

REMARKS ON THE NUMBER OF LOGIC NETWORKS WITH SAME COMPLEXITY DERIVED FROM SPECTRAL TRANSFORM DECISION DIAGRAMS

Radomir S. Stanković *Mark G. Karpovsky*¹

Dept. of Computer Science, Faculty of Electronics, Beogradska 14,
18 000 Niš, Serbia

¹ Dept. of Electrical and Computer Engineering, Boston University, 8 Saint Marry's Street
Boston, Ma 02215, USA

Abstract

Decision diagrams (DDs) for representation of discrete functions permit a direct technology mapping into multi-level logic networks. In these networks, the number of logic elements is equal to the number of non-terminal nodes in DDs. Complexity of interconnections is proportional to the number of paths from the root node to the constant nodes. Therefore, the complexity of the network can be estimated through the basic characteristics of DDs. For a given function f , different DDs may result into networks of different complexity. Complexity of a network derived from a DD is expressed through the number of non-terminal nodes, the width, and the number of paths from the root node to the constant nodes. When realizations for all functions for a given number of variables are considered, the problem relates to the classification of switching functions. In this settings, DDs defined with respect to different spectral transforms may result in a different number of networks required to realize the representative functions for classes of switching functions.

This paper discusses the number of different multi-level logic networks of the same complexity derived from different word-level decision diagrams for functions of three and four variables. These networks can be used as basic building blocks in realization of

functions of an larger arbitrary number of variables. For uniform realizations, it might be useful to have a small number of different basic modules.

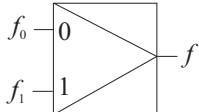
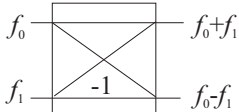
1 MULTI-LEVEL LOGIC NETWORKS

Multi-level logic networks permit realizations of switching functions with fewer circuits and reduced interconnections compared to two-level logic networks. However, designs of multi-level logic networks are far more complex than those of two-level logic networks. As is noted in [9], unlike two-level networks, where there are minimization algorithms, for multi-level networks there are no established automation design algorithms and the designs are done by using combinations of ad hoc methods.

The optimization of multi-level networks may be directed towards reduction of the number of elements, the number of levels, i.e., timing optimization, reduction of interconnections, and the reduction of the complexity of the design procedure.

Decision diagrams (DDs) for representation of discrete functions can be directly translated into n -level networks, n - the number of variables. Therefore, DDs provide for simple design procedure, at the price that the number of levels is equal to the number of

Table 1: Expansion rules and their realizations.

Shannon	Walsh
$f = \bar{x}_i f_0 \oplus x_i f_1$	$f = \frac{1}{2}(w_0 + (1 - 2x_i)w_1)$ $w_0 = (f_0 + f_1), w_1 = (f_0 - f_1)$
	

variables, which can be a disadvantage. For that reason, methods for design of small depth networks from DDs by exploiting associativity of matrix representations of interconnections have been proposed [4]. These methods are extended to multiple-valued logic functions and functions in Fibonacci topologies, see for example, [18].

For simplicity of the design procedures, this approach is especially efficient for FPGA technology mapping [3], [10], [13]. Due to that, DD-methods have been incorporated into commercial FPGA synthesis systems [6].

1.1 Design from DD

The design from DD is quite simple and consists of traversing of the DD in topological order and replacement of all the non-terminal nodes by the corresponding elements realizing the expansion rules used in definition of the DD. It is assumed that a fixed library of elements related to the expansion rules in DDs is provided. In this paper, we consider Binary DDs (BDDs) [1], Haar spectral transform DDs (HSTDDs) [17], and Walsh DDs (WDDs) [16].

Example 1 *Fig. 1 shows BDD, HSTDT, and WDT for $n = 3$. Table 1 shows the expansion rules used in these DDs and the corresponding elements to realize these rules.*

DDs can be described by their basic characteristics

1. Number of non-terminal nodes (ntn),

Table 2: Characteristics of DDs and logic networks.

