INTERNATIONAL STANDARD

ISO/IEC 29192-2

Second edition
2019-11

Information security — Lightweight cryptography —

Part 2: Block ciphers

Sécurité de l'information — Cryptographie pour environnements contraints —
Partie 2: Chiffrements par blocs

Reference number
ISO/IEC 29192-2:2019(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

This second edition cancels and replaces the first edition (ISO/IEC 29192-2:2012), which has been technically revised.

The main changes compared to the previous edition are as follows:

— the LEA algorithm has been added to 6.3;

— numerical examples and feature tables of LEA have been added to B.3 and Annex C.

A list of all parts in the ISO/IEC 29192 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

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 International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

ISO and IEC takes no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO and IEC that he/she 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 this patent right is registered with ISO and IEC. Information may be obtained from the patent database available at www.iso.org/patents.

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

Information security — Lightweight cryptography —

Part 2:
Block ciphers

1 Scope

This document specifies three 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;

— LEA: 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 in this document.

3 Terms and definitions

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

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
block
string of bits of defined length

[SOURCE: ISO/IEC 18033-1:2015, 2.8]

3.2
block cipher
symmetric encipherment system with the property that the encryption algorithm operates on a *block* (3.1) of *plaintext* (3.6), i.e. a string of bits of a defined length, to yield a block of *ciphertext* (3.3)

[SOURCE: ISO/IEC 18033-1:2015, 2.9]

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

[SOURCE: ISO/IEC 10116:2017, 3.2]

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

[SOURCE: ISO/IEC 18033-1:2015, 2.27]

3.5
***n*-bit block cipher**
block cipher (3.2) with the property that *plaintext* (3.6) blocks and *ciphertext* (3.3) blocks are *n* bits in length

[SOURCE: ISO/IEC 18033-1:2015, 2.29]

3.6
plaintext
unenciphered information

[SOURCE: ISO/IEC 9798-1:2010, 3.19]

3.7
round key
sequence of symbols derived from the *key* (3.4) using the key schedule, and used to control the transformation in each round of the *block cipher* (3.2)

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 General

In this clause, a 64-bit lightweight block cipher is specified: PRESENT in 5.2.

Annex A defines the object identifiers which shall be used to identify the algorithm specified in Clause 5. Annex B provides numerical examples of the block ciphers described in this document. Annex C summarizes the lightweight properties of the block ciphers described in this document. Annex D gives a limit on the number of block cipher encryption operations that should be performed using a single key.

5.2 PRESENT

5.2.1 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.

5.2.2 PRESENT specific notation

$K_i = k^i_{63} \ldots k^i_0$	64-bit round key that is used in round i
k^i_b	bit b of round key K_i
$K = k_{79} \ldots k_0$	80-bit key register
k_b	bit b of key register K

STATE	64-bit internal state
b_i	bit i of the current *STATE*
w_i	4-bit word where $0 \le i \le 15$

