Fast Diffusion Block for Secret Key Cryptography

Vlastimil Klíma [1]

April 26, 2022

Abstract. We present a diffusion block (DB), which is extraordinarily fast. After one round, it reaches complete diffusion, which means only 16 memory reads and 15 $\oplus$ operations. It uses only the most common operations available in any microprocessor. The diffusion and speed are based on a large key, about 64 kB for encryption and 34 kB for decryption, expanded from the classical key size of 128, 256, or more bits. The basic block length is 128 bits and could be expanded to 192, 256, or more. DB uses the same core idea as uses AES, DES, and others, which has been studied for more than 50 years by many cryptanalysts.

Keywords: diffusion, design principles, secret key cryptography.

1 Introduction

We hope that the idea of the proposed diffusion block is new[2]. For example, DB uses different procedures and tables for encryption and decryption. One diffusion block (one round) reaches complete diffusion, and every new round fundamentally improves the cryptographic, statistical, algebraical, and complexity properties. Every block increases the degree of underlying polynomials by 8. In the paper, we will concentrate on DB with block length 128.

2 Notation

The scheme works with bytes, vectors, and matrices. We will identify bytes with the elements of the Galois field $GF(2^8)$. All details of computing in $GF(2^8)$ are well described in the numerous literature on AES [AES01]. We use just the same Galois field $GF(2^8)$, operations $\oplus, \bullet$, and the irreducible polynomial $m(x) = x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1$. The byte $a = (a_7, a_6 \dots a_0)$, where a_7 is the highest bit and a_0 is the lowest bit, is identified with the polynomial $a = a_7x^7 \oplus a_6x^6 \oplus \dots \oplus a_0x^0$. The multiplication of two bytes $a \bullet b$ is defined as the multiplication of the corresponding polynomials modulo $m(x)$. Recall that

[1] Independent cryptologist, Prague, v.klima@volny.cz, http://cryptography.hyperlink.cz

[2] if not, let us know please, and we will update the references accordingly

except for zero, each byte $x \in GF(2^8)$ has its inverse element $x^{-1} \in GF(2^8)$ such that $x \bullet x^{-1} = 1$. We will also use row and column vectors of bytes. We mark their coordinates by lower indexes, for instance, $C = (C_0, C_1 \ldots C_{15})$. The transposition of a row vector C into a column vector is referred to as C^T. We denote the matrix of 16x16 bytes by $M = (M_{i,j}), i = 0 \ldots 15, j = 0 \ldots 15$. The column i of M could be written shortly as M_i. Multiplying the vector C by byte b is defined as $C \bullet b = (C_0, C_1 \ldots C_{15}) \bullet b = (C_0 \bullet b, C_1 \bullet b \ldots C_{15} \bullet b)$, multiplication matrix by a vector is written as
$M \bullet C^T = (\sum_{j=0}^{15} M_{0,j} \bullet C_j, \sum_{j=0}^{15} M_{1,j} \bullet C_j \ldots \sum_{j=0}^{15} M_{15,j} \bullet C_j)^T$ and multiplication of two matrices gives the matrix $D = A \bullet B = A_{0-15,0-15} \bullet B_{0-15,0-15} = D_{0-15,0-15}$, where $D_{i,j} = \sum_{k=0}^{15} A_{i,k} \bullet B_{k,j}$ for every $i, j = 0 \ldots 15$. Every non-singular matrix M has its inverse M^{-1}, such that $M \bullet M^{-1} = I$, where I is the identity matrix. For a short notation we denote $m \equiv M^{-1}$ and we will also omit $\bullet$ in some products. We will also use substitution boxes $S_{i,j}, i = 0 \ldots 15, j = 0 \ldots 15$ over $GF(2^8)$. When we ought to index the bit of variable already indexed, we will use the last index, for instance the bit r of $S_{k,n}$ is denoted as $S_{k,n,r}$. We denote $C = EncDB(P)$ encryption of the plaintext P and $P = DecDB(C)$ decryption of the ciphertext C.

3 Definition of DB

We will use the terms Encryption Key and Decryption Key because different tables form them.

3.1 Definition of the Decryption Key

We define **S** as a set of 256 S-boxes $S_{i,j}, i = 0 \ldots 15, j = 0 \ldots 15$. They are mappings $GF(2^8) \rightarrow GF(2^8) : x \rightarrow S_{i,j}(x)$ with the following properties:

- 120 S-boxes above the diagonal, $S_{i,j}, i < j$, are random **mappings**
- 16 S boxes on the diagonal, $S_{i,j}, i = j$, are random **bijections**
- 120 S-boxes under the diagonal, $S_{i,j}, i > j$, are constant **zero**

We define $\mathbf{M} = M_{0-15,0-15}$ as non-singular matrix 16x16 over $GF(2^8)$ with nonzero elements $M_{i,j}$. **Decryption Key** is (**S**, **M**).

