Specification of MISTY1 — a 64-bit Block Cipher

Version 1.00

Mitsuru MATSUI

Mitsubishi Electric Corporation

September 18, 2000

Contents

1 Introduction

This document shows a complete description of MISTY1, which is a secret-key cipher with a 64-bit data block, a 128-bit secret key and a variable number of rounds n, where n is a multiple of four.

2 Notations and Conventions

2.1 Radix

We use the prefix **0x** to indicate **hexadecimal** numbers.

2.2 Notations

Throughout this document, the following notations are used.

- An element with the suffix $_{(n)}$ (e.g. $x_{(n)}$) shows that the element is n-bit long.
- An element with the suffix $_L$ (e.g. x_L) denotes left-half part of x.
- An element with the suffix $_R$ (e.g. x_R) denotes right-half part of x.

The suffix $_{(n)}$ will be omitted if no ambiguity is expected. See section 2.4 for numerical examples of "left" and "right".

2.3 List of Symbols

$\oplus$	The bitwise exclusive-OR operation.
$\|\|$	The concatenation of the two operands.
$\cap$	The bitwise AND operation.
$\cup$	The bitwise OR operation.

2.4 Bit/Byte Ordering

We adopt big endian ordering. The following example shows how to compose a 64-bit value $L_{(64)}$ of two 32-bit values $W_{i(32)}$ $(i = 1, 2)$, four 16-bit values $H_{i(16)}$ $(i = 1, 2, 3, 4)$, eight 8-bit values $B_{i(8)}$ $(i = 1, 2, \ldots, 8)$, or 64 1-bit values $E_{i(1)}$ $(i = 1, 2, \ldots, 64)$, respectively.

$$
\begin{aligned}
L_{(64)} &= W_{1(32)}||W_{2(32)} \\
&= H_{1(16)}||H_{2(16)}||H_{3(16)}||H_{4(16)} \\
&= B_{1(8)}||B_{2(8)}||B_{3(8)}||B_{4(8)}||B_{5(8)}||B_{6(8)}||B_{7(8)}||B_{8(8)} \\
&= E_{1(1)}||E_{2(1)}||E_{3(1)}||E_{4(1)}||\ldots\ldots\ldots\ldots\ldots\ldots||E_{61(1)}||E_{62(1)}||E_{63(1)}||E_{64(1)}
\end{aligned}
$$

Numerical examples:

$$
\begin{array}{lllllllllllll}
L_{(64)} & = & \texttt{0x0123456789ABCDEF}_{(64)} & & & & & & & & & & \\
W_{1(32)} & = & L_{L(32)} & = & \texttt{0x01234567}_{(32)} & & & & & & & & \\
W_{2(32)} & = & L_{R(32)} & = & \texttt{0x89ABCDEF}_{(32)} & & & & & & & & \\
H_{1(16)} & = & W_{1L(16)} & = & \texttt{0x0123}_{(16)} & & & & & & & & \\
H_{2(16)} & = & W_{1R(16)} & = & \texttt{0x4567}_{(16)} & & & & & & & & \\
H_{3(16)} & = & W_{2L(16)} & = & \texttt{0x89AB}_{(16)} & & & & & & & & \\
H_{4(16)} & = & W_{2R(16)} & = & \texttt{0xCDEF}_{(16)} & & & & & & & & \\
B_{1(8)} & = & \texttt{0x01}_{(8)}, & B_{2(8)} & = & \texttt{0x23}_{(8)}, & B_{3(8)} & = & \texttt{0x45}_{(8)}, & B_{4(8)} & = & \texttt{0x67}_{(8)}, \\
B_{5(8)} & = & \texttt{0x89}_{(8)}, & B_{6(8)} & = & \texttt{0xAB}_{(8)}, & B_{7(8)} & = & \texttt{0xCD}_{(8)}, & B_{8(8)} & = & \texttt{0xEF}_{(8)}, \\
E_{1(1)} & = & \texttt{0}_{(1)}, & E_{2(1)} & = & \texttt{0}_{(1)}, & E_{3(1)} & = & \texttt{0}_{(1)}, & E_{4(1)} & = & \texttt{0}_{(1)}, \\
E_{5(1)} & = & \texttt{0}_{(1)}, & E_{6(1)} & = & \texttt{0}_{(1)}, & E_{7(1)} & = & \texttt{0}_{(1)}, & E_{8(1)} & = & \texttt{1}_{(1)}, \\
 & \vdots & & & & & & & & & & & \\
E_{57(1)} & = & \texttt{1}_{(1)}, & E_{58(1)} & = & \texttt{1}_{(1)}, & E_{59(1)} & = & \texttt{1}_{(1)}, & E_{60(1)} & = & \texttt{0}_{(1)}, \\
E_{61(1)} & = & \texttt{1}_{(1)}, & E_{62(1)} & = & \texttt{1}_{(1)}, & E_{63(1)} & = & \texttt{1}_{(1)}, & E_{64(1)} & = & \texttt{1}_{(1)}.
\end{array}
$$

