Int. J. Appl. Math. Comput. Sci., 2010, Vol. 20, No. 2, 401–411
DOI: 10.2478/v10006-010-0030-6

amcs

SCALABLE PP–1 BLOCK CIPHER

KRZYSZTOF BUCHOLC, KRZYSZTOF CHMIEL, ANNA GROCHOLEWSKA-CZURYŁO,
EWA IDZIKOWSKA, IZABELA JANICKA-LIPSKA, JANUSZ STOKŁOSA

Institute of Control and Information Engineering
Poznań University of Technology, pl. Marii Skłodowskiej-Curie 5, 60–965 Poznań, Poland
e-mail: {krzysztof.bucholc,krzysztof.chmiel,anna.grocholewska-czurylo}@put.poznan.pl
{ewa.idzikowska,izabela.janicka-lipska,janusz.stoklosa}@put.poznan.pl

A totally involutional, highly scalable PP-1 cipher is proposed, evaluated and discussed. Having very low memory requirements and using only simple and fast arithmetic operations, the cipher is aimed at platforms with limited resources, e.g., smartcards. At the core of the cipher's processing is a carefully designed S-box. The paper discusses in detail all aspects of PP-1 cipher design including S-box construction, permutation and round key scheduling. The quality of the PP-1 cipher is also evaluated with respect to linear cryptanalysis and other attacks. PP-1's concurrent error detection is also discussed. Some processing speed test results are given and compared with those of other ciphers.

Keywords: symmetric cipher, scalable cipher, S-box construction, resistance against cryptanalysis, error detection.

1. Introduction

Block ciphers constructed as a product of involutions are not new in cryptography. In fact, one of the most popular constructions, the Feistel permutation, is an involution. However, substitution and permutation encryption networks (SPNs) resulting from a product of involutions (e.g., both the nonlinear S-box layer and the affine bit permutation layer are involutions (Biryukov, 2003)) have not been intensively studied.

In this paper, we propose an n-bit ($n = 64, 128, 192, \ldots$) scalable block cipher which is an involutional SPN. We use one S-box which is an involution and one bit permutation which is also an involution. As a result, we get a totally involutional cipher. This means that we use the same network, in particular, the same S-box S ($S = S^{-1}$) and the same permutation P ($P = P^{-1}$) in both encryption and decryption phases.

Partial results were published in (Bucholc and Idzikowska, 2007; Chmiel *et al.*, 2008a; 2008b). The paper is organized as follows: Section 2 describes basic assumptions which lay at the base of the PP-1 project. The algorithm is described in detail in Sections 3–5. Quality evaluation of the PP-1 cipher is presented in Section 6. In Section 7, resistance against various attacks (differential, linear and algebraic cryptanalysis) is discussed. Avalanche and statistical properties of PP-1 are presented in Section 8. Section 9 presents processing speed obtained in test implementations of PP-1. In Section 10, concurrent error detection in hardware implementations of the cipher is considered. Final remarks are presented in Section 11.

2. Basic assumptions of the PP-1 cipher project

The main objective of the PP-1 project was to develop a block cipher which can be implemented on a platform with limited resources. Two other important requirements were

- scalability, which allows using different data block sizes and key sizes,
- easy and efficient implementation in software and hardware.

The ability to implement the algorithm on platforms with limited resources implies that operations should be simple and memory requirements as low as possible. One S-box is preferred, and the same resources (e.g., round keys) should be used for encryption and decryption. Since some simple processors, especially those used in embedded systems, do not support multiplication and division, these operations are costly because, to perform them, a sequence of more elementary instructions must be executed.

Therefore, multiplication and division should be avoided. Floating point operations are even more expensive when implemented in software and should also be avoided. Therefore, the preferred operation set contains sum modulo 2, addition and subtraction modulo 256, and shifts.

3. Processing path

Let $n = t \cdot 64$, where $t = 1, 2, 3, \ldots$. The scalable PP-1 cipher is a symmetric block cipher that in r rounds processes data blocks of n bits, using cipher keys with lengths of n or $2n$ bits. Let m denote the plaintext and let c be the ciphertext. Both the input (m or c) and output (c or m) of the PP-1 algorithm consist of sequences of n bits called blocks. The subblock is understood in the paper as a sequence of 64 or eight bits.

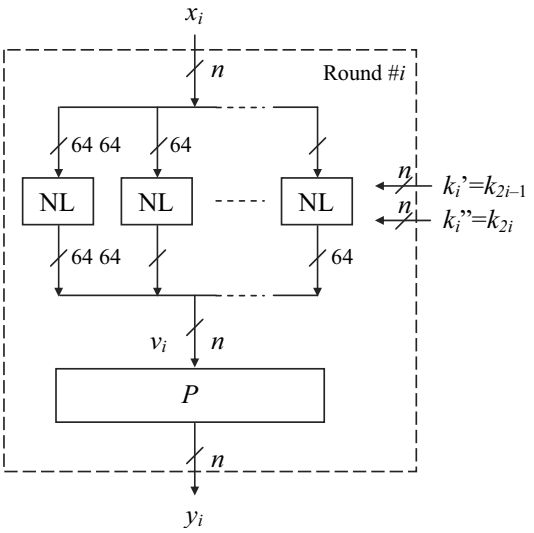


Fig. 1. One round of PP-1 ($i = 1, 2, \ldots, r$).

The PP-1 algorithm is an SP network. One round of the algorithm is presented in Fig. 1. It consists of $t = n/64$ parallel processing paths. A 64-bit nonlinear operation NL is performed in each path. Additionally, the n-bit permutation P that is an involution is used, i.e., $P^{-1} = P$. In the last round, called output transformation, the permutation P is not performed (i.e., in round #r, the permutation P = identity). Two n-bit round keys are used in each round.

The nonlinear element NL is shown in Fig. 2. In each round (#1 to #r), a 64-bit subblock is processed as eight 8-bit subblocks by four types of transformations, 8×8 S-boxes S, XOR ($\oplus$), addition ($\boxplus$) and subtraction ($\boxminus$) modulo 256 of integers represented by respective bytes. S-box S is an involution, i.e., $S^{-1} = S$.