3.2 Definition of the Encryption Key

Let us denote $T = (T_0, T_1 \ldots T_{15})$ 16 tables $T_k : GF(2^8) \rightarrow GF(2^8)^{16}, k = 0 \ldots 15$, which maps a byte to 16 bytes. In the following definition, every table uses the same matrix M, but a unique set of 16 S-boxes, different from the others. For each $x \in GF(2^8)$ and $k = 0 \ldots 15$ we define:

$$T_k(x) = M_0 \bullet S_{k,0}(x) \oplus M_1 \bullet S_{k,1}(x) \oplus \ldots \oplus M_{15} \bullet S_{k,15}(x) \qquad (1)$$

Encryption Key is (**T**).

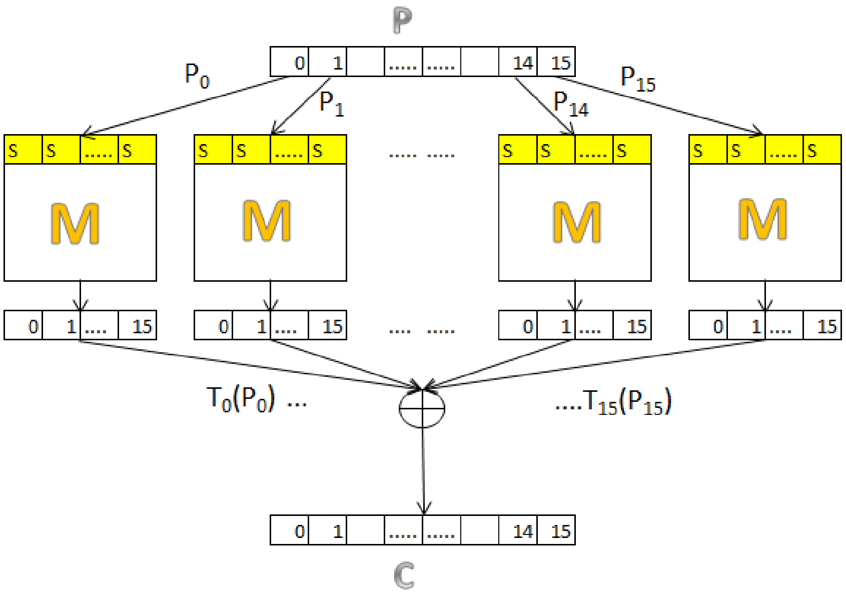
3.3 Encryption

On the side of encryption, it suffices to have only the Encryption Key T, while S-boxes and matrix M are unnecessary. The encryption takes a block of plaintext $P = (P_0, P_1 \dots P_{15})^T$, $P \in GF(2^8)^{16}$ and outputs a block of ciphertext $C = (C_0, C_1 \dots C_{15})^T$, $C \in GF(2^8)^{16}$, where

$$C = \begin{bmatrix} C_0 \\ C_1 \\ \vdots \\ C_{15} \end{bmatrix} = T_0(P_0) \oplus T_1(P_1) \oplus \dots \oplus T_{15}(P_{15}) = \sum_{k=0}^{15} T_k(P_k) \tag{2}$$

The equation (2) says that for each position $k = 0 \dots 15$ of the plaintext byte, we use a different table T_k, which maps it to a 16-byte pseudo-random vector. The ciphertext is the sum of these vectors. From (2) follows that **encryption uses only 16 memory reads and 15 $\oplus$ operations**.

Figure 1: The scheme of encryption



In Figure 1, every singular byte of P goes through 16 unique S-boxes giving a vector, which is then multiplied by the matrix M. This is a contribution of one plaintext byte to the ciphertext. **A single table lookup realizes this idea**. The ciphertext is the sum of all these contributions. It creates fast diffusion, not perfect, but the diffusion of all plaintext bytes into all ciphertext bytes. Underline that this is the core idea of the paper. We forward that two or more consecutive rounds of DB will give better and better diffusion from a statistical, complexity, algebraic, and cryptographic point of view while still very fast.

3.4 Decryption

Here we know the Decryption Key **M(m) and S**. At first, we introduce temporary variables $CS_0 = S_{0,0}(P_0), CS_1 = S_{0,1}(P_0) \oplus S_{1,1}(P_1), \ldots, CS_{15} = \sum_{k=0}^{15} S_{k,15}(P_k)$, generally

$$CS_n = \sum_{k=0}^{n} S_{k,n}(P_k), n = 0 \ldots 15 \tag{3}$$

According to (1) $C = \sum_{k=0}^{15} T_k(P_k) = \sum_{k=0}^{15} M \bullet (S_{k,0}(P_k), S_{k,1}(P_k) \ldots S_{k,15}(P_k))^T = M \bullet (\sum_{k=0}^{15} S_{k,0}(P_k), \sum_{k=0}^{15} S_{k,1}(P_k) \ldots \sum_{k=0}^{15} S_{k,15}(P_k))^T$. So we have $C = M \bullet (CS_0, CS_1, \ldots, CS_{15})^T$ and from it we compute the values $CS_0, CS_1 \ldots CS_{15}$:

$$(CS_0, CS_1 \ldots CS_{15})^T = m \bullet C. \tag{4}$$

Then from the values

$CS_0 = S_{0,0}(P_0)$
$CS_1 = S_{0,1}(P_0) \oplus S_{1,1}(P_1)$
$CS_2 = S_{0,2}(P_0) \oplus S_{1,2}(P_1) \oplus S_{2,2}(P_2)$
. . . .
$\mathrm{CS}_{13} = S_{0,13}(P_0) \oplus S_{1,13}(P_1) \oplus \ldots \oplus S_{13,13}(P_{13})$
$CS_{14} = S_{0,14}(P_0) \oplus S_{1,14}(P_1) \oplus S_{2,14}(P_2) \oplus \ldots \oplus S_{14,14}(P_{14})$
$CS_{15} = S_{0,15}(P_0) \oplus S_{1,15}(P_1) \oplus S_{2,15}(P_2) \oplus \ldots \oplus S_{14,15}(P_{14}) \oplus S_{15,15}(P_{15})$

we will compute $P_0, P_1 \ldots P_{15}$, what finishes the decryption.