3 Structure of MISTY1

3.1 List of Functions and Variables

$P_{(64)}$	The plaintext block.
$C_{(64)}$	The ciphertext block.
$K_{(128)}$	The secret key, which is 128-bit long.
$KL_{t(32)}, KO_{u(64)}, KI_{v(48)}, KI_{vw(16)}$	The subkeys of MISTY1 with n rounds. $(t = 1, 2, \ldots, n+2)$ $(u = 1, 2, \ldots, n)$ $(v = 1, 2, \ldots, n)$ $(w = 1, 2, 3)$
$Y_{(32)} = FL(X_{(32)}, KL_{(32)})$	The function FL that transforms a 32-bit input $X_{(32)}$ to a 32-bit output $Y_{(32)}$ using a 32-bit subkey $KL_{(32)}$.
$Y_{(32)} = FL^{-1}(X_{(32)}, KL_{(32)})$	The function FL^{-1} that transforms a 32-bit input $X_{(32)}$ to a 32-bit output $Y_{(32)}$ using a 32-bit subkey $KL_{(32)}$.
$Y_{(32)} = FO(X_{(32)}, KO_{(64)}, KI_{(48)})$	The function FO that transforms a 32-bit input $X_{(32)}$ to a 32-bit output $Y_{(32)}$ using a 64-bit subkey $KO_{(64)}$ and a 48-bit subkey $KI_{(48)}$.
$Y_{(16)} = FI(X_{(16)}, KI_{vw(16)})$	The function FI that transforms a 16-bit input $X_{(16)}$ to a 16-bit output $Y_{(16)}$ using a 16-bit subkey $KI_{vw(16)}$.
$y_{(7)} = S7(x_{(7)})$	The look-up table $S7$ that transforms an 7-bit input $x_{(7)}$ to an 7-bit output $y_{(7)}$.
$y_{(9)} = S9(x_{(9)})$	The look-up table $S9$ that transforms an 9-bit input $x_{(9)}$ to an 9-bit output $y_{(9)}$.

3.2 Data Randomizing Part

Figure 1 and 2 show the encryption and decryption procedure of MISTY1, respectively. In MISTY1, for the sake of flexibility of their security level, the number of rounds n is variable on condition that n is a multiple of four.

3.2.1 Encryption Procedure

Figure 1 shows the encryption procedure of MISTY1 with n rounds. The 64-bit plaintext $P_{(64)}$ is divided into the left 32-bit string and the right 32-bit string, which are transformed into the 64-bit ciphertext $C_{(64)}$ by means of bitwise XOR operations and sub-functions FO_i $(1 \leq i \leq n)$ and FL_i $(1 \leq i \leq n+2)$. FO_i uses a 64-bit subkey KO_i and a 48-bit subkey KI_i. FL_i uses a 32-bit subkey KL_i. The key scheduling part generates these subkeys from the secret key $K_{(128)}$; see section 3.3 for details of the key scheduling part.

In the data randomizing part, first the plaintext $P_{(64)}$ is separated into $L_{0(32)}$ and $R_{0(32)}$ of equal length, i.e., $P_{(64)} = L_{0(32)}||R_{0(32)}$. Then, the following operations are performed.

For odd rounds $(i = 1, 3, ..., n-1)$ we define:

$$\begin{aligned} R_i &= FL_i(L_{i-1}, KL_i), \\ L_i &= FL_{i+1}(R_{i-1}, KL_{i+1}) \oplus FO_i(R_i, KO_i, KI_i). \end{aligned}$$

For even rounds $(i = 2, 4, ..., n)$ we define:

$$\begin{aligned} R_i &= L_{i-1}, \\ L_i &= R_{i-1} \oplus FO_i(R_i, KO_i, KI_i). \end{aligned}$$

After the last round, the FL functions are carried out again;

$$\begin{aligned} R_{n+1} &= FL_{n+1}(L_n, KL_{n+1}), \\ L_{n+1} &= FL_{n+2}(R_n, KL_{n+2}). \end{aligned}$$

Finally, $L_{n+1(32)}$ and $R_{n+1(32)}$ are concatenated. The resultant value is the ciphertext, i.e., $C_{(64)} = L_{n+1(32)}||R_{n+1(32)}$. See section 4 for details of the function FO and FL.

3.2.2 Decryption Procedure

The decryption procedure of MISTY1 can be done in the same way as the encryption by reversing the order of the subkeys and replacing the function FL with the function FL^{-1}.

