Chapter 7

Block Cipher Design

7.1 Introduction

In this chapter we further elaborate our block cipher design strategy that was introduced in Chapter 4. We start with describing a new type of self-reciprocal cipher structure that is a widely applicable alternative for the Feistel round structure. The round transformation is composed of a small number of simple basic transformations that must satisfy certain algebraic conditions.

We describe a number of transformations that are especially suited for the proposed cipher structure. For each of these transformations the difference propagation and correlation properties are treated. We show that, for the proposed cipher structure and basic transformations, the behavior of differential and linear trails are governed by the same equations. This allows the evaluation and optimization of the resistance against LC and DC in a single effort. We also discuss the risk of weaknesses due to symmetry and the measures that have to be taken with respect to these aspects.

This chapter contains two fully specified block ciphers with high portability and a short and elegant description. For both we give our findings with respect to LC and DC. We conclude with describing an example filtered counter stream encryption scheme that can be built using one of the two proposed block ciphers. A large part of this chapter has already been published in our paper [22].

7.2 A self-reciprocal cipher structure

The basic building blocks of our block ciphers are a number of different invertible transformations, each with its own specific contribution:

γ: a local **nonlinear** transformation,

θ: a local linear transformation for **diffusion**,

σ: bitwise addition of a round **key** κ_j,

π_i: blockwise bit permutations for **bit dispersion**.

These transformations must be arranged into a portable and simple block cipher. For block encryption and decryption to be executable by a single cryptographic finite state machine, this cipher must be structurally self-reciprocal. In this section we show how such a nontrivial self-reciprocal structure can be built by introducing a simple bit permutation μ and imposing that the basic transformations interact with μ in a specific way.

The block cipher consists of a certain number m of iterations of a round transformation $\rho[\kappa_j]$, followed by an output transformation $\omega[\kappa_m]$ and the bit permutation μ. The round keys κ_i are derived from the cipher key κ by the key schedule. The round transformation ρ and the output transformation ω are composed from the basic transformations in the following way:

$$\rho[\kappa_j] = \pi_2 \circ \gamma \circ \pi_1 \circ \theta \circ \sigma[\kappa_j] \ , \tag{7.1}$$

$$\omega[\kappa_m] = \theta \circ \sigma[\kappa_m] \ . \tag{7.2}$$

Consider the simple single-round block cipher

$$\mathrm{B}_1[\kappa_0, \kappa_1] = \mu \circ \omega[\kappa_1] \circ \rho[\kappa_0] \tag{7.3}$$

or

$$\mathrm{B}_1[\kappa_0, \kappa_1] = \mu \circ \theta \circ \sigma[\kappa_1] \circ \pi_2 \circ \gamma \circ \pi_1 \circ \theta \circ \sigma[\kappa_0] \ . \tag{7.4}$$

Its inverse is given by

$$\mathrm{B}_1[\kappa_0, \kappa_1]^{\ 1} = \sigma[\kappa_0] \circ \theta^{\ 1} \circ {\pi_1}^{\ 1} \circ \gamma^{\ 1} \circ {\pi_2}^{\ 1} \circ \sigma[\kappa_1] \circ \theta^{\ 1} \circ \mu^{\ 1} \ . \tag{7.5}$$

Since θ is linear we can switch θ and σ resulting in

$$\mathrm{B}_1[\kappa_0, \kappa_1]^{\ 1} = \theta^{\ 1} \circ \sigma[\theta(\kappa_0)] \circ {\pi_1}^{\ 1} \circ \gamma^{\ 1} \circ {\pi_2}^{\ 1} \circ \theta^{\ 1} \circ \sigma[\theta(\kappa_1)] \circ \mu^{\ 1} \ . \tag{7.6}$$

We choose the basic transformations in such a way that they satisfy the following conditions:

$$\theta^{\ 1} = \mu^{\ 1} \circ \theta \circ \mu \ , \tag{7.7}$$

$$\gamma^{\ 1} = \mu^{\ 1} \circ \gamma \circ \mu \ , \tag{7.8}$$

$${\pi_2}^{\ 1} = \mu^{\ 1} \circ \pi_1 \circ \mu, \tag{7.9}$$

$${\pi_1}^{\ 1} = \mu^{\ 1} \circ \pi_2 \circ \mu \ . \tag{7.10}$$

Using these relations, we can eliminate all occurrences of inverse transformations in (7.6), with the exception of $\mu^{\ 1}$, i.e.,

$$\begin{aligned}
\mathrm{B}_1[\kappa_0,\kappa_1]^{\ 1} &= \theta^{\ 1}\circ\sigma[\theta(\kappa_0)]\circ\pi_1^{\ 1}\circ\gamma^{\ 1}\circ\pi_2^{\ 1}\circ\theta^{\ 1}\circ\sigma[\theta(\kappa_1)]\circ\mu^{\ 1}\\
&= \theta^{\ 1}\circ\sigma[\theta(\kappa_0)]\circ\pi_1^{\ 1}\circ\gamma^{\ 1}\circ\pi_2^{\ 1}\circ\mu^{\ 1}\circ\theta\circ\sigma[\mu(\theta(\kappa_1))]\\
&= \theta^{\ 1}\circ\sigma[\theta(\kappa_0)]\circ\pi_1^{\ 1}\circ\gamma^{\ 1}\circ\mu^{\ 1}\circ\pi_1\circ\theta\circ\sigma[\mu(\theta(\kappa_1))]\\
&= \theta^{\ 1}\circ\sigma[\theta(\kappa_0)]\circ\pi_1^{\ 1}\circ\mu^{\ 1}\circ\gamma\circ\pi_1\circ\theta\circ\sigma[\mu(\theta(\kappa_1))]\\
&= \theta^{\ 1}\circ\sigma[\theta(\kappa_0)]\circ\mu^{\ 1}\circ\pi_2\circ\gamma\circ\pi_1\circ\theta\circ\sigma[\mu(\theta(\kappa_1))]\\
&= \mu^{\ 1}\circ\theta\circ\sigma[\mu(\theta(\kappa_0))]\circ\pi_2\circ\gamma\circ\pi_1\circ\theta\circ\sigma[\mu(\theta(\kappa_1))]\ .
\end{aligned}\tag{7.11}$$

