DE GRUYTER OPEN

Journal of ELECTRICAL ENGINEERING, VOL 67 (2016), NO1, 29–35

NOVEL APPROACH TO SYNTHESIS OF LOGIC CIRCUITS BASED ON MULTIFUNCTIONAL COMPONENTS

Adam Crha — Richard Růžička — Václav Šimek *

Multifunctional logic continuously becomes an important way how to implement compact and cheap circuits with intrinsic reconfiguration features. Polymorphic electronics concept with its substantial technological independency opens a way to fulfil this objective through the adoption of emerging semiconductor technologies and advanced synthesis methods. The paper comes with a proposal of a novel synthesis method oriented on the exploitation of polymorphic electronics principles. Key part of it is based on Boolean divisor identification and function kernelling technique. The proposed method is evaluated with several test circuits.

K e y w o r d s: digital circuits, reconfiguration, polymorphic electronics, synthesis methods

1 INTRODUCTION

Nowadays, it is possible to identify a lot of manifold application areas where a digital circuit with the inherent ability to perform a set of different functions at particular moments in time may prove to be a very efficient means of solution. Obviously the most immediate approach, how to address this specific need, is to design as many different circuits as the overall number of functions that are actually needed in a given situation. As a next step involved within the execution flow, individual outputs of these circuits are switched in such a way that only the presently required function will be taken into account. However, the main drawback behind this conception, and its essential limitation as well, will emerge in direct connection with the overall size of the resulting implementation on the circuit level.

Another possible answer to the outlined purpose lies in the adoption of reconfiguration principles [1, 2]. This course of action clearly enables a more flexible way (*eg* circuits that have not been prepared yet during the design phase could be implemented as well, when the conception of evolvable hardware is employed [3]) how to obtain corresponding design with noticeably improved area-aware properties. Nevertheless, the chosen procedure may turn out to be less effective in terms of the necessary processing time — in order to invoke the function change, the structure of the circuit must be adjusted and this takes a bit of time. Moreover, this approach assumes the availability of a suitable infrastructure — a reconfigurable circuit with adequate granularity of functional elements and interconnection fabrics.

Recent advancements within the field of digital design techniques and components for digital circuits provide vital evidence that yet another feasible strategy may be employed - area and time-efficient design of multifunctional circuits based on utilization of individual structural elements with multifunctional features [4]. In this case, the entity of multifunctional circuit is devised as a compact structure involving a set of multifunctional components, where their mutual, low-level interconnection scheme remains untouched in all allowable operating modes and only the active function of these components is expected to change intentionally. It is important to note that the gate-level granularity is typically chosen today for design purposes in the case of these circuits, while the individual components (gates) are conceived in most cases predominantly at a transistor level. An alternative path with promising outlook for the future is marked by the gradually increasing adoption of unconventional devices that bring significant advantages to this type of circuits — especially in terms of functional characteristics.

A special case related to these multifunctional circuits is based on the adoption of the polymorphic electronics paradigm [5]. From the technical perspective, circuits with these attributes typically change their function in accordance with the actual state of the target operating environment. The environment in this particular case is represented by a physical quantity that has a notable influence on some of the physical parameters of the electronic structures - power supply voltage level, voltage amplitude of a signal, temperature etc. It may seem quite impractical on the first sight to consider these properties for any useful circuit behaviour but, in fact, it may help to achieve new (better than existing) solutions for some application classes. The most notable benefit here dwells in the scheme of utterly distributed sensitivity to the well-established factors that bring about the intentional function change. In addition, no configuration network with a global scope or dedicated input pins of these components are required [6].

It is important to point out that the change of the active function which is executed by the polymorphic circuit

* IT4Innovations Centre of Excellence, Faculty of Information Technology, Brno University of Technology Božetěchova 2, Brno, Czech Republic, icrha@fit.vutbr.cz

DOI: 10.1515/jee-2016-0004, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

takes place immediately (with no extensive delay associated with this step) and the function change triggering mechanism that happens due to the outlined circuit sensitivity to suitable phenomena from its operating environments is therefore naturally embedded into the circuit itself. Today's applications are often based on unipolar semiconductor transistors, but the concept of polymorphic electronics is more general and allows to conveniently employ new emerging devices like graphene [7] or nanowire structures [8], ambipolar devices utilizing suitable organic polymers with semiconductor-like properties [9], *etc*, which makes it possible to obtain a new generation of advanced multi-functional logic elements.

The structure of this paper is organized as follows: the opening section clarifies the basic aspects related to the multifunctional circuits and their benefits in comparison with a conventional approach. Section 2 is briefly explaining the key theoretical aspects behind polymorphic electronics and implications for digital circuits design. A review of selected circuit synthesis methods and their properties can be found in Section 3. Then, Section 4 contains the introduction of a novel synthesis method for polymorphic circuits based on the adoption of the so-called Boolean divisors identification and function kernelling technique. The obtained results are demonstrated in Section 5. Finally, Section 6 provides the conclusions.

