The Cipher SHARK

Vincent Rijmen* Joan Daemen
Bart Preneel** Antoon Bosselaers
Erik De Win*

Katholieke Universiteit Leuven, ESAT-COSIC
K. Mercierlaan 94, B-3001 Heverlee, Belgium

{vincent.rijmen,joan.daemen,bart.preneel,antoon.bosselaers,erik.dewin}
@esat.kuleuven.ac.be

Abstract. We present the new block cipher SHARK. This cipher combines highly non-linear substitution boxes and maximum distance separable error correcting codes (MDS-codes) to guarantee a good diffusion. The cipher is resistant against differential and linear cryptanalysis after a small number of rounds. The structure of SHARK is such that a fast software implementation is possible, both for the encryption and the decryption. Our C-implementation of SHARK runs more than four times faster than SAFER and IDEA on a 64-bit architecture.

1 Introduction

The best known and most used block cipher today is the DES [FIPS46]. The operation of the DES can be described in the following way: the message input X is divided into two halves X_1 and X_2. These halves are then processed in 16 rounds. The odd-numbered rounds perform the following transformation:

$$Y_1 = X_1 \oplus F(K, X_2)$$
$$Y_2 = X_2 ,$$

while in even-numbered rounds:

$$Y_1 = X_1$$
$$Y_2 = X_2 \oplus F(K, X_1) .$$

After the last round, both halves are swapped. This structure is called the "Feistel structure" [F73]. Many proposed alternatives for the DES use the same structure, which has the nice feature that it is invertible for all choices of the F-function. An important weakness however is that each round transformation always keeps one half of the block constant. This fact is used in many attacks,

* N.F.W.O research assistant, sponsored by the National Fund for Scientific Research (Belgium).
** N.F.W.O. postdoctoral researcher, sponsored by the National Fund for Scientific Research (Belgium).

including differential and linear cryptanalysis. This structure is generalized in [SB95] to a "generalized unbalanced Feistel network," where the fraction of the round input that always equals the output is made variable.

The round transformation of SHARK, which we describe in this paper, is more uniform:

$$Y = F(K, X),$$

where $F(K, X)$ is an invertible function. This structure is similar to a substitution-permutation network [FNS75] and is also used in MMB [DGV93], SAFER [M94, M95], and 3-WAY [DGV94b]. Each round transforms the whole round input. The combination of strong diffusion and uniform non-linearity allows the reduction of the number of rounds, but, compared with a Feistel cipher, the amount of work per round increases.

An important design criterion for an encryption algorithm is its performance. Designers search for round functions that allow to reduce the number of rounds in order to get maximal performance. CAST [AT95] and SAFER [M94] can be seen as attempts in this direction. The lurking danger is that a small number of rounds makes a whole range of new attacks possible, e.g., the differential-linear attack on eight rounds of the DES [LH94], truncated differentials in SAFER [K95, K96], and imbalance of the round function in CAST [RP95]. We believe SHARK is resistant against these attacks.

In Sect. 2 we explain our design strategy and select components for SHARK. Section 3 gives some cryptanalytic benchmarks. In Sect. 4 we make some remarks about the implementation, and Sect. 5 discusses further work.

2 Design Strategy

In our design strategy the round transformation is composed of three distinct building blocks:

- a non-linear layer (e.g., substitution boxes);
- a diffusion layer;
- a key scheduling to produce round keys from the key.

The design strategy assigns to each of these components a function. The components are selected to fulfill this function in an optimal way.

By considering each building block separately, we get a robust cipher. We select a diffusion layer with uniform and good diffusion properties. The non-linear layer has uniform nonlinear properties, such that when measuring the resistance of the cipher against cryptanalysis we don't have to take the details of the interaction between the non-linear and the diffusion layer into account. If, for example, the S-boxes are replaced by other S-boxes, with equivalent non-linearity properties, the resistance of the cipher remains constant. This strategy is a variant of the *wide trail strategy* [D95]. Since each building block is selected and examined separately, it is not attempted to compensate weaknesses of the non-linear layer by additional properties of the linear layer.

In the remainder of this section we define a clear criterion for each of the building blocks and make design decisions. The S-boxes are m–bit permutations. The number of parallel S-boxes is denoted with n, and the number of rounds with R. Fig. 1 shows the general structure of SHARK. Note that this figure represents the conceptual structure, which differs slightly from the actual implementation.

