VLSI Design, 2002 Vol. 14 (1), pp. 23–34

Taylor & Francis
Taylor & Francis Group

Circuit Synthesis from Fibonacci Decision Diagrams

RADOMIR S. STANKOVIĆ[a,*], JAAKKO ASTOLA[b], MILENA STANKOVIĆ[a] and KAREN EGIAZARIAN[b]

[a]*Faculty of Electronics, Department of Computer Science, University of Niš, Beogradska 14, 18 000 Niš, Yugoslavia;* [b]*International Center for Signal Processing, Tampere University of Technology, Tampere, Finland*

(Received 20 January 2000; In final form 4 October 2000)

In decision diagrams (DDs) methods for circuit synthesis, it is possible to directly transfer a DD for a given function *f* into a network realizing *f* by the replacement of non-terminal nodes in the DD with the corresponding circuit modules.

The chief bottleneck of mapping a DD into a network is the inherent feature that the depth of the network produced, is equal to the number of variables in *f*. For this reason, it is proposed a method for small depth circuit synthesis through reachability matrices describing connections among the nodes in the DD for *f*.

In this paper, we first generalized DD methods for circuit design to Fibonacci interconnection topologies through the Fibonacci decision diagrams (FibDDs). Then, we extended the small depth circuit synthesis method to FibDDs. In this way, design methods through DDs are completely transferred from Boolean to Fibonacci topologies.

Keywords: Switching functions; Spectral transforms; Circuit synthesis; Fibonacci sequences; Fibonacci transforms; Decision diagrams

INTRODUCTION

The hypercube architectures and related Boolean interconnection topologies are extensively used in systems design and logic design. However, in some applications, they express some inconveniences originating in their inherent features, as for example, restrictions to the power of two in the number of nodes or inputs, etc. For this reason, the generalized Fibonacci interconnection topologies are offered as an alternative [11–14,18,31].

Example 1 Table I shows the coding of first eight non-negative integers in the Boolean topology. Table II *shows the corresponding Fibonacci 1-code. Figure 1compares the Boolean cube of order* $n = 3$, *and the Fibonacci cube of the same order derived from these codings.*

There are few reasons to study the generalized Fibonacci topologies. We want to point out the following:

1. Boolean n-cube is involved in the set of generalized Fibonacci cubes.
2. The dimension of a generalized Fibonacci cube which can be embeded in the Boolean n-cube with $k = 1, 2$ faulty nodes is greater than 2^{n-1}.
3. The k-th order Fibonacci cube of the dimension $n + k$ is equivalent to a Boolean n-cube for $0 \leq n < k$. It follows that algorithms developed for a generalized Fibonacci cube are executable on the Boolean cube of the corresponding order.

Binary decision diagrams (BDDs) [3], and their different generalizations [29,35,37], are a standard data structure in many CAD systems [10] related to functions on Boolean topologies. Fibonacci DDs (FibDDs) [39] are extensions of DDs representations to functions on Fibonacci topologies. Therefore, it could be interesting to transfer design methods through different DDs in Boolean topologies to Fibonacci DDs.

In this paper, we extend the method for circuit synthesis through DDs [22], and Kronecker DDs [16], for switching functions, and through Galois field DDs (GFDDs) for MV functions [33,34], to Fibonacci DDs. We also generalized the method for small depth circuit synthesis through the reachability matrices [17,36] to Fibonacci DDs. Some basic concepts about the Fibonacci numbers, codes and related transforms are given in the Addendum.

*Corresponding author.

ISSN 1065-514X print/ISSN 1563-5171 online © 2002 Taylor & Francis Ltd
DOI: 10.1080/10655140290009783

TABLE I Binary coding in Boolean topology

x	x_1	x_2	x_3
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

TABLE II Fibonacci 1-code

w	k_1	k_2	k_3	k_4
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	1	0	0
4	0	1	0	1
5	1	0	0	0
6	1	0	0	1
7	1	0	1	0

FIBONACCI DECISION DIAGRAMS

FibDTs

Fibonacci decision diagrams (FibDDs) are a data structure for efficient in terms of space and time representation of functions defined in a set of points whose cardinality is equal to a generalized Fibonacci p-number [2,14].

We define the Fibonacci DTs by using the decomposition of the cardinal numbers of the domains for the represented functions in terms of the generalized Fibonacci numbers $\phi_p(i)$.

We assume that f is defined on a set of points $w \in \{0, \ldots, \phi_p(i) - 1\}$ coded by binary sequences in the Fibonacci p-code. Thus, f is defined by its values $f(w) = f(w_0, \ldots, w_n)$, where w_j are the coordinates in binary representation for w in the Fibonacci p-code.

DEFINITION 1 *For given values of p* and *i, the Fibonacci DT (Fib$_p$*DT(i) for $f(w_1, \ldots, w_n)$ *is defined by the recursive application of the Fibonacci-Shannon decomposition rule* $f = \overline{w}_i f_0 + w_i f_1$, *where* $f_0 = f(w_1, \ldots, w_{i-1}, 0, w_{i+1}, \ldots, w_n)$, *and* $f_1 = f(w_1, \ldots, w_{i-1}, 1, w_{i+1}, \ldots, w_n)$, *to all the variables* w_j in f.

To provide a possibility to discuss relationship and differences with Binary DDs [3], we consider as an example the generalized Fibonacci numbers for $p = 1$.

Example 2 *If the cardinality of the domain for* f is $N = 8 = \phi_1(5)$, *then* $\phi_1(5) = \phi_1(4) + \phi_1(3) = (\phi_1(3) + \phi_1(2) + \phi_1(3)$, *and we have the decomposition* $8 = 5 + 3 = (3 + 2) + 3$. *With this decomposition, the Fibonacci DT for* $p = 1$ *and* $i = 5$ *is built up from the basic Fibonacci DTs corresponding to the additive factors 3 and 2, respectively.*

Figure 2 shows the Fib$_1$DT(5) build up as a combination of the basic Fibonacci DTs. This combination is determined by the assumed decomposition of N in terms of the generalized Fibonacci numbers. The constant nodes represent values of f at particular points. Thus, Fib$_1$DT(5) has four levels, since the additive decomposition of 8 in terms of the generalized Fibonacci 1-numbers is done in four steps as $8 = 5 + 3 = (3 + 2) + 3 = (2 + 1) + 2) + (2 + 1) = ((1 + 1) + 1) + (1 + 1) + (1 + 1) + 1)$.

This Fib$_1$DT(5) represents f in the form of the Fibonacci expression

$$f = c_0\overline{w}_1\overline{w}_2\overline{w}_3\overline{w}_4 + c_1\overline{w}_1\overline{w}_2\overline{w}_3 w_4 + c_2\overline{w}_1\overline{w}_2 w_3 + c_3\overline{w}_1 w_2\overline{w}_4 + c_4\overline{w}_1 w_2 w_4 + c_5 w_1\overline{w}_3\overline{w}_4 + c_6 w_1\overline{w}_3 w_4 + c_7 w_1 w_3.$$

This example illustrates the chief properties of Fibonacci DTs.

A Fib$_p$DT(i) has $(i - p)$ levels, with each level corresponding to a step in the recursive determination of $\phi_p(i)$ through addition of $\phi_p(i-1)$ and $\phi_p(i-p-1)$. Since the step in this recursion is equal to p, the outgoing edges of a node at the j-th level point to the nodes at the $(j-1)$-th level and the $(j-p-1)$-th level in the Fib$_p$DT(i). Thus, in a Fib$_p$DT, the left outgoing edges are of the length 1, since connect the successing levels. The right outgoing edges are of the length $p + 1$.

FibDDs

DEFINITION 2 *FibDDs are derived by the reduction of FibDTs with the generalized BDD reduction rules* [38].