4 Semi-linearity in $GF(2^8)$

We are working over $GF(2^8)$, which means that the formula

$T_k(x) = M_0 \bullet S_{k,0}(x) \oplus M_1 \bullet S_{k,1}(x) \oplus \ldots \oplus M_{15} \bullet S_{k,15}(x)$

is linear over $GF(2^8)$ when M is fixed, or when S is fixed. Because the ciphertext is a sum of $T_k(x)$, it is semi-linear also.

5 Variants and Parameters

5.1 Key Generation

The tables T are computed from M and S. We have two methods of their generation. The first consists of the generation of the matrix and S-boxes directly from the random source. The second one is the key expansion. In this case, we generate only a random secret seed (the real KEY). Then we use this seed and well-known techniques of key expansion to generate a pseudo-random sequence as long as we need. We derive the matrix M and all S-boxes with the requested properties from it. Note that the pseudo-random sequence has to be generated so that it would be impossible to derive a part of the sequence from any other part. For instance, for this purpose, AES key expansion isn't good, because we can derive other round keys and the seed itself from one round key. The sequence SHA-256(seed || counter) for counter= 1, 2, 3, ... is a simple example of good key expansion.

5.2 The Block Length

We can define DB variants for the block lengths 16 + 4*B bytes ($B = 1, 2 \ldots$), i.e. 128, 160, 192, 256,... bits analogously as for 128 bits. Let us describe DB-256. The formulas are the same as for DB-128, with the following changes. The matrix M is of type 32x32, and the vectors have 32 bytes, there are 32 tables T_k for $k = 0 \ldots 31$, the values $T_k(x)$, $x \in GF(2^8)$ are 32-bytes vectors, and the plaintexts and the ciphertexts are 32 bytes long. **S** is a set of 1024 S-boxes $S_{i,j}, i = 0 \ldots 31, j = 0 \ldots 31$. The encryption and decryption use the same formulas and procedures.

5.3 The Number of Rounds

The number of rounds is a tool for better diffusion from statistical, complexity, algebraic, and cryptographic points of view. We define NR rounds DB simply as a composition of NR functions DB: $ciphertext = DB^{NR}(plaintext)$. Note that in the second round we can use the same or another key (T, M, S), etc.

6 Cryptanalysis

6.1 Algebraic analysis in $GF(2^8)$

Let us express the ciphertext according to (2) and (1): $C_n = \sum_{k=0}^{15}(T_k(P_k))_n = (\sum_{k=0}^{15}\sum_{q=0}^{15} M_q \bullet S_{k,q}(P_k))_n$. So the byte C_n is equal to

$$C_n = \sum_{k=0}^{15}\sum_{q=0}^{15} M_{n,q} \bullet S_{k,q}(P_k), n = 0 \ldots 15. \tag{5}$$

We can define 16*16*16*256 new variables $Y_{n,q,k,x} = M_{n,q} \bullet S_{k,q}(x)$, $n, q, k = 0 \ldots 15, x = 0 \ldots 255$. Every pair of plaintext and ciphertext creates 16 equations in $GF(2^8)$ for these variables:

$C_n = \sum_{k=0}^{15}\sum_{q=0}^{15} Y_{n,q,k,P_k}, n = 0 \ldots 15.$

So, we need only 16*16*16*256/16 = 65536 pairs of (P, C) to determine all unknowns $Y_{n,q,k,x}$.

Another way is using (4): We have $CS_n = (m \bullet C)_n = \sum_{q=0}^{15} m_{n,q} \bullet C_q$, where on the other hand $CS_n = \sum_{k=0}^{n} S_{k,n}(P_k), n = 0 \ldots 15$. So we have 16 equations

$$\sum_{q=0}^{15} m_{n,q} \bullet C_q = \sum_{k=0}^{n} S_{k,n}(P_k), n = 0 \ldots 15. \tag{6}$$

For different known P and C, the variables C_q and P_k become known coefficients, and the quadratic equation (6) becomes linear with varying coefficients for 256 unknowns $m_{n,q}$ and 136*256 = 34816 unknowns $S_{k,n}(P_k)$. To determine m and all S, we need only (34816+256)/16 = 2192 known pairs (P, C).

Knowing the input and output of one round DB enables us to determine its insider secrets. It is the property following semi-linearity.