The figure shows that SHARK consist of R rounds with a key addition, non-linear substitution, and a diffusion layer. This is followed by an extra key addition and an extra diffusion layer, which is the inverse of the round diffusion layers. The purpose of the extra key addition is to prohibit an attacker from peeling off the last round. The extra diffusion layer is needed for an easy implementation of the decryption. This will be explained in Sect. 4.

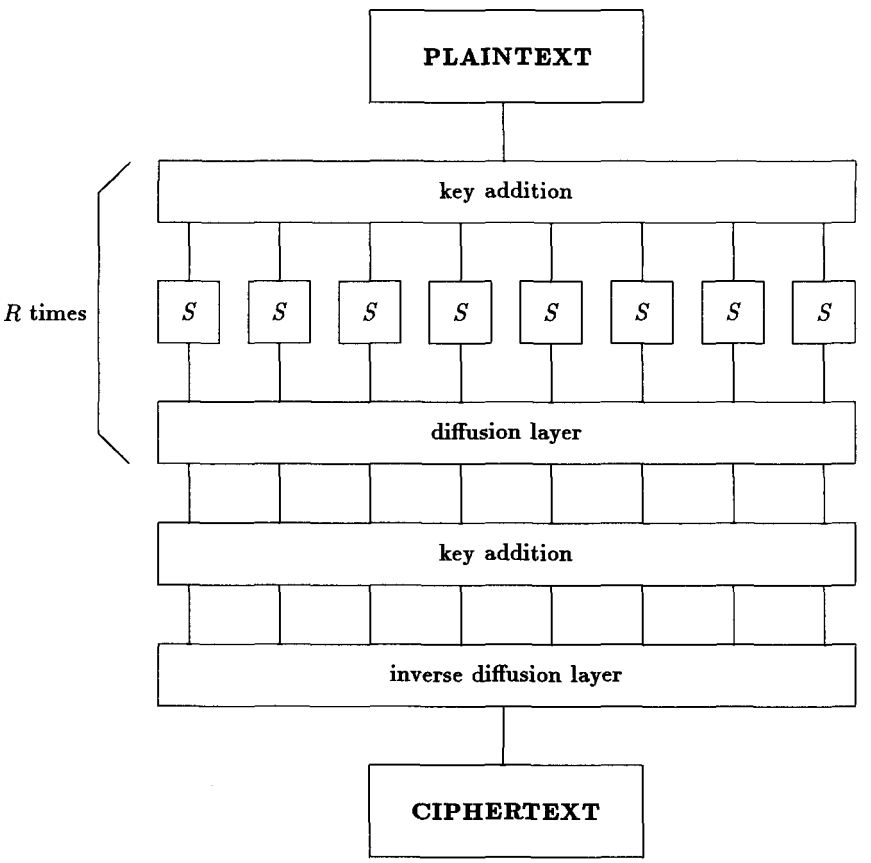

Fig. 1. Structure of SHARK.

2.1 Diffusion Layer

For the design of the diffusion layer, we consider the m-bit outputs of the S-boxes as elements of $GF(2^m)$. The diffusion layer takes n m-bit values as input, and gives n m-bit outputs.

The purpose of the diffusion layer is to provide an avalanche effect, both in the context of differences and linear approximations. In the linear context this means that there should be no correlations between linear combinations of a small set of (m-bit) inputs and linear combinations of a small set of (m-bit) outputs. In the differential context this means that small input changes should cause large output changes, and conversely, to produce a small output change, a large input change should be necessary (where we consider again m-bit values as inputs and outputs). For an invertible linear mapping θ, this effect can be quantified by its *branch number* $\mathcal{B}$ [D95].