Reduction consists of sharing isomorphic subtrees. Therefore, in a FibDD, the left edges longer than one, and the right edges longer than $(p + 1)$ may appear. The crossing of such an edge with a level in the DT is denoted

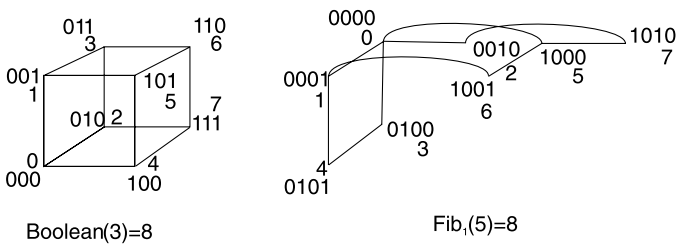

FIGURE 1 Boolean and Fibonacci cubes.

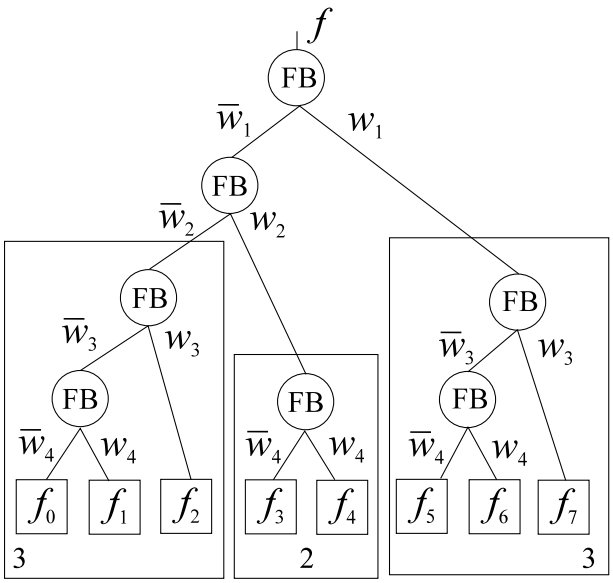

FIGURE 2 Fib_1DT(5).

as the cross point [38]. The cross points take into account the impact of deleted nodes in transferring a FibDT into the corresponding FibDD.

Example 3 *Figure 3 shows Fib_1DD(5) for f given by the truth-vector* $\mathbf{F} = [1,0,1,1,0,1,1,0]^{\mathrm{T}}$.

CIRCUIT SYNTHESIS FROM DDs

DDs are efficiently used to design logical networks for realization of switching functions, see for example, Refs. [9,17,20,22,25,29,40]. These realization architectures are based upon the multiplexers, Reed-Muller modules or suitable FPGAs [27,30]. Some recent results in that area are given in Ref. [15].

Design of circuits from DDs is efficiently used in FPGA synthesis [9,27,30]. Such realizations often express high testability properties [6,8].

The application of Lattice DDs [24], defined as a generalization of BDDs adapted to the synthesis with regular layout networks [23], in the synthesis of fuzzy logic and analog circuit design was considered in Ref. [24].

DDs design methods are extended to the realization of MV functions, see for example Ref. [36], and to the architectures for calculation of spectral transforms, see for example, Ref. [22].

The generalization to the architectures for calculation of spectral transforms is easy done if is expressed in terms of spectral interpretation of DDs [38].

A direct Fib_pDDs based synthesis method is to substitute each node and each cross point by a circuit realizing the Fibonacci-Shannon expansion [39]. That is a straightforward generalization of the corresponding DDs based methods for switching functions [4–6,8,21,28]. Such realizations for Galois field DDs (GFDDs) with

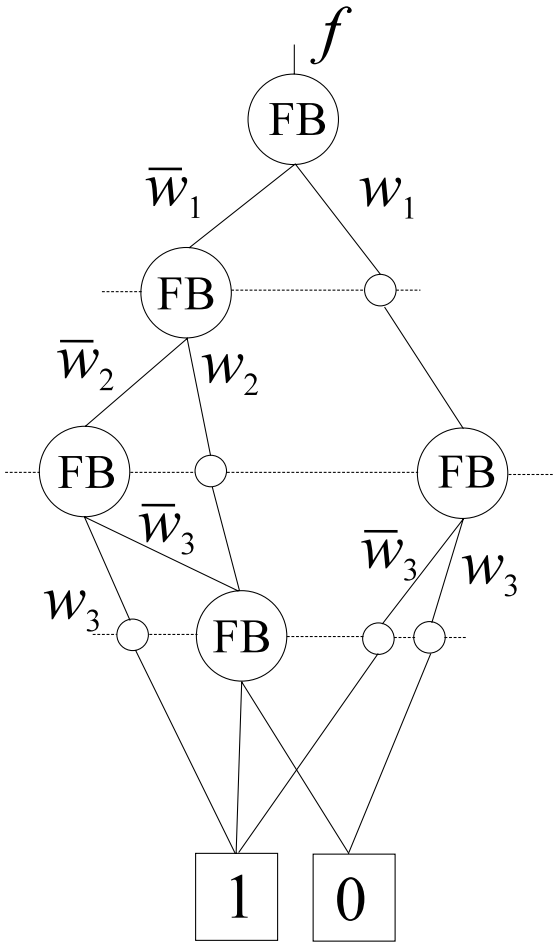

FIGURE 3 Fib_1DD(5).

4-positive Davio nodes and in Gibbs algebras are considered in Refs. [33,34]. The same method applies to Fibonacci DDs.

Main disadvantage of the presented DDs based design methods is due to the propagation delay, since the depth of the circuit produced is equal to the number of variables. Therefore, the design methods for small depth circuits are proposed for BDDs [17], and KDDs [16]. Some other solutions use the functional decomposition [7,26].

Method for Circuit Synthesis from DDs

In what follows, we will consider design of architectures for realization of a given function *f* and the calculation of some spectral transform for *f*. We denote these architectures as the realization and calculation architectures, respectively.

In Ref. [34], design of both the realization and calculation architectures from DDs was formulated as follows.

Given a discrete function *f* by a DD that is defined by using the decomposition rule Q_j for the nodes at the *j*-th level in the corresponding DT. In spectral interpretation of DDs, Q_j is considered as the basic operation in the

FFT-like algorithm for a spectral transform Q used to assign f to the DT.

1. To realize f, assign a module $\mathcal{Q}$ performing the operation inverse to Q_j to each node and the cross point in the DD for f.
2. To calculate the Q-spectrum for f, assign a multiplexer-like module to each node and the cross point in the DD for f.

Note that in FibDTs, the same as in BDTs and MTBDTs, we use the identical mapping to assign f to the DT. The identical mapping is a self-inverse mapping. It follows that the architectures for realization of f and the calculation of the corresponding Fibonacci p-spectra for f can be designed from the FibDDs for f in the following way.

1. To realize f, assign a multiplexer-like module to each node in the FibDT for f.
2. To calculate the Fibonacci p-spectrum for f, assign to each node and the cross point in the FibDT for f a module realizing the basic operation used in calculation of this transform through DDs using the top-down strategy.

The method will be explained and illustrated by the examples of architectures for realization of a given f defined in $\phi_1(5)$ points and for the calculation of the FWHT and the Fibonacci-Haar spectra of order $\phi_1(5)$.

Architecture for Realization of *f*

In the FibDD for a given f, the constant nodes shows the values $f(w)$, $w \in \{0, \ldots, \phi_p(i) - 1\}$, for f at particular points of the domain of definition for f. These points are denoted by binary sequences in the Fibonacci p-code. The FibDD consists of nodes with two outgoing edges. Therefore, to design an architecture which realizes f, it is enough to assign a (2×1) multiplexer to each node in the FibDD for f.

The sequence $k(w) = \{k_1, \ldots, k_{i-p}\}$ of the values of control inputs for the multiplexers at the moment w is equal to the code word w in the Fibonacci p-code. The following example illustrates the design of a realization architecture for a given f derived from the FibDTs. The same method applies to the FibDDs, since the reduction of a DT into a DD by the generalized BDD reduction rules [38] does not destroy nor diminish the information content of a DT. If the design is based on the FibDD for f, some multiplexers can be saved, since the number of nodes is reduced. However, these savings depend on the peculiar properties of f permitting reduction of the FibDT into the FibDD for f.

