BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES
Vol. 54, No. 4, 2006

BDD-based decompositions of multiple output logic functions

A. DZIKOWSKI* and E. HRYNKIEWICZ

Institute of Electronics, Silesian University of Technology, 16 Akademicka St., 44-100 Gliwice, Poland

Abstract. The paper presents modification of the method dedicated to a complex area decomposition of a set of logic functions whereas the altered method is dedicated to implement the considered logic circuits within FPGA structures. The authors attempted to reach solutions where the number of configurable logic blocks and the number of structural layer would be reasonably balanced on the basis of the minimization principle. The main advantage of the procedure when the decomposition is carried out directly on the BDD diagram is the opportunity of immediate checking whether the decomposed areas of the diagram do not exceed the resources of logic blocks incorporated into the integrated circuits that are used for implementation of the logic functions involved.

Key words: Binary Decision Diagrams, decomposition, FPGA.

1. Introduction

Designing of digital circuits that are later on implemented within programmable circuits is mostly targeted to programmable structures of CPLD and FPGA types. Since such structures comprise a number of identical logic blocks that enable realization of very simple logic functions, employment of decomposition becomes indispensable, whereas the decomposition is understood as subdivision of a complex digital circuit into a number of submodules that can be fitted one after another into PLD blocks of CPLD circuits or CLB blocks of FPGA circuits. The problem becomes extremely crucial in case if FPGAs are used as such circuits because they are made up of thousands of identical logic blocks (CLB – Configurable Logic Blocks). Thus, if the FPGA circuits are used the designing process of a digital circuit involves multiple process of decomposition that leads eventually to cover the entire and complex digital circuit by a number of CLB blocks of a FPGA matrix [1].

2. Decomposition of a multiple output logic function

In order to present various solutions that can be reached if decomposition of a set of logic functions (a multiple-output logic function) is carried out directly on the BDD (BDD – Binary Decision Diagram) [1,2] let us go through the decomposition process for the rd84 test function.

Based on the obtained results advantages and disadvantages of various decomposition methods with use of BDD diagrams shall be presented.

The test function rd84 [14] has 8 inputs and 4 outputs. The 4-bit output vector represents the binary value corresponding to the number of "ones" that occur in the 8-bit input vector. The rd84 function, being expressed in the Berkeley format, comprises as many as 256 terms. The BDD for the rd84 function is shown in Fig. 1.

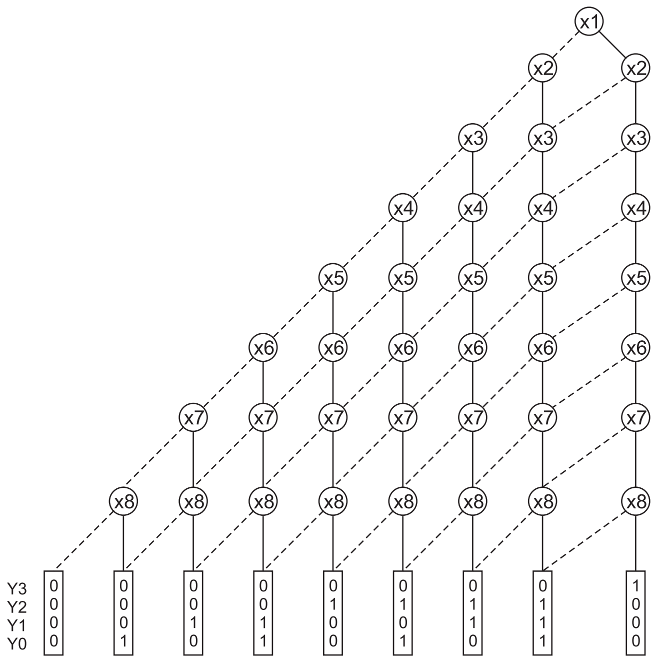

Fig. 1. The BDD for the rd84 test function

2.1. Simple disjunctive decomposition of BDDs. The most primary type of the decomposition method (Fig. 2) is so called simple or trivial decomposition that was defined by Ashenhurst [3] for the first time and that can be expressed by the following formula:

$$F(X) = H(A, G(B))$$

where:

$$A \cup B = X \quad A \cap B = \oslash$$

The simple decomposition is carried out on the ROBDD by searching for the location where the BDD can be split to sub-BDDs. However, such a location depends on the structure of

*e-mail: andrzej.dzikowski@polsl.pl

configurable logic blocks (CLBs) that are used for implementation of the functions handled. It can be determined by means of the following definition:

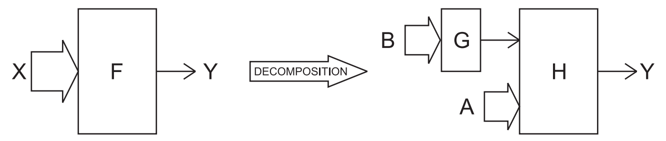

Fig. 2. Schematic outline of the simple Ashenhurst-like decomposition

DEFINITION 1. After having started analysis of the BDD from the initial node (the root), the breaking line should be inserted beyond as many nodes of the BDD as the number of inputs for the LUT (Look-up Tables) of CLB blocks exists for a specific FPGA applied.

Since we can assume that majority of contemporary FPGA-type logic structures incorporates CLBs where LUTs have four inputs (circuits from Xilinx of the series Spartan, Virtex and XC4000) the breaking line of the BDD should be drawn beyond the fourth row of nodes included into the BDD. Alternatively, if the internal structure of CLBs in a specific FPGA enables combining for instance two LUTs into a block with 5 inputs, the breaking line can be drawn beyond the fifth row of nodes (the circuits from Xilinx as listed above).

Obviously, the BDD of the rd84 test function cannot be split by means of the simple decomposition method and the advanced decomposition must be applied.

2.2. Advanced (non-trivial) decomposition of BDDs. If the BDD of a logic function is excessively sophisticated and cannot be implemented within CLBs by means of the simple decomposition method, the advanced decomposition approach has to be applied. This is the method that in its simplest embodiment can be reduced to subsequent application of the simple decomposition method (of either serial or parallel type) to the areas of the BDD that reciprocates the function that is to be decomposed.

The general scheme of the advanced decomposition method is outlined in Fig. 3.

