# The FEAL Cipher Family

*Shoji MIYAGUCHI*

*Communications and Information Processing Laboratories, NTT*
*1-2356, Take, Yokosuka-shi, Kanagawa, 238-03 Japan*

## 1 FEAL cipher family

FEAL-8 has been expanded to FEAL-N (N round FEAL with 64-bit key), where FEAL-N with N=4/8 is identical to FEAL-4/-8 which have been previously published respectively. N is a user defined parameter ($N \geq 4$, N:even, $N=2^x$, $x \geq 2$ is recommended). FEAL-N has also been expanded to FEAL-NX (X: expansion, N round FEAL with 128-bit key) that accepts 128-bit keys. When the right half of the 128-bit key is all zeros, FEAL-NX works as FEAL-N. Upward compatibility is an important concept of the FEAL cipher family [1].

## 2 Round number N

The author believes that most cipher applications can avoid chosen plaintext attacks by the countermeasures described in Annex-1. Increased N in FEAL-N or FEAL-NX can avoid chosen plaintext attacks. Where the countermeasures are applicable, small values of N (eg. N=8) should be used. If none of the countermeasures can be applied or their effectiveness is unclear, the value of N in FEAL-N or FEAL-NX should be increased.

## 3 64-bit key and 128-bit key

The author thinks that exhaustive searches for FEAL-N 64-bit keys may be possible if LSI technology advances sufficiently, as shown in Annex-2. He feels that FEAL-N may weaken against exhaustive search within one to two decades. Therefore, FEAL-NX which accepts 128-bit keys has been designed.

A.J. Menezes and S.A. Vanstone (Eds.): Advances in Cryptology - CRYPTO '90, LNCS 537, pp. 627-638, 1991.

ECB mode. This case includes the situation where the attacker uses the tool without permission of the victim.

## 2 Countermeasures

One of the following may be effective to prevent the attack.

**(1) Elimination of cooperation and improper use**

A cipher user should not encipher data from outside (possible chosen plaintexts) in the ECB mode using his secret key, and then return the ciphertexts so generated. He also should not provide an encipherment tool that outputs data enciphered in the ECB mode using the user's secret key, i.e. eliminate user cooperation with the attacker, improper uses of the cipher as in Case-1 or Case-2 above.

**(2) Inhibitation of the ECB**

ECB mode should be inhibited, and CBC or CFB modes (feedback modes, see ISO8372) should be used(†). Initial values of each mode are changed with each use of the cipher. The cipher user determines the initial value, IV, with his rule that is secret to others (possible attackers). The value of IV may be revealed after the first block of plaintexts is given. If the initial value IV must be sent to the receiver, it can be appended to the head of the ciphertext.

† : International standards, ISO8732, ISO9160 and ISO10126, recommend the use of CBC or CFB modes in cipher communications.

**(3) Key changes** : The key, the target of chosen plaintext attack should be changed after each key use (††).

†† : International standard, ISO8732, recommends that the key be changed for each communication session.

**(4) Miscellaneous**: Individual countermeasures can be developed for each cipher application. For instance, if the volume of chosen plaintexts is 1 Mega-byte, maximum data is limited to a lesser volume (eg. 100 kilo-byte) within one key lifetime.

# Annex-2 Exhaustive search for 64-bit keys

## 1 Progress of LSI C-MOS technology

A rough approximation is that LSI processing speed is inversely proportional to the channel width while LSI integrated transistor density is inversely proportional to the square of the channel width. The past decade has shown that LSI channel width has decreased to one third of the original width; consequently, processing speeds

have roughly tripled and the transister density has increased by a factor of 10. It is predicted that similar LSI technology advances will continue in the future.

## 2 FEAL-attack LSI chip

Assume that a special FEAL-attack LSI chip can be fabricated, i.e., enciphering speed of the chip is 3 times the current speed, 300 Mbps $\approx 3 \times 96$ Mbps (3×speed of current FEAL-8 chip). The chip has ten FEAL-N processing elements, while chip price is 20 US dollars.

