On Cipher Design Based on Switchable Controlled Operations

Nikolay A. Moldovyan
Specialized Center of Program Systems "SPECTR"
Kantemirovskaya, 10, St-Petersburg 197342, Russia (Email: nmold@cobra.ru)
(Received Jan. 23, 2007; revised and accepted Oct. 23, 2007)

Abstract

This paper concerns the problem of reducing the implementation cost of the switchable data-dependent operations (SDDOs) that are a new cryptographic primitive oriented to the design of fast ciphers suitable to applications in constrained environments. The SDDOs are performed with the controlled substitution-permutation networks that transforms binary vectors depending on some controlling data subblock. Three new desings of the SDDOs are proposed. The first one is characterized by dividing the controlling data subblock into two subvectors that are extended in correspondence with mirror symmetry. Reversing the operation mode is defind by swapping the subvectors. The second one is characterized by using the inversion of the controlling bits as the switching mechanism. The third one is characterized by using two mutual inverse controlled operational boxes and provides construction of the SDDOs of different orders. The last design supplements the known particular constructions of SDDOs having symmetric topology. A new SDDO-based block cipher is presented and estimated.

Keywords: Data-dependent operations, fast encryption, hardware-oriented cipher, switchable operations, variable operations

1 Introduction

Encryption is an efficient way to provide secure communications. Successful deployment of ad hoc and sensor networks is connected with the problem of embedding security mechanisms in constrained environments. Therefore designing ciphers suitable to cheap hardware implementation represents practical importance. Recently [7, 13, 14] the data-dependent (DD) permutations (DDPs) have been proposed to provide high performance while constrained hardware resources are used. Security analysis of the DDP-based ciphers [5] demonstrates efficiency of DDP. A new class of the DD operations (DDOs) is proposed in [9] for designing fast harwdare-suitable ciphers. Different types of the DDOs can be implemented using the controlled operations (Definition 1) and defining dependence of the controlling vector on the transformed data.

Definition 1. *Let* $\{\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_{2^m}\}$ *be some set of the single-type operations defined by formula* $Y = \mathbf{F}_i(X_1, X_2, ..., X_q)$*, where* $i = 1, 2, ..., 2^m$ *and* $X_1, X_2, ..., X_q$ *are input n-dimensional binary vectors (operands) and* Y *is the output n-dimensional binary vector. Then the* V*-dependent operation* $\mathbf{F}^{(V)}$ *defined by formula* $Y = \mathbf{F}^{(V)}(X_1, X_2, ..., X_q) = \mathbf{F}_V(X_1, X_2, ..., X_q)$*, where* V *is the m-dimensional controlling vector, we call the controlled q-place operation. The operations* $\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_{2^m}$ *are called modifications of the controlled operation* $\mathbf{F}^{(V)}$*.*

Many network applications of the encryption require development of the ciphers that are fast in the case of frequent change of keys. Such ciphers should use no time consuming key preprocessing, i.e. they should use very simple key scheduling. An attempt to simplify the key scheduling is the use of the DD transformation of the subkeys, which is called internal key scheduling [7]. To implement data-driven processing of the subkeys different variants of the so called controlled operations (COs) are suitable [10]. Switchable DDOs (SDDOs) [8] have been also proposed as a primitive suitable to designing efficient ciphers with simple key scheduling. Implementation results of the SDDO-based ciphers shows they provide high performance while implemented in cheap hardware [12]. The SDDOs are performed with switchable controlled operations (SCO) defined below.

Definition 2. *Let* $\{\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_{2^m}\}$ *be the set of the modifications of the controlled operation* $\mathbf{F}^{(V)}$*. The operation* $(\mathbf{F}^{-1})^{(V)}$ *containing modifications* $\mathbf{F}_1^{-1}, \mathbf{F}_2^{-1}, ..., \mathbf{F}_{2^m}^{-1}$ *is called inverse of* $\mathbf{F}^{(V)}$*, if* $\mathbf{F}_V^{-1}$ *and* $\mathbf{F}_V$ *for all* V *are mutual inverses.*

Definition 3. *Let* $\mathbf{F}^{(e)}$*, where* $e \in \{0, 1\}$*, be some e-dependent operation containing two modifications* $\mathbf{F}^{(0)} = \mathbf{F}_1$ *and* $\mathbf{F}^{(1)} = \mathbf{F}_2$*, where* $\mathbf{F}_2 = \mathbf{F}_1^{-1}$*. Then the operation* $\mathbf{F}^{(e)}$ *is called switchable.*

Definition 4. *Let two modifications of the switchable operation* $\mathbf{F}^{(e)}$ *be mutual inverses* $\mathbf{F}^{(0)} = \mathbf{F}^{(V)}$ *and* $\mathbf{F}^{(1)} =$

$(\mathbf{F}^{-1})^{(V)}$. *Then* $\mathbf{F}^{(e)}$ *is called switchable controlled operation* $\mathbf{F}^{(V,e)}$.

In this paper we consider new designs of the SDDOs providing cheaper implementation. We propose also a new fast SDDO-based cipher suitable to implementation while constrained hardware resources. In Section 2 we consider three designs of the SDDOs based on controlled substitution-permutation networks (CSPNs). The feature of the first design is the symmetric distribution of the controlling bits over controlled elements (CEs) of some CSPN having symmetric topology. In the second design the inversion of the controlling bits is used to reverse the transformation performed with SDDOs. The third design is oriented to construction of the SDDOs of different orders. Section 3 introduces a new SDDO-based cipher using extremely simple key scheduling. Section 4 presents discussion and conclusion.

2 New Designs of Switchable Controlled Operations

Throughout the paper we use the following notations:

$\diamond$ $\{0,1\}^n$ denotes the set of all n-bit binary vectors $X=(x_1,...,x_n)$, where $x_i \in \{0,1\}$, $i=1,...,n$;

$\diamond$ $X \oplus Y$ denotes the bit-wise XOR operation performed on X and Y: $X, Y \in \{0,1\}^n$;

$\diamond$ $(A,B,...,Z)$ denotes the concatenation of the binary vectors A, B,..., Z;

$\diamond$ "$+_n$" ("$-_n$") denotes the addition (subtraction) modulo 2^n; $[s/2]$ denotes the integer part of $s/2$.

$\diamond$ $Y = X^{\lll k}$ denote the to-left rotation of the word X by k bits, where $\forall i \in \{1,...,n-k\}$ we have $y_i = x_{i+k}$ and $\forall i \in \{n-k+1,...,n\}$ we have $y_i = x_{i+k-n}$.

$\diamond$ The binary vector $(f(0,0,0),\ f(0,0,1),...,\ f(1,1,1))$ denotes the truth table (TT) of Boolean function (BF) $f=(x_1,x_2,x_3)$.

2.1 Networks Based on Minimum Size Controlled Elements

The known controlled operations are implemented as uniform CSPNs constructed using the minimum size CEs as standard building blocks. Figure 1a shows general topology of CSPN. Let CSPN with n-bit input and n-bit output be controlled with m-bit vector V. Then we shall denote such CSPN as controlled operation (CO) $\mathbf{F}_{n;m}$. Selecting a set of the fixed permutations connecting active layers we define some particular topology of CO. Each active layer represents $n/2$ parallel CEs. In present paper dotted lines corresponding to CO boxes indicate the controlling bits. A minimum size CE (Figure 1b) is denoted as the $\mathbf{F}_{2;1}$ box. It transforms the two-bit input vector (x_1,x_2) into the two-bit output (y_1,y_2) depending on some controlling bit v. The $\mathbf{F}_{2;1}$ element can be described with a pair of BFs in three variables (Figure 1b): $y_1=f_1(x_1,x_2,v)$ and $y_2=f_2(x_1,x_2,v)$.