By imposing the additional condition

$$\mu^{\ 1}=\mu\ ,\tag{7.12}$$

we have

$$\mathrm{B}_1[\kappa_0,\kappa_1]^{\ 1}=\mathrm{B}_1[\kappa'_0,\kappa'_1]\ ,\tag{7.13}$$

with the inverse round keys given by $\kappa'_j=\mu(\theta(\kappa_{1\ j}))$. If line 5 of (7.11) is rewritten in terms of the round transformation and the output transformation, we have for $j\geq 1$,

$$\rho[\kappa_{j\ 1}]^{\ 1}\circ\omega[\kappa_j]^{\ 1}\circ\mu^{\ 1}=\omega[\kappa_{j\ 1}]^{\ 1}\circ\mu^{\ 1}\circ\rho[\mu(\theta(\kappa_j))]\ .\tag{7.14}$$

A cipher B_m with m rounds is defined by

$$\mathrm{B}_m[\kappa_0,\kappa_1,\kappa_2,\ldots,\kappa_m]=\mu\circ\omega[\kappa_m]\circ\rho[\kappa_{m\ 1}]\circ\ldots\circ\rho[\kappa_1]\circ\rho[\kappa_0]\ .$$

By iteratively applying (7.14) we obtain

$$\mathrm{B}_m[\kappa_0,\kappa_1,\kappa_2,\ldots,\kappa_m]^{\ 1}=\mathrm{B}_m[\kappa'_0,\kappa'_1,\kappa'_2,\ldots,\kappa'_m]\ ,\tag{7.15}$$

with $\kappa'_j=\mu(\theta(\kappa_{m\ j}))$.

We propose to use a cipher key κ of length equal to the block length. The round keys κ_j are derived from this cipher key by the bitwise addition of so-called *round constants* c^j:

$$\kappa_j=\kappa+\mathrm{c}^j\ .\tag{7.16}$$

If $\kappa'=\mu(\theta(\kappa))$ is considered to be the inverse cipher key, the inverse round keys κ'_j can be derived from κ' in a similar manner: $\kappa'_j=\kappa'+\mathrm{c}'^j$ with

$$\mathrm{c}'^j=\mu(\theta(\mathrm{c}^{m\ j}))\ .\tag{7.17}$$

This cipher structure lends itself to a straightforward implementation as a cryptographic finite state machine. Encryption and decryption can be

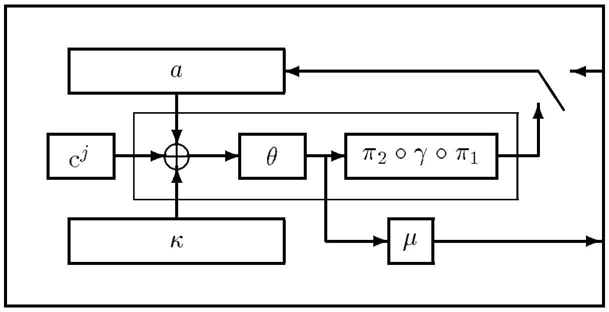

Figure 7.1: Cryptographic finite state machine for the proposed cipher structure.

performed by the same finite state machine with ρ as updating transformation. An encryption operation consists of initializing the state register by loading the plaintext and doing m state updating iterations. During these iterations the key schedule is executed by a separate round-constant generating module. The additional application of the output transformation can be accomplished by reading out the intermediate stage after θ of the round transformation logic, instead of the internal state itself. The bit permutation μ can be hardwired in the connections to the output pins. The block scheme of the resulting cryptographic finite state machine is shown in Fig. 7.1. Additionally, it can be observed from this scheme that if the cipher key is loaded into the state register and the key register is set to 0, the inverse cipher key appears at the output. Hence, the calculation of the inverse cipher key from the cipher key can be executed by the module itself.

If the specific permutation μ is hard to implement in software and the inverses of θ, γ, π_1 and π_2 are easy, a variant of the proposed structure may be more suitable. In this variant the final application of μ in the encrypting operation is omitted. This must be compensated by an additional application of μ at the beginning of decryption. We have

$$\mathrm{B}_m[\kappa_0, \kappa_1, \ldots, \kappa_m] = \omega[\kappa_m] \circ \rho[\kappa_{m\ \ 1}] \circ \ldots \circ \rho[\kappa_1] \circ \rho[\kappa_0] \ , \tag{7.18}$$

and

$$\mathrm{B}_m[\kappa_0, \kappa_1, \ldots, \kappa_m]^{\ \ 1} = \mu \circ \mathrm{B}_m[\kappa'_0, \kappa'_1, \ldots, \kappa_m] \circ \mu \ , \tag{7.19}$$

with $\kappa'_j = \mu(\theta(\kappa_{m\ \ j}))$.

The cryptographic finite state machine implementation scheme of this variant differs from that given in Fig. 7.1 by the presence of two functional

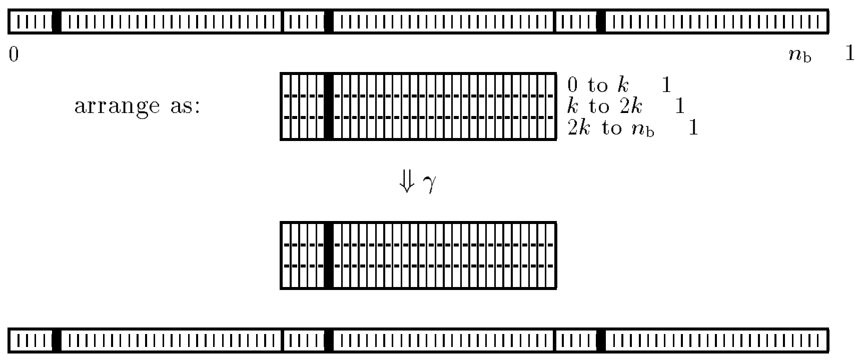

Figure 7.2: Arrangement of the bits in the transformation γ. The bits of an individual triplet are marked in black.

blocks responsible for μ in decryption mode. One of these is placed at the input, the other at the output. These blocks are switched "off" for encryption and "on" for decryption.

