Stream Cipher KCipher-2
(Document Version 1.2)

KDDI Corporation

Document History

Version 1.2	March 31, 2017	Editorial corrections These four corrections are intended to describe the behavior of the FSRs correctly. 1) The values shift in the direction from the fourth to zeroth register of FSR-A, and in the direction from the tenth to zeroth register of FSR-B. 2) The time on the right-hand side is one cycle before the time on the left-hand side of the definition formula of A and B. 3) The left-hand side indicates the i-th register and the right-hand side indicates the value of $(i+1)$-th register. Page 7: •$A_{t+i+1} = A_{t+i} \quad (i = 0, 1, 2, 3) \to A_{t+i+1} = A_{t+i+1} \quad (i = 0, 1, 2, 3)$ The value of $(i+1)$-th register of FSR-A at time t $(A_{t+(i+1)})$ is assigned to i-th register of FSR-A at time $(t+1)$ $(A_{(t+1)+i})$. Page 7: •$B_{t+i+1} = B_{t+i} \quad (i = 0, 1, \ldots, 9) \to B_{t+i+1} = B_{t+i+1} (i = 0, 1, \ldots, 9)$ The value of $(i+1)$-th register of FSR-B at time t $(B_{t+(i+1)})$ is assigned to i-th register of FSR-B at time $(t+1)$ $(B_{(t+1)+i})$. Page 8: •$A_{j+i} = A_{j+i-1} \quad (i = 0, 1, 2, 3) \to A_{j+i} = A_{j+i} \quad (i = 0, 1, 2, 3)$ The value of $(i+1)$-th register of FSR-A at time $(j-1)$ $(A_{(j-1)+(i+1)})$ is assigned to i-th register of FSR-A at time j (A_{j+i}). Page 8: •$B_{j+i} = B_{j+i-1} \quad (i = 0, 1, \ldots, 9) \to B_{j+i} = B_{j+i} \quad (i = 0, 1, \ldots, 9)$ The value of $(i+1)$-th register of FSR-B at time $(j-1)$ $(B_{(j-1)+(i+1)})$ is assigned to i-th register of FSR-B at time j (B_{j+i}).
Version 1.1	Feburuary 1, 2010	No changes (Only Japanese version was updated.)
Version 1.0	November 30, 2009	Initial release

1 Introduction

This documentation gives a detailed description of a stream cipher (keystream generator) KCipher-2. KCipher-2 uses two independent parameters as input, a 128-bit initial key and a 128-bit initial vector.

The document is organized as follows: In Sect.2, we describe the design rationale of KCipher-2. Next, we give some notations and definitions in Sect.3. In Sect.4, we describe the components and functions used in KCipher-2 in detail and show the schematic of KCipher-2. We present the processing step of KCipher-2 in Sect.5. Sect.6 provides methods to implement KCipher-2. Finally, we give several usage notes in Sect.7 and introduce products and systems using KCipher-2 in Sect.8.

2 Design Rationale of KCipher-2

A basic stream cipher uses several independent linear feedback shift registers (LFSRs) together with nonlinear functions in order to produce a keystream. Some stream ciphers use a general nonlinear function to clock one or more LFSR(s) irregularly. Various clock-controlled stream ciphers and attacks on them have been proposed. A5 is a well-known clock-controlled stream cipher designed to ensure the confidentiality of mobile communications.

The clock control mechanism of a stream cipher generally either controls LFSR clocking or shrinks or thins output. A clock control that shrinks or thins output reduces the performance of the stream cipher because some output bits are discarded. If one applies shrinking to a word-oriented stream cipher, the performance is markedly reduced. The bit-oriented clock control mechanism for updating an LFSR is also inefficient when the mechanism controls the LFSR for each register. On the other hand, a dynamic feedback control for an LFSR is an effective method for improving the security of stream ciphers. The stream cipher MICKEY[1] has a dynamic feedback control mechanism for a bit-wise LFSR. POMARANCH[2] uses a cascade jump controlled sequence generator to modify the feedback function.

KCipher-2 is a stream cipher that operates on words and has an efficient dynamic feedback control as irregular clocking. The basic idea of the design is to modify the mixing operation during the state update. Feedback polynomials for word-oriented LFSR are described with coefficients; multiplying an input word by a coefficient means mixing the words. A typical example is a LFSR of SNOW2.0[3]. Generally, the coefficients are selected such that the feedback polynomial is a primitive polynomial. We apply irregular clocking for this mixing operation, and the modification causes only a minimal decrease in the encryption/decryption speed. In other words, at least one FSR is irregularly clocked to dynamically modify the feedback function to the dynamic feedback controller that receives the outputs of the other FSR(s). For example, the feedback function is defined as

$$s_{t+a} = \alpha_0^{\{0,1\}} s_{t+b} \oplus \alpha_1^{\{0,1\}} s_{t+c} \oplus \alpha_2^{\{0,1\}} s_{t+d},$$

where {0, 1}s are selected by the dynamic feedback controller. The FSR controlled by the dynamic feedback controller is named dynamic feedback shift register (DFSR). The dynamic feedback control mechanism improves the security of a stream cipher because it changes the deterministic linear recurrence of some registers into a probabilistic recurrence. This property effectively protects against several attacks. An attacker has to obtain the linear recurrence of the keystream derived from the linear recurrence of some registers. By an irregular modification, the linear recurrence exists with a low probability. An attacker has to guess some inputs to the non-linear function for an attack; however, an irregular modification makes it impossible: the attacker has to guess the inputs to the dynamic feedback controller first. Thus, irregular modification of the feedback function improves the security of the stream cipher. We think that a dynamic feedback control mechanism is potentially effective against several attacks, not only existing attacks but also a novel attack. Above all, KCipher-2 achieves not only high performance similar to LFSR-based stream ciphers but also high security.

The keysteam generation speed of KCipher-2 is 4.97 cycle/byte on Pentium 4 series. Thus, the cipher is competitive against existing stream ciphers in the list of CRYPTREC. Furthermore, KCipher-2 was designed under considerations of two attacks on SNOW 2.0: algebraic attack and distinguishing attack. SNOW 2.0 is vulnerable against the two attacks; however, KCipher-2 has much resistance against the two attacks. In fact, no attack less than 2^{256} operations has been found on KCipher-2 so far. Thus, KCipher-2 achieves high security more than that of the existing stream ciphers. Detailed information is written in our evaluation report.

3 Preliminaries

3.1 Notations

$\oplus$	bitwise Exclusive-OR (XOR)
$\boxplus$	32-bit integer addition
0x	prefix for hexadecimal values
$GF(2^n)$	finite field of exactly 2^n values
$\gg n$	n-bit right shift in a 32-bit register
$\ll n$	n-bit left shift in a 32-bit register

3.2 Data Structure

The elemental-data size of KCipher-2 is 32-bit. In the algorithm, bit/byte order is big-endian, i.e. if the key and initialization vector are given as a sequence of bits/bytes, the first/leftmost bit/byte is the most significant of the corresponding data.

4 The Components and Functions of KCipher-2

In this section, we describe components and functions of the stream cipher algorithm KCipher-2 that has a dynamic feedback control mechanism.

KCipher-2 consists of two Feedback Shift Registers (FSRs), *FSR-A* and *FSR-B*, a non-linear function with four internal registers $R1$, $R2$, $L1$, and $L2$, and a dynamic feedback controller as shown in Fig.1. *FSR-B* is a dynamic feedback shift register. The size of each register is 32 bits.

4.1 Feedback Shift Registers

FSR-A has five registers, and *FSR-B* has eleven registers. Let β be the roots of the primitive polynomial $x^8 + x^7 + x^6 + x + 1 \in GF(2)[x]$. A byte string y denotes $(y_7, y_6, ..., y_1, y_0)$, where y_7 is the most significant bit and y_0 is the least significant bit. y is represented by $y = y_7\beta^7 + y_6\beta^6 + ... + y_1\beta + y_0$. In the same way, let γ, δ, ζ be the roots of the primitive polynomials,

$$x^8 + x^5 + x^3 + x^2 + 1 \in GF(2)[x],$$
$$x^8 + x^6 + x^3 + x^2 + 1 \in GF(2)[x],$$
$$x^8 + x^6 + x^5 + x^2 + 1 \in GF(2)[x],$$

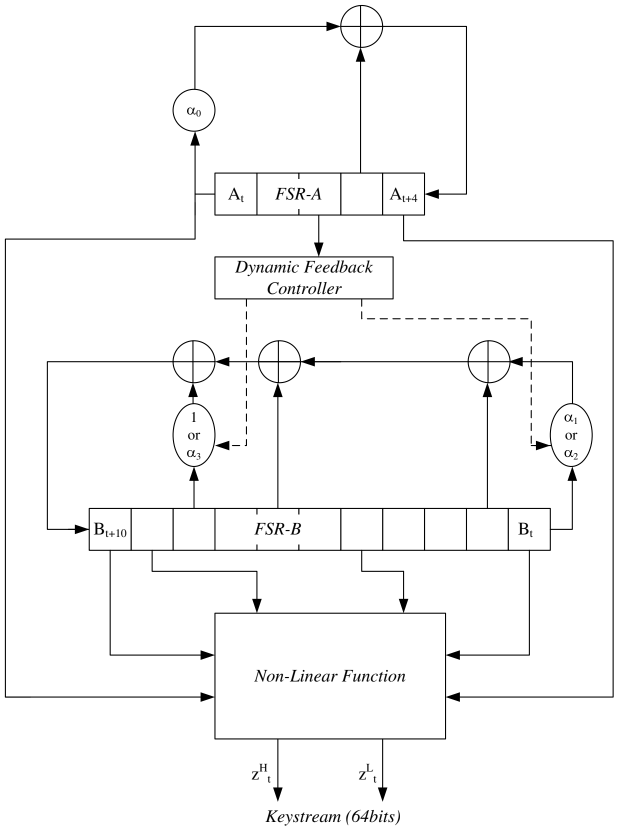

Fig. 1. Schematic Draw of KCipher-2

respectively. Let α_0 be the root of the irreducible polynomial of degree four

$$x^4 + \beta^{24}x^3 + \beta^3x^2 + \beta^{12}x + \beta^{71} \in GF(2^8)[x].$$

A 32-bit string Y denotes (Y_3, Y_2, Y_1, Y_0), where Y_i is a byte string and Y_3 is the most significant byte. Y is represented by $Y = Y_3\alpha_0^3 + Y_2\alpha_0^2 + Y_1\alpha_0 + Y_0$. Let α_1, α_2, α_3 be the roots of the irreducible polynomials of degree four

$$x^4 + \gamma^{230}x^3 + \gamma^{156}x^2 + \gamma^{93}x + \gamma^{29} \in GF(2^8)[x],$$
$$x^4 + \delta^{34}x^3 + \delta^{16}x^2 + \delta^{199}x + \delta^{248} \in GF(2^8)[x],$$
$$x^4 + \zeta^{157}x^3 + \zeta^{253}x^2 + \zeta^{56}x + \zeta^{16} \in GF(2^8)[x],$$

respectively.