Two n-bit round keys $k'_i = k_{2i-1}$ and $k''_i = k_{2i}$ are used in round i, $i = 1, 2, \ldots, r$. Let us denote the parallel

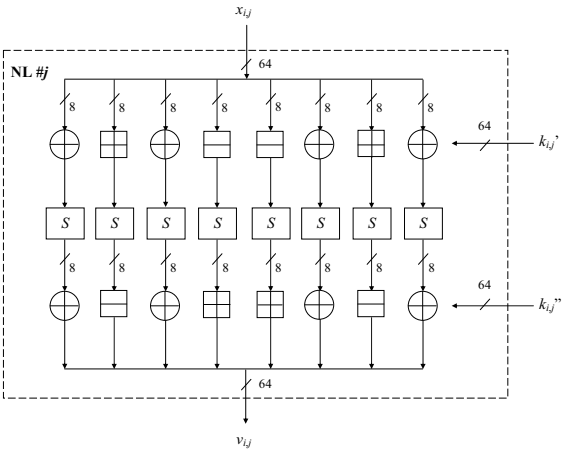


Fig. 2. Nonlinear element NL ($j = 1, 2, \ldots, t$).

processing paths from left to right as $j = 1, 2, \ldots, t$. Then

$$k'_i = k'_{i,1} \| k'_{i,2} \| \ldots \| k'_{i,t}, \quad k''_i = k''_{i,1} \| k''_{i,2} \| \ldots \| k''_{i,t}.$$

The 64-bit round subkeys $k'_{i,j}$ and $k''_{i,j}$ used in the element NL #j consist of eight 8-bit elementary keys

$$k_{i,j,l} \quad (l = 1, 2, \ldots, 8),$$

so that

$$k'_{i,j} = k'_{i,j,1} \| k'_{i,j,2} \| \ldots \| k'_{i,j,8}$$

and

$$k''_{i,j} = k''_{i,j,1} \| k''_{i,j,2} \| \ldots \| k''_{i,j,8}.$$

The same algorithm is used for encryption and decryption. However, if in the encryption process we use round keys $k_1, k_2, \ldots, k_{2r}$, then in the decryption process these keys must be used in reverse order, i.e., $k_{2r}, k_{2r-1}, \ldots, k_1$.

4. Elementary components of PP-1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	9E	BC	C3	82	A2	7E	41	5A	51	36	3F	AC	E3	68	2D	2A
1	EB	9B	1B	35	DC	1E	56	A5	B2	74	34	12	D5	64	15	DD
2	B6	4B	8E	FB	CE	E9	D9	A1	6E	DB	0F	2C	2B	0E	91	F1
3	59	D7	3A	F4	1A	13	09	50	A9	63	32	F5	C9	CC	AD	0A
4	5B	06	E6	F7	47	BF	BE	44	67	7B	B7	21	AF	53	93	FF
5	37	08	AE	4D	C4	D1	16	A4	D6	30	07	40	8B	9D	BB	8C
6	EF	81	A8	39	1D	D4	7A	48	0D	E2	CA	B0	C7	DE	28	DA
7	97	D2	F2	84	19	B3	B9	87	A7	E4	66	49	95	99	05	A3
8	EE	61	03	C2	73	F3	B8	77	E0	F8	9C	5C	5F	BA	22	FA
9	F0	2E	FE	4E	98	7C	D3	70	94	7D	EA	11	8A	5D	00	EC
A	D8	27	04	7F	57	17	E5	78	62	38	AB	AA	0B	3E	52	4C
B	6B	CB	18	75	C0	FD	20	4A	86	76	8D	5E	01	ED	46	45
C	B4	FC	83	02	54	D0	DF	6C	CD	3C	6A	B1	3D	C8	24	E8
D	C5	55	71	96	65	1C	58	31	A0	26	6F	29	14	1F	6D	C6
E	88	F9	69	0C	79	A6	42	F6	CF	25	9A	10	9F	BD	80	60
F	90	2F	72	85	33	3B	E7	43	89	E1	8F	23	C1	B5	92	4F

Fig. 3. S-box S.

4.1. Substitution S. The S-box S in Fig. 3 is a substitution function taking eight inputs and producing eight outputs. It is selected in such a way that it is its own inverse, i.e., $S^{-1} = S$.

This S-box has been generated using the multiplicative inverse procedure similar to AES (Daemen and Rijmen, 1999) with a randomly chosen primitive polynomial defining a Galois field. The nonlinearity of this S-box is 110 and its nonlinear degree is 7. Eight Boolean functions that constitute this S-box have nonlinearities equal to 110 or 112 and are all of degree 7.

According to recent studies (Fuller and Millan, 2002; 2003), S-boxes based on a multiplicative inverse in a finite field have a peculiar property that all component functions of the S-box are from the same affine equivalence class (all the output functions of the S-box can be mapped onto one another using affine transformations). Our S-box has been processed to remove this linear redundancy, so that all Boolean functions are now from different affine equivalence classes, while still maintaining the exceptionally high nonlinearity of the inverse mapping. The proposed S-box has a maximum XOR difference distribution table value of 4, which is extremely good.

Removing this linear redundancy is carried out by taking at random a pair of S-box elements and rearranging four (because of the self-inverse property) corresponding S-box entries in such a way that the S-box still remains its own inverse. After such a change, a test for linear redundancy is performed.

So how to check if an affine equivalence exists in an S-box? Many properties of Boolean functions covered by various cryptographic criteria remain unchanged by the affine transform, such as the algebraic degree and nonlinearity. The absolute values of the Walsh transform and the autocorrelation function are both only rearranged by affine transforms. The frequency distribution of the absolute values in these transforms is invariant under such affine transforms. To prove that two functions are from different equivalence classes it is then sufficient to show that either of their respective Walsh transform and autocorrelation function frequency distributions are different.

The S-box table can be represented as a two-dimensional table (Fig. 3). The input represented as a two-digit hexadecimal number is divided—the low order digit is on the horizontal axis, and the high order digit is on the vertical one. For example, to see what is the S-box output at input 6F, take 6 on the vertical axis and F on the horizontal axis. The S-box output is DA. As this S-box is its own inverse, it is easy to check that the S-box output at input DA is of course 6F.

4.2. Permutation P. The permutation P of the PP-1 block cipher is an n-bit involution, i.e., $P^{-1} = P$. Its main role is to scatter 8-bit output subblocks of S-boxes S in the n-bit output block of a round. The permutation P of PP-1 used in round #r is the identity operation.

