Pure DDP-Based Cipher: Architecture Analysis, Hardware Implementation Cost and Performance up to 6.5 Gbps

Nikolay Moldovyan[1], Nicolas Sklavos[2], and Odysseas Koufopavlou[2]
[1]Specialized Center of Program Systems, SPECTR, Russia
[2]Electrical and Computer Engineering Department, University of Patras, Greece

Abstract: *Using Data-Dependent (DD) Permutations (DDP) as main cryptographic primitive, a new 64-bit block cipher is presented, ten-round DDP-64. Since the sum of all outputs of the conventional DDP is a linear Boolean function, non-linear DDP-based operation F is used additionally in DDP-64. The DDP-64 is a pure DDP-based cipher, i. e. it uses only permutations and the XOR operation. The designed cipher uses very simple key scheduling that defines high performance, especially in the case of frequent key refreshing. A novel feature of DDP-64 is the use of the switchable operation preventing the weak keys. The offered high level security strength does not sacrifice the implementation performance of DDP-64. Design and hardware implementation architectures of this cipher are presented. The synthesis results for both Field Programmable Gate Arrays (FPGA) and Application Specific Integrated Circuits (ASIC) implementations prove that DDP-64 is very flexible and powerful new cipher, especially for high speed WLANs and WPANs. The achieved hardware performance up to 6.5 Gbps and the implementation area cost of DDP-64 are compared with other ciphers, used in security layers of wireless protocols (Bluetooth, WAP, OMA, UMTS and IEEE 802.11). From these comparisons, it is proven that DDP-64 is a flexible new cipher with better performance in most of the cases, suitable for wireless communications networks of present and future.*

Keywords: *Data-dependent permutations, hardware implementation, fast encryption, block cipher, security.*

Received August 7, 2003; accepted December 23, 2003

1. Introduction

Security is a primary requirement of any wired and wireless communication. Encryption algorithms are meant to provide secure communications applications. However, if the system is not designed property, it may fail. New encryption algorithms have to perform efficiently in a variety of current and future applications, doing different encryption tasks. All hardware implementations have to be efficient, with the minimum allocated logic gates. This means simplicity in cipher's architectures with enough “clever” data transformation components. A communication protocol implementation, demands low power devices and fast computation components which imply that the number and complexity of the encryption operations should be kept as simply as possible. The ciphers of the near future have to be key agile. Many applications need a small amount of text to be encrypted with keys that are frequently changed. Many well know applications, like IPsec, use this way of algorithm's operation. Although the most widely used mode of operation is encryption with the same key for all the amount of transport data, the previous mode is also very useful for future applications. Ciphers that requiring subkeys precomputation have a lower key agility due to the precomputation time, and they also require extra RAM to hold the precomputed subkeys. This RAM requirement does not exist in the implementations of algorithms, which compute their keys during the encryption/ decryption operation. WLANs and WPANs technology demands specified characteristics of the cryptography science. Ciphers have to be compatible with wireless devices restricted standards in hardware resources.

One of the most important problems of the applied cryptography is the design of ciphers that are very fast with cheap hardware implementation. Data-Dependent (DD) Permutations (DDP) performed with so called Controlled Permutation (CP) boxes [6, 16, 17] appears to be very attractive cryptographic primitive for fast hardware encryption. Security estimation of the DDP-based ciphers CIKS-1 [13] and SPECTR-H64 [11] against linear cryptanalysis has shown that DDP are efficient, provided they are combined with other non-linear operations.

The DDP-based ciphers are proposed for cheap hardware implementation, ASIC and FPGA implementations of any DDP-based cipher are not yet described in the literature though. In this paper we present a new DDP-based cipher DDP-64 and the results of its ASIC and FPGA implementations. The design of the presented cipher takes into account some recommendations arising from the linear and differential analysis of other DDP-based ciphers [7, 11, 13]. A novel feature of DDP-64 is the use of the non-linear operation constructed as a special type of DDP demonstrating evidently that DDP are an efficient cryptographic primitive. The proposed cipher DDP-64 has been implemented in ASIC and FPGA hardware modules. Two different VLSI architectures are examined for each one of the two hardware devices (ASIC and FPGA). The synthesis results for all the hardware integrations are presented in detail.

The paper is organized in the following way. In section 2, we consider construction of the controlled operational boxes performing DDP. In section 3, we describe the structure of the cipher DDP-64 that is an example of the pure DDP-based ciphers based only on the use of XOR and bit permutation operations. Additional non-linear function F used in DDP-64 represents a special type DDP operation. The DDP-64 is designed to demonstrate efficiency of DDP in pure form. In section 4, we present results on the security strength of the proposed cipher. Section 5 describes the two examined architectures for both FPGA and ASIC integrations. The synthesis results for all the implementations are given in detail. Comparison of the proposed cipher performance with other ciphers used in security layers of both WLANs and WPANs protocols are presented. Finally, conclusions are discussed in the last section 6.

2. Design of the Controlled Permutations

Controlled permutations can be easy performed with well known Interconnection Networks (IN) [1, 4] which were proposed to construct key-dependent permutations [12, 19]. However such use of IN do not effectively thwarts differential cryptanalysis [20]. Regarding cryptographic applications it is more attractive to use IN to perform DDP on data subblocks [18] and subkeys [14]. An operational box $P_{n/m}$ performing permutations on n-bit binary vectors depending on some controlling m-bit vector V we shall call Controlled Permutation (CP) box. When controlling vector depends on some data subblock the CP box performs DDP.

Notation:

- Let $\{0, 1\}^n$ denote the set of all n-bit binary vectors $X = (x_1,\ldots x_n)$.
- Let X denote also decimal value (or simply value) of the vector.
- Let $X \oplus Y$ denote the bit-wise XOR operation performed on $X, Y \in \{0, 1\}^n$.
- Let $Y = X^{<<<k}$ denote rotation of the word X by k bits, where $Y = (y_1,\ldots, y_n)$ is the output $\forall I \in \{1,\ldots, n - k\}$ we have $y_i = x_{i+k}$, and $\forall I \in \{n - k + 1,\ldots, n\}$ we have $y_i = x_{i+k-n}$.
- Let $X_l = (x_1,\ldots, x_{n/2})$ and $X_h = (x_{n/2+1},\ldots, x_n)$ denote the least and the most significant halves of $X \in \{0, 1\}^n$.