2 CONCEPT OF POLYMORPHIC ELECTRONICS

The notion of polymorphic electronics [5] determines, in its own essence, a standalone category of reconfigurable circuits, which represents a highly appealing prospect how to implement all the required functional properties in a resource-efficient way. In the case of these circuits assuming the principles of polymorphic electronics, various modifications in the key physical characteristics of building components (*eg* in a transistor's operation point, usage of ambipolar charge carrier conductivity) are predominantly involved behind the change of their behaviour as a straight response to the influence of external stimuli — temperature, power supply voltage, light intensity, special signal, *etc*. However, the structure of the circuit itself remains unchanged on the interconnection level for all the intended functions.

2.1 Formal background

From a formal point of view, polymorphic circuit is an electronic digital circuit which can be described by a graph defined as $G = (V, E, \varphi)$, where V is a set of vertices (ports of circuit components), $E = \{(a, b) \mid a, b \in V\}$ is a set of edges (connections in the circuit) and φ is a mapping which assigns a component from the set K to each element that belongs to V, $\varphi\colon V \to K$. Then, graph G explicitly determines interconnection of the individual components from set K and, therefore, particular structure of a given circuit which is able to realize one of the meaningful intended functions from a set $\Phi = \{F_1, \ldots, F_n\}$ and $|\Phi| > 1$.

Furthermore, let X be a physical quantity, assuming values of the real numbers domain $\mathbf{R}$ and describing an operating environment of the circuit. Mapping $\pi\colon Y \to \Phi$, where $Y = \{I_i \mid I_i \subset \mathbf{R}\}$ is a set of intervals of values of quantity X. If quantity X has a value $X(t_1) \in I_k$ at time t_1, where $I_k \subset \mathbf{R}$ is an interval from $\mathbf{R}$, then the circuit represented by graph G performs function $F_k \in \Phi$ at time t_1, briefly $\pi(I_k) = F_k$. If quantity X has a value $X(t_2) \in I_m$ at time t_2, where $I_m \subset \mathbf{R} \wedge I_m \cap I_k = \emptyset$, then the circuit represented by graph G executes function $F_m \in \Phi$ at time t_2, briefly $\pi(I_m) = F_m$. Note that even such intervals of X may exist on which the function of the circuit is not defined.

2.2 Implications for circuit design

Definition of a polymorphic circuit introduced above reveals that the structure of the circuit — graph G, *ie*, specific rendition of circuit components interconnection, always keeps its layout. On the other hand the function of the circuit is, of course, allows its transition from one mode to another, and the function of individual components must therefore vary for different modes (functions to be performed). So the key to the circuit polymorphism lies in the set of fundamental building components. These are exactly the devices that change their function in accordance with the value of the physical quantity describing the actual state of the environments. This observation is regarded as a key pillar of the approach. It also makes the whole concept more universal and independent of specific technology used for implementation of the logic.

3 SYNTHESIS METHODS FOR POLYMORPHIC CIRCUITS

Synthesis methods of ordinary digital circuits have to solve the problem of interconnection graph G searching just for one particular function F. If a suitable canonical form of F is found, the structure of G can be easily inferred from it. For polymorphic circuits, this approach tends to exhibit higher complexity because just one graph needs to cover already several functions from the existing set $\Phi = \{F_1, \ldots, F_n\}$, which makes up the given circuit and fulfil the demand of multifunctional operation (see details in Section 2). The task to find the same form for all the functions F_1 to F_n (with different elementary functions on the same position) is, therefore, not so trivial at all.

Nowadays, design of polymorphic circuits is performed almost exclusively at a gate level. Results of practical experiments in this domain indicate that meaningful polymorphic circuits always include a combination of several polymorphic gates alongside the selection of ordinary gates. In most situations only one type of polymorphic gate (*eg* NAND/NOR) is employed in the whole design procedure [11]. The overall design efficiency (in terms of circuit size or speed) could be further improved if a more

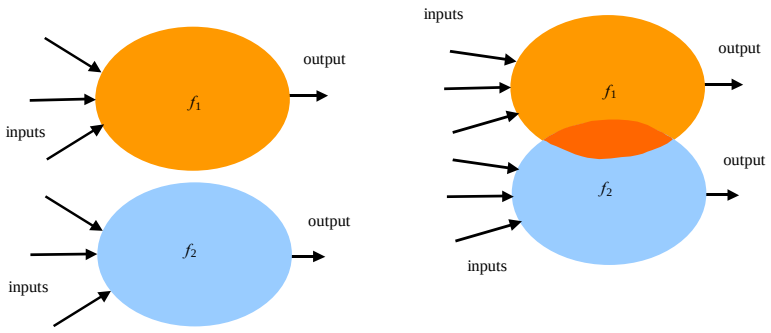

Fig. 1. Polymorphic circuit consisting of two functions — f_1 and f_2 with partial resource sharing

diverse selection of polymorphic gates (other types besides NAND/NOR variant as well) is taken into account. However, it would be definitely paid by a more complicated synthesis process due to the state space growth.

