INTERNATIONAL STANDARD

**ISO/IEC 29192-2**

First edition
2012-01-15

# Information technology — Security techniques — Lightweight cryptography

## Part 2: **Block ciphers**

*Technologies de l'information — Techniques de sécurité — Cryptographie pour environnements contraints*

*Partie 2: Chiffrements par blocs*

Reference number
ISO/IEC 29192-2:2012(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

## Contents

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

ISO/IEC 29192-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Security techniques*.

ISO/IEC 29192 consists of the following parts, under the general title *Information technology — Security techniques — Lightweight cryptography*:

— *Part 1: General*

— *Part 2: Block ciphers*

— *Part 3: Stream ciphers*

— *Part 4: Mechanisms using asymmetric techniques*

Further parts may follow.

# Introduction

This part of ISO/IEC 29192 specifies block ciphers suitable for lightweight cryptography, which are tailored for implementation in constrained environments.

ISO/IEC 29192-1 specifies the requirements for lightweight cryptography.

A block cipher maps blocks of *n* bits to blocks of *n* bits, under the control of a key of *k* bits.

The International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holder of these patent rights has assured ISO and IEC that he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of these patent rights is registered with ISO and IEC. Information may be obtained from:

Sony Corporation
System Technologies Laboratories
Attn Masanobu Katagi
Gotenyama Tec. 5-1-12 Kitashinagwa Shinagawa-ku
Tokyo
141-0001 Japan
Tel. +81-3-5448-3701
Fax +81-3-5448-6438
E-mail Masanobu.Katagi@jp.sony.com

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

# Information technology — Security techniques — Lightweight cryptography

# Part 2: Block ciphers

## 1 Scope

This part of ISO/IEC 29192 specifies two block ciphers suitable for applications requiring lightweight cryptographic implementations:

- PRESENT: a lightweight block cipher with a block size of 64 bits and a key size of 80 or 128 bits;
- CLEFIA: a lightweight block cipher with a block size of 128 bits and a key size of 128, 192 or 256 bits.

## 2 Normative references

There are no normative references for this part of ISO/IEC 29192.

## 3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

**3.1**
**block**
string of bits of defined length

[ISO/IEC 18033-1]

**3.2**
**block cipher**
symmetric encipherment system with the property that the encryption algorithm operates on a block of plaintext, i.e. a string of bits of a defined length, to yield a block of ciphertext

[ISO/IEC 18033-1]

**3.3**
**ciphertext**
data which has been transformed to hide its information content

[ISO/IEC 9798-1]

**3.4**
**key**
sequence of symbols that controls the operation of a cryptographic transformation (e.g. encipherment, decipherment)

NOTE Adapted from ISO/IEC 11770-1.

**3.5**
***n*-bit block cipher**
block cipher with the property that plaintext blocks and ciphertext blocks are *n* bits in length

[ISO/IEC 10116]

**3.6**
**plaintext**
unenciphered information

NOTE Taken from ISO/IEC 9797-1:1999.

**3.7**
**round key**
sequence of symbols derived from the key using the key schedule, and used to control the transformation in each round of the block cipher

## 4 Symbols

| | |
|---|---|
| 0x | A prefix for a binary string in hexadecimal notation |
| \|\| | Concatenation of bit strings |
| $a \leftarrow b$ | Updating a value of $a$ by a value of $b$ |
| $\oplus$ | Bitwise exclusive-OR operation |

## 5 Lightweight block cipher with a block size of 64 bits

### 5.1 PRESENT

#### 5.1.1 PRESENT algorithm

The PRESENT algorithm [10] is a symmetric block cipher that can process data blocks of 64 bits, using a key of length 80 or 128 bits. The cipher is referred to as PRESENT-80 or PRESENT-128 when using an 80-bit or 128-bit key respectively.

#### 5.1.2 PRESENT specific notations

| | |
|---|---|
| $\boldsymbol{K_i} = \boldsymbol{k^i_{63} \ldots k^i_0}$ | 64-bit round key that is used in round $i$ |
| $\boldsymbol{k^i_b}$ | bit $b$ of round key $\boldsymbol{K_i}$ |
| $\boldsymbol{K} = \boldsymbol{k_{79} \ldots k_0}$ | 80-bit key register |
| $\boldsymbol{k_b}$ | bit $b$ of key register $\boldsymbol{K}$ |
| *STATE* | 64-bit internal state |
| $b_{\mathrm{i}}$ | bit $i$ of the current *STATE* |
| $w_i$ | 4-bit word where $0 \leq i \leq 15$ |

