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

Evolutionary design and optimization of combinational digital circuits with respect to transistor count

A. SŁOWIK* and M. BIAŁKO

Department of Electronics and Computer Science, Technical University of Koszalin, 2 Śniadeckich St., 75-453 Koszalin, Poland

Abstract. In the paper an application of evolutionary algorithm to design and optimization of combinational digital circuits with respect to transistor count is presented. Multiple layer chromosomes increasing the algorithm efficiency are introduced. Four combinational circuits with truth tables chosen from literature are designed using proposed method. Obtained results are in many cases better than those obtained using other methods.

Key words: artificial intelligence, evolutionary algorithm, combinational digital circuits, design, optimization.

1. Introduction

In the case of combinational digital circuit design we can distinguish two optimization criteria: based on gate count, and based on transistor count. In both cases minimization leads to the decrease of physical implementation costs of a given circuit. In the case when desired logic function has to be realized using existing processing elements (PE), contained in e.g. FPGA circuit, the minimization of gate number is especially recommended. In this case if the circuit has lower number of gates, then lower number of PE will be required to its realization. However, the minimization of transistor number is especially important in the case of direct circuit implementation in silicon. In this case, the lower number of transistors, leads to lower circuit size on chip and production cost will be cheaper too.

Among design methods of combinational digital circuits, two of them are most popular: Karnaugh Maps [1], and Quine-McCluskey method [2,3]. Recently also evolutionary algorithms are used to the discussed problem. The process of evolutionary circuit design is fundamentally different form traditional design process, because it is not based on designer knowledge and experience, but on the evolution process [4]. The evolutionary circuit design has less constraints than the design based on designer knowledge and experience; the designers are not only limited by the technology in which the circuit will be produced, but also by own habits (routines), intelligence, imagination and creative thinking [4]. An application of evolutionary methods to circuit design allows to escape from limitations characterised earlier and to obtain the access to the new possibilities [4]. Among evolutionary methods of combinational circuit design we can enumerate algorithms: NGA (Genetic Algorithm with N-cardinality representation) [5,6], MGA (Multiobjective Genetic Algorithm) [7], or the MLCEA (Multi-Layer Chromosome Evolutionary Algorithm) [8] introduced by the authors of this paper. Design and optimization of combinational digital circuits based on minimum number of gates were the main goal of these algorithms.

In this paper evolutionary method useful for design and optimization of combinational digital circuits, with regard to transistor number, is described. In presented algorithm a new representation of multilayer chromosomes is introduced. This method, being the modification of the algorithm MLCEA, is named MLCEA-TC (Multi-Layer Chromosome Evolutionary Algorithm – Transistor Count). Results obtained using the proposed method are compared with the results obtained by other methods.

2. Multiple-layer chromosome

Electronic systems are composed of sub-blocks (differential amplifiers, modulators, logic gates, etc.) that are characterized by a set of some features (parameters) such as inverting and non inverting inputs of differential amplifiers, inputs of multiple-input logical gates, width and length of MOS transistors, and so on. In a design process, when genetic algorithm are used for optimization of the system structure, the features of the system sub-blocks (sub-circuits) are usually represented by genes in a single-layer chromosome. Then, during the crossover operation the set of the features can be "broken" destroying the internal structure of the sub-block and its properties.

As an example of a disadvantage of the single-layer chromosome representation, let us consider digital combinational circuits realized using multiple input logical gates; each logical gate can be described by a logical function which it realizes and its input and output connections inside the digital circuit. Let us assume that during the optimization of the circuit structure two sub-circuits, shown in Fig. 1a and Fig. 1b, are chosen for crossing-over.

When single layer chromosomes are used, the structures of these sub-circuits can be represented by individuals (chromosomes), as is shown in Fig. 2, where individual A corresponds to the sub-circuit of Fig. 1a, and individual B – to the sub-circuit of Fig. 1b.

