Int. J. Appl. Math. Comput. Sci., 2017, Vol. 27, No. 1, 207–222
DOI: 10.1515/amcs-2017-0015

AREA–ORIENTED TECHNOLOGY MAPPING FOR LUT–BASED LOGIC BLOCKS

MARCIN KUBICA [a,*], DARIUSZ KANIA [b]

[a]Faculty of Mechanical Engineering and Computer Science
University of Bielsko-Biała, ul. Willowa 2, 43-309 Bielsko-Biała, Poland
e-mail: mkinz@wp.pl

[b]Institute of Electronics
Silesian University of Technology, ul. Akademicka 2A, 44-100 Gliwice, Poland
e-mail: dkania@polsl.pl

One of the main aspects of logic synthesis dedicated to FPGA is the problem of technology mapping, which is directly associated with the logic decomposition technique. This paper focuses on using configurable properties of CLBs in the process of logic decomposition and technology mapping. A novel theory and a set of efficient techniques for logic decomposition based on a BDD are proposed. The paper shows that logic optimization can be efficiently carried out by using multiple decomposition. The essence of the proposed synthesis method is multiple cutting of a BDD. A new diagram form called an SMTBDD is proposed. Moreover, techniques that allow finding the best technology mapping oriented to configurability of CLBs are presented. In the experimental section, the presented method (MultiDec) is compared with academic and commercial tools. The experimental results show that the proposed technology mapping strategy leads to good results in terms of the number of CLBs.

Keywords: SMTBDD, FPGA, synthesis, decomposition.

1. Introduction

It can be observed that the popularity of FPGA (field programmable gate array) circuits has increased considerably in the last decade. A flexible architecture has turned out to be the key to success. Configurable logic blocks (CLBs) are the core of FPGA structures and may be treated as a kind of memory. That is why the most popular group of FPGA circuits is called look-up table FPGAs. The number of CLBs included inside FPGA structures is high enough to implement complex digital circuits (Wyrwoł and Hrynkiewicz, 2013). In addition, FPGA structures have series of specialized blocks such as I/O blocks, DCMs (digital clock managers), PLLs (phase locked loops), and DSP (digital signal processing). Logic resources inside FPGA structures are arranged in the form of a symmetrical matrix, which provides the opportunity to lead connecting paths between them. Unfortunately, the number of possible connections is limited because of the limited flow of the paths between blocks. Due to limited connecting resources, some of the synthesis stages such as placement or routing become significant from the point of view of implementation of time effective structures.

The synthesis process dedicated to FPGA structures is usually automatic. The producers of FPGA structures very often deliver appropriate software tools. It has been already shown by Cong and Minkovich (2007), that the results of synthesis obtained using commercial tools may be far from ideal. The crucial synthesis element, whose solutions are still not satisfactory, is decomposition. It can be treated as a mathematical model of circuit division between CLBs. Good decomposition should be connected with effective technology mapping dedicated to FPGAs. In the process of technology mapping, it is vital to take into account specific features of logic blocks. The classic model of decomposition theory was devised by Ashenhurst (1957) and Curtis (1962). This model of decomposition is a theoretical background for logic synthesis dedicated to FPGA structures. The first synthesis tools created solutions that were far from optimal. The algorithms of technology mapping were

[*]Corresponding author

originally directed towards gate structures and adapted to the needs of FPGA structures. The most popular algorithms, such as MIS-PGA (Murgai *et al.*, 1991) and ASYL (Abouzeid *et al.*, 1993), are focused on carrying out functions based on multiplexers. Other tools, such as Chortle (Francis *et al.*, 1990) and Xmap (Amap) (Karplus, 1993), used various kinds of logic networks or acyclic graphs in the process of synthesis or the technique of technology mapping (FlowMap) (Cong and Ding, 1994).

The specific features of the first logic synthesis tools dedicated to FPGA structures were procedures of factorization of Boolean functions, lexicographical variable ordering, and iterative network division. In the second half of the 1990s, the development of synthesis tools brought a considerable improvement in terms of synthesis results. The structures had small delays and occupied a very small area of silicon because decomposition was generalized into a multi-output function, effective methods of decomposition of the logic function (Rawski *et al.*, 1997) were devised, and inseparable decomposition was used in the process of synthesis. Tools such as Demain (Rawski *et al.*, 1997), Trade (Wan and Perkowski, 1992), BDDsyn (Chang and Marek-Sadowska, 1992), LGsyn (Lai *et al.*, 1996), and Decomp (Kania, 2004) played a vital role at the time and are worth mentioning.

The effectiveness of logic synthesis process depends on many elements (Fiser and Schmidt, 2009; 2012). Due to the complexity of decomposition algorithms, function representation is especially crucial. There are many methods with which a logic function can be represented. The most popular are the table description (Curtis, 1962), the cube description (Micheli, 1994), and graph methods. Binary decision diagrams (BDDs) (Akers, 1978; Bryant, 1986) have become the most popular form for presenting logic functions lately. Synthesis algorithms using BDDs have been developed since the 1990s. The first algorithms such as BDDsyn (Chang and Marek-Sadowska, 1992) and LGsyn (Lai *et al.*, 1996) gave better results than algorithms directed towards division of a network of gates (Chortle (Francis *et al.*, 1990), MIS-PGA (Murgai *et al.*, 1991)).

Function representation using BDDs guaranteed small memory occupation to store data as well as good time effectiveness. Moreover, BDDs can be easily used for function representation of multi-output functions (Sasao and Butler, 1996). As a result of all these advantages, the BDD has been often used in tools that supported the process of synthesis such as the BDS (Yang and Ciesielski, 2002), the DDBDD (Cheng *et al.*, 2007), and dekBDD (Opara and Kania, 2010). The essence of using BDDs in the process of synthesis dedicated to FPGA structures has been presented by Scholl (2001). In a BDS PGA system using BDDs, resynthesis occurs (Vemuri *et al.*, 2002).

Together with the development of FPGA circuits, various synthesis tools dedicated to FPGA structures were developed. Daomap (Chen and Cong, 2004) or ABC (using the AIG—and inversion graph) (Brayton and Mishchenko, 2010) systems may be regarded as some of the most important achievements after 2000 as they are considerably fast in the process of synthesis.

Upon analysing the architecture development of FPGA circuits, it can be observed that more complex circuits tend to be more flexible as far as CLBs are concerned. The most basic CLBs had relatively small configuration abilities and could only have the number of inputs (LUT5/1 or LUT4/2) defined. At present, apart from defining the number of inputs, which becomes higher, it is possible to describe the operating mode such as arithmetic or shared mode (Altera, 2012). In the literature, an interest in the specificity of new logic cells may be observed (Anderson and Wang, 2011; Ray *et al.*, 2012). Recently, new synthesis tools have begun to appear. Their ability to reconfigure logic blocks is partly used by ALMmap (Liang *et al.*, 2012). The synthesis strategies included in these tools are focused on various optimization targets such as delays of the structures, minimization of the area, and minimization of power consumption. In each of these targets, it is crucial to match CLB configuration to the circuits that were obtained in the process of synthesis and are the result of decomposition. The main purpose of logic synthesis is to map a designed circuit to very universal CLBs.

The purpose of this paper is to present a novel logic synthesis strategy targeted at FPGAs which is based on multiple decomposition. New FPGA architectures, enabling configuration of CLB blocks (particularly LUTs), are considered. Therefore, logic synthesis is based on resource-aware decomposition of logic functions taking advantage of such structures. Logic decomposition is directly related to the technology mapping process. The main contribution is twofold: first, the concept of shared multi-terminal BDDs (SMTBDDs) is introduced; second, a flexible technology mapping algorithm based on these is proposed.

2. Theoretical background

A function $y = f(i_n, \dots, i_2, i_1) = f(X_f, X_b)$ is subject to simple disjoint decomposition, that is, $f(X_f, X_b) = F[g(X_b), X_f]$, if and only if the column multiplicity of the Karnaugh map (partition matrix) $\nu(X_f \mid X_b) \leq 2$, where $X_b \cup X_f = \{i_n, \dots, i_2, i_1\}$ and $X_b \cap X_f = \phi$ (Ashenhurst, 1957) (Curtis, 1962). The X_b and X_f sets are called the bound and the free set, respectively. The primary theorem of simple disjoint decomposition is the base for functional decomposition of multi-output functions. A function $f : B^n \to B^m$ is subject to decomposition if and only if the column multiplicity of the Karnaugh map (partition matrix) $\nu(Xf|Xb) \leq 2^p$, that is,

$$(X_f|X_b) \leq 2^p \Leftrightarrow f(X_f, X_b) = F[g_1(X_b), g_2(X_b), \dots, g_p(X_b), X_f], \tag{1}$$

where $X_b \cup X_f = \{i_n, \dots, i_2, i_1\}$ and $X_b \cap X_f = \phi$.

Simple disjoint decomposition becomes the basis of n-input and m-output partitioning of a combinational circuit into two blocks: bound and free (Fig. 1).