For round #i, where $i = 1, 2, \ldots, r-1$, the permutation P is constructed using two algorithms, i.e., the auxiliary algorithm (Fig. 4) to compute auxiliary permutation Prm, and the main algorithm (Fig. 5) to compute permutation P.

```
Prm(x, nBb, nSb)                    {argument, number of block bits (e.g., 128),
                                     number of S-box bits (e.g., 8)}
1. nS ← nBb div nSb                 {number of S-boxes}
2. Sno ← x mod nS +1                {S-box number (from 1)}
3. Sb ← (x – 1) div nS + 1          {S-box bit (from 1)}
4. y ← (Sno – 1)· nSb + Sb          {value of auxiliary permutation}
5. return y
```

Fig. 4. Algorithm which computes auxiliary permutation Prm.

```
P(pno, nBb, nSb)                    {pair number (from 1), number of block bits
                                     (e.g., 128), number of S-box bits (e.g., 8)}
1. y ← Prm(pno, nBb div 2, nSb div 2) {value of auxiliary permutation}
2. px ← 2· pno – 1                  {odd argument (value) of involution}
3. py ← 2· y                        {even value (argument) of involution}
4. return (px, py)
```

Fig. 5. Algorithm which computes permutation P for rounds #1 to #$r-1$.

Algorithm Prm calculates bit mappings in permutation Prm, to scatter 4-bit subblocks in the $n/2$-bit block. Algorithm P calculates involutional pairs of bit mappings, in the n-bit permutation P. For each bit mapping in Prm, is constructed an involutional pair of bit mappings in P (Fig. 6).

The 128-bit permutation P is obtained as a result of 64 calls of Algorithm P for a pair numbered as pno from 1 to 64, the number of block bits $nBb = 128$ and the number of S-box bits $nSb = 8$. For example, for $pno = 2$, the value y of permutation Prm is equal to 9 and the resultant pair $(px, py) = (3, 18)$. Bit No. 18 in the output of P permutation has the same value as the third bit of its input and, moreover, since P is an involution, the third bit in the output has the same value as the 18-th bit of the input.

5. Round key scheduling

Round key scheduling is performed in $2r + 1$ iterations ($i = 0, 1, \ldots, 2r$), where r is the number of rounds. One iteration of key scheduling is presented in Fig. 7. The round keys $k_1, k_2, \ldots, k_{2r}$ are produced on outputs of iterations #1 to #$2r$.

The element KS of the iteration, shown in Fig. 8, is composed of substitution S, XOR ($\oplus$), addition ($\boxplus$) and subtraction ($\boxminus$) modulo 256. The operations are analogous to those in the data processing path described in Section 3.

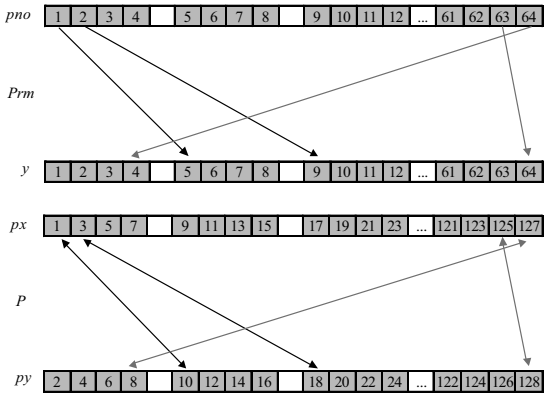


Fig. 6. Illustration for the construction of permutation P for $n = 128$.

The operation $RR(e_i)$ is the rotation of an n-bit block V_i by e_i bits to the right. The 4-bit integer e_i is obtained as the result of the XOR operation on four most significant bits (4MSBs) of the output of the two leftmost S-boxes. Thus for $V_i = b_1 b_2 \dots b_n$, where b_1 is the most significant bit, the value of e_i is calculated as follows:

$$e_i = E(b_1, b_2, \dots b_n) = (b_1 \oplus b_9)(b_2 \oplus b_{10})(b_3 \oplus b_{11})(b_4 \oplus b_{12}). \quad (1)$$

The entry X_0 of iteration #0 is supplied by an n-bit constant:

$$B = B_1 \| B_2 \| \dots \| B_t, \quad (2)$$

where 64-bit $B_1 =$ 912B4769B2496E7C (in the hexadecimal form), $B_j = Prm(B_{j-1})$ for $j = 2, 3, \dots, t$ and Prm is the auxiliary permutation calculated for parameters $nBb = 64$ and $nSb = 8$.

The inputs K_i for the iterations #0 and #1, $i = 0, 1$, are calculated in the following way. The cipher key k for the PP-1 algorithm is a sequence of n or $2n$ bits. If the key k has the length equal to n, then we put $K_0 = k$ and $K_1 = 0^n$, where 0^n denotes the concatenation of n zeros (analogously, the concatenation of n 1's will be denoted by 1^n). Otherwise, if the key k has the length equal to $2n$, then k is divided into two parts, k_H and k_L, of equal length ($k = k_H \| k_L$), and we set $K_0 = k_H$ and $K_1 = k_L$.

The values of K_i for the iterations #2 to #2r ($i = 2, 3, \dots, 2r$) are defined as follows:

- for the iteration #2 we take $K_2 = RL(B \oplus (A \wedge (K_0 \oplus K_1)))$, where $\wedge$ is the Boolean AND function, RL is the rotation by one bit to the left, and

$$A = \begin{cases} 0^n & \text{if} \quad |k| = 2, \\ 1^n & \text{if} \quad |k| = 2n, \end{cases} \quad (3)$$

- for the iterations #3 to #2r we take $K_i = RL(K_{i-1})$.

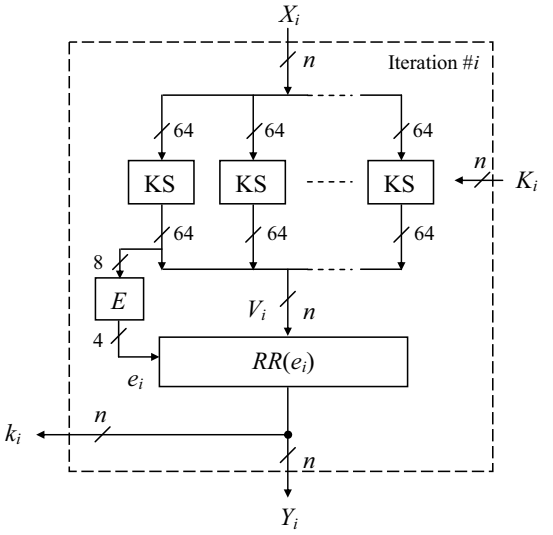


Fig. 7. One iteration of key scheduling ($i = 0, 1, \dots, 2r$).