In software implementations of this variant the encrypting transformation does not contain μ. In the decrypting transformation the need for μ can be avoided by implementing it using the inverses of θ, γ, π_1 and π_2.

7.3 The Nonlinear transformation γ

The transformation γ is defined for vectors with a dimension n_b divisible by 3. Let $b = \gamma(a)$ with a and b vectors of length $n_b = 3k$. We have

$$b_i = a_i + (a_{i+k} + 1)a_{i+2k} + 1 \ . \tag{7.20}$$

This is in fact a simple variant of the invertible shift-invariant transformation χ. Every 3-tuple, or *triplet* (b_i, b_{i+k}, b_{i+2k}) is completely determined by the triplet (a_i, a_{i+k}, a_{i+2k}), i.e., γ is a juxtaposed transformation consisting of k equal 3-bit substitution boxes. The arrangement of the bits in triplets is illustrated in Fig. 7.2. In this figure it can also be seen that in software implementations a number of substitution boxes equal to the processor word length can be handled simultaneously by using bitwise Boolean operations. The block length n_b must be a multiple of 3. Since computer word lengths are typically powers of 2, we restrict the block length to $n_b = 2^\ell 3$ for some ℓ.

The γ substitution box is given in Table 7.1. Its effect can be described as: a single 1 at the input is shifted one position to the left, a single 0

x	000	001	010	100	110	101	011	111
$\gamma(x)$	111	010	100	001	011	110	101	000

Table 7.1: The 3-bit γ substitution box.

	000	001	010	100	011	101	110	111
000	1	-	-	-	-	-	-	-
001	-	1/4	-	-	1/4	1/4	-	1/4
010	-	-	1/4	-	1/4	-	1/4	1/4
100	-	-	-	1/4	-	1/4	1/4	1/4
011	-	1/4	1/4	-	-	1/4	1/4	-
101	-	1/4	-	1/4	1/4	-	1/4	-
110	-	-	1/4	1/4	1/4	1/4	-	-
111	-	1/4	1/4	1/4	-	-	-	1/4

Table 7.2: Prop ratios for the γ substitution box.

one position to the right and three equal input bits are complemented. The description of the inverse of γ is obtained by simply interchanging the words "left" and "right". Hence, for

$$\gamma^{\ 1} = \mu^{\ 1} \circ \gamma \circ \mu \tag{7.21}$$

to hold, μ must respect the division in triplets and invert the order of the components within the triplets.

7.3.1 Propagation and correlation properties

Table 7.2 lists the prop ratios of difference propagation in the γ substitution box. The input differences are listed above and the output differences at the left. Every nonzero input (output) difference is compatible with exactly four output (input) differences. All nontrivial difference propagations have a prop ratio of 1/4 and a restriction weight of 2. An input difference triplet and an output difference triplet are *compatible* if they have an odd number of 1-bits in common, i.e., if the parity of their bitwise product (AND) is odd.

An input difference and an output difference to γ are compatible if all their component triplets are compatible. If an input difference has ℓ nonzero triplets it is compatible with $2^{2\ell}$ different output differences. All

	000	001	010	100	011	101	110	111
000	1	-	-	-	-	-	-	-
001	-	1/2	-	-	1/2	1/2	-	1/2
010	-	-	1/2	-	1/2	-	1/2	1/2
100	-	-	-	1/2	-	1/2	1/2	1/2
011	-	1/2	1/2	-	-	1/2	1/2	-
101	-	1/2	-	1/2	1/2	-	1/2	-
110	-	-	1/2	1/2	1/2	1/2	-	-
111	-	1/2	1/2	1/2	-	-	-	1/2

Table 7.3: Correlation matrix of the γ substitution box.

possible difference propagations from this input difference have restriction weight 2ℓ. Hence, $\mathrm{w_r}(a')$, the restriction weight of a difference vector a' with respect to γ, is equal to twice the number of nonzero component triplets in a'.

Table 7.3 gives the correlation matrix of the γ substitution box. Every nonzero output selection triplet is correlated with exactly four input selection triplets, all with correlation $\pm 1/2$. By comparing Tables 7.3 and 7.2 it can be seen that the condition for compatibility between input and output selection triplets is the same as that between input and output differences. This is not the case in general but a consequence of the symmetry properties of γ.

An input selection and an output selection to γ are compatible if all their component triplet selections are compatible. Since linear combinations corresponding to different triplets are disjunct, $\mathrm{C}(u^{\mathrm{t}}\gamma(a), w^{\mathrm{t}}a)$ is equal to $\pm 2^{\ \ \ell}$ with ℓ the number of nonzero component triplets in u. Every output selection vector is compatible to exactly $2^{2\ell}$ input selection vectors. Hence, the correlation weight of a selection vector $\mathrm{w_c}(u)$ is equal to the number of nonzero component triplets in u. We introduce the *triplet weight* $\mathrm{w_t}(a)$ to be the number of nonzero triplets in a vector. We have

$$\mathrm{w_c}(a) = \mathrm{w_t}(a) \qquad \text{and} \qquad \mathrm{w_r}(a) = 2\mathrm{w_t}(a)\ . \tag{7.22}$$

7.4 The linear transformation θ

The transformation θ is defined for vectors with a dimension n_{b} divisible by 12. Let $b = \theta(a)$, with a and b vectors of length $n_{\mathrm{b}} = 12h$. We have

$$b(x) = e(x^h)a(x) \bmod (1 + x^{12h})\ , \tag{7.23}$$

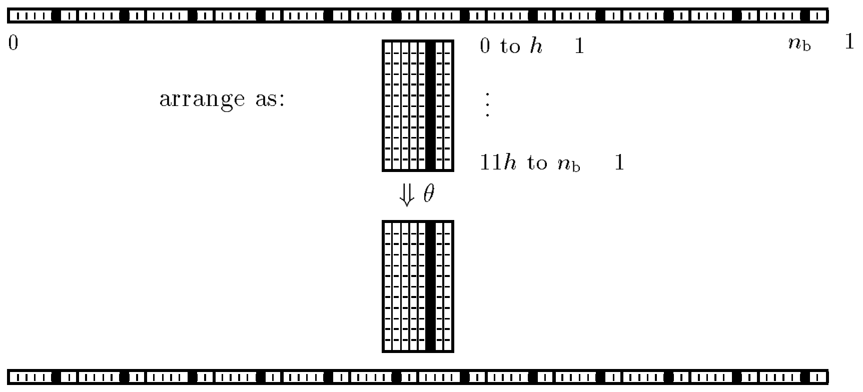

Figure 7.3: Arrangement of the bits in the transformation θ. The bits of an individual 12-tuple are marked in black.