DD	Logic network
Non-terminal nodes (ntn)	Circuits
Constant nodes (cn)	Inputs
Size $s = ntn + cn$	Area
Width (w)	Area
Paths t	Interconnections

2. Number of constant nodes (cn),
3. Size $s = ntn + cn$,
4. Width (w) defined as the maximum number of nodes per a level in DD,
5. Number of paths t .

These characteristics determine the shape of DDs. Therefore, in what follows, we assume that the shape of a DD is represented through the number of non-terminal nodes, the width, and the number of paths. In circuits synthesis from DDs, each non-terminal node corresponds to a basic element realizing the mapping performed at each node to assign a given f to the DD. In BDDs, the Shannon expansion rule is used for all the nodes. In WDDs [16], the Walsh expansion rule is used for all the nodes. In HSTDDs [17], the Walsh expansion rule is used at the leftmost nodes in the decision tree (DT), and the Shannon expansion rule for the other nodes. It is assumed that these characteristics of DDs correspond to the basic characteristics of n -level networks, thus, express their complexity. Table 2 shows the correspondence between the basic characteristics of DDs and those of logic networks.

1.2 Complexity of logic networks

Shannon proved in [14] that the least number of contacts $L(n)$ to realize any switching function of n -variable by an arbitrary contact network is $\frac{2^n}{n} \leq L(n) \leq 4\frac{2^n}{n}$ as $n \rightarrow \infty$. Lupanov [8] improved the upper bound to $L(n) \sim \frac{2^n}{n}$ as $n \rightarrow \infty$. In [9], it is shown that when n is sufficiently large, an arbitrary

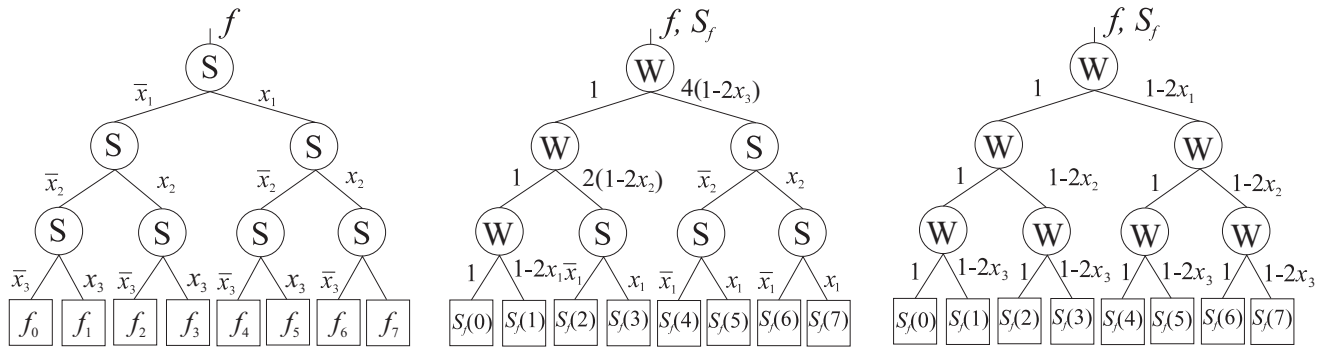


Figure 1: BDT, HSDT, and WDT for $n = 3$.

n -variable switching function is realized by at most $\frac{2^n}{n}(2 + \epsilon)$ copies of (1×2) multiplexers, where ϵ is an arbitrary small positive number such that $0 < \epsilon < 1$. Since the multiplexers realize the Shannon expansion rule, the similar complexity applies to BDDs. In this setting, it is shown in [7] that the average complexity of BDDs is $O(C \frac{2^n}{n})$. It is believed that these results extends to arbitrary two input, single output circuits. Therefore, these results may be applied to other DDs defined by using different expansion rules related to basic spectral transform matrices and realized by the corresponding two input modules.

