REDUCTION OF THE NUMBER OF COEFFICIENTS IN ARITHMETIC EXPRESSIONS BY AUTOCORRELATION FUNCTIONS

Radomir S. Stanković
Dept. of Computer Science
Faculty of Electronics
Beogradska 14
18 000 Niš
Serbia

Mark G. Karpovsky
Dept. of Electrical and Computer Engineering
8 Saint Marry's Street
Boston University
Boston, Ma 02215
USA

Jaakko T. Astola
Tampere Int. Center for Signal Processing
Tampere University of Technology
Tampere
Finland

Abstract

This paper proposes to combine classical approaches to the optimization of arithmetic expressions by choosing polarity of variables with the linearization of switching functions (LSF).

Experimental results presented in the paper, show that in classical approaches, transition from Positive-polarity arithmetic expressions to Fixed-polarity arithmetic expressions may provide for 30-40% of reduced number of coefficients. However, when combined with LSF another 20% of reduction in the non-zero coefficients count can be achieved.

1 INTRODUCTION

Arithmetic expressions (ARs) for switching functions are an alternative way to represent binary valued logic functions efficiently in terms of space and time. These expressions were used in the description of logic networks from the beginning of development of this area [2], [14], [27], see also discussions in [15], and [35] and related references in [36]. Although ARs for switching functions are hybrid expressions in the sense that the coefficients are integers for the logic-valued functions, there is apparent a renewed interest in application of ARs, partially due just to this feature, for the following reasons.

1. ARs are useful in parallelization of algorithms for calculations with switching functions [15],

2. ARs can represent the multi-output switching functions by a single expression which cannot be done with bit-level expressions [16], [21].

3. ARs belong to the class of word-level expressions for switching functions in form of which some word-level decision diagrams represent the switching functions [31], [37]. For detailed considerations of this subject, see [33].

4. ARs can be used to estimate the error probability in logic networks [15], and can be useful in testing, equivalence checking, and verification [13], [20], [29].

Applications of ARs in logic design and related areas of signal processing and digital systems design are discussed in [1], [15], and several related papers in [16], [17]. The most recent applications are briefly reviewed in [6]. Various extensions to multiple-valued logic functions are discussed in [16], [17], [38], and [39]. For recent applications, we refer to [29], and [13] and references given there.

In practical applications of ARs, given a function f, it might be of an immense importance to determine an arithmetic expression for f with the minimum number of non-zero coefficients, since this determines the complexity of the overall application procedures in terms of both space and time. Therefore, similar as in applications of Reed-Muller expressions, a lot of efforts should be paid to the minimization of ARs. This paper considers the minimization of ARs and points out that combination of FPARs and the method for linearization of switching functions (LSF) by autocorrelation functions [10] can be useful in determination of reduced ARs in terms of the non-zero coefficients count.

2 ARITHMETIC EXPRESSIONS

Arithmetic expressions can be defined in different ways. For example, they can be derived from Boolean expressions by replacing the Boolean operations with the corresponding arithmetic operations assuming that switching variables and binary logic values for functions are interpreted as integers 0 and 1 instead of logic values 0 and 1 [14], [38]. Thus, calculations over $GF(2)$ are replaced with calculation over the field of rational numbers Q. Alternatively, ARs can be considered as integer counterpart of the Reed-Muller expressions for switching functions, see for example, [33]. Therefore, ARs can be derived by the recursive application of the integer counterpart of the positive Davio expansion $f = 1 \cdot f_0 + x_i(-f_0 + f_1)$ to all the variables in a given function f, where f_0 and f_1 are co-factors of f with respect to the variable x_i. This interpretation of ARs is useful, since many methods that have been already developed for Reed-Muller AND-EXOR synthesis and related minimization algorithms [25] can be transferred to ARs directly or under some appropriate modifications. For example, the exact minimization of Reed-Muller expressions through EXOR ternary decision diagrams (EXOR-TDDs) [5], is extended to ARs assuming introducing the corresponding data structure as Arithmetic spectral transform ternary decision diagrams (AC-TDDs) [32].

In a more general setting, AR-expressions can be regarded as the Taylor expressions in the space of complex functions on finite dyadic groups [9], with switching functions considered as a particular subset of two-valued complex functions on these groups. In this interpretation, coefficients in AR-expressions are the partial Gibbs derivatives on finite dyadic groups [8], the complex-valued counterparts of the Boolean differences [9].