Denote by $w_h(a)$ the Hamming weight of a, i.e., the number of non–zero components of a. These components can be bits, as in [D95], or elements from $GF(2^m)$ as here. Then

$$\mathcal{B}(\theta) = \min_{a \neq 0}(w_h(a) + w_h(\theta(a))) .$$

$\mathcal{B}$ gives a measure for the worst case diffusion: it is a lower bound for the number of active S-boxes in two consecutive rounds of a linear trail or a differential characteristic (we will define the term *active S-boxes* in Sect. 3). Since a cryptanalyst will always exploit the worst case, this is a good measure for the diffusion property. Note that $w_h(a) \leq n$, for every choice of θ; if $w_h(a) = 1$, this implies that $\mathcal{B} \leq n + 1$. We call an invertible linear mapping γ for which $\mathcal{B} = n + 1$ *optimal.*

The framework of linear codes over the field $GF(2^m)$ gives us an elegant way to construct a diffusion layer with optimal branch number. A linear code C of length n, dimension k, and with minimum distance d between the codewords is denoted as an (n, k, d)–code. An (n, k, d)-code is a k-dimensional subspace of the vector space of n-tuples over $GF(2^m)$. The Hamming distance between two codewords is equal to the number of elements in which they differ.

A linear code can be represented by a *generation matrix* $G_{k \times n}$. This matrix has dimensions $k \times n$, and is always of full rank. C is formed by the subspace of dimension k that is spanned by the rows of G.

$$C = \{G \cdot X | X \in GF(2^m)^k\} .$$

The generation matrix of a code is not unique. If G is a generation matrix of C, then every matrix $G_1 = T_{k \times k} \cdot G$, where $T_{k \times k}$ is of full rank, is also a generation matrix of C. The matrix $G_e = T \cdot G = [I_{k \times k} B_{k \times (n-k)}]$ is called the *echelon form* of G.

Codes with $d = n - k + 1$ are called *maximal distance separable* codes. A well–known example of MDS–codes are the Reed–Solomon codes. RS–codes can have lengths up to $q + 1$, where q is the number of elements in the finite field in which the codes are defined (here $q = 2^m$) [MS77, pp. 294–306].

Proposition 1 *Let C be a $(2n, n, n+1)$-code over the Galois field $GF(2^m)$. Let G_e be the generator matrix of C in echelon form:*

$$G_e = [I_{n\times n} B_{n\times n}] .$$

Then C defines an optimal invertible linear mapping γ:

$$\gamma : GF(2^m)^n \rightarrow GF(2^m)^n : X \mapsto Y = B \cdot X .$$

Proof: First we show that $\mathcal{B} = n + 1$. The definition of $\mathcal{B}$ gives:

$$\begin{aligned} \mathcal{B}(\gamma) &= \min_{X\neq 0}(w_h(X) + w_h(\gamma(X))) \\ &= \min_{X\neq 0}(w_h(X, \gamma(X)). \end{aligned}$$

It follows from the definition of γ that all the $2n$-tuples $(X, \gamma(X))$ are codewords of the code C. The minimum of the Hamming weights of the non-zero codewords is by definition equal to $d = n + 1$.
We prove by contradiction that γ is invertible. Suppose γ is not invertible. This means there are two vectors a, b such that $a \neq b$ and $\gamma(a) = \gamma(b)$. Since γ is linear, $\gamma(a - b) = \gamma(a) - \gamma(b) = 0$. Then

$$\mathcal{B} = \min_{c\neq 0}(w_h(c) + w_h(\gamma(c))) \leq w_h(a - b) + w_h(\gamma(a - b)) \leq n + 0 < n + 1 .$$

This is in contradiction with $\mathcal{B} = n + 1$. ∎

The branch number of a diffusion layer is very important. In [K96], Knudsen breaks five rounds of SAFER, making use of the low branch number of its diffusion layer (the Pseudo Hadamard Transform). The low diffusion of the DES is used by Matsui [M93] to construct a linear approximation of the cipher with high correlation.

2.2 Substitution Boxes

Non-linear S-boxes provide resistance against linear and differential cryptanalysis. A large amount of criteria, which are sometimes conflicting, have been published (see, for example, [C94, DT91, KMI91, N91, N94]).

The exor table E of a mapping γ is defined as follows [BS90]:

$$E_{ij} = \#\{x | \gamma(x) \oplus \gamma(i \oplus x) = j\} .$$

High entries in the exor table can lead to differential characteristics with a high probability making the cipher susceptible to a differential attack.

In [N94] Nyberg proposes several classes of non-linear substitution boxes. For SHARK, we choose an S-box that is based on the mapping $F(x) = x^{-1}$ over $GF(2^m)$. This class of S-boxes has the following properties (when m is even):

- Differentially 4-uniform. This means that the highest value in the exor table equals 4. In fact, every row of the exor table contains exactly one 4, the other possible values are 2 and 0.

