Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

BORON: an ultra-lightweight and low power encryption design for pervasive computing

Gaurav BANSOD†1, Narayan PISHAROTY2, Abhijit PATIL3

(*1Pune Institute of Computer Technology, Pune 411043, India*)

(*2Glocal University, Saharanpur 247121, India*)

(*3Symbiosis Institute of Technology, Pune 412115, India*)

†E-mail: gaurav249@gmail.com

Received Nov. 28, 2015; Revision accepted May 15, 2016; Crosschecked Feb. 28, 2017

Abstract: We propose an ultra-lightweight, compact, and low power block cipher BORON. BORON is a substitution and permutation based network, which operates on a 64-bit plain text and supports a key length of 128/80 bits. BORON has a compact structure which requires 1939 gate equivalents (GEs) for a 128-bit key and 1626 GEs for an 80-bit key. The BORON cipher includes shift operators, round permutation layers, and XOR operations. Its unique design helps generate a large number of active S-boxes in fewer rounds, which thwarts the linear and differential attacks on the cipher. BORON shows good performance on both hardware and software platforms. BORON consumes less power as compared to the lightweight cipher LED and it has a higher throughput as compared to other existing SP network ciphers. We also present the security analysis of BORON and its performance as an ultra-lightweight compact cipher. BORON is a well-suited cipher design for applications where both a small footprint area and low power dissipation play a crucial role.

Key words: Lightweight cryptography; SP network; Block cipher; Internet of Things (IoT); Encryption; Embedded security
http://dx.doi.org /10.1631/FITEE.1500415 **CLC number:** TP309.7

1 Introduction

Lightweight cryptography is an emerging field, which is well suited for applications like Internet of Things (IoT), pervasive computing, and embedded security. In recent years, many lightweight ciphers have been designed and implemented for providing security in applications like wireless sensor nodes and RFID tags. The constraints about these applications are their footprint areas, number of gate equivalents (GEs), and power consumption. The cipher, which meets all these constraints, can be implemented for providing security in the field of embedded security. A lightweight cipher should need less than 2000 GEs for its implementation in hardware. It should also consume less flash memory to be more compact. Ciphers, like PRESENT (Bogdanov *et al.*, 2007), TWINE (Suzaki *et al.*, 2011), PICCOLO (Shibutani *et al.*, 2011), SIMON and SPECK (Beaulieu *et al.*, 2013), RECTANGLE (Zhang *et al.*, 2014), and LED (Guo *et al.*, 2011), are the existing ultra-lightweight ciphers, among which the cipher PRESENT is the most compact and has the best performance on both hardware and software platforms. The PRESENT cipher is designed to provide a compact hardware structure. The cipher LED (Guo *et al.*, 2011) is known for its robust architecture, as it inherits some elements of the AES (NIST, 2001) cipher, but it has a high energy-per-bit, which results in high power dissipation. The fault-based attack is mounted on the LED cipher. Recently, NSA launched the most compact ciphers, SIMON and SPECK, which should be studied further for all possible types of attacks. In this paper, we propose a design (BORON) that requires

ORCID: Gaurav BANSOD, http://orcid.org/0000-0002-4089-9714

less footprint area and less power consumption and has good cryptographic properties. The proposed BORON cipher, as an SP network (Menezes *et al.*, 1996) cipher, executes at a higher throughput as compared to other Feistel based ciphers. Multi-round permutations are used to build the strong cryptographic properties in BORON. While designing a complex structure, we also take into consideration the memory size and the number of GEs required for the implementation of the cipher.

The nonlinear layer in an SP network plays a very important role in deciding the strength of a cipher. In BORON, we have designed a strong nonlinear layer suitable for BORON, which results in a larger number of active S-boxes. BORON shows a good resistance against linear and differential cryptanalysis. Our experimentation also shows that BORON's permutation layer design is one of the most robust designs among other existing lightweight ciphers. We have used a computer-based approach to find the linear and differential trails, the minimum number of active S-boxes, and to mount linear and differential attacks on this cipher. Key scheduling of BORON is motivated by the PRESENT cipher. Overall, the BORON cipher shows good cryptographic properties, has a robust design, a compact structure, and has a very low power consumption.

For the BORON cipher, we use the following notations:

A_j	→	input plaintext block of j=64 bits
C_j	→	output cipher text block of j=64 bits
K_i	→	64-bit round sub-key for round i
$\oplus$	→	bitwise exclusive-OR operation
$<<<n$	→	left cyclic shift by n bits
$<<n$	→	left shift by n bits
$>>n$	→	right shift by n bits
RC_j^i	→	round counter i of j=5 bits
$\|\|$	→	concatenation of two strings
!	→	bitwise NOT operation
&	→	bitwise AND operation
$\|$	→	bitwise OR operation

2 Block cipher BORON

BORON is a substitution permutation network (Menezes *et al.*, 1996) and has a total of 25 rounds. The block length is 64 bits and supports 80- and 128-bit key lengths. Fig. 1 shows the block diagram of a BORON cipher and Fig. 2 shows the detailed BORON block cipher.

As shown in Fig. 1, the BORON cipher contains an S-box, which acts as a nonlinear layer followed by a block shuffle of four bits. The shuffled bits are fed to the round permutation layer, which is then followed by an XOR operation. Twenty-five different keys are generated from the 80/128-bit key register and these keys are applied in each round of the BORON cipher. One extra key will be generated and will be XOR-ed to produce the final ciphertext.

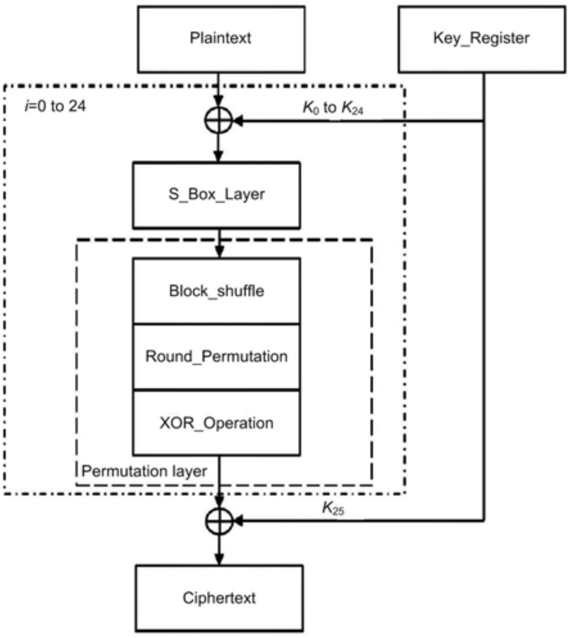

Fig. 1 Block diagram of a BORON cipher

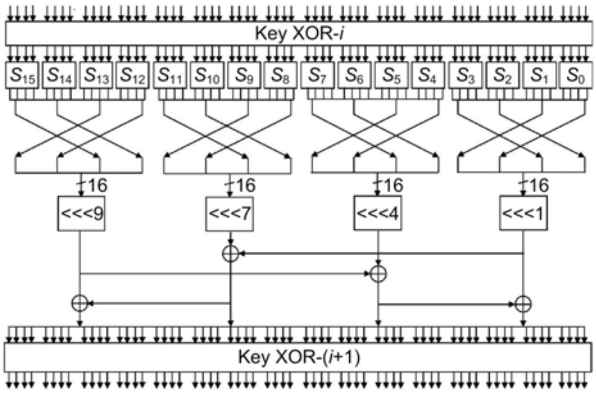
Key XOR-i
S_{15} S_{14} S_{13} S_{12} S_{11} S_{10} S_9 S_8 S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0
16
16
16
16
<<<9
<<<7
<<<4
<<<1
Key XOR-(i+1)

Fig. 2 Block cipher BORON

Pseudo code of the BORON cipher is given as

```
$A=a^{63}\ a^{62}\ ...\ a^0$
RoundKeys()
**for** $i$=0 to 24 **do**
    Add_round_key($A$, $K_i$)
    S_Box_Layer($A$)
    Block_Shuffle($A$)
    Round_Permutation($A$)
    XOR_Operation($A$)
**end for**
Add_round_key($A$, $K_{25}$)
```

Each round of the BORON cipher consists of the operations described in the following sections.