As a result of the partition, separate variable sets are connected with the bound and the free block, respectively. The number of connections between these blocks (p) corresponds to that of bound functions $g_1, g_2, \dots, g_p$.

It is obvious that the method of searching for the decomposition depends on a function representation. In the case of BDDs, the essence of searching for a simple serial decomposition is to find an appropriate horizontal diagram cutting (Fig. 2). The variables which are above the cutting line are associated with a bound set, while those which are below the cutting line are connected with a free set.

It turns out that the column multiplicity $\nu(X_f|X_b)$ of the Karnaugh map (table partition) is equal to the number of cut nodes of the BDD (Scholl, 2001). Cut nodes are situated below the cutting line and are indicated by the edges coming from the upper part of a diagram. The number of cut nodes, which is equal to the column multiplicity of the Karnaugh map $\nu(X_f|X_b)$, defines the number of necessary bound functions in accordance with Eqn. (2),

$$p = \log_2 \lceil \nu(X_f|X_b) \rceil, \tag{2}$$

Simple serial decomposition is the basis of the partition in which a single bound block and a single free block are present. When analysing various partitions, several may be chosen by finding the appropriate complex decomposition (Curtis, 1962). Two basic models of complex decompositions are known: iterative and multiple.

Theorem 1. (Iterative decomposition) *A function $f : B^n \to B^m$ is subject to q various decompositions, that is,*

$$f = F_1[G_1(X_{bq}, X_{fq-1}, X_{fq-2}, \dots, X_{f1}), X_f],$$
$$f = F_2[G_2(X_{bq}, X_{fq-1}, \dots, X_{f2}), X_{f1}, X_f],$$
$$\vdots$$
$$f = F_q[G_q(X_{bq}), X_{fq-1}, X_{fq-2}, \dots, X_{f2}, X_{f1}, X_f],$$

where

$$\begin{aligned} G_1(X_{bq}, &X_{fq-1}, \dots, X_{f2}, X_{f1}) \\ &= [g_{1\_1}(X_{bq}, X_{fq-1}, \dots, X_{f2}, X_{f1}), \\ &\quad g_{1\_2}(X_{bq}, X_{fq-1}, \dots, X_{f2}, X_{f1}), \\ &\quad \dots, \\ &\quad g_{1\_p_1}(X_{bq}, X_{fq-1}, \dots, X_{f2}, X_{f1})], \end{aligned}$$

$$\begin{aligned} G_2(X_{bq}, &X_{fq-1}, \dots, X_{f2}) \\ &= [g_{2\_1}(X_{bq}, X_{fq-1}, \dots, X_{f2}), \\ &\quad g_{2\_2}(X_{bq}, X_{fq-1}, \dots, X_{f2}), \\ &\quad \dots, \\ &\quad g_{2\_p_2}(X_{bq}, X_{fq-1}, \dots, X_{f2})], \end{aligned}$$

$$\vdots$$

$$G_q(X_{bq}) = [g_{q\_1}(X_{bq}), g_{q\_2}(X_{bq}), \dots, g_{q\_p_q}(X_{bq})],$$

if, and only if, $X_{bq}, X_{fq-1}, \dots, X_{f1}, X_f$, are mutually disjoint. Then

$$f = F_0[G_1[G_2 \dots [G_{bq}(X_q), X_{fq-1}], \dots, X_{f1}), X_f]. \tag{3}$$

The above theorem (proved by Curtis (1962)) serves as a background to draw up an algorithm for multi-level

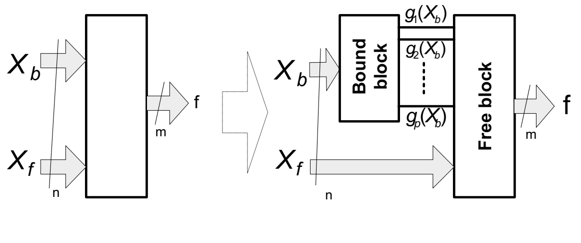
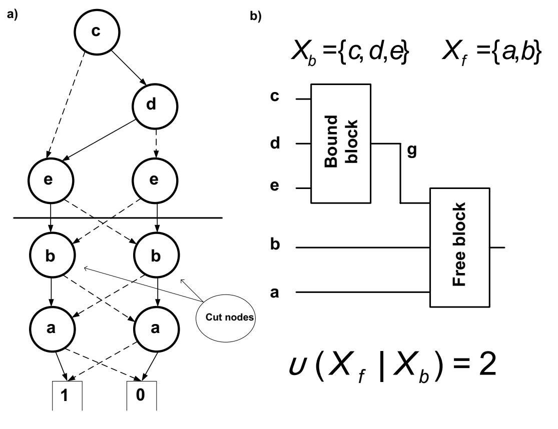


Fig. 1. Simple disjoint decomposition of a multi-output function.



a)
c
d
e
e
b
b
a
a
1
0
Cut nodes
b)
$X_b = \{c, d, e\}$ $X_f = \{a, b\}$
c
d
e
Bound block
g
b
a
Free block
$\upsilon(X_f | X_b) = 2$

Fig. 2. Simple serial decomposition using a BDD: function description (a), result of implementing the function (b).

implementation of multi-output logic functions by means of LUT blocks.

The usage of this decomposition model leads to the structure presented in Fig. 3.

Obviously, this type of decomposition can be carried out by a cyclic search for a simple serial decomposition in the subsequent steps. The application of this type of decomposition in the synthesis process has negative influence on the delays of the obtained structures. Fortunately, a multiple decomposition is free from this defect. Its essence is searching for the partition variables into common disjoint bound subsets.

Theorem 2. (Multiple decomposition) *A function $f : B^n \to B^m$ is subject to q different decompositions:*

$$\begin{aligned} f &= F_1[G_1(X_{b1}), X_{b2}, X_{b3}, \ldots, X_{bq}, X_f], \\ f &= F_2[X_{b1}, G_2(X_{b2}), X_{b3}, \ldots, X_{bq}, X_f], \\ &\vdots \\ f &= F_q[X_{b1}, X_{b2}, X_{b3}, \ldots, G_q(X_{bq}), X_f], \end{aligned}$$

where

$$\begin{aligned} G_1(X_{b1}) &= [g_{1\_1}(X_{b1}), g_{1\_2}(X_{b1}), \ldots, g_{1\_p_1}(X_{b1})], \\ G_2(X_{b2}) &= [g_{2\_1}(X_{b2}), g_{2\_2}(X_{b2}), \ldots, g_{2\_p_2}(X_{b2})], \\ &\vdots \\ G_q(X_{bq}) &= [g_{q\_1}(X_{bq}), g_{q\_2}(X_{bq}), \ldots, g_{q\_p_q}(X_{bq})], \end{aligned}$$

if, and only if, $X_{bq}, X_{bq-1}, \ldots, X_{b1}, X_f$ are mutually disjoint. Then

$$f = F[G_1(X_{b1}), G_2(X_{b2}), \ldots, G_q(X_{bq}), X_f]. \tag{4}$$

The use of this theorem (proved by Curtis (1962)) leads to the circuits partition presented in Fig. 4.

Many synthesis strategies carried out in FPGA structures of LUT type use the elements of iterative or multiple decomposition in a direct or an indirect way. It turns out that the process of searching for an appropriate multiple decomposition may be directed towards looking for an effective technology mapping to flexible logic blocks included in the FPGA. In this case, it is vital to find the multiple decomposition as fast as possible. In the case of logic functions given in the form of a BDD, there are two alternative methods of searching for such decomposition: the method based on a cyclic change of variable ordering (classic method) and the multiple cutting method.

The core of the classic method is carrying out simple serial decomposition in a cyclic way by a single cutting of a BDD. It is also important to provide variable ordering in which the nodes corresponding to bound functions are below the cutting line (Opara, 2008). An alternative method uses a multiple cutting of a BDD. In the first step, the cutting of a diagram on several levels at the same time is performed. Separate BDD sub-diagrams between cutting lines are called BDD extracts. It turns out that the segments are associated with various forms of BDDs. If they have one root and more than two multi-bit terminal nodes, we call them MTBDDs (multi-terminal BDDs) (Mikusek and Dvorak, 2009; Mikusek, 2009; Scholl *et al.*, 2001). However, if they have more than one root but only two terminal nodes, they are called SBDDs (shared BDDs) (Minato *et al.*, 1990; Ochi *et al.*, 1991; Thornton *et al.*, 1999). When there is more than one root and more than two multi-bit terminal nodes in a given extract, such a BDD extract can be described as an SMTBDD (shared multi-terminal BDD) (Kubica and Kania, 2016; 2015; Kubica, 2014; Babu and Sasao, 1998).

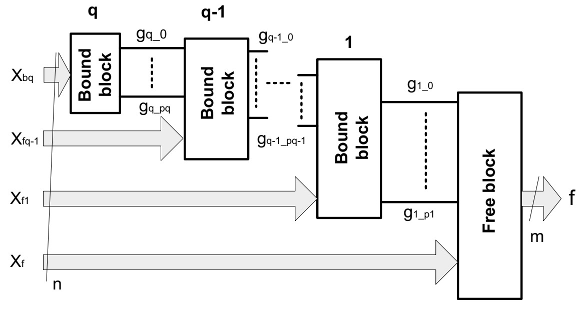


Fig. 3. Structure of the circuit after the usage of iterative decomposition.