## 3 Exhaustive search equipment

The equipment includes 100,000 FEAL-attack chips. The eqipment inputs one plaintext ($P$) and its ciphertext ($C$) which was enciphered by a secret key, and enciphers $P$ repeatedly and in parallel using key $K_i$ for i from i=1 to $2^{64}$, producing ciphertext $C_i$ and comparing $C_i$ to $C$. Here, the key $K_i$ is generated in the equipment. If the equipment finds $C_i = C$, it outputs the value of $K_i$. The enciphering speed (V) of the equipment is V = $300 \times 10^6$ (bps)$\times 10^5$(chips)×10 (FEAL-8 processing elements/chip) = 300 Tera-bps. Assume that the equipment price is ten times the total chip cost. For reference, *the equipment cannot be re-designed to input and/or output texts at a speed of 300 Tera-bps* because it is technically impossible.

## 4 Equipment performance

Let p be the probability to discover the secret key. Then the price of the equipment and the time to discover the key are given as: (a) the price = 20 US (dollars/chip)× 100,000 (chips) × 10 (times) = 20 million US dollars, (b) the time = $((2^{64} \times 64(\text{bits}))/(300 \times 10^6$ (bps)$\times 10^5$(chips)×10 (FEAL-8 processing elements/chip)$\times 8.64 \times 10^4$ (sec/day) ) × p ≈ 45 days × p).

That is: Equipment price (million dollars) × Time to discover the key (days) ≈ 20 × 45× p (million dollars×days)

## 5 Conclusion

FEAL-N (64-bit key) may weaken against exhaustive search within one to two decades.

**Reference**: Exhaustive search for DES 56-bit keys
(a) the price: 20 million US dollars (the same as above)
(b) the time = $((2^{56} \times 64$ (bits))/(3 × 20(†)$\times 10^6$(bps)$\times 10^5$ (chips) ×10 (DES elements/chip)×8.64 × $10^4$ (sec/day) ) ≈ 1 days ).

† 20 Mbps (approximate speed of DES chip)

## Annex-3 Comparison of Chosen plaintext attack and Exhaustive search

The author believes that a chosen plaintext attack (CPA) on $2^{64}$ blocks is extremely ineffective, and cannot be equated with exhaustive search on $2^{64}$ keys, *even if the countermearsures to CPA described in Anndex-1 are ignored.* The reasons are:

(a) A cipher user (victim) or attacker has to use encipherment equipment that input/outputs texts at a speed of 300 Tera-bps which cannot be realized, if the speed of CPA is comparable with that for exhaustive search (see Clause 3 of Annex-2).
(b) The sheer volume of data to be transferred, $2^{64} \times 8$ bytes ($1.47 \times 10^{20}$ bytes), prevents CPA within any reasonable period or price.

To the author, it seems to be questionable to compare the number of chosen plaintext attacks with *that of exhaustive search* when the number is very big such as $2^{64}$. This comparison may lead to the misunderstanding that both attacks might be equally strong.

## Annex-4 Programming techniques for FEAL

Programming examples of $S$ functions of FEAL are shown below.

**1 Program example using 1-bit left rotation instruction**

If a 1-bit (or 2-bit) left rotation is used, $S_0/S_1$ functions can be coded easily. $S_1(X_1, X_2) = \mathrm{Rot2}(\ (X_1 + X_2 + 1\ )\ \mathrm{mod}\ 256\ )$ is given below in typical 16-bit $\mu$p assembly language.

```
add   R1, R2     ; R1 ← (R1) + (R2) mod 256,
                   where X1/X2 is in R1/R2.
inc   R1         ; R1 ← (R1) + 1
rot   R1         ; R1 ← 1-bit left rotation on (R1)
rot   R1         ; R1 ← 1-bit left rotation on (R1)
```

**2 Table search technique:** This is suitable for processors that have a base register to point the table. The idea is shown for $S_1(X_1, X_2) = \mathrm{Rot2}(\ (X_1 + X_2 + 1)\ \mathrm{mod}\ 256)$
Step-1: X $\leftarrow$ $X_1 + X_2$ (add $X_1$ and $X_2$, then sum is stored into X)
Step-2: Y $\leftarrow$ 2-bit left rotation on (X+1) from table pointed X

**3 Note:** Program coding of paired stages in FEAL data randomization is very useful to decrease dynamic program steps.

## Annex-5 FEAL Specifications

This Annex uses the following conventions and notations.

(1) A,Ar,... : blocks

(2) $(A, B, \ldots)$: concatenation of blocks in this order

(3) A B: exclusive-or operation of A and B

(4) $\phi$ : zero block, 32-bits long

(5) =: Transfer from left side to right side

(6) Bit position: 1,2,3,... from the first left side bit (MSB) in a block towards the right.