The fast CP boxes can be constructed using switching elements $P_{2/1}$ (Figure 1-a) as elementary building blocks performing controlled transposition of two input bits x_1 and x_2. The $P_{2/1}$-box controlled with one bit v transforms two-bit input vector (x_1, x_2) into the output (y_1, y_2), where $y_1 = x_{1+v}$ and $y_2 = x_{2-v}$. Taking into account that it is very desirable to minimize the time delay while performing CP-box permutations the layered topology of IN can be considered as the main one since it permits to design very fast CPB. Layered CPB are constructed as superposition of $s = 2m/n$ active layers separated with $s - 1$ fixed permutations π_1 ,..., π_{s-1} that are implemented in hardware as simple connections. Each active layer (Figure 1-b) in a CPB with 2n-bit input is represented by the set of n parallel elementary boxes P2/ 1. General structure of the layered CPB is shown in Figure 1-c. Its notation is presented in in Figure 1-d. In all figures in this paper the solid lines indicate data movement, while dotted lines corresponding to CP-boxes indicate the controlling bits. A CP-box inverse of the box Pn/m can be denoted as P-1n/m. We assume that in a layered CP-box all elementary switching elements are consecutively numbered from left to right and from top to bottom and the jth bit of vector V controls the jth switching element P2/1. In accordance with the number of layers the vector V can be represented as concatenation of 2m/ n vectors V1, V2, ..., V2m/ n ({0, 1}n/ 2, i. e. V = (V1, V2, ..., V2m/ n).

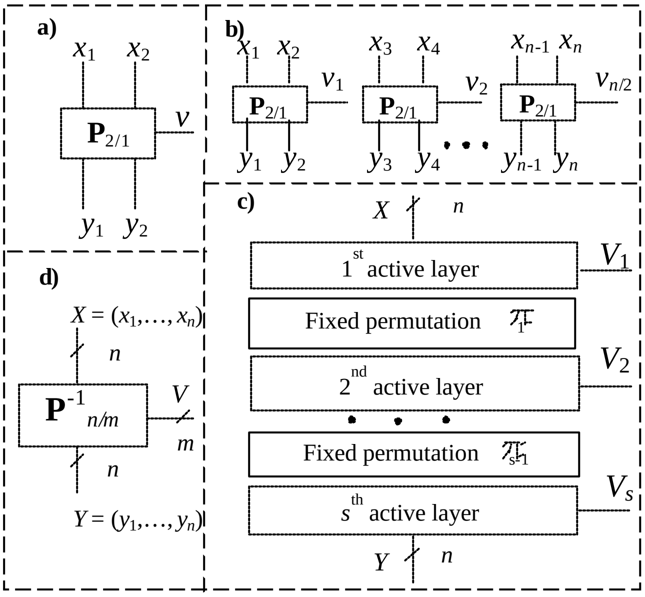


Figure 1. Notation of the $P_{2/1}$- (a), structure of one active layer (b), general structure of the layered CP-boxes (c), and $P^{-1}_{n/m}$-boxes (d).

Controlled permutations performed with the box $P_{n/m}$ can be characterized using an ordered set of the modifications $\{\Pi_0, \Pi_1,..., \Pi_{2m-1}\}$, where each modification Π_i, $i = 0, 1, ..., 2^{m-1}$, is a fixed permutation of some set of n bits. Permutations ? *i* we shall call CP-modifications. The following two definitions we use according to [17].

Definition: The P-boxes $P_{n/m}$ and $P^{1}_{n/m}$ are mutual inverses, if for all possible values of the vector *V* the corresponding CP-modifications Π_V and Π^{-1}_V are mutual inverses.

One active layer can be considered as some single-layer CP box L_n. It is evidently that $P_{2/1} = P^{-1}_{2/1}$, therefore $L_n = L^{-1}_n$. A layered CP box $P_{n/m}$ can be represented as superposition:

$$P_{n/m} = L_n^{(V_1)} \cdot p_1 \cdot L_n^{(V_2)} \cdot ... \cdot p_{s-1} \cdot L_n^{(V_s)}$$

The inverse box $P^{-1}_{n/m}$ has the following structure:

$$P^{-1}_{n/m} = L_n^{(V_s)} \cdot p^{-1}_{s-1} \cdot L_n^{(V_{s-1})} \cdot p^{-1}_{s-2} \cdot ... \cdot p_1 \cdot L_1^{(V_1)}$$

Thus, to construct inverse of the CP box Pn/m it is sufficient to number the boxes P2/1 from left to right and from bottom to top and to replace π_i by π^{-1}_{s-i} for all i = 1, …, S-1. We shall assume that in the boxes $P^{-1}_{n/m}$ switching elements $P_{2/1}$ are consecutively numbered from left to right from bottom to top. Note that the vector V_j corresponding to the *j*th active layer in the box $P_{n/m}$ controls the (2m/ n – j + 1)th active layer in $P^{-1}_{n/m}$.

The proposed cipher DDP-64 uses the boxes $P_{32/96}$ and $P^{-1}_{32/96}$. Each one of them is constructed using four parallel boxes $P_{8/12}$ and four parallel boxes $P^{-1}_{8/12}$ shown in Figure 2-a, b. The first and second groups of boxes are separated with permutational involution:

(1)(2,9)(3,17)(4, 25)(5)(6,13) (7,21)(8, 29)(10)(11,18) (12,26)(14)(15,22)(16,30)(19)(20,27)(23)(24,31)(28) (32).

The structures of the boxes $P_{32/96}$ and $P^{-1}_{32/96}$ are presented in Figure 2.