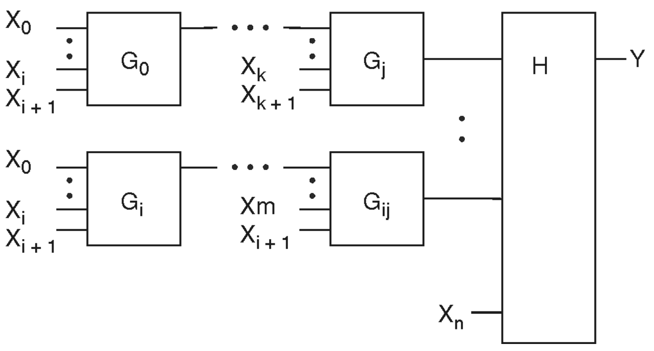

Fig. 3. Schematic outline of the advanced decomposition approach

2.3. Advanced decomposition of the rd84 test function. To carry out the advanced decomposition of the rd84 test function directly on the BDD we have to draw that diagram in such a way that would enable implementation of that diagram within CLBs (logic blocks) of an FPGA circuit [4–6]. Since the CLBs has only two outputs we have to draw two diagrams for the two sets of Boolean functions that correspond to respective pairs of the outputs of the rd84 test circuit, e.g. Y3,Y2 and Y1,Y0. The relevant BDDs are presented in Fig. 4. Next, such areas of the BDDs must be marked off (circled) that are destined for implementation by means of individual CLBs.

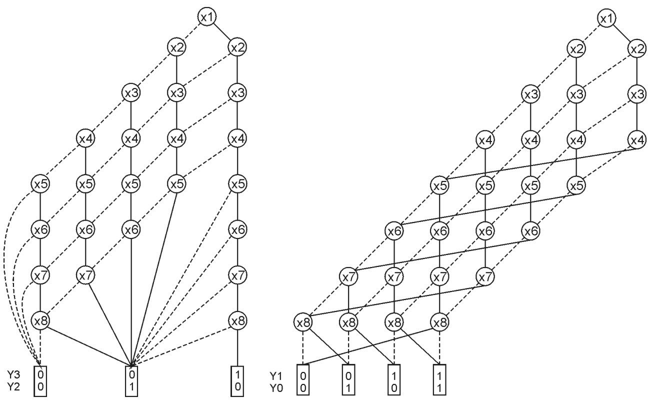

Fig. 4. BDDs for two sets of Boolean functions included into the rd84 test function

Figure 5 precisely explains all the details related to the selected area as well as to encoding the output variables.

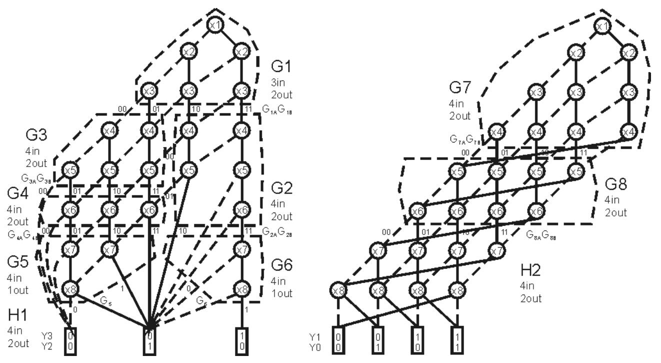

Fig. 5. Advanced (non-trivial) decomposition of the rd84 test function

Logic structure of combined CLBs is obtained as a result of the advanced decomposition of the rd84 test function as presented in Fig. 5. The eventual structure is outlined in Fig. 6. In such a case the set of functions (the multiple-output function) Y1, Y2 subjects to serial decomposition.

The logic structure that has been obtained after the advanced decomposition of the rd84 test function needs 9 CLBs and is a 5-layer structure.

In order to define logic functions corresponding to individual CLB modules the reduced sub-BDDs that are covered by these CLBs are presented in Fig. 7.

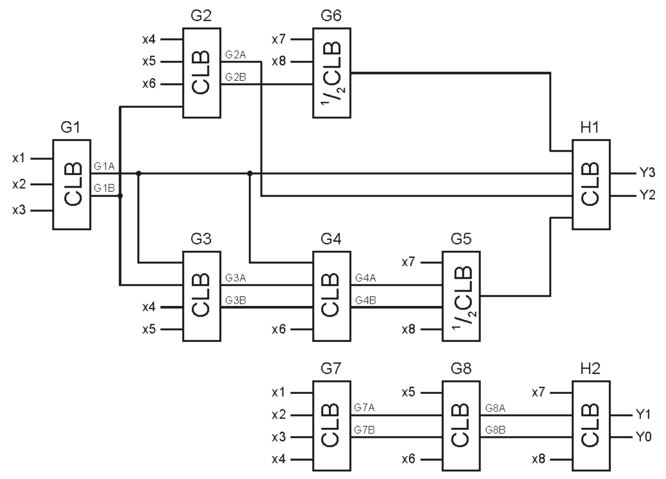

Fig. 6. Logic structure of the rd84 test function after the advanced decomposition process

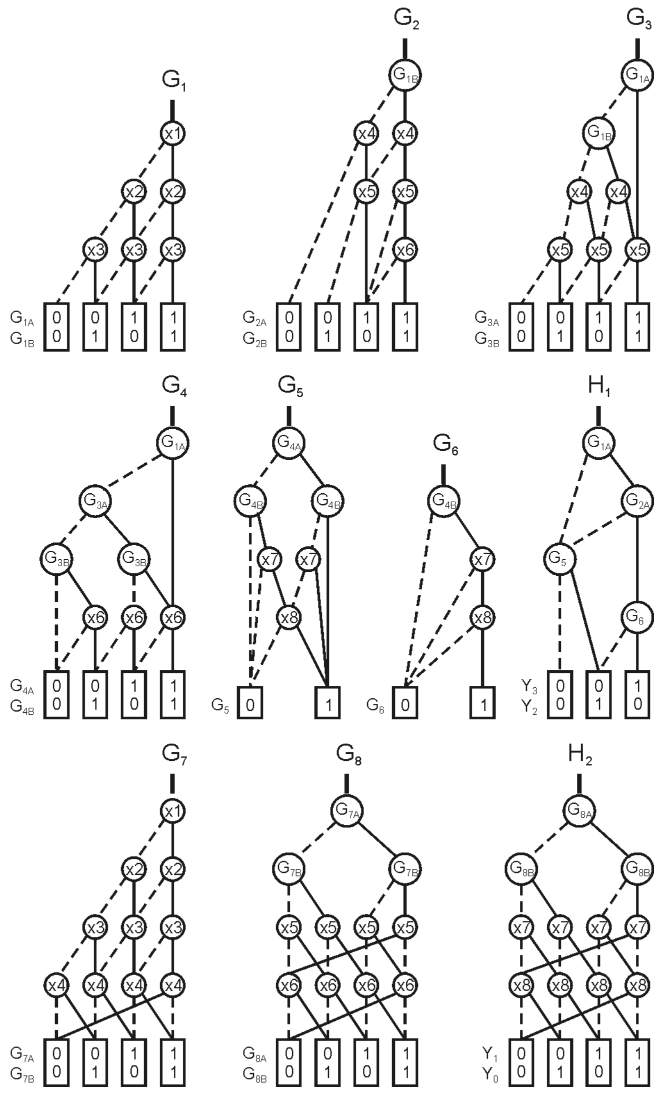

Fig. 7. Sub-BDDs coverable by standard CLBs after advanced decomposition of the rd84 test function

2.4. Advanced decomposition of the rd84 test function based on BDD splitting. The idea of splitting a BDD consists in eliminating those branches that link the areas of the BDD that subject to parallel decomposition [7–9].