### 1 FEAL options

(1) Round number (N): Determines the round number (N) for FEAL data randomization, where N $\geq$ 4 and even. $2^x$, x$\geq$2 is recommended.

(2) Key parity: Indicates: (a) Use of key parity bits in a key-block, or (b) Non-use of key parity bits in a key-block

### 2 Enciphering algorithm

Plaintext $P$ is separated into $L_0$ and $R_0$ of equal lengths (32 bits), i.e., $P = (L_0, R_0)$.

First, $$(L_0, R_0) = (L_0, R_0) \oplus (K_N, K_{N+1}, K_{N+2}, K_{N+3})$$

Next, $$(L_0, R_0) = (L_0, R_0) \oplus (\phi, L_0)$$

Next, and calculate the equations below for r from 1 to N iteratively,

$$R_r = L_{r-1} \oplus f(R_{r-1}, K_{r-1})$$
$$L_r = R_{r-1}$$

where extended keys $K_i$s are defined in Clause 4, and function $f$ is defined in Clause 5. Output of r-th round is $(L_r, R_r)$.

Interchange the final output of the iterative calculation, $(L_N, R_N)$, into $(R_N, L_N)$. Next calculate:

$$(R_N, L_N) = (R_N, L_N) \oplus (\phi, R_N)$$

Lastly, $$(R_N, L_N) = (R_N, L_N) \oplus (K_{N+4}, K_{N+5}, K_{N+6}, K_{N+7})$$

Ciphertext is given as $(R_N, L_N)$.

## 3 Deciphering algorithm

Ciphertext $(R_N, L_N)$ is separated into $R_N$ and $L_N$ of equal lengths.

First, $$(R_N, L_N) = (R_N, L_N) \oplus (K_{N+4}, K_{N+5}, K_{N+6}, K_{N+7})$$

Next, $$(R_N, L_N) = (R_N, L_N) \oplus (\phi, R_N)$$

Next, calculate the equations below for r from N to 1 iteratively,

$$L_{r-1} = R_r \oplus f(L_r, K_{r-1})$$
$$R_{r-1} = L_r$$

Interchange the final output of the iterative calculation, $(R_0, L_0)$, into $(L_0, R_0)$. Next calculate:

$$(L_0, R_0) = (L_0, R_0) \oplus (\phi, L_0)$$

Lastly, $$(L_0, R_0) = (L_0, R_0) \oplus (K_N, K_{N+1}, K_{N+2}, K_{N+3})$$

Plaintext is given as $(L_0, R_0)$. Data randomization for the enciphering / deciphering algorithms is shown in Figure 1.

## 4 Key schedule

First , the key schedule of FEAL-NX is described, where the functions used are defined in clause 5. The key schedule yields the extended key $K_i$ (i=0,1,2,...,N+7) from the 128-bit key.

### 4.1 Definition of left key $K_L$ and right key $K_R$

Inputted 128-bit key is equally divided into a 64-bit left key, $K_L$, and a 64-bit right key, $K_R$, i.e., $(K_L, K_R)$ is the inputted 128-bit key.

### 4.2 Parity bit processing

(1) Non-use of key parity bits: There is no special processing here.

(2) Use of key parity bits: Bit positions, 8, 16, 24, 32, 40, 48, 56, 64, of both $K_L$ and $K_R$ are set to zeros, i.e., all parity bits in the key block are set to zero.

Note: How to use parity bits is outside the scope of the FEAL-NX.

### 4.3 Iterative calculation

(1) Processing of the right key $K_R$

$K_R$ is divided into left half $K_{R1}$ and right half $K_{R2}$, ($K_R = (K_{R1}, K_{R2})$) and the temporary variable, $Q_r$, is defined as:

$Q_r = K_{R1} \oplus K_{R2}$ for r=1,4,7,..., (r=3i+1; i=0,1,...)

$Q_r = K_{R1}$ for r=2,5,8,..., (r=3i+2; i=0,1,...)

$Q_r = K_{R2}$ for r=3,6,9,..., (r=3i+3; i=0,1,...)

Where $1 \leq r \leq (N/2)+4$. ($N \geq 4$, N:even)

Note: For FEAL-N, $K_R = (\phi, \phi)$ (64 zeros) and $Q_r = \phi$ (32 zeros).

