FILOMAT 17 (2003), 47–58

# DUAL POLARITY IN OPTIMIZATION OF POLYNOMIAL REPRESENTATIONS OF SWITCHING FUNCTIONS

DRAGAN JANKOVIC, RADOMIR S. STANKOVIC AND CLAUDIO MORAGA

ABSTRACT. **Compact representations of switching functions provide for simplified realizations of logic networks. Recent advent of digital technology and VLSI raised a considerable interest in polynomial expressions for switching functions. Complexity of these expressions is estimated through the number of product terms. For a given function and the selected class of polynomial expressions, the number of products can be reduced by a suitable selection of polarities for switching variables, which results in Fixed Polarity Polynomial Expressions (FPPEs).**

**This paper proposes a method for optimization of polynomial representation of switching functions. The method exploits the notion of dual polarity of switching variables and takes advantages of a simple relationship between two FPPEs for dual polarities. Calculation of FPPEs is performed along the extended dual polarity route so that each FPPE is calculated from its extended dual polarity FPPE. Conversion from one FPPE to another is carried out by using one-bit check, which ensures the efficiency of the method. The proposed method can be applied to arbitrary polynomial expressions providing that the corresponding transform matrix has the Kronecker product structure.**

## 1. INTRODUCTION

Polynomial representations of switching functions are an alternative way to represent this class of discrete functions that appears convenient in many applications. There are various polynomial representations for switching functions that differ mutually by the set of basis functions used and their range. Nowadays, the most widely used are Reed-Muller expressions and Kronecker expressions among bit-level representations, while the arithmetic

*Key words and phrases.* switching functions, polynomial representation, optimization, dual polarity.

representations are an important class of word-level polynomial expressions [1], [6], [8], [9]. The terms bit-level and word-level representations relate to the range allowed for the coefficients in polynomial expressions although expressions being applied to two-valued logic functions. Similar consideration applies to multiple-valued logic functions, in which case each bit can be multiple- valued [2], [7].

Optimization of polynomial representations is possible by using different polarities for variables. If variables appear in either complemented or non-complemented form, but not both in the same expression, polynomial expressions are known as Fixed polarity polynomial expressions (FPPEs). The number of non-zero coefficients is usually used as the optimization criterion in comparing different polynomial expressions for a given function in terms of time and space complexity. The term polarity is used to denote particular assignments of positive and negative literals to variables. For a given function $f$, the polarity resulting in the polynomial expressions with the minimum number of non-zero coefficients is considered as the optimal polarity, and this expression is the optimal fixed polarity expression for $f$.

There is a variety of methods for the determination of the optimal polarity. These methods can be classified as exact and heuristic methods. Exact methods mainly reduce to efficient implementation of an exhaustive search over all possible combinations of literals for variables, while heuristic methods reduce the search space by some heuristic rules. The main disadvantage of heuristic methods is that they do not provide any guarantee for the quality of the solutions produced. However, for the reduced calculation complexity, these methods are usually applicable for large functions.

The chief drawback of exact methods is large run-time which is often a restricting factor in practical applications. With this motivation, in this paper we propose an exact method with reduced implementation time. We consider exploitation of the dual polarity property in optimization of fixed polarity polynomial expressions. The main idea is that calculation of all possible fixed polarity expressions for a given $n$-variable function $f$ is performed along the dual polarity route where each two neighboring polarities are dual. The standard definition of dual polarity is adapted to various polynomial expressions (as, for example, Kronecker expressions) and the term extended dual polarity is used. We show the relation between dual polarity polynomial expressions and give a unified method for calculation of different fixed polarity expression starting from the dual polarity expressions. In this way, the calculation cost is significantly reduced compared to the methods where all fixed polarities are calculated starting from the truth vectors. We also propose procedures to generate the dual polarity route and extended dual polarity route.

The algorithm proposed is an exhaustive-search algorithm, but conversion from a given fixed polarity expression to another polarity expression is carried out by using one-bit check. Due to that, and for the simplicity of the related processing, the algorithm appears efficient in terms of both space and time as confirmed by experimental results over benchmark functions. It should be noticed that the algorithm proposed expresses high possibilities for parallelization, which is a feature important for practical applications.