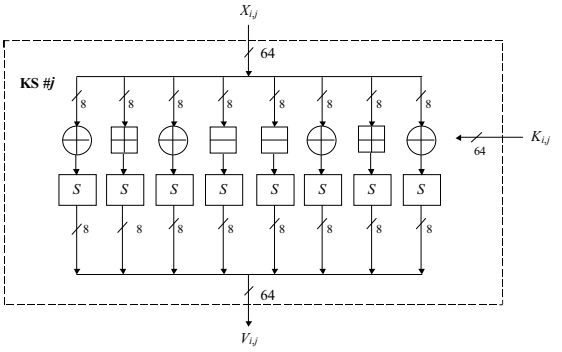


Fig. 8. KS—the main part of an iteration ($j = 1, 2, \dots, t$).

6. Evaluation of PP-1 cipher quality with respect to linear cryptanalysis

In the case of ciphers using sufficiently large S-boxes that are most resistant against differential and linear cryptanalysis (like our S-box S), linear cryptanalysis is more effective than differential one. Therefore, as an evaluation criterion of the PP-1 cryptographic quality we have chosen the upper bound of the effectiveness $|\Delta p_a^+|$ of the best nonzero linear approximation of the cipher. It is assumed that the PP-1 cipher quality is not worse than that of a comparative cipher with the same block length. For a given quality of the round function h of the PP-1 cipher, the evaluation of cipher quality reduces in fact to verification whether a sufficient number r of rounds is applied (Chmiel, 2006a; 2006b; 2006c).

6.1. Definitions. The basic idea of linear cryptanalysis is to describe a cipher by a linear approximate expression,

the so-called linear approximation. In general, the linear approximation of a function $y = f(x) : \{0,1\}^n \mapsto \{0,1\}^m$ is defined as an arbitrary equation of the form:

$$\bigoplus_{i \in y'} y_i = \bigoplus_{j \in x'} x_j, \tag{4}$$

which is fulfilled with approximation probability $p = N(x', y')/2^n$, where $x' \subseteq \{1, 2, \ldots, n\}$, $y' \subseteq \{1, 2, \ldots, m\}$ and $N(x', y')$ denotes the number of pairs (x, y) for which the equation holds. In particular, distinguish the zero approximation for which $x' = y' = \Phi$. The probability p of the zero approximation is equal to 1 for an arbitrary function f.

The effectiveness of the linear approximation of the function f is represented by the magnitude $|\Delta p| = |p - 1/2|$. Approximations with a positive value of the effectiveness measure are said to be effective. The effectiveness of the zero approximation $|\Delta p^0| = 1/2$; for the effectiveness of the nonzero approximation $|\Delta p^+| \leq 1/2$.

For an arbitrary function f, the only effective approximation such that $y' = \Phi$ is the zero approximation. A function f is said to be properly constructed if the only effective approximation such that $x' = \Phi$ is the zero approximation.

We say that a given S-box is of quality class q, if for the effectiveness of the nonzero approximation of its function f the following holds:

$$|\Delta p^+| \leq q/2^{\lfloor n/2 \rfloor + 1}. \tag{5}$$

6.2. Comparative algorithm. The comparative algorithm (Fig. 9) is a block cipher with a single round, which encrypts the n-bit plaintext m into the n-bit ciphertext c using n-bit key k in the following way:

$$c = S_p(m \oplus k). \tag{6}$$

Decryption is performed as follows:

$$m = S_p^{-1}(c) \oplus k. \tag{7}$$

The quality of the comparative algorithm depends on that of the S-box S_p. Assuming that the S-box S_p is of quality class q_p, we have

$$|\Delta p_p^+| \leq q_p/2^{\lfloor n/2 \rfloor + 1}. \tag{8}$$

6.3. Quality of the PP-1 cipher. For a properly constructed round function h of a block cipher with r rounds, an effective nonzero approximation of the cipher is a composition of r effective nonzero approximations of function h. Then the following formula holds:

$$\Delta p_a^+ = 2^{r-1} \prod_{i=1}^{r} \Delta p_i^+. \tag{9}$$

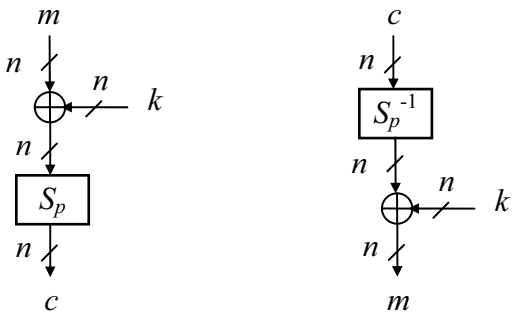


Fig. 9. Encryption and decryption performed by the comparative algorithm.

Assume that for the round function h, the following condition is fulfilled:

$$|\Delta p_i^+| \leq q_a/2^{\lfloor s/2 \rfloor + 1}, \tag{10}$$

where $i = 1, 2, \ldots, r$ and s is a constant. This assumption means that the effectiveness $|\Delta p_i^+|$ of function h approximation is not greater than the effectiveness $|\Delta p^+|$ of the best nonzero approximation of the S-box of quality class q_a, with s input bits. Then we obtain

$$|\Delta p_a^+| \leq (1/2) \cdot (q_a/2^{\lfloor s/2 \rfloor})^r. \tag{11}$$

Let us determine the number r of rounds required for a block cipher to reach the quality of the comparative algorithm. For the best nonzero approximations of the cipher and algorithm, the following should hold:

$$|\Delta p_a^+| \leq |\Delta p_p^+|\ . \tag{12}$$

Substituting upper bounds of effectiveness $|\Delta p_a^+|$ and effectiveness $|\Delta p_p^+|$, we have

$$(q_a/2^{\lfloor s/2 \rfloor})^r \leq q_p/2^{\lfloor n/2 \rfloor}. \tag{13}$$

Thus, for the number r of rounds, we obtain

$$r \geq (\lfloor n/2 \rfloor - \log q_p)/(\lfloor s/2 \rfloor - \log q_a). \tag{14}$$

The lower bound of r, for various n, $s = 8$, $q_a = 2$ and $q_p = 1$, is presented in Table 1.

Table 1. Lower bound of r for block length n ($s = 8, q_a = 2, q_p = 1$) and the number r of rounds.

n	64	128	192	256
Lower bound of r	10.7	21.3	32.0	42.7
r	11	22	32	43

7. Resistance against other attacks

The cipher key k for PP-1 is a sequence of n or $2n$ bits. We will use a general symbol κ to denote the length of the cipher key, $\kappa = n$ or $2n$. Let us consider the brute force attack and its time and memory complexities. The plaintext attack requires time $O(2^\kappa)$. Alternatively, the time for finding the key is $O(1)$, and the creation of the table for storing all keys requires $\kappa \cdot 2^\kappa$ bits of memory. If $n \geq 128$, then the brute force attack is inefficient. If we use the cipher with the block of the length $n = 64$, then the length of the key should be equal to 128.