The feedback polynomials $f_A(x)$, and $f_B(x)$ of *FSR-A* and *FSR-B*, respectively, are as follows;

$$f_A(x) = \alpha_0x^5 + x^2 + 1,$$
$$f_B(x) = (\alpha_1^{cl1_t} + \alpha_2^{1-cl1_t} - 1)x^{11} + x^{10} + x^5 + \alpha_3^{cl2_t}x^3 + 1.$$

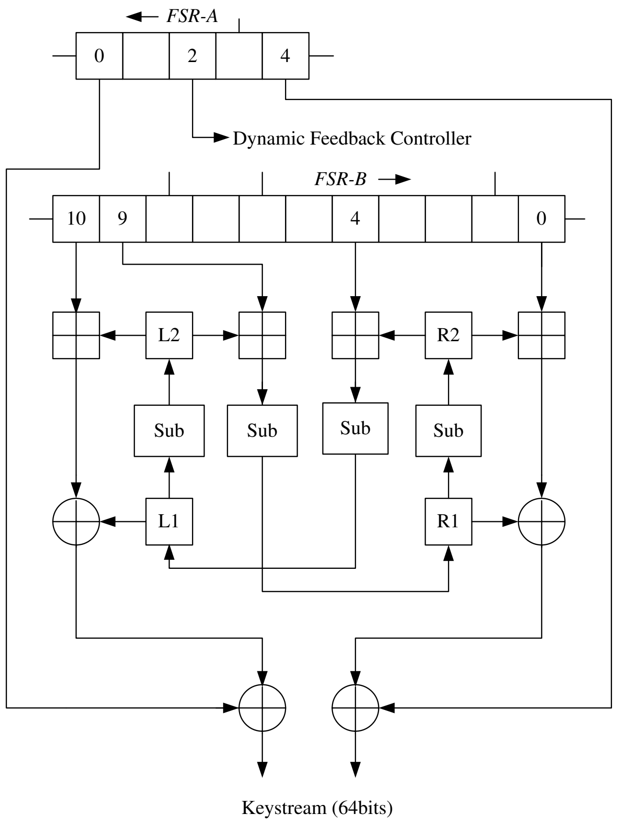

Fig. 2. Non-Linear Function of KCipher-2

Let $cl1$ and $cl2$ be the sequences describing the output of the dynamic feedback controller. The outputs at time t are defined in terms of some bits of *FSR-A*. Let A_x denote the output of *FSR-A* at time x, and $A_x[y] = \{0, 1\}$ denote the yth bit of A_x, where $A_x[31]$ is the most significant bit of A_x. Then $cl1$ and $cl2$ (called clock control bits) are described as $cl1_t = A_{t+2}[30]$, $cl2_t = A_{t+2}[31]$. Both $cl1_t$ and $cl2_t$ are binary variables; more precisely, $cl1_t = \{0, 1\}$, and $cl2_t = \{0, 1\}$. Stop-and-go clocking is effective in terms of computational cost, because no computation is required in the case of 0. However, the feedback function has no transformation for feedback registers with a probability 1/4 where all clockings are stop-and-go clockings. Thus, we use two types of clocking for the feedback function. *FSR-B* is defined by a primitive polynomial, where $cl2_t = 0$.

4.2 Nonlinear Function

The non-linear function of KCipher-2 is fed the values of two registers of *FSR-A* and four registers of *FSR-B* and that of internal registers $R1, R2, L1, L2$, and outputs 64 bits of the keystream every cycle. Fig.2 shows the non-linear function of KCipher-2. The nonlinear function includes four substitution steps that are indicated by *Sub*.

$$Sub: \ A \in GF(2^{32}) \mapsto B = Sub(A) \in GF(2^{32}).$$

The *Sub* step first divides the 32-bit input string into four 1-byte strings and applies a non-linear permutation to each byte using an 8-to-8 bit substitution, and then applies a 32-to-32 bit linear permutation. The 8-to-8 bit substitution is the same as s-boxes of AES [4], and the linear permutation is the same as AES *Mix Column* operation. We refer to Sect.6 for description of implementation of *Sub*.

8-to-8 substitution in *Sub*: The 8-to-8 bit substitution consists of two functions, g and f, and it is defined by a composite function $f \circ g$. The g calculates the multiplicative inverse modulo the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$ without 0x00, and 0x00 is transformed to itself (0x00).

$$g: \ a \in GF(2^8) \mapsto b = a^{-1} \in GF(2^8).$$

On the other hand, $f: \ a \in GF(2^8) \mapsto b \in GF(2^8)$ is an affine transformation defined by;

$$\begin{bmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 11111000 \\ 01111100 \\ 00111110 \\ 00011111 \\ 10001111 \\ 11000111 \\ 11100011 \\ 11110001 \end{bmatrix} \times \begin{bmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix},$$

where $a = (a_7, ..., a_0)$ is the input and $b = (b_7, ..., b_0)$ is an output, and a_0 and b_0 are the least significant bit (LSB).

32-to-32 linear permutation in *Sub*: Let C be an input 32-bit string denoted by a four 8-bit values (c_3, c_2, c_1, c_0). And let D be an output 32-bit string also defined by a for 8-bit values (d_3, d_2, d_1, d_0). The linear permutation $p: \ C \in GF(2^{32}) \mapsto D = p(C) \in GF(2^{32})$ is described as follows;

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

in $GF(2^8)$ of the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$.

5 Processes of KCipher-2

In this section, the detailed description of processes in KCipher-2 is provided. KCipher-2 has two processes, keystream output process and initialization process. After the initialization process, we run the keystream output process.

5.1 Keystream Output Process

Let keystream at time t be $Z_t = (z_t^H, z_t^L)$ (each z_t^x is a 32-bit value, and z_t^H is a higher string). The keystream z_t^H, z_t^L is calculated as follows:

$$z_t^L = B_t \boxplus R2_t \oplus R1_t \oplus A_{t+4},$$
$$z_t^H = B_{t+10} \boxplus L2_t \oplus L1_t \oplus A_t,$$

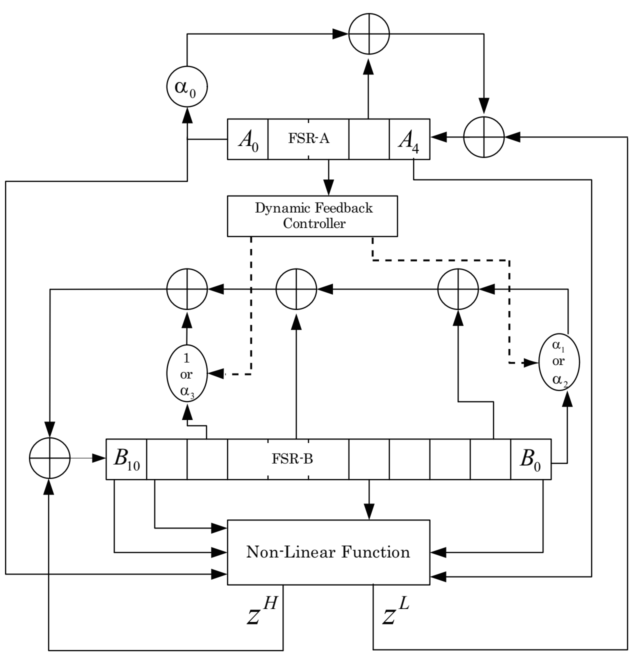

Fig. 3. Initialization for KCipher-2

where A_x and B_x denote outputs of *FSR-A* and *FSR-B* at time x, and $R1_x$, $R2_x$, $L1_x$, and $L2_x$ denote the internal registers at time x.

Finally, the internal registers are updated as follows;

$$R1_{t+1} = Sub(L2_t \boxplus B_{t+9}), \quad R2_{t+1} = Sub(R1_t),$$
$$L1_{t+1} = Sub(R2_t \boxplus B_{t+4}), \quad L2_{t+1} = Sub(L1_t),$$
$$A_{t+i+1} = \begin{cases} A_{t+i+1} & (i = 0, 1, 2, 3), \\ A_{t+3} \oplus \alpha_0 A_t & (i = 4) \end{cases}$$
$$B_{t+i+1} = \begin{cases} B_{t+i+1} & (i = 0, 1, \ldots, 9) \\ (\alpha_1^{cl1_t} + \alpha_2^{1-cl1_t} - 1)B_t \oplus B_{t+1} \oplus B_{t+6} \oplus \alpha_3^{cl2_t} B_{t+8} & (i = 10) \end{cases}$$

where $Sub(X)$ is an output of the *Sub* step for X. The set of $\{B_t, B_{t+3}, B_{t+8}, B_{t+10}\}$ is a *Full Positive Difference Set* (FPDS)[5]. We refer to section Sect.6 for description of the function *Sub* and the finite field arithmetic involving the fixed elements, $\alpha_0, \alpha_1, \alpha_2$ and α_3 to implement KCipher-2.

5.2 Initialization Process

The initialization process of KCipher-2 consists of two steps, a key loading step and an internal state initialization step.

Key Loading Step: First, an initial internal state is generated from a 128-bit initial key, and a 128-bit initial vector (IV) by using the key scheduling algorithm. The key scheduling algorithm is similar to the round key generation function of AES and the algorithm extends the 128-bit initial key to 384 bits. The key scheduling algorithm for a 128-bit key is described as;

$$K_i = \begin{cases} IK_i & (0 \le i \le 3) \\ K_{i-4} \oplus Sub((K_{i-1} \ll 8) \oplus (K_{i-1} \gg 24)) \oplus Rcon[i/4-1] & (i = 4n) \\ K_{i-4} \oplus K_{i-1} & (i \ne 4n) \end{cases},$$

where $IK = (IK_0, IK_1, IK_2, IK_3)$ is the initial key, i is a positive integer $0 \le i \le 11$, and n is a positive integer. The function $Sub(X)$ in the key scheduling algorithm is the same as that in the non-linear function. This function is different from the round key generation function of AES, and the other part of the key scheduling algorithm is same as the AES round key generation. $Rcon[i]$ denotes $(x^i \; mod \; x^8 + x^4 + x^3 + x + 1, 0x00, 0x00, 0x00)$ and x is 0x02. Hence, $Rcon[0] = (\text{0x01}, \text{0x00}, \text{0x00}, \text{0x00})$ for $i = 4$, and $Rcon[1] = (\text{0x02}, \text{0x00}, \text{0x00}, \text{0x00})$ for $i = 8$. The internal state is initialized with K_i and $IV = (IV_0, IV_1, IV_2, IV_3)$ as follows:

$$\begin{aligned} &A_m = K_{4-m} \; (m = 0, ..., 4), \\ &B_0 = K_{10}, B_1 = K_{11}, B_2 = IV_0, B_3 = IV_1, \\ &B_4 = K_8, B_5 = K_9, B_6 = IV_2, B_7 = IV_3, \\ &B_8 = K_7, B_9 = K_5, B_{10} = K_6. \end{aligned}$$

The internal registers, $R1$, $R2$, $L1$, and $L2$ are set to 0x00.

Internal State Initialization Step: After the above processes, the cipher clocks 24 times ($j = 1, \ldots, 24$), updating the internal states as shown in Fig.3. The internal states are updated as follows:

$$\begin{aligned} &R1_j = Sub(L2_{j-1} \boxplus B_{j+8}), \quad R2_j = Sub(R1_{j-1}), \\ &L1_j = Sub(R2_{j-1} \boxplus B_{j+3}), \quad L2_j = Sub(L1_{j-1}), \\ &A_{j+i} = \begin{cases} A_{j+i} & (i = 0,1,2,3) \\ \alpha_0 A_{j-1} \oplus A_{j+2} \oplus z^L_{j-1} & (i = 4) \end{cases}, \\ &B_{j+i} = \begin{cases} B_{j+i} & (i = 0,1,\ldots,9) \\ (\alpha_1^{cl1_{j-1}} + \alpha_2^{1-cl1_{j-1}} - 1) B_{j-1} \oplus B_j \oplus B_{j+5} \oplus \alpha_3^{cl2_{j-1}} B_{j+7} \oplus z^H_{j-1} & (i = 10) \end{cases}. \end{aligned}$$

We refer to section Sect.6 for description of the function Sub and the finite field arithmetic involving the fixed elements, $\alpha_0, \alpha_1, \alpha_2$ and α_3 to implement KCipher-2.

6 Implementation of KCipher-2

6.1 Fast Implementation using Lookup Tables

Multiplicative operations with $\alpha_0, \alpha_1, \alpha_2$ and α_3: Multiplicative operations with 32-bit values and fixed elements, $\alpha_0, \alpha_1, \alpha_2, \alpha_3$ are executed in *FSR-A* and *FSR-B*. The operations can be done rapidly by using lookup tables. Let w be a 32-bit values. And let `alpha_i[j]` be a j-th element ($0 \le j \le 255$) in a lookup table `alpha_i[256]` ($i = 0,1,2,3$). A multiplication, $\alpha_i w$, is obtained by

$$\alpha_i w = (w \ll 8) \oplus \texttt{alpha\_i}[(w \gg 24)].$$

The tables `alpha_0[256]`, `alpha_1[256]`, `alpha_2[256]` and `alpha_3[256]` are provided in Appendix A. For example, we have the following result for a multiplication with α_0 and $w = 0x01234567$.

$$\begin{aligned}\alpha_0 \cdot (0x01234567) &= ((0x01234567) \ll 8) \oplus \texttt{alpha\_0}[0x01]\\ &= 0x23456700 \oplus 0xB6086D1A\\ &= 0x954D0A1A.\end{aligned}$$

Lookup Tables for *Sub*: 32-to-32 *Sub* operations in the nonlinear function can be done rapidly by using Lookup tables. Let a 32-bit A be an input of *Sub*. And let `T_i[j]` be a j-th element $(0 \le j \le 255)$ in a lookup table `T_i[256]` $(i = 0, 1, 2, 3)$. An output $Sub(A)$ is obtained by

$$Sub(A) = \texttt{T\_0}[A \,\&\, 0xFF] \oplus \texttt{T\_1}[(A \gg 8) \,\&\, 0xFF] \oplus \texttt{T\_2}[(A \gg 16) \,\&\, 0xFF] \oplus \texttt{T\_3}[A \gg 24],$$

where & denotes the bitwise AND operations. The tables `T_0[256]`, `T_1[256]`, `T_2[256]` and `T_3[256]` are provided in Appendix B. For example, we have the following result for $A = 0x01234567$.

$$\begin{aligned}Sub(0x01234567) &= \texttt{T\_0}[0x67] \oplus \texttt{T\_1}[0x45] \oplus \texttt{T\_2}[0x23] \oplus \texttt{T\_3}[0x01]\\ &= 0x94858511 \oplus 0x6E6EDCB2 \oplus 0x264C6A26 \oplus 0xF8847C7C\\ &= 0x24234FF9.\end{aligned}$$

6.2 Compact Implementation without Lookup Tables

Here we present a compact implementation without lookup tables for α_i and *Sub*.