A CE can be also represented as a pair of the 2×2 substitutions (elementary S-boxes) selected depending on bit v (Figure 1c). Such subsitutions are denoted as $\mathbf{F}_{2;1}^{(0)}$ and $\mathbf{F}_{2;1}^{(1)}$ and CE implements the transformation $(y_1,y_2)=\mathbf{F}_{2;1}^{(v)}(x_1,x_2)$. There exist only 24 different S-boxes of the 2×2 type (Figure 2), all of them being liner transformations. Different pairs of the 2×2 boxes define different variants of CE. An elementary S-box can be described as a pair of BFs in two variables x_1 and x_2. If the substitution $\mathbf{F}_{2;1}^{(0)}$ is described with two BFs $y_1'=f_1'(x_1,x_2)$ and $y_2'=f_2'(x_1,x_2)$, $\mathbf{F}_{2;1}^{(1)}$ is described with BFs $y_1''=f_1''(x_1,x_2)$ and $y_2''=f_1''(x_1,x_2)$, then CE $\mathbf{F}_{2;1}^{(v)}$ is described with two BFs in three variables x_1, x_2, and v:

$$\begin{aligned} y_1 &= (v\oplus 1)f_1'(x_1,x_2)\oplus v f_1''(x_1,x_2),\\ y_2 &= (v\oplus 1)f_2'(x_1,x_2)\oplus v f_2''(x_1,x_2). \end{aligned}$$

One of important cases of CEs is the switching element $\mathbf{P}_{2;1}$ that performs controlled swapping of two input bits x_1 and x_2. The output bits are $y_1=x_2$ and $y_2=x_1$, if $v=1$, or $y_1=x_1$ and $y_2=x_2$, if $v=0$. The $\mathbf{P}_{2;1}$ element is described with non-linear BFs $y_1=vx_1\oplus vx_2\oplus x_1$ and $y_2=vx_1\oplus vx_2\oplus x_2$. It can be alternatively described as the (a,e) pair of the 2×2 substitutions. The $\mathbf{P}_{2;1}$ element is a linear cryptographic primitive, since the sum of output is linear BF: $y_1\oplus y_2=x_1\oplus x_2$. Networks based on the $\mathbf{P}_{2;1}$ elements are used to perform DDPs that are also linear primitive. For block cipher synthesis the non-linear DDOs are more interesting. They can be designed using the non-linear CEs. Among non-linear CEs, only 24 of them are involutions (see Table 1, where BFs f_1 and f_2 are described with TTs). For elementary controlled involutions $\mathbf{F}_{2;1}^{(v)}$ we have:

$$(y_1,y_2)=\mathbf{F}_{2;1}^{(v)}(x_1,x_2) \Leftrightarrow (x_1,x_2)=\mathbf{F}_{2;1}^{(v)}(y_1,y_2),$$

where $v\in\{0,1\}$.

The $\mathbf{F}_{n;m}$ box can be represented as a superposition of the operations (see Figure 3a) performed on binary vectors:

$$\mathbf{F}_{n;m}=\mathbf{L}^{(V_1)}\circ\pi_1\circ\mathbf{L}^{(V_2)}\circ\pi_2\circ...\circ\pi_{s-1}\circ\mathbf{L}^{(V_s)},$$

where π_j, $j=1,2,...,s-1$, are fixed permutations, $V=(V_1,V_2,...,V_s)$, V_j is the component of V, which controls the jth active layer, and $s=2m/n$ is the number of active layers $\mathbf{L}^{(V_j)}$. Design of the CO boxes with required properties consists in selecting respective topology and CEs. Controlled permutation (CP) boxes of different orders can be constructed using the switching element $\mathbf{P}_{2;1}$.

Definition 5. [7]. *The CP box* $\mathbf{P}_{n;m}$ *is called a CP box of the order* h $(1\le h\le n)$, *if for arbitrary index set* $i_1,i_2,...,i_h$ *and arbitrary index set* $j_1,j_2,...,j_h$ $(i_\alpha\ne i_\beta$ *and* $j_\alpha\ne j_\beta$ *for* $\alpha\ne\beta)$ *there is at least one vector* V *which specifies a permutation* $\mathbf{P}_V$ *moving* x_{i_α} *to* y_{j_α} *for all* $\alpha=1,2,...,h$.

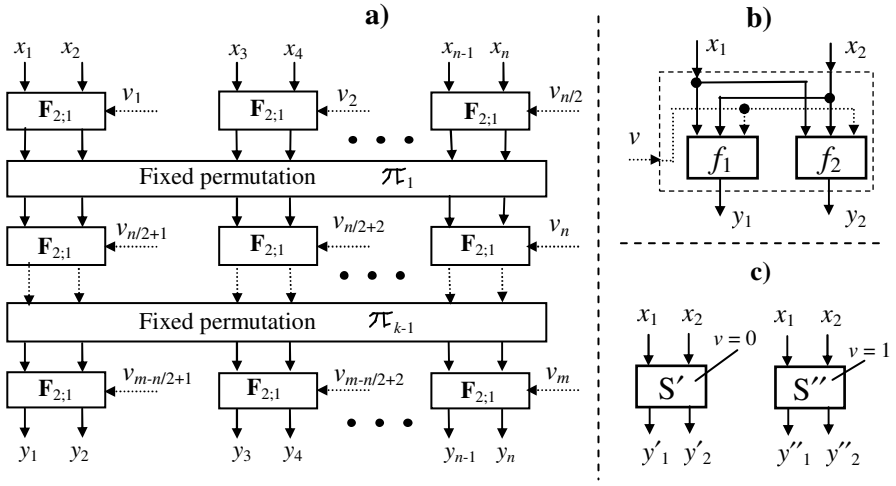

Figure 1: General structure of the $\mathbf{F}_{n;m}$ boxes (a) and representation of the $\mathbf{F}_{2;1}$ element as a pair of BFs in three variables (b) and as switchable 2×2 substitution (c)

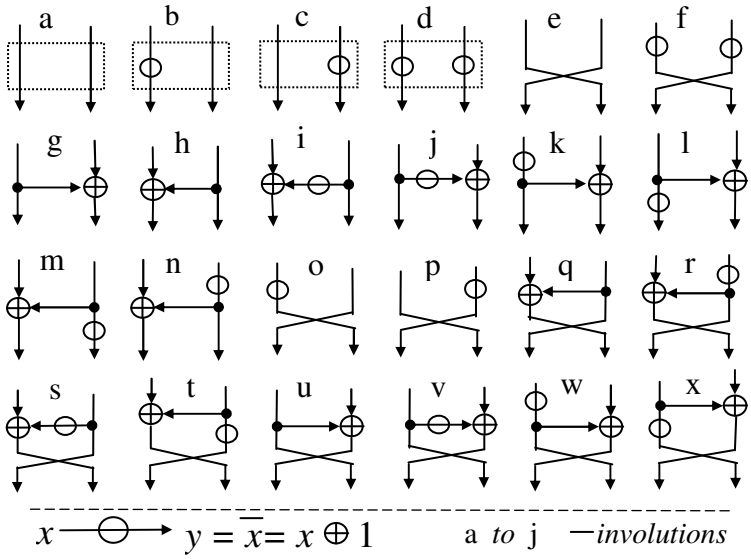

Figure 2: All existing 2×2 substitutions represented as elementary transformations

The CP boxes of different orders are described in [10]. For $n = 2^k$, where $k \geq 1$ is a natural number, the implementation of the box $\mathbf{P}_{n;m}$ of the first, second, ..., $(n/4)$th, and nth order require the use of $\log_2 n$, $\log_2 n + 1$,...,$2\log_2 n - 2$, and $2\log_2 n - 1$ active layers, respectively (the last figure corresponds also to implementing the CP box of the order $n/2$). The notion of the order be extended as follows:

Definition 6. *A CO box $\mathbf{F}_{n;m}$ is called the CO box of the order h $(1 \leq h \leq n)$, if replacement of all CEs of the CO box by the switching elements produces the CP box of the order h.*

Suppose a CO box is constructed using CEs that are involutions. Then one can easy construct the layered box $\mathbf{F}^{-1}_{n;m}$ which is inverse of $\mathbf{F}_{n;m}$-box:

$$\mathbf{F}^{-1}_{n;m} = \mathbf{L}^{(V_s)} \circ \pi^{-1}_{s-1} \circ \mathbf{L}^{(V_{s-1})} \circ \pi^{-1}_{s-2} \circ ... \circ \pi^{-1}_{1} \circ \mathbf{L}^{(V_1)}.$$

In accordance with the structure of the CO boxes $\mathbf{F}_{n;m}$ and $\mathbf{F}^{-1}_{n;m}$ we shall assume that in the direct CO boxes the $\mathbf{F}_{2;1}$ elements are consecutively numbered from left to right and *from top to bottom* and in the inverse CO boxes the CEs are numbered from left to right and *from bottom to top*. Thus, for all $i \in \{1, 2, ..., m\}$ the ith bit of the controlling vector V controls the ith box $\mathbf{F}_{2;1}$ in both the $\mathbf{F}_{n;m}$ and $\mathbf{F}^{-1}_{n;m}$ boxes. For $j = 1, 2, ..., s$ the V_j component of V controls the j-th active layer in the box $\mathbf{F}_{n;m}$ and the $(s-j+1)$-th layer in $\mathbf{F}^{-1}_{n;m}$. For example, the mutual