Figure 2 shows the decryption procedure of MISTY1 with n rounds. The 64-bit ciphertext $C_{(64)}$ is divided into the left 32-bit string and the right 32-bit string, which are transformed into the 64-bit plaintext $P_{(64)}$ by means of bitwise XOR operations and sub-functions FO_i $(n \geq i \geq 1)$ and FL_i $(n+2 \geq i \geq 1)$.

In the data randomizing part, first the ciphertext $C_{(64)}$ is separated into $L_{n+1(32)}$ and $R_{n+1(32)}$ of equal length, i.e., $C_{(64)} = L_{n+1(32)}||R_{n+1(32)}$. Then, the following operations are performed.

For odd rounds $(i = n, \cdots, 4, 2)$ we define:

$$\begin{aligned} R_i &= {FL^{-1}}_{i+2}(L_{i+1}, KL_{i+2}), \\ L_i &= {FL^{-1}}_{i+1}(R_{i+1}, KL_{i+1}) \oplus FO_i(R_i, KO_i, KI_i). \end{aligned}$$

For even rounds $(i = n-1, \cdots, 3, 1)$ we define:

$$\begin{aligned} R_i &= L_{i+1}, \\ L_i &= R_{i+1} \oplus FO_i(R_i, KO_i, KI_i). \end{aligned}$$

After the last round, the FL functions are carried out again:

$$\begin{aligned} R_0 &= {FL^{-1}}_2(L_1, KL_2), \\ L_0 &= {FL^{-1}}_1(R_1, KL_1). \end{aligned}$$

Finally, L_0 and R_0 are concatenated. The resultant value is the plaintext, i.e., $P_{(64)} = L_{0(32)}||R_{0(32)}$. See section 4 for details of the function FO and FL^{-1}.

3.3 Key Scheduling Part

MISTY1 has a 128-bit secret key $K_{(128)}$, which is subdivided into eight 16-bit values $K_1, K_2, ..., K_8$ where

$$K_{(128)} = K_{1(16)}||K_{2(16)}||K_{3(16)}||K_{4(16)}||K_{5(16)}||K_{6(16)}||K_{7(16)}||K_{8(16)}.$$

A second array of subkeys, $K'_i (1 \leq i \leq 8)$ is the output of FI when the input of FI is assigned to K_i and the key KI is set to K_{i+1}, where K_9 is identified with K_1, as shown in figure 7:

$$\begin{aligned} K'_{1(16)} &= FI(K_{1(16)}, K_{2(16)}), \\ K'_{2(16)} &= FI(K_{2(16)}, K_{3(16)}), \\ K'_{3(16)} &= FI(K_{3(16)}, K_{4(16)}), \\ K'_{4(16)} &= FI(K_{4(16)}, K_{5(16)}), \\ K'_{5(16)} &= FI(K_{5(16)}, K_{6(16)}), \\ K'_{6(16)} &= FI(K_{6(16)}, K_{7(16)}), \\ K'_{7(16)} &= FI(K_{7(16)}, K_{8(16)}), \\ K'_{8(16)} &= FI(K_{8(16)}, K_{1(16)}). \end{aligned}$$

The correspondence between the round subkeys KO_{ij}, KI_{ij}, KL_{ij} and the actual subkeys K_i, K'_i is shown in table 1, where i is identified with $i - 8$ when $i > 8$:

Table 1: Subkey mapping table

Round	KO_{i1}	KO_{i2}	KO_{i3}	KO_{i4}	KI_{i1}	KI_{i2}	KI_{i3}	KL_{iL}	KL_{iR}
Actual	K_i	K_{i+2}	K_{i+7}	K_{i+4}	K'_{i+5}	K'_{i+1}	K'_{i+3}	$K_{\frac{i+1}{2}}$(odd i)	$K'_{\frac{i+1}{2}+6}$(odd i)
								$K'_{\frac{i}{2}+2}$(even i)	$K_{\frac{i}{2}+4}$(even i)

4 Components of MISTY1

4.1 Function *FL*

The FL-function is shown in figure 3. The input to the function FL comprises a 32-bit data input $X_{(32)}$ and a 32-bit subkey $KL_{i(32)}$. The input data is split into two 16-bit halves, $X_{L(16)}$ and $X_{R(16)}$

where

$$X_{(32)} = X_{L(16)}||X_{R(16)}.$$

The subkey is split into two 16-bit subkeys, $KL_{iL(16)}$ and $KL_{iR(16)}$ where

$$KL_{i(32)} = KL_{iL(16)}||KL_{iR(16)}.$$

We define:

$$\begin{aligned} Y_{R(16)} &= (X_{L(16)} \cap KL_{iL(16)}) \oplus X_{R(16)}, \\ Y_{L(16)} &= (Y_{R(16)} \cup KL_{iR(16)}) \oplus X_{L(16)}. \end{aligned}$$

The function returns the 32-bit value, i.e., $Y_{(32)} = Y_{L(16)}||Y_{R(16)}$.