Let us also note that polymorphic circuit synthesis methods do not aim at dealing with the question of environment intentionally and how it is physically involved in the circuit operation. This is the subject delegated to the chosen and employed polymorphic gates — building components of the circuit. Simple circuits could be obviously designed by hand but the growing complexity renders this approach virtually unfeasible. Proper synthesis techniques have to be obviously considered. Conventional optimization methods for ordinary digital circuits are quite unusable. Hence the evolutionary optimization methods could bring the solution [12, 13].

3.1 Evolutionary approach

Digital circuit synthesis and optimization techniques based on thoughtful exploitation of convenient evolutionary-inspired paradigms, as demonstrated by Sekanina [14] (and before initially suggested by Miller [10], Koza [15] and Thompson [16]), could establish a way how to achieve a rather unconventional but, at the same time, interesting and useful solution. Needless to say, also the original concept of polymorphic electronics emerged virtually as a side effect of evolutionary design experiments [5]. Almost all polymorphic circuits, more complex than just a few gates, have been designed using Carthesian Genetic Programming (CGP) [10] till now.

In terms of CGP, the circuit is laid out as an array of $\boldsymbol{u}$ (columns) $\times \boldsymbol{v}$ (rows) of programmable elements (gates). The number of circuit inputs, n_i, and outputs, n_o, is fixed and no feedback is allowed. Each gate is programmed to perform one of the functions defined at the beginning of the experiment. Every individual is encoded using $\boldsymbol{u} \times \boldsymbol{v} \times 3 + n_\mathrm{o}$ integers. Only a mutation operator is applied which changes one gene of the encoded chromosome. The fitness function is constructed to minimize the Hamming distance between the output vectors of a candidate circuit and the required output vectors. Typically, all possible input vectors are applied to obtain the set of output vectors for the two required functions F_1 and F_2.

3.2 Conventional methods

One of the first examples of conventional design methods focused on polymorphic circuits was introduced by Gajda [11]. The first of these methods involves the so-called polymorphic multiplexing. This approach falls on the borderline between conventional and polymorphic digital circuits. For each function, a digital circuit is synthesized and the outputs of these circuits are then multiplexed by a polymorphic multiplexor. The principle is shown in Fig. 1. The structure of a circuit designed by this method shows a relatively low optimality. However, possible workaround towards the desirable improvement dwells in the partial sharing of some logic resources, as it is also shown in Fig. 1. The output stage is based on polymorphic multiplexers which deliver the corresponding result with regard to the execution mode.

In addition to that, Gajda [11] proposed a method of polymorphic circuit synthesis utilizing binary decision diagrams (BDD). The method is called PolyBDD. Its core part is using Multi-terminal BDD (MTBDD), which is an extension of binary decision diagrams. Terminal nodes of MTBDDs could contain integer values. These integers represent possible relations between the input binary value and the required output value, while elementary polymorphic sub-circuits are defined for them. For desired functions F_1 and F_2, a MTBDD is created. Then the MTBDD is converted into a circuit, where the nodes assume the role of multiplexers and the terminals are replaced by a proper polymorphic sub-circuit according to the number in a given leaf.

4 PROPOSED SYNTHESIS METHOD

The fundamental purpose behind the logic function is to provide an accurate and unambiguous specification of the target circuit behaviour. However, it may now assume the most optimal form at first. This is why such a kind of initial description could be further handed over to diverse minimization and synthesis techniques where the purpose is to achieve the best possible mapping onto the available resources or building blocks, to minimize the delay along the data path, *etc.*

n fact the essence of logic function itself can appear in several, mutually different forms. An elaborate discussion on five of the most common instances using two-level arrangement can be found in [17]. These cases are mostly focused on variations of the truth table forms together with disjunctive/conjunctive representation. For the sake of completeness it is important to note that synthesis and minimization techniques in digital circuit domain are based extensively on multi-level representations as well [18–20], especially due to a reasonable compromise between compact representation and efficient manipulation. Probably one of the most illustrative examples here is tied with decision diagrams or, to be precise, BDD (Binary Decision Diagrams) as the widely adopted scheme in various situations.