Table 1: Full set of the $\mathbf{F}_{2;1}$ nonlinear boxes that are involutions

#	f_1 / f_2	(S′,S″) pair	#	f_1 / f_2	(S′,S″) pair	#	f_1 / f_2	(S′,S″) pair
1	00011011 / 00101101	(g,e)	9	00011011 / 10000111	(j,e)	17	00011110 / 00111001	(g,h)
2	00011110 / 10010011	(j,h)	10	00100111 / 00011110	(e,g)	18	00100111 / 01001011	(e,j)
3	00101101 / 00110110	(h,g)	11	00101101 / 01100011	(h,j)	19	00110110 / 00011011	(e,h)
4	00111001 / 00100111	(h,e)	12	01001011 / 00111001	(g,i)	20	01001011 / 10010011	(j,i)
5	01001110 / 01111000	(g,f)	13	01001110 / 11010010	(j,f)	21	01100011 / 00011011	(e,i)
6	01101100 / 01110010	(h,f)	14	10000111 / 00110110	(i,g)	22	10000111 / 01100011	(i,j)
7	10001101 / 10110100	(f,g)	15	10001101 / 11100001	(f,j)	23	10010011 / 00100111	(i,e)
8	10011100 / 10110001	(f,h)	16	11000110 / 01110010	(i,f)	24	11001001 / 10110001	(f,i)

inverses $\mathbf{F}_{8;12}$ and $\mathbf{F}_{8;12}^{-1}$ are shown in Figure 3c,d. Note that the box $\mathbf{F}_{n;m}^{-1}$ is of the order h, if the box $\mathbf{F}_{n;m}$ is of the order h, and the implementation of the CO boxes $\mathbf{F}_{n;m}$ and $\mathbf{F}_{n;m}^{-1}$ of the given order requires the use of the same minimum number of the active layers. Below we will use the following definition ($\mathbf{L}_j$ denotes the jth active layer):

Definition 7. *Suppose a CO box $\mathbf{F}_{n;m}$ contains s active layers. The CO box $\mathbf{F}_{n;m}$ is called symmetric if $\forall j = 1, ..., s-1$ the following relation holds: $\mathbf{L}_j = \mathbf{L}_{s-j+1}^{-1}$ (or $\mathbf{L}_j = \mathbf{L}_{s-j+1}$, if $\mathbf{L}_j$ is involution) and $\pi_j = \pi_{s-j}^{-1}$.*

2.2 Switchable Data-dependent Operations

Earlier [8] the symmetric SCOs $\mathbf{F}_{32;96}^{(V,e)}$ and $\mathbf{F}_{64;192}^{(V,e)}$ have been considered as new primitive suitable to the design of the 64- and 128-bit ciphers, correspondingly. Such SCOs are based on i) the use of the six-layer CSPNs $\mathbf{F}_{32;96}^{(V)}$ (Figure 4a) and $\mathbf{F}_{64;192}^{(V)}$ (Figure 4c) having mirror-symmetry topology and ii) swapping the V_j and V_{7-j} components of the controlling vector V for $j = 1, ..., 6$. Due to symmetric structure of the CO boxes the modifications $\mathbf{F}^{(V)}$, where $V = (V_1, V_2, ...V_6)$, and $\mathbf{F}^{(V')}$, where $V' = (V_6, V_5..., V_1)$ are mutually inverses. In the $\mathbf{F}_{32;96}^{(V,e)}$ box swapping the V_j and V_{7-j} components is performed with very simple transposition box $\mathbf{P}_{96;1}^{(e)}$ that is implemented as some single layer CP box consisting of three parallel single-layer boxes $\mathbf{P}_{16\times2;1}^{(e)}$ (Figure 5a). Input of each $\mathbf{P}_{16\times2;1}^{(e)}$-box is divided into 16-bit left and 16-bit right inputs. The box $\mathbf{P}_{16\times2;1}^{(e)}$ represents 16 parallel $\mathbf{P}_{2;1}^{(e)}$-boxes controlled with the same bit e. The right (left) input (output) of 16 parallel boxes $\mathbf{P}_{2;1}^{(e)}$ compose the right (left) 16-bit input (output) of the box $\mathbf{P}_{16\times2;1}^{(e)}$. Thus, each of three boxes $\mathbf{P}_{16\times2;1}^{(e)}$ performs the e-dependent swapping of the respective pair of the 16-bit components of the controlling vector V.

For example, $\mathbf{P}_{16\times2;1}^{(0)}(V_1, V_6) = (V_1, V_6)$ and $\mathbf{P}_{16\times2;1}^{(1)}(V_1, V_6) = (V_6, V_1)$. If the input vector of the box $\mathbf{P}_{96;1}^{(e)}$ is $(V_1, V_2, ...V_6)$, then at the output of $\mathbf{P}_{96;1}^{(e)}$ we have $V' = (V_1, V_2, ..., V_6)$ (if $e = 0$) or $V' = (V_6, V_5, ..., V_1)$ (if $e = 1$). Structure of the SCO box $\mathbf{F}_{32;96}^{(V,e)}$ is shown in Figure 5b.

The $\mathbf{F}_{64;192}^{(V,e)}$ box can be constructed with the use of transposition box $\mathbf{P}_{192;1}^{(e)}$ that represents three parallel single-layer boxes $\mathbf{P}_{32\times2;1}^{(e)}$ (Figure 5c). Each $\mathbf{P}_{32\times2;1}^{(e)}$-box is a set of 32 parallel $\mathbf{P}_{2;1}^{(e)}$-boxes all of which are controlled with the bit e. The structure of the $\mathbf{F}_{64;192}^{(V,e)}$-box is shown in Figure 5d.

The SCO design considered above imposes no restrictions on the distribution of the controlling bits, however in the DDO-based ciphers usually the m-bit controlling vector V depends on some n-bit controlling data subblock L. The V vector is formed as output of the extension operation $\mathbf{E}$ performed on L. In the case $s = 6$ we have $m = 3n$ and using the properly designed extension box $\mathbf{E}$ it is possible to swap the vectors V_j and V_{7-j} for $j = 1, 2, 3$ performing the transposition of the two halves of the controlling data subblock. This provides possibility to reduce significantly the implementation cost of the SCO boxes $\mathbf{F}_{32;96}^{(e)}$ and $\mathbf{F}_{64;192}^{(e)}$. We propose the following design.

Let L_1 and L_2 be two halves of $L = (l_1, l_2, ..., l_n)$, i. e. $L = (L_1, L_2)$. The L_1 and L_2 components are extended using two symmetric extension boxes $\mathbf{E}_1$ and $\mathbf{E}_2$. The outputs of $\mathbf{E}_1$ and $\mathbf{E}_2$ are $V' = (V_1, V_2, V_3)$ and $V'' = (V_4, V_5, V_6)$, respectively. Thus, the boxes $\mathbf{E}_1$ and $\mathbf{E}_2$ represent a single extention box $\mathbf{E}$ with symmetric structure, which produces the controlling vector $V = (V', V'')$ corresponding to symmetric distribution of the bits of the $L_1 = (l_1, l_2, ..., l_{n/2})$ and $L_2 = (l_{n/2+1}, l_{n/2+2}, ..., l_n)$ components. In this case swapping the L_1 and L_2 components defines swapping components V_j and V_{7-j} for $j = 1, 2, 3$. For the $\mathbf{F}_{32;96}^{(L,e)}$ operation we propose the $\mathbf{E}$ box having the following structure:

$$V_1 = L_1; \quad V_2 = L_1^{\lll 6}; \quad V_2 = L_1^{\lll 12};$$

$$V_4 = L_2^{\lll 12}; V_5 = L_2^{\lll 6}; \quad V_6 = L_2.$$

For the $\mathbf{F}_{64;192}^{(L,e)}$ operation we propose the $\mathbf{E}$ box described as follows:

$$V_1 = L_1; \quad V_2 = L_1^{\lll 14}; \quad V_3 = L_1^{\lll 28};$$

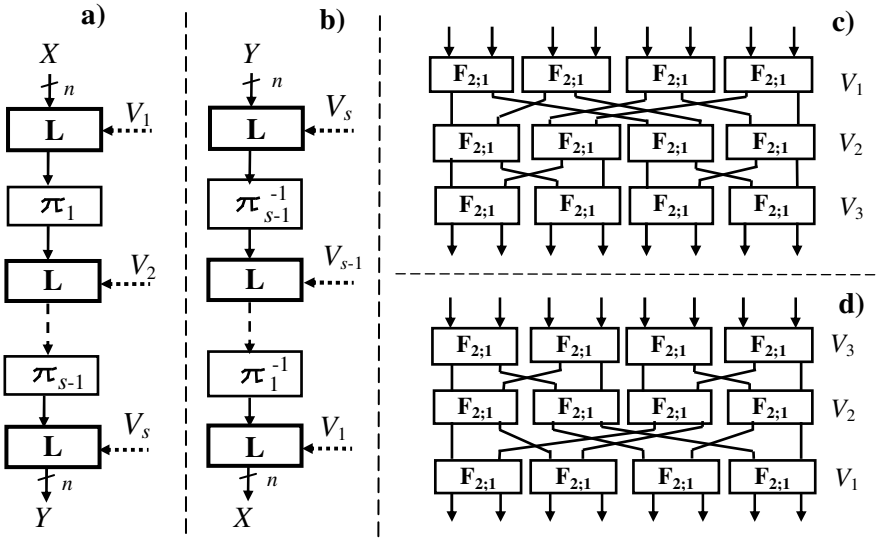

Figure 3: General structure of the direct (a) and inverse (b) COs and the $\mathbf{F}_{8;12}$ (c) and $\mathbf{F}^{-1}_{8;12}$ (d) boxes