Let us to carry out the advanced decomposition of the rd84 test function with splitting the BDD into the left and right parts. For this purpose we have to modify the diagram for the set of functions Y3,Y2 as is shown in Fig. 5. Since two linking branches exist that connect the block G2 with the blocks G3 and G4 we have to proceed in order to eliminate these links, which is shown in Fig. 8. The BDD for the set of functions Y1,Y0 remains unaltered and keeps subjecting to serial decomposition.

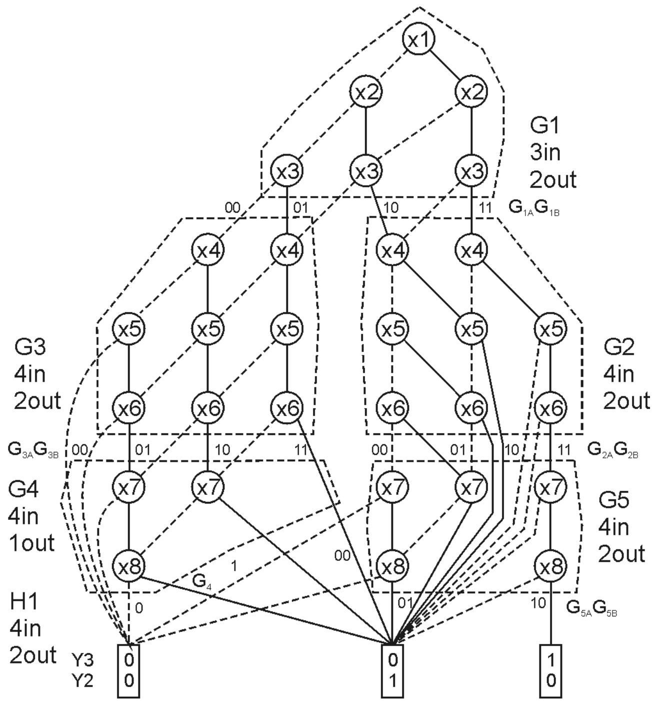

Fig. 8. Advanced decomposition of the set of Boolean functions for the outputs Y3, Y2 of the rd84 test function after having the BDD split

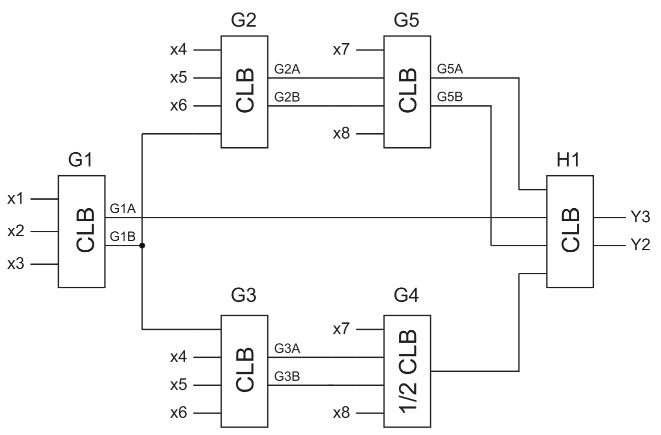

Fig. 9. The logic structure after the process so advanced decomposition with splitting the BDD for the Y3 and Y2 outputs

As a result of such decomposition procedure applied to the rd84 test function the logic structure as shown in Fig. 9 has been obtained. As it can be easily noticed the resulting propagation time of the structure has been reduced as the final structure has only four rows, hence it is faster by 20% in comparison to the previous layout.

The number of used logic blocks (CLBs) remained virtually the same and equals to 81/2 CLBs to cover the entire rd84 function.

Similarly to the preceding drawings, Fig. 10 shows sub-BDDs of logic functions that correspond to individual CLBs

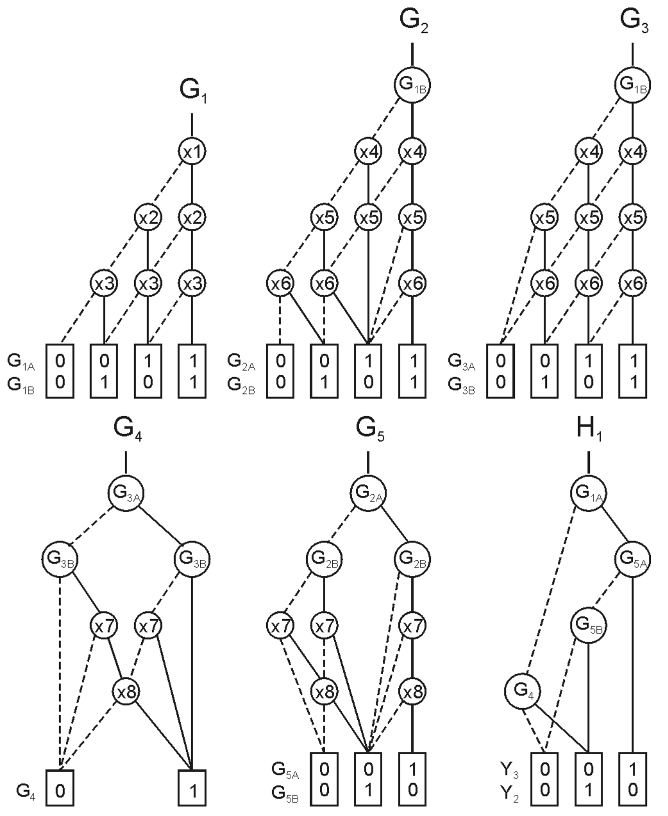

Fig. 10. The sub-BDDs corresponding to single CLBs after the advanced decomposition process that employs splitting the BDD for the set of outputs Y3, Y2 of the rd84 test function

2.5. Serial decomposition of the rd84 function. The process of serial decomposition of the rd84 test function is carried out on the entire BDD as it is shown in Fig. 1. Figure 11 presents the process of serial decomposition of the rd84 test function along with encoding of resulting (output) variables for individual CLB modules.