## 2. Basic definitions

**Definition 2.1.** A $q$-valued function of $n$ $p$-valued variables is a mapping $f : \{0, 1, \dots, q-1\}^n \to \{0, 1, \dots, r-1\}$.

For $q$=2 and $r$=2, $f$ is a Boolean (binary-vlued) function, while for $q = r > 2$, $f$ is a multiple-valued (MV) function.

### 2.1. **Polynomial expressions.**

**Definition 2.2.** (Polynomial expressions (PE))
Each $n$-variable $q$-valued function $f$ given by the truth- vector

$$\mathbf{F} = [f_0, \cdots, f_{q^n-1}]^T$$

can be represented by the polynomial expression defined as

$$f(x_1, \dots, x_n) = \mathbf{X}(n)\mathbf{T}(n)\mathbf{F},$$

where

$$\mathbf{X}(n) = \bigotimes_{i=1}^{n} \begin{bmatrix} 1 & x_i & x_i^2 & \cdots & x_i^{q-1} \end{bmatrix},$$

and

$$\mathbf{T}(n) = \textstyle\bigotimes_{i=1}^{n} \mathbf{T}(1), \qquad \mathbf{T}(1) = (\mathbf{X}(1))^{-1},$$

where $\bigotimes$ denotes the Kronecker product, and the basic transform matrix $\mathbf{T}(1)$ is defined as the inverse of $\mathbf{X}(1)$, assuming that symbolic notation for columns of $\mathbf{X}(1)$ is replaced by the corresponding truth-vectors. Addition and multiplication (and, hence, exponentiation) are defined in the algebraic structure assumed for this consideration depending on the intended applications.

### 2.2. **Optimization of polynomial expressions.**

Optimization of polynomial expressions, which means determination of an expressions with the minimum number of non-zero coefficients (i.e., the number of terms), can be done by introducing different polarity for variables. The representations produced in this way are the Fixed polarity polynomial expressions (FPPE) where each variable appears as either the positive or the negative literal. i.e., as uncomplemented or complemented, but not both at the same time.

A polarity vector $P$ is introduced to denote the polarity of variables, and correspondingly, the polarity of the representation.

**Definition 2.3.** Polarity vector $P = (p_1, \dots, p_n)$, $p_i \in \{0, 1, \dots, q-1\}$ is a vector of the length $n$, where $n$ is the number of variables, whose elements specify polarity of variables in FPPE for an $n$-variable function $f$, i.e., $p_i = j$ shows that the $j$-th complement is assigned to the $i$-th variable which is denoted as ${}^{j}\overline{x}_i$.

FPPEs are uniquely characterized by specifying decomposition rules assigned to variables and the polarity vectors [8].

**Definition 2.4.** For the polarity is $p = (p_1, \dots, p_n)$, each $n$-variable $q$-valued function $f$ given by the truth-vector $\mathbf{F} = [f_0, \cdots, f_{q^n-1}]^T$, can be represented by the following FPPE

$$f(x_1, \dots, x_n) = \mathbf{X}^{<p>}(n)\mathbf{G}^{<p>}(n)\mathbf{F}$$

where

$$\mathbf{X}^{<p>}(n) = \bigotimes_{i=1}^{n} \left[ \begin{array}{cccc} 1 & {}^{p_i}\overline{x}_i & ({}^{p_i}\overline{x}_i)^2 & \dots & ({}^{p_i}\overline{x}_i)^{q-1} \end{array} \right],$$

$$\mathbf{G}^{<p>}(n) = \bigotimes_{i=1}^{n} \mathbf{G}^{<p_i>}(1).$$

Therefore, for a given polarity $P = (p_1, \dots, p_n)$, $p_i \in \{0, 1, \dots, q-1\}$, FPPEs are uniquely specified by the vector of coefficients, usually denoted as spectrum, $\mathbf{S}_f^{<p>}$ defined as

$$\mathbf{S}_f^{<p>} = \mathbf{G}^{<p>}(n)\mathbf{F}.$$

## 3. Extended dual polarity

In minimization of FPPE in the number of non-zero coefficients count, it appears convenient to exploit the notion of the dual polarity [5], the extended dual polarity, and the related dual polarity vectors.