For a given function f , different DDs produce the networks of different complexity. Conversely, a number of different functions can be realized by a given DD by manipulation with labels at the edges and values of constant nodes. The probably simplest example is realization of a function f and its logic complement \bar{f} by BDDs, where it is sufficient to permute the logic values 0 and 1 of the constant nodes in $BDD(f)$ to derive $BDD(\bar{f})$.

In this paper, we are interested in the number of different multi-level networks required for realization of switching functions of $n = 3$ and $n = 4$ variables by BDDs, HSTDDs, and WDDs. These networks can be used as basic building blocks in realization of functions of a larger number of variables.

2 CLASSIFICATION OF SWITCHING FUNCTIONS

Classification of switching functions splits the set of all switching functions of a given number of variables into the classes of similar functions, with similarity expressed through a collection of classification rules. Chief goal of a classification is to put functions similar with respect to the specified rules into the same class. Functions within the same class may be converted from one to another by using the classification rules. Each class Q is represented by a representative function f_Q . In circuit synthesis, that means all functions in the class may be realized by the network for the representative function f_Q after a simple manipulation corresponding to the application of the classification rules in the order reverse to that in converting a given f to f_Q .

The number of different networks to realize all functions of a given number of variables is equal to the number of different classes. The LP-classification is the most compact, and provides the fewer number of classes.

LP-classification of Switching Functions

LP-classification of switching functions has been adapted to AND-EXOR representations and provides fewer number of equivalence classes compared

to NPN classification [12].

Definition 1 *LP-classification of switching functions is performed by using the following transformation rules.*

1. $LP_0(f) = \bar{x}_i f_0 \oplus x_i f_1$, Identity
2. $LP_1(f) = \bar{x}_i f_0 \oplus 1 \cdot f_1$, $x_i \rightarrow 1$,
3. $LP_2(f) = 1 \cdot f_0 \oplus x_i f_1$, $\bar{x}_i \rightarrow 1$,
4. $LP_3(f) = x_i f_0 \oplus \bar{x}_i f_1$, $x_i \rightarrow \bar{x}_i$,
5. $LP_4(f) = x_i f_0 \oplus 1 \cdot f_1$, $x_i \rightarrow 1 \rightarrow \bar{x}_i \rightarrow x_i$,
6. $LP_5(f) = 1 \cdot f_0 \oplus \bar{x}_i f_1$, $\bar{x}_i \rightarrow 1 \rightarrow x_i \rightarrow \bar{x}_i$.

Definition 2 *LP-equivalence relation is defined recursively as follows*

1. $f \equiv f$,
2. If $f_1 \equiv f(x_1, x_2)$ and $f_2 \equiv f(x_1, x_2)$, then $f_1 \equiv f_2$.
3. Denote by F an ESOP for a function $f(x_1, \dots, x_n)$, and by G an ESOP derived from F by a LP-transform over F . If g is a function represented by G , then $f \equiv g$.

Example 2 *Functions $f(x, y) = x \oplus y$ and $g(x, y) = 1 \oplus xy$ are LP-equivalent functions.*

3 NUMBER OF DIFFERENT NETWORKS

We are interested to determine the number of different networks to realize the representative functions in LP-classification for $n = 3$ and $n = 4$ derived from BDDs, HSTDDs, and WDDs. In the case of WDDs, the values of switching functions are encoded as $(0, 1) \rightarrow (1, -1)$ to take advantages of the properties of Walsh spectrum in this encoding. We determine the basic characteristics of these DDs, the size, the width, and the number of paths, and enumerate the number of different DDs with respect to these characteristics.

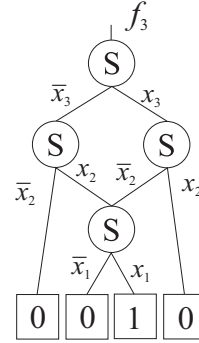


Figure 3: BDD for f_3 with reordered variables.

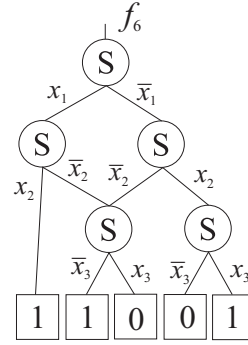


Figure 4: BDD for f_6 with permuted edges.