with

$$e(x) = 1 + x + x^2 + x^3 + x^5 + x^6 + x^{10} \ . \tag{7.24}$$

Every 12-tuple $(b_i, b_{i+h}, b_{i+2h}, \ldots, b_{i+11h})$ is completely determined by the 12-tuple $(a_i, a_{i+h}, a_{i+2h}, \ldots, a_{i+11h})$, i.e., θ can be seen as a juxtaposed transformation with h linear 12-bit substitutions boxes. This is illustrated in Fig. 7.3. Similar to γ, in software implementations a number of linear θ substitutions can be handled simultaneously using bitwise Boolean addition.

The inverse of θ consists of multiplication by the inverse polynomial of $e(x^h)$ modulo $1 + x^{n_b}$. The selected polynomial has been chosen from the subclass of polynomials with the following property:

$$e(x)^{\ 1} \equiv e(x^{\ 1}) \pmod{1 + x^{12}} \ , \tag{7.25}$$

hence, the inverse of θ is given by

$$a(x) = e(x^{\ h}) b(x) \bmod 1 + x^{12h} \ . \tag{7.26}$$

The effect of substituting x by $x^{\ 1}$ in the argument of a polynomial $a(x)$ corresponds to a bit permutation in the vector a. It can be seen as a reflection around component 0, interchanging the components in pairs $(i, n_b \ \ i)$. We can rewrite (7.26) as

$$a(x^{\ 1}) = e(x^h) b(x^{\ 1}) \bmod 1 + x^{12h} \ , \tag{7.27}$$

i.e., the inverse of θ applied to the reflected vectors corresponds to θ itself. Hence, for

$$\theta^{\ 1} = \mu^{\ 1} \circ \theta \circ \mu \tag{7.28}$$

to hold, μ must respect the division in 12-tuples and invert the order of the components within the 12-tuples.

The selection of $e(x)$ was governed by two additional design decisions. To have high diffusion we decided that $e(x)$ should have a Hamming weight of 7. This has the additional benefit that in hardware the combination of θ and $\sigma[\kappa_j]$ can be implemented with balanced-tree circuits consisting of three stages with respectively 4, 2 and 1 EXOR gates. The modulus exponent m was chosen to be the smallest value larger than 7 of the form $2^\ell 3$, since m must divide the block length n_b.

7.4.1 Propagation and correlation properties

As explained in Chapter 6, the difference propagation through θ can be expressed as

$$b'(x) = e(x^h)a'(x) \bmod (1 + x^{n_\mathrm{b}}) \ , \tag{7.29}$$

and a linear combination of output bits specified by u is equal to the linear combination of input bits specified by w with

$$w(x) = e(x^{\ h})u(x) \bmod (1 + x^{n_\mathrm{b}}) \ . \tag{7.30}$$

Using (7.25) this can be converted to

$$u(x) = e(x^h)w(x) \bmod (1 + x^{n_\mathrm{b}}) \ . \tag{7.31}$$

Hence, for θ difference propagation and correlation are governed by essentially the same equation. This is not the case in general, nor is it a coincidence. It is the consequence of the design decision specified in (7.25).

The combination of the shift-invariance and (7.25) causes the matrix M_θ corresponding to θ to be orthogonal with respect to $\mathbb{Z}_2^{n_\mathrm{b}}$, i.e.,

$$M_\theta^{\ 1} = M_\theta^{\mathrm{t}} \ . \tag{7.32}$$

The Hamming weight distribution table (see p. 120) of modular multiplication by $e(x)$ is given in Table 7.4. It can be observed that the branch number $\mathcal{B}$ of this linear shift-invariant transformation is 8, the maximum attainable value both for a polynomial with 7 terms and for a polynomial multiplication modulo $1 + x^{12}$ (see p. 122).

	1	2	3	4	5	6	7	8	9	10	11
1	-	-	-	-	-	-	12	-	-	-	-
2	-	-	-	-	-	60	-	-	-	6	-
3	-	-	-	-	180	-	-	-	40	-	-
4	-	-	-	255	-	-	-	240	-	-	-
5	-	-	180	-	-	-	600	-	-	-	12
6	-	60	-	-	-	804	-	-	-	60	-
7	12	-	-	-	600	-	-	-	180	-	-
8	-	-	-	240	-	-	-	255	-	-	-
9	-	-	40	-	-	-	180	-	-	-	-
10	-	6	-	-	-	60	-	-	-	-	-
11	-	-	-	-	12	-	-	-	-	-	-

Table 7.4: Hamming weight distribution table of polynomial multiplication by $e(x)$ modulo $1 + x^{12}$.

7.5 The bit permutations μ and π

μ is a bit permutation with $\mu^{\ 1} = \mu$ that respects the grouping of the triplets and 12-tuples, but inverts the order of the components within them. The simplest example of such a bit permutation is μ_1, that simply inverts the order of the components of the vector. Hence, if $b = \mu_1(a)$ we have

$$b_i = a_{n_b \ \ 1 \ \ i} \text{ for } 0 \leq i < n_b \ . \tag{7.33}$$

This bit permutation requires the handling of individual bits and is not very well suited for software implementations. This is not the case for the bit permutation μ_h that inverts the order of h-bit subblocks. If $b = \mu_h(a)$ we have

$$b_{i+jh} = a_{i+(11 \ \ j)h} \ , \tag{7.34}$$

for $0 \leq i < h$ and $0 \leq j < 12$.