Subsequently, Fig. 12 presents the eventual structure of mutually interconnected CLB modules that has been obtained after completion of serial decomposition of the rd84 test function.

As it is show on the diagram, after completion of the serial decomposition process the logic structure of the rd84 test function employs only 8 CLB modules and presents a three-layer hierarchy, thus it is faster than the original one by 40%.

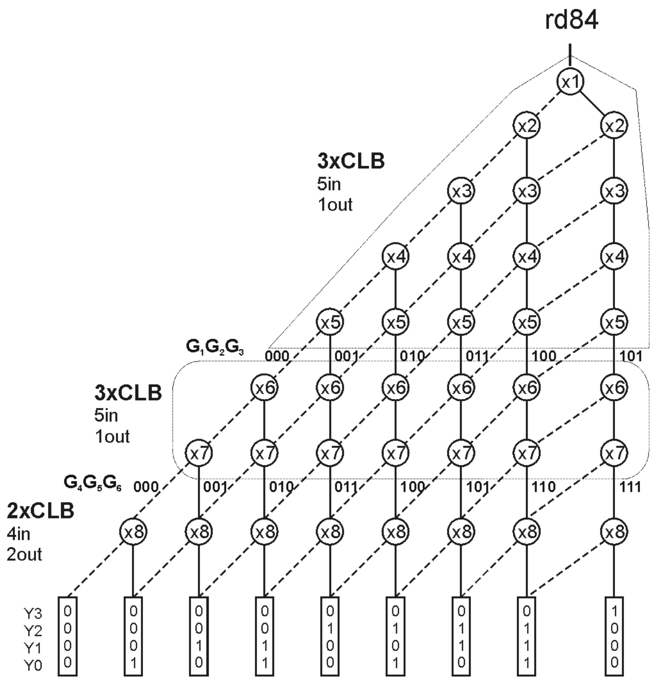

Fig. 11. Serial decomposition of the rd84 test function

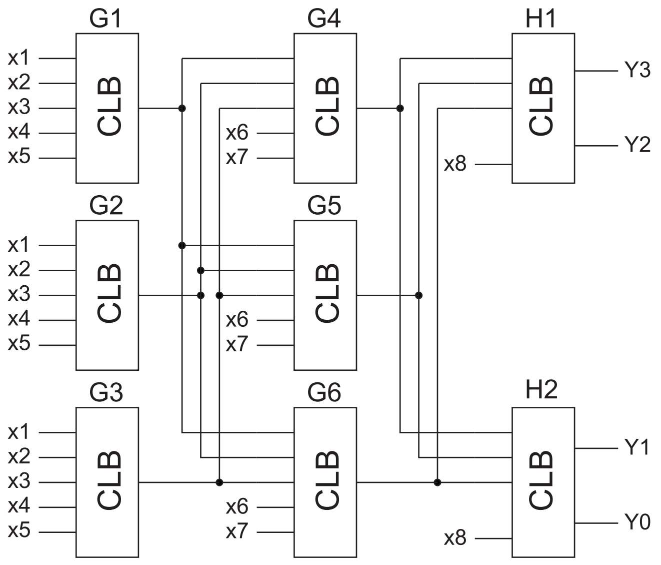

Fig. 12. Eventual logic structure after completion of the serial decomposition of the rd84 test function

Similarly to the previous deliberations let us consider once again the sub-BDDs that are covered by individual CLB modules as shown in Fig. 13.

As one can see on the above drawing the sub-BDD for the G_6 module uses only three variables so only a half of a CLB module is enough for its realization. Thus implementation of the entire rd84 test function after serial decomposition needs 7.5 of a CLB modules and presents a three-level hierarchy.

In order to explain the rules which govern extraction of the sub-BDDs that are realized by individual CLBs the subsequent stages of developing the G_1 sub-BDD are shown in Fig. 14 [10].

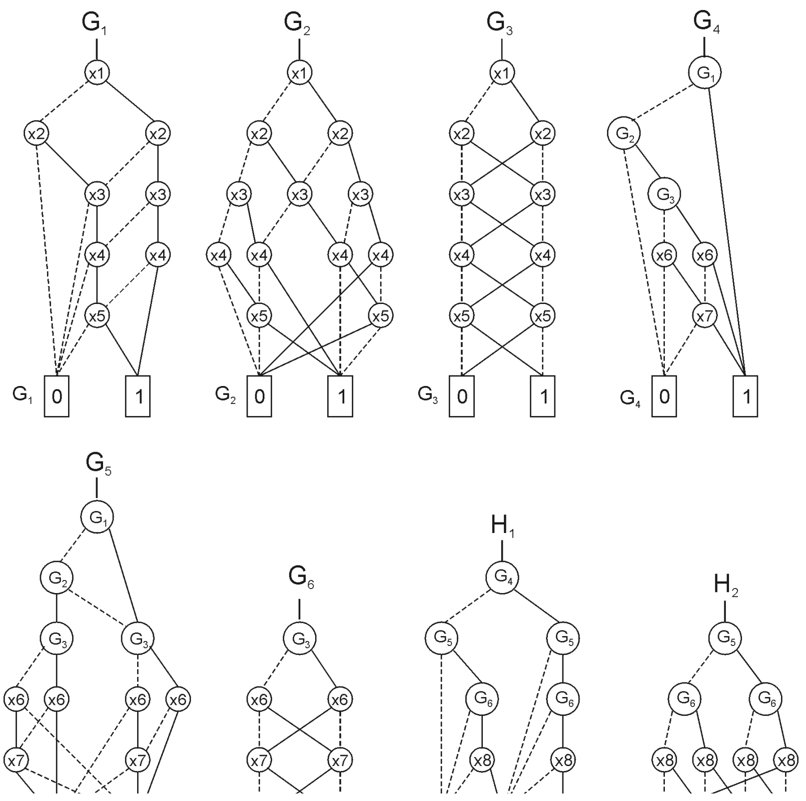

Fig. 13. The sub-BDDs that are covered by individual CLB modules after serial decomposition of the rd84 test function

