Proceedings of the 5th WSEAS Int. Conf. on Electronics, Hardware, Wireless and Optical Communications, Madrid, Spain, February 15-17, 2006 (pp176-181)

# Low complexity Linear Decomposition at the Disjoint Cubes Domain

OSNAT KEREN
School of Engineering
Bar Ilan University, Israel
E-mail: kereno@eng.biu.ac.il

*Abstract :-* Linear decomposition proved to be efficient technique for reducing the number of terms and literal of two-level implementation of logic functions. The decomposed system consists of linear function followed by nonlinear function of minimal realization cost. The complexity of determining the linear part makes this approach inapplicable for systems of large number of inputs when represented by truth table or decision diagrams. This paper presents a method for linearization of systems represented as a set of disjoint cubes. The reduction in complexity is achieved by a) calculation of the autocorrelation function on the disjoint cubes domain; b) representing the linear part as a superposition of linear transforms of a special form. Experimental benchmark results allow comparing the proposed technique with known linearization methods, and show the high efficiency of the proposed approach.

*Key-Words :-* Logic design, Linear decomposition, Spectral technique, Disjoint cubes, Autocorrelation, Complexity, K-procedure, Walsh transform, Wiener-Khinchin

## 1 Introduction

The linear decomposition is a well known technique that allows implementation of a logical function $f : GF(2^n) \rightarrow GF(2^k)$ as a superposition of a linear transform function $\sigma$ followed by a non-linear part, $f_\sigma$ ( figure 1). The linear part is implemented by XOR gates and $f_\sigma$ is typically implemented as two-level (or multi level) with a complete set of gates, e.g. AND-OR-NOT. For systems of large number of inputs the linear part has negligible effect on the synthesized circuit size, while the number of terms and literals of linearized function $f_\sigma$ can be orders of magnitude smaller than the original system. The optimization problem is to find a linear transform $\sigma$ such that the corresponding $f_\sigma$ has low realization cost.

The cost measure for linearization, denoted by $\mu(f)$, is the number minterm-pairs of Hamming distance one for which the function outputs are equal. $\mu$ equals to the sum of the autocorrelation function $R(\tau)$ at $\tau = 2^i, i = 0, 1, \ldots n-1$. The linearization problem as defined by Karpovsky [5] is to determine a set of independent $\tau$'s for which the sum autocorrelation values is maximal, this set defines $\sigma$.

The complexity of linearization procedures lies in the calculation of the autocorrelation function and in constructing a set of independent $\tau$'s; For systems of large number of inputs it is impossible to calculate $R$ at the truth table domain nor over decision diagrams. In these cases, calculations over disjoint cubes have to be used since it allows processing separately each cube or cube pair.

$R$ can be calculated directly according to its definition, or using the Wiener-Khinchin theorem. Extensive work has been done in spectra calculations of switching functions defined by disjoint cubes [2, 3]. However, employing this approach for calculation of $R$ using the Wiener-Khinchin theorem may be inefficient. Another method for calculation of $R$ on the disjoint cubes is a tabular technique introduced by Almaini et al. [1]. A spectral interpretation of tabular techniques for fixed-polarity Reed-Muller expressions in terms of the correlation on finite dyadic groups was proposed in [6]. However, this approach works on minterms rather

then on cubes, and thus the complexity depends on the number of minterms the cubes cover.

A linearizarion algorithm for efficient minimization of logic functions on the disjoint cubes domain was suggested by Varma and Trachtenberg [7]. The authors calculate $R(\tau)$ directly, according to its definition. This method may not produce the maximal $\mu$ since the final set of $\tau$'s depends on the order of processing the cubes and on the subspace defined by $\tau$'s of previous produced cubes. Moreover, the transformed function $f_\sigma$ is obtained from $\sigma$ by applying the linear transform on the cubes. When the cubes are not minterms the linear transform may break a cube into a number of cubes of smaller order; obtaining these smaller cubes is not straightforward.

Recently an algorithm for linearization of decision diagrams by using the autocorrelation function was proposed by Karpovsky et al [4] and known as K-procedure. The procedure reduces the average size of a Binary Decision Diagram (BDD) by applying a linear transformation $\sigma$ on the input variables. The algorithm solves the problem of determining an independent set of $\tau$'s by performing a linear transform and folding the function after each step. However, it may be impractical for functions of many input variables.

In this paper, we suggest an algorithm for linear decomposition of a multi-output system at the disjoint cubes domain. The proposed algorithm calculates the autocorrelation function for several $\tau$'s simultaneously by defining the the subspace determined by the nonzero values of cross correlation of two cubes. To reduce complexity the Hamming weight of $\tau$ is restricted. In practice, $\tau$'s of the Hamming weight up to three give the same results as without the restriction. By defining $\sigma$ as a product of simple matrices, it is possible to calculate $f_\sigma$ by steps and to simplify the construction of the set of $\tau$'s. The complexity of the algorithm is polynomial in the number of inputs and number of cubes, $(\mathcal{O}(n^4N^2))$, where $N$ is the number of products. Therefore the new method is efficient of systems for large number of inputs and moderate number of cubes, $N^2 < 2^n/n^2$.

The paper is organized as follows. Section 2 contains mathematical background. The linearization procedure at the disjoint cubes domain is presented in section 3. Section 4 describes in details the calculation of the autocorrelation. In section 5 show that the effect of restricting the Hamming weight of $\tau$ is negligible, then we compare the performance of K-procedure at the truth-table domain to the new linearization method in terms of the cost function $\mu$ and execution time, and show the significant improvement in execution time. The conclusions summarizing the results are presented in Section 6.

## 2 Mathematical background

### 2.1 Definitions

Let $f : GF(2)^n \rightarrow GF(2^k)$ a system of $k$ logic functions of $n$ variables or Multioutput Logic Function. Let $\mathcal{G} = \{0, 1, \phi\}$ , where $\phi$ stands for don't-care. The representation of $f$ at the cubes domain is a set of $N$ pairs

$$F = \{(P_i, Y_i)\}_{i=1}^{N}$$

where $P_i \in \mathcal{G}^n$, is a product and $Y_i \in GF(2^k)$ is the corresponding output.
Two cubes are called disjoint if they do not have any minterm in common. If any pair of cubes is disjoint the function is said to be of a disjoint cubes representation . Clearly any non-disjoint set can be expanded into a disjoint set, and therefore without loss of generality, we assume that the system consists of $N$ disjoint (orthogonal) products.
The products of a multi-output logical function can be partitioned into sets having identical output pattern, called characteristic sets. The characteristic set, $F_u$, $(u \in GF(2^k),)$ is the set

$$F_u = \{(P_i, Y_i) | (P_i, Y_i) \in F, Y_i = u\} \qquad (1)$$

The switching function defined by the characteristic set $F_u$ is the characteristic function $f_u(x)$.

### 2.2 Optimization criterion

In this paper we measure the realization complexity of a system by $\mu(f)$. $\mu(f)$ counts the number of adjacent minterms having the same output,

As shown in [5], the complexity measure $\mu(f)$ can be written in terms of the autocorrelation function values at $\tau$'s of Hamming weight one; Denote by $R_g(\tau)$ the autocorrelation function of a binary function $g$,

$R_g(\tau) = \sum_{x \in GF(2^n)} g(x)g(x+\tau)$, then

$$\mu(f_u) = \sum_{||\tau||=1} R_u(\tau)$$

where $||\tau||$ is the Hamming weight of $\tau$ and $\mu(f) = \sum_{u \in GF(2^k)} \mu(f_u)$

### 2.3 Linear decomposition

The linear decomposition of a function, presented in [5], allows implementation of $f$ as a superposition of a linear transform function $\sigma$ implemented by XOR gates followed by a non-linear part, $f_\sigma$,

$$f(x) = f_\sigma(\sigma x),$$

of minimal implementation complexity as sum of products (see figure 1).

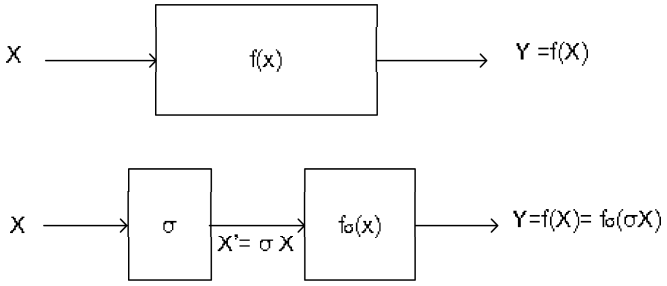


Figure 1: Linear Decomposition

The optimization problem is to find for a given function $f$, a nonsingular $(n \times n)$ linearization matrix $\sigma$, such that $\mu(f_\sigma)$ is maximal.

The autocorrelation functions of $f(x)$ and $f_\sigma(x)$ carry the same values but in a different positions, i.e. $R_\sigma(\tau) = R(\sigma^{-1}\tau)$. Therefore, the minimization problem is to determine a nonsingular matrix $\sigma = T^{-1}$, $T = (\tau_{n-1}, \ldots \tau_1, \tau_0)$, such that $\mu(f_\sigma) = \sum_i R(\tau_i)$ is maximal.

## 3 Linearization Algorithm

The complexity of linearization algorithms lies in the calculation of the autocorrelation function and the construction of the set $\{\tau_i\}_{i=0}^{n-1}$.

The k-procedure presented in [4] provides a deterministic greedy algorithm for BDD minimization of the number of nodes at each level. Karpovsky et al. addressed the complexity of the constructing the set of

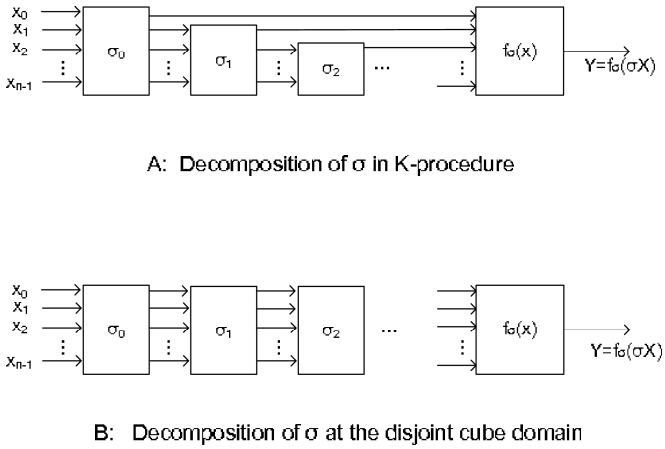


Figure 2: Decomposition of $\sigma$

$\tau_i$ by folding the binary tree and by constructing $\sigma$ in steps. This can be referred as a superposition of matrices $\sigma_i$ of decreasing degree, figure 2-A. Folding ensures that the $\tau_i$ vector derived at step $i$ is independent in previous $\tau$'s and therefore the overall linear transform matrix $\sigma$ is nonsingular.

This approach is not applicable at the disjoint cubes domain since folding be done only by expending the cubes into minterms. For cubes of high order it is not practical. Instead, it is possible to represent the linear transform matrix $\sigma$ as a product of $n'$ matrices $n' \leq n$,

$$\sigma = \sigma_{n'-1} \cdots \sigma_1 \sigma_0,$$

as shown in figure 2-B. The matrix $\sigma_i$ is called an **instantaneous** linearization matrix of step $i$. Equivalently, $T = T_0 T_1 \cdots T_{n'-1}$, where $\sigma_i T_i = I$. The matrix $T_i = (t_{n-1}, \ldots, t_1, t_0)$ carries on its $i$'th column the $\tau$ vector derived at step $i$, of decimal value $\geq 2^i$. The right columns of $T_i$ correspond to $\tau$'s determined in previous steps, therefore, the decimal value of $t_j$, $j < i$ is $2^j$. The remaining columns $t_j$, $j > i$ are of Hamming weight one and are determined such that $T_i$ is nonsingular.

Decomposition of $\sigma$ into product of matrices of this special form is essential for two reasons; To simplify the search for the next $\tau$ by an instantaneous linearization of the set of products and the simplicity of calculation of the linearized set of cubes.

To reduce complexity of the calculation of the autocorrelation function, it is calculated only for $\tau$'s of Hamming weight less or equal $w$. In practice (see section 5), $\tau$'s of the Hamming weight up to three give the

same results as without the Hamming weight restriction. Thus about $n^3$ values of $R$ are to be considered for efficient linearization.

The following linearization procedure at the disjoint cubes domain results the linear transform matrix $\sigma$ and a set of linearized disjoint cubes representing $f_\sigma$.

**Disjoint cube Linearization procedure**
Set $\sigma = I_{(n\times n)}$
Set $i = 0$
While $i \leq n-1$

1) For all $\tau \in GF(2^n)$ , $||\tau|| \leq w$ and $\tau \geq 2^{i-1}$. calculate the autocorrelation $R(\tau)$ as defined in section 4.
2) If $R(\tau) = 0$ for all calculated $\tau$'s then break.
3) Determine $\tau$ that maximizes $R(\tau)$
4) Construct the matrix $\sigma_i$
5) Perform an instantaneous linear transform on the set of products
6) Update $\sigma, \sigma = \sigma_i\sigma$
7) Increment $i$
□

## 4 calculation of the autocorrelation function on disjoint cube domain

Let $N_u$ the number of products $\{P_i\}_{i=1}^{N_u}$ associated with the Characteristic Set $F_u$ as defined by (1), $\sum_{u\in GF(2^k)} N_u = N$. Since $F$ is an orthogonal set of products, so does $N_u$ and thus $f_u(x) = \sum_{i=1}^{N_u} P_i(x)$ and

$$R_u(\tau) = \sum_{i=0}^{N_u}\sum_{j=0}^{N_u}\sum_{x\in GF(2^n)} P_i(x)P_j(x+\tau) = \sum_{i=0}^{N_u}\sum_{j=0}^{N_u} R_{i,j}^{(u)}(\tau)$$

To simplify the notations, when it is clear form the context we omit the $u$, i.e. instead of $R_{i,j}^{(u)}$ we write $R_{i,j}$.

There is no need to go over all the $x$'s for calculating the correlation function $R_{i,j}(\tau)$, it can be computed directly by comparing the products as shown below;

For a product $P_i = (p_{n-1}^{(i))}, \ldots, p_1^{(i))}p_0^{(i))}) \in \mathcal{G}^n$ denote by $n_\phi$ the number of symbols of $P_i$ that carry don't care. It is well known (see e.g. [5]) that for any $\tau$ of the form $(\tau_{n-1}, \ldots, \tau_1, \tau_0)$, where

$$\tau_k = \begin{cases} \phi & p_k^{(i))} = \phi \\ 0 & otherwise \end{cases}$$

$(k = 1, 2, \ldots n-1)$ the autocorrelation $R_{i,i}(\tau)$ of $P_i(x)$ equals $2^{n_\phi}$ and is zero elsewhere.

For two products $P_i$ and $P_j \in \mathcal{G}^n$ denote by $p_k^{(i)}$ and $p_k^{(j)}$ the $k$'th symbol of $P_i$ and $P_j$ respectively. There are nine possible $(p_k^{(i)}, p_k^{(j)})$ pair types, i.e.

$$(p_k^{(i)}, p_k^{(j)}) \in \left\{ \begin{array}{l} T_1 = (0,0), T_2 = (0,1), T_3 = (0,\phi), \\ T_4 = (1,0), T_5 = (1,1), T_6 = (1,\phi), \\ T_7 = (\phi,0), T_8 = (\phi,1), T_9 = (\phi,\phi) \end{array} \right\}.$$

Let $n_1, n_2, \ldots n_9$ stand for the number of pairs of each type,

$$n_l = |\{k|(p_k^{(i)}, p_k^{(j)}) = T_l\}| \;\; l = 1, 2 \ldots 9.$$

For example, if $P_i = (0\phi 11\phi\phi)$ and $P_j = (00\phi 1\phi\phi)$ then $n_1$ - the number of times the pair $(0,0)$ appears is one. $n_2 = 0$ since there is no position where $p_k^{(i)} = 0$ and $p_k^{(i)} = 1$. Similarly, $n_3 = 0, n_4 = 0, n_5 = 1, n_6 = 0, n_7 = 1, n_8 = 0$ and $n_9 = 2$.

**Theorem 1** *Let $P_i = (p_{n-1}^{(i))}, \ldots, p_1^{(i))}p_0^{(i))})$ and $P_j = (p_{n-1}^{(j))}, \ldots, p_1^{(j))}p_0^{(j))}) \in \mathcal{G}^n$. Denote by $n_\phi$ the number of pairs $(p_k^{(i)}, p_k^{(j)})$ of type $T_9$. For any $\tau$ of the form $(\tau_{n-1}, \ldots, \tau_1, \tau_0)$, where*

$$\tau_k = \begin{cases} 0 & (p_k^{(i)}, p_k^{(j)}) \in \{T_1, T_5\} \\ 1 & (p_k^{(i)}, p_k^{(j)}) \in \{T_2, T_4\} \\ \phi & otherwise \end{cases}$$

*$(k = 1, 2, \ldots n-1,)$ the cross-correlation $R_{i,j}(\tau)$ of $P_i(x)$ and $P_j(x)$ equals $2^{n_\phi}$ and is zero elsewhere.*

**Example 1** *Consider the following four products*

$$\begin{array}{lllllll} P_1 & = & (0,0,\phi,\phi) & \qquad & P_3 & = & (1,0,\phi,0) \\ P_2 & = & (0,1,1,\phi) & & P_4 & = & (1,1,1,1) \end{array}$$

*Consider $P_3$ and $P_4$. The corresponding pair types are $(T_5, T_2, T_8, T_2)$ and the $\tau$ pattern for which the cross-correlation $R_{3,4}(\tau)$ is not zero is $(0,1,\phi,1)$. The value of $R_{3,4}$ at these $\tau$'s is $2^{n_9} = 2^0 = 1$.*

*The table below shows the $\tau$'s for which $R_{i,j}$ is not zero,*

| $\tau_{i,j}$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | $(0,0,\phi,\phi)$ | | | |
| 2 | $(0,1,\phi,\phi)$ | $(0,0,0,\phi)$ | | |
| 3 | $(1,0,\phi,\phi)$ | $(1,1,\phi,\phi)$ | $(0,0,\phi,0)$ | |
| 4 | $(1,1,\phi,\phi)$ | $(1,0,0,\phi)$ | $(0,1,\phi,1)$ | $(0,0,0,0)$ |

*Clearly each $\tau$ may appear only once at each column and each row. Therefore $R(\tau)$ is the sum of at most four $R_{i,j}$'s.*

*The value of the non-zero elements of $R_{i,j}(\tau)$ is*

| $R_{i,j}$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 4 | | | |
| 2 | 2 | 2 | | |
| 3 | 2 | 1 | 2 | |
| 4 | 1 | 1 | 1 | 1 |

*For example $R(2) = R(0010)$, the pairs that their correlation is not zero for the pattern 0010 are $(i,j) = (1,1),(3,3)$, and hence $R(2) = R_{1,1}+R_{3,3} = 4+2 = 6$.*

## 5 Experimental Results

In this section we provide simulation results on several benchmarks. The performance of the suggested linearization algorithm is examined in terms of the cost function and run time. The performance is compared to the original function and to the linearized function after applying k-procedure at the truth-table domain.

Table 1 shows that restricting the Hamming weight of $\tau$ to be less or equal to $w$, doesn't degrade the performance significantly. This justifies the use of $w = 3$ when comparing the performance in terms of $\mu$ and run time to the K-procedure.

| benchmark | $n$ | $k$ | $w=1$ | 2 | 3 | 5 | 7 |
|---|---|---|---|---|---|---|---|
| sqrt8.pla | 8 | 4 | 1164 | 1284 | 1268 | 1286 | 1286 |
| radd.pla | 8 | 5 | 824 | 1304 | 1304 | 1304 | 1304 |
| root.pla | 8 | 5 | 868 | 932 | 940 | 958 | 958 |

Table 1: Cost function $\mu$ versus the restriction $w$ on the Hamming weight of $\tau$. The $\mu$ of the original function equals to the $\mu$ calculated with $w = 1$.

Table 2 shows the cost functions for standard benchmarks. The value of the original function is denoted by $\mu$. $\mu_{upb}$ is an upper bound on cost function, it is defined as the sum of the $n$ maximal values of the autocorrelation values. This bound is not always achievable. The costs $\mu_{k-proc}$ and $\mu_{DCD}$ were obtained by the k-procedure and the new method with $w = 3$, respectively.

| Benchmark | $n$ | $k$ | $\mu$ | $\mu_{k-proc}$ | $\mu_{DCD}$ | $\mu_{up}$ |
|---|---|---|---|---|---|---|
| radd | 8 | 5 | 1088 | 1494 | 1524 | 1914 |
| root | 8 | 5 | 1224 | 1224 | 1308 | 1904 |
| mlp4 | 8 | 8 | 616 | 616 | 642 | 958 |
| f51m | 8 | 8 | 858 | 1050 | 1210 | 1568 |
| adr4 | 8 | 5 | 1228 | 1328 | 1390 | 1718 |
| dc2 | 8 | 7 | 1066 | 1066 | 1150 | 1568 |
| clip | 9 | 5 | 2490 | 3092 | 3208 | 3816 |

Table 2: $\mu$ of the original and linearized benchmark functions and upper bound on $\mu$

Figure 3 shows the execution time of the K-procedure a the truth-table domain and the proposed method versus the number of inputs. The execution time was measured on Intel-Centrino, 1.2Ghz, 0.99GB RAM, for random PLA's of four outputs and 50 products. The variance of the measurements was less than 3%. It is clear that linearization at the disjoint cubes domain outperforms linearization based on Wiener-Khinchin theorem.

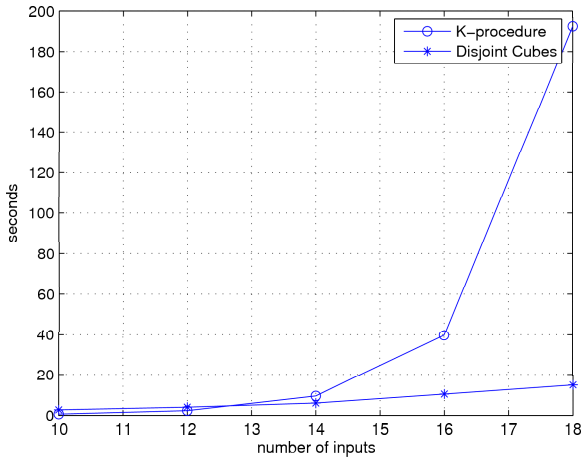


Figure 3: Execution time versus number of inputs of K-procedure and Disjoint cubes linearization alg. for random PLA of 4 outpus and 50 products.

Figure 4 shows the execution time of the linearization procedure with $w = 3$ for random PLAs of 10 to 50 inputs and four outputs with 25,50 and 100 prod-

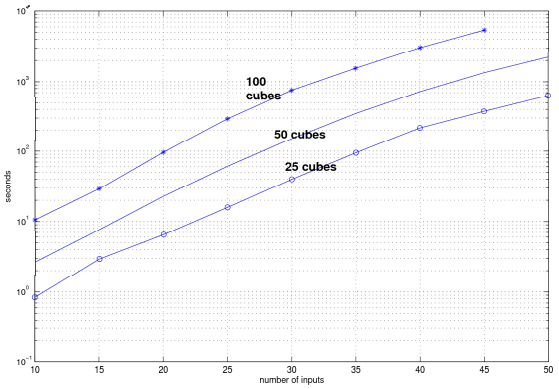


Figure 4: Execution time of Disjoint cubes linearization procedure versus the number of input variables for random PLA of 4 outputs and 25,50 and 100 products.

ucts. As expected the complexity is polynomial with the number of cubes ($N^2$), and from the slop of the curves it is clear that the complexity is increasing as $n^4$ with the number of inputs and not exponentially ($n^2 2^n$) as the complexity of the calculating $n$ times the autocorrelation function by the Wiener-Khinchin theorem.

## 6 Conclusion

Linear decomposition proved to be an effective tool for reduction of the realization cost of a system. The present work may be considered as a natural continuation of the [7] and [4] for functions of large number of inputs represented in disjoint cubes form. The main contribution of the paper can be summarized as follows:

1. It proposes a method for calculating the autocorrelation function for a logic function defined by its disjoint sum of products.

2. It describes a technique for simplifying the construction of the set of $n$ independent vectors of high correlation that define $\sigma$, by representing $\sigma$ as a product of matrices of a special form.

The proposed technique is checked by using a set of standard benchmarks. The experimental results clearly demonstrate efficiency of the proposed techniques.

## References

[1] Almaini, A. E. A., Thomson, P., and Hanson, D., Tabular techniques for Reed-Muller logic. *International Journal of Electronics*, 70,, 1991 pp.23-34.

[2] Falkowski, B.J., Kannurao, S., Circuits and Systems, *ISCAS 2001*. Volume 5, 6-9 May 2001, 61 - 64, Digital Object Identifier 10.1109/ISCAS.2001.921985

[3] Falkowski, B.J., Schafer, I., Perkowski, M.A., Calculation of the Rademacher-Walsh spectrum from a reduced representation of Boolean functions, *Design Automation Conference, 1992. EURO-VHDL '92, EURO-DAC '92.* 7-10 Sept. 1992, pp.181 - 186,Digital Object Identifier 10.1109/EURDAC.1992.246245.

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

[5] M.G. Karpovsky, *Finite orthogonal series in the design of digital devices*, New York, Wiley, 1976.

[6] Stankovic, R.S., and Falkovski, B.J., Spectral Interpretation of Fast Tabular Technique for Fixed-Polarity Reed-Muller Expressions, *International Journal of Electronics*, United Kingdom, vol. 87, no. 6, June 2000, pp. 641-648.

[7] Varma, D. Trachtenberg, E.A., Design automation tools for efficient implementation of logic functions by decomposition; *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Volume 8, Issue 8, Aug. 1989, pp.901 - 916 .

[8] R. Stankovic, and M.G. Karpovsky, Remarks on Calculation of Autocorrelation on Finite Dyadic Groups by Local Transformations of Decision Diagrams, *Lecture Notes Springer and Verlag*, 2005.