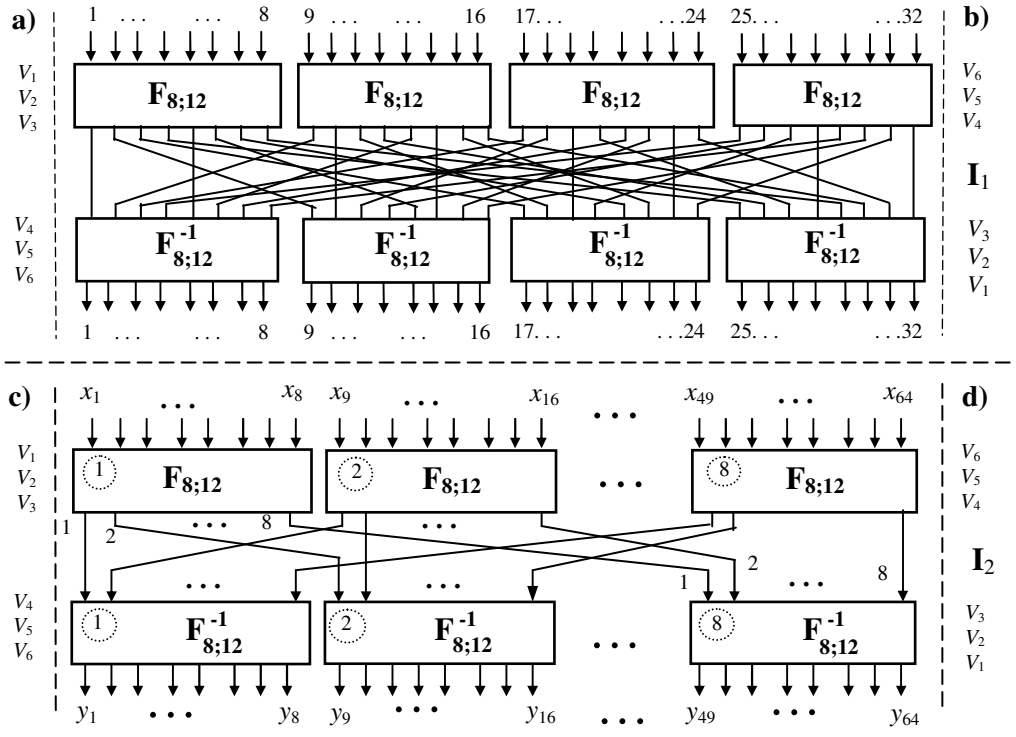

Figure 4: Structure of the $\mathbf{F}_{32;96}$ (a), $\mathbf{F}^{-1}_{32;96}$ (b), $\mathbf{F}_{64;192}$ (c), and $\mathbf{F}^{-1}_{64;192}$ (d) boxes

$$V_4 = L_2^{\lll 28}; \quad V_5 = L_2^{\lll 14}; \quad V_6 = L_2.$$

The described desing of the SCO boxes $\mathbf{F}^{(L,e)}_{32;96}$ and $\mathbf{F}^{(L,e)}_{64;192}$ is shown in Figure 6a,b. The proposed method allows one to embed the switchability mechanism in the $\mathbf{F}_{32;96}$ and $\mathbf{F}_{64;192}$ boxes using only 96 and 192 extra nand gates, correspondingly, versus 288 and 576 extra nand gates for the design proposed in [8].

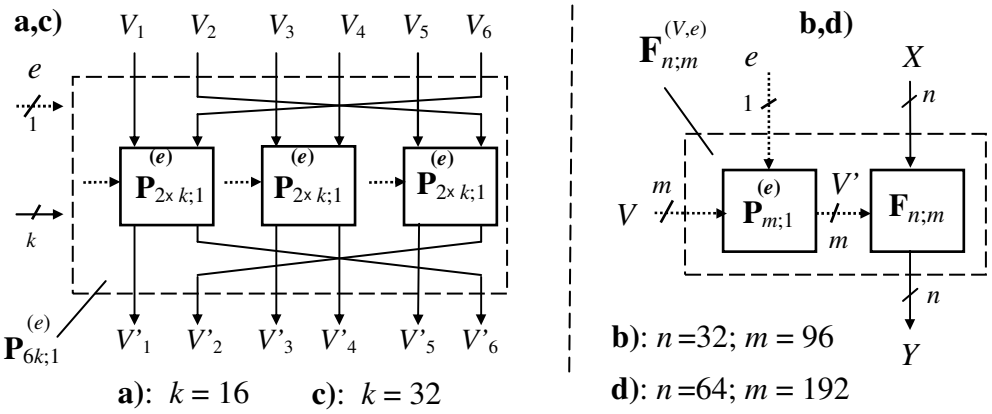

Figure 5: Structure of the boxes $\mathbf{P}^{(e)}_{96;1}$ (a), $\mathbf{F}^{(V,e)}_{32;96}$ (b), $\mathbf{P}^{(e)}_{192;1}$ (c), and $\mathbf{F}^{(V,e)}_{64;192}$ (d)

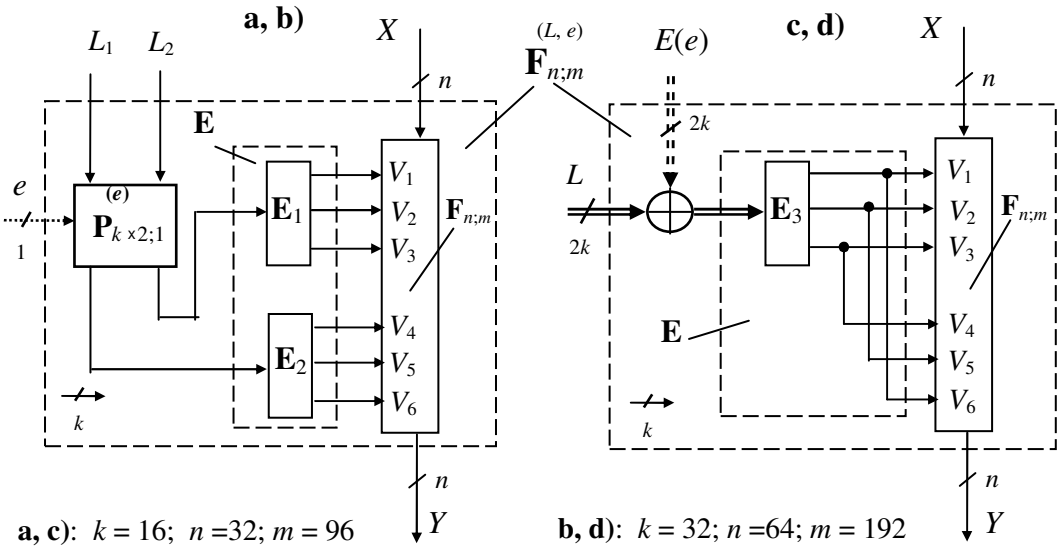

Figure 6: The $\mathbf{F}^{(L,e)}_{32;96}$ (a,c) and $\mathbf{F}^{(L,e)}_{64;192}$ (b,d) boxes with reduced implementation cost: a,b) - switching by swapping the left and right halves of the controlling data subblock; c,d) - switching by inverting the controlling bits

2.3 Design Based on CE with Mutual Inverse Modifications $\mathbf{F}^{(0)}_{2;1}$ and $\mathbf{F}^{(1)}_{2;1}$

Other hardware efficient design of the SCO boxes is based on the use of CEs that implements mutually inverse modifications $\mathbf{F}^{(0)}_{2;1}$ and $\mathbf{F}^{(1)}_{2;1}$ for which we have $\mathbf{F}^{(0)}_{2;1} = \left(\mathbf{F}^{(1)}_{2;1}\right)^{-1}$. In this design the symmetric topology of the CO boxes combined with symmetric distribution of the controlling bits (see Definition 8) is used. Figure 6c,d, where $E = \{e\}^{2k}$ is concatenation of $2k$ bits all of which are equal to e, illustrates the construction of the SCO boxes $\mathbf{F}^{(L,e)}_{32;96}$ (c) and $\mathbf{F}^{(L,e)}_{64;192}$ (d), where the $\mathbf{F}_{32;96}$ and $\mathbf{F}_{64;192}$ boxes have the topology shown in Figure 4. It is easy to show that this construction provides the relation $\mathbf{F}^{(L,e)}_{n;m} = \left(\mathbf{F}^{(L,e\oplus 1)}_{n;m}\right)^{-1}$.