(2) Processing of the left key $K_L$

Let $A_0$ be the left half of $K_L$ and let $B_0$ be the right half, i.e., $K_L = (A_0, B_0)$ and $D_0 = \phi$. Then calculate $K_i$( i = 0 to N+7 ) for r = 1 to $(N/2) + 4$, (N≥4, N:even)

$$D_r = A_{r-1}, \quad A_r = B_{r-1}$$

$$B_r = f_K(\alpha, \beta) = f_K(A_{r-1}, (B_{r-1} \oplus D_{r-1} \oplus Q_r))$$

$$K_{2(r-1)} = (B_{r0}, B_{r1}), \quad K_{2(r-1)+1} = (B_{r2}, B_{r3})$$

where $A_r, B_r$, $D_r$ and $Q_r$are auxiliary variables. $B_r = (B_{r0}, B_{r1}, B_{r2}, B_{r3})$. $B_{r0}, \ldots, B_{r3}$ are each 8 bits long. Function $f_K$ is the same as in FEAL-N. The key schedule of FEAL-NX is shown in Figure 2.

### 4.4 Key schedule of FEAL-N

The FEAL-N key schedule is equivalent to the FEAL-NX key schedule when $K_L$ is the 64-bit key of FEAL-N and $K_R$ is all zeros, where the temporary variable Qr=$\phi$.

Let $A_0$ be the left half of the 64-bit key and let $B_0$ be the right, i.e., the 64-bit key= $(A_0, B_0)$ and $D_0 = \phi$. Then calculate $K_i$( i = 0 to N+7 ) for r = 1 to $(N/2)+4$, ($N \geq 4$, N:even)

$$D_r = A_{r-1}, \quad A_r = B_{r-1}$$
$$B_r = f_K(\alpha, \beta) = f_K(A_{r-1}, (B_{r-1} \oplus D_{r-1}))$$
$$K_{2(r-1)} = (B_{r0}, B_{r1}), \quad K_{2(r-1)+1} = (B_{r2}, B_{r3})$$

## 5 Functions

### 5.1 Function $f$ (see also Figure 3)

$f(\alpha, \beta)$ is shortened to $f$. $\alpha$ and $\beta$ are divided as follows, where $\alpha_i$, and $\beta_i$ are 8-bits long. Functions $S_0$ and $S_1$ are defined in clause 5.3.

$$\alpha = (\alpha_0, \alpha_1, \alpha_2, \alpha_3), \quad \beta = (\beta_0, \beta_1).$$

$f = (f_0, f_1, f_2, f_3)$ are calculated in the sequence (1) to (8).

(1) $f_1 = \alpha_1 \oplus \beta_0$, (2) $f_2 = \alpha_2 \oplus \beta_1$
(3) $f_1 = f_1 \oplus \alpha_0$, (4) $f_2 = f_2 \oplus \alpha_3$
(5) $f_1 = S_1(f_1, f_2)$, (6) $f_2 = S_0(f_2, f_1)$
(7) $f_0 = S_0(\alpha_0, f_1)$, (8) $f_3 = S_1(\alpha_3, f_2)$

*Example in hex*: Inputs: $\alpha$ = `00FF FF00`, $\beta$ = `FFFF`,
Output: $f$ = `1004 1044`

### 5.2 Function $f_K$ (see also Figure 4)

Inputs of function $f_K$, $\alpha$ and $\beta$, are divided into four 8-bit blocks as:

$$\alpha = (\alpha_0, \alpha_1, \alpha_2, \alpha_3), \quad \beta = (\beta_0, \beta_1, \beta_2, \beta_3).$$

$f_K(\alpha, \beta)$ is shortened to $f$. $f_K = (f_{K0}, f_{K1}, f_{K2}, f_{K3})$ are calculated in the sequence (1) to (6).

(1) $f_{K1} = \alpha_1 \oplus \alpha_0$, (2) $f_{K2} = \alpha_2 \oplus \alpha_3$
(3) $f_{K1} = S_1(f_{K1}, (f_{K2} \oplus \beta_0))$, (4) $f_{K2} = S_0(f_{K2}, (f_{K1} \oplus \beta_1))$
(5) $f_{K0} = S_0(\alpha_0, (f_{K1} \oplus \beta_2))$, (6) $f_{K3} = S_1(\alpha_3, (f_{K2} \oplus \beta_3))$

*Example in hex*: Inputs: $\alpha$ = `0000 0000`, $\beta$ = `0000 0000`,
$f_K$ = `1004 1044`

### 5.3 Function $S$

$S_0$ and $S_1$ are defined as follows:

$$S_0(X_1, X_2)=\text{Rot2}((X_1 + X_2) \bmod 256)$$
$$S_1(X_1, X_2)=\text{Rot2}((X_1 + X_2 + 1) \bmod 256)$$

where $X_1 and X_2$ are 8-bit blocks and $Rot2(T)$ is the result of a 2-bit left rotation operation on 8-bit block, $T$.