From Section 6 it follows that the cipher is resistant against linear cryptanalysis. The resistance of PP-1 against differential cryptanalysis follows from the fact that both for randomly chosen and best constructed S-boxes of size $s \times s$, starting from some value of s, the linear approximation of S-box functions becomes more effective than differential approximation. This advantage of linear approximation rises with the increase in s. For DES size S-boxes it is not yet visible but for the S-box S of PP-1 it is (Chmiel, 2006a).

The best nonzero linear approximation of the S-box S of PP-1 has effectiveness $|\Delta p_S^+| = 18/256$. The effectiveness of the best nonzero differential approximation of the S-box is $\pi_S^+ = 4/256$. Assume that for the round function h we obtain the same values for the effectiveness of the best linear and differential approximation, i.e., $|\Delta p_h^+| = 18/256$ and $\pi_h^+ = 4/256$. Moreover, assume that the best nonzero approximations of PP-1 are composed of r best nonzero approximations of function h. Then for the best nonzero approximations of PP-1 the values of effectiveness $|\Delta p_a^+|$ and π_a^+ presented in Table 2 are obtained. The best nonzero linear approximation of PP-1 is evidently more effective than the differential one.

Table 2. Upper bounds of the effectiveness of PP-1 nonzero linear and differential approximations.

(n, r)	(64,11)	(128,22)	(192,32)	(256,43)
$\lvert\Delta p_a^+\rvert$	$1.83/2^{33}$	$1.67/2^{64}$	$1.35/2^{92}$	$1.24/2^{123}$
π_a^+	$1/2^{66}$	$1/2^{132}$	$1/2^{192}$	$1/2^{258}$

The PP-1 cipher is also resistant against algebraic attacks. Every S-box of dimension $s \times s$ can be described by e algebraic equations of multiple variables (Courtois and Pieprzyk, 2002). For a specific degree d of equations (usually $d = 2$) we can determine the actual number e of such equations $E(x_1, \ldots, x_s, y_1, \ldots, y_s)$. We are also interested in the number v of monomials that appear in these equations. Such a system of algebraic equations can be (approximately) sufficient (if it fully describes the S-box; this is the case if $e = s$), overdefined (if $e \gg s$) or sparse (if $v \ll \binom{s}{d}$).

For this reason it is possible to use the ratio v/e to estimate the quality of the system of equations. If v/e is close to 1, the S-box is considered bad. From this point of view, both overdefined systems (large e) and sparse systems (small v) will be bad. Otherwise, if the system is not overdefined and not sparse, $v/e \cong O(s^{d-1})$, then the S-box will be good. In the case of PP-1 we have $e \gg s = 8$. The complexity of the XSL attack described in (Courtois and Pieprzyk, 2002) with respect to PP-1, with block length n, equals $C(n) = v^{P \cdot \omega} \binom{u}{P}^\omega$, where u is the total number of S-boxes used in the cipher. As in (Courtois and Pieprzyk, 2002), we can compute other values: $v = 81$, $P = 8$, and $\omega = 2.3$. Hence, $C(n) = 81^{18.4} \binom{u}{8}^{2.3}$. In Table 3 we present the resulting complexities for different n (see Table 1). Theoretical analysis shows that algebraic attacks are not effective for PP-1.

Table 3. Complexity $C(n)$ of algebraic attacks.

n	64	128	192	256
$C(n)$	2^{199}	2^{236}	2^{257}	2^{273}

8. Avalanche and statistical properties of PP-1

8.1. Introduction. A number of statistical tests were carried out to check the quality of PP-1. We investigated the quality of the ciphertext and statistical properties of generated round keys. The avalanche effect was also studied. Three versions of the PP-1 were considered: 64-bit data block—128-bit key (PP-1/64_128), 128-bit data block—256-bit key (PP-1/128_256), and 256-bit data block—512-bit key (PP-1/256_512). The statistical test suite STS v. 1.8 (NIST, 2005), consisting of 15 tests, was used to check statistical properties of generated ciphertexts and round keys.

8.2. Testing ciphertext quality. Three modes of operation were considered: ECB, CBC and OFB. For ECB we used tests with a variable plaintext and a variable key. One bit was changed and the process of encryption was compared with the previous one to check if there are any regularities. For CBC and OFB modes, a message consisting of 1048576 bits (2^{20}), all zeros, was encrypted. The ciphertext was examined using the NIST test suite. This procedure was repeated for seven different keys. None of the executed tests showed any regularity.

8.3. Avalanche effect. For each version of the cipher, 1000 keys and 1000 plaintexts were generated randomly. Encryption was performed for each plaintext–key pair. As the next step, one bit in the plaintext was changed.

The modified plaintext was encrypted again. The encrypted text in each round was compared with that before the bit flip. The average Hamming distance (in per cents) for consecutive rounds is shown in Table 4. We can see that it takes five, six, and seven rounds, respectively, to have more than 49.9% of changed bits. We apply 11, 22, and 43 rounds. This means that the avalanche criterion is satisfied.

Table 4. Avalanche effect—the percentage of changed bits.

Round	Cipher version		
	PP-1/64_128	PP-1/128_256	PP-1/256_512
1	6.22	3.11	1.55
2	21.58	11.61	5.99
3	41.30	29.58	17.25
4	48.83	45.40	35.43
5	49.90	49.50	47.18
6	49.98	49.96	49.74
7	50.00	50.01	49.98
8	49.99	49.99	49.99
9	50.02	50.02	49.99
10	50.03	50.01	50.00
11	50.00	50.00	49.99
12		50.01	50.00
21		...	...
22		50.02	50.00
42			...
43			50.01

8.4. Testing of round keys. The round key scheduling procedure should generate round keys which are statistically independent. To test the round keys, we concatenated all round keys generated for a given key. Such a sequence was tested using the STS 1.8 package. Due to the small sequence length (1408, 5632, and 22016 bits), not all tests could be carried out. We used seven tests out of 15: Block Frequency, Serial, Approximate Entropy, Cumulative Sums, Runs, Spectral DFT, and Frequency tests. The procedure was repeated for seven different keys. No regularities in the tested sequences were found.