Multiplication with $\alpha_0, \alpha_1, \alpha_2$ and α_3: Let $x = (x_{31}, \ldots, x_0)$ be a 32-bit variable. And let $y(i) = (y(i)_{31}, \ldots, y(i)_0)$ be also a variable representing the result of multiplication $y(i) = \alpha_i x$ for $i = 0, 1, 2, 3$. Then, each bit in $y(i)$ can be obtained as follows:

- For the case of $y(0)$.

$$\begin{aligned}
y(0)_0 &= x_{28} \oplus x_{30}, & y(0)_{16} &= x_8 \oplus x_{29} \oplus x_{30},\\
y(0)_1 &= x_{24} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(0)_{17} &= x_9 \oplus x_{29} \oplus x_{31},\\
y(0)_2 &= x_{25} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(0)_{18} &= x_{10} \oplus x_{30},\\
y(0)_3 &= x_{24} \oplus x_{26} \oplus x_{30} \oplus x_{31}, & y(0)_{19} &= x_{11} \oplus x_{24} \oplus x_{31},\\
y(0)_4 &= x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{31}, & y(0)_{20} &= x_{12} \oplus x_{25},\\
y(0)_5 &= x_{25} \oplus x_{26} \oplus x_{28}, & y(0)_{21} &= x_{13} \oplus x_{26},\\
y(0)_6 &= x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(0)_{22} &= x_{14} \oplus x_{27} \oplus x_{29} \oplus x_{30},\\
y(0)_7 &= x_{27} \oplus x_{29} \oplus x_{31}, & y(0)_{23} &= x_{15} \oplus x_{28} \oplus x_{29} \oplus x_{31},\\
y(0)_8 &= x_0 \oplus x_{24} \oplus x_{26} \oplus x_{28}, & y(0)_{24} &= x_{16} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{31},\\
y(0)_9 &= x_1 \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29}, & y(0)_{25} &= x_{17} \oplus x_{24} \oplus x_{25} \oplus x_{29} \oplus x_{31},\\
y(0)_{10} &= x_2 \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(0)_{26} &= x_{18} \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{30},\\
y(0)_{11} &= x_3 \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(0)_{27} &= x_{19} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{31},\\
y(0)_{12} &= x_4 \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(0)_{28} &= x_{20} \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{28},\\
y(0)_{13} &= x_5 \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(0)_{29} &= x_{21} \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{29},\\
y(0)_{14} &= x_6 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{30} \oplus x_{31}, & y(0)_{30} &= x_{22} \oplus x_{27} \oplus x_{29} \oplus x_{30} \oplus x_{31},\\
y(0)_{15} &= x_7 \oplus x_{25} \oplus x_{27} \oplus x_{31}, & y(0)_{31} &= x_{23} \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{30}.
\end{aligned}$$

- For the case of $y(1)$.

$$\begin{aligned}
y(1)_0 &= x_{27} \oplus x_{29} \oplus x_{31}, & y(1)_{16} &= x_8 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{29} \oplus x_{30},\\
y(1)_1 &= x_{24} \oplus x_{28} \oplus x_{30}, & y(1)_{17} &= x_9 \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{30} \oplus x_{31},\\
y(1)_2 &= x_{24} \oplus x_{25} \oplus x_{27}, & y(1)_{18} &= x_{10} \oplus x_{24} \oplus x_{25} \oplus x_{28} \oplus x_{30} \oplus x_{31},\\
y(1)_3 &= x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{31}, & y(1)_{19} &= x_{11} \oplus x_{27} \oplus x_{30} \oplus x_{31},\\
y(1)_4 &= x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(1)_{20} &= x_{12} \oplus x_{24} \oplus x_{28} \oplus x_{31},\\
y(1)_5 &= x_{24} \oplus x_{26} \oplus x_{28} \oplus x_{30}, & y(1)_{21} &= x_{13} \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{30},\\
y(1)_6 &= x_{25} \oplus x_{27} \oplus x_{29} \oplus x_{31}, & y(1)_{22} &= x_{14} \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{31},\\
y(1)_7 &= x_{26} \oplus x_{28} \oplus x_{30}, & y(1)_{23} &= x_{15} \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29},\\
y(1)_8 &= x_0 \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{30}, & y(1)_{24} &= x_{16} \oplus x_{25} \oplus x_{27} \oplus x_{28},\\
y(1)_9 &= x_1 \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{31}, & y(1)_{25} &= x_{17} \oplus x_{26} \oplus x_{28} \oplus x_{29},\\
y(1)_{10} &= x_2 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(1)_{26} &= x_{18} \oplus x_{25} \oplus x_{28} \oplus x_{29} \oplus x_{30},\\
y(1)_{11} &= x_3 \oplus x_{24} \oplus x_{29} \oplus x_{31}, & y(1)_{27} &= x_{19} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31},\\
y(1)_{12} &= x_4 \oplus x_{24} \oplus x_{25} \oplus x_{30}, & y(1)_{28} &= x_{20} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31},\\
y(1)_{13} &= x_5 \oplus x_{24} \oplus x_{27} \oplus x_{30} \oplus x_{31}, & y(1)_{29} &= x_{21} \oplus x_{24} \oplus x_{25} \oplus x_{29} \oplus x_{30} \oplus x_{31},\\
y(1)_{14} &= x_6 \oplus x_{24} \oplus x_{25} \oplus x_{28} \oplus x_{31}, & y(1)_{30} &= x_{22} \oplus x_{25} \oplus x_{26} \oplus x_{30} \oplus x_{31},\\
y(1)_{15} &= x_7 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{29}, & y(1)_{31} &= x_{23} \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{31}.
\end{aligned}$$

- For the case of $y(2)$.

$$\begin{aligned}
y(2)_0 &= x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{31}, & y(2)_{16} &= x_8 \oplus x_{25} \oplus x_{26} \oplus x_{29} \oplus x_{30} \oplus x_{31},\\
y(2)_1 &= x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(2)_{17} &= x_9 \oplus x_{26} \oplus x_{27} \oplus x_{30} \oplus x_{31},\\
y(2)_2 &= x_{26} \oplus x_{28} \oplus x_{30}, & y(2)_{18} &= x_{10} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30},\\
y(2)_3 &= x_{25} \oplus x_{28}, & y(2)_{19} &= x_{11} \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28},\\
y(2)_4 &= x_{24} \oplus x_{26} \oplus x_{29}, & y(2)_{20} &= x_{12} \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29},\\
y(2)_5 &= x_{25} \oplus x_{27} \oplus x_{30}, & y(2)_{21} &= x_{13} \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{29} \oplus x_{30},\\
y(2)_6 &= x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{29}, & y(2)_{22} &= x_{14} \oplus x_{24} \oplus x_{27} \oplus x_{28} \oplus x_{29},\\
y(2)_7 &= x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{30}, & y(2)_{23} &= x_{15} \oplus x_{24} \oplus x_{25} \oplus x_{28} \oplus x_{29} \oplus x_{30},\\
y(2)_8 &= x_0 \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{30}, & y(2)_{24} &= x_{16} \oplus x_{24} \oplus x_{26} \oplus x_{29} \oplus x_{31},\\
y(2)_9 &= x_1 \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{31}, & y(2)_{25} &= x_{17} \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{30},\\
y(2)_{10} &= x_2 \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(2)_{26} &= x_{18} \oplus x_{25} \oplus x_{28} \oplus x_{29},\\
y(2)_{11} &= x_3 \oplus x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{29} \oplus x_{31}, & y(2)_{27} &= x_{19} \oplus x_{24} \oplus x_{30} \oplus x_{31},\\
y(2)_{12} &= x_4 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29} \oplus x_{30}, & y(2)_{28} &= x_{20} \oplus x_{24} \oplus x_{25} \oplus x_{31},\\
y(2)_{13} &= x_5 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(2)_{29} &= x_{21} \oplus x_{25} \oplus x_{26},\\
y(2)_{14} &= x_6 \oplus x_{24} \oplus x_{25} \oplus x_{28} \oplus x_{31}, & y(2)_{30} &= x_{22} \oplus x_{24} \oplus x_{27} \oplus x_{29} \oplus x_{31},\\
y(2)_{15} &= x_7 \oplus x_{25} \oplus x_{26} \oplus x_{29}, & y(2)_{31} &= x_{23} \oplus x_{25} \oplus x_{28} \oplus x_{30}.
\end{aligned}$$

- For the case of $y(3)$.

$$\begin{aligned}
y(3)_0 &= x_{24} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{31}, & y(3)_{16} &= x_8 \oplus x_{24} \oplus x_{26},\\
y(3)_1 &= x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29}, & y(3)_{17} &= x_9 \oplus x_{25} \oplus x_{27},\\
y(3)_2 &= x_{26} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(3)_{18} &= x_{10} \oplus x_{28},\\
y(3)_3 &= x_{24} \oplus x_{27} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(3)_{19} &= x_{11} \oplus x_{24} \oplus x_{29},\\
y(3)_4 &= x_{25} \oplus x_{28} \oplus x_{30} \oplus x_{31}, & y(3)_{20} &= x_{12} \oplus x_{24} \oplus x_{25} \oplus x_{30},\\
y(3)_5 &= x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29}, & y(3)_{21} &= x_{13} \oplus x_{25} \oplus x_{31},\\
y(3)_6 &= x_{25} \oplus x_{26} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(3)_{22} &= x_{14} \oplus x_{24},\\
y(3)_7 &= x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{30} \oplus x_{31}, & y(3)_{23} &= x_{15} \oplus x_{25},\\
y(3)_8 &= x_0 \oplus x_{26} \oplus x_{29} \oplus x_{30}, & y(3)_{24} &= x_{16} \oplus x_{24} \oplus x_{26} \oplus x_{28} \oplus x_{29},\\
y(3)_9 &= x_1 \oplus x_{24} \oplus x_{27} \oplus x_{30} \oplus x_{31}, & y(3)_{25} &= x_{17} \oplus x_{25} \oplus x_{27} \oplus x_{29} \oplus x_{30},\\
y(3)_{10} &= x_2 \oplus x_{24} \oplus x_{25} \oplus x_{26} \oplus x_{28} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(3)_{26} &= x_{18} \oplus x_{24} \oplus x_{29} \oplus x_{30} \oplus x_{31},\\
y(3)_{11} &= x_3 \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{29} \oplus x_{30} \oplus x_{31}, & y(3)_{27} &= x_{19} \oplus x_{25} \oplus x_{30} \oplus x_{31},\\
y(3)_{12} &= x_4 \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{30} \oplus x_{31}, & y(3)_{28} &= x_{20} \oplus x_{26} \oplus x_{31},\\
y(3)_{13} &= x_5 \oplus x_{25} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{30} \oplus x_{31}, & y(3)_{29} &= x_{21} \oplus x_{26} \oplus x_{27} \oplus x_{28} \oplus x_{29},\\
y(3)_{14} &= x_6 \oplus x_{24} \oplus x_{27} \oplus x_{28} \oplus x_{30} \oplus x_{31}, & y(3)_{30} &= x_{22} \oplus x_{24} \oplus x_{26} \oplus x_{27} \oplus x_{30},\\
y(3)_{15} &= x_7 \oplus x_{25} \oplus x_{28} \oplus x_{29} \oplus x_{31}, & y(3)_{31} &= x_{23} \oplus x_{25} \oplus x_{27} \oplus x_{28} \oplus x_{31}.
\end{aligned}$$

Function *Sub***:** The 32-to-32 function *Sub* in the nonlinear function can be also implemented without lookup tables. *Sub* consists of two steps, Substitution step and Permutation step. In Substitution step, operations in the subfield $GF(2^4)$ of $GF(2^8)$ are required for the compact implementation. First, we thus describe operations in the subfield.