The bit permutations π_1 and π_2 treat the vector in 3 or 12 subblocks. The bits of each subblock are cyclically shifted by a specified number of positions. The bit permutations can be specified by an array that contains these rotation constants. The effect of a bit permutation $b = \pi(a)$ specified by the 3-tuple (p_0, p_1, p_2) is described by

$$\left.\begin{array}{lcl} b_i & = & a_{i \ \ p_0 \bmod k} \\ b_{i+k} & = & a_{(i \ \ p_1 \bmod k)+k} \\ b_{i+2k} & = & a_{(i \ \ p_2 \bmod k)+2k} \end{array}\right\} \text{ for } 0 \leq i < k \ .$$

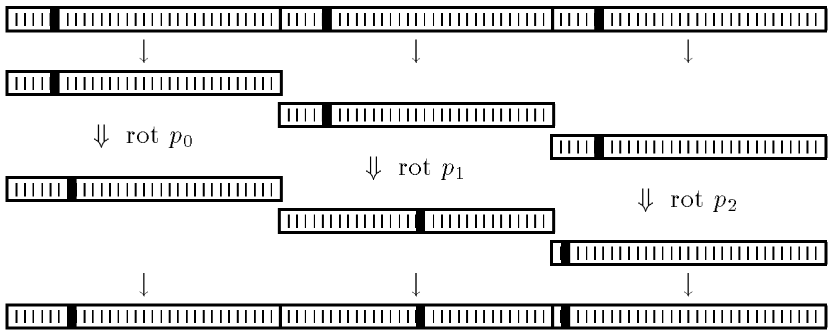

Figure 7.4: The arrangements of bits in the π bit permutation with 3 blocks specified by $(p_0, p_1, p_2) = (2, 11, 28)$. The effect is shown on the three bits of a triplet.

Figure 7.4 illustrates the arrangement of the bits in such a bit permutation. In software, the transformations π can be implemented using bitwise shift operations.

The choice of one of the π bit permutations and μ determines the other π bit permutation through (7.9). For μ_1 we have

$$\pi_1 : (p_0, p_1, \ldots) \Leftrightarrow \pi_2 : (\ldots, p_1, p_0) \ . \tag{7.35}$$

For μ_h it can easily be checked that the bit permutations must necessarily split the vector into 12 subblocks. We have

$$\pi_1 : (p_0, p_1, \ldots, p_{11}) \Leftrightarrow \pi_2 : (\quad p_{11}, \ldots, \quad p_1, \quad p_0) \ . \tag{7.36}$$

7.5.1 Propagation and correlation properties

Since bit permutations are linear, the propagation of differences through π_1 and π_2 is governed by

$$b' = \pi_i(a') \ . \tag{7.37}$$

For the specification of linear combinations of input bits w in terms of output bits u, we write the bit permutation in a matrix M_π. We have

$$w = M_\pi^{\mathrm{t}} u \ . \tag{7.38}$$

Using the fact that the permutation of components is an orthogonal transformation and therefore $M_\pi^{\quad 1} = M_\pi^{\mathrm{t}}$, we have $u = M_\pi w$, or equivalently

$$u = \pi_i(w) \ . \tag{7.39}$$

As in the case of γ and θ it can be seen that the difference propagation and the input-output selection correlation are governed by the same equation. This is an inherent property of bit permutations.

7.6 Propagation analysis

In this section we describe the behavior of differential and linear trails in the proposed block cipher structure.

The block cipher can be described as the repeated application of alternating nonlinear transformations γ and linear (in fact affine) transformations $\lambda[\kappa_j] = \pi_1 \circ \theta \circ \sigma[\kappa_j] \circ \pi_2$. For the nonlinear step the correlation and difference propagation properties are described by the compatibility conditions and the triplet weight. The propagation of differences through λ is governed by

$$b' = \lambda[0](a') \ . \tag{7.40}$$

A linear combination of input bits specified by w is correlated to a linear combination of output bits specified by u given by

$$u = \lambda[0](w) \ , \tag{7.41}$$

with correlation $(\ 1)^{u^t \pi_1(\theta(\kappa_j))}$. In the following we will omit the $[0]$ in $\lambda[0](a)$.

In the context of propagation analysis we consider a round to be $\gamma \circ \lambda[\kappa_j]$. A differential *step* consists of a couple of difference vectors (a', b') with $\lambda(a')$ γ-compatible with b'. Its restriction weight is $2\mathrm{w_t}(b')$. A linear *step* consists of a couple of selection vectors (w, u) with $\lambda(w)$ compatible with u. Its correlation weight is $\mathrm{w_t}(u)$. Hence, a couple (a, b) with $\lambda(a)$ γ-compatible with b can be interpreted both as a differential and as a linear step and is called a *propagation* step. These steps can be chained to form *propagation* trails.

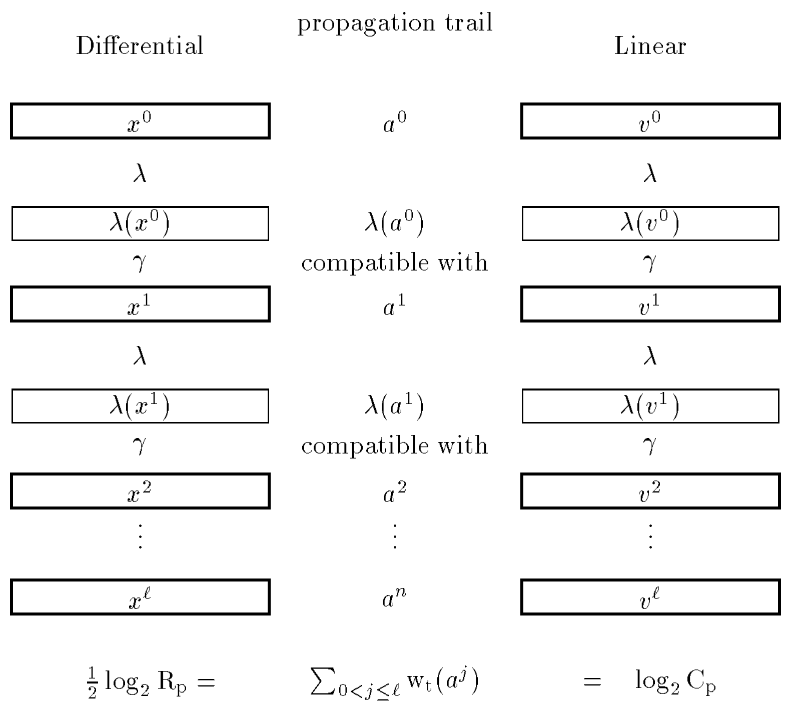

Figure 7.5: The differential and linear interpretations of propagation trails for the self-reciprocal block cipher structure.