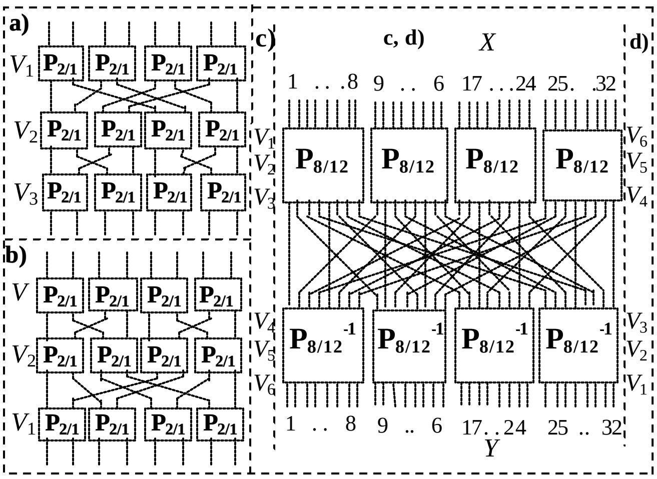


Figure 2. Structure of the boxes $P_{8/12}$ (a), $P^{-1}_{8/12}$ (b), $P_{32/96}$ (c), and $P^{-1}_{32/96}$ (d).

3. The Block Cipher DDP-64

While designing the single key cryptosystem DDP-64 our strategy was oriented to the extensive use of the controlled permutations that are fast and inexpensive while implementing in hardware. Our design criteria were the following:

1. The cryptosystem should be an iterated 64-bit cipher.
2. The cipher should be fast in the case of frequent change of keys, therefore the cryptalgorithm should be able to perform encryption and decryption with simple and fast change of the sequence of the used subkeys.
3. Round transformation of data subblocks should be characterized by high parallelism.
4. Except DDP some additional non-linear operation should be used in the round transformation.

The DDP-64 is a new ten-round iterated block cipher with 64-bit input and 128-bit secret key. The general encryption scheme is described by the following formula:

$$C = T^{(e=0)}(M, K) \text{ and } M = T^{(e=1)}(C, K)$$

where *M* is the plaintext, *C* is the ciphertext ($M, C \in \{0, 1\}^{64}$), *K* is the secret key ($K \in \{0, 1\}^{128}$), T is the transformation function, and $e \in \{0, 1\}$ is a parameter defining encryption (*e* = 0) or decryption (*e* = 1) mode.

The secret key is considered as concatenation of four 32-bit subkeys K_i, *i* = 1, 2, 3, 4: $K = (K_1, K_2, K_3, K_4)$. The cipher uses no preprocessing to transform subkeys. The extended key $Q^{(e)}$ is formed as simple sequence of subkeys K_i taken in respective order.

The general structure of DDP-64 is shown in Figure 3-a. The DDP-64's encryption round (procedure $Crypt^{(e)}$) is presented in Figure 3-b. The encryption procedure can be described as follows.

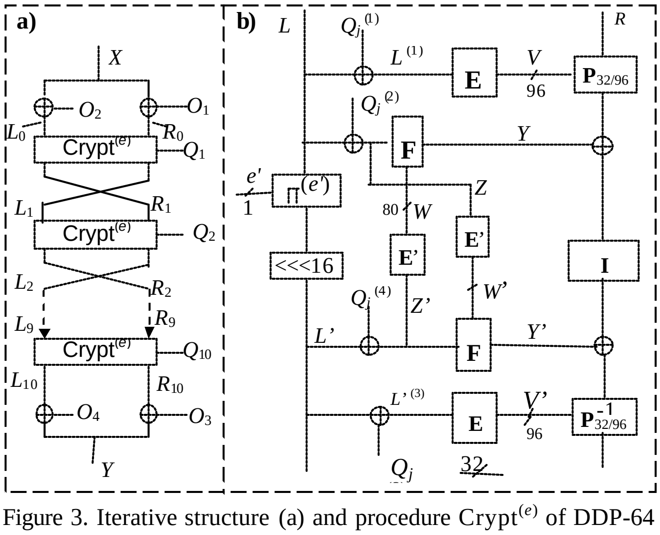


Figure 3. Iterative structure (a) and procedure $Crypt^{(e)}$ of DDP-64 (b).

Encryption begins with initial transformation. Then 10 rounds with procedure *Crypt*$^{(e)}$ followed by final transformation are performed. First data block M is divided into two 32-bit blocks L and R. Then initial transformation is executed which consists in XORing each data subblock with different 32-bit subkeys: $L_0 = L \oplus O_2$ and $R_0 = R \oplus O_1$. Ten consecutive encryption rounds are performed in accordance with the following algorithm:

```
1. For j = 1 to 9 do:
   {
     Execute transformation $(L_j, R_j) = Crypt^{(e)}(L_{j-1}, R_{j-1}, Q^{(e)}_j)$
     Swap the data subblocks: $T = R_j, R_j := L_j, L_j := T$
   }
2. Execute transformation
   $(L_{10}, R_{10}) = Crypt^{(e)}(L_9, R_9, Q^{(e)}_{10})$
```

Procedure *Crypt*$^{(e)}$ represents round encryption function, where $Q^{(e)}_j$ is the 128-bit round key used in the jth encryption round. Encryption finishes with procedure of final transformation: $L¢ = L_{10} \oplus O_4$ and $R¢ = R_{10} \oplus O_3$. The ciphertext block is $C = (L¢, R¢)$.

While designing DDP-64 our intention was to demonstrate the efficiency of DDP as cryptographic primitive. For this aim we have used only DDP except fixed permutations and the XOR operation. Thus, DDP-64 represents a pure DDP-based cryptosystem. One of important problems in the design of the pure DDP-based ciphers is the construction of the DDP-based non-linear operations (sum of outputs of which is highly non-linear Boolean function). The function F used in DDP-64 represents a variant of such operations.