- **Multiplications in $GF(2^4)$:** Let $x = (x_3, x_2, x_1, x_0)$ and $y = (y_3, y_2, y_1, y_0)$ be 4-bit variables. And let $z = (z_3, z_2, z_1, z_0)$ be also a 4-bit variable representing the result of multiplication $z = x \times_4 y$, where $\times_4$ is a operator of multiplication in $GF(2^4)$. Then, $z = x \times_4 y$ is obtained as follows:
  1. Let $t_6, \ldots, t_0$ be local 1-bit variables. We obtain

$$\begin{aligned} t_6 &= (x_3 \& y_3), \\ t_5 &= (x_3 \& y_2) \oplus (x_2 \& y_3), \\ t_4 &= (x_3 \& y_1) \oplus (x_2 \& y_2) \oplus (x_1 \& y_3), \\ t_3 &= (x_3 \& y_0) \oplus (x_2 \& y_1) \oplus (x_1 \& y_2) \oplus (x_0 \& y_3), \end{aligned} \qquad \begin{aligned} t_2 &= (x_2 \& y_0) \oplus (x_1 \& y_1) \oplus (x_0 \& y_2), \\ t_1 &= (x_1 \& y_0) \oplus (x_0 \& y_1), \\ t_0 &= (x_0 \& y_0), \end{aligned}$$

     where & denotes a bitwise AND operation.
  2. From $t_6, \ldots, t_0$, we obtain z as follows:

$$\begin{aligned} z_3 &= t_3 \oplus t_6, & z_1 &= t_1 \oplus t_4 \oplus t_5, \\ z_2 &= t_2 \oplus t_5 \oplus t_6, & z_0 &= t_0 \oplus t_4. \end{aligned}$$

- **Multiplicative inverse in $GF(2^4)$:** Let $x = (x_3, x_2, x_1, x_0)$ be a 4-bit variable. And let $y = (y_3, y_2, y_1, y_0)$ be also a 4-bit variable representing $y = x^{-1} \in GF(2^4)$. Then, y is obtained from the following mappings.

$x \mapsto y$	$x \mapsto y$
$(0,0,0,0) \mapsto (0,0,0,0)$	$(1,0,0,0) \mapsto (1,1,1,1)$
$(0,0,0,1) \mapsto (0,0,0,1)$	$(1,0,0,1) \mapsto (0,0,1,0)$
$(0,0,1,0) \mapsto (1,0,0,1)$	$(1,0,1,0) \mapsto (1,1,0,0)$
$(0,0,1,1) \mapsto (1,1,1,0)$	$(1,0,1,1) \mapsto (0,1,0,1)$
$(0,1,0,0) \mapsto (1,1,0,1)$	$(1,1,0,0) \mapsto (1,0,1,0)$
$(0,1,0,1) \mapsto (1,0,1,1)$	$(1,1,0,1) \mapsto (0,1,0,0)$
$(0,1,1,0) \mapsto (0,1,1,1)$	$(1,1,1,0) \mapsto (0,0,1,1)$
$(0,1,1,1) \mapsto (0,1,1,0)$	$(1,1,1,1) \mapsto (1,0,0,0)$

With the above operations in $GF(2^4)$, Substitution step and Permutations step are implemented as follows:

- **Substitution step:** Substitution step is an 8-to-8-bit function. Let $x = (x_7, \ldots, x_0)$ be an input and let $y = (y_7, \ldots, y_0)$ be an output. Then, Substitution step is implemented as follows.
  1. Let $a = (a_7, \ldots, a_0)$ be an 8-bit local variable. We obtain

$$\begin{aligned} a_7 &= x_7 \oplus x_5, & a_3 &= x_7 \oplus x_6 \oplus x_2 \oplus x_1, \\ a_6 &= x_7 \oplus x_5 \oplus x_3 \oplus x_2, & a_2 &= x_6 \oplus x_4 \oplus x_1, \\ a_5 &= x_7 \oplus x_6 \oplus x_4 \oplus x_1, & a_1 &= x_3 \oplus x_1, \\ a_4 &= x_6 \oplus x_5 \oplus x_4, & a_0 &= x_7 \oplus x_6 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_0. \end{aligned}$$

  2. Let (b_3, b_2, b_1, b_0) be a 4-bit local variable. We obtain

$$\begin{aligned} b_3 &= a_7 \oplus a_3, & b_1 &= a_5 \oplus a_1, \\ b_2 &= a_6 \oplus a_2, & b_0 &= a_4 \oplus a_0. \end{aligned}$$

3. Let (c_3, c_2, c_1, c_0) be a 4-bit local variable. We execute a multiplication $(c_3, c_2, c_1, c_0) = (a_3, a_2, a_1, a_0) \times_4 (b_3, b_2, b_1, b_0)$, where $\times_4$ denotes a multiplicative operation over $GF(2^4)$.
4. Let (d_3, d_2, d_1, d_0) be a 4-bit local variable. We execute a multiplication $(d_3, d_2, d_1, d_0) = (a_7, a_6, a_5, a_4) \times_4 (a_7, a_6, a_5, a_4)$, where $\times_4$ denotes a multiplicative operation over $GF(2^4)$.
5. Let (e_3, e_2, e_1, e_0) be a 4-bit local variable. We execute a multiplication $(e_3, e_2, e_1, e_0) = (1, 0, 0, 1) \times_4 (d_3, d_2, d_1, d_0)$, where $\times_4$ denotes a multiplicative operation over $GF(2^4)$.
6. Let (f_3, f_2, f_1, f_0) be a 4-bit local variable. We obtain
$$\begin{aligned} f_3 &= c_3 \oplus e_3, & f_1 &= c_1 \oplus e_1, \\ f_2 &= c_2 \oplus e_2, & f_0 &= c_0 \oplus e_0. \end{aligned}$$
7. Let (g_3, g_2, g_1, g_0) be a 4-bit local variable. We calculate the multiplicative inverse $(g_3, g_2, g_1, g_0) = (f_3, f_2, f_1, f_0)^{-1} \in GF(2^4)$.
8. Let (h_3, h_2, h_1, h_0) be a 4-bit local variable. We execute a multiplication $(h_3, h_2, h_1, h_0) = (b_3, b_2, b_1, b_0) \times_4 (g_3, g_2, g_1, g_0)$, where $\times_4$ denotes a multiplicative operation over $GF(2^4)$.
9. Let (h_7, h_6, h_5, h_4) be a 4-bit local variable. We execute a multiplication $(h_7, h_6, h_5, h_4) = (a_7, a_6, a_5, a_4) \times_4 (g_3, g_2, g_1, g_0)$, where $\times_4$ denotes a multiplicative operation over $GF(2^4)$.
10. Let $(i_7, \ldots, i_0)$ be 8-bit local variable. We obtain
$$\begin{aligned} i_7 &= h_3 \oplus 1, & i_3 &= h_7, \\ i_6 &= h_6 \oplus h_4 \oplus h_2 \oplus 1, & i_2 &= h_5 \oplus h_2, \\ i_5 &= h_7 \oplus h_5 \oplus h_1, & i_1 &= h_7 \oplus h_3 \oplus h_1 \oplus 1, \\ i_4 &= h_5 \oplus h_2 \oplus h_0, & i_0 &= h_4 \oplus h_2. \end{aligned}$$
11. We obtain the output of Substitution step as follows:
$$\begin{aligned} y_7 &= i_5 \oplus i_2, & y_3 &= i_7 \oplus i_5 \oplus i_4 \oplus i_1, \\ y_6 &= i_7 \oplus i_6 \oplus i_5 \oplus i_3 \oplus i_2 \oplus i_1, & y_2 &= i_6 \oplus i_5 \oplus i_4 \oplus i_1, \\ y_5 &= i_7 \oplus i_5 \oplus i_2, & y_1 &= i_7 \oplus i_5 \oplus i_4, \\ y_4 &= i_6 \oplus i_4 \oplus i_3 \oplus i_1, & y_0 &= i_6 \oplus i_4 \oplus i_0. \end{aligned}$$

– **Permutation step:** Permutation step is a 32-to-32-bit function. Let $x = (x_{31}, \ldots, x_0)$ be an input and let $y = (y_{31}, \ldots, y_0)$ be an output. Then, Permutation step is implemented as follows.

$$\begin{aligned}
y_{31} &= x_{15} \oplus x_{23} \oplus x_7 \oplus x_6 \oplus x_{30}, & y_{15} &= x_{31} \oplus x_7 \oplus x_{23} \oplus x_{22} \oplus x_{14}, \\
y_{30} &= x_{14} \oplus x_{22} \oplus x_6 \oplus x_5 \oplus x_{29}, & y_{14} &= x_{30} \oplus x_6 \oplus x_{22} \oplus x_{21} \oplus x_{13}, \\
y_{29} &= x_{13} \oplus x_{21} \oplus x_5 \oplus x_4 \oplus x_{28}, & y_{13} &= x_{29} \oplus x_5 \oplus x_{21} \oplus x_{20} \oplus x_{12}, \\
y_{28} &= x_{12} \oplus x_{20} \oplus x_4 \oplus x_3 \oplus x_7 \oplus x_{27} \oplus x_{31}, & y_{12} &= x_{28} \oplus x_4 \oplus x_{20} \oplus x_{19} \oplus x_{23} \oplus x_{11} \oplus x_{15}, \\
y_{27} &= x_{11} \oplus x_{19} \oplus x_3 \oplus x_2 \oplus x_7 \oplus x_{26} \oplus x_{31}, & y_{11} &= x_{27} \oplus x_3 \oplus x_{19} \oplus x_{18} \oplus x_{23} \oplus x_{10} \oplus x_{15}, \\
y_{26} &= x_{10} \oplus x_{18} \oplus x_2 \oplus x_1 \oplus x_{25}, & y_{10} &= x_{26} \oplus x_2 \oplus x_{18} \oplus x_{17} \oplus x_9, \\
y_{25} &= x_9 \oplus x_{17} \oplus x_1 \oplus x_0 \oplus x_7 \oplus x_{24} \oplus x_{31}, & y_9 &= x_{25} \oplus x_1 \oplus x_{17} \oplus x_{16} \oplus x_{23} \oplus x_8 \oplus x_{15}, \\
y_{24} &= x_8 \oplus x_{16} \oplus x_0 \oplus x_7 \oplus x_{31}, & y_8 &= x_{24} \oplus x_0 \oplus x_{16} \oplus x_{23} \oplus x_{15}, \\
y_{23} &= x_7 \oplus x_{15} \oplus x_{31} \oplus x_{30} \oplus x_{22}, & y_7 &= x_{23} \oplus x_{31} \oplus x_{15} \oplus x_{14} \oplus x_6, \\
y_{22} &= x_6 \oplus x_{14} \oplus x_{30} \oplus x_{29} \oplus x_{21}, & y_6 &= x_{22} \oplus x_{30} \oplus x_{14} \oplus x_{13} \oplus x_5, \\
y_{21} &= x_5 \oplus x_{13} \oplus x_{29} \oplus x_{28} \oplus x_{20}, & y_5 &= x_{21} \oplus x_{29} \oplus x_{13} \oplus x_{12} \oplus x_4, \\
y_{20} &= x_4 \oplus x_{12} \oplus x_{28} \oplus x_{27} \oplus x_{31} \oplus x_{19} \oplus x_{23}, & y_4 &= x_{20} \oplus x_{28} \oplus x_{12} \oplus x_{11} \oplus x_{15} \oplus x_3 \oplus x_7, \\
y_{19} &= x_3 \oplus x_{11} \oplus x_{27} \oplus x_{26} \oplus x_{31} \oplus x_{18} \oplus x_{23}, & y_3 &= x_{19} \oplus x_{27} \oplus x_{11} \oplus x_{10} \oplus x_{15} \oplus x_2 \oplus x_7, \\
y_{18} &= x_2 \oplus x_{10} \oplus x_{26} \oplus x_{25} \oplus x_{17}, & y_2 &= x_{18} \oplus x_{26} \oplus x_{10} \oplus x_9 \oplus x_1, \\
y_{17} &= x_1 \oplus x_9 \oplus x_{25} \oplus x_{24} \oplus x_{31} \oplus x_{16} \oplus x_{23}, & y_1 &= x_{17} \oplus x_{25} \oplus x_9 \oplus x_8 \oplus x_{15} \oplus x_0 \oplus x_7, \\
y_{16} &= x_0 \oplus x_8 \oplus x_{24} \oplus x_{31} \oplus x_{23}, & y_0 &= x_{16} \oplus x_{24} \oplus x_8 \oplus x_{15} \oplus x_7.
\end{aligned}$$