### 5.1.3 PRESENT encryption

The PRESENT block cipher consists of 31 ‘rounds’, i.e. 31 applications of a sequence of simple transformations. A pseudocode description of the complete encryption algorithm is provided in Figure 1, where *STATE* denotes the internal state. The individual transformations used by the algorithm are defined in 5.1.5. Each round of the algorithm uses a distinct round key $K_i$ ($1 \leq i \leq 31$), derived as specified in 5.1.6. Two consecutive rounds of the algorithm are shown for illustrative purposes in Figure 2.

```
generateRoundKeys()
for $i = 1$ to 31 do
    addRoundKey(STATE,$K_i$)
    sBoxLayer(STATE)
    pLayer(STATE)
end for
addRoundKey(STATE,$K_{32}$)
```

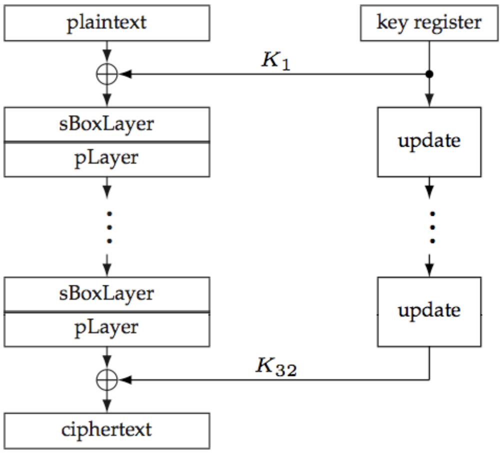

**Figure 1 — The encryption procedure of PRESENT**

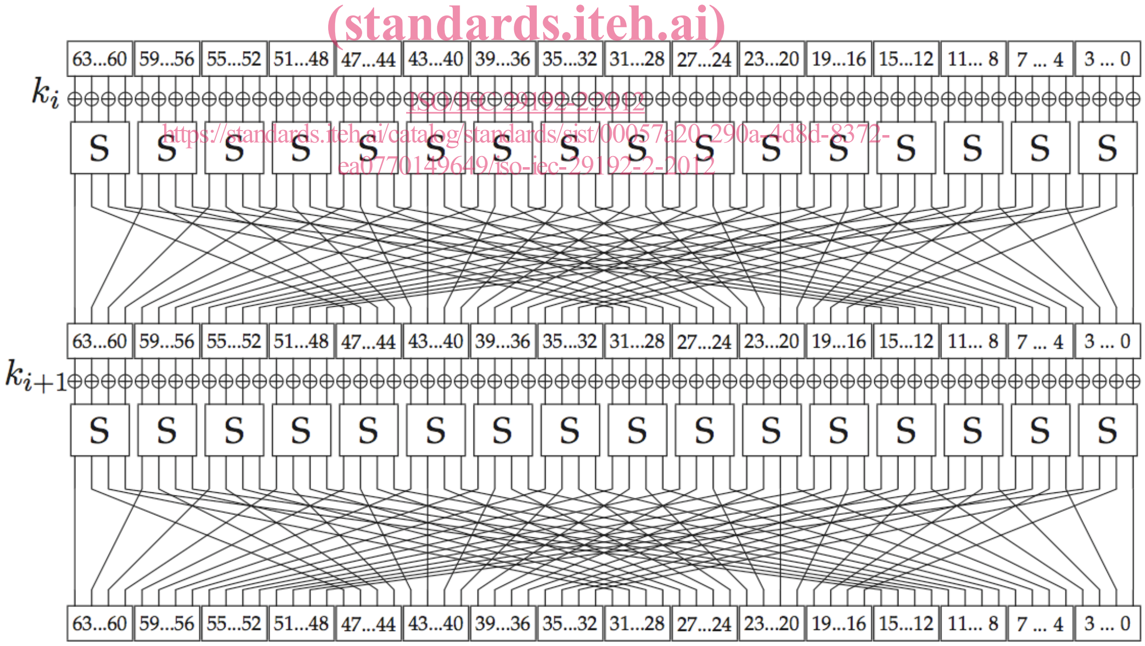

**Figure 2 — Two rounds of PRESENT**

### 5.1.4 PRESENT decryption

The complete PRESENT decryption algorithm is given in Figure 3. The individual transformations used by the algorithm are defined in 5.1.5. Each round of the algorithm uses a distinct round key $K_i$ ($1 \leq i \leq 31$), derived as specified in 5.1.6.

```
generateRoundKeys()
addRoundKey(STATE,$K_{32}$)
for $i = 31$ downto 1 do
    invpLayer(STATE)
    invsBoxLayer(STATE)

    addRoundKey(STATE,$K_i$)
end for
```