- Minimal distance to an affine function is $2^{m/2}$.
- The non-linear order of every linear combination of output bits equals $m-1$.

This is the only class of mappings of [N94] that can be used for boxes with dimensions $2^m \times 2^m$, with m even. In order to remove the fixed points $0 \rightarrow 0$, and $1 \rightarrow 1$, we apply an invertible affine transformation to the output bits.

The disadvantage of these boxes is that they have a simple description in $GF(2^m)$, which is also the field in which the diffusion layer is linear. This may create uneasy feelings, but we are not aware of any vulnerability caused by this property. For the time being we challenge cryptanalysts to demonstrate any vulnerability caused by this property. Should such a vulnerability exist, one can always replace the S-boxes by S-boxes with similar properties, that are not algebraic over $GF(2^m)$.

2.3 Key Scheduling

The key scheduling expands the key K to the round keys K_i. A good key scheduling produces round keys with maximal entropy.

First we present two alternative ways to introduce the round key in the round function: one is a simple exor of the round key with the input, the second is a key controlled affine transform. Then we explain how these subkeys are generated from the key.

Exor The nm input bits of the round are exored with nm key bits. This method is fast and uniform: there are no keys that are stronger or weaker in the sense that the diffusion and the non–linear layer have the same properties for all keys. This notion of weak keys is the same as in [DGV94]. A disadvantage of the simple scheme is that the entropy of the round key is at most nm.

Affine Transformation Let κ_i be a key dependent invertible $(n \times n)$–matrix over $GF(2^m)$. Define the key operation as:

$$Y = \kappa_i \cdot X \oplus K_i \,.$$

This operation is still linear and thus it introduces no weak keys. Each round now introduces more key material, raising the entropy of the round keys to $\mathcal{O}(mn^2)$. The computational overhead of this operation is very large. We can restrict κ_i to a certain subspace, for instance let κ_i be a diagonal matrix. The entropy of the round keys then becomes close to $2mn$. In Sect. 4 it will be explained how to implement this variant in an efficient way.

Subkey generation Many attacks on iterated ciphers first recover (part of) a round key. This knowledge is subsequently used to recover other round keys and/or the key. To make these attacks less efficient, one can generate the round keys by hashing the key with a preimage resistant function, like in Blowfish [S94] or CAST [AT95].

In SHARK, the round key generation is as follows. The $R+1$ mn-bit values K_i are initialized with the first $R+1$ entries of the substitution table T_0, which is defined in Sect. 4. The matrices κ_i are initialized to the unit matrix I. The user selected key is then concatenated with itself until it has a length of $2(R+1)mn$ bits. This is used as input of SHARK in 64-bit CFB mode [ISO10116]. The $2(R+1)mn$ output bits are used as the actual round keys for the encryption of the message: the first $(R+1)mn$ bits are the values K_i, the next bits are interpreted as $(R+1)n$ elements of $GF(2^m)$ and form the diagonal elements of the κ_i. If one of these elements is zero, it is discarded. Subsequently all the following values are shifted down one place and an extra encryption of the all-zero string is added at the end.

While this mechanism for subkey generation in principle makes it possible to use a key of $2(R+1)mn$ bits, we suggest that the key length should not exceed 128 bits.

3 Resistance Against Differential and Linear Cryptanalysis

In a differential characteristic, S-boxes that have a non-zero input exor are called *active S-boxes*; these S-boxes produce the required output exor with a certain probability. The S-boxes of SHARK are chosen such that this probability is at most 2^{2-m}. Inactive S-boxes have a zero input exor and consequently they have always a zero output exor. The diffusion layer ensures that two consecutive rounds have in total at least $\mathcal{B} = n+1$ active S-boxes.

In a linear attack, the cryptanalyst tries to find correlations between linear combinations of input bits and linear combinations of output bits. S-boxes from which some input bits and some output bits are involved in the linear combinations, are called active S-boxes. S-boxes from which no input or output bits are involved in the linear combinations, are called inactive S-boxes. Assuming that the inputs to different S-boxes are independent, we can calculate the correlation by multiplying the correlations for the active S-boxes. The correlations in SHARK's S-boxes are at most $2^{1-m/2}$, which means that every active S-box increases the amount of needed texts by a factor of at least 2^{m-2} [M93].