Those minimization and synthesis techniques could be, as a matter of fact, roughly classified as two-level or multi-level oriented. In the case of two-level methods the final circuit composition is delivered as the logic expressions in conjunctive or disjunctive notation. This approach then leads to the situation when input signals will only pass through two logic gates at most. On the other hand, multi-level techniques are generating the so-called nested expressions with the resulting data path (or interconnection of the individual gates) spanning even far more than two circuit elements within the final circuit arrangement.

4.1 Key aspects behind the synthesis method

In the case of all the required functions that need to be accommodated by the polymorphic circuitry, searching for the corresponding interconnection graph G (see closer explanation in Section 2) may not be an easy task at all. Nowadays several design and synthesis methods suitable for the domain of polymorphic circuits have been already introduced in order to tackle this challenge. However, each of these methods (some examples were discussed in Section 3) is evidently spoiled by certain shortcoming or limitation.

Due to this obvious reason it is highly desirable to continue with the research of minimization and synthesis techniques. Directly related to this observation is the endeavour to propose a novel synthesis method that would be addressing the weak spots of the previous attempts. The main idea behind the novel approach is based on the undeniable identification of common parts across the input circuits which are virtually shared between them as so-called common divisors by means of exploiting techniques of function kernelling [19, 21] and Boolean division [19].

A typical execution scheme of the proposed method consists of the following sequence of steps:

1) Minimized expressions in DNF representation (Disjunctive Normal Form) depict the input functions — F_1 and F_2. Both functions are initially provided in two-level PLA format as a truth table.

$$F_1 = ab\bar{d} + b\bar{c}\bar{d} + \bar{b}\bar{c}d + a\bar{c} + \bar{a}\bar{b}c\bar{d}\,, \tag{1}$$

$$F_2 = ab\bar{c} + a\bar{c}d + \bar{a}\bar{b} + \bar{b}c\bar{d} + \bar{a}c\,. \tag{2}$$

2) Intersection table at dimensions given by $m \times n$, where m denotes the number of term groups of F_1 and n has the same meaning for F_2. This table is laid out in such a way that the first column contains terms groups belonging to F_1 and the first column holds the number of terms of F_2.

F_1 \ F_2	$ab\bar{c}$	$a\bar{c}d$	$\bar{a}\bar{b}$	$\bar{b}c\bar{d}$	$\bar{a}c$
$ab\bar{d}$	$ab(\bar{d}\|\bar{c})$	$a(b\bar{d}\|\bar{c}d)$	$\emptyset$	$\bar{d}(ab\|\bar{b}c)$	$\emptyset$
$b\bar{c}\bar{d}$	$b\bar{c}(\bar{d}\|a)$	$\bar{c}(b\bar{d}\|ad)$	$\emptyset$	$\bar{d}(b\bar{c}\|\bar{b}c)$	$\emptyset$
$\bar{b}\bar{c}d$	$\bar{c}(\bar{b}d\|ab)$	$\bar{c}d(\bar{b}\|a)$	$\bar{b}(\bar{c}d\|\bar{a})$	$\bar{b}(\bar{c}d\|c\bar{d})$	$\emptyset$
$a\bar{c}$	$a\bar{c}(1\|b)$	$a\bar{c}(1\|d)$	$\emptyset$	$\emptyset$	$\emptyset$
$\bar{a}\bar{b}c\bar{d}$	$\emptyset$	$\emptyset$	$\bar{a}\bar{b}(c\bar{d}\|1)$	$\bar{b}c\bar{d}(\bar{a}\|1)$	$\bar{a}c(\bar{b}\bar{d}\|1)$

(3)

Individual boxes within the table are filled up in the following way: *group of terms intersection* (*remaining terms of* F_1 | *remaining terms of* F_2).

3) The first pass through the completed table is performed. The purpose is to identify those boxes that exhibit the mutual intersection of a maximum size, *eg minterm* $(1|1)$. The first minterm to be successfully recognized is then put at its place into the final expression. These minterms are basically common for both input functions and, thus, it is not required to deal with them in a polymorphic way. Once the minterm is registered in the final expression, corresponding row and column are eliminated from the table.

This time, no relevant minterm was found for both functions F_1 and F_2.

4) The second pass through the table constructed in step 2) is commenced. This time, the task is to find the largest intersection. The box fulfilling this requirement is then rewritten into the final expression, the whole row and column with this particular box are eliminated from the table.