Definition 8. *Distribution of the controlling bits of the* $V = V_1, ..., V_{[s/2]}$ *vector is called symmetric if* $\forall j = 1, ..., [s/2]$*, the following relation is hold:* $V_j = V_{s-j+1}$.

Table 2 lists all non-linear CEs having mutual inverse modifications $\mathbf{F}^{(0)}_{2;1}$ and $\mathbf{F}^{(1)}_{2;1}$. Note that for these CEs the sum $y_1 \oplus y_2$ is a non-linear BF.

2.4 Design of the SCO Boxes of different Orders

For odd values s the symmetrical topologies of the $\mathbf{F}_{n;m}$ boxes are not typical (in the case $h = n$ we have odd val-

Table 2: Full set of the $\mathbf{F}_{2;1}$ nonlinear boxes with mutual inverse modifications $\mathbf{F}_{2;1}^{(0)}$ and $\mathbf{F}_{2;1}^{(1)}$

(S′,S″)	$y_1 = f_1(x_1, x_2, v)$	$y_2 = f_2(x_1, x_2, v)$	$y_1 \oplus y_2$
(q,u)	$vx_1 \oplus x_2$	$vx_2 \oplus x_1 \oplus x_2$	$vx_1 \oplus vx_2 \oplus x_1$
(u,q)	$vx_1 \oplus x_1 \oplus x_2$	$vx_2 \oplus x_1$	$vx_1 \oplus vx_2 \oplus x_2$
(s,v)	$vx_1 \oplus v \oplus x_2$	$vx_2 \oplus v \oplus x_1 \oplus x_2 \oplus 1$	$vx_1 \oplus vx_2 \oplus x_1 \oplus 1$
(v,s)	$vx_1 \oplus v \oplus x_1 \oplus x_2 \oplus 1$	$vx_2 \oplus v \oplus x_1$	$vx_1 \oplus vx_2 \oplus x_2 \oplus 1$
(t,w)	$vx_1 \oplus x_2 \oplus 1$	$vx_2 \oplus v \oplus x_1 \oplus x_2$	$vx_1 \oplus vx_2 \oplus v \oplus x_1 \oplus 1$
(w,t)	$vx_1 \oplus x_1 \oplus x_2 \oplus 1$	$vx_2 \oplus v \oplus x_1 \oplus 1$	$vx_1 \oplus vx_2 \oplus v \oplus x_2$
(r,x)	$vx_1 \oplus v \oplus x_2 \oplus 1$	$vx_2 \oplus x_1 \oplus x_2 \oplus 1$	$vx_1 \oplus vx_2 \oplus v \oplus x_1$
(x,r)	$vx_1 \oplus v \oplus x_1 \oplus x_2$	$vx_2 \oplus x_1 \oplus 1$	$vx_1 \oplus vx_2 \oplus v \oplus x_2 \oplus 1$

ues s for minimum number of active layers and simmetrical topology), therefore it is interesting to develop a more general method to synthesize the SCO boxes than that considered in Subsections 2.2 and 2.3. Below we present a new method that allows one to construct the SCO boxes of the orders $h = 1, 2, ..., n/4$ using minimum number of active layers. The method is illustrated in Figure 7, where $\mathbf{F}_{n;m}$ and $\mathbf{F}_{n;m}^{-1}$ are the CO boxes of order h and the fixed permutation π_0 is described as follows:

$$\begin{pmatrix} 1 & 2 & ... & 2k-1 & 2k & ... & 2n-1 & 2n \\ 1 & n+1 & ... & k & n+k & ... & n & 2n \end{pmatrix}.$$

Due to the use of such permutation and its inverse π_0^{-1} each elementary box $\mathbf{F}_{2;1}$ of the top and bottom active layers $\mathbf{L}$ is connected with both the $\mathbf{F}_{n;m}$-box and the $\mathbf{F}_{n;m}^{-1}$-box providing doubling the order h. The method is based on the use of two mutually inverse CO boxes $\mathbf{F}_{n;m}^{V^{(1)}}$ and $\left(\mathbf{F}^{-1}\right)_{n;m}^{V^{(2)}}$ of the same order h and allows one to synthesize the SCO box $\mathbf{F}_{2n;2(m+n)}^{(V,e)}$ of the order $2h$, where $V = (V_1, V^{(1)}, V^{(2)}, V_s)$ is the controlling vector of the synthesized CO box and $s = 2(m/n+1)$ is the number of active layers in the last. The size of the vectors $V^{(1)}$ and $V^{(2)}$ differs from the size of V_1 and V_s. It is easy to show that for all values V and $e \in \{0, 1\}$ we have

$$\mathbf{F}_{2n;2(m+n)}^{(V,e\oplus 1)} = \left(\mathbf{F}_{2n;2(m+n)}^{(V,e)}\right)^{-1}.$$

In the case $n = 2^k$ the minimum number of the active layers in the box $\mathbf{F}_{n;m}$ (or $\mathbf{F}_{n;m}^{-1}$) of the order $h = 1, 2, ..., n/4$ is $s_{\min} = \log_2 hn$ (in the case $h = n$ we have $s_{\min} = \log_2 hn - 1$) [10]. To construct the SCO box $\mathbf{F}_{n';m'}^{(V,e)}$, where $n' = 2n$ and $m' = 2(n + m)$, of the order $h' = 2, 4, ..., n'/4$ using the described synthesis method one should use the boxes $\mathbf{F}_{n;m}$ and $\mathbf{F}_{n;m}^{-1}$ of the order $h = h'/2$ and add two additional active layers. The minimum number of the required active layrs is

$$s'_{\min} = \log_2 hn + 2 = \log_2 4hn = \log_2 h'n'.$$

The SCO boxes of the maximum order (i.e. $h' = n'$) can be constructed using the $(2\log_2 n - 1)$-layer boxes $\mathbf{F}_{n;m}$ and $\mathbf{F}_{n;m}^{-1}$ of the order $h = n$ [10]. For this case we have

$$s'_{\min} = (2\log_2 n - 1) + 2 = 2(\log_2 n + 1) - 1 = 2\log_2 n' - 1.$$

Thus, for the given order $h = 2, 4, ..., 2^k$ the implementation of switchable ($\mathbf{F}^{(V,e)}$) and ordinary ($\mathbf{F}^{(V)}$) CO boxes requires the same minimum number of the active layers. If $n = 2^k$, then it is easy to construct the maximum-order box $\mathbf{F}_{n;m}^{(V)}$ with symmetric structure. Therefore the maximum-order switchable box $\mathbf{F}_{n;m}^{(V,e)}$ can be also synthesized analogously to different constructions of the SCO boxes with symmetric topology.

3 Fast SCO-based Cipher Hawk-64

The property of the controllability of the operations used as cryptographic primitive provides possibility to design different types of the iterative block cryptoschemes with simple key scheduling, which can be implemented in cheap hardware. The property of the switchability allows avoiding the weak keys while using the simple key scheduling. Figure 8 presents new 64-bit cipher Hawk-64 particular feature of which is the combining SPN (S_i operation performed on the right data subblock) with CSPNs (two SCO boxes $\mathbf{F}_{32;96}^{(L,e_1)}$ and $\mathbf{F}_{32;96}^{(L,e_2)}$ in the left branch of the round cryptoscheme). The design of the used SCO boxes is explained in Figures 3d,c,4a,b, and 6a, where the (i,j) CE is used as building block $\mathbf{F}_{2;1}$. Hawk-64 uses 128-bit key $K = (K_1, K_2, K_3, K_4)$ ($K \in \{0, 1\}^{32}$) and very simple key scheduling that is the same while enciphering and deciphering. However different scheduling of the bits e_1 and e_2 is used while encryption and decryption.