2.1 Add_round_key

Add_round_key performs an XOR ('⊕') operation with a 64-bit plaintext and a 64-bit sub-key which is extracted from the 128-bit key register. Sub-keys are denoted by K_i (i ranges from 0 to 24) and the current state output $A \rightarrow a^{63}\ a^{62}\ \ldots\ a^0$ is given as

$$A \rightarrow A \oplus K^i.$$

2.2 S_Box_Layer

The S-box used in the BORON cipher design is a 4-bit to 4-bit S-box, S: $F_2^4 \rightarrow F_2^4$. Table 1 presents the hexadecimal values for the substitution layer.

Table 1 S-box of the BORON cipher

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S[x]$	e	4	b	1	7	9	c	a	d	2	0	f	8	5	3	6

The current state output $A_{64} \rightarrow a^{63}\ a^{62}\ ...\ a^0$ is divided into sixteen 4-bit blocks $W^{15}\ W^{14}\ \ldots\ W^0$. Each W^i is of a 4-bit size and $W^i=a^{4i+3}||a^{4i+2}||a^{4i+1}||a^{4i}$, where i ranges from 0 to 15 and i is updated by $i=i+4$. W^i is fed to the 4-bit S-box and the output nibble $S[W^i]$ provides the updated values.

The 4-bit to 4-bit S-box of BORON is described by the following equations.

Let $X=x_3x_2x_1x_0$ be the input of the S-box and $Y=y_3y_2y_1y_0$ be the output. Then,

$$y_0 = (!x_3\&!x_2\&x_1) \mid (!x_2\&x_1\&x_0) \mid (!x_3\&x_2\&!x_1) \mid (x_2\&!x_1\&x_0) \mid (x_3\&!x_2\&!x_1\&!x_0) \mid (x_3\&x_2\&x_1\&!x_0),$$

$$y_1=(!x_3\&!x_2\&!x_0) \mid (!x_3\&!x_1\&!x_0) \mid (x_2\&x_1\&x_0) \mid (x_3\&!x_2\&x_0) \mid (x_3\&x_2\&x_1),$$

$$y_2=(!x_3\&!x_2\&!x_1) \mid (!x_2\&!x_1\&!x_0) \mid (!x_3\&x_2\&!x_0) \mid (x_3\&x_1\&x_0) \mid (x_3\&x_2\&x_0),$$

$$y_3=(!x_3\&!x_2\&!x_0) \mid (!x_3\&x_1\&!x_0) \mid (!x_3\&x_2\&x_0) \mid (x_3\&!x_1\&!x_0) \mid (x_3\&!x_2\&x_1\&x_0).$$

2.3 Permutaion_Layer

The Permutation_Layer of the BORON cipher has three sub-permutation layers, which are described in the following.

2.3.1 Block_Shuffle

The Block_Shuffle layer takes the 16-bit input and gives the 16-bit shuffled output. The block permutation is presented in Table 2. Block x of a 4-bit size is substituted to the block position $P[x]$ of a 4-bit size. The Block_Shuffle P: $\{0, 1\}_{16} \leftarrow \{0, 1\}_{16}$ divides a 16-bit input into four 4-bit data as $P=P_4^3||P_4^2||P_4^1||P_4^0$ and then permutes them following the manner shown in Fig. 3.

Table 2 Block_Shuffle layer of the BORON cipher

x	0	1	2	3
$P[x]$	2	3	0	1

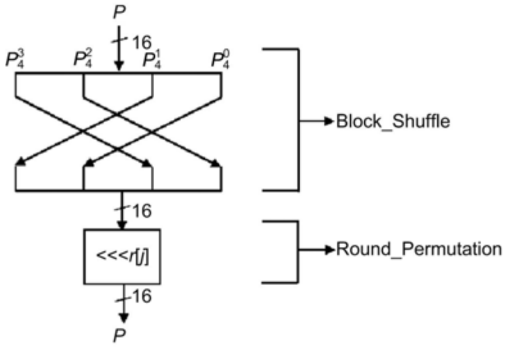

Fig. 3 Block_Shuffle and Round_Permutaion layers

For the 64-bit block size, the Block_Shuffle operation is repeated four times for each 16-bit size block (W^0, W^1, W^2, W^3), where $W^j=a^{i+15}\ ||\ a^{i+14}\ ||\ \ldots\ ||\ a^{i+1}\ ||\ a^i$ for $0 \le 16 \cdot i \le 48$ and j ranges from 0 to 3. W^j is updated in the following manner:

$$W^0=a^{15}\,\|a^{14}\|\ldots\|a^1\|a^0,$$
$$W^1=a^{31}\,\|a^{30}\|\ldots\|a^{17}\|a^{16},$$
$$W^2=a^{47}\,\|a^{46}\|\ldots\|a^{33}\|a^{32},$$
$$W^3=a^{63}\,\|a^{30}\|\ldots\|a^{47}\|a^{48}.$$

2.3.2 Round_Permutation

The Round_Permutation performs the left circular shift operation on the 16-bit block as shown in Fig. 3 and is given as

$$W^j=W^j<<<r[j],\ \ 0\le j\le 3.$$

Table 3 shows the left circular shifted values for each 16-bit block computed from the 64-bit block.

Table 3 Round_Permutation: left circular shift values

j	0	1	2	3
$r[j]$	1	4	7	9

2.3.3 XOR_Operation

The XOR_Operation performs XOR operation (⊕) between 16-bit inputs and produces 16-bit output. This layer produces a 64-bit output in the following way:

$$A_{64}\rightarrow(W^3\oplus W^2\oplus W^0)\,\|\,(W^2\oplus W^0)\,\|\,(W^3\oplus W^1)$$
$$\|\,(W^3\oplus W^1\oplus W^0).$$

Algorithm 1 summaries the encryption process described in Sections 2.1–2.3.

2.4 Key schedule of 80- and 128-bit key lengths

The key schedule of the BORON cipher is motivated by the PRESENT (Bogdanov *et al.*, 2007) cipher key scheduling design—no attack until dates are reported on the PRESENT cipher key scheduling. In BORON cipher key scheduling, there are a total of 25 sub-keys (each of size 64-bit). We strengthen the key scheduling by using a larger number of nonlinear operations in key scheduling.

1. 128-bit key scheduling

The user-defined 128-bit key is stored in the register key, and the 64-bit least significant bits (LSBs) from the KEY register are extracted as follows:

$$K^i=K_{63}\,K_{62}\ldots K_0,$$
$$\text{KEY}=K_{127}\,K_{126}\ldots K_0.$$

Algorithm 1 Encryption

```
Input: Plaintext $A_{64}\rightarrow a^{63}\,a^{62}\ldots a^0$, $S[16]$, $P[4]$, $r[4]$
Output: Ciphertext $C_{64}$
for $i$=0 to 24 do
    $A_{64}\rightarrow a^{63}\,a^{62}\ldots a^0$
    for $j$=0 to 3 do
        $\text{temp}_{16}\rightarrow(A_{64}>>16\cdot j)\oplus(K^i_{64}>>16\cdot j)$
        $\text{temp}_{16}\rightarrow S[\text{temp}_{16}]$   // S-box
        $W^{[j]}_{16}=0$
        for $k$=0 to 4 do
            $W^{[j]}_{16}$ +=$((\text{temp}_{16}>>4k)\&0\text{xF})<<4P[k]$
                                          // Block_shuffle
        end for
        $W^{[j]}_{16}=W^{[j]}_{16}<<<r[j]$   // Round_Permutation
    end for
    $A_{64}\rightarrow X(W^3_{16},W^2_{16},W^0_{16})\,\|\,X(W^2_{16},W^0_{16})\,\|\,X(W^3_{16},W^1_{16})$
        $\|\,X(W^3_{16},W^1_{16},W^0_{16})$   // $X\rightarrow$XOR_Operation
end for
for $j$=0 to 3 do
    $C_{64}$+=$(A_{64}>>16\cdot j)\oplus(K^{25}_{64}>>16\cdot j)<<16\cdot j$
end for
```

After extracting the keys of 64 bits, the register KEY is updated in the following manner:

(1) KEY<<<13;
(2) $[K_3\,K_2\,K_1\,K_0]\leftarrow S[K_3\,K_2\,K_1\,K_0]$;
(3) $[K_7\,K_6\,K_5\,K_4]\leftarrow S[K_7\,K_6\,K_5\,K_4]$;
(4) $[K_{63}\,K_{62}\,K_{61}\,K_{60}\,K_{59}]\leftarrow[K_{63}\,K_{62}\,K_{61}\,K_{60}\,K_{59}]\oplus\text{RC}^i$.