An ℓ-round propagation trail Ω is specified by an $\ell + 1$-tuple

$$(\omega^0, \omega^1, \ldots, \omega^\ell) \ ,$$

with $\lambda(\omega^{i \quad 1})$ compatible with ω^i for $0 < i \leq \ell$. Its triplet weight is given by

$$\mathrm{w_t}(\Omega) = \sum_{0<i\leq\ell} \mathrm{w_t}(\omega^i) \ . \tag{7.42}$$

This propagation trail represents both a differential trail with restriction weight $2\mathrm{w_t}(\Omega)$ and a linear trail with correlation weight $\mathrm{w_t}(\Omega)$. Figure 7.5 shows the two interpretations of a propagation trail.

Once the block size n_b and the permutation μ have been fixed, all that remains is the specification of π_1. The shift constants must be chosen to eliminate the occurrence of propagation trails with low triplet weight, addressing the resistance against DC and LC in a single effort. The selection of the array of rotation constants involves finding and comparing the critical propagation trails for a small number of rounds by computer.

After the specification of the rotation constants, it has to be verified that the triplet weight of the propagation trails appropriately reflects the resistance against LC and DC. The most important effect that has to be investigated is the potential systematic clustering of low-weight trails.

7.7 Symmetry considerations

Even if a block cipher is resistant against LC and DC, symmetry in its structure can be the cause of serious weaknesses. The best known example of a block cipher with such weaknesses is DES. The first symmetry-based weakness is the existence of weak and semi-weak keys for DES [30]. For the four weak keys, encryption is an involution. For the six pairs of semi-weak keys, encryption with one key of a pair is the same as decryption with the other key of the pair. The second symmetry-based weakness of DES is the complementation property [6, 48]. This property can be exploited to reduce exhaustive key search of DES by a factor of 2. More recent examples can be found in [7], where the regularity in key schedules is used to construct efficient chosen-key attacks and to speed up exhaustive key search.

Many undesirable symmetry properties are special cases of one of the two following properties:

- There are affine mappings $\lambda_{\mathrm{k}}, \lambda_{\mathrm{p}}$ and λ_{c}, such that for some keys $\lambda_{\mathrm{c}} \circ \mathrm{B}[\lambda_{\mathrm{k}}(\kappa)] \circ \lambda_{\mathrm{p}}$ is equal to $\mathrm{B}[\kappa]$ or $\mathrm{B}[\kappa]^{\ \ 1}$;
- There are keys for which the last $r \quad p$ rounds of the cipher (or its inverse) under one key perform the same transformation as the first $r \quad p$ rounds of the cipher (or its inverse) under another key, with p small.

The round constants c^j must be chosen in such a way that all symmetry-related weaknesses are eliminated. This choice is not affected in any way by propagation trail considerations. The round constants are derived from the state q of a linear feedback shift register with length 8. In polynomial representation we have

$$q^j(x) = (1 + x + x^3)x^j \bmod (1 + x^4 + x^8) \ . \tag{7.43}$$

The order of the feedback polynomial is 12, hence, $x^{12} = 1 \bmod (1+x^4+x^8)$. The calculation of the round constants c^j in polynomial representation is

$$\mathrm{c}^j(x) = (x^{2h} + x^{3h} + x^{8h} + x^{9h})q^j(x) \ , \tag{7.44}$$

with $12h = n_\mathrm{b}$.

The round constants are chosen in such a way that the difference between the round constants of any pair of subsequent encryption or decryption rounds is different. The decryption round constants depend on the block length and on the bit permutation μ.

7.8 3-Way

7.8.1 Specification

3-Way is a block cipher with the self-reciprocal structure. It is designed to be hermetic and K-secure with respect to sound initialization mappings. It is specified by

1. $n_\mathrm{b} = 96$,
2. $\mu = \mu_1$,
3. $\pi_1 : (10, 0, \quad 1)$ and $\pi_2 : (\quad 1, 0, 10)$,
4. encryption: 11 rounds and a single output transformation,
5. decryption: 11 rounds and a single output transformation, preceded and followed by μ.

μ_1 has been chosen to allow the π permutations to act on 32-bit words. The rotation constants $(10, 0, \quad 1)$ have been selected in the following way. 0 and 1 have been fixed in advance because of their economy. 10 was selected from the candidate constants $\{3, 5, 6, 7, 9, 10, 11, 13, 14, 15\}$ as realizing the best propagation properties in the short term. It would of course be better to select the rotation constant with the best propagation properties in the long term, but this turns out to be computationally infeasible.

7.8.2 Implementation aspects

The number of rounds 11 is motivated by its convenience in a cryptographic finite state machine implementation. The encryption of a single block takes 12 clock cycles: 11 state updating iterations and 1 simultaneous plaintext load and ciphertext read operation. This results in an encryption (and decryption) rate of 8 bits per clock cycle. The total gate delay of the cryptographic finite state machine can be made as small as that of 4 exors, 1 nand and 1 mux (multiplexor), allowing clock speeds of over 100 MHz and encryption rates of over 800 Mbit/s, even with conventional technology. The small number of basic operations also allows for extremely compact hardware implementations with a small 8-bit processor with instructions bitwise addition, AND and shift, some program memory (ROM) and some data memory (RAM).

In software, the steps γ and θ can be efficiently programmed using bitwise EXOR, OR, complementation and shifting. A straightforward C implementation allows an encryption speed of over 2 Mbit/s on a 66 MHz 80486 processor. We expect that optimization and the use of coding in assembler language allow a speedup by at least a factor of 5.

7.8.3 Decryption

For the inverse round constants in 3-WAY we have $\mathrm{c}'^{j} = \mu_1(\theta(\mathrm{c}^{11\;\;j}))$. It can be seen that

$$\mathrm{c}'^{j}(x) = (x^{2h} + x^{3h} + x^{8h} + x^{9h})q'^{j}(x) \ , \tag{7.45}$$

with $q'^{j}(x)$ given by

$$q'^{j}(x) = (1 + x^4 + x^5 + x^7)x^j \bmod (1 + x^4 + x^8) \ . \tag{7.46}$$

In a cryptographic finite state machine the encryption and decryption round constants can be generated and applied with the same circuitry. The only difference is the initial value q^0 of the 8-bit linear feedback shift register.