Each round key $Q^{(e)}_j$ consists of four e-dependent round subkeys: $Q^{(e)}_j = (Q^{(1)}, Q^{(2)}, Q^{(3)}, Q^{(4)})^{(e)}_j$. The left data subblock combined with subkeys $Q^{(1)}$ and $Q^{(3)}$ is used to form the controlling vectors V and V' which specify the current modifications of the DDP performed on the right data subblock with boxes $P_{32/96}$ and $P^{-1}_{32/96}$, respectively. The left data subblock combined with subkeys $Q^{(2)}$ and $Q^{(4)}$ is also transformed with two F-boxes implementing special type of DDP. Figure 4 and Table 1 specify round subkeys and their correspondence to the secret key.

Change of the encryption mode is performed as simple swapping subkeys K_i with single-layer box $P^{(e)}_{128/1}$ which is represented by two parallel boxes $P^{(e)}_{64/1}$. The box $P^{(e)}_{64/1}$ is some single-layer CP box in which all elementary switching elements are controlled with the same bit e.

The pairs (K_1, K_3) and (K_2, K_4) are inputs of the corresponding boxes $P^{(e)}_{64/1}$ (see Figure 4). Four 32-bit outputs of two boxes $P^{(e)}_{64/1}$ are the e-dependent subkeys O_j (j = 1, 2, 3, 4). Thus, we have $O_j = K_j$, if $e = 0$, and $O_1 = K_3$, $O_2 = K_4$, $O_3 = K_1$, $O_4 = K_2$, if $e = 1$.

Correct change of the encryption mode for the decryption one is also defined by the respective change of the fixed permutation $\Pi^{(e¢)}$ in procedure *Crypt*$^{(e)}$ presented in Figure 4-b.

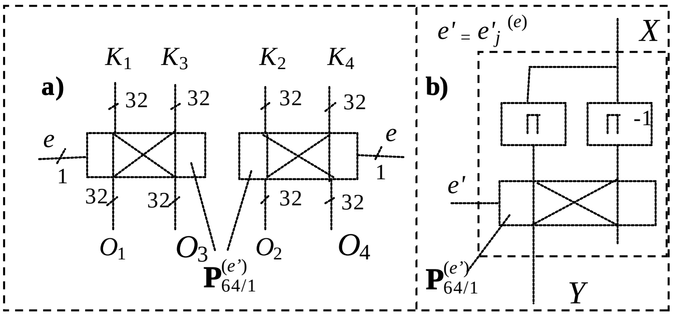


Figure 4. Swapping subkeys (a) and structure of the switchable fixed permutation (b).

The e′-dependent fixed permutation $\Pi^{(e¢)}$ in the left branch of the cryptoscheme is used to prevent homogeneity of the encryption procedure in the case of the key having structure K = (X, X, X, X). For this reason the schedule of the switching bit e(is non-periodic (see Table 1). The structure of the switchable operations $\Pi^{(e¢)}$ is shown in Figure 4-b, where $\Pi^{(0)}$ and $\Pi^{(1)}$ are mutually inverse fixed permutations and $\Pi^{(0)}$ is specified as follows:

(1, 4, 7, 2, 5, 8, 3, 6) (9, 12, 15, 10, 13, 16, 11, 14)
(17, 20, 23, 18, 21, 24, 19, 22)
(25, 28, 31, 26, 29, 32, 27, 30)

Thus, we have $\Pi^{(e¢ \oplus 1)}(Y) = X$, if $Y = \Pi^{(e¢)}(X)$.

Table 1. Specification of the round subkeys and switching bit *e*¢.

j **=**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$Q^{(1)}_j =$	O_3	O_2	O_1	O_4	O_3	O_3	O_4	O_1	O_2	O_3
$Q^{(2)}_j =$	O_4	O_3	O_2	O_1	O_2	O_2	O_1	O_2	O_3	O_4
$Q^{(3)}_j =$	O_1	O_4	O_3	O_2	O_1	O_1	O_2	O_3	O_4	O_1
$Q^{(4)}_j =$	O_2	O_1	O_4	O_3	O_4	O_4	O_3	O_4	O_1	O_2
$e¢^{(e=0)}$	1	0	1	1	0	1	1	1	0	1
$e¢^{(e=1)}$	0	1	0	0	0	1	0	0	1	0

Variable permutations are performed with the CP boxes of the second order $P^{(V)}_{32/96}$ and $(P^{-1}_{32/96})^{(V¢)}$ (see section 2) and F-boxes. Controlling vectors V and V' are formed using the extension box E described below. The F-boxes represent special type of DDP. Construction of the F-boxes provides arbitrary change of the output vector weight. Indeed, depending on L, $Q^{(2)}$, and $Q^{(4)}$ eight of 32 input bits are replaced by bits of the constant C = (10101010) while performing the operation F. Structure of the F-box is presented in Figure 5.

The F-box comprises two three-layer CP boxes $P_{32/48}$ and $P^{-1}_{32/48}$ separated with fixed permutation Π¢ which is described as follows:

(1,33)(2,9)(3,17)(4,25)(5)(6, 13)(7, 21)(8, 34, 29, 40)
(10, 35) (11, 18) (12, 26) (14) (15, 36, 22, 38)
(16, 30) (19, 37) (20, 27) (23) (24, 31) (28, 39) (32)

The 80-bit controlling vector $W = (W_1, W_2, W_3, W_4, W_5)$, where $W_1, W_2, ..., W_5 \in \{0,1\}^{16}$, of the F-box is divided into 48-bit controlling vector (W_1, W_2, W_3) of the CP box $P_{32/48}$ and 32-bit part (W_4, W_5) of the controlling vector of the CP box $P^1_{32/48}$. The 16-bit vector W_6 is formed with the extension box "Ext" (Figure 5-a) using eight of the most significant bits of the output $H = (H_1, H_2, H_3, H_4, H_5)$, where $H_1, H_2,..., H_5 \in \{0, 1\}^8$. The output of the "Ext" box is the vector $W_6 = (H_5, H_5)$.