**Definition 3.1.** (Extended dual polarity)
For a given polarity $P = (p_1, \dots, p_{i-1}, p_i, p_{i+1}, \dots, p_n)$, the extended dual polarity is $P' = (p'_1, \dots, p'_{i-1}, p'_i, p'_{i+1}, \dots, p'_n)$, iff $p'_j = p_j$, $j \neq i$ and $p'_j \neq p_i$.

*Example* 3.2. Extended dual polarities for the polarity $P = (1, 0)$ are the polarities $(0, 0)$, $(2, 0)$, $(3, 0)$, $(1, 1)$, $(1, 2)$, and $(1, 3)$.

3.1. **Dual polarity route.** The number of polarity vectors characterizing all possible FPPEs for an $n$-variable $q$-valued function is $q^n$. It is possible to order these polarities such that each two successive polarities are the extended dual polarities. We denote this order as the extended dual polarity route. Traversing a $q$-valued $n$-dimensional hypercube can generate one of many possible extended dual polarity routes.

*Example* 3.3. An extended dual polarity route generated by traversing a 4-valued $n$-dimensional hypercube is given by

(000)—(001)—(002)—(003)—(013)—(012)—(011)—(010)—(020)—(021)—(022)—
(023)—(033)—(032)—(031)—(030)—(130)—(131)—(132)—(133)—(123)—(122)—
(121)—(120)—(110)—(111)—(112)—(113)—(103)—(102)—(101)—(100)—(200)—
(201)—(202)—(203)—(213)—(212)—(211)—(210)—(220)—(221)—(222)—(223)—
(233)—(232)—(231)—(230)—(330)—(331)—(332)—(333)—(323)—(322)—(321)—
(320)—(310)—(311)—(312)—(313)—(303)—(302)—(301)—(300).

An extended dual polarity route can be constructed by using the recursive procedure ***route***(*level*, *direction*) given in Fig. 3.1 for the particular case $q = 4$. Extension to an arbitrary $q$ is straightforward. The extended dual polarity route will be produced if the procedure is called with arguments equal 0, i.e., *route*(0,0).

## 4. The relationship between extended dual polarity FPPEs

In this section, the relationship between two extended dual polarity expressions will be explained on the example of fixed polarity polynomial expressions defined on GF(4), short, FPGFs [3], [4].

The relationship between two FPGFs can be expressed as

$$\mathbf{S}^{<p'>} = \mathbf{G}^{<p'>} \cdot \mathbf{F} = \mathbf{G}^{<p'>} \cdot \left(\mathbf{G}^{<p>}\right)^{-1} \cdot \mathbf{S}^{<p>}.$$

If polarities $p' = (p_1 \cdots p_{i-1} p_i' p_{i+1} \cdots p_n)$ and $p = (p_1 \cdots p_{i-1} p_i p_{i+1} \cdots p_n)$ are the extended dual polarities, and $p_i' \neq p_i$, the relationship between the corresponding FPGFs is given by

$$\begin{aligned}
&\mathbf{S}^{<p'>} = \mathbf{G}^{<p'>} \cdot \mathbf{F} = \\
&\left(\left(\bigotimes_{j=1}^{i-1} \mathbf{G}^{<p_j>}\right) \otimes \mathbf{G}^{<p_i'>} \otimes \left(\bigotimes_{j=i+1}^{n} \mathbf{G}^{<p_j>}\right)\right) \cdot \mathbf{F} = \\
&\left(\left(\bigotimes_{j=1}^{i-1} \left(\mathbf{G}^{<p_j>}\right)^{-1}\right) \otimes \left(\mathbf{G}^{<p_i>}\right)^{-1} \otimes \left(\bigotimes_{j=i+1}^{n} \left(\mathbf{G}^{<p_j>}\right)^{-1}\right)\right) \cdot \mathbf{S}^{<p>}
\end{aligned}$$