*e-mail: aslowik@ie.tu.koszalin.pl

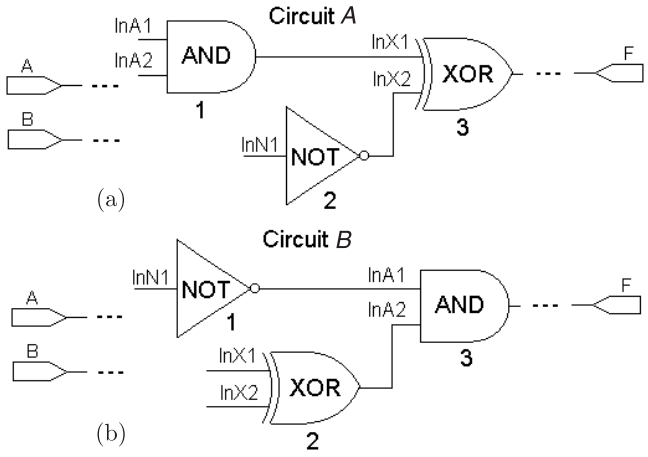

Fig. 1. Examples of combinational circuits

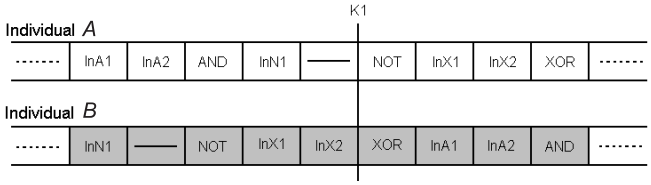

Fig. 2. Single-layer chromosomes corresponding to the circuits from Fig. 1

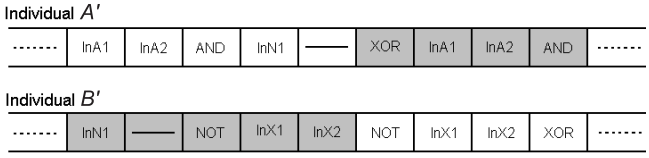

Fig. 3. Single-layer chromosomes after crossover of chromosomes from Fig. 2

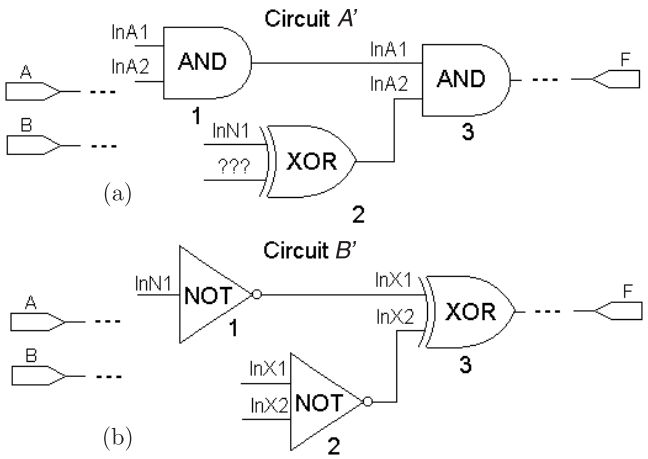

Fig. 4. Circuits corresponding to the child individuals from Fig. 3

After crossing-over in the point $K1$ we obtain new individuals *A'* and *B'* with chromosomes shown in Fig. 3.

These new individuals represent new sub-circuit structures shown in Fig. 4.

It is seen from Fig. 4 that in the sub-circuit *A'* the second input of XOR gate is not connected, and the NOT gate in the sub-circuit *B'* has two connected inputs, even though physically it has only single input. Thus, both new individuals (sub-circuits) are unacceptable solutions. Thus, it is required to apply repair procedures to eliminate this disadvantage.

Therefore, because of these reasons, we have introduced the *multiple-layer* chromosomes in which the whole set of features (parameters) representing given sub-block is coded in a single column of the chromosome. A general concept of the multiple-layer chromosome for representation of individuals (possible solutions of the circuit) is shown in Fig. 5.