For 0 to 24 rounds, five bits of the round counter i is XOR-ed with the five bits of key register KEY, i.e., from K_{59} to K_{63}.

2. 80-bit key scheduling

The user-defined 80-bit key is stored in the key register KEY and the LSB bits from it are used as round sub-keys:

$$K_i=K_{63}\,K_{62}\ldots K_0,$$
$$\text{KEY}=K_{79}\,K_{78}\ldots K_0.$$

After extracting the 64-bit key, register KEY is updated as follows:

(1) KEY<<<13;
(2) $[K_3\,K_2\,K_1\,K_0]\leftarrow S[K_3\,K_2\,K_1\,K_0]$;
(3) $[K_{63}\,K_{62}\,K_{61}\,K_{60}\,K_{59}]\leftarrow[K_{63}\,K_{62}\,K_{61}\,K_{60}\,K_{59}]\oplus\text{RC}^i$.

3 Security analysis of BORON

There are different cryptanalysis techniques to find whether the cipher is resistant to attacks or not. In this study, we focus on basic attacks, like differential attack, linear attack, algebraic attack, key scheduled attack, and key collision attack. In an SP network, S-box selection plays a very important role in deciding whether the structure is secure against linear and differential attacks. S-box is the only nonlinear element in the entire cipher design. Generation of a large number of active S-boxes in a design results in a robust architecture and can thwart all possible types of attacks. Computer-based techniques are used in this study for the selection of good S-boxes and to find the minimum number of active S-boxes.

3.1 Design criteria of the S-box

Use of a different S-box for each round will result in an increase in the gate count. Similarly, the use of different S-boxes does not provide a sensible improvement in the resistance against known attacks (NIST, 2001). The use of an 8-bit to 8-bit S-box in the cipher will increase the number of GEs, so we choose a single 4-bit to 4-bit S-box for the BORON cipher.

Compactness and resistance against linear and differential attacks are the two parameters we consider while designing the S-box. A compact S-box requires less memory and a smaller number of GEs for implementation.

The BORON S-box takes a 4-bit input and produces a 4-bit output, i.e., $S: F_2^4 \to F_2^4$. Properties essential for a good S-box design are listed as follows:

Property 1 (Linear property) Let $a \in F_2^4$ be the input to the S-box, A the input mask, B the output mask, and $A, B \in F_2^4$. LC(A, B) is defined as

$$\mathrm{LC}(A, B)=\#\{a\in F_2^4 \,|\, A\bullet a=B\bullet S(a)\}-8,$$

where LC(·) represents linear cryptanalysis (Matsui 1993; Heys 2001), '•' denotes mask operation on F_2^4, and #{·} indicates the number of matches in the linear approximation table (LAT) for input mask A and output mask B.

Property 2 (Differential property) Let $a \in F_2^4$ be the input to the S-box, ΔA and ΔB the input and output differences, respectively, and $\Delta A, \Delta B \in F_2^4$. DC($\Delta A$, ΔB) is defined as

$$\mathrm{DC}(\Delta A, \Delta B)=\#\{a\in F_2^4 \,|\, S(a)\oplus S(a\oplus\Delta A)=\Delta B\},$$

where DC(·) represents differential cryptanalysis (Biham and Shamir, 1991; Heys, 2001). This property is used to form the difference distribution table (DDT).

The complete design criteria of the S-box, which we have used in the design of the BORON cipher, are given as follows:

Criterion 1 For any nonzero input difference $\Delta A \in F_2^4$ and output differences $\Delta B \in F_2^4$, we have

$$\mathrm{DC}(\Delta A, \Delta B)=\#\{a\in F_2^4 \,|\, S(a)\oplus S(a\oplus\Delta A)=\Delta B\}\le 4.$$

Criterion 2 For any nonzero input differences $\Delta A \in F_2^4$ and output differences $\Delta B \in F_2^4$ such that Hw(ΔA)=Hw(ΔB)=1, where Hw(x) denotes the Hamming weight of x, we have

$$\begin{aligned}\mathrm{Set_{DC}}&=\mathrm{DC}(\Delta A, \Delta B)\\ &=\#\{a\in F_2^4 \,|\, S(a)\oplus S(a\oplus\Delta A)=\Delta B\}=0.\end{aligned}$$

Cardinality of $\mathrm{Set_{DC}}$ can be given as $\mathrm{Car_{DC}}$, and we have $\mathrm{Car_{DC}}$=0.

Criterion 3 For any nonzero input mask $A \in F_2^4$ and output mask such that $B \in F_2^4$, we have

$$\mathrm{LC}(A, B)=\#\{a\in F_2^4 \,|\, A\bullet a=B\bullet S(a)\}-8\le 4.$$

Criterion 4 For any nonzero input mask $A \in F_2^4$, output mask $B \in F_2^4$, and Hw(A)=Hw(B)=1, we have

$$\mathrm{Set_{LC}}=\mathrm{LC}(A, B)=\#\{x\in F_2^4 \,|\, A\bullet x=B\bullet S(x)\}-8\ne 0.$$

Cardinality of $\mathrm{Set_{LC}}$ can be given as $\mathrm{Car_{LC}}$, and we have $\mathrm{Car_{LC}}$=4.

Criterion 5 (Bijective) $S(a)\ne S(b)$ for all values of $a\ne b$.

Criterion 6 (No static point) $S(a)\ne a$ for all values of $a \in F_2^4$.

The strength of the S-box depends on cardinality; for example, in the case of the PRESENT cipher, S-box (Bogdanov *et al.*, 2007; Zhang *et al.*, 2014) has $\mathrm{Car_{DC}}$=0 and $\mathrm{Car_{LC}}$=8; in case of the RECTANGLE cipher, S-box (Zhang *et al.*, 2014) has Car_{DC}=2 and Car_{LC}=2; in case of the BORON cipher, S-box has Car_{DC}=0 and Car_{LC}=4.

3.2 BORON cipher S-box selection

Selection of the S-box in the BORON cipher is driven by two definitions:

Definition 1 (Permutation-then-XOR equivalence (PE) (Leander and Poschmann, 2007; Zhang *et al.*, 2014)) If there exist two 4×4 permutation matrices $\boldsymbol{m}_0$ and $\boldsymbol{m}_1$ and constants a, $b\in F_2^4$ for two S-boxes such that $S'(x)=\boldsymbol{m}_1(S(\boldsymbol{m}_0(x)+a))+b$, then the equivalence is called PE.

When an S-box satisfies criteria 1 to 5 in Section 3.1, then its PE equivalent S-box also satisfies criteria 1 to 5.

Definition 2 (Affine equivalence (Leander and Poschmann, 2007; Zhang *et al.*, 2014)) If there exist a bijective linear mapping $\boldsymbol{A}$, $\boldsymbol{B}$ and constants a, $b \in F_2^4$ for two S-boxes such that $S'(x)=\boldsymbol{B}(S(\boldsymbol{A}(x)+a))+b$, then the equivalence is called affine equivalence.

When an S-box satisfies criteria 1, 3, and 5 in Section 3.1, then its affine equivalent S-box also satisfies criteria 1, 3, and 5.

Definitions 1 and 2 are considered while designing the S-box for the BORON cipher.

3.3 Linear cryptanalysis

Linear cryptanalysis (Matsui, 1993; Heys, 2001) is applicable to symmetric-key block ciphers and the cipher needs to resist such a kind of attack. This attack is a known plaintext attack. It uses the high probability occurrences of the linear expression containing plaintext bits, cipher text bits, and sub-key bits. This expression is used for mounting linear attacks on a cipher. To mount a linear attack, the attacker needs to have the knowledge about a subset of plaintext and its corresponding ciphertext. The attacker will find the relationship between them. S-box is examined by forming a LAT. S-box is the only nonlinear component in the encryption design. If P_{L} is the linear probability, then the bias can be given as $|P_{\text{L}}-1/2|$, and bias (ε) for the BORON cipher S-box is 2^{-2}. Matsui's piling-up lemma (Matsui, 1993) is used to calculate the probability bias for n rounds.