Fixed-polarity Reed-Muller expressions (FPRMs) were introduced in order to reduce the number of coefficients in these representations by changing polarity of literals for variables, see for example [21]. The analogy of ARs with Reed-Muller expressions and an alternative interpretation of AR-coefficients as Fourier-series like coefficients, see for example [3], [28], [30], permits introduction of Fixed-polarity AR (FPAR) expressions, as the integer counterparts of Fixed-polarity Reed-Muller (FPRM) expressions, see for example [15], [24], [39]. In this spectral interpretation, FPAR-expressions are defined through permutation of basic functions used in definition of AR expressions. Note that the discrete Walsh series (Fourier series on finite dyadic groups) become the AR-expressions if the Walsh functions are written in terms of switching variables [30].

Definition 1 *(Arithmetic expressions)*
For n-variable switching functions given by truth-vectors $\mathbf{F} = [f(0), \ldots, f(2^n - 1)]^T$, the arithmetic expression is defined as $f = \mathbf{X}\mathbf{A}(n)\mathbf{F}$, where $\mathbf{X} = \bigotimes_{i=1}^{n} \mathbf{X}_i$, with $\mathbf{X}_i = \begin{bmatrix} 1 & x_i \end{bmatrix}$, and $\mathbf{A}(n) = \bigotimes_{i=1}^{n} \mathbf{A}(1)$, with $\mathbf{A}(1) = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$.

Definition 2 *(Arithmetic spectrum)*
The coefficients in the arithmetic expressions are usually denoted as the arithmetic spectrum represented in matrix notation as a vector $\mathbf{S}_f = [S_f(0), \ldots, S_f(2^n - 1)]^T$ defined as $\mathbf{S}_f = \mathbf{A}(n)\mathbf{F}$.

3 MINIMIZATION OF AR

Similar as in Reed-Muller and other spectral representations, applications of ARs are based upon the same simple principle. Manipulations and calculations with function values are replaced by dealing with spectral coefficients, hoping that this will possibly

1. Reduce complexity of the problem/solutions by reducing its dimensionality (number of coefficients),
2. Permit exploiting some properties and relationships that are not obvious or do not exist in the original domain,
3. Provide for faster methods due to fast calculation algorithms for spectral coefficients.

When, searching for some solutions or solutions better than existing with respect to some appropriate criteria, we decide to use arithmetic expressions, then a usual way of reasoning would be as follows.

1. Given a function f, determine AR for f.
2. If the number of coefficients is acceptable for the available space and time resources, start the applications using this AR, otherwise try to minimize AR for f.

Minimization of ARs can be performed by selecting literals of different polarity for switching variables in the same way as that is done in Reed-Muller expressions [22], [26]. In this way, Fixed-polarity ARs (FPARs) are defined, assuming restriction that a variable can appear as either the positive or the negative literal, but not both in the same expression for a given switching function f. FPARs with all the variables appearing as positive literals are zero- or positive-polarity arithmetic expressions (PPARs), short ARs. There are 2^n possible FPARs for an n-variable function f. The negative literal $\overline{x}_i$ for a variable x_i corresponds to the permutation of columns in the basic transform matrix $\mathbf{A}(1)$ at the i-th position in the Kronecker product generating $\mathbf{A}(n)$, the same as in FPARs [30]. Alternatively, this permutation in the transform matrix can be expressed as a permutation of function values in the truth-vector for f. This interpretation of the influence of negative literals to the coefficients in ARs will be exploited further in this paper.

In spectral interpretation of ARs, FPARs correspond to Fourier series-like expressions defined by permutation and/or shift of the same set of basis functions described, as in the Reed-Muller expressions, by primary products of switching variables, however, over Q instead $GF(2)$ [33].

Whatever is the interpretation of the basic minimization principle, the minimization by FAPRs is restricted to 2^n possible combinations. An alternative to overcome this restriction is to enlarge the set of decomposition rules by using the integer equivalent of the Shannon expansion $f = \overline{x}_i f_0 + x_i f_1$, which leads to the Kronecker ARs permitting 3^n combinations. However, the disadvantages are lack of criteria how to select and assign decomposition rules to variables in a given function f and increased complexity of manipulation with such representations due to the overhead of storing these assignments and always taking them into account. This problem is even stronger in the case of further generalizations as Pseudo-Kronecker ARs [33] derived by allowing further freedom in assigning decomposition rules in the same way as in the Pseudo-Kronecker Reed-Muller expressions [26].

Due to the above mentioned lack of assignment criteria, the exact minimization algorithms perform the exhaustive search of all possible expressions and count the number of coefficients in each of them to select the minimum expression for the given function. Although efficiency of these algorithms can be assured by selecting suitable data structures for functions and spectral coefficients, complexity of them can be considerable.