Example 4 Figure 4 shows the architecture for the realization of functions represented by the Fib$_1$DT(5). Table II shows the values of control inputs for the multiplexers.

Example 5 Figure 5 shows a realization for f in Example 3 from the Fib$_1$DD(5) in Fig. 3, assuming that the multiplexer-like modules FibS realizing the Fibonacci-Shannon expansion are provided.

Calculation Architecture for FWHT

The FW$_1$HT is calculated through FibDDs by using the basic transform matrix [14]

$$\mathbf{FW} = \begin{bmatrix} 1 & 0 & \sqrt{2} \\ 0 & \sqrt{2} & 0 \\ 1 & 0 & -\sqrt{2} \end{bmatrix}.$$

For the design purposes, we transfer this matrix into a

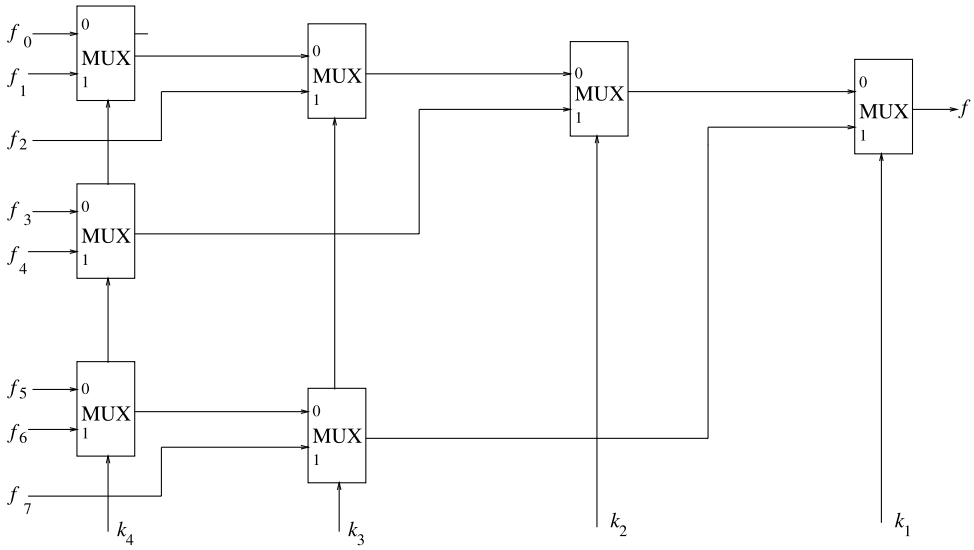

FIGURE 4 Architecture for realization of f represented by the Fib$_1$DT(5).

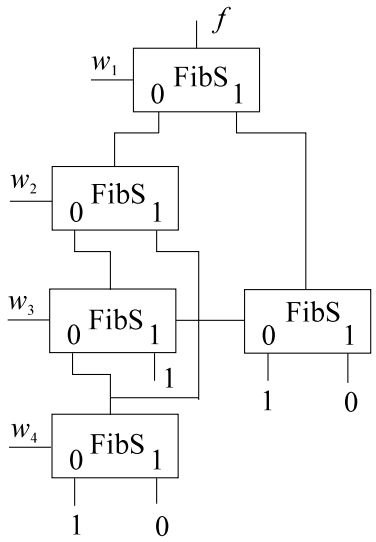

FIGURE 5 Direct realization from $\text{Fib}_1\text{DD}(5)$.

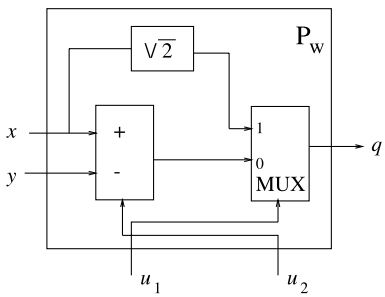

FIGURE 6 Basic FW_1HT-module.

matrix

$$\mathbf{FW}_r = \begin{bmatrix} 1 & 0 & 1 \\ 0 & \sqrt{2} & 0 \\ 1 & 0 & -1 \end{bmatrix}.$$

The remaining multiplications by $\sqrt{2}$ are realized as the multiplications at the input connections. The network calculates the values of the FW_1HT-spectrum $\mathbf{X}_{\text{FW},f}(w)$, $w \in \{0, \ldots, \phi_p(i) - 1$. Figure 6 shows the basic module for calculation of FWHT for $p = 1$. For the inputs x and y, and the output q, the FH_1HT-module performs the operation defined by

$$q = \begin{cases} x + y, & u_1 = 0,\ u_2 = 0, \\ x - y, & u_1 = 0,\ u_2 = 1, \\ \sqrt{2}x, & u_1 = 1,\ u_2 = 0. \end{cases}$$

At the moment w, the control variables $u_1(j) = k_{i-p-j}$, $u_2(j) = k_{i-p-j+1}$, where k_r is the value of the r-th bit in the Fibonacci code word w. Combination $u_1 = u_2 = 1$ is impossible for the properties of the Fibonacci 1-code.

Example 6 Figure 7 shows the architecture for calculation of FW$_1$HT of order $\phi_1(5)$. The values of control inputs are determined by the Fibonacci 1-code. For k(w) applied at the control inputs, the network generates the FW$_1$HT-coefficient $\mathbf{X}_{FW,f}(w)$.

Calculation Architecture for the Fibonacci-Haar Transform

Direct Realization from FibDDs

The FH_1HT is a local transform of order $\phi_p(i)$. It follows that in the FFT-like algorithm for FH_1HT, some of the

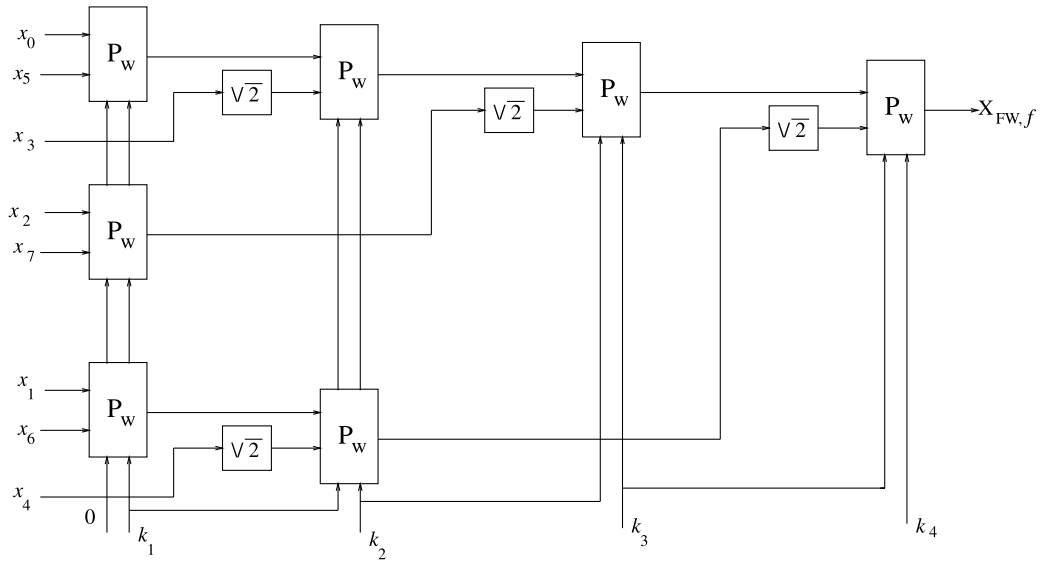

FIGURE 7 Architecture for calculation of FW_1HT-spectrum of order $\phi_1(5)$.

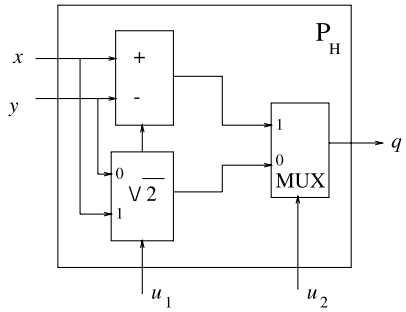

FIGURE 8 Basic FH_1HT-module.