The 80-bit controlling vector W is formed with the extension box E′ input of which is the vector $Z¢$. Relation between $Z¢$ and W is the following:

$$W_1 = Z'_l;\ W_2 = Z'^{<<<5}_l;\ W_3 = Z'^{<<<10}_l;$$
$$W_4 = Z'_h;\quad W_5 = Z'^{<<<5}_h$$

The vectors W_1, W_2, and W_3 control the 1st, 2nd, and 3d active layers of the $P_{32/48}$-box and the vectors W_4, W_5, and W_6 control the 1st, 2nd, and 3d active layers of the $P^{-1}_{32/48}$-box, correspondingly. The vector D that is the output of $P_{32/48}$ is concatenated with constant C forming the vector (D_1, D_2, D_3, D_4, C) (input of the fixed permutation). The output vector of the fixed permutation Π is (H1, H2, H3, H4, H5), where H5 = (d1, d8, d10, d15, d19, d22, d28, d29). Taking into account the structure of the P32/48-box one can see that superposition P32/48 · Π′ moves arbitrary two bits of each byte Zi of the vector Z = (Z1, Z2, Z_3, Z_4) to H_5 with the same probability. Arbitrary single bit of each byte Z_i moves to H_5 with probability 2^{-2}. Thus, the vector H_5 is composed of eight bits of $Z = L \oplus Q^{(4)}$ which are replaced by 8 bits of C at the output of the F-box. Depending on W different bits of Z are replaced, therefore the oddness of the output vector of the F-box changes arbitrarily.

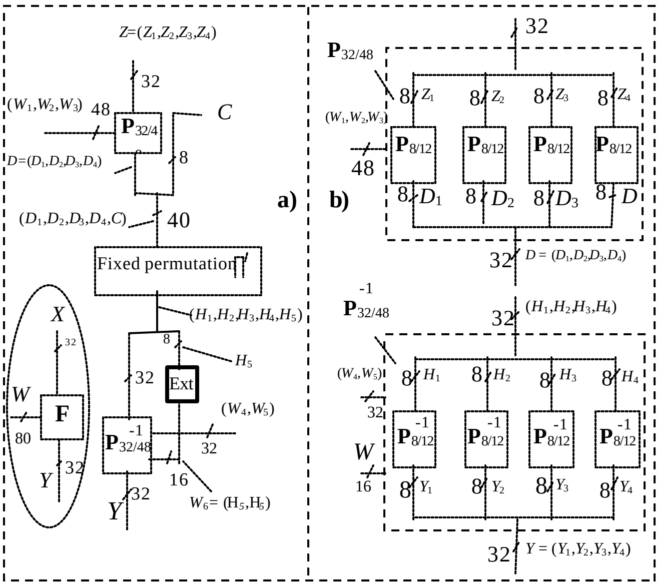


Figure 5: Structure of the F-box (a), the CP boxes $P_{32/48}$, and $P^{-1}_{32/48}$ (b).

Rotation operation "<<<16" performed on the left data subblock is used as permutational involution saving the "symmetric" use of the most significant (L_h) and least significant (L_l) halves of L while performing two F-box operations. The fixed permutation $\Pi^{(e¢)}$ has bee? selected to provide condition $(\Pi^{(e¢)}\ (L))^{<<<16} = (\Pi^{(e¢)}\ (L^{<<<16})$ for $e¢ \in \{0, 1\}$ which is necessary to provide correct decryption. Permutational involution I in the right branch provides each bit at the input of the box $P_{32/96}$ influences 31 bits at the output of the box $P^{-1}_{32/96}$ even in the case $V = V'$ (without I in that case each input bit of $P_{32/96}$ influences only one output bit of $P^{-1}_{32/96}$). The involution I is described with two rotations by eight bits: $Y = \mathrm{I}(X_1, X_2) = (X_1^{<<<8}, X_2^{<<<8})$, where $X_1, X_2 \in \{0, 1\}^{16}$. This permutation improves the resultant VBP corresponding to subsequently performed operations $P_{32/96}$ and $P^{-1}_{32/96}$. Indeed, even in the case $V = V¢$ the superposition $P^{(V)}_{32/96}$ · I · (P-132/96) ($V¢$) forms an effective CP box permutation all modifications of which are different permutational involutions. In general case we have V? V¢, since the data are combined with different subkeys while forming the controlling vectors corresponding to the operations $P_{32/96}$ and $P^{-1}_{32/96.}$ Investigating the role of the fixed permutation between two mutually inverse CP box operations we have performed many statistic experiments which have shown that the use of such permutation significantly improves the properties of the transformation performed with two mutually inverse CP boxes.

4. Peculiarities of DDP-64 and Security Remarks

The DDP-64 is an example of the extensive use of the DDP operations. They are used in three different ways:

1. As DDP that are one of two basic cryptographic primitives.
2. To swap subkeys when changing encryption mode for decryption one.
3. To switch permutation $\Pi^{(e¢)}$ when changing ciphering mode.

Due to high parallelism of the general structure of the both ciphers performing one round takes only about $15t_\oplus$, where $t_\oplus$ is the time delay of the XOR operation. Time delay of ten rounds is about 150 $t_\oplus$. Encryption speed can be estimated as ˜ 0.43 bit/$t_\oplus$. This figure is close to that of the cipher SPECTR-H64 (˜ 0.44 bit/ $t_\oplus$). Analogously to SPECTR-H64 the cipher DDP-64 is fast in the case of frequent change of keys, since they are free of "external" key scheduling. In comparison with SPECTR-H64 the cipher DDP-64 has the following features:

1. The DDP-64 uses the whole secret key in each round.

2. Its round transformation includes permutational involution performed on one of the data subblocks.
3. Its round transformation includes special switchable permutation $\Pi^{(e')}$ performed on one of the data subblocks. Permutation $\Pi^{(e')}$ prevents appearance of the weak keys with the structure $K = (X, X, X, X)$, where $X \in \{0, 1\}^{32}$.
4. Two operations F in DDP-64 introduce significantly more non-linearity than one non-linear operation G used in SPECTR-H64. This makes DDP-64 more secure against linear attacks, the more efficient against DDP-based ciphers is the differential attack though.