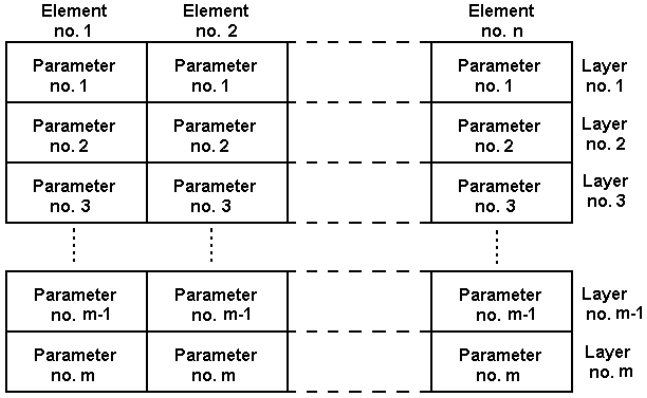

Fig. 5. Structure of multilayer chromosome

It consists of n elements (sub-blocks of the circuit), each having m features (parameters) located in the single column. Thus, during the crossover operation the whole sub-block can be transferred (moved) to another place of the system structure without influencing its internal structure. An utilisation of multiple-layer chromosomes to the problem discussed in the example (Figs. 1–4) leads to a pair of individuals, shown in Fig. 6.

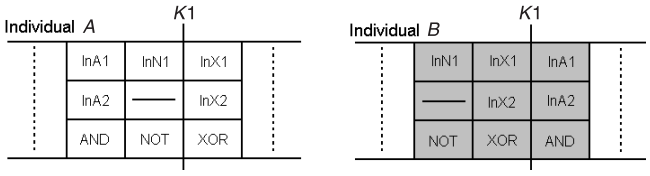

Fig. 6. Multi-layer chromosomes corresponding to the circuits form Fig. 1

Similarly as in previous case the individual *A* represents the circuit *A* of Fig. 1a, and the individual *B* corresponds to the circuit *B* of Fig. 1b.

After crossover of the individuals *A* and *B* in the randomly chosen point $K1$, the pair of child chromosomes *A'* and *B'*, shown in Fig. 7, are obtained.

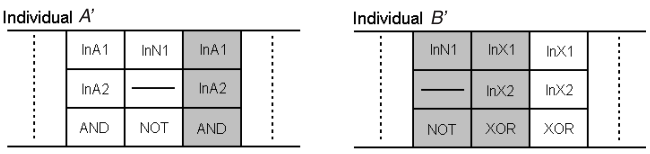

Fig. 7. Multi-layer chromosomes after crossover from Fig. 6

The circuits *A'* and *B'*, corresponding to the individuals of Fig. 7 are shown in Fig. 8.

From Fig. 7, and especially from Fig. 8 we can see, that thanks to the application of multi-layer chromosomes it is possible to transfer whole gates, along the circuit with all their parameters. Thanks to this the gates are not "torn", and both child individuals are acceptable solutions, so the repair procedures are not necessary.