6.2 Algebraic analysis of more DB rounds in $GF(2^8)$

If we know the plaintext and ciphertext of one round DB, we can substitute them into P, C, M, and S relations. These variables are then changed into coefficients and "disappear". Only linear relations remain. If we use two rounds, the intermediate ciphertext is unknown, so the relations have quadratic members, which don't disappear now. Moreover, the unknown variables (not constants) now index other unknown variables (S-boxes), which creates the main obstacle for composing simple equations, as we did before. Let us denote $C^{(1)} = Enc(P), C^{(2)} = Enc(C^{(1)})$. For the first round we have $C_k^{(1)} = \sum_{l=0}^{15} M_{k,l} \bullet \sum_{t=0}^{l} S_{t,l}(P_t)$ and for the second round (backwards) $(m \bullet C^{(2)})_q = CS_q(C^{(1)}) = \sum_{k=0}^{q} S_{k,q}(C_k^{(1)})$, what together gives a relation between the plaintext P and the final ciphertext $C^{(2)}$:

$$(m \bullet C^{(2)})_q = \sum_{k=0}^{q} S_{k,q}(\sum_{l=0}^{15} M_{k,l} \bullet \sum_{t=0}^{l} S_{t,l}(P_t)) \tag{7}$$

Now, we can see what an obstacle in the equation (7) is. It is the sum of expressions, which enters S-boxes $S_{k,q}$ on the right side. S-boxes are not linear, so we must use their arguments as new unknowns. As a maximum, we can express $C^{(1)}$ as a sum of new variables:

$C_k^{(1)} = \sum_{l=0}^{15} M_{k,l} \bullet \sum_{t=0}^{l} S_{t,l}(P_t) = \sum_{l=0}^{15} \sum_{t=0}^{l} M_{k,l} \bullet S_{t,l}(P_t)$
Let us denote $Y_{k,l,t} = M_{k,l} \bullet S_{t,l}(P_t)$,then

$C_k^{(1)} = \sum_{l=0}^{15} \sum_{t=0}^{l} Y_{k,l,t}$

So we have expressed $C^{(1)}$ as a sum of unknowns, but this sum now enters the S-box in the second round:

$(m \bullet C^{(2)})_q = \sum_{k=0}^{q} S_{k,q}(C_k^{(1)})$
When we express $S_{k,q}(x)$ as Lagrange polynomial of a variable $x \in GF(2^8)$, it

will have a degree of 255:

$$S_{k,q}(x) = \sum_{i=0}^{255} a_i \bullet x^i$$

and will contain the variable $C_k^{(1)}$ in degrees up to 255. More rounds will, of course, increase the degree of polynomials. We see the same situation as in the second round of AES, where a linear combination of S-boxes outputs enters S-boxes in the next round. So far, no simple solution to this problem has been found.

6.3 Algebraic analysis in $GF(2)$

Let us work in $GF(2)$. In GF(2), we will understand S-box as 8 functions determined by the i-th bit of the output $S(x)_i$. An argument (byte) x we understand as 8 variables, which are bits of x. Then for every $i = 0, 1 \ldots 7, S(x)_i$ is a polynomial of the degree 8 in variables $x_0, x_1 \ldots x_7$:

$$S(x)_i = \sum_{\alpha_0,\alpha_1\ldots\alpha_7=0}^{1} (x_0 \oplus \alpha_0 \oplus 1)(x_1 \oplus \alpha_1 \oplus 1)\ldots(x_7 \oplus \alpha_7 \oplus 1)S(\alpha_0, \alpha_1 \ldots \alpha_7)_i.$$

Recall that DB uses multiplications in $GF(2^8)$, so we need to express these products in bits. Let $a = (a_7, a_6 \ldots a_0)$, $b = (b_7, b_6 \ldots b_0)$, then the product $c = a \bullet b$ is in $GF(2^8)$ defined as the multiplication of the corresponding polynomials modulo $m(x) = x^8 \oplus x^4 \oplus x^3 \oplus x \oplus 1$:
$c = a \bullet b = ((a_7x^7 \oplus a_6x^6 \oplus ... \oplus a_0x^0) * (b_7x^7 \oplus b_6x^6 \oplus \ldots \oplus b_0x^0) \bmod m(x)$
$= (a_7b_7)x^{14} \oplus (a_7b_6 \oplus a_6b_7)x^{13} \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^{12} \oplus \ldots \oplus (a_0b_0)x^0) \bmod m(x) = (c_7x^7 \oplus c_6x^6 \oplus \ldots \oplus c_0x^0)$.

At first we will prepare reductions:
$x^8 = x^4 \oplus x^3 \oplus x^1 \oplus 1$
$x^9 = x^5 \oplus x^4 \oplus x^2 \oplus x^1$
$x^{10} = x^6 \oplus x^5 \oplus x^3 \oplus x^2$
$x^{11} = x^7 \oplus x^6 \oplus x^4 \oplus x^3$
$x^{12} = x^8 \oplus x^7 \oplus x^5 \oplus x^4 = x^4 \oplus x^3 \oplus x^1 \oplus 1 \oplus x^7 \oplus x^5 \oplus x^4 = x^7 \oplus x^5 \oplus x^3 \oplus x^1 \oplus 1$
$x^{13} = x^8 \oplus x^6 \oplus x^4 \oplus x^2 \oplus x^1 = x^4 \oplus x^3 \oplus x^1 \oplus 1 \oplus x^6 \oplus x^4 \oplus x^2 \oplus x^1 = x^6 \oplus x^3 \oplus x^2 \oplus 1$
$x^{14} = x^7 \oplus x^4 \oplus x^3 \oplus x^1$