The best way to resist against linear cryptanalysis is:

1. Optimize the bias in LAT. For an ideal S-box, the value of bias is 1/8 and this is practically not possible to achieve.

2. Increase the number of active S-boxes in the cipher structure.

Lemma 1 (Matsui's pilling-up lemma (Matsui, 1993)) For random binary variables X_1, X_2, …, X_n (n is the number of active S-boxes) and $X_1\oplus X_2\oplus\cdots\oplus X_n=0$, the total bias can be derived as

$$\varepsilon = 2^{n-1}\prod_{i=1}^{n}\varepsilon_i,$$

where ε_i represents the bias of X_i.

Two standard methods can be used to count the minimum number of active S-boxes:

1. Matusi's branch and bound algorithm (Matsui, 1994);

2. the mixed-integer programming technique (Sun *et al.*, 2014a; 2014b).

In this study, Matusi's branch and bound algorithm is used to count the minimum number of active S-boxes.

Table 4 presents the linear trails for the BORON cipher. Fig. 4 shows the linear trails for two rounds of the BORON cipher and the red S-box represents the active S-box. The maximum bias is 2^{-2} for the BORON cipher S-box. Table 5 presents the minimum number of active S-boxes from the linear trails.

Table 4 Linear trails for the BORON cipher

Round index	Input to S-box	Output of S-box
1	0000 0000 0008 0000	0000 0000 0005 0000
2	0000 0000 5000 5000	0000 0000 3000 3000
3	0060 0060 0300 0360	0010 0010 0600 0610
4	2024 2004 0040 204c	e0e9 e009 0090 e095

Table 5 Minimum number of active S-boxes from the linear trails

Round index	Minimum number of active S-boxes
1	1
2	3
3	8
4	17

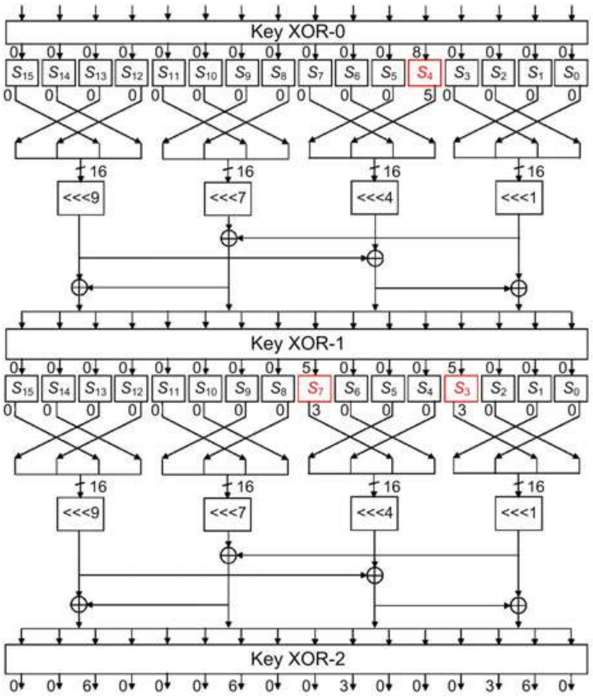
Key XOR-0
0 0 0 0 0 0 0 0 0 0 0 8 0 0 0 0
S_{15} S_{14} S_{13} S_{12} S_{11} S_{10} S_9 S_8 S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0
0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0
16
<<<9
<<<7
<<<4
<<<1
Key XOR-1
0 0 0 0 0 0 0 0 5 0 0 0 5 0 0 0
S_{15} S_{14} S_{13} S_{12} S_{11} S_{10} S_9 S_8 S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0
0 0 0 0 0 0 0 0 3 0 0 0 3 0 0 0
Key XOR-2
0 0 6 0 0 0 6 0 0 3 0 0 0 3 6 0

Fig. 4 Linear trails for two rounds of BORON cipher (References to color refer to the online version of this figure)

Theorem 1 For 18 rounds of BORON it has a total of 48 active S-boxes and the total bias for the 18 rounds is 2^{-49}.

Proof We find that for three rounds BORON has a minimum of eight active S-boxes.

The maximum bias for the BORON cipher S-box is 2^{-2} by using Matsui's pilling-up lemma for three rounds of BORON cipher and the total bias can be given as

$$2^7\times(2^{-2})^8=2^{-9}.$$

By applying the same lemma for 18 rounds, the total bias ε can be given as

$$\varepsilon=2^5\times(2^{-9})^6=2^{-49}.$$

By calculating the required number of known plaintext/ciphertext, we can compute the complexity of the linear attack as

$$N_{\mathrm{L}}=1/\varepsilon^2.$$

For 18 rounds of BORON cipher, the required number of known plaintext/ciphertext can be given as

$$N_{\mathrm{L}}=1/\varepsilon^2=1/(2^{-49})^2=2^{98}.$$

The required number of known plaintext/ciphertexts is 2^{98}, which is larger than the available limit, i.e., 2^{64}. Thus, the complete number of rounds of BORON cipher shows good resistance against a linear attack.

3.4 Differential cryptanalysis

Differential cryptanalysis (Biham and Shamir, 1991; Heys, 2001) is the most significant attack applicable to the symmetric key block cipher. Biham and Shamir (1991) first applied the differential attack on the Data Encryption Standard (DES). To mount the differential attack for a specific number of rounds in an encryption system, pairs of high probability input and output occurrences are used to recover the round keys. S-box is a nonlinear component in our design and it gets examined by forming a DDT. Differential trails are formed by considering high probability input and output differences for each round, and the S-box that has non-zero input differences or non-zero output differences is referred to as an active S-box.

The differential probability for the BORON cipher S-box is $4/16=2^{-2}$.

There are two approaches to providing security against differential cryptanalysis:

1. By minimizing the differential probability, for the ideal S-box, this probability is 1/16.

2. Find a structure that maximizes the minimum number of active S-boxes.

Table 6 presents the differential trails for the BORON cipher. Non-zero input differences to the S-box or non-zero output differences from the S-box are referred to as an active S-box. Table 7 presents the minimum number of active S-boxes from differential trails.

Table 6 Differential trails for the BORON cipher

Round index	Input to S-box	Output of S-box
1	0000 0000 000e 0000	0000 0000 0002 0000
2	0000 0000 2000 2000	0000 0000 3000 3000
3	0060 0060 0300 0360	0080 0080 0200 0280
4	0145 0045 0120 0125	0a91 0091 0a30 0a31

For three rounds of BORON cipher, there are a minimum of eight active S-boxes, so for 18 rounds there will be a minimum of 48 active S-boxes. The total differential probability P_{d} is $(2^{-2})^{48}=2^{-96}$.

Table 7 Minimum number of active S-boxes from the differential trails

Round index	Minimum number of active S-boxes
1	1
2	3
3	8
4	18

We can compute the complexity of the differential attack by calculating the required number of plaintext/ciphertexts chosen and this can be given as

$$N_{\mathrm{d}}=C/P_{\mathrm{d}},$$

where C=1 and $P_{\mathrm{d}}=2^{-96}$, so the required number of plaintext/ciphertexts chosen is

$$N_{\mathrm{d}}=1/2^{-96}=2^{96}.$$

The required number of chosen plaintext/ciphertexts is 2^{96}, which is larger than the available limit, i.e., 2^{64}. Therefore, the complete number of rounds of the BORON cipher shows good resistance against differential attacks.

3.5 Zero-correlation attack

Zero-correlation attack (Bogdanov and Rijmen, 2011; Soleimany and Nyberg, 2012) is the extension of linear cryptanalysis. The block ciphers should resist a zero correlation attack. The zero-correlation attack is based on the linear approximations with a correlation value of zero. The zero-correlation attack is considered as a counterpart of the impossible differential cryptanalysis in the domain of linear cryptanalysis. We have applied a matrix method (Soleimany and Nyberg, 2012) to mount a zero-correlation attack, which is explained below. Following are the three lemmas used to mount a zero-coorelation attack on the cipher:

Lemma 2 (XOR approximation) Either the three linear selection patterns at an XOR '⊕' are equal or the correlation over the ⊕ operator is exactly zero.