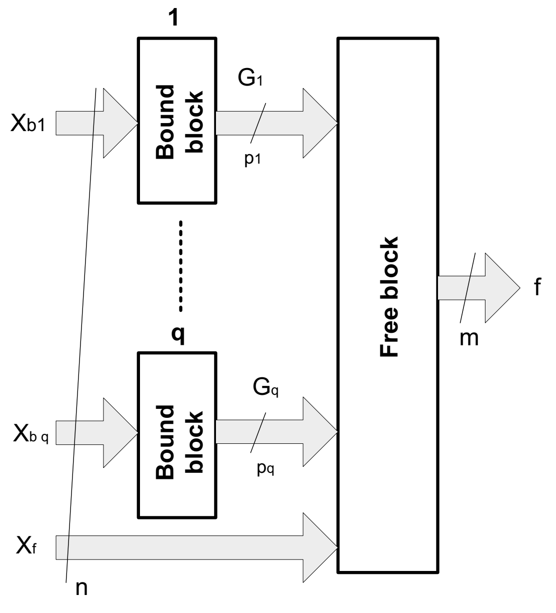


Fig. 4. Structure of the circuit after using multiple decomposition.

Example 1. Let us consider a double cutting of a diagram presented in Fig. 5. As a result of this process, three segments, which are an MTBDD, an SMTBDD, and an SBDD, are obtained.

Let E_i be the variable set for the i-th extract of a diagram. The elements of set E_i create bound sets (X_{bi}) of multiple decomposition. The choice of the i-th bound set is justified when $card(Ei)$ is greater than the number of bound functions $(numb\_of\_g)$ connected with a given extract. Thus, the method of indicating the number of bound functions for separate extracts becomes vital. If a given extract is a diagram including only one root, to indicate the number of bound functions it is enough to determine only the number of cut nodes. In the case of diagrams that have more roots (i.e., the SMTBDD), it is necessary to define the column multiplicity of a root table (Kubica and Kania, 2015). The root table can be defined as a combination of cut nodes that correspond to the paths of the SMTBDD. It is worth mentioning that the starting points of the SMTBDD paths are roots corresponding to separate lines in the root table. In Fig. 6, the essence of creating a root table as well as defining column multiplicity for a sample SMTBDD is presented.

Multiple decomposition makes it possible to implement several bound blocks into LUTs simultaneously. LUT cells can have different numbers of inputs. Therefore, the core of mapping to logic blocks is the appropriate choice of cutting lines, resulting in the minimum number of cut nodes. ♦

3. Logic synthesis oriented to LUT-based logic blocks

Configurable logic blocks (CLBs) are the main logic resources of the FPGA. In general, a CLB consists of a few logical cells (called the slice, adaptive logic module (ALM), logic element (LE), etc.). A typical elementary cell is based on LUTs. At present, it is possible to modify the functionality of configurable logic blocks, especially the number of inputs of LUTs. In the XC3000 CLB (Xilinx, 1997), a single 5-input LUT (LUT5/1) or two 4-input LUTs (LUT4/2) with shared inputs are implemented. In Spartan (Xilinx, 2011), a similar configuration of the CLB is possible but the inputs of LUT4/2 are independent. In the most technologically advanced FPGAs, very flexible blocks, such as the ALM (Altera, 2012), are embedded.

3.1. Configuration features of logic cells. Configuration capabilities of contemporary logic cells have already been described in many scientific papers (Anderson *et al.*, 2012; Garg *et al.*, 2005; Mao *et al.*, 2011; Rohani and Zarandi, 2009). One of their characteristic features resulting in better configurability is a considerably higher number of inputs compared with older constructions. Blocks that include seven or more inputs are now widely available (Lattice, 2012). The following example of ALM-based blocks included in the popular FPGA Stratix series by Altera shows the configurations abilities of moderns CLBs. Blocks of this FPGA may be configured in six different ways, as illustrated in Fig. 7.

Possible configurations of ALM-based blocks may be divided into two groups. The first one is characterized by existing independent LUT-based blocks. In this

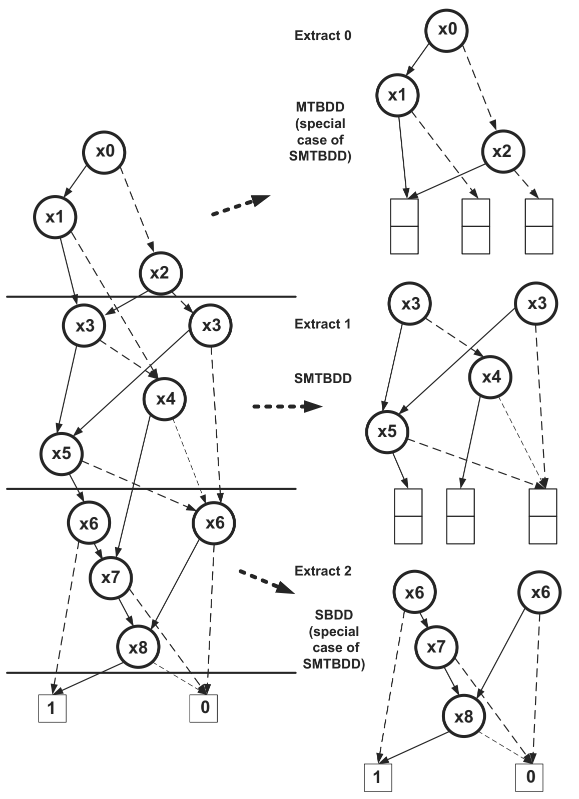
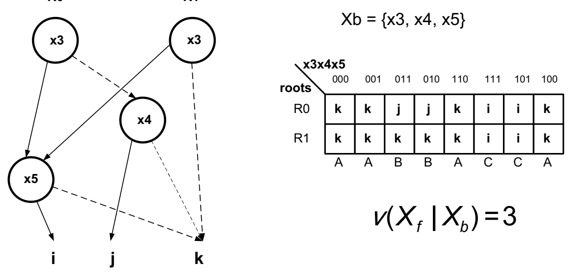


Fig. 5. Multiple ROBDD cutting—various forms of extracts.



roots \ x3x4x5	000	001	011	010	110	111	101	100
R0	k	k	j	j	k	i	i	k
R1	k	k	k	k	k	i	i	k
	A	A	B	B	A	C	C	A

Fig. 6. Essence of determining column multiplicity of a root table associated with an SMTBDD.

case, there are no common inputs for LUT-based blocks included in the ALM-based block. An ALM-based block can carry out independent functions whose arguments create disjoint variable subsets. The configurations presented in Figs. 7(a), (c), and (d) may be classified into this group. The second configuration group is characterized by the existence of a given number of common inputs. These common inputs give ALM blocks with more inputs. The configurations of this group are presented in Figs. 7(b), (e), and (f). The above shown flexibility guarantees more effective implementation than in the case of firm logic blocks.

3.2. Decomposition models directed towards configuration capabilities of logic cells.

Let k be the number of logic blocks inputs. The essence of decomposition is the choice of appropriate cutting line in BDDs. In the case of the simple serial decomposition, cutting line should be chosen on the k-th level from a root, which is clearly depicted in Fig. 8. Such a choice of the cutting line makes cardinality of bound set elements equal to k. In this way, all of the LUT-based blocks inputs are used.

In the case of multiple decomposition carried out using the multiple cutting method, cutting levels should be chosen in such a way that the numbers of elements of separate bound sets $(E_0, \ldots, E_n)$ correspond to those of inputs of LUTs for the chosen configuration $(k_0, \ldots, k_n)$. The idea of such a cutting is presented in Fig. 9.

It can be observed that multiple decomposition carried out using the multiple cutting method is a suitable choice for ALM-based configuration blocks in which independent LUT-based blocks are present (Fig. 7(a), (c), (d)). For instance, by carrying out decomposition defined by cutting lines at levels 3 and 8 or levels 5 and 8 counting from the root of the BDD, the configurations presented in Fig. 10 will be found.

The following example shows that the choice of the cutting level is essential from the point of view of the number of bound functions.

Example 2. Let us consider the decomposition of the function $f(x_0, x_1, \ldots, x_6)$ described using the BDD that is illustrated in Fig. 11(a) into logic blocks of LUT 4/1 type.