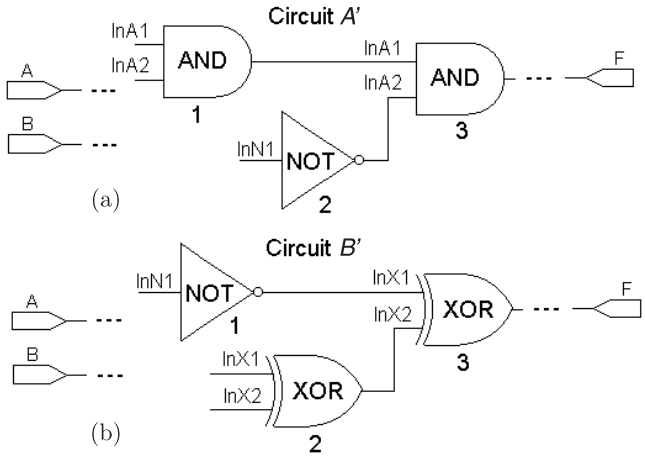

Fig. 8. Circuits corresponding to the child individuals of Fig. 7

3. The MLCEA-TC method

The MLCEA-TC method is different than the MLCEA method [8] elaborated earlier, since now the minimization of the number of transistors, rather than number of gates, is the criterion of the optimisation. Similarly as in work [9,10] the following set of gates is used: NOT, NOR, XOR, NAND, DC (direct connection of gate input with its output) instead of the set: NOT, OR, XOR, AND, DC. The set of the gates was changed, because physical implementation of NAND and NOR gates requires lower number of transistors than AND and OR gates, and also because of result comparability of the presented method and the results obtained in the work [10].

The MLCEA-TC method is operating in the following way. In order to create an initial population, we create a pattern (template) of the designed circuit, which structure is shown in Fig. 9a. The structure and coding of chromosomes representing the pattern is shown in Fig. 9b.

In the section of the chromosome marked as "Input no. x" we put the number of the circuit input or the gate number from the pattern, which output is to be connected to this input; in the place "Gate Type no. x" we put one of the five digits, which represent respectively: 1-NOT gate, 2-NAND gate, 3-XOR gate, 4-NOR gate, 5-direct connection (DC) of a given gate input with its output. Digit "0" represents lack of connection of a given gate input in the pattern, and digit "6" represents the circuit output. In the place "Output no. x" we put the pattern gate number, which output is to be connected to this circuit output. All circuit inputs are represent by negative number, that is, the first input is represented by the number "–1", etc.

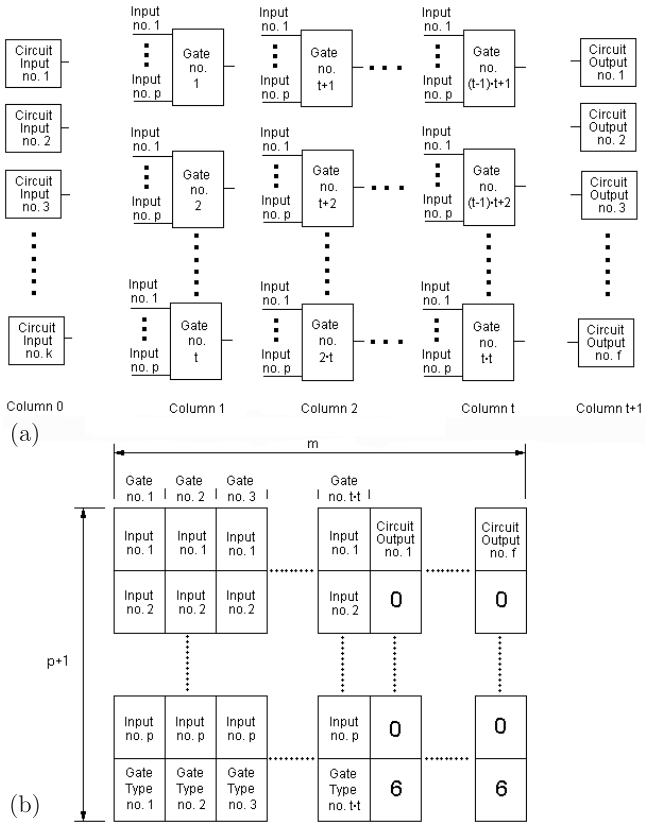

Fig. 9. Structure of circuit coding: a-pattern, b-multilayer chromosome

Each individual (chromosome) in the population represents selected combinational circuit. For example in Fig. 10 the three-input circuit realizing the truth table of "Circuit no. 1" of Table 1 is shown. The pattern (template) of the two-input gates, with connections between them corresponding to this circuit, is shown in Fig. 11; this pattern is represented in a population by the multi-layer chromosome, shown in Fig. 12, in which each column corresponds to one gate of the template. In Fig. 11, and Fig. 12 the gates from the circuit (Fig. 10) are marked by grey colour.