3.1 Binary DD

Example 3 *Fig. 2, shows BDDs for LP-representants for $n = 3$. It follows that BDDs of six different shapes are required to represent these functions. Fig. 3 shows that the function f_3 may be represented by a BDD with smaller size by optimization based on reordering of variables. Reordering of variables does not reduce the size neither change the shape of any other BDD in this example. However, Fig. 4 shows that f_6 can be represented by the BDD of the same shape as for f_4 if permuted edges are allowed. In this figure, the root node and the leftmost node at the level corresponding to the variable x_2 are nodes with permuted edges.*

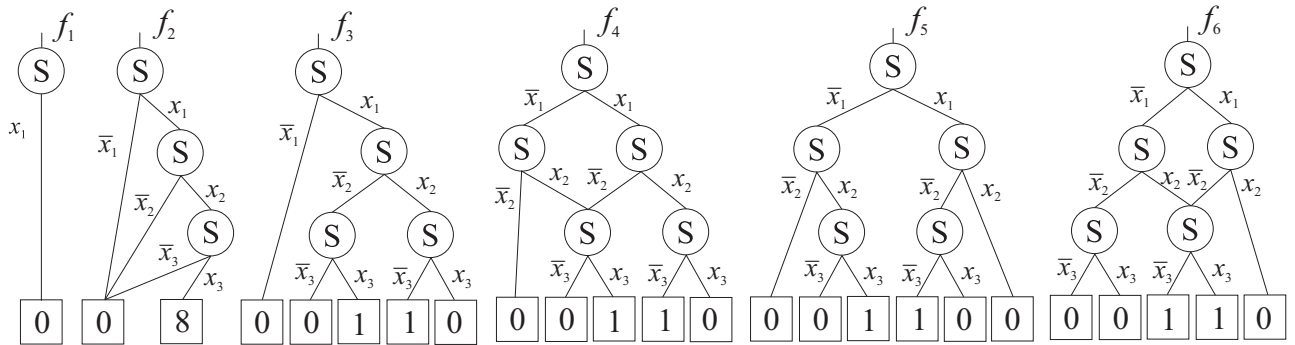


Figure 2: BDDs for LP-representants for $n = 3$.

It should be noted that $\text{BDD}(f_4)$ and $\text{BDD}(f_6)$ have the same number of non-terminal nodes, the width and the number of paths.

Table 3 shows the number of different BDDs for LP-representative functions for $n = 3$ and Table 4 for $n = 4$. In these tables, and in what follows, when the width is considered, we show i/j , where i is the number of nodes at the level j in the DD. In Table 4, for simplicity, we show the BDD classes with few LP-functions. Each of the other LP-representative functions forms a separate singleton BDD class. Therefore, there is a total of 21 BDD classes, which implies 21 different networks derived from BDDs to realize all LP-representative functions for $n = 4$.

Table 3: BDD classes for $n = 3$.

	ntn	w	t	f
1.	1	1/1	1	1
2.	5	2/2	7	2
3.	6	3/3	7	3, 4
4.	7	4/3	8	5
5.	7	4/3	6	6

3.2 Haar Spectral Transform DD

Haar spectral transform DDs (HSTDDs) represent functions in terms of the discrete Haar spectra [5].

Table 4: BDD classes for $n = 4$.

	ntn	w	t	f
1.	5	2/4	6	3, 5
2.	8	3/3	11	10, 14, 20, 21, 23
3.	8	3/3	10	13, 29
4.	7	2/2	12	17, 18, 19
5.	8	3/3	13	26, 30

The reduction in BDDs is possible due to constant or equal subvectors in the truth-vector \mathbf{F} for a given function f . Therefore, compact representations are derived by exploiting properties of the functions represented. In HSTDDs, the reduction is possible due to constant and equal subvectors in the Haar spectrum for f . Therefore, advantages are taken from both properties of f and the properties of Haar functions, when their waveforms may relate to the constant or equal subvectors in \mathbf{F} , which results in zero or constant subvectors in the Haar spectrum. For some functions, HSTDDs may be of smaller size than for other DDs, since for any class of DDs there are functions for which this particular class provides for the simplest DDs. This is a justification to consider and use in practice different DDs.