Two alternative diagram cuttings are possible (Fig.11(a)). In the first one, the zero part is connected with the bound set $X_b = E_0 = \{x0, x1, x2\}$ (A cutting line). In the second one, the zero part is associated with the bound block $X_b = E_0 = \{x0, x1, x2, x3\}$ (B cutting line). The A and C cutting lines create parts for which $card(E_0) = 3$ and $card(E_1) = 4$, and lead to multiple decomposition in which three bound functions exist (Fig. 11(b)). The B and C cutting lines create parts for which $card(E_0) = 4$ and $card(E_1) = 3$, results in five bound functions (Fig. 11(c)). The solution from Fig. 11(c) is

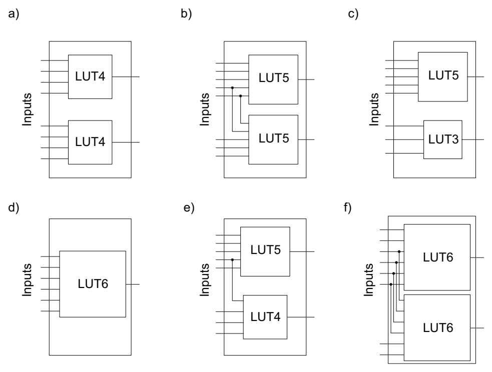


Fig. 7. ALM-based blocks configurations (Altera, 2012).

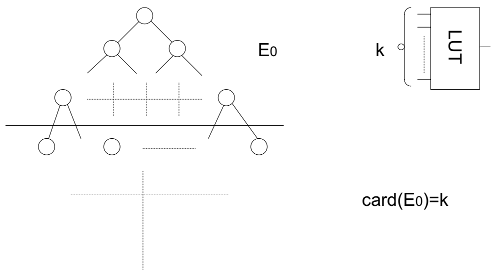


Fig. 8. Core of technology mapping to a block that has k inputs for simple serial decomposition.

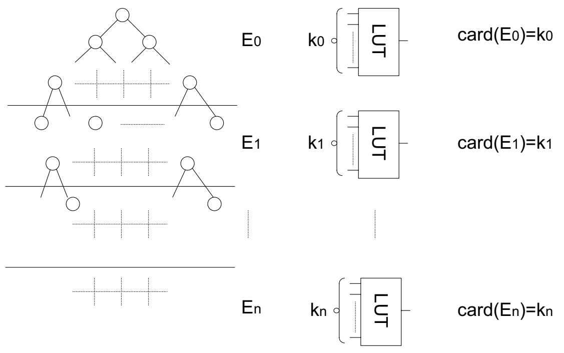


Fig. 9. Essence of technology mapping for decomposition carried out using several cutting lines $(E_0, \ldots, E_n)$: separate bound sets.

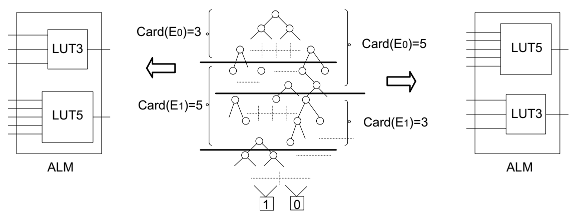


Fig. 10. Technology mapping for an ALM-based cell.

much worse in terms of the number of LUT-based blocks. ♦

In the above example, it can be seen that different cuttings of a BDD give different mapping results in terms of effectiveness. Thus, to choose an optimum solution, some monotone coefficient of mapping efficiency is necessary.

3.3. Choice of the decomposition path based on the mapping efficiency coefficient.

While mapping a function to resources of the programmable structure used, it is necessary to take into consideration the number of LUT inputs and outputs and the CLB structure.

Let us consider carrying out decomposition of the function $f : B^n \to B^m$ into LUT-based FPGA structures including CLBs in which one of two configurations of LUTs is possible. It is symbolically described as LUT 5/1 or LUT 4/2. While searching for the best technology mapping, it is necessary to adjust the decomposition process to resources of the structure used. The problem of mapping is based on the choice of an appropriate decomposition path that should be carried out in such a way as to use the lowest number of configurable logic cells. Minimization of the inputs of the free block is required, too. That is why the coefficient of mapping efficiency may be defined by

$$\delta = numb\_of\_blocks - (card(X_b) - numb\_of\_g), \quad (5)$$

where $numb\_of\_blocks$ indicates the number of CLBs used in the i-th stage and $card(X_b)$ the number of bound sets ($numb\_of\_g$ indicates the number of bound functions).

Let us consider the i-th stage of decomposition of function $f : B^n \to B$. As a result of decomposition defined by an ordered pair $(card(X_b), numb\_of\_g)$, a circuit, in which a free block has $n - (card(X_b) - numb\_of\_g)$ inputs, is obtained and described by the function $f' : B^{n-(card(X_b)-numb\_of\_g)} \to B$. Therefore, the expression $(card(X_b) - numb\_of\_g)$ stands for the number by which the number of function arguments has been reduced before the $(i + 1)$-th stage of decomposition. The three parameters of the mapping efficiency coefficient δ correspond to three different aspects of the decomposition process. $Card(X_b)$ results from the strategy of partitioning arguments, $(numb\_of\_g)$ is the effect of coding cut nodes, while $(numb\_of\_blocks)$ depends on the logic block configuration used.

Let us present the value of the mapping efficiency coefficient δ in the form of a triangular table in which the rows are associated with the number of bound functions $(numb\_of\_g)$ and the columns are connected with the cardinality of a bound set $card(X_b)$. The values of the coefficient δ calculated in accordance with Eqn. (5) are placed in separate table cells (Fig. 12).

The process of searching for appropriate decomposition results directly from the values included in the triangular table is presented in Fig. 12. The lower the parameter δ, the better the mapping of a function to a circuit structure. The usage of the table from Fig. 12 in the process of searching for an appropriate decomposition path for technology mapping of a function in CLB LUT 5/1 and LUT 4/2 will be discussed using the following example.

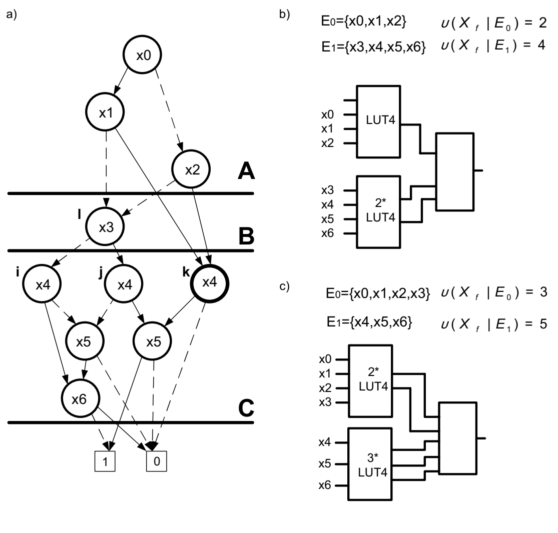

a)
x0
x1
x2
A
l
x3
B
i
j
k
x4
x4
x4
x5
x5
x6
C
1
0
b)
E0={x0,x1,x2} υ(X_f | E_0) = 2
E1={x3,x4,x5,x6} υ(X_f | E_1) = 4
x0
x1
x2
LUT4
x3
x4
x5
x6
2*
LUT4
c)
E0={x0,x1,x2,x3} υ(X_f | E_0) = 3
E1={x4,x5,x6} υ(X_f | E_1) = 5
x0
x1
x2
x3
2*
LUT4
x4
x5
x6
3*
LUT4

Fig. 11. Multiple decomposition carried out using the multiple cutting method: ROBDD diagram that underwent multiple cutting (a), blocks associated with the first logic level for the A cutting line (b), blocks connected with the first logic level for the B cutting line (c).

numb_of_g \ card(Xb)	5	4	3	2
1	-3	-2.5	-1.5	-0.5
2	-1	-1	0	
3	1	0.5		
4	3			

Fig. 12. Triangle table used to evaluate mapping efficiency for configurable logic blocks LUT 5/1 and LUT 4/2.

Example 3. Let us consider a logic function described with the use of the BDD with a given variable ordering (Fig. 13). In order to find the decomposition that will give the best technology mapping, three different cutting lines on levels 3, 4, and 5 from the root are considered. The bound sets have three, four, and five variables, respectively. Each diagram is associated with a circle in the triangular table $card(X_b)$. The search for the decomposition that will guarantee the most effective mapping should be started with the lowest number of $\delta = -3$ corresponding with the decomposition connected to a pair of numbers $(card(X_b), numb\_of\_g) = (5, 1)$.

Such a value δ exists only in the case of a five-element bound block (Fig. 13(c)). Thus, searching shall be started with the case of $card(X_b) = 5$. For diagram 13(c), there are five cut nodes. In order to distinguish them, three bound functions are needed. The value of the δ factor for decomposition $(card(X_b), numb\_of\_g) = (5, 3)$ is 1. This case is indicated in the triangular table (Fig. 13(c)) with a circle. In the remaining cases, δ may take a value lower than 1 for partition of the set of arguments for which $card(X_b) < 5$. For bound sets in which the number of elements is lower than five, the minimal value $\delta = -2.5$ is reached for a four-element set in which $(card(X_b), numb\_of\_g) = (4, 1)$. For the corresponding cutting line (diagram 13b), there are three cut nodes. Thus, it is necessary to use two bound functions in order to distinguish them.