7.8.4 Propagation analysis

Propagation analysis mainly consists of the search for propagation trails with low weight. For this purpose we have written and ran programs that scan the space of propagation trails in a recursive pruned tree search.

Table 7.5 is the (partial) triplet weight distribution table of $\lambda = \pi_1 \circ \theta \circ \pi_2$ for 3-WAY. The element in row i and column j denotes the number of couples $(a, \lambda(a))$ with $\mathrm{w_t}(a) = i$ and $\mathrm{w_t}(\lambda(a)) = j$. This table illustrates the high quality of the short-term propagation properties of the 3-WAY round transformation. It can be seen that for any couple $(a, \lambda(a))$ the sum of their triplet weight is at least 8. It follows that there are no 2-round propagation trails with triplet weight below 8 and consequently that the minimum triplet weight for propagation trails of even length is 4 per round. The triplet weight distribution table inherits this property from the Hamming weight distribution table of θ. This is a consequence of the application of the π bit permutations before and after θ. These simple bit permutations contribute to the single-round diffusion by spreading the bits in a single triplet over several other triplets.

	1	2	3	4	5	6	7
1	-	-	-	-	-	-	96
2	-	-	-	-	-	480	-
3	-	-	-	-	1440	-	-
4	-	-	-	2040	-	7	168
5	-	-	1440	-	25	313	12480
6	-	480	-	7	313	?	?
7	96	-	-	168	12480	?	?
8	-	-	55	5335	71138	?	?
9	-	19	1122	28012	265865	?	?
10	-	195	6381	90042	431964	?	?
11	39	836	18775	119868	457174	?	?
12	25	1883	20751	113010	776241	?	?
13	32	2017	17408	159098	2682584	?	?
14	13	1677	21418	469917	6262878	?	?

Table 7.5: Partial triplet weight distribution table of λ for 3-WAY.

The number of vectors with a triplet weight $\mathrm{w_t}$ is given by

$$7^{\mathrm{w_t}}\binom{n_\mathrm{b}/3}{\mathrm{w_t}} \ . \qquad (7.47)$$

For $n_\mathrm{b} = 96$ and $\mathrm{w_t} = 6$ we have 3.4×10^9 vectors. This turned out to be too large for our exhaustive program to end within a reasonable time span, hence the question marks in Table 7.5. The values that are actually listed in column 6 and 7 are known because the partial triplet weight distribution table is symmetrical. This is a consequence of the fact that λ is an orthogonal linear transformation in $\mathbb{Z}_2^{n_\mathrm{b}}$.

The bit permutations play an important role in the multiple-round propagation properties in preventing the clustering of propagation trails. This can be illustrated by considering the hypothetical case of all three rotation constants in π_1 being 0. In that case the bits of the 12-tuples are not mixed and the propagation trails are restricted to stay within the 12-tuples, inevitably giving rise to clustering. Moreover, the bit permutations π prevent the iterative chaining of propagation steps $(w_{j\ 1}, w_j)$ with $\mathrm{w_t}(w_{j\ 1}) + \mathrm{w_t}(w_j) = 8$, by destroying the alignment required for these low triplet weights.

If 3-WAY has no 9-round propagation trails with a triplet weight below 48, there are no 9-round differential trails with prop ratio above $2^{\ 96}$ and no linear trails with correlation above $2\ ^{48}$. Both are too insignificant to be exploited in an attack.

From observing Table 7.5 it can easily be seen that there are no (even-length) differential trails with a prop ratio below $2\ ^{8}$ per round, neither (even-length) linear trails with a correlation contribution below $2\ ^{4}$ per round. This is more than a factor 2 better than the round function of DES, with its iterative differential trails with a prop ratio of $2\ ^{3.6}$ per round [5] and its 14-round linear trail with a correlation contribution of $2\ ^{20.2}$ or $2\ ^{1.4}$ per round [71].

By listing the critical propagation weights of 1,2,... rounds, a *propagation weight profile* can be specified. We have been able to determine this profile for up to 5 3-Way rounds: $(1, 8, 11, 16, 22)$. For 6 rounds no propagation trails were found with a propagation weight below 36. The relatively low weights of the critical propagation trails for a small number of rounds are due to the inability of π to destroy certain occurrences of local alignment. As the number of rounds grows, these alignment conditions become increasingly restrictive and for 6 rounds we were already unable to exploit it. For this reason we believe the triplet weight of the critical 9-round propagation trails to be much higher than 48.

We once more indicate that the function of our propagation investigations is to support the choice of the rotation constants and the verification that 11 rounds are sufficient, not to give any proof of security. The security will eventually be based on the inability of cryptologists to find exploitable weaknesses.

Attacks can be devised where part of the key is known. The knowledge of some key material can be exploited to fix part of a differential trail. The large diffusion ensures that, already after two rounds, the unknown part of the key is diffused over the complete encryption state. Squeezing off more than a single round requires the knowledge of too many key bits to be a threat to K-security.

7.9 BaseKing

7.9.1 Specification

BaseKing is a block cipher with the self-reciprocal structure and is designed to be hermetic and K-secure with respect to sound initialization mappings. It is specified by

1. $n_{\mathrm{b}} = 192$,
2. $\mu = \mu_{16}$,

3. $\pi_1 : (0, 8, 1, 15, 5, 10, 7, 6, 13, 14, 2, 3)$ and
   $\pi_2 : (13, 14, 2, 3, 10, 9, 6, 11, 1, 15, 8, 0)$,
4. encryption: 11 rounds, a single output transformation and μ,
5. decryption: 11 rounds, a single output transformation and μ.

The block and key length impose that the triplet weight of 9-round propagation trails must exceed 96. From some early experiments it was concluded that this could not be realized by π permutations with only three rotation constants. Therefore, the π permutations act on 12 16-bit words, allowing the adoption of μ_{16} which is easily implementable in software. The rotation constants have been determined after a coarse analysis of their most elementary interaction (combined addition) and may be susceptible of improvement. The hardware and software implementation aspects of BaseKing are almost identical to those of 3-Way. In most implementations, encryption with BaseKing requires the same effort per bit as with 3-Way. In hardware, the doubling of the block length with respect to 3-Way allows a multiplication of the encryption speed by a factor close to two.