Due to the features of Kronecker product (assuming consistent dimensions of related matrices), the above relation can be written as

$$\mathbf{S}^{<p'>} = \left(\mathbf{I}_i \otimes \mathbf{G}^{<p'>} \cdot \left(\mathbf{G}^{<p>}\right)^{-1} \otimes \mathbf{I}_{n-i}\right) \cdot \mathbf{S}^{<p>} \tag{4.1}$$

where $\mathbf{I}_k$ is the identity matrix of order $k$.

```
void route(int level, int direction){
    if( direction == 0){
        if(level == no_variable){
            – out new polarity vector h }
        else {
            h[level] = 0;
            route(level+1, 0);
            h[level] = 1;
            route(level+1, 1);
            h[level] = 2;
            route(level+1, 0);
            h[level] = 3;
            route(level+1, 1);}}
    else{
        if(level == no_variable){
            – out new polarity vector h }
        else {
            h[level] = 3;
            route(level+1, 0);
            h[level] = 2;
            route(level+1, 1);
            h[level] = 1;
            route(level+1, 0);
            h[level] = 0;
            route(level+1, 1); } } }
```

FIGURE 1. Procedure ***route***().

4.1. **The processing rule.** ¿From the relation (4.1), i.e. from all possible matrix products $\mathbf{G}^{<p'>} \cdot (\mathbf{G}^{<p>})^{-1}$ given in Table 1, it is possible to derive a rule for processing terms in $\mathbf{S}^{<p>}$ to calculate $\mathbf{S}^{<p'>}$.

Let $m = m_1^{i-1} m_i m_{i+1}^n$ be a term in a FPGF expression of a function $f$ for the polarity $p = (p_1 \cdots p_{i-1} p_i p_{i+1} \cdots p_n)$. The value of $f$ for the term $m$ is $\mathbf{S}^{<p>}(m)$. The term $m$ produces new terms in the FPGF expression $\mathbf{S}^{<p'>}$ for the function $f$ for the dual polarity $p' = (p'_1 \cdots p'_{i-1} p'_i p'_{i+1} \cdots p'_n)$ depending on the value $p_i$ and $p'_i$.

The value of the FPGF on a newly generated term is given by $v\mathbf{S}^{<p>}(m)$. Table 2 shows processing rules for all possible cases. It is obvious that these processing rules are simple and due to that, efficient in terms of space and time. After processing all terms by using the given rules, it starts a procedure for deleting terms whose value is equal to zero after summation of contributions of the processed terms.

*Example* 4.1. For the polarity (12), the fixed polarity GF(4) expression of a two-variable four-valued function $f$ given by the truth vector $\mathbf{F} = [0,1,2,2,1,3,0,0,0,1,1,0,0,1,2,3]^T$ is represented by the spectrum

$$\mathbf{S}^{<1,2>} = [0,2,0,2,1,0,0,1,2,3,3,1,1,1,2,3]^T .$$

The extended dual polarities and the corresponding FPGFs are given in the Table 3.

Calculation procedure for determination of (02) dual polarity FPGF expression from (12) polarity FPGF is shown in Table 4.

TABLE 1. Relationship between dual polarities

| |
|---|
| $\mathbf{G}^{<0>} \cdot (\mathbf{G}^{<1>})^{-1} = \mathbf{G}^{<1>} \cdot (\mathbf{G}^{<0>})^{-1} = \mathbf{G}^{<2>} \cdot (\mathbf{G}^{<3>})^{-1}$ $= \mathbf{G}^{<3>} \cdot (\mathbf{G}^{<2>})^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |
| $\mathbf{G}^{<0>} \cdot (\mathbf{G}^{<2>})^{-1} = \mathbf{G}^{<2>} \cdot (\mathbf{G}^{<0>})^{-1} = \mathbf{G}^{<1>} \cdot (\mathbf{G}^{<3>})^{-1}$ $= \mathbf{G}^{<3>} \cdot (\mathbf{G}^{<1>})^{-1} = \begin{bmatrix} 1 & 2 & 3 & 1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |
| $\mathbf{G}^{<0>} \cdot (\mathbf{G}^{<3>})^{-1} = \mathbf{G}^{<3>} \cdot (\mathbf{G}^{<0>})^{-1} = \mathbf{G}^{<1>} \cdot (\mathbf{G}^{<2>})^{-1}$ $= \mathbf{G}^{<2>} \cdot (\mathbf{G}^{<1>})^{-1} = \begin{bmatrix} 1 & 3 & 2 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |

## 5. Optimization algorithm

The example 4.1 shows that it is possible to calculate all possible FPGF expressions by using the proposed method for transforming fixed polarity GF(4) expressions into extended dual polarity GF(4) expressions along the route without repetitive calculations. Therefore, we can perform the optimization of FPGF expressions by using an exhaustive-search algorithm consisting of the following steps.

(1) Initialization:
    Set the polarity vector $p'$ to $p' = (0,0,\cdots,0)$
    Calculate the spectrum $\mathbf{S}^{<p'>}$
    $C_{\min}$ =the number of non-zero coefficients in $\mathbf{S}^{<p'>}$
(2) List all the terms for $\mathbf{S}^{<p'>}$.

(3) Determine the next extended dual polarity $p$', according to the recursive route.
(4) Calculate the spectrum $\mathbf{S}^{<p'>}$by using the proposed rule. Calculate the total number of non-zero coefficients $C'_{\min}$.
If $C'_{\min} < C_{\min}$ then $C_{\min} = C'_{\min}$.
(5) Stop if all polarities have been treated. Otherwise go to the step 2.

The algorithm starts from the zero polarity FPGF, but it can start from any other polarity. Therefore, FPGF for the zero polarity for a given function $f$ should be calculated from the truth-vector. For this task, we can use some of known methods, for example, the tabular technique [3], [10].

## 6. Experimental results

In this section, we present some experimental results estimating features and efficiency of the proposed algorithm for minimization of FPGF expressions. We developed a program in C for determination of optimal FPGF expression for arbitrary four-valued functions represented by minterms. The experiments were carried out on 1GHz PC Celeron with 128Mb of main memory and all runtimes are given in CPU seconds. Table 5 compares the runtimes for optimization of FPGF expressions by the CTT method in [3] (columns TT) with the algorithm proposed in this paper (columns Dual). We consider randomly generated four-valued functions with 25% and 75% of non-zero minterms (columns 25% and 75%, respectively) where the number of four-valued variables $n$ ranges from 4 to 7. Columns %d show the ratio $(CTT - Dual)/\ Dual$ where $CTT$ and $Dual$ refer to the method in [3] and the proposed algorithm, respectively.

The proposed algorithm is faster than CTT for the simplify processing by using the dual polarity property. This ratio increases with a number of variables. Furthermore, the number of non-zero minterms has smaller influence to the runtime of the proposed algorithm as compared to CTT.

## 7. Particular examples

In this section, we discuss processing rules for some particular cases of FPPEs.

he Table 6 shows the processing rule for optimization of Kronecker expression, while Table 7 shows the processing rule for optimization of arithmetic expressions. After processing all terms, by using above rules, a procedure for deleting terms whose contribution is equal to 0 starts.

## 8. Conclusions

We have introduced the notion of dual polarities for Fixed polarity Galois field (FPGF) expressions for functions defined on GF(4) and present a

TABLE 2. GF(4) processing rule.