Lemma 3 (Branching approximation) Either the three linear selection patterns at a branching point '•' sum up to 0 or the correlation over '•' is exactly zero.

Lemma 4 (Permutation approximation) Over a permutation ϕ, if the input and output selection patterns are neither both zero nor both non-zero, the correlation over ϕ is exactly zero.

1. The matrix method (Soleimany and Nyberg, 2012)

The miss-in-the-middle approach is considered to find the impossible differential characteristics of a cipher. This approach is used to construct the impossible differential characteristic by two (truncated) differential paths with a probability of one, which leads to a contradiction in the middle. The matrix method for finding the linear approximation with correlation zero is given below:

The linear masks applied to the words can be one of the following five types:

(1) zero mask denoted by 0,

(2) an arbitrary non-zero mask denoted by 0,

(3) non-zero mask with a fixed value $\overline{a}$,

(4) the exclusive-or of a fixed non-zero mask a and an arbitrary non-zero mask, denoted by $\overline{a}$, and

(5) any other mask denoted by '∗'.

The matrix shows how a linear mask of each output word is affected by the linear mask of an input word. Arithmetic rules for multiplication and addition are given in Tables 8 and 9, respectively.

Table 8 Arithmetic rules multiplication by 0, 1, and 1_{F}

	0	1	1_{F}
0	0	0	0
$\overline{0}$	0	$\overline{0}$	$\overline{0}$
a	0	a	
$\overline{a}$	0	$\overline{a}$	∗
∗	0	∗	∗

Table 9 Arithmetic rules for addition between two masks

	0	$\overline{0}$	a	$\overline{a}$	∗
0	0	0	a	$\overline{a}$	∗
$\overline{0}$	0	∗	a	∗	∗
b	b	$\overline{b}$	$a+b$	∗	∗
$\overline{b}$	$\overline{b}$	∗	∗	∗	∗
∗	∗	∗	∗	∗	∗

2. Zero-correlation for four rounds of BORON (000a000000000000)→(00000000000000b0) has a correlation of exactly zero for the values a and b that are non-zero. The trails for zero correlation attacks are shown in Table 10 and we find contradictions at

round 2 for the BORON cipher. Contradictions are presented in Table 10 by using characters in bold.

Table 10 Trails for zero-correlation for the BORON cipher

Round index	Trails
0	0000 0000 0000 *aaaa* 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
1	$000\ 0000\ 000\bar{0}\ \bar{0}\bar{0}\bar{0}0$ $0000\ 0000\ 000\bar{0}\ \bar{0}\bar{0}\bar{0}0$ 0000 0000 0000 0000 0000 0000 0000 0000
2	$0000\ 000\bar{0}\ \bar{0}\bar{0}\bar{0}\bar{0}\ \bar{0}\bar{0}\bar{0}0$ $\bar{0}\mathbf{000}\ \mathbf{000}\bar{0}\ \bar{0}{*}{*}{*}\ {*}{*}{*}0$ $\bar{\mathbf{0}}000\ 0000\ 0\bar{0}\bar{0}\bar{0}\ \bar{0}\bar{0}\bar{0}\bar{0}$ $\bar{0}\mathbf{000}\ \mathbf{0000}\ \mathbf{0}\bar{0}\bar{0}\bar{0}\ \bar{0}\bar{0}\bar{0}\bar{0}$
2	${*}{*}{*}{*}\ {*}{*}{*}{*}\ \bar{0}\bar{0}\bar{0}\bar{0}\ \bar{0}\bar{0}\bar{0}\bar{0}$ $\bar{0}\bar{\mathbf{0}}\bar{\mathbf{0}}\bar{\mathbf{0}}\ \bar{\mathbf{0}}\bar{\mathbf{0}}\bar{\mathbf{0}}\bar{0}\ \bar{0}\bar{0}\bar{0}\bar{0}\ 0000$ $\mathbf{0}000\ 0000\ \bar{0}\bar{0}\bar{0}\bar{0}\ {*}{*}{*}{*}$ $\bar{0}\bar{\mathbf{0}}\bar{\mathbf{0}}\bar{\mathbf{0}}\ \bar{\mathbf{0}}\bar{\mathbf{0}}\bar{\mathbf{0}}\bar{\mathbf{0}}\ \bar{\mathbf{0}}\bar{0}\bar{0}\bar{0}\ \bar{0}\bar{0}\bar{0}\bar{0}$

3	0000 0000 0000 0000
	0000 0000 0000 0000
	0000 0000 0000 0000
	0000 0000 0000 0000

4	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 *bbbb* 0000

Bold characters indicate the contradictions

3.6 Biclique attack

Biclique attack (Bogdanov *et al.*, 2011; Jeong *et al.*, 2012) is an extension of the meet-in-the-middle attack. In this study, we apply biclique cryptanalysis on both BORON-80 and BORON-128. Based on the attack results, comparisons are made with the most popular lightweight block ciphers such as PRESENT, Piccolo, and LED.

We have constructed a three-dimensional biclique for rounds 22–25 of BORON-80. For these rounds the partial keys used are (K^{22}, K^{23}, K^{24}, K^{25}), described as follows:

$$K^{22}=K_{17}, K_{16}, \ldots, K_0, K_{79}, \ldots, K_{34}$$
$$K^{23}=K_4, K_3, \ldots, K_0, K_{79}, \ldots, K_{21}$$
$$K^{24}=K_{71}, K_{70}, \ldots, K_8$$
$$K^{25}= K_{58}, K_{57}, \ldots, K_0, K_{79}, \ldots, K_{75}$$

From the above equations we find that varying the following sub-keys (K^{23}, K^{22}, K^{21}) and (K^{74}, K^{73}, K^{72}) gives the bicliques on the full BORON-80.

To construct the Δi-differential, we consider sub-keys (K^{23}, K^{22}, K^{21}) and for the ∇j-differential, we consider sub-keys (K^{74}, K^{73}, K^{72}). Let f be a sub-cipher from rounds 22 to 25. The Δi-differential affects 41 bits of the ciphertext (Fig. 5), and the data complexity does not exceed 2^{41}. The red arrows at the 25th round in Fig. 5 show the data complexity.

Figs. 6a and 6b present the recomputation in the forward and backward directions, respectively. The total computational complexity of BORON-80 is computed as follows:

$$C_{\text{total}}=2^{k-2d}(C_{\text{biclique}}+C_{\text{precomp}}+C_{\text{recomp}}+C_{\text{falsepos}}),$$
$$=2^{80-6}(2^{0.954}+2^{2.88}+2^{5.239}+2^{2})=2^{79.564}.$$

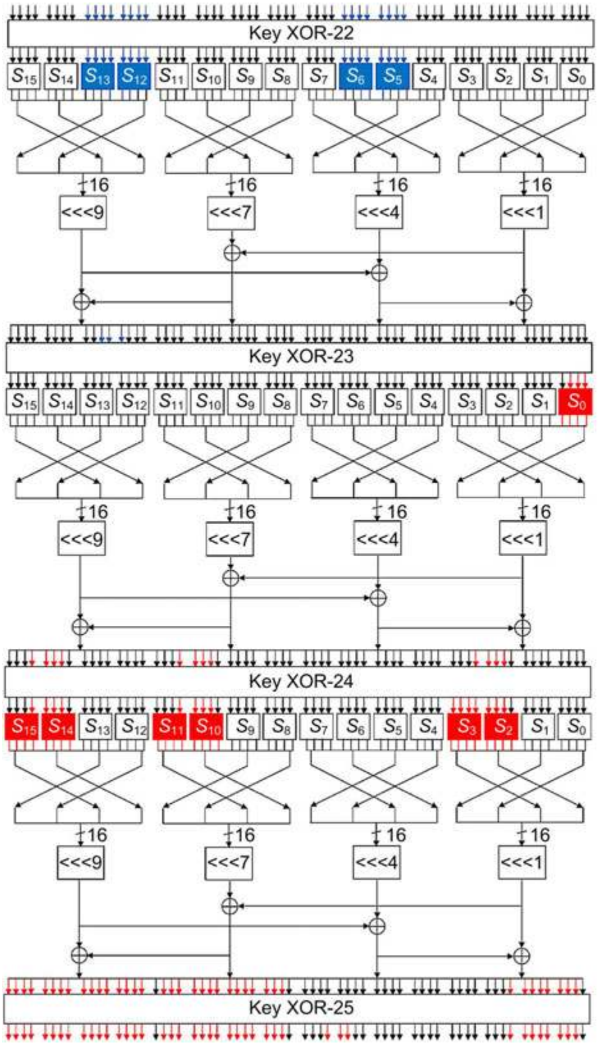

Fig. 5 Three-dimensional biclique for BORON-80 (References to color refer to the online version of this figure)