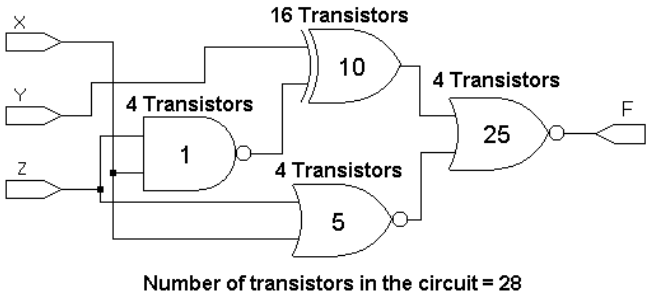

Fig. 10. Example of digital circuit fulfilling the truth table for "Circuit no. 1" from Tab. 1

The pattern size *t* for designed circuit is determined experimentally using following formulae:

$$t = MAX(NI, NO) \tag{1}$$

where: NI – number of circuit inputs, NO – number of circuit outputs. In the case when the computed size *t* not allows to find the circuit fulfilling a given truth table, then it is increased by one, and the design process is repeated.

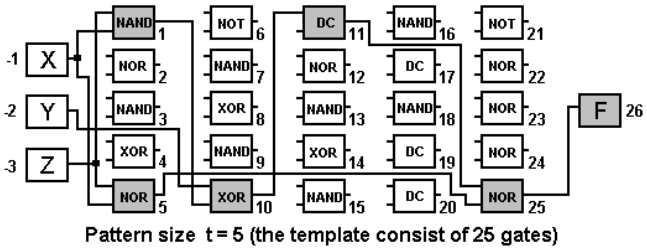

Fig. 11. Pattern of gates corresponding to the circuit from Fig. 10

-3	-1	-3	-1	-3	2	1	-1	-2	-2	10	1	1	-2	6	13	9	11	6	5	16	9	9	13	11	25
-1	-2	-3	-3	-1	0	4	2	5	1	0	7	2	-3	2	3	0	6	0	0	0	7	3	13	5	0
2	4	2	3	4	1	2	3	2	3	5	4	2	3	2	2	5	2	5	5	1	4	4	4	4	6

Fig. 12. Multilayer chromosome corresponding to the pattern of gates from Fig. 11

The MLCEA-TC algorithm operates in two phases. After creation the initial population each individual is evaluated using following objective function $FC1$:

$$fc1_i = \begin{cases} v \cdot j, & \text{when} \quad O(c_i) \neq C_i \\ 0, & \text{when} \quad O(c_i) = C_i \end{cases} \tag{2}$$

$$FC1 = \sum_{i=1}^{I} fc1_i + Sc \cdot \left(t^2 \cdot NT_{MAX} - NT\right) \tag{3}$$

where: v – positive real number (during experiments assumed $v = 10$), c_i – i-th vector of combination of input signals (truth table), $O(c_i)$ – circuit response for vector c_i applied to the circuit input, C_i – correct response vector (truth table), j – number of differences in the particular positions of vector $O(c_i)$ and vector C_i, I – number of input vectors in the truth table, t – pattern size, NT – number of transistors in designed circuit, NT_{MAX} – maximum number of transistors corresponding to the gate form the used set of gates, Sc – scaling coefficient determined by following formulae (during experiments assumed $Sc = 0.007$):