Heuristic algorithms reduce the search space by using some reasonable restrictions and preassumptions at the price of providing non-minimum results, that, however, may often be acceptable in practice.

4 LINEARIZATION OF SWITCHING FUNCTIONS

Linearization of switching functions (LSF) assumes representing a given system of Boolean functions as the superposition of a system of linear Boolean functions and a residual nonlinear part of minimal complexity. Liearization assumes expressing a given function f in variables x_i, $i = 1, \ldots, n$, in terms of another set of variables y_i, $i = 1, \ldots n$, determined as linear combinations of primary variables [11], [19]. The linearization problem consists in determination of a transformation σ that maps primary variables x_i into assigned variables y_i. In this way, a function $f_\sigma(y_1, \ldots, y_n)$ is assigned to a given function $f(x_1, \ldots, x_n)$. A solution of this problem is given in terms of the total autocorrelation functions [10]. Definition of the autocorrelation functions and the linearization procedure by using these functions [10] are given in the Addendum.

We note that the complexity of computing a total autocorrelation function $B_f(\tau)$ by the Wiener-Khinchin theorem [10] and by the fast Walsh transform [1], [10], expressed in the number of arithmetic operations does not exceed $O(n2^{n+k})$ and this approach is efficient only for small k. The straightforward application of the definition of B_f requires at most $O(2^{2n})$ computations for any k. It should be noted that the Walsh transform, can be performed over Multi-terminal binary decision diagrams (MTBDDs) [4], [7], which reduces the limitations to the number of variables in calculations related to the implementation of Wiener-Khinchin theorem.

In LSF, linear combination of primary variables corresponds to some permutation of function values in the vector of function values $\mathbf{F}$. The linearization method exploiting the total autocorrelation function performs a permutation that produces the maximum number of pairs of equal values. Due to this feature, LSF permits a considerable reduction of the number of non-zero coefficients in the Haar expressions [10], since the half of the Haar transform coefficients is calculated by subtracting successive function values. In matrix notation calculation of these coefficients equals to the appearance of submatrices $[1, -1]$. In other coefficients, the reduction to zero is possible if some combination of pairs of equal values appear, since the corresponding Haar functions are representable by sequences consisting of -1, 0, and 1 values. The arithmetic transform matrix $\mathbf{A}(n)$ used in calculation of coefficients of arithmetic expressions is defined as the Kronecker product of basic arithmetic transform matrices $\mathbf{A}(1) = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$. Due to the appearance of the row consisting of -1, and 1, in many cases, permutation of function values by the autocorrelation function may result in reduction of the arithmetic spectrum in the number of non-zero coefficients count. Experimental results show that this conjecture appears to be true.

5 LSF AND FPAR

Interpretation of the application of negative literals for variables in ARs as a permutation of function values put links between FPARs and LSF. In both cases, certain permutation of function values is performed. In FPARs, the permutations allowed are restricted to these corresponding to the permutation of logic values for variables to which negative literals are assigned. In LSF, a larger class of permutation matrices can be applied to $\mathbf{F}$, consisting of matrices induced by various possible linear combinations of variables. Therefore, due to this unified interpretation of both methods, and since different classes of permutation matrices are exploited, it may be interesting to combine these two approaches, FPARs and LSF, to get improved reduction of arithmetic spectra of switching functions in the number of non-zero coefficients count. Fig. 1 expresses the basic principles of minimization of ARs by combination of FPARs and LSF. In this figure, $\mathbf{P}_v$ and $\mathbf{P}_{LSF}$ denote permutation matrices determined by selecting negative literals for variables and by linear combination of variables, respectively. In practice, searching for a reduced arithmetic expression for a given function f, we chose the minimum among PPARs and FPARs with and without LSF. Features of these approaches will be illustrated by the following example.

It should be noted that the interpretation of FPARs and LSF in terms of permutation matrices is used for the theoretical considerations purposes. In practical implementations, we do not actually perform matrix calculations, instead we work with reduced representations for switching functions, as cubes and decision diagrams.

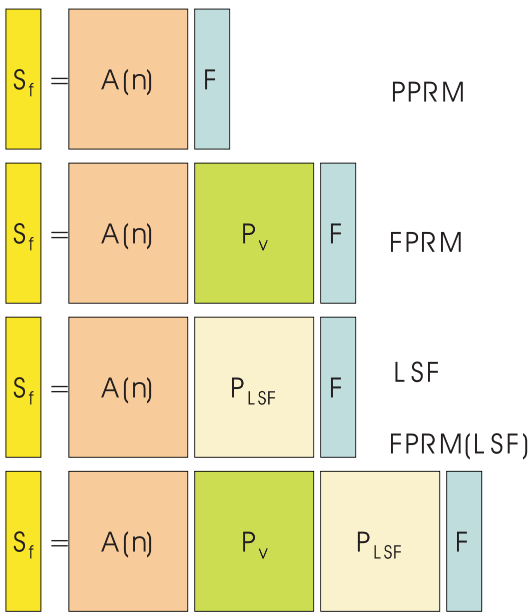

Figure 1. Calculation of ARs spectra.