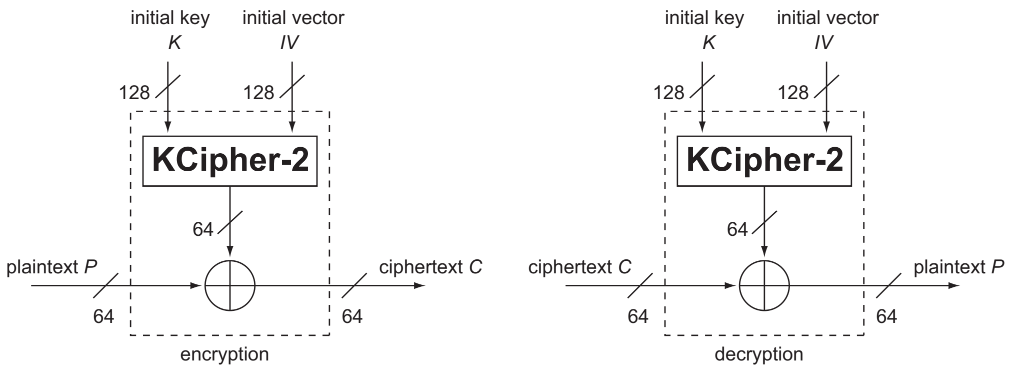

Fig. 4. Encryption and Decryption using KCipher-2

7 Usage Notes

7.1 How to Use Keys and Initial Vectors

The maximum number of cycles for KCipher-2 without re-initializing and re-keying is 2^{58} cycles (2^{64} keystream bits). The initial key and the initial vector will be updated after 2^{58} cycles.

7.2 Encryption and Decryption

To execute encryption and decryption by using KCipher-2, we first divide the plaintext data into 64-bit blocks. Then XOR pieces of plaintext and 64-bit output blocks (keystream) generated by the initial key and initial vector, as shown in Fig.4. The decryption can be implemented by the same manner.

7.3 Version Information

We describe version information of KCipher-2 in Appendix C.

7.4 IPR and License Information

Information about IPR and license of KCipher-2 is as follows.

- All intellectual property lights related to KCipher-2 belong to KDDI corporation.
- Evaluators of CRYPTREC can use KCipher-2 without any license fee.
- KCipher-2 will be provided to any governmental organization at a reasonable price and reasonable proviso.

8 Products and Systems of KCipher-2

KDDI R & D Laboratories Inc. has produced a software development kit (SDK) of KCipher-2. KCipher-2 has been used for the following systems/applications.

- Mobile-Phone Communication System for a Governmental Organization (2,000 licenses)
- Location Management System for a Governmental Organization (5,000 licenses)
- Web-based Groupware (1,000 licenses)
- Multimedia Contents Player for Consumer Application (about one million users)

References

1. S. Babbage and M. Dodd, “The stream cipher MICKEY-128 2.0.” The eSTREAM Project, 2006. available at `http://www.ecrypt.eu.org/stream/p3ciphers/mickey/mickey_p3.pdf`.
2. C. Jansen, T. Helleseth, and A. Kolosha, “Cascade jump controlled sequence generator and pomaranch stream cipher.” The eSTREAM Project, 2006. available at `http://www.ecrypt.eu.org/stream/p3ciphers/pomaranch/pomaranch_p3.pdf`.
3. P. Ekdahl and T. Johansson, “A new version of the stream cipher snow,” Proceeding of SAC 2002, Lecture Notes in Computer Science, vol.2595, pp.47–61, Springer-Verlag, 2002.
4. J. Daemen and V. Rijmen, The Design of Rijndael: AES – The Advanced Encryption Standard, Information Security and Cryptography, Springer-Verlag, 2002.
5. J.D. Golic, “On security of nonlinear filter generators,” Proceeding of FSE 1996, Lecture Notes in Computer Science, vol.1039, pp.173–188, 1996.
6. S. Kiyomoto, T. Tanaka, and K. Sakurai, “A word-oriented stream cipher using clock control,” Workshop Recod of SASC 2007, pp.260–274, January 2007.
7. S. Kiyomoto, T. Tanaka, and K. Sakurai, “K2: A stream cipher algorithm using dynamic feedback control,” Proceeding of SECRYPT 2007, pp.204–213, July 2007.

A Lookup Tables for Multiplication with α_0 α_1, α_2 and α_3

Lookup table `alpha_0[256]` for multiplicative operation $\alpha_0 w$:

```
alpha_0[256]={
0x00000000,0xB6086D1A,0xAF10DA34,0x1918B72E,0x9D207768,0x2B281A72,0x3230AD5C,
0x8438C046,0xF940EED0,0x4F4883CA,0x565034E4,0xE05859FE,0x646099B8,0xD268F4A2,
0xCB70438C,0x7D782E96,0x31801F63,0x87887279,0x9E90C557,0x2898A84D,0xACA0680B,
0x1AA80511,0x03B0B23F,0xB5B8DF25,0xC8C0F1B3,0x7EC89CA9,0x67D02B87,0xD1D8469D,
0x55E086DB,0xE3E8EBC1,0xFAF05CEF,0x4CF831F5,0x62C33EC6,0xD4CB53DC,0xCDD3E4F2,
0x7BDB89E8,0xFFE349AE,0x49EB24B4,0x50F3939A,0xE6FBFE80,0x9B83D016,0x2D8BBD0C,
0x34930A22,0x829B6738,0x06A3A77E,0xB0ABCA64,0xA9B37D4A,0x1FBB1050,0x534321A5,
0xE54B4CBF,0xFC53FB91,0x4A5B968B,0xCE6356CD,0x786B3BD7,0x61738CF9,0xD77BE1E3,
0xAA03CF75,0x1C0BA26F,0x05131541,0xB31B785B,0x3723B81D,0x812BD507,0x98336229,
0x2E3B0F33,0xC4457C4F,0x724D1155,0x6B55A67B,0xDD5DCB61,0x59650B27,0xEF6D663D,
0xF675D113,0x407DBC09,0x3D05929F,0x8B0DFF85,0x921548AB,0x241D25B1,0xA025E5F7,
0x162D88ED,0x0F353FC3,0xB93D52D9,0xF5C5632C,0x43CD0E36,0x5AD5B918,0xECDDD402,
0x68E51444,0xDEED795E,0xC7F5CE70,0x71FDA36A,0x0C858DFC,0xBA8DE0E6,0xA39557C8,
0x159D3AD2,0x91A5FA94,0x27AD978E,0x3EB520A0,0x88BD4DBA,0xA6864289,0x108E2F93,
0x099698BD,0xBF9EF5A7,0x3BA635E1,0x8DAE58FB,0x94B6EFD5,0x22BE82CF,0x5FC6AC59,
0xE9CEC143,0xF0D6766D,0x46DE1B77,0xC2E6DB31,0x74EEB62B,0x6DF60105,0xDBFE6C1F,
0x97065DEA,0x210E30F0,0x381687DE,0x8E1EEAC4,0x0A262A82,0xBC2E4798,0xA536F0B6,
0x133E9DAC,0x6E46B33A,0xD84EDE20,0xC156690E,0x775E0414,0xF366C452,0x456EA948,
0x5C761E66,0xEA7E737C,0x4B8AF89E,0xFD829584,0xE49A22AA,0x52924FB0,0xD6AA8FF6,
0x60A2E2EC,0x79BA55C2,0xCFB238D8,0xB2CA164E,0x04C27B54,0x1DDACC7A,0xABD2A160,
0x2FEA6126,0x99E20C3C,0x80FABB12,0x36F2D608,0x7A0AE7FD,0xCC028AE7,0xD51A3DC9,
0x631250D3,0xE72A9095,0x5122FD8F,0x483A4AA1,0xFE3227BB,0x834A092D,0x35426437,
0x2C5AD319,0x9A52BE03,0x1E6A7E45,0xA862135F,0xB17AA471,0x0772C96B,0x2949C658,
0x9F41AB42,0x86591C6C,0x30517176,0xB469B130,0x0261DC2A,0x1B796B04,0xAD71061E,
0xD0092888,0x66014592,0x7F19F2BC,0xC9119FA6,0x4D295FE0,0xFB2132FA,0xE23985D4,
0x5431E8CE,0x18C9D93B,0xAEC1B421,0xB7D9030F,0x01D16E15,0x85E9AE53,0x33E1C349,
0x2AF97467,0x9CF1197D,0xE18937EB,0x57815AF1,0x4E99EDDF,0xF89180C5,0x7CA94083,
0xCAA12D99,0xD3B99AB7,0x65B1F7AD,0x8FCF84D1,0x39C7E9CB,0x20DF5EE5,0x96D733FF,
0x12EFF3B9,0xA4E79EA3,0xBDFF298D,0x0BF74497,0x768F6A01,0xC087071B,0xD99FB035,
0x6F97DD2F,0xEBAF1D69,0x5DA77073,0x44BFC75D,0xF2B7AA47,0xBE4F9BB2,0x0847F6A8,
0x115F4186,0xA7572C9C,0x236FECDA,0x956781C0,0x8C7F36EE,0x3A775BF4,0x470F7562,
0xF1071878,0xE81FAF56,0x5E17C24C,0xDA2F020A,0x6C276F10,0x753FD83E,0xC337B524,
0xED0CBA17,0x5B04D70D,0x421C6023,0xF4140D39,0x702CCD7F,0xC624A065,0xDF3C174B,
0x69347A51,0x144C54C7,0xA24439DD,0xBB5C8EF3,0x0D54E3E9,0x896C23AF,0x3F644EB5,
0x267CF99B,0x90749481,0xDC8CA574,0x6A84C86E,0x739C7F40,0xC594125A,0x41ACD21C,
0xF7A4BF06,0xEEBC0828,0x58B46532,0x25CC4BA4,0x93C426BE,0x8ADC9190,0x3CD4FC8A,
0xB8EC3CCC,0x0EE451D6,0x17FCE6F8,0xA1F48BE2};
```