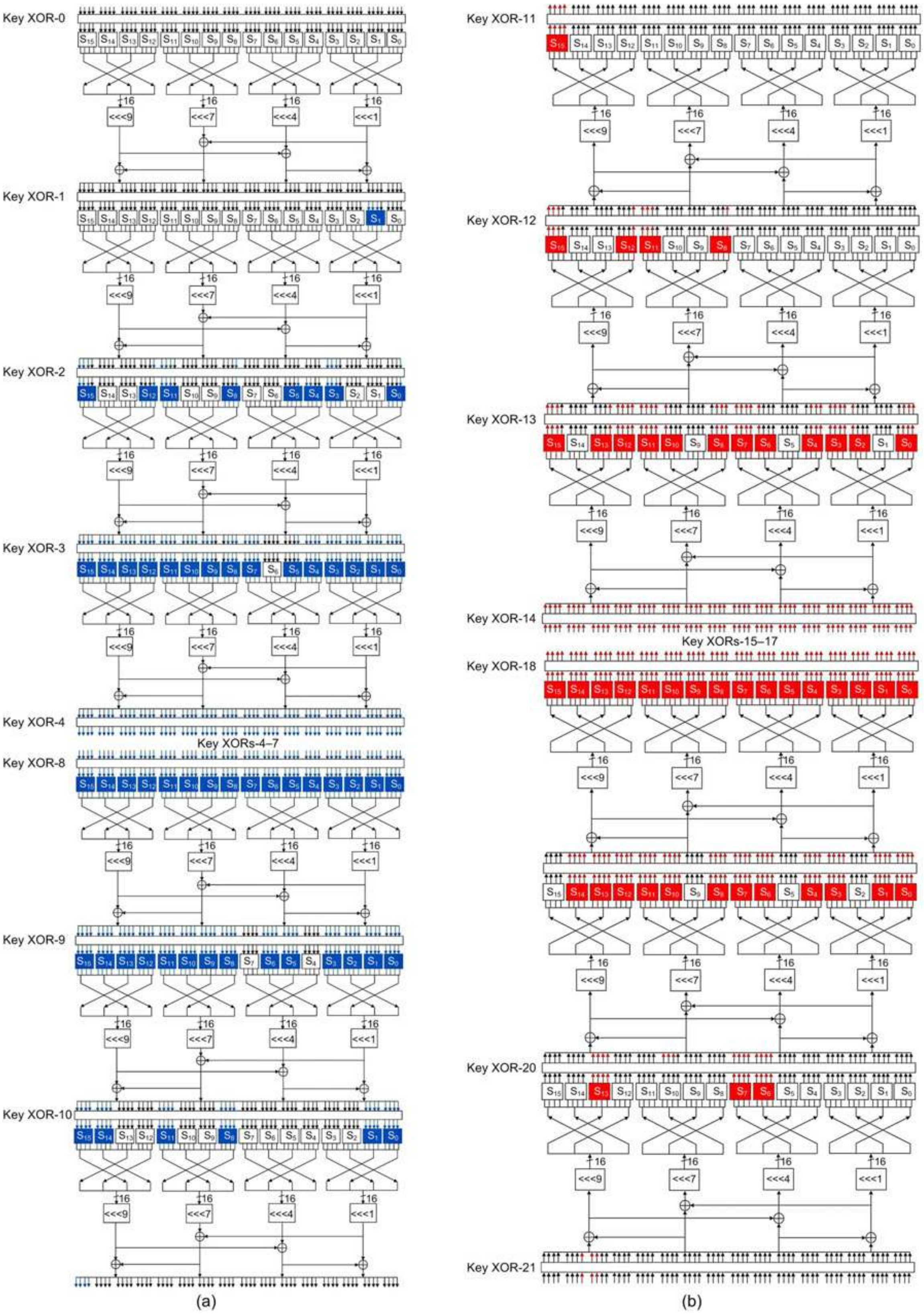

Fig. 6 Recomputation in forward directions for BORON-80 (a) and recomputation in backward directions for BORON-80 (b) (References to color refer to the online version of this figure)

3.7 Algebraic attack

The attacker applies the algebraic attack (Albrecht and Cid, 2009) usually on a stream cipher, because it is easier to succeed on a stream cipher than on a block cipher. The 4-bit to 4-bit S-box can be described by a minimum of 21 equations. $x=a\times 21$ quadratic equations in $y=a\times 8$ variables are used to examine the complete cipher (a represents the number of S-boxes used in the encryption algorithm and key scheduling algorithm).

In our cipher design for a single round of encryption, a total of 16 S-boxes are used, and for the 128-bit key scheduling, two S-boxes are used. For the 25 rounds of cipher, there are 25×16=400 S-boxes in the encryption system, and 25×2=50 S-boxes in the key scheduling algorithm.

The number of quadratic equations is given as

$$a=(400+50)\times 21=9450,$$

and the number of variables is given as

$$b=(400+50)\times 8=3600.$$

By applying the same method for the 80-bit key scheduling algorithm, we will obtain 8925 quadratic equations which can be formed with 3400 variables. We believe that by requiring 9450 quadratic equations with 3600 variables, the BORON cipher shows resistance against the algebraic attack.

3.8 Related key and slide attacks

No specific guidelines have been provided to design key scheduling algorithms. A large variety of algorithms can be formed and a wide variety of key related attacks can be mounted. Related key attack (Biham, 1993) and slide attack (Biryukov and Wagner, 2000) are two important attacks, which show weaknesses related to the key scheduling algorithms. The related key attack is also known as the chosen key attack and is applied successfully on a reduced round AES-256 (Biryukov *et al.*, 2009).

There is no successful key related attack that has been found on the key scheduling algorithm of PRESENT. Therefore, we adopt a similar style of key scheduling to the PRESENT block cipher. For design of the key scheduling algorithm, we have considered two approaches, which are given as follows:

1. use of the nonlinear component, i.e., S-box, in the design;

2. XOR operation of 5-bit from key register with round constant RC^i.

By using two S-boxes in 128-bit key scheduling and one S-box in 80-bit key scheduling algorithms, the BORON cipher increases the strength of the key scheduling algorithm.

3.9 Key collision attack

Key collision attack (Anderson *et al.*, 1998) can be mounted on any block cipher and depends on the key length regardless of the key scheduling algorithm. Key collision attack creates a message with a complexity of $2^{k/2}$, where k denotes the length of the key. The complexity of the created message is given as $2^{128/2}=2^{64}$.

3.10 Avalanche effect (Shi and Lee, 2000)

When a single bit in the input changes, the output changes considerably, resulting in an avalanche effect. For example, flipping a single bit in the input or in a key could change half of the bits in the cipher text. A cipher with a good avalanche effect has a higher probability to resist all possible types of attacks.

In case of a robust design of block ciphers, drastic changes in the cipher text are visible when a small change in the key or the plaintext takes place. Poor randomization occurs when a block cipher does not show the avalanche effect to a significant degree.

We have observed the output by applying single bit change in input plaintext/key bits. In the case of the BORON cipher, any single bit change in the key results changes more than half of the bits of the cipher text. Tables 11 and 12 show the avalanche effect.

4 Security comparison with standard algorithms

In this section, the security of BORON is compared with that of other standard algorithms. Table 13 compares the linear and differential complexities by considering the minimum number of active S-boxes for particular rounds. Table 14 compares the data and computational complexities of BORON with those of other ciphers.