Table 5 and Table 6 show the number of different HSTDDs for $n = 3$ and $n = 4$. In Table 6, we show the classes with more than a single element. There

are in total 18 HSTDD classes, which means, 18 different networks derived from HSTDDs are required to realize all LP-representative functions for $n = 4$.

Table 5: HSTDD classes for $n = 3$.

	ntn	w	t	f
1.	1	1/1	1	1
2.	5	2/2	7	2
3.	6	3/3	7	3
4.	7	4/3	8	4, 5, 6

Table 6: HSTDD classes for $n = 4$.

	ntn	w	t	f
1.	11	4/3	14	7, 12, 29
2.	12	5/4	15	8, 10, 11, 15, 25
3.	12	5/4	14	14, 18
4.	12	6/4	13	16, 23
5.	13	6/4	15	17, 19, 21, 27
6.	13	6/4	16	24, 26

3.3 Walsh DD

In WDDs, the advantages are taken from properties of the Walsh spectra of switching functions in $(0, 1) \rightarrow (1, -1]$ encoding. Due to that, a number of different functions can be represented by WDDs of the same shape, similar as a given function f and its logic complement \bar{f} can be represented by BDDs that differ in the order of values of constant nodes. In [15], it is shown that for $n = 3$, the total of 63 switching functions can be represented by WDDs of the same shape and having three non-terminal nodes.

Table 7 shows the number of different WDDs for LP-representative functions for $n = 3$. Table 8 shows the number of different WDDs for LP-representative functions for $n = 4$. In WDDs, there are four singleton WDD classes. Therefore, there are 14 different networks derived from WDDs to realize all LP-representative functions.

Table 7: WDD classes for $n = 3$.

	ntn	w	t	f
1.	3	1/1	4	1, 4
2.	6	3/3	8	2, 3, 5
3.	4	2/2	5	6

Remark 1 It is shown in [12] that, for a sufficiently large n , the number of LP-classes for functions of n variables is $n!6^n$. We denote by $K(n)$ the number of WDDs for LP-representative functions that are different in the number of non-terminal nodes, the width, and the number of paths. If the networks are derived from WDDs, then $K(n)$ is the number of different networks to realize all LP-representative functions of a given number of variables. From considerations of functions for $n = 3$ and $n = 4$, and a discussion of properties of Walsh spectra of switching functions, it is believed that $K(n)$ is not larger than $\frac{1}{2}n!6^n$. When a LP-representative function is realized, the same network can be used to realize other function in the same LP-class through the LP-classification rules.

Table 8: WDD classes for $n = 4$.

	ntn	w	paths	f
1.	4	1/1	5	1, 25
2.	8	3/4	10	6, 8, 16
3.	9	4/4	10	29
4.	9	3/3	16	2, 24
5.	9	3/3	12	18
6.	9	4/4	12	23, 27
7.	10	4/3	12	9, 10, 11, 14
8.	10	4/4	16	7, 15, 21, 30
9.	11	5/4	16	3, 17, 19
10.	11	4/3	16	13
11.	11	4/3	12	4, 22
12.	11	6/4	12	28
13.	12	5/4	16	5, 12
14.	13	6/4	16	20, 26

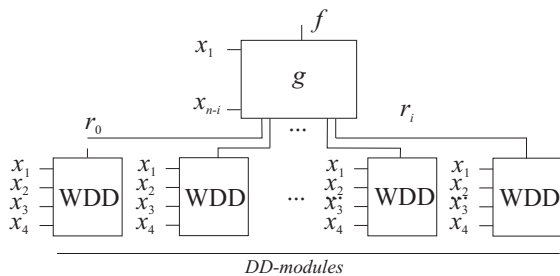


Figure 5: Design method.