5.2.3 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 5.2.5. Each round of the algorithm uses a distinct round key K_i ($1 \le i \le 31$), derived as specified in 5.2.6. 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)
```

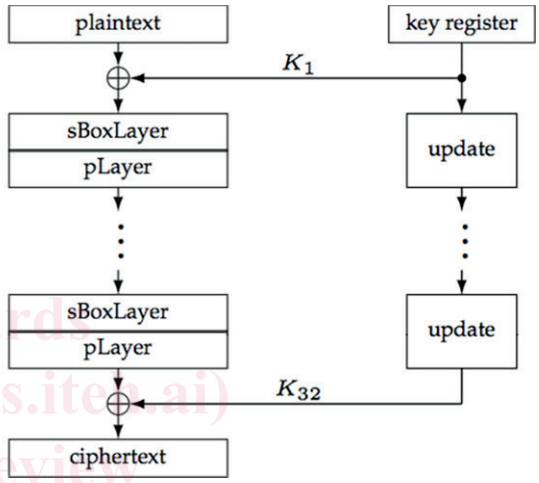

Figure 1 — The encryption procedure of PRESENT

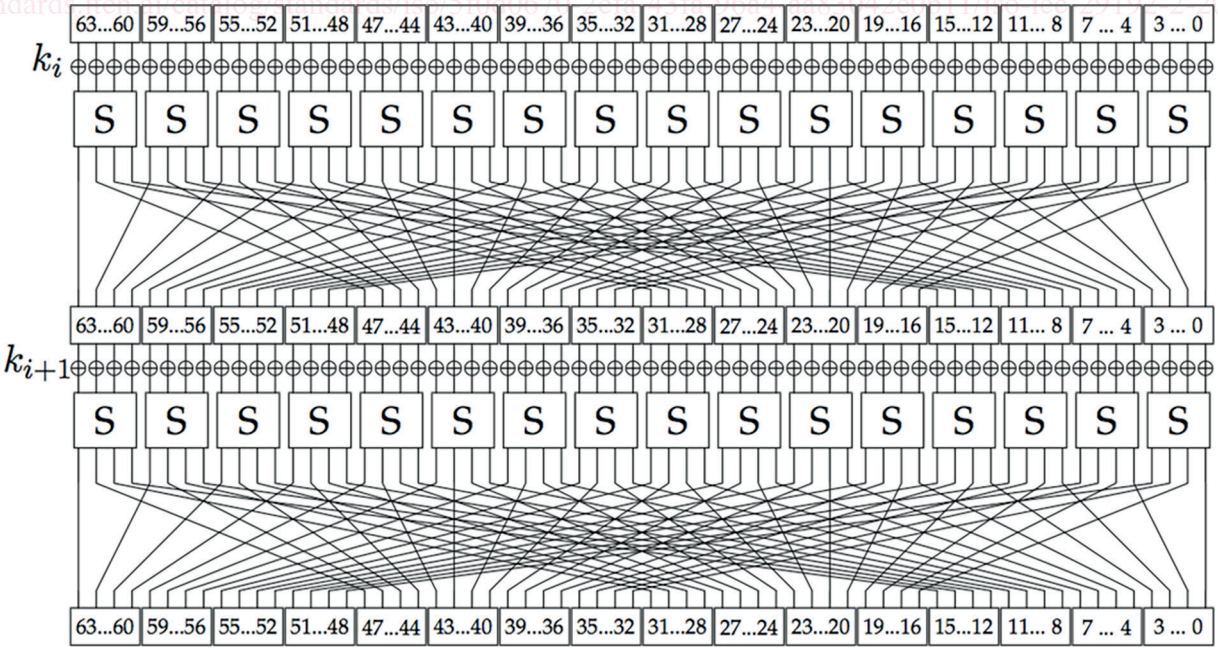

Figure 2 — Two rounds of PRESENT

5.2.4 PRESENT decryption

The complete PRESENT decryption algorithm is given in Figure 3. The individual transformations used by the algorithm are defined in 5.2.5. Each round of the algorithm uses a distinct round key K_i ($1 \le i \le 31$), derived as specified in 5.2.6.

```
generateRoundKeys()
addRoundKey(STATE,K32)
for i = 31 downto 1 do
   invpLayer(STATE)
   invsBoxLayer(STATE)