Example 1 *Consider the following randomly chosen functions of four variables given by the vectors of function values.*

$$
\begin{aligned}
\mathbf{F} &= [0,2,2,3,2,0,3,2,2,3,0,2,3,2,2,0]^T \\
\mathbf{F}_1 &= [0,2,3,3,1,2,1,3,3,1,1,2,0,0,2,0]^T \\
\mathbf{F}_2 &= [0,0,1,0,1,0,0,0,1,0,0,1,0,1,0,0]^T \\
\mathbf{F}_3 &= [0,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0]^T \\
\mathbf{F}_4 &= [0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0]^T \\
\mathbf{F}_5 &= [0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1]^T
\end{aligned}
$$

It should be noted that the vector $\mathbf{F}$ represents an integer function f corresponding to a two output switching function $f = (f^0, f^1)$ and is determined as $f = 2f^1 + f^0$. Therefore, binary representations for elements in $\mathbf{F}$ represents separate outputs. The same is for $\mathbf{F}_1$.

LSF procedure assigns the following functions to the functions considered. For details about performing LSF procedure and related calculations we refer to [10] and [12].

$$
\begin{aligned}
\mathbf{F}_\sigma &= [0,0,0,0,2,2,2,2,2,2,2,3,3,3,3]^T \\
\mathbf{F}_{1,\sigma} &= [0,0,0,3,1,1,3,2,2,1,3,3,0,3,2,2]^T \\
\mathbf{F}_{2,\sigma} &= [0,0,0,0,0,0,0,0,1,1,0,1,1,1,0,0]^T \\
\mathbf{F}_{3,\sigma} &= [0,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0]^T \\
\mathbf{F}_{4,\sigma} &= [0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1]^T \\
\mathbf{F}_{5,\sigma} &= [0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1]^T
\end{aligned}
$$

Table 1 shows the number of coefficients in the vectors of function values, arithmetic expressions, FPARs, arithmetic expressions and FPARs arithmetic expressions of functions ordered by the autocorrelation functions.

For functions f_3 and f_5 we cannot get reduction in the number of coefficients count. For f_1 the reduction is small and for f and f_2 the reduction is considerable. For all functions, the minimum FPARs with LSF are smaller or at least equal to FPARs without LSF. For functions f and f_2 the reduction in FPARs by LSF is considerable.

As shown in [12], further reduction can be achieved if LSF is performed recursively by encoding equal subvectors in the reordered functions. We have found interesting to try this for functions f_1 to f_4 by performing LSF over the functions $f_{i,\sigma}$, $i = 1,2,3,4$. In this way, we get the reordered vectors of function values as

$$
\begin{aligned}
\mathbf{F}_{1,\sigma} &= [0,0,2,2,0,3,0,3,3,3,1,1,2,1,1,2]^T \\
\mathbf{F}_{2,\sigma} &= [0,0,0,0,0,0,0,0,1,1,1,1,0,1,0,0]^T \\
\mathbf{F}_{3,\sigma} &= [0,0,0,0,1,1,1,1,0,0,1,0,0,0,0,0]^T \\
\mathbf{F}_{4,\sigma} &= [0,0,0,0,1,1,1,1,0,0,0,0,1,0,0,0]^T
\end{aligned}
$$

This reordering reduces the number of non-zero coefficients in both ARs and FPARs from 11 to 9 for the function f_1. For other functions, f_2, f_3, and f_4, we cannot further reduce the number of coefficients.