Ciphering procedure of Hawk-64 is described as follows: $C = \mathbf{T}^{(e=0)}(M, K)$ and $M = \mathbf{T}^{(e=1)}(C, K)$, where M is the plaintext, C is the ciphertext ($M, C \in \{0, 1\}^{64}$), $\mathbf{T}$ is the transformation function, and $e \in \{0, 1\}$ is a parameter defining encryption ($e = 0$) or decryption ($e = 1$) mode. Iterative structure of Hawk-64 is shown in Figure 8a. First data block is divided into two 32-bit subblocks A and B and then using the procedure $\mathbf{Crypt}^{(e)}$ eight encryption rounds are performed. The last round is followed by final transformation (FT). The structure of the procedure $\mathbf{Crypt}^{(e)}$ is shown in Figure 8b. The FT is performed as XORing data subblocks with respective

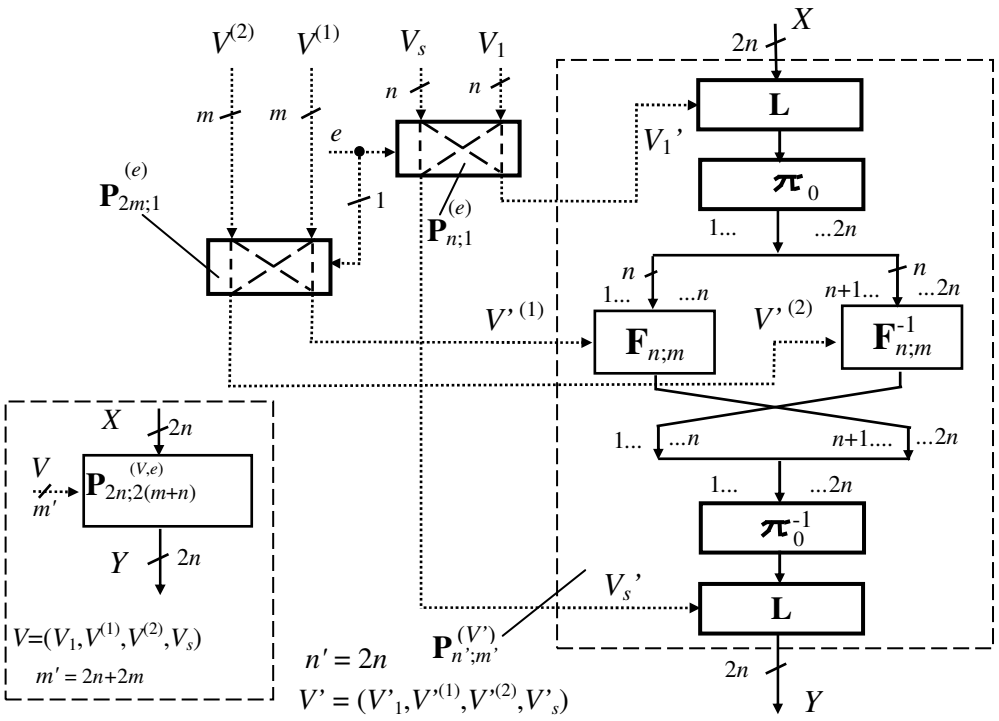

Figure 7: Synthesis of the SCO boxes of different orders

Table 3: Key scheduling and specification of the switching bits e' and e''

j	1	2	3	4	5	6	7	8	FT
Q_j	K_1	K_2	K_3	K_4	K_4	K_1	K_2	K_3	K_1
U_j	K_3	K_3	K_2	K_1	K_4	K_4	K_3	K_2	K_3
e'	1	0	1	1	0	1	0	0	-
e''	0	0	1	0	1	1	0	1	-

Table 4: The $\mathbf{I}_1$, Π_1, Π_2, and Π_3 permutations

$\mathbf{I}_1$	(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)
Π_1	(1)(2,5)(3,17)(4,21)(6)(7,18)(8,22)(9)(10,13)(11,25)(12,29)(14)(15,26)(16,30)(19)(20,23)(24)(27)(28,31)(32)
Π_2	(1)(2,5)(3,9)(4,13)(6)(7,10)(8,14)(11)(12,15)(16)(17)(18,21)(19,25)(20,29)(22)(23,26)(24,30)(27)(28,31)(32)
Π_3	(1,3,19,17)(2,7,20,21)(4,23,18,5)(6,8,24,22)(11,27,25,9)(10,15,28,29)(12,31,26,13)(14,16,32,30)

subkeys: $A := A \oplus K_1$ and $B := B \oplus K_3$.

The $\mathbf{F}_{32;96}^{(B,e_1)}$ and $\mathbf{F}_{32;96}^{(B,e_2)}$ boxes are constructed using the (i,j) elements as standard building blocks in correspondence with topology described in Section 2.1. The e_1 and e_2 values depend on e and round number: $e_1 = e' \oplus e$ and $e_2 = e'' \oplus e$, where e' and e'' are specified in Table 3. The permutational involution $\mathbf{I}_1$ in the left branch of the round transformation is the same as that corresponding to connection between cascade of four parallel boxes $\mathbf{F}_{8;12}$ and cascade of four boxes $\mathbf{F}_{8;12}^{-1}$ in the box $\mathbf{F}_{32;96}$ (see Figure 4a,b). The S_i operation is involution (see Appendix 1 for more detailed comments). It is a SPN constructed using the Π_1, Π_2, and Π_3 permutations (specified in Table 4) and the following 4×4 S-box substitutions: direct ones S_0,..., S_7 and respective inverses S_0^{-1},..., S_7^{-1} (specified in Table 5). Eight 4×4 S-boxes of the DES cipher (one from each of eight 6×4 S-boxes) have been selected as the S_0,..., S_7 boxes of Hawk-64 in order to ispire a high level of public confidence that no trapdoor are inserted in Hawk-64. Similar justification of the S-boxes selection has been earlier used in the design of the Serpent cipher [2].

4 Discussion and Conclusion

The use of the DDO is oriented to the cipher design using simple key scheduling [7, 14]. Earlier [8] the SCOs have ben proposed to prevent the weak keys while designing cryptosystems with simple key scheduling. In this paper we have proposed three new designs of SCOs. The FPGA (ASIC) implementation cost of SCOs $\mathbf{F}_{n;m}^{(L,e)}$ based on the designs described in Sections 2.2 and 2.3 is 117% ($\approx 112-125\%$ for different types of CEs) in comparison with the corresponding COs $\mathbf{F}_{n;m}$. For SCOs described in [8] the respective figures are 150% (FPGA) and $\approx 135 - 175\%$ (ASIC).

One of the designed SCOs has been used in new cipher Hawk-64. We have implemented Hawk-64 using FPGA Xilinx Vitrex Device and the loop unrolling architecture (denoted as LU-N, where N is number of the unrolled en-

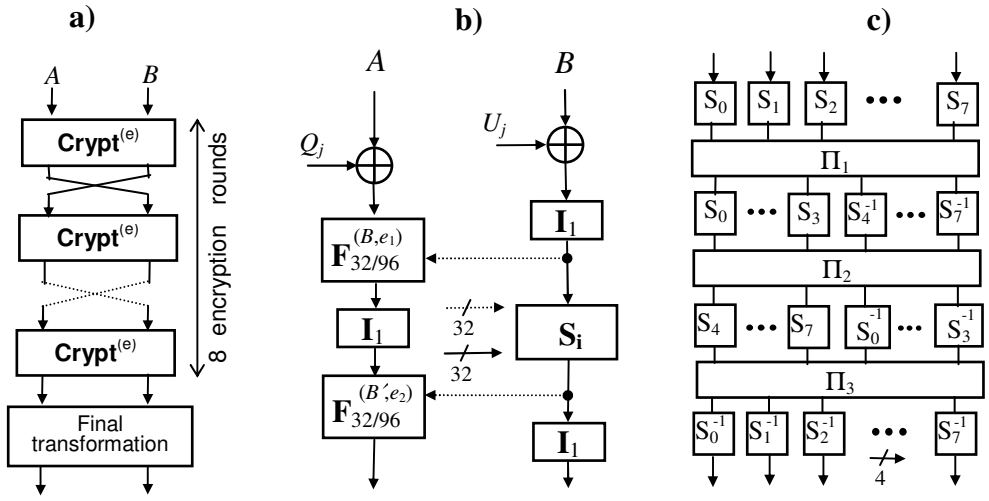

Figure 8: Hawk-64: a) - iterative structure, b) - procedure $\mathbf{Crypt}^{(e)}$, and c) - topology of the S_i operation

cryption rounds [1]; the iterative looping architecture corresponds to LU-1). Due to the use of the FPGA-oriented primitives this cipher is significantly more efficient for the FPGA implementation against known ciphers DES, SAFER+, Cobra-H64 [14], SCO-1(2,3) [12] and many others including AES finalists. Table 6 (where DFFs - D Flip-Flops, CLBs - Configurable Logic Blocks, FGs - Function Generators) compares FPGA implementation efficiency (estimated as ratio "performance/cost" [1] and "performance/(cost·frequency)") of Hawk-64 with some known 64- and 128-bit DDO-based ciphers, AES, Serpent, RC6, and Twofish.