$c = (a_7b_7)x^{14} \oplus (a_7b_6 \oplus a_6b_7)x^{13} \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^{12} \oplus (a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7)x^{11} \oplus (a_7b_3 \oplus a_6b_4 \oplus a_5b_5 \oplus a_4b_6 \oplus a_3b_7)x^{10} \oplus (a_7b_2 \oplus a_6b_3 \oplus a_5b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_2b_7)x^9 \oplus (a_7b_1 \oplus a_6b_2 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)x^8 \oplus (a_7b_0 \oplus a_6b_1 \oplus a_5b_2 \oplus a_4b_3 \oplus a_3b_4 \oplus a_2b_5 \oplus a_1b_6 \oplus a_0b_7)x^7 \oplus (a_6b_0 \oplus a_5b_1 \oplus a_4b_2 \oplus a_3b_3 \oplus a_2b_4 \oplus a_1b_5 \oplus a_0b_6)x^6 \oplus (a_5b_0 \oplus a_4b_1 \oplus a_3b_2 \oplus a_2b_3 \oplus a_1b_4 \oplus a_0b_5)x^5 \oplus (a_4b_0 \oplus a_3b_1 \oplus a_2b_2 \oplus a_1b_3 \oplus a_0b_4)x^4 \oplus (a_3b_0 \oplus a_2b_1 \oplus a_1b_2 \oplus a_0b_3)x^3 \oplus (a_2b_0 \oplus a_1b_1 \oplus a_0b_2)x^2 \oplus (a_1b_0 \oplus a_0b_1)x^1 \oplus a_0b_0$

$= (a_7b_7)(x^7 \oplus x^4 \oplus x^3 \oplus x^1) \oplus (a_7b_6 \oplus a_6b_7)(x^6 \oplus x^3 \oplus x^2 \oplus 1) \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)(x^7 \oplus x^5 \oplus x^3 \oplus x^1 \oplus 1) \oplus (a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7)(x^7 \oplus x^6 \oplus x^4 \oplus x^3) \oplus (a_7b_3 \oplus a_6b_4 \oplus a_5b_5 \oplus a_4b_6 \oplus a_3b_7)(x^6 \oplus x^5 \oplus x^3 \oplus x^2) \oplus (a_7b_2 \oplus a_6b_3 \oplus a_5b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_2b_7)(x^5 \oplus x^4 \oplus x^2 \oplus x^1) \oplus (a_7b_1 \oplus a_6b_2 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)(x^4 \oplus x^3 \oplus x^1 \oplus 1) \oplus (a_7b_0 \oplus a_6b_1 \oplus a_5b_2 \oplus a_4b_3 \oplus a_3b_4 \oplus a_2b_5 \oplus a_1b_6 \oplus a_0b_7)x^7 \oplus (a_6b_0 \oplus a_5b_1 \oplus a_4b_2 \oplus a_3b_3 \oplus a_2b_4 \oplus a_1b_5 \oplus a_0b_6)x^6 \oplus (a_5b_0 \oplus a_4b_1 \oplus a_3b_2 \oplus a_2b_3 \oplus a_1b_4 \oplus a_0b_5)x^5 \oplus (a_4b_0 \oplus a_3b_1 \oplus a_2b_2 \oplus a_1b_3 \oplus a_0b_4)x^4 \oplus (a_3b_0 \oplus a_2b_1 \oplus a_1b_2 \oplus a_0b_3)x^3 \oplus (a_2b_0 \oplus a_1b_1 \oplus a_0b_2)x^2 \oplus (a_1b_0 \oplus a_0b_1)x^1 \oplus a_0b_0$