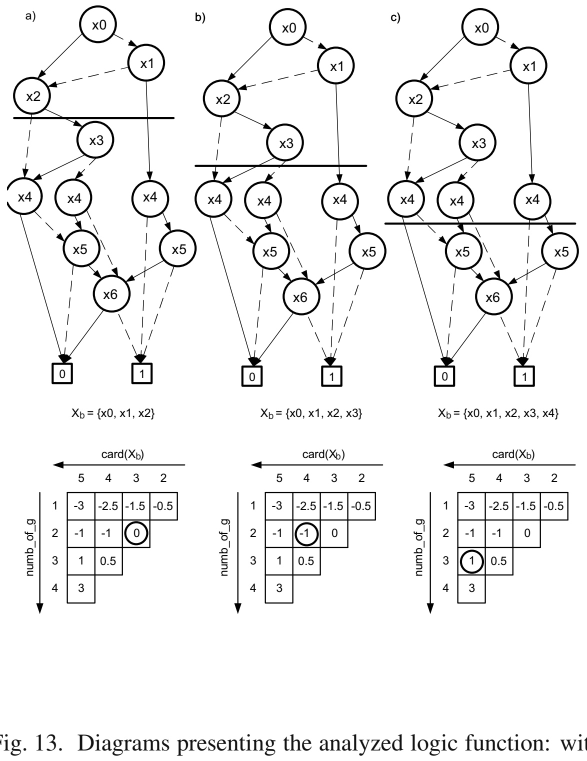
For $(card(X_b), numb\_of\_g) = (4, 2)$, the value $\delta = -1$ and is lower than the coefficient δ obtained in the previous stage of the analysis. We mark this value with a circle. For a three-element bound set $(card(X_b), numb\_of\_g) = (3, 1)$, there is only one case in which $\delta < -1$. In Fig. 13(a), it can be seen that the decomposition for which $card(X_b) = 3$ needs two bound functions. This means that no better solution than $\delta = 0$ has been found. An appropriate symbol is placed in the triangular table in Fig. 13(a).

There is no point in considering the case in which $card(X_b) = 2$ because the corresponding column of the table has only one element equal to $-0,5$ which is higher than the value already obtained for $card(X_b) = 4$. In such a situation, decomposition in which the bound set has four elements, which results in two bound functions $((card(X_b), numb\_of\_g) = (4, 2))$, guarantees the best mapping. The method of calculating the coefficient δ for LUT5/1 and LUT 4/2 blocks may also be applied for another configurable blocks. The presented strategy may be used in classic decomposition methods as well as in the multiple cutting method using an SMTBDD. It can be also generalized to multi-output functions.

4. Method for technology mapping optimization

The process of optimizing technology mapping is associated with techniques that enable minimization of the value of the coefficient δ. As can be seen in Eqn. (5), one of the ways of lowering the value of the coefficient δ is reducing the number of bound functions $(numb\_of\_g)$. This may be done by replacing some parts of the bound function g with the variables x associated with circuit inputs. This may cause a situation in which some inputs are connected with a bound block as well as to the free block. Such a decomposition model is called non-disjoint decomposition (Scholl, 2001). The essence of this decomposition is partitioning the variables set into a bound set, a free set, and common set $X_s = X_b \cap X_f$.

Non-disjoint decomposition is generalization of a simple serial decomposition in which bound and free sets are disjoint. In some cases, non-disjoint decomposition may lead to the reduction of the number of logic blocks. In the case of multi-root SMTBDDs searching for non-disjoint decomposition starts from disjoint decomposition. All the variables which belong to the SMTBDD are analysed, taking into consideration



	5	4	3	2
1	-3	-2.5	-1.5	-0.5
2	-1	-1	0	
3	1	0.5		
4	3			

Fig. 13. Diagrams presenting the analyzed logic function: with a cutting line on level 3 (a), with a cutting line on level 4 (b), with a cutting line on level 5 (c) together with triangle tables.

their ability to replace bound functions. This means joining variables to the set X_s and checking whether this is profitable in terms of number of logic blocks. When the attachment of the appropriate variable x_i to the set X_s is profitable, variable x_i itself becomes a bound function ($g_i = x_i$). The crucial part of searching for non-disjoint decomposition is judging whether the attachment of variable x_i to the set X_s leads to a reduction in the number of bound functions g.

Each variable x_i corresponds to a node at a given level in the SMTBDD. Variable x_i may take a value of 0 ($x_i = 0$) or 1 ($x_i = 1$), which is denoted by the respective coming out from a given node. These edges indicate respective sub-diagrams for $x_i = 0$ and $x_i = 1$. Each sub-diagrams points to a given number of cut nodes for a given root. There is a possibility of creating root tables for $x_i = 0$ and $x_i = 1$ for which column multiplicity may be defined. The number of different column patterns determines that of bits (bound functions) for variable value $x_i = 0$ as well as $x_i = 1$ used to distinguish them from each other. When the number of bits (bound functions) necessary to distinguish column patterns of a root table for the nodes indicated by a sub-diagram connected with $x_i = 0$ is lower than that of bits for disjoint decomposition and the number of bits for a sub-diagram associated with $x_i = 1$ fulfills the same condition, variable x_i may play the role of the bound function. ♦

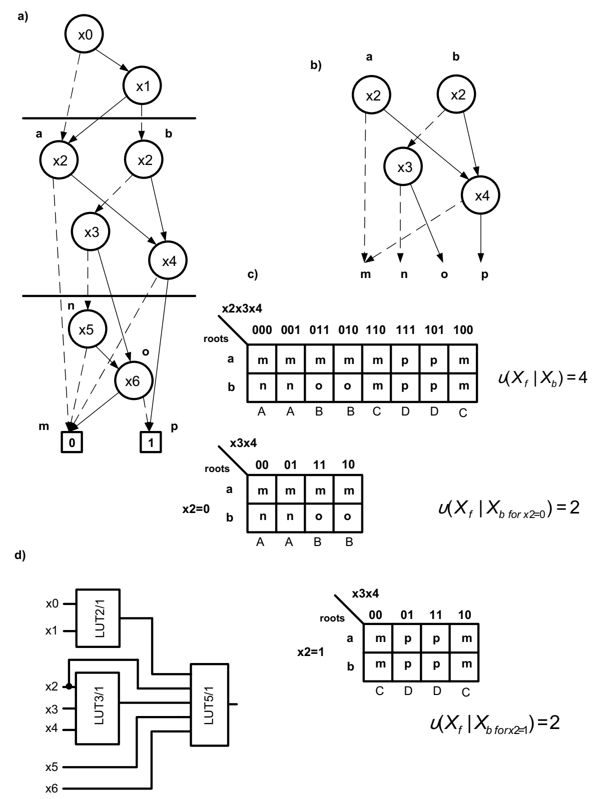
Example 4. For the function described using the diagram presented in Fig. 14(a), the part between two cutting lines was separated. This part includes three variables, $E = \{x2, x3, x4\}$. As a result of cutting, an SMTBDD, which has two roots, a and b, was created. The SMTBDD is associated with four cut nodes: m, n, o, and p (Fig. 14(b)). In order to define the number of bound functions, a root table in which four column patterns occur was created (Fig. 14(c)). Because the column multiplicity of the root table is 4, it is necessary to create two bound functions. In order to replace one of them with variable x, non-disjoint decomposition has to be found. Let us use the variable $x2$ as a switch over first. In Fig. 14(c), two root tables connected with $x2 = 0$ and $x2 = 1$, respectively, are presented. In both cases, the column multiplicity is 2. Thus, a single bit is sufficient to distinguish them (single bound function). Because of the fact that for both $x2 = 0$ and $x2 = 1$ that of bound functions is lower than the number of bound functions for disjoint decomposition, variable $x2$ may fulfill the role of a bound function. The obtained circuit structure is shown in Fig. 14(d).

♦

5. Synthesis algorithm directed towards using configurability of logic blocks

The synthesis methods described in the paper were implemented in the prototype MultiDec program. MultiDec makes it possible to conduct decomposition while taking into account technology mapping for a given FPGA structure. The program generates a description in Verilog HDL that may be used in commercial tools which carry out the final stages of synthesis (placement and routing). The essence of the MultiDec operation is presented in the form of Algorithm 1.