Investigation of the statistic properties of Hawk-64 has been carried out with standard tests which have been used in [11] for five AES finalists. The obtained results have shown that two rounds of Hawk-64 are sufficient to satisfy the test criteria. Our preliminary security estimation of Hawk-64 shows that it is indistinguishable from a random cipher with differential, linear and other attacks. Differential analysis appears to be more efficient than linear attack. This corresponds to ealier results [4, 5, 14] on analysis of the DDP-based ciphers, which have shown that linear analysis is less efficient than the differential attack even against ciphers based on the DDP operations that are a linear primitive. Our best differential characteristics correspond to the two-round differences with one active bit. Such characteristics have probability $P(2) \leq 2^{-32}$. The differences pass through one round with probability $P' = 2^{-12}$, if the active bit passes through the left branch of the round transformation, and with probability $P'' \leq 2^{-20}$, if the active bit passes through the right branch of the round transformation (formation scheme of the iterative two-round differential characteristic is resented in Appendix 2).

Four rounds of Hawk-64 are sufficient to thwart the differential attacks. Additional four rounds have been added to get the 100% security margine (measured as $\frac{R-R_{\min}}{R_{\min}} \cdot 100\%$, where R and $R_{\min}$ are the specified and minimum secure number of rounds, correspondingly). The slide attacks [3] against Hawk-64 are not efficient, since this cipher uses non-periodic key scheduling and non-periodic specification of the bits e' and e'' that difine direct or inverse transformations are performed with the operations $\mathbf{F}^{(B,e_1)}_{32;96}$ and $\mathbf{F}^{(B',e_2)}_{32;96}$ in each of eight encryption rounds. The related-key differential analysis [6] is also not efficient, since each of the subkeys is used at least in three different rounds.

The main results of this paper can be formulated as follows:

1) Three new desings of SCOs that are efficient for hardware implementation have been developed.

2) A new SCO-based cipher Hawk-64 that is efficient for application in the constrained environments has been proposed. One of the features of Hawk-64 is using the same key scheduling for both the data encryption and the data decryption. Analysis of the algorithm Hawk-64 has shown that it is secure against known attacks.

3) The FPGA implementation cost and integral efficacy of Hawk-64 has been estimated and compared with conventional ciphers.

References

[1] A. J. Albirt, W. Yip, B. Ghetwynd, and C. Paar, "FPGA implementation and performance evaluation of the AES block cipher candidateAlgorithm finalists," *3rd Advanced Encryption Standard Conference Proceedings*, New York, NY, USA, Apr. 13-14, 2000.

[2] R. Anderson, E. Biham, and L. Knudsen, "Serpent: A proposal for the advanced encryption standard,"

Table 5: The S-boxes used in the S_i operation

S-box	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
S_1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
S_2	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
S_3	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
S_4	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
S_5	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
S_7	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
S_0^{-1}	14	3	4	8	1	12	10	15	7	13	9	6	11	2	0	5
S_1^{-1}	9	10	5	0	2	15	12	3	6	13	11	14	8	1	7	4
S_2^{-1}	1	8	14	5	13	7	4	11	15	2	0	12	10	9	3	6
S_3^{-1}	12	0	15	5	1	13	10	6	11	14	8	2	4	3	7	9
S_4^{-1}	3	9	13	10	15	12	1	6	14	2	0	15	4	7	11	8
S_5^{-1}	10	4	6	15	13	14	8	3	1	11	12	0	2	7	5	9
S_6^{-1}	12	9	3	14	2	7	8	4	15	6	0	13	5	10	11	1
S_7^{-1}	12	0	15	5	7	9	10	6	3	14	4	11	8	2	13	1

Table 6: The FPGA (Xilinx Virtex Devices) implementation synthesis results for the LU-1 architecture

Cipher	Block size	# rounds	Covered area			F (*MHz*)	Rate (*Mbps*)	Integral efficacy	
			CLBs	FGs	DFFs			$\frac{Mbps}{CLBs}$	$\frac{Mbps}{CLBs \cdot GHz}$
Hawk-64 (proposed)	64	8	560	1,019	166	85	670	1.2	14.1
SPECTR-H64 [13]	64	12	713	1,320	203	83	443	0.62	7.5
Cobra-H64 [14]	64	10	615	1,229	204	82	525	0.85	10.4
Cobra-H128 [14]	128	12	2,364	4,728	399	86	917	0.39	4.5
AES [13]	128	10	2,358	-	-	22	259	0.11	5.0
AES [1]	128	10	3,528	-	-	25.3	294	0.083	3.3
Serpent [1]	128	32	5,511	-	-	15.5	61.9	0.011	0.6
RC6 [1]	128	20	2,638	-	-	13.8	88.5	0.034	2.4
Twofish [1]	128	16	2,666	-	-	13	104	0.039	3.0

1st Advanced Encryption Standard Candidate Conference Proceedings, Venture, California, Aug. 20-22, 1998.

[3] A. Biryukov, and D. Wagner, "Advanced slide attacks," *Eurocrypt '00*, LNCS 1807, pp. 589-606, Springer-Verlag, 2000.

[4] N. D. Goots, V. B. Izotov, A. A. Moldovyan, and N. A. Moldovyan, "Fast ciphers for cheap hardware: Differential analysis of SPECTR-H64," *Proceedings of the Second International workshop on Mathematical Methods, Models, and Architectures for Computer Network Security*, LNCS 2776, pp. 449-452, Springer-Verlag, 2003.

[5] Y. Ko et al., "Linear cryptanalysis on SPECTR-H64 with higher order differential property," *Proceedings of the Second International workshop on Mathematical Methods, Models, and Architectures for Computer Network Security*, Springer-Verlag, LNCS 2776, pp. 298-307, 2003.

[6] Y. Ko, S. Hong, W. Lee, S. Lee, and J.-S. Kang, "Related key differential attacks on 27 round of XTE and full-round GOST," *Proceedings of the 11th International Workshop, Fast Software Encryption - FSE '2004*, LNCS 3017, pp. 299-316, Springer-Verlag, 2004.

[7] A. A. Moldovyan, and N. A. Moldovyan, "A cipher based on data-dependent permutations," *Journal of Cryptology*, vol. 15, no. 1, pp. 61-72, 2002.

[8] N. A. Moldovyan, "On cipher design based on switchable controlled operations," *Proceedings of the Second International workshop on Mathematical Methods, Models, and Architectures for Computer Network Security*, Springer-Verlag, LNCS 2776, pp. 316-327, 2003.

[9] N. A. Moldovyan, A. A. Moldovyan, M. A. Eremeev, and N. Sklavos, "New class of cryptographic primitives and cipher design for network security," *International Journal of Network Security*, vol. 2, no. 2, pp. 114-125, 2006.

[10] N. A. Moldovyan, and A. A. Moldovyan, *Innovative Cryptography*, pp. 386, Charles River Media, Boston, Massachusetts, 2006.

[11] B. Preneel et al., *Comments by the NESSIE project on the AES finalists*, May 24, 2000. http:// www.nist.gav/aes.

[12] N. Sklavos, and O. Koufopavlou, "Architectures and FPGA implementations of the SCO (-1,-2,-3) ciphers family," *Proceedings of the of 12th International Conference on Very Large Scale Integration, (IFIP VLSI SOC '03)*, pp. 68-73, Darmstadt, Germany, Dec. 1-3, 2003.

[13] N. Sklavos, A. A. Moldovyan and O. Koufopavlou, "Encryption and data dependent permutations: Implementation cost and performance evaluation," *Proceedings of the Second International workshop on Mathematical Methods, Models, and Architectures for Computer Network Security*, LNCS 2776, pp. 337-348, 2003.

[14] N. Sklavos, N. A. Moldovyan, and O. Koufopavlou, "High speed networking security: Design and implementation of two new DDP-based ciphers," *Mobile Networks and Applications*, vol. 10, no. 1, pp. 237-249, 2005.

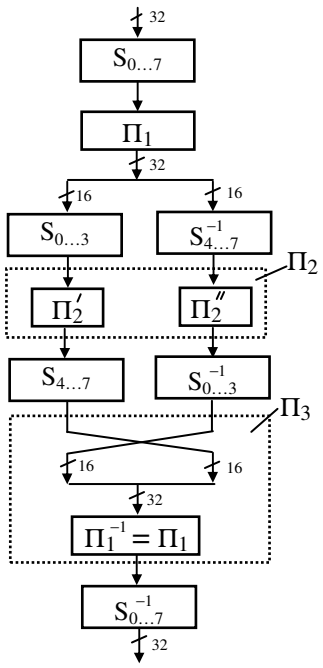

Figure 9: Structure of the S_i operation

Appendix 1

Figure 9 explains the structure of the operation S_i, where the following designations are used:

$S_{0...7}$ is the cascade of the boxes S_0, S_1,..., S_7;
$S_{0...7}^{-1}$ is the cascade of the boxes S_0^{-1}, S_0^{-1},..., S_7^{-1};
$S_{0...3}$ is the cascade of the boxes S_0, ..., S_3;
$S_{0...3}^{-1}$ is the cascade of the boxes S_0^{-1}, ..., S_3^{-1};
$S_{4...7}$ is the cascade of the boxes S_4, ..., S_7;
$S_{4...7}^{-1}$ is the cascade of the boxes S_4^{-1}, ..., S_7^{-1}.

The Π_3 permutation is represented as superposition $\mathbf{T} \circ \Pi_1^{-1}$ ($\mathbf{T}$ is the transposition operation and $\Pi_1^{-1} = \Pi_1$). The Π_2 permutational involution is represented as cascade of two involutions Π_2' and $\Pi_2'' = \Pi_2'$.

Thus, the S_i can be represented as the following superposition:

$$S_{0...7} \circ \Pi_1 \circ \left(S_{0...3}, S_{4...7}^{-1}\right) \circ \Pi_1 \circ \left(S_{4...7}, S_{0...3}^{-1}\right) \circ \mathbf{T} \circ \Pi_1^{-1} \circ S_{0...7}^{-1}.$$

Taking into account such representation of the S_i operation it is easy to see that

$$S_i\left(S_i(X)\right) = X,$$

where X is arbitrary 32-bit binary vector.

Appendix 2

Formation scheme of the two-round differential characteristic (DC) is presented in Figure 10, where Δ_i^A and Δ_i^B denote differences of the A and B data subblocks. The index i indicates the weight of the difference, i. e. the number of active bits in the difference. The numbers of the digits to which the active bits correspond are not taken into account, i. e. we consider some "integral" DCs that relates to the set of DC with the indicated weight i. Efficient DCs correspond to the differences with few active bits. The iterative two-round differences $\left(\Delta_1^A, \Delta_0^B\right)$ and $\left(\Delta_0^A, \Delta_1^B\right)$ pass through two rounds with probability $P(2) \leq 2^{-32}$.

For example, the formation scheme of the DC with the difference $\left(\Delta_1^A, \Delta_0^B\right)$ is presented in Figure 10. In the first round the active bit passes 12 active layers of the boxes $\mathbf{F}_{32;96}^{(B,e_1)}$ and $\mathbf{F}_{32;96}^{(B',e_2)}$ with the probability $P' \approx 2^{-12}$. Inded, one active bit passes one active layer with probability $p = 2^{-1}$ [10]. When passing through the box $\mathbf{F}_{32;96}^{(B,e_1)}$ (or $\mathbf{F}_{32;96}^{(B',e_2)}$), six different bits of the data subblock B control the $\mathbf{F}_{2;1}$ elements through which the active bit passes. Therefore the active bit passes through the box $\mathbf{F}_{32;96}^{(B,e_1)}$ (or $\mathbf{F}_{32;96}^{(B',e_2)}$) with probability $p = 2^{-6}$. In the second round the active bit passes through the S_i operation with probability $p_s \leq 2^{-8}$ as it is shown in Figure 10. Besides, it controls three CEs in each of the boxes $\mathbf{F}_{32;96}^{(B,e_1)}$ and $\mathbf{F}_{32;96}^{(B',e_2)}$. Each of the indicated CEs generates no active bit with probability $p_1 = 2^{-2}$ [10], therefore no active bits are generated in the boxes $\mathbf{F}_{32;96}^{(B,e_1)}$ and $\mathbf{F}_{32;96}^{(B',e_2)}$ with probability $p_2 = (p_1)^6 = 2^{-12}$, therefore we have $P'' = p_s p_2 \leq 2^{-20}$.

Thus, we have $P(2) = P'P'' \leq 2^{-32}$.

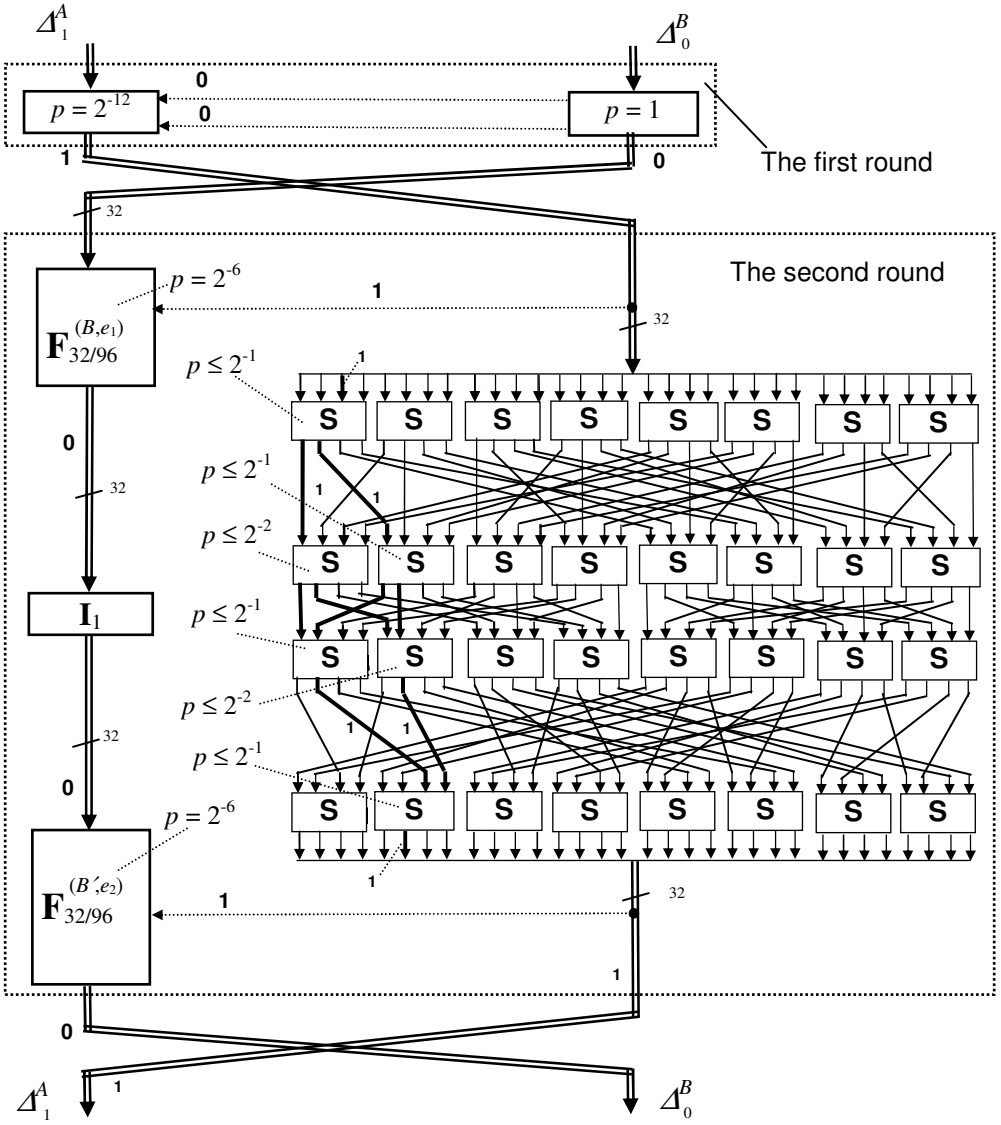

Figure 10: Formation of the two-round iterative differential characteristic with the difference $\left(\Delta_1^A, \Delta_0^B\right)$ and probability $P(2) \le 2^{-32}$

Nikolay A. Moldovyan is an honored inventor of Russian Federation (2002), a chief researcher with the Specialized Center of Program Systems "SPECTR", and a Professor with the Saint Petersburg Electrical Engineering University. His research interests include computer security and cryptography. He has authored or co-authored more than 50 inventions and 200 scientific articles, books, and reports. He received his Ph.D. from the Academy of Sciences of Moldova (1981). Contact him at: nmold@cobra.ru.