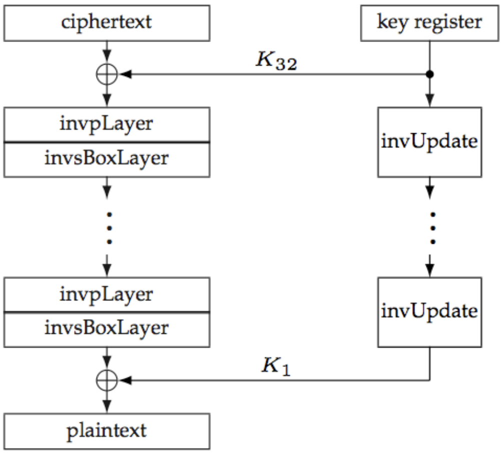

**Figure 3 — The decryption procedure of PRESENT**

### 5.1.5 PRESENT transformations

#### 5.1.5.1 addRoundKey

Given round key $\boldsymbol{K_i} = \boldsymbol{k^i_{63}\dots k^i_0}$ for $1 \le i \le 32$ and current *STATE* $b_{63}\dots b_0$, **addRoundKey** consists of the operation for $0 \le j \le 63$, $b_j \leftarrow b_j \oplus k^i_j$.

#### 5.1.5.2 sBoxLayer

The non-linear **sBoxLayer** of the encryption process of PRESENT uses a single 4-bit to 4-bit S-box $S$ which is applied 16 times in parallel in each round. The S-box transforms the input $x$ to an output $S(x)$ as given in hexadecimal notation in Table 1.

**Table 1 — PRESENT S-box**

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S(x)$ | C | 5 | 6 | B | 9 | 0 | A | D | 3 | E | F | 8 | 4 | 7 | 1 | 2 |

For **sBoxLayer** the current *STATE* $b_{63}\dots b_0$ is considered as sixteen 4-bit words $w_{15}\dots w_0$ where $w_i = b_{4*i+3} \,||\, b_{4*i+2} \,||\, b_{4*i+1} \,||\, b_{4*i}$ for $0 \le i \le 15$ and the output nibble $S(w_i)$ provides the updated state values as a concatenation $S(w_{15}) \,||\, S(w_{14}) \,||\, \dots \,||\, S(w_0)$.

#### 5.1.5.3 invsBoxLayer

The S-box used in the decryption procedure of PRESENT is the inverse of the 4-bit to 4-bit S-box $S$ that is described in 5.1.5.2. The inverse S-box transforms the input $x$ to an output $S^{-1}(x)$ as given in hexadecimal notation in Table 2.

**Table 2 — PRESENT inverse S-box**

| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S^{-1}(x)$ | 5 | E | F | 8 | C | 1 | 2 | D | B | 4 | 6 | 3 | 0 | 7 | 9 | A |

### 5.1.5.4 pLayer

The bit permutation **pLayer** used in the encryption routine of PRESENT is given by Table 3. Bit $i$ of *STATE* is moved to bit position $P(i)$.

**Table 3 — PRESENT permutation layer pLayer**

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(i)$ | 0 | 16 | 32 | 48 | 1 | 17 | 33 | 49 | 2 | 18 | 34 | 50 | 3 | 19 | 35 | 51 |

| $i$ | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(i)$ | 4 | 20 | 36 | 52 | 5 | 21 | 37 | 53 | 6 | 22 | 38 | 54 | 7 | 23 | 39 | 55 |

| $i$ | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(i)$ | 8 | 24 | 40 | 56 | 9 | 25 | 41 | 57 | 10 | 26 | 42 | 58 | 11 | 27 | 43 | 59 |

| $i$ | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(i)$ | 12 | 28 | 44 | 60 | 13 | 29 | 45 | 61 | 14 | 30 | 46 | 62 | 15 | 31 | 47 | 63 |

### 5.1.5.5 invpLayer

The inverse permutation layer **invpLayer** used in the decryption routine of PRESENT is given by Table 4. Bit $i$ of *STATE* is moved to bit position $P^{-1}(i)$.