F_1 \ F_2	$ab\bar{c}$	$a\bar{c}d$	$\bar{a}\bar{b}$	$\bar{b}c\bar{d}$	$\bar{a}c$
$ab\bar{d}$	$ab(\bar{d}\|\bar{c})$	$a(b\bar{d}\|\bar{c}d)$	$\emptyset$	$\bar{d}(ab\|\bar{b}c)$	$\emptyset$
$b\bar{c}\bar{d}$	$b\bar{c}(\bar{d}\|a)$	$\bar{c}(b\bar{d}\|ad)$	$\emptyset$	$\bar{d}(b\bar{c}\|\bar{b}c)$	$\emptyset$
$\bar{b}\bar{c}d$	$\bar{c}(\bar{b}d\|ab)$	$\bar{c}d(\bar{b}\|a)$	$\bar{b}(\bar{c}d\|\bar{a})$	$\bar{b}(\bar{c}d\|c\bar{d})$	$\emptyset$
$a\bar{c}$	$a\bar{c}(1\|b)$	$a\bar{c}(1\|d)$	$\emptyset$	$\emptyset$	$\emptyset$
$\bar{a}\bar{b}c\bar{d}$	$\emptyset$	$\emptyset$	$\bar{a}\bar{b}(c\bar{d}\|1)$	$\underline{\bar{b}c\bar{d}(\bar{a}\|1)}$	$\bar{a}c(\bar{b}\bar{d}\|1)$

(4)

The resulting expression obtained at this step is

$$F = \bar{b}c\bar{d}\,(\bar{a}|1) + \tag{5}$$

5) Previous step 4) is continuously repeated until the table contains uncovered boxes with at least some intersection. Once all the intersection are covered, it is possible to proceed with a next step.

F_1 \ F_2	$a\bar{c}d$	$\bar{a}\bar{b}$	$\bar{a}c$
$ab\bar{d}$	$a(b\bar{d}\|\bar{c}d)$	$\emptyset$	$\emptyset$
$\bar{b}\bar{c}d$	$\underline{\bar{c}d(\bar{b}\|a)}$	$\bar{b}(\bar{c}d\|\bar{a})$	$\emptyset$
$a\bar{c}$	$a\bar{c}(1\|b)$	$\emptyset$	$\emptyset$

(6)

Table 1. Detailed specification of the circuits in two-level PLA representation

Test #	Circuit properties #1				Circuit properties #2			
	Name	Inputs	Term groups	% $f=(1)$	Name	Inputs	Term groups	% $f=(1)$
1	parity3	3	8	50.00 %	majority3	3	8	50.00 %
2	parity5	5	32	50.00 %	majority5	5	32	50.00 %
3	4i-5t-a	4	5	31.25 %	4i-5t-b	4	5	31.25 %
4	26i-16t-a	26	16	43.75 %	26i-16t-b	26	16	50.00 %
5	inverse_outputs-a	5	32	50.00 %	inverse_outputs-b	5	32	50.00 %
6	inverse_inputs-a	5	32	50.00 %	inverse_inputs-b	5	32	50.00 %
7	con1-a	7	9	44.40 %	con1-b	7	9	55.00 %
8	12i-4096t-50p-a	12	4096	50.00 %	12i-4096t-50p-b	12	4096	50.00 %
9	12i-2048t-50p-a	12	2048	50.00 %	12i-2048t-50p-b	12	2048	50.00 %
10	12i-1024t-50p-a	12	1024	50.00 %	12i-1024t-50p-b	12	1024	50.00 %
11	12i-2048t-20p-a	12	2048	20.00 %	12i-2048t-20p-b	12	2048	20.00 %
12	12i-1024t-20p-a	12	1024	20.00 %	12i-1024t-20p-b	12	1024	20.00 %
13	7i-128t-50p-a	7	128	50.00 %	7i-128t-50p-b	7	128	50.00 %
14	7i-128t-75p-a	7	128	75.00 %	7i-128t-75p-b	7	128	75.00 %
15	7i-128t-20p-a	7	128	25.00 %	7i-128t-20p-b	7	128	25.00 %

Table 2. Results in terms of logic resources

Test #	Polymoprhic synthesis tool						Espresso — two outputs mode hfil				Utilization
	INV	2-AND	2-OR	P_MUX	P_INV	sum	INV	2-AND	2-OR	sum	
1	3	8	3	0	3	17	3	11	5	19	89.47 %
2	4	11	4	4	2	25	6	8	16	30	83.33 %
3	5	64	15	0	5	89	7	24	70	101	88.12 %
4	26	200	7	1	22	256	27	109	229	365	70.14 %
5	5	25	7	6	2	45	5	40	13	58	77.59 %
6	5	22	6	3	2	38	5	40	12	57	66.67 %
7	6	10	4	0	5	25	5	11	10	26	96.15 %
8	12	9406	965	864	12	11259	14	1286	10265	11565	97.35 %
9	12	6597	639	515	12	7775	14	1038	8701	9753	79.72 %
10	12	4047	377	259	12	4707	14	668	5804	6486	72.57 %
11	12	5707	570	497	12	6798	14	858	6984	7856	86.53 %
12	12	3168	307	254	11	3752	14	505	4279	4798	78.20 %
13	7	154	30	26	0	217	9	52	222	283	76.68 %
14	7	138	28	24	4	201	9	44	182	235	85.53 %
15	7	140	28	27	2	204	9	47	182	238	85.71 %