Lookup table `alpha_1[256]` for multiplicative operation $\alpha_1 w$:

```
alpha_1[256]={
0x00000000,0xA0F5FC2E,0x6DC7D55C,0xCD322972,0xDAA387B8,0x7A567B96,0xB76452E4,
0x1791AECA,0x996B235D,0x399EDF73,0xF4ACF601,0x54590A2F,0x43C8A4E5,0xE33D58CB,
0x2E0F71B9,0x8EFA8D97,0x1FD646BA,0xBF23BA94,0x721193E6,0xD2E46FC8,0xC575C102,
0x65803D2C,0xA8B2145E,0x0847E870,0x86BD65E7,0x264899C9,0xEB7AB0BB,0x4B8F4C95,
0x5C1EE25F,0xFCEB1E71,0x31D93703,0x912CCB2D,0x3E818C59,0x9E747077,0x53465905,
0xF3B3A52B,0xE4220BE1,0x44D7F7CF,0x89E5DEBD,0x29102293,0xA7EAAF04,0x071F532A,
0xCA2D7A58,0x6AD88676,0x7D4928BC,0xDDBCD492,0x108EFDE0,0xB07B01CE,0x2157CAE3,
0x81A236CD,0x4C901FBF,0xEC65E391,0xFBF44D5B,0x5B01B175,0x96339807,0x36C66429,
0xB83CE9BE,0x18C91590,0xD5FB3CE2,0x750EC0CC,0x629F6E06,0xC26A9228,0x0F58BB5A,
0xAFAD4774,0x7C2F35B2,0xDCDAC99C,0x11E8E0EE,0xB11D1CC0,0xA68CB20A,0x06794E24,
0xCB4B6756,0x6BBE9B78,0xE54416EF,0x45B1EAC1,0x8883C3B3,0x28763F9D,0x3FE79157,
0x9F126D79,0x5220440B,0xF2D5B825,0x63F97308,0xC30C8F26,0x0E3EA654,0xAECB5A7A,
0xB95AF4B0,0x19AF089E,0xD49D21EC,0x7468DDC2,0xFA925055,0x5A67AC7B,0x97558509,
0x37A07927,0x2031D7ED,0x80C42BC3,0x4DF602B1,0xED03FE9F,0x42AEB9EB,0xE25B45C5,
0x2F696CB7,0x8F9C9099,0x980D3E53,0x38F8C27D,0xF5CAEB0F,0x553F1721,0xDBC59AB6,
0x7B306698,0xB6024FEA,0x16F7B3C4,0x01661D0E,0xA193E120,0x6CA1C852,0xCC54347C,
0x5D78FF51,0xFD8D037F,0x30BF2A0D,0x904AD623,0x87DB78E9,0x272E84C7,0xEA1CADB5,
0x4AE9519B,0xC413DC0C,0x64E62022,0xA9D40950,0x0921F57E,0x1EB05BB4,0xBE45A79A,
0x73778EE8,0xD38272C6,0xF85E6A49,0x58AB9667,0x9599BF15,0x356C433B,0x22FDEDF1,
0x820811DF,0x4F3A38AD,0xEFCFC483,0x61354914,0xC1C0B53A,0x0CF29C48,0xAC076066,
0xBB96CEAC,0x1B633282,0xD6511BF0,0x76A4E7DE,0xE7882CF3,0x477DD0DD,0x8A4FF9AF,
0x2ABA0581,0x3D2BAB4B,0x9DDE5765,0x50EC7E17,0xF0198239,0x7EE30FAE,0xDE16F380,
0x1324DAF2,0xB3D126DC,0xA4408816,0x04B57438,0xC9875D4A,0x6972A164,0xC6DFE610,
0x662A1A3E,0xAB18334C,0x0BEDCF62,0x1C7C61A8,0xBC899D86,0x71BBB4F4,0xD14E48DA,
0x5FB4C54D,0xFF413963,0x32731011,0x9286EC3F,0x851742F5,0x25E2BEDB,0xE8D097A9,
0x48256B87,0xD909A0AA,0x79FC5C84,0xB4CE75F6,0x143B89D8,0x03AA2712,0xA35FDB3C,
0x6E6DF24E,0xCE980E60,0x406283F7,0xE0977FD9,0x2DA556AB,0x8D50AA85,0x9AC1044F,
0x3A34F861,0xF706D113,0x57F32D3D,0x84715FFB,0x2484A3D5,0xE9B68AA7,0x49437689,
0x5ED2D843,0xFE27246D,0x33150D1F,0x93E0F131,0x1D1A7CA6,0xBDEF8088,0x70DDA9FA,
0xD02855D4,0xC7B9FB1E,0x674C0730,0xAA7E2E42,0x0A8BD26C,0x9BA71941,0x3B52E56F,
0xF660CC1D,0x56953033,0x41049EF9,0xE1F162D7,0x2CC34BA5,0x8C36B78B,0x02CC3A1C,
0xA239C632,0x6F0BEF40,0xCFFE136E,0xD86FBDA4,0x789A418A,0xB5A868F8,0x155D94D6,
0xBAF0D3A2,0x1A052F8C,0xD73706FE,0x77C2FAD0,0x6053541A,0xC0A6A834,0x0D948146,
0xAD617D68,0x239BF0FF,0x836E0CD1,0x4E5C25A3,0xEEA9D98D,0xF9387747,0x59CD8B69,
0x94FFA21B,0x340A5E35,0xA5269518,0x05D36936,0xC8E14044,0x6814BC6A,0x7F8512A0,
0xDF70EE8E,0x1242C7FC,0xB2B73BD2,0x3C4DB645,0x9CB84A6B,0x518A6319,0xF17F9F37,
0xE6EE31FD,0x461BCDD3,0x8B29E4A1,0x2BDC188F};
```

Lookup table `alpha_2[256]` for multiplicative operation $\alpha_2 w$:

```
alpha_2[256]={
0x00000000,0x5BF87F93,0xB6BDFE6B,0xED4581F8,0x2137B1D6,0x7ACFCE45,0x978A4FBD,
0xCC72302E,0x426E2FE1,0x19965072,0xF4D3D18A,0xAF2BAE19,0x63599E37,0x38A1E1A4,
0xD5E4605C,0x8E1C1FCF,0x84DC5E8F,0xDF24211C,0x3261A0E4,0x6999DF77,0xA5EBEF59,
0xFE1390CA,0x13561132,0x48AE6EA1,0xC6B2716E,0x9D4A0EFD,0x700F8F05,0x2BF7F096,
0xE785C0B8,0xBC7DBF2B,0x51383ED3,0x0AC04140,0x45F5BC53,0x1E0DC3C0,0xF3484238,
0xA8B03DAB,0x64C20D85,0x3F3A7216,0xD27FF3EE,0x89878C7D,0x079B93B2,0x5C63EC21,
0xB1266DD9,0xEADE124A,0x26AC2264,0x7D545DF7,0x9011DC0F,0xCBE9A39C,0xC129E2DC,
0x9AD19D4F,0x77941CB7,0x2C6C6324,0xE01E530A,0xBBE62C99,0x56A3AD61,0x0D5BD2F2,
0x8347CD3D,0xD8BFB2AE,0x35FA3356,0x6E024CC5,0xA2707CEB,0xF9880378,0x14CD8280,
0x4F35FD13,0x8AA735A6,0xD15F4A35,0x3C1ACBCD,0x67E2B45E,0xAB908470,0xF068FBE3,
0x1D2D7A1B,0x46D50588,0xC8C91A47,0x933165D4,0x7E74E42C,0x258C9BBF,0xE9FEAB91,
0xB206D402,0x5F4355FA,0x04BB2A69,0x0E7B6B29,0x558314BA,0xB8C69542,0xE33EEAD1,
0x2F4CDAFF,0x74B4A56C,0x99F12494,0xC2095B07,0x4C1544C8,0x17ED3B5B,0xFAA8BAA3,
0xA150C530,0x6D22F51E,0x36DA8A8D,0xDB9F0B75,0x806774E6,0xCF5289F5,0x94AAF666,
0x79EF779E,0x2217080D,0xEE653823,0xB59D47B0,0x58D8C648,0x0320B9DB,0x8D3CA614,
0xD6C4D987,0x3B81587F,0x607927EC,0xAC0B17C2,0xF7F36851,0x1AB6E9A9,0x414E963A,
0x4B8ED77A,0x1076A8E9,0xFD332911,0xA6CB5682,0x6AB966AC,0x3141193F,0xDC0498C7,
0x87FCE754,0x09E0F89B,0x52188708,0xBF5D06F0,0xE4A57963,0x28D7494D,0x732F36DE,
0x9E6AB726,0xC592C8B5,0x59036A01,0x02FB1592,0xEFBE946A,0xB446EBF9,0x7834DBD7,
0x23CCA444,0xCE8925BC,0x95715A2F,0x1B6D45E0,0x40953A73,0xADD0BB8B,0xF628C418,
0x3A5AF436,0x61A28BA5,0x8CE70A5D,0xD71F75CE,0xDDDF348E,0x86274B1D,0x6B62CAE5,
0x309AB576,0xFCE88558,0xA710FACB,0x4A557B33,0x11AD04A0,0x9FB11B6F,0xC44964FC,
0x290CE504,0x72F49A97,0xBE86AAB9,0xE57ED52A,0x083B54D2,0x53C32B41,0x1CF6D652,
0x470EA9C1,0xAA4B2839,0xF1B357AA,0x3DC16784,0x66391817,0x8B7C99EF,0xD084E67C,
0x5E98F9B3,0x05608620,0xE82507D8,0xB3DD784B,0x7FAF4865,0x245737F6,0xC912B60E,
0x92EAC99D,0x982A88DD,0xC3D2F74E,0x2E9776B6,0x756F0925,0xB91D390B,0xE2E54698,
0x0FA0C760,0x5458B8F3,0xDA44A73C,0x81BCD8AF,0x6CF95957,0x370126C4,0xFB7316EA,
0xA08B6979,0x4DCEE881,0x16369712,0xD3A45FA7,0x885C2034,0x6519A1CC,0x3EE1DE5F,
0xF293EE71,0xA96B91E2,0x442E101A,0x1FD66F89,0x91CA7046,0xCA320FD5,0x27778E2D,
0x7C8FF1BE,0xB0FDC190,0xEB05BE03,0x06403FFB,0x5DB84068,0x57780128,0x0C807EBB,
0xE1C5FF43,0xBA3D80D0,0x764FB0FE,0x2DB7CF6D,0xC0F24E95,0x9B0A3106,0x15162EC9,
0x4EEE515A,0xA3ABD0A2,0xF853AF31,0x34219F1F,0x6FD9E08C,0x829C6174,0xD9641EE7,
0x9651E3F4,0xCDA99C67,0x20EC1D9F,0x7B14620C,0xB7665222,0xEC9E2DB1,0x01DBAC49,
0x5A23D3DA,0xD43FCC15,0x8FC7B386,0x6282327E,0x397A4DED,0xF5087DC3,0xAEF00250,
0x43B583A8,0x184DFC3B,0x128DBD7B,0x4975C2E8,0xA4304310,0xFFC83C83,0x33BA0CAD,
0x6842733E,0x8507F2C6,0xDEFF8D55,0x50E3929A,0x0B1BED09,0xE65E6CF1,0xBDA61362,
0x71D4234C,0x2A2C5CDF,0xC769DD27,0x9C91A2B4};
```

Lookup table `alpha_3[256]` for multiplicative operation $\alpha_3 w$:

```
alpha_3[256]={
0x00000000,0x4559568B,0x8AB2AC73,0xCFEBFAF8,0x71013DE6,0x34586B6D,0xFBB39195,
0xBEEAC71E,0xE2027AA9,0xA75B2C22,0x68B0D6DA,0x2DE98051,0x9303474F,0xD65A11C4,
0x19B1EB3C,0x5CE8BDB7,0xA104F437,0xE45DA2BC,0x2BB65844,0x6EEF0ECF,0xD005C9D1,
0x955C9F5A,0x5AB765A2,0x1FEE3329,0x43068E9E,0x065FD815,0xC9B422ED,0x8CED7466,
0x3207B378,0x775EE5F3,0xB8B51F0B,0xFDEC4980,0x27088D6E,0x6251DBE5,0xADBA211D,
0xE8E37796,0x5609B088,0x1350E603,0xDCBB1CFB,0x99E24A70,0xC50AF7C7,0x8053A14C,
0x4FB85BB4,0x0AE10D3F,0xB40BCA21,0xF1529CAA,0x3EB96652,0x7BE030D9,0x860C7959,
0xC3552FD2,0x0CBED52A,0x49E783A1,0xF70D44BF,0xB2541234,0x7DBFE8CC,0x38E6BE47,
0x640E03F0,0x2157557B,0xEEBCAF83,0xABE5F908,0x150F3E16,0x5056689D,0x9FBD9265,
0xDAE4C4EE,0x4E107FDC,0x0B492957,0xC4A2D3AF,0x81FB8524,0x3F11423A,0x7A4814B1,
0xB5A3EE49,0xF0FAB8C2,0xAC120575,0xE94B53FE,0x26A0A906,0x63F9FF8D,0xDD133893,
0x984A6E18,0x57A194E0,0x12F8C26B,0xEF148BEB,0xAA4DDD60,0x65A62798,0x20FF7113,
0x9E15B60D,0xDB4CE086,0x14A71A7E,0x51FE4CF5,0x0D16F142,0x484FA7C9,0x87A45D31,
0xC2FD0BBA,0x7C17CCA4,0x394E9A2F,0xF6A560D7,0xB3FC365C,0x6918F2B2,0x2C41A439,
0xE3AA5EC1,0xA6F3084A,0x1819CF54,0x5D4099DF,0x92AB6327,0xD7F235AC,0x8B1A881B,
0xCE43DE90,0x01A82468,0x44F172E3,0xFA1BB5FD,0xBF42E376,0x70A9198E,0x35F04F05,
0xC81C0685,0x8D45500E,0x42AEAAF6,0x07F7FC7D,0xB91D3B63,0xFC446DE8,0x33AF9710,
0x76F6C19B,0x2A1E7C2C,0x6F472AA7,0xA0ACD05F,0xE5F586D4,0x5B1F41CA,0x1E461741,
0xD1ADEDB9,0x94F4BB32,0x9C20FEDD,0xD979A856,0x169252AE,0x53CB0425,0xED21C33B,
0xA87895B0,0x67936F48,0x22CA39C3,0x7E228474,0x3B7BD2FF,0xF4902807,0xB1C97E8C,
0x0F23B992,0x4A7AEF19,0x859115E1,0xC0C8436A,0x3D240AEA,0x787D5C61,0xB796A699,
0xF2CFF012,0x4C25370C,0x097C6187,0xC6979B7F,0x83CECDF4,0xDF267043,0x9A7F26C8,
0x5594DC30,0x10CD8ABB,0xAE274DA5,0xEB7E1B2E,0x2495E1D6,0x61CCB75D,0xBB2873B3,
0xFE712538,0x319ADFC0,0x74C3894B,0xCA294E55,0x8F7018DE,0x409BE226,0x05C2B4AD,
0x592A091A,0x1C735F91,0xD398A569,0x96C1F3E2,0x282B34FC,0x6D726277,0xA299988F,
0xE7C0CE04,0x1A2C8784,0x5F75D10F,0x909E2BF7,0xD5C77D7C,0x6B2DBA62,0x2E74ECE9,
0xE19F1611,0xA4C6409A,0xF82EFD2D,0xBD77ABA6,0x729C515E,0x37C507D5,0x892FC0CB,
0xCC769640,0x039D6CB8,0x46C43A33,0xD2308101,0x9769D78A,0x58822D72,0x1DDB7BF9,
0xA331BCE7,0xE668EA6C,0x29831094,0x6CDA461F,0x3032FBA8,0x756BAD23,0xBA8057DB,
0xFFD90150,0x4133C64E,0x046A90C5,0xCB816A3D,0x8ED83CB6,0x73347536,0x366D23BD,
0xF986D945,0xBCDF8FCE,0x023548D0,0x476C1E5B,0x8887E4A3,0xCDDEB228,0x91360F9F,
0xD46F5914,0x1B84A3EC,0x5EDDF567,0xE0373279,0xA56E64F2,0x6A859E0A,0x2FDCC881,
0xF5380C6F,0xB0615AE4,0x7F8AA01C,0x3AD3F697,0x84393189,0xC1606702,0x0E8B9DFA,
0x4BD2CB71,0x173A76C6,0x5263204D,0x9D88DAB5,0xD8D18C3E,0x663B4B20,0x23621DAB,
0xEC89E753,0xA9D0B1D8,0x543CF858,0x1165AED3,0xDE8E542B,0x9BD702A0,0x253DC5BE,
0x60649335,0xAF8F69CD,0xEAD63F46,0xB63E82F1,0xF367D47A,0x3C8C2E82,0x79D57809,
0xC73FBF17,0x8266E99C,0x4D8D1364,0x08D445EF};
```