The most important advantage of BaseKing over 3-Way is its large block length. Because of this, BaseKing can be used as the main building block in the elegant and efficient block cipher based cryptographic hash function and checksum schemes described and/or proposed by Bart Preneel in [84] and standardized by ISO in [54, 53]. In this way the different modes of BaseKing cover the complete spectrum of single-key encryption and hashing.

7.9.2 Decryption

For the inverse round constants in BaseKing we have $\mathrm{c}'^j = \mu_{16}(\theta(\mathrm{c}^{11\ \ j}))$. It can be seen that

$$\mathrm{c}'^j(x) = (x^{2h} + x^{3h} + x^{8h} + x^{9h}) q'^j(x) \ , \tag{7.48}$$

with $q'_j(x)$ given by

$$q'_j(x) = (1 + x^2 + x^3 + x^7) x^{\ \ j} \bmod (1 + x^4 + x^8) \ . \tag{7.49}$$

In a cryptographic finite state machine the encryption and decryption round constants can be generated with two simple linear feedback shift registers and be applied using the same connection circuitry.

7.9.3 Propagation analysis

Table 7.6 is the (partial) reduced triplet weight distribution table of λ for BaseKing. Since λ has a rotational symmetry over the 16-bit subblocks, all entries of Table 7.6 would be a multiple of 16. This common factor has been removed.

Because of its key and block length of 192 bits, BaseKing is in principle a much more ambitious design as 3-Way. The required triplet weight of a 9-round propagation trail is 96, or almost 11 per round. This is more than a factor 7 better than DES, i.e., a single application of the round function of BaseKing must be as effective as *seven* rounds of DES. However, in experiments with propagation trails for only a few rounds, the 12-component permutations π appeared to be very powerful in their task of disrupting locally propagating structures. These observations have given us a high degree of confidence in the assumption that the triplet weight of the critical 9-round propagation trails is significantly larger than 96.

7.9.4 Alternative software implementations

Both 3-Way and BaseKing require the handling of words that have a length smaller than 32, the common processor word length in modern computers. However, by a simple rearranging of the blocks both can be implemented using only 32-bit instructions (or any larger power of 2). We illustrate this for the case of BaseKing.

For BaseKing the operations on 16-bit words can be turned into operations on 32-bit words by encrypting the messages in blocks of 384 bits. In this alternative scheme γ and θ are substituted by their $n_b = 384$ versions and μ_{16} is substituted by μ_{32}. The subblock size and the rotation constants of the π bit permutations are doubled. The key and the round constants are doubled in length by doubling every bit. The resulting cipher can be considered to be the parallel application of BaseKing to the 192 bits on the odd positions and the 192 bits on the even positions.

7.10 Filtered counter stream encryption

The ease of changing the cipher key for 3-Way and BaseKing allows the specification of a very simple filtered counter stream encryption scheme.

The state updating is governed by a linear feedback shift register of length n_b. The feedback polynomial must be primitive and needs to have a Hamming weight of only 3. The initial counter state and cipher key both depend on the parameter Q and the key K. The dependence of the cipher key on the parameter Q prevents the choice of two parameter values

	1	2	3	4	5	6	7
1	-	-	-	-	-	-	12
2	-	-	-	-	-	60	-
3	-	-	-	-	180	-	-
4	-	-	-	255	-	-	-
5	-	-	180	-	-	-	624
6	-	60	-	-	-	?	?
7	12	-	-	-	624	?	?
8	-	-	-	255	1282	?	?
9	-	-	44	509	17547	?	?
10	-	6	89	6087	88972	?	?
11	-	4	1254	27588	136398	?	?
12	-	150	5498	36569	4284	?	?
13	3	618	6060	689	35274	?	?
14	9	562	59	5474	167937	?	?
15	-	1	491	25142	690142	?	?
16	-	13	2064	103494	2228150	?	?
17	-	50	8886	324518	4206530	?	?
18	-	295	27539	563180	4083993	?	?
19	1	775	42592	479688	2533248	?	?
20	1	942	31118	233210	4435549	?	?
21	2	357	11037	346777	12752974	?	?

Table 7.6: Partial reduced (divided by 16) triplet weight distribution table of the linear transformation λ for BaseKing.

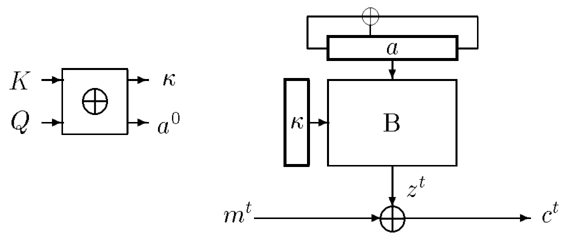

Figure 7.6: Proposed filtered counter encryption scheme using 3-Way or BaseKing as a component.

that give rise to partly overlapping sequences. All n_b bits can be used for encryption, hence, $n_s = n_b$. This scheme is illustrated in Fig. 7.6 and described by

$$\begin{aligned}
\kappa &= K + Q \ , \\
a^0 &= \tau_1(K) + Q \ , \\
a^{t+1}(x) &= x \cdot a^t(x) \bmod m(x) \ , \\
z^t &= \mathrm{B}[\kappa](a^t) \ .
\end{aligned}$$

7.11 Conclusions

In this chapter we have presented a new self-reciprocal structure for block ciphers. This structure is quite general since it gives the designer a high degree of freedom in the choice of the specific transformations.

By adopting some specific step transformations, we have shown that it is possible to build block ciphers that have the unique property that differential and linear trails are governed by exactly the same equations.

We have shown that applying the wide trail strategy yields efficient and portable round transformations that are superior to the DES round function with respect to LC and DC.

The specific designs 3-Way and BaseKing have been investigated with respect to their propagation behavior. We have not been able to determine the critical propagation trails for more than a few rounds. We think that it is an interesting opportunity for further research to improve the efficiency of the exhaustive propagation trail search procedures.