4 DESIGN FROM DD CLASSES

We assume that for some reasons, as for example, testing and verification, the uniformity of modules is an important issue in a particular design.

Consider a function f of n variables that can be represented as a function g in terms of a subset of variables in f and subfunctions r_i of three or four variables of f . Therefore, f can be represented as $f = g(x_1, \dots, x_{n-k}, r_1, \dots, r_i)$, where $k = 2, 3$. For each r_i , we determine the LP-class to which r_i belongs and select the corresponding DD-module. If possible, we search over different DDs for that where the majority of subfunctions r_k , belong to the same DD-class. We realize f by a network consisting of the subnetwork for g and a subnetwork of possibly identical DD-modules. Fig. 5 shows the proposed design method. The method can be generalized recursively by allowing that each subfunction r_i may be represented in the same way as a function of subfunctions of three or four variables, etc.

Example 4 Fig. 6 shows the realization of a function $f = \bar{x}_1\bar{x}_2r_8 \oplus \bar{x}_1x_2r_{10} \oplus x_1\bar{x}_2r_{11} \oplus x_1x_2r_{15}$, where r_8, r_{10}, r_{11} , and r_{15} , may be arbitrary functions that can be reduced to the LP-representative functions f_8, f_{10}, f_{11} , and f_{15} . Since, all these functions belong to the same HSTDD-class, it follows that the corresponding network has the identical HSDT-modules $HSDT(f_8)$ for these subfunctions.

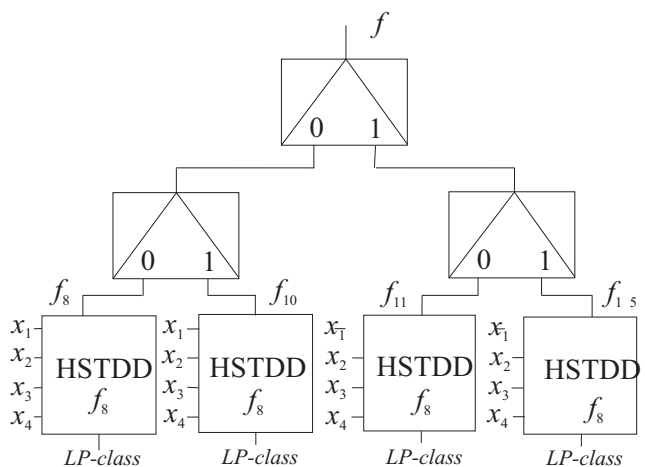


Figure 6: Realization of f in Example 4.

5 CLOSING REMARKS

Technology mapping from decision diagrams results in n -level networks, n - number of variables, with average number of elements approximating $O(2^n/n)$. The design procedure is quite simpler and reduced to the replacement of non-terminal nodes by basic elements from a fixed circuit library, where each circuit corresponds to an expansion rule used in DDs.

When there is interest in synthesis using uniform modules, the realizations uses classification of switching functions and networks are derived by manipulation with networks realizing the representative functions for the classification classes. In this respect, the LP-classification is the strongest, since provides for the fewer number of different classes of switching functions. The design from DDs can further reduce the number of different modules, since some representative functions can be realized by DDs of the same shape.

We consider the number of BDDs, HSTDDs, and WDDs of different shape to represent the representative functions in LP-classification of switching functions for $n = 3$ and $n = 4$. Table 9 shows the number of DD classes for switching functions of $n = 3$ and $n = 4$ variables with respect to BDDs, HSTDDs, and WDDs. There are five, four, and three BDD,

HSTDD, and WDD classes for $n = 3$, respectively. For $n = 4$, there are 21, 18, and 14, BDD, HSTDD, and WDD classes. Compared to BDDs and HSTDDs, WDDs provide the fewer number of different networks to realize all LP-representative functions due to the properties of Walsh spectra of switching functions in $(1, -1)$ encoding. It follows that networks derived from WDDs have the fewer number of different basic WDD modules for $n = 3$ and $n = 4$ variables.