We have considered different variants of the differential cryptanalysis. We have obtained that the fewer active bits in the difference the higher the probability of the differential characteristic. This corresponds to the results of the analysis of other DDP-based ciphers. Our best attack against DDP-64 corresponds to two-round difference with one active bit. This difference passes two rounds with probability $p(2) = 1.37 \times 2^{-17}$. Probability of the best two-round characteristic of SPECTR-H64 is $p(2) = 1.15 \times 2^{-13}$ [5]. Minimal number of rounds required to thwart differential attacks is 8 for DDP-64 and 10 for SPECTR-H64.

Our preliminary linear analysis of DDP-64 has shown that they are secure against linear attacks for number of rounds $r \geq 5$. Accordingly to [11] SPECTR-H64 is secure against linear attack for values $r \geq 6$. High degree of the algebraic normal form and the complexity of the Boolean function describing round transformation of the described ciphers prevent the interpolation and high order differential attacks. The used key scheduling is secure against basic related-key attacks. In spite of the simplicity of the key schedule the keys $K' = (X, Y, X, Y)$, $K'' = (X, X, X, X)$, where $X, Y \in \{0, 1\}^{32}$, are not weak, since encryption and decryption require change of the parameter e. It seems to be difficult to calculate a semi-weak key-pair for presented ciphers, if it is possible at all. This shows the other role of the switchable permutation $\Pi^{(e')}$. For example, for SPECTR-H64 which uses no switchable operations for all X the 256-bit key $K = (X, X, X, X, X, X, X, X)$ is weak. Very simple key scheduling is connected with the problem of homogeneity of the encryption procedure which can be used to undertake a slide attack [2]. To prevent such attacks one can efficiently use the switchable operations defining non-periodic scheduling of the switching bit. This idea is illustrated by the cipher DDP-64.

5. ASIC and FPGA Implementations

DDP64 is examined on the hardware implementations by using two different architectures: Full rolling and pipelined for both ASIC and FPGA devices. The used full rolling architecture is shown in Figure 6-a. It is a typical architecture for secret key block cipher implementation. The architecture operates efficiently for both encryption and decryption processes. According to this architecture only one block of plaintext/ ciphertext is transformed at a time. The necessary number of clock cycles to encrypt/ decrypt a data block is equal to the specified number of cipher rounds (10). The key expansion unit produces the appropriate round keys which are stored and loaded in the used RAM blocks. One round of the encryption algorithm is performed by the Data Transformation Round Core. This core is a flexible combinational logic circuit and it is supported by a 64-bit register and 2x64-bit multiplexer. In the first clock cycle, the plaintext/ ciphertext is forced into the data transformation round core. Then in each clock cycle, one round of the cipher is performed and the transformed data are stored into the register. According to Full Rolling architecture a 64-bit data block is completely transformed every 10 clock cycles.

The second proposed architecture, Figure 6-b, is a pipelined architecture. The main characteristics of this are:

1. The pipelined used technique.
2. The usage of a RAM for the round keys storage and loading, which are precomputed.

Pipelined is not possible to be applied in many cryptographic applications. However, DDP64 structure provides the availability to be implemented with pipelining technique. The pipelined architecture offers the benefit of the high-speed performance. The implementation can be applied in applications with hard throughput needs. This goal is achieved by using a number of operating blocks with a final cost to the covered area. The proposed architecture uses 10 basic round blocks and produces a new plaintext/ciphertext block every clock cycle. The synthesis results are shown in Table 2, for both hardware implementation devices (ASIC and FPGA).

Table 2. Implementations synthesis results.

Hardware Device / **Architecture**	**FPGA Technology (Xilinx)**				
	Covered Area			**F (MHz)**	**Rate (Mbps)**
	CLBs	FGs	DFFs		
DDP-64 (FR)	615	1230	207	85	544
DDP-64 (P)	3440	6880	640	95	6.1 Gbps
	ASIC Technology (0.33 um)				
DDP-64 (FR)	2620 sqmil			92	589
DDP-64 (P)	14050 sqmil			101	6.5 Gbps

The above synthesis results for both ASIC and FPGA devices prove that the pipelined architecture implementations of DDP-64 achieve very high speed performance. Especially the achieved throughput is up to 6.1 and 6.5 Gbps for FPGA and ASIC respectively. On the other hand, the full rolling architecture allocates minimized area resources with good data rate. Of course for the full rolling architecture the main goal is the minimized allocated area resources with good achieved throughput. The operation frequency for both proposed architectures and for all the examined hardware modules is very high. It is obvious that according to the applications major demands, area or performance, the designer can use the full rolling or the pipelined architecture respectively. The type of communication and the characteristics of the application itself, will determine the use of the FPGA or the ASIC as the integration device, for the implementation of this new cipher. DDP-64 architecture's simplicity in addition to the offered high-level security strength, make the cipher's hardware integration very flexible for communications networks. In order to evaluate the very good performance, in Hardware terms, we compare the proposed cipher with the encryption algorithms that are widely used in today's communications protocols. Especially in Figure 7, the DDP-64 performance is compared with the best hardware implementations of other ciphers, used in wireless communications protocols.

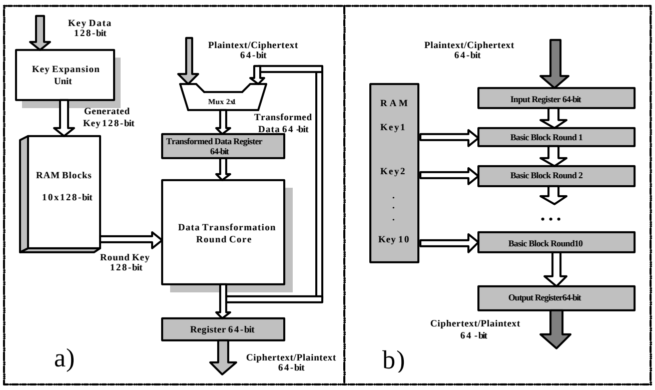


Figure 6. Full rolling architecture (a) and pipelined architecture (b).