$= (a_7b_7)x^7 \oplus (a_7b_7)x^4 \oplus (a_7b_7)x^3 \oplus (a_7b_7)x^1 \oplus (a_7b_6 \oplus a_6b_7)x^6 \oplus (a_7b_6 \oplus a_6b_7)x^3 \oplus (a_7b_6 \oplus a_6b_7)x^2 \oplus (a_7b_6 \oplus a_6b_7)x^0 \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^7 \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^5 \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^3 \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^1 \oplus (a_7b_5 \oplus a_6b_6 \oplus a_5b_7)x^0 \oplus (a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7)x^7 \oplus (a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7)x^6 \oplus (a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7)x^4 \oplus (a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7)x^3 \oplus (a_7b_3 \oplus a_6b_4 \oplus a_5b_5 \oplus a_4b_6 \oplus a_3b_7)x^6 \oplus (a_7b_3 \oplus a_6b_4 \oplus a_5b_5 \oplus a_4b_6 \oplus a_3b_7)x^5 \oplus (a_7b_3 \oplus a_6b_4 \oplus a_5b_5 \oplus a_4b_6 \oplus a_3b_7)x^3 \oplus (a_7b_3 \oplus a_6b_4 \oplus a_5b_5 \oplus a_4b_6 \oplus a_3b_7)x^2 \oplus (a_7b_2 \oplus a_6b_3 \oplus a_5b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_2b_7)x^5 \oplus (a_7b_2 \oplus a_6b_3 \oplus a_5b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_2b_7)x^4 \oplus (a_7b_2 \oplus a_6b_3 \oplus a_5b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_2b_7)x^2 \oplus (a_7b_2 \oplus a_6b_3 \oplus a_5b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_2b_7)x^1 \oplus (a_7b_1 \oplus a_6b_2 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)x^4 \oplus (a_7b_1 \oplus a_6b_2 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)x^3 \oplus (a_7b_1 \oplus a_6b_2 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)x^1 \oplus (a_7b_1 \oplus a_6b_2 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)x^0 \oplus (a_7b_0 \oplus a_6b_1 \oplus a_5b_2 \oplus a_4b_3 \oplus a_3b_4 \oplus a_2b_5 \oplus a_1b_6 \oplus a_0b_7)x^7 \oplus (a_6b_0 \oplus a_5b_1 \oplus a_4b_2 \oplus a_3b_3 \oplus a_2b_4 \oplus a_1b_5 \oplus a_0b_6)x^6 \oplus (a_5b_0 \oplus a_4b_1 \oplus a_3b_2 \oplus a_2b_3 \oplus a_1b_4 \oplus a_0b_5)x^5 \oplus (a_4b_0 \oplus a_3b_1 \oplus a_2b_2 \oplus a_1b_3 \oplus a_0b_4)x^4 \oplus (a_3b_0 \oplus a_2b_1 \oplus a_1b_2 \oplus a_0b_3)x^3 \oplus (a_2b_0 \oplus a_1b_1 \oplus a_0b_2)x^2 \oplus (a_1b_0 \oplus a_0b_1)x^1 \oplus (a_0b_0)x^0$

$= (a_7b_7 \oplus a_7b_5 \oplus a_7b_4 \oplus a_7b_0 \oplus a_6b_6 \oplus a_6b_5 \oplus a_6b_1 \oplus a_5b_7 \oplus a_5b_6 \oplus a_5b_2 \oplus a_4b_7 \oplus a_4b_3 \oplus a_3b_4 \oplus a_2b_5 \oplus a_1b_6 \oplus a_0b_7)x^7 \oplus$

$(a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7 \oplus a_7b_6 \oplus a_7b_3 \oplus a_6b_7 \oplus a_6b_4 \oplus a_6b_0 \oplus a_5b_5 \oplus a_5b_1 \oplus a_4b_6 \oplus a_4b_2 \oplus a_3b_7 \oplus a_3b_3 \oplus a_2b_4 \oplus a_1b_5 \oplus a_0b_6)x^6 \oplus$

$(a_7b_5 \oplus a_7b_3 \oplus a_7b_2 \oplus a_6b_6 \oplus a_6b_4 \oplus a_6b_3 \oplus a_5b_7 \oplus a_5b_5 \oplus a_5b_4 \oplus a_5b_0 \oplus a_4b_6 \oplus a_4b_5 \oplus a_4b_1 \oplus a_3b_7 \oplus a_3b_6 \oplus a_3b_2 \oplus a_2b_7 \oplus a_2b_3 \oplus a_1b_4 \oplus a_0b_5)x^5 \oplus$

$(a_7b_7 \oplus a_7b_4 \oplus a_7b_2 \oplus a_7b_1 \oplus a_6b_5 \oplus a_6b_3 \oplus a_6b_2 \oplus a_5b_3 \oplus a_5b_4 \oplus a_5b_6 \oplus a_4b_7 \oplus a_4b_5 \oplus a_4b_0 \oplus a_4b_4 \oplus a_3b_5 \oplus a_3b_6 \oplus a_3b_1 \oplus a_2b_7 \oplus a_2b_2 \oplus a_2b_6 \oplus a_1b_7 \oplus a_1b_3 \oplus a_0b_4 \oplus)x^4 \oplus$

$(a_7b_7 \oplus a_7b_6 \oplus a_7b_5 \oplus a_7b_4 \oplus a_7b_3 \oplus a_7b_1 \oplus a_6b_5 \oplus a_6b_7 \oplus a_6b_6 \oplus a_6b_4 \oplus a_6b_2 \oplus a_5b_3 \oplus a_5b_5 \oplus a_5b_7 \oplus a_5b_6 \oplus a_4b_7 \oplus a_4b_6 \oplus a_4b_4 \oplus a_3b_5 \oplus a_3b_7 \oplus a_3b_0 \oplus a_2b_1 \oplus a_2b_6 \oplus a_1b_7 \oplus a_1b_2 \oplus a_0b_3)x^3 \oplus$

$(a_7b_6 \oplus a_7b_3 \oplus a_7b_2 \oplus a_6b_7 \oplus a_6b_4 \oplus a_6b_3 \oplus a_5b_5 \oplus a_5b_4 \oplus a_4b_6 \oplus a_4b_5 \oplus a_3b_7 \oplus a_3b_6 \oplus a_2b_0 \oplus a_2b_7 \oplus a_1b_1 \oplus a_0b_2)x^2 \oplus$