MultiDec uses a non-commercial CMU BDD library (Long, 2008). The choice of this library was motivated by relatively small memory usage (Miczulski, 2000). A comparison of several available libraries can be found in the paper of Long (1998). It should be mentioned that MultiDec is able to perform the synthesis process focused on specific logic cells automatically. After logic synthesis, the number of LUT-based blocks used and that of logic levels of the structure are reported.



x2x3x4 / roots	000	001	011	010	110	111	101	100
a	m	m	m	m	m	p	p	m
b	n	n	o	o	m	p	p	m
	A	A	B	B	C	D	D	C

$u(X_f \mid X_b) = 4$

x2=0	x3x4 / roots	00	01	11	10
	a	m	m	m	m
	b	n	n	o	o
		A	A	B	B

$u(X_f \mid X_{b\ for\ x2=0}) = 2$

x2=1	x3x4 / roots	00	01	11	10
	a	m	p	p	m
	b	m	p	p	m
		C	D	D	C

$u(X_f \mid X_{b\ for x2=1}) = 2$

Fig. 14. Non-disjoint decomposition in SMTBDD diagrams: ROBDD diagram together with cutting lines (a), SMTBDD diagram (b), root tables (c), structure obtained (d).

```
Algorithm 1. MultiDec.
Step 1. Read the description of the multi-output function in the PLA format and define the variable sets on which separate functions that are parts of the multi-output are dependent.
Step 2. Group initial functions into multi-output functions.
Step 3. Create MTBDDs for created multi-output functions:
for 0 to $Number\_of\_pattern\_of\_cutting\_line$ do
{
Step 4. Choose the set of cutting lines for a given number of LUT-based block inputs:
 for 0 to $Number\_of\_bdd\_ordering$ do
 {
Step 5. Change the variable ordering in the BDD.
Step 6. Cut the MTBDD (decomposition of the multi-output function).
Step 6a. Define sets E.
Step 6b. Create root tables.
Step 6c. Determine column multiplicity for separate root tables.
Step 6d. Check whether decomposition is profitable. If not, return to Step 5.
Step 6e. Search for non-disjoint decomposition.
Step 6f. Search for common $g$ functions (unicoding).
Step 6g. Determine the efficiency coefficient of technology mapping:
Compare the solution with the solutions obtained in the previous stages. The best solution, that is, the one for which efficiency coefficient $\delta$ mapping has the lowest value, is remembered.
  }endfor
 }endfor
Step. 7. Check whether multiple decomposition is found. If not, a search for decomposition with a single cutting line, for which the mapping efficiency coefficient has the minimum value, should be started (Step 4).
Step. 8. Choose the decomposition for which the mapping efficiency coefficient has the minimum value.
Step. 9. Define bound functions.
Step. 10. Was the mapping of the entire circuit carried out? If not, go to Step 2.
Step. 11. Create a description of a gained solution in Verilog.
```

6. Results of experiments

The key element of estimating the effectiveness of the analysed methods of technology mapping is comparison of the results gained with those for other academic as well as commercial tools. In the case of academic tools, it is very difficult to objectively compare the results obtained with the use of various tools supporting the synthesis process. The main reasons for this are following:

- The results of academic tools are usually presented for a chosen group of benchmarks. It often happens that the number of common tests is so small that it is difficult to get a conclusion about the advantage of one system over another.
- The minimization process may have influence on the results obtained in the decomposition process.
- The results given in the literature are not always complete, e.g., usually only the number of logic blocks used is given, without presenting synthesis time and the delays obtained for the solutions.

In the literature, comparison of academic systems is often presented for blocks that may be configured as LUT4/2 or LUT5/1. Thus, it was decided to focus the technology mapping carried out by MultiDec on such structures. It is worth mentioning that the functionality of the system is limited to very small benchmarks due to directing the MultiDec system towards multiple decomposition carried out using the multiple cutting method and limitations resulting from the non-commercial form of the BDD library. The results of the experiments performed on a group of popular benchmarks (McElvain, 1993) are presented in Table 1. In the first column, the name of the benchmark used is given. Two other columns contain the number of inputs (in) and the number of outputs (out) of a combinational circuit subjected to the synthesis process. In the next columns, the numbers of blocks obtained with the use of various academic synthesis tools are presented.

In order to compare various synthesis methods, it is worth considering the total number of blocks used and the total number of levels achieved for separate systems. In order to illustrate average reduction of the number of blocks, the following coefficient is used:

$$Profit\_A = \frac{Sum\_of\_the\_numb\_of\_blocks\_(X\,system)}{Sum\_of\_the\_numb\_of\_blocks\_(MultiDec)}. \tag{6}$$

The value $Profit\_A > 1$ indicates that the MultiDec system led to solutions which use $Profit\_A$ times less blocks than those obtained with the use of system x. The values of $Profit\_A < 1$ prove the superiority of system x. Similarly, the profit in terms of the number of levels

Table 1. Comparison of results gained with the use of the MultiDec program for academic systems for the blocks (LUT5/1 LUT4/2).

Benchmark			ILR+BDS		DDBDD		Demain		Decomp		ALTO		MultiDec	
	in	out	Bl	L	Bl	L	Bl	L	Bl	L	Bl	L	Bl	L
9sym	9	1			8	3	5	3	5	3	7	3	5	4
t481	16	1			5	2							6	5
rd73	7	3					5	2	5	2	8	2	4	3
rd84	8	4			16	3	7	3	7	3	13	3	6	3
5xp1	7	10	15	2	19	2	9	2	11	2	19	2	10	2
z5xp1	7	10											9	3
con1	7	2											3	2
sqr6	6	11											12	2
inc	7	9											20	2
sqn	7	3											11	3
misex1	8	7			14	2	8	1	10	3	14	2	11	3
f51m	8	8	14	3			10	2	10	3	15	3	7	3
b12	15	9											17	3
z4m1	7	4											6	3
x2	10	7											11	3
clip	9	5			32	3	16	4			33	3	16	4
misex2	25	18					24	3			37	2	31	3
ldd	9	19											21	2
Profit_A			1.71		1.74		0.93		1.12		1.62		1.00	
Profit_L				1.00		0.71		0.80		0.89		0.80		1.00

was calculated:

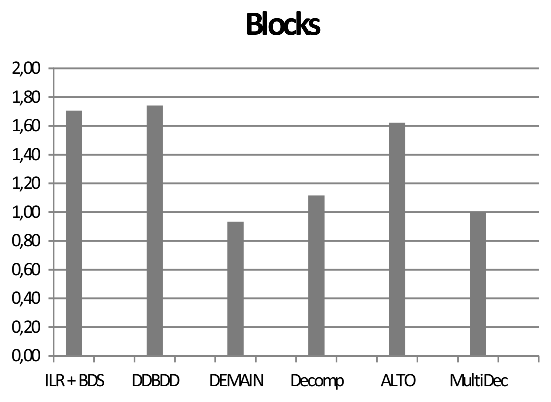
$$Profit\_L = \frac{Sum\_of\_the\_numb\_of\_levels\_(Xsystem)}{Sum\_of\_the\_numb\_of\_levels\_(MultiDec)}. \tag{7}$$

In order to indicate appropriate values of $Profit\_A$ and $Profit\_L$, only those benchmarks are taken into account for which the result is given for both system x and MultiDec. Figure 15 illustrates graphically comparison of separate academic synthesis systems.

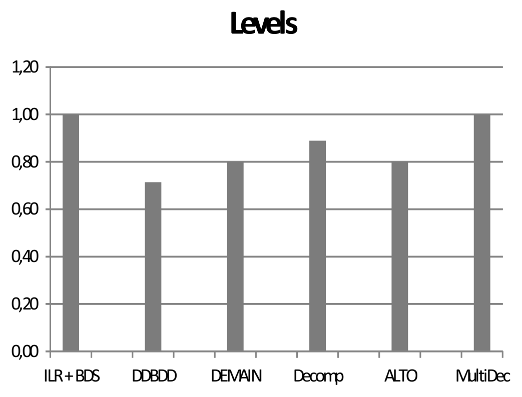
In the chart in Fig. 15(a), it can be observed that the majority of academic tools lead to worse solutions than the MultiDec system with regard to the area. In extreme cases such as the DDBDD (Cheng *et al*., 2007), ILR+BDS (Tang *et al*., 2007), and ALTO (Huang *et al*., 2000), the MultiDec system obtained results that are 1.74, 1.71, and 1.62 times better, respectively. For the Decomp system (Kania, 2004), the results are nearly the same. MultiDec obtained slightly better results, but this may may be due the limited number of experiments. Comparison with the DEMAIN system (Rawski *et al*., 1997) indicates that the results gained by MultiDec are 7% worse. It seems that the superiority of DEMAIN systems results from the implementation of the algorithms which allow parallel decomposition to be carried out. Taking into account the number of logic levels (15(b)), it turns out that in most cases the MultiDec strategy leads to solutions that are about 20% worse. One exception is the system ILR+BDS. The obtained results are nearly the same with regard to the number of logic levels as the solutions gained using the MultiDec synthesis strategy.

For the purpose of comparison of the proposed synthesis method with commercial systems, two commercial tools offered by the producers of FPGAs were chosen. These are ISE by Xilinx and Quartus by Altera. The comparison seem to be most valuable when the MultiDec system is used to carry out the initial synthesis stages and commercial systems are used for the final stages of synthesis (fitting and routing). This method of conducting the synthesis process makes it possible to use the analysed synthesis strategies immediately by an engineer.