In IEEE 802.11 first versions (a to d), the Wireless Application Protocol (WEP) is widely used to ensure privacy in the transmission channel. Especially WEP is based on RC4. The latest working group (802.11i) adopts the AES as the block cipher of this IEEE protocol. Bluetooth security is based on SAFER+ cipher. UMTS uses both AES and SAFER + in order to achieve security due to the external attacks over the transmitted data. Finally the Wireless Transport Layer Security (WTLS) ensure encryption in both Wireless Application Protocol (WAP) and Open Mobile Alliance (OMA). DES, IDEA and RC5 are the alternative ciphers that can be used for bulk encryption in WTLS. For the FPGA implementation approach as it is shown in Figure 7-a, the Full Rolling (FR) architecture of the proposed DDP-64 has minimized area resources compared with FPGA approaches of AES [15], SAFER + [10], IDEA [3], and DES [9]. The pipeline architecture (P) of DDP-64, as it was expected needs more resources than FR. The great advantage of the pipelined architecture, as it is proven from Figure 7, is the very high speed performance, compared with the achieved throughput of the other conventional ciphers [3, 8, 9, 10] and a little bit less than AES [15]. In addition the proposed architectures are compared with other ASIC implementations [21- 24] in Figure 7-b. For the ASIC approach, a covered area comparison is not efficient because of the variety of the used area units (mm2, sqmil, gates, and transistors) of the reported area of the conventional architectures [21-24]. Both DDP-64 implementations have higher performance compared with AES [22], IDEA [24], and RC5 [21]. The pipelined implementation of DES [23] is quite better compared with our cipher implementation.

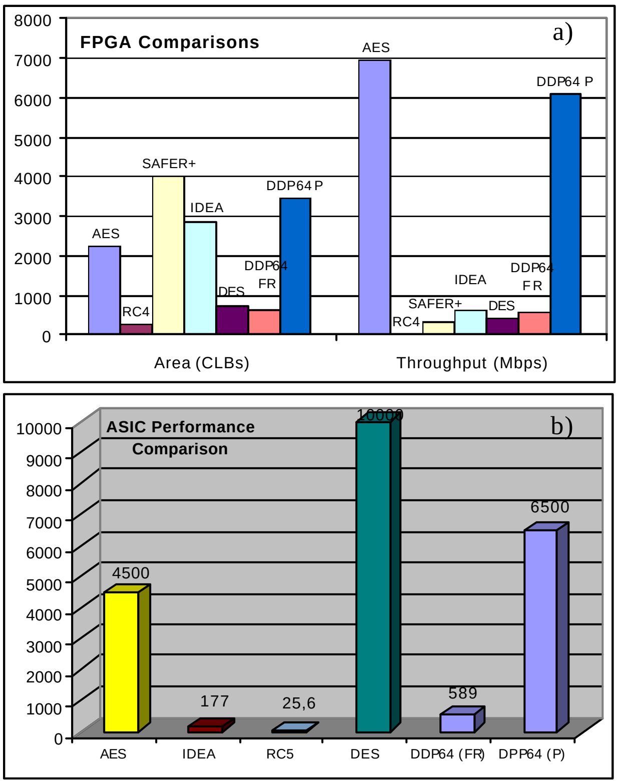


Figure 7. Implementations comparisons: FPGA devices (a) and ASIC devices (b).

6. Conclusions and Outlook

In this paper we propose a pure DDP-based cipher that uses only bit permutations and the XOR operations. Security analysis has shown that it is secure against known attacks. The DDP-64 evidently shows efficiency of DDP as cryptographic primitive. Due to high parallelism of computations in one round and the use of switchable operations one can use very simple key scheduling that makes hardware implementation cheaper and faster in the case of frequent change of keys. The DDP-64 achieves high-speed performance in FPGA devices and especially for ASIC approaches. The implementation cost and the performance of the proposed cipher are compared with the security layers of the most widely used wireless protocols: IEEE

802.11, Bluetooth, WAP, OMA, and UMTS. These comparisons present the advantages of the proposed cipher in term of area resources, operating frequency, and throughput.

References

[1] Benes V. E., *Mathematical Theory of Connecting Networks and Telephone Traffic*, Academic Press, New York, 1965.

[2] Biryukov D. and Wagner D., "Slide Attacks," *in Proceedings of the 6th International Workshop Fast Software Encryption*, LNCS, Springer-Verlag, vol. 1636, pp. 245-259, 1999.

[3] Cheung O. Y. H., Tsoi K. H., Leong P. H. W., and Leong M. P., "Tradeoffs in Parallel and Serial Implementations of the International Data Encryption Algorithm," *in Proceedings of CHES'2001*, LNCS, Springer-Verlag, vol. 2162, pp. 333-37, 2001.

[4] Clos C., "A Study of Nonblocking Switching Networks," *Bell System Technical Journal*, vol.32, pp.406-424, 1953.

[5] Goots N., Izotov B., Moldovyan A. A., and Moldovyan N. A., "Fast Ciphers for Cheap Hardware: Differential Analysis of SPECTR-H64," *in Proceedings of the 1st International Workshop, Methods, Models, and Architectures for Network Security*, LNCS, Springer-Verlag, vol. 2776, pp. 449-452, 2003.

[6] Goots N., Izotov B., Moldovyan A., and Moldovyan N., *Modern Cryptography: Protect Your Data with Fast Block Ciphers*, Wayne, A-LIST Publishing, 2003.

[7] Goots N., Moldovyan A. A., and Moldovyan N. A., "Fast Encryption Algorithm SPECTR-H64," *in Proceedings of the 1st International Workshop, Methods, Models, and Architectures for Network Security*, LNCS, Springer-Verlag, vol. 2052, pp. 275-286, 2001.