FH_1HT-spectral coefficients are calculated up to a multiplicative constant in k steps, $1 \le k \le (i-p)$. These coefficients are forwarded from the k-th step to the output of the algorithm multiplied by $\sqrt{2}$ in each step. In calculation through FibDDs, this property implies that these coefficients are determined by processing of nodes at the k-th level in the FibDD. To get their final values, these coefficients should be multiplied by a weighting coefficient $z_k = (\sqrt{2})^{k-1}$, $k = 1, \ldots i-p$, This property requires a modification in the basic FH_1HT-module, compared to the FW_1HT-module. The basic FH_1HT-module perform the operation defined by

$$q = \begin{cases} x+y, & u_1 = 0,\ u_2 = 0, \\ x-y. & u_1 = 1,\ u_2 = 0, \\ \sqrt{2}x, & u_1 = 0,\ u_2 = 1, \\ \sqrt{2}y, & u_1 = 1,\ u_2 = 1. \end{cases}$$

The values of control inputs are determined by the Fibonacci p-code. However, the control input u_2 is driven through an OR circuit, which makes the combination $u_1 = u_2 = 1$ possible. At the moment w, the values of control inputs at the j-th level are determined by

$$u_1(j) = k_{i-p-j+1},$$

$$u_2 = (j) = \vee_{r=1}^{p-i-j} k_r.$$

Figure 8 shows the basic FH_1HT-module.

Example 7 Figure 9 shows the architecture for calculation of FH_1HT of order $\phi_1(5)$. The values of the sequence k_1, k_2, k_3, k_4 determining the values of control inputs in the FH_1HT-modules are taken from the Fibonacci 1-code as is shown in Table II. As in the case of FW_1HT, for $k(w)$ applied at the control inputs, the network generates the FH_1HT-coefficient $\mathbf{X}_{\mathrm{FH},f}(w)$.

Optimized Realization from FibDDs

In the procedure for calculation of the Fibonacci-Haar transform with the reduced number of calculations, we can use the following property of this transform, in the same way as that is done in the calculation of the Haar transform [32].

The FH_1HT-coefficients determined at the k-th level do not contribute to the values of FH-coefficients determined at the upper levels in the FibDD. In terms of FFT-like algorithms, these values appear as subsets of independent data at the input into the $(k+1)$-th step of the algorithm. In FibDDs methods, this property permits to determine the FH_1HT-spectrum by calculation at each node or the cross point with just the first values of two subfunctions represented by the subtrees rooted at the nodes to which point the outgoing edges of the processed nodes. In the network design from FibDDs, this property permits simplification of the basic FH_1HT-module into the reduced FH_1HT-module and some reconfiguration of the corresponding network for calculation of

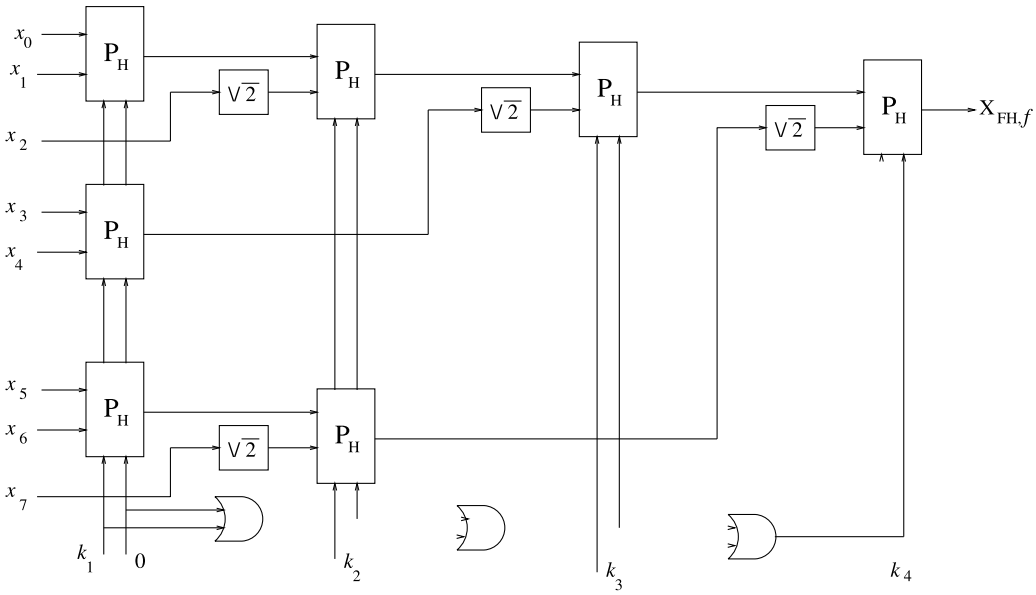

FIGURE 9 Architecture for calculation of FH_1HT-spectrum of order $\phi_1(5)$.

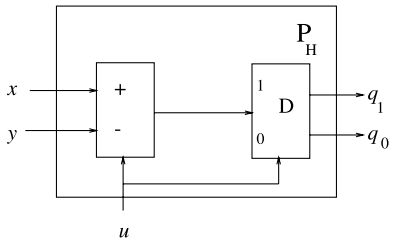

FIGURE 10 Reduced FH_1HTB-module.

FH_1HT-coefficients. Figure 10 shows the reduced FH_1HT-module, which performs the operation defined by

$$q_0 = x + y, \quad u = 0,$$

$$q_1 = x - y, \quad u = 1.$$

The control signals are not related to the Fibonacci 1-code, since the calculations at each node are performed just over two values taken from the calculations in the preceding levels in the FibDD. Therefore, the control signals are the clock impulses, each impulse corresponding to a level in the FibDD. Therefore, this network produces the FH_1HT-spectrum after $(i - p + 1)$ combinations of the values for the control inputs. In this way, by using properties of the FH_1HT-matrix, we perform the time optimization of the network.

Example 8 Figure 11 shows the architecture for calculation of the FH_1HT of order $\phi_1(5)$. Table III shows the values of control sequences $t_1, \ldots, t_4$.

TABLE III Control inputs for the optimized FH_1HT-network

k	t_1	t_2	t_3	t_4	t_5
0	0	1	0	0	0
1	0	0	1	0	0
2	0	0	0	1	0
3	0	0	0	0	1

SMALL DEPTH CIRCUIT SYNTHESIS FROM FibDDs

The chief drawback of networks derived directly from DDs is that their depth is equal to the number of variables in the functions realized.

In Ref. [17], a method of synthesizing multi-level logic networks with small depth directly from BDDs is proposed. The method is based upon a study of connections among nodes and cross points in the BDDs. These connections between levels are described by reachability matrices. The associativity of matrices is used in factorization of the matrix relation describing interconnections in the BDD for a given *f*. The method was extended to Kronecker decision diagrams (KDDs) in Ref. [16]. In Ref. [36], it is given a generalization of the approach in Ref. [16] to multiple-valued (MV) logic functions. In what follow, the method is generalized to small depth circuit synthesis from Fibonacci DDs.

Reachability Matrices

We describe interconnections among nodes and cross points in a FibDD by the reachability matrices defined as follows.