In order to compare MultiDec with the ISE tool by Xilinx (Xilinx, 2013), it was also decided to choose technology mapping directed towards popular logic blocks included in Spartan 3 circuits (LUT4/2 or LUT5/1). The results of the comparison are presented in Table 2. The first three columns of the table contain the name of the benchmarks, the number of inputs, and the number of outputs. The column named ISE contains the synthesis results in which all of the synthesis stages were conducted using ISE (Xilinx, 2013). Two other columns (MultiDec(equ.)+ISE and MultiDec(tab.)+ISE) include synthesis results in which the initial stages of synthesis were carried out in the MultiDec system and the result was described in Verilog HDL. The difference between the columns MultiDec(equ.)+ISE and MultiDec(tab.)+ISE is the form of description of a circuit after decomposition. A decomposed circuit may be specified in two forms: either a tabular description of separate modules connected with k-input LUT-based blocks or a description with the use



(a)



(b)

Fig. 15. Comparison of the academic systems in terms of the area (a) and the number of logic levels (b).

of logic equations with no more than k variables. It seems that a description of separate modules with the use of logic equations gives more freedom in the process of additional optimization done by the ISE system. The last column of Table 2 presents the number of logic blocks obtained in MultiDec without the final structure mapping to the FPGA circuit done with the commercial tool.

The results obtained indicate that the use of the MultiDec system leads to improvement of the results. The total number of blocks obtained as a result of the synthesis of the circuit described in a table and described using equations (similar results in MultiDec columns (table description)+ISE, MultiDec(equation description)+ISE, MultiDec, is lower than the total number of the blocks obtained as a result of the synthesis carried out only in the ISE system. It is also noticeable that the Verilog description of a decomposed circuit in the form of equations gives much better results. In this case, the ISE system probably uses additional optimization possibilities related to the specificity of the circuits used. It is worth emphasizing that the expected number of blocks at the decomposition stage (MultiDec column) coincides with the results obtained for both the table description and the description with the use of equations.

The analysed synthesis strategy was also compared with Quartus II (Altera, 2010). The experiments were conducted for circuit 5CEBA2F17C7 from the Cyclone V series. This structure includes LUT-based blocks that have a maximum of seven inputs. The results of the experiments are illustrated in Table 3. They are presented in the form of the numbers of ALM blocks.

Comparing the synthesis results obtained using the analysed methods with those gained in the synthesis process carried out exclusively in the Quartus II system, the following conclusions may be drawn. First, thanks to the use of the MultiDec algorithm, it was possible to reduce the number of blocks used by about 30%. It is also worth mentioning that the results predicted at the decomposition stage (MultiDec) coincide with the final synthesis results (MultiDec(equ.)+Quartus II). On the basis of the above comparison, it can be said that there is a possibility to use the analysed methods to improve the synthesis results obtained by the Quartus II system. Comparing the MultiDec system with commercial tools indicates that the use of MultiDec at the initial synthesis stages results in better solutions. It should also be stated that the use of the technique for evaluation of mapping efficiency may be generalized to all LUT-based FPGAs.

7. Conclusion

The essence of the presented logic synthesis concepts is based on mapping the decomposition process to the configurability of logic blocks. The decomposition path is chosen on the basis of the analysis of the mapping efficiency coefficient, which makes it possible to choose the best decomposition, taking into account the use of programmable structure resources. In the technology mapping process, multiple decomposition, which is carried out using the method of multiple cutting of the BDD, plays a crucial role.

The presented experimental results clearly indicate the possibility of an improvement of the synthesis results obtained using the most popular commercial tools. The use of multiple decomposition carried out by the multiple cutting method leads to a reduction in the employed logic resources of FPGAs. Despite the fact that the paper only focuses on selected families of FPGA structures, whose CLB cells are characterized by special configuration abilities, the presented concepts of technology mapping are more general.

One of the aims of a further development of the MultiDec system is to generalize the proposed methods to a wider spectrum of FPGAs. It would be also desired to be able to carry out decomposition of circuits that have a

Table 2. Results of synthesis for the Spartan 3 circuit done in ISE.

Bench.	in	out	ISE	MultiDec(equ.)+ISE	MultiDecD(tab.)+ISE	MultiDec
9sym	9	1	8	5	5	5
t481	16	1	3	6	6	6
rd73	7	3	9	4	4	4
rd84	8	4	13	7	6	6
5xp1	7	10	12	10	10	10
z5xp1	7	10	13	9	9	9
con1	7	2	3	3	3	3
sqr6	6	11	13	12	12	12
inc	7	9	16	21	20	20
sqn	7	3	10	10	11	11
misex1	8	7	9	10	11	11
f51m	8	8	13	7	7	7
b12	15	9	15	17	17	17
z4ml	7	4	3	5	6	6
x2	10	7	8	8	11	11
clip	9	5	25	17	17	16
misex2	25	18	21	30	32	31
ldd	9	19	17	21	22	21
SUM:			211	202	209	206

Table 3. Comparison of synthesis results for the Cyclone V circuit.

Bench.	in	out	Quartus II	MultiDec(equ.)+Quartus II	MultiDec
9sym	9	1	4	3	3
t481	16	1	3	5	4.5
rd73	7	3	8	3	3
rd84	8	4	40	6	6
5xp1	7	10	12	8	8
z5xp1	7	10	13	8	8
con1	7	2	1	1	1.5
sqr6	6	11	7	7	7
inc	7	9	12	11	9
sqn	7	3	8	8	5
misex1	8	7	8	6	6.5
f51m	8	8	11	6	5.5
b12	15	9	10	11	11.5
z4ml	7	4	5	3	3
x2	10	7	7	11	6
clip	9	5	32	11	13
misex2	25	18	15	21	23
ldd	9	19	14	18	18.5
SUM:			210	147	142

considerably higher number of inputs. Moreover, it would be preferable to develop techniques reducing the number of logic levels. This would certainly minimize delays of the obtained circuits.

Taking into account the above recommendations would undoubtedly improve the effectiveness of logic synthesis focused on LUT-based FPGAs. It should also be mentioned that the present form of technology mapping algorithms implemented in the MultiDec system makes it possible to use the developed methods in the practice of circuits design.

Acknowledgment

This work was partially supported by the Ministry of Science and Higher Education funding for statutory activities (BK-220/RAu-3/2016).

References

Abouzeid, P., Babba, B., Crastes de Paulet, M. and Saucier, G. (1993). Input-driven partitioning methods and application to synthesis on table-lookup-based FPGAs, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **12**(7): 913–925.

Akers, S. (1978). Binary decision diagrams, *IEEE Transactions on Computers* **C-27**(6): 509–516.

Altera (2010). Introduction to the Quartus II software, ver. 10.0, `www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/manual`.

Altera (2012). Logic array blocks and adaptive logic modules in Stratix V devices, `www2.engr.arizona.edu/~ece506/readings/project-reading/6-cad/`.

Anderson, J. and Wang, Q. (2011). Area-efficient FPGA logic elements: Architecture and synthesis, *16th Asia and South Pacific Design Automation Conference (ASP-DAC), Yokohama, Japan*, pp. 369–375.

Anderson, J., Wang, Q. and Ravishankar, C. (2012). Raising FPGA logic density through synthesis-inspired architecture, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* **20**(3): 537–550.

Ashenhurst, R. (1957). The decomposition of switching functions, *Proceedings of an International Symposium on the Theory of Switching, Cambridge, MA, USA*, pp. 74–116.

Babu, H.M.H. and Sasao, T. (1998). Shared multi-terminal binary decision diagrams for multiple-output functions, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **81**(12): 2545–2553.

Brayton, R. and Mishchenko, A. (2010). ABC: An academic industrial-strength verification tool, *in* T. Touili *et al.* (Eds.), *Proceedings of the 22nd International Conference on Computer Aided Verification, CAV'10*, Springer-Verlag, Berlin/Heidelberg, pp. 24–40, DOI: 10.1007/978-3-642-14295-6_5.

Bryant, R. (1986). Graph-based algorithms for Boolean function manipulation, *IEEE Transactions on Computers* **C-35**(8): 677–691.

Chang, S.-C. and Marek-Sadowska, M. (1992). Technology mapping via transformations of function graphs, *IEEE 1992 International Conference on Computer Design: VLSI in Computers and Processors, Washington, DC, USA*, pp. 159–162.

Chen, D. and Cong, J. (2004). DAOMAP: A depth-optimal area optimization mapping algorithm for FPGA designs, *IEEE/ACM International Conference on Computer Aided Design, ICCAD-2004, San Jose, CA, USA*, pp. 752–759.

Cheng, L., Chen, D. and Wong, M. (2007). DDBDD: Delay-driven BDD synthesis for FPGAs, *44th ACM/IEEE Design Automation Conference, DAC'07, San Diego, CA, USA*, pp. 910–915.

Cong, J. and Ding, Y. (1994). FlowMap: An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **13**(1): 1–12.

Cong, J. and Minkovich, K. (2007). Optimality study of logic synthesis for LUT-based FPGAS, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **26**(2): 230–239.

Curtis, H. (1962). *A New Approach to the Design of Switching Circuits*, Chin Jih, Princeton, NJ.

Fiser, P. and Schmidt, J. (2009). The case for a balanced decomposition process, *12th Euromicro Conference on Digital Systems Design (DSD), Patras, Greece*, pp. 601–604.

Fiser, P. and Schmidt, J. (2012). On using permutation of variables to improve the iterative power of resynthesis, *10th International Workshop on Boolean Problems (IWSBP), Freiberg, Germany*, pp. 107–114.