The dimension of the S-boxes m, and the number of parallel S-boxes n have both been chosen equal to eight. This means that the resulting cipher works on message blocks of 64 bits. We propose the cipher with six rounds. For applications that require only 40 bits security, four rounds may suffice.

Table 1 gives some numerical values for the probabilities of the best possible differential characteristics and squared correlations for the best linear approximations as a function of the number of rounds R, compared with the values for the DES. Not that since the DES is a Feistel cipher, the number of rounds has to be doubled to obtain a fair comparison.

Note that a cryptanalyst who attacks an R-round scheme doesn't need an R-round approximation or characteristic. One can assume that for SHARK an $(R-2)$-round characteristic or approximation can be used. Also, the probability

of the best differential can be several times higher than the probability of the best characteristic. Equivalently, the correlation between input bits and output bits of the cipher is only approximated by the product of the correlations in each round. It is clear that for a 64-bit cipher with a fixed key the probability of a differential is larger than 2^{-63}, or it is equal to zero. Also the correlations between input and output bits are multiples of 2^{-63}. More specific, in a 64-bit block cipher the expected value for the probability of a differential is upper bounded by $128 \cdot 2^{-64}$ [O94]. The probabilities and correlations in the table were calculated by assuming independent and variable round keys; they are probabilities over the input and round key space. These values give only an indication of the safety margin against linear and differential attacks. When the probability of a characteristic or the correlation of a linear approximation drops below 2^{-63}, it can be considered as irrelevant.

For applications where a conservative security margin is much more important than encryption speed, one can use more rounds. If one uses the "probabilities" and "correlations" of table 1 as a measure, eight rounds of SHARK give a security level equivalent to triple-DES. More conservative people could use SHARK with eight or ten rounds, and with a 128-bit key.

SHARK			DES		
R	p (dc)	c^2 (lc)	R	p (dc)	c^2 (lc)
2	2^{-54}	2^{-54}	4	$2^{-9.6}$	2^{-6}
4	2^{-108}	2^{-108}	8	$2^{-30.5}$	$2^{-19.5}$
6	2^{-162}	2^{-162}	12	$2^{-46.2}$	$2^{-31.5}$
			48	2^{-128}	2^{-148}

Table 1. Probabilities for the best differential characteristics and linear approximations as a function of the number of rounds. We give the values for R-round characteristics/approximations.

4 Implementation Considerations

First we show how to combine the S-boxes and the diffusion layer in one operation. Then we show how the structure of SHARK enables us to exploit this feature both in the encryption and the decryption mode. Let $X_1, \ldots, X_n$ denote the input of a round, after the key addition, and let $Y_1, \ldots, Y_n$ denote the output. We have:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \ldots \\ Y_n \end{bmatrix} = A \cdot \begin{bmatrix} S_1[X_1] \\ S_2[X_2] \\ \ldots \\ S_n[X_n] \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} \\ a_{21} \\ \dots \\ a_{n1} \end{bmatrix} \cdot S_1[X_1] \oplus \begin{bmatrix} a_{12} \\ a_{22} \\ \dots \\ a_{n2} \end{bmatrix} \cdot S_2[X_2] \oplus \dots \oplus \begin{bmatrix} a_{1n} \\ a_{2n} \\ \dots \\ a_{nn} \end{bmatrix} \cdot S_n[X_n] .$$

Here the S_i are the $m \times m$ substitution tables, "$\oplus$" and "$\cdot$" denote addition and multiplication in $GF(2^n)$, and A is the matrix that defines the diffusion layer. We can write this as follows:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \dots \\ Y_n \end{bmatrix} = \begin{bmatrix} a_{11} \cdot S_1[X_1] \\ a_{21} \cdot S_1[X_1] \\ \dots \\ a_{n1} \cdot S_1[X_1] \end{bmatrix} \oplus \begin{bmatrix} a_{12} \cdot S_2[X_2] \\ a_{22} \cdot S_2[X_2] \\ \dots \\ a_{n2} \cdot S_2[X_2] \end{bmatrix} \oplus \dots \oplus \begin{bmatrix} a_{1n} \cdot S_n[X_n] \\ a_{2n} \cdot S_n[X_n] \\ \dots \\ a_{nn} \cdot S_n[X_n] \end{bmatrix} .$$

With the expanded $m \times nm$ substitution tables T_i:

$$T_i[X] = \begin{bmatrix} a_{1i} \cdot S_i[X_i] \\ a_{2i} \cdot S_i[X_i] \\ \dots \\ a_{ni} \cdot S_i[X_i] \end{bmatrix} ,$$

the combined operation becomes:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \dots \\ Y_n \end{bmatrix} = T_1[X_1] \oplus T_2[X_2] \oplus \dots \oplus T_n[X_n] .$$

This operation needs only n table lookups and $n-1$ bitwise additions and shifts (of nm-bit values).