5 Hardware and software performances of the BORON cipher

The BORON cipher is designed in such a way that it provides optimum performance on both the software platform and the hardware platform. The compact structure of the BORON cipher results in a small footprint area in the hardware and requires less memory in the software. We have considered the 32-bit ARM 7 LPC2129 processor for analyzing the software performance of the BORON cipher.

The footprint areas (number of GEs) are computed with a standard cell library based on the UMCL 180 0.18 μm logic process (UMCL18G212T3) (Poschmann, 2009; Bansod *et al.*, 2015). The memory size required for the BORON cipher on a 32-bit processor is 2408 bytes as flash memory and 1256 bytes as RAM memory. All other ciphers are programed in embedded C and implemented on a 32-bit processor for comparison. Fig. 7 shows the memory comparison between existing lightweight ciphers and the BORON cipher.

The round based architecture data path for the BORON cipher is shown in Fig. 8. The numbers of GEs are computed based on a UMCL180 standard cell library (Table 15). GE calculation for the BORON cipher is presented in Table 16. For the 128-bit

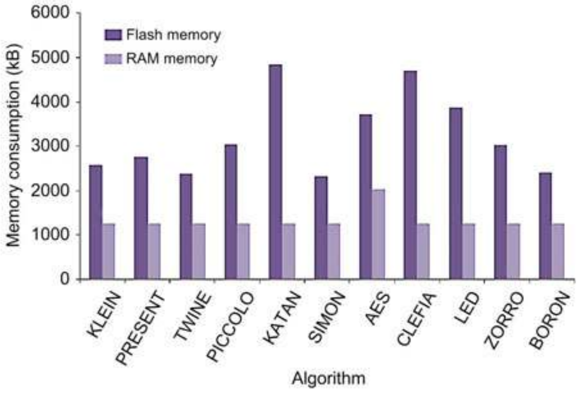

Fig. 7 Flash memory and RAM memory comparison between standard algorithms and the BORON cipher implemented on LPC2129

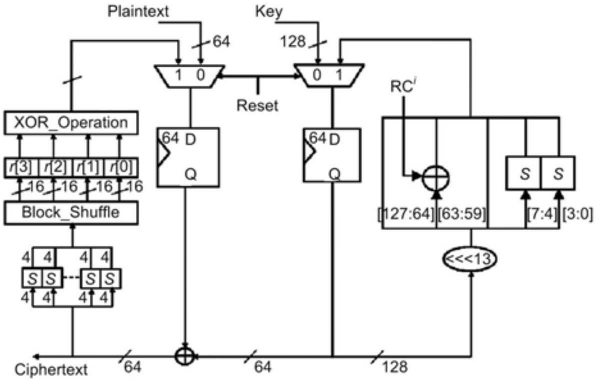

Fig. 8 Data path for the BORON cipher for 64-bit plaintext and 128-bit key

Table 11 Avalanche effect for BORON-80

Plaintext	Key	Ciphertext	Number of bits changed
0000 0000 0000 0000	0000 0000 0000 0000 0000	3cf7 2a8b 7518 e6f7	
	0010 0000 0000 0000 0000	fdf9 f345 3448 197a	32
	0000 0000 0000 0000 0010	fe91 0aec bee3 29b3	33

Table 12 Avalanche effect for BORON-128

Plaintext	Key	Ciphertext	Number of bits changed
0000 0000 0000 0000	0000 0000 0000 0000 0000 0000 0000 0000	94a1 05a7 d2f2 de42	
	0000 0000 0000 0000 0000 0000 0000 0010	7946 b520 9d6e c210	34
	0000 8000 0000 0000 0000 0000 0000 0000	2dcc 3b8d e115 e67c	36

Table 13 Linear and differential attack comparison

Cipher	Number of rounds	Number of active S-boxes	Number of known plaintext	Number of chosen plaintext	Reference
BORON	18	48	2^{98}	2^{96}	This paper
PRESENT	25	50	2^{102}	2^{100}	Bogdanov *et al.*, 2007
L-Block	15	32	2^{66}	2^{64}	Wu and Zhang, 2011
FEW	27	45	2^{90}	2^{90}	Kumar *et al.*, 2014
PICCOLO	30	30	2^{120}	2^{120}	Shibutani *et al.*, 1990

Table 14 Biclique attack comparison

Cipher	Data complexity	Computational complexity	Reference
BORON-80	2^{41}	$2^{79.56}$	This paper
PRESENT-80	2^{23}	$2^{79.54}$	Jeong *et al.*, 2012
PRESENT-128	2^{19}	$2^{127.42}$	
PICCOLO-80	2^{48}	$2^{79.13}$	
PICCOLO-128	2^{24}	$2^{127.35}$	
LED-64	2^{64}	$2^{63.58}$	
LED-80	2^{64}	$2^{79.37}$	
LED-96	2^{64}	$2^{95.37}$	
LED-128	2^{64}	$2^{127.37}$	

Table 15 Gate count of the UMCL18G212T3 library

Standard cell	Process	Number of GEs
NOT	0.18 μm	0.67
AND	0.18 μm	1.33
XOR	0.18 μm	2.67
D F.F.	0.18 μm	6.00

Table 16 Calculation of the number of GEs for the BORON-128 cipher

Data layer	Number of GEs	Key layer	Number of GEs
D register	384	Key register	768
S-Box	384	Shift operator	0
P-Layer	0	S-box	48
XOR	170.84	XOR RC	13.35
		Key XOR	170.84
Total	938.24	Total	1000.19

Total number of gates required for 128-bit key is 1938.43≈1939

BORON, the key scheduling complete cipher has a total of 1939 GEs, and for the 80-bit, the key scheduling complete cipher has a total of 1626 GEs.

The related key attack on TEA was mounted by Kelsey *et al.* (1997), which leads to developing an extension of TEA, called XTEA (Wheeler and Needham, 1997). Hardware implementation of TEA requires 2355 GEs (Poschmann, 2009) and XTEA requires at least 2000 GEs (Bogdanov *et al*., 2007).

Table 17 shows the comparison of lightweight ciphers with BORON based on parameters such as execution time, throughput, and the number of cycles required to convert plain text to cipher text. Throughput is computed at a 12 MHz frequency on the software platform. Fig. 9 shows the numbers of GEs (Bansod *et al*., 2015; 2016) of existing ciphers and the BORON cipher. All other versions have block size of 64 bits and key size of 128 bits. Table 18 shows the throughput comparison of the BORON cipher and the existing SP network cipher.

6 Conclusions

In this paper, we presented BORON, an ultra-lightweight and low power cipher. BORON has a compact design, resulting in a smaller foot print area and lower power consumption. BORON performs efficiently on both hardware and software platforms. It has achieved a great speed while encrypting the text, as it is based on the SP network as compared to the Feistel based ciphers. We showed the resistance of the BORON cipher mainly against linear, differential,

Table 17 Comparison with respect to throughput, execution time, and the number of cycles

Structure	Cipher	Block size (bit)	Key size (bit)	Execution time (μs)	Throughput (kb/s)	Number of cycles
SP network	LED	64	128	7092.86	9.00	425 572.00
	KLEIN	64	96	887.51	72.00	10 650.12
	BORON	**64**	**128**	**666.46**	**96.02**	**7997.52**
	HUMMINGBIRD-2	16	128	316.51	51.00	3798.12
	PRESENT	64	128	2648.65	24.16	31 783.80
Feistel	SPECK	64	128	49.02	1305.00	588.24
	SIMON	64	128	105.67	605.00	1268.04
	PICCOLO	64	128	227.68	281.00	2732.16
	CLEFIA	128	128	1048.01	122.00	12 576.12
	TWINE	64	128	592.87	108.00	7114.44

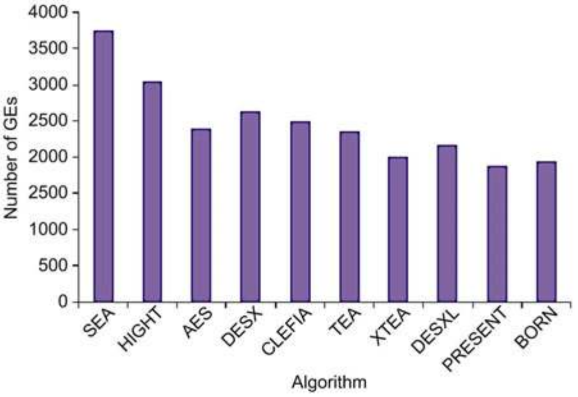

Fig. 9 Comparison of the number of GEs between standard algorithms and the BORON cipher