The largest intersection found is then moved again into the final expression:

$$F = \bar{b}c\bar{d}\,(\bar{a}|1) + b\bar{c}\,(\bar{d}|a) + \bar{c}d\,(\bar{b}|a) + \qquad (7)$$

6) Now, the table contains just the remaining groups of terms which do not have any common divisor, *ie*, they are not mutually related. It is necessary to apply a special functional block called polymorphic multiplexer (labelled as "|" in the expression), which isolates contradictory parts of functions F_1 and F_2.

F_1 \ F_2	$\bar{a}\bar{b}$	$\bar{a}c$
$ab\bar{d}$	$\emptyset$	$\emptyset$
$a\bar{c}$	$\emptyset$	$\emptyset$

(8)

The largest intersection found is then moved again into the final expression:

$$F = \bar{b}c\bar{d}(\bar{a}|1) + b\bar{c}(\bar{d}|a) + \bar{c}d(\bar{b}|a) + (ab\bar{d} + a\bar{c}|\bar{a}\bar{b} + \bar{a}c)\,. \quad (9)$$

Now, decomposition of the obtained expression will be done as a measure towards the best possible mapping onto the set of available circuit components:

$$A_2 = (\bar{a}|a) \qquad (10)$$

$$B_2 = (\bar{b}|b) \qquad (11)$$

$$C_1 = (c|\bar{c}) \qquad (12)$$

$$Z = (\bar{d}|1) \qquad (13)$$

And now, the resulting expression ready for technology mapping phase will have the following composition

$$F = \bar{b}c\bar{d}(\bar{a}|1) + b\bar{c}(\bar{d}|a) + \bar{c}d(\bar{b}|a) + A_2B_2Z + A_2C_1\,. \quad (14)$$

7) In case both functions F_1 and F_2 have dissimilar numbers of term groups, there will remain for sure a certain number of uncovered terms belonging to the function with the higher number of term groups. It is therefore necessary to include those solitary and uncovered terms into the resulting expression by means of using polymorphic operator "|" and neutral element for addition denoted as "0".

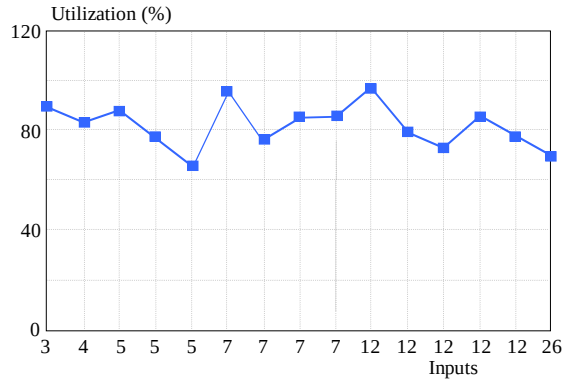

Fig. 2. Evaluation of the number of inputs in the case of test circuits from Table 2

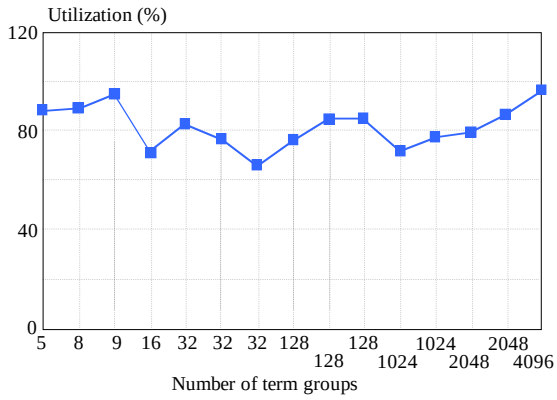

Fig. 3. Impact on the synthesis method efficiency compared with Espresso tool as 100 % reference

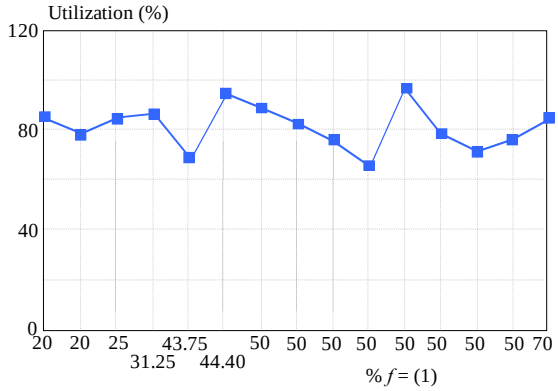

Fig. 4. Illustration of the synthesis method behaviour compared with Espresso tool as 100 %

5 EXPERIMENTAL RESULTS

The proposed synthesis technique for polymorphic circuits has been tested on several circuits defined by a truth table in a two-level PLA format. Detailed specification of these circuits that were used for evaluation of the synthesis methodcan be found in Table 1. These circuits were either randomly generated or taken from ISCAS benchmark set. The column entitled as % $f = (1)$ denotes the percentage of inputs with a logic 1 value, where some of the inputs could be labelled as do not care items and, therefore, did not influence the circuit at all.