Francis, R., Rose, J. and Chung, K. (1990). CHORTLE: A technology mapping program for lookup table-based field programmable gate arrays, *27th ACM/IEEE Design Automation Conference, Orlando, FL, USA*, pp. 613–619.

Garg, V., Chandrasekhar, V., Sashikanth, M. and Kamakoti, V. (2005). A novel CLB architecture and circuit packing algorithm for logic-area reduction in SRAM-based FPGAs, *Asia and South Pacific Design Automation Conference ASP-DAC 2005, Shanghai, China*, Vol. 2, pp. 791–794.

Huang, J.-D., Jou, J.-Y. and Shen, W.-Z. (2000). Alto: An iterative area/performance tradeoff algorithm for LUT-based FPGA technology mapping, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* **8**(4): 392–400.

Kania, D. (2004). Decomposition elements dedicated for LUT-based FPGAs, *Archiwum Informatyki Teoretycznej i Stosowanej* **16**(1): 45–62.

Karplus, K. (1993). Xtmap: Generate-and-test mapper for table-lookup gate arrays, *Compcon Spring'93, San Francisco, CA, USA*, pp. 391–399.

Kubica, M. (2014). *Decomposition and Technology Mapping Using Binary Decision Diagrams*, PhD thesis, Silesian University of Technology, Gliwice, (in Polish).

Kubica, M. and Kania, D. (2015). New concept of graph for function decomposition, *IFAC Conference on Programmable Devices and Embedded Systems, PDES 2015, Cracow, Poland*, pp. 61–66.

Kubica, M. and Kania, D. (2016). Decomposition of multi-output functions oriented to configurability of logic blocks, *Bulletin of the Polish Academy of Sciences: Technical Sciences*, (accepted).

Lai, Y.-T., Pan, K.-R. and Pedram, M. (1996). OBDD-based function decomposition: Algorithms and implementation, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **15**(8): 977–990.

Lattice (2012). Lattice ECP3 family data sheet, `www.latticesemi.com/.../LatticeSemi/.../DataSheets/Lattice/LatticeECP3EAFamilyData`.

Liang, Y.-Y., Kuo, T.-Y., Wang, S.-H. and Mak, W.-K. (2012). Almmap: Technology mapping for FPGAs with adaptive logic modules, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **31**(7): 1134–1139.

Long, D. (1998). The design of a cache-friendly BDD library, *IEEE/ACM International Conference on Computer-Aided Design, 1998, San Jose, CA, USA*, pp. 639–645.

Long, D. (2008). Carnegie Mellon University BDD Library, `http://www.cs.cmu.edu/afs/cs/project/modck/pub/www/`.

Mao, Z., Chen, L., Wang, Y. and Lai, J. (2011). A new configurable logic block with 4/5-input configurable LUT and fast/slow-path carry chain, *IEEE 9th International Conference on ASIC (ASICON), Xiamen, China*, pp. 67–70.

McElvain, K. (1993). IWLS'93 benchmark set: Version 4.0, `https://ddd.fit.cvut.cz/prj/Benchmarks/IWLS93.pdf`.

Micheli, G.D. (1994). *Synthesis and Optimization of Digital Circuits*, 1st Edn., McGraw-Hill Higher Education, New York, NY.

Miczulski, P. (2000). Analysis of the efficiency of BDD libraries, *International Scientific Symposium for Students and Young Scientists, Zielona Góra, Poland*, pp. 65–71, (in Polish).

Mikusek, P. (2009). Multi-terminal BDD synthesis and applications, *International Conference on Field Programmable Logic and Applications, FPL 2009, Prague, Czech Republic*, pp. 721–722.

Mikusek, P. and Dvorak, V. (2009). Heuristic synthesis of multi-terminal BDDs based on local width/cost minimization, *12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, DSD'09, Patras, Greece*, pp. 605–608.

Minato, S., Ishiura, N. and Yajima, S. (1990). Shared binary decision diagram with attributed edges for efficient Boolean function manipulation, *27th ACM/IEEE Design Automation Conference, Orlando, FL, USA*, pp. 52–57.

Murgai, R., Shenoy, N., Brayton, R. and Sangiovanni-Vincentelli, A. (1991). Improved logic synthesis algorithms for table look up architectures, *IEEE International Conference on Computer-Aided Design, ICCAD-91, Santa Clara, CA, USA*, pp. 564–567.

Ochi, H., Ishiura, N. and Yajima, S. (1991). Breadth-first manipulation of SBDD of Boolean functions for vector processing, *28th ACM/IEEE Design Automation Conference, San Francisco, CA, USA*, pp. 413–416.

Opara, A. (2008). *Decomposition Synthesis Methods of Combinational Circuits using Binary Decision Diagrams*, PhD thesis, Silesian University of Technology, Gliwice, (in Polish).

Opara, A. and Kania, D. (2010). Decomposition-based logic synthesis for PAL-based CPLDs, *International Journal of Applied Mathematics and Computer Science* **20**(2): 367–384, DOI: 10.2478/v10006-010-0027-1.

Rawski, M., Jozwiak, L., Nowicka, M. and Luba, T. (1997). Non-disjoint decomposition of boolean functions and its application in FPGA-oriented technology mapping, *23rd EUROMICRO Conference EUROMICRO 97: New Frontiers of Information Technology, Budapest, Hungary*, pp. 24–30.

Ray, S., Mishchenko, A., Een, N., Brayton, R., Jang, S. and Chen, C. (2012). Mapping into LUT structures, *Proceedings of the Conference on Design, Automation and Test in Europe, DATE'12, San Jose, CA, USA*, pp. 1579–1584.

Rohani, A. and Zarandi, H. (2009). A new CLB architecture for tolerating SEU in SRAM-based FPGAs, *International Conference on Reconfigurable Computing and FPGAs, ReConFig'09*, pp. 83–88.

Sasao, T. and Butler, J. (1996). A method to represent multiple-output switching functions by using multi-valued decision diagrams, *26th International Symposium on Multiple-Valued Logic, Santiago De Compostela, Spain*, pp. 248–254.

Scholl, C. (2001). *Functional Decomposition with Application to FPGA Synthesis*, Kluwer Academic Publishers, Norwell, MA.

Scholl, C., Becker, B. and Brogle, A. (2001). The multiple variable order problem for binary decision diagrams: Theory and practical application, *Proceedings of the Design Automation Conference, Asia and South Pacific, Yokohama, Japan*, pp. 85–90.

Tang, W.-C., Lo, W.-H. and Wu, Y.-L. (2007). Further improve excellent graph-based FPGA technology mapping by rewiring, *IEEE International Symposium on Circuits and Systems, ISCAS 2007, New Orleans, LA, USA*, pp. 1049–1052.

Thornton, M., Williams, J., Drechsler, R., Drechsler, R. and Wessels, D. (1999). SBDD variable reordering based on probabilistic and evolutionary algorithms, *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, Canada*, pp. 381–387.

Vemuri, N., Kalla, P. and Tessier, R. (2002). BDD-based logic synthesis for LUT-based FPGAs, *ACM Transactions on Design Automation of Electronic Systems* **7**(4): 501–525, DOI: 10.1145/605440.605442.

Wan, W. and Perkowski, M.A. (1992). A new approach to the decomposition of incompletely specified multi-output functions based on graph coloring and local transformations and its application to FPGA mapping, *Proceedings of the Conference on European Design Automation, EURO-DAC'92, Hamburg, Germany*, pp. 230–235.

Wyrwoł, B. and Hrynkiewicz, E. (2013). Decomposition of the fuzzy inference system for implementation in the FPGA structure, *International Journal of Applied Mathematics and Computer Science* **23**(2): 473–483, DOI: 10.2478/amcs-2013-0036.

Xilinx (1997). XC3000 technical information, xapp024, `www.xilinx.com/support/documentation/application_notes/xapp024.pdf`.

Xilinx (2013). ISE Design Suite 14, UG631, `www.xilinx.com/products/design-tools/ise-design-suite.html`.

Yang, C. and Ciesielski, M. (2002). BDS: A BDD-based logic optimization system, *IEEE Transactions on*

Computer-Aided Design of Integrated Circuits and Systems **21**(7): 866–876.

Marcin Kubica received his MSc and PhD degrees from the Silesian University of Technology, Gliwice, Poland, in 2010 and 2014, respectively. He has been an assistant professor at the University of Bielsko-Biała. His main interests and research areas involve programmable devices and systems, and logic synthesis.

Dariusz Kania received his MSc and PhD degrees from the Silesian University of Technology, Gliwice, Poland, in 1989 and 1995, respectively. He initially worked as an assistant lecturer (1989–1995) and then as an assistant professor (1995–2004). He has been a professor at the Silesian University of Technology and a full professor since 2006 and 2014, respectively. His main interests and research areas involve programmable devices and systems, logic synthesis and optimization dedicated to a wide range of programmable logic devices (CPLD, FPGA), and the implementation of digital circuits. He is also interested in applications of computer posturography in postural control diagnostics and motor functions rehabilitation.

Received: 6 October 2015
Revised: 1 April 2016
Re-revised: 13 October 2016
Accepted: 24 October 2016