**Table 4 — PRESENT inverse permutation Layer invpLayer**

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P^{-1}(i)$ | 0 | 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 60 |

| $i$ | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P^{-1}(i)$ | 1 | 5 | 9 | 13 | 17 | 21 | 25 | 29 | 33 | 37 | 41 | 45 | 49 | 53 | 57 | 61 |

| $i$ | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P^{-1}(i)$ | 2 | 6 | 10 | 14 | 18 | 22 | 26 | 30 | 34 | 38 | 42 | 46 | 50 | 54 | 58 | 62 |

| $i$ | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P^{-1}(i)$ | 3 | 7 | 11 | 15 | 19 | 23 | 27 | 31 | 35 | 39 | 43 | 47 | 51 | 55 | 59 | 63 |

## 5.1.6 PRESENT key schedule

### 5.1.6.1 PRESENT-80 and PRESENT-128

PRESENT can take keys of either 80 or 128 bits. In 5.1.6.2 the version with an 80-bit key (PRESENT-80) and in 5.1.6.3 the 128-bit version (PRESENT-128) is described.

### 5.1.6.2 80-bit key for PRESENT-80

The user-supplied key is stored in a key register $K$ and represented as $k_{79}k_{78} \ldots k_0$. At round $i$ the 64-bit round key $K_i = k^i_{63}k^i_{62} \ldots k^i_0$ consists of the 64 leftmost bits of the current contents of register $K$. Thus at round $i$ we have that:

$$K_i = k^i_{63}k^i_{62} \ldots k^i_0 = k_{79}k_{78} \ldots k_{16}.$$

After extracting the round key $K_i$, the key register $K = k_{79}k_{78} \ldots k_0$ is updated as follows.

1) $k_{79}k_{78} \ldots k_1k_0 \leftarrow k_{18}k_{17} \ldots k_{20}k_{19}$

2) $k_{79}k_{78}k_{77}k_{76} \leftarrow S[k_{79}k_{78}k_{77}k_{76}]$

3) $k_{19}k_{18}k_{17}k_{16}k_{15} \leftarrow k_{19}k_{18}k_{17}k_{16}k_{15} \oplus round\_counter$

In words, the key register is rotated by 61 bit positions to the left, the left-most four bits are passed through the PRESENT S-box, and the *round_counter* value $i$ is exclusive-ORed with bits $k_{19}k_{18}k_{17}k_{16}k_{15}$ of $K$ where the least significant bit of *round_counter* is on the right. The rounds are numbered from $1 \leq i \leq 31$ and $round\_counter = i$. Figure 4 depicts the key schedule for PRESENT-80 graphically.

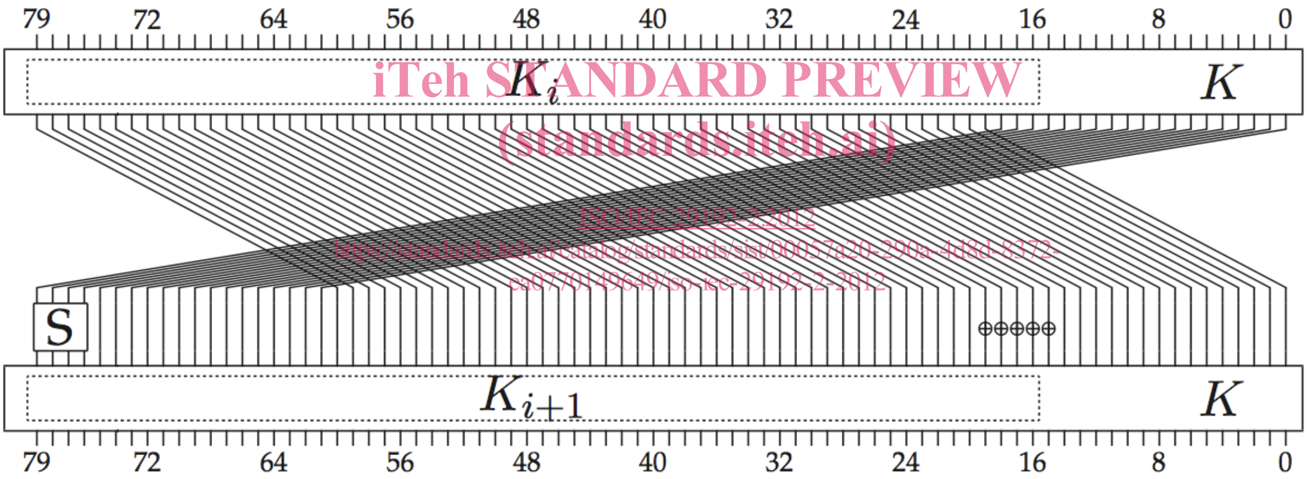

**Figure 4 — PRESENT-80 key schedule**

### 5.1.6.3 128-bit key for PRESENT-128

Similar to the 80-bit variant the user-supplied key is stored initially in a key register $K$ and is represented as $k_{127}k_{126} \ldots k_0$. At round $i$ the 64-bit round key $K_i = k^i_{63}k^i_{62} \ldots k^i_0$ consists of the 64 leftmost bits of the current contents of register $K$. Thus at round $i$ we have that:

$$K_i = k^i_{63}k^i_{62} \ldots k^i_0 = k_{127}k_{126} \ldots k_{64}.$$

After extracting the round key $K_i$, the key register $K = k_{127}k_{126} \ldots k_0$ is updated as follows.

1) $k_{127}k_{126} \ldots k_1k_0 \leftarrow k_{66}k_{65} \ldots k_{68}k_{67}$

2) $k_{127}k_{126}k_{125}k_{124} \leftarrow S[k_{127}k_{126}k_{125}k_{124}]$

3) $k_{123}k_{122}k_{121}k_{120} \leftarrow S[k_{123}k_{122}k_{121}k_{120}]$

4) $k_{66}k_{65}k_{64}k_{63}k_{62} \leftarrow k_{66}k_{65}k_{64}k_{63}k_{62} \oplus round\_counter$

In words, the key register is rotated by 61 bit positions to the left, the left-most eight bits are passed through the PRESENT S-box, and the *round_counter* value $i$ is exclusive-ORed with bits $\boldsymbol{k_{66}k_{65}k_{64}k_{63}k_{62}}$ of $\boldsymbol{K}$ where the least significant bit of *round_counter* is on the right. The rounds are numbered from $1 \le i \le 31$ and *round_counter* $= i$. Figure 5 depicts the key schedule for PRESENT-128 graphically.

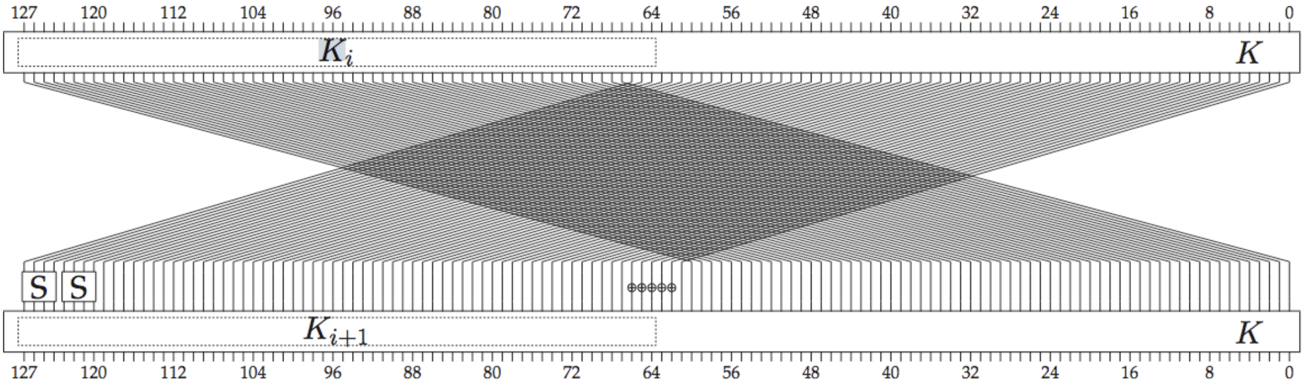

**Figure 5 — PRESENT-128 key schedule**

## 6 Lightweight block cipher with a block size of 128 bits

### 6.1 CLEFIA

#### 6.1.1 CLEFIA algorithm

The CLEFIA algorithm [14] is a symmetric block cipher that can process data blocks of 128 bits using a cipher key of length 128, 192, or 256 bits. The number of rounds is 18, 22 and 26 for CLEFIA with 128-bit, 192-bit and 256-bit keys, respectively. The total number of round keys depends on the key length. The CLEFIA encryption and decryption functions require 36, 44 and 52 round keys for 128-bit, 192-bit and 256-bit keys, respectively.

#### 6.1.2 CLEFIA specific notations

$a_{(b)}$ bit string of bit length $b$

$\{0,1\}^n$ A set of $n$-bit binary strings

$\bullet$ Multiplication in GF($2^n$)

$<<<i$ $i$-bit left cyclic shift operation

~ $a$ Bitwise complement of bit string $a$

$\Sigma^n$ $n$ times operations of the DoubleSwap function $\Sigma$