Table 9: DD classes.

DD	n	
	3	4
BDD	5	21
HSTDD	4	18
WDD	3	14

Acknowledgments

The work of M.G. Karpovsky was supported by the BSF Grant 1998154. The work of R.S. Stanković supported by the Academy of Finland, Finnish Center of Excellence Programme, Grant No. 44876, and the EXSITE Project No. 51520.

References

- [1] Bryant, R.E., "Graph-based algorithms for Boolean functions manipulation," *IEEE Trans. Comput.*, Vol.C-35, No.8, 1986, 667-691.
- [2] Clarke, E.M., Zhao, X., Fujita, M., Matsunaga, Y., McGeer, R., "Fast Walsh transform computation with Binary Decision Diagram", *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, September 16-17, 1993, Hamburg, Germany, 82-85.
- [3] Drechsler, R., Becker, B., "OKFDD - Algorithms, applications and extensions", in [11], 163-190.
- [4] Ishiura, N., "Synthesis of multi-level logic circuits from binary decision diagrams", *SASIMI'92*, 74-83.
- [5] Karpovsky, M.G., *Finite Orthogonal Series in the Design of Digital Devices*, Wiley and JUP, New York and Jerusalem, 1976.
- [6] Le, V.V., Besson, T., Abbara, A., Brasen, D., Bogushevitch, H., Saucier, G., Crastes, M., "ASIC prototyping with area oriented mapping for ALTEA/FLEX devices", *Proc. SASIMI-95*, August 1995, 176-183.
- [7] Liaw, H.-T., Lin, C.-S., "On the OBDD representation of generalized Boolean functions", *IEEE Trans. on Computers*, Vol. 41, No. 6, 1992, 661-664.
- [8] Lupanov, O.B., "Asymtotic bounds on the complexity of formulas which realize the functions of logical algebra", *Doklady A.N.*, Vol. 119, No. 1, 1958, 23-26.
- [9] Sasao, T., *Switching Theory for Logic Synthesis*, Kluwer Academic Publishers, 1999.
- [10] Sasao, T., Butler, J.T., "A design method for look-up table type FPGA by pseudo-Kronecker expansions," *ISMVL-24*, May 1994, 97-104.
- [11] Sasao, T., Fujita, M., (eds.), *Representations of Discrete Functions*, Kluwer Academic Publishers, 1996.
- [12] Koda, N., Sasao, T., "LP equivalence class of logic fucntions", *Proc. IFIP Wg 10.5 Workshop on Applications of Reed-Muller Expression in Circuit Design*, Hamburg, Geramny, September 1993.
- [13] Schaefer, I., Perkowski, M.A., Wu, H., "Multi-level logic synthesis for cellular FPGAs based on orthogonal expressions", *First Int. Workshop on Application of Reed-Muller Expansion in Circuit Design*, Hamburg, Germany, September 1993, 42-51.
- [14] Shannon, C.E., "The synthesis of two-terminal switching circuits", *Bel System Tech. J.*, Vol. 28, No. 1, 1948, 59-98.
- [15] Stanković, R.S., Sasao, T., Moraga, C., "Spectral transform decision diagrams", *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expanssion in Circuit Design*, Makuhari, Japan, August 27-29, 1995.
- [16] Stanković, R.S., Sasao, T., Moraga, C., "Spectral transform decision diagrams" in [11], 55-92.
- [17] Stanković, R.S., Stanković, M., Astola, J.T., Egiazarain, K., "Haar spectral transform decision diagrams with exact algorithm for minimization of the number of paths", *Proc. 4th Int. Workshop on Boolean Problems*, Freiberg, Germany, September 21-22, 2000.
- [18] Stanković, M., Stanković, R.S., Astola, J.T., Egiazarain, K., *Fibonacci Decision Diagrams*, TICSP Series # 8, TICSP, Finland, 2000.