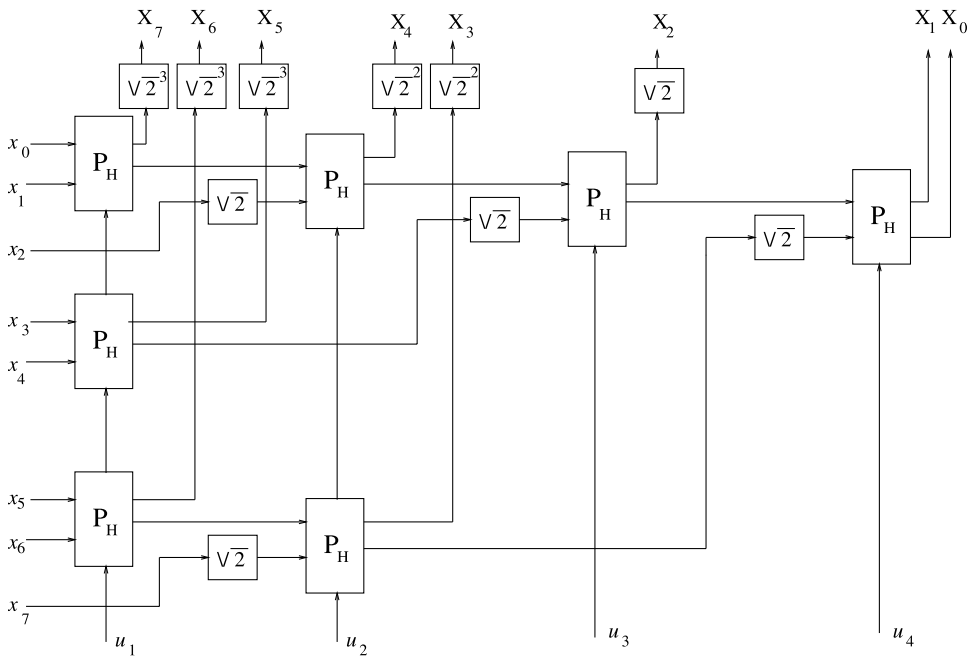

FIGURE 11 Architecture for calculation of the FH_1HT-spectrum with optimization of the calculation time.

Definition 3 *Denote by q_i and q_{i+1} the total number of nodes and cross points at the i-th level in a Fib$_p$DD (i). Denote by $l_{kj}^{i,i+1}$ the label at the edge connecting the k-th node or cross point at the i-th level to the j-th node or cross point at the (i + 1)-th level in the Fib$_p$DD(i).*

The reachability matrix $\mathbf{R}_{i,i+1}$ describing the connections between the i-th and (i + 1)-th level is defined as a $(q_i \times q_{i+1})$ matrix $\mathbf{R}_{i,i+1} = [r_{kj}^{i,i+1}]$, where

$$r_{kj}^{i,i+1} = \begin{cases} Sl_{kj}^{i,i+1}, & k \text{ and } j \text{ connected}, \\ 0, & \text{otherwise}. \end{cases}$$

Clearly, for Fib$_p$DD(i) representing single output functions which we consider in this paper, $\mathbf{R}_{1,2}$ and $\mathbf{R}_{n,n+1}$ have a single row. Recall that the first level in a Fib$_p$DD(i) corresponds to the root node, while $(i - p + 1)$-th level is determined by the constant nodes.

Definition 4 *The total reachability matrix for a Fib$_p$-DD(i) is*

$$\mathbf{R}_{1,n+1} = \mathbf{R}_{12} \cdot \mathbf{R}_{23} \cdot \ldots \cdot \mathbf{R}_{n,n+1},$$

where "·" denotes the matrix multiplication (M).

The reachability matrix $\boldsymbol{R}_{1,n+1}$ is the description of Fib$_p$DD(i) in the matrix form. By definition, the entries of $\boldsymbol{R}_{1,n+1}$ are cofactors of f. Thus, the same as Fib$_p$DD(i), $\boldsymbol{R}_{1,n+1}$ represents f, since the values of constant nodes are the values of f.

Statement 1 *Given f by a Fib$_p$DD (i). Then,*

$$f = \sum_{i=0}^{q_{n+1}} c_i \overset{1,n+1}{\underset{i}{r}}.$$

This statement follows directly from the spectral interpretation of DDs [35,38] and is explained by the following examples.

Example 9 *For Fib$_1$DT(5) in Fig. 2, the reachability matrices are given by*

$$\mathbf{R}_{12} = \begin{bmatrix} \overline{w}_1 & w_1 \end{bmatrix},$$

$$\mathbf{R}_{23} = \begin{bmatrix} \overline{w}_2 & w_2 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{R}_{34} = \begin{bmatrix} \overline{w}_3 & w_3 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \overline{w}_3 & w_3 \end{bmatrix},$$

$$\mathbf{R}_{45} = \begin{bmatrix} c_0\overline{w}_4 & c_1 w_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 \cdot c_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_3\overline{w}_4 & c_4 w_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_5\overline{w}_4 & c_6 w_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \cdot c_7 \end{bmatrix}.$$

The matrix $\mathbf{R}_{15} = \mathbf{R}_{12}\mathbf{R}_{23}\mathbf{R}_{34}\mathbf{R}_{45}$ describes interconnections in Fib$_1$DT(5). From Statement 1, after the matrix multiplication generating $\mathbf{R}_{15}$, we get the Fibonacci expression for *f*.

Example 10 *For Fib$_p$DD(5) in Fig. 3, the reachability matrices are given by*

$$\mathbf{R}_{12} = \begin{bmatrix} \overline{w}_1 & w_1 \end{bmatrix},$$

$$\mathbf{R}_{23} = \begin{bmatrix} \overline{w}_2 & w_2 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{R}_{34} = \begin{bmatrix} \overline{w}_3 w_3 & 0 & 0 & \\ 1 & 0 & 0 & 0 \\ 0 & 0 & \overline{w}_3 & w_3 \end{bmatrix}$$

$$\mathbf{R}_{45} = \begin{bmatrix} 1 \cdot \overline{w}_4 & 0 \cdot w_4 \\ 1 \cdot 1 & 0 \\ 1 \cdot 1 & 0 \\ 0 & 0 \cdot 1 \end{bmatrix} = \begin{bmatrix} \overline{w}_4 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

Thus, this Fib$_1$DD(5) represents f in the form of the Fibonacci expression

$$f = \overline{w}_1\overline{w}_2\overline{w}_3\overline{w}_4 + \overline{w}_1\overline{w}_2 w_3 + \overline{w}_1 w_2\overline{w}_4 + w_1\overline{w}_3.$$

In the matrix notation, this Fibonacci expression is given by $\boldsymbol{R}_{15} = \boldsymbol{R}_{12}\boldsymbol{R}_{23}\boldsymbol{R}_{34}\boldsymbol{R}_{45}$.

Circuit Realization

A network that realizes a function represented by the Fib$_p$DD is designed by the direct implementation of $\mathbf{R}_{1,n+1}$. As in the case of BDDs, KDDs, and KGFDDs realizations of switching and MV functions, we take advantages from the associativity properties of matrix multiplication to reduce the depth of the network. We consider the networks with the structure of a tree and perform reduction of the depth in the same way as in the case of corresponding BDDs, KDDs, and Kronecker Galois field decision diagrams (KGFDDs) realizations of switching and MV functions [16,17,36]. Therefore, further discussion will be omitted. Instead, for details we refer to Refs. [16,17,36], and the method is explained by the following example.

Example 11 The total reachability matrix for a function f represented by the Fib$_1$DT(5) in Fig. 2 can be written as

$$\mathbf{R}_{15} = (\mathbf{R}_{12}\cdot\mathbf{R}_{23})\cdot(\mathbf{R}_{34}\cdot\mathbf{R}_{45}) = \mathbf{R}_{13}\mathbf{R}_{35},$$

where

$$\mathbf{R}_{13} = \mathbf{R}_{12}\mathbf{R}_{23} = \begin{bmatrix} \overline{w}_1\overline{w}_2 & \overline{w}_1 w_2 & w_1 \end{bmatrix},$$

$$\mathbf{R}_{35} = \mathbf{R}_{34}\mathbf{R}_{45} = \begin{bmatrix} \overline{w}_3\overline{w}_4 & \overline{w}_3 w_4 & w_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \overline{w}_4 & w_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \overline{w}_3\overline{w}_4 & \overline{w}_3 w_4 & w_3 \end{bmatrix}.$$

Thanks to this representation, f can be realized by a network of the structure shown in Fig. 12. The blocks $\mathbf{R}_{ij}$ *realize labels at the edges between the levels i and j. The inputs in the circuits are 0, 1,* $\overline{w}_i$, w_i. *These blocks represent subfunctions in f. Blocks for matrix multiplication (M) realize the Fibonacci expression for f. The values of constant nodes in the Fib$_1$DT(5) for f can be alternatively used as inputs in the output block.*

Figure 13 shows a realization from Fib$_1$DT(5) with Programmable logic arrays (PLA) structures for realization of multiplications and additions in the Fibonacci expression for f.