4.2 Function FL^{-1}

The function FL^{-1} is shown in figure 4. The input to the function $FL^{-1}{}_i$ comprises a 32-bit data input $Y_{(32)}$ and a 32-bit subkey $KL_{i(32)}$. The input data is split into two 16-bit halves, $Y_{L(16)}$ and $Y_{R(16)}$ where

$$Y_{(32)} = Y_{L(16)}||X_{R(16)}.$$

The subkey is split into two 16-bit subkeys, $KL_{iL(16)}$ and $KL_{iR(16)}$ where

$$KL_{i(32)} = KL_{iL(16)}||KL_{iR(16)}.$$

We define:

$$\begin{aligned} X_{L(16)} &= (Y_{R(16)} \cup KL_{iR(16)}) \oplus Y_{L(16)}, \\ X_{R(16)} &= (X_{L(16)} \cap KL_{iL(16)}) \oplus Y_{R(16)}. \end{aligned}$$

The function returns the 32-bit value, i.e., $X_{(32)} = X_{L(16)}||X_{R(16)}$.

4.3 Function FO

The function FO is shown in figure 5. The input to the function FO_i comprises a 32-bit data input $X_{i(32)}$ and two sets of subkeys, a 64-bit subkey $KO_{i(64)}$ and 48-bit subkey $KI_{i(48)}$. The input data is split into two 16-bit halves, L_0 and R_0 where

$$X_{i(32)} = L_{0(16)}||R_{0(16)}.$$

The subkeys are subdivided into 16-bit subkeys where

$$\begin{aligned} KO_{i(64)} &= KO_{i1(16)}||KO_{i2(16)}||KO_{i3(16)}||KO_{i4(16)}, \\ KI_{i(48)} &= KI_{i1(16)}||KI_{i2(16)}||KI_{i3(16)}. \end{aligned}$$

Then for each integer j with $1 \leq j \leq 3$ we define:

$$\begin{aligned} R_j &= FI_{ij}(L_{j-1} \oplus KO_{ij}, KI_{ij}) \oplus R_{j-1}, \\ L_j &= R_{j-1}. \end{aligned}$$

Finally, $L_{3(16)}$ is XORed with KO_{i4} and concatenated with $R_{3(16)}$. The function returns the 32-bit value, i.e., $Y_{i(32)} = (L_{3(16)} \oplus KO_{i4})||R_{3(16)}$. See sections 4.4 for the function FI.

4.4 Function *FI*

The function FI is shown in figure 6. The function FI_j takes a 16-bit data input $X_{j(16)}$ and 16-bit subkey $KI_{ij(16)}$. The input data is split into two unequal components, a 9-bit left half $L_{0(9)}$ and a 7-bit right half $R_{0(7)}$ where $X_{j(16)} = L_{0(9)}||R_{0(7)}$. Similary the key $KI_{ij(16)}$ is split into two unequal components, a 7-bit component $KI_{ijL(7)}$ and a 9-bit componet $KI_{ijR(9)}$ where $KI_{ij(16)} = KI_{ijL(7)}||KI_{ijR(9)}$.

The function FI uses two S-boxes, $S7$ which maps a 7-bit input to 7-bit output, and $S9$ which maps a 9-bit input to 9-bit output. These are fully defined in section 4.5. It also uses two additional functions which we designate $ZE()$ and $TR()$. We define these as:

$y_{(9)} = ZE(x_{(7)})$	The function ZE that takes the 7-bit value $x_{(7)}$ and converts it to a 9-bit value $y_{(9)}$ by adding two zero bits to the most-signigicant end.
$y_{(7)} = TR(x_{(9)})$	The function TR that takes the 9-bit value $x_{(9)}$ and converts it to a 7-bit value $y_{(7)}$ by discarding the two most-signigicant bits.

We define the following series of operations:

$$\begin{aligned} L_{1(7)} &= R_{0(7)}, & R_{1(9)} &= S9(L_{0(9)}) \oplus ZE(R_{0(7)}); \\ L_{2(9)} &= R_{1(9)} \oplus KI_{ijR(9)}, & R_{2(7)} &= S7(L_{1(7)}) \oplus TR(R_{1(9)}) \oplus KI_{ijL(7)}; \\ L_{3(7)} &= R_{2(7)}, & R_{3(9)} &= S9(L_{2(9)}) \oplus ZE(R_{2(7)}); \end{aligned}$$

Finally, $L_{3(7)}$ and $R_{3(9)}$ are concatenated. The function returns the 16-bit value, i.e., $Y_{(16)} = L_{3(7)}||R_{3(9)}$.

4.5 *S*-boxes

The two S-boxes have been designed so that they may be easily implemented in combinational logic as well as by a look-up table. Both forms are given for each table. In the logic equations, $x_1x_2x_3$ implies $x_1 \cap x_2 \cap x_3$.