8.5. Conclusion. None of the tests we carried out indicated any regularity, either in round keys or in encrypted messages, which could cause hazard for PP-1 cipher safety.

9. Reference implementations

For test purposes, the PP-1 cipher was implemented in several programming languages (C++, Python, ZC-Basic) and environments (PC, Nokia 6600 mobile phone, BasicCard Pro) (Socha, 2008). As a reference, the Khazad cipher was implemented in two versions: one optimized for speed and the other optimized to run in a limited resources environment.

The test procedure involved running a full cipher cycle (in the case of the 64-bit version of PP-1 this was 11 rounds, based on results from Section 6.3, Table 1) and the result was the average speed of that iteration. Between 10^2 and 10^8 iterations were run to calculate the average speed, depending on the tested language/environment combination. Test results are summarized in Table 5.

Table 5. Speed results of the test implementation of PP-1.

	Phone		c| Card
	C++	Python	ZC-Basic
PP-1	2 Mb/s	4 kb/s	164 b/s
PP-1 (no perm)	4 Mb/s	7 kb/s	275 b/s
Khazad	1 Mb/s	2 kb/s	10 b/s
Khazad (opt.)	7 Mb/s	13 kb/s	—
	PC		
	C++	Python	ZC-Basic
PP-1	83 Mb/s	232 kb/s	243 b/s
PP-1 (no perm)	269 Mb/s	534 kb/s	402 b/s
Khazad	23 Mb/s	140 kb/s	14 b/s
Khazad (opt.)	301 Mb/s	533 kb/s	—

Clearly, the very visible difference in speed between the full PP-1 cipher and the version without permutation shows that the permutation operation is very costly in software implementations but in a hardware implementation it is negligible.

The full version of PP-1 is clearly faster than the similar Khazad cipher in its limited resources version, particularly in the case of the smart card implementation.

10. Concurrent error detection

10.1. Introduction. Several attacks on symmetric and public-key cryptosystems have been described in the literature and some dedicated error-detection techniques have been proposed to foil them. Various schemes for detecting faults in hardware implementations of several symmetric key encryption algorithms have been developed.

The motivation behind the increased interest in such detection schemes is based on two important observations.

First, ciphered communication is very sensitive to errors in input data or faults occurring during computation due to strong nonlinearity of encryption functions. The analysis of the effect of faults occurring during the encryption process for the advanced encryption standard algorithm (Bertoni *et al*., 2003b) for RC5 (Bertoni *et al*., 2003a) and for PP-1 (Bucholc and Idzikowska, 2007) has shown that even a single-bit error leads, after a few rounds of the algorithm, to a completely corrupted result. The second reason for the increased importance of error detection is the observation that attacks based on fault injection are feasible. In (Biham and Shamir, 1992), it is shown that a cryptographic device computing the Data Encryption Standard (DES) can be compromised by injecting a fault during computation. Depending on the cipher employed, useful data can be extracted by analyzing the resulting erroneous output.

The attacker induces a fault during cryptographic computations and the faulty results are used for key recovery. Concurrent Error Detection (CED) followed by the suppression of the corresponding faulty output can thwart fault injection attacks on symmetric block ciphers. By detecting the fault, either the output can be blocked (by producing a constant value such as all zeros) or a random output can be generated, misleading the attacker.

The feasibility of a fault attack or at least its efficiency depends on the exact capabilities of the attacker and the type of faults he or she can induce. Concurrent error detection techniques are widely used to enhance system dependability. All CED techniques introduce some form of redundancy. It may be noted that the general architecture of CED relies on the use of hardware redundancy for error detection, but time redundancy techniques (alternate data-retry and recomputation with shifted operands) can also be used for concurrent error detection. The hardware cost of time redundancy techniques is generally smaller than that of hardware redundancy, but system performance is directly affected.

The PP-1 cipher was designed for platforms with limited resources. Therefore it can be implemented in embedded systems, e.g., in simple smart cards, where a small area overhead and high reliability are very important.

10.2. Analysis of the influence of errors on PP-1 cipher behaviour. Errors in digital circuits can be either permanent or transient. Transient error detection methods also detect permanent errors. As the technology shrinks the circuit dimension, the probability of transient errors increases. Therefore in this section we will focus on transient error detection. We will present a detailed analysis for a 64-bit data block 128-bit key version of the PP-1 cipher.

Let us consider the data path of the PP-1 cipher shown in Fig. 10. We inserted and observed errors at five levels (0 to 4). Level 4 for round #i is the same as level 0 for round #$(i+1)$.

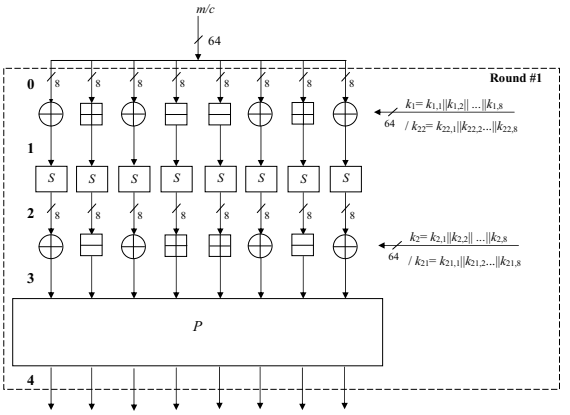


Fig. 10. Levels at which errors were inserted and observed.

To analyze errors distribution, a single transient error was induced. Then the state of signal lines was compared with those for an error-free circuit. This was repeated for all signal lines and all levels. Results are shown in Fig. 11. We can see that, due to the diffusion properties of the cipher, after about four rounds nearly half of the bits are faulty. The influence of the error during encryption on the decryption process is shown in Fig. 12.

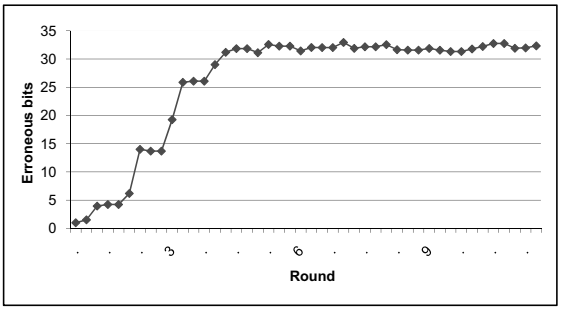


Fig. 11. Number of erroneous bits in the encrypted data block. Single error inserted in round #1 level 0.