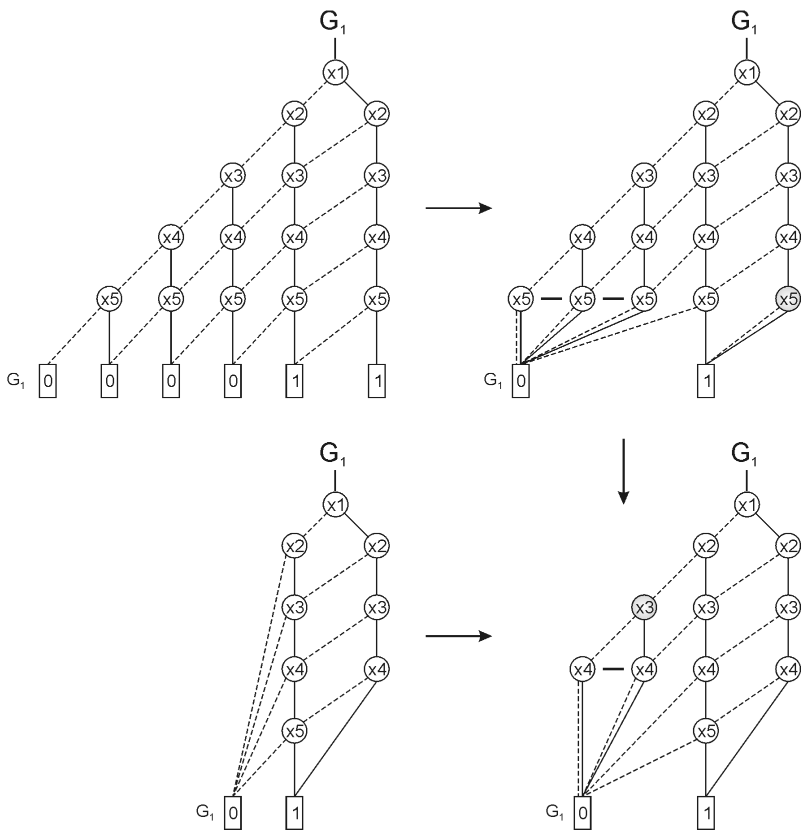

Fig. 14. Subsequent stages of the process of the G_1 sub-BDD reduction

Reduction of the G$_1$ sub-diagram comprises the following stages:

– Elimination of leaves of the same type
– Clustering the notes that interpret the same information
– Elimination of insignificant nodes

In case of sub-BDDs such as G$_4$ the BDD root is defined as a result of applying the rules as presented in the papers [6–8].

2.6. Decomposition of the rd84 test function based on searching for a sub-BDD of a one-bit adder. Since the rd84 test function calculates the number of “ones” in the input vector it is worth considering realization of that function by means of one-bit adders.

For that purpose Fig. 15 presents BDDs for an adder and a half-adder

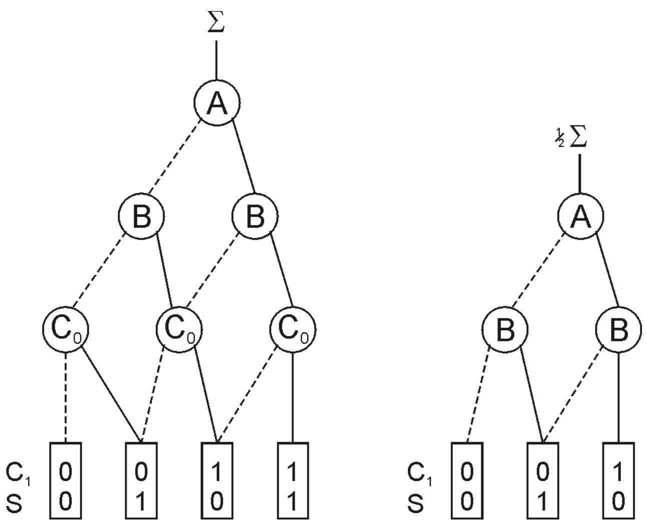

Fig. 15. The BDDs for an adder and a half-adder

Let us search for the structures like the ones on the drawing within the BDD of the rd84 test function. The first adder-type sub-BDD can be spotted at the first sight when the BDD is cut with a line beyond the third row of nodes and the nodes in the subsequent row (the forth one) are encoded with natural binary numbers, which is shown in Fig. 16.

Then the remaining part of the BDD for the rd84 test function just below the x4 variable should be taken into consideration. In that part the adder-like sub-BDD can also be found, whereas the other variables can make up the BDDs of half-adders, which is shown in Fig. 17.

Such a procedure of decomposing the rd84 test function has led to development of a resulting circuit that converts the input information into three vectors. Then these three vectors should be merely added up, which can be executed by means of adders and half-adders as well in order to achieve the layout as presented in Fig. 18.

Each of the adders can be implemented within a single CLB module, so the four-level hierarchy that employs CLBs is obtained.

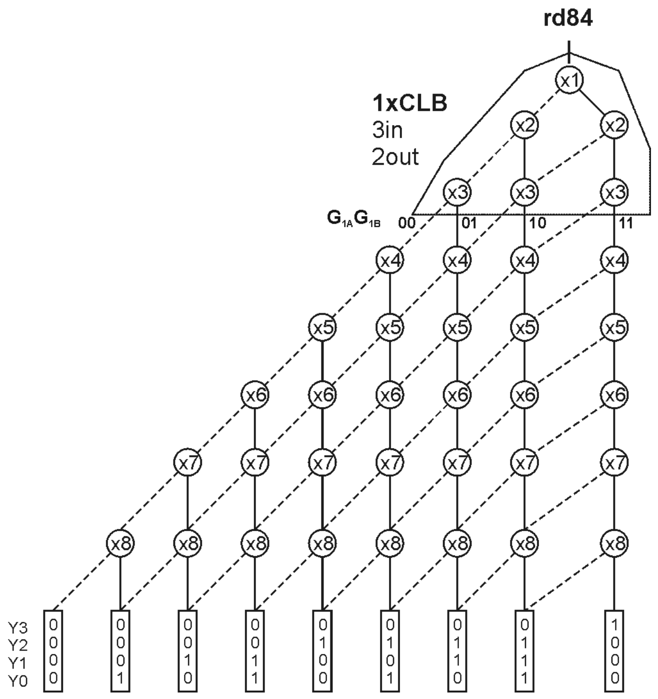

Fig. 16. The BDD for the rd84 test function with the adder-type sub-BDD found

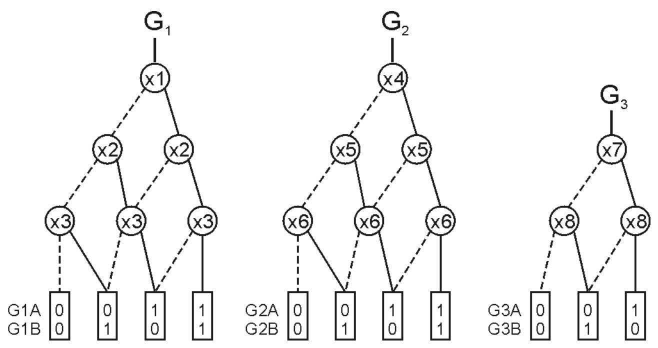

Fig. 17. The sub-diagrams of adders and half-adders disclosed within the BDD of the rd84 test function