The results gathered in Table 2 provide an insight into the efficiency of the proposed method in comparison with the conventional tool Espresso. Nearly 20 % improvement in average was demonstrated. These were obtained by means of executing the polymorphic synthesis method. Lower value in "Utilization" column signifies better solution achieved by the polymorphic approach in comparison with the conventional Espresso tool (100 % reference threshold).

Beside the direct comparison of the results delivered by the proposed synthesis method and those obtained with the help of the conventional Espresso tool (in terms of the overall circuit size comprising 2-input logic elements and some special polymorphic functional blocks - see corresponding sections in Table 2), several additional tests were performed in order to find whether the proposed method exhibits any dependence on the following attributes

- Figure 2, total number of logic inputs (test circuits defined in Table 1 are shown from left to right in the order as follows — 1, 2, 3, 5, 6, 7, 13, 14, 15, 8, 9, 10, 11, 12, 4), to show the effect on the resulting synthesis method efficiency (compared with Espresso tool as 100 % reference).
- Figure 3, number of term groups used (test circuits defined in Table 1 are shown from left to right in the order as follows — 2, 1, 7, 4, 3, 5, 6, 13, 14, 15, 10, 12, 9, 11, 8), in the case of different term groups number for the individual tests circuits as defined in Table 2.
- Figure 4, percentage of the inputs with logic 1 value (test circuits defined in Table 1 are shown from left to right in the order as follows — 11, 12, 15, 2, 4, 7, 1, 3, 5, 6, 8, 9, 10, 13, 14). Reference in the case of the different percentage of inputs with a logic 1 value is shown here for circuits defined in Table 2.

The evidence provided by Figs. 2 to 4 clearly reveals that from the perspective of different values in the case of the suggested attributes under observation, the method does not exhibit any unsuspected difference of its efficiency. In fact, it is possible to observe some local fluctuations (*eg* Fig. 2, slowly decreasing efficiency in the case of using the function with 5 and 7 inputs).

6 CONCLUSIONS

The deployment of reconfiguration, and especially if it is combined with the principles of polymorphic electronics, undoubtedly offers notable advantages over the standard or conventional solution. However, the conventional-based existing approaches, and some of the previous methods from the polymorphic electronics domain alike, produce a relatively inefficient solution in terms of the overall circuit size.

Due to this reason, a novel method based on the formal basis has been formulated. Various tests have been performed in order to evaluate the properties or behaviour

of the proposed method. The obtained results indicate that it is possible to achieve nearly 20 % improvement especially in comparison with the standard Espresso tool. The proposed method is currently using only a basic version of Boolean divisors identification, still the method tends to yield more than satisfactory results. Nevertheless, further improvements are planned from the side of extending the divisors identification procedure also for multi-level circuit specification.

Acknowledgments

This work was generously supported by the IT4Innovations Centre of Excellence (no.CZ.1.05/1.1.00/02.0070) and by the national COST grant Unconventional Design Techniques for Intrinsic Reconfiguration of Digital Circuits: From Materials to Implementation (no. LD14055).

References

[1] BOBDA, C.: Introduction to Reconfigurable Computing: Architectures, Springer Science+Business Media B.V., Dordrecht, 2007, pp. 15–66.

[2] CARDOSO, J.—HÜBNER, M.: Reconfigurable Computing, From FPGAs to Hardware/Software Codesign, Springer-Verlag, New York, 2011, pp. 7–34.

[3] TREFZER, M. A.—TYRRELL, A. M.: Evolvable Hardware: From Practice to Application, Springer Science+Business Media B.V., Dordrecht, 2015, pp. 14–22.

[4] RŮŽIČKA, R.—ŠIMEK, V.: More Complex Polymorphic Circuits: A Way to Implementation of Smart Dependable Systems, ElectroScope **07** No. 05 (2013), 01–06.

[5] STOICA, A.—ZEBULUM, R. S.—KEYMEULEN, D.: Polymorphic Electronics, Proceedings of Evolvable Systems: From Biology to Hardware Conference, volume 2210 of LNCS, Springer 2001, 291–302.