From Figs. 11 and 12 we see that a faulty bit inserted in the first round of encryption causes a large number of erroneous bits in the final encrypted data. Applying decryption to the corrupted data reconstructs the data block containing one faulty bit. Injecting a single error in the encrypted data block in rounds #2–#11 results in a decrypted block which significantly differs from the original message.

The influence of multiple errors on the encrypted data was analyzed by the insertion of two, three and four errors. Results are shown in Fig. 13. In comparison with a single error, multiple errors inserted in round #1 lead to

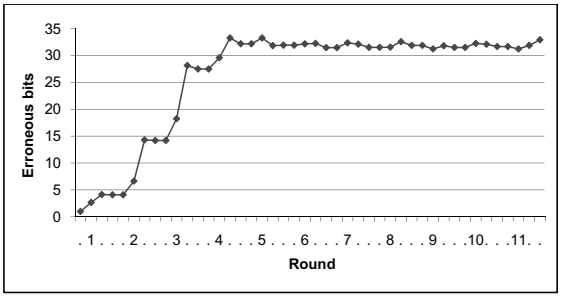


Fig. 12. Number of erroneous bits in the decrypted data block versus the injection round of the faulty bit. Error injected during encryption.

more erroneous data bits. But there is no significant difference in the number of erroneous bits in the encrypted message.

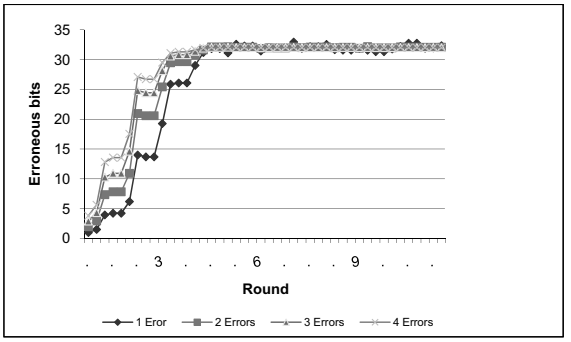


Fig. 13. Number of erroneous bits in the encrypted data block. Multiple errors and a single error inserted in round #1 level 0.

10.3. Faults in S-boxes. The S-box is a basic component of block ciphers, used to obscure the relationship between the plaintext and the ciphertext. It is an important element of the cryptographic algorithm and it should possess some properties which make linear and differential cryptanalysis as difficult as possible. Concurrent error detection in S-boxes of cryptographic hardware is very important.

In our discussion we use a realistic fault model in which either transient or permanent faults are induced randomly into the device. For example, a transient fault disturbs the smart card during its processing and affects a single execution of the algorithm. We speak about a permanent fault if there is a permanent damage of the smart card, such as cutting a wire or destroying a memory cell. We analysed the possibilities of detecting errors in the S-box of PP-1 block cipher implementation.

Let $D^1_{m-1}, \ldots, D^1_1, D^1_0$ be an error-free input vector of bits, and let $D^3_{m-1}, \ldots, D^3_1, D^3_0$ be an output vector. Let $E_{m-1}, \ldots, E_1, E_0$ be an error vector, where $E_i \in \{0, 1\}$; $E_i = 1$ indicates that bit i is faulty; the fault flips the bit. The number of ones in this vector is equal to the number of inserted faults. As a result, vector $D^2_{m-1}, \ldots, D^2_1, D^2_0$ is the erroneous vector, where $D^2_i = D^1_i \oplus E_i$, and the error is observable only on the S-box output (Idzikowska and Bucholc, 2007).

10.4. Parity prediction and simulation results. Parity prediction is a widely used CED technique. The even/odd parity function indicates whether the number of 1's in a set of binary digits is even or odd. The idea of using a single parity bit can be extended to multiple parity bits. This technique partitions the primary outputs into different parity groups. There is a parity bit associated with outputs in each parity group. The outputs of each parity group are checked using a parity checker.

We considered parity based CED schemes with one, two, four and eight parity bits. Particularly interesting results were observed for the scheme Par0_8 with eight parity bits $P_0, \ldots, P_7$ (Fig. 14). The bits are calculated in the following way:

$$P_j = \sum_{i=0}^{7} D^3_{i \cdot 8 + j} \bmod 2, \quad j = 0, 1, \ldots, 7. \tag{15}$$

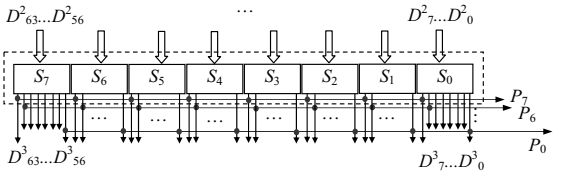


Fig. 14. Parity bit prediction scheme—Par0_8.

This means that one parity bit P_j controls outputs belonging to all boxes $S_0 - S_7$.

We used VHDL to model the S-box, and simulation was realized using Active-HDL simulation and verification environment. The results, i.e., the dependence of the error detection probability on the number of injected faults, are presented in Fig. 15 and compared with results obtained for only one parity bit (Par1) and those for for eight parity bits organized in such a way that there is one parity bit for each of S-boxes (Par8).

One of the conclusions of our work is that error detection using the parity code based approach can be successfully used in concurrent error detection in substitution blocks. It is possible to detect not only single but also multiple errors.

Another conclusion is that it is not the number of parity bits that is most important, but how the parity is predicted. Scheme Par0_8 with eight parity bits detects errors much better than Par8, which also has eight parity bits.

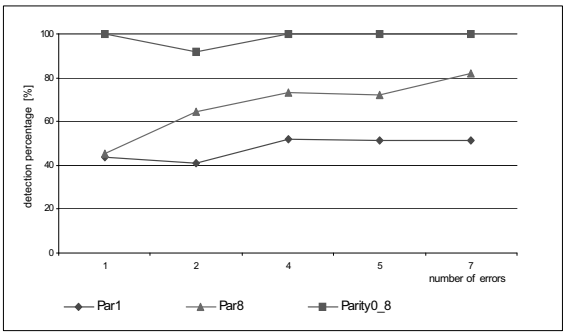


Fig. 15. Percentage of detected faults.

The proposed solution can be very useful for concurrent checking of cryptographic chips. We have shown that multiple errors in S-boxes can be detected with very high probability. Using the Par0_8 scheme of parity prediction, we achieved error detection which is close to 100%.