Fibonacci DDs are derived by sharing isomorphic subtrees. Therefore, the networks produced from DDs instead of DTs, are often quite simpler.

Example 12 The total reachability matrix for f represented by the Fib$_1$DD(5) in Fig. 2 can be written as

$$\mathbf{R}_{15} = (\mathbf{R}_{12}\cdot\mathbf{R}_{23})\cdot(\mathbf{R}_{34}\cdot\mathbf{R}_{45}), = \mathbf{R}_{13}\mathbf{R}_{35},$$

where

$$\mathbf{R}_{13} = \begin{bmatrix} \overline{w}_1\overline{w}_2 & \overline{w}_1 w_2 & w_1 \end{bmatrix},$$

$$\mathbf{R}_{35} = \begin{bmatrix} \overline{w}_3\overline{w}_4 + w_3 & 0 \\ \overline{w}_4 & 0 \\ \overline{w}_3 & 0 \end{bmatrix},$$

since $c_0 = 1$, *and* $c_1 = 0$. *Figure 14 shows the realization for f derived from the factorization of the Fibonacci polynomial for f.*

CLOSING REMARKS

DDs can be directly converted into circuit realizations. We generalized and transferred these methods to Fibonacci interconnection topologies. We show that the optimization of the networks produced for calculation of spectral transforms is possible by exploiting peculiar properties a transform may possess.

A drawback of direct mapping a DD into a network is the property that depth of networks thus derived is equal to the number of levels in the DD for *f*. The method using reachability matrices describing DDs permits to reduce the depth of the networks produced from DDs. Therefore, we generalized this method to the circuit realizations from Fibonacci DDs. As in other DDs, the reachability matrices provide an efficient formalism for factorization of Fibonacci expressions assigned to the Fibonacci DDs. In this way, DD methods for circuit synthesis are completely generalized and transferred to Fibonacci interconnection topologies.

Acknowledgements

This paper was prepared during the stay of R.S. Stanković and M. Stanković at Int. Center for Signal Processing at Tampere University of Technology (TICSP). The support and the facilities provided by the TICSP are gratefully appreciated and acknowledged.

References

[1] Agaian, S., Astola, J. and Egiazarian, K. (1995) *Binary Polynomial Transforms and Nonlinear Digital Filters* (Marcel Dekker, New York).

[2] Agaian, S., Astola, J., Egiazarian, K., Kuosmanen, P. (1995) "Decompositional methods for stack filtering using Fibonacci *p*-codes", *Signal Processing* **41**(1), 101–110.

[3] Akers, S.B. (1978) "Binary decision diagrams", *IEEE Transactions on Computers* **C-27**(6), 509–516.

[4] Ashar, P., Devadas, S. and Keutzer, K. (1991) "Gate-delay-fault testability of multiplexer-based networks", *International Test Conference*, 887–896.

[5] Ashar, P., Devadas, S. and Keutzer, K. (1993) "Path-delay-fault testability of multiplexer-based networks", *Integration of VLSI Journal* **15**(1), 1–23.

[6] Becker, B. and Drechsler, R. (1995) "Synthesis for testability: Circuits derived form ordered Kronecker decision diagrams", *European Design and Test Conference*, 592.

[7] Bullmann, J. and Kebschull, U. (1996) "Multiple-domain logic synthesis", In: Sasao, T. and Fujita, M., eds, *Representations of Discrete Functions* (Kluwer Academic Publishers, Dordrecht), pp 211–232.

[8] Dechesler, R. and Becker, B. (1993) "Rapid prototyping of fully testable multi-level AND/EXOR networks", *Proceedings of IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, 126–133.

[9] Drechsler, R. and Becker, B. (1996) "OKFDDs-algorithms, applications and extensions", In: Sasao, T. and Fujita, M., eds, *Representations of Discrete Functions* (Kluwer Academic Publishers, Dordrecht), pp 163–190.

[10] Drechsler, R. and Becker, B. (1998) *Binary Decision Diagrams, Theory and Implementation* (Kluwer Academic Publishers, Dordrecht).

[11] Egiazarian, K. and Astola, J. (1996) "Discrete orthogonal transforms bassed on Fibonacci-type recursion", *Proceedings of IEEE Digtal Signal Processing Workshop (DSPWS-96), Norway*.

[12] Egiazarian, K., Gevorkian, D. and Astola, J. (1997) "Time-varying filter banks and multiresolution transforms based on generalized Fibonacci topology", *Proceedings of 5th IEEE International Workshop on Intelligent Signal Proceedings and Communication Systems*, S1651–S1654, Kuala Lumpur, Malaysia 11–13, November.

[13] Egiazarian, K., Astola, J. and Agaian, S. (1999) "Orthogonal transforms based on generalized Fibonacci recursions", *Proceedings of 2nd International Workshop on Spectral Techniques and Filter Banks*, Brandenburg, Germany, March 5–7.