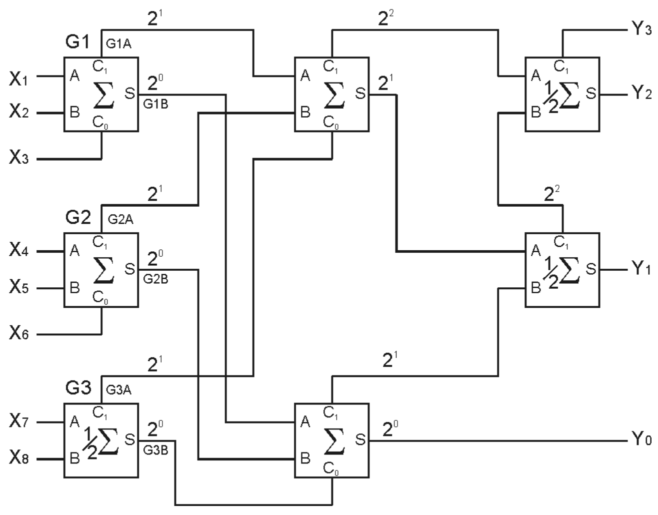

Fig. 18. Structural hierarchy of the rd84 test function after the decomposition process that employs searching for adders

2.7. Modification of the decomposition result for the rd84 test function based on searching for a sub-BDD of a one-bit adder. After more thorough analysis of the diagram that was developed in the previous paragraph the conclusion can be made that for some CLBs only a small part of their resources is used. Hence let us to modify the obtained solution. On the first step let us avoid using all the input variables straight off but only the vector of first seven arguments will be involved. Since the fact that the rd73 test function is actually comprised by the rd84 function the rd73 test function is virtually obtained, which is disclosed in Fig. 19.

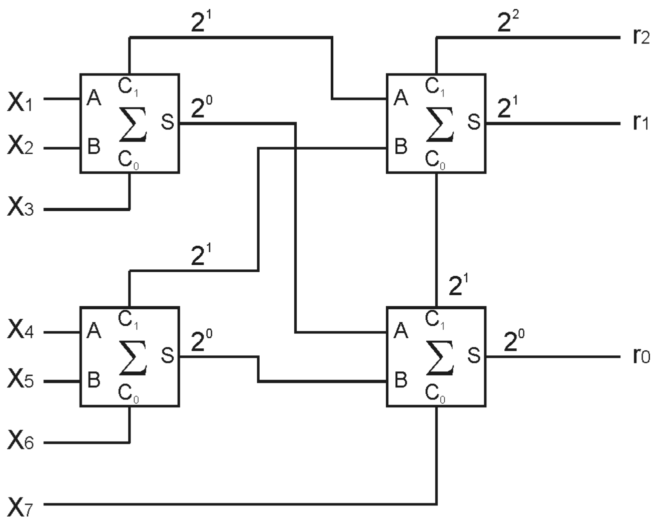

Fig. 19. Realization of the rd73 test function by means of adders

Now it is enough to take account for the eighth variable by combining it with the three-bit result in accordance with the Karnaugh map below and Fig. 20.

Table 1
The Karnaugh map of the rd84 test function being expressed as the sum of the rd73 function and X_8 $(\mathrm{rd73} + X_8)$

X_3 \ $r_2r_1r_0$	000	001	011	010	110	111	101	100
0	0000	0001	0011	0010	0110	0111	0101	0100
1	0001	0010	0100	0011	0111	1000	0110	0101

$Y_3Y_2Y_1Y_0$

$$Y_3 = X_8r_2r_1r_0 \qquad Y_1 = r_0(X_8 \oplus r_1) + r_1\bar{r}_0$$
$$Y_2 = r_2 \oplus X_8r_1r_0 \qquad Y_0 = X_8 \oplus r_0$$

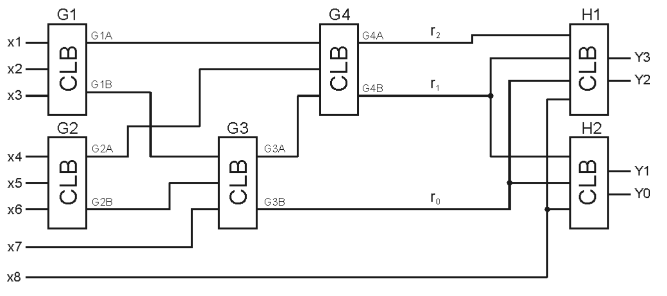

Fig. 20. Modified layout of the rd84 test function after the decomposition process with searching for adders

The eventual solution employs only 6 CLBs but still keeps on being the four-level hierarchy. Thus let us try to cut down the propagation time of the resulting logic structure. For that purpose only the modules G1 and G2 shall be implemented in unaltered manner. As the G1 and G2 modules have four outputs altogether we have to cover five variables, including ×7, by the remaining part of the circuit. It is why three 5-input CLBs are sufficient to implement the entire rd73 test function as it is shown in Fig. 21.

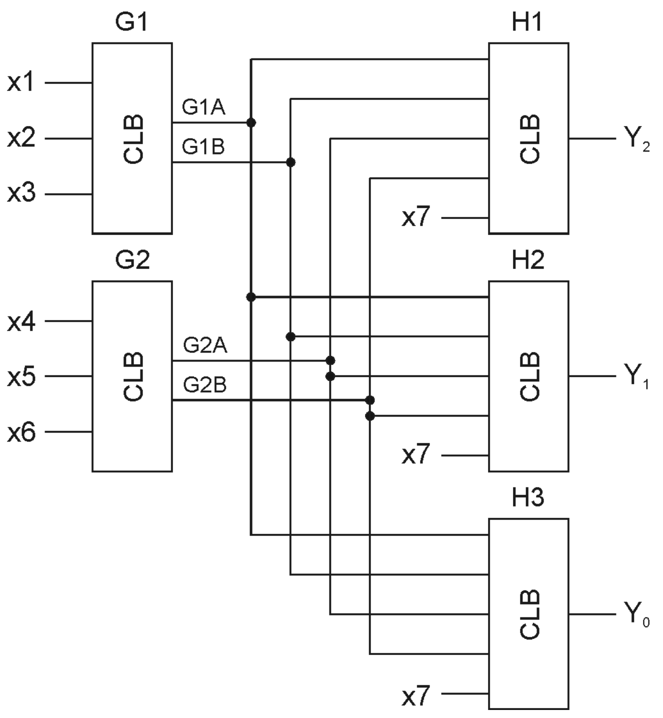

Fig. 21. The logic structure of the rd73 test function after modification

Eventually a vary fast solution has been reached that fits in a two-level hierarchy even though it needs 5 CLB modules. However the attention should be paid to the sub-BDDs that are covered by individual CLBs as they are presented in Fig. 22.

