-designs with a practical implementation on SCREAM," in *Proc. Int. Workshop Constructive Side-Channel Anal. Secure Design, (COSADE)*, vol. 10348. Paris, France: Springer, Apr. 2017, pp. 223–247.

[20] S. Lim, J. Han, T. Lee, and D. Han, "Differential fault attack on lightweight block cipher PIPO," in *Proc. Int. Conf. Inf. Secur. Cryptol. (ICISC)*, 2021, pp. 291–302. [Online]. Available: https://eprint.iacr.org/2021/1190

[21] V. Grosso, G. Leurent, F. Standaert, and K. Varici, "Ls-designs: Bitslice encryption for efficient masked software implementations," in *Proc. 21st Int. Workshop*, in Lecture Notes in Computer Science, vol. 8540, C. Cid and C. Rechberger, Eds. London, U.K.: Springer, Mar. 2014 pp. 18–37, doi: 10.1007/978-3-662-46706-0_2.

[22] W. T. Zhang, Z. Z. Bao, D. D. Lin, V. Rijmen, B. Yang, and I. Verbauwhede, "RECTANGLE: A bit-slice lightweight block cipher suitable for multiple platforms," *Sci. China Inf. Sci.*, vol. 58, no. 12, pp. 1–15, Dec. 2015, doi: 10.1007/s11432-015-5459-7.

[23] M. R. Albrecht, B. Driessen, E. B. Kavun, G. Leander, C. Paar, and T. Yalçin, "Block ciphers–focus on the linear layer (feat. PRIDE)," in *Proc. 34th Annu. Cryptol. Conf.*, vol. 8616. Santa Barbara, CA, USA: Springer, Aug. 2014, pp. 57–76, doi: 10.1007/978-3-662-44371-2_4.

[24] B. Lac, M. Beunardeau, A. Canteaut, J. Fournier, and R. Sirdey, "A first DFA on PRIDE: From theory to practice (extended version)," *IACR Cryptol. ePrint Arch.*, to be published. [Online]. Available: http://eprint.iacr.org/2017/075

[25] Riscure. *EM-FI Transient Probe*. Accessed: Jun. 3, 2022. [Online]. Available: https://getquote.riscure.com/en/quote/2101068/em-fi-%transient-probe.htm

[26] Riscure. *Spider*. Accessed: Jun. 3, 2022. [Online]. Available: https://getquote.riscure.com/en/quote/2101116/spider%.htm

[27] Riscure. *Inspector Subscription FI Professional*. Accessed: Jun. 3, 2022. [Online]. Available: https://getquote.riscure.com/en/quote/2101094/inspec%tor-subscription-fi-professional.htm

[28] Riscure. *Pinata Board*. Accessed: Jun. 3, 2022. [Online]. Available: https://www.riscure.com/product/pinata-training-targ%et/

[29] Advanced RISC Machines. *ARM Cortex-M4F Microcontroller*. Accessed: Jun. 3, 2022. [Online]. Available: https://developer.arm.com/ip-products/processors/cor%tex-m/cortex-m4

[30] GNU General Public License. *GNU Arm Embedded Toolchain Version 4.8*. Accessed: Jun. 3, 2022. [Online]. Available: https://launchpad.net/gcc-arm-embedded/4.8

[31] W. He, J. Breier, and S. Bhasin, "Cheap and cheerful: A low-cost digital sensor for detecting laser fault injection attacks," in *Proc. Int. Conf. Secur., Privacy, Appl. Cryptogr. Eng.*, vol. 10076. Hyderabad, India: Springer, Dec. 2016, pp. 27–46, doi: 10.1007/978-3-319-49445-6_2.

[32] J. Choi and Y. Kim, "An improved LEA block encryption algorithm to prevent side-channel attack in the IoT system," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA)*, Dec. 2016, pp. 1–4, doi: 10.1109/APSIPA.2016.7820845.

[33] H. Seo, T. Park, J. Ji, and H. Kim, "Lightweight fault attack resistance in software using intra-instruction redundancy, revisited," in *Proc. Int. Workshop Inf. Secur. Appl.*, vol. 10763. Jeju Island, South Korea: Springer, Aug. 2017, pp. 3–15, doi: 10.1007/978-3-319-93563-8_1.

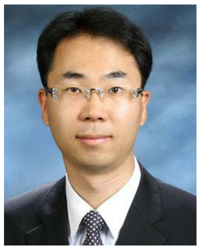
[34] A. Baksi, S. Bhasin, J. Breier, M. Khairallah, T. Peyrin, S. Sarkar, and S. M. Sim, "DEFAULT: Cipher level resistance against differential fault attack," in *Proc. 27th Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, vol. 13091. Singapore: Springer, Dec. 2021, pp. 124–156, doi: 10.1007/978-3-030-92075-3_5.

[35] B. Lac, A. Canteaut, J. J. A. Fournier, and R. Sirdey, "Thwarting fault attacks against lightweight cryptography using SIMD instructions," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5, doi: 10.1109/ISCAS.2018.8351693.

[36] J. Lee and D. Han, "Realistic multiple fault injection system based on heterogeneous fault sources," *J. Korea Inst. Inf. Secur. Cryptol.*, vol. 30, no. 6, pp. 1247–1254, 2020. [Online]. Available: https://www.koreascience.or.kr/article/JAKO202006763002870.kr&sa=U

**SEONGHYUCK LIM** received the M.S. degree in financial information security from Kookmin University, Seoul, Republic of Korea, in 2022, where he is currently pursuing the Ph.D. degree in financial information security. His research interests include symmetric key cryptography, AI-based side-channel analysis, fault injection attacks, and security of financial IC cards.

**JAESEUNG HAN** received the M.S. degree in financial information security from Kookmin University, Seoul, Republic of Korea, in 2022, where he is currently pursuing the Ph.D. degree in financial information security. His research interests include side-channel attacks, symmetric key cryptography, AI-based side-channel analysis, and lattice-based cryptography.



**DONG-GUK HAN** received the B.S. and M.S. degrees in mathematics and the Ph.D. degree in engineering in information security from Korea University, Seoul, Republic of Korea, in 1999, 2002, and 2005, respectively. He was a Postdoctoral Researcher at Future University Hakodate, Hokkaido, Japan. After finishing his doctoral course, he was then an Exchange Student with the Department of Computer Science and Communication Engineering, Kyushu University, Japan, from April 2004 to March 2005. From 2006 to 2009, he was a Senior Researcher at the Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea. He is currently working as a Professor with the Department of Information Security, Cryptology, Mathematics, Kookmin University, Seoul. He is a member of KIISC, IEEK, and IACR.