   addRoundKey(STATE,Ki)
end for
```

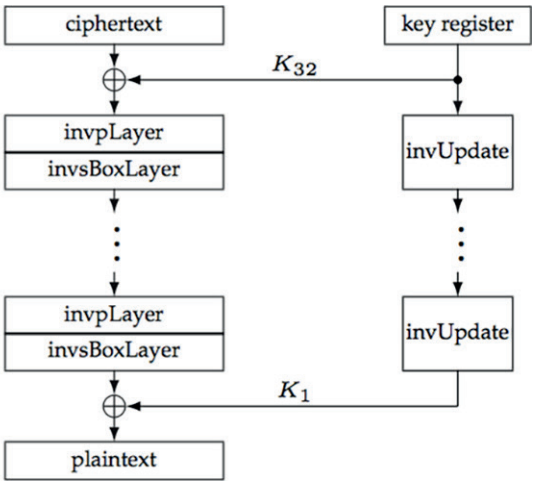

Figure 3 — The decryption procedure of PRESENT

5.2.5 PRESENT transformations

5.2.5.1 addRoundKey

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

5.2.5.2 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* $b_{63} \ldots b_0$ is considered as sixteen 4-bit words w_{15} ... 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 $S(w_{15}) \| S(w_{14}) \| \ldots \| S(w_0)$.

5.2.5.3 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 5.2.5.2. 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

5.2.5.4 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

5.2.5.5 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

5.2.6 PRESENT key schedule

5.2.6.1 PRESENT-80 and PRESENT-128

PRESENT can take keys of either 80 or 128 bits. In 5.2.6.2, the version with an 80-bit key (PRESENT-80) and in 5.2.6.3 the 128-bit version (PRESENT-128) is described.

5.2.6.2 80-bit key for PRESENT-80

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

$$K_i = k_{63}^i k_{62}^i \ldots k_0^i = k_{79}k_{78}\ldots k_{16}$$

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

1) $k_{79}k_{78}...k_1k_0 \leftarrow k_{18}k_{17}...k_{20}k_{19}$
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 $1 \le i \le 31$ and *round_counter* = i. Figure 4 depicts the key schedule for PRESENT-80 graphically.

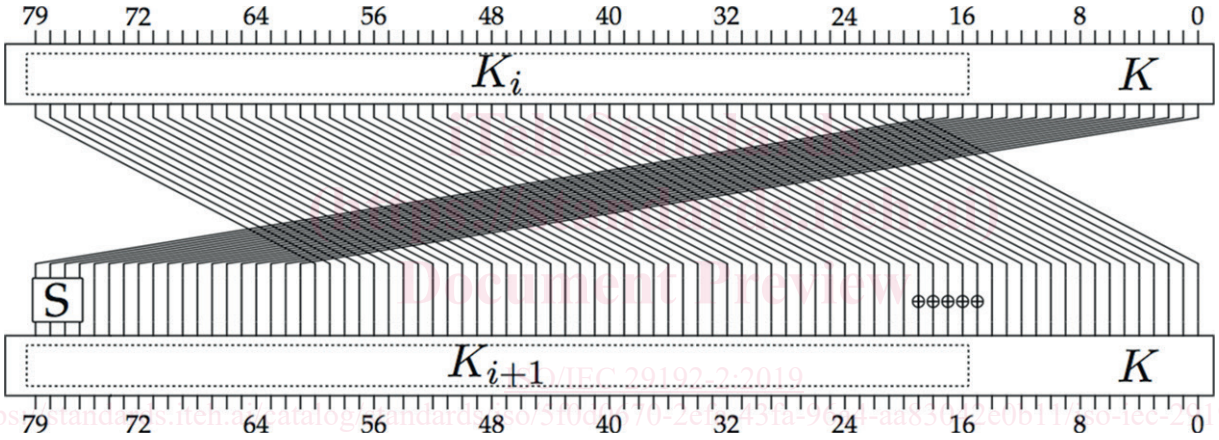

Figure 4 — PRESENT-80 key schedule

5.2.6.3 128-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 $k_{127}k_{126}...k_0$. At round i the 64-bit round key $K_i = k_{63}^i k_{62}^i \ldots k_0^i$ consists of the 64 leftmost bits of the current contents of register K. Thus at round i, K_i is as follows:

$$K_i = k_{63}^i k_{62}^i \ldots k_0^i = k_{127}k_{126}\ldots k_{64}$$

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

1) $k_{127}k_{126}...k_1k_0 \leftarrow k_{66}k_{65}...k_{68}k_{67}$
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 $k_{66}k_{65}k_{64}k_{63}k_{62}$

of 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.

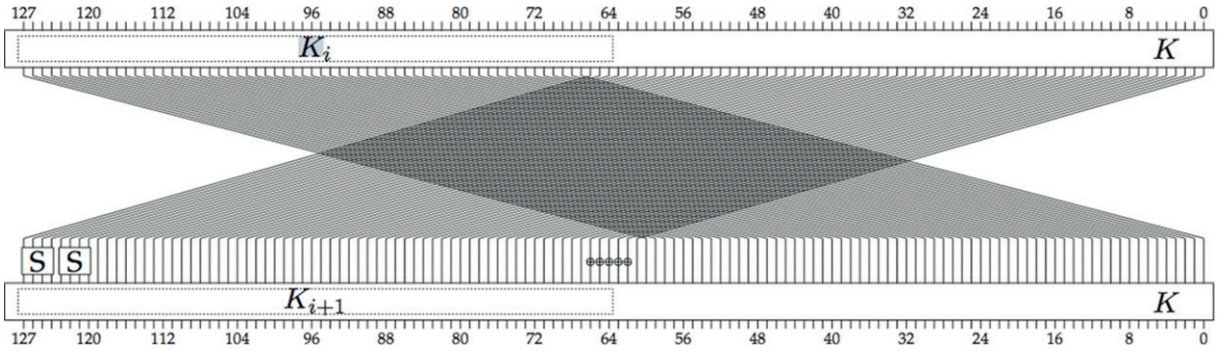

Figure 5 — PRESENT-128 key schedule

6 Lightweight block ciphers with a block size of 128 bits

6.1 General

In this clause, two 128-bit lightweight block ciphers are specified: CLEFIA in 6.2 and LEA in 6.3.