B Lookup Tables for Function *Sub*

Lookup table T_0[256] for *Sub* in Nonlinear Function:

```
T_0[256]={
0xa56363c6,0x847c7cf8,0x997777ee,0x8d7b7bf6,0x0df2f2ff,0xbd6b6bd6,0xb16f6fde,
0x54c5c591,0x50303060,0x03010102,0xa96767ce,0x7d2b2b56,0x19fefee7,0x62d7d7b5,
0xe6abab4d,0x9a7676ec,0x45caca8f,0x9d82821f,0x40c9c989,0x877d7dfa,0x15fafaef,
0xeb5959b2,0xc947478e,0x0bf0f0fb,0xecadad41,0x67d4d4b3,0xfda2a25f,0xeaafaf45,
0xbf9c9c23,0xf7a4a453,0x967272e4,0x5bc0c09b,0xc2b7b775,0x1cfdfde1,0xae93933d,
0x6a26264c,0x5a36366c,0x413f3f7e,0x02f7f7f5,0x4fcccc83,0x5c343468,0xf4a5a551,
0x34e5e5d1,0x08f1f1f9,0x937171e2,0x73d8d8ab,0x53313162,0x3f15152a,0x0c040408,
0x52c7c795,0x65232346,0x5ec3c39d,0x28181830,0xa1969637,0x0f05050a,0xb59a9a2f,
0x0907070e,0x36121224,0x9b80801b,0x3de2e2df,0x26ebebcd,0x6927274e,0xcdb2b27f,
0x9f7575ea,0x1b090912,0x9e83831d,0x742c2c58,0x2e1a1a34,0x2d1b1b36,0xb26e6edc,
0xee5a5ab4,0xfba0a05b,0xf65252a4,0x4d3b3b76,0x61d6d6b7,0xceb3b37d,0x7b292952,
0x3ee3e3dd,0x712f2f5e,0x97848413,0xf55353a6,0x68d1d1b9,0x00000000,0x2cededc1,
0x60202040,0x1ffcfce3,0xc8b1b179,0xed5b5bb6,0xbe6a6ad4,0x46cbcb8d,0xd9bebe67,
0x4b393972,0xde4a4a94,0xd44c4c98,0xe85858b0,0x4acfcf85,0x6bd0d0bb,0x2aefefc5,
0xe5aaaa4f,0x16fbfbed,0xc5434386,0xd74d4d9a,0x55333366,0x94858511,0xcf45458a,
0x10f9f9e9,0x06020204,0x817f7ffe,0xf05050a0,0x443c3c78,0xba9f9f25,0xe3a8a84b,
0xf35151a2,0xfea3a35d,0xc0404080,0x8a8f8f05,0xad92923f,0xbc9d9d21,0x48383870,
0x04f5f5f1,0xdfbcbc63,0xc1b6b677,0x75dadaaf,0x63212142,0x30101020,0x1affffe5,
0x0ef3f3fd,0x6dd2d2bf,0x4ccdcd81,0x140c0c18,0x35131326,0x2fececc3,0xe15f5fbe,
0xa2979735,0xcc444488,0x3917172e,0x57c4c493,0xf2a7a755,0x827e7efc,0x473d3d7a,
0xac6464c8,0xe75d5dba,0x2b191932,0x957373e6,0xa06060c0,0x98818119,0xd14f4f9e,
0x7fdcdca3,0x66222244,0x7e2a2a54,0xab90903b,0x8388880b,0xca46468c,0x29eeeec7,
0xd3b8b86b,0x3c141428,0x79dedea7,0xe25e5ebc,0x1d0b0b16,0x76dbdbad,0x3be0e0db,
0x56323264,0x4e3a3a74,0x1e0a0a14,0xdb494992,0x0a06060c,0x6c242448,0xe45c5cb8,
0x5dc2c29f,0x6ed3d3bd,0xefacac43,0xa66262c4,0xa8919139,0xa4959531,0x37e4e4d3,
0x8b7979f2,0x32e7e7d5,0x43c8c88b,0x5937376e,0xb76d6dda,0x8c8d8d01,0x64d5d5b1,
0xd24e4e9c,0xe0a9a949,0xb46c6cd8,0xfa5656ac,0x07f4f4f3,0x25eaeacf,0xaf6565ca,
0x8e7a7af4,0xe9aeae47,0x18080810,0xd5baba6f,0x887878f0,0x6f25254a,0x722e2e5c,
0x241c1c38,0xf1a6a657,0xc7b4b473,0x51c6c697,0x23e8e8cb,0x7cdddda1,0x9c7474e8,
0x211f1f3e,0xdd4b4b96,0xdcbdbd61,0x868b8b0d,0x858a8a0f,0x907070e0,0x423e3e7c,
0xc4b5b571,0xaa6666cc,0xd8484890,0x05030306,0x01f6f6f7,0x120e0e1c,0xa36161c2,
0x5f35356a,0xf95757ae,0xd0b9b969,0x91868617,0x58c1c199,0x271d1d3a,0xb99e9e27,
0x38e1e1d9,0x13f8f8eb,0xb398982b,0x33111122,0xbb6969d2,0x70d9d9a9,0x898e8e07,
0xa7949433,0xb69b9b2d,0x221e1e3c,0x92878715,0x20e9e9c9,0x49cece87,0xff5555aa,
0x78282850,0x7adfdfa5,0x8f8c8c03,0xf8a1a159,0x80898909,0x170d0d1a,0xdabfbf65,
0x31e6e6d7,0xc6424284,0xb86868d0,0xc3414182,0xb0999929,0x772d2d5a,0x110f0f1e,
0xcbb0b07b,0xfc5454a8,0xd6bbbb6d,0x3a16162c};
```

Lookup table `T_1[256]` for *Sub* in Nonlinear Function:

```
T_1[256]={
0x6363c6a5,0x7c7cf884,0x7777ee99,0x7b7bf68d,0xf2f2ff0d,0x6b6bd6bd,0x6f6fdeb1,
0xc5c59154,0x30306050,0x01010203,0x6767cea9,0x2b2b567d,0xfefee719,0xd7d7b562,
0xabab4de6,0x7676ec9a,0xcaca8f45,0x82821f9d,0xc9c98940,0x7d7dfa87,0xfafaef15,
0x5959b2eb,0x47478ec9,0xf0f0fb0b,0xadad41ec,0xd4d4b367,0xa2a25ffd,0xafaf45ea,
0x9c9c23bf,0xa4a453f7,0x7272e496,0xc0c09b5b,0xb7b775c2,0xfdfde11c,0x93933dae,
0x26264c6a,0x36366c5a,0x3f3f7e41,0xf7f7f502,0xcccc834f,0x3434685c,0xa5a551f4,
0xe5e5d134,0xf1f1f908,0x7171e293,0xd8d8ab73,0x31316253,0x15152a3f,0x0404080c,
0xc7c79552,0x23234665,0xc3c39d5e,0x18183028,0x969637a1,0x05050a0f,0x9a9a2fb5,
0x07070e09,0x12122436,0x80801b9b,0xe2e2df3d,0xebebcd26,0x27274e69,0xb2b27fcd,
0x7575ea9f,0x0909121b,0x83831d9e,0x2c2c5874,0x1a1a342e,0x1b1b362d,0x6e6edcb2,
0x5a5ab4ee,0xa0a05bfb,0x5252a4f6,0x3b3b764d,0xd6d6b761,0xb3b37dce,0x2929527b,
0xe3e3dd3e,0x2f2f5e71,0x84841397,0x5353a6f5,0xd1d1b968,0x00000000,0xededc12c,
0x20204060,0xfcfce31f,0xb1b179c8,0x5b5bb6ed,0x6a6ad4be,0xcbcb8d46,0xbebe67d9,
0x3939724b,0x4a4a94de,0x4c4c98d4,0x5858b0e8,0xcfcf854a,0xd0d0bb6b,0xefefc52a,
0xaaaa4fe5,0xfbfbed16,0x434386c5,0x4d4d9ad7,0x33336655,0x85851194,0x45458acf,
0xf9f9e910,0x02020406,0x7f7ffe81,0x5050a0f0,0x3c3c7844,0x9f9f25ba,0xa8a84be3,
0x5151a2f3,0xa3a35dfe,0x404080c0,0x8f8f058a,0x92923fad,0x9d9d21bc,0x38387048,
0xf5f5f104,0xbcbc63df,0xb6b677c1,0xdadaaf75,0x21214263,0x10102030,0xffffe51a,
0xf3f3fd0e,0xd2d2bf6d,0xcdcd814c,0x0c0c1814,0x13132635,0xececc32f,0x5f5fbee1,
0x979735a2,0x444488cc,0x17172e39,0xc4c49357,0xa7a755f2,0x7e7efc82,0x3d3d7a47,
0x6464c8ac,0x5d5dbae7,0x1919322b,0x7373e695,0x6060c0a0,0x81811998,0x4f4f9ed1,
0xdcdca37f,0x22224466,0x2a2a547e,0x90903bab,0x88880b83,0x46468cca,0xeeeec729,
0xb8b86bd3,0x1414283c,0xdedea779,0x5e5ebce2,0x0b0b161d,0xdbdbad76,0xe0e0db3b,
0x32326456,0x3a3a744e,0x0a0a141e,0x494992db,0x06060c0a,0x2424486c,0x5c5cb8e4,
0xc2c29f5d,0xd3d3bd6e,0xacac43ef,0x6262c4a6,0x919139a8,0x959531a4,0xe4e4d337,
0x7979f28b,0xe7e7d532,0xc8c88b43,0x37376e59,0x6d6ddab7,0x8d8d018c,0xd5d5b164,
0x4e4e9cd2,0xa9a949e0,0x6c6cd8b4,0x5656acfa,0xf4f4f307,0xeaeacf25,0x6565caaf,
0x7a7af48e,0xaeae47e9,0x08081018,0xbaba6fd5,0x7878f088,0x25254a6f,0x2e2e5c72,
0x1c1c3824,0xa6a657f1,0xb4b473c7,0xc6c69751,0xe8e8cb23,0xdddda17c,0x7474e89c,
0x1f1f3e21,0x4b4b96dd,0xbdbd61dc,0x8b8b0d86,0x8a8a0f85,0x7070e090,0x3e3e7c42,
0xb5b571c4,0x6666ccaa,0x484890d8,0x03030605,0xf6f6f701,0x0e0e1c12,0x6161c2a3,
0x35356a5f,0x5757aef9,0xb9b969d0,0x86861791,0xc1c19958,0x1d1d3a27,0x9e9e27b9,
0xe1e1d938,0xf8f8eb13,0x98982bb3,0x11112233,0x6969d2bb,0xd9d9a970,0x8e8e0789,
0x949433a7,0x9b9b2db6,0x1e1e3c22,0x87871592,0xe9e9c920,0xcece8749,0x5555aaff,
0x28285078,0xdfdfa57a,0x8c8c038f,0xa1a159f8,0x89890980,0x0d0d1a17,0xbfbf65da,
0xe6e6d731,0x424284c6,0x6868d0b8,0x414182c3,0x999929b0,0x2d2d5a77,0x0f0f1e11,
0xb0b07bcb,0x5454a8fc,0xbbbb6dd6,0x16162c3a};
```