$(a_7b_7 \oplus a_7b_5 \oplus a_7b_2 \oplus a_7b_1 \oplus a_6b_3 \oplus a_6b_6 \oplus a_6b_2 \oplus a_5b_7 \oplus a_5b_4 \oplus a_5b_3 \oplus a_4b_4 \oplus a_4b_5 \oplus a_3b_6 \oplus a_3b_5 \oplus a_2b_6 \oplus a_2b_7 \oplus a_1b_7 \oplus a_1b_0 \oplus a_0b_1)x^1 \oplus$

$(a_0b_07b_1 \oplus a_7b_6 \oplus a_7b_5 \oplus a_6b_7 \oplus a_6b_6 \oplus a_6b_2 \oplus a_5b_7 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus a_1b_7)x^0$

So, the bits of the product $c = a \bullet b$ are

$c_7 = a_7b_7 \oplus a_7b_5 \oplus a_7b_4 \oplus a_7b_0 \oplus a_6b_6 \oplus a_6b_5 \oplus a_6b_1 \oplus a_5b_7 \oplus a_5b_6 \oplus a_5b_2 \oplus$
$a_4b_7 \oplus a_4b_3 \oplus a_3b_4 \oplus a_2b_5 \oplus a_1b_6 \oplus a_0b_7$
$c_6 = a_7b_4 \oplus a_6b_5 \oplus a_5b_6 \oplus a_4b_7 \oplus a_7b_6 \oplus a_7b_3 \oplus a_6b_7 \oplus a_6b_4 \oplus a_6b_0 \oplus a_5b_5 \oplus a_5b_1 \oplus$
$a_4b_6 \oplus a_4b_2 \oplus a_3b_7 \oplus a_3b_3 \oplus a_2b_4 \oplus a_1b_5 \oplus a_0b_6$
$c_5 = a_7b_5 \oplus a_7b_3 \oplus a_7b_2 \oplus a_6b_6 \oplus a_6b_4 \oplus a_6b_3 \oplus a_5b_7 \oplus a_5b_5 \oplus a_5b_4 \oplus a_5b_0 \oplus a_4b_6 \oplus$
$a_4b_5 \oplus a_4b_1 \oplus a_3b_7 \oplus a_3b_6 \oplus a_3b_2 \oplus a_2b_7 \oplus a_2b_3 \oplus a_1b_4 \oplus a_0b_5$
$c_4 = a_7b_7 \oplus a_7b_4 \oplus a_7b_2 \oplus a_7b_1 \oplus a_6b_5 \oplus a_6b_3 \oplus a_6b_2 \oplus a_5b_3 \oplus a_5b_4 \oplus a_5b_6 \oplus a_4b_7 \oplus$
$a_4b_5 \oplus a_4b_0 \oplus a_4b_4 \oplus a_3b_5 \oplus a_3b_6 \oplus a_3b_1 \oplus a_2b_7 \oplus a_2b_2 \oplus a_2b_6 \oplus a_1b_7 \oplus a_1b_3 \oplus a_0b_4 \oplus$
$c_3 = a_7b_7 \oplus a_7b_6 \oplus a_7b_5 \oplus a_7b_4 \oplus a_7b_3 \oplus a_7b_1 \oplus a_6b_5 \oplus a_6b_7 \oplus a_6b_6 \oplus a_6b_4 \oplus a_6b_2 \oplus$
$a_5b_3 \oplus a_5b_5 \oplus a_5b_7 \oplus a_5b_6 \oplus a_4b_7 \oplus a_4b_6 \oplus a_4b_4 \oplus a_3b_5 \oplus a_3b_7 \oplus a_3b_0 \oplus a_2b_1 \oplus$
$a_2b_6 \oplus a_1b_7 \oplus a_1b_2 \oplus a_0b_3$
$c_2 = a_7b_6 \oplus a_7b_3 \oplus a_7b_2 \oplus a_6b_7 \oplus a_6b_4 \oplus a_6b_3 \oplus a_5b_5 \oplus a_5b_4 \oplus a_4b_6 \oplus a_4b_5 \oplus a_3b_7 \oplus$
$a_3b_6 \oplus a_2b_0 \oplus a_2b_7 \oplus a_1b_1 \oplus a_0b_2$
$c_1 = a_7b_7 \oplus a_7b_5 \oplus a_7b_2 \oplus a_7b_1 \oplus a_6b_3 \oplus a_6b_6 \oplus a_6b_2 \oplus a_5b_7 \oplus a_5b_4 \oplus a_5b_3 \oplus a_4b_4 \oplus$
$a_4b_5 \oplus a_3b_6 \oplus a_3b_5 \oplus a_2b_6 \oplus a_2b_7 \oplus a_1b_7 \oplus a_1b_0 \oplus a_0b_1$
$c_0 = a_7b_6 \oplus a_7b_5 \oplus a_7b_1 \oplus a_6b_7 \oplus a_6b_6 \oplus a_6b_2 \oplus a_5b_7 \oplus a_5b_3 \oplus a_4b_4 \oplus a_3b_5 \oplus a_2b_6 \oplus$
$a_1b_7 \oplus a_0b_0$