[14] Egiazarian, K. and Astola, J. (1997) "On generalized Fibonacci cubes and unitary transforms", *Applicable Algebra in Engineering, Communication and Computing* **AAECC 8**, 371–377.
[15] H.Md., Hasan Babu and Sasao, T. (1998) "Design of multiple-output networks using time domain multiplexing and shared multi-terminal multiple-valued decision diagrams", *Proceedings of 28th International Symposium on Multiple-Valued Logic*, Fukuoka, Japan, May 27–29.
[16] Hengster, H., Drechsler, R., Eckrich, S., Pfeiffer, T. and Becker, B. (1996) "AND/EXOR based synthesis of testable KFDD-circuits with small depth", *Proceedings of Asian Test Symposium.*
[17] Ishiura, N. (1992) "Synthesis of multi-level logic circuits form binary decision diagrams", *SASIMI*, 74–83.
[18] Jiang, F.-S., Horng, S.-J. and Kao, T.-W. (1997) "Embeding of generalized Fibonacci cubes in hypercubes with faulty nodes", *IEEE Transactions on Parallel and Distributed Systems* **8**(7), 727–737.
[19] Karpovsky, M.G. (1976) *Finite Orthogonal Series in the Design of Digital Devices* (Wiley and JUP, New York and Jerusalem).
[20] Kebschull, U., Schubert, E. and Rosenstiel, W. (1992) "Multilevel logic synthesis based on functional decision diagrams", *European Conference on Design Automation*, 43–47.
[21] Le, V.V, Besson, T., Abbara, A., Brasen, D., Bogushevitsh, H., Saucier, G. and Crastes, M. (1995) "ASIC prototyping with area oriented mapping for ALTERA/FLEX devices", *SASIMI*, 176–183.
[22] McKenzie, L., Xu, L. and Almaini, A. (1993) "Graphical representations of generalized Reed-Muller Expansions", In: Kebschull, U., Schubert, E. and Rosenstiel, W., eds, *Proceedings of IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, September 16–17, Hamburg, Germany, pp 181–187.
[23] Perkowski, M.A., Jozwiak, L. and Drechsler, R. (1997) "New hierarchies of AND/EXOR trees, decision diagrams, lattice diagrams, canonical forms, and regular layouts", *Proceedings of IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design Reed-Muller '97*, 115–132, Oxford, England.
[24] Perkowski, M.A., Pierzchala, E. and Drechsler, R. (1997) "Ternary and quaternary lattice diagrams for linearly-independent logic, multiple-valued logic, and analog synthesis", *Proceedings of International Conference on Information, Communication and Signal Processing, ICICS '97*, 269–273, Singapore, September 9–12.
[25] Sarabi, A., Ho, P.F., Iravani, K., Daasch, W.R. and Perkowski, M.A. (1993) "Minimal multi-level realization of switching functions based on Kronecker functional decision diagrams", *Proceedings of International Workshop on Logic Synthesis*, 3a1–3a6, Lake Tahoe, CA, USA.
[26] Sasao, T. (1993) "FPGA design by generalized functional decomposition", In: Sasao, T., ed, *Logic Synthesis and Optimization* (Kluwer Academic Publishers, Dordrecht), pp 233–258.
[27] Sasao, T. and Butler, J.T. (1994) "A design method for look-up table type FPGA by pseudo-Kronecker expansions", *Proceedings of 24th International Symposium on Multiple-Valued Logic*, 97–106, Boston, Massachusetts, May 25–27.
[28] Sasao, T., Hamachi, H., Wado, S. and Matsuura, M. (1995) "Multi-level logic synthesis based on pseudo-Kronecker decision diagrams and local trasnformation", *Proceedings of IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design, Reed-Muller '95*, 152–160.
[29] Sasao, T. and Fujita, M. (eds) (1996) In: *Representations of Discrete Functions* (Kluwer Academic Publishers, Dordrecht).
[30] Schaefer, I., Perkowski, M.A. and Wu, H. (1993) "Multilevel logic synthesis for cellular FPGAs based on orthogonal expansions", In: Kebschull, U., Schubert, E. and Rosenstiel, W., eds, *Proceedings of IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, September 17–19, Hamburg, Germany, pp 42–51.
[31] Stakhov, A.P. (1979) *Algorithmic Measurement Theory* (Znanie, Moscow), No. 6, 64 p, in Russian.
[32] Stanković, M., Janković, D. and Stanković, R.S. (1996) "Efficient algorithms for Haar spectrum calculation", *Scientific Review* **21–22**, 171–182.
[33] Stanković, R.S. (1995) "Functional decision diagrams for multiple-valued functions", *Proceedings of 25-th International Symposium on Multiple-Valued Logic*, 284–289.
[34] Stanković, R.S. (1997) "Functional decision diagrams for multiple-valued functions", *Multiple-Valued Logic Journal.*
[35] Stanković, R.S. (1998) *Spectral Tansform Decision Diagrams in Simple Questions and Simple Answers* (Nauka, Belgrade).
[36] Stanković, R.S. and Drechsler, R. (1997) "Circuit design from Kronecker Galois field decision diagrams for multiple-valued logic functions", *Proceedings of IEEE International Symposium on Multiple-Valued Logic*, 275–280, Antigonish, Nova Scotia, Canada.
[37] Stanković, R.S. and Sasao, T. (1998) "Decision diagrams for representation of discrete functions: uniform interpretation and classification", *Proceedings of ASP-DAC '98*, Yokohama, Japan, February 13–17.
[38] Stanković, R.S., Sasao, T., Moraga, C. "Spectral transform decision diagrams", in: [29], 55–92.
[39] Stanković, M., Stanković, R.S., Astola, J.T. and Egiazarian, K. (2000) "Fibonacci decision diagrams and spectral transform Fibonacci decision diagrams", *Proceedings of 30th International Symposium on Multiple-Valued Logic*, Portland, Oregon, USA, May 23–26.
[40] Wu, H., Perkowski, M.A. and Zhuang, N. (1993) "Synthesis of multiplexer directed-acyclic-graph network with application to FPGA and BDDs", *Proceedings of International Workshop on Logic Synthesis*, Tahoe City, USA, May 23–26, 8d/1–8.

ADDENDUM

Fibonacci *p*-numbers

Definition 5 A sequence $\phi(n)$ is the Fibonacci sequence if for each $n \geq 1$,

$$\phi(n) = \phi(n-1) + \phi(n-2),$$

with initial values $\phi(0) = 1$, $\phi(n) = 0, n < 0$. Elements of this sequence are the Fibonacci numbers.

A generalization of Fibonacci numbers is given in Refs. [13,31] as follows.

Definition 6 A sequence $\phi_p(i)$ is the generalized Fibonacci p-sequence if

$$\phi_p(i) = \begin{cases} 0, & i < 0, \\ 1, & i = 0, \\ \phi_p(i-1) + \phi_p(i-p-1), & i > 0. \end{cases}$$

Elements of this sequence are the generalized Fibonacci p-numbers.

Example 13 Table AI shows the generalized Fibonacci p-numbers for $p = 0, 1, 2,$ and 3, and $i = 0, 1, \ldots, 9$.

Fibonacci *p*-codes

The Fibonacci *p*-representation of a natural number B is defined as

$$B = \sum_{i=p}^{n-1} a_i \phi_p(i).$$

The sequence $\mathbf{a} = (a_{n-1}, \ldots, a_p)_p$ is the Fibonacci *p*-code for B [13]. Since with thus defined weighting coefficients,

TABLE AI Generalized Fibonacci numbers

$\phi_p(i)$	$i=1$	1	2	3	4	5	6	7	8	9
$p=0$	1	2	4	8	16	32	64	128	256	512
1	1	1	2	3	5	8	13	21	34	55
2	1	1	1	2	3	4	6	9	13	19
3	1	1	1	1	2	3	4	5	7	10

a given number B may be represented by few different code sequences, the normal unique Fibonacci p-code is introduced by the requirement

$$B = \phi_p(n-1) + m,$$

where $\phi(n-1)$ is the greatest Fibonacci p-number smaller or equal to B, and $0 \le m < \phi_p(n-p-1)$.

Contracted Fibonacci p-codes

The following property of normal Fibonacci p-codes permit definition of the contracted Fibonacci p-codes.

Lemma 1 [13] In the normal Fibonacci p-code for a given number B, if $a_i = 1$, then $a_{i-1} = a_{i-2} = \cdots = a_{i-p} = 0$.

Utilizing this property, the contracted Fibonacci p-code is defined by deleting p zeros after each 1 in the Fibonacci p-code for B, except for the rightmost 1, in which case we should delete $\max(i,p)$ zeros where there are i zeros to the right of the rightmost 1.

Generalized Fibonacci Transforms

Classical discrete orthogonal transforms, for example, the Walsh transform, the Haar transform, etc., are defined in the matrix notation by the square matrices of orders 2^n [1,19]. Some extensions and generalizations are done for MV functions by using the square matrices of orders q^n, where q is an integer. In Ref. [13], the definition of some discrete orthogonal transforms is generalized into transforms defined by transform matrices whose orders are equal to the arbitrary Fibonacci p-numbers. These transforms are denoted as the generalized Fibonacci transforms.

Let f be a function defined in $\phi_p(i)$ points. We assume that f is given by a sequence $x = \{x(0), \ldots, x(\phi_p(i)-1)\}$. We denote by $\mathbf{T}_p(i)$ a transform matrix defining a particular Fibonacci p-transform of order $\phi_p(i)$.

Definition 7 The Fibonacci p-spectrum for a function f given by the vector $\boldsymbol{F}$*, with respect to* $\boldsymbol{T}_p(i)$ *is defined as a sequence*

$$\mathbf{X}_{T,f} = \mathbf{T}_p(i)\mathbf{F}.$$

Fibonacci-Walsh Hadamard Transform

Definition 8 The Fibonacci-Walsh p-transform in the Hadamard ordering (FW_pHT) is defined by the transform matrix determined as

$$\mathbf{W}^{(p,n)} = \begin{bmatrix} \overline{\mathbf{W}}^{(p,n-1)} & \sqrt{2}\mathbf{W}^{(p,n-1)}_{\wedge} & \overline{\mathbf{W}}^{(p,n-1)} \\ (\sqrt{2})^p \mathbf{W}^{(p,n-p-1)} & \mathbf{0} & -(\sqrt{2})^p \mathbf{W}^{(p,n-p-1)} \end{bmatrix},$$

for $n > p$, *and*

$$\mathbf{W}^{(p,m)} = [1], \quad \text{for } m \le p, \quad \mathbf{W}^{(p,p+1)} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

where $\overline{\mathbf{W}}^{(p,n-1)}$ *and* $\mathbf{W}^{(p,n-1)}_{\wedge}$ *are the rectangular matrices formed from the matrix* $\mathbf{W}^{(p,n-1)}$ *by taking its first* $\phi_p(n-p-1)$ *columns, and its last* $\phi_p(n-1) - \phi_p(n-p-1)$ *columns, respectively.*