The key addition can be incorporated into the S-boxes as well. Addition with a fixed key before a substitution table is equivalent to a simple rearrangement of the rows of the table. In the case of the key dependent affine transform it is even more important for the performance to incorporate the key operation into the substitution tables. If κ is a diagonal matrix, this operation is again equivalent to a rearrangement of the rows of the substitution table.

A nice property of the Feistel structure is that encryption mode and decryption mode of the block cipher are very similar: only the order of the round keys has to be reversed. For SHARK the conversion from encrypting mode to decrypting mode is a bit more involved. We explain now how the round keys and the combination of the S-boxes and the diffusion layer have to be modified for the decrypting mode. The function of the inverse diffusion layer at the end of the cipher becomes clear.

Consider a two-round version of SHARK. Denote by l the linear operation, by s the non-linear substitution, and by a_{k_i} the key addition with key k_i. The encryption function is then:

$$Y = (l^{-1} \circ a_{k_3} \circ l \circ s \circ a_{k_2} \circ \underbrace{l \circ s}_{r} \circ a_{k_1})(X) , \qquad (1)$$

where r denotes the combined S-box-diffusion operation. Since the key addition and the diffusion layer are both linear operations, we can interchange their order:

$$\begin{aligned}(l \circ a_k)(X) &= B \cdot (\kappa \cdot X \oplus K) = (B \cdot \kappa \cdot X) \oplus (B \cdot K) \\ &= ((B \cdot \kappa \cdot B^{-1}) \cdot (B \cdot X)) \oplus (B \cdot K) \\ &= (a_{k'} \circ l)(X), \end{aligned} \tag{2}$$

with

$$\begin{aligned} a_{k'}(X) &= \kappa' \cdot X \oplus K' \\ &= (B \cdot \kappa \cdot B^{-1}) \cdot X \oplus (B \cdot K) \end{aligned}$$

By applying (2) to $l^{-1} \circ a_{k_3}$ the encryption becomes:

$$Y = (a_{k_3'} \circ l^{-1} \circ l \circ s \circ a_{k_2} \circ r \circ a_{k_1})(X) \tag{3}$$

$$= (a_{k_3'} \circ s \circ a_{k_2} \circ r \circ a_{k_1})(X). \tag{4}$$

The last equation is actually implemented. The equation contains R table lookups and $R+1$ key additions; R key additions can be incorporated into the table lookups. By inverting (1), we obtain the decryption operation:

$$X = (a_{k_1^{-1}} \circ s^{-1} \circ l^{-1} \circ a_{k_2^{-1}} \circ s^{-1} \circ l^{-1} \circ a_{k_3^{-1}} \circ l)(Y).$$

By interchanging key addition and diffusion we obtain:

$$X = (a_{k_1^{-1}} \circ s^{-1} \circ a_{k_2^{-1'}} \circ \underbrace{l^{-1} \circ s^{-1}}_{r'} \circ a_{k_3^{-1'}})(Y).$$

This equation has the same structure as the encryption operation, as given by (4).

5 Performance

Since SHARK operates on 64-bit words, it will benefit from a 64-bit architecture. Table 2 compares the performance of C-implementations of SHARK, SHARK* (SHARK with key dependent S-boxes), SAFER, IDEA, and MD5 on a 266 MHz DEC–ALPHA and on a 90 MHz Pentium. On a Pentium SHARK runs at approximately the same speed as SAFER. Experiments with smaller S-boxes showed that this degradation of performance is due to the limited on-chip cache size; doubling this size will probably result in a speedup from 0.80 Mbyte/s to 2.3 Mbyte/s.