4.5.1 *S7*

Gate Logic:

$$y_7 = x_7 \oplus x_6x_4 \oplus x_7x_4x_3 \oplus x_6x_2 \oplus x_7x_5x_2 \oplus x_3x_2 \oplus x_7x_6x_1 \oplus x_5x_1 \oplus x_7x_2x_1 \oplus x_4x_2x_1 \oplus 1$$

$$
\begin{aligned}
y_6 &= x_7x_5 \oplus x_7x_3 \oplus x_4x_3 \oplus x_6x_2 \oplus x_5x_3x_2 \oplus x_1 \oplus x_7x_1 \oplus x_4x_1 \oplus x_5x_4x_1 \oplus x_6x_3x_1 \oplus x_7x_2x_1 \oplus 1 \\
y_5 &= x_6x_5 \oplus x_7x_5x_4 \oplus x_3 \oplus x_6x_3 \oplus x_7x_6x_3 \oplus x_7x_2 \oplus x_7x_3x_2 \oplus x_4x_3x_2 \oplus x_6x_1 \oplus x_4x_1 \oplus x_7x_4x_1 \\
&\quad \oplus x_3x_1 \oplus x_5x_3x_1 \\
y_4 &= x_7 \oplus x_6 \oplus x_7x_6x_5 \oplus x_7x_4 \oplus x_5x_3 \oplus x_6x_3x_2 \oplus x_5x_1 \oplus x_6x_4x_1 \oplus x_7x_3x_1 \oplus x_2x_1 \oplus 1 \\
y_3 &= x_5x_4 \oplus x_7x_3 \oplus x_6x_4x_3 \oplus x_2 \oplus x_5x_2 \oplus x_6x_5x_2 \oplus x_7x_4x_2 \oplus x_6x_1 \oplus x_6x_2x_1 \oplus x_3x_2x_1 \oplus 1 \\
y_2 &= x_7 \oplus x_6 \oplus x_5 \oplus x_7x_6x_5 \oplus x_7x_4 \oplus x_6x_5x_4 \oplus x_6x_3 \oplus x_7x_5x_3 \oplus x_7x_2 \oplus x_7x_6x_2 \oplus x_4x_2 \oplus x_7x_1 \\
&\quad \oplus x_5x_2x_1 \\
y_1 &= x_7x_6 \oplus x_4 \oplus x_7x_4 \oplus x_5x_4x_3 \oplus x_7x_2 \oplus x_5x_2 \oplus x_4x_2 \oplus x_6x_4x_2 \oplus x_6x_1 \oplus x_6x_5x_1 \oplus x_7x_4x_1 \\
&\quad \oplus x_3x_1 \oplus x_5x_2x_1
\end{aligned}
$$

Decimal Table:

27	50	51	90	59	16	23	84	91	26	114	115	107	44	102	73
31	36	19	108	55	46	63	74	93	15	64	86	37	81	28	4
11	70	32	13	123	53	68	66	43	30	65	20	75	121	21	111
14	85	9	54	116	12	103	83	40	10	126	56	2	7	96	41
25	18	101	47	48	57	8	104	95	120	42	76	100	69	117	61
89	72	3	87	124	79	98	60	29	33	94	39	106	112	77	58
1	109	110	99	24	119	35	5	38	118	0	49	45	122	127	97
80	34	17	6	71	22	82	78	113	62	105	67	52	92	88	125

4.5.2 *S*9

Gate Logic:

$$
\begin{aligned}
y_9 &= x_9x_5 \oplus x_9x_4 \oplus x_8x_4 \oplus x_8x_3 \oplus x_7x_3 \oplus x_7x_2 \oplus x_6x_2 \oplus x_6x_1 \oplus x_5x_1 \oplus 1 \\
y_8 &= x_9x_7 \oplus x_6 \oplus x_8x_6 \oplus x_7x_6 \oplus x_6x_5 \oplus x_5x_4 \oplus x_9x_3 \oplus x_7x_3 \oplus x_2 \oplus x_9x_1 \oplus x_6x_1 \oplus x_4x_1 \oplus 1 \\
y_7 &= x_9x_8 \oplus x_8x_6 \oplus x_5 \oplus x_9x_5 \oplus x_7x_5 \oplus x_6x_5 \oplus x_5x_4 \oplus x_9x_3 \oplus x_4x_3 \oplus x_8x_2 \oplus x_6x_2 \oplus x_1 \\
y_6 &= x_9 \oplus x_8x_7 \oplus x_7x_5 \oplus x_4 \oplus x_8x_4 \oplus x_6x_4 \oplus x_5x_4 \oplus x_4x_3 \oplus x_8x_2 \oplus x_3x_2 \oplus x_7x_1 \oplus x_5x_1 \\
y_5 &= x_8 \oplus x_9x_6 \oplus x_7x_6 \oplus x_9x_4 \oplus x_6x_4 \oplus x_3 \oplus x_7x_3 \oplus x_5x_3 \oplus x_4x_3 \oplus x_3x_2 \oplus x_7x_1 \oplus x_2x_1 \\
y_4 &= x_7 \oplus x_9x_6 \oplus x_8x_5 \oplus x_6x_5 \oplus x_8x_3 \oplus x_5x_3 \oplus x_2 \oplus x_6x_2 \oplus x_4x_2 \oplus x_3x_2 \oplus x_9x_1 \oplus x_2x_1 \\
y_3 &= x_9x_8 \oplus x_6 \oplus x_8x_5 \oplus x_7x_4 \oplus x_5x_4 \oplus x_7x_2 \oplus x_4x_2 \oplus x_1 \oplus x_9x_1 \oplus x_5x_1 \oplus x_3x_1 \oplus x_2x_1 \oplus 1 \\
y_2 &= x_8 \oplus x_9x_8 \oplus x_8x_7 \oplus x_7x_6 \oplus x_9x_5 \oplus x_4 \oplus x_8x_3 \oplus x_6x_3 \oplus x_9x_2 \oplus x_5x_2 \oplus x_3x_2 \oplus x_8x_1 \oplus 1 \\
y_1 &= x_9 \oplus x_9x_8 \oplus x_8x_7 \oplus x_5 \oplus x_9x_4 \oplus x_7x_4 \oplus x_6x_3 \oplus x_4x_3 \oplus x_9x_2 \oplus x_9x_1 \oplus x_6x_1 \oplus x_3x_1 \oplus 1
\end{aligned}
$$

Decimal Table:

451	203	339	415	483	233	251	53	385	185	279	491	307	9	45	211
199	330	55	126	235	356	403	472	163	286	85	44	29	418	355	280
331	338	466	15	43	48	314	229	273	312	398	99	227	200	500	27
1	157	248	416	365	499	28	326	125	209	130	490	387	301	244	414
467	221	482	296	480	236	89	145	17	303	38	220	176	396	271	503
231	364	182	249	216	337	257	332	259	184	340	299	430	23	113	12
71	88	127	420	308	297	132	349	413	434	419	72	124	81	458	35
317	423	357	59	66	218	402	206	193	107	159	497	300	388	250	406
481	361	381	49	384	266	148	474	390	318	284	96	373	463	103	281
101	104	153	336	8	7	380	183	36	25	222	295	219	228	425	82
265	144	412	449	40	435	309	362	374	223	485	392	197	366	478	433
195	479	54	238	494	240	147	73	154	438	105	129	293	11	94	180
329	455	372	62	315	439	142	454	174	16	149	495	78	242	509	133
253	246	160	367	131	138	342	155	316	263	359	152	464	489	3	510
189	290	137	210	399	18	51	106	322	237	368	283	226	335	344	305
327	93	275	461	121	353	421	377	158	436	204	34	306	26	232	4
391	493	407	57	447	471	39	395	198	156	208	334	108	52	498	110
202	37	186	401	254	19	262	47	429	370	475	192	267	470	245	492
269	118	276	427	117	268	484	345	84	287	75	196	446	247	41	164
14	496	119	77	378	134	139	179	369	191	270	260	151	347	352	360
215	187	102	462	252	146	453	111	22	74	161	313	175	241	400	10
426	323	379	86	397	358	212	507	333	404	410	135	504	291	167	440
321	60	505	320	42	341	282	417	408	213	294	431	97	302	343	476
114	394	170	150	277	239	69	123	141	325	83	95	376	178	46	32
469	63	457	487	428	68	56	20	177	363	171	181	90	386	456	468
24	375	100	207	109	256	409	304	346	5	288	443	445	224	79	214
319	452	298	21	6	255	411	166	67	136	80	351	488	289	115	382
188	194	201	371	393	501	116	460	486	424	405	31	65	13	442	50
61	465	128	168	87	441	354	328	217	261	98	122	33	511	274	264
448	169	285	432	422	205	243	92	258	91	473	324	502	173	165	58
459	310	383	70	225	30	477	230	311	506	389	140	143	64	437	190
120	0	172	272	350	292	2	444	162	234	112	508	278	348	76	450