6 EXPERIMENTAL RESULTS

In this section, we present a number of experimental results highlighting different aspect of the approach proposed. We used a program for calculation of autocorrelation functions and performing the Procedure for linearization of switching functions working with vector representations of switching functions. For this reason, the experiments are

Table 1. Number of coefficients in the truth-vector (SOP), arithmetic expressions (AR), Fixed-polarity arithmetic expressions (FPAR), arithmetic expression with LSF (AR-LSF), and Fixed-polarity arithmetic expressions with LSF (FPAR-LSF) for two times applied LSF.

f	In	Out	SOP	AR	FPAR	Polarity	AR-LSF	FPAR-LSF	Polarity
f	4	1	12	15	15	0000	4	4	0000
f_1	4	1	12	15	12	0011	11	11	0000
f_2	4	1	5	14	9	0111	4	3	0100
f_3	4	1	5	6	4	0001	6	4	0001
f_4	4	1	5	6	5	0101	3	3	0000
f_5	4	1	3	3	2	1001	3	2	0011
av.	4	1	7	9.8	7.8	-	5.1	4.5	-

restricted to functions of a small number of variables. However, if the calculation are performed over BDDs by using the DD-methods for computation of Walsh transform, and therefore, to implement the Wiener-Khinchin theorem, the number of variables corresponds to that which can be processed in other DD-methods, since the main complexity of the proposed method relates to the calculation of the total autocorrelation functions.

We performed experiments by always taking the smallest value for the assignment of input variables τ_f where the autocorrelation coefficient takes the maximum value B_f. Selection of other values for τ_f will be discussed in more details in what follows. In some cases, different selection of this parameter and the mapping σ for variables, may provide better solutions.

There are examples where reordering by autocorrelation coefficients increases the number of coefficients. We may assume that in these cases, the original ordering matches the best the structure of the arithmetic transform matrix. Such examples are in Table 2.

There are examples where arithmetic expressions require fewer coefficients than the number of non-zero function values, as for functions $clip$, $mul2$, $mul3$, $rd73$, and $rd84$. Another interesting example, which is not shown in this table since has 16 inputs, is the function $t481$, with 42016 non-zero values, for which AR and minimum polarity FPAR require 5329 non-zero coefficients.

Reordering reduces the number of coefficients in arithmetic expressions for functions $9sym$, $clip$, $rd53$, $sao2$, and $xor5$. For functions bw, $ex1010$, $t481$, we cannot reduce the number of coefficients. For other functions, reordering increases the number of non-zero coefficients. However, this increasing is quite smaller than the reduction achieved in the cases when it is possible.

In this table, we show the average number of coefficient to represent the functions considered. FPARs provide reduction of 37.29% compared to ARs. When we compare ARs and FPARs for functions reordered with LSF, we get the reduction for 44.02%. LSF reduces ARs for 8.4% and FPARs for 18.19%.

7 FEATURES OF THE METHOD

The method is sensitive to the choice of the values τ_f and the matrices σ determined depending on these values. It is the same as in the reduction of the number of nodes in decision diagrams by autocorrelation functions [12]. In this case, we are actually reducing the number of paths in the Arithmetic spectral transform decision diagrams.

The following example illustrates that the method depends on the choice of the value for τ_f.

Example 2 *(Dependency on τ_f and σ)*

For the function f_3 in 1, $B_{f,max}$ is achieved for $\tau_{f,max} = 1, 8$, and 9. For $\tau_{f,max} = 8$, we selected the matrix $\sigma_8 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$ and we performed reordering by using the inverse matrix $\sigma_8^{-1} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$. The resulting spectrum has 6 non-zero coefficients, the same as for the initial order of function values.

For $\tau_{f,max} = 9$, we selected two matrices $\sigma_{9,1} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$ and $\sigma_{9,2} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, and determined the corresponding inverse matrices as $\sigma_{9,1}^{-1} =$

Table 2. Number of coefficients in Arithmetic expressions, Fixed-polarity arithmetic expressions, and Arithmetic expression and Fixed-polarity arithmetic expressions for reordered benchmark functions.

f	In	Out	F	Arith	FPAR	Polarity	Arithr	FPARr	Polarityr
9sym	9		148	465	352	111100000	188	36	111111100
bw	5		22	32	22	10111	32	22	10110
clip	9		439	264	255	100001000	248	241	100001000
con1	7		20	21	18	1000000	21	18	1000000
ex1010	10		800	1023	1011	0010010010	1023	1008	0000010110
misex1	8		13	60	20	11111000	80	27	10110001
mul2	4	5	9	4	4	0000	6	6	0000
mul3	6	7	49	9	9	000000	12	12	000000
rd53	5	3	31	31	31	00000	25	21	00010
rd73	7	3	127	71	71	0000000	80	65	0000010
rd84	8	4	255	171	171	00000000	175	116	11111110
sao2	10		118	1022	100	0010110011	1010	89	0000110111
sqrt8	8		52	210	37	11011111	212	64	10111101
squar5	5		14	14	00000	14	17	17	00000
xor5	5	1	16	31	31	00000	15	15	00000
av.	7		140	228	143	-	209	117	-

$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$ *and* $\sigma_{9,2}^{-1} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$. *The resulting spectra for* $\sigma_{9,1}$ *and* $\sigma_{9,2}$ *have 5 and 6 non-zero coefficients.*

Table 3 summarizes the discussion in this example.