*Example*: Suppose $X_1$ = 00010011, $X_2$ = 11110010 then

$$T = (X_1 + X_2 + 1) \bmod 256 = 00000110$$
$$S_1(X_1, X_2) = Rot2(T) = 00011000$$

## 6 Example of working data in hex

(1) Working data for FEAL-8

Parameters: Round number N=8 and non-use of key parity bits.

(a) Key: $K$ = 0123 4567 89AB CDEF

(b) Extended key:

$K_0$ = DF3B, $K_1$ = CA36, $K_2$ = F17C, $K_3$ = 1AEC

$K_4$ = 45A5, $K_5$ = B9C7, $K_6$ = 26EB, $K_7$ = AD25

$K_8$ = 8B2A, $K_9$ = ECB7, $K_{10}$ = AC50, $K_{11}$ = 9D4C

$K_{12}$ = 22CD, $K_{13}$ = 479B, $K_{14}$ = A8D5,$K_{15}$ = 0CB5

(c) Plaintext: $P$ = 0000 0000 0000 0000

(d) Ciphertext: $C$ = CEEF 2C86 F249 0752

(2) Working data for FEAL-4X/-8X/-16X/-32X/-64X

Parameters: N=4,8,16,32,64 and non-use of key parity bits.

(a) Key: $K$ =0123 4567 89AB CDEF 0123 4567 89AB CDEF

(b) Plaintext: $P$ = 0000 0000 0000 0000

(c) Ciphertext:

$C_4$ = DF7B EDD3 D59C 7C4B , $C_8$ = 92BE B65D 0E93 82FB

$C_{16}$ = 01A9 4383 EB19 BA07 , $C_{32}$ = 9C9B 5497 3DF6 85F8

$C_{64}$ = E2B0 F1C2 98EB 5030

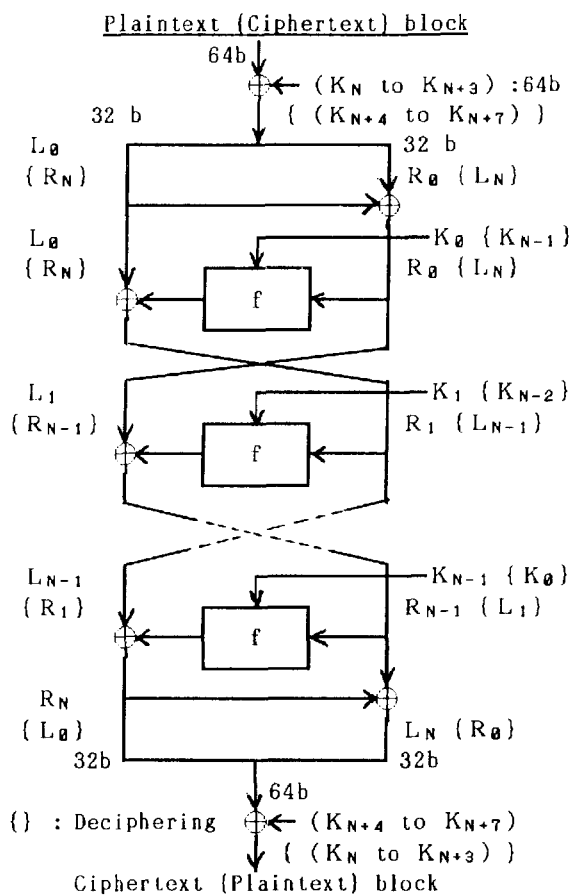



Fig.1 Data Randomization

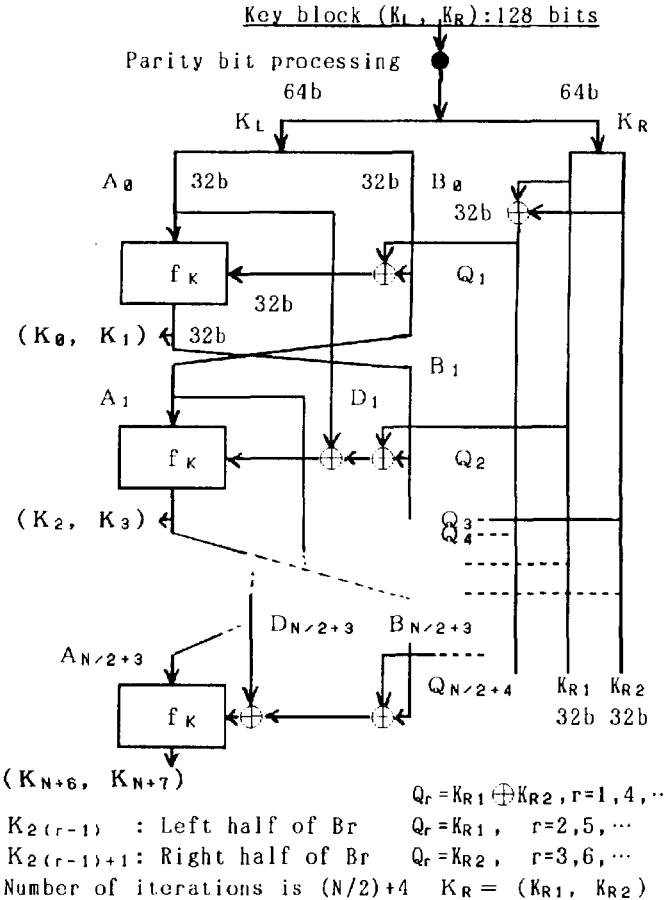
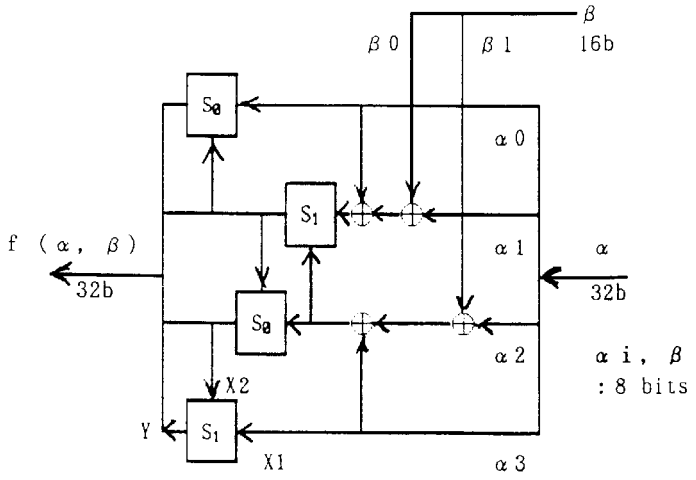



Fig.2 Key Schedule of FEAL (FEAL-NX)




Y = $S_0$( X1,X2 ) = Rot2((X1+X2) mod256)
Y = $S_1$( X1,X2 ) = Rot2((X1+X2+1) mod256)
Y: output(8 bits), X1/X2(8 bits): inputs,
Rot2(Y): a 2-bit left rotation on 8-bit data Y

Fig. 3 f-function

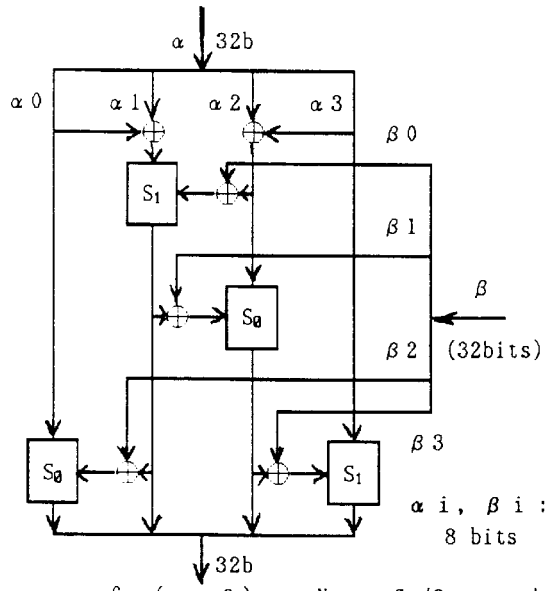



Fig. 4 $f_K$-function