11. Final remarks

A new scalable block cipher was described in this paper.It is a simple, efficient and secure block cipher. Scalable PP-1 is aimed to be used on platforms with limited resources, and especially with a very limited amount of memory. Due to the fact that it uses only very simple arithmetic operations, the cipher can be implemented on different platforms such as smart cards, TV decoders, mobiles, etc. We could not find any significant constraint in it and did not insert any hidden weakness.

Acknowledgment

This work was partially supported by the Polish Ministry of Science as a 2005–2008 research project and partially by the PUT grant no. DS 45-083/09.

References

Bertoni, G., Breveglieri, L., Koren, I., Maistri, P. and Piuri, V. (2003a). Concurrent fault detection in a hardware implementation of the RC5 encryption algorithm, *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures and Processors, The Hague, The Netherlands*, pp. 410–419.

Bertoni, G., Breveglieri, L., Koren, I., Maistri, P. and Piuri, V. (2003b). Error analysis and detection procedures for a hardware implementation of the advanced encryption standard, *IEEE Transactions on Computers* **52**: 492–505.

Biham, E. and Shamir, A. (1992). Differential cryptanalysis of the full 16-round DES, *in* E. F. Brickell (Ed.), *CRYPTO*, Lecture Notes in Computer Science, Vol. 740, Springer, Heidelberg, pp. 487–496.

Biryukov, A. (2003). Analysis of involutional ciphers: Khazad and Anubis, *in* T. Johansson (Ed.), *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24–26, 2003, Revised Papers*, Lecture Notes in Computer Science, Vol. 2887, Springer, New York, NY, pp. 45–53.

Bucholc, K. and Idzikowska, E. (2007). Analysis of the influence of errors on the encryption and decryption in PP-1 block cipher, *Studia z Automatyki i Informatyki* **32**: 17–22.

Chmiel, K. (2006a). Distribution of the best nonzero differential and linear approximations of S-box functions, *Journal of Telecommunications and Information Technology* **3**: 8–13.

Chmiel, K. (2006b). Intermediate evaluation of block ciphers, *Proceedings of the 13th International Multi-Conference on Advanced Computer Systems ACS 2006, Międzyzdroje, Poland*, Vol. 1, pp. 331–342.

Chmiel, K. (2006c). On differential and linear approximation of S-box functions, *Biometrics, Computer Security Systems and Artificial Intelligence Applications, New York, NY, USA,* pp. 111–120.

Chmiel, K., Grocholewska, A., Socha, P. and Stoklosa, J. (2008a). Involutional block cipher for limited resources, *Global Communications Conference—GLOBECOM, New Orleans, LA, USA*, pp. 1852–1856.

Chmiel, K., Grocholewska, A., Socha, P. and Stoklosa, J. (2008b). Scalable cipher for limited resources, *Polish Journal of Environmental Studies* **17**(4C): 371–377.

Courtois, N. and Pieprzyk, J. (2002). Cryptanalysis of block ciphers with overdefined systems of equations, *in* Y. Zheng (Ed.), *ASIACRYPT*, Lecture Notes in Computer Science, Vol. 2501, Springer, Berlin/Heidelberg, pp. 267–287.

Daemen, J. and Rijmen, V. (1999). AES proposal: Rijndael, *Proceedings of the First Advanced Encryption Standard Candidate Conference, Ventura, CA, USA*.

Fuller, J. and Millan, W. (2002). On linear redundancy in the AES S-Box, *Cryptology ePrint Archive*, `http://eprint.iacr.org`.

Fuller, J. and Millan, W. (2003). Linear redundancy in S-boxes, *in* T. Johansson (Ed.) *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24–26, 2003, Revised Papers*, Lecture Notes in Computer Science, Vol. 2887, Springer-Verlag, New York, NY, pp. 74–86.

Idzikowska, E. and Bucholc, K. (2007). Concurrent error detection in S-boxes, *International Journal of Computer Science and Applications* **4**(1): 27–32.

Johansson, T. (Ed.) (2003). *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24–26, 2003, Revised Papers*, Lecture Notes in Computer Science, Vol. 2887, Springer-Verlag, New York, NY.

NIST (2005). Statistical test suite (version 1.8), `http://csrc.nist.gov/rng/rng2.html`.

Socha, P. (2008). Scalable PP-1 block cipher—Implementation, *Report No. 558*, Poznań University of Technology, Institute of Control and Information Engineering, Poznań.

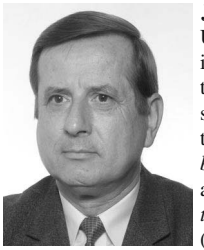
Krzysztof Bucholc is a senior lecturer at the Poznań University of Technology. His main research areas are computer architecture, embedded systems, reliability and diagnosis of computer hardware. He received a Ph.D. degree from the Poznań University of Technology in 1989. He is an author or coauthor of more than 50 published papers, two patents and one textbook.

Krzysztof Chmiel is an assistant professor at the Poznań University of Technology, Poland. His research and scientific interests focus on data security in information systems and cryptology, especially methods of designing and cryptanalysis of cryptographic algorithms. He is the author of a number of publications on differential and linear approximations of block ciphers and their component functions.

Anna Grocholewska-Czuryło received an M.Sc. degree in computer science from the Poznań University of Technology in 1993 and a Ph.D. in 2002 also from the PUT, where she gives many computer science courses. Her interests are mainly in field cryptography. She has published papers on block ciphers, Boolean functions properties and constructions (especially bent functions) as well as S-box design and analysis.

Ewa Idzikowska received an M.Sc. degree in computer science from the Wrocław University of Technology and the Ph.D. degree in computer science from the AGH University of Science and Technology, Cracow. She is a researcher at the Poznań University of Technology. Her research interests include reliability and diagnosis of logical circuits, test generation, fault diagnosis, and concurrent error detection, especially in hardware implementations of cryptosystems.

Izabela Janicka-Lipska is a senior lecturer at the Poznań University of Technology. She received a Ph.D. from the Poznań University of Technology in 2002, where she gives numerous computer science courses. Her main research areas are data security in information systems and cryptology.



Janusz Stokłosa is a professor at the Poznań University of Technology, Poland. His research interests include data security in information systems and cryptology, especially methods of designing cryptographic algorithms. He is the author of a number of publications, including *Algebraic and Structural Automata Theory* (1991, coauthor), *Cryptographic Method of Data Protection* (1992, in Polish), *Cryptographic Algorithms* (1994, in Polish), *Data Security in Information Systems* (2001, in Polish, coauthor), *Data Protection and Safeguards in IT Systems* (2003, in Polish, coauthor).

Received: 18 February 2009
Revised: 14 July 2009