Example 3 *(Dependency on* τ_f*)*

The function $misex1$ *has the maximum value for* $B_f = 240$ *at the points* $\tau_{f,max} = 2, 4, 8$*. After reordering, the arithmetic spectrum consists of 58, 60 and 80 non-zero coefficients, respectively. The arithmetic spectrum for* $misex1$ *for the initial ordering has 60 non-zero coefficients. It follows, that the method proposed can reduce the spectrum for two coefficients if* $\tau_{f,max} = 2$ *is selected.*

However, there are functions where the size of the spectrum in the number of non-zero coefficients count does not depend on the selection of $\tau_{f,max}$*. For example, the function* $ex1010$ *has* $B_{f,max}$ *for* $\tau_{f,max} = 7$*, and* 10*. In both cases, the spectrum has 1023 non-zero coefficients, which is the same size as for the initial order of function values. However, minimum FAPR for the initial ordering has 1011 coefficients for the polarity 0010010010, and after reordering the minimum FPAR has 1008 coefficients for the polarity 0000010110. Thus, the proposed method reduces the FPAR for three coefficients.*

8 CLOSING REMARKS

Arithmetic expressions has been used for long in switching theory and logic design. Ever increasing complexity of circuits and systems in these areas, demands for the optimization in terms of space and time, power consumption, etc., and search for alternative solutions bring some new interest to this transform. For the same reason of complexity, it is interesting to provide for given switching functions arithmetic expressions as compact as possible in the number of non-zero coefficients count.

This paper proposes to combine classical approaches to the optimization of arithmetic expressions by choosing polarity of variables with the linearization of switching functions (LSF).

Experimental results presented in the paper, show that in classical approaches, transition from Positive-polarity arithmetic expressions to Fixed-polarity arithmetic expressions may provide for 30-40% of reduced number of coefficients. However, when combined with LSF another 20% of reduction in the non-zero coefficients count can be achieved.

LSF can be performed by an algorithm, which although deterministic, allows freedom in choosing parameters for reordering of function values. Due to this feature, when interest, further reduction of arithmetic expressions for particular given functions can be achieved by selecting appropriately the corresponding parameters.

Table 3. Function f_3 and the arithmetic spectra.

f	Spectrum	# r	Polarity
f_3	$[0,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0]^T$		
$f_{3,8}$	$[0,0,0,0,0,1,0,0,1,1,1,1,0,0,0,0]^T$		
$f_{3,9,1}$	$[0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0]^T$		
$f_{3,9,2}$	$[0,0,0,0,0,0,1,0,1,1,1,1,0,0,0,0]^T$		
S_{f_3}	$[0,0,0,0,1,0,-1,0,0,0,1,-1,0,0,-1,1]^T$	6	(0000)
$S_{f_3,min}$	$[0,0,0,0,1,0,-1,0,0,0,0,1,0,0,0,-1]^T$	4	(0001)
$S_{f_{3,8}}$	$[0,0,0,0,0,1,0,-1,1,0,0,0,-1,-1,0,1]^T$	6	(0000)
$S_{f_{3,8},min}$	$[0,0,0,0,0,0,0,1,1,0,0,0,-1,0,0,-1]^T$	4	(0010)
$S_{f_{3,9,1}}$	$[0,0,0,0,0,0,0,1,1,0,0,0,-1,0,0,-1]^T$	4	(0000)
$S_{f_{3,9,1},min}$	$[0,0,0,0,0,0,0,1,1,0,0,0,-1,0,0,-1]^T$	4	(0000)
$S_{f_{3,9,2}}$	$[0,0,0,0,0,0,1,-1,1,0,0,0,-1,0,-1,1]^T$	6	(0000)
$S_{f_{3,9,2},min}$	$[0,0,0,0,0,0,0,1,1,0,0,0,-1,0,0,-1]^T$	4	(0001)

References

[1] S. Agaian, J.T. Astola, K. Egiazarian, *Binary Polynomial Transforms and Nonlinear Digital Filtering*, Marcel Dekker, 1995.

[2] H. H. Aiken, *The Annals of the Computation Laboratory of Harvard University, Volume XXVII, Synthesis of Electronic Computing and Control Circuits*, Cambridge, MA, Harvard Univ., 1951.

[3] Ph.W. Besslich, “Efficient computer method for XOR logic design”, *IEE Proc.*, Part E, Vol. 129, 1982, 15-20.