[8] Hamalainen P., Hannikainen M., Hamalainen T., and Saarinen J., "Hardware Implementation of the Improved WEP and RC4 Encryption Algorithms for Wireless Terminals," *in Proceedings of the European Signal Processing Conference (EUSIPCO'2000)*, Finland, September 2000.

[9] Kaps J. and Paar C., "Fast DES Implementations for FPGAs and its Application to a Universal Key-Search Machine," *in Proceedings of the 5th Annual Workshop on Selected Areas in Cryptography*, Canada, August 1998.

[10] Kitsos P., Sklavos P., Papadomanolakis K., and Koufopavlou O., "Hardware Implementation of the Bluetooth Security," *IEEE Pervasive Computing, Mobile and Ubiquitous Systems*, vol. 2, no. 1, January-March 2003.

[11] Ko Y. , Hong D., Hong S., Lee S., and Lim J., "Linear Cryptanalysis on SPECTR-H64 with Higher Order Differential Property," *in Proceedings of the 1st International Workshop, Methods, Models, and Architectures for Network Security*, LNCS, vol. 2776, pp. 298-307, 2003.

[12] Kwan M., "The Design of the ICE Encryption Algorithm," *in Proceedings of the 4th International Workshop Fast Software Encryption (FSE'97)*, LNCS, Springer-Verlag, vol. 1267, pp. 69-82, 1997.

[13] Lee C., Hong D., Lee S., Yang H., and Lim J., "A Chosen Plaintext Linear Attack on Block Cipher CIKS-1," *LNCS, Springer-Verlag*, vol. 2513, pp. 456-468, 2002.

[14] Maslovsky V. M., Moldovyan A. A., and Moldovyan N. A., "A Method of the Block Encryption of Discrete Data," *Russian patent # 2140710*. Bull, no 30, 1999.

[15] McLoone M. and McCanny J. V., "High Performance Single-Chip FPGA Rijndael Algorithm Implementation," *in Proceedings of CHES'2001*, LNCS 2162, Springer-Verlag, pp. 65-76, 2001.

[16] Moldovyan A. A., "Fast Block Ciphers Based on Controlled Permutations," *Computer Science Journal of Moldova*, vol. 8, no. 3, pp. 270-283, 2000.

[17] Moldovyan A. A. and Moldovyan N. A., "A Cipher Based on Data-Dependent Permutations," *Journal of Cryptology*, vol. 15, no. 1, pp.61-72, 2002.

[18] Moldovyan A. A. and Moldovyan N. A., "A Method of the Cryptographical Transformation of Binary Data Blocks," *Russian patent # 2141729*. Bull, no 32, 1999.

[19] Portz M., "A Generallized Description of DES-Based and Benes-Based Permutation Generators," *Advances in Cryptology (AUSCRYPT'92)*, LNCS, Springer-Verlag, vol.718, pp. 397-409, 1992.

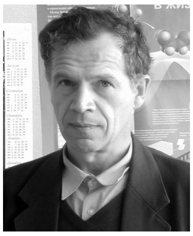
[20] Rompay V. B., Knudsen L. R., and Rijmen V., "Differential Cryptanalysis of the ICE Encryption Algorithm," *in Proceedings of the 5th International Workshop Fast Software Encryption (FSE'98)*, LNCS, Springer-Verlag, vol. 1372, pp. 270-283, 1998.

[21] Schubert A. and Anheier W., "Efficient VLSI Implementation of Modern Symmetric Block Ciphers," *in Proceedings of ICECS'99*, Cyprus, 1999.

[22] Weeks B., Bean M., Rozylowicz T., and Ficke C., "Hardware Performance Simulations of Round 2 Advanced Encryption Standard Algorithms," *in Proceedings of the 3rd Advanced Encryption Standard (AES) Candidate Conference*, New York, USA, April 13-14, 2000.

[23] Wilcox D. C., Pierson L. G., Robertson P. J., Witzke E. L., and Gass K., "A DES ASIC Suitable for Network Encryption at 10 Gbps and Beyoned," *in Proceedings of CHES'99*, LNCS, Springer-Verlag, vol. 1717, pp. 37-48, 1999.

[24] Zimmermann R., Curiger A., Bonnenberg H., Kaeslin H., Felber N., and Fichtner W., "A 177 Mb/s VLSI Implementation of the International Data Encryption Algorithm," *IEEE Journal of Solid State Circuits*, vol. 29, no. 3, March 1994.



Nikolay Moldovyan is an honoured inventor of Russian Federation in 2002, a chief researcher with the Specialized Center of Program Systems "SPECTR", and a professor with the Saint Petersburg Electrical Engineering University. He received his Diploma and PhD from Academy of Sciences of Moldova, 1981. His research interests include computer security, cryptography, and currently developed concept of the variable transformations as a new direction in applied cryptography. He is a member of the IACR.

Nicolas Sklavos is a PhD researcher with the Electrical and Computer Engineering Department of the University of Patras, Greece. His interests include computer security, new encryption algorithms design, wireless communications, and reconfigurable computing. He holds an award for his PhD research on "VLSI Designs of Wireless Communications Security Systems" from IFIP VLSI SOC 2003. He is a referee of International Journals and Conferences. He is a member of the IEEE, the Technical Chamber of Greece, and the Greek Electrical Engineering Society. He has authored or coauthored up to 50 scientific articles in the areas of his research.

Odysseas Koufopavlou received the Diploma of electrical engineering in 1983 and the PhD degree in electrical engineering in 1990, both from University of Patras, Greece. From 1990 to 1994 he was at the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. He is currently an associate professor with the Department of Electrical and Computer Engineering, University of Patras. His research interests include VLSI, low power design, VLSI crypto systems, and high performance communication subsystems architecture and implementation. He has published more than 100 technical papers and received patents and inventions in these areas.