These equations define constants $\alpha_{s,t,r} \in \{0,1\}$, which tell us if the element a_sb_t is present in the equation for c_r. So we have

$$c_r = \sum_{s,t=0}^{7} \alpha_{s,t,r} a_s b_t \tag{8}$$

Note that there are 512 constants $\alpha_{s,t,r}$, but only 151 are ones. For a known pair (P, C), from (4), we have $m \bullet C = (CS_0, CS_1 \dots CS_{15})^T$, so $CS_n = (m \bullet C)_n = \sum_{q=0}^{15}(m_{n,q} \bullet C_q), n = 0 \dots 15$ and for the bit r of it we have: $CS_{n,r} = \sum_{q=0}^{15}(m_{n,q} \bullet C_q)_r = \sum_{q=0}^{15} \sum_{s,t=0}^{7} \alpha_{r,s,t} m_{n,q,s} C_{q,t}$. On the other hand from the definition of CS (3) we have: $CS_{n,r} = \sum_{k=0}^{n} S_{k,n,r}(P_k), n = 0 \dots 15, r = 0 \dots 7$. Together we have:

$$\sum_{q=0}^{15} \sum_{s,t=0}^{7} \alpha_{r,s,t} m_{n,q,s} C_{q,t} = \sum_{k=0}^{n} S_{k,n,r}(P_k) \qquad n = 0 \dots 15, r = 0 \dots 7. \tag{9}$$

Note that $S_{k,n,r}(P_k)$ is bit r of the byte $S_{k,n}(P_k)$. For S-box $S_{k,n}$ we usually have a table, which is indexed by an argument x, and the table value is a byte $S(x)$. Equivalently we can write the byte $S(x)$ per bit in 8 tables, indexed by x and the bit r. So we will have tables of bits $S_{k,n,r,x}$ for $r = 0 \dots 7$, and $x = 0 \dots 255$. Then we can write $S_{k,n,r}(P_k)$ as S_{k,n,r,P_k} and the equation (9) will be:

$$\sum_{q=0}^{15} \sum_{s,t=0}^{7} \alpha_{r,s,t} m_{n,q,s} C_{q,t} = \sum_{k=0}^{n} S_{k,n,r,P_k} \qquad n = 0 \dots 15, r = 0 \dots 7. \tag{10}$$

When we look at the left side of (10), there is only a sum of unknown bits $m_{n,q,s}$, and on the right side there is only a sum of unknown bits S_{k,n,r,P_k}. For known plaintext P and ciphertext C it constitutes 128 simple linear equations, containing 16*16*8 = 2048 unknown bits of m and 136*8*256 = 278528 unknown bits of S-boxes. The coefficients are changing with (P, C). Thus we are obtaining different linear equations. We need only (278528+2048)/128 = 280576/128 = 2192 pairs of (P, C) to solve this simple linear system of 280576 unknowns. We can suppose that the equations aren't linearly dependent so that we will determine the private key (m,S). So, when we know input (P) and output (C) bits of one round, we can express linear relations between bits of m and S in $GF(2)$ and determine them. We have obtained the same result as on a byte level.

6.4 Algebraic analysis of more rounds DB in GF(2)

According to (10) we will have for two rounds these relations:

$$\sum_{q=0}^{15}\sum_{s,t=0}^{7}\alpha_{r,s,t}m_{n,q,s}C^{(2)}_{q,t}=\sum_{k=0}^{n}S_{k,n,r,C^{(1)}_k}\qquad ,n=0\ldots 15, r=0\ldots 7 \tag{11}$$

and

$$\sum_{q=0}^{15}\sum_{s,t=0}^{7}\alpha_{r,s,t}m_{n,q,s}C^{(1)}_{q,t}=\sum_{k=0}^{n}S_{k,n,r,P_k}\qquad ,n=0\ldots 15, r=0\ldots 7. \tag{12}$$

The plaintext P and the ciphertext $C^{(2)}$ are changing on the left side of the equation (11) and the right side of the equation (12). But on the right side of the equation (11) we cannot say which variable $S_{k,n,r,C^{(1)}_k}$ is used. We need to use the concrete index, but we don't have it. Therefore, it is not a linear expression. It is a polynomial of the degree 8, where variables are unknown bits $C^{(1)}_{k,t}, t = 0\ldots 7$ of $C^{(1)}_k$. These variables are used in products with bits of m on the left side of the equation (12). Together we have nonlinear equations of the degree 8 + 1. More rounds will quickly increase the degree of polynomials. Again, we encountered a major barrier where a nonlinear S-box has to process the sum of some expressions. This problem has been studied for decades because it is the core idea of modern block ciphers. Up to now, there are no known techniques for effectively bypassing this obstacle, so we believe this construction isn't breakable.

7 Conclusion

We presented Diffusion Block, which is very fast. It uses a large key, different for encryption and decryption. Its cryptographic security is based on the same core idea as uses AES, DES, and others, which has been studied for more than 50 years by many cryptanalysts.

8 Post-quantum Cryptography

DB is resistant to attack by a quantum computer.

9 Acknowledgements

I would like to thank Danilo Gligoroski, Martin Stanek, Jozef Vyskoc, Pavel Vondruska, and Tomas Rosa for their helpful comments. Special thanks to Viktor Dohnal.

References

[AES01] AES-NIST. “Specification for the Advanced Encryption Standard (AES)”. In: *Federal Information Processing Standards Publication 197* (2001).