[4] E.M. Clarke, K.L. Mc Millan, X. Zhao, M. Fujita, “Spectral transforms for extremely large Boolean functions”, in Kebschull, U., Schubert, E., Rosenstiel, W., Eds., *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expression in Circuit Design*, Hamburg, Germany, September 16-17, 1993, 86-90. Workshop Reed-Muller’93, 86-90.

[5] D. Debnath, T. Sasao, “Exact minimization of FPRMs for incompletely specified functions”, *Proc. 4th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, August 20-12, 1999, Victoria, B.C., Canada, 253-264.

[6] B.J. Falkowski, C.H. Chang, “Properties and methods of calculation generalized arithmetic and adding transforms”, *IEE Proc., Circuits, Devices, Syst.*, Vol. 144, No. 5, 1997, 249-258.

[7] M. Fujita, J. C.-H. Yang, E.M. Clarke, X. Zhao, M.C. Geer, ”Fast spectrum computation for logic functions using binary decision diagrams”, *ISCAS-94*, 1994, 275-278.

[8] J.E. Gibbs, “Walsh spectrometry a form of spectral analysis well suited to binary digital computation”, *NPL DES Repts.*, National Physical Lab., Teddington, Middlesex, England, 1967.

[9] J.E. Gibbs, “Local and global views of differentiation”, in: Butzer, P.L., Stanković, R.S., Eds., *Theory and Applications of Gibbs Derivatives and Applications*, Matematički institut, Beograd, 1990, 1-19.

[10] M.G. Karpovsky, *Finite Orthogonal Series in the Design of Digital Devices*, Wiley and JUP, New York and Jerusalem, 1976.

[11] M.G. Karpovsky, E.S. Moskalev, ”Utilization of autocorrelation characteristics for the realization of systems of logical functions”, *Avtomatika i Telemekhanika*, No. 2, 1970, 83-90, English translation *Automatic and Remote Control*, Vol. 31, 1970, 342-350.

[12] Karpovsky, M.G., Stanković, R.S., Astola, J.T., ”Reduction of sizes of decision diagrams by autocorrelation functions”, *IEEE Trans. on Computers*, Vol. 52, No. 5, 2003, 592-606.

[13] R. Krenz, E. Dubrova, A. Kuhelmann, ”Circuit-based evaluation of the arithmetic transform of Boolean functions”, *Int. Workshop on Logic Synthesis, (IWLS 2002)*, New Orleans, Louisiana, USA, June 4-7, 2002, 321-326.

[14] Y. Komamiya, “Theory of relay networks for the transformation between the decimal and binary system”, *Bull. of E.T.L.*, Vol. 15, No. 8, August 1951, 188-197.

5July 10, 1959, pages 40.

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

[16] C. Moraga, (ed.), “Advances in Spectral Techniques”, Berichte zur angewandten Informatik, Universität Dortmund, 1998.2, ISSN 0946-2341.

[17] C. Moraga, and Qi-Shan Zhang, (ed.), *Proc. 5th Int. Workshop on Spectral Techniques*, Beijing University of Aeronautics and Astronautics, Beijing, P.R. China, March 15-17, 1994,

[18] C. Moraga, T. Sasao, R.S. Stanković, ”A generalized approach to edge-valued decision diagrams for arithmetic transforms”, *Avtomatika i Telemekhanika*, No. 1, 2002, 140-153, (in Russian).

[19] E.I. Nechiporuk, “On the synthesis of networks using linear transformations of variables”, *Dokl. AN SSSR*. Vol. 123, No. 4, December 1958, 610-612.

[20] S. Rahardja, B.J. Falkowski, ”Application of linearly independent arithmetic transform in testing of digital circuits”, *Electronics Letters*, Vol. 35, No. 5, 1999, 363 -364.

[21] T. Sasao, “Representations of logic functions by using EXOR operators,” in: [26], 29-54.

[22] T. Sasao, *Switching Theory for Logic Synthesis*, Kluwer Academic Publishers, 1999.

[23] T. Sasao, "Arithmetic ternary decision diagrams and their applications", *Proc. 4th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, August 20-12, 1999. Victoria, B.C., Canada, 149-155.

[24] T. Sasao, Ph.W. Besslich, "On the complexity of mod 2 PLA's", *IEEE Trans. Comput.*, Vol. C-39, No. 2, 1991, 262-266.

[25] T. Sasao, F. Izuhara, "Exact minimization of FPRMs using multi-terminal EXOR-TDDs", in [26], 191-210.

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

[27] *Synthesis of electronic calculation and control networks*, translation form English under supervision by B.I. Shestakov, Izv. Inostranoi Literaturi, Moscow, 1954.