Annex A defines the object identifiers which shall be used to identify the algorithms specified in Clause 6. Annex B provides numerical examples of the block ciphers described in this document. Annex C summarizes the lightweight properties of the block ciphers described in this document. Annex D gives a limit on the number of block cipher encryption operations that should be performed using a single key.

6.2 CLEFIA

6.2.1 CLEFIA algorithm

The CLEFIA algorithm[15] 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.

6.2.2 CLEFIA specific notation

$a_{(b)}$	bit string of bit length b
$\{0,1\}^n$	a set of n-bit binary strings
$\cdot$	multiplication in GF(2^n)
$<<<i$	i-bit left cyclic shift operation
$\sim a$	bitwise complement of bit string a
Σ^n	n times operations of the DoubleSwap function Σ

6.2.3 CLEFIA encryption

The encryption process of CLEFIA is based on the 4-branch r-round generalized Feistel structure $GFN_{4,r}$. Let $P,\ C \in \{0,1\}^{128}$ be a plaintext and a ciphertext. Let $P_i,\ C_i \in \{0,1\}^{32}$ $(0 \le i < 4)$ be divided plaintexts

and ciphertexts where $P = P_0 \,||\, P_1 \,||\, P_2 \,||\, P_3$ and $C = C_0 \,||\, C_1 \,||\, C_2 \,||\, C_3$. Let WK_0, WK_1, WK_2, $WK_3 \in \{0,1\}^{32}$ be whitening keys and $RK_i \in \{0,1\}^{32}$ $(0 \le i < 2r)$ be round keys provided by the key schedule. Then, r-round encryption function ENC_r is defined as follows:

ENC_r:

1) $T_0 \,||\, T_1 \,||\, T_2 \,||\, T_3 \leftarrow P_0 \,||\, (P_1 \oplus WK_0) \,||\, P_2 \,||\, (P_3 \oplus WK_1)$

2) $T_0 \,||\, T_1 \,||\, T_2 \,||\, T_3 \leftarrow GFN_{4,r}\,(RK_0, \ldots, RK_{2r-1}, T_0, T_1, T_2, T_3)$

3) $C_0 \,||\, C_1 \,||\, C_2 \,||\, C_3 \leftarrow T_0 \,||\, (T_1 \oplus WK_2) \,||\, T_2 \,||\, (T_3 \oplus WK_3)$

6.2.4 CLEFIA decryption

The decryption function DEC_r is defined as follows:

DEC_r:

1) $T_0 \,||\, T_1 \,||\, T_2 \,||\, T_3 \leftarrow C_0 \,||\, (C_1 \oplus WK_2) \,||\, C_2 \,||\, (C_3 \oplus WK_3)$

2) $T_0 \,||\, T_1 \,||\, T_2 \,||\, T_3 \leftarrow GFN_{4,r}^{-1}\,(RK_0, \ldots, RK_{2r-1}, T_0, T_1, T_2, T_3)$

3) $P_0 \,||\, P_1 \,||\, P_2 \,||\, P_3 \leftarrow T_0 \,||\, (T_1 \oplus WK_0) \,||\, T_2 \,||\, (T_3 \oplus WK_1)$

Figure 6 illustrates both ENC_r and DEC_r.