Lookup table `T_2[256]` for *Sub* in Nonlinear Function:

```
T_2[256]={
0x63c6a563,0x7cf8847c,0x77ee9977,0x7bf68d7b,0xf2ff0df2,0x6bd6bd6b,0x6fdeb16f,
0xc59154c5,0x30605030,0x01020301,0x67cea967,0x2b567d2b,0xfee719fe,0xd7b562d7,
0xab4de6ab,0x76ec9a76,0xca8f45ca,0x821f9d82,0xc98940c9,0x7dfa877d,0xfaef15fa,
0x59b2eb59,0x478ec947,0xf0fb0bf0,0xad41ecad,0xd4b367d4,0xa25ffda2,0xaf45eaaf,
0x9c23bf9c,0xa453f7a4,0x72e49672,0xc09b5bc0,0xb775c2b7,0xfde11cfd,0x933dae93,
0x264c6a26,0x366c5a36,0x3f7e413f,0xf7f502f7,0xcc834fcc,0x34685c34,0xa551f4a5,
0xe5d134e5,0xf1f908f1,0x71e29371,0xd8ab73d8,0x31625331,0x152a3f15,0x04080c04,
0xc79552c7,0x23466523,0xc39d5ec3,0x18302818,0x9637a196,0x050a0f05,0x9a2fb59a,
0x070e0907,0x12243612,0x801b9b80,0xe2df3de2,0xebcd26eb,0x274e6927,0xb27fcdb2,
0x75ea9f75,0x09121b09,0x831d9e83,0x2c58742c,0x1a342e1a,0x1b362d1b,0x6edcb26e,
0x5ab4ee5a,0xa05bfba0,0x52a4f652,0x3b764d3b,0xd6b761d6,0xb37dceb3,0x29527b29,
0xe3dd3ee3,0x2f5e712f,0x84139784,0x53a6f553,0xd1b968d1,0x00000000,0xedc12ced,
0x20406020,0xfce31ffc,0xb179c8b1,0x5bb6ed5b,0x6ad4be6a,0xcb8d46cb,0xbe67d9be,
0x39724b39,0x4a94de4a,0x4c98d44c,0x58b0e858,0xcf854acf,0xd0bb6bd0,0xefc52aef,
0xaa4fe5aa,0xfbed16fb,0x4386c543,0x4d9ad74d,0x33665533,0x85119485,0x458acf45,
0xf9e910f9,0x02040602,0x7ffe817f,0x50a0f050,0x3c78443c,0x9f25ba9f,0xa84be3a8,
0x51a2f351,0xa35dfea3,0x4080c040,0x8f058a8f,0x923fad92,0x9d21bc9d,0x38704838,
0xf5f104f5,0xbc63dfbc,0xb677c1b6,0xdaaf75da,0x21426321,0x10203010,0xffe51aff,
0xf3fd0ef3,0xd2bf6dd2,0xcd814ccd,0x0c18140c,0x13263513,0xecc32fec,0x5fbee15f,
0x9735a297,0x4488cc44,0x172e3917,0xc49357c4,0xa755f2a7,0x7efc827e,0x3d7a473d,
0x64c8ac64,0x5dbae75d,0x19322b19,0x73e69573,0x60c0a060,0x81199881,0x4f9ed14f,
0xdca37fdc,0x22446622,0x2a547e2a,0x903bab90,0x880b8388,0x468cca46,0xeec729ee,
0xb86bd3b8,0x14283c14,0xdea779de,0x5ebce25e,0x0b161d0b,0xdbad76db,0xe0db3be0,
0x32645632,0x3a744e3a,0x0a141e0a,0x4992db49,0x060c0a06,0x24486c24,0x5cb8e45c,
0xc29f5dc2,0xd3bd6ed3,0xac43efac,0x62c4a662,0x9139a891,0x9531a495,0xe4d337e4,
0x79f28b79,0xe7d532e7,0xc88b43c8,0x376e5937,0x6ddab76d,0x8d018c8d,0xd5b164d5,
0x4e9cd24e,0xa949e0a9,0x6cd8b46c,0x56acfa56,0xf4f307f4,0xeacf25ea,0x65caaf65,
0x7af48e7a,0xae47e9ae,0x08101808,0xba6fd5ba,0x78f08878,0x254a6f25,0x2e5c722e,
0x1c38241c,0xa657f1a6,0xb473c7b4,0xc69751c6,0xe8cb23e8,0xdda17cdd,0x74e89c74,
0x1f3e211f,0x4b96dd4b,0xbd61dcbd,0x8b0d868b,0x8a0f858a,0x70e09070,0x3e7c423e,
0xb571c4b5,0x66ccaa66,0x4890d848,0x03060503,0xf6f701f6,0x0e1c120e,0x61c2a361,
0x356a5f35,0x57aef957,0xb969d0b9,0x86179186,0xc19958c1,0x1d3a271d,0x9e27b99e,
0xe1d938e1,0xf8eb13f8,0x982bb398,0x11223311,0x69d2bb69,0xd9a970d9,0x8e07898e,
0x9433a794,0x9b2db69b,0x1e3c221e,0x87159287,0xe9c920e9,0xce8749ce,0x55aaff55,
0x28507828,0xdfa57adf,0x8c038f8c,0xa159f8a1,0x89098089,0x0d1a170d,0xbf65dabf,
0xe6d731e6,0x4284c642,0x68d0b868,0x4182c341,0x9929b099,0x2d5a772d,0x0f1e110f,
0xb07bcbb0,0x54a8fc54,0xbb6dd6bb,0x162c3a16};
```

Lookup table T_3[256] for *Sub* in Nonlinear Function:

```
T_3[256]={
0xc6a56363,0xf8847c7c,0xee997777,0xf68d7b7b,0xff0df2f2,0xd6bd6b6b,0xdeb16f6f,
0x9154c5c5,0x60503030,0x02030101,0xcea96767,0x567d2b2b,0xe719fefe,0xb562d7d7,
0x4de6abab,0xec9a7676,0x8f45caca,0x1f9d8282,0x8940c9c9,0xfa877d7d,0xef15fafa,
0xb2eb5959,0x8ec94747,0xfb0bf0f0,0x41ecadad,0xb367d4d4,0x5ffda2a2,0x45eaafaf,
0x23bf9c9c,0x53f7a4a4,0xe4967272,0x9b5bc0c0,0x75c2b7b7,0xe11cfdfd,0x3dae9393,
0x4c6a2626,0x6c5a3636,0x7e413f3f,0xf502f7f7,0x834fcccc,0x685c3434,0x51f4a5a5,
0xd134e5e5,0xf908f1f1,0xe2937171,0xab73d8d8,0x62533131,0x2a3f1515,0x080c0404,
0x9552c7c7,0x46652323,0x9d5ec3c3,0x30281818,0x37a19696,0x0a0f0505,0x2fb59a9a,
0x0e090707,0x24361212,0x1b9b8080,0xdf3de2e2,0xcd26ebeb,0x4e692727,0x7fcdb2b2,
0xea9f7575,0x121b0909,0x1d9e8383,0x58742c2c,0x342e1a1a,0x362d1b1b,0xdcb26e6e,
0xb4ee5a5a,0x5bfba0a0,0xa4f65252,0x764d3b3b,0xb761d6d6,0x7dceb3b3,0x527b2929,
0xdd3ee3e3,0x5e712f2f,0x13978484,0xa6f55353,0xb968d1d1,0x00000000,0xc12ceded,
0x40602020,0xe31ffcfc,0x79c8b1b1,0xb6ed5b5b,0xd4be6a6a,0x8d46cbcb,0x67d9bebe,
0x724b3939,0x94de4a4a,0x98d44c4c,0xb0e85858,0x854acfcf,0xbb6bd0d0,0xc52aefef,
0x4fe5aaaa,0xed16fbfb,0x86c54343,0x9ad74d4d,0x66553333,0x11948585,0x8acf4545,
0xe910f9f9,0x04060202,0xfe817f7f,0xa0f05050,0x78443c3c,0x25ba9f9f,0x4be3a8a8,
0xa2f35151,0x5dfea3a3,0x80c04040,0x058a8f8f,0x3fad9292,0x21bc9d9d,0x70483838,
0xf104f5f5,0x63dfbcbc,0x77c1b6b6,0xaf75dada,0x42632121,0x20301010,0xe51affff,
0xfd0ef3f3,0xbf6dd2d2,0x814ccdcd,0x18140c0c,0x26351313,0xc32fecec,0xbee15f5f,
0x35a29797,0x88cc4444,0x2e391717,0x9357c4c4,0x55f2a7a7,0xfc827e7e,0x7a473d3d,
0xc8ac6464,0xbae75d5d,0x322b1919,0xe6957373,0xc0a06060,0x19988181,0x9ed14f4f,
0xa37fdcdc,0x44662222,0x547e2a2a,0x3bab9090,0x0b838888,0x8cca4646,0xc729eeee,
0x6bd3b8b8,0x283c1414,0xa779dede,0xbce25e5e,0x161d0b0b,0xad76dbdb,0xdb3be0e0,
0x64563232,0x744e3a3a,0x141e0a0a,0x92db4949,0x0c0a0606,0x486c2424,0xb8e45c5c,
0x9f5dc2c2,0xbd6ed3d3,0x43efacac,0xc4a66262,0x39a89191,0x31a49595,0xd337e4e4,
0xf28b7979,0xd532e7e7,0x8b43c8c8,0x6e593737,0xdab76d6d,0x018c8d8d,0xb164d5d5,
0x9cd24e4e,0x49e0a9a9,0xd8b46c6c,0xacfa5656,0xf307f4f4,0xcf25eaea,0xcaaf6565,
0xf48e7a7a,0x47e9aeae,0x10180808,0x6fd5baba,0xf0887878,0x4a6f2525,0x5c722e2e,
0x38241c1c,0x57f1a6a6,0x73c7b4b4,0x9751c6c6,0xcb23e8e8,0xa17cdddd,0xe89c7474,
0x3e211f1f,0x96dd4b4b,0x61dcbdbd,0x0d868b8b,0x0f858a8a,0xe0907070,0x7c423e3e,
0x71c4b5b5,0xccaa6666,0x90d84848,0x06050303,0xf701f6f6,0x1c120e0e,0xc2a36161,
0x6a5f3535,0xaef95757,0x69d0b9b9,0x17918686,0x9958c1c1,0x3a271d1d,0x27b99e9e,
0xd938e1e1,0xeb13f8f8,0x2bb39898,0x22331111,0xd2bb6969,0xa970d9d9,0x07898e8e,
0x33a79494,0x2db69b9b,0x3c221e1e,0x15928787,0xc920e9e9,0x8749cece,0xaaff5555,
0x50782828,0xa57adfdf,0x038f8c8c,0x59f8a1a1,0x09808989,0x1a170d0d,0x65dabfbf,
0xd731e6e6,0x84c64242,0xd0b86868,0x82c34141,0x29b09999,0x5a772d2d,0x1e110f0f,
0x7bcbb0b0,0xa8fc5454,0x6dd6bbbb,0x2c3a1616};
```

C Version of KCipher-2

Date	**Version**	**Change History**
Jan. 2007	K2 Ver.1.0[6]	First published in a international conference.
Jul. 2007	K2 Ver.2.0[7]	The key loading step in the initialization process has been changed in order to diffuse the key and an initial vector into the internal state more efficiently.
∗ 2008	KCipher-2 Ver.2.0	Only the name of the cipher has been changed from ‘K2’ to ‘KCipher-2’.