	ALPHA	Pentium
SHARK	6.30 Mbyte/s	0.800 Mbyte/s
SHARK*	5.10 Mbyte/s	
SAFER	1.03 Mbyte/s	0.725 Mbyte/s
IDEA	1.53 Mbyte/s	
MD5	16.00 Mbyte/s	7.500 Mbyte/s

Table 2. Performance of SHARK, SAFER, IDEA, and MD5 on a 64-bit workstation, and on a Pentium.

6 Further work

With the same building blocks, one could also construct a Feistel cipher. If one considers the combination of non-linear and diffusion layer as the S-boxes, this cipher would have about the same form as CAST. Both ciphers use S-boxes with small inputs and large outputs, but the F-function of the Feistel variant of SHARK has a guaranteed diffusion and non-linearity. Also, because it is invertible, the round function is balanced, which is a desirable property for the round function of a Feistel cipher with a small number of rounds [RP95].

If one incorporates the key addition into the S-boxes, key-dependent S-boxes are the result. However these key-dependent S-boxes have no "weak keys" where one can find differentials with high probability [V96].

Because of the guaranteed diffusion, there are no good characteristics or linear relations. Therefore we expect to obtain a higher resistance against linear and differential cryptanalysis with a smaller number of rounds.

The modular design allows for easy extension of the cipher to a 128-bit cipher. This can be done by doubling the number of parallel S-boxes n, or by doubling the input size of the boxes m. A simple calculation shows that the resistance against cryptanalysis is higher for the scheme with doubled m. However, S-boxes with input size 16 impose high memory requirements, and therefore a doubled number of boxes n seems a better choice. For both schemes it is easy to find an RS-code that can be used in the diffusion layer.

References

[AT95] C.M. Adams, S.E. Tavares, "Designing S-boxes for ciphers resistant to differential cryptanalysis," *Proc. of the 3rd symposium on State and Progress of Research in Cryptography*, W. Wolfowicz, Ed., Fondazione Ugo Bordoni, 1993, pp. 181–190.

[BS90] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *Journal of Cryptology*, Vol. 4, No. 1, 1991, pp. 3–72.

[C94] D. Coppersmith, "The data encryption standard (DES) and its strength against attacks," *IBM Journal of Research and Development*, Vol. 38, No. 3, May 1994, pp. 243–250.

[DGV93] J. Daemen, R. Govaerts, J. Vandewalle, "Block ciphers based on modular arithmetic," *Proc. of the 3rd Symposium on the State and Progress of Research in Cryptography*, W. Wolfowicz, Ed., Fondazione Ugo Bordoni, Roma, 1993, pp. 80–89.

[DGV94] J. Daemen, R. Govaerts, J. Vandewalle, "Weak keys of IDEA," *Advances in Cryptology, Proc. Crypto'93, LNCS 773*, D. Stinson, Ed., Springer-Verlag, 1994, pp. 224–231.

[DGV94b] J. Daemen, R. Govaerts, J. Vandewalle, "A new approach to block cipher design," *Fast Software Encryption, LNCS 809*, R. Anderson, Ed., Springer-Verlag, 1994, pp. 18–32.

[D95] J. Daemen, "Cipher and hash function design strategies based on linear and differential cryptanalysis," *Doctoral Dissertation*, March 1995, K.U.Leuven.

[DT91] M.H. Dawson, S.E. Tavares, "An expanded set of S-box design criteria based on information theory," *Advances in Cryptology, Proc. Eurocrypt'91, LNCS 547*, D.W. Davies, Ed., Springer-Verlag, 1991, pp. 352–367.

[F73] H. Feistel, "Cryptography and computer privacy," *Scientific American*, Vol. 228, No. 5, May 1973, pp. 15–23.

[FNS75] H. Feistel, W.A. Notz, J.L. Smith, "Some cryptographic techniques for machine-to-machine data communications," *Proc. IEEE*, Vol. 63, No. 11, November 1975, pp. 1543–1554.

[FIPS46] *Data Encryption Standard*, Federal Information Processing Standard (FIPS), Publication 46, National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January 1977.

[ISO10116] *"Information technology – Security techniques – Modes of operation of an n-bit block cipher algorithm,"* IS 10116, ISO/IEC, 1991.