| $p_i$ | $p_i'$ | new terms |
|---|---|---|
| 0 | 1 | if $m_i$=1 then generate $m_1^{i-1}0m_{i+1}^n$ $v=1$ |
| 1 | 0 | if $m_i$=2 then generate $m_1^{i-1}0m_{i+1}^n$ $v=1$ |
| 2 | 3 | if $m_i$=3 then generate $m_1^{i-1}0m_{i+1}^n$ $v=1$ |
| 3 | 2 | $m_1^{i-1}1m_{i+1}^n$ $v=1$ |
| | | $m_1^{i-1}2m_{i+1}^n$ $v=1$ |
| 0 | 2 | if $m_i$=1 then generate $m_1^{i-1}0m_{i+1}^n$ $v=2$ |
| 1 | 3 | if $m_i$=2 then generate $m_1^{i-1}0m_{i+1}^n$ $v=3$ |
| 2 | 0 | if $m_i$=3 then generate $m_1^{i-1}0m_{i+1}^n$ $v=1$ |
| 3 | 1 | $m_1^{i-1}1m_{i+1}^n$ $v=3$ |
| | | $m_1^{i-1}2m_{i+1}^n$ $v=2$ |
| 0 | 3 | if $m_i$=1 then generate $m_1^{i-1}0m_{i+1}^n$ $v=3$ |
| 1 | 2 | if $m_i$=2 then generate $m_1^{i-1}0m_{i+1}^n$ $v=2$ |
| 2 | 1 | if $m_i$=3 then generate $m_1^{i-1}0m_{i+1}^n$ $v=1$ |
| 3 | 0 | $m_1^{i-1}1m_{i+1}^n$ $v=2$ |
| | | $m_1^{i-1}2m_{i+1}^n$ $v=3$ |

TABLE 3. Extended dual polarity GF(4) expressions.

| polarity | spectrum |
|---|---|
| 02 | (2011012232121123) |
| 22 | (1230323010231123) |
| 32 | (2100231301001123) |
| 10 | (1332032100111333) |
| 11 | (3112023101012003) |
| 13 | (0022011132211213) |

method for conversion of FPGF expressions from one polarity to the dual polarity. Based on this method, we define an algorithm for calculation of all FPGF expressions. Calculation is performed by starting from the truth-vectors of functions considered. All FPGF expressions are calculated along a route that provides calculation of each FPGF expressions exactly once.

The proposed method for transformation of FPGF expression from one to another extended dual polarity is simple. Due to that, although being an exhaustive-search method, the proposed method for minimization of FPGF expressions is efficient as documented by experimental results.

The proposed method can be applied to the optimization of all FPPEs with Kronecker product representable transform matrices. The proposed

TABLE 4. Polarity (12) to (02)

| polarity (12) | New terms | polarity (02) |
|---|---|---|
| 01 – 2 | | 00 – 2 |
| 03 – 2 | | 02 – 1 |
| 10 – 1 | 00 – 1 | 03 – 1 |
| 13 – 1 | 03 – 1 | 11 – 1 |
| 20 – 2 | 00 – 2 | 12 – 2 |
| 21 – 3 | 01 – 3 | 13 – 2 |
| 22 – 3 | 02 – 3 | 20 – 3 |
| 23 – 1 | 03 – 1 | 21 – 2 |
| 30 – 1 | 00 – 1; 10 – 1; 20 – 1 | 22 – 1 |
| 31 – 1 | 01 – 1; 11 – 1; 21 – 1 | 23 – 2 |
| 32 – 2 | 02 – 2; 12 – 2; 22 – 2 | 30 – 1 |
| 33 – 3 | 03 – 3; 13 – 3; 23 – 3 | 31 – 1 |
| | | 32 – 2 |
| | | 33 – 3 |

TABLE 5. Experimental results.

| $n$ | 25% nonzero | | | 75% nonzero | | |
|---|---|---|---|---|---|---|
| | Dual | TT | %d | Dual | TT | %d |
| 4 | 0.04 | 0.06 | -33.33 | 0.4 | 0.16 | -75.0 |
| 5 | 0.54 | 2.17 | -75.11 | 0.56 | 6.33 | -91.15 |
| 6 | 10.34 | 89.75 | -88.48 | 12.69 | 266.54 | -95.24 |
| 7 | 251.32 | 3922.05 | -93.59 | 299.28 | 12429.99 | -97.59 |

TABLE 6. Processing rule for optimization of Kronecker expressions.