[28] V.P. Shmerko, "Synthesis of arithmetical forms of Boolean functions through the Fourier transforms", *Automatics and Telemechanics*, No.5, 1989, 134-142.

[29] A.K., Singh, R. Mishra, A. Mohan, "Synthesis and fault detection of combinatorial networks usign arithmetic spectrum", *Proc. 4-th Int. Conf. on Information Technology*, Gopalpur-on-Sea, India, December 20-23, 2001.

[30] R.S. Stanković, "A note on the relation between Reed-Muller expansions and Walsh transform", *IEEE Transactions on Electromagnetic Compatibility*, Vol. EMC-24, No. 1, 1982, 68-70.

[31] R.S. Stanković, "Some remarks about spectral transform interpretation of MTBDDs and EVBDDs," *ASP-DAC'95*, Makuhari Messe, Chiba, Japan, August 29-September 1, 1995, 385-390.

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

[33] R.S. Stanković, J.T. Astola, *Spectral Interpreatation of Decision Diagremas*, Springer, 2003.

[34] R.S. Stankovicć, C. Moraga, J.T. Astola, "From Fourier expansions to arithmetic-Haar expressions on quaternion groups", *Applicable Algebra in Engineering, Communication and Computing*, Vol. AAECC 12, 2001, 227-253.

[35] R.S. Stanković, T. Sasao, "A discussion on the history of research in arithmetic and Reed-Muller expressions", *IEEE Trans. on CAD*, Vol. 20, No. 9, 2001, 1177-1179.

[36] Stankovi c, R.S., Sasao, T., Astola, J.T., *Publications in the First Twenty Years of Switching Theory and Logic Design*, TICSP Series #14, ISBN 952-15-0679-2, Tampere, 2001.

[37] R.S. Stanković, T. Sasao, C. Moraga, "Spectral transform decision diagrams", in [26], 55-92.

[38] Ž. Tošić, "Arithmetical representations of linear functions", in *Discrete Automata and Logical Networks*, Nauka, Moscow, 1970, 131-136, (in Russian).

[39] S.N. Yanushkevich, *Logic Differential Calculus in Multi-Valued Logic Design*, Techn. University of Szczecin Academic Publisher, Poland, 1998.

ADDENDUM

Autocorrelation Function

For a given n-variable switching function f, the autocorrelation function B_f is defined as

$$B_f(\tau) = \sum_{x=0}^{2^n-1} f(x)f(x \oplus \tau), \quad \tau \in \{0, \ldots, 2^n - 1\},$$

The Winer-Khinchin theorem [10] states a relationship between the autocorrelation function and Walsh (Fourier) coefficients

$$B_f = 2^n W^{-1}(Wf)^2.$$

Total autocorrelation function

For a system of k switching functions $f^{(i)}(x_1, \ldots, x_n)$, $i = 0, \ldots, k-1$, the total autocorrelation function is defined as the sum of autocorrelation functions of each function in the system. Thus,

$$B_f(\tau) = \sum_{i=0}^{k-1} B_{f^{(i)}}(\tau).$$

Note that for any $\tau \neq 0$, $B_f(\tau) \leq B_f(0)$. The set $G_I(f)$ of all values for τ such that $B_f(\tau) = B_f(0) = \sum_{i=0}^{k-1}\sum_{x=0}^{2^m-1} f^{(i)}(x)$ is a group with respect to the EXOR as the group operation which is denoted as the inertia group of the system f.

LSF Procedure

The following procedure provides for a solution of the linearization problem.

Linearization procedure

1. Construct by the Wiener-Khinchin theorem and Fast Walsh Hadamard Transform (FWHT) the autocorrelation function $B_f(\tau) = \sum_x f(x)f(x \oplus \tau)$,
2. Find τ_0 such that $B(\tau_0) = max_{\tau \neq 0} B(\tau)$.
3. Find τ_i, $i = 1, \ldots, n-1$, such that $B(\tau_i) = max_{\tau \notin T_i} B(\tau)$, where $T_i = \{c_0\tau_0 \oplus c_1\tau_1 \oplus \cdots \oplus c_{i-1}\tau_{i-1}\}$, $c_i \in \{0, 1\}$.
4. Construct $\mathbf{T} = \left[\ \tau_0, \tau_1, \cdot, \tau_{n-1}\ \right]^T$, and determine $\sigma = \mathbf{T}^{-1}$, where all the calculations are in $GF(2)$.

Complexity of solving the linearization problem for a given f in terms of the arithmetic operations does not exceed $O(n2^n)$ and may be much smaller than this if we have a compact description of f [10].