$$0 < Sc < \frac{v}{t^2 \cdot NT_{MAX}} \tag{4}$$

The value of the $FC1$ function is bigger when given individual (circuit) is fulfilling the given truth table in the smaller rate. The function $FC1$ specifies the number of constraints that follow from the truth table, which are not fulfilled by a particular individual (first factor of $FC1$ function). Besides, we are looking for circuits that fulfil the constraints posed by the truth table, but with the highest number of transistors allowed by the circuit pattern size (second factor of $FC1$ function). This treatment increases a search space in the second phase of the algorithm. The algorithm is minimizing this function during its operation. In the case when the circuit fulfilling the truth table is not found ($FC1 \geqslant v$) following operators are applied: crossover, mutation and fan selection [9]. The crossover operator depends on cutting all layers of two randomly chosen individuals in one randomly chosen point, and exchange of the cut fragments between them. Thanks to the application of multi-layer chromosomes, cutting of the whole gate of the pattern is possible without damage of its structure is possible. The mutation operator causes a random change of the gate type in the pattern (in the case of choosing the gene belonging to the last layer of the chromosome to the mutation) or change of the connections between gates in the pattern (when the gene from remaining layers is chosen to the mutation).

In the case when the circuit fulfilling the truth table ($FC1 < v$) is found, the objective function is changed to $FC2$ and the algorithm starts the second phase of its operation. Minimization of the transistor number in the circuit is the goal of the new objective function $FC2$, which is determined by the following formulae:

$$FC2 = \begin{cases} NT, & \text{when} \quad FC1 < v \\ w + NT, & \text{when} \quad FC1 \geqslant v \end{cases} \tag{5}$$

where: w – value of penalty, which should be higher than $t^2 \cdot NT_{MAX}$ (during experiments it is assumed $w = 10^5$), remaining symbols are identical as in formulae (2) and (3).

The algorithm minimizes the function $FC2$. In the second phase of the algorithm the elitist selection, instead fan selection, was used in order not to lose the solution (the circuit fulfilling the truth table) found during the first phase of the algorithm. The algorithm termination criterion was its convergence (lack of changes of the best solution).

4. Description of experiment

Four test digital circuits were chosen (identical as in work [8]) in experiments. In Table 1–4 the truth tables for each designed circuit are presented. Symbol "In" represent inputs, and symbol "O" corresponds to circuit outputs. During performed experiments the parameters of evolutionary algorithm were: population size = 100, crossover probability = 0.5, mutation probability = 0.05, fan selection [9] coefficient $a = 0.3$. The pattern size was determined experimentally according to the formulae (1); for all circuits $t = 5$ was used (the pattern consists of 25 gates).

In Tables 5–8 results obtained using MLCEA-TC (marked as M-TC) method and other methods: Human Design (HD), Genetic Algorithm (GA) (taken from works [8, 10]) are presented. Symbol GA-TC represents results obtained using method described in paper [10]. In Tab. 5 – 8 symbols are as follows: "NG" – number of gates, "NT" – number of transistors, and character " ' " in logic function description represents negation. All circuits were designed using two-input gates. Similarily as in work [10], it is assumed that NOT gates consist of 2 transistors, NAND and NOR gates of 4 transistors, and XOR gates consist of 16 transistors. The results obtained by MLCEA-TC method require: 42 generations for circuit no. 1, 967 generations for circuits no. 2, 549 generations for circuits no. 3, 875 generations for circuit no. 4.

Table 1
Truth table for circuit no. 1

Circuit no. 1			
In			O
X	Y	Z	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Table 2
Truth table for circuit no. 2

Circuit no. 2				
In				O
Z	W	X	Y	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

Table 3
Truth table for circuit no. 3

Circuit no. 3				
In				O
A	B	C	D	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Table 4
Truth table for circuit no. 4

Circuit no. 4						
In				O		
A_1	A_0	B_1	B_0	X_2	X_1	X_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