The BDDs for the G1 and G2 CLBs have not been shown on the drawing below as they are exactly the same as in Fig. 17.

As it can be easily seen, the BDD for the H3 block uses only 3 variables thus it actually uses only a half of the resources attributable to a CLB module of a FPGA circuit.

Consequently, realization of the rd73 test function needs only 4.5 of CLB modules instead of 5 ones. When the X_8 variable is combined with the rd73 function, similarly to the previous example, the three-layer hierarchy can be reached for the rd84 test function although the solution employs 6.5 CLBs, which is shown in Fig. 23.

The sub-BDDs that are covered by individual CLBs employed by the above solution are shown in Fig. 24.

Results of decomposition procedures that were carried out for the rd84 test function directly on the BDD by means of various methods are presented in Table 2.

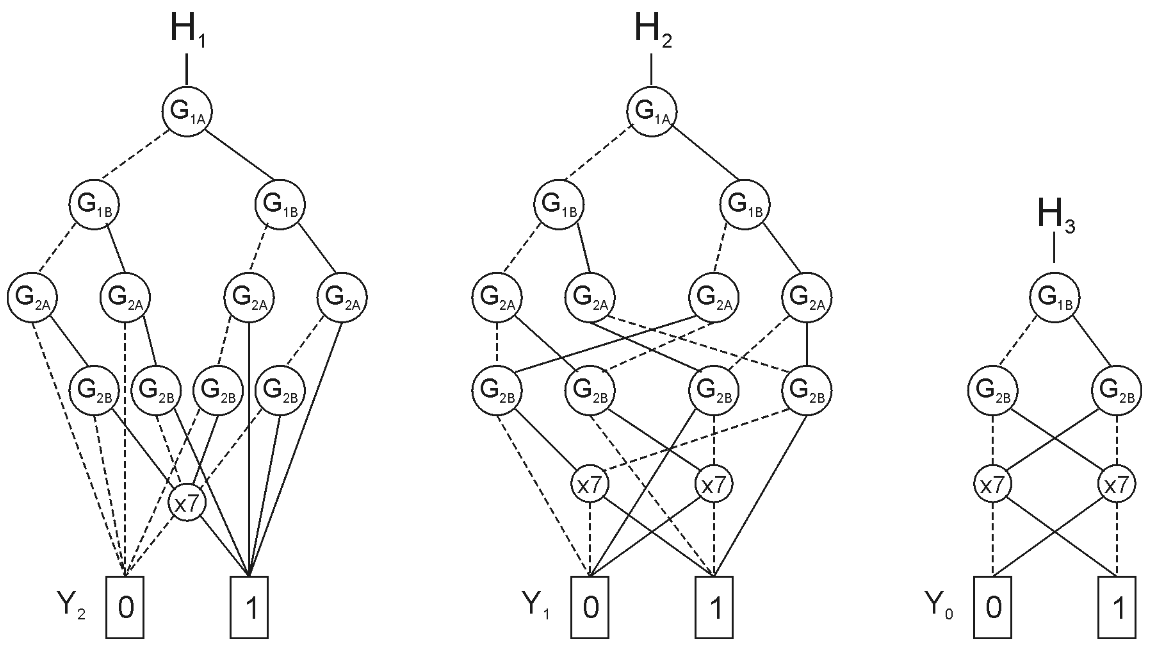

Fig. 22. The sub-BDDs covered by CLB modules after modification of the results from decomposition of the rd73 test function

Table 2
Results of decomposition procedures for the rd84 test function

Decomposition method	Number of CLBs employed	Number of layers	Reduction of the propagation time as compared to the advanced decomposition
Advanced decomposition	9	5	0%
Advanced decomposition with splitting the BDD	8.5	4	20%
Serial decomposition	71.5	3	40%
Advanced decomposition with searching for adders	7	4	20%
Modified advanced decomposition with searching for sub-BDDs of adders	6 or 6.5	4 or 3	20% or 40%

On the background of the obtained results that are brought together in Table 2 the following conclusions can be formulated:

– The advanced decomposition leads to relatively poor results.
– The advanced decomposition with splitting the BDD leads to better results than the advanced decomposition of a multiple-output logic function.
– The serial decomposition leads to exceptional results in terms of propagation time achieved. Generally, for unspecific functions, it seems to be the best solution.
– For arithmetical function a preferable solution can be achieved that employs less number of logic blocks (CLBs) if the advanced decomposition method with searching for adders is applied. However, such a hardware solution is featured by longer propagation time as compared to the serial decomposition method. The propagation time can be optimized by modification of the method of advanced decomposition with searching for adders.

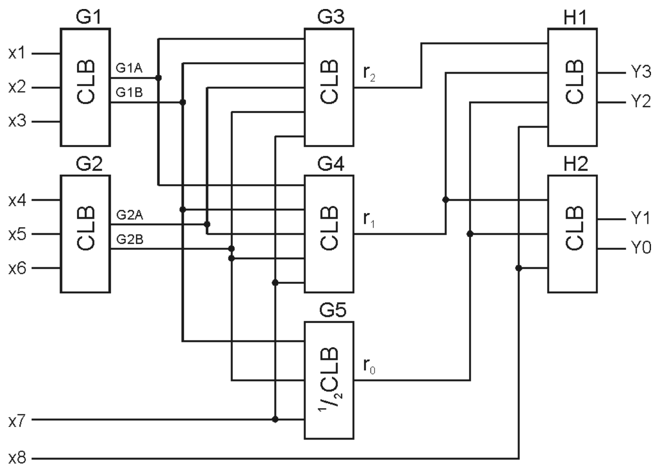

Fig. 23. The logic structure of the rd84 test function after modification

3. Conclusions

After having applied the method of decomposition directly on the BDD to a series of test functions [9] the obtained results were compared with other results of logic synthesis as published in [11–14] and the result of comparison were brought together in the Table 3. The gathered information can serve as a background to state that pretty good results can be achieved for most circuits by means of the method of serial decomposition.