Table 18 Throughput improvement provided by the BORON cipher compared with other algorithms

Algorithm	Throughput improvement
PRESENT	297.43%
LED	96.00%
KLEIN	33.36%
HUMMINGBIRD 2	88.27%

key schedule, and key collide attacks. During the cipher design, we conducted extensive computer based searches for good S-boxes, the minimum number of active S-boxes, and the calculation of the Hamming weight for specific entries in LAT and DDT. The BORON cipher has a strong S-box and a robust permutation layer, which prevent the cipher design from undergoing the clustering of linear and differential trails. This property represents the robust design of the BORON cipher. In designing BORON, we have achieved a very small gate count, so it can be implemented for security in any small-scale embedded system. For applications like RFID tags and wireless sensor nodes, where a small number of GEs and low power consumption are required, we believe BORON is the best design. The BORON cipher can be further tested with a variety of advanced attacks.

References

Albrecht, M., Cid, C., 2009. Algebraic techniques in differential cryptanalysis. *LNCS*, **5665**:193-208. http://dx.doi.org/10.1007/978-3-642-03317-9_12

Anderson, R., Biham, E., Knudsen, L., 1998. Serpent: a proposal for the advanced encryption standard. 1st Advanced Encryption Standard (AES) Conf., p.1-23.

Bansod, G., Raval, N., Pisharoty, N., 2015. Implementation of a new lightweight encryption design for embedded security. *IEEE Trans. Inform. Forens. Secur.*, **10**(1):142-151. http://dx.doi.org/10.1109/TIFS.2014.2365734

Bansod, G. Pisharoty, N., Patil, A., 2016. PICO: an ultra lightweight and low power encryption design for pervasive computing. *Def. Sci. J.*, **66**(3):259-265. http://dx.doi.org/10.14429/dsj.66.9276

Beaulieu, R., Shors, D., Smith, J., *et al.*, 2013. The SIMON and SPECK families of lightweight block ciphers. Cryptology ePrint Archive, Report 2013/404.

Biham, E., 1993. New types of cryptanalytic attacks using related keys. EUROCRYPT, p.398-409. http://dx.doi.org/10.1007/3-540-48285-7_34

Biham, E., Shamir, A., 1991. Differential cryptanalysis of DES-like cryptosystems. *J. Cryptol.*, **4**(1):3-72. http://dx.doi.org/10.1007/BF00630563

Biryukov, A., Wagner, D., 2000. Advanced slide attacks. EUROCRYPT, p.589-606. http://dx.doi.org/10.1007/3-540-45539-6_41

Biryukov, A., Khovratovich, D., Nikolić, I., 2009. Distinguisher and related-key attack on the full AES-256. Cryptology ePrint Archive, Report 2009/241.

Bogdanov, A., Rijmen, V., 2011. Linear hulls with correlation zero and linear cryptanalysis of block ciphers. Cryptology ePrint Archive, Report 2011/123.

Bogdanov, A., Knudsen, L.R., Leander, G., *et al.*, 2007. PRESENT: an ultra-lightweight block cipher. *LNCS*, **4727**:450-466. http://dx.doi.org/10.1007/978-3-540-74735-2_31

Bogdanov, A., Khovratovich, D., Rechberger, C., 2011. Biclique cryptanalysis of the full AES. *LNCS*, **7073**: 344-371. http://dx.doi.org/10.1007/978-3-642-25385-0_19

Guo, J., Peyrin, T., Poschmann, A., *et al.*, 2011. The LED block cipher. *LNCS*, **6917**:326-341. http://dx.doi.org/10.1007/978-3-642-23951-9_22

Heys, H.M., 2001. A tutorial on linear and differential cryptanalysis. *Cryptologia*, **26**(3):189-221. http://dx.doi.org/10.1080/0161-110291890885

Jeong, K., Kang, H., Lee, C., *et al.*, 2012. Biclique cryptanalysis of lightweight block ciphers PRESENT, Piccolo and LED. Cryptology ePrint Archive, Report 2012/621.

Kelsey, J., Schneier, B., Wagner, D., 1997. Related-key cryptanalysis of 3-WAY, Biham DES, CAST, DES-X, new DES, RC2, and TEA. *LNCS*, **1334**:233-246. http://dx.doi.org/10.1007/BFb0028479

Kumar, M., Pal, S.K., Panigrahi, A., 2014. FeW: a lightweight block cipher. Cryptology ePrint Archive, Report 2014/326.

Leander, G., Poschmann, A., 2007. On the classification of 4 bit S-boxes. *LNCS*, **4547**:159-176. http://dx.doi.org/10.1007/978-3-540-73074-3_13

Matsui, M., 1993. Linear cryptanalysis method for DES cipher. *LNCS*, **765**:386-397. http://dx.doi.org/10.1007/3-540-48285-7_33

Matsui, M., 1994. On correlation between the order of S-boxes and the strength of DES. *LNCS*, **950**:366-375. http://dx.doi.org/10.1007/BFb0053451

Menezes, A.J., van Oorschot, P.C., Vanstone, S.A., 1996. Handbook of Applied Cryptography. CRC Press. http://dx.doi.org/10.1201/9781439821916

National Institute of Standards and Technology (NIST), 2001. Advanced Encryption Standard (AES). FIPS 197. http://csrc.nist.gov/publications/PubsFIPS.html

Poschmann, A., 2009. Lightweight Cryptography: Cryptographic Engineering for a Pervasive World. PhD Thesis, Ruhr-University Bochum, Germany.

Shi, Z., Lee, R.B., 2000. Bit permutation instructions for accelerating software cryptography. Proc. IEEE Int. Conf. on Application-Specific Systems, Architectures, and Processors, p.138-148. http://dx.doi.org/10.1109/ASAP.2000.862385

Shibutani, K., Isobe, T., Hiwatari, H., *et al.*, 2011. Piccolo: an ultra-lightweight blockcipher. *LNCS*, **6917**:342-357. http://dx.doi.org/10.1007/978-3-642-23951-9_23

Soleimany, H., Nyberg, K., 2012. Zero-correlation linear cryptanalysis of reduced-round LBlock. Cryptology ePrint Archive, Report 2012/570.

Sun, S., Hu, L., Wang, M., *et al.*, 2014a. Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties. Cryptology ePrint Archive, 2014/747.

Sun, S., Hu, L., Wang, P., *et al.*, 2014b. Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. *LNCS*, **8873**:158-178. http://dx.doi.org/10.1007/978-3-662-45611-8_9

Suzaki, T., Minematsu, K., Morioka, S., *et al.*, 2011. TWINE: a lightweight, versatile block cipher. ECRYPT Workshop on Lightweight Cryptography, p.146-169.

Wu, W., Zhang, L., 2011. LBlock: a lightweight block cipher. *LNCS*, **6715**:327-344. http://dx.doi.org/10.1007/978-3-642-21554-4_19

Zhang, W., Bao, Z., Lin, D., *et al.*, 2014. RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. Cryptology ePrint Archive, Report 2014/084.

Appendix: Test vectors

Table A1 Test vectors for 80-bit key

Plaintext:	0000 0000 0000 0000
Key:	0000 0000 0000 0000 0000
Ciphertext:	3cf7 2a8b 7518 e6f7
Plaintext:	0123 4567 89ab cdef
Key:	0000 0000 0000 0000 0000
Ciphertext:	5a66 4928 b961 c619

Table A2 Test vectors for 128-bit key

Plaintext:	0000 0000 0000 0000
Key:	0000 0000 0000 0000 0000 0000 0000 0000
Ciphertext:	94a1 05a7 d2f2 de42
Plain Text:	0123 4567 89ab cdef
Key:	0000 0000 0000 0000 0000 0000 0000 0000
Ciphertext:	953b e55b d5f2 68ba