Table 5
Results obtained for circuit no. 1

Method	Obtained logic function	NG	NT
HD	$\mathrm{F} = (\mathrm{X}' \cdot \mathrm{Y} \cdot \mathrm{Z}) + (\mathrm{X} \cdot (\mathrm{Y} \oplus \mathrm{Z}))$	6	42
GA	$\mathrm{F} = (\mathrm{Z} \cdot (\mathrm{X} + \mathrm{Y})) \oplus (\mathrm{X} \cdot \mathrm{Y})$	4	34
GA-TC	$\mathrm{F} = (((\mathrm{Y} \cdot \mathrm{Z})' \oplus \mathrm{X}) + (\mathrm{Y} + \mathrm{Z})')'$	4	28
M-TC	$\mathrm{F} = (((\mathrm{X} \cdot \mathrm{Z})' \oplus \mathrm{Y}) + (\mathrm{X} + \mathrm{Z})')'$	4	28

Table 6
Results obtained for circuit no. 2

Method	Obtained logic function	NG	NT
HD	$\mathrm{F} = ((\mathrm{Z}' \cdot \mathrm{X}) + (\mathrm{Y}' \cdot \mathrm{W}'))$ $+ ((\mathrm{X}' \cdot \mathrm{Y})(\mathrm{Z} \oplus \mathrm{W}'))$	11	70
GA	$\mathrm{F} = (((\mathrm{W} \oplus \mathrm{Y}) + (\mathrm{W} \cdot \mathrm{X}))$ $\oplus((\mathrm{Z} + \mathrm{X} + \mathrm{Y}) \oplus \mathrm{Z}))'$	8	74
GA-TC	$\mathrm{F} = ((\mathrm{Z} \oplus \mathrm{Y}) + (\mathrm{X} + \mathrm{Y})' \oplus ((\mathrm{X} \cdot \mathrm{Y})' \cdot \mathrm{W})'$	6	48
M-TC	$\mathrm{F} = (((\mathrm{X} + \mathrm{Y})' + \mathrm{Z})' \oplus \mathrm{Y}) \oplus ((\mathrm{X} \cdot \mathrm{Y})' \cdot \mathrm{W})'$	6	48

Table 7
Results obtained for circuit no. 3

Method	Obtained logic function	NG	NT
HD	$\mathrm{F} = ((\mathrm{A} \oplus \mathrm{B}) \oplus ((\mathrm{A} \cdot \mathrm{D})(\mathrm{B} + \mathrm{C})))$ $+ ((\mathrm{A} + \mathrm{C}) + \mathrm{D})'$	9	74
GA	$F = ((A \oplus B) \oplus A \cdot D) + (C + (A \oplus D))'$	7	68
GA-TC	$\mathrm{F} = ((\mathrm{A}+\mathrm{D})' \oplus (\mathrm{B} \oplus \mathrm{D}) \cdot ((\mathrm{C} + (\mathrm{B} \oplus \mathrm{D}))'$ $+ ((\mathrm{A} \cdot \mathrm{D})' + \mathrm{C})')')'$	8	56
M-TC	$\mathrm{F} = (((\mathrm{A} \oplus \mathrm{B}) \oplus (\mathrm{A} \cdot \mathrm{D})')$ $\cdot (((\mathrm{A} \cdot \mathrm{D})' \cdot \mathrm{D})' \cdot (\mathrm{B} + \mathrm{C})'))'$	7	52