[KMI91] K. Kim, T. Matsumoto, H. Imai, "A recursive construction method of S-boxes satisfying strict avalanche criterion," *Advances in Cryptology, Proc. Crypto'90, LNCS 537*, S. Vanstone, Ed., Springer-Verlag, 1991, pp. 564–575.

[K95] L.R. Knudsen, "Truncated and higher order differentials," *Fast Software Encryption, LNCS 1008*, B. Preneel, Ed., Springer-Verlag, 1995, pp. 196–211.

[K96] L.R. Knudsen, "Truncated differentials of SAFER," *Fast Software Encryption* (this volume), 1996.

[LH94] S.K. Langford, M.E. Hellman, "Differential-linear cryptanalysis," *Advances in Cryptology, Proc. Crypto'94, LNCS 839*, Y. Desmedt, Ed., Springer-Verlag, 1994, pp. 17–25.

[MS77] F.J. MacWilliams, N.J.A. Sloane, *"The Theory of Error-Correcting Codes,"*, North-Holland, Amsterdam, 1977.

[M94] J. Massey, "SAFER K-64: a byte-oriented block-ciphering algorithm," *Fast Software Encryption, LNCS 809*, R. Anderson, Ed., Springer-Verlag, 1994, pp. 1–17.

[M95] J. Massey, "SAFER K-64: One year later," *Fast Software Encryption, LNCS 1008*, B. Preneel, Ed., Springer-Verlag, 1995, pp. 212–241.

[M93] M. Matsui, "Linear cryptanalysis method for DES cipher," *Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765*, T. Helleseth, Ed., Springer-Verlag, 1994, pp. 386–397.

[N91] K. Nyberg, "Perfect nonlinear S-boxes," *Advances in Cryptology, Proc. Eurocrypt'91, LNCS 547*, D.W. Davies, Ed., Springer-Verlag, 1991, pp. 378–386.

[N94] K. Nyberg, "Differentially uniform mappings for cryptography," *Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765*, T. Helleseth, Ed., Springer-Verlag, 1994, pp. 55–64.aa

[O94] L. O'Connor, "On the distribution of characteristics in bijective mappings," *Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765*, T. Helleseth, Ed., Springer-Verlag, 1994, pp. 360–370.

[PW72] W.W. Peterson, E.J. Weldon, *"Error-Correcting Codes,"* The MIT Press, Cambridge, 1972.

[RP95] V. Rijmen, B. Preneel, "On weaknesses of non-surjective round functions," *Workshop on Selected Areas in Cryptography – SAC'95*, Ottawa, May 18–19, 1995, pp. 100–106.

[S94] B. Schneier, "Description of a new variable-length key, 64-bit block cipher (Blowfish)," *Fast Software Encryption, LNCS 809*, R. Anderson, Ed., Springer-Verlag, 1994, pp. 191–204.

[SB95] B. Schneier, M. Blaze, "MacGuffin: an unbalanced Feistel network block cipher," *Fast Software Encryption, LNCS 1008*, B. Preneel, Ed., Springer-Verlag, 1995, pp. 97–110.

[V96] S. Vaudenay, "On the weak keys of Blowfish," *Fast Software Encryption*(this volume), 1996.

A Construction of G

We explain in more detail the construction of a $(2n, n, n+1)$-Reed-Solomon code over the Galois field $GF(2^m)$ and the calculation of G, the echelon form of the generation matrix.

Let α be a primitive element in $GF(2^m)$. Then the polynomial

$$g(x) = (x + \alpha) \cdot (x + \alpha^2) \cdot \ldots \cdot (x + \alpha^n)$$

generates a $(2n, n, n+1)$-Reed-Solomon code. The code words are formed by the polynomials of degree $< 2n$ that are multiples of $g(x)$. G can be constructed in the following way [PW72]. Let $b_i(x)$ be the remainder after dividing x^i by $g(x)$:

$$x^i = g(x) \cdot q(x) + b_i(x)\,.$$

Then

$$x^i + b_i(x) = g(x) \cdot q(x)$$

is a code word. We take these code words, for $i = 2n-1, 2n-2, \ldots, n$, as rows of G. Then

$$G = [I_{n \times n}\, B_{n \times n}]$$

is the echelon form of the generation matrix of C.

B Reference Implementation

A reference implementation of SHARK can be retrieved by anonymous ftp from `ftp.esat.kuleuven.ac.be/pub/COSIC/rijmen`.