[6] SEKANINA, L.—RŮŽIČKA, R.—VAŠÍČEK, Z.—PROKOP, R.—FUJCIK, L.: REPOMO32 — New Reconfigurable Polymorphic Integrated Circuit for Adaptive Hardware, Proceedings of the 2009 IEEE Symposium Series on Computational Intelligence — Workshop on Evolvable and Adaptive Hardware, 2009, pp. 39–46.

[7] TANACHUTIWAT, S.—LEE, J. U.—WANG, W.—SUNG, C. Y.: Reconfigurable Multi-Function Logic based on Graphene P-N Junctions, Proceedings of the 47th ACM/IEEE Design Automation Conference, 2010, pp. 883–888.

[8] WEBER, W. M.—HEINZIG, A.—TROMMER, J.—MARTIN, D.—GRUBE, M.—MIKOLAJICK, T.: Reconfigurable Nanowire Electronics - A Review, Journal on Solid-State Electronics **102** (2014), 12–24.

[9] TESAŘ, R.—ŠIMEK, V.—RŮŽIČKA, R.—CRHA, A.: Polymorphic Electronics Based on Ambipolar OFETs, EDS 2014 IMAPS CS International Conference Proceedings, pp. 106–111.

[10] MILLER, J.—THOMSON, P.: Cartesian Genetic Programming, Proceedings of the 3rd European Conference on Genetic Programming EuroGP 2000, vol. 1802, 2000, pp. 121–132.

[11] GAJDA, Z.—SEKANINA, L.: On Evolutionary Synthesis of Compact Polymorphic Combinational Circuits, Journal of Multiple-Valued Logic and Soft Computing **17** No. 062011, 607–331.

[12] BÄCK, T.: Evolutionary Algorithms in Theory and Practice, Oxford University Press, Oxford, Great Britain, 1996, pp. 91–105.

[13] GAJDA, Z.—SEKANINA, L.: Reducing the Number of Transistors in Digital Circuits Using Gate-Level Evolutionary Design, Proceedings of the 2007 Genetic and Evolutionary Computation Conference, 2007, pp. 245–252.

[14] SEKANINA, L.: Ubiquity Symposium: Evolutionary Computation and the Processes of Life: Evolutionary Computation in Physical World, Ubiquity No. 022013 (2013), 1–7.

[15] KOZA, J. R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, Massachusetts, USA, 1992, pp. 527–552.

[16] THOMSON, A.: Silicon Evolution, Genetic Programming 1996: Proceedings of the First Annual Conference (Complex Adaptive Systems), 1996, pp. 444–452.

[17] WAKERLY, J. F.: Digital Design: Principles and Practices, Prentice Hall, New Jersey, USA, 2000, pp. 183–236.

[18] DARRINGERET, J. A.: Logic Synthesis through Local Transformations, IBM Journal of Research and Development **25** No. 04 (1981), 272–280.

[19] HACHTEL, G. D.—SOMENZI, F.: Logic Synthesis and Verification Algorithms, Kluwer Academic Publishers, Boston, Massachusetts, USA, 1996, pp. 409–454.

[20] HASSOUN, S.—SASAO, T.: Logic Synthesis and Verification, Kluwer Academic Publishers, Boston, Massachusetts, USA, 2002, pp. 29–64.

[21] BRAYTON, R. K.—MCMULLEN, C.: The Decomposition and Factorization of Boolean Expressions, Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), 1982, pp. 49–54.

Received 16 November 2015

Václav Šimek obtained MSc degree (Ing) in 2006 in Electronics and Information Technologies at the Faculty of Information Technology, Brno University of Technology, Czech Republic. Since 2009 he has been with the same institution comprising various teaching and research tasks. His research interests are primarily diverse aspects of digital circuits design with special emphasis given to exploiation of emerging materials and beyond CMOS devices, his professional interests also include embedded systems, reconfigurable hardware design and data compression techniques.

Adam Crha has obtained MSc degree (Ing) in 2013 in Computer and Embedded Systems at the Faculty of Information Technology, Brno University of Technology, Czech Republic. His interests are focused mainly on the issues of digital circuit synthesis optimization by means of using unconventional techniques, multifunctional or polymorphic electronics and also research dealing with the adoption of emerging materials for construction of digital circuit elements.

Richard Růžička obtained his PhD in 2002 at the Faculty of Information Technology, Brno University of Technology. He is currently associated professor at Department of Computer Systems, Faculty of Information Technology, Brno University of Technology and vice-dean for MSc Education of the faculty. His research interests are digital circuits design and diagnostics, unconventional electronics and applications of post-silicon devices, dependable systems, embedded systems. He is a leader or a team member of several research projects of polymorphic electronics and multifunctional logic.