Table 8
Results obtained for circuit no. 4

Method	Obtained logic function	NG	NT
HD	$\mathrm{X}_0 = \mathrm{A}_0 \oplus \mathrm{B}_0;$ $\mathrm{X}_1 = (\mathrm{A}_1 \oplus \mathrm{B}_1) \cdot \mathrm{B}_0' + ((\mathrm{A1} \oplus \mathrm{B1}) \oplus \mathrm{A}_0) \cdot \mathrm{B}_0$ $\mathrm{X}_2 = (\mathrm{A}_1 \cdot \mathrm{B}_1) + (\mathrm{A}_0 \cdot \mathrm{B}_0) \cdot (\mathrm{A}_1 + \mathrm{B}_1)$	12	98
GA	$\mathrm{X}_0 = \mathrm{A}_0 \oplus \mathrm{B}_0;\ \mathrm{X}_1 = (\mathrm{A}_0 \cdot \mathrm{B}_0) \oplus (\mathrm{A}_1 \oplus \mathrm{B}_1)$ $\mathrm{X2} = (\mathrm{A}_1 \cdot \mathrm{B}_1) + (\mathrm{A}_0 \cdot \mathrm{B}_0) \cdot (\mathrm{A}_1 \oplus \mathrm{B}_1)$	7	72
GA-TC	—	—	—
M-TC	$\mathrm{X}_0 = \mathrm{A}_0 \oplus \mathrm{B}_0;\ \mathrm{X}_1 = (\mathrm{B}_1 \oplus \mathrm{A}_1)' \oplus (\mathrm{A}_0 \cdot \mathrm{B}_0)'$ $\mathrm{X}_2 = (((\mathrm{A}_0 \cdot \mathrm{B}_0)' + (\mathrm{B}_1 \oplus \mathrm{A}_1)')' \cdot (\mathrm{B}_1 \cdot \mathrm{A}_1)')'$	9	68

5. Conclusions

From Tables 5–8 we can see that using the proposed method, the design and optimization of combinational digital circuits with respect to transistors count is possible. Results obtained using MLCEA-TC method are better (circuits consist of lower number of transistors in 9 cases on 11 possible) or comparable (in other 2 cases) than the results obtained using other methods. Also we can notice that the minimization of transistor number in the circuit does not lead to the minimization of gate number, what we can see in Table 8 (circuit no. 4) especially. The realization of logic function obtained using GA require 7 gates (72 transistors), but this same logic function obtained using MLCEA-TC (M-TC) method require 9 gates (68 transistors).

Acknowledgments. This work was supported by the Polish Ministry of Scientific Research and Information Technology (MNiI) under Grant No. 3 T11B 025 29.

References

[1] M. Karnaugh, "A map method for synthesis of combinational logic circuits", *AIEE Trans. Communications and Electronic* 72 (I), 593–599 (1953).

[2] W.V. Quine, "A way to simplify truth function", *American Mathematical Monthly* 62 (9), 627–631 (1955).

[3] E.J. McCluskey, "Minimization of Boolean function", *Bell Systems Technical Journal* 35 (5), 1417–1444 (1956).

[4] J. Greene, "Simulated evolution and adaptive search in engineering design", in *2^{nd} Online Workshop on Soft Computing*, 1997.

[5] C. A. Coello, A. D. Christiansen, and A. H. Aguirre, "Use of evolutionary techniques to automate the design of combinational circuits", *Int. J. Smart Engineering System Design* 2 (4), 299–314 (2001).

[6] C. A. Coello, A. D. Christiansen, and A. H. Aguirre, "Automated design of combinational logic circuits using genetic algorithms", *Proc. Int. Conf. on Artificial Neural Nets and Genetic Algorithms*, 335–338 (1997).

[7] C. A. Coello, A. H. Aguirre, and B. P. Buckles, "Evolutionary multiobjective design of combinational logic circuits", *Proc. 2^{nd} NASA/DoD Workshop on Evolvable Hardware*, 161–170 (2000).

[8] A. Słowik and M. Białko, "Design and optimization of combinational digital circuits using modified evolutionary algorithm", *Proc. 7^{th} Int. Conf. on Artifficial Intelligence and Soft Computing* 3070, 468–473 (2004).

[9] A. Słowik and M. Białko, "Modified version of roulette selection for evolution algorithm – the fan selection", *Proc. 7^{th} Int. Conf. on Artifficial Intelligence and Soft Computing ICAISC* 3070, 474–479 (2004).

[10] P. Nilagupta and N. Ou-thong, "Logic function minimization base on transistor count using genetic algorithm", *Proc. 3^{rd} Information and Computer Engineering Postgraduate Workshop*, Songkla, Thailand, 2003.