| $p_i$ | $p_i'$ | new terms |
|---|---|---|
| 0 | 1 | if $m_i$=1 then generate $m_1^{i-1}0m_{i+1}^n$ |
| 1 | 0 | |
| 0 | 2 | if $m_i$=0 then generate $m_1^{i-1}1m_{i+1}^n$ |
| 2 | 0 | |
| 1 | 2 | if $m_i$=1 then set $m_i$=0<br>if $m_i$=0 then generate $m_1^{i-1}1m_{i+1}^n$ |
| 2 | 1 | if $m_i$=0 then set $m_i$=1<br>if $m_i$=1 then generate $m_1^{i-1}0m_{i+1}^n$ |

TABLE 7. Processing rule for optimization of arithmetic expressions.

| $h_i$ | $h'_i$ | new terms |
|---|---|---|
| 0 | 1 | if $m_i$=1 then a) generate $m' = m_1^{i-1}0m_{i+1}^n$ and $v(m') = v(m)$ |
| 1 | 0 | b) set $v(m)$ to $-v(m)$ |

methods offers high possibilities for parallelization which can be considered as an advantage in practical applications.

Future work in this subject will be devoted to the extension of the method proposed and the related implementation algorithm for minimization of polynomial representations for multiple-valued functions.

## REFERENCES


[1] Falkowski, B.J., "A note on the polynomial form of Boolean functions and related topics", *IEEE Trans. on Computers*, Vol. 48, No. 8, 1999, 860-864.

[2] Falkowski, B.J., Rahardja, S., "Efficient computation of quateranry fixed polarity Reed-Muller expansions," *IEE Proc. Comput. Digit. Tech.*, Vol. 142, No. 5, 1995, 345-352.

[3] Janković D., Stanković, R.S., , Drechsler R., "Efficient calculation of fixed polarity polynomial expressions for multi-valued logic functions", *Proc. 32nd Int. Symp. on Multiple-Valued Logic, Boston, USA*, (May 2002), 76-82.

[4] Janković, D., Stanković, R.S., Moraga, C.,"Reed-Muller-Fourier versus Galois field representations of four-valued logic functions", *Proc. 28th Int. Symp. on Multiple-Valued Logic, Fukuoka, Japan* (May 29-31, 1998).

[5] D. Janković, R.S. Stanković, C. Moraga, "Optimization of Kronecker expressions using the extended dual polarity property", *Proc. ICEST* , Niš, Serbia, 2002, 749-752.

[6] Malyugin, V.D., *Paralleled Calculations by Means of Arithmetic Polynomials*, Physical and Mathematical Publishing Company, Russian Academy of Sciences, Moscow, 1997, (in Russian).

[7] Rahardja, S., Falkowski, B.J., "Efficient algorithm to calculate Reed-Muller expansions over $GF(4)$," *IEE Proc. Circuits, Devices and Systems*, Vol. 148, No. 6, 2001, 289-295.

[8] Sasao, T., Fujita, M., (ed.), *Representations of Discrete Functions*, Kluwer Academic Publishers, 1996.

[9] Stanković, R.S., "Word-level ternary decision diagrams and arithmetic expressions", *Proc. 5th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, Starkville, Mississippi, USA, August 10-11, 2001, 34-50.

[10] Stanković, R.S., Falkowski, B.J., "Spectral interpretation of fast tabualr technique for fixed polarity Reed-Muller expressions", *Int. J. Electronics*, Vol. 87, No. 6, 2000, 641-648.



(D. Jankovic) FACULTY OF ELECTRONIC ENGINEERING, UNIVERSITY OF NIS, 18000 NIS, SERBIA AND MONTENEGRO

(R. S. Stankovic) FACULTY OF ELECTRONIC ENGINEERING, UNIVERSITY OF NIS, 18000 NIS, SERBIA AND MONTENEGRO

(C. Moraga) Dept. of Computer Science, University of Dortmund, Germany and Dept. of Artificial Intelligence, Technical University of Madrid, Spain
*E-mail address*, Dragan Jankovic: gaga@elfak.ni.ac.yu
*E-mail address*, Radomir S. Stankovic: rstankovic@elfak.ni.ac.yu
*E-mail address*, Claudio Moraga: claudio@moraga.de