A Figures of the MISTY1 Algorithm

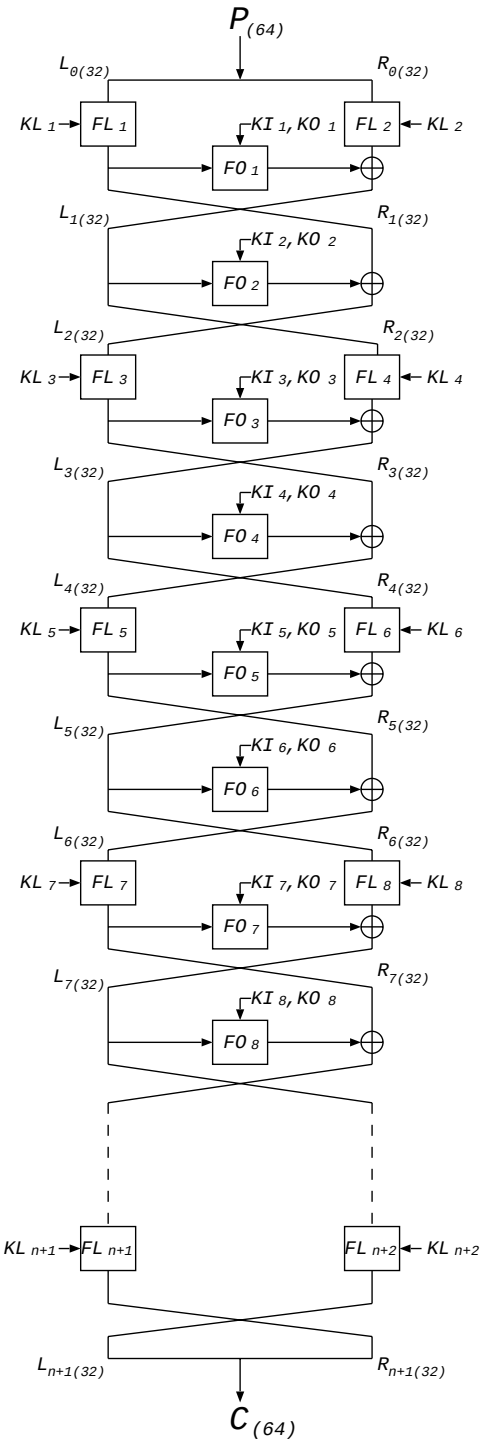

Figure 1: Encryption Procedure of MISTY1

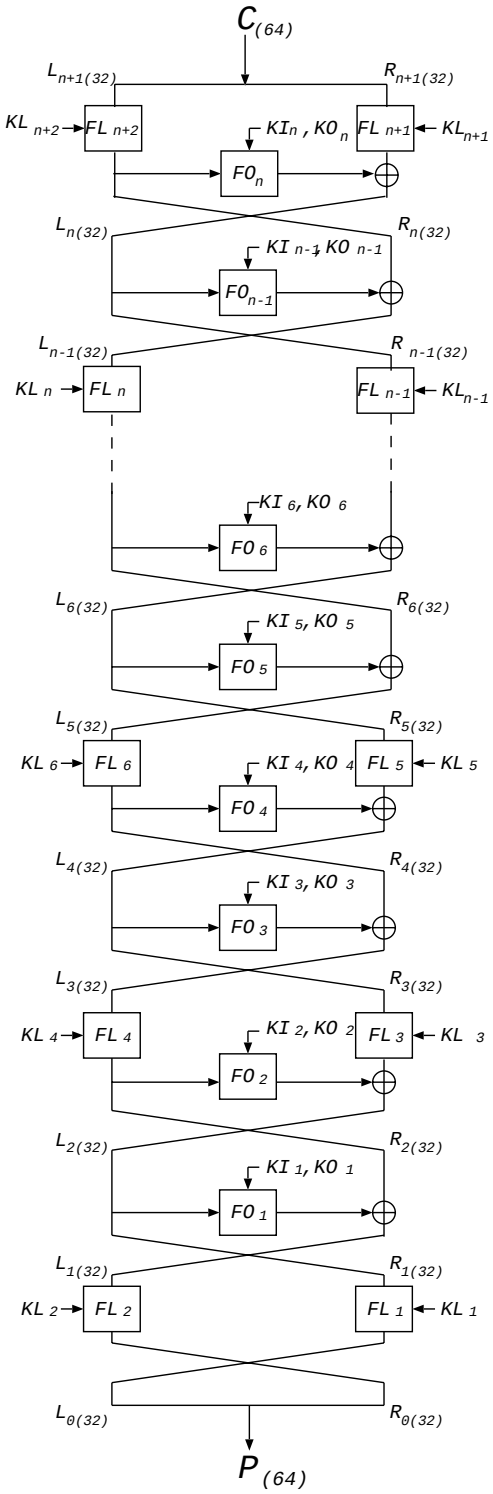

Figure 2: Decryption Procedure of MISTY1

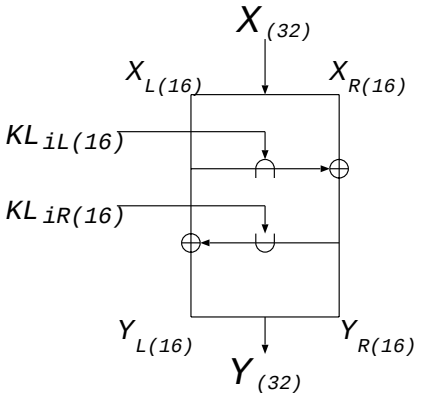

Figure 3: FL_i

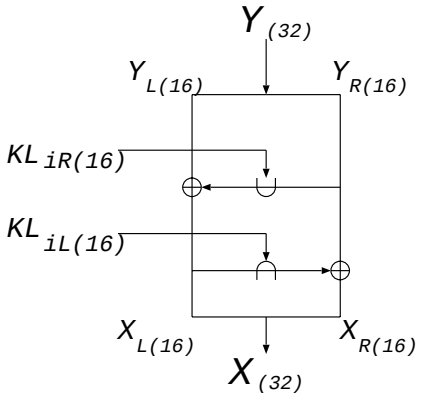

Figure 4: $FL^{-1}{}_i$

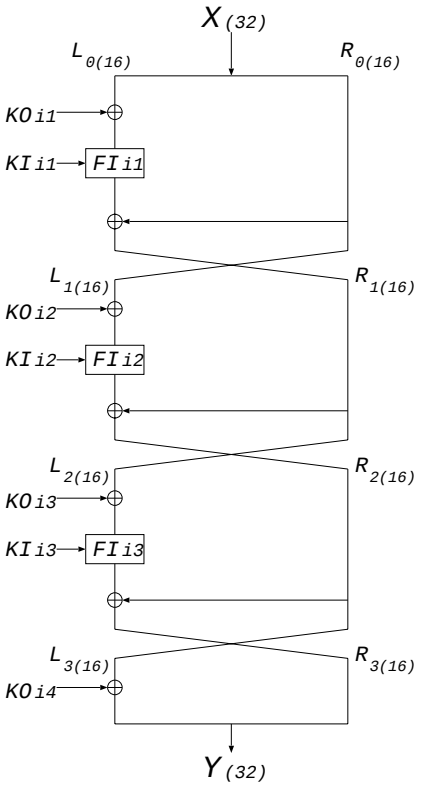

Figure 5: FO_i

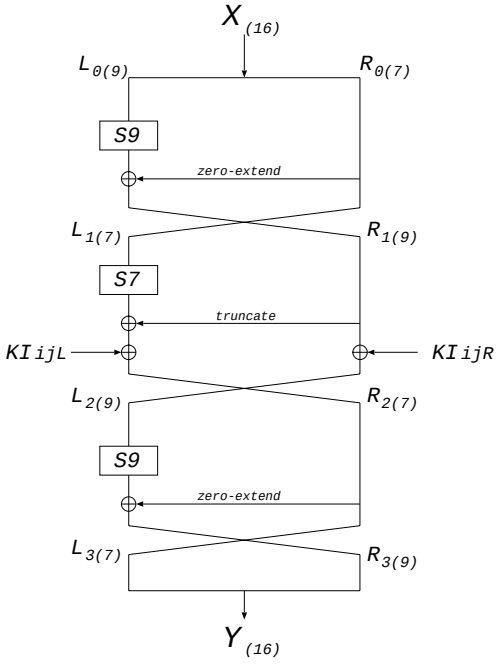