For such test functions as rd53, rd73 and rd84 the method of advanced decomposition with searching for adders quite frequently leads to much better solutions. To improve the quality of solutions as obtained for the method with searching for adders one can carry out modification of the eventual result whereas the applied modification techniques are determined by the desired objective.

The objective of modification can be specified either unambiguously as minimization of the hardware (number of CLB modules) or reducing the number of hierarchic layers (propagation time) or sometimes a compromise between the both strategies can be reached. In many cases the detailed and thorough analysis of logic functions covered by individual CLB modules makes it possible to improve optimization of the final logic circuit.

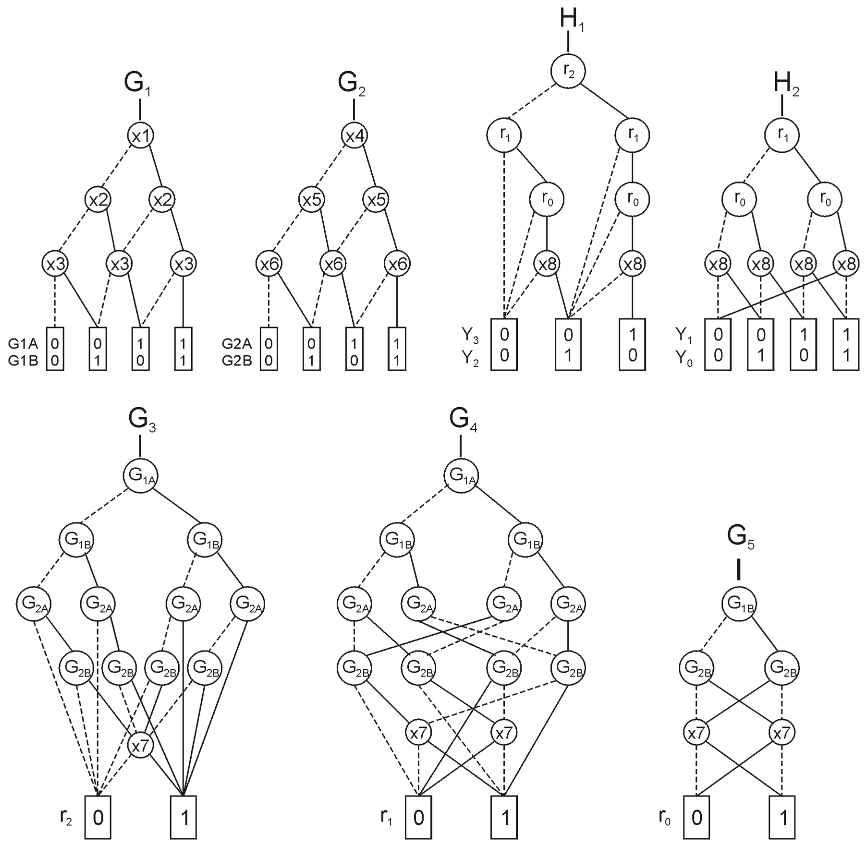

Fig. 24. The sub-BDDs covered by the CLBs after modification of decomposition results for the rd84 test function

Table 3
Results of experiments (number of CLBs /number of layers)

Decomposition tools	Test function									
	5xp1	9sym	Bw	F51m	Misex1	Rd73	Rd84	Root	Sao2	Z4
ALTO	19/2	7/3		15/3	14/3	8/2	13/3		37/5	5/2
ASYL	13/4	8/3	27/6	14/6	13/2	8/4	14/4		30/3	4/2
Chortle	20/3	41/5			14/2		41/4			
Demain	9/2	5/3		10/2	8/2	5/2	7/2	16/3	18/3	4/2
Mispga	17/3	7/3	27/1	11/4	9/3	7/3	12/3		28/5	4/2
TRADE	11/2	6/3	27/1	9/3	14/2	5/2	8/3	21/3	27/3	4/2
Decomp	11/2	5/3	26/1	10/3	10/3	5/2	7/3	19/3	20/5	4/2
The best result as was achieved by means of the methods presented in this paper	18/2	7/3	27/1	18/3	11/2	4/3	6/4	21/3	23/4	4/2

References

[1] R.E. Bryant, "Graph-based algorithms for Boolean function manipulation", *IEEE Trans. Computers* C-35, 667–691 (1986).

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

[3] R.L. Ashenhurst, "The decomposition of switching function", *Proc. Int. Symposium on the Theory of Switching*, 1957.

[4] R. Murgai, R.K. Brayton, and A. Sangiovanni-Vincentelli, *Logic Synthesis for Field-Programmable Gate Arrays*, Kluwer Academic Publishers, Boston–Dordrecht–London, 1995.

[5] A. Dzikowski and E. Hrynkiewicz, "Methods of decomposition of multiple output logic functions with using ROBDD diagrams", *IV National Scientific Conference*, *Programmable Logic Devices*, 19–28 (2001).

[6] A. Dzikowski and E. Hrynkiewicz, "Advanced area decomposition of multiple output logic functions with using ROBDD diagrams", *II National Electronics Conference* II, 393–398 (2003).

[7] A. Dzikowski and E. Hrynkiewicz, "Modification of area decomposition of multiple output logic functions with using of ROBDD diagrams", *III National Electronics Conference* I, 285–290 (2004).

[8] A. Dzikowski and E. Hrynkiewicz, "Modification of area decomposition of multiple output logic functions in shorting propagation time", *IV National Electronics Conference* I, 171–17 (2005).

[9] A benchmark set, University of California, Los Angeles, VLSI CAD Laboratory, `http://vlsicad.cs.ucla.edu/ cheese/ benchmarks.html`

[10] R. Lasocki, "Decomposition of functional interrelationships determined on finite sets", *PhD Thesis*, Warsaw University of Technology, Warsaw, 1998, (in Polish).

[11] M. Nowicka, T. Łuba, and H. Selvaraj, "Multilevel decomposition strategies in decomposition-based algorithms and tools", *Int. Workshop on Logic and Architecture Synthesis*, 129–136 (1997).

[12] J-D. Huang, J-Y. Jou, and W-Z Shen, "An iterative area/performance trade off algorithm for LUT-based FPGA technology mapping", *IEEE Trans. on Very Large Integration (VLSI) Systems* 8(4), 392–400 (2000).

[13] D. Kania, "The heuristic method for decomposition of Boolean functions and dedicated for matrix-type FPGA circuits with application of the techniques of advanced decomposition", *Quarterly of Electronics and Telecommunication* 2 (46), 191–206 (2000).

[14] D. Kania: "Multiple decomposition in logic synthesis for FPGA devices", *Electronics* 2–3, 43–46 (2002).