Example 14 The FWHT for $\phi_1(5)$ *is given by the matrix*

$$\mathbf{W}^{(1,5)} = \begin{bmatrix} 1 & \sqrt{2} & \sqrt{2} & \sqrt{2} & 2 & 1 & \sqrt{2} & \sqrt{2} \\ 1 & -\sqrt{2} & \sqrt{2} & \sqrt{2} & -2 & 1 & -\sqrt{2} & \sqrt{2} \\ \sqrt{2} & 0 & -2 & 2 & 0 & \sqrt{2} & 0 & -2 \\ \sqrt{2} & \sqrt{2} & 0 & -2 & -2 & \sqrt{2} & \sqrt{2} & 0 \\ \sqrt{2} & -\sqrt{2} & 0 & -2 & 2 & \sqrt{2} & -\sqrt{2} & 0 \\ \sqrt{2} & 2 & \sqrt{2} & 0 & 0 & -\sqrt{2} & -2 & -\sqrt{2} \\ \sqrt{2} & -2 & \sqrt{2} & 0 & 0 & -\sqrt{2} & 2 & -\sqrt{2} \\ 2 & 0 & -2 & 0 & 0 & -2 & 0 & 2 \end{bmatrix}.$$

For a function f given by a vector $\boldsymbol{F} = [1,0,1,0,1,1,1,1]^{\mathrm{T}}$, *the FWHT-spectrum for f is given by* $\boldsymbol{X}_{FW,f} = \{4+3\sqrt{2}, \sqrt{2}, -4+2\sqrt{2}, -2+3\sqrt{2}, 2+\sqrt{2}, -2, 2, 0\}$.

Fibonacci-Haar Transform

The Fibonacci-Haar p-transform in the Hadamard ordering (FH_pHT) is defined by the transform matrix determined as

$$\mathbf{H}^{(\mathrm{Had},p,n)} = \begin{bmatrix} \overline{\mathbf{H}}^{(\mathrm{Had},p,n-1)} & \sqrt{2}\mathbf{H}^{(\mathbf{Had},p,n-1)}_{\wedge} & \overline{\mathbf{H}}^{(\mathrm{Had},p,n-1)} \\ (\sqrt{2})^{n-p-1}\mathbf{I} & \mathbf{0} & -(\sqrt{2})^{n-p-1}\mathbf{I} \end{bmatrix},$$

for $n > p$, and the initial matrices are defined as in Definition 5, where $\overline{\mathbf{H}}^{(\mathrm{Had},p,n-1)}$ and $\hat{\mathbf{H}}^{(\mathrm{Had},p,n-1)}$ are the rectangular matrices formed from the matrix $\mathbf{H}^{(\mathrm{Had},p,n-1)}$ by taking its first $\phi_p(n-p-1)$ columns, and its last $\phi_p(n-1) - \phi_p(n-p-1)$ columns, respectively, and $\mathbf{I}$ is the identity matrix of order $\phi_p(n-p-1)$.Note that in Ref. [13], the FFT-like algorithm for the Fibonacci-Haar transform is derived for the Hadamard ordering. Therefore, it is assumed that the input sequences is given in the bit-reverse ordering with respect to the Fibonacci p-code,

and the Fibonacci-Haar spectrum is obtained in the direct ordering.

This convention will be adapted also in the further considerations of the Fibonacci-Haar spectrum in this paper.

Example 15 The Fibonacci-Haar transform for $\phi_1(5)$ is given by the transform matrix determined as

$$\mathbf{H}^{(\mathrm{Had},1,5)} = \begin{bmatrix} 1 & \sqrt{2} & \sqrt{2} & \sqrt{2} & 2 & 1 & \sqrt{2} & \sqrt{2} \\ 1 & -\sqrt{2} & \sqrt{2} & \sqrt{2} & -2 & 1 & -\sqrt{2} & \sqrt{2} \\ \sqrt{2} & 0 & -2 & 2 & 0 & \sqrt{2} & 0 & -2 \\ 2 & 0 & 0 & -2\sqrt{2} & 0 & 2 & 0 & 0 \\ 0 & 2 & 0 & 0 & -2\sqrt{2} & 0 & 2 & 0 \\ 2\sqrt{2} & 0 & 0 & 0 & 0 & -2\sqrt{2} & 0 & 0 \\ 0 & 2\sqrt{2} & 0 & 0 & 0 & 0 & -2\sqrt{2} & 0 \\ 0 & 0 & 2\sqrt{2} & 0 & 0 & 0 & 0 & -2\sqrt{2} \end{bmatrix}.$$

For f in Example 14, the function f in the bit-reverse ordering with respect to the Fibonacci p-code is given by $\boldsymbol{F}' = [1,1,0,1,1,0,1,1]^T$. The multiplication of $\boldsymbol{H}^{(Had,1,5)}$ with $\boldsymbol{F}'$ produces the Fibonacci-Haar spectrum in the direct ordering as $\boldsymbol{X}_{FH,f} = \{3+4\sqrt{2}, -1, \sqrt{2}, 2-2\sqrt{2}, 4-2\sqrt{2}, 2\sqrt{2}, 0, -2\sqrt{2}\}$.

Authors' Biographies

Radomir S. Stanković received B.E. degree in Electronic Engineering from Faculty of Electronics, University of Niš, in 1976, and M.Sc., and Ph.D. degrees in Applied Mathematics from Faculty of Electrical Engineering, University of Belgrade, in 1984, and 1986, respectively.

He was with High School of Electrotechnic, Niš, from 1976 to 1987. From 1987 to date he is with Faculty of Electronic, Niš. Presently, he is a Professor teaching Logic Design.

His research interests include switching theory and multiple-valued logic, signal processing and spectral techniques.

Jaakko Astola was born in Helsinki, Finland, in 1949. He received the B.Sc., M.Sc., Licenciate, and Ph.D. degrees in Mathematics from Turku University, Turku, Finland, in 1972, 1973, 1975, and 1978, respectively.

From 1976 to 1977, he was a Research Assistant at the Research Institute for Mathematical Sciences of Kyoto University, Kyoto, Japan. Between 1979 and 1987, he was with the Department of Information Technology, Lappeenraanta University of Technology, Lappeenraanta, Finland, holding various teaching positions in mathematics, applied mathematics, and computer science. From 1988 to 1993, he was an Associate Professor in applied mathematics at Tampere University, Tampere, Finland. Currently, he is a Professor of digital signal processing at Tampere University of Technology, the head of Signal Processing Laboratory and the Director of the Tampere International Center in Signal Processing (TICSP). His research interests include signal processing, coding theory, and statistics.

Milena Stanković received B.E. degree in Electronic Engineering in 1976, and M.Sc., and Ph.D. degrees in Computer Science in 1982 and 1988 from Faculty of Electronics, University of Niš.

She was with High School of Electrotechnic, Niš, from 1976 to 1978. From 1978 to date she is with Faculty of Electronic, Niš. Presently, she is a Professor teaching Programming Languages and Compilers. Her research interests include programming languages, spectral techniques, and signal processing.

Karen Egiazarian was born in Yerevan, Armenia, in 1959. He received the M.Sc. degree in Mathematics from Yerevan State University in 1981 and the Ph.D. degree in Physics and Mathematics from Moscow State University, Russia, in 1986. In 1994, he received the degree of Doctor of Technology from the Tampere.

University of Technology, Tampere, Finland. He has been Senior Researcher at the Department of Digital Signal Processing of the Institute of Information Problems and Automation, National Academy of Sciences of Armenia. Since 1996 he was the Assistant Professor at the Signal Processing Laboratory at Tampere University of Technology, where he is currently a Professor leading the group of Spectral and Algebraic Methods in DSP. His research interests are in areas of applied mathematics, signal processing and digital logic.