Figure 6: FI_{ij}

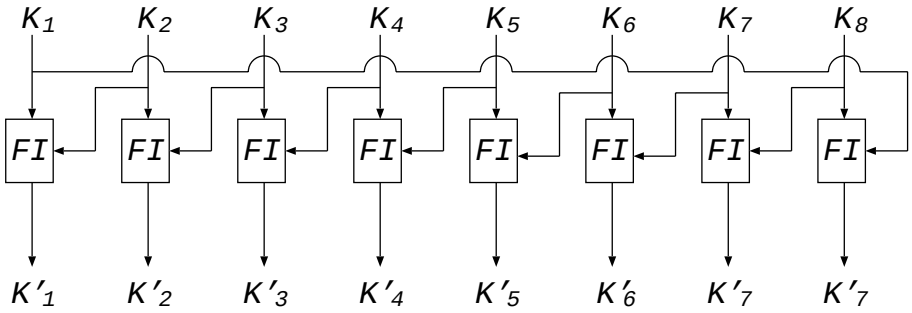

Figure 7: Key Scheduling

B Test Data

The following is test data for MISTY1 with eight rounds in hexadecimal form:

Key (K_1 to K_8)	`00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff`
Subkey (K'_1 to K'_8)	`cf 51 8e 7f 5e 29 67 3a cd bc 07 d6 bf 35 5e 11`
Plaintext	`01 23 45 67 89 ab cd ef`
Ciphertext	`8b